ΠΑΝΕΠΙΣΤΗΜΙΟ ΔΥΤΙΚΗΣ ΑΤΤΙΚΗΣ

ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Ανασκόπηση μεθόδων και συστημάτων δημιουργίας πανοραμικών εικόνων και υλοποίηση της μεθόδου ORB με Python

Ιάσων Χαραλαμπίδης
Α.Μ. 711171016

Εισηγητής: Νικόλαος Βασιλάς, ΔΕΠ ΠΑΔΑ

ΑΙΓΑΛΕΩ, Οκτώβρης 2022

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Ανασκόπηση μεθόδων και συστημάτων δημιουργίας πανοραμικών εικόνων και υλοποίηση της μεθόδου ORB με Python

Ιάσων Χαραλαμπίδης
Α.Μ. 711171016

Επιβλέπων Καθηγητής: Νικόλαος Βασιλάς, ΔΕΠ ΠΑΔΑ

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την Παρασκευή 14/10/2022:

(Υπογραφή)	(Υπογραφή)	(Υπογραφή)
Nikolaos Vasilas Digitally signed by Nikolaos Vasilas Date: 2022.10.14 18:00:57 +03'00'	Georgios Bardis Digitally signed by Georgios Bardis Date: 2022.10.16 23:15:21 +03'00'	Christina Georgoulaki Digitally signed by Christina Georgoulaki Date: 2022.10.17 16:59:03 +03'00'
.....	.....	.....
Νικόλαος Βασιλάς	Γεώργιος Μπαρδής	Χριστίνα Γεωργουλάκη
ΔΕΠ ΠΑΔΑ	ΔΕΠ ΠΑΔΑ	ΕΔΙΠ ΠΑΔΑ

ΔΗΛΩΣΗ ΣΥΓΓΡΑΦΕΑ ΠΤΥΧΙΑΚΗΣ/ΔΙΠΛΩΜΑΤΙΚΗΣ ΕΡΓΑΣΙΑΣ

Ο κάτωθι υπογεγραμμένος Ιάσων Χαραλαμπίδης του Σάββα, με αριθμό μητρώου 711171016 φοιτητής του Πανεπιστημίου Δυτικής Αττικής της Σχολής Μηχανικών του Τμήματος Μηχανικών Πληροφορικής και Υπολογιστών, δηλώνω υπεύθυνα ότι:

«Είμαι συγγραφέας αυτής της διπλωματικής εργασίας και ότι κάθε βοήθεια την οποία είχα για την προετοιμασία της είναι πλήρως αναγνωρισμένη και αναφέρεται στην εργασία. Επίσης, οι όποιες πηγές από τις οποίες έκανα χρήση δεδομένων, ιδεών ή λέξεων, είτε ακριβώς είτε παραφρασμένες, αναφέρονται στο σύνολό τους, με πλήρη αναφορά στους συγγραφείς, τον εκδοτικό οίκο ή το περιοδικό, συμπεριλαμβανομένων και των πηγών που ενδεχομένως χρησιμοποιήθηκαν από το διαδίκτυο. Επίσης, βεβαιώνω ότι αυτή η εργασία έχει συγγραφεί από μένα αποκλειστικά και αποτελεί προϊόν πνευματικής ιδιοκτησίας τόσο δικής μου, όσο και του Ιδρύματος.

Παράβαση της ανωτέρω ακαδημαϊκής μου ευθύνης αποτελεί ουσιώδη λόγο για την ανάκληση του πτυχίου μου».

Ο/Η Δηλών/ούσα

……

Ιάσων Χαραλαμπίδης

ΠΕΡΙΛΗΨΗ

Στην παρούσα διπλωματική εργασία έγινε μια ανασκόπηση μεθόδων για τη δημιουργία πανοραμικών εικόνων από δύο ή περισσότερες εικόνες με επικαλυπτόμενες περιοχές ακόμη και αν έχουν ληφθεί υπό διαφορετικές συνθήκες φωτισμού, γωνίες θέασης και χωρικές κλίμακες. Αναφέρονται τα στάδια για την συρραφή εικόνων, οι αλγόριθμοι και μέθοδοι που χρησιμοποιούνται για το ταιριάσματα δύο ή περισσοτέρων εικόνων. Ακόμα, παρουσιάζονται αναφορές σχετικά με την απόδοση των μεθόδων συρραφής. Επιπλέον, παρουσιάζεται η υλοποίηση της μεθόδου συρραφής εικόνων ORB στην γλώσσα Python.

ABSTRACT

This thesis was a review of methods for creating panoramic images from two or more images with overlapping regions even if they are taken under different lighting conditions, viewing angles and spatial scales.The steps for stitching images, algorithms and methods used for matching two or more images are mentioned. Also, reports on the performance of the stitching methods are presented. In addition, the implementation of the ORB image stitching method in Python language is presented.

ΕΠΙΣΤΗΜΟΝΙΚΗ ΠΕΡΙΟΧΗ: Όραση Υπολογιστών

ΛΕΞΕΙΣ ΚΛΕΙΔΙΑ: πανοραμική, αλγόριθμοι, SIFT, SURF, ORB

ΠΕΡΙΕΧΟΜΕΝΑ

ΚΑΤΑΛΟΓΟΣ ΕΙΚΟΝΩΝ

ΚΑΤΑΛΟΓΟΣ ΣΧΗΜΑΤΩΝ

Σχ 6-5: Το μοτίβο δειγματοληψίας BRISK με τους μπλε κύκλους υποδηλώνουν τις θέσεις δειγματοληψίας και τους κόκκινους διακεκομμένους κύκλους, να αντιστοιχούν στην τυπική απόκλιση του Gaussian πυρήνα που χρησιμοποιείται για την εξομάλυνση των τιμών έντασης στα σημεία δειγματοληψίας. Πηγή [41]

ΚΑΤΑΛΟΓΟΣ ΠΙΝΑΚΩΝ

ΣΥΝΤΟΜΟΓΡΑΦΙΕΣ

FAST Features from Accelerated Segment Test

AGAST Adaptive and Generic Accelerated Segment Test

BRIEF Binary robust independent elementary features

FREAK Fast Retina Keypoint

SIFT Scale Invariant Feature Transform

SURF Speeded Up Robust Feautures

ORB Oriented FAST and rotated BRIEF

BRISK Binary Robust Invariant Scalable Keypoint

RANSAC RANdom SAmple Consensus

LMeS Least Median of Squares

ΚΕΦΑΛΑΙΟ 1

ΕΙΣΑΓΩΓΗ

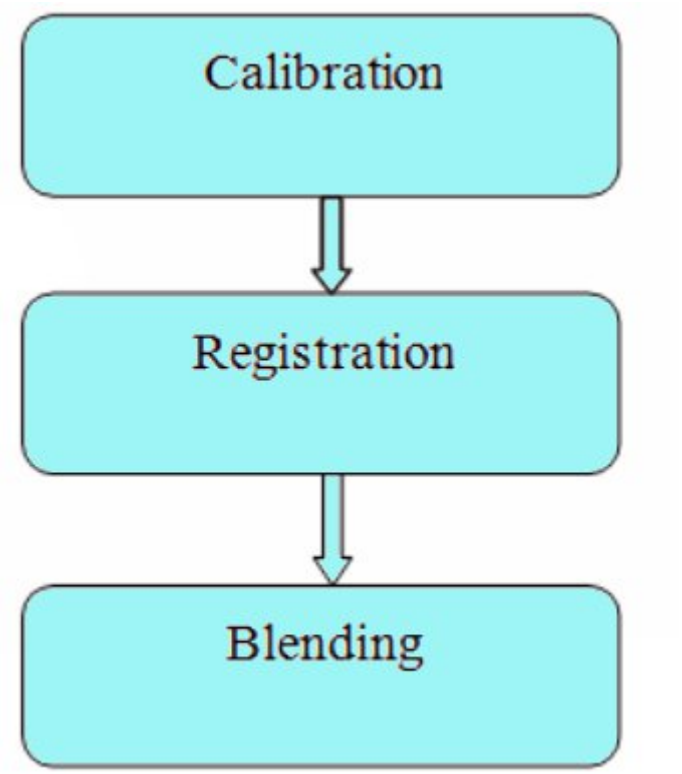
Η συρραφή εικόνων (image stitching) αποτελεί έναν από τους παλιότερους κλάδους της όραση υπολογιστών. Πλέον εφαρμόζεται σε πολλούς τομείς όπως η ιατρική, χαρτογράφηση, ρομποτική, ανίχνευση και παρακολούθηση (tracking) αντικειμένων [81], στην κατασκευή 3D αναπαραστάσεων. Ταυτόχρονα η συρραφή εικόνων συνδέεται άμεσα με την καθημερινή ζωή, ως εργαλείο για τον μέσο άνθρωπο. Στην τωρινή εποχή, η συρραφή εικόνων έχει αναπτυχθεί αρκετά, μέσω προγραμμάτων, σε φορητές και μη συσκευές που έχουν την ικανότητα να συγκολλούν πολλαπλές εικόνες με μεγάλη απόδοση για την δημιουργία πανοραμικών προβόλων. Η ανάπτυξη αυτή δεν είναι τυχαία καθώς έχουν δημιουργηθεί πολλοί αλγόριθμοι με τον σκοπό την συρραφή εικόνων, βίντεο, οπτικών πεδίων και την δημιουργία πανοραμικών εικόνων [1].

Με τη χρήση υπολογιστών, η συρραφή εικόνων, είναι η τεχνική της συνένωσης πολλών φωτογραφικών εικόνων ή καρέ με επικαλυπτόμενα οπτικά πεδία για τη δημιουργία μιας εικόνας φωτομωσαϊκού υψηλής ανάλυσης. Η συρραφή εικόνων δέχεται δύο ή περισσότερες φωτογραφίες ως εισόδους, με την ίδια έκθεση και επικαλυπτόμενα πεδία για να δημιουργήσει μια απρόσκοπτη εικόνα υψηλής ανάλυσης. Μέσω της διαδικασίας της συρραφής εικόνων, πολλές φωτογραφίες μπορούν να συνδυαστούν για να παραχθεί μια μεγαλύτερη εικόνα που υπερβαίνει την τυπική αναλογία διαστάσεων και την ποιότητα των μεμονωμένων εικόνων της φωτογραφικής μηχανής. Η κατασκευή πανοραμικών φωτογραφιών, οι οποίες χρησιμοποιούνται συχνά για τοπία, είναι η πιο γνωστή εφαρμογή της συρραφής εικόνων. Στη δημιουργική φωτογραφία, την ιατρική απεικόνιση, τη δορυφορική φωτογραφία και άλλους τομείς, χρησιμοποιούνται εικόνες ευρείας γωνίας και υψηλής ανάλυσης που παράγονται μέσω της συρραφής εικόνων [1][2].

ΚΕΦΑΛΑΙΟ 2

Τα κύρια σκέλη της συρραφής εικόνων

Η συρραφή εικόνων χωρίζεται σε τρία κύρια σκέλη. Το πρώτο σκέλος είναι η βαθμονόμηση (calibration) της κάμερας, το δεύτερο είναι η εγγραφή εικόνας (image registration) και το τελευταίο είναι η ανάμειξη των εικόνων (image blending). Ο στόχος του calibration της κάμερας είναι η εκτίμηση των παραμέτρων της κάμερα. Έπειτα, κατά την διάρκεια του registration, πολλαπλές εικόνες συγκρίνονται μεταξύ τους και προσδιορίζονται οι μετατοπίσεις για την ευθυγράμμιση τους στον τρισδιάστατο χώρο. Τέλος στην ανάμειξη εικόνων, οι πολλαπλές ευθυγραμμισμένες εικόνες συγχωνεύονται για τη δημιουργία μιας ενιαίας εικόνας, δηλαδή της πανοραμικής [1].



Σχ 2.1: Τα κύρια σκέλη της συρραφής εικόνων

2.1 Βαθμονόμηση εικόνων

Η βαθμονόμηση εικόνας προσπαθεί να ελαχιστοποιήσει οπτικά ελαττώματα και τις διαφορές δύο εικόνων, όπως παραμορφώσεις, διαφορές έκθεσης μεταξύ των εικόνων και διαφορές μεταξύ ιδανικών μοντέλων φακών. Η βαθμονόμηση των εικόνων βοηθά στην σωστή συρραφή δύο ή παραπάνω εικόνων καθώς, η μείωση των διαφορών μεταξύ των εικόνων διευκολύνει τις μεθόδους συρραφής στην διαδικασία καταγραφής. Ταυτόχρονα η βαθμονόμηση συνεισφέρει και στην ανάμιξη των εικόνων, μιας και οι εικόνες που είναι πιο κοντά σε σχέση με φωτεινότητα τους, τα χρώματα τους, την θέσης και απόστασης τους αναμειγνύονται πιο αποτελεσματικά αποδίδοντας ένα πιο ρεαλιστικό αποτέλεσμα [3].

2.2 Καταγραφή εικόνων

Ο στόχος της καταγραφής εικόνων είναι η δημιουργία γεωμετρικής συμφωνίας μεταξύ των εικόνων. Δηλαδή, στην τεχνική αυτή δύο ή περισσότερες εικόνες που έχουν ληφθεί από διαφορετικές οπτικές γωνίες ευθυγραμμίζονται στον τρισδιάστατο χώρο. Η καταγραφή εικόνων αποτελείται από το κομμάτι ανίχνευσης κοινών περιοχών και την ευθυγράμμιση των εικόνων Στην ανίχνευση κοινών περιοχών, τα δεδομένα των εικόνων επεξεργάζονται, ανάλογα με κάποια κριτήρια. Οι μέθοδοι για την ανίχνευση κοινών περιοχών χωρίζονται σε δύο κατηγορίες, τις άμεσες και αυτές που βασίζονται στα χαρακτηριστικά των εικόνων. Μετά την ανίχνευση, οι εικόνες ενώνονται με βάση τις κοινές τους περιοχές, έτσι ώστε οι γεωμετρικές αντιστοιχίσεις των εικόνων αυτών να ευθυγραμμίζονται μεταξύ τους για να μπορούν να συγκριθούν οι ομοιότητες τους. Στο κομμάτι της ευθυγράμμισης, οι εικόνες μετατοπίζονται στον τρισδιάστατο χώρο με βάση τις κοινές τους περιοχές για την ενοποίηση τους. Η καταγραφή εικόνων είναι το θεμελιώδες βήμα σε κάθε διαδικασία συρραφής εικόνων.[4].

2.3 Ανάμειξη εικόνων

Η ανάμειξη εικόνων έχει ως σκοπό την βελτίωση των προσαρμογών που προσδιορίζονται στη βαθμονόμηση εικόνων μέσω της επαναπροσαρμογής εικόνων και της αντιστοίχισης χρωμάτων. Για την παραγωγή μιας απρόσκοπτης τεράστιας εικόνας, οι εικόνες συνδυάζονται, άλλα τα σημεία που τοποθετούνται είναι πολλές

φόρες διακριτά λόγο της απότομης διαφοράς του χρώματος. Οπότε, η ανάμειξη εικόνων έχει ως σκοπό να κάνει την ένωση των δύο εικόνων πιο ομαλή [1][2].

ΚΕΦΑΛΑΙΟ 3

Μέθοδοι καταχώρισης εικόνων

Οι μέθοδοι που χρησιμοποιούνται συνήθως για τη συρραφή εικόνων κατηγοριοποιούνται κυρίως ως μέθοδοι βασισμένη σε περιοχές και μέθοδοι βασισμένοι σε χαρακτηριστικά. ΟΙ μέθοδοι που βασίζεται σε περιοχές, οι οποίες ξεκινάνε από την τιμή του ενός εικονοστοιχείου στην εικόνα και υπολογίζουν την διαφορά έντασης στην ίδια περιοχή μεταξύ, της ίδιας εικόνας και της εικόνας με την οποία θα γίνει η ραφή. Οι συγκεκριμένες μέθοδοι έχουν μεγάλη υπολογιστική πολυπλοκότητα, αλλά χαμηλή ακρίβεια καταχώρισης. Οι μέθοδοι που βασίζονται στα χαρακτηριστικά δεν χρησιμοποιούν απευθείας την τιμή του εικονοστοιχείου σε αντίθεση με τις άμεσες. Αρχικά, εξάγουν χαρακτηριστικά από τις εικόνες και επιλέγουν επικαλυπτόμενες περιοχές χρησιμοποιώντας αυτά τα χαρακτηριστικά που εξάγονται από τις εικόνες δύο εικόνες. Αυτή η κατηγορία μεθόδων, έχει μεγάλη αντοχή και ακρίβεια ακόμα και σε εικόνες με υψηλό θόρυβο ή διαφορετική φωτεινότητα [2].

3.1 Άμεσες τεχνικές

Σε αυτές τις τεχνικές, επιλέγονται κομμάτια της κάθε εικόνας, γνωστά και ως παράθυρα ή μπλοκς, όπου κάθε εικονοστοιχείο σε αυτά τα μπλοκς συγκρίνεται με το αντίστοιχο εικονοστοιχείο σε άλλη εικόνα ελαχιστοποιώντας το άθροισμα των απόλυτων διαφορών μεταξύ των επικαλυπτόμενων εικονοστοιχείων των εικόνων. Για να λειτουργήσει η συγκεκριμένη κατηγορία μεθόδων, θα πρέπει να γίνει αρχικοποίηση, είτε με συσχέτιση είτε με χειροκίνητο ορισμό κάποιων σημείων αντιστοιχίας. Τα παράθυρα που ορίζονται θα πρέπει να έχουν προκαθορισμένο μέγεθος. Κάποιες φορές γίνεται η χρήση ακόμη και ολόκληρων εικόνων για να εκτιμηθεί η αντιστοιχία μεταξύ των εικόνων [5]. Για να γίνει η ευθυγράμμιση των εικόνων, μετατοπίζεται στον τρισδιάστατο χώρο η μία εικόνα σε σχέση με την άλλη, έτσι ώστε τα κοινά τους παράθυρα να εφάπτονται. Όμως, για να μετατοπιστεί μία εικόνα, έτσι ώστε να είναι κατάλληλα ευθυγράμμιση με μία άλλη, πρέπει να

εξαντληθούν όλες οι πιθανές μετατοπίσεις. Στην περίπτωση αυτή, η υπολογιστική πολυπλοκότητα είναι πολύ μεγάλη, χρειάζονται αρκετοί υπολογιστική πόροι για την υλοποίηση και τα αποτελέσματα θα χρειαστούν αρκετό χρόνο για να δημιουργηθούν. Για την επίλυση αυτού του προβλήματος,έχουν δημιουργηθεί διάφοροι μέθοδοι που παράγουν πιο βελτιωμένα αποτελέσματα. Οι μέθοδοι αυτοί χωρίζονται στις εξής τέσσερις βασικές κατηγορίες [6]:

- οι μέθοδοι που βασίζονται στην διασταυρούμενη συσχέτιση
- οι μέθοδοι που βασίζονται στον Fourier
- οι μέθοδοι που βασίζονται στην αμοιβαία πληροφορία
- οι μέθοδοι που βασίζονται στην βελτιστοποίηση

3.1.1 Πλεονεκτήματα και μειονεκτήματα

Το πλεονέκτημα των άμεσων μεθόδων είναι ότι γίνεται πιο δυνατή χρήση των πληροφοριών που είναι διαθέσιμες για την ευθυγράμμιση των εικόνων. Ωστόσο, το περιορισμένο εύρος σύγκλισης είναι το μεγαλύτερο μειονέκτημα που έχουν οι άμεσες μέθοδοι. Πρόκειται για μια πολύπλοκη τεχνική, που δεν παραμένει αναλλοίωτη στην κλίμακα, την περιστροφή, τον θόρυβο και την συμπίεση των εικόνων.Ταυτόχρονα, οι άμεσες τεχνικές έχουν υψηλό υπολογιστικό κόστος και τα αποτελέσματα τους χρειάζονται χρόνο για να δημιουργηθούν. Ως αποτέλεσμα, οι τεχνικές αυτές να θεωρούνται ακατάλληλες για εφαρμογές πραγματικού χρόνου, όπου τα αποτελέσματα πρέπει να παράγονται σε σύντομο χρονικό διάστημα. Ένα άλλο μειονέκτημα των τεχνικών άμεσης συρραφής είναι ότι χρειάζονται αρχικοποίηση, ανθρώπινη επίβλεψη και αλληλεπίδραση για να διασφαλιστεί ότι η συρραφή γίνεται σωστά. Αυτή η αρχικοποίηση περιλαμβάνει, μεταξύ άλλων, την περιστροφή (προσανατολισμός) των εικόνων που πρόκειται να συρραφούν [2].

3.2 Μέθοδοι βασισμένη σε χαρακτηριστικά

Οι τεχνικές που βασίζονται σε χαρακτηριστικά αποσκοπούν στον προσδιορισμό μιας σχέσης μεταξύ των εικόνων. Η σχέση αυτή θα είναι ο προσανατολισμός και η ευθυγράμμιση των εικόνων ώστε να γίνει η συρραφή. Αυτή η σχέση εντοπίζεται με την εξαγωγή διαφορετικών χαρακτηριστικών από τις εικόνες. Οι εικόνες που λαμβάνονται από τον ανιχνευτή χαρακτηριστικών αναλύονται και εξάγονται πληροφορίες για την κάθε εικόνα. Έπειτα, τα χαρακτηριστικά αυτά

περιγράφονται από τον αλγόριθμο περιγραφής της μεθόδου. Αφού γίνει η περιγραφή, τα χαρακτηριστικά της μίας εικόνας συγκρίνονται με τα χαρακτηριστικά της άλλη εικόνας, για να βρεθεί η ευθυγράμμιση τους στον τρισδιάστατο χώρο. Ο σκοπός της σύγκρισης των χαρακτηριστικών είναι η ομαδοποίηση σε ζευγάρια για την επικόλληση των εικόνων [7][9][10][78].

Εικ 3-1: Παράδειγμα ανίχνευσης χαρακτηριστικών

3.2.1 Σημεία κλειδιά

Τα σημεία κλειδιά είναι οι περιοχές που ένας αλγόριθμος ανίχνευσης χαρακτηριστικών εντοπίζει σε μία εικόνα. Ο λόγος που επιλέγονται σημεία κλειδιά από τους αλγόριθμους είναι επειδή έχουν μια ιδιαιτερότητα που μπορεί να τα κατηγοριοποιήσει ως χαρακτηριστικά. Κάθε μέθοδος που βασίζεται στα χαρακτηριστικά των εικόνων, χρησιμοποιεί έναν αλγόριθμο που ανιχνεύει ένα είδος χαρακτηριστικού. Τα είδη χαρακτηριστικών είναι οι γωνίες, οι ακμές ή κηλίδες.

- Οι ακμές συνήθως περιγράφουν τα όρια διαφορετικών τμημάτων ενός αντικειμένου ή δύο αντικειμένων και μέσω της εξέτασης της κατεύθυνσης αυτών των ακμών μπορούν να ανιχνευθούν οι γωνίες [8][79].

Ανασκόπηση μεθόδων και συστημάτων δημιουργίας πανοραμικών εικόνων και υλοποίηση της μεθόδου ORB με Python

- Οι γωνίες είναι σημεία που σχηματίζονται από την τομή δύο ή περισσότερων ακμών .
- Οι κηλίδες είναι περιοχές στις οποίες μια ομάδα εικονοστοιχείων μοιράζεται τις ίδιες ιδιότητες. Κάθε περιοχή έχει διαφορετικές ιδιότητες σε σύγκριση με τις γειτονικές της περιοχές, καθιστώντας έτσι κάθε κηλίδα διαφορετική από κάθε άλλη. Μια κηλίδα αναπαρίσταται, μαθηματικά, ως μία μικρή περιοχή που έχει αξιοσημείωτη διαφορά με τις υπόλοιπες περιοχές γύρω της [82].

Οι αλγόριθμοι ανίχνευσης χαρακτηριστικών αναλύονται στο κεφάλαιο 4.

3.2.2 Περιγραφή εικονοστοιχείων

Μόλις βρεθεί το σημείο κλειδί, το επόμενο βήμα είναι η κατασκευή ενός περιγραφέα που περιέχει πληροφορίες για τα οπτικά χαρακτηριστικά γύρω από το σημείο κλειδί. Ο περιγραφέας δεν πρέπει να είναι ευαίσθητος στην περιστροφή και τον φωτισμό της εικόνας [11]. Γίνεται αναφορά στους αλγόριθμους περιγραφής στο κεφάλαιο 5.

3.2.3 Μέθοδοι ανιχνεύσεις και περιγραφής χαρακτηριστικών

Για να γίνει η αντιστοίχιση των σημείων μεταξύ τον εικόνων από τους αλγόριθμους, θα πρέπει να είναι γνωστά τα σημεία άλλα και η περιγραφή τους. Επειδή όμως ένας αλγόριθμος ανίχνευσης δεν είναι και αλγόριθμος περιγραφής, γίνεται συνδυασμός τον αλγόριθμων ανίχνευσης και περιγραφής, ώστε να ομαδοποιηθούν όλα τα δεδομένα για κάθε σημείο κλειδί [12]. Οι μέθοδοι που συνδυάζουν και το κομμάτι της ανίχνευσης στοιχείων και το κομμάτι της περιγραφής τους, αναλύονται στο κεφάλαιο 6.

3.3 Σύγκριση των δύο κατηγοριών

Οι άμεσες μέθοδοι είναι αρκετά ακριβείς στην συρραφή εικόνων, καθώς ελέγχουν κάθε εικονοστοιχείο μιας εικόνες. Παρόλα αυτά, οι άμεσες τεχνικές είναι αρκετά αργές σε σχέση με τις τεχνικές που βασίζονται στα χαρακτηριστικά των εικόνων. Ταυτόχρονα, χρησιμοποιούν παραπάνω υπολογιστικούς πόρους από ότι οι μέθοδοι που βασίζονται στα χαρακτηριστικά. Έπειτα, η ακρίβεια των άμεσων μεθόδων μειώνεται όσο πιο πολύ θόρυβο ή συμπίεση έχουν υποστεί οι εικόνες [2]. Πλέον οι μέθοδοι που βασίζονται στην ανίχνευση και περιγραφή χαρακτηριστικών,

είναι πιο βελτιωμένες και έχουν αρκετά καλή απόδοση ακόμα και σε εικόνες με έντονο θόρυβο και συμπίεση. Ταυτόχρονα, ο χρόνος και οι πόροι που χρησιμοποιούνται από τους αλγόριθμους ανίχνευσης και περιγραφής χαρακτηριστικών είναι αρκετά λιγότερη από τις άμεσες μεθόδους. Στην συνέχεια της διπλωματικής αυτής, θα γίνει ανάλυση κάποιων γνωστών μεθόδων ανίχνευσης και περιγραφής, καθώς αυτές οι μέθοδοι χρησιμοποιούνται πιο συχνά πλέον.

ΚΕΦΑΛΑΙΟ 4

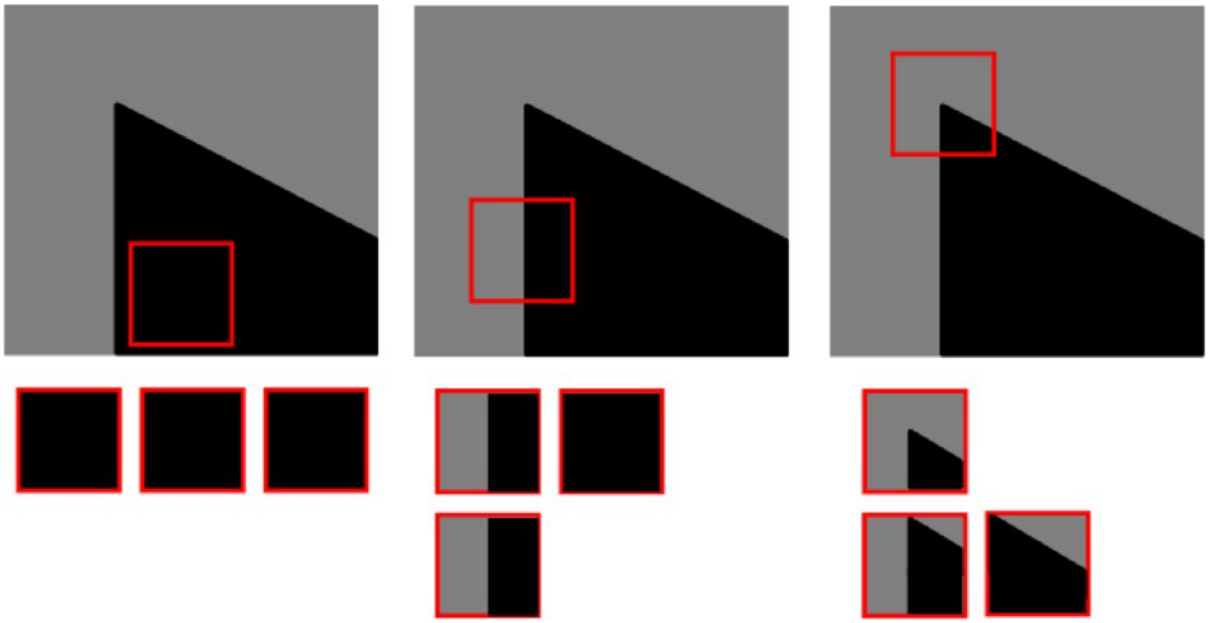
Ανιχνευτές χαρακτηριστικών

4.1 Moravec Corner Detector

Ο ανιχνευτής γωνιών Moravec [15] κάνει χρήση ενός τοπικό παραθύρου στην εικόνα, με στόχο να υπολογίσει τις μέσες αλλαγές στην ένταση της εικόνας που προκύπτουν από τη μετακίνηση του παραθύρου κατά ένα μικρό ποσό προς διάφορες κατευθύνσεις στους επί των δισδιάστατων αξόνων. Συμβολίζοντας τις εντάσεις της εικόνας με I, η μεταβολή E που παράγεται από μια μετατόπιση (x,y) δίνεται από τον τύπο:

$$E(x,y) = \sum_{u,v} w(u,v)\,[I_x + u,y + v - I(u,v)]^2$$

όπου το w καθορίζει το παράθυρο της εικόνας. Οι μετατοπίσεις (x,y) στις τέσσερις κατευθύνσεις, είναι ένα σύνολο μέσα σε μια καθορισμένη ορθογώνια περιοχή $\{(1,0),(1,1),(0,1),(-1,1)\}$. Έτσι, ο ανιχνευτής γωνιών αναζητά τοπικά μέγιστα στο $min\{E\}$ τα οποία έχουν τιμές πάνω από κάποιο κατώφλι [13][14][15].



Σχ 4-1: Περιπτώσεις του αλγόριθμου ανίχνευσης Moravec

Από τον παραπάνω έλεγχο προκύπτουν τα εξής αποτελέσματα που φαίνονται και στην εικόνα .

1. Στην πρώτη περίπτωση, το τμήμα που έχει επιλεχθεί έχει σταθερή ένταση και θεωρείται επίπεδο από τον ανιχνευτή, οπότε όλες οι μετατοπίσεις θα έχουν ως αποτέλεσμα μόνο μια μικρή αλλαγή [14].
2. Στην δεύτερη περίπτωση, εάν υπάρχει μια ακμή που εφάπτεται με το παράθυρο, μια μετατόπιση κατά μήκος της ακμής θα έχει ως αποτέλεσμα μια μικρή μεταβολή, αλλά μια μετατόπιση κατά μήκος της ακμής κάθετη στην άκρη θα έχει ως αποτέλεσμα μεγάλη μεταβολή [14].
3. Στην τρίτη και τελευταία περίπτωση, όπου το patch είναι γωνία ή απομονωμένο σημείο, τότε όλες τις μετατοπίσεις σε μεγάλη αλλαγή. Μια γωνία μπορεί να ανιχνευθεί με την εύρεση της ελάχιστης μεταβολής που παράγεται από οποιαδήποτε από τις μετατοπίσεις είναι μεγάλη [14].

4.1.1 Αδυναμίες του Moravec’s Corner Detector

- Αρχικά, λαμβάνεται υπόψη μόνο ένα διακριτό σύνολο μετατοπίσεων σε διαστήματα 45 μοιρών, καθιστώντας την απόκριση ανισοτροπική [14].
- Επειδή το παράθυρο είναι δυαδικό και ορθογώνιο, η απόκριση είναι θορυβώδης [14].

- Δεδομένου ότι λαμβάνεται υπόψη μόνο το ελάχιστο του E, ο ανιχνευτής ανταποκρίνεται στις ακμές πολύ γρήγορα κάνοντας συχνά λάθη [14] [13].

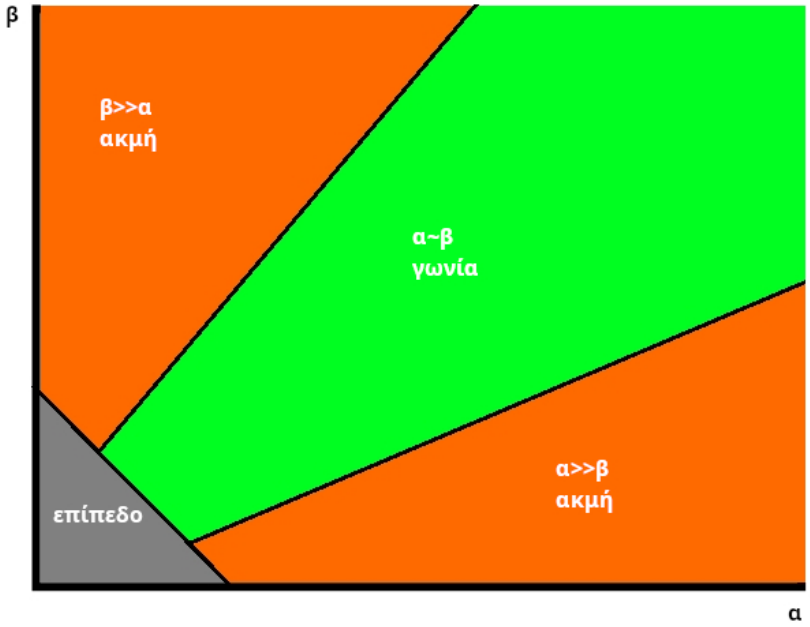
4.2 Harris Corner Detector

Ο Harris Corner Detector [13] δημιουργήθηκε από τους Chris Harris και Mike Stephens με σκοπό να αντιμετωπίσουν τις αδυναμίες του Moravec's και να συνδυάσουν την ανίχνευση γωνιών με την ανίχνευση ακμών και της απόδοσης τους. Το μέτρο απόδοσης R, το οποίο σύμφωνα με την περιστροφή παραμένει αναλλοίωτο, πρέπει να είναι συνάρτηση μόνο των α και β, όπου α και β οι δύο κατευθύνσεις όπου ένα παράθυρο κινείται. Η συμπερίληψη των $Tr(M)$ και $Det(M)$ γίνεται καθώς εξαλείφουν τη ρητή διάσπαση των ιδιοτυπιών του M, δίνοντας τους παρακάτω τύπους:

$$Tr(M) = \alpha + \beta = A + B$$
$$Det(M) = \alpha\beta = AB - C^2$$

Με Tr και Det k μια σταθερά από το 0.04 έως το 0.06, προκύπτει ο τύπος για την απόδοση της γωνιάς:

$$R = Det - kTr^2$$



Σχ 4.2: Περιοχή για κάθε σημείο που δείχνει αν το σημείο είναι γωνία, ακμή ή επίπεδο

Όπως φαίνεται στο σχήμα:

- Πράσινο: το R γίνεται θετικό στις περιοχές με γωνία όπου το α και β έχουν υψηλές τιμές
- Πορτοκαλί: το R γίνεται αρνητικό στις περιοχές στις περιοχές με ακμή όπου υπάρχει μεγάλη διαφορά στα α και β .
- Γκρι: το R έχει χαμηλή τιμή και η περιοχή είναι επίπεδη

Χρησιμοποιώντας αυτόν τον μηχανισμό για την απόδοση της γωνιάς, η αναγνώριση των γωνιών γίνεται εύκολα. Ακόμα, ο ανιχνευτής Harris επιτυγχάνει τον εντοπισμό επιπλέον σημείων έναντι του ανιχνευτή Moravec με το παράθυρο $w(u,v)$ που χρησιμοποιείται για την ανίχνευση. Ο ανιχνευτής του Moravec κάνει χρήση μιας δυαδικής μάσκας, καθώς περιλαμβάνει όλα τα 1 εντός της περιοχής του παραθύρου και όλα τα 0 εκτός αυτής της περιοχής του παραθύρου. Αυτό έχει σαν συνέπεια, το παράθυρο να είναι πολύ πιο ασταθές όταν μία εικόνα έχει θόρυβο. Σε αντίθεση, ο ανιχνευτής γωνιών Harris χρησιμοποιεί ένα ομαλό και κυκλικό παράθυρο το οποίο επιτυγχάνεται με τη βοήθεια της ακόλουθης συνάρτησης Gauss:

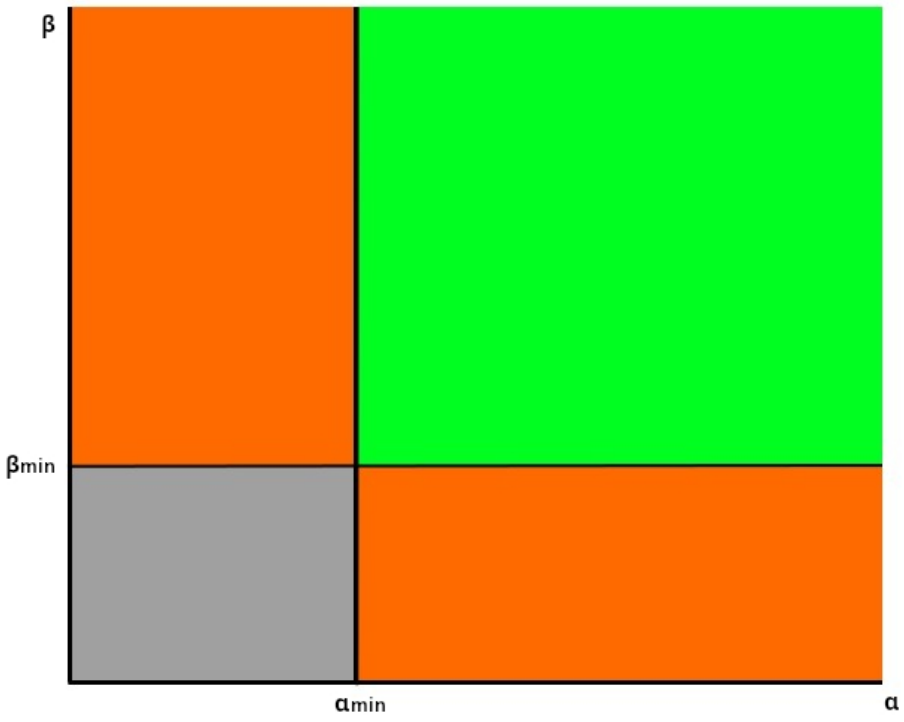
$$w(u,v) = exp - (u^2 + v^2)/2(\sigma^2)$$

4.3 Shi-Tomasi Corner detector

Ο ανιχνευτής Shi-Tomasi ή GoodFeaturesToTrack [16] είναι βασισμένος πάνω στον ανιχνευτή Harris. Ωστόσο, έχει μία αλλαγή σε σχέση με τον ανιχνευτή Harris στο κριτήριο επιλογής των σημείων κλειδιών. Όπως και ο Harris έτσι και ο Shi-Tomasi, υπολογίζει τον βαθμό R για όλα τα εικονοστοιχεία και στη συνέχεια συγκρίνει τον βαθμό R με μια συγκεκριμένη τιμή. Αν ο βαθμός R είναι μεγαλύτερος από την τιμή, τότε το σημείο είναι γωνία. Όμως, ο Harris για την εύρεση της τιμής R κάνει χρήση των δύο ιδιοτιμών $Tr(M)$ και $Det(M)$. Στον ανιχνευτη Shi-Tomasi ο υπολογισμός του R δεν γίνεται με την χρήση των ιδιοτιμών αλλά ισούται με:

$$R = min(a,\beta)$$

Η τιμή R συγκρίνεται με μια ορισμένη τιμή, και ανάλογα το αποτέλεσμα θα χαρακτηριστεί ως σημείο ενδιαφέροντος ή όχι. Στο ακόλουθο γράφημα φαίνεται σε ποια περιοχή ένα σημείο θεωρείται σημείο κλειδί:



Σχ 4.3: Περιοχή για κάθε σημείο που δείχνει αν το σημείο είναι γωνία, ακμή ή επίπεδο

- Πράσινο: τα α και β είναι μεγαλύτερα από μια ορισμένη τιμή. Έτσι, αυτή η περιοχή εικονοστοιχείων θεωρείται από τον αλγόριθμο ως σημείο ενδιαφέροντος [18].
- Πορτοκαλί: Μία από τις ιδιοτιμές είναι μικρότερη από την προκαθορισμένη τιμή της [18].
- Γκρι: Και οι δύο ιδιοτιμές είναι μικρότερες από μια προκαθορισμένη τιμή [18].

Συγκρίνοντας αυτό το σχήμα με τον αλγόριθμο ανίχνευσης Harris. Οι πορτοκαλί περιοχές είναι ίσες με την περιοχή των ακμών, η γκρι περιοχή είναι ανάλογη της επίπεδης περιοχής και η πράσινη περιοχή αντιπροσωπεύει το σημείο ενδιαφέροντος [16][18]. Συμπερασματικά, ο Shi-Tomasi ανιχνευτής λειτουργεί αποτελεσματικά ακόμη και όταν ο ανιχνευτής Harris αποτυγχάνει [17][18].

4.4 SUSAN

Ο αλγόριθμος Smallest Univalue Segment Assimilating Nucleus (SUSAN) [19], ορίζει έναν κύκλο μικρής ακτίνας ως πρότυπο που ονομάζει κυκλική μάσκα ή παράθυρο, και ολισθαίνει το πρότυπο στην εικόνα με μια συγκεκριμένη σειρά. Οι τιμές φωτεινότητας όλων των εικονοστοιχείων εντός της μάσκας συγκρίνονται με εκείνες του κεντρικού εικονοστοιχείου που ονομάζεται "πυρήνας" (nucleus) με τον ακόλουθο τύπο.

$$c\left(\vec{r},\overrightarrow{r_0}\right)=\begin{cases}1 & if|I(\vec{r})-I(\overrightarrow{r_0})|\le t\\ 0 & if|I(\vec{r})-I(\overrightarrow{r_0})|>t\end{cases}$$

Όπου $\vec{r}$ το κάθε εικονοστοιχείο της μάσκας, $\overrightarrow{r_0}$ ο πυρήνας της μάσκας, $I(\vec{r})$ η φωτεινότητα οποιουδήποτε εικονοστοιχείου, t το κατώφλι διαφοράς φωτεινότητας και το c είναι η έξοδος της σύγκρισης. Κάθε m εικονοστοιχείο θα συγκρίνεται με τον πυρήνα με τον ακόλουθο τύπο:

$$c\left(\vec{r},\overrightarrow{r_0}\right)=e^{-\left(\frac{I(\vec{r})-I(\overrightarrow{r_0})}{t}\right)^6}$$

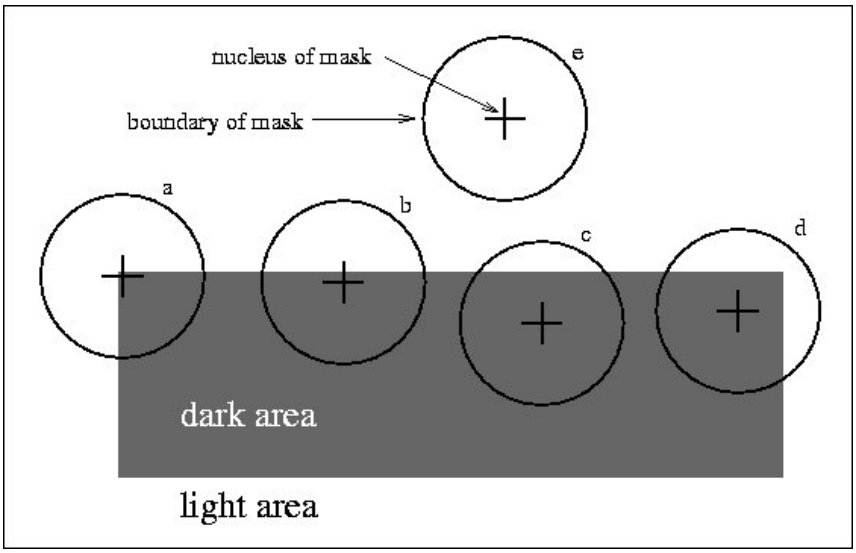
Η χρήση της έκτης δύναμης, μπορεί να αποδειχθεί ότι είναι η θεωρητικά βέλτιστη από τους δημιουργούς του SUSAN, και έχει να κάνει με την σύγκριση που έκαναν με φίλτρα Gauss. Αυτή η σύγκριση γίνεται για κάθε εικονοστοιχείο εντός της μάσκας και προκύπτει ένα τρέχον σύνολο, n, των εξόδων c:

$$n(\overrightarrow{r_0})=\sum_{\vec{r}} c\left(\vec{r},\overrightarrow{r_0}\right)$$

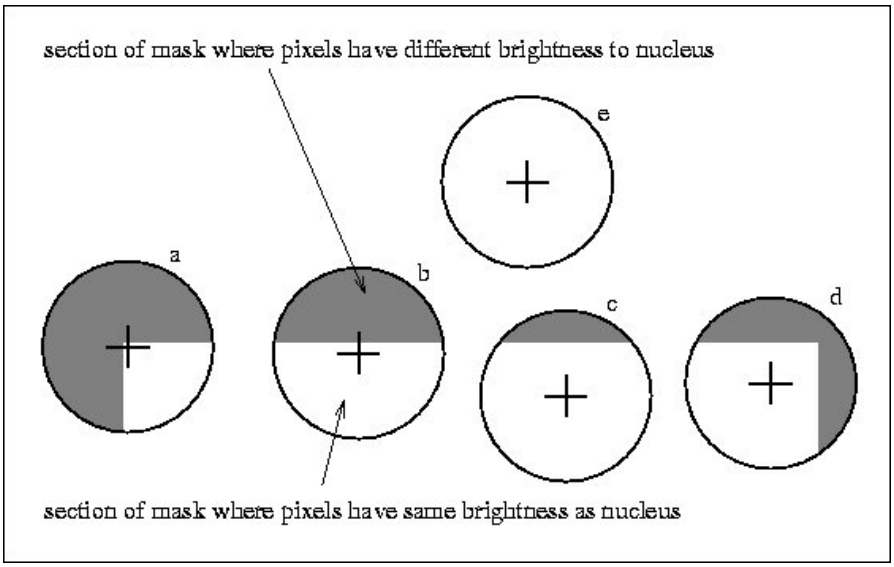
Αν μία περιοχή της μάσκας έχει παρόμοια φωτεινότητα με τον πυρήνα τότε αυτή η περιοχή ονομάζεται Univalue Segment Assimilating Nucleus (USAN) [19]. Η περιοχή USAN περιέχει τις περισσότερες πληροφορίες σχετικά με τη δομή της εικόνας [19].

Το USAN:

- Παίρνει την υψηλότερη τιμή όταν ο πυρήνας βρίσκεται μέσα σε μια επίπεδη περιοχή της εικόνας,
- Μειώνεται στο μισό της μέγιστης τιμής του όταν βρίσκεται πολύ κοντά σε μια ευθεία ακμή
- Μειώνεται ακόμη περισσότερο όταν βρίσκεται μέσα σε μια γωνία.



Σχ 4.4: Τέσσερις κυκλικές μάσκες σε διαφορετικά σημεία μιας απλής εικόνας, με μια μαύρη περιοχή. Πηγή [19]



Σχ 4.5: Τέσσερις κυκλικές μάσκες με χρωματισμό ομοιότητας- οι USAN εμφανίζονται ως τα λευκά μέρη των μασκών. Πηγή [19]

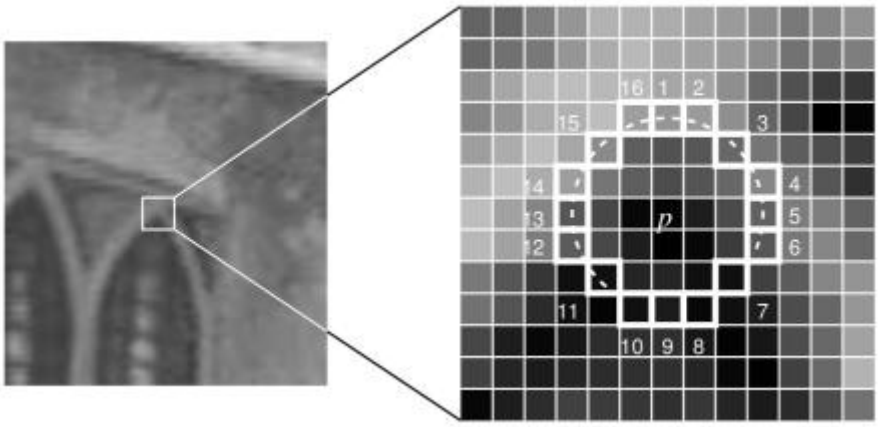
Όπου n ο αριθμός των εικονοστοιχείων στο USAN, αποδίδει το εμβαδόν του USAN. Το n συγκρίνεται με ένα σταθερό κατώφλι g, το οποίο ορίζεται σε $3n_{max}/4$, όπου n_{max} είναι η μέγιστη τιμή που μπορεί να λάβει το n. Στη συνέχεια, η αρχική απόκριση ακμής δημιουργείται με τη χρήση του ακόλουθου κανόνα:

$$R(\vec{r_0}) = \begin{cases} g - n(\vec{r_0}) & if\ n(\vec{r_0}) < g \\ 0 & αλλίως \end{cases}$$

όπου $R(\vec{r_0})$ η θέση της αρχικής αντίδρασης της ακμής.

4.5 FAST Detector

Ο Features from Accelerated Segment Test (FAST) δημιουργήθηκε το 2006 [80] [20]. Ο ανιχνευτής FAST αποδίδει καλά στην ανίχνευση εικονοστοιχείων σε πραγματικά βίντεο. Ο FAST λαμβάνει ένα κατώφλι (threshold) ως παράμετρο έντασης μεταξύ του κεντρικού εικονοστοιχείου (x,y) και εκείνων που βρίσκονται σε έναν κυκλικό δακτύλιο γύρω από το κέντρο. Η εκδοχή FAST-9 του αλγορίθμου FAST, έχει κυκλική ακτίνα 3 pixels με κέντρο το σημείο που θα γίνει η ανίχνευση. Ο FAST-9 είναι η πιο γνωστή και ευρέως διαδεδομένη εκδοχή του FAST.



Εικ 4.1: Αναπαράσταση του κυκλικού δακτύλιου του FAST

4.5.1 Πως λειτουργεί ο FAST

Αρχικά έχει επιλεχθεί ένα κατώφλι t το οποίο θα χρησιμοποιηθεί από τον αλγόριθμο για να κριθεί αν κάποιο εικονοστοιχείο μπορεί να θεωρηθεί ως σημείο ενδιαφέροντος. Η εικόνα στην οποία θα γίνει η επεξεργασία πρέπει πρώτα να γίνει γκριζόχρωμη (grayscaled), έτσι κάθε εικονοστοιχείο να περιγράφεται από μία τιμή I_p που δείχνει την φωτεινότητα του, με τιμές κοντά στο 0 (μαύρο) να είναι σκούρες και τιμές I_p κοντά στο 255 (λευκό) να είναι φωτεινές. Για κάθε εικονοστοιχείο p, δημιουργείται ένας κύκλος 16 εικονοστοιχείων με κέντρο το p και ακτίνα 3 εικονοστοιχεία. Ένα εικονοστοιχείο p θα θεωρηθεί από τον FAST ως γωνία εάν υπάρχει ένα σύνολο n συνεχόμενων εικονοστοιχείων στον κύκλο (των 16 εικονοστοιχείων) τα οποία είναι όλα φωτεινότερα από $I_p + t$, ή όλα σκοτεινότερα από $I_p - t$.

$$S_{p\to x} = \begin{cases} d, & I_{p\to x} \le I_p - t & (\sigma\kappa o\tau\varepsilon\iota\nu\acute{o}\tau\varepsilon\rho o) \\ s, & I_p - t \le I_{p\to x} \le I_p + t & (\acute{o}\mu o\iota o) \\ b, & I_p + t \le I_{p\to x} & (\varphi\omega\tau\varepsilon\iota\nu\acute{o}\tau\varepsilon\rho o) \end{cases}$$

Ο αλγόριθμος μπορεί να βελτιωθεί σε ταχύτητα εάν, συγκρίνετε πρώτα την ένταση των εικονοστοιχείων 1, 5, 9 και 13 του κύκλου με το I_p. Όπως προκύπτει από το παραπάνω σχήμα, τουλάχιστον τρία από αυτά τα τέσσερα εικονοστοιχεία θα πρέπει να ικανοποιούν το κριτήριο “κατώφλι” ώστε να υπάρχει σημείο ενδιαφέροντος. Για ακόμη μεγαλύτερη ακρίβεια, ελέγχονται πρώτα τα $I1$ και $I9$ πρώτα. Αν αυτά τα δυο έχουν τιμή μεγαλύτερη ή μικρότερη του κεντρικού εικονοστοιχείου με βάση τα κριτήρια $I_p + t$ ή $I_p - t$, ανάλογα, τότε ελέγχονται τα $I5$ και $I13$. Εάν τουλάχιστον τρεις από τις τιμές των τεσσάρων εικονοστοιχείων $I1$, $I5$, $I9$, $I13$ δεν είναι πάνω $I_p + t$ ή κάτω $I_p - t$, τότε το p δεν είναι σημείο ενδιαφέροντος (γωνία). Σε αυτή την περίπτωση απορρίπτεται το εικονοστοιχείο p ως πιθανό σημείο ενδιαφέροντος. Διαφορετικά, εάν τουλάχιστον τρία από τα εικονοστοιχεία είναι πάνω $I_p + t$ ή κάτω $I_p - t$, τότε συνεχίζεται ο έλεγχος και για τα υπόλοιπα εικονοστοιχεία. Αυτά τα βήματα επαναλαμβάνονται για κάθε εικονοστοιχείο της εικόνας [80] [20].

4.5.2 Μη μέγιστη καταστολή

Η μη μέγιστη καταστολή [20] δεν μπορεί να εφαρμοστεί απευθείας στα παραγόμενα χαρακτηριστικά, δεδομένου ότι η δοκιμή τμημάτων δεν υπολογίζει μια συνάρτηση απόκρισης γωνίας. Για κάθε εντοπισμένη γωνία πρέπει να υπολογιστεί μια συνάρτηση βαθμολόγησης που ονομάζεται V, και εφαρμόζεται μη μέγιστη καταστολή προκειμένου να διαγραφούν οι γωνίες που έχουν μια γειτονική γωνία με υψηλότερο V και να αφαιρεθούν.Το V έχει διάφορους ορισμούς:

1. Η μεγίστη τιμή τον n ($I_p + t$ ή $I_p - t$) για την οποία το εικονοστοιχείο παραμένει σημείο ενδιαφέροντος.
2. Η μεγίστη τιμή τον t για την οποία το εικονοστοιχείο παραμένει σημείο ενδιαφέροντος.
3. Το άθροισμα της απόλυτης διαφοράς μεταξύ των εικονοστοιχείων στο συνεχόμενο τόξο και του κεντρικού εικονοστοιχείου.

4.6 AGAST

Ο αλγόριθμος Adaptive and Generic Accelerated Segment Test (AGAST) [21] προτάθηκε με σκοπό να λύσει κάποια προβλήματα του FAST. Βασίστηκε στο ίδιο κριτήριο χαρακτηριστικών όπως ο FAST, άλλα με διαφορετικό δέντρο αποφάσεων. Η λογική του δυαδικού δέντρου AGAST είναι η εξής:

1. Επιλογή ενός εικονοστοιχείου για δοκιμή
2. Τίθεται μία ερώτηση σχετικά με την φωτεινότητα του εικονοστοιχείου.
3. Από την απάντηση προσδιορίζεται η ερώτηση και του επόμενου εικονοστοιχείου

Έτσι, η εύρεση μιας γωνίας γίνεται τόσο απλή όσο και η διέλευση από ένα δυαδικό δέντρο αποφάσεων [21]. Δεδομένου ότι ο τύπος της ερώτησης που θα χρησιμοποιηθεί, θα πρέπει να καθοριστεί μαζί με το ποιο εικονοστοιχείο θα ερωτηθεί. Επιπλέον, προσθέτονται δύο ακόμα καταστάσεις στο δέντρο, η κατάσταση “όχι φωτεινότερο” $\left(\overline{b}\right)$ και η κατάσταση “όχι σκοτεινότερο” $\left(\overline{d}\right)$. Όποτε, με την ίδια λογική του αλγορίθμου FAST για ένα εικονοστοιχείο x, η κατάσταση του σε σχέση με το πυρήνα n (όπου εικονοστοιχείο συμβολίζεται ως $n \rightarrow x$) παρουσιάζεται ως:

$$S_{n\to x} = \begin{cases} d, & I_{n\to x} < I_n - t & (darker) \\ \overline{d}, & I_{n\to x} \not< I_n - t \bigwedge S'_{n\to x} = u & (not\ darker) \\ s, & I_{n\to x} \not< I_n - t \bigwedge S'_{n\to x} = \overline{d} & (similar) \\ s, & I_{n\to x} \not> I_n + t \bigwedge S'_{n\to x} = \overline{d} & (similar) \\ \overline{b}, & I_{n\to x} \not> I_n + t \bigwedge S'_{n\to x} = u & (not\ brighter) \\ b, & I_{n\to x} > I_n + t & (brighter) \end{cases}$$

Όπου $S'_{n\to x}$ την προηγούμενη κατάσταση, I η φωτεινότητα ενός εικονοστοιχείου και u να σημαίνει ότι η κατάσταση είναι ακόμη άγνωστη. Αυτό έχει ως αποτέλεσμα, την αναπαράσταση ενός δυαδικού δέντρου, αλλά σε αντίθεση με ένα τριμερές δέντρο, επιτρέποντας μια μόνο αξιολόγηση σε κάθε κόμβο. Είναι αξιοσημείωτο, ότι αυξάνεται το μέγεθος του χώρου διαμόρφωσης σε 6^N, το οποίο αναλογεί σε $6^{16} \approx 2 * 10^{12}$ πιθανούς κόμβους για $N = 16$ [21].

Συνεπώς, μπορούμε εύκολα να συμπεράνουμε ότι το κόστος υπολογισμού για τον AGAST επηρεάζεται από το δέντρο που δημιουργείται. Οι δημιουργοί του αλγόριθμου επισημάνουν ότι, το κόστος υπολογισμού ποικίλλει λόγω διαφορετικών χρόνων πρόσβασης στη μνήμη.

4.7 Ανασκοπήσεις μεθόδων

Στο άρθρο [22] οι συγγραφείς σύγκριναν τους αλγόριθμους SUSAN και HARRIS σε 50 ζευγάρια εικόνων, στην σταθερότητα τους και την ανταπόκριση τους στην αύξηση θορύβου στην εικόνα. Τα αποτελέσματα τους έδειξαν ότι ο Harris είχε την καλύτερη απόδοση και σε θέμα σταθερότητας και σε θέμα αντοχής σε θόρυβο. Έπειτα, έκαναν σύγκριση στην πολυπλοκότητα των δυο αλγορίθμων, και κατέληξαν σε ότι ο Harris είναι πιο γρήγορος από τον SUSAN.

Στο άρθρο [18] έγινε σύγκριση των Harris, Shi-Tomasi και Fast σε 5 εικόνες στην κανονική τους μορφή άλλα και ενισχύοντάς τες με το φίλτρο Adaptive Contrast Enhancement. Τα αποτελέσματα τους έδειξαν ότι ο Shi-Tomasi μπορούσε να ανιχνεύσει περισσότερα σημεία στις εικόνες σε σχέση με των Harris και Fast, άλλα ο Fast είχε την καλύτερη απόδοση χρόνου σε σχέση με τους Harris και Shi-Tomasi, κάνοντας τον Fast προτιμότερο για εφαρμογές πραγματικού χρόνου .

Στο [23] οι συγγραφείς έκαναν σύγκριση τους αλγόριθμους SUSAN, Harris, Harris-Laplace, DoG, Shi-Tomasi, Fast-n (με n ίσο 9 και 12) και την δικιά τους

εκδοχή του FAST που ονομάζουν FAST-ER, πάνω στο σύνολο δεδομένων Oxford [100] μαζί με ένα δικό τους σύνολο δεδομένων. Εξέτασαν τους αλγόριθμους στην επαναληψιμότητα τους πάνω στις εικόνες όταν αυξάνεται ο θόρυβος. Τα αποτελέσματα τους έδειξαν ότι η οικογένεια FAST απέδωσε πολύ καλύτερα από τους υπόλοιπους αλγόριθμους, με ανώτερη την δική τους εκδοχή του FAST, δηλαδή, τον FAST-ER και δεύτερο τον FAST-9. Από τα αποτελέσματα τους η κατάταξη των αλγορίθμων είναι η εξής:

FAST-ER > FAST-9 > DoG > Shi & Tomasi > Harris > Harris Laplace > FAST-12 > SUSAN

Ακόμη, εφάρμοσαν τον ίδιο έλεγχο και για άλλες εκδοχές της οικογένεια FAST με $n \leq 8$, όπου βρήκαν ότι το FAST-9 έχει την καλύτερη επαναληψιμότητα.

Στο άρθρο [21] όπου είναι δεδομένη η προηγούμενης έρευνα [23], οι συγγραφείς σύγκριναν των FAST με τον AGAST και τα βέλτιστα OAST δέντρα, με διαφορετικά μεγέθη μασκών. Τα αποτελέσματα και το συμπέρασμα τους, ήταν πως με την χρήση του δέντρου αποφάσεων AGAST-5 (μάσκα 8 εικονοστοιχείων), είχαν 48.7% καλύτερη απόδοση από τον FAST-9.

Σύμφωνα με τις παραπάνω έρευνες μπορούμε να συμπεράνουμε, ότι οι αλγόριθμοι με δέντρα αποφάσεων είναι προτιμότεροι για την χρήση εφαρμογών πραγματικού χρόνου, καθώς μπορούν να εντοπίσουν περισσότερα στοιχεία σε λιγότερο χρόνο. Η μηχανική μάθηση με αυτούς τους αλγορίθμους είναι μία αρκετά καλή επιλογή, που προσφέρει ακρίβεια και βέλτιστα αποτελέσματα. Τέλος, η διάμετρος της μάσκας του αλγόριθμου είναι σημαντικό κριτήριο για την απόδοση του.

ΚΕΦΑΛΑΙΟ 5

Αλγόριθμοι Περιγραφής χαρακτηριστικών

5.1 BRIEF

Ο αλγόριθμος περιγραφής Binary robust independent elementary features, BRIEF [24] παίρνει όλα τα σημεία κλειδιά που βρέθηκαν από τον αλγόριθμο ανίχνευσης και κατασκευάζει ένα διάνυσμα δυαδικών χαρακτηριστικών. Το διάνυσμα δυαδικών χαρακτηριστικών, γνωστό και ως δυαδικός περιγραφέας χαρακτηριστικών, είναι ένα διάνυσμα χαρακτηριστικών γνωρισμάτων που περιέχει μόνο τις τιμές 1 και 0. Κάθε σημείο κλειδί περιγράφεται από ένα διάνυσμα χαρακτηριστικών γνωρισμάτων που είναι μία συμβολοσειρά 128, 256 και 512 bit, με 256 την πιο γνωστή μορφή. Η πιο συνήθης αναπαράσταση των bits είναι η $k = n_d/8$, όπου η συμβολοσειρά αποτελείται από bytes (8 bits) και κάθε byte έχει τιμή από 0 έως 256. Έστω ένα patch τμήμα μίας εικόνας p με μέγεθος SxS. Ένα δυαδικό τεστ τ ορίζεται ως εξής:

$$\tau(p;x,y) = \begin{cases} 1 : p(x) < p(y) \\ 0 : p(x) \geq p(y) \end{cases}$$

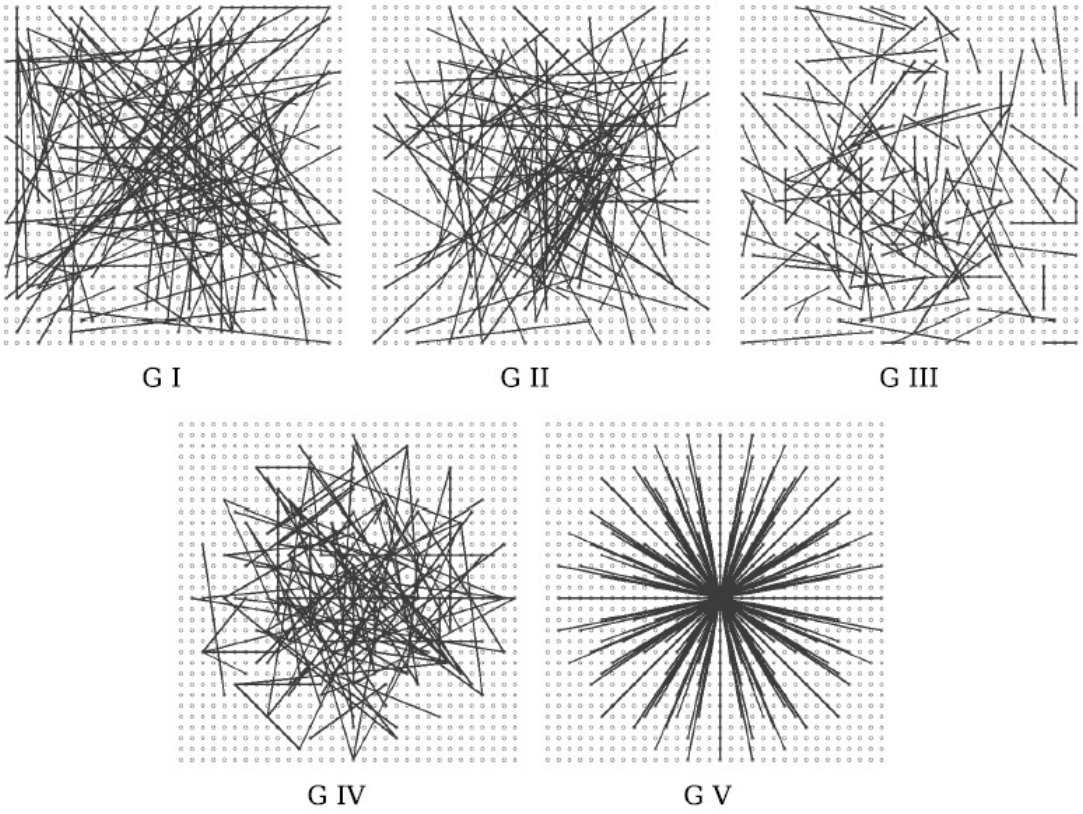
όπου $p(x)$ και $p(y)$ είναι ένταση δύο σημείων στο p. Το χαρακτηριστικό ορίζεται ως ένα διάνυσμα *n* δυαδικών δοκιμών:

$$f(n) = \sum_{1<i<b} 2^{i-1} \tau(p;x_i,y_i)$$

Η επιλογή των συντεταγμένων (x_i,y_i) των ζευγαριών $p(x)$ και $p(y)$ γίνεται με βάση μία από 5 διαφορετικές στρατηγικές.

1. Τα x_i και y_i λαμβάνονται τυχαία με ομοιόμορφη δειγματοληψία.

2. Τα x_i και y_i δειγματοληπτούνται τυχαία χρησιμοποιώντας μια γκαουσιανή κατανομή. Τα ζευγάρια που θα επιλεχθούν θα είναι πιο κοντά στο κέντρο του p.
3. Τα x_i και y_i δειγματοληπτούνται τυχαία χρησιμοποιώντας μια Γκαουσιανή κατανομή, όπου πρώτα το x_i δειγματοληπτείται με τυπική απόκλιση $0{,}04 * S\text{^}2$ και στη συνέχεια τα y_i δειγματοληπτούνται χρησιμοποιώντας μια Γκαουσιανή κατανομή. Η κατανομή του κάθε y_i δειγματοληπτείται με μέση τιμή το x_i και τυπική απόκλιση $0{,}01 * S\text{^}2$.
4. Τα x_i και y_i δειγματοληπτούνται τυχαία από διακριτή θέση μιας χονδροειδούς πολικής ζώνης.
5. Για κάθε i το x_i παίρνει την θέση $(0{,}0)$ και το y_i παίρνει όλες τις δυνατές τιμές σε ένα χονδροειδές πολικό πλέγμα.



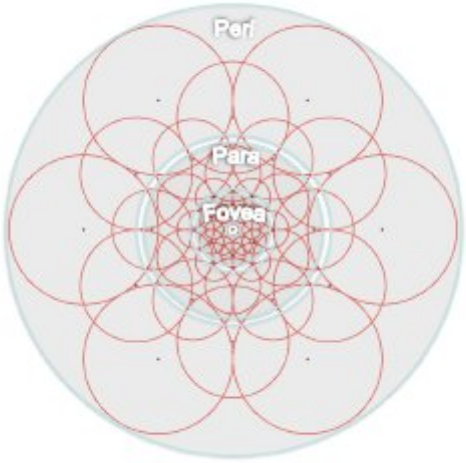
Εικ 5.1: Οι 5 στρατηγικές επιλογής ζευγαριών

5.2 Περιγραφέας FREAK

ο Fast Retina Keypoint (FREAK) [25] είναι ένας γρήγορος, συμπαγής και αξιόπιστος περιγραφέας σημείων κλειδιών. Συγκρίνοντας αποτελεσματικά ζεύγη

εντάσεων εικόνας σε ένα μοτίβο δειγματοληψίας αμφιβληστροειδούς, δημιουργείται ένα διάνυσμα δυαδικών συμβολοσειρών. Η επιλογή ζευγών μπορεί να κάνει τον περιγραφέα λιγότερο διαστατικό, που παράγει ένα εξαιρετικά δομημένο μοτίβο που μοιάζει με την σακκαδική αναζήτηση του ανθρώπινου ματιού [25].

Ο περιγραφέας FREAK είναι βασισμένος στην χρήση του πλέγματος δειγματοληψίας αμφιβληστροειδούς το οποίο είναι επίσης κυκλικό με τη διαφορά ότι έχει μεγαλύτερη πυκνότητα σημείων και βρίσκεται κοντά στο κέντρο [25]. Η ακόλουθη εικόνα δείχνει πώς η πυκνότητα των σημείων μειώνεται εκθετικά:



Εικ 5.2: Απεικόνιση του μοτίβου δειγματοληψίας FREAK παρόμοιο με την αμφιβληστροειδή κατανομή των γαγγλιακών κυττάρων με τα αντίστοιχα τους πεδία. Κάθε κύκλος αντιπροσωπεύει ένα δεκτικό πεδίο όπου η εικόνα εξομαλύνεται με το αντίστοιχο πυρήνα Gauss. Πηγή [25]

Ο περιγραφέας F είναι μια δυαδική συμβολοσειρά που σχηματίζεται από μια ακολουθία διαφορών Γκαουσιανών ενός bit (DoG) [25]:

$$F = \sum_{0 \le a < N} 2^a T(P_a),$$

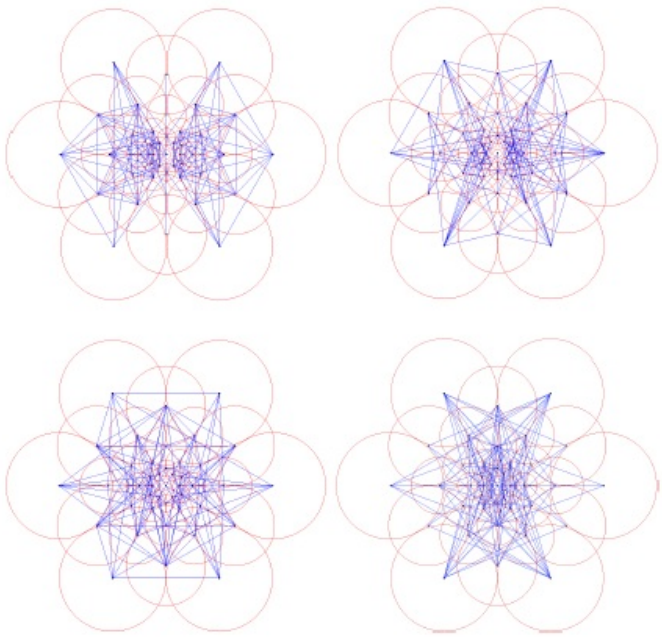
όπου $\underline{P_a}$ είναι ένα ζεύγος δεκτικών πεδίων, N είναι το επιθυμητό μέγεθος του περιγραφέα, και

$$T(P_a) = \begin{cases} 1 & \alpha\nu \ \left(I\left(P_a^{r_1}\right) - I\left(P_a^{r_2}\right)\right) > 0 \\ 0 & \alpha\lambda\lambda\iota\acute{\omega}\varsigma \end{cases}$$

$I\left(P_a^{r_1}\right)$ η εξομαλυμένη ένταση της πρώτης υποδοχής του ζεύγους P_a [25].

Με δεκάδες δεκτικά πεδία, από τα οποία χιλιάδες ζεύγη επιλέγονται. Ως αποτέλεσμα να δημιουργείται ένα μεγάλος περιγραφέας, στον οποίο να υπάρχουν πολλά ζευγάρια τα οποία να μην είναι χρήσιμα. Με σκοπό να μην υπάρχουν όλα τα ζεύγη άλλα μόνο τα καλύτερα για την περιγραφή, τα ζεύγη ελέγχονται από έναν αλγόριθμο ο οποίος τα επιλέγει. Ο αλγόριθμος αυτός λειτουργεί ως εξής [25]:

1. Δημιουργείται ένας πίνακας D από σχεδόν πενήντα χιλιάδες εξαγόμενα σημεία κλειδιά. Κάθε γραμμή αντιστοιχεί σε ένα σημείο κλειδί που αναπαρίσταται με τον μεγάλο περιγραφέα του και αποτελείται από όλα τα πιθανά ζεύγη στο μοτίβο δειγματοληψίας του αμφιβληστροειδούς όπως απεικονίζεται στην εικόνα Εικ 0-1. Χρησιμοποιούνται 43 δεκτικά πεδία που οδηγούν σε περίπου χίλια ζεύγη .
2. Υπολογίζεται ο μέσος όρος κάθε στήλης, προκειμένου να έχουμε ένα διακριτικό χαρακτηριστικό, επιθυμητή είναι η υψηλή διακύμανση. Ένας μέσος όρος 0,5 οδηγεί στην υψηλότερη διακύμανση μιας δυαδικής κατανομής.
3. Οι στήλες ταξινομούνται με βάση την υψηλότερη διακύμανση.
4. Η βέλτιστη στήλη (μέσος όρος 0,5) διατηρείται και προστίθενται επαναληπτικά οι υπόλοιπες στήλες που έχουν χαμηλή συσχέτιση με τις επιλεγμένες στήλες.



Εικ 5.3: Απεικόνιση της ανάλυσης από coarse-to-fine. Η πρώτη συστάδα περιλαμβάνει κυρίως περιφθάλμια δεκτικά πεδία και τα τελευταία βοθρία (fovea).

Τα πρώτα ζεύγη που επιλέγονται, συγκρίνουν κυρίως σημεία δειγματοληψίας στους εξωτερικούς δακτυλίους του προτύπου, ενώ τα τελευταία ζεύγη συγκρίνουν κυρίως σημεία στους εσωτερικούς δακτυλίους του προτύπου. Αυτό είναι συγκρίσιμο με τον τρόπο λειτουργίας του ανθρώπινου ματιού, το οποίο χρησιμοποιεί τα περιφθάλμια δεκτικά πεδία για να καθορίσει που βρίσκεται αρχικά ένα αντικείμενο ενδιαφέροντος. Ο FREAK εκμεταλλεύεται αυτή τη δομή από coarse-to-fine για να επιταχύνει περαιτέρω την αντιστοίχιση, χρησιμοποιώντας μια κλιμακωτή προσέγγιση. Κατά την αντιστοίχιση δύο περιγραφών, συγκρίνονται πρώτα μόνο τα αρχικά 128 bit. Εάν η απόσταση είναι μικρότερη από ένα κατώφλι, τότε συγκρίνονται και τα υπόλοιπα 128 bit. Ως αποτέλεσμα, εκτελείται ένας καταρράκτης συγκρίσεων που επιταχύνει ακόμη περισσότερο την αντιστοίχιση, καθώς πάνω από το 90% των υποψηφίων σημείων απορρίπτεται με τα πρώτα 128 bits του περιγραφέα [25].

ΚΕΦΑΛΑΙΟ 6

Μέθοδοι Ανίχνευσης και Περιγραφής Χαρακτηριστικών

6.1 Scale Invariant Feature Transform

Η μέθοδος SIFT [26] μετασχηματίζει την εικόνα σε μία συλλογή διανυσμάτων χαρακτηριστικών. Αυτά τα διανύσματα χαρακτηριστικών είναι διακριτά και αναλλοίωτα σε οποιαδήποτε κλίμακα, περιστροφή ή μετατόπιση της εικόνας.

6.1.1 Επιλογή κορυφής

Αρχικά, ο αλγόριθμος SIFT εξάγει χαρακτηριστικά από την εικόνα που δίνεται ως είσοδος. Έπειτα, δημιουργεί μία πυραμίδα εικόνων από την εικόνα εισόδου. Κάθε επίπεδο της πυραμίδας, είναι μία εικόνα που έχει υποστεί εξομάλυνση κατά Gauss και υποδειγματοληψία της ανάλυσης της εικόνας του προηγούμενου επιπέδου. Ως αποτέλεσμα, όσο πιο ψηλά είναι τοποθετημένη στην πυραμίδα, τόσο πιο δύσκολο είναι να γίνει εξαγωγή χαρακτηριστικών. Τα χαρακτηριστικά που θα εξάγει ο ανιχνευτής της μεθόδου SIFT ως σημεία ενδιαφέροντος, είναι τα τοπικά ακρότατα, είτε ελάχιστα είτε μέγιστα σε κάθε επίπεδο της πυραμίδας. Ο εντοπισμός θέσεων και μεγεθών που μπορούν να διατεθούν επανειλημμένα για το ίδιο αντικείμενο, υπό διαφορετικές οπτικές γωνίες, είναι το αρχικό βήμα στη διαδικασία ανίχνευσης σημείων-κλειδιών [26]. Βεβαίως, η ανίχνευση θέσεων που δεν μεταβάλλονται στην αλλαγή κλίμακας της εικόνας, απαιτεί την αναζήτηση σταθερών χαρακτηριστικών σε όλες τις πιθανές διακυμάνσεις της κλίμακας, χρησιμοποιώντας μια συνεχή συνάρτηση της κλίμακας γνωστή ως χώρος κλίμακας [26].

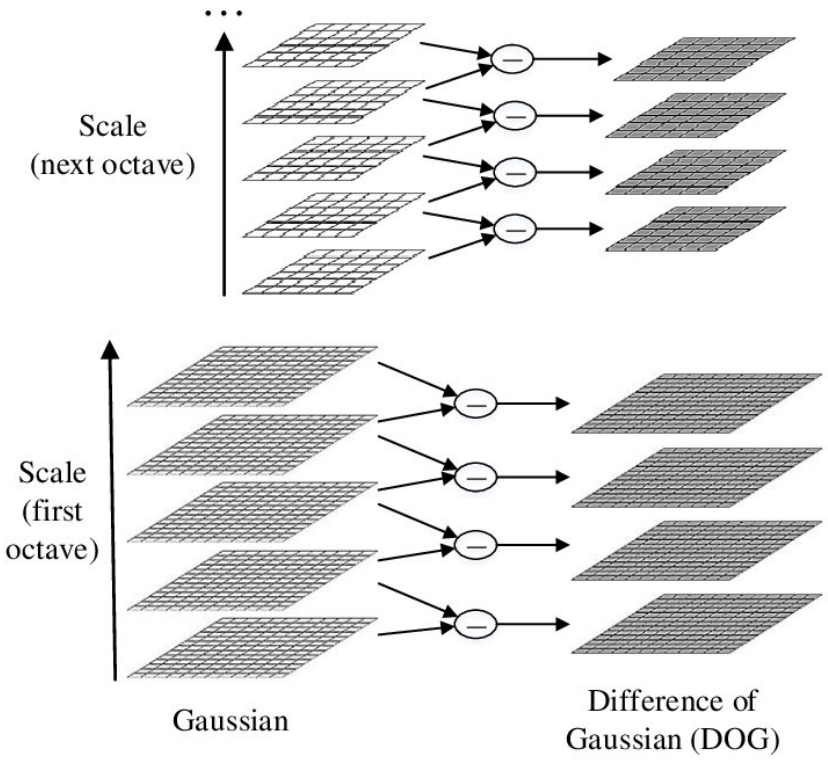
6.1.2 Χώρος κλίμακας και Γκαουσιανή διαφορά

Ο χώρος κλίμακας που αναφέρθηκε παραπάνω, θα ορίζεται ως $L(x,y,\sigma)$ και παράγεται από τη συνέλιξη μιας Γκαουσιανής μεταβλητής κλίμακας $G(x,y,\sigma)$ [26] [27]. Για μία εικόνα ως είσοδο που συμβολίζεται με $I(x,y)$ έχουμε τους παρακάτω τύπους:

$$L(x,y,\sigma) = G(x,y,\sigma) * I(x,y)$$

$$G(x,y,\sigma) = \frac{1}{2\pi\sigma^2} e^{-(x^2+y^2)/2\sigma^2}$$

Έπειτα, γίνεται η χρήση της συνάρτησης διαφοράς γκαουσιανών (DoG) [83], για τον εντοπισμό τοπικών ακρότατων στο χώρο κλίμακας. Η διαφορά γκαουσιανών $D(x,y,\sigma)$ ισούται με την διαφορά δύο εικόνων όπως φαίνεται στην εικόνα Εικ-0-1.



Σχ 6.1: Difference of Gaussianς

Η διαφορά κάθε εικόνας χώρου κλίμακας, με την επόμενη από αυτήν παράγει μια εικόνα DoG. Αυτή η διαδικασία επαναλαμβάνεται μέχρι το τελευταίο επίπεδο της πυραμίδας, όπου είναι δυνατή η δειγματοληψία. Ο υπολογισμός γίνεται με την διαφορά δύο κοντινών κλιμάκων, διαιρούμενη με μια σταθερά παράγοντα k:

$$D(x,y,\sigma) = (G(x,y,\sigma) - G(x,y,\sigma)) * I(x,y) = L(x,y,k\sigma) - L(x,y,\sigma)$$

Τα τοπικά ακρότατα (μέγιστα ή ελάχιστα) της συνάρτησης DoG ανιχνεύονται συγκρίνοντας κάθε εικονοστοιχείο με τα γειτονικά του στον χώρο κλίμακας. Η

αναζήτηση ακρότατων αποκλείει την πρώτη και την τελευταία εικόνα σε κάθε οκτάβα, επειδή δεν έχουν κλίμακα πάνω και κλίμακα κάτω αντίστοιχα. Η ανίχνευση ακρότατων στον χώρο κλίμακας παράγει πάρα πολλά υποψήφια σημεία-κλειδιά, όπου ορισμένα από αυτά είναι ασταθή και λιγότερο χρήσιμα. Για αυτόν τον λόγο γίνεται υπολογισμός της παρεμβολής, χρησιμοποιώντας το τετραγωνικό ανάπτυγμα Taylor της συνάρτησης του χώρου κλίμακας DoG, $D(x,y,\sigma)$ με το υποψήφιο σημείο κλειδί ως αρχή. Αυτή η ανάλυση Taylor δίνεται ως εξής:

$$D(x) = D + \frac{\delta D^T}{\delta x}x + \frac{1}{2}x^T\frac{\delta^2 D}{\delta x^2}x$$

που το D και οι παραγωγές του, αξιολογούνται στον υποψήφιο σημείο και $x = (x,y,\sigma)$ είναι η μετατόπιση από το σημείο αυτό [26][27].

Για να γίνει ο εντοπισμός σημείων κλειδιών, τα εικονοστοιχεία σε μια εικόνα συγκρίνονται με τα 8 γειτονικά τους, τα 9 εικονοστοιχεία στην επόμενη κλίμακα και τα 9 εικονοστοιχεία στις προηγούμενες κλίμακες. Με αυτόν τον τρόπο, πραγματοποιούνται συνολικά 26 έλεγχοι. Εάν πρόκειται για ακρότατο, τότε το εικονοστοιχείο θεωρείται σημείο κλειδί.

6.1.3 Προσανατολισμός σημείου

Για κάθε κορυφή, όπου έχει εντοπιστεί συγκρίνοντας ένα εικονοστοιχείο με τα γειτονικά του εικονοστοιχεία, ένας ή περισσότεροι προσανατολισμοί αποδίδονται με βάση τις τοπικές κατευθύνσεις κλίσης της εικόνας. Για τον προσδιορισμό αυτού του προσανατολισμού, υπολογίζεται ένα ιστόγραμμα προσανατολισμού κλίσης στη γειτονιά του σημείου-κλειδιού. Η κλίμακα του σημείου κλειδιού χρησιμοποιείται για την επιλογή της εικόνας με εξομάλυνση κατά Gauss, $L(x,y,\sigma)$ με την πλησιέστερη κλίμακα σ του σημείου κλειδιού να λαμβάνεται, με σκοπό όλοι οι υπολογισμοί να εκτελούνται με τρόπο αναλλοίωτο ως προς την κλίμακα [26] [27]. Για μία εικόνα $L(x,y)$ στην κλίμακα σ, το μέγεθος της κλίσης $m(x,y)$, και ο προσανατολισμός $\theta(x,y)$, υπολογίζονται εκ των προτέρων χρησιμοποιώντας διαφορές εικονοστοιχείων ως:

$$m(x,y) = \sqrt{(L(x+1,y) - L(x-1,y))^2 + (L(x+1,y) - L(x,y-1))}$$

$$\theta(x,y) = tan^{-1}\left(\frac{L(x,y+1) - L(x,y-1)}{L(x+1,y) - L(x-1,y)}\right)$$

6.1.4 Περιγραφή των σημείων κλειδιών

Στον υπολογισμό της κλίμακας, της θέσης και του προσανατολισμού για κάθε σημείο κλειδί, εφαρμόζεται ένα δισδιάστατο σύστημα συντεταγμένων για την περιγραφή των τοπικών περιοχών που αντιστοιχούν σε κάθε σημείο. Ακολουθεί, ο υπολογισμός ενός περιγραφέα για την τοπική περιοχή της εικόνας που είναι οριστικός αλλά και αμετάβλητος σε πρόσθετες μεταβολές (αλλαγή του φωτισμού, μετακίνηση στο τρισδιάστατο χώρο). Το αποτέλεσμα είναι ένα διάνυσμα 128 διαστάσεων [26] [27].

6.1.5 Ανασκόπηση της μεθόδου SIFT

Η μέθοδος SIFT είναι αμετάβλητη σε κλιμάκωση και περιστροφή, έχει αντοχή στον θόρυβο και μπορεί να ανιχνεύσει και να περιγράψει πολλά σημεία σε μια εικόνα. Παρόλα αυτά, είναι μία μέθοδος η οποία έχει μεγάλο υπολογιστικό κόστος, χρειάζεται αρκετό χρόνο εκτέλεσης και αρκετά από τα σημεία που εντοπίζει είναι άχρηστα στην ενοποίηση των εικόνων [28][29].

6.1.6 Τροποποιήσεις του αλγόριθμου SIFT

Ο SIFT αποδείχθηκε αρκετά καλός αλγόριθμος για την αναζήτηση εικονοστοιχείων. Παρόλα αυτά, όπως αναφέρθηκε παραπάνω χρειάζεται αρκετούς πόρους ώστε να λειτουργήσει αποδοτικά. Για το λόγο αυτό έχουν προταθεί παραλλαγές του SIFT.

Ο PCA-SIFT περιγραφέας που δημιουργήθηκε από τους [29], δέχεται τις ίδιες εισόδους όπως ο SIFT, δηλαδή θέση, κλίμακα και προσανατολισμό, και εξάγει ένα 41x41 τμήμα από την κάθε κλίμακα, με κέντρο το σημείο κλειδί, το οποίο τμήμα περιστρέφεται για να ευθυγραμμίσει τον κυρίαρχο προσανατολισμό του προς μια κανονική κατεύθυνση. Ο PCA-SIFT αποτελεί μια πιο απλοποιημένη εκδοχή του SIFT που ακολουθεί την τεχνική ανάλυσης κύριων συνιστωσών (PCA). Στην πρώτη ανασκόπηση [29] ο PCA-SIFT είχε καλύτερα αποτελέσματα από τον SIFT, όμως στην δεύτερη [30] αποδείχθηκε ότι είναι λιγότερο διακριτή από τον SIFT.

Το ιστόγραμμα θέσης-προσανατολισμού κλίσης (GLOH) [30] είναι μια επέκταση του περιγραφέα SIFT που σχεδιάστηκε για να αυξήσει την ευκρίνεια και τη διακριτικότητά του. Γίνεται υπολογισμός του περιγραφέα SIFT για ένα λογαριθμοπολικό πλέγμα θέσης με τρία bins στην ακτινική κατεύθυνση (η ακτίνα ορίζεται σε 6, 11 και 15) και 8 στη γωνιακή κατεύθυνση, το οποίο οδηγεί σε 17 bins θέσης, με το κεντρικό bin να μην χωρίζεται σε γωνιακές κατευθύνσεις. Οι προσανατολισμοί της κλίσης κβαντίζονται σε 16 bins. Αυτό δίνει ένα ιστόγραμμα 272 δυαδικών ψηφίδων. Το μέγεθος αυτού του περιγραφέα μειώνεται με PCA. Ο πίνακας συνδιακύμανσης για την PCA εκτιμάται σε 47.000 τμήματα εικόνας, που συλλέγονται από πολλαπλές εικόνες. Τα 128 μεγαλύτερα ιδιοδιανύσματα χρησιμοποιούνται για την περιγραφή των σημείων κλειδιών. Ο GHOL στην ανασκόπηση [30] αποδείχθηκε ότι είναι ακόμη πιο διακριτικός με τον ίδιο αριθμό διαστάσεων, αλλά είναι υπολογιστικά πιο ακριβής.

Ο αλγόριθμος Affine-SIFT (ASIFT) [31] προσομοιώνει ένα σύνολο δειγματικών προβολών των αρχικών εικόνων, που μπορούν να ληφθούν μεταβάλλοντας τις δύο παραμέτρους προσανατολισμού του άξονα της κάμερας, δηλαδή τις γωνίες γεωγραφικού πλάτους και γεωγραφικού μήκους, οι οποίες δεν αντιμετωπίζονται από τη μέθοδο SIFT. Στη συνέχεια εφαρμόζει την ίδια τη μέθοδο SIFT σε όλες τις εικόνες που δημιουργούνται με αυτόν τον τρόπο. Έτσι, η ASIFT καλύπτει αποτελεσματικά και τις έξι παραμέτρους του Affine μετασχηματισμού.

6.2 Speeded Up Robust Feautures SURF

Η μέθοδος SURF προτάθηκε από τους [32] με σκοπό να δημιουργηθεί ένας αλγόριθμος εντοπισμού και περιγραφής στοιχείων μιας εικόνας που να είναι άξιος και πιο αποδοτικός της μεθόδου SIFT. Ο ανιχνευτής του SURF βασίζεται στον πίνακα Hessian [33], αλλά χρησιμοποιεί μια πολύ βασική προσέγγιση. Η τροποποίηση του πινάκα Hessian που χρησιμοποιεί η μέθοδος SURF ονομάζεται ανιχνευτής "Fast-Hessian". Ο περιγραφέας, από την άλλη πλευρά, περιγράφει μια κατανομή των αποκρίσεων Haar-wavelet στη γειτονιά του σημείου ενδιαφέροντος. Επιπλέον, χρησιμοποιούνται μόνο 64 διαστάσεις, μειώνοντας το χρόνο για τον υπολογισμό των χαρακτηριστικών και την αντιστοίχιση, αυξάνοντας ταυτόχρονα την ευρωστία [32].

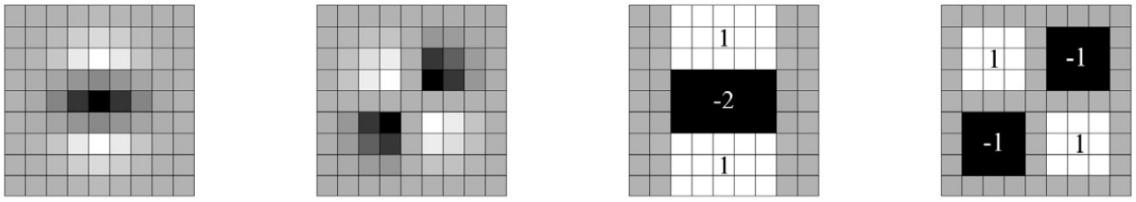
6.2.1 Fast Hessian detector

Ο πίνακας Hessian $H(x,y)$ για ένα σημείο $p(x,y)$ μιας εικόνας σε κλίμακα σ, ορίζεται ως [23]:

$$H(p,\sigma) = \begin{bmatrix} L_{xx}(p,\sigma) & L_{xy}(p,\sigma) \\ L_{xy}(p,\sigma) & L_{yy}(p,\sigma) \end{bmatrix},$$

Όπου $L_{xx}(p,\sigma)$ η συνέλιξη της γκαουσιανής παραγώγου δεύτερης τάξης $\frac{\partial^2}{\partial x^2} g(\sigma)$ με την εικόνα I στο σημείο p, και ομοίως για τα $L_{xy}(p,\sigma$ και $L_{yy}(p,\sigma)$.

Σε αντίθεση με τον SIFT, οι προσεγγίσεις αντί με γκαουσιανά φίλτρα, γίνονται με φίλτρα κουτιού(box filters). Αυτά προσεγγίζουν τις παραγώγους Γκαουσιανής δεύτερης τάξης και μπορούν να αξιολογηθούν πολύ γρήγορα με τη χρήση ολοκληρωτικών εικόνων, ανεξάρτητα από το μέγεθος [23].



Εικ 6.2: Αριστερά προς τα δεξιά: η μερική Gaussian δεύτερης τάξης παραγώγων(διακριτοποιημένη και περικομμένη) στην y-κατεύθυνση και στην xy-κατεύθυνση, και οι προσεγγίσεις τους με τη χρήση box filters. Οι γκρίζες περιοχές είναι ίσες με το μηδέν. Πηγή [23]

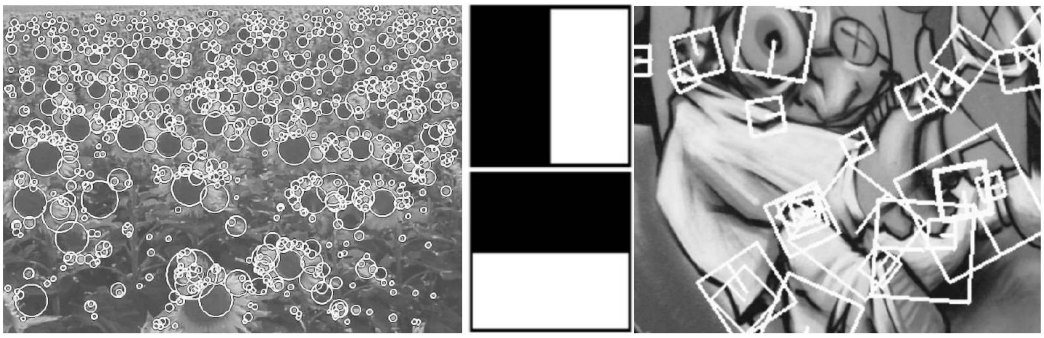
Στην εικόνα **Εικ0.1** απεικονίζονται,τα φίλτρα 9 × 9 box, που είναι προσεγγίσεις για γκαουσιανές παραγώγους δεύτερης τάξης με $\sigma = 1.2$ και αντιπροσωπεύουν τη χαμηλότερη κλίμακα (δηλαδή την υψηλότερη χωροταξικήι ανάλυση) [23]. Οι προσεγγίσεις συμβολίζονται με D_{xx}, D_{xy} και D_{yy}. Οι βαρύτητες που εφαρμόζονται στις ορθογώνιες περιοχές, διατηρούνται απλά για υπολογιστική αποτελεσματικότητα, όμως χρειάζεται να εξισορροπηθούν τα σχετικα βάρη περαιτέρω στην έκφραση για τον προσδιοριστή Hessian με $\frac{|L_{xy}(1.2)|_F |D_{xx}(9)|_F}{|L_{xy}(1.2)|_F |D_{xy}(9)|_F} = 0.912$, με $|x|_F$ να είναι ο κανόνας Frobenius. Αυτό αποδίδει:

$$det\,(H_{approx}) = D_{xx}D_{yy} - (0.9D_{xy})^2$$

Επιπλέον, οι αποκρίσεις των φίλτρων κανονικοποιούνται σε σχέση με το μέγεθος της μάσκας. Αυτό εγγυάται μια σταθερή βάση Frobenius για οποιοδήποτε μέγεθος φίλτρου.

6.2.2 Προσανατολισμός σημείου

Ο προσανατολισμός των σημείων γίνεται με σκοπό, η μέθοδος SURF να είναι αναλλοίωτη στην περιστροφή και στην κλίμακα. Η SURF υπολογίζει πρώτα τις αποκρίσεις Haar-wavelet στις κατευθύνσεις x,y σε μια γειτονιά ακτίνας $6s$ γύρω από το σημείο κλειδί, με s να είναι η κλίμακα στην οποία το σημείο κλειδί εντοπίστηκε. Στη συνέχεια γίνεται ο υπολογισμός του αθροίσματος των κάθετων και οριζόντιων αποκρίσεων wavelet σε μια σαρωμένη περιοχή. Ακολούθως, ο προσανατολισμός της σάρωσης αλλάζει κατά $\pi/3$ και επαναλαμβάνεται ο ίδιος υπολογισμός. Η διαδικασία αυτή συνεχίζει μέχρι να βρεθεί ο προσανατολισμός με τη μεγαλύτερη τιμή αθροίσματος.



Σχ 6.3: Αριστερά: Ανιχνευμένα σημεία ενδιαφέροντος για ένα χωράφι με ηλίανθους. Αυτού του είδους οι σκηνές δείχνουν σαφώς τη φύση των χαρακτηριστικών από ανιχνευτές που βασίζονται στην Hessian.

Μέση: Τύποι Haar walet που χρησιμοποιούνται για την SURF.

Δεξιά: Λεπτομέρεια της σκηνής Graffiti που δείχνει το μέγεθος του περιγραφέα σε διαφορετικές κλίμακες. Πηγή [23].

6.2.3 Περιγραφέας SURF

Για την εξαγωγή του περιγραφέα, κατασκευάζεται για κάθε σημείο κλειδί, μια τετράγωνη περιοχή μεγέθους $20s$, με κέντρο το σημείο και περιστρέφεται ανάλογα με την κατεύθυνση του σημείου. Η περιοχή χωρίζεται σε 4×4 υποπεριοχές τετραγώνων, όπου για κάθε υποπεριοχή, υπολογίζονται απλά χαρακτηριστικά σε 5×5 διατεταγμένα σημεία δειγματοληψίας. Η απόκριση Haar wavelet στην οριζόντια κατεύθυνση συμβολίζεται με d_x και στην κατακόρυφη με d_y. Για να είναι πιο σωστά τα αποτελέσματα και αμετάβλητα στην μετατροπή, οι αποκρίσεις d_x και d_y στην αρχή έχουν Gaussian βάρος $\sigma = 3.3s$ με κέντρο το σημείο κλειδί. Για κάθε υποπεριοχή οι αποκρίσεις wavelet αθροίζονται. Τα αθροίσματα που δημιουργούνται, αποτελούν τα πρώτα στοιχεία για το διάνυσμα της υποπεριοχής. Επιπρόσθετα, αθροίζονται και τα απόλυτα των αποκρίσεων wavelet με σκοπό να εισαχθούν πληροφορίες σχετικά με την πολικότητα των μεταβολών της έντασης. Το αποτέλεσμα για την κάθε υποπεριοχή είναι ένα τετραδιάστατο διάνυσμα $v = (\sum d_x, \sum d_y, \sum |d_x|, \sum |d_y|)$. Ο περιγραφέας συνδυάζει όλες τις περιοχές 4x4 σε ένα μοναδιαίο διάνυσμα(δηλαδή έχει τιμές 0 και 1) μεγέθους 64. Το διάνυσμα του περιγραφέα SURF μπορεί να είναι και 32 θέσεων, αν οι υποπεριοχές είναι 3x3, άλλα και 128 θέσεων αν το άθροισμα των d_x και $|d_x|$ γίνει ξεχωριστά ανάλογα το πρόσημο του d_y και το άθροισμα d_y και $|d_y|$ αντιστοίχωςd_x[32].

6.2.4 Ανασκόπηση της μεθόδου

Η μέθοδος SURF ήταν μία εναλλακτική λύση που δημιουργήθηκε για το πρόβλημα του χρόνου εκτέλεσης της SIFT. Αποδείχτηκε να είναι μια πιο γρήγορη μέθοδος με μικρότερο υπολογιστικό κόστος. Η ανίχνευση της SURF είναι επίσης καλύτερη από την SIFT, αλλά η ακρίβεια της στην περιστροφή και την αλλαγή κλίμακας δεν είναι τόσο αποτελεσματική όσο η SIFT [34][35].

6.3 Oriented FAST and rotated BRIEF

Η μέθοδος ORB [36], αποτελεί τον συνδυασμό του αλγόριθμου εύρεσης εικονοστοιχείων FAST και τον αλγόριθμο BRIEF. Είναι μια μέθοδος περιγραφής εικονοστοιχείων με τη χρήση δυαδικής συμβολοσειράς. Δεδομένου ότι το χαρακτηριστικό σημείο του ORB ανιχνεύεται με τη βελτιωμένη ανίχνευση χαρακτηριστικών FAST και περιγράφεται με τη χρήση ενός βελτιωμένου περιγραφέα χαρακτηριστικών BRIEF. Οι δυο αλγόριθμοι FAST και BRIEF είναι γνωστοί για την ταχύτητα τους, το οποίο καθιστά το ORB μια πολύ καλή επιλογή για περιπτώσεις που χρειάζεται απόδοση. Το σπουδαιότερο χαρακτηριστικό αυτού του αλγορίθμου είναι ότι παραμένει γρήγορος, έχει αναλλοίωτη περιστροφή και μειώνει την ευαισθησία στο θόρυβο.

6.3.1 OFAST Detector

Για την λειτουργία του ORB αρχικά γίνεται η ανίχνευση σημείων μέσω του αλγορίθμου FAST. Ο FAST λαμβάνει παράμετρο το κατώφλι (threshold) έντασης μεταξύ του κεντρικού εικονοστοιχείου (x,y) και εκείνων που βρίσκονται σε έναν κυκλικό δακτύλιο γύρω από το κέντρο. Η εκδοχή FAST-9 του αλγορίθμου FAST, έχει κυκλική ακτίνα 3 pixels με κέντρο το σημείο όπου θα γίνει η ανίχνευση. Ο FAST-9 έχει καλή απόδοση στην εύρεση εικονοστοιχείων [36].

6.3.2 Βαθμός γωνιάς εικονοστοιχείων

Ο FAST δεν μπορεί να υπολογίσει ένα σκορ για τις γωνίες που εντοπίζει. Για αυτό γίνεται η χρήση του αλγορίθμου Harris [14] για την διάταξη των σημείων κλειδιών τα οποία βρήκε ο FAST. Ο αλγόριθμος ανίχνευσης γωνιών Harris υλοποιείται με τον υπολογισμό της κλίσης (gradient) κάθε εικονοστοιχείου. Εάν οι απόλυτες τιμές της κλίσης σε δύο κατευθύνσεις είναι και οι δύο υψηλές, τότε κρίνεται το εικονοστοιχείο ως γωνία. Αρχικά, επιλέγεται ο μέγιστος αριθμός n εικονοστοιχείων που θα χρησιμοποιηθούν. Έπειτα, ορίζεται ένα κατώφλι με αρκετά χαμηλή τιμή με σκοπό να επιλεχθούν περισσότερα εικονοστοιχεία από το μέγιστο αριθμό n. Τα $N > n$ εικονοστοιχεία, ταξινομούνται σύμφωνα με το μέτρο Harris, στην συνέχεια επιλέγονται τα κορυφαία n σημεία. Το FAST δεν παράγει χαρακτηριστικά

πολλαπλών κλιμάκων. Χρησιμοποιούμε μια πυραμίδα κλίμακας της εικόνας, και παράγουμε χαρακτηριστικά FAST (φιλτραρισμένα κατά Harris) σε κάθε επίπεδο της πυραμίδας [36].

6.3.3 Προσανατολισμός με βάση την κεντροειδή ένταση

Ο αλγόριθμος FAST δεν μπορεί να βρει την συνιστώσα προσανατολισμού και τα χαρακτηριστικά πολλαπλών κλιμάκων, οπότε ο ORB δημιουργεί μια πυραμίδα της εικόνας, η οποία αναπαριστά την εικόνα σε διαφορετικά μεγέθη. Έπειτα, κάθε επίπεδο στην πυραμίδα επεξεργάζεται από την FAST για να βρεθούν τα ενδιαφέρον εικονοστοιχεία για το κάθε επίπεδο. Με αυτόν το τρόπο, η ORB γίνεται μερικώς αναλλοίωτη σε κλίμακα. Αφού εντοπιστούν τα σημεία κλειδιά, ορίζεται ένας προσανατολισμός σε κάθε σημείο κλειδί, ανάλογα με το πώς αλλάζουν τα επίπεδα έντασης γύρω από αυτό το σημείο κλειδί. Για την ανίχνευση της αλλαγής έντασης η ORB χρησιμοποιεί το κεντροειδές έντασης. Το κεντροειδές έντασης υποθέτει ότι η ένταση μιας γωνίας είναι μετατοπισμένη από το κέντρο της και αυτό το διάνυσμα μπορεί να χρησιμοποιηθεί για να προσδώσει έναν προσανατολισμό [36]. Για τη βελτίωση της αναλλοίωτης περιστροφής, οι ροπές υπολογίζονται με x και y, που πρέπει να βρίσκονται σε μια κυκλική περιοχή ακτίνας r, όπου r είναι το μέγεθος του τμήματος της γωνίας (patch). Το τμήμα μιας γωνίας ή εικονοστοιχείου ορίζεται ως:

$$m_{pq} = \sum_{x,y \in B} x^p y^q I(x,y), \quad p,q = \{0,1\}$$

Με αυτές τις ροπές η ORB μπορεί να βρει το κεντροειδές, το "κέντρο μάζας" του μπαλώματος ως εξής:

$$C = \left(\frac{m_{10}}{m_{00}}, \frac{m_{01}}{m_{00}}\right)$$

Ο προσανατολισμός του μπαλώματος δίνεται τότε από:

$$\theta = atan2(m_{01}, m_{10})$$

6.3.4 Περιγραφέας rBRIEF

Η μέθοδος ORB για την περιγραφή των χαρακτηριστικών χρησιμοποιεί τον αλγόριθμο BRIEF [24] αλλά με παραλλαγές.Ο νέος αλγόριθμος BRIEF, ο rBRIEF είναι η εκδοχή [26] που έδωσαν η συγγραφείς στην μέθοδο ORB. Ο αλγόριθμος rBRIEF προσδιορίζεται ως εξής [36]:

1. Εκτελέστε κάθε δοκιμή έναντι όλων των επιφανειών εκπαίδευσης.
2. Ταξινομήστε τις δοκιμές ανάλογα με την απόστασή τους από μια μέση τιμή 0,5, σχηματίζοντας το διάνυσμα T.
3. Άπληστη αναζήτηση:

Τοποθετείτε η πρώτη δοκιμή στο διάνυσμα αποτελεσμάτων R και αφαιρείτε την από το T. Το επόμενο τεστ από το T συγκρίνετε το με όλα τα τεστ στο R. Εάν η απόλυτη συσχέτισή του είναι μεγαλύτερη από ένα κατώφλι, τότε απορρίπτεται, διαφορετικά προσθέτετε το στο R. Επαναλαμβάνετε το προηγούμενο βήμα έως ότου υπάρχουν 256 δοκιμές στο R. Εάν είναι λιγότερες από 256, αυξάνετε το κατώφλι και προσθέτετε ξανά. Το rBRIEF παρουσιάζει σημαντική βελτίωση της διακύμανσης και της συσχέτισης σε σχέση με το κατευθυνόμενο BRIEF [36].

6.3.5 Ανασκόπηση της μεθόδου ORB

Η μέθοδος ORB είναι μία αρκετά πιο γρήγορη εναλλακτική των SIFT και SURF. Τα σημεία χαρακτηριστικών που εντοπίζει είναι σε ομοιόμορφη κατανομή, και η ταχύτητα συρραφής είναι πολύ ταχύτερη από τις SIFT και SURF. Επίσης, ο αλγόριθμος είναι αρκετά ανθεκτικός στην περιστροφή. Ωστόσο, η ORB έχει χαμηλή απόδοση σε εικόνες με αρκετά διαφορετικό φωτισμό και θόρυβο.

6.3.6 Τροποποιήσεις της μεθόδου ORB

Η μέθοδος ORB είναι αρκετά διάσημη στον κόσμο της όρασης υπολογιστών, αλλά οι αδυναμίες της αποτελούν λόγο για την βελτίωση της.

Στο πλαίσιο της ανίχνευσης δυναμικών χαρακτηριστικών, έχει αναπτυχθεί μια τεχνική γρήγορης ανίχνευσης χαρακτηριστικών από [37] που βασίζεται στην ORB. Συνδυάζοντας αλγόριθμους προσαρμοστικού ιστογραφικού φιλτραρίσματος, υπολογίζοντας το τοπικό ιστόγραμμα της εικόνας εισόδου και στη συνέχεια ανακατανέμοντας την φωτεινότητα για την αλλαγή της αντίθεσης της εικόνας

εισόδου. Έτσι βελτιώνεται η τοπική αντίθεση της εικόνας εισόδου και ανιχνεύονται περισσότερες λεπτομέρειες της εικόνας.

Στο άρθρο [38] για το ζήτημα της ανομοιομορφίας στην εξαγωγή χαρακτηριστικών της ORB, παρουσιάστηκε μια προσέγγιση εξαγωγής χαρακτηριστικών με βάση το φιλτράρισμα πλέγματος και ενσωματώθηκε η μέθοδος RANSAC για την αύξηση της ακρίβειας αντιστοίχισης.

Στο άρθρο [39] οι συγγραφείς πρότεινα έναν αλγόριθμο βελτιστοποίησης χαρακτηριστικών, που βασίζεται σε στατιστικά στοιχεία κίνησης πλέγματος και μετατρέπει το ζήτημα των περιορισμών εξομάλυνσης της κίνησης σε στατιστική μέτρηση. Συνάμα, πρότειναν έναν αποτελεσματικό εκτιμητή κλασμάτων, με βάση το πλέγμα για την επίτευξη απόρριψης ψευδούς ταύτισης.

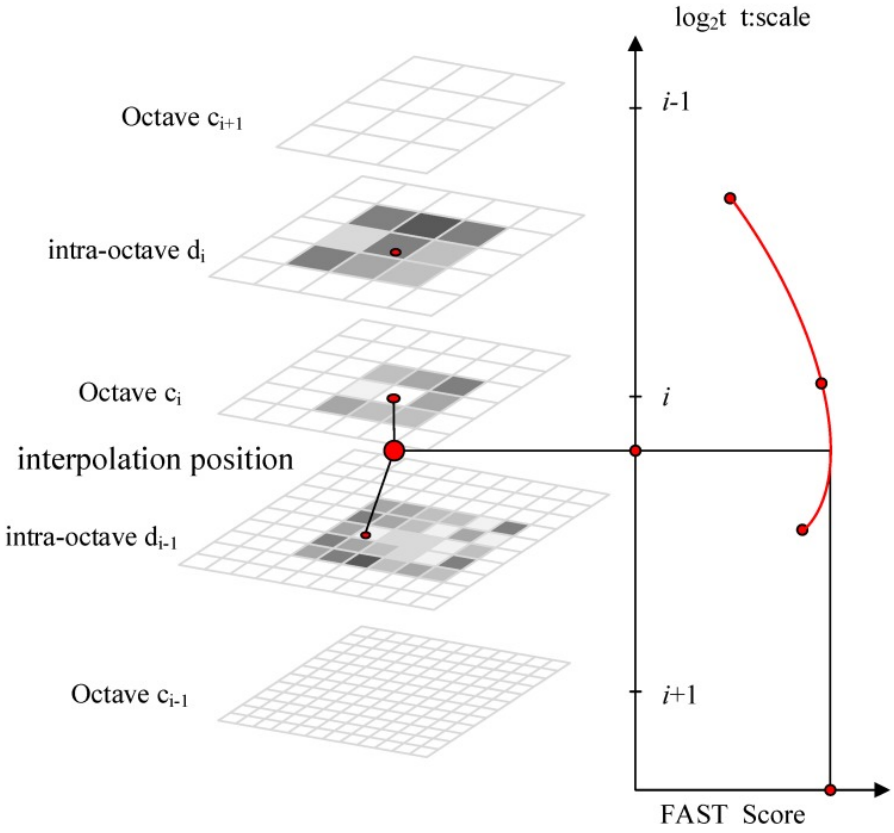
Πρότειναν [40] έναν συνδυασμό της μεθόδου SIFT με τη μέθοδο ORB που ονόμασαν SIRB (SIFT και ORB). Ο SIRB σχεδιάστηκε για να λύσει το ελάττωμα της ασυνέπειας της κλίμακας που έχει η μέθοδος ORB, διατηρώντας παράλληλα μια ανάλογη ταχύτητα με αυτήν.

6.4 Binary Robust Invariant Scalable Keypoint

Η BRISK προτάθηκε από τον [41] ως μία υψηλής ποιότητας μέθοδος, γρήγορης ανίχνευσης σημείων κλειδιών, περιγραφής και αντιστοίχισης. Είναι αναλλοίωτη σε σημαντικό βαθμό τόσο στην περιστροφή, όσο και στην κλίμακα, και επιτυγχάνει επιδόσεις συγκρίσιμες με άλλες μεθόδους, όπως SIFT και SURF ενώ έχει μειωμένο υπολογιστικό κόστος σε σχέση με αυτές.

6.4.1 Κλιμακούμενο σημείο κλειδί

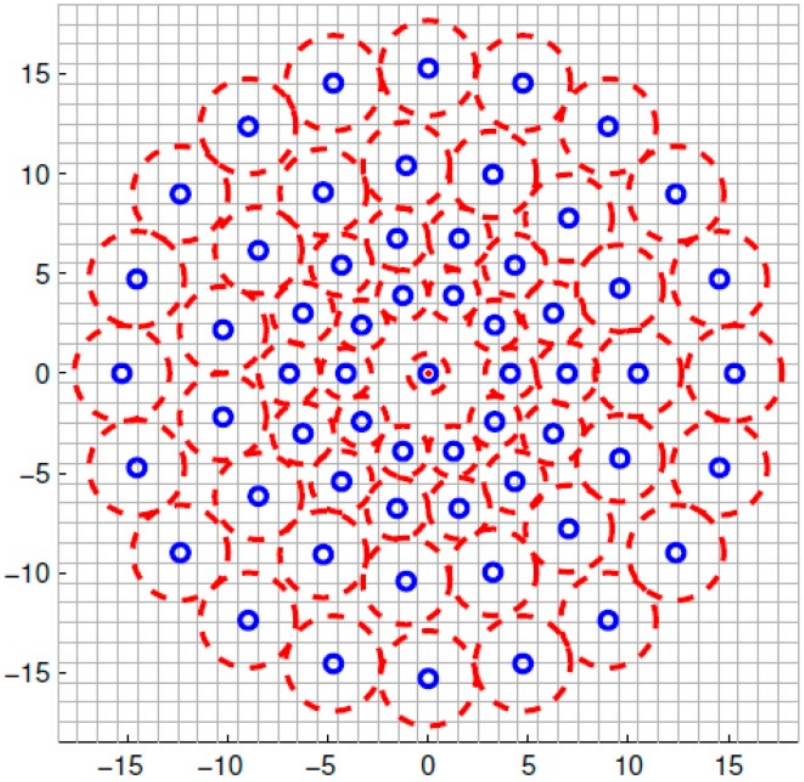
Για τον εντοπισμό τον σημείων κλειδιών, χρησιμοποιείται ο αλγόριθμος αναζήτησης AGAST [21]. Με στόχο την επίτευξη αναλλοίωτης κλίμακας, η οποία είναι ζωτικής σημασίας για υψηλής ποιότητας σημείων κλειδιά. Η έκδοση AGAST της μεθόδου BRISK πηγαίνει ένα βήμα παραπέρα, αναζητώντας τα μέγιστα όχι μόνο σε ένα επίπεδο της εικόνας, αλλά και σε διαφορετικά επίπεδα στην πυραμίδα χώρου-κλίμακας, χρησιμοποιώντας το σκορ του αλγόριθμου FAST ως μέτρο για την προεξοχή [41]. Ο χώρος της πυραμίδας κλίμακας αποτελείται από n οκτάβες c_i και n ενδοκτάβες d_i, όπου $i = \{0,1, \cdots, n-1\}$ και τυπικά $n = 4$. Η πρώτη ενδοκτάβα d_i δημιουργείται με υποδειγματοληψία της αρχικής εικόνας c_0 κατά έναν παράγοντα 1.5, ενώ τα υπόλοιπα στρώματα ενδοκτάβας προκύπτουν με διαδοχικές μισές δειγματοληψίες. Επομένως, εάν το t δηλώνει την κλίμακα το $t(c_i) = 2^i$ και το $t(d_i) = 2^i * 1.5$ [41].



Σχ 6-4: Ανίχνευση σημείου σε διαφορετικές οκτάβες. Πηγή [41]

6.4.2 Περιγραφέας BRISK

Κατά την εξέταση κάθε σημείου δειγματοληψίας της εικόνας, εφαρμόζεται εξομάλυνση Gauss σε ένα τμήμα γύρω από το σημείο κλειδί. Όπως φαίνεται στην εικόνα ΕΙΚ, ο κόκκινος κύκλος απεικονίζει το μέγεθος της τυπικής απόκλισης του φίλτρου Gauss που εφαρμόζεται σε κάθε σημείο της δειγματοληψίας.



Ο περιγραφέας της μεθόδου BRISK εφαρμόζει το μοτίβο δειγματοληψίας περιστρεφόμενο κατά $\alpha = arctan2\,(g_y, g_x)$ γύρω από το σημείο κλειδί. Ο περιγραφέας συναρμολογείται, εκτελώντας όλες τις συγκρίσεις έντασης μικρής απόστασης των ζευγών σημείων, έτσι ώστε κάθε bit να αντιστοιχεί σε [41]:

$$b = \begin{cases} 1, & I\left(p_j^a, \sigma_j\right) > I\left(p_i^a, \sigma_i\right) \\ 0, & \alpha\lambda\lambda\iota\acute{\omega}\varsigma \end{cases}, \forall \left(p_i^a, p_j^a\right) \in S$$

Με p_i^a, p_j^a το ζεύγος δύο σημείων, b ένα bit της συμβολοσειράς που δημιουργεί ο περιγραφέας και σ η ακτίνα, που αντιστοιχεί στην τυπική απόκλιση του Gaussian πυρήνα που χρησιμοποιείται για την εξομάλυνση των τιμών έντασης στα σημεία δειγματοληψίας.

6.4.3 Ανασκόπηση της μεθόδου BRISK

Η μέθοδος BRISK όπως θα δούμε και στην ενότητα 7, είναι μια μέθοδος που δεν θα έχει τα καλύτερα αποτελέσματα, αλλά καταλήγει σε μία από τις κορυφαίες μεθόδους. Η απόδοση της ανίχνευσης και περιγραφής των στοιχείων, η ικανότητα να παραμένει αναλλοίωτη στην αλλαγή κλίμακας και περιστροφής και ο χαμηλός χρόνος εκτέλεσης της, την καθιστούν μία αρκετά δυνατή μέθοδο.

6.4.4 Τροποποίηση της BRISK

Στο [42] παρουσιάζεται μια προσαρμοστική μετακίνηση για ένα αυτόνομο ρομπότ πολλαπλών άκρων βάδισης (multi-legged), μια μέθοδο πλήρωσης εικόνας για την ταξινόμηση του εδάφους, που βασίζεται σε ένα συνδυασμό επιταχυνόμενων ισχυρών χαρακτηριστικών και δυαδικών ισχυρών αναλλοίωτων κλιμακούμενων σημείων κλειδιών (SURF-BRISK). Η μέθοδος αυτή έχει το μέρος ανίχνευσης χαρακτηριστικών της SURF, δηλαδή το Hessian πίνακα, ενώ για περιγραφέα έχει το μοντέλο του περιγραφέα της BRISK.

Στο [43] παρουσιάστηκε η μέθοδος CBRISK, μια μετατροπή της BRISK η οποία ανιχνεύει χαρακτηριστικά σε χρωματιστές εικόνες. Με βάση τον περιγραφέα δυαδικής έντασης BRISK, η προτεινόμενη προσέγγιση ενσωματώνει τη δυαδική αναλλοίωτη χρωματική παρουσίαση στον αλγόριθμο περιγραφής CBRISK. Επιπλέον, με βάση τα αποτελέσματα που παρουσιάζονται, η τροποποίηση CBRISK είναι πιο διακριτική και πιο ισχυρή από την μέθοδο BRISK συγκρινόμενη με τη φωτομετρική μεταβολή.

Στο [44] παρουσιάζεται μια μετατροπή του BRISK που καταφέρνει να αξιοποιήσει την πληροφορία βάθους. Η ανίχνευση γίνεται με τον αλγόριθμο FAST, οι αποκρίσεις δίνονται από τον αλγόριθμο Harris και ο συντελεστής κλίμακας του σημείου κλειδιού, υπολογίστηκε απευθείας με την πληροφορία βάθους της εικόνας. Ακολουθεί ο υπολογισμός του κεντροειδούς έντασης του κύκλου με κέντρο το σημείο κλειδί και ο προσανατολισμός του σημείου κλειδί υπολογίστηκε με βάση τη μετατόπιση από το κεντροειδές της έντασής του. Στο τέλος, γίνεται σύγκριση των μεθόδων και τα αποτελέσματα δείχνουν πολύ καλύτερες αποδόσεις για την μετατροπή του BRISK.

Στο άρθρο [45] παρουσιάζεται μια μετατροπή την BRISK, η λεγόμενη speed-up BRISK (SBRISK). Η μέθοδος SBRISK υιοθετεί έναν σχεδόν κυκλικό συμμετρικό αστερισμό για να περιγράψει το μοτίβο των σημείων κλειδιών. Για να προσαρμοστεί στον χαρακτηριστικό προσανατολισμό του σημείου κλειδιού, το SBRISK μετατοπίζει το δυαδικό διάνυσμα αντί να περιστρέφει το μοτίβο ή τον αστερισμό της εικόνας, όπως έκαναν πολλοί άλλοι περιγραφείς. Εγκαταλείπει την παρεμβολή για να πάρει την ένταση σε θέση υπο-πίξελ, δεδομένου ότι ο αστερισμός δεν περιορίζεται αυστηρά σε κυκλική συμμετρία. Στα πειράματα που παρουσιάζονται στο άρθρο, φαίνεται πως η SBRISK έχει πιο γρήγορη και καλύτερη απόδοση κάνοντας λιγότερη κατανάλωση πόρων.

6.5 KAZE

Ο KAZE [46] είναι ένας αλγόριθμος ανίχνευσης και περιγραφής δισδιάστατων χαρακτηριστικών πολλαπλών κλιμάκων σε μη γραμμικούς χώρους κλίμακας. Ανιχνεύει 2D χαρακτηριστικά από την εικόνα εισόδου που παρουσιάζουν το μέγιστο του κανονικοποιημένου κατά κλίμακα προσδιοριστή της απόκρισης Hessian, μέσω του μη γραμμικού χώρου κλίμακας. Τέλος, υπολογίζει τον κύριο προσανατολισμό του σημείου κλειδιού και λαμβάνει έναν αμετάβλητο σε κλίμακα και περιστροφή περιγραφέα, λαμβάνοντας υπόψη τις παραγώγους πρώτης τάξης της εικόνας[46].

6.5.1 Μη γραμμικά ακρότατα κλίμακας-χώρου

Ο αλγόριθμος αυτός χρησιμοποιεί μη γραμμικό φιλτράρισμα [84] διάχυσης σε συνδυασμό με την τεχνική AOS [85]. Όπως και στην μέθοδο SIFT, υπάρχουν οκτάβες και υπό-επίπεδα. Ωστόσο, σε αντίθεση με την SIFT οι εικόνες σε κάθε οκτάβα έχουν το ίδιο μέγεθος με την αρχική εικόνα. Δεδομένης μιας εικόνας εισόδου, υπολογίζεται το ιστόγραμμα κλίσης της εικόνας με σκοπό να βρεθεί η παράμετρος της αντίθεσης. Με την παράμετρο της αντίθεσης και το σύνολο των χρόνων εξέλιξης t_i, κατασκευάζεται ο μη γραμμικός χώρος κλίμακας χρησιμοποιώντας τα σχήματα AOS [46].

6.5.2 Εντοπισμός σημείου κλειδιού

Για τον εντοπισμό σημείων ενδιαφέροντος, η KAZE υπολογίζει την απόκριση της κανονικοποιημένης κλίμακας προσδιορισμού του πίνακα Hessian σε πολλαπλά επίπεδα κλίμακας. Για την ανίχνευση χαρακτηριστικών πολλαπλών κλιμάκων, το σύνολο των διαφορικών τελεστών πρέπει να κανονικοποιηθεί ως προς την κλίμακα, δεδομένου ότι γενικά το πλάτος των χωρικών παραγώγων μειώνεται με την κλίμακα:

$$L_{Hessian} = \sigma^2(L_{xx}L_{yy} - L_{xy}^2)$$

όπου L_{xx} η οριζόντια παράγωγος, L_{yy} η κατακόρυφη παράγωγος και L_{xy}^2 η διασταυρούμενη παράγωγος σε κλίμακα σ [46].

6.5.3 Περιγραφή χαρακτηριστικών

Η KAZE χρησιμοποιεί μια προσαρμοσμένη έκδοση του περιγραφέα SURF, δεδομένου ότι οι περιγραφείς πρέπει να λειτουργούν σε ένα μη γραμμικό μοντέλο χώρου κλίμακας. Οι παραχθείσες αποκρίσεις υπολογίζονται και αθροίζονται σε ένα διάνυσμα περιγραφής σημείου-κλειδιού και στη συνέχεια το διάνυσμα κεντράρεται στο σημείο-κλειδί. Τέλος, ο περιγραφέας κανονικοποιείται σε ένα μοναδιαίο διάνυσμα. Δεδομένου ότι το KAZE υπολογίζει παραγώγους πολλαπλών κλιμάκων (κλίσεις) για κάθε εικονοστοιχείο, είναι πιο ακριβός ο υπολογισμός του από το SURF, αν και εξακολουθεί να είναι συγκρίσιμος με το SIFT, αλλά εξοικονομεί υπολογιστικές προσπάθειες περιγραφής του σημείου κλειδιού, επειδή το ίδιο σύνολο παραγώγων χρησιμοποιείται για την περιγραφή ενός σημείου κλειδιού [46].

6.5.4 Ανασκόπηση της μεθόδου KAZE

Η μέθοδος KAZE είναι μια τροποποίηση της SIFT και SURF που προσφέρει ακρίβεια και επαναληψιμότητα [46]. Η ταχύτητα της είναι καλύτερη από την μέθοδο SIFT αλλά όχι από την SURF. Ταυτόχρονα η υπολογιστική πολυπλοκότητα της είναι μεγαλύτερη της SURF. Ως αποτέλεσμα, η KAZE να είναι μια ενδιάμεση επιλογή μεταξύ SIFT και SURF.

6.5.5 Τροποποίηση της KAZE

Η μέθοδος του KAZE για τη χρήση μη γραμμικού φίλτρου διάχυσης απαιτεί την επίλυση μιας σειράς από μερικώς διαφορικές εξισώσεις. Αυτό δεν μπορεί να γίνει αναλυτικά και αναγκάζει το KAZE να χρησιμοποιήσει μια αριθμητική μέθοδο που ονομάζεται σχήμα AOS για την επίλυση των μερικώς διαφορικών εξισώσεων. Ωστόσο, αυτή η διαδικασία είναι υπολογιστικά δαπανηρή και για το λόγο αυτό δημιουργήθηκε μια επιταχυνόμενη έκδοση του KAZE. Αυτή η έκδοση ονομάζεται Accelerated KAZE ή AKAZE [47]. Ο αλγόριθμος accelerated KAZE λειτουργεί με τον ίδιο τρόπο όπως ο KAZE, αλλά σε αντίθεση με τον KAZE, χρησιμοποιεί μια ταχύτερη μέθοδο για τη δημιουργία του μη γραμμικού χώρου κλίμακας που ονομάζεται Fast Explicit Diffusion (FED). Εκτός από τη χρήση των FED, το AKAZE χρησιμοποιεί έναν δυαδικό περιγραφέα για περαιτέρω αύξηση της ταχύτητας. Για τη δημιουργία της περιγραφής σημείων κλειδιών, ο AKAZE κάνει χρήση μιας τροποποιημένης έκδοσης του δυαδικού περιγραφέα τοπικής διαφοράς (LDB). Ο LDB ακολουθεί τις ίδιες αρχές

με τον αλγόριθμο BRIEF, αλλά υπολογίζει δυαδικές δοκιμές στην περιοχή που περιβάλλει το σημείο κλειδί [47].

ΚΕΦΑΛΑΙΟ 7

Ανασκοπήσεις μεθόδων ανίχνευσης και περιγραφής χαρακτηριστικών

7.1 Άρθρα με ανασκοπήσεις των μεθόδων

Στο άρθρο [48] χρησιμοποιήθηκαν 96 RGB εικόνες χωραφιών, που λήφθηκαν από δύο διαφορετικές κάμερες. Αυτές οι εικόνες διαχωρίστηκαν στα RGB κανάλια, μπλε, κόκκινο και πράσινο. Οι εικόνες χωρίστηκαν σε $CIEL*a*b$ χρωματικό χώρο. Άρα, σε κάθε εικόνα αντιστοιχούν 6 διαφορετικές περιπτώσεις (RGB, $CIEL*a*b$). Εκτελέστηκε ανασκόπηση των εικόνων με τις ακόλουθες μεθόδους BRISK,FAST, SURF, SIFT, ORB, BRIEF, FREAK, KAZE, AKAZE. Η μέθοδος FAST συνδυάστηκε μαζί με την μέθοδο BRISK (καθώς η FAST δεν περιγράφει τα εικονοστοιχεία), έτσι ώστε τα εικονοστοιχεία που βρίσκει η FAST να περιγράφονται από την BRISK. Επιπλέον η FAST ανίχνευσε τα εικονοστοιχεία για την BRIEF και FREAK, καθώς αυτές οι μέθοδοι δεν ανιχνεύουν εικονοστοιχεία αλλά τα περιγράφουν. Έπειτα, χρησιμοποίησαν τους αλγόριθμους FLANN και BF για τον ταίριασμα των εικονοστοιχείων. Τα αποτελέσματα τους έδειξαν τις μεθόδους BRISK, FAST, SIFT και SURF να ανιχνεύουν τα περισσότερα εικονοστοιχεία, με την BRISK να ανιχνεύει τα περισσότερα από όλα τα κανάλια. Η μέθοδος ORB ανίχνευσε τα λιγότερα εικονοστοιχεία από όλες τις μεθόδους σε όλα τα κανάλια, εξαιρουμένης της KAZE στο κανάλι α*. Παρόλα αυτά, η ORB δεν είχε μεγάλες αλλαγές στον αριθμό εικονοστοιχείων που εντόπισε σε όλα τα κανάλια. Στην περίπτωση της περιγραφής εικονοστοιχείων, η μέθοδος FREAK είχε την καλύτερη απόδοση σε όλα τα κανάλια, με την BRIEF να είναι δεύτερη. Η ORB όμοια με την ανίχνευση εικονοστοιχείων, είχε την χαμηλότερη άλλα και σταθερότερη απόδοση σε όλα τα κανάλια. Σε θέμα χρόνου ανίχνευσης, η μέθοδος ORB ήταν η καλύτερη σε όλες τις περιπτώσεις ενώ ο KAZE και ο συνδυασμός FAST με FREAK είχαν την πιο αργή απόδοση. Στο ταίριασμα εικονοστοιχείων οι μέθοδοι FAST και SIFT είχα τα περισσότερα ταιριάσματα. Το συμπέρασμα μέσω των αποτελεσμάτων είναι ότι, η μέθοδος ORB ήταν η πιο γρήγορη από όλες άλλα είχε την χαμηλότερη απόδοση στην αντιστοίχιση

χαρακτηριστικών, ενώ παράλληλα, η SIFT και η FAST έδειξαν τις καλύτερες αντιστοιχίες χρησιμοποιώντας FLANN και BF, και η μέθοδος που παρουσίασε την υψηλότερη απόδοση τόσο σε χρόνο όσο και σε αριθμό σωστών αντιστοιχιών, ήταν ο συνδυασμός του ανιχνευτή χαρακτηριστικών FAST και του περιγραφέα BRISK.

Στο άρθρο [49] έγινε ανασκόπηση των μεθόδων SIFT, SURF, ORB, KAZE, BRISK. Σε αυτήν την ανασκόπηση, οι συγγραφείς αποσκοπούν να αξιολογήσουν την επαναληψιμότητα των εξεταζόμενων περιγραφικών σημείων κλειδιών. Για την επαναληψιμότητα εφαρμόστηκε χρήση τον αλγόριθμων για την εύρεση των σημείων κλειδιών και του αλγόριθμου Brute Force για την αντιστοίχηση τους. Η επαναληψιμότητα υπολογίστηκε χρησιμοποιώντας τις σωστές αντιστοιχίες σε συνδυασμό με το συνολικό ποσό των αντιστοιχιών [50]. Το συνολικό ποσό των σωστών αντιστοιχιών διαιρείται με το ελάχιστο ποσό των αρχικών αντιστοιχίσεων σε σχέση με τις δύο εικόνες. Όμοια για την ευρωστία, έγινε χρήση των σημείων κλειδιών που προέκυψαν και υπολογίστηκε η ομογραφία, η οποία περιέχει την ακριβή μετάφραση μεταξύ των δύο εικόνων. Για κάθε τύπο ακτινικής παραμόρφωσης (10%, 25% και 40%) δημιουργείται ένα boxplot. Κάθε boxplot περιλαμβάνει την επίπεδη σκηνή και τη σκηνή αντικειμένου σε σχέση με την ακτινική παραμόρφωση και τον αλγόριθμο. Σε γενικές γραμμές μπορεί να διαπιστωθεί ότι ανεξάρτητα από τον αλγόριθμο η επίπεδη σκηνή έχει υψηλότερη βαθμολογία σε σύγκριση με τη σκηνή αντικειμένου. Ωστόσο όταν αυξάνεται η ακτινική παραμόρφωση, η διαφορά αυτή εξακολουθεί να υπάρχει. αλλά λιγότερο εμφανής.Τα συμπεράσματα τους ήταν ότι η επαναληψιμότητα του SIFT και του KAZE ήταν αυτές με το μεγαλύτερο σκορ ενώ η ORB είχε το χαμηλότερο. Οι μέθοδοι SIFT και KAZE είναι όμοιοι σε θέμα πολυπλοκότητας, η μέθοδος SURF έχει χαμηλότερη πολυπλοκότητα αλλά δεν μπόρεσε να ξεπεράσει την ORB και την KAZE. Η ORB είχε χαμηλή αλλά σταθερή επαναληψιμότητα. Τέλος, συμπέραναν ότι η μέθοδος SURF ήταν καλύτερη από την BRISK και η BRISK καλύτερη από την ORB.

Στο άρθρο [51] έγινε ανασκόπηση των μεθόδων Harris, SIFT, SURF, GoodFeaturesToTrack, FAST, MSER, ORB στην ανίχνευση εικονοστοιχείων και στον χρόνο που χρειάστηκε η κάθε μέθοδος ώστε να τα εντοπίσει. Το πείραμα έγινε με την χρήση της OpenCV σε δύο σύνολα δεδομένων. Τα αποτελέσματα έδειξαν ότι η μέθοδος Harris και SIFT είχαν το μεγαλύτερο αριθμό χαρακτηριστικών, αλλά ταυτόχρονα χρειάστηκαν περισσότερο χρόνο. Από την άλλη, η μέθοδος ORB βρήκε

τα λιγότερα χαρακτηριστικά αλλά χρειάστηκε το ένα δέκατο του χρόνου της SIFT και HARRIS. Έπειτα, για τα δύο σύνολα δεδομένων έκανα το ταίριασμα των σημείων και ως αποτέλεσμα είχαν ότι, η SIFT χρειάστηκε τον περισσότερο χρόνο στο ταίριασμα ενώ η ORB είχε τον καλύτερο. Επιπλέον, ο αριθμός τον ζευγαριών που προέκυψαν για την μέθοδο SIFT και SUFT ήταν πολύ λιγότερα σε σχέση με όσα είχαν εντοπιστεί. Η μέθοδος ORB είχε και αυτή σημεία τα οποία ενώθηκαν άλλα πολύ λιγότερα σε σχέση με τις άλλες μεθόδους και σε σχέση με αυτά που βρήκε κατά την διάρκεια της αναζήτησης. Συμπέραναν στο τέλος ότι η ORB ήταν η καλύτερη μέθοδος καθώς έβρισκε λίγα αλλά σωστά σημεία, σε πολύ καλύτερο χρόνο από τις άλλες μεθόδους. Επιπλέον, η SIFT χρειάστηκε πάρα πολύ χρόνο γιατί έβρισκε σημεία τα οποία δεν ήταν χρήσιμα στο τέλος.

Στο άρθρο [52] έγιναν έλεγχοι με την χρήση της OpenCV 3.3 στην MATLAB-2017α για τις μεθόδους SUFT για περιγραφέα 64 και 128 Floats, SIFT, KAZE, AKAZE και τις μεθόδους ORB και BRISK δύο φορές, μία κανονικά και μία με όριο 1000 εικονοστοιχεία. Χρησιμοποιήθηκαν δύο σύνολα δεδομένων. Το σύνολο δεδομένων-Α χρησιμοποιήθηκε για τη σύγκριση διαφορετικών πτυχών των μεθόδων για εικόνες, ενώ το σύνολο δεδομένων-Β χρησιμοποιήθηκε για να διερευνηθεί η ικανότητα αναλλοίωτης κλίμακας και περιστροφής των μεθόδων. Επίσης χρησιμοποιήθηκαν τιμές βασικής αλήθειας για τους μετασχηματισμούς εικόνας για τον υπολογισμό και την επίδειξη του σφάλματος στα ανακτηθέντα αποτελέσματα με κάθε μέθοδο. Για την αξιολόγηση της αναλλοίωτης κλίμακας και περιστροφής, δημιουργήθηκαν συνθετικά βασικές αλήθειες για κάθε εικόνα στο σύνολο δεδομένων Β, αλλάζοντας το μέγεθος και περιστρέφοντας την σε γνωστές τιμές κλίμακας (5% έως 500%) και περιστροφής (0° έως 360°). Η τακτική για το ταίριασμα των στοιχείων ήταν αυτή που χρησιμοποιήθηκε από [30] και [10] με κατώφλι 0.7. Ακόμα, για την απόρριψη ακραίων τιμών και τον υπολογισμό ομογραφίας έγινε χρήση του RANSAC. Στο κομμάτι της ποσότητας τον στοιχείων που εντοπίστηκαν, βρέθηκε ότι η μέθοδος ORB είχε τα περισσότερα, με δεύτερη την BRISK. Η SURF ξεπέρασε την SIFT και η SURF(64D) ανίχνευσε περισσότερα στοιχεία από την SURF(128D). Η μέθοδος AKAZE ανίχνευσε περισσότερα στοιχεία από την KAZE. Τα στοιχεία που ανιχνεύτηκαν από την SIFT και SURF ήταν διάσπαρτα και η SURF είχε τα πυκνότερα αποτελέσματα από την SIFT. Τα χαρακτηριστικά που βρήκαν οι μέθοδοι ORB και BRISK είναι πιο συγκεντρωμένα. Στην ταχύτητα αναγνώρισης και

περιγραφής, η μέθοδος ORB είχε την καλύτερη απόδοση με δεύτερη την μέθοδο BRISK. Η AKAZE ήταν τρίτη σε χρόνο αλλά είχε μεγαλύτερο υπολογιστικό κόστος. Έπειτα, η KAZE κατατάχθηκε τέταρτη και ακολουθούμενη από την SURF(64D), SURF(128D) και τέλος της SIFT. Ακόμα, στην απόρριψη στοιχείων είναι όλοι οι μέθοδοι παρόμοιοι αλλά η ORB έχει την καλύτερη απόδοση χρόνου. Για την επαναληψιμότητα τα αποτελέσματα έδειξαν ότι, η ORB είχε την καλύτερη απόδοση και δεύτερη ήταν η BRISK. Ταυτόχρονα, οι μέθοδοι ORB και BRISK με το όριο των 1000 στοιχείων είχαν καλύτερη απόδοση από τις ίδιες μεθόδους χωρίς το όριο. Στο ταίριασμα σημείων η SIFT είχε τα καλύτερα αποτελέσματα με την BRISK δεύτερη. Τα αποτελέσματα των ORB και BRISK ήταν καλύτερα από τον ORB(1000) και BRISK(1000). Η AKAZE είναι πιο ακριβής από την KAZE. Ωστόσο, για τις μεταβολές περιστροφής ο KAZE είναι πιο ακριβέστερος από τον AKAZE. Το συμπέρασμα σε αυτό το πείραμα ήταν ότι η μέθοδος ORB είχε την καλύτερη απόδοση και σε θέμα ανίχνευσης, περιγραφής, αντιστοίχισης άλλα και σε θέμα χρόνου. Δεύτερη σε απόδοση ήταν η BRISK και τρίτη η AKAZE. Οι υπόλοιπες μέθοδοι είχαν διαφορετική απόδοση ανάλογα την περίπτωση.

Στο άρθρο [53]οι συγγραφείς έκανα πειράματα με της μεθόδους SIFT, KAZE, AKZE και ORB στην περιστροφική αμεταβλητότητα, αμεταβλητότητα φωτισμού, αμεταβλητότητα κλίμακας, γενικό έλεγχο ανίχνευσης και στον χρόνο εκτέλεσης. Κάθε μέθοδος ανίχνευσε χαρακτηριστικά σε εικόνες με σκοπό να βρει ένα αντικείμενο. Τα αποτελέσματα τους έδειξαν ότι η SIFT είχε την καλύτερη απόδοση σε όλους του ελέγχους εκτός του φωτισμού και του χρόνου εκτέλεσης. Η AKAZE είχε το καλύτερο αποτέλεσμα στην αμεταβλητότητα φωτισμού, ενώ η KAZE ήταν δεύτερη σε ποσοστό. Τέλος η ORB είχε τα χαμηλότερα ποσοστά ταιριάσματος σε όλες τις περιπτώσεις, αλλά είχε την καλύτερη απόδοση στον χρόνο εκτέλεσης.

Στο άρθρο [54] εφαρμόστηκαν έλεγχοι πάνω στην ανίχνευση στοιχείων, στην περιγραφή στοιχείων, τον χρόνο εκτέλεσης και σύγκριση στο ποσοστού σφάλματος για της μεθόδους SIFT, SURF, BRISK, ORB και τους αλγόριθμους FAST για ανίχνευση στοιχείων και FREAK για την περιγραφή. Τα αποτελέσματα έδειξαν ότι η μέθοδος SURF είχε την καλύτερη απόδοση στην ανίχνευση χαρακτηριστικών, ενώ η SIFT είχε την καλύτερη ανίχνευση σωστών σημείων αλλά ταυτόχρονα χρειάστηκε τον περισσότερο χρόνο.

Η έρευνα [55], είχε σκοπό να συγκρίνει τις μεθόδους SIFT, SURF, BRISK, ORB και AKAZE στην απόδοση της ανάλυσης τους σε εικόνες σεληνιακού εδάφους από ένα σεληνιακό όχημα. Τα πειράματα,έδειξαν ότι οι SIFT και AKAZE είχαν τα πιο ισχυρά αποτελέσματα. Η μέθοδος AKAZE ανίχνευσε μικρότερη ποσότητα χαρακτηριστικών σημείων από την SIFT, αλλά τα χαρακτηριστικά σημεία ανιχνεύονταν και αντιστοιχίζονταν με υψηλή ακρίβεια και το μικρότερο υπολογιστικό κόστος. Η AKAZE είναι επαρκής για γρήγορες και ακριβείς πληροφορίες πλοήγησης, αν και η SIFT έχει το υψηλότερο υπολογιστικό κόστος, η μεγαλύτερη ποσότητα χαρακτηριστικών σημείων ανιχνεύθηκε και τακτοποιήθηκε σταθερά. Το όχημα στέλνει περιοδικά εικόνες εδάφους στη Γη. Συνάμα, η SIFT είναι κατάλληλη για την κατασκευή παγκόσμιων τρισδιάστατων χαρτών εδάφους, δεδομένου ότι μπορεί να υποβληθεί σε επεξεργασία μεγάλη ποσότητα εικόνων εδάφους στη Γη.

Στο άρθρο [56]παρουσιάστηκε μία λεπτομερής ανάλυση των SIFT, SURF, ORB, BRISK και AKAZE για υποβρύχια περιβάλλοντα προς την εφαρμογή τους στο vSLAM. Σε αυτή την ανάλυση, οι ανιχνευτές που επιλέχθηκαν παρουσίασαν ικανοποιητική απόδοση σε εικόνες που περιέχουν χρωματική παραμόρφωση, χαμηλό ανομοιόμορφο φωτισμό και χαμηλή θολερότητα. Αμμώδη περιβάλλοντα με φύκια κηλίδες, άλγη που καταγράφονται από κοντά και μακριά, μικρά σωματίδια, όπως συντρίμμια και βράχοι, και αντικείμενα, όπως στύλοι και βράχοι παρουσίασαν ανιχνεύσιμα χαρακτηριστικά για τους ανιχνευτές. Η μέθοδος ORB ξεχώρισε στην ανίχνευση και απόδοση αντιστοίχισης, διαμορφώνοντας μια καλή επιλογή για την υλοποίηση του vSLAM, με το χαμηλότερο χρόνο υπολογισμού.

το άρθρο [57] έγινε ερευνά μεταξύ διαφορών αλγορίθμων ανίχνευσης και περιγραφής στοιχείων, καθώς και συνδυασμός αυτών τών μεθόδων μεταξύ τους, για παράδειγμα ο ανιχνευτής SURF και ο περιγραφέας FREAK. Τα πειράματα έγιναν πάνω σε διάφορα σύνολα δεδομένων και οι μέθοδοι συγκρίθηκαν σύμφωνα με τα παρακάτω κριτήρια:

- The Putative Match Ratio
- Precision
- Matching Score
- Reliability
- Mean Execution Time

Η ερευνά ήταν αρκετά μεγάλη λόγω των πολλών συνδυασμών αλγορίθμων και κριτηρίων. Τα κύρια συμπεράσματα που ανέφεραν οι συγγραφείς, ήταν πως ο συνδυασμός του ανιχνευτή FAST με τον περιγραφέα SIFT είχε αρκετά καλά αποτελέσματα, ο αλγόριθμος BRIEF είχε επίσης καλά αποτελέσματα στους συνδυασμούς του και ο ORB έδειξε αξιόπιστη απόδοση στα περισσότερα πειράματα.

Στο άρθρο [58] πραγματοποιήθηκε συγκριτική εκτίμηση της ποιότητας και του χρονικού κόστους των περιγραφέων πλήρους κύκλου SIFT, SURF128, SURF64, BRISK, ORB, ORB (1000), KAZE, AKAZE. Οι συγγραφείς, συνιστούν τη χρήση του περιγραφέα SIFT για γενικές εργασίες όπου είναι απαραίτητη η επεξεργασία εικόνων με σκηνές για καλύτερα αποτελέσματα. Παρόλα αυτά, η SIFT είναι αρκετά απαιτητική σε θέμα χρόνου. Όποτε, στην περίπτωση που χρειάζεται μια μέθοδος που έχει γρήγορα αποτελέσματα και η ποιότητα μπορεί να παραμεληθεί εντός λογικών ορίων, είναι προτιμότερο να χρησιμοποιούνται δυαδικοί αλγόριθμοι περιγραφής ORB1000 ή AKAZE.

7.2 Εξέταση και συμπεράσματα

Από τις παραπάνω έρευνες μπορούμε να συμπεράνουμε ότι δεν υπάρχει μια απόλυτη μέθοδος ανίχνευσης/περιγραφής στοιχείων για την υλοποίηση πανοραμικών εικόνων. Αρκετές ανασκοπήσεις κατέληξαν στον συμπέρασμα ότι η μέθοδος SIFT, έχει η πιο ακριβής μέθοδος στην αναγνώριση και περιγραφή στοιχείων. Παρόλα αυτά, ο χρόνος που απαιτεί είναι περισσότερος από άλλες μεθόδους. Στην απόδοση του χρόνου, πολλές ανασκοπήσεις έβγαλαν την ORB ως την καλύτερη μέθοδο, αλλά πολύ συχνά η σταθερότητας της δεν παρέμενε όσο καλή όσο οι άλλες. Οι μέθοδοι BRISK, SURF ήταν ενδιάμεσα, με την SURF να θεωρείται μια πιο γρήγορη SIFT και την BRISK μια πιο σταθερή ORB. Η AKAZE σε όλες τις έρευνες έδειχνε καλύτερα αποτελέσματα από την KAZE. Έτσι, δεν μπορεί να θεωρηθεί μία μόνο μέθοδος ως η καλύτερη, αλλά μπορεί να θεωρηθεί καλύτερη ή αρκετά αποδοτική σε κάποιες περιπτώσεις. Η SIFT είναι κατάλληλη για περιπτώσεις που χρειάζεται μεγάλη ακρίβεια αλλά δεν είναι απαραίτητο τα αποτελέσματα να δημιουργούνται σε γρήγορο χρόνο. Ενώ η ORB βρίσκει γρήγορα αποτελέσματα από τις υπόλοιπες μεθόδους, ωστόσο τα αποτελέσματα της ORB έχουν λιγότερη ακρίβεια και σταθερότητα.

Ένα συγκεκριμένο πρόβλημα που είχαν πολλές ανασκοπήσεις, ήταν ότι δεν συνδύαζαν διάφορους αλγόριθμους μεταξύ τους για να δουν αν υπάρχουν ενδιαφέροντα αποτελέσματα. Με τον συνδυασμό διαφορετικών αλγορίθμων ανίχνευσης και περιγραφής, υπάρχουν διαφορετικά αποτελέσματα τα οποία ίσως να έχουν καλύτερη απόδοση από τις γνώστες μεθόδους. Οπότε θα ήταν ενδιαφέρον σε μελλοντικές ανασκοπήσεις να γίνεται συνδυασμός των αλγόριθμων ανίχνευσης και περιγραφής.

ΚΕΦΑΛΑΙΟ 8

Ευθυγράμμιση και Συρραφή

Αφού γίνει η ανίχνευση και η περιγραφή τον χαρακτηριστικών, ακολουθεί η διαδικασία του ταιριάσματος των στοιχείων για την ενοποίηση των εικόνων. Η σύγκριση όλων των χαρακτηριστικών της μιας εικόνας με όλα τα χαρακτηριστικά της άλλης, χρησιμοποιώντας έναν από τους τοπικούς αλγόριθμους περιγραφής που αναφέρθηκαν παραπάνω, είναι η απλούστερη προσέγγιση για τον εντοπισμό όλων των σχετικών σημείων χαρακτηριστικών σε ένα ζεύγος εικόνων. Η πολυπλοκότητα της τεχνικής αυτής είναι $O(n_2)$, κάτι που για πολλές εφαρμογές δεν είναι κατάλληλο. Για τον λόγο αυτό, χρησιμοποιούνται αλγόριθμοι αντιστοίχισης που αντιστοιχούν τα χαρακτηριστικά των εικόνων με βάση κάποια κριτήρια.

8.1 Αλγόριθμοι αντιστοίχησης

8.1.1 Brute Force Matcher

Ο Brute-Force matcher παίρνε την περιγραφή ενός χαρακτηριστικού στο τις πρώτης εικόνας και το ταιριάζει με κάθε άλλο χαρακτηριστικό στο δεύτερο σύνολο. Το χαρακτηριστικό με την κοντινότερη απόσταση, επιστρέφεται από τον αλγόριθμο BF. Ο BF χρησιμοποιεί την ευκλείδεια ή την απόσταση Hamming, ως μέτρο για τον υπολογισμό της απόστασης των χαρακτηριστικών[59].

Εικ 8.1: Αντιστοίχιση σημείων κλειδιών με τον αλγόριθμο Brute Force

8.1.2 KNN matcher

Ο KNN matcher λειτουργεί ακριβώς όπως ο BF αλλά, αντί να επιστρέφει το χαρακτηριστικό με την μικρότερη απόσταση, επιστρέφει της k καλύτερες αντιστοιχίες, όπου το k καθορίζεται από το χρήστη [59].

8.1.3 FLANN matcher

Ο αλγόριθμος FLANN κάνει χρήση του τυχαίου δέντρου KD και δέντρου K-means. Με την ταυτόχρονη εξερεύνηση πολυάριθμων δέντρων, ο τυχαίος KD έχει καλή ικανότητα να εντοπίζει τα σημεία σε οποιοδήποτε δέντρο KD που είναι πλησιέστερα σε ένα σημείο εισόδου. Μπορεί να εξαλειφθεί γρήγορα ένα τμήμα του χώρου αναζήτησης, χρησιμοποιώντας τις ιδιότητες του δέντρου αναζήτησης. Τα τμήματα των δεδομένων διαχωρίζονται χρησιμοποιώντας τον αλγόριθμο του δέντρου K-means, ο οποίος στη συνέχεια αναδιοργανώνει κάθε περιοχή, έως ότου κάθε κόμβος φύλλου περιέχει περισσότερα από *M* στοιχεία [60].

8.2 Ευθυγράμμισης εικόνων

Αφού γίνει η αντιστοίχηση των χαρακτηριστικών δυο εικόνων, θα πρέπει να επιλεχθεί η σωστή ευθυγράμμιση των εικόνων με βάση αυτά. Η σωστή ευθυγράμμιση θα είναι αυτή που η πλειοψηφία των στοιχείων είναι παράλληλη. Υπάρχουν δυο αρκετά διαδεδομένοι μέθοδοι για την επιλογή της ευθυγράμμισης.

8.2.1 RANdom SAmple Consensus

Ο RANdom SAmple Consensus ή RANSAC [61] [62] είναι ένας αλγόριθμος προσέγγισης και εκτίμησης παραμέτρων που έχει σχεδιαστεί για να αντιμετωπίζει ένα μεγάλο ποσοστό λανθασμένων τιμών στα δεδομένα εισόδου. Ο αλγόριθμος αρχικά επιλέγει ένα τυχαίο υποσύνολο k αντιστοιχιών, το οποίο στη συνέχεια χρησιμοποιεί για να υπολογίσει μία εκτίμηση κίνησης u.

$$1-p = (1-u^k)^n$$

Όπου *n* ο αριθμός των επαναλήψεων, u η πιθανότητα κάθε επιλεγμένο σημείο δεδομένων να είναι ακραίο, $v = 1 - u$ η πιθανότητα παρατήρησης ενός ακραίου σημείου και p η πιθανότητα.

Ένα πλεονέκτημα του RANSAC είναι η ικανότητά του να κάνει ισχυρή εκτίμηση ακόμη και με σημαντικό αριθμό ακραίων τιμών που υπάρχουν στα δεδομένα. Η έξοδος του RANSAC αποκτά σχετικά υψηλή ακρίβεια. Ωστόσο λόγω της τυχαίας φύσης του RANSAC, δεν είναι πάντα βέβαιο ότι τα αποτέλεσμα θα είναι σωστά σε κάθε εκτέλεση. Ένα άλλο μειονέκτημα είναι πως ο χρόνος υπολογισμού του RANSAC είναι αρκετά υψηλός [63].

8.2.2 Least Median of Squares

Η Least Median of Squares η LMeS [64] μέθοδος εκτιμά τις παραμέτρους επιλύοντας το μη γραμμικό πρόβλημα ελαχιστοποίησης. Η LMS λειτουργεί όπως η RANSAC με τη διαφορά ότι η βαθμολόγηση είναι με βάση την διάμεση απόσταση των δεδομένων [65].

8.3 Ομογραφία και στρέβλωση εικόνων

Μετά την αντιστοίχιση των ζευγών μεταξύ δύο ή περισσοτέρων εικόνων, η σχέση των ζευγών αντιστοίχισης αποθηκεύεται ως πίνακας ομογραφίας (homography matrix). Έπειτα, κάθε εικόνα μετατοπίζεται τρισδιάστατα και προβάλλεται δισδιάστατα. Η διαδικασία μετάφρασης και προβολής των εικόνων, λέγεται στρέβλωση εικόνας ή μετατόπιση εικόνας, όπου κάθε εικόνα θα στρεβλωθεί, έτσι ώστε να μπορέσει να ενωθεί με τις άλλες γύρω της. Οι εικόνες μετατοπίζονται κατάλληλα, έτσι ώστε τα κοινά τους σημεία να εφάπτονται μεταξύ τους για να παρουσιαστούν σαν μία ενιαία εικόνα.

Εικ 8.2: Εικόνα πριν την στρέβλωση

Εικ 8.3: Εικόνα μετά την στρέβλωση

8.4 Ανάμειξη και σύνθεση

Με τις ραφές των εικόνων να έχουν προσδιοριστεί, το επόμενο βήμα στην συρραφή είναι η ανάμειξη των εικόνων. Στην ανάμειξη οι δύο εικόνες θα πρέπει να συρραφτούν με τέτοιο τρόπο που να υπάρχει μια ισορροπία στα χρώματα τους χωρίς απότομες μεταβάσεις. Όπως φαίνεται στην Εικ 0.1 η πανοραμική που δημιουργήθηκε έχει μεγάλες διαφορές χρώματος και φωτεινότητας στα σημεία που οι εικόνες ενώθηκαν. Για την αντιμετώπιση αυτού του προβλήματος έχουν δημιουργηθεί τεχνικές ανάμειξης εικόνων.

Εικ 8.4: Συρραφή εικόνας χωρίς blend

8.4.1 Laplacian pyramid blending

Η τεχνική ανάμειξης με Πυραμίδα Laplacian που αναπτύχθηκε από τους [66] διασπά πρώτα τις εικόνες σε ένα σύνολο συστατικών εικόνων που έχουν φιλτραριστεί με Laplacian φίλτρο. Στη συνέχεια, οι συστατικές εικόνες σε κάθε ζώνη χωρικών συχνοτήτων συναρμολογούνται σε ένα αντίστοιχο μωσαϊκό Laplacian φίλτρου. Οι συστατικές εικόνες ενώνονται χρησιμοποιώντας έναν σταθμισμένο μέσο όρο, εντός μιας μεταβατικής ζώνης, που είναι ανάλογη σε μέγεθος με τα μήκη κύματος που αντιπροσωπεύονται στη ζώνη. Όταν τα χονδροειδή χαρακτηριστικά εμφανίζονται κοντά στα σύνορα, αυτά αναμειγνύονται σταδιακά σε σχετικά μεγάλη απόσταση χωρίς να θολώνουν ή να υποβαθμίζονται με άλλο τρόπο οι λεπτομέρειες της εικόνας στη γειτονιά του συνόρου [67][68].

Εικ 8.5: Συρραφή εικόνας με blend

8.4.2 Gradient domain blending

Μια εναλλακτική προσέγγιση για την ανάμειξη εικόνων πολλαπλών ζωνών είναι η εκτέλεση των λειτουργιών στο πεδίο της κλίσης gradient domain. Σε αυτήν την

τεχνική η ανάμειξη εικόνων γίνεται με την δημιουργία ενός νέου διανυσματικού πεδίου κλίσης αντιγράφοντας τις τιμές κλίσης από τις εικόνες που το αποτελούν. Οι κλίσεις στις ραφές μηδενίζονται για την εξομάλυνση των χρωματικών διαφορών. Μια νέα σύνθετη εικόνα μπορεί να ανακτηθεί από το διανυσματικό πεδίο κλίσης επιλύοντας μια εξίσωση Poisson με οριακές συνθήκες. Η σύνθετη εικόνα είναι μια συνολικά βέλτιστη λύση, όπου η μετάβαση χρώματος εξομαλύνεται σε ολόκληρη την εικόνα [69][70].

8.4.3 Exposure compensation

Η συγκεκριμένη τεχνική αποτελεί μια εναλλακτική των παραπάνω για περιπτώσεις που οι διάφορες στις εικόνες είναι αρκετά μεγάλες [71]. Για να εκτιμηθεί μια τοπική διόρθωση μεταξύ κάθε εικόνας για την δημιουργία ενός σύνθετου συνδυασμού, προσαρμόζεται πρώτα μια τετραγωνική συνάρτηση μεταφοράς, με βάση το μπλοκ μεταξύ κάθε εικόνας πηγής και ενός αρχικού σύνθετου συνδυασμού με εξομάλυνση. Για την επίτευξη ομαλότερης απεικόνισης, οι συναρτήσεις μεταφοράς υπολογίζονται στη συνέχεια κατά μέσο όρο με τις γειτονικές τους. Οι συναρτήσεις μεταφοράς ανά εικονοστοιχείο υπολογίζονται στη συνέχεια με συρραφή μεταξύ γειτονικών τιμών μπλοκ [72][73].

8.4.4 Generative Adversarial Network

Μια αρκετά διαφορετική τεχνική για την ανάμειξη εικόνων είναι η χρήση των Γενετικών αντιθετικών δικτύων (GANs) [74][75][86]. Τα GAN με τη χρήση πραγματικών στατιστικών στοιχείων εικόνας και μαθημένων σημασιολογικών πληροφοριών, επιδιώκουν να συμπληρώσουν τυχόν κενά εικονοστοιχεία σε μια εικόνα. Τα GANs χρησιμοποιούνται για σύνθεση εικόνων μεγάλης ποιότητας, για το γέμισμα κενών με την τεχνική Gaussian-Poisson Generative Adversarial Network (GP-GAN) [76], αποτελώντας μία τροποποίηση της τεχνικής GAN που αξιοποιεί τα πλεονεκτήματα της κλασικής προσέγγισης με Gradient domain. Παρόλα αυτά, τα GANs μοντέλα δεν είναι αρκετά ακριβή στα αποτελέσματα τους και δημιουργούν μη ρεαλιστικά όρια και φωτισμό [70].

ΚΕΦΑΛΑΙΟ 9

Υλοποίηση της μεθόδου ORB με την γλώσσα Python

ΣΣε αυτό το κεφάλαιο παρουσιάζεται η υλοποίηση της μεθόδου ORB στην γλώσσα προγραμματισμού Python. Έγινε υλοποίηση του αλγόριθμου FAST για την εύρεση σημείων κλειδιών και υλοποίηση του αλγόριθμου Harris για το score των σημείων. Υλοποιήθηκε η μέθοδος εύρεσης γωνίας με βάση την κεντροειδή ένταση. Υλοποιήθηκε ο rBRIEF για την περιγραφή του κάθε χαρακτηριστικού. Τέλος, γίνεται ταίριασμα των σημείων κλειδιών μεταξύ δύο εικόνων και δημιουργείται η ομογραφία για την ένωση δύο εικόνων.

Βιβλιοθήκες

Για την υλοποίηση αυτού του προγράμματος έγινε χρήση βιβλιοθηκών της Python. Αρχικά έγινε χρήση της βιβλιοθήκης NumPy. Η NumPy είναι μια βιβλιοθήκη που παρέχει υποστήριξη για μεγάλους πολυδιάστατους πίνακες και συμβολοσειρές, μαζί με μια μεγάλη συλλογή μαθηματικών συναρτήσεων υψηλού επιπέδου για τη λειτουργία αυτών των πινάκων. Η Numpy βοηθά στην υλοποίηση της μεθόδου επεξεργασίας εικόνας, λόγω της ικανότητας της να μετατρέπει εικόνες σε πολυδιάστατους πίνακες, με αποτέλεσμα η επεξεργασία εικόνων να γίνεται εύκολη. Έπειτα έγινε η χρήση των βιβλιοθηκών OpenCV. Η OpenCV είναι μια συλλογή μεθόδων επεξεργασίας εικόνων και επεξεργασίας πολυδιάστατων πινάκων. Είναι γραμμένη σε C++ και έχει μεγάλες αποδόσεις με μικρή κατανάλωση πόρων. Η OpenCV χρησιμοποιήθηκε για την αναπαράσταση, καταχώριση εικόνων, για τις συναρτήσεις αντιστοίχισης στοιχείων όπως και η Brute Force για την συνάρτηση στρέβλωσης εικόνας. Ακόμα, έγινε η χρήση βιβλιοθηκών της Python concurrent.futures για την διαμερισμό των εικόνων σε πολλούς πυρήνες με σκοπό την μείωση του χρόνου επεξεργασίας των εικόνων. Τέλος, έγινε η χρήση της βιβλιοθήκης os της Python για την αναζήτηση αρχείων σε υπολογιστή.

Εκτέλεση του Προγράμματος

Αρχικά το πρόγραμμα διαβάζει δυο εικόνες που του δίνει ο χρήστης. Οι εικόνες αυτές γίνονται ασπρόμαυρες (grayscale) και μπαίνουν σε μία λίστα. Το πρόγραμμα δημιουργεί για κάθε εικόνα 4 Γκαουσιανά επίπεδα, δηλαδή μία Γκαουσιανή πυραμίδα τεσσάρων επιπέδων. Από τα τέσσερα αυτά επίπεδα, το πρώτο επίπεδο είναι η αρχική εικόνα. Κάθε ένα από τα υπόλοιπα επίπεδα είναι το μισό σε μέγεθος του προηγούμενου επίπεδου του. Τα 3 επίπεδα που παράγονται, αυξάνονται στο μέγεθος της αρχικής εικόνας με θόλωση gauss. Αυτό γίνεται έτσι ώστε η μέθοδος να είναι αναλλοίωτη στην κλίμακα. Η ανίχνευση των στοιχείων θα γίνει σε όλα τα επίπεδα. Στόχος είναι να βρεθούν τα καλύτερα σημεία κλειδιά σε όλα τα επίπεδα της πυραμίδας.

Αλγόριθμος FAST

Αφού δημιουργηθούν οι πυραμίδες για τις εικόνες, θα περάσουν από τον αλγόριθμο FAST. Για την όλες τις εικόνες θα γίνει ανίχνευση σημείων, τα οποία τηρούν τις προϋποθέσεις:

$$S_{p\to x}=\begin{cases} d, I_{p\to x}\le I_p - t\ (\sigma\kappa o\tau\varepsilon\iota\nu\acute{o}\tau\varepsilon\rho o) \\ b, I_p + t \le I_{p\to x}\ (\varphi\omega\tau\varepsilon\iota\nu\acute{o}\tau\varepsilon\rho o) \end{cases}$$

Η ακτίνα FAST και το όριο των σημείων που πρέπει να τηρούν τις προϋποθέσεις για να είναι ένα εικονοστοιχείο σημείο κλειδί είναι:

Επίπεδο πυραμίδας	Ακτίνα	Όριο
1	3	9
2	7	18
3	16	32
4	28	64

Πιν 9.1: Τροποποιήσεις του FAST

Αυτό γίνεται με σκοπό να βρεθούν χαρακτηριστικά σε όλα τα επίπεδα και να μπορούν αυτά τα χαρακτηριστικά να αντιστοιχιστούν με τα ίδια χαρακτηριστικά κάθε

επιπέδου τις ίδιας πυραμίδας. Έτσι κάθε χαρακτηριστικό που διαλέγεται από τον αλγόριθμο θα παραμένει αναλλοίωτο στην αλλαγή κλίμακας.

Εικόνες που χρησιμοποιήθηκαν

Αλγόριθμος Harris

Αφού βρεθούν τα χαρακτηριστικά, θα εξεταστούν από τον αλγόριθμο Harris. Ο Harris θα δώσει σε κάθε σημείο κλειδί που βρήκε ο FAST ένα σκορ. Με αυτό το σκορ θα ταξινομηθούν όλα τα στοιχεία του κάθε επιπέδου τις πυραμίδας. Αφού γίνει η ταξινόμηση, θα επιλεχθούν τα καλύτερα από αυτά για να συνεχίσει η εξέταση τους.

Γωνία σημείου

Για κάθε στοιχείο του κάθε επιπέδου που παρέμεινε στην λίστα, θα υπολογιστεί η γωνία του. Ο υπολογισμός του προσανατολισμού του κάθε στοιχείου θα γίνει με βάση την κεντροειδή ένταση του. Ο τύπος για τον υπολογισμό είναι ο ακόλουθος:

$$\theta = atan2(m_{01}, m_{10})$$

Επιλογή των καλύτερων σημείων κλειδιών

Αφού βρεθεί ο προσανατολισμός του κάθε στοιχείου, τα σημεία κλειδιά του κάθε επιπέδου μπαίνουν σε μία κοινή λίστα. Η μορφή τους θα αλλάξει σε cv2.Keypoint αντικείμενα. Το cv2.Keypoint αντικείμενο είναι της OpenCV και θα χρησιμοποιηθεί για την αναπαράσταση των εικονοστοιχείων στις εικόνες. Μετά κάθε εικονοστοιχείο στην λίστα αντιστοιχείται με τα υπόλοιπα εικονοστοιχεία που βρίσκονται στα τέσσερα επίπεδα της Gaussian πυραμίδα της εικόνας. Αφού βρεθούν όλα τα εικονοστοιχεία τα οποία μένουν αναλλοίωτα στην κλίμακα, δηλαδή βρεθούν και στα τέσσερα επίπεδα, τότε μένουν στην λίστα. Αυτά που δεν βρέθηκαν και στα

τέσσερα επίπεδα δεν παραμένουν στην λίστα. Έπειτα, με βάση τα σκορ που τους δίνεται από το αλγόριθμο Harris, επιλέγονται τα 300 καλύτερα.

Αποτελέσματα καλύτερων σημείων κλειδιών

Περιγραφή σημείων κλειδιών

Τα 300 σημεία που μένουν περιγράφονται από τον αλγόριθμο rBRIEF. Ο rBRIEF χρησιμοποιεί μία λίστα με ζευγάρια τα οποία θα γίνει ο έλεγχος

$$\tau\,(p;x,y) = \begin{cases} 1 : p\,(x) < p(y) \\ 0 : p(x) \geq p(y) \end{cases}$$

Το αποτέλεσμα είναι μία συμβολοσειρά 256 bits. Αυτή η συμβολοσειρά μετατρέπεται σε συμβολοσειρά 32 byte.

Ένωση εικόνων

Οι παραπάνω διαδικασίες γίνονται και για τις δύο εικόνες. Αφού περιγραφούν όλα τα χαρακτηριστικά γίνεται η αντιστοίχιση τους με τον αλγόριθμο Brute Force. Τα πιο δυνατά ζευγάρια επιλέγονται για να δημιουργηθεί η ομογραφία για την δεύτερη εικόνα.

Αφού δημιουργηθεί η ομογραφία, στρεβλώνεται η δεύτερη εικόνα σε σχέση με την πρώτη και συνδυάζεται με αυτήν.

Αποτελέσματα του προγράμματος

ΚΕΦΑΛΑΙΟ 10

Συμπεράσματα

Η δημιουργία πανοραμικών εικόνων αποτελεί έναν κλάδο στην όραση υπολογιστών για αρκετά χρόνια και ακόμα εξελίσσεται με νέες μεθόδους και τεχνικές. Συγκεκριμένα υπάρχουν πάρα πολλές μέθοδοι για την υλοποίηση πανοραμικών εικόνων, που συνδυάζουν διάφορους αλγόριθμους για να δημιουργήσουν μια ενιαία εικόνα από άλλες εικόνες μικρότερου μήκους. Στην συγκεκριμένη διπλωματική έγινε μία ανασκόπηση διαφόρων μεθόδων δημιουργίας πανοραμικών εικόνων, καθώς και τα βήματα που χρειάζονται για την υλοποίηση τους. Οι μέθοδοι που αναλύθηκαν βασίζονται στα χαρακτηριστικά εικόνων επειδή είναι συνηθέστερα χρησιμοποιούμενες και έχουν χαμηλότερο υπολογιστικό κόστος. Αρχικά έγινε αναφορά στις μεθόδους ανίχνευσης χαρακτηριστικών γιατί αποτελούν το πρώτο σκέλος για τις μεθόδους που βασίζονται στα χαρακτηριστικά. Έπειτα, έγινε αναφορά σε αλγόριθμους περιγραφής των χαρακτηριστικών. Ακολούθως, έγινε εκτεταμένη αναφορά στις μεθόδους που βασίζονται στα χαρακτηριστικά. Παρουσιάστηκαν οι αρχιτεκτονικές τους στο κομμάτι της ανίχνευσης, στον υπολογισμό της γωνίας άλλα και στην περιγραφή των σημείων κλειδιών για την κάθε μέθοδο. Για κάθε μέθοδο που αναφέρθηκε, παρουσιάστηκαν ορισμένες τροποποιήσεις τους. Κατόπιν, παρουσιάστηκαν διάφορες ανασκοπήσεις των μεθόδων αυτών, που ελέγχουν την απόδοση τους στην ανίχνευση και περιγραφή σημείων κλειδιών, στον χρόνο ολοκλήρωσης τους, στη επαναληψιμότητα τους και στην ανάμειξη τους με άλλους αλγόριθμους. Από τις ανασκοπήσεις καταλήξαμε στο ότι δεν υπάρχει μια απόλυτη τεχνική που να λειτουργεί σε όλες τις περιπτώσεις καλύτερα από τις άλλες. Από τις μεθόδους επιλέχθηκε η ORB για να υλοποιηθεί με την γλώσσα προγραμματισμού Python. Μετέπειτα, έγινε αναφορά σε αλγόριθμους αντιστοίχισης στοιχείων ανάμεσα σε δύο εικόνες. Τέλος, παρουσιάστηκαν μέθοδοι που χρησιμοποιούνται κατά την διαρκεί ενοποίησης εικόνων σε μία πανοραμική, με σκοπό να υπάρχει μια ενιαία εικόνα που να έχει ρεαλιστική απεικόνιση του περιβάλλοντος των εικόνων που ενώθηκαν.

ΠΑΡΑΡΤΗΤΜΑ Α’

```
def sortScore(val):
    return val[2]
all_keypoints = []

def keypoint_details_processing(pyramid_details):
    octave = pyramid_details[0]
    threshold = pyramid_details[1]
    image = pyramid_details[2]

    keypoints_of_image = fast_detect(image, threshold, octave)
    keypoints_of_image                                          =
find_harris_corners(image,1000.0,keypoints_of_image)

    keypoints_of_image = corner_orientations(image,keypoints_of_image)
    print("Finished keypoint detection")
    return keypoints_of_image

if __name__ == '__main__':
    os.system('clear')
    path = "images/"
    DOWNSCALE = 2
    N_LAYERS = 4
    threshold = 20
    images = []
    pattern = generate_pattern()
    names = []
    all_descriptors = []
    all_keypoints = []

    for i in os.listdir(path_pada):
        print("Processing image {}".format(i))
        names.append(i.strip(".jpg"))
        img = cv2.imread(image_path)
        images.append(img)
    keypoints_and_octaves = []
    name = 0

    for image in images:
        image_keypoints = ()
        gaussian_pyramid = []
        gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
        gaussian_pyramid.append([0,threshold,gray])
        layer = gray
```

```
        for i in range(1, N_LAYERS):
            downscale = gray
            for j in range(i):
                downscale = cv2.pyrDown(downscale)
            for j in range(i):
                downscale = cv2.pyrUp(downscale)

            gaussian_pyramid.append([i, threshold,downscale])

        with concurrent.futures.ProcessPoolExecutor() as executor:
            results = [executor.submit(keypoint_details_processing, gp)
for gp in gaussian_pyramid]
            for future in concurrent.futures.as_completed(results):


                for kp in future.result():

                    image_keypoints = image_keypoints + (cv2.KeyPoint(
                        x = float(kp.x),
                        y = float(kp.y),
                        size = 7 * (2**kp.octave),
                        angle = kp.orientation,
                        response = kp.score,
                        octave = kp.octave,
                        class_id = -1),)

        image_keypoints = topN_kepoints(image_keypoints, 300)
        image_with_keypoints          =          cv2.drawKeypoints(image,
image_keypoints, None,flags=cv2.DRAW_MATCHES_FLAGS_DRAW_RICH_KEYPOINTS)
        cv2.imwrite("keypoints/"   +   names[name]   +   "_keypoints.jpg",
image_with_keypoints)

        all_keypoints.append(image_keypoints)
        all_descriptors.append(brief_descriptor_function(gray,
image_keypoints, pattern = pattern))

        print (name)

        if name % 2 != 0:
            good_matches               =               match(images[name-
1],images[name],all_keypoints[0],all_keypoints[1],all_descriptors[0],al
l_descriptors[1], name, names)
            M          =          homography_stitching(all_keypoints[0],
all_keypoints[1], good_matches, reprojThresh=4)

            if M is None:
                print("Error!")
```

```
            else:
                (matches, Homography_Matrix, mask) = M

                print(Homography_Matrix)

                width = images[name].shape[1] + images[name-1].shape[1]

                # h, w = images[name-1].shape[:2]
                # print(h, w)

                height    =    max(images[name].shape[0],    images[name-
1].shape[0])

                result        =        cv2.warpPerspective(images[name],
Homography_Matrix,  (width, height))
                #save the transformed image
                cv2.imwrite("transformed/"      +      names[name]      +
"_transformed.jpg", result)

                i1x , i1y = images[name-1].shape[:2]
                i2x , i2y = result.shape[:2]
                for i in range(0, i1x):
                    for j in range(0, i1y):
                        try:
                            if(np.array_equal(images[name-
1][j,i],np.array([0,0,0])) and  \

np.array_equal(result[j,i],np.array([0,0,0]))):
                                result[j,i] = [0, 0, 0]
                            else:

if(np.array_equal(result[j,i],[0,0,0])):
                                    result[j,i] = images[name-1][j,i]
                                else:
                                    if  not  np.array_equal(images[name-
1][j,i], [0,0,0]):

                                        bl,gl,rl = images[name-1][j,i]
                                        result[j, i] = [bl,gl,rl]
                        except:
                            pass

                cv2.imwrite("stitched/"      +      names[name]      +
"_stitched_images.jpg", result)

                crop_result = trim(result)
                #  cv2.imshow(  names[name]  +  "cropped  image  mix",
```

```
crop_result)
                cv2.imwrite("results/" + names[name] + "_finished.jpg",
crop_result)
                #cv2.waitKey()
                name = name + 1
                image_keypoints = ()
                gaussian_pyramid = []
                keypoints_and_octaves = []
                all_keypoints = []
                all_descriptors = []
        elif name % 2 == 0:
            name = name + 1
```

```
def fast_detect(image, thres, octave):

    print("in fast_detect")
    height, width = image.shape
    keypoints = []

    radian = math.floor((7 * (2**octave))/2)
    match octave:
        case 0:
            max = 9
        case 1:
            max = 18
        case 2:
            max = 32
        case 3:
            max = 64

    for h in range(radian + 30,height-radian - 30):
        for w in range(radian + 30,width-radian - 30):

            above_thres = 0
            below_thres = 0
            # main 4 corners
            if image[h-radian, w] > image[h, w] + thres: above_thres =
above_thres + 1
            elif image[h-radian, w] < image[h, w] - thres: below_thres
= below_thres + 1
            if image[h+radian, w] > image[h, w] + thres: above_thres =
above_thres + 1
            elif image[h+radian, w] < image[h, w] - thres: below_thres
= below_thres + 1
            if above_thres == 2:
                if image[h, w-radian] > image[h, w] + thres:
above_thres = above_thres + 1
```

```
                if  image[h,  w+radian]  >  image[h,  w]  +  thres:
above_thres = above_thres + 1
                if above_thres >= 3 :
                    if  image[h-radian,  w-1]  >  image[h,  w]  +  thres:
above_thres = above_thres + 1
                    if  image[h-radian,  w+1]  >  image[h,  w]  +  thres:
above_thres = above_thres + 1
                    if  image[h+radian,  w-1]  >  image[h,  w]  +  thres:
above_thres = above_thres + 1
                    if  image[h+radian,  w+1]  >  image[h,  w]  +  thres:
above_thres = above_thres + 1

                    if  image[h-1,  w-radian]  >  image[h,  w]  +  thres:
above_thres = above_thres + 1
                    if  image[h+1,  w-radian]  >  image[h,  w]  +  thres:
above_thres = above_thres + 1
                    if  image[h-1,  w+radian]  >  image[h,  w]  +  thres:
above_thres = above_thres + 1
                    if  image[h+1,  w+radian]  >  image[h,  w]  +  thres:
above_thres = above_thres + 1

                    for i in range(1,radian-1):
                        if  image[h+(radian-i),  w+i+1]  >  image[h,  w]  +
thres:  above_thres = above_thres + 1
                        if  image[h-(radian-i),  w+i+1]  >  image[h,  w]  +
thres:  above_thres = above_thres + 1
                        if  image[h+(radian-i),  w-i+1]  >  image[h,  w]  +
thres:  above_thres = above_thres + 1
                        if  image[h-(radian-i),  w-i+1]  >  image[h,  w]  +
thres:  above_thres = above_thres + 1
                    if above_thres >= max:
                        kp = keypoint(w,h, octave=octave)
                        keypoints.append(kp)

            elif below_thres == 2:
                if  image[h,  w-radian]  <  image[h,  w]  -  thres:
below_thres = below_thres + 1
                if  image[h,  w+radian]  <  image[h,  w]  -  thres:
below_thres = below_thres + 1
                if below_thres >= 3 :
                    if  image[h-radian,  w-1]  <  image[h,  w]  -  thres:
below_thres = below_thres + 1
                    if  image[h-radian,  w+1]  <  image[h,  w]  -  thres:
below_thres = below_thres + 1
                    if  image[h+radian,  w-1]  <  image[h,  w]  -  thres:
below_thres = below_thres + 1
                    if  image[h+radian,  w+1]  <  image[h,  w]  -  thres:
```

```
below_thres = below_thres + 1

                    if image[h-1, w-radian] < image[h, w] - thres:
below_thres = below_thres + 1
                    if image[h+1, w-radian] < image[h, w] - thres:
below_thres = below_thres + 1
                    if image[h-1, w+radian] < image[h, w] - thres:
below_thres = below_thres + 1
                    if image[h+1, w+radian] < image[h, w] - thres:
below_thres = below_thres + 1
                    for i in range(1,radian-1):
                        if image[h+(radian-i), w+i+1] < image[h, w] -
thres: below_thres = below_thres + 1
                        if image[h-(radian-i), w+i+1] < image[h, w] -
thres: below_thres = below_thres + 1
                        if image[h+(radian-i), w-i+1] < image[h, w] -
thres: below_thres = below_thres + 1
                        if image[h-(radian-i), w-i+1] < image[h, w] -
thres: below_thres = below_thres + 1

                    if below_thres >= max:
                        kp = keypoint(w,h, octave= octave)
                        keypoints.append(kp)

    print(f"Total features detected for layer {octave}:
{len(keypoints)}")

    return keypoints
```

```
def sort(corner_list):
    size = len(corner_list)

    for index in range(size):
        min_index = index

        for j in range(index + 1, size):
            if corner_list[j].score > corner_list[min_index].score:
                min_index = j

        corner_list[index], corner_list[min_index] =
corner_list[min_index], corner_list[index]

    return corner_list[:1000]
```

```
def find_harris_corners(input_img, threshold, fast_keypoints):
    print("in find_harris_corners")
    corner_list = []

    offset = int(5/2)

    dy, dx = np.gradient(input_img)
    Ixx = dx**2
    Ixy = dy*dx
    Iyy = dy**2
    for kp in fast_keypoints:

        x = kp.x
        y = kp.y
        octave = kp.octave
        #Values of sliding window
        start_y = y - offset
        end_y = y + offset + 1
        start_x = x - offset
        end_x = x + offset + 1

        #The variable names are representative to
        #the variable of the Harris corner equation
        windowIxx = Ixx[start_y : end_y, start_x : end_x]
        windowIxy = Ixy[start_y : end_y, start_x : end_x]
        windowIyy = Iyy[start_y : end_y, start_x : end_x]

        #Sum of squares of intensities of partial derevatives
        Sxx = windowIxx.sum()
        Sxy = windowIxy.sum()
        Syy = windowIyy.sum()

        #Calculate determinant and trace of the matrix
        det = (Sxx * Syy) - (Sxy**2)
        trace = Sxx + Syy

        #Calculate r for Harris Corner equation
        r = det - 0.06*(trace**2)

        if r > threshold:
            kp = keypoint(x,y, score = r, octave = octave)
            corner_list.append(kp)

    return sort(corner_list)
```

```
def corner_orientations(img, corners):
    print("in corner_orientations")
```

```
    # mask shape must be odd to have one centre point which is the
corner
    OFAST_MASK = np.zeros((31, 31), dtype=np.int32)
    OFAST_UMAX = [15, 15, 15, 15, 14, 14, 14, 13, 13, 12, 11, 10, 9, 8,
6, 3]
    for i in range(-15, 16):
        for j in range(-OFAST_UMAX[abs(i)], OFAST_UMAX[abs(i)] + 1):
            OFAST_MASK[15 + j, 15 + i] = 1
    mrows, mcols = OFAST_MASK.shape
    mrows2 = int((mrows - 1) / 2)
    mcols2 = int((mcols - 1) / 2)

    img_pad = np.pad(img, (mrows2, mcols2), mode='constant',
constant_values=0)

    # Calculating orientation by the intensity centroid method
    for i in range(len(corners)):
        r0, c0 = corners[i].x, corners[i].y
        m01, m10 = 0, 0
        for r in range(mrows):
            m01_temp = 0
            for c in range(mcols):
                if OFAST_MASK[r,c]:
                    I = img_pad[c0+c-1,r0+r-1]
                    m10 = m10 + I*(c-mcols2)
                    m10 = m10 + I*(-3 + c-1)
                    m01 = m01 + I*(-3 + r-1)
                    m01_temp = m01_temp + I
            m01 = m01 + m01_temp*(r-mrows2)
        corners[i].orientation = np.arctan2(m01,m10)

    return corners
```

```
def find_keypoints_across_octaves(keypoints, octaves):
    print("Finding keypoints across octaves")
    to_return_keypoints = ()
    for kp_one in keypoints:
        if kp_one.octave == 0:
            count = 0
            kp_family = []
            best_of_octave = kp_one
            for kp_two in keypoints:
                kp_family.append(kp_one)
                if kp_one.pt == kp_two.pt and kp_one.octave !=
kp_two.octave and kp_one.angle == kp_two.angle:
                    kp_family.append(kp_two)
                    count += 1
```

```
                    if kp_one.response < kp_two.response:
                        best_of_octave = kp_two
            if count == octaves:
                to_return_keypoints += (best_of_octave,)
                kp_temp = list(keypoints)
                for kp in kp_family:
                    kp_temp.remove(kp)
                kp_family.clear()
                keypoints = tuple(kp_temp)
    print("Finished finding keypoints across octaves")
    return to_return_keypoints

def topN_kepoints(keypoints, n):
    print("Finding top {} keypoints".format(n))
    size = len(keypoints)
    keypoints = list(keypoints)
    for index in range(size):
        min_index = index

        for j in range(index + 1, size):
            if keypoints[j].response > keypoints[min_index].response:
                min_index = j

        keypoints[index], keypoints[min_index] = keypoints[min_index],
keypoints[index]
    print("Finished finding top {} keypoints".format(n))
    return tuple(keypoints[:n])
```

```
def brief_descriptor_function(img , keypoints_object, pattern, n=256,
mode='uniform'):

    kernel = np.array([[1, 4,  7,  4,  1],
                       [4, 16, 26, 16, 4],
                       [7, 26, 41, 26, 7],
                       [4, 16, 26, 16, 4],
                       [1, 4,  7,  4,  1]])/273        # 5x5 Gaussian
Window

    img = convolve2d(img, kernel, mode='same')

    descriptors = np.zeros((len(keypoints_object), n), dtype=int)
    for ek, kp in enumerate(keypoints_object):
        cosangle = np.cos(kp.angle)
        sinangle = np.sin(kp.angle)
        for i in range(0, len(pattern)):
```

```
            y1 = pattern[i][0]
            x1 = pattern[i][1]
            y2 = pattern[i][2]
            x2 = pattern[i][3]

            spy0 = round(sinangle*y1 + cosangle*x1)
            spx0 = round(cosangle*y1 - sinangle*x1)
            spy1 = round(sinangle*y2 + cosangle*x2)
            spx1 = round(cosangle*y2 - sinangle*x2)

            if  img[int(kp.pt[1])  +  spy0,  int(kp.pt[0])  +  spx0]  <
img[int(kp.pt[1]) + spy1, int(kp.pt[0]) + spx1]:
                descriptors[ek][i] = 1
    end_descriptors =  bits_to_bytes(descriptors)

    return end_descriptors

def bits_to_bytes(desc):
    descriptors = ()
    for byte in desc:
        descriptor = []
        for bit_pair in range(0,32):
            desc_byte = 0
            for bit in range(0,8):
                desc_byte += (byte[bit + bit_pair] * (2 ** bit))
            descriptor.append(desc_byte)
        descriptors = descriptors + (descriptor,)

    return np.array(descriptors).astype(np.uint8)
```

```
def match(img1, img2, kp1, kp2, des1, des2, name, names):

    bf = cv2.BFMatcher(cv2.NORM_HAMMING,crossCheck=False)

    matches = []

    matches = bf.match(des1, des2)

    # sorted by distance
    matches = sorted(matches, key=lambda x: x.distance)
    img3                                                    =
cv2.drawMatches(img1,kp1,img2,kp2,matches[:30],None,flags=cv2.DrawMatch
esFlags_NOT_DRAW_SINGLE_POINTS)

    #save image
    cv2.imwrite("matches/"       +       names[name].strip("_2")       +
'_matches.jpg',img3)
```

```
    return matches[:30]
```

```
def homography_stitching(keypoints_train_img, keypoints_query_img, matches, reprojThresh):

    keypoints_train_img = np.float32([keypoint.pt for keypoint in keypoints_train_img])
    keypoints_query_img = np.float32([keypoint.pt for keypoint in keypoints_query_img])


    if len(matches) > 4:
        # construct the two sets of points
        points_train = np.float32([keypoints_train_img[m.queryIdx] for m in matches])
        points_query = np.float32([keypoints_query_img[m.trainIdx] for m in matches])

        # Calculate the homography between the sets of points
        (H, mask) = cv2.findHomography(points_query, points_train, cv2.RANSAC, 5)

        return (matches, H, mask)
    else:
        return None
```

```
def trim(frame):
    #crop top
    if not np.sum(frame[0]):
        return trim(frame[1:])
    #crop top
    if not np.sum(frame[-1]):
        return trim(frame[:-2])
    #crop top
    if not np.sum(frame[:,0]):
        return trim(frame[:,1:])
    #crop top
    if not np.sum(frame[:,-1]):
        return trim(frame[:,:-2])
    return frame
```

ΒΙΒΛΙΟΓΡΑΦΙΑ

[1] Wei, L. Y. U., Z. H. O. U. Zhong, C. H. E. N. Lang, and Z. H. O. U. Yi. "A survey on image and video stitching." Virtual Reality & Intelligent Hardware 1, no. 1 (2019): 55-83.

[2] Szeliski, Richard. "Image alignment and stitching: A tutorial." Foundations and Trends® in Computer Graphics and Vision 2, no. 1 (2007): 1-104.

[3] JShashank, K., N. SivaChaitanya, G. Manikanta, Ch NV Balaji, and V. V. S. Murthy. "A Survey and Review over Image Alignment and Stitching Methods. The." In International Journal of Electronics & Communication Technology (IJECT), ISSN. 2014.

[4] Patel, Hetal M., Asst. Prof. Pinal. J. Patel and Asst. Prof. Pinal. J. Patel. “Comprehensive Study And Review Of Image Mosaicing Methods.” International journal of engineering research and technology 1 (2012): n. pag.

[5] L. M. G. Fonseca and B. S. Manjunath, “Registration Techniques for Multisensor Remotely Sensed Imagery”, Journal of Photogrammetry Engineering and Remote Sensing, vol. 62, no. 9, pp. 1049-1056, Sep. 1996.

[6] Joglekar, Jyoti, and Shirish S. Gedam. "Area based image matching methods—A survey." Int. J. Emerg. Technol. Adv. Eng 2, no. 1 (2012): 130-136.

[7] Alomran, Murtadha, and Douglas Chai. "Feature-based panoramic image stitching." In 2016 14th International Conference on Control, Automation, Robotics and Vision (ICARCV), pp. 1-6. IEEE, 2016.

[8] Rosin, Paul L. "Measuring corner properties." Computer Vision and Image Understanding 73, no. 2 (1999): 291-307.

[9] Adel, Ebtsam, Mohammed Elmogy, and Hazem Elbakry. "Image stitching based on feature extraction techniques: a survey." International Journal of Computer Applications 99, no. 6 (2014): 1-8.

[10] Lowe, David G. "Distinctive image features from scale-invariant keypoints." International journal of computer vision 60, no. 2 (2004): 91-110.

[11] Liu, Wen, Shuo Wang, Zhongliang Deng, Tingting Fan, Mingjie Jia, and Hong Chen. "A Review of Image Feature Descriptors in Visual Positioning." IPIN-WiP (2021).

[12] Hassaballah, Mahmoud, Aly Amin Abdelmgeid, and Hammam A. Alshazly. "Image features detection, description and matching." In Image Feature Detectors and Descriptors, pp. 11-45. Springer, Cham, 2016.

[13] Dey, Nilanjan, Pradipti Nandi, Nilanjana Barman, Debolina Das, and Subhabrata Chakraborty. "A comparative study between Moravec and Harris corner detection of noisy images using adaptive wavelet thresholding technique." arXiv preprint arXiv:1209.1558 (2012).

[14] Harris, Chris, and Mike Stephens. "A combined corner and edge detector." In Alvey vision conference, vol. 15, no. 50, pp. 10-5244. 1988.

[15] Moravec, Hans P. "Techniques towards automatic visual obstacle avoidance." (1977).

[16] Shi, Jianbo. "Good features to track." In 1994 Proceedings of IEEE conference on computer vision and pattern recognition, pp. 593-600. IEEE, 1994.

[17] Juranek, Lubas, Jiri Stastny, and Vladislav Skorpil. "Effect of Low-Pass Filters as a Shi-Tomasi Corner Detector's Window Functions." In 2018 41st International Conference on Telecommunications and Signal Processing (TSP), pp. 1-5. IEEE, 2018.

[18] Kadhim, Haydar Abdulameer, and Waleed Abdullah Araheemah. "A method to improve corner detectors (Harris, Shi-Tomasi & FAST) using adaptive contrast enhancement filter." Periodicals of Engineering and Natural Sciences (PEN) 8, no. 1 (2020): 508-515.

[19] Smith, Stephen M., and J. Michael Brady. "SUSAN—a new approach to low level image processing." International journal of computer vision 23, no. 1 (1997): 45-78.

[20] Rosten, Edward, and Tom Drummond. "Machine learning for high-speed corner detection." In European conference on computer vision, pp. 430-443. Springer, Berlin, Heidelberg, 2006.

[21] Mair, Elmar, Gregory D. Hager, Darius Burschka, Michael Suppa, and Gerhard Hirzinger. "Adaptive and generic corner detection based on the accelerated segment test." In European conference on Computer vision, pp. 183-196. Springer, Berlin, Heidelberg, 2010.

[22] Chen, Jie, Li-hui Zou, Juan Zhang, and Li-hua Dou. "The Comparison and Application of Corner Detection Algorithms." Journal of multimedia 4, no. 6 (2009).

[23] Rosten, Edward, Reid Porter, and Tom Drummond. "Faster and better: A machine learning approach to corner detection." IEEE transactions on pattern analysis and machine intelligence 32, no. 1 (2008): 105-119.

[24] Calonder, Michael, Vincent Lepetit, Christoph Strecha, and Pascal Fua. "Brief: Binary robust independent elementary features." In European conference on computer vision, pp. 778-792. Springer, Berlin, Heidelberg, 2010.

[25] Alahi, Alexandre, Raphael Ortiz, and Pierre Vandergheynst. "Freak: Fast retina keypoint." In 2012 IEEE conference on computer vision and pattern recognition, pp. 510-517. Ieee, 2012.

[26] Lowe, David G. "Object recognition from local scale-invariant features." In Proceedings of the seventh IEEE international conference on computer vision, vol. 2, pp. 1150-1157. Ieee, 1999.

[27] Yuvaraju, M., K. Sheela, and S. S. Rani. "Feature extraction of real-time image using Sift algorithm." International Journal of Research in Electrical & Electronics Engineering 3, no. 4 (2015): 01-07.

[28] Shaikh, Taherim S., and Archana B. Patankar. "Multiple feature extraction techniques in image stitching." International Journal of Computer Applications 123, no. 15 (2015).

[29] Ke, Yan, and Rahul Sukthankar. "PCA-SIFT: A more distinctive representation for local image descriptors." In Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004., vol. 2, pp. II-II. IEEE, 2004.

[30] Mikolajczyk, Krystian, and Cordelia Schmid. "A performance evaluation of local descriptors." IEEE transactions on pattern analysis and machine intelligence 27, no. 10 (2005): 1615-1630.

[31] Yu, Guoshen, and Jean-Michel Morel. "ASIFT: An algorithm for fully affine invariant comparison." Image Processing On Line 1 (2011): 11-38.

[32] Bay, Herbert, Tinne Tuytelaars, and Luc Van Gool. "Surf: Speeded up robust features." In European conference on computer vision, pp. 404-417. Springer, Berlin, Heidelberg, 2006.

[33] Mikolajczyk, Krystian, and Cordelia Schmid. "Indexing based on scale invariant interest points." In Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001, vol. 1, pp. 525-531. IEEE, 2001.

[34] Güzel, Mehmet Serdar. "A hybrid feature extractor using fast hessian detector and SIFT." Technologies 3, no. 2 (2015): 103-110.

[35] Zhang, Tian, Rui Zhao, and Zhongsheng Chen. "Application of migration image registration algorithm based on improved SURF in remote sensing image mosaic." IEEE Access 8 (2020): 163637-163645.

[36] Rublee, Ethan, Vincent Rabaud, Kurt Konolige, and Gary Bradski. "ORB: An efficient alternative to SIFT or SURF." In 2011 International conference on computer vision, pp. 2564-2571. Ieee, 2011

[37] Xie, Yinggang, Quan Wang, Yuanxiong Chang, and Xueyuan Zhang. "Fast Target Recognition Based on Improved ORB Feature." Applied Sciences 12, no. 2 (2022): 786.

[38] Yunsheng, Zhang, and Zou Zhengrong. "Automatic registration method for remote sensing images based on improved ORB algorithm." Remote Sensing for Land & Resources 25, no. 3 (2013): 20-24.

[39] Bian, JiaWang, Wen-Yan Lin, Yasuyuki Matsushita, Sai-Kit Yeung, Tan-Dat Nguyen, and Ming-Ming Cheng. "Gms: Grid-based motion statistics for fast, ultra-robust feature correspondence." In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 4181-4190. 2017.

[40] Qin, Yanyan, Hongke Xu, and Huiru Chen. "Image feature points matching via improved ORB." In 2014 IEEE International Conference on Progress in Informatics and Computing, pp. 204-208. IEEE, 2014.

[41] Leutenegger, Stefan, Margarita Chli, and Roland Y. Siegwart. "BRISK: Binary robust invariant scalable keypoints." In 2011 International conference on computer vision, pp. 2548-2555. Ieee, 2011.

[42] Zhu, Yaguang, Chaoyu Jia, Chao Ma, and Qiong Liu. "SURF-BRISK–based image infilling method for terrain classification of a legged robot." Applied Sciences 9, no. 9 (2019): 1779.

[43] Jing, Huiyun, Xin He, Qi Han, and Xiamu Niu. "CBRISK: Colored binary robust invariant scalable keypoints." IEICE TRANSACTIONS on Information and Systems 96, no. 2 (2013): 392-395.

[44] ZHANG, Heng, Dayong LIU, Yanli LIU, and Chenxi NIE. "Improved binary robust invariant scalable keypoints algorithm fusing depth information." Journal of Computer Applications 35, no. 8 (2015): 2285.

[45] Yang, Shuqiang, Biao Li, and Kun Zeng. "SBRISK: speed-up binary robust invariant scalable keypoints." Journal of Real-Time Image Processing 12, no. 3 (2016): 583-591.

[46] Alcantarilla, Pablo Fernández, Adrien Bartoli, and Andrew J. Davison. "KAZE features." In European conference on computer vision, pp. 214-227. Springer, Berlin, Heidelberg, 2012.

[47] Alcantarilla, Pablo F., and T. Solutions. "Fast explicit diffusion for accelerated features in nonlinear scale spaces." IEEE Trans. Patt. Anal. Mach. Intell 34, no. 7 (2011): 1281-1298.

[48] Forero, Manuel G., Claudia L. Mambuscay, María F. Monroy, Sergio L. Miranda, Dehyro Méndez, Milton Orlando Valencia, and Michael Gomez Selvaraj. "Comparative analysis of detectors and feature descriptors for multispectral image matching in rice crops." Plants 10, no. 9 (2021): 1791.

[49] Dijkstra, Willem, and Tonnie Boersma. "2D keypoint detection and description." 14th SC@ RUG 2016-2017: 70.

[50] Mikolajczyk, Krystian, Tinne Tuytelaars, Cordelia Schmid, Andrew Zisserman, Jiri Matas, Frederik Schaffalitzky, Timor Kadir, and L. Van Gool. "A comparison of affine region detectors." International journal of computer vision 65, no. 1 (2005): 43-72.

[51] Adel, Ebtsam, Mohammed Elmogy, and Hazem Elbakry. "Image stitching system based on ORB feature based technique and compensation blending." International Journal of Advanced Computer Science and Applications 6, no. 9 (2015).

[52] Tareen, Shaharyar Ahmed Khan, and Zahra Saleem. "A comparative analysis of sift, surf, kaze, akaze, orb, and brisk." In 2018 International conference on computing, mathematics and engineering technologies (iCoMET), pp. 1-10. IEEE, 2018.

[53] Andersson, Oskar, and Steffany Reyna Marquez. "A comparison of object detection algorithms using unmanipulated testing images: Comparing SIFT, KAZE, AKAZE and ORB." (2016).

[54] Truong, Mai Thanh Nhat, and Sanghoon Kim. "A review on image feature detection and description." In Proceedings of the Korea Information Processing Society Conference, pp. 677-680. Korea Information Processing Society, 2016.

[55] Hong, Sungchul, and Hyu-Soung Shin. "Comparative Performance Analysis of Feature Detection and Matching Methods for Lunar Terrain Images." KSCE Journal of Civil and Environmental Engineering Research 40, no. 4 (2020): 437-444.

[56] Hidalgo, Franco, and Thomas Bräunl. "Evaluation of several feature detectors/extractors on underwater images towards vslam." Sensors 20, no. 15 (2020): 4343.

[57] Mukherjee, Dibyendu, Q. M. Jonathan Wu, and Guanghui Wang. "A comparative experimental study of image feature detectors and descriptors." Machine Vision and Applications 26, no. 4 (2015): 443-466.

[58] Yakovleva, Olena, and Kateryna Nikolaieva. "Research of descriptor based image normalization and comparative analysis of SURF, SIFT, BRISK, ORB, KAZE, AKAZE descriptors." Advanced Information Systems 4, no. 4 (2020): 89-101.

[59] Bakar, Siddique Abu, Xiaoming Jiang, Xiangfu Gui, Guoquan Li, and Zhangyong Li. "Image stitching for chest digital radiography using the SIFT and SURF feature extraction by RANSAC algorithm." In Journal of Physics: Conference Series, vol. 1624, no. 4, p. 042023. IOP Publishing, 2020.

[60] Abbadi, Nidhal K. EL, Safaa Alwan Al Hassani, and Ali Hussein Abdulkhaleq. "A Review Over Panoramic Image Stitching Techniques." In Journal of Physics: Conference Series, vol. 1999, no. 1, p. 012115. IOP Publishing, 2021.

[61] Fischler, Martin A., and Robert C. Bolles. "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography." Communications of the ACM 24, no. 6 (1981): 381-395.

[62] Derpanis, Konstantinos G. "Overview of the RANSAC Algorithm." Image Rochester NY 4, no. 1 (2010): 2-3.

[63] Nenkov, Ivaylo, and Miroslav Galabov. "RANSAC robust estimation algorithm overview [C]." In VIII International scientific conference. 2015.

[64] Rousseeuw, Peter J. "Least median of squares regression." Journal of the American statistical association 79, no. 388 (1984): 871-880.

[65] Twetman, Theodor. "Multi view image stitching of planar surfaces on mobile devices: large surface analog notes scanning." (2015).

[66] Xiong, Yalin, and Kenneth Turkowski. "Registration, calibration and blending in creating high quality panoramas." In Proceedings Fourth IEEE Workshop on

Applications of Computer Vision. WACV'98 (Cat. No. 98EX201), pp. 69-74. IEEE, 1998.

[67] Burt, Peter J., and Edward H. Adelson. "A multiresolution spline with application to image mosaics." ACM Transactions on Graphics (TOG) 2, no. 4 (1983): 217-236.

[68] Ataky, Steve Tsham Mpinda, Jonathan De Matos, Alceu de S. Britto, Luiz ES Oliveira, and Alessandro L. Koerich. "Data augmentation for histopathological images based on gaussian-laplacian pyramid blending." In 2020 International Joint Conference on Neural Networks (IJCNN), pp. 1-8. IEEE, 2020.

[69] Xiong, Yingen, and Kari Pulli. "Gradient domain image blending and implementation on mobile devices." In International Conference on Mobile Computing, Applications, and Services, pp. 293-306. Springer, Berlin, Heidelberg, 2009.

[70] Zhang, Lingzhi, Tarmily Wen, and Jianbo Shi. "Deep image blending." In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, pp. 231-240. 2020.

[71] Szeliski, Richard. Computer vision: algorithms and applications. Springer Science & Business Media, 2010.

[72] Uyttendaele, Matthew, Ashley Eden, and Richard Skeliski. "Eliminating ghosting and exposure artifacts in image mosaics." In Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001, vol. 2, pp. II-II. IEEE, 2001.

[73] Ian, Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, and David Warde-Farley. "Generative adversarial nets." In Advances in neural information processing systems." (2014): 2672-2680.

[74] Liu, Mingcong, Qiang Li, Zekui Qin, Guoxin Zhang, Pengfei Wan, and Wen Zheng. "Blendgan: implicitly gan blending for arbitrary stylized face generation." Advances in Neural Information Processing Systems 34 (2021): 29710-29722.

[75] Zhan, Fangneng, Hongyuan Zhu, and Shijian Lu. "Spatial fusion gan for image synthesis." In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 3653-3662. 2019.

[76] Wu, Huikai, Shuai Zheng, Junge Zhang, and Kaiqi Huang. "Gp-gan: Towards realistic high-resolution image blending." In Proceedings of the 27th ACM international conference on multimedia, pp. 2487-2495. 2019.

[78] Bonny, Moushumi Zaman, and Mohammad Shorif Uddin. "Feature-based image stitching algorithms." In 2016 International Workshop on Computational Intelligence (IWCI), pp. 198-203. IEEE, 2016.

[79] Marr, David, and Ellen Hildreth. "Theory of edge detection." Proceedings of the Royal Society of London. Series B. Biological Sciences 207, no. 1167 (1980): 187-217.

[80] Rosten, Edward, and Tom Drummond. "Fusing points and lines for high performance tracking." In Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1, vol. 2, pp. 1508-1515. Ieee, 2005.

[81] Marchand, Eric, Hideaki Uchiyama, and Fabien Spindler. "Pose estimation for augmented reality: a hands-on survey." IEEE transactions on visualization and computer graphics 22, no. 12 (2015): 2633-2651.

[82] Dixit, Gaurav. "Improved algorithm for blob detection in document images." In 2014 5th International Conference-Confluence The Next Generation Information Technology Summit (Confluence), pp. 703-708. IEEE, 2014.

[83] Bundy, Alan, and Lincoln Wallen. "Difference of gaussians." In Catalogue of Artificial Intelligence Tools, pp. 30-30. Springer, Berlin, Heidelberg, 1984.

[84] Gangeh, Mehrdad J., Amir H. Shabani, and Mohamed S. Kamel. "Nonlinear scale space theory in texture classification using multiple classifier systems." In International Conference Image Analysis and Recognition, pp. 147-156. Springer, Berlin, Heidelberg, 2010.

[85] Zhang, Yi, and Xiaozhong Yang. "On the acceleration of AOS schemes for nonlinear diffusion filtering." Journal of Multimedia 5, no. 6 (2010): 605.

[86] Creswell, Antonia, Tom White, Vincent Dumoulin, Kai Arulkumaran, Biswa Sengupta, and Anil A. Bharath. "Generative adversarial networks: An overview." IEEE signal processing magazine 35, no. 1 (2018): 53-65.