UNIVERSITY OF WEST ATTICA

&

UNIVERSITY OF LIMOGES

Master's Thesis

Low Light Image Enhancement: Classical and Deep Learning methods using the RELLISUR Dataset

Student: Panagiotis Koutsaftis

(aivc21010)

Supervisor: Anastasios L. Kesidis, Professor

Athens, September 2023

Low Light Image Enhancement: Classical and Deep Learning methods using the RELLISUR Dataset

Μέλη εξεταστικής επιτροπής συμπεριλαμβανομένου και του Εισηγητή

Η διπλωματική εργασία εξετάστηκε επιτυχώς από την κάτωθι Εξεταστική Επιτροπή

Α/α	ΟΝΟΜΑ ΕΠΩΝΥΜΟ	ΒΑΘΜΙΔΑ/ΙΔΙΟΤΗΤΑ	ΨΗΦΙΑΚΗ ΥΠΟΓΡΑΦΗ
1	Αναστάσιος Κεσίδης	Καθηγητής	Anastasios Kesidis Digitally signed by Anastasios Kesidis Date: 2023.09.18 11:02:17 +03'00'
2	Πάρις Μαστοροκώστας	Καθηγητής	Paris Mastorokostas Digitally signed by Paris Mastorokostas Date: 2023.09.18 11:13:57 +03'00'
3	Αθανάσιος Βουλόδημος	Επίκουρος Καθηγητής	Athanasios Voulodimos Digitally signed by Athanasios Voulodimos Date: 2023.09.20 20:05:26 +03'00'

This thesis was carried out under the supervision of Dr. Anastasios L. Kesidis, to whom I am very grateful for his guidance and patience.

Dedicated to my family
and to my friends, for the
patience and support that
they showed these last months

Abstract

It is well known that digital images are used every day in many fields, which include medicine, agriculture and law enforcement. The systems used in the above fields require the input images to be of high quality without noise. However, there are many factors that can reduce the quality of an image. One of them is low light conditions, e.g when the image is taken at night time and/or indoors. Due to these conditions, the light reflected from the scene and reaching the sensor is very weak resulting in final images characterized by low contrast and brightness, with pixel values concentrated in the left part of the histogram, which introduces noise and color distortions. Such images must be enhanced before being used in vision systems. The purpose of this work is the study and application of such image processing methods, that are known as Low Light Image Enhancement (LLIE) methods, and aim to enhance dark images without introducing additional noise. Initially, a variety of learning free algorithms are studied, which apply a mathematical operation to each image. Then, in an attempt to improve the results, machine learning methods are involved, specifically deep learning models, which use information from the available data set, to learn the representation between the dark and ground truth images. For this purpose, two variations of a Convolutional Neural Network architecture are proposed. All the methods are applied to the RELLISUR dataset, which includes truly dark images, rather than images produced by applying a transformation to the ground truth images, and is divided into different darkness levels. The results of the methods are evaluated by metrics that either use a reference image or not. This evaluation led to the conclusion that classical methods, although they give satisfactory results, show weakness in excessively dark images, introducing additional noise and color distortions. On the other hand, the deep learning methods that were applied, and especially the two proposed variations, produced more accurate visual results where the experimental images are characterized by high contrast and brightness.

Contents

Chapter 1 – An Introduction

At this first chapter we will introduce basic concepts of Digital Image Processing, with a brief summarization of historical background. After that we will describe the Low Light Image Enhancement (LLIE) problem that we will try to solve and the dataset to be used. At the end of the chapter we will include a brief list with the algorithms that will be used, and describe the quality metrics which will be used for the assessment of each method.

1.1 Image Processing

As humans we rely heavily on vision to make sense of the world around us. By making sense we mean processes such as recognizing objects, identifying differences between objects and generally gaining a more general sense of the landscape/scene that we see with our eyes. Thus, we understand that as an image we could define any scene (moving or static) that represents something. This scene, however, can be altered by phenomena such as low lighting, fog, or even by movement since in reality no scene is static. Nevertheless, in most cases, we remain able to interpret what we see, which is due to the fact that the human brain "processes" the image, improves it, and gives it meaning, with this whole process happening instantaneously. In this sense, we could define as image processing the process that the human brain follows to give meaning to what we see. So, we understand how important a process it is, since our survival is largely based on it.

But images appear not only as the scenes we see in the world around us, but also as the results of digital processes such as taking digital images and producing digital scenes using a computer. These images are called digital, and they can be corrupted by the same factors, such as the introduction of noise during the transfer or compression of an image, or even the partial destruction of a related file. In this case, appropriate image processing methods exist and are applied, to recover and improve each image. At this point we can give a clearer and stricter definition of the field of image processing.

Digital image processing: *is defined as the processing of a digital image, using a computer, with the ultimate goal of either improving/retrieving visual information for use by humans or machines, or extracting additional visual information such as image segmentation.*[11]

At this point let us mention that a more rigorous definition of a digital image is a 2D function, f(x,y), where the spatial coordinates (x,y) as well as the intensity f(x,y) are discrete quantities.

As we will see below, digital images are used in every aspect of everyday life, from agriculture and industry, as well as in every form of science such as medicine, astronomy. In each of these areas, we process digital images to extract some useful information from them. For example, in agriculture we can use satellite images to learn about the distribution of crops on a land surface, in medicine we can use CAT scans to find the location of a malignant tumor, and in astronomy we can use data from telescopes to find the structure of a distant celestial body.

In order to produce reliable results, all these applications must use high-quality images in visual information that contain minimal noise and visual distortions. This is almost never possible since digital images can be subject to various kinds of distortions such as noise, blurring, etc. (See subsection 1.3). This is where the field of digital image processing (or pre-processing as it is called, in the context of each application) enters, where we process an image, before feeding it to any of the above applications, with the aim of either improving it or extracting additional information. More details will be mentioned in a later section, in this chapter.

Based on what was mentioned above we understand the importance of the field of digital image processing. In the following we will study in a little more detail the applications where digital images are used as well as the various stages of processing to which these images are subject in order to remove noise and extract information. Then we will analyze the part on which the rest of the work will focus, strictly defining the problem to be tackled and describing the course we will follow.

1.2 Digital Image Processing applications

As mentioned above, digital images are used in a multitude of applications. In this section we will refer to each of them in a little more detail by describing basic examples. The main areas we will look at are applications in medicine, agriculture and law enforcement.

Medicine

This is one of the most important areas of using digital images and digital image processing. The most basic use is the processing of images from examinations such as MRI and CT scans to extract (visual) information that can lead to a diagnosis and selection of appropriate treatment for each patient.

MRI, or Magnetic Resonance Imaging, is a medical imaging technique that uses strong magnetic fields and radio waves to produce images that depict the organs and general anatomy of the (human) body. The end result is a cross-section depicting the anatomy of the region of interest. For example, figure 1.1 depicts images of the brain as obtained from MRI scans. Doctors can then use these images to find abnormalities or even conditions in the area of interest (here in the brain) such as cancerous tumors, cysts and tissue diseases. For example, figure 1.2 shows an MRI of a brain in which a cancerous tumor has been detected.

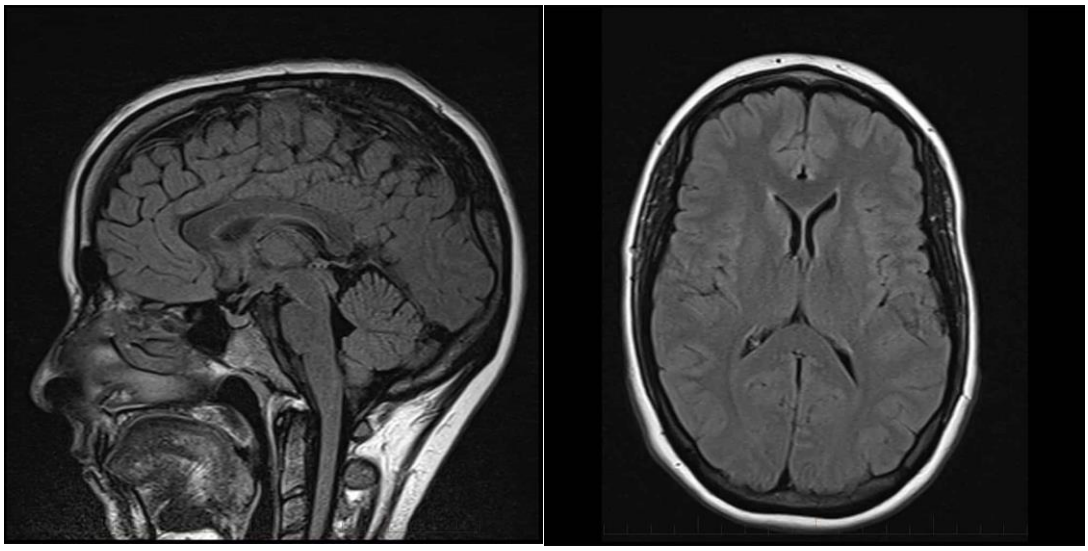


Figure 1.1: MRI Scans of the Human brain[1]

[1] From https://www.melbourneradiology.com.au/diagnostic-imaging/mri-scan-brain/

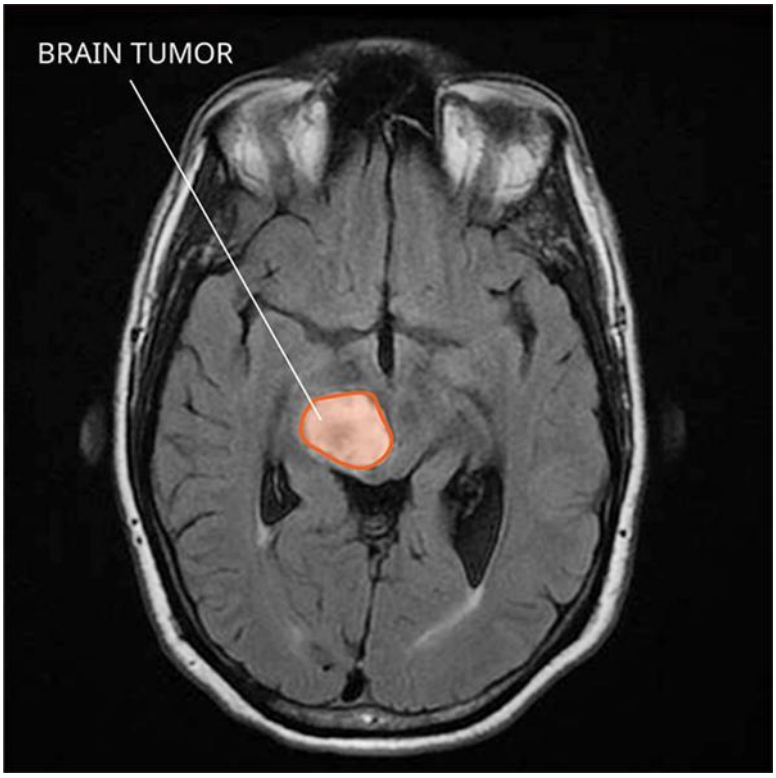


Figure 1.2: Cancerous tumor as detected by MRI scan[2]

CT scans can lead to the same or even better conclusions, the results of which are 2d slices that can be combined to construct a 3d representation of the body's anatomy. Unlike MRI scans, CT scans do not use strong magnetic fields, but a set of thin X-ray beams that scans the patient. It is usually used in patients who have a metal implant for which an MRI scan is not indicated. For example, figure 1.3 demonstrates an image from CT scans of soft tissues. CT scans can then be used by doctors to detect abnormalities in the area of interest (here in the soft tissues) such as cancerous tumors and internal bleeding. Figure 1.4 depicts cancerous tumors (early stages above and advanced below) as detected by CT scans [1].

[2] From https://www.cancer.gov/rare-brain-spine-tumor/tumors/diffuse-midline-gliomas

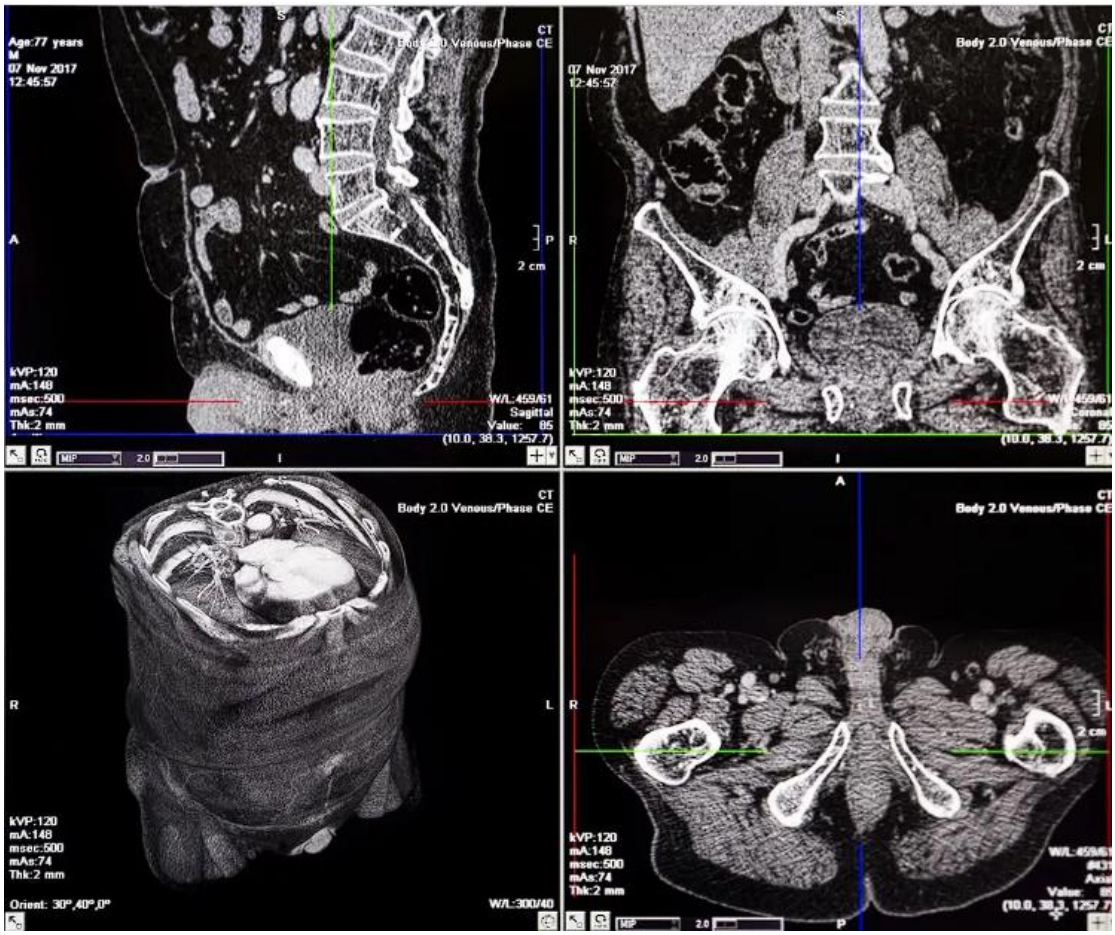
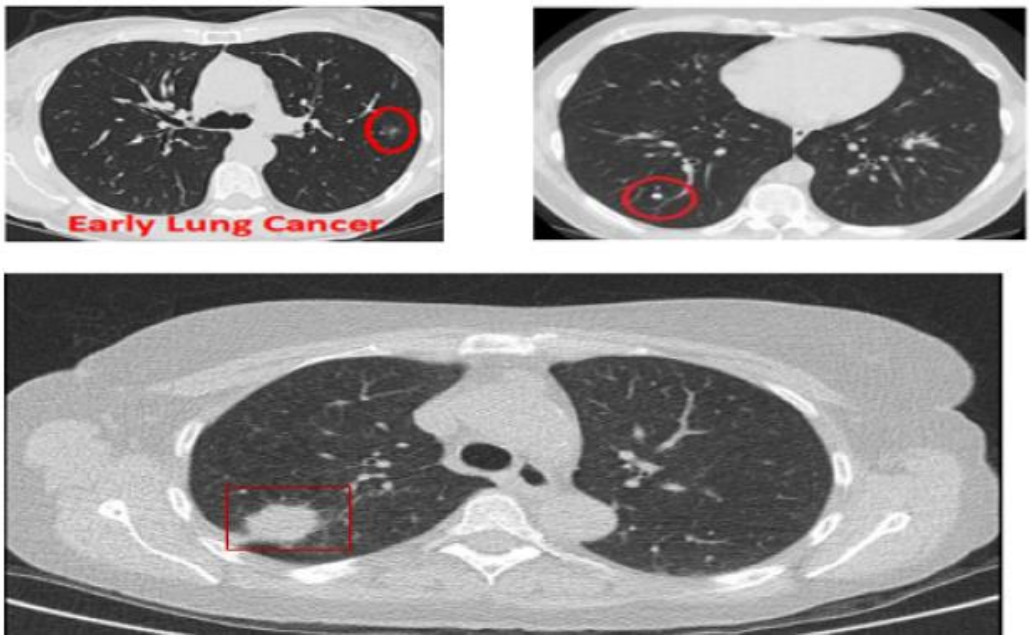


Figure 1.3: CT scans of soft tissues[3]



Εικόνα 1.4: Cancerous tumors in the lungs as detected by CT scans [1]

[3] From https://www.medicalnewstoday.com/articles/153201

In an attempt to automate the diagnosis process, digital images produced by MRI and CT scans can be used in classification algorithms, to classify tissues into healthy and non-healthy, as well as to locate cancerous tumors [2].

Agriculture

Another large area of application of digital imaging is agriculture. In this case, remote sensing images, which have been taken from a satellite, are used. The images taken and processed usually depict pieces of land which may include forested areas, are intended for cultivation or are already cultivated and we want to examine the distribution and health of the crops.

For example, if the area of interest consists of forest land, roads and residential areas, then on the satellite images a classification algorithm is applied so that each pixel is classified into one of these three categories. In this way one have the possibility, using these results, to control (using historical data) the evolution of urban expansion and the results of human activity, as well as the evolution of forest expansion (forestation/deforestation)[3]. For example, in figure 1.5 a satellite image of the city of Maanshan in China is depicted, and figure 1.6 demonstrates the classification in each class [4]. These images come from the Landsat-5TM satellite.

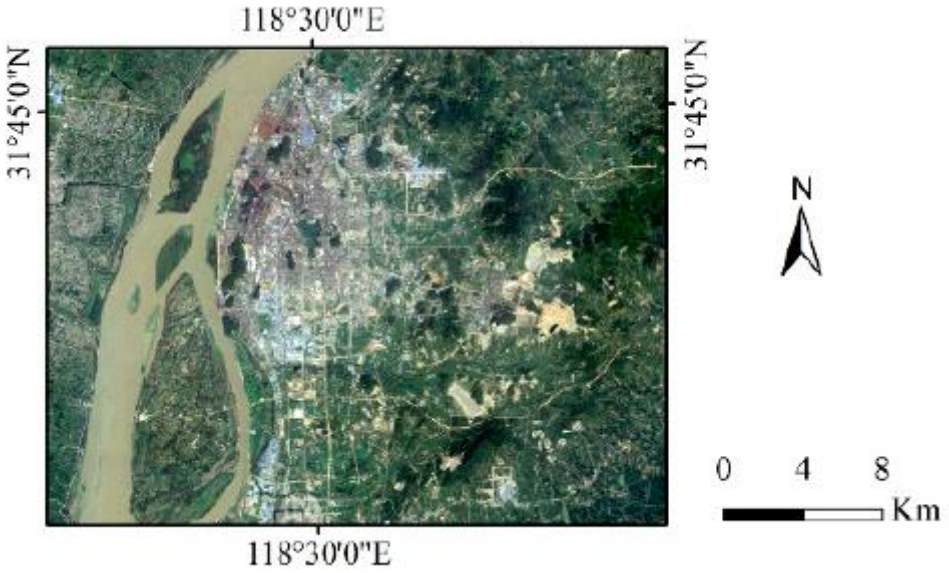


Figure 1.5: Satellite image of Maanshan city in China[4]

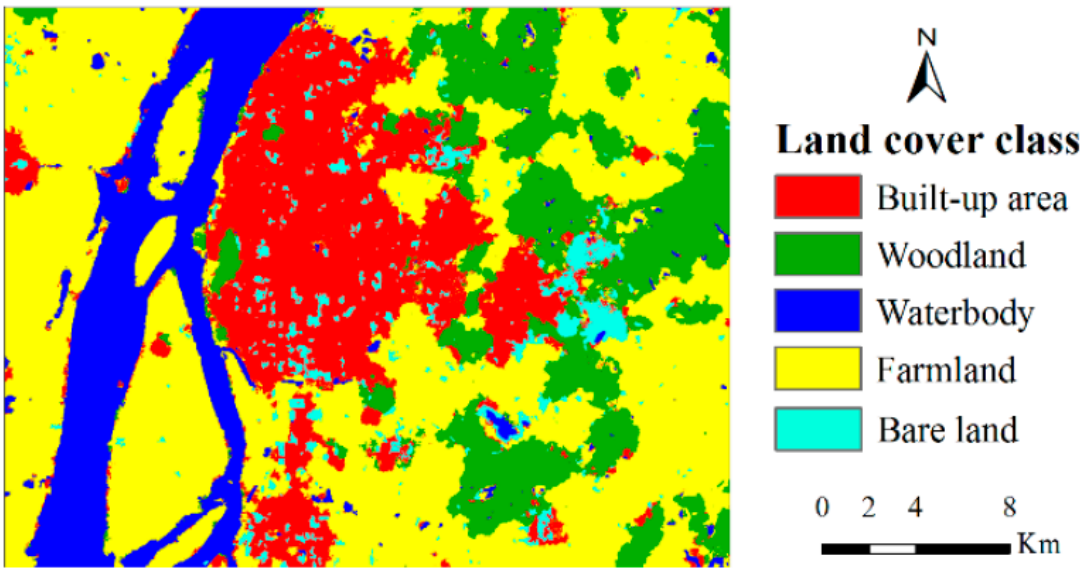


Figure 1.6: Result after classifying each pixel in the respective class[4]

Law enforcement and video surveillance

The last area to be examined is law enforcement and video surveillance. In this case, key applications include digital images of fingerprints as well as images from security cameras (derived from video frames) that can be used to identify the perpetrator or to prevent suspicious activity.

In the first case, digital fingerprint images are used to build a huge database of such images. These can then be used to identify the perpetrator at a crime scene where fingerprints have been collected. An example of such an image [5] is shown in figure 1.7 below. Once a fingerprint is collected from a crime scene, it is compared to this massive database until a match is found. To do this, a matching algorithm is applied to the digital image of the fingerprint that through a process looks for the appropriate match for the digital image it gets as input. One such example is shown in figure 1.8 where the Minutia Cylinder-Code (MCC) method is boosted using GPU for greater performance [6].

Figure 1.7: Digital images of fingerprints of an adult (left) and a child (right) [5]

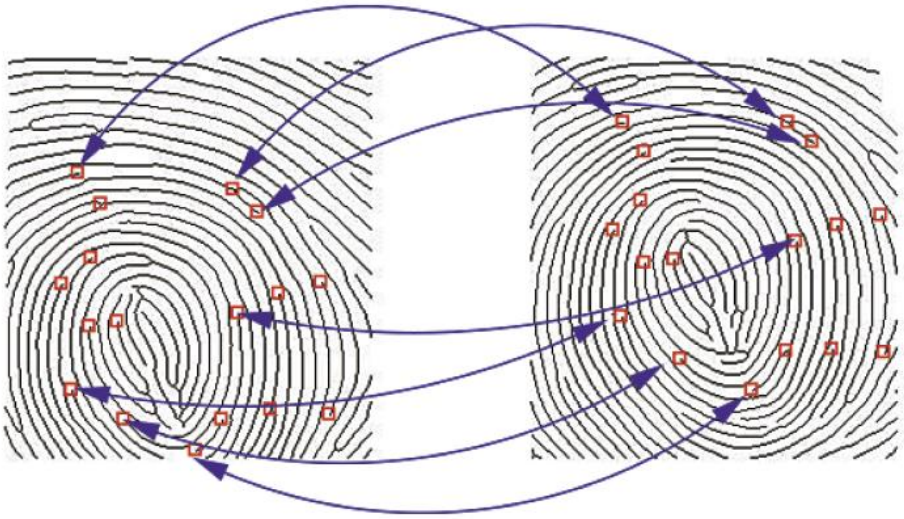


Figure 1.8: Fingerprint matching using MCC algorithm boosted with GPU [6]

In the second case we refer to digital images as obtained from security cameras or traffic control cameras (from their frames). These images (or series of images) can be used to detect illegal activity and prevent a crime. For example, in figure 1.9 images from security cameras are depicted that have recorded the theft of objects. These images come from the DCSASS dataset created by Sultani et al [7] and contains 16853 videos, which are

used to train suspicious activity prevention and recognition systems. As for example in [8] where the authors have trained a CNN to recognize robbery and shooting footage.

A different approach is to use digital images from traffic control cameras, either to detect license plates of vehicles exceeding the speed limit or to identify and report accidents for faster response. For example, figure 1.10 shows images from a traffic control camera before and after a car accident. Such images were used by [9] to train an automatic intersection accident detection system.

Figure 1.9: Theft shots as recognized by the algorithm of [8]

Figure 1.10: Images from a traffic control camera before (left) and after (right) automatic accident detection [9]

1.3 Noise and degradations

Based on what was mentioned in the previous subsection, it is understood how important digital images are in our daily lives. In all the above cases, the digital images are used as input to some digital image processing system with the final aim of extracting useful (visual) information. For example, in MRI scans, a segmentation algorithm could be applied that segments a brain scan into healthy tissue and a cancerous tumor, leading to automated diagnosis [10]. However, in order to make a correct and reliable diagnosis, the images that are used as input of the diagnosis system must be of high quality and not be altered by factors such as noise and artifacts. In remote sensing images, classification algorithms can be applied that assign each pixel to a class, such as land, crops, bare ground, etc. For this classification to be correct, the images must also be of high quality and not been altered by factors such as fog/cloudiness and noise due to transmission from the satellite to the terrestrial systems. Finally, and in the automatic accident detection system, the images are passed through a pipeline in which appropriate features are extracted, which are used to detect accidents. Even in this case, introducing noise/blurring into the image can dramatically reduce system performance leading to incorrect results.

It can be seen that in all cases the input images of the respective system must be clear and of high visual quality, otherwise there is a risk that the results produced will be incorrect and unreliable. In the real world, however, this is very rare, as digital images can be corrupted by many factors. The two dominant factors are either of a technical nature such as for example the noise introduced during the capture and transmission of an image, or of natural origin due to the environment where the capture is made (low lighting, moving objects/blurring, fog) [11].

For noise due to the technical part (reception, transmission) two major categories can be distinguished, Gaussian noise and salt and pepper noise. Below a brief description of these two categories is given.

Gaussian Noise

This type of noise can be introduced into a digital image both during reception and during transmission. During shooting Gaussian noise can occur from the sensor and can be due to low lighting levels or high temperature. Gaussian noise is additive noise and changes the value of each pixel slightly by an amount that follows a Gaussian distribution [12].

$$I_{noise}(x,y) = I_{clean}(x,y) + g \text{ (Σχέση 1.1)}$$

where I_{noise} is the image with noise, I_{clean} is the clean image, (x,y) are the spatial coordinates of each pixel, $I_{noise}(x,y)$ and $I_{clean}(x,y)$ are the gray values of the pixel (x,y) of the image with noise and without noise respectively. Finally, g is the amount by which the value of each pixel changes, and has a probability distribution that of the Gaussian distribution:

$$P(g) = \sqrt{\frac{1}{2\pi\sigma^2}} * e^{-\frac{(g-\mu)^2}{2\sigma^2}} \text{ (Σχέση 1.2)}$$

Below is an example of Gaussian noise. This result was produced using MATLAB and the image of Lena[4] and introducing Gaussian noise with mean value μ=0.005 and variance σ=0.01.

Figure 1.11: Image without noise (left) and with Gaussian noise (right) with μ=0.005 and σ=0.01

[4] http://www.lenna.org/

Salt and Pepper noise

This is the appearance of black and white pixels randomly distributed across the image. The appearance of this kind of noise is due to variations in the image signal during its transmission. During the transmission of the image, some pixels become corrupted and are replaced by the maximum or minimum value that a pixel can take (255 and 0 respectively for an 8-bit image) [12].

In Figure 1.12 an example of salt and pepper noise is depicted, again using Lena's image.

Figure 1.12: Image without noise (left) and with salt and pepper noise (right)

In addition to technical reasons, the visual information in an image can also be altered by the (environmental) conditions that prevail during the capture of the image. Two very basic examples are blurring due to the movement of an object at the time of shooting and low light conditions (which is also the problem we will try to solve at the following chapters).

Blurring

It is a distortion of an image that is mainly due to movement of either the object of interest or the camera itself [13]. Typical examples are the images and videos resulting from traffic control cameras. As mentioned above, such images can be used to recognize the license plate number of a car, but because the objects are in motion, the characteristic noise of blurring

occurs, making it difficult to recognize the license plates. Figures 1.13 and 1.14 demonstrate an example of a blurred image of a car and a license plate as derived from the study in [14].

Figure 1.13: Blurred car and license plate images [14]

Figure 1.14: Blurred license plate image [14]

Low light conditions

This is noise introduced into an image when the lighting level of the scene is extremely low [15]. This is mainly because the light reflected from the surface of the objects is weak leading to chromatic aberrations and noise. This kind of noise can dramatically reduce the performance of systems using digital images. As a typical example images from security system

cameras are mentioned, which during the night face a problem with low light conditions [16]. Figure 1.15 depicts an example of such an image.

Figure 1.15: Images as they appear in low light conditions [16]

Such images cannot be fed to the systems mentioned in the previous sub-section because they will produce unreliable results, which can lead to wrong decisions. Before feeding them to these systems we should first denoise them by applying a suitable image denoising algorithm to improve the input quality of the system. Noise due to low light conditions, as well as low light image enhancement are the main topic of this thesis, during which we will develop and test/evaluate a set of algorithms, commenting on the results and comparing the performance of the algorithms with each other.

In the next sub-section, the problem of low light conditions will be described in more detail and we will study the data set, which will be used in the rest of the thesis for the application and evaluation of each low light image enhancement method.

1.4 Low Light Images and the RELLISUR Dataset

As mentioned above, low light images are images taken in low light conditions, such as indoors, at night or on cloudy days. Due to these conditions, the light reflected from the surface of the objects in the scene is weak, resulting in reduced image quality due to noise and color distortions. The quality of such images can become even worse after processes such as conversion, storage and transfer, processes that themselves introduce additional noise.

More specifically, for an image to be considered as low light, the lighting conditions must not satisfy certain criteria. But it is impossible to define theoretical values for a low-light environment, as a result of which there are no unified standards. The standard can change depending on the manufacturer/researcher, for example Hikvision uses illuminance (lux measurement units) as a criterion and has the following categories i) dark level (0.01lux-0.1lux), ii) moonlight level(0.001lux-0.01lux) and iii) starlight level (smaller than 0.001lux). In [17], where the dataset we will use comes from, as brightness levels they use the Exposure Value (EV) which will be defined later when the dataset is going to be described.

Images taken in such conditions show low brightness levels, low contrast, narrow gray range and color distortions. To see this better we will use the RELLISUR dataset [17], which we will describe next. Specifically, the average brightness and contrast for all images of each brightness level separately (details at the end of this sub-section) will be calculated, and then a random low light image from each brightness level and its respective histogram will be displayed. Also, for comparison purposes the same procedure will be applied for the corresponding ground truth images. Histograms have been generated by MATLAB software.

Darkness Level: 3.0

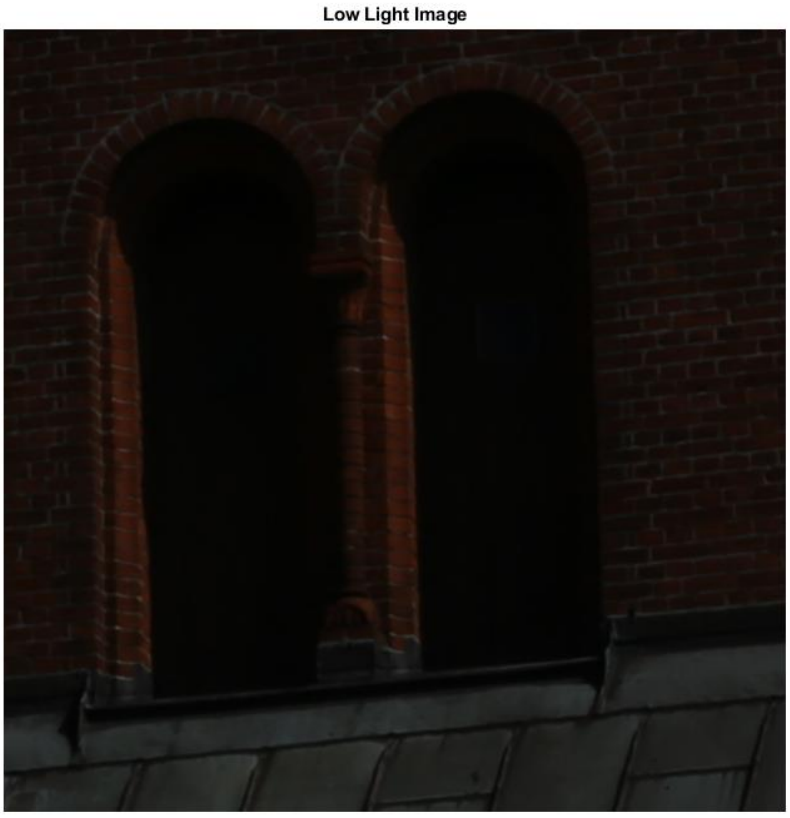


Figure 1.16: Random low light image from Darkness Level 3.0

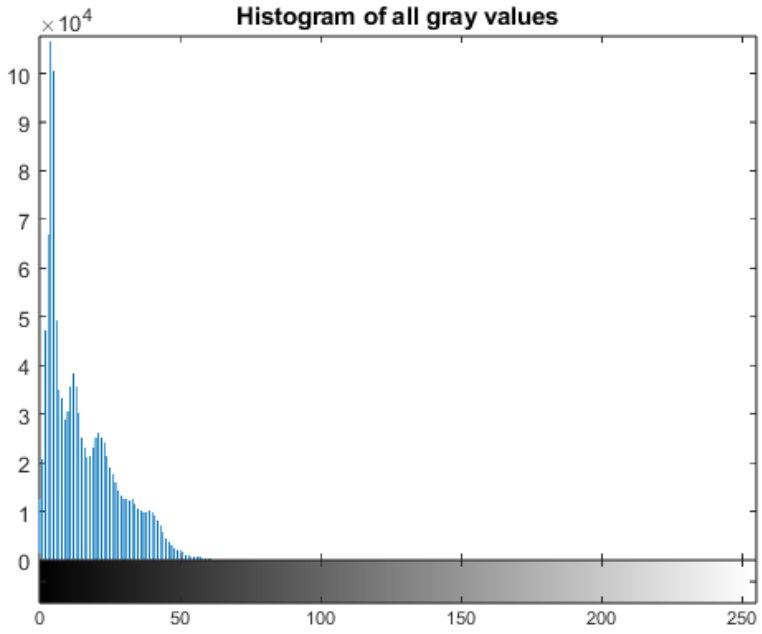


Figure 1.17: Histogram of all gray values of the figure 1.16

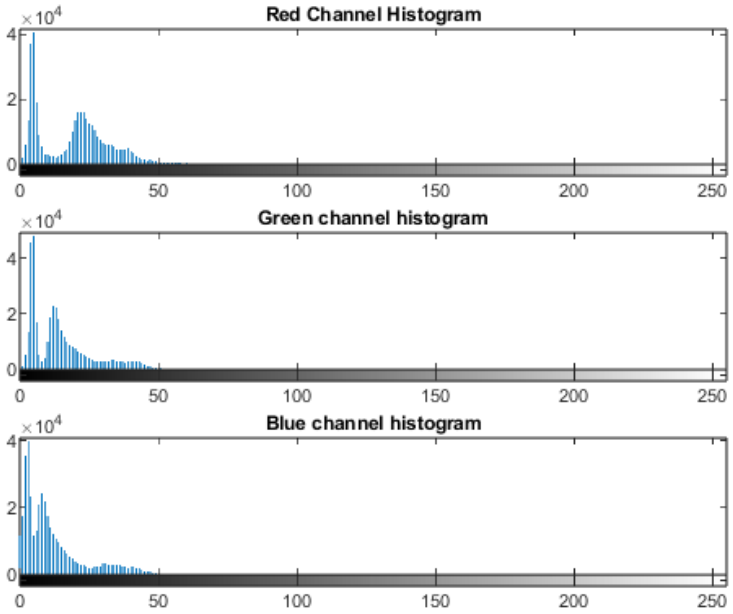


Figure 1.18: histogram of Figure 1.16 per channel

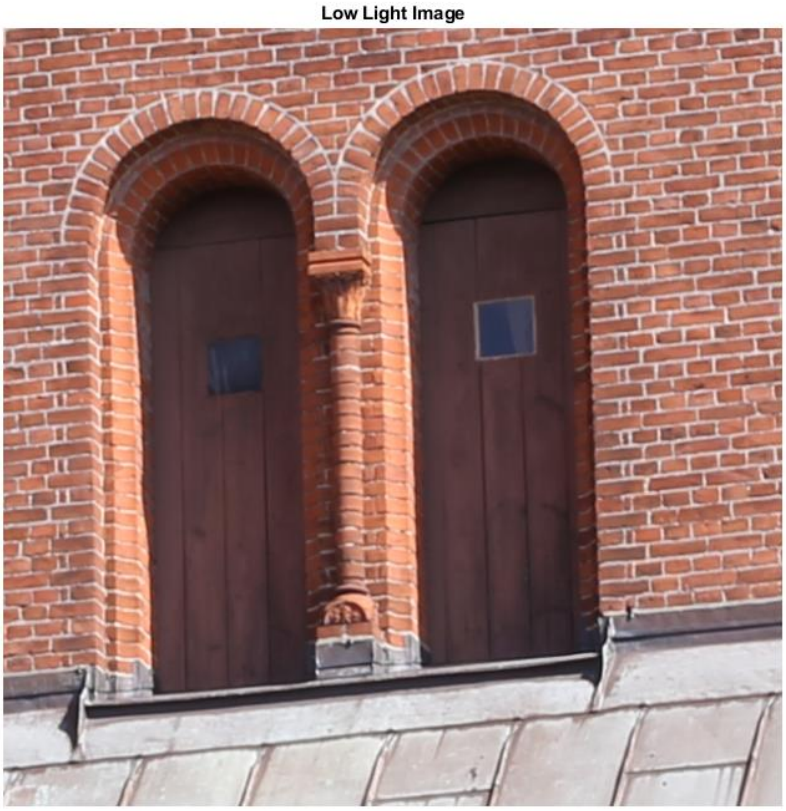


Figure 1.19: Normal light version of 1.16 figure

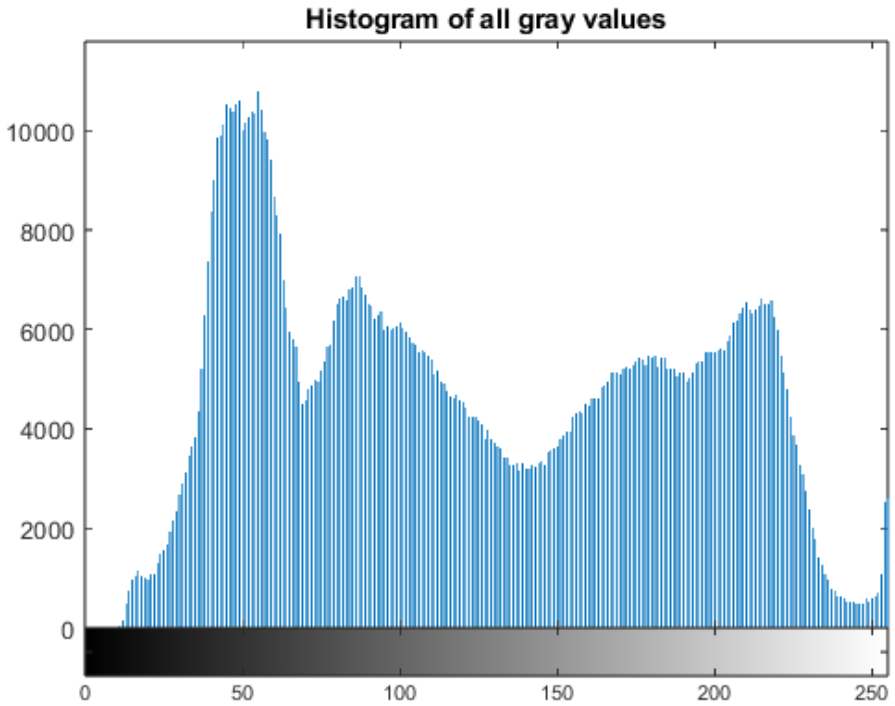


Figure 1.20: Histogram of all gray values of the figure 1.19

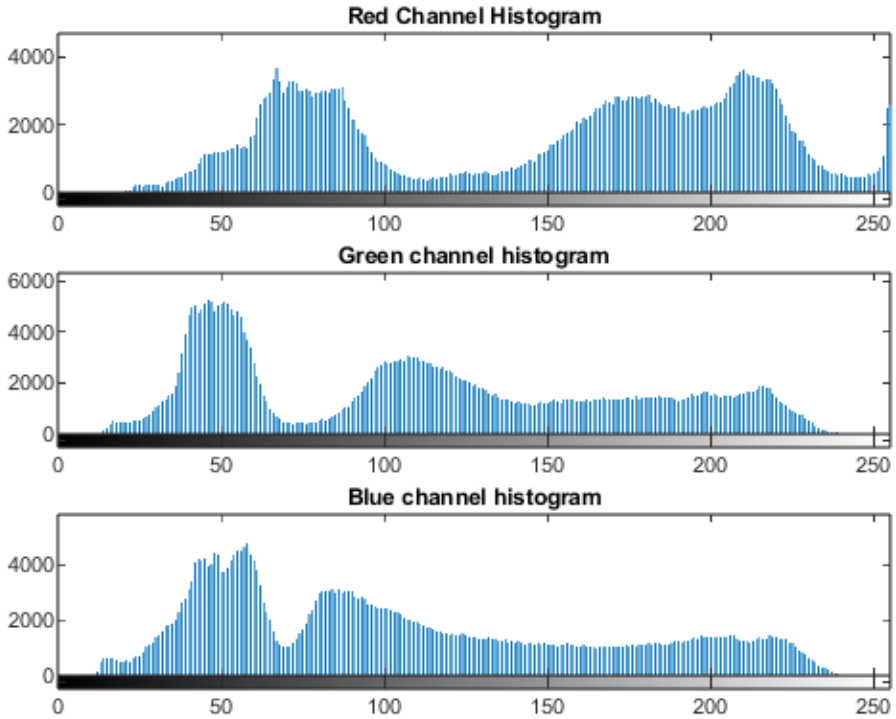


Figure 1.21: histogram of Figure 1.19 per channel

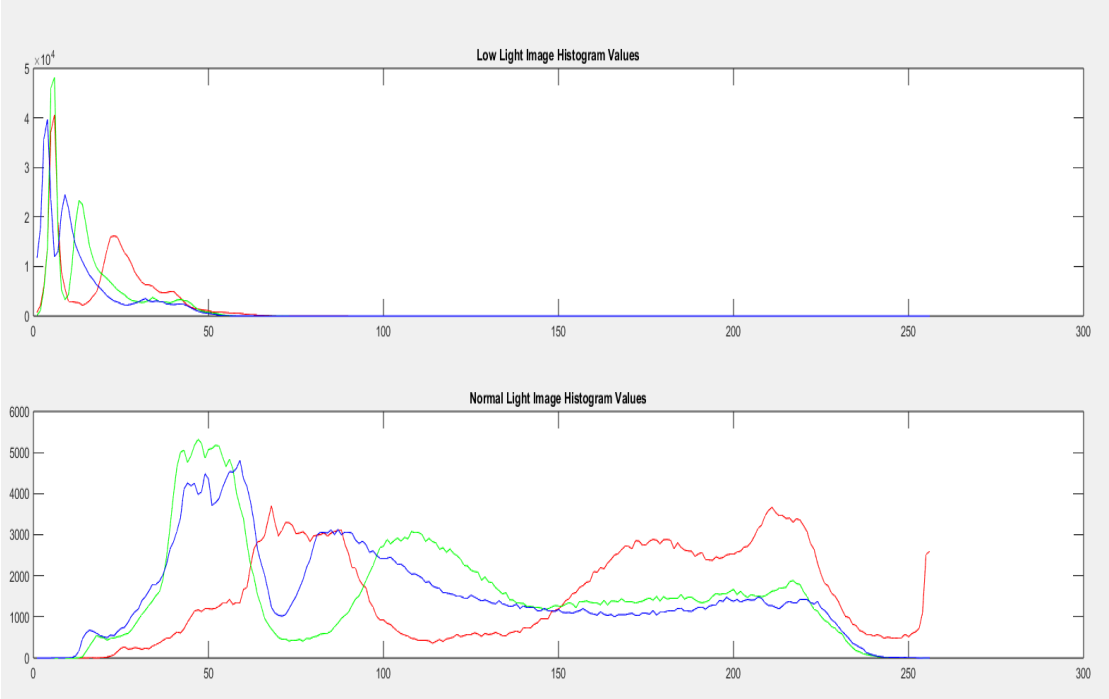


Figure 1.22: Comparison of histograms per channel

Darkness Level: 3.5

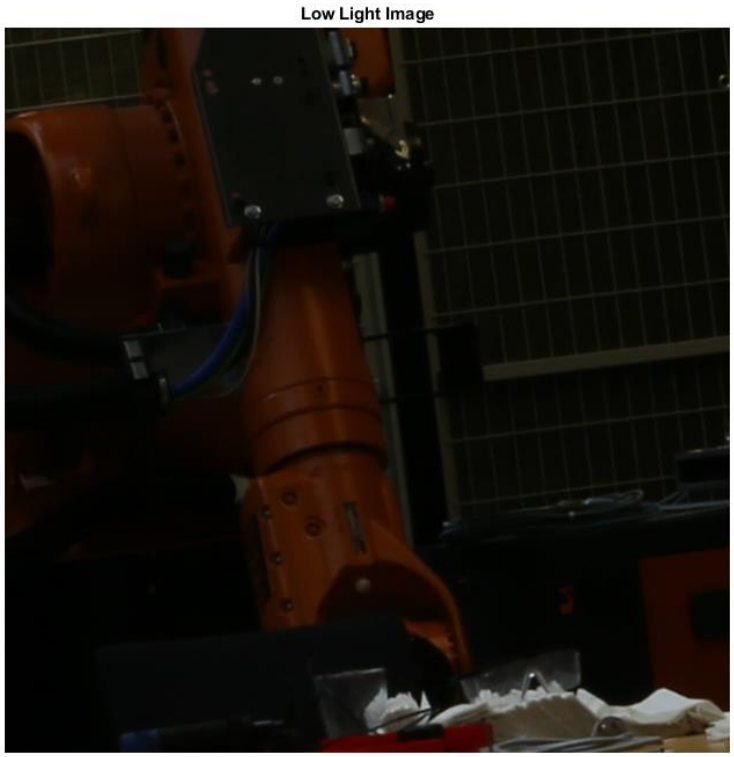


Figure 1.23: Random low light image from Darkness Level 3.5

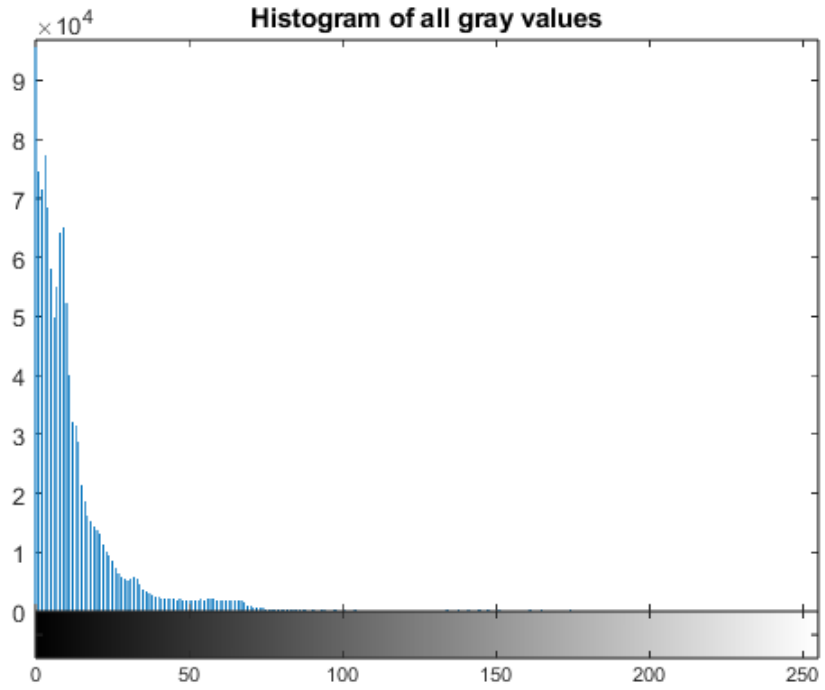


Figure 1.24: Histogram of all gray values of the figure 1.23

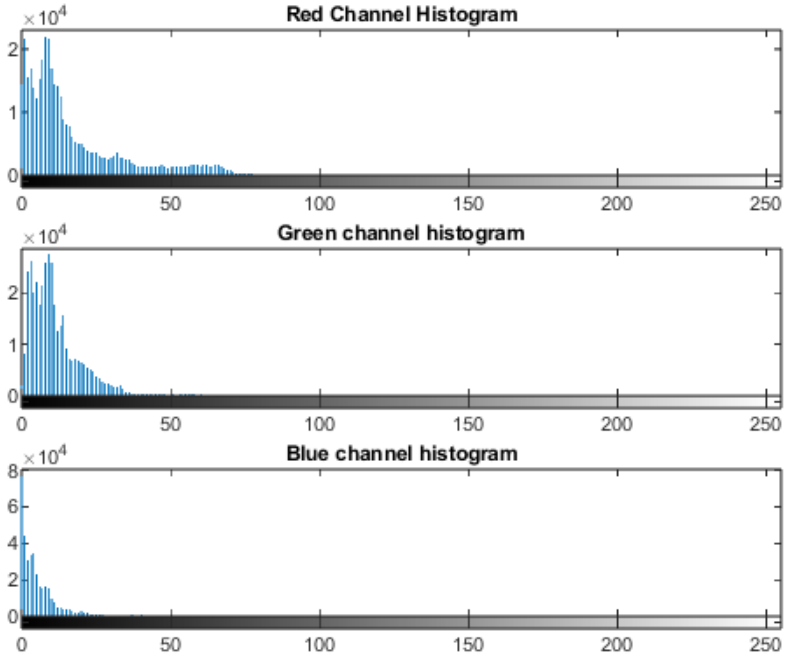


Figure 1.25: histogram of Figure 1.23 per channel

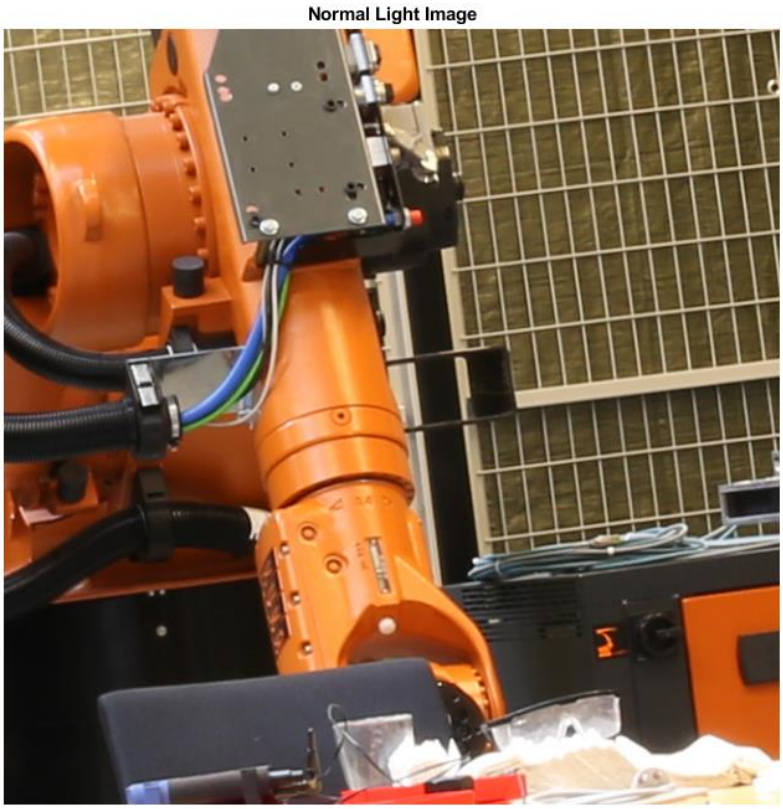


Figure 1.26: Normal light version of 1.23 figure

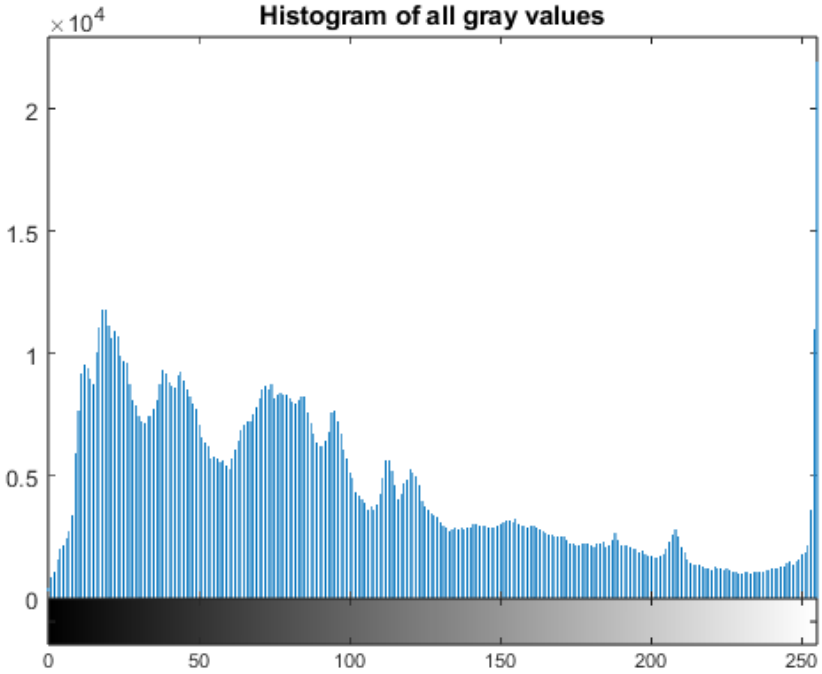


Figure 1.27: Histogram of all gray values of the figure 1.26

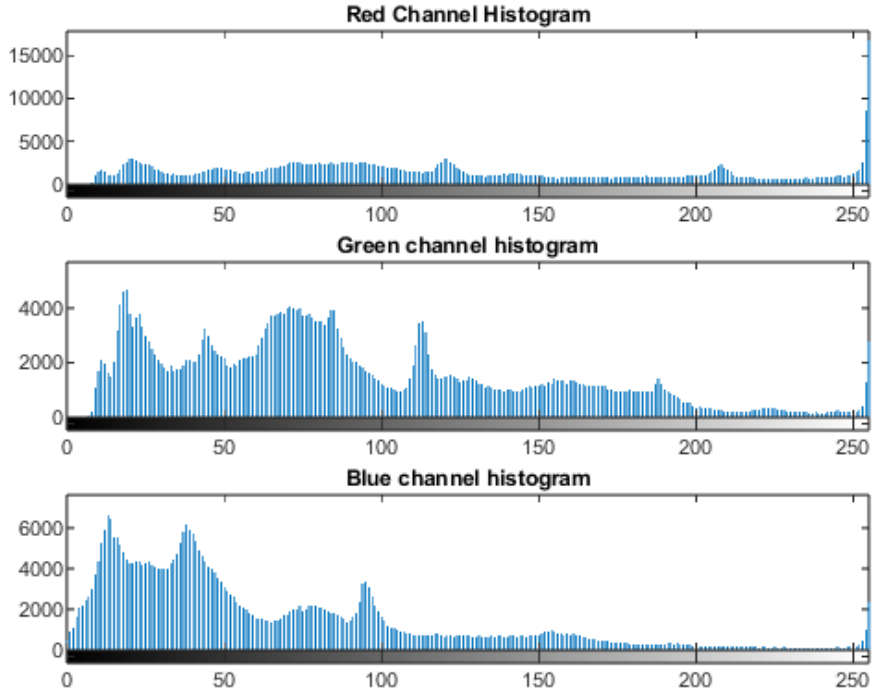


Figure 1.28: histogram of Figure 1.26 per channel

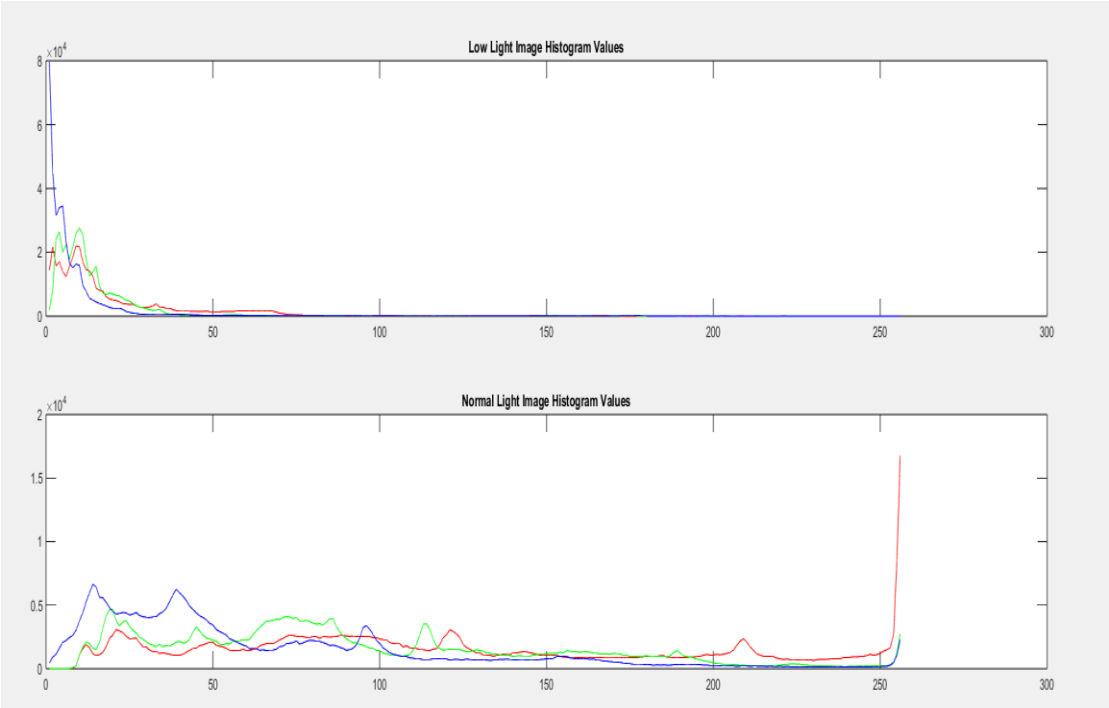


Figure 1.29: Comparison of histograms per channel

Darkness Level: 4.0

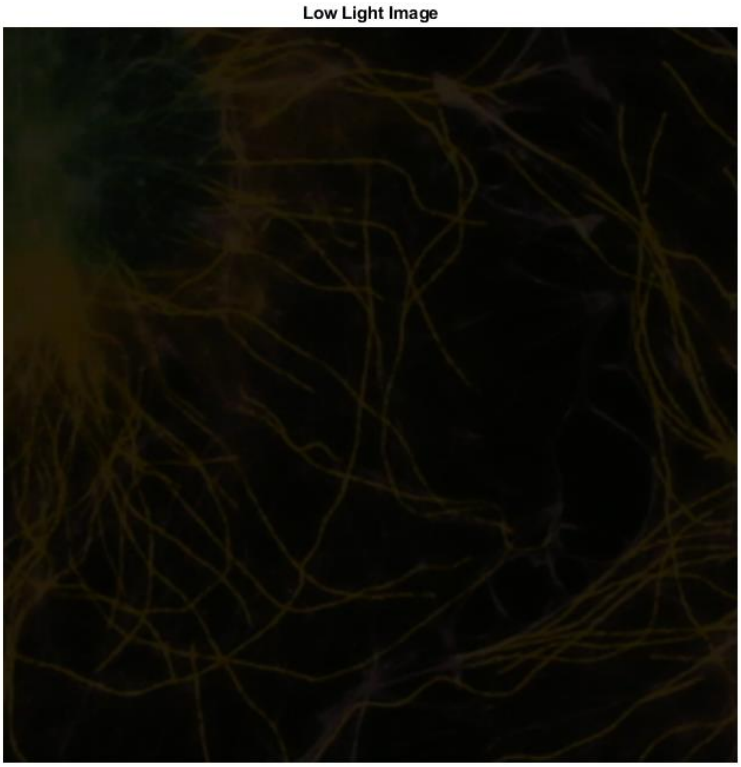


Figure 1.30: Random low light image from Darkness Level 4.0

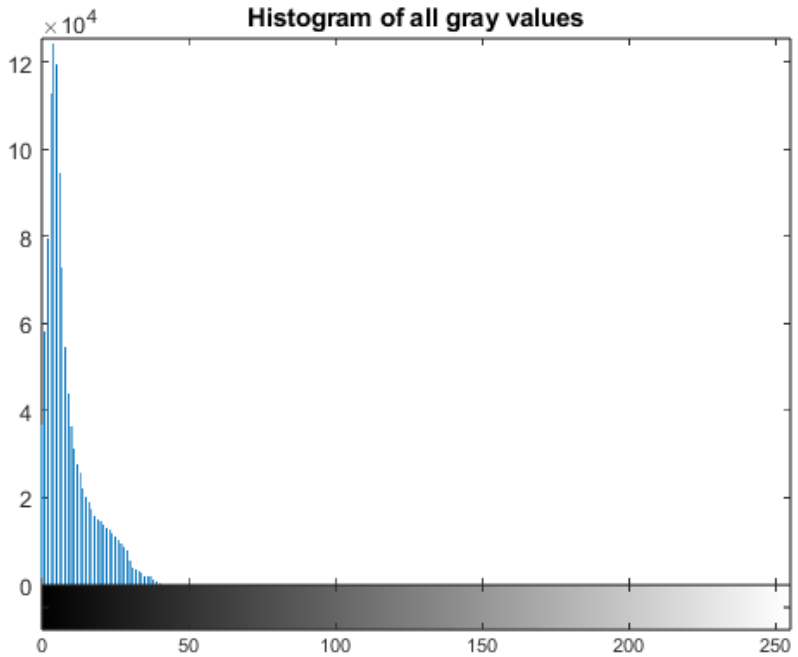


Figure 1.31: Histogram of all gray values of the figure 1.30

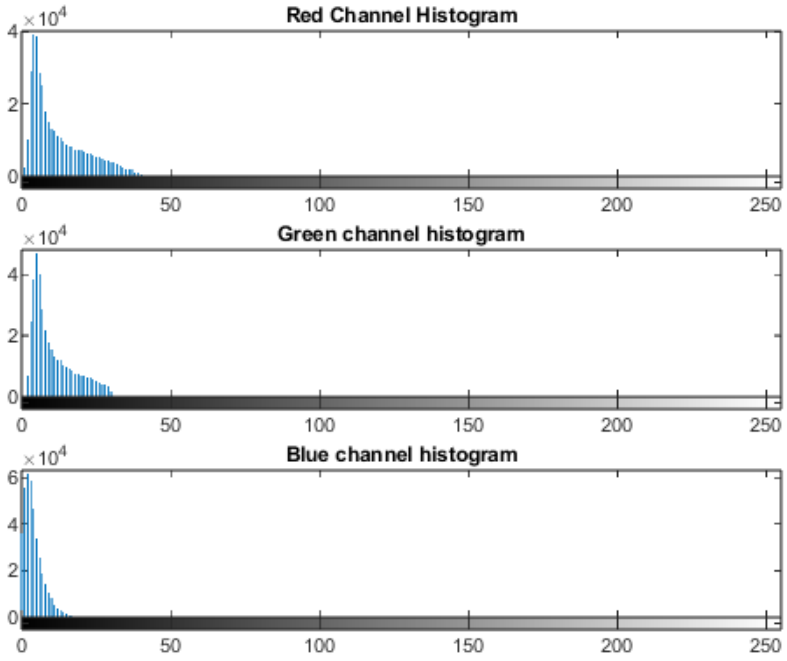


Figure 1.32: histogram of Figure 1.30 per channel

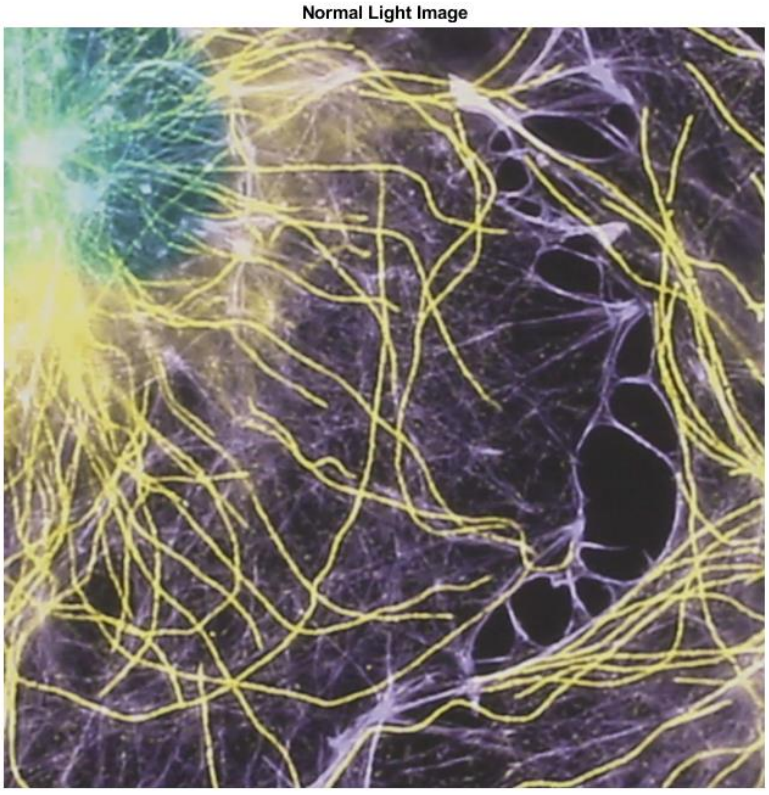


Figure 1.33: Normal light version of 1.30 figure

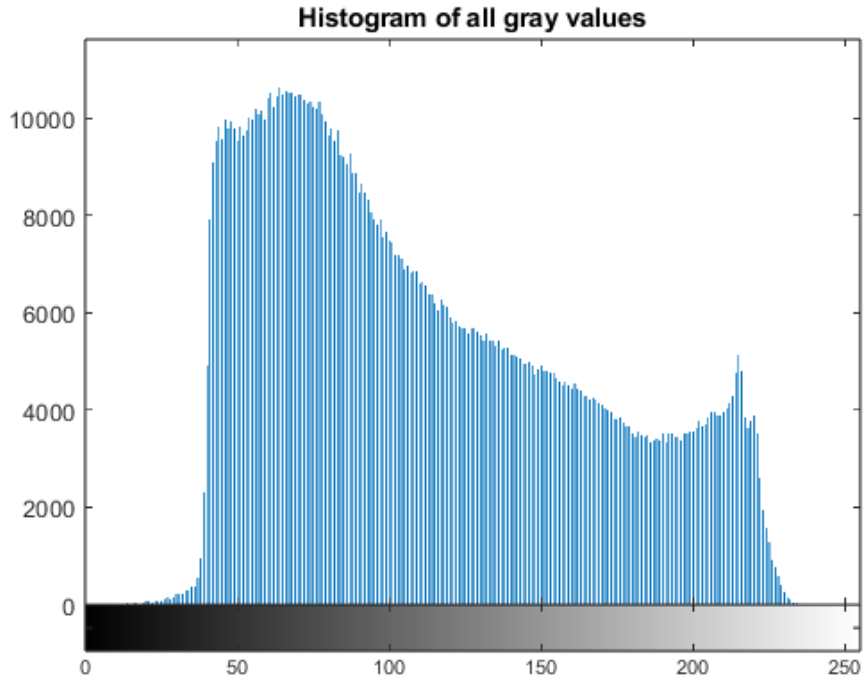


Figure 1.34: Histogram of all gray values of the figure 1.33

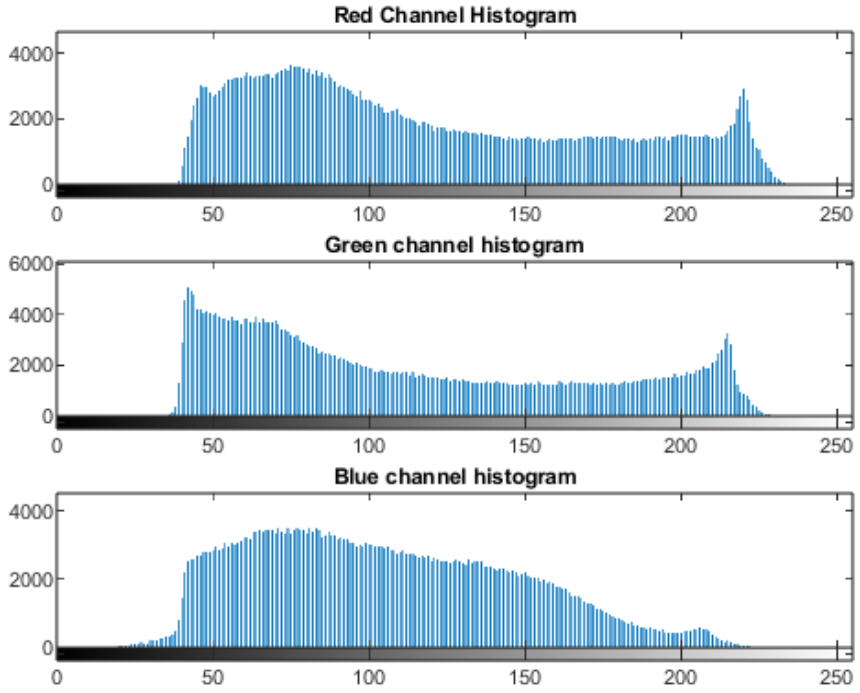


Figure 1.35: histogram of Figure 1.33 per channel

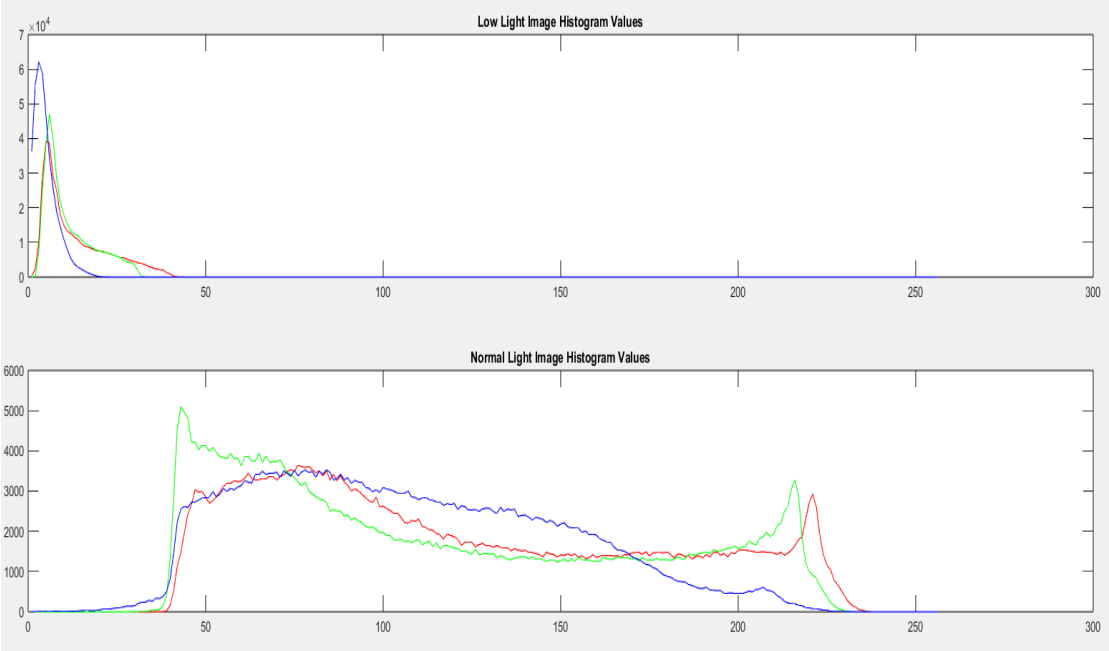


Figure 1.36: Comparison of histograms per channel

Darkness Level: 4.5

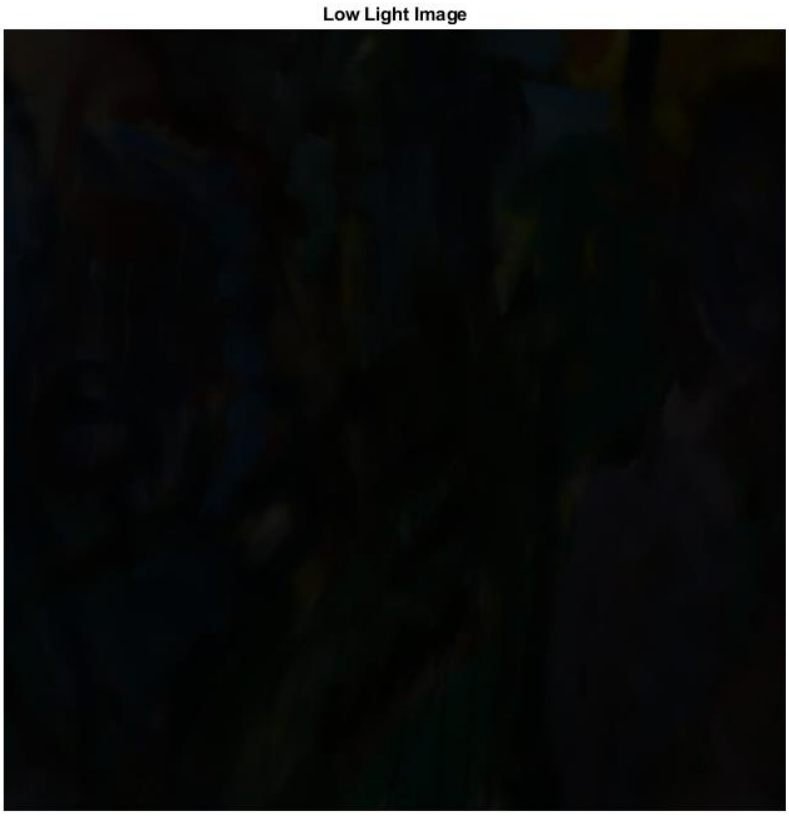


Figure 1.37: Random low light image from Darkness Level 4.5

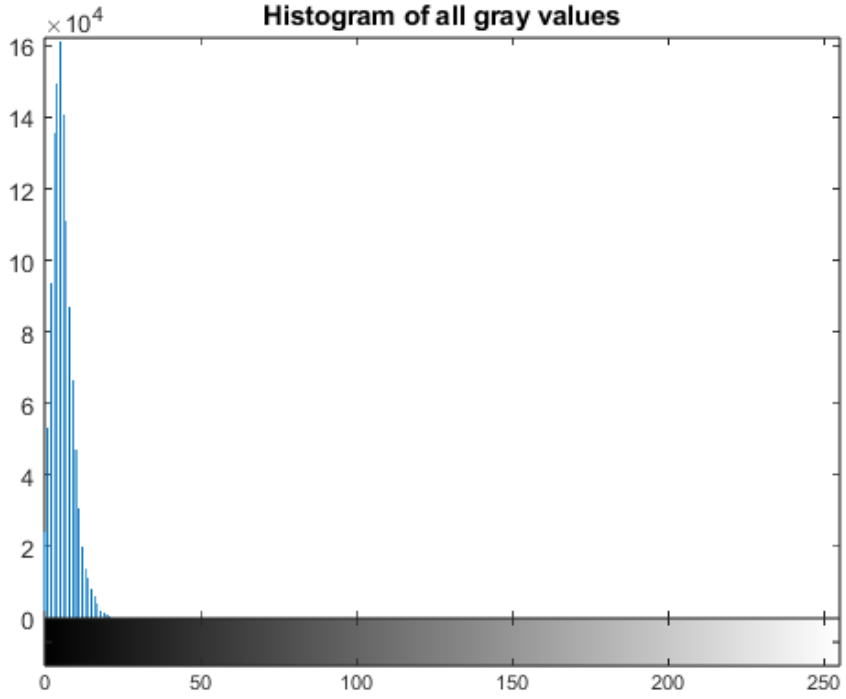


Figure 1.38: Histogram of all gray values of the figure 1.37

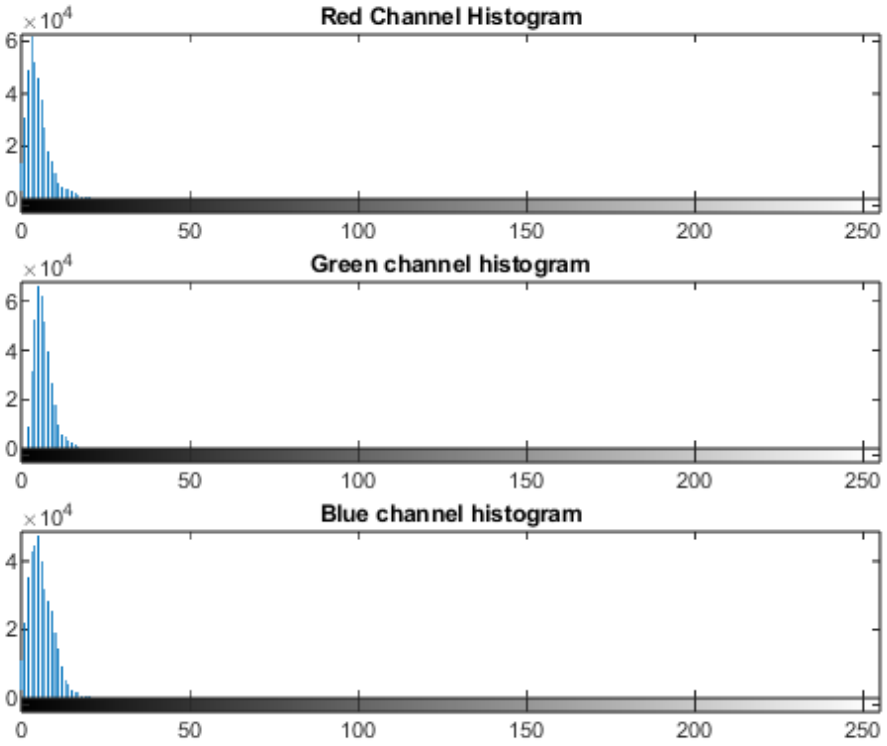


Figure 1.39: histogram of Figure 1.37 per channel

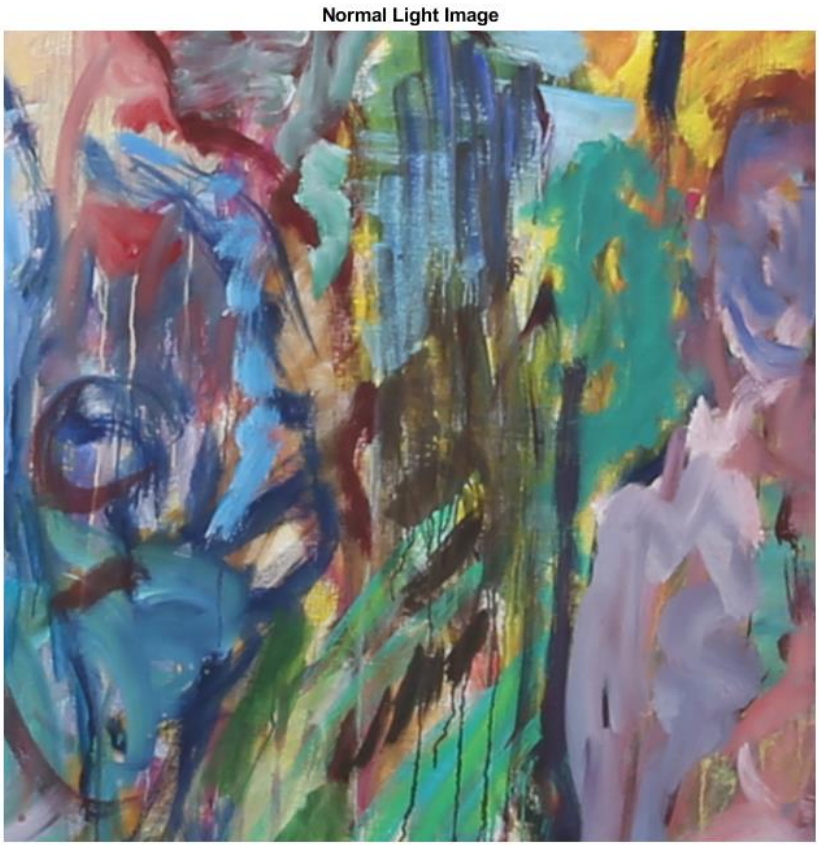


Figure 1.40: Normal light version of 1.37 figure

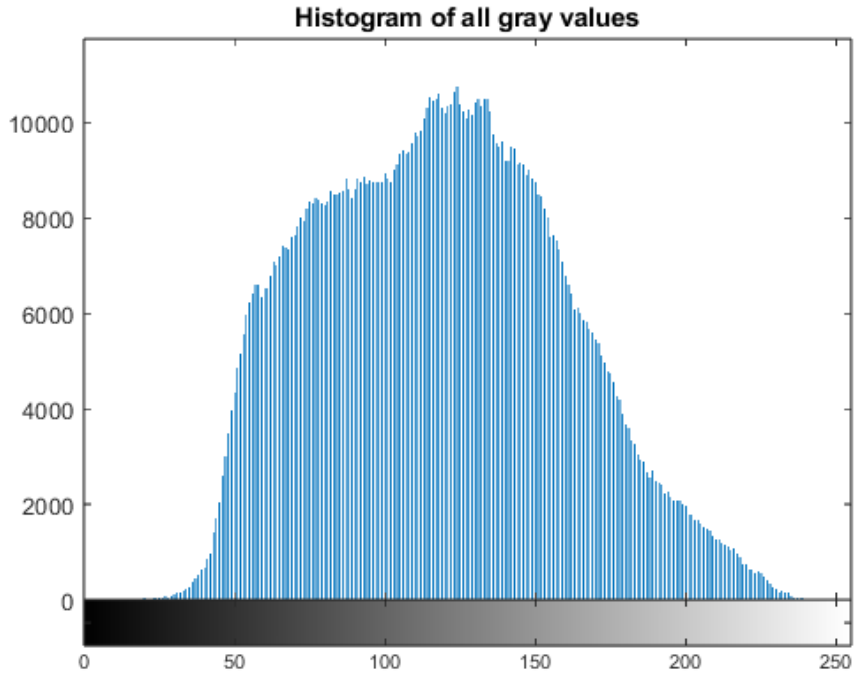


Figure 1.41: Histogram of all gray values of the figure 1.40

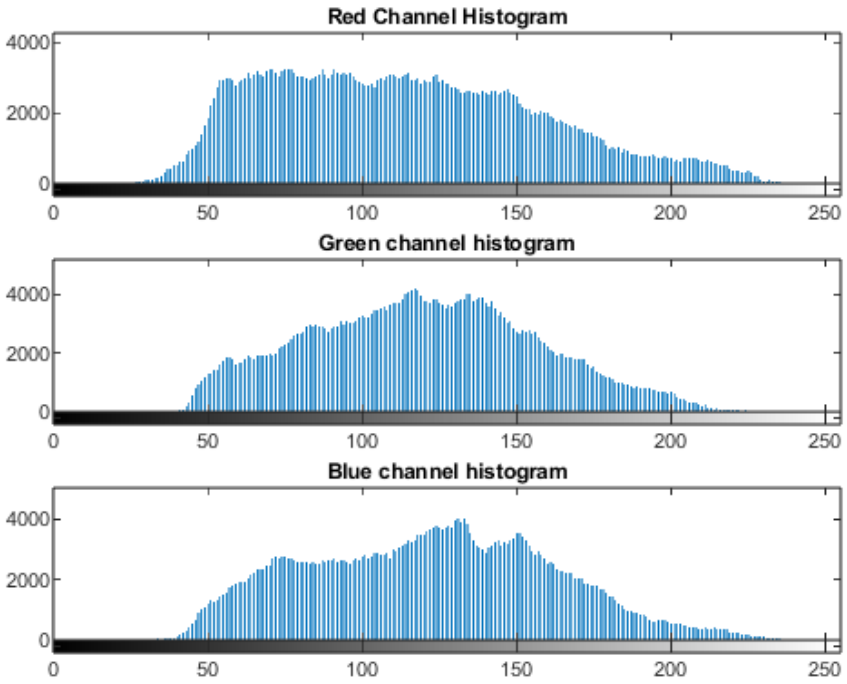


Figure 1.42: histogram of Figure 1.40 per channel

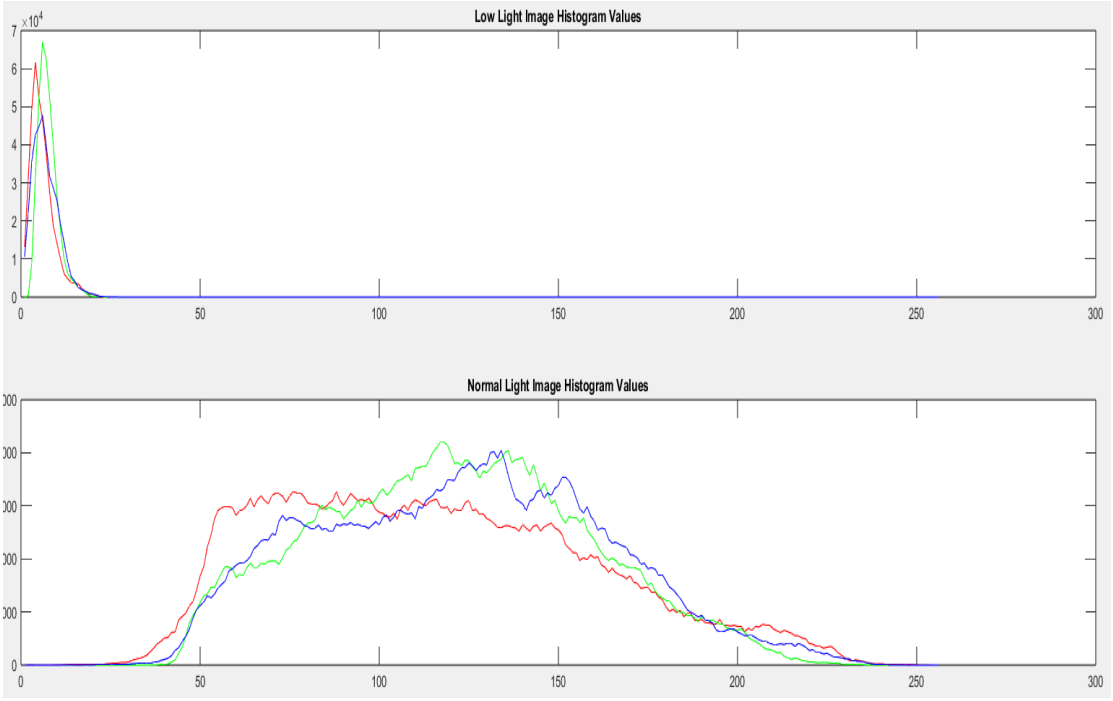


Figure 1.43: Comparison of histograms per channel

Darkness Level: 5.0

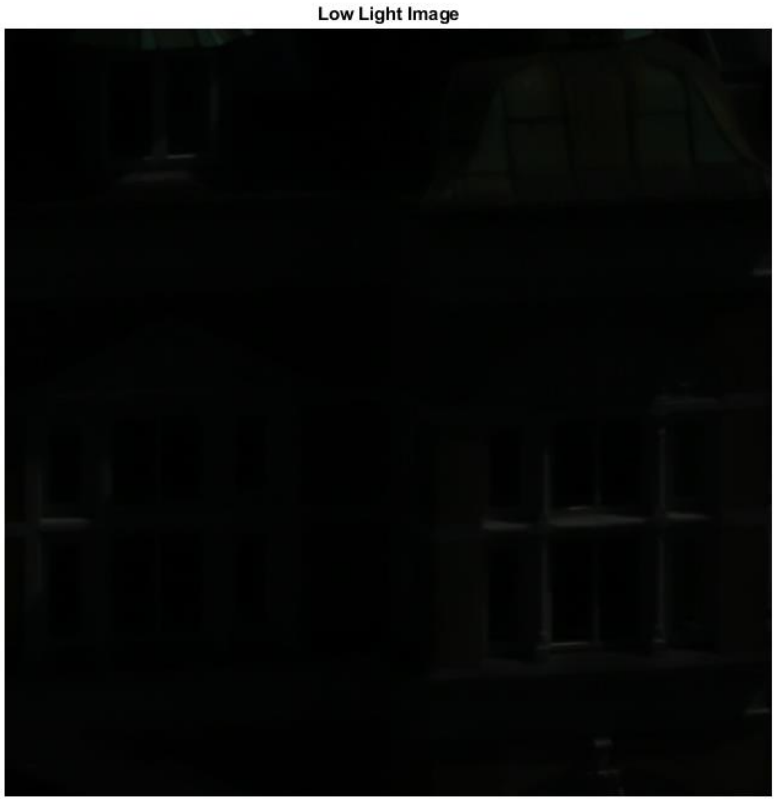


Figure 1.44: Random low light image from Darkness Level 5.0

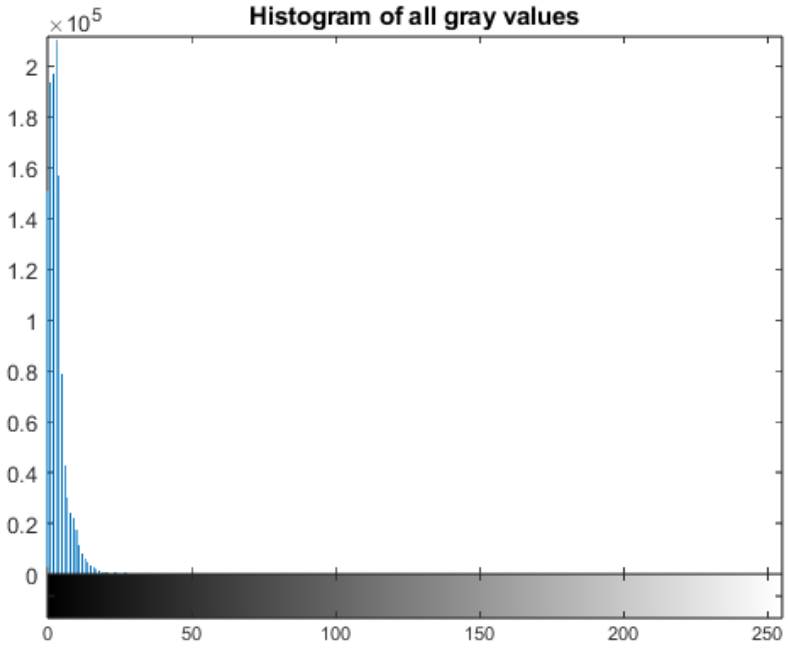


Figure 1.45: Histogram of all gray values of the figure 1.44

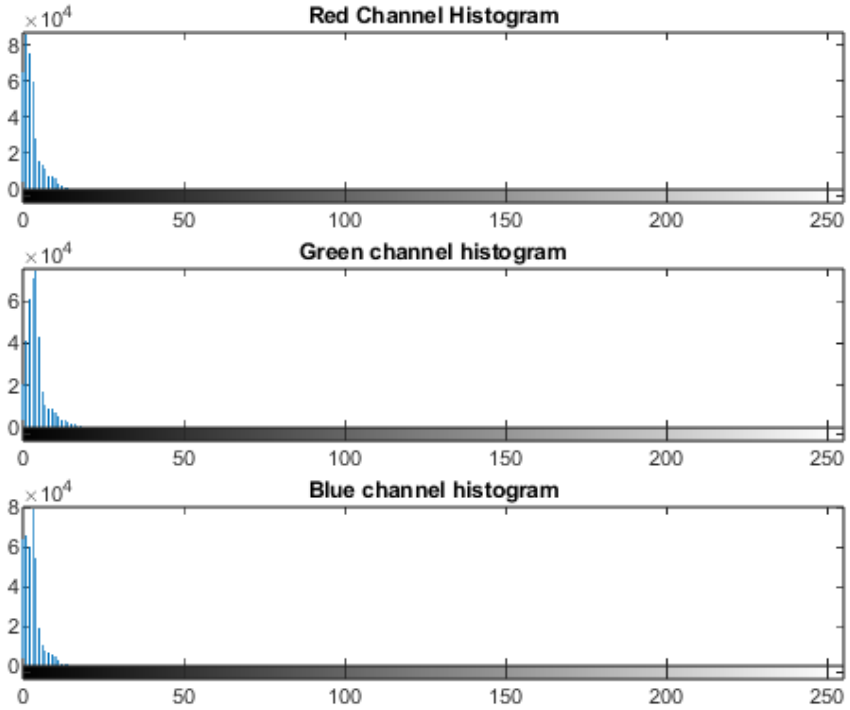


Figure 1.46: histogram of Figure 1.44 per channel

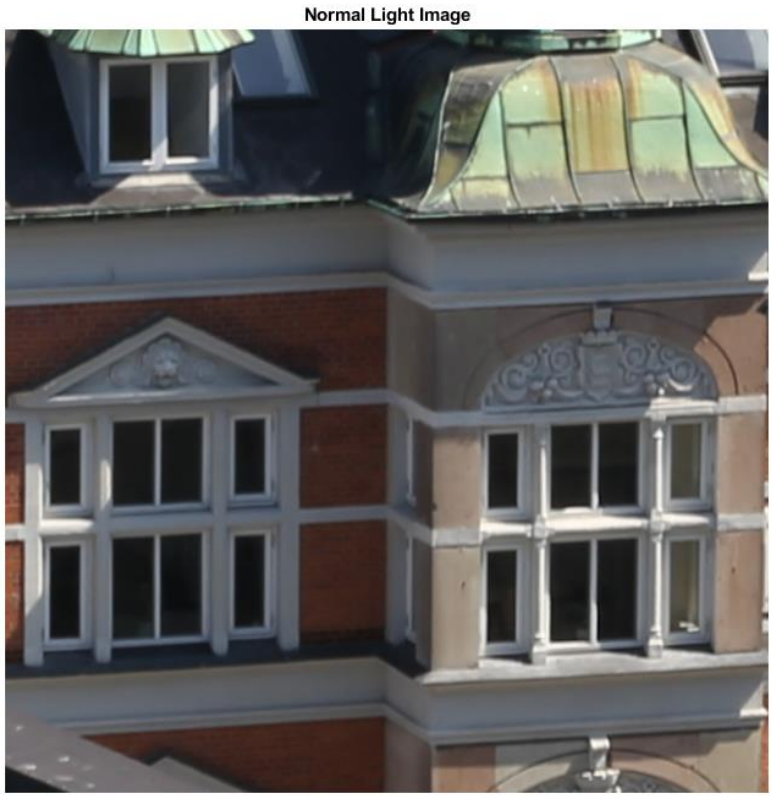


Figure 1.47: Normal light version of 1.44 image

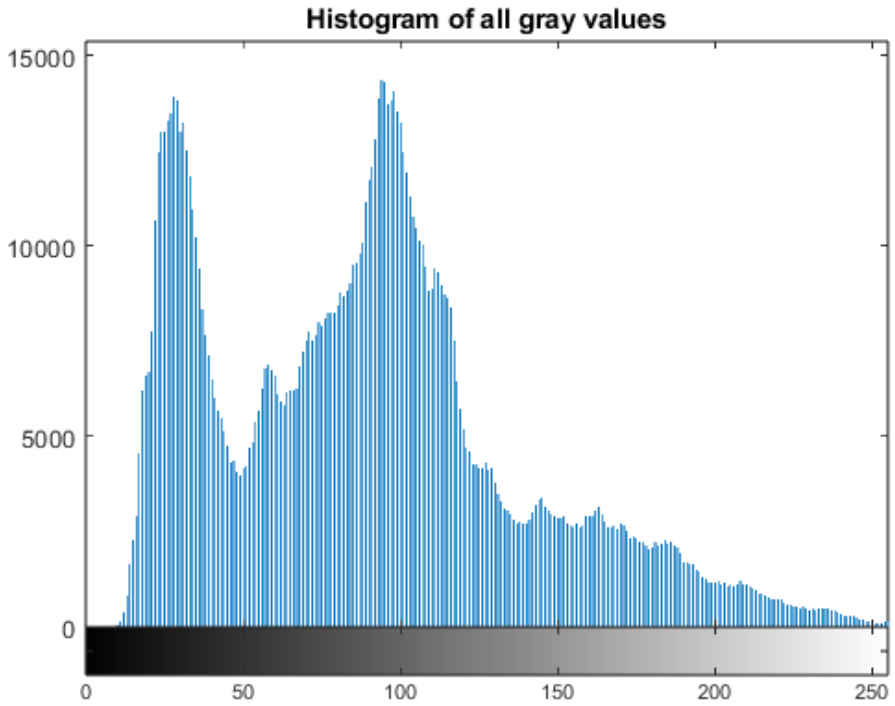


Figure 1.48: Histogram of all gray values of the figure 1.47

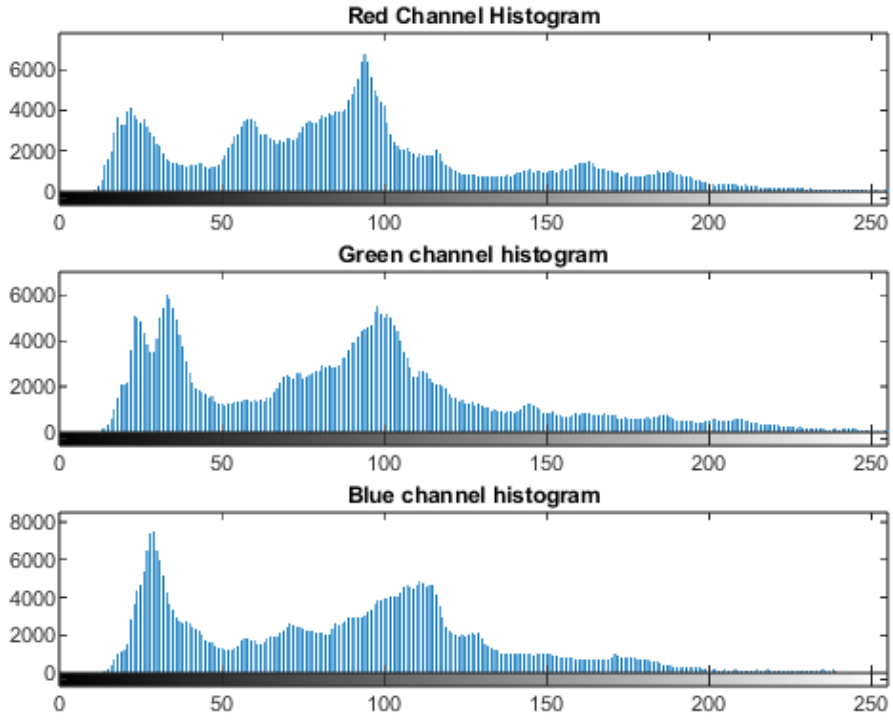


Figure 1.49: histogram of Figure 1.47 per channel

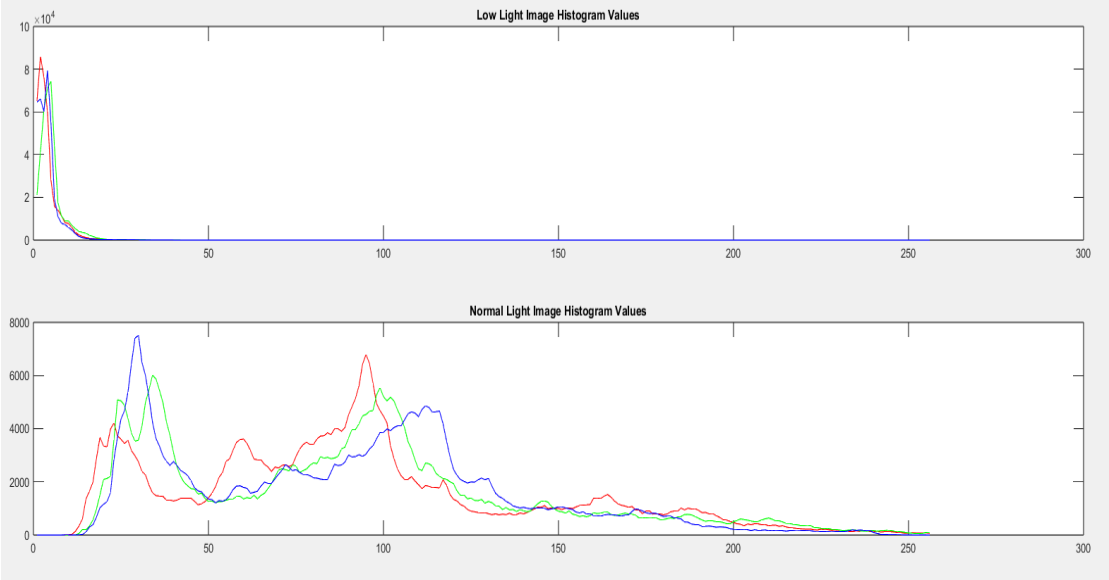


Figure 1.50: Comparison of histograms per channel

In all cases of darkness level one can see, from the histograms, that the pixel values for all color channels are concentrated in the left part of the histogram due to the low light conditions. As the level increases, this accumulation on the left side becomes more pronounced leading to increasingly darker images. Moreover, for all color channels the difference between maximum and minimum pixel value is very small, which confirms that low light images show low contrast. It is also worth commenting that the differences between pixel values between channels are very small. This leads to not being able to distinguish color and introduces color distortions. We see this become more pronounced at higher levels, with levels 4.5 and 5.0 being so dark that we can't make out any color at all. As a little spoiler note that this will be seen in the continuation of the thesis, where the algorithms to be applied will have a very difficult time restoring the color information for these levels.

Regarding the brightness values table 1.1 presents these values of the average brightness for each darkness level of the data set that will be used. These values have been calculated with the help of MATLAB software. In addition, a diagram of these values per partition of the data set (train, test, validation, see the description of the set below) is presented, from where the decreasing trend of the values as the Darkness Level increases can be confirmed.

Darkness Level	Train	Validation	Test	Total
3.0	16,7090	16,8166	16,7994	16,7750
3.5	11,7838	11,8761	11,8420	11,8340
4.0	8,2093	8,3013	8,2430	8,2512
4.5	5,6584	5,7407	5,6592	5,6861
5.0	3,9589	3,9634	3,8872	3,9365

Table 1.1: Brightness per data set per darkness level

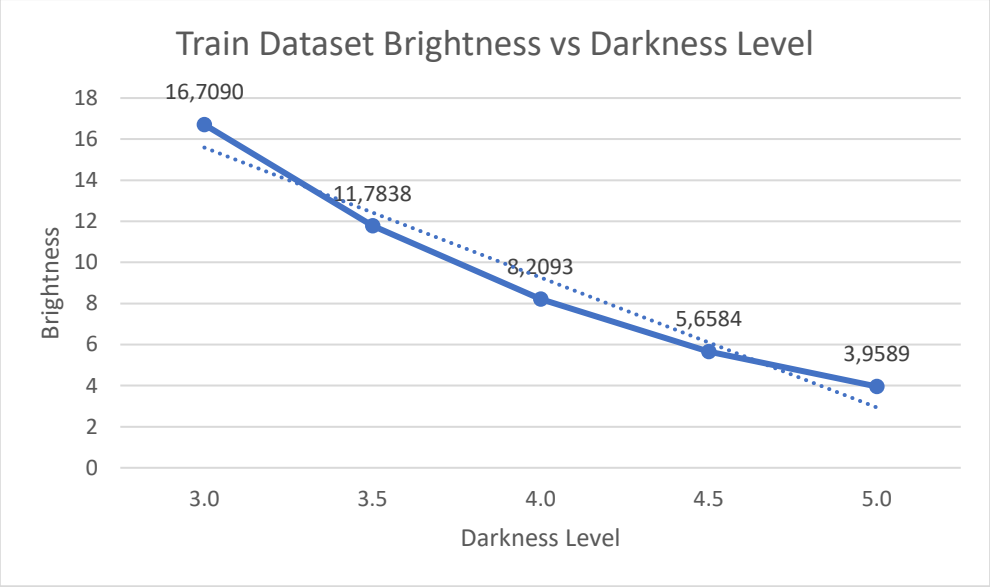


Figure 1.51: Brightness vs Darkness level for low light train dataset

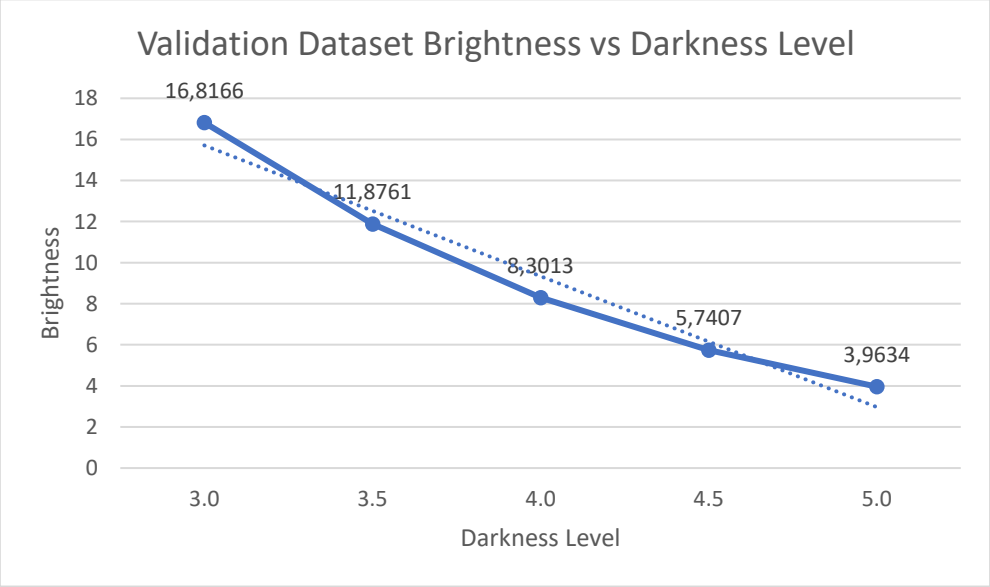


Figure 1.52: Brightness vs Darkness level for low light validation dataset

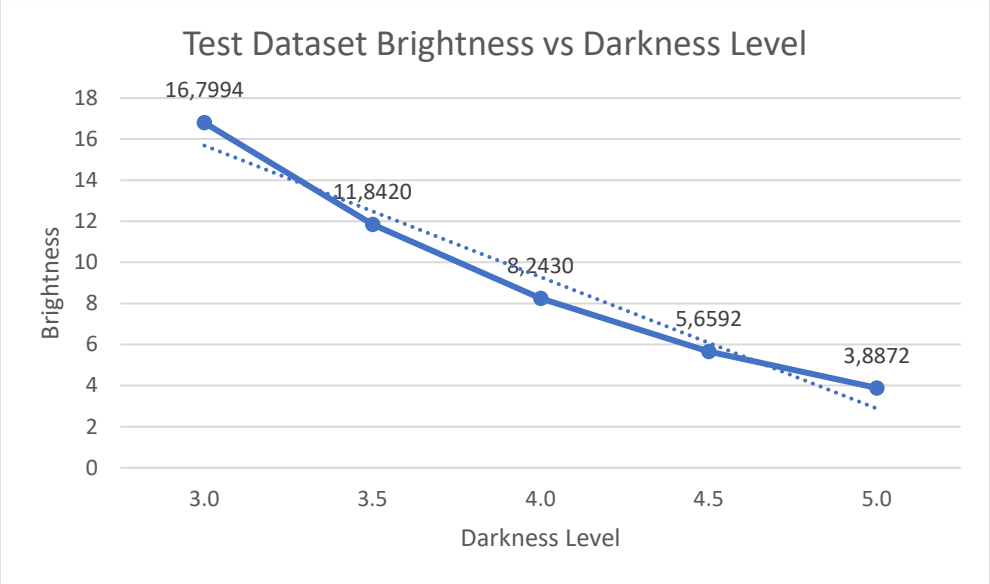


Figure 1.53: Brightness vs Darkness level for low light test dataset

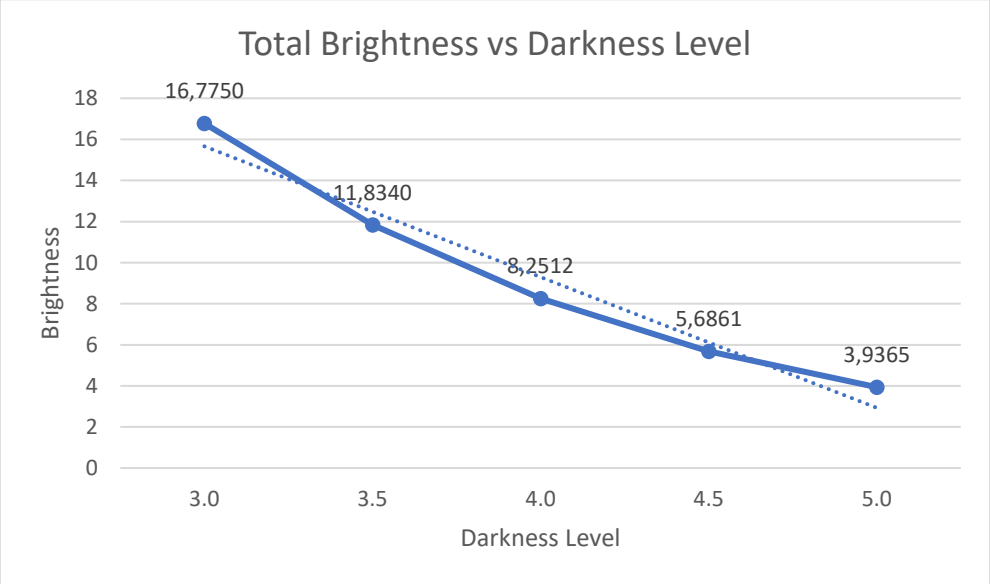


Figure 1.54: Brightness vs Darkness level for the whole low light dataset

In contrast to what have been mentioned so far, in ground truth images the pixel values are spread over the entire width of the histogram, leading to high contrast values and images rich in visual information. The differences between pixel values between channels are large enough to provide the user rich color information. These features are what the algorithms will try to restore in the low light images, in the continuation of the work. Each algorithm will be applied to the RELLISUR dataset, which is described immediately below.

RELLISUR Dataset

In [17] the authors present a data set that includes real low light low resolution images and the corresponding Normal light high resolution ground truth images, hence the name of the set, Real Low-Light Image Super Resolution. Their main purpose is to study Low Light Image Enhancement and Super Resolution together and not separately as is usually done. For this purpose, the authors collected images of different resolutions for the same static scene, varying the focal length of the camera, while at the same time they also collected images of different low light levels by gradually reducing the exposure time. Before proceeding to the detailed description of the data set, let us comment that due to the change in the focal length, misalignment of the image pairs may occur, something that is dealt with by a suitable post processing pipeline.

More specifically, the set consists of a sequence of Normal Light images of static scenes, scaled x1, x2 and x4, together with 5 low light images scaled x1, one for each darkness level. The scales express the resolution of the image, and since doubling the focal length leads to a doubling of the scale, images of different resolutions are obtained using focal lengths 70mm (x1), 140mm (x2) and 280mm (x4). The x1 images are 625x625 pixels, x2 are 1250x1250 pixels and x4 are 2500x2500 pixels. For the different brightness levels the Exposure Value is used, which is calculated as $log_2\left(\frac{N^2}{t}\right)$, where N is the f-stop number of the lens and t is the exposure time. A reduction of -1.0EV corresponds to half the exposure time, for constant N. This way images are taken for different values of under exposure, using exposure values -2.5,-3.0,-3.5,-4.0,-4.5, -5.0EV from the camera's auto exposure setting. Based on this, the Darkness Levels mentioned above are defined as 2.5, 3.0, 3.5, 4.0, 4.5 and 5.0 (here we do not use 2.5 for reasons that will be mentioned later). As the Darkness Level increases, the images get darker with a narrow gray value range, the pixel values are concentrated close to 0 and the average brightness of the set decreases, as can be seen from figures 1.16 – 1.54. For the sake of completeness, a sequence of such images is listed, using x1 scale images and darkness level 3.0-5.0, since we will deal with them in the rest of the thesis.

3.0 3.5 4.0

4.5 5.0 Normal Light

The set consists of 850 such image sequences, and if we take into account all resolution scale levels and all darkness levels, 12750 pairs of LLLR - NLHR images are obtained.

It was mentioned above that changing the focal length can lead to misalignment of the image pairs. In addition to this, other external factors can affect the final result, for example during shooting the wind may shake the camera leading to motion blur. For this reason, the authors apply a post-processing pipeline to the images they have collected. First, they check the collected images one by one to discard those that are out of focus, incorrectly exposed or contain moving objects. Then, using appropriate software [18] they remove chromatic aberration and lens distortion. At this point it should be commented that since it is difficult to remove the distortion from the corner areas of each image, they apply cropping of the x4 NLHR images to a size of 2500x2500 pixels. To deal with the problem of misalignment introduced due to a change in focal length, they register all images to match the x4 NLHR image of each sequence. They first detect and match SURF features [19] between the x1 and x4 NL images, using a downsampled version of the x4 image as the target. The coordinates they

calculated from the above step are used to calculate a homography through MSAC [20] and with the translation parameters calculated they align the x1 LL and NL images to the x4 NLHR reference image. The same procedure is applied for x2 images. Figure 1.55 shows a summary of the post-processing pipeline.

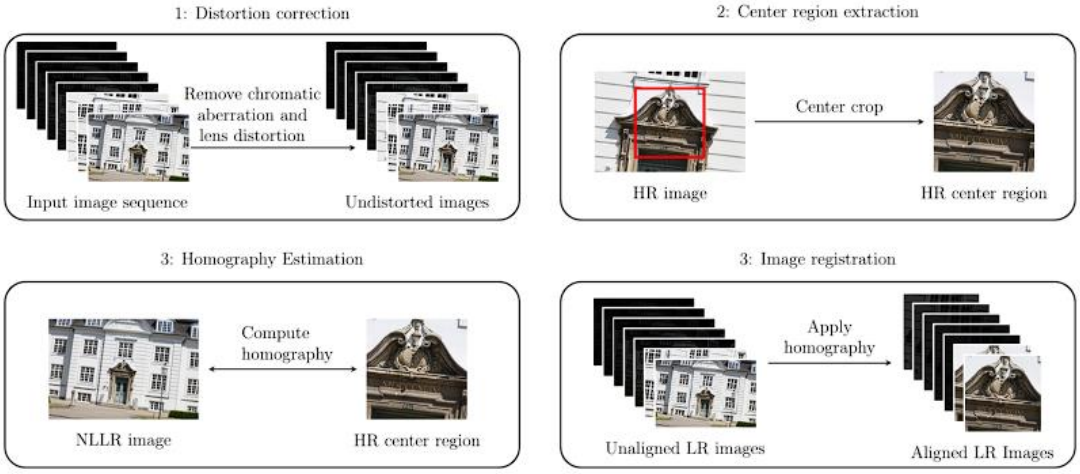


Figure 1.55: Summary of the post-processing pipeline [17]

Finally, let's comment on some statistics of the images of the data set. As it can be seen from figures 1.16 to 1.50, listed above, the NL images are characterized by sufficient exposure and have clear details and sharpness, with their histogram evenly distributed throughout the gray range. On the contrary, the LL images have very small contrast and intense color distortion, something that becomes more intense as the darkness level increases, while the values of their histogram are concentrated in the left part of it, at values mainly lower than 50, making the problem of LLIE even harder.

1.5 LLIE: Techniques and Evaluation

As mentioned above, these types of images cannot be fed to computer vision systems, because the low brightness, strong color distortion and the accompanying noise will lead to unreliable results. To deal with this problem there are two approaches. The first is the improvement of the image acquisition system hardware, using low light circuits [15]. Such systems use high performance charge-coupled devices or complementary

metal-oxide-semiconductors, so their technology and manufacturing method are extremely complex [21]. Although such cameras have been released on the market, we specifically mention the Nocturn XL by Photonis[5], they are not used in everyday applications because their cost is particularly high.

Figure 1.56: Nocturn XL low light camera from Photonis[6]

The second and preferable solution is the development of digital image processing algorithms to improve low light images. This area of research is called Low Light Image Enhancement (LLIE) and has the main purpose of improving the contrast of such images, restoring the brightness and preserving/enhancing the visual information of LL images, while avoiding noise enhancement, so that they can then be used in optical systems such as those mentioned above. The purpose of this thesis is the presentation, implementation and evaluation of such algorithms, using the RELLISUR data set.

The images used on a daily basis are mostly colored, i.e. they consist of the three color channels Red, Green and Blue, so the algorithms that will be studied are designed for colored image enhancement. This can be done in two approaches. The first approach is to use the RGB color space directly

[5] https://www.adept.net.au/cameras/photonis/nocturnXL.shtml
[6] https://www.photonis.com/products/nocturn-xl

and apply the respective algorithm to each color band separately, following a grayscale image enhancement process, as shown in the figure below.

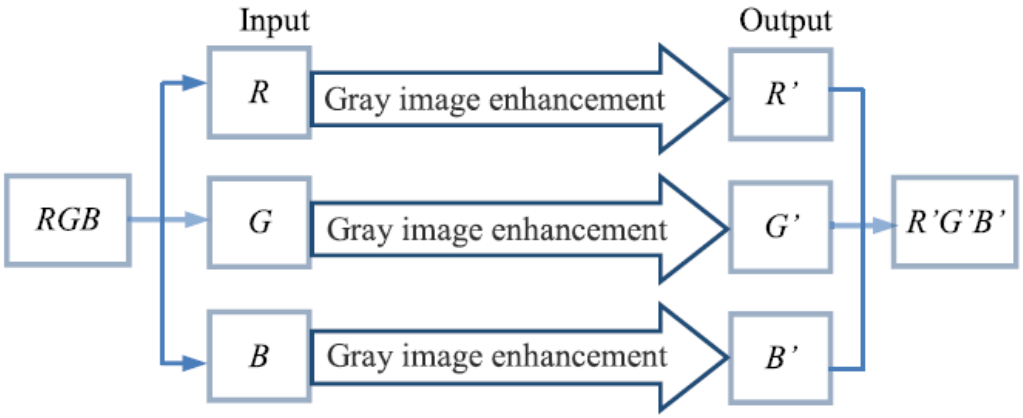


Figure 1.57: LLIE pipeline in the RGB color space [15]

The second approach is to use a different color space, like for example HSI (Hue, Saturation and Intensity). In this case the brightness component I and the Saturation component S, are extracted and enhanced separately, while keeping the Hue component constant, as shown in figure 1.58. This approach has the advantage of preserving the color information contained in H, but is obviously more complex.

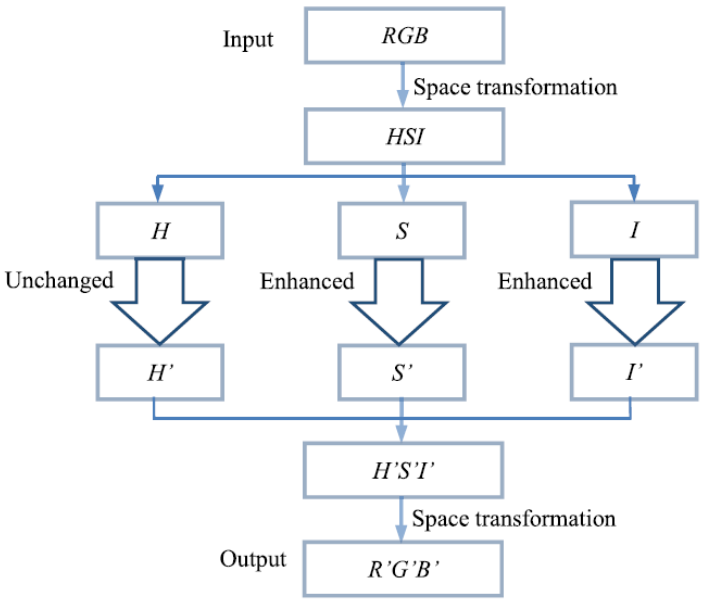


Figure 1.58: LLIE pipeline in the HSI color space [15]

In the continuation of the work the first approach will be used, as it is simpler and can lead to faster results helping the real time application of such algorithms.

Regarding the algorithms that will be developed in the following chapters, they can be categorized into the following two major categories:

Learning free algorithms

These are digital image processing algorithms, which do not involve learning using the ground truth images, but directly process the LL image using some sort of mathematical operation. We mention by name the classic algorithms that will be developed during the work, and they will be described in more detail in the respective chapter.

· Linear Stretching

· Gamma transformation

· Logarithmic transformation

· Histogram equalization

· Single Scale Retinex

· Multi Scale Retinex

· Defogging/Dehaze

Machine learning – deep learning based algorithms

These are algorithms that use the ground truth images to learn features based on which they will enhance the low light image. Here we will use deep learning paradigm, which essentially is a black box that at its input we give it the low light image and this in turn gives us the Normal Light image at the output. In particular we will use an architecture called Low Light CNN (LLCNN) which will be described in its respective chapter.

The above algorithms will be applied per darkness level so that we can compare their performance on increasingly darker images. But since the authors have put all the images in one folder, we split them into separate folders, using the script presented at the end of the chapter. Furthermore, for reasons that will be explained in a later chapter, the data set must be divided into training set, validation set and test set. This has already been done by the authors. Finally, since we are only interested in the LLIE, we

will only use the LL images and the corresponding x1 NL ground truth images. The distribution of the data is shown in the next table.

Darkness Level	Train	Validation	Test	Total	Training(%)	Validation(%)	Test(%)
2.5	181	10	16	**207**	87%	5%	8%
3.0	722	43	85	**850**	85%	5%	10%
3.5	722	43	85	**850**	85%	5%	10%
4.0	722	43	85	**850**	85%	5%	10%
4.5	718	43	84	**845**	85%	5%	10%
5.0	541	33	69	**643**	84%	5%	11%
Total	**3606**	**215**	**424**	**4245**	**85%**	**5%**	**10%**

Table 1.2: Separation to training set, validation set and test set of RELLISUR

The division that has been made is 85% training set, 10% test set and 5% validation set. In the continuation of the work, the 2.5 darkness level will not be used because it has few images compared to the other levels.

As mentioned above we will apply these algorithms by darkness level so that we can compare their performance. To do this, however, some evaluation metrics should be defined from which the performance of each algorithm will be derived, so that we have some number to make comparisons. The metrics that will be used next can be divided into two large categories depending on whether they need a ground truth image or not. These two categories are i) Full reference evaluation metrics, in which to calculate the value of the metric we need reference images, and ii) No-Reference evaluation metrics, which do not need ground truth images. The metrics we will use are presented below.

Full Reference Evaluation Metrics

Let $I_{en}(i,j)$ be the image obtained after LLIE of an LL image, and $I_{gt}(i,j)$ the ground truth image, with i=1,…,M και j=1,…,N. Then the following evaluation metrics can be defined:

Mean Square Error (MSE)

It expresses the deviation of the enhanced image from the ground truth and is calculated as follows:

$$MSE = \frac{1}{M * N} * \sum_{i=1}^{M} \sum_{j=1}^{N} \left[I_{gt}(i,j) - I_{en}(i,j) \right]^2$$

This definition can be extended to RGB images as well

$$MSE_{RGB} = \frac{1}{3} [MSE_{red} + MSE_{green} + MSE_{blue}]$$

Based on the definition we understand that a small value of MSE implies a greater similarity between enhanced and ground truth image.

Peak signal-to-noise ratio (PSNR)

The PSNR is calculated as follows:

$$\text{PSNR} = 10 * \log_{10} \frac{MAX_{I_{en}}^2}{MSE}$$

Where $MAX_{I_{en}}$ is the maximum gray value that a pixel can take, i.e. 255 in our case. This is the most commonly used metric for evaluating a denoising method. From the definition we understand that the higher the value of PSNR the more similar the enhanced and ground truth images are.

Structural Similarity Index Metric (SSIM)

The two metrics we mentioned above simply calculate an error between the two images, without taking into account the characteristics of the human visual system. For this reason in [22] SSIM was proposed where it takes into account the structural similarity of the two images. Specifically, the evaluation is done taking into account the luminance $l(I_{en},I_{gt})$, contrast $c(I_{en},I_{gt})$, and structure $s(I_{en},I_{gt})$ between the two images. These three measures are combined in the final SSIM value:

$$SSIM = F[l(I_{en}, I_{gt}), c(I_{en}, I_{gt}), s(I_{en}, I_{gt})]$$

The smallest value this index can take is 0, and the largest 1, and the closer to 1 the value is, the more similar the two images are.

Based on [22] the functions used in the above expressions are:

$$l(x,y) = \frac{2\mu_x\mu_y + C_1}{\mu_x^2 + \mu_y^2 + C_1}$$

$$c(x,y) = \frac{2\sigma_x\sigma_y + C_2}{\sigma_x^2 + \sigma_y^2 + C_2}$$

$$s(x,y) = \frac{\sigma_{xy} + C_3}{\sigma_x\sigma_y + C_3}$$

Where $C_1 = (K_1 L)^2$ a small constant for the case that the $\mu_x^2 + \mu_y^2$ is close to zero, L is the range of the pixel values (here 255) and $K_1 << 1$ a small constant. Also $C_2 = (K_2 L)^2$, with $K_2 << 1$, and $C_3 = \frac{C_2}{2}$. The μ_x, σ_x are the average brightness and the standard deviation (as a measure of contrast), and are defined in the known manner:

$$\mu_x = \frac{\sum_{i=1}^{N} x_i}{N}$$

$$\sigma_x = \left(\frac{\sum_{i=1}^{N}(x_i - \mu_x)^2}{N-1}\right)^{\frac{1}{2}}$$

While the σ_{xy} is the correlation coefficient and is defined as:

$$\sigma_{xy} = \left(\frac{1}{N-1}\right)\sum_{\iota=1}^{N}(x_i - \mu_x)(y_i - \mu_y)$$

Based on these it follows that the final form of the SSIM metric is:

$$SSIM(x,y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)}$$

and is also the relation implemented by the MATLAB function.

For the evaluation metrics with reference a special function in MATLAB is built, which will be described at the end of this chapter.

No Reference Evaluation Metrics

For an image I(i,j), with size MxN the following metrics can be defined without a ground truth image:

Mean Value (MV)

This is the average value of the pixel values of the image, i.e. its brightness. The smaller this average value, the darker the image, and correspondingly, the higher the average value, the brighter the colors. The formula by which it is calculated is:

$$MV = \frac{1}{M * N}\sum_{i=1}^{M}\sum_{j=1}^{N} I(i,j)$$

It can be generalized to RGB images as follows:

$$MV_{RGB} = \frac{MV_{Red} + MV_{Green} + MV_{Blue}}{3}$$

Standard Difference (STD)

It is the variation of pixel values around the mean value, and thus can be a measure of contrast. A large value corresponds to an image with high contrast, so more visual information. The STD for a gray scale image is calculated by the formula:

$$STD = \sqrt{\frac{\sum_{i=1}^{M}\sum_{j=1}^{N}[I(i,j) - MV]^2}{M * N}}$$

The formula can also be generalized to RGB images, in the known way:

$$STD_{RGB} = \frac{STD_{Red} + STD_{Green} + STD_{Blue}}{3}$$

Blind/Referenceless Image Spatial QUality Evaluator (BRISQUE)

This is a metric based on the fact that undistorted images exhibit certain statistical properties that change when distortions are introduced into the image. Characteristically, we mention that the power spectrum of an image without distortions is a function of the frequency f, and has the form $\frac{1}{f^{\gamma}}$, with γ being an exponent, the value of which varies little from image to image [23]. Images containing distortions will deviate from these natural statistics, so in this way the quality of an image can be quantified. Based on this we understand that the lower the value of BRISQUE, the closer to undistorted images the image under consideration is, so the better its visual quality is. To properly model natural statistics, however, a function must be trained on a huge set of undistorted images. In the context of the thesis, we use the corresponding MATLAB function[7].

Naturalness Image Quality Evaluator (NIQE)

This is a metric based on Natural Scene Statistics [24]. Specifically, the authors extract features from a set of natural images, and use them to train/learn a multivariate gaussian distribution. Then, for each image, the quality of which they want to calculate, they extract NSS characteristics and based on these they fit a multivariate gaussian distribution. As image quality, they define the distance between the gaussian distribution as derived from the image, and the gaussian distribution they trained in the previous step. Based on this we understand that the lower the value of the

[7] https://ch.mathworks.com/help/images/ref/brisque.html

NIQE metric, the higher the quality of the image under consideration. In the context of this thesis, we use the ready-made function of MATLAB[8].

Experiments and evaluation of results

It was mentioned at the beginning of the subsection that we will apply a variety of low light image enhancement techniques. The process of experimentation and evaluation of the results, which we will follow for each technique, consists of the following steps:

· Each technique/algorithm will be applied to each darkness level separately so that the performance of the algorithm in comparison to how dark an image is can be evaluated. At this point, let us mention that all techniques will be applied exclusively to the RGB color space for reasons of simplicity. In addition, we will not use the darkness level 2.5, because it has few images compared to the rest darkness levels.

· For the result that will be obtained for each darkness level for each algorithm, the value of all the metrics we mentioned will be calculated, for each image, and we will save the result in an excel file. From this excel file a table will be constructed that will contain the minimum, maximum and average value of each metric, per darkness level, as well as the corresponding values for the original dataset. Based on this table we will make line charts of PSNR and SSIM.

· The original, ground truth and the image resulting from each method will be depicted, corresponding to the smallest and largest experimental PSNR, as well as their histograms for comparison. We will also do this for a random image, per darkness level, so that we can have a more complete view of the results (depending on the method we are studying, details in the next 2 chapters).

Here it should be mentioned that we will do this for each technique/algorithm separately, but in the last chapter we will combine all these results in order to compare the techniques with each other, commenting on the respective results.

In the following subsection the scripts used/referred to during this chapter will be presented, and briefly describe the steps that they follow.

[8] https://ch.mathworks.com/help/images/ref/niqe.html

1.6 Scripts Used

We mentioned above that in [17] the authors provide all the images in one folder and that we should split the images by darkness level (for each of the train/test/validation classes). This was done with the help of the script shown below.

```
1   close all; clear; clc; format compact;
2
3   %FOR CREATING LOW LIGHT TRAINING SET
4   %images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\RELLISUR-Dataset\Train\LLLR');
5
6   %FOR CREATING LOW LIGHT VALIDATION SET
7   %images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\RELLISUR-Dataset\Val\LLLR');
8
9   %FOR CREATING LOW LIGHT TEST SET
10  images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\RELLISUR-Dataset\Test\LLLR');
11
12  data_size = size(images.Files);
13  numfiles = data_size(1);
14
15  for index=1:numfiles
16      image_name = char(images.Files(index));
17      if image_name((end-6):(end-4)) == '5.0' %this will change depending the darkness level
18          img = readimage(images,index);
19          %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\5.0\LL-5.0\',image_name((end-12):(end-8)),'.png'];
20          %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\5.0\LL-5.0\',image_name((end-12):(end-8)),'.png'];
21          file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\5.0\LL-5.0\',image_name((end-12):(end-8)),'.png'];
22          imwrite(img,file_path);
23      end
24  end
25
```

Figure 1.59: Script for creating datasets per darkness level for LL images

In lines 3 to 10 we read the data set we are interested in. Because we have train/test/validation each time we will have 2 of the 3 data sets commented. For each set case, we run the script 6 times, once for each darkness level, changing the value in darkness level on lines 17,19,20,21. Essentially we run a loop through all train/test/split images and if their name contains the darkness level we are interested in, then we save them in the appropriate folder.

We follow a similar procedure for the ground truth images, as shown in figure 1.60. At the end of this process we have at our disposal train/test/validation datasets, per darkness level.

```
close all; clear; clc; format compact;

%FOR CREATING NORLMAL LIGHT TRAINING SET
%NL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\RELLISUR-Dataset\Train\NLHR\X1');
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\5.0\LL-5.0');

%FOR CREATING NORMAL LIGHT VALIDATION DET
%NL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\RELLISUR-Dataset\Val\NLHR\X1');
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\5.0\LL-5.0');

%FOR CREATING NORMAL LIGHT TEST DET
NL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\RELLISUR-Dataset\Test\NLHR\X1');
LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\5.0\LL-5.0');

data_size = size(NL_images.Files);
numfiles = data_size(1);

low_size = size(LL_images.Files);
LLnum = low_size(1);

for index=1:numfiles
    nl_name = char(NL_images.Files(index));
    for i=1:LLnum
        ll_name = char(LL_images.Files(i));
        if nl_name((end-8):(end-4)) == ll_name((end-8):(end-4))
            img = readimage(NL_images,index);
            %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\5.0\NL-5.0\',nl_name((end-8):(end-4)),'.png'];
            %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\5.0\NL-5.0\',nl_name((end-8):(end-4)),'.png'];
            file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\5.0\NL-5.0\',nl_name((end-8):(end-4)),'.png'];
            imwrite(img,file_path);
            break
        end
    end
end

```

Figure 1.60: Script for creating datasets per darkness level for NL images

In this case we read both NL images and LL images per darkness level, and compare with the file names (each image has a unique name). By doing a loop through the normal light images, if the LL dataset contains an image with the same name as that of the respective NL image, then we save this normal light image in an appropriate folder. So now we have ground truth images per darkness level, and by extension image pairs that can be used for evaluation with reference or training of a deep learning model.

In subsection 1.4 we showed a random image from each darkness level along with the corresponding ground truth, their histograms as well as a comparison of these histograms. This was done using the following scripts. In figure 1.61 we see the final script that produces the results of section 1.4. In lines 4-16 we read the LL and NL data sets, and since we have 5 different darkness levels, we will run the script 5 times, each time having a different level uncommented. In lines 18-23 we find the size of each data set, and generate a random integer in the range 1 to the size of the set. Finally, we read the LL and NL version of the image corresponding to this integer and display them along with their histograms using the HISTOGRAMS function shown in Figures 1.62 and 1.63. This function takes as input the LL and NL images, first displays the LL image, the histogram for all gray values of the pixels, regardless of channel, and then

the histogram per channel using the sub plots feature. Then it does the same process for the NL image, and finishes by calling the histogram_comparison function, to compare their histograms. This function is shown in figure 1.64, takes as input the LL and NL images and initially calculates the histogram of each channel separately. With this information, and using the sub plot feature, it displays a figure containing the images as well as their histograms per channel.

```
close all; clear; clc; format compact;


%loading low light data
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\3.0\LL-3.0');
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\3.5\LL-3.5');
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\4.0\LL-4.0');
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\4.5\LL-4.5');
LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\5.0\LL-5.0');

%loading normal light data
%NL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\3.0\NL-3.0');
%NL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\3.5\NL-3.5');
%NL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\4.0\NL-4.0');
%NL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\4.5\NL-4.5');
NL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\5.0\NL-5.0');

%number of images
ims = size(LL_images.Files);
numOfImages = ims(1);

%generating a random number in the range [1,numOfImages]
idx = randi(numOfImages,1);


%low light random image
LL = readimage(LL_images,idx);
%normal light random image
NL = readimage(NL_images,idx);

%histograms figures
HISTOGRAMS(LL,NL)

```

Figure 1.61: Script to display random images and their histograms

```
function HISTOGRAMS(LL_image,NL_image)

%=========================================================================
% Function that prints the low and normal light images, their respective
% histogram of gray values and the histograms per channel. Needed for
% the thesis analysis.
% Author: Panagiotis Koutsaftis aivc21010
%=========================================================================

%====================================
%Histograms for the Low Light image
%====================================

%low light image
figure
imshow(LL_image)
title('Low Light Image')

%histogram of gray values
figure
imhist(LL_image)
title("Histogram of all gray values")
xlim([0,255])
ylim([0,max(imhist(LL_image))+1000])


%histogram per channel
figure
subplot(3,1,1)
imhist(LL_image(:,:,1));
title("Red Channel Histogram")
xlim([0,255])
ylim([0,max(imhist(LL_image(:,:,1)))+1000])
subplot(3,1,2)
imhist(LL_image(:,:,2));
title('Green channel histogram')
xlim([0,255])
ylim([0,max(imhist(LL_image(:,:,2)))+1000])
subplot(3,1,3)
imhist(LL_image(:,:,3));
title('Blue channel histogram')
xlim([0,255])
ylim([0,max(imhist(LL_image(:,:,3)))+1000])

```

Figure 1.62: Function for histograms part 1

```
44
45  %====================================
46  %Histograms for the Normal Light image
47  %====================================
48
49  %low light image
50  figure
51  imshow(NL_image)
52  title('Normal Light Image')
53
54  %histogram of gray values
55  figure
56  imhist(NL_image)
57  title("Histogram of all gray values")
58  xlim([0,255])
59  ylim([0,max(imhist(NL_image))+1000])
60
61
62  %histogram per channel
63  figure
64  subplot(3,1,1)
65  imhist(NL_image(:,:,1));
66  title("Red Channel Histogram")
67  xlim([0,255])
68  ylim([0,max(imhist(NL_image(:,:,1)))+1000])
69  subplot(3,1,2)
70  imhist(NL_image(:,:,2));
71  title('Green channel histogram')
72  xlim([0,255])
73  ylim([0,max(imhist(NL_image(:,:,2)))+1000])
74  subplot(3,1,3)
75  imhist(NL_image(:,:,3));
76  title('Blue channel histogram')
77  xlim([0,255])
78  ylim([0,max(imhist(NL_image(:,:,3)))+1000])
79
80  %============================================
81  %         COMPARING THE HISTOGRAMS
82  %============================================
83  histogram_comparison(LL_image,NL_image)
```

Figure 1.63: Function for histograms part 2

```
function histogram_comparison(LL_image,NL_image)

%=========================================================================
% Function that prints the low light and normal light images, and their
% respective histograms per channel, needed for comparison.
% Author: Panagiotis Koutsaftis aivc21010
%=========================================================================
%histogram values array of each LL_image component
LL_red = imhist(LL_image(:,:,1));
LL_green = imhist(LL_image(:,:,2));
LL_blue = imhist(LL_image(:,:,3));

%histogram values array of each NL_image component
NL_red = imhist(NL_image(:,:,1));
NL_green = imhist(NL_image(:,:,2));
NL_blue = imhist(NL_image(:,:,3));

figure
%setting the size of the plot
set(gcf, 'Position', get(0,'Screensize'));
%plot of low light image
subplot(3,3,1)
imshow(LL_image)
title('Low Light Image');
%plot of normal light image
subplot(3,3,3)
imshow(NL_image)
title("Normal Light Image");
%histogram of low light image per channel
subplot(3,3,4:6)
plot(LL_red,'r')
hold on
plot(LL_green,'g')
plot(LL_blue,'blue')
title('Low Light Image Histogram Values')
hold off
%histogram of normal light image per channel
subplot(3,3,7:9)
plot(NL_red,'r')
hold on
plot(NL_green,'g')
plot(NL_blue,'blue')
title("Normal Light Image Histogram Values")
hold off
```

Figure 1.64: Function to compare histograms

Following, at section 1.4, we calculated the average brightness of the LL images, per darkness level and per partition. This was done with the help of the scripts shown in figures 1.65 and 1.66.

```
close all; clear; clc; format long;

%==========================================================================
%                           LOW LIGHT IMAGES
%==========================================================================

%Training Data
%--------------
%LL_images_train = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\3.0\LL-3.0');
%LL_images_train = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\3.5\LL-3.5');
%LL_images_train = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\4.0\LL-4.0');
%LL_images_train = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\4.5\LL-4.5');
LL_images_train = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\5.0\LL-5.0');

%Validation Data
%-----------------
%LL_images_val = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\3.0\LL-3.0');
%LL_images_val = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\3.5\LL-3.5');
%LL_images_val = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\4.0\LL-4.0');
%LL_images_val = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\4.5\LL-4.5');
LL_images_val = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\5.0\LL-5.0');

%Test Data
%----------
%LL_images_test = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\3.0\LL-3.0');
%LL_images_test = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\3.5\LL-3.5');
%LL_images_test = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\4.0\LL-4.0');
%LL_images_test = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\4.5\LL-4.5');
LL_images_test = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\5.0\LL-5.0');

[LL_Train,LL_val,LL_test] = Brightness(LL_images_train,LL_images_val,LL_images_test);
```

Figure 1.65: Brightness calculation for the LL dataset

In the script of figure 1.65, we read the data set for each partition (train, test and validation) of the LL dataset and using the Brightness function we calculate the average value of the brightness of each set. We repeat this process 5 times, once for each value of the darkness level. We store these values in an excel file, where we construct the table presented in section 1.4, as well as the corresponding line charts. The Brightness function we mentioned earlier, shown in figure 1.66, takes as input the data stores of each partition of the data set. For each datastore it calculates the average value of the brightness of all the images contained in it, and returns this result.

```
function [BR_Train,BR_Val,BR_Test] = Brightness(train_datastore,val_datastore,test_datastore)

%=========================================================================
% Function that calculates the mean brightness of a dataset. For our
% purpose we have train, test and validation datasets so we need the
% results for all 3 of them.
% Author: Panagiotis Koutsaftis
%=========================================================================
%TRAIN DATASTORE
%size of the dataset
files = size(train_datastore.Files);
numOfImages = files(1);

brs = zeros([1,numOfImages]);

for index=1:numOfImages
    img = readimage(train_datastore,index);
    brs(index) = mean(mean(mean(img)));
end

BR_Train = mean(brs);

%VALIDATION DATASTORE
%size of the dataset
files = size(val_datastore.Files);
numOfImages = files(1);

brs = zeros([1,numOfImages]);
for index=1:numOfImages
    img = readimage(val_datastore,index);
    brs(index) = mean(mean(mean(img)));
end

BR_Val = mean(brs);

%TEST DATASTORE
%size of the dataset
files = size(test_datastore.Files);
numOfImages = files(1);

brs = zeros([1,numOfImages]);
for index=1:numOfImages
    img = readimage(test_datastore,index);
    brs(index) = mean(mean(mean(img)));
end

BR_Test = mean(brs);
```

Figure 1.66: Brightness calculation function for datastore

Finally, we must also mention the functions which will calculate the quality metrics we described above. We have constructed 3 different functions, a function that calculates metrics with reference (figure 1.67), a function that calculates metrics with no reference (figure 1.68), and a function that combines the two previous functions (figure 1.69).

```
function [MSE,PSNR,SSIM] = Ref_QA(input_image,ref_image)

%========================================================================
% Function that calculates full reference quality metrics. It takes as
% input the (probably distorted) image and the respective reference images
% (some times called ground truth) and calculates and returns the MSE, the
% PSNR and the SSIM. For the MSE we are using a custom calculation and for
% PSNR and SSIM we are using the MATLAB functions.
% Author: Panagiotis Koutsaftis aivc21010
%========================================================================

%checking the size of the images
if size(input_image) ~= size(ref_image)
    error('The size of the images must be the same.')
end

%converting the images to double, needed for the calculations
double_input = double(input_image);
double_ref = double(ref_image);

%size of the images
[rows,columns,bands] = size(ref_image);

%calculating the MSE metric value
diff_value = (double_ref-double_input).^2;
sum_value = sum(diff_value,"all");
MSE = sum_value/(rows*columns*bands);

%calculating the PSNR metric value
PSNR = psnr(input_image,ref_image);

%calculating the SSIM metric value
SSIM = ssim(input_image,ref_image);
```

Figure 1.67: Function that calculates metrics with reference

The function presented at figure 1.67 takes as input an image (which here will have resulted from some LLIE method), as well as the corresponding ground truth image, and calculates the MSE, the PSNR and the SSIM. Before that, of course, it checks if the images have the same size, in case a mistake is made and images of a different size are entered. If the images are not the same size, then an error is thrown and the metrics are not calculated. For MSE we use a custom calculation, utilizing the vectorization capabilities of MATLAB, while for PSNR and SSIM, we use the default functions of MATLAB.

```
function [MV,STD,BR,NQ] = noRef_QA(input_image)

%=========================================================================
% Function that takes as input an image (probably distorted), and
% calculates no reference quality metrics. For Mean Value and STD we are
% using custom calculations and for BRISQUE and NIQE we are using the
% MATLAB functions.
% Author: Panagiotis Koutsaftis aivc21010
%=========================================================================

%converting the image to double for the calculations
double_image = double(input_image);

%size of the input image
[rows,columns,bands] = size(input_image);

%calculating the Mean Value metric
summ = sum(double_image,"all");
prod = rows*columns*bands;

MV = summ/prod;

%calculating the STD metric
sum_value = 0;
for band=1:bands
    for row=1:rows
        for column=1:columns
            prod_value = double_image(row,column,band)*(double_image(row,columns,band)-MV)^2;
            sum_value = sum_value + prod_value;
        end
    end
end
STD = sqrt(sum_value/(rows*columns*bands));

%calculating the BRISQUE metric
BR = brisque(input_image);

%calculating the NIQE metric
NQ = niqe(input_image);
```

Figure 1.68: Function that calculates metrics without a reference

The function presented at figure 1.68 takes as input an image, which here will have been obtained as the result of an LLIE method, and calculates quality metrics without a reference. Specifically, it calculates MV and STD, using a custom calculation, as well as BRISQUE and NIQE, using MATLAB built-in functions.

The above two functions are combined in the function shown in figure 1.69. This function takes as input the datastores that contain the ground truth images and the images that have been derived from the LLIE method. Then, for each LLIE datastore image, they calculate 7 metrics and store them in an array. This array has 7 columns, and each row corresponds to a datastore image. As we will see in the next chapter, the results of this function, for each method, will be saved in an excel file, which we will use for our analysis.

```
function assessment_array = assessment(GT_images,EXP_images)

%========================================================================
% Returns a matrix whos each column is a assessment metric. The columns
% of the matrix are of the form:
% MSE | PSNR | SSIM | MV | STD | BRISQUE | NIQE
% and each row corresponds to an image of the datastore inputed.
% The inputs are the Ground Truth images datastore (GT_images) and the
% results of the experiments (LLIE) images datastore.
% Author: Panagiotis Koutsaftis aivc21010
%========================================================================

%size of the dataset
files = size(GT_images.Files);
numOfImages = files(1);

%size of the assessment array
array_size = [numOfImages,7];
ass_array = zeros(array_size);

for index=1:numOfImages
    exp_image = readimage(EXP_images,index);
    ref_image = readimage(GT_images,index);
    %reference based metric
    [ass_array(index,1),...
        ass_array(index,2),...
        ass_array(index,3)] = Ref_QA(exp_image,ref_image);

    %no reference metrics
    [ass_array(index,4),...
        ass_array(index,5),...
        ass_array(index,6),...
        ass_array(index,7)] = noRef_QA(exp_image);
end

%returning the final result
assessment_array = ass_array;

```

Figure 1.69: Function to calculate metrics for each image of an image datastore

The rest of the thesis is structured as follows. In chapter 2 we will describe, develop and implement a set of classical algorithms, each time evaluating their performance. In chapter 3 we will study deep learning methods, specifically the LLCNN architecture. We will implement 3 different variations of the architecture, which we will evaluate one by one. Finally, in chapter 4 we will combine the results of these two chapters, comparing the methods with each other and making the final comments.

Chapter 2 – Classical Algorithms

In this chapter we will study classic algorithms, classic in the sense that no learning process is applied, just a procedure on all the gray values of the image. The algorithms we will use are the following:

· Linear Transformation

· Gamma Correction

· Log Correction

· Histogram Equalization

· Single Scale Retinex

· Multi Scale Retinex

· Defogging

We will dedicate a separate section to each algorithm, where we will describe the basic theory on which it is based, as well as the algorithm we developed to implement it. Then, after we have implemented this algorithm, we will do the evaluation process that we mentioned in the previous chapter, commenting in detail on the performance of the algorithm. Specifically, after we have applied the respective algorithm to all darkness levels, we will calculate the metrics for all images, per darkness level, and find the average, minimum and maximum value. We will put this information in a table, which will help us see the performance of the algorithm, compared to increasing the darkness level. In addition, we will also visualize some of these results, making line charts of PSNR and SSIM, having the darkness level on the horizontal axis and the average/minimum/maximum value of these metrics on the vertical axis.

In addition to the comparison per darkness level, however, we should also compare with the original images. For this reason, we list the following tables, which contain the values of all metrics per darkness level, for the original LL images.

Training Set

MSE					
Level	3.0	3.5	4.0	4.5	5.0
MIN	694,2298	987,9748	1116,853	1217,962	1287,082
MAX	22113,74	23930,72	25217,72	26171,44	26857,84
AVERAGE	8120,938	9210,818	10058,92	10716,88	11386,35
	PSNR(dB)				
Level	3.0	3.5	4.0	4.5	5.0
MIN	4,684181	4,341247	4,113745	3,952528	3,840093
MAX	19,71577	18,18335	17,65084	17,27446	17,03474
AVERAGE	9,389779	8,812748	8,413043	8,124518	7,86331
	SSIM				
Level	3.0	3.5	4.0	4.5	5.0
MIN	0,037753	0,026298	0,019684	0,011422	0,006831
MAX	0,821144	0,667332	0,506031	0,372401	0,269124
AVERAGE	0,186715	0,121845	0,078906	0,051244	0,033721
	MV				
Level	3.0	3.5	4.0	4.5	5.0
MIN	3,903747	2,607797	1,795818	1,237321	0,96892
MAX	40,07623	29,92136	22,01305	15,99964	11,30301
AVERAGE	16,70904	11,78385	8,209293	5,65843	3,95891
	STD				
Level	3.0	3.5	4.0	4.5	5.0
MIN	3,030453	2,164115	1,400256	1,290644	1,013056
MAX	466,8792	354,4259	248,8485	170,0167	109,9066
AVERAGE	63,8632	39,63141	24,20026	14,58906	8,716472
	BRISQUE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	26,2064	27,80528	28,61415	32,65158	33,99084
MAX	53,5648	52,32394	53,49069	54,157	53,12867
AVERAGE	42,86169	44,69262	45,91238	46,30607	46,25342
	NIQE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	3,624159	3,791547	3,834637	4,060596	4,215045
MAX	28,38801	29,98147	9,986371	8,761582	8,356537
AVERAGE	5,953999	6,430537	6,791194	7,070335	7,256138

Table 2.1: Metrics for the LL training set per darkness level

Validation Set

MSE					
Level	3.0	3.5	4.0	4.5	5.0
MIN	1375,33	1510,026	1610,815	1682,786	1728,921
MAX	16874,81	18427,87	19575,38	20368,45	20970,15
AVERAGE	8645,529	9782,52	10662,97	11324,07	12103,18
	PSNR(dB)				
Level	3.0	3.5	4.0	4.5	5.0
MIN	5,858414	5,476053	5,213701	5,041224	4,914787
MAX	16,74674	16,34096	16,06035	15,87051	15,75305
AVERAGE	9,1413	8,592476	8,210792	7,94494	7,698427
	SSIM				
Level	3.0	3.5	4.0	4.5	5.0
MIN	0,098815	0,060259	0,036179	0,021741	0,013508
MAX	0,322285	0,222141	0,150546	0,132669	0,123763
AVERAGE	0,174184	0,112741	0,073196	0,048115	0,031549
	MV				
Level	3.0	3.5	4.0	4.5	5.0
MIN	3,773228	2,541548	1,676796	1,116654	0,782438
MAX	27,52754	19,98084	14,30895	10,17944	7,029027
AVERAGE	16,81664	11,87607	8,301261	5,740736	3,963351
	STD				
Level	3.0	3.5	4.0	4.5	5.0
MIN	6,463488	3,835223	2,23666	1,402841	0,980341
MAX	532,613	478,0768	422,7642	366,5849	316,3779
AVERAGE	73,01146	48,25011	32,32744	21,81565	17,0961
	BRISQUE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	35,5969	39,259	41,82411	43,55633	43,67322
MAX	49,64644	49,76786	50,78904	51,29625	50,97988
AVERAGE	43,1338	44,79345	45,70872	46,86152	46,93666
	NIQE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	4,403839	4,338387	5,169145	5,183885	5,634729
MAX	8,495525	8,61579	7,875118	8,09575	8,169879
AVERAGE	5,763418	6,324112	6,730714	7,102945	7,329499

Table 2.2: Metrics for the LL validation set per darkness level

Test Set

MSE					
Level	3.0	3.5	4.0	4.5	5.0
MIN	1746,415	1932,691	2048,551	2127,227	2170,74
MAX	17362,83	19102,99	20490,78	21484,75	22175,88
AVERAGE	8096,235	9201,305	10060,86	10691,6	10996,82
	PSNR(dB)				
Level	3.0	3.5	4.0	4.5	5.0
MIN	5,734599	5,319789	5,015219	4,8095	4,671995
MAX	15,70933	15,26918	15,01634	14,85267	14,76473
AVERAGE	9,390119	8,80953	8,408226	8,14007	8,048578
	SSIM				
Level	3.0	3.5	4.0	4.5	5.0
MIN	0,071396	0,041151	0,024618	0,01545	0,012159
MAX	0,518529	0,383841	0,28422	0,202743	0,147703
AVERAGE	0,178076	0,11419	0,072755	0,04675	0,031035
	MV				
Level	3.0	3.5	4.0	4.5	5.0
MIN	4,872527	3,260436	2,052688	1,28439	0,896013
MAX	30,96231	22,77383	16,68102	11,81068	8,350514
AVERAGE	16,79939	11,84197	8,242994	5,659169	3,887245
	STD				
Level	3.0	3.5	4.0	4.5	5.0
MIN	7,536119	4,363014	2,368673	1,370221	0,957235
MAX	306,4076	216,4918	146,3811	95,435	32,1226
AVERAGE	60,86651	37,35876	22,52707	13,4657	7,593418
	BRISQUE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	27,67109	28,61396	30,19841	37,9339	34,70331
MAX	51,90809	51,9671	52,22648	53,48977	51,22521
AVERAGE	42,4467	44,38108	45,89925	46,69552	46,64028
	NIQE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	3,913654	3,966301	3,903091	4,265955	4,312168
MAX	9,995018	9,474245	9,219952	10,14361	8,689374
AVERAGE	5,944108	6,356478	6,686632	7,085731	7,245217

Table 2.3: Metrics for the LL test set per darkness level

We notice that in all cases, as the darkness level increases, the quality of the images decreases, which is also reflected in the values of the metrics. The PSNR and SSIM values decrease significantly, MSE, BRISQUE and NIQE increase, as the darkness level increases. Furthermore, from the MV we see that the average value of the pixels, as the darkness level increases,

is concentrated closer to zero, with the accumulation of values around this average value becoming more intense, as can also be seen from the STD. For the sake of completeness, we also list PSNR and SSIM values line charts for each set per darkness level.

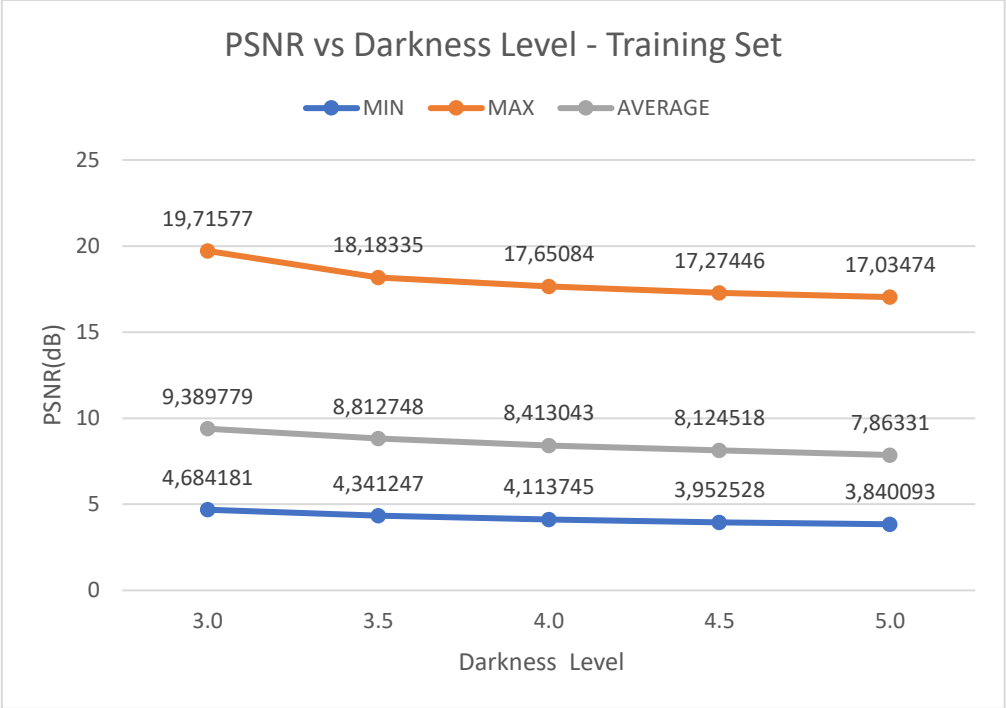


Figure 2.1: PSNR vs Darkness Level for LL training set

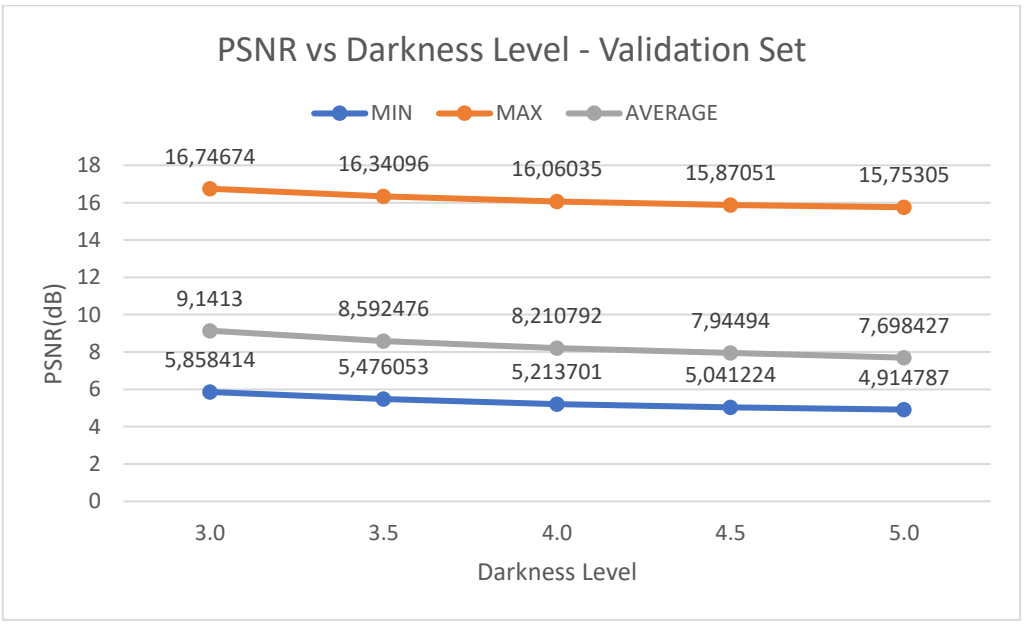


Figure 2.2: PSNR vs Darkness Level for LL validation set

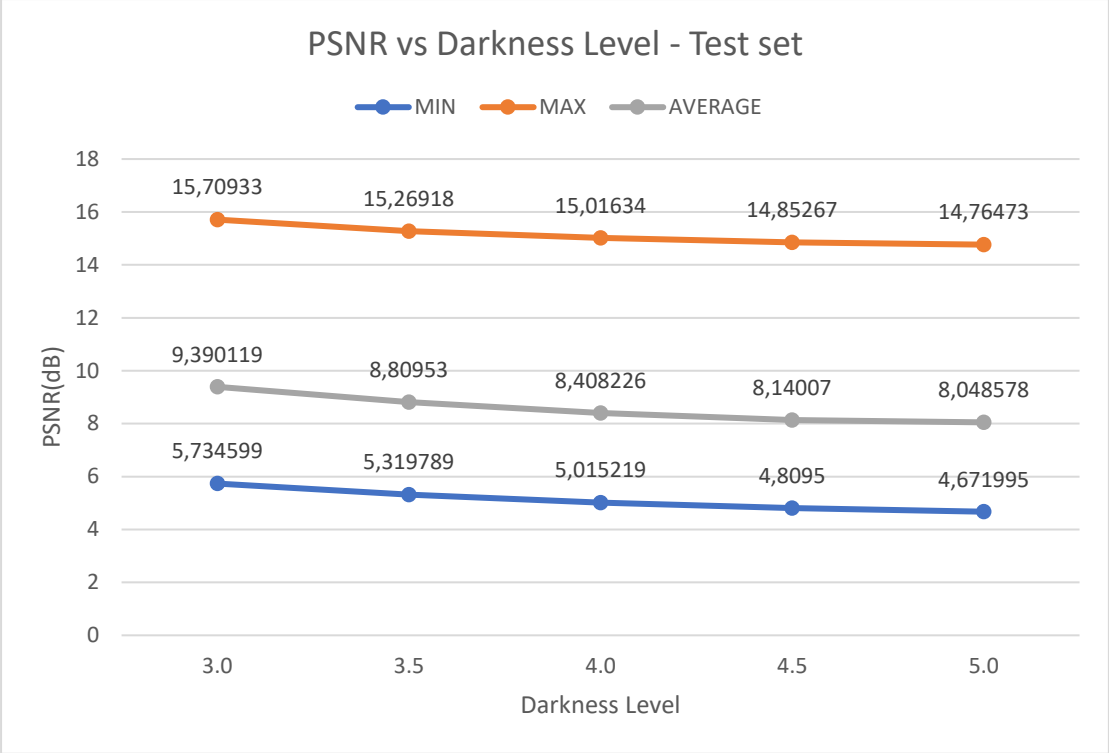


Figure 2.3: PSNR vs Darkness Level for LL test set

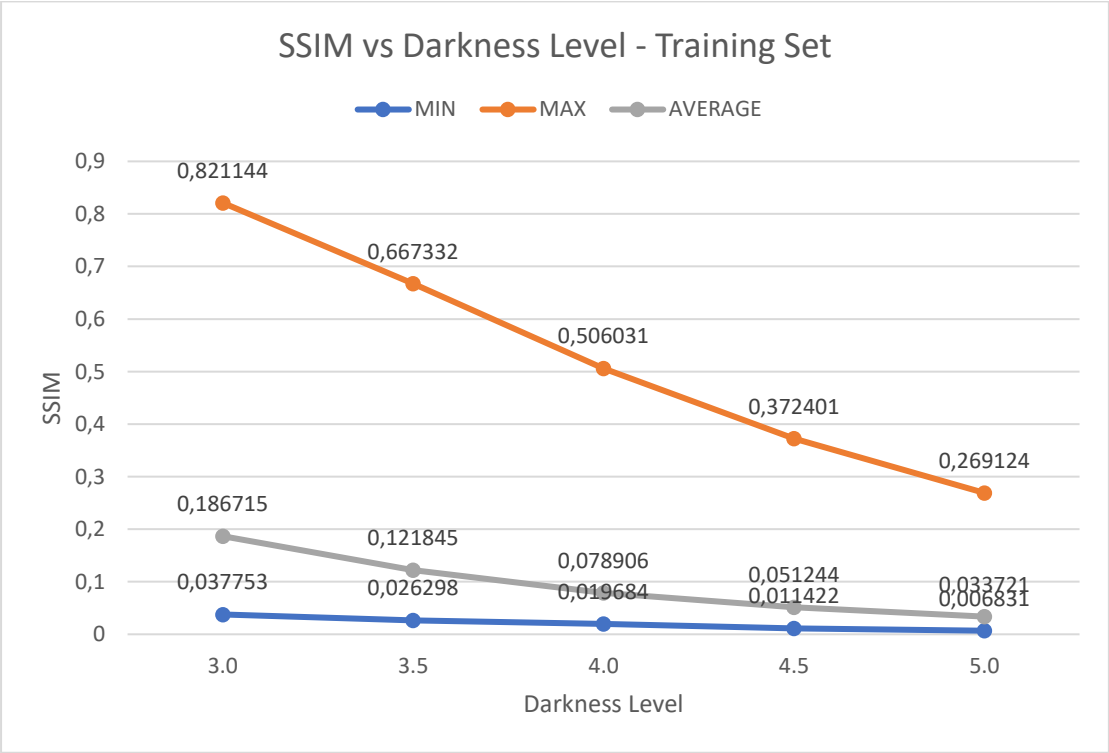


Figure 2.4: SSIM vs Darkness Level for LL training set

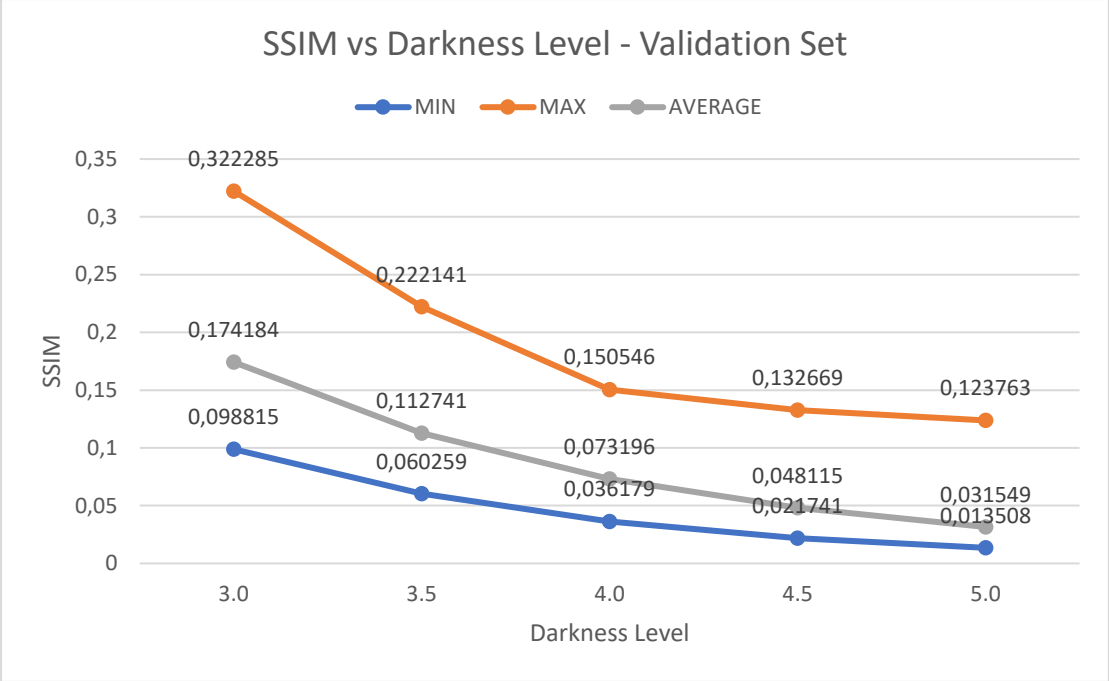


Figure 2.5: SSIM vs Darkness Level for LL validation set

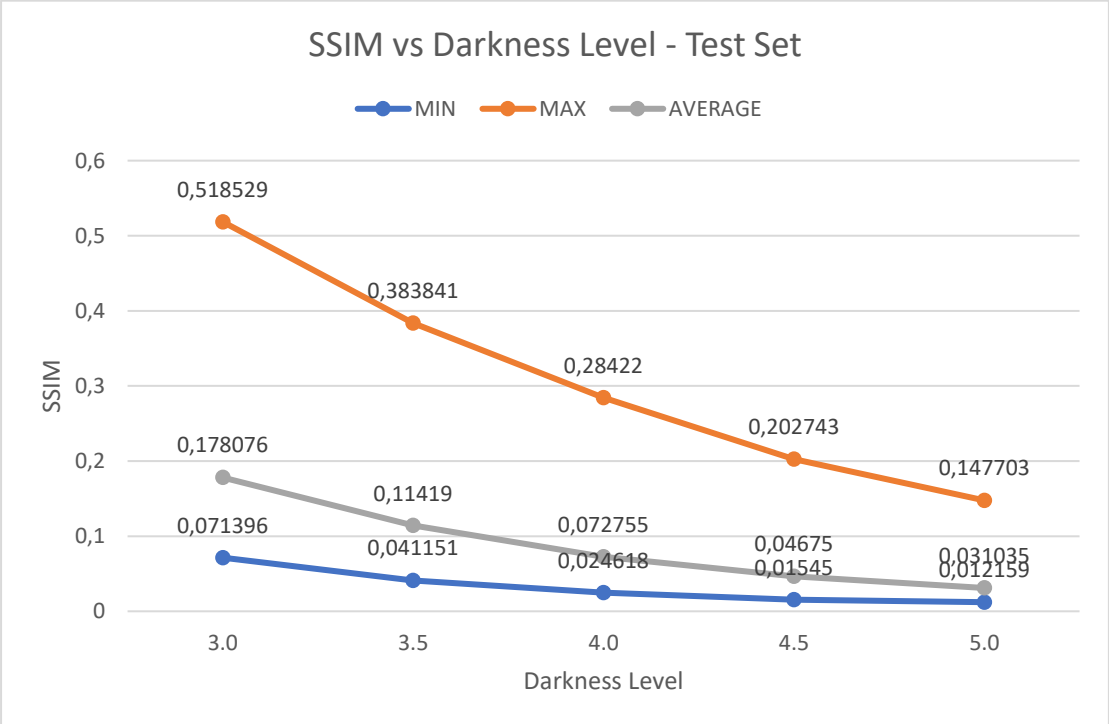


Figure 2.6: SSIM vs Darkness Level for LL test set

From the above figures we clearly see the decreasing course of the metrics, with the increase of the darkness level, something that we expect to observe

in the experimental results as well. That is, we expect that as the darkness level increases, the performance of each algorithm decreases. Furthermore, we expect that as the darkness level increases, the difficulty of restoring the color information also increases, as the gray range between the different channels is very small, even non-existent, making it difficult for the algorithms to distinguish between colors.

In the rest of the chapter, we will initially study simple point transformation algorithms, i.e. algorithms that apply some mathematical function to each pixel individually. After that, we will study algorithms that, instead of pointwise operations, apply a function per region of an image, mainly through convolution with some window.

2.1 Linear Stretching

The first LLIE method we will study is the simple linear transformation. It is a simple linear transformation whose purpose is to expand the pixel values of each image, over the entire available range of brightness values. The mathematical formula we apply has the general form:

$$y = ax + b$$

Assuming the available range of brightness values is $[x_{min},x_{max}]$, and that the range of brightness values of the image under study is $[x_{low},x_{high}]$ then the following relations should hold:

$$x_{min} = ax_{low} + b$$
$$x_{max} = ax_{high} + b$$

From these relations the following arise:

$$a = \frac{x_{max} - x_{min}}{x_{high} - x_{low}}$$
$$b = x_{min} - ax_{low}$$

By substituting into the original relation, we obtain the final transformation that we will apply to the value of each pixel. The final transformation is as follows:

$$y = \left[\frac{x_{max} - x_{min}}{x_{high} - x_{low}} * x + x_{min}\right]$$

The brackets [...] symbolize rounding to the nearest integer. At this point we must comment that this transformation assumes that the image contains at least two different brightness values, i.e. that $x_{low} \neq x_{high}$, otherwise the whole calculation breaks down.

This is a very simple method in terms of implementation. The function that implements the simple linear transformation is shown in Figure B.2.1 of Appendix B. Having this function at our disposal, we can apply it to the LL images for the training/validation/test sets. This is done using the scripts presented in Figures B.2.2 and B.2.3 in Appendix B. From the execution of these scripts, the experimental results of the method are obtained, the evaluation of which is done using the evaluation script, as shown in figure B.2.4 of appendix B. This process results in 3 excel files, one for training, one for validation and one for test set, containing the values of the metrics per darkness level. We summarize these results in the following tables. In addition to the tables of summary results, we also present line charts of PSNR and SSIM per darkness level to see how performance is affected as the Darkness Level increases. Finally, we also present a line chart of the average PSNR and average SSIM, of the experimental results, compared to the corresponding values for the LL images, so that we can see to what extent the result has improved in each case.

Training Set

MSE					
Level	3.0	3.5	4.0	4.5	5.0
MIN	140,6326	203,7791	262,9136	317,5228	475,9967
MAX	17965,93	19081,58	19543,25	19628,85	20239,29
AVERAGE	3731,675	3974,85	4111,352	4253,159	4595,576
	PSNR(dB)				
Level	3.0	3.5	4.0	4.5	5.0
MIN	5,586306	5,324661	5,220836	5,201856	5,06885
MAX	26,64994	25,03921	23,93267	23,11305	21,35476
AVERAGE	14,22736	14,01798	13,8992	13,64032	13,08758
	SSIM				
Level	3.0	3.5	4.0	4.5	5.0
MIN	-0,01566	-0,04379	-0,03538	-0,08685	-0,03875
MAX	0,899859	0,894662	0,881832	0,798625	0,707873
AVERAGE	0,465584	0,415016	0,367744	0,314237	0,25971
	MV				
Level	3.0	3.5	4.0	4.5	5.0
MIN	4,821523	3,772983	3,017588	2,442978	2,823903
MAX	191,9387	182,5086	170,0278	163,4803	141,7101
AVERAGE	56,61093	54,97912	53,25364	50,95261	48,39107
	STD				
Level	3.0	3.5	4.0	4.5	5.0
MIN	9,913017	7,069264	4,359558	2,907033	2,215631
MAX	2081,952	2096,541	2009,683	1901,554	1046,678
AVERAGE	347,094	338,1003	334,4234	328,8191	324,4536
	BRISQUE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	11,94034	11,57572	13,45012	14,41438	11,7375
MAX	50,66308	53,24597	54,05845	55,83886	55,7311
AVERAGE	36,12506	36,49375	36,58132	37,1757	37,97463
	NIQE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	2,875239	3,135501	3,338467	3,198216	3,14447
MAX	24,4297	25,14077	14,33871	7,857469	10,02405
AVERAGE	4,541229	4,631631	4,716879	4,914912	5,25733

Table 2.4: Metrics for linear transformation results on the training set

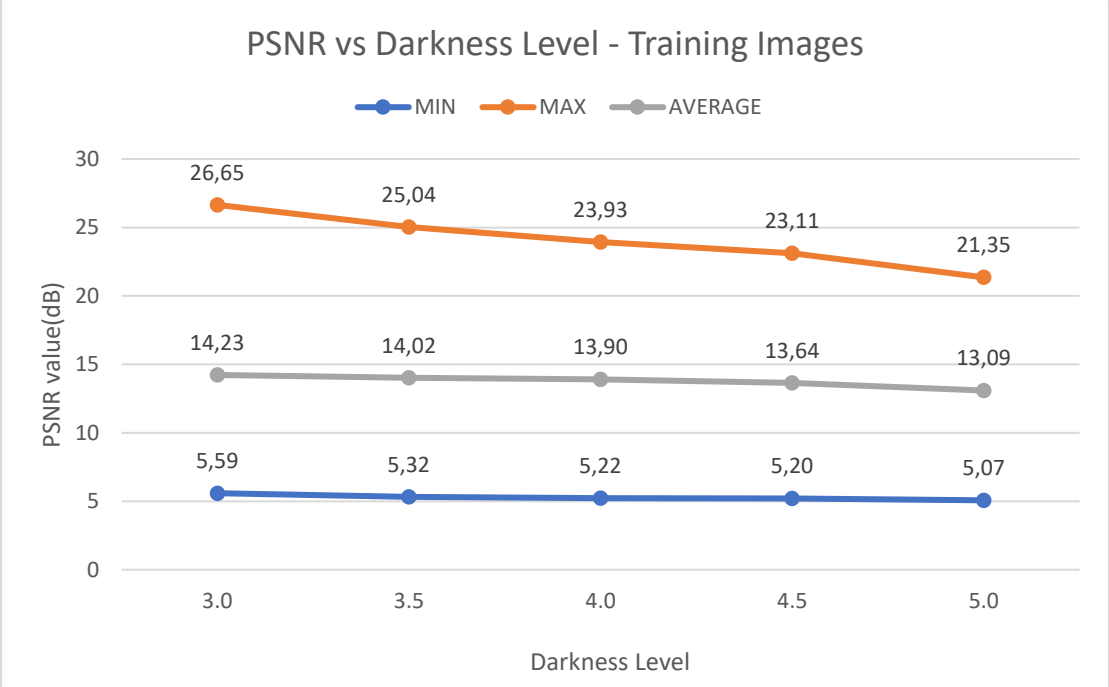


Figure 2.7: Experimental results PSNR vs Darkness level for training set

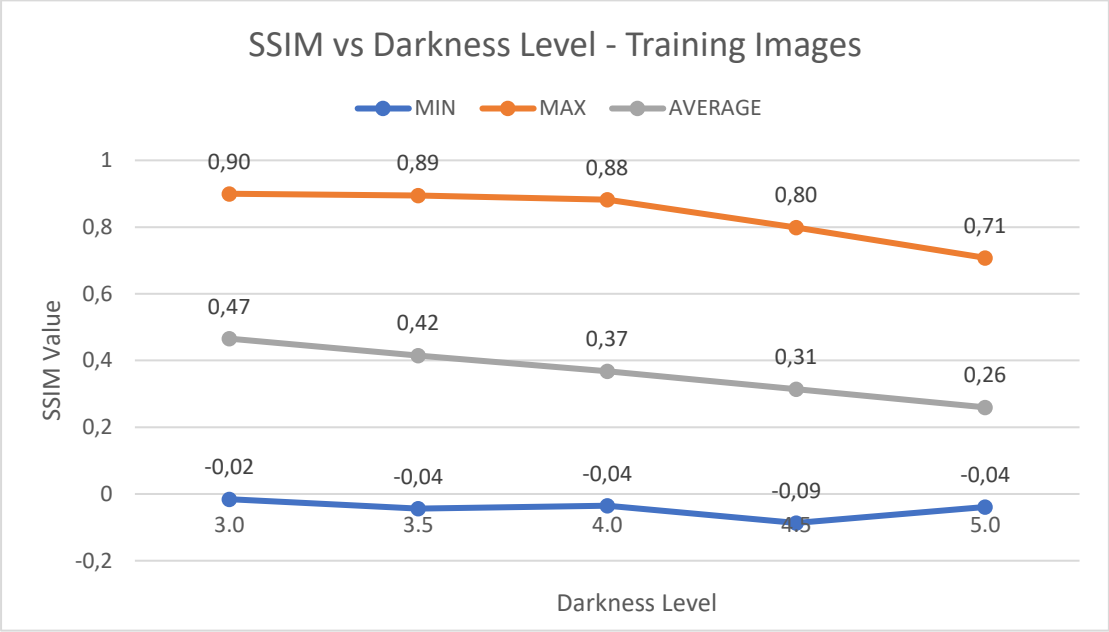


Figure 2.8: Experimental results SSIM vs Darkness level for training set

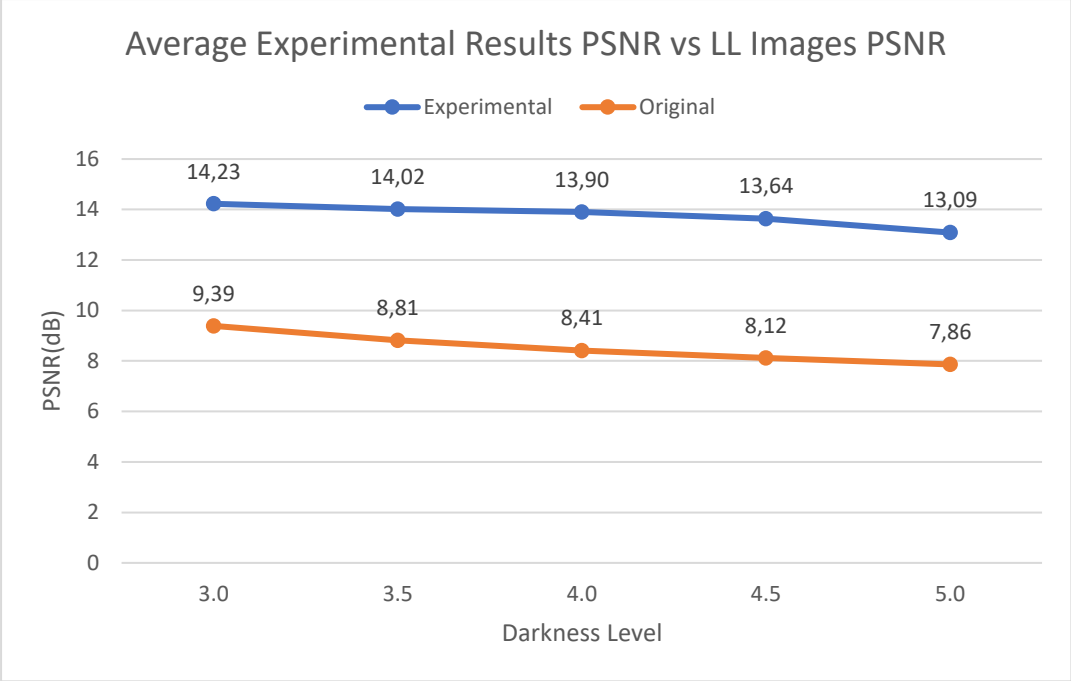


Figure 2.9: Experimental vs LL images PSNR

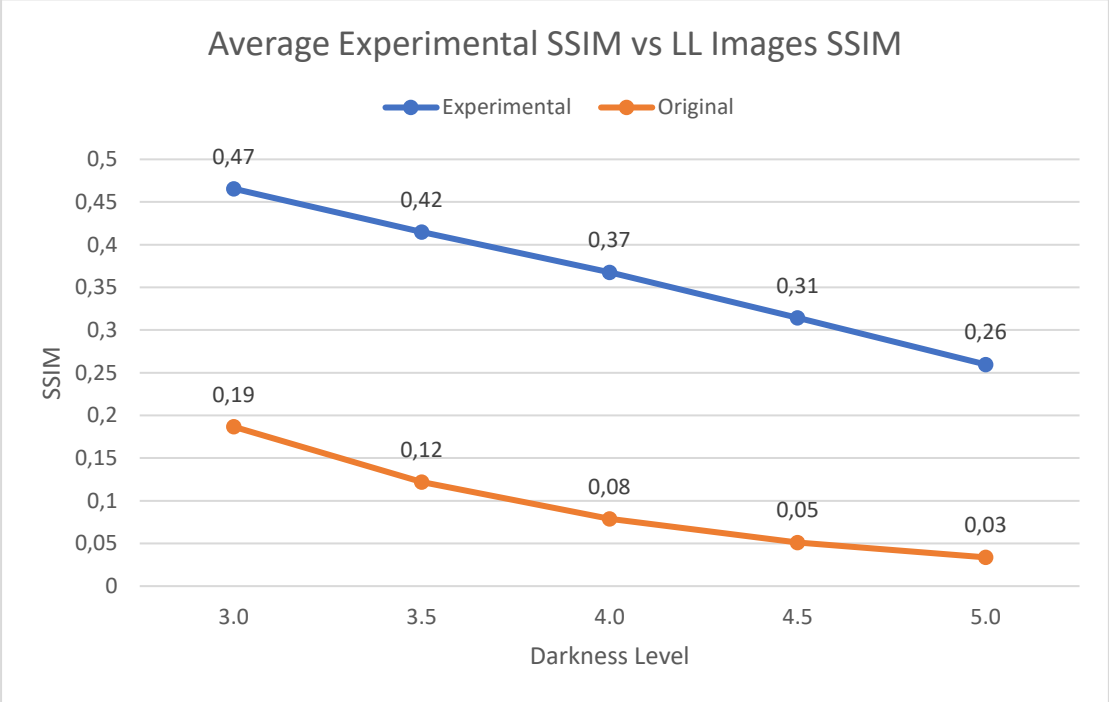


Figure 2.10: Experimental vs LL Images SSIM

Validation Set

MSE					
Level	3.0	3.5	4.0	4.5	5.0
MIN	308,5986	284,0276	441,5432	526,1866	820,5039
MAX	13437,68	14964,13	16550,91	17377,8	18087,02
AVERAGE	4208,435	4531,364	4755,887	4834,363	5105,572
	PSNR(dB)				
Level	3.0	3.5	4.0	4.5	5.0
MIN	6,847561	6,380288	5,942584	5,730857	5,557133
MAX	23,23686	23,5972	21,68107	20,91941	18,99
AVERAGE	13,82967	13,52229	13,22249	13,09242	12,69521
	SSIM				
Level	3.0	3.5	4.0	4.5	5.0
MIN	0,134799	0,089868	0,065766	0,049744	0,00626
MAX	0,780555	0,715798	0,686459	0,668334	0,609025
AVERAGE	0,449906	0,409326	0,363171	0,324284	0,288493
	MV				
Level	3.0	3.5	4.0	4.5	5.0
MIN	10,25712	8,253354	5,962295	4,101435	4,43837
MAX	152,0089	150,41	149,4178	126,1608	124,0191
AVERAGE	50,11081	48,76431	47,93793	46,1699	46,07687
	STD				
Level	3.0	3.5	4.0	4.5	5.0
MIN	34,76049	33,50606	19,62656	10,98974	24,1064
MAX	952,968	928,3534	887,8849	860,0366	788,0675
AVERAGE	289,4232	283,093	283,9726	281,2327	286,2259
	BRISQUE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	28,19167	25,37716	19,39476	13,79219	20,01727
MAX	50,63807	47,29578	51,15442	50,13438	51,51757
AVERAGE	38,82453	38,53557	38,26206	38,49837	36,71361
	NIQE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	2,956701	3,262675	3,305161	3,583528	3,916476
MAX	5,891986	6,225532	7,497119	7,183078	8,788216
AVERAGE	4,498095	4,54172	4,673231	4,815871	5,133486

Table 2.5: Metrics for linear transformation results on the validation set

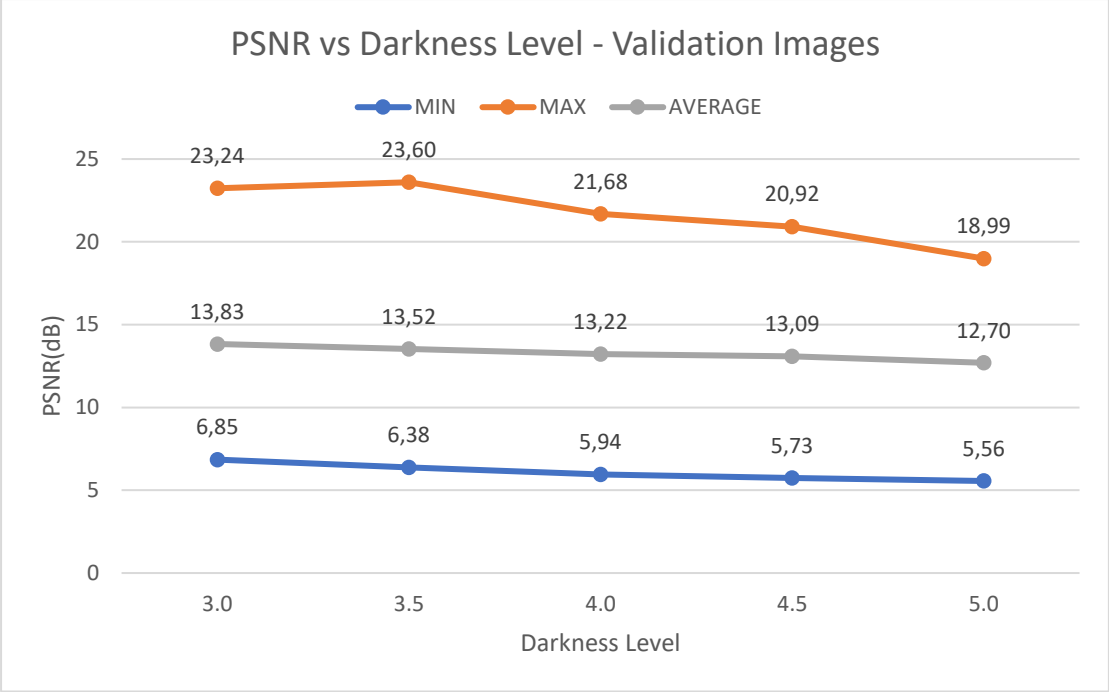


Figure 2.11: Experimental results PSNR vs Darkness level for validation set

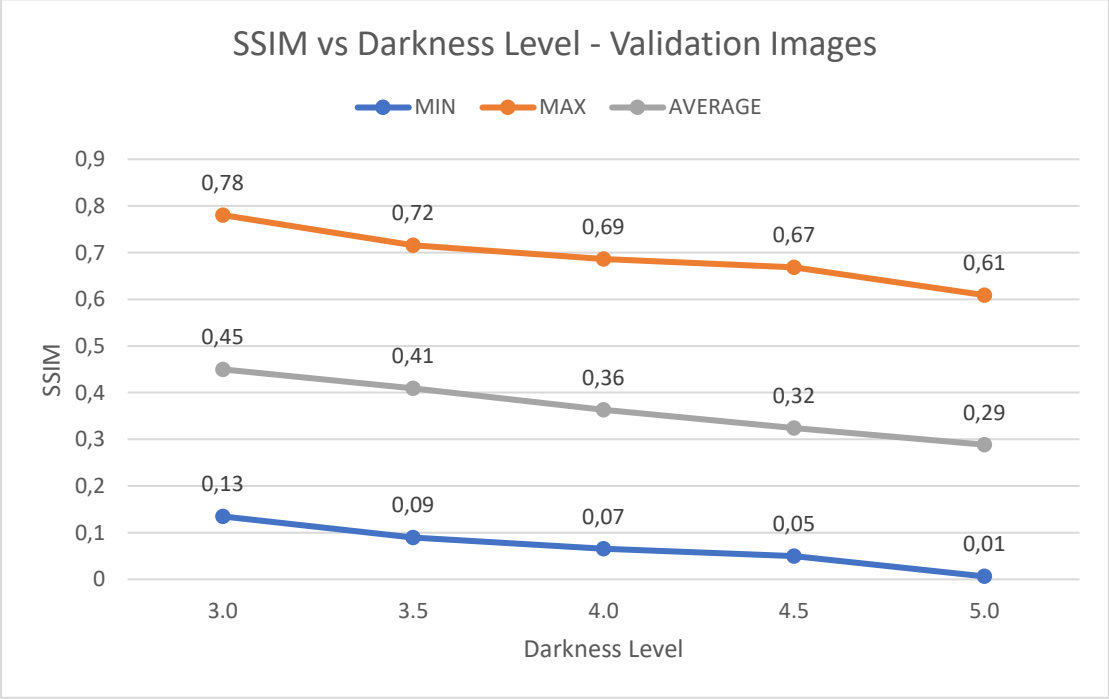


Figure 2.12: Experimental results SSIM vs Darkness level for validation set

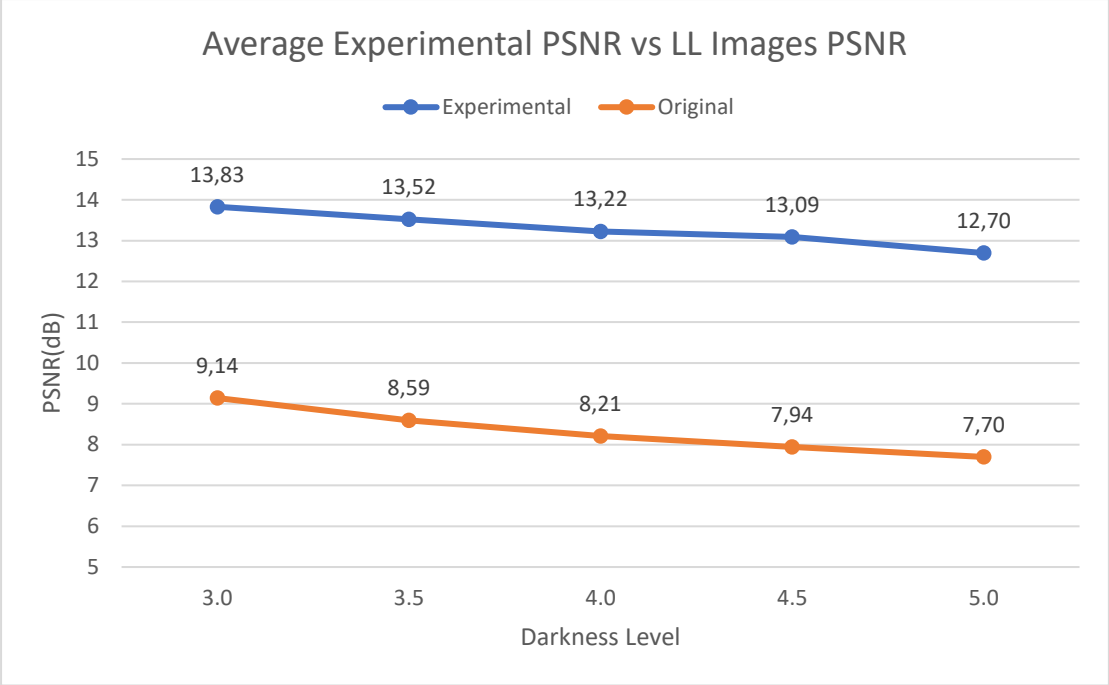


Figure 2.13: Experimental vs LL images PSNR for validation images

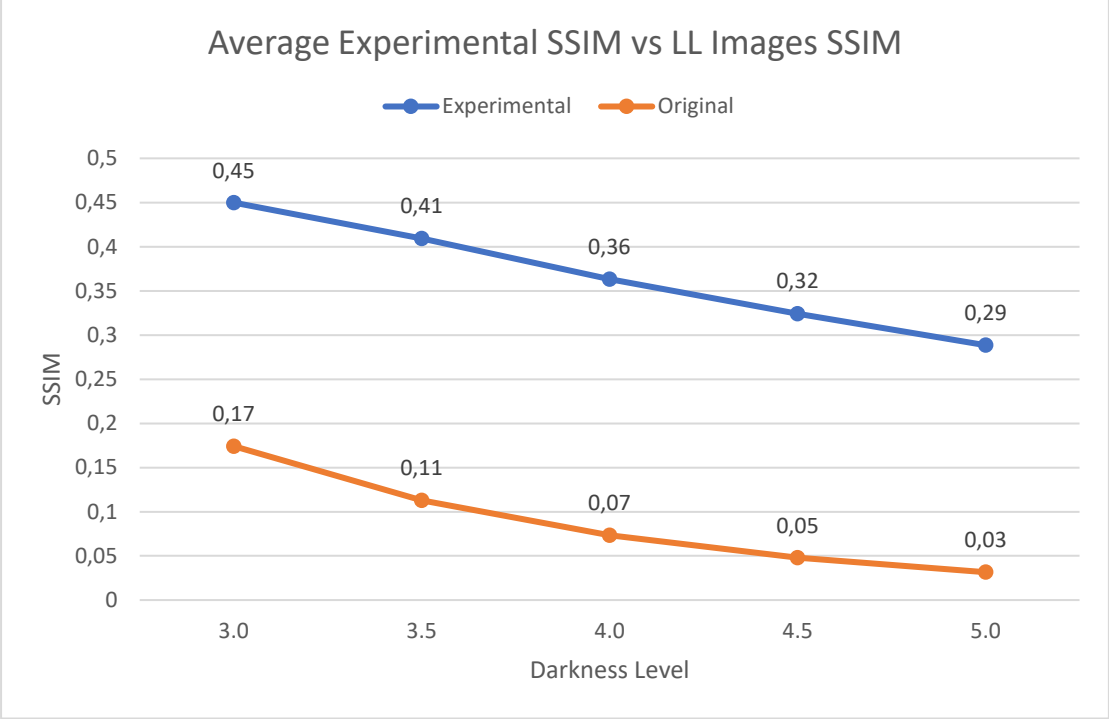


Figure 2.14: Experimental vs LL Images SSIM for validation images

Test Set

MSE					
Level	3.0	3.5	4.0	4.5	5.0
MIN	167,7695	249,8631	231,3624	273,2801	387,9315
MAX	11909,9	14153,97	15975,41	17498,34	18647,51
AVERAGE	3740,161	3986,346	4168,522	4342,931	4556,062
	PSNR(dB)				
Level	3.0	3.5	4.0	4.5	5.0
MIN	7,371722	6,622021	6,096283	5,700836	5,424595
MAX	25,88367	24,15378	24,48788	23,76472	22,24325
AVERAGE	13,86573	13,6508	13,4439	13,22458	12,8527
	SSIM				
Level	3.0	3.5	4.0	4.5	5.0
MIN	0,053248	0,014873	-0,03912	-0,02264	-0,06225
MAX	0,904539	0,85597	0,819707	0,749933	0,683889
AVERAGE	0,462487	0,417892	0,366401	0,316112	0,246587
	MV				
Level	3.0	3.5	4.0	4.5	5.0
MIN	4,872527	3,326587	2,284477	1,704316	1,32702
MAX	144,0537	137,4864	145,3222	134,9407	120,8073
AVERAGE	54,34299	53,33926	52,16029	49,44718	47,2227
	STD				
Level	3.0	3.5	4.0	4.5	5.0
MIN	28,8886	17,63814	10,08556	6,524338	4,096829
MAX	1156,826	1166,538	1166,294	1129,775	1039,747
AVERAGE	324,3299	322,9576	320,3612	310,9592	299,8686
	BRISQUE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	16,12207	17,48292	13,68804	13,65398	16,21911
MAX	45,7412	47,02508	53,74219	52,27514	52,15374
AVERAGE	35,82426	36,04668	36,136	36,85172	37,98579
	NIQE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	3,152656	3,260483	3,343861	3,659053	3,815
MAX	6,995996	6,806179	7,691173	9,328607	10,33267
AVERAGE	4,532523	4,59056	4,685158	4,977698	5,202868

Table 2.6: Metrics for linear transformation results on the test set

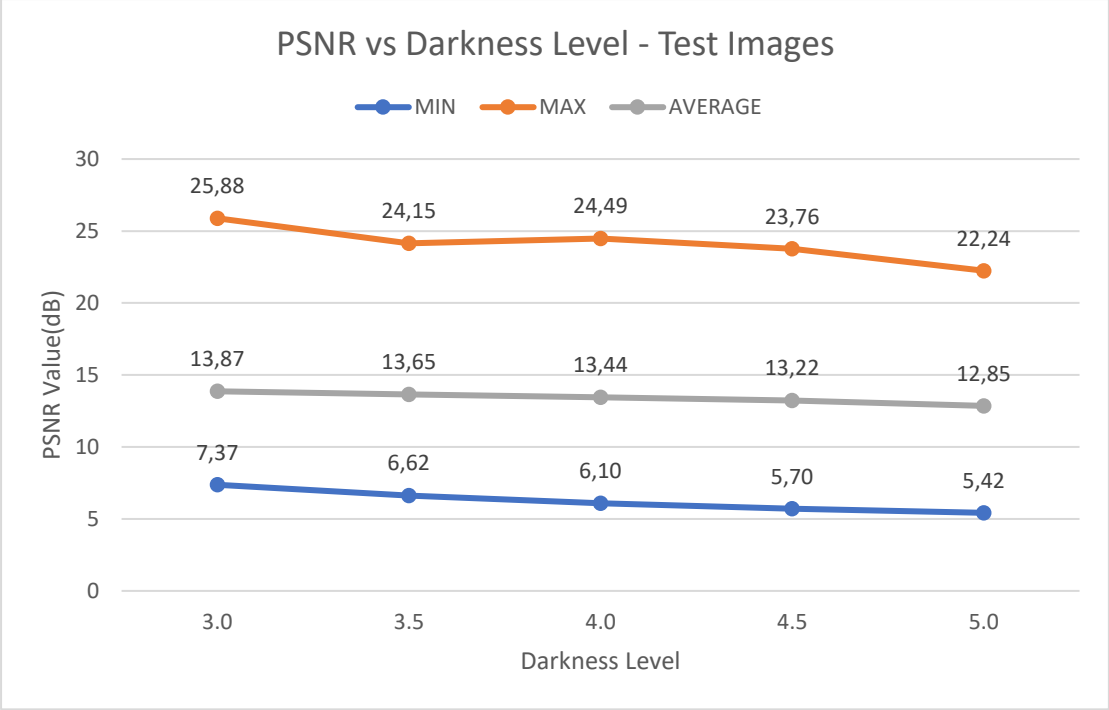


Figure 2.15: Experimental results PSNR vs Darkness level for test set

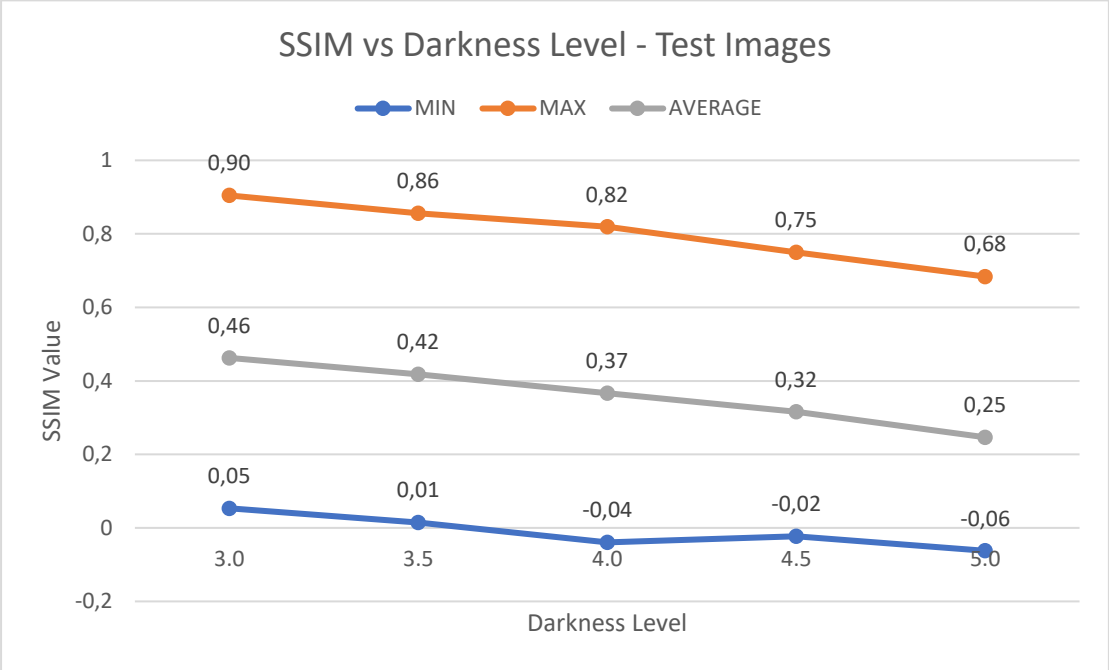


Figure 2.16: Experimental results SSIM vs Darkness level for test set

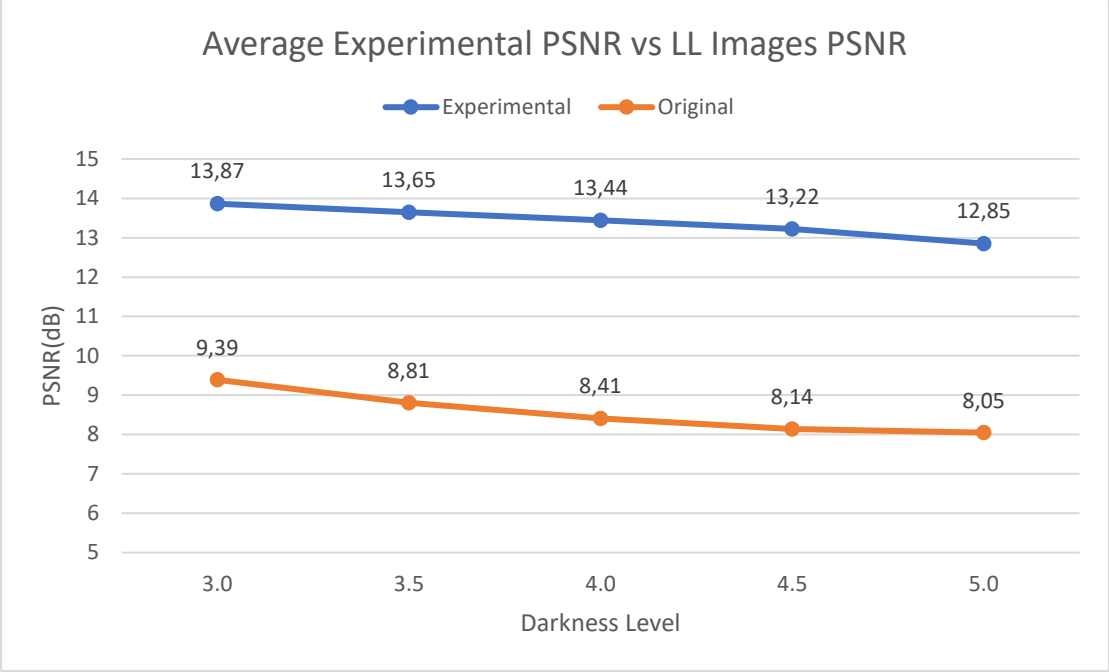


Figure 2.17: Experimental vs LL images PSNR for test images

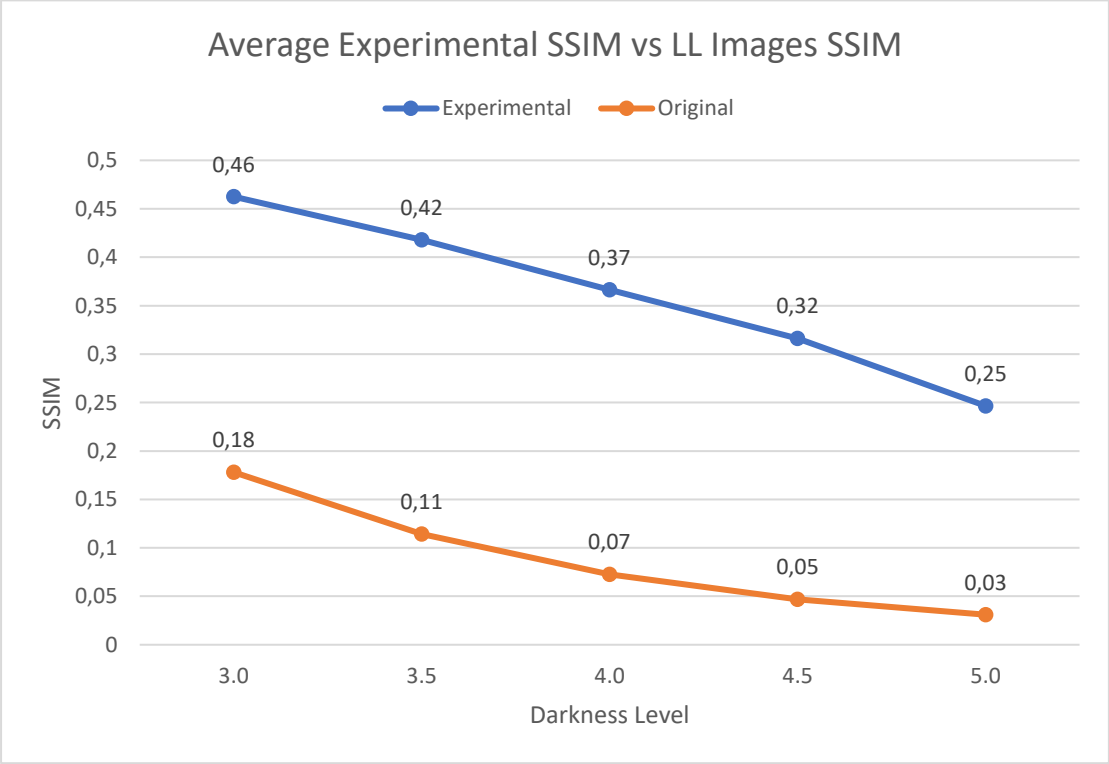


Figure 2.18: Experimental vs LL Images SSIM for test images

We notice that the linear transformation, although simple, improves the results quite a bit. Regarding the metrics with reference, in all cases, the MSE decreases and the PSNR, SSIM increases. The reduction of MSE with

the simultaneous increase of PSNR means that the image is visually improved and comes closer to the ground truth image. The increase in SSIM shows us that the images are not only improved visually but also improve the structural information, recovering a large part of this information. We see a similar improvement in the metrics without reference, where the MV increases, which means that the brightness of the image also increases, while at the same time the STD increases, which means that the pixel values are more spread around the average value, which is desirable, as we want these values to be evenly distributed over the entire range of the histogram. At the same time, the values of BRISQUE and NIQE decrease, which shows us that the images we recover from the linear transformation are closer to natural statistics, i.e. they include visual information of better quality.

This improvement can also be seen visually in the line charts of PSNR and SSIM per darkness level for average PSNR and SSIM of both the experimental results and original LL images. In all cases, we observe that for each darkness level the result improves, as the method leads to higher average values of these metrics, compared to the original LL images. Finally, we also present the line charts per darkness level for the minimum, maximum and average PSNR and SSIM, based on which we can evaluate the performance of the algorithm in relation to the darkness level. We notice that as the darkness level increases, the performance of the algorithm decreases, since the chart follows a downward path. This performance degradation is to be expected, as as the darkness level increases, the pixel values are more and more concentrated at zero, and the gray range is almost non-existent, making it difficult for the algorithm to retrieve any kind of information.

For the visual representation of the results, we will also display images from the experimental results. Specifically, from each darkness level we select the images that correspond to the maximum and minimum PSNR. For these images, we will show the corresponding Ground Truth and LL image, as well as their histograms, to enable comparisons, and to confirm the comments we made above.

Darkness Level: 3.0

MIN PSNR

Original Low Light **Normal Light** **Experimental Result**

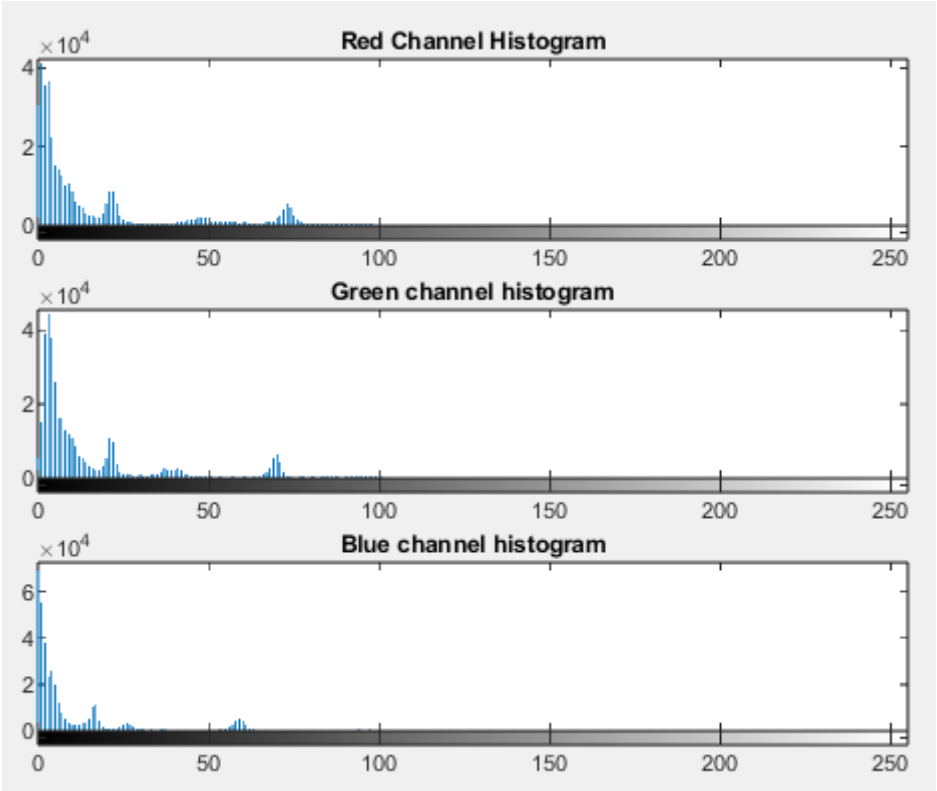


Figure 2.19: Histogram of original 3.0 LL image

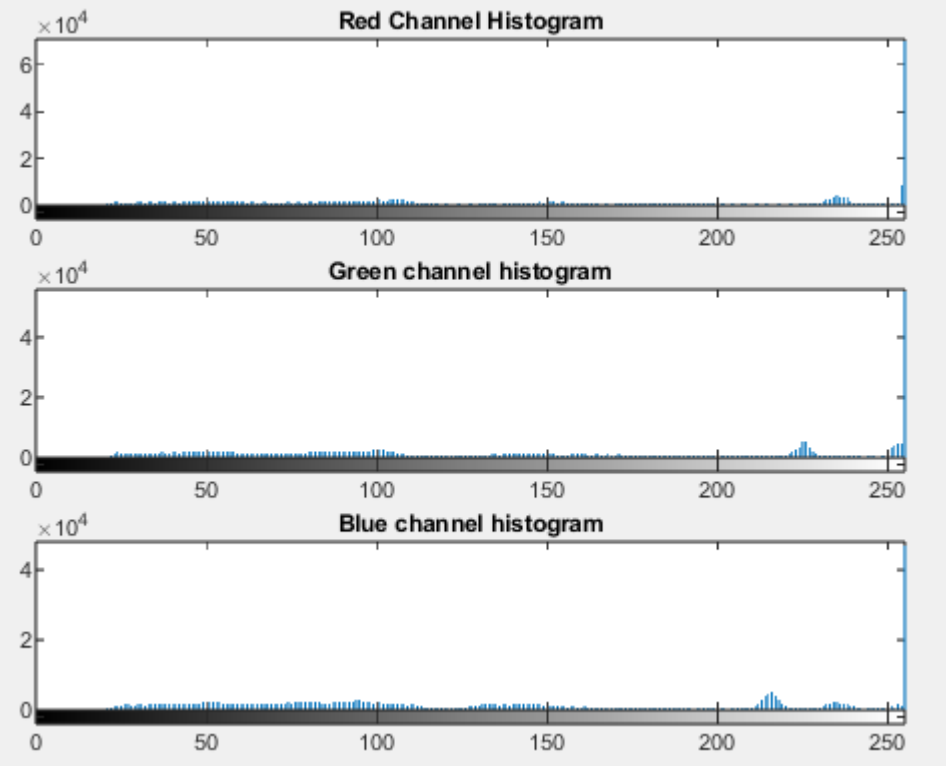


Figure 2.20: Histogram of 3.0 NL image

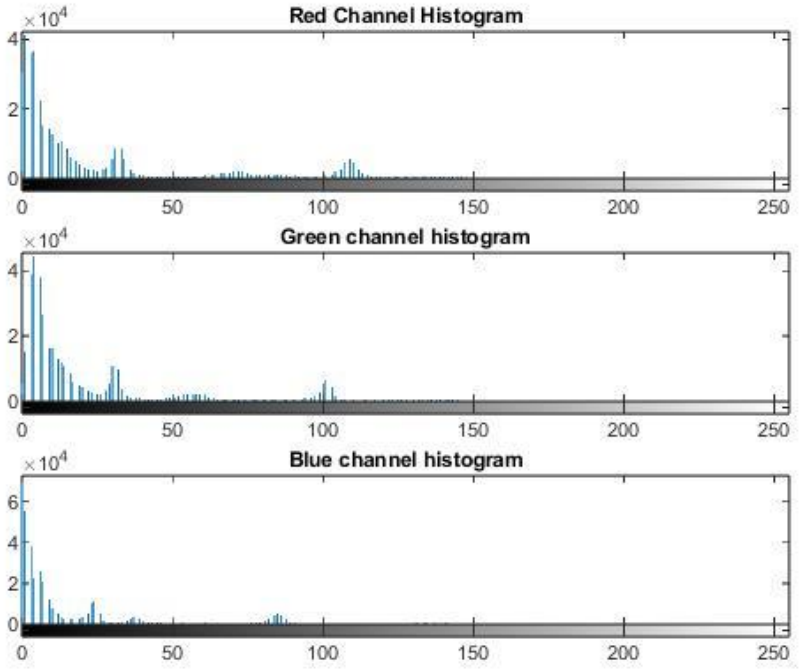


Figure 2.21: Histogram of 3.0 Experimental result

In the image corresponding to the minimum experimental PSNR, the method struggles to restore all the visual information. Observing the

histograms, we see that the Ground Truth image consists mainly of white color, that is, the number of pixel values are concentrated in the right part of the histogram, near the value 255. This is not an image with histogram that is spread over the entire available gray value range. The linear transformation aims to spread the histogram over the entire available range of values, as a result of which it is impossible to reproduce the histogram of the Ground Truth image, which is also confirmed by the above results. Comparing the histograms of the LL image and the experimental result, indeed the algorithm has flattened the histogram, but it has failed to reproduce the Ground Truth image, which is expected, for the reasons we mentioned above.

MAX PSNR

Original Low Light **Normal Light** **Experimental Result**

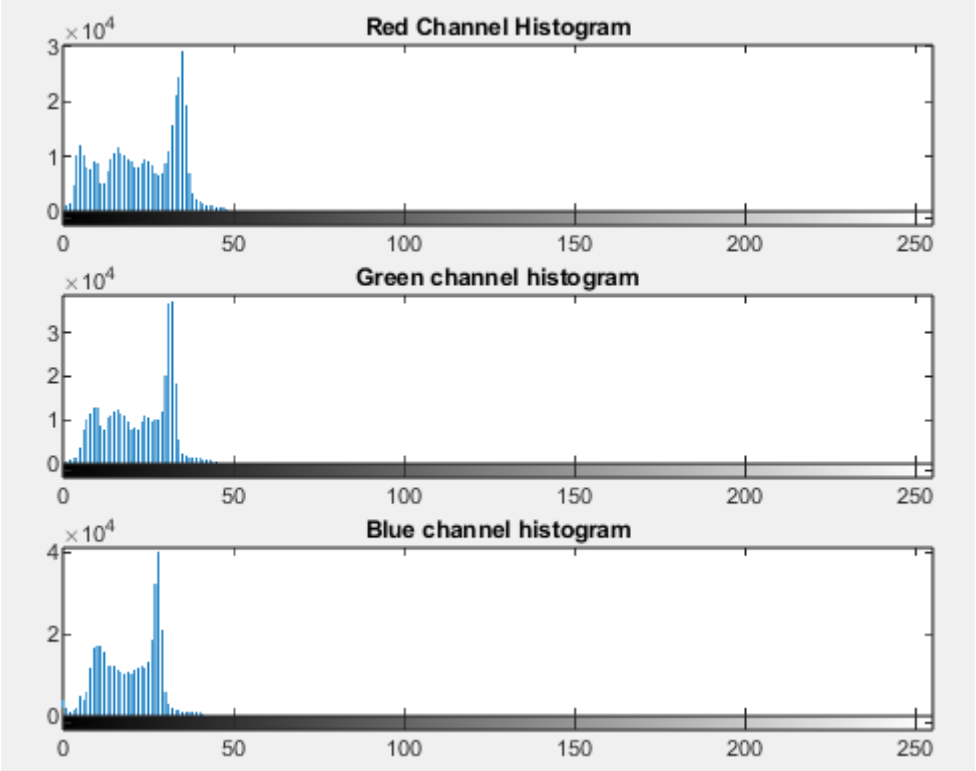


Figure 2.22: Histogram of original 3.0 LL image

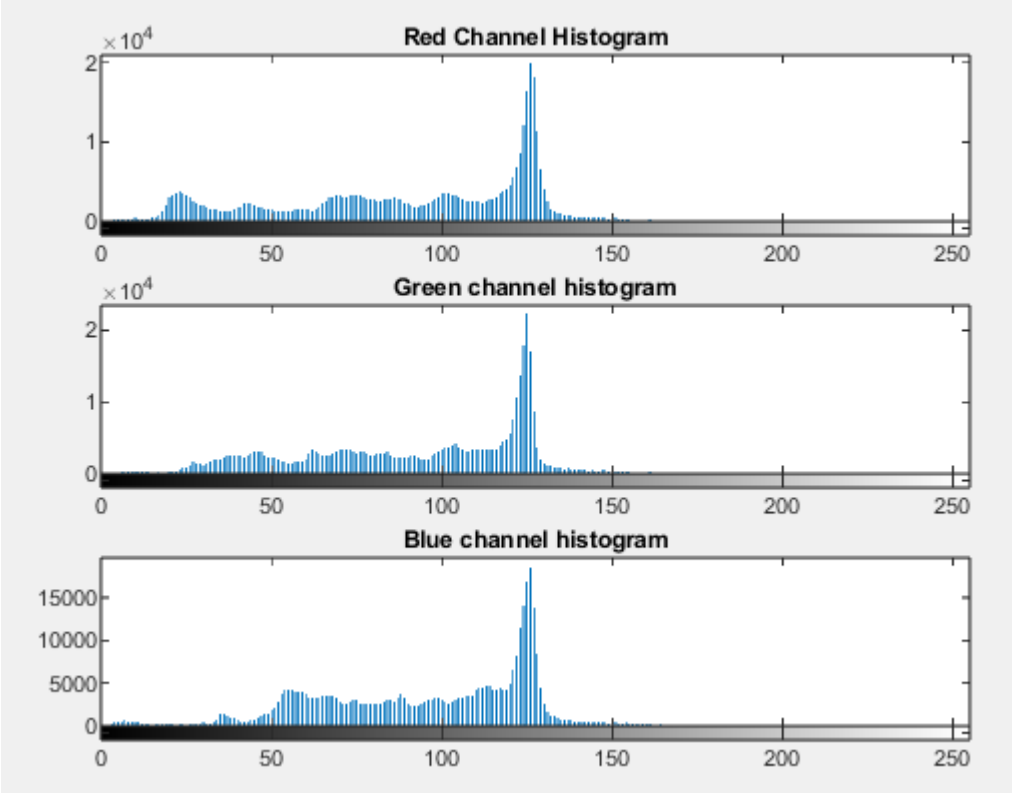


Figure 2.23: Histogram of 3.0 NL image

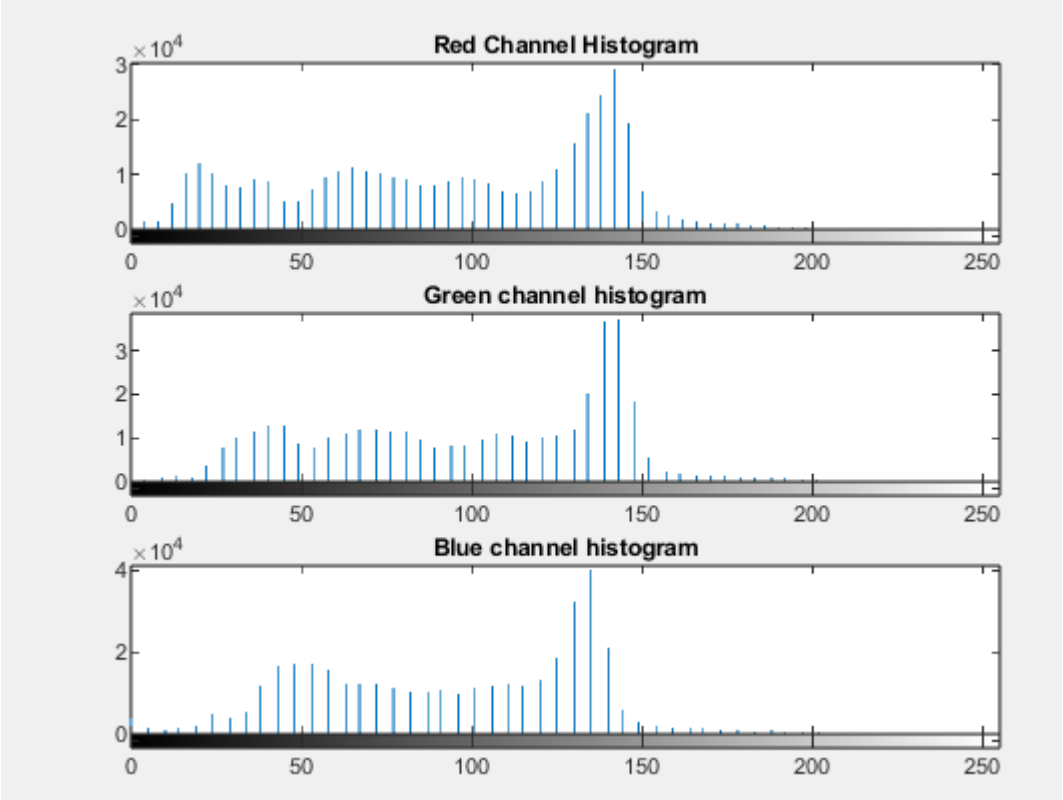


Figure 2.24: Histogram of 3.0 experimental result

In the case of the maximum experimental PSNR, we have a Ground Truth image with a histogram that spans the entire available value range, so we expect the linear transformation to be able to reproduce the result. Indeed, we see that the experimental result is visually very close to the ground truth. The method has spread the histogram of the LL image over the entire available value range, with the form of the histogram of the experimental result being similar to that of the histogram of the ground truth image, justifying the visual result as well.

Darkness Level: 3.5

MIN PSNR

Original Low Light **Normal Light** **Experimental Result**

And in the case of Darkness Level 3.5 we notice that the same image corresponds to the minimum experimental PSNR, for the same reasons we mentioned in Darkness Level 3.0. The ground truth image consists mainly of white color, causing its histogram to be clustered on the right side near the value 255, which is impossible to reproduce by the linear transformation.

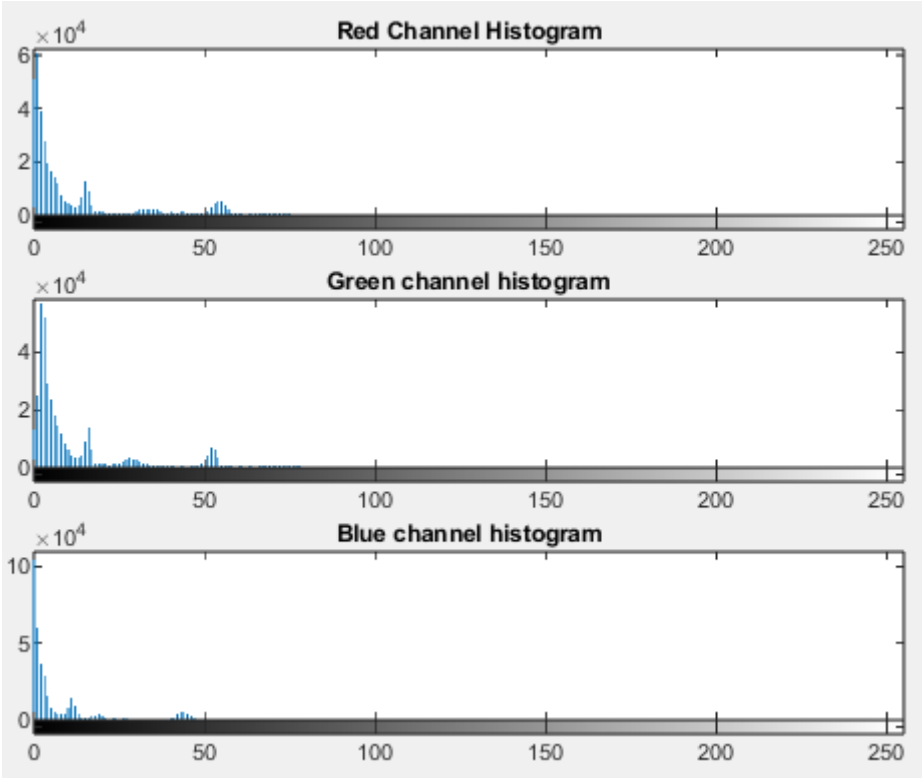


Figure 2.25: Histogram of 3.5 LL image

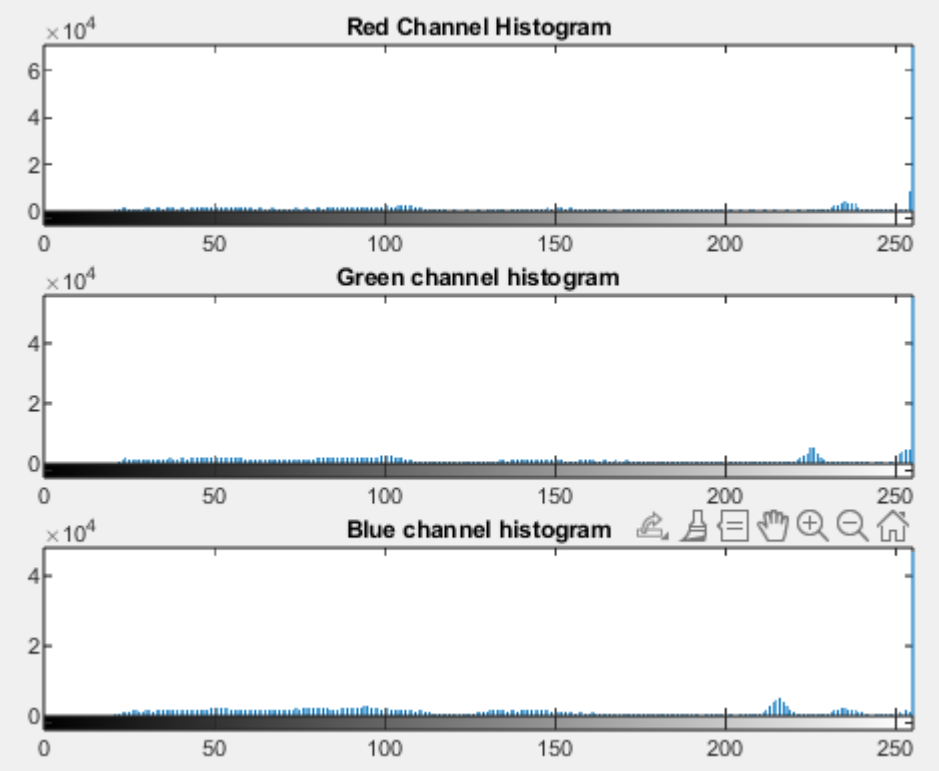


Figure 2.26: Histogram of 3.5 NL image

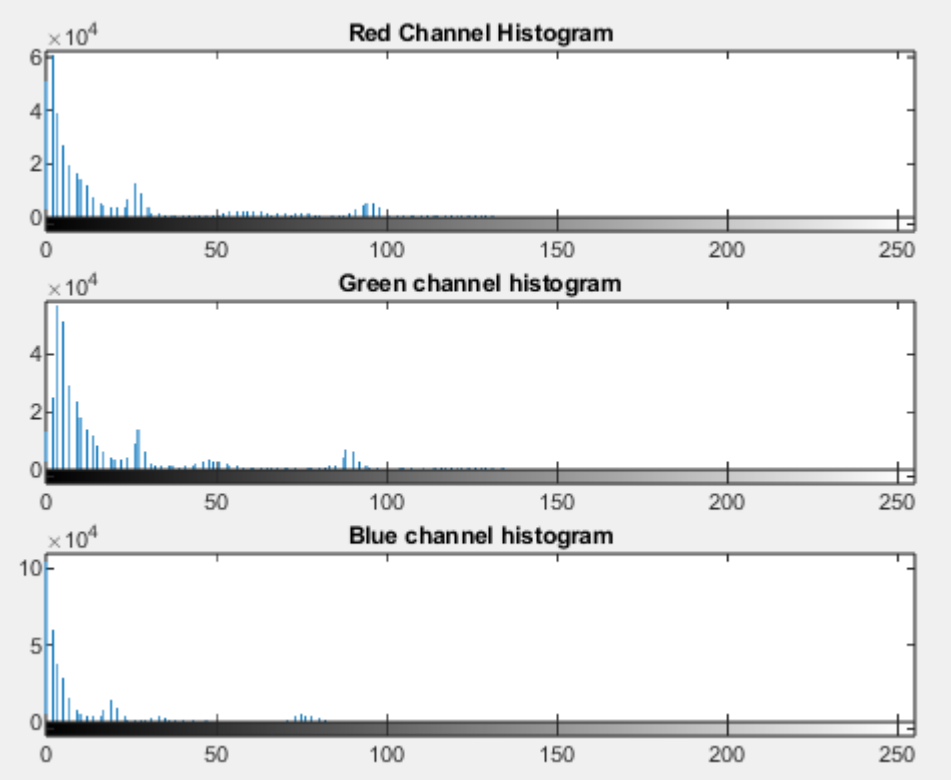


Figure 2.27: Histogram of 3.5 experimental result

MAX PSNR

Original Low Light **Normal Light** **Experimental Result**

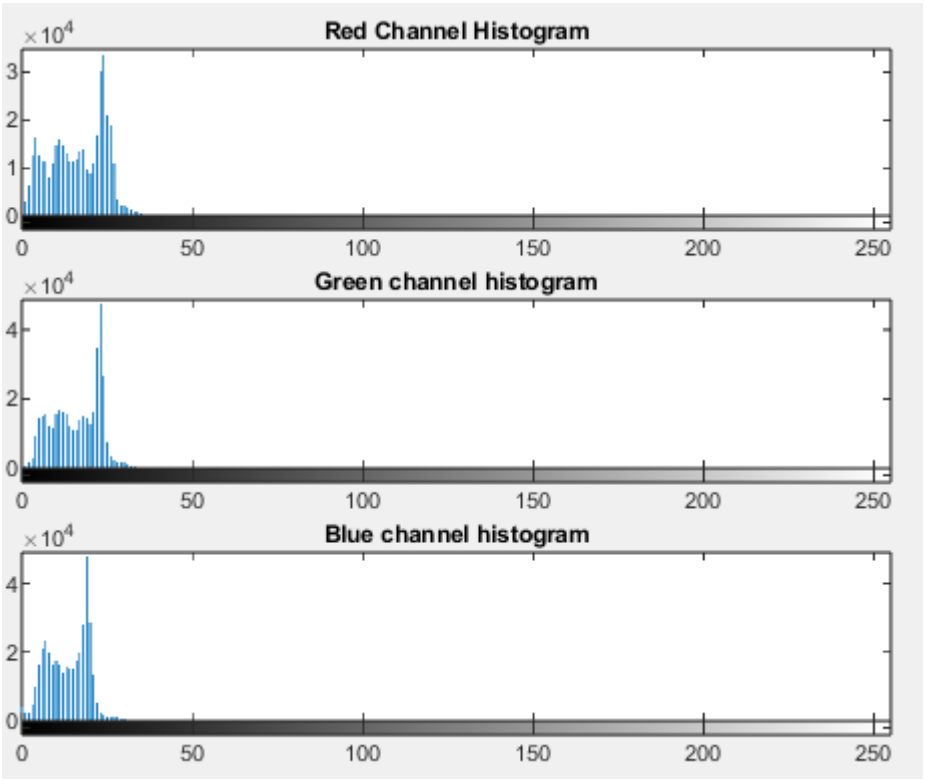


Figure 2.28: Histogram of 3.5 LL image

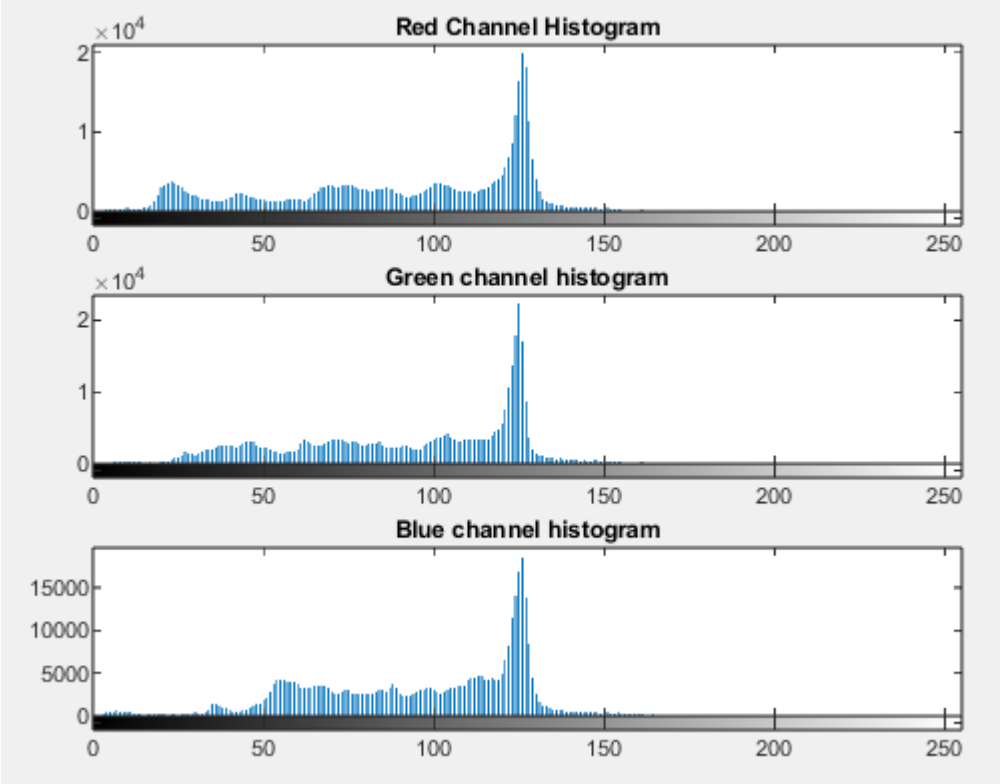


Figure 2.29: Histogram of 3.5 NL image

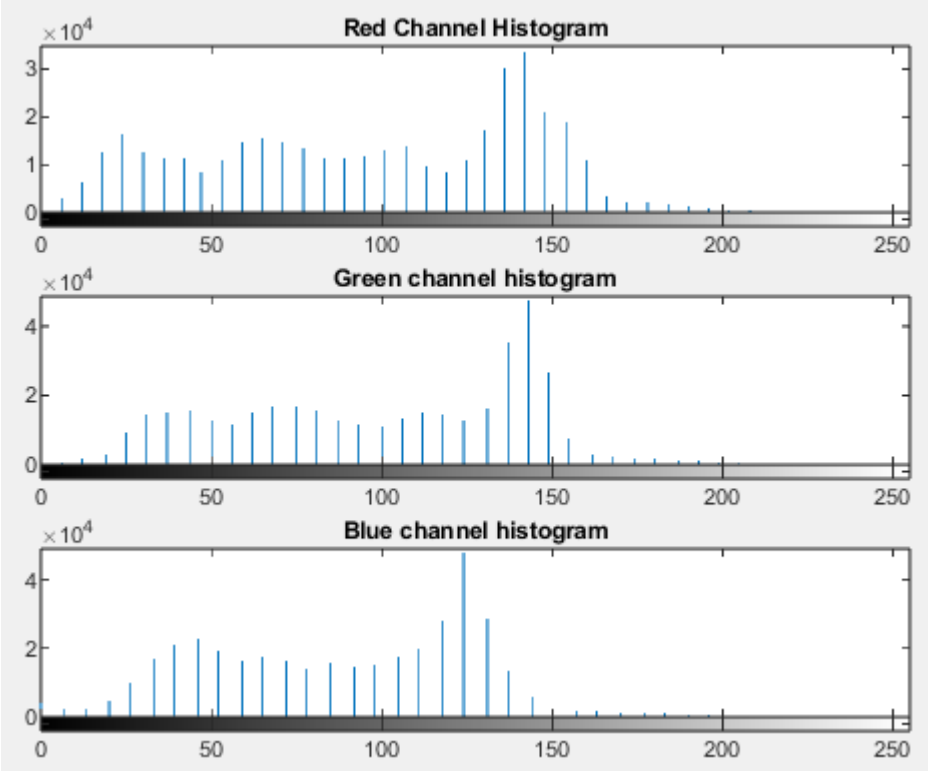
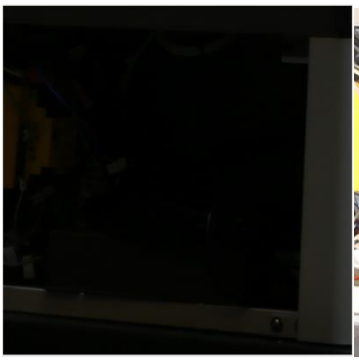


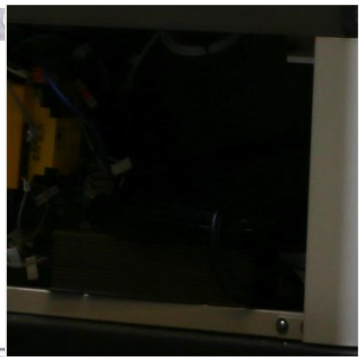
Figure 2.30: Histogram of 3.5 experimental result

In the case of the maximum experimental PSNR, the same image corresponds to that of Darkness Level 3.0, for the same reasons mentioned above. The ground truth image consists of a histogram spread over the entire available range of values, which can be reproduced by linear transformation. Indeed, the linear transformation succeeds in reproducing it and recovering much of the visual information, which can be seen from the images themselves, with the experimental result being visually very close to the ground truth.

Darkness Level: 4.0

MIN PSNR





Original Low Light **Normal Light** **Experimental Result**

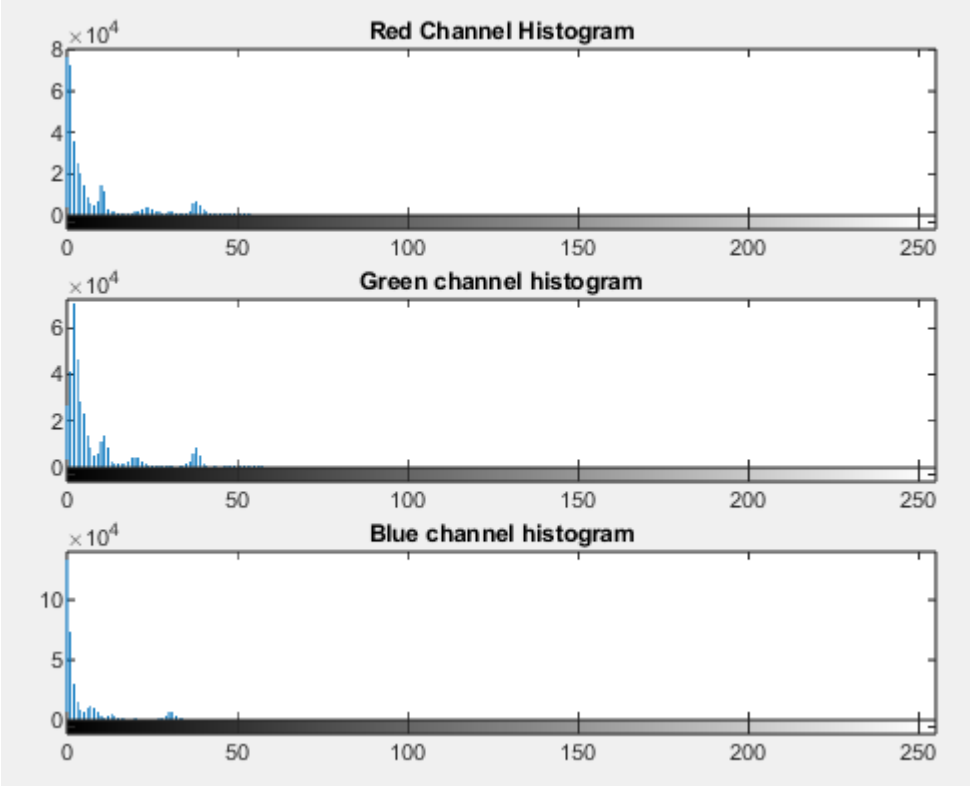


Figure 2.31: Histogram of 4.0 LL Image

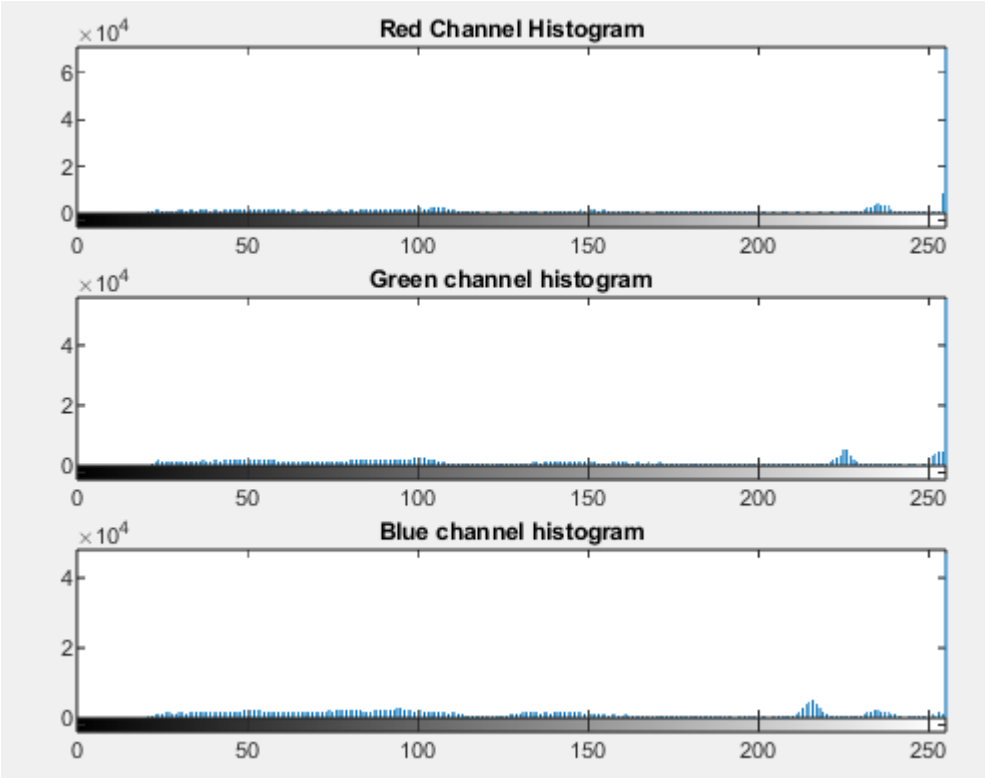


Figure 2.32: Histogram of 4.0 NL Image

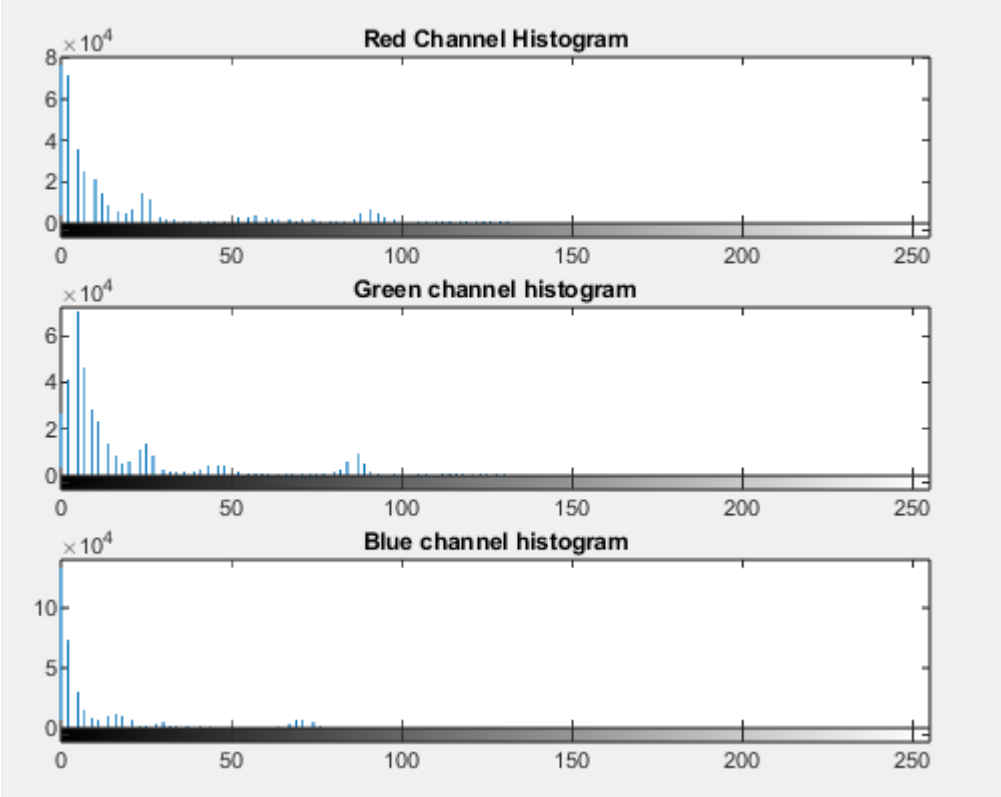
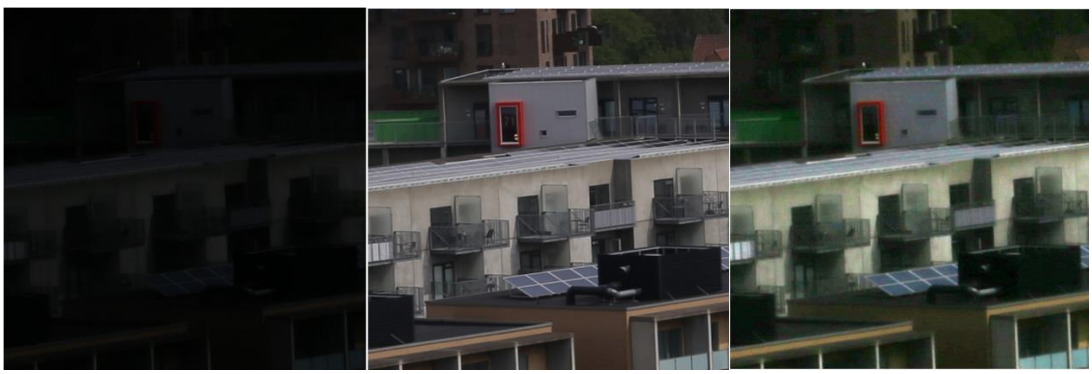


Figure 2.33: Histogram of 4.0 experimental result

And in the case of Darkness Level 4.0, the image corresponding to the minimum experimental PSNR is the same as that of Darkness Levels 3.0 and 3.5, for the same reasons. The histogram of the ground truth image is concentrated near the value 255, which cannot be reproduced by the linear transformation, since its logic is to take a histogram with low contrast, and spread it over the entire available range of brightness values , increasing the contrast in this way.

MAX PSNR



Original Low Light **Normal Light** **Experimental Result**

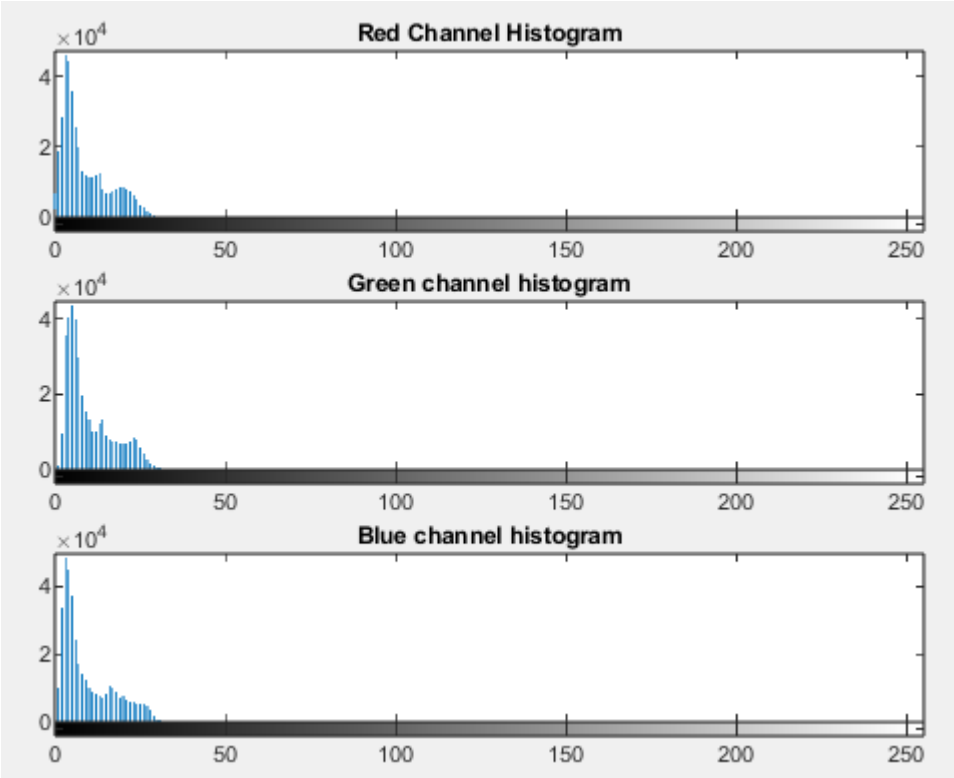


Figure 2.34: Histogram of 4.0 LL Image

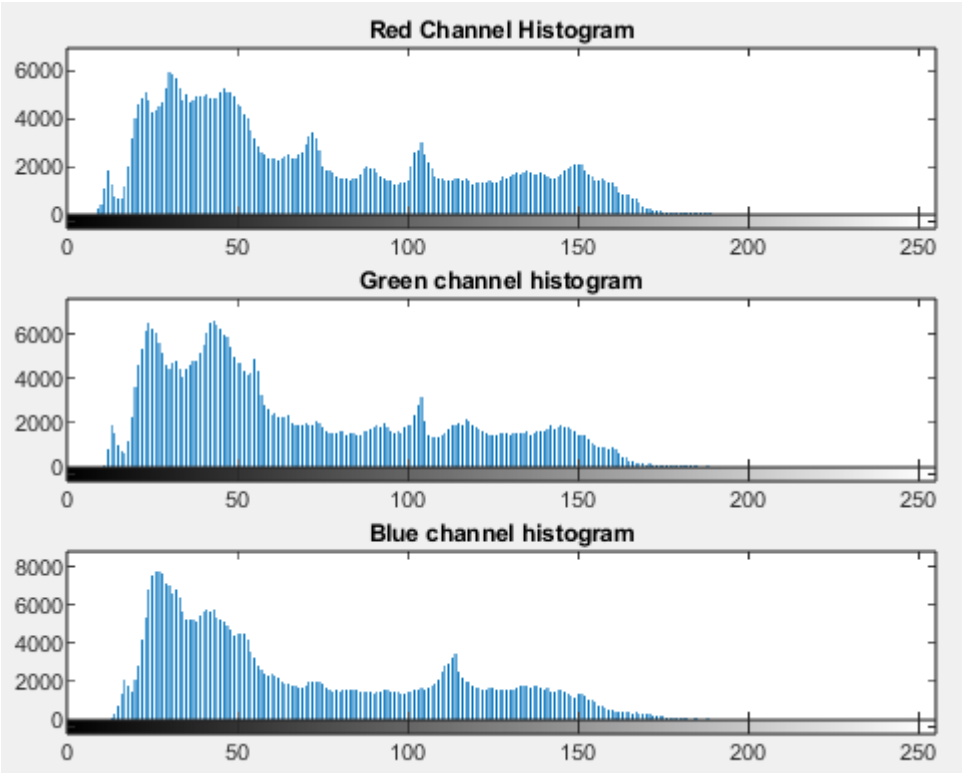


Figure 2.35: Histogram of 4.0 NL Image

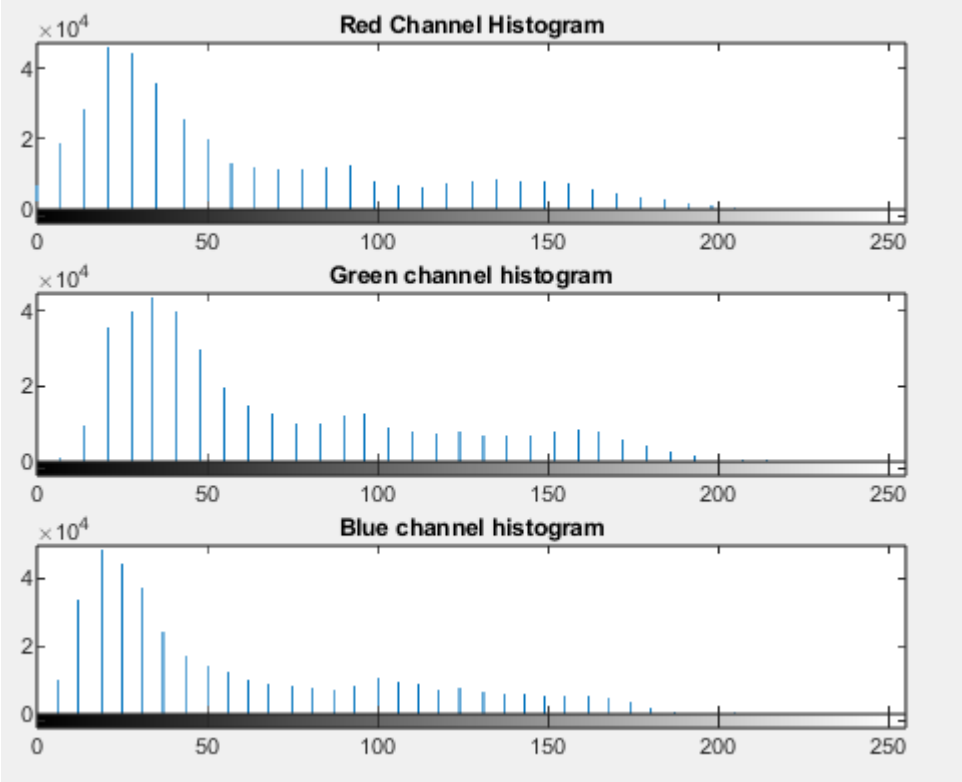


Figure 2.36: Histogram of experimental result

In the case of the maximum experimental PSNR, we have an image where the ground truth case consists of a histogram that is spread over the entire range of available values, so we expect the method to be able to reproduce this. Indeed, the image resulting from the application of the linear transformation is visually very close to the ground truth image. The histogram of the LL image has been spread over the whole range of luminance values, with the histogram of the experimental result having the same form as the ground truth. Something else that is noticed is that in the experimental result there have started to be some color distortions, which is due to the fact that the LL image has started to become too dark, making it difficult to fully recover the color information. This is also reflected in the histogram, as the pixel values of the experimental result have been distributed in fewer values in the histogram, compared to the histogram of the ground truth image, which means that we will also have less color information.

Darkness Level: 4.5

MIN PSNR

Original Low Light **Normal Light** **Experimental Result**

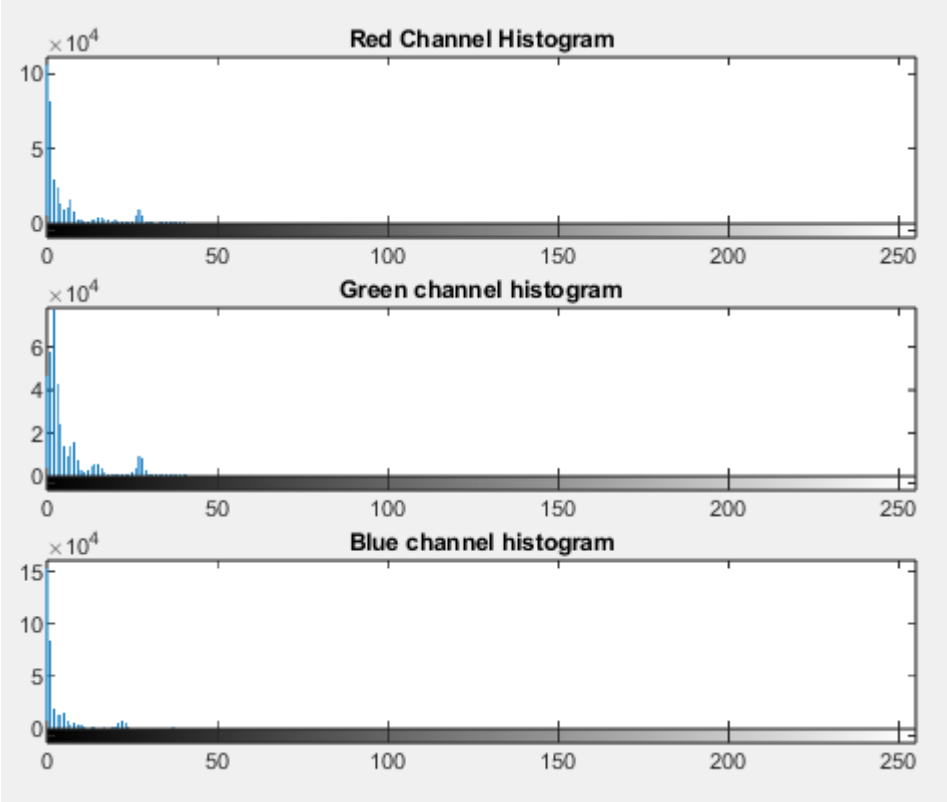


Figure 2.37: Histogram of 4.5 LL Image

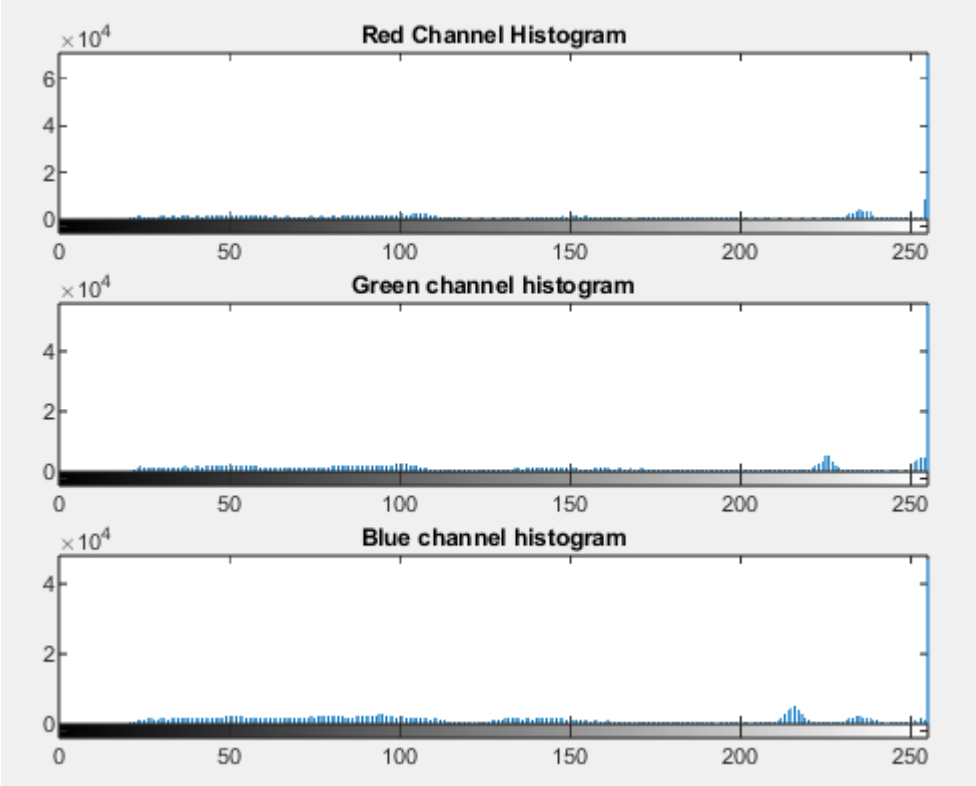


Figure 2.38: Histogram of 4.5 NL Image

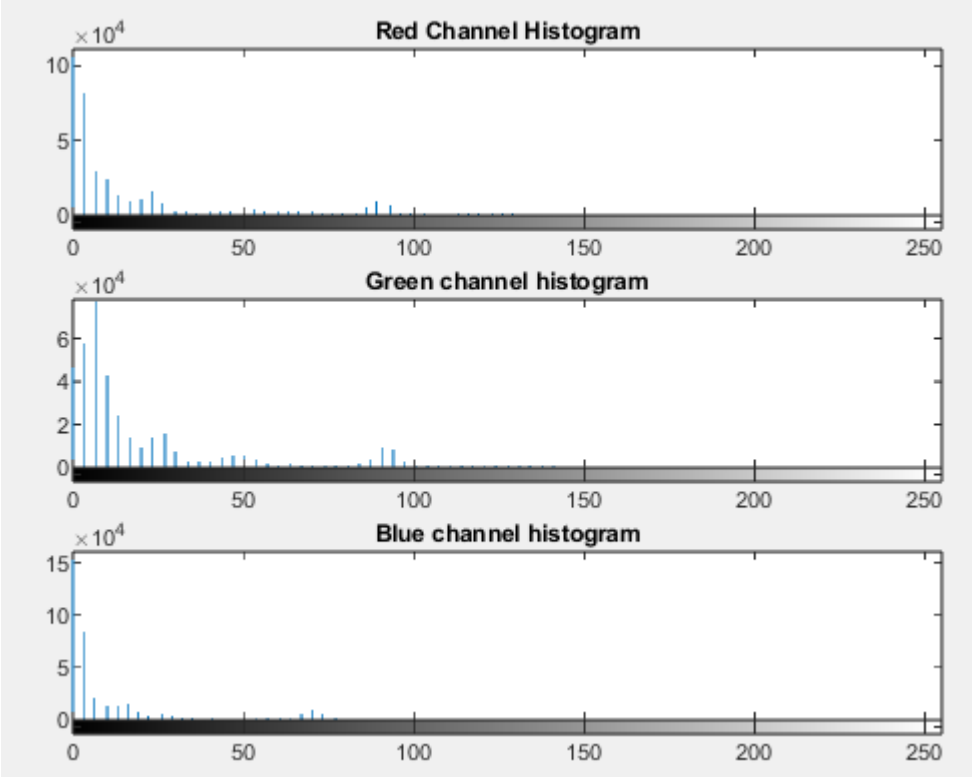
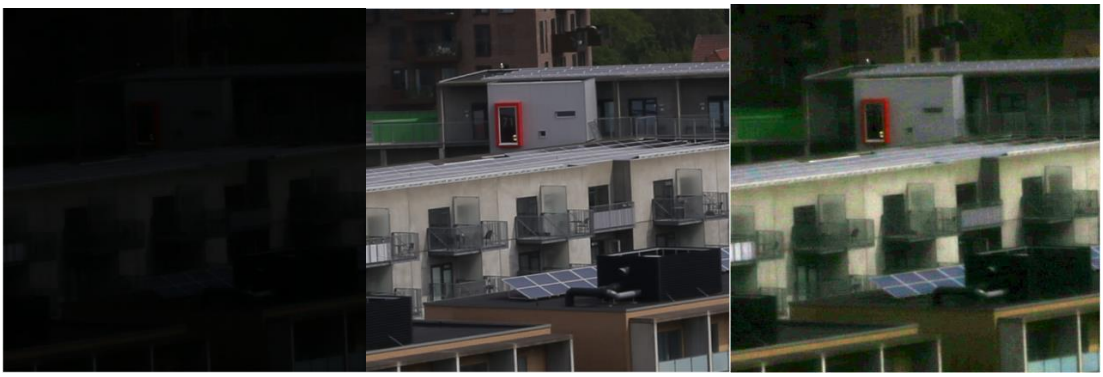


Figure 2.39: Histogram of 4.5 experimental result

Here again, the minimum experimental PSNR corresponds to the same image as in all previous cases, for the same reasons we mentioned. Furthermore, the fact that the images have become too dark at this level makes it even more difficult for the algorithm to recover visual information.

MAX PSNR



Original Low Light **Normal Light** **Experimental Result**

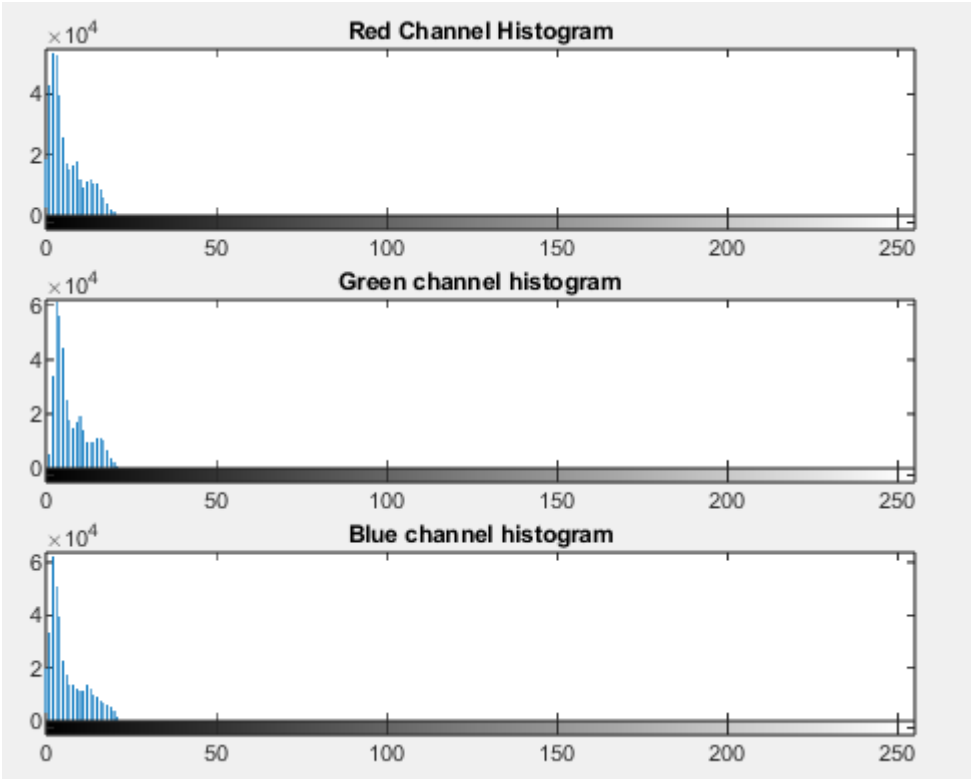


Figure 2.40: Histogram of 4.5 LL Image

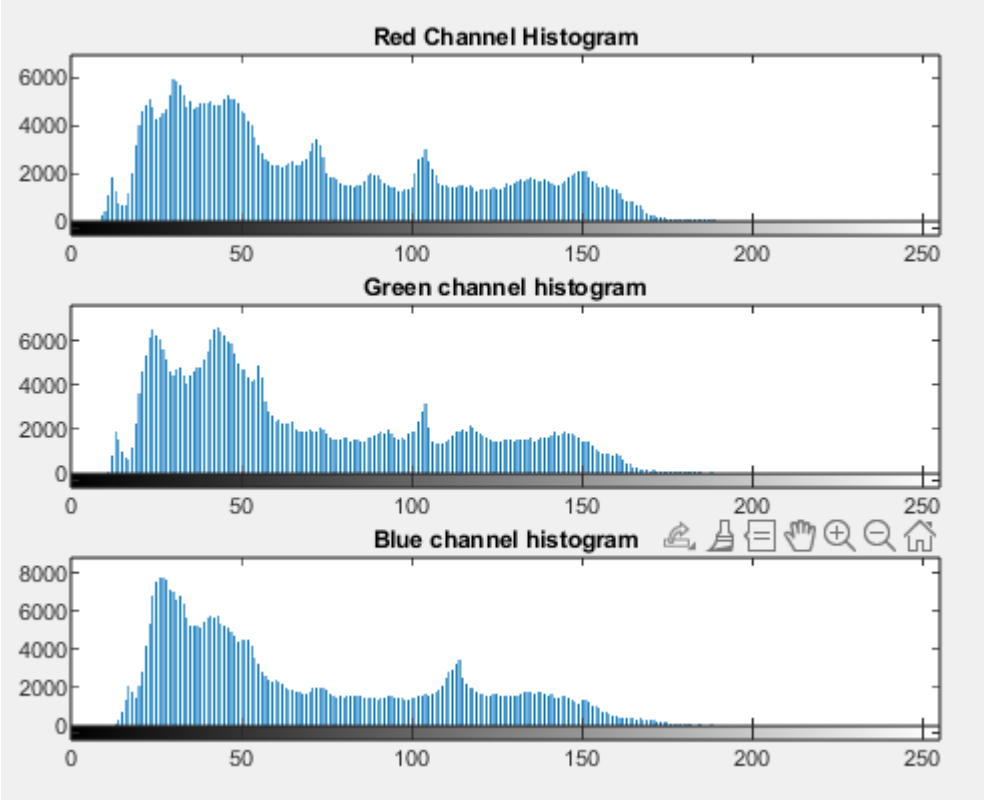


Figure 2.41: Histogram of 4.5 NL Image

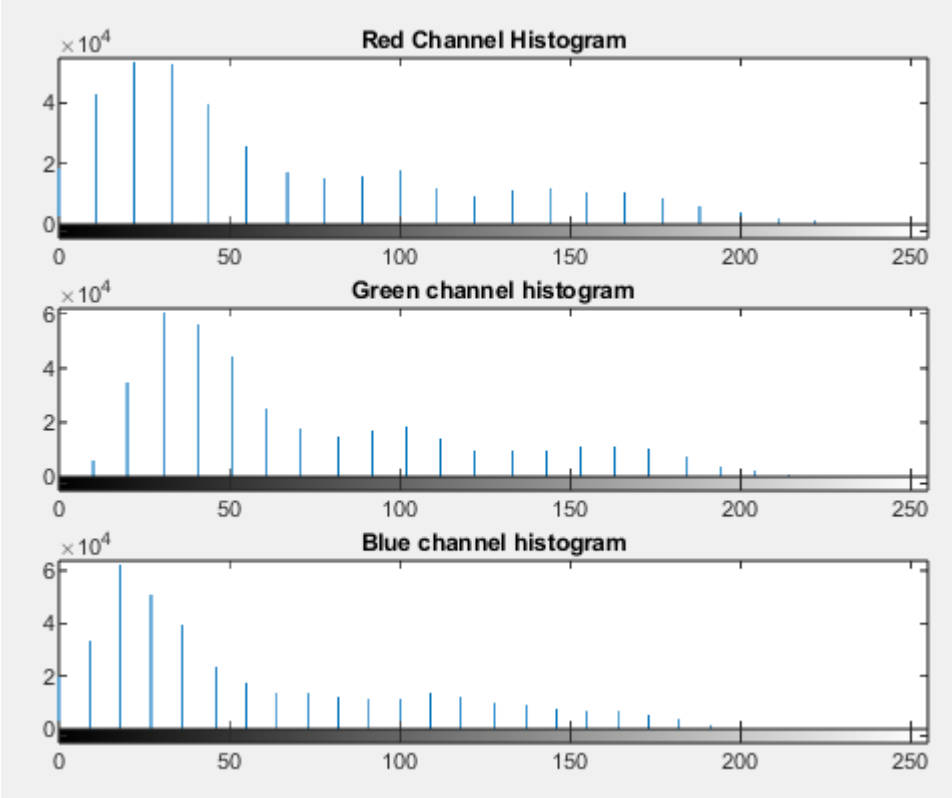
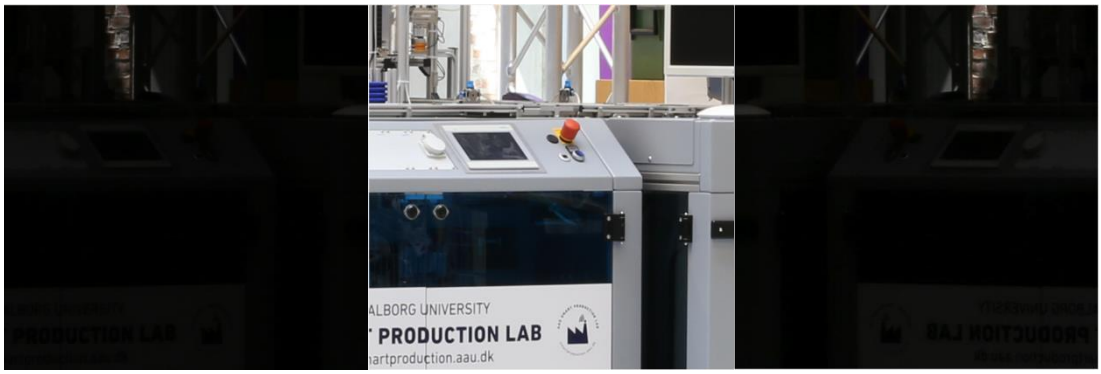


Figure 2.42: Histogram of 4.5 experimental result

In the case of the maximum experimental PSNR, we have a ground truth image with a high contrast histogram, which spans the entire available value range, so we expect the algorithm to be able to recover a large part of the visual information. Indeed, the experimental result is visually very close to the ground truth image, with the histogram having the same form as the ground truth. However, we have to comment that the experimental result shows chromatic aberrations, which is due to the fact that the images have now become too dark, making it difficult for the algorithm to recover all the visual information. We can see this by comparing the experimental histogram with the corresponding ground truth. In particular, we see that in the experimental result the pixel values are distributed in fewer gray levels, compared to the ground truth case, which means that we have less visual information, leading to the observed color changes.

Darkness Level: 5.0

MIN PSNR



Original Low Light **Normal Light** **Experimental Result**

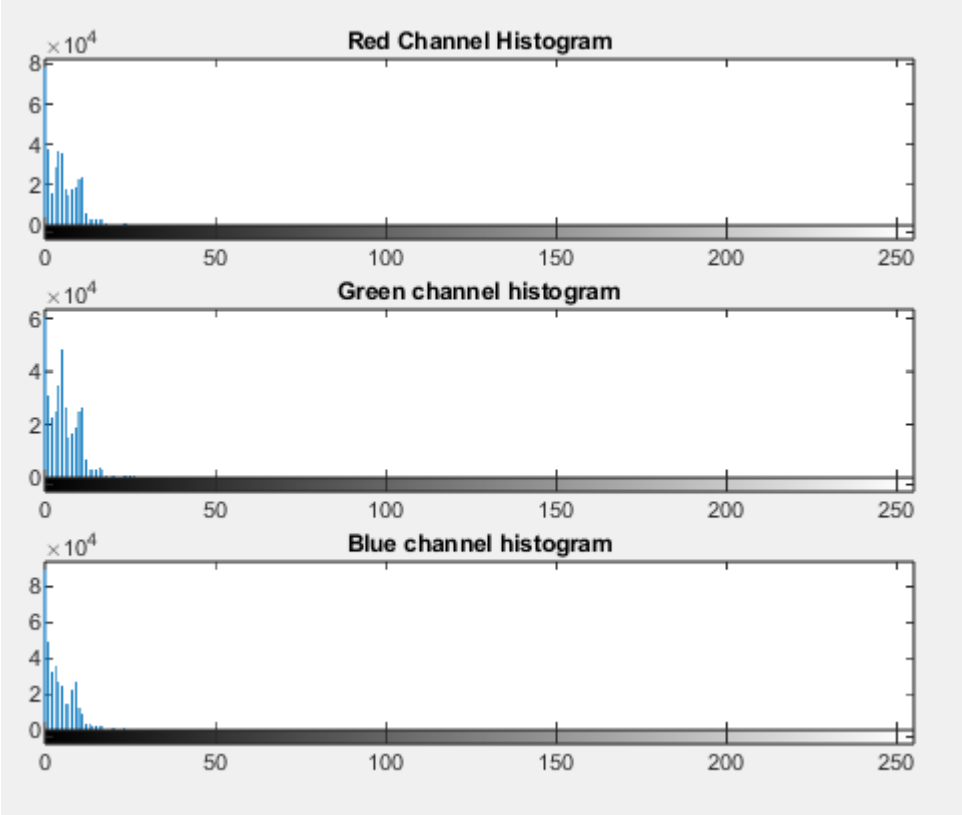


Figure 2.43: Histogram of 5.0 LL Image

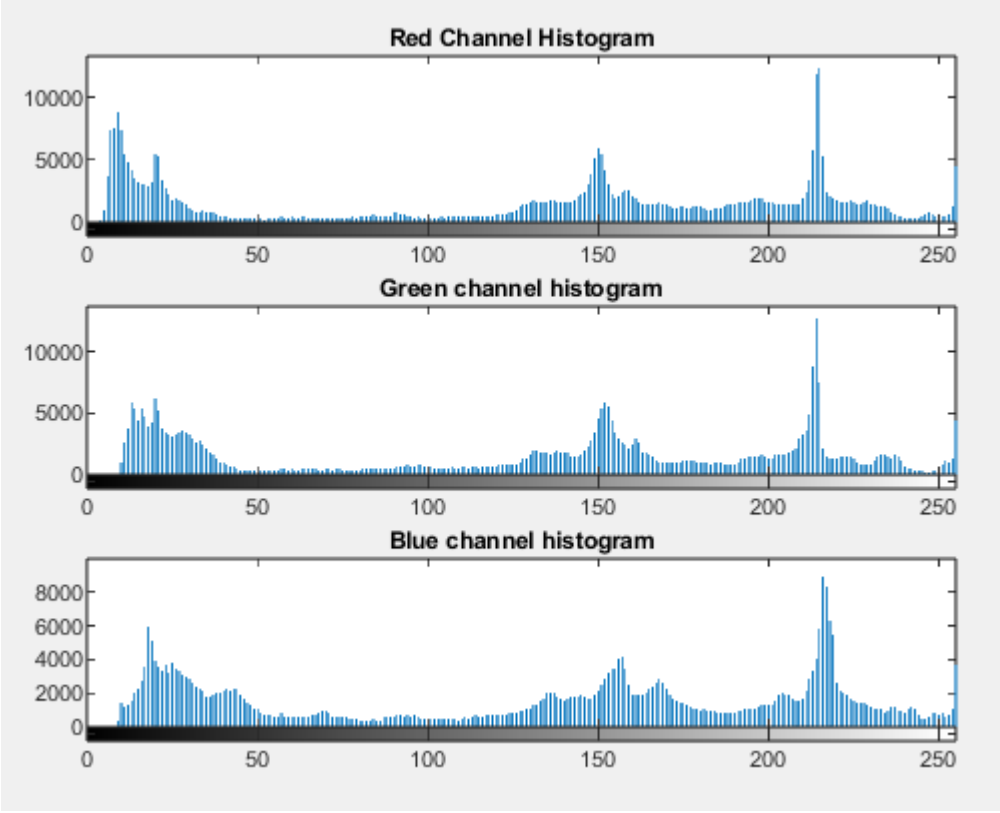


Figure 2.44: Histogram of 5.0 NL Image

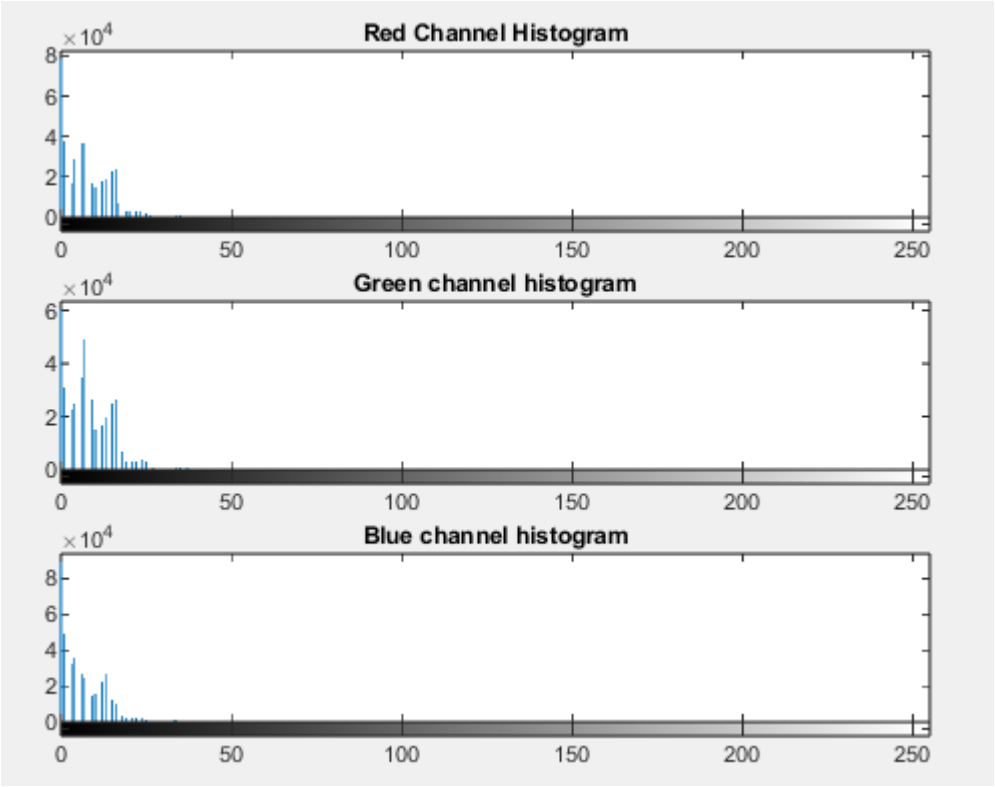
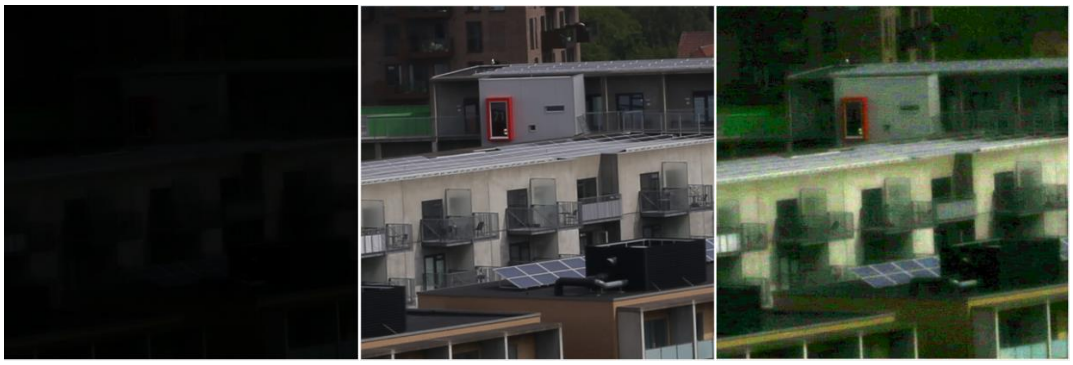


Figure 2.45: Histogram of 5.0 experimental result

At Darkness Level 5.0, the case of minimum experimental PSNR corresponds to an image with a ground truth histogram spread over the entire value range. We would expect the method to be able to recover most of the visual information. But as we can see it fails to do so, with the experimental result remaining dark and the corresponding histogram remaining piled up on the left side, close to 0. This is due to the fact that we are at the highest darkness level, as a result of which the images are too dark making it difficult to retrieve visual information.

MAX PSNR



Original Low Light **Normal Light** **Experimental Result**

In the case of the maximum experimental PSNR, we see that the ground truth histogram is spread over the entire range of available values and not concentrated at one point. According to the comment we made above, we would expect the method not to give good results, and the experimental image to remain dark. But we see that in this case, the linear transformation manages to recover a large part of the visual information, with the experimental result being, visually, very close to the ground truth image. Nevertheless, there are color distortions, which are more intense compared to the previous darkness levels. This is due to the fact that the images have become too dark, making it difficult to restore the color information.

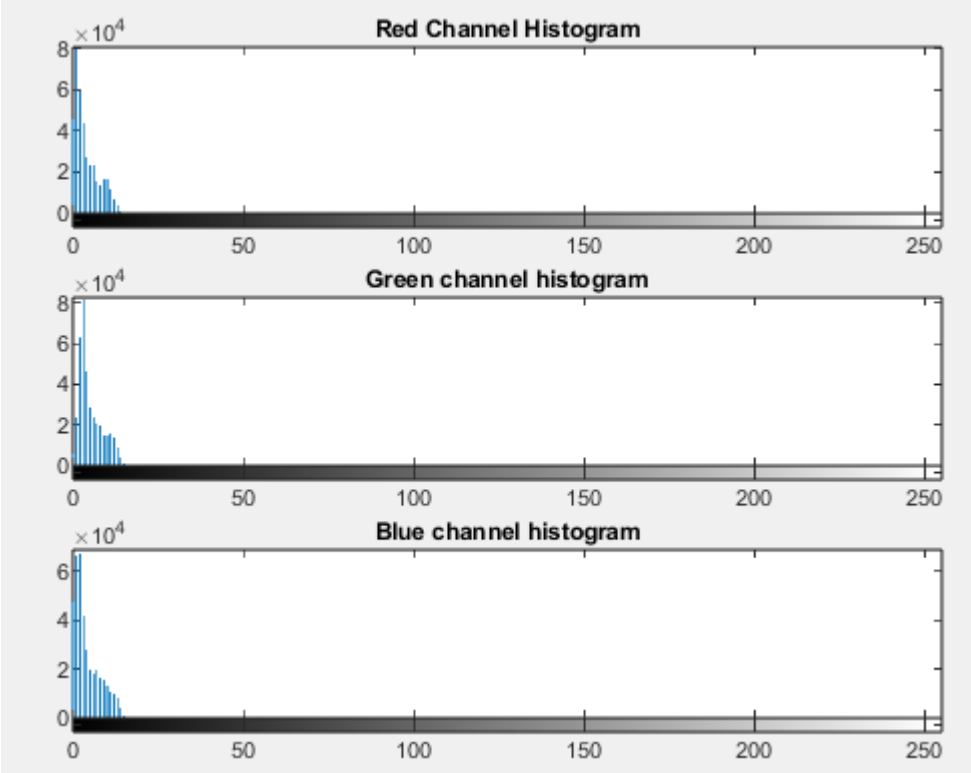


Figure 2.46: Histogram of 5.0 LL Image

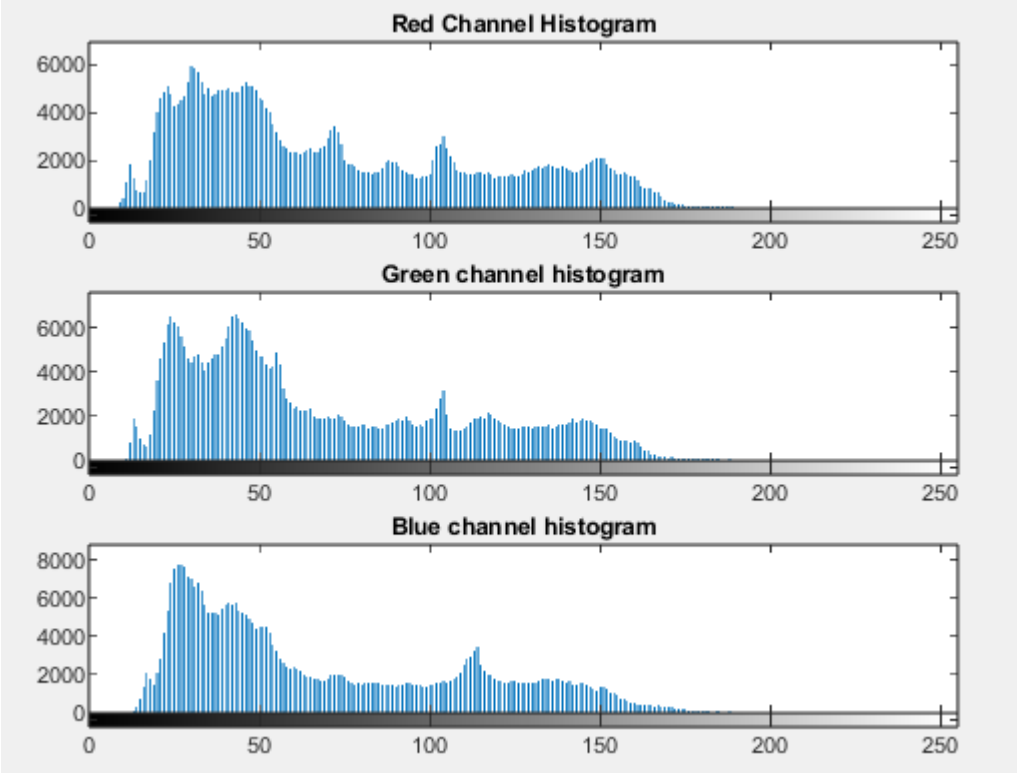


Figure 2.47: Histogram of 5.0 NL Image

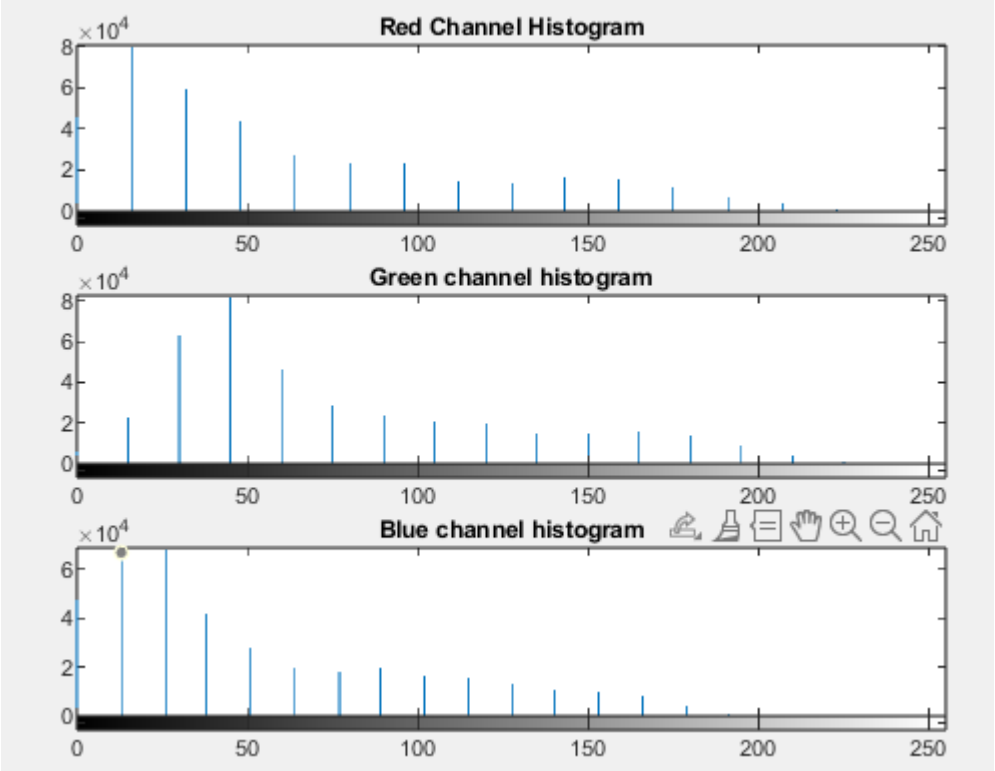


Figure 2.48: Histogram of 5.0 experimental result

At this point it is worth commenting on the results as a whole, starting with the case of the images corresponding to the minimum experimental PSNR. From the above, we notice that the method has difficulty in images that consist mainly of white pixels, i.e. in images where the pixel values in the histogram are accumulated in the right part of it, close to the value 255. The purpose of the linear transformation is to take an image that has low contrast, and output a high-contrast image whose histogram is spread over all available luminance values. Based on this, we understand that the behavior we mentioned for the images with the minimum PSNR is expected, as the ground truth images are characterized by low contrast, with the difference that the pixel values are concentrated in the right part of the histogram. So, by its very nature, it is impossible for the algorithm to retrieve this kind of information. An exception to this is Darkness Level 5.0, where all we can comment on is that the image is so dark that it becomes almost impossible to retrieve visual information, in most cases of images.

For the images corresponding to the maximum experimental PSNR, at all darkness level values, the method gives very good results since the experimental result is optically very close to the ground truth case, recovering a large part of the visual information. In addition, in all cases,

the experimental histogram has a similar form to the ground truth, with the only difference being that in the experimental, the graylevel values are distributed in fewer levels of the histogram, compared to the ground truth case. This becomes more pronounced as the darkness level increases, which is to be expected since LL images become too dark and due to the very narrow gray range between the bands it becomes too difficult to recover the correct distribution of values. This results in the chromatic aberrations observed, which become more pronounced as the darkness level increases.

In summary, the linear transformation, although simple, managed to increase the average PSNR of the images by 5.3dB (on average), and the SSIM by 0.3, noticeably improving the LL images and recovering much of the visual information. A noted weakness is that it struggles with images that contain mostly white, i.e. images for which the graylevel values are clustered on the right side of the histogram. This is due to the very nature of the method, as the linear transformation aims to take a low-contrast image, and output a high-contrast image whose graylevel values are evenly spread over the entire available range. So, it is practically impossible to reproduce the result we described above. In addition, color distortions of the experimental results were observed, which become more intense as the darkness level increases. These distortions are due to the fact that by increasing the darkness level, the images become too dark, making it difficult to retrieve color information. This can be confirmed by studying the histograms of the LL images and the experimental results. For the LL images, as we increase the darkness level, the graylevel values accumulate more and more intensively in the left part of the histogram, with the dynamic range between the color bands decreasing, and becoming almost zero, as a result of which the algorithm is struggling to retrieve the correct color information. For the histograms of the experimental results, we notice that as the darkness level increases, the graylevel values are distributed in fewer and fewer levels in the histogram, indicating that we have less visual/color information at our disposal, confirming that the method struggles to retrieve it.

The problems we described above are to be expected, as we implemented a very simplistic algorithm, using a simple linear function. In the next two subsections, we go a step further, increasing the complexity of the algorithm by applying non-linear functions to the value of each pixel. Specifically, we will apply gamma transformation and logarithmic transformation, commenting on the results.

2.2 Gamma Correction

This is a point transformation, like the linear one, except that in this case we apply a non-linear function. Specifically, we raise the value of each pixel to a power denoted by γ, and is called the correction constant [15]. The mathematical formula we apply is:

$$I(i,j) = 255 * \left(\frac{I_{LL}(i,j)}{255}\right)^{\gamma}$$

Based on the value of γ, the gamma transform can take several forms, which are shown in the figure below:

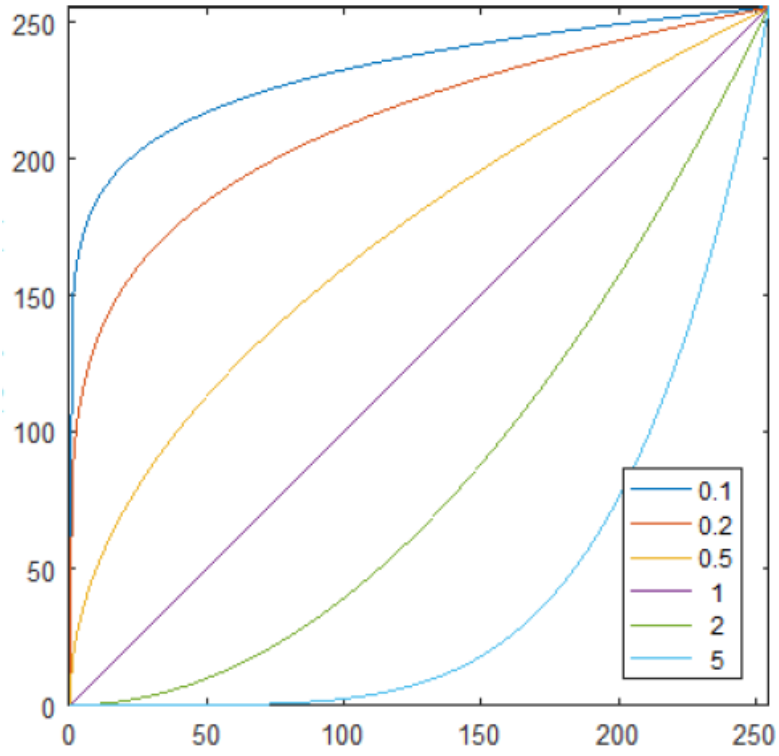


Figure 2.49: Gamma correction for different γ values[9]

We notice that for γ<1 the transform can enhance the dark areas of an image while preserving the bright areas, making it a suitable algorithm for LLIE. This is also a simple algorithm in terms of implementation, which is done

[9] From Image Processing courses' presentations on point transformations (AIVC 2021-2022)

by the function shown in Figure B.2.5, of Appendix B. Obviously there is no ideal value for the correction constant, so we will do 3 different experiments for 3 different values of the exponent. Specifically, we will use a small value γ=0.1, a medium range value γ=0.3 and a large value γ=0.8. We notice that all 3 values are smaller than 1, since we are interested in enhancing the dark areas of each image. The experimental procedure is implemented with the script presented in Figures B.2.6 and B.2.7 of Appendix B. After the application of this script, we have at our disposal the experimental results, the evaluation of which is done with the program presented in figure B.2.8 of appendix B. This results in 3 excel files for each value of the correction constant, which contain the metric values for each image, per darkness level. We use these values for the construction of summary tables, presented below, as well as for line charts with the maximum/minimum/average value of PSNR and SSIM per Darkness Level. In addition, we also present line charts of the average PSNR and SSIM both for the experimental results (for each value of the correction constant) and for the original LL images, per Darkness Level, to compare how much the result improved in each case.

Training Set

Gamma: 0.1

MSE					
Level	3.0	3.5	4.0	4.5	5.0
MIN	3492,335	2913,357	2209,13	1763,437	1393,58
MAX	20898,08	18814,88	16538,25	14034,5	11870,66
AVERAGE	9656,502	8303,092	7022,815	5842,988	4982,011
	PSNR(dB)				
Level	3.0	3.5	4.0	4.5	5.0
MIN	4,92974	5,385788	5,945909	6,658835	7,386053
MAX	12,69964	13,48687	14,68859	15,6672	16,68949
AVERAGE	8,462102	9,135615	9,873598	10,67336	11,35184
	SSIM				
Level	3.0	3.5	4.0	4.5	5.0
MIN	0,138094	0,0746	0,072032	0,070804	0,07431
MAX	0,784723	0,764291	0,792453	0,773365	0,660347
AVERAGE	0,43329	0,416952	0,393062	0,360888	0,316111
	MV				
Level	3.0	3.5	4.0	4.5	5.0
MIN	120,9224	104,6519	93,97881	85,44764	79,44174
MAX	201,6598	194,7715	187,9614	181,0711	174,7407
AVERAGE	183,8029	174,6706	164,3499	152,2717	140,1474

	STD				
Level	3.0	3.5	4.0	4.5	5.0
MIN	33,72569	38,89726	46,72629	59,29285	67,64118
MAX	1753,653	1930,043	1737,16	1763,661	1509,233
AVERAGE	332,5491	383,1862	454,6074	542,2311	609,6619
	BRISQUE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	7,30566	8,238828	10,33431	8,726207	14,80831
MAX	53,42801	52,09375	50,56757	50,83908	56,06952
AVERAGE	34,01972	30,15561	26,92062	27,10983	30,46322
	NIQE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	2,733553	2,733897	2,836254	2,896855	3,084281
MAX	9,966678	11,21531	10,33209	11,1693	13,15784
AVERAGE	4,184109	4,288651	4,558946	4,973441	5,543359

Table 2.7: Gamma correction results with gamma=0.1 on training set

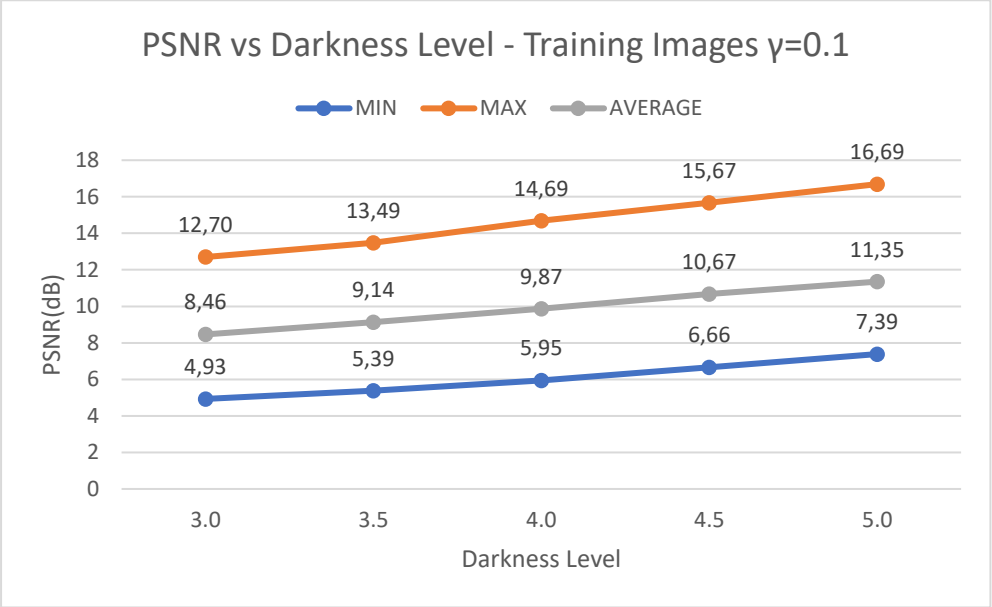


Figure 2.50: Experimental results PSNR vs darkness level for training images

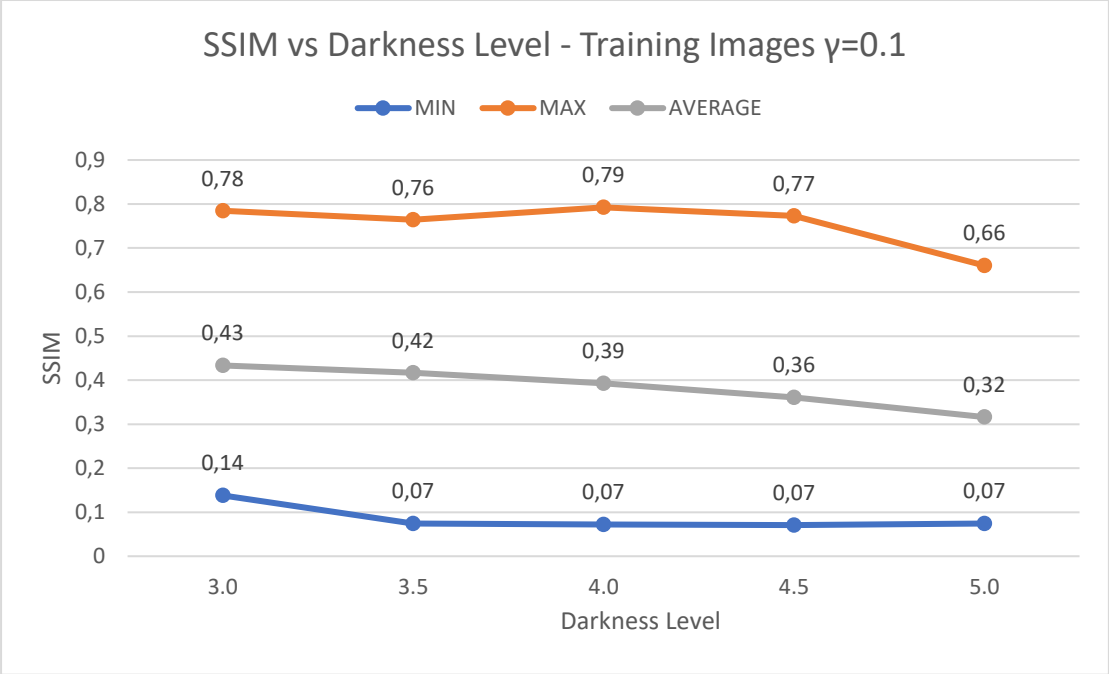


Figure 2.51: Experimental results SSIM vs Darkness Level for training images

For a correction constant of 0.1 a phenomenon is observed which is opposite to what we have explained and seen so far, the PSNR increases with the increase of the darkness level, and the SSIM remains almost constant. Nevertheless, if we look at figure 2.56, we see that the experimental PSNR is smaller than the corresponding LL images and as the darkness level increases it becomes slightly better than the original. Similar behavior for the average experimental SSIM, which is only slightly better than the original one. This is because we use too small correction constant value, the dark areas are over-enhanced, and the result resembles an image taken in over-exposed conditions. We can confirm this from the values of the metrics. MSE and SSIM decrease, while PSNR increases, which means that in terms of brightness there is an improvement, due to over-amplification, but we lose visual information (color, texture, etc.). MV and STD increase, which means that the brightness of the experimental results increases, with a greater variation of pixel values, confirming the over-amplification we mentioned above. Finally, BRISQUE appears to decrease, while NIQE increases with increasing darkness level. Nevertheless, their values remain high, which means that we are far from natural statistics and the quality of the images is low.

Gamma: 0.3

MSE					
Level	3.0	3.5	4.0	4.5	5.0
MIN	77,67357	174,9375	217,7657	356,7575	411,034
MAX	4999,258	6527,41	8416,371	10295,27	12372,87
AVERAGE	1088,054	1165,344	1486,076	2036,705	2859,597
	PSNR(dB)				
Level	3.0	3.5	4.0	4.5	5.0
MIN	11,14175	9,983395	8,879555	8,004427	7,2061
MAX	29,22807	25,70198	24,75091	22,60707	21,99203
AVERAGE	18,27623	18,0992	17,15091	15,76744	14,31027
	SSIM				
Level	3.0	3.5	4.0	4.5	5.0
MIN	0,2624	0,166688	0,173362	0,12898	0,079298
MAX	0,905391	0,87733	0,834898	0,728988	0,665505
AVERAGE	0,625351	0,585148	0,530487	0,462446	0,383664
	MV				
Level	3.0	3.5	4.0	4.5	5.0
MIN	52,49691	43,30077	36,96539	31,8751	30,00851
MAX	126,3678	113,7167	102,4598	91,86411	86,89999
AVERAGE	102,63	91,22191	80,30041	69,747	60,55807
	STD				
Level	3.0	3.5	4.0	4.5	5.0
MIN	28,52611	38,21951	47,61383	55,1259	51,11637
MAX	1100,658	1020,813	840,9665	730,1673	568,798
AVERAGE	288,1971	261,18	241,2811	227,3242	213,4258
	BRISQUE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	12,91418	8,124384	7,212181	10,75759	7,86179
MAX	51,79565	51,43248	49,08752	51,30185	53,2092
AVERAGE	32,73577	30,65061	29,04878	29,23709	31,08535
	NIQE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	2,281133	2,205605	2,522284	2,845994	2,980289
MAX	27,7329	14,82724	7,045798	7,259057	8,408454
AVERAGE	3,861293	3,905891	4,026759	4,234008	4,520591

Table 2.8: Gamma correction results with gamma=0.3 on training set

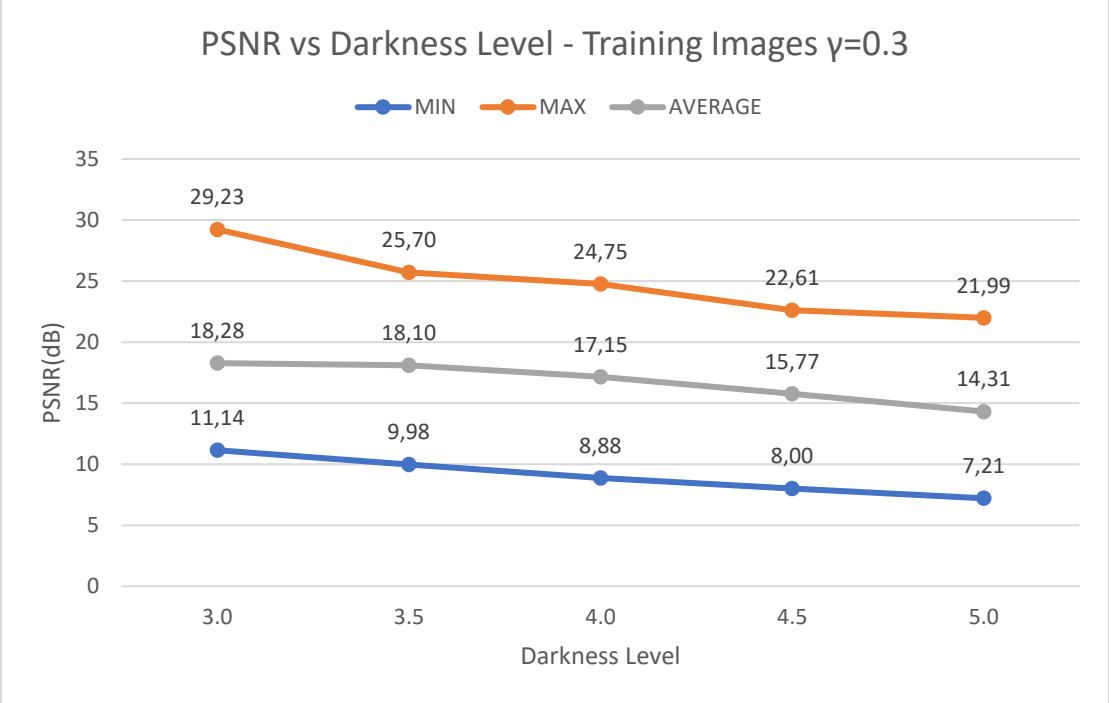


Figure 2.52: Experimental results PSNR vs Darkness Level for training images

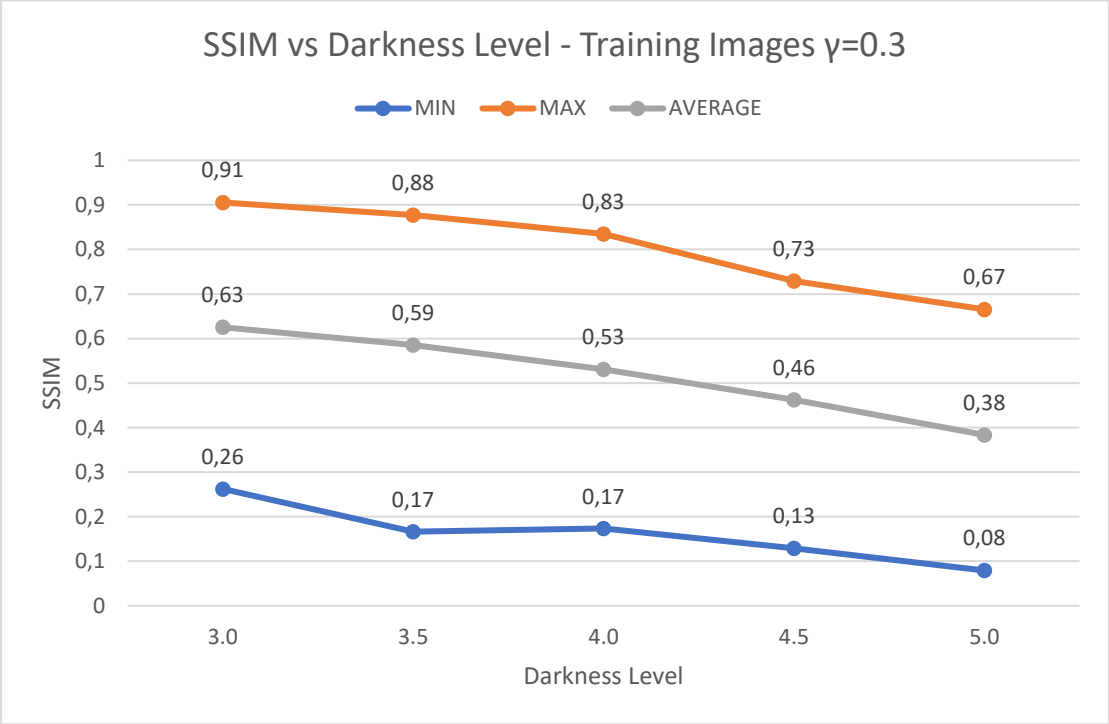


Figure 2.53: Experimental results SSIM vs Darkness Level for training images

For a correction constant of 0.3 the best results are obtained. From Figures 2.56 and 2.57 we see that the PSNR and SSIM increase significantly, especially compared to the other fixed correction cases, indicating the

improvement of the result. From the values of the quality metrics, we observe the expected behavior. MSE increases, and PSNR and SSIM decrease with increasing darkness level. For the metrics without reference, we observe that MV and STD decrease with increasing darkness level, meaning that the experimental results become darker with pixel values clustering more strongly around the mean brightness. BRISQUE appears to decrease, but at darkness level 5.0 its value increases again, while NIQE increases as the darkness level increases, which means that as the darkness level increases, the experimental result moves away from the natural statistics. This is to be expected as the LL images are so dark that it is very difficult to recover all the visual information.

Gamma: 0.8

MSE					
Level	3.0	3.5	4.0	4.5	5.0
MIN	235,4911	638,7423	818,7235	978,2881	1102,84
MAX	18599,15	20853,84	22573,11	23968,91	25075,65
AVERAGE	6245,303	7493,574	8559,884	9459,695	10350,67
	PSNR(dB)				
Level	3.0	3.5	4.0	4.5	5.0
MIN	5,435872	4,938944	4,594889	4,334321	4,138282
MAX	24,41106	20,07755	18,99943	18,22614	17,70568
AVERAGE	10,61229	9,75991	9,146126	8,68727	8,296137
	SSIM				
Level	3.0	3.5	4.0	4.5	5.0
MIN	0,090189	0,063361	0,047123	0,030086	0,019451
MAX	0,900089	0,84427	0,723619	0,578283	0,429113
AVERAGE	0,333679	0,237395	0,165022	0,113012	0,077469
	MV				
Level	3.0	3.5	4.0	4.5	5.0
MIN	7,706682	5,479142	4,01214	2,960862	2,56154
MAX	53,15215	41,73805	32,42197	24,91649	18,68465
AVERAGE	27,17741	20,44234	15,19282	11,15423	8,272058
	STD				
Level	3.0	3.5	4.0	4.5	5.0
MIN	8,157182	6,703032	4,874029	4,382126	3,802528
MAX	555,2159	425,6451	319,3716	235,7316	166,5757
AVERAGE	102,1569	69,65883	47,04096	31,50935	21,05261

	BRISQUE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	19,25237	24,59563	30,63588	26,80585	32,21923
MAX	52,24875	50,75109	50,85141	50,66531	49,1188
AVERAGE	40,43959	42,28604	43,38832	43,94171	44,09995
	NIQE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	3,07697	2,947374	3,360021	3,646621	3,706228
MAX	28,89557	29,00329	8,645698	7,143216	7,207026
AVERAGE	4,891538	5,039388	5,11548	5,189293	5,229769

Table 2.9: Gamma correction results with gamma=0.8 on training set

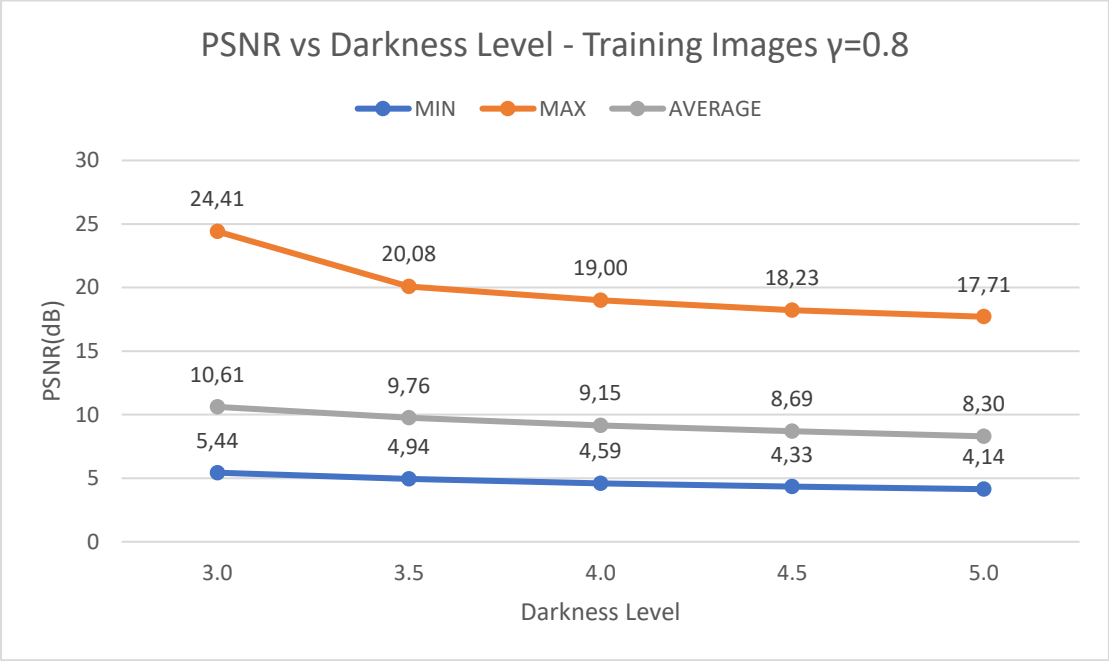


Figure 2.54: Experimental results PSNR vs Darkness Level for training images

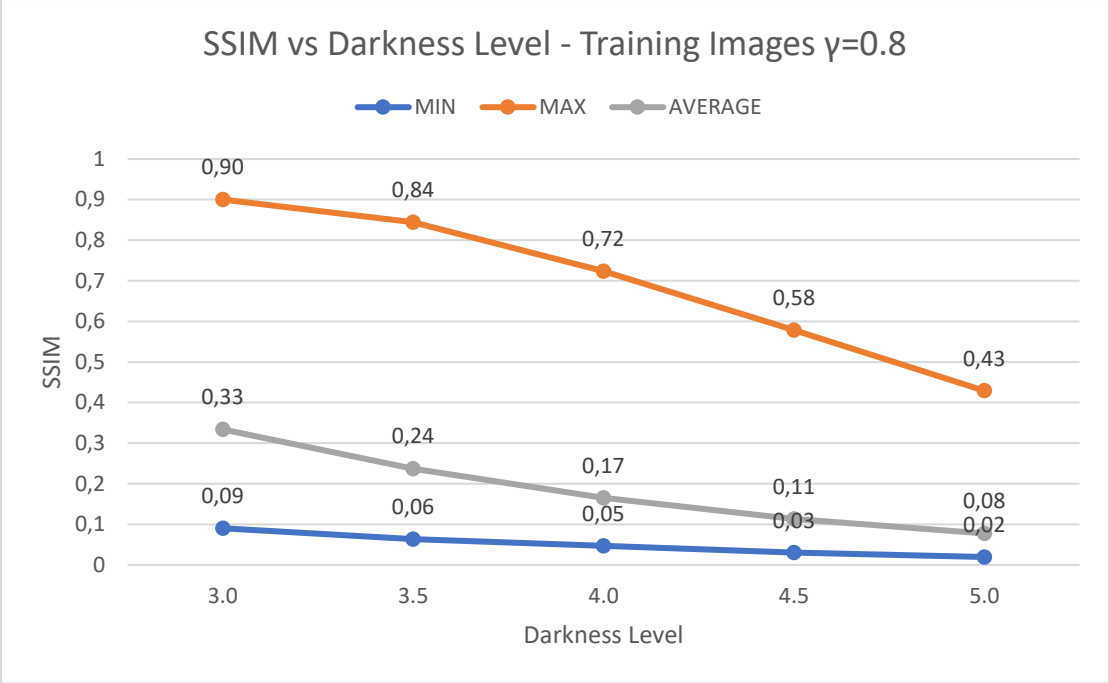
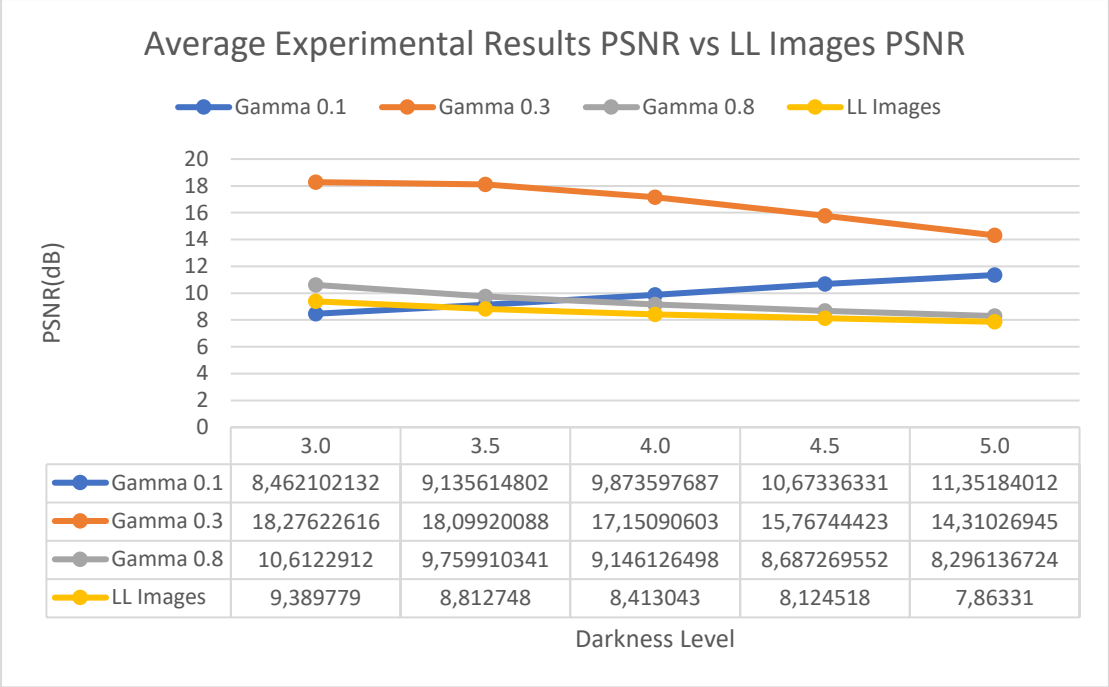


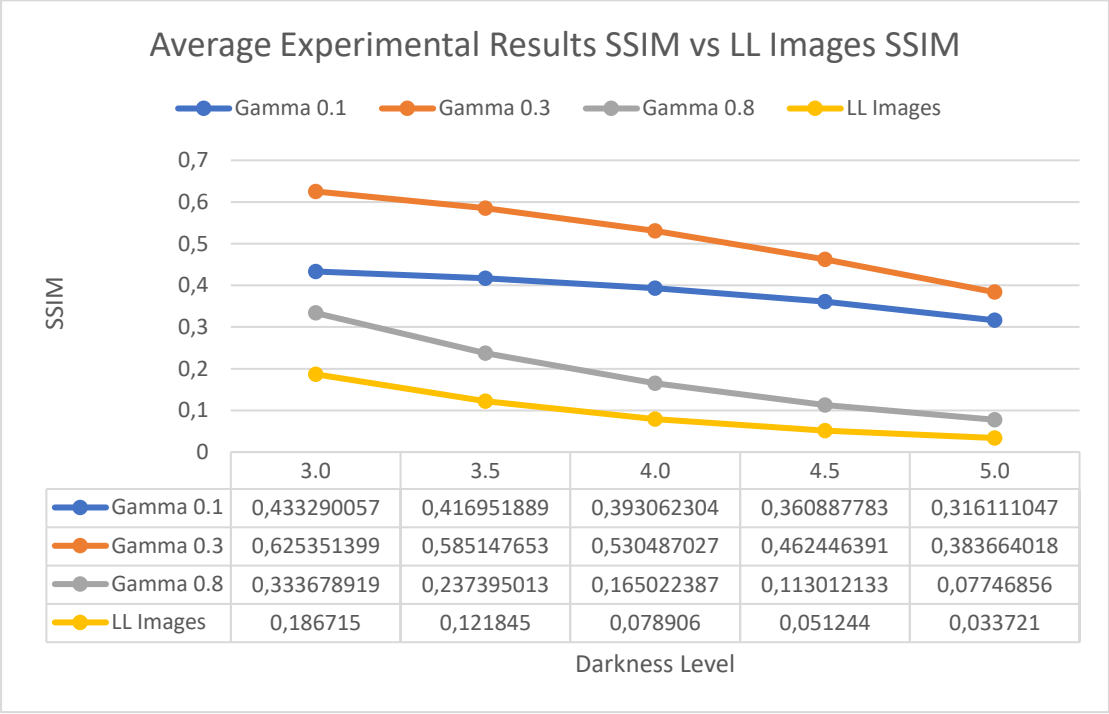
Figure 2.55: Experimental results SSIM vs Darkness Level for training images

First let's comment that for a correction constant of 0.8, the LL images improve little, if at all. From figure 2.56 we see that the average experimental PSNR is almost the same as that of the LL images, while from figure 2.57 we notice that the SSIM increases slightly compared to the equivalent of the LL images. This is to be expected, as 0.8 is quite close to unity, meaning it leaves images almost the same, with the enhancement of dark areas being minimal. Regarding the quality metrics, we notice that as the darkness level increases, the MSE increases and the PSNR, SSIM decrease. This means that as the darkness level increases, the performance of the algorithm decreases. The same is observed in the metrics without reference, where MV and STD decrease, with the increase of darkness level, which means that the experimental result is darker and the accumulation of gray value values in the left part of the histogram is more intense as the darkness level increases. Finally, BRISQUE and NIQE increase with the increase of the darkness level, which means that we move further and further away from natural statistics, i.e. the quality of the result decreases.



	3.0	3.5	4.0	4.5	5.0
Gamma 0.1	8,462102132	9,135614802	9,873597687	10,67336331	11,35184012
Gamma 0.3	18,27622616	18,09920088	17,15090603	15,76744423	14,31026945
Gamma 0.8	10,6122912	9,759910341	9,146126498	8,687269552	8,296136724
LL Images	9,389779	8,812748	8,413043	8,124518	7,86331

Figure 2.56: Experimental results average PSNR vs LL Images PSNR training set



	3.0	3.5	4.0	4.5	5.0
Gamma 0.1	0,433290057	0,416951889	0,393062304	0,360887783	0,316111047
Gamma 0.3	0,625351399	0,585147653	0,530487027	0,462446391	0,383664018
Gamma 0.8	0,333678919	0,237395013	0,165022387	0,113012133	0,07746856
LL Images	0,186715	0,121845	0,078906	0,051244	0,033721

Figure 2.57: Experimental results average SSIM vs LL Images SSIM training set

Validation Set

Gamma: 0.1

MSE					
Level	3.0	3.5	4.0	4.5	5.0
MIN	4716,067	4024,531	3303,853	2913,828	2459,264
MAX	14488,02	12326,77	10161,99	8703,311	7534,811
AVERAGE	8925,79	7618,675	6405,466	5312,38	4417,184
	PSNR(dB)				
Level	3.0	3.5	4.0	4.5	5.0
MIN	6,520713	7,222311	8,061014	8,733959	9,36008
MAX	11,395	12,08365	12,9406	13,48616	14,22275
AVERAGE	8,76159	9,455652	10,21875	11,03285	11,82816
	SSIM				
Level	3.0	3.5	4.0	4.5	5.0
MIN	0,134004	0,107176	0,099659	0,097444	0,079812
MAX	0,715257	0,70296	0,670455	0,627034	0,531918
AVERAGE	0,418485	0,405261	0,386034	0,356182	0,314588
	MV				
Level	3.0	3.5	4.0	4.5	5.0
MIN	121,8897	107,9297	95,06113	83,04376	72,29713
MAX	197,1613	189,545	182,0081	173,4699	162,6872
AVERAGE	181,7992	172,7059	162,7023	151,1262	139,8752
	STD				
Level	3.0	3.5	4.0	4.5	5.0
MIN	44,24661	55,27024	67,43236	134,1586	169,3051
MAX	1039,836	1079,864	1083,361	1220,335	1274,665
AVERAGE	366,2691	406,7496	485,8542	574,629	661,4552
	BRISQUE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	19,52548	14,57927	16,43556	16,10552	18,73141
MAX	49,12868	45,45907	45,50902	47,4785	45,14583
AVERAGE	34,93686	31,9376	28,21715	27,37557	28,83888
	NIQE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	3,10295	2,920346	3,083566	3,2515	3,685535
MAX	5,58288	7,080189	7,782515	9,071174	9,721552
AVERAGE	4,167536	4,285659	4,512699	4,790659	5,545879

Table 2.10: Gamma correction results with gamma=0.1 on validation set

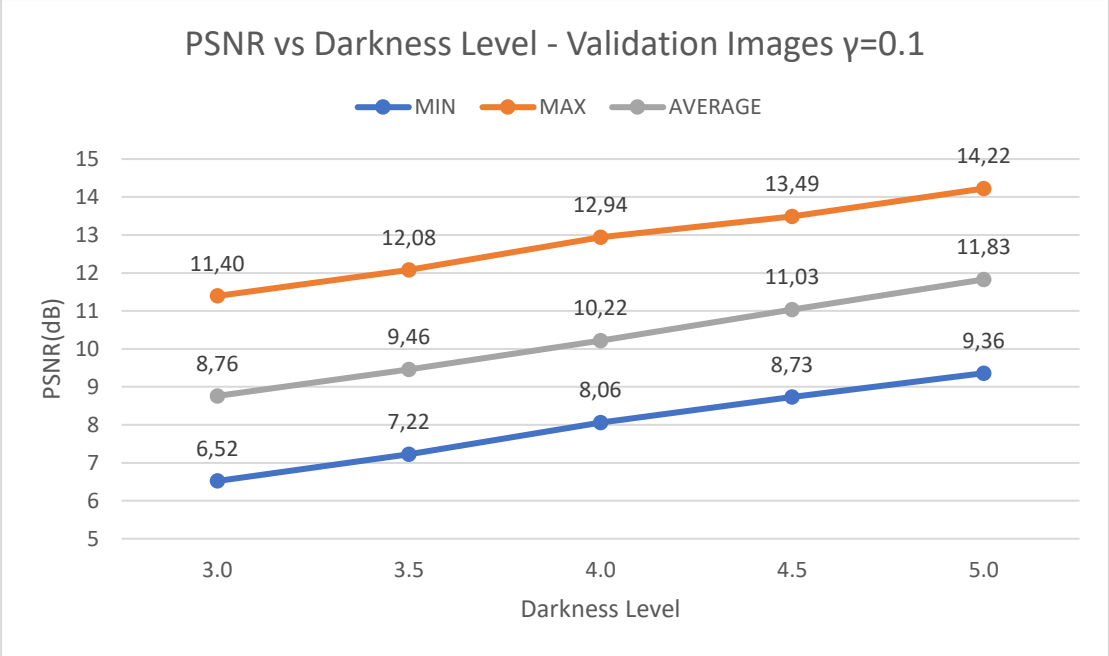


Figure 2.58: Experimental results PSNR vs Darkness Level for validation images

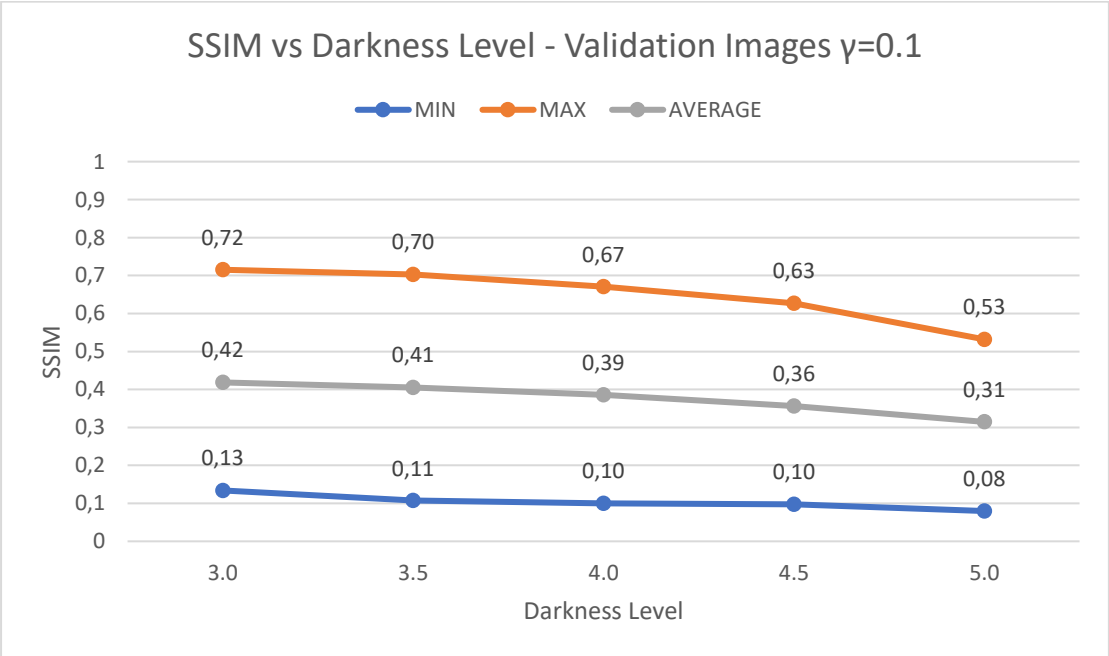


Figure 2.59: Experimental results SSIM vs Darkness Level for validation images

Gamma: 0.3

MSE					
Level	3.0	3.5	4.0	4.5	5.0
MIN	267,843	230,9084	425,646	738,7127	856,3532
MAX	2304,938	3317,243	4518,973	5774,478	7306,562
AVERAGE	1014,28	1167,498	1556,901	2154,386	3053,646
	PSNR(dB)				
Level	3.0	3.5	4.0	4.5	5.0
MIN	14,50421	12,92303	11,58041	10,51568	9,493673
MAX	23,852	24,49641	21,84032	19,44605	18,80427
AVERAGE	18,50834	17,97769	16,76959	15,32995	13,84067
	SSIM				
Level	3.0	3.5	4.0	4.5	5.0
MIN	0,33885	0,319395	0,280134	0,227697	0,187511
MAX	0,857265	0,820759	0,75274	0,646054	0,534375
AVERAGE	0,609748	0,57124	0,520152	0,456374	0,378221
	MV				
Level	3.0	3.5	4.0	4.5	5.0
MIN	57,3758	48,914	41,17625	31,55225	25,70099
MAX	120,6786	108,1126	96,53481	85,42025	74,69721
AVERAGE	101,8351	90,56641	79,88459	69,61952	60,43669
	STD				
Level	3.0	3.5	4.0	4.5	5.0
MIN	51,55341	57,72348	65,08179	70,35501	87,21818
MAX	917,072	862,5249	795,6935	726,7177	663,2902
AVERAGE	306,4455	277,1235	258,5589	243,794	232,7557
	BRISQUE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	19,94405	20,15696	16,79414	16,31987	11,59378
MAX	47,22796	46,42544	48,30353	49,67849	46,94207
AVERAGE	34,05078	32,32407	29,77512	29,68226	29,76644
	NIQE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	2,717332	2,913007	2,942696	3,082985	3,076332
MAX	4,988848	5,646397	5,590788	6,18522	7,292711
AVERAGE	3,780538	3,889061	4,002844	4,163379	4,414815

Table 2.11: Gamma correction results with gamma=0.3 on validation set

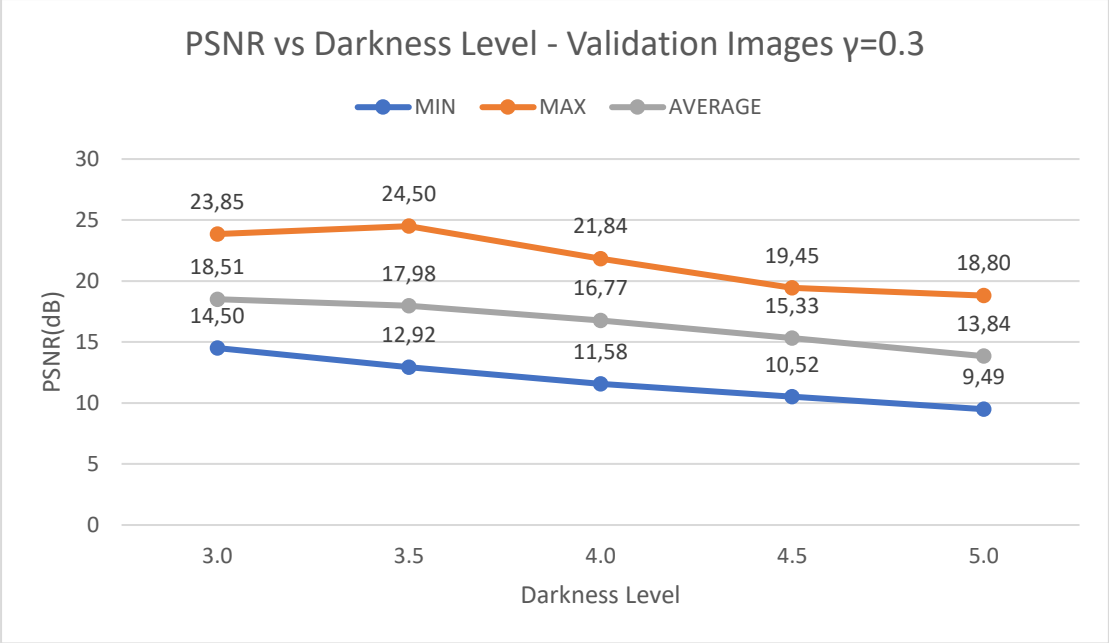


Figure 2.60: Experimental results PSNR vs Darkness Level for validation images

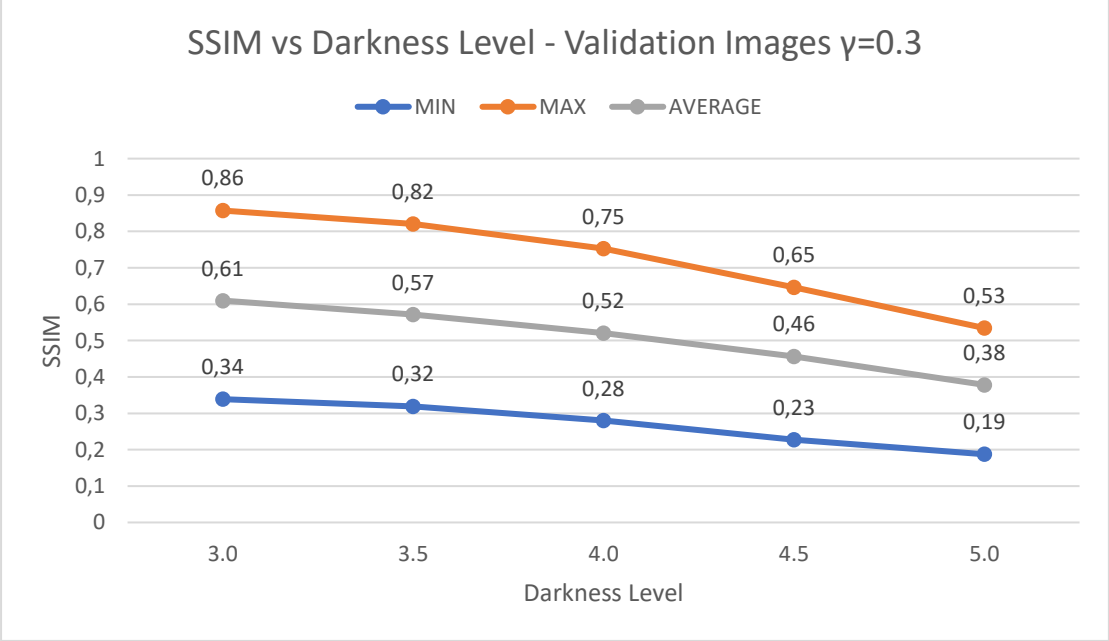


Figure 2.61: Experimental results SSIM vs Darkness Level for validation images

Gamma: 0.8

MSE					
Level	3.0	3.5	4.0	4.5	5.0
MIN	1042,077	1232,974	1392,912	1519,05	1604,615
MAX	14014,97	15906,57	17436,79	18598,68	19565,15
AVERAGE	6682,69	7989,379	9097,949	10013,54	11032,63
	PSNR(dB)				
Level	3.0	3.5	4.0	4.5	5.0
MIN	6,664881	6,115037	5,716138	5,435981	5,215971
MAX	17,95181	17,22126	16,69157	16,31508	16,07709
AVERAGE	10,29957	9,496342	8,914999	8,488085	8,109008
	SSIM				
Level	3.0	3.5	4.0	4.5	5.0
MIN	0,195894	0,138413	0,089786	0,056434	0,036445
MAX	0,499425	0,385344	0,28504	0,21245	0,189438
AVERAGE	0,316549	0,22328	0,155066	0,106681	0,071292
	MV				
Level	3.0	3.5	4.0	4.5	5.0
MIN	7,915052	5,663216	3,955056	2,75103	2,027544
MAX	40,09343	30,82955	23,5171	17,79241	13,12119
AVERAGE	27,25292	20,52525	15,30684	11,2737	8,259593
	STD				
Level	3.0	3.5	4.0	4.5	5.0
MIN	15,55257	11,41576	7,37885	5,041852	3,755176
MAX	591,8927	531,6727	472,207	412,9465	360,7552
AVERAGE	112,265	79,04753	56,03444	39,41447	30,0183
	BRISQUE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	33,51088	36,04395	36,75638	38,91164	40,66837
MAX	51,57402	49,54069	49,70543	47,08922	47,449
AVERAGE	40,93952	42,63984	43,77389	43,87909	43,80592
	NIQE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	3,626792	3,933872	4,040729	4,417297	4,53521
MAX	6,025387	6,097099	6,51747	6,33248	6,798714
AVERAGE	4,763561	4,92659	5,051903	5,206314	5,285254

Table 2.12: Gamma correction results with gamma=0.8 on validation set

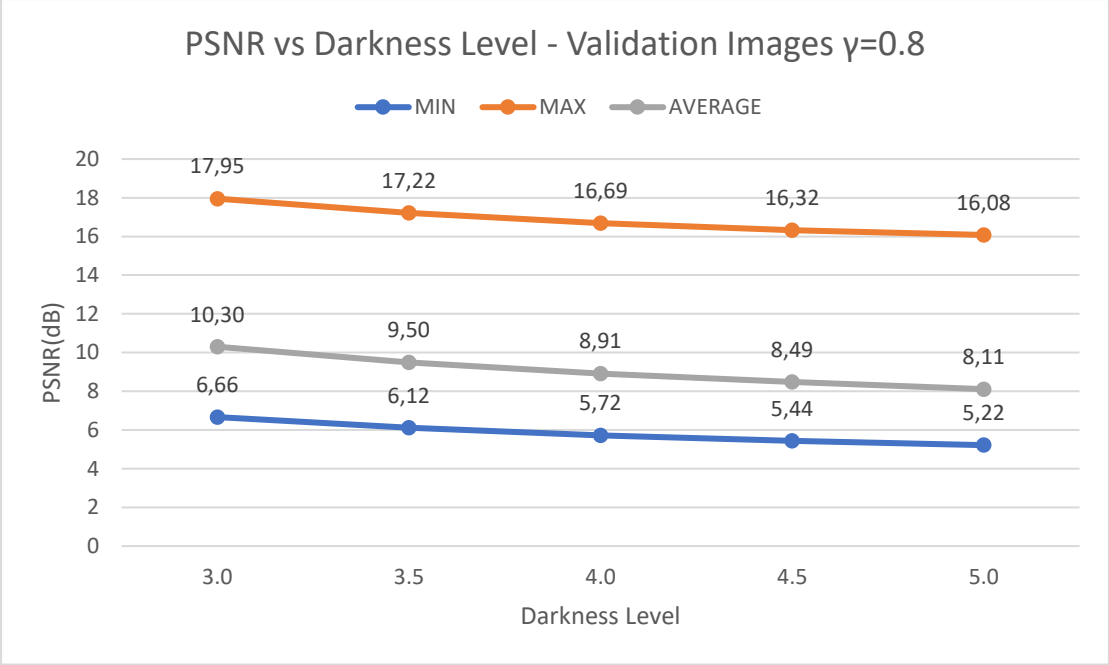


Figure 2.62: Experimental results PSNR vs Darkness Level for validation images

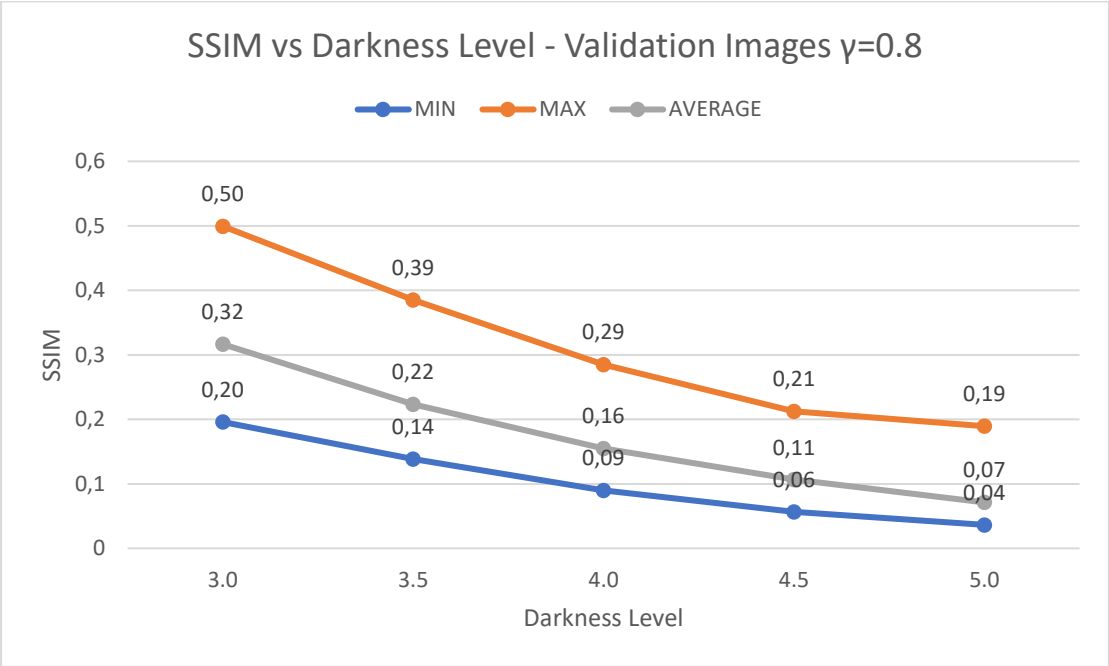
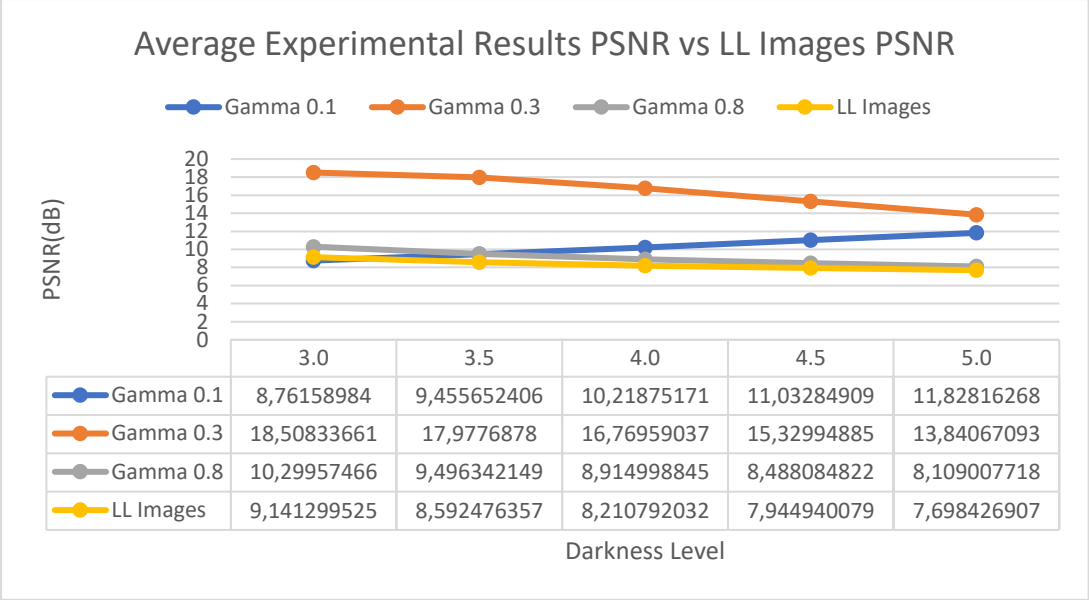
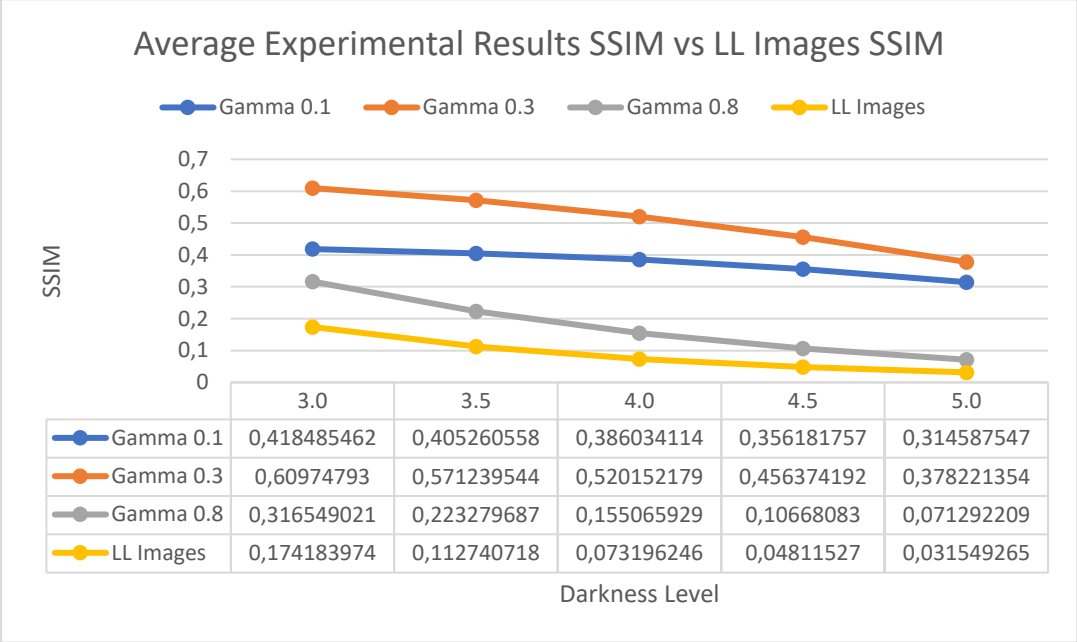


Figure 2.63: Experimental results SSIM vs Darkness Level for validation images



	3.0	3.5	4.0	4.5	5.0
Gamma 0.1	8,76158984	9,455652406	10,21875171	11,03284909	11,82816268
Gamma 0.3	18,50833661	17,9776878	16,76959037	15,32994885	13,84067093
Gamma 0.8	10,29957466	9,496342149	8,914998845	8,488084822	8,109007718
LL Images	9,141299525	8,592476357	8,210792032	7,944940079	7,698426907

Figure 2.64: Experimental results average PSNR vs LL Images PSNR validation set



	3.0	3.5	4.0	4.5	5.0
Gamma 0.1	0,418485462	0,405260558	0,386034114	0,356181757	0,314587547
Gamma 0.3	0,60974793	0,571239544	0,520152179	0,456374192	0,378221354
Gamma 0.8	0,316549021	0,223279687	0,155065929	0,10668083	0,071292209
LL Images	0,174183974	0,112740718	0,073196246	0,04811527	0,031549265

Figure 2.65: Experimental results average SSIM vs LL Images SSIM validation set

In the validation set, the same behavior that we described above is observed. For a correction constant of 0.1 an increase in PSNR is observed with the increase in darkness level accompanied by an increase in MV. This means that there is an over-enhancement of the dark areas. For a correction constant of 0.8, the experimental results are almost the same as the original ones, with the metrics improving little to none. For a correction constant of 0.3 the best experimental results are obtained with the metrics improving

noticeably and following the expected behavior where with an increase in the darkness level the performance of the method decreases.

Test set

Gamma: 0.1

MSE					
Level	3.0	3.5	4.0	4.5	5.0
MIN	4121,143	3382,117	2860,049	2314,887	2110,043
MAX	17608,52	15647,18	13558,57	11249,82	8859,983
AVERAGE	9782,883	8404,342	7080,542	5884,045	5107,944
	PSNR(dB)				
Level	3.0	3.5	4.0	4.5	5.0
MIN	5,673576	6,186443	6,808664	7,619346	8,656475
MAX	11,98063	12,83892	13,56707	14,48551	14,88789
AVERAGE	8,364756	9,039792	9,793367	10,5984	11,20805
	SSIM				
Level	3.0	3.5	4.0	4.5	5.0
MIN	0,138341	0,117251	0,098057	0,046338	0,034257
MAX	0,63151	0,637687	0,634567	0,619031	0,59981
AVERAGE	0,416709	0,40099	0,37642	0,351118	0,301781
	MV				
Level	3.0	3.5	4.0	4.5	5.0
MIN	137,3061	116,933	98,82618	86,23649	76,65392
MAX	195,4895	188,4743	181,6946	175,192	165,912
AVERAGE	184,0343	174,7313	163,899	151,1306	138,2066
	STD				
Level	3.0	3.5	4.0	4.5	5.0
MIN	80,66917	91,22746	92,03232	90,85099	110,0861
MAX	1670,4	1840,856	1680,723	1575,155	1520,272
AVERAGE	366,52	419,9366	494,5804	576,0743	638,7289
	BRISQUE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	17,02536	16,04095	14,40754	15,52033	12,72685
MAX	52,34262	54,92836	50,32158	54,84874	47,87191
AVERAGE	33,58486	30,43935	27,54319	27,37307	30,54417
	NIQE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	2,786929	3,024351	3,213892	3,063518	3,389537
MAX	6,869445	8,670685	9,212725	9,863945	12,66689
AVERAGE	4,100519	4,224289	4,614678	5,138227	5,806921

Table 2.13: Gamma correction results with gamma=0.1 on test set

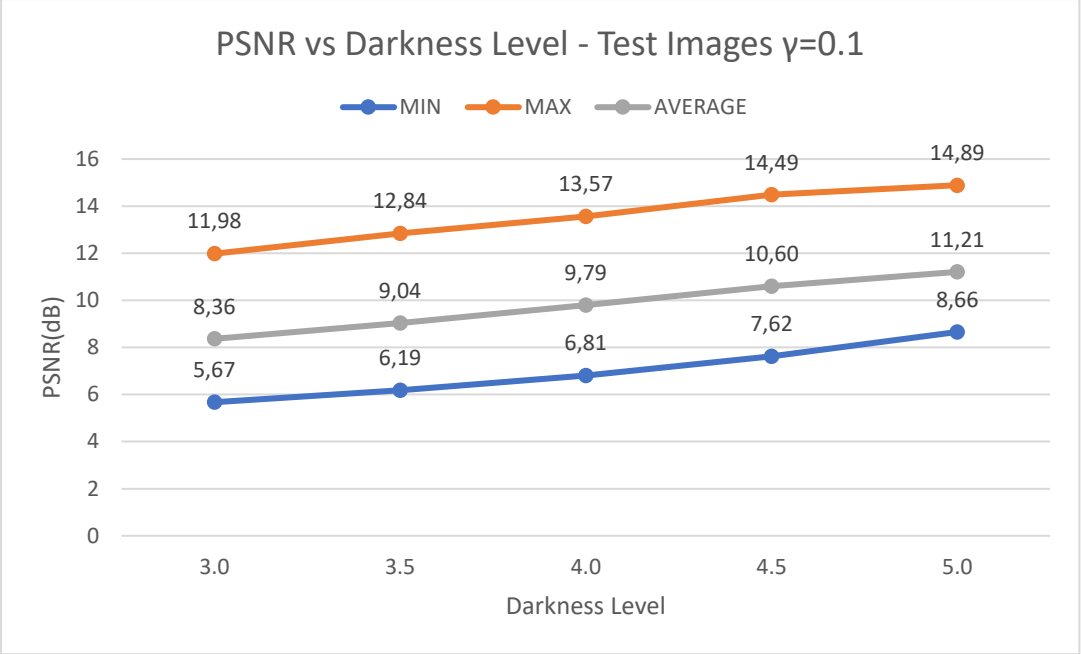


Figure 2.66: Experimental results PSNR vs Darkness Level for test images

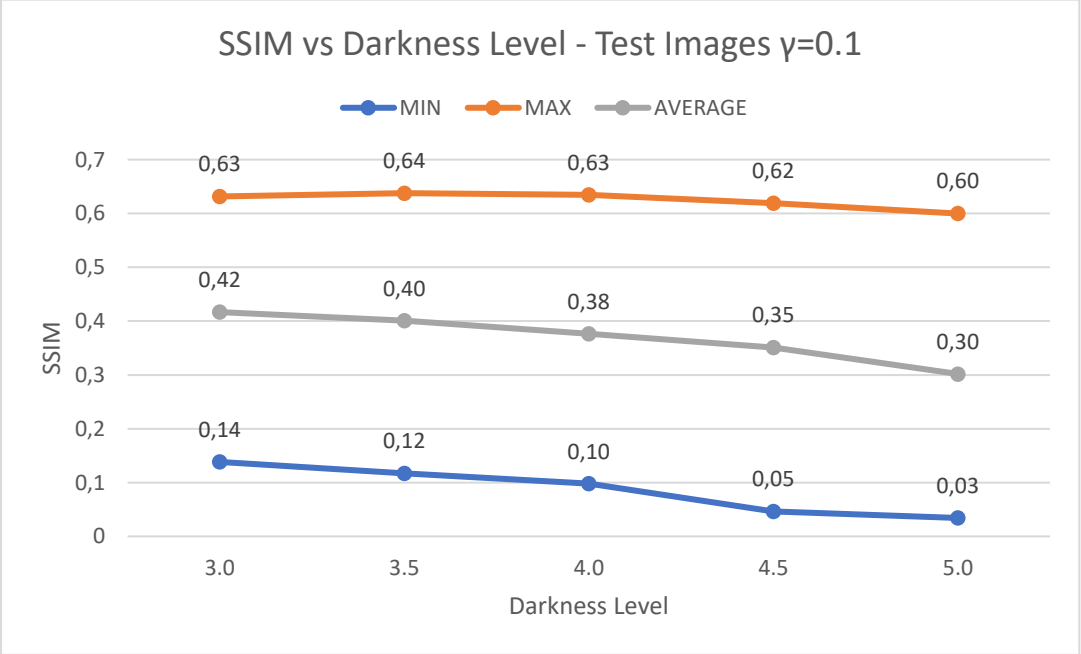


Figure 2.67: Experimental results SSIM vs Darkness Level for test images

Gamma: 0.3

MSE					
Level	3.0	3.5	4.0	4.5	5.0
MIN	566,9636	381,9993	293,9843	308,2724	444,748
MAX	2912,935	3984,445	5393,18	6899,346	8480,456
AVERAGE	1112,552	1186,356	1511,129	2062,235	2753,428
	PSNR(dB)				
Level	3.0	3.5	4.0	4.5	5.0
MIN	13,4875	12,12712	10,81235	9,742725	8,846612
MAX	20,59525	22,31018	23,44756	23,24146	21,64966
AVERAGE	17,97308	17,85597	16,96888	15,65196	14,34958
	SSIM				
Level	3.0	3.5	4.0	4.5	5.0
MIN	0,244846	0,241308	0,218396	0,136132	0,101705
MAX	0,814859	0,792386	0,751306	0,730607	0,639024
AVERAGE	0,622504	0,581671	0,524011	0,458485	0,371508
	MV				
Level	3.0	3.5	4.0	4.5	5.0
MIN	60,5788	49,18751	39,77025	33,23326	28,35013
MAX	119,0886	105,7416	94,83806	83,94127	74,21792
AVERAGE	102,7424	91,26644	80,18108	69,3653	59,60535
	STD				
Level	3.0	3.5	4.0	4.5	5.0
MIN	55,19374	55,14246	68,3997	81,92215	74,94261
MAX	971,9228	893,3015	739,4919	619,7472	533,662
AVERAGE	297,6778	269,6679	248,3169	232,4155	213,8727
	BRISQUE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	19,35772	12,73897	13,81803	16,29003	14,95675
MAX	53,45456	53,69587	50,2357	52,12136	53,59033
AVERAGE	33,82035	31,89493	30,05075	30,16164	31,54129
	NIQE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	2,610174	2,800546	2,839478	3,045056	3,27112
MAX	6,322647	6,274885	6,635347	7,212518	10,00072
AVERAGE	3,756558	3,820402	3,979833	4,291749	4,641307

Table 2.14: Gamma correction results with gamma=0.3 on test set

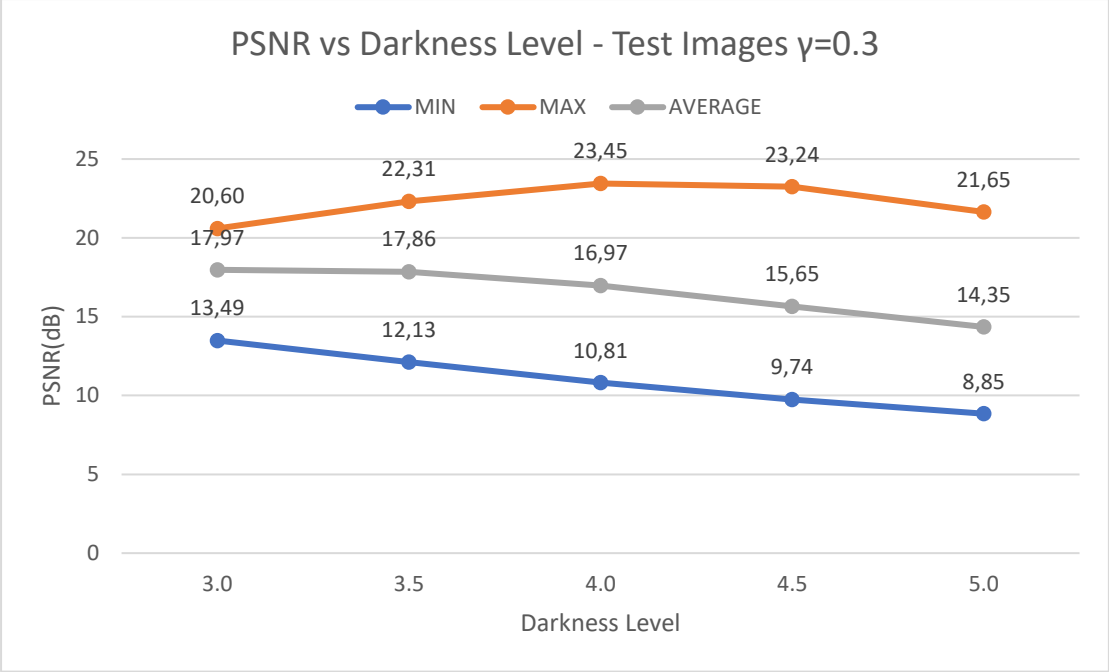


Figure 2.68: Experimental results PSNR vs Darkness Level for test images

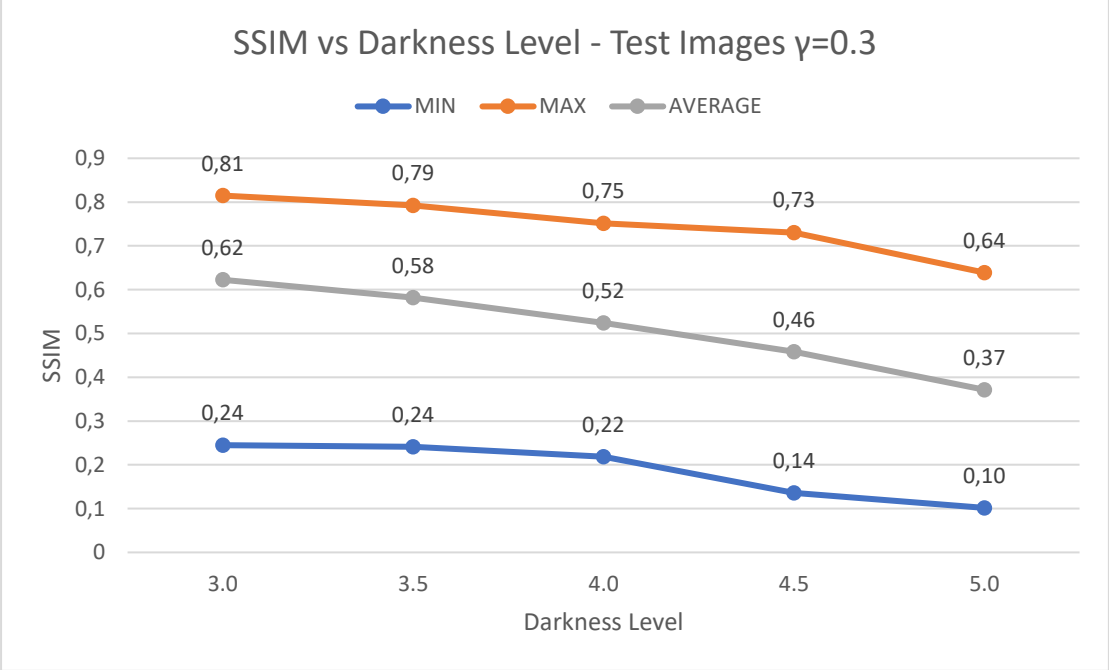


Figure 2.69: Experimental results SSIM vs Darkness Level for test images

Gamma: 0.8

MSE					
Level	3.0	3.5	4.0	4.5	5.0
MIN	1313,444	1567,784	1780,979	1941,367	2034,901
MAX	14408,37	16430,43	18199,92	19591,55	20641,49
AVERAGE	6218,285	7477,161	8554,739	9431,377	9987,48
	PSNR(dB)				
Level	3.0	3.5	4.0	4.5	5.0
MIN	6,544655	5,974314	5,530109	5,210115	4,983394
MAX	16,94669	16,17794	15,62421	15,24973	15,04537
AVERAGE	10,60835	9,753801	9,138732	8,700986	8,477435
	SSIM				
Level	3.0	3.5	4.0	4.5	5.0
MIN	0,125241	0,096621	0,071167	0,046685	0,033329
MAX	0,722072	0,601766	0,487604	0,369566	0,27941
AVERAGE	0,322691	0,226356	0,154698	0,105021	0,072121
	MV				
Level	3.0	3.5	4.0	4.5	5.0
MIN	9,320111	6,640524	4,723703	3,214721	2,348037
MAX	43,36353	33,60762	26,04361	19,59635	14,71899
AVERAGE	27,27331	20,51481	15,23529	11,1478	8,11725
	STD				
Level	3.0	3.5	4.0	4.5	5.0
MIN	20,4017	13,33742	8,047375	5,025998	3,72357
MAX	383,6804	289,9309	211,8665	150,1097	59,73326
AVERAGE	99,43363	67,27887	44,9797	29,93183	19,02574
	BRISQUE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	21,94246	29,96811	31,93656	28,4657	34,37305
MAX	50,11899	52,61293	50,16068	48,63274	48,54284
AVERAGE	40,10249	42,14368	43,34135	43,89441	43,82847
	NIQE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	3,404385	3,525715	3,478853	3,729426	3,793956
MAX	8,517159	7,958126	6,778532	7,287785	7,794703
AVERAGE	4,930637	5,035766	5,161385	5,220602	5,30716

Table 2.15: Gamma correction results with gamma=0.8 on test set

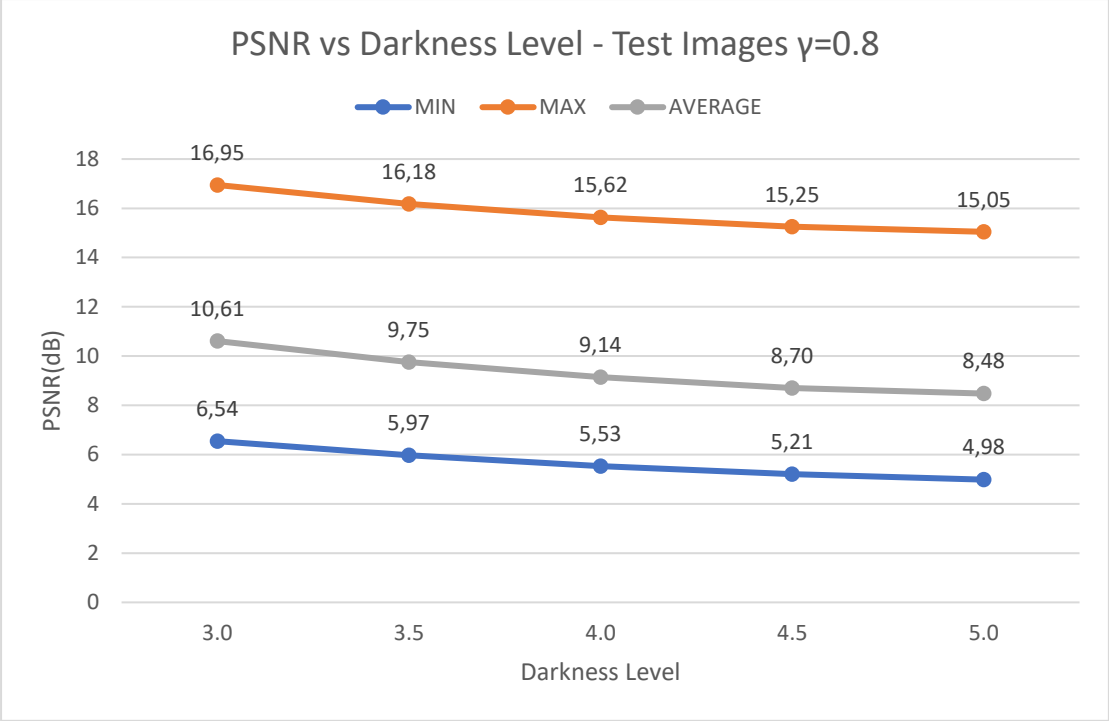


Figure 2.70: Experimental results PSNR vs Darkness Level for test images

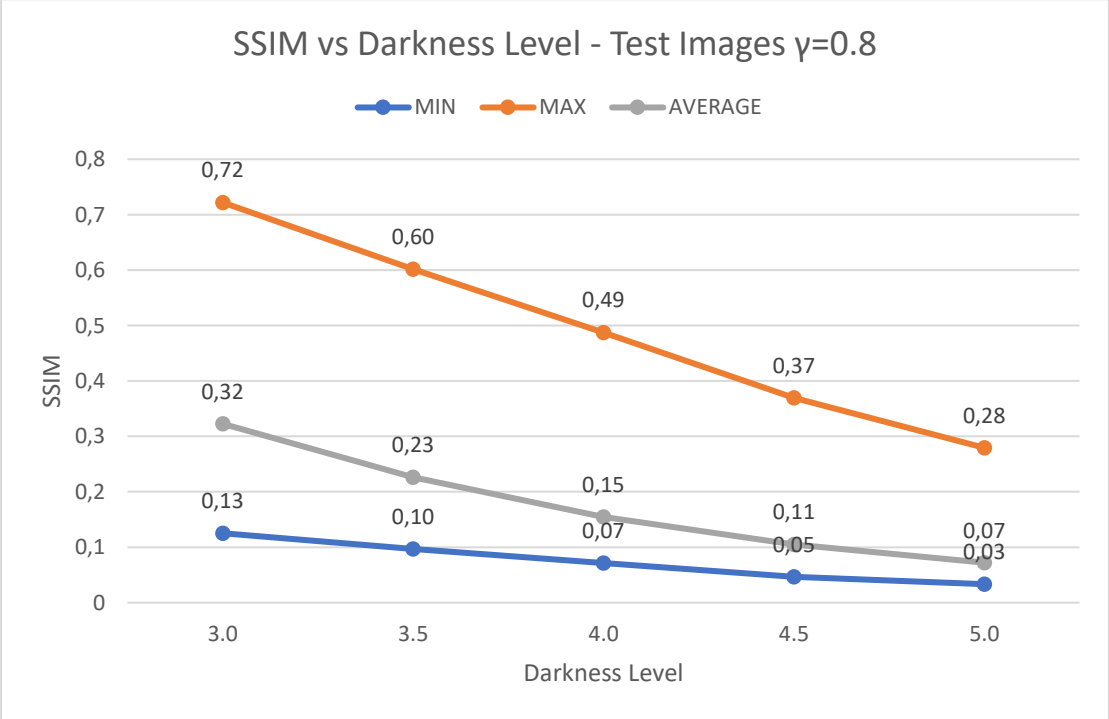
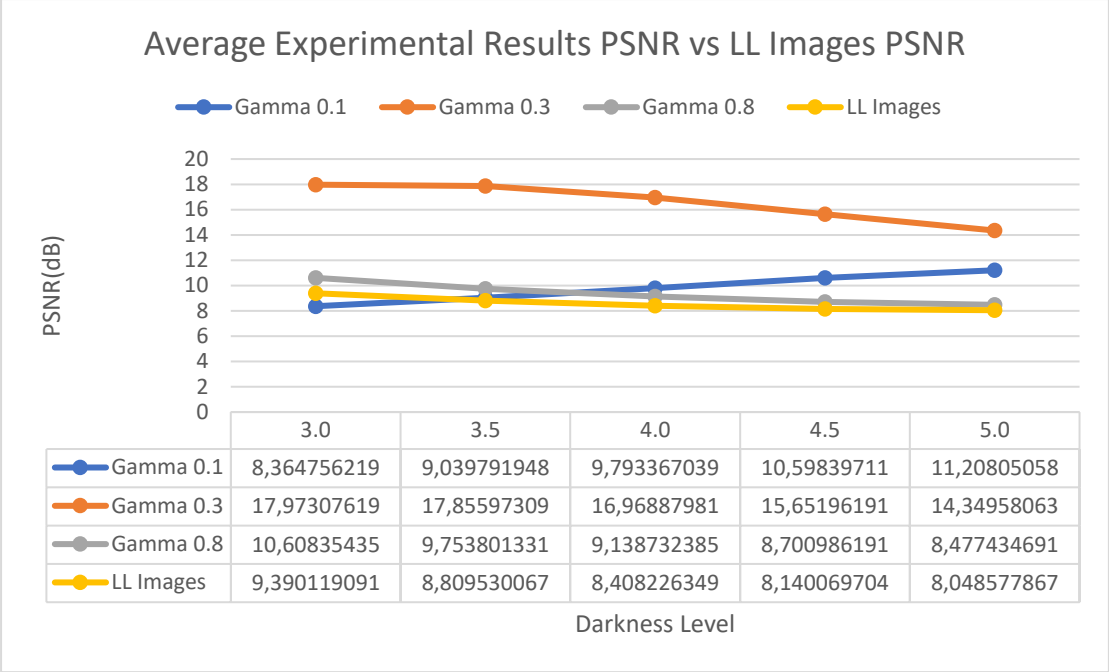
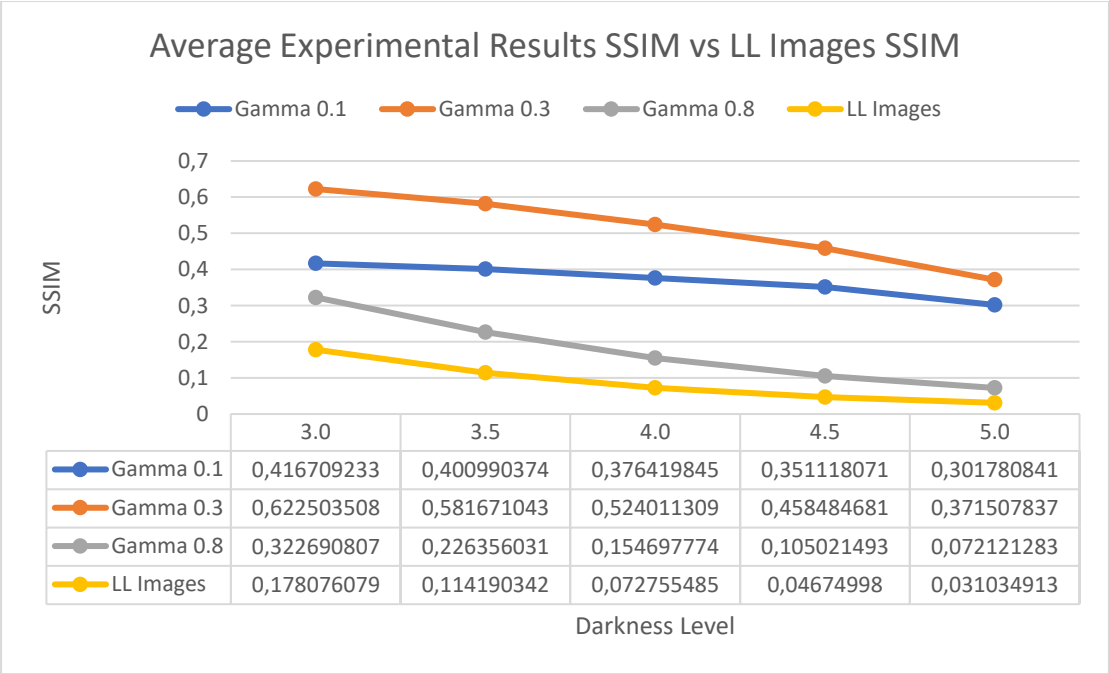


Figure 2.71: Experimental results SSIM vs Darkness Level for test images



	3.0	3.5	4.0	4.5	5.0
Gamma 0.1	8,364756219	9,039791948	9,793367039	10,59839711	11,20805058
Gamma 0.3	17,97307619	17,85597309	16,96887981	15,65196191	14,34958063
Gamma 0.8	10,60835435	9,753801331	9,138732385	8,700986191	8,477434691
LL Images	9,390119091	8,809530067	8,408226349	8,140069704	8,048577867

Figure 2.72: Experimental results average PSNR vs LL Images PSNR test set



	3.0	3.5	4.0	4.5	5.0
Gamma 0.1	0,416709233	0,400990374	0,376419845	0,351118071	0,301780841
Gamma 0.3	0,622503508	0,581671043	0,524011309	0,458484681	0,371507837
Gamma 0.8	0,322690807	0,226356031	0,154697774	0,105021493	0,072121283
LL Images	0,178076079	0,114190342	0,072755485	0,04674998	0,031034913

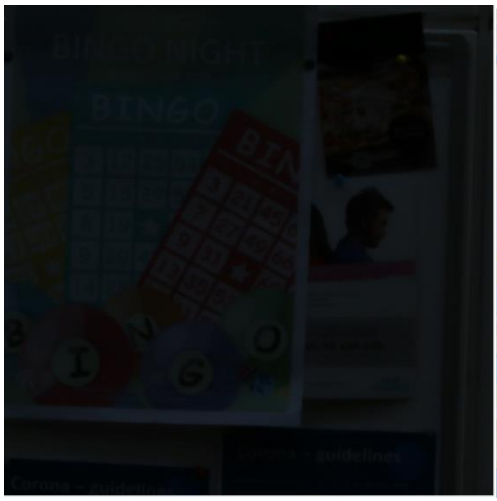
Figure 2.73: Experimental results average SSIM vs LL Images SSIM test set

And for the test set, the same conclusions as above are obtained, so we will not comment anything further.

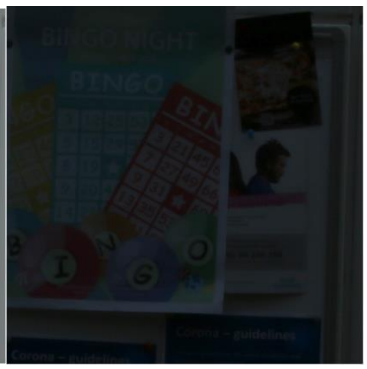
Based on the above we can reach some conclusions about the gamma transformation. First, values of the correction constant close to unity (here 0.8) improve the LL images little or not at all, which is expected due to the

mathematical function we apply. For very small values of the correction constant (here 0.1) from the evaluation metrics we came to the conclusion that there is an excessive enhancement of the dark regions, which can lead to the further distortion of the visual information, with a result that is visually far from the ground truth case. Finally, we must comment that for medium range values of the correction constant (here 0.3) the method gives the best results, significantly improving the values of the metrics, as can be seen from the corresponding diagrams. All three of these conclusions will be confirmed next, where for each darkness level we will display a random image for each correction constant value, along with the corresponding LL and Ground truth cases, as well as their respective histograms.

Darkness Level: 3.0



Original Low Light **Normal Light**



Gamma: 0.1 **Gamma: 0.3** **Gamma: 0.8**

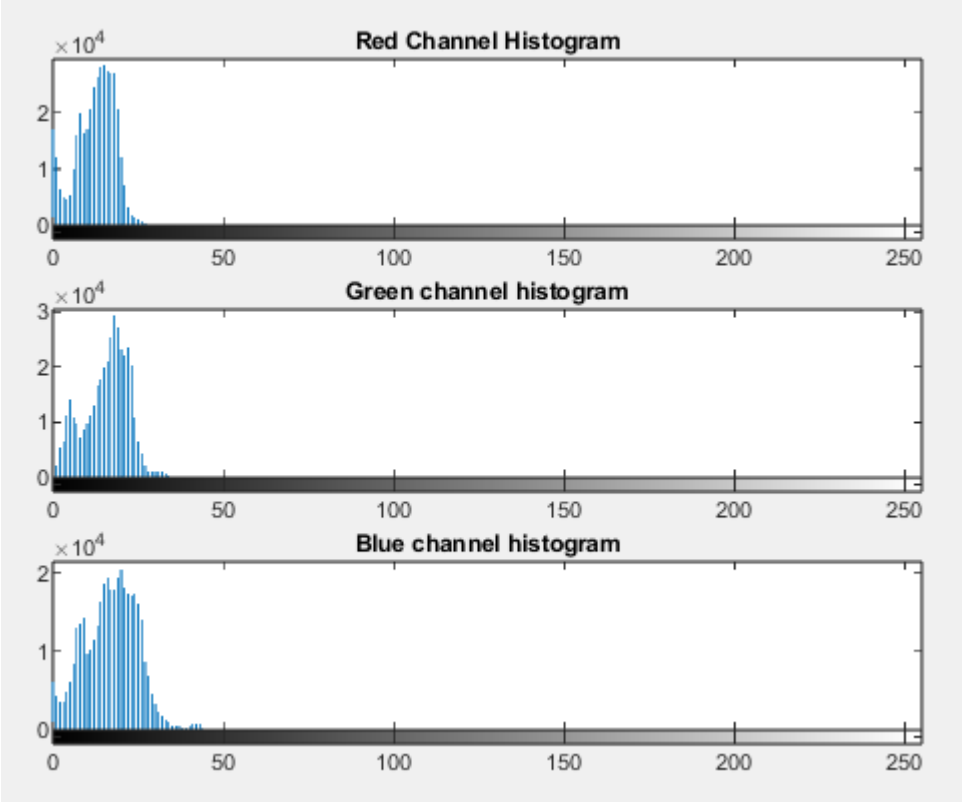


Figure 2.74: Histogram of 3.0 LL image

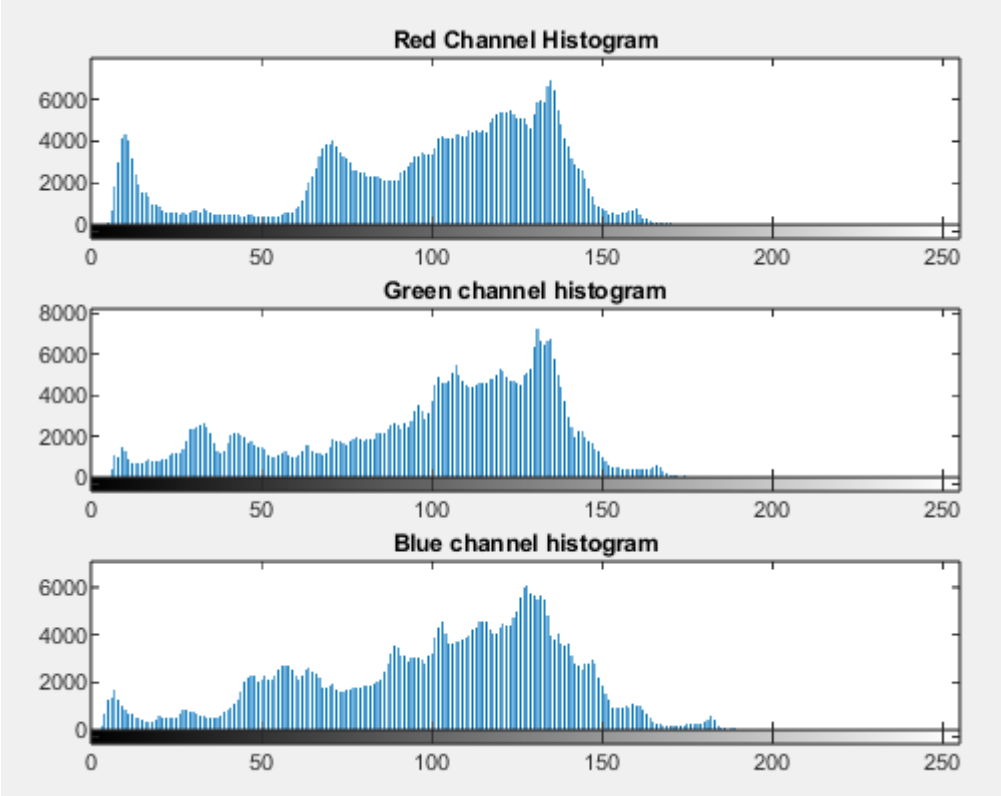


Figure 2.75: Histogram of 3.0 NL Image

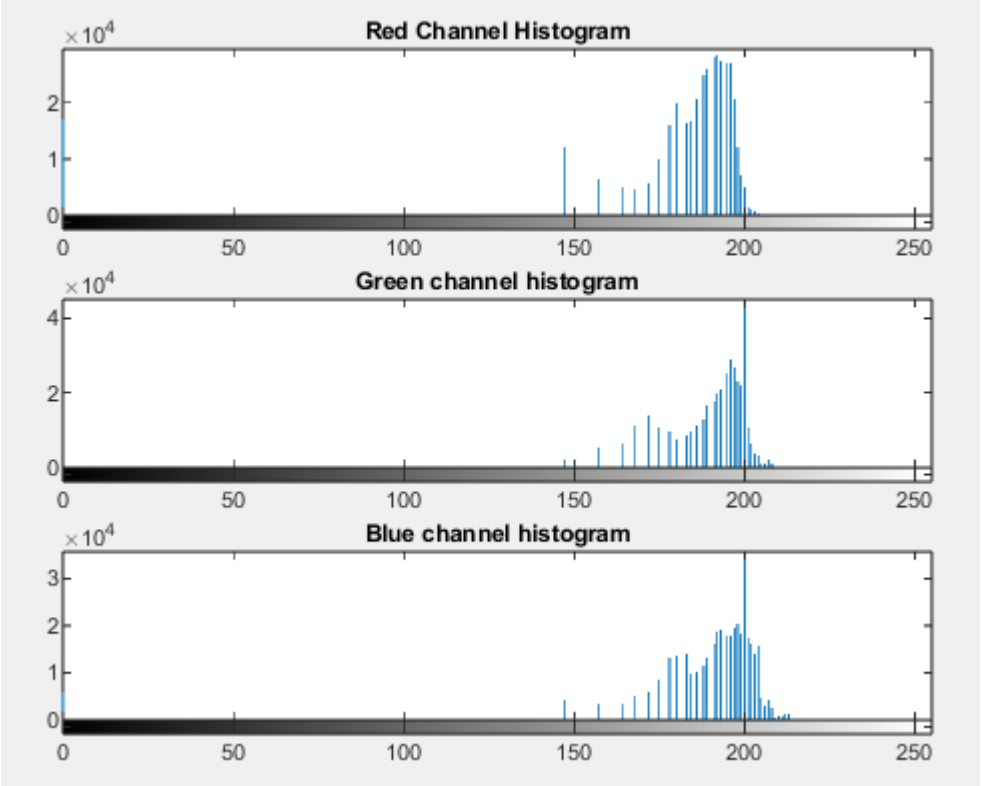


Figure 2.76: Histogram of 3.0 experimental result with gamma=0.1

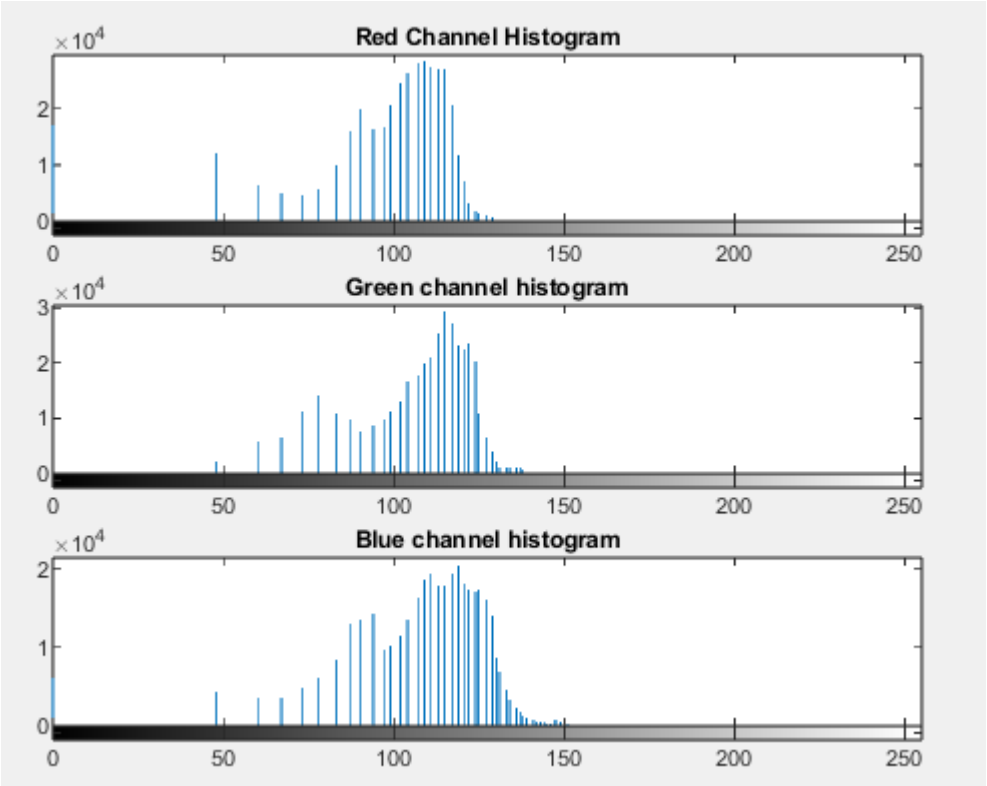


Figure 2.77: Histogram of 3.0 experimental result with gamma=0.3

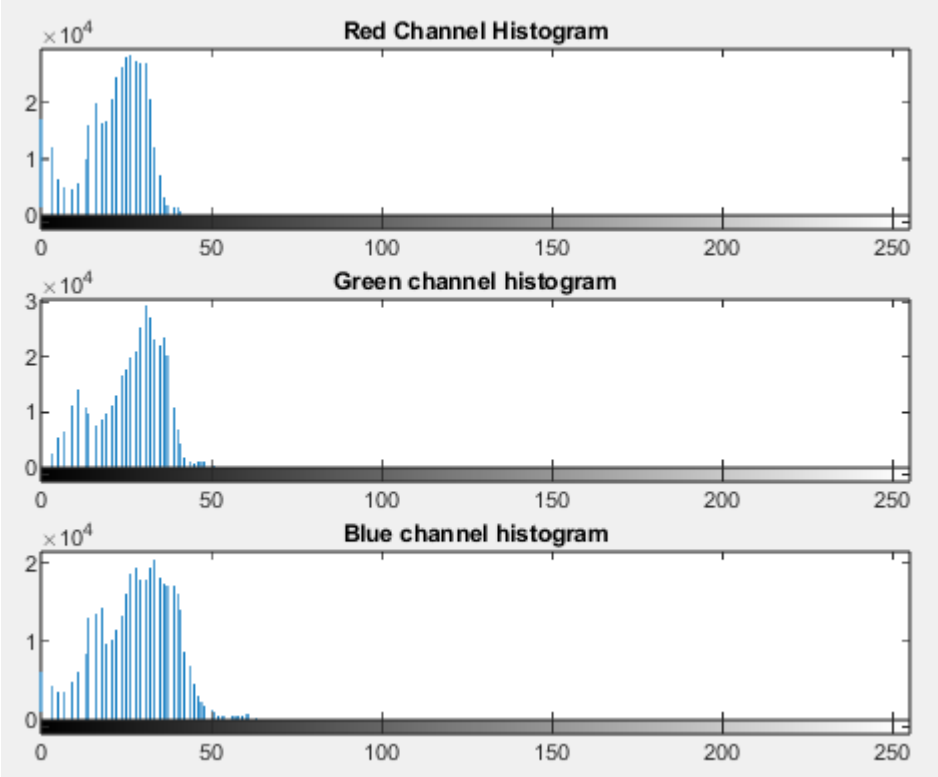
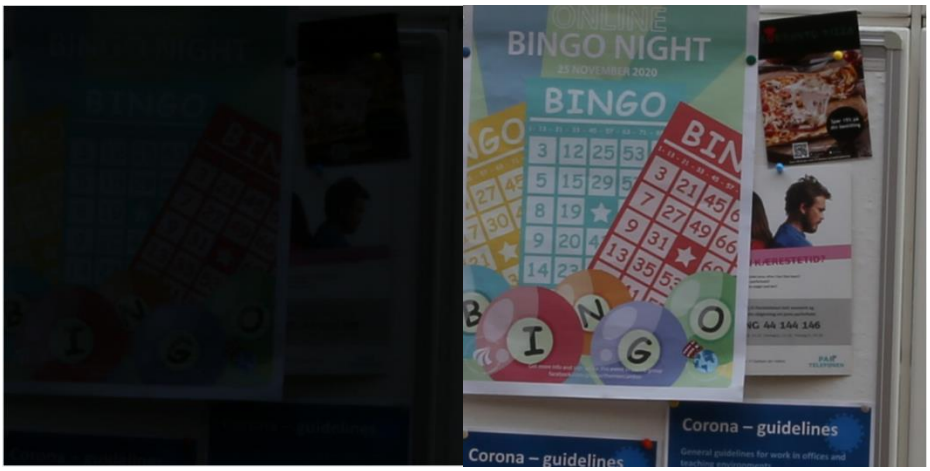


Figure 2.78: Histogram of 3.0 experimental result with gamma=0.8

We notice that the conclusions reached above are confirmed. For a correction constant of 0.8, the experimental result remains dark and is almost the same as the original LL image, which can also be seen from the corresponding histograms, in which in both cases the values accumulate in the left part of them. For a correction constant of 0.1, the excessive enhancement of the dark areas mentioned above is observed, with the experimental result characterized by strong color distortions. This can also be seen from the corresponding histogram, in which we notice that the pixel values accumulate in its right part. Finally, for a correction constant of 0.3, the best result is obtained, with the experimental image being visually very close to the ground truth. Nevertheless, some color distortions are still observed, but most of the visual information has been recovered.

Darkness Level: 3.5



Original Low Light **Normal Light**

Gamma: 0.1 **Gamma: 0.3** **Gamma: 0.8**

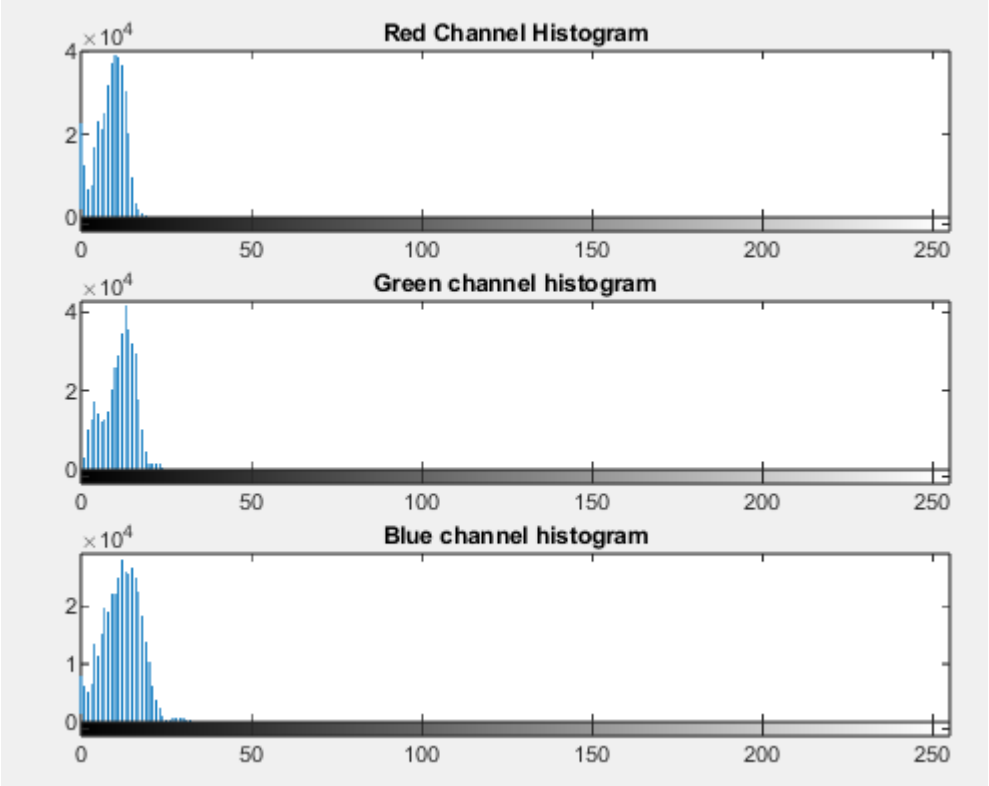


Figure 2.79: Histogram of 3.5 LL image

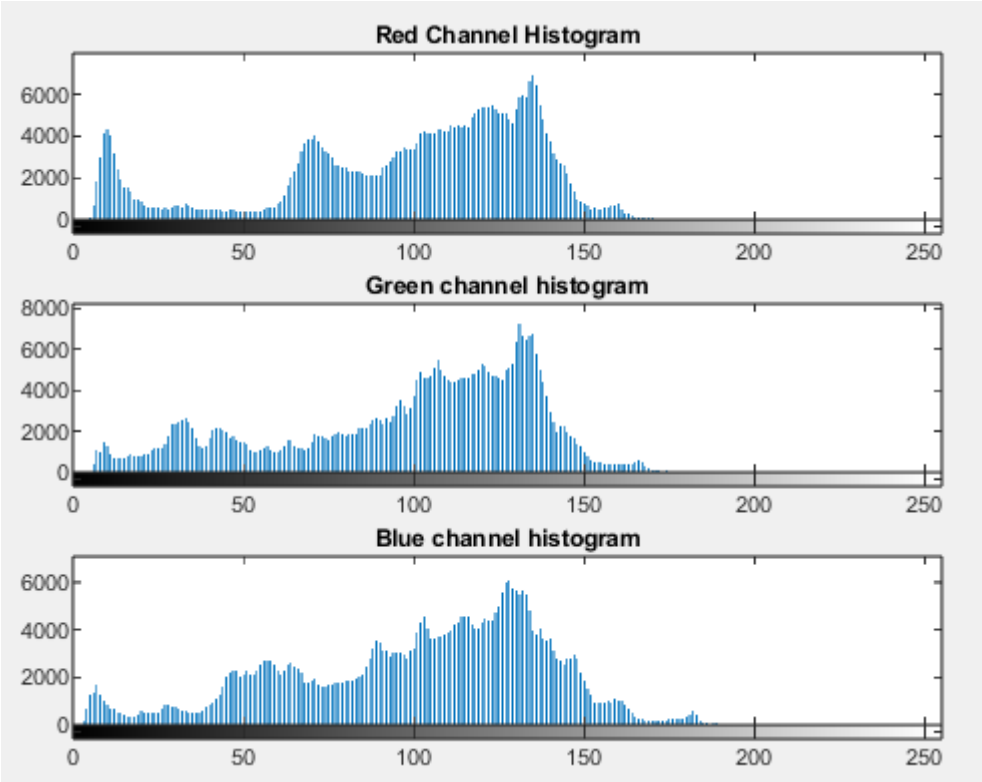


Figure 2.80: Histogram of 3.5 NL image

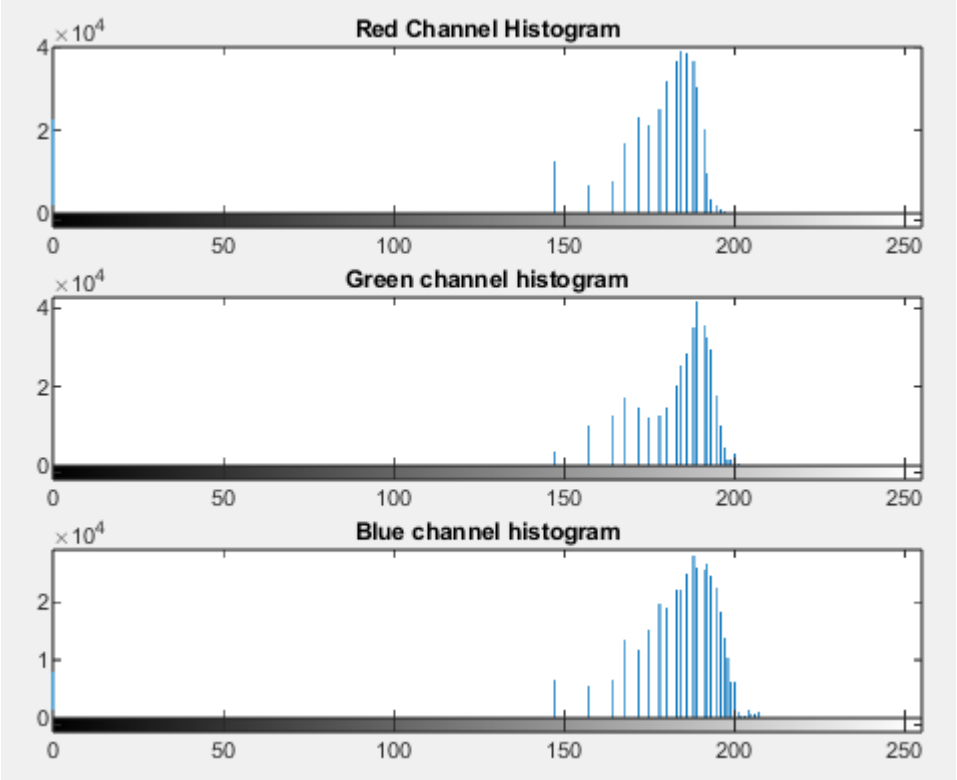


Figure 2.81: Histogram of 3.5 experimental result with gamma=0.1

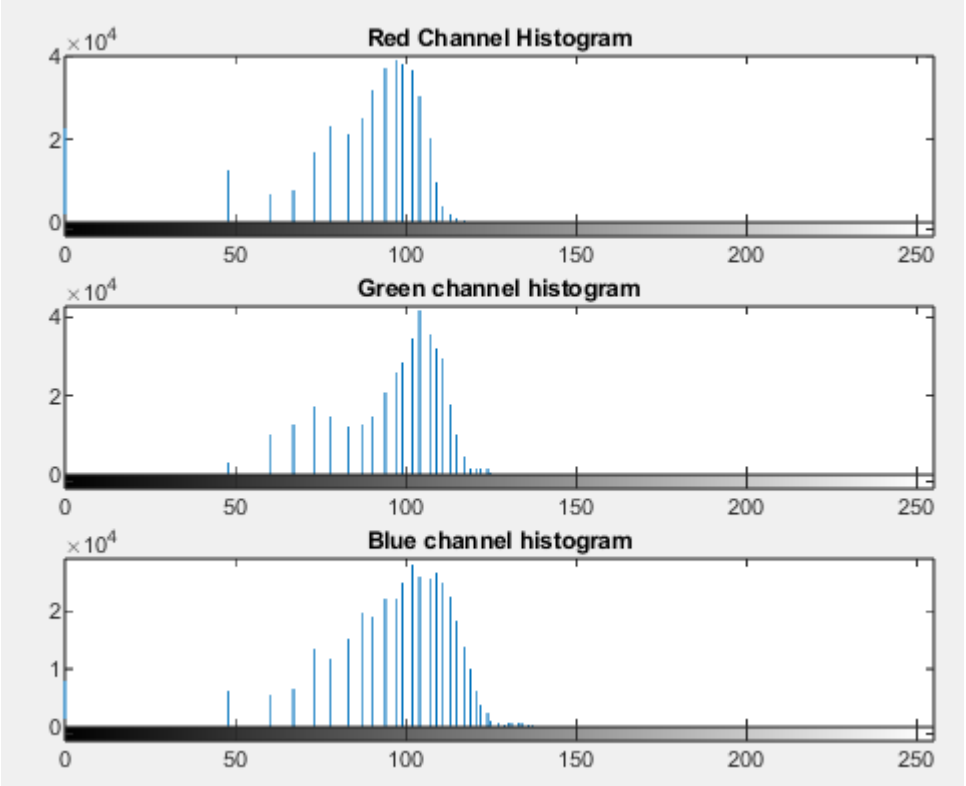


Figure 2.82: Histogram of 3.5 experimental result with gamma=0.3

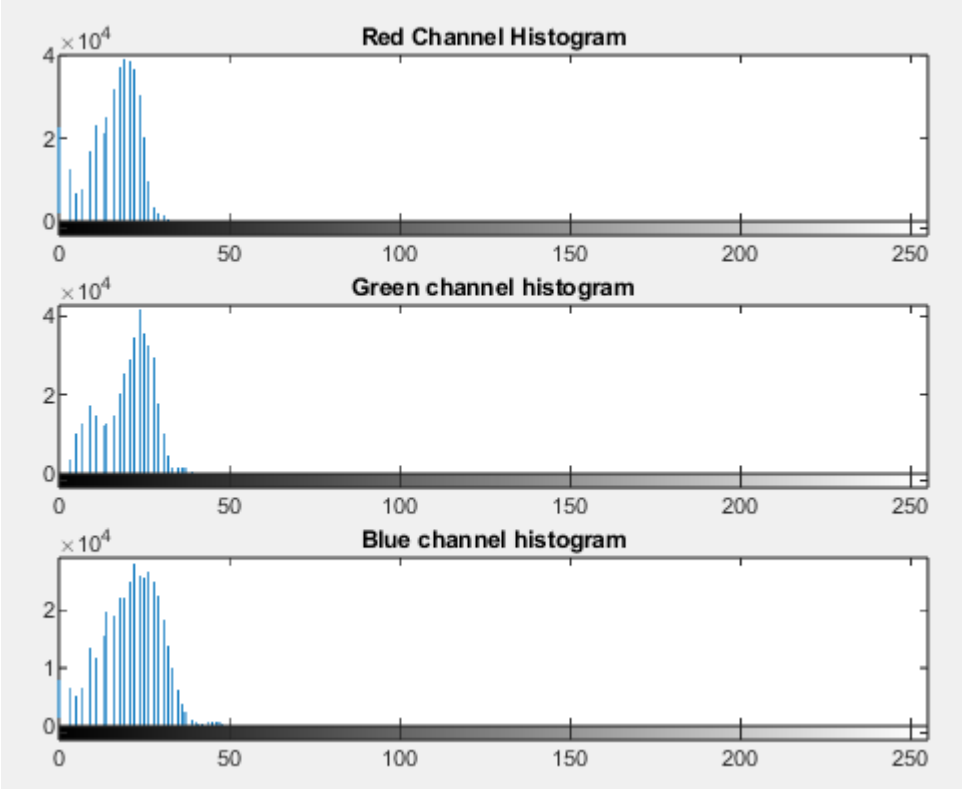
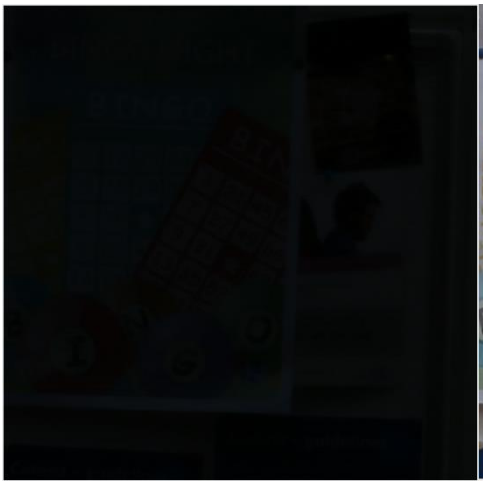


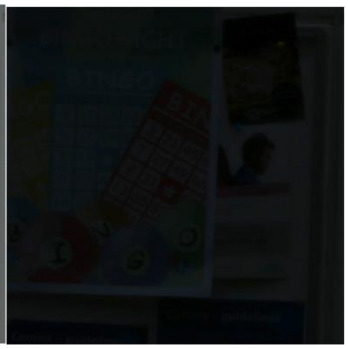
Figure 2.83: Histogram of 3.5 experimental result with gamma=0.8

For darkness level 3.5 we can also observe what we mentioned above. For a correction constant of 0.8, the experimental result remains dark and the same as the original LL image, which is also confirmed by the corresponding histograms, in which the pixel values accumulate on the left side, with very little contrast. Similarly, for a correction constant of 0.1, the dark areas are overenhanced, introducing strong color distortions, and with the pixel values in the histogram clustered to the right of it. Finally, for correction constant 0.3 again the best result is obtained, having recovered most of the color information.

Darkness Level: 4.0



Original Low Light **Normal Light**



Gamma: 0.1 **Gamma: 0.3** **Gamma: 0.8**

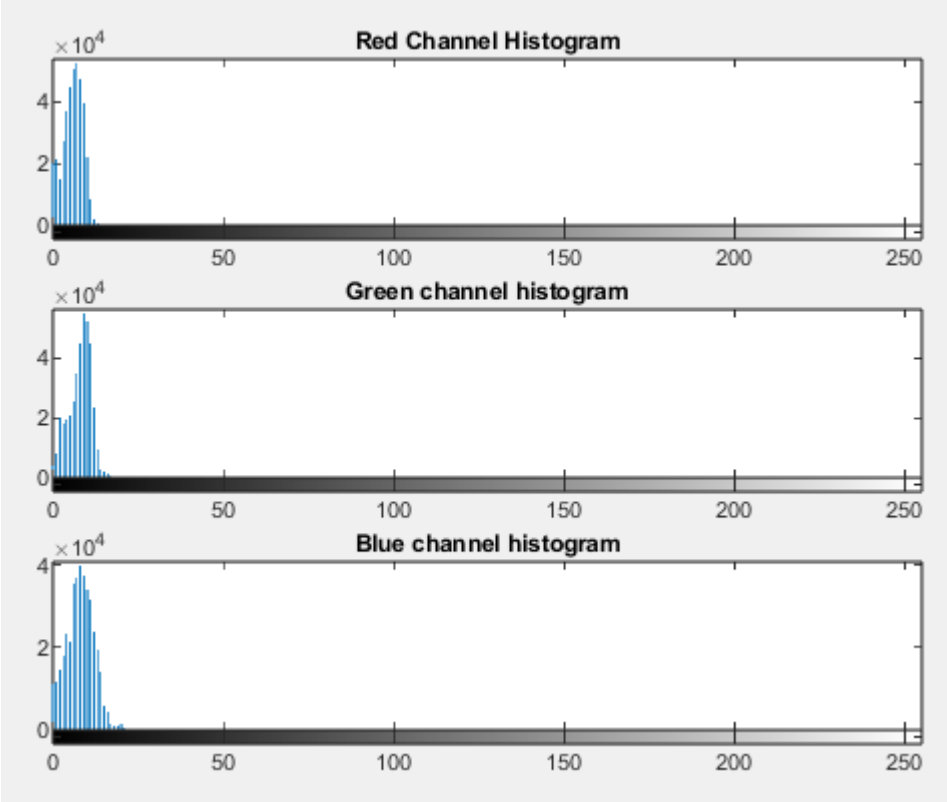


Figure 2.84: Histogram of 4.0 LL image

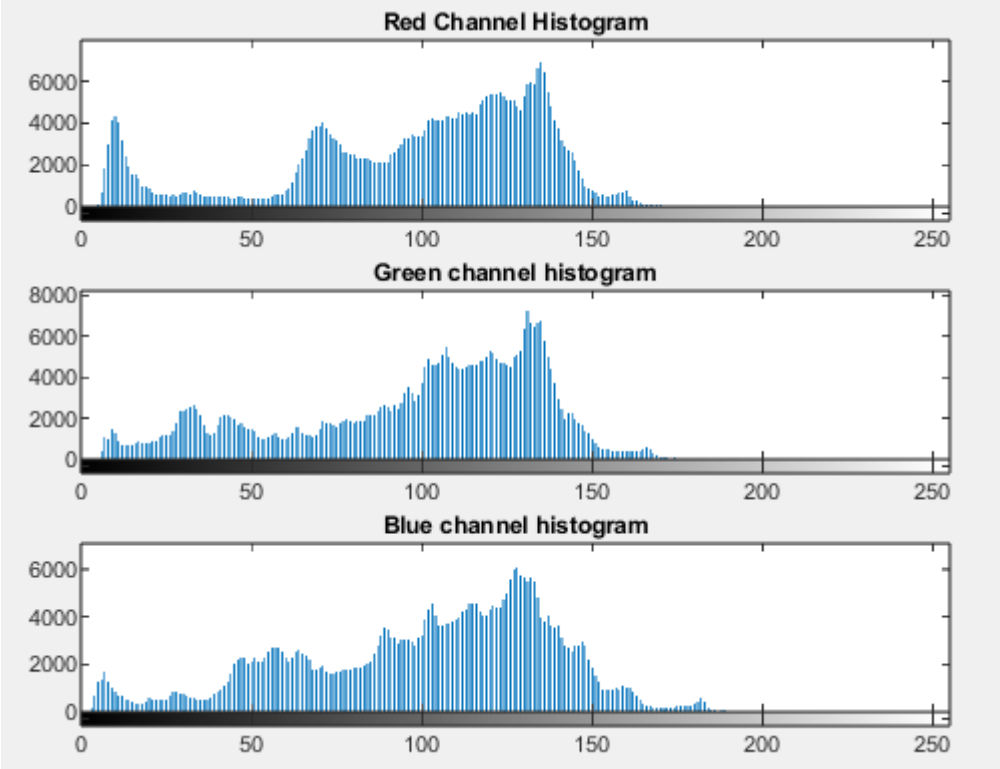


Figure 2.85: Histogram of 4.0 NL image

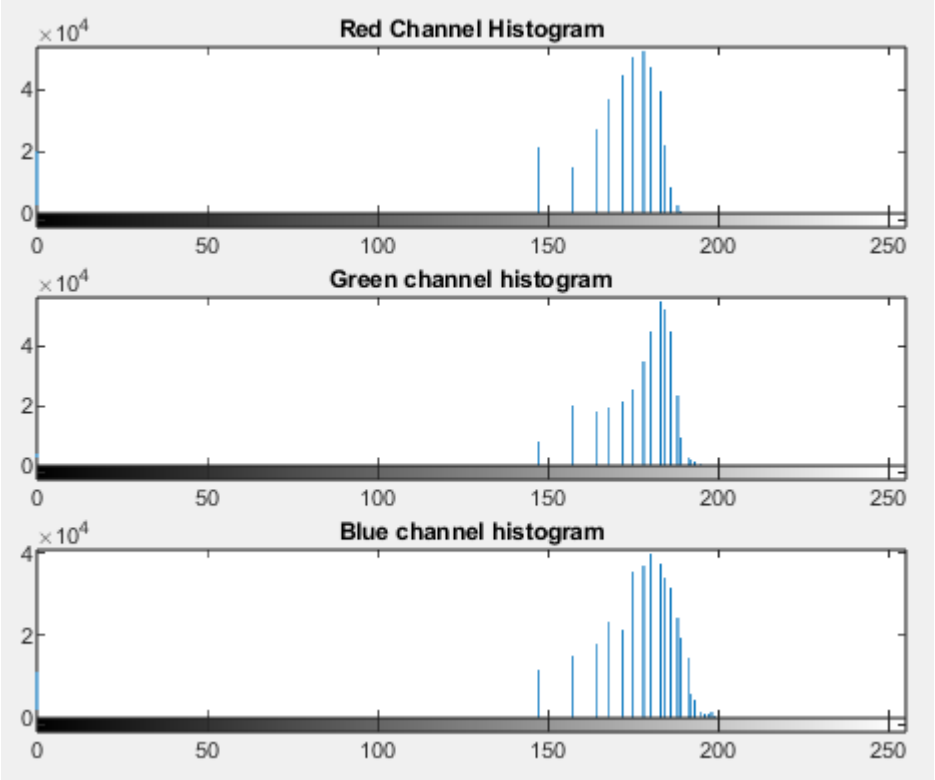


Figure 2.86: Histogram of 4.0 experimental result with gamma=0.1

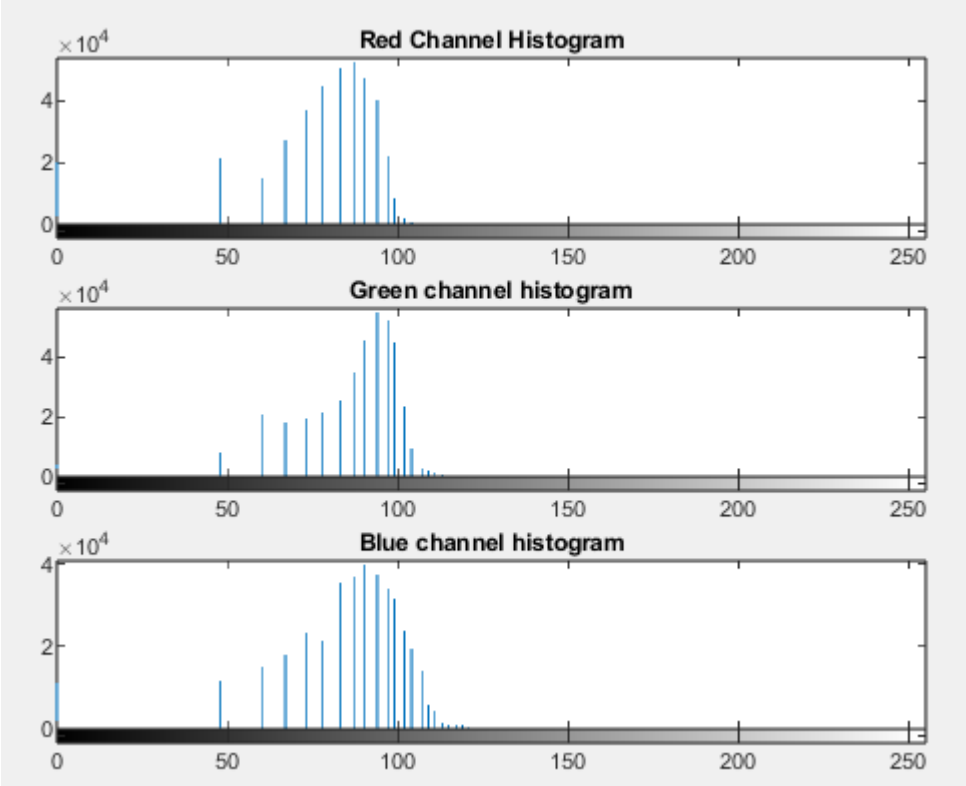


Figure 2.87: Histogram of 4.0 experimental result with gamma=0.3

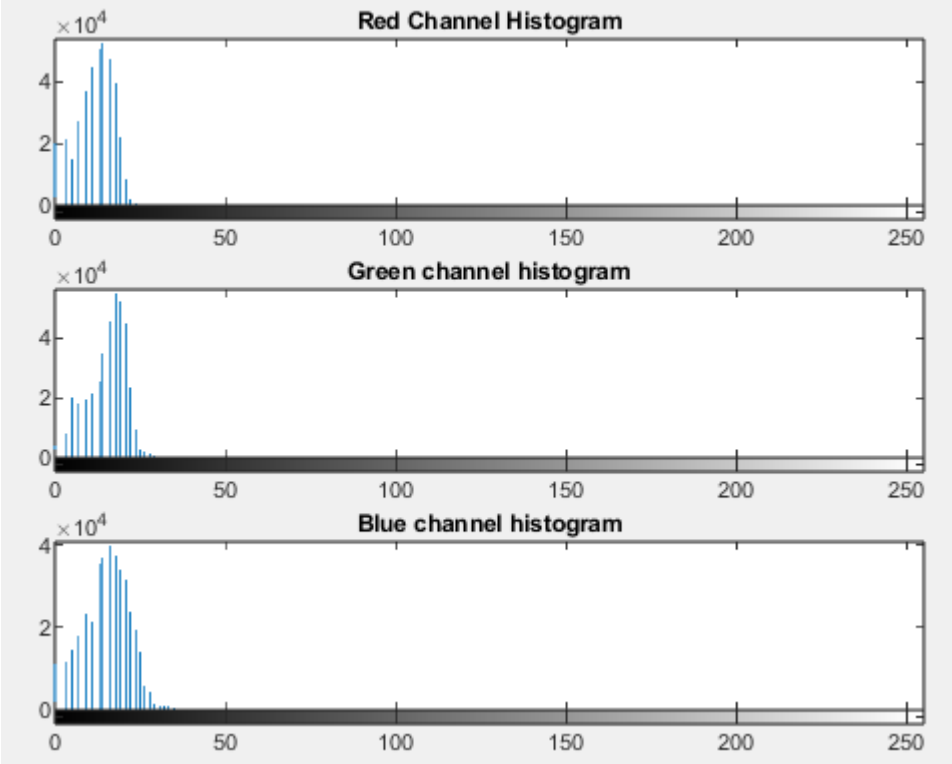
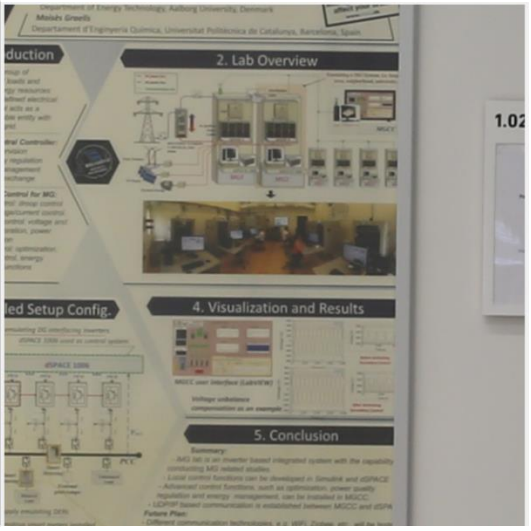


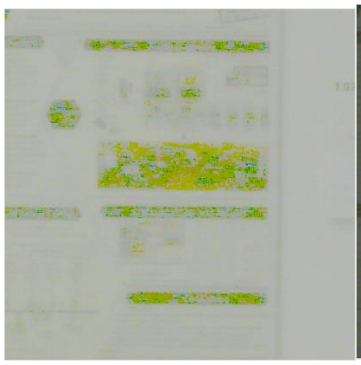
Figure 2.88: Histogram of 4.0 experimental result with gamma=0.8

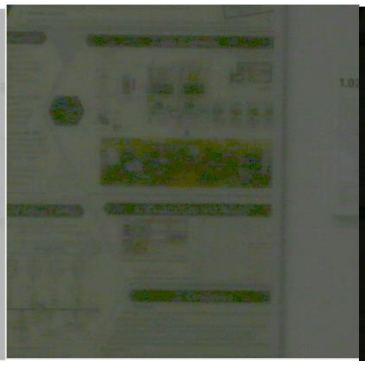
For darkness level 4.0 we can make the same observations as above. The case of a correction constant of 0.8 does not improve the result at all, with the experimental image remaining dark and the pixel values clustered in the left part of the histogram. For a correction constant of 0.1, there is an over-enhancement of dark regions, with chromatic distortions becoming more pronounced, and pixel values clustering in the right part of the experimental histogram. Finally, for a correction constant of 0.3, the best result is obtained, and we recover a large part of the visual information. Here it is worth commenting on that for γ=0.3 we have color distortions, which are more pronounced compared to darkness level 3.0 and 3.5, which is expected due to the increase in darkness level.

Darkness Level: 4.5



Original Low Light **Normal Light**





Gamma: 0.1 **Gamma: 0.3** **Gamma:0.8**

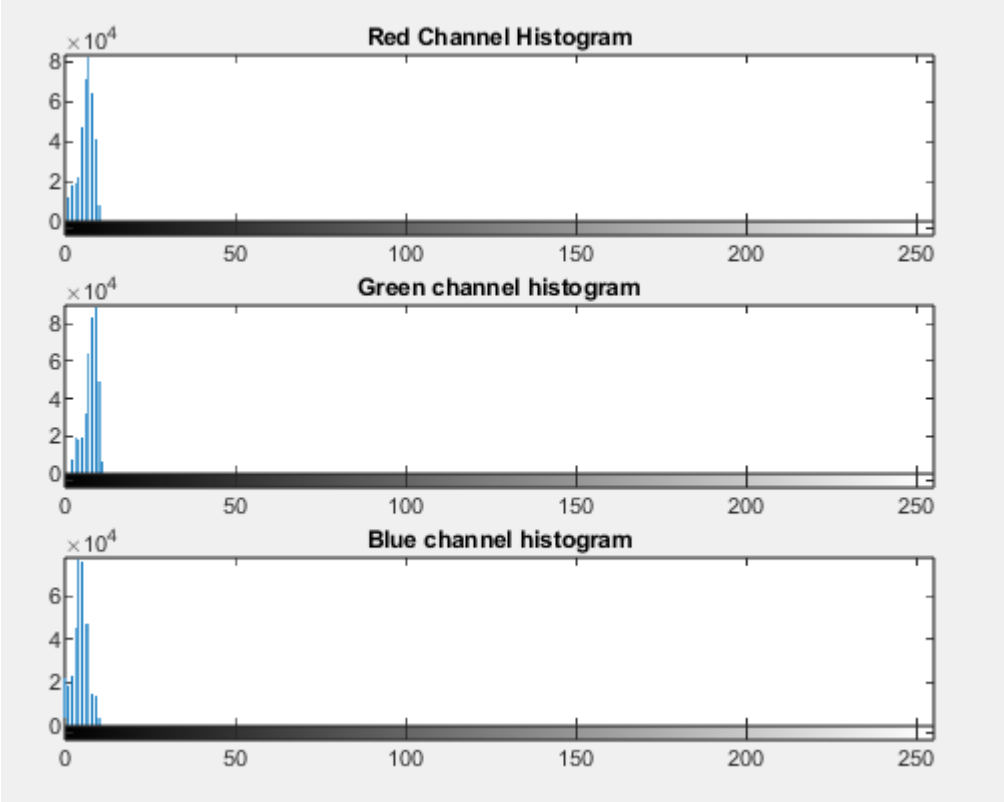


Figure 2.89: Histogram of 4.5 LL image

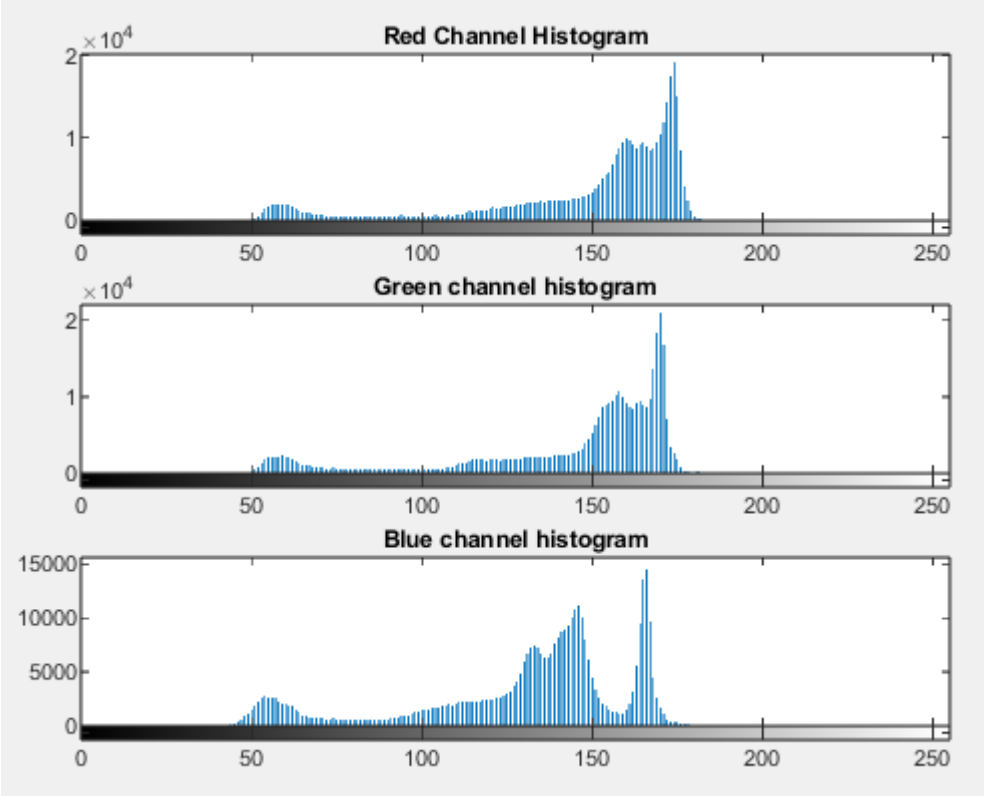


Figure 2.90: Histogram of 4.5 NL image

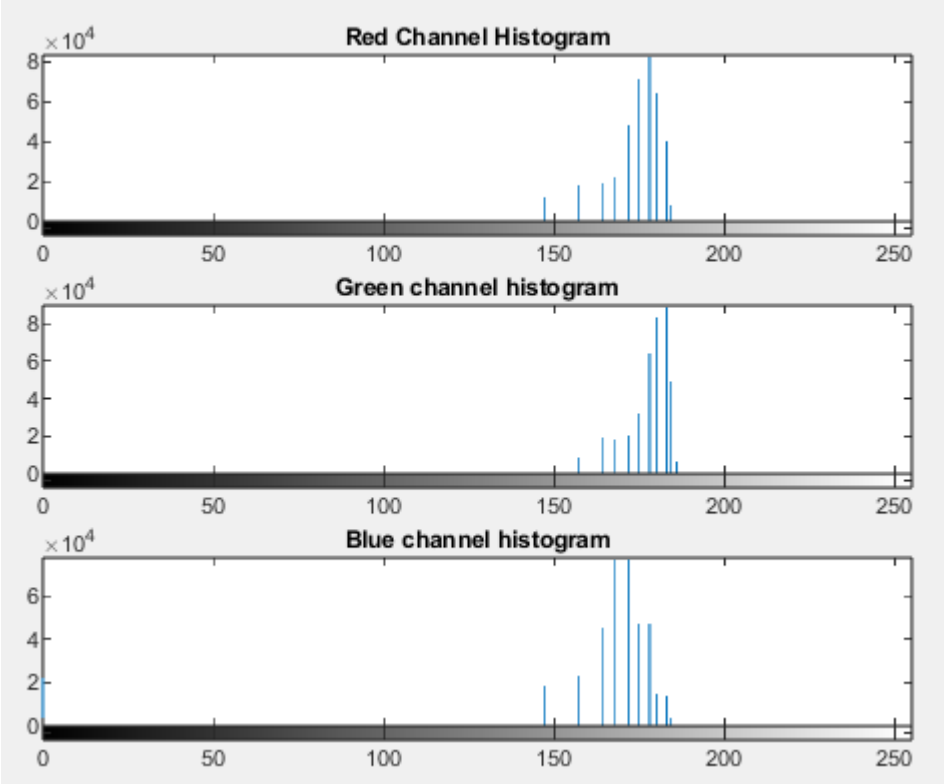


Figure 2.91: Histogram of 4.5 experimental result with gamma=0.1

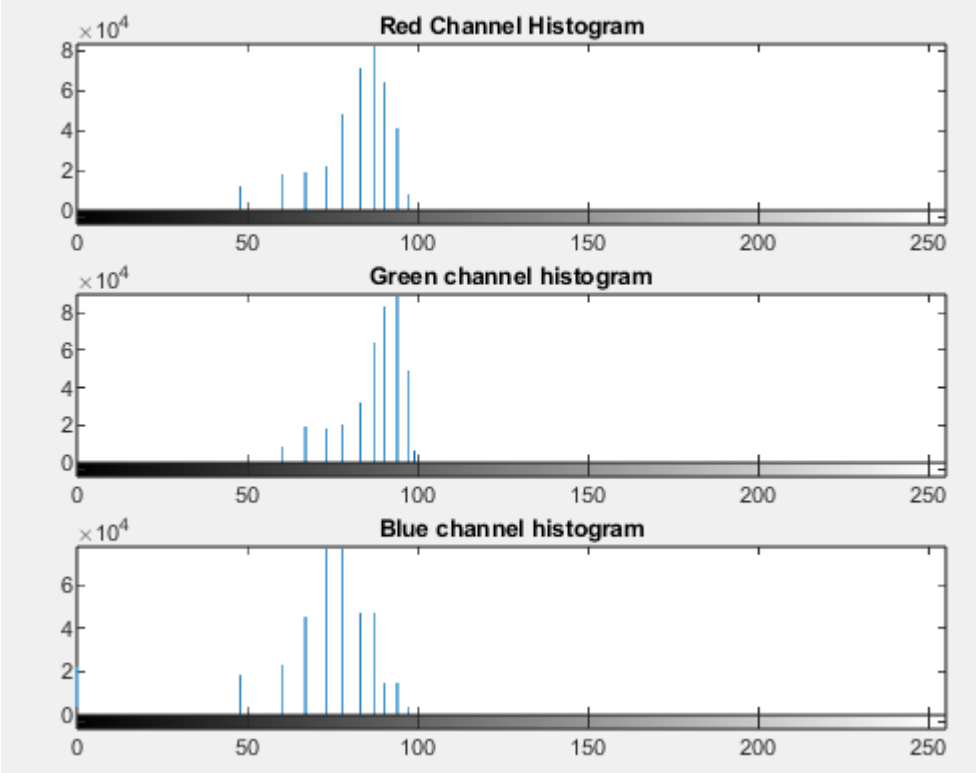


Figure 2.92: Histogram of 4.5 experimental result with gamma=0.3

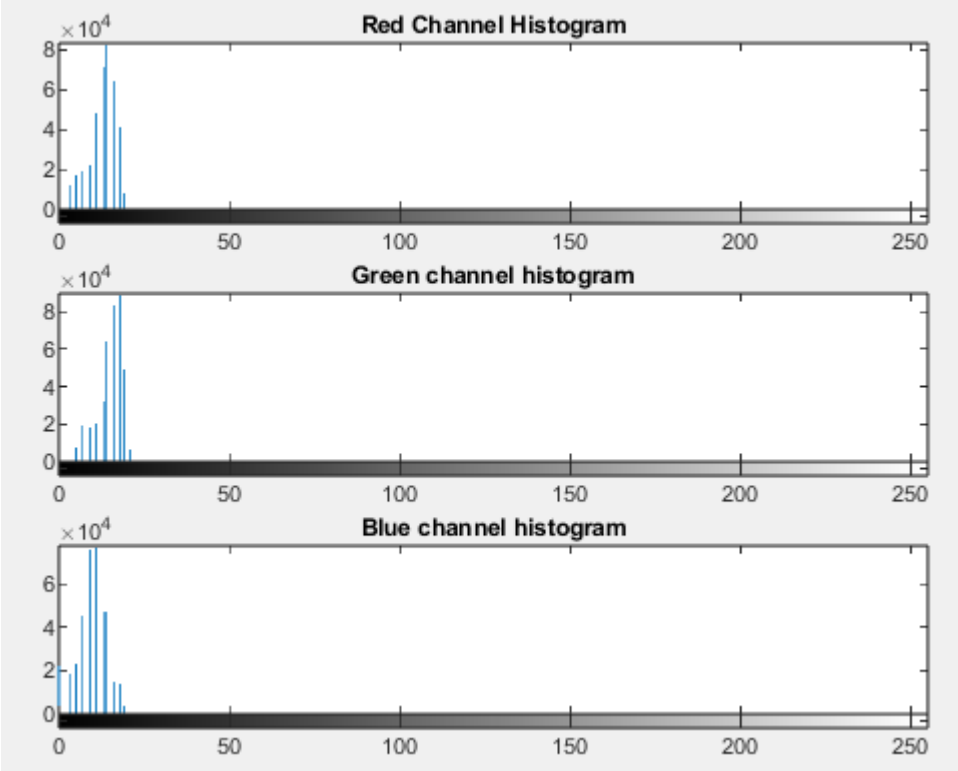
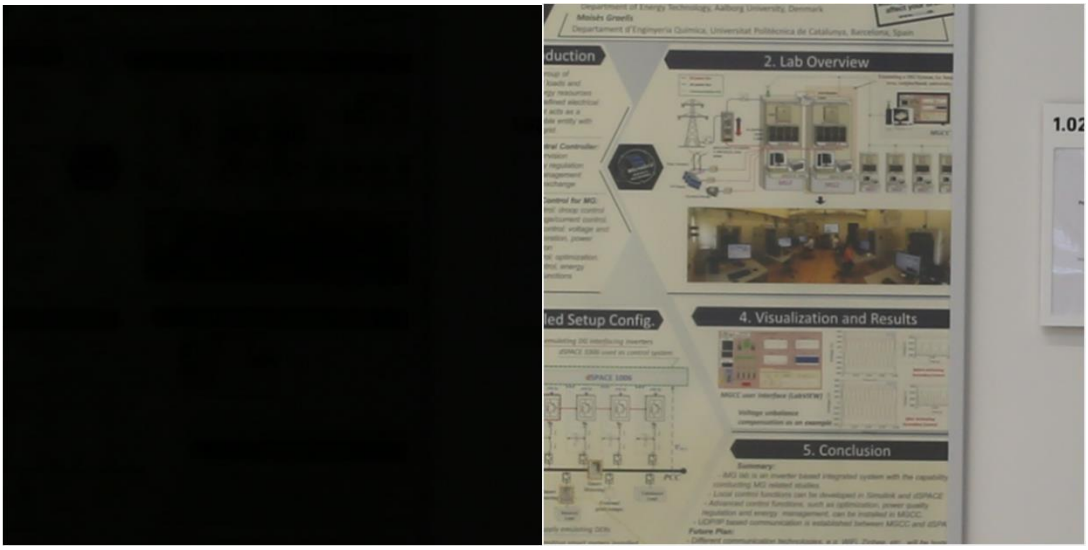


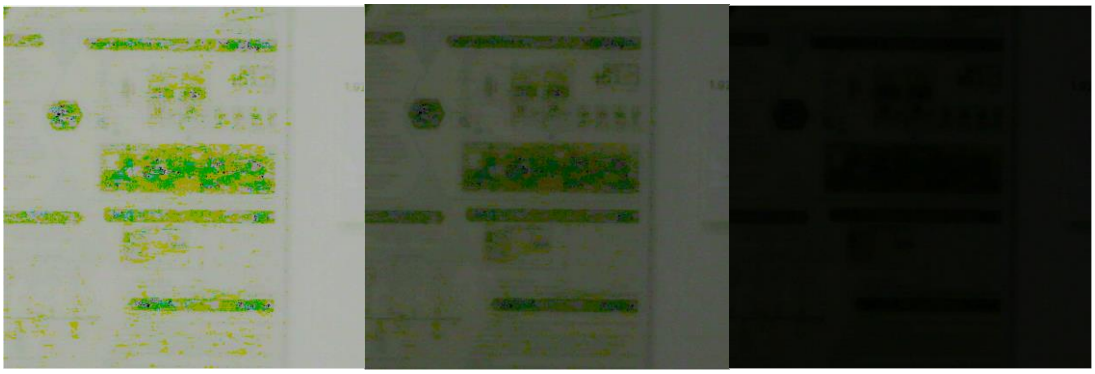
Figure 2.93: Histogram of 4.5 experimental result with gamma=0.8

For darkness level 4.5 we notice that again for a correction constant of 0.8 the LL image does not improve at all, with the pixel values being accumulated in the left part of the histogram. For a correction constant of 0.1 there is again an over-enhancement of the dark areas, which leads to the enhancement of chromatic aberrations and the introduction of more noise. Finally, for a correction constant of 0.3 the best result is obtained, compared to the other two cases. Nevertheless, the experimental result is characterized by color distortions, more intense compared to the previous darkness levels, which is due to the fact that the images are very dark and make it difficult to retrieve the correct color information.

Darkness Level: 5.0



Original Low Light **Normal Light**



Gamma: 0.1 **Gamma: 0.3** **Gamma: 0.8**

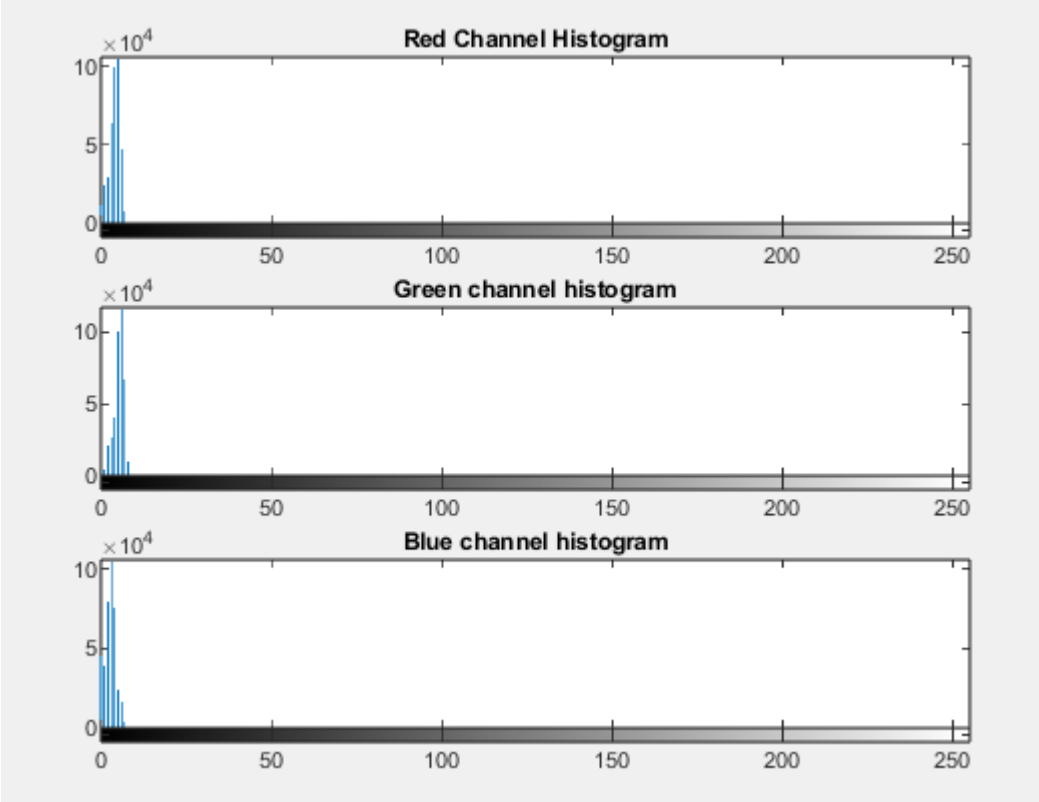


Figure 2.94: Histogram of 5.0 LL image

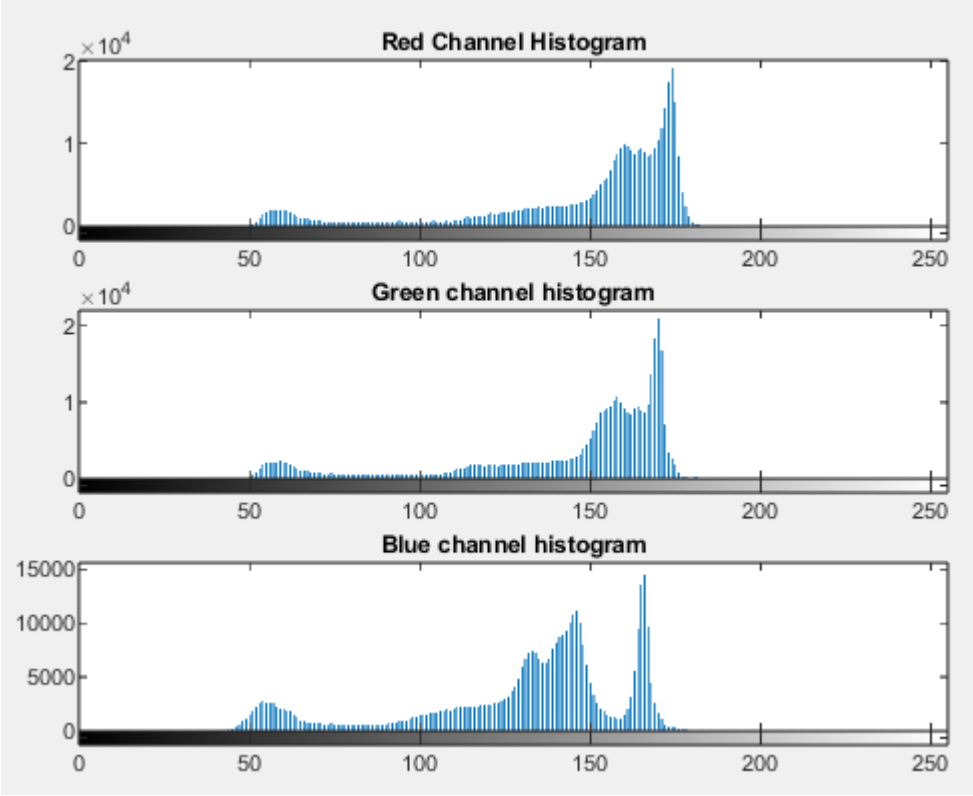


Figure 2.95: Histogram of 5.0 NL image

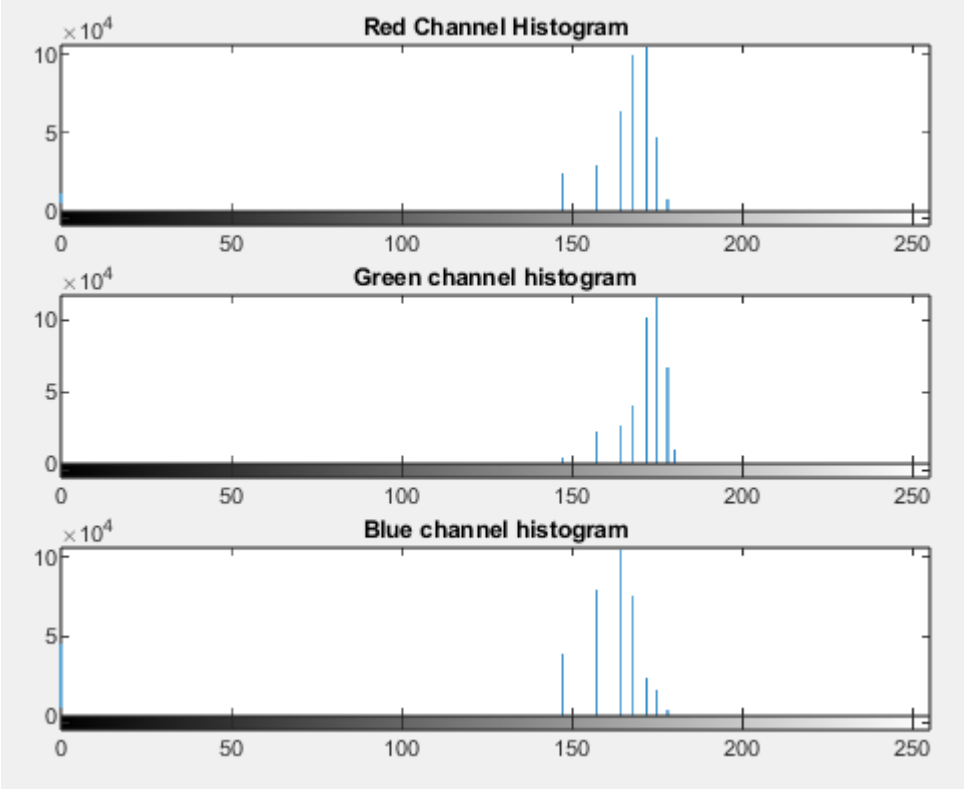


Figure 2.96: Histogram of 5.0 experimental result with gamma=0.1

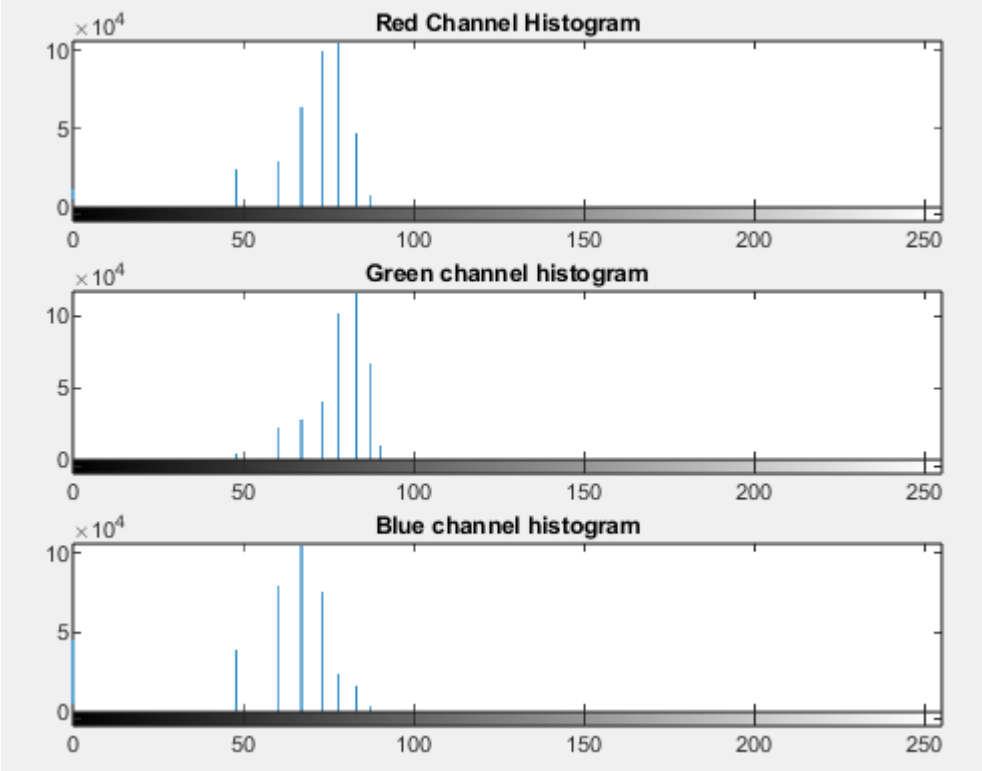


Figure 2.97: Histogram of 5.0 experimental result with gamma=0.3

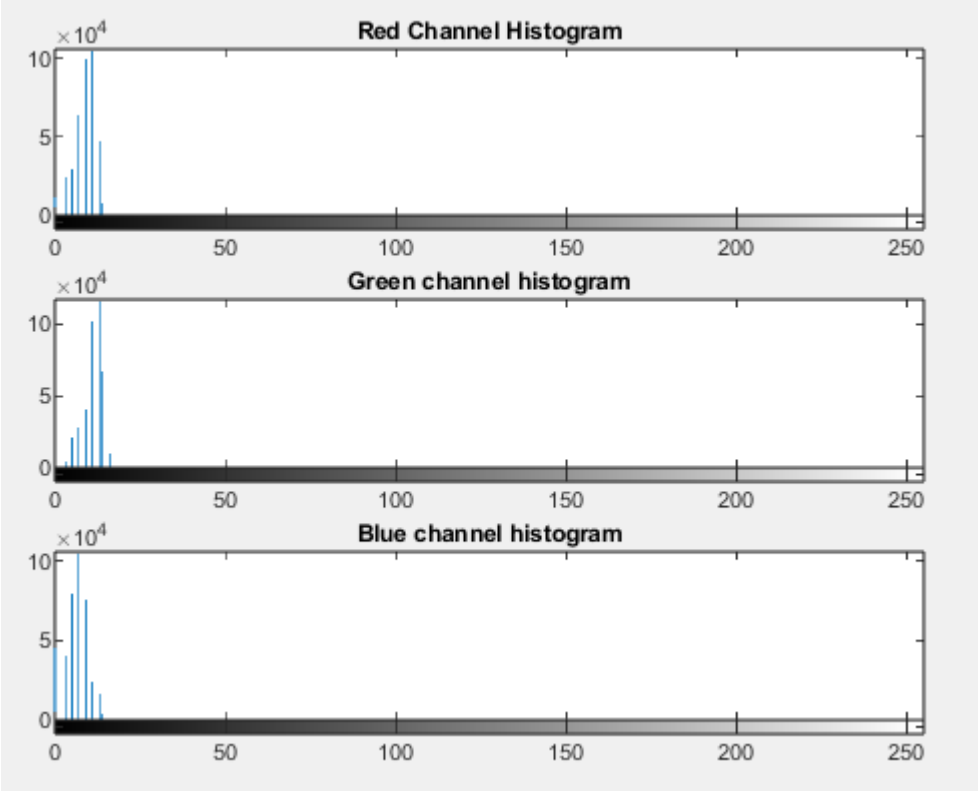


Figure 2.98: Histogram of 5.0 experimental result with gamma=0.8

For darkness level 5.0 we observe that for a correction constant of 0.8 the LL image does not improve at all, and the pixel values remain concentrated in the left part of the histogram. For a correction constant of 0.1 there is an over-enhancement of the dark areas resulting in the further introduction of chromatic aberrations, which greatly reduce the quality of the result. Finally, for a correction constant of 0.3, the best result is obtained, compared to the other 2 cases, but strong color distortions are detected. This is due to the fact that the images have become too dark, making it difficult for the method to recover all the visual information.

It is worth at this point to comment on the results as a whole. For a correction constant of 0.8 we observe that at all darkness levels, the LL images are improved little or not at all. We can see this, firstly from the values of the quality metrics, which in all cases are almost the same as the values of the original LL images, but also from the random images we displayed for each darkness level. The experimental result remains dark, and the pixel values remain accumulated in the left part of the histogram. If we look at figure 2.49, with the gamma transformation curves for each value of the correction constant, we can understand that this fact was expected. The specific value of the correction constant is very close to

unity, so after applying the gamma transform, it leaves the respective LL image almost unchanged.

For a correction constant of 0.1 we noticed at all darkness levels that the dark areas are over-enhanced, and the end result is as if the images were taken in over-exposed lighting conditions. This can first be seen from the metrics, as this is the only case so far where MV increases with increasing darkness level, which means that the average brightness of the given image also increases. In addition to the metrics, it can also be seen from the random images we displayed for each darkness level, where in each case the pixels were concentrated in the right part of the histogram. This phenomenon had the effect of amplifying the existing noise as well as increasing the color distortions, greatly reducing the quality of the final experimental result. This fact was again expected, as from figure 2.49 we can see that values of the correction constant close to 0 have the property of strongly enhancing dark areas. For the sake of completeness, however, we had to test all value cases.

In the case where we used a correction constant of 0.3, the best results were obtained for all darkness levels. Quality metrics improved noticeably, with PSNR and SSIM taking their maximum values (compared to the other 2 cases). From the random images we displayed for each darkness level, we see that the experimental results for $\gamma=0.3$ are visually closer to the ground truth case, having recovered much of the visual information. However, we must comment that for all darkness levels, color distortions appeared in the experimental results, which became more intense with the increase of the darkness level. This is justified by the fact that the images become darker and darker, and correspondingly the dynamic range becomes smaller and smaller, making it extremely difficult to fully and correctly recover the color information.

2.3 Log Correction

This is a pointwise, non-linear transformation, like the gamma transformation, except in this case we assume that between pixels of the LL image and the pixels of the NL image there is a logarithmic relationship. The mathematical function we apply to the value of each pixel has the form:

$$I(i,j) = \log_{10}\left(1 + c * I_{LL}(i,j)\right)$$

Where c is a control constant, depending on the value of which, the function can take various forms. Some examples are shown in figure 2.99.

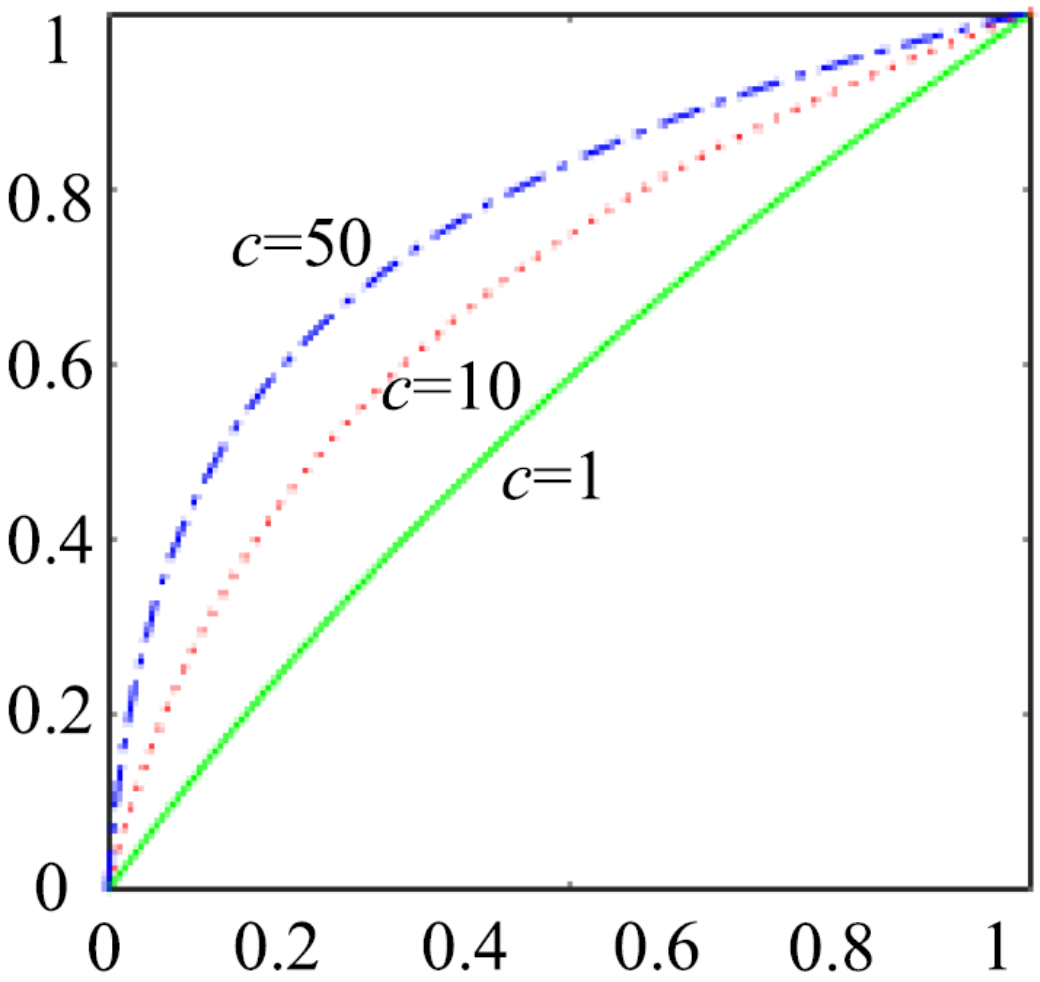


Figure 2.99: Log correction for different control parameter values

For values of the control constant greater than 1 we see that the function can enhance the dark areas of an image, so it could be applied for LLIE. This is also a simple algorithm, in terms of implementation. The implementation of the method is done with the function presented in image B.2.9 of appendix B. This function is applied to all LL images for each darkness level, which is done through the script presented in images B.2.10 and B. 2.11 of appendix B. Obviously, as in the case of the gamma transform, there is no ideal value of the constant c, and for this reason we will experiment with 3 different values. Specifically, we will apply the method with a small value of c = 1, a medium range value of c=10, and a large value of c=50. From this process, the experimental results are obtained, for each value of the parameter, the evaluation of which is done through the script presented in figure B.2.12 of appendix B. From this process, 3 excel files are produced for each value of the constant c, which contain the values of the metrics for each image of the experimental results, which will be used for the construction of summary tables, presented below. In addition, we also construct line charts with maximum/minimum/average PSNR and SSIM per darkness level, to see how the performance is affected by the darkness level. Finally, we also present line charts of the average PSNR and SSIM, per darkness level, both for the experimental results and for the original LL images, to see how much the result improves in each case.

Training Set

c=1

MSE					
Level	3.0	3.5	4.0	4.5	5.0
MIN	1150,396	1230,74	1294,044	1343,92	1373,67
MAX	25750,42	26525,18	27074,25	27510,78	27783,26
AVERAGE	10438,65	10920,6	11292,59	11593,25	12014,23
	PSNR(dB)				
Level	3.0	3.5	4.0	4.5	5.0
MIN	4,02296	3,894221	3,80524	3,735775	3,692972
MAX	17,52233	17,22914	17,01131	16,84707	16,75198
AVERAGE	8,246039	8,043762	7,893969	7,773268	7,620914
	SSIM				
Level	3.0	3.5	4.0	4.5	5.0
MIN	0,014623	0,010004	0,005747	0,003263	0,002069
MAX	0,391483	0,302258	0,234753	0,185888	0,150379
AVERAGE	0,067145	0,045045	0,030031	0,019979	0,01319

	MV				
Level	3.0	3.5	4.0	4.5	5.0
MIN	1,592122	1,044902	0,68809	0,424698	0,263826
MAX	15,11501	11,56136	8,694798	6,439869	4,607421
AVERAGE	6,876141	4,914041	3,447475	2,371986	1,641106
	STD				
Level	3.0	3.5	4.0	4.5	5.0
MIN	0,927363	0,59725	0,418809	0,419849	0,333316
MAX	95,58026	72,70247	54,30994	39,16158	26,71166
AVERAGE	15,71822	10,13654	6,405503	3,974903	2,458903
	BRISQUE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	31,54555	35,82797	38,55014	39,18579	41,19538
MAX	64,45784	61,20449	61,1918	56,83263	55,68347
AVERAGE	45,65275	45,51481	45,18829	44,84094	44,83776
	NIQE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	4,032914	4,19765	4,769732	4,956976	5,177708
MAX	13,04435	12,70734	11,64523	10,03232	10,37437
AVERAGE	6,924142	7,375425	7,782276	8,133508	8,38073

Table 2.16: Log correction results with c=1 on training set

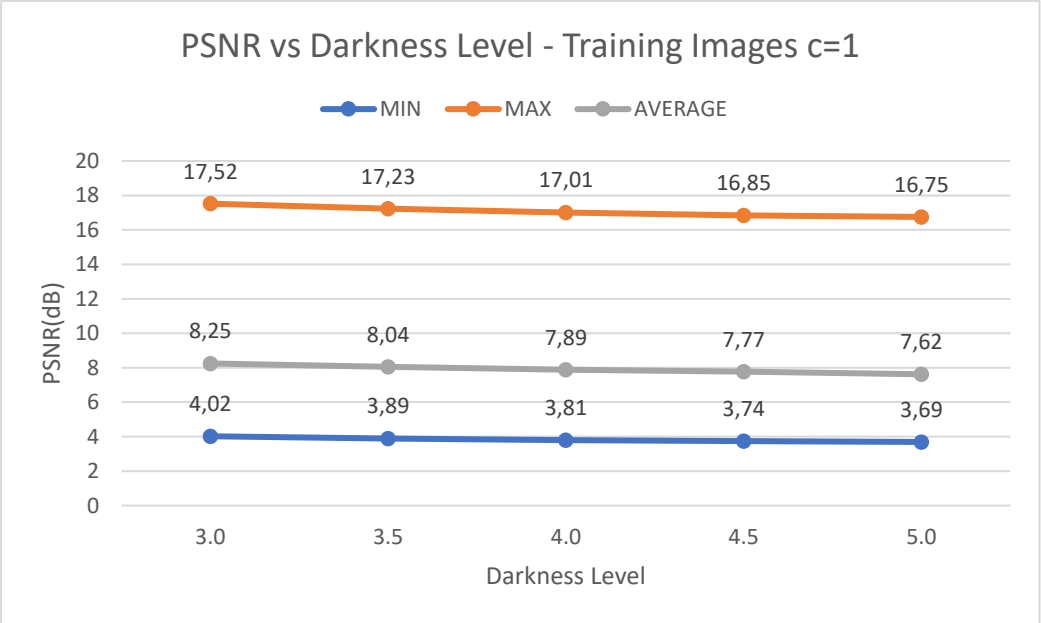


Figure 2.100: Experimental results PSNR vs Darkness Level for training images

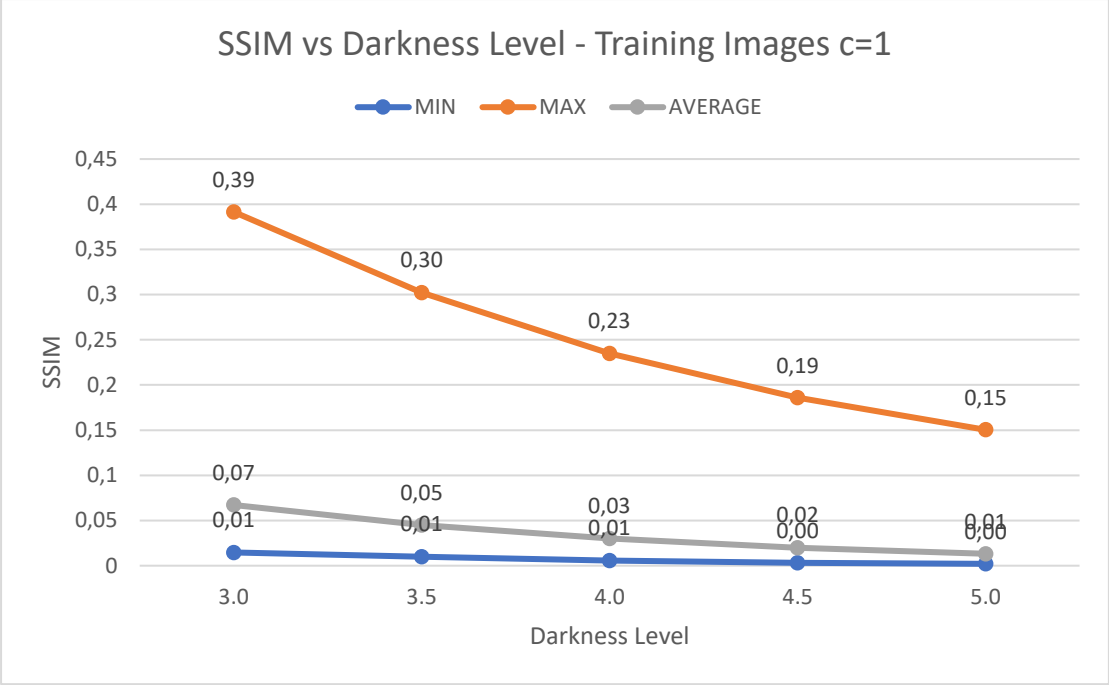


Figure 2.101: Experimental results SSIM vs Darkness Level for training images

For a control constant of 1 we see that the LL images do not improve at all. MSE is very large and increases with increasing darkness level, while PSNR and SSIM are very small and decrease with increasing darkness level. Regarding the metrics with no reference, we notice that MV and STD are very small and decrease with the increase of darkness level, which means that the experimental results remain dark and the pixel values accumulate close to zero. BRISQUE and NIQE increase with increasing darkness level, which means that the image quality gets worse.

c = 10

MSE					
Level	3.0	3.5	4.0	4.5	5.0
MIN	42,11946	169,5279	159,4157	506,3822	906,1422
MAX	12458,48	14923,75	17766,71	20328,87	22417,82
AVERAGE	2933,97	4435,592	5962,914	7399,069	8773,965
	PSNR(dB)				
Level	3.0	3.5	4.0	4.5	5.0
MIN	7,176152	6,392023	5,634734	5,04967	4,62487
MAX	31,88598	25,83839	26,10549	21,08602	18,55884
AVERAGE	14,2882	12,22883	10,81663	9,806838	9,054303

	SSIM				
Level	3.0	3.5	4.0	4.5	5.0
MIN	0,155937	0,108413	0,077067	0,056754	0,037932
MAX	0,884471	0,839788	0,860778	0,792306	0,642864
AVERAGE	0,573592	0,439176	0,313687	0,212219	0,141129
	MV				
Level	3.0	3.5	4.0	4.5	5.0
MIN	13,77962	9,595425	6,810115	4,803523	3,874113
MAX	85,06351	69,68224	56,105	44,31212	33,91846
AVERAGE	50,01966	37,99691	28,15938	20,38493	14,79352
	STD				
Level	3.0	3.5	4.0	4.5	5.0
MIN	21,38145	16,62244	11,08291	9,911588	8,102828
MAX	868,4807	745,0139	603,2324	478,8048	357,9306
AVERAGE	233,8997	170,6357	120,3637	82,02715	54,81495
	BRISQUE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	14,79688	17,84928	21,70912	25,00103	25,54882
MAX	50,35989	49,89327	48,38213	47,79668	48,59574
AVERAGE	36,311	37,22384	37,85137	38,28925	38,61274
	NIQE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	2,644477	2,714278	3,099516	3,285837	3,409037
MAX	26,92967	27,54845	12,37783	6,954204	7,303726
AVERAGE	4,486564	4,603463	4,669125	4,757833	4,844949

Table 2.17: Log correction results with c=10 on training set

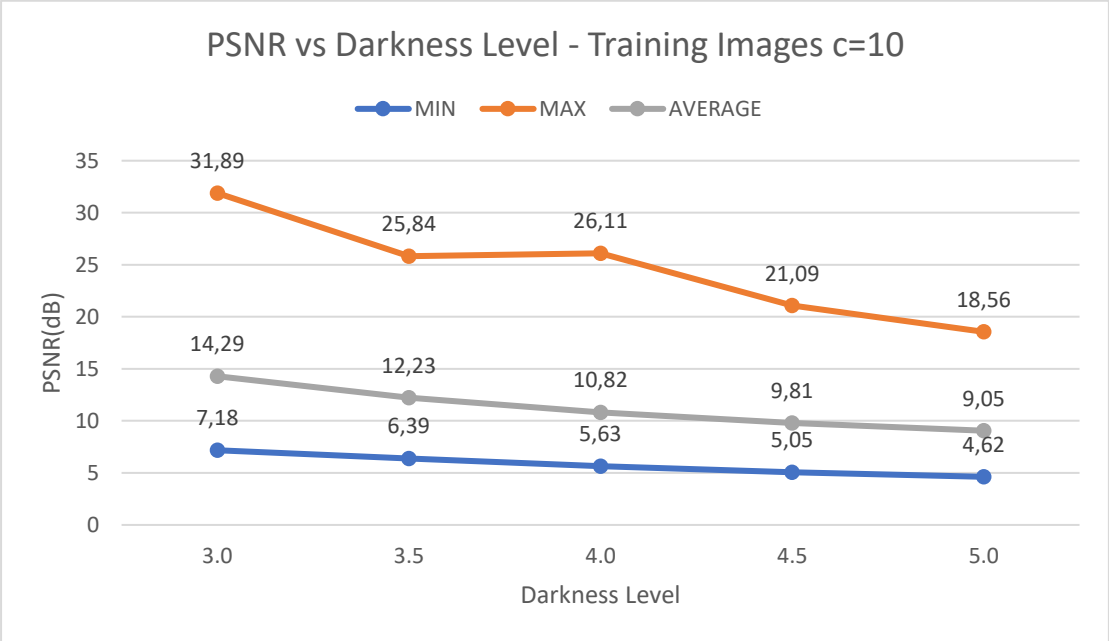


Figure 2.102: Experimental results PSNR vs Darkness Level for training images

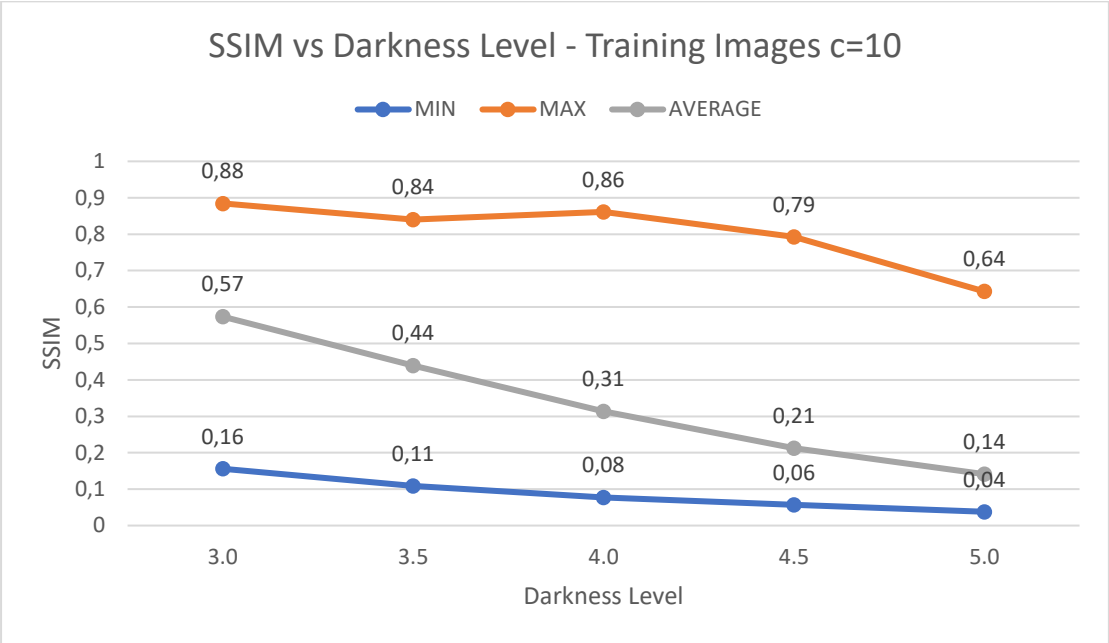


Figure 2.103: Experimental results SSIM vs Darkness Level for training images

For a control constant of 10, there is an improvement in the result. The quality metrics improve with respect to the corresponding values of the LL images, and the expected behavior is observed with the increase of the darkness level. MSE increases, while PSNR and SSIM decrease with increasing darkness level, indicating that the performance of the method decreases with increasing darkness level. Regarding the metrics with no reference, MV and STD are larger than the c=1 case, and decrease with increasing darkness level. Furthermore, BRISQUE and NIQE increase as the darkness level increases, which means that we move away from natural statistics, with the visual quality of the experimental result decreasing with increasing darkness level.

c = 50

	MSE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	188,0061	137,5447	103,7085	148,5744	198,1153
MAX	15318,49	11072,97	7479,269	9138,27	12133,61
AVERAGE	2582,41	1026,314	706,503	1394,934	2798,44
	PSNR(dB)				
Level	3.0	3.5	4.0	4.5	5.0
MIN	6,278643	7,688162	9,392212	8,522164	7,290905
MAX	25,38908	26,74637	27,97266	26,41136	25,16162
AVERAGE	14,85651	19,15856	21,01419	17,75816	14,62238

	SSIM				
Level	3.0	3.5	4.0	4.5	5.0
MIN	0,165409	0,148303	0,119653	0,060882	0,016329
MAX	0,900087	0,930409	0,90992	0,849867	0,776973
AVERAGE	0,594567	0,599284	0,564726	0,487891	0,387825
	MV				
Level	3.0	3.5	4.0	4.5	5.0
MIN	48,12243	35,75146	27,04635	20,31815	17,99765
MAX	194,585	163,5684	135,5424	116,0806	100,3914
AVERAGE	137,3111	112,9854	90,36633	70,33211	54,49079
	STD				
Level	3.0	3.5	4.0	4.5	5.0
MIN	80,21196	89,77202	95,60216	81,21205	75,39581
MAX	2085,714	1799,995	1472,419	1314,607	1208,574
AVERAGE	713,2332	612,9932	502,3531	395,2466	302,7045
	BRISQUE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	5,358579	4,680931	7,847411	11,00519	12,55915
MAX	51,44345	50,52788	53,30538	54,53145	55,62918
AVERAGE	32,09664	30,95293	31,97628	33,86062	35,9871
	NIQE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	2,242354	2,173021	2,566859	2,948489	3,051233
MAX	25,61689	26,02209	8,833151	7,446648	8,212591
AVERAGE	3,928974	4,089681	4,30353	4,587479	4,888292

Table 2.18: Log correction results with c=50 on training set

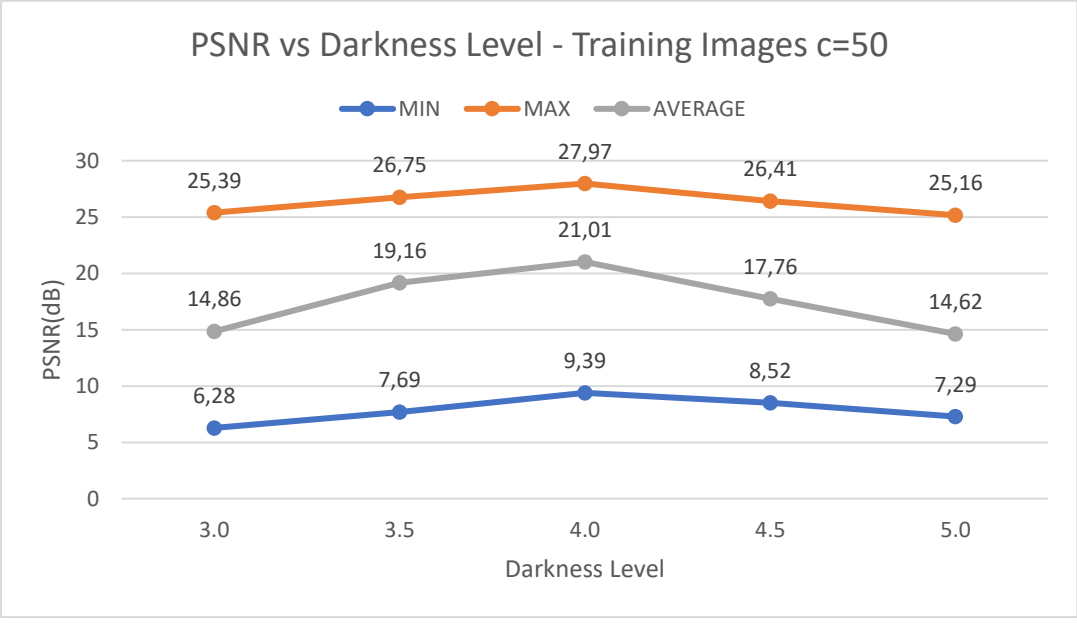


Figure 2.104: Experimental results PSNR vs Darkness Level for training images

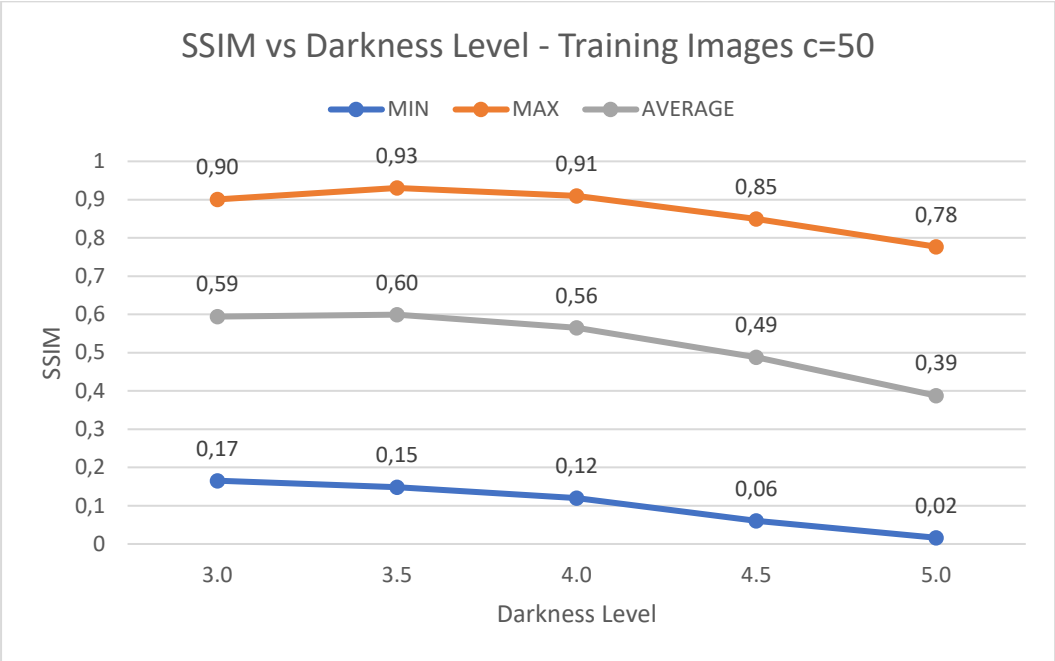
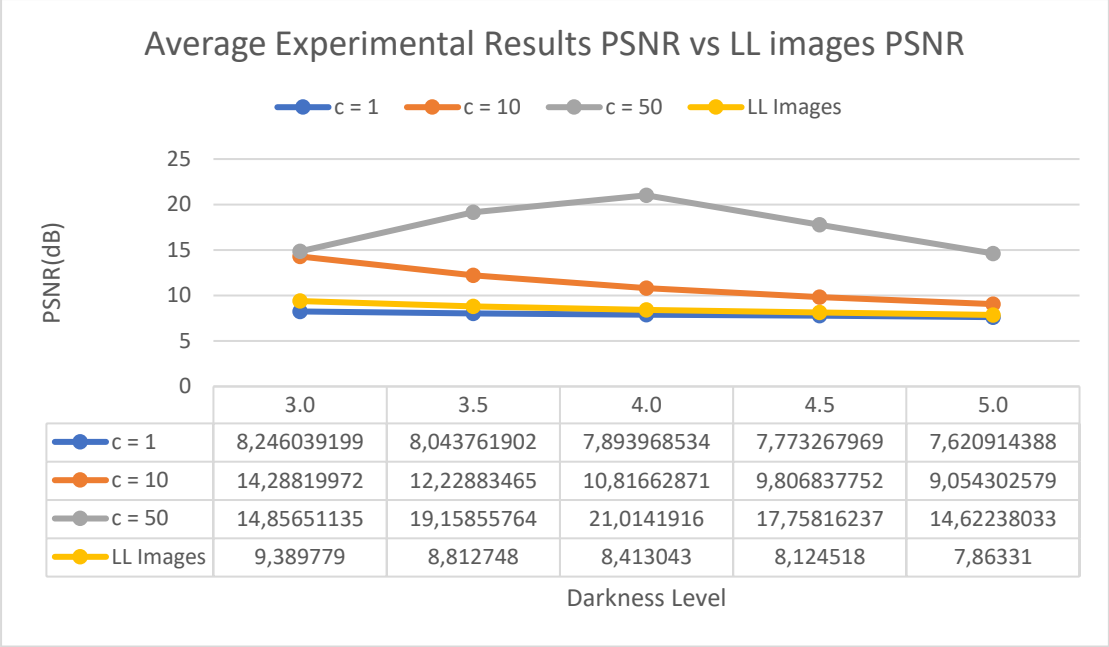


Figure 2.105: Experimental results SSIM vs Darkness Level for training images

For a control constant of 50 the best results are obtained, compared to the other two cases. All quality metrics improve significantly compared to the corresponding values of LL images, and follow the expected behavior with increasing darkness level.

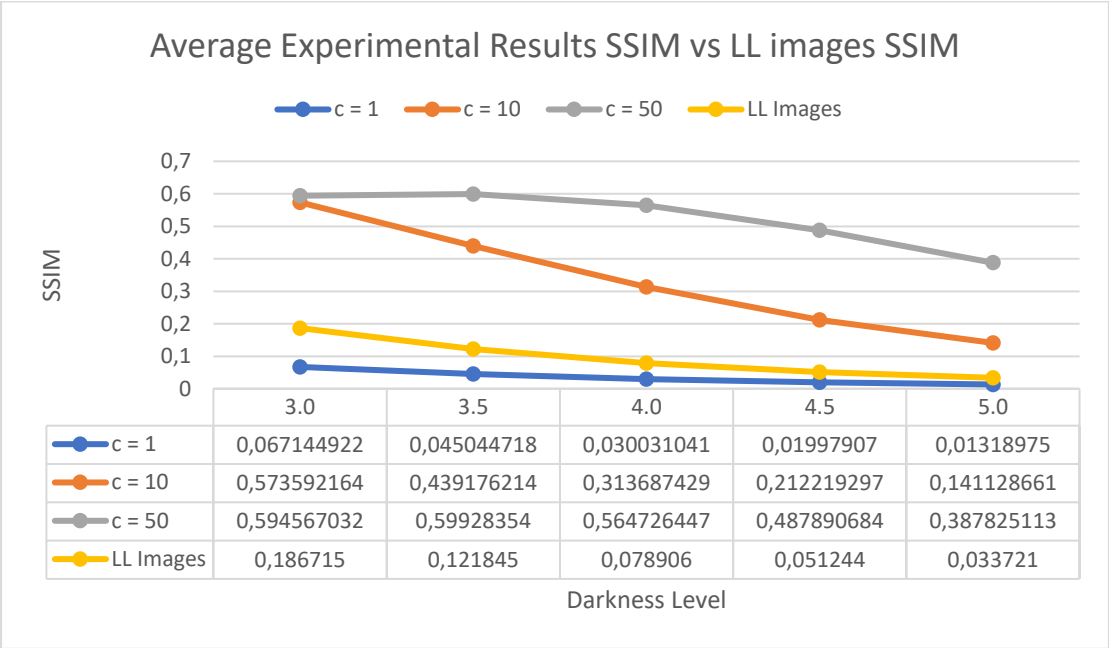
From Figures 2.106 and 2.107 we see that indeed the best results are obtained for a control constant of 50, with the PSNR and SSIM values improving noticeably. In addition, we notice that for control constant 1 the result is worse than the original LL images, i.e. instead of an improvement we have a degradation of the result. Finally, for a control constant of 10, the improvement is minimal.



	3.0	3.5	4.0	4.5	5.0
c = 1	8,246039199	8,043761902	7,893968534	7,773267969	7,620914388
c = 10	14,28819972	12,22883465	10,81662871	9,806837752	9,054302579
c = 50	14,85651135	19,15855764	21,0141916	17,75816237	14,62238033
LL Images	9,389779	8,812748	8,413043	8,124518	7,86331

Darkness Level

Figure 2.106: Experimental results average PSNR vs LL Images PSNR training set



	3.0	3.5	4.0	4.5	5.0
c = 1	0,067144922	0,045044718	0,030031041	0,01997907	0,01318975
c = 10	0,573592164	0,439176214	0,313687429	0,212219297	0,141128661
c = 50	0,594567032	0,59928354	0,564726447	0,487890684	0,387825113
LL Images	0,186715	0,121845	0,078906	0,051244	0,033721

Darkness Level

Figure 2.107: Experimental results average SSIM vs LL Images SSIM training set

Validation Set

c = 1

MSE					
Level	3.0	3.5	4.0	4.5	5.0
MIN	1628,71	1690,142	1737,611	1769,059	1790,034
MAX	19961	20625,95	21126,67	21462,78	21729,31
AVERAGE	11055,66	11556,84	11941,92	12231,32	12747,47
	PSNR(dB)				
Level	3.0	3.5	4.0	4.5	5.0
MIN	5,12898	4,986664	4,882492	4,813944	4,760345
MAX	16,01237	15,85157	15,73128	15,65338	15,60219
AVERAGE	8,04165	7,846894	7,702929	7,59776	7,458581
	SSIM				
Level	3.0	3.5	4.0	4.5	5.0
MIN	0,029973	0,018938	0,011746	0,007474	0,0045
MAX	0,126596	0,10804	0,097283	0,090328	0,086229
AVERAGE	0,060913	0,041013	0,027805	0,019165	0,013376
	MV				
Level	3.0	3.5	4.0	4.5	5.0
MIN	1,549638	1,000642	0,593799	0,37321	0,220961
MAX	10,91523	8,069728	5,872225	4,221559	2,959349
AVERAGE	6,916212	4,947855	3,482149	2,408004	1,63909
	STD				
Level	3.0	3.5	4.0	4.5	5.0
MIN	1,852095	1,164383	0,662383	0,414455	0,284926
MAX	95,10262	85,18198	75,6174	66,21395	57,91399
AVERAGE	17,35171	11,61218	7,815267	5,234261	3,926796
	BRISQUE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	41,96625	41,62356	41,54807	42,05184	41,65584
MAX	57,80742	55,87842	52,83205	55,46545	52,22314
AVERAGE	46,50328	46,10468	45,56008	45,2162	44,80103
	NIQE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	4,730215	5,706043	6,273572	6,443305	6,878924
MAX	9,099827	9,292974	9,07556	9,178307	9,672329
AVERAGE	6,824987	7,475221	7,90187	8,264813	8,483055

Table 2.19: Log correction results with c=1 on validation set

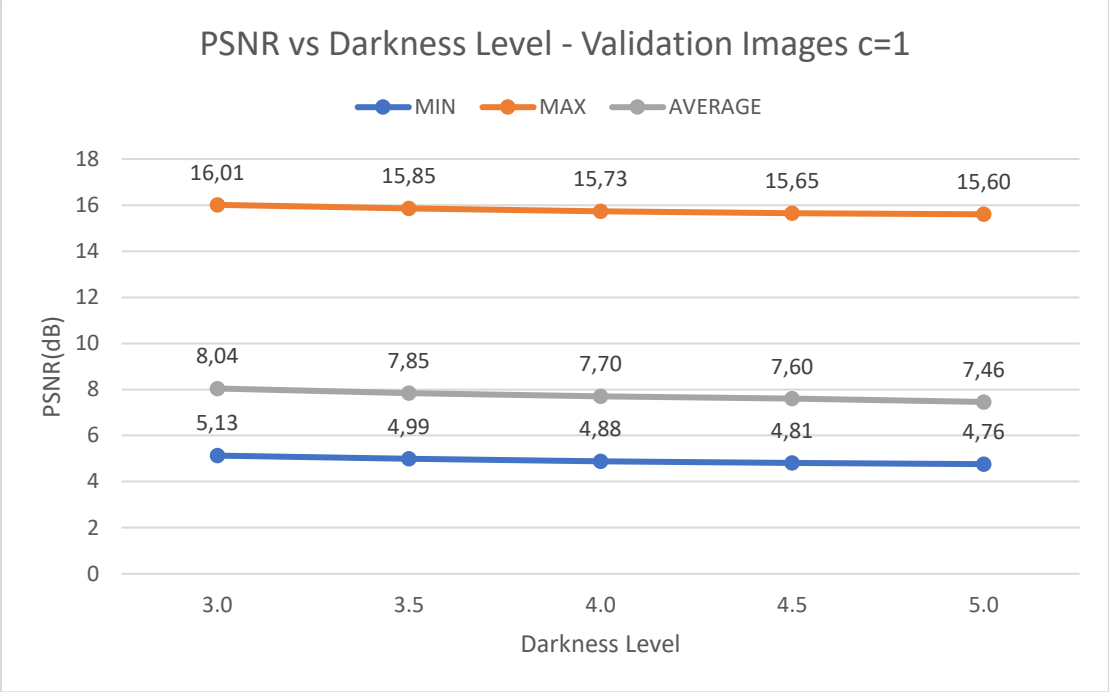


Figure 2.108: Experimental results PSNR vs Darkness Level for validation images

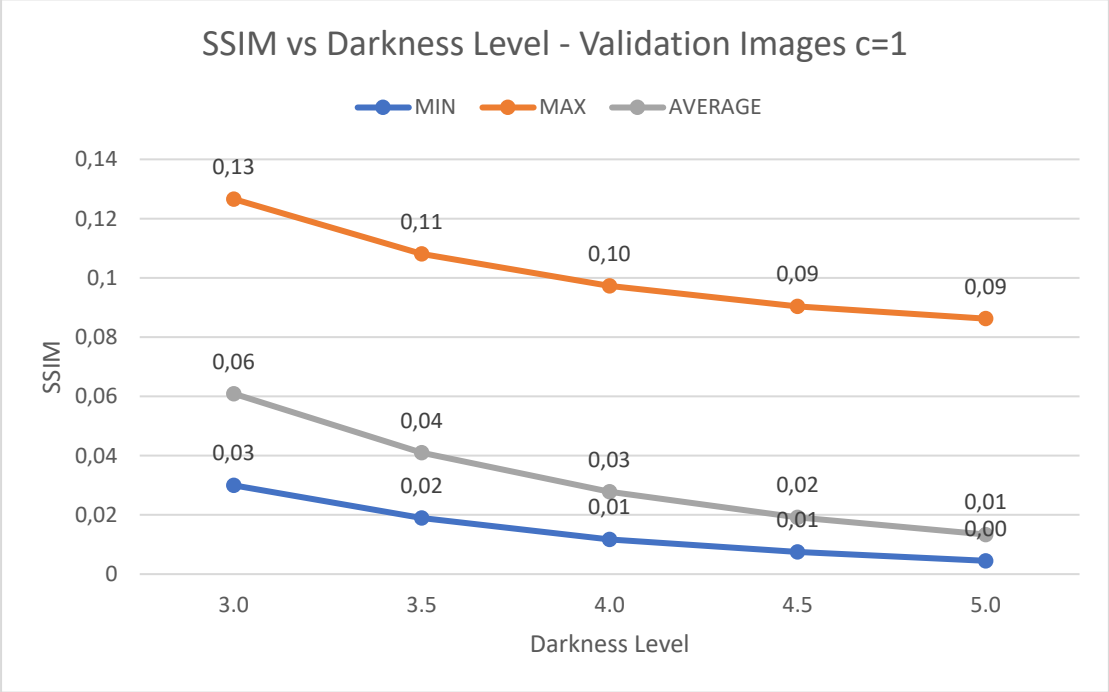


Figure 2.109: Experimental results SSIM vs Darkness Level for validation images

For control constant 1 the same behavior as the training set is observed. The values of the metrics do not improve at all compared to the original LL images, with the results being worse than the original ones, as can be seen from Figures 2.114 and 2.115.

c = 10

MSE					
Level	3.0	3.5	4.0	4.5	5.0
MIN	584,9529	860,4201	1115,709	1326,344	1473,978
MAX	8480,949	11176,47	13659,3	15697,94	17476,44
AVERAGE	3172,155	4764,539	6369,222	7853,734	9409,65
	PSNR(dB)				
Level	3.0	3.5	4.0	4.5	5.0
MIN	8,846359	7,647755	6,776521	6,172377	5,706274
MAX	20,45959	18,7837	17,6553	16,90424	16,44589
AVERAGE	13,69775	11,81964	10,50093	9,561354	8,814844
	SSIM				
Level	3.0	3.5	4.0	4.5	5.0
MIN	0,302136	0,257723	0,180801	0,109948	0,067585
MAX	0,722033	0,584069	0,472778	0,370078	0,242925
AVERAGE	0,557293	0,424795	0,303331	0,20648	0,132153
	MV				
Level	3.0	3.5	4.0	4.5	5.0
MIN	14,00318	9,689803	6,508856	4,391124	3,103696
MAX	71,03485	55,81298	43,6163	32,70043	24,20066
AVERAGE	50,17504	38,19036	28,40812	20,63152	14,72766
	STD				
Level	3.0	3.5	4.0	4.5	5.0
MIN	41,73803	28,80771	17,60765	11,17202	7,832456
MAX	729,9616	657,4974	592,1293	529,202	474,4819
AVERAGE	246,4542	182,063	131,3782	91,59303	64,6523
	BRISQUE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	23,89007	28,73192	32,41299	29,22142	34,34498
MAX	47,97598	46,69466	47,14542	46,89696	44,59126
AVERAGE	37,40734	37,92862	38,44142	38,55464	38,44289
	NIQE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	3,329063	3,49355	3,578824	4,017817	3,989238
MAX	5,402191	5,584824	5,932847	6,623467	6,163965
AVERAGE	4,388928	4,52681	4,657536	4,760438	4,851294

Table 2.20: Log correction results with c=10 on validation set

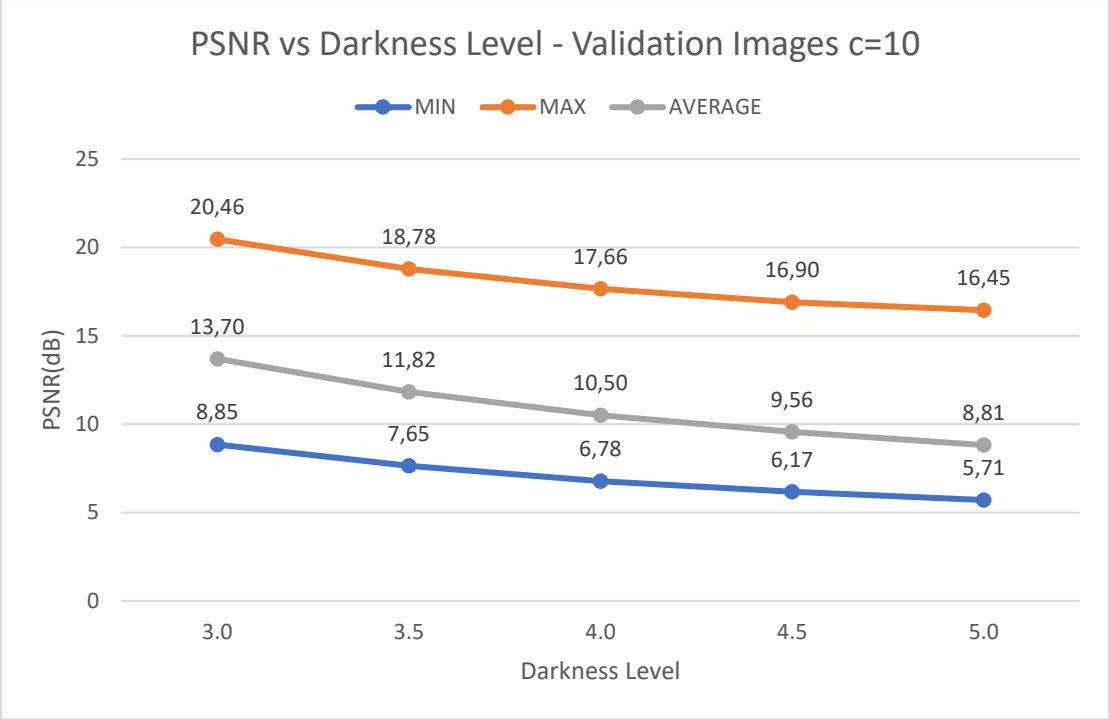


Figure 2.110: Experimental results PSNR vs Darkness Level for validation images

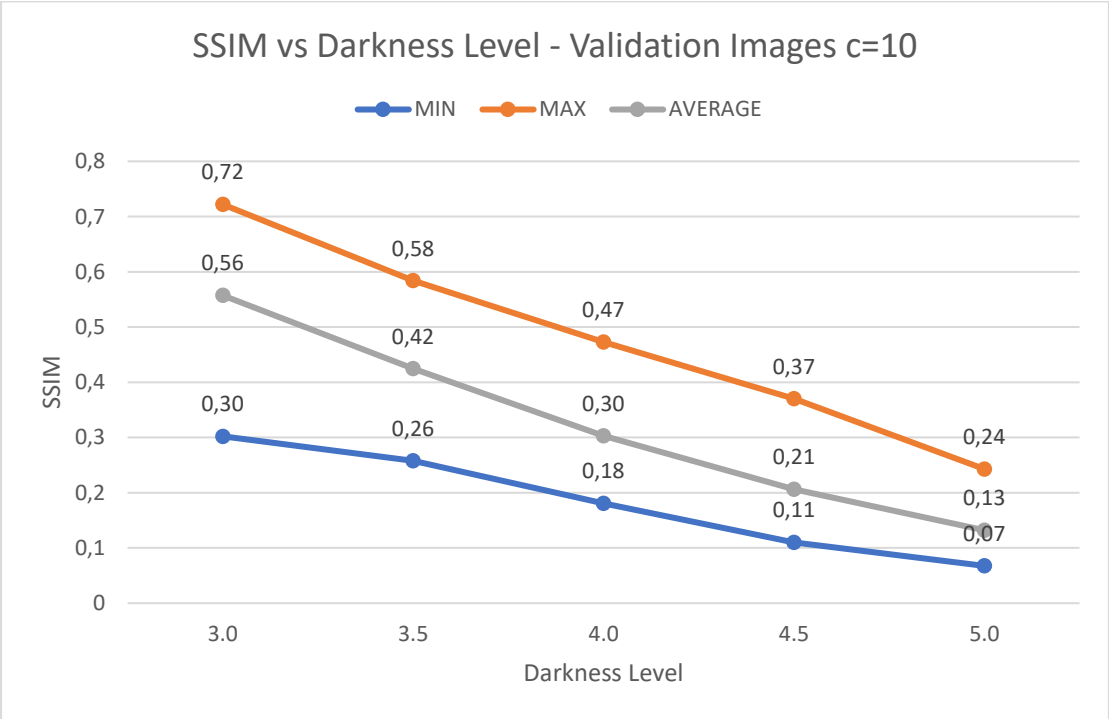


Figure 2.111: Experimental results SSIM vs Darkness Level for validation images

For a control constant of 10, the validation set shows the same behavior as the training set. There is a slight improvement in the quality metric values,

but with the increase in darkness level, the experimental results are not much different from those for LL images.

c = 50

MSE					
Level	3.0	3.5	4.0	4.5	5.0
MIN	173,0703	236,5667	163,379	427,6937	545,1057
MAX	8242,691	4688,932	2437,267	4800,684	7751,772
AVERAGE	2276,441	851,6959	666,5707	1444,341	3045,971
	PSNR(dB)				
Level	3.0	3.5	4.0	4.5	5.0
MIN	8,970114	11,42006	14,26177	11,31777	9,236794
MAX	25,74858	24,39127	25,99884	21,81947	20,766
AVERAGE	15,52221	19,72104	21,10747	17,25842	13,98398
	SSIM				
Level	3.0	3.5	4.0	4.5	5.0
MIN	0,209478	0,203483	0,141076	0,094871	0,14184
MAX	0,837241	0,851423	0,832643	0,7503	0,608247
AVERAGE	0,598857	0,606628	0,575211	0,505534	0,39874
	MV				
Level	3.0	3.5	4.0	4.5	5.0
MIN	52,20364	38,42516	27,23087	19,16659	14,10746
MAX	179,8483	150,9461	126,153	101,9985	81,69774
AVERAGE	137,03	113,0062	90,73357	70,94576	54,28716
	STD				
Level	3.0	3.5	4.0	4.5	5.0
MIN	152,4524	151,1566	137,5207	97,73425	73,37161
MAX	1235,666	1228,703	1166,116	980,6299	798,9623
AVERAGE	737,1489	634,7624	524,3859	412,5428	313,7163
	BRISQUE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	12,28138	7,513229	10,44922	11,58201	11,80987
MAX	45,81359	48,7408	54,85708	52,25491	50,84027
AVERAGE	34,03565	32,62372	32,83111	33,14358	34,43288
	NIQE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	2,761001	3,017186	3,114857	3,519655	3,699833
MAX	5,54555	5,553362	6,460767	6,504754	7,424094
AVERAGE	3,838583	3,971662	4,205644	4,483011	4,811183

Table 2.21: Log correction results with c=50 on validation set

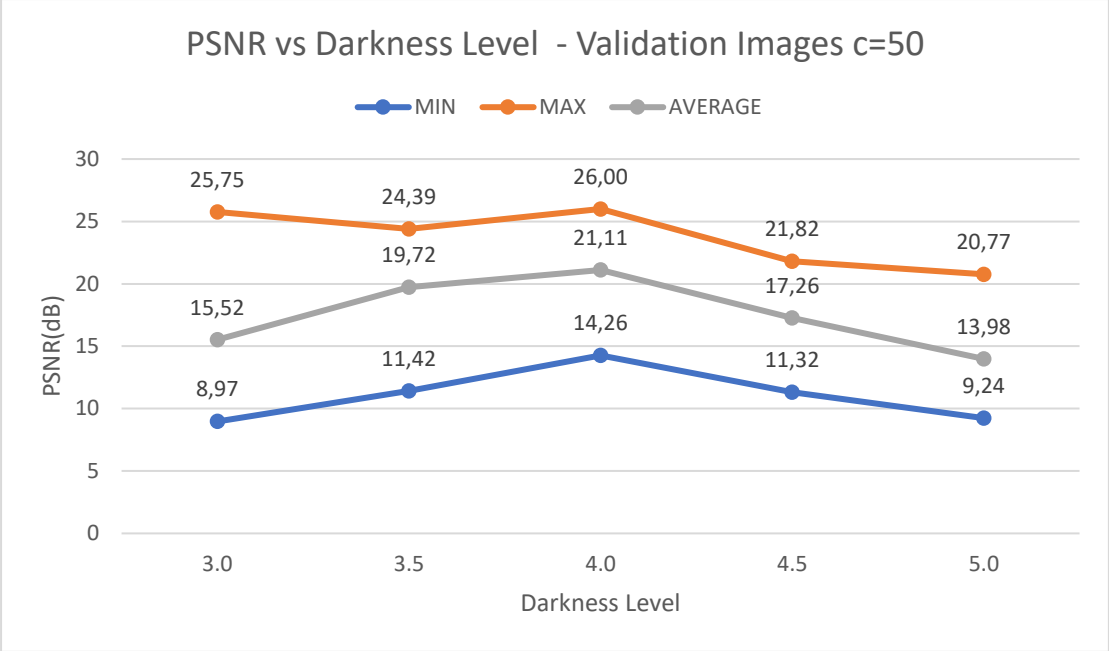


Figure 2.112: Experimental results PSNR vs Darkness Level for validation images

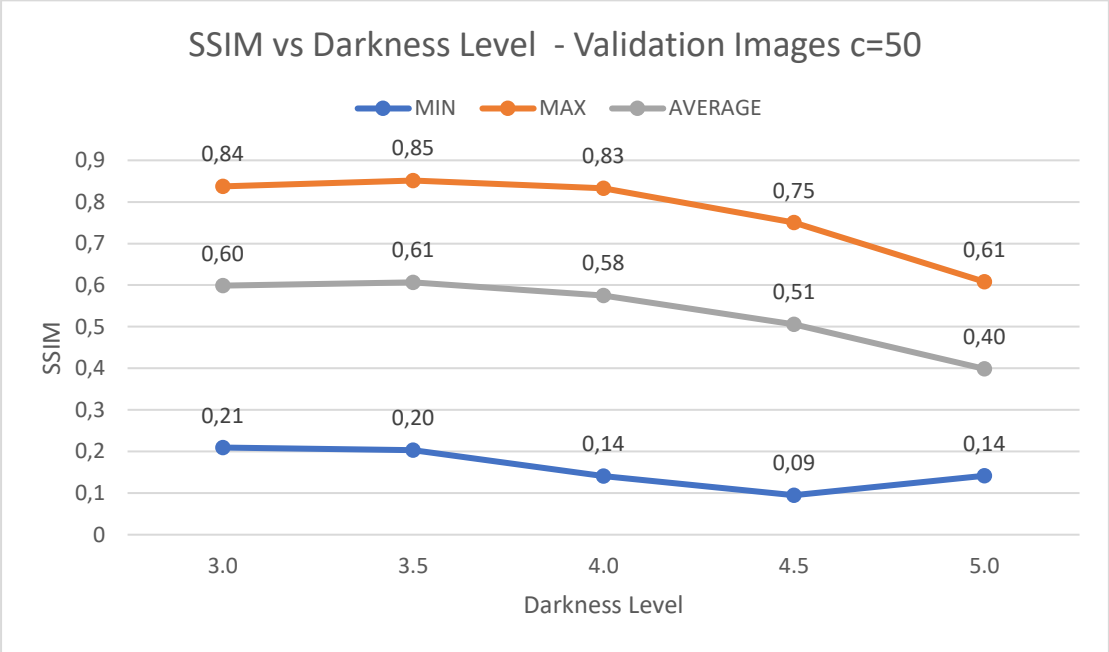
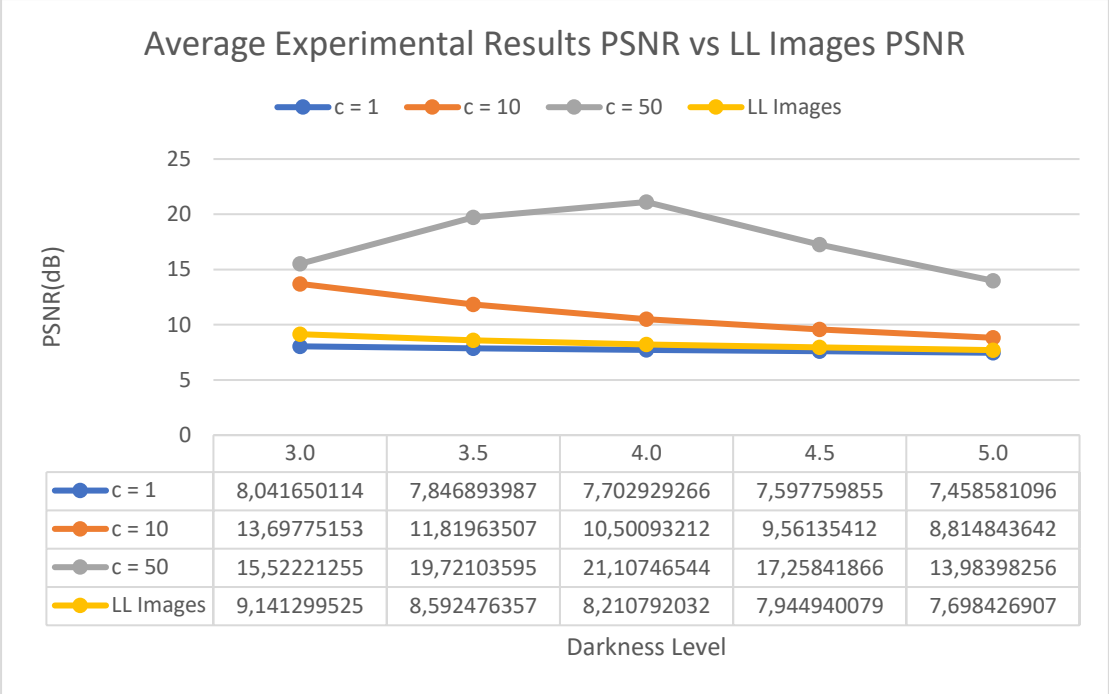


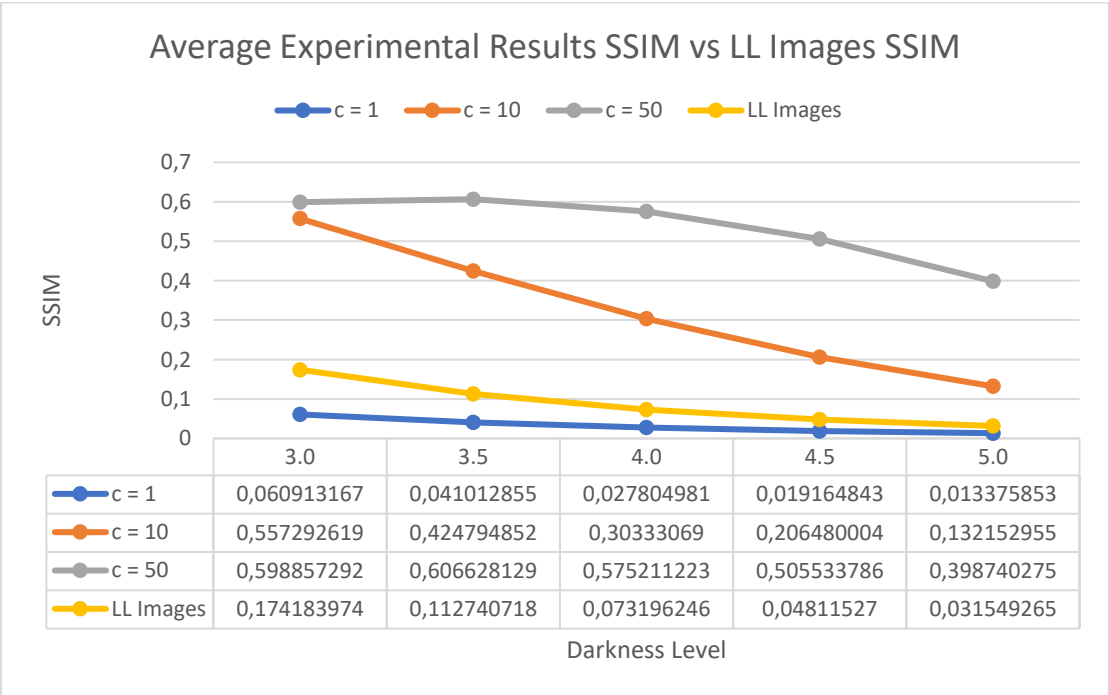
Figure 2.113: Experimental results SSIM vs Darkness Level for validation images

For a control constant of 50, the best results are obtained for the validation set, as it was for the training set. There is a clear improvement in the quality metric values compared to the original LL images, as can be seen in Figures 2.114 and 2.115.



	3.0	3.5	4.0	4.5	5.0
c = 1	8,041650114	7,846893987	7,702929266	7,597759855	7,458581096
c = 10	13,69775153	11,81963507	10,50093212	9,56135412	8,814843642
c = 50	15,52221255	19,72103595	21,10746544	17,25841866	13,98398256
LL Images	9,141299525	8,592476357	8,210792032	7,944940079	7,698426907

Figure 2.114: Experimental results average PSNR vs LL Images PSNR validation set



	3.0	3.5	4.0	4.5	5.0
c = 1	0,060913167	0,041012855	0,027804981	0,019164843	0,013375853
c = 10	0,557292619	0,424794852	0,30333069	0,206480004	0,132152955
c = 50	0,598857292	0,606628129	0,575211223	0,505533786	0,398740275
LL Images	0,174183974	0,112740718	0,073196246	0,04811527	0,031549265

Figure 2.115: Experimental results average SSIM vs LL Images SSIM validation set

Test Set

c = 1

MSE					
Level	3.0	3.5	4.0	4.5	5.0
MIN	2043,187	2116,679	2174,85	2212,273	2231,654
MAX	21006,29	21750,28	22342,17	22774,62	23073,27
AVERAGE	10441,65	10929,08	11306,31	11572,29	11615,42
	PSNR(dB)				
Level	3.0	3.5	4.0	4.5	5.0
MIN	4,90731	4,756155	4,63955	4,556292	4,499712
MAX	15,02772	14,87425	14,75651	14,68242	14,64454
AVERAGE	8,243062	8,039969	7,889197	7,789404	7,806113
	SSIM				
Level	3.0	3.5	4.0	4.5	5.0
MIN	0,021674	0,012547	0,006574	0,003317	0,00357
MAX	0,244173	0,171878	0,125829	0,092113	0,069254
AVERAGE	0,061807	0,040918	0,02688	0,017739	0,01183
	MV				
Level	3.0	3.5	4.0	4.5	5.0
MIN	1,966409	1,320028	0,781811	0,444646	0,273638
MAX	11,98828	9,002481	6,723855	4,823458	3,445938
AVERAGE	6,911006	4,940685	3,461869	2,375729	1,607916
	STD				
Level	3.0	3.5	4.0	4.5	5.0
MIN	2,237172	1,388126	0,717591	0,400911	0,31117
MAX	63,59741	47,36323	33,6482	22,98484	8,388868
AVERAGE	15,13708	9,651144	6,017383	3,714627	2,151359
	BRISQUE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	32,76911	38,39742	38,34528	41,59701	41,65263
MAX	64,61011	68,47998	59,28402	52,70005	52,49523
AVERAGE	46,08738	46,21883	45,42459	45,1587	45,26535
	NIQE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	4,342711	4,619607	5,13221	5,223913	5,369609
MAX	9,555313	10,0075	10,01027	9,726514	9,414107
AVERAGE	6,906518	7,368854	7,747159	8,085139	8,42788

Table 2.22: Log correction results with c=1 on test set

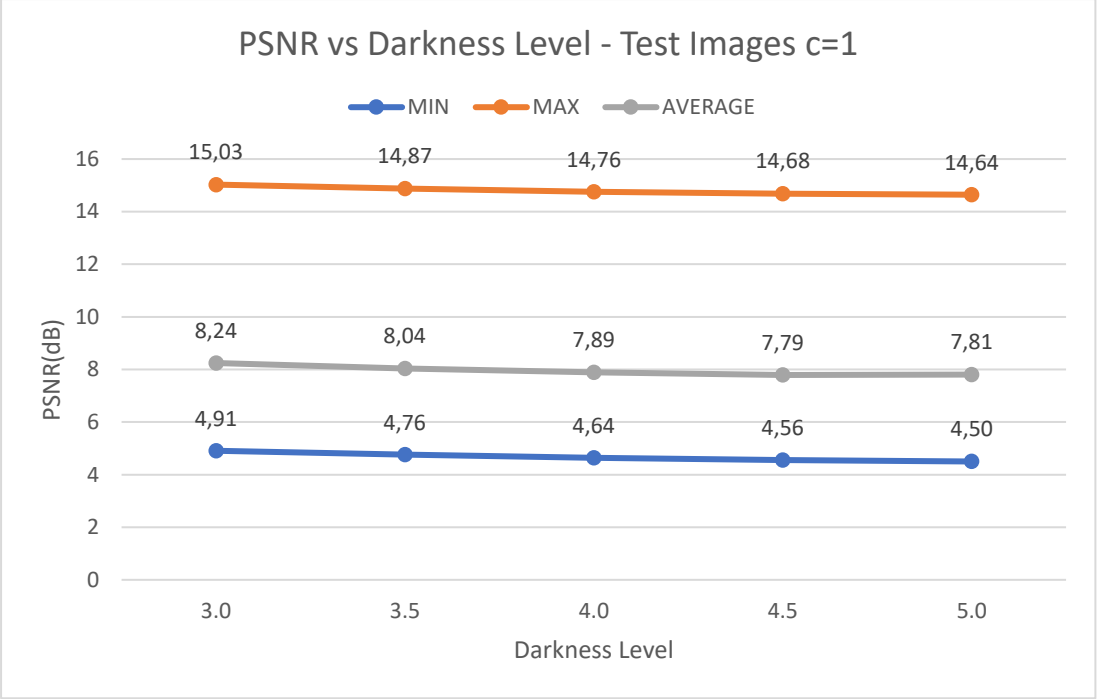


Figure 2.116: Experimental results PSNR vs Darkness Level for test images

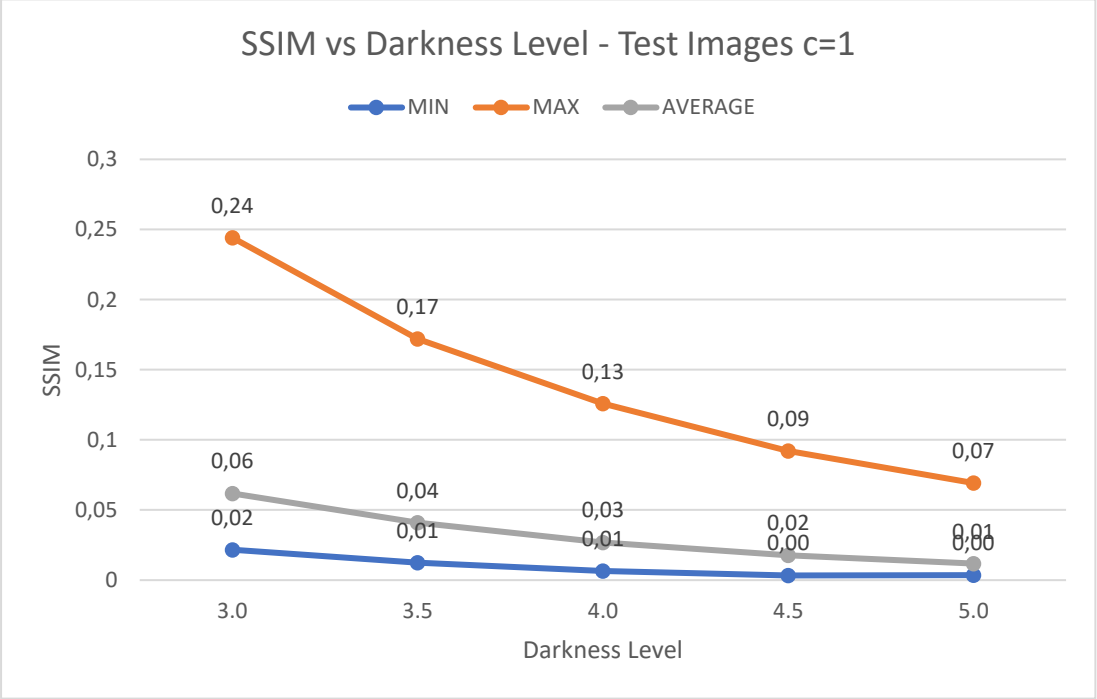


Figure 2.117: Experimental results SSIM vs Darkness Level for test images

For a control constant of 1, the test set exhibits the same behavior as the training and validation sets. The quality metrics do not improve at all, but become worse than those of the original LL images. This can also be

confirmed from Figures 2.122 and 2.123, with the average values of PSNR and SSIM for each control constant value.

c = 10

MSE					
Level	3.0	3.5	4.0	4.5	5.0
MIN	78,54687	339,1828	1028,517	1644,707	1866,827
MAX	8578,332	11223,44	13933,41	16293,11	18203,23
AVERAGE	2895,467	4396,903	5932,394	7355,09	8445,688
	PSNR(dB)				
Level	3.0	3.5	4.0	4.5	5.0
MIN	8,796775	7,629543	6,69023	6,010764	5,52932
MAX	29,17951	22,82647	18,00869	15,96992	15,41976
AVERAGE	14,31675	12,24365	10,81243	9,821376	9,229135
	SSIM				
Level	3.0	3.5	4.0	4.5	5.0
MIN	0,214692	0,176644	0,128681	0,074542	0,050204
MAX	0,90248	0,821121	0,674199	0,554673	0,429683
AVERAGE	0,572003	0,43341	0,304707	0,203949	0,132794
	MV				
Level	3.0	3.5	4.0	4.5	5.0
MIN	16,45095	11,50642	7,997039	5,13745	3,584051
MAX	73,0239	58,57621	46,51969	35,59909	26,87461
AVERAGE	50,16085	38,14374	28,27685	20,40455	14,50323
	STD				
Level	3.0	3.5	4.0	4.5	5.0
MIN	47,87727	33,55352	18,83544	10,96006	7,657869
MAX	606,5987	506,9348	406,7248	315,5567	150,3359
AVERAGE	231,6704	167,1262	116,2235	78,27165	49,53739
	BRISQUE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	22,40827	21,88402	23,65389	26,01816	28,05007
MAX	48,54499	50,25315	49,03543	49,23788	48,00299
AVERAGE	36,60648	37,70654	38,23703	38,70577	38,68672
	NIQE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	3,182201	3,338981	3,401796	3,702826	3,751078
MAX	7,200654	6,932764	6,261882	7,044526	7,330747
AVERAGE	4,437645	4,548818	4,679047	4,815784	4,906994

Table 2.23: Log correction results with c=10 on test set

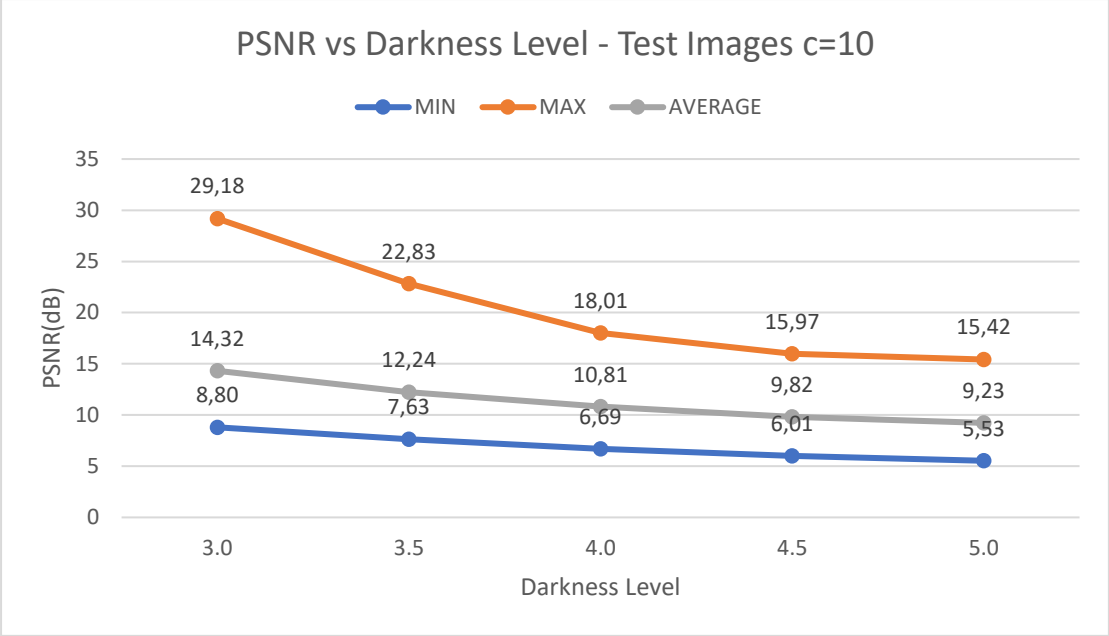


Figure 2.118: Experimental results PSNR vs Darkness Level for test images

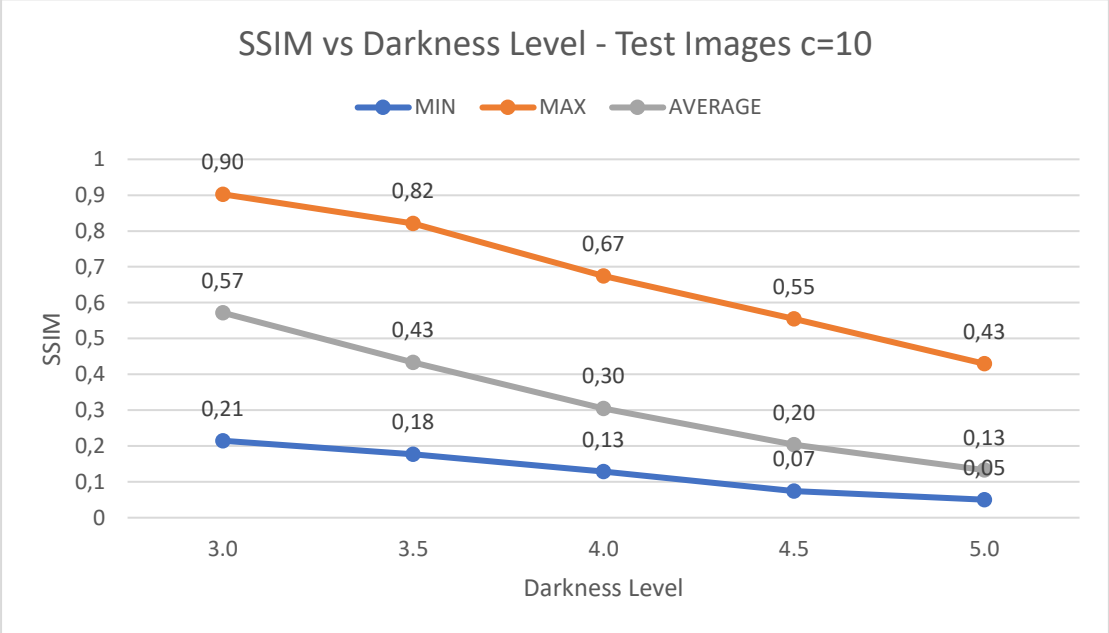


Figure 2.119: Experimental results SSIM vs Darkness Level for test images

For a control constant of 10 we observe that there is a minimal improvement of the quality metric values, following the same behavior as the training and validation sets. The metric values are only marginally better than those of the original LL images, and become worse as the darkness level increases.

c = 50

MSE					
Level	3.0	3.5	4.0	4.5	5.0
MIN	239,579	270,5548	154,0493	230,494	191,5334
MAX	11332,93	7503,139	3853,38	6294,927	8112,887
AVERAGE	2614,328	1038,526	703,5616	1372,141	2649,829
	PSNR(dB)				
Level	3.0	3.5	4.0	4.5	5.0
MIN	7,587383	9,378374	12,27239	10,1409	9,03905
MAX	24,33632	23,80825	26,25421	24,50421	25,30836
AVERAGE	14,82221	19,0108	20,9166	17,57403	14,59449
	SSIM				
Level	3.0	3.5	4.0	4.5	5.0
MIN	0,194941	0,174166	0,121044	0,082028	0,047548
MAX	0,847754	0,892556	0,908981	0,824637	0,720324
AVERAGE	0,610923	0,61743	0,581146	0,502621	0,385608
	MV				
Level	3.0	3.5	4.0	4.5	5.0
MIN	56,65485	41,68285	30,39575	22,80462	16,68435
MAX	172,1099	149,3428	126,0709	103,3519	83,8651
AVERAGE	137,4292	113,1232	90,4949	70,26011	53,44228
	STD				
Level	3.0	3.5	4.0	4.5	5.0
MIN	174,912	130,5094	115,6348	98,48667	73,50882
MAX	1894,964	1575,038	1240,717	938,2727	708,0402
AVERAGE	733,639	619,883	499,3963	387,9718	284,3082
	BRISQUE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	16,1751	12,84582	11,00716	11,23866	16,30634
MAX	46,18415	48,72177	55,369	53,35234	58,48296
AVERAGE	33,46303	32,71173	32,01516	33,30971	36,2968
	NIQE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	2,593586	2,888011	3,109754	3,388232	3,577005
MAX	6,42902	6,330552	6,556469	8,485191	8,281337
AVERAGE	3,837928	4,016739	4,272004	4,62412	4,918593

Table 2.24: Log correction results with c=50 on test set

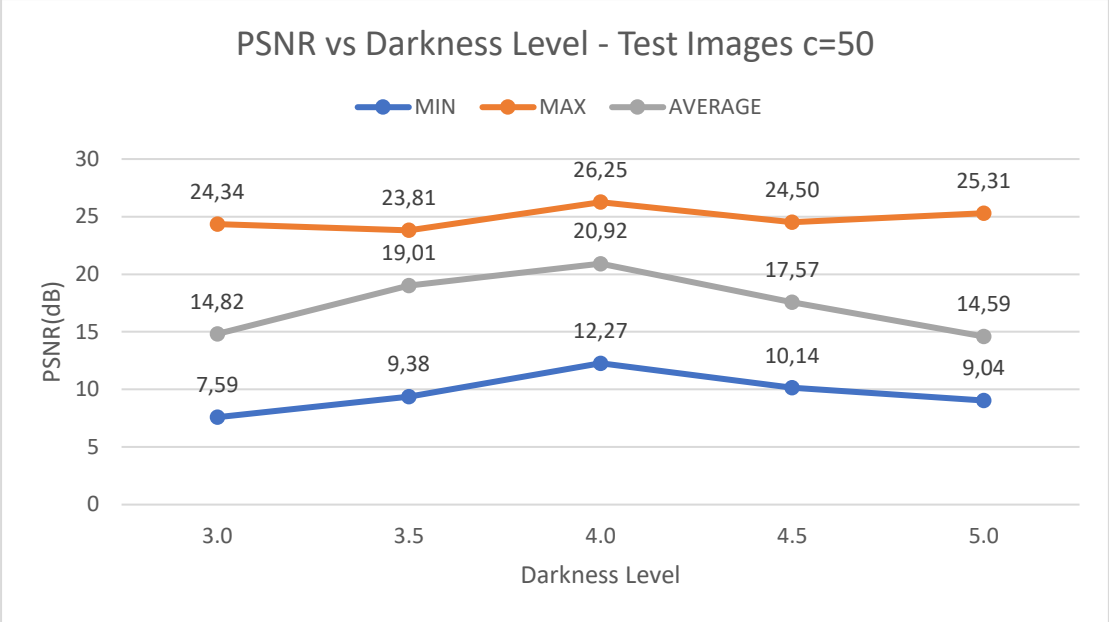


Figure 2.120: Experimental results PSNR vs Darkness Level for test images

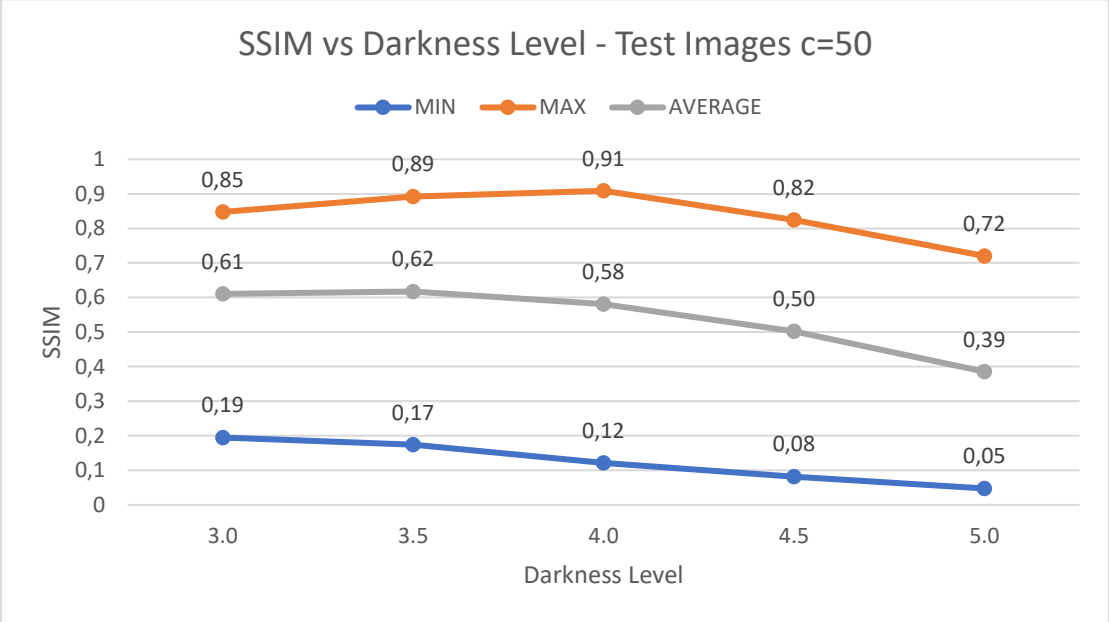
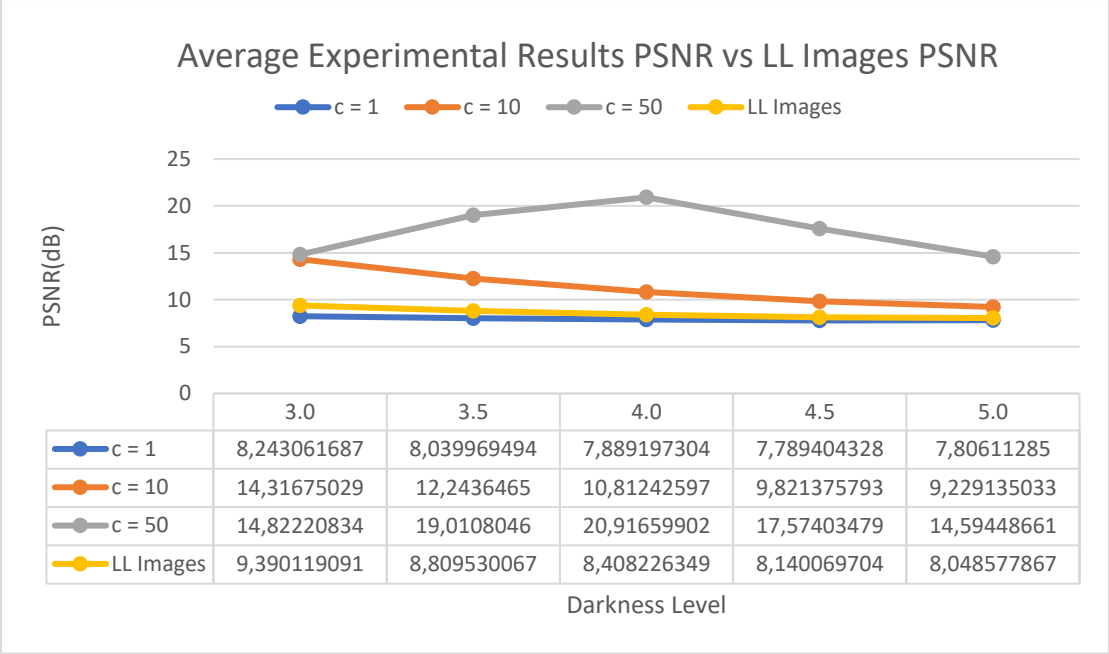


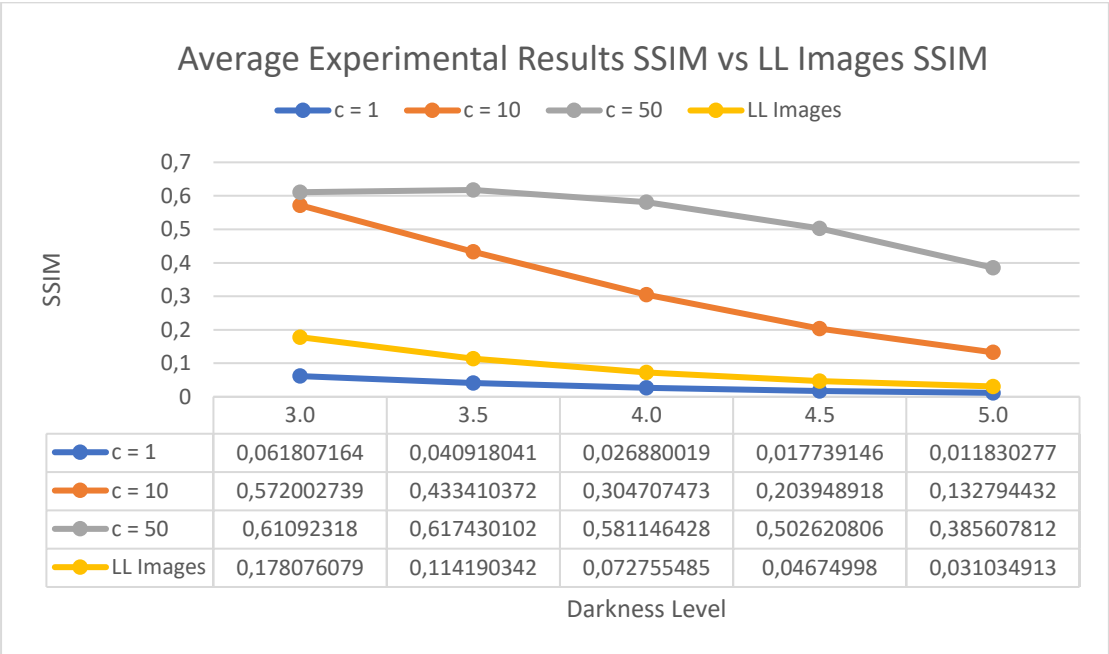
Figure 2.121: Experimental results SSIM vs Darkness Level for test images

The case of control constant 50 gives the best results, compared to the other two values, just as it happens in the training and validation sets, with the values of the quality metrics improving noticeably.



	3.0	3.5	4.0	4.5	5.0
c = 1	8,243061687	8,039969494	7,889197304	7,789404328	7,80611285
c = 10	14,31675029	12,2436465	10,81242597	9,821375793	9,229135033
c = 50	14,82220834	19,0108046	20,91659902	17,57403479	14,59448661
LL Images	9,390119091	8,809530067	8,408226349	8,140069704	8,048577867

Figure 2.122: Experimental results average PSNR vs LL Images PSNR test set



	3.0	3.5	4.0	4.5	5.0
c = 1	0,061807164	0,040918041	0,026880019	0,017739146	0,011830277
c = 10	0,572002739	0,433410372	0,304707473	0,203948918	0,132794432
c = 50	0,61092318	0,617430102	0,581146428	0,502620806	0,385607812
LL Images	0,178076079	0,114190342	0,072755485	0,04674998	0,031034913

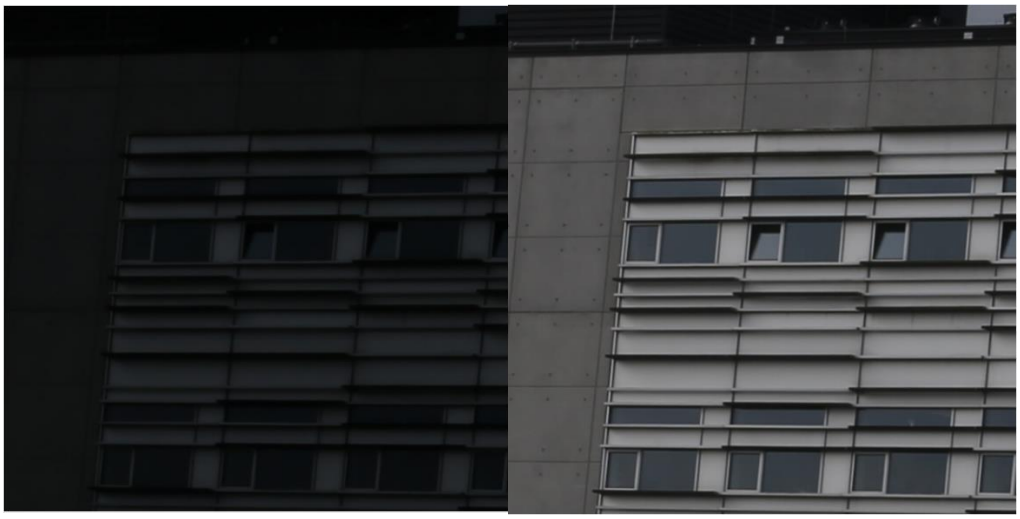
Figure 2.123: Experimental results average SSIM vs LL Images SSIM test set

Based on what we mentioned above, we notice that for a control constant of 1 the logarithmic transformation does not improve the original LL images at all, but on the contrary worsens their quality, which is reflected in the values of the quality metrics, which in all cases of darkness level are worse than the original ones before applying the method. For a control

constant of 10, the situation improves slightly, as for small darkness levels (3.0 and 3.5) the values of the quality metrics become better, but for the rest of the darkness levels the results have little difference from the initial values for the LL images. This is due to a combination of 2 factors, the first of which is that for small values of the correction constant the method does not work, and the second is that the images become very dark, making it even more difficult to retrieve visual information. These two factors lead to the low-quality experimental results of control constants 1 and 10. In the case where we use control constant 50, the best results are obtained, compared to the other 2 cases, with the metric values improving significantly. Specifically, the average PSNR improves by 8.96dB and the average SSIM increases by 0.43. Based on these we can conclude that the logarithmic transformation works better for large values of the control constant, although we should be careful with the values we choose, to avoid the phenomenon of over-amplification of the dark areas, which appeared during the transformation Gamma.

To confirm the above conclusions, we will show a random image from each darkness level, for each value of the control constant, together with the corresponding LL and ground truth, accompanied by the respective histograms.

Darkness Level: 3.0



Original Low Light **Normal Light**

c = 1 **c = 10** **c = 50**

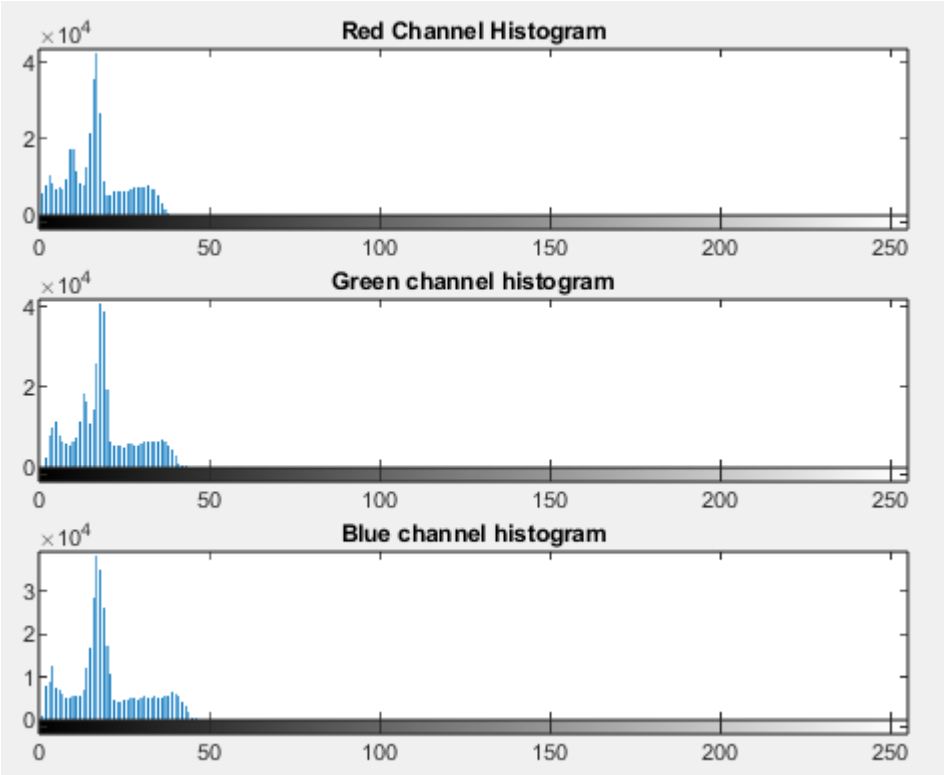


Figure 2.124: Histogram of 3.0 LL image

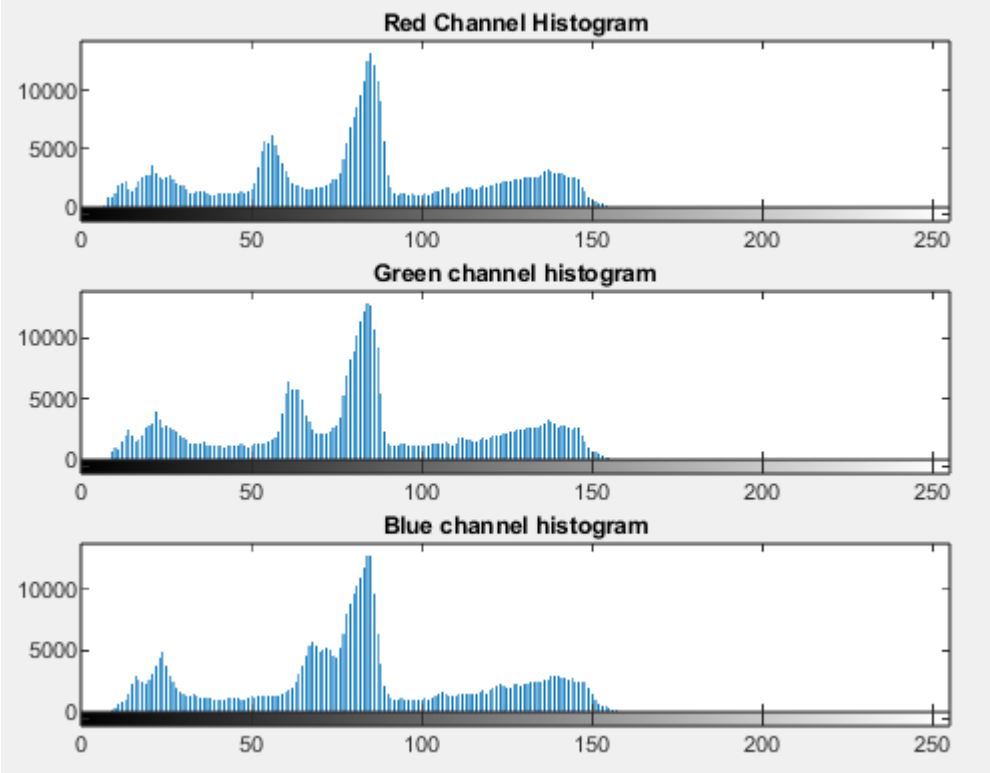


Figure 2.125: Histogram of 3.0 NL image

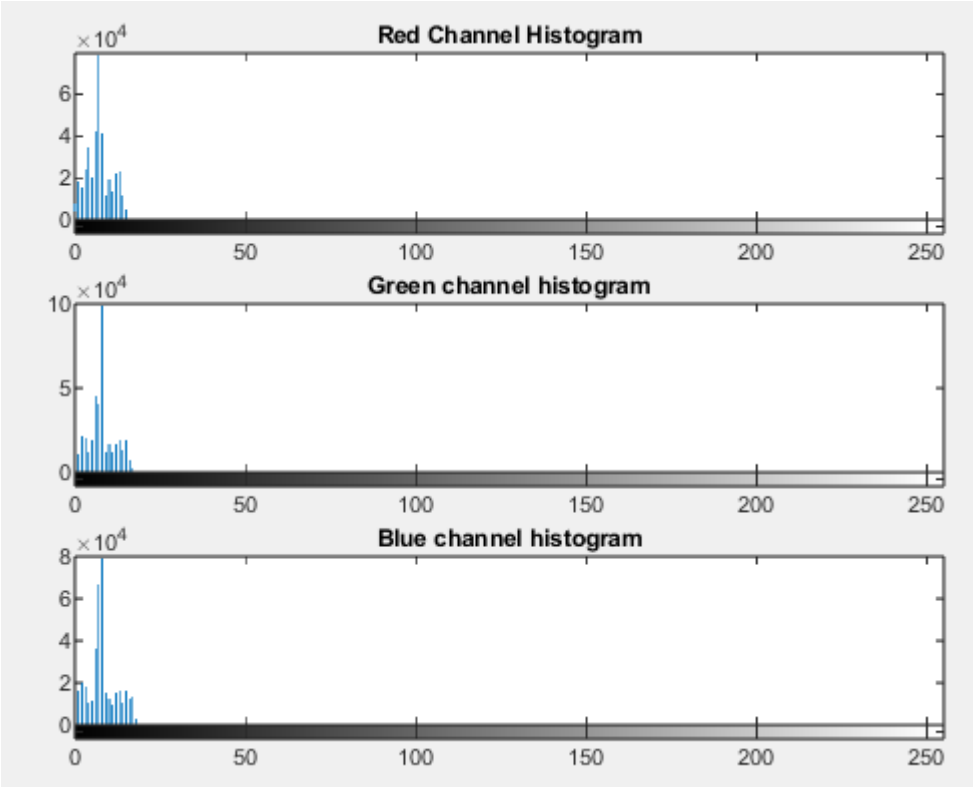


Figure 2.126: Histogram of 3.0 Experimental result image with c=1

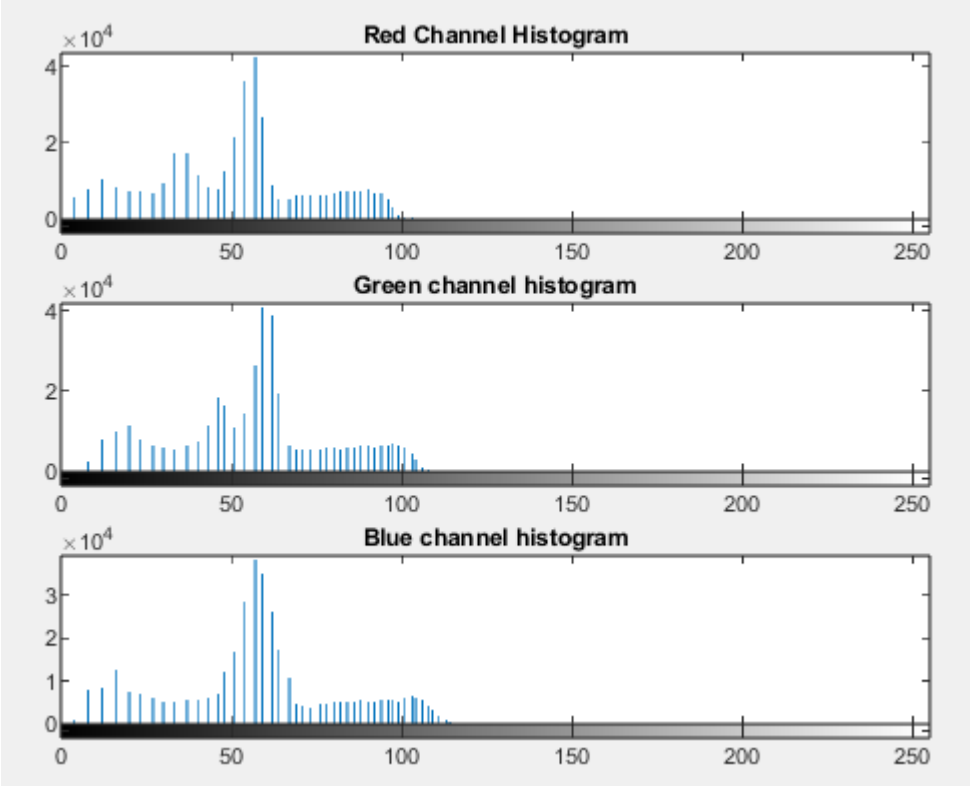


Figure 2.127: Histogram of 3.0 Experimental result image with c=10

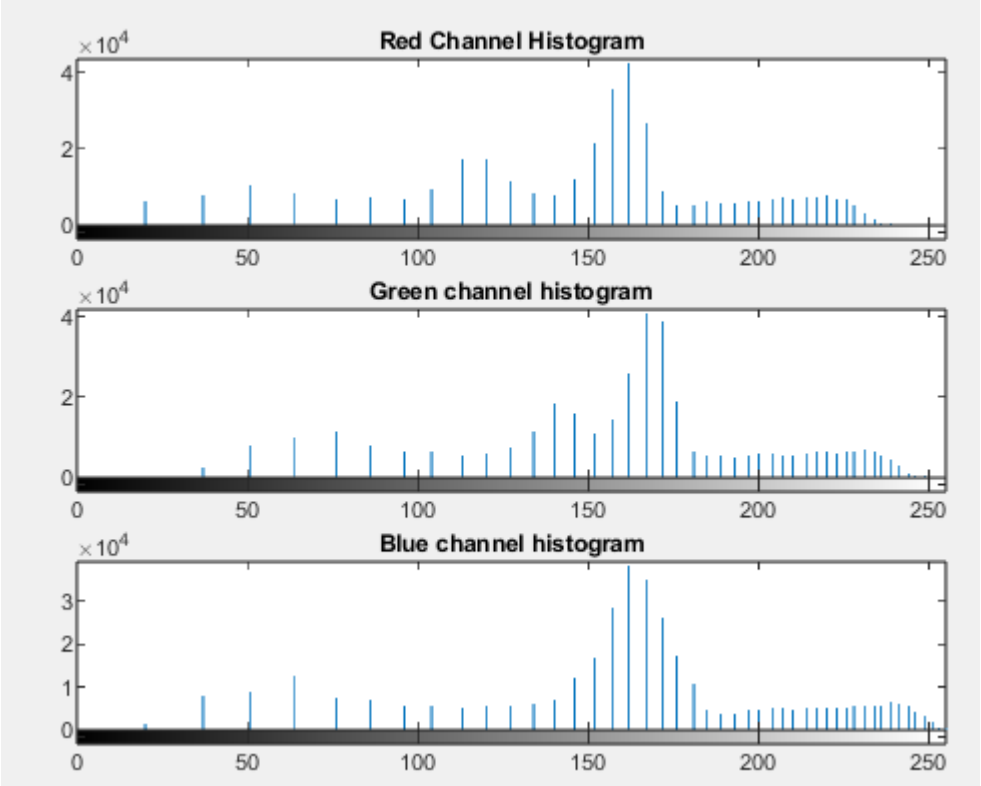


Figure 2.128: Histogram of 3.0 Experimental result image with c=50

For darkness level 3.0 we can observe the behavior we mentioned above. For a control constant of 1 the LL image does not improve at all, with the image remaining dark. This can also be seen from the histograms, where in figure 2.126 we see that in the experimental histogram the pixel values accumulate in its left part, with this accumulation being more intense compared to the histogram of LL image, confirming that the result is worse than the original pictures. For a control constant of 10 there is a slight improvement in the image, with dark areas enhanced and details now discernible. The histogram, as we see from figure 2.127, has started to be distributed over a larger range of values, with its form being similar to that of the ground truth case. Nevertheless, the result can be further improved, which is done in the case of the control constant 50. For this value of the control constant the best result is obtained, the dark regions are fully enhanced and most of the visual information is recovered. Furthermore, as we can see from figure 2.128, the histogram has spread over the entire available value range, with its form being very close to that of the ground truth case. Finally, let us comment that no color distortions are observed, like those we saw in the case of the gamma transformation.

Darkness Level: 3.5

Original Low Light **Normal Light**

c = 1 **c = 10** **c = 50**

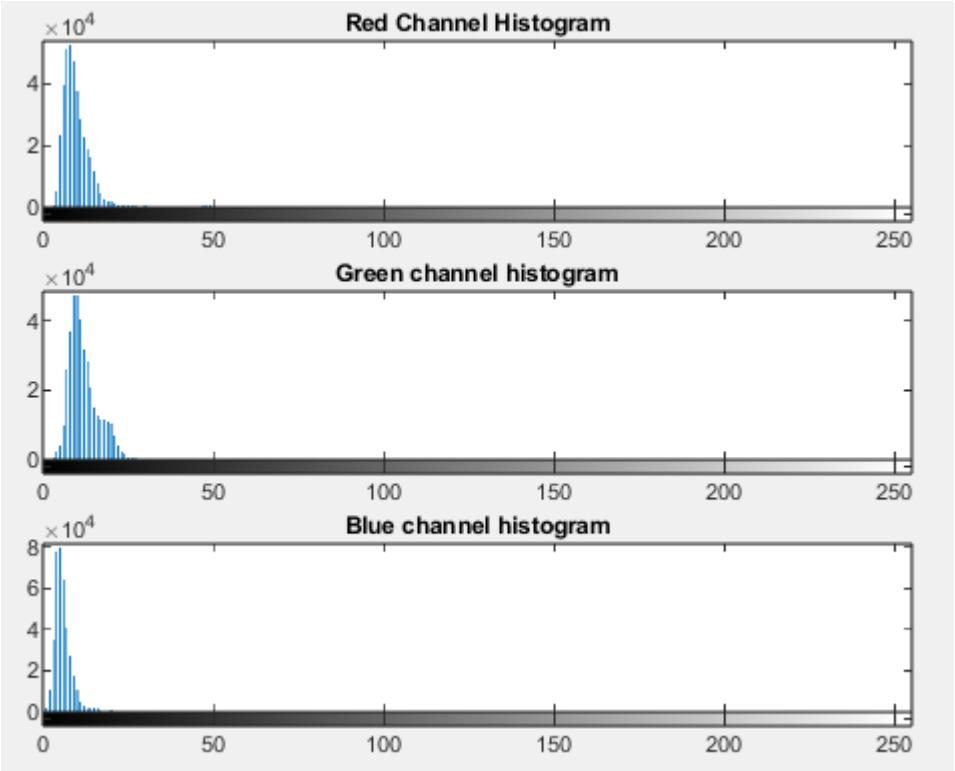


Figure 2.129: Histogram of 3.5 LL image

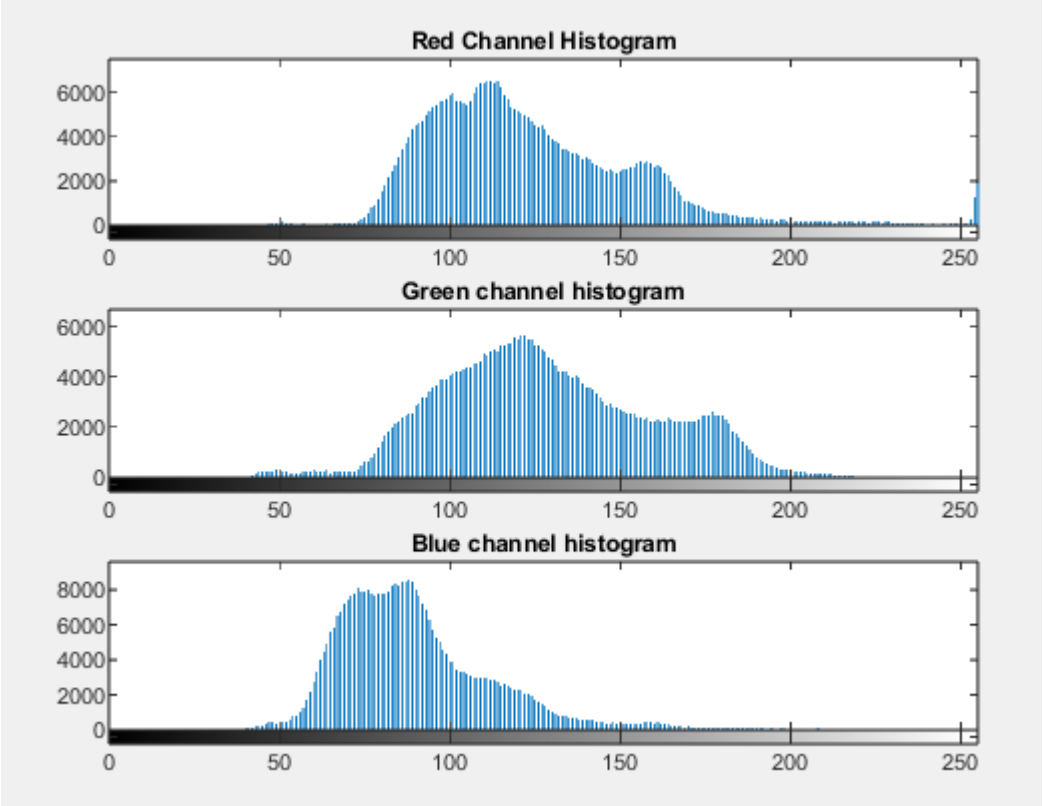


Figure 2.130: Histogram of 3.5 NL image

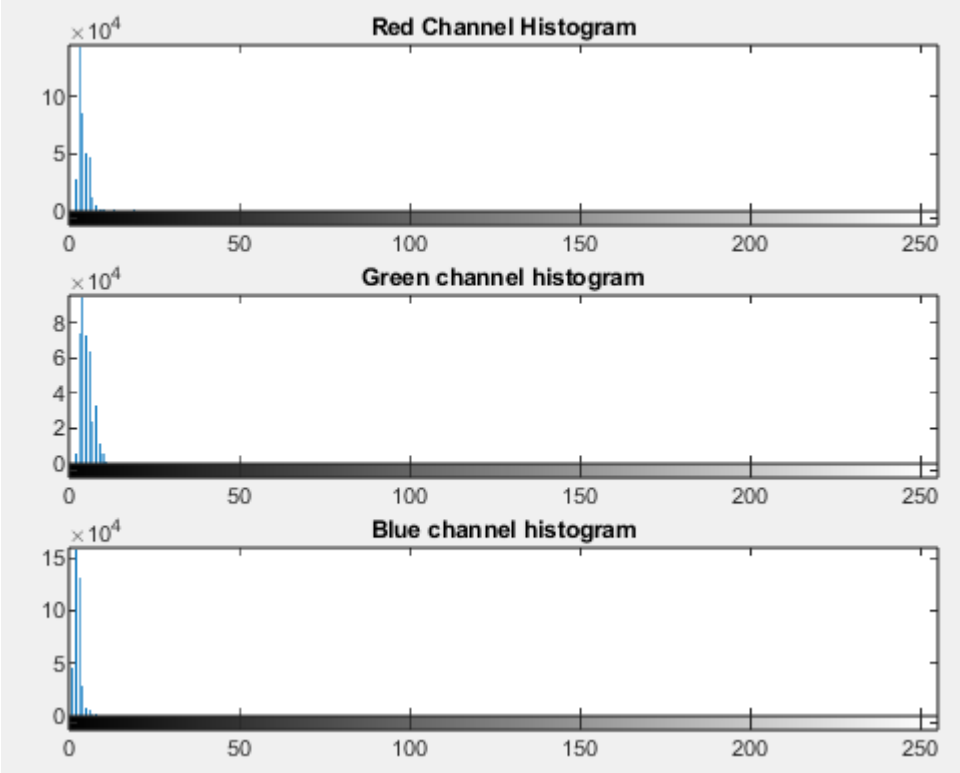


Figure 2.131: Histogram of 3.5 Experimental result image with c=1

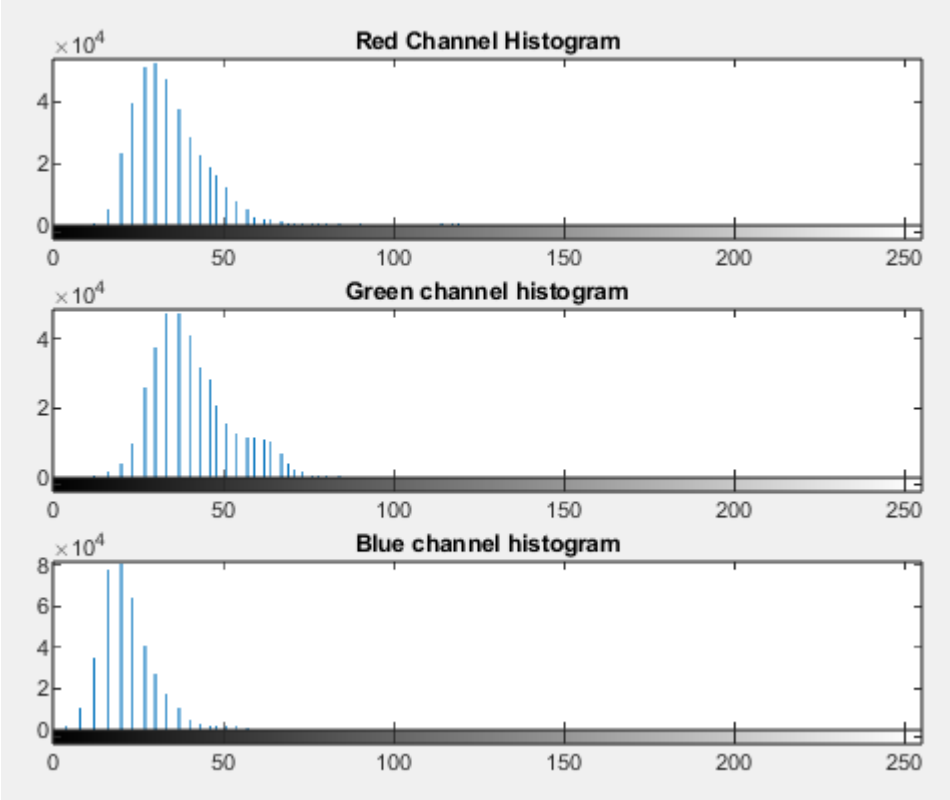


Figure 2.132: Histogram of 3.5 Experimental result image with c=10

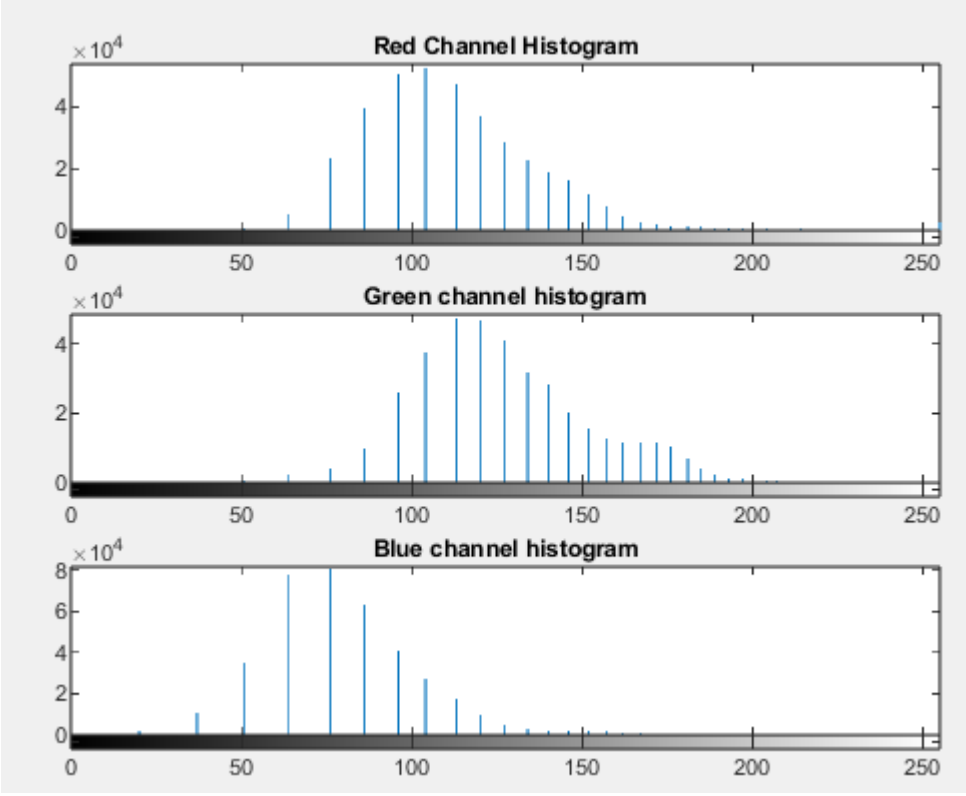


Figure 2.133: Histogram of 3.5 Experimental result image with c=50

In the case of darkness level 3.5, the same behavior is observed again. For control constant 1 the experimental result is worse than the original LL image, which can also be confirmed by the histogram. In figure 2.131, with the histogram of the experimental result, we see that the pixel values are more strongly accumulated in the left part, compared to the LL image, that is, the experimental result is darker than the original LL image, and of worse quality. For a control constant of 10, the result improves slightly, as we can now distinguish details, with the histogram spread over a larger range of values. The result improves even more in the case where we use a control constant of 50, with the dark areas fully enhanced and a large percentage of the visual information recovered. Moreover, from figure 2.133 we see that the experimental histogram has a form that is very close to the ground truth case, confirming the improvement of the result.

Darkness Level: 4.0

Original Low Light **Normal Light**

c = 1 **c = 10** **c = 50**

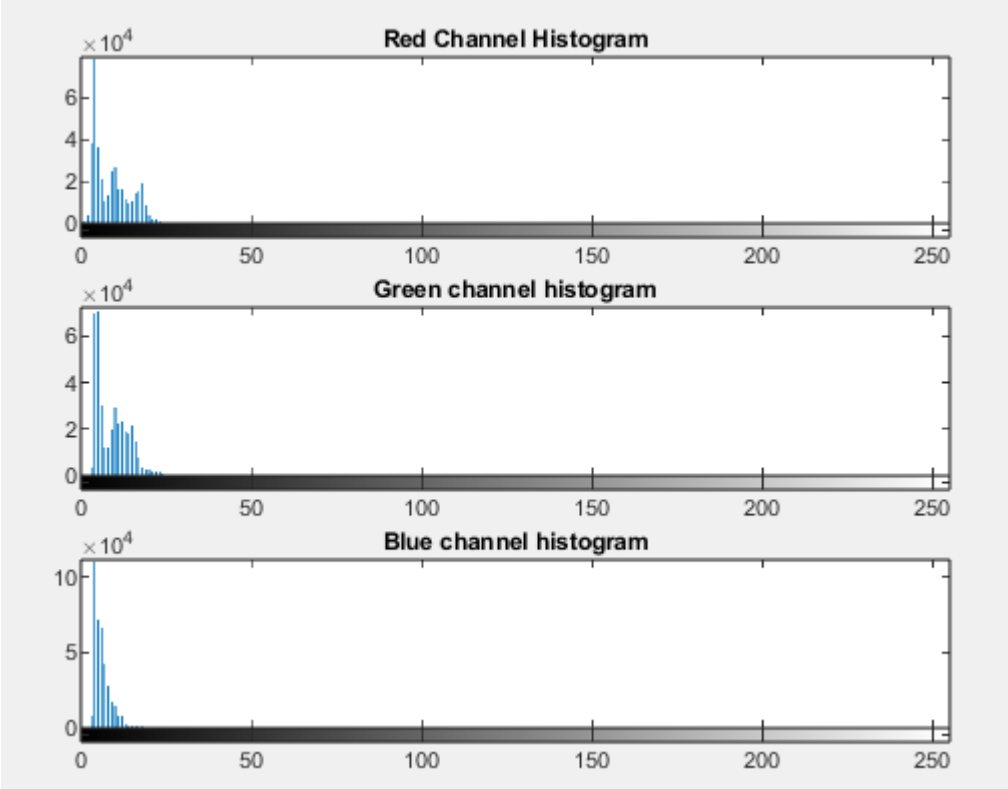


Figure 2.134: Histogram of 4.0 LL image

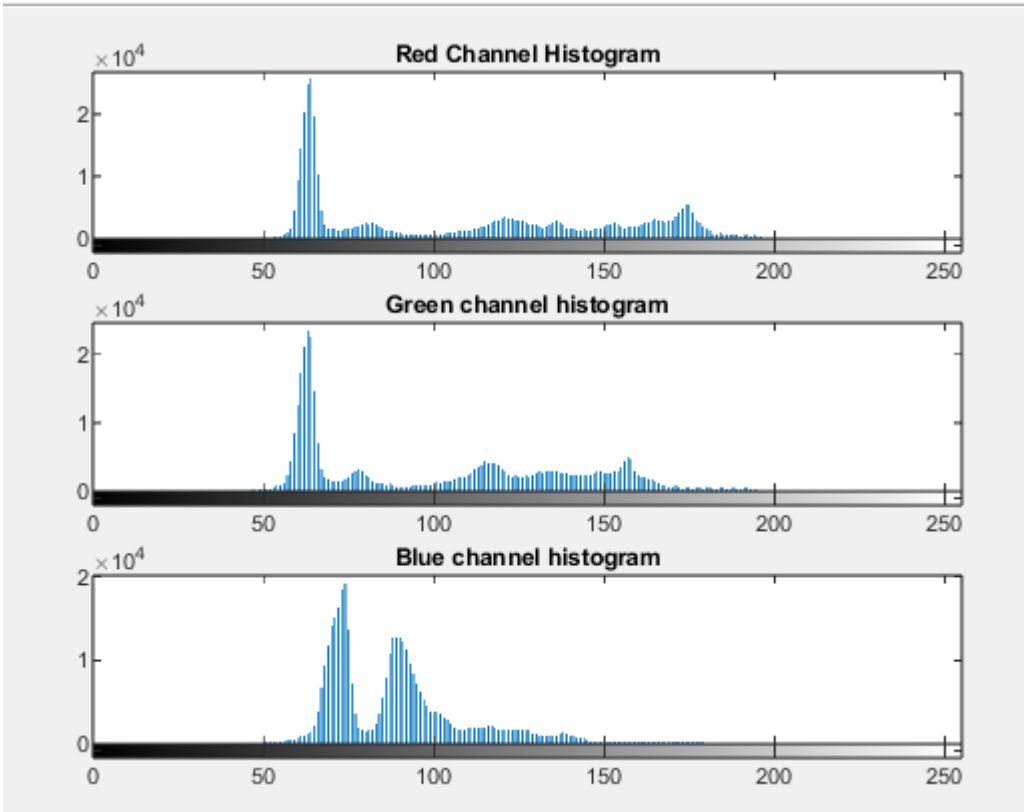


Figure 2.135: Histogram of 4.0 NL image

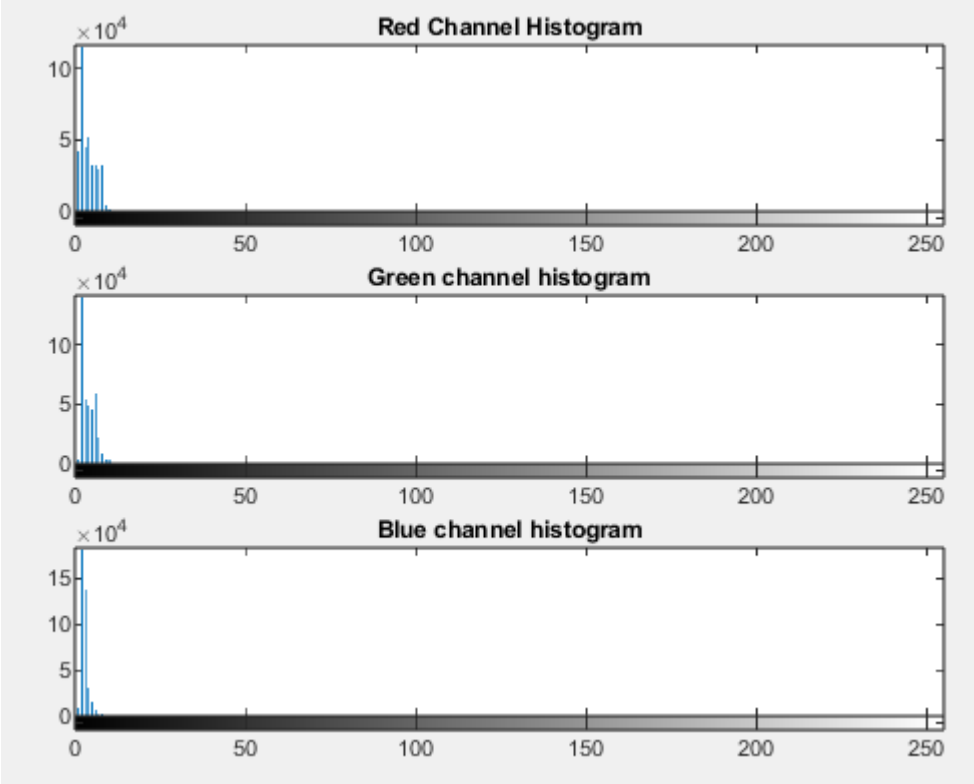


Figure 2.136: Histogram of 4.0 Experimental result image with c=1

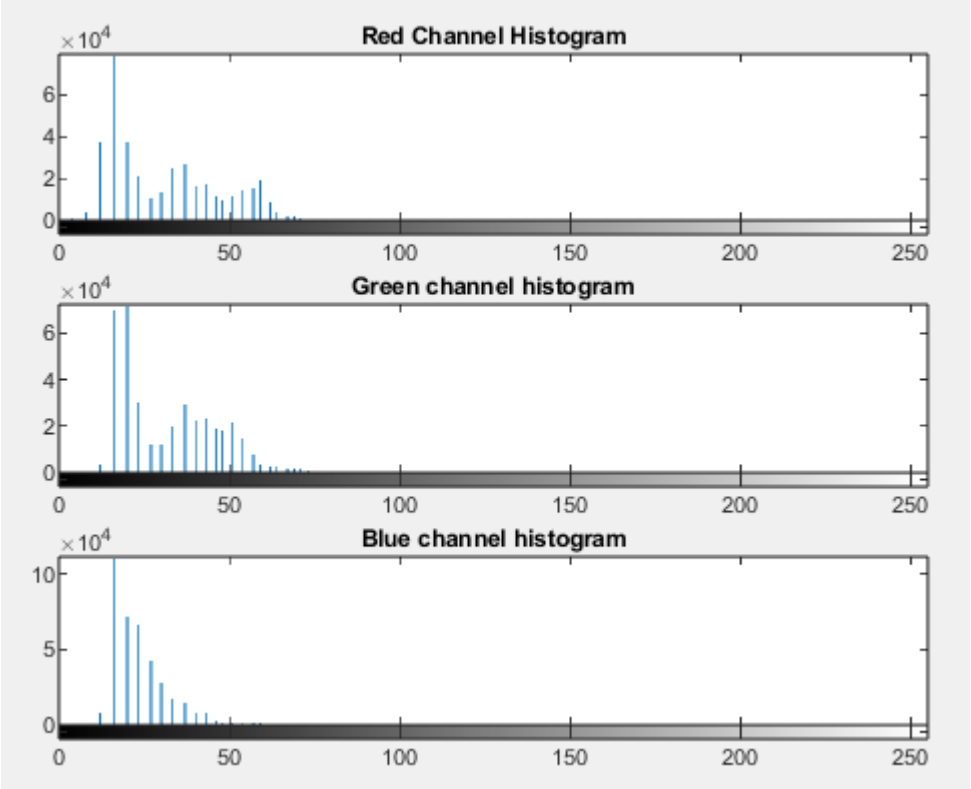


Figure 2.137: Histogram of 4.0 Experimental result image with c=10

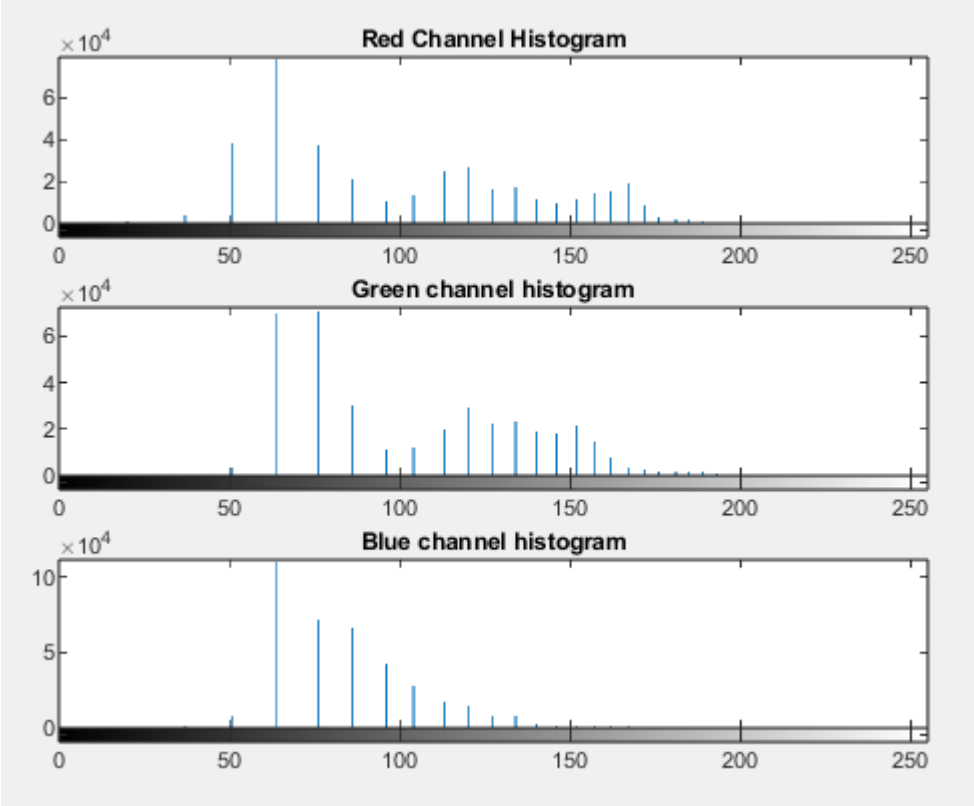


Figure 2.138: Histogram of 4.0 Experimental result image with c=50

In the case of darkness level 4.0 we can observe the expected behavior. For a control constant of 1 the result, instead of improving, becomes worse, with the image remaining dark. This is also confirmed by the experimental histogram, where from figure 2.136 we see that the pixel values accumulate in the left part, much more strongly than in relation to the LL image. For a control constant of 10, the result improves little, as the image remains dark, although we can make out quite a bit of detail. Finally, for a control constant of 50 we recover most of the visual information, with dark areas fully enhanced. From figure 2.138 we see that the histogram is spread over a larger range of values, with its form close to that of the ground truth case.

Darkness Level: 4.5

Original Low Light **Normal Light**

c = 1 **c = 10** **c = 50**

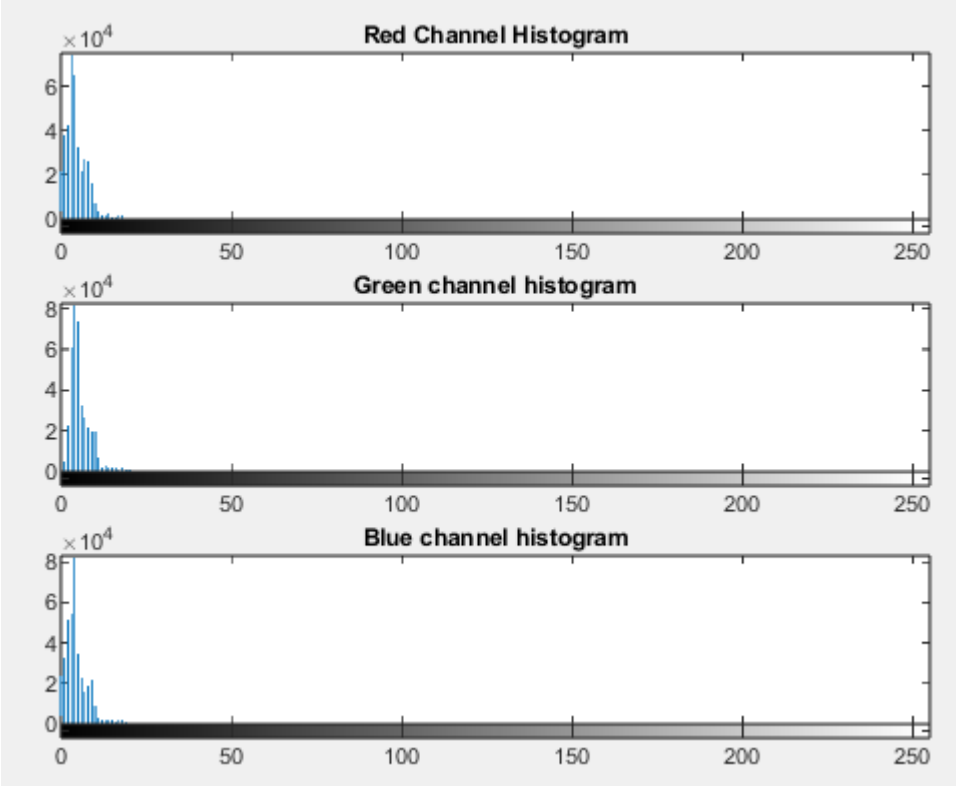


Figure 2.139: Histogram of 4.5 LL image

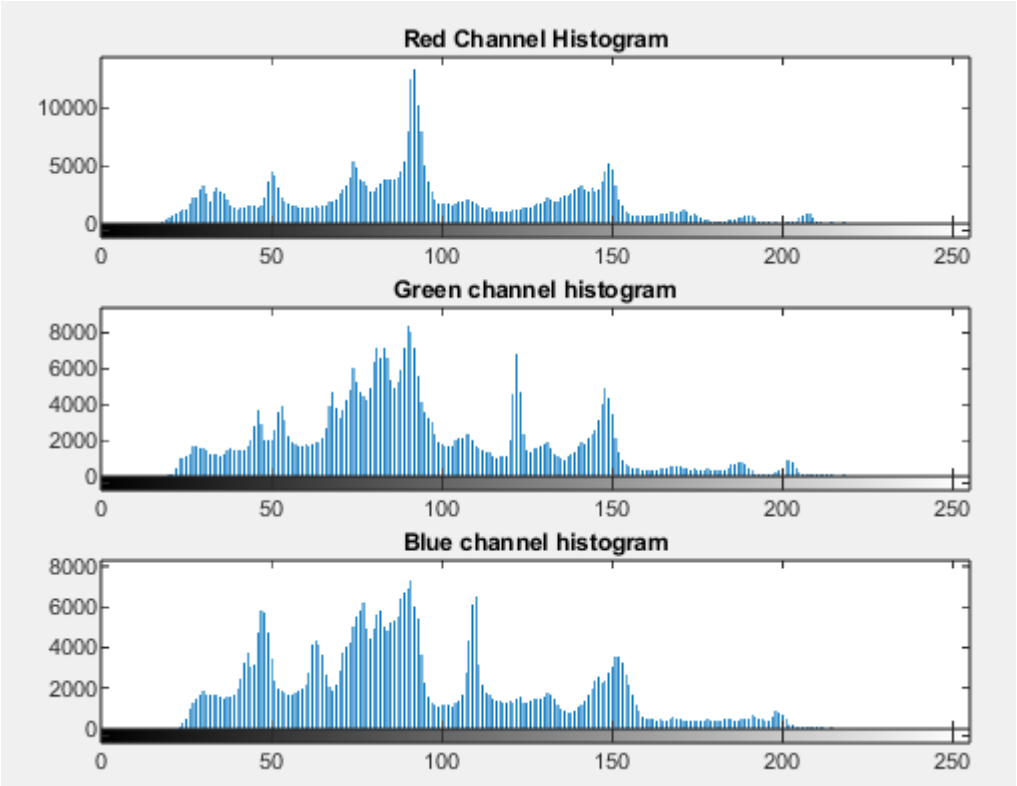


Figure 2.140: Histogram of 4.5 NL image

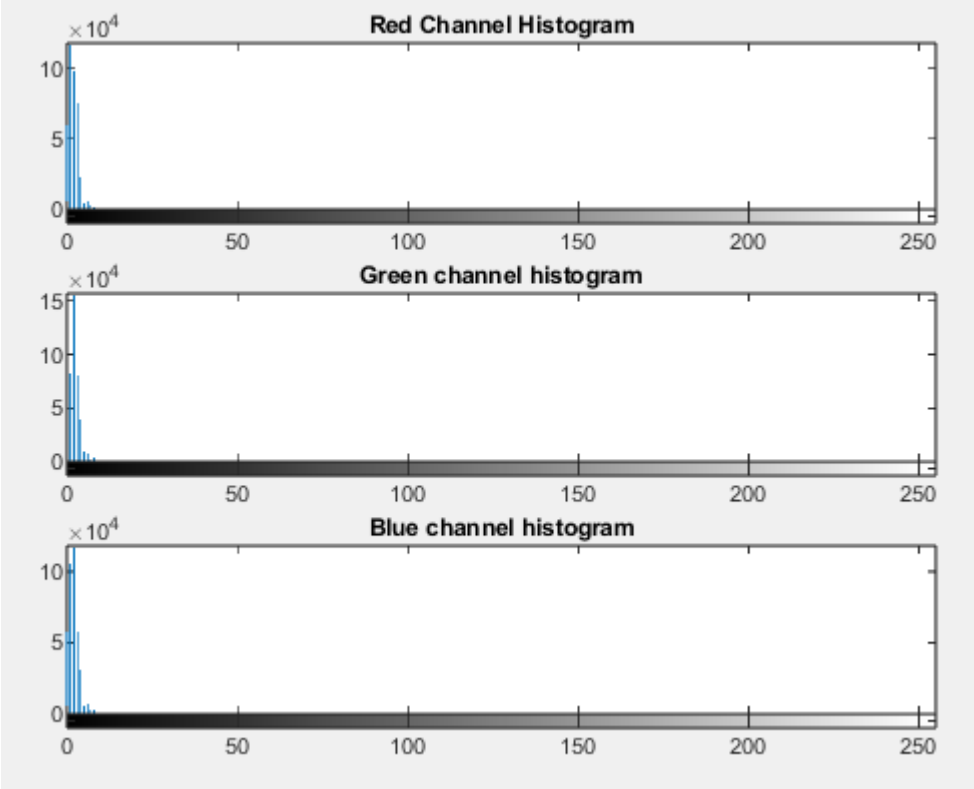


Figure 2.141: Histogram of 4.5 Experimental result image with c=1

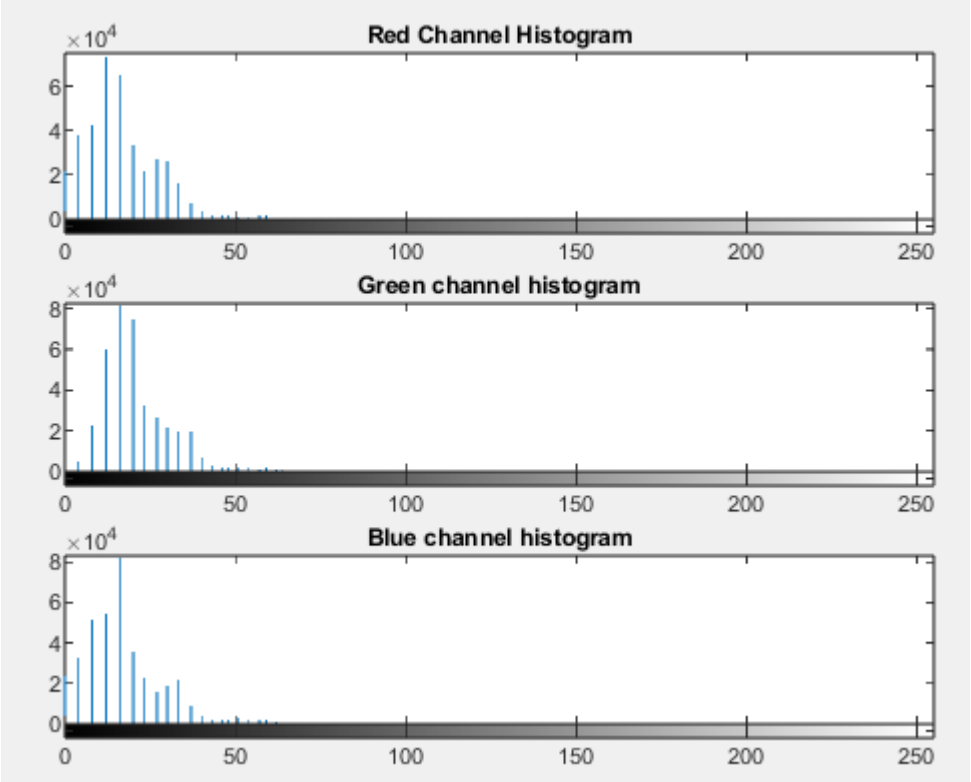


Figure 2.142: Histogram of 4.5 Experimental result image with c=10

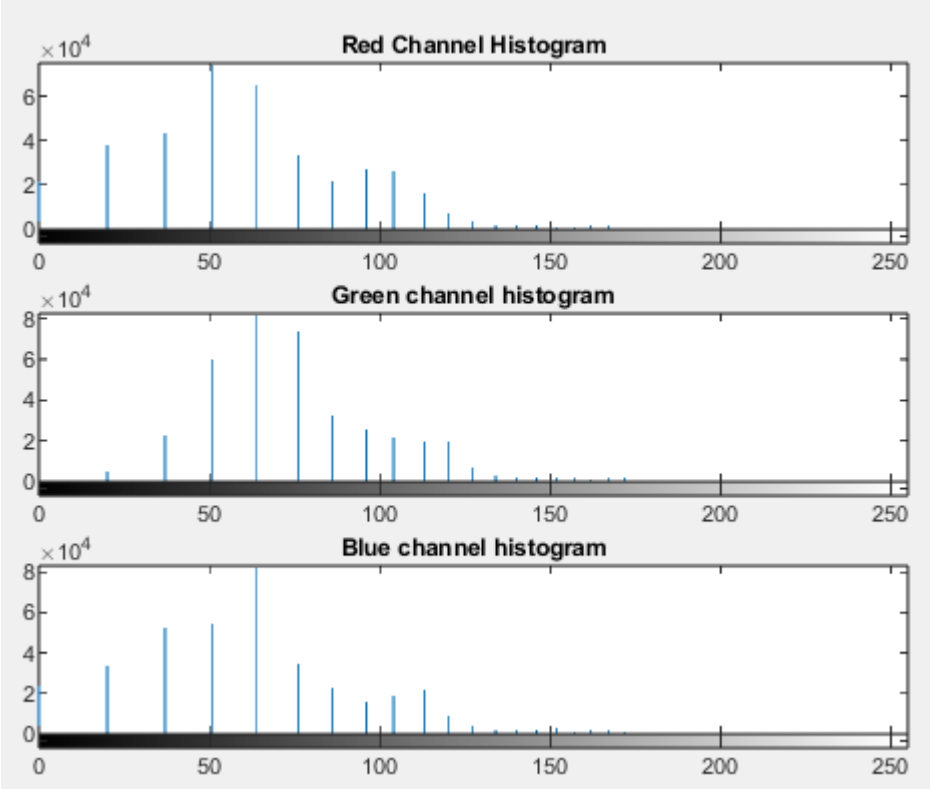


Figure 2.143: Histogram of 4.5 Experimental result image with c=50

For a control constant of 1 we observe that the result does not improve at all, but instead the image becomes darker and of lower quality. This is also confirmed by the histogram in figure 2.141, from where we see that the pixel values are all clustered in the left part of it, more strongly than in relation to the LL image. For a control constant of 10 the result improves little, since the image remains dark, and the pixel values remain concentrated in the left part of the histogram. In the case where we use a control constant of 50, the best result is obtained compared to the other 2 cases. Most of the visual information has been recovered, and the pixel values have been spread over a wider range of values. What we can observe, however, is the appearance of color distortions, which are due to the fact that the images have become too dark, making it difficult to retrieve complete and correct color information.

Darkness Level: 5.0

Original Low Light **Normal Light**

c = 1 **c = 10** **c = 50**

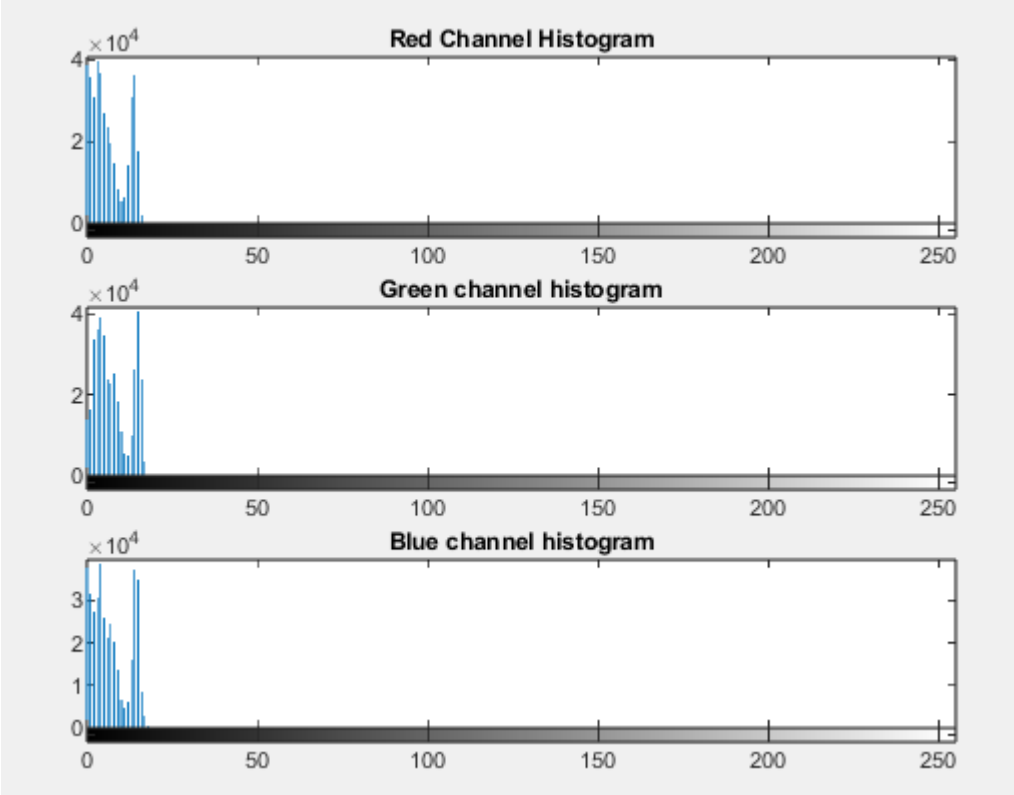


Figure 2.144: Histogram of 5.0 LL image

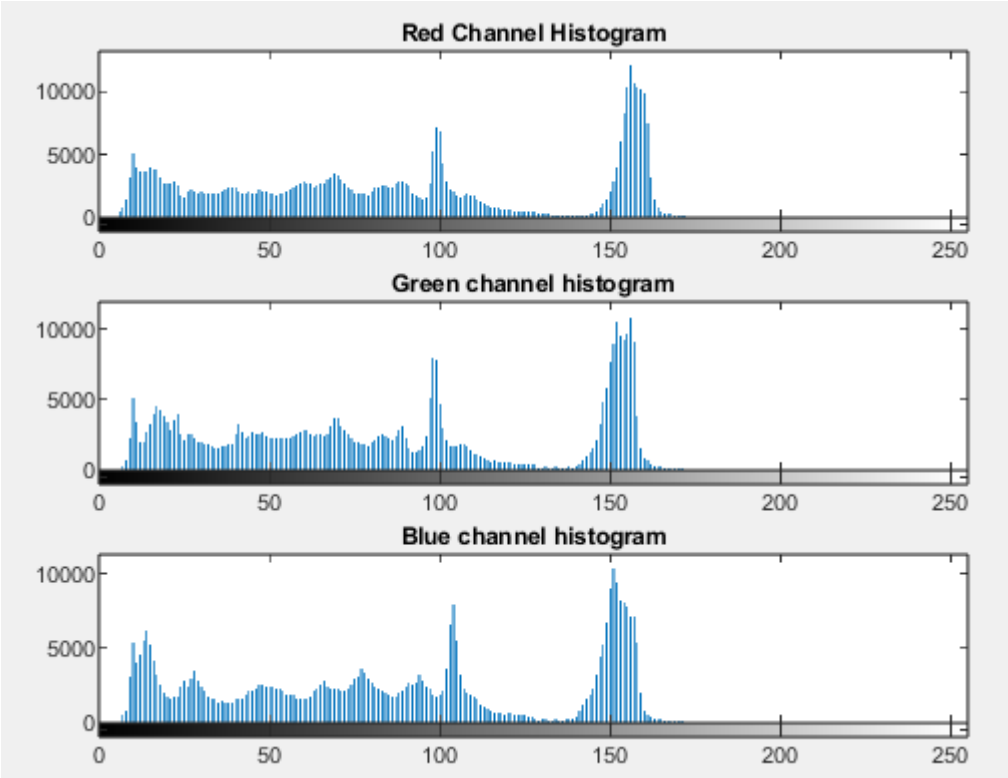


Figure 2.145: Histogram of 5.0 NL image

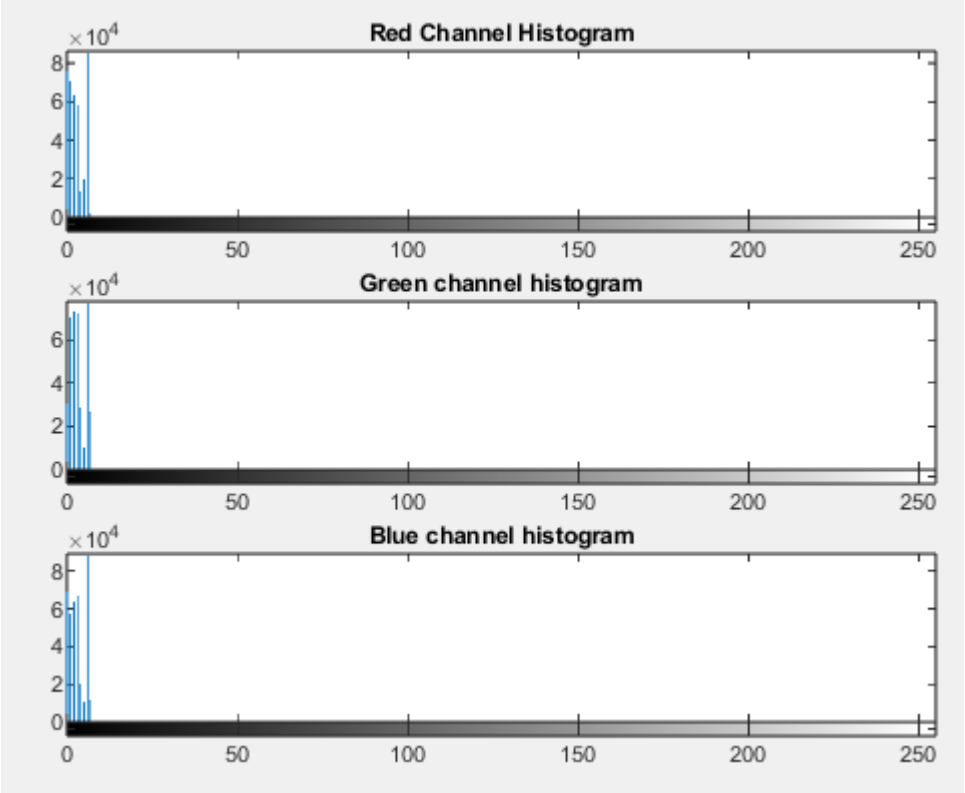


Figure 2.146: Histogram of 5.0 Experimental result image with c=1

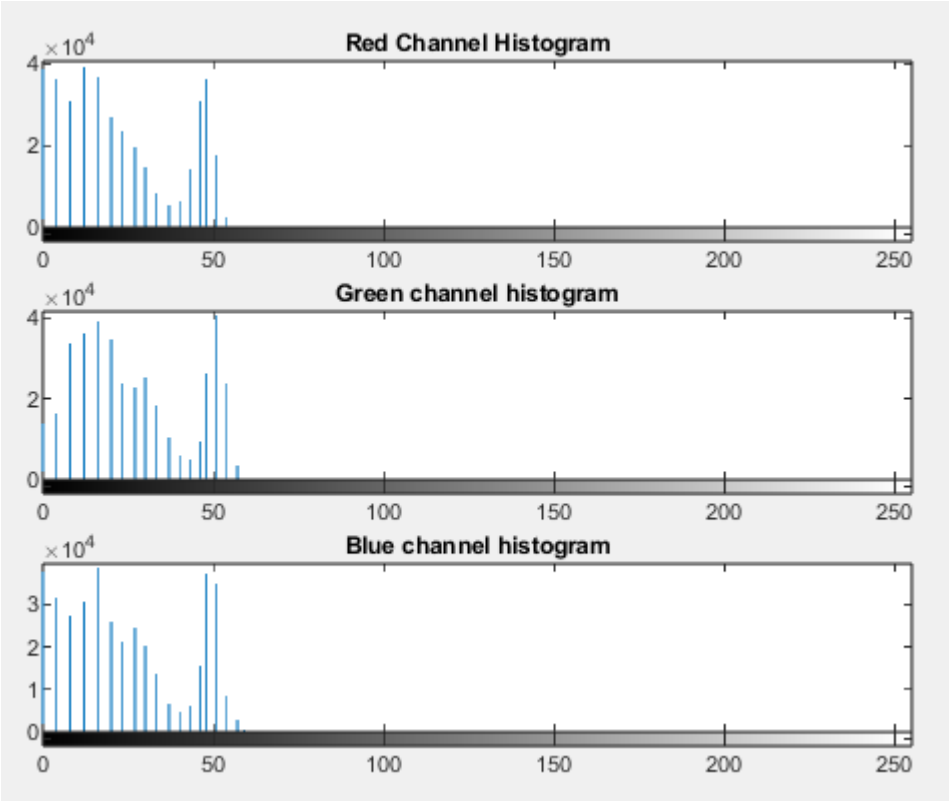


Figure 2.147: Histogram of 5.0 Experimental result image with c=10

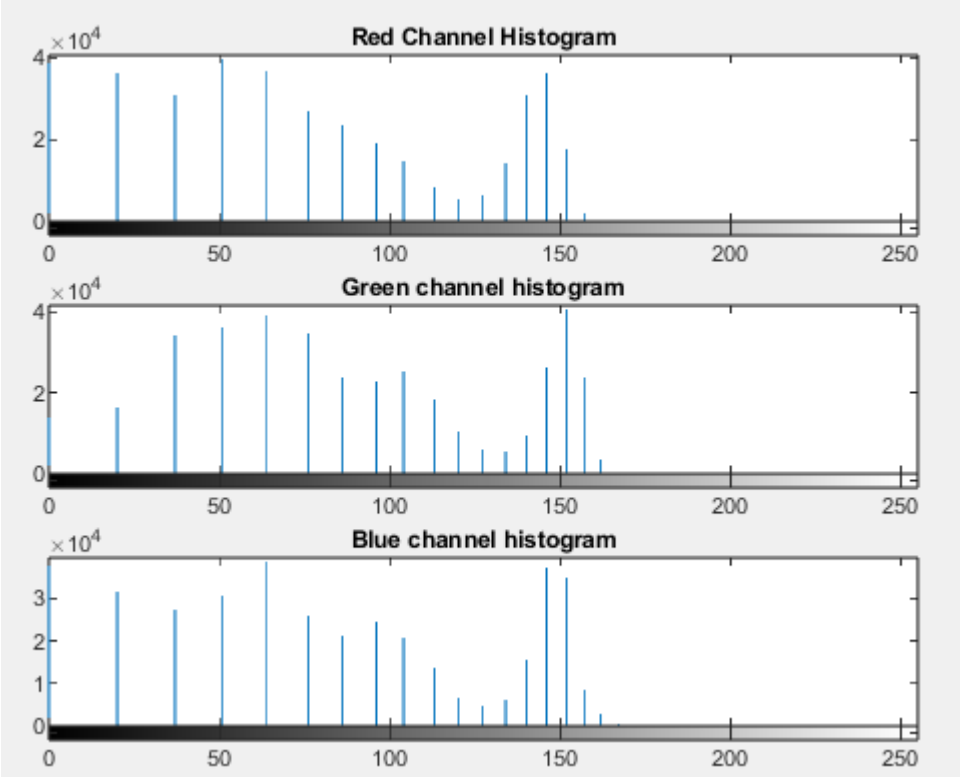


Figure 2.147: Histogram of 5.0 Experimental result image with c=50

For darkness level 5.0 and control constant 1 again the result does not improve at all, but becomes worse, as the image becomes darker and the pixel values are concentrated in the left part of the histogram, more strongly than in the original LL image. For a control constant of 10, again there is little improvement as the pixel values remain clustered on the left side of the histogram, as seen in Figure 2.147, with the image remaining dark having recovered little visual information. For a control constant of 50 again the best result is obtained, compared to the other two cases, as we have recovered a large part of the visual information. Furthermore, from the histogram in Figure 2.147 we can see that the pixel values are spread over a larger range of values, with the histogram shape resembling that of the ground truth case. Finally, we have to comment that color distortions appear here as well, due to the fact that the images are too dark to fully recover the color information.

At this point it is worth briefly commenting on all the above results of the logarithmic transformation. For a control constant of 1 we observed that the LL images do not improve at all, but instead become worse. Images become darker, with pixel values clustering more heavily on the left side of the histogram. For a control constant of 10, the result is slightly improved, which is confirmed by the improvement of the quality metrics

as well as the recovery of a small part of the visual information. However, as the darkness level increases, the performance of the method degenerates and the results for a control constant of 10 become comparable to those for a control constant of 1. In the case where we use a control constant of 50, the best results are obtained, and the algorithm manages to recover most of the visual information. This is also reflected in the values of the quality metrics, which improve noticeably, as well as in the histograms, which in any case have a form that resembles the ground truth case. However, at high darkness levels (4.5, 5.0) color distortions begin to be observed, as well as a decrease in the performance of the algorithm. This is because the images have become too dark, with very little to zero dynamic range, making it very difficult to fully and correctly recover color information.

2.4 Histogram Equalization

In this section we will apply a pointwise transformation called histogram equalization, the logic of which is based on the fact that if the pixel values are evenly distributed over all possible gray levels, the image will have high contrast and high dynamic range [15]. The goal is to find a suitable mathematical function that takes as input a low-contrast image and outputs an image whose histogram will approximate the normal distribution. In this way we will manage to enhance the dark areas of each image, so histogram equalization is suitable for LLIE.

Histogram equalization uses two basic concepts of image processing, the probability density function (pdf) and the cumulative distribution function (cdf). If we consider I(i,j) as the image under consideration, with N being the total number of pixels and L the number of gray levels, then the pdf is defined as:

$$p(k) = \frac{n_k}{N}$$

Where n_k is the number of pixels with gray value k (k=0,1,…,L-1). The pdf expresses the percentage of pixels with gray value k, or otherwise the probability that a pixel has gray value k. Obviously, by definition, it does:

$$\sum_{k=1}^{L-1} p(k) = 1$$

Accordingly, the cdf is defined as follows:

$$P(m) = \sum_{k=0}^{m} p(k) = \frac{1}{N} \sum_{k=0}^{m} n_k$$

and expresses the percentage of pixels in the image, which have a gray value less than or equal to m, or otherwise the probability that a pixel has a gray value less than or equal to m.

Based on these, the transformation that we will apply to each pixel of the image is derived, which is:

$$I_{NL}(k) = [(L-1)P(k)]$$

This is an implementationally simple transformation that is implemented with the function shown in Figure B.2.13 of Appendix B. We will apply this function to all images in the dataset, which is done using the scripts shown in figures B.2.14 and B.2.15, of appendix B. After the application of these scripts, we have at our disposal the experimental results, per darkness level and per set (training, validation, test). Based on these we will calculate the evaluation metrics in order to evaluate the performance of the method. The calculation is done with the script presented in figure B.2.16 of appendix B. After its application, 3 excel files are generated (one for each set) that contain the values of the metrics per image and darkness level, and we can use them for the construction summary tables, which will contain the minimum/maximum/average value of each metric, per darkness level. In addition, we will also construct line charts with the minimum/maximum/average value of PSNR and SSIM per darkness level, to see how the performance of the method is affected as the darkness level increases. Finally, we will also present line charts with the average value of PSNR and SSIM, before and after applying the method, to see how much the quality metrics of then LL images are improved. First we will present

these results, and then, we will display the images corresponding to the minimum and maximum experimental PSNR, accompanied by the corresponding LL and ground truth cases as well as the related histograms, in order to visually comment on the result.

Training Set

MSE					
Level	3.0	3.5	4.0	4.5	5.0
MIN	421,0046	441,9709	517,6390989	590,5857	660,5967
MAX	12408,14	12359,92	12639,4027	13519,73	14621,51
AVERAGE	3071,221	3145,256	3262,668015	3468,21	3659,15
	PSNR(dB)				
Level	3.0	3.5	4.0	4.5	5.0
MIN	7,193738281	7,210649	7,113538	6,821124	6,480882
MAX	21,88793497	21,67687	20,99053	20,41797	19,93144
AVERAGE	14,01144036	13,89799	13,72839	13,4587	13,17517
	SSIM				
Level	3.0	3.5	4.0	4.5	5.0
MIN	-0,02341408	-0,04635	-0,06808	-0,15138	-0,15752
MAX	0,787580525	0,764571	0,761787	0,736809	0,725542
AVERAGE	0,400328107	0,364868	0,3249	0,280182	0,230345
	MV				
Level	3.0	3.5	4.0	4.5	5.0
MIN	127,0685056	126,9022	126,7785	127,0456	127,0612
MAX	142,1202654	151,1714	154,5217	157,6636	161,1524
AVERAGE	127,9949542	128,5551	129,5186	131,3005	133,734
	STD				
Level	3.0	3.5	4.0	4.5	5.0
MIN	158,809322	178,1034	208,1213	227,0157	329,4894
MAX	1396,637581	1395,106	1398,889	1377,956	1357,603
AVERAGE	870,9182373	864,2766	853,7109	835,8393	812,0992
	BRISQUE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	3,716687142	3,154421	4,505201	6,952224	9,015325
MAX	52,79433739	54,91105	58,1652	58,19425	59,30204
AVERAGE	28,24526558	29,50091	31,61575	34,67411	38,37626
	NIQE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	2,46136462	2,623508	2,847146	3,009361	3,203958
MAX	24,53955714	12,42223	9,341568	10,73782	11,23613
AVERAGE	4,005421566	4,330952	4,710459	5,164884	5,671135

Table 2.25: HE quality metrics on training set

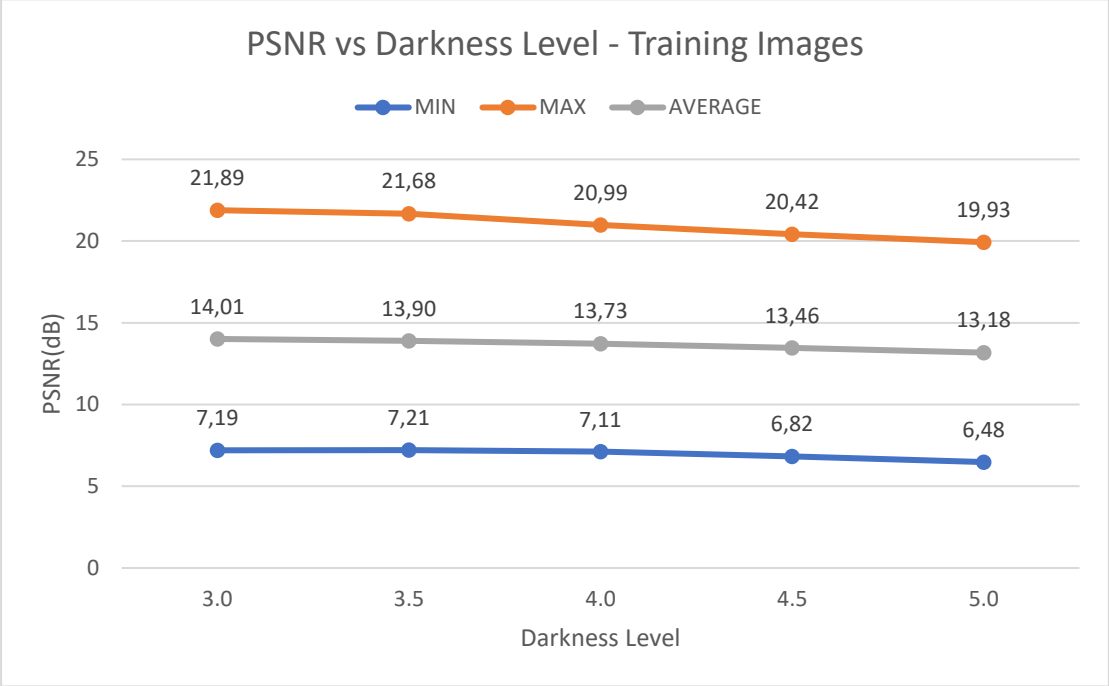


Figure 2.148: Experimental results PSNR vs Darkness level for training set

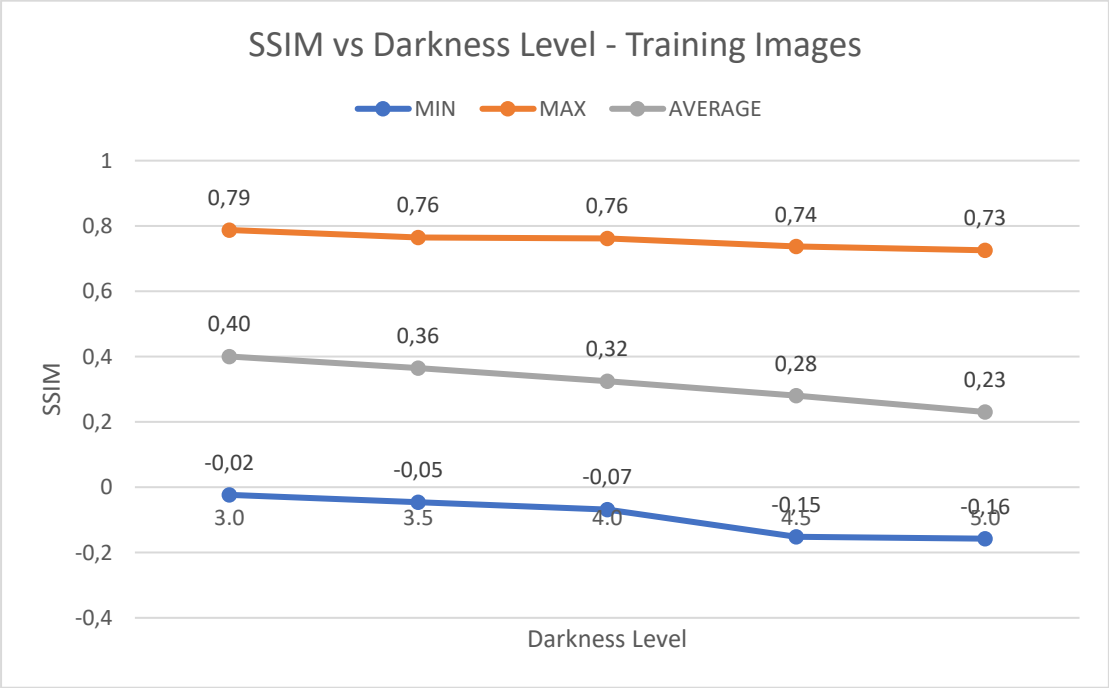


Figure 2.149: Experimental results SSIM vs Darkness level for training set

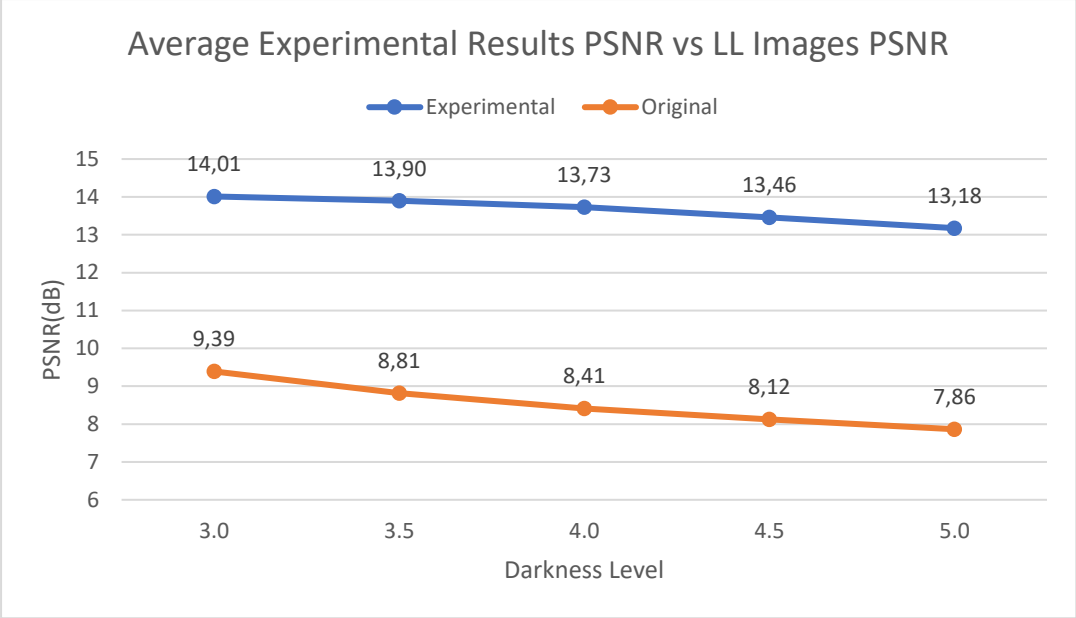


Figure 2.150: Experimental vs LL images PSNR

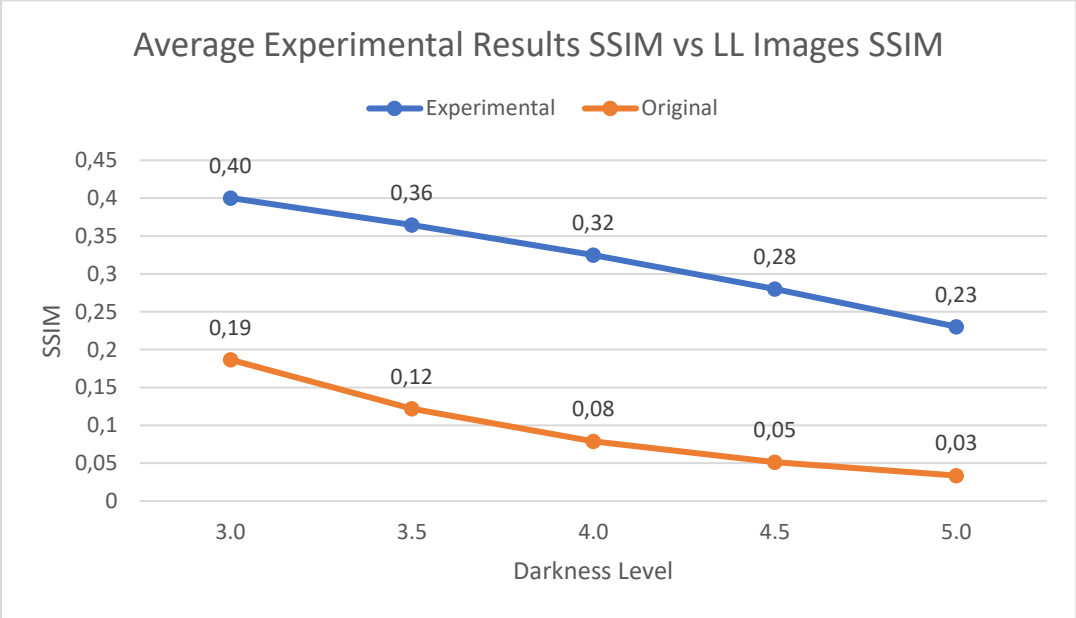


Figure 2.151: Experimental vs LL images SSIM

We notice that histogram equalization improves the quality metrics values significantly, with the average PSNR increasing by an average of 5.14dB, and the average SSIM by 0.22. Moreover, the expected behavior is observed with the increase of the darkness level, as we observe that the MSE increases, while the PSNR and SSIM decrease, which means that the performance of the algorithm decreases with the increase of the darkness level. At this point we have to comment that for PSNR and SSIM the rate

of decrease of their values with the increase of the darkness level is smaller compared to the methods we have seen so far, which means that it is more resistant to the change of the darkness level. Regarding the values of the quality metrics with no reference, we see that the MV increases, i.e. the images become brighter, and the STD is also larger, compared to the original images, which means that the pixel values are spread over a wider range of values around average. Finally, BRISQUE and NIQE increase with increasing darkness level, which means that we are moving away from natural statistics. This is to be expected as, as the darkness level increases, the images become very dark making it much more difficult to retrieve the full visual information.

Validation Set

MSE					
Level	3.0	3.5	4.0	4.5	5.0
MIN	439,5438	480,4772813	529,7034	584,3981	703,0401
MAX	12809,33	13628,13055	14841,78	18367,84	19703,56
AVERAGE	3009,335	3120,370316	3302,063	3579,684	3677,492
	PSNR(dB)				
Level	3.0	3.5	4.0	4.5	5.0
MIN	7,05554	6,786440756	6,415942	5,490224	5,185356
MAX	21,70078	21,31407503	20,89048	20,46372	19,661
AVERAGE	14,51815	14,3807657	14,16873	13,89425	13,83293
	SSIM				
Level	3.0	3.5	4.0	4.5	5.0
MIN	0,083985	0,070406396	0,055704	0,021274	0,021279
MAX	0,783691	0,750853004	0,716853	0,680148	0,610063
AVERAGE	0,415319	0,382625468	0,346071	0,302454	0,279778
	MV				
Level	3.0	3.5	4.0	4.5	5.0
MIN	127,1019	127,088425	126,8991	127,1712	127,7105
MAX	139,1523	144,9227998	151,3749	159,7386	167,1147
AVERAGE	128,538	129,2522622	130,2573	131,9691	133,7172
	STD				
Level	3.0	3.5	4.0	4.5	5.0
MIN	521,6171	479,0979246	410,3607	321,6802	507,0534
MAX	1345,086	1345,688009	1337,153	1310,609	1273,174
AVERAGE	881,7714	871,5300366	859,2557	836,429	827,838
	BRISQUE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	7,375161	6,534256878	10,97686	18,7654	14,85068
MAX	48,79476	49,62103107	54,99201	53,48019	55,02543
AVERAGE	30,19114	30,40059761	31,95058	34,43156	36,31509

	NIQE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	2,54717	3,12256287	3,39889	3,711442	4,055303
MAX	6,625679	7,575652513	9,891647	8,9151	9,910555
AVERAGE	3,908225	4,217832807	4,564158	5,074119	5,642139

Table 2.26: HE quality metrics on validation set

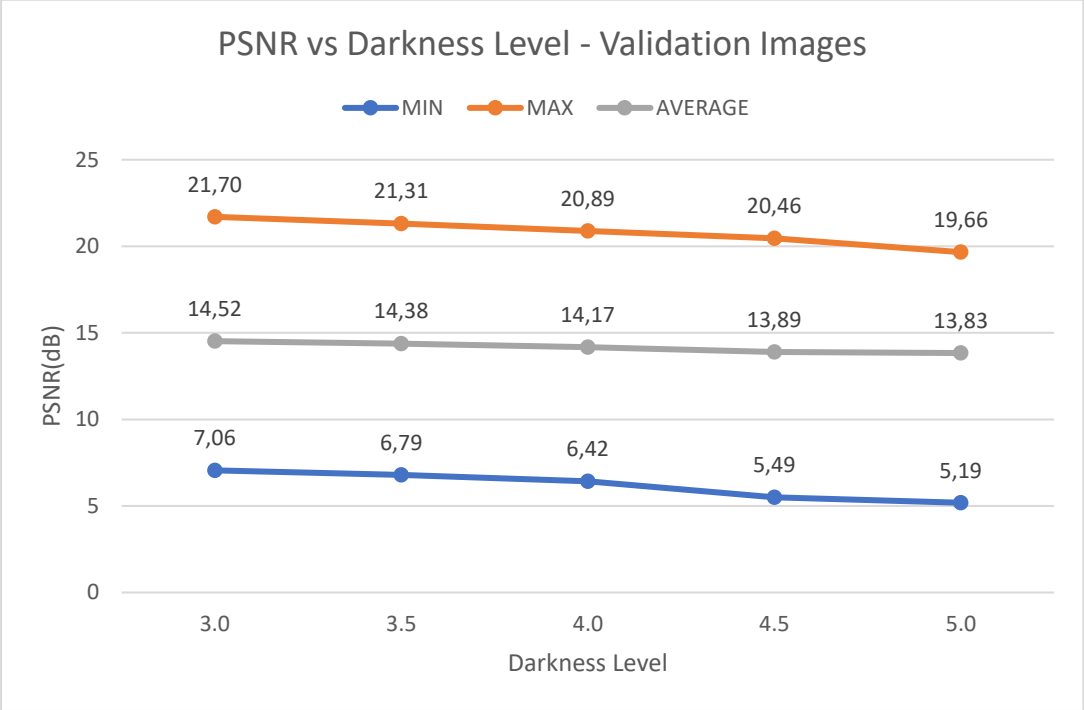


Figure 2.152: Experimental results PSNR vs Darkness level for validation set

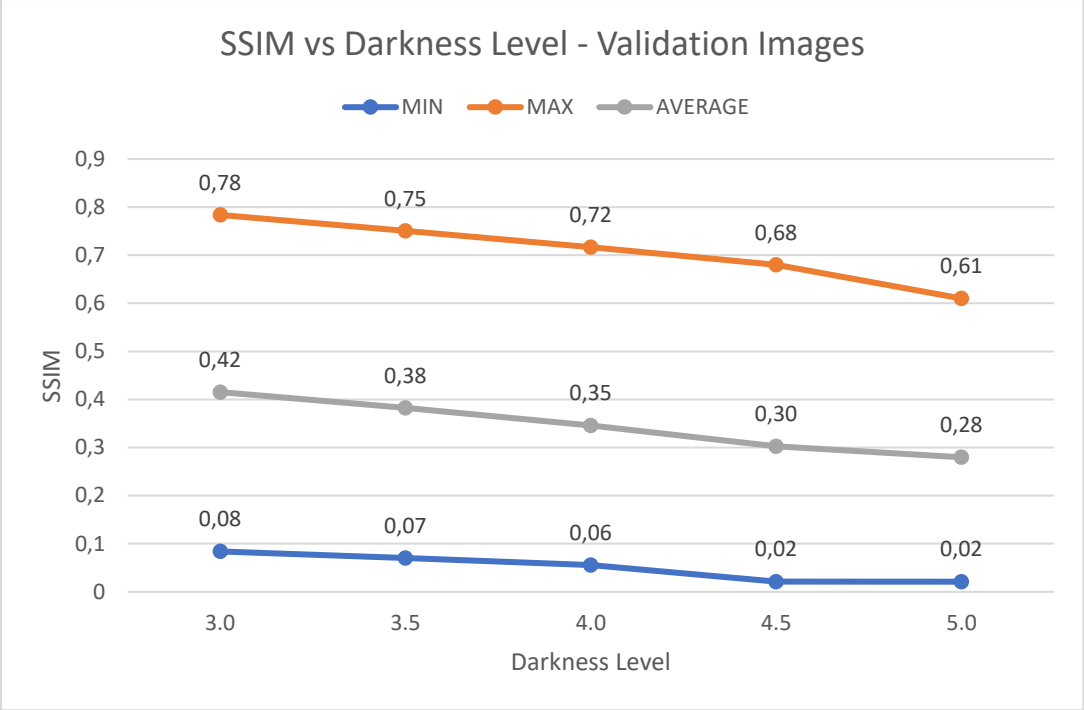


Figure 2.153: Experimental results SSIM vs Darkness level for validation set

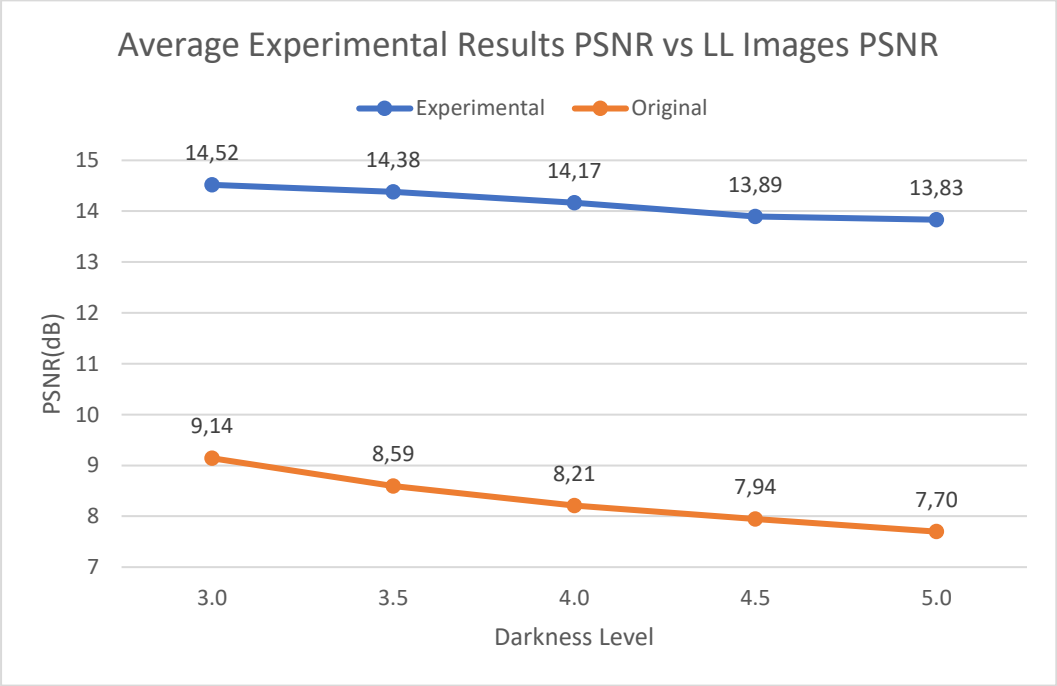


Figure 2.154: Experimental vs LL images PSNR

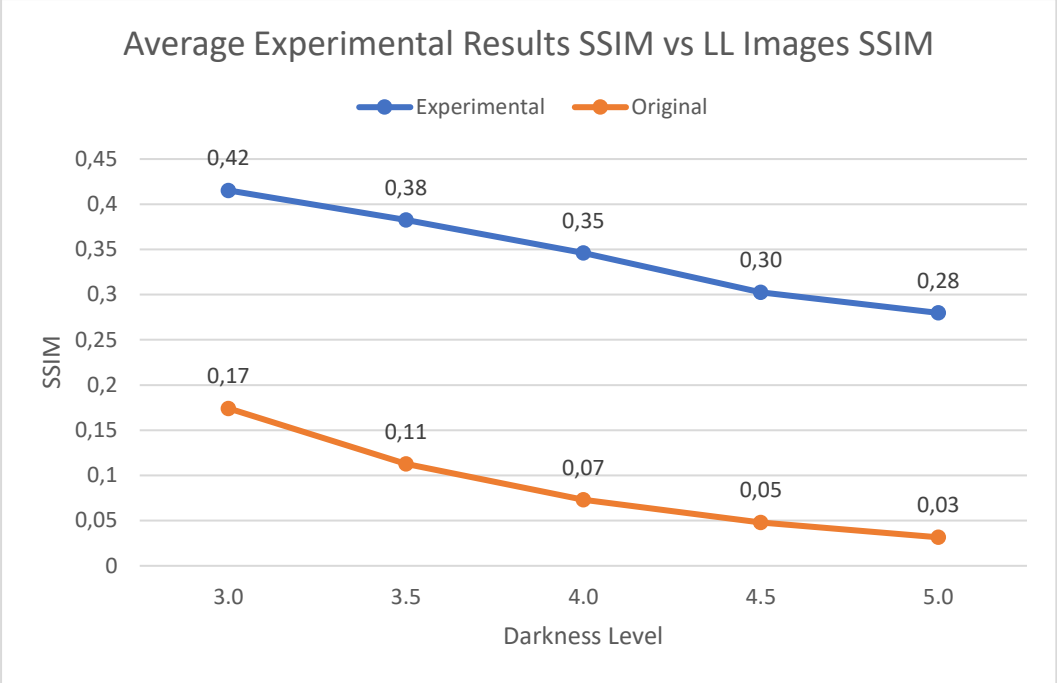


Figure 2.155: Experimental vs LL images SSIM

The same behavior is observed in the validation set as well. The results improve noticeably, and the metrics follow the expected path as the darkness level increases. MSE increases while PSNR and SSIM decrease with increasing darkness level. Let us comment that here too the rate of fall of these values is lower compared to the methods we have seen so far.

Test Set

	MSE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	239,1835597	258,6719	288,2718	372,3504	588,9472
MAX	12536,48023	15140,82	17434,88	18912,68	19070,45
AVERAGE	3000,909179	3111,417	3271,808	3603,883	4024,52
	PSNR(dB)				
Level	3.0	3.5	4.0	4.5	5.0
MIN	7,149047407	6,329309	5,716613	5,363272	5,327195
MAX	24,34349036	24,00331	23,53278	22,42129	20,43004
AVERAGE	14,31840327	14,16896	13,96122	13,59259	13,09024
	SSIM				
Level	3.0	3.5	4.0	4.5	5.0
MIN	0,005895394	0,003846	-0,00881	-0,11043	-0,15504
MAX	0,724767459	0,691906	0,654576	0,615887	0,558024
AVERAGE	0,436750452	0,3975	0,352682	0,299397	0,251513

	MV				
Level	3.0	3.5	4.0	4.5	5.0
MIN	127,1964766	127,0717	127,2591	127,3106	127,2778
MAX	137,1473075	148,0392	157,8103	164,3262	166,6853
AVERAGE	127,9494737	128,6499	129,8455	132,1473	134,6972
	STD				
Level	3.0	3.5	4.0	4.5	5.0
MIN	386,139099	425,2195	442,058	470,8506	395,7738
MAX	1343,650473	1312,27	1260,695	1249,68	1184,382
AVERAGE	862,7810241	855,5218	842,2514	821,2246	786,2158
	BRISQUE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	4,401809978	7,89315	7,270009	13,12065	12,85632
MAX	59,64634688	59,56688	56,33996	57,22262	54,78736
AVERAGE	29,55965034	30,80847	31,31449	33,91571	38,08425
	NIQE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	2,545997664	2,755651	3,246542	3,530169	3,828247
MAX	8,500301515	7,463091	9,121375	9,93999	11,37396
AVERAGE	3,933127676	4,234365	4,663744	5,121773	5,646432

Table 2.27: HE quality metrics on test set

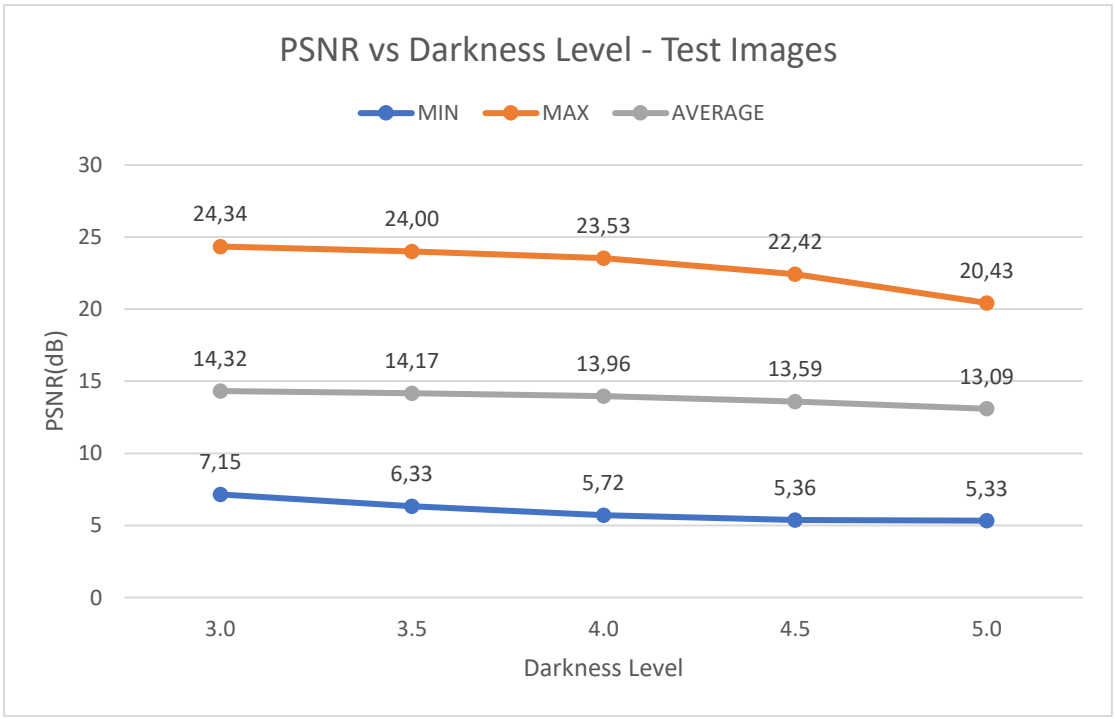


Figure 2.156: Experimental results PSNR vs Darkness level for test set

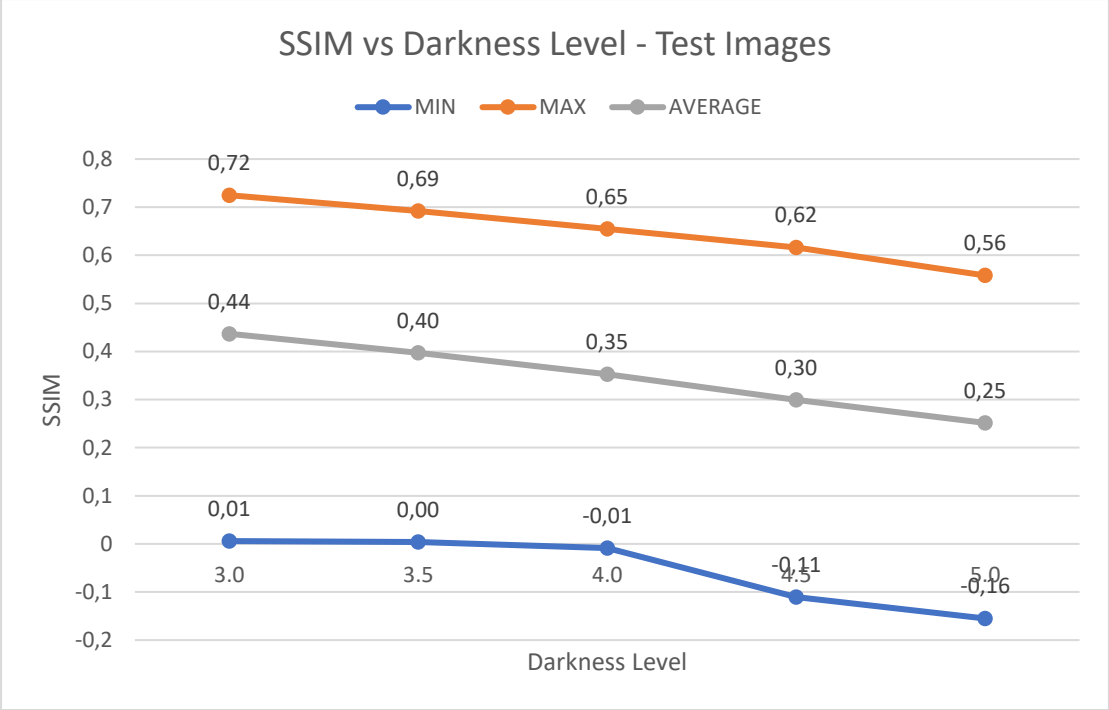


Figure 2.157: Experimental results SSIM vs Darkness level for test set

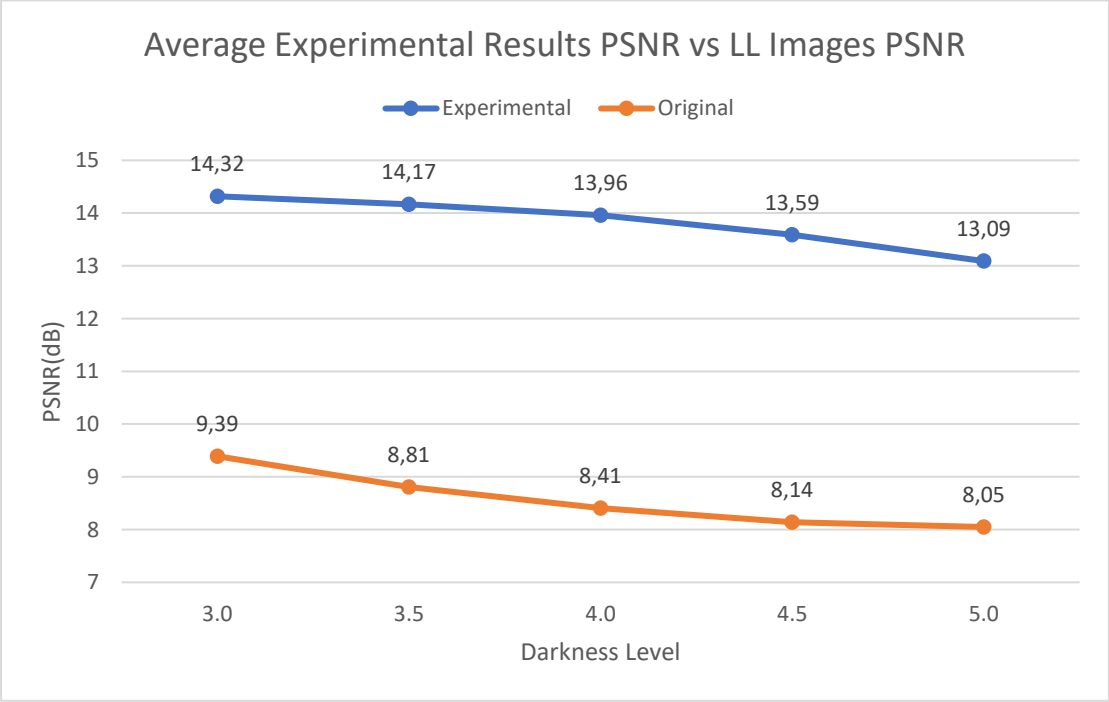


Figure 2.158: Experimental vs LL images PSNR

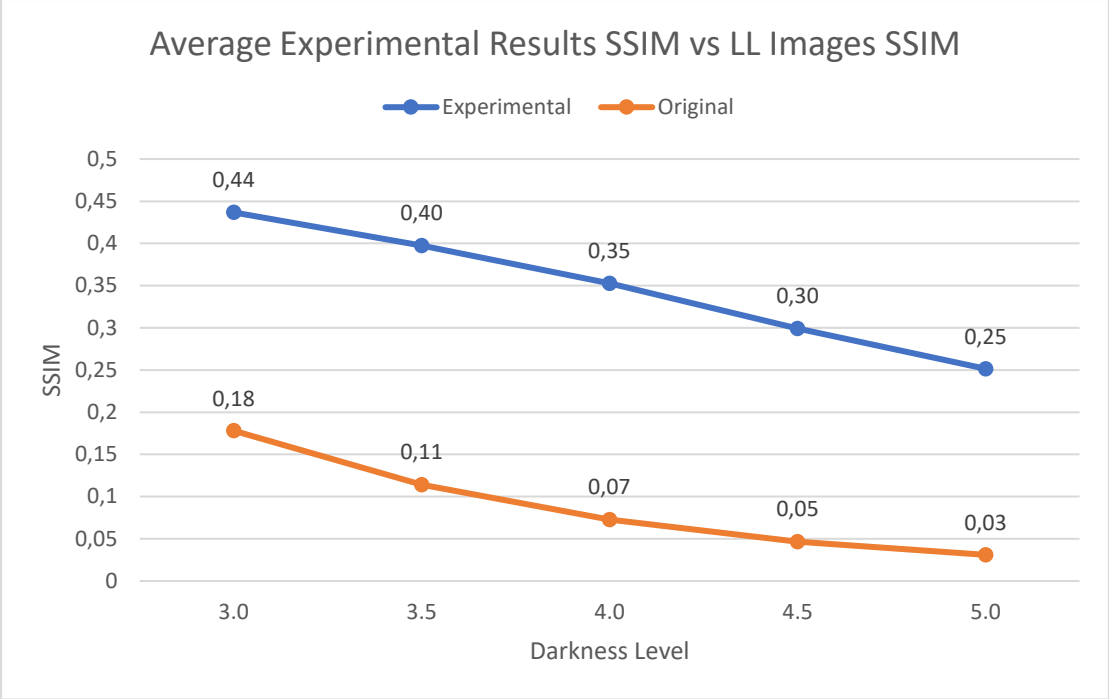


Figure 2.159: Experimental vs LL images SSIM

And in the case of the test set, we see the same behavior, with the increase in the darkness level reducing the performance of the algorithm.

Commenting on the results as a whole, we notice that the histogram equalization noticeably improves the values of the metrics, and by extension the quality of the LL images. Furthermore, with increasing darkness level, the expected behavior is observed, as the values of the quality metrics decrease with increasing darkness level. In particular, the MSE increases, while the PSNR and SSIM decrease with the increase of the darkness level, which means that the image quality decreases and that we move away from the ground truth case. At this point we have to comment, however, that the rate at which these two metrics decrease is very small, which indicates that the method is resistant to variations in the darkness level. For the quality metrics with no reference, we see that MV remains roughly constant with increasing darkness level, and similar behavior is observed for STD. This confirms the fact that the method is resistant to changes in the darkness level. Finally, BRISQUE and NIQE increase with increasing darkness level, which means that we are moving away from natural statistics. This is because the images become very dark, making it difficult to retrieve the full visual information.

To justify the results visually, we will present the images corresponding to the minimum and maximum PSNR for each darkness level. In addition,

together we will present the corresponding LL and ground truth cases, together with their histograms, commenting each time on the results.

Darkness Level: 3.0

MIN PSNR

Original Low Light **Normal Light** **Experimental Result**

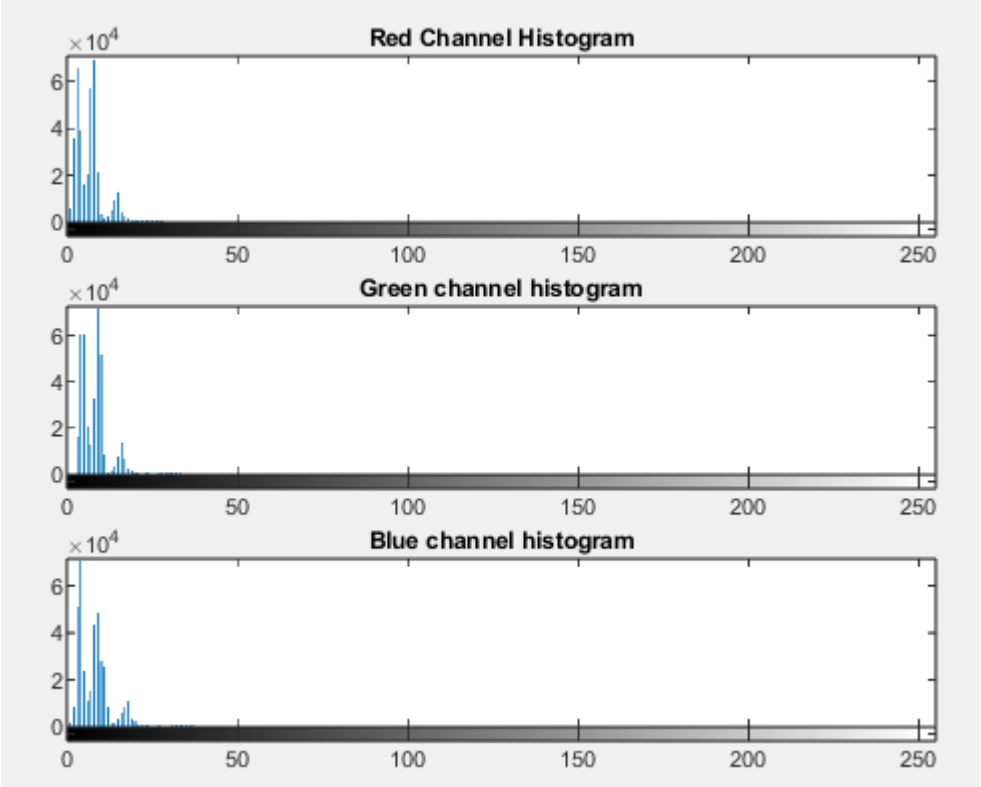


Figure 2.160: Histogram of original LL Image

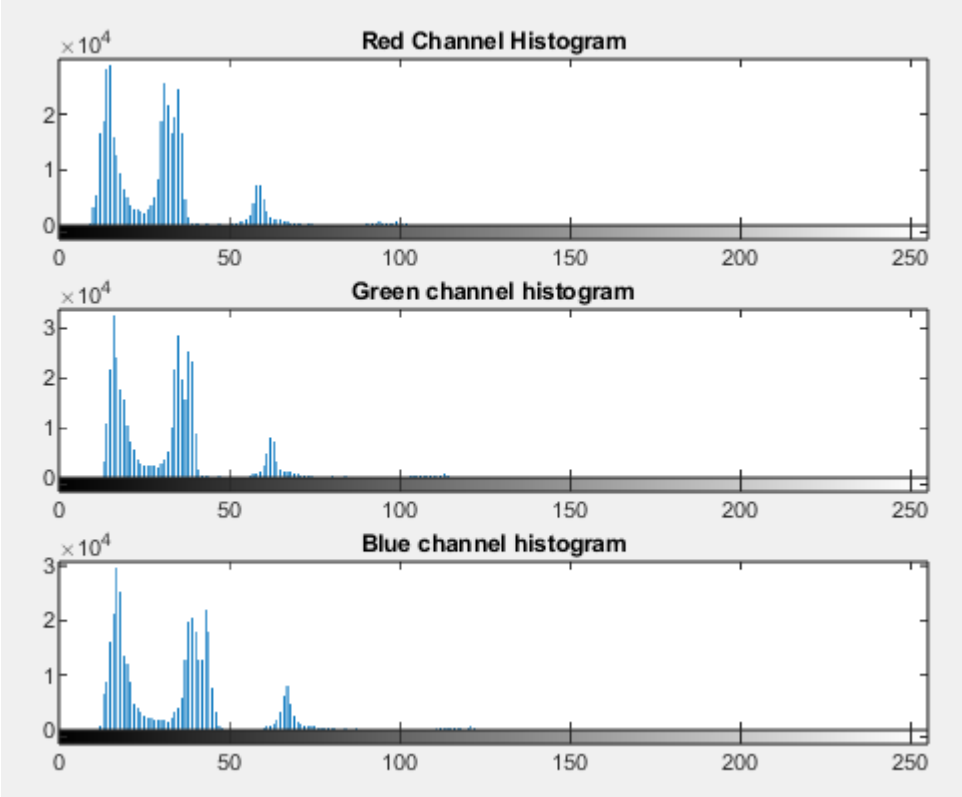


Figure 2.161: Histogram of NL Image

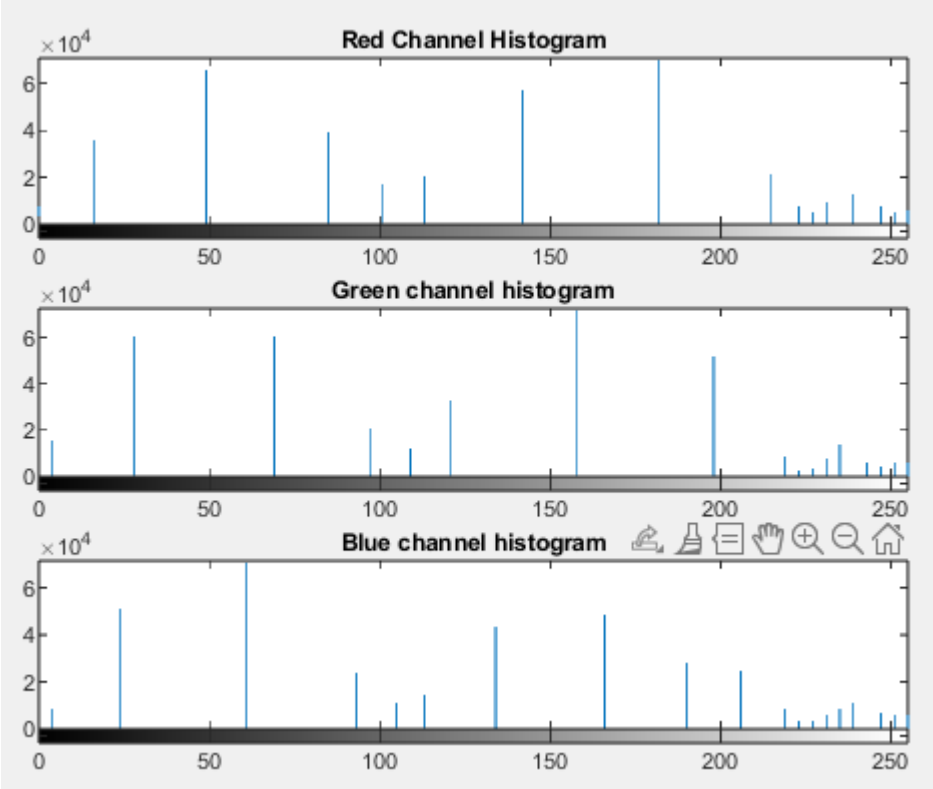


Figure 2.162: Histogram of experimental result with min PSNR

For darkness level 3.0 the experimental result corresponding to the minimum PSNR is characterized by strong color distortions and, although a large part of visual details has been recovered, it remains far from the ground truth case. If we observe the histogram of the ground truth case, we see that the pixel values are not distributed over the entire available range, but remain in the left part of the histogram, in a larger range compared to the LL image. We know that the purpose of histogram equalization is to spread the pixel values across the available range of values, which is achieved as we can see from the histogram in Figure 2.162. Based on this we understand that it is impossible to reproduce the ground truth case, and for that reason the experimental result shows large color distortions and is far from the ground truth image.

MAX PSNR

Original Low Light **Normal Light** **Experimental Result**

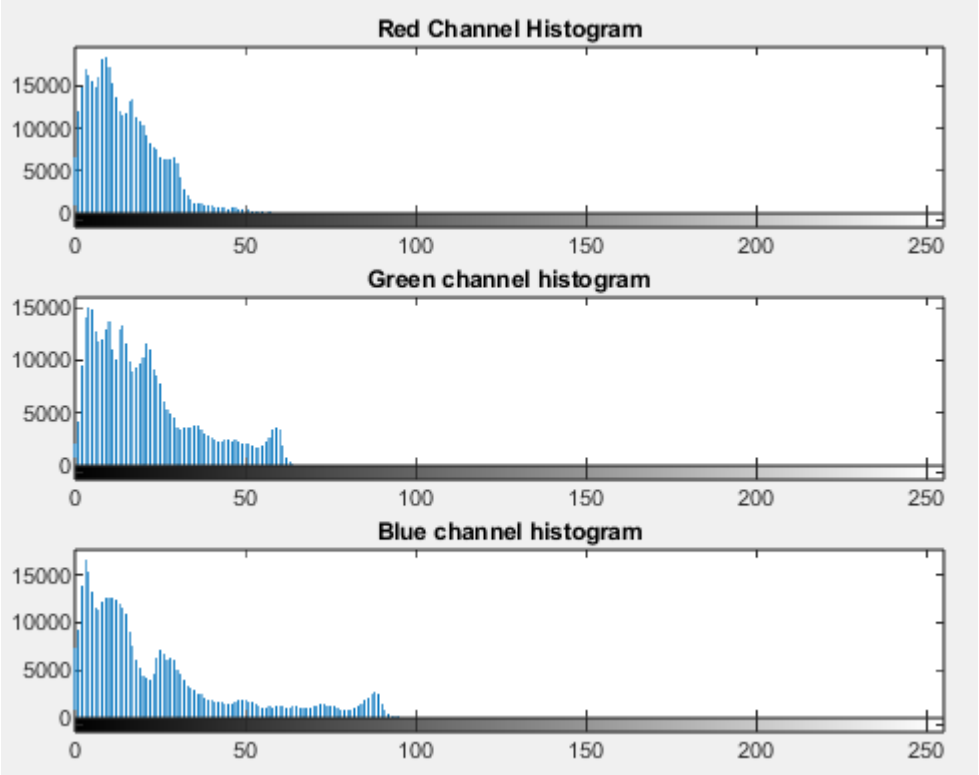


Figure 2.163: Histogram of original LL Image

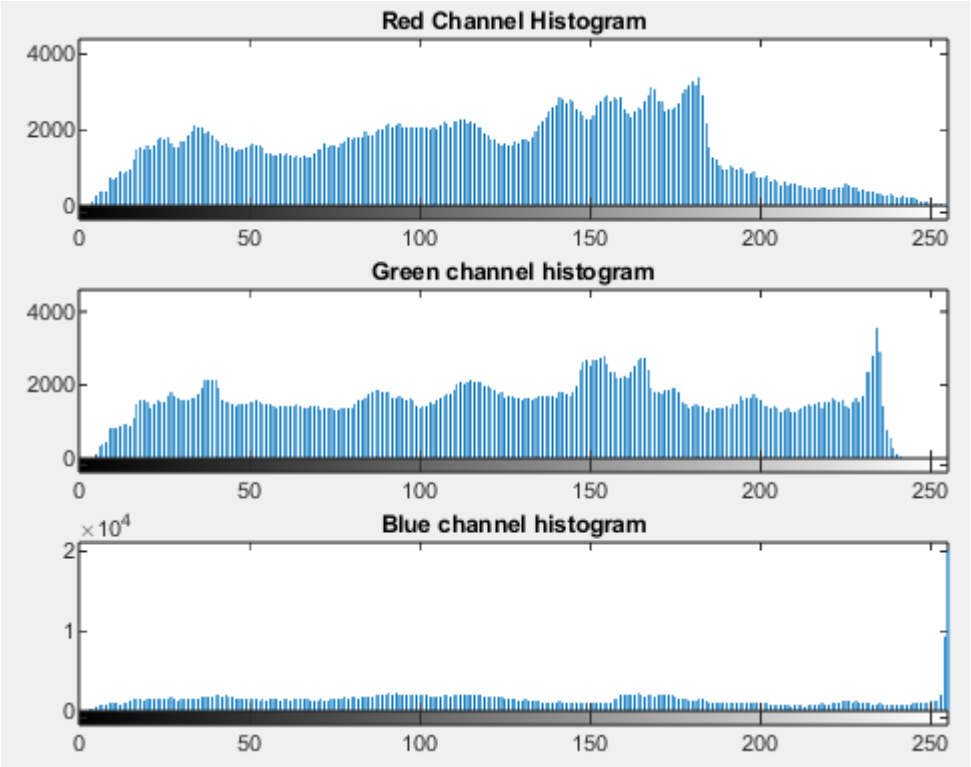


Figure 2.164: Histogram of NL Image

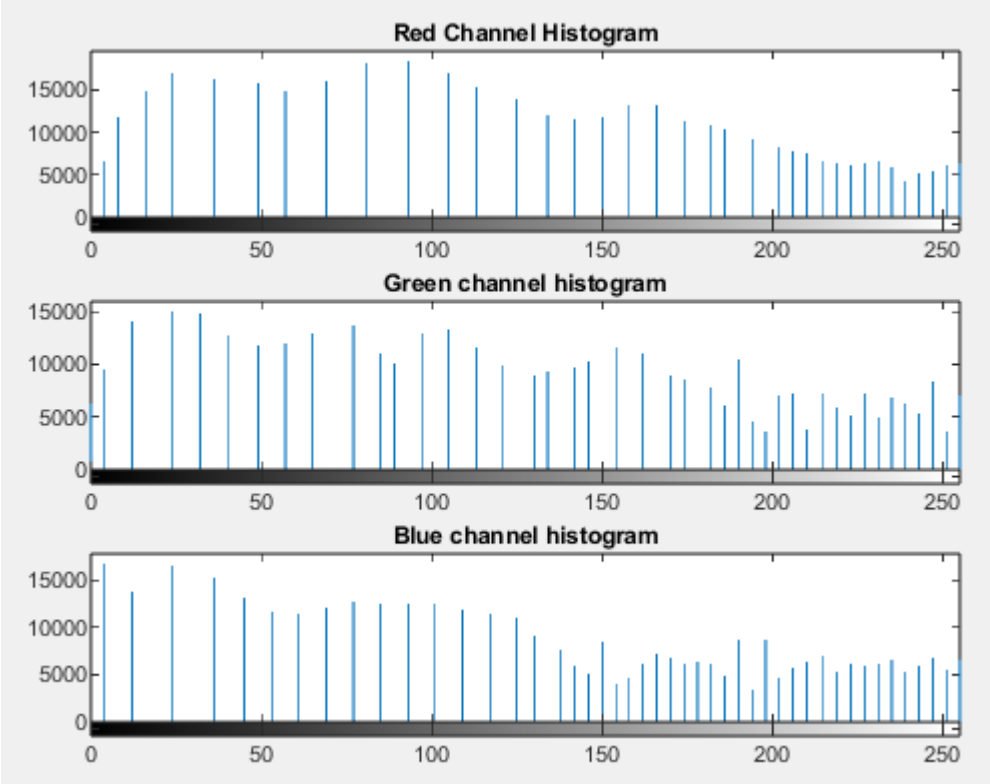


Figure 2.165: Histogram of experimental result with max PSNR

In this case we see from the histogram in figure 2.164 that the ground truth image is characterized by a histogram with values evenly distributed throughout the range, so we expect the experimental result to correctly reproduce the ground truth case. Indeed, as we see the experimental result is visually the same as the ground truth image, and most of the color and visual information has been recovered. This is also confirmed by the histogram, in figure 2.165, whose form is very close to the ground truth case, with the pixel values distributed almost uniformly throughout the available value range, simultaneously increasing the contrast of the image.

Darkness Level: 3.5

MIN PSNR

Original Low Light **Normal Light** **Experimental Result**

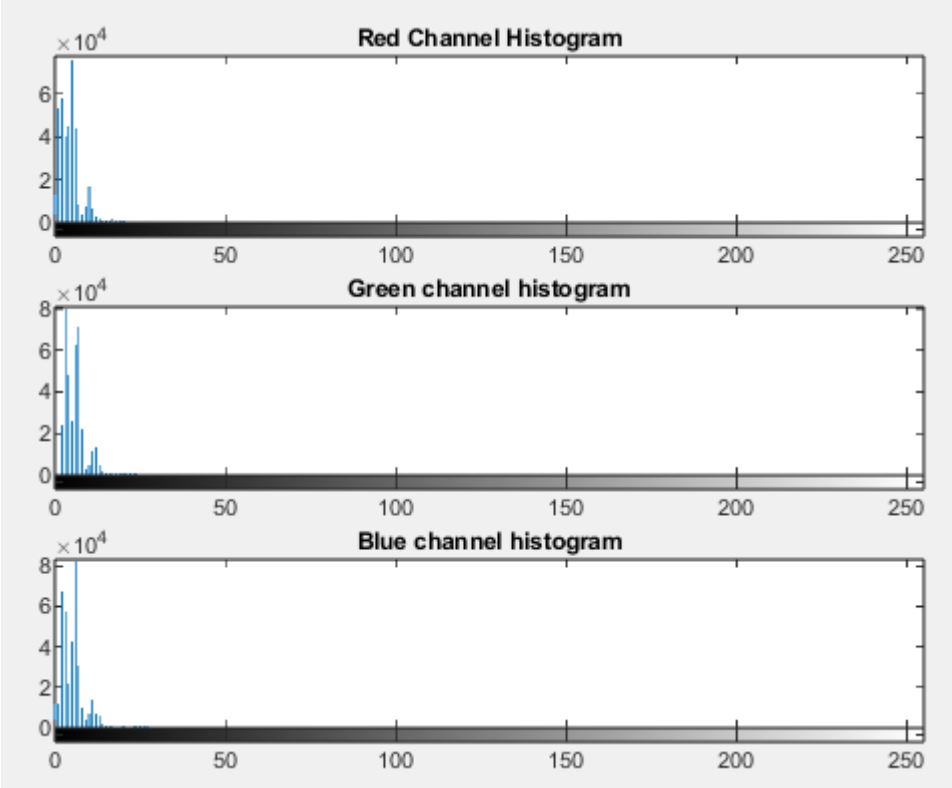


Figure 2.166: Histogram of original LL Image

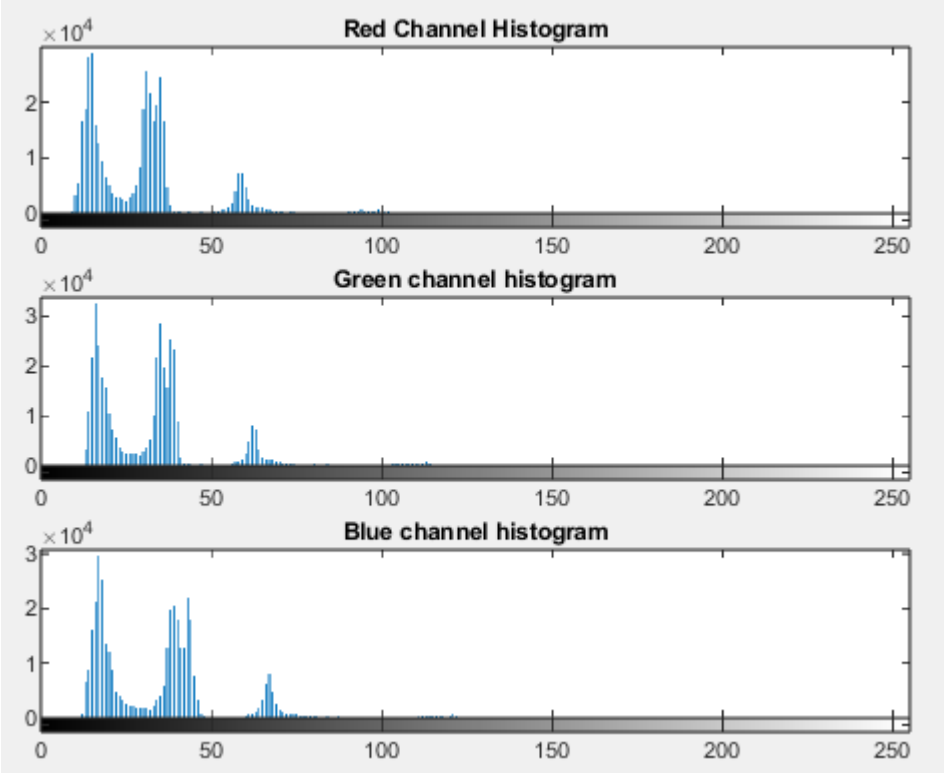


Figure 2.167: Histogram of NL Image

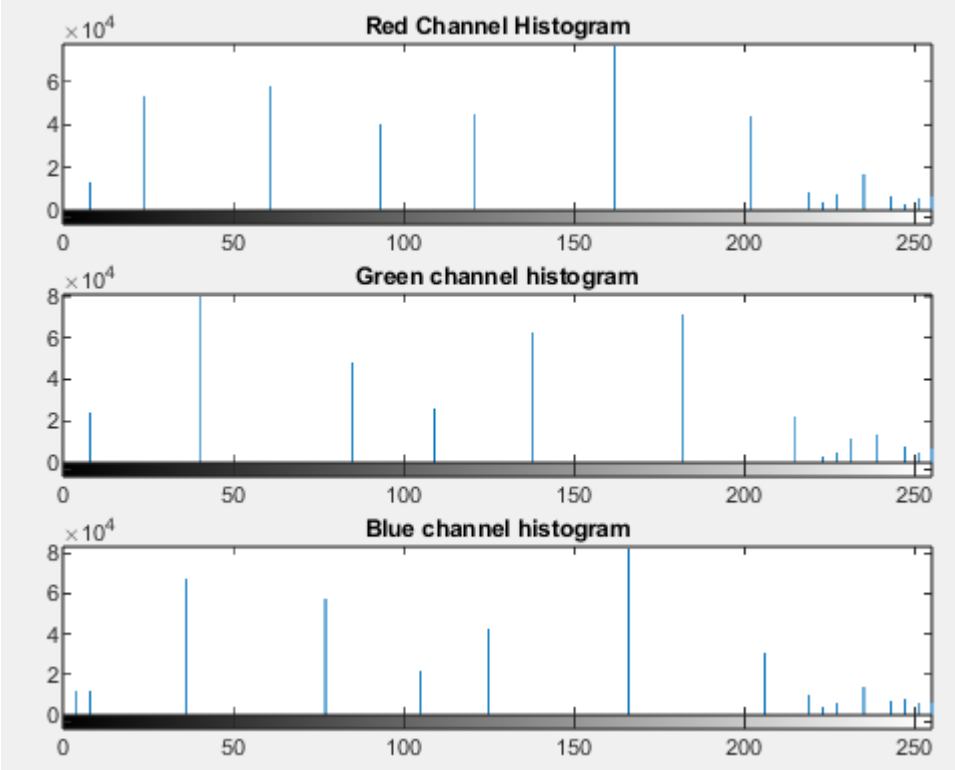


Figure 2.168: Histogram of experimental result with min PSNR

For darkness level 3.5, the image with the minimum experimental PSNR is the same as that obtained for darkness level 3.0. Here again, the histogram of the ground truth case is characterized by values that are concentrated in the left part of the histogram, and not evenly distributed throughout the available range. So, we understand that histogram equalization cannot reproduce this effect, as its purpose is to produce an image with histogram values evenly distributed throughout the available range of values. Thus, here is an image with strong color distortions, far from the ground truth case.

MAX PSNR

Original Low Light **Normal Light** **Experimental Result**

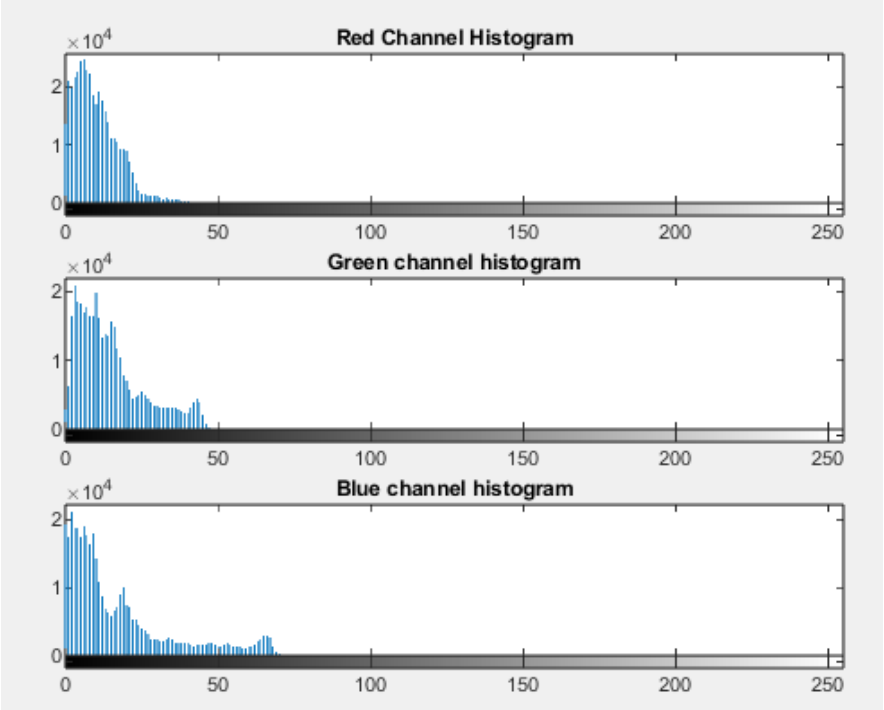


Figure 2.169: Histogram of original LL Image

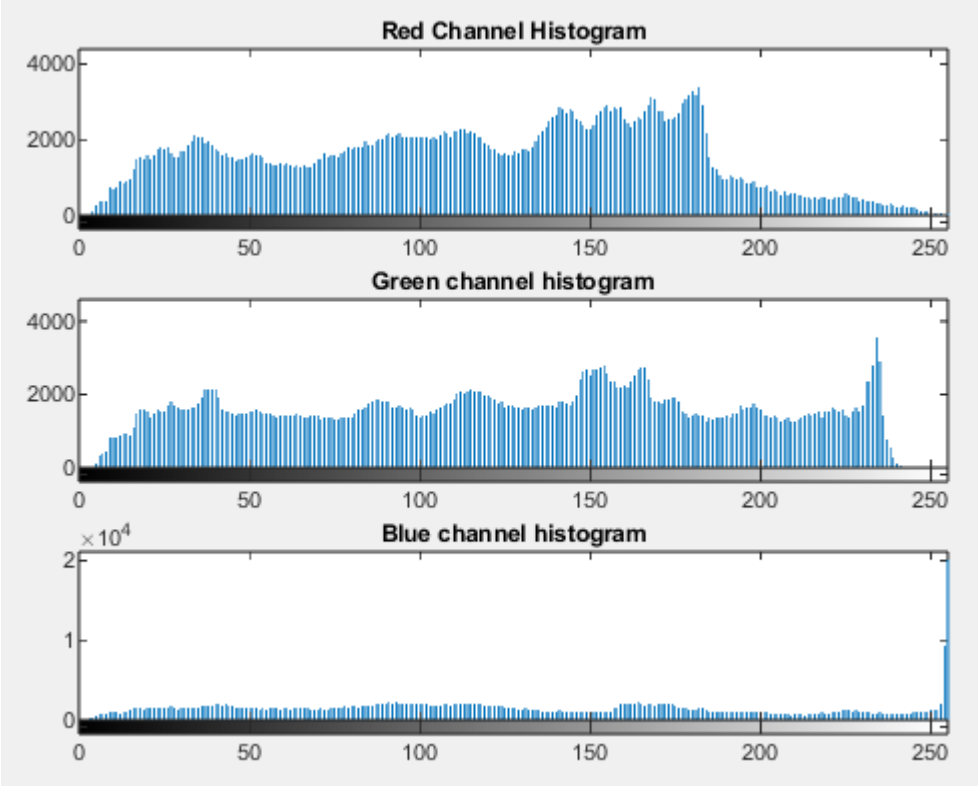


Figure 2.170: Histogram of NL Image

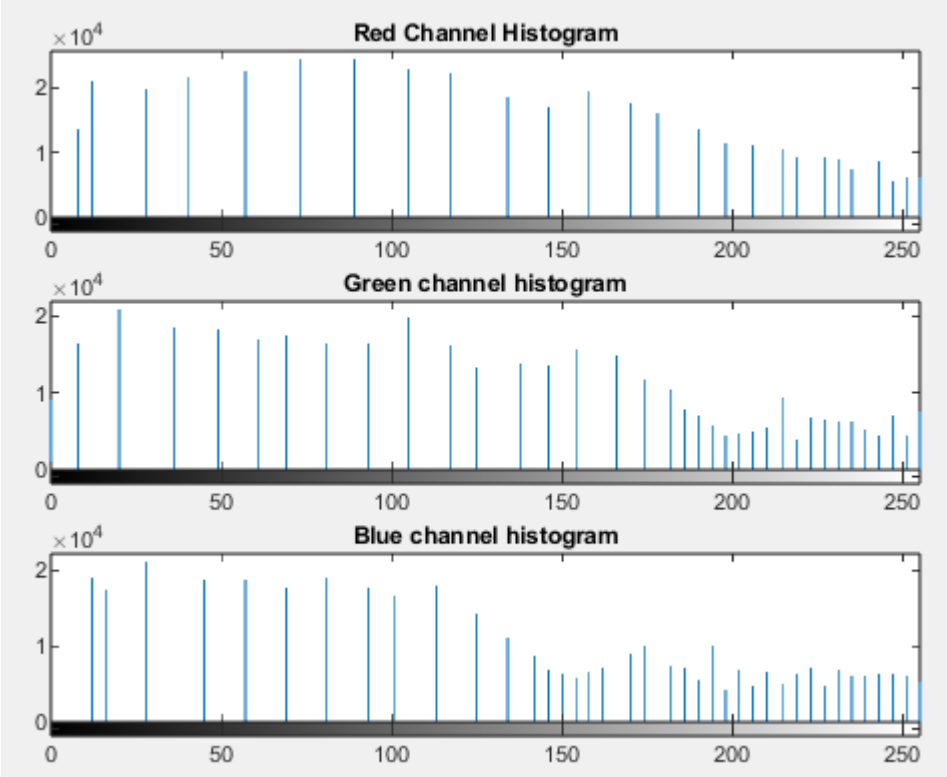
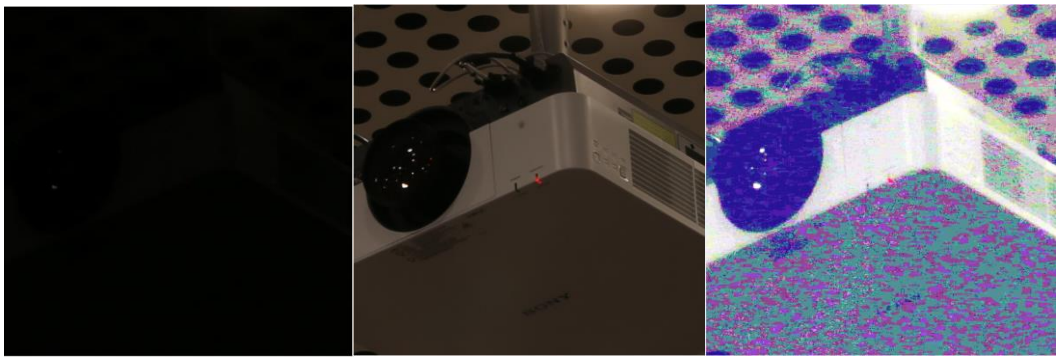


Figure 2.171: Histogram of experimental result with max PSNR

For the case of the maximum experimental PSNR, the same picture is obtained with darkness level 3.0. The ground truth image is characterized by a histogram with values spread over the entire available range, so it is easy to reproduce by applying histogram equalization. Indeed, the experimental result is visually very close to the ground truth case, having managed to recover most of the visual information. This is also confirmed by the histogram in figure 2.171, which is very close to the ground truth case, with the pixel values almost evenly distributed over the entire range of available values.

Darkness Level: 4.0

MIN PSNR



Original Low Light **Normal Light** **Experimental Result**

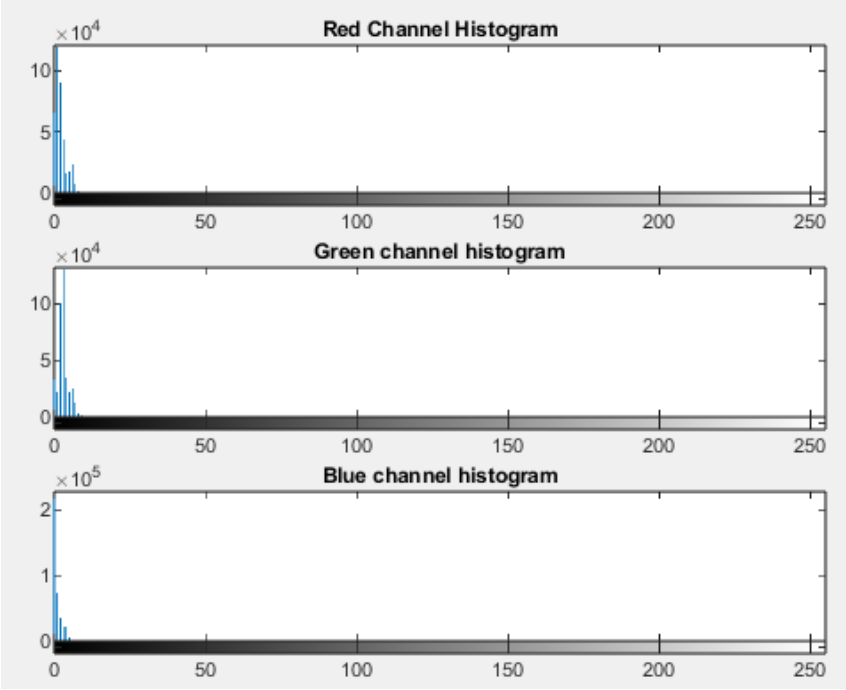


Figure 2.172: Histogram of original LL Image

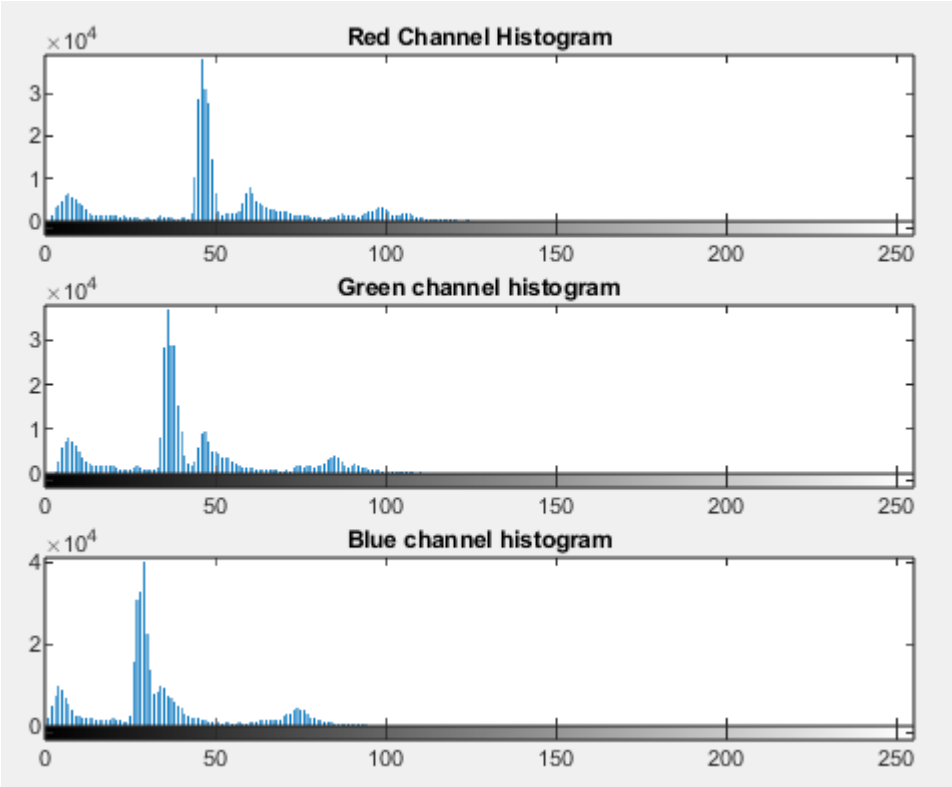


Figure 2.173: Histogram of NL Image

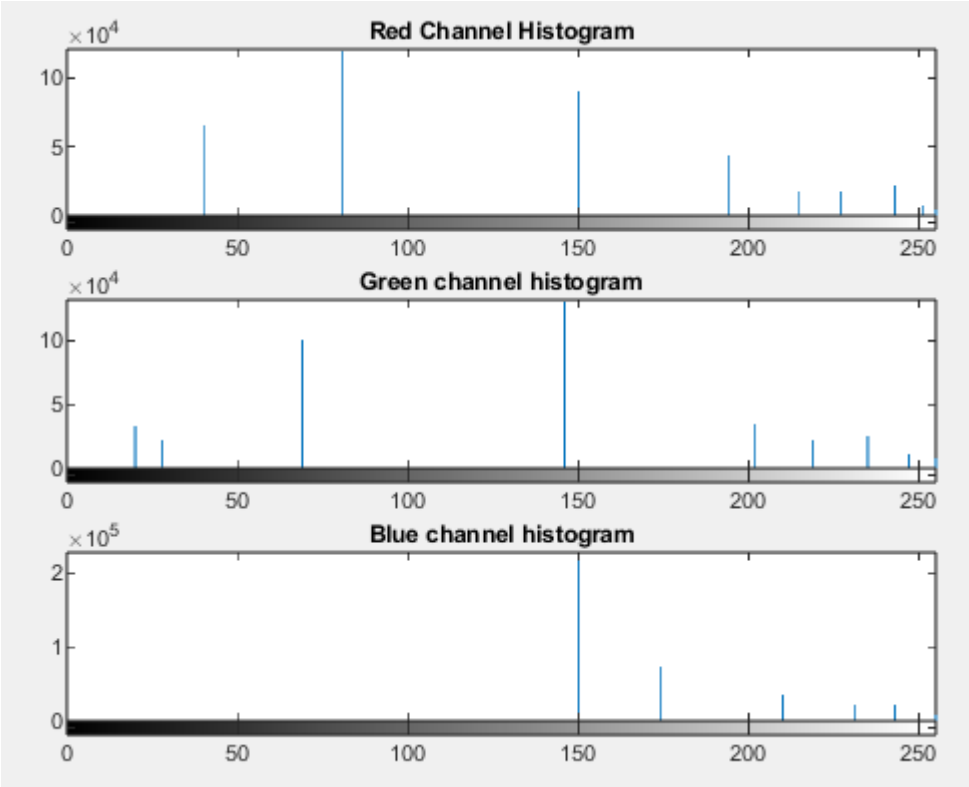


Figure 2.174: Histogram of experimental result with min PSNR

In the case of darkness level 4.0 for the minimum experimental PSNR, a similar case arises as what we saw above. The first thing we can comment on is that the original LL image is too dark, which we can also confirm from its histogram in figure 2.172, where we see that the values are clustered in a narrow range in the left part. So, we expect that full and correct visual information cannot be retrieved. Second, the ground truth image is characterized by a histogram with values that are not distributed over the entire available range of values, as seen in Figure 2.173, which histogram equalization cannot reproduce. The combination of these two factors leads to a result with the minimum experimental PSNR, with the image characterized by strong chromatic aberrations, and its pixel values distributed in very few gray levels, which justifies the chromatic aberrations as well.

MAX PSNR

Original Low Light **Normal Light** **Experimental Result**

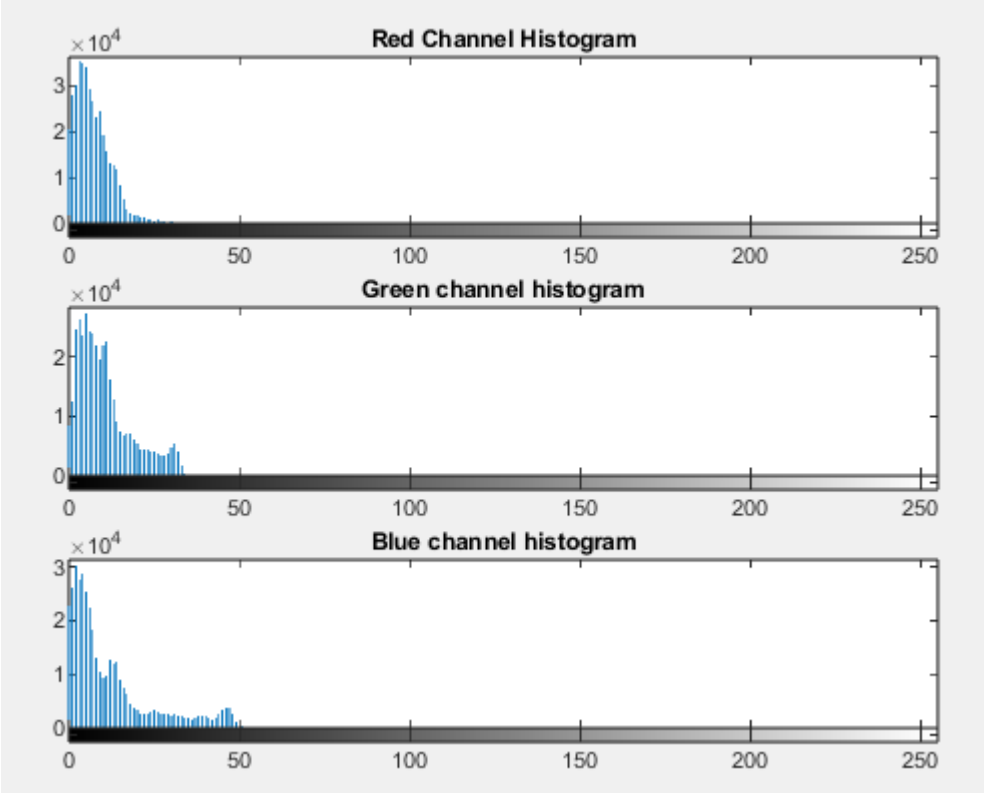


Figure 2.175: Histogram of original LL Image

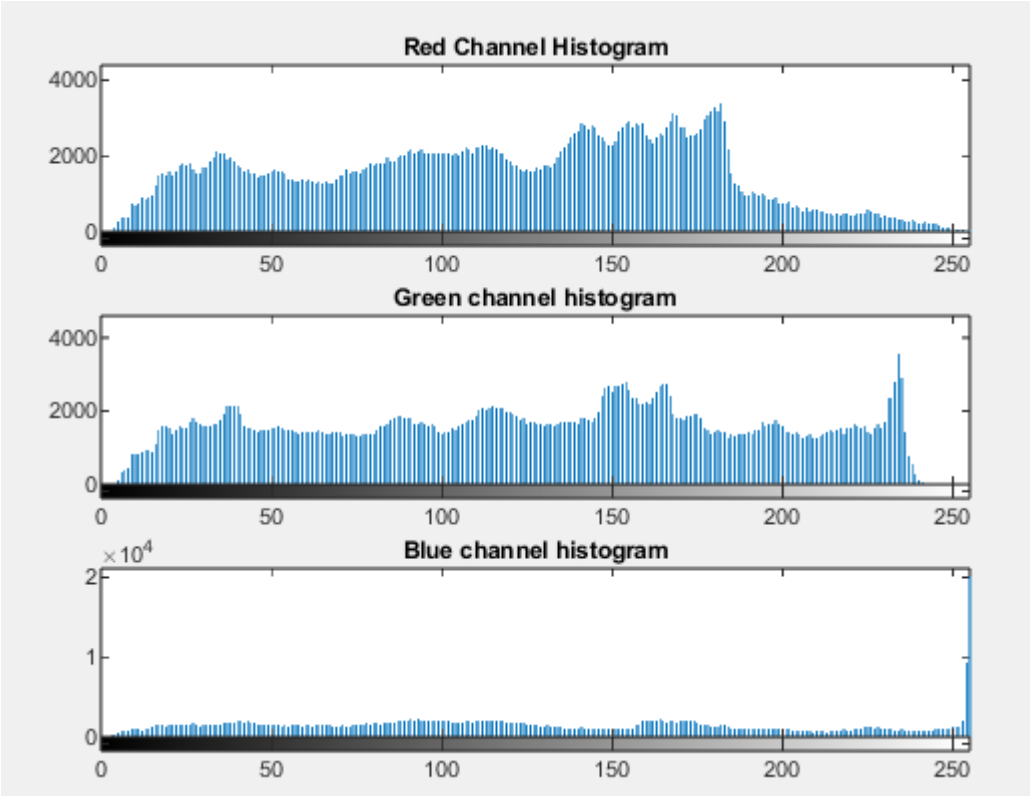


Figure 2.176: Histogram of NL Image

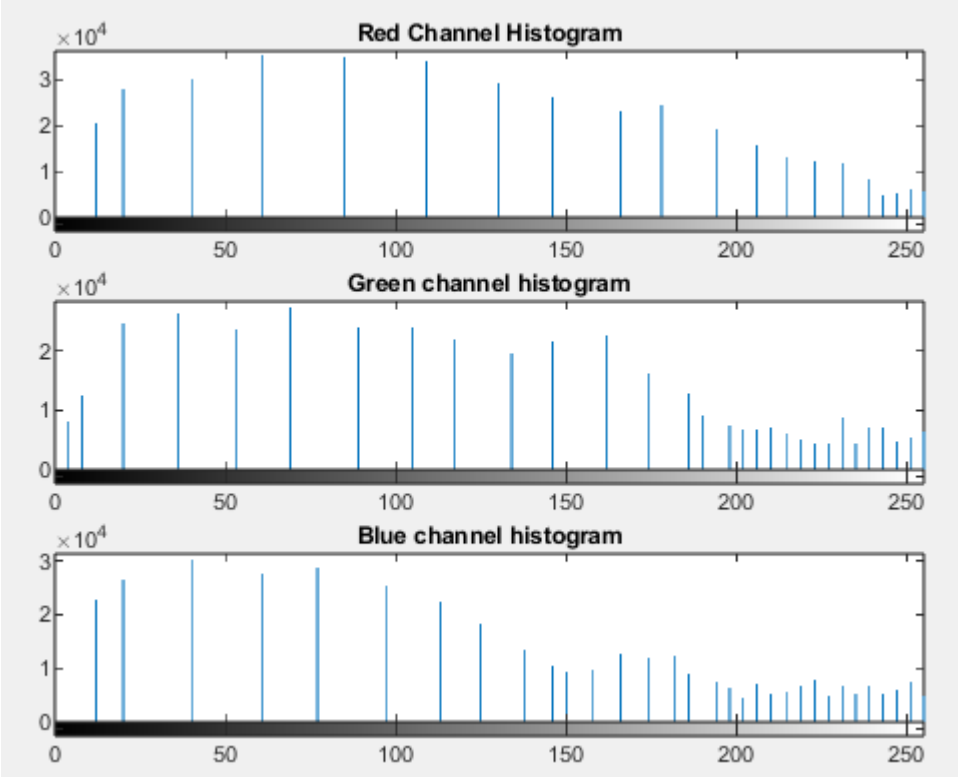


Figure 2.177: Histogram of experimental result with max PSNR

For the case of the maximum experimental PSNR, the same picture is obtained with darkness levels 3.0 and 3.5. The ground truth case of this image is characterized by a histogram with values spread across the available range, which is easy to reproduce by histogram equalization. Indeed, the experimental result is visually very close to the ground truth case, and the only negative we can notice is a fading of the colors. This can be justified by the fact that the LL image is very dark, which makes it difficult to recover the full color information, which is also confirmed by the experimental histogram, where we observe that the pixel values are distributed in fewer gray levels, compared to the earlier cases, explaining the less color information.

Darkness Level: 4.5

MIN PSNR

Original Low Light **Normal Light** **Experimental Result**

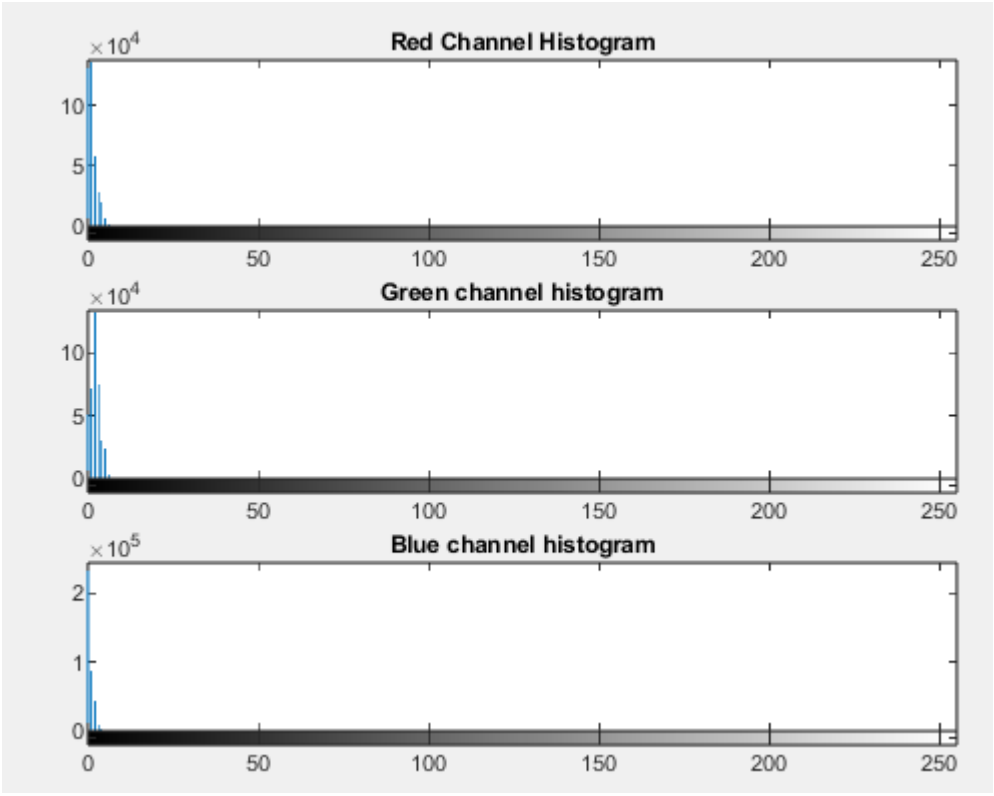


Figure 2.178: Histogram of original LL Image

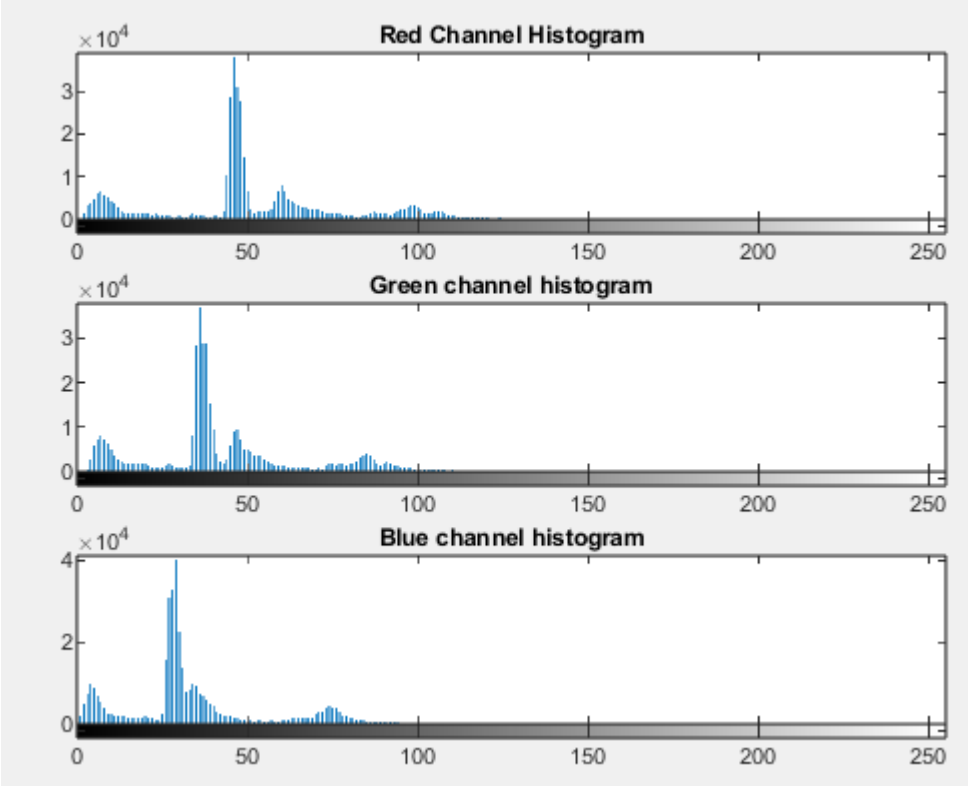


Figure 2.179: Histogram of NL Image

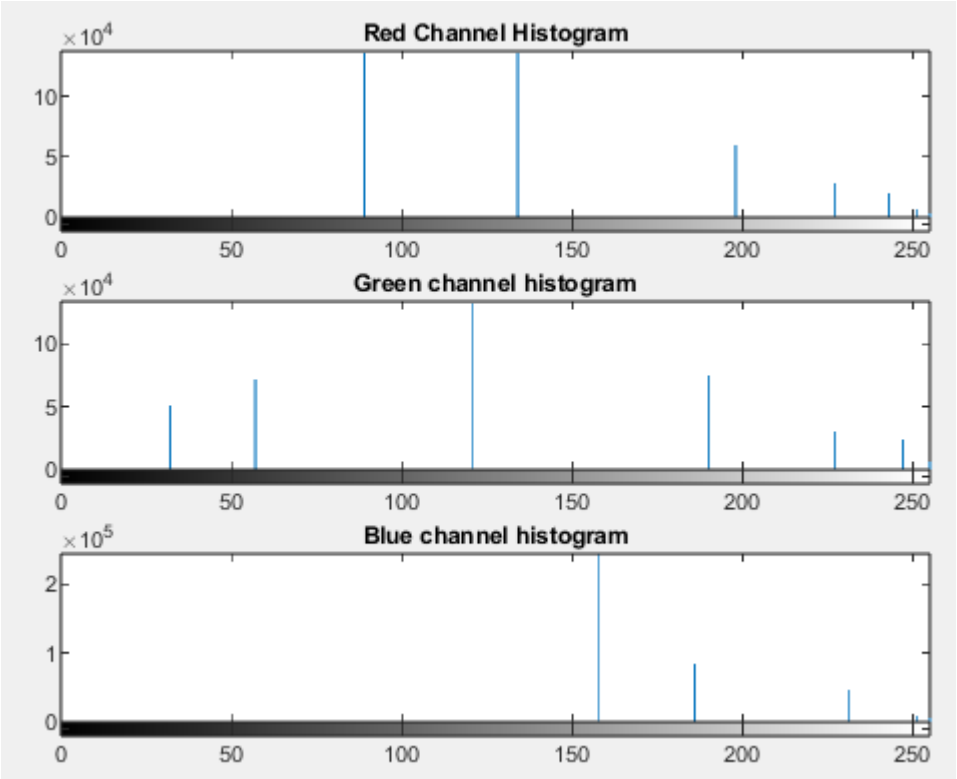


Figure 2.180: Histogram of experimental result with min PSNR

For darkness level 4.5 the image with the minimum experimental PSNR is the same as the one obtained for darkness level 4.0. We notice that the experimental result is characterized by strong color distortions and is visually far from the ground truth case. This is due to the two main factors, which we mentioned above. First, the image is very dark, with pixel values clustered in the left part of the histogram, resulting in a very small dynamic range, making it very difficult to retrieve full and correct visual information. Second, the ground truth image has a histogram with values that are not distributed across the available range, which histogram equalization cannot reproduce. The combination of these two factors leads to the low quality result observed.

MAX PSNR

Original Low Light **Normal Light** **Experimental Result**

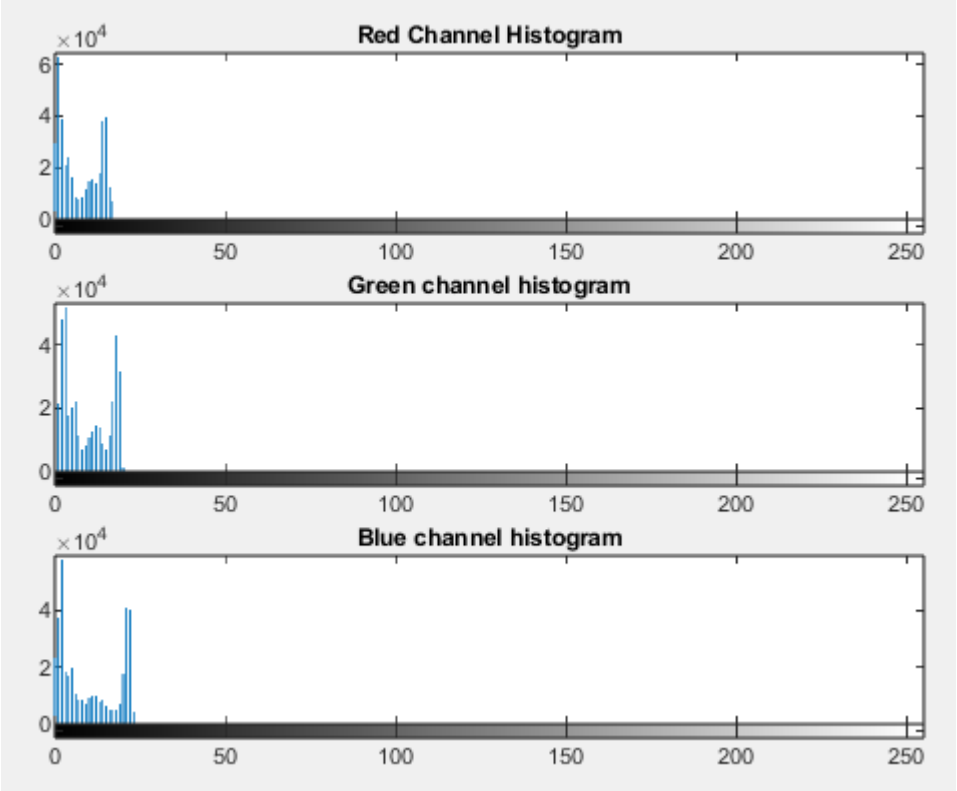


Figure 2.181: Histogram of original LL Image

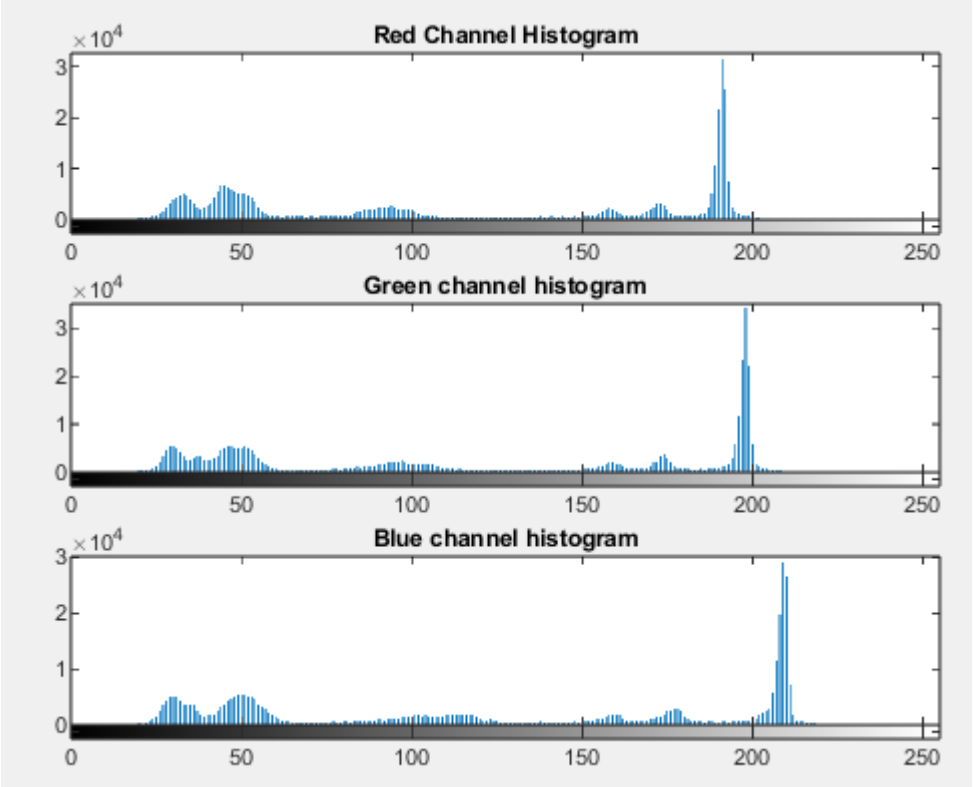


Figure 2.182: Histogram of NL Image

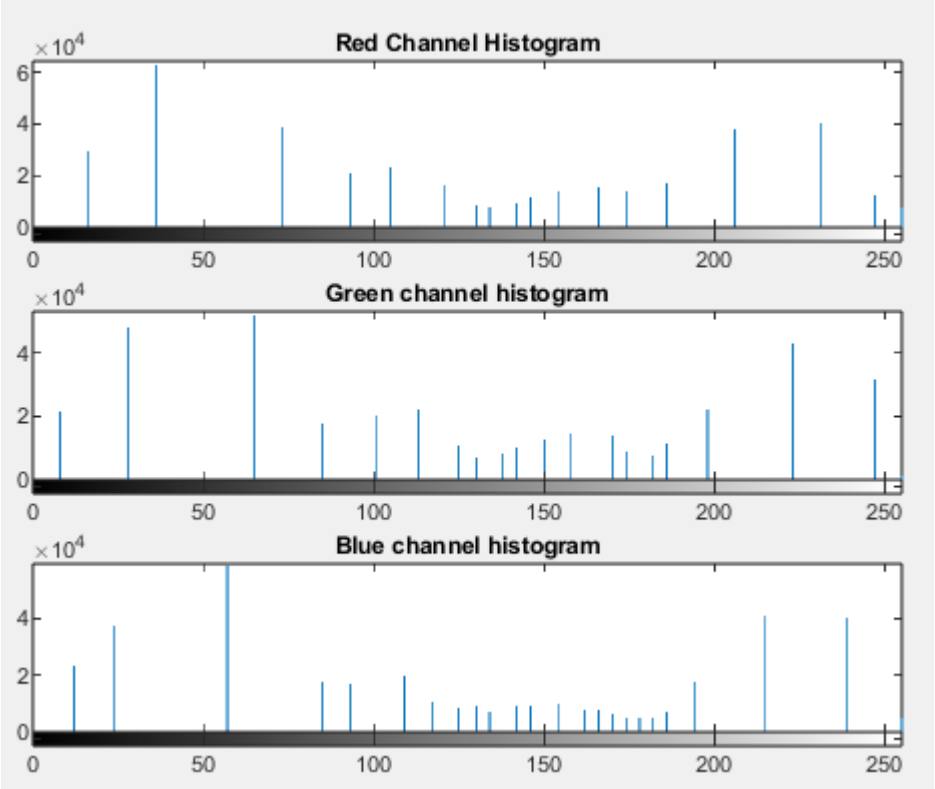
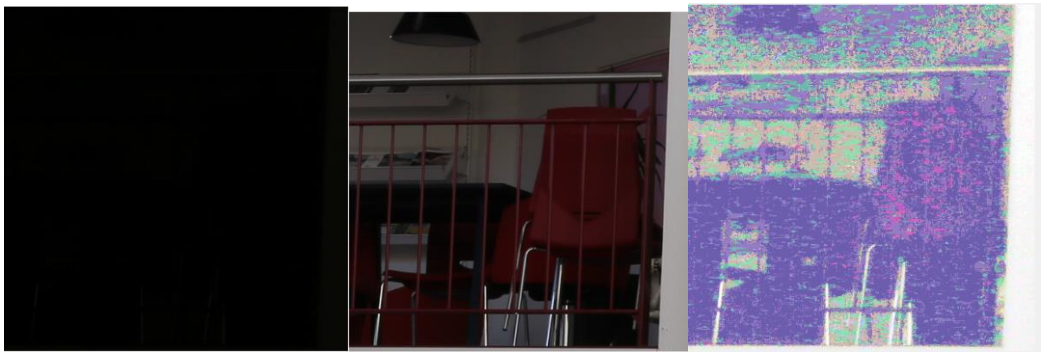


Figure 2.183: Histogram of experimental result with max PSNR

In the case of the maximum experimental PSNR we observe that the experimental result is visually very close to the ground truth case. The histogram of the ground truth image is characterized by values spread over the entire range of available values, so histogram equalization can reproduce this. Indeed, the resulting experimental histogram is close to the ground truth case, with the only difference being that in the experimental the pixel values are distributed in fewer gray value levels. This is because the image is too dark making it very difficult to recover the full visual information. Nevertheless, the resulting image is satisfactory given the darkness level.

Darkness Level: 5.0

MIN PSNR



Original Low Light **Normal Light** **Experimental Result**

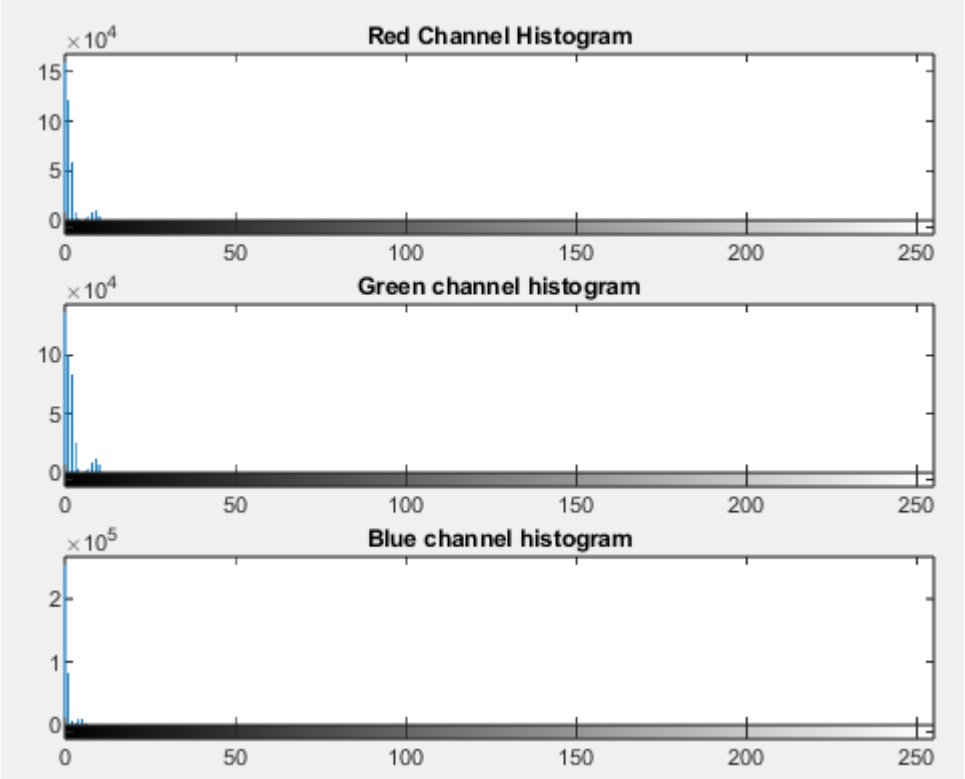


Figure 2.184: Histogram of original LL Image

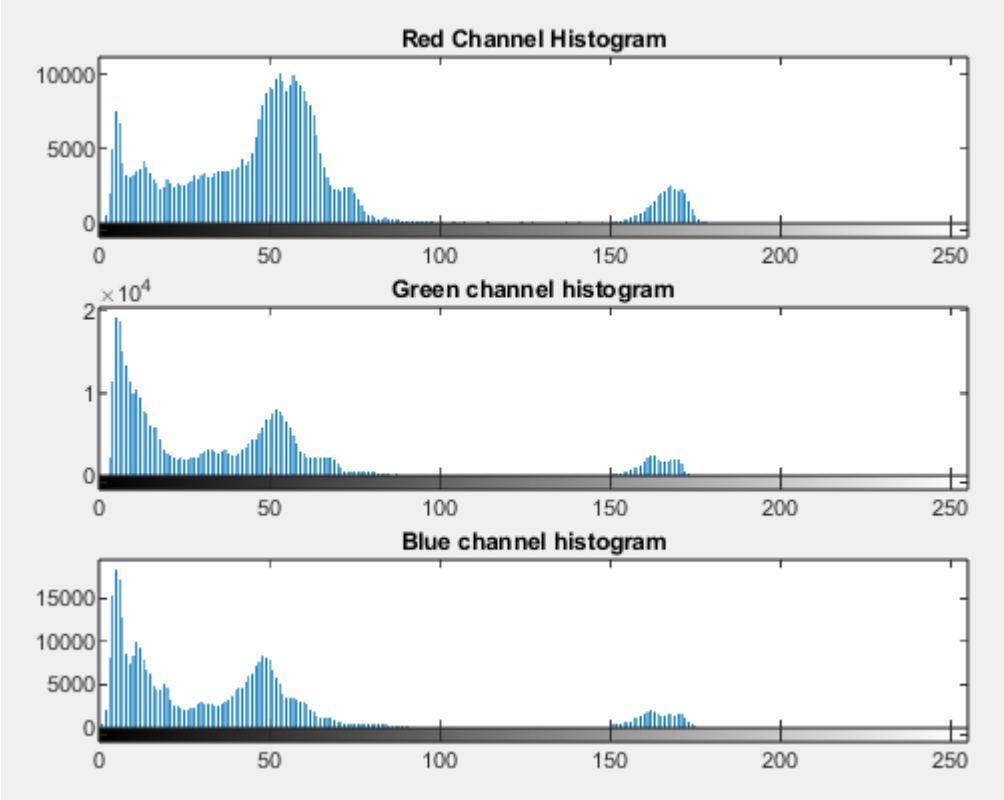


Figure 2.185: Histogram of NL Image

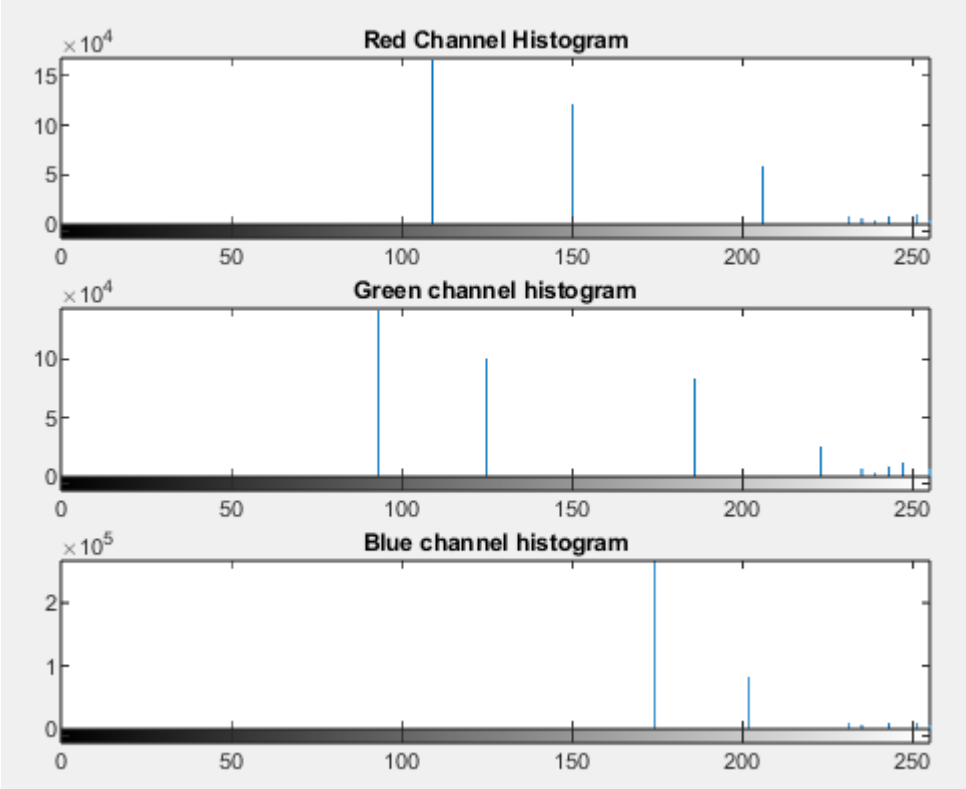


Figure 2.186: Histogram of experimental result with min PSNR

For darkness level 5.0, the experimental result corresponding to the minimum PSNR is characterized by strong color distortions and limited details. This is because the image is too dark, with pixel values accumulating on the left side of the histogram, making it difficult to retrieve visual information. This is also reflected in the experimental histogram of figure 2.186, where we see that the pixel values have been distributed in minimal gray value levels, explaining the intense color distortions and the low quality of the experimental result.

MAX PSNR

Original Low Light **Normal Light** **Experimental Result**

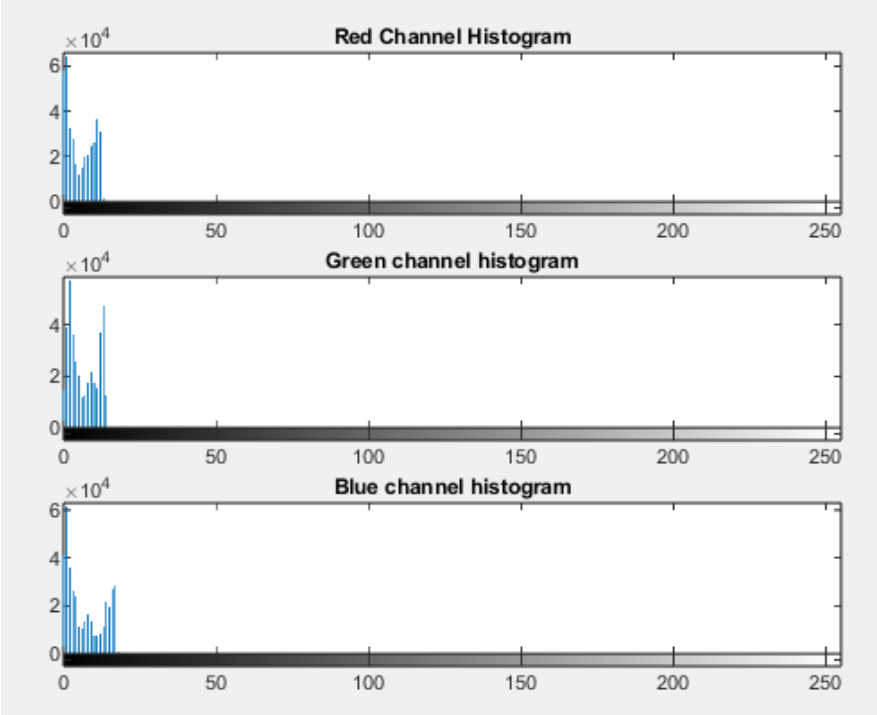


Figure 2.187: Histogram of original LL Image

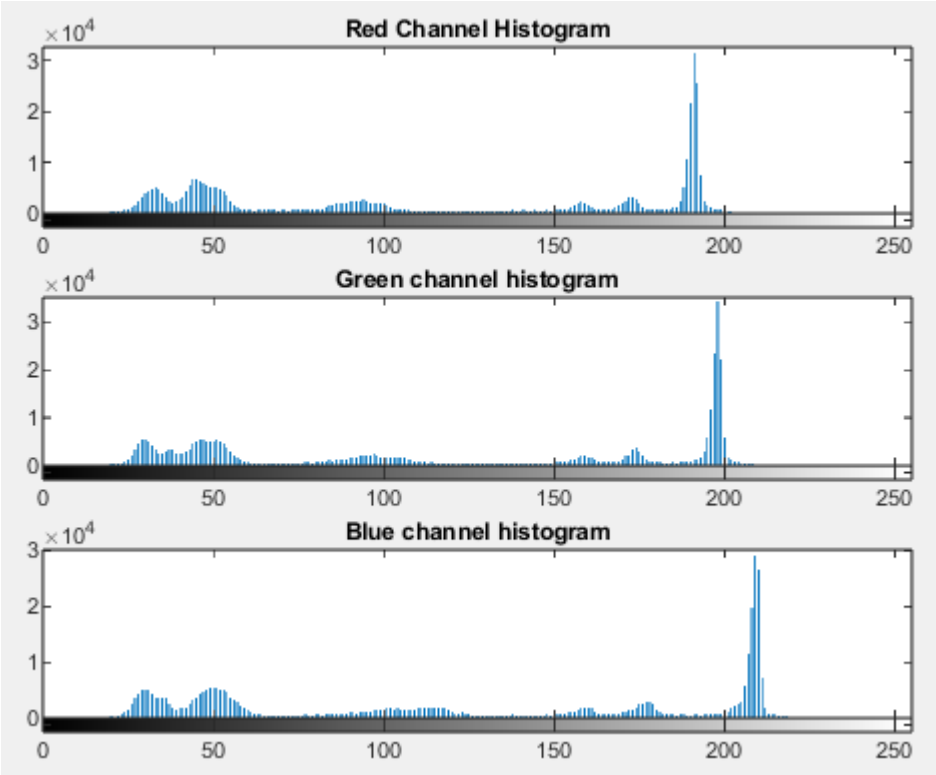


Figure 2.188: Histogram of NL Image

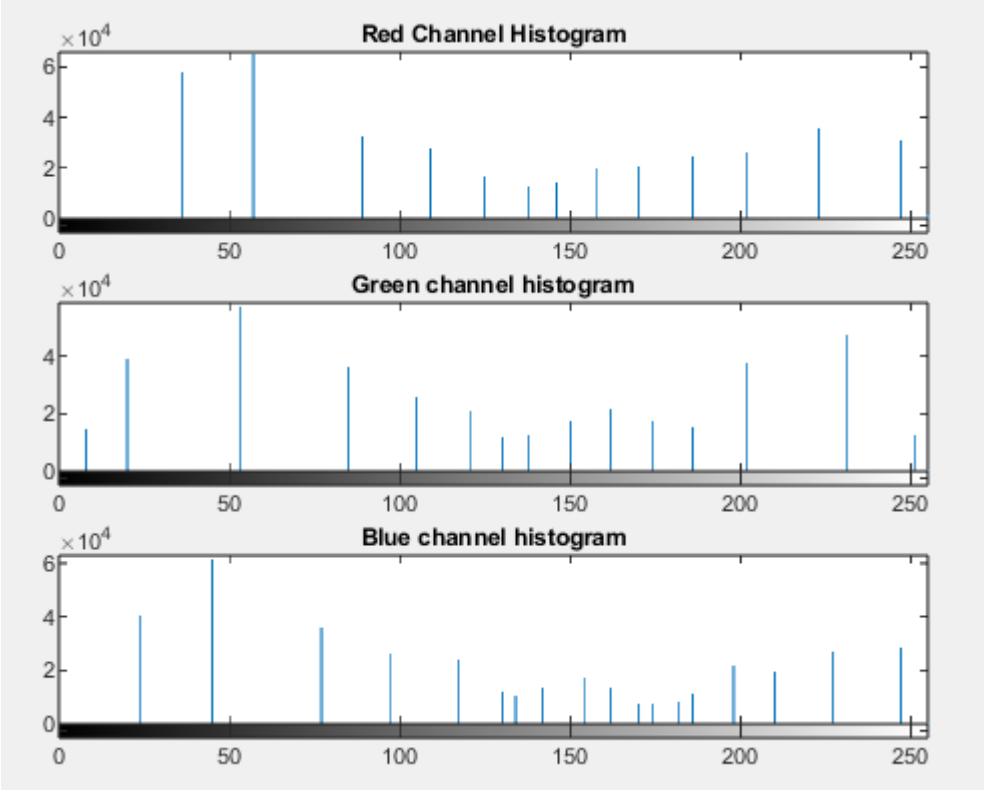


Figure 2.189: Histogram of experimental result with max PSNR

For the case of the maximum experimental PSNR, the same picture as that of darkness level 4.5 is obtained. The experimental result is visually very close to the ground truth case, with the only negative we have to comment on are some color distortions that appear. These are due to the fact that, as we study the highest darkness level, the images are too dark so that it is impossible to fully recover the visual information. Nevertheless, the resulting image is impressive, if we take into account that we are studying the highest darkness level.

Overall commenting on the results of the histogram equalization, we notice that it gives very good results with the performance of the algorithm being resistant to the variations of the darkness level, as we observed in the above. One of the negatives observed is that the method has difficulty retrieving images, which the respective ground truth case has histograms with values that are not distributed over the entire available value range. In these cases, it gives images with strong color distortions, retrieving the wrong visual information. This is due to the nature of the algorithm, as its purpose is to take a low-contrast image and output an image whose pixel values are evenly distributed throughout the available value range. So, we understand that it is impossible to reproduce the results we mentioned above, leading to the wrong retrieval of visual information. On the other hand, in the cases where the ground truth images have histograms with values distributed throughout the available range, impressive results are obtained, even at high darkness levels, reflecting the strength and robustness of the method.

2.5 Single Scale Retinex

In this case, we will take a different approach to the LLIE problem. Until now we have applied pointwise transformations, using various mathematical functions on the pixel values. A disadvantage of this approach is that it does not take into account possible correlations between neighboring pixels. For this reason we will try to depart from this logic and apply spatial filters using the act of convolution, thus taking into account such possible correlations.

The algorithm that we will study first belongs to a large family of algorithms, which are based on the Retinex Theory developed by Land and McCann [25], and studies the perception of color by the human eye and the modeling of color invariance. The main goal of the theory is to calculate the reflective nature of an object in the scene, removing the effect of brightness. According to the theory, the human visual system keeps only the information related to the characteristics of the objects in the scene, such as the reflection coefficient, and for this reason it is easy for us to adapt regardless of the brightness or changes in the brightness. The model derived from the theory says that an image can be expressed as follows:

$$I(x,y) = R(x,y)L(x,y)$$

where R(x,y) is the reflectivity coefficient, and expresses the reflective nature of the surface of an object, and L(x,y) is the luminance factor that depends on the luminance conditions of the scene. Based on these we understand that R(x,y) determines the nature of the image, while L(x,y) determines the dynamic range of the image. According to the theory, if we can estimate L(x,y) from the image, then we can isolate R(x,y). This will lead us to a result that is independent of the amount of brightness, thereby enhancing the image.

The above logic can be applied to enhance dark images, so a way to calculate R(x,y) remains to be found. One of the many algorithms that have been proposed for this purpose is the Single Scale Retinex, proposed by Jobson et al.[26]. According to this algorithm, R(x,y) can be calculated as follows:

$$R_i(x,y) = \log[I_i(x,y)] - \log[G(x,y) * I_i(x,y)]$$

where R(x,y) is the refrection image, I(x,y) is the input image, i is for the color channels (I = {R, G, B}), * denotes the convolution operation and G(x,y) is the surround function. The authors use a Gaussian distribution as the surround function, which has the form:

$$G(x,y) = Ke^{-\left(\frac{x^2+y^2}{c^2}\right)}$$

where c is the scale parameter, and K is a normalization constant that ensures that the function satisfies the criterion:

$$\iint G(x,y)dxdy = 1$$

The final result obtained from the above must be mapped to the range [0,255], something that is done with a simple linear transformation, which has the form:

$$R_i'(x,y) = 255[\frac{R_i(x,y) - R_{min}}{R_{max} - R_{min}}]$$

Practically, we create a 2D mask with the same dimensions as these images (disregarding the third dimension with the color bands) and Gaussian distribution values, and apply convolution of this mask with each channel of the image. The shape of the mask, also called the surround function, can be seen in the image below.

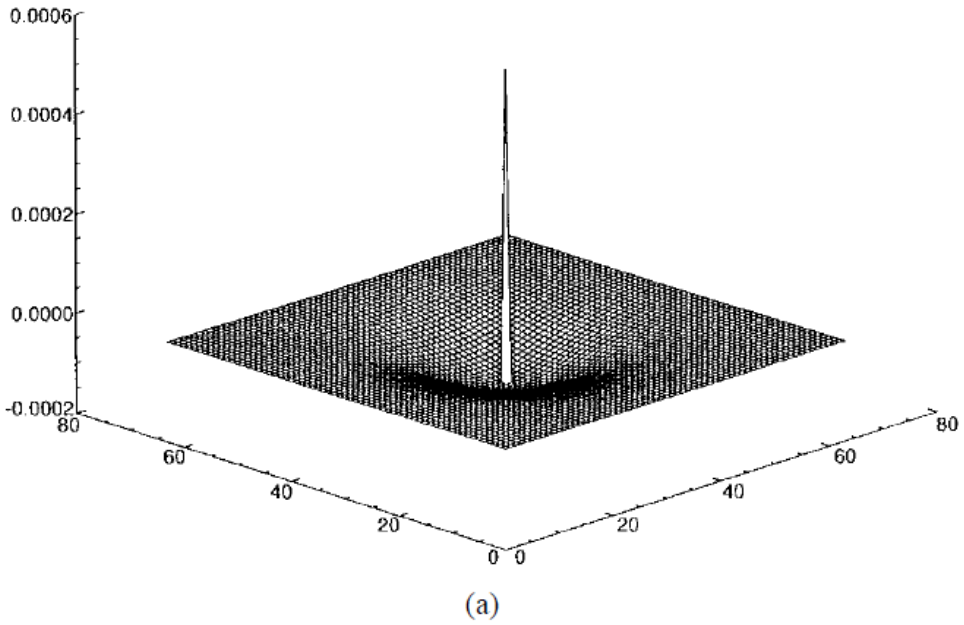


Figure 2.190: Surround function with c = 30 [26]

This function can take different forms depending on the value of the constant c. Obviously there is no ideal value of the constant, it depends on the problem we are studying. After applying the convolution, we calculate the logarithms, we also apply the linear transformation, which we mentioned above, and the final form of the coefficient R is obtained. R is the final enhanced image, which we will compare with the ground truth images, and it is an image that is independent of the lighting conditions.

The method we have described is more complicated in terms of implementation than what we have done so far. It is implemented with the function shown in figure B.2.17 of appendix B. We will apply this function to each image of the data set, which is done with the scripts shown in figures B.2.18 and B.2.19 of appendix B. As we said, there is no ideal value of the constant c, for this reason we use 3 different values, a small, a medium and a large value, in order to evaluate the effect of the constant on the performance of the algorithm. After applying these scripts, we have the experimental results at our disposal. We will use them to calculate performance metrics, based on the value of which we will evaluate the performance of the algorithm. This is done with the script presented in Figure B.2.20 of Appendix B. After its application we have values of the metrics per image, which will be used to build summary tables with the minimum/maximum/average value, per darkness level, constant value c

and set, presented next. In addition, we will also construct line charts of these values per darkness level, for PSNR and SSIM. Finally, we will construct line charts with the average value of PSNR and SSIM, for the original images and for the experimental results so that we can quantify how much the result improves and which value of the constant c is more appropriate.

Training Set

c = 10

MSE					
Level	3.0	3.5	4.0	4.5	5.0
MIN	665,0725	668,5667	787,96	569,1219	1015,734
MAX	17684,25	17719,33	16972,41	18009,17	17169,11
AVERAGE	5103,033	5229,953	5259,419	5167,714	4885,421
	PSNR(dB)				
Level	3.0	3.5	4.0	4.5	5.0
MIN	5,654936	5,646332	5,833368	5,575866	5,783326
MAX	19,90211	19,87936	19,16576	20,57875	18,063
AVERAGE	11,58772	11,47048	11,42904	11,46914	11,65905
	SSIM				
Level	3.0	3.5	4.0	4.5	5.0
MIN	-0,15197	-0,45574	-0,54387	-0,49599	-0,03388
MAX	0,788078	0,754543	0,770589	0,702689	0,619275
AVERAGE	0,334179	0,321747	0,303168	0,280441	0,242273
	MV				
Level	3.0	3.5	4.0	4.5	5.0
MIN	84,96666	87,15809	89,15464	81,3479	75,70744
MAX	197,6954	200,0791	187,466	178,3255	172,9067
AVERAGE	142,4572	143,0743	142,8123	141,3077	138,2819
	STD				
Level	3.0	3.5	4.0	4.5	5.0
MIN	42,81076	40,52304	41,78316	45,93321	65,11519
MAX	487,8006	520,2408	564,6657	581,5751	464,9702
AVERAGE	212,9648	213,6606	218,6065	228,4065	238,1422
	BRISQUE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	6,345484	9,577115	8,348036	9,684713	14,49088
MAX	53,72307	56,14981	58,0542	58,96327	58,90108
AVERAGE	32,54928	34,67815	37,46238	40,2265	42,32244

	NIQE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	2,451916	2,557915	3,076157	3,380398	3,333449
MAX	27,08438	26,2094	8,73375	9,039466	9,068266
AVERAGE	3,96099	4,251106	4,60689	5,081881	5,616927

Table 2.28: SSR results with c=10 on training set

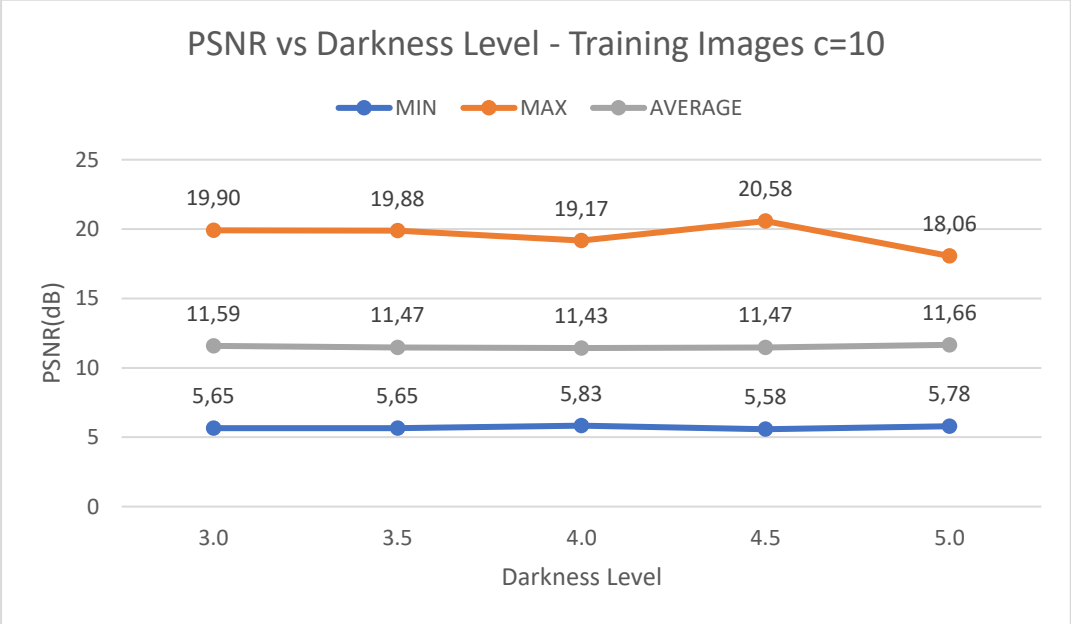


Figure 2.191: Experimental results PSNR vs Darkness Level for training images

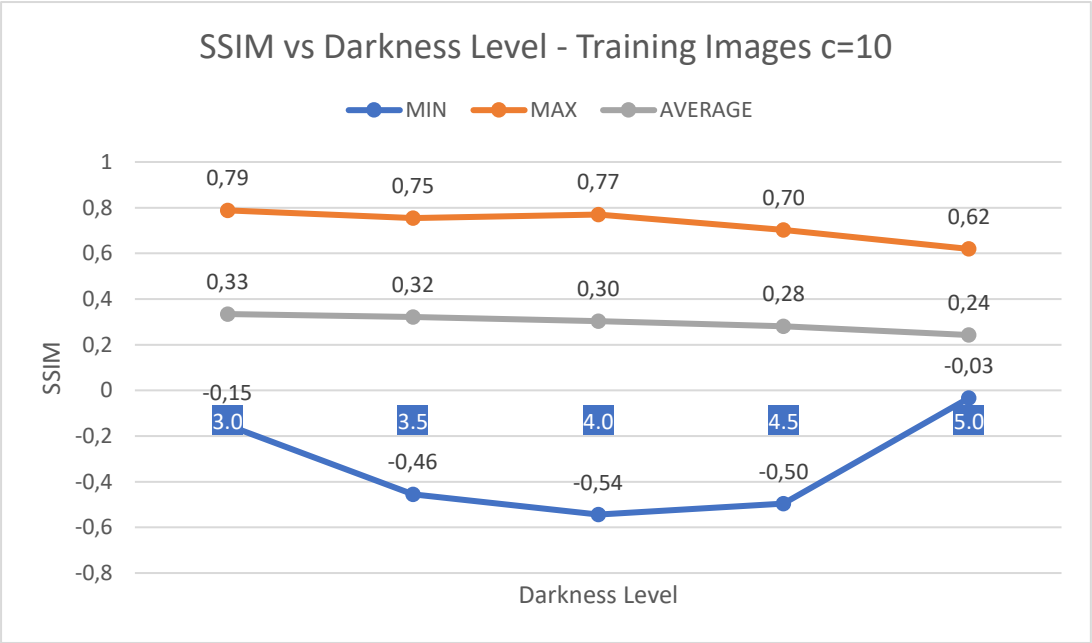


Figure 2.192: Experimental results SSIM vs Darkness Level for training images

For c=10 the values of the metrics improve little, which indicates that small values of the constant c are not suitable for enhancing LL images. What is worth commenting on is that increasing the darkness level does not seem to affect the performance of the algorithm, as the changes in the metrics from level to level are minimal.

c = 120

MSE					
Level	3.0	3.5	4.0	4.5	5.0
MIN	218,03	235,6003	255,956	337,5466	540,311
MAX	13485,81	12750,26	11954,9	10911,53	9032,903
AVERAGE	3311,263	3176,813	2996,014	2812,846	2720,194
	PSNR(dB)				
Level	3.0	3.5	4.0	4.5	5.0
MIN	6,832034	7,075613	7,355344	7,751948	8,57253
MAX	24,74564	24,40905	24,04915	22,84747	20,80437
AVERAGE	13,97501	14,15994	14,34833	14,47918	14,4895
	SSIM				
Level	3.0	3.5	4.0	4.5	5.0
MIN	-0,19865	-0,00447	-0,02796	-0,00581	-0,0676
MAX	0,842349	0,859943	0,846912	0,804219	0,77709
AVERAGE	0,504164	0,487171	0,459649	0,41515	0,355029
	MV				
Level	3.0	3.5	4.0	4.5	5.0
MIN	70,57968	61,64816	51,29523	40,76593	45,28514
MAX	224,0603	218,3537	204,2371	190,0457	179,4277
AVERAGE	134,2252	130,946	126,6804	121,3403	115,3027
	STD				
Level	3.0	3.5	4.0	4.5	5.0
MIN	75,57473	78,30564	63,78141	65,60575	58,90308
MAX	963,797	933,9817	895,0092	887,7652	791,0142
AVERAGE	360,4645	353,5516	349,6594	348,4279	351,556
	BRISQUE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	8,35327	4,095686	6,280565	5,089578	8,70187
MAX	48,51535	49,13184	54,29911	54,68566	57,18713
AVERAGE	29,7353	28,685	28,71063	30,15051	32,47091
	NIQE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	2,117876	2,165019	2,446189	2,558543	2,958166
MAX	25,60202	25,89071	7,755804	7,782287	8,664149
AVERAGE	3,634559	3,828597	4,066188	4,41904	4,877193

Table 2.29: SSR results with c=120 on training set

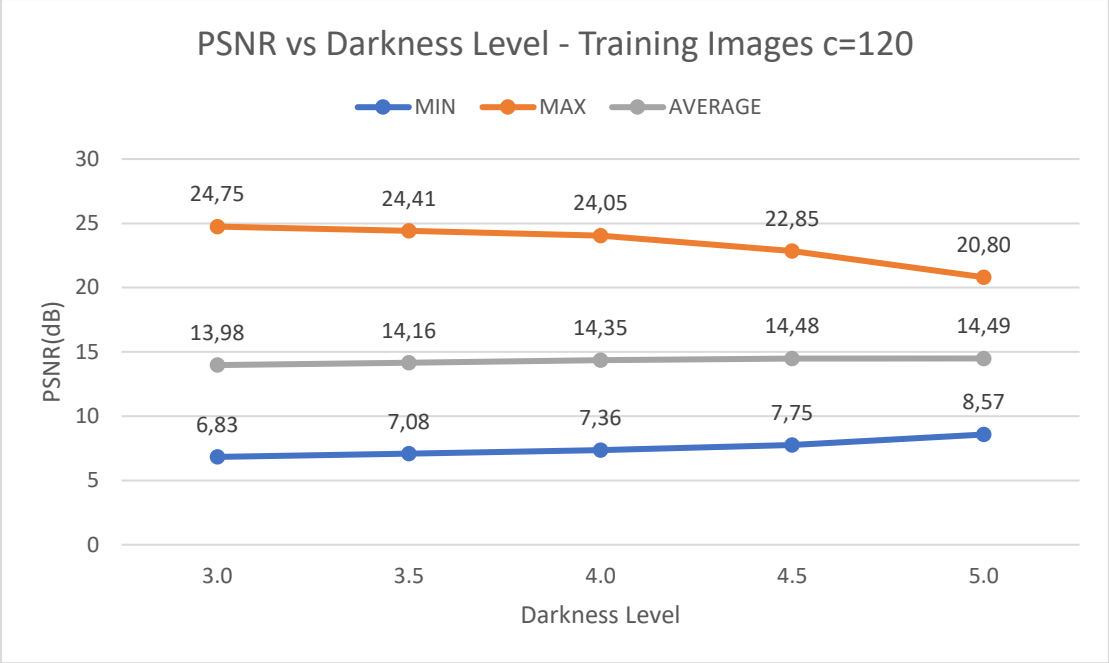


Figure 2.193: Experimental results PSNR vs Darkness Level for training images

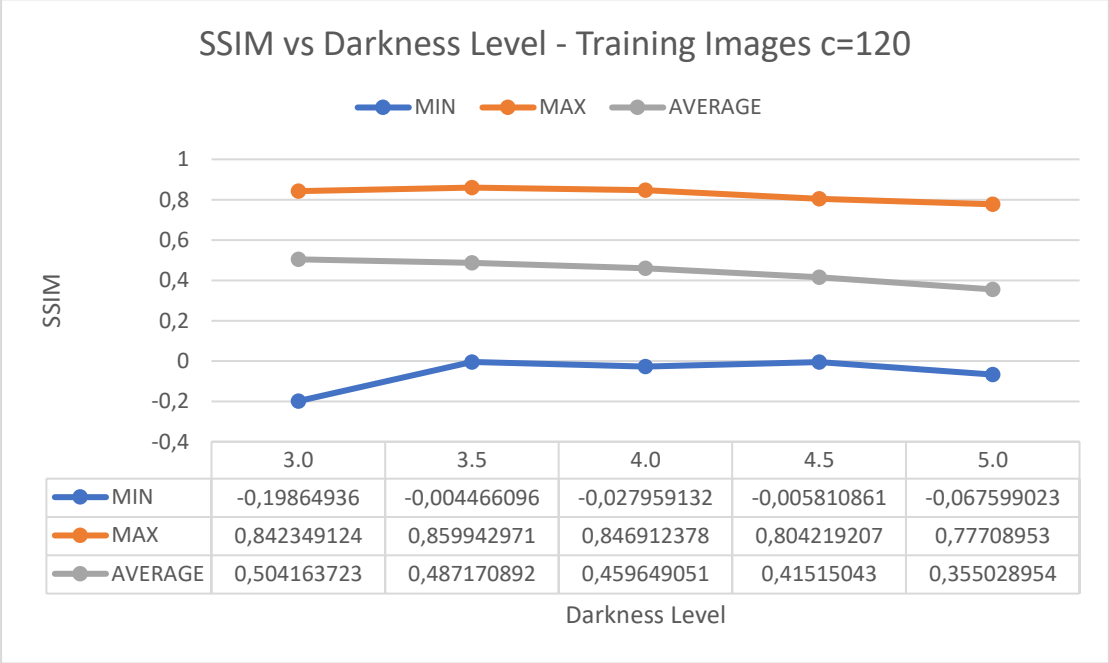


Figure 2.194: Experimental results SSIM vs Darkness Level for training images

For c=120 the values of the metrics improve noticeably, which shows us that large values of the constant perform better. Here we can also observe that the increase in the darkness level does not seem to affect the performance of the algorithm, as the changes in the values of the metrics

are minimal. Trying to further increase the performance of the method we will further increase the value of the constant c.

c = 400

MSE					
Level	3.0	3.5	4.0	4.5	5.0
MIN	136,4565	225,9383	290,3078	226,6083	393,7303
MAX	14258,36	12868,81	11466,55	8946,979	7741,064
AVERAGE	2848,523	2638,634	2415,037	2236,694	2232,624
	PSNR(dB)				
Level	3.0	3.5	4.0	4.5	5.0
MIN	6,590108	7,035419	7,536475	8,61404	9,242797
MAX	26,78086	24,5909	23,50222	24,57804	22,17882
AVERAGE	14,95259	15,29433	15,53013	15,6157	15,429
	SSIM				
Level	3.0	3.5	4.0	4.5	5.0
MIN	-0,18055	-0,04623	-0,01665	-0,06182	-0,02408
MAX	0,87093	0,881541	0,849421	0,818976	0,77949
AVERAGE	0,550661	0,534302	0,499555	0,445963	0,377521
	MV				
Level	3.0	3.5	4.0	4.5	5.0
MIN	57,21979	47,82103	39,29376	30,53282	38,73608
MAX	226,6883	220,3872	205,393	189,9874	183,0885
AVERAGE	131,8968	127,1471	121,253	114,3453	107,133
	STD				
Level	3.0	3.5	4.0	4.5	5.0
MIN	44,75635	47,47545	47,16199	50,09331	69,29461
MAX	1452,56	1403,184	1347,318	1248,426	1084,216
AVERAGE	434,7157	427,9926	422,7907	419,0516	417,0082
	BRISQUE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	7,131121	3,211743	6,125323	6,185697	9,250425
MAX	50,37515	51,07172	54,55468	55,01573	57,19508
AVERAGE	30,31216	29,29636	29,5549	31,09955	33,60713
	NIQE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	2,182783	2,245904	2,526631	2,870625	3,028016
MAX	25,55784	25,82074	7,75195	7,93017	9,222819
AVERAGE	3,717744	3,923541	4,175947	4,542315	4,995335

Table 2.29: SSR results with c=400 on training set

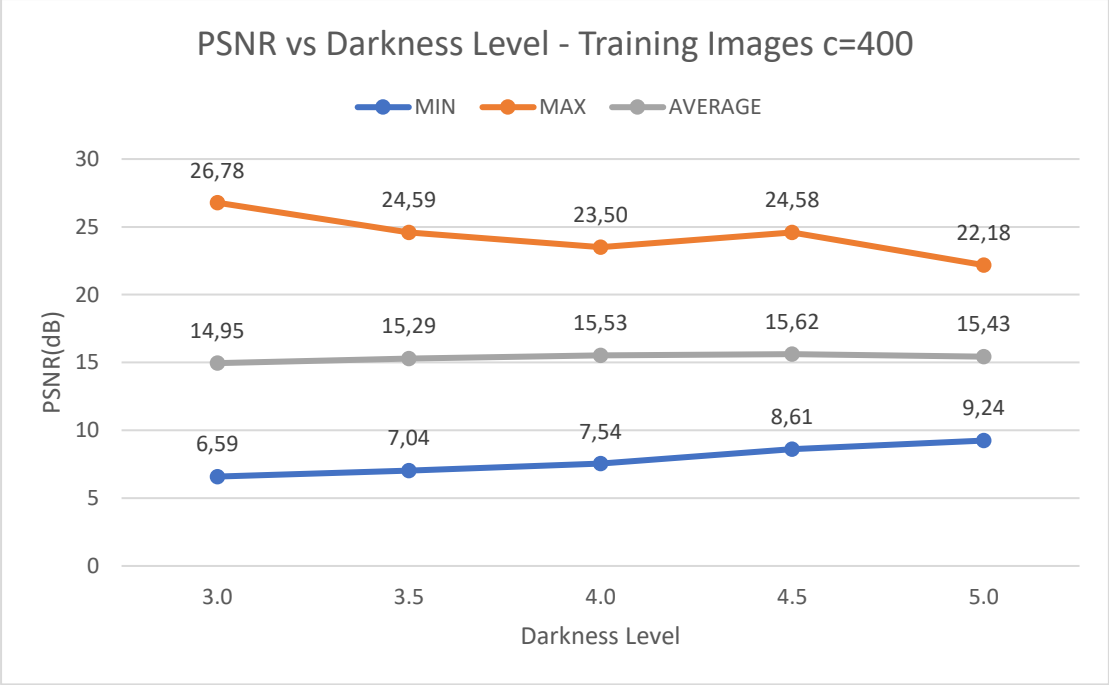
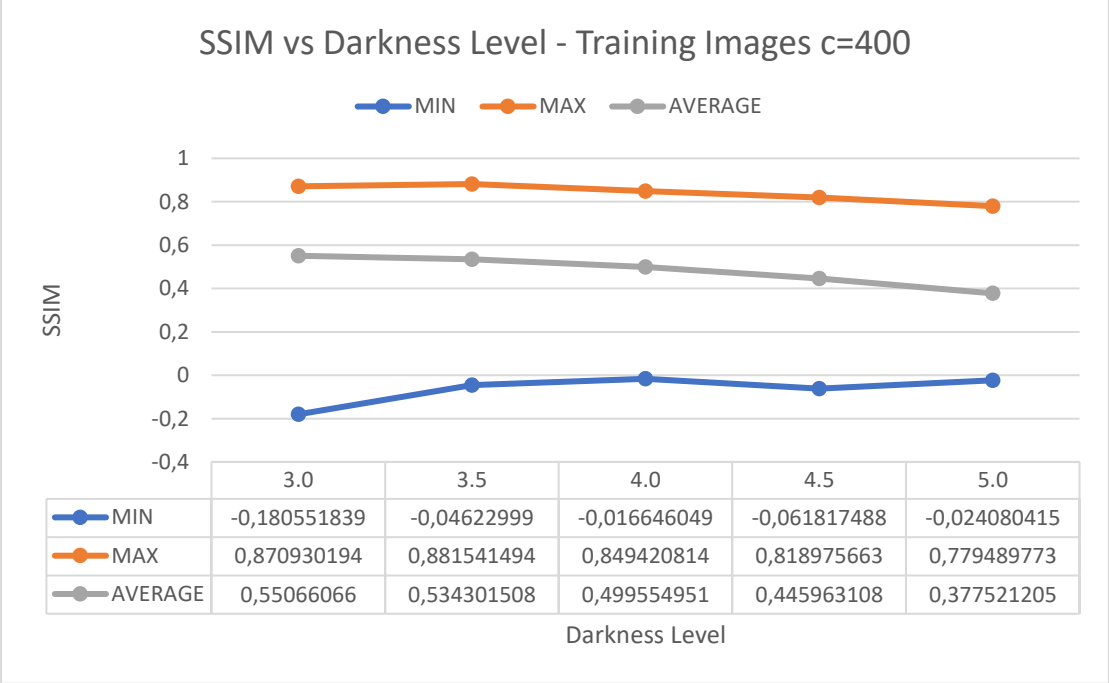


Figure 2.195: Experimental results PSNR vs Darkness Level for training images

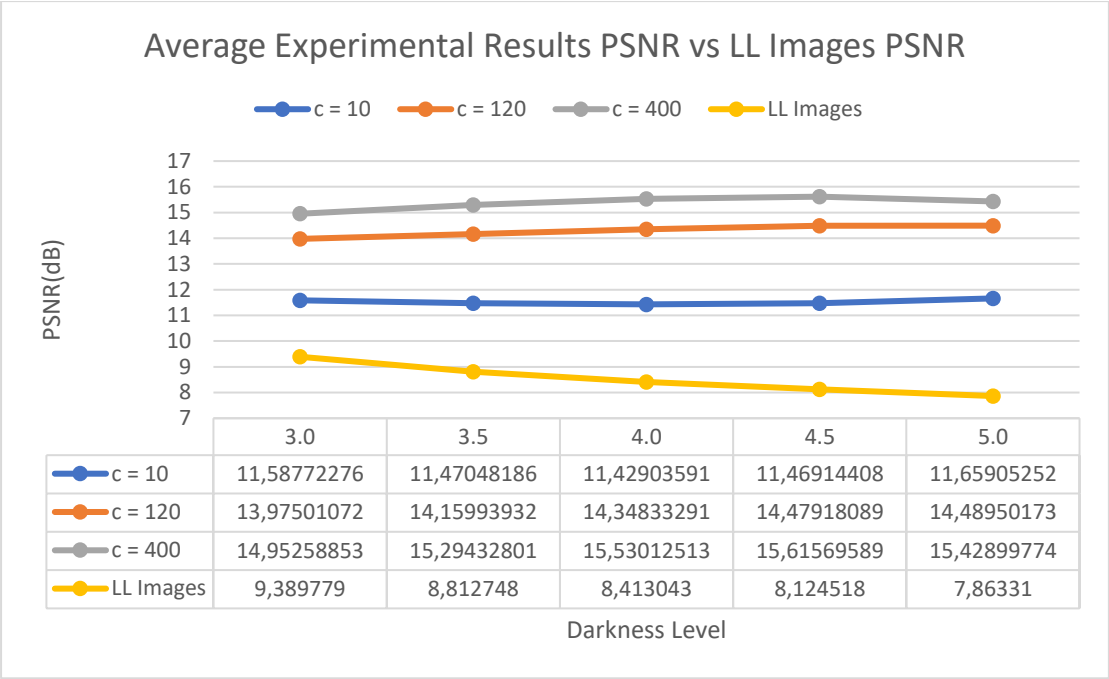


	3.0	3.5	4.0	4.5	5.0
MIN	-0,180551839	-0,04622999	-0,016646049	-0,061817488	-0,024080415
MAX	0,870930194	0,881541494	0,849420814	0,818975663	0,779489773
AVERAGE	0,55066066	0,534301508	0,499554951	0,445963108	0,377521205

Figure 2.196: Experimental results SSIM vs Darkness Level for training images

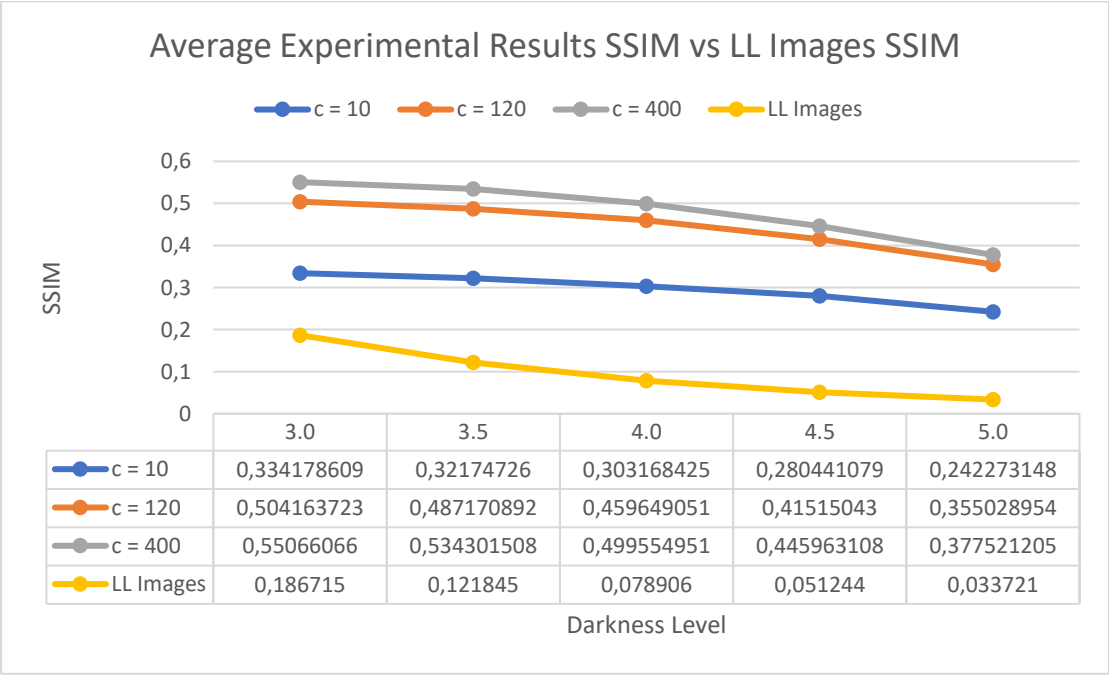
For a value of the constant c=400 the best results are obtained and, as we expected, the performance of the algorithm increases. We can also observe that the values of the metrics are not much affected by changes in the

darkness level, which indicates that the method is robust to changes in brightness.



	3.0	3.5	4.0	4.5	5.0
c = 10	11,58772276	11,47048186	11,42903591	11,46914408	11,65905252
c = 120	13,97501072	14,15993932	14,34833291	14,47918089	14,48950173
c = 400	14,95258853	15,29432801	15,53012513	15,61569589	15,42899774
LL Images	9,389779	8,812748	8,413043	8,124518	7,86331

Figure 2.197: Experimental results average PSNR vs LL Images PSNR training set



	3.0	3.5	4.0	4.5	5.0
c = 10	0,334178609	0,32174726	0,303168425	0,280441079	0,242273148
c = 120	0,504163723	0,487170892	0,459649051	0,41515043	0,355028954
c = 400	0,55066066	0,534301508	0,499554951	0,445963108	0,377521205
LL Images	0,186715	0,121845	0,078906	0,051244	0,033721

Figure 2.198: Experimental results average SSIM vs LL Images SSIM training set

The above 2 diagrams confirm what we have mentioned so far. For c=10 the worst results are obtained, as the values of the metrics improve little. By increasing the value of the constant to 120 the results improve significantly with both PSNR and SSIM increasing, which means that the quality of the experimental images becomes better. For a value of the constant 400 the best results of the method are obtained, giving the greatest improvement of the metrics, which means that the method works better for large values of the constant. Finally, what is observed in all cases is that with the change of the darkness level, the values of the quality metrics are not strongly affected, which indicates that the method presents a robustness to the changes of the darkness level.

Validation Set

c = 10

MSE					
Level	3.0	3.5	4.0	4.5	5.0
MIN	1120,861	1136,636	1220,439	1286,684	1387,72
MAX	11827,88	13695,22	13859,43	14608,67	14516,86
AVERAGE	4743,751	4867,5	4898,797	4887,402	4696,366
	PSNR(dB)				
Level	3.0	3.5	4.0	4.5	5.0
MIN	7,401733	6,765115	6,71335	6,484697	6,512077
MAX	17,63529	17,57459	17,26564	17,03609	16,70778
AVERAGE	11,94941	11,82721	11,74782	11,73686	11,88849
	SSIM				
Level	3.0	3.5	4.0	4.5	5.0
MIN	0,055716	-0,09136	0,053598	0,026899	0,016744
MAX	0,606157	0,573933	0,569404	0,504718	0,434409
AVERAGE	0,331513	0,289061	0,290769	0,261503	0,247608
	MV				
Level	3.0	3.5	4.0	4.5	5.0
MIN	102,0629	93,97228	91,04463	85,05681	76,88142
MAX	196,3755	187,0055	167,4845	163,0169	162,6015
AVERAGE	141,238	141,5049	141,2411	139,9921	138,2873
	STD				
Level	3.0	3.5	4.0	4.5	5.0
MIN	95,96961	61,06201	67,95074	114,2522	111,416
MAX	518,8461	498,8457	495,6769	444,6703	352,6008
AVERAGE	218,941	214,1372	219,347	230,9905	246,0822

	BRISQUE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	10,89029	17,53461	11,23896	11,84823	22,44026
MAX	51,67118	50,84251	53,15401	55,9085	54,57432
AVERAGE	33,71082	32,75258	35,35772	39,877	42,90613
	NIQE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	2,998674	3,119284	3,294057	3,701252	4,107469
MAX	5,18835	5,764339	7,407522	7,423493	7,909965
AVERAGE	3,888575	4,108558	4,470427	4,914164	5,440554

Table 2.30: SSR results with c=10 on validation set

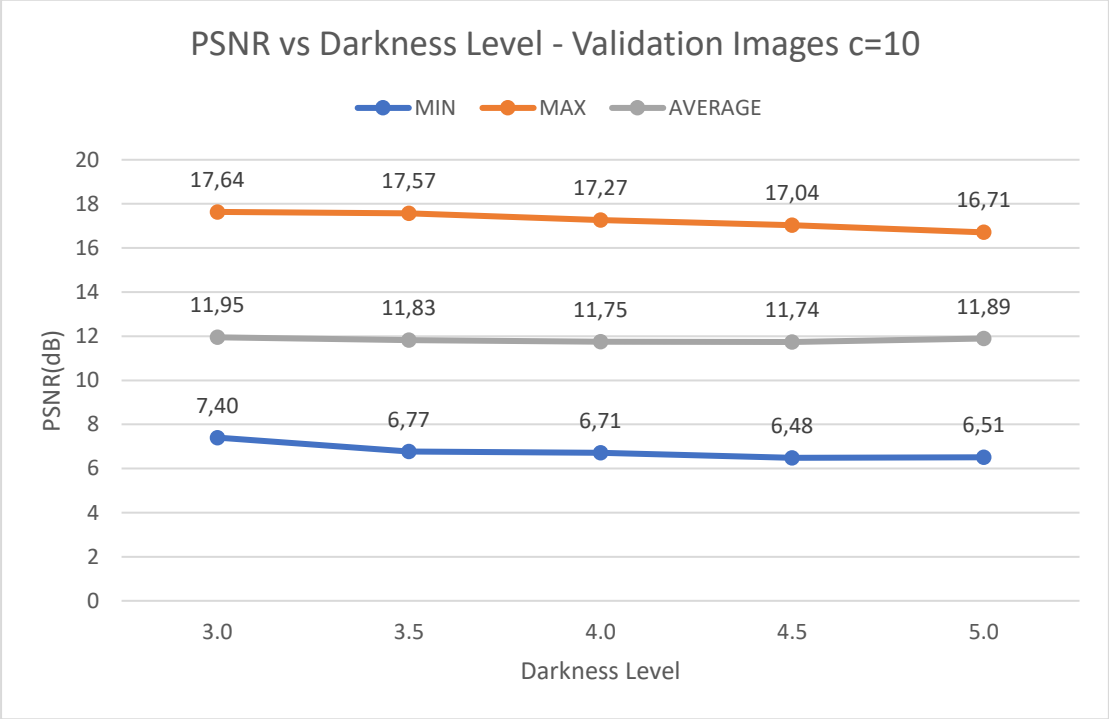


Figure 2.199: Experimental results PSNR vs Darkness Level for validation images

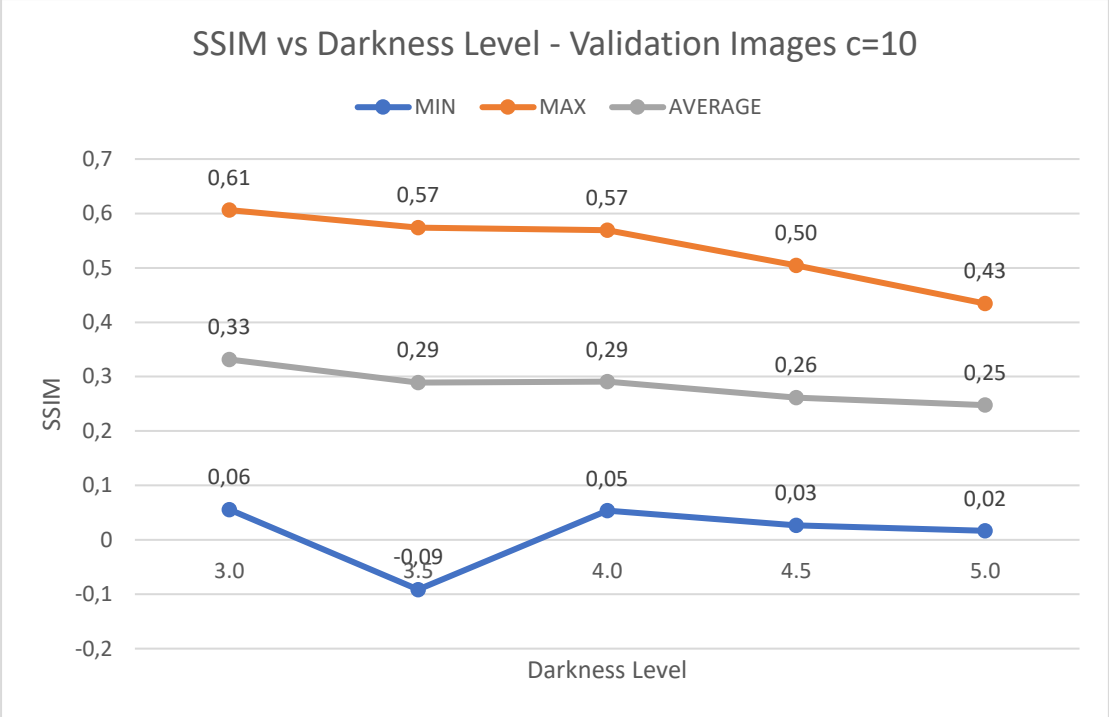


Figure 2.200: Experimental results SSIM vs Darkness Level for validation images

In the validation set for a constant value of 10, the same result is observed. Metric values are slightly improved, while not affected by changes in darkness level, as values change little with increasing darkness level.

c = 120

MSE					
Level	3.0	3.5	4.0	4.5	5.0
MIN	490,6158	491,8833	613,7923	709,3058	938,7886
MAX	6883,106	6909,615	6551,809	6772,402	7231,468
AVERAGE	2535,213	2475,623	2323,985	2266,112	2301,947
	PSNR(dB)				
Level	3.0	3.5	4.0	4.5	5.0
MIN	9,752959	9,736265	9,967191	9,823377	9,538539
MAX	21,22339	21,21218	20,25059	19,62247	18,40513
AVERAGE	14,84071	14,95595	15,04602	15,07053	15,02285
	SSIM				
Level	3.0	3.5	4.0	4.5	5.0
MIN	0,220902	0,240388	0,215231	0,137386	0,119603
MAX	0,640065	0,662773	0,645681	0,62625	0,54312
AVERAGE	0,494436	0,478924	0,465298	0,420243	0,375224

	MV				
Level	3.0	3.5	4.0	4.5	5.0
MIN	78,25968	72,95907	67,13712	56,02913	50,97294
MAX	189,1468	189,2707	179,6815	174,2548	167,1306
AVERAGE	129,2179	126,1003	121,6357	117,1295	113,4445
	STD				
Level	3.0	3.5	4.0	4.5	5.0
MIN	91,2851	91,64222	140,0873	106,5155	197,2576
MAX	629,984	666,9648	685,6692	666,2329	592,769
AVERAGE	362,7206	351,2691	345,4371	343,0534	351,3859
	BRISQUE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	15,02842	15,17382	11,58418	9,339646	8,835785
MAX	44,73476	44,56322	48,02092	46,56028	50,93568
AVERAGE	32,18605	31,25244	30,43138	29,77621	30,64459
	NIQE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	2,582034	2,886905	2,956562	3,471019	3,646924
MAX	5,275927	5,311754	6,107417	6,806349	6,775985
AVERAGE	3,579442	3,741029	4,009803	4,335317	4,72591

Table 2.31: SSR results with c=120 on validation set

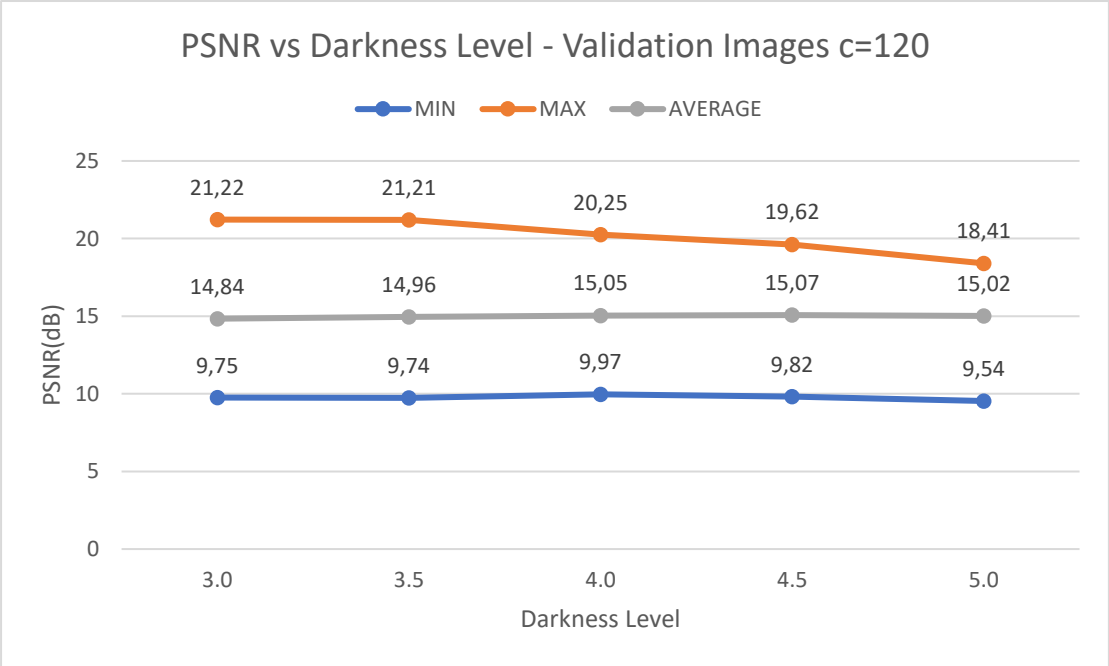


Figure 2.201: Experimental results PSNR vs Darkness Level for validation images

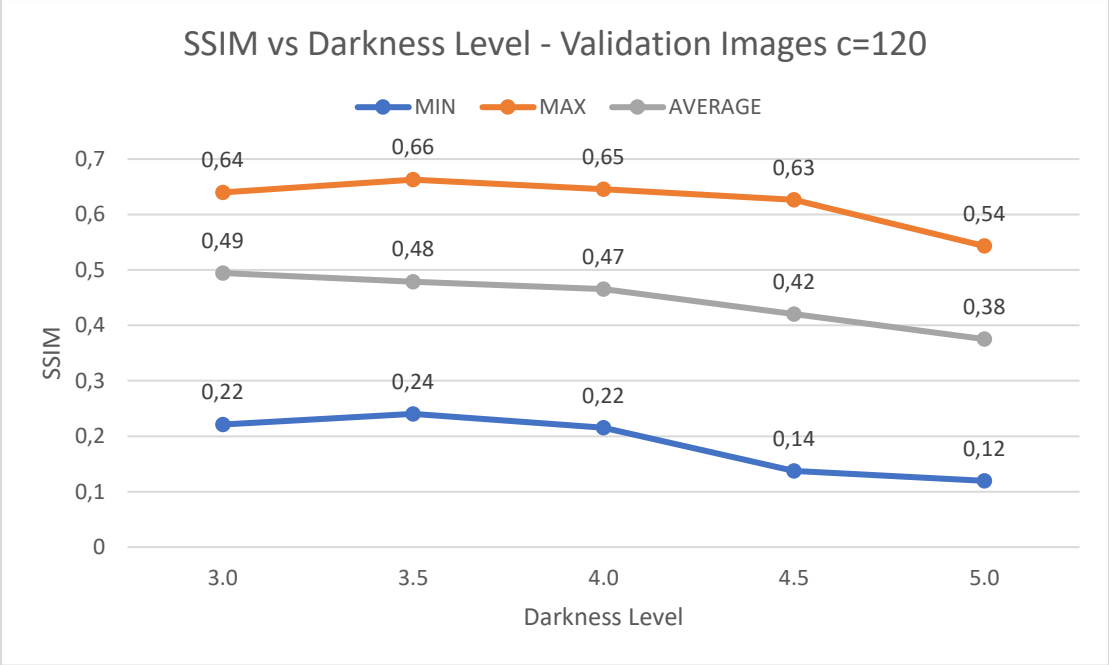


Figure 2.202: Experimental results SSIM vs Darkness Level for validation images

For c=120 the results improve more than the value 10, with the quality metric values improving noticeably. Here it is also observed that the increase of the darkness level affects the performance of the algorithm little.

c = 400

			MSE		
Level	3.0	3.5	4.0	4.5	5.0
MIN	314,9185	362,2658	571,7531	661,3695	817,1816
MAX	6559,855	5988,816	5158,321	4690,48	6068,914
AVERAGE	2128,25	2029,028	1892,113	1798,513	1906,452
			PSNR(dB)		
Level	3.0	3.5	4.0	4.5	5.0
MIN	9,961862	10,35739	11,00572	11,41863	10,29969
MAX	23,14882	22,54053	20,55872	19,92636	19,00762
AVERAGE	15,79225	15,98274	16,11191	16,18117	15,93109
			SSIM		
Level	3.0	3.5	4.0	4.5	5.0
MIN	0,253195	0,241165	0,236888	0,093702	0,081437
MAX	0,731272	0,723638	0,73018	0,723783	0,637829
AVERAGE	0,538236	0,533056	0,508235	0,458337	0,399252

	MV				
Level	3.0	3.5	4.0	4.5	5.0
MIN	69,68754	61,55004	54,69286	41,49248	36,5903
MAX	189,1467	185,3203	179,2269	172,5822	160,2555
AVERAGE	127,1889	122,989	117,0598	110,8203	105,698
	STD				
Level	3.0	3.5	4.0	4.5	5.0
MIN	102,8901	97,18159	139,1741	181,0856	223,9553
MAX	795,8389	887,3056	921,0668	904,8474	687,3111
AVERAGE	424,4696	413,3726	408,8238	406,291	411,9462
	BRISQUE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	17,96417	15,91031	11,18123	9,36628	10,39591
MAX	44,88108	44,59353	47,42726	47,79321	51,70994
AVERAGE	32,54338	31,84641	31,12422	30,73773	31,37985
	NIQE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	2,619685	2,904302	2,987092	3,471582	3,701899
MAX	5,34221	5,398016	6,827672	6,814667	7,480987
AVERAGE	3,655544	3,831344	4,109938	4,443149	4,861035

Table 2.32: SSR results with c=400 on validation set

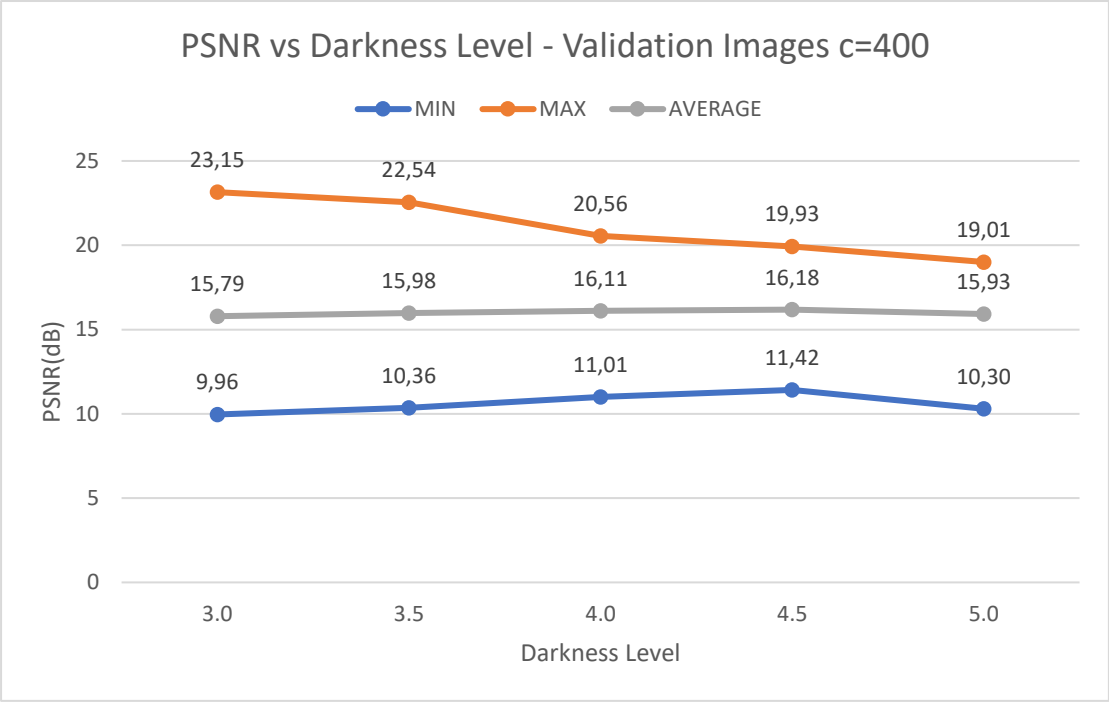


Figure 2.203: Experimental results PSNR vs Darkness Level for validation images

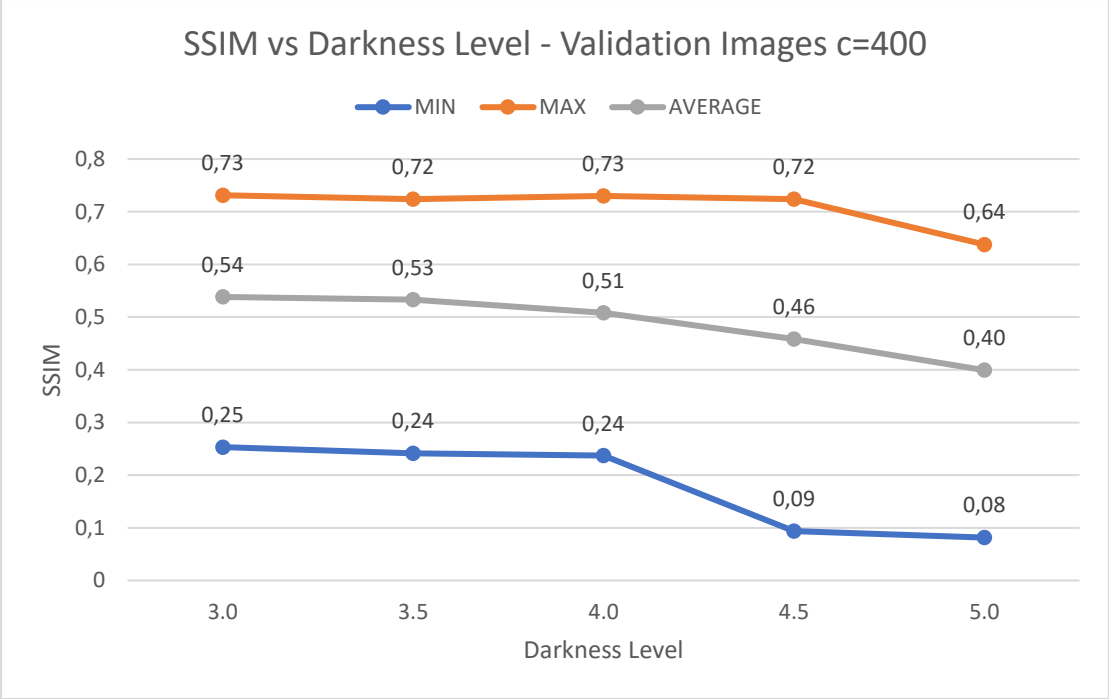
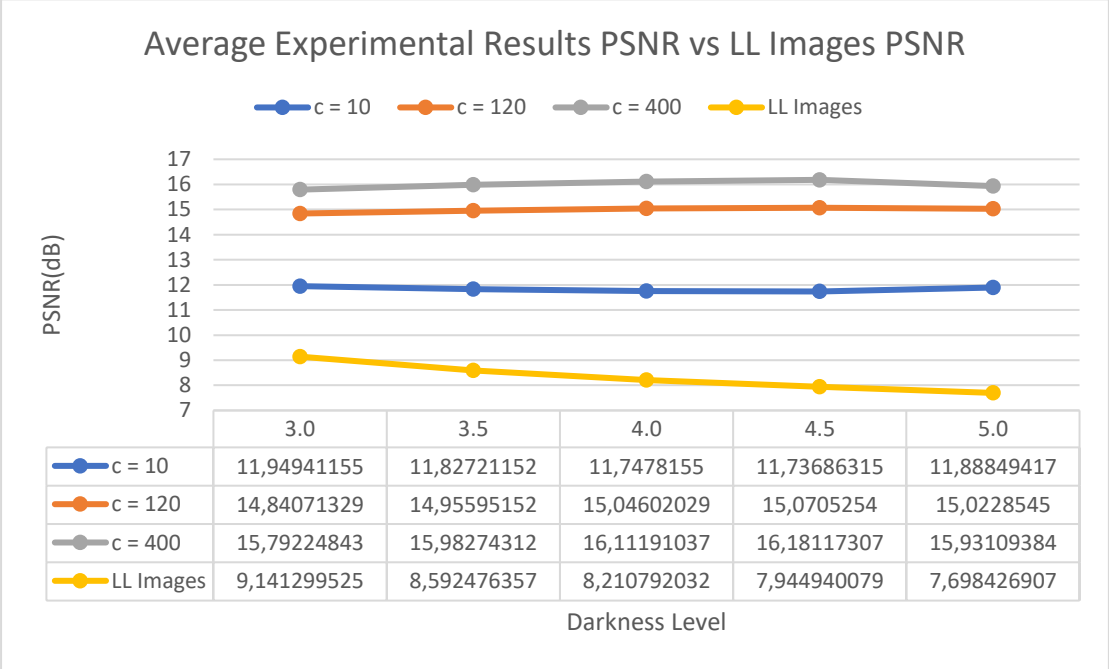


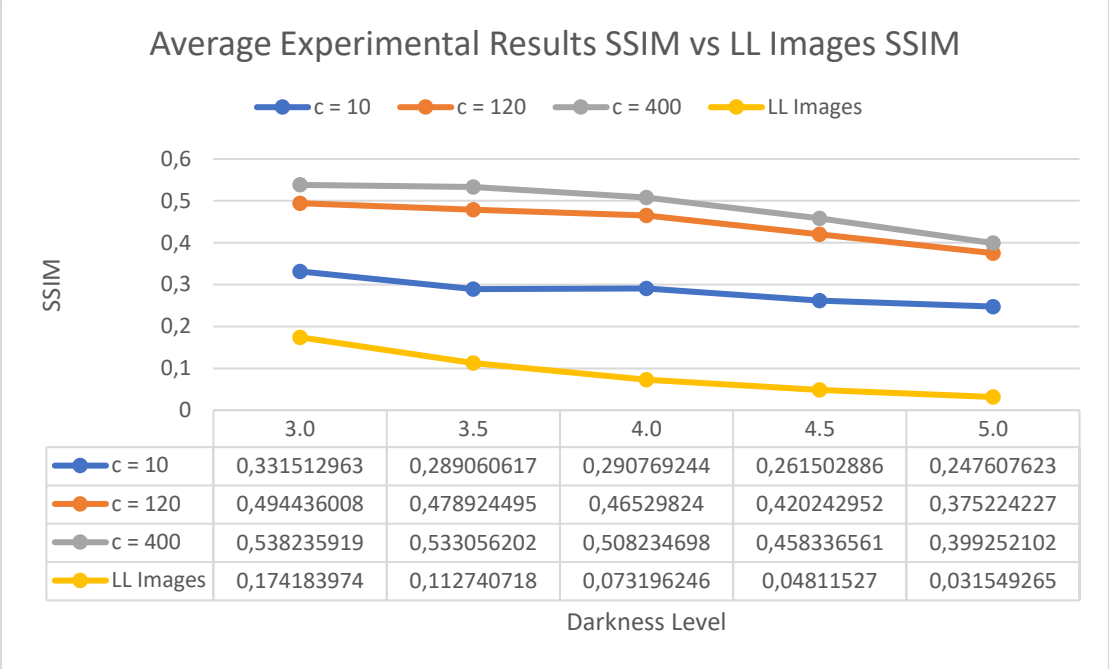
Figure 2.204: Experimental results SSIM vs Darkness Level for validation images

For a value of c=400 the best results are obtained, compared to the other two values, with the values of the quality metrics improving even further. And here it is observed that the increase of the darkness level has little effect on the performance of the method.



	3.0	3.5	4.0	4.5	5.0
c = 10	11,94941155	11,82721152	11,7478155	11,73686315	11,88849417
c = 120	14,84071329	14,95595152	15,04602029	15,0705254	15,0228545
c = 400	15,79224843	15,98274312	16,11191037	16,18117307	15,93109384
LL Images	9,141299525	8,592476357	8,210792032	7,944940079	7,698426907

Figure 2.205: Experimental results average PSNR vs LL Images PSNR validation set



	3.0	3.5	4.0	4.5	5.0
c = 10	0,331512963	0,289060617	0,290769244	0,261502886	0,247607623
c = 120	0,494436008	0,478924495	0,46529824	0,420242952	0,375224227
c = 400	0,538235919	0,533056202	0,508234698	0,458336561	0,399252102
LL Images	0,174183974	0,112740718	0,073196246	0,04811527	0,031549265

Figure 2.206: Experimental results average SSIM vs LL Images SSIM validation set

From figures 2.205 and 2.206 we can confirm what we have mentioned so far about the validation set. For a value constant of 10 the worst results are obtained, compared to the other 2 cases, as the values of the quality metrics improve little. For c=120 there is a noticeable improvement, while the best results are obtained for a constant value of 400, which indicates that the method works better for large values of the constant c. Furthermore, we have to comment that for all 3 constant cases, increasing the darkness level seems to have little effect on the result, as the changes in the metrics between the darkness levels are very small.

Test Set

c = 10

MSE					
Level	3.0	3.5	4.0	4.5	5.0
MIN	793,3752	854,0801	764,9538	890,5549	1094,199
MAX	15923,63	15935,55	16486,47	16825,83	16977,58
AVERAGE	5117,849	5203,936	5150,267	5068,238	5042,712
	PSNR(dB)				
Level	3.0	3.5	4.0	4.5	5.0
MIN	6,110382	6,107134	5,959528	5,871038	5,832046
MAX	19,13602	18,81582	19,29445	18,6342	17,73984
AVERAGE	11,53249	11,44863	11,4401	11,47188	11,48913
	SSIM				
Level	3.0	3.5	4.0	4.5	5.0
MIN	-0,15662	-0,12027	-0,01056	-0,08647	-0,13553
MAX	0,654343	0,597969	0,568566	0,542466	0,441656
AVERAGE	0,315422	0,318593	0,304157	0,283947	0,23884
	MV				
Level	3.0	3.5	4.0	4.5	5.0
MIN	94,72846	94,1652	84,7259	80,57854	75,04263
MAX	178,9696	180,6366	173,5919	168,6112	168,5379
AVERAGE	141,3639	141,9851	141,3764	139,7564	137,1447
	STD				
Level	3.0	3.5	4.0	4.5	5.0
MIN	99,49659	84,50468	72,18497	97,15905	98,4362
MAX	416,7925	408,1921	405,5388	414,4966	439,6693
AVERAGE	230,2727	226,4084	226,598	229,9208	246,9516
	BRISQUE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	8,608202	7,733923	10,20985	15,71469	22,6036
MAX	51,82803	54,12057	54,92622	55,80246	58,10413
AVERAGE	33,21045	35,64141	38,17466	39,86305	42,12886
	NIQE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	2,625362	2,759295	3,025844	3,436691	3,750788
MAX	6,666938	7,255504	8,072716	8,661863	9,355186
AVERAGE	3,965499	4,184468	4,566907	5,041816	5,54784

Table 2.33: SSR results with c=10 on test set

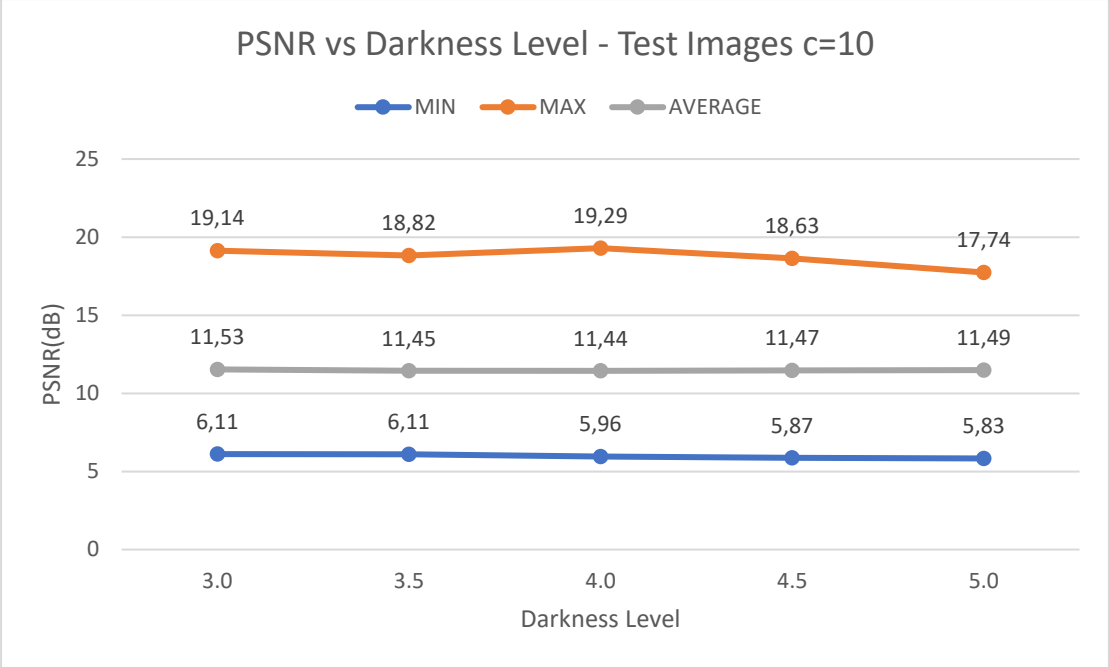


Figure 2.207: Experimental results PSNR vs Darkness Level for test images

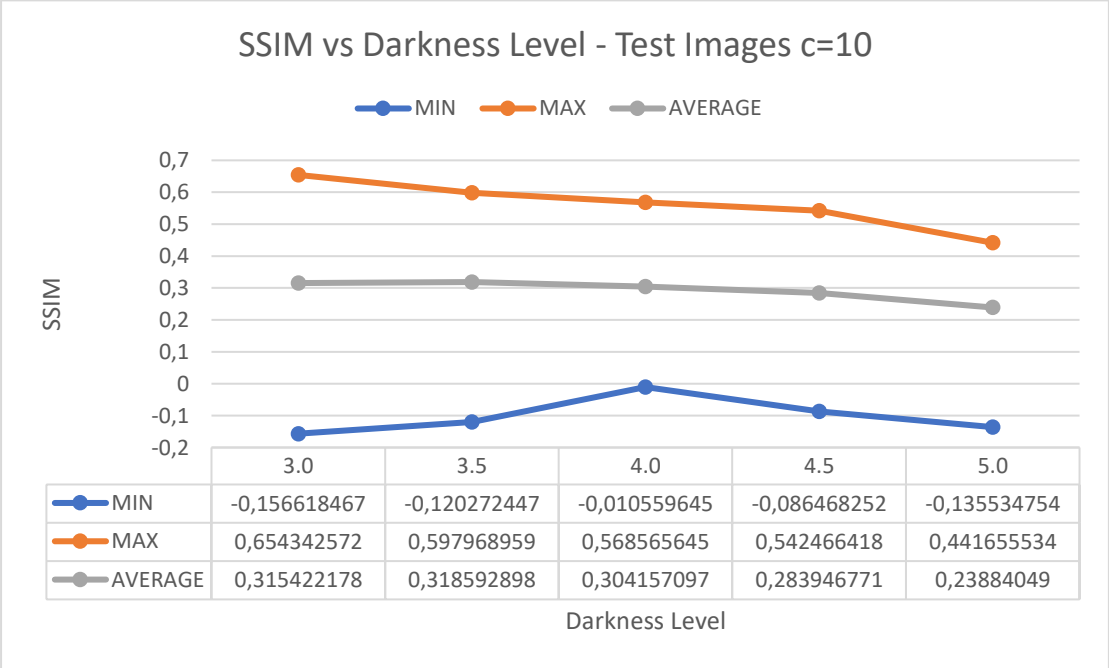


Figure 2.208: Experimental results SSIM vs Darkness Level for test images

In the test set for c=10 the values of the metrics improve slightly, and again it is observed that the increase of the darkness level affects the performance of the algorithm little.

c = 120

MSE					
Level	3.0	3.5	4.0	4.5	5.0
MIN	623,7859	428,9748	503,6118	671,3737	873,2447
MAX	12116,68	10413,52	9699,958	7622,337	6402,572
AVERAGE	2984,535	2834,202	2624,815	2427,107	2373,824
	PSNR(dB)				
Level	3.0	3.5	4.0	4.5	5.0
MIN	7,296966	7,954827	8,263105	9,309922	10,06726
MAX	20,18045	21,80649	21,10984	19,86116	18,71944
AVERAGE	14,3581	14,56672	14,80539	14,97838	14,90557
	SSIM				
Level	3.0	3.5	4.0	4.5	5.0
MIN	-0,02609	0,034913	-0,0211	0,07162	0,053441
MAX	0,835326	0,840405	0,807879	0,766564	0,665627
AVERAGE	0,505492	0,503125	0,472478	0,43676	0,368593
	MV				
Level	3.0	3.5	4.0	4.5	5.0
MIN	78,30545	77,83582	68,79508	65,87347	55,82128
MAX	185,9098	181,6558	174,9467	168,1169	161,4083
AVERAGE	132,0241	128,9872	124,5424	118,4373	110,883
	STD				
Level	3.0	3.5	4.0	4.5	5.0
MIN	152,2116	155,6314	144,3233	139,727	122,0124
MAX	716,7778	654,963	632,1502	627,576	618,0282
AVERAGE	381,8004	373,9314	365,5278	364,2542	354,8834
	BRISQUE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	13,34064	16,18832	10,71426	13,63365	11,31916
MAX	45,86726	45,35414	56,43702	55,29666	59,15522
AVERAGE	31,13592	30,45156	29,65138	30,92537	32,78221
	NIQE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	2,233963	2,450123	2,677566	3,172146	3,396478
MAX	6,310102	6,366701	7,328085	7,965271	10,40969
AVERAGE	3,599169	3,740196	4,02374	4,445925	4,93059

Table 2.34: SSR results with c=120 on test set

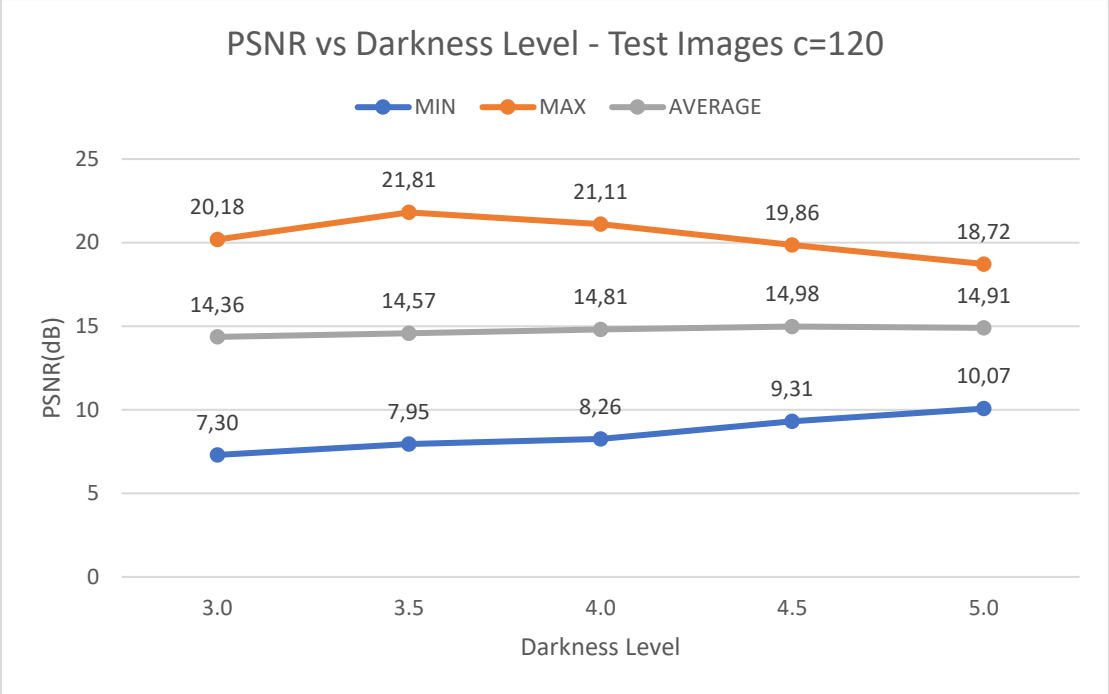


Figure 2.209: Experimental results PSNR vs Darkness Level for test images

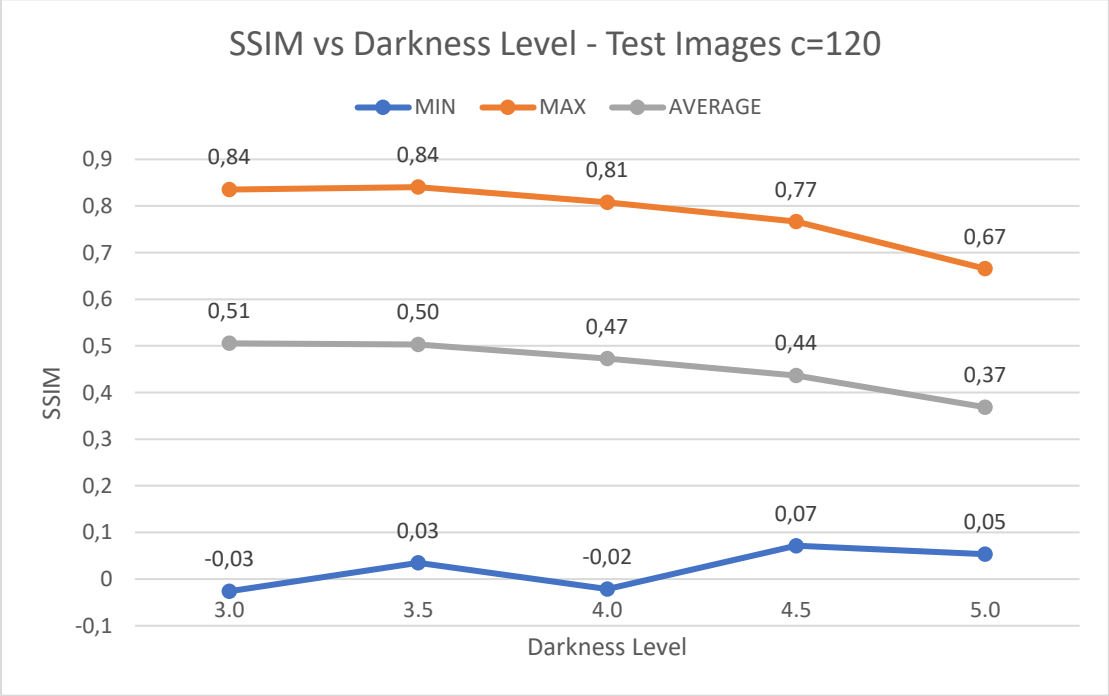


Figure 2.210: Experimental results SSIM vs Darkness Level for test images

For c=120 better results are obtained, with the values of the metrics improving noticeably. In this case we also notice that the increase in the darkness level does not affect the performance of the algorithm, as the changes in the values of the metrics are minimal.

c = 400

MSE					
Level	3.0	3.5	4.0	4.5	5.0
MIN	593,094	373,2489	435,7023	532,9734	502,9805
MAX	12334,22	9799,854	8303,944	6114,674	5485,475
AVERAGE	2590,489	2351,502	2103,758	1943,646	1943,74
	PSNR(dB)				
Level	3.0	3.5	4.0	4.5	5.0
MIN	7,219688	8,218608	8,937959	10,26707	10,73866
MAX	20,39957	22,41082	21,73891	20,86375	21,11529
AVERAGE	15,29788	15,72012	16,08146	16,18764	15,9054
	SSIM				
Level	3.0	3.5	4.0	4.5	5.0
MIN	0,025812	0,067173	0,014516	0,069533	0,004629
MAX	0,860339	0,856229	0,834269	0,808534	0,694088
AVERAGE	0,567147	0,558782	0,521125	0,469443	0,392017
	MV				
Level	3.0	3.5	4.0	4.5	5.0
MIN	78,68453	73,90002	61,9698	55,92087	45,83023
MAX	185,2713	179,1413	172,5438	163,9449	154,4602
AVERAGE	130,7205	125,9893	119,6509	112,4539	102,4706
	STD				
Level	3.0	3.5	4.0	4.5	5.0
MIN	142,8349	137,0771	134,3869	134,4727	131,3789
MAX	955,9174	896,8672	811,1136	805,7538	759,6219
AVERAGE	448,1803	441,0832	432,825	427,1154	404,4057
	BRISQUE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	14,85065	16,15412	6,683483	11,81652	12,25513
MAX	46,03544	47,14661	56,53913	53,42747	59,2341
AVERAGE	31,71538	31,16946	30,34369	31,5162	33,75235
	NIQE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	2,332474	2,545658	2,879262	3,23918	3,417642
MAX	6,315058	6,513556	7,272534	8,615225	10,53776
AVERAGE	3,660502	3,817886	4,119616	4,551726	5,036026

Table 2.35: SSR results with c=400 on test set

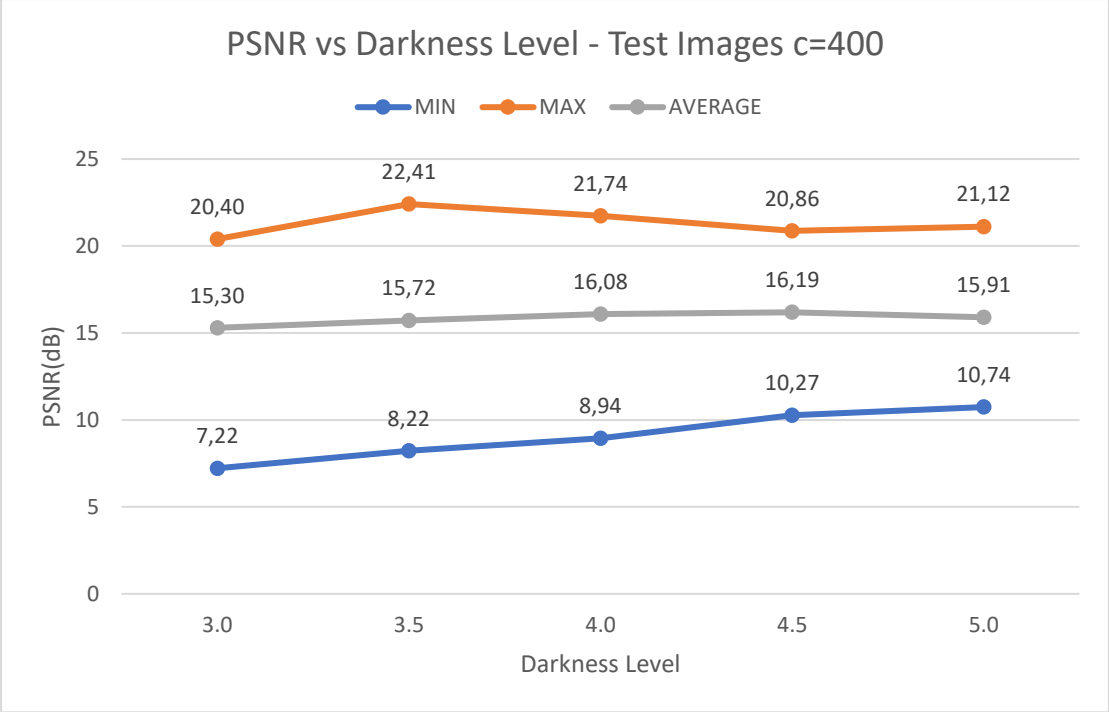


Figure 2.211: Experimental results PSNR vs Darkness Level for test images

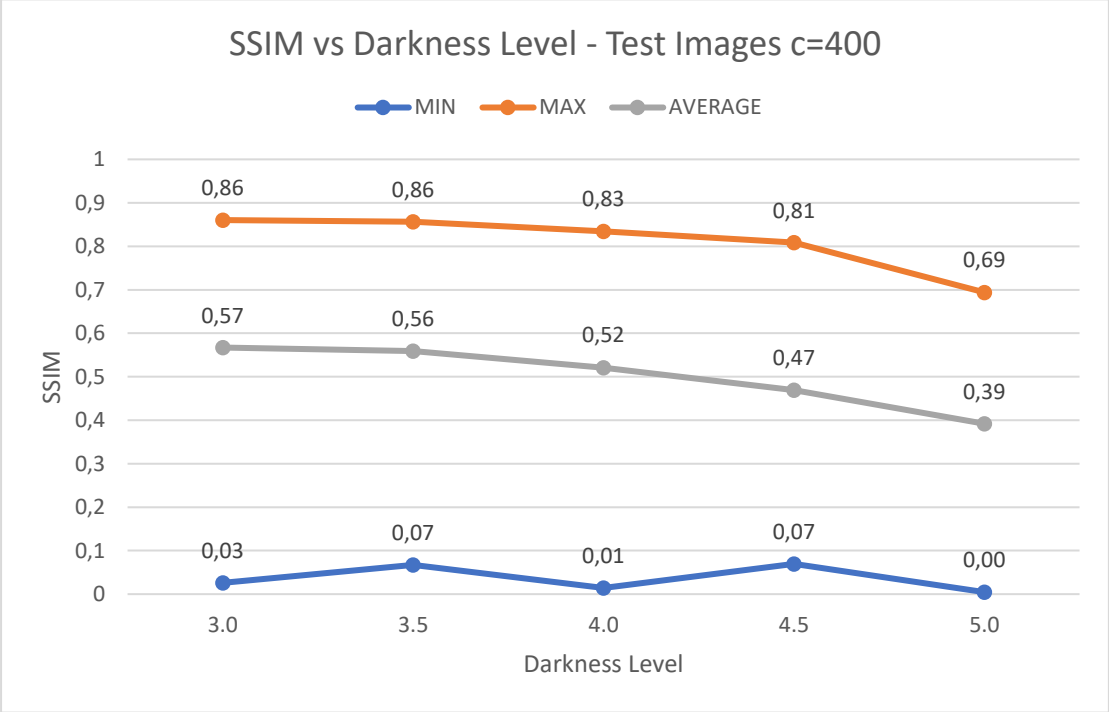
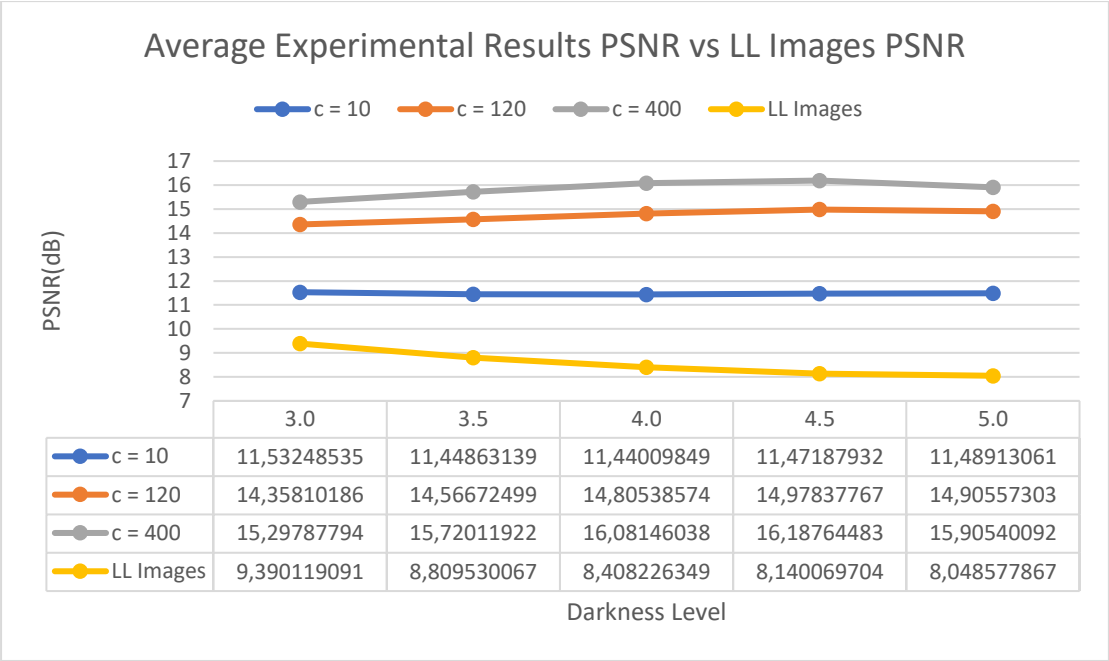


Figure 2.212: Experimental results SSIM vs Darkness Level for test images

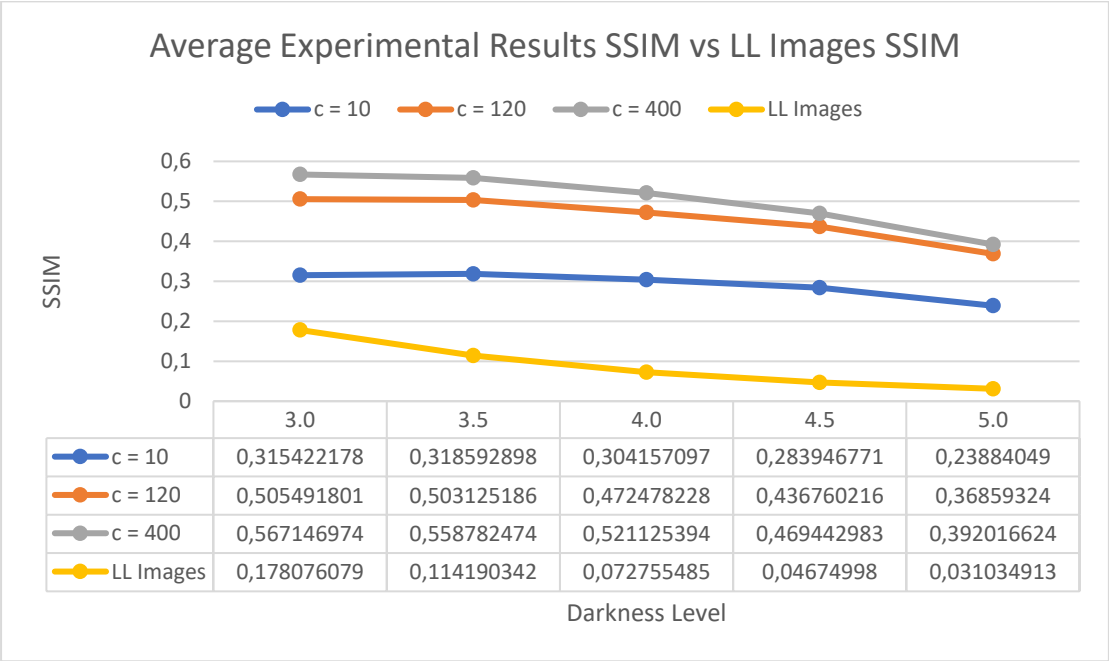
For c=400 again the best results are obtained, with the values of the metrics improving further. Here it is also observed that the increase in the darkness

level has little effect on the performance of the algorithm, as can be seen from the values of the metrics.



	3.0	3.5	4.0	4.5	5.0
c = 10	11,53248535	11,44863139	11,44009849	11,47187932	11,48913061
c = 120	14,35810186	14,56672499	14,80538574	14,97837767	14,90557303
c = 400	15,29787794	15,72011922	16,08146038	16,18764483	15,90540092
LL Images	9,390119091	8,809530067	8,408226349	8,140069704	8,048577867

Figure 2.213: Experimental results average PSNR vs LL Images PSNR test set



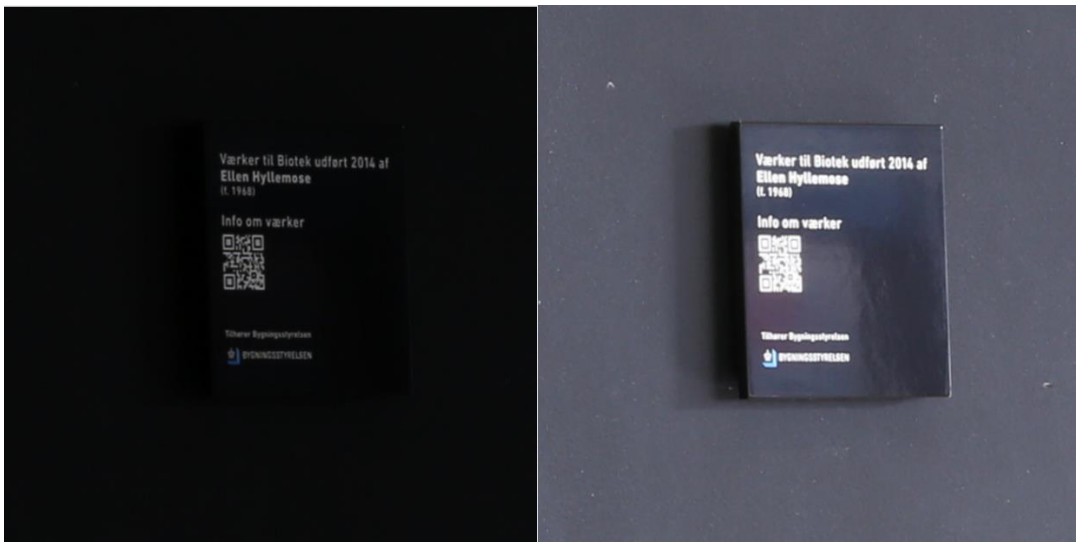
	3.0	3.5	4.0	4.5	5.0
c = 10	0,315422178	0,318592898	0,304157097	0,283946771	0,23884049
c = 120	0,505491801	0,503125186	0,472478228	0,436760216	0,36859324
c = 400	0,567146974	0,558782474	0,521125394	0,469442983	0,392016624
LL Images	0,178076079	0,114190342	0,072755485	0,04674998	0,031034913

Figure 2.214: Experimental results average SSIM vs LL Images SSIM test set

At this point it is worth commenting on the results as a whole. For a constant value of 10 we saw that the worst results occur for the quality metric values, as they improve little. Compared to PSNR we have that it improves on average by 3.01dB and SSIM by 0.2. From these we understand that a small value of c is not enough to enhance the LL images. By increasing the value of the constant to 120, the quality metric values improve noticeably, with PSNR increasing by an average of 5.77dB and SSIM by 0.35. We understand that larger values of the constant are needed to clearly enhance dark images. Further increasing the value of the constant to 400, the quality metric values improve even more, with PSNR increasing by 6.85dB and SSIM by 0.39. Based on this we understand that large values of the constant c are needed to improve and enhance the LL images, which is explained by the fact that the higher the value of the constant, the more information we get from the neighboring pixels. This above information helps to increase the quality of the final result. Finally, it was observed that increasing the darkness level has little effect on the performance of the algorithm, since the values of the quality metrics from level to level do not change strongly. This can be explained by the nature of the algorithm since, as we mentioned in the theoretical part, the final result recovered is independent of the lighting conditions, which is also confirmed by the experimental results of the quality metric values. Any reduction in these values is due to the fact that the images become too dark, with the result that the information of the reflectance coefficients cannot be fully recovered.

To confirm our results visually we will present, for each darkness level, the result for a random image per value of constant c. In addition, they will be accompanied by the original LL image and the corresponding ground truth so that we can compare them with each other. Finally, together we will present the respective histogram of each image.

Darkness Level: 3.0



Original Low Light **Normal Light**

c = 10 **c = 120** **c = 400**

For darkness level 3.0 we notice that the experimental result obtained for c=10 does not visually resemble the ground truth case at all, and is characterized by strong color distortions and noise. This fact justifies what we have reported so far and explains the minimal improvement of the quality metric values. For c=120 the result is noticeably improved, as we recover a large part of the visual information, while for c=400 the best experimental result is obtained, with the image being visually very close to the ground truth case. From the experimental histograms we observe that in all 3 cases histograms similar to those of the ground truth case are obtained, but for c=400 the best result is again obtained.

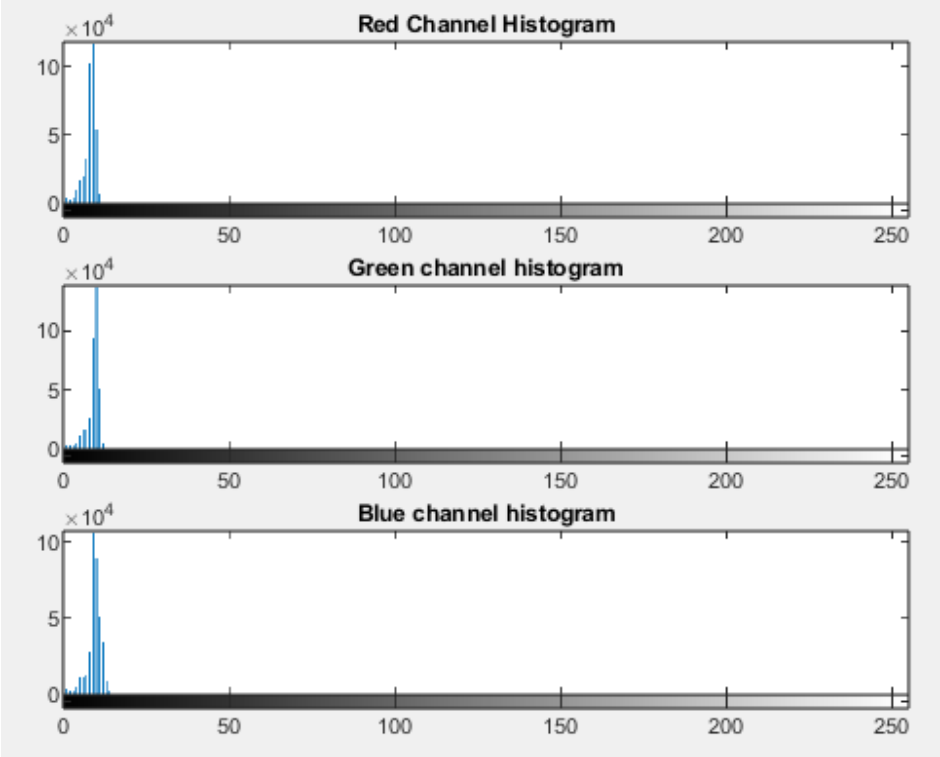


Figure 2.215: Histogram of 3.0 LL image

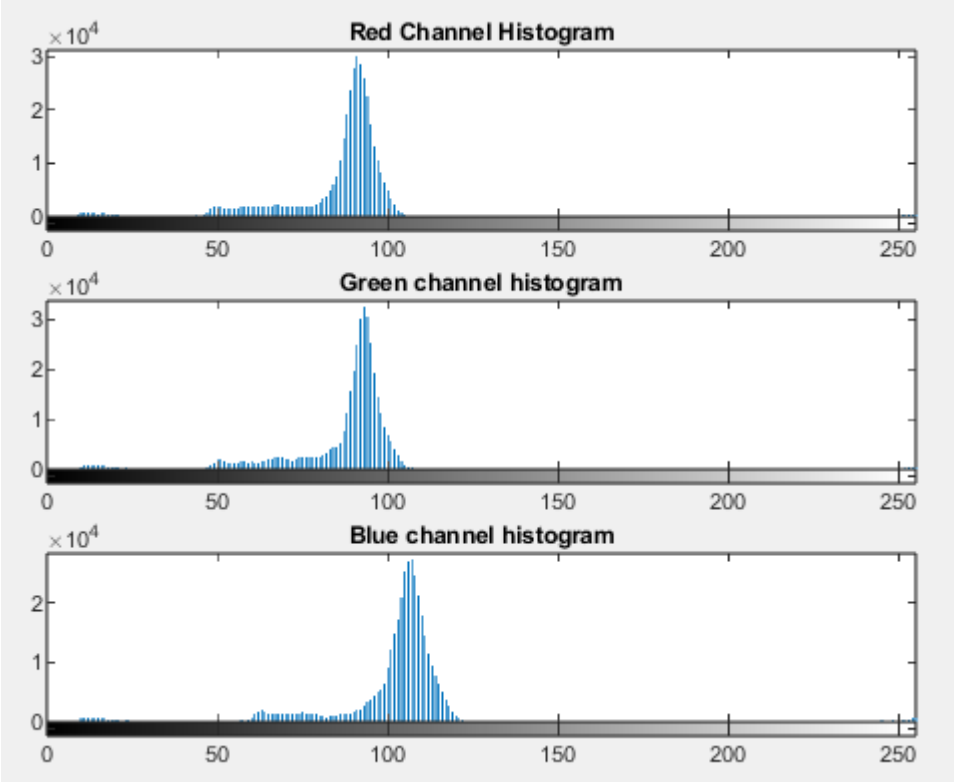


Figure 2.216: Histogram of 3.0 NL image

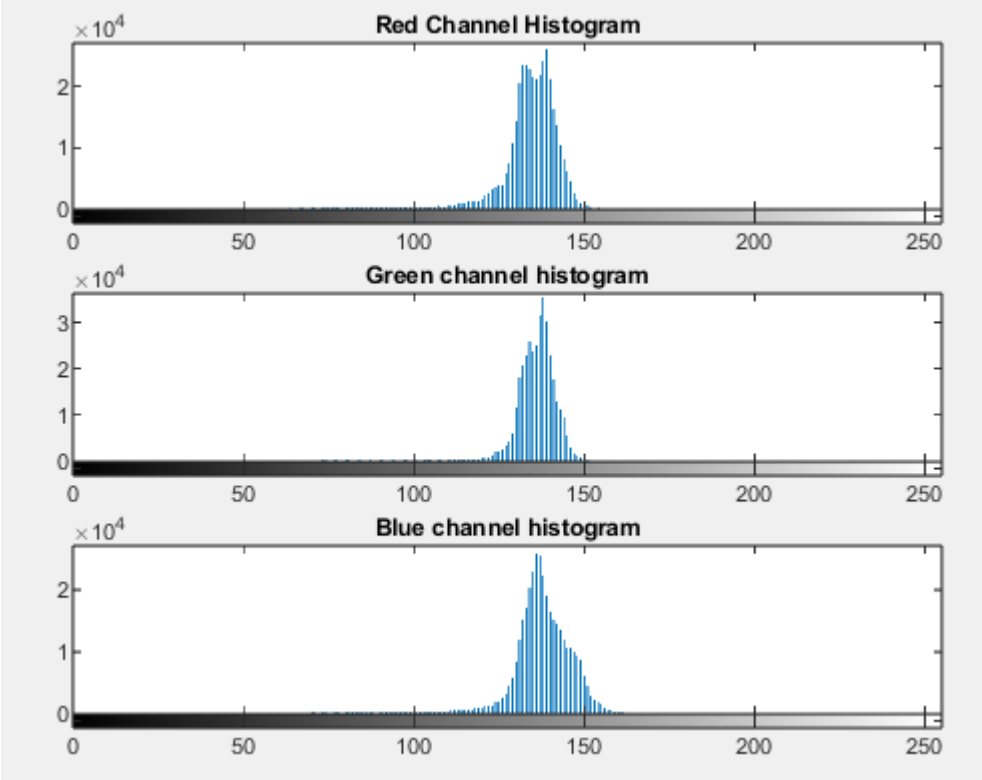


Figure 2.217: Histogram of 3.0 Experimental result image with c=10

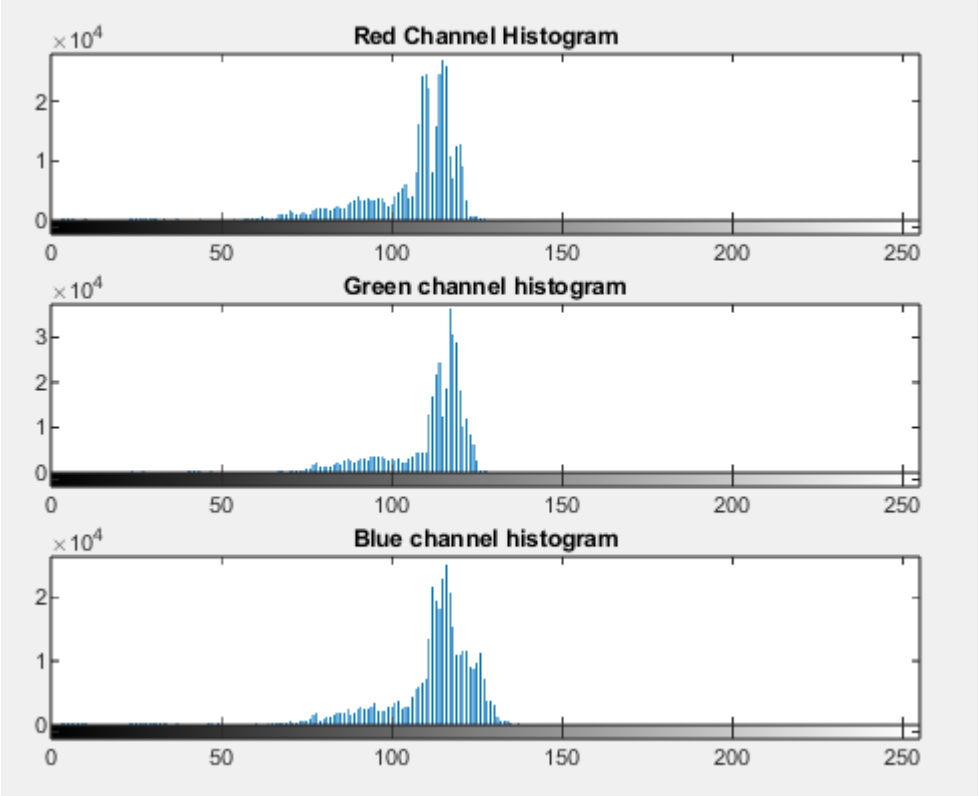


Figure 2.218: Histogram of 3.0 Experimental result image with c=120

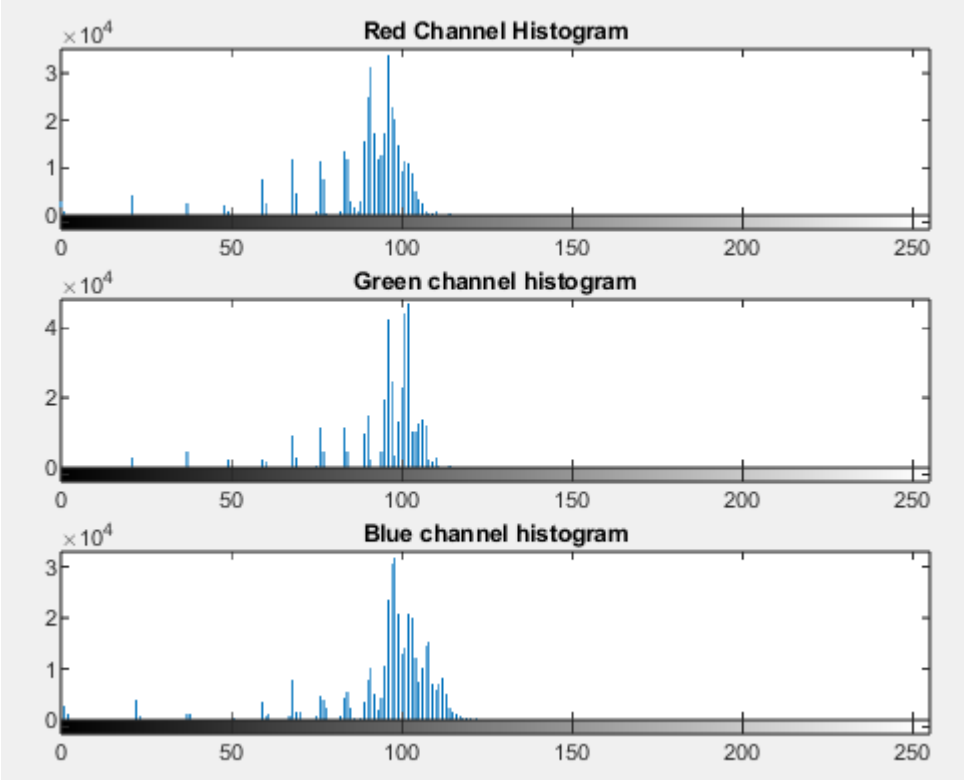


Figure 2.219: Histogram of 3.0 Experimental result image with c=400

Darkness Level: 3.5

Original Low Light **Normal Light**

c = 10 **c = 120** **c = 400**

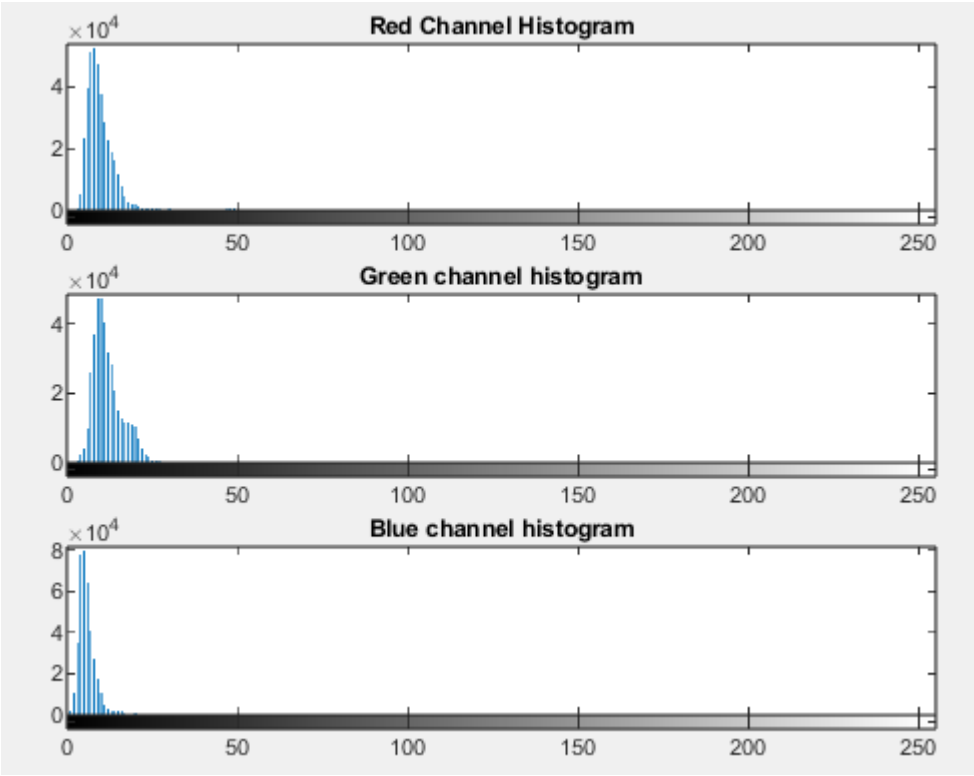


Figure 2.220: Histogram of 3.5 LL image

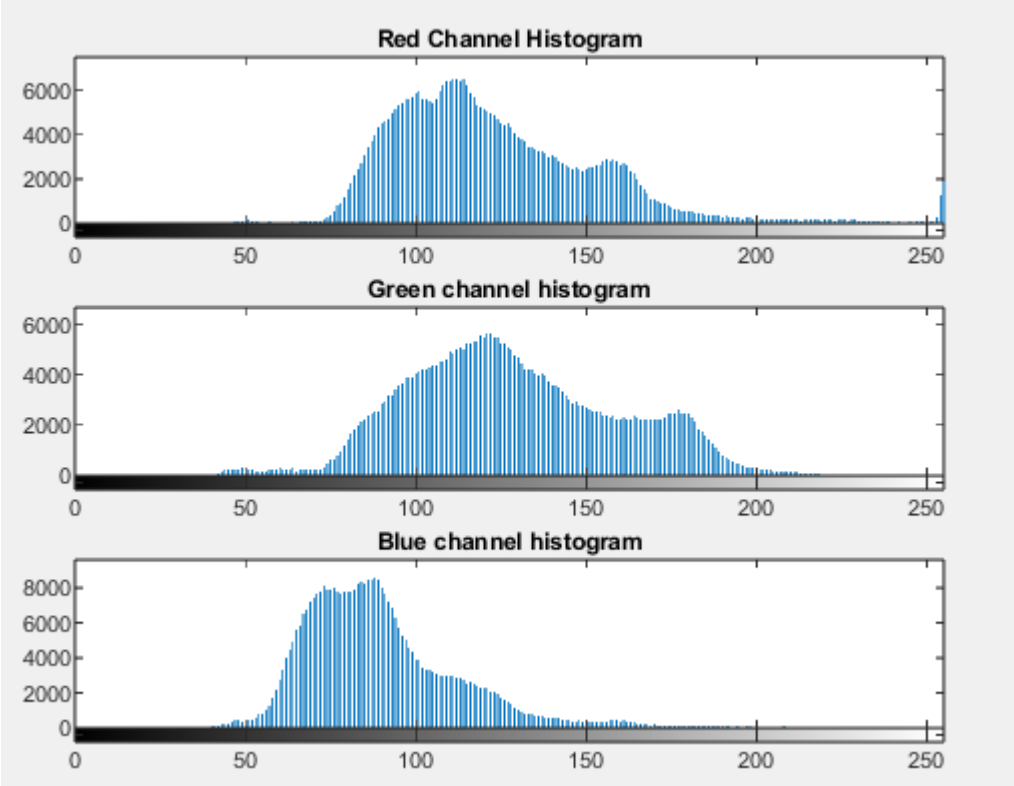


Figure 2.221: Histogram of 3.5 NL image

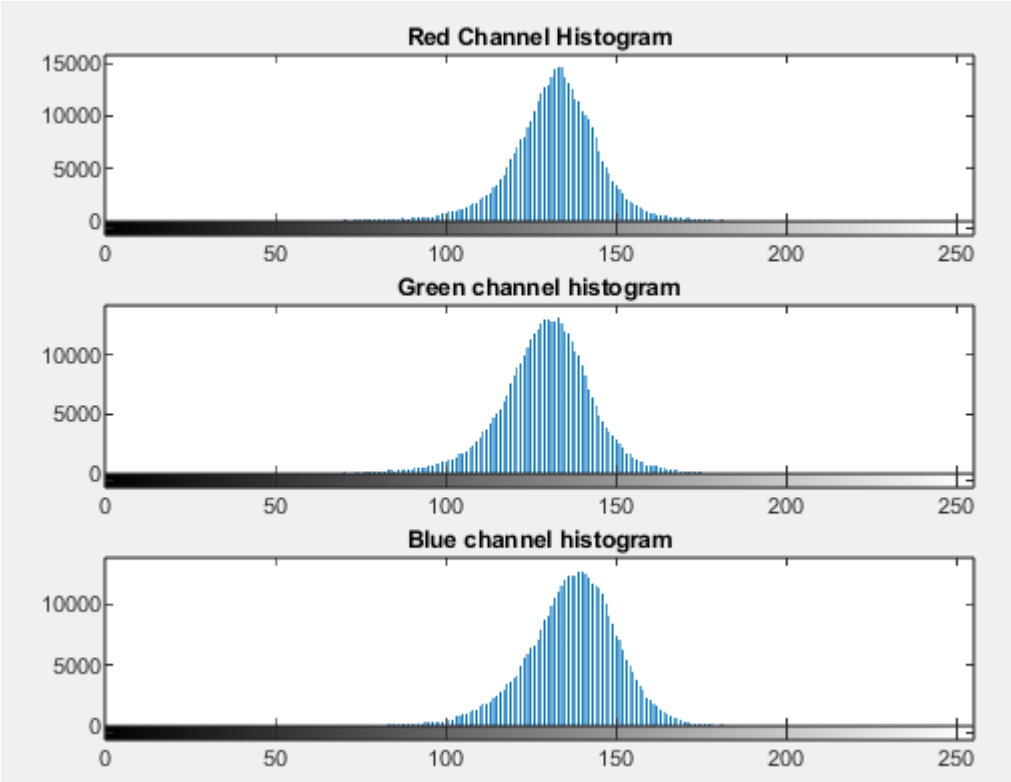


Figure 2.222: Histogram of 3.5 Experimental result image with c=10

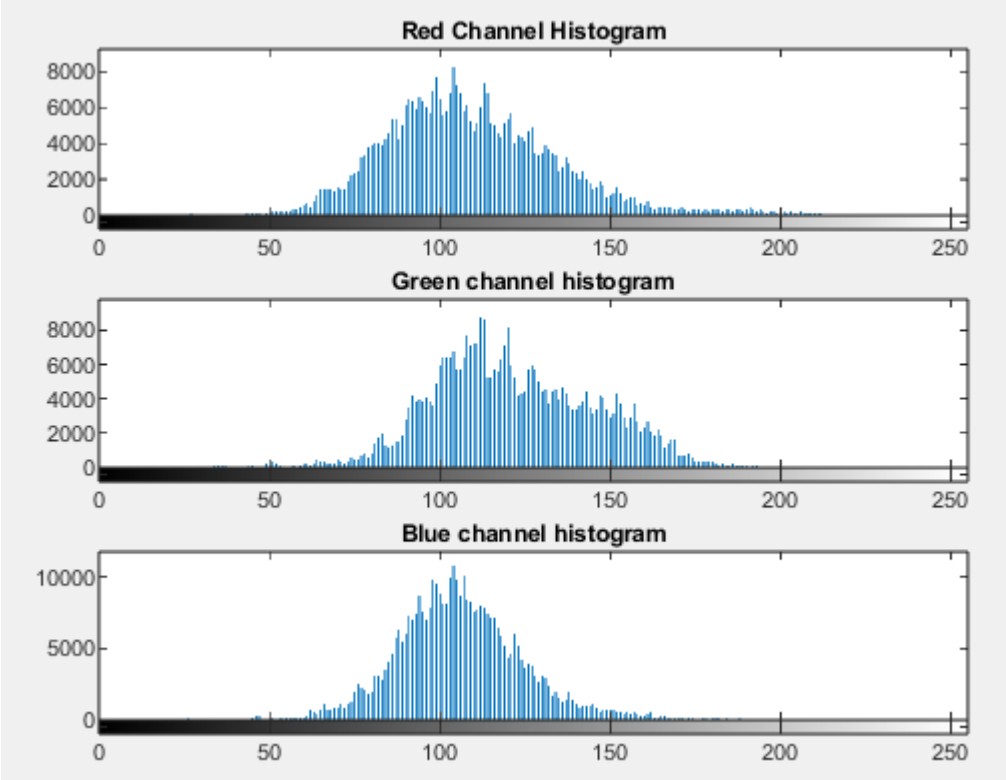


Figure 2.223: Histogram of 3.5 Experimental result image with c=120

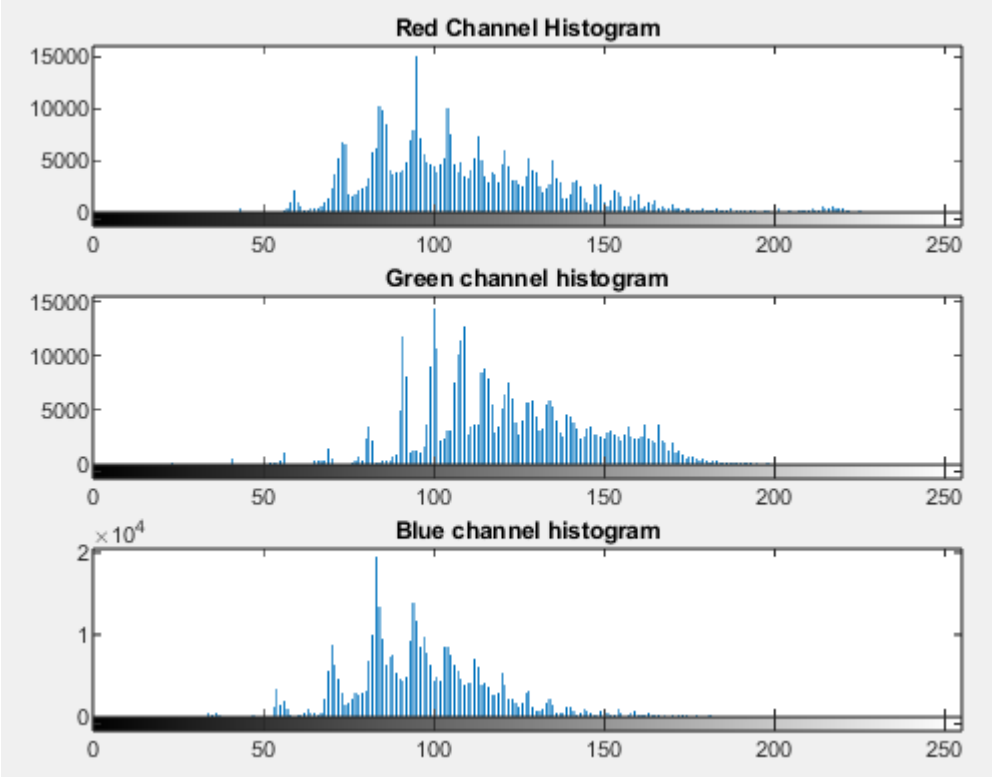


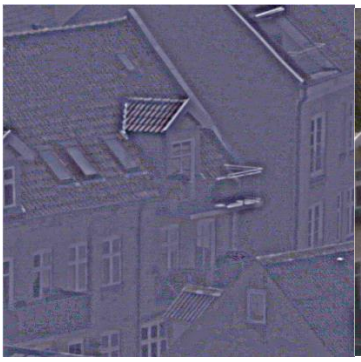
Figure 2.224: Histogram of 3.5 Experimental result image with c=400

For darkness level 3.5 we also observe that the experimental result for a constant of 10 is characterized by intense noise and color distortions, explaining the minimal improvement of the quality metric values. For c=120 the result is noticeably improved, as we also recover the color information while the noise is reduced. For c=400 the best result is obtained, which we can also confirm from the experimental histogram of figure 2.224, from where we see that its form is very close to that of the ground truth case.

Darkness Level: 4.0

Original Low Light

Normal Light



c = 10

c = 120

c = 400

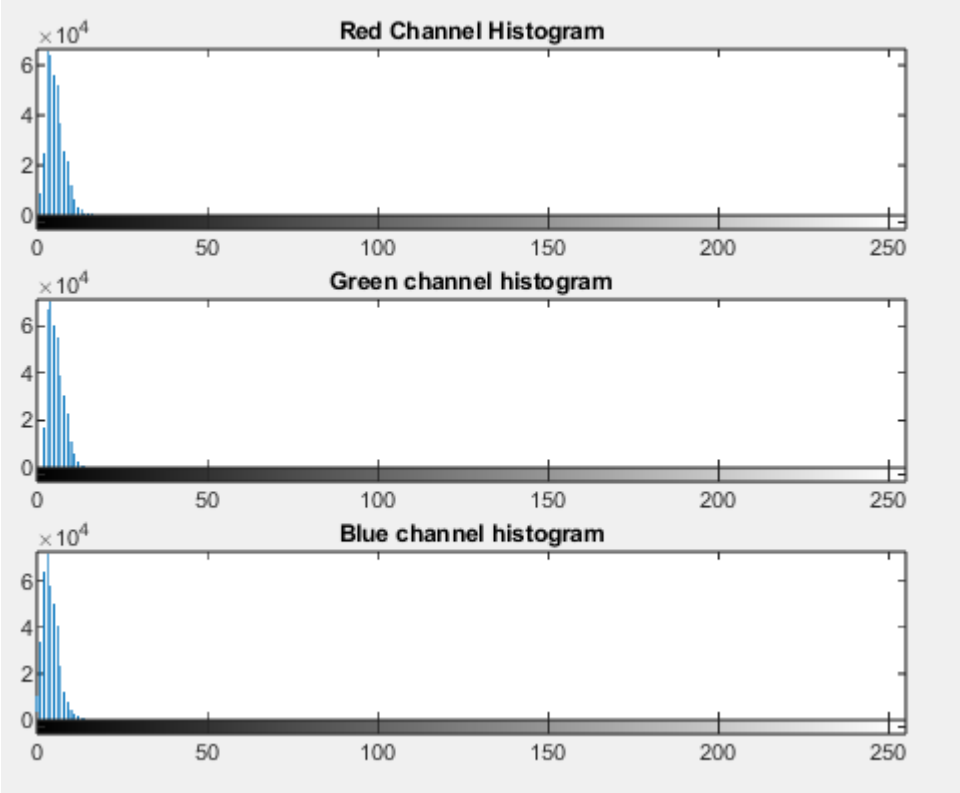


Figure 2.225: Histogram of 4.0 LL image

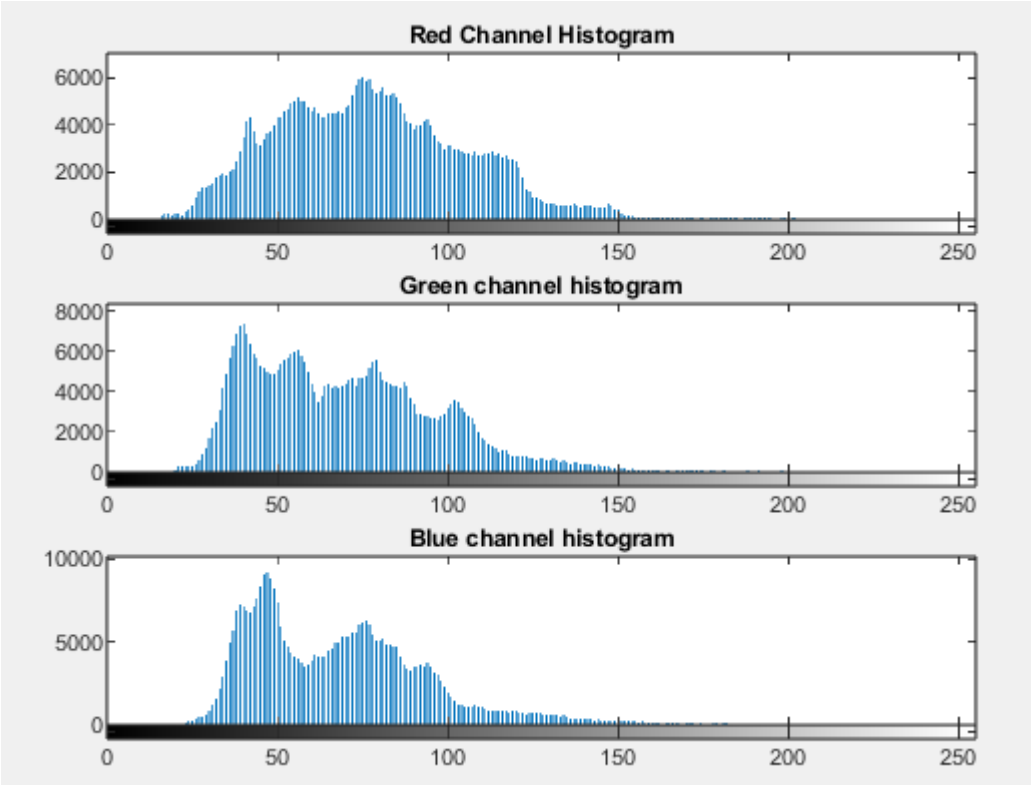


Figure 2.226: Histogram of 4.0 NL image

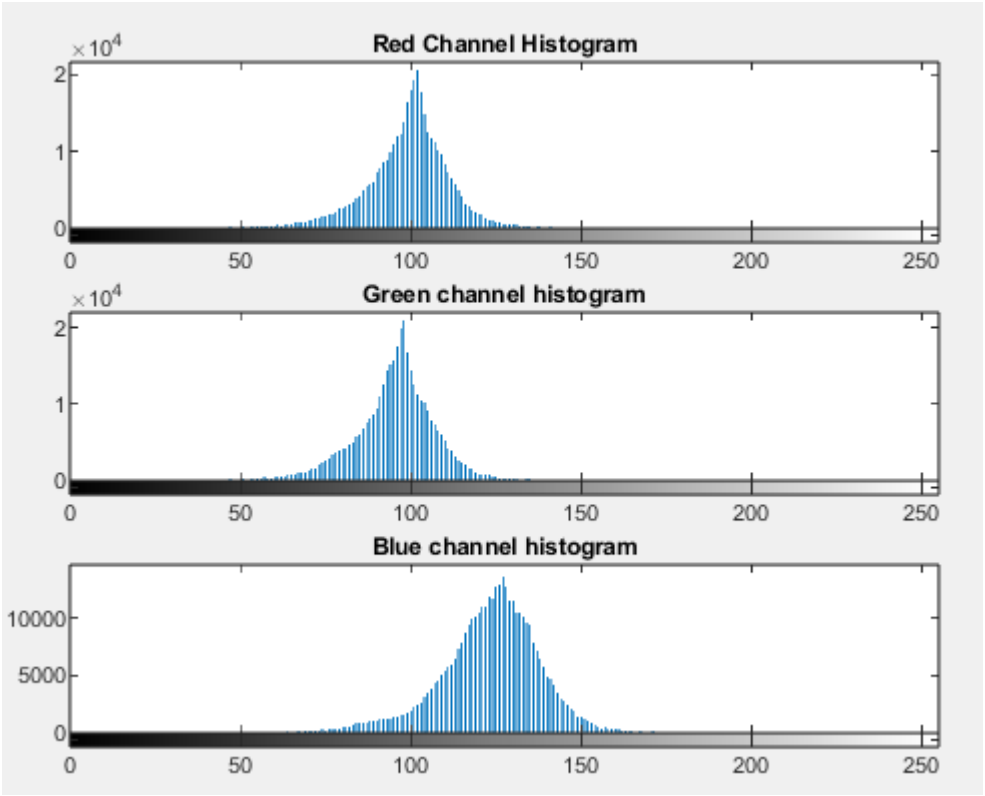


Figure 2.227: Histogram of 4.0 Experimental result image with c=10

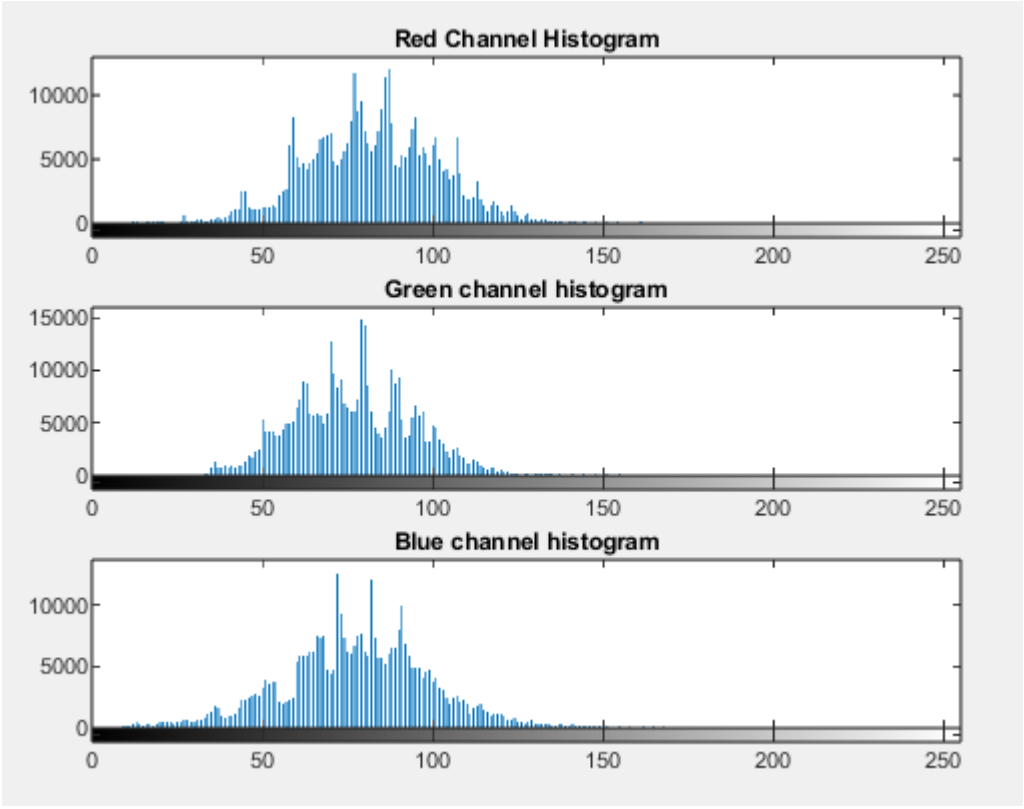


Figure 2.228: Histogram of 4.0 Experimental result image with c=120

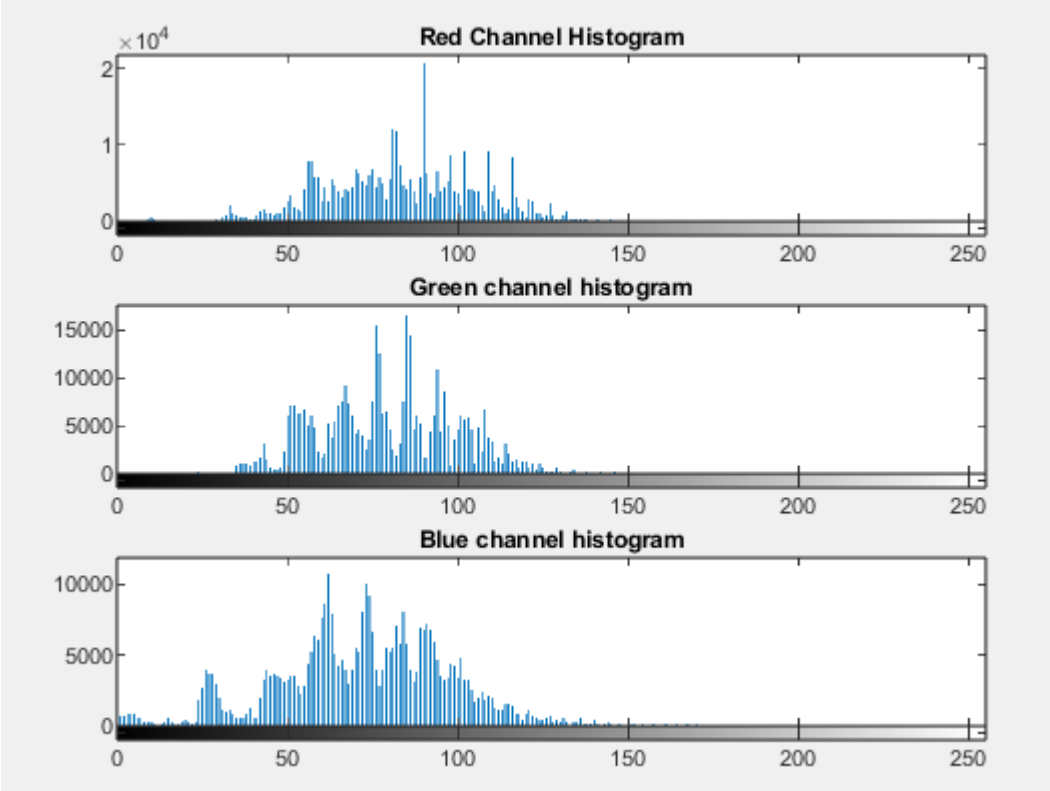


Figure 2.229: Histogram of 4.0 Experimental result image with c=400

In the case of darkness level 4.0 again for c=10 the worst result is obtained, with the image characterized by intense noise and color distortions. The algorithm succeeds in recovering the edge and general texture information of the image but fails to recover the color information. This is achieved by increasing the value of the constant to c=120, which significantly improves the result, with the optimal gain being at c=400. Then the best result is obtained, which can also be confirmed by the experimental histogram in figure 2.229, where we see that a form has been recovered that is very close to the ground truth case.

Darkness Level: 4.5

Original Low Light **Normal Light**

c = 10 **c = 120** **c = 400**

For darkness level 4.5 we observe that for c=10 the worst result is obtained, compared to the other two constant cases. The general texture of the image is recovered, but not the color information, resulting in a noisy image. Increasing the value of the constant allows us to recover the color information, partially for c=120 and fully for c=400, which indicates that large values of the constant work better for this problem. This is also confirmed by the histograms, where in figure 2.233 the histogram for c=120 is very close to the form of the ground truth case, while in figure 2.234 we see that for c=400 the histogram improves even more.

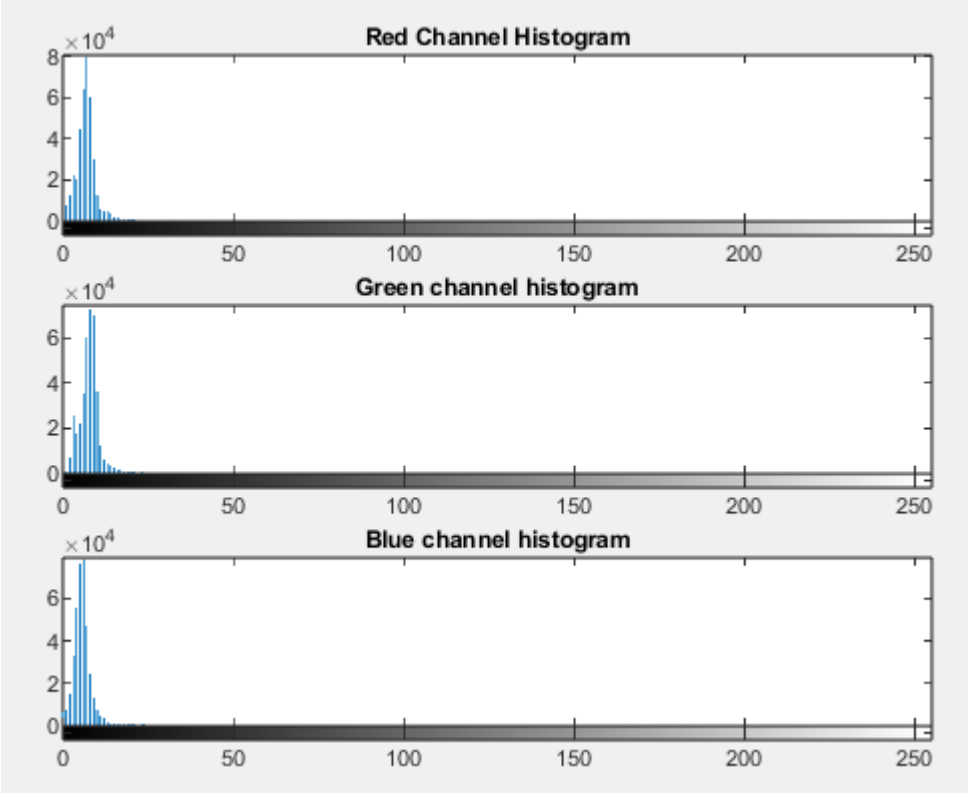


Figure 2.230: Histogram of 4.5 LL image

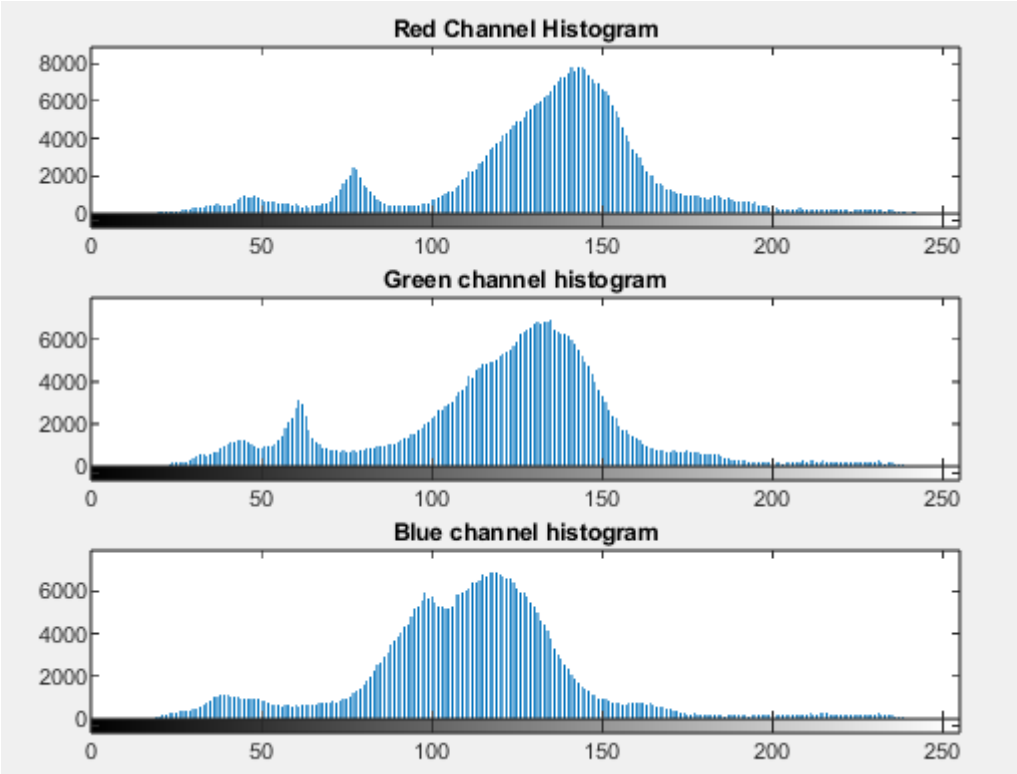


Figure 2.231: Histogram of 4.5 NL image

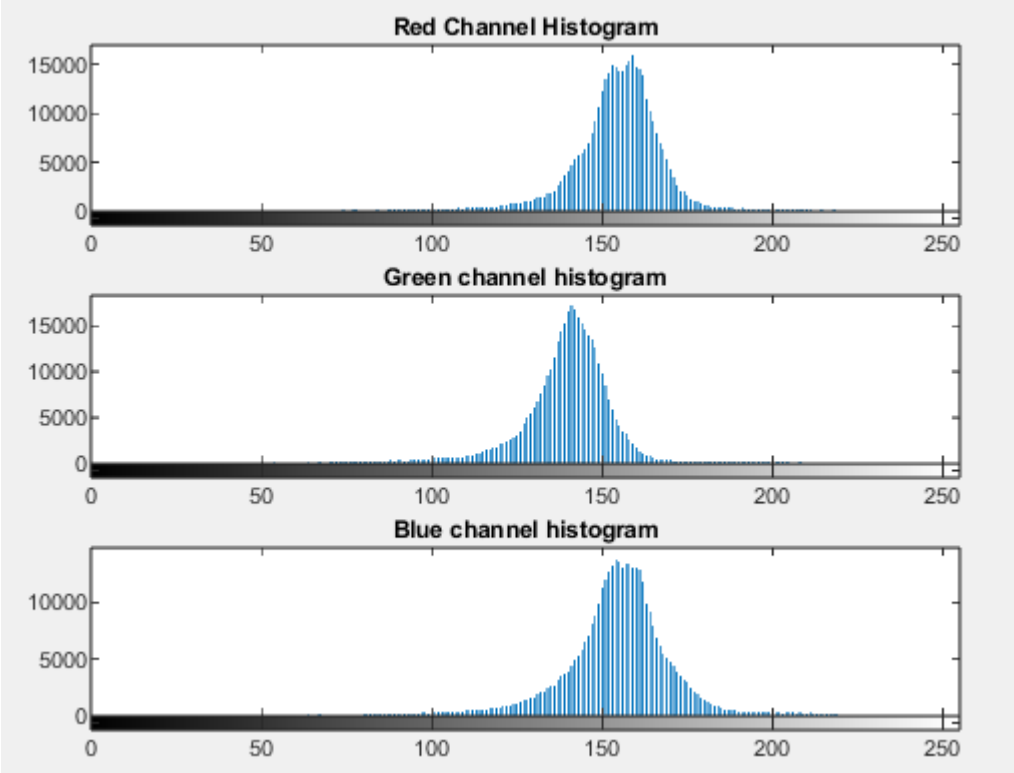


Figure 2.232: Histogram of 4.5 Experimental result image with c=10

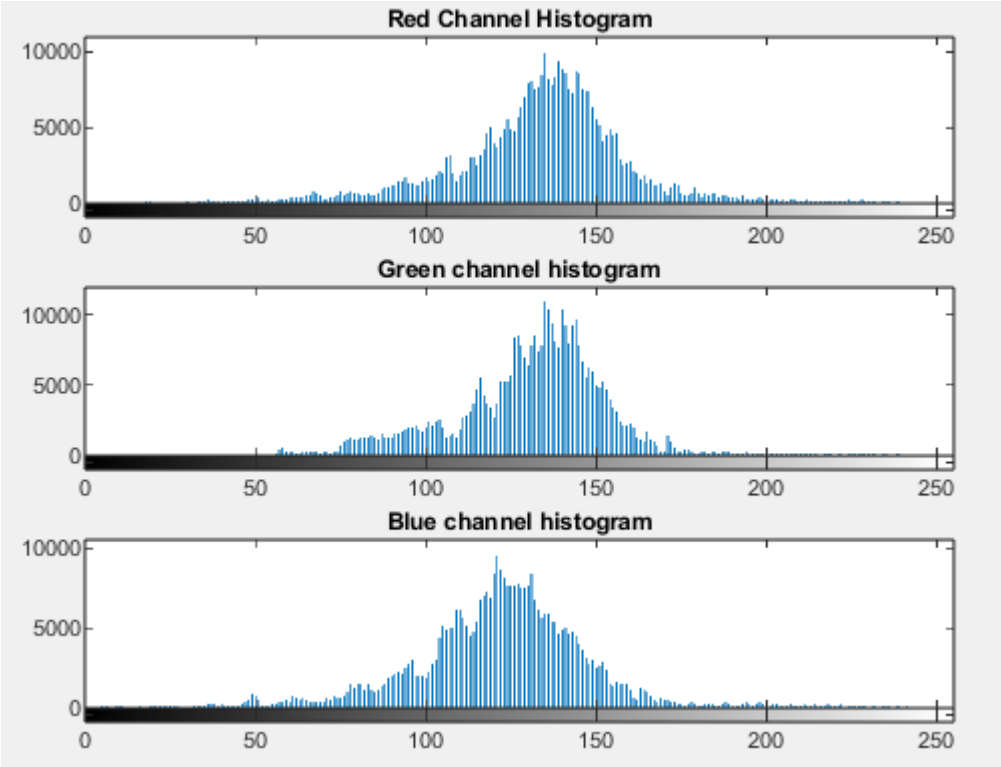


Figure 2.233: Histogram of 4.5 Experimental result image with c=120

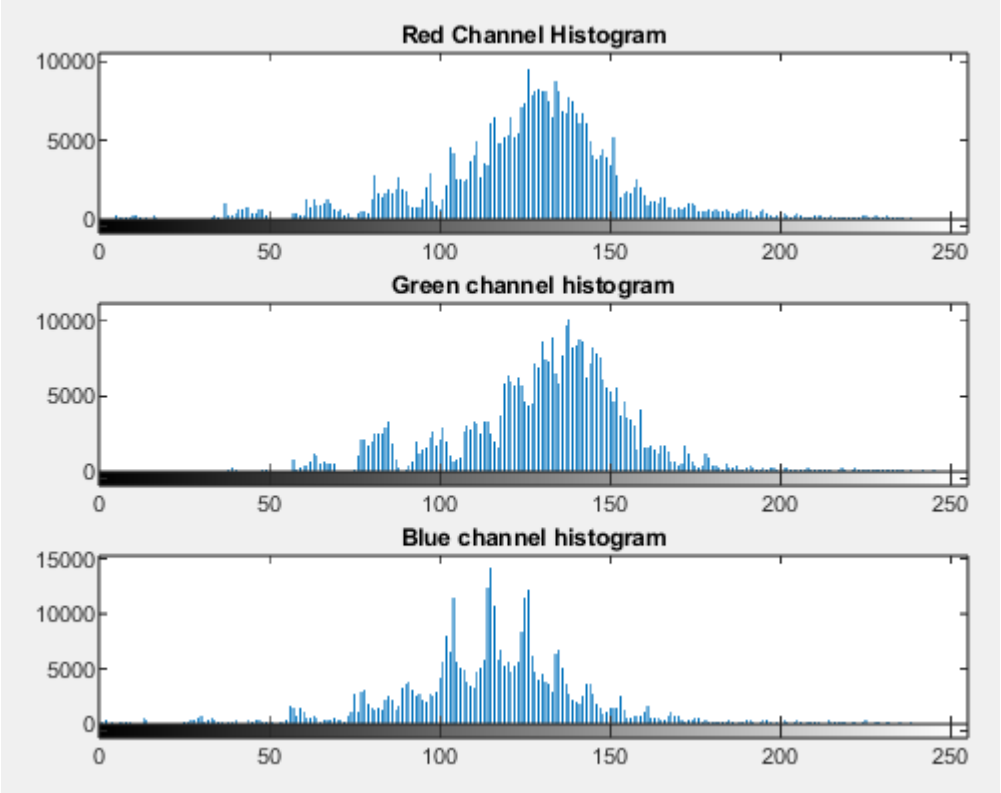
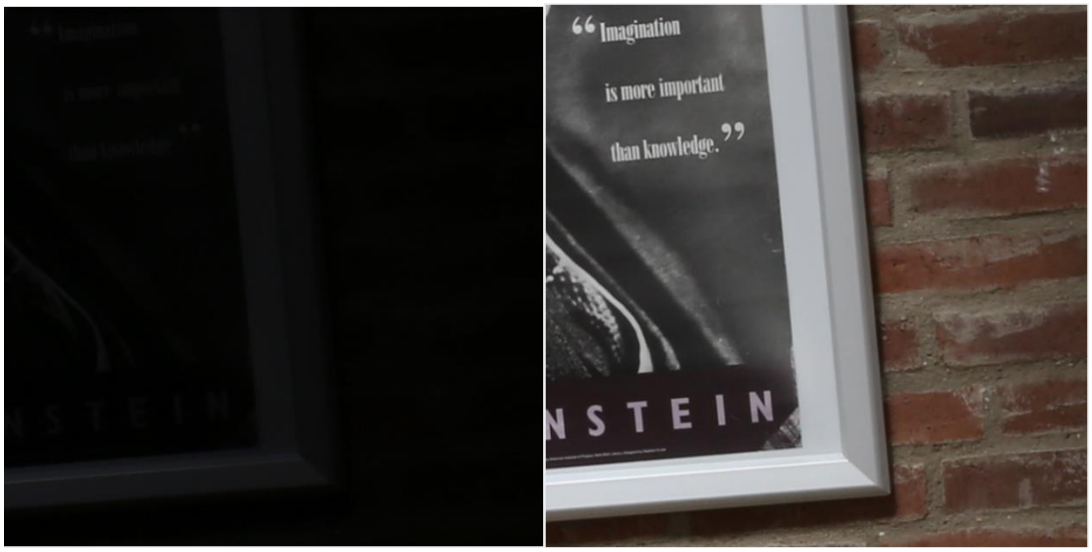


Figure 2.234: Histogram of 4.5 Experimental result image with c=400

Darkness Level: 5.0



Original Low Light **Normal Light**

c = 10 **c = 120** **c = 400**

In the case of darkness level 5.0 we again observe that for c=10 the worst result is obtained, compared to the other two constant values. Here again we see that the general texture of the image is recovered, but not the color information. By increasing the value of the constant, we manage to recover the color information, with c=400 giving the best result. Nevertheless, we see that there are starting to be some color distortions. This is because the images have now become very dark, making it difficult to even retrieve the reflectance index.

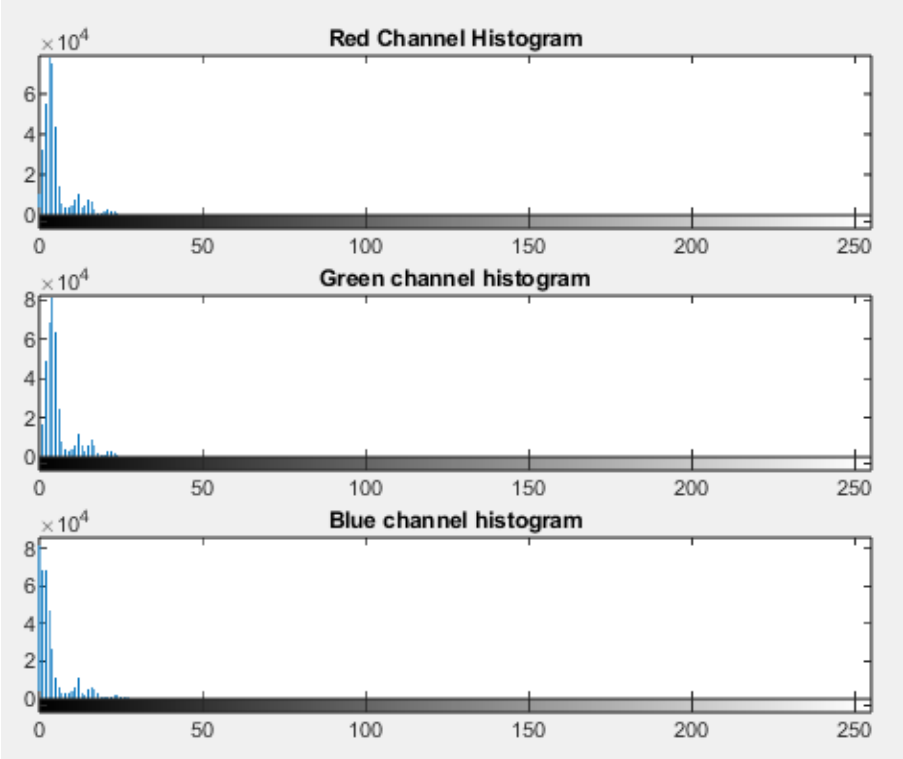


Figure 2.235: Histogram of 5.0 LL image

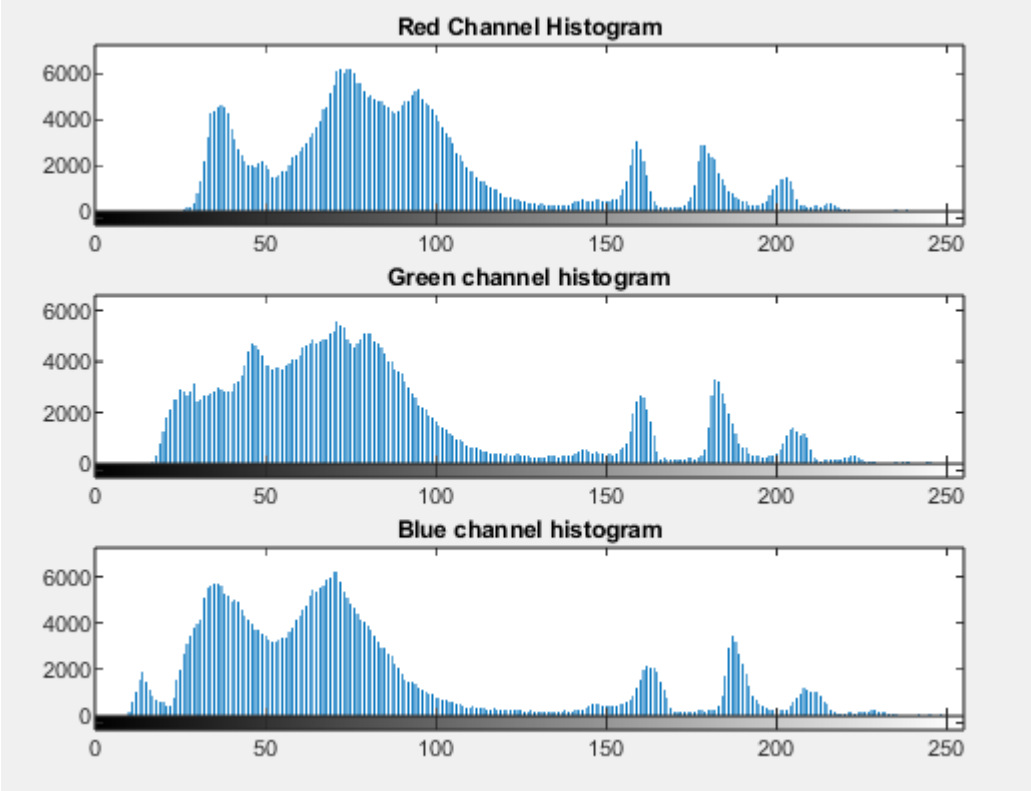


Figure 2.236: Histogram of 5.0 NL image

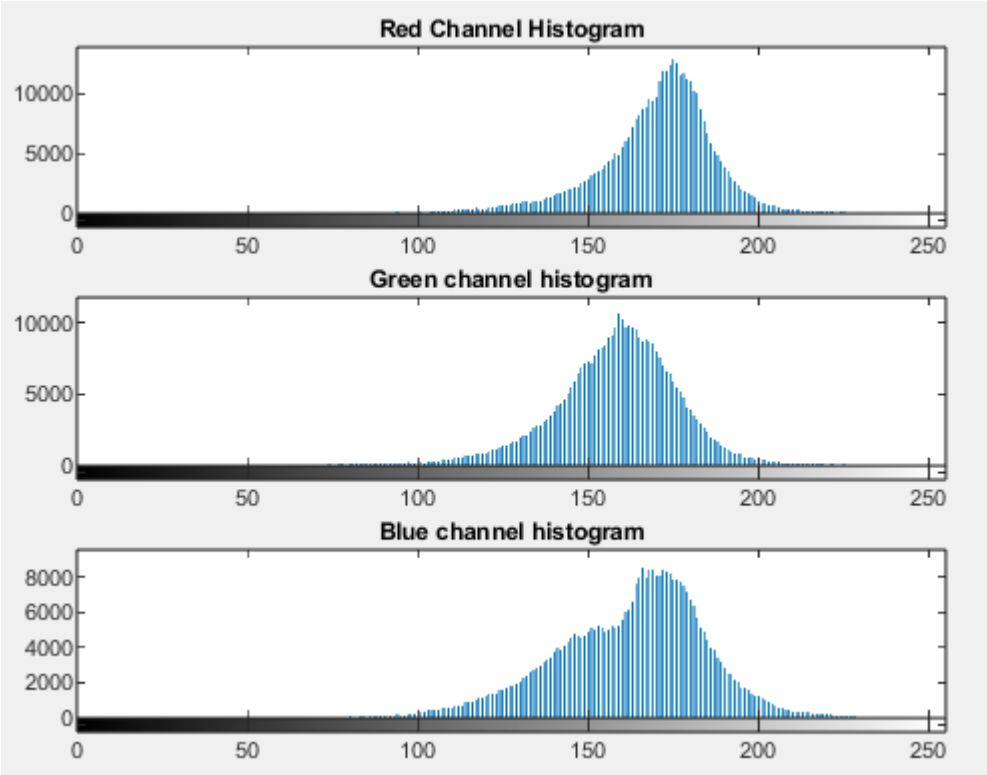


Figure 2.237: Histogram of 5.0 Experimental result image with c=10

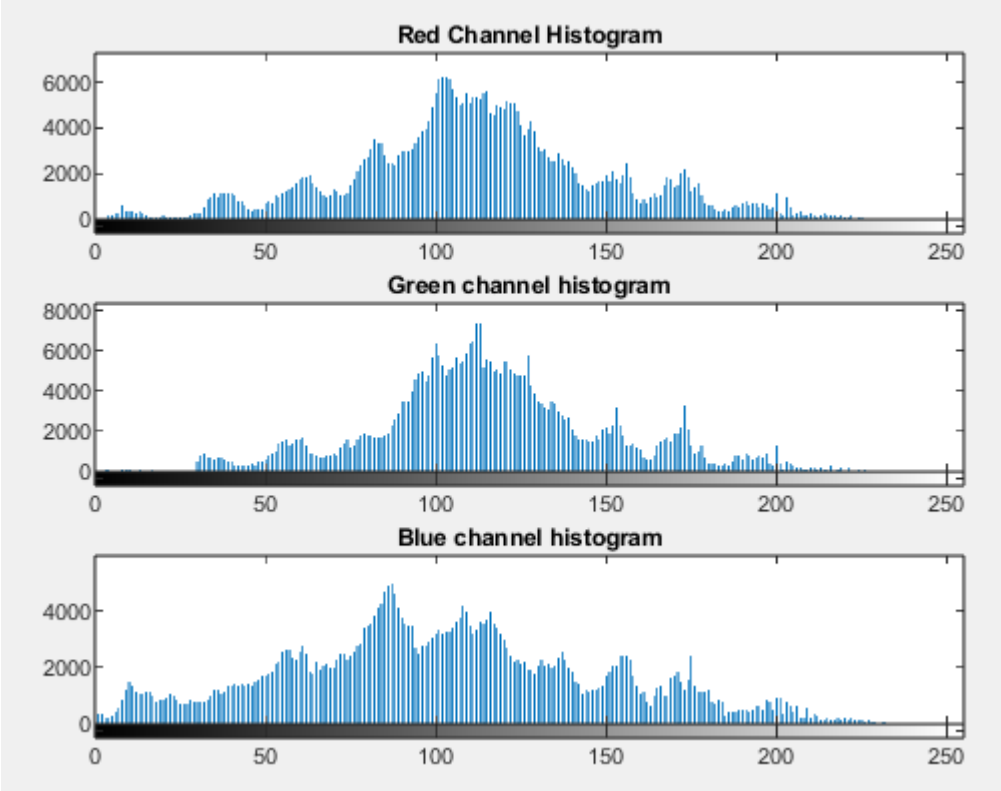


Figure 2.238: Histogram of 5.0 Experimental result image with c=120

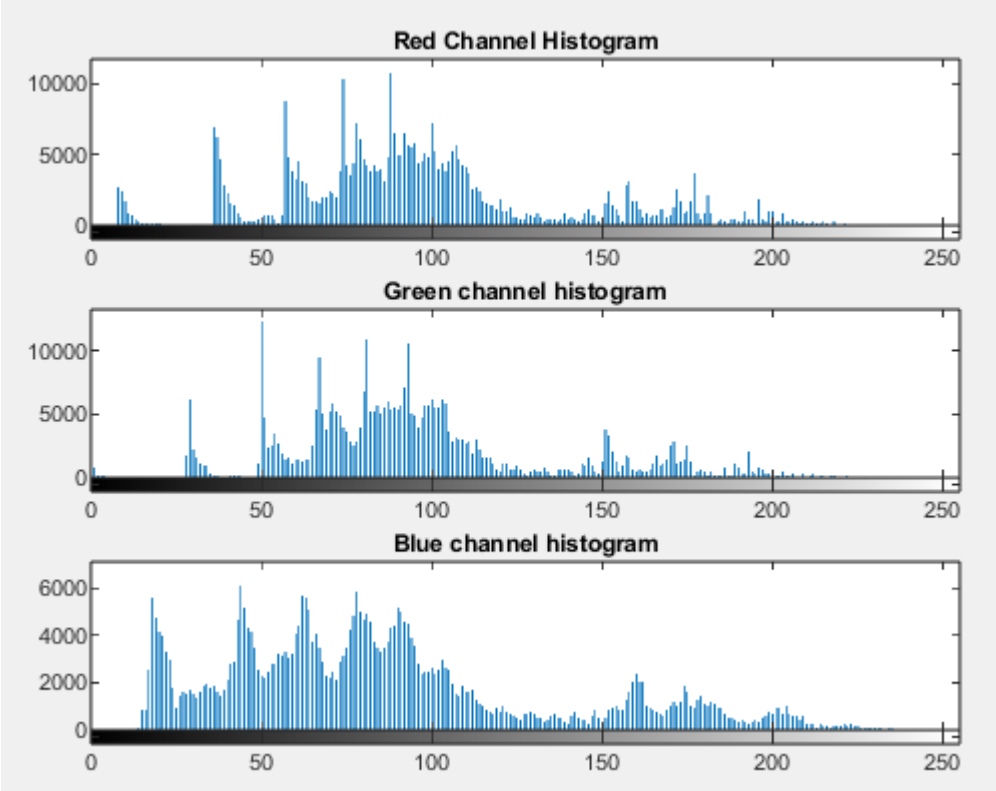


Figure 2.239: Histogram of 5.0 Experimental result image with c=400

At this point it is worth commenting in general on the results obtained from all the above analysis. We observed that the method for a constant value of c=10, slightly improves the values of the quality metrics, with the images remaining of low quality. This is also confirmed by the experimental results we showed above, where we saw that the images are characterized by strong noise and color distortions. However, the method managed to recover the general texture of the image (edges etc), which means that small values of the constant c can recover such local information but cannot restore the color information. The recovery of the visual information is achieved by using larger values of the constant c. The best result is obtained for c=400, where the greatest improvement of the quality metric values is observed, and in addition the recovery of most of the visual information. This is also confirmed by the histograms we presented above, where we see that for each darkness level case the experimental histogram, for c=400, has a form very close to the ground truth case. Finally, we observed that the performance of the method is not affected by the increase of the darkness level and the values of the quality metrics change little with its increase, and the visual results remain of high quality. This resistance to the change of the darkness level is due to the nature of the algorithm, as its purpose is to recover the reflectance coefficient of the scene, which is independent of the distribution and amount of brightness.

In summary, the method gives impressive results, but again we need to choose an appropriate value of the parameter. In the next section we take the method a step further by choosing multiple parameter values and combining the results of the individual SSRs to capture the advantages of each value range.

2.6 Multi Scale Retinex

This is an extension of the algorithm analyzed in the previous section. One of the drawbacks noted by the authors was that the SSR algorithm cannot strike a balance between detailed enhancement information and color fidelity, which we also observed in our experiments, a problem due to the method's use of only one parameter. The solution they came up with was to use more than one parameter and combine the results [27]. The new algorithm is called Multi Scale Retinex, from the fact that it uses more than one parameter, and the final result comes from the relation:

$$R_{MSR_i} = \sum_{n=1}^{N} \omega_n R_{n_i}$$

where I = {R, G, B} is the respective color band, R_{MSR} is the final result from the application of the MSR algorithm, R_n is the result of the SSR method for parameter c_n, N is the total number of parameters we combine, and ω_n is the weight of each parameter. The only difference with before is that the surround function is now defined as follows:

$$G_n(x, y) = Ke^{-(\frac{x^2+y^2}{c_n^2})}$$

where K is the known normalization parameter.

The question that arises is how many parameters we need to combine and what value the weight of each parameter should take. After experimentation the authors [27] concluded that N = 3 parameters are sufficient, with the weights taking values $\omega_n = \frac{1}{N} = \frac{1}{3}$. Basically we calculate the SSR for 3 different parameters, one small, one large and one of medium range, and the average is the MSR result. This is more complex, from the point of view of implementation, algorithm, which is implemented with the function MultiScaleRetinex, shown in Figure B.2.21 of Appendix B. We will apply this function to all images in the data set, which is done with the scripts presented in Figures B.2.22 and B.2.23 of appendix B. After applying them we have at our disposal the experimental

results, based on which we should evaluate the performance of the algorithm. To do this we will calculate the performance metrics per image, which we do with the script shown in Figure B.2.24 of Appendix B. With the results we will construct summary tables with the min/max/average value of each quality metric, and line charts of PSNR and SSIM per darkness level, in order to evaluate to what extent the performance of the algorithm is affected by the darkness level. In addition, we will construct line charts with the average PSNR and SSIM for both the experimental results and the original LL images, to see how much the quality of the images improves.

Training Set

MSE					
Level	3.0	3.5	4.0	4.5	5.0
MIN	239,0187383	376,0939	364,2164	422,496	477,2309
MAX	13097,86103	11641,21	10765,59	9428,147	8680,177
AVERAGE	3379,384613	3274,808	3102,219	2897,322	2729,704
	PSNR(dB)				
Level	3.0	3.5	4.0	4.5	5.0
MIN	6,958799827	7,470822	7,810424	8,38654	8,745518
MAX	24,34648411	22,37784	22,51721	21,87258	21,34352
AVERAGE	13,65861109	13,79953	13,98663	14,17075	14,34421
	SSIM				
Level	3.0	3.5	4.0	4.5	5.0
MIN	-0,06493951	0,009564	0,01649	0,023541	-0,04538
MAX	0,832428477	0,806773	0,815764	0,800235	0,744977
AVERAGE	0,495980155	0,480622	0,452868	0,411343	0,35308

	MV				
Level	3.0	3.5	4.0	4.5	5.0
MIN	77,61995349	69,77116	63,98045	57,80047	57,72048896
MAX	213,5505903	204,8339	199,0324	186,1193	174,0790682
AVERAGE	136,1930133	133,7225	130,2485	125,6644	120,2392155
	STD				
Level	3.0	3.5	4.0	4.5	5.0
MIN	60,08477997	61,94379	63,75354	64,29322	57,9565
MAX	824,9398009	802,2118	766,6946	708,9291	654,083
AVERAGE	308,7757309	305,1568	304,9774	307,9945	314,0277
	BRISQUE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	3,670254295	4,144557	5,335673	4,507405	10,16488
MAX	49,91066681	52,96108	54,67282	57,06991	57,57594
AVERAGE	30,62401427	30,70467	31,95871	34,1668	36,15179

	NIQE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	2,272376124	2,323353	2,693318	2,990678	3,165905
MAX	26,04609304	25,93341	7,916273	8,079607	8,777403
AVERAGE	3,780759247	4,011987	4,291982	4,685047	5,166678

Table 2.36: MSR results on training set

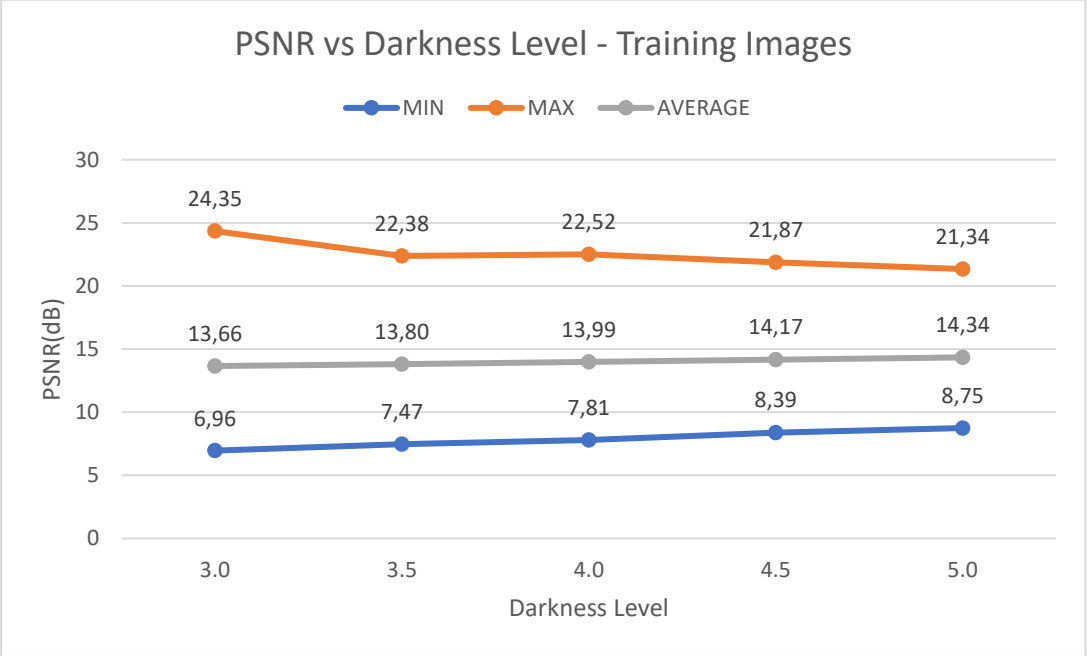


Figure 2.240: Experimental results PSNR vs Darkness level for training set

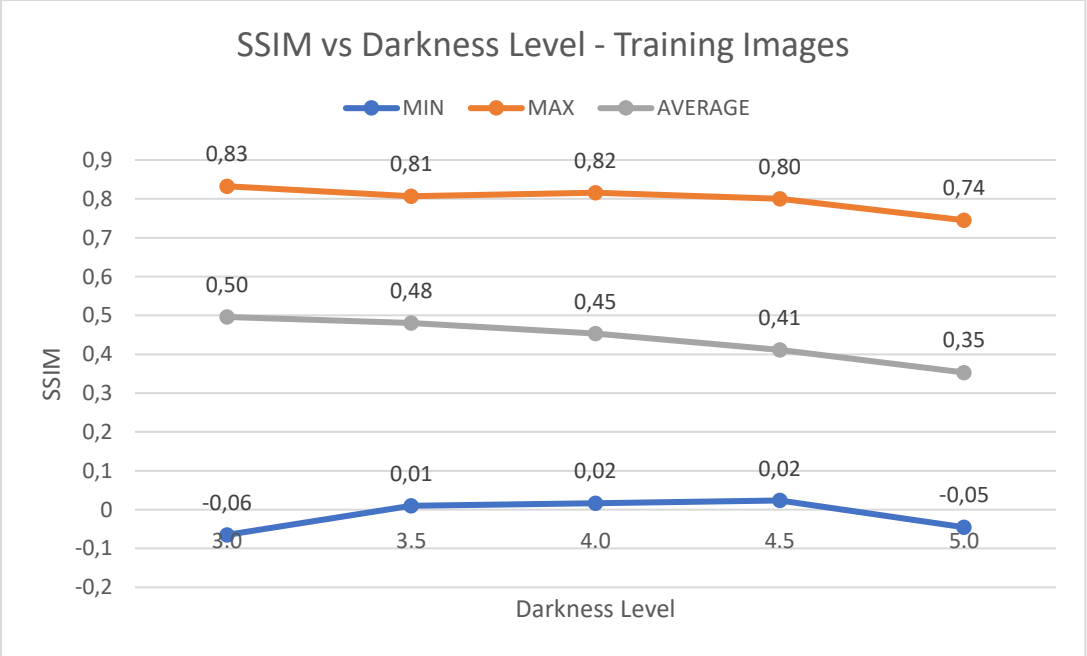


Figure 2.241: Experimental results SSIM vs Darkness level for training set

In the training set we notice that the method noticeably improves the values of the metrics. For the quality metrics with reference, we see that with the increase of the darkness level the MSE and PSNR improve while the SSIM decreases. From this we understand that in the experimental result the image brightness increases but the image texture information is not recovered. Based on this, we expect the experimental results to be characterized by over-exposure, i.e. the dark areas are over-enhanced and there are chromatic aberrations. For the quality metrics with no reference, we see that MV decreases with increasing darkness level, albeit slightly, while STD increases, meaning that the experimental results become slightly darker, but the pixel values are spread over a wider range around the mean value, i.e. we have a greater contrast. Finally, BRISQUE and NIQE increase with increasing darkness level, which means that we are moving away from natural statistics. This is to be expected as increasing the darkness level makes the images darker making it more difficult to recover the full visual information.

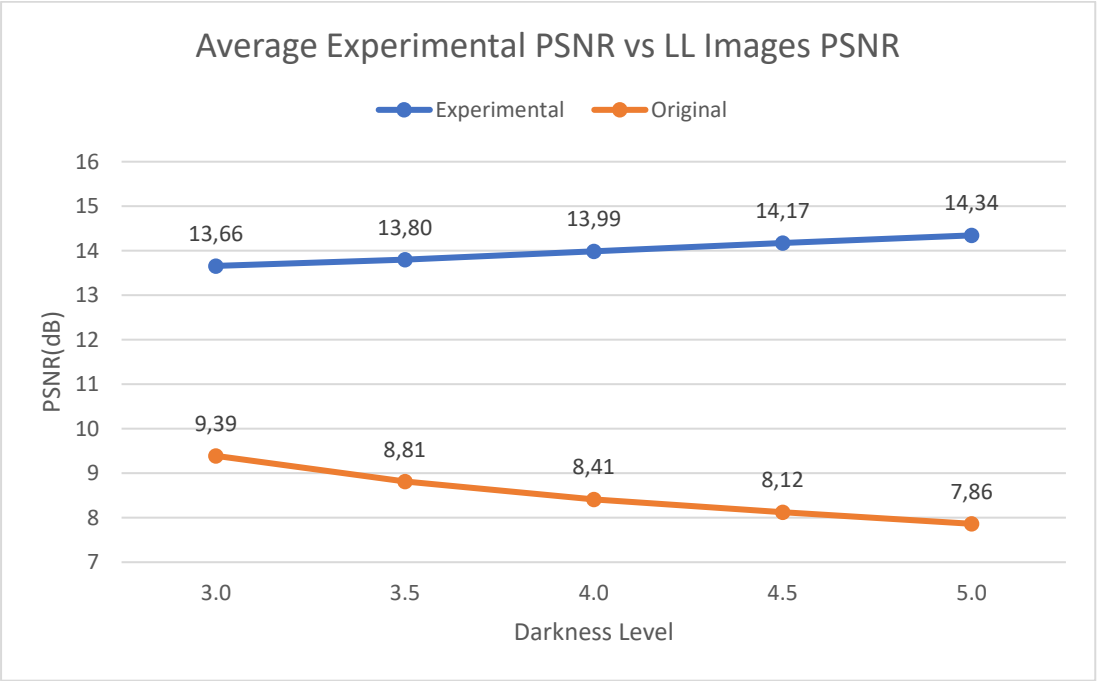


Figure 2.242: Experimental results vs LL Images PSNR

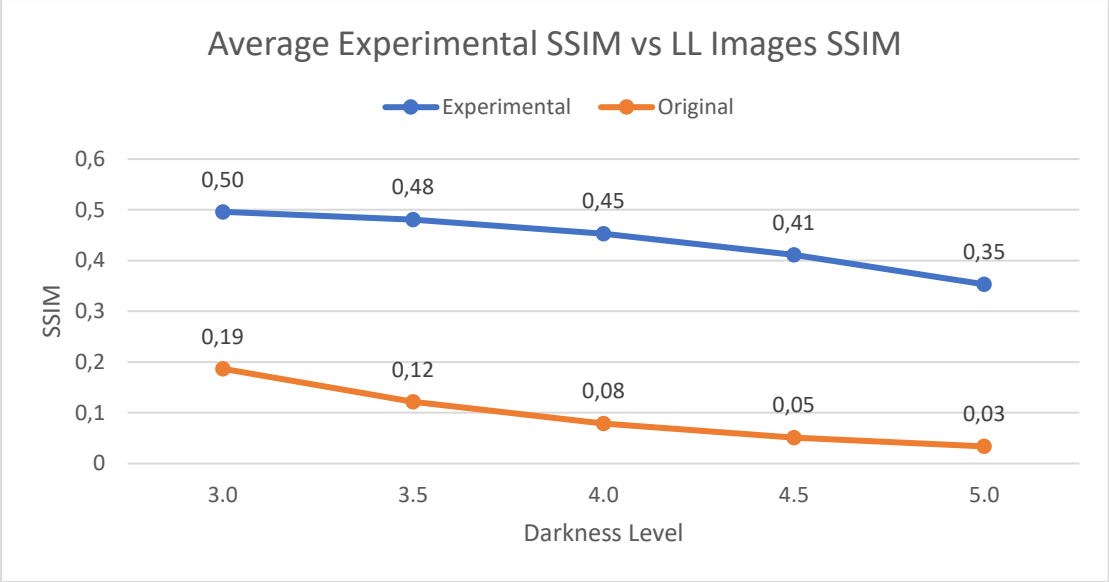


Figure 2.243: Experimental results vs LL Images SSIM

Validation Set

			MSE		
Level	3.0	3.5	4.0	4.5	5.0
MIN	623,1883076	623,749	617,2673	731,4565	783,202
MAX	7182,040078	7251,731	7427,834	7707,048	8060,856
AVERAGE	2769,145168	2725,16	2582,451	2467,49	2439,683
			PSNR(dB)		
Level	3.0	3.5	4.0	4.5	5.0
MIN	9,568325366	9,526387	9,422182	9,261923	9,066992
MAX	20,18461064	20,18071	20,22607	19,48892	19,19207
AVERAGE	14,3556775	14,43342	14,56165	14,6954	14,74779
			SSIM		
Level	3.0	3.5	4.0	4.5	5.0
MIN	0,236413655	0,226961	0,217423	0,140034	0,074072
MAX	0,661268243	0,652107	0,639521	0,613577	0,564632
AVERAGE	0,484409754	0,466641	0,450391	0,407962	0,366332
			MV		
Level	3.0	3.5	4.0	4.5	5.0
MIN	89,6359808	85,63742	78,46438	71,66087	66,51949
MAX	191,5559151	187,1986	173,8436	169,2548	161,553
AVERAGE	132,5484274	130,1981	126,6455	122,6473	119,1433
			STD		
Level	3.0	3.5	4.0	4.5	5.0
MIN	89,20510441	87,31398	155,5189	156,3432	182,4837
MAX	520,4262103	536,5168	561,4693	553,4804	510,1748
AVERAGE	310,2819892	301,3072	300,84	303,3144	314,9064

	BRISQUE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	20,39048869	13,77273	13,30271	9,728214	19,90264
MAX	44,75672323	45,07539	49,79098	51,09212	53,61671
AVERAGE	33,49175538	31,3582	32,26555	34,02478	36,77653
	NIQE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	2,753264454	2,945994	3,081627	3,568085	3,790819
MAX	5,450169652	5,528314	6,857618	7,060859	7,510053
AVERAGE	3,731817046	3,896872	4,198798	4,551309	4,993498

Table 2.37: MSR results on validation set

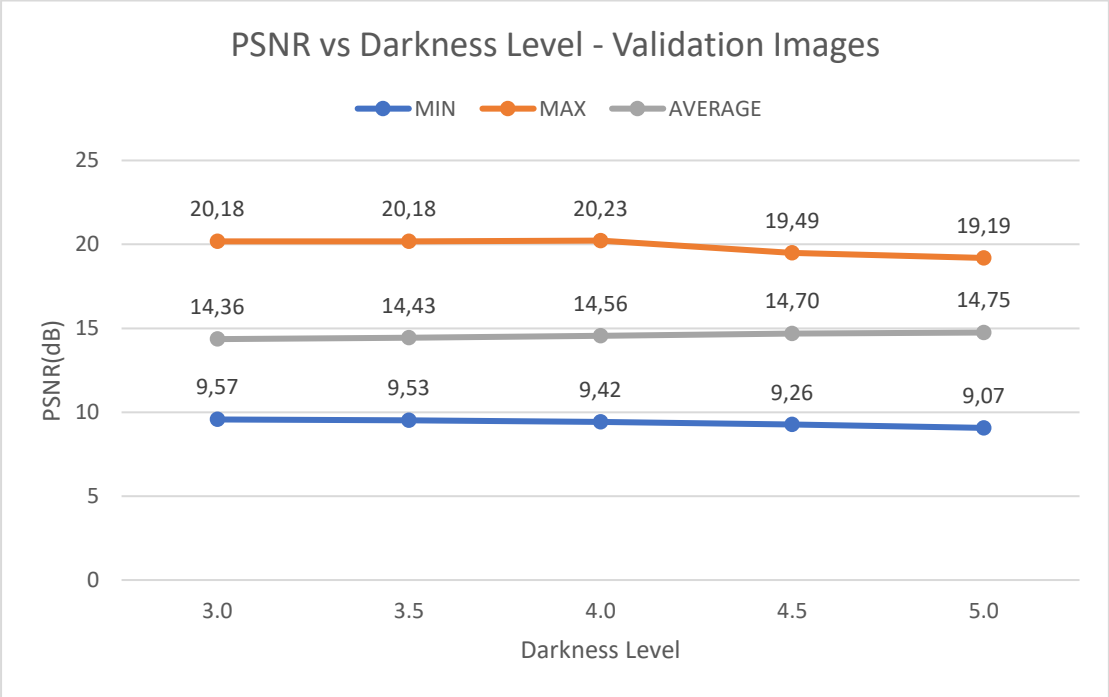


Figure 2.244: Experimental results PSNR vs Darkness level for validation set

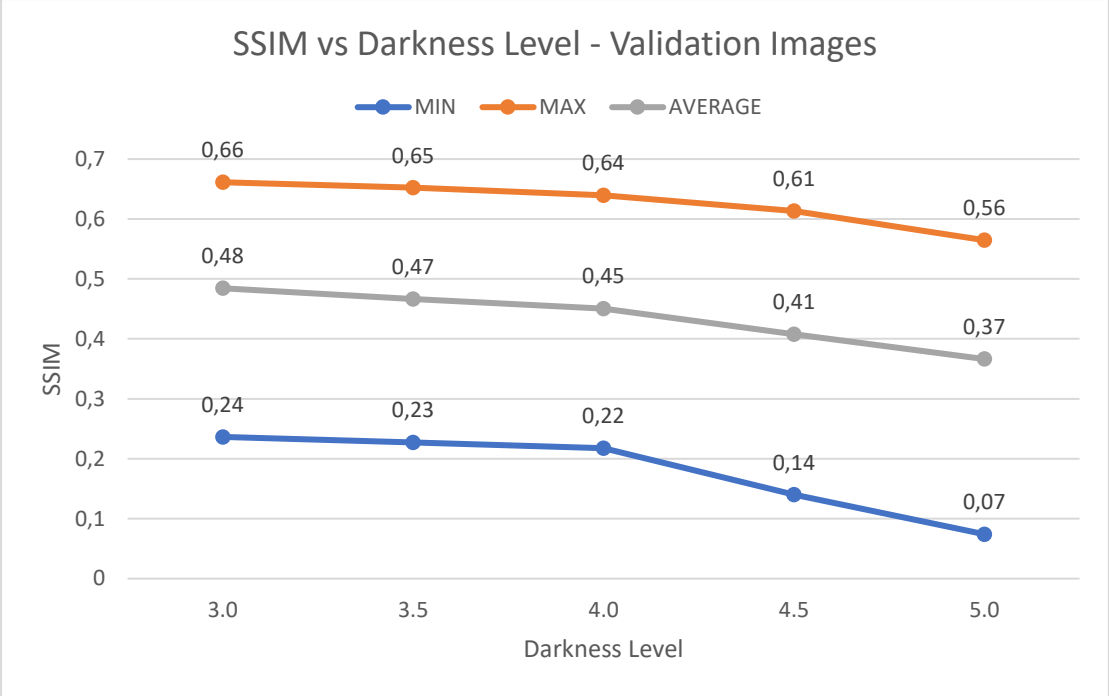


Figure 2.245: Experimental results SSIM vs Darkness level for validation set

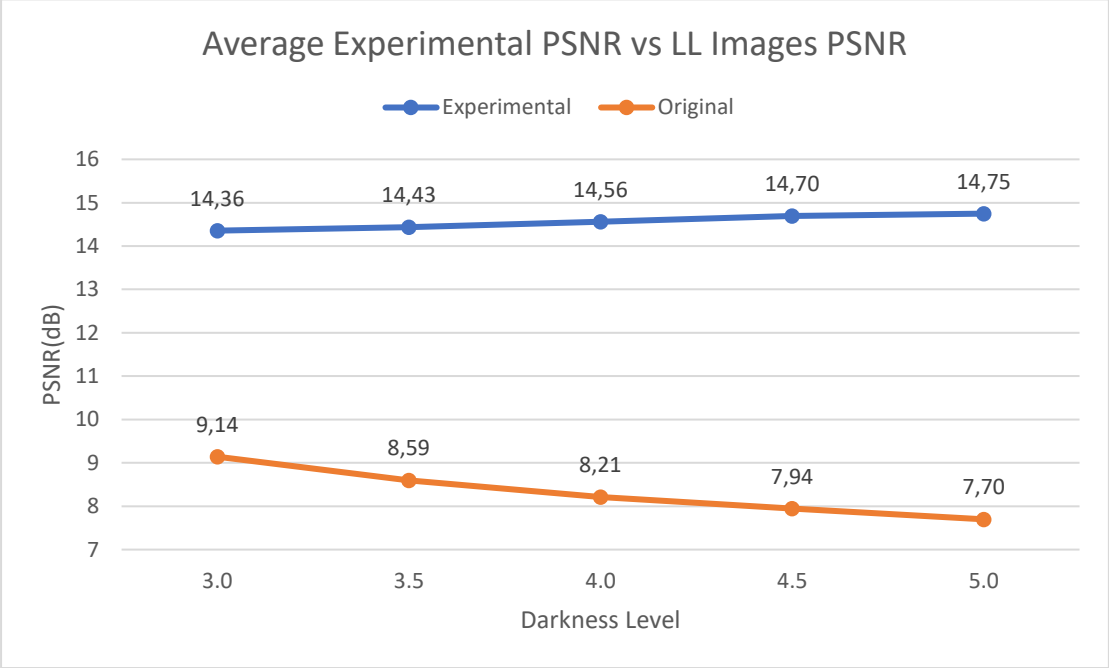


Figure 2.246: Experimental results vs LL Images PSNR

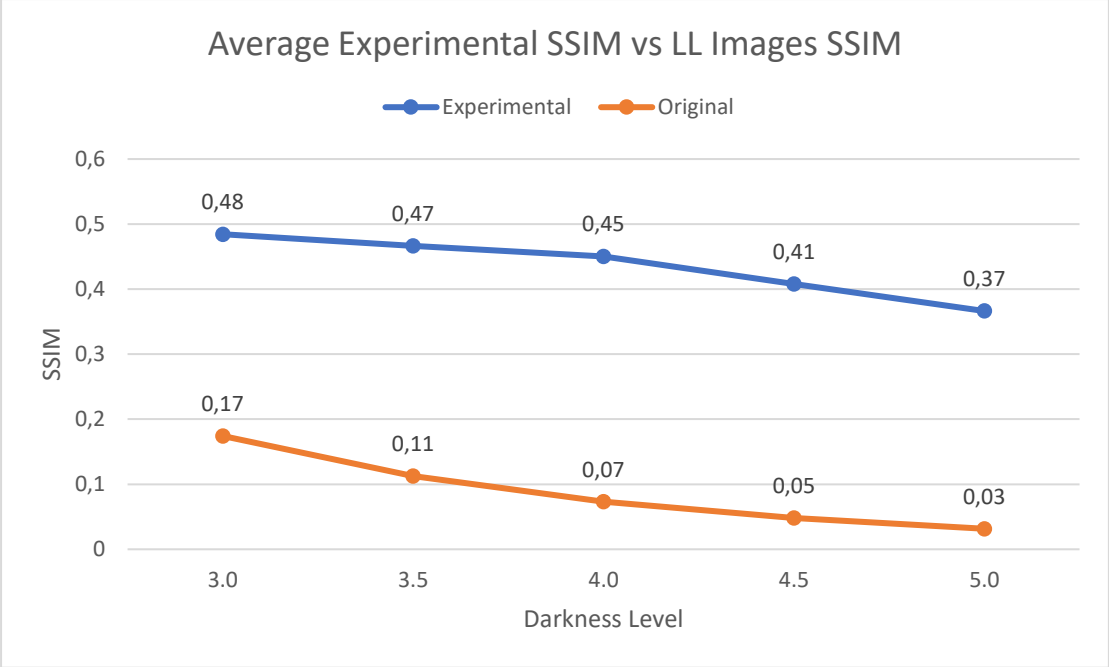


Figure 2.247: Experimental results vs LL Images SSIM

The same behavior as described above is observed in the validation set. MSE and PSNR improve with increasing darkness level, while SSIM decreases. As for the quality metrics with no reference, MV decreases while STD increases with increasing darkness level, meaning that the experimental results become slightly darker, with pixel values spread more widely around the mean value. Finally, both BRISQUE and NIQE follow an upward trend with the increase of the darkness level, so we move away from natural statistics, for the same reasons we mentioned above.

Test Set

MSE					
Level	3.0	3.5	4.0	4.5	5.0
MIN	595,1434001	454,3975	503,6561	746,8116	1056,331
MAX	12233,56487	10107,89	8248,23	6891,934	7025,85
AVERAGE	3172,983224	3034,013	2809,739	2604,865	2522,586
	PSNR(dB)				
Level	3.0	3.5	4.0	4.5	5.0
MIN	7,255273315	8,084198	8,967196	9,747393	9,663815
MAX	20,38458739	21,55644	21,10946	19,39869	17,8928
AVERAGE	13,8473982	14,0304	14,26856	14,49206	14,52812

	SSIM				
Level	3.0	3.5	4.0	4.5	5.0
MIN	-0,04963782	-0,00582	0,001933	0,03819	0,001959
MAX	0,804322955	0,812055	0,800866	0,75322	0,662687
AVERAGE	0,49444828	0,494892	0,465544	0,42746	0,359022
	MV				
Level	3.0	3.5	4.0	4.5	5.0
MIN	92,23430912	90,68733	80,99062	73,28001	66,8349
MAX	181,9084305	179,722	170,8068	165,9089	159,2396
AVERAGE	134,7027633	132,3205	128,5233	123,5492	116,8328
	STD				
Level	3.0	3.5	4.0	4.5	5.0
MIN	132,7014967	131,6914	140,3584	132,8207	130,7277
MAX	546,5499845	537,9095	531,1892	521,0306	548,5769
AVERAGE	324,7389508	320,5775	316,2193	318,1805	314,5282
	BRISQUE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	5,515242743	6,978605	7,375897	9,15082	19,42719
MAX	44,96910758	47,34848	53,40312	55,24281	56,39208
AVERAGE	31,62241267	31,72421	31,88589	33,40184	36,94428
	NIQE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	2,37114704	2,53802	2,896025	3,262259	3,546717
MAX	6,389086432	6,629393	7,433644	8,502454	9,709264
AVERAGE	3,748432604	3,916484	4,239833	4,686582	5,16485

Table 2.38: MSR results on test set

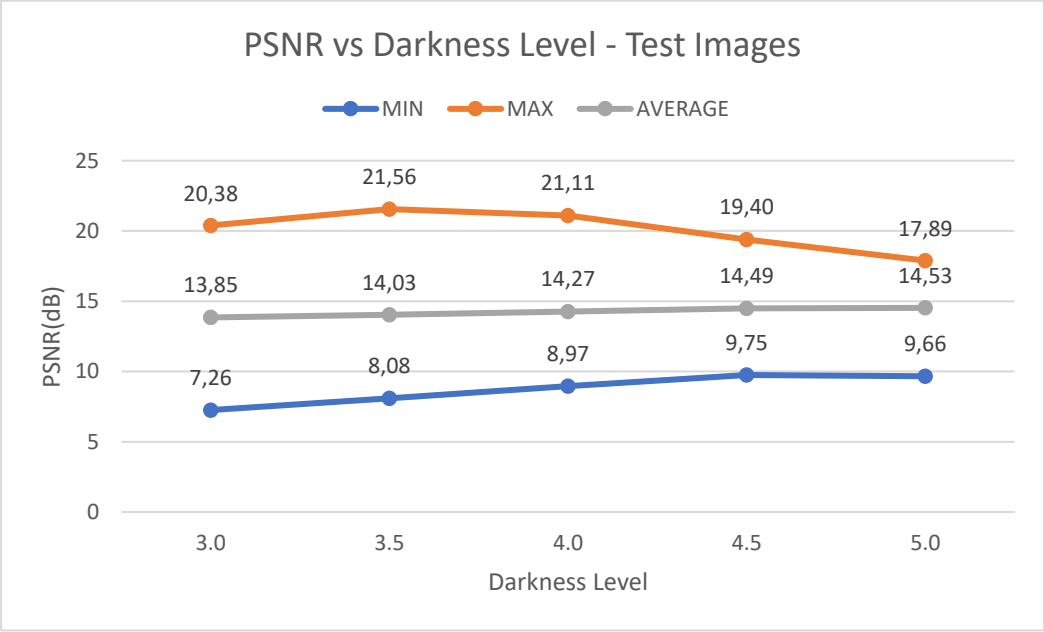


Figure 2.248: Experimental results PSNR vs Darkness level for test set

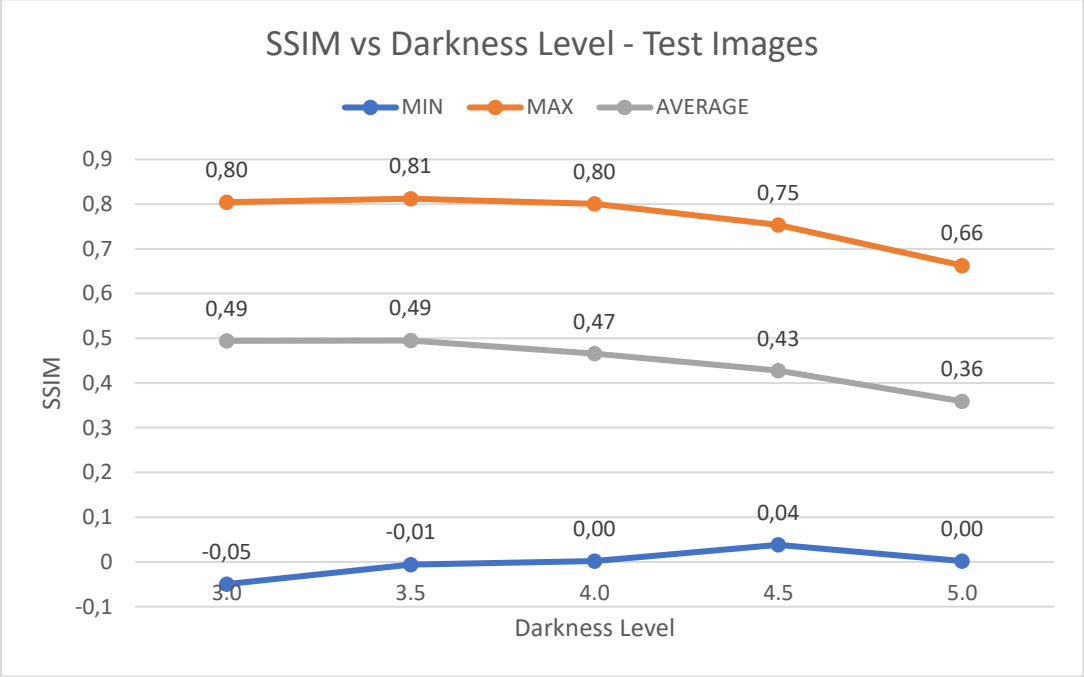


Figure 2.249: Experimental results SSIM vs Darkness level for test set

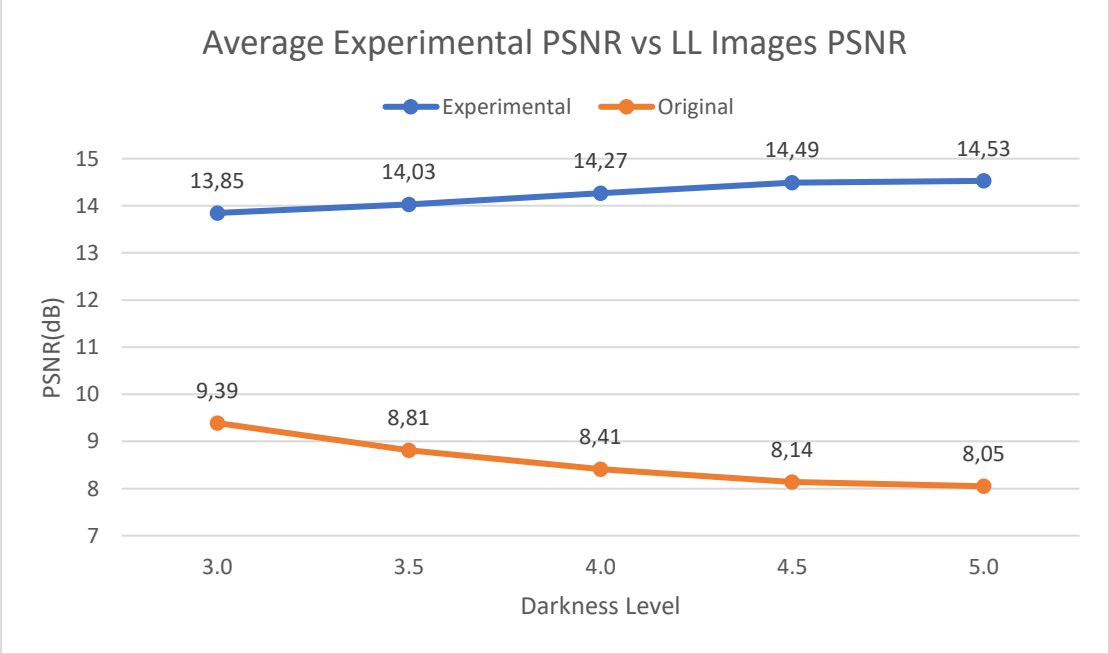


Figure 2.250: Experimental results vs LL Images PSNR

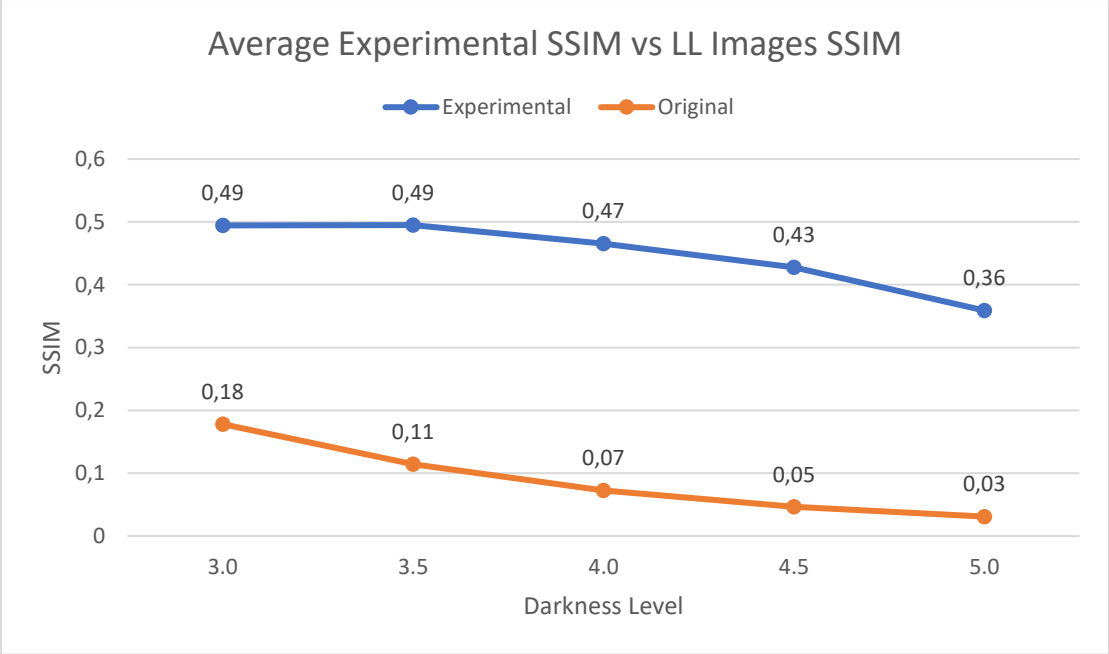


Figure 2.251: Experimental results vs LL Images SSIM

In the test set the same behavior as we mentioned above is observed. MSE and PSNR improve with increasing darkness level, while SSIM decreases. Accordingly, the MV decreases with the increase of the darkness level, while the STD remains approximately constant (but larger than the corresponding one in the LL case). Finally, BRISQUE and NIQE increase with increasing darkness level, which means we are moving away from natural statistics for the same reasons.

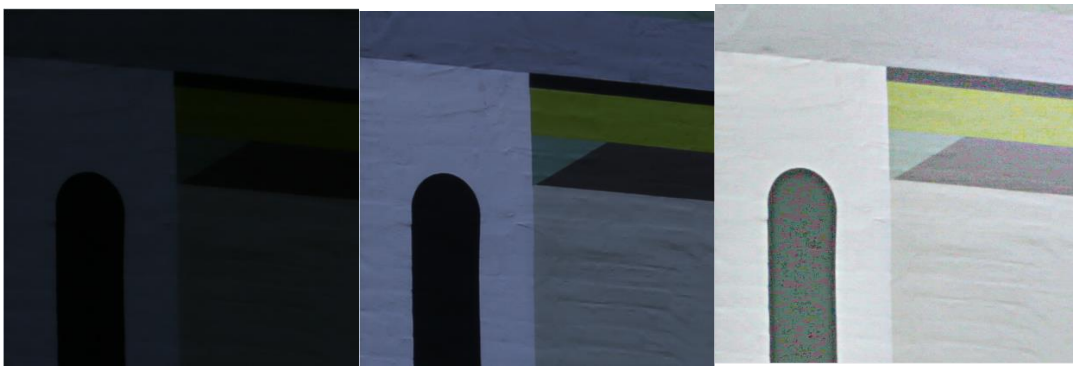
Commenting on the results as a whole, we noticed that the values of the quality metrics improved significantly by applying the method, with PSNR increasing by an average of 5.48dB and SSIM by 0.34. Moreover, it was observed that the increase of the darkness level has little effect on the values of the metrics, as the value change from level to level is very small, which means that the method is characterized by robustness to the changes of the darkness level, a characteristic inherited from the SSR algorithm. For the quality metrics with reference, we saw that MSE and PSNR improve with increasing darkness level while SSIM decreases. Based on this we expect there to be instances of images where dark areas have been over-enhanced, without preserving image texture, leading to these metric results. For the quality metrics with no reference, we see that MV decreases with increasing darkness level, while STD generally increases, meaning that images become slightly darker with increasing darkness level, but with pixel values spread over a wider range around average brightness. Finally,

BRISQUE and NIQE follow an upward path with the increase of the darkness level, which means that we are moving away from natural statistics. This is to be expected as with the increase of the darkness level the images become very dark making it difficult to recover the full visual information.

To see the results of the method visually we will display, for each darkness level, the image corresponding to the minimum experimental PSNR and the image corresponding to the maximum experimental PSNR, together with the LL image and ground truth cases. In addition, we will also display the corresponding histograms, so that we can evaluate the result.

Darkness Level: 3.0

MIN PSNR



Original Low Light **Normal Light** **Experimental Result**

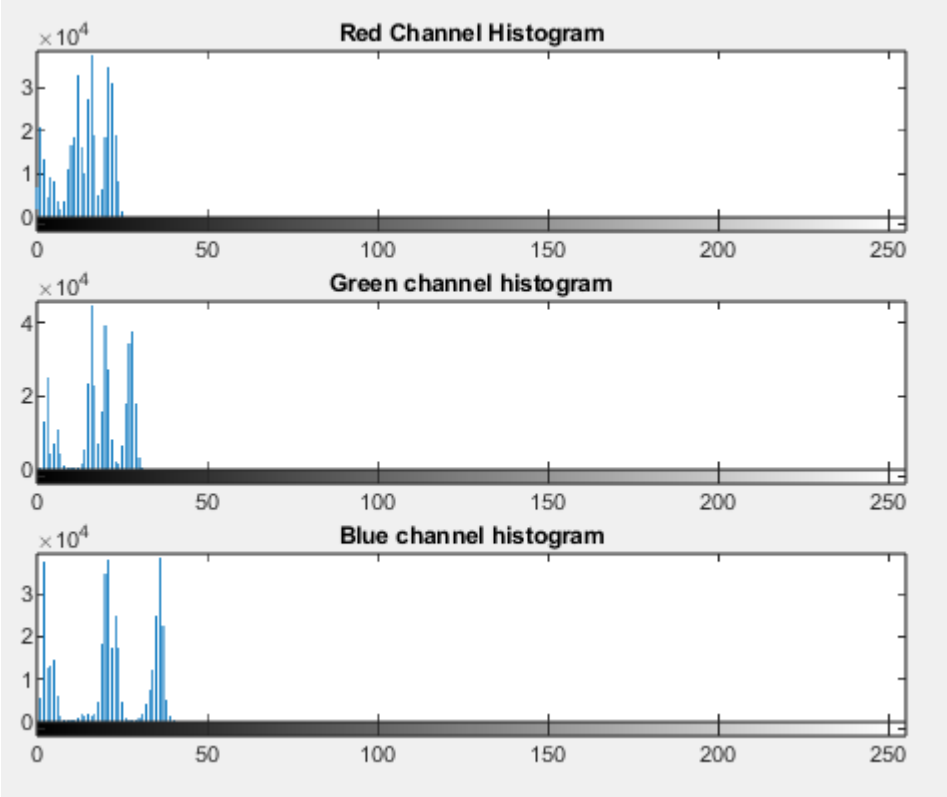


Figure 2.252: Histogram of LL image

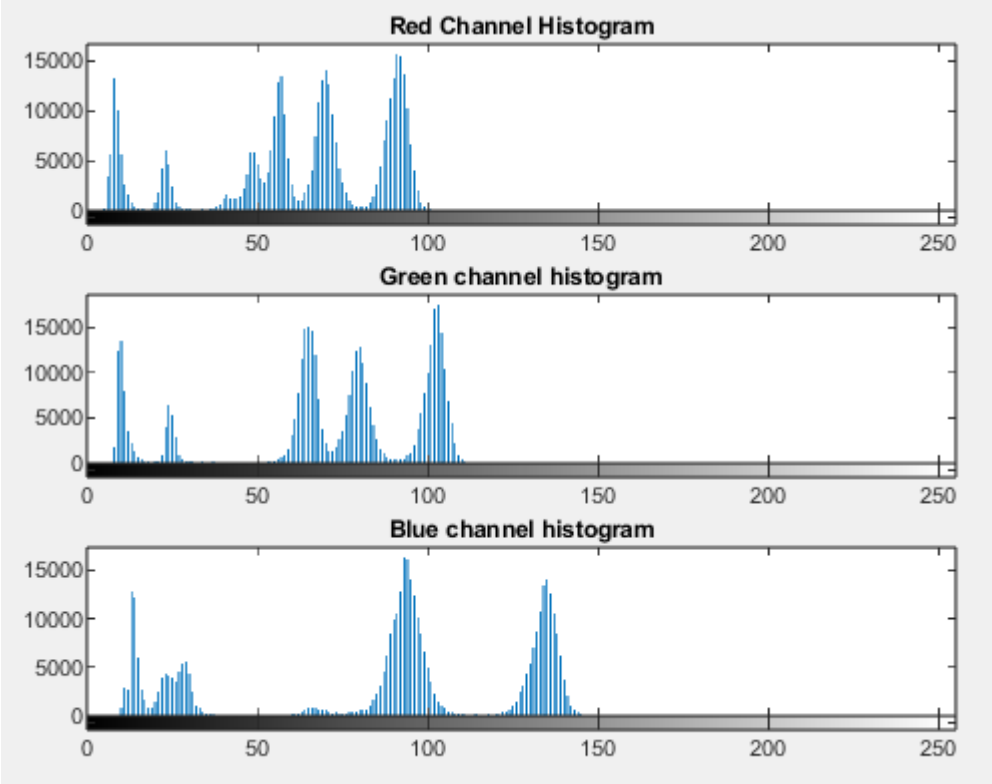


Figure 2.253: Histogram of NL image

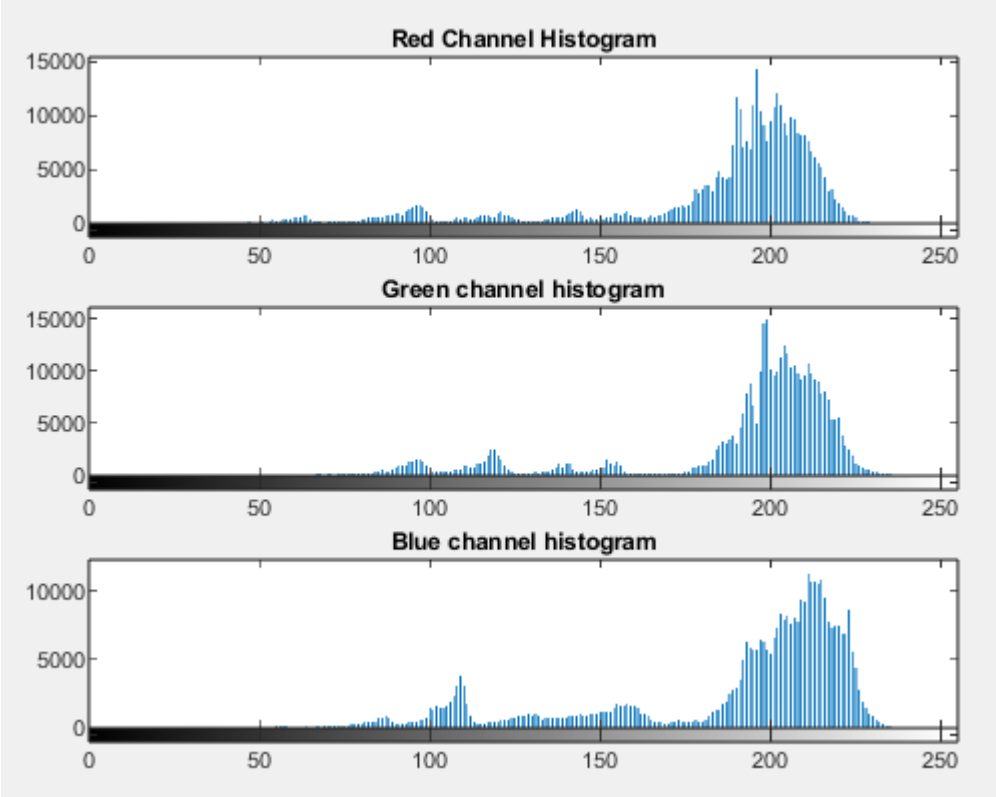


Figure 2.254: Histogram of 3.0 Experimental Result with min PSNR

For darkness level 3.0 in the case of the minimum experimental PSNR we see that the experimental result is characterized by a strong enhancement of the dark areas. If we compare it with the ground truth case it is as if the photo was taken in bright lighting conditions, confirming what we mentioned above. The same conclusion can be reached from the histograms. In the histogram of the ground truth case (figure 2.253) the values are distributed in the gray value levels less than 150, while in the experimental histogram (figure 2.254) the values are accumulated in the right part of the histogram. Nevertheless, the original dark image has been enhanced, recovering the color and visual information, just more than it should be compared to the ground truth case. At higher darkness levels we expect this over-amplification to lead to color distortions as well.

MAX PSNR

Original Low Light **Normal Light** **Experimental Result**

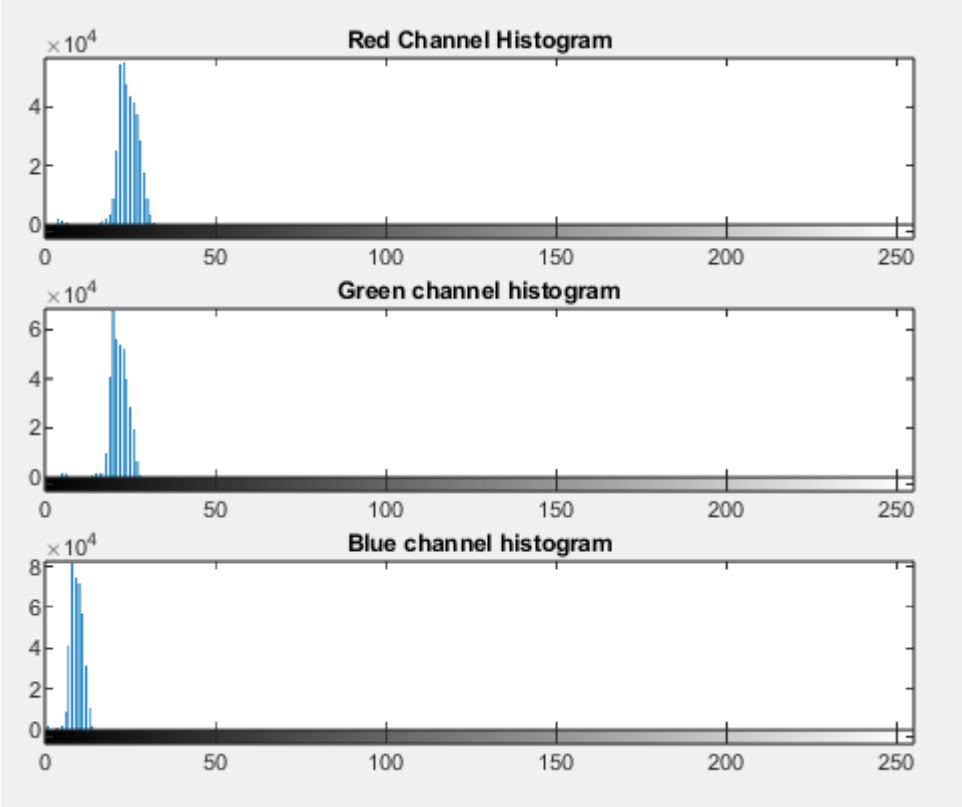


Figure 2.255: Histogram of LL image

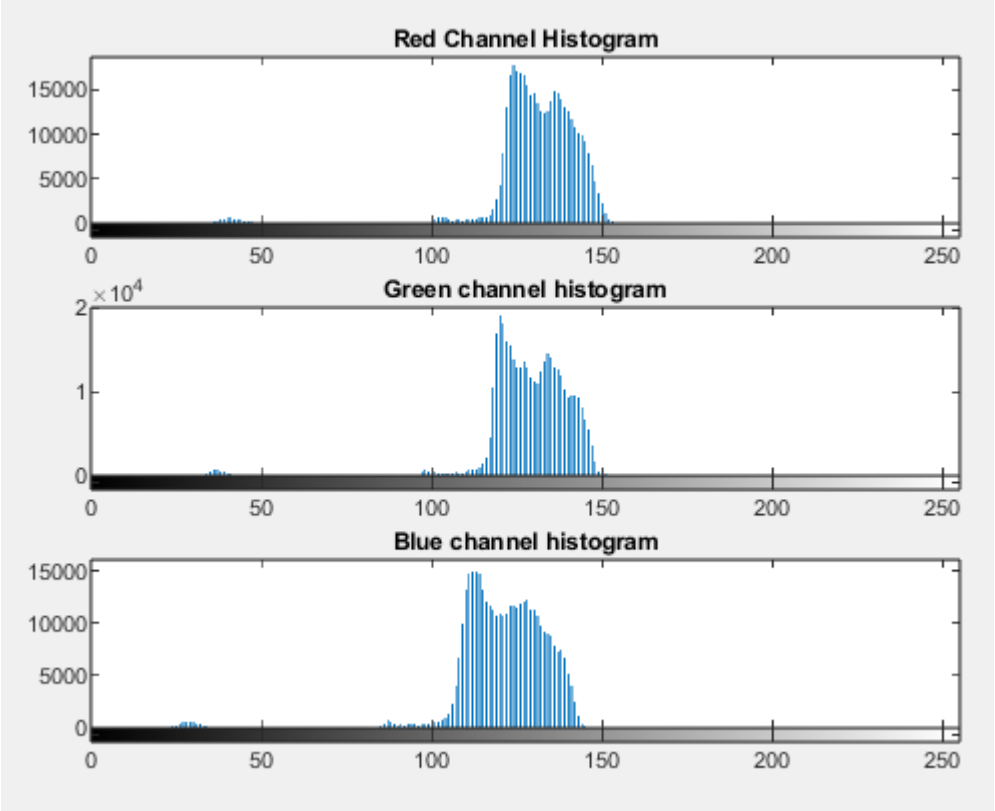


Figure 2.256: Histogram of NL image

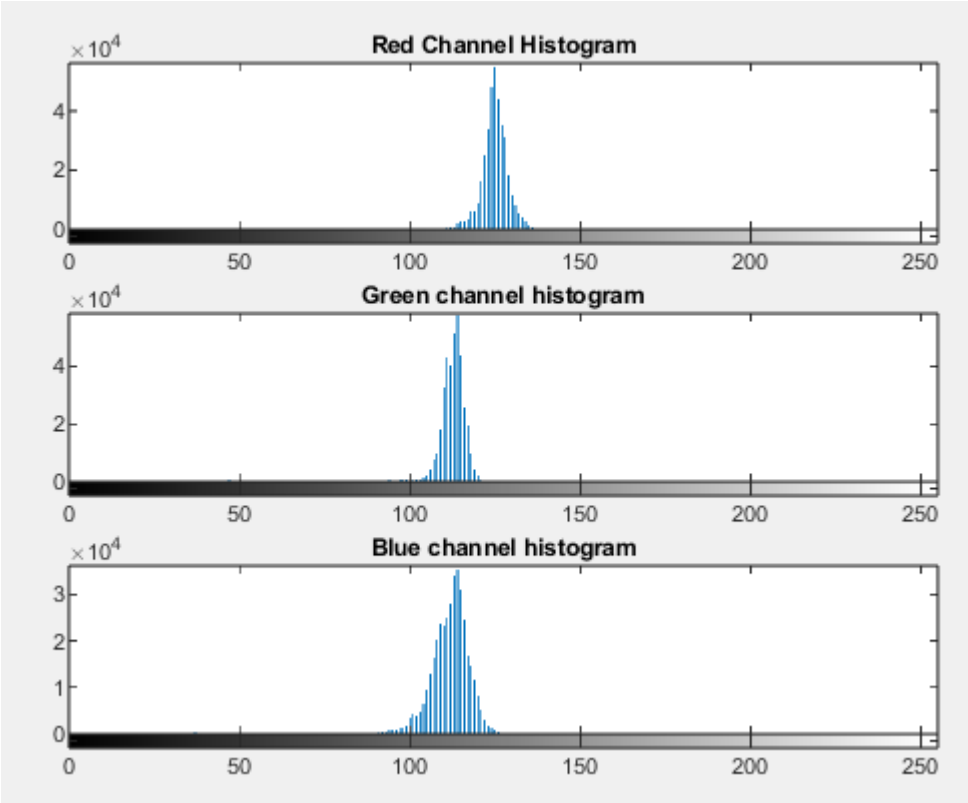
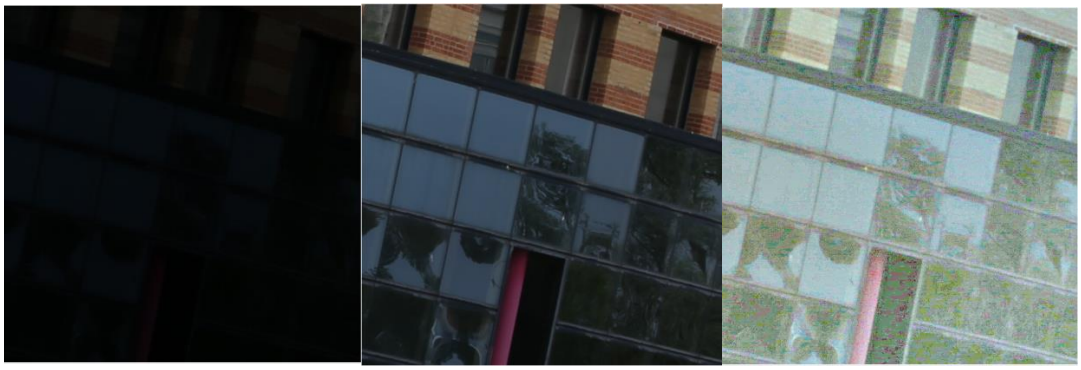


Figure 2.257: Histogram of 3.0 Experimental Result with max PSNR

In the case of the maximum PSNR we see that in the experimental result the visual information has been fully recovered, which can also be confirmed by the experimental histogram. From figure 2.257 we see that the shape of the experimental histogram is very close to the shape of the ground truth case (figure 2.256).

Darkness Level: 3.5

MIN PSNR



Original Low Light **Normal Light** **Experimental Result**

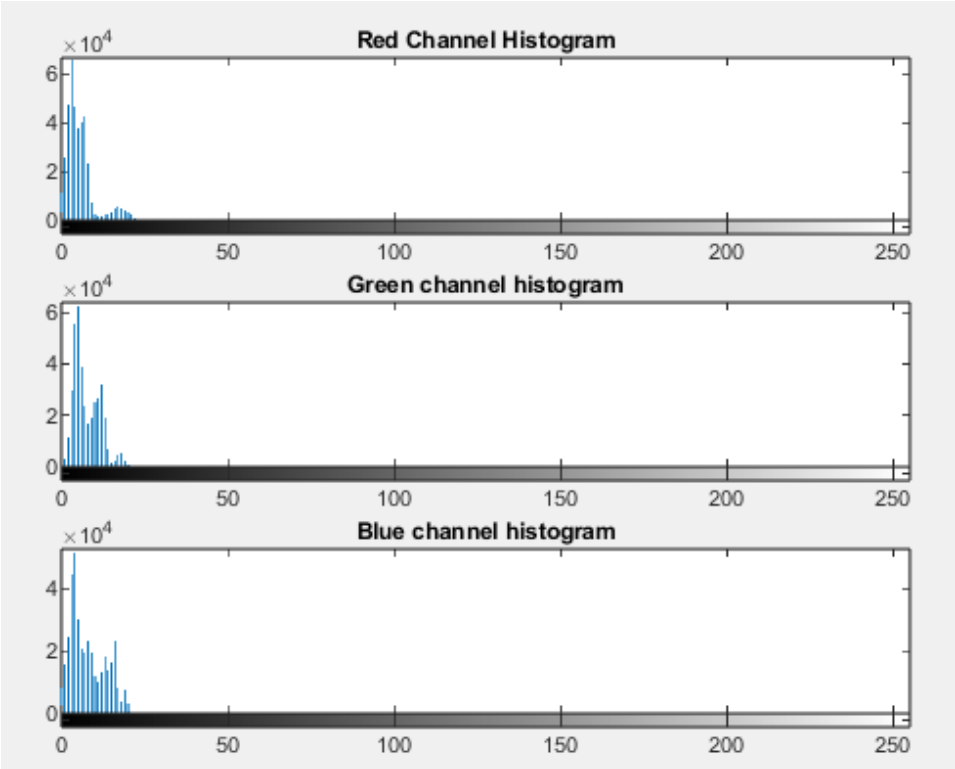


Figure 2.258: Histogram of LL image

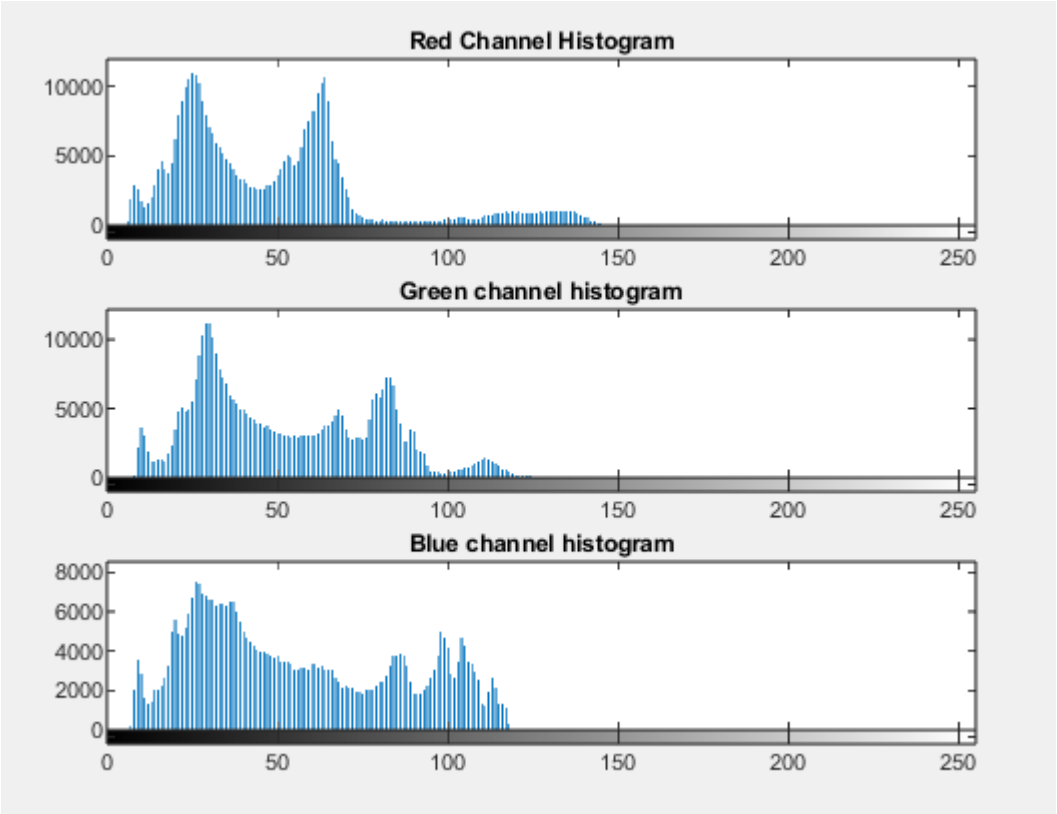


Figure 2.259: Histogram of NL image

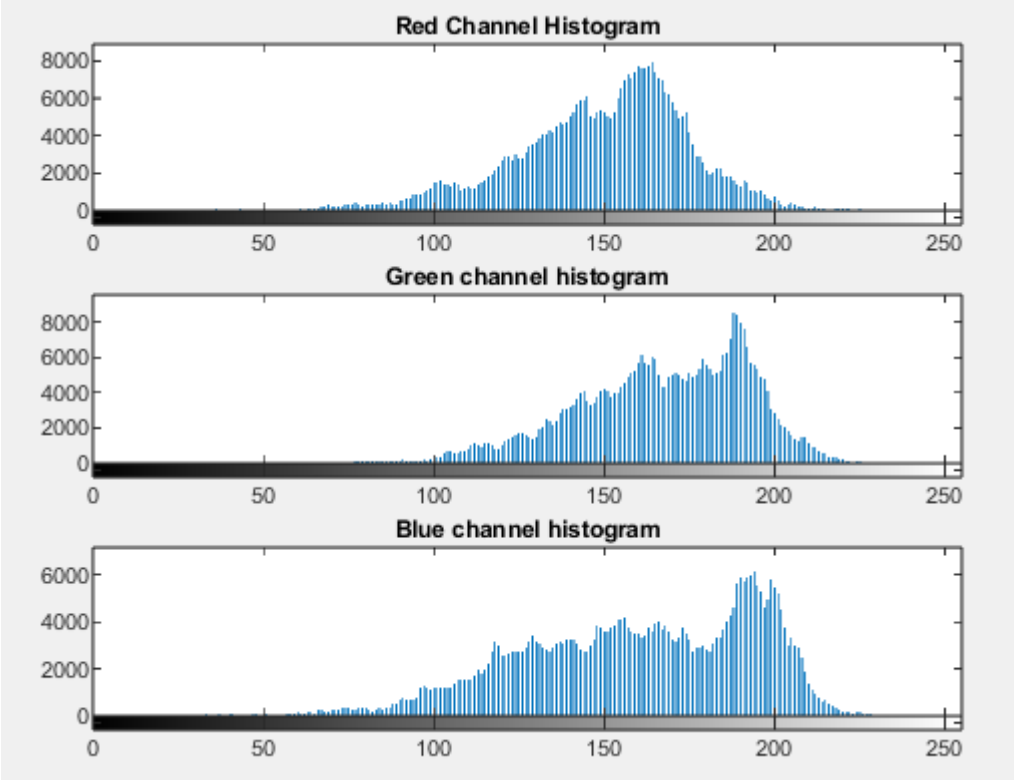


Figure 2.260: Histogram of 3.5 Experimental Result with min PSNR

For darkness level 3.5 in the case of the minimum experimental PSNR, the same problem is observed as we saw at darkness level 3.0, i.e. the dark areas have been enhanced more than they should be. Here we see that color distortions are also introduced, which we mentioned above. From the histogram we see that, while in the ground truth case (figure 2.259) the values are distributed up to gray value level 150, in the experimental case (figure 2.260) the values are accumulated in the right part of the histogram, explaining also the over-amplification of dark areas.

MAX PSNR

Original Low Light **Normal Light** **Experimental Result**

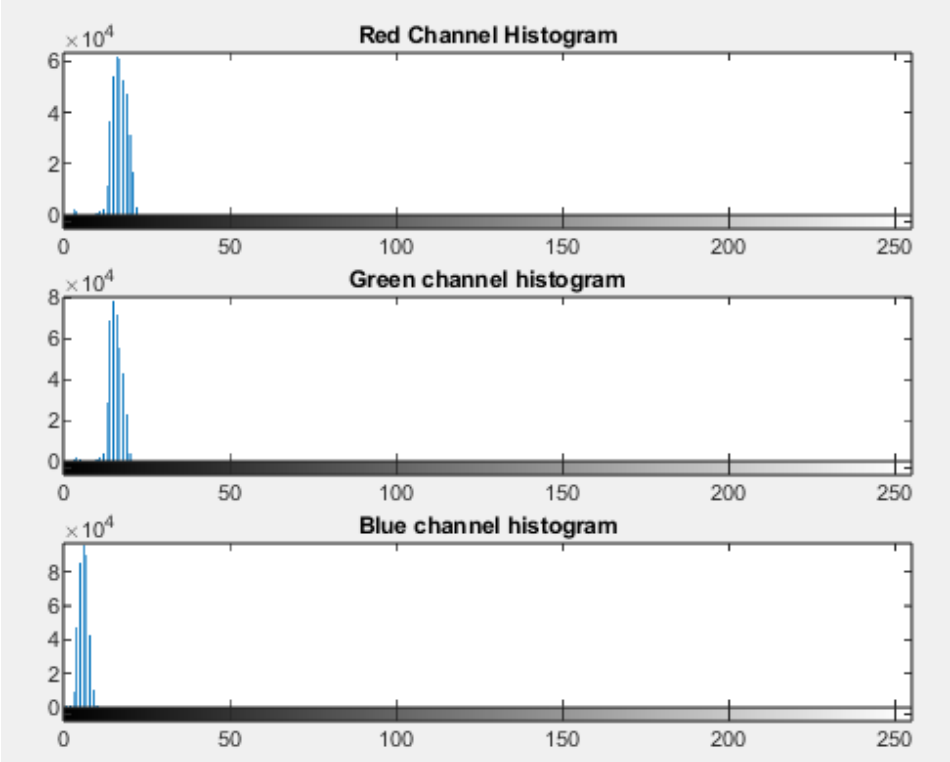


Figure 2.261: Histogram of LL image

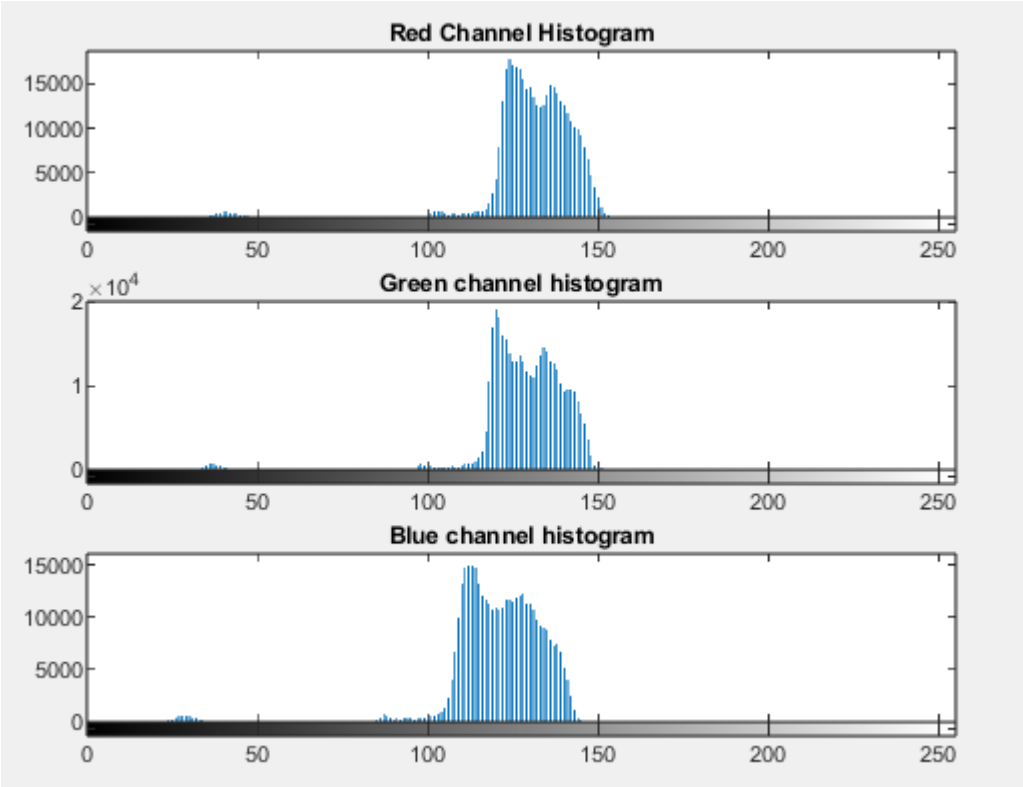


Figure 2.262: Histogram of NL image

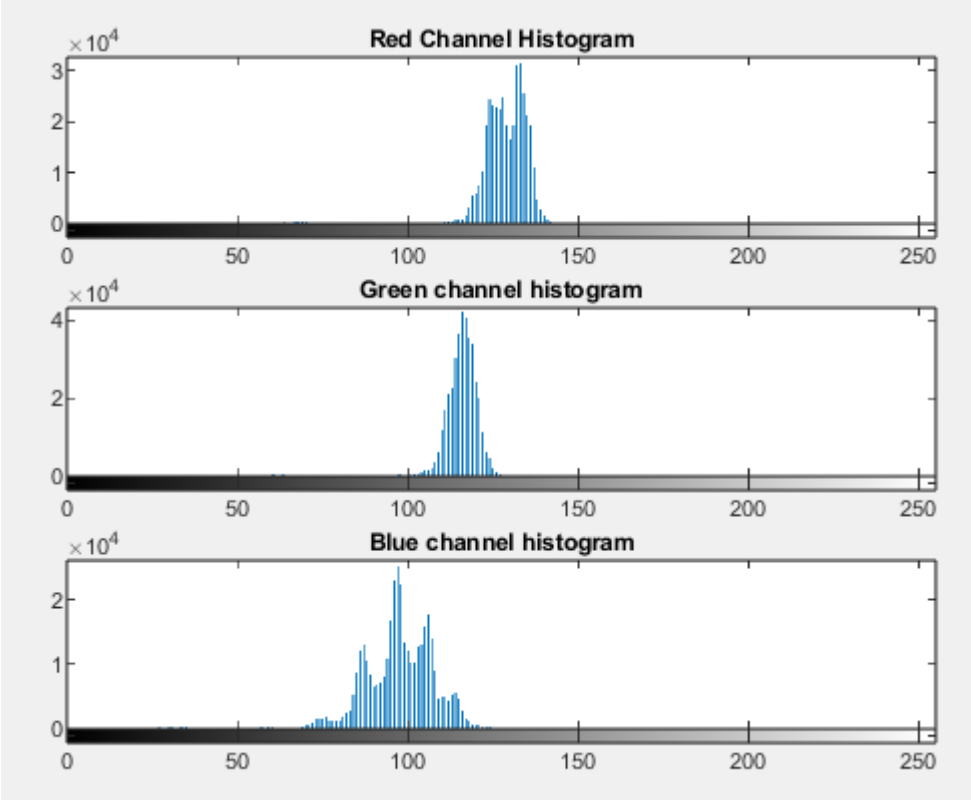
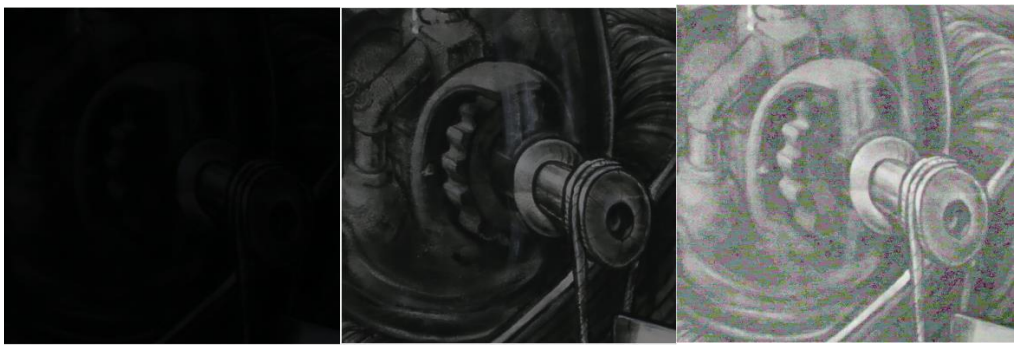


Figure 2.263: Histogram of 3.5 Experimental Result with max PSNR

In the case of the maximum experimental PSNR, the same image is obtained with darkness level 3.0. We see that the method has recovered most of the visual information, which is also confirmed by the histograms. From figure 2.263 with the experimental histogram we see that its form is very close to the ground truth case (figure 2.262) confirming the high quality of the result.

Darkness Level: 4.0

MIN PSNR



Original Low Light **Normal Light** **Experimental Result**

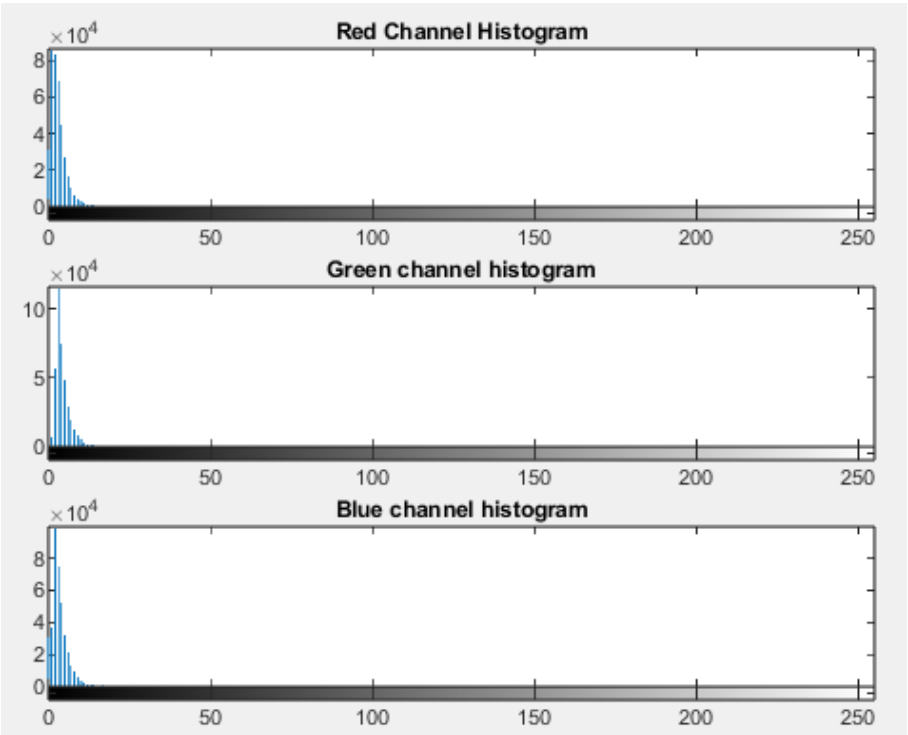


Figure 2.264: Histogram of LL image

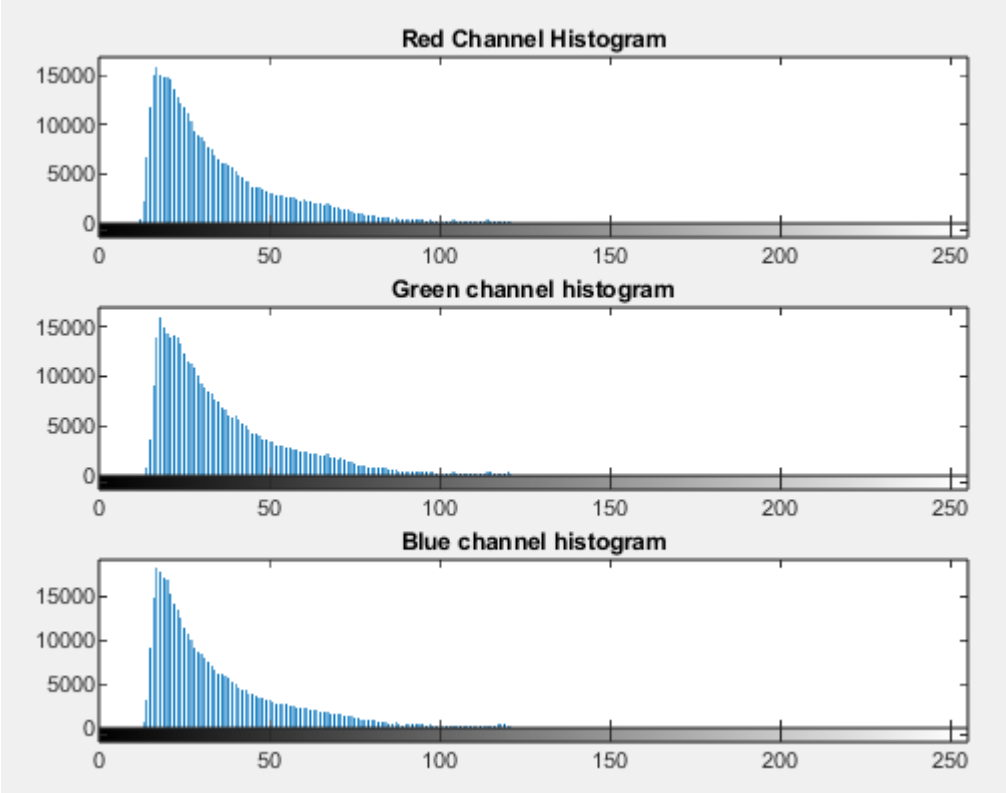


Figure 2.265: Histogram of NL image

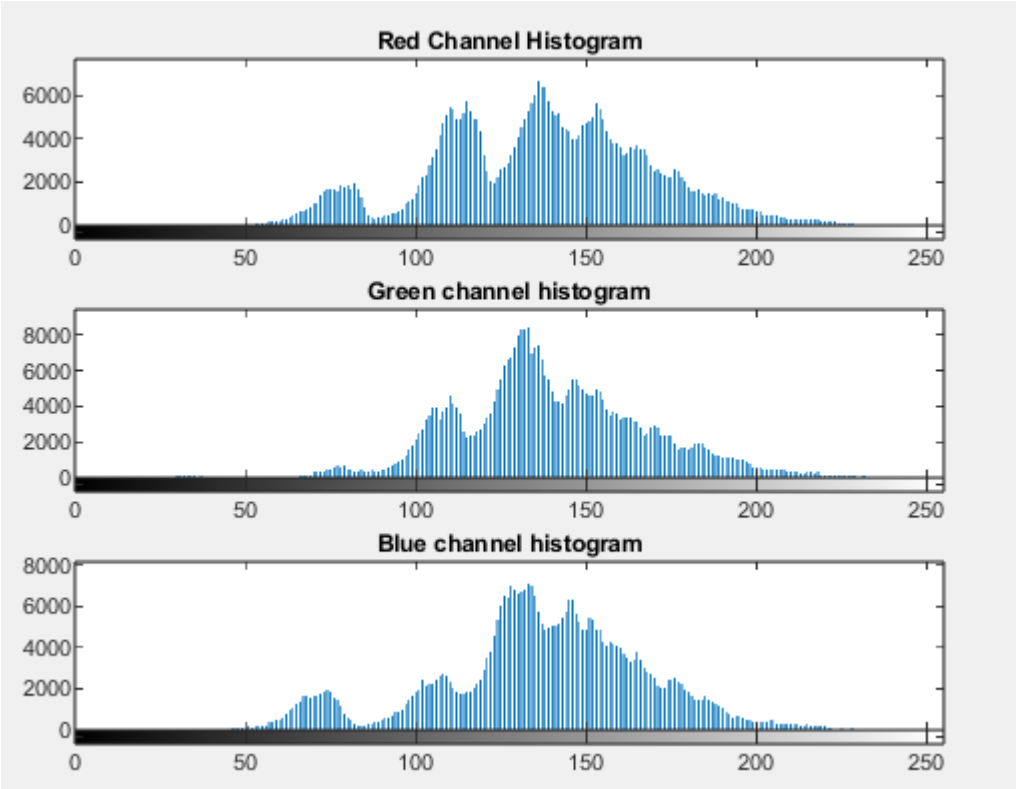


Figure 2.266: Histogram of 4.0 Experimental Result with min PSNR

For darkness level 4.0 in the case of the minimum experimental PSNR we see that again the image has been enhanced more than it should be, thus introducing additional noise and color distortions. We can see this from the corresponding histograms, where in figure 2.265 with the histogram of the ground truth case the pixels are distributed up to the gray value level 100, while in the experimental histogram (figure 2.266) we see that they are distributed throughout the available range of values . This results in the experimental image having greater contrast and brightness than the ground truth case.

MAX PSNR

Original Low Light **Normal Light** **Experimental Result**

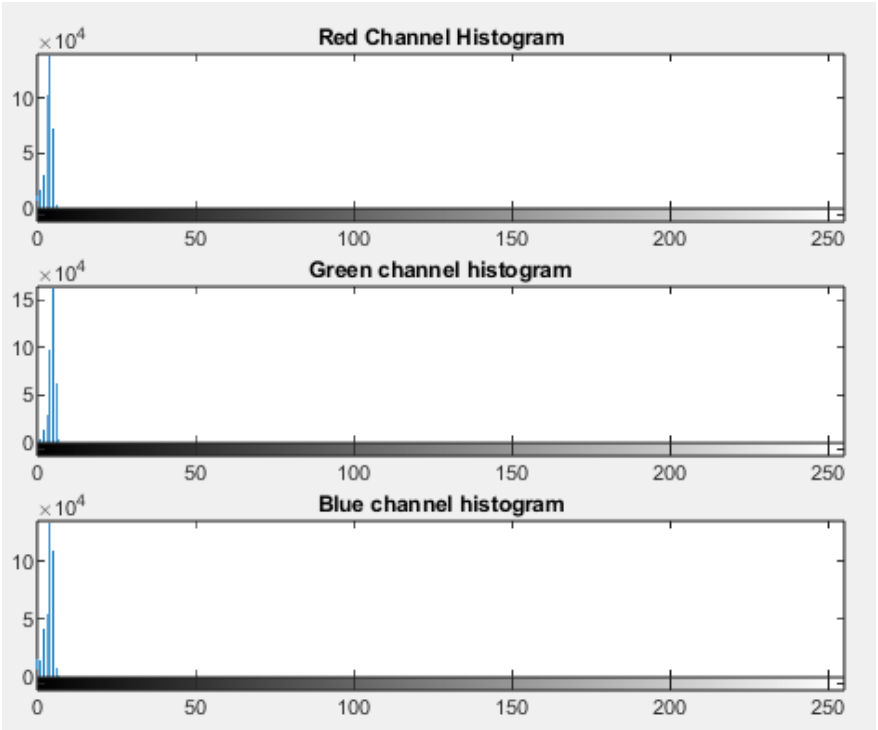


Figure 2.267: Histogram of LL image

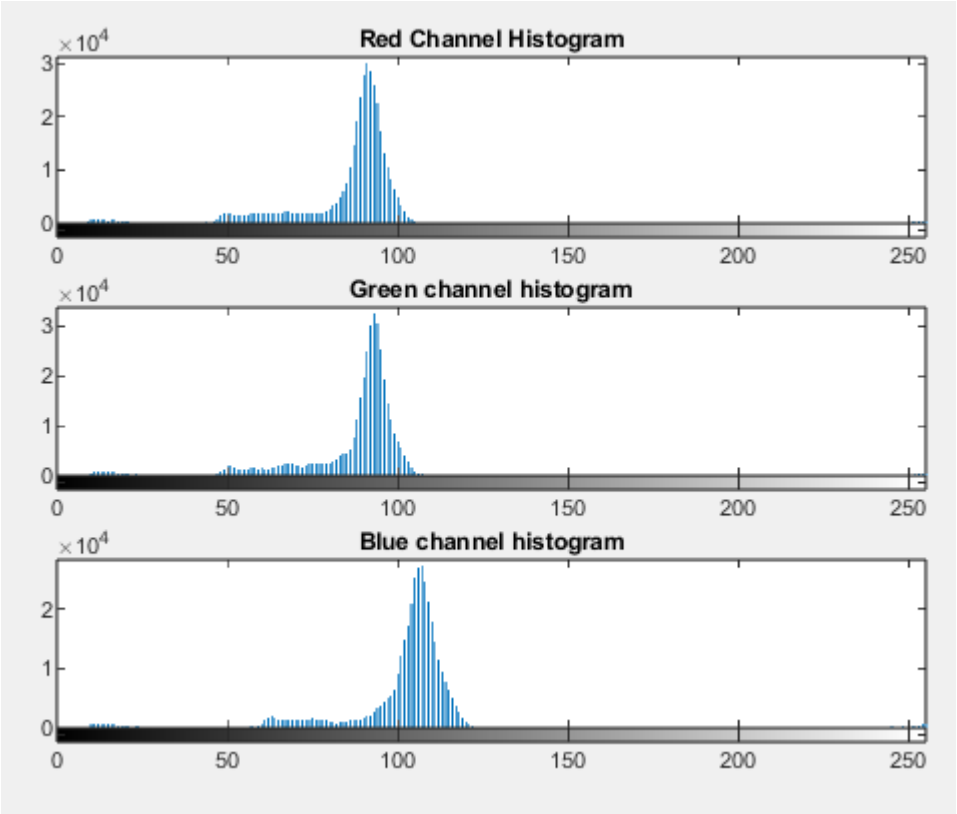


Figure 2.268: Histogram of NL image

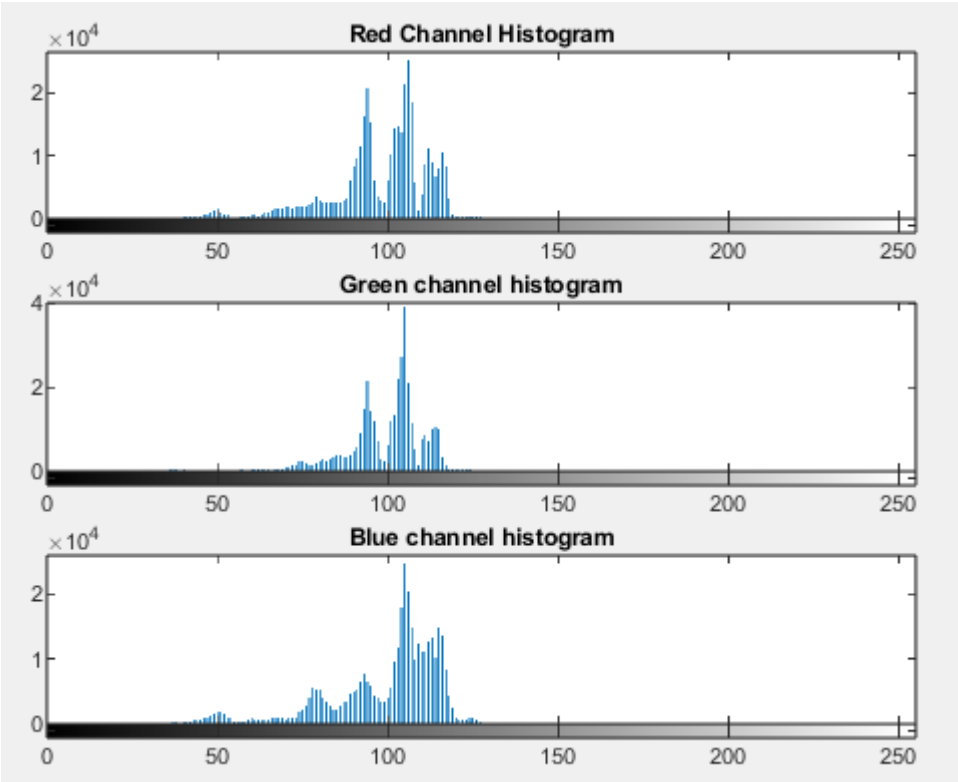


Figure 2.269: Histogram of 4.0 Experimental Result with max PSNR

In the case of the maximum experimental PSNR the experimental result is optically very close to the ground truth case, but we have to notice that color distortions have started to appear, which means that we recover less visual information. This is due to the fact that we now combine the results from 3 different SSRs with different parameter values, and the result corresponding to parameter c=10 causes these distortions. We recall that for c=10 in the case of SSR we recovered the texture of the image but not the color information, which seems to appear here too, just less strongly. Nevertheless, the experimental histogram (figure 2.269) has a form very similar to that of the ground truth case (figure 2.268).

Darkness Level: 4.5

MIN PSNR

Original Low Light **Normal Light** **Experimental Result**

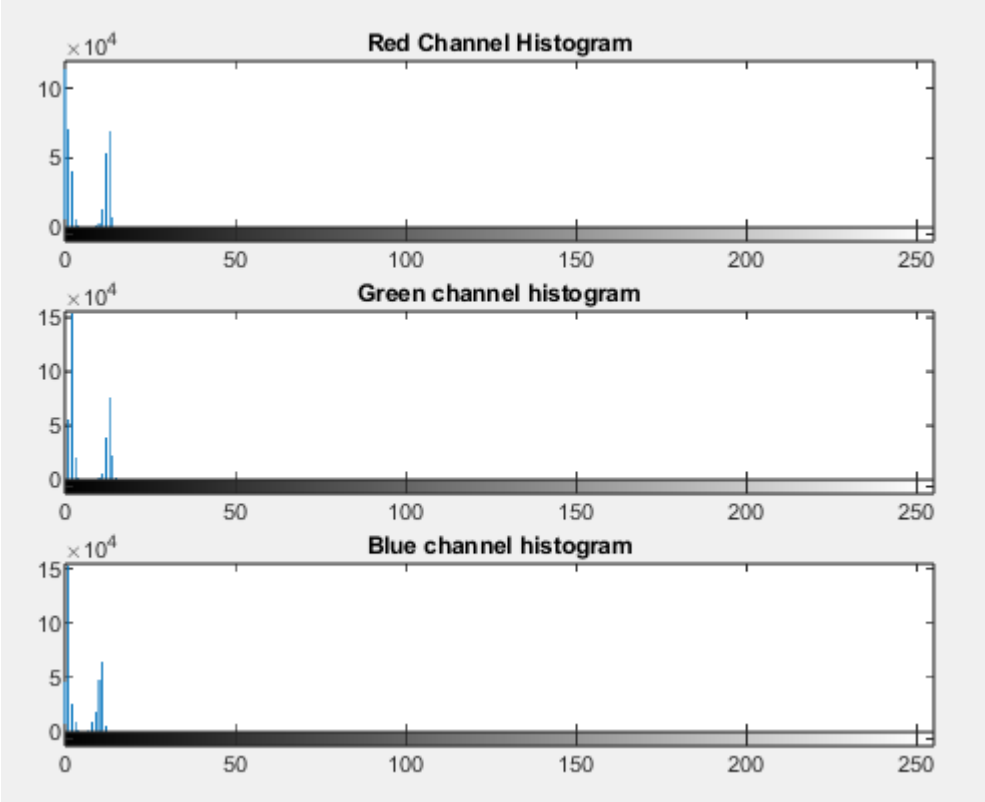


Figure 2.270: Histogram of LL image

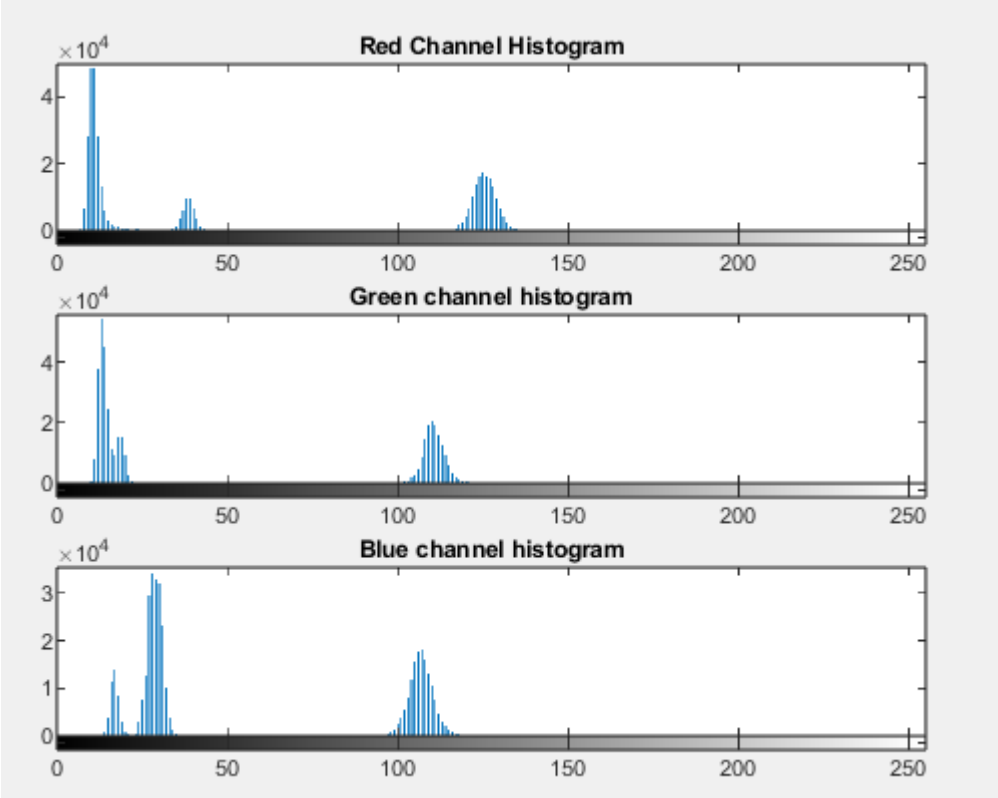


Figure 2.271: Histogram of NL image

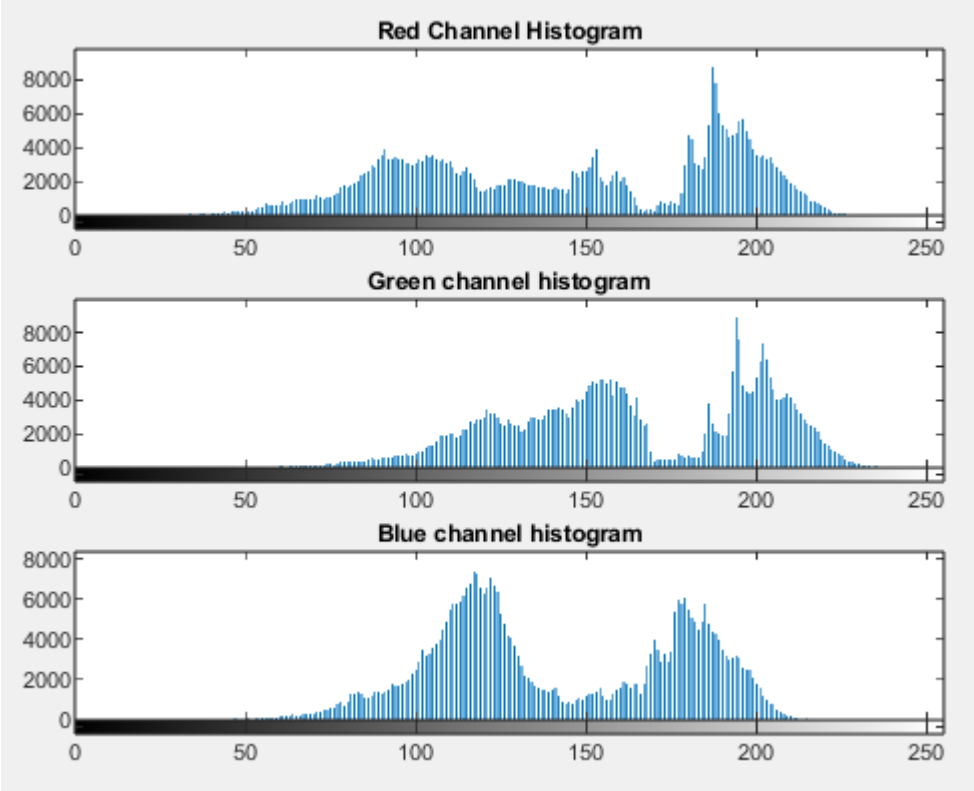


Figure 2.272: Histogram of 4.5 Experimental Result with min PSNR

For darkness level 4.5 in the case of the minimum experimental PSNR we again observe that the experimental result is characterized by intense color distortions due to the fact that there is an over-enhancement of the dark areas of the image. This is also confirmed by the histograms as in the ground truth histogram (figure 2.271) we see that the pixel values are concentrated below the gray level 120-150, while in the experimental case (figure 2.272) they are distributed throughout the available range. That is, it is characterized by greater contrast and brightness, explaining the experimental result.

MAX PSNR

Original Low Light **Normal Light** **Experimental Result**

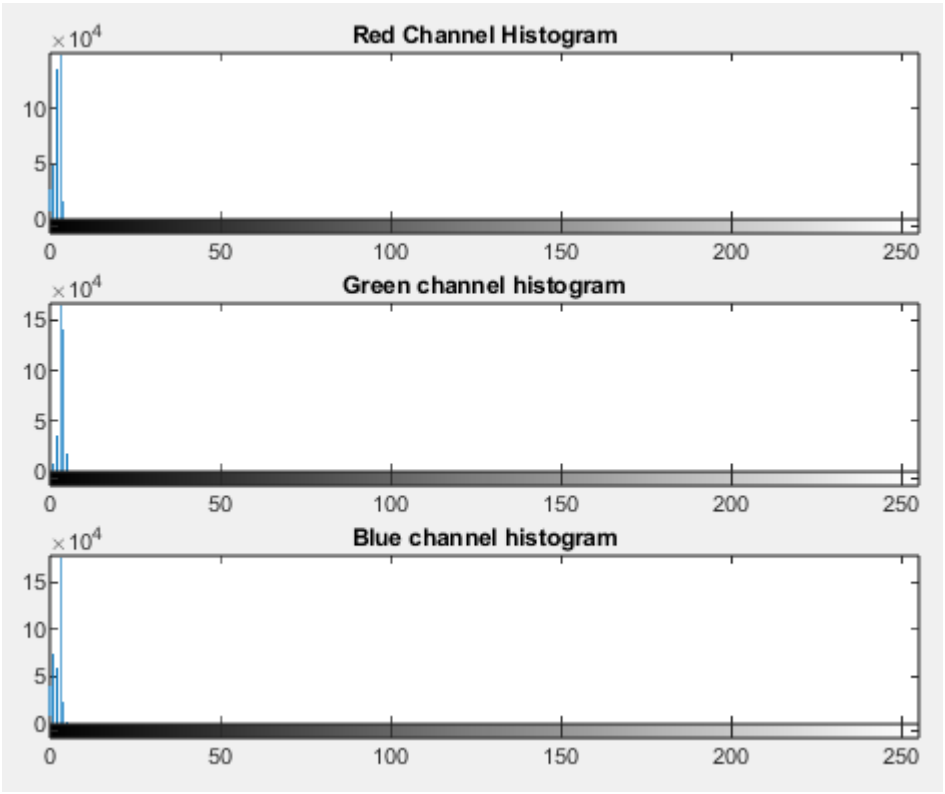


Figure 2.273: Histogram of LL image

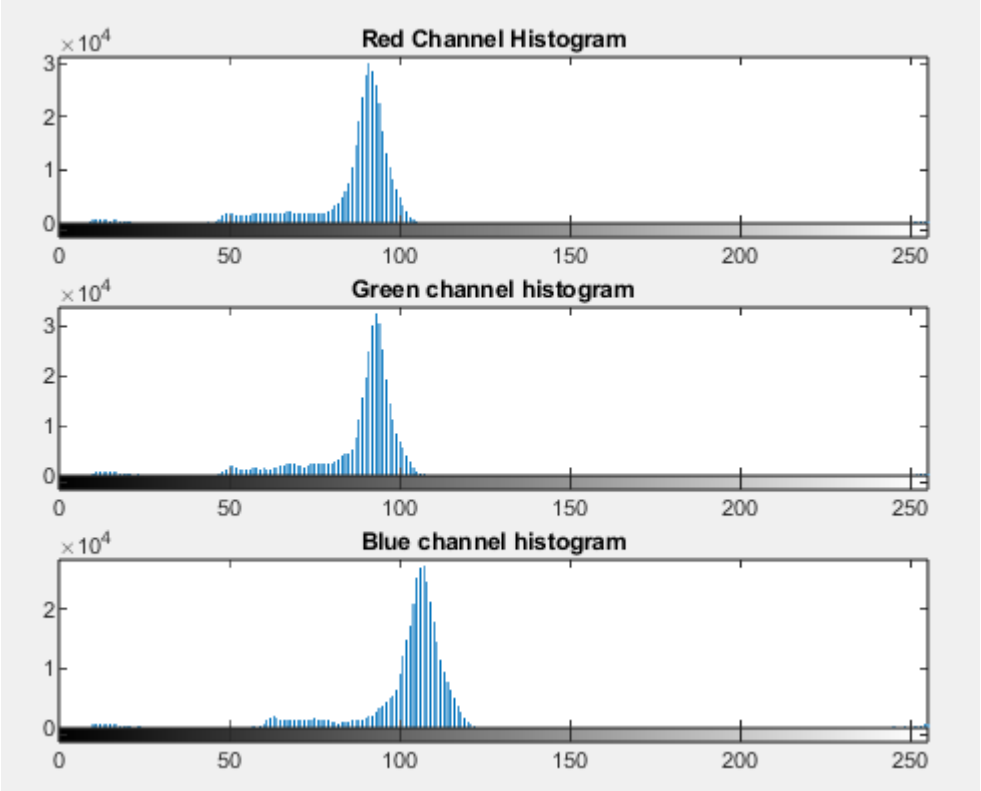


Figure 2.274: Histogram of NL image

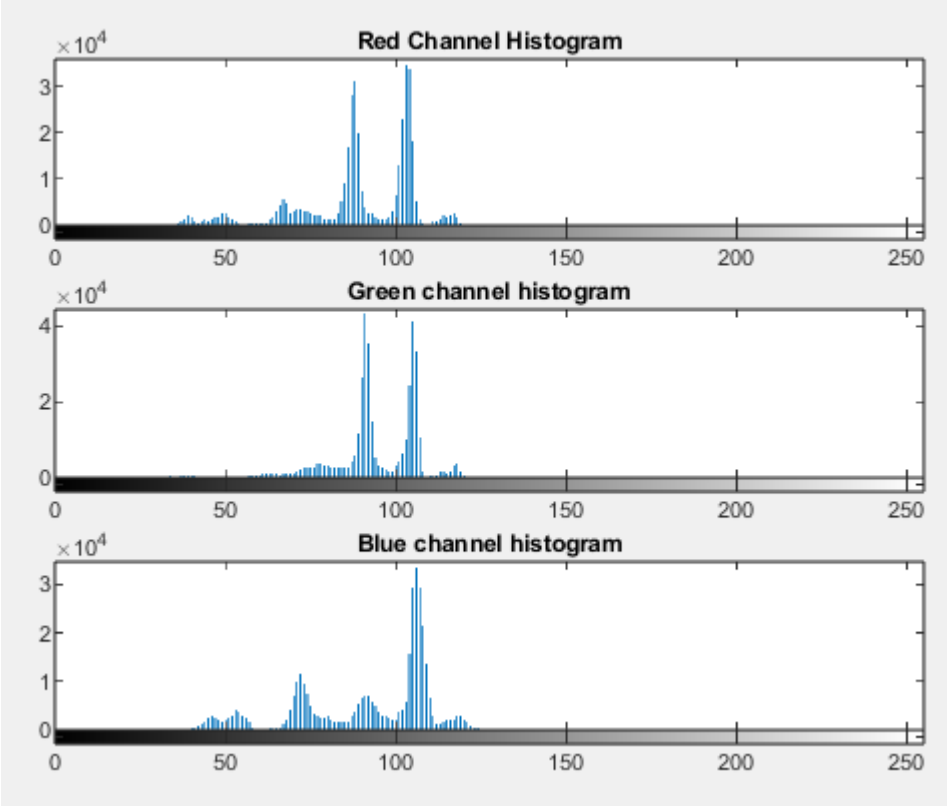
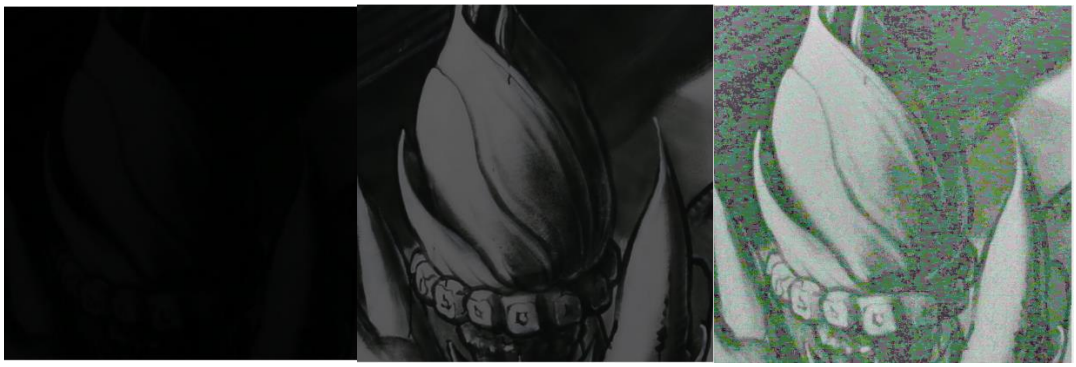


Figure 2.275: Histogram of 4.5 Experimental Result with max PSNR

In the case of the maximum experimental PSNR, the same image is obtained with the darkness level 4.0, again we can observe that it is characterized by color distortions. This is due to the reasons we mentioned above, but also to the fact that the images have now become very dark, making it difficult to recover the full visual information. This is also justified by the histogram of the original LL image (figure 2.273) where we see that the pixel values are accumulated at the left end of the histogram, with the dynamic range being too small for the method to recover information.

Darkness Level: 5.0

MIN PSNR



Original Low Light **Normal Light** **Experimental Result**

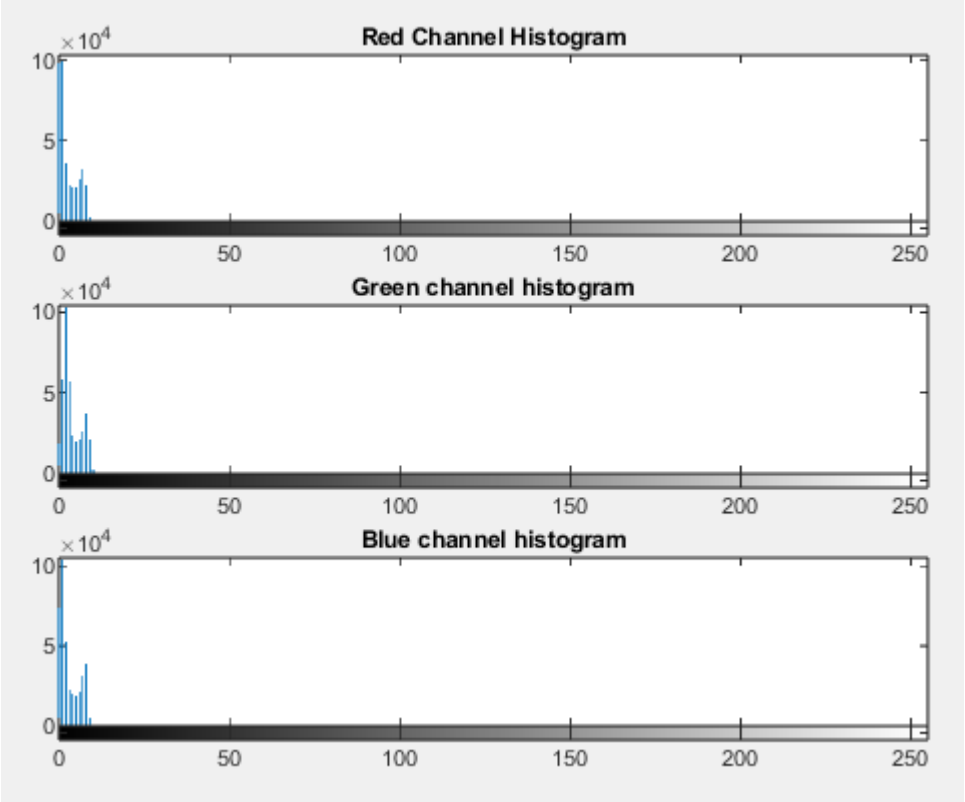


Figure 2.276: Histogram of LL image

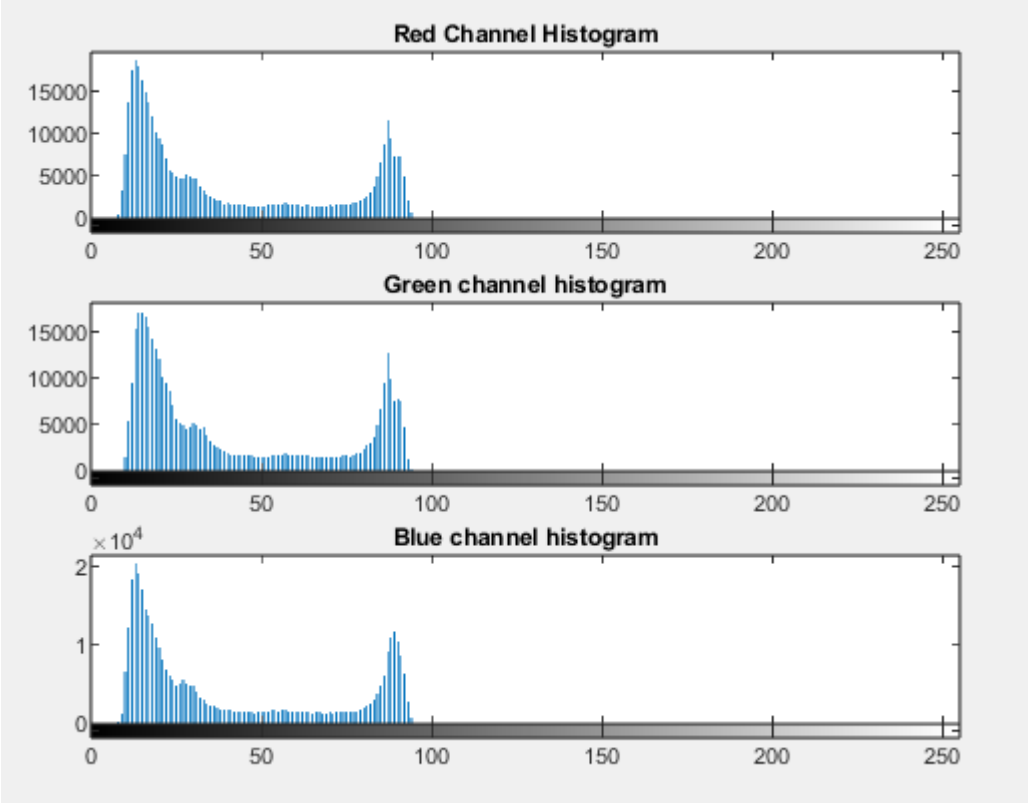


Figure 2.277: Histogram of NL image

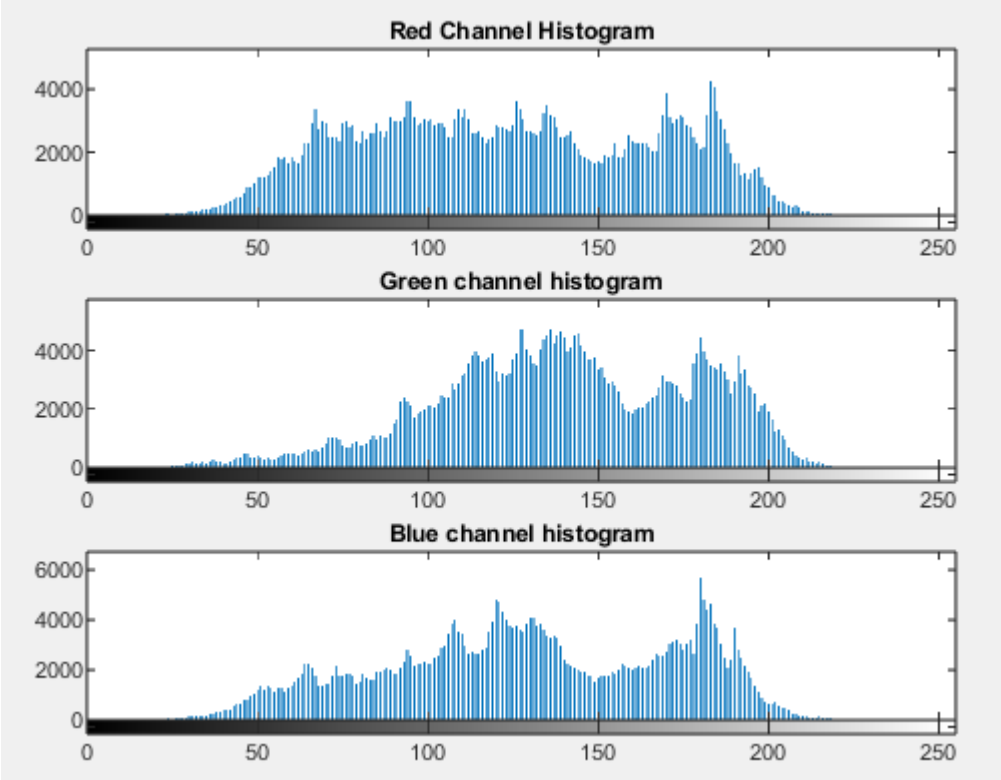


Figure 2.278: Histogram of 5.0 Experimental Result with min PSNR

At darkness level 5.0 in the case of the minimum experimental PSNR we again observe that the experimental result is characterized by strong color distortions, which is due to the fact that the dark image has been enhanced more than it should. This is also confirmed by the histograms, where in the ground truth histogram (figure 2.277) we see that the pixel values are distributed in gray values below 100, while in the experimental histogram (figure 2.278) they are distributed throughout the available range explaining the excessive amplification of the image.

MAX PSNR

Original Low Light **Normal Light** **Experimental Result**

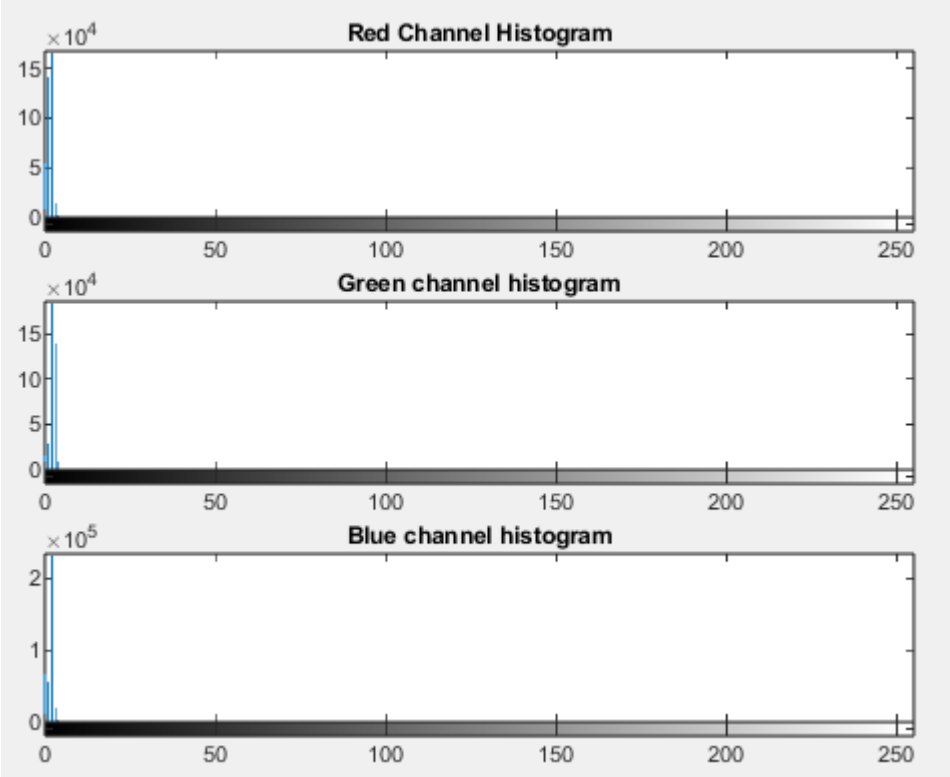


Figure 2.279: Histogram of LL image

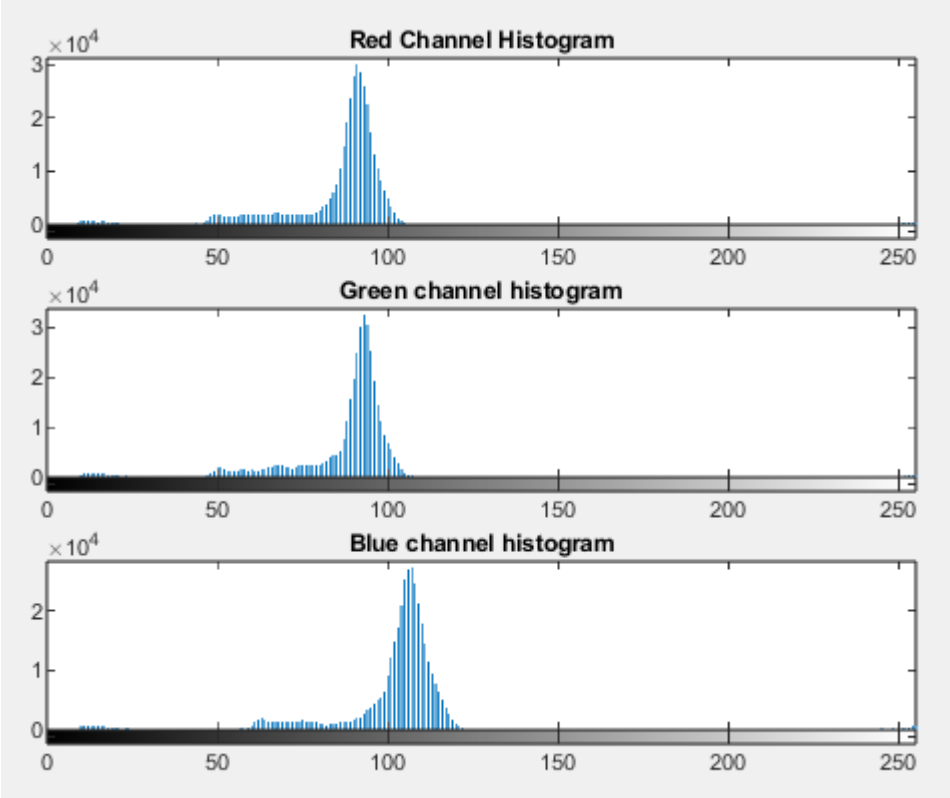


Figure 2.280: Histogram of NL image

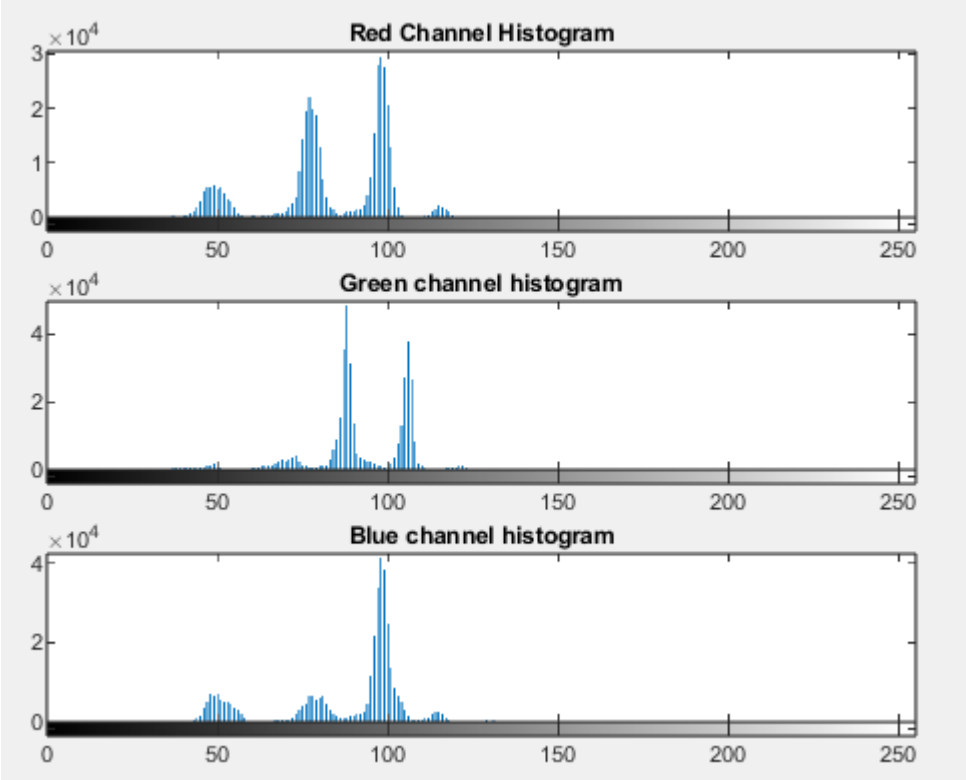


Figure 2.281: Histogram of 5.0 Experimental Result with max PSNR

In the case of the maximum experimental PSNR, the same image is obtained again and we notice here too that, while we have recovered the general texture of the image, intense color distortions are detected. This is due to 2 factors. First, because we are at the highest darkness level, the images have become too dark making it difficult to retrieve visual information. Second, as we mentioned above, we combine 3 different SSR parameters, and the parameter c=10 seems to negatively affect the result, since as we saw in the case of SSR, with this parameter we recover the general texture of the image but not the color information, something which is also found here.

At this point it is worth commenting on the MSR results in general. For the case of the minimum experimental PSNR we observed that at all darkness levels the image appears to be over-enhanced, leading to the introduction of additional noise and color distortions. This is something we predicted would happen from when we analyzed the values of the performance metrics, and it is confirmed by the histograms. In particular, it was found that this problem occurs in cases where in the ground truth histogram the pixel values are concentrated in gray values smaller than 120-150, while the experimental histogram has values that are distributed throughout the available range. This leads the experimental image to be characterized by

greater brightness and contrast, explaining the difference between ground truth and experiment.

In the case of the maximum experimental PSNR it was observed that, while the experimental histogram was very close to the ground truth, color distortions began to appear, which became more pronounced with the increase of the darkness level. The main reason for this is that we combine three different SSRs with different c values (10, 120 and 400), thus keeping not only the advantages but also the disadvantages of each constant. In the previous section we saw that for c=10 we recover the general texture of the image but not the color information, which is also found in MSR to some extent. So, c=10 introduces this problem into the MSR algorithm, leading to what we mentioned above. A solution to this problem could be either to only use large values of constant c or to reduce the weight of the constant c=10 so that it affects the final result less.

2.7 Defogging

In this section we will follow a completely different logic from what we have done so far. The basic idea we will use is that if we invert an RGB LL image, the result will visually resemble an NL image taken in a foggy environment [28]. An example from [28] is shown in the figure below.

Figure 2.282: Example of inverted and foggy images. LL images (up), inverted LL images (middle) and foggy images (down)

So, we understand that if we apply a defogging algorithm we can enhance the dark areas of each image. The logic we will follow is the following, first we will invert the respective LL image, and then we will apply the defogging algorithm. We will also invert the resulting image, leading to the enhanced image. The defogging algorithm we will use was developed by He et al [29], and is implemented in MATLAB. We used the ready-made MATLAB algorithm, as it will be much more optimized than any custom implementation we tried, which could affect the final result.

The function with which we implement the above logic is shown in figure B.2.25 of appendix B. This function should be applied to all the images in the data set, which is done with the scripts shown in figures B.2.26 and B.2.27. By implementing these scripts, we have at our disposal the experimental results, based on which we should evaluate the performance of the algorithm. For this purpose, we calculate the values of the performance metrics, which we mentioned in the previous chapter, by applying the script B.2.28. With the results obtained from this process we

construct summary tables with the minimum/maximum/average value of the metrics, per darkness level, as well as the corresponding line charts for PSNR and SSIM per darkness level, in order to evaluate how much it affects the performance of the algorithm. In addition, we will construct line charts of the average value of PSNR and SSIM for the original data and the experimental results, to see how much the quality of the original images improves.

Training Set

MSE					
Level	3.0	3.5	4.0	4.5	5.0
MIN	166,4880299	204,0602	249,1574	266,8422	419,855
MAX	13455,74764	11434,72	9091,67	10719,28	12630,2
AVERAGE	1831,89316	1595,682	1517,898	1778,449	2450,031
	PSNR(dB)				
Level	3.0	3.5	4.0	4.5	5.0
MIN	6,841725275	7,548547	8,544367	7,829147	7,1167
MAX	25,91697347	25,03322	24,16607	23,86826	21,89981
AVERAGE	17,01193611	17,55716	17,44126	16,52327	15,07257
	SSIM				
Level	3.0	3.5	4.0	4.5	5.0
MIN	0,102150389	0,066611	0,018291	0,002594	-0,03143
MAX	0,847232814	0,859476	0,850552	0,816003	0,759134
AVERAGE	0,565489767	0,541727	0,496939	0,426593	0,343103
	MV				
Level	3.0	3.5	4.0	4.5	5.0
MIN	38,30934784	29,74768	24,71322	20,23132	16,48683
MAX	217,123014	206,4308	198,5501	185,169	165,4016
AVERAGE	109,8345775	101,9563	91,80097	80,64505	71,54522
	STD				
Level	3.0	3.5	4.0	4.5	5.0
MIN	124,2078657	78,44636	56,16059	48,50621	60,01531
MAX	2478,126117	2445,413	2382,601	2299,339	1302,489
AVERAGE	683,841285	664,65	637,6705	591,4989	546,1223
	BRISQUE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	5,693960706	5,048779	6,94949	8,300922	10,11482
MAX	53,29122631	54,40798	55,38682	55,85574	55,08724
AVERAGE	30,43225908	31,71846	32,65922	34,73858	36,38779

	NIQE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	2,53731623	2,647062	2,97515	2,981025	3,18421
MAX	24,02631122	24,30622	16,72756	8,061458	10,26958
AVERAGE	3,938023694	4,220215	4,498758	4,856623	5,28723

Table 2.39: Dehaze results on training set

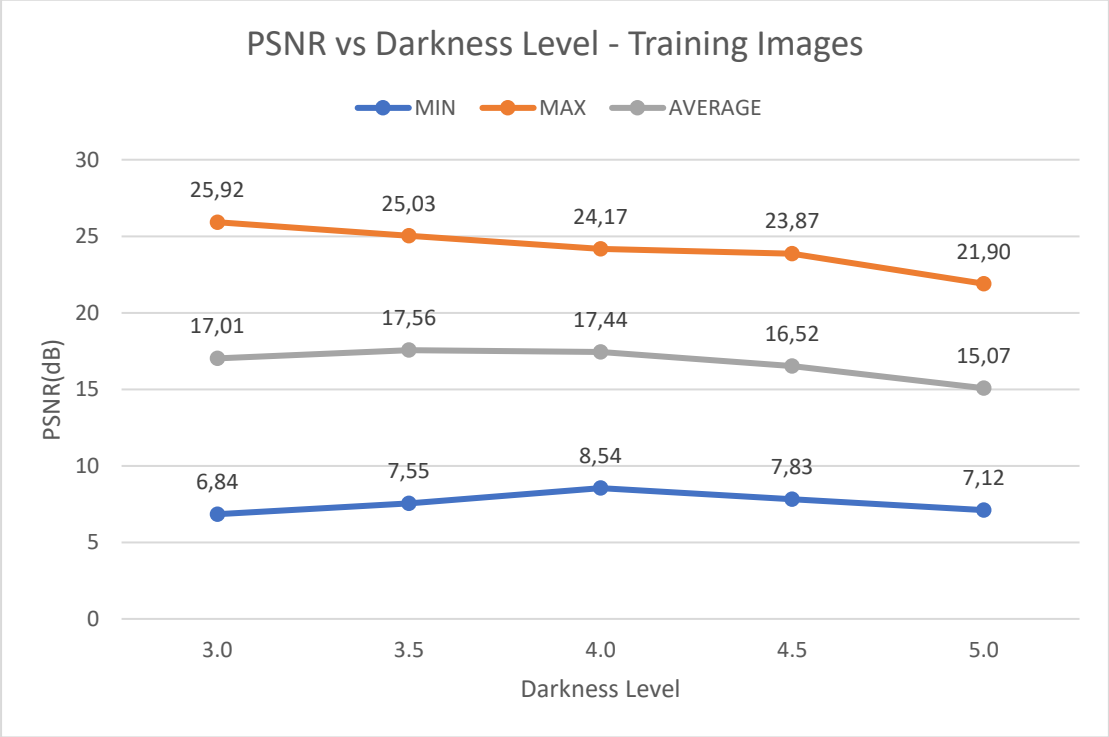


Figure 2.283: Experimental results PSNR vs Darkness level for training set

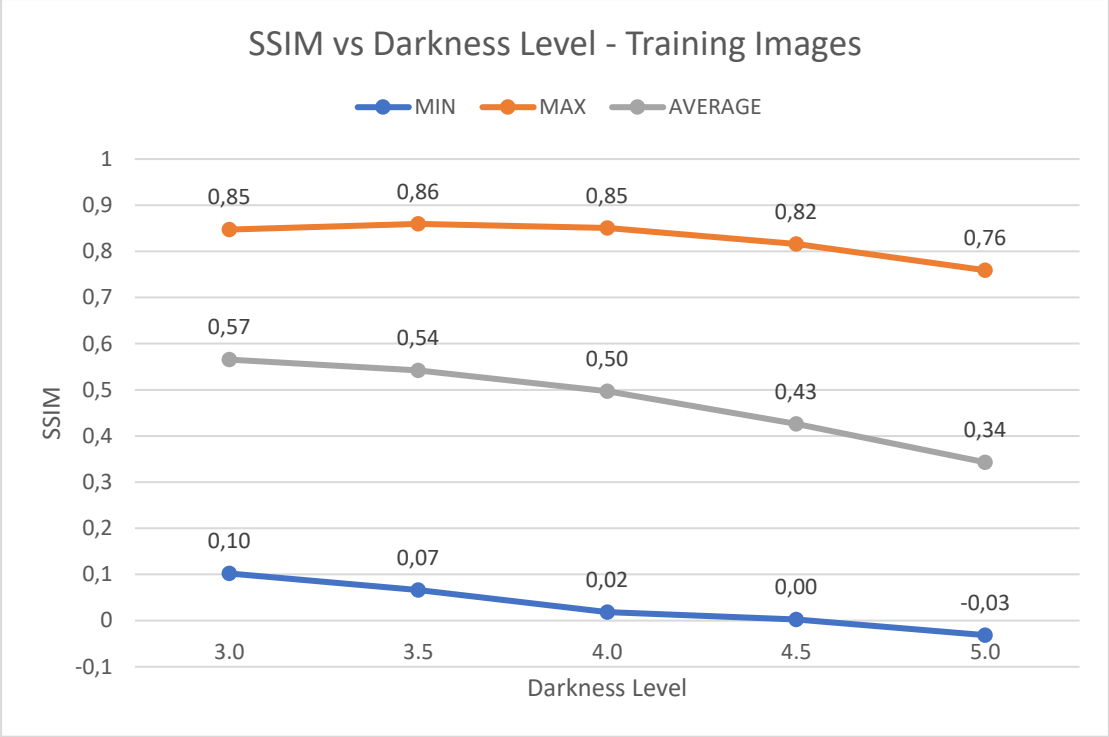


Figure 2.284: Experimental results SSIM vs Darkness level for training set

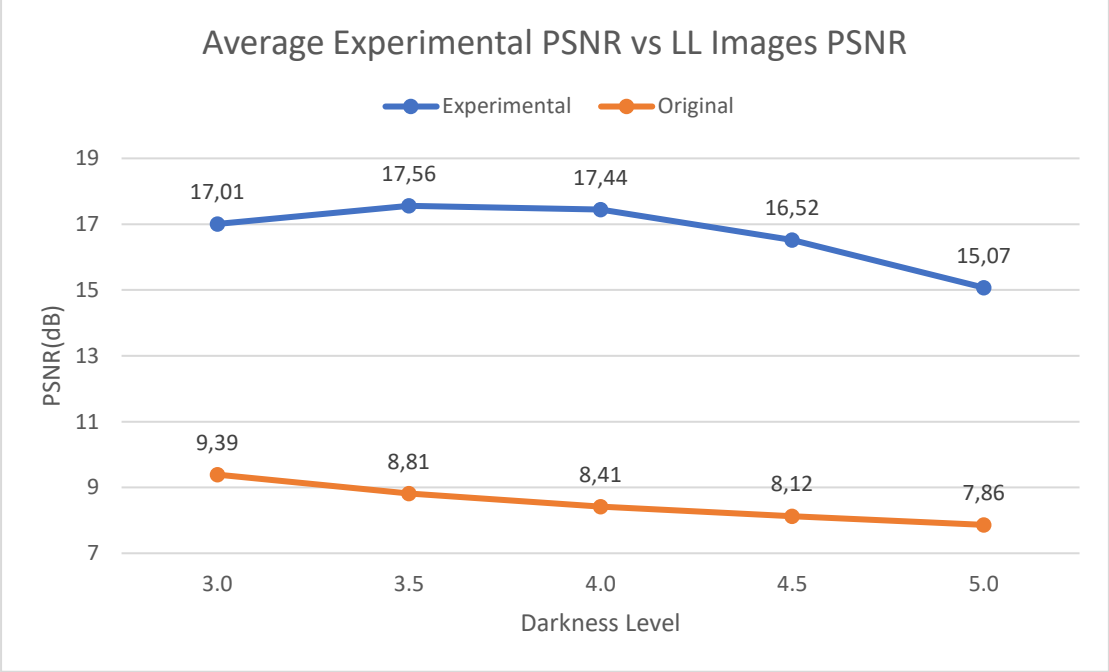


Figure 2.285: Experimental results vs LL Images PSNR

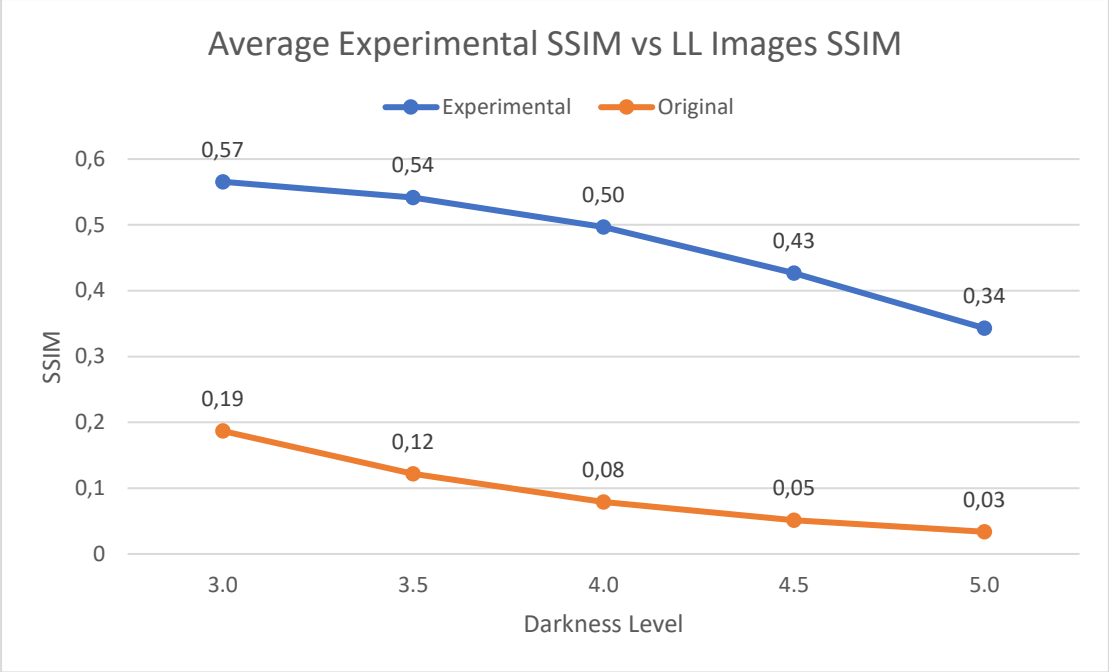


Figure 2.286: Experimental results vs LL Images SSIM

In the training set we notice that the values of the metrics improve noticeably. In addition, we can comment that the metrics follow the expected course with the increase of the darkness level. Specifically, MSE increases while PSNR and SSIM decrease with increasing darkness level. As for the quality metrics with no reference, MV and STD decrease with increasing darkness level, which means that the experimental result becomes darker, with pixel values distributed in a smaller range around the mean value. BRISQUE and NIQE increase with increasing darkness level, which means we are moving away from natural statistics. This behavior is due to the fact that with the increase of the darkness level the pixel values accumulate in the left part of the histogram, more and more strongly, with the result that the dynamic range is too small, and it becomes difficult to recover the full and correct visual information.

Validation Set

MSE					
Level	3.0	3.5	4.0	4.5	5.0
MIN	220,5636787	295,5619	349,3644	499,9832	664,5735
MAX	6770,723197	6757,127	5549,348	5070,605	6688,351
AVERAGE	1420,379441	1341,641	1425,283	1787,256	2580,837
	PSNR(dB)				
Level	3.0	3.5	4.0	4.5	5.0
MIN	9,824453017	9,833183	10,68838	11,08021	9,877613
MAX	24,69546364	23,42432	22,69802	21,14125	19,90537
AVERAGE	17,68237967	17,97748	17,57373	16,52677	14,90225
	SSIM				
Level	3.0	3.5	4.0	4.5	5.0
MIN	0,314569225	0,232451	0,233714	0,14826	0,042888
MAX	0,836163065	0,816592	0,783494	0,738112	0,681188
AVERAGE	0,588453739	0,568545	0,523373	0,451531	0,363531
	MV				
Level	3.0	3.5	4.0	4.5	5.0
MIN	29,06169173	22,84662	18,17203	14,50566	12,13161
MAX	178,6966605	180,1882	166,0237	162,3518	149,3092
AVERAGE	105,8838475	98,91091	89,84609	79,82501	69,71872
	STD				
Level	3.0	3.5	4.0	4.5	5.0
MIN	139,0443437	146,8853	159,1865	116,7613	105,9819
MAX	1227,630152	1263,146	1265,366	1182,203	1081,7
AVERAGE	674,6028981	651,7221	629,6494	579,8391	523,3451
	BRISQUE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	12,05318979	8,842169	11,23205	12,71975	14,15758
MAX	46,86835028	48,49695	48,38407	49,83659	51,71048
AVERAGE	31,73859324	31,66663	32,98724	34,83542	34,37112
	NIQE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	2,818848052	3,124162	3,07862	3,713084	3,952176
MAX	5,524429442	6,299291	7,878939	7,572838	8,866386
AVERAGE	3,842801574	4,080517	4,376685	4,727797	5,160134

Table 2.40: Dehaze results on validation set

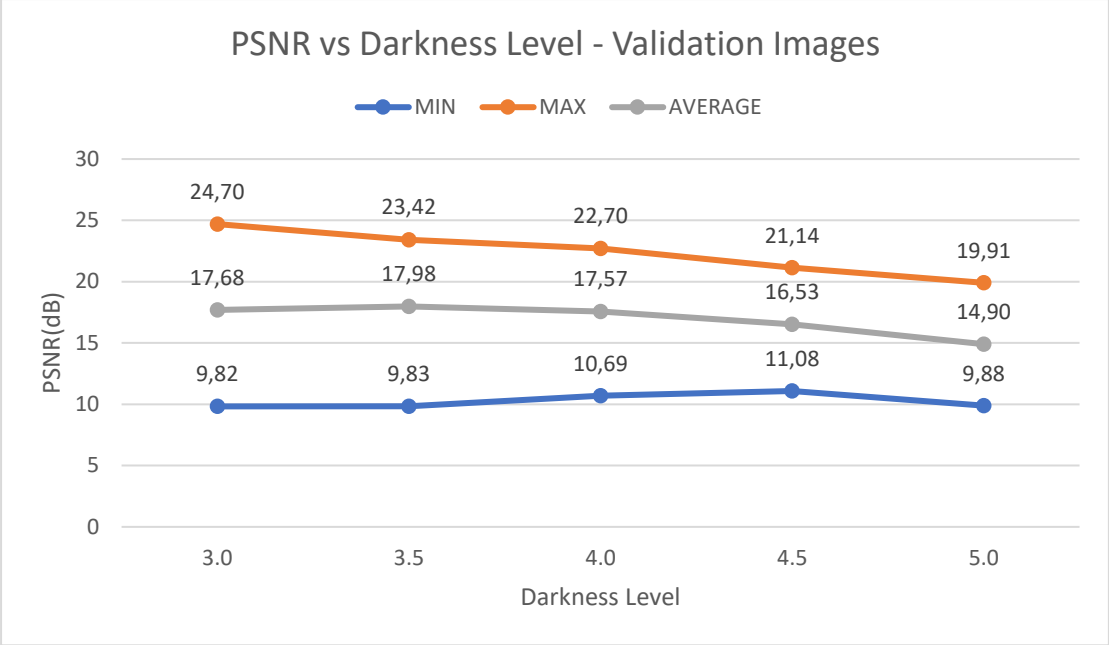


Figure 2.287: Experimental results PSNR vs Darkness level for validation set

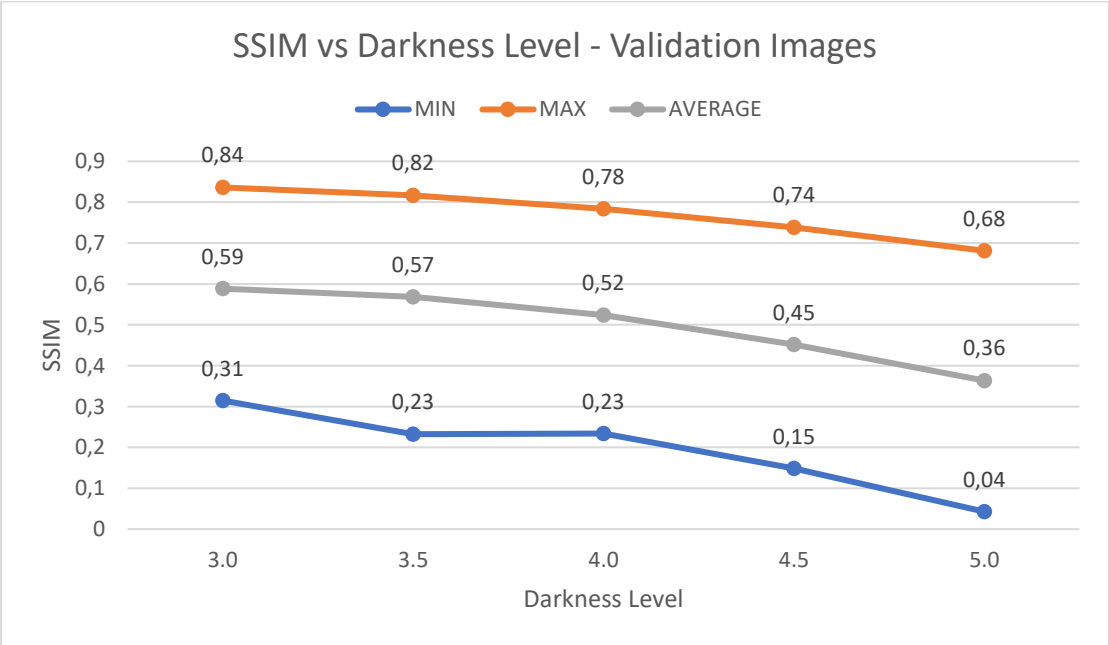


Figure 2.288: Experimental results SSIM vs Darkness level for validation set

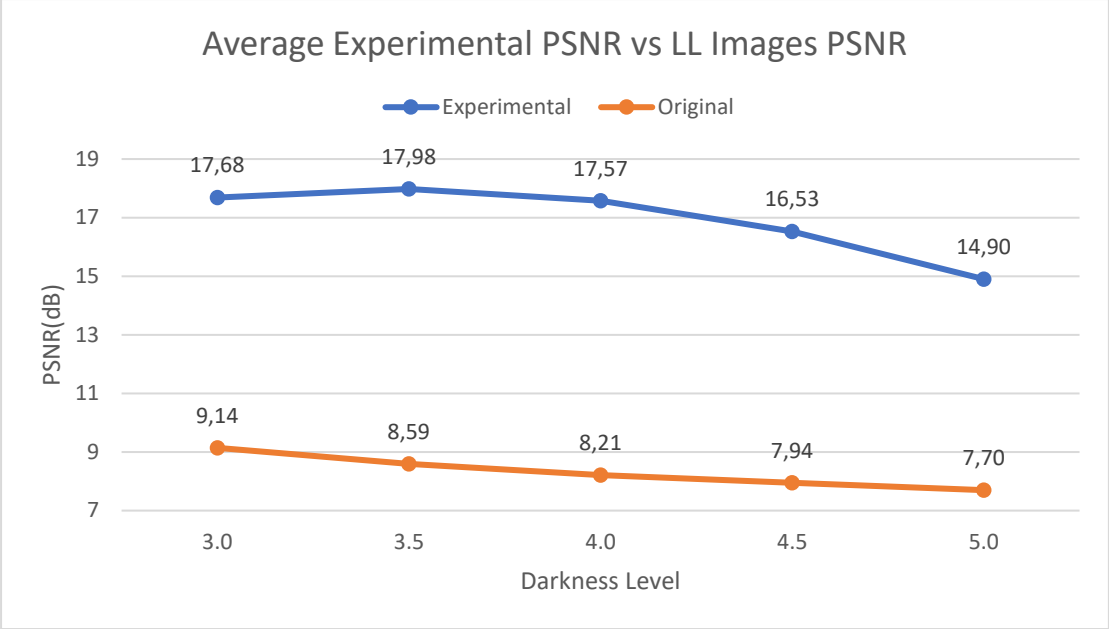


Figure 2.289: Experimental results vs LL Images PSNR

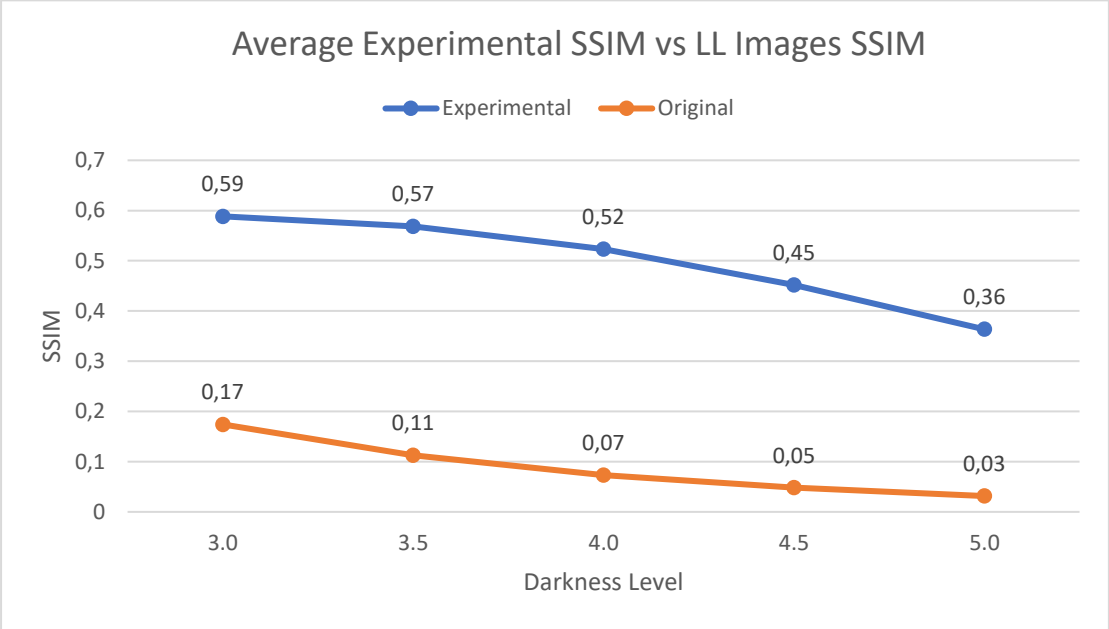


Figure 2.290: Experimental results vs LL Images SSIM

In the validation set the values of the metrics improve noticeably, and we observe the same behavior as the darkness level increases. The MSE increases and the PSNR and SSIM decrease with the increase of the darkness level, which means that the experimental result moves away from the ground truth case. MV and STD decrease with increasing darkness

level, meaning that images become darker with pixel values spread over a smaller range around the average brightness.

Test Set

MSE					
Level	3.0	3.5	4.0	4.5	5.0
MIN	233,5081993	300,7754	332,1464	439,0632	535,4297
MAX	10244,75109	8765,05	6329,959	7127,002	5362,924
AVERAGE	1770,137098	1653,327	1655,357	1972,755	2646,74
	PSNR(dB)				
Level	3.0	3.5	4.0	4.5	5.0
MIN	8,025789496	8,70326	10,11679	9,601735	10,83679
MAX	24,44778226	23,34838	22,91751	21,70553	20,84378
AVERAGE	17,18743152	17,47945	17,13891	16,00111	14,42391
	SSIM				
Level	3.0	3.5	4.0	4.5	5.0
MIN	0,18713404	0,129753	0,124088	0,065689	-0,01252
MAX	0,870310321	0,87548	0,850346	0,770045	0,58176
AVERAGE	0,594287549	0,561553	0,5086	0,438122	0,337125
	MV				
Level	3.0	3.5	4.0	4.5	5.0
MIN	32,55782741	25,02718	14,00675	13,26788	10,89123
MAX	170,8152124	173,0631	161,0322	152,8835	145,1221
AVERAGE	106,3222952	99,77421	90,64965	78,97358	68,62438
	STD				
Level	3.0	3.5	4.0	4.5	5.0
MIN	168,8599085	92,12444	76,77816	72,33594	88,56258
MAX	1731,38169	1678,286	1570,726	1355,347	1187,504
AVERAGE	686,6604479	662,5162	627,4498	569,5787	491,4949
	BRISQUE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	5,022448887	6,432561	10,37047	15,05212	15,02235
MAX	55,9694651	56,59765	52,56129	52,69716	57,20963
AVERAGE	30,22222332	31,29291	32,54937	34,7753	37,39497
	NIQE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	2,747940028	2,839296	2,944278	3,360973	3,34478
MAX	6,655770079	7,049703	7,965978	9,226072	10,22829
AVERAGE	3,858973321	4,133813	4,486066	4,90713	5,256686

Table 2.41: Dehaze results on test set

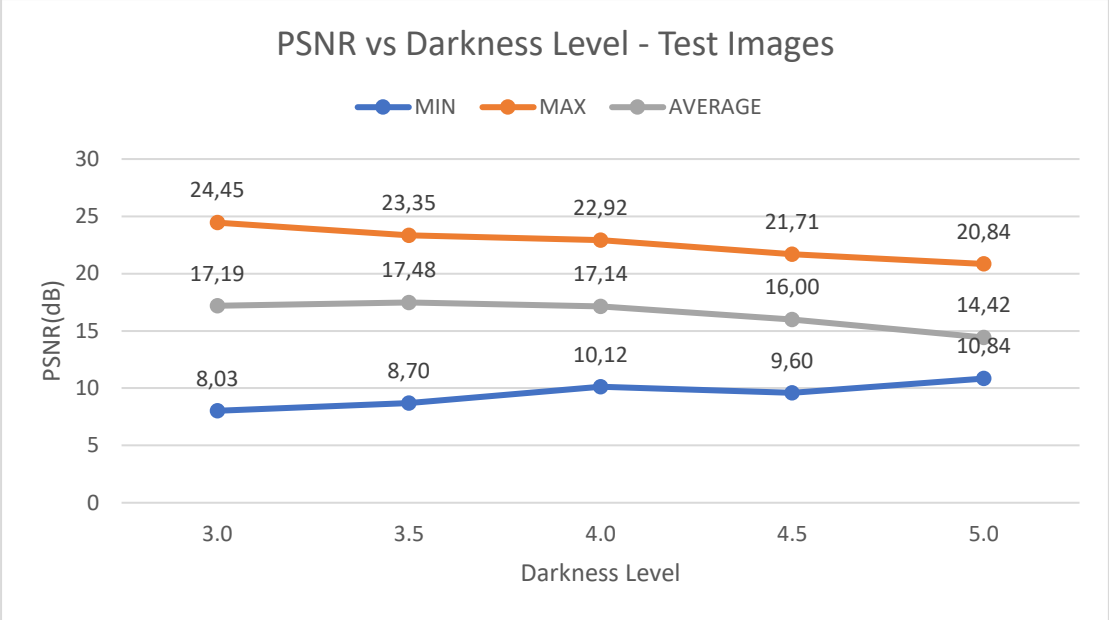


Figure 2.291: Experimental results PSNR vs Darkness level for test set

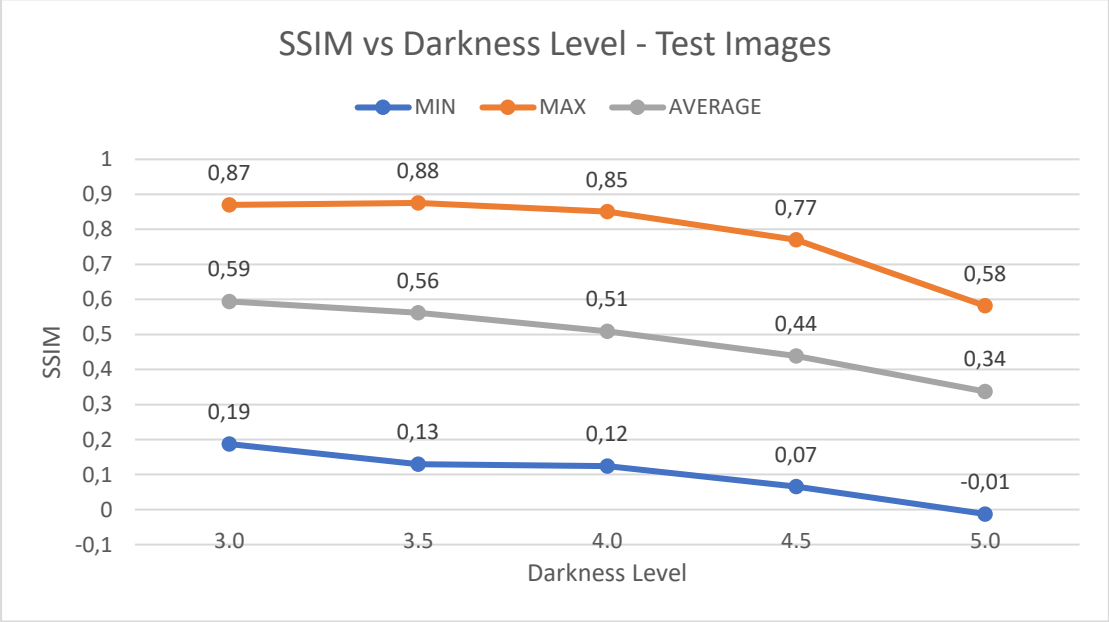


Figure 2.292: Experimental results SSIM vs Darkness level for test set

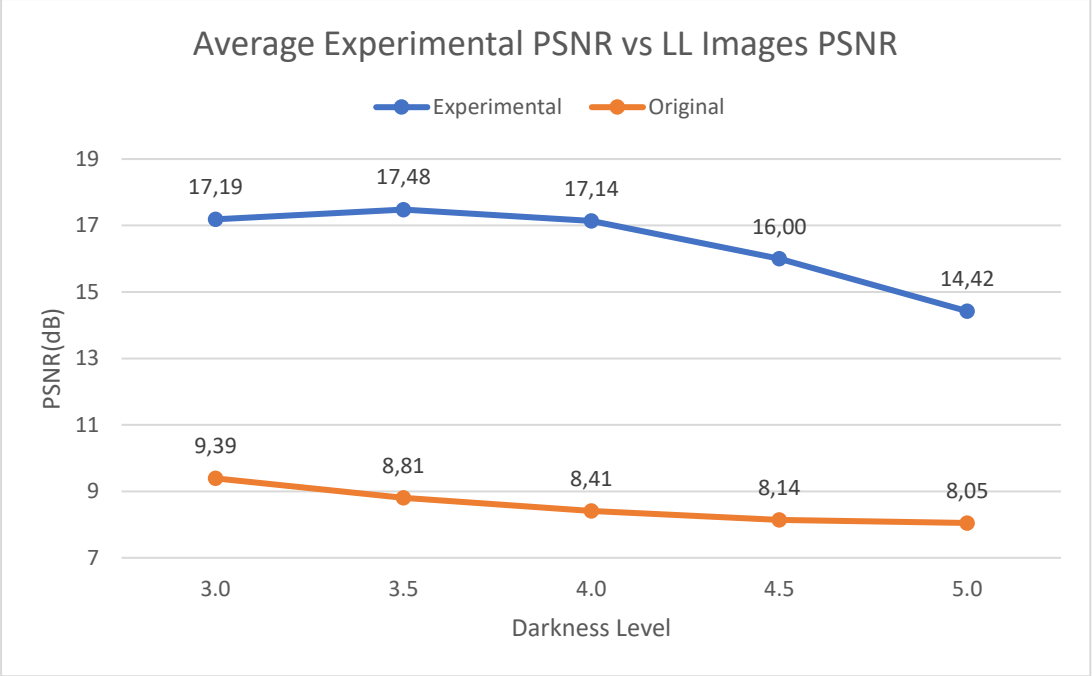


Figure 2.293: Experimental results vs LL Images PSNR

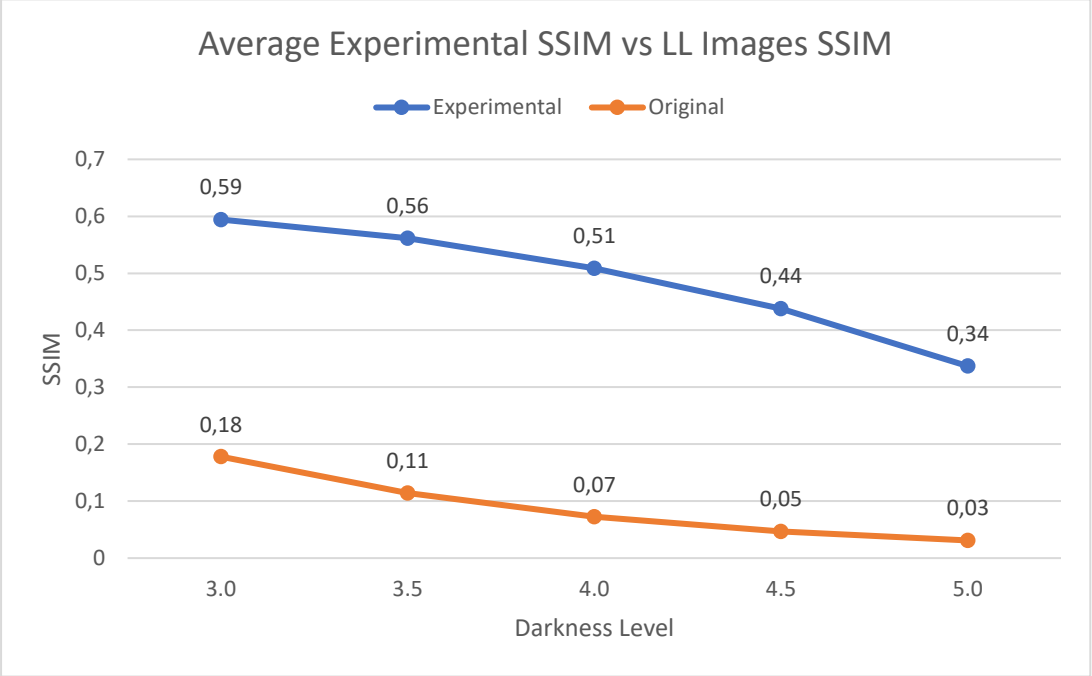


Figure 2.294: Experimental results vs LL Images SSIM

In the test set we observe the same behavior, with the values of the metrics improving and following the expected course with the increase of the darkness level.

Commenting on the results as a whole, we see that the metric values improve noticeably, with PSNR increasing by an average of 8.2dB and SSIM by 0.38. In addition, for this method we observe that the expected course is followed with the increase of the darkness level, as from the values of the metrics we see that the performance of the algorithm decreases with the increase of the darkness level. This is because by increasing the darkness level, the images become very dark, which means that the inverted images will be characterized by large percentages of foggyness, resulting in the defogging algorithm not being able to remove it completely. This will lead to the experimental result not being fully amplified and introducing extra noise.

To visually see the results of the algorithm, but also to confirm what we have mentioned so far, we will display for each darkness level the images that correspond to the minimum and maximum experimental PSNR. In addition, we will display the corresponding LL and ground truth images, as well as their histograms.

Darkness Level: 3.0

MIN PSNR

Original Low Light **Normal Light** **Experimental Result**

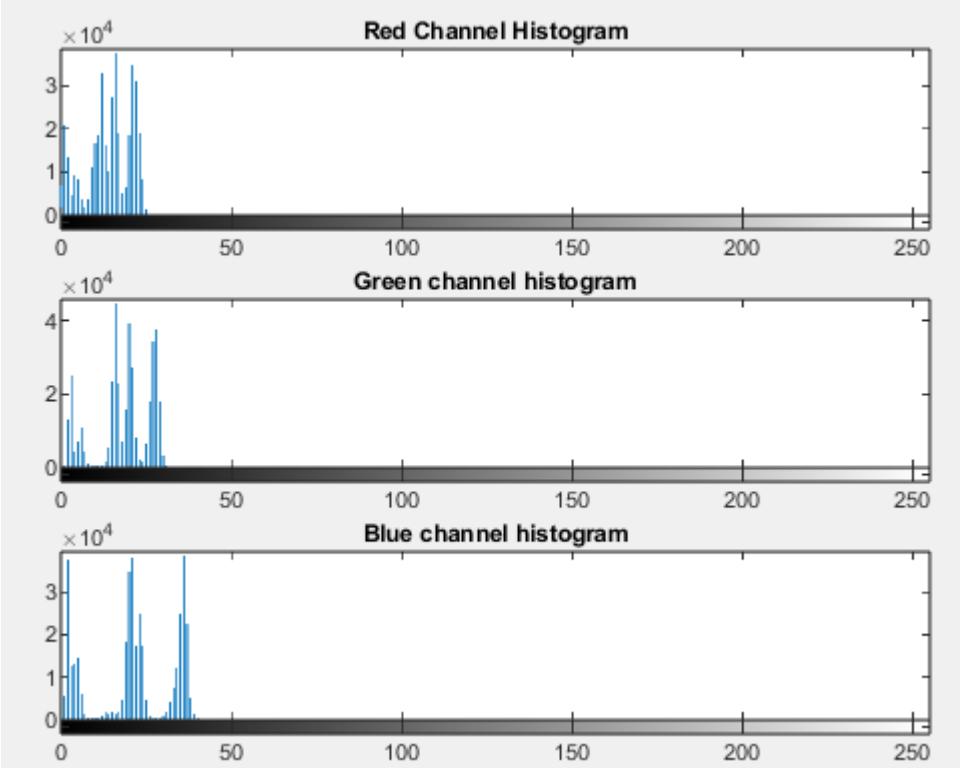


Figure 2.295: Histogram of LL image

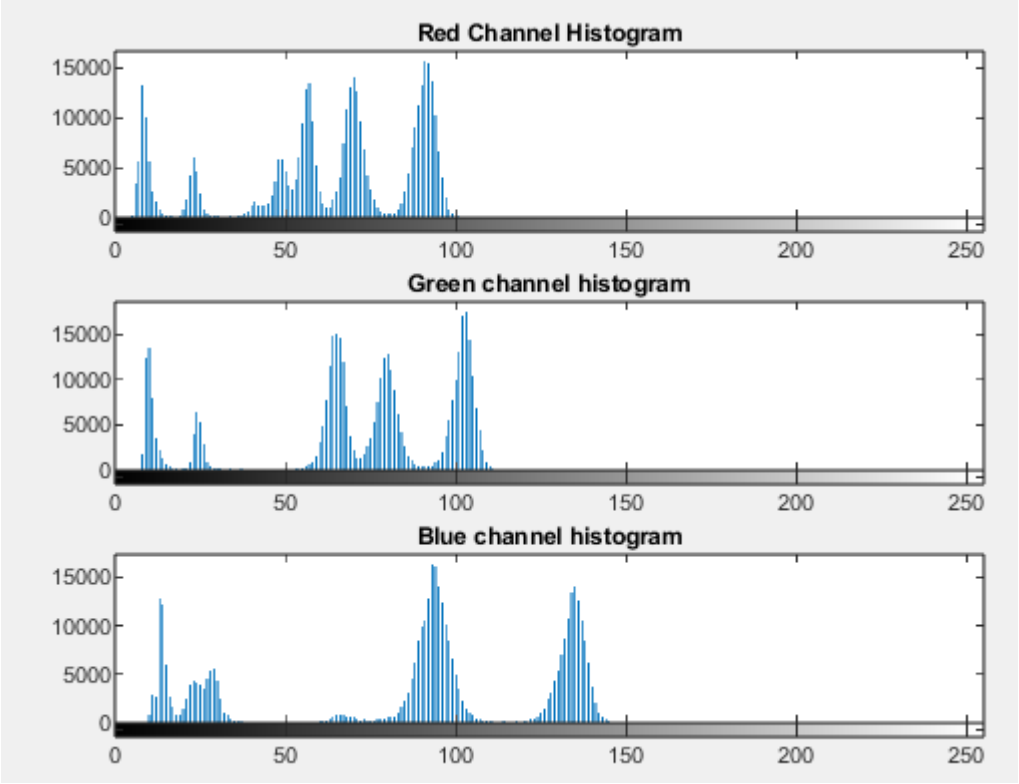


Figure 2.296: Histogram of NL image

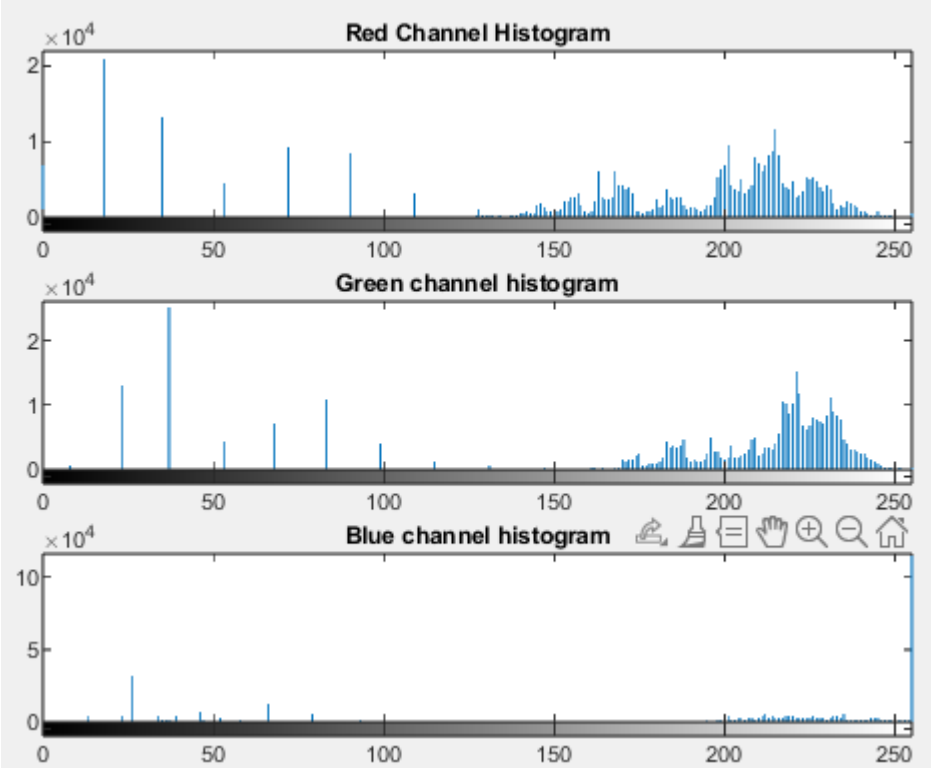


Figure 2.297: Histogram of 3.0 Experimental Result with min PSNR

For darkness level 3.0 in the case of the minimum experimental PSNR we see that the image is overenhanced, with the experimental result being brighter than the ground truth case. Nevertheless, the color information is fully recovered and no color distortions or extra noise appear.

MAX PSNR

Original Low Light **Normal Light** **Experimental Result**

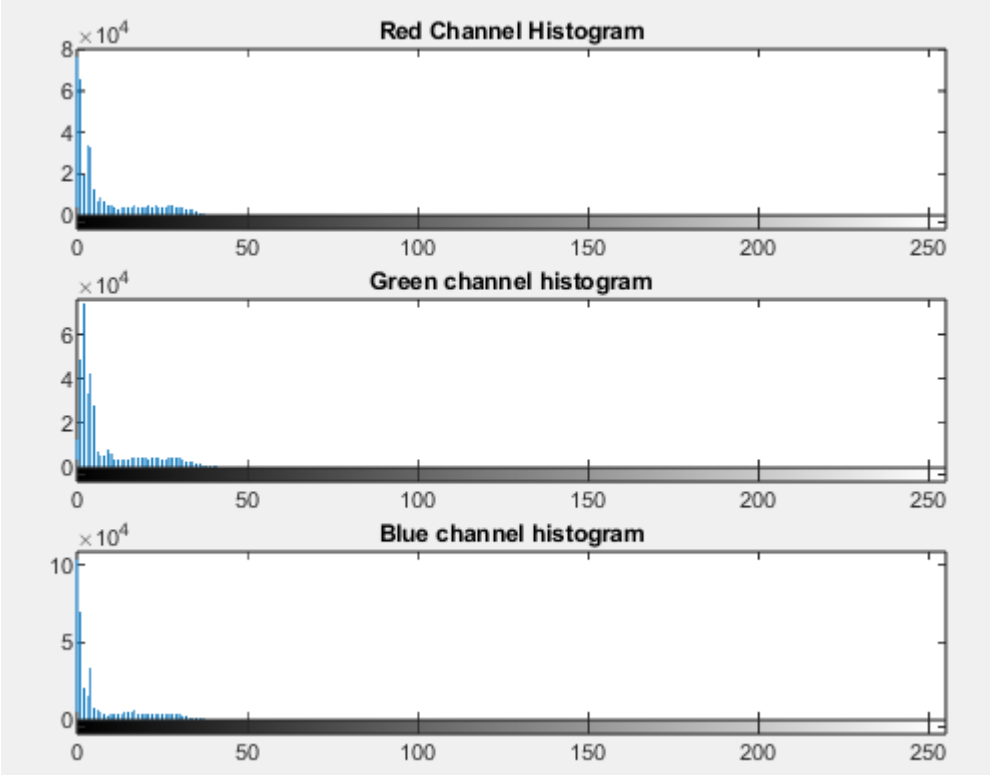


Figure 2.298: Histogram of LL image

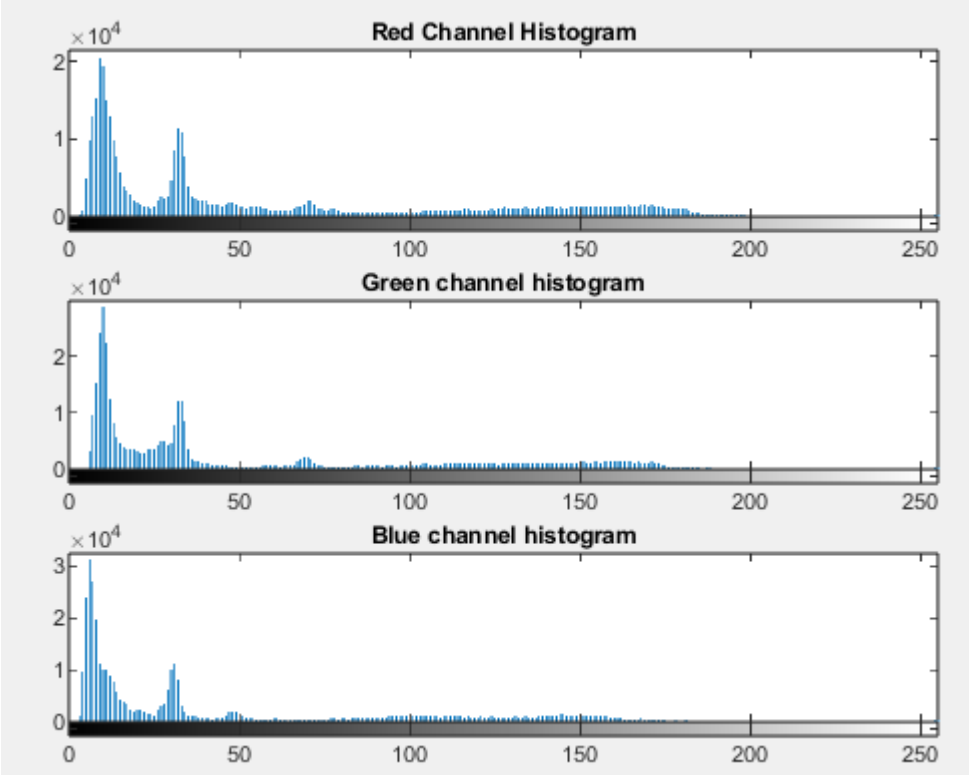


Figure 2.299: Histogram of NL image

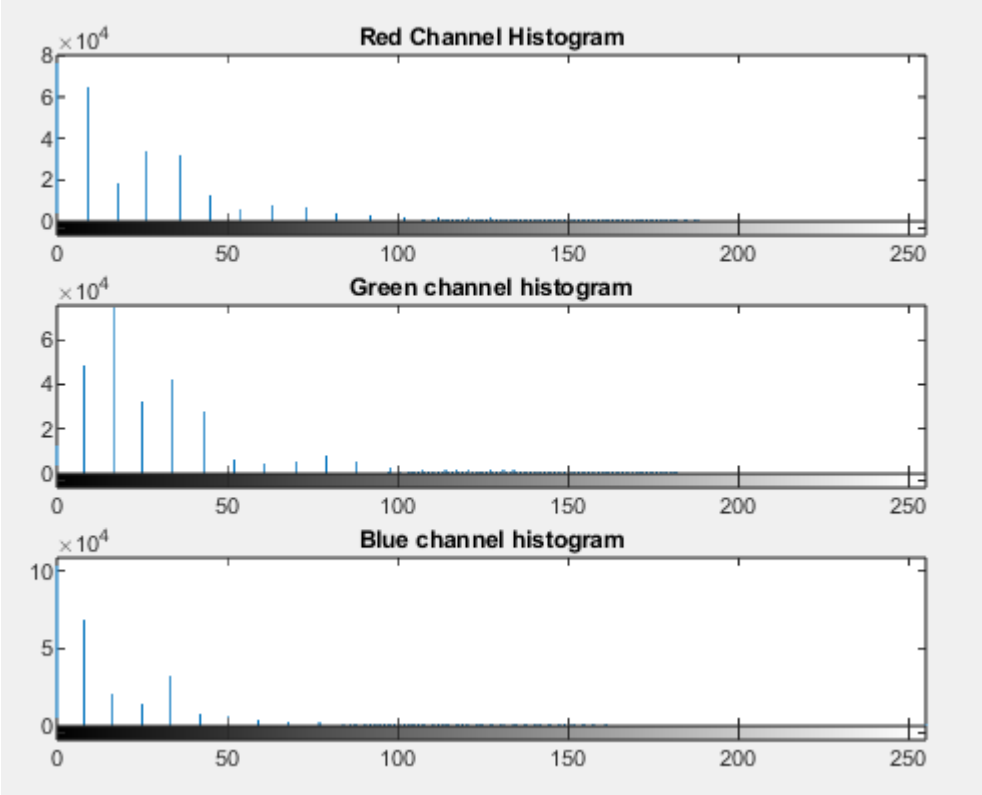


Figure 2.300: Histogram of 3.0 Experimental Result with max PSNR

In the case of the maximum experimental PSNR we see that the experimental result is visually very close to the ground truth case. Image texture as well as color information are fully recovered, justifying the maximum value of PSNR. This is also confirmed by the experimental histogram (figure 2.300) whose form is similar to the ground truth case (figure 2.299).

Darkness Level: 3.5

MIN PSNR

Original Low Light **Normal Light** **Experimental Result**

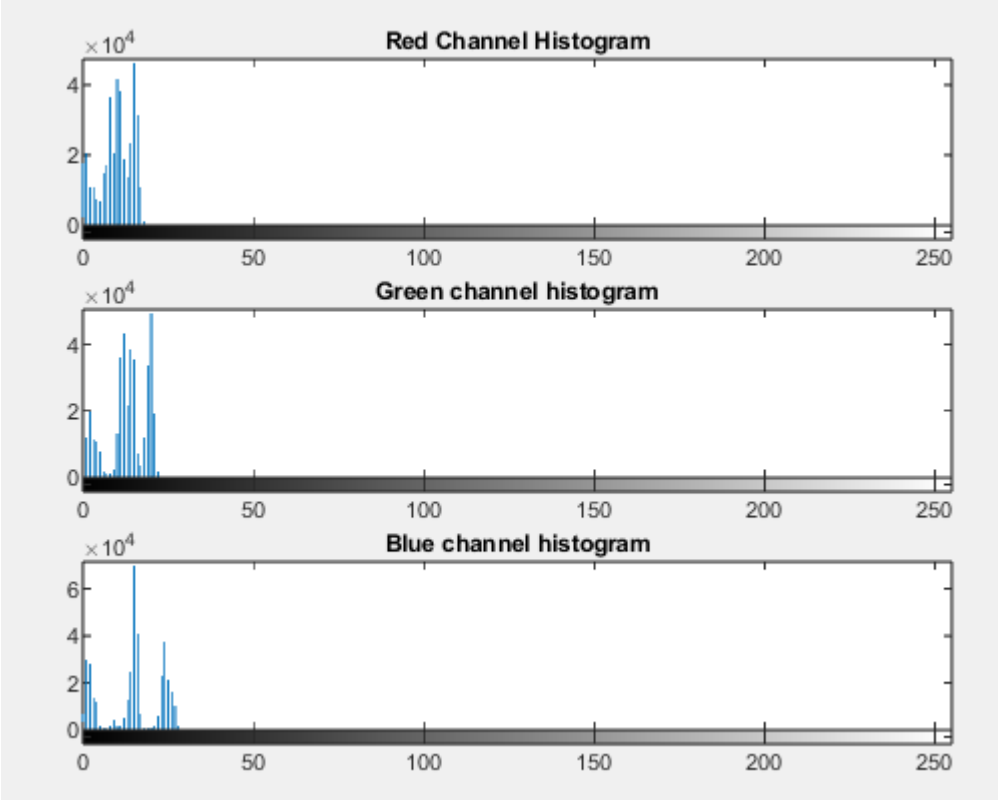


Figure 2.301: Histogram of LL image

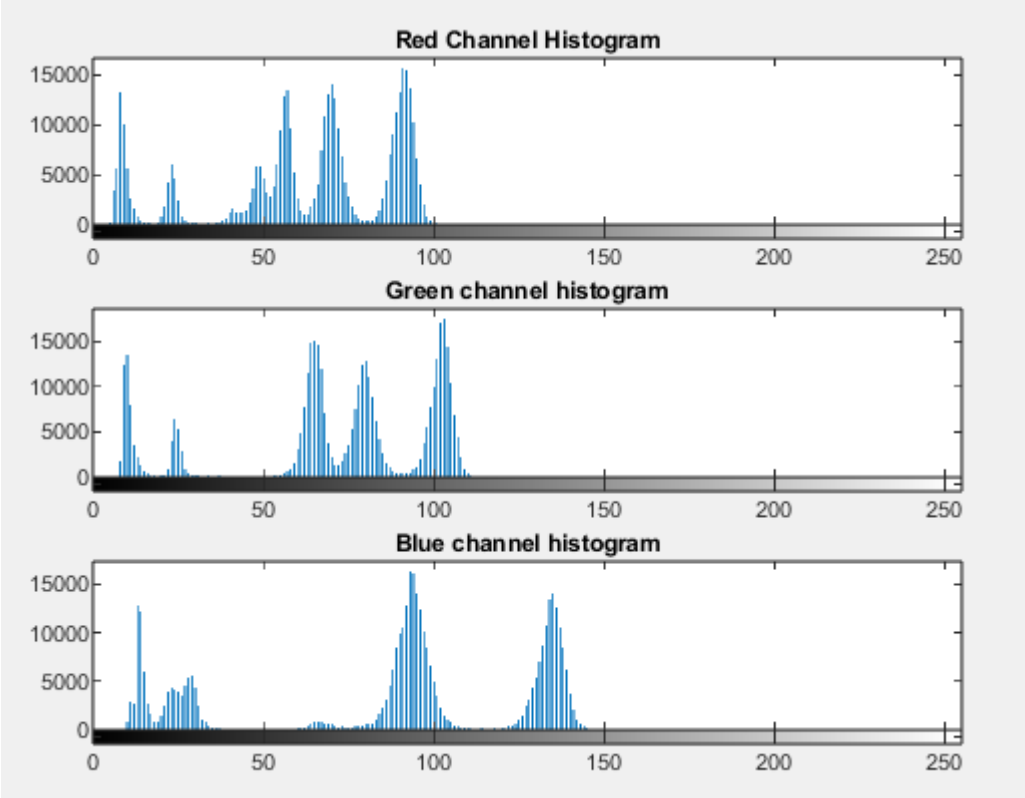


Figure 2.302: Histogram of NL image

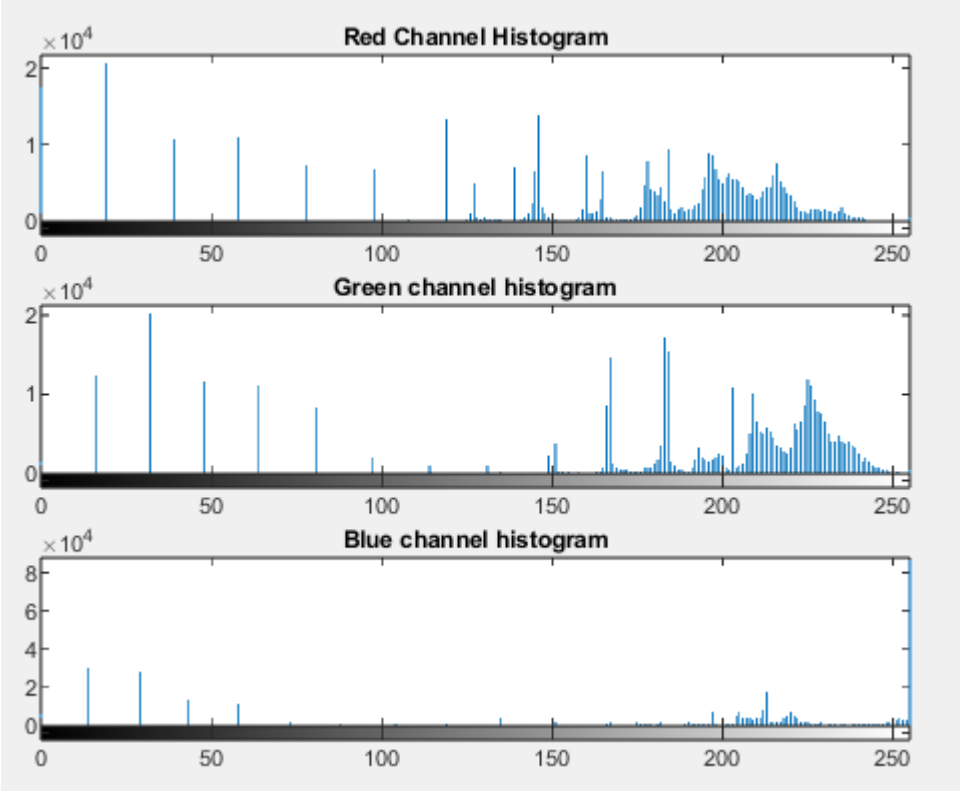


Figure 2.303: Histogram of 3.5 Experimental Result with min PSNR

For darkness level 3.5 in the case of the minimum experimental PSNR the same image is obtained as for darkness level 3.0, again we notice that it has been enhanced more than it should be. Here too, much of the visual information has been recovered, such as colors and texture. We must notice, however, that color distortions begin to appear, which is due to the fact that we are at a greater darkness level than before, as a result of which it is more difficult to retrieve the color information.

MAX PSNR

Original Low Light **Normal Light** **Experimental Result**

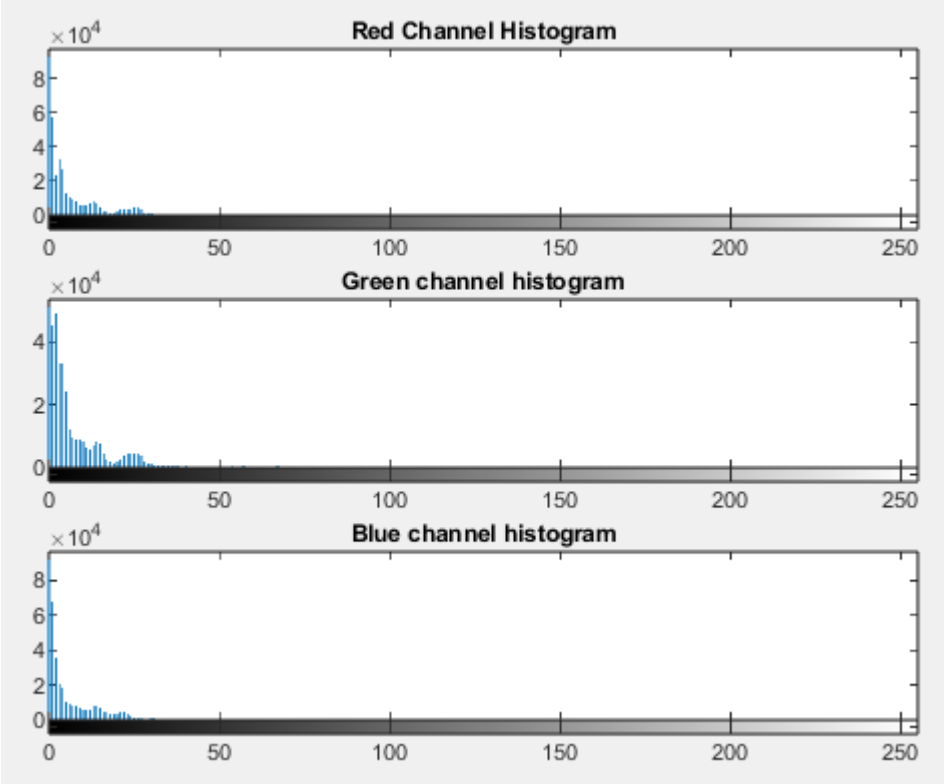


Figure 2.304: Histogram of LL image

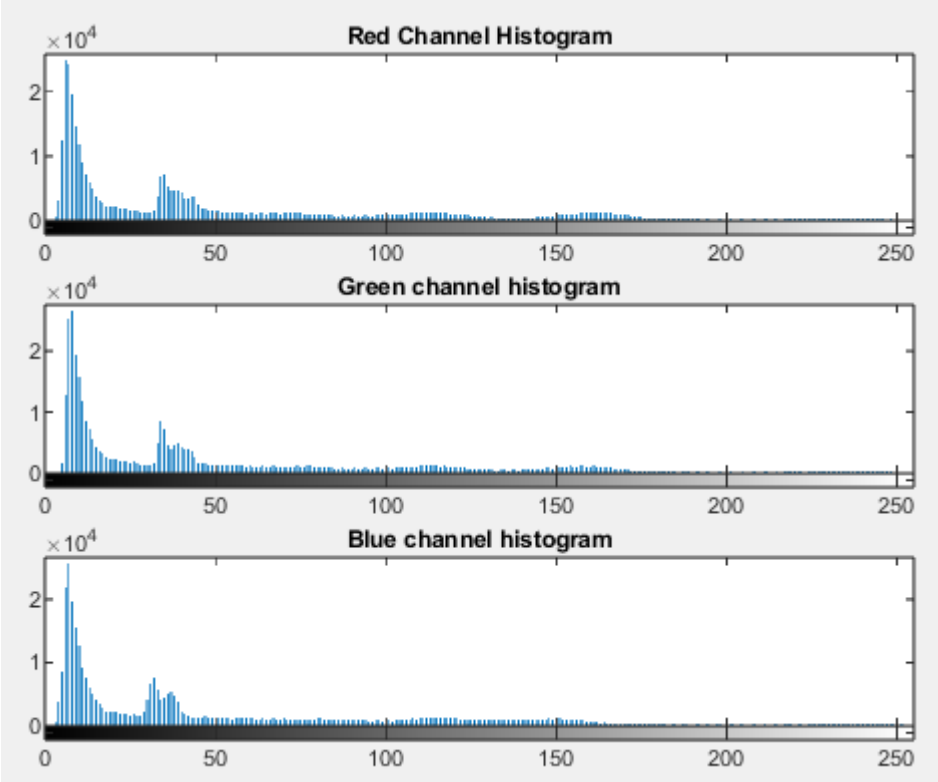


Figure 2.305: Histogram of NL image

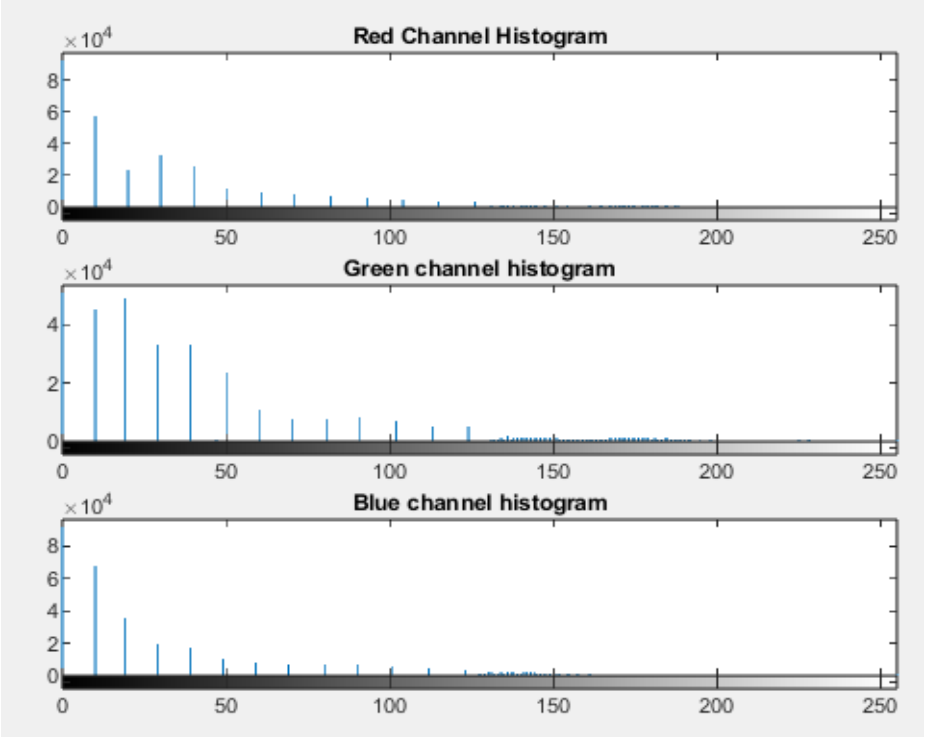


Figure 2.306: Histogram of 3.5 Experimental Result with max PSNR

In the case of the maximum experimental PSNR we observe that the experimental result is visually very close to the ground truth case, which is also confirmed by the experimental histogram (figure 2.306) whose form is very close to the ground truth case (figure 2.305). The only thing we have to comment on is that, as can be seen from image 2.306, the pixel values of the experimental results are distributed in specific gray value levels, something we had also observed in the histogram equalization.

Darkness Level: 4.0

MIN PSNR

Original Low Light **Normal Light** **Experimental Result**

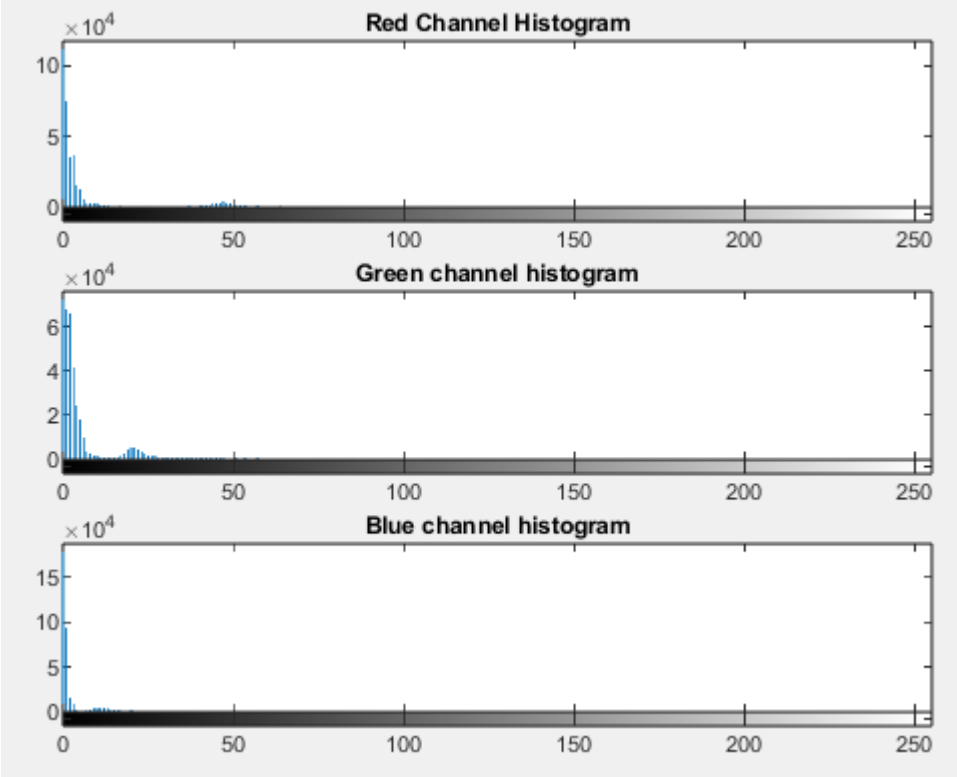


Figure 2.307: Histogram of LL image

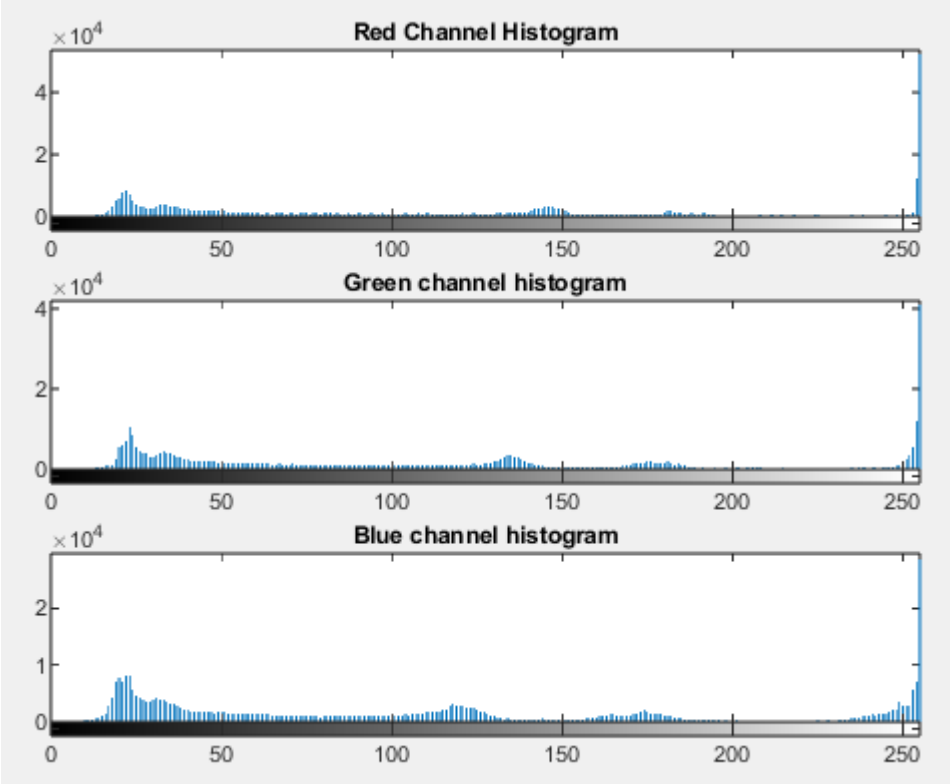


Figure 2.308: Histogram of NL image

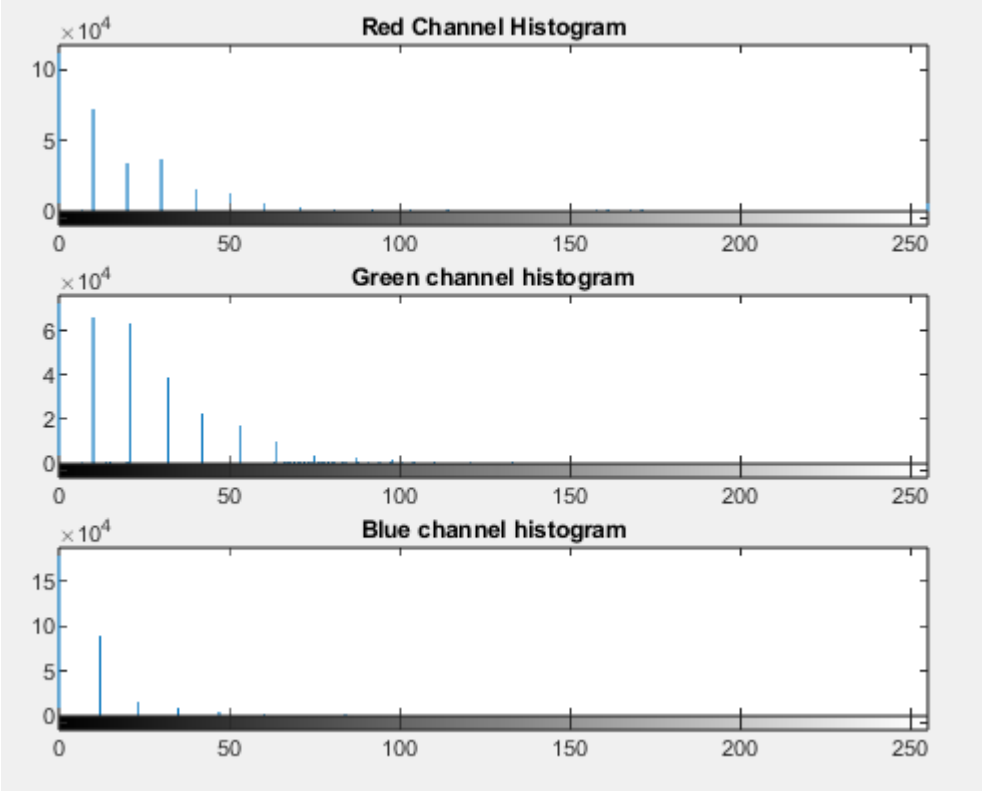


Figure 2.309: Histogram of 4.0 Experimental Result with min PSNR

For darkness level 4.0 in the case of the minimum experimental PSNR we see that the experimental result does not resemble the ground truth case, as the background of the image has been enhanced and the vehicle has remained dark, explaining the minimum value of the PSNR. This is also confirmed by the histograms, where in the experimental histogram (figure 2.309) we see that the pixel values remain concentrated in the left part of the histogram, while in the ground truth histogram (figure 2.310) the values are distributed throughout the available range, with the more in the right part of the histogram.

MAX PSNR

Original Low Light **Normal Light** **Experimental Result**

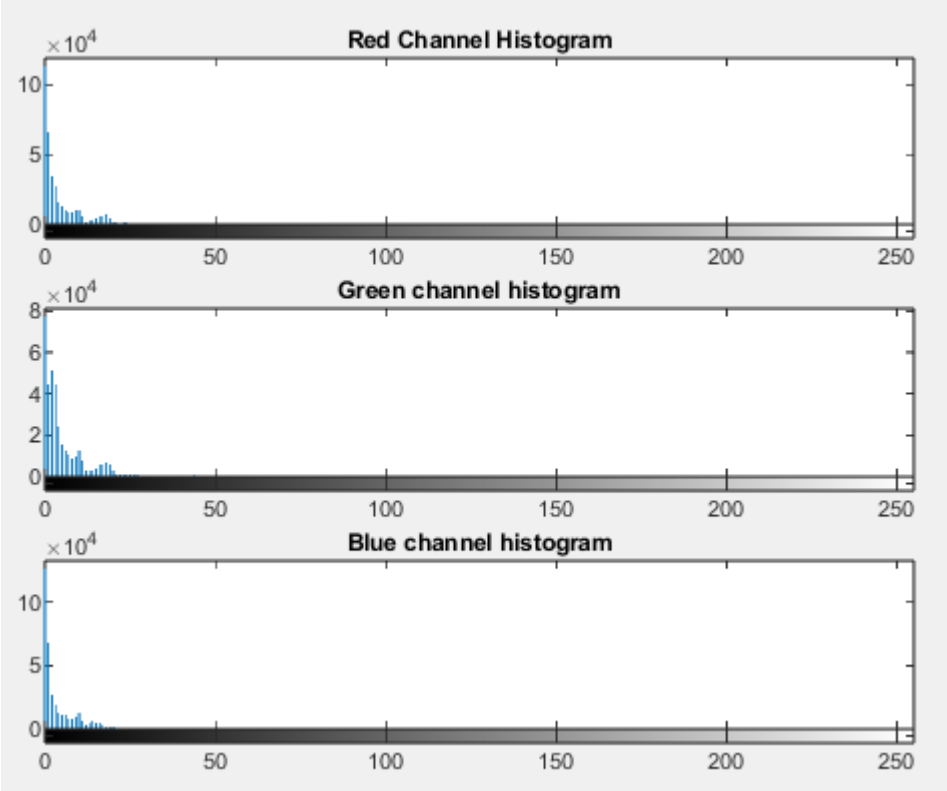


Figure 2.310: Histogram of LL image

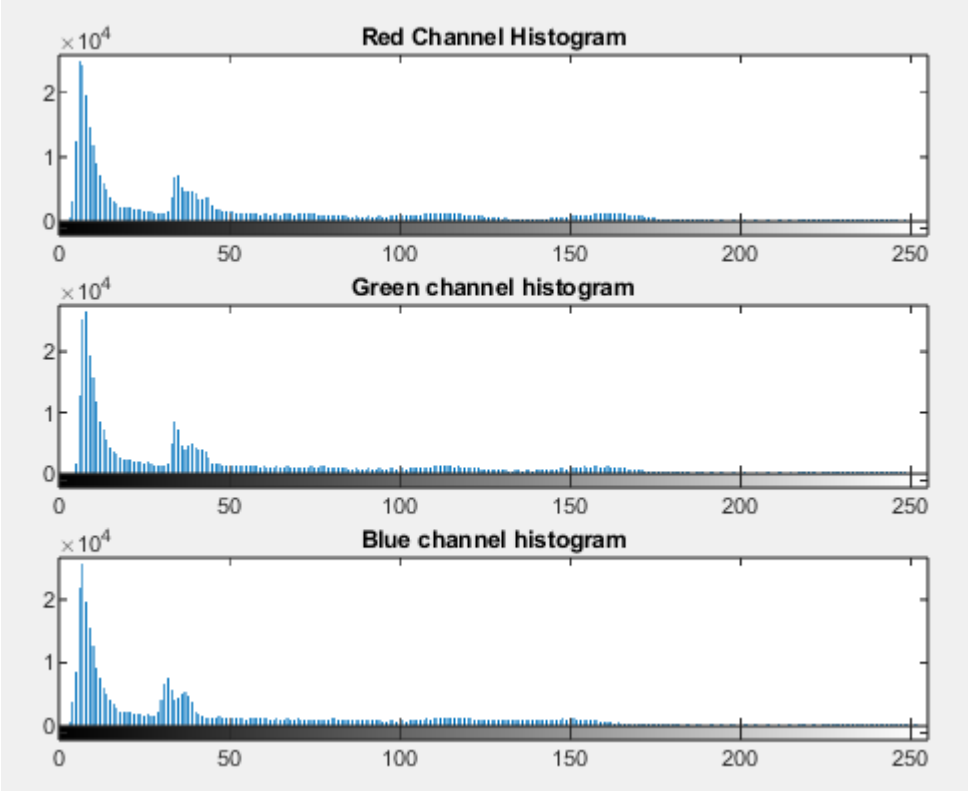


Figure 2.311: Histogram of NL image

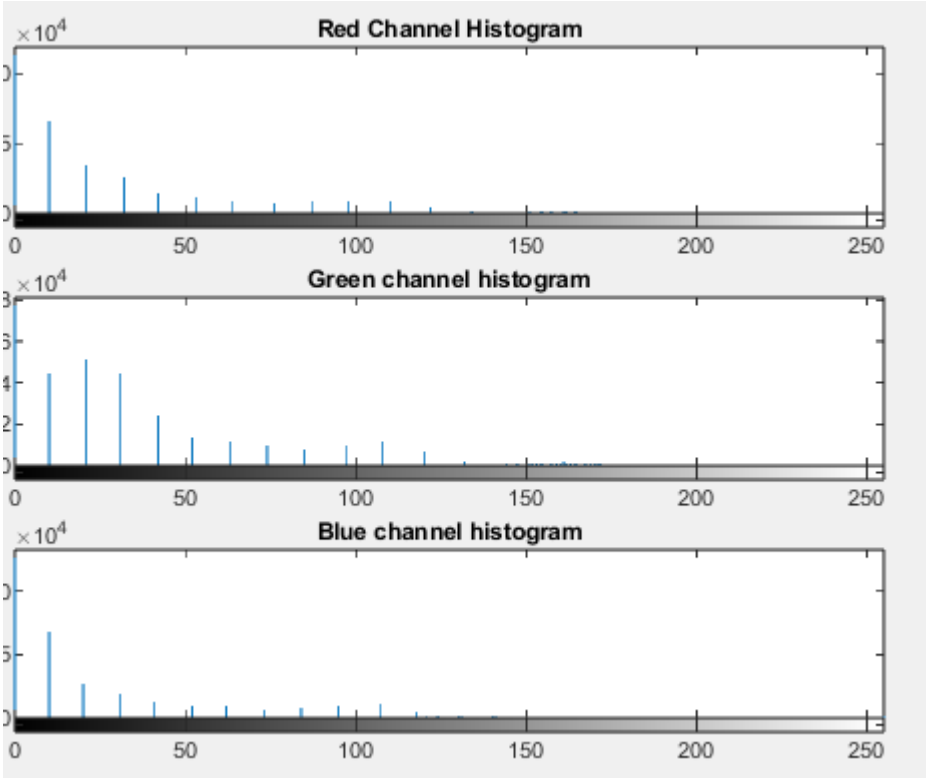
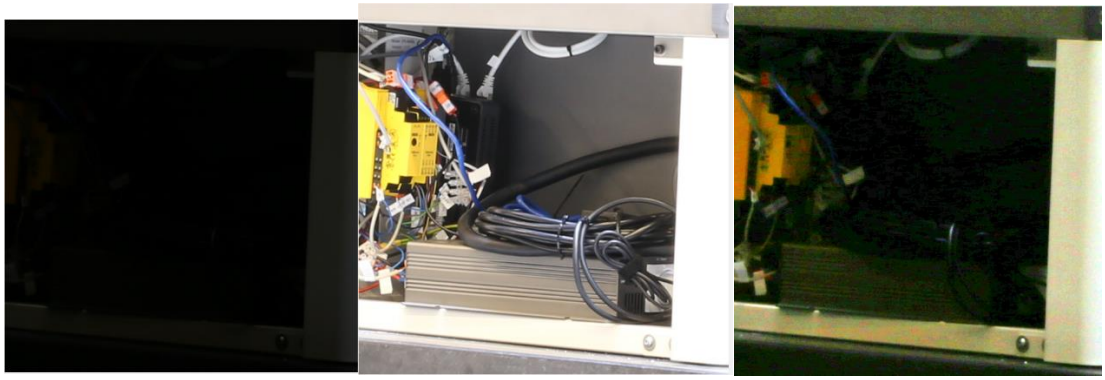


Figure 2.312: Histogram of 4.0 Experimental Result with max PSNR

In the case of the maximum experimental PSNR we see that the experimental result is visually very close to the ground truth case, explaining the maximum value that the metric takes. This is also confirmed by the experimental histogram (figure 2.312) where we see that its form is very close to the ground truth case, with the only difference being that the pixel values are distributed in specific gray value levels.

Darkness Level: 4.5

MIN PSNR



Original Low Light **Normal Light** **Experimental Result**

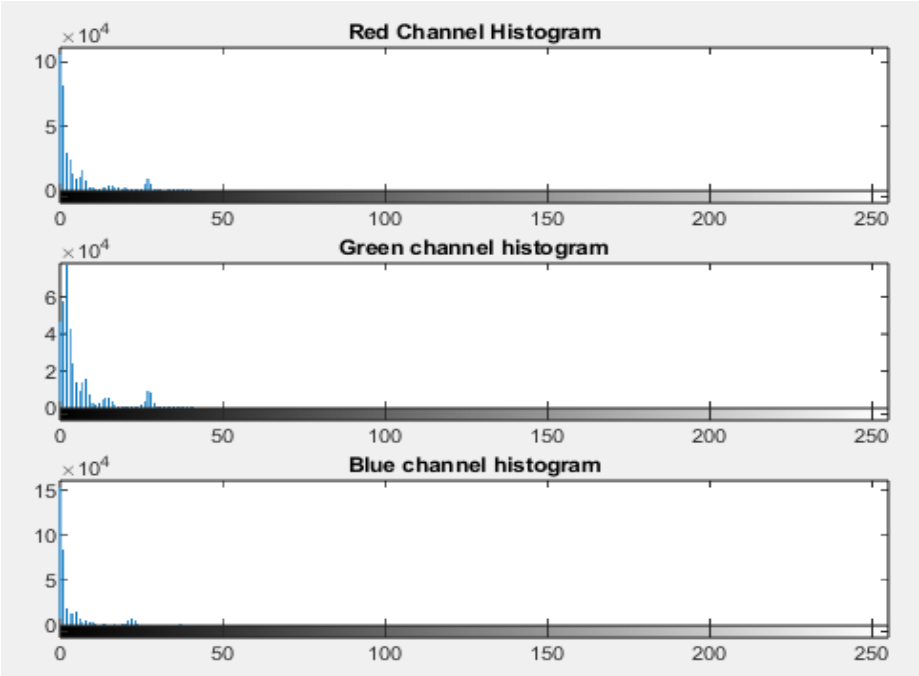


Figure 2.313: Histogram of LL image

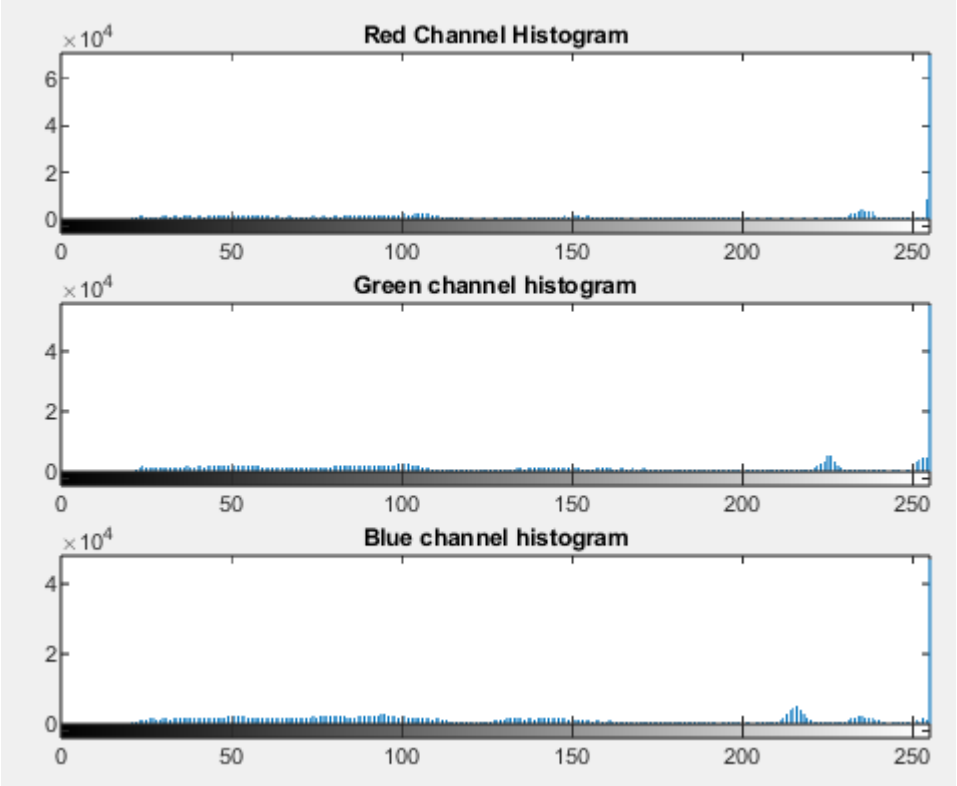


Figure 2.314: Histogram of NL image

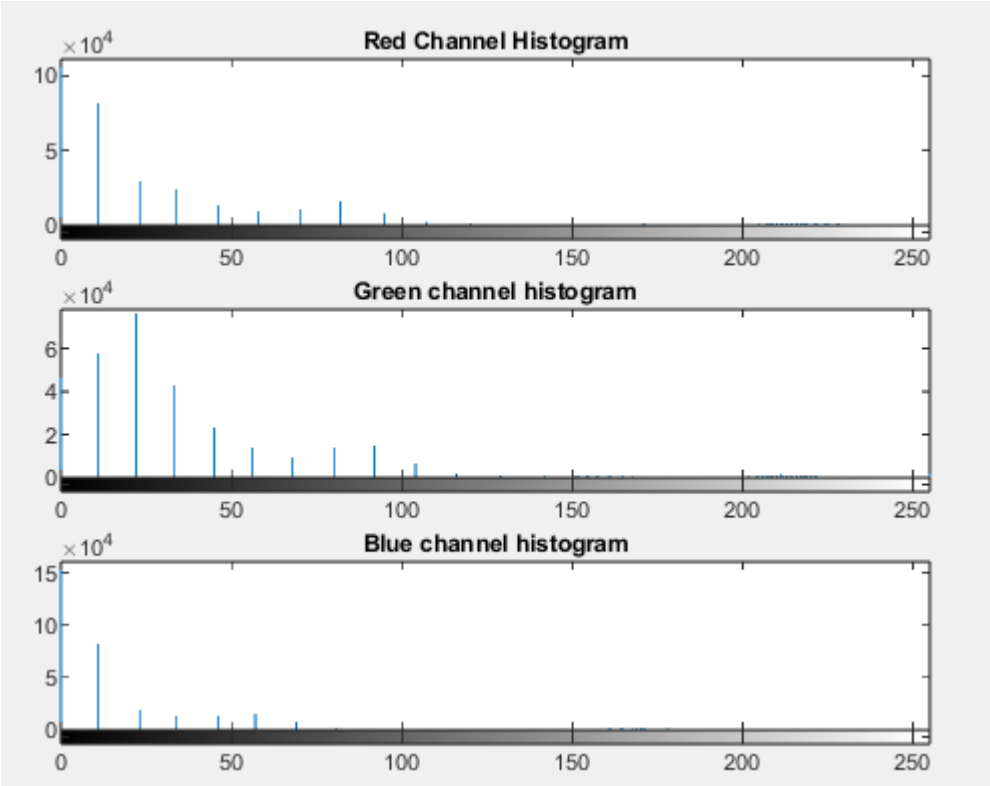


Figure 2.315: Histogram of 4.5 Experimental Result with min PSNR

For darkness level 4.5 in the case of the minimum experimental PSNR we see that the experimental result is characterized by intense color distortions, and remains darker than the ground truth case, explaining the minimum value of the metric. And from the histograms we see that the experimental histogram (figure 2.315) is completely different from the corresponding ground truth (figure 2.314). The same problem was also detected at darkness level 4.0 at the minimum experimental PSNR, where it was observed that while the ground truth histogram is distributed throughout the available range with most values accumulated in the right part, the corresponding experimental one fails to reproduce this information. Based on these we understand that the method has difficulties in such cases, producing low quality results with intense color distortions and noise.

MAX PSNR

Original Low Light **Normal Light** **Experimental Result**

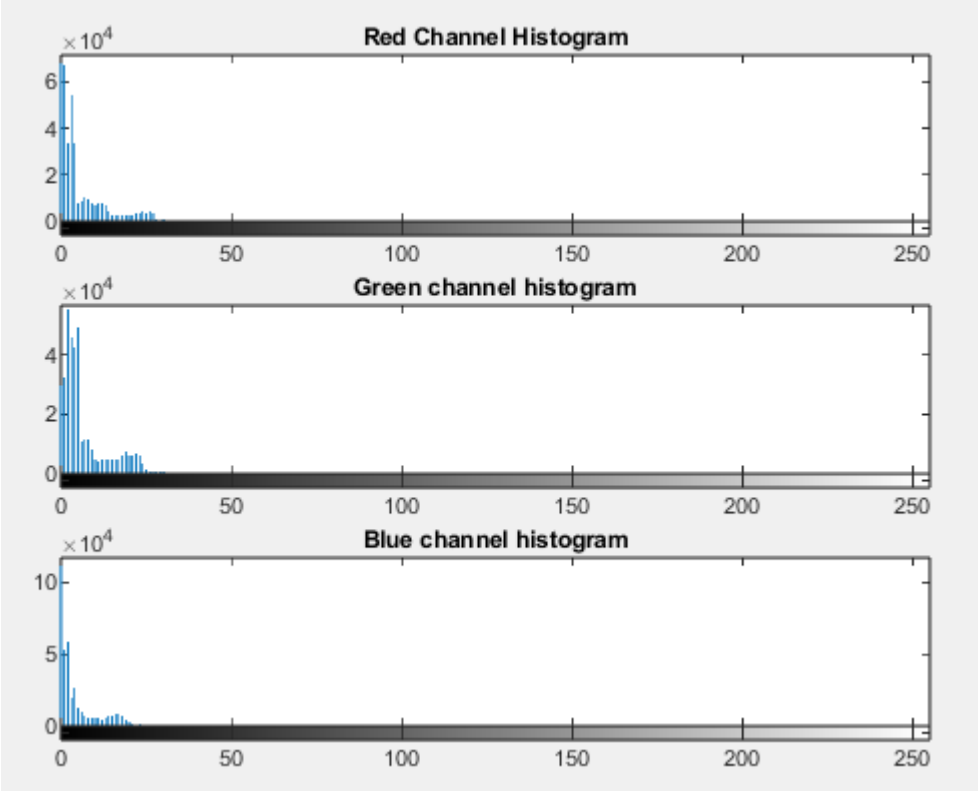


Figure 2.316: Histogram of LL image

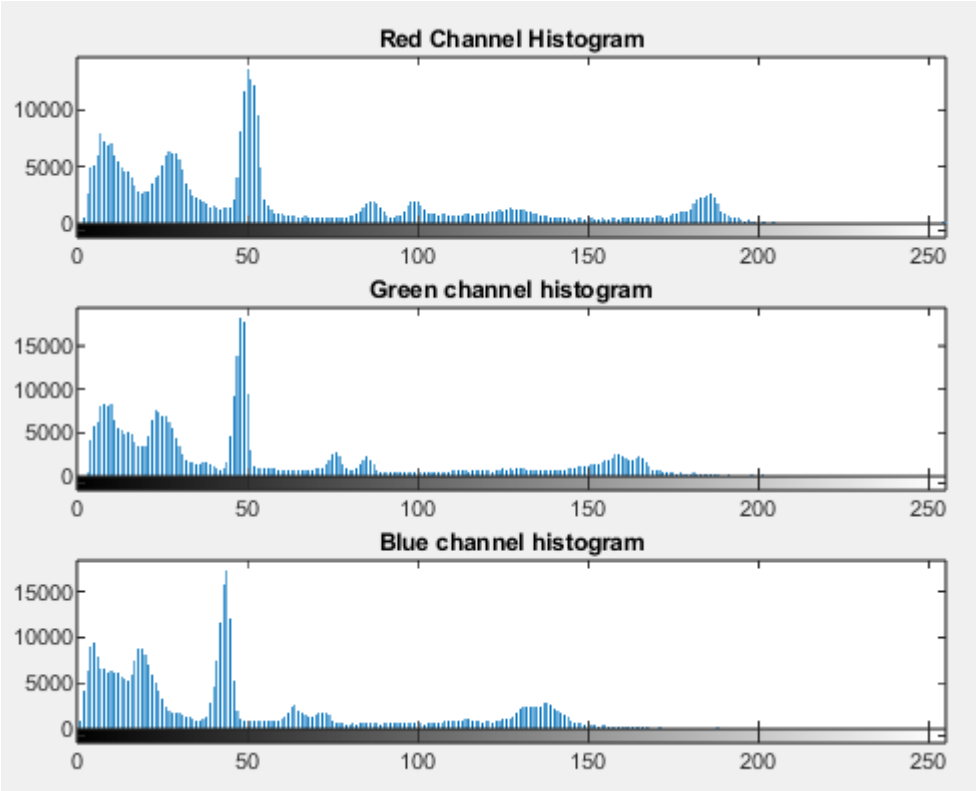


Figure 2.317: Histogram of NL image

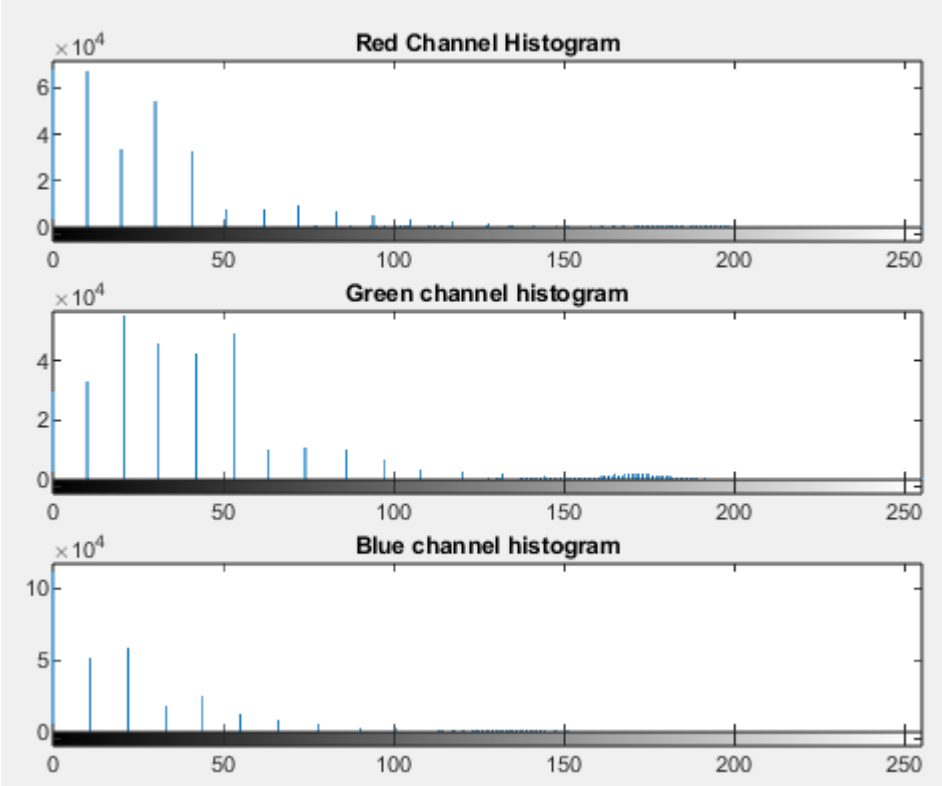
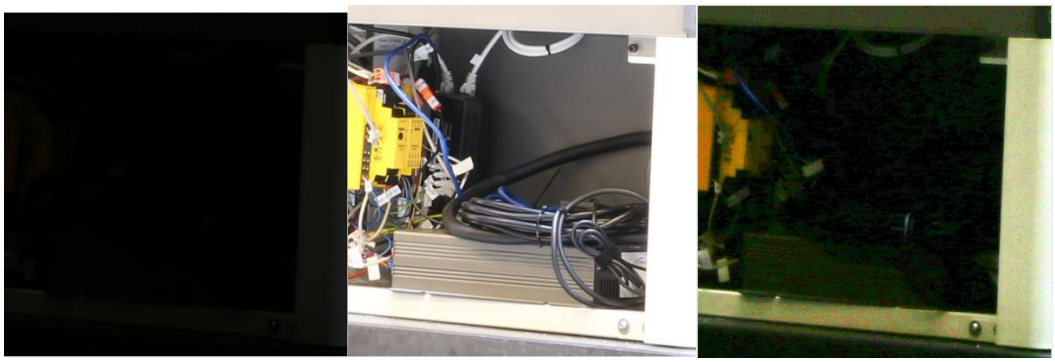


Figure 2.318: Histogram of 4.5 Experimental Result with max PSNR

In the case of the maximum experimental PSNR we see that the experimental result is visually very close to the ground truth case, with the impressive thing being that the reflections of the objects in the image have also been recovered. This is also confirmed by the histograms where we see that the experimental (figure 2.318) has a form very close to the corresponding ground truth (figure 2.317) with the only difference being that the pixel values are distributed in fewer gray value levels. This difference also explains some color changes observed in the experimental image.

Darkness Level: 5.0

MIN PSNR



Original Low Light **Normal Light** **Experimental Result**

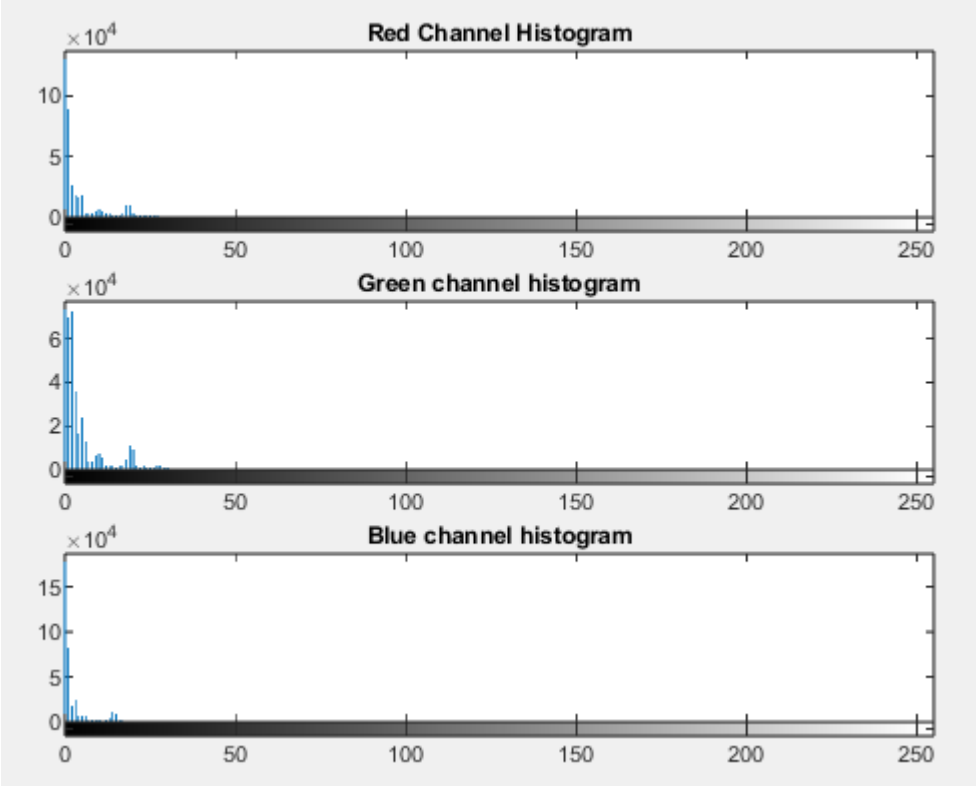


Figure 2.319: Histogram of LL image

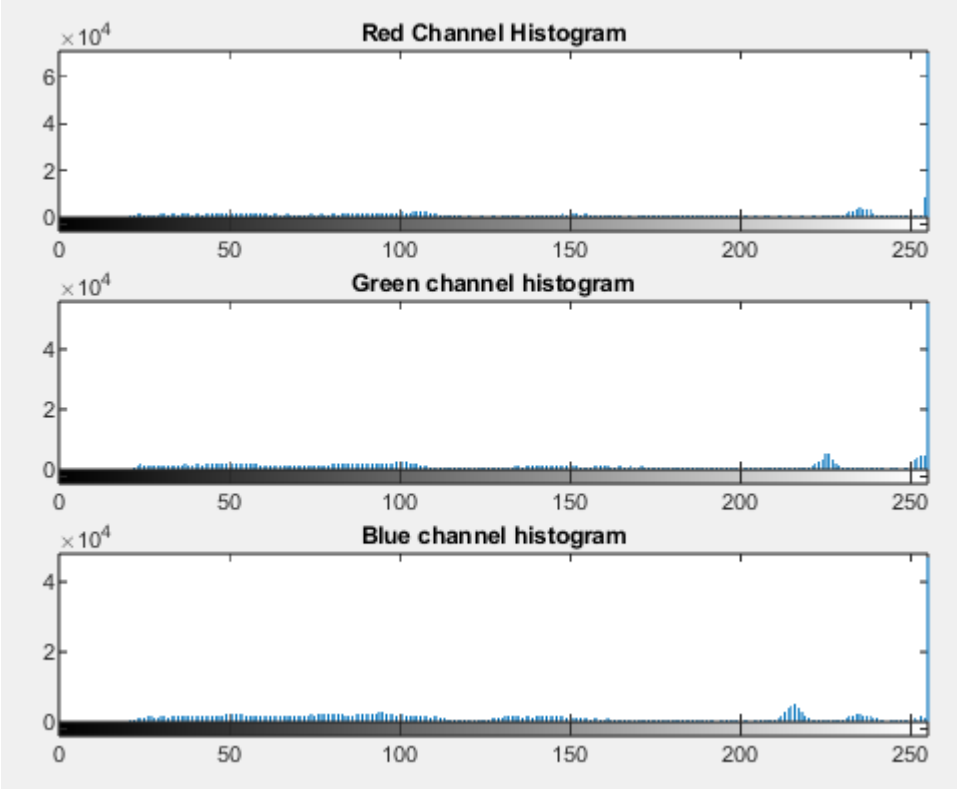


Figure 2.320: Histogram of NL image

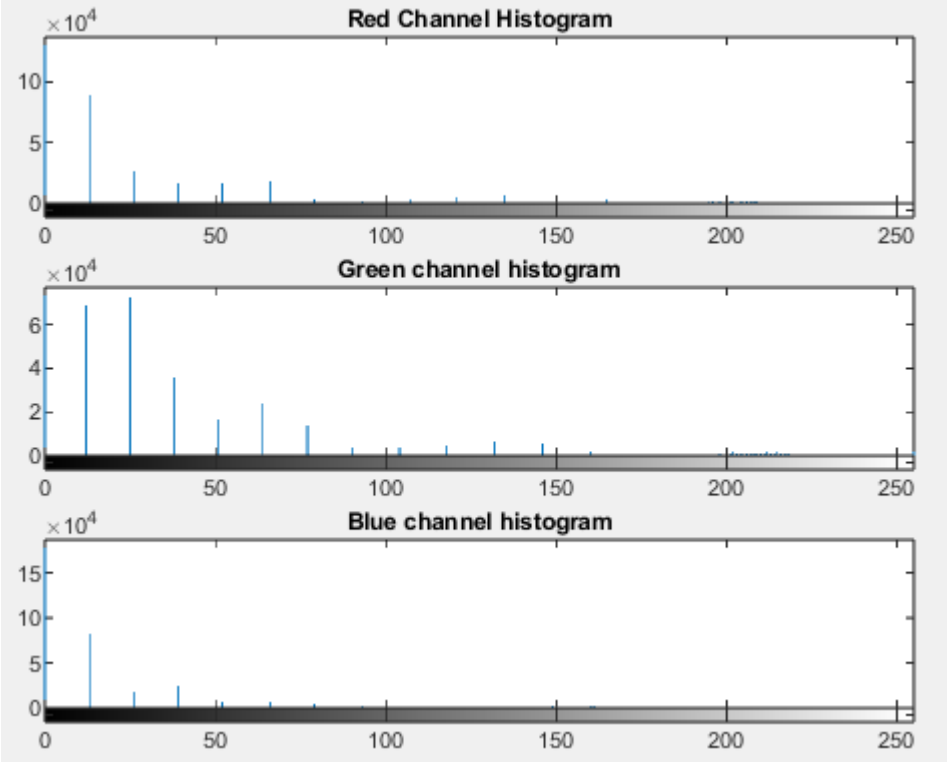
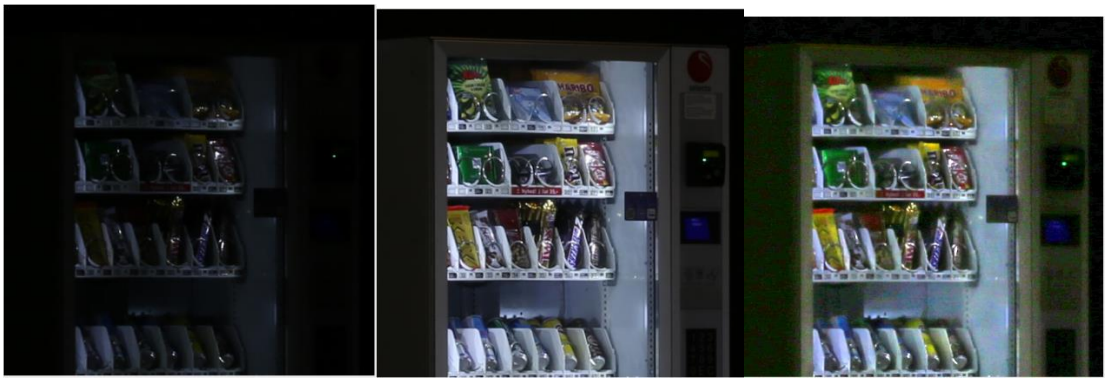


Figure 2.321: Histogram of 5.0 Experimental Result with min PSNR

For darkness level 5.0 in the case of the minimum experimental PSNR, the same image is obtained as for darkness level 4.5, confirming what we mentioned above. That is, the method has difficulty in images where the ground truth case is characterized by a histogram in which the values are clustered in the right part of the histogram.

MAX PSNR



Original Low Light **Normal Light** **Experimental Result**

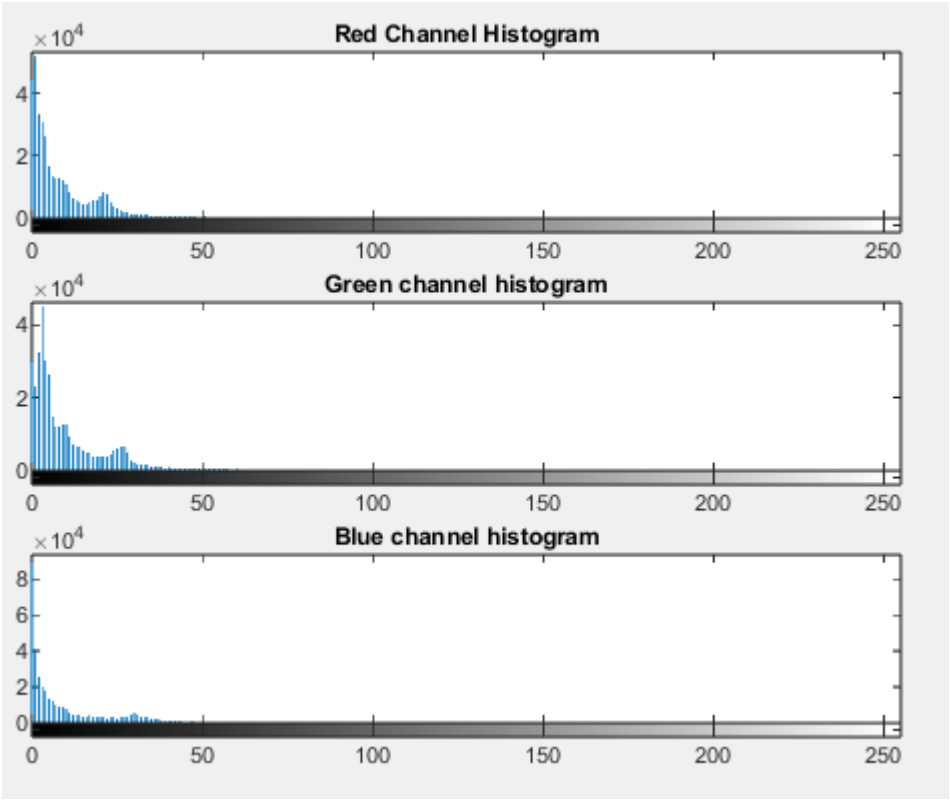


Figure 2.322: Histogram of LL image

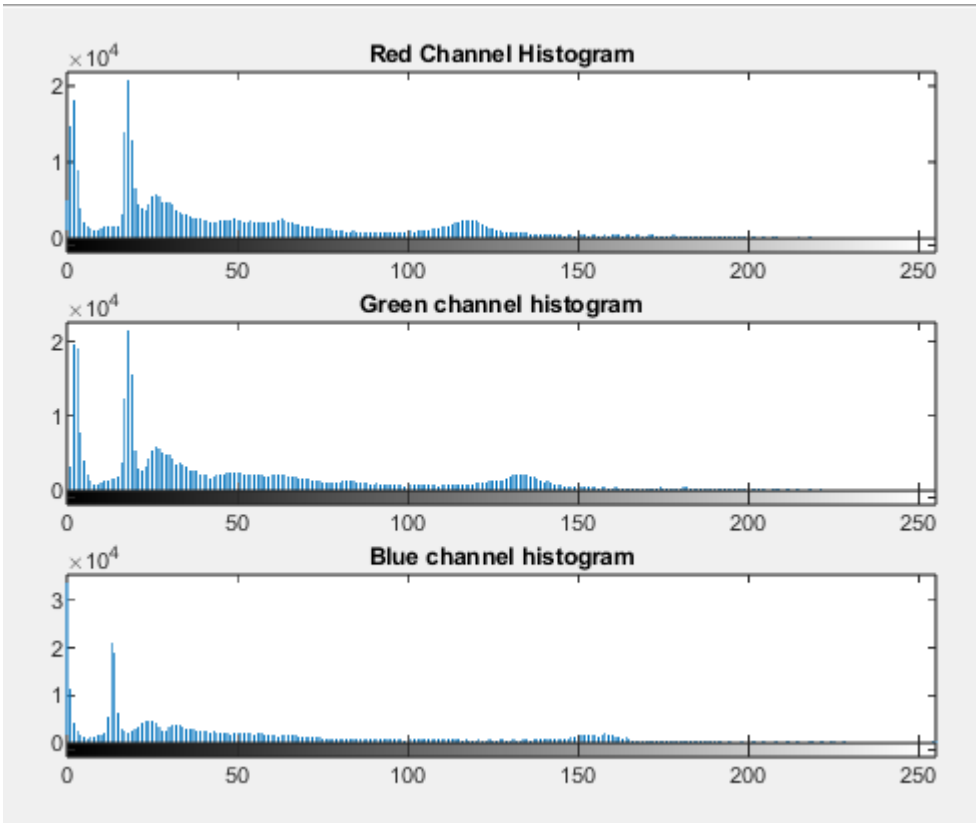


Figure 2.323: Histogram of NL image

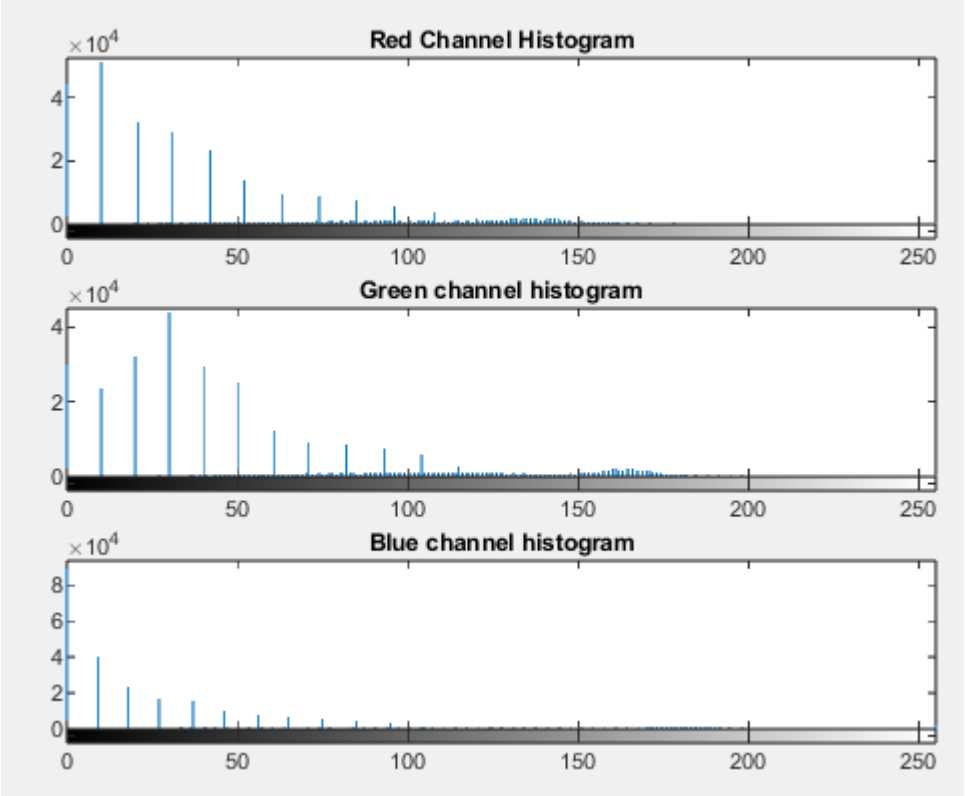


Figure 2.324: Histogram of 5.0 Experimental Result with max PSNR

In the case of the maximum experimental PSNR, the experimental result is visually very close to the ground truth case, and is characterized by minimal color distortions, which is particularly impressive as we are at the maximum darkness level. The high quality of the result is also confirmed by the histograms. From figure 2.324, with the experimental histogram, we see that it has a form very close to the corresponding ground truth case (figure 2.323), with the only difference being that the pixel values are distributed in fewer gray value levels. This difference also explains the color changes we mentioned above.

At this point it is worth commenting on the results of the method as a whole. We saw that the method significantly improves LL images by increasing the average PSNR by 8.2dB and the average SSIM by 0.38. In the case of the minimum experimental PSNR we saw that the method either over-enhances the image, leading to the introduction of additional noise and color distortions, or fails to recover certain features. We also saw that the method struggles on images where the pixel values of the ground truth image are clustered in the right part of the histogram. This is due to the nature of the method, as its purpose is to recover an image with pixel values spread over the entire available range and with high contrast. So, we understand that it is impossible to reproduce such cases. In the case of the maximum experimental PSNR we saw that the experimental results are visually very close to the ground truth case, with the method having recovered most of the visual information. Even at a very high darkness level the method performs very well, specifically mentioning the result at darkness level 4.5 where even the reflections of the objects have been recovered, as well as the result at darkness level 5.0 where most of the color information has been recovered. The high quality of the results is also confirmed by the histograms, as in all cases the experimental histogram has a form similar to the corresponding ground truth. The only difference observed is that in the experimental case, the pixel values are distributed in fewer gray value levels. This can be explained by the fact that the images are very dark, and the dynamic range between the bands is very small, making it difficult to fully recover the histogram distribution.

Chapter 3 – Deep Learning Techniques

3.1 Introduction

In this chapter we will use a completely different approach to the solution of the LLIE problem. A more general way to express the problem is through the relationship:

$$I_{LL} = f(I_{NL})$$

where LL image and NL image are connected through a function f(·), which introduces noise into the image making it darker. From this we understand that if we can find or approximate the function f, then we will have solved the LLIE problem. The methods we studied in the previous chapter are all model based, as they try to guess a model that describes the function f, which has 2 major drawbacks. The first is that in all cases constants are introduced, the values of which we must choose. Because every problem is different, we have to experiment each time with different values of the constants, and choose the best one from them, which does not guarantee us that we have actually found the optimal value, and it is also time-consuming. The second is that none of the models use information from the dataset, i.e. they are not data driven models. Both disadvantages can be addressed with learning methods, specifically Deep Learning which we will use in this section. deep learning techniques can be used to learn mappings, or more precisely approximations of mappings, between a specific input and a specific output using an iterative process, which minimizes the error between the desired output and the actual output of the DL architecture [30].

Since both the input and the output of the problem we are studying are images, we will use convolutional neural networks, which were first introduced by Lecun et al. [31] for automating the categorization of handwritten characters. The basic logic behind CNNs is to pass the input image through a convolutional layer which will extract features that can help solve the problem we are studying. The result of the first level is passed through a second level, extracting higher level features and so on.

In the case of character classification from [31] the final level features are flattened and passed through a set of neurons that will give the final result of the classification.

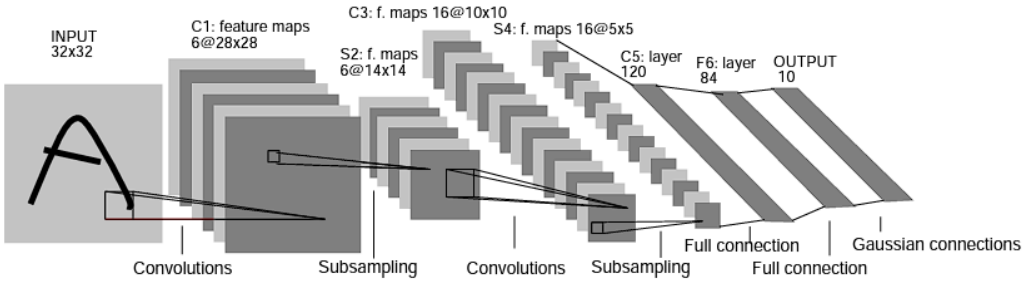


Figure 3.1: LeNet-5 architecture for handwritten digits [31]

To train such models we need a data set where each input corresponds to a unique defined output. In this case we have the LL images as well as the corresponding ground truth/NL images. This set should be broken into train/validation/test, with the training set used to train the model, the validation set to check that the model is not overfitting the training set during training, and the test set for the final control of the model's generalizability, i.e. its performance on data it has not encountered during training. In the training stage, an iterative algorithm called back-propagation [32] is applied, which minimizes the error between the desired output (y_{real}) and the actual output of the model (y_{model}), which is a function of the form:

$$Error = D(y_{real}, y_{model})$$

Based on the above we understand that we have to make some decisions, such as what form the error function will have and how many convolutional levels the architecture we will build will have. CNNs combine a set of feature maps to achieve the final result, so we expect to get a better result the deeper the network, i.e. the more convolutional layers the model has. One of the first problems created by increasing the number of convolutional levels is the vanishing/exploding gradients, the existence of which slows down the training of the model [33]. The second is the degradation problem where it was observed that as the depth of the network increases, its accuracy saturates and then begins to decline [34]. The two

main ideas proposed to deal with these problems, as well as to increase the accuracy of the models, are Residual Learning [34] and the Inception module [35], which we will implement during the construction of the architecture we will use. The basic idea of Residual Learning is to use shortcut connections between convolutional layers, as shown in figure 3.2.

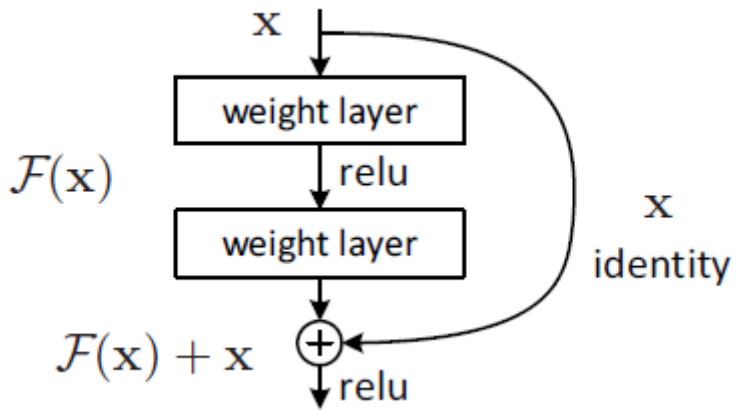


Figure 3.2: Shortcut connection in Residual Learning net [34]

In the Inception module the basic idea is that multiple layers connect to the same previous layer, and their outputs are concatenated into an output vector [35].

In this chapter we will try to construct a DL architecture, which will solve the LLIE problem by learning the mapping between the space of LL images and the space of NL images, using the two basic ideas we mentioned above. We will then enrich this architecture in an effort to improve the result each time, studying different variations. The data is already broken into train/test/validation, and we will use the train data to train the architecture, the validation to confirm that the architecture does not overfits the train data, and the test data to evaluate the generalization ability of the model. At this point we should comment that since we are only interested in the generalization ability of each model, we will present and comment on the results only from the test set, which includes images that the model did not encounter during training.

3.2 LL-CNN: Original Architecture

The basic architecture we will use was proposed by Tao et al. [36] and is called Low Light Convolutional Neural Network (LL-CNN). Its main purpose is to learn suitable feature maps, the combination of which produces enhanced images, which appear as if they were taken under normal lighting conditions. The architecture used by the authors is shown in the figure below.

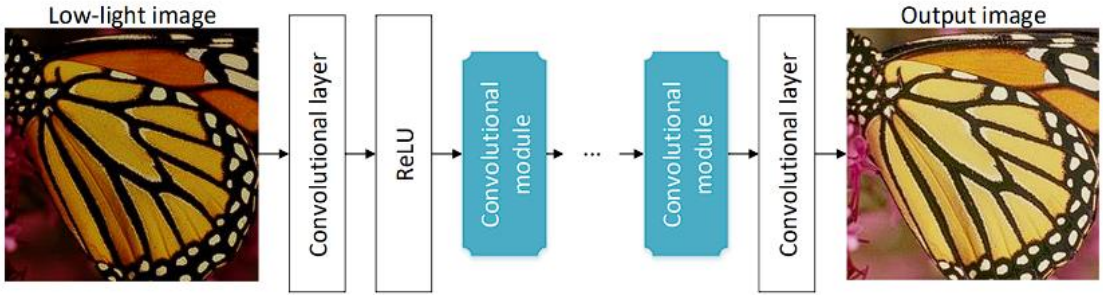


Figure 3.3: LL-CNN architecture [36]

We see that the input image first passes through a convolutional layer, the purpose of which is pre-processing. At the end, before the output, there is another convolutional layer whose purpose is to combine the feature maps into the final output image, i.e. to make the output have the appropriate size. In between, there are properly designed convolutional modules, which are designed to deal with the problem of vanishing gradients, offering the possibility to make the architecture deeper. The form of the convolutional module is shown in figure 3.4 below.

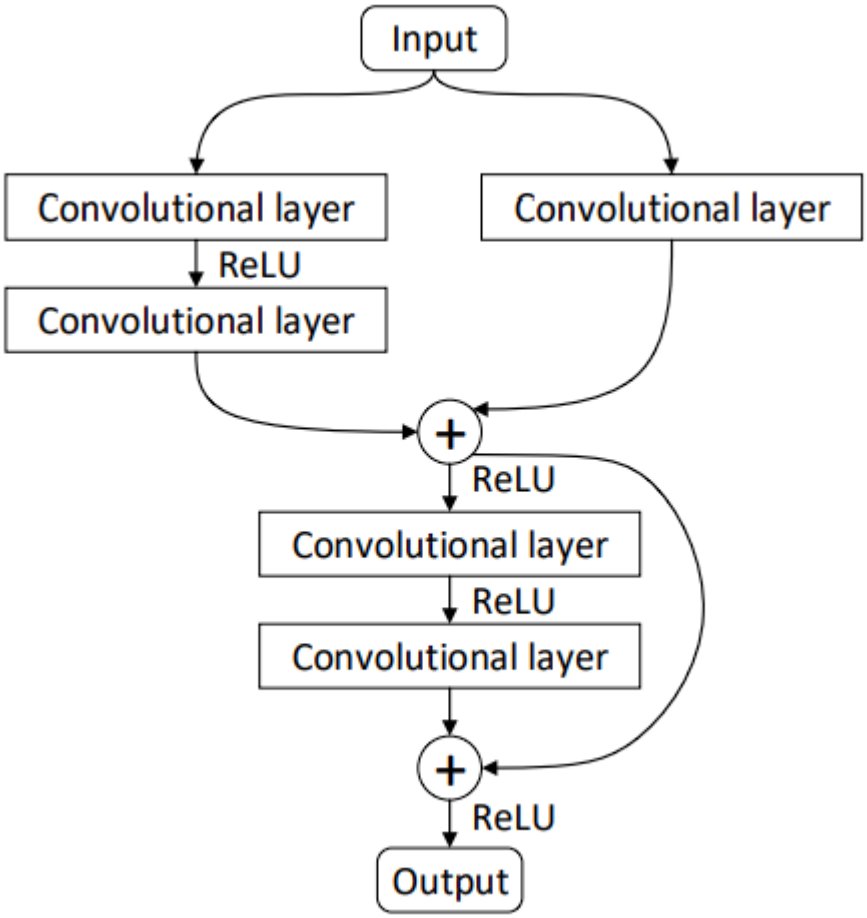


Figure 3.4: Convolutional module architecture [36]

We see that the convolutional module is inspired by the Inception module and Residual Learning. Initially, the input of the module is processed in two different ways, on the left it goes through 2 convolutional layers (3x3) and on the right through one convolutional layer (1x1). The results are combined into a vector, simply instead of being concatenated, they are added, a process inspired by Inception Modules. Then, the resulting vector goes through two paths, first through 2 convolutional layers (3x3) and secondly, it goes straight to the output bypassing the two convolutional layers. These two results are added together, a process inspired by Residual Learning. Each convolutional layer of the architecture uses 64 filters (ie 64 feature maps), except for the last one where the number of filters depends on the number of color channels we want the output to have. Based on these we understand that we have to choose the number of convolutional modules that we will use. For the sake of experimentation, we will apply the model three different times with 1, 3 and 5 convolutional modules respectively, so that we can compare the performance in relation to the number of modules. Finally, each convolutional layer has ReLU as

activation function, while as error function we use Mean Squared Error (the authors define an error function based on SSIM, but for simplicity we change it and use MSE). More details on the implementation, training and testing of the model are presented in appendix B.

In the following, we present the results of each architecture on the test data only, since we are interested in the generalization ability of the models. We construct tables with the minimum/maximum/average value of the quality metrics, as well as line charts with the corresponding PSNR and SSIM, in order to see how the darkness level affects the performance of the model. In addition, we also construct line charts with the average value of PSNR and SSIM for all modules as well as for the LL data, in order to see which number of modules gives the best results and how much the images improve.

1 module

MSE					
Level	3.0	3.5	4.0	4.5	5.0
MIN	64,00632	65,71979	117,2084	117,3277	626,9481
MAX	3743,257	3609,557	5276,802	4208,941	7417,438
AVERAGE	589,5228	595,127	999,0249	858,434	2655,512
	PSNR(dB)				
Level	3.0	3.5	4.0	4.5	5.0
MIN	12,39831	12,55626	10,9071	11,88908	9,428264
MAX	30,06858	29,95384	27,44122	27,4368	20,15849
AVERAGE	21,99681	21,9135	19,22709	20,01199	14,22402
	SSIM				
Level	3.0	3.5	4.0	4.5	5.0
MIN	0,24571	0,237992	0,14365	0,024684	0,028431
MAX	0,899974	0,900537	0,792575	0,753381	0,605255
AVERAGE	0,697193	0,682368	0,578808	0,485678	0,361786
	MV				
Level	3.0	3.5	4.0	4.5	5.0
MIN	45,49428	45,63054	28,00752	47,55223	99,52001
MAX	128,4906	127,5039	115,8126	129,9237	156,0764
AVERAGE	99,37198	98,72828	79,32872	98,06009	133,7876
	STD				
Level	3.0	3.5	4.0	4.5	5.0
MIN	59,83054	74,57957	49,11963	52,83164	66,23288
MAX	930,2407	975,8653	832,8613	978,381	1056,601
AVERAGE	328,6329	341,9338	255,5311	345,315	467,8872

	BRISQUE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	37,15248	38,68091	38,67962	42,59008	43,46492
MAX	53,66808	54,99189	57,07783	57,97389	57,34084
AVERAGE	45,59591	46,18324	48,91179	49,24229	50,01282
	NIQE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	4,231179	3,967828	4,305972	4,113166	4,130589
MAX	7,562505	7,48527	6,764829	5,998782	5,515432
AVERAGE	5,073763	4,99647	5,289502	4,768394	4,596427

Table 3.1: LLCNN results on test set – 1 CNN module

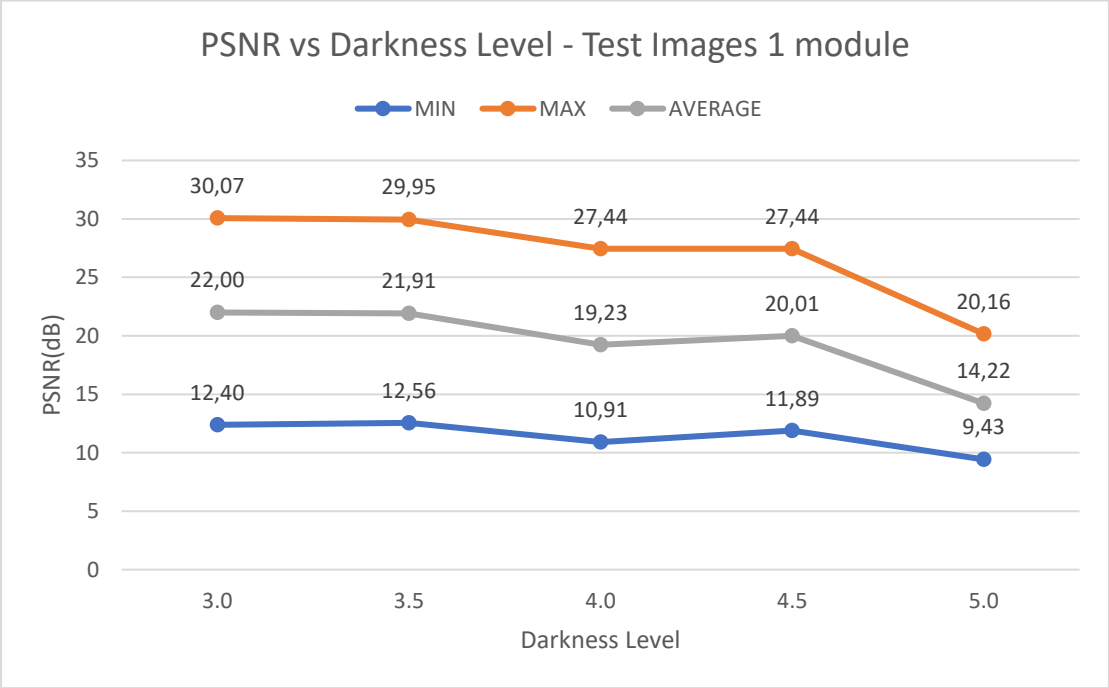


Figure 3.5: Experimental results PSNR vs Darkness level for test set

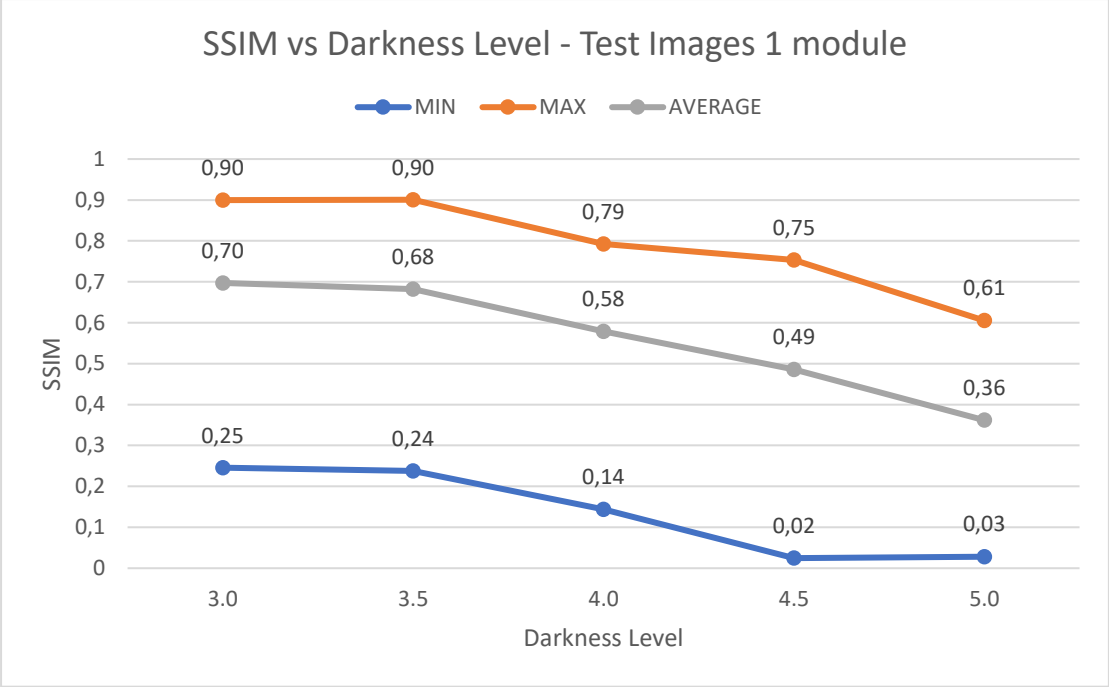


Figure 3.6: Experimental results SSIM vs Darkness level for test set

For one convolutional module we see that the values of the quality metrics improve noticeably. The MSE is greatly reduced, compared to the original images, and the PSNR and SSIM are increased. For the quality metrics with no reference we notice that MV and STD increase, which means that the images are now indeed brighter, with pixel values spread more widely around the mean value, i.e. the images also have bigger contrast. BRISQUE and NIQE continue to have a large value, which means that DL techniques cannot recover the complete information of natural statistics. Furthermore, we see that the expected path is followed with increasing darkness level, as the quality of the metrics decreases as it increases. MSE increases, while PSNR and SSIM decrease. This is due to the fact that by increasing the darkness level, the images become very dark, making it difficult to retrieve the full information.

3 modules

MSE					
Level	3.0	3.5	4.0	4.5	5.0
MIN	59,77004	36,24526	70,22316	392,4378	241,2912
MAX	3918,951	3963,741	3545,339	5257,203	3509,249
AVERAGE	634,722	717,4708	653,3445	1130,7	1002,31
	PSNR(dB)				
Level	3.0	3.5	4.0	4.5	5.0
MIN	12,19911	12,14975	12,63423	10,92326	12,67866
MAX	30,36597	32,53829	29,666	22,19309	24,30539
AVERAGE	21,80508	21,04742	21,32991	18,20765	18,90393
	SSIM				
Level	3.0	3.5	4.0	4.5	5.0
MIN	0,324578	0,315974	0,103897	0,240917	0,039864
MAX	0,913249	0,900375	0,888413	0,764861	0,672576
AVERAGE	0,695826	0,690745	0,607084	0,498138	0,413824
	MV				
Level	3.0	3.5	4.0	4.5	5.0
MIN	34,23039	37,90919	46,00376	65,432	56,01839
MAX	119,4801	114,8552	127,4471	140,9266	125,7178
AVERAGE	86,52921	84,36512	98,0996	112,9711	97,62692
	STD				
Level	3.0	3.5	4.0	4.5	5.0
MIN	67,43443	46,93583	47,66101	96,73802	32,45479
MAX	771,1021	649,7872	868,719	938,4278	759,7665
AVERAGE	238,4436	224,6997	290,6476	339,4225	256,864
	BRISQUE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	34,3626	36,70928	40,5214	43,04002	43,45819
MAX	54,27313	52,29309	55,40144	57,22629	58,06572
AVERAGE	46,0759	45,55895	47,57644	49,40058	51,77529
	NIQE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	4,036433	4,351935	4,643276	4,514113	4,776201
MAX	8,754551	7,740933	7,474711	6,651145	5,918743
AVERAGE	5,230862	5,098118	5,175296	5,147363	5,219693

Table 3.2: LLCNN results on test set – 3 CNN modules

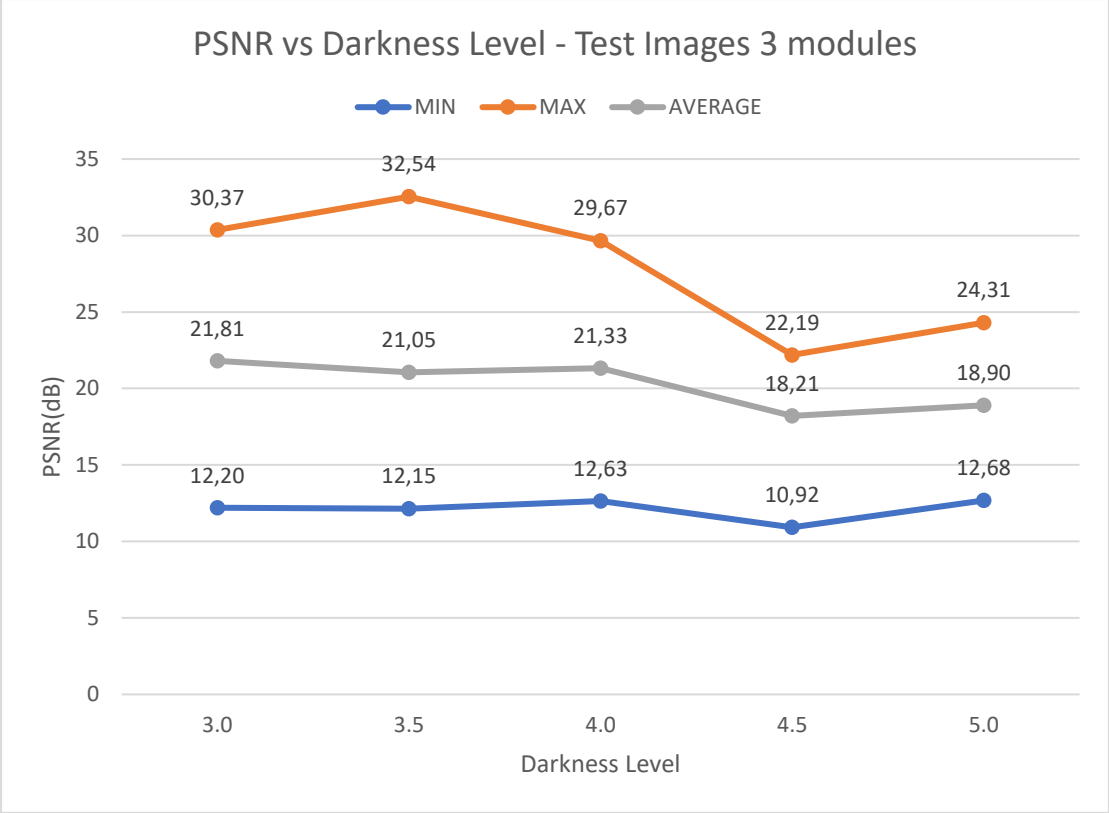


Figure 3.7: Experimental results PSNR vs Darkness level for test set

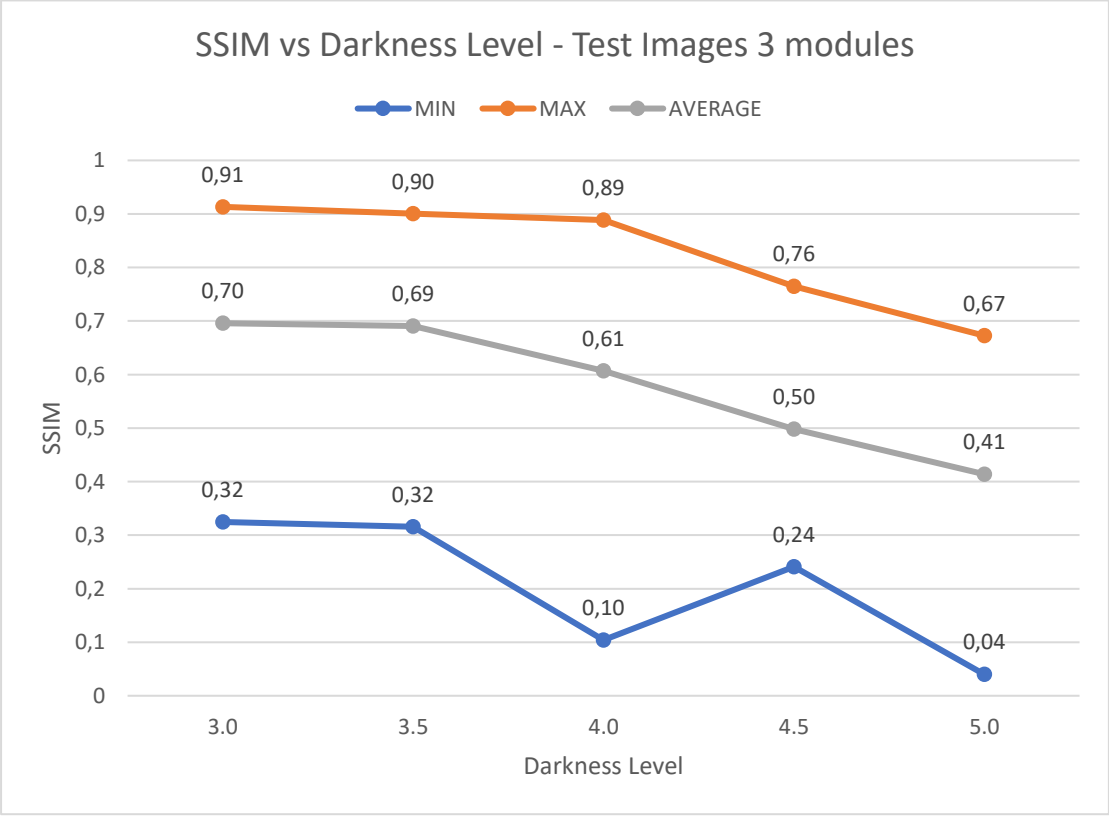


Figure 3.8: Experimental results SSIM vs Darkness level for test set

For 3 convolutional modules the values of the metrics are greatly improved, with the MSE decreasing and the PSNR and SSIM increasing indicating the increase in the quality of the results. Also, MV and STD are increased which means images are brighter, and with more contrast, as the pixel values are spread over a wider range around the average brightness. BRISQUE and NIQE also have large values here, which means that we are moving away from natural statistics, even if the result improve visually. Finally, we see that the expected behavior is followed with the increase of the darkness level, as the quality of the metrics decreases with its increase.

5 modules

MSE					
Level	3.0	3.5	4.0	4.5	5.0
MIN	43,95319	270,1427	177,359	191,175	458,578
MAX	3362,776	4603,534	5294,692	5870,252	6906,46
AVERAGE	521,8074	809,4739	810,0255	1300,905	2159,329
	PSNR(dB)				
Level	3.0	3.5	4.0	4.5	5.0
MIN	12,86382	11,49989	10,8924	10,44424	9,738249
MAX	31,7009	23,81487	25,64227	25,31649	21,51667
AVERAGE	23,10539	19,89816	20,15362	17,76384	15,30689
	SSIM				
Level	3.0	3.5	4.0	4.5	5.0
MIN	0,178961	0,224381	0,249354	0,020598	0,028057
MAX	0,927742	0,881537	0,82451	0,718493	0,704068
AVERAGE	0,735481	0,696412	0,586369	0,359358	0,326991
	MV				
Level	3.0	3.5	4.0	4.5	5.0
MIN	38,91114	61,88751	52,09949	55,82323	62,1317
MAX	124,3378	136,6255	139,8324	119,9074	99,69276
AVERAGE	95,72208	111,1108	106,6565	86,97369	76,94136
	STD				
Level	3.0	3.5	4.0	4.5	5.0
MIN	60,9497	93,02488	86,39266	149,7656	103,4771
MAX	929,1197	879,7557	1028,407	572,7684	436,5832
AVERAGE	303,6118	298,8985	330,4392	316,7163	310,8114
	BRISQUE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	35,93147	36,67733	43,41745	43,45818	44,95327
MAX	52,97696	53,56573	56,73413	63,55916	66,15108
AVERAGE	44,40987	45,9194	49,06635	55,40452	56,17681

	NIQE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	4,020012	4,294638	4,452095	5,002217	5,451129
MAX	8,350197	7,704777	7,106451	9,033405	9,518737
AVERAGE	4,964992	5,456095	5,265027	5,838698	6,648088

Table 3.3: LLCNN results on test set – 5 CNN modules

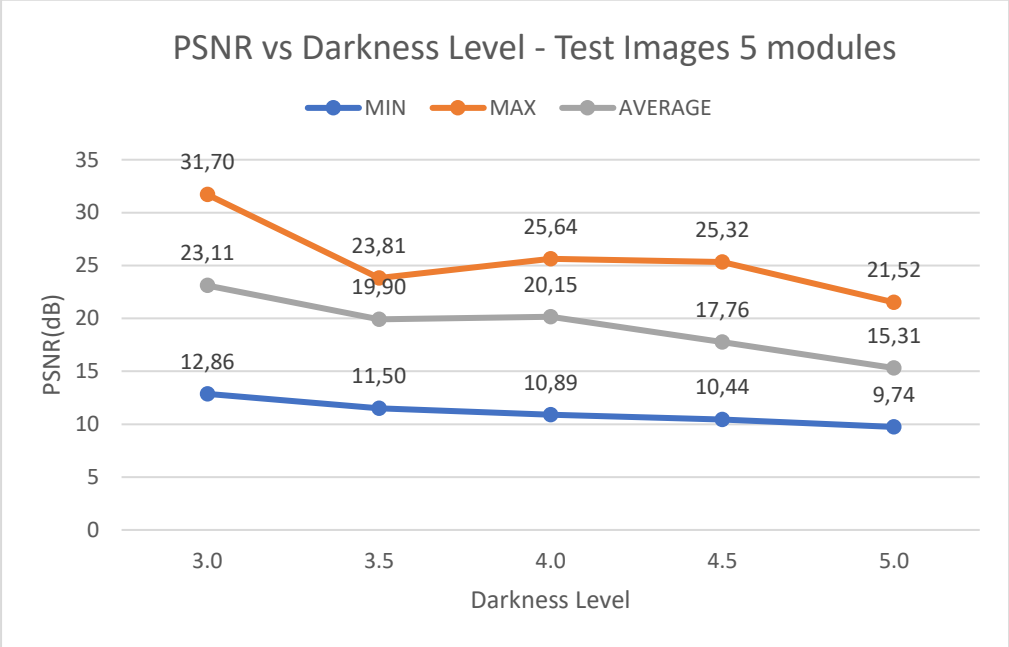


Figure 3.9: Experimental results PSNR vs Darkness level for test set

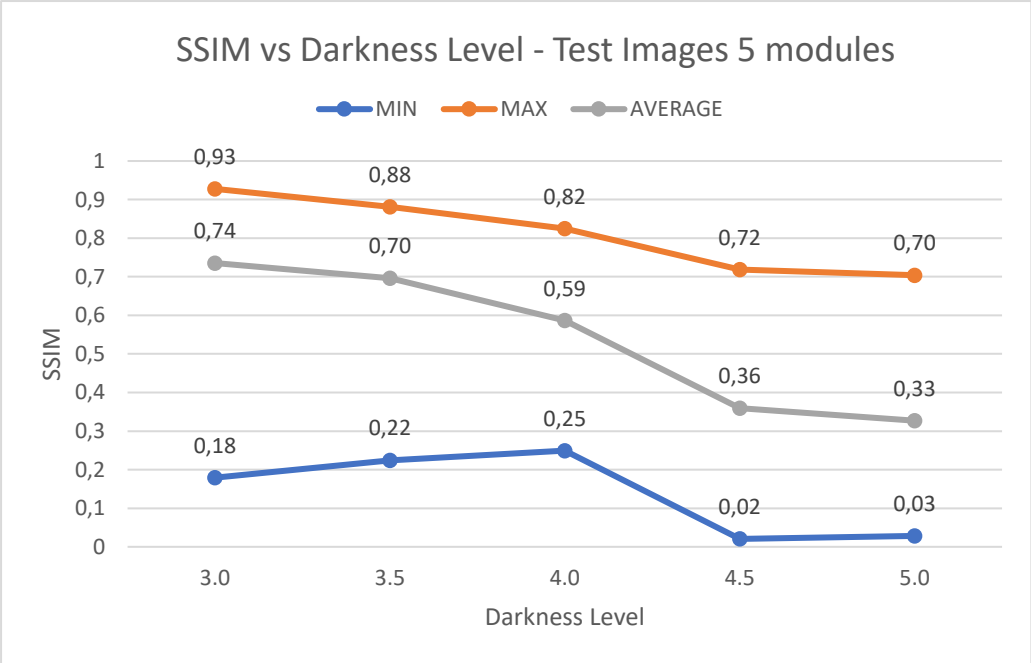
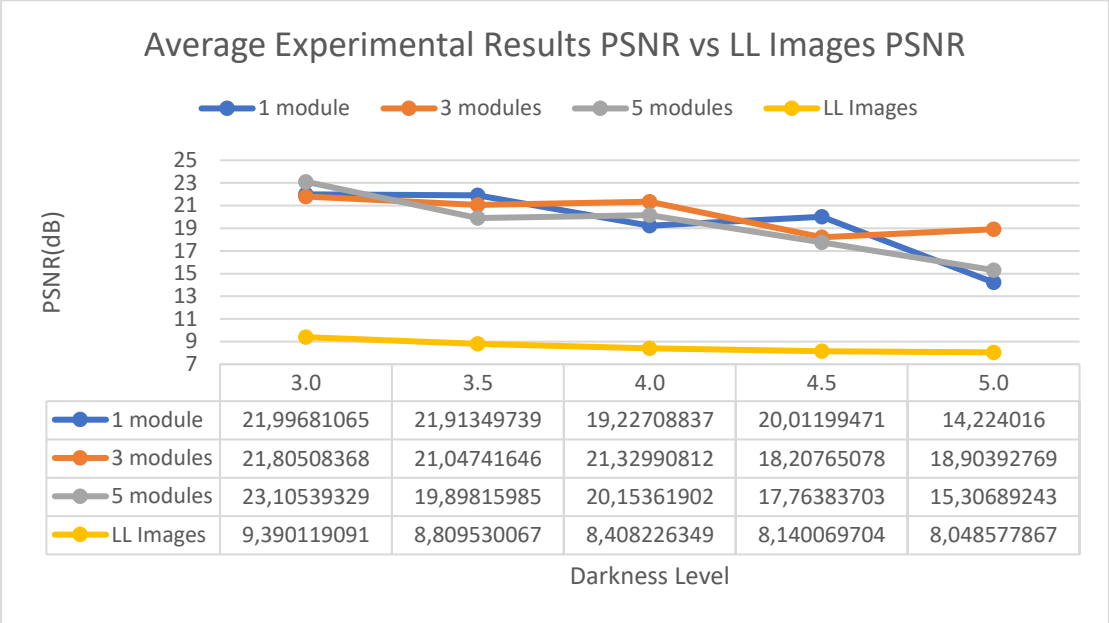


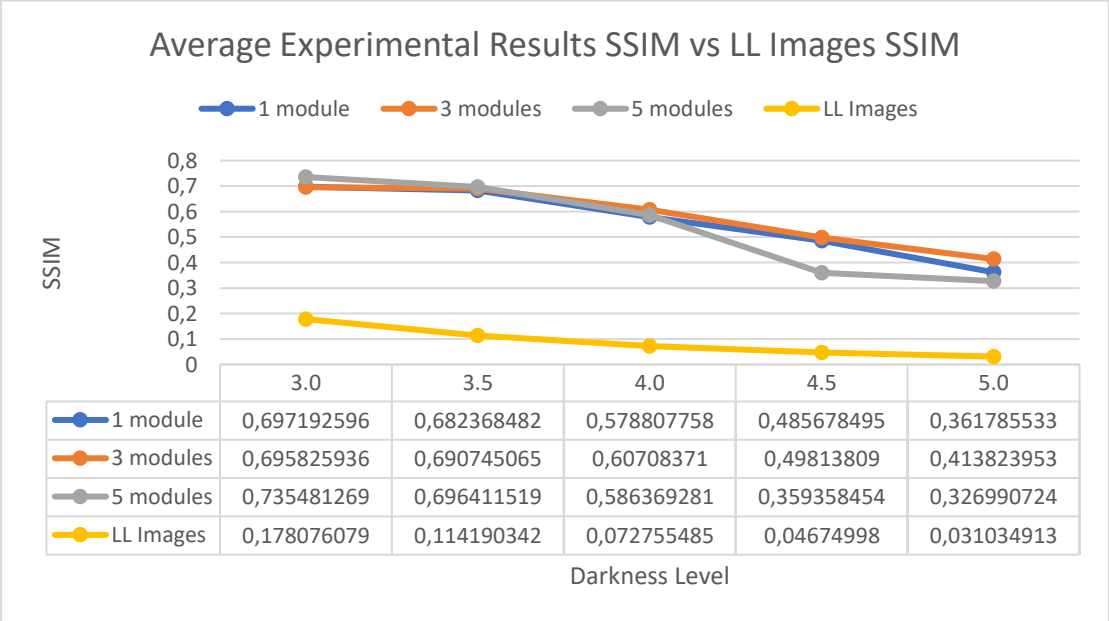
Figure 3.10: Experimental results SSIM vs Darkness level for test set

For 5 convolutional modules again the values of the metrics improve noticeably as the MSE decreases and the PSNR and SSIM increase, compared to the original LL images. Also, MV and STD are increased, meaning images become brighter and with bigger contrast. BRISQUE and NIQE continue to have large values, which confirms to us that DL models have difficulty recovering natural statistics. Finally, we must comment that again with the increase in the darkness level the performance of the model decreases, which is to be expected for the reasons we described above.



	3.0	3.5	4.0	4.5	5.0
1 module	21,99681065	21,91349739	19,22708837	20,01199471	14,224016
3 modules	21,80508368	21,04741646	21,32990812	18,20765078	18,90392769
5 modules	23,10539329	19,89815985	20,15361902	17,76383703	15,30689243
LL Images	9,390119091	8,809530067	8,408226349	8,140069704	8,048577867

Figure 3.11: Experimental results average PSNR vs LL Images PSNR test set



	3.0	3.5	4.0	4.5	5.0
1 module	0,697192596	0,682368482	0,578807758	0,485678495	0,361785533
3 modules	0,695825936	0,690745065	0,60708371	0,49813809	0,413823953
5 modules	0,735481269	0,696411519	0,586369281	0,359358454	0,326990724
LL Images	0,178076079	0,114190342	0,072755485	0,04674998	0,031034913

Figure 3.12: Experimental results average PSNR vs LL Images PSNR test set

At this point it is worth commenting on the results as a whole. In all cases we saw that the metric values were greatly improved, with MSE decreasing and PSNR and SSIM increasing, indicating the effectiveness of this method. In addition, MV and STD are increased, which means that the experimental images are brighter and with higher contrast, since the pixel values are spread over a larger range around the mean brightness. The only negative thing we have to comment on is that BRISQUE and NIQE do not improve, with their values remaining large, which shows us that the results obtained from DL methods are far from natural statistics. This is because as the image passes through the various layers of the model, it is subjected to several filters leading to this effect. Finally, in all cases we observe that with the increase of the darkness level the performance of the model decreases, as the quality of the values of the metrics decreases. This is due to the fact that as the darkness level increases, the images become too dark, and the dynamic range too small, which makes it difficult to retrieve visual information. A solution to this would be to train the architecture more for larger darkness levels to give the model time to learn more. Nevertheless, the improvement in the metrics is impressive, and the fact that these results are obtained on our test set shows that the model has a strong generalization ability.

Regarding the number of convolutional modules, from figures 3.11 and 3.12 we see that there are no big differences in the performance of each case. One module achieves an increase in average PSNR by 10.91dB, 3 by 11.71dB and 5 by 10.68dB, while for SSIM the improvement is by 0.47, 0.5 and 0.45 respectively. In addition, we see that for large darkness levels the 3 modules give slightly better results, so in the end these are the ones that perform best. Finally, let us comment that all 3 cases have been trained for the same number of epochs, so there is a possibility that the 5 modules would have given better results if we had trained them more, since they have more parameters.

Overall, we saw very impressive results in terms of metric values. To see the results visually we will present, for each darkness level, the results for a random image, together with the corresponding LL and ground truth cases. We will also present their histograms to see if the correct information for the histogram was retrieved.

Darkness Level: 3.0


Original Low Light **Normal Light**



1 module **3 modules** **5 modules**

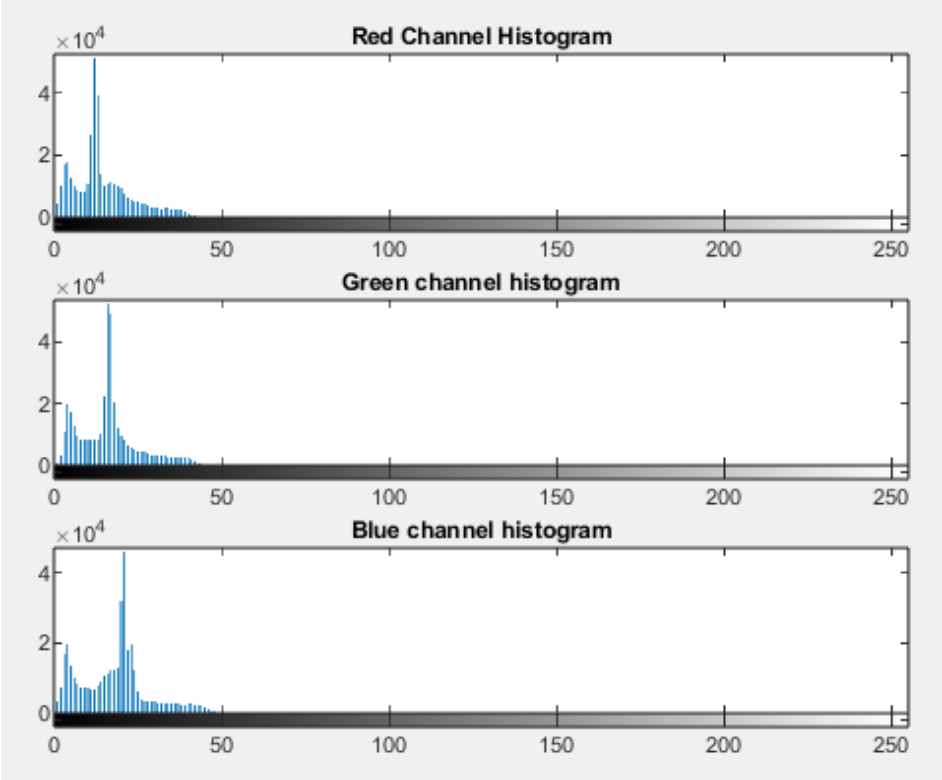


Figure 3.13: Histogram of 3.0 LL image

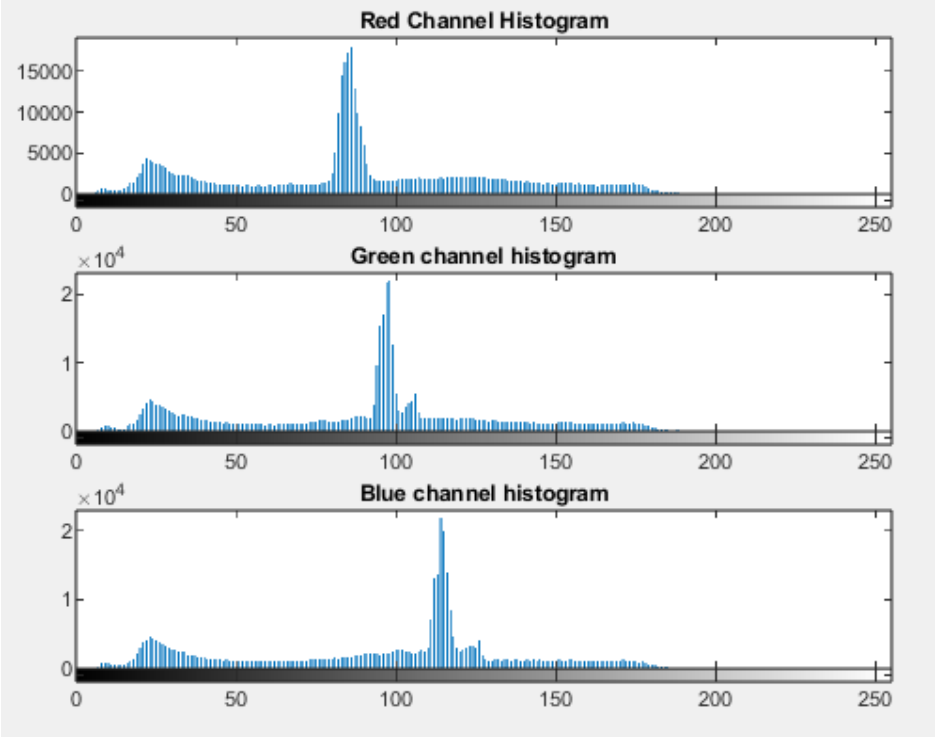


Figure 3.14: Histogram of 3.0 NL image

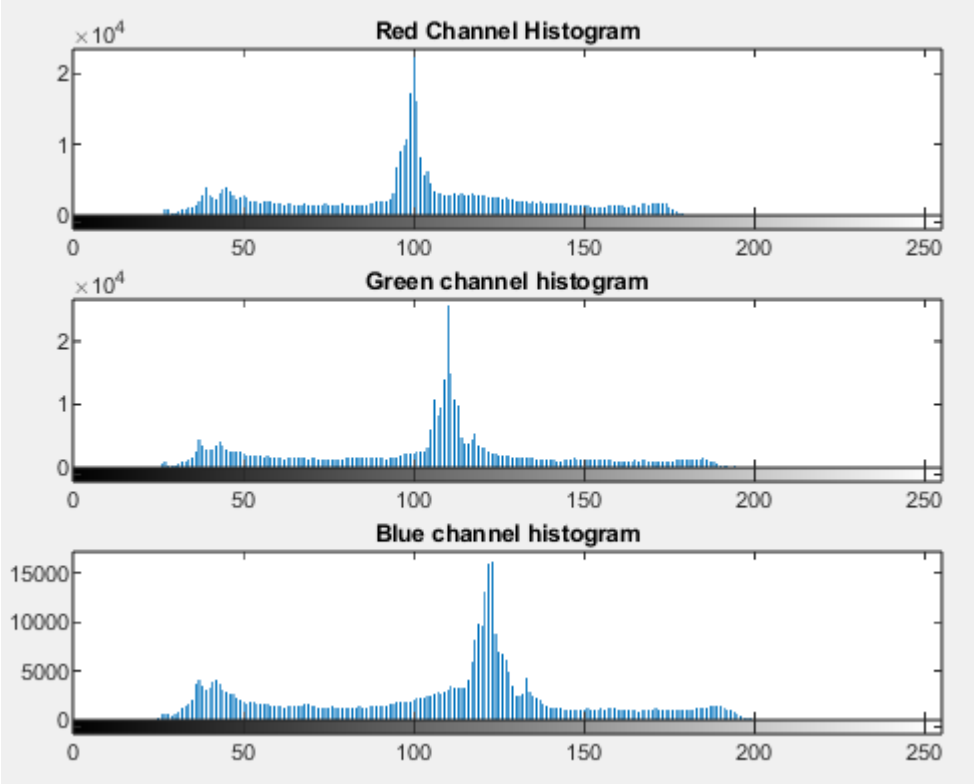


Figure 3.15: Histogram of 3.0 experimental result with 1 module

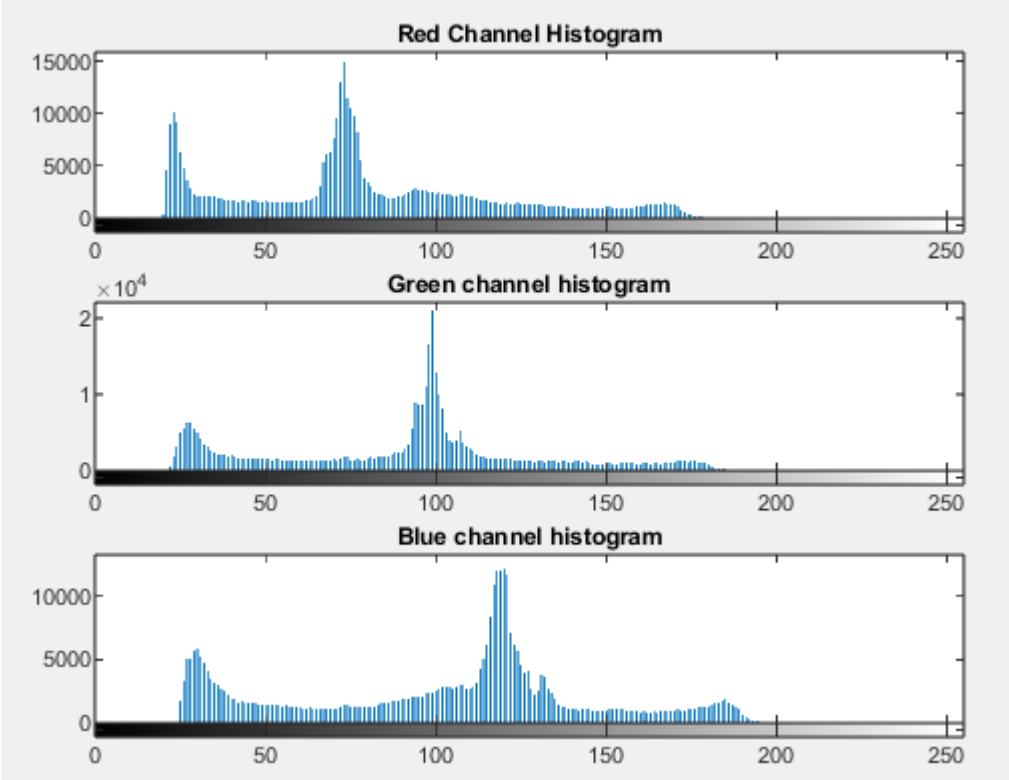


Figure 3.16: Histogram of 3.0 experimental result with 3 modules

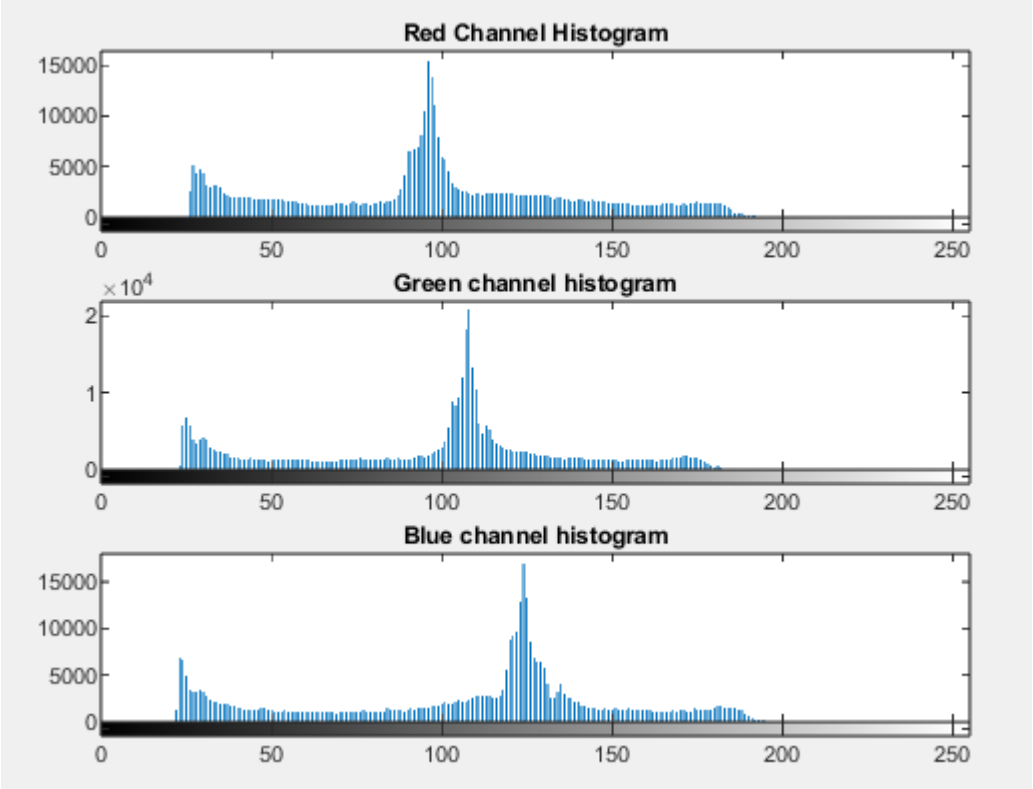


Figure 3.17: Histogram of 3.0 experimental result with 5 modules

For darkness level 3.0 we see that in all three cases the experimental result is visually very close to the ground truth case, as a large part of the visual information has been recovered. This is also confirmed by the experimental histograms, which in all cases, their form is very close to that of the ground truth image. Nevertheless, we must comment that for 3 convolutional modules, a slightly better result is obtained, as more correct color information has been recovered, and the histogram is almost the same as the corresponding ground truth.

Darkness Level: 3.5

Original Low Light **Normal Light**

1 module **3 modules** **5 modules**

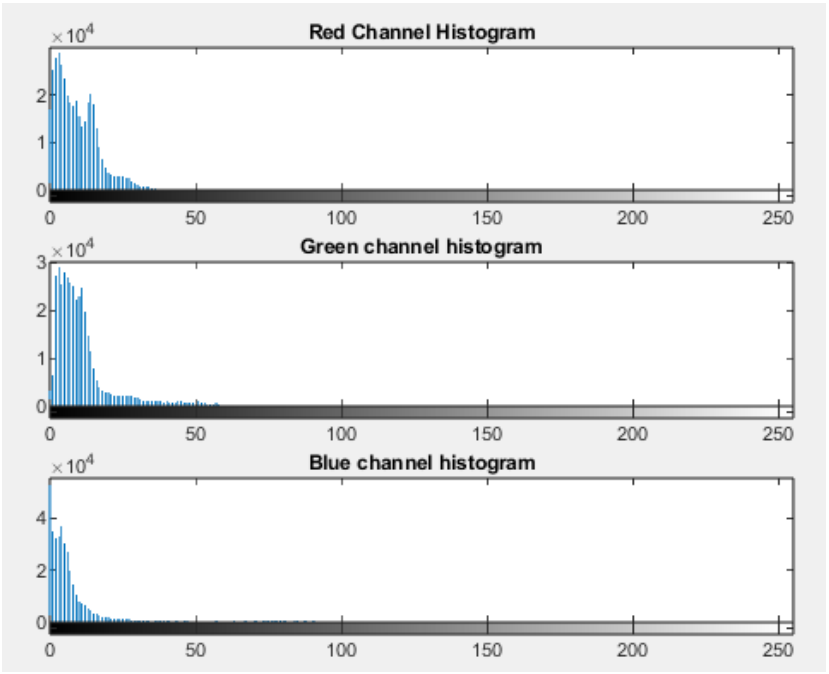


Figure 3.18: Histogram of 3.5 LL image

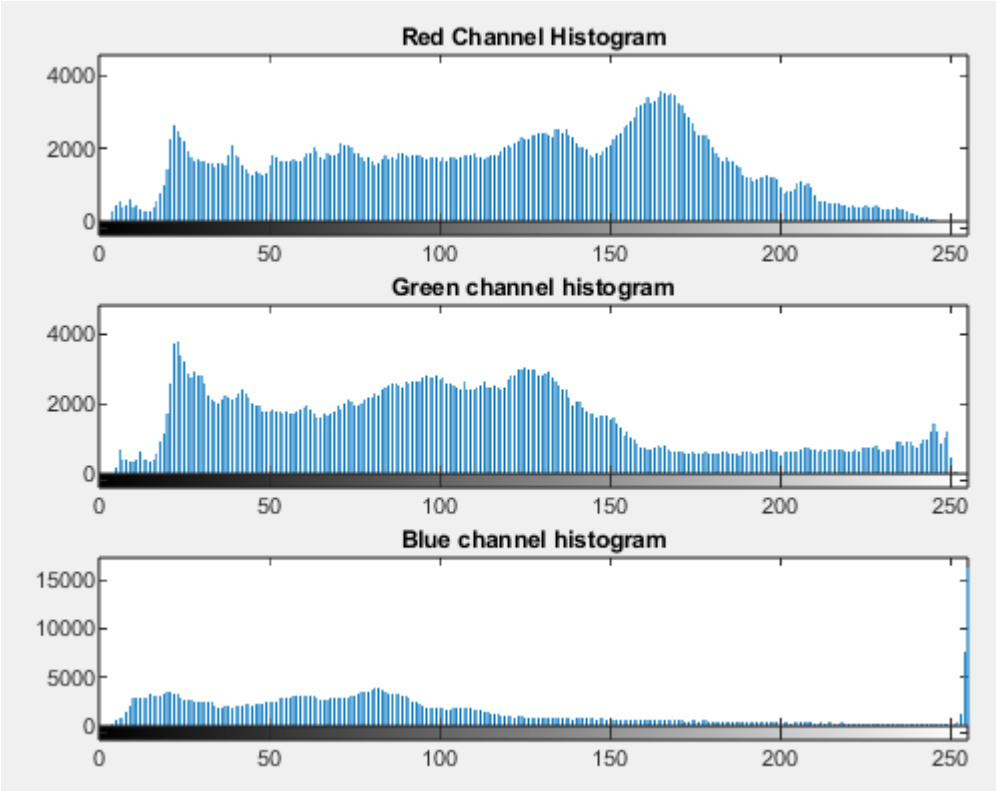


Figure 3.19: Histogram of 3.5 NL image

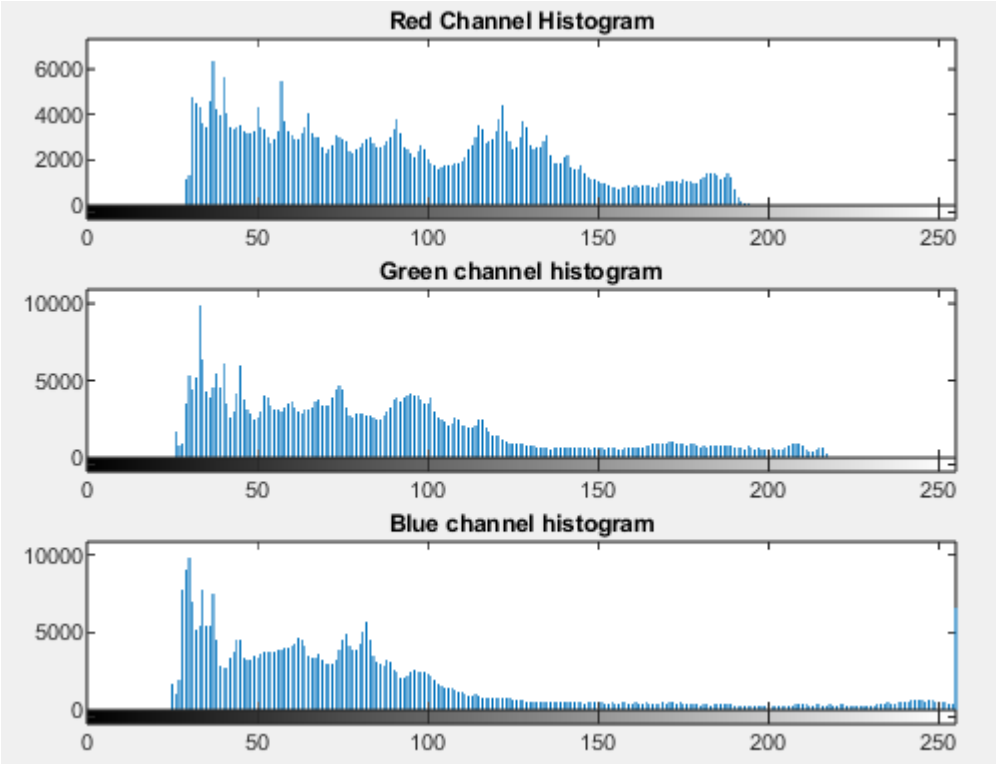


Figure 3.20: Histogram of 3.5 experimental result with 1 module

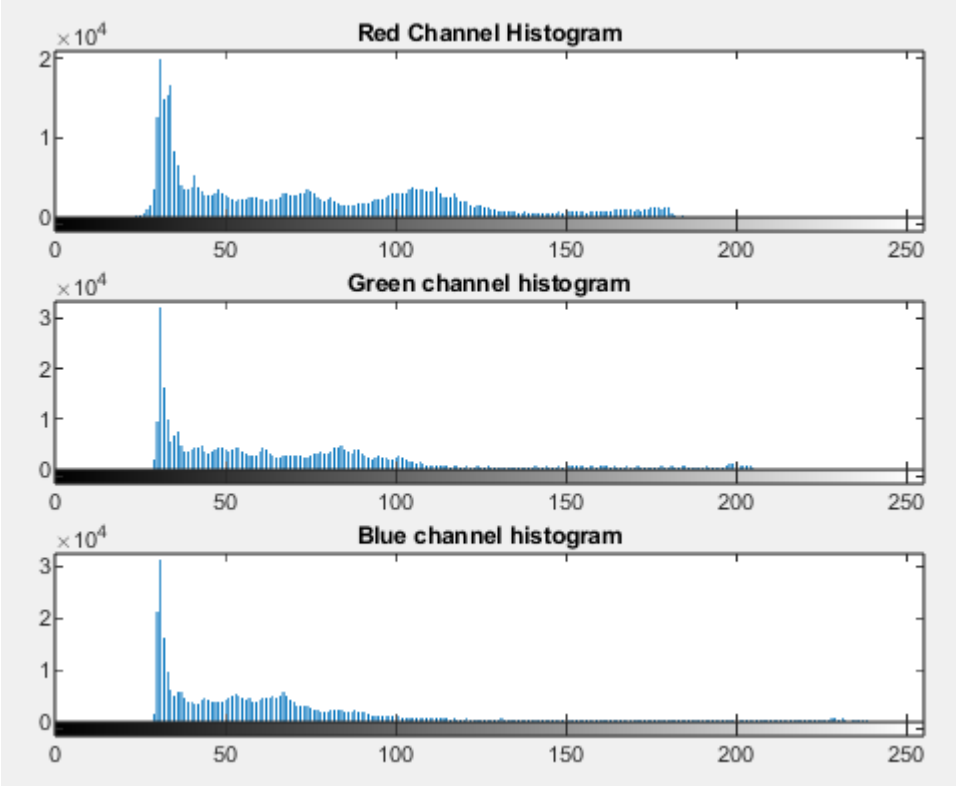


Figure 3.21: Histogram of 3.5 experimental result with 3 modules

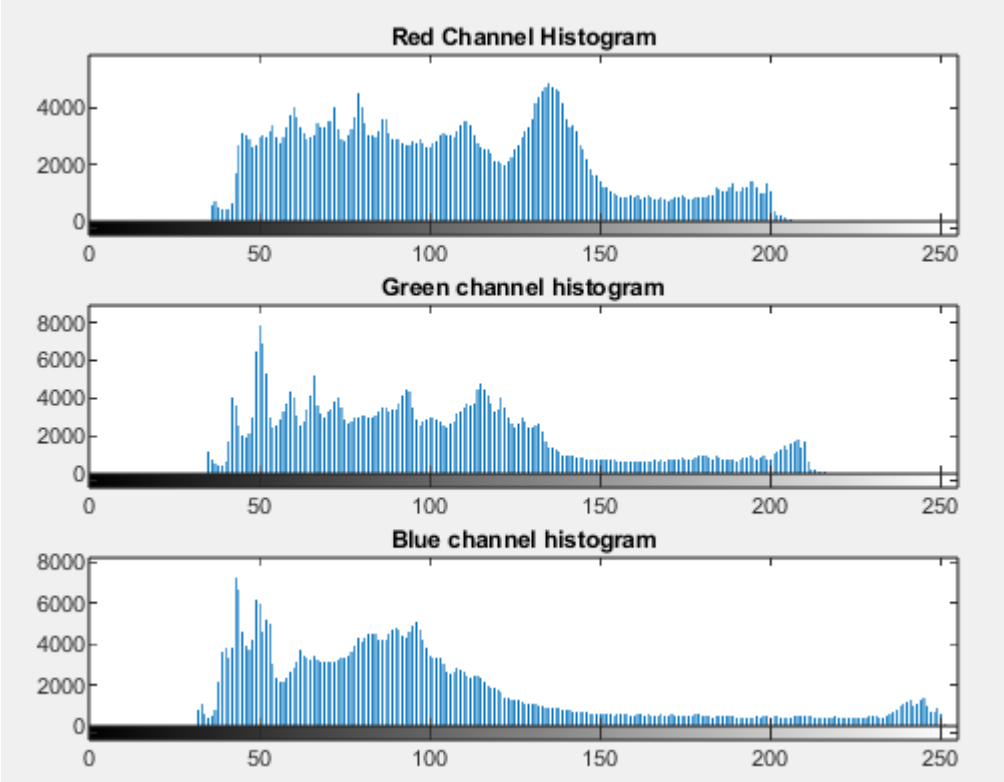
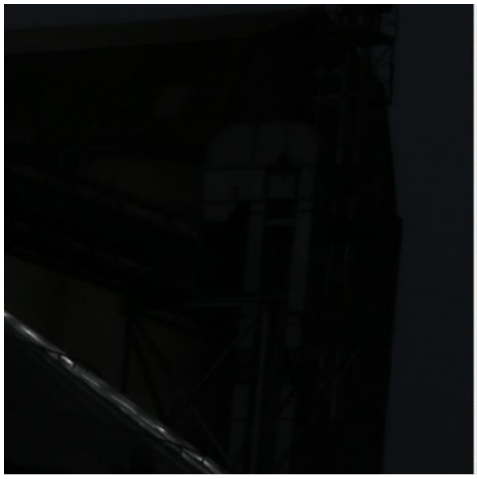


Figure 3.22: Histogram of 3.5 experimental result with 5 modules

For darkness level 3.5 impressive results are obtained as well, with the LL image being fully enhanced and most of the visual information recovered, for all cases of convolutional modules. This is also confirmed by the experimental histograms, where in all cases their form is very close to that of the ground truth image. Moreover, here too the 3 convolutional modules seem to produce a slightly better result, indicating what we mentioned above.

Darkness Level: 4.0



Original Low Light

Normal Light

1 module

3 modules

5 modules

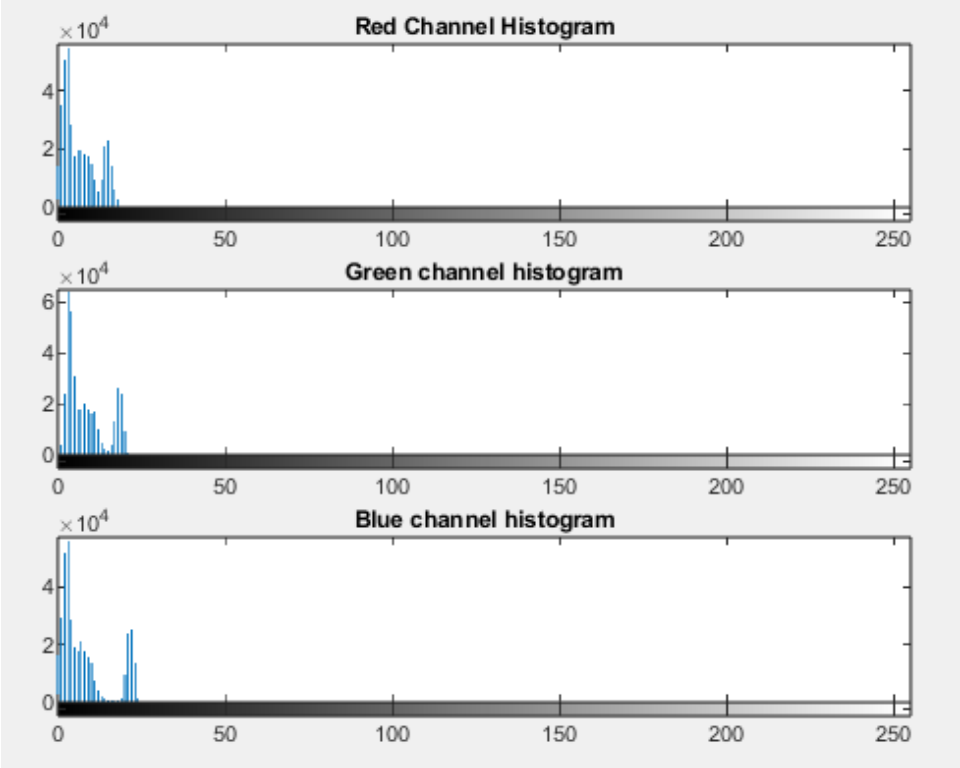


Figure 3.23: Histogram of 4.0 LL image

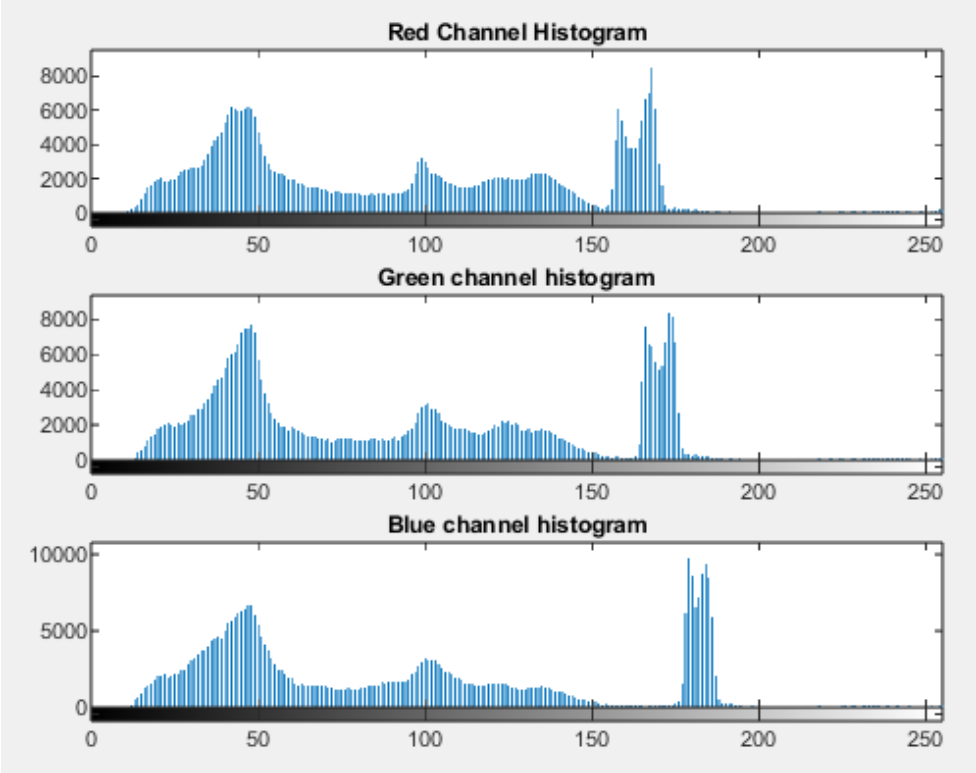


Figure 3.24: Histogram of 4.0 NL image

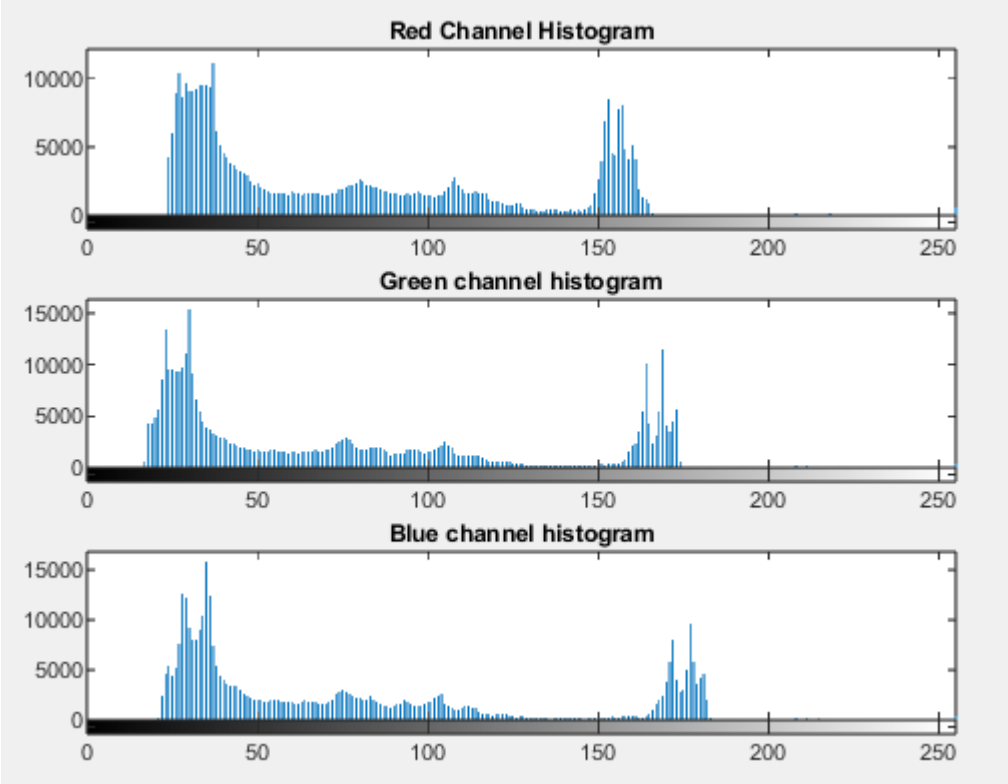


Figure 3.25: Histogram of 4.0 experimental result with 1 module

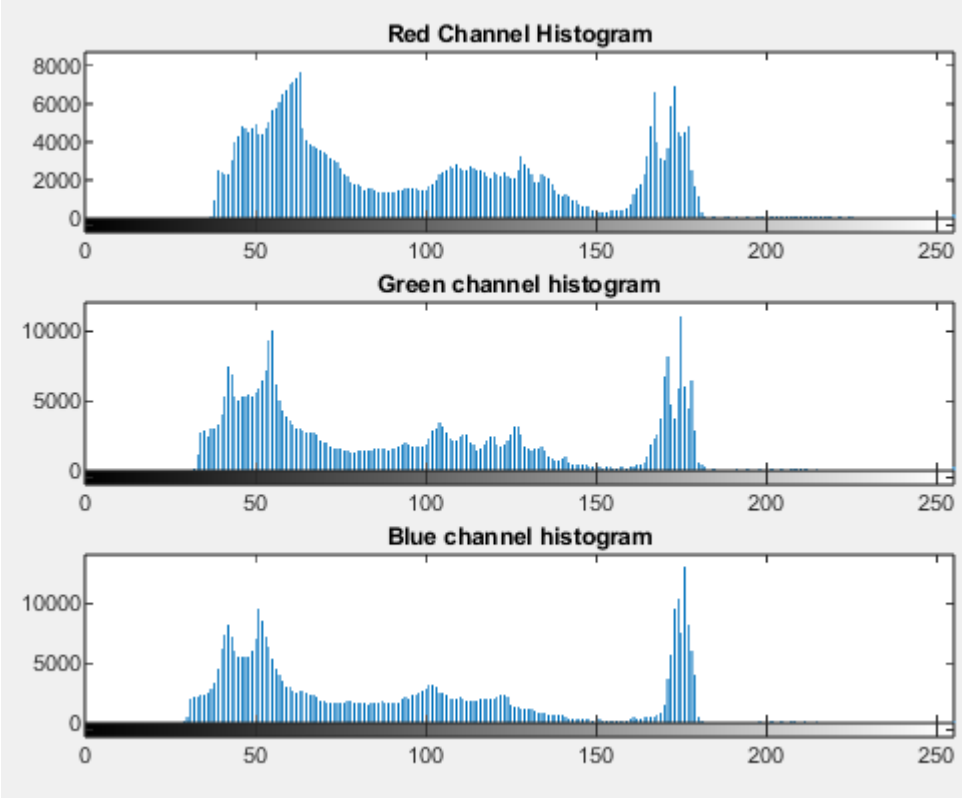


Figure 3.26: Hstogram of 4.0 experimental result with 3 modules

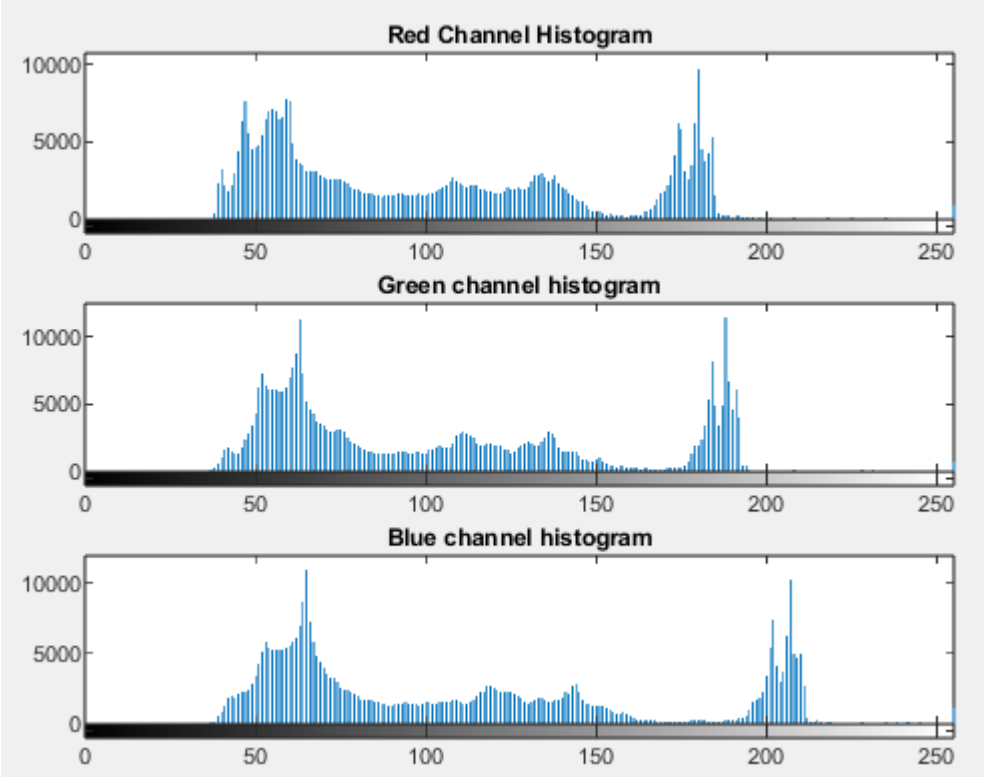


Figure 3.27: Histogram of 4.0 experimental result with 5 modules

For darkness level 4.0 the first effects of increasing the darkness level start to be seen, as in the experimental results the general texture of the image has been recovered but not the color information. The experimental histograms have a shape very close to the corresponding ground truth case, confirming the enhancement of the dark image. However, the LL image has started to become too dark, with the dynamic range being too small, making the retrieval of color information difficult, leading to the results we see.

Darkness Level: 4.5

Original Low Light **Normal Light**

1 module **3 modules** **5 modules**

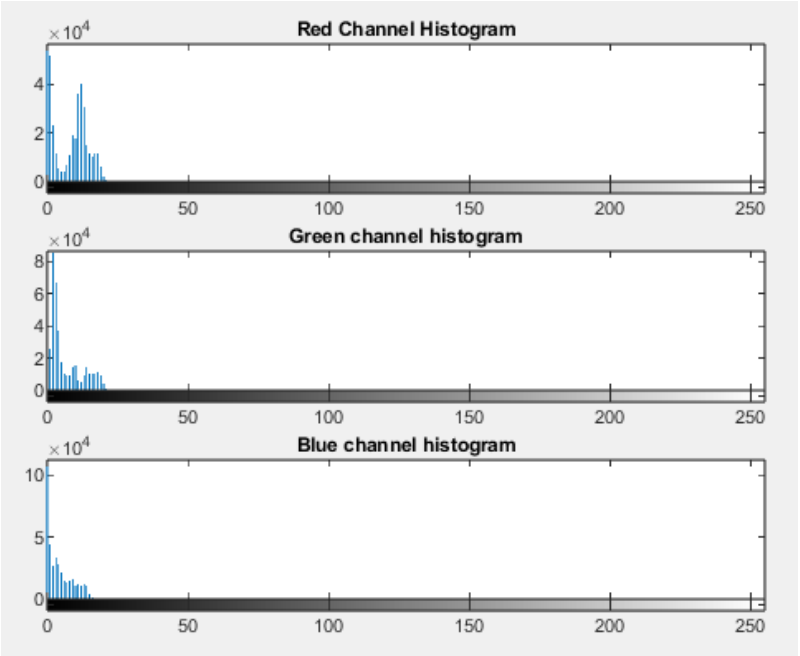


Figure 3.28: Histogram of 4.5 LL image

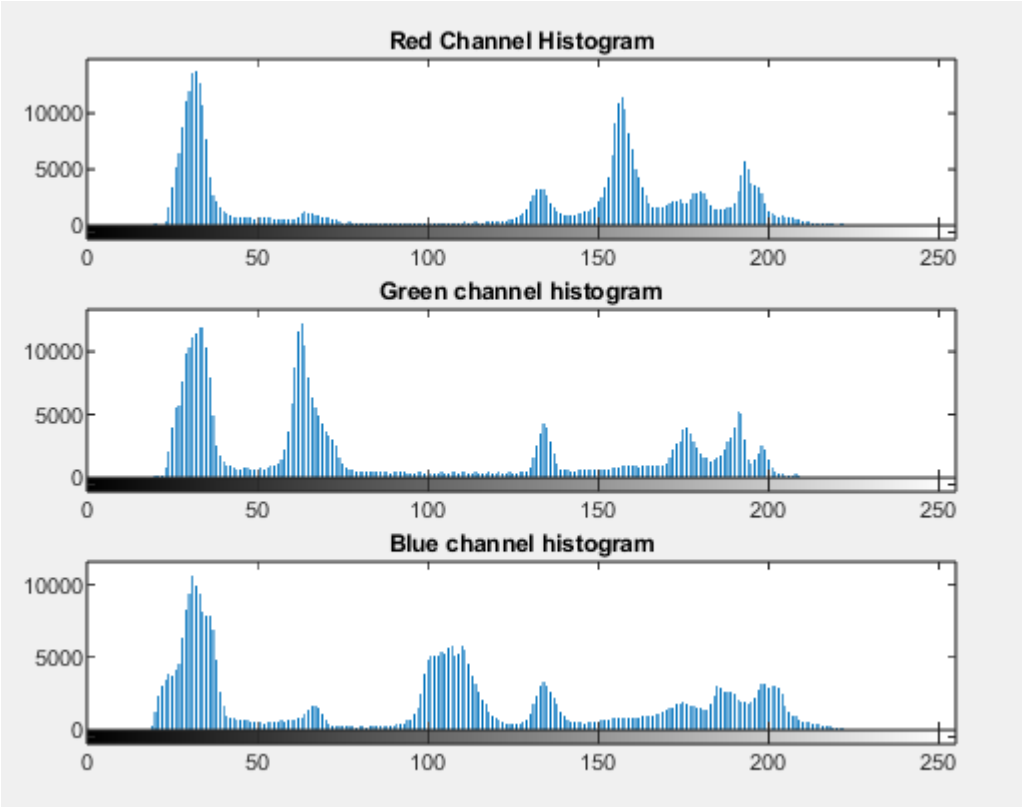


Figure 3.29: Histogram of 4.5 NL image

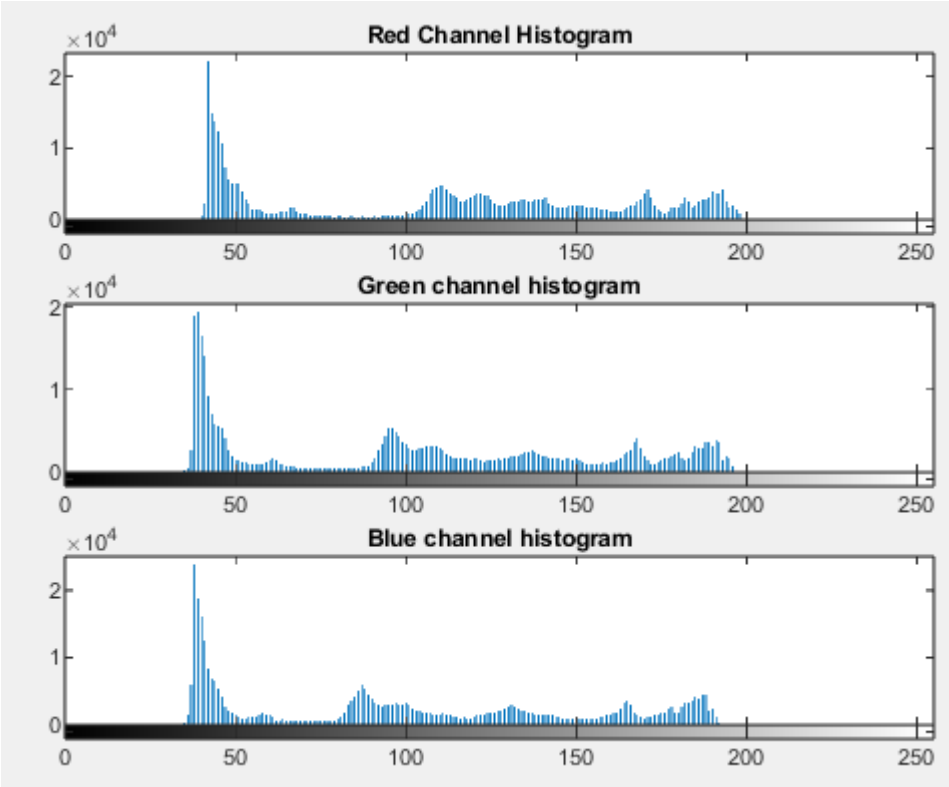


Figure 3.30: Histogram of 4.5 experimental result with 1 module

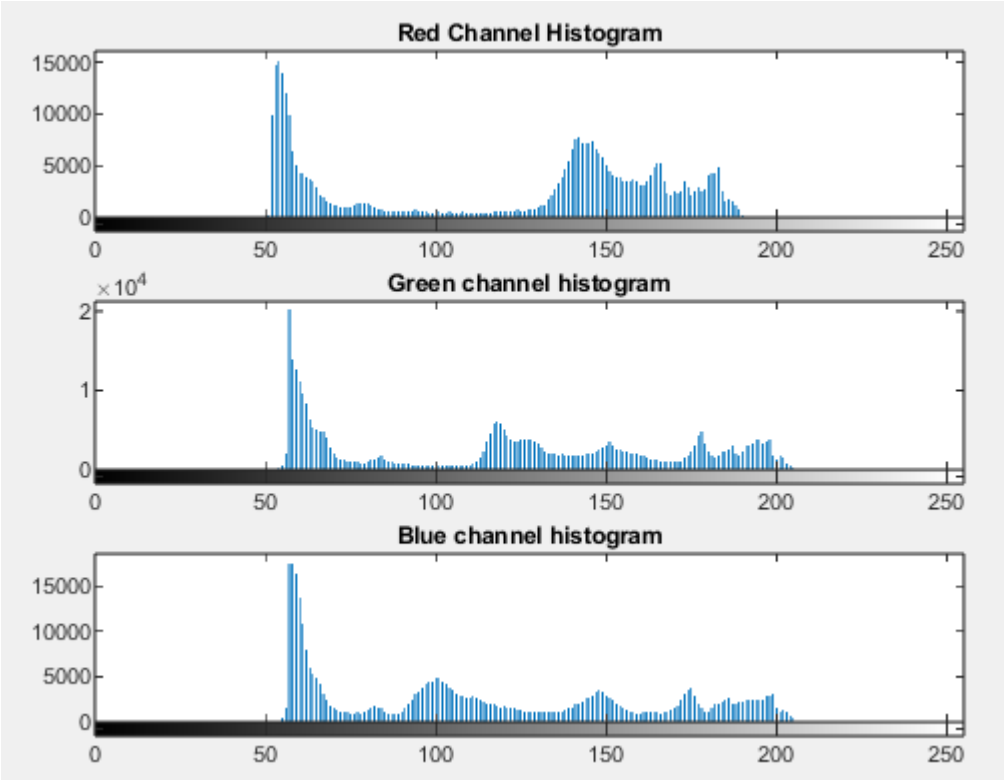


Figure 3.31: Histogram of 4.5 experimental result with 3 modules

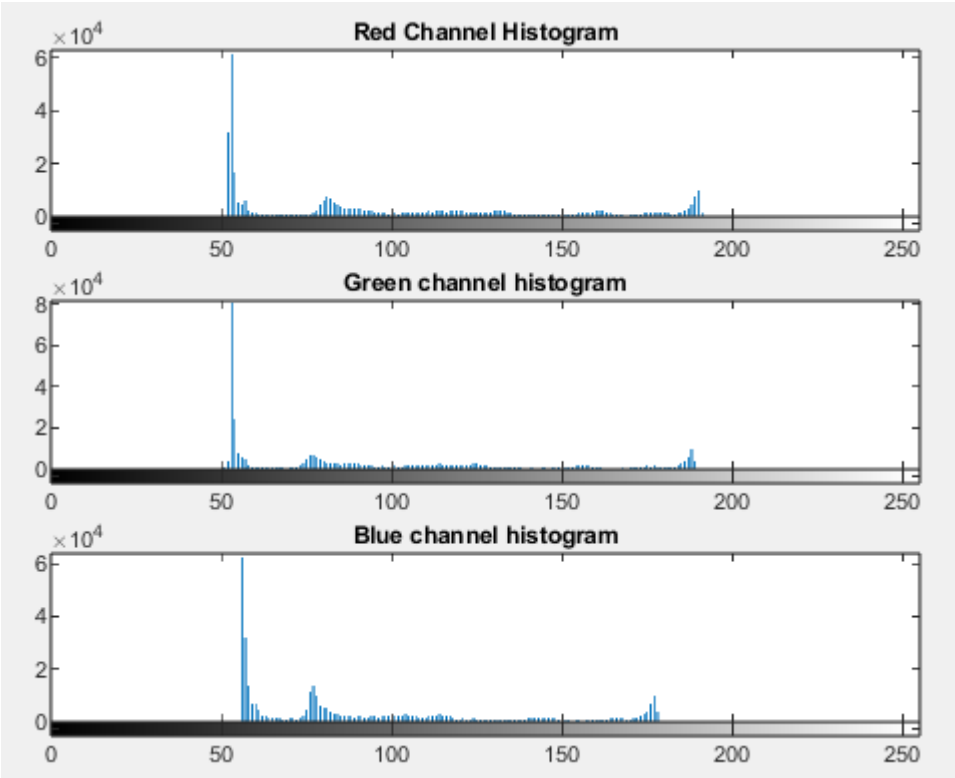


Figure 3.32: Histogram of 4.5 experimental result with 5 modules

In the case of darkness level 4.5 we again notice that, while the general texture of the image has been recovered and the LL image has been enhanced, the color information has not been restored. Here again the lack of color is because the images are too dark making it difficult to retrieve the color information. A solution to this problem would be to train the model more for large darkness levels so that it has time to learn the information needed to retrieve the color.

Darkness Level: 5.0

Original Low Light **Normal Light**

1 module **3 modules** **5 modules**

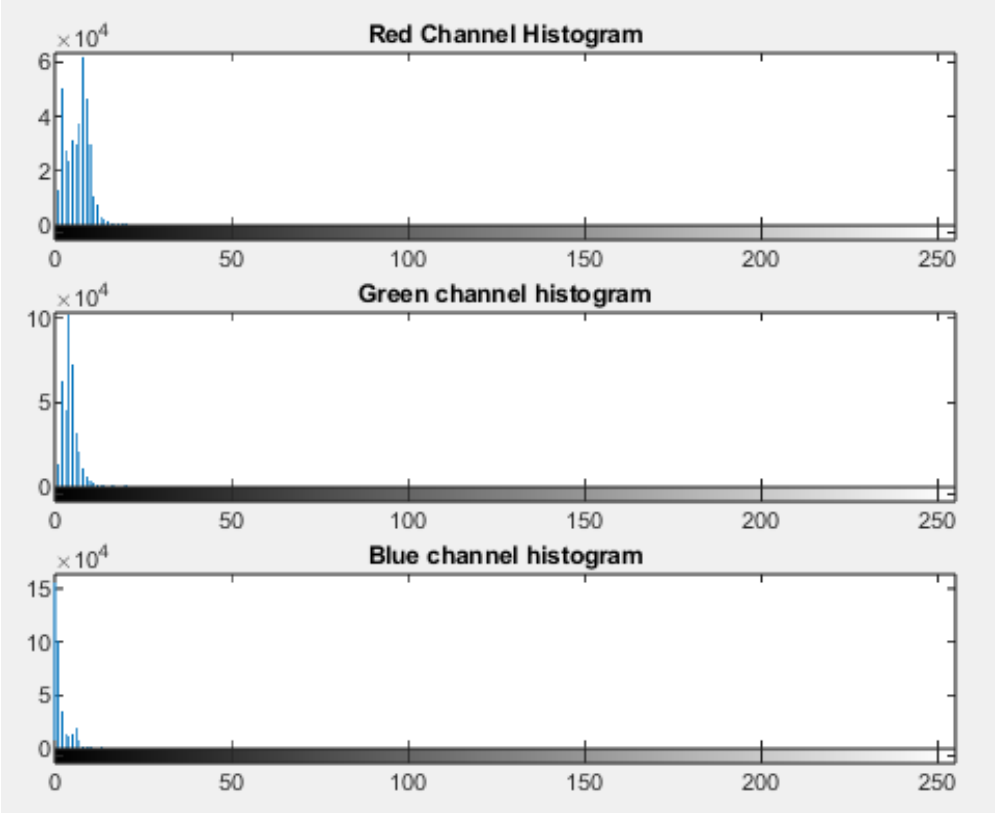


Figure 3.33: Histogram of 5.0 LL image

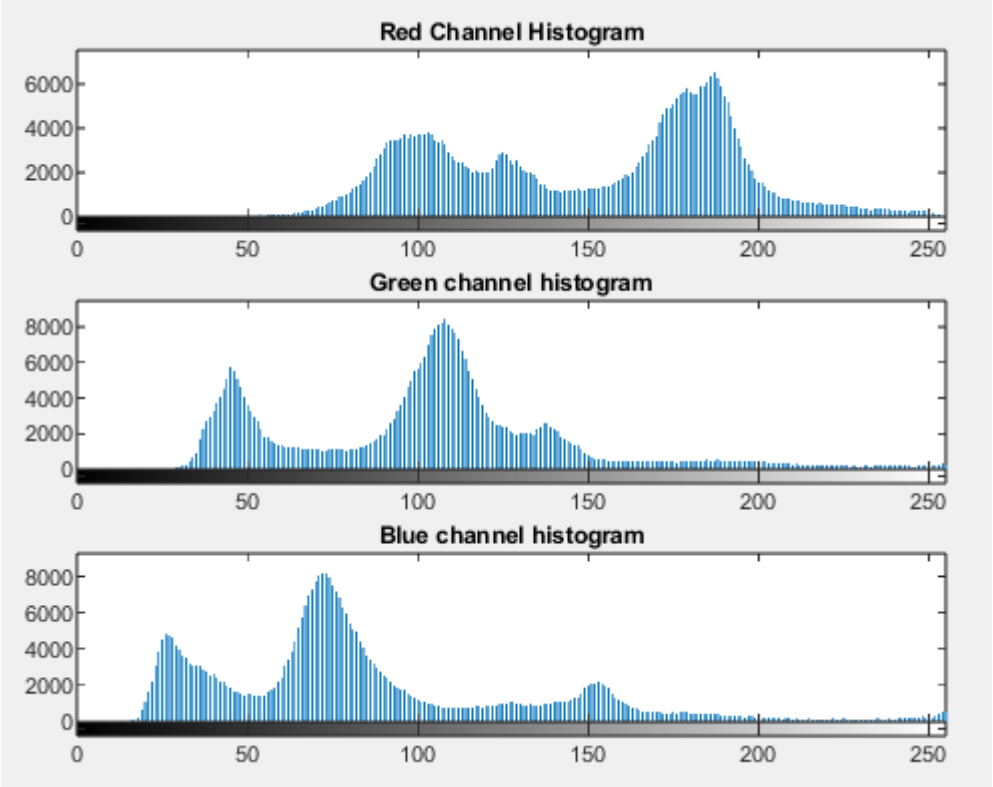


Figure 3.34: Histogram of 5.0 NL image

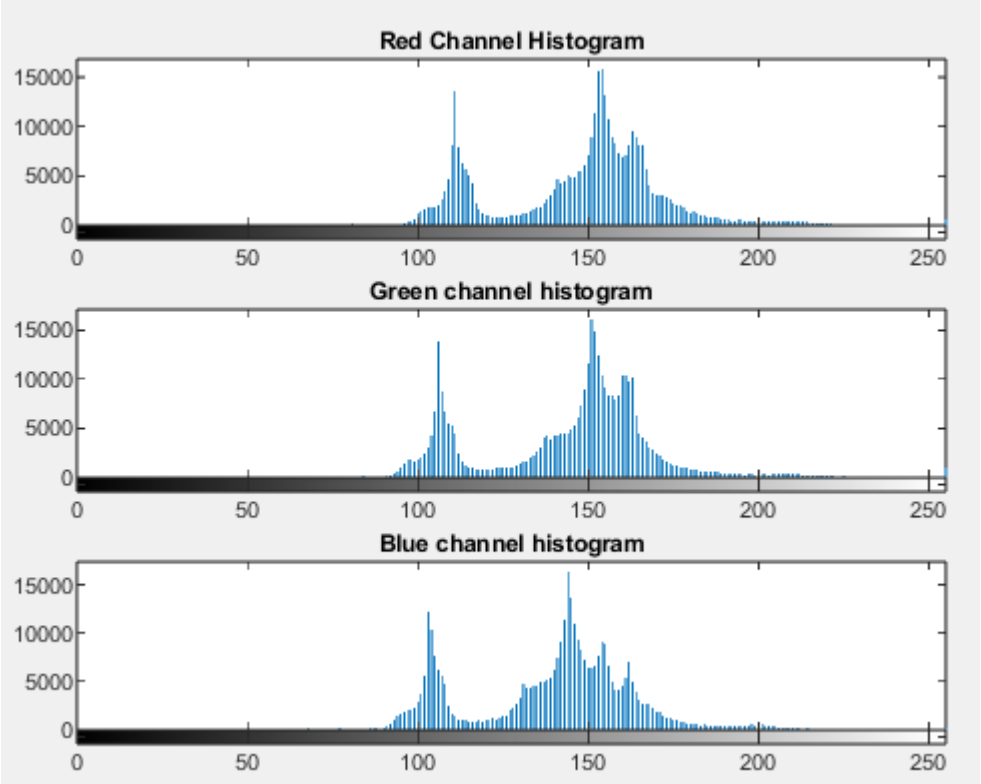


Figure 3.35: Histogram of 5.0 experimental result with 1 module

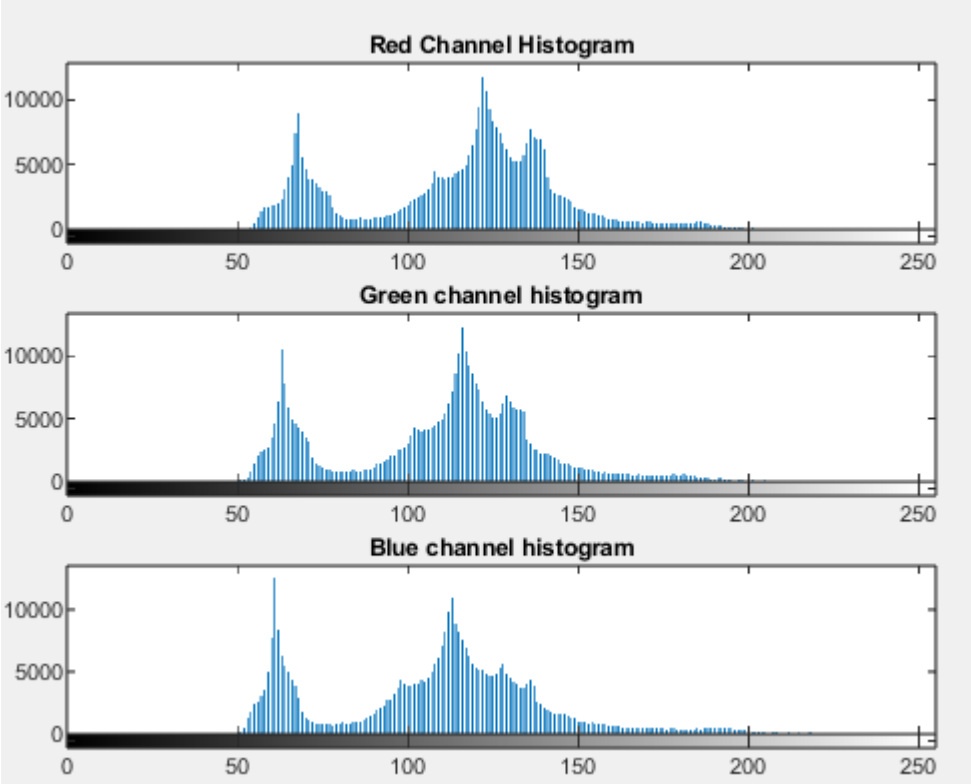


Figure 3.36: Histogram of 5.0 experimental result with 3 modules

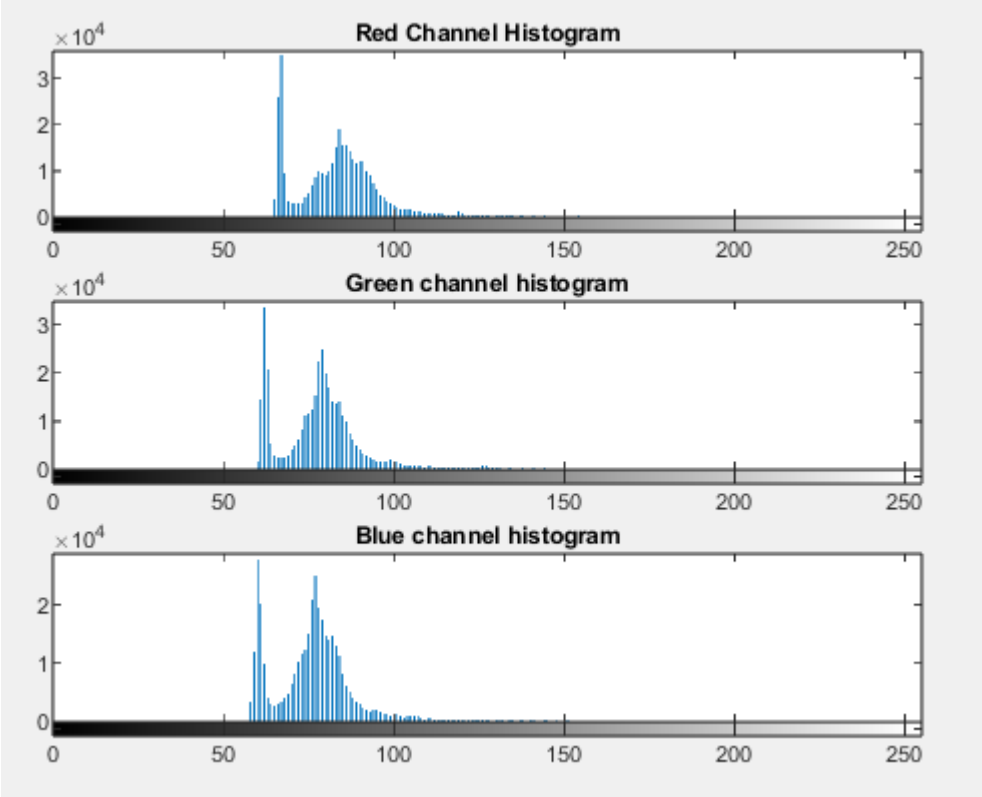


Figure 3.37: Histogram of 5.0 experimental result with 5 modules

For darkness level 5.0 we also notice that the color information is not restored, but the model has managed to recover the general texture of the image, as well as the details, without introducing additional noise and artifacts, which is particularly impressive. The experimental histograms, in all cases, have a form very close to the corresponding ground truth, with the histogram for 3 convolutional modules being the best. As we mentioned before, the recovery of the color information could also be recovered by further training the model.

Overall, we saw that the architecture we built gave very impressive results, greatly improving the values of the metrics, with the average PSNR increasing up to 11.71dB and the SSIM by 0.5 (values for the case with 3 convolutional modules that gave the best results). The improvement in the quality of the images was also confirmed by the visual results we presented, with the LL images being fully enhanced without the introduction of additional noise or artifacts (which was strongly observed in the case of classical methods). In addition, it was observed that with the increase of the darkness level it becomes more difficult to retrieve the color information, since the resulting images are black and white. This is due to the fact that for large darkness levels the images are too dark, with the

dynamic range too small, making it more difficult to learn the full mapping that will lead to color restoration. The fact that the general texture of the image is recovered means that if we gave more time (and data) to the model it could also recover the color information. So, a solution to this is the further training of the model for higher darkness levels. Finally, from the three cases of convolutional modules, we saw that in the given training conditions the model with 3 convolutional modules performs better, as it leads to the greatest improvement of the metrics and performs better in the large darkness levels, compared to the other two cases.

3.3 LL-CNN: 1st Variation

In the previous section we saw that the architecture we designed produces very impressive results, but struggles to recover color information for higher darkness levels, resulting in black and white images. One solution we mentioned is to further train the model for large darkness levels to recover the color as well. Due to limited computing resources, another way to deal with it would be to make the model learn more low level features that might help to recover the color as well. For this reason, in the architecture we presented earlier, we add an additional 3 convolutional layers, between the convolutional modules and the output layer. The implementation details of these layers are presented in appendix B. These additional convolutional layers act both as post processing steps and for learning additional features that can help in the recovery of color information as well. We repeat the experiments with the new form of the architecture, and present the results on the test set since we are interested in the generalizability of the model.

1 module

MSE					
Level	3.0	3.5	4.0	4.5	5.0
MIN	96,80228	65,58468	72,6814	97,79351	528,1886
MAX	4167,11	4159,735	4878,609	4774,116	5669,801
AVERAGE	594,7516	682,6434	806,4392	867,1724	1371,427
	PSNR(dB)				
Level	3.0	3.5	4.0	4.5	5.0
MIN	11,93245	11,94015	11,24784	11,34187	10,59513
MAX	28,27195	29,96278	29,51657	28,2277	20,90291
AVERAGE	21,98932	21,37298	20,34762	20,02487	17,36206
	SSIM				
Level	3.0	3.5	4.0	4.5	5.0
MIN	0,255508	0,251793	-0,16438	-0,08179	-0,18834
MAX	0,915518	0,914636	0,859552	0,767284	0,763961
AVERAGE	0,729288	0,696488	0,452581	0,452772	0,34826
	MV				
Level	3.0	3.5	4.0	4.5	5.0
MIN	47,72041	32,19816	46,2511	45,11351	71,33703
MAX	131,8304	117,613	129,0201	129,6187	143,1904
AVERAGE	102,9921	85,37628	97,92211	97,94818	115,8154
	STD				
Level	3.0	3.5	4.0	4.5	5.0
MIN	89,75246	50,56022	56,36618	46,75957	68,2928
MAX	1035,856	869,4787	947,505	887,7119	1010,549
AVERAGE	358,8593	336,6523	329,3787	304,9104	387,3231
	BRISQUE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	35,85155	35,84404	37,22929	40,01978	43,27264
MAX	51,24883	54,05877	54,65608	58,02733	56,17784
AVERAGE	43,95986	46,22144	46,00282	49,00711	49,93145
	NIQE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	3,486145	3,655138	3,980192	3,950654	4,08138
MAX	7,709424	6,987334	6,541648	5,749939	5,430819
AVERAGE	4,581822	4,635425	4,528655	4,460164	4,511685

Table 3.4: LLCNN 1st variation results on test set – 1 CNN module

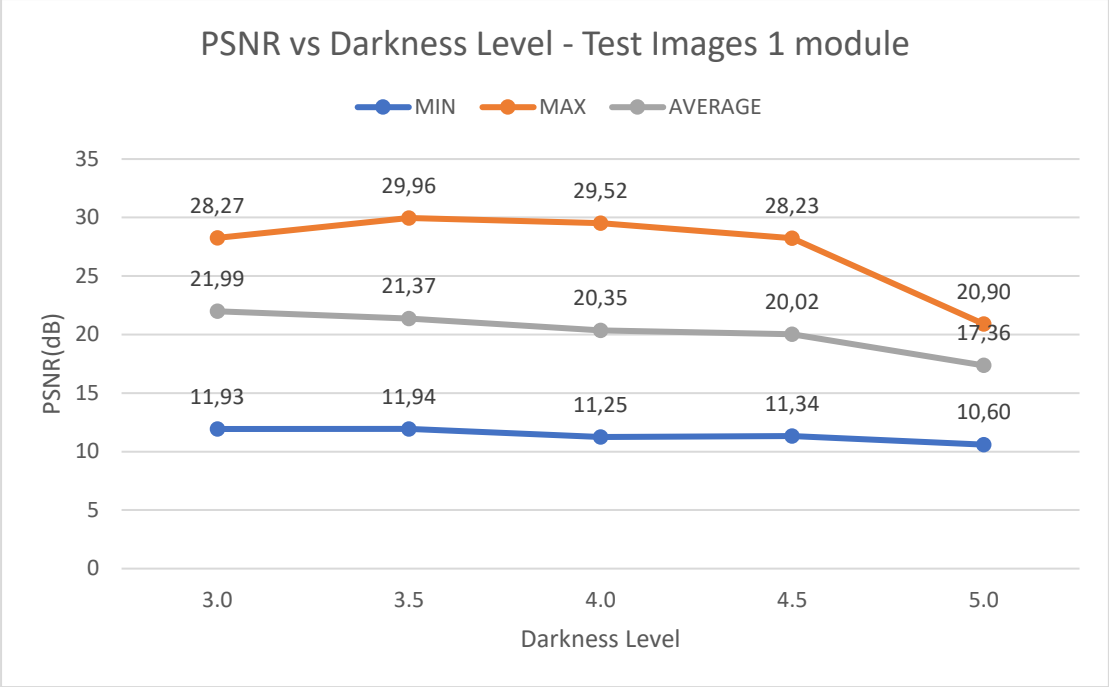
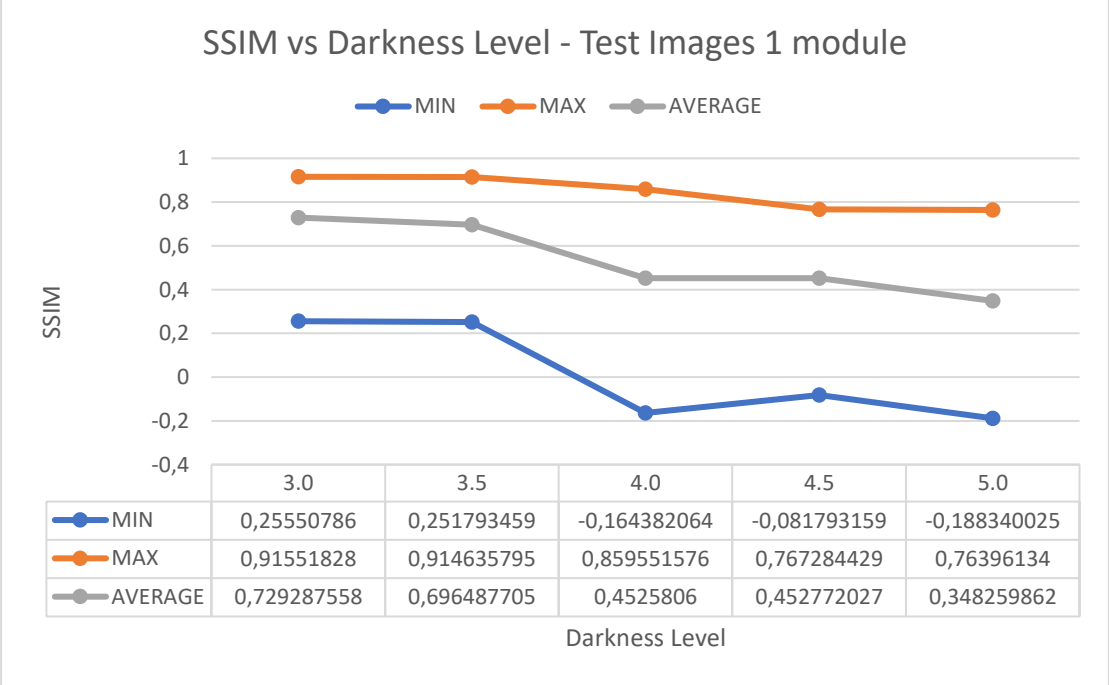


Figure 3.38: Experimental results PSNR vs Darkness level for test set



	3.0	3.5	4.0	4.5	5.0
MIN	0,25550786	0,251793459	-0,164382064	-0,081793159	-0,188340025
MAX	0,91551828	0,914635795	0,859551576	0,767284429	0,76396134
AVERAGE	0,729287558	0,696487705	0,4525806	0,452772027	0,348259862

Figure 3.39: Experimental results SSIM vs Darkness level for test set

For 1 convolutional module the values of the metrics improve noticeably, as the MSE decreases and the PSNR and SSIM increase, compared to the original LL images. In addition, MV and STD increase, which means that the experimental images are brighter, since they have a higher average

brightness, and have a higher contrast as the pixel values are spread over a larger range around the average brightness. BRISQUE and NIQE continue to have large values, which shows that with DL models we cannot recover the natural statistics, something we also found in the previous section. Finally, as the darkness level increases, the expected behavior is observed, as the quality of the metrics decreases. Specifically, MSE increases and PSNR and SSIM decrease with increasing darkness level, and similarly MV and STD also decrease with increasing darkness level. As we mentioned above, this could be solved by further training the model for large darkness levels.

3 modules

MSE					
Level	3.0	3.5	4.0	4.5	5.0
MIN	139,8357	55,47963	62,18725	107,7509	182,9754
MAX	4673,288	3123,43	3793,452	3812,375	4110,048
AVERAGE	634,6368	554,7044	603,7259	657,5905	887,8519
	PSNR(dB)				
Level	3.0	3.5	4.0	4.5	5.0
MIN	11,43458	13,18449	12,34046	12,31885	11,99233
MAX	26,67462	30,68947	30,19379	27,80659	25,50688
AVERAGE	21,50457	22,46352	21,94701	21,39252	19,6565
	SSIM				
Level	3.0	3.5	4.0	4.5	5.0
MIN	0,177864	0,065313	0,312374	0,140107	-0,05215
MAX	0,934763	0,943638	0,856044	0,911381	0,709297
AVERAGE	0,715544	0,685708	0,676907	0,623833	0,435932
	MV				
Level	3.0	3.5	4.0	4.5	5.0
MIN	51,56754	43,73285	51,93238	46,87704	51,93238
MAX	135,8206	123,6079	132,1517	129,5823	132,1517
AVERAGE	105,6586	94,97591	99,04177	95,81691	99,04177
	STD				
Level	3.0	3.5	4.0	4.5	5.0
MIN	119,3738	90,00752	35,43465	62,67123	60,01391
MAX	1032,159	916,947	913,3763	806,6738	881,5942
AVERAGE	375,6467	313,5944	300,1116	245,5469	271,1893
	BRISQUE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	38,12286	36,39868	40,92492	37,49894	43,4405
MAX	53,93185	51,19811	53,97783	54,31783	57,36054
AVERAGE	45,39954	44,68683	47,82419	46,59224	49,7923

	NIQE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	4,252357	4,16975	4,362417	4,478127	4,424179
MAX	8,018875	8,321446	7,286521	6,786895	6,167481
AVERAGE	5,192974	5,066215	5,127456	5,119233	5,233816

Table 3.5: LLCNN 1st variation results on test set – 3 CNN modules

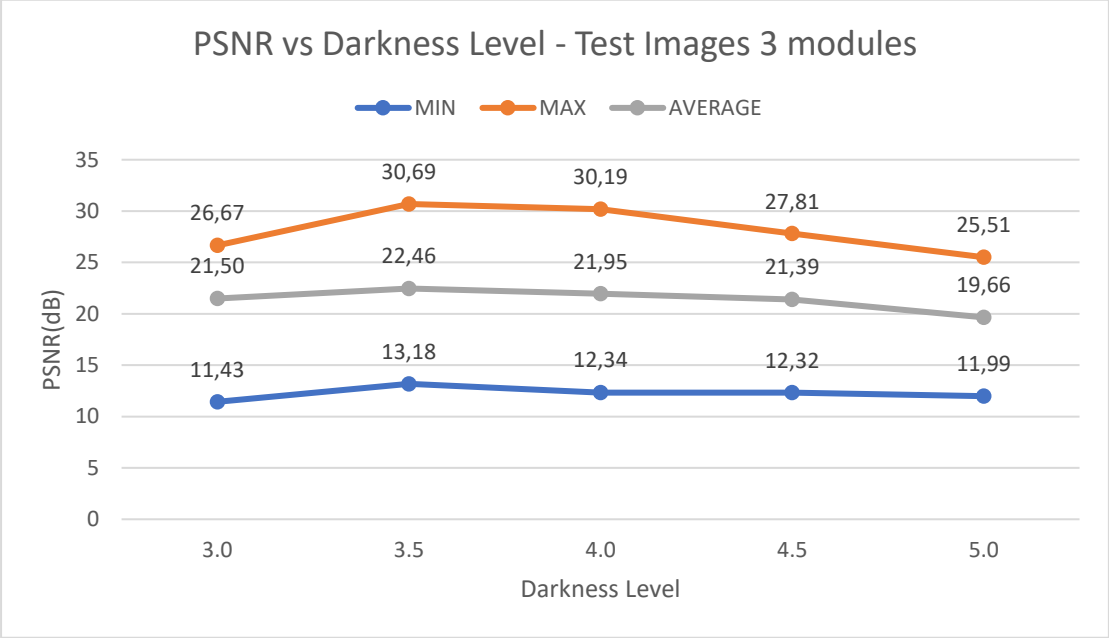


Figure 3.40: Experimental results PSNR vs Darkness level for test set

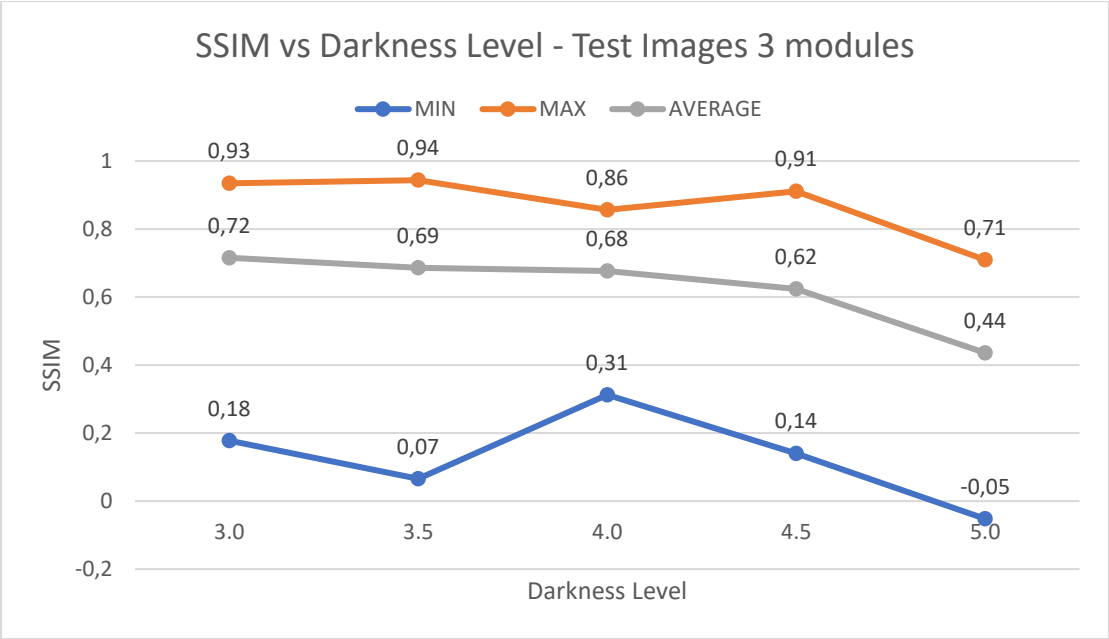


Figure 3.41: Experimental results SSIM vs Darkness level for test set

For 3 convolutional modules again, we can observe an improvement of the metrics, as the MSE decreases and the PSNR and SSIM increase, which means that the experimental images come closer visually to the ground truth case. Moreover, MV and STD are increased, compared to the original LL images, which means that the experimental images are brighter and with higher contrast. Despite this, BRISQUE and NIQE continue to have large values, something we found and explained in the previous section. Finally, the expected behavior is again observed with the increase of the darkness level, as the performance of the model decreases.

5 modules

MSE					
Level	3.0	3.5	4.0	4.5	5.0
MIN	58,64497	72,65532	155,9949	97,63911	1239,511
MAX	3707,553	3119,224	3288,394	5027,542	6936,387
AVERAGE	623,8515	573,6459	672,0668	1140,519	3163,393
	PSNR(dB)				
Level	3.0	3.5	4.0	4.5	5.0
MIN	12,43993	13,19034	12,96096	11,11725	9,71947
MAX	30,4485	29,51813	26,1997	28,23457	17,1983
AVERAGE	22,11363	22,20917	21,07984	18,66252	13,47252
	SSIM				
Level	3.0	3.5	4.0	4.5	5.0
MIN	0,195842	0,162419	0,166974	0,168237	-0,00482
MAX	0,920853	0,904978	0,828577	0,771529	0,609581
AVERAGE	0,740001	0,717316	0,568654	0,492206	0,263974
	MV				
Level	3.0	3.5	4.0	4.5	5.0
MIN	34,12043	48,12263	50,20952	38,88862	95,47895
MAX	117,0227	124,0975	125,1285	113,5511	102,2477
AVERAGE	85,91909	98,02158	96,81714	78,89687	96,95219
	STD				
Level	3.0	3.5	4.0	4.5	5.0
MIN	54,75704	64,092	53,02597	54,39942	108,8808
MAX	934,404	862,7025	867,199	684,8469	153,6646
AVERAGE	316,6737	300,166	269,842	212,4173	136,1414
	BRISQUE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	33,35861	35,78578	37,15804	34,86287	45,17703
MAX	54,26366	52,49631	50,64247	53,24163	50,80452
AVERAGE	46,78519	44,80433	44,45097	47,20074	47,83929

	NIQE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	4,269911	4,123655	4,634809	4,265357	7,041879
MAX	8,313357	8,024193	9,543192	7,866149	11,70887
AVERAGE	5,203072	5,236256	5,723349	5,007355	9,882166

Table 3.6: LLCNN 1st variation results on test set – 5 CNN modules

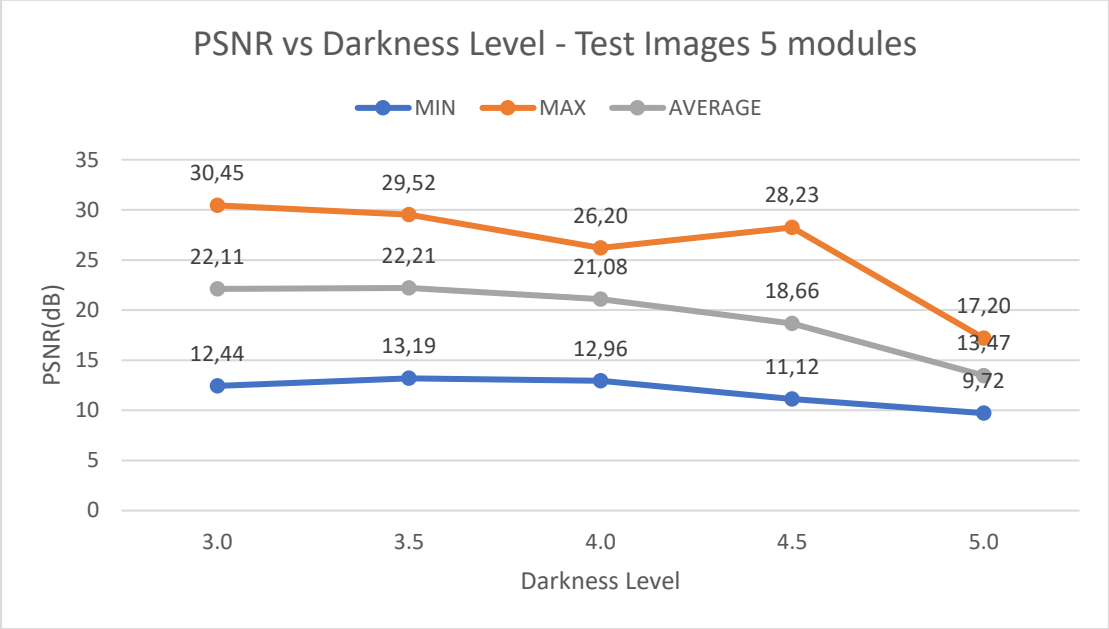


Figure 3.42: Experimental results PSNR vs Darkness level for test set

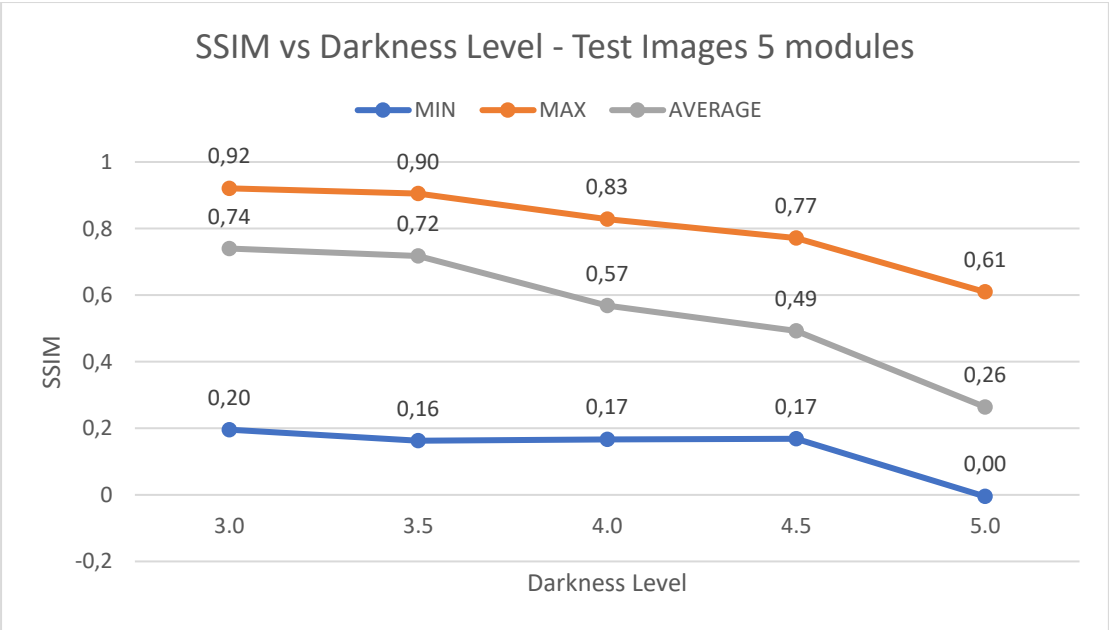
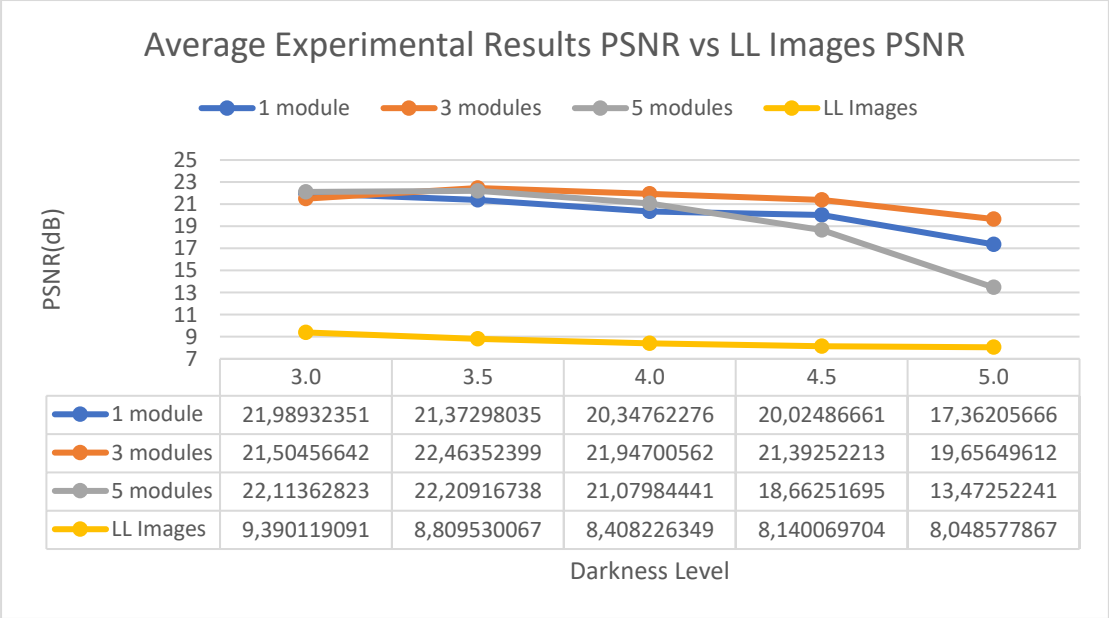


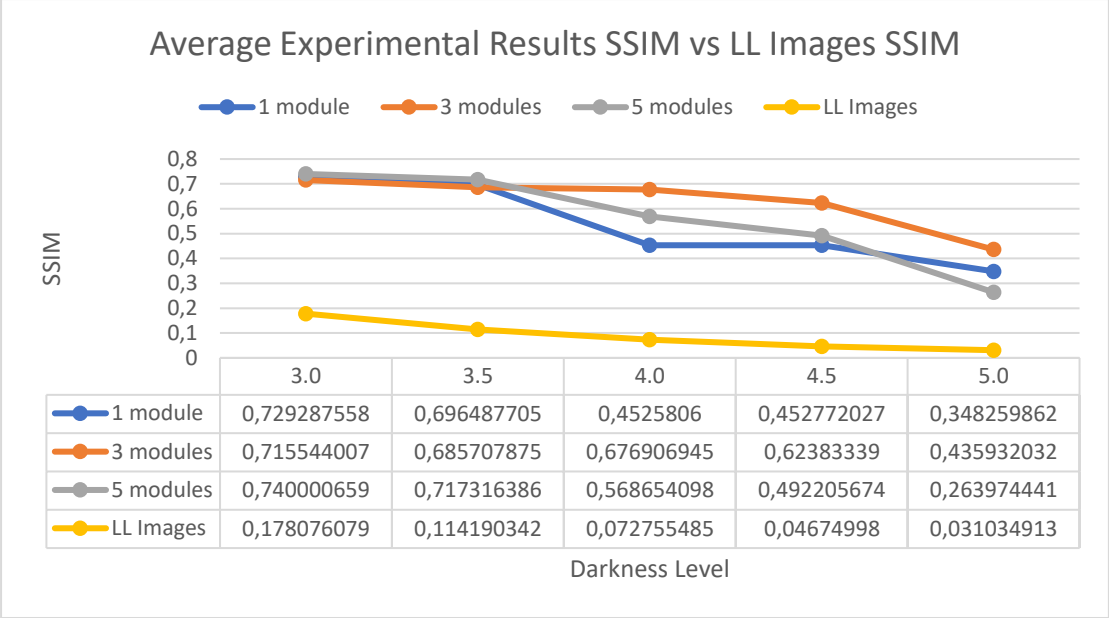
Figure 3.43: Experimental results SSIM vs Darkness level for test set

For all 5 convolutional modules we observe a clear improvement in the metric values as we see that the MSE decreases and the PSNR and SSIM increase, which indicates the improvement in the quality of the images. For quality metrics with no reference, MV increases, meaning the experimental images are brighter, and STD also increases, meaning the experimental results are also characterized by higher contrast. BRISQUE and NIQE continue to have large values, which is seen in all cases so far. Finally, with the increase of the darkness level, the expected behavior is observed, as the images become darker with the result that the model has difficulty learning the representation between LL space and NL space.



	3.0	3.5	4.0	4.5	5.0
1 module	21,98932351	21,37298035	20,34762276	20,02486661	17,36205666
3 modules	21,50456642	22,46352399	21,94700562	21,39252213	19,65649612
5 modules	22,11362823	22,20916738	21,07984441	18,66251695	13,47252241
LL Images	9,390119091	8,809530067	8,408226349	8,140069704	8,048577867

Figure 3.44: Experimental results average PSNR vs LL Images PSNR test set



	3.0	3.5	4.0	4.5	5.0
1 module	0,729287558	0,696487705	0,4525806	0,452772027	0,348259862
3 modules	0,715544007	0,685707875	0,676906945	0,62383339	0,435932032
5 modules	0,740000659	0,717316386	0,568654098	0,492205674	0,263974441
LL Images	0,178076079	0,114190342	0,072755485	0,04674998	0,031034913

Figure 3.45: Experimental results average SSIM vs LL Images SSIM test set

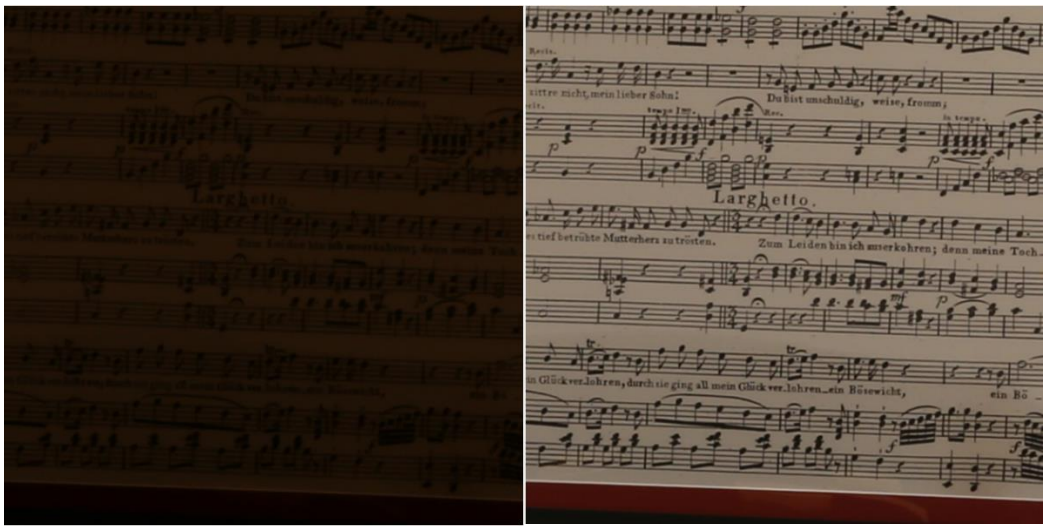
Overall, we observed that all three cases of convolutional modules significantly improve the values of the metrics, as the MSE decreases and the PSNR and SSIM increase, compared to the initial values for the LL images, which means that after applying the model, the images come visually closer to the ground truth case. For the quality metrics with no reference, we saw that MV and STD increase, i.e. the images become brighter, since their average brightness increases, and have higher contrast, since the pixel values are spread over a larger range around the average brightness. BRISQUE and NIQE continue to have large values even after applying the model, which means that with DL techniques we cannot recover the natural statistics, which is due to the continuous filtering that the images undergo as they pass through the model, such as also explained in the previous subsection. Finally, with the increase in the darkness level, a decrease in the performance of the model was observed, as expected. Increasing the darkness level means that the images become very dark, with the dynamic range being too small, making it difficult to learn the model, leading to a decrease in its performance.

Testing three different cases of number of convolutional modules we saw that 1 module led to an increase in average PSNR by 11.65dB and SSIM by 0.45, 3 convolutional modules increased PSNR by 12.83dB and SSIM by 0.54, and 5 convolutional modules increased the PSNR by 10.94dB and SSIM by 0.47, so based on these values it shows that the 3 convolutional modules perform better. From figures 3.44 and 3.45 we see that at darkness

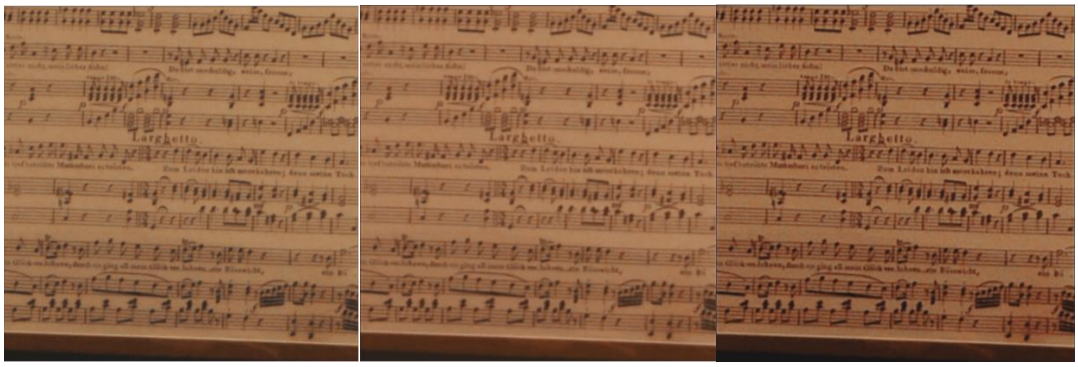
levels 3.0 and 3.5 all three cases of convolutional modules have almost the same performance, while at higher darkness levels there starts to be a clearer separation. In particular we see that the 3 convolutional modules perform better for large darkness levels, while the cases with 1 and 5 modules have decreasing performance. For the case of 1 module, the reduced performance is explained by the fact that we have fewer features compared to the other cases, thus less information for the approximation of the representation. On the contrary, for the case of 5 modules we have more features, and because we train all cases for the same number of epochs, the model does not have time to fully learn the representation, leading to decreasing performance. All this confirms that, for the specific training conditions, the 3 convolutional modules have the best performance.

All cases, however, produce an impressive improvement in quality metric values, which shows us that our architecture is characterized by very good generalization ability. To see the results visually, we will display, for each darkness level, a random image for each case of convolutional modules. In addition, we will display the corresponding ground truth and LL images, along with all the respective histograms, so that we can compare the results.

Darkness Level: 3.0



Original Low Light **Normal Light**



1 module **3 modules** **5 modules**

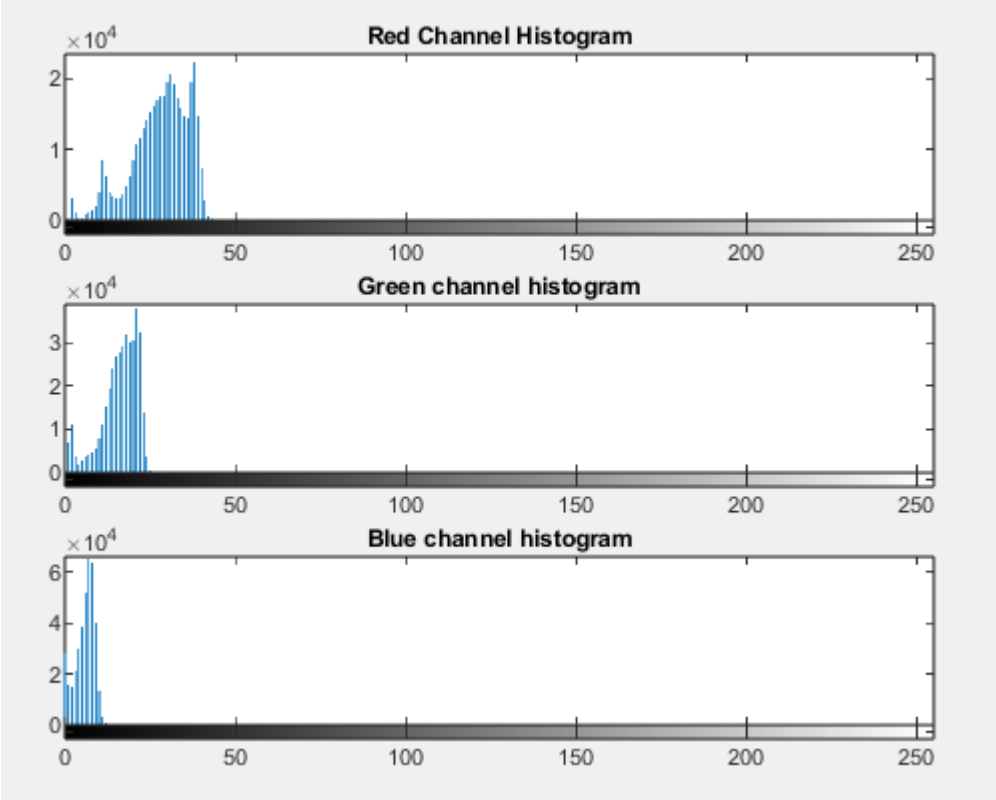


Figure 3.46: Histogram of 3.0 LL image

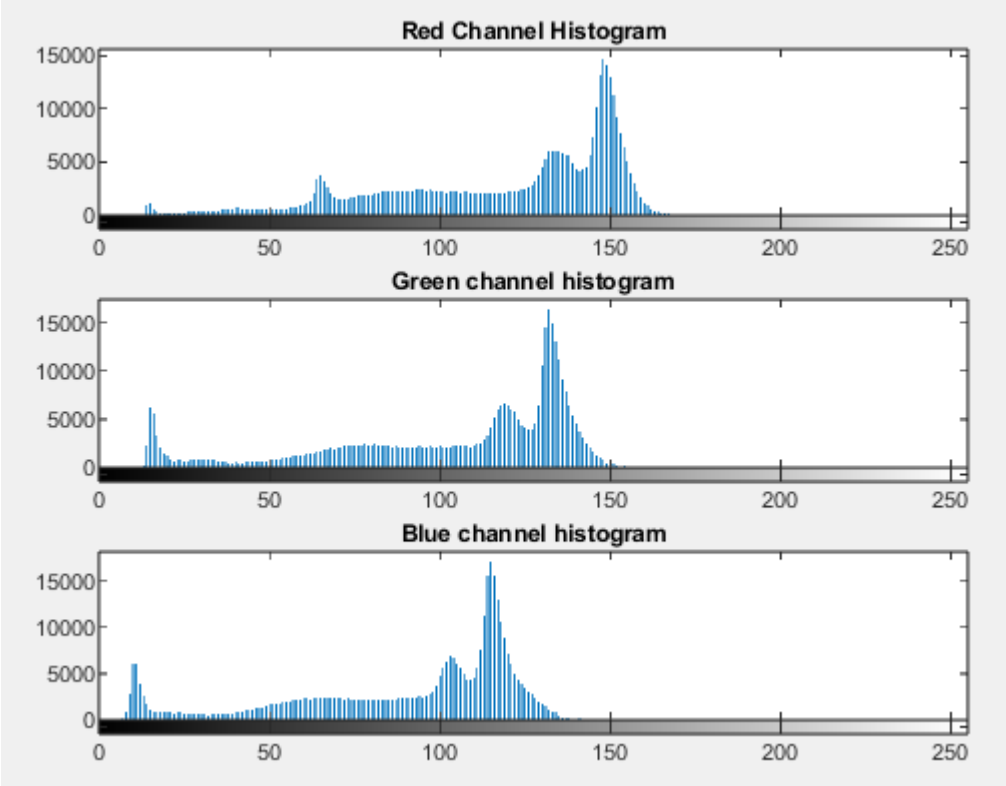


Figure 3.47: Histogram of 3.0 NL image

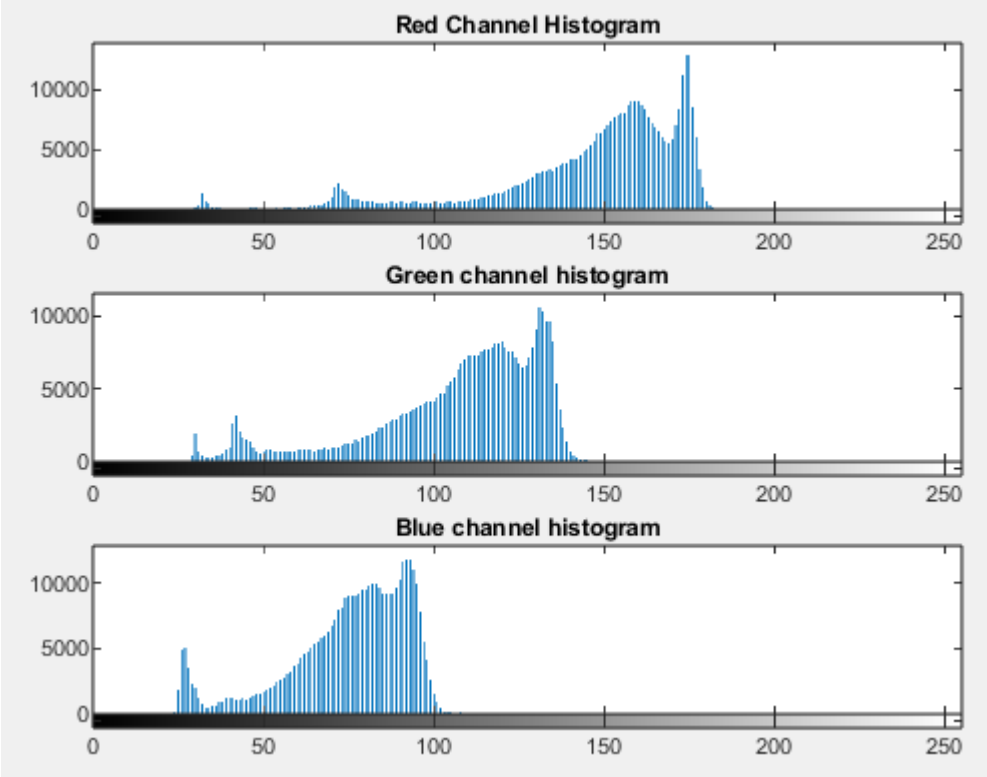


Figure 3.48: Histogram of 3.0 experimental result with 1 module

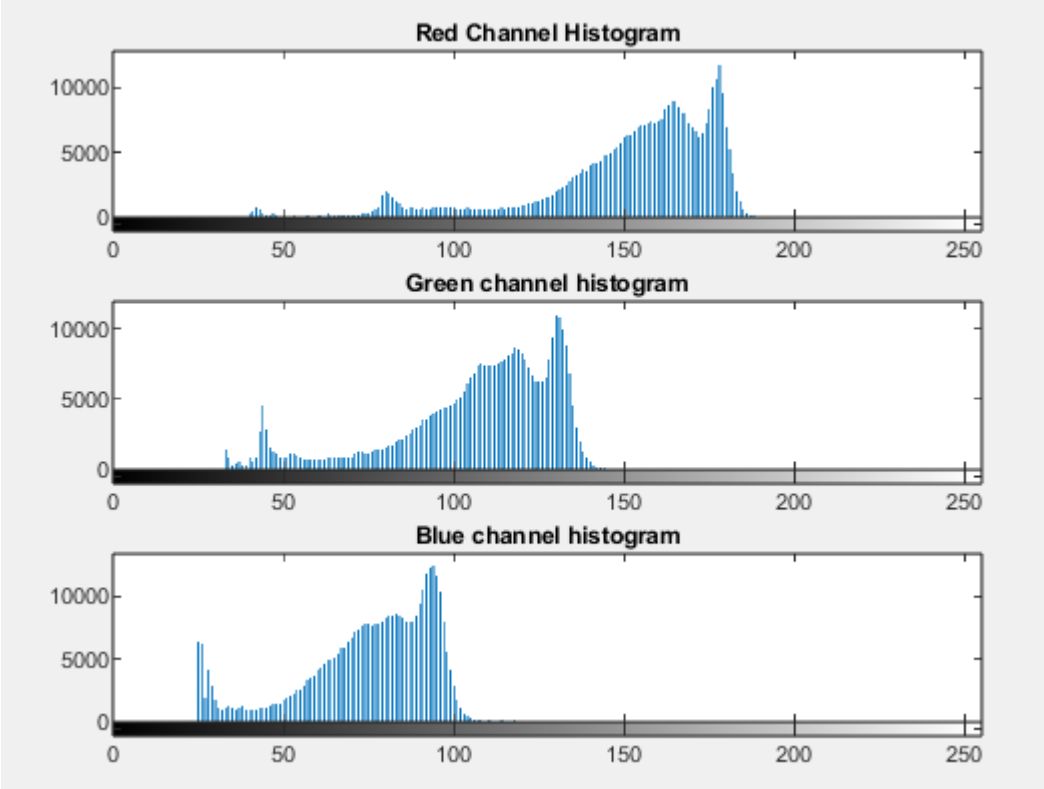


Figure 3.49: Histogram of 3.0 experimental result with 3 modules

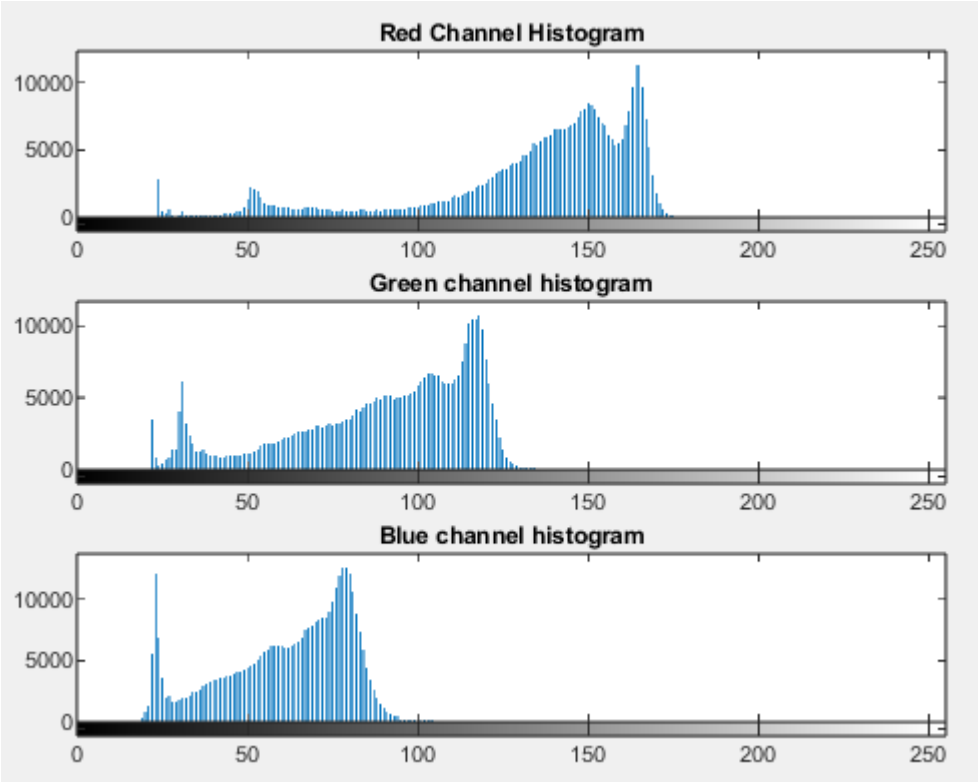
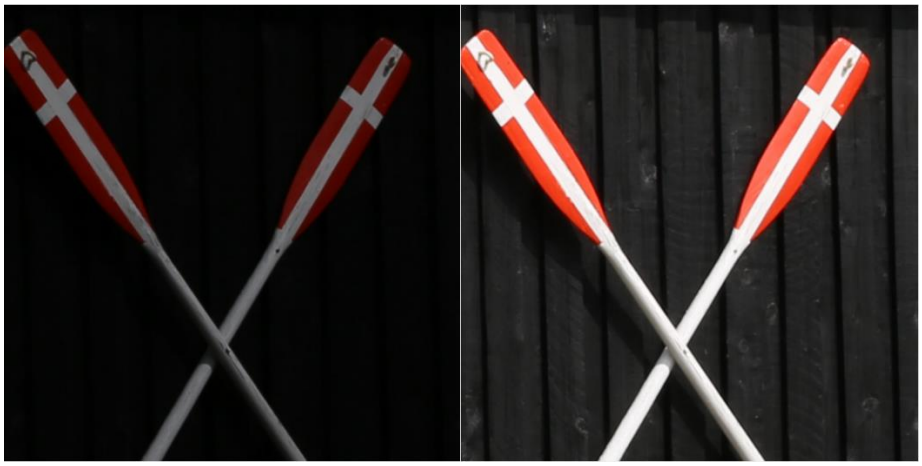


Figure 3.50: Histogram of 3.0 experimental result with 5 modules

For darkness level 3.0 we see that in all three cases the experimental images are visually very close to the ground truth case, with the LL image being fully enhanced. This is also confirmed by the histograms, where we see that the experimental histograms are characterized by a large contrast, and have a form similar to that of the ground truth histogram.

Darkness Level: 3.5



Original Low Light **Normal Light**

1 module **3 modules** **5 modules**

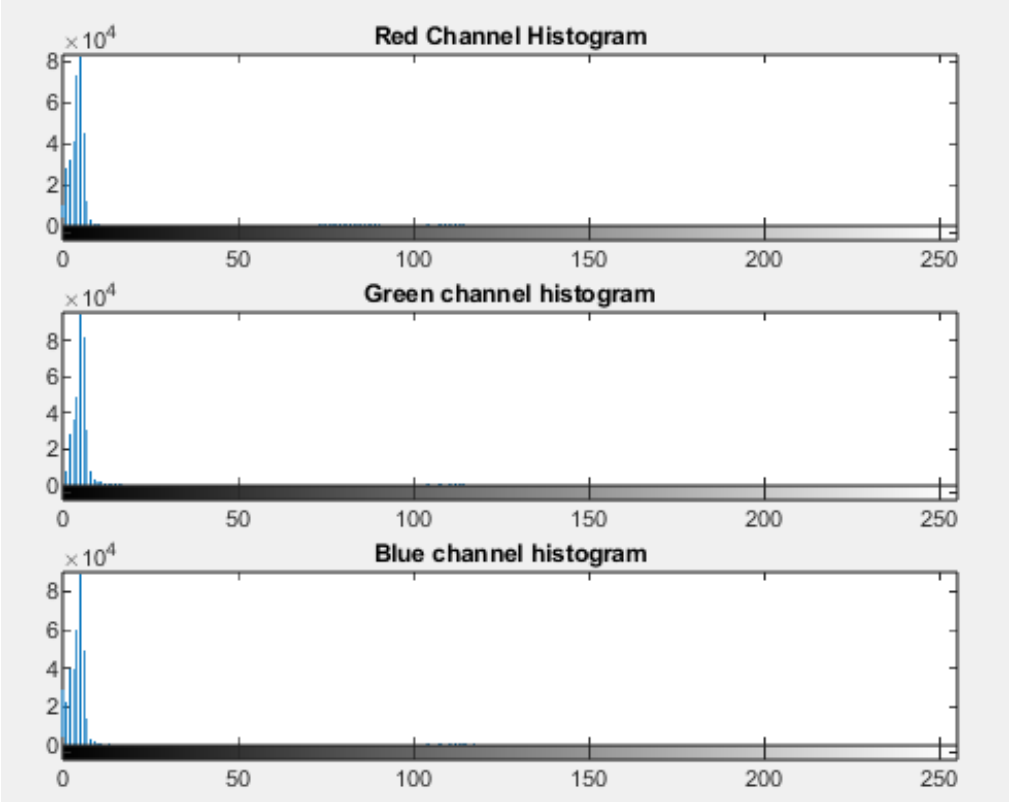


Figure 3.51: Histogram of 3.5 LL image

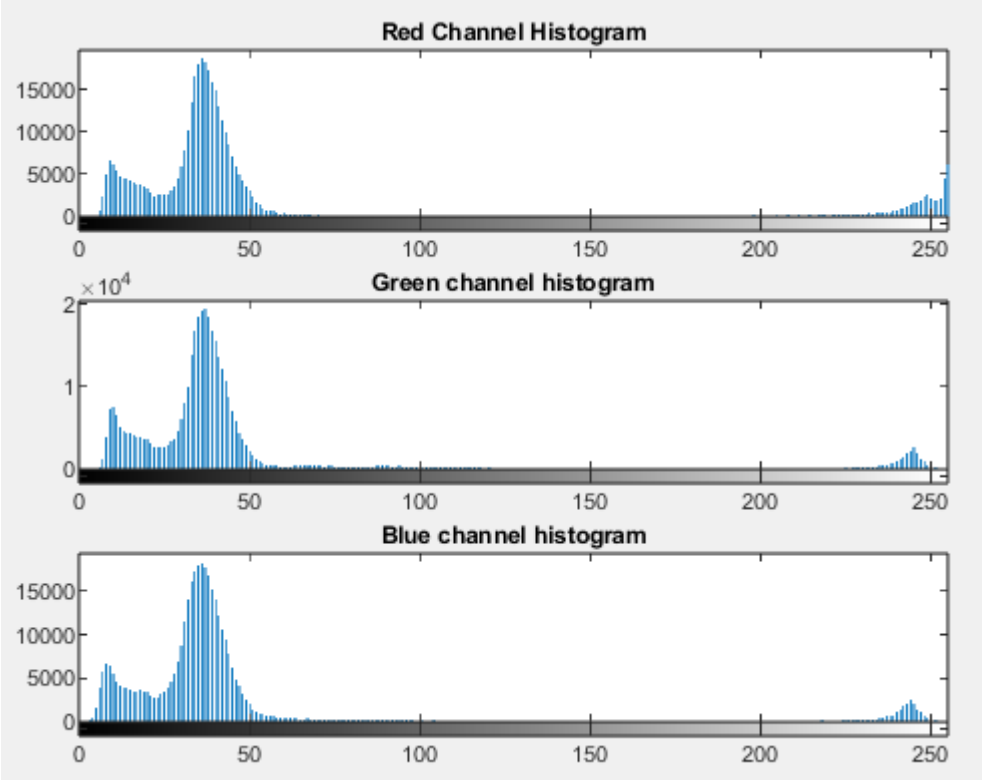


Figure 3.52: Histogram of 3.5 NL image

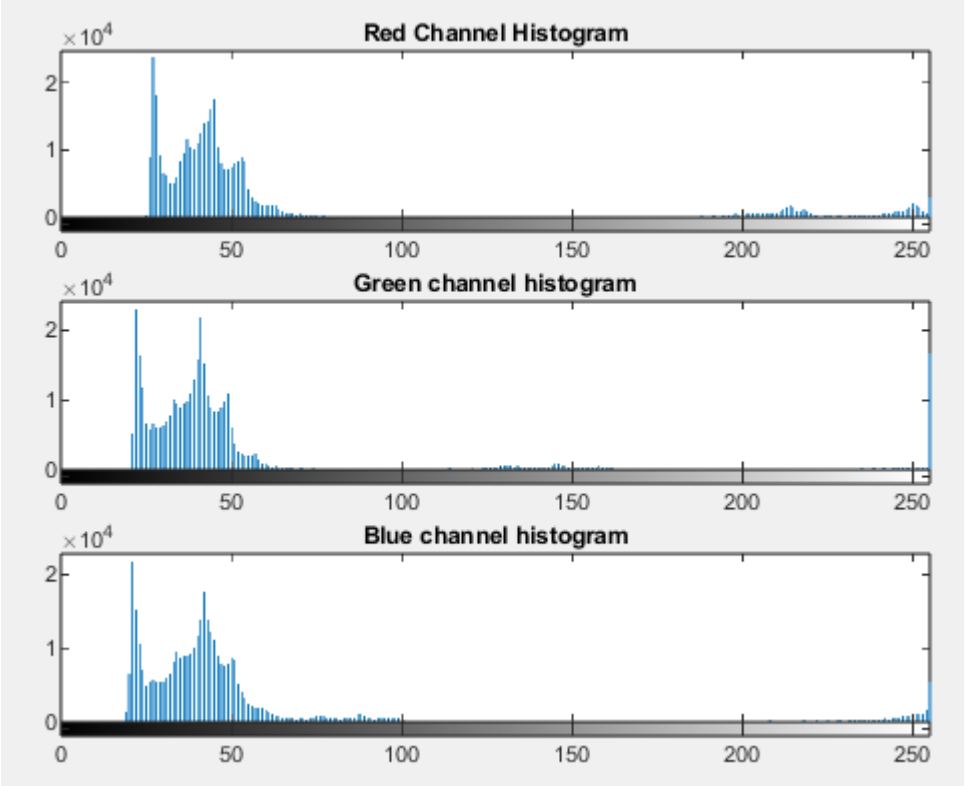


Figure 3.53: Histogram of 3.5 experimental result with 1 module

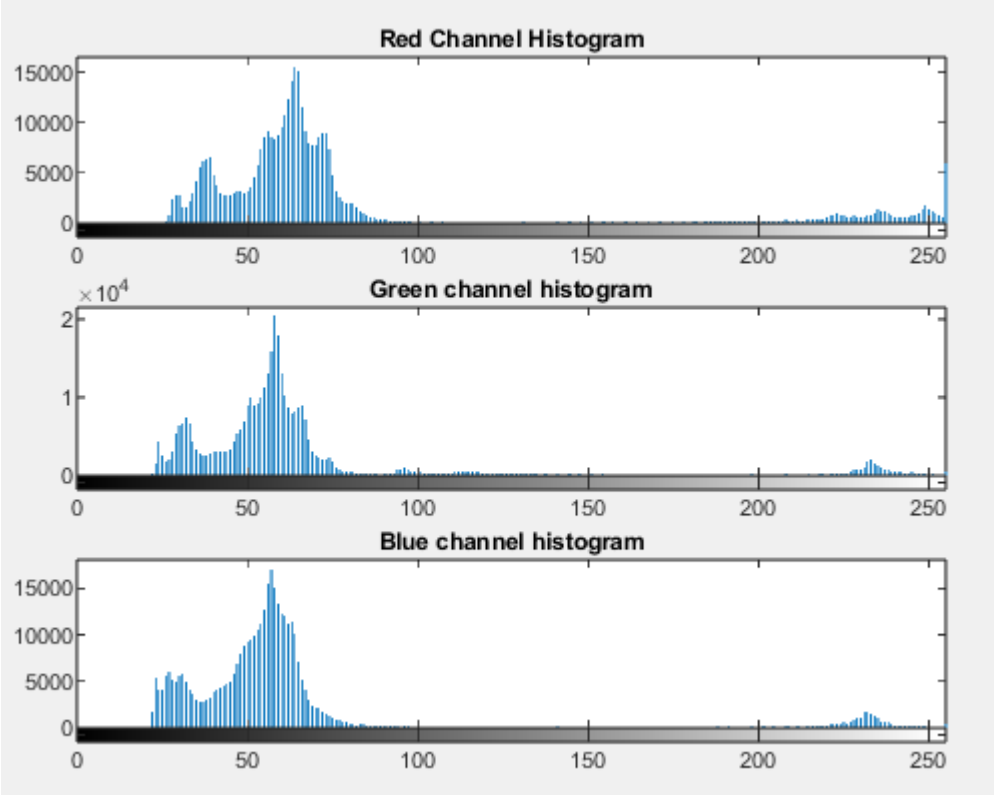


Figure 3.54: histogram of 3.5 experimental result with 3 modules

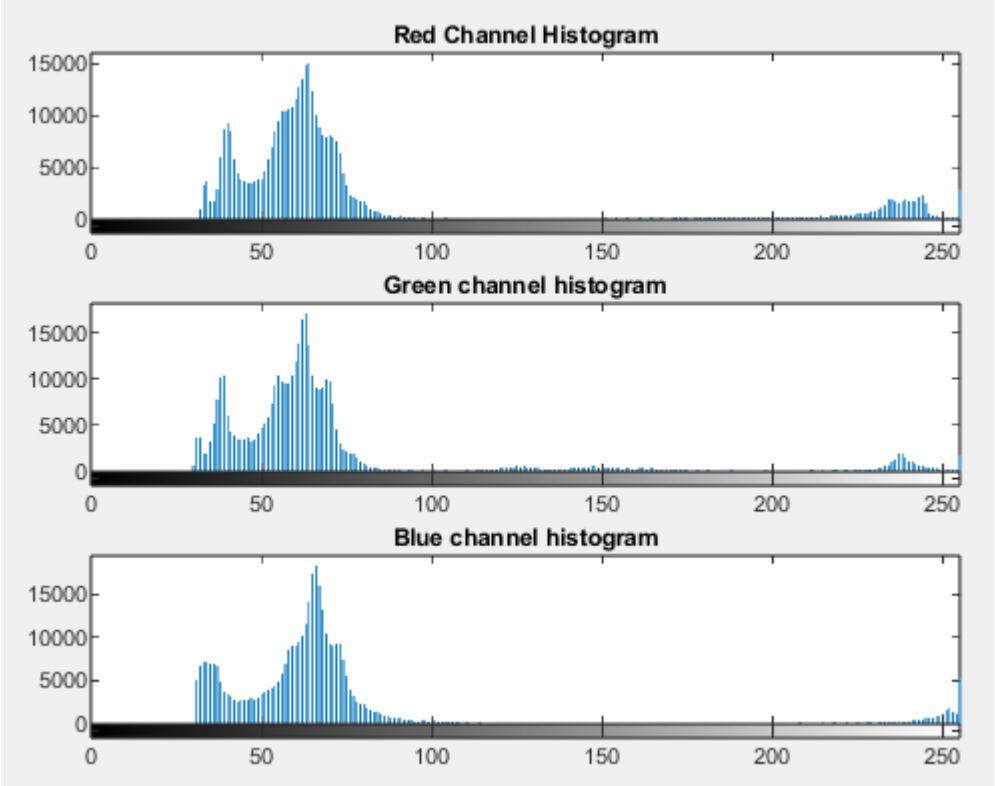


Figure 3.55: histogram of 3.5 experimental result with 5 modules

For darkness level 3.5 again we notice that the LL image has been fully enhanced, for all three cases of convolutional modules. Here it starts to look like the 3 convolutional modules produce a slightly better result, as we mentioned before. The image is enhanced better compared to the other two cases, since for 1 and 5 convolutional modules some color distortions appear, and in addition the histogram for 3 convolutional modules is closer to the ground truth form than the other two cases.

Darkness Level: 4.0

Original Low Light **Normal Light**

1 module **3 modules** **5 modules**

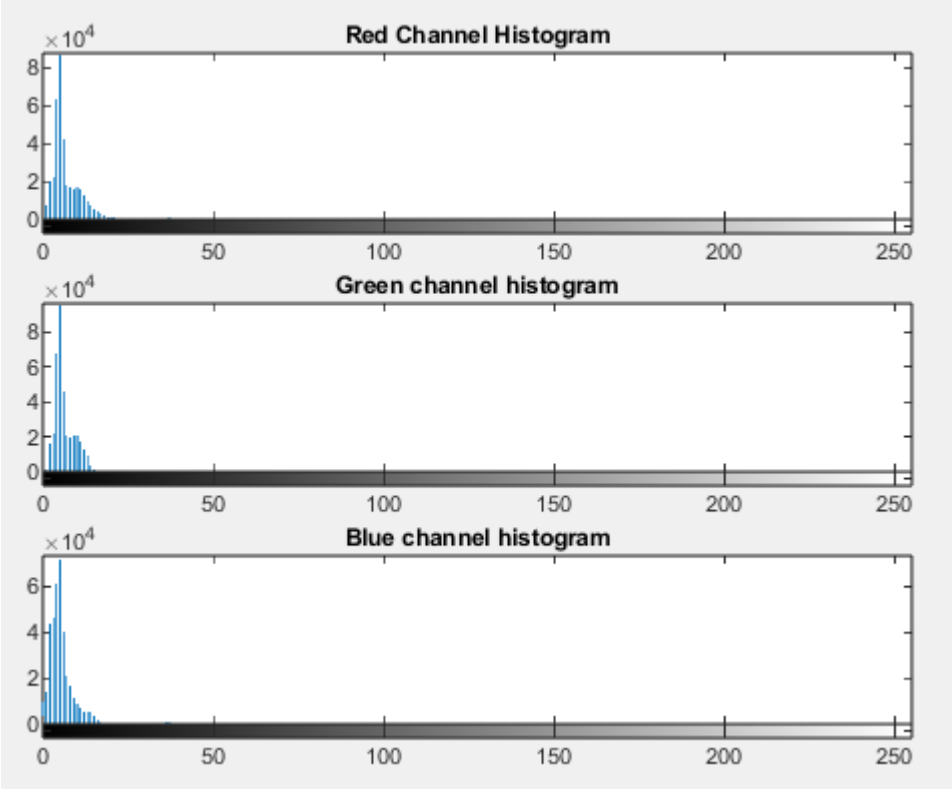


Figure 3.56: Histogram of 4.0 LL image

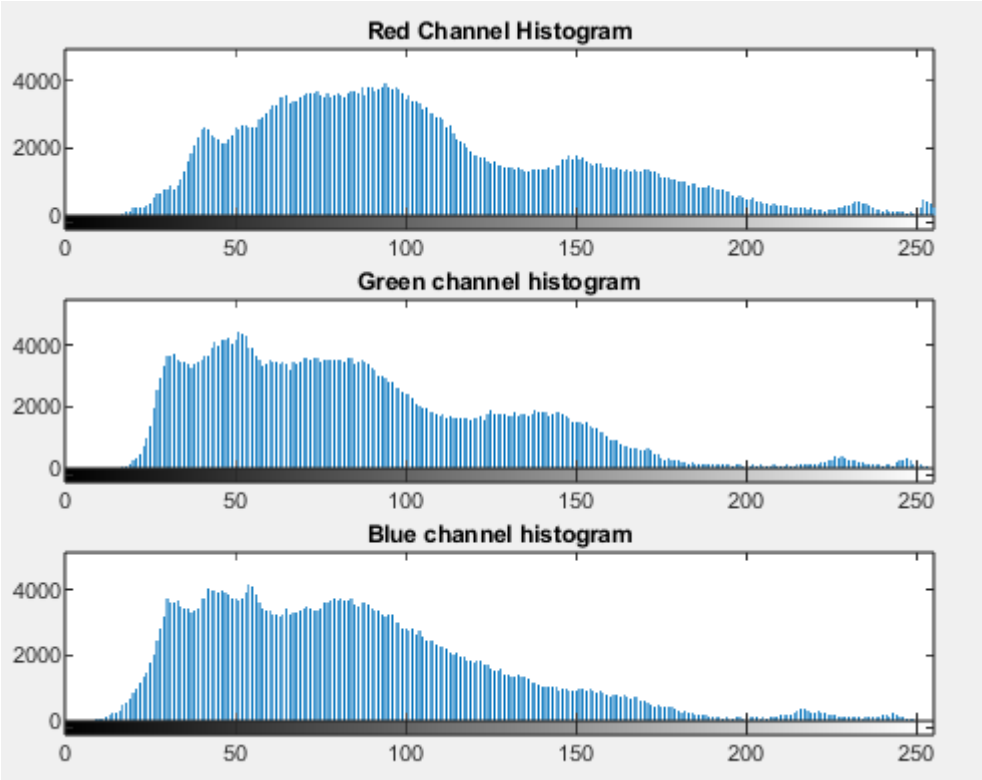


Figure 3.57: Histogram of 4.0 NL image

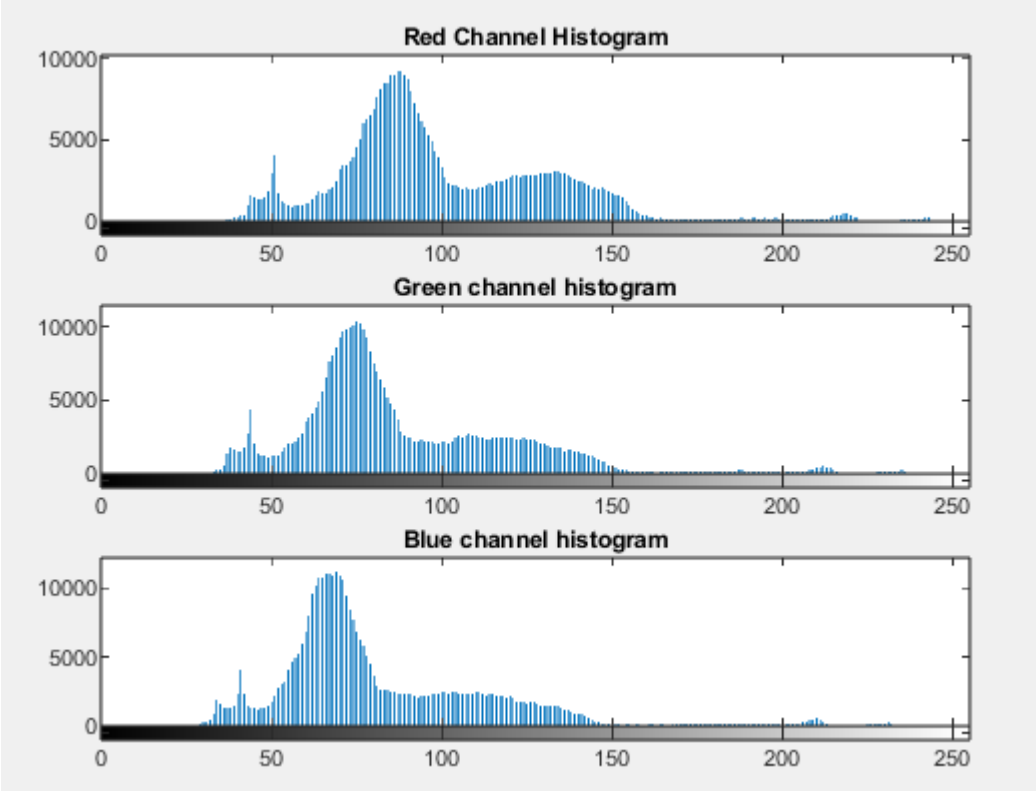


Figure 3.58: Histogram of 4.0 experimental result with 1 module

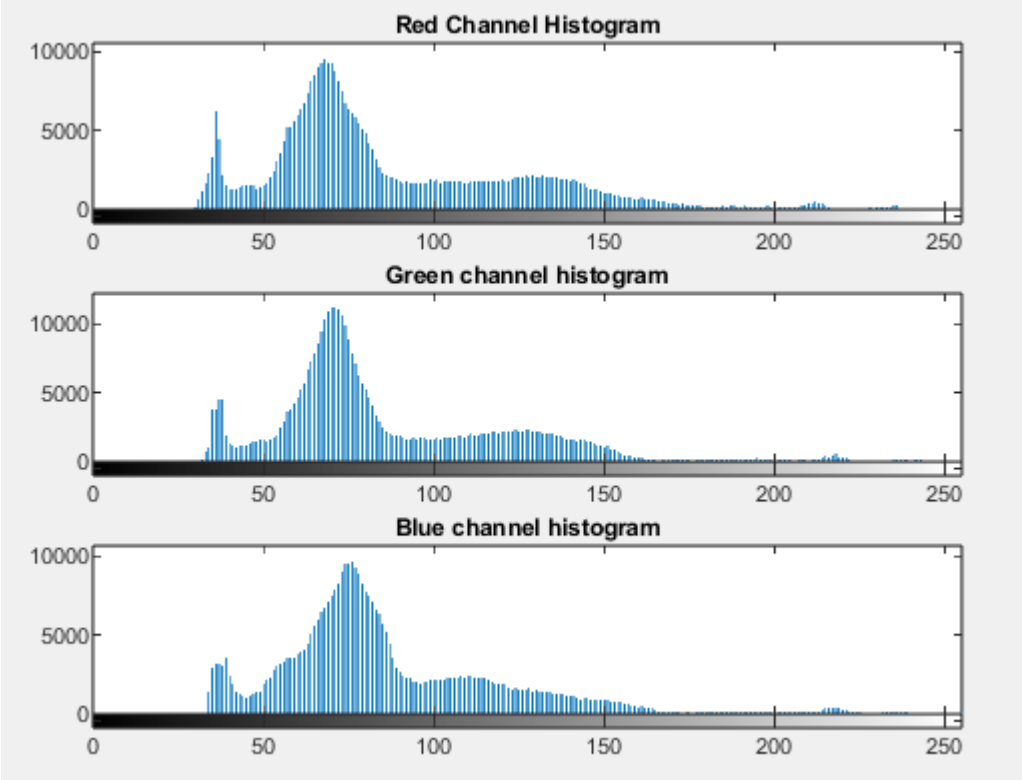


Figure 3.59: Histogram of 4.0 experimental result with 3 modules

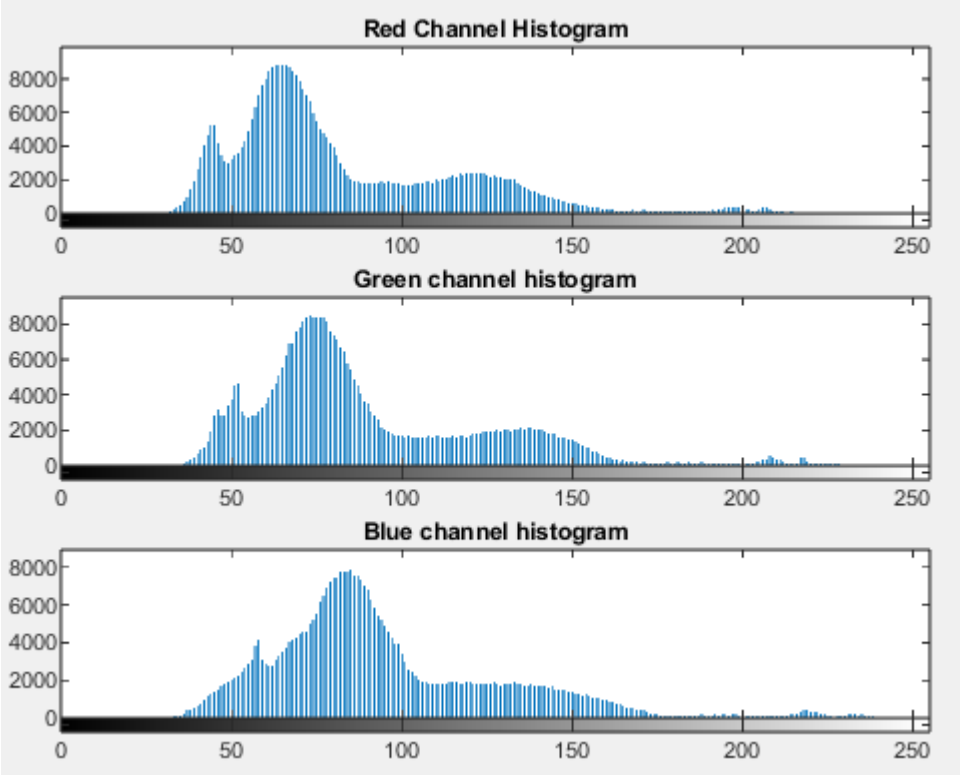


Figure 3.60: Histogram of 4.0 experimental result with 5 modules

For darkness level 4.0 we see that the experimental results are visually very close to the ground truth case. Nevertheless, due to the increase in the darkness level, we see that the model is starting to struggle, as was also found from the values of the metrics. In particular, for 1 convolutional module it seems that the color information has not been fully recovered, with the image appearing almost black and white, while for 5 convolutional modules the color seems to fade. For 3 convolutional modules the best result is obtained, with the LL image being fully enhanced, recovering a large part of the color information.

Darkness Level: 4.5

Original Low Light **Normal Light**

1 module **3 modules** **5 modules**

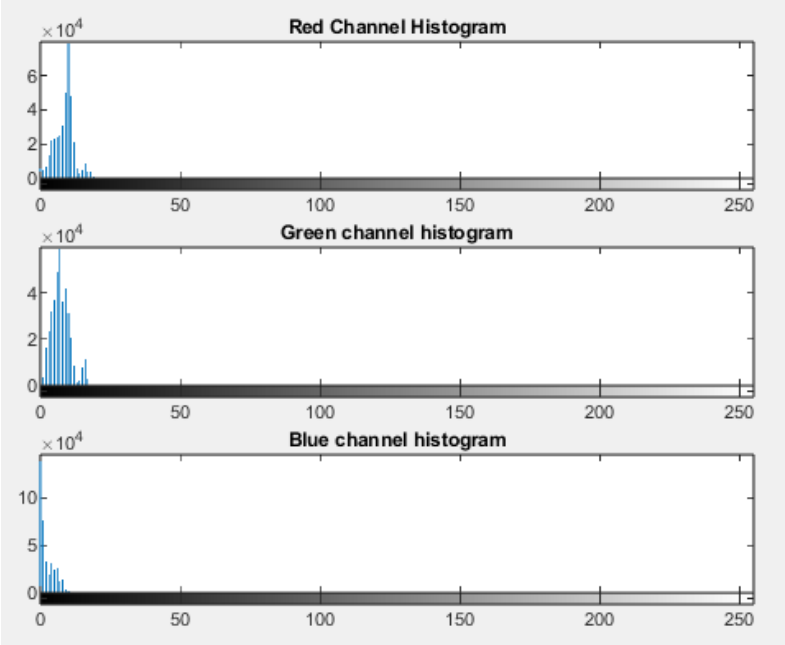


Figure 3.61: Histogram of 4.5 LL image

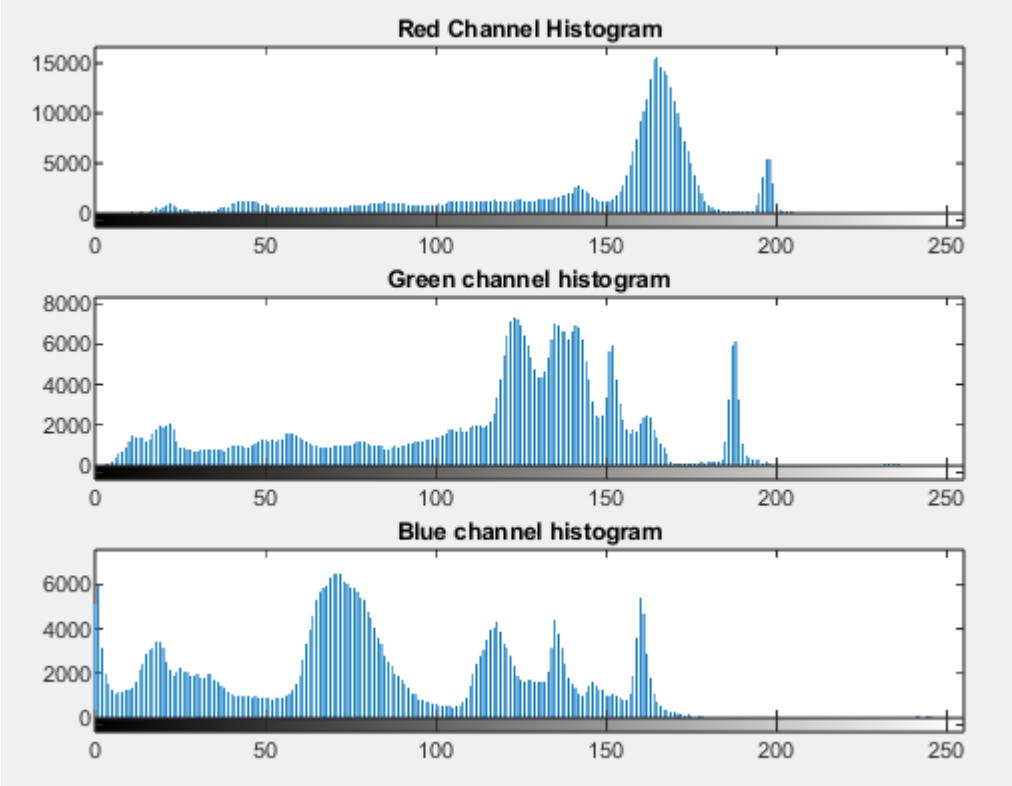


Figure 3.62: Histogram of 4.5 NL image

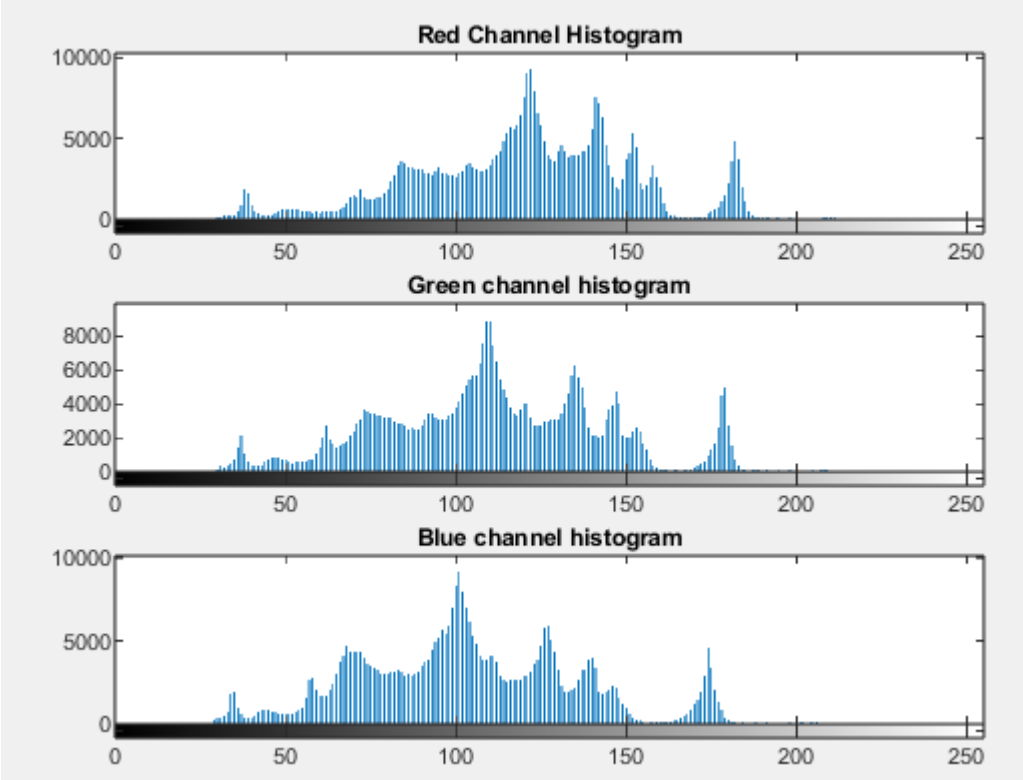


Figure 3.63: Histogram of 4.5 experimental result with 1 module

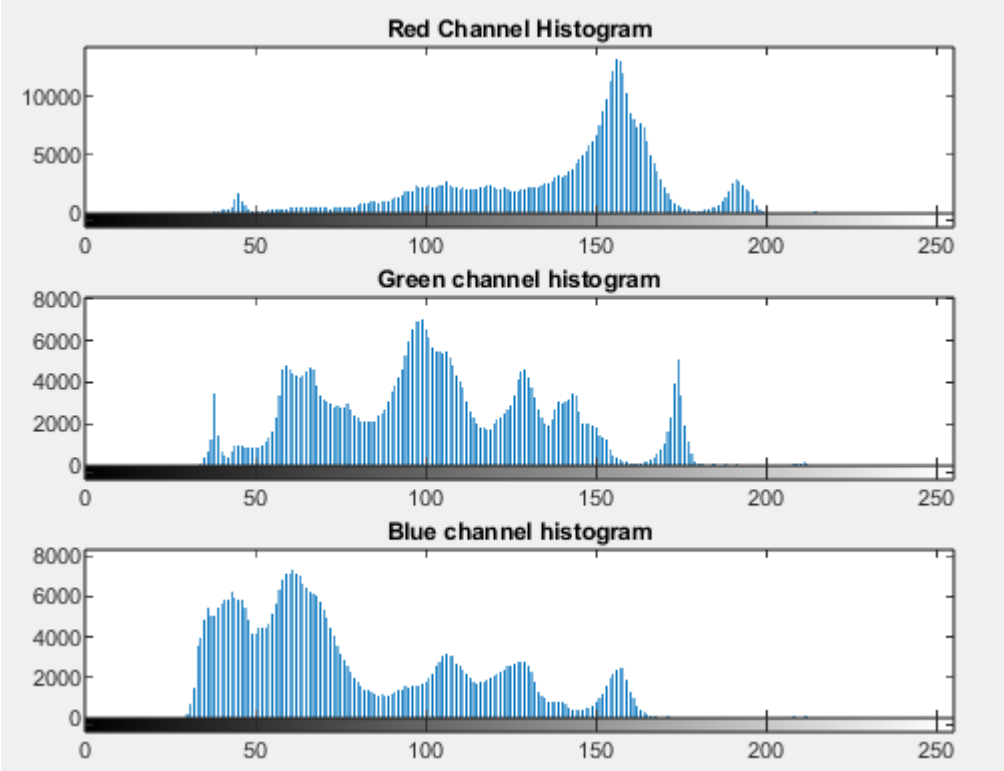


Figure 3.64: Histogram of 4.5 experimental result with 3 modules

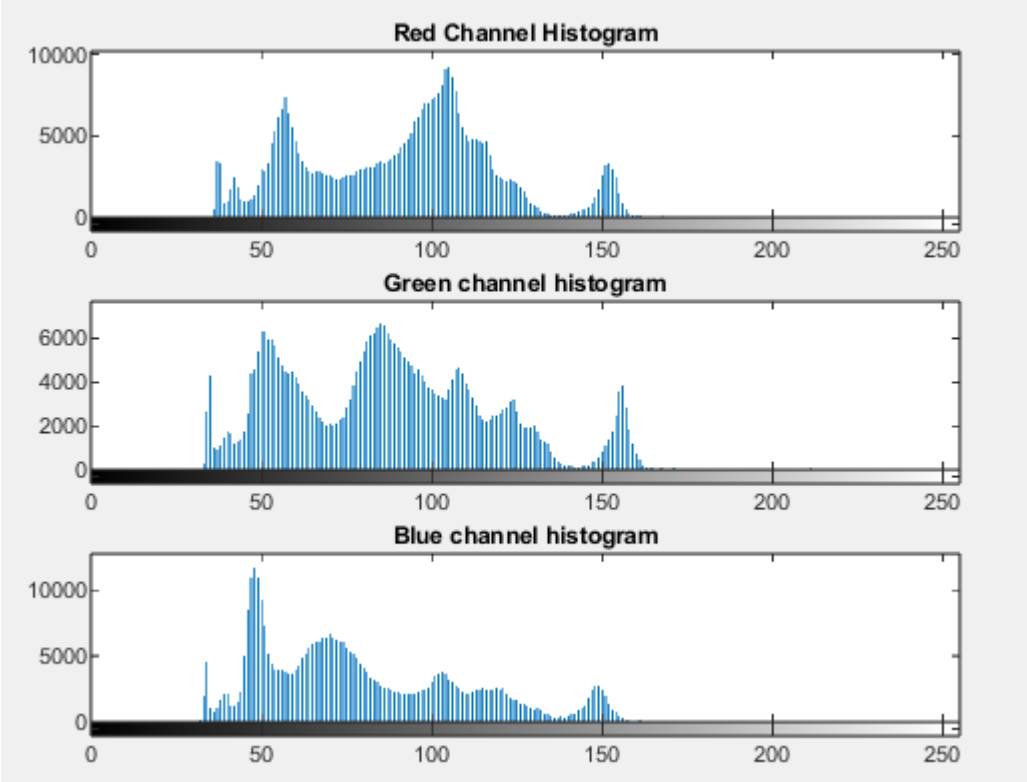
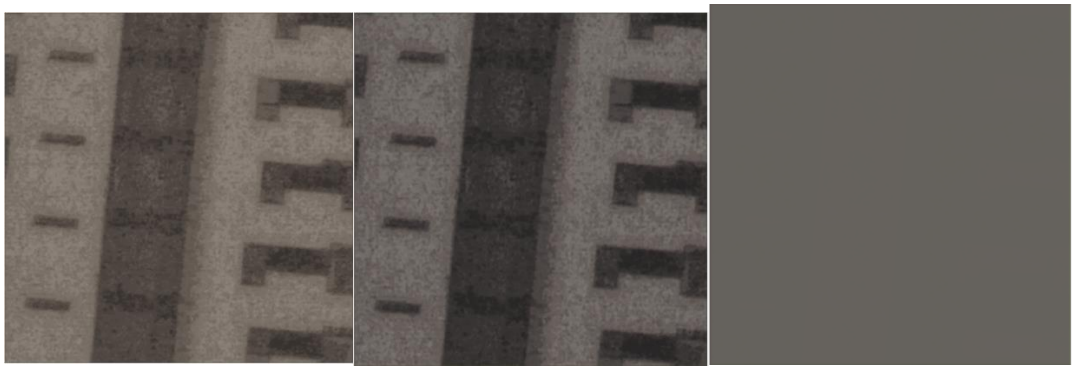


Figure 3.65: Histogram of 4.5 experimental result with 5 modules

For darkness level 4.5 we also notice that the LL image has been enhanced and we have recovered most of the visual information in all three cases of number of convolutional modules. However, again we see that for 1 and 5 convolutional modules the color information has not been recovered, while the 3 convolutional modules perform better, as we noticed from the values of the quality metrics.

Darkness Level: 5.0

Original Low Light **Normal Light**



1 module **3 modules** **5 modules**

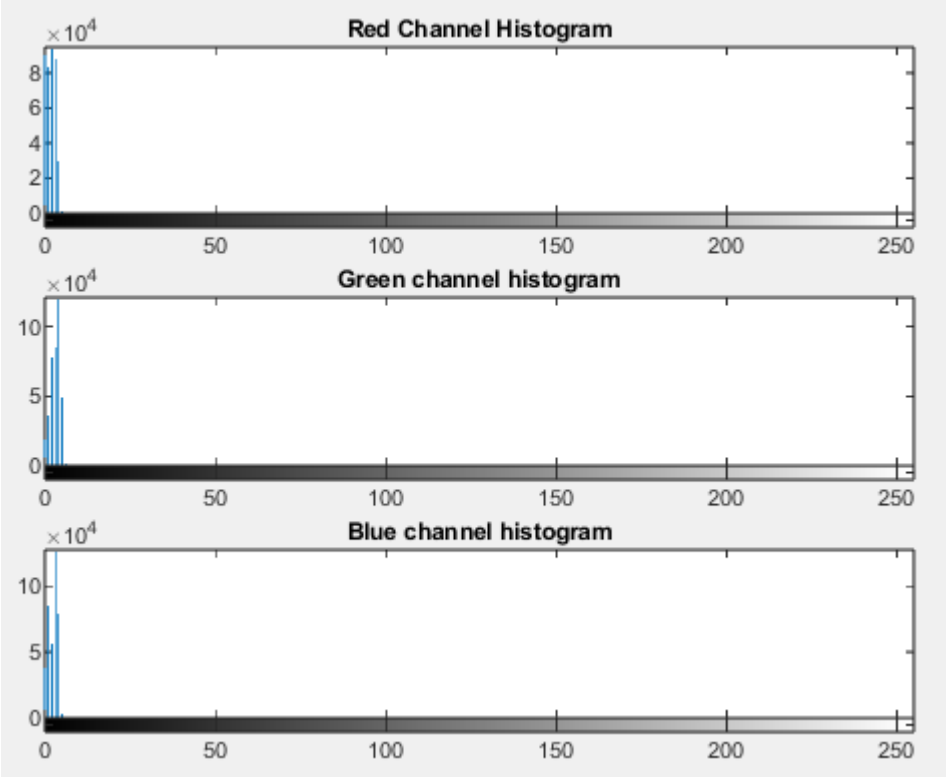


Figure 3.66: Histogram of 5.0 LL image

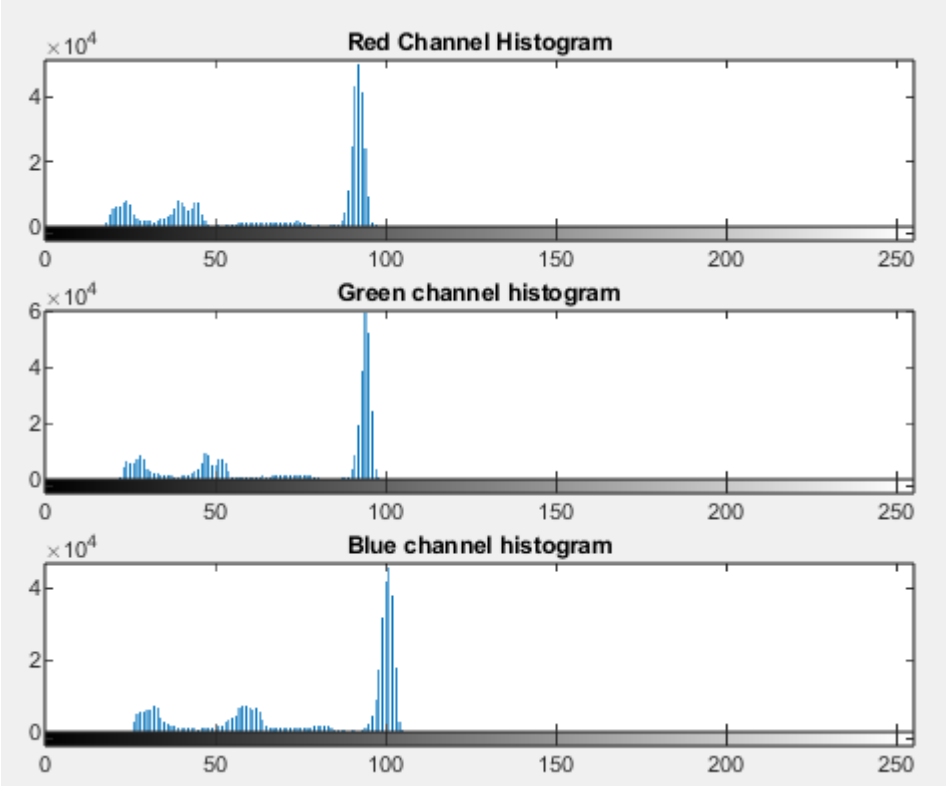


Figure 3.67: Histogram of 5.0 NL image

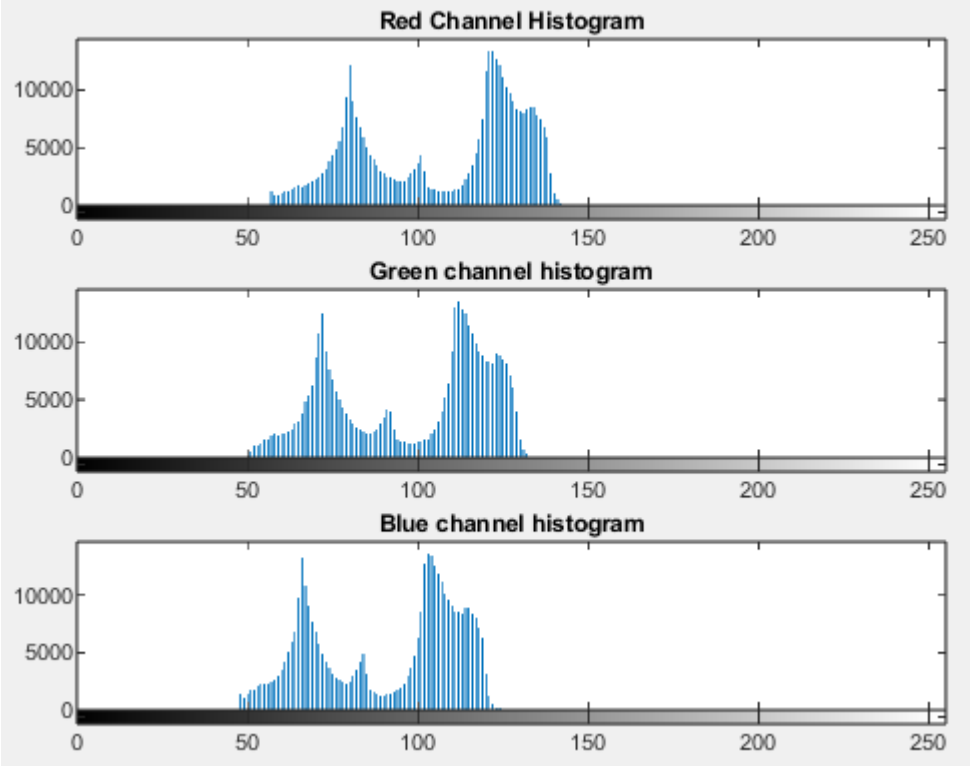


Figure 3.68: Histogram of 5.0 experimental result with 1 module

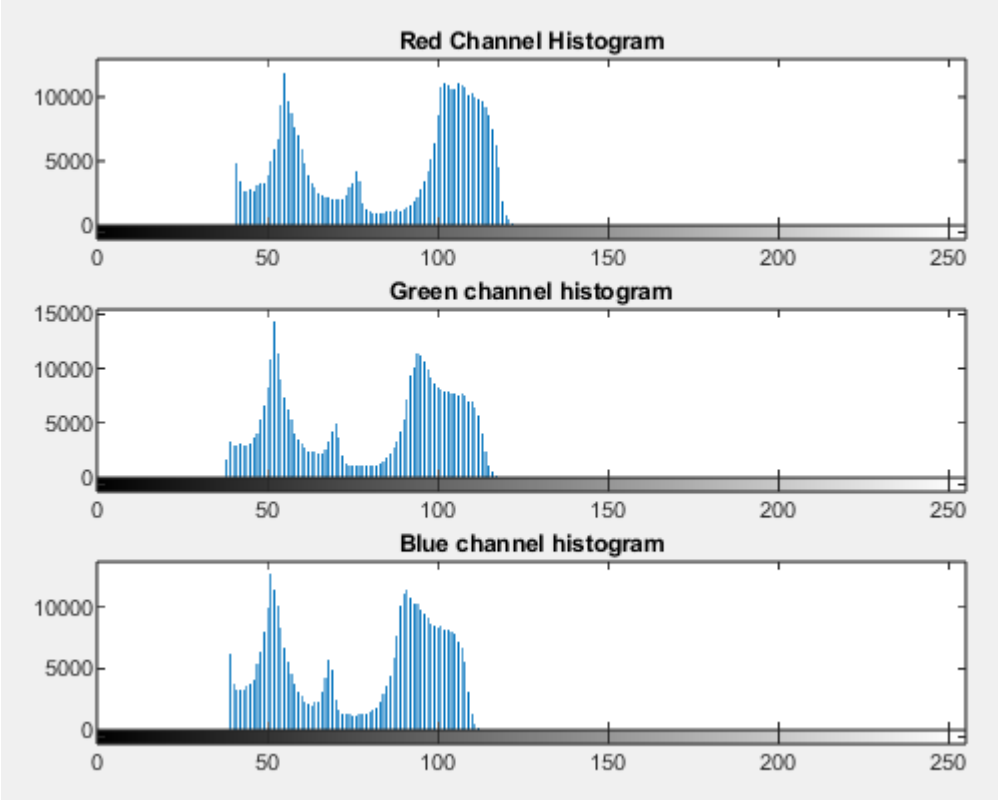


Figure 3.69: Histogram of 5.0 experimental result with 3 modules

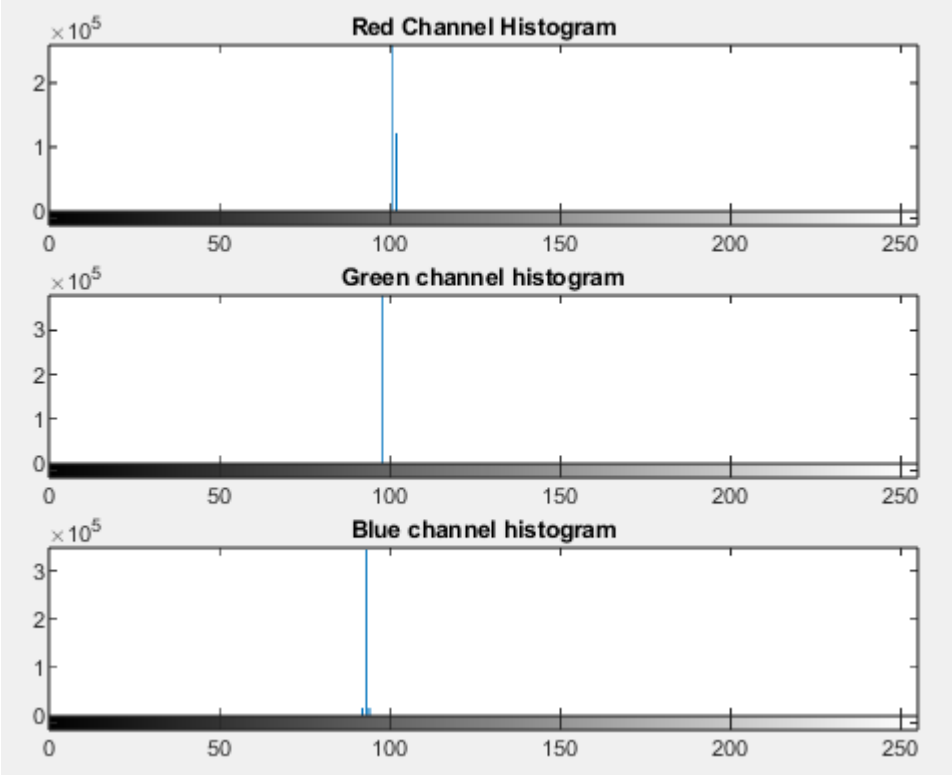


Figure 3.70: Histogram of 5.0 experimental result with 5 modules

For darkness level 5.0 we see that the model fails to fully restore the LL image. For 1 and 3 convolutional modules the general texture of the image is recovered, but without color information, while for 5 convolutional modules the result collapses completely and nothing is recovered. This is also confirmed by the values of the quality metrics we saw earlier, where for darkness level 5.0 and 5 convolutional modules the values of the metrics were of very low quality.

Based on what we have seen so far, we can notice that by adding the extra convolutional layers at the end of the architecture the performance improves, which is reflected both in the values of the performance metrics and in the visual results we presented above. Also, we saw that here too the 3 convolutional modules perform better than the other two cases, as they give the greatest improvement in the values of the metrics and recover most of the visual information. In the case of the 1 convolutional module, the reduced performance, compared to the three, is due to the fact that it learns fewer features, so it cannot properly approximate the mapping we want. For the 5 convolutional modules, the reduced performance is because we train the model for the same number of epochs as the 3 modules counterpart, and because it has more parameters, it does not have time to learn the correct mapping between LL space and NL space. These reasons

lead to the reduced performance, for all darkness levels, where we saw that modules 1 and 5 cannot restore the correct color information.

Furthermore, we noticed that as the darkness level increased the performance of the models decreased, until we reached the point where no image was even recovered for darkness level 5.0 with 5 convolutional modules. This is due to the fact that as the darkness level increases, the images become darker, and the dynamic range very small, as a result of which it becomes more and more difficult to learn the appropriate features to enhance the image. A solution to this problem would be the further training of each model for the highest darkness levels.

3.4 LL-CNN: 2nd Variation

In this section we will take a completely different approach to training our model, which is inspired by the article by He et al. [37], whose main purpose is the binarization of historical documents. The authors first assume that the ground truth image is degraded by various factors, which is expressed as:

$$x = x_u + e$$

where x_u it is the ground truth picture, x is the degraded image and e is the degradation. In this case, the basic approach to binarization with a CNN implementation is:

$$x_b = CNN(x)$$

i.e. to give the network the degraded image directly, and it should give us the binary map x_b. In this case, however, the model has to learn the degradation e, so as to remove it and recover the x_u, as well as the appropriate threshold to produce the correct binary map x_b. An easier way suggested by the authors is for the model to learn the degradation only, so

that it simply aims to enhance the degraded image, and then the result is the input of a simple binarization method. This is expressed as:

$$x_u = CNN(x) + x \rightarrow CNN(x) = -(x - x_u) = -e$$

that is, we train the model on the differences of the ground truth from the degraded image, so that it learns the noise and then remove it, giving in the end the approximation of the ground truth image, which can be an input for a simple binarization method.

Inspired by this logic, we consider that an LL image is the result of a degradation of the NL case. Specifically, we consider that a distribution e has been subtracted from the distribution of the NL image causing the pixel values of the image to decrease and by extension its brightness is also decreased. This is expressed as:

$$x_{LL} = x_{NL} - e$$

So, we could train the model in such a way that it learns the e distribution subtracted from the NL images, and then simply take the LL image, add e, and the end result is the experimental approximation of the NL image. Practically, to do this we will give as input to CNN the LL image and as output the difference of LL from the NL image, which is expressed as:

$$CNN(x_{LL}) = x_{NL} - x_{LL} = e$$

More comments and details on the implementation of the model can be found in appendix B. Below we present the results of the method.

1 module

MSE					
Level	3.0	3.5	4.0	4.5	5.0
MIN	44,8655	51,40863	72,33348	67,07643	124,8454
MAX	3426,169	3265,252	3514,585	3872,334	3797,664
AVERAGE	557,1199	560,0896	643,0971	746,0152	940,0249
	PSNR(dB)				
Level	3.0	3.5	4.0	4.5	5.0
MIN	12,78272	12,99164	12,67206	12,25108	12,33564
MAX	31,61168	31,02044	29,53741	29,8651	27,16708
AVERAGE	22,57655	22,50693	21,55051	20,81169	19,35751
	SSIM				
Level	3.0	3.5	4.0	4.5	5.0
MIN	0,16662	0,331274	0,356948	0,188938	-0,00692
MAX	0,873029	0,914021	0,84586	0,795518	0,69899
AVERAGE	0,70848	0,719564	0,66433	0,593542	0,457478
	MV				
Level	3.0	3.5	4.0	4.5	5.0
MIN	40,35464	37,98863	40,55524	39,07245	44,94147
MAX	125,3565	121,9536	122,1946	125,3441	123,0417
AVERAGE	95,69192	91,97912	90,66982	91,22758	88,1989
	STD				
Level	3.0	3.5	4.0	4.5	5.0
MIN	81,0865	43,56196	39,04175	36,89214	15,40106
MAX	1015,306	940,3215	875,3764	905,8084	879,953
AVERAGE	392,1139	357,2528	337,8879	333,0688	306,8235
	BRISQUE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	30,66738	32,14165	30,38641	34,68553	39,89657
MAX	52,5706	54,24775	52,44835	55,14162	55,30365
AVERAGE	42,27799	44,36251	45,26857	46,31306	48,73257
	NIQE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	3,226166	3,457054	3,172748	2,901554	3,362073
MAX	7,459132	6,97466	5,820608	5,202869	4,938754
AVERAGE	4,372061	4,328679	3,892451	3,879764	3,978083

Table 3.7: LLCNN 2nd variation results on test set – 1 CNN module

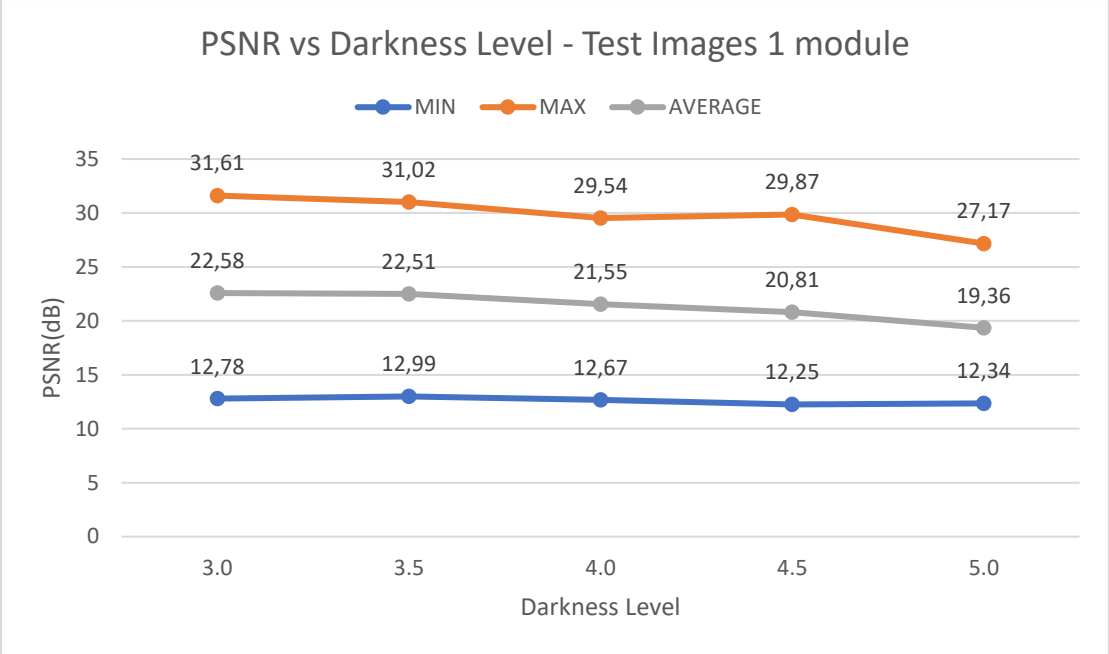


Figure 3.71: Experimental results PSNR vs Darkness level for test set

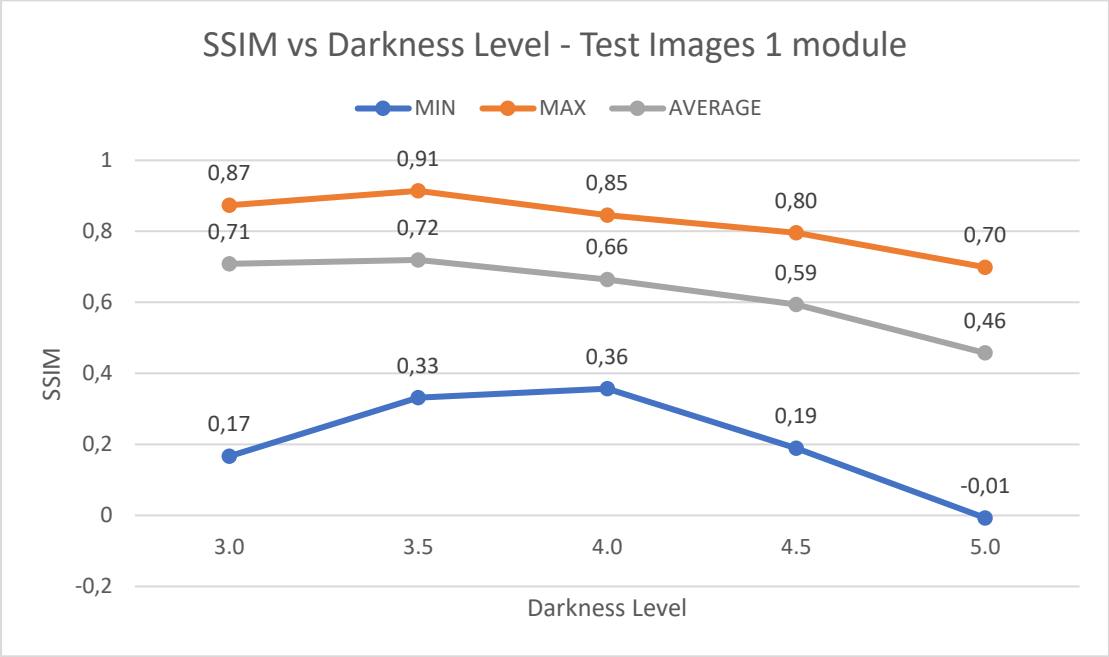


Figure 3.72: Experimental results SSIM vs Darkness level for test set

For 1 convolutional module we see that there is a clear improvement in the metric values. In comparison, the MSE decreases and the PSNR and SSIM increase, compared to the values of the LL images, which indicates to us that the experimental results are closer to the ground truth. For the quality metrics with no reference, we see that MV and STD increase, which means

the average brightness of the images increase, and they have more contrast since the pixel values are spread over a larger range around the mean value. BRISQUE and NIQE continue to have large values, which was also observed in the previous models. Finally, with the increase of the darkness level, the expected behavior of the quality metric values is observed. MSE increases while PSNR and SSIM decrease. Similarly, MV and STD decrease, indicating that for higher darkness levels the experimental results are slightly darker and with lower contrast, compared to those for lower darkness levels.

3 modules

MSE					
Level	3.0	3.5	4.0	4.5	5.0
MIN	40,71857	55,4298	101,387	71,92499	158,7336
MAX	3210,264	3832,068	3375,599	4332,126	3975,209
AVERAGE	505,5116	531,9541	629,4446	763,5596	1041,046
	PSNR(dB)				
Level	3.0	3.5	4.0	4.5	5.0
MIN	13,0654	12,29647	12,84729	11,76379	12,1372
MAX	32,03288	30,69337	28,07098	29,56201	26,12412
AVERAGE	23,13364	22,87258	21,55021	20,66279	18,77477
	SSIM				
Level	3.0	3.5	4.0	4.5	5.0
MIN	0,278397	0,238313	0,345479	0,31257	-0,0438
MAX	0,929208	0,93818	0,844109	0,817252	0,715171
AVERAGE	0,752665	0,736254	0,652447	0,6392	0,438079
	MV				
Level	3.0	3.5	4.0	4.5	5.0
MIN	42,08232	45,0009	48,48013	35,9042	49,0481
MAX	123,7666	129,0055	126,2409	122,4265	124,1735
AVERAGE	95,20668	97,77026	95,55316	86,37298	86,58998
	STD				
Level	3.0	3.5	4.0	4.5	5.0
MIN	101,4863	48,80962	35,34517	72,4157	55,66089
MAX	939,3368	958,1431	695,7557	750,4654	739,3788
AVERAGE	340,2848	339,7319	271,9623	269,1615	221,3384
	BRISQUE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	31,00837	30,99294	33,74013	38,46947	31,22574
MAX	51,46777	52,74651	51,6601	54,08553	57,80188
AVERAGE	42,1966	43,83368	44,74185	46,4601	48,53072

	NIQE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	3,492939	3,838286	3,712709	4,164159	4,183028
MAX	8,414258	8,292538	6,986006	6,658586	6,336475
AVERAGE	4,581849	4,648733	4,379942	4,662573	4,852334

Table 3.8: LLCNN 2nd variation results on test set – 3 CNN modules

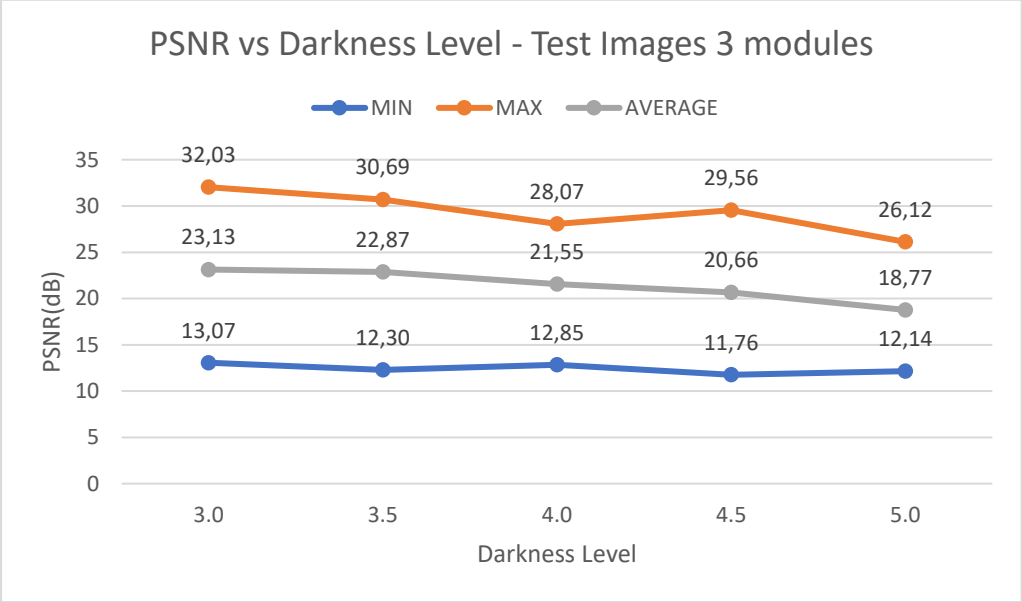


Figure 3.73: Experimental results PSNR vs Darkness level for test set

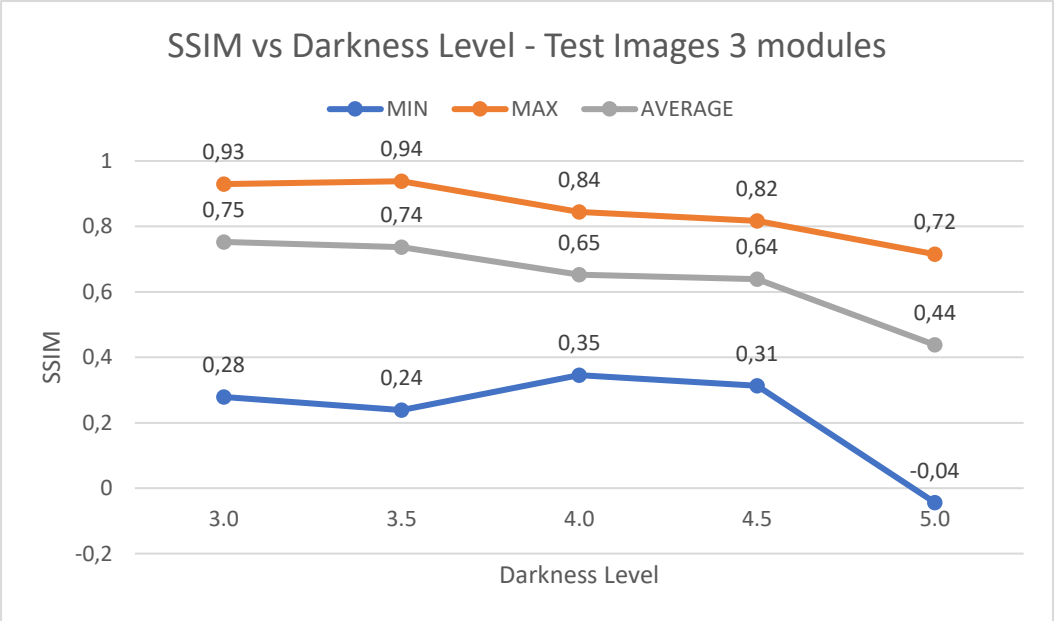


Figure 3.74: Experimental results SSIM vs Darkness level for test set

For 3 convolutional modules, a large improvement in metric values is observed, with MSE decreasing and PSNR and SSIM increasing, which indicates that the experimental images are approaching the ground truth case. Accordingly, MV and STD are increased which means that the images become brighter and with higher contrast. BRISQUE and NIQE continue to have great values, which is to be expected based on what we've seen so far in this chapter. Moreover, with the increase of the darkness level, the expected behavior is observed, as the MSE increases, while the PSNR and SSIM follow a decreasing course. Similarly, MV and STD decrease with increasing darkness level which means that the experimental results become slightly darker.

5 modules

MSE					
Level	3.0	3.5	4.0	4.5	5.0
MIN	69,88142	82,00898	144,7691	90,87492	452,1589
MAX	3888,196	3522,887	4038,085	4556,142	4911,079
AVERAGE	530,5903	582,4703	642,3333	773,1256	1219,048
	PSNR(dB)				
Level	3.0	3.5	4.0	4.5	5.0
MIN	12,23332	12,66182	12,06905	11,54483	11,21903
MAX	29,68719	28,99219	26,52404	28,54636	21,57789
AVERAGE	22,73854	22,30665	21,36374	20,738	17,80868
	SSIM				
Level	3.0	3.5	4.0	4.5	5.0
MIN	0,073512	-0,05811	0,284734	0,16876	-0,17205
MAX	0,956513	0,938152	0,820341	0,82127	0,710033
AVERAGE	0,709187	0,625161	0,633963	0,625798	0,408104
	MV				
Level	3.0	3.5	4.0	4.5	5.0
MIN	46,41683	39,9024	48,69199	33,00827	73,51005
MAX	129,7605	125,8774	131,0014	122,3284	139,6465
AVERAGE	100,0741	93,9319	102,0196	85,69117	113,1274
	STD				
Level	3.0	3.5	4.0	4.5	5.0
MIN	119,7383	24,86499	85,98113	41,80059	113,7915
MAX	1027,345	928,3512	845,0857	845,8448	670,4729
AVERAGE	367,6629	309,1584	291,2608	269,2211	257,0771
	BRISQUE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	31,72861	32,89929	32,55808	34,581	36,95518
MAX	51,29019	51,67033	50,84819	52,75801	52,28734
AVERAGE	43,25488	44,42939	44,73829	45,79551	45,65563

	NIQE				
Level	3.0	3.5	4.0	4.5	5.0
MIN	3,963563	4,201967	3,947916	4,076733	3,857366
MAX	7,907503	8,180129	8,56234	7,693572	8,485583
AVERAGE	5,080761	5,187741	4,88935	4,947641	5,4042

Table 3.9: LLCNN 2nd variation results on test set – 5 CNN modules

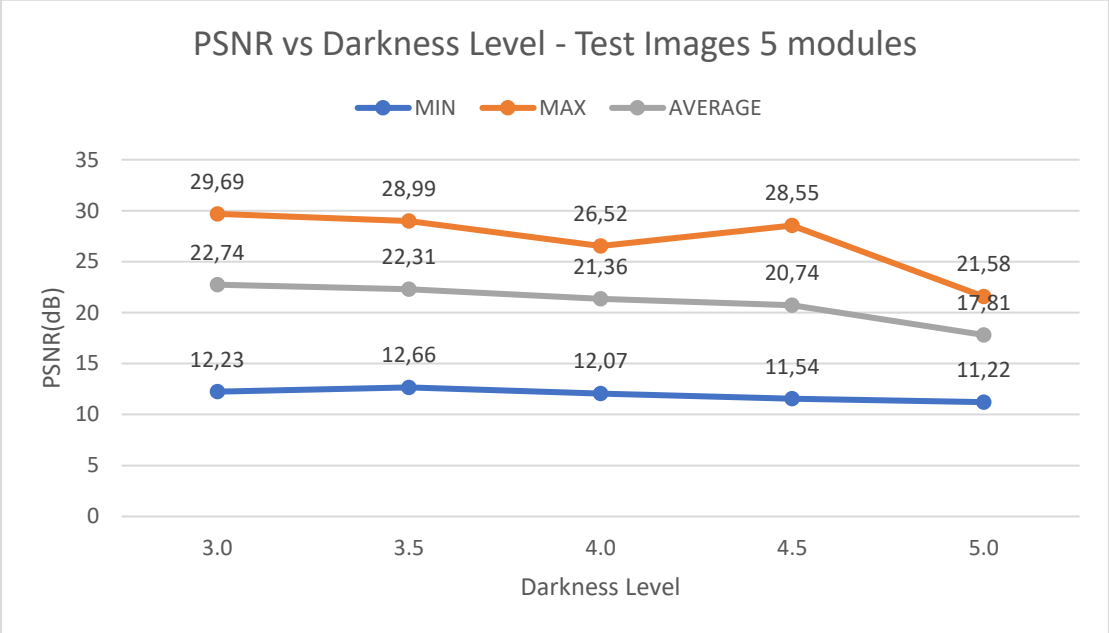


Figure 3.75: Experimental results PSNR vs Darkness level for test set

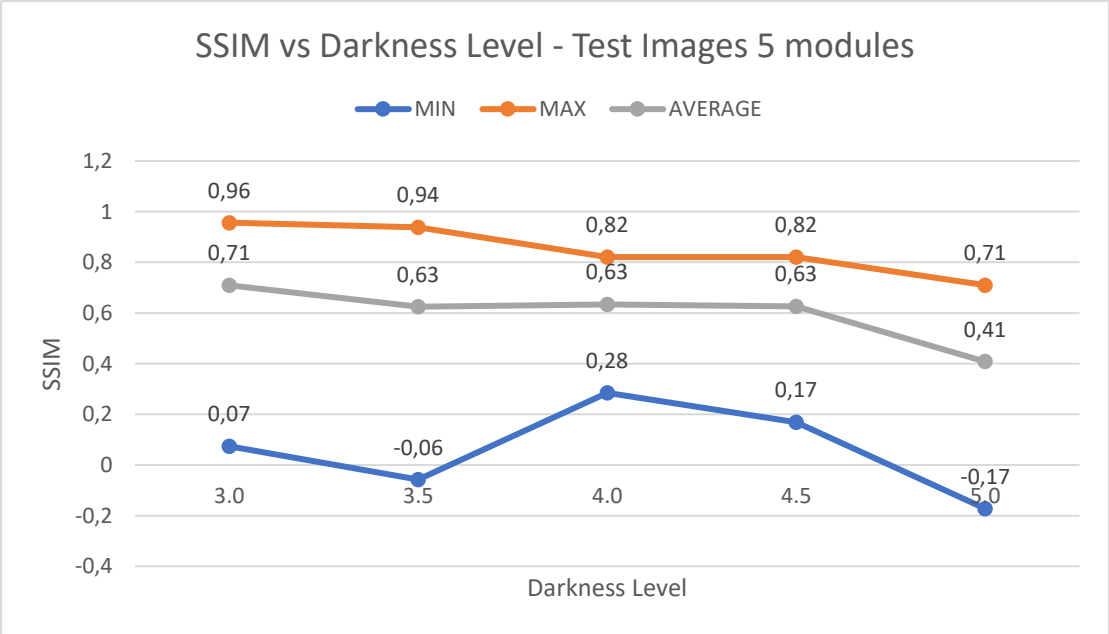
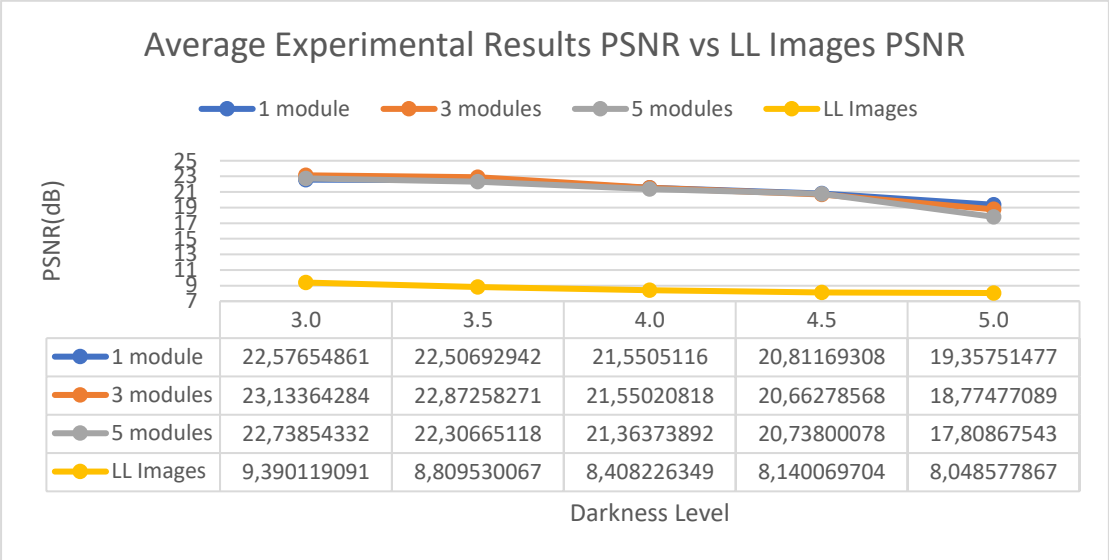


Figure 3.76: Experimental results SSIM vs Darkness level for test set

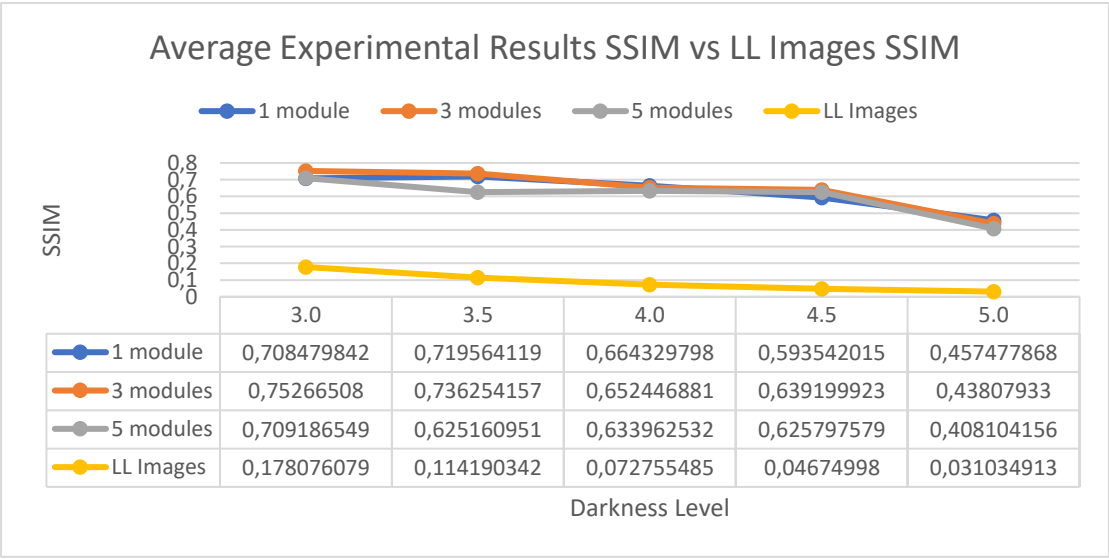
For 5 convolutional modules we again observe an improvement in the values of the metrics. The MSE decreases and the PSNR and SSIM increase, compared to the values for the LL images, which means that there is a visual improvement of the results. Also, MV and STD increase, so experimental images will be brighter and characterized by higher contrast, which confirms the enhancement of LL images. BRISQUE and NIQE continue to have great values, as expected. Finally, with the increase in the darkness level, the expected course is followed here as well, as the quality of the metrics decreases with its increase.



	3.0	3.5	4.0	4.5	5.0
1 module	22,57654861	22,50692942	21,5505116	20,81169308	19,35751477
3 modules	23,13364284	22,87258271	21,55020818	20,66278568	18,77477089
5 modules	22,73854332	22,30665118	21,36373892	20,73800078	17,80867543
LL Images	9,390119091	8,809530067	8,408226349	8,140069704	8,048577867

Darkness Level

Figure 3.77: Experimental results average PSNR vs LL Images PSNR test set



	3.0	3.5	4.0	4.5	5.0
1 module	0,708479842	0,719564119	0,664329798	0,593542015	0,457477868
3 modules	0,75266508	0,736254157	0,652446881	0,639199923	0,43807933
5 modules	0,709186549	0,625160951	0,633962532	0,625797579	0,408104156
LL Images	0,178076079	0,114190342	0,072755485	0,04674998	0,031034913

Darkness Level

Figure 3.78: Experimental results average SSIM vs LL Images SSIM test set

Overall, we saw that for all cases of number of convolutional modules there is a clear improvement of the metric values, which also indicates the enhancement of LL images. For 1 convolutional module the average PSNR increases by 12.8dB, for 3 it increases by 12.64dB and for 5 it increases by 12.43dB. Similarly, the mean SSIM increases by 0.54 for 1 convolutional module, and by 0.56 and 0.52 for 3 and 5 convolutional modules respectively. From figures 3.77 and 3.78 we see that there is no great variation in the values of the metrics with the increase in the number of convolutional modules, which is also confirmed by the average increase in PSNR and SSIM that we mentioned above. We understand that based on metric values alone, we cannot decide which case of number of convolutional modules is better. Ideally, we would choose the case with 1 convolutional module as it has fewer parameters and is easier to train. Finally, we noticed that as the darkness level increases, the performance of the model decreases. This is to be expected as with the increase of the darkness level the images become very dark, with the dynamic range being small, as a result of which it is difficult to learn the necessary features to enhance the images. As we mentioned in the previous models, a solution to this problem could be the further training of the model at higher darkness levels, so that it has time to learn all the features it needs.

To visually see the results of the model we will display, for each darkness level, a random image for each case of number of convolutional modules, together with the corresponding LL and ground truth images. In addition, we will also display the respective histograms, to confirm that the visual information has been completely recovered.

Darkness Level: 3.0

Original Low Light **Normal Light**

1 module **3 modules** **5 modules**

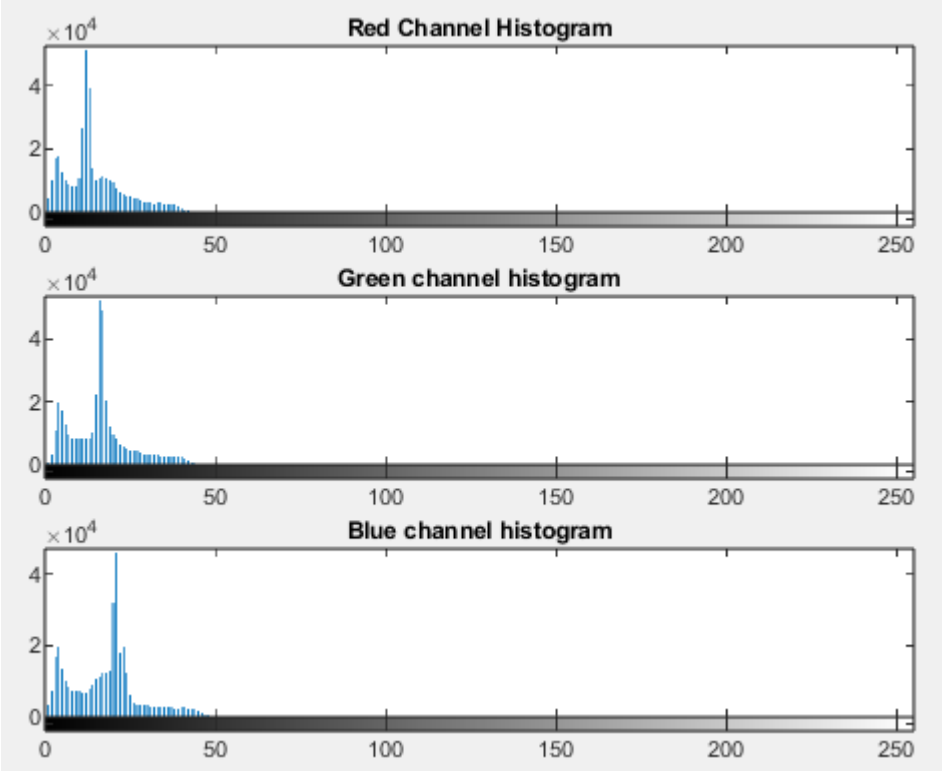


Figure 3.79: Histogram of 3.0 LL image

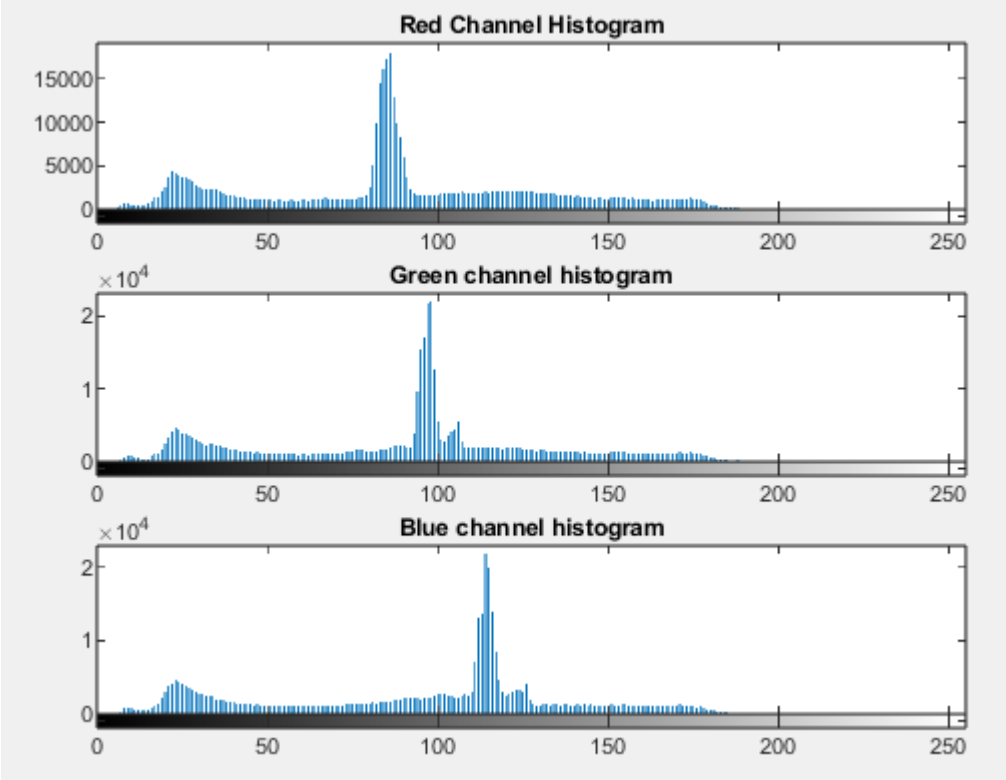


Figure 3.80: Histogram of 3.0 NL image

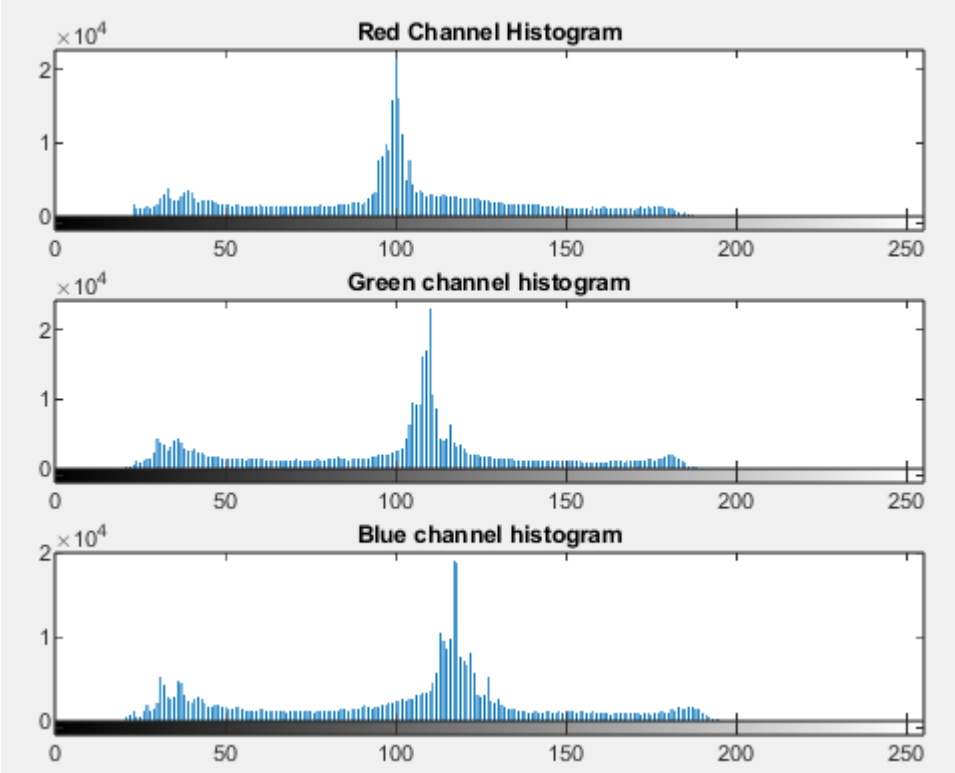


Figure 3.81: Histogram of 3.0 experimental result with 1 module

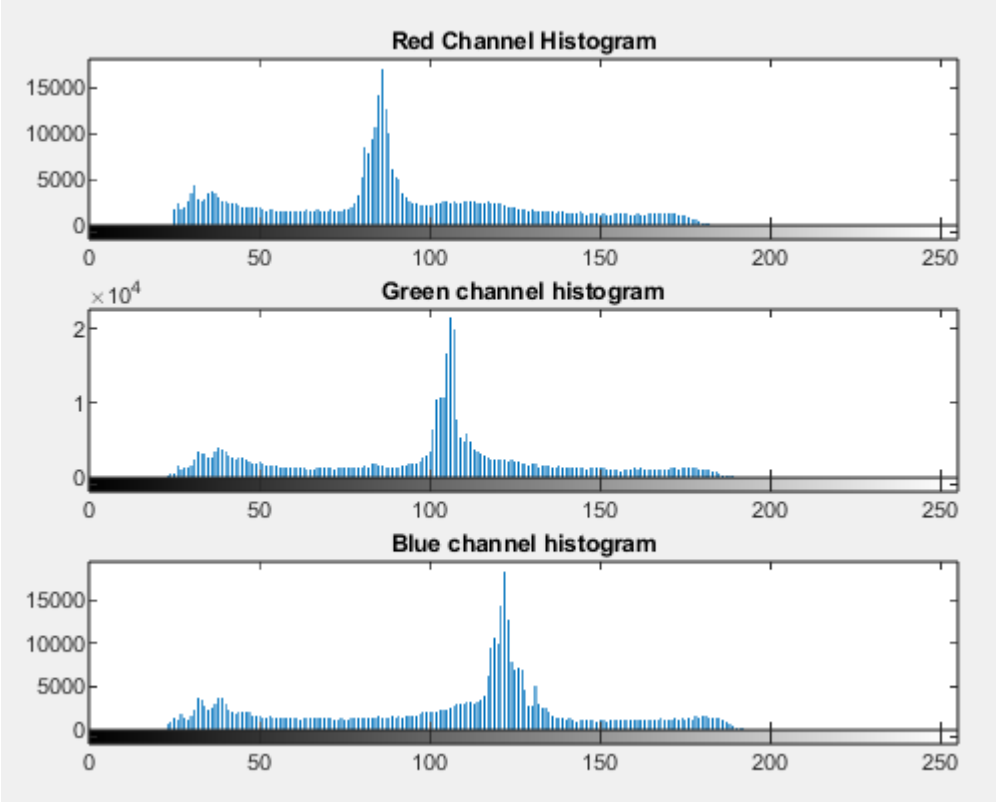


Figure 3.82: Histogram of 3.0 experimental result with 3 modules

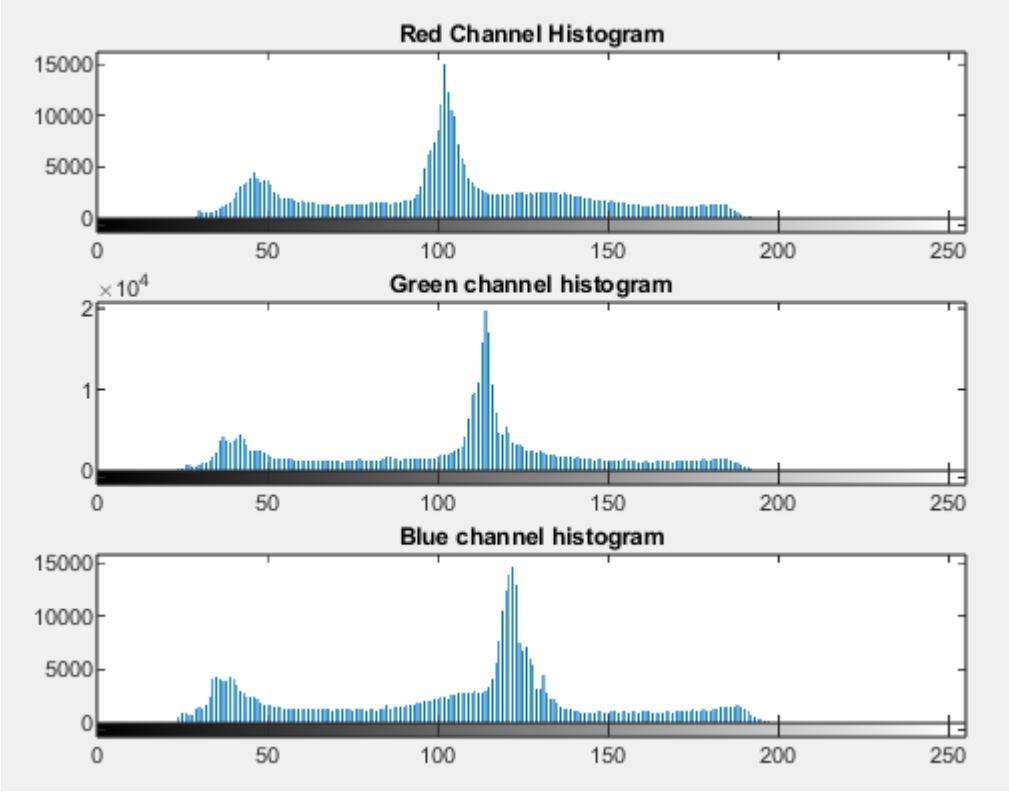


Figure 3.83: Histogram of 3.0 experimental result with 5 modules

For darkness level 3.0 we see that the LL image has been fully enhanced for all three cases of the number of convolutional modules, as both the color information and the details of the image have been recovered, with the experimental results being visually very close to the ground truth case. The quality of the result is also confirmed by the histograms, where in all three cases, the experimental histograms have a shape almost identical to the corresponding ground truth, which means that all the visual information has been recovered. As we mentioned above, we cannot decide which case of convolutional modules is better, as all three give impressive results.

Darkness Level: 3.5

Original Low Light **Normal Light**

1 module **3 modules** **5 modules**

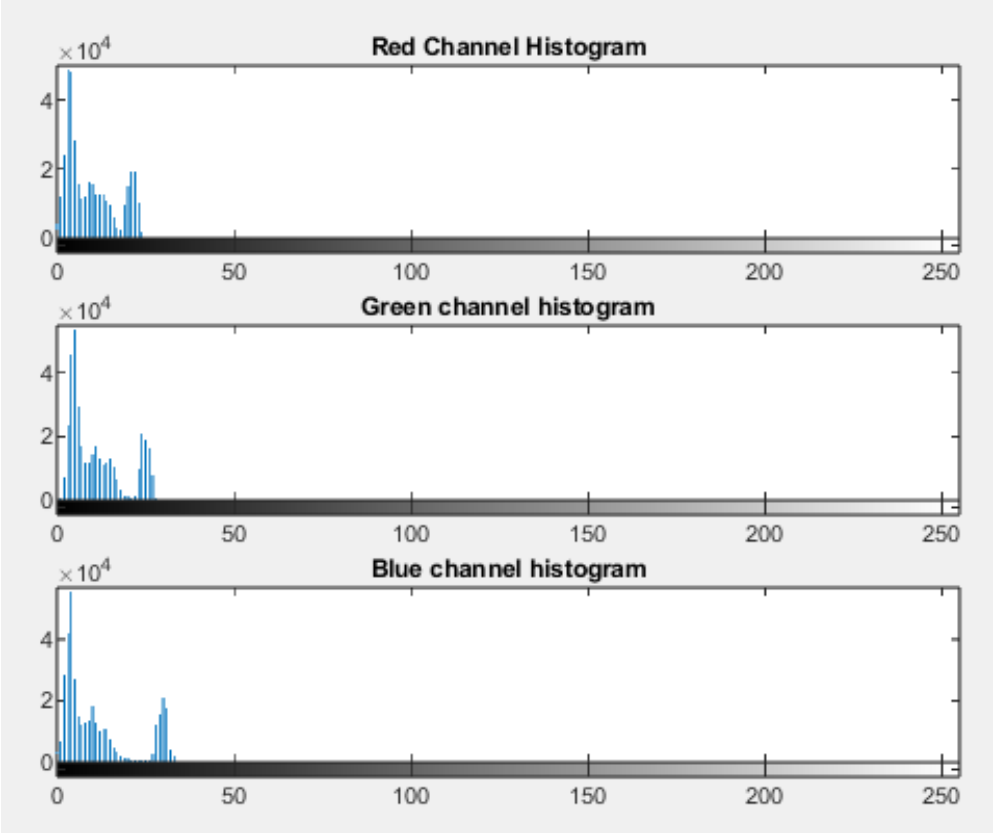


Figure 3.84: Histogram of 3.5 LL image

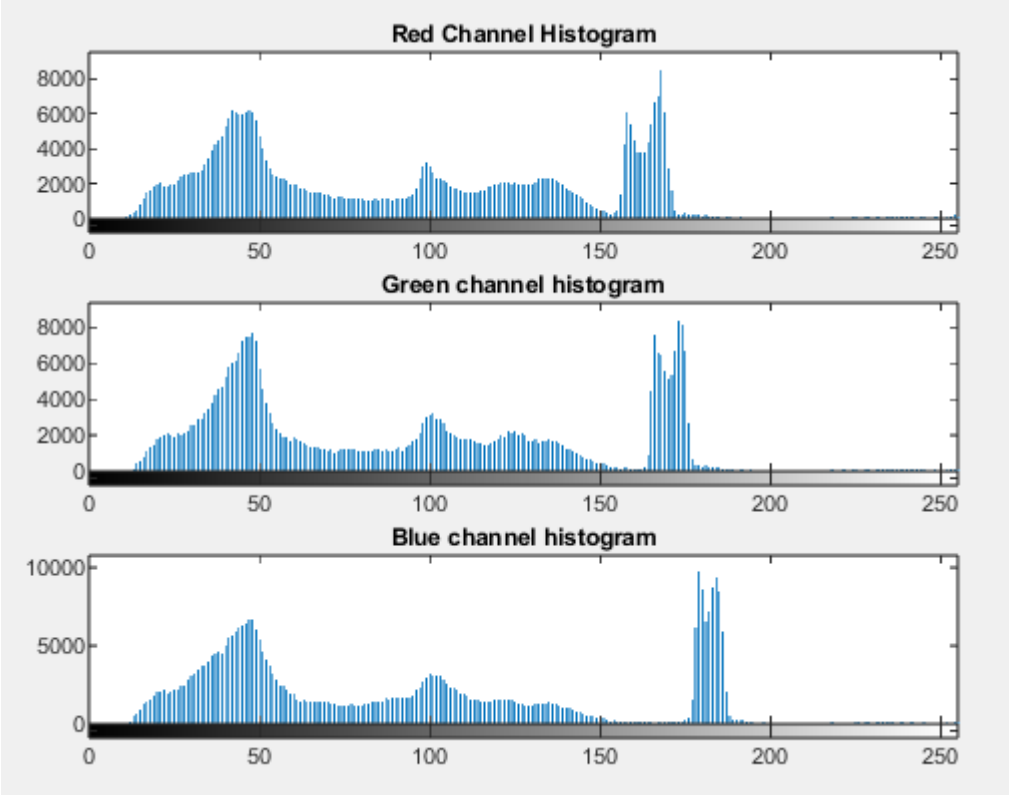


Figure 3.85: Histogram of 3.5 NL image

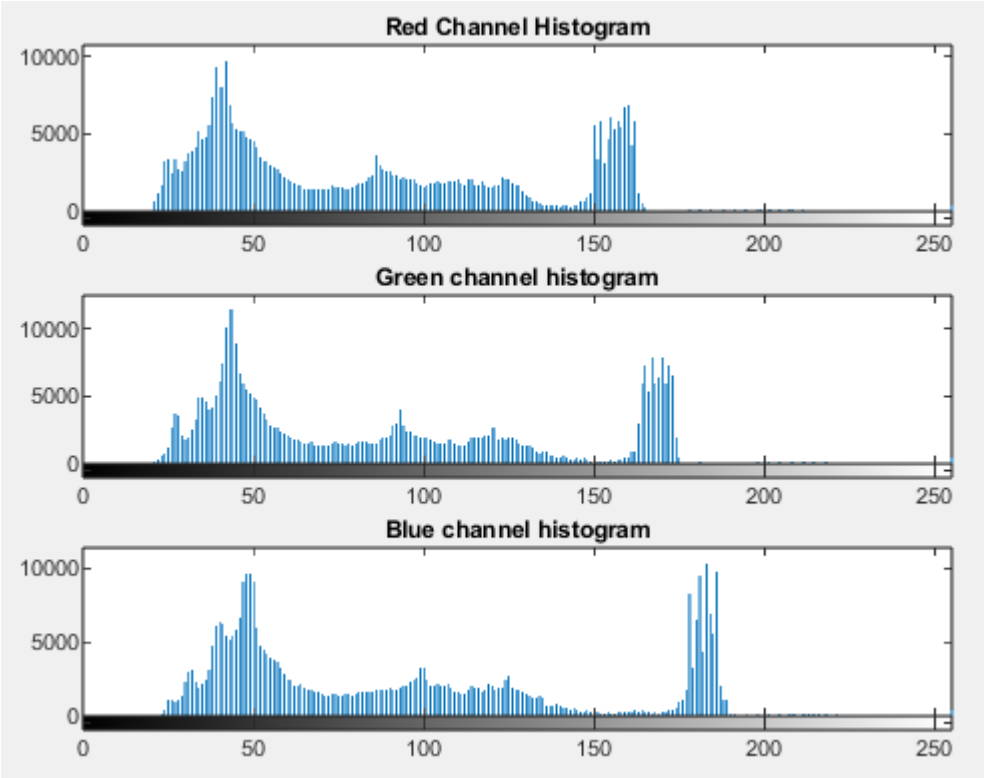


Figure 3.86: Histogram of 3.5 experimental result with 1 module

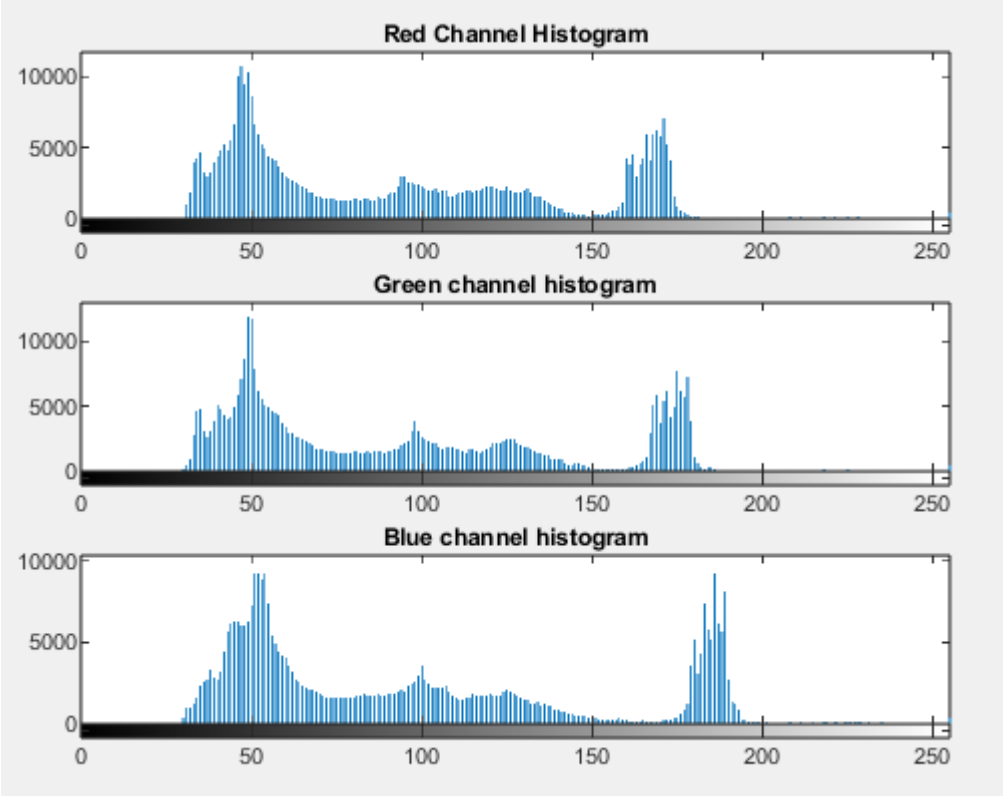


Figure 3.87: Histogram of 3.5 experimental result with 3 modules

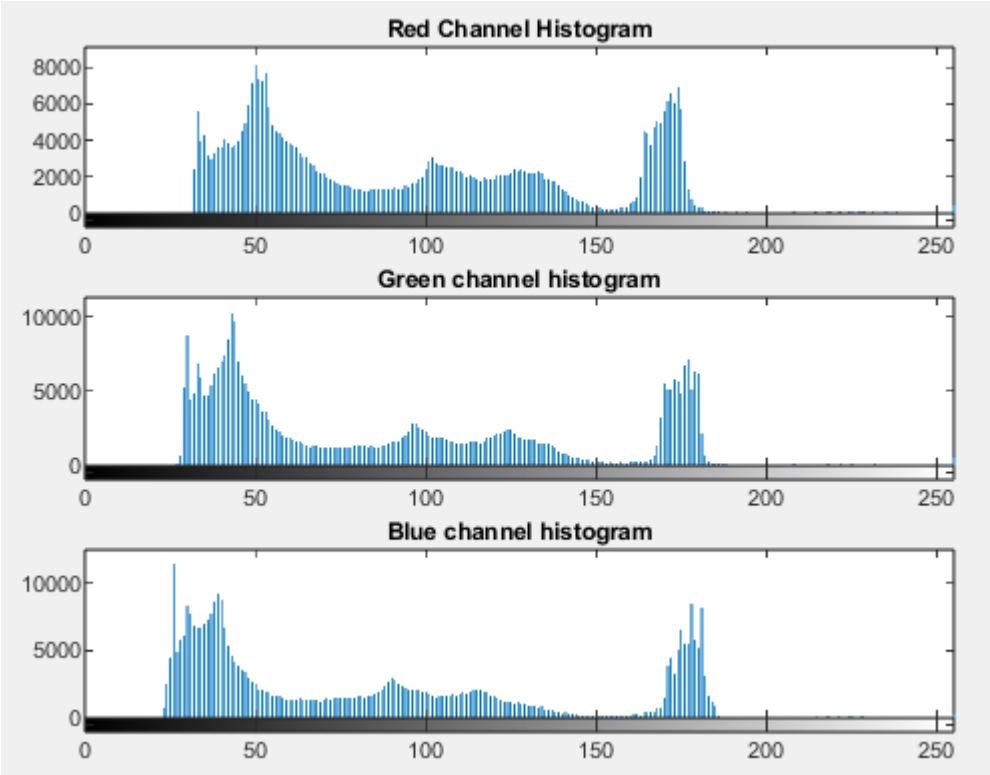


Figure 3.88: Histogram of 3.5 experimental result with 5 modules

For darkness level 3.5 again impressive results are obtained with the LL image being fully enhanced in all 3 cases of number of convolutional modules. This is also confirmed by the histograms, as all the three experimental histograms have a shape almost identical to the corresponding ground truth.

Darkness Level: 4.0

Original Low Light **Normal Light**

1 module **3 modules** **5 modules**

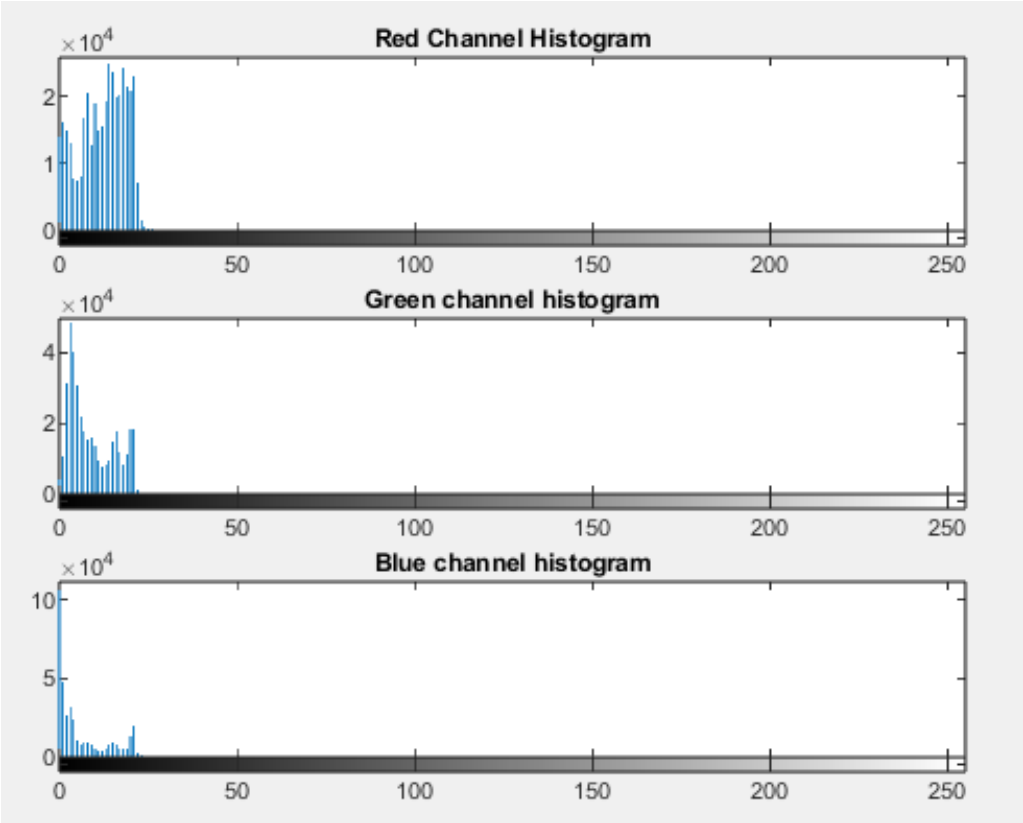


Figure 3.89: Histogram of 4.0 LL image

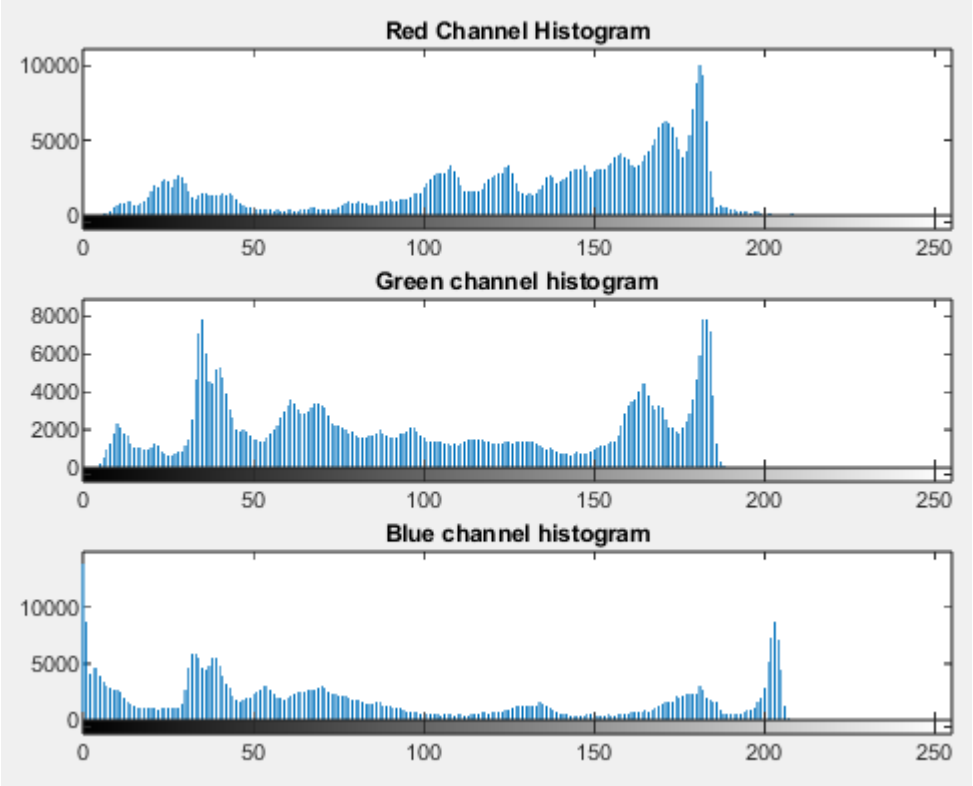


Figure 3.90: Histogram of 4.0 NL image

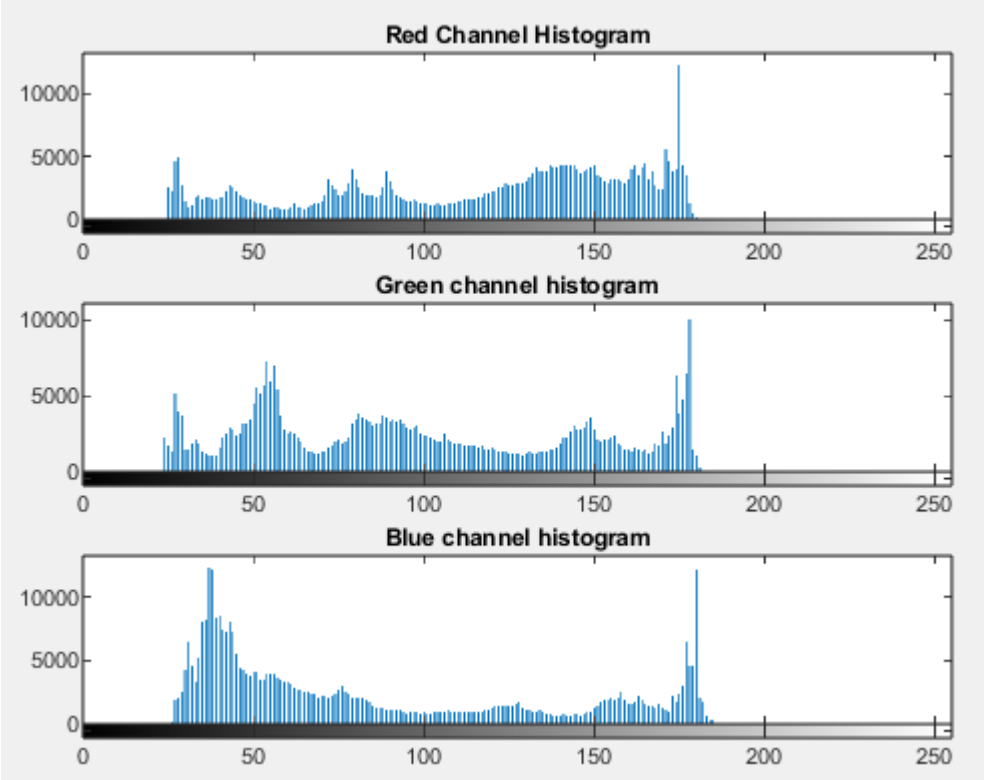


Figure 3.91: Histogram of 4.0 experimental result with 1 module

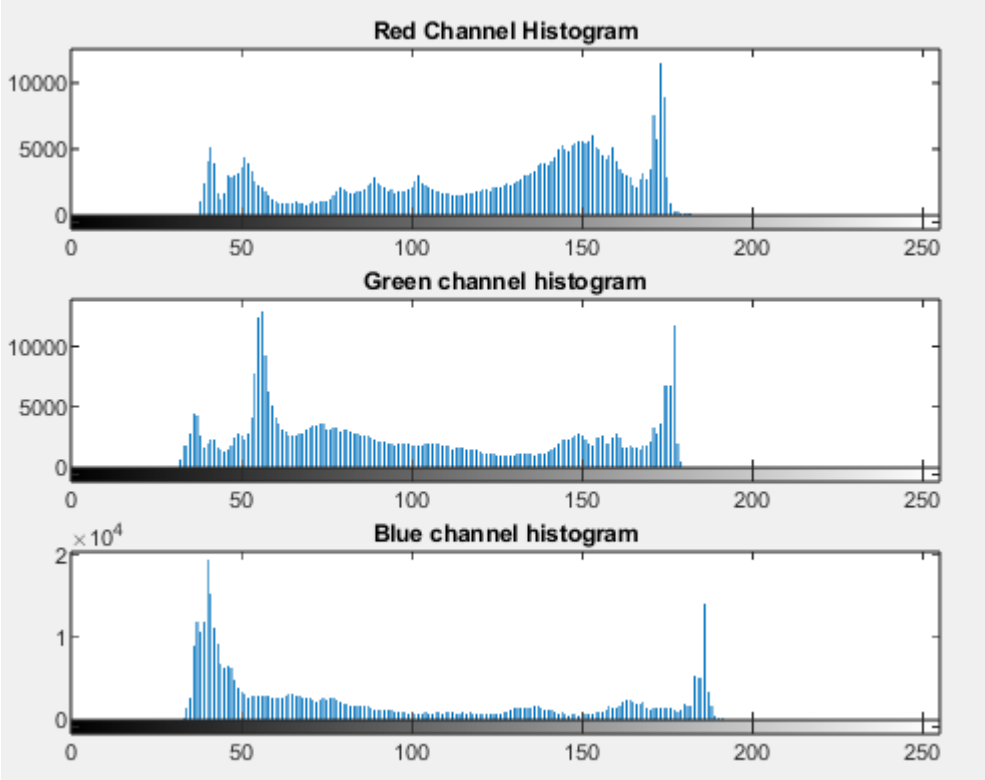


Figure 3.92: Histogram of 4.0 experimental result with 3 modules

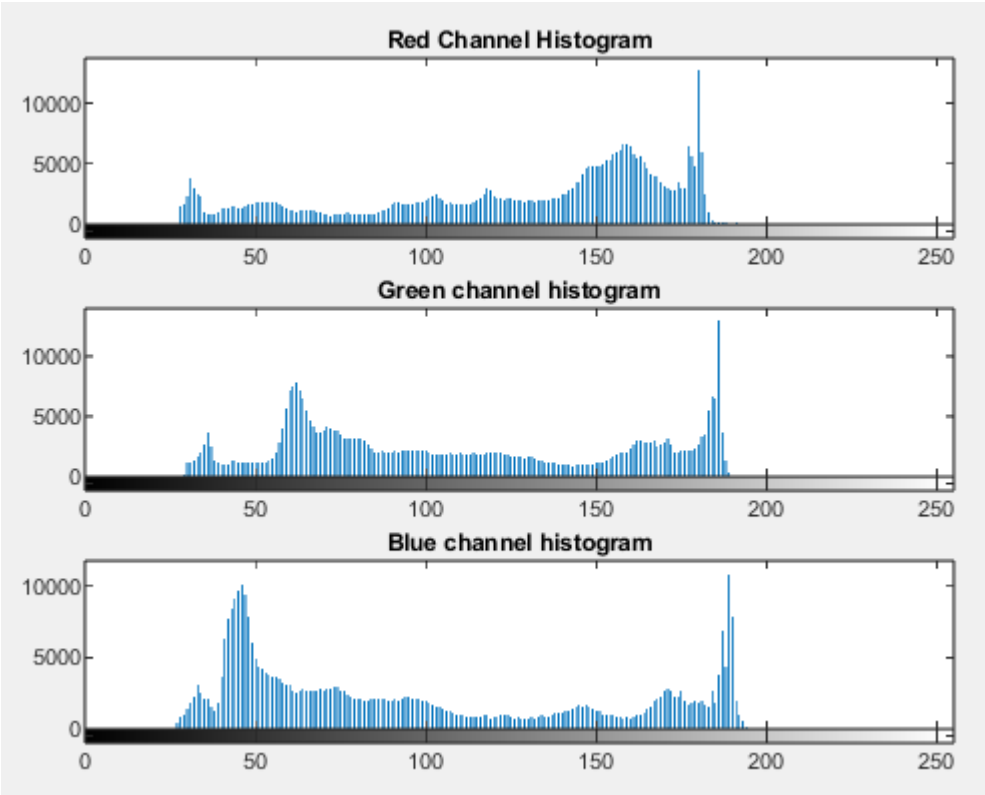


Figure 3.93: Histogram of 4.0 experimental result with 5 modules

For darkness level 4.0 we also see that the LL image is fully enhanced, for all three cases of number of convolutional modules, without the introduction of additional noise or color distortions. The experimental histograms confirm the quality of the results, and in all 3 cases, their shape is very close to the shape of the corresponding ground truth. What we can observe, however, is that the result for 5 convolutional modules is slightly better, as the color information has been restored to a greater extent. From this we understand that the difficulty has started to appear due to the increase in the darkness level, as we analyzed above.

Darkness Level: 4.5

Original Low Light **Normal Light**

1 module **3 modules** **5 modules**

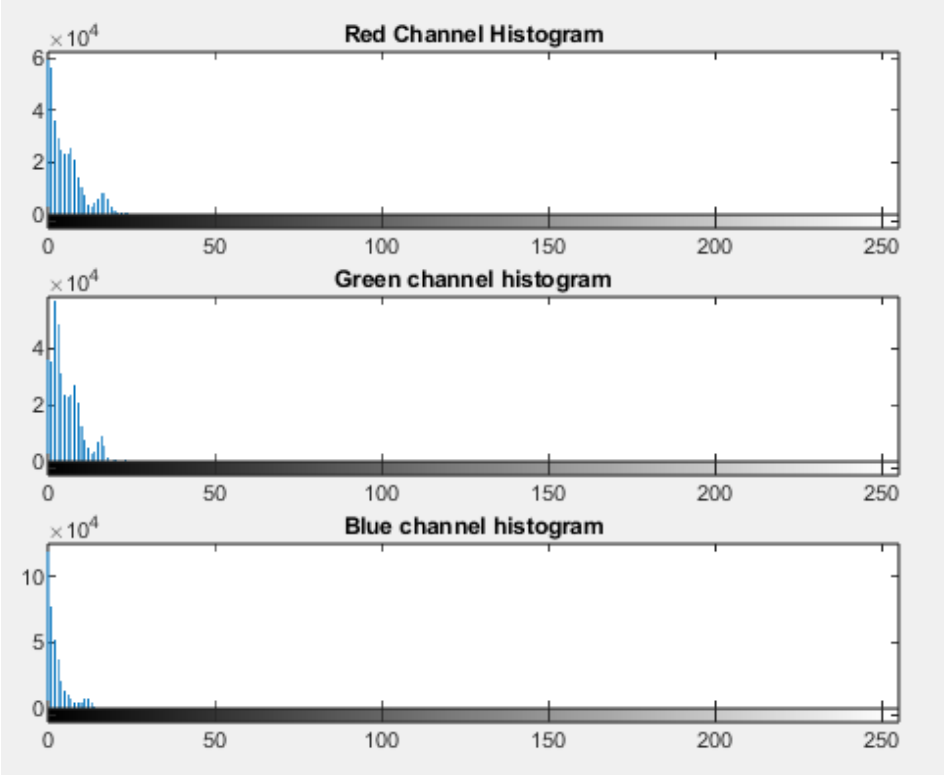


Figure 3.94: Histogram of 4.5 LL image

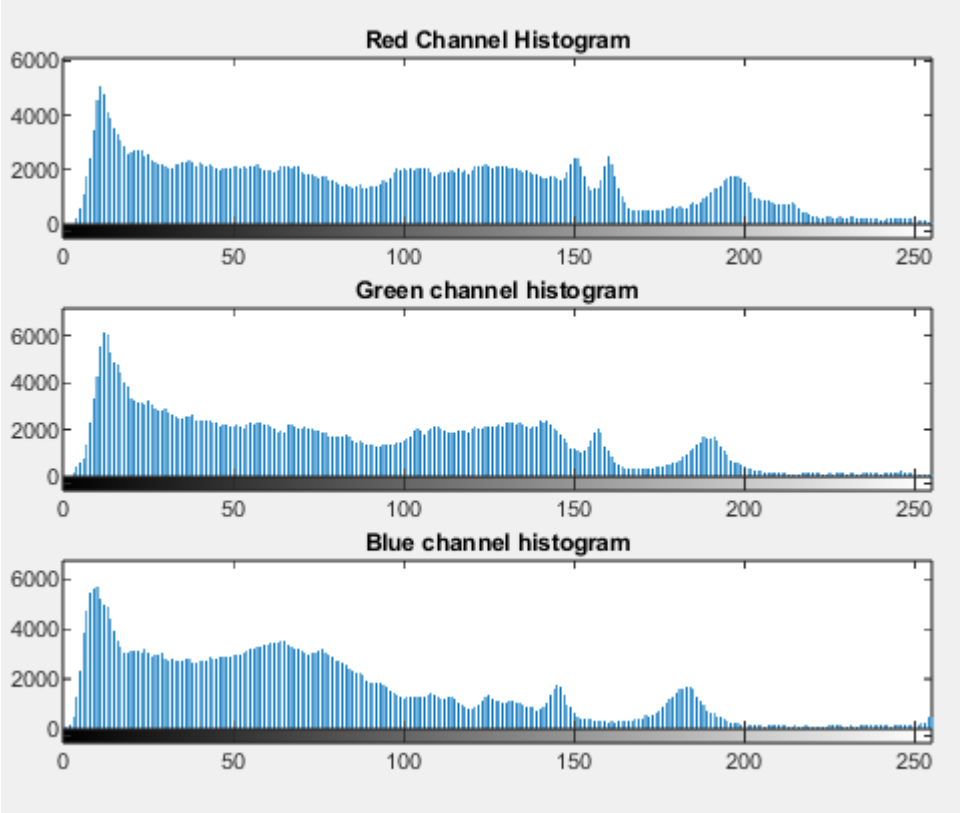


Figure 3.95: Histogram of 4.5 NL image

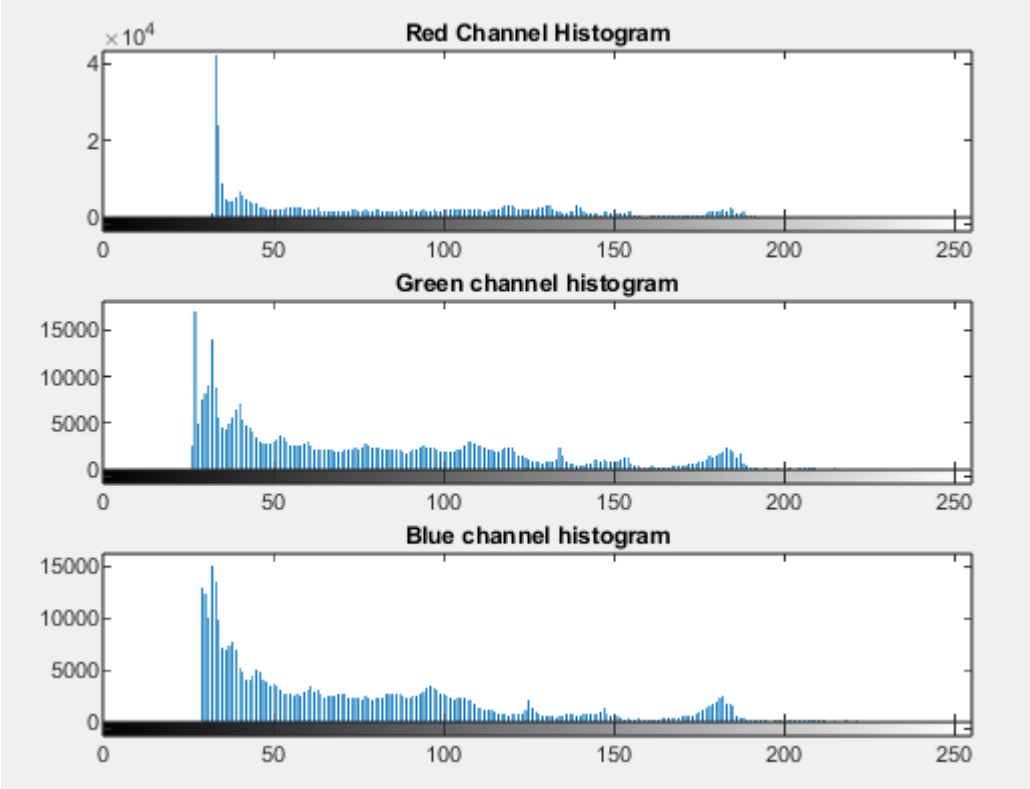


Figure 3.96: Histogram of 4.5 experimental result with 1 module

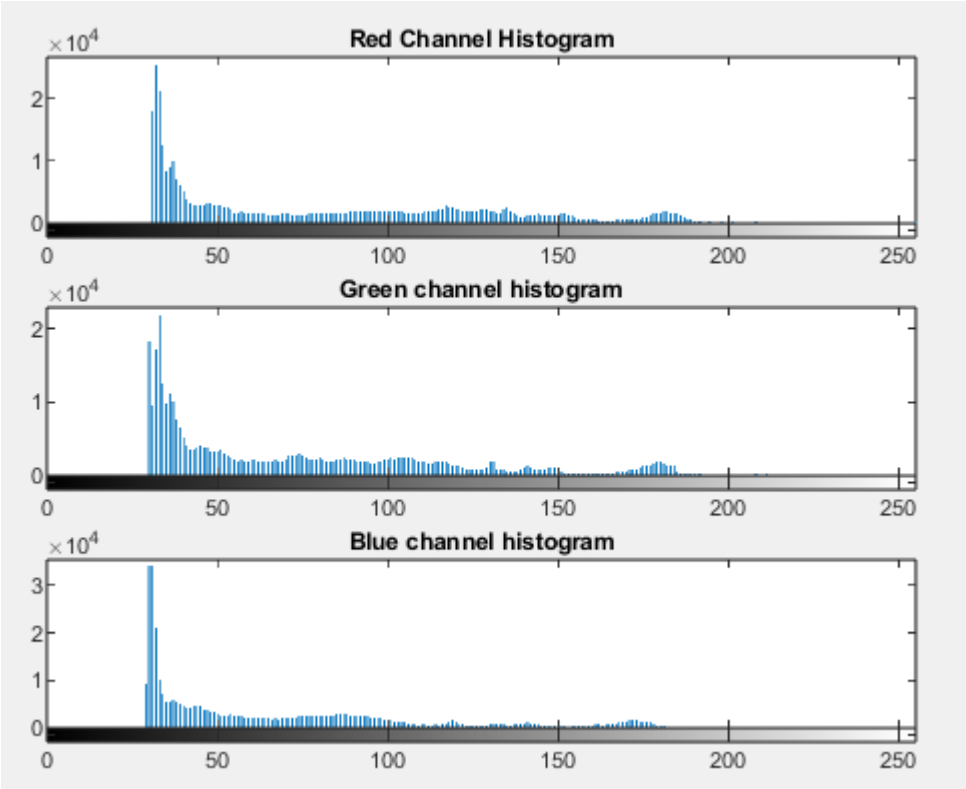


Figure 3.97: Histogram of 4.5 experimental result with 3 modules

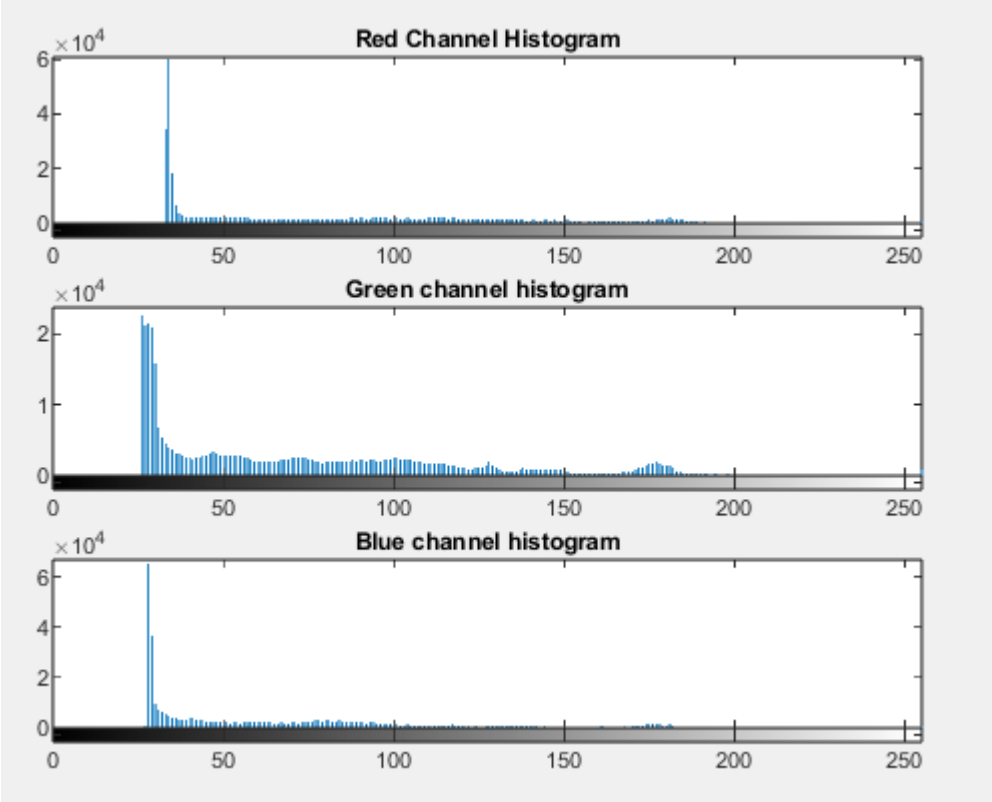
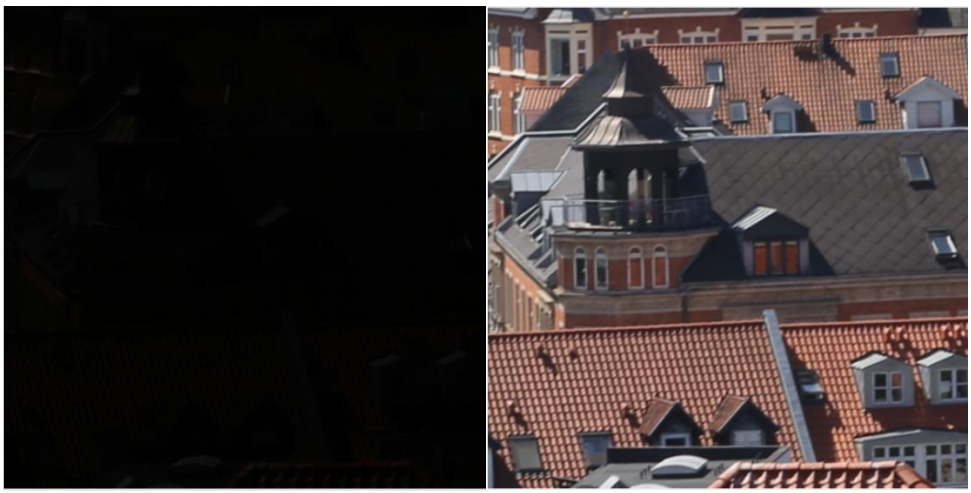


Figure 3.98: Histogram of 4.5 experimental result with 5 modules

For darkness level 4.5 we observe that the LL image is fully enhanced in all 3 cases of convolutional modules. Nevertheless, we see that color distortions appear, as the colors in the experimental results are not exactly the same as the corresponding ones in the ground truth case. This is due to the fact that the images have started to become very dark making it very difficult to retrieve the full visual information.

Darkness Level: 5.0



Original Low Light **Normal Light**

1 module **3 modules** **5 modules**

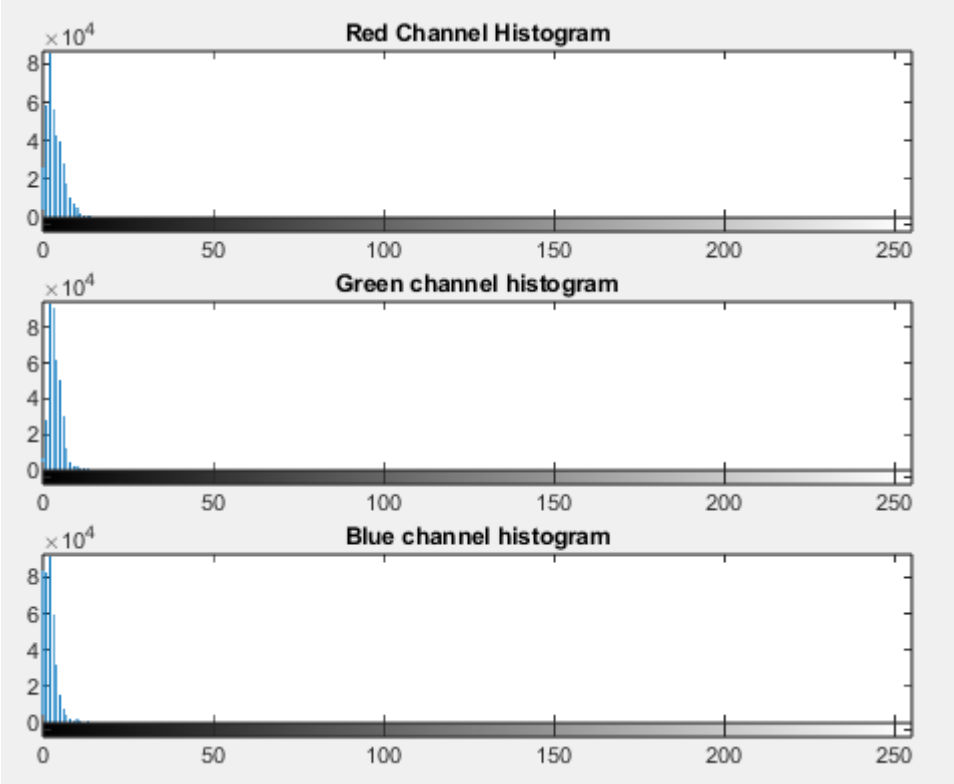


Figure 3.99: Histogram of 5.0 LL image

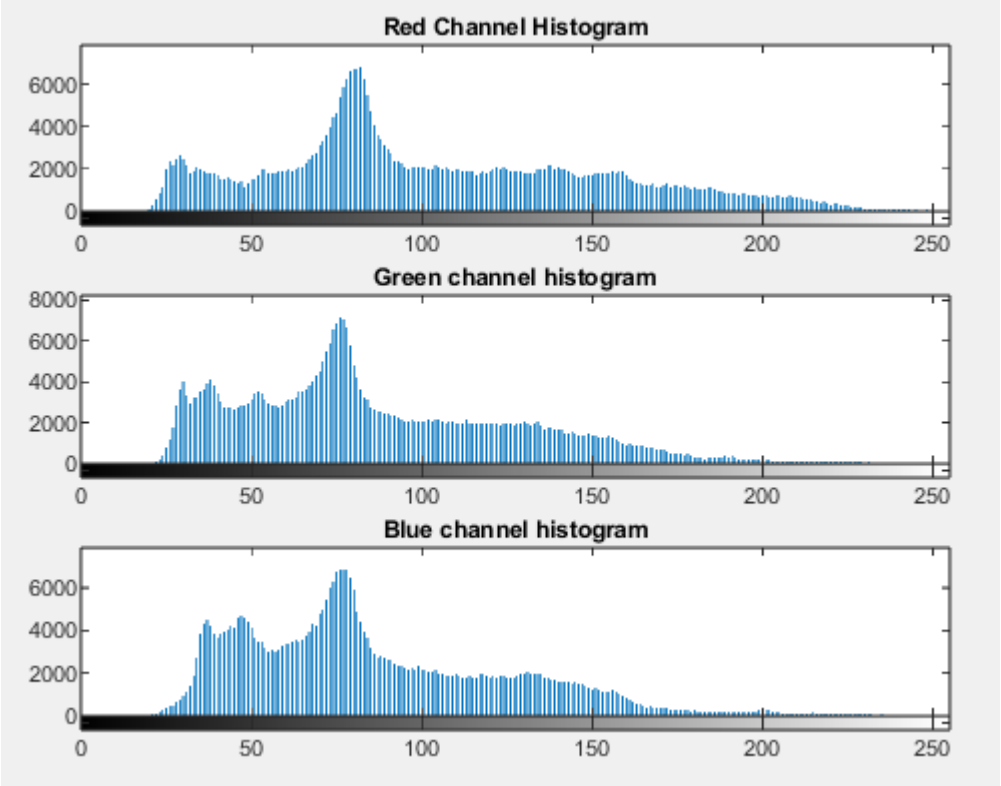


Figure 3.100: Histogram of 5.0 NL image

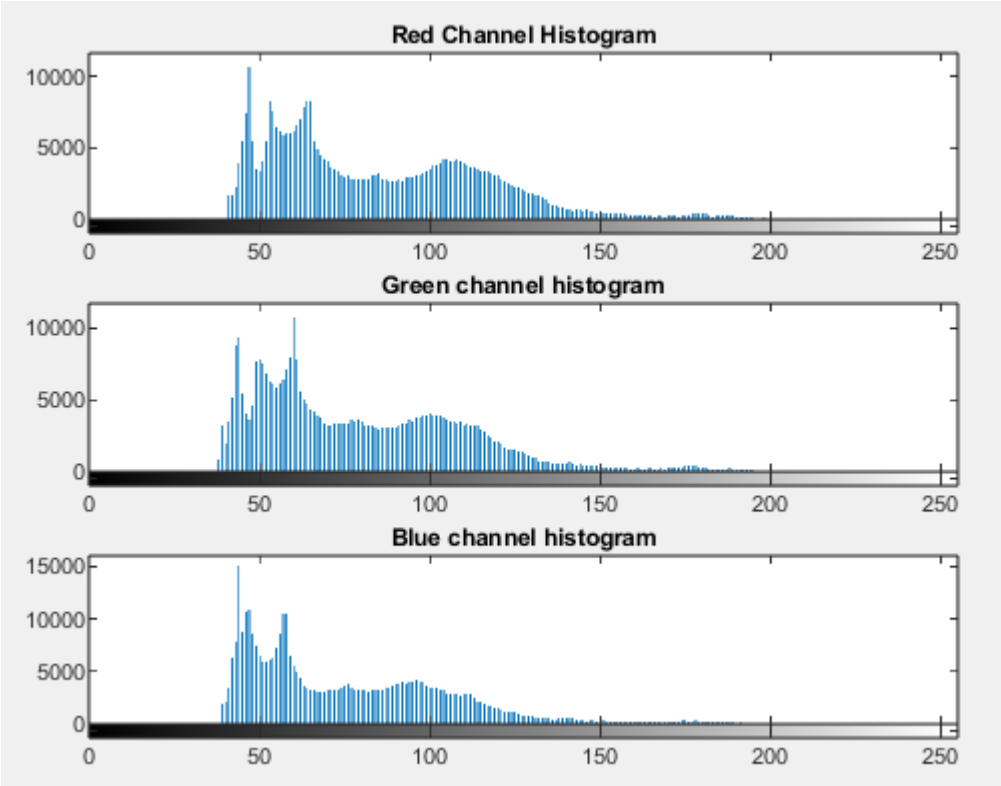


Figure 3.101: Histogram of 5.0 experimental result with 1 module

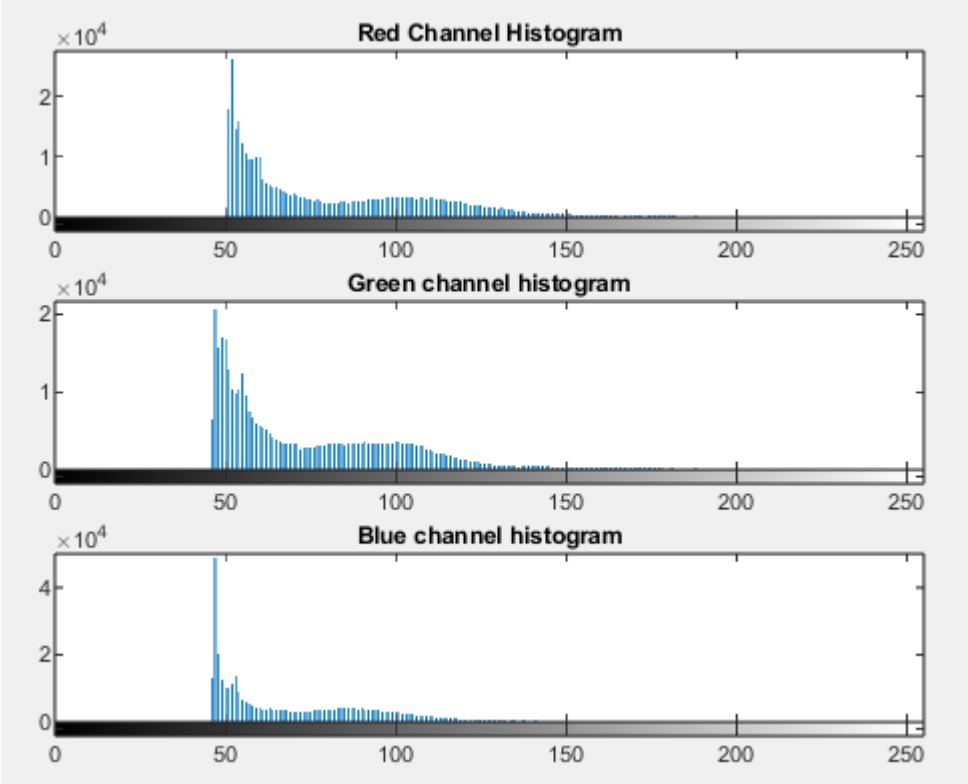


Figure 3.102: Histogram of 5.0 experimental result with 3 modules

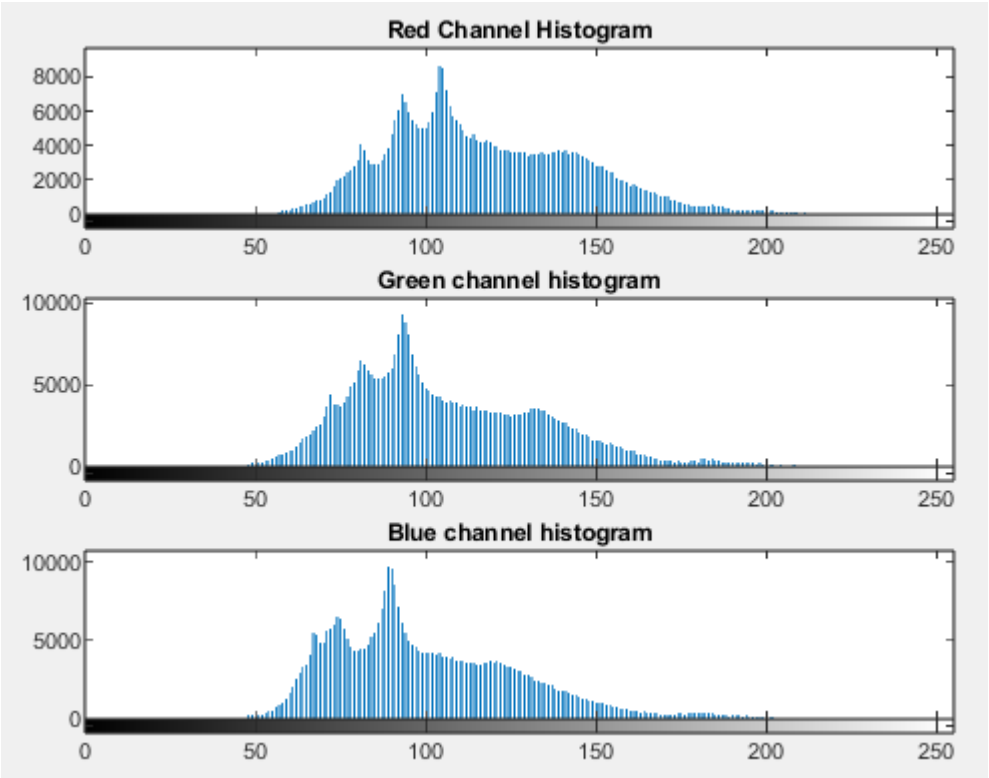


Figure 3.103: Histogram of 5.0 experimental result with 5 modules

For darkness level 5.0 we notice that the general texture and details of the ground truth image are restored, but not the color information. This also explains the decrease in the values of the metrics with the increase in the darkness level that we observed above. The images have now become too dark, resulting in the model being unable to learn the features needed to restore color, leading to the visual effects observed.

In general, we noticed that this particular method gave impressive results. At all darkness levels the metric values were greatly improved, and most of the visual information was recovered. It was observed that the number of convolutional modules does not strongly affect the result, and the values of the metrics remain more or less the same, and the visual results are similar to each other. Based on this we understand that we can simply choose the architecture with the fewest parameters since it will be easier and faster to train, and produce comparable results to the more complex architectures. In addition, we saw that with the increase of the darkness level, the model faced difficulties in retrieving the color information, since at high darkness levels the images are almost black and white. This is due to the fact that as the darkness level increases, the images become too dark, with the dynamic range being too small, making it difficult to learn features that can also restore color information. A solution to this problem would be to give the model more time to learn these features, that is, to train it for more epochs, so that it has time to learn the correct mapping from the LL space to the NL space. Nevertheless, the resulting images are very promising, as even for darkness level 5.0 where there is no color at all, the details of the ground truth image such as texture and edges have been fully restored.

Chapter 4 – Final Techniques Comparison

In this chapter we will make a more general comparison of the methods we developed in the previous two chapters. We will start from the classic methods, we will move on to the DL methods and finally we will compare them all with each other. The comparison will be based on how much the values of the quality metrics improve, as well as how complex each method is. At the end of the chapter, we will summarize our results and refer to future research that can be done in the field.

4.1 Classical Methods Comparison

During the second chapter we implemented and applied a set of classic image processing techniques to enhance LL images. We started with point processing methods, which included linear transformation, logarithmic and exponential transformation, and histogram equalization. Then we went a step further and applied more complex techniques, namely the SSR and MSR that are based on the Retinex theory, which we mentioned. Finally, we applied a method based on image dehazing, as it was observed that the negative of an LL image is as if the image was taken in foggy conditions. In all cases we have seen that satisfactory results are obtained, but there are also cases where the experimental result is characterized by additional noise and color distortions. In this section we will compare the classic methods with each other to see which one performs better. The comparison will be based on how much the values of the evaluation metrics improve. Specifically, we will use PSNR, SSIM and MV, displaying their average values per method and per darkness level, and we will also include the corresponding values for LL images to see how much they improve for each method. For methods where we experimented with multiple parameter values, we will only use the quality metric values of the parameter that gave the best results. Finally, we must comment that since we will compare with the DL methods as well, here the results we will display are based on the test set.

	Average PSNR(dB)				
Level	3.0	3.5	4.0	4.5	5.0
Original LL	9,390119	8,80953	8,408226	8,14007	8,048578
Linear Transformation	13,86573	13,6508	13,4439	13,22458	12,8527
Gamma (0.3)	17,97308	17,85597	16,96888	15,65196	14,34958
Log (50)	14,82221	19,0108	20,9166	17,57403	14,59449
HE	14,3184	14,16896	13,96122	13,59259	13,09024
SSR (400)	15,29788	15,72012	16,08146	16,18764	15,9054
MSR	13,8474	14,0304	14,26856	14,49206	14,52812
Dehaze	17,18743	17,47945	17,13891	16,00111	14,42391

Table 4.1: Average PSNR per method

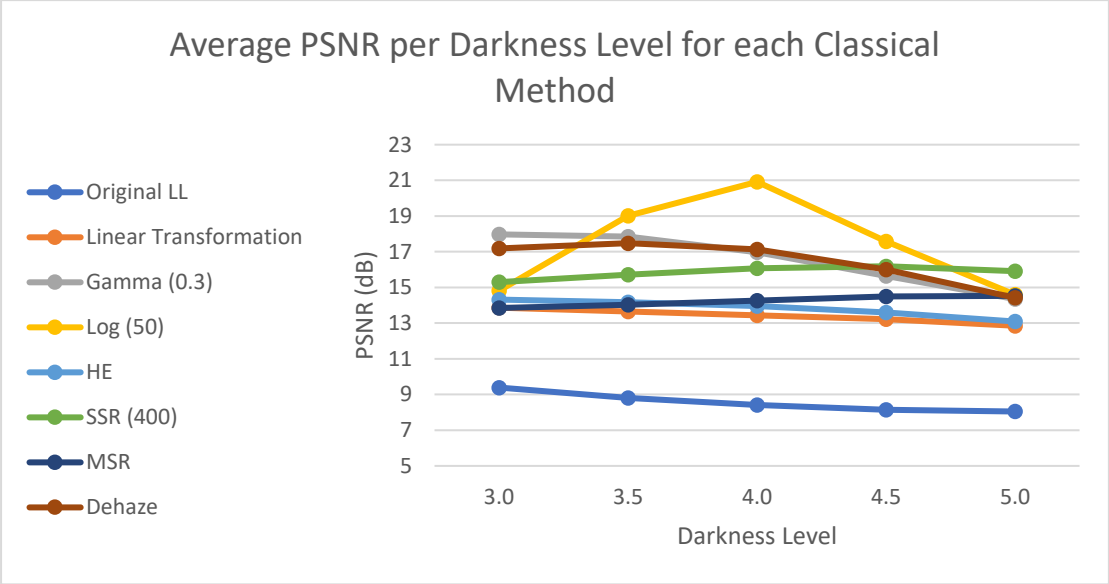


Figure 4.1: Average PSNR per darkness level per method

We see that the average PSNR improves noticeably in all cases. The linear transformation performs the least, compared to the other cases, increasing the average PSNR by only 4.85dB, which is to be expected as it is the simplest method of all, applying a simple linear function. The next two worst performing methods are histogram equalization which increases the average PSNR by 5.27dB, on average, and MSR which increases the average PSNR by 5.67Db. The histogram equalization method performs less well than the other methods due to its simplicity, as it is a simple transformation based on the cumulative probability function. In addition, we saw that there are several ground truth images, whose histogram is not spread evenly over the entire available range. Histogram equalization by its nature aims for the final image to be characterized by a histogram with

pixel values evenly distributed, so we understand that it cannot reproduce the results we mentioned above, leading to the reduced performance observed. Regarding the MSR, we described in the second chapter that it is the combination of 3 different SSRs with parameters 10, 120 and 400. When we were studying the SSR we saw that for a parameter of 10 intense noise and color distortions are introduced into the experimental result. Because of this, the corresponding result in MSR is also affected, causing the reduced performance, compared to the rest of the algorithms. Then above follow the SSR with parameter 400, the method based on image dehazing, and the gamma transformation with parameter 0.3, which increase the average PSNR by 7.28dB, 7.89dB and 8dB respectively. The most efficient method appears to be the logarithmic transformation, which increases the average PSNR by 8.82dB. We see that the 2 most efficient methods are pointwise non-linear transformations. The reason they give better overall results is because they are selected in such a way that they only enhance the dark areas of an image and not the entire image. Because of this, at low darkness levels they give very good results, and indeed they are the most efficient methods, but at high darkness levels, where the images consist only of dark areas, their performance drops, as can be seen from figure 4.1.

A general behavior we observe is that as the darkness level increases, the average experimental PSNR decreases. An exception to this are the SSR and MSR algorithms, which follow an increasing path with the increase of the darkness level. For the remaining methods, the decrease in PSNR is due to the fact that the images become very dark, with the dynamic range becoming smaller and smaller, making it very difficult to retrieve visual information. The SSR and MSR algorithms are an exception because their ultimate purpose is not simply to retrieve visual information, but the reflectance coefficient of the scene. This information is independent of the brightness level of the scene, and for this reason the performance of the methods is not affected by increasing the darkness level. Moreover, we see that with increasing darkness level the performance of methods based on point transformations decreases, and for large darkness levels algorithms based on spatial filters are much more efficient. For these high darkness levels the images are too dark, as we mentioned above, with the result that a simple point transformation that looks at the value of only one pixel is not enough to fully improve the image. This explains the fact that more complex algorithms are more efficient for large darkness levels.

	Average SSIM				
Level	3.0	3.5	4.0	4.5	5.0
Original LL	0,178076	0,11419	0,072755	0,04675	0,031035
Linear Transformation	0,462487	0,417892	0,366401	0,316112	0,246587
Gamma (0.3)	0,622504	0,581671	0,524011	0,458485	0,371508
Log (50)	0,610923	0,61743	0,581146	0,502621	0,385608
HE	0,43675	0,3975	0,352682	0,299397	0,251513
SSR (400)	0,567147	0,558782	0,521125	0,469443	0,392017
MSR	0,494448	0,494892	0,465544	0,42746	0,359022
Dehaze	0,594288	0,561553	0,5086	0,438122	0,337125

Table 4.2: Average SSIM per method

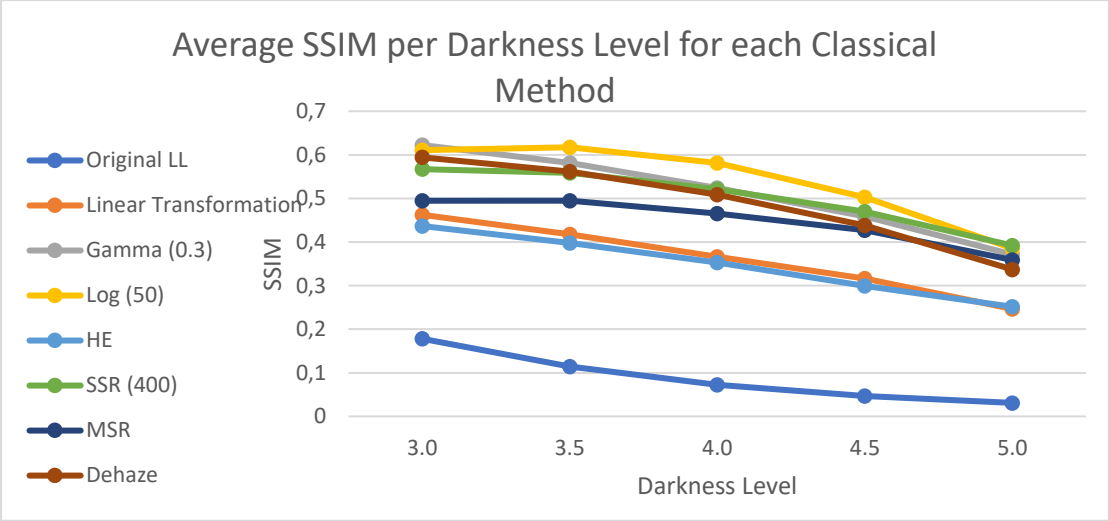


Figure 4.2: Average SSIM per darkness level per method

From the SSIM values we see that the two least efficient methods are histogram equalization and linear transformation, which increase the average SSIM by 0.26 and 0.27 respectively, same thing we also observed by studying the PSNR. This is followed by MSR, the method based on dehazing, and SSR with a parameter of 400, which increase SSIM by 0.36, 0.4, and 0.41, respectively. The most efficient methods appear to be the gamma transformation with a parameter of 0.3, which increases the mean SSIM by 0.42, and the logarithmic transformation with a parameter of 50, which increases the mean SSIM by 0.45. Here again, as in the case of PSNR, we see that the simplest methods are also the least efficient. The simplicity of the function we apply to each pixel is not sufficient to fully recover the visual information, leading to the reduced performance observed. Furthermore, the algorithms that increase SSIM the most are

non-linear transformations, as in the case of PSNR. However, we notice that with an increase in the darkness level, the performance of these algorithms decreases faster, compared to the other methods.

By increasing the darkness level, the performance of all algorithms decreases, with the exception of MSR and SSR with parameter 400, where this decrease is less noticeable. As we mentioned above, the purpose of MSR and SSR is to recover the reflectance coefficient of the scene, and not to directly enhance the image, which is independent of the amount of brightness of the scene, thus explaining the minimal dependence of the final result on the darkness level. Also, we see that the non-linear methods give very good results for small darkness levels, but at large levels their performance is greatly reduced, which is due to the fact that the images have become very dark, resulting in the fact that a mathematical operation on a single pixel is not enough, and we need for information from neighboring pixels as well, something that the other methods (MSR, SSR, dehazing) do.

	Average MV				
Level	3.0	3.5	4.0	4.5	5.0
Original LL	16,79939	11,84197	8,242994	5,659169	3,887245
Linear Transformation	54,34299	53,33926	52,16029	49,44718	47,2227
Gamma (0.3)	102,7424	91,26644	80,18108	69,3653	59,60535
Log (50)	137,4292	113,1232	90,4949	70,26011	53,44228
HE	127,9495	128,6499	129,8455	132,1473	134,6972
SSR (400)	130,7205	125,9893	119,6509	112,4539	102,4706
MSR	134,7028	132,3205	128,5233	123,5492	116,8328
Dehaze	106,3223	99,77421	90,64965	78,97358	68,62438

Table 4.3: Average MV per method

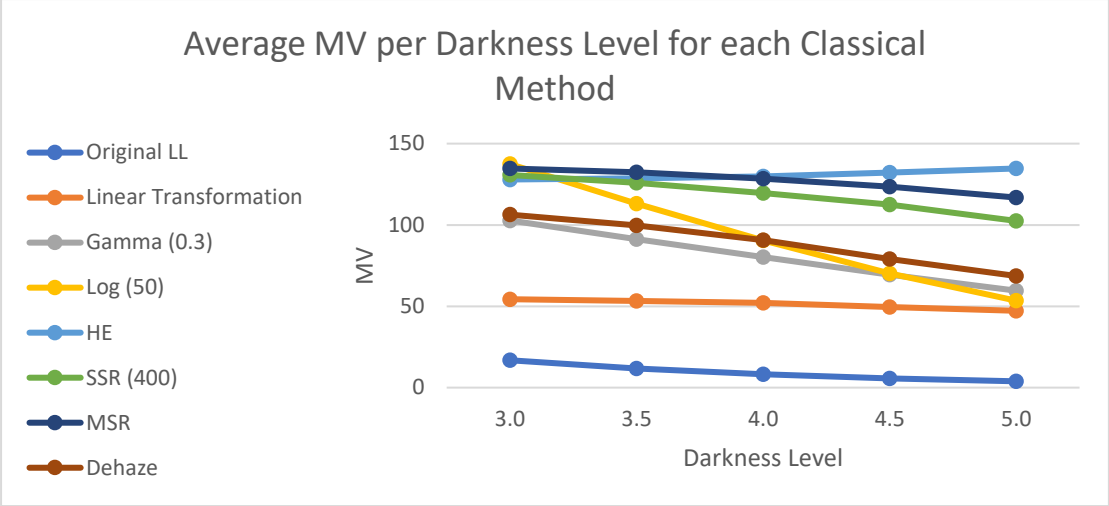


Figure 4.3: Average MV per darkness level per method

Studying the average brightness, we see that the linear transformation remains the least efficient method, as it only increases the average brightness by 42.01. The order of performance of the remaining algorithms changes, compared to what we saw before, as now the most efficient method for increasing brightness is histogram equalization, which increases the average brightness by 121.37. This is expected due to the nature of the method, which aims for the experimental result to be characterized by pixel values that are uniformly distributed throughout the available value range. This results in images with greater contrast, which also leads to greater brightness. MSR and SSR with a parameter of 400 follow immediately after, increasing the average brightness by 117.9 and 108.97 respectively. Less efficient are the logarithmic transformation, the dehazing-based method, and the gamma transformation, which increase the average brightness by 83.66, 79.58, and 71.34, respectively. The decrease in performance with the increase in darkness level is observed only in the logarithmic transformation and the gamma transformation, while in the other cases the increase in brightness remains almost constant.

In general, from all the above analysis, we saw that the linear transformation is the least efficient method of enhancing LL images, compared to the other methods, which is due to the simplicity of the mathematical expression we apply to the pixel values. In addition, we saw that non-linear transformations give the greatest increase in metrics with reference (PSNR and SSIM), but their performance drops very quickly with the increase in darkness level, which combined with the very small increase in average brightness, makes them unreliable algorithms. It was also observed that histogram equalization, although it produces the greatest

increase in average brightness, the results it produces do not recover all the information needed for the ground truth case. This is mainly due to the nature of the algorithm and dataset, as the dataset contains several ground truth images whose pixel values are not uniformly distributed over the entire available value range. Histogram equalization aims for the final result to be characterized by pixel values evenly distributed throughout the histogram, so we understand that it is impossible to reproduce many of the data set results. This explains the reduced performance in the quality metrics with reference, while it is the method that gives the greatest increase in average brightness. Based on this we understand that histogram equalization could be used as a general method of increasing the brightness of an image, but it depends on the data set we are studying, as it seems that in RELLISUR it is not efficient enough. Then we saw that the method based on dehazing gives moderate results but has a relatively stable performance with the increase of the darkness level. Something similar was observed for MSR and SSR which produce moderate results, but have stable performance with increasing darkness level. As we described above, this is due to the fact that their purpose is to recover the reflectivity of the scene, and not to directly enhance the image, information that is independent of the amount and distribution of brightness in the image. In chapter 2, however, we saw that these two methods introduce additional noise and intense color distortions in the experimental result, something that does not appear in the dehazing method, which produces impressive visual results even at high darkness levels. From all this we understand that, of the classical methods, the best one for the solution of LL is the one based on dehazing, as it noticeably increases the values of the metrics, its performance does not depend strongly on the darkness level, and produces impressive visual results, with the experimental images being very close to the ground truth case, having recovered most of the visual information. Obviously, there is also a lot of room for improvement as we can also improve the fog removal model, making the algorithm even more efficient.

4.2 Deep Learning Methods Comparison

In this section we will compare the Deep Learning methods we developed during the third chapter. In chapter 3 we used the LL-CNN architecture [36], where we first implemented it exactly as described in the related article by the authors. Then we made 2 variations, where in the first we introduced additional convolutional layers, which acted as post-processing filters, while in the second variation we trained the network based on the difference of LL from the ground truth image. In all cases we experimented with a different number of convolutional modules, and in particular we experimented with 1, 3 and 5 modules. In the first two cases we saw that the 3 convolutional modules perform better, so we will use the metric values for this case only. In the last variation we saw that there is no great differentiation of the results, but the case with 1 convolutional module, performs a little better at large darkness levels. Because of this, and the fact that it gives the same results while being less complex, we will use the 1 module case in our analysis.

	Average PSNR(dB)				
Level	3.0	3.5	4.0	4.5	5.0
Original LL	9,390119	8,80953	8,408226	8,14007	8,048578
LLCNN (3)	21,80508	21,04742	21,32991	18,20765	18,90393
LLCNN++ (3)	21,50457	22,46352	21,94701	21,39252	19,6565
LLCNN+DO (1)	22,57655	22,50693	21,55051	20,81169	19,35751

Table 4.4: Average PSNR per DL method

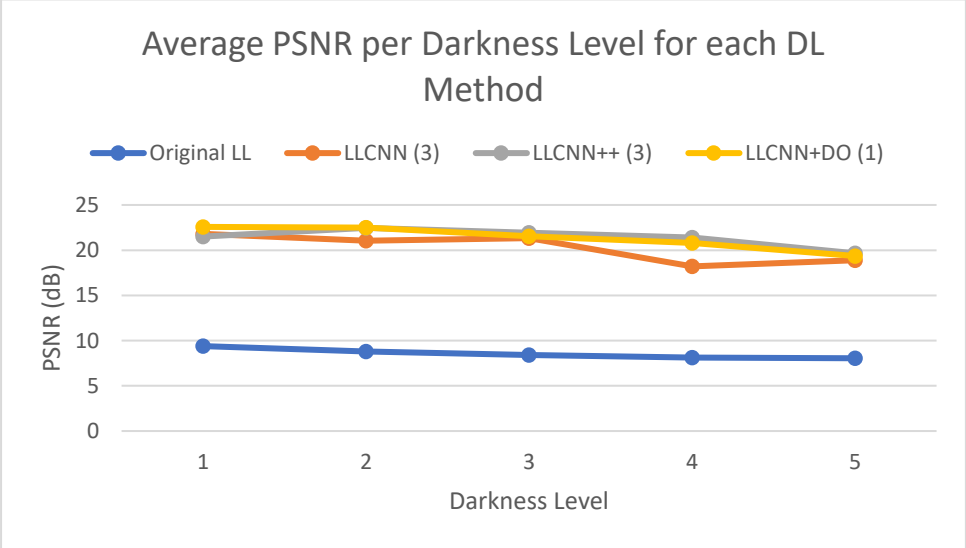


Figure 4.4: Average PSNR per darkness level per DL method

Regarding the average experimental PSNR we see from figure 4.4 that there are no strong differences. LLCNN++ with 3 modules increases the average PSNR by 12.83dB, followed by the second variant which increases the average PSNR by 12.8dB, and finally it is the original architecture which increases the PSNR by 11.7dB, confirming that the differences are minimal. Furthermore, we notice that increasing the darkness level does not affect the result much, as the change of the values of the metrics is small from level to level. At small darkness levels, all three architectures have almost the same performance, but at larger ones we see that the original architecture lags behind, and the two variations we implemented prevail.

	Average SSIM				
Level	3.0	3.5	4.0	4.5	5.0
Original LL	0,178076	0,11419	0,072755	0,04675	0,031035
LLCNN (3)	0,695826	0,690745	0,607084	0,498138	0,413824
LLCNN++ (3)	0,715544	0,685708	0,676907	0,623833	0,435932
LLCNN+DO (1)	0,70848	0,719564	0,66433	0,593542	0,457478

Table 4.5: Average SSIM per DL method

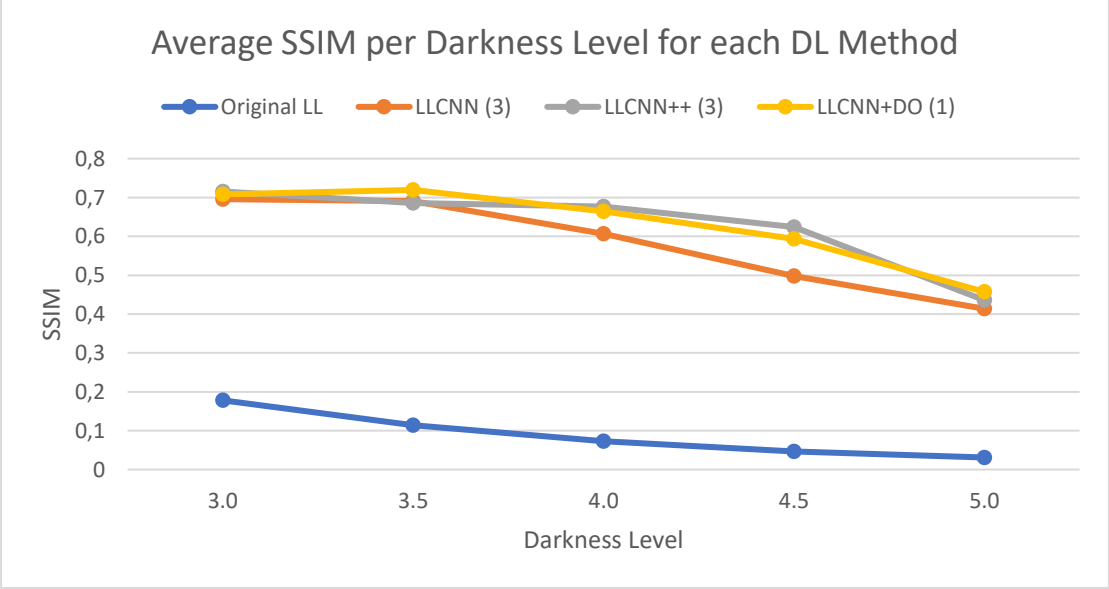


Figure 4.5: Average SSIM per darkness level per DL method

Similar conclusions can be reached by studying the average value of SSIM. From figure 4.5 we see that again there are no strong differences in the performance of the architectures, which is also confirmed by the values of

the average SSIM increase. The original architecture increases the average SSIM by 0.49, while the first and second variants both increase it by 0.54. In this case, performance decreases with increasing darkness level, which is more pronounced in the original architecture. The two variants we implemented have almost the same performance at each darkness level. We find that the second variant has the advantage, as it produces equally good results using fewer parameters.

	Average MV				
Level	3.0	3.5	4.0	4.5	5.0
Original LL	16,79939	11,84197	8,242994	5,659169	3,887245
LLCNN (3)	86,52921	84,36512	98,0996	112,9711	97,62692
LLCNN++ (3)	105,6586	94,97591	99,04177	95,81691	99,04177
LLCNN+DO (1)	95,69192	91,97912	90,66982	91,22758	88,1989

Table 4.6: Average MV per DL method

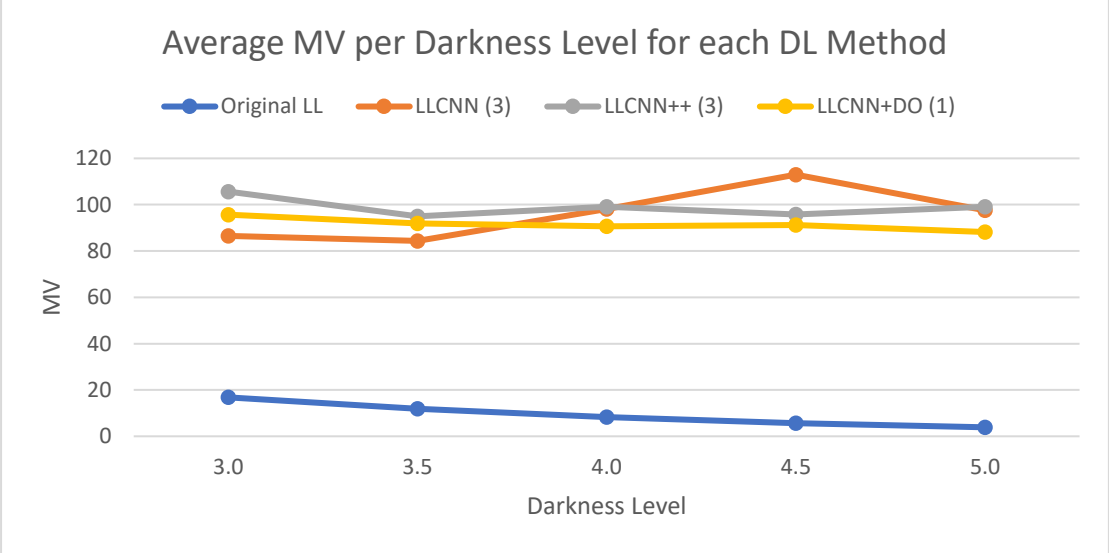


Figure 4.6: Average MV per darkness level per DL method

In the case of the average brightness, we again observe that there are no strong differences. The original architecture increases the average brightness by 86.63, and the first and second variants increase the brightness by 89.62 and 82.27 respectively. In addition, we observe that the increase of the darkness level does not affect the result, since the change of the values of the metric from level to level is small for all three cases of architectures.

Overall, we saw that all three cases of architectures have a comparable performance, as the values of the metrics of interest are not characterized by sharp variations. If we look more closely at the results, however, we can notice that at high darkness levels the original architecture performs less well compared to the two variants. The lower performance is due to the fact that this architecture learns fewer features, as it has fewer convolutional layers than the other two architectures, and also because it is trained for fewer epochs. As for the other two architectures, we noticed that at all darkness levels they perform almost the same, with the first variant being slightly more efficient at higher darkness levels. Nevertheless, the second variant has the advantage that it consists of fewer convolutional modules (i.e. fewer parameters) and for the same training conditions produces the same qualitative results as the first variant. The reason it performs better with fewer parameters is because it only has to learn the distribution extracted from the ground truth case, rather than the entire mapping, making the model's work easier. Based on all the above analysis we understand that the best architecture we could use is the second variant, as with fewer parameters it can produce the same quality results, and it is easier to train. Obviously there is also a lot of room for improvement, as one could further optimize the architecture with hyperparameter tuning, more training, and using patches rather than the whole image, procedures that were not done in the present work due to limited resources.

4.3 Final Comparison

In this subsection we will compare all the methods against each other, based on how well they improve the evaluation metrics we are interested in, as well as on how complex they are. The values of the metrics are shown in the tables of the previous two subsections, so here we will only show the line charts for visualization purposes.

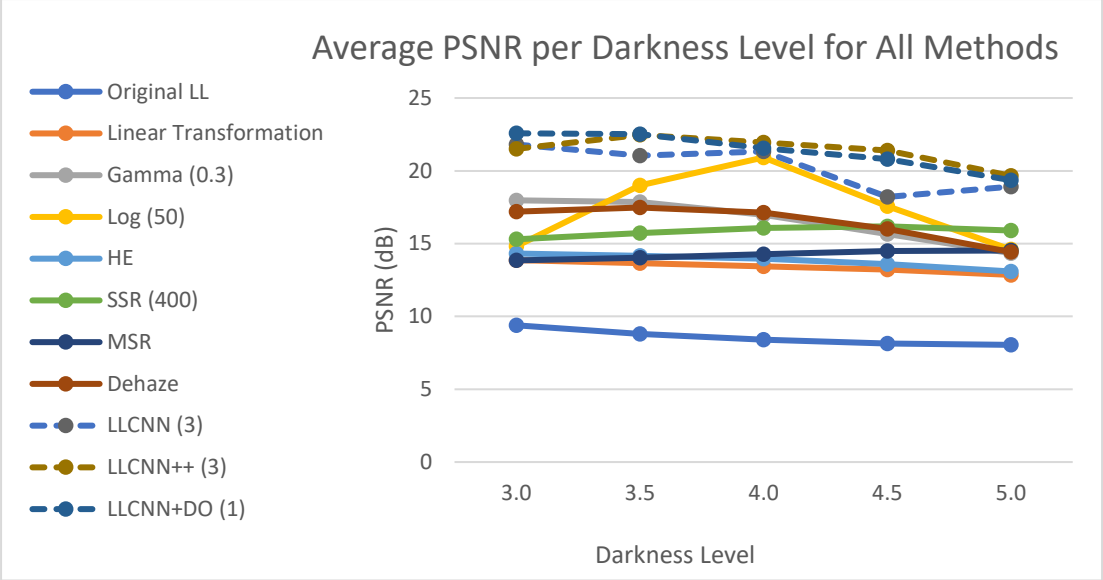


Figure 4.7: Average PSNR per darkness level for all LLIE methods

We see that all DL architectures outperform all classical algorithms, indicating that in general, techniques with learning are superior to classical techniques. In addition to performing better, we saw that in classical techniques we had to choose the value of a set of parameters, showing here only the cases of parameters that give the best results. However, we don't know if there is another better value of these parameters, and experimenting to find it would be time-consuming. Models based on learning avoid this search, as they use information extracted from the data to correctly model the relationship between LL space and NL space.

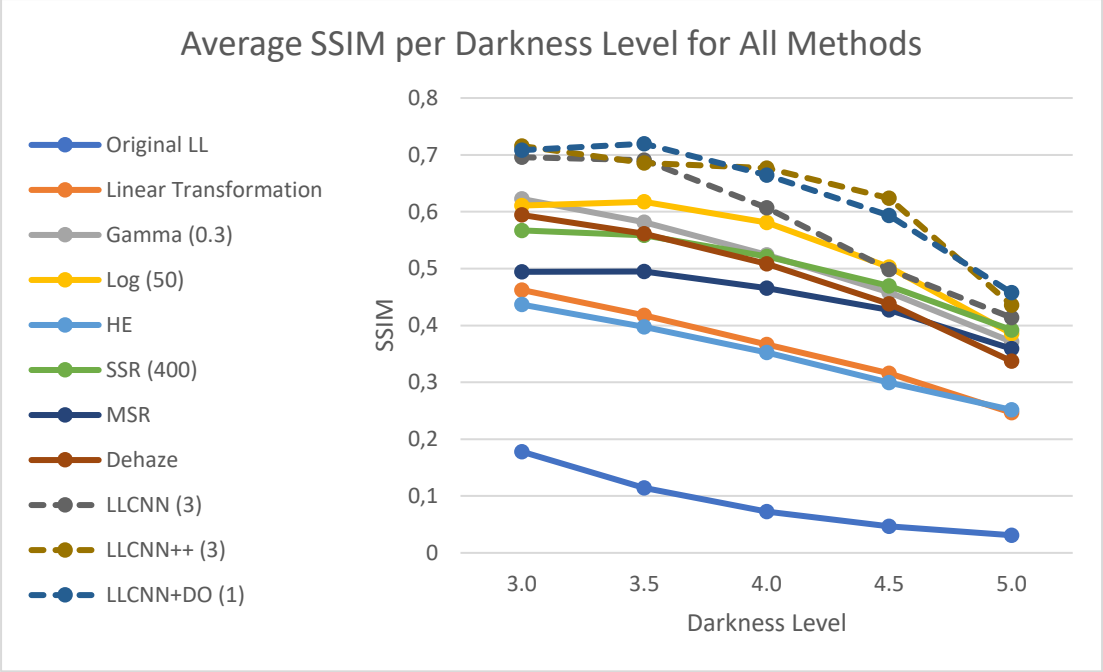


Figure 4.8: Average SSIM per darkness level for all LLIE methods

We reach the same conclusion by studying SSIM, as from figure 4.8 we see that for all darkness levels the Deep Learning methods perform better than the classic methods, giving higher values of the specific quality metric. The DL techniques we applied learn many low level features that help them recover local information (texture etc.), something that is not done in classical algorithms, thus explaining the increased performance of DL techniques.

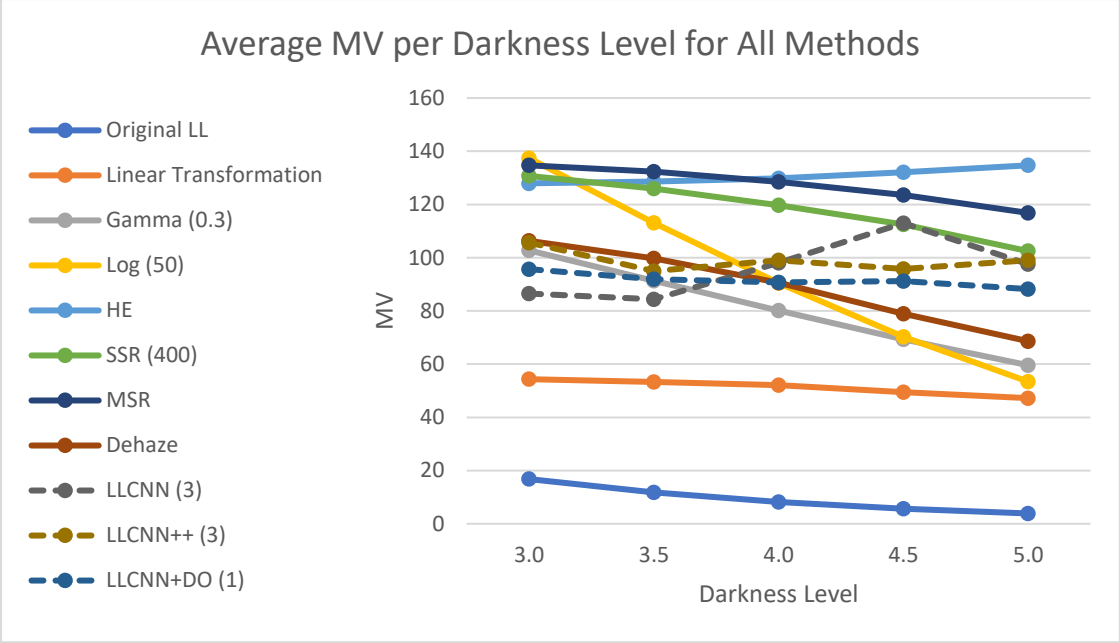


Figure 4.9: Average MV per darkness level for all LLIE methods

In the case of MV we see that the order changes as now the DL techniques give an average improvement, while the classic techniques of histogram equalization and Retinex Based methods give a greater increase in the average brightness for each darkness level. The reason this happens is that the specific classical techniques aim to produce an image with high contrast, the histogram equalization by spreading the pixel values across the available range, and the Retinex based methods by recovering the reflectivity of the scene. But we saw that the ground truth data set also consists of images that do not have high contrast and pixel values distributed throughout the available range, which have been experimentally over-enhanced, as we also saw in chapter 2. DL techniques aim to approximate the appropriate mapping between the LL set that we give as input and the NL set that we give as output. Since the NL set also consists of images that are not characterized by high brightness, it is expected that they will increase the average brightness only as much as necessary. This explains the fact that in this specific metric with no reference, the classical algorithms seem to perform better, but in reality they simply over-enhance the images, which does not correspond to the reality of the data set we are studying, and is also justified by the fact that in the metrics with reference the DL techniques give much better results.

Based on the above analysis, we conclude that DL techniques are more efficient than classical techniques, as they improve more the values of the metrics with reference, and better respond to the needs of the data set (e.g.

they do not cause over-enhancement as for example equalization histogram). Which approach we actually use depends on the problem we have to solve as well as the data we have available. In the case where we have ground truth data available, as in the RELLISUR set we studied, it is better to use methods with learning, such as DL techniques we saw in chapter 3, adapting and optimizing the model to the needs. However, if we do not have ground truth data, it is obviously better to use some of the classical techniques. A combination of techniques could also be used, where as a first step we apply one of the classical techniques (preferably the dehaze-based method, for the reasons we mentioned in section 4.1) to produce quasi ground truth images, and as a next step to train a DL model based on these.

4.4 Conclusion and Future work

In this last subsection we will summarize the analysis so far, drawing final conclusions and outlining steps for future research. We have seen that all applications involving the use of digital images require these images to be of high quality without noise. However, there are many factors that can reduce the quality of an image. One of them is low-light conditions, which result in images characterized by low contrast and brightness, and the introduction of additional noise and color distortions. To use these images, they must first be appropriately enhanced, an area of research called Low Light Image Enhancement (LLIE). The first approach we took to solve the LLIE problem was to apply classical image processing methods, which either applied a mathematical function to the value of each pixel or performed a convolution operation with a window. We first used a simple linear transformation, and while it did enhance the images to a degree, it was the least efficient algorithm, for each darkness level. Then we applied non-linear functions, such as the gamma transform and the logarithmic transform, where for small darkness levels they were among the most efficient classical methods, but their performance decreased drastically with the increase of the darkness level, making these methods unreliable for solving the problem. We then applied histogram equalization, and we saw that it did enhance the LL images but much more than it should, which led to reduced performance on metrics with reference as it introduced more noise and chromatic distortions. Nevertheless, it could be used as a general method of enhancing dark images, but not in the context of the data set we studied. Following, we applied methods based on the Retinex theory,

where we saw that their great advantage is their resistance to the darkness level, since their performance remained almost constant with its increase. But we also noticed that color distortions appeared here, resulting in a decrease in the quality of the images. Finally, we applied a method based on defogging, with the logic that the negative of an LL image is as if the image was taken in foggy conditions. If we apply a fog removal method to the negative and reverse the result then we see that an enhanced LL image is obtained. We saw that this particular method gave impressive results, even at high darkness levels, without introducing additional noise or color distortions. For the reasons we described above (no need to select parameters, it has room for improvement, it is resistant to increasing the darkness level, etc.) we chose it as the best of the classic methods.

Continuing, in chapter 3 we followed a different path, using Deep Learning techniques, in an attempt to exploit all the information present in the data set, to approximate the mapping between LL images space and NL images space. We started by applying the original LL-CNN architecture, and we saw that it gave impressive results improving all metric values, much more than the classical algorithms. However, it was observed that for large darkness levels it faced difficulty in retrieving the color information. In an attempt to solve this problem, we proposed a variation of the original architecture, where we added additional convolutional layers in order to learn additional features, which could help in recovering the color information. We saw that it did improve the results, but by making the architecture more complex and increasing the training difficulty. In an effort to facilitate the training process we introduced a different logic. Specifically, we assumed that the LL images have been derived from the NL images by subtracting a degradation image, e. So, we made the network learn this degradation, by giving it as input the LL images and as expected output the difference between NL and LL images. This way we managed to get performance results comparable to the other architectures, using fewer parameters. For this reason, we concluded that this particular architecture with this training method is the best approach for solving the LLIE problem.

At this point, before concluding the thesis, it is worth referring to future research steps that could be implemented. Starting from the classic point processing algorithms, an exhaustive experimentation could be done either to find a suitable mathematical function whose application to the value of each pixel will give better results, or to find the optimal parameter values (for the transformations that have parameters such as gamma transformation and logarithmic transformation). For the methods using

Retinex theory, one could first experiment with different parameter values of the surround function, since here we have seen that small values have very poor performance. In addition, we could experiment with different forms of the surround function, since in this work we used the default function proposed by the authors [26, 27]. Finally, for the method that uses the defogging/dehaze algorithm, we could experiment with different defogging algorithms and see if the performance of the method can be further improved.

Obviously, what we mentioned in the previous paragraph are fine-tuning approaches to the specific data set, which are time-consuming and tedious. For this reason, we applied Deep Learning methods, where we saw that they are characterized by increased performance. On the architecture we presented we could make changes such as different loss function and different loss minimization algorithms, and in addition apply hyperparameter tuning, to further increase the performance of the model. Also, either different architectures or even a different DL paradigm could be tested, by applying generative architectures (GANs, diffusion models). Finally, we could apply a combination of classical methods with DL techniques. For example, we could implement a CNN architecture that approximates the fog model of negative LL images, and then apply this defogging model to classical method based on dehaze.

In conclusion, all the methods we described during this thesis are applied to the RGB color space. Although this approach gives satisfactory results, it has the disadvantage that it ignores the possible correlations that exist between the color bands. Based on this we understand that we could also experiment with different color spaces, such as HSI, and compare the results between them.

A – References

[1] Shafi, I.; Din, S.; Khan, A.; Díez, I.D.L.T.; Casanova, R.d.J.P.; Pifarre, K.T.; Ashraf, I. An Effective Method for Lung Cancer Diagnosis from CT Scan Using Deep Learning-Based Support Vector Network. *Cancers* **2022**, *14*, 5457. https://doi.org/10.3390/cancers14215457

[2] Despotović I, Goossens B, Philips W. MRI segmentation of the human brain: challenges, methods, and applications. *Comput Math Methods Med*. 2015;2015:450341. doi:10.1155/2015/450341

[3] Qin, R.; Liu, T. A Review of Landcover Classification with Very-High Resolution Remotely Sensed Optical Images—Analysis Unit, Model Scalability and Transferability. *Remote Sens.* **2022**, *14*, 646. https://doi.org/10.3390/rs14030646

[4] Wang, W.; Li, W.; Zhang, C.; Zhang, W. Improving Object-Based Land Use/Cover Classification from Medium Resolution Imagery by Markov Chain Geostatistical Post-Classification. *Land* **2018**, *7*, 31. https://doi.org/10.3390/land7010031

[5] J. S. Bartunek, M. Nilsson, B. Sallberg and I. Claesson, "Adaptive Fingerprint Image Enhancement With Emphasis on Preprocessing of Data," in IEEE Transactions on Image Processing, vol. 22, no. 2, pp. 644-656, Feb. 2013, doi: 10.1109/TIP.2012.2220373.

[6] Le Hong, Hai & Nguyễn, Hoá & Nguyen, Tri-Thanh. (2016). A Complete Fingerprint Matching Algorithm on GPU for a Large Scale Identification System. 10.1007/978-981-10-0557-2_67.

[7] Sultani, Waqas & Chen, Chen & Shah, Mubarak. (2018). Real-world Anomaly Detection in Surveillance Videos.

[8] Selvi, E.; Adimoolam, M.; Karthi, G.; Thinakaran, K.; Balamurugan, N.M.; Kannadasan, R.; Wechtaisong, C.; Khan, A.A. Suspicious Actions Detection System Using Enhanced CNN and Surveillance Video. *Electronics* **2022**, *11*, 4210. https://doi.org/10.3390/electronics11244210

[9] Y. -K. Ki and D. -Y. Lee, "A Traffic Accident Recording and Reporting Model at Intersections," in IEEE Transactions on Intelligent Transportation Systems, vol. 8, no. 2, pp. 188-194, June 2007, doi: 10.1109/TITS.2006.890070.

[10] L.P. Clarke, R.P. Velthuizen, M.A. Camacho, J.J. Heine, M. Vaidyanathan, L.O. Hall, R.W. Thatcher, M.L. Silbiger, MRI segmentation: Methods and applications, Magnetic Resonance Imaging, Volume 13, Issue 3, 1995, Pages 343-368

[11] Gonzalez, R. C., Woods, R. E. (2008). *Digital image processing*. Upper Saddle River, N.J.: Prentice Hall. ISBN: 9780131687288 013168728X 9780135052679 013505267X

[12] Boyat, Ajay & Joshi, Brijendra. (2015). A Review Paper: Noise Models in Digital Image Processing. Signal & Image Processing : An International Journal. 6. 10.5121/sipij.2015.6206.

[13] Wang, Ruxin & Tao, Dacheng. (2014). Recent Progress in Image Deblurring.

[14] Q. Lu, W. Zhou, L. Fang and H. Li, "Robust Blur Kernel Estimation for License Plate Images From Fast Moving Vehicles," in IEEE Transactions on Image Processing, vol. 25, no. 5, pp. 2311-2323, May 2016, doi: 10.1109/TIP.2016.2535375.

[15] W. Wang, X. Wu, X. Yuan and Z. Gao, "An Experiment-Based Review of Low-Light Image Enhancement Methods," in IEEE Access, vol. 8, pp. 87884-87917, 2020, doi: 10.1109/ACCESS.2020.2992749.

[16] Ai, S.; Kwon, J. Extreme Low-Light Image Enhancement for Surveillance Cameras Using Attention U-Net. *Sensors* **2020**, *20*, 495. https://doi.org/10.3390/s20020495

[17] Aakerberg, A., Nasrollahi, K., & Moeslund, T. B. (2021). RELLISUR: A Real Low-Light Image Super-Resolution Dataset. In *Advances in Neural Information Processing Systems 35 (NeurIPS 2021)*

[18] Adobe Lightroom Classic, version 10.0. Adobe Inc., 2020.

[19] Bay, H., Tuytelaars, T., Van Gool, L. (2006). SURF: Speeded Up Robust Features. In: Leonardis, A., Bischof, H., Pinz, A. (eds) Computer Vision – ECCV 2006. ECCV 2006. Lecture Notes in Computer Science, vol 3951. Springer, Berlin, Heidelberg. https://doi.org/10.1007/11744023_32

[20] P.H.S. Torr, A. Zisserman, MLESAC: A New Robust Estimator with Application to Estimating Image Geometry, Computer Vision and Image Understanding, Volume 78, Issue 1, 2000, Pages 138-156, ISSN 1077-3142, https://doi.org/10.1006/cviu.1999.0832

[21] Yuantao, Z., et al. "Digital TDI technology based on global shutter sCMOS image sensor for low-light-level imaging." *Acta Optica Sinica* 38.9 (2018).

[22] Zhou Wang, A. C. Bovik, H. R. Sheikh and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," in IEEE Transactions on Image Processing, vol. 13, no. 4, pp. 600-612, April 2004, doi: 10.1109/TIP.2003.819861.

[23] A. Mittal, A. K. Moorthy and A. C. Bovik, "No-Reference Image Quality Assessment in the Spatial Domain," in IEEE Transactions on Image Processing, vol. 21, no. 12, pp. 4695-4708, Dec. 2012, doi: 10.1109/TIP.2012.2214050.

[24] A. Mittal, R. Soundararajan and A. C. Bovik, "Making a “Completely Blind” Image Quality Analyzer," in IEEE Signal Processing Letters, vol. 20, no. 3, pp. 209-212, March 2013, doi: 10.1109/LSP.2012.2227726.

[25] Edwin H. Land and John J. McCann, "Lightness and Retinex Theory," J. Opt. Soc. Am. 61, 1-11 (1971)

[26] D. J. Jobson, Z. Rahman and G. A. Woodell, "Properties and performance of a center/surround retinex," in IEEE Transactions on Image Processing, vol. 6, no. 3, pp. 451-462, March 1997, doi: 10.1109/83.557356.

[27] D. J. Jobson, Z. Rahman and G. A. Woodell, "A multiscale retinex for bridging the gap between color images and the human observation of scenes," in IEEE Transactions on Image Processing, vol. 6, no. 7, pp. 965-976, July 1997, doi: 10.1109/83.597272.

[28] Xuan Dong et al., "Fast efficient algorithm for enhancement of low lighting video," 2011 IEEE International Conference on Multimedia and Expo, Barcelona, 2011, pp. 1-6, doi: 10.1109/ICME.2011.6012107.

[29] Kaiming He, Jian Sun and Xiaoou Tang, "Single image haze removal using dark channel prior," 2009 IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, 2009, pp. 1956-1963, doi: 10.1109/CVPR.2009.5206515.

[30] Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep learning. The MIT Press.

[31] Y. Lecun, L. Bottou, Y. Bengio and P. Haffner, "Gradient-based learning applied to document recognition," in Proceedings of the IEEE, vol. 86, no. 11, pp. 2278-2324, Nov. 1998, doi: 10.1109/5.726791.

[32] Shun-ichi Amari, Backpropagation and stochastic gradient descent method, Neurocomputing, Volume 5, Issues 4–5, 1993, Pages 185-196, ISSN 0925-2312, https://doi.org/10.1016/0925-2312(93)90006-O

[33] X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedfoward neural networks. In *AISTATS*, 2010.

[34] K. He, X. Zhang, S. Ren and J. Sun, "Deep Residual Learning for Image Recognition," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 2016, pp. 770-778, doi: 10.1109/CVPR.2016.90.

[35] C. Szegedy, et al., "Going deeper with convolutions," in 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 2015 pp. 1-9. doi: 10.1109/CVPR.2015.7298594

[36] L. Tao, C. Zhu, G. Xiang, Y. Li, H. Jia and X. Xie, "LLCNN: A convolutional neural network for low-light image enhancement," 2017 IEEE Visual Communications and Image Processing (VCIP), St. Petersburg, FL, USA, 2017, pp. 1-4, doi: 10.1109/VCIP.2017.8305143.

[37] Sheng He, Lambert Schomaker, DeepOtsu: Document enhancement and binarization using iterative deep learning, Pattern Recognition, Volume 91, 2019, Pages 379-390, ISSN 0031-3203, https://doi.org/10.1016/j.patcog.2019.01.025

B – Code

B.2: Chapter 2 Code

Here we present the scripts we used during chapter 2, detailing the steps for each section of the chapter.

Linear Streaching

The implementation function of the linear transformation is:

```
function NL_image = simple_linear_transformation(LL_image,range_min,range_max)

%=========================================================================
% Function for linear stretching of an image. The function takes as input
% the image we want to stretch and the range of the available gray values
% (usually range_min=0 and range_max=255). After that the function applies
% the simple linear stretching algorithm and returns the result.
% Author: Panagiotis Koutsaftis aivc21010
%=========================================================================

%converting the image to double for calculations
LL_image = double(LL_image);

%size of the image
[rows,columns,bands] = size(LL_image);

%dynamic range of the NL image gray values
range_diff = range_max-range_min;

%preallocating the matrix for the NL image with zeros
NL_image = zeros(size(LL_image));

%appliying linear stratching to the RGB components
for band=1:bands
    %x_low
    f_min = min(min(LL_image(:,:,band)));
    %x_high
    f_max = max(max(LL_image(:,:,band)));
    diff = (LL_image(:,:,band)-f_min)./(f_max-f_min);
    temp = round((diff.*range_diff)+range_min);
    NL_image(:,:,band) = temp;
end

%converting the result to uint8
NL_image = uint8(NL_image);

```

Figure B.2.1: Simple linear transformation implementation function

This function accepts as input the image to which we want to apply the linear transformation, as well as the range of available brightness values (x_{min}=$range_{min}$ and x_{max}=$range_{max}$). It then converts the image into a double data type so that operations can be performed on the pixel values, and calculates the dimensions of the image and the dynamic range of the available brightness values. Then it goes and applies the linear transformation formula to every pixel in the image. Specifically, here we apply it to each channel separately, utilizing the vectorization capabilities of MATLAB. We calculate for each channel x_{low} and x_{high} and we apply the linear transformation formula. We store the result in an array, which we initialized with zeros, and convert its values to uint8, so that it can be used as an image and not as a simple array with double values.

Now that we have the function to calculate the linear transformation, we need to apply it to all the images in our dataset. We will apply the function per darkness level, saving the results in separate folders, which we will then use to evaluate the method. The scripts where we apply the function are presented in the following images.

```
close all; clear; clc; format compact;

%loading the data
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\2.5\LL-2.5');
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\3.0\LL-3.0');
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\3.5\LL-3.5');
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\4.0\LL-4.0');
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\4.5\LL-4.5');
LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\5.0\LL-5.0');


%size of the dataset
files = size(LL_images.Files);
numOfImages = files(1);



%applying Simple Linear Transformation
for index=1:numOfImages
    LL_img = readimage(LL_images,index);
    image_name = char(LL_images.Files(index));
    %NL_img = readimage(NL_images,index);
    NL_linear = simple_linear_transformation(LL_img,0,255);
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\2.5\Linear Transformation\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\3.0\Linear Transformation\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\3.5\Linear Transformation\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\4.0\Linear Transformation\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\4.5\Linear Transformation\',image_name((end-8):(end-4)),'.png'];
    file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\5.0\Linear Transformation\',image_name((end-8):(end-4)),'.png'];
    imwrite(NL_linear,file_path);
end

```

Figure B.2.2: Script to apply linear transformation to the training data

```
close all; clear; clc; format compact;

%========================================================================
%                          VALIDATION IMAGES
%========================================================================
%reading the Low Light Data (VALIDATION)
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\3.0\LL-3.0');
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\3.5\LL-3.5');
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\4.0\LL-4.0');
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\4.5\LL-4.5');
LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\5.0\LL-5.0');

%size of the dataset
files = size(LL_images.Files);
numOfImages = files(1);

disp('Working on validation images...')
%applying Simple Linear Transformation on Validation Images
for index=1:numOfImages
    LL_img = readimage(LL_images,index);
    image_name = char(LL_images.Files(index));
    %NL_img = readimage(NL_images,index);
    NL_linear = simple_linear_transformation(LL_img,0,255);
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\3.0\Linear Transformation\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\3.5\Linear Transformation\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\4.0\Linear Transformation\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\4.5\Linear Transformation\',image_name((end-8):(end-4)),'.png'];
    file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\5.0\Linear Transformation\',image_name((end-8):(end-4)),'.png'];
    imwrite(NL_linear,file_path);
end

disp('Done with Validation Images!')
%========================================================================
%                              TEST IMAGES
%========================================================================
%reading the Low Light Data (TEST)
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\3.0\LL-3.0');
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\3.5\LL-3.5');
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\4.0\LL-4.0');
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\4.5\LL-4.5');
LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\5.0\LL-5.0');

%size of the dataset
files = size(LL_images.Files);
numOfImages = files(1);

disp('Working on Test images...')
%applying Simple Linear Transformation on Validation Images
for index=1:numOfImages
    LL_img = readimage(LL_images,index);
    image_name = char(LL_images.Files(index));
    %NL_img = readimage(NL_images,index);
    NL_linear = simple_linear_transformation(LL_img,0,255);
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\3.0\Linear Transformation\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\3.5\Linear Transformation\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\4.0\Linear Transformation\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\4.5\Linear Transformation\',image_name((end-8):(end-4)),'.png'];
    file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\5.0\Linear Transformation\',image_name((end-8):(end-4)),'.png'];
    imwrite(NL_linear,file_path);
end
disp('Done with Test Images!')
disp('DONE!')
```

Figure B.2.3: Script to apply linear transformation to validation and test data

In all cases we read the images and put them into a datastore, and calculate the size of the datastore (needed for the loop). Then, for each image of each set, we apply the linear transformation function, and save the result in a suitable folder, with the same name as the original LL image. Note that the script is executed 5 times, once for each darkness level, each time having the respective darkness level out of comment.

After this process is completed, we have at our disposal the experimental results of the linear transformation method. For these results we will calculate the metrics we mentioned above so that we can evaluate the performance of the algorithm. To calculate the metrics we applied the script presented in figure B.2.4.

```
close all; clear; clc; format long;
tic
%=========================================================================
%=========================================================================
%                            TRAINING IMAGES
%=========================================================================
%=========================================================================
disp('Woriking on Training Images...')
%=========================================================================
%                            GROUND TRUTH IMAGES
%=========================================================================

NL_images30 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\3.0\NL-3.0');
NL_images35 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\3.5\NL-3.5');
NL_images40 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\4.0\NL-4.0');
NL_images45 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\4.5\NL-4.5');
NL_images50 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\5.0\NL-5.0');

%=========================================================================
%               IMAGES FROM THE LINEAR TRANSFORMATION METHOD
%=========================================================================

NL_exp_res30 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\3.0\Linear Transformation');
NL_exp_res35 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\3.5\Linear Transformation');
NL_exp_res40 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\4.0\Linear Transformation');
NL_exp_res45 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\4.5\Linear Transformation');
NL_exp_res50 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\5.0\Linear Transformation');

%assessment for 3.0 image dataset
metrics_arr30 = assessment(NL_images30,NL_exp_res30);
%assessment for 3.5 image dataset
metrics_arr35 = assessment(NL_images35,NL_exp_res35);
%assessment for 4.0 image dataset
metrics_arr40 = assessment(NL_images40,NL_exp_res40);
%assessment for 4.5 image dataset
metrics_arr45 = assessment(NL_images45,NL_exp_res45);
%assessment for 5.0 image dataset
metrics_arr50 = assessment(NL_images50,NL_exp_res50);

%columns titles
var_names = {'MSE','PSNR','SSIM','MV','STD','BRISQUE','NIQE'};
```

```
%convert the matrices to tables with column names
table30 = array2table(metrics_arr30,'VariableNames',var_names);
table35 = array2table(metrics_arr35,'VariableNames',var_names);
table40 = array2table(metrics_arr40,'VariableNames',var_names);
table45 = array2table(metrics_arr45,'VariableNames',var_names);
table50 = array2table(metrics_arr50,'VariableNames',var_names);

%saving the results to an excel file
writetable(table30,'linear_transformation_assessment_training.xlsx','Sheet','3.0');
writetable(table35,'linear_transformation_assessment_training.xlsx','Sheet','3.5');
writetable(table40,'linear_transformation_assessment_training.xlsx','Sheet','4.0');
writetable(table45,'linear_transformation_assessment_training.xlsx','Sheet','4.5');
writetable(table50,'linear_transformation_assessment_training.xlsx','Sheet','5.0');

disp('Done with Training Images!')
%testing 4.0 images
%I_rel = readimage(NL_images40,69);
%I_exp = readimage(NL_exp_res40,69);
%figure
%montage({I_rel,I_exp})

%=========================================================================
%=========================================================================
%                           VALIDATION IMAGES
%=========================================================================
%=========================================================================
disp('Working on Validation Images...')
%=========================================================================
%                           GROUND TRUTH IMAGES
%=========================================================================

NL_images30 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\3.0\NL-3.0');
NL_images35 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\3.5\NL-3.5');
NL_images40 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\4.0\NL-4.0');
NL_images45 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\4.5\NL-4.5');
NL_images50 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\5.0\NL-5.0');

%=========================================================================
%              IMAGES FROM THE LINEAR TRANSFORMATION METHOD
%=========================================================================

NL_exp_res30 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\3.0\Linear Transformation');
NL_exp_res35 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\3.5\Linear Transformation');
NL_exp_res40 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\4.0\Linear Transformation');
NL_exp_res45 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\4.5\Linear Transformation');
NL_exp_res50 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\5.0\Linear Transformation');

%assessment for 3.0 image dataset
metrics_arr30 = assessment(NL_images30,NL_exp_res30);
%assessment for 3.5 image dataset
metrics_arr35 = assessment(NL_images35,NL_exp_res35);
%assessment for 4.0 image dataset
metrics_arr40 = assessment(NL_images40,NL_exp_res40);
%assessment for 4.5 image dataset
metrics_arr45 = assessment(NL_images45,NL_exp_res45);
%assessment for 5.0 image dataset
metrics_arr50 = assessment(NL_images50,NL_exp_res50);

%columns titles
var_names = {'MSE','PSNR','SSIM','MV','STD','BRISQUE','NIQE'};

%convert the matrices to tables with column names
table30 = array2table(metrics_arr30,'VariableNames',var_names);
table35 = array2table(metrics_arr35,'VariableNames',var_names);
table40 = array2table(metrics_arr40,'VariableNames',var_names);
table45 = array2table(metrics_arr45,'VariableNames',var_names);
table50 = array2table(metrics_arr50,'VariableNames',var_names);

%saving the results to an excel file
writetable(table30,'linear_transformation_assessment_validation.xlsx','Sheet','3.0');
writetable(table35,'linear_transformation_assessment_validation.xlsx','Sheet','3.5');
writetable(table40,'linear_transformation_assessment_validation.xlsx','Sheet','4.0');
writetable(table45,'linear_transformation_assessment_validation.xlsx','Sheet','4.5');
writetable(table50,'linear_transformation_assessment_validation.xlsx','Sheet','5.0');

disp('Done with Validation Images!')

%=========================================================================
%=========================================================================
%                          TEST IMAGES
%=========================================================================
%=========================================================================
disp('Working on Test Images...')
```

```
%=========================================================================
%                          GROUND TRUTH IMAGES
%=========================================================================

NL_images30 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\3.0\NL-3.0');
NL_images35 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\3.5\NL-3.5');
NL_images40 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\4.0\NL-4.0');
NL_images45 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\4.5\NL-4.5');
NL_images50 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\5.0\NL-5.0');

%=========================================================================
%              IMAGES FROM THE LINEAR TRANSFORMATION METHOD
%=========================================================================

NL_exp_res30 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\3.0\Linear Transformation');
NL_exp_res35 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\3.5\Linear Transformation');
NL_exp_res40 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\4.0\Linear Transformation');
NL_exp_res45 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\4.5\Linear Transformation');
NL_exp_res50 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\5.0\Linear Transformation');

%assessment for 3.0 image dataset
metrics_arr30 = assessment(NL_images30,NL_exp_res30);
%assessment for 3.5 image dataset
metrics_arr35 = assessment(NL_images35,NL_exp_res35);
%assessment for 4.0 image dataset
metrics_arr40 = assessment(NL_images40,NL_exp_res40);
%assessment for 4.5 image dataset
metrics_arr45 = assessment(NL_images45,NL_exp_res45);
%assessment for 5.0 image dataset
metrics_arr50 = assessment(NL_images50,NL_exp_res50);

%columns titles
var_names = {'MSE','PSNR','SSIM','MV','STD','BRISQUE','NIQE'};

%convert the matrices to tables with column names
table30 = array2table(metrics_arr30,'VariableNames',var_names);
table35 = array2table(metrics_arr35,'VariableNames',var_names);
table40 = array2table(metrics_arr40,'VariableNames',var_names);
table45 = array2table(metrics_arr45,'VariableNames',var_names);
table50 = array2table(metrics_arr50,'VariableNames',var_names);

%saving the results to an excel file
writetable(table30,'linear_transformation_assessment_test.xlsx','Sheet','3.0');
writetable(table35,'linear_transformation_assessment_test.xlsx','Sheet','3.5');
writetable(table40,'linear_transformation_assessment_test.xlsx','Sheet','4.0');
writetable(table45,'linear_transformation_assessment_test.xlsx','Sheet','4.5');
writetable(table50,'linear_transformation_assessment_test.xlsx','Sheet','5.0');
```

Figure B.2.4: Linear transformation valuation script

We start by reading the ground truth, and experimentally produced images, for the training set, and store them in image datastores. Then, we use the assessment function, which we described above, and calculate the matrix with the metrics for each image of each datastore, saving the results in an excel file. We apply the same procedure for the validation and test sets.

Gamma Correction

```
function NL_image = gamma_correction(LL_image,gamma)

%=========================================================================
% Function that implements the Gamma correction algorithm. It takes as
% input the Low Light image and the value of the exponent and outputs the
% image after applying gamma correction.
% Author: Panagiotis Koutsaftis aivc21010
%=========================================================================

%converting to double needed for operations
LL_image = double(LL_image);

%applying the gamma correction
NL_image = round(((LL_image./255).^gamma).*255);

%converting back to uint8 cause we need to return an image
NL_image = uint8(NL_image);
```

Figure B.2.5: Implementation of the gamma correction algorithm

The function accepts as input the LL image to which we want to apply the gamma transformation, as well as the value of the correction constant. It then converts the image to the double data type, which is needed to do arithmetic operations, and then applies the transformation. We divide the image values by 255 so that the number we raise to γ is in the range [0,1], then raise to the correction constant and multiply by 255 so that the result is in the range [0,255]. Finally, we round the result and convert it to uint8, since we want to return an image and not just a matrix with values.

We will apply this function to enhance the LL images, which we do with the script shown in the following images. First, we read the images, where we want to apply the algorithm, using Image Datastore. Then, for each image contained in the Datastore, we apply the function and store the result in a folder in an appropriate path, applying the function 3 separate times, one for each value of the correction constant. We execute this script 5 times, once for each brightness level, and we follow the same procedure for the validation and test datasets, as shown in figure B.2.7.

```
close all; clear; clc; format compact;


%loading the data
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\3.0\LL-3.0');
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\3.5\LL-3.5');
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\4.0\LL-4.0');
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\4.5\LL-4.5');
LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\5.0\LL-5.0');

%size of the dataset
files = size(LL_images.Files);
numOfImages = files(1);

%applying Gamma Transformation with gamma = 0.1
for index=1:numOfImages
    LL_img = readimage(LL_images,index);
    image_name = char(LL_images.Files(index));
    %NL_img = readimage(NL_images,index);
    NL_gam = gamma_correction(LL_img,0.1);
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\3.0\Gamma correction\gamma_01\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\3.5\Gamma correction\gamma_01\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\4.0\Gamma correction\gamma_01\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\4.5\Gamma correction\gamma_01\',image_name((end-8):(end-4)),'.png'];
    file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\5.0\Gamma correction\gamma_01\',image_name((end-8):(end-4)),'.png'];
    imwrite(NL_gam,file_path);
end

%applying Gamma Transformation with gamma = 0.3
for index=1:numOfImages
    LL_img = readimage(LL_images,index);
    image_name = char(LL_images.Files(index));
    %NL_img = readimage(NL_images,index);
    NL_gam = gamma_correction(LL_img,0.3);
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\3.0\Gamma correction\gamma_03\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\3.5\Gamma correction\gamma_03\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\4.0\Gamma correction\gamma_03\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\4.5\Gamma correction\gamma_03\',image_name((end-8):(end-4)),'.png'];
    file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\5.0\Gamma correction\gamma_03\',image_name((end-8):(end-4)),'.png'];
    imwrite(NL_gam,file_path);
end


%applying Gamma Transformation with gamma = 0.8
for index=1:numOfImages
    LL_img = readimage(LL_images,index);
    image_name = char(LL_images.Files(index));
    %NL_img = readimage(NL_images,index);
    NL_gam = gamma_correction(LL_img,0.8);
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\3.0\Gamma correction\gamma_08\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\3.5\Gamma correction\gamma_08\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\4.0\Gamma correction\gamma_08\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\4.5\Gamma correction\gamma_08\',image_name((end-8):(end-4)),'.png'];
    file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\5.0\Gamma correction\gamma_08\',image_name((end-8):(end-4)),'.png'];
    imwrite(NL_gam,file_path);
end
```

Figure B.2.6: applying gamma correction to training dataset

After this process is completed, we have the experimental results of the specific method, and we can apply the evaluation algorithm, as presented in the previous section. Here it is configured a bit differently, as we have 3 different experiments (3 different correction constants). The script we apply for the evaluation is shown in figure B.2.8.

```
close all; clear; clc; format compact;


%=========================================================================
%                          VALIDATION IMAGES
%=========================================================================

%reading the Low Light Data (VALIDATION)
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\3.0\LL-3.0');
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\3.5\LL-3.5');
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\4.0\LL-4.0');
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\4.5\LL-4.5');
LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\5.0\LL-5.0');

%size of the dataset
files = size(LL_images.Files);
numOfImages = files(1);

disp('Working on validation images...')
%applying Gamma Transformation with gamma = 0.1
for index=1:numOfImages
    LL_img = readimage(LL_images,index);
    image_name = char(LL_images.Files(index));
    %NL_img = readimage(NL_images,index);
    NL_gam = gamma_correction(LL_img,0.1);
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\3.0\Gamma correction\gamma_01\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\3.5\Gamma correction\gamma_01\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\4.0\Gamma correction\gamma_01\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\4.5\Gamma correction\gamma_01\',image_name((end-8):(end-4)),'.png'];
    file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\5.0\Gamma correction\gamma_01\',image_name((end-8):(end-4)),'.png'];
    imwrite(NL_gam,file_path);
end
disp('Done with gamma=0.1!')

%applying Gamma Transformation with gamma = 0.3
for index=1:numOfImages
    LL_img = readimage(LL_images,index);
    image_name = char(LL_images.Files(index));
    %NL_img = readimage(NL_images,index);
    NL_gam = gamma_correction(LL_img,0.3);
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\3.0\Gamma correction\gamma_03\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\3.5\Gamma correction\gamma_03\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\4.0\Gamma correction\gamma_03\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\4.5\Gamma correction\gamma_03\',image_name((end-8):(end-4)),'.png'];
    file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\5.0\Gamma correction\gamma_03\',image_name((end-8):(end-4)),'.png'];
    imwrite(NL_gam,file_path);
end
disp('Done with gamma=0.3!')

%applying Gamma Transformation with gamma = 0.8
for index=1:numOfImages
    LL_img = readimage(LL_images,index);
    image_name = char(LL_images.Files(index));
    %NL_img = readimage(NL_images,index);
    NL_gam = gamma_correction(LL_img,0.8);
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\3.0\Gamma correction\gamma_08\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\3.5\Gamma correction\gamma_08\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\4.0\Gamma correction\gamma_08\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\4.5\Gamma correction\gamma_08\',image_name((end-8):(end-4)),'.png'];
    file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\5.0\Gamma correction\gamma_08\',image_name((end-8):(end-4)),'.png'];
    imwrite(NL_gam,file_path);
end
disp('Done with gamma=0.8!')
disp('Done with validation images!')

%=========================================================================
%                             TEST IMAGES
%=========================================================================
%reading the Low Light Data (TEST)
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\3.0\LL-3.0');
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\3.5\LL-3.5');
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\4.0\LL-4.0');
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\4.5\LL-4.5');
LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\5.0\LL-5.0');

%size of the dataset
files = size(LL_images.Files);
numOfImages = files(1);

disp('Working on Test images...')
disp('Working on Test images...')
%applying Gamma Transformation with gamma = 0.1
for index=1:numOfImages
    LL_img = readimage(LL_images,index);
    image_name = char(LL_images.Files(index));
    %NL_img = readimage(NL_images,index);
    NL_gam = gamma_correction(LL_img,0.1);
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\3.0\Gamma correction\gamma_01\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\3.5\Gamma correction\gamma_01\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\4.0\Gamma correction\gamma_01\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\4.5\Gamma correction\gamma_01\',image_name((end-8):(end-4)),'.png'];
    file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\5.0\Gamma correction\gamma_01\',image_name((end-8):(end-4)),'.png'];
    imwrite(NL_gam,file_path);
end
disp('Done with gamma=0.1!')

%applying Gamma Transformation with gamma = 0.3
for index=1:numOfImages
    LL_img = readimage(LL_images,index);
    image_name = char(LL_images.Files(index));
    %NL_img = readimage(NL_images,index);
    NL_gam = gamma_correction(LL_img,0.3);
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\3.0\Gamma correction\gamma_03\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\3.5\Gamma correction\gamma_03\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\4.0\Gamma correction\gamma_03\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\4.5\Gamma correction\gamma_03\',image_name((end-8):(end-4)),'.png'];
    file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\5.0\Gamma correction\gamma_03\',image_name((end-8):(end-4)),'.png'];
    imwrite(NL_gam,file_path);
end
disp('Done with gamma=0.3!')

%applying Gamma Transformation with gamma = 0.8
for index=1:numOfImages
    LL_img = readimage(LL_images,index);
    image_name = char(LL_images.Files(index));
    %NL_img = readimage(NL_images,index);
    NL_gam = gamma_correction(LL_img,0.8);
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\3.0\Gamma correction\gamma_08\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\3.5\Gamma correction\gamma_08\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\4.0\Gamma correction\gamma_08\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\4.5\Gamma correction\gamma_08\',image_name((end-8):(end-4)),'.png'];
    file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\5.0\Gamma correction\gamma_08\',image_name((end-8):(end-4)),'.png'];
    imwrite(NL_gam,file_path);
end
disp('Done with gamma=0.8!')
disp('Done with test images!')
disp('Done!')
```

Figure B.2.7: applying gamma correction to validation and test datasets

```
close all; clear; clc; format long;
tic

%=========================================================================
%                           TRAINING IMAGES
%=========================================================================
disp('Working on Training Images...')

%Ground Truth images
NL_images30 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\3.0\NL-3.0');
NL_images35 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\3.5\NL-3.5');
NL_images40 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\4.0\NL-4.0');
NL_images45 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\4.5\NL-4.5');
NL_images50 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\5.0\NL-5.0');

%Images from Gamma correction with Gamma=0.1
NL_exp_res30 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\3.0\Gamma correction\gamma_01');
NL_exp_res35 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\3.5\Gamma correction\gamma_01');
NL_exp_res40 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\4.0\Gamma correction\gamma_01');
NL_exp_res45 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\4.5\Gamma correction\gamma_01');
NL_exp_res50 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\5.0\Gamma correction\gamma_01');

%assessment for 3.0 image dataset
metrics_arr30 = assessment(NL_images30,NL_exp_res30);
%assessment for 3.5 image dataset
metrics_arr35 = assessment(NL_images35,NL_exp_res35);
%assessment for 4.0 image dataset
metrics_arr40 = assessment(NL_images40,NL_exp_res40);
%assessment for 4.5 image dataset
metrics_arr45 = assessment(NL_images45,NL_exp_res45);
%assessment for 5.0 image dataset
metrics_arr50 = assessment(NL_images50,NL_exp_res50);

%columns titles
var_names = {'MSE','PSNR','SSIM','MV','STD','BRISQUE','NIQE'};

%convert the matrices to tables with column names
table30 = array2table(metrics_arr30,'VariableNames',var_names);
table35 = array2table(metrics_arr35,'VariableNames',var_names);
table40 = array2table(metrics_arr40,'VariableNames',var_names);
table45 = array2table(metrics_arr45,'VariableNames',var_names);
table50 = array2table(metrics_arr50,'VariableNames',var_names);

%saving the results to an excel file
writetable(table30,'Gamma01_assessment_training.xlsx','Sheet','3.0');
writetable(table35,'Gamma01_assessment_training.xlsx','Sheet','3.5');
writetable(table40,'Gamma01_assessment_training.xlsx','Sheet','4.0');
writetable(table45,'Gamma01_assessment_training.xlsx','Sheet','4.5');
writetable(table50,'Gamma01_assessment_training.xlsx','Sheet','5.0');
disp('Done with gamma=0.1!')

%Images from Gamma correction with Gamma=0.3
NL_exp_res30 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\3.0\Gamma correction\gamma_03');
NL_exp_res35 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\3.5\Gamma correction\gamma_03');
NL_exp_res40 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\4.0\Gamma correction\gamma_03');
NL_exp_res45 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\4.5\Gamma correction\gamma_03');
NL_exp_res50 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\5.0\Gamma correction\gamma_03');

%assessment for 3.0 image dataset
metrics_arr30 = assessment(NL_images30,NL_exp_res30);
%assessment for 3.5 image dataset
metrics_arr35 = assessment(NL_images35,NL_exp_res35);
%assessment for 4.0 image dataset
metrics_arr40 = assessment(NL_images40,NL_exp_res40);
%assessment for 4.5 image dataset
metrics_arr45 = assessment(NL_images45,NL_exp_res45);
%assessment for 5.0 image dataset
metrics_arr50 = assessment(NL_images50,NL_exp_res50);

%columns titles
var_names = {'MSE','PSNR','SSIM','MV','STD','BRISQUE','NIQE'};

%convert the matrices to tables with column names
table30 = array2table(metrics_arr30,'VariableNames',var_names);
table35 = array2table(metrics_arr35,'VariableNames',var_names);
table40 = array2table(metrics_arr40,'VariableNames',var_names);
table45 = array2table(metrics_arr45,'VariableNames',var_names);
table50 = array2table(metrics_arr50,'VariableNames',var_names);

%saving the results to an excel file
writetable(table30,'Gamma03_assessment_training.xlsx','Sheet','3.0');
writetable(table35,'Gamma03_assessment_training.xlsx','Sheet','3.5');
writetable(table40,'Gamma03_assessment_training.xlsx','Sheet','4.0');
writetable(table45,'Gamma03_assessment_training.xlsx','Sheet','4.5');
writetable(table50,'Gamma03_assessment_training.xlsx','Sheet','5.0');
disp('Done with gamma=0.3!')
```

```
%Images from Gamma correction with Gamma=0.8
NL_exp_res30 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\3.0\Gamma correction\gamma_08');
NL_exp_res35 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\3.5\Gamma correction\gamma_08');
NL_exp_res40 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\4.0\Gamma correction\gamma_08');
NL_exp_res45 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\4.5\Gamma correction\gamma_08');
NL_exp_res50 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\5.0\Gamma correction\gamma_08');

%assessment for 3.0 image dataset
metrics_arr30 = assessment(NL_images30,NL_exp_res30);
%assessment for 3.5 image dataset
metrics_arr35 = assessment(NL_images35,NL_exp_res35);
%assessment for 4.0 image dataset
metrics_arr40 = assessment(NL_images40,NL_exp_res40);
%assessment for 4.5 image dataset
metrics_arr45 = assessment(NL_images45,NL_exp_res45);
%assessment for 5.0 image dataset
metrics_arr50 = assessment(NL_images50,NL_exp_res50);

%columns titles
var_names = {'MSE','PSNR','SSIM','MV','STD','BRISQUE','NIQE'};

%convert the matrices to tables with column names
table30 = array2table(metrics_arr30,'VariableNames',var_names);
table35 = array2table(metrics_arr35,'VariableNames',var_names);
table40 = array2table(metrics_arr40,'VariableNames',var_names);
table45 = array2table(metrics_arr45,'VariableNames',var_names);
table50 = array2table(metrics_arr50,'VariableNames',var_names);

%saving the results to an excel file
writetable(table30,'Gamma08_assessment_training.xlsx','Sheet','3.0');
writetable(table35,'Gamma08_assessment_training.xlsx','Sheet','3.5');
writetable(table40,'Gamma08_assessment_training.xlsx','Sheet','4.0');
writetable(table45,'Gamma08_assessment_training.xlsx','Sheet','4.5');
writetable(table50,'Gamma08_assessment_training.xlsx','Sheet','5.0');
disp('Done with gamma=0.8!')
disp('Done with Training Images!')

%=========================================================================
%                           VALIDATION IMAGES
%=========================================================================
disp('Working with Validation Images...')

%Ground Truth images
NL_images30 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\3.0\NL-3.0');
NL_images35 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\3.5\NL-3.5');
NL_images40 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\4.0\NL-4.0');
NL_images45 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\4.5\NL-4.5');
NL_images50 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\5.0\NL-5.0');

%Images from Gamma correction with Gamma=0.1
NL_exp_res30 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\3.0\Gamma correction\gamma_01');
NL_exp_res35 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\3.5\Gamma correction\gamma_01');
NL_exp_res40 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\4.0\Gamma correction\gamma_01');
NL_exp_res45 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\4.5\Gamma correction\gamma_01');
NL_exp_res50 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\5.0\Gamma correction\gamma_01');

%assessment for 3.0 image dataset
metrics_arr30 = assessment(NL_images30,NL_exp_res30);
%assessment for 3.5 image dataset
metrics_arr35 = assessment(NL_images35,NL_exp_res35);
%assessment for 4.0 image dataset
metrics_arr40 = assessment(NL_images40,NL_exp_res40);
%assessment for 4.5 image dataset
metrics_arr45 = assessment(NL_images45,NL_exp_res45);
%assessment for 5.0 image dataset
metrics_arr50 = assessment(NL_images50,NL_exp_res50);

%columns titles
var_names = {'MSE','PSNR','SSIM','MV','STD','BRISQUE','NIQE'};

%convert the matrices to tables with column names
table30 = array2table(metrics_arr30,'VariableNames',var_names);
table35 = array2table(metrics_arr35,'VariableNames',var_names);
table40 = array2table(metrics_arr40,'VariableNames',var_names);
table45 = array2table(metrics_arr45,'VariableNames',var_names);
table50 = array2table(metrics_arr50,'VariableNames',var_names);

%saving the results to an excel file
writetable(table30,'Gamma01_assessment_validation.xlsx','Sheet','3.0');
writetable(table35,'Gamma01_assessment_validation.xlsx','Sheet','3.5');
writetable(table40,'Gamma01_assessment_validation.xlsx','Sheet','4.0');
writetable(table45,'Gamma01_assessment_validation.xlsx','Sheet','4.5');
writetable(table50,'Gamma01_assessment_validation.xlsx','Sheet','5.0');
disp('Done with gamma=0.1!')

%Images from Gamma correction with Gamma=0.3
NL_exp_res30 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\3.0\Gamma correction\gamma_03');
NL_exp_res35 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\3.5\Gamma correction\gamma_03');
NL_exp_res40 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\4.0\Gamma correction\gamma_03');
NL_exp_res45 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\4.5\Gamma correction\gamma_03');
```

```
NL_exp_res35 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\3.5\Gamma correction\gamma_03');
NL_exp_res40 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\4.0\Gamma correction\gamma_03');
NL_exp_res45 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\4.5\Gamma correction\gamma_03');
NL_exp_res50 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\5.0\Gamma correction\gamma_03');

%assessment for 3.0 image dataset
metrics_arr30 = assessment(NL_images30,NL_exp_res30);
%assessment for 3.5 image dataset
metrics_arr35 = assessment(NL_images35,NL_exp_res35);
%assessment for 4.0 image dataset
metrics_arr40 = assessment(NL_images40,NL_exp_res40);
%assessment for 4.5 image dataset
metrics_arr45 = assessment(NL_images45,NL_exp_res45);
%assessment for 5.0 image dataset
metrics_arr50 = assessment(NL_images50,NL_exp_res50);

%columns titles
var_names = {'MSE','PSNR','SSIM','MV','STD','BRISQUE','NIQE'};

%convert the matrices to tables with column names
table30 = array2table(metrics_arr30,'VariableNames',var_names);
table35 = array2table(metrics_arr35,'VariableNames',var_names);
table40 = array2table(metrics_arr40,'VariableNames',var_names);
table45 = array2table(metrics_arr45,'VariableNames',var_names);
table50 = array2table(metrics_arr50,'VariableNames',var_names);

%saving the results to an excel file
writetable(table30,'Gamma03_assessment_validation.xlsx','Sheet','3.0');
writetable(table35,'Gamma03_assessment_validation.xlsx','Sheet','3.5');
writetable(table40,'Gamma03_assessment_validation.xlsx','Sheet','4.0');
writetable(table45,'Gamma03_assessment_validation.xlsx','Sheet','4.5');
writetable(table50,'Gamma03_assessment_validation.xlsx','Sheet','5.0');
disp('Done with gamma=0.3!')

%Images from Gamma correction with Gamma=0.8
NL_exp_res30 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\3.0\Gamma correction\gamma_08');
NL_exp_res35 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\3.5\Gamma correction\gamma_08');
NL_exp_res40 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\4.0\Gamma correction\gamma_08');
NL_exp_res45 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\4.5\Gamma correction\gamma_08');
NL_exp_res50 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\5.0\Gamma correction\gamma_08');

%assessment for 3.0 image dataset
metrics_arr30 = assessment(NL_images30,NL_exp_res30);
%assessment for 3.5 image dataset
metrics_arr35 = assessment(NL_images35,NL_exp_res35);
%assessment for 4.0 image dataset
metrics_arr40 = assessment(NL_images40,NL_exp_res40);
%assessment for 4.5 image dataset
metrics_arr45 = assessment(NL_images45,NL_exp_res45);
%assessment for 5.0 image dataset
metrics_arr50 = assessment(NL_images50,NL_exp_res50);

%columns titles
var_names = {'MSE','PSNR','SSIM','MV','STD','BRISQUE','NIQE'};

%convert the matrices to tables with column names
table30 = array2table(metrics_arr30,'VariableNames',var_names);
table35 = array2table(metrics_arr35,'VariableNames',var_names);
table40 = array2table(metrics_arr40,'VariableNames',var_names);
table45 = array2table(metrics_arr45,'VariableNames',var_names);
table50 = array2table(metrics_arr50,'VariableNames',var_names);

%saving the results to an excel file
writetable(table30,'Gamma08_assessment_validation.xlsx','Sheet','3.0');
writetable(table35,'Gamma08_assessment_validation.xlsx','Sheet','3.5');
writetable(table40,'Gamma08_assessment_validation.xlsx','Sheet','4.0');
writetable(table45,'Gamma08_assessment_validation.xlsx','Sheet','4.5');
writetable(table50,'Gamma08_assessment_validation.xlsx','Sheet','5.0');
disp('Done with gamma=0.8!')
disp('Done with Validation Images!')

%========================================================================
%                              TEST IMAGES
%========================================================================
disp('Working with Test Images...')

%Ground Truth images
NL_images30 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\3.0\NL-3.0');
NL_images35 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\3.5\NL-3.5');
NL_images40 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\4.0\NL-4.0');
NL_images45 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\4.5\NL-4.5');
NL_images50 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\5.0\NL-5.0');

%Images from Gamma correction with Gamma=0.1
NL_exp_res30 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\3.0\Gamma correction\gamma_01');
NL_exp_res35 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\3.5\Gamma correction\gamma_01');
NL_exp_res40 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\4.0\Gamma correction\gamma_01');
NL_exp_res45 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\4.5\Gamma correction\gamma_01');
NL_exp_res50 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\5.0\Gamma correction\gamma_01');
```

```
264
265     %assessment for 3.0 image dataset
266     metrics_arr30 = assessment(NL_images30,NL_exp_res30);
267     %assessment for 3.5 image dataset
268     metrics_arr35 = assessment(NL_images35,NL_exp_res35);
269     %assessment for 4.0 image dataset
270     metrics_arr40 = assessment(NL_images40,NL_exp_res40);
271     %assessment for 4.5 image dataset
272     metrics_arr45 = assessment(NL_images45,NL_exp_res45);
273     %assessment for 5.0 image dataset
274     metrics_arr50 = assessment(NL_images50,NL_exp_res50);
275
276     %columns titles
277     var_names = {'MSE','PSNR','SSIM','MV','STD','BRISQUE','NIQE'};
278
279     %convert the matrices to tables with column names
280     table30 = array2table(metrics_arr30,'VariableNames',var_names);
281     table35 = array2table(metrics_arr35,'VariableNames',var_names);
282     table40 = array2table(metrics_arr40,'VariableNames',var_names);
283     table45 = array2table(metrics_arr45,'VariableNames',var_names);
284     table50 = array2table(metrics_arr50,'VariableNames',var_names);
285
286     %saving the results to an excel file
287     writetable(table30,'Gamma01_assessment_test.xlsx','Sheet','3.0');
288     writetable(table35,'Gamma01_assessment_test.xlsx','Sheet','3.5');
289     writetable(table40,'Gamma01_assessment_test.xlsx','Sheet','4.0');
290     writetable(table45,'Gamma01_assessment_test.xlsx','Sheet','4.5');
291     writetable(table50,'Gamma01_assessment_test.xlsx','Sheet','5.0');
292     disp('Done with gamma=0.1!')
293
294     %Images from Gamma correction with Gamma=0.3
295     NL_exp_res30 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\3.0\Gamma correction\gamma_03');
296     NL_exp_res35 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\3.5\Gamma correction\gamma_03');
297     NL_exp_res40 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\4.0\Gamma correction\gamma_03');
298     NL_exp_res45 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\4.5\Gamma correction\gamma_03');
299     NL_exp_res50 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\5.0\Gamma correction\gamma_03');
300
301     %assessment for 3.0 image dataset
302     metrics_arr30 = assessment(NL_images30,NL_exp_res30);
303     %assessment for 3.5 image dataset
304     metrics_arr35 = assessment(NL_images35,NL_exp_res35);
305     %assessment for 4.0 image dataset
306     metrics_arr40 = assessment(NL_images40,NL_exp_res40);
307     %assessment for 4.5 image dataset
308     metrics_arr45 = assessment(NL_images45,NL_exp_res45);
308     metrics_arr45 = assessment(NL_images45,NL_exp_res45);
309     %assessment for 5.0 image dataset
310     metrics_arr50 = assessment(NL_images50,NL_exp_res50);
311
312     %columns titles
313     var_names = {'MSE','PSNR','SSIM','MV','STD','BRISQUE','NIQE'};
314
315     %convert the matrices to tables with column names
316     table30 = array2table(metrics_arr30,'VariableNames',var_names);
317     table35 = array2table(metrics_arr35,'VariableNames',var_names);
318     table40 = array2table(metrics_arr40,'VariableNames',var_names);
319     table45 = array2table(metrics_arr45,'VariableNames',var_names);
320     table50 = array2table(metrics_arr50,'VariableNames',var_names);
321
322     %saving the results to an excel file
323     writetable(table30,'Gamma03_assessment_test.xlsx','Sheet','3.0');
324     writetable(table35,'Gamma03_assessment_test.xlsx','Sheet','3.5');
325     writetable(table40,'Gamma03_assessment_test.xlsx','Sheet','4.0');
326     writetable(table45,'Gamma03_assessment_test.xlsx','Sheet','4.5');
327     writetable(table50,'Gamma03_assessment_test.xlsx','Sheet','5.0');
328     disp('Done with gamma=0.3!')
329
330     %Images from Gamma correction with Gamma=0.8
331     NL_exp_res30 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\3.0\Gamma correction\gamma_08');
332     NL_exp_res35 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\3.5\Gamma correction\gamma_08');
333     NL_exp_res40 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\4.0\Gamma correction\gamma_08');
334     NL_exp_res45 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\4.5\Gamma correction\gamma_08');
335     NL_exp_res50 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\5.0\Gamma correction\gamma_08');
336
337     %assessment for 3.0 image dataset
338     metrics_arr30 = assessment(NL_images30,NL_exp_res30);
339     %assessment for 3.5 image dataset
340     metrics_arr35 = assessment(NL_images35,NL_exp_res35);
341     %assessment for 4.0 image dataset
342     metrics_arr40 = assessment(NL_images40,NL_exp_res40);
343     %assessment for 4.5 image dataset
344     metrics_arr45 = assessment(NL_images45,NL_exp_res45);
345     %assessment for 5.0 image dataset
346     metrics_arr50 = assessment(NL_images50,NL_exp_res50);
```

```
metrics_arr45 = assessment(NL_images45,NL_exp_res45);
%assessment for 5.0 image dataset
metrics_arr50 = assessment(NL_images50,NL_exp_res50);

%columns titles
var_names = {'MSE','PSNR','SSIM','MV','STD','BRISQUE','NIQE'};

%convert the matrices to tables with column names
table30 = array2table(metrics_arr30,'VariableNames',var_names);
table35 = array2table(metrics_arr35,'VariableNames',var_names);
table40 = array2table(metrics_arr40,'VariableNames',var_names);
table45 = array2table(metrics_arr45,'VariableNames',var_names);
table50 = array2table(metrics_arr50,'VariableNames',var_names);

%saving the results to an excel file
writetable(table30,'Gamma08_assessment_test.xlsx','Sheet','3.0');
writetable(table35,'Gamma08_assessment_test.xlsx','Sheet','3.5');
writetable(table40,'Gamma08_assessment_test.xlsx','Sheet','4.0');
writetable(table45,'Gamma08_assessment_test.xlsx','Sheet','4.5');
writetable(table50,'Gamma08_assessment_test.xlsx','Sheet','5.0');
disp('Done with gamma=0.8!')
disp('Done with Test Images!')
disp('Done!')
toc
```

Figure B.2.8: Assessment script for Gamma correction algorithm

As in the case of the linear transformation, we read the ground truth images as well as the results of the experiments, using Image Datastores, and apply the assessment function, which we described in the chapter 1. This process is done 3 times, once for each value of the correction constant, and we repeat it for the validation and test sets as well. We save the results in excel files, which we will use for the summarization.

Log Correction

The function with which we implement the logarithmic transformation is shown in figure B.2.9, which follows below. This function takes as input the LL image and the value of the constant c. First, it converts the data type of each pixel value to double, so that we can do arithmetic operations freely, and divides by 255 so that the values of each pixel are in the range [0,1]. Then we apply the logarithmic function, as presented in chapter 2, and multiply the result by 255, so that the values are in the range [0.255]. Finally, we convert the result to data type uint8, since we want to return an image, and not a simple matrix of values.

```
function NL_image = log_correction(LL_image,control_parameter)

%=========================================================================
% Function that implements the Log correction algorithm. It takes as
% input the Low Light image and the value of the control parameter and
% outputs the image after applying log correction.
% Author: Panagiotis Koutsaftis aivc21010
%=========================================================================

%converting to double needed for operations
LL_image = double(LL_image);

%normalizing the image values before applying log function
norm_LL = LL_image./255;

%applying the log correction function
NL_image = (log10(norm_LL.*control_parameter+1)).*255;

%converting back to uint8 cause we need to return an image
NL_image = uint8(NL_image);
```

Figure B.2.9: Implementation of the log correction algorithm

We will apply this function to all LL images, for each darkness level, which is done by applying the scripts shown in images B.2.10 and B.2.11. As we can see, each script reads the images for the darkness level we are interested in, using the image datastores of MATLAB. It then counts the number of images contained in the dataset, which is used to loop over those images. Inside the loop we read the respective LL Image of the datastore, apply the log correction function and save the result in a suitable folder. We apply this loop 3 times, once for each value of the parameter. In addition, we execute these scripts 5 times, once for each darkness level case, uncommenting each time the path of the darkness level we are interested in.

```
close all; clear; clc; format compact;


%loading the data
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\2.5\LL-2.5');
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\3.0\LL-3.0');
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\3.5\LL-3.5');
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\4.0\LL-4.0');
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\4.5\LL-4.5');
LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\5.0\LL-5.0');

%size of the dataset
files = size(LL_images.Files);
numOfImages = files(1);

%applying Logarithmic Transformation with control parameter = 1
for index=1:numOfImages
    LL_img = readimage(LL_images,index);
    image_name = char(LL_images.Files(index));
    %NL_img = readimage(NL_images,index);
    NL_log = log_correction(LL_img,1);
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\2.5\Log correction\control_1\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\3.0\Log correction\control_1\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\3.5\Log correction\control_1\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\4.0\Log correction\control_1\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\4.5\Log correction\control_1\',image_name((end-8):(end-4)),'.png'];
    file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\5.0\Log correction\control_1\',image_name((end-8):(end-4)),'.png'];
    imwrite(NL_log,file_path);
end

%applying Logarithmic Transformation with control parameter = 10
for index=1:numOfImages
    LL_img = readimage(LL_images,index);
    image_name = char(LL_images.Files(index));
    %NL_img = readimage(NL_images,index);
    NL_log = log_correction(LL_img,10);
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\2.5\Log correction\control_10\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\3.0\Log correction\control_10\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\3.5\Log correction\control_10\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\4.0\Log correction\control_10\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\4.5\Log correction\control_10\',image_name((end-8):(end-4)),'.png'];
    file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\5.0\Log correction\control_10\',image_name((end-8):(end-4)),'.png'];
    imwrite(NL_log,file_path);
end

%applying Logarithmic Transformation with control parameter = 50
for index=1:numOfImages
    LL_img = readimage(LL_images,index);
    image_name = char(LL_images.Files(index));
    %NL_img = readimage(NL_images,index);
    NL_log = log_correction(LL_img,50);
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\2.5\Log correction\control_50\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\3.0\Log correction\control_50\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\3.5\Log correction\control_50\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\4.0\Log correction\control_50\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\4.5\Log correction\control_50\',image_name((end-8):(end-4)),'.png'];
    file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\5.0\Log correction\control_50\',image_name((end-8):(end-4)),'.png'];
    imwrite(NL_log,file_path);
end
```

Figure B.2.10: script for applying log correction to training dataset per darkness level

```
close all; clear; clc; format compact;


%=========================================================================
%                          VALIDATION IMAGES
%=========================================================================

%reading the Low Light Data (VALIDATION)
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\3.0\LL-3.0');
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\3.5\LL-3.5');
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\4.0\LL-4.0');
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\4.5\LL-4.5');
LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\5.0\LL-5.0');

%size of the dataset
files = size(LL_images.Files);
numOfImages = files(1);

disp('Working on validation images...')
%applying Gamma Transformation with gamma = 0.1
for index=1:numOfImages
    LL_img = readimage(LL_images,index);
    image_name = char(LL_images.Files(index));
    %NL_img = readimage(NL_images,index);
    NL_gam = gamma_correction(LL_img,0.1);
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\3.0\Gamma correction\gamma_01\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\3.5\Gamma correction\gamma_01\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\4.0\Gamma correction\gamma_01\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\4.5\Gamma correction\gamma_01\',image_name((end-8):(end-4)),'.png'];
    file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\5.0\Gamma correction\gamma_01\',image_name((end-8):(end-4)),'.png'];
    imwrite(NL_gam,file_path);
end
disp('Done with gamma=0.1!')

%applying Gamma Transformation with gamma = 0.3
for index=1:numOfImages
    LL_img = readimage(LL_images,index);
    image_name = char(LL_images.Files(index));
    %NL_img = readimage(NL_images,index);
    NL_gam = gamma_correction(LL_img,0.3);
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\3.0\Gamma correction\gamma_03\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\3.5\Gamma correction\gamma_03\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\4.0\Gamma correction\gamma_03\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\4.5\Gamma correction\gamma_03\',image_name((end-8):(end-4)),'.png'];
    file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\5.0\Gamma correction\gamma_03\',image_name((end-8):(end-4)),'.png'];
    imwrite(NL_gam,file_path);
end
disp('Done with gamma=0.3!')

%applying Gamma Transformation with gamma = 0.8
for index=1:numOfImages
    LL_img = readimage(LL_images,index);
    image_name = char(LL_images.Files(index));
    %NL_img = readimage(NL_images,index);
    NL_gam = gamma_correction(LL_img,0.8);
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\3.0\Gamma correction\gamma_08\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\3.5\Gamma correction\gamma_08\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\4.0\Gamma correction\gamma_08\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\4.5\Gamma correction\gamma_08\',image_name((end-8):(end-4)),'.png'];
    file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\5.0\Gamma correction\gamma_08\',image_name((end-8):(end-4)),'.png'];
    imwrite(NL_gam,file_path);
end
disp('Done with gamma=0.8!')
disp('Done with validation images!')

%=========================================================================
%                              TEST IMAGES
%=========================================================================
%reading the Low Light Data (TEST)
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\3.0\LL-3.0');
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\3.5\LL-3.5');
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\4.0\LL-4.0');
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\4.5\LL-4.5');
LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\5.0\LL-5.0');

%size of the dataset
files = size(LL_images.Files);
numOfImages = files(1);

disp('Working on Test images...')
%applying Gamma Transformation with gamma = 0.1
for index=1:numOfImages
    LL_img = readimage(LL_images,index);
    image_name = char(LL_images.Files(index));
    %NL_img = readimage(NL_images,index);
    NL_gam = gamma_correction(LL_img,0.1);
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\3.0\Gamma correction\gamma_01\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\3.5\Gamma correction\gamma_01\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\4.0\Gamma correction\gamma_01\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\4.5\Gamma correction\gamma_01\',image_name((end-8):(end-4)),'.png'];
    file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\5.0\Gamma correction\gamma_01\',image_name((end-8):(end-4)),'.png'];
    imwrite(NL_gam,file_path);
end
disp('Done with gamma=0.1!')

%applying Gamma Transformation with gamma = 0.3
for index=1:numOfImages
    LL_img = readimage(LL_images,index);
    image_name = char(LL_images.Files(index));
    %NL_img = readimage(NL_images,index);
    NL_gam = gamma_correction(LL_img,0.3);
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\3.0\Gamma correction\gamma_03\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\3.5\Gamma correction\gamma_03\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\4.0\Gamma correction\gamma_03\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\4.5\Gamma correction\gamma_03\',image_name((end-8):(end-4)),'.png'];
    file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\5.0\Gamma correction\gamma_03\',image_name((end-8):(end-4)),'.png'];
    imwrite(NL_gam,file_path);
end
disp('Done with gamma=0.3!')
```

```
%applying Gamma Transformation with gamma = 0.8
for index=1:numOfImages
    LL_img = readimage(LL_images,index);
    image_name = char(LL_images.Files(index));
    %NL_img = readimage(NL_images,index);
    NL_gam = gamma_correction(LL_img,0.8);
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\3.0\Gamma correction\gamma_08\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\3.5\Gamma correction\gamma_08\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\4.0\Gamma correction\gamma_08\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\4.5\Gamma correction\gamma_08\',image_name((end-8):(end-4)),'.png'];
    file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\5.0\Gamma correction\gamma_08\',image_name((end-8):(end-4)),'.png'];
    imwrite(NL_gam,file_path);
end
disp('Done with gamma=0.8!')
disp('Done with test images!')
disp('Done!')
```

Figure B.2.11: script for applying log correction to test dataset per darkness level

After this process is completed, we will have the experimental results available. Based on these we will evaluate the performance of the method, which is done by applying the script shown in figure B.2.12. We read the ground truth images and experimental results, using Image Datastores, and apply the assessment function. The results of the function are saved in an excel file, with an appropriate name, which has a sheet for each darkness level. This procedure is applied for each value of the control constant, and for each set (training, validation and test), so at the end we will have 9 excel files that we can work with.

```
%==========================================================================
%                           TRAINING IMAGES
%==========================================================================
disp('Working on Training Images...')

%Ground Truth images
NL_images30 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\3.0\NL-3.0');
NL_images35 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\3.5\NL-3.5');
NL_images40 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\4.0\NL-4.0');
NL_images45 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\4.5\NL-4.5');
NL_images50 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\5.0\NL-5.0');

%Images from Log correction with control=1
NL_exp_res30 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\3.0\Log correction\control_1');
NL_exp_res35 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\3.5\Log correction\control_1');
NL_exp_res40 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\4.0\Log correction\control_1');
NL_exp_res45 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\4.5\Log correction\control_1');
NL_exp_res50 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\5.0\Log correction\control_1');

%assessment for 3.0 image dataset
metrics_arr30 = assessment(NL_images30,NL_exp_res30);
%assessment for 3.5 image dataset
metrics_arr35 = assessment(NL_images35,NL_exp_res35);
%assessment for 4.0 image dataset
metrics_arr40 = assessment(NL_images40,NL_exp_res40);
%assessment for 4.5 image dataset
metrics_arr45 = assessment(NL_images45,NL_exp_res45);
%assessment for 5.0 image dataset
metrics_arr50 = assessment(NL_images50,NL_exp_res50);

%columns titles
var_names = {'MSE','PSNR','SSIM','MV','STD','BRISQUE','NIQE'};

%convert the matrices to tables with column names
table30 = array2table(metrics_arr30,'VariableNames',var_names);
table35 = array2table(metrics_arr35,'VariableNames',var_names);
table40 = array2table(metrics_arr40,'VariableNames',var_names);
table45 = array2table(metrics_arr45,'VariableNames',var_names);
table50 = array2table(metrics_arr50,'VariableNames',var_names);

%saving the results to an excel file
writetable(table30,'LOG_control1_assessment_training.xlsx','Sheet','3.0');
writetable(table35,'LOG_control1_assessment_training.xlsx','Sheet','3.5');
writetable(table40,'LOG_control1_assessment_training.xlsx','Sheet','4.0');
writetable(table45,'LOG_control1_assessment_training.xlsx','Sheet','4.5');
writetable(table50,'LOG_control1_assessment_training.xlsx','Sheet','5.0');
disp('Done with control=1!')
```

```
%Images from Log correction with control=10
NL_exp_res30 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\3.0\Log correction\control_10');
NL_exp_res35 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\3.5\Log correction\control_10');
NL_exp_res40 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\4.0\Log correction\control_10');
NL_exp_res45 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\4.5\Log correction\control_10');
NL_exp_res50 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\5.0\Log correction\control_10');

%assessment for 3.0 image dataset
metrics_arr30 = assessment(NL_images30,NL_exp_res30);
%assessment for 3.5 image dataset
metrics_arr35 = assessment(NL_images35,NL_exp_res35);
%assessment for 4.0 image dataset
metrics_arr40 = assessment(NL_images40,NL_exp_res40);
%assessment for 4.5 image dataset
metrics_arr45 = assessment(NL_images45,NL_exp_res45);
%assessment for 5.0 image dataset
metrics_arr50 = assessment(NL_images50,NL_exp_res50);

%columns titles
var_names = {'MSE','PSNR','SSIM','MV','STD','BRISQUE','NIQE'};

%convert the matrices to tables with column names
table30 = array2table(metrics_arr30,'VariableNames',var_names);
table35 = array2table(metrics_arr35,'VariableNames',var_names);
table40 = array2table(metrics_arr40,'VariableNames',var_names);
table45 = array2table(metrics_arr45,'VariableNames',var_names);
table50 = array2table(metrics_arr50,'VariableNames',var_names);

%saving the results to an excel file
writetable(table30,'LOG_control10_assessment_training.xlsx','Sheet','3.0');
writetable(table35,'LOG_control10_assessment_training.xlsx','Sheet','3.5');
writetable(table40,'LOG_control10_assessment_training.xlsx','Sheet','4.0');
writetable(table45,'LOG_control10_assessment_training.xlsx','Sheet','4.5');
writetable(table50,'LOG_control10_assessment_training.xlsx','Sheet','5.0');
disp('Done with control=10!')

%Images from Log correction with control=50
NL_exp_res30 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\3.0\Log correction\control_50');
NL_exp_res35 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\3.5\Log correction\control_50');
NL_exp_res40 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\4.0\Log correction\control_50');
NL_exp_res45 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\4.5\Log correction\control_50');
NL_exp_res50 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\5.0\Log correction\control_50');

%assessment for 3.0 image dataset
metrics_arr30 = assessment(NL_images30,NL_exp_res30);
%assessment for 3.5 image dataset
metrics_arr35 = assessment(NL_images35,NL_exp_res35);
```

```
%assessment for 4.0 image dataset
metrics_arr40 = assessment(NL_images40,NL_exp_res40);
%assessment for 4.5 image dataset
metrics_arr45 = assessment(NL_images45,NL_exp_res45);
%assessment for 5.0 image dataset
metrics_arr50 = assessment(NL_images50,NL_exp_res50);

%columns titles
var_names = {'MSE','PSNR','SSIM','MV','STD','BRISQUE','NIQE'};

%convert the matrices to tables with column names
table30 = array2table(metrics_arr30,'VariableNames',var_names);
table35 = array2table(metrics_arr35,'VariableNames',var_names);
table40 = array2table(metrics_arr40,'VariableNames',var_names);
table45 = array2table(metrics_arr45,'VariableNames',var_names);
table50 = array2table(metrics_arr50,'VariableNames',var_names);

%saving the results to an excel file
writetable(table30,'LOG_control50_assessment_training.xlsx','Sheet','3.0');
writetable(table35,'LOG_control50_assessment_training.xlsx','Sheet','3.5');
writetable(table40,'LOG_control50_assessment_training.xlsx','Sheet','4.0');
writetable(table45,'LOG_control50_assessment_training.xlsx','Sheet','4.5');
writetable(table50,'LOG_control50_assessment_training.xlsx','Sheet','5.0');
disp('Done with control=50!')
disp('Done with Training Images!')

%========================================================================
%                          VALIDATION IMAGES
%========================================================================
disp('Working with Validation Images...')

%Ground Truth images
NL_images30 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\3.0\NL-3.0');
NL_images35 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\3.5\NL-3.5');
NL_images40 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\4.0\NL-4.0');
NL_images45 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\4.5\NL-4.5');
NL_images50 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\5.0\NL-5.0');

%Images from Log correction with control=1
NL_exp_res30 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\3.0\Log correction\control_1');
NL_exp_res35 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\3.5\Log correction\control_1');
NL_exp_res40 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\4.0\Log correction\control_1');
NL_exp_res45 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\4.5\Log correction\control_1');
NL_exp_res50 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\5.0\Log correction\control_1');

%assessment for 3.0 image dataset
metrics_arr30 = assessment(NL_images30,NL_exp_res30);
%assessment for 3.5 image dataset
metrics_arr35 = assessment(NL_images35,NL_exp_res35);
%assessment for 4.0 image dataset
metrics_arr40 = assessment(NL_images40,NL_exp_res40);
%assessment for 4.5 image dataset
metrics_arr45 = assessment(NL_images45,NL_exp_res45);
%assessment for 5.0 image dataset
metrics_arr50 = assessment(NL_images50,NL_exp_res50);

%columns titles
var_names = {'MSE','PSNR','SSIM','MV','STD','BRISQUE','NIQE'};

%convert the matrices to tables with column names
table30 = array2table(metrics_arr30,'VariableNames',var_names);
table35 = array2table(metrics_arr35,'VariableNames',var_names);
table40 = array2table(metrics_arr40,'VariableNames',var_names);
table45 = array2table(metrics_arr45,'VariableNames',var_names);
table50 = array2table(metrics_arr50,'VariableNames',var_names);

%saving the results to an excel file
writetable(table30,'LOG_control1_assessment_validation.xlsx','Sheet','3.0');
writetable(table35,'LOG_control1_assessment_validation.xlsx','Sheet','3.5');
writetable(table40,'LOG_control1_assessment_validation.xlsx','Sheet','4.0');
writetable(table45,'LOG_control1_assessment_validation.xlsx','Sheet','4.5');
writetable(table50,'LOG_control1_assessment_validation.xlsx','Sheet','5.0');
disp('Done with control=1!')

%Images from Log correction with control=10
NL_exp_res30 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\3.0\Log correction\control_10');
NL_exp_res35 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\3.5\Log correction\control_10');
NL_exp_res40 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\4.0\Log correction\control_10');
NL_exp_res45 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\4.5\Log correction\control_10');
NL_exp_res50 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\5.0\Log correction\control_10');

%assessment for 3.0 image dataset
metrics_arr30 = assessment(NL_images30,NL_exp_res30);
%assessment for 3.5 image dataset
metrics_arr35 = assessment(NL_images35,NL_exp_res35);
%assessment for 4.0 image dataset
metrics_arr40 = assessment(NL_images40,NL_exp_res40);
%assessment for 4.5 image dataset
metrics_arr45 = assessment(NL_images45,NL_exp_res45);
%assessment for 5.0 image dataset
metrics_arr50 = assessment(NL_images50,NL_exp_res50);
```

```
%columns titles
var_names = {'MSE','PSNR','SSIM','MV','STD','BRISQUE','NIQE'};

%convert the matrices to tables with column names
table30 = array2table(metrics_arr30,'VariableNames',var_names);
table35 = array2table(metrics_arr35,'VariableNames',var_names);
table40 = array2table(metrics_arr40,'VariableNames',var_names);
table45 = array2table(metrics_arr45,'VariableNames',var_names);
table50 = array2table(metrics_arr50,'VariableNames',var_names);

%saving the results to an excel file
writetable(table30,'LOG_control10_assessment_validation.xlsx','Sheet','3.0');
writetable(table35,'LOG_control10_assessment_validation.xlsx','Sheet','3.5');
writetable(table40,'LOG_control10_assessment_validation.xlsx','Sheet','4.0');
writetable(table45,'LOG_control10_assessment_validation.xlsx','Sheet','4.5');
writetable(table50,'LOG_control10_assessment_validation.xlsx','Sheet','5.0');
disp('Done with control=10!')

%Images from Log correction with control=50
NL_exp_res30 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\3.0\Log correction\control_50');
NL_exp_res35 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\3.5\Log correction\control_50');
NL_exp_res40 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\4.0\Log correction\control_50');
NL_exp_res45 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\4.5\Log correction\control_50');
NL_exp_res50 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\5.0\Log correction\control_50');

%assessment for 3.0 image dataset
metrics_arr30 = assessment(NL_images30,NL_exp_res30);
%assessment for 3.5 image dataset
metrics_arr35 = assessment(NL_images35,NL_exp_res35);
%assessment for 4.0 image dataset
metrics_arr40 = assessment(NL_images40,NL_exp_res40);
%assessment for 4.5 image dataset
metrics_arr45 = assessment(NL_images45,NL_exp_res45);
%assessment for 5.0 image dataset
metrics_arr50 = assessment(NL_images50,NL_exp_res50);

%columns titles
var_names = {'MSE','PSNR','SSIM','MV','STD','BRISQUE','NIQE'};

%convert the matrices to tables with column names
table30 = array2table(metrics_arr30,'VariableNames',var_names);
table35 = array2table(metrics_arr35,'VariableNames',var_names);
table40 = array2table(metrics_arr40,'VariableNames',var_names);
table45 = array2table(metrics_arr45,'VariableNames',var_names);
table50 = array2table(metrics_arr50,'VariableNames',var_names);

%saving the results to an excel file
writetable(table30,'LOG_control50_assessment_validation.xlsx','Sheet','3.0');
writetable(table35,'LOG_control50_assessment_validation.xlsx','Sheet','3.5');
writetable(table40,'LOG_control50_assessment_validation.xlsx','Sheet','4.0');
writetable(table45,'LOG_control50_assessment_validation.xlsx','Sheet','4.5');
writetable(table50,'LOG_control50_assessment_validation.xlsx','Sheet','5.0');
disp('Done with control=1!')
disp('Done with Validation Images!')

%=========================================================================
%                              TEST IMAGES
%=========================================================================
disp('Working with Test Images...')

%Ground Truth images
NL_images30 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\3.0\NL-3.0');
NL_images35 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\3.5\NL-3.5');
NL_images40 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\4.0\NL-4.0');
NL_images45 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\4.5\NL-4.5');
NL_images50 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\5.0\NL-5.0');

%Images from Log correction with control=1
NL_exp_res30 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\3.0\Log correction\control_1');
NL_exp_res35 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\3.5\Log correction\control_1');
NL_exp_res40 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\4.0\Log correction\control_1');
NL_exp_res45 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\4.5\Log correction\control_1');
NL_exp_res50 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\5.0\Log correction\control_1');

%assessment for 3.0 image dataset
metrics_arr30 = assessment(NL_images30,NL_exp_res30);
%assessment for 3.5 image dataset
metrics_arr35 = assessment(NL_images35,NL_exp_res35);
%assessment for 4.0 image dataset
metrics_arr40 = assessment(NL_images40,NL_exp_res40);
%assessment for 4.5 image dataset
metrics_arr45 = assessment(NL_images45,NL_exp_res45);
%assessment for 5.0 image dataset
metrics_arr50 = assessment(NL_images50,NL_exp_res50);

%columns titles
var_names = {'MSE','PSNR','SSIM','MV','STD','BRISQUE','NIQE'};

%convert the matrices to tables with column names
table30 = array2table(metrics_arr30,'VariableNames',var_names);
table35 = array2table(metrics_arr35,'VariableNames',var_names);
table40 = array2table(metrics_arr40,'VariableNames',var_names);
table45 = array2table(metrics_arr45,'VariableNames',var_names);
table50 = array2table(metrics_arr50,'VariableNames',var_names);
```

```
%saving the results to an excel file
writetable(table30,'LOG_control1_assessment_test.xlsx','Sheet','3.0');
writetable(table35,'LOG_control1_assessment_test.xlsx','Sheet','3.5');
writetable(table40,'LOG_control1_assessment_test.xlsx','Sheet','4.0');
writetable(table45,'LOG_control1_assessment_test.xlsx','Sheet','4.5');
writetable(table50,'LOG_control1_assessment_test.xlsx','Sheet','5.0');
disp('Done with control=1!')

%Images from Log correction with control=10
NL_exp_res30 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\3.0\Log correction\control_10');
NL_exp_res35 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\3.5\Log correction\control_10');
NL_exp_res40 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\4.0\Log correction\control_10');
NL_exp_res45 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\4.5\Log correction\control_10');
NL_exp_res50 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\5.0\Log correction\control_10');

%assessment for 3.0 image dataset
metrics_arr30 = assessment(NL_images30,NL_exp_res30);
%assessment for 3.5 image dataset
metrics_arr35 = assessment(NL_images35,NL_exp_res35);
%assessment for 4.0 image dataset
metrics_arr40 = assessment(NL_images40,NL_exp_res40);
%assessment for 4.5 image dataset
metrics_arr45 = assessment(NL_images45,NL_exp_res45);
%assessment for 5.0 image dataset
metrics_arr50 = assessment(NL_images50,NL_exp_res50);

%columns titles
var_names = {'MSE','PSNR','SSIM','MV','STD','BRISQUE','NIQE'};

%convert the matrices to tables with column names
table30 = array2table(metrics_arr30,'VariableNames',var_names);
table35 = array2table(metrics_arr35,'VariableNames',var_names);
table40 = array2table(metrics_arr40,'VariableNames',var_names);
table45 = array2table(metrics_arr45,'VariableNames',var_names);
table50 = array2table(metrics_arr50,'VariableNames',var_names);

%saving the results to an excel file
writetable(table30,'LOG_control10_assessment_test.xlsx','Sheet','3.0');
writetable(table35,'LOG_control10_assessment_test.xlsx','Sheet','3.5');
writetable(table40,'LOG_control10_assessment_test.xlsx','Sheet','4.0');
writetable(table45,'LOG_control10_assessment_test.xlsx','Sheet','4.5');
writetable(table50,'LOG_control10_assessment_test.xlsx','Sheet','5.0');
disp('Done with control=10!')

%Images from Log correction with control=50
NL_exp_res30 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\3.0\Log correction\control_50');
NL_exp_res35 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\3.5\Log correction\control_50');
NL_exp_res40 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\4.0\Log correction\control_50');
NL_exp_res45 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\4.5\Log correction\control_50');
NL_exp_res50 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\5.0\Log correction\control_50');

%assessment for 3.0 image dataset
metrics_arr30 = assessment(NL_images30,NL_exp_res30);
%assessment for 3.5 image dataset
metrics_arr35 = assessment(NL_images35,NL_exp_res35);
%assessment for 4.0 image dataset
metrics_arr40 = assessment(NL_images40,NL_exp_res40);
%assessment for 4.5 image dataset
metrics_arr45 = assessment(NL_images45,NL_exp_res45);
%assessment for 5.0 image dataset
metrics_arr50 = assessment(NL_images50,NL_exp_res50);

%columns titles
var_names = {'MSE','PSNR','SSIM','MV','STD','BRISQUE','NIQE'};

%convert the matrices to tables with column names
table30 = array2table(metrics_arr30,'VariableNames',var_names);
table35 = array2table(metrics_arr35,'VariableNames',var_names);
table40 = array2table(metrics_arr40,'VariableNames',var_names);
table45 = array2table(metrics_arr45,'VariableNames',var_names);
table50 = array2table(metrics_arr50,'VariableNames',var_names);

%saving the results to an excel file
writetable(table30,'LOG_control50_assessment_test.xlsx','Sheet','3.0');
writetable(table35,'LOG_control50_assessment_test.xlsx','Sheet','3.5');
writetable(table40,'LOG_control50_assessment_test.xlsx','Sheet','4.0');
writetable(table45,'LOG_control50_assessment_test.xlsx','Sheet','4.5');
writetable(table50,'LOG_control50_assessment_test.xlsx','Sheet','5.0');
disp('Done with control=50!')
disp('Done with Test Images!')
disp('Done!')
toc
```

Figure B.2.12: Assessment script for Log correction algorithm

Histogram Equalization

The function with which we implement the histogram equalization is shown in figure B.2.13 below.

```
function NL_image = HistogramEqualization(LL_image)

%=========================================================================
% Function that implements the Histogram Equalization Algorithm.
% We are going to use MATLAB's histeq function for optimization reasons.
% The function expects a grayscale image as an input.
% For that reason we apply the function to each channel of the RGB image
% separately.
% Author: Panagiotis Koutsaftis aivc21010
%=========================================================================

%RGB image channels
Red = LL_image(:,:,1);
Green = LL_image(:,:,2);
Blue = LL_image(:,:,3);

%applying histreq
Red_he = histeq(Red);
Green_he = histeq(Green);
Blue_he = histeq(Blue);

%defining an empty matrix with the same size as the input image
NL = zeros(size(LL_image));
NL(:,:,1) = Red_he;
NL(:,:,2) = Green_he;
NL(:,:,3) = Blue_he;

%converting the result to uint8
NL_image = uint8(NL);
```

Figure B.2.13: Implementation of histogram equalization

In this particular case, we do not implement the method from scratch, but use the ready-made MATLAB function, as it will be much faster and optimized than any attempt at a custom implementation. The histeq function of MATLAB expects in its input a grayscale image, and for this reason we apply per-channel equalization of the input image. So, the function shown above accepts as input the LL image that we want to enhance. It then separates the channels of the input image, and applies histogram equalization to each of them separately. Then, we simply define a matrix with the same dimensions as the input image, and store there the

result of the histogram equalization. Finally, we convert this matrix to uint8 data type, since we want to return an image and not just a table of values.

We apply this function to each image of the data set with the scripts presented in Figures B.2.14 and B.2.15.

```
close all; clear; clc; format long;


%loading the data
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\2.5\LL-2.5');
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\3.0\LL-3.0');
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\3.5\LL-3.5');
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\4.0\LL-4.0');
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\4.5\LL-4.5');
LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\5.0\LL-5.0');

%size of the dataset
files = size(LL_images.Files);
numOfImages = files(1);

%applying Histogram Equalization
for index=1:numOfImages
    LL_img = readimage(LL_images,index);
    image_name = char(LL_images.Files(index));
    %NL_img = readimage(NL_images,index);
    NL_he = HistogramEqualization(LL_img);
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\2.5\HE\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\3.0\HE\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\3.5\HE\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\4.0\HE\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\4.5\HE\',image_name((end-8):(end-4)),'.png'];
    file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\5.0\HE\',image_name((end-8):(end-4)),'.png'];
    imwrite(NL_he,file_path);
end
```

Figure B.2.14: script for applying HE to training dataset per darkness level

As we can see, we read the images using Image Datastores, find the size of the set, information that we will use in the following loop. Inside the loop, we apply HE to each image in the dataset and save the result to an appropriate folder. We apply this script 5 times, once for each darkness level, uncommenting the appropriate line each time.

We apply the same procedure for the validation and test sets, as shown in figure B.2.15.

```
close all; clear; clc; format long;

%=========================================================================
%                           VALIDATION IMAGES
%=========================================================================
%reading the Low Light Data (VALIDATION)
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\3.0\LL-3.0');
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\3.5\LL-3.5');
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\4.0\LL-4.0');
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\4.5\LL-4.5');
LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\5.0\LL-5.0');

%size of the dataset
files = size(LL_images.Files);
numOfImages = files(1);

disp('Working on validation images...')
%applying Histogram Equalization
for index=1:numOfImages
    LL_img = readimage(LL_images,index);
    image_name = char(LL_images.Files(index));
    %NL_img = readimage(NL_images,index);
    NL_he = HistogramEqualization(LL_img);
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\3.0\HE\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\3.5\HE\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\4.0\HE\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\4.5\HE\',image_name((end-8):(end-4)),'.png'];
    file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\5.0\HE\',image_name((end-8):(end-4)),'.png'];
    imwrite(NL_he,file_path);
end
disp('Done with validation images!')
```

```
%=========================================================================
%                               TEST IMAGES
%=========================================================================
%reading the Low Light Data (TEST)
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\3.0\LL-3.0');
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\3.5\LL-3.5');
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\4.0\LL-4.0');
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\4.5\LL-4.5');
LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\5.0\LL-5.0');

%size of the dataset
files = size(LL_images.Files);
numOfImages = files(1);

disp('Working on Test images...')
%applying Histogram Equalization
for index=1:numOfImages
    LL_img = readimage(LL_images,index);
    image_name = char(LL_images.Files(index));
    %NL_img = readimage(NL_images,index);
    NL_he = HistogramEqualization(LL_img);
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\3.0\HE\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\3.5\HE\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\4.0\HE\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\4.5\HE\',image_name((end-8):(end-4)),'.png'];
    file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\5.0\HE\',image_name((end-8):(end-4)),'.png'];
    imwrite(NL_he,file_path);
end
disp('Done with test images!')
disp('Done!')
```

Figure B.2.15: script for applying HE to validation and test datasets per darkness level

Then we have to calculate the metric evaluations for the results obtained from the above, This is done with the script shown in figure B.2.16.

```
close all; clear; clc; format long;
tic
%=========================================================================
%=========================================================================
%                           TRAINING IMAGES
%=========================================================================
%=========================================================================
disp('Woriking on Training Images...')
%=========================================================================
%                           GROUND TRUTH IMAGES
%=========================================================================

NL_images30 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\3.0\NL-3.0');
NL_images35 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\3.5\NL-3.5');
NL_images40 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\4.0\NL-4.0');
NL_images45 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\4.5\NL-4.5');
NL_images50 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\5.0\NL-5.0');

%=========================================================================
%               IMAGES FROM THE HISTOGRAM EQUALIZATION METHOD
%=========================================================================

NL_exp_res30 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\3.0\HE');
NL_exp_res35 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\3.5\HE');
NL_exp_res40 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\4.0\HE');
NL_exp_res45 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\4.5\HE');
NL_exp_res50 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\5.0\HE');

%assessment for 3.0 image dataset
metrics_arr30 = assessment(NL_images30,NL_exp_res30);
%assessment for 3.5 image dataset
metrics_arr35 = assessment(NL_images35,NL_exp_res35);
%assessment for 4.0 image dataset
metrics_arr40 = assessment(NL_images40,NL_exp_res40);
%assessment for 4.5 image dataset
metrics_arr45 = assessment(NL_images45,NL_exp_res45);
%assessment for 5.0 image dataset
metrics_arr50 = assessment(NL_images50,NL_exp_res50);

%columns titles
var_names = {'MSE','PSNR','SSIM','MV','STD','BRISQUE','NIQE'};

%convert the matrices to tables with column names
table30 = array2table(metrics_arr30,'VariableNames',var_names);
table35 = array2table(metrics_arr35,'VariableNames',var_names);
table40 = array2table(metrics_arr40,'VariableNames',var_names);
table45 = array2table(metrics_arr45,'VariableNames',var_names);
table50 = array2table(metrics_arr50,'VariableNames',var_names);

%saving the results to an excel file
writetable(table30,'Histogram_Equalization_assessment_training.xlsx','Sheet','3.0');
writetable(table35,'Histogram_Equalization_assessment_training.xlsx','Sheet','3.5');
writetable(table40,'Histogram_Equalization_assessment_training.xlsx','Sheet','4.0');
writetable(table45,'Histogram_Equalization_assessment_training.xlsx','Sheet','4.5');
writetable(table50,'Histogram_Equalization_assessment_training.xlsx','Sheet','5.0');

disp('Done with Training Images!')

%=========================================================================
%=========================================================================
%                           VALIDATION IMAGES
%=========================================================================
%=========================================================================
disp('Working on Validation Images...')
%=========================================================================
%                           GROUND TRUTH IMAGES
%=========================================================================

NL_images30 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\3.0\NL-3.0');
NL_images35 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\3.5\NL-3.5');
NL_images40 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\4.0\NL-4.0');
NL_images45 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\4.5\NL-4.5');
NL_images50 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\5.0\NL-5.0');

%=========================================================================
%               IMAGES FROM THE HISTOGRAM EQUALIZATION METHOD
%=========================================================================

NL_exp_res30 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\3.0\HE');
NL_exp_res35 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\3.5\HE');
NL_exp_res40 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\4.0\HE');
NL_exp_res45 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\4.5\HE');
NL_exp_res50 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\5.0\HE');
```

```
%assessment for 3.0 image dataset
metrics_arr30 = assessment(NL_images30,NL_exp_res30);
%assessment for 3.5 image dataset
metrics_arr35 = assessment(NL_images35,NL_exp_res35);
%assessment for 4.0 image dataset
metrics_arr40 = assessment(NL_images40,NL_exp_res40);
%assessment for 4.5 image dataset
metrics_arr45 = assessment(NL_images45,NL_exp_res45);
%assessment for 5.0 image dataset
metrics_arr50 = assessment(NL_images50,NL_exp_res50);

%columns titles
var_names = {'MSE','PSNR','SSIM','MV','STD','BRISQUE','NIQE'};

%convert the matrices to tables with column names
table30 = array2table(metrics_arr30,'VariableNames',var_names);
table35 = array2table(metrics_arr35,'VariableNames',var_names);
table40 = array2table(metrics_arr40,'VariableNames',var_names);
table45 = array2table(metrics_arr45,'VariableNames',var_names);
table50 = array2table(metrics_arr50,'VariableNames',var_names);

%saving the results to an excel file
writetable(table30,'Histogram_Equalization_assessment_validation.xlsx','Sheet','3.0');
writetable(table35,'Histogram_Equalization_assessment_validation.xlsx','Sheet','3.5');
writetable(table40,'Histogram_Equalization_assessment_validation.xlsx','Sheet','4.0');
writetable(table45,'Histogram_Equalization_assessment_validation.xlsx','Sheet','4.5');
writetable(table50,'Histogram_Equalization_assessment_validation.xlsx','Sheet','5.0');

disp('Done with Validation Images!')

%=========================================================================
%=========================================================================
%                              TEST IMAGES
%=========================================================================
%=========================================================================
disp('Working on Test Images...')
%=========================================================================
%                              GROUND TRUTH IMAGES
%=========================================================================

NL_images30 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\3.0\NL-3.0');
NL_images35 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\3.5\NL-3.5');
NL_images40 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\4.0\NL-4.0');
NL_images45 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\4.5\NL-4.5');
NL_images50 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\5.0\NL-5.0');

%=========================================================================
%              IMAGES FROM THE HISTOGRAM EQUALIZATION METHOD
%=========================================================================

NL_exp_res30 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\3.0\HE');
NL_exp_res35 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\3.5\HE');
NL_exp_res40 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\4.0\HE');
NL_exp_res45 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\4.5\HE');
NL_exp_res50 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\5.0\HE');

%assessment for 3.0 image dataset
metrics_arr30 = assessment(NL_images30,NL_exp_res30);
%assessment for 3.5 image dataset
metrics_arr35 = assessment(NL_images35,NL_exp_res35);
%assessment for 4.0 image dataset
metrics_arr40 = assessment(NL_images40,NL_exp_res40);
%assessment for 4.5 image dataset
metrics_arr45 = assessment(NL_images45,NL_exp_res45);
%assessment for 5.0 image dataset
metrics_arr50 = assessment(NL_images50,NL_exp_res50);

%columns titles
var_names = {'MSE','PSNR','SSIM','MV','STD','BRISQUE','NIQE'};

%convert the matrices to tables with column names
table30 = array2table(metrics_arr30,'VariableNames',var_names);
table35 = array2table(metrics_arr35,'VariableNames',var_names);
table40 = array2table(metrics_arr40,'VariableNames',var_names);
table45 = array2table(metrics_arr45,'VariableNames',var_names);
table50 = array2table(metrics_arr50,'VariableNames',var_names);

%saving the results to an excel file
writetable(table30,'Histogram_Equalization_assessment_test.xlsx','Sheet','3.0');
writetable(table35,'Histogram_Equalization_assessment_test.xlsx','Sheet','3.5');
writetable(table40,'Histogram_Equalization_assessment_test.xlsx','Sheet','4.0');
writetable(table45,'Histogram_Equalization_assessment_test.xlsx','Sheet','4.5');
writetable(table50,'Histogram_Equalization_assessment_test.xlsx','Sheet','5.0');

disp('Done with Test Images!')
disp('Done!')
toc
```

Figure B.2.16: Assessment script for HE

We read the ground truth images and experimental results, using Image Datastores, and apply the assessment function. We save the results in an excel file with a suitable name, with each sheet corresponding to a darkness level. We apply this process for all three cases (training, validation and test sets), so at the end of the process we will have 3 excel files that we can work with.

Single Scale Retinex

The function that implements the Single Scale Retinex method is shown in figure B.2.17 below. The function takes as input the LL image we want to enhance, as well as the value of the constant c needed for the surround function. Then we convert the image into a double data type, which is needed for the operations, and we normalize the values by dividing by 255. At this point, let us comment that we also add a very small value, here 0.01, because then we will apply a logarithmic function, which at 0 goes to infinity, and we want to avoid that. Then we construct an empty table, with the same dimensions as these images, and there we store the values of the surround function. Then, we apply the operation we saw in chapter 2 to calculate R, on each channel of the image, and convert the final result to uint8, since we want to return an image and not just an array of values.

We will apply this function to all images in the dataset. This is done with the scripts presented in images B.2.18 and B.2.19.

```
function NL_image = SingleScaleRetinex(LL_image,c)

%========================================================================
%   This function implements the Single Scale Retinex method for low
%   light image enhancement reasons. It takes as input the low light image
%   and the surround parameter c, used by the surround gaussian function,
%   and outputs the enhanced normal light image.
%   Article Reference:
%   D. J. Jobson, Z. Rahman, and G. A. Woodell,
%   ``Properties and performance of a center/surround retinex,''
%   IEEE Trans. Image Process., vol. 6, no. 3, pp. 451□462, Mar. 1997.
%
%   Function Author: Panagiotis Koutsaftis
%
%========================================================================

%size of the input image
[rows,columns,bands] = size(LL_image);

%converting the image to double for calculations
LL_double = double(LL_image);
%normalizing the image by dividing with 255, adding 0.01 for the log
LL_double = (LL_double/255)+0.01;

%center of the image
cent = ceil(rows/2);
%initializing the values of thee filter
filt = zeros(rows,columns);
%initializing a sum needed for normalization reasons
summ = 0;
%creating the gaussian filter
for i=1:rows
    for j=1:columns
        r = ((cent-i)^2+(cent-j)^2);
        filt(i,j) = exp(-(r/(c^2)));
        summ = summ + filt(i,j);
    end
end
%normalizing the values of the filter
filt = filt/summ;

%initializing the output normal light image
NL_image = zeros(rows,columns,bands);

%calculating the retinex for each color band
for band=1:bands
    %convolution with the filter
    conv = imfilter(LL_double(:,:,band),filt,'replicate','same');
    %retinex output
    R = log(LL_double(:,:,band))-log(conv);
    %min value
    R_min = min(min(R));
    R_max = max(max(R));
    %normalizing to [0,255]
    R_t = 255*((R-R_min)./(R_max-R_min));
    %final band output
    NL_image(:,:,band) = R_t;
end

%converting to uint8
NL_image = uint8(NL_image);

```

Figure B.2.17: Implementation of SSR

```
close all; clear; clc; format compact;


%loading the data
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\3.0\LL-3.0');
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\3.5\LL-3.5');
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\4.0\LL-4.0');
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\4.5\LL-4.5');
LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\5.0\LL-5.0');

%size of the dataset
files = size(LL_images.Files);
numOfImages = files(1);

%applying SSR with surround parameter c=10
for index=1:numOfImages
    LL_img = readimage(LL_images,index);
    image_name = char(LL_images.Files(index));
    %NL_img = readimage(NL_images,index);
    NL_SSR = SingleScaleRetinex(LL_img,10);
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\3.0\SSR\c_10\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\3.5\SSR\c_10\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\4.0\SSR\c_10\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\4.5\SSR\c_10\',image_name((end-8):(end-4)),'.png'];
    file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\5.0\SSR\c_10\',image_name((end-8):(end-4)),'.png'];
    imwrite(NL_SSR,file_path);
end


%applying SSR with surround parameter c=120
for index=1:numOfImages
    LL_img = readimage(LL_images,index);
    image_name = char(LL_images.Files(index));
    %NL_img = readimage(NL_images,index);
    NL_SSR = SingleScaleRetinex(LL_img,120);
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\3.0\SSR\c_120\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\3.5\SSR\c_120\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\4.0\SSR\c_120\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\4.5\SSR\c_120\',image_name((end-8):(end-4)),'.png'];
    file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\5.0\SSR\c_120\',image_name((end-8):(end-4)),'.png'];
    imwrite(NL_SSR,file_path);
end


%applying SSR with surround parameter c=400
for index=1:numOfImages
    LL_img = readimage(LL_images,index);
    image_name = char(LL_images.Files(index));
    %NL_img = readimage(NL_images,index);
    NL_SSR = SingleScaleRetinex(LL_img,400);
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\3.0\SSR\c_400\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\3.5\SSR\c_400\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\4.0\SSR\c_400\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\4.5\SSR\c_400\',image_name((end-8):(end-4)),'.png'];
    file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\5.0\SSR\c_400\',image_name((end-8):(end-4)),'.png'];
    imwrite(NL_SSR,file_path);
end
```

Figure B.2.18: script for applying SSR to training dataset per darkness level

```
close all; clear; clc; format long;

%=========================================================================
%                            VALIDATION IMAGES
%=========================================================================
%reading the Low Light Data (VALIDATION)
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\3.0\LL-3.0');
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\3.5\LL-3.5');
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\4.0\LL-4.0');
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\4.5\LL-4.5');
LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\5.0\LL-5.0');

%size of the dataset
files = size(LL_images.Files);
numOfImages = files(1);

disp('Working on validation images...')
%applying SSR with surround parameter c=10
for index=1:numOfImages
    LL_img = readimage(LL_images,index);
    image_name = char(LL_images.Files(index));
    %NL_img = readimage(NL_images,index);
    NL_SSR = SingleScaleRetinex(LL_img,10);
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\3.0\SSR\c_10\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\3.5\SSR\c_10\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\4.0\SSR\c_10\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\4.5\SSR\c_10\',image_name((end-8):(end-4)),'.png'];
    file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\5.0\SSR\c_10\',image_name((end-8):(end-4)),'.png'];
    imwrite(NL_SSR,file_path);
end
disp('Done with parameter c=10!')

%applying SSR with surround parameter c=120
for index=1:numOfImages
    LL_img = readimage(LL_images,index);
    image_name = char(LL_images.Files(index));
    %NL_img = readimage(NL_images,index);
    NL_SSR = SingleScaleRetinex(LL_img,120);
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\3.0\SSR\c_120\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\3.5\SSR\c_120\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\4.0\SSR\c_120\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\4.5\SSR\c_120\',image_name((end-8):(end-4)),'.png'];
    file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\5.0\SSR\c_120\',image_name((end-8):(end-4)),'.png'];
    imwrite(NL_SSR,file_path);
end
disp('Done with parameter c=120!')

%applying SSR with surround parameter c=400
for index=1:numOfImages
    LL_img = readimage(LL_images,index);
    image_name = char(LL_images.Files(index));
    %NL_img = readimage(NL_images,index);
    NL_SSR = SingleScaleRetinex(LL_img,400);
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\3.0\SSR\c_400\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\3.5\SSR\c_400\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\4.0\SSR\c_400\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\4.5\SSR\c_400\',image_name((end-8):(end-4)),'.png'];
    file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\5.0\SSR\c_400\',image_name((end-8):(end-4)),'.png'];
    imwrite(NL_SSR,file_path);
end
disp('Done with parameter c=400!')
disp('Done with validation data!')

%=========================================================================
%                                TEST IMAGES
%=========================================================================
%reading the Low Light Data (TEST)
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\3.0\LL-3.0');
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\3.5\LL-3.5');
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\4.0\LL-4.0');
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\4.5\LL-4.5');
LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\5.0\LL-5.0');

%size of the dataset
files = size(LL_images.Files);
numOfImages = files(1);

disp('Working on Test images...')
%applying SSR with surround parameter c=10
for index=1:numOfImages
    LL_img = readimage(LL_images,index);
    image_name = char(LL_images.Files(index));
    %NL_img = readimage(NL_images,index);
    NL_SSR = SingleScaleRetinex(LL_img,10);
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\3.0\SSR\c_10\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\3.5\SSR\c_10\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\4.0\SSR\c_10\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\4.5\SSR\c_10\',image_name((end-8):(end-4)),'.png'];
    file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\5.0\SSR\c_10\',image_name((end-8):(end-4)),'.png'];
    imwrite(NL_SSR,file_path);
end
disp('Done with parameter c=10!')

%applying SSR with surround parameter c=120
for index=1:numOfImages
    LL_img = readimage(LL_images,index);
    image_name = char(LL_images.Files(index));
    %NL_img = readimage(NL_images,index);
    NL_SSR = SingleScaleRetinex(LL_img,120);
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\3.0\SSR\c_120\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\3.5\SSR\c_120\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\4.0\SSR\c_120\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\4.5\SSR\c_120\',image_name((end-8):(end-4)),'.png'];
    file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\5.0\SSR\c_120\',image_name((end-8):(end-4)),'.png'];
    imwrite(NL_SSR,file_path);
end
disp('Done with parameter c=120!')

%applying SSR with surround parameter c=400
for index=1:numOfImages
    LL_img = readimage(LL_images,index);
    image_name = char(LL_images.Files(index));
    %NL_img = readimage(NL_images,index);
    NL_SSR = SingleScaleRetinex(LL_img,400);
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\3.0\SSR\c_400\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\3.5\SSR\c_400\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\4.0\SSR\c_400\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\4.5\SSR\c_400\',image_name((end-8):(end-4)),'.png'];
    file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\5.0\SSR\c_400\',image_name((end-8):(end-4)),'.png'];
    imwrite(NL_SSR,file_path);
end
disp('Done with parameter c=400!')
disp('Done with test data!')
disp('Done!')
```

Figure B.2.19: script for applying SSR to validation and test datasets per darkness level

In both cases we read the LL images using Image Datastores, and apply the SSR function to each image in the respective dataset. Note that since there is no ideal value of the constant c, we apply the method 3 times, for 3 different values of the constant. The results are saved in a folder with an appropriate name. In addition, we apply the method per darkness level, so we run the script 5 times, once for each darkness level, uncommenting the appropriate line each time. We apply this procedure for all sets (training, validation and test).

After applying this script, we have at our disposal the experimental results per darkness level and per value of the constant c. We will use these images to calculate performance metric values. This is done with the script shown in Figure B.2.20.

```
close all; clear; clc; format long;
tic

%==========================================================================
%                           TRAINING IMAGES
%==========================================================================
disp('Working on Training Images...')

%Ground Truth images
NL_images30 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\3.0\NL-3.0');
NL_images35 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\3.5\NL-3.5');
NL_images40 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\4.0\NL-4.0');
NL_images45 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\4.5\NL-4.5');
NL_images50 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\5.0\NL-5.0');

%Images from SSR with c=10
NL_exp_res30 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\3.0\SSR\c_10');
NL_exp_res35 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\3.5\SSR\c_10');
NL_exp_res40 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\4.0\SSR\c_10');
NL_exp_res45 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\4.5\SSR\c_10');
NL_exp_res50 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\5.0\SSR\c_10');

%assessment for 3.0 image dataset
metrics_arr30 = assessment(NL_images30,NL_exp_res30);
%assessment for 3.5 image dataset
metrics_arr35 = assessment(NL_images35,NL_exp_res35);
%assessment for 4.0 image dataset
metrics_arr40 = assessment(NL_images40,NL_exp_res40);
%assessment for 4.5 image dataset
metrics_arr45 = assessment(NL_images45,NL_exp_res45);
%assessment for 5.0 image dataset
metrics_arr50 = assessment(NL_images50,NL_exp_res50);

%columns titles
var_names = {'MSE','PSNR','SSIM','MV','STD','BRISQUE','NIQE'};

%convert the matrices to tables with column names
table30 = array2table(metrics_arr30,'VariableNames',var_names);
table35 = array2table(metrics_arr35,'VariableNames',var_names);
table40 = array2table(metrics_arr40,'VariableNames',var_names);
table45 = array2table(metrics_arr45,'VariableNames',var_names);
table50 = array2table(metrics_arr50,'VariableNames',var_names);
```

```
%saving the results to an excel file
writetable(table30,'SSR_c10_assessment_training.xlsx','Sheet','3.0');
writetable(table35,'SSR_c10_assessment_training.xlsx','Sheet','3.5');
writetable(table40,'SSR_c10_assessment_training.xlsx','Sheet','4.0');
writetable(table45,'SSR_c10_assessment_training.xlsx','Sheet','4.5');
writetable(table50,'SSR_c10_assessment_training.xlsx','Sheet','5.0');
disp('Done with c=10!')

%Images from SSR with c=120
NL_exp_res30 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\3.0\SSR\c_120');
NL_exp_res35 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\3.5\SSR\c_120');
NL_exp_res40 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\4.0\SSR\c_120');
NL_exp_res45 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\4.5\SSR\c_120');
NL_exp_res50 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\5.0\SSR\c_120');

%assessment for 3.0 image dataset
metrics_arr30 = assessment(NL_images30,NL_exp_res30);
%assessment for 3.5 image dataset
metrics_arr35 = assessment(NL_images35,NL_exp_res35);
%assessment for 4.0 image dataset
metrics_arr40 = assessment(NL_images40,NL_exp_res40);
%assessment for 4.5 image dataset
metrics_arr45 = assessment(NL_images45,NL_exp_res45);
%assessment for 5.0 image dataset
metrics_arr50 = assessment(NL_images50,NL_exp_res50);

%columns titles
var_names = {'MSE','PSNR','SSIM','MV','STD','BRISQUE','NIQE'};

%convert the matrices to tables with column names
table30 = array2table(metrics_arr30,'VariableNames',var_names);
table35 = array2table(metrics_arr35,'VariableNames',var_names);
table40 = array2table(metrics_arr40,'VariableNames',var_names);
table45 = array2table(metrics_arr45,'VariableNames',var_names);
table50 = array2table(metrics_arr50,'VariableNames',var_names);

%saving the results to an excel file
writetable(table30,'SSR_c120_assessment_training.xlsx','Sheet','3.0');
writetable(table35,'SSR_c120_assessment_training.xlsx','Sheet','3.5');
writetable(table40,'SSR_c120_assessment_training.xlsx','Sheet','4.0');
writetable(table45,'SSR_c120_assessment_training.xlsx','Sheet','4.5');
writetable(table50,'SSR_c120_assessment_training.xlsx','Sheet','5.0');
disp('Done with c=120!')

%Images from SSR with c=400
NL_exp_res30 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\3.0\SSR\c_400');
NL_exp_res35 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\3.5\SSR\c_400');
NL_exp_res40 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\4.0\SSR\c_400');
NL_exp_res45 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\4.5\SSR\c_400');
NL_exp_res50 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\5.0\SSR\c_400');

%assessment for 3.0 image dataset
metrics_arr30 = assessment(NL_images30,NL_exp_res30);
%assessment for 3.5 image dataset
metrics_arr35 = assessment(NL_images35,NL_exp_res35);
%assessment for 4.0 image dataset
metrics_arr40 = assessment(NL_images40,NL_exp_res40);
%assessment for 4.5 image dataset
metrics_arr45 = assessment(NL_images45,NL_exp_res45);
%assessment for 5.0 image dataset
metrics_arr50 = assessment(NL_images50,NL_exp_res50);

%columns titles
var_names = {'MSE','PSNR','SSIM','MV','STD','BRISQUE','NIQE'};

%convert the matrices to tables with column names
table30 = array2table(metrics_arr30,'VariableNames',var_names);
table35 = array2table(metrics_arr35,'VariableNames',var_names);
table40 = array2table(metrics_arr40,'VariableNames',var_names);
table45 = array2table(metrics_arr45,'VariableNames',var_names);
table50 = array2table(metrics_arr50,'VariableNames',var_names);

%saving the results to an excel file
writetable(table30,'SSR_c400_assessment_training.xlsx','Sheet','3.0');
writetable(table35,'SSR_c400_assessment_training.xlsx','Sheet','3.5');
writetable(table40,'SSR_c400_assessment_training.xlsx','Sheet','4.0');
writetable(table45,'SSR_c400_assessment_training.xlsx','Sheet','4.5');
writetable(table50,'SSR_c400_assessment_training.xlsx','Sheet','5.0');
disp('Done with c=400!')
disp('Done with Training Images!')
```

```
%=========================================================================
%                            VALIDATION IMAGES
%=========================================================================
disp('Working with Validation Images...')

%Ground Truth images
NL_images30 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\3.0\NL-3.0');
NL_images35 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\3.5\NL-3.5');
NL_images40 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\4.0\NL-4.0');
NL_images45 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\4.5\NL-4.5');
NL_images50 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\5.0\NL-5.0');

%Images from SSR with c=10
NL_exp_res30 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\3.0\SSR\c_10');
NL_exp_res35 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\3.5\SSR\c_10');
NL_exp_res40 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\4.0\SSR\c_10');
NL_exp_res45 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\4.5\SSR\c_10');
NL_exp_res50 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\5.0\SSR\c_10');

%assessment for 3.0 image dataset
metrics_arr30 = assessment(NL_images30,NL_exp_res30);
%assessment for 3.5 image dataset
metrics_arr35 = assessment(NL_images35,NL_exp_res35);
%assessment for 4.0 image dataset
metrics_arr40 = assessment(NL_images40,NL_exp_res40);
%assessment for 4.5 image dataset
metrics_arr45 = assessment(NL_images45,NL_exp_res45);
%assessment for 5.0 image dataset
metrics_arr50 = assessment(NL_images50,NL_exp_res50);

%columns titles
var_names = {'MSE','PSNR','SSIM','MV','STD','BRISQUE','NIQE'};

%convert the matrices to tables with column names
table30 = array2table(metrics_arr30,'VariableNames',var_names);
table35 = array2table(metrics_arr35,'VariableNames',var_names);
table40 = array2table(metrics_arr40,'VariableNames',var_names);
table45 = array2table(metrics_arr45,'VariableNames',var_names);
table50 = array2table(metrics_arr50,'VariableNames',var_names);

%saving the results to an excel file
writetable(table30,'SSR_c10_assessment_validation.xlsx','Sheet','3.0');
writetable(table35,'SSR_c10_assessment_validation.xlsx','Sheet','3.5');
writetable(table40,'SSR_c10_assessment_validation.xlsx','Sheet','4.0');
writetable(table45,'SSR_c10_assessment_validation.xlsx','Sheet','4.5');
writetable(table50,'SSR_c10_assessment_validation.xlsx','Sheet','5.0');
disp('Done with c=10!')

%Images from SSR with c=120
NL_exp_res30 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\3.0\SSR\c_120');
NL_exp_res35 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\3.5\SSR\c_120');
NL_exp_res40 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\4.0\SSR\c_120');
NL_exp_res45 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\4.5\SSR\c_120');
NL_exp_res50 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\5.0\SSR\c_120');

%assessment for 3.0 image dataset
metrics_arr30 = assessment(NL_images30,NL_exp_res30);
%assessment for 3.5 image dataset
metrics_arr35 = assessment(NL_images35,NL_exp_res35);
%assessment for 4.0 image dataset
metrics_arr40 = assessment(NL_images40,NL_exp_res40);
%assessment for 4.5 image dataset
metrics_arr45 = assessment(NL_images45,NL_exp_res45);
%assessment for 5.0 image dataset
metrics_arr50 = assessment(NL_images50,NL_exp_res50);

%columns titles
var_names = {'MSE','PSNR','SSIM','MV','STD','BRISQUE','NIQE'};

%convert the matrices to tables with column names
table30 = array2table(metrics_arr30,'VariableNames',var_names);
table35 = array2table(metrics_arr35,'VariableNames',var_names);
table40 = array2table(metrics_arr40,'VariableNames',var_names);
table45 = array2table(metrics_arr45,'VariableNames',var_names);
table50 = array2table(metrics_arr50,'VariableNames',var_names);

%saving the results to an excel file
writetable(table30,'SSR_c120_assessment_validation.xlsx','Sheet','3.0');
writetable(table35,'SSR_c120_assessment_validation.xlsx','Sheet','3.5');
writetable(table40,'SSR_c120_assessment_validation.xlsx','Sheet','4.0');
writetable(table45,'SSR_c120_assessment_validation.xlsx','Sheet','4.5');
writetable(table50,'SSR_c120_assessment_validation.xlsx','Sheet','5.0');
disp('Done with c=120!')

%Images from SSR with c=400
NL_exp_res30 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\3.0\SSR\c_400');
NL_exp_res35 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\3.5\SSR\c_400');
NL_exp_res40 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\4.0\SSR\c_400');
NL_exp_res45 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\4.5\SSR\c_400');
NL_exp_res50 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\5.0\SSR\c_400');
```

```
%assessment for 3.0 image dataset
metrics_arr30 = assessment(NL_images30,NL_exp_res30);
%assessment for 3.5 image dataset
metrics_arr35 = assessment(NL_images35,NL_exp_res35);
%assessment for 4.0 image dataset
metrics_arr40 = assessment(NL_images40,NL_exp_res40);
%assessment for 4.5 image dataset
metrics_arr45 = assessment(NL_images45,NL_exp_res45);
%assessment for 5.0 image dataset
metrics_arr50 = assessment(NL_images50,NL_exp_res50);

%columns titles
var_names = {'MSE','PSNR','SSIM','MV','STD','BRISQUE','NIQE'};

%convert the matrices to tables with column names
table30 = array2table(metrics_arr30,'VariableNames',var_names);
table35 = array2table(metrics_arr35,'VariableNames',var_names);
table40 = array2table(metrics_arr40,'VariableNames',var_names);
table45 = array2table(metrics_arr45,'VariableNames',var_names);
table50 = array2table(metrics_arr50,'VariableNames',var_names);

%saving the results to an excel file
writetable(table30,'SSR_c400_assessment_validation.xlsx','Sheet','3.0');
writetable(table35,'SSR_c400_assessment_validation.xlsx','Sheet','3.5');
writetable(table40,'SSR_c400_assessment_validation.xlsx','Sheet','4.0');
writetable(table45,'SSR_c400_assessment_validation.xlsx','Sheet','4.5');
writetable(table50,'SSR_c400_assessment_validation.xlsx','Sheet','5.0');
disp('Done with c=400!')
disp('Done with Validation Images!')

%=========================================================================
%                              TEST IMAGES
%=========================================================================
disp('Working with Test Images...')

%Ground Truth images
NL_images30 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\3.0\NL-3.0');
NL_images35 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\3.5\NL-3.5');
NL_images40 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\4.0\NL-4.0');
NL_images45 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\4.5\NL-4.5');
NL_images50 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\5.0\NL-5.0');

%Images from SSR with c=10
NL_exp_res30 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\3.0\SSR\c_10');
NL_exp_res35 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\3.5\SSR\c_10');
NL_exp_res40 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\4.0\SSR\c_10');
NL_exp_res45 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\4.5\SSR\c_10');
NL_exp_res50 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\5.0\SSR\c_10');

%assessment for 3.0 image dataset
metrics_arr30 = assessment(NL_images30,NL_exp_res30);
%assessment for 3.5 image dataset
metrics_arr35 = assessment(NL_images35,NL_exp_res35);
%assessment for 4.0 image dataset
metrics_arr40 = assessment(NL_images40,NL_exp_res40);
%assessment for 4.5 image dataset
metrics_arr45 = assessment(NL_images45,NL_exp_res45);
%assessment for 5.0 image dataset
metrics_arr50 = assessment(NL_images50,NL_exp_res50);

%columns titles
var_names = {'MSE','PSNR','SSIM','MV','STD','BRISQUE','NIQE'};

%convert the matrices to tables with column names
table30 = array2table(metrics_arr30,'VariableNames',var_names);
table35 = array2table(metrics_arr35,'VariableNames',var_names);
table40 = array2table(metrics_arr40,'VariableNames',var_names);
table45 = array2table(metrics_arr45,'VariableNames',var_names);
table50 = array2table(metrics_arr50,'VariableNames',var_names);

%saving the results to an excel file
writetable(table30,'SSR_c10_assessment_test.xlsx','Sheet','3.0');
writetable(table35,'SSR_c10_assessment_test.xlsx','Sheet','3.5');
writetable(table40,'SSR_c10_assessment_test.xlsx','Sheet','4.0');
writetable(table45,'SSR_c10_assessment_test.xlsx','Sheet','4.5');
writetable(table50,'SSR_c10_assessment_test.xlsx','Sheet','5.0');
disp('Done with c=10!')
```

```
293
294     %Images from SSR with c=120
295     NL_exp_res30 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\3.0\SSR\c_120');
296     NL_exp_res35 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\3.5\SSR\c_120');
297     NL_exp_res40 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\4.0\SSR\c_120');
298     NL_exp_res45 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\4.5\SSR\c_120');
299     NL_exp_res50 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\5.0\SSR\c_120');
300
301     %assessment for 3.0 image dataset
302     metrics_arr30 = assessment(NL_images30,NL_exp_res30);
303     %assessment for 3.5 image dataset
304     metrics_arr35 = assessment(NL_images35,NL_exp_res35);
305     %assessment for 4.0 image dataset
306     metrics_arr40 = assessment(NL_images40,NL_exp_res40);
307     %assessment for 4.5 image dataset
308     metrics_arr45 = assessment(NL_images45,NL_exp_res45);
309     %assessment for 5.0 image dataset
310     metrics_arr50 = assessment(NL_images50,NL_exp_res50);
311
312     %columns titles
313     var_names = {'MSE','PSNR','SSIM','MV','STD','BRISQUE','NIQE'};
314
315     %convert the matrices to tables with column names
316     table30 = array2table(metrics_arr30,'VariableNames',var_names);
317     table35 = array2table(metrics_arr35,'VariableNames',var_names);
318     table40 = array2table(metrics_arr40,'VariableNames',var_names);
319     table45 = array2table(metrics_arr45,'VariableNames',var_names);
320     table50 = array2table(metrics_arr50,'VariableNames',var_names);
321
322     %saving the results to an excel file
323     writetable(table30,'SSR_c120_assessment_test.xlsx','Sheet','3.0');
324     writetable(table35,'SSR_c120_assessment_test.xlsx','Sheet','3.5');
325     writetable(table40,'SSR_c120_assessment_test.xlsx','Sheet','4.0');
326     writetable(table45,'SSR_c120_assessment_test.xlsx','Sheet','4.5');
327     writetable(table50,'SSR_c120_assessment_test.xlsx','Sheet','5.0');
328     disp('Done with c=120!')
329
330     %Images from SSR with c=400
331     NL_exp_res30 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\3.0\SSR\c_400');
332     NL_exp_res35 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\3.5\SSR\c_400');
333     NL_exp_res40 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\4.0\SSR\c_400');
334     NL_exp_res45 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\4.5\SSR\c_400');
335     NL_exp_res50 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\5.0\SSR\c_400');
336
```

```
336
337     %assessment for 3.0 image dataset
338     metrics_arr30 = assessment(NL_images30,NL_exp_res30);
339     %assessment for 3.5 image dataset
340     metrics_arr35 = assessment(NL_images35,NL_exp_res35);
341     %assessment for 4.0 image dataset
342     metrics_arr40 = assessment(NL_images40,NL_exp_res40);
343     %assessment for 4.5 image dataset
344     metrics_arr45 = assessment(NL_images45,NL_exp_res45);
345     %assessment for 5.0 image dataset
346     metrics_arr50 = assessment(NL_images50,NL_exp_res50);
347
348     %columns titles
349     var_names = {'MSE','PSNR','SSIM','MV','STD','BRISQUE','NIQE'};
350
351     %convert the matrices to tables with column names
352     table30 = array2table(metrics_arr30,'VariableNames',var_names);
353     table35 = array2table(metrics_arr35,'VariableNames',var_names);
354     table40 = array2table(metrics_arr40,'VariableNames',var_names);
355     table45 = array2table(metrics_arr45,'VariableNames',var_names);
356     table50 = array2table(metrics_arr50,'VariableNames',var_names);
357
358     %saving the results to an excel file
359     writetable(table30,'SSR_c400_assessment_test.xlsx','Sheet','3.0');
360     writetable(table35,'SSR_c400_assessment_test.xlsx','Sheet','3.5');
361     writetable(table40,'SSR_c400_assessment_test.xlsx','Sheet','4.0');
362     writetable(table45,'SSR_c400_assessment_test.xlsx','Sheet','4.5');
363     writetable(table50,'SSR_c400_assessment_test.xlsx','Sheet','5.0');
364     disp('Done with c=400!')
365     disp('Done with Test Images!')
366     disp('Done!')
```

Figure B.2.20: Assessment script for SSR

We read the experimental results as well as the ground truth images, using Image Datastores, and apply the assessment function that calculates the values of the metrics per image. The result is saved in an excel file with a suitable name. This procedure is applied for all darkness levels, so the final excel will have 5 sheets, one for each darkness level. In addition, this procedure is applied for all values of the constant c and for all sets (training validation and test), so at the end we will have 9 excel files that will each have 5 sheets, one for each darkness level. This data can then be used to evaluate the performance of the algorithm.

Multi Scale Retinex

The function that implements the MSR method is shown in figure B.2.21 below. The function takes as input the LL image that we want to enhance, as well as the 3 values of the constants c that the method needs. Then we check if the values of the constants belong to the desired range, and if they don't then the algorithm stops and displays a relevant error. If the values belong to the range we want, then we define an empty table with the same dimensions as the input image. Then we apply the method per band. Specifically, we apply the Single Scale Retinex function, described above, for each value of the constant, sum the results and divide by 3. Finally, we convert the result to uint8, since we want to return an image and not just an array of values.

```
function NL_image = MultiScaleRetinex(LL_image,c_small,c_medium,c_big)

%========================================================================
%   This function implements the Multi Scale Retinex method for low
%   light image enhancement reasons. Function inputs:
%   1) LL_image: the low light image
%   2) c_small: value of the small scale (<=20)
%   3) c_medium: value of the medium scale (>20, <=200)
%   4) c_big: value of the big scale (>200)
%   Outputs the enhanced normal light image
%   Article Reference:
%   D. J. Jobson, Z. Rahman and G. A. Woodell,
%   "A multiscale retinex for bridging the gap between color images
%   and the human observation of scenes," in IEEE Transactions on Image
%   Processing, vol. 6, no. 7, pp. 965-976, July 1997,
%   doi: 10.1109/83.597272.
%
%   Function Author: Panagiotis Koutsaftis
%
%========================================================================

%checking the values of the surround parameters
if c_small > 20
    error('c_small must be smaller of or equal to 20.')
end
if c_medium<=20 || c_medium>200
    error('c_medium must be in the range [20,200).')
end
if c_big<=200
    error('c_big must be bigger than 200.')
end

%size of the input image
[rows,columns,bands] = size(LL_image);

%initializing the Retinex output with zeros
R_msr = zeros(rows,columns,bands);

%calculating the Retinex output
for band=1:bands
    R_msr(:,:,band) = (double(SingleScaleRetinex(LL_image(:,:,band),c_small)) ...
        +double(SingleScaleRetinex(LL_image(:,:,band),c_medium)) ...
        +double(SingleScaleRetinex(LL_image(:,:,band),c_big))).*(1/3);
end

%final output as uint8
NL_image = uint8(R_msr);
```

Figure B.2.21: Implementation of MSR

We will apply this function to all images in the dataset. This is done with the scripts presented in Figures B.2.22 and B.2.23, below.

```
close all; clear; clc; format compact;


%loading the data
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\3.0\LL-3.0');
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\3.5\LL-3.5');
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\4.0\LL-4.0');
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\4.5\LL-4.5');
LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\5.0\LL-5.0');

%size of the dataset
files = size(LL_images.Files);
numOfImages = files(1);

%applying MSR with surround parameters c_small=10, c_medium=120, c_big=400
for index=1:numOfImages
    LL_img = readimage(LL_images,index);
    image_name = char(LL_images.Files(index));
    %NL_img = readimage(NL_images,index);
    NL_MSR = MultiScaleRetinex(LL_img,10,120,400);
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\3.0\MSR\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\3.5\MSR\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\4.0\MSR\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\4.5\MSR\',image_name((end-8):(end-4)),'.png'];
    file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\5.0\MSR\',image_name((end-8):(end-4)),'.png'];
    imwrite(NL_MSR,file_path);
end
```

Figure B.2.22: script for applying MSR to training dataset per darkness level

We read the images using Image Datastores and then apply the method to each image in each set. For values of the constants we use 10, 120 and 400, which are also the experimental values we used in the case of SSR. The results are saved in an appropriate folder. We apply this process 5 times, once for each darkness level, uncommenting the appropriate line each time. We apply the same procedure for the validation and test sets, as shown in figure B.2.23.

```
close all; clear; clc; format compact;

%========================================================================
%                          VALIDATION IMAGES
%========================================================================
%reading the Low Light Data (VALIDATION)
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\3.0\LL-3.0');
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\3.5\LL-3.5');
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\4.0\LL-4.0');
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\4.5\LL-4.5');
LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\5.0\LL-5.0');

%size of the dataset
files = size(LL_images.Files);
numOfImages = files(1);

disp('Working on validation images...')
%applying MSR with surround parameters c_small=10, c_medium=120, c_big=400
for index=1:numOfImages
    LL_img = readimage(LL_images,index);
    image_name = char(LL_images.Files(index));
    %NL_img = readimage(NL_images,index);
    NL_MSR = MultiScaleRetinex(LL_img,10,120,400);
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\3.0\MSR\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\3.5\MSR\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\4.0\MSR\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\4.5\MSR\',image_name((end-8):(end-4)),'.png'];
    file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\5.0\MSR\',image_name((end-8):(end-4)),'.png'];
    imwrite(NL_MSR,file_path);
end
disp('Done with validation images!')

%========================================================================
%                              TEST IMAGES
%========================================================================
%reading the Low Light Data (TEST)
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\3.0\LL-3.0');
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\3.5\LL-3.5');
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\4.0\LL-4.0');
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\4.5\LL-4.5');
LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\5.0\LL-5.0');

%size of the dataset
files = size(LL_images.Files);
numOfImages = files(1);

disp('Working on Test images...')
%applying MSR with surround parameters c_small=10, c_medium=120, c_big=400
for index=1:numOfImages
    LL_img = readimage(LL_images,index);
    image_name = char(LL_images.Files(index));
    %NL_img = readimage(NL_images,index);
    NL_MSR = MultiScaleRetinex(LL_img,10,120,400);
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\3.0\MSR\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\3.5\MSR\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\4.0\MSR\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\4.5\MSR\',image_name((end-8):(end-4)),'.png'];
    file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\5.0\MSR\',image_name((end-8):(end-4)),'.png'];
    imwrite(NL_MSR,file_path);
end
disp('Done with test images!')
disp('Done!')
```

Figure B.2.23: script for applying MSR to validation and test datasets per darkness level

After applying this procedure, we have the experimental results at our disposal. Based on these we will calculate values of performance metrics, which will be used to evaluate the algorithm. This is done with the script shown in Figure B.2.24.

```
close all; clear; clc; format long;
tic
%=========================================================================
%=========================================================================
%                           TRAINING IMAGES
%=========================================================================
%=========================================================================
disp('Woriking on Training Images...')
%=========================================================================
%                           GROUND TRUTH IMAGES
%=========================================================================

NL_images30 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\3.0\NL-3.0');
NL_images35 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\3.5\NL-3.5');
NL_images40 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\4.0\NL-4.0');
NL_images45 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\4.5\NL-4.5');
NL_images50 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\5.0\NL-5.0');

%=========================================================================
%               IMAGES FROM THE MSR METHOD
%=========================================================================

NL_exp_res30 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\3.0\MSR');
NL_exp_res35 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\3.5\MSR');
NL_exp_res40 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\4.0\MSR');
NL_exp_res45 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\4.5\MSR');
NL_exp_res50 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\5.0\MSR');

%assessment for 3.0 image dataset
metrics_arr30 = assessment(NL_images30,NL_exp_res30);
%assessment for 3.5 image dataset
metrics_arr35 = assessment(NL_images35,NL_exp_res35);
%assessment for 4.0 image dataset
metrics_arr40 = assessment(NL_images40,NL_exp_res40);
%assessment for 4.5 image dataset
metrics_arr45 = assessment(NL_images45,NL_exp_res45);
%assessment for 5.0 image dataset
metrics_arr50 = assessment(NL_images50,NL_exp_res50);

%columns titles
var_names = {'MSE','PSNR','SSIM','MV','STD','BRISQUE','NIQE'};

%convert the matrices to tables with column names
table30 = array2table(metrics_arr30,'VariableNames',var_names);
table35 = array2table(metrics_arr35,'VariableNames',var_names);
table40 = array2table(metrics_arr40,'VariableNames',var_names);
table45 = array2table(metrics_arr45,'VariableNames',var_names);
table50 = array2table(metrics_arr50,'VariableNames',var_names);

%saving the results to an excel file
writetable(table30,'MSR_assessment_training.xlsx','Sheet','3.0');
writetable(table35,'MSR_assessment_training.xlsx','Sheet','3.5');
writetable(table40,'MSR_assessment_training.xlsx','Sheet','4.0');
writetable(table45,'MSR_assessment_training.xlsx','Sheet','4.5');
writetable(table50,'MSR_assessment_training.xlsx','Sheet','5.0');

disp('Done with Training Images!')

%=========================================================================
%=========================================================================
%                           VALIDATION IMAGES
%=========================================================================
%=========================================================================
disp('Working on Validation Images...')
%=========================================================================
%                           GROUND TRUTH IMAGES
%=========================================================================

NL_images30 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\3.0\NL-3.0');
NL_images35 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\3.5\NL-3.5');
NL_images40 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\4.0\NL-4.0');
NL_images45 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\4.5\NL-4.5');
NL_images50 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\5.0\NL-5.0');

%=========================================================================
%               IMAGES FROM THE MSR METHOD
%=========================================================================

NL_exp_res30 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\3.0\MSR');
NL_exp_res35 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\3.5\MSR');
NL_exp_res40 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\4.0\MSR');
NL_exp_res45 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\4.5\MSR');
NL_exp_res50 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\5.0\MSR');
```

```
%assessment for 3.0 image dataset
metrics_arr30 = assessment(NL_images30,NL_exp_res30);
%assessment for 3.5 image dataset
metrics_arr35 = assessment(NL_images35,NL_exp_res35);
%assessment for 4.0 image dataset
metrics_arr40 = assessment(NL_images40,NL_exp_res40);
%assessment for 4.5 image dataset
metrics_arr45 = assessment(NL_images45,NL_exp_res45);
%assessment for 5.0 image dataset
metrics_arr50 = assessment(NL_images50,NL_exp_res50);

%columns titles
var_names = {'MSE','PSNR','SSIM','MV','STD','BRISQUE','NIQE'};

%convert the matrices to tables with column names
table30 = array2table(metrics_arr30,'VariableNames',var_names);
table35 = array2table(metrics_arr35,'VariableNames',var_names);
table40 = array2table(metrics_arr40,'VariableNames',var_names);
table45 = array2table(metrics_arr45,'VariableNames',var_names);
table50 = array2table(metrics_arr50,'VariableNames',var_names);

%saving the results to an excel file
writetable(table30,'MSR_assessment_validation.xlsx','Sheet','3.0');
writetable(table35,'MSR_assessment_validation.xlsx','Sheet','3.5');
writetable(table40,'MSR_assessment_validation.xlsx','Sheet','4.0');
writetable(table45,'MSR_assessment_validation.xlsx','Sheet','4.5');
writetable(table50,'MSR_assessment_validation.xlsx','Sheet','5.0');

disp('Done with Validation Images!')

%==========================================================================
%==========================================================================
%                              TEST IMAGES
%==========================================================================
%==========================================================================
disp('Working on Test Images...')
%==========================================================================
%                              GROUND TRUTH IMAGES
%==========================================================================

NL_images30 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\3.0\NL-3.0');
NL_images35 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\3.5\NL-3.5');
NL_images40 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\4.0\NL-4.0');
NL_images45 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\4.5\NL-4.5');
NL_images50 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\5.0\NL-5.0');

%==========================================================================
%              IMAGES FROM THE MSR METHOD
%==========================================================================

NL_exp_res30 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\3.0\MSR');
NL_exp_res35 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\3.5\MSR');
NL_exp_res40 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\4.0\MSR');
NL_exp_res45 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\4.5\MSR');
NL_exp_res50 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\5.0\MSR');

%assessment for 3.0 image dataset
metrics_arr30 = assessment(NL_images30,NL_exp_res30);
%assessment for 3.5 image dataset
metrics_arr35 = assessment(NL_images35,NL_exp_res35);
%assessment for 4.0 image dataset
metrics_arr40 = assessment(NL_images40,NL_exp_res40);
%assessment for 4.5 image dataset
metrics_arr45 = assessment(NL_images45,NL_exp_res45);
%assessment for 5.0 image dataset
metrics_arr50 = assessment(NL_images50,NL_exp_res50);

%columns titles
var_names = {'MSE','PSNR','SSIM','MV','STD','BRISQUE','NIQE'};

%convert the matrices to tables with column names
table30 = array2table(metrics_arr30,'VariableNames',var_names);
table35 = array2table(metrics_arr35,'VariableNames',var_names);
table40 = array2table(metrics_arr40,'VariableNames',var_names);
table45 = array2table(metrics_arr45,'VariableNames',var_names);
table50 = array2table(metrics_arr50,'VariableNames',var_names);

%saving the results to an excel file
writetable(table30,'MSR_assessment_test.xlsx','Sheet','3.0');
writetable(table35,'MSR_assessment_test.xlsx','Sheet','3.5');
writetable(table40,'MSR_assessment_test.xlsx','Sheet','4.0');
writetable(table45,'MSR_assessment_test.xlsx','Sheet','4.5');
writetable(table50,'MSR_assessment_test.xlsx','Sheet','5.0');

disp('Done with Test Images!')
disp('Done!')
```

Figure B.2.24: Assessment script for MSR

We read the experimental results and ground truth images using Image Datastores, and apply the assessment function to calculate the metrics. The results are saved in excel files with a suitable name. We apply this process per darkness level and per set (training, validation, test) so at the end we have 3 excel files (one for each set), each of which has 5 sheets (one for each darkness level) with the values of the metrics per image.

Dehaze

The function that implements the LLIE with Dehaze method is shown in figure B.2.25 below.

```
function NL_image = MatlabDehaze(LL_image)

%=========================================================================
% Function that implements the Dehaze LLIE algorithm. The functions takes
% as input the LL image we want to enhance and outputs the enhanced image.
% Author: Panagiotis Koutsaftis aivc21010
%=========================================================================

%size of the image
[rows,columns,bands] = size(LL_image);

%converting the image to double for the calculations
LL = double(LL_image);

%inverting the image
temp = 255*ones(rows,columns,bands);
inv = uint8(temp - LL);

%applying matlab's dehazing function
final_inv = imreducehaze(inv,Method='approxdcp');

%inverting again
final_inv = temp -double(final_inv);

%final result
NL_image = uint8(final_inv);
```

Figure B.2.25: Implementation of Dehaze LLIE algorithm

The function takes as input the LL image we want to enhance, calculates the dimensions and converts it into a double data type needed for the calculations. It then inverts it and applies MATLAB's imreducehaze function. This function applies a dehazing algorithm. We used a ready-made MATLAB function and did not implement it from scratch for

optimization reasons. Finally, it reverses the result of dehaze again, converts it to uint8 and this is the final result returned.

We will apply this function to all images in the dataset. This is done with the scripts presented in images B.2.26 and B.2.27.

```
close all; clear; clc; format compact;


%loading the data
LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\3.0\LL-3.0');
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\3.5\LL-3.5');
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\4.0\LL-4.0');
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\4.5\LL-4.5');
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\5.0\LL-5.0');

%size of the dataset
files = size(LL_images.Files);
numOfImages = files(1);

tic
%applying matlab's dehaze function
for index=1:numOfImages
    LL_img = readimage(LL_images,index);
    image_name = char(LL_images.Files(index));
    NL_MD = MatlabDehaze(LL_img);
    file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\3.0\Matlab Dehaze\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\3.5\Matlab Dehaze\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\4.0\Matlab Dehaze\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\4.5\Matlab Dehaze\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\5.0\Matlab Dehaze\',image_name((end-8):(end-4)),'.png'];
    imwrite(NL_MD,file_path);
end
toc
disp('Done!')
```

Figure B.2.26: script for applying dehaze to training dataset per darkness level

We read the LL images using Image Datastores and apply the function to each image in the set. The result is saved in a folder with an appropriate name. We apply this procedure to each darkness level, i.e. we run the script 5 times, uncommenting the appropriate line each time. We apply the exact same procedure for the validation and test sets, as shown in image B.2.27.

```
close all; clear; clc; format compact;

%==========================================================================
%                           VALIDATION IMAGES
%==========================================================================
%reading the Low Light Data (VALIDATION)
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\3.0\LL-3.0');
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\3.5\LL-3.5');
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\4.0\LL-4.0');
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\4.5\LL-4.5');
LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\5.0\LL-5.0');

%size of the dataset
files = size(LL_images.Files);
numOfImages = files(1);

tic
%applying matlab's dehaze function
for index=1:numOfImages
    LL_img = readimage(LL_images,index);
    image_name = char(LL_images.Files(index));
    NL_MD = MatlabDehaze(LL_img);
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\3.0\Matlab Dehaze\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\3.5\Matlab Dehaze\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\4.0\Matlab Dehaze\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\4.5\Matlab Dehaze\',image_name((end-8):(end-4)),'.png'];
    file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\5.0\Matlab Dehaze\',image_name((end-8):(end-4)),'.png'];
    imwrite(NL_MD,file_path);
end
toc
disp('Done with Matlab Dehaze function!')
disp('Done with validation images!')

%==========================================================================
%                               TEST IMAGES
%==========================================================================
%reading the Low Light Data (TEST)
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\3.0\LL-3.0');
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\3.5\LL-3.5');
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\4.0\LL-4.0');
%LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\4.5\LL-4.5');
LL_images = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\5.0\LL-5.0');

%size of the dataset
files = size(LL_images.Files);
numOfImages = files(1);

tic
%applying matlab's dehaze function
for index=1:numOfImages
    LL_img = readimage(LL_images,index);
    image_name = char(LL_images.Files(index));
    NL_MD = MatlabDehaze(LL_img);
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\3.0\Matlab Dehaze\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\3.5\Matlab Dehaze\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\4.0\Matlab Dehaze\',image_name((end-8):(end-4)),'.png'];
    %file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\4.5\Matlab Dehaze\',image_name((end-8):(end-4)),'.png'];
    file_path = ['D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\5.0\Matlab Dehaze\',image_name((end-8):(end-4)),'.png'];
    imwrite(NL_MD,file_path);
end
toc
disp('Done with Matlab Dehaze function!')
disp('Done with test images!')
disp('Done!')
```

Figure B.2.27: script for applying dehaze to validation and test datasets per darkness level

Upon completion of this process, we have at our disposal the experimental results, which we will use to calculate performance metrics. This is done with the script shown in Figure B.2.28 below.

```
close all; clear; clc; format long;
tic
%=========================================================================
%=========================================================================
%                           TRAINING IMAGES
%=========================================================================
%=========================================================================
disp('Woriking on Training Images...')
%=========================================================================
%                           GROUND TRUTH IMAGES
%=========================================================================

NL_images30 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\3.0\NL-3.0');
NL_images35 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\3.5\NL-3.5');
NL_images40 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\4.0\NL-4.0');
NL_images45 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\4.5\NL-4.5');
NL_images50 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TRAINING IMAGES\5.0\NL-5.0');

%=========================================================================
%               IMAGES FROM THE MATLAB DEHAZE METHOD
%=========================================================================

NL_exp_res30 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\3.0\Matlab Dehaze');
NL_exp_res35 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\3.5\Matlab Dehaze');
NL_exp_res40 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\4.0\Matlab Dehaze');
NL_exp_res45 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\4.5\Matlab Dehaze');
NL_exp_res50 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods\5.0\Matlab Dehaze');

%assessment for 3.0 image dataset
metrics_arr30 = assessment(NL_images30,NL_exp_res30);
%assessment for 3.5 image dataset
metrics_arr35 = assessment(NL_images35,NL_exp_res35);
%assessment for 4.0 image dataset
metrics_arr40 = assessment(NL_images40,NL_exp_res40);
%assessment for 4.5 image dataset
metrics_arr45 = assessment(NL_images45,NL_exp_res45);
%assessment for 5.0 image dataset
metrics_arr50 = assessment(NL_images50,NL_exp_res50);

%columns titles
var_names = {'MSE','PSNR','SSIM','MV','STD','BRISQUE','NIQE'};

%convert the matrices to tables with column names
table30 = array2table(metrics_arr30,'VariableNames',var_names);
table35 = array2table(metrics_arr35,'VariableNames',var_names);
table40 = array2table(metrics_arr40,'VariableNames',var_names);
table45 = array2table(metrics_arr45,'VariableNames',var_names);
table50 = array2table(metrics_arr50,'VariableNames',var_names);

%saving the results to an excel file
writetable(table30,'Matlab_Dehaze_assessment_training.xlsx','Sheet','3.0');
writetable(table35,'Matlab_Dehaze_assessment_training.xlsx','Sheet','3.5');
writetable(table40,'Matlab_Dehaze_assessment_training.xlsx','Sheet','4.0');
writetable(table45,'Matlab_Dehaze_assessment_training.xlsx','Sheet','4.5');
writetable(table50,'Matlab_Dehaze_assessment_training.xlsx','Sheet','5.0');

disp('Done with Training Images!')

%=========================================================================
%=========================================================================
%                           VALIDATION IMAGES
%=========================================================================
%=========================================================================
disp('Working on Validation Images...')
%=========================================================================
%                           GROUND TRUTH IMAGES
%=========================================================================

NL_images30 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\3.0\NL-3.0');
NL_images35 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\3.5\NL-3.5');
NL_images40 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\4.0\NL-4.0');
NL_images45 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\4.5\NL-4.5');
NL_images50 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\VALIDATION IMAGES\5.0\NL-5.0');

%=========================================================================
%               IMAGES FROM THE MATLAB DEHAZE METHOD
%=========================================================================

NL_exp_res30 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\3.0\Matlab Dehaze');
NL_exp_res35 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\3.5\Matlab Dehaze');
NL_exp_res40 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\4.0\Matlab Dehaze');
NL_exp_res45 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\4.5\Matlab Dehaze');
NL_exp_res50 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_ValidationImages\5.0\Matlab Dehaze');
```

```
%assessment for 3.0 image dataset
metrics_arr30 = assessment(NL_images30,NL_exp_res30);
%assessment for 3.5 image dataset
metrics_arr35 = assessment(NL_images35,NL_exp_res35);
%assessment for 4.0 image dataset
metrics_arr40 = assessment(NL_images40,NL_exp_res40);
%assessment for 4.5 image dataset
metrics_arr45 = assessment(NL_images45,NL_exp_res45);
%assessment for 5.0 image dataset
metrics_arr50 = assessment(NL_images50,NL_exp_res50);

%columns titles
var_names = {'MSE','PSNR','SSIM','MV','STD','BRISQUE','NIQE'};

%convert the matrices to tables with column names
table30 = array2table(metrics_arr30,'VariableNames',var_names);
table35 = array2table(metrics_arr35,'VariableNames',var_names);
table40 = array2table(metrics_arr40,'VariableNames',var_names);
table45 = array2table(metrics_arr45,'VariableNames',var_names);
table50 = array2table(metrics_arr50,'VariableNames',var_names);

%saving the results to an excel file
writetable(table30,'Matlab_Dehaze_assessment_validation.xlsx','Sheet','3.0');
writetable(table35,'Matlab_Dehaze_assessment_validation.xlsx','Sheet','3.5');
writetable(table40,'Matlab_Dehaze_assessment_validation.xlsx','Sheet','4.0');
writetable(table45,'Matlab_Dehaze_assessment_validation.xlsx','Sheet','4.5');
writetable(table50,'Matlab_Dehaze_assessment_validation.xlsx','Sheet','5.0');

disp('Done with Validation Images!')

%=========================================================================
%=========================================================================
%                              TEST IMAGES
%=========================================================================
%=========================================================================
disp('Working on Test Images...')
%=========================================================================
%                          GROUND TRUTH IMAGES
%=========================================================================

NL_images30 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\3.0\NL-3.0');
NL_images35 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\3.5\NL-3.5');
NL_images40 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\4.0\NL-4.0');
NL_images45 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\4.5\NL-4.5');
NL_images50 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\TEST IMAGES\5.0\NL-5.0');

%=========================================================================
%              IMAGES FROM THE MATLAB DEHAZE METHOD
%=========================================================================

NL_exp_res30 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\3.0\Matlab Dehaze');
NL_exp_res35 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\3.5\Matlab Dehaze');
NL_exp_res40 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\4.0\Matlab Dehaze');
NL_exp_res45 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\4.5\Matlab Dehaze');
NL_exp_res50 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Classical Methods_TestImages\5.0\Matlab Dehaze');

%assessment for 3.0 image dataset
metrics_arr30 = assessment(NL_images30,NL_exp_res30);
%assessment for 3.5 image dataset
metrics_arr35 = assessment(NL_images35,NL_exp_res35);
%assessment for 4.0 image dataset
metrics_arr40 = assessment(NL_images40,NL_exp_res40);
%assessment for 4.5 image dataset
metrics_arr45 = assessment(NL_images45,NL_exp_res45);
%assessment for 5.0 image dataset
metrics_arr50 = assessment(NL_images50,NL_exp_res50);

%columns titles
var_names = {'MSE','PSNR','SSIM','MV','STD','BRISQUE','NIQE'};

%convert the matrices to tables with column names
table30 = array2table(metrics_arr30,'VariableNames',var_names);
table35 = array2table(metrics_arr35,'VariableNames',var_names);
table40 = array2table(metrics_arr40,'VariableNames',var_names);
table45 = array2table(metrics_arr45,'VariableNames',var_names);
table50 = array2table(metrics_arr50,'VariableNames',var_names);

%saving the results to an excel file
writetable(table30,'Matlab_Dehaze_assessment_test.xlsx','Sheet','3.0');
writetable(table35,'Matlab_Dehaze_assessment_test.xlsx','Sheet','3.5');
writetable(table40,'Matlab_Dehaze_assessment_test.xlsx','Sheet','4.0');
writetable(table45,'Matlab_Dehaze_assessment_test.xlsx','Sheet','4.5');
writetable(table50,'Matlab_Dehaze_assessment_test.xlsx','Sheet','5.0');
```

Figure B.2.28: Assessment script for Dehaze

We read the experimental as well as ground truth images using Image Datastores, and apply the assessment function to each image in the set. The results are saved in an excel file with an appropriate name. We apply this procedure for all darkness levels and for all sets (training, validation, test). So in the end we will have 3 excel files (one for each set) that will have 5 sheets (one for each darkness level).

B.3: Chapter 3 Code

LLCNN – Original Architecture

The architecture we study consists of a series of convolutional modules. The implementation of the convolutional module is shown in figure B.3.1 below.

```
#defining the Convolutional Module of LLCNN article
def conv_module(x,filter = 64):
  '''Implementation of the convolutional module of LLCNN article.
     Input: The output x of the previous layer
            The number of filters. If not definned then 64 filters are used'''
  #left_part
  x_left = tfl.Conv2D(filters=filter,kernel_size=(3,3),strides=1,padding='same')(x)
  x_left = tfl.ReLU()(x_left)
  x_left = tfl.Conv2D(filters=filter,kernel_size=(3,3),strides=1,padding='same')(x_left)

  #right_part
  x_right = tfl.Conv2D(filters=filter,kernel_size=(1,1),strides=1,padding='same')(x)

  #adding the parts
  x_add = tfl.Add()([x_left,x_right])
  #bottom part
  x_bottom = tfl.ReLU()(x_add)
  x_bottom = tfl.Conv2D(filters=filter,kernel_size=(3,3),strides=1,padding='same')(x_bottom)
  x_bottom = tfl.ReLU()(x_bottom)
  x_bottom = tfl.Conv2D(filters=filter,kernel_size=(3,3),strides=1,padding='same')(x_bottom)

  #adding the x_add and x_bottom
  x_final = tfl.Add()([x_add,x_bottom])

  #final output
  output = tfl.ReLU()(x_final)

  return output
```

Figure B.3.1: Implementation of the convolutional module

We see that the function takes as input the output of the previous layer, as well as the number of filters, which if not defined then is by default 64.

First, we define the inception based part, with the left part consisting of the two convolutional layers with 64 filters and 3x3 each, with the output passing through a ReLU function, and the right part consisting of a 1x1 convolutional layer with 64 filters. We then add these two outputs, with the result passing through the Residual Learning based part. This part consists of two 3x3 convolutional layers with 64 filters and ReLU activation function, as well as a shortcut connection that is added to the result of these two layers. As output, the function gives the output of the module, which will be passed to the next layer.

Based on this we can define the overall architecture of the model, which is shown in figure B.3.2 below.

```
#defining the LLCNN final model
def LLCNN_model(input_tensor,num_of_modules=5):
  '''Implementing the LLCNN architecture. Inputs are the input tensor
     (the normalized image dataset) and the number of Convolutional modules
     the architecture is going to use. If the number of modules is not definned
     we are using 5 convolutional modules.'''
  #first layer of the architecture
  x = tfl.Conv2D(filters=64,kernel_size=(1,1),strides=1,padding='same')(input_tensor)
  x = tfl.ReLU()(x)

  #series of convolutional modules
  for l in range(num_of_modules):
    x = conv_module(x)

  #output of the model
  outputs = tfl.Conv2D(filters=3,kernel_size=(1,1),strides=1,padding='same')(x)

  return outputs
```

Figure B.3.2: defining the LLCNN architecture

As input, the function takes the input tensor, which we define next, and its size depends on the data set we use, as well as the number of convolutional modules we want the architecture to have, with the default being 5. First, we define the pre-processing 1x1 convolutional layer, which will consist of 64 filters and will have a ReLU activation function. The output of this layer then goes through the series of convolutional modules we defined above. Finally, the result goes through a 1x1 convolutional layer, which consists of only 3 filters, since we are studying RGB images.

Having defined the architecture, we can initialize and train the model. The initialization is shown in figure B.3.3 below.

```
[ ]  #building the model
     #input of the model
     model_input = Input((625,625,3))
     #layers/output of the model
     model_output = LLCNN_model(model_input,5)
     #grouping the layers to form the final model architecture
     model = Model(model_input,model_output)

[ ]  #defining the loss function of the model
     loss = keras.losses.mean_squared_error
     #defining the oprimizer of the model
     opt = keras.optimizers.SGD(learning_rate=0.01, momentum=0.9, nesterov=False)

[ ]  #compiling the model and printing the summary
     model.compile(optimizer=opt,loss=loss,metrics=['mean_squared_error'])
     model.summary()
```

Figure B.3.3: initialization of the DL model

First, we define the input of the model, which will have the same size as that of the respective image. Then we initialize the LLCNN architecture, where at this point we choose the number of convolutional modules we want it to have (1, 3 or 5). Next, we group the input layer and the LLCNN architecture, and then define the error function as well as the minimization algorithm to use. For simplicity we use MSE as the error function, and SGD as the optimizer. Finally, we compile the model, and we can proceed with its training.

To train the model we need to read and normalize the data. This is done as shown in figure B.3.4 below.

```
#training data
print('Loading the training data...')
#input_path = r'/content/drive/MyDrive/DiplomaThesisImageDataset/DataPerLight/dataset/training/3.0/Train_LL_3_0.npy'
input_path = r'/content/drive/MyDrive/DiplomaThesisImageDataset/DataPerLight/dataset/training/3.5/Train_LL_3_5.npy'
#input_path = r'/content/drive/MyDrive/DiplomaThesisImageDataset/DataPerLight/dataset/training/4.0/Train_LL_4_0.npy'
#input_path = r'/content/drive/MyDrive/DiplomaThesisImageDataset/DataPerLight/dataset/training/4.5/Train_LL_4_5.npy'
#input_path = r'/content/drive/MyDrive/DiplomaThesisImageDataset/DataPerLight/dataset/training/5.0/Train_LL_5_0.npy'

train_input = np.load(input_path)

#output_path = r'/content/drive/MyDrive/DiplomaThesisImageDataset/DataPerLight/dataset/training/3.0/Train_NL_3_0.npy'
output_path = r'/content/drive/MyDrive/DiplomaThesisImageDataset/DataPerLight/dataset/training/3.5/Train_NL_3_5.npy'
#output_path = r'/content/drive/MyDrive/DiplomaThesisImageDataset/DataPerLight/dataset/training/4.0/Train_NL_4_0.npy'
#output_path = r'/content/drive/MyDrive/DiplomaThesisImageDataset/DataPerLight/dataset/training/4.5/Train_NL_4_5.npy'
#output_path = r'/content/drive/MyDrive/DiplomaThesisImageDataset/DataPerLight/dataset/training/5.0/Train_NL_5_0.npy'
train_output = np.load(output_path)
print('\nData loaded!')
```

```
Loading the training data...

Data loaded!
```

```
#shape of input and output
print('================================================================================')
print('\t\t\tTRAINING DATA')
print('================================================================================\n')
print(f'Input shape: {train_input.shape} ')
print(f'Output shape: {train_output.shape}')

print('Reshaping and normalizing the data...')

data = []
for index in range(train_input.shape[0]):
    data.append(train_input[index,0,:,:,:].reshape(train_input.shape[2],train_input.shape[3],train_input.shape[4]))

train_input = np.array(data)
train_input = train_input/255


data = []
for index in range(train_output.shape[0]):
    data.append(train_output[index,0,:,:,:].reshape(train_output.shape[2],train_output.shape[3],train_output.shape[4]))

train_output = np.array(data)
train_output = train_output/255


print(f'Input shape: {train_input.shape} ')
print(f'Output shape: {train_output.shape}')
```

```
#validation data
print('Loading the validation data...')
#input_path = r'/content/drive/MyDrive/DiplomaThesisImageDataset/DataPerLight/dataset/validation/3.0/Validation_LL_3_0.npy'
input_path = r'/content/drive/MyDrive/DiplomaThesisImageDataset/DataPerLight/dataset/validation/3.5/Validation_LL_3_5.npy'
#input_path = r'/content/drive/MyDrive/DiplomaThesisImageDataset/DataPerLight/dataset/validation/4.0/Validation_LL_4_0.npy'
#input_path = r'/content/drive/MyDrive/DiplomaThesisImageDataset/DataPerLight/dataset/validation/4.5/Validation_LL_4_5.npy'
#input_path = r'/content/drive/MyDrive/DiplomaThesisImageDataset/DataPerLight/dataset/validation/5.0/Validation_LL_5_0.npy'

val_input = np.load(input_path)

#output_path = r'/content/drive/MyDrive/DiplomaThesisImageDataset/DataPerLight/dataset/validation/3.0/Validation_NL_3_0.npy'
output_path = r'/content/drive/MyDrive/DiplomaThesisImageDataset/DataPerLight/dataset/validation/3.5/Validation_NL_3_5.npy'
#output_path = r'/content/drive/MyDrive/DiplomaThesisImageDataset/DataPerLight/dataset/validation/4.0/Validation_NL_4_0.npy'
#output_path = r'/content/drive/MyDrive/DiplomaThesisImageDataset/DataPerLight/dataset/validation/4.5/Validation_NL_4_5.npy'
#output_path = r'/content/drive/MyDrive/DiplomaThesisImageDataset/DataPerLight/dataset/validation/5.0/Validation_NL_5_0.npy'

val_output = np.load(output_path)
print('\nData loaded!')
```

```
Loading the validation data...

Data loaded!
```

```
#shape of validation input and output
print('================================================================================')
print('\t\t\tVALIDATION DATA')
print('================================================================================\n')
print(f'Input shape: {val_input.shape} ')
print(f'Output shape: {val_output.shape}')

print('Reshaping and normalizing the data...')

data = []
for index in range(val_input.shape[0]):
    data.append(val_input[index,0,:,:,:].reshape(val_input.shape[2],val_input.shape[3],val_input.shape[4]))

val_input = np.array(data)
val_input = val_input/255


data = []
for index in range(val_output.shape[0]):
    data.append(val_output[index,0,:,:,:].reshape(val_output.shape[2],val_output.shape[3],val_output.shape[4]))

val_output = np.array(data)
val_output = val_output/255


print(f'Input shape: {val_input.shape} ')
print(f'Output shape: {val_output.shape}')

del data
```

Figure B.3.4: reading and normalizing the training and validation data

First, we read the training set we are interested in (since we have 5 different darkness levels) and normalize them by dividing by 255. Then we apply the same process to the validation data. Upon completion of this process we have at our disposal what we need to train the model, which is done as shown in figure B.3.5.

```
#training the model
print('Training the model...')
model.fit(train_input, train_output, epochs=10, batch_size=10, verbose=1,validation_data=(val_input,val_output))
print('\n\nTraining completed!')
```

Figure B.3.5: training of the LLCNN model

Due to a lack of computing resources, the training is only done for 10 epochs with the batch size also being 10. With the completion of this process, we can apply the trained model to the test set to evaluate its generalization ability. First we read and normalize the test set, in the same way we applied for the training and validation sets (figure B.3.6).

```
#test data
print('Loading the test data...')
#input_path = r'/content/drive/MyDrive/DiplomaThesisImageDataset/DataPerLight/dataset/test/3.0/Test_LL_3_0.npy'
input_path = r'/content/drive/MyDrive/DiplomaThesisImageDataset/DataPerLight/dataset/test/3.5/Test_LL_3_5.npy'
#input_path = r'/content/drive/MyDrive/DiplomaThesisImageDataset/DataPerLight/dataset/test/4.0/Test_LL_4_0.npy'
#input_path = r'/content/drive/MyDrive/DiplomaThesisImageDataset/DataPerLight/dataset/test/4.5/Test_LL_4_5.npy'
#input_path = r'/content/drive/MyDrive/DiplomaThesisImageDataset/DataPerLight/dataset/test/5.0/Test_LL_5_0.npy'

test_input = np.load(input_path)

#output_path = r'/content/drive/MyDrive/DiplomaThesisImageDataset/DataPerLight/dataset/test/3.0/Test_NL_3_0.npy'
output_path = r'/content/drive/MyDrive/DiplomaThesisImageDataset/DataPerLight/dataset/test/3.5/Test_NL_3_5.npy'
#output_path = r'/content/drive/MyDrive/DiplomaThesisImageDataset/DataPerLight/dataset/test/4.0/Test_NL_4_0.npy'
#output_path = r'/content/drive/MyDrive/DiplomaThesisImageDataset/DataPerLight/dataset/test/4.5/Test_NL_4_5.npy'
#output_path = r'/content/drive/MyDrive/DiplomaThesisImageDataset/DataPerLight/dataset/test/5.0/Test_NL_5_0.npy'

test_output = np.load(output_path)
print('\nTraining Data Loaded!')
```

```
Loading the test data...

Training Data Loaded!
```

```
#shape of validation input and output
print('=================================================================')
print('\t\t\tTEST DATA')
print('=================================================================\n')
print(f'Input shape: {test_input.shape} ')
print(f'Output shape: {test_output.shape}')

print('Reshaping and normalizing the data...')

data = []
for index in range(test_input.shape[0]):
    data.append(test_input[index,0,:,:,:].reshape(test_input.shape[2],test_input.shape[3],test_input.shape[4]))

test_input = np.array(data)
#normalization
test_input = test_input/255


data = []
for index in range(test_output.shape[0]):
    data.append(test_output[index,0,:,:,:].reshape(test_output.shape[2],test_output.shape[3],test_output.shape[4]))

test_output = np.array(data)
#normalization
test_output = test_output/255


print(f'Input shape: {test_input.shape} ')
print(f'Output shape: {test_output.shape}')

#deleting the temporary list for space economy reasons
del data
```

Figure B.3.6: reading and normalizing the test set

We then apply the model to the test set, and save the results in a suitable folder so that we can later use them to evaluate the method.

```
#making prediction with the training set
test_pred = model.predict(test_input)
```

```
3/3 [==============================] - 2s 1s/step
```

```
#==================
#saving the test prediction images
#==================
print('Saving the result...')
#paths
#================================================================================
#                          1 convolutional module
#================================================================================
#path = r'/content/drive/MyDrive/DiplomaThesisImageDataset/DataPerLight/LLCNN_Experiment_Results/3.0/1 module'
#path = r'/content/drive/MyDrive/DiplomaThesisImageDataset/DataPerLight/LLCNN_Experiment_Results/3.5/1 module'
#path = r'/content/drive/MyDrive/DiplomaThesisImageDataset/DataPerLight/LLCNN_Experiment_Results/4.0/1 module'
#path = r'/content/drive/MyDrive/DiplomaThesisImageDataset/DataPerLight/LLCNN_Experiment_Results/4.5/1 module'
#path = r'/content/drive/MyDrive/DiplomaThesisImageDataset/DataPerLight/LLCNN_Experiment_Results/5.0/1 module'

#================================================================================
#                          3 convolutional modules
#================================================================================
#path = r'/content/drive/MyDrive/DiplomaThesisImageDataset/DataPerLight/LLCNN_Experiment_Results/3.0/3 modules'
#path = r'/content/drive/MyDrive/DiplomaThesisImageDataset/DataPerLight/LLCNN_Experiment_Results/3.5/3 modules'
#path = r'/content/drive/MyDrive/DiplomaThesisImageDataset/DataPerLight/LLCNN_Experiment_Results/4.0/3 modules'
#path = r'/content/drive/MyDrive/DiplomaThesisImageDataset/DataPerLight/LLCNN_Experiment_Results/4.5/3 modules'
#path = r'/content/drive/MyDrive/DiplomaThesisImageDataset/DataPerLight/LLCNN_Experiment_Results/5.0/3 modules'

#================================================================================
#                          5 convolutional modules
#================================================================================
#path = r'/content/drive/MyDrive/DiplomaThesisImageDataset/DataPerLight/LLCNN_Experiment_Results/3.0/5 modules'
#path = r'/content/drive/MyDrive/DiplomaThesisImageDataset/DataPerLight/LLCNN_Experiment_Results/3.5/5 modules'
#path = r'/content/drive/MyDrive/DiplomaThesisImageDataset/DataPerLight/LLCNN_Experiment_Results/4.0/5 modules'
#path = r'/content/drive/MyDrive/DiplomaThesisImageDataset/DataPerLight/LLCNN_Experiment_Results/4.5/5 modules'
#path = r'/content/drive/MyDrive/DiplomaThesisImageDataset/DataPerLight/LLCNN_Experiment_Results/5.0/5 modules'


#saving at the particular path
#cv2.imwrite(path+'/image1.png',test_input[0,:,:,:]*255)

for index in range(test_pred.shape[0]):
  cv2.imwrite(path+'/'+str(index+1)+'.png',test_pred[index,:,:,:]*255)

#deleting test predictions for space reasons
del test_pred
```

Figure B.3.7: applying the model to the test set and saving the results

Upon completion of this process, we have the experimental results available, which we can use to calculate performance metrics. This is done with the script shown in Figure B.3.8 below.

```
%=========================================================================
%                           TEST IMAGES
%=========================================================================
disp('Working on Test Images...')

%Ground Truth images
NL_images30 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\GROUND TRUTH DEEP LEARNING\Test\3.0');
NL_images35 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\GROUND TRUTH DEEP LEARNING\Test\3.5');
NL_images40 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\GROUND TRUTH DEEP LEARNING\Test\4.0');
NL_images45 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\GROUND TRUTH DEEP LEARNING\Test\4.5');
NL_images50 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\GROUND TRUTH DEEP LEARNING\Test\5.0');


%=========================================================================
%                           LLCNN-1 MODULE
%=========================================================================
NL_exp_res30 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Deep Learning Results\LLCNN\LLCNN_Experiment_Results\3.0\1 module');
NL_exp_res35 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Deep Learning Results\LLCNN\LLCNN_Experiment_Results\3.5\1 module');
NL_exp_res40 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Deep Learning Results\LLCNN\LLCNN_Experiment_Results\4.0\1 module');
NL_exp_res45 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Deep Learning Results\LLCNN\LLCNN_Experiment_Results\4.5\1 module');
NL_exp_res50 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Deep Learning Results\LLCNN\LLCNN_Experiment_Results\5.0\1 module');

%assessment for 3.0 image dataset
metrics_arr30 = assessment(NL_images30,NL_exp_res30);
%assessment for 3.5 image dataset
metrics_arr35 = assessment(NL_images35,NL_exp_res35);
%assessment for 4.0 image dataset
metrics_arr40 = assessment(NL_images40,NL_exp_res40);
%assessment for 4.5 image dataset
metrics_arr45 = assessment(NL_images45,NL_exp_res45);
%assessment for 5.0 image dataset
metrics_arr50 = assessment(NL_images50,NL_exp_res50);

%columns titles
var_names = {'MSE','PSNR','SSIM','MV','STD','BRISQUE','NIQE'};

%convert the matrices to tables with column names
table30 = array2table(metrics_arr30,'VariableNames',var_names);
table35 = array2table(metrics_arr35,'VariableNames',var_names);
table40 = array2table(metrics_arr40,'VariableNames',var_names);
table45 = array2table(metrics_arr45,'VariableNames',var_names);
table50 = array2table(metrics_arr50,'VariableNames',var_names);

%saving the results to an excel file
writetable(table30,'LLCNN_1Module_assessment_test.xlsx','Sheet','3.0');
writetable(table35,'LLCNN_1Module_assessment_test.xlsx','Sheet','3.5');
writetable(table40,'LLCNN_1Module_assessment_test.xlsx','Sheet','4.0');
writetable(table45,'LLCNN_1Module_assessment_test.xlsx','Sheet','4.5');
writetable(table50,'LLCNN_1Module_assessment_test.xlsx','Sheet','5.0');

disp('Done with 1 module!')

%=========================================================================
%                           LLCNN-3 MODULES
%=========================================================================
NL_exp_res30 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Deep Learning Results\LLCNN\LLCNN_Experiment_Results\3.0\3 modules');
NL_exp_res35 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Deep Learning Results\LLCNN\LLCNN_Experiment_Results\3.5\3 modules');
NL_exp_res40 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Deep Learning Results\LLCNN\LLCNN_Experiment_Results\4.0\3 modules');
NL_exp_res45 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Deep Learning Results\LLCNN\LLCNN_Experiment_Results\4.5\3 modules');
NL_exp_res50 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Deep Learning Results\LLCNN\LLCNN_Experiment_Results\5.0\3 modules');

%assessment for 3.0 image dataset
metrics_arr30 = assessment(NL_images30,NL_exp_res30);
%assessment for 3.5 image dataset
metrics_arr35 = assessment(NL_images35,NL_exp_res35);
%assessment for 4.0 image dataset
metrics_arr40 = assessment(NL_images40,NL_exp_res40);
%assessment for 4.5 image dataset
metrics_arr45 = assessment(NL_images45,NL_exp_res45);
%assessment for 5.0 image dataset
metrics_arr50 = assessment(NL_images50,NL_exp_res50);

%columns titles
var_names = {'MSE','PSNR','SSIM','MV','STD','BRISQUE','NIQE'};

%convert the matrices to tables with column names
table30 = array2table(metrics_arr30,'VariableNames',var_names);
table35 = array2table(metrics_arr35,'VariableNames',var_names);
table40 = array2table(metrics_arr40,'VariableNames',var_names);
table45 = array2table(metrics_arr45,'VariableNames',var_names);
table50 = array2table(metrics_arr50,'VariableNames',var_names);

%saving the results to an excel file
writetable(table30,'LLCNN_3Modules_assessment_test.xlsx','Sheet','3.0');
writetable(table35,'LLCNN_3Modules_assessment_test.xlsx','Sheet','3.5');
writetable(table40,'LLCNN_3Modules_assessment_test.xlsx','Sheet','4.0');
writetable(table45,'LLCNN_3Modules_assessment_test.xlsx','Sheet','4.5');
writetable(table50,'LLCNN_3Modules_assessment_test.xlsx','Sheet','5.0');
```

```
352
353     disp('Done with 3 modules!')
354
355     %=========================================================================
356     %                            LLCNN-5 MODULES
357     %=========================================================================
358     NL_exp_res30 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Deep Learning Results\LLCNN\LLCNN_Experiment_Results\3.0\5 modules');
359     NL_exp_res35 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Deep Learning Results\LLCNN\LLCNN_Experiment_Results\3.5\5 modules');
360     NL_exp_res40 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Deep Learning Results\LLCNN\LLCNN_Experiment_Results\4.0\5 modules');
361     NL_exp_res45 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Deep Learning Results\LLCNN\LLCNN_Experiment_Results\4.5\5 modules');
362     NL_exp_res50 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Deep Learning Results\LLCNN\LLCNN_Experiment_Results\5.0\5 modules');
363
364     %assessment for 3.0 image dataset
365     metrics_arr30 = assessment(NL_images30,NL_exp_res30);
366     %assessment for 3.5 image dataset
367     metrics_arr35 = assessment(NL_images35,NL_exp_res35);
368     %assessment for 4.0 image dataset
369     metrics_arr40 = assessment(NL_images40,NL_exp_res40);
370     %assessment for 4.5 image dataset
371     metrics_arr45 = assessment(NL_images45,NL_exp_res45);
372     %assessment for 5.0 image dataset
373     metrics_arr50 = assessment(NL_images50,NL_exp_res50);
374
375     %columns titles
376     var_names = {'MSE','PSNR','SSIM','MV','STD','BRISQUE','NIQE'};
377
378     %convert the matrices to tables with column names
379     table30 = array2table(metrics_arr30,'VariableNames',var_names);
380     table35 = array2table(metrics_arr35,'VariableNames',var_names);
381     table40 = array2table(metrics_arr40,'VariableNames',var_names);
382     table45 = array2table(metrics_arr45,'VariableNames',var_names);
383     table50 = array2table(metrics_arr50,'VariableNames',var_names);
384
385     %saving the results to an excel file
386     writetable(table30,'LLCNN_5Modules_assessment_test.xlsx','Sheet','3.0');
387     writetable(table35,'LLCNN_5Modules_assessment_test.xlsx','Sheet','3.5');
388     writetable(table40,'LLCNN_5Modules_assessment_test.xlsx','Sheet','4.0');
389     writetable(table45,'LLCNN_5Modules_assessment_test.xlsx','Sheet','4.5');
390     writetable(table50,'LLCNN_5Modules_assessment_test.xlsx','Sheet','5.0');
391
392     disp('Done with 5 modules!')
393     disp('Done with Test Images!')
394     disp('Done!')
```

Figure B.3.8: Assessment script for test set

We first read the ground truth and experimental images using image datastores, and then apply the assessment function to calculate the performance metrics. The results are saved in a suitable excel file that will be used later for the construction of summary tables. We apply this process for each darkness level and each number of convolutional modules, so at the end of the process we will have 3 excel files (one for each number of modules) with each having 5 sheets (one for each darkness level).

LLCNN – 1st variation

We follow exactly the same process as the original LLCNN architecture, with the only difference being the schema of the architecture as well as some choices of the training process. The architecture in this case is shown in figure B.3.9 below.

```
#defining the LLCNN final model
def LLCNN_model(input_tensor,filters,num_of_modules=5):
  '''Implementing the LLCNN architecture. Inputs are the input tensor
     (the normalized image dataset) and the number of Convolutional modules
     the architecture is going to use. If the number of modules is not definned
     we are using 5 convolutional modules.'''
  #first layer of the architecture
  x = tfl.Conv2D(filters=64,kernel_size=(1,1),strides=1,padding='same')(input_tensor)
  x = tfl.ReLU()(x)

  #series of convolutional modules
  for l in range(num_of_modules):
    x = conv_module(x)

  #three more conv layers
  x = tfl.Conv2D(filters=filters,kernel_size=(1,1),strides=1,padding='same')(x)
  x = tfl.ReLU()(x)
  x = tfl.Conv2D(filters=filters,kernel_size=(1,1),strides=1,padding='same')(x)
  x = tfl.ReLU()(x)
  x = tfl.Conv2D(filters=filters,kernel_size=(1,1),strides=1,padding='same')(x)
  x = tfl.ReLU()(x)
  #output of the model
  outputs = tfl.Conv2D(filters=3,kernel_size=(1,1),strides=1,padding='same')(x)

  return outputs
```

Figure B.3.9: first variation of the LLCNN architecture

We see that the implementation is exactly the same, with the only difference being that we have added three 1x1 convolutional layers between the convolutional modules and the output layer. The number of filters in these layers can be defined by the user, but we will use 64. The purpose of these filters is to act as a post-processing step, possibly increasing the performance of the model. The next steps are exactly the same as the previous case, with the only difference being that we train the model for 20 epochs instead of 10.

With the completion of the whole process, we have at our disposal the experimental results. These can be used to calculate performance metrics, which is done with the script shown in Figure B.3.10 below.

```
%=========================================================================
%                          TEST IMAGES
%=========================================================================
disp('Working on Test Images...')

%Ground Truth images
NL_images30 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\GROUND TRUTH DEEP LEARNING\Test\3.0');
NL_images35 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\GROUND TRUTH DEEP LEARNING\Test\3.5');
NL_images40 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\GROUND TRUTH DEEP LEARNING\Test\4.0');
NL_images45 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\GROUND TRUTH DEEP LEARNING\Test\4.5');
NL_images50 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\GROUND TRUTH DEEP LEARNING\Test\5.0');


%=========================================================================
%                            LLCNN-1 MODULE
%=========================================================================
NL_exp_res30 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Deep Learning Results\LLCNN++\LLCNN++_Experiment_Results\3.0\1 module');
NL_exp_res35 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Deep Learning Results\LLCNN++\LLCNN++_Experiment_Results\3.5\1 module');
NL_exp_res40 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Deep Learning Results\LLCNN++\LLCNN++_Experiment_Results\4.0\1 module');
NL_exp_res45 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Deep Learning Results\LLCNN++\LLCNN++_Experiment_Results\4.5\1 module');
NL_exp_res50 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Deep Learning Results\LLCNN++\LLCNN++_Experiment_Results\5.0\1 module');

%assessment for 3.0 image dataset
metrics_arr30 = assessment(NL_images30,NL_exp_res30);
%assessment for 3.5 image dataset
metrics_arr35 = assessment(NL_images35,NL_exp_res35);
%assessment for 4.0 image dataset
metrics_arr40 = assessment(NL_images40,NL_exp_res40);
%assessment for 4.5 image dataset
metrics_arr45 = assessment(NL_images45,NL_exp_res45);
%assessment for 5.0 image dataset
metrics_arr50 = assessment(NL_images50,NL_exp_res50);

%columns titles
var_names = {'MSE','PSNR','SSIM','MV','STD','BRISQUE','NIQE'};

%convert the matrices to tables with column names
table30 = array2table(metrics_arr30,'VariableNames',var_names);
table35 = array2table(metrics_arr35,'VariableNames',var_names);
table40 = array2table(metrics_arr40,'VariableNames',var_names);
table45 = array2table(metrics_arr45,'VariableNames',var_names);
table50 = array2table(metrics_arr50,'VariableNames',var_names);

%saving the results to an excel file
writetable(table30,'LLCNN_PLUS_1Module_assessment_test.xlsx','Sheet','3.0');
writetable(table35,'LLCNN_PLUS_1Module_assessment_test.xlsx','Sheet','3.5');
writetable(table40,'LLCNN_PLUS_1Module_assessment_test.xlsx','Sheet','4.0');
writetable(table45,'LLCNN_PLUS_1Module_assessment_test.xlsx','Sheet','4.5');
writetable(table50,'LLCNN_PLUS_1Module_assessment_test.xlsx','Sheet','5.0');

disp('Done with 1 module!')

%=========================================================================
%                            LLCNN-3 MODULES
%=========================================================================
NL_exp_res30 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Deep Learning Results\LLCNN++\LLCNN++_Experiment_Results\3.0\3 modules');
NL_exp_res35 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Deep Learning Results\LLCNN++\LLCNN++_Experiment_Results\3.5\3 modules');
NL_exp_res40 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Deep Learning Results\LLCNN++\LLCNN++_Experiment_Results\4.0\3 modules');
NL_exp_res45 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Deep Learning Results\LLCNN++\LLCNN++_Experiment_Results\4.5\3 modules');
NL_exp_res50 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Deep Learning Results\LLCNN++\LLCNN++_Experiment_Results\5.0\3 modules');

%assessment for 3.0 image dataset
metrics_arr30 = assessment(NL_images30,NL_exp_res30);
%assessment for 3.5 image dataset
metrics_arr35 = assessment(NL_images35,NL_exp_res35);
%assessment for 4.0 image dataset
metrics_arr40 = assessment(NL_images40,NL_exp_res40);
%assessment for 4.5 image dataset
metrics_arr45 = assessment(NL_images45,NL_exp_res45);
%assessment for 5.0 image dataset
metrics_arr50 = assessment(NL_images50,NL_exp_res50);

%columns titles
var_names = {'MSE','PSNR','SSIM','MV','STD','BRISQUE','NIQE'};

%convert the matrices to tables with column names
table30 = array2table(metrics_arr30,'VariableNames',var_names);
table35 = array2table(metrics_arr35,'VariableNames',var_names);
table40 = array2table(metrics_arr40,'VariableNames',var_names);
table45 = array2table(metrics_arr45,'VariableNames',var_names);
table50 = array2table(metrics_arr50,'VariableNames',var_names);

%saving the results to an excel file
writetable(table30,'LLCNN_PLUS_3Modules_assessment_test.xlsx','Sheet','3.0');
writetable(table35,'LLCNN_PLUS_3Modules_assessment_test.xlsx','Sheet','3.5');
writetable(table40,'LLCNN_PLUS_3Modules_assessment_test.xlsx','Sheet','4.0');
writetable(table45,'LLCNN_PLUS_3Modules_assessment_test.xlsx','Sheet','4.5');
writetable(table50,'LLCNN_PLUS_3Modules_assessment_test.xlsx','Sheet','5.0');
```

```
disp('Done with 3 module!')

%========================================================================
%                            LLCNN-5 MODULES
%========================================================================
NL_exp_res30 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Deep Learning Results\LLCNN++\LLCNN++_Experiment_Results\3.0\5 modules');
NL_exp_res35 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Deep Learning Results\LLCNN++\LLCNN++_Experiment_Results\3.5\5 modules');
NL_exp_res40 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Deep Learning Results\LLCNN++\LLCNN++_Experiment_Results\4.0\5 modules');
NL_exp_res45 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Deep Learning Results\LLCNN++\LLCNN++_Experiment_Results\4.5\5 modules');
NL_exp_res50 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Deep Learning Results\LLCNN++\LLCNN++_Experiment_Results\5.0\5 modules');

%assessment for 3.0 image dataset
metrics_arr30 = assessment(NL_images30,NL_exp_res30);
%assessment for 3.5 image dataset
metrics_arr35 = assessment(NL_images35,NL_exp_res35);
%assessment for 4.0 image dataset
metrics_arr40 = assessment(NL_images40,NL_exp_res40);
%assessment for 4.5 image dataset
metrics_arr45 = assessment(NL_images45,NL_exp_res45);
%assessment for 5.0 image dataset
metrics_arr50 = assessment(NL_images50,NL_exp_res50);

%columns titles
var_names = {'MSE','PSNR','SSIM','MV','STD','BRISQUE','NIQE'};

%convert the matrices to tables with column names
table30 = array2table(metrics_arr30,'VariableNames',var_names);
table35 = array2table(metrics_arr35,'VariableNames',var_names);
table40 = array2table(metrics_arr40,'VariableNames',var_names);
table45 = array2table(metrics_arr45,'VariableNames',var_names);
table50 = array2table(metrics_arr50,'VariableNames',var_names);

%saving the results to an excel file
writetable(table30,'LLCNN_PLUS_5Modules_assessment_test.xlsx','Sheet','3.0');
writetable(table35,'LLCNN_PLUS_5Modules_assessment_test.xlsx','Sheet','3.5');
writetable(table40,'LLCNN_PLUS_5Modules_assessment_test.xlsx','Sheet','4.0');
writetable(table45,'LLCNN_PLUS_5Modules_assessment_test.xlsx','Sheet','4.5');
writetable(table50,'LLCNN_PLUS_5Modules_assessment_test.xlsx','Sheet','5.0');

disp('Done with 5 modules!')
disp('Done with Test Images!')
disp('Done!')
```

Figure B.3.10: Assessment script for the test set

The process is the same as in the previous case, and in the end 3 excel files (one for each number of convolutional modules) with 5 sheets each (one for each darkness level) result. These files will be used to build summary tables, based on which we will evaluate the performance of the model.

LLCNN – 2nd variation

As we mentioned in chapter 3, the model will be trained taking as input the classic LL image input and as output the difference of the LL image from the ground truth case. As an architecture we will use the first variation we described above, so the only thing that changes is the data handling we do. This is shown in the figures below.

```
[ ]  #shape of input and output
     print('======================================================================')
     print('\t\t\tTRAINING DATA')
     print('======================================================================\n')
     print(f'Input shape: {train_input.shape} ')
     print(f'Output shape: {train_output.shape}')

     print('Reshaping and normalizing the data...')

     data = []
     for index in range(train_input.shape[0]):
         data.append(train_input[index,0,:,:,:].reshape(train_input.shape[2],train_input.shape[3],train_input.shape[4]))

     train_input = np.array(data)
     train_input = train_input/255


     data = []
     for index in range(train_output.shape[0]):
         data.append(train_output[index,0,:,:,:].reshape(train_output.shape[2],train_output.shape[3],train_output.shape[4]))

     train_output = np.array(data)
     train_output = train_output/255


     print(f'Input shape: {train_input.shape} ')
     print(f'Output shape: {train_output.shape}')
```

```
======================================================================
                        TRAINING DATA
======================================================================

Input shape: (722, 1, 625, 625, 3)
Output shape: (722, 1, 625, 625, 3)
Reshaping and normalizing the data...
Input shape: (722, 625, 625, 3)
Output shape: (722, 625, 625, 3)
```

```
[ ]  #real output of the CNN
     cnn_output = train_output-train_input
     print(cnn_output.shape)
```

Figure B.3.11: data handling for the training data

We see that we read the data in the same way as before, and then calculate the difference between the ground truth image and the LL image, which will be the output of the model. We follow the same procedure for the validation data, as shown in figure B.3.12.

```
[ ]  #shape of validation input and output
     print('=====================================================================')
     print('\t\t\tVALIDATION DATA')
     print('=====================================================================\n')
     print(f'Input shape: {val_input.shape} ')
     print(f'Output shape: {val_output.shape}')

     print('Reshaping and normalizing the data...')

     data = []
     for index in range(val_input.shape[0]):
         data.append(val_input[index,0,:,:,:].reshape(val_input.shape[2],val_input.shape[3],val_input.shape[4]))

     val_input = np.array(data)
     val_input = val_input/255


     data = []
     for index in range(val_output.shape[0]):
         data.append(val_output[index,0,:,:,:].reshape(val_output.shape[2],val_output.shape[3],val_output.shape[4]))

     val_output = np.array(data)
     val_output = val_output/255


     print(f'Input shape: {val_input.shape} ')
     print(f'Output shape: {val_output.shape}')

     del data
```

```
=====================================================================
                        VALIDATION DATA
=====================================================================

Input shape: (33, 1, 625, 625, 3)
Output shape: (33, 1, 625, 625, 3)
Reshaping and normalizing the data...
Input shape: (33, 625, 625, 3)
Output shape: (33, 625, 625, 3)
```

```
[ ]  #real validation output
     cnn_val = val_output-val_input
     print(cnn_val.shape)
```

Figure B.3.12: data handling for the validation data

Once we have the data ready in the format we want, we need to train the model. This can be seen in figure B.3.13 where we see that as input we give the LL images, while as output we now have the differences we calculated above. The training is done for 20 epochs, as in the previous case.

```
#training the model
print('Training the model...')
model.fit(train_input, cnn_output, epochs=20, batch_size=5, verbose=1,validation_data=(val_input,cnn_val))
print('\n\nTraining completed!')
```

Image B.3.13: training of the model

After completing the training of the model, we need to apply it to the test data to evaluate its generalization ability. First, we apply the model to the

test data to calculate the appropriate difference e for each image, and then we add this predicted difference to the test LL images to produce the final experimental images. This process is shown in figure B.3.14.

```
[ ]  #making prediction with the training set
     test_dif_pred = model.predict(test_input)
```

```
[ ]  #real test predictions
     test_pred = test_input + test_dif_pred
     del test_dif_pred
```

Figure B.3.14: making predictions on the test set

After the final experimental images are produced, we save them in exactly the same way as we analyzed in the first variation. These images will be used to evaluate the method, calculating the evaluation metrics we mentioned above. The metrics are calculated using the script shown in Figure B.3.15 below.

```
%==========================================================================
%                            TEST IMAGES
%==========================================================================
disp('Working on Test Images...')

%Ground Truth images
NL_images30 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\GROUND TRUTH DEEP LEARNING\Test\3.0');
NL_images35 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\GROUND TRUTH DEEP LEARNING\Test\3.5');
NL_images40 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\GROUND TRUTH DEEP LEARNING\Test\4.0');
NL_images45 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\GROUND TRUTH DEEP LEARNING\Test\4.5');
NL_images50 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\GROUND TRUTH DEEP LEARNING\Test\5.0');


%==========================================================================
%                            LLCNN-1 MODULE
%==========================================================================
NL_exp_res30 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Deep Learning Results\LLCNN+DeepOtsu\Actual\LLCNN+DeepOtsu_Experiment_Results\3.0\1 module');
NL_exp_res35 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Deep Learning Results\LLCNN+DeepOtsu\Actual\LLCNN+DeepOtsu_Experiment_Results\3.5\1 module');
NL_exp_res40 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Deep Learning Results\LLCNN+DeepOtsu\Actual\LLCNN+DeepOtsu_Experiment_Results\4.0\1 module');
NL_exp_res45 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Deep Learning Results\LLCNN+DeepOtsu\Actual\LLCNN+DeepOtsu_Experiment_Results\4.5\1 module');
NL_exp_res50 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Deep Learning Results\LLCNN+DeepOtsu\Actual\LLCNN+DeepOtsu_Experiment_Results\5.0\1 module');

%assessment for 3.0 image dataset
metrics_arr30 = assessment(NL_images30,NL_exp_res30);
%assessment for 3.5 image dataset
metrics_arr35 = assessment(NL_images35,NL_exp_res35);
%assessment for 4.0 image dataset
metrics_arr40 = assessment(NL_images40,NL_exp_res40);
%assessment for 4.5 image dataset
metrics_arr45 = assessment(NL_images45,NL_exp_res45);
%assessment for 5.0 image dataset
metrics_arr50 = assessment(NL_images50,NL_exp_res50);

%columns titles
var_names = {'MSE','PSNR','SSIM','MV','STD','BRISQUE','NIQE'};

%convert the matrices to tables with column names
table30 = array2table(metrics_arr30,'VariableNames',var_names);
table35 = array2table(metrics_arr35,'VariableNames',var_names);
table40 = array2table(metrics_arr40,'VariableNames',var_names);
table45 = array2table(metrics_arr45,'VariableNames',var_names);
table50 = array2table(metrics_arr50,'VariableNames',var_names);
```

```
%saving the results to an excel file
writetable(table30,'LLCNN_DO_1Module_assessment_test.xlsx','Sheet','3.0');
writetable(table35,'LLCNN_DO_1Module_assessment_test.xlsx','Sheet','3.5');
writetable(table40,'LLCNN_DO_1Module_assessment_test.xlsx','Sheet','4.0');
writetable(table45,'LLCNN_DO_1Module_assessment_test.xlsx','Sheet','4.5');
writetable(table50,'LLCNN_DO_1Module_assessment_test.xlsx','Sheet','5.0');

disp('Done with 1 module!')

%========================================================================
%                          LLCNN-3 MODULES
%========================================================================
NL_exp_res30 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Deep Learning Results\LLCNN+DeepOtsu\Actual\LLCNN+DeepOtsu_Experiment_Results\3.0\3 modules');
NL_exp_res35 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Deep Learning Results\LLCNN+DeepOtsu\Actual\LLCNN+DeepOtsu_Experiment_Results\3.5\3 modules');
NL_exp_res40 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Deep Learning Results\LLCNN+DeepOtsu\Actual\LLCNN+DeepOtsu_Experiment_Results\4.0\3 modules');
NL_exp_res45 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Deep Learning Results\LLCNN+DeepOtsu\Actual\LLCNN+DeepOtsu_Experiment_Results\4.5\3 modules');
NL_exp_res50 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Deep Learning Results\LLCNN+DeepOtsu\Actual\LLCNN+DeepOtsu_Experiment_Results\5.0\3 modules');

%assessment for 3.0 image dataset
metrics_arr30 = assessment(NL_images30,NL_exp_res30);
%assessment for 3.5 image dataset
metrics_arr35 = assessment(NL_images35,NL_exp_res35);
%assessment for 4.0 image dataset
metrics_arr40 = assessment(NL_images40,NL_exp_res40);
%assessment for 4.5 image dataset
metrics_arr45 = assessment(NL_images45,NL_exp_res45);
%assessment for 5.0 image dataset
metrics_arr50 = assessment(NL_images50,NL_exp_res50);

%columns titles
var_names = {'MSE','PSNR','SSIM','MV','STD','BRISQUE','NIQE'};

%convert the matrices to tables with column names
table30 = array2table(metrics_arr30,'VariableNames',var_names);
table35 = array2table(metrics_arr35,'VariableNames',var_names);
table40 = array2table(metrics_arr40,'VariableNames',var_names);
table45 = array2table(metrics_arr45,'VariableNames',var_names);
table50 = array2table(metrics_arr50,'VariableNames',var_names);

%saving the results to an excel file
writetable(table30,'LLCNN_DO_3Modules_assessment_test.xlsx','Sheet','3.0');
writetable(table35,'LLCNN_DO_3Modules_assessment_test.xlsx','Sheet','3.5');
writetable(table40,'LLCNN_DO_3Modules_assessment_test.xlsx','Sheet','4.0');
writetable(table45,'LLCNN_DO_3Modules_assessment_test.xlsx','Sheet','4.5');
writetable(table50,'LLCNN_DO_3Modules_assessment_test.xlsx','Sheet','5.0');

disp('Done with 3 module!')

%========================================================================
%                          LLCNN-5 MODULES
%========================================================================
NL_exp_res30 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Deep Learning Results\LLCNN+DeepOtsu\Actual\LLCNN+DeepOtsu_Experiment_Results\3.0\5 modules');
NL_exp_res35 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Deep Learning Results\LLCNN+DeepOtsu\Actual\LLCNN+DeepOtsu_Experiment_Results\3.5\5 modules');
NL_exp_res40 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Deep Learning Results\LLCNN+DeepOtsu\Actual\LLCNN+DeepOtsu_Experiment_Results\4.0\5 modules');
NL_exp_res45 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Deep Learning Results\LLCNN+DeepOtsu\Actual\LLCNN+DeepOtsu_Experiment_Results\4.5\5 modules');
NL_exp_res50 = imageDatastore('D:\ΔΙΠΛΩΜΑΤΙΚΗ\Experiments Results\Deep Learning Results\LLCNN+DeepOtsu\Actual\LLCNN+DeepOtsu_Experiment_Results\5.0\5 modules');

%assessment for 3.0 image dataset
metrics_arr30 = assessment(NL_images30,NL_exp_res30);
%assessment for 3.5 image dataset
metrics_arr35 = assessment(NL_images35,NL_exp_res35);
%assessment for 4.0 image dataset
metrics_arr40 = assessment(NL_images40,NL_exp_res40);
%assessment for 4.5 image dataset
metrics_arr45 = assessment(NL_images45,NL_exp_res45);
%assessment for 5.0 image dataset
metrics_arr50 = assessment(NL_images50,NL_exp_res50);

%columns titles
var_names = {'MSE','PSNR','SSIM','MV','STD','BRISQUE','NIQE'};

%convert the matrices to tables with column names
table30 = array2table(metrics_arr30,'VariableNames',var_names);
table35 = array2table(metrics_arr35,'VariableNames',var_names);
table40 = array2table(metrics_arr40,'VariableNames',var_names);
table45 = array2table(metrics_arr45,'VariableNames',var_names);
table50 = array2table(metrics_arr50,'VariableNames',var_names);

%saving the results to an excel file
writetable(table30,'LLCNN_DO_5Modules_assessment_test.xlsx','Sheet','3.0');
writetable(table35,'LLCNN_DO_5Modules_assessment_test.xlsx','Sheet','3.5');
writetable(table40,'LLCNN_DO_5Modules_assessment_test.xlsx','Sheet','4.0');
writetable(table45,'LLCNN_DO_5Modules_assessment_test.xlsx','Sheet','4.5');
writetable(table50,'LLCNN_DO_5Modules_assessment_test.xlsx','Sheet','5.0');

disp('Done with 5 modules!')
disp('Done with Test Images!')
disp('Done!')
```

Zoom: 100%

Figure B.3.15: Assessment script for test set

We start by reading the ground truth images as well as the experimental results for each case of number of convolutional modules, using the feature of image datastores. Then we apply the assessment function to calculate the metrics, saving the final results in excel files. At the end of the process we have at our disposal 3 excel files (one for each case of number of convolutional modules) with each one having 5 sheets, one for each darkness level, which will be used to construct summary tables and diagrams for the evaluation of the method.