ΠΑΝΕΠΙΣΤΗΜΙΟ ΔΥΤΙΚΗΣ ΑΤΤΙΚΗΣ

ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΒΙΟΜΗΧΑΝΙΚΗΣ ΣΧΕΔΙΑΣΗΣ & ΠΑΡΑΓΩΓΗΣ

Διπλωματική Εργασία:

Μοντέλο Εντοπισμού Θέσης και Πλοήγησης σε Εσωτερικό Χώρο με Χρήση Πρωτοκόλλου Bluetooth Low Energy

Φοιτητής: Δημοσθένης Μαργαρίτης
ΑΜ: 71445184

Επιβλέπουσα Καθηγήτρια:

Λελίγκου Ελένη Αικατερίνη

Αθήνα-Αιγάλεω, Ιούνιος 2023

Η Διπλωματική Εργασία έγινε αποδεκτή και βαθμολογήθηκε από την εξής τριμελή επιτροπή:

(Ονοματεπώνυμο), (βαθμίδα)	(Ονοματεπώνυμο), (βαθμίδα)	(Ονοματεπώνυμο), (βαθμίδα)
Eleni Aikaterini Leligkou Digitally signed by Eleni Aikaterini Leligkou Date: 2023.07.19 10:49:42 +03'00'	Evangelos Pallis Digitally signed by Evangelos Pallis Date: 2023.07.19 11:18:54 +03'00'	Christos Drosos Digitally signed by Christos Drosos Date: 2023.07.19 16:40:25 +03'00'
(Υπογραφή)	(Υπογραφή)	(Υπογραφή)

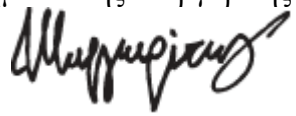
ΔΗΛΩΣΗ ΣΥΓΓΡΑΦΕΑ ΔΙΠΛΩΜΑΤΙΚΗΣ ΕΡΓΑΣΙΑΣ

Ο/η κάτωθι υπογεγραμμένος Δημοσθένης Μαργαρίτης του Γεωργίου, με αριθμό μητρώου 71445184 φοιτητής του Πανεπιστημίου Δυτικής Αττικής της Σχολής ΜΗΧΑΝΙΚΩΝ του Τμήματος Μηχανικών Βιομηχανικού Σχεδιασμού και Παραγωγής,

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 διπλώματός μου.

Ο Δήλων
Δημοσθένης Μαργαρίτης



0: Ευχαριστίες

Η παρούσα εργασία εκπονήθηκε στο Εργαστήριο Υπολογιστικής Νοημοσύνης και Ευφυών Συστημάτων του Πανεπιστημίου Δυτικής Αττικής. Για την ευκαιρία να συμμετάσχω στην ομάδα του θα ήθελα να ευχαριστήσω την Δρ. Ελένη Αικατερίνη Λελίγκου, της οποίας οι εργασίες εξαμήνου ήταν οι πιο σημαντικές της φοίτησης μου.

Ευχαριστώ επίσης τους γονείς μου για την ατελείωτη υπομονή τους.

Εσείς που πιστεύατε σε εμένα ακόμα και όταν εγώ όχι, ξέρετε ποιοι είστε. Άσχημος θα ήταν ο κόσμος μου χωρίς εσάς.

1: Εισαγωγή

1.1: Περίληψη

Μία πτυχή της ιδέας του Internet of Things είναι η δημιουργία φυσικών εγκαταστάσεων στις οποίες οι αλληλεπιδράσεις περνούν το όριο ανάμεσα στον πραγματικό και τον εικονικό χώρο. Ένα κοινό πρόβλημα στην δημιουργία τέτοιων εγκαταστάσεων είναι ο προσδιορισμός της τοποθεσίας του χρήστη. Οι υπάρχουσες λύσεις που δεν βασίζονται σε εξειδικευμένο εξοπλισμό είτε δεν παρέχουν επαρκή ακρίβεια είτε βασίζονται στην μέθοδο fingerprinting, που καθιστά την δημιουργία και συντήρηση τέτοιων εγκαταστάσεων μη πρακτική. Ο σκοπός της παρούσας διπλωματικής εργασίας είναι η πρόταση μιας ενδιάμεσης λύσης με την χρήση πομπών Bluetooth Low Energy με εστίαση στην πρακτικότητα και η πρόταση μιας δομής δεδομένων για τέτοιες εγκαταστάσεις. Επίσης, προτείνεται μια λύση στο ζήτημα πλοήγησης σε τουριστικούς χώρους, οπού ο χρήστης δεν αναζητεί την συντομότερη διαδρομή, μα ένα δρομολόγιο. Επιπλέον, εξετάζονται οι απαιτήσεις backend τέτοιων εγκαταστάσεων και οι επιπτώσεις της έκδοσης 5.1 του Bluetooth Specification στο εγγύς μέλλον.

Λέξεις κλειδιά

Bluetooth Low-Energy, Πλοήγηση εσωτερικού χώρου, Χαρτογραφία εσωτερικού χώρου, Τουριστικές εφαρμογές, Βελτιστοποίηση δρομολογίου, Διάδωση σήματος, Πανταχού παρούσα πληροφορική

1.2: Abstract

One aspect of the Internet of Things paradigm is the creation of physical installations in which interactions seamlessly cross the boundary between real space and cyberspace. A common problem in the creation of such spaces is determining the user’s location. Exisitng solutions that do not involve specialized equipment either fail to yield usable accuracy or require the use of fingerprinting, which renders the creation and maintenance of such applications impractical. The aim of this thesis is to propose an in-between solution focused on practicality using Bluetooth Low Energy beacons and a data structure for such installations. A solution for navigating tourist spaces where the user is needs an itinerary rather than an optimal path is also provided. Additionally, backend requirements and the near-future implications of version 5.1 of the Bluetooth specification are examined.

Keywords

Bluetooth Low-Energy, Indoor Navigation, Indoor cartography, Tourist Applications, Itinerary Optimization, Signal propagation, Ubiquitous Computing

1.3: Περιγραφή της παρούσας εργασίας

Η παρούσα εργασία ξεκίνησε με την ιδέα μιας εφαρμογής για κινητές συσκευές η οποία δρα ως ξεναγός σε μουσεία και άλλους χώρους τουριστικού ενδιαφέροντος. Τα κύρια προβλήματα προς επίλυση κατά την ανάπτυξη μιας τέτοιας εφαρμογής είναι τα εξής: - Ο προσδιορισμός της τοποθεσίας του χρήστη. - Η αναγωγή της σε κάποια αναπαράσταση κτηρίου. - Η αναπαράσταση των προτιμήσεων του χρήστη και η χρήση τους για την πρόταση ενός δρομολογίου.

Σημαντικότερο από αυτά τα προβλήματα είναι το πρώτο. Η χρήση GPS δεν παράγει επαρκή ακρίβεια σε εσωτερικούς χώρους. Κατά συνέπεια, οποιαδήποτε εγκατάσταση φέρει υποστήριξη εφαρμογής ξεναγού πρέπει να διαθέτει ένα σύνολο ασύρματων πομπών που δρουν ως φάροι εντός της. Το πιο κατάλληλο πρωτόκολλο για τον σκοπό αυτό είναι το BLE, αν και οι δέκτες BLE που βρίσκονται στις περισσότερες έξυπνες κινητές συσκευές δεν προορίζονται για τέτοιες εφαρμογές.

Δυστυχώς, το συμπέρασμα που προέκυψε από την βιβλιογραφία είναι πως οποιαδήποτε λύση στο ζήτημα της επεξεργασίας των δεδομένων που παράγονται από αυτούς τους πομπούς πρέπει να επιλέξει ανάμεσα στην ακρίβεια και την ευκολία εφαρμογής της. Στην μία άκρη βρίσκεται η συμβατική λύση του τριπλευρισμού, η οποία για δύο διαστάσεις χρειάζεται τρεις πομπούς και τις συντεταγμένες τους. Στην άλλη άκρη βρίσκεται η χρήση Fingerprinting, η οποία χρειάζεται σημαντική δαπάνη ώρας για την παραγωγή δεδομένων σήματος σε αρκετά σημεία στον χώρο. Στην περίπτωση επεξεργασίας των δεδομένων με χρήση νευρωνικών δικτύων, χρειάζονται και σημαντικοί υπολογιστικοί πόροι για την εκπαίδευση τους.

Καθώς οι μεν λύσεις δεν παρέχουν αρκετή ακρίβεια ώστε να είναι χρήσιμες ενώ οι δε είναι εξαιρετικά δύσκολο να εφαρμοστούν στον πραγματικό κόσμο, σκοπός της παρούσας εργασίας ως προς το πρόβλημα προσδιορισμού τοποθεσίας είναι η εύρεση μίας μέσης λύσης.

Το δεύτερο ζήτημα είναι η διερμηνεία της τοποθεσίας του χρήστη από κάποιο μοντέλο κτηρίου. Αυτό θέτει και την ελάχιστη απαιτούμενη ακρίβεια κατά τον προσδιορισμό τοποθεσίας. Καθώς η τοποθέτηση πομπών προϋποθέτει δαπάνη χρημάτων, το μοντέλο αυτό πρέπει να ρυθμίζεται στις ανάγκες της εγκατάστασης ανά χώρο. Παραδείγματος χάριν, σε ορισμένα δωμάτια μπορεί να απαιτείται γνώση της θέσης του χρήστη σε δύο διαστάσεις εντός του δωματίου, ενώ σε άλλα να αρκεί η γνώση του αν ο χρήστης βρίσκεται μέσα στο δωμάτιο ή όχι. Για αυτό τον λόγο μαζί με την λύση που προτείνεται, προτείνονται και οι αρχές κατά τις οποίες πρέπει να κρίνονται πιθανές λύσεις.

Για το τελευταίο πρόβλημα, η δημιουργία διαδρομής αντιμετωπίζεται με την χρήση εξελικτικού αλγορίθμου. Για τον σκοπό αυτό, ορίζονται πράξεις οι οποίες μπορούν να εκτελεστούν σε ένα δρομολόγιο και συνάρτηση καταλληλότητας σε σχέση με τις απαιτήσεις του χρήστη. Ακολουθεί η δομή της εργασίας.

1.4: Δομή της εργασίας

Παρακάτω βρίσκεται η ανασκόπηση της βιβλιογραφίας. Στο κεφάλαιο 2 δίνονται πληροφορίες για το πρωτόκολλο BLE και τα πρωτόκολλα που ορίζουν τους πομπούς. Στο κεφάλαιο 3 περιγράφεται ο τρόπος που αξιοποίησα τα πρωτόκολλα BLE και iBeacon, αλλά και η μέθοδος που χρησιμοποίησα για να αυξήσω την ακρίβεια προσδιορισμού απόστασης. Στα κεφάλαια 4 και 5 προτείνεται ένα μοντέλο αναπαράστασης κτηρίου φτιαγμένο ώστε να διευκολύνει την πλοήγηση σε εσωτερικό χώρο και πως αυτό προσδιορίζει την τοποθεσία του χρήστη. Το κεφάλαιο 6 εξηγεί τον τρόπο που το μοντέλο κτηρίου χρησιμοποιείται για την παραγωγή διαδρομών. Το 7ο κεφάλαιο πραγματεύεται τις απαιτήσεις που έχει μια τέτοια εφαρμογή από το backend της. Το Κεφάλαιο 8 παρουσιάζει της εφαρμογές που δημιουργήθηκαν ως μέρος αυτής της εργασίας. Το Κεφάλαιο 9 περιέχει τον επίλογο και τα συμπεράσματα, και στο παράρτημα βρίσκεται η υλοποίηση των ιδεών της εργασίας στην γλώσσα προγραμματισμού Dart.

1.5: Βιβλιογραφική ανασκόπηση

Ubiquitous Computing

Αρχικά, αξίζει μία εξέταση του παρόντος φιλοσοφικού πλαισίου γύρω από την πρόοδο της πληροφορικής. Στο άρθρο του "The Computer for the 21st Century"[1], έτους 1991, ο Mark Weiser προέβλεψε ένα μέλλον στο οποίο οι υπολογιστές ακολούθησαν την πορεία του κινητήρα, δηλαδή εξελίχτηκαν από τεράστιες μηχανές που κάθε εργοστάσιο δεν είχε παρά μία, της οποίας η παρουσία δεν μπορούσε να παραμεληθεί σε κάτι πανταχού παρών μα απαρατήρητο, με το οποίο ο χρήστης αλληλεπιδρά χωρίς να το σκέφτεται. Σε αυτό το μέλλον, τερματικά βρίσκονται παντού και παίρνουν την μορφή συσκευών που σήμερα θα αναγνωρίζαμε ως tablets και smartphones. Στην ιδέα αυτή έδωσε το όνομα ubiquitous computing, δηλαδή πληροφορική πανταχού παρούσα μα απαρατήρητη. Στο όραμα του Weiser όμως, παρατηρείται ένα μεγάλο επίπεδο κεντροποίησης, καθώς οι αμέτρητες συσκευές ανά δωμάτιο όλες είναι προσωπεία ενός δικτύου. Επίσης η αλληλεπίδραση είναι από χρήστη σε υπολογιστή ή από χρήστη σε χρήστη με μεσάζοντα τους υπολογιστές.

Η ιδέα του Ubiquitous Computing επεκτάθηκε από τους Lyytinen & Yoo εν έτει 2002[2]. Στο άρθρο τους, οι συγγραφείς λαμβάνοντας, υπόψιν τους τις εξελίξεις των έντεκα χρόνων από την δημοσίευση του "The Computer for the 21st Century" επεκτείνουν στην ιδέα του Ubiquitous Computing θέτοντας ως απόλυτη μορφή του την συσκευή η οποία κινούμενη με τον χρήστη, χτίζει μοντέλα του περίγυρου της και τα χρησιμοποιεί για να παρέχει ανάλογες υπηρεσίες στον χρήστη.

Το τελευταίο κομμάτι του παζλ είναι η αλληλεπίδραση του χρήστη με το περιβάλλον του και η αλληλεπίδραση του περιβάλλοντος με τον εαυτό του, εισάγοντας σε αυτό ένα σύμπλεγμα αισθητήρων και ενεργοποιητών οι οποίοι είναι συνδεδεμένοι στο διαδίκτυο, επιτρέποντας έτσι να παραχθεί το προαναφερόμενο μοντέλο του περίγυρου. Αυτή η ιδέα είναι πλέον γνωστή ως Internet of Things[3]. Ένα σημαντικό πρόβλημα στην δημιουργία τέτοιων χώρων είναι ο προσδιορισμός της θέσης του χρήστης εντός τους.

Προσδιορισμός τοποθεσίας

Την τελευταία δεκαετία, διάφορες τεχνικές έχουν εξεταστεί για τον ακριβή προσδιορισμό της θέσης μιας συσκευής σε έναν χώρο, οι κυριότερες εκ των οποίων περιγράφονται το 2021 από τους Raza et al.[4]. Συγκεκριμένα, εξετάζουν την χρήση Wi-Fi Received Signal Strength Indication, Wi-Fi Round Trip Time και ορατών σημαδιών, τα οποία επεξεργάζονται με τριπλευρισμό, Fingerprinting με Νευρωνικά δίκτυα και Fingerprinting με αλγόριθμο Nearest Neighbor. Η μετατροπή RSSI σε απόσταση d για χρήση σε τριπλευρισμό έγινε ως

$$d = 10^{\left\lceil \frac{(27.55-(20*log_{10}(f))-s_{rssi})}{20} \right\rceil}$$

Όπου s_{rssi} το RSSI και f η συχνότητα του σήματος σε MHz (2400).

Από τις εισόδους, βέλτιστη απόδοση είχαν τα ορατά σημάδια, με τα RSSI και RTT να επιδεικνύουν παρόμοιες επιδόσεις. Από τους αλγορίθμους, μέγιστη ακρίβεια απέδωσαν οι μεθόδοι νευρωνικών δικτύων και Nearest Neighbor, με μικρή διαφορά μεταξύ τους. Ο τριπλευρισμός απέδωσε χειρότερη ακρίβεια κατά μία τάξη μεγέθους.

Μια πιο λεπτομερής μελέτη στην μέθοδο του τριπλευρισμού παρήγαγαν οι Pakanon et al. το 2020[5]. Το συγκεκριμένο έργο αρχίζει με τους συγγραφείς να περιγράφουν λεπτομερώς την μέθοδο του τριπλευρισμου. Στη συνέχεια, περιγράφουν μια πειραματική

διάταξη με τρεις BLE πομπούς σε ένα δωμάτιο μήκους 9.0m και πλάτους 10.2m. Χρησιμοποιώντας μία κινητή συσκευή, μέτρησαν το σφάλμα της μεθόδου σε 163 σημεία στο δωμάτιο. Οι μετρήσεις τους έγιναν με μεγέθη δείγματος 1, 5, και 10 και παρήγαγαν σφάλματα με μέσους όρους 39.06m, 8.96m και 7.97m και διάμεσους 8.04, 5.54m και 4.23m αντίστοιχα. Η μετατροπή από RSSI σε απόσταση d έγινε με την χρήση της εκθετικής εξίσωσης

$$d = 10^{(s_{rssi} - s_{0.1m} n)}$$

Όπου s_{rssi} το RSSI και s_{1m} η ισχύς του σήματος σε απόσταση ενός μέτρου (-69.72dBmW). n είναι σταθερά και δίνεται ως 1.27.

Στην ίδια αίθουσα του ίδιου ιδρύματος, με ίδια διάταξη πομπών και σημείων μέτρησης έγινε επίσης μια μελέτη πάνω στο Fingerprinting[6], παράγοντας μέσα σφάλματα απόστασης 3.45m και 2.59m με 1 και 5 μετρήσεις αντίστοιχα.

Παρατηρείται πως οι μετρήσεις των τελευταίων δύο μελετών παρήγαγαν σημαντικά υψηλότερα σφάλματα από αυτά των Raza et al.[4] με την ίδια μέθοδο. Αυτό κατά πάσα πιθανότητα οφείλεται σε σημαντικό μέρος στην υψηλότερη ισχύ μετάδοσης του τυπικού Wi-Fi AP σε σχέση με τον τυπικό BLE πομπό (20 dBmW και 0 dBmW αντίστοιχα). Η διαφορά πρωτοκόλλου όμως δεν εξηγεί την διαφορά ανάμεσα στις εξισώσεις που χρησιμοποιήθηκαν για την παραγωγή αποστάσεως από το RSSI, καθώς τα δύο πρωτόκολλα χρησιμοποιούν την ίδια συχνότητα. Η χρήση διαφορετικών εξισώσεων κρίθηκε άξια περαιτέρω έρευνας.

Κατά την αναζήτηση απαντήσεων, βρέθηκε το paper των Astafiev et al., “Development of indoor positioning algorithm based on Bluetooth Low Energy beacons for building RTLS-Systems”[7], στα πλαίσια του οποίου οι συγγραφείς αρχικά παρήγαγαν μία διάταξη τριπλευρισμού παρατάσσοντας τρεις BLE πομπούς σε τριγωνική διάταξη σε ένα τραπέζι. Χρησιμοποιώντας ένα κινητό τηλέφωνο, μέτρησαν το σφάλμα του τριπλευρισμού με υπάρχουσες εξισώσεις και παρήγαγαν σφάλμα τάξης 12-25%. Έπειτα, εκτέλεσαν μετρήσεις του RSSI ως συνάρτηση της απόστασης ανάμεσα στον πομπό και την κινητή συσκευή. Αυτές οι μετρήσεις έφεραν απροσδόκητα αποτελέσματα, κατά τα οποία το RSSI σε απόσταση 0.5m είναι μικρότερο από αυτό σε απόσταση 1m. Όμως από τα αποτελέσματα αυτά παράχθηκε ένα πολυώνυμο 5ου βαθμού, του οποίου η χρήση στον τριπλευρισμό μείωσε το σφάλμα από 12-25% στο 3-7%, οδηγώντας στο συμπέρασμα πως ο τριπλευρισμός έχει όντως πρακτικές εφαρμογές, μα απαιτείται βαθμονόμηση των πομπών. Φυσικά, η επιφάνεια ενός τραπεζιού δεν είναι αντιπροσωπευτική ενός δωματίου.

2: Περί του Πρωτοκόλλου BLE[8]

Το πρωτόκολλο BLE είναι μια τεχνολογία ασύρματης δικτύωσης η οποία δημιουργήθηκε με σκοπό να καλύψει τα κενά του "κλασσικού" πρωτοκόλλου Bluetooth. Συγκεκριμένα, τις απαιτήσεις που έχουν συσκευές που ανάγονται στην φιλοσοφία του ubiquitous computing[2]. Η ασύρματη δικτύωση, σύμφωνα με αυτό τον τρόπο σκέψης, χρειάζεται να είναι παθητική (δηλαδή η ανταλλαγή δεδομένων να μπορεί να γίνεται χωρίς παρεμβολή του χρήστη), εύκολη (ώστε οι πομποδέκτες να είναι μικροί και οικονομικοί), να επικεντρώνεται στην ανταλλαγή σύντομων μηνυμάτων και να καταναλώνει ελάχιστη ενέργεια. Το BLE δίνει έμφαση σε αυτές τις ποιότητες και διαφοροποιείται από το "κλασσικό" Bluetooth ως εξής:

- Σημαντικά μειωμένη ισχύς, επιτρέποντας μήνες συνεχούς λειτουργίας χωρίς αλλαγή μπαταρίας
- Ικανότητα αποστολής δεδομένων χωρίς σύζευξη
- Μικρότερος ελάχιστος χρόνος αποστολής δεδομένων
- Μικρότερος ρυθμός αποστολής δεδομένων.

Παρότι μοιράζεται το όνομα με το Bluetooth και είναι μέρος του Bluetooth Specification από την έκδοση 4.0, το BLE είναι ανεξάρτητη τεχνολογία η οποία μπορεί να υπάρχει σε ξεχωριστά ή σε συνύπαρξη με το "κλασσικό Bluetooth". Στα περισσότερα σύγχρονα κινητά τηλέφωνα ισχύει το δε, και μάλιστα χρησιμοποιούν την ίδια κεραία, καθώς εκπέμπουν στο ίδιο εύρος συχνοτήτων. Ακολουθούν μερικά χαρακτηριστικά για το Physical Layer και το Data Link layer του BLE.

2.1: BLE Physical Layer

Το BLE είναι ασύρματο πρωτόκολλο το οποίο εκπέμπει στην μπάντα των 2.4 GHz, όπως το Wi-Fi. Το συγκεκριμένο εύρος συχνοτήτων που χρησιμοποιείται είναι από τα 2400 MHz μέχρι τα 2480 MHz, τα οποία χωρίζονται σε 40 κανάλια, 37 εκ των οποίων χρησιμοποιούνται για μεταδώσεις δεδομένων και τα υπόλοιπα για advertising.

Η μέθοδος Διαμόρφωσης σήματος που χρησιμοποιείται είναι το Gaussian FSK[9]. Ομοίως με το συμβατικό FSK, το σήμα διαμορφώνεται ως αυξομειώσεις συχνότητας στο φέρον σήμα. Η διαφορά με το συμβατικό FSK είναι πως οι αυξομειώσεις γίνονται σταδιακά αντί για ακαριαία. Αυτό έχει ως αποτέλεσμα την μείωση των παρεμβολών σε άλλες συχνότητες. Η προβλεπόμενη ισχύς μετάδοσης είναι από -10 έως 10 dBmW Το μέγιστο επιτρεπόμενο bandwidth είναι 1 Mb/s.

2.2: BLE Link Layer

Το Link Layer του πρωτοκόλλου BLE ορίζει μονόπλευρη επικοινωνία(αν και αλλαγή ρόλου είναι εφικτή), κατά την οποία η μία συσκευή δρα αποκλειστικά ως αποστολέας και η άλλη αποκλειστικά ως δέκτης, χωρίς μηνύματα επιβεβαίωσης.

Το BLE ορίζει μόνο έναν τύπο packet. Αυτό το packet μπορεί να χρησιμοποιηθεί είτε για μετάδοση δεδομένων είτε για advertising, ανάλογα με τα περιεχόμενα του.

BLE Packet

Preamble	Access Address	PDU	CRC
1 Byte	4 Bytes	2-257 Bytes	3 Bytes

- Preamble: Μια εναλλασσόμενη ακολουθία υψηλών και χαμηλών bit, χρησιμοποιείται από τον δέκτη για συγχρονισμό και αυτόματη ρύθμιση gain.
- Access Address: 0x8E89BED6 για advertisement packets, τυχαία τιμή για data packets.
- PDU: Είτε Advertising Channel PDU είτε Data Channel PDU, ανάλογα με τον τύπο του packet.
- CRC: Χρησιμοποιείται για εντοπισμό εσφαλμένης λήψης.

Data Channel PDU

Header	Payload
2 Bytes	≥ 255 Bytes

Advertising Channel PDU

Header	Payload
2 Bytes	0-37 Bytes

2.3: Πρωτόκολλα πομπών BLE

Τα Advertisement Packets προορίζονταν αρχικά μόνο για ανακάλυψη συσκευών, μα πλέον υπάρχουν διάφορα πρωτόκολλα που ορίζουν το PDU του advertisement packet τα οποία προβλέπουν αποκλειστική χρήση advertisement packets ως τρόπο μονόδρομης μετάδοσης σύντομων μυνημάτων από συσκευές-πομπούς προς όλες τις συσκευές χρηστών στην εμβέλεια τους.

Ο λόγος για τον οποίο χρειάζονται τέτοιου είδους πρωτόκολλα, πέρα από την μετάδοση στοιχείων βαθμονόμησης της ισχύος του πομπού είναι η αντικαταστησιμότητα. Αν και το MAC address του πομπού γίνεται γνωστό κατά το advertisement, η χρήση του ως "ταυτότητα" θα απαιτούσε να γίνουν αλλαγές στην εφαρμογή που τους εκμεταλλεύεται κάθε φορά που χρειαστεί να αντικατασταθεί κάποιος πομπός. Ακολουθούν τα κυριότερα από αυτά τα πρωτόκολλα.

iBeacon[10]

Δημιουργήθηκε από την Apple στα μέσα του 2013. Η προτιθέμενη χρήση του είναι για εγκαταστάσεις σε χώρους λιανικής οι οποίες επιτρέπουν σε εφαρμογές κινητών να αντιδρούν στην προσέγγιση συγκεκριμένων σημείων ενδιαφέροντος από τον χρήστη. Προβλέπει, μαζί με την μετάδοση στοιχείων που ταυτοποιούν τον πομπό και ένα byte βαθμονόμησης της ισχύος του πομπού, με σκοπό να επιτρέπεται η χρήση του Relative Signal Strength Indication για εκτίμηση της απόστασης από τον πομπό.

Καθώς λόγω ευρείας διαθεσιμότητας πομπών αυτό είναι το πρωτόκολλο που τελικά επέλεξα να χρησιμοποιήσω, ακολουθούν περισσότερες λεπτομέρειες για την λειτουργία του. Το iBeacon Payload ορίζεται ως εξής:

iBeacon Payload

iBeacon Prefix	UUID	Major	Minor	dBmW@1m
9 Bytes	16 Bytes	2 Bytes	2 Bytes	1 Byte

- iBeacon Prefix: Ορίζει στον δέκτη το μήκος του πακέτου, διευκρινίζει τον τύπο του πακέτου και πως η συσκευή που το έστειλε δεν υποστηρίζει σύζευξη

- UUID, Major, Minor: Ταυτοποιούν τον πομπό
- dBmW@1m: Παράγοντας βαθμονόμησης ισχύος πομπού. Εκφράζει το μετρητέο RSSI σε απόσταση ενός μέτρου από τον πομπό, ως two's complement. Χρησιμοποιείται από τον δέκτη στον υπολογισμό απόστασης.

Η προτιθέμενη χρήση του πρωτοκόλλου είναι το UUID να αντιστοιχεί στην εταιρία στην οποία ανήκει μια εγκατάσταση με iBeacons και το Major να είναι κοινό σε όλους τους πομπούς μιας εγκατάστασης ή ενός τμήματος μιας εγκατάστασης, αφήνοντας μόνο το Minor ως διαφοροποιητικό στοιχείο σε ένα σμήνος πομπών. Ο προσδιορισμός της θέσης γίνεται σε τρεις διακριτές κατηγορίες, Άμεση (<1m), Κοντινή (>1, <3m) και Απόμακρη (>3m). Ο τρόπος με τον οποίο προσδιορίζεται αυτή η απόσταση από το iOS δεν έχει γνωστοποιηθεί.

Eddystone[11]

Το Eddystone ήταν ένα παρόμοιο πρωτόκολλο της Google. Ορίζει τα εξής τέσσερα διαφορετικά Payload:

- Eddystone-UID: Παρόμοιο με το iBeacon, περιέχει στοιχεία ταυτοποίησης του πομπού και byte βαθμονόμησης ισχύος.
- Eddystone-EID: Όμοιο με το παραπάνω, μα κρυπτογραφημένο για προστασία από spoofing.
- Eddystone-TLM: Για χρήση από διαχειριστές εγκατάστασης. περιέχει δεδομένα τηλεμετρίας του πομπού
- Eddystone-URL: Περιέχει έναν υπερσύνδεσμο και byte βαθμονόμησης ισχύος.

Αν και διαθέτει περισσότερες δυνατότητες από το iBeacon, η υποστήριξη του έπαψε το 2018 λόγω περιορισμένης υιοθέτησης [12]. Αυτό οδήγησε στο να μην επιλέξω την χρήση του συγκεκριμένου πρωτοκόλλου, παρότι το Eddystone-EID είναι το μόνο πρωτόκολλο που από μόνο του παρέχει ασφάλεια στο δίκτυο.

AltBeacon [13]

Μία ανοιχτού κώδικα εναλλακτική στα παραπάνω πρωτόκολλα, σχεδιασμένη για να εκπληρώνει τις ίδιες λειτουργίες με το iBeacon. Ομοίως με τα παραπάνω πρωτόκολλα, το advertising packet του περιέχει στοιχεία ταυτοποίησης του πομπού και βαθμονόμησης ισχύος μετάδοσης. Δυστυχώς, η υιοθέτηση του από κατασκευαστές πομπών είναι εξαιρετικά περιορισμένη.

2.4: Bluetooth 5[14]

Οι παραπάνω διευκρινίσεις ισχύουν για το BLE σύμφωνα με την έκδοση 4.0 του Bluetooth Specification. Αν και η έκδοση 5.0 υπάρχει και υποστηρίζεται από τις περισσότερες σύγχρονες κινητές συσκευές, δεν χρησιμοποιείται από τους πομπούς που επιλέχθηκαν για την εκπόνηση της παρούσας εργασίας.

Αξιοσημείωτες αλλαγές που εισήγαγε το Bluetooth 5.0 στο BLE περιλαμβάνουν τον διπλασιασμό του Bandwidth, την αύξηση της μέγιστης επιτρεπόμενης ισχύος μετάδοσης και την αύξηση του μέγιστου μήκους των packets.

3: Αξιοποίηση πρωτοκόλλου BLE

3.1: Αξιοποίηση πρωτοκόλλου iBeacon

Κατά την προσέγγιση που προτείνεται, το UUID παράγεται κρυπτογραφικά με τρόπο που περιγράφεται στην επόμενη υποενότητα. Τα Major και Minor αντιμετωπίζονται ως ένα στοιχείο το οποίο ταυτοποιεί τον πομπό, με προτιθέμενη σύμβαση την χρήση διαφορετικού Major ανά δωμάτιο.

Εντός της εφαρμογής, οι πομποί αναπαρίστανται ως αντικείμενα κλάσης iBeacon. Η κλάση αυτή αποθηκεύει τις πληροφορίες του εμπεριέχονται στο advertising packet, καθώς και το RSSI του πομπού κατά την τελευταία ενημέρωση. Επίσης εντός αυτής της κλάσης βρίσκεται και η υλοποίηση της εκτίμησης αποστάσεως από τον πομπό, η οποία γίνεται με τρόπο που περιγράφεται παρακάτω.

Το σύνολο των πομπών που χρησιμοποιούνται σε ένα κτήριο υπάρχει επίσης ως αντικείμενο στην εφαρμογή. Εκεί οι πομποί αποθηκεύονται σε ένα hashmap, στο οποίο το αντικείμενο που εκπροσωπεύει κάθε πομπό καταλογίζεται με βάση το Major και Minor του. Το αντικείμενο αυτό επίσης περιέχει την υλοποίηση του stream ενημέρωσης πομπών το οποίο περιγράφεται παρακάτω, όπως και συναρτήσεις για την αρχικοποίηση του συνόλου και την προσθήκη νέων πομπών στην λίστα.

3.2: Ασφάλιση δικτύου πομπών

Μια πρόκληση στην εφαρμογή BLE πομπών σε δημόσιες εγκαταστάσεις είναι αυτό των κυβερνοεπιθέσεων. Συγκεκριμένα, της κακόβουλης μίμησης πομπών. Αν οι πομποί προτίθενται για πυροδότηση ειδοποιήσεων, η πιθανή ζημια περιορίζεται σε πρόωρη ή εσφαλμένη ενεργοποίηση τους. Στην εφαρμογή που πραγματεύεται η παρούσα εργασία, όμως, ένας κακόβουλος παράγοντας είναι εύκολα ικανός να προκαλέσει δυσλειτουργία μεγάλου μέρους ή ολόκληρης της εγκατάστασης για όλους τους χρήστες.

Για να εκτελέσει μία τέτοια επίθεση, ο επιτιθέμενος αρκεί να προμηθευτεί έναν BLE πομπό και μια εφαρμογή που προβάλλει τα BLE advertising πακέτα που δέχεται η κινητή συσκευή του. Το πρώτο βήμα είναι η ανάγνωση του Major και Minor κάποιου πομπού σε μία εγκατάσταση. Αφού ο επιτιθέμενος τα αντιγράψει στον πομπό του, εκχωρεί ως παράγοντα βαθμονόμησης του πομπού του μία υπερβολικά χαμηλή ποσότητα ενώ θέτει την ισχύ μετάδοσης του στο μέγιστο. Καθώς η εκτίμηση απόστασης βασίζεται στην ορθότητα του στοιχείου βαθμονόμησης και κατ' επέκτασιν στην ορθή ρύθμιση των πομπών, ο κακόβουλος αυτός πομπός θα εμφανίζεται πως είναι πολύ πλησιέστερος από ότι στην πραγματικότητα είναι. Στο μοντέλο των χώρων που περιγράφεται παρακάτω, αυτό μπορεί ως αποτέλεσμα να οδηγήσει στο "κλείδωμα" των θέσεων των χρηστών στο δωμάτιο που βρίσκεται ο μιμούμενος πομπός.

Ο τρόπος προστασίας από μία τέτοια επίθεση είναι η χρήση κρυπτογραφίας. Καθώς μόνο το πρωτόκολλο Eddystone-EID διαθέτει ανάλογη λειτουργία από μόνο του, κρίθηκε σκόπιμο να προταθεί λύση που μπορεί να υλοποιηθεί πάνω στο iBeacon και άλλα πρωτόκολλα.

Στην λύση αυτή, το UUID του advertising packet χρησιμοποιείται ως κρυπτογραφική υπογραφή. Η υπογραφή αυτή παράγεται κρυπτογραφώντας τα Major και Minor του πακέτου σε συνδυασμό με το MAC address του πομπού με ένα Private Key που διαθέτει ο διακομιστής της εγκατάστασης. Για να γίνει επιβεβαίωση, ο πομπός δέχεται ένα Public Key από τον διακομιστή, το οποίο χρησιμοποιεί για να αποκρυπτογραφήσει το UUID του

έκαστου πομπού. Εφόσον το αποτέλεσμα αντιστοιχεί στα Major, Minor και MAC address του πομπού, ο πομπός θεωρείται έμπιστος.

Ένας κατάλληλος αλγόριθμος υπογραφών για την συγκεκριμένη εφαρμογή είναι ο αλγόριθμος Schnorr[15], λόγω του μικρού μεγέθους των παραχθέντων υπογραφών, παράγοντας κρίσιμος καθώς το UUID είναι μόνο 128 bits. Αξίζει όμως να σημειωθεί πως η παρούσα προσέγγιση δεν είναι τέλεια και προτίθεται μόνο προς την αποτροπή περιστασιακών επιθέσεων, καθώς ένας πιο επίμονος παραβάτης χρειάζεται μόνο έναν πομπό που υποστηρίζει MAC address spoofing.

3.3: Σάρωση πομπών

Η σάρωση των πομπών γίνεται με χρήση της βιβλιοθήκης flutter_blue_plus.[16], η οποία αποτελεί fork της πιο δημοφιλούς βιβλιοθήκης flutter_blue[17], σκοπός της οποίας είναι να παρέχει ένα στρώμα συμβατότητας ώστε ο ίδιος κώδικας να μπορεί να εκμεταλλευτεί τις BLE λειτουργίες αμφότερων των λειτουργικών συστημάτων Android και iOS. Η επιλογής της flutter_blue_plus εναντίων της flutter_blue έγινε για λόγους συμβατότητας με τις πιο νέες εκδόσεις του Android.

Στην εφαρμογή, η ίδια η σάρωση υλοποιείται ως μία στοίβα από streams. Το θεμέλιο αυτής της στοίβας είναι το stream σάρωσης του flutter_blue_plus. Η λειτουργία του είναι να επιστρέφει ένα αντικείμενο που αναπαριστά ένα πακέτο advertisement κάθε φορά που η συσκευή το λαμβάνει.

Το επόμενο στοιχείο της στοίβας είναι ένα stream που δρα ως συλλέκτης αποτελεσμάτων. Ο ρόλος του είναι να παράγει μία λίστα από εντοπισθέντες BLE συσκευές και να την επιστρέφει ανά τακτά χρονικά διαστήματα. Η λίστα αυτή υποθέτει ότι το advertisement packet της συσκευής δεν αλλάζει, μα κρατάει όλα τα RSSI που παράγονται από κάθε συσκευή με σκοπό να παραχθεί μέσος όρος για χρήση στην εκτίμηση απόστασης.

Τελευταίο βήμα είναι η επεξεργασία των αποτελεσμάτων. Υπεύθυνο για αυτό είναι ένα τελευταίο stream, το οποίο υλοποιείται ως συνάρτηση με είσοδο τα αποτελέσματα του προηγούμενου. Η λειτουργία του είναι να ξεχωρίζει τα μέλη της προαναφερόμενης λίστας που αντιστοιχούν σε πομπούς που ανήκουν στο κτήριο και να τα χρησιμοποιεί για να ενημερώσει την λίστα με τους πομπούς που διατηρεί η εφαρμογή. Το παρών stream παράγει συμβάντα void τα οποία χρησιμοποιούνται για να πυροδοτήσουν συναρτήσεις που χρείαζεται να τρέχουν σε κάθε ενημέρωση των πομπών, όπως αυτές τις ενημέρωσης της διεπαφής χρήστη και την πιθανή ενεργοποίηση σημείων ενδιαφέροντος.

3.4: Μέθοδος εκτίμησης απόστασης

Στα πρώτα στάδια της εργασίας, έγιναν μετρήσεις για να επιβεβαιωθεί η αξιοπιστία του Received Signal Strength Indication σε τέσσερις διαφορετικές συσκευές. Για τον σκοπό αυτό, παράχθηκε η εφαρμογή OPTORER-Benchmark. Οι συσκευές και ο πομπός τοποθετήθηκαν σε ξύλινο πάτωμα με απόσταση τουλάχιστον 1.5m από κάποιον τοίχο. Ο πομπός ρυθμίστηκε με συχνότητα advertisement 10Hz και ένταση εκπομπής 0dBmW. Έπειτα μετρήθηκαν τα RSSI σε αποστάσεις 1m και 4m. Ακολουθούν τα αποτελέσματα.

Συσκευή	1m			4m		
	ν	$\bar{x}$	s	ν	$\bar{x}$	s
Xiaomi Poco X3 Pro (2021)	178	-69.179	3.415	163	-83.042	4.113
Samsung Galaxy A72 (2021)	52	-71.269	4.132	53	-81.924	4.290
Huawei P9 (2016)	147	-69.755	4.808	128	-84.039	7.089
LG Nexus 5 (2013)	142	-64.492	5.520	148	-73.337	6.211

Αρχικό συμπέρασμα είναι πως σύγχρονες συσκευές παρουσιάζουν μικρότερη τυπική απόκλιση, καθώς πέρα από την τελευταία συσκευή, όλες, αν και διαφορετικών κατασκευαστών παρήγαγαν αποτελέσματα εντός μιας τυπικής απόκλισης τα αποτελέσματα κρίθηκαν μη απαγορευτικά.

Αρχικά, για την εκτίμηση της απόστασης ακολουθήθηκε η προσέγγιση της Android Beacon Library[18], κατά την οποία η απόσταση d ανορθόδοξα εκτιμάται ως

$$r = \frac{s_{rssi}}{s_{1m}} \, d = \begin{cases} r^{10} & r < 1 \\ \alpha * r^{\beta} + \gamma & r \geq 1 \end{cases}$$

Όπου s_{rssi} το RSSI, s_{1m} Η ισχύς σήματος 1 μέτρο από την πηγή, αμφότερα σε dBmW. Οι σταθερές α, β και γ παράγονται μέσω curve fitting και δίνονται ενδεικτικά από την βιβλιοθήκη ως 0.89976, 7.7095 και 0.111.

Τα αποτελέσματα αυτής της προσέγγισης ήταν λιγότερο από ικανοποιητικά. Στην συσκευή μου (Xiaomi Poco X3 Pro) παρατήρησα σημαντική υποτίμηση μικρών (<1.5m) αποστάσεων και εξαιρετικά μεγάλη υπερτίμηση μεγαλύτερων αποστάσεων(>3m).

Η συσκευή με την οποία παρήχθησαν οι παραπάνω σταθερές α, β, γ ήταν η LG Nexus 4, η οποία κυκλοφόρησε στην αγορά εν έτει 2012. Από μόνο του, αυτό θα έχρηζε σκόπιμο τον προσδιορισμό νέων σταθερών. Σε συνδυασμό όμως με την διαφωνία των μεθόδων λήψης απόστασης που παρατηρήθηκε στην βιβλιογραφία και τα ανώμαλα αποτελέσματα των Astafiev et al.[7], κρίθηκε ως απαραίτητη η πλήρης επανεξέταση των εξισώσεων.

Για τον σκοπό αυτό μετρήθηκε η σχέση απόστασης από τον πομπό και RSSI. Ο πομπός τοποθετήθηκε σε εσωτερικό χώρο, σε τοίχο διαστάσεων 245*270 cm, στο γεωμετρικό κέντρο του τοίχου, στην μέση της κάθετης αιχμής που σχηματίζει με άλλο τοίχο και στην κορυφή που σχηματίζουν οι δύο τοίχοι με το ταβάνι. Η απόσταση προς τον απέναντι τοίχο ήταν 750cm. Ακολουθούν οι μετρήσεις που ελήφθησαν.

Μετρήσεις από έδρα

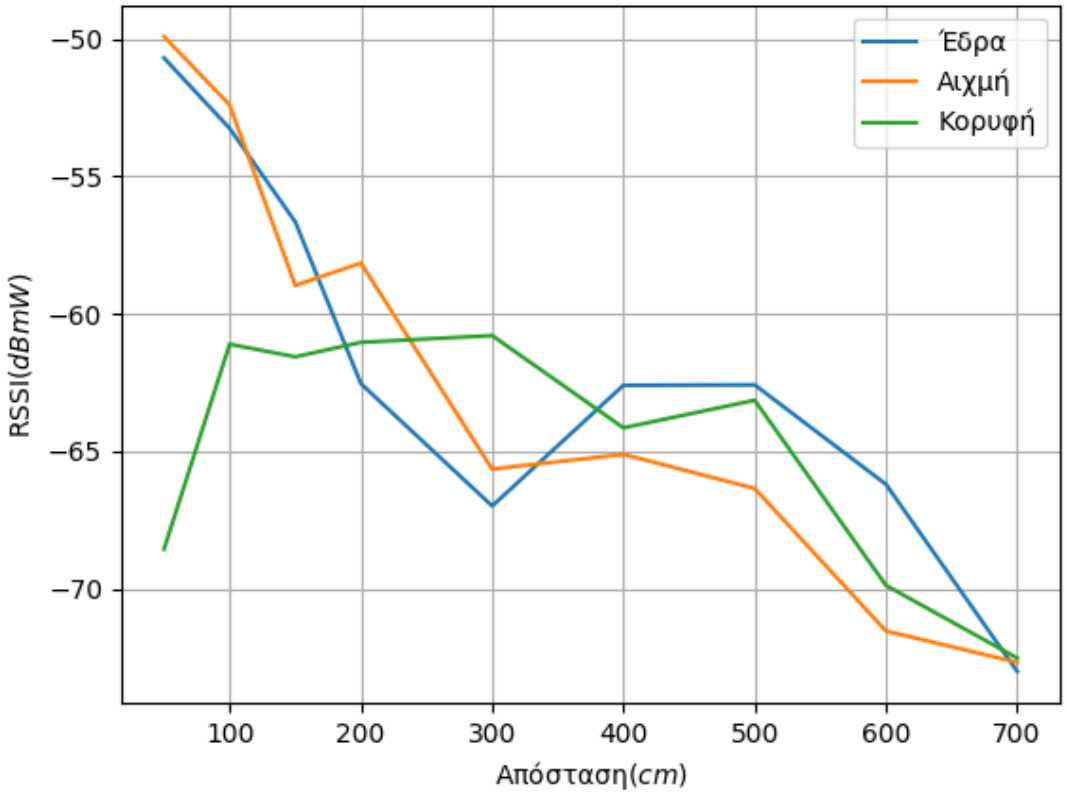
$d(cm)$	v	$R\bar{S}SI$	s
50	280	-50.682	2.798
100	277	-53.245	2.215
150	268	-56.649	2.991
200	280	-62.539	4.176
300	282	-66.992	4.832
400	274	-62.598	2.948
500	280	-62.582	4.131
600	275	-66.203	4.563
700	282	-73.017	4.790

Μετρήσεις από αιχμή

$d(cm)$	v	$R\bar{S}SI$	s
50	281	-49.903	2.709
100	267	-52.393	1.708
150	272	-58.963	4.869
200	277	-58.148	3.251
300	274	-65.649	5.581
400	285	-65.112	5.563
500	279	-66.354	5.070
600	276	-71.543	5.506
700	275	-72.687	5.080

Μετρήσεις από κορυφή

$d(cm)$	v	$R\bar{S}SI$	s
50	272	-68.566	3.661
100	275	-61.101	2.260
150	272	-61.562	1.637
200	280	-61.035	3.209
300	278	-60.787	1.477
400	272	-64.147	4.306
500	285	-63.143	3.658
600	274	-69.883	4.346
700	274	-72.529	4.896



RSSI ως συνάρτηση της απόστασης

Ιδανικά, το RSSI ως συνάρτηση της απόστασης θα ήταν φθίνων σε όλο το εύρος των μετρήσεων, διότι αλλιώς δεν ορίζεται απόσταση ως συνάρτηση RSSI. Και στις τρεις διατάξεις όμως, παρατηρούνται σημεία όπου το RSSI είναι μεγαλύτερο σε σχέση με το προηγούμενο. Αυτό θέτει όριο στην ακρίβεια που μπορεί να επιτευχθεί, αλλά επίσης υποδεικνύει πως η ποιότητα εκτίμησης τοποθεσίας εξαρτάται και από την τοποθεσία των πομπών.

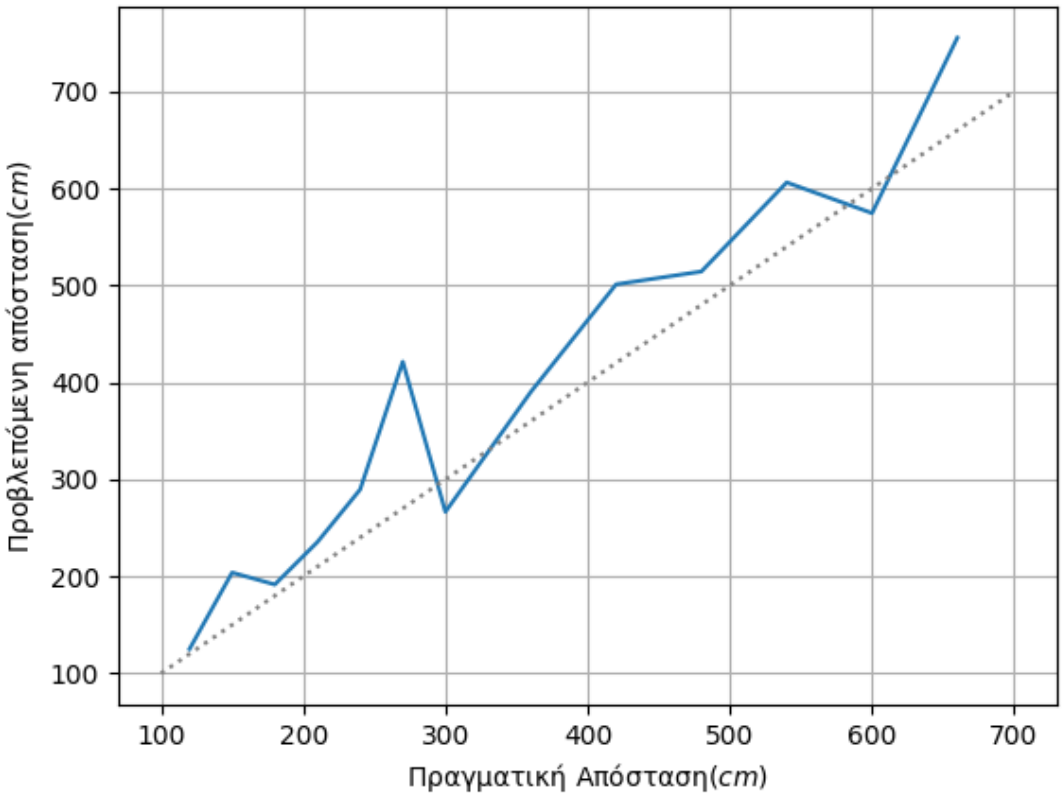
Για να χρησιμοποιηθούν αυτά τα δεδομένα για την εκτίμηση απόστασης, προτείνεται η διαγραφή στοιχείων ή τροποποίηση του πίνακα απόστασης προς RSSI ώστε η σχέση να γίνει σταθερά φθίνουσα. Από τον τροποποιημένο πίνακα, η απόσταση μπορεί να παραχθεί με την μέθοδο της γραμμικής παρεμβολής.

$$d_b = d_\downarrow + (s_{rssi} - s_\downarrow)\frac{d_\uparrow - d_\downarrow}{s_\uparrow - s_\downarrow}$$

Όπου d_b η απόσταση από τον πομπό και s_{rssi} το RSSI. $d_\uparrow$ και $d_\downarrow$ οι αμέσως μεγαλύτερες και μικρότερες γνωστές αποστάσεις. $s_\uparrow$, $s_\downarrow$ αντιστοιχούν στα γνωστά RSSI των αμέσως μεγαλύτερων και μικρότερων γνωστών αποστάσεων. Σε περίπτωση που το RSSI είναι μικρότερο ή μεγαλύτερο από όλες τις γνωστές τιμές, χρησιμοποιούνται οι δύο κοντινότερες.

Για να επιβεβαιωθεί η μέθοδος αυτή, επιλέχθηκε η συμπεριφορά του πομπού στην αιχμή των τοίχων ως βέλτιστη. Οι ποσότητες d = 150, s =-58.963 και d = 300, s =-65.549 αντικαταστάθηκαν από τις d = 140, s= -57.649 και d = 290, s = -64.899, οι οποίες παράχθηκαν μέσω γραμμικής παρεμβολής. Παρήχθησαν καινούριες μετρήσεις και

συγκρίθηκαν οι πραγματικές αποστάσεις με τις προβλεπόμενες. Το σφάλμα που προέκυψε είχε μέσο όρο 16.3% και διάμεσο 12.1%. Τα σφάλματα αυτά είναι αρκετά μικρότερα απο αυτά των Pakanon et al.[5], μα μεγαλύτερα απο αυτά των Astafiev et al.[7], το οποίο ήταν αναμενόμενο, λόγω των πιο ευνοϊκών συνθηκών υπό τις οποίες εκτελέστηκε η δε μελέτη.



Προβλεπόμενη απόσταση σε σχέση με την πραγματική

4: Μοντέλο αναπαράστασης κτηρίου

Για την κατασκευή του μοντέλου αναπαράστασης ενός κτηρίου, ξεκίνησα λαμβάνοντας υπόψιν τις ανάγκες που χρειάζονταν να καλύπτει. Αυτές καθορίστηκαν ως:

- Να είναι χρήσιμο στην εξακρίβωση τοποθεσίας
- Να είναι χρήσιμο για τον αλγόριθμο pathfinding και την παραγωγή οδηγιών προς τον χρήστη.
- Να είναι χρήσιμο για το GUI.
- Να στρέφεται προς την πρακτικότητα, απαιτώντας τον ελάχιστο αριθμό πομπών και έργου κατά την εγκατάσταση που χρειάζεται για να παράξει μια εφαρμογή χρήσιμη στον χρήστη.

Για αυτούς τους λόγους, αποφάσισα την κατάτμηση του κτηρίου σε διακριτούς χώρους. Το μοντέλο του κτηρίου αποτελείται από το σύνολο αυτών των χώρων, το σύνολο των πομπών και το σύνολο των σημείων ενδιαφέροντος. Το σύνολο των πομπών εξηγείται σε ξεχωριστή ενότητα. Αυτά των χώρων και των σημείων ενδιαφέροντος εξηγούνται στις ακόλουθες υποενότητες.

4.1: Χώροι

Αυτοί οι χώροι, με την σειρά τους, μπορούν να έχουν διάφορους τύπους, ανάλογα με τις ανάγκες. Οι παρόντες τύποι χώρων είναι οι εξής:

- Σημειακή αίθουσα. Χαρακτηρίζεται από έναν πομπό και προορίζεται για μικρές και μεσαίες αίθουσες οι οποίες δεν χρήζουν ακρίβεια πέρα από το αν είναι ο χρήστης στην αίθουσα.
- Μονοδιάστατος διάδρομος. Χαρακτηρίζεται από δύο πομπούς, προορίζεται για διαδρόμους στους οποίους επιζητείται η γνώση της θέσης του χρήστη κατά μήκος του διαδρόμου
- Δισδιάστατη αίθουσα. Χαρακτηρίζεται από τρεις πομπούς και η χρήση της είναι για μεγαλύτερους αίθουσες στις οποίες απαιτείται ο προσδιορισμός της τοποθεσίας του χρήστη εντός του χώρου

Η παραπάνω λίστα, φυσικά, μπορεί να επεκταθεί ανάλογα με τις ανάγκες της εφαρμογής. Χώροι με παραπάνω πομπούς, ή ακόμα και χώροι που κάνουνε χρήση fingerprinting για επιπλέον ακρίβεια εντός τους είναι υλοποιήσιμοι, εφόσον μπορούν να παράξουν τις εξής τιμές:

- Το σκορ απόστασης του χρήστη από τον χώρο. Δεν εκφράζει κάποιο συγκεκριμένο μέγεθος, αλλά ο χώρος με το μικρότερο σκορ απόστασης πρέπει να είναι αυτός στον οποίο βρίσκεται ο χρήστης.
- Την θέση του χρήστη, εφόσον ο χρήστης βρίσκεται στον συγκεκριμένο χώρο.
- Το σχήμα τους. Αυτό ορίζεται ως λίστα σημείων τα οποία συνδέονται μεταξύ τους με ευθύγραμμα τμήματα και σχηματίζουν την περίμετρο του χώρου. Προτίθεται για χρήση στο GUI.
- Μια λίστα με τους χώρους με τους οποίους είναι συνδεδεμένοι.
- Την λίστα των πομπών που εμπεριέχουν.
- Τον συντελεστή μεγέθους. Αυτός χρησιμοποιείται στον προσδιορισμό του σκορ απόστασης.
- Ετικέτες που χαρακτηρίζουν τον χώρο ως προς τα πιθανά ενδιαφέροντα στα οποία απευθύνεται. Αυτές χρησιμοποιούνται για την εξατομίκευση της πορείας του χρήστη.

Εκ των οποίων οι τελευταίες πέντε ορίζονται κατά την δημιουργία της εγκατάστασης. Οι τρόποι υπολογισμού του σκορ απόστασης και της θέσης του χρήστη διαφέρουν ανά τύπο χώρου και βρίσκονται στην επόμενη ενότητα.

Στον κώδικα, οι χώροι εκφράζονται ως interfaces (που στην Dart εκφράζονται ώς abstract κλάσεις) οι οποίες υλοποιούνται από κλάσεις που εκφράζουν τις κατηγορίες χώρων.

4.2: Σημεία ενδιαφέροντος

Η λειτουργία των σημείων ενδιαφέροντος προτίθεται για σημεία, σημαδεμένα από κάποιον πομπό, τα οποία όταν προσεγγίζονται από τον χρήστη προκαλούν την απεικόνιση μηνύματος στην συσκευή του. Χαρακτηρίζονται από:

- Τον πομπό τους.
- Το μήνυμα προς απεικόνιση.
- Την απόσταση ενεργοποίησης.

Για να αποφευχθεί η επαναλαμβανόμενη ενεργοποίηση τους, τα σημεία δρουν ως finite state machines με δύο καταστάσεις, ενεργοποιημένη και μη. Η μετάβαση στην ενεργοποιημένη κατάσταση παράγει το προαναφερθέν συμβάν και γίνεται όταν χρήστης βρεθεί πιο κοντά από την απόσταση ενεργοποίησης, ενώ το αντίστροφο γίνεται όταν ο χρήστης βρεθεί πιο μακρυά από το τριπλάσιο της.

Στον κώδικα, το σημείο ενδιαφέροντος είναι κλάση με παραμέτρους που εκφράζουν τα προαναφερθέντα χαρακτηριστικά του. Επιπλέον, διαθέτει δύο μεθόδους, μία για να ελέγχει αν πληρούνται οι συνθήκες ενεργοποίησης και μία για να εκτελεί την ενεργοποίηση. Στις μεθόδους αυτές μπορεί προαιρετικά να περαστεί κάποιο stream στο οποίο θα σταλεί το γεγονός της ενεργοποίησης.

5: Τρόπος εύρεσης τοποθεσίας χρήστη

Όπως αναφέρθηκε προηγουμένως, το κτήριο καταμερίζεται σε χώρους και η αναπαράσταση του κάθε χώρου εντός της εφαρμογής είναι υπεύθυνη για τον προσδιορισμό του σκορ απόστασης του χρήστη σε σχέση με αυτήν και την θέση του χρήστη εντός της. Στο πιο αφηρημένο επίπεδο, ο αλγόριθμος έχει δύο βήματα.

- Προσδιορισμός του σκορ απόστασης για κάθε χώρο
- Υπολογισμός της θέσης του χρήστη από τον χώρο με το ελάχιστο σκορ απόστασης.

Ακολουθούν οι λεπτομέρειες της υλοποίησης του κάθε τύπου χώρου.

5.1: Σημειακή αίθουσα

Το σκορ απόστασης του χρήστη από μία σημειακή αίθουσα s_d υπολογίζεται ως

$$s_d = d_b - m$$

Όπου d_b η απόσταση του χρήστη από τον πομπό και m ο συντελεστής μεγέθους της αίθουσας.

Καθώς οι σημειακές αίθουσες προορίζονται για χώρους χωρίς ανάγκη εξακρίβωσης της θέσης των χρηστών πέρα από το αν βρίσκονται μέσα ή όχι, διαθέτουν έναν πομπό, και η θέση του χρήστη p_u είναι απλά

$$x_u = x_b$$
$$y_u = y_b$$

Όπου p_b η θέση του πομπού της αίθουσας.

5.2: Μονοδιάστατος διάδρομος

Στην περίπτωση ενός μονοδιάστατου διαδρόμου, το σκορ απόστασης s_d του χρήστη υπολογίζεται ως

$$s_d = d_{b1} + d_{b2} - (l + m)$$

Όπου d_{b1} και d_{b2} οι αποστάσεις από τους δύο πομπούς. m όπως παραπάνω, και l το μήκος του διαδρόμου, το οποίο υπολογίζεται με το Πυθαγόρειο θεώρημα από τις συντεταγμένες των δύο πομπών. Ουσιαστικά το σκορ απόστασης του μονοδιάστατου διαδρόμου βγαίνει από το σφάλμα του συνόλου των αποστάσεων του χρήστη από τους δύο πομπούς σε σχέση με το μήκος του.

Καθώς ο μονοδιάστατος διάδρομος έχει μόνο δύο πομπούς, η τοποθέτηση του χρήστη γίνεται πάνω στο ευθύγραμμο τμήμα που ορίζεται από τους πομπούς του.

$$x_u = x_{b2} * f_p + x_{b1} * (1 - f_p)$$
$$y_u = y_{b2} * f_p + y_{b1} * (1 - f_p)$$

Όπου f_p Ο συντελεστής θέσης, που εκφράζει που βρίσκεται ο χρήστης σχετικά με τους δύο πομπούς σε κλίμακα 0-1.

$$f_p = \frac{d_{b1}}{d_{b1} + d_{b2}}$$

Εδώ πρέπει να σημειωθεί πως με τον τρόπο που υπολογίζεται το s_d προκύπτουν ισοποσοτικές καμπύλες σχήματος οβάλ, όχι παραλληλόγραμμου. Στις δοκιμές, αυτό δεν αποτέλεσε πρόβλημα.

5.3: Δισδιάστατη αίθουσα

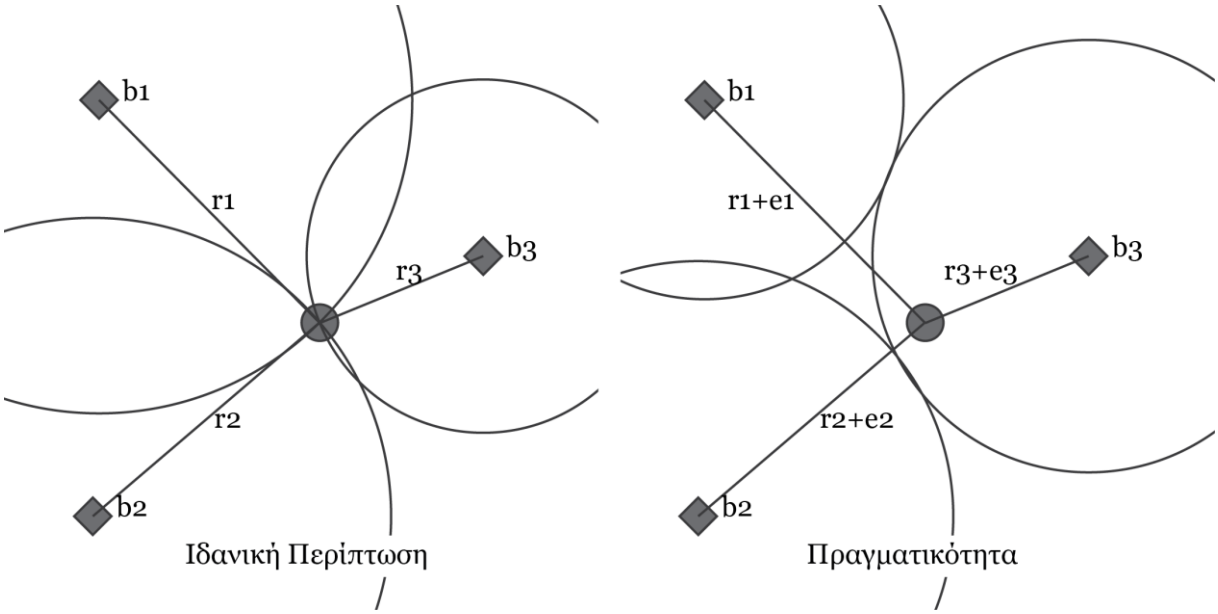
Χαρακτηριζόμενη από τρεις πομπούς, η δισδιάστατη αίθουσα επιτρέπει τον προσδιορισμό της θέσης του χρήστη σε δύο διαστάσεις εντός της. Αυτό γίνεται με την μέθοδο του τριπλευρισμού.

Η αρχή λειτουργίας του τριπλευρισμού είναι πως παρατάσσοντας τρεις πομπούς b_1, b_2, b_3 σε γνωστές και διαφορετικές τοποθεσίες και μετρώντας τις αποστάσεις d_1, d_2, d_3 του χρήστη από κάθε πομπό, υπάρχει μόνο ένα σημείο $[x_u, y_u]$ στο οποίο μπορεί να βρίσκεται ο χρήστης ώστε να ισχύουν και οι τρεις αποστάσεις. Αυτό εκφράζεται ως τρεις κυκλικές εξισώσεις με κέντρα τους πομπούς και ακτίνες τις αποστάσεις τους, των οποίων η κοινή λύση είναι η τοποθεσία του χρήστη.

$$(x_u - x_{b1})^2 + (y_u - y_{b1})^2 = d_{b1}^2$$
$$(x_u - x_{b2})^2 + (y_u - y_{b2})^2 = d_{b2}^2$$
$$(x_u - x_{b3})^2 + (y_u - y_{b3})^2 = d_{b3}^2$$

Αυτό όμως προϋποθέτει μηδενικό σφάλμα στην μέτρηση των αποστάσεων. Στην πραγματικότητα, αυτό δεν υφίσταται ποτέ, οπότε οι αποστάσεις d θεωρούνται άθροισμα μέτρησης και σφάλματος e, με την θέση του χρήστη να βρίσκεται λύνοντας ώστε να ελαχιστοποιεί το άθροισμα των τετραγώνων των σφαλμάτων.

$$e_1^2 + e_2^2 + e_3^2$$



Απεικόνιση της μεθόδου του τριπλευρισμού

Το σκορ απόστασης παράγεται υπολογίζοντας την απόσταση του χρήστη από ένα σημείο $[x_c, y_c]$ το οποίο έχει οριστεί ως το κέντρο της αίθουσας και αφαιρώντας τον παράγοντα μεγέθους της αίθουσας m.

$$s_d = \sqrt{(x_u - x_c)^2 + (y_u - y_c)^2} - m$$

Αν ο διακομιστής της εφαρμογής δεν ορίσει το σημείο $[x_c, y_c]$, ορίζεται ως το μέσο των θέσεων των πομπών.

$$\left[\frac{x_{b1} + x_{b2} + x_{b3}}{3}, \frac{y_{b1} + y_{b2} + y_{b3}}{3}\right]$$

6: Μοντέλο πλοήγησης

Μέρος του έργου στο πλαίσιο του οποίου εκπονήθηκε αυτή η εργασία είναι επίσης και η χρήση της εφαρμογής ως τρόπο πλοήγησης σε χώρους τουριστικού ενδιαφέροντος, π.χ μουσεία. Καθώς οι περισσότεροι επισκέπτες σε τέτοιους χώρους δεν έχουν πλήρη επίγνωση των περιεχομένων τους, το πρόβλημα προς επίλυση δεν είναι απλά η εύρεση διαδρομής ανάμεσα σε δύο γνωστά σημεία, μα η βελτιστοποίηση διαδρομής με βάση τα ενδιαφέροντα του χρήστη.

Η λύση που προτείνεται υποθέτει τα εξής:

- Οι χρήστες ξέρουν τα ενδιαφέροντα τους, όχι τα εκθέματα.
- Οι χρήστες είναι ανοικτοί και σε εκθέματα εκτός του γούστου τους.
- Ο περιοριστικός παράγοντας των χρηστών είναι ο χρόνος.
- Οι διαδρομές δεν χρειάζεται να είναι "τέλειες", μα για την ίδια είσοδο πρέπει να μπορούν να παράγονται πολλές έξοδοι.

Το πρόβλημα μπορεί να χωριστεί σε δύο μέρη, αυτό της αναπαράστασης των ενδιαφερόντων και αυτό της παραγωγής ενός δρομολογίου με βάση αυτά.

Το πρώτο πρόβλημα λύνεται με την ανάθεση ετικετών σε χώρους. Οι ετικέτες αυτές προτίθεται να είναι γενικοί όροι που κατηγοριοποιούν τα περιεχόμενα ενός χώρου, π.χ "Σιδηρουργία", "Ωρολογοποιία", "Αγγειοπλαστική". Για την δημιουργία δρόμου, όλες οι ετικέτες που διαθέτει το κτήριο προβάλλονται στον χρήστη και αυτός επιλέγει αυτές που τον ενδιαφέρουν. Από τον χρήστη επίσης ζητείται η επιθυμητή διάρκεια της παραμονής του στο κτήριο.

Τελικά, η διαδρομή παράγεται με την χρήση εξελικτικού αλγορίθμου, με συνάρτηση καταλληλότητας την εξής:

$$f = p + \sum_{i=1}^{s} \sum_{j=1}^{t} t_{ij} \cdot u_j$$

$$p = l_a - l_d \cdot 0.2 \text{ αν } l_a > l_d$$

$$p = l_d - l_a \cdot 2 \text{ αν } l_d \geq l_a$$

Όπου 1 έως s η αρίθμηση των χώρων στην διαδρομή, 1 έως j η αρίθμηση των ετικετών του κτηρίου. t_{ij} Το βάρος της ετικέτας j στον χώρο i και u_j το βάρος της πού έχει αναθέσει ο χρήστης στην ετικέτα j. p είναι η ποινή εσφαλμένου μήκους και υπολογίζεται ως

Όπου l_d το επιθυμητό μήκος διαδρομής και l_c το παρών μήκος διαδρομής.

Συγκεκριμένα, τα βήματα του αλγορίθμου είναι τα εξής:

1. Έστω μία πορεία α που ξεκινάει στην είσοδο του κτηρίου, περνάει από έναν χώρο με τον οποίο η είσοδος συνδέεται και επιστρέφει ξανά στην είσοδο. Η πορεία αυτή ορίζεται ως ακολουθία χώρων που συνδέονται μεταξύ τους.
2. Αντιγράφοντας την πορεία α, ορίζουμε πορείες α_1 έως α_ν
3. Σε κάθε πορεία εκτός από μία εφαρμόζουμε μία τυχαία μετάλλαξη από τις ακόλουθες:

- Αφαίρεση: Δύο η παραπάνω τυχαίοι συνεχόμενοι χώροι αφαιρούνται από την πορεία. Στην θέση τους εισάγεται η συντομότερη διαδρομή ανάμεσα στους χώρους πριν και μετά από αυτούς που αφαιρέθηκαν.
- Πρόσθεση: Ένας τυχαίος χώρος στο κτήριο εισάγεται σε τυχαίο σημείο στην πορεία. Έπειτα, πριν και μετά από αυτό το χώρο εισάγονται οι συντομότερες διαδρομές ανάμεσα στον προηγούμενο στην διαδρομή χώρο και τον νέο και τον νέο χώρο και τον επόμενο στην διαδρομή αντίστοιχα.

4. Η συνάρτηση καταλληλότητας f εκτελείται για κάθε πορεία. Η πορεία που θα παράξει το υψηλότερο αποτέλεσμα ορίζεται ως πορεία α
5. Αν η εκτέλεση έχει φτάσει ως αυτό το βήμα λιγότερες από κ φορές, επιστροφή στο βήμα 2. Αλλιώς, ο αλγόριθμος τερματίζει, επιστρέφοντας ως έξοδο την πορεία α.

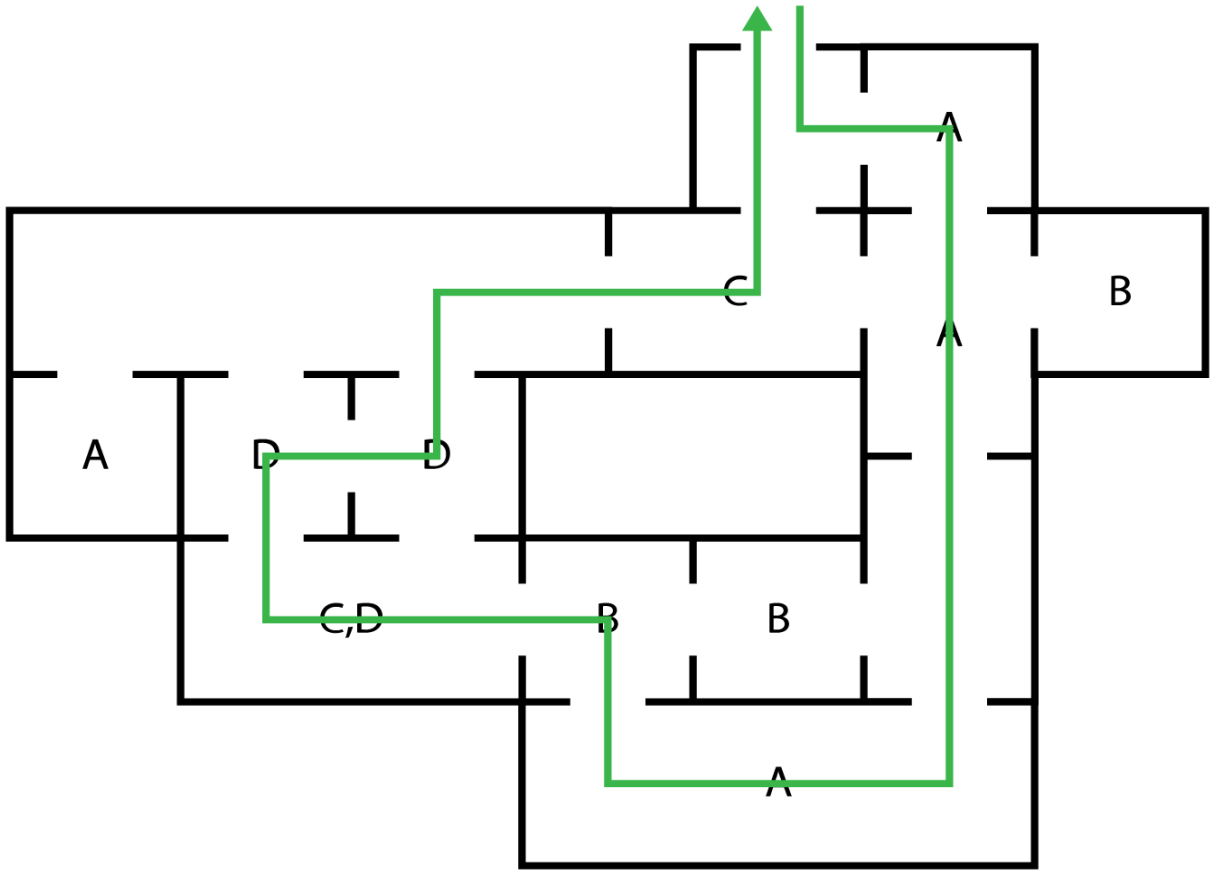
Η εύρεση συντομότερης διαδρομής γίνεται με την χρήση αναδρομής, ορίζοντας την συντομότερη διαδρομή από τον χώρο A στον χώρο B ως

$$p(A, B) = A, p(A_B, B)$$

όπου A_B ο χώρος εκ των χώρων συνδεδεμένων στο A με την μικρότερη απόσταση d ως το B. Η απόσταση $d(A, B)$ ορίζεται ως το πλήθος δωματίων ανάμεσα στο A και το B, επίσης αναδρομικά ως

$$d(A, B) = 1 + d(A_B, B)$$

Η δοκιμή του αλγορίθμου αυτού έγινε με την χρήση αναπαράστασης φανταστικού κτηρίου που φτιάχτηκε ειδικά για τον σκοπό. Το κτήριο αυτό διαθέτει ετικέτες A, B, C και D, κατανεμημένες σε 14 χώρους. Οι δοκιμές έγιναν με $\nu = 1$, $\kappa = 1000$ και $\kappa = 2000$. Και για τις δύο τιμές κ, ο αλγόριθμος παρήγαγε ποιοτικές διαδρομές χωρίς περιττά δωμάτια. Σε υπολογιστή με επεξεργαστή Intel Core i7 3700, έτους κυκλοφορίας 2013, η εκτέλεση για $\kappa = 1000$ ολοκληρωνόταν σε 90ms και 127ms για $\kappa = 2000$. Η χρήση caching κατέβασε τον χρόνο εκτέλεσης σε 65ms και 85ms αντίστοιχα. Ακολουθεί η διαδρομή που παράχθηκε με κ = 1000, l_d = 12, βάρη ετικετών $[A, B, C, D] = [1,0,1,0]$ και αρχικό και τελικό δωμάτιο την είσοδο του χώρου.



Το δοκιμαστικό κτήριο και ο δρόμος που παράχθηκε.

7: Backend

Κατά την συγγραφή αυτού του εγγράφου, η υλοποίηση της εφαρμογής βρίσκεται σε στάδιο proof of concept. Ως εκ τούτου, η αρχικοποίηση της εφαρμογής γίνεται από το αρχείο που περιέχει τις σταθερές της εφαρμογής. Η παρούσα ενότητα περιγράφει τις απαιτήσεις που πρέπει να εκπληρώνει κάποιο backend προκειμένου να μπορεί να χρησιμοποιηθεί από την εφαρμογή και εξερευνεί τρόπους με τους οποίους αυτές θα μπορούσαν να υλοποιηθούν.

7.1: Γνωστοποίηση

Η εφαρμογή πρέπει να γνωρίζει πότε βρίσκεται σε εγκατάσταση που την υποστηρίζει, και εφόσον βρίσκεται σε τέτοια εγκατάσταση, να της γίνεται γνωστή η διεύθυνση του αντίστοιχου διακομιστή. Αυτό μπορεί να γίνει με διάφορους τρόπους, ο καθένας με τα δικά του πλεονεκτήματα και μειονεκτήματα.

QR Code

Κατά την είσοδο του στο κτήριο, ο χρήστης σαρώνει ένα QR code. Αυτό εμπεριέχει ένα URL με το όνομα της εφαρμογής ως πρωτόκολλο και την διεύθυνση του διακομιστή. Αυτή η προσέγγιση έχει το πλεονέκτημα πως δεν χρειάζεται ο χρήστης και ο διακομιστής να βρίσκονται στο ίδιο τοπικό δίκτυο και δεν βασίζεται σε κάποιον κεντρικό κατάλογο. Το μειονέκτημα της προσέγγισης αυτής είναι πως απαιτείται σήμανση εντός της εγκατάστασης, η οποία θα χρειαστεί να επικαιροποιηθεί αν για κάποιο λόγο αλλάξει η διεύθυνση του διακομιστή.

Τοπικός διακομιστής

Κατά αυτή την προσέγγιση, στο τοπικό δίκτυο του κτηρίου χρειάζεται να βρίσκεται ένας διακομιστής ο οποίος ανά τακτά χρονικά διαστήματα ενημερώνει καινούριες συσκευές για την παρουσία του μέσω IP multicast. Αυτό επιτρέπει στην εφαρμογή να είναι έτοιμη για χρήση χωρίς ανάγκη για άλλες δράσεις από τον χρήστη, εφόσον είναι συνδεδεμένος στο ασύρματο δίκτυο του χώρου. Όμως αυτό από μόνο του είναι σημαντικό μειονέκτημα, καθώς εφόσον το δίκτυο χρησιμοποιεί κρυπτογράφηση, η λογική κατάληξη είναι χρήση σήμανσης όπως στην προηγούμενη μέθοδο. Το άλλο μειονέκτημα της συγκεκριμένης προσέγγισης είναι πως εξαναγκάζει την ύπαρξη διακομιστή στο τοπικό δίκτυο της εγκατάστασης.

Κεντρικός κατάλογος

Η τελευταία προσέγγιση είναι η ύπαρξη κεντρικού καταλόγου, στον οποίο βρίσκεται μια λίστα με όλες τις εγκαταστάσεις που υποστηρίζουν την εφαρμογή. Αυτή η προσέγγιση επιφέρει στην εφαρμογή την ικανότητα να ενημερώνει τον χρήστη για εγκαταστάσεις που την χρησιμοποιούν και να τις προωθεί. Επιπλέον, γίνεται εφικτό η εφαρμογή να αρχικοποιείται και να είναι σε ετοιμόχρηστη κατάσταση χωρίς καμία δράση εκ μέρους του χρήστη. Το αρνητικό αυτής της λύσης είναι πως απαιτείται κάποιος κεντρικός κατάλογος, στον οποίο πρέπει να εγγράφονται όλες οι εγκαταστάσεις κατά την ενσωμάτωση τους. Επιπλέον, οποιαδήποτε διακοπή στην διαθεσιμότητα του καταλόγου συνεπάγεται και διακοπή λειτουργίας όλης της εφαρμογής.

Φυσικά, οι παραπάνω λύσεις δεν είναι αποκλείουν η μία την άλλη και η καλύτερη λύση από πλευράς του χρήστη θα ήταν όλες οι παραπάνω λύσεις ως επιλογές.

7.2: Αρχικοποίηση

Αφού ο χρήστης βρεθεί στον χώρο και η εφαρμογή ενημερωθεί για την ύπαρξη του διακομιστή, πρέπει να σταλεί αίτημα για τα δεδομένα του χώρου και ο διακομιστής να απαντήσει. Το αίτημα μπορεί να υλοποιηθεί ως απλό HTTP GET request με την απάντηση ναι παίρνει την μορφή αρχείου JSON που περιέχει την λίστα με τους χώρους του κτηρίου, τα σημεία ενδιαφέροντος και τα στοιχεία βαθμονόμησης των πομπών Η επεξεργασία της απάντησης αυτής μπορεί να γίνει με την χρήση εναλλακτικών constructors για τα αντικείμενα που εκπροσωπεύουν το μοντέλο του κτηρίου.

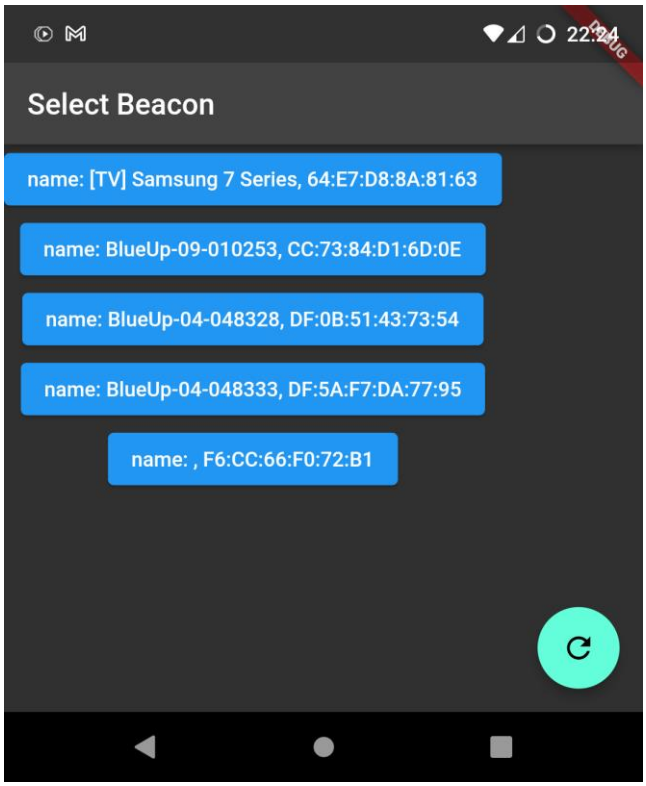
8: Παρουσίαση Εφαρμογών

8.1: Benchmark

Η πρώτη εφαρμογή που παράχθηκε ήταν η optorer_benchmark. Σκοπός της παραγωγής αυτής της εφαρμογής ήταν η δημιουργία ενός συστήματος μέτρησης RSSI και της εξοικείωσης με την γλώσσα Dart και το framework Flutter.

Αρχικά προορίζονταν για αποκλειστική χρήση στις μετρήσεις που παρήγαγαν τον πρώτο πίνακα της ενότητας 4.4. Ως εκ τούτου, Η μόνες της λειτουργίες ήταν η αυτοματοποίηση των μετρήσεων RSSI σε αποστάσεις 1 και 4 μέτρα από κάποιον πομπό, καθώς και η καταγραφή των αποτελεσμάτων σε έγγραφο Google Sheets. Αργότερα, η εμβάθυνση στην μέθοδο εκτίμησης απόστασης έχρησε σκόπιμη την επέκταση της εφαρμογής, καθώς μετρήσεις από περισσότερες αποστάσεις ήταν αναγκαίες. Για αυτό το λόγο, η τελική έκδοση της περιλαμβάνει και λειτουργία ελεύθερων μετρήσεων.

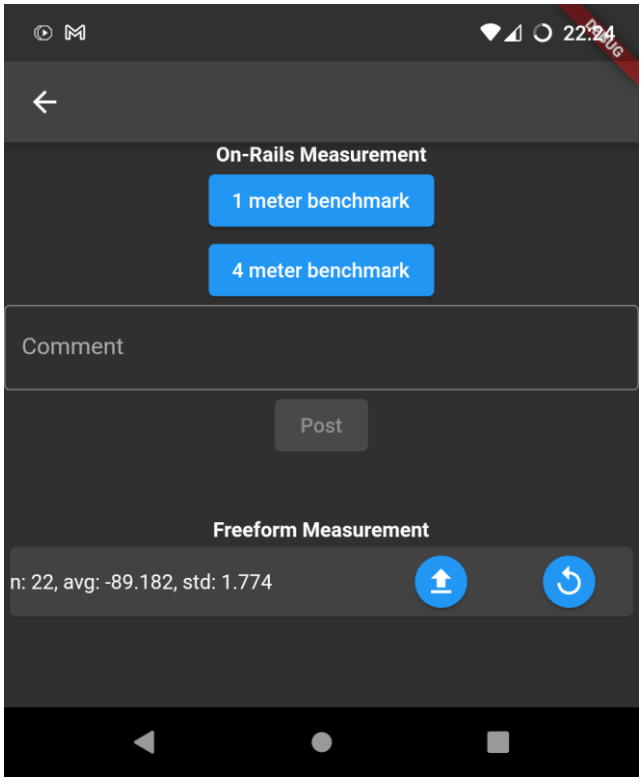
Πρώτη Οθόνη



Όταν ο χρήστης ανοίγει την εφαρμογή, βλέπει την παραπάνω οθόνη. Αρχικά περιλαμβάνει μόνο το κουμπί ανανέωσης στην κάτω δεξιά γωνία. Κατά την εκκίνηση της, η συσκευή ξεκινάει μια σάρωση για συσκευές BLE, διάρκειας ενός δευτερολέπτου. Όταν αυτή ολοκληρωθεί, Οι εντοπισθέντες συσκευές απεικονίζονται ως κουμπιά. Πατώντας κάποιο από αυτά ορίζει την αντίστοιχη συσκευή ως στόχο μέτρησης και πηγαίνει τον χρήστη στην δεύτερη οθόνη. Αν πατηθεί το κουμπί ανανέωσης, η λίστα συσκευών αδειάζει και εκκινείται καινούρια σάρωση.

Δεύτερη Οθόνη

Η δεύτερη οθόνη είναι χωρισμένη σε δύο τμήματα, On-Rails measurement και Freeform measurement.



Το τμήμα On Rails measurement περιλαμβάνει τα κουμπιά 1 meter test, 4 meter test, το πεδίο comment και το πεδίο post. Πατώντας τα κουμπιά 1 meter test ή 4 meter test, για τα επόμενα δέκα δευτερόλεπτα η συσκευή δέχεται τα πακέτα advertising από την συσκευή στόχο, αποθηκεύει τα RSSI του καθενός, και παράγει τον μέσο όρο και την τυπική απόκλιση τους. Για το κάθε κουμπί αυτό μπορεί να γίνει μόνο μία φορά, καθώς όταν πατηθούν απενεργοποιούνται. Ο χρήστης μπορεί προαιρετικά να καταγράψει κάποια παρατήρηση για τις συνθήκες των μετρήσεων στο πεδίο Comment. Το κουμπί Post είναι ανενεργό μέχρι να ολοκληρωθούν οι μετρήσεις ενός και τεσσάρων μέτρων. Όταν πατηθεί, τα αποτελέσματα αυτών των μετρήσεων ανεβαίνουν σε ειδικά διαμορφωμένο spreadsheet στο Google Sheets, μαζί με το σχόλιο που άφησε ο χρήστης, την ώρα που έγιναν οι μετρήσεις και το μοντέλο της συσκευής.

Το δεύτερο τμήμα επιτρέπει στον χρήστη να εκτελεί μη καθοδηγούμενες μετρήσεις. Σε αντίθεση με το προηγούμενο, τα αποτελέσματα απεικονίζονται. Τα κουμπιά που διαθέτει είναι αυτό της σάρωσης και της μεταφόρτωσης των δεδομένων στο αντίστοιχο spreadsheet. Η σάρωση κανονικά διαρκεί δεκαπέντε δευτερόλεπτα, αλλά μπορεί να διακοπεί ξαναπατώντας το κουμπί σάρωσης, το οποίο όσο εκτελείται σάρωση παίρνει την μορφή κουμπιού στοπ.

Σενάρια Χρήσης

Ο χρήστης Α θέλει να ελέγξει αν η ευαισθησία του BLE δέκτη της συσκευής του αλλάζει όταν φορτίζει. Αποφασίζει να χρησιμοποιήσει το καθοδηγούμενο τεστ. Τοποθετεί σε ένα τραπέζι έναν BLE πομπό και δύο σημάδια στο πάτωμα σε αποστάσεις 1m και 4m από τον

πομπό. Ανοίγει την εφαρμογή και επιλέγει στην πρώτη οθόνη τον πομπό του. Πατάει στο σημάδι του ενός μέτρου και κρατώντας ακίνητη την συσκευή του, πατάει το κουμπί 1 meter benchmark και περιμένει να ολοκληρωθεί η μέτρηση. Έπειτα μεταβαίνει στο σημάδι τον τεσσάρων μέτρων και πατάει το 4 meter benchmark. Γράφει στο πεδίο comment πως η συσκευή του δεν φόρτιζε όσο έκανε τα τεστ, πατάει το κουμπί Post και επαναλαμβάνει την διαδικασία με την συσκευή του να φορτίζει. Στο τέλος ανοίγει στον υπολογιστή του την διαδικτυακή εφαρμογή Google Sheets και συγκρίνει τα αποτελέσματα ώστε να παράξει συμπεράσματα.

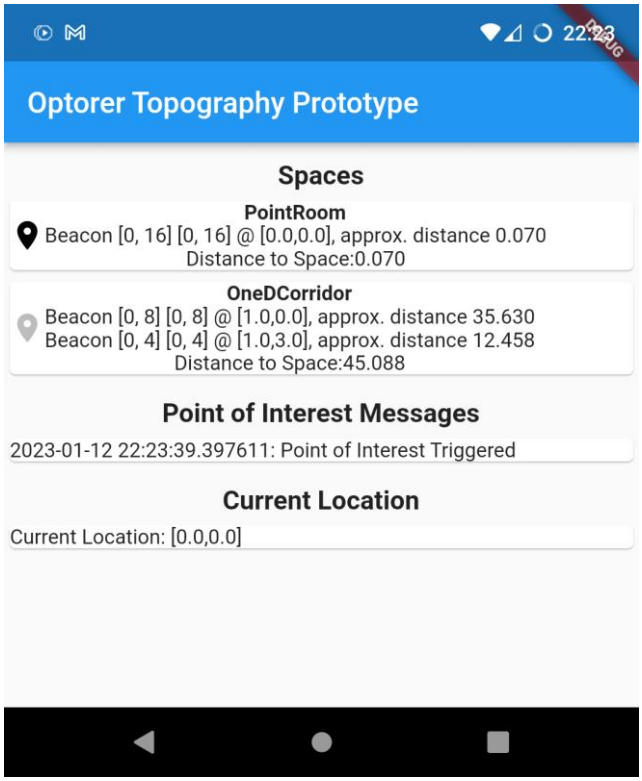
Ο χρήστης Β έχει πρόθεση να δημιουργήσει μια εγκατάσταση που εκμεταλλεύεται BLE πομπούς και χρειάζεται λεπτομερή δεδομένα βαθμονόμησης. Αφού στήσει τον πομπό του και ανοίξει την εφαρμογή, σε κάθε σημείο μέτρησης πατάει το κουμπί σάρωσης του τμήματος Freeform Measurement και μετά το κουμπί μεταφόρτωσης. Όπως και ο χρήστης Α, αφού τελειώσει τις μετρήσεις του, τα αποτελέσματα του τα βρίσκει στο spreadsheet στο Google Sheets.

8.2: Πρωτότυπο τοπογραφίας

Η κύρια εφαρμογή που δημιουργήθηκε ήταν το πρωτότυπο τοπογραφίας. Η πρόθεση κατά την δημιουργία της ήταν να δέχεται τα δεδομένα από τις υλοποιήσεις των μοντέλων που περιγράφηκαν στις προηγούμενες ενότητες αυτού του έγγραφου και να τις απεικονίζει στον χρήστη.

Ένας δευτερεύοντας σκοπός κατά την ανάπτυξή της ήταν η μη επανάληψη των λαθών που έγιναν κατά την ανάπτυξη της προηγούμενης. Συγκεκριμένα, στην εφαρμογή optorer_benchmark, η λογική της εφαρμογής υλοποιήθηκε εντός των widgets που απαρτίζουν το GUI της. Αυτό, ως αποτέλεσμα, παρήγαγε κώδικα που αν και λειτουργικός, είναι δύσκολος στην επέκταση και μη επαναχρησιμοποιήσιμος.

Η παρούσα εφαρμογή αποσυνδέει τα widgets από την λογική της εφαρμογής, όπως προτείνεται από την φιλοσοφία σχεδιασμού BLoC[19]. Κατ’ αυτή τη προσέγγιση, η νοητή σύνδεση ανάμεσα στα λειτουργικά στοιχεία της εφαρμογής και το GUI γίνεται συσχετίζοντας τα διάφορα αντικείμενα της εφαρμογής με widgets φτιαγμένα για να δρουν ως απεικονίσεις τους, με την επικοινωνία ανάμεσα τους να γίνεται με την χρήση streams. Με αυτόν τον διαχωρισμό, τα τμήματα που απαρτίζουν την εφαρμογή γίνονται αρκετά απλούστερα και η διαδικασία της ανάπτυξης ευκολότερη.



Η οθόνη της εφαρμογής είναι χωρισμένη σε τρία τμήματα. Το πρώτο, με τίτλο “Spaces”, είναι η λίστα των πομπών του κτηρίου και των αντίστοιχων πληροφοριών, ομαδοποιημένες κατά τους χώρους στους οποίους ανήκουν. Επιπλέον, με το εικονίδιο τοποθεσίας φαίνεται ο χώρος στον οποίο βρίσκεται ο χρήστης. Το δεύτερο, με τίτλο “Point of Interest Messages” καταλογίζει συμβάντα ενεργοποιήσεων σημείων ενδιαφέροντος. Το τελευταίο δείχνει την παρούσα θέση του χρήστη στο κτήριο.

Σενάριο Χρήσης

Ο χρήστης συντηρεί μια εγκατάσταση βασισμένη στο έργο της παρούσας εργασίας. Ανοίγει την εφαρμογή εντός της εγκατάστασης και χρησιμοποιεί τις ενδείξεις Approximate Distance από κάθε πομπό για να βεβαιωθεί πως δεν λείπει κάποιος και πως οι εκτιμώμενες αποστάσεις είναι επαρκούς ακρίβειας. Έπειτα χρησιμοποιεί την ένδειξη του δωματίου που βρίσκεται ο χρήστης για να ελέγξει πως οι μεταβάσεις ανάμεσα στους χώρους γίνονται στα σωστά σημεία. Καταγράφει τις παρατηρήσεις του, παίρνει τις απαραίτητες ενέργειες (εγκατάσταση νέων πομπών, βαθμονόμιση υπαρχόντων) και επαναλαμβάνει την διάγνωση του για να βεβαιωθεί για την ορθότητα των αλλαγών του.

9: Συμπεράσματα - Επίλογος

Στο πλαίσιο αυτής της εργασίας, εξετάστηκε το πρόβλημα της πλοήγησης σε εσωτερικό χώρο με χρήση πομπών BLE και προτάθηκαν πιθανές λύσεις για δύο υποπροβλήματα του, συγκεκριμένα αυτό του εντοπισμού τοποθεσίας και αυτό της δημιουργίας ενός μοντέλου κτηρίου που εξυπηρετεί τον σκοπό της πλοήγησης σε εσωτερικό χώρο. Επιπλέον, διερευνήθηκε η χρησιμότητα εξελικτικών αλγορίθμων για πρόταση διαδρομών σύμφωνα με τα ενδιαφέροντα του χρήστη.

Η πρώτη διαπίστωση ήταν η αναξιοπιστία των υπαρχουσών προσεγγίσεων για την εκτίμηση της απόστασης μιας κινητής συσκευής από έναν πομπό. Η βιβλιογραφία περιείχε διάφορες εξισώσεις βασισμένες στην εξίσωση FSPL, η οποία εξ' ορισμού υποθέτει κενό χώρο. Οι μετρήσεις μου υπέδειξαν πως η επίτευξη οποιουδήποτε βαθμού ακρίβειας απαιτεί να ληφθεί ως παράγοντας η θέση του πομπού ως προς τους τοίχους του δωματίου. Η προσέγγιση αυτή παρήγαγε μέσο σφάλμα 16.3%, μικρότερο κατά πολύ σε σχέση με την συμβατική προσέγγιση, η χρήση της οποίας σε τριπλευρισμό από προηγούμενη μελέτη παρήγαγε μέσο σφάλμα 7.97 μέτρα σε δωμάτιο μεγέθους 9.0 x 10.2m.[5]

Ως προς την μοντελοποίηση του κτηρίου, προτάθηκαν αρχές που πρέπει να ακολουθούν μοντέλα με παρόμοιο σκοπό και προτάθηκε μία επεκτάσιμη λύση που μπορεί να προσαρμοστεί στις ανάγκες για ακρίβεια της κάθε εγκατάστασης χωρίς να θυσιάζει την πρακτικότητα της για πραγματικές εφαρμογές.

Όσο για την πρόταση πορείας, προτάθηκε η χρήση εξελικτικού αλγορίθμου, η οποία αποδείχτηκε πολλά υποσχόμενη, ξεπερνώντας τις προσδοκίες μου στους τομείς ποιότητας διαδρομών και επιδόσεων.

9.1: Μελλοντικό έργο

Για την εύρεση αποστάσεως, αναγκάστηκα να χρησιμοποιήσω δεδομένα βαθμονόμησης πομπού τα οποία δεν στέλνονται από τον ίδιο τον πομπό, μα από το backend της εφαρμογής. Αυτό αποκαλύπτει ανάγκη για καινούριο πρωτόκολλο πομπού για προσδιορισμό αποστάσεως. Αυτό το πρωτόκολλο θα πρέπει να υποστηρίζει μεταβαλλόμενο αριθμό ισχύων σήματος. Επιπλέον, οι τιμές αυτές χρειάζεται να εκφράζονται με περισσότερη ακρίβεια από αυτήν που παρέχεται από ακέραιους αριθμούς. Μια λύση σε αυτό είναι να εκφράζονται ως 16 bit floating point τιμές. Περισσότερη ακρίβεια δεν χρειάζεται (καθώς στο εύρος τιμών 64-128, οι 16 bit floats έχουν ανάλυση 2^{-4}) αν και λόγω του μεγαλύτερου μεγέθους πακέτου που θεσπίζει το Bluetooth Specification 5, δεν είναι ανέφικτη.

Ένα ανοιχτό ερώτημα είναι αυτό μιας γενικευμένης εξίσωσης αποστάσεως σε σχέση με την ισχύ σήματος. Με παραπάνω μετρήσεις πιθανώς μπορεί να παραχθεί μια εξίσωση που μοντελοποιεί τις αλλαγές στην κατανομή του σήματος σε σχέση με τις αποστάσεις του πομπού από τις γωνίες των τοίχων εντός του δωματίου. Αυτό όμως, στο εγγύς μέλλον, πιθανώς να καταστεί λιγότερο χρήσιμο λόγω των προσθηκών στην έκδοση 5.1 του Bluetooth Specification.

Bluetooth 5.1[20]

Στην έκδοση 5.1 του Bluetooth Specification ορίζεται υποστήριξη για δέκτες οι οποίοι μαζί με το RSSI, μπορούν να παράξουν και εκτίμηση της γωνίας προέλευσης ενός σήματος. Αυτό προϋποθέτει ο δέκτης να διαθέτει πολλαπλές κεραίες σε κάποια απόσταση η μία από την

άλλη και γίνεται μετρώντας την διαφορά φάσης του σήματος ανάμεσα στις κεραίες. Με την χρήση τέτοιων πομπών, ο τρόπος εύρεσης θέσης του χρήστη αλλάζει σημαντικά.

Τέτοιοι δέκτες θα έχουν τουλάχιστον δύο κεραίες, με προτεινόμενη απόσταση ανάμεσα τους $\lambda/2$., Το οποίο για 2.4 GHz ισούται με 6.246cm. Λόγω του απαιτούμενου όγκου, λογικά τέτοιοι δέκτες θα περιορίζονται σε ειδικές συσκευές και δεν θα είναι διαθέσιμοι σε συσκευές χρηστών. Άρα για να γίνει εκμετάλλευση αυτής της τεχνολογίας, οι πομποί στην εγκατάσταση θα πρέπει να δέχονται adveritising από συσκευές χρήστη και είτε να απαντάνε με την γωνία του χρήστη μέσω BLE, είτε να στέλνουν την απάντηση σε κάποιον διακομιστή ο οποίος έπειτα την προωθεί στον χρήστη με την χρήση διαδικτυακών πρωτοκόλλων.

Ανεξάρτητα από τον τρόπο που φτάνουν στην συσκευή του χρήστη, ο τρόπος που η απόσταση d του χρήστη από τον πομπό και η γωνία του θ μετατρέπονται σε θέση είναι

$$[x_u, y_u] = [dcos\theta + x_b, dsin\theta + y_b]$$

Όπου $[x_u, y_u]$ η θέση του χρήστη και $[x_b, y_b]$ η θέση του πομπού. Δεδομένα από πολλαπλούς πομπούς μπορούν να χρησιμοποιηθούν με την λήψη μέσου όρου από τις θέσεις χρήστη που παράγονται. Σε περίπτωση πολλών πομπών (>3), άξια δοκιμής θα ήταν και η λήψη δεύτερου μέσου όρου, με τους πομπούς που έχουν σημαντική απόκλιση από τον πρώτο μέσο όρο να μην συνυπολογίζονται. Καθώς τέτοιοι πομποδέκτες είναι προς το παρών αρκετά δυσεύρετοι, η υλοποίηση αυτής της μεθόδου μένει ως άσκηση προς τον μέλλοντα αναγνώστη.

10: Παράρτημα

10.1: Ακρωνύμια

AP: Access Point

BLE: Bluetooth Low Energy

CRC: Cyclic Redundancy Check

FSPL: Free Space Path Loss

FSK: Frequency Shift Keying

GPS: Global Positioning System

GUI: Graphical User Interface

JSON: JavaScript Object Notation

PDU: Protocol Data Unit

QR: Quick Response

RSS(I): Received Signal Strength(Indication)

RTLS: Real-Time Locating Systems

RTT: Round Trip Time

URL: Uniform Resource Locator

UUID: Universally Unique Identifier

10.2: Κώδικας

Ακολουθεί ο κώδικας του δικτύου πομπών, της δομής του κτηρίου και της εύρεσης διαδρομής.

beacon_tools.dart

```
part of 'building_tools.dart';

///Expresses the set of [IBeacon]s in an installation and their RSSIs
class BeaconSet {
  ///Initializes BeaconSet with a list of [iBeacon]s.
  BeaconSet.fromList(List<IBeacon> list) {
    for (IBeacon i in list) {
      addBeacon(i);
    }
  }

  ///Initializes BeaconSet from a [Building].
  BeaconSet.fromBuilding(Building building) {
    for (Space i in building.spaces.values) {
      for (IBeacon j in i.beacons) {
        if (!iBeacons.containsValue(j)) {
          addBeacon(j);
        }
      }
    }
    for (PointOfInterest i in building.pointsOfInterest) {
      if (!iBeacons.containsValue(i.beacon)) {
```

```
        addBeacon(i.beacon);
      }
    }
  }

  Map<MajorMinor, IBeacon> iBeacons = <MajorMinor, IBeacon>{};

  Stream<void> get updateStream =>
      BleScanner.instance.scan().map(updateBeacons);

  ///Updates every Beacon in the set with new rssi and dbAt1m values from
the
  ///map produced by [BleScanner]'s stream. Returns [this].
  void updateBeacons(
      Map<DeviceIdentifier, BleScannerResult> bleScannerResults) {
    for (IBeacon i in iBeacons.values) {
      i.rssi = null;
    }
    for (BleScannerResult i in bleScannerResults.values) {
      if (i.isIBeacon) {
        if (iBeacons[i.majorMinor] != null) {
          //Assertions safe due to ifs.
          iBeacons[i.majorMinor!]!.dbAt1m =
decodeTwosComplement(i.dbAt1m!);
          iBeacons[i.majorMinor!]!.rssi = i.rssi;
        }
      }
    }
  }

  ///Adds a beacon to the beacon set.
  addBeacon(IBeacon newBeacon) {
    iBeacons[newBeacon.majorMinor] = newBeacon;
  }

  @override
  toString() {
    String string = "";
    for (IBeacon i in iBeacons.values) {
      string += i.toString();
    }
    return string;
  }
}

///Singleton that implements BLE scanning for rangefinding by taking
multiple
///samples to maximise accuracy.
class BleScanner {
  //Singleton constructor, there should only be one instance of this class.
  BleScanner._privateConstructor();
  static final BleScanner instance = BleScanner._privateConstructor();

  FlutterBluePlus flutterBlue = FlutterBluePlus.instance;

  ///Wrapper for flutterblueplus' scan function. Starts scanning
indefinitely
  ///when listened to, stops when cancelled. The returned stream
periodically
```

```
  /// yields a [DeviceIdentifier], [BLEScannerResult] map.The period is
  ///defined by [constants.locationUpdateRate].
  Stream<Map<DeviceIdentifier, BleScannerResult>> scan() async* {
    Map<DeviceIdentifier, BleScannerResult> bleScannerResults =
        <DeviceIdentifier, BleScannerResult>{};
    //test beacons
    await flutterBlue.stopScan();
    DateTime? lastTimestamp;
    Stream flutterBlueScan = flutterBlue.scan(allowDuplicates: true);
    debugPrint("Starting FlutterBluePlus Scan!");

    //runs whenever a BLE advertisement is picked up
    StreamSubscription flutterBlueScanSub = flutterBlueScan.listen((result)
{
      if (result.timeStamp != lastTimestamp) {
        lastTimestamp = result.timeStamp;
        if (bleScannerResults.containsKey(result.device.id)) {
          bleScannerResults[result.device.id]?.update(result);
        } else {
          bleScannerResults[result.device.id] =
              BleScannerResult.fromScanResult(result);
        }
      }
    });

    try {
      //try is used because a finally block is needed.
      while (true) {
        await Future.delayed(constants.locationUpdateRate);
        yield bleScannerResults;
        bleScannerResults = <DeviceIdentifier, BleScannerResult>{};
      }
    } finally {
      //The finally block executes when the subscription is cancelled
      flutterBlueScanSub.cancel();
      flutterBlue.stopScan();
      debugPrint("FlutterBluePlus Scan stopping!");
    }
  }
}

///Expresses a BLE beacon found by [BeaconScanner], constructed using a
///[flutter_blue_plus] [scanResult]. Not necessarily one of our beacons.
class BleScannerResult {
  BleScannerResult.fromScanResult(ScanResult scanResult) {
    id = scanResult.device.id;
    manufacturerData = scanResult.advertisementData.manufacturerData;
    rssiSamples = [scanResult.rssi];
  }

  late DeviceIdentifier id;
  late Map<int, List<int>> manufacturerData;
  late List<int> rssiSamples;

  update(ScanResult scanResult) {
    rssiSamples.add(scanResult.rssi);
    //For some reason, sometimes the beacons don't transmit their iBeacon
    //advertisement. The following lines make it so that if one of these
cases
```

```
    //is used to construct a BleScannerResult the program doesn't
temporarily
    //lose the beacon.
    if (!isIBeacon &&
        scanResult.advertisementData.manufacturerData.containsKey(76)) {
      //safe assertion, we just checked
      manufacturerData[76] =
scanResult.advertisementData.manufacturerData[76]!;
      debugPrint("Corrected bad ScannerResult!");
    }
  }

  ///Returns the average RSSI from rssiSamples
  double get rssi {
    double rssiSum = 0;
    for (int i in rssiSamples) {
      rssiSum += i.toDouble();
    }
    return rssiSum / rssiSamples.length.toDouble();
  }

  List<int>? get iBeaconAd => manufacturerData[76];

  ///Guarantees that [iBeaconAd] will not be null and be 23 bytes long.
  bool get isIBeacon => iBeaconAd != null && iBeaconAd?.length == 23;

  ///If this is a valid [IBeacon], gets major and Minor.
  MajorMinor? get majorMinor {
    List<int>? iBeaconAd = manufacturerData[76];
    if (isIBeacon) {
      //safe assertions due to isIBeacon.
      return MajorMinor(iBeaconAd!.sublist(18, 20), iBeaconAd.sublist(20,
22));
    }
    return null;
  }

  ///Gets dBAt1m as a two's complement that needs decoding before use.
  int? get dbAt1m {
    if (isIBeacon) {
      //safe assertion due to isIBeacon.
      return iBeaconAd![22];
    } else {
      return null;
    }
  }

  @override
  String toString({bool noId = true}) {
    if (noId) {
      return "${rssi.toString()} (n=${rssiSamples.length})";
    } else {
      return "$id: ${rssi.toString()} (n=${rssiSamples.length})";
    }
  }
}

///Describes an iBeacon.
class IBeacon {
```

```
  IBeacon(
      {required this.majorMinor,
      required this.coords,
      this.uuid,
      this.calibration});

  ///Should be used to make sure the beacon is actually ours.
  List<int>? uuid;

  ///Major and minor fields in the iBeacon advertisement.
  ///Unique per beacon per installation.
  final MajorMinor majorMinor;

  List<int> get major => majorMinor.major;
  List<int> get minor => majorMinor.minor;

  ///Beacon calibration data, used for more precise distance calculation.
  BeaconCalibration? calibration;

  ///Used for distance calculation. Value to be obtained while scanning.
Null
  ///if not known
  int? dbAt1m;

  /// Last known RSSI. Null if not seen in latest scan.
  double? rssi;

  /// Coordinates of the beacon.
  final Vector2 coords;

  ///returns x coordinate of the beacon
  double get x => coords.x;

  ///returns y coordinate of the beacon
  double get y => coords.y;

  /// Estimated distance to the beacon in meters. Copied from how android
beacon
  /// library. Calculates iBeacon distance.
  double? get distance {
    //Constants derived from curve fitting, not found by me.
    double a = 0.89976, b = 7.7095, c = 0.111;
    if (rssi == null || dbAt1m == null) {
      return null;
    }
    //Safe assertions, we just checked.
    double ratio = rssi! / dbAt1m!;
    if (ratio < 1.0) {
      return pow(ratio, 10).toDouble();
    } else {
      return a * pow(ratio, b) + c;
    }
  }

  @override
  String toString() {
    double? dist = distance;
    String distString = dist == null ? "?" : dist.toStringAsFixed(3);
    return "Beacon $major $minor @ $coords, approx. distance $distString";
```

```
  }
}

///Stores Beacon calibration data, calculates distance through linear
///interpolation.
class BeaconCalibration {
  BeaconCalibration(Map<double, double> rssiDistancePairs) {
    rssiList.addAll(rssiDistancePairs.keys);
    rssiList.sort();
    for (int i = 0; i < rssiList.length; ++i) {
      distanceList.add(rssiDistancePairs[rssiList[i]]!);
    }
  }
  List<double> rssiList = [];
  List<double> distanceList = [];

  ///Gets the distance from the calibrated beacon using linear
interpolation
  double getDistance(double rssi) {
    //index of the biggest known RSSI that's smaller than the passed RSSI.
    int biggestSmallerIndex;
    double rs, rl, ds, dl;

    //if smaller than smallest known rssi
    if (rssi < rssiList[0]) {
      biggestSmallerIndex = 0;
    } else if (rssi > rssiList.last) {
      biggestSmallerIndex = rssiList.length - 2;
    } else {
      biggestSmallerIndex = 0;
      while (rssiList[biggestSmallerIndex + 1] < rssi) {
        biggestSmallerIndex++;
      }
    }

    ds = distanceList[biggestSmallerIndex];
    dl = distanceList[biggestSmallerIndex + 1];
    rs = rssiList[biggestSmallerIndex];
    rl = rssiList[biggestSmallerIndex + 1];

    return ds + (rssi - rs) * (dl - ds) / (rl - rs);
  }
}

///Expresses the Major and Minor fields in an iBeacon advertisement packet
class MajorMinor {
  MajorMinor(
    List<int> major,
    List<int> minor,
  ) {
    majorMinor[0] = major[0];
    majorMinor[1] = major[1];
    majorMinor[2] = minor[0];
    majorMinor[3] = minor[1];
  }
  final List<int> majorMinor = List<int>.filled(4, 0);

  List<int> get major => [majorMinor[0], majorMinor[1]];
  List<int> get minor => [majorMinor[2], majorMinor[3]];
```

```
  @override
  get hashCode =>
      (major[0] << 24) + (major[1] << 16) + (major[0] << 8) + major[1];

  @override
  bool operator ==(Object other) =>
      (other.runtimeType == MajorMinor && hashCode == other.hashCode);
}

///Converts deciBel-milliWatts to milliWatts.
double dbmToMw(num dBm) => pow(10.0, dBm / 10.0).toDouble();

///Converts milliWatts to deciBel-milliWatts.
double mwToDbm(num milliWatts) => 10 * log(milliWatts) / ln10;

///Decodes an 8 bit two's complement integer.
int decodeTwosComplement(int encoded) =>
    encoded > 127 ? encoded - 256 : encoded;
```

building_tools.dart

```
import 'dart:math';
import 'dart:async';
import 'dart:core';

import 'package:flutter_blue_plus/flutter_blue_plus.dart';
import 'package:vector_math/vector_math.dart' hide Colors;
import 'package:flutter/material.dart';

import 'constants.dart' as constants;

part 'beacon_tools.dart';

///Represents an OPTORER installation. Made up of many [Space]s. Listen to
///updateStream to start fetching the user's location.
class Building {
  Building({
    required this.spaces,
    Map<String, List<String>> connections = const {},
    this.pointsOfInterest = const [],
  }) {
    beaconSet = BeaconSet.fromBuilding(this);

    for (Space i in spaces.values) {
      i.parentBuilding = this;
    }

    for (MapEntry<String, List<String>> i in connections.entries) {
      for (String j in i.value) {
        connectSpaces(i.key, j);
      }
    }
  }

  List<PointOfInterest> pointsOfInterest;

  ///A Map pairing space ids and spaces.
  Map<String, Space> spaces;
```

```
  late BeaconSet beaconSet;

  ///Connects two spaces.
  connectSpaces(String space1Id, String space2Id) {
    if (spaces.containsKey(space1Id) && spaces.containsKey(space2Id)) {
      //safe assertions, we just checked.
      spaces[space1Id]!.connect(spaces[space2Id]!);
    }
  }

  ///Listen to this to start scanning for beacons. Emits whenever
[beaconSet] is
  ///updated. [constants.locationUpdateRate] controls how often
  ///the user's location updates.
  Stream<void> get updateStream =>
      beaconSet.updateStream.map(updatePointsOfInterest);

  ///Emits whenever a Point of Interest is triggered.
  Stream<PointOfInterest> get pointofInterestStream =>
      pointOfInterestStreamController.stream.asBroadcastStream();

  StreamController<PointOfInterest> pointOfInterestStreamController =
      StreamController<PointOfInterest>();

  ///For use when listening to a beacon update stream. Updates points of
  ///interest and sinks the results to [pointOfInterestStreamController]
  void updatePointsOfInterest(void event) {
    for (PointOfInterest i in pointsOfInterest) {
      i.update(poiUpdateController: pointOfInterestStreamController);
    }
  }

  ///Returns the space the user is currently in.
  Space? get currentSpace {
    Space? minSpace;
    for (Space i in spaces.values) {
      if (minSpace == null) {
        minSpace = i;
      } else if (minSpace.distanceScore > i.distanceScore) {
        minSpace = i;
      }
    }
    return minSpace;
  }

  ///Returns the user's estimated position,
  Vector2? get userPosition => currentSpace?.position;

  @override
  toString() {
    String string = "";
    for (Space i in spaces.values) {
      bool isCurrentSpace = i == currentSpace;
      string += "${i.toString()} $isCurrentSpace\n";
    }
    string += "User's Position: $userPosition";
    return string;
  }
}
```

```
///Represents the discrete spaces that make up a [Building]. Can either be
a
///[PointRoom] or a [OneDCorridor].
abstract class Space {
  ///Tags used for navigation as tag-score pairs.
  Map<String, double> tags = {};

  ///A list of connections to other spaces. Must be doubly linked, use
  ///[addconnection] to add new ones.
  List<Space> connections = [];

  ///Bilaterally connects [this] [Space] to the other.
  void connect(Space other) {
    if (!connections.contains(other)) {
      connections.add(other);
      other.connections.add(this);
    }
  }

  ///The [Building] [this] [Space] belongs to. Initialized when [this] is
passed
  ///to a [Building] constructor.
  Building? parentBuilding;

  ///A metric of how close the user is to a [Space]. The space with the
  ///smallest distanceScore should be the space the user is in.
  double get distanceScore;

  ///A representation of how big the room is. Usually subtracted from
  ///[distanceScore]
  double get magnitude;

  List<IBeacon> get beacons;

  ///Returns the position of the user, assuming the user is in [this]
[Space].
  Vector2? get position;

  ///Whether the user is in this space. Returns true if not part of a
building.
  bool get userIsWithin {
    if (parentBuilding == null) {
      debugPrint("userIsWithin was called from a space not inside a
building!");
      return true;
    } else {
      //safe assertion, we just checked.
      return parentBuilding!.currentSpace == this;
    }
  }

  @override
  String toString() {
    String beaconCoords = "";
    for (IBeacon i in beacons) {
      beaconCoords += i.coords.toString();
    }
    return "${runtimeType.toString()} at $beaconCoords:"
```

```
        "ds = $distanceScore pos = $position\n";
  }
}

///A [Space] defined by a single [IBeacon]. The user is either in it or
not.
class PointRoom extends Space {
  PointRoom(
    this.beacon, {
    this.magnitude = 0,
    Map<String, double> tags = const {},
  }) {
    this.tags = tags;
  }
  IBeacon beacon;

  @override
  double get distanceScore {
    return (beacon.distance ?? double.infinity) - magnitude;
  }

  @override
  get beacons {
    return [beacon];
  }

  @override
  double magnitude;

  @override
  get position => beacon.coords;
}

///A [Space] defined by two [IBeacon]s. The user's position in a
OneDCorridor is
///one-dimensional.
class OneDCorridor extends Space {
  OneDCorridor(
    IBeacon beacon1,
    IBeacon beacon2, {
    this.magnitude = 0,
    Map<String, double> tags = const {},
  }) {
    beacons = List.from([beacon1, beacon2], growable: false);
    this.tags = tags;
  }
  @override
  double magnitude;

  @override
  late List<IBeacon> beacons;

  @override
  double get distanceScore {
    if (beacons[0].distance == null || beacons[1].distance == null) {
      return double.infinity;
    } else {
      return beacons[0].distance! + beacons[1].distance! - (length +
magnitude);
```

```
    }
  }

  ///Returns where between the two beacons the user is as a scale from
  ///0 (on beacon 0) to 1 (on beacon 1).
  double? get corridorPosition {
    if (beacons[0].distance != null && beacons[1].distance != null) {
      //safe assertions, we just checked.
      return beacons[0].distance! /
          (beacons[0].distance! + beacons[1].distance!);
    } else if (beacons[0].distance != null) {
      return 0;
    } else if (beacons[1].distance != null) {
      return 1;
    } else {
      return null;
    }
  }

  @override
  get position {
    if (corridorPosition != null) {
      //safe asserions, we just checked
      return beacons[1].coords.scaled(corridorPosition!) +
          beacons[0].coords.scaled(1 - corridorPosition!);
    } else {
      return null;
    }
  }

  ///returns the length of the [Corridor]
  double get length => (beacons[0].coords - beacons[1].coords).length;
}

///A [Space] defined by three [IBeacon]s. The user's position in a TwoDRoom
is
///two-dimensional
class TwoDRoom extends Space {
  TwoDRoom(
    IBeacon beacon1,
    IBeacon beacon2,
    IBeacon beacon3, {
    this.magnitude = 0,
    Vector2? center,
    Map<String, double> tags = const {},
  }) {
    beacons = List.from([beacon1, beacon2, beacon3], growable: false);
    this.tags = tags;
    this.center =
        center ?? (beacon1.coords + beacon2.coords + beacon3.coords) / 3;
  }

  @override
  double magnitude;

  ///The center of the room. Either specified in the constructor or
initialized
  ///as the center point of the three beacons.
  Vector2 center = Vector2(0, 0);
```

```
  @override
  late List<IBeacon> beacons;

  ///Implements trilateration using the equations from Pakanon et al.
  @override
  get position {
    double x1 = beacons[0].x;
    double x2 = beacons[1].x;
    double x3 = beacons[2].x;
    double y1 = beacons[0].y;
    double y2 = beacons[1].y;
    double y3 = beacons[2].y;

    double d1Squared = beacons[0].distance ?? double.infinity;
    d1Squared *= d1Squared;
    double d2Squared = beacons[1].distance ?? double.infinity;
    d2Squared *= d2Squared;
    double d3Squared = beacons[2].distance ?? double.infinity;
    d3Squared *= d3Squared;

    double a1 = 2 * (x2 - x1);
    double a2 = 2 * (x3 - x1);
    double a3 = 2 * (x3 - x2);

    double b1 = 2 * (y2 - y1);
    double b2 = 2 * (y3 - y1);
    double b3 = 2 * (y3 - y2);

    double c1 = d1Squared - d2Squared + x2 * x2 - x1 * x1 + y2 * y2 - y1 *
y1;
    double c2 = d1Squared - d3Squared + x3 * x3 - x1 * x1 + y3 * y3 - y1 *
y1;
    double c3 = d2Squared - d3Squared + x3 * x3 - x2 * x2 + y3 * y3 - y2 *
y2;

    Matrix2 m1 = Matrix2.columns(
        Vector2(a1 * a1 + a2 * a2 + a3 * a3, a1 * b1 + a2 * b2 + a3 * b3),
        Vector2(a1 * b1 + a2 * b2 + a3 * b3, b1 * b1 + b2 * b2 + b3 * b3));
    m1.invert();

    Vector2 m2 =
        Vector2(a1 * c1 + a2 * c2 + a3 * c3, b1 * c1 + b2 * c2 + b3 * c3);
    return m1 * m2;
  }

  @override
  get distanceScore {
    Vector2? userCoords = position;
    if (userCoords == null) {
      return double.infinity;
    }

    Vector2 distance;
    distance = userCoords - center;
    return distance.length - magnitude;
  }
}
```

```
///Represents a point of interest in an installation, a beacon that
triggers
///a notification when approached.
class PointOfInterest {
  PointOfInterest({
    required this.beacon,
    required this.message,
    this.threshold = 1.5,
  });

  ///The beacon [this] is tied to.
  final IBeacon beacon;

  ///The message displayed when triggered
  final String message;

  ///Minimum distance to trigger the notification.
  final double threshold;

  ///Used to not repeatedly trigger notifications
  bool hasTriggered = false;

  ///How far from the point of interest the user has to go before [this]
can
  ///trigger again.
  double get upperThreshold => 3 * threshold;

  ///Checks if the user is in triggering range of the beacon.
  ///If [this] is triggered, returns true and pushes [this] to
  ///poiUpdateController.sink().
  ///Returns false otherwise.
  bool update({StreamController<PointOfInterest>? poiUpdateController}) {
    if (hasTriggered) {
      if ((beacon.distance ?? 0) > upperThreshold) {
        hasTriggered = false;
      }
    } else if ((beacon.distance ?? double.infinity) < threshold) {
      trigger(poiUpdateController: poiUpdateController);
      return true;
    }
    return false;
  }

  ///Triggers the [PointOfInterest] and pushes [this] to poiUpdateStream
  void trigger({StreamController<PointOfInterest>? poiUpdateController}) {
    hasTriggered = true;
    if (poiUpdateController != null) {
      debugPrint("Point of Interest Triggered!");
      poiUpdateController.sink.add(this);
    }
  }
}
```

pathfinding_tools.dart

```
part of 'main.dart';

Random randomInstance = Random();
```

```
///A Course expressed as a series of Spaces. Implements course
manipulation.
class Course {
  Course({
    required this.building,
    required this.router,
    this.courseList = const [],
  });

  ///Creates a new course from an existing course. The building and router
are
  ///copied by reference, the courseList is a shallow copy.
  ///If mutate is set to true, a random mutation will be introduced using
  ///shorten or addDetour. If preserveStartEnd is true, the first and last
  ///rooms will not be affected.
  Course.from(
    Course course, {
    bool mutate = false,
    bool preserveStartEnd = true,
  }) {
    building = course.building;
    router = course.router;
    courseList = List.from(course.courseList);

    if (mutate) {
      int a, b;
      if (preserveStartEnd) {
        a = randomInstance.nextInt(courseList.length - 2) + 1;
        b = randomInstance.nextInt(courseList.length - 2) + 1;
      } else {
        a = randomInstance.nextInt(courseList.length);
        b = randomInstance.nextInt(courseList.length);
      }

      if (randomInstance.nextBool() || courseList.length < 3) {
        //if true, addDetour, else shorten
        int roomIndex =
randomInstance.nextInt(building.spaces.values.length);
        addDetour(a, building.spaces.values.toList()[roomIndex]);
      } else {
        if (a > b) {
          int swap = a;
          a = b;
          b = swap;
        }
        shorten(a, b);
      }
    }
  }

  ///The building being navigated
  late Building building;

  ///The handler for pathfinding calculations and caching.
  late Router router;

  ///The resulting course itself, as a List<Space>
  List<Space> courseList = [];
```

```
  ///Implmenents evolutionary optimization by maximizing fitnessFunction
over
  ///a given amount of iterations.
  void evolutionaryOptimizer(
      int iterations, double Function(Course) fitnessFunction) {
    for (iterations; iterations > 0; iterations--) {
      Course mutated = Course.from(this, mutate: true, preserveStartEnd:
true);
      if (fitnessFunction(this) <= fitnessFunction(mutated)) {
        courseList = mutated.courseList;
      }
    }
  }

  ///Adds a detour to the course before the given index.
  void addDetour(int index, Space targetSpace) {
    courseList.insert(index, targetSpace);
    makeValid();
  }

  ///Replaces the spaces from startIndex to (but not including) endIndex
with
  ///the shortest path from the room befor startIndex to endindex.
  void shorten(int startIndex, int endIndex) {
    courseList.removeRange(startIndex, endIndex);
    makeValid();
  }

  ///Fixes non-connected sequential rooms by splicing in shortest paths.
  void makeValid() {
    for (int i = 0; i < courseList.length - 1; ++i) {
      if (courseList[i] == courseList[i + 1]) {
        courseList.removeAt(i);
      }
      if (!courseList[i].connections.contains(courseList[i + 1])) {
        Course splice = router.shortestPath(
          courseList[i],
          courseList[i + 1],
          omitSpaceA: true,
          omitSpaceB: true,
        );
        courseList = courseList.sublist(0, i + 1) +
            splice.courseList +
            courseList.sublist(i + 1);
      }
    }
  }

  ///Returns whether every [Space] in the course connects to the next. 0
length
  ///courses are considered invalid.
  bool get isValid {
    if (courseList.isEmpty) {
      return false;
    }

    for (int i = 0; i < courseList.length - 1; ++i) {
      if (!courseList[i].connections.contains(courseList[i + 1])) {
        return false;
```

```
      }
    }
    return true;
  }

  operator [](int index) => courseList[index];

  @override
  String toString() => courseList.toString();
}

///Handles shortest path algorithm related tasks. Calculated distances are
///cached.
class Router {
  Router(this.building);
  Building building;

  ///The cached distances. The key is the sum of the sum of the hashcodes
of the
  ///space pair, calculated by [spacePairKey()]
  Map<int, int> _cache = {};

  void purgeCache() {
    _cache = {};
  }

  int cacheKey(spaceA, spaceB) => spaceA.hashCode + spaceB.hashCode;

  ///Returns the shortest path from SpaceA to SpaceB as a list of spaces.
  ///omitSpaceA and omitSpaceB toggle whether the origin and destination
spaces
  ///should be excluded from the resulting list.
  Course shortestPath(
    Space spaceA,
    Space spaceB, {
    bool omitSpaceA = false,
    bool omitSpaceB = false,
  }) {
    List<Space> course = _shortestPath(spaceA, spaceB);
    if (course.isNotEmpty && omitSpaceA) {
      course = course.sublist(1);
    }
    if (course.isNotEmpty && omitSpaceB) {
      course = course.sublist(0, course.length - 1);
    }
    return Course(building: building, router: this, courseList: course);
  }

  List<Space> _shortestPath(Space spaceA, Space spaceB) {
    if (spaceA == spaceB) {
      return [spaceA];
    }

    if (spaceA.connections.contains(spaceB)) {
      return [spaceA, spaceB];
    }

    int? shortestDistance;
    Space? shortestDistanceSpace;
```

```
    int? iDistance;

    for (Space i in spaceA.connections) {
      iDistance = distanceBetween(i, spaceB);
      if (iDistance != null) {
        if (iDistance <= (shortestDistance ?? iDistance)) {
          shortestDistance = iDistance;
          shortestDistanceSpace = i;
        }
      }
    }

    if (shortestDistanceSpace != null) {
      return [spaceA] + _shortestPath(shortestDistanceSpace, spaceB);
    } else {
      return [];
    }
  }

  ///Calculates the distance between two rooms and caches it. Retrieves
known
  ///distances from cache so it can be used multiple times with no
performance
  ///impact. Returns null if no possible path exists.
  int? distanceBetween(Space spaceA, Space spaceB) {
    return _distanceBetween(spaceA, spaceB, []);
  }

  int? _distanceBetween(
    Space spaceA,
    Space spaceB,
    List<Space> excludedSpaces,
  ) {
    //If same room then zero
    if (spaceA == spaceB) {
      return 0;
    }

    //If one room away return 1
    if (spaceA.connections.contains(spaceB)) {
      return 1;
    }
    //If already cached, get cached value
    if (_cache.containsKey(cacheKey(spaceA, spaceB))) {
      return _cache[cacheKey(spaceA, spaceB)]!;
    }

    List<int> distanceTable = [];

    for (Space i in spaceA.connections) {
      if (!excludedSpaces.contains(i)) {
        int? distance = _distanceBetween(i, spaceB, excludedSpaces +
[spaceA]);
        if (distance != null) {
          distanceTable.add(distance + 1);
        }
      }
    }
```

```
    if (distanceTable.isNotEmpty) {
      return _cache[cacheKey(spaceA, spaceB)] = distanceTable.reduce(min);
    }
    return null;
  }
}

///Fitness function, used for optimizing course.
double courseFitness(
  Course course,
  Map<String, double> interestTags,
  int targetLength,
) {
  double fitnessScore = 0;
  for (Space i in course.building.spaces.values) {
    if (course.courseList.contains(i)) {
      for (String j in i.tags.keys) {
        fitnessScore += i.tags[j]! * (interestTags[j] ?? 0);
      }
    }
  }
  if (course.courseList.length > targetLength) {
    fitnessScore -= (course.courseList.length - targetLength) * 2;
  }
  if (course.courseList.length < targetLength) {
    fitnessScore -= (targetLength - course.courseList.length) * 0.1;
  }

  return fitnessScore;
}
```

10.3: Βιβλιογραφία

[1] M. Weiser, 'The Computer for the 21st Century', *Scientific american*, τ. 265, τχ. 3, σσ. 94–105, 1991.

[2] K. Lyytinen και Y. Yoo, 'Ubiquitous computing', *Communications of the ACM*, τ. 45, τχ. 12, σσ. 63–96, 2002.

[3] K. Rose, S. Eldridge, και L. Chapin, 'The internet of things: An overview', *The internet society (ISOC)*, τ. 80, σσ. 1–50, 2015.

[4] A. Raza, L. Lolic, S. Akhter, και M. Liut, 'Comparing and Evaluating Indoor Positioning Techniques', Νοεμβρίου 2021, σσ. 1–8. doi: 10.1109/IPIN51156.2021.9662632.

[5] N. Pakanon, M. Chamchoy, και P. Supanakoon, 'Study on Accuracy of Trilateration Method for Indoor Positioning with BLE Beacons', Ιουλίου 2020, σσ. 1–4. doi: 10.1109/ICEAST50382.2020.9165464.

[6] K. Phutcharoen, M. Chamchoy, και P. Supanakoon, 'Accuracy Study of Indoor Positioning with Bluetooth Low Energy Beacons', Μαρτίου 2020, σσ. 24–27. doi: 10.1109/ECTIDAMTNCON48261.2020.9090691.

[7] A. Astafiev, A. Zhiznyakov, και D. Privezentsev, 'Development of Indoor Positioning Algorithm Based on Bluetooth Low Energy beacons for Building RTLS-Systems', Σεπτεμβρίου 2019, σσ. 1–5. doi: 10.1109/RUSAUTOCON.2019.8867751.

[8] A. Nikoukar, S. Raza, A. Poole, M. Günes, και B. Dezfouli, 'Low-Power Wireless for the Internet of Things: Standards and Applications', *IEEE Access*, τ. PP, σσ. 1–1, Νοεμβρίου 2018, doi: 10.1109/ACCESS.2018.2879189.

[9] D. Sweeney, 'An introduction to bluetooth a standard for short range wireless networking', στο *15th Annual IEEE International ASIC/SOC Conference*, 2002, σσ. 474–475. doi: 10.1109/ASIC.2002.1158106.

[10] 'Apple Developer - iBeacon'. https://developer.apple.com/ibeacon/ (ημερομηνία πρόσβασης 13 Δεκέμβριος 2022).

[11] M. Ashbridge και Contributors, 'Eddystone Protocol Specification', 2015-2016. https://github.com/google/eddystone (ημερομηνία πρόσβασης 13 Δεκέμβριος 2022).

[12] R. Nayak, 'Discontinuing support for Android Nearby Notifications', 2018. https://android-developers.googleblog.com/2018/10/discontinuing-support-for-android.html (ημερομηνία πρόσβασης 13 Δεκέμβριος 2022).

[13] C. Sexton και Contributors, 'AltBeacon Protocol Specification v1.0', 2014-2015. https://github.com/AltBeacon/spec/ (ημερομηνία πρόσβασης 13 Δεκέμβριος 2022).

[14] 'Bluetooth Core Specification 5'. https://www.bluetooth.com/specifications/specs/core-specification-5/

[15] C.-P. Schnorr, 'Efficient identification and signatures for smart cards', στο *Conference on the Theory and Application of Cryptology*, 1989, σσ. 239–252.

[16] C. boskokg, 'Flutter_blue_plus Flutter Package'. https://pub.dev/packages/flutter_blue_plus (ημερομηνία πρόσβασης 17 Δεκέμβριος 2022).

[17] C. Paul DeMarco, 'Flutter_blue Flutter Package'. https://pub.dev/packages/flutter_blue (ημερομηνία πρόσβασης 17 Δεκέμβριος 2022).

[18] C. David G. Young, 'Android Beacon Library', 2014-2021. https://github.com/AltBeacon/android-beacon-library

[19] D. Boelens, 'Reactive Programming - Streams - BLoC', 2018. https://www.didierboelens.com/2018/08/reactive-programming-streams-bloc/

[20] 'Bluetooth Core Specification 5.1'. https://www.bluetooth.com/specifications/specs/core-specification-5-1/ (ημερομηνία πρόσβασης 13 Δεκέμβριος 2022).