ΕΘΝΙΚΟ ΚΑΙ ΚΑΠΟΔΙΣΤΡΙΑΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ

ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Εσωτερική Χωροθέτηση με Χρήση Wi-fi

Άγγελος Γ. Βαλσαμής
Ελευθέριος Π. Σκανδαλέλλης

Επιβλέποντες: **Νάνσυ Αλωνιστιώτη**, Αναπληρώτρια Καθηγήτρια.
Κωνσταντίνος Χατζηκοκολάκης, Διδακτορικός φοιτητής.
Σωκράτης Μπαρμπουνάκης, Διδακτορικός φοιτητής.

ΑΘΗΝΑ

ΙΑΝΟΥΑΡΙΟΣ 2014

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Εσωτερική Χωροθέτηση με Χρήση WI-fi

Άγγελος Γ. Βαλσαμής
Α.Μ.: 1115200800215

Ελευθέριος Π. Σκανδαλέλλης
Α.Μ.: 1115200800153

ΕΠΙΒΛΕΠΟΝΤΕΣ: **Νάνσυ Αλωνιστιώτη**, Αναπληρώτρια Καθηγήτρια.
Κωνσταντίνος Χατζηκοκολάκης, Διδακτορικός φοιτητής.
Σωκράτης Μπαρμπουνάκης, Διδακτορικός φοιτητής.

ΠΕΡΙΛΗΨΗ

Ο σκοπός της παρούσας πτυχιακής είναι η απαρίθμηση των τεχνικών και των συστημάτων που χρησιμοποιούνται για χωροθέτηση (positioning) και πιο συγκεκριμένα εσωτερική χωροθέτηση (indoor positioning), δηλαδή για τον εντοπισμό της τοποθεσίας του χρήστη μέσα σε ένα συγκεκριμένο εσωτερικό χώρο (μουσείο, εμπορικό πολυκατάστημα, νοσοκομείο και λοιπά). Επιπλέον γίνεται μια προσπάθεια υλοποίησης ενός συστήματος πάνω σε λογισμικό android, χρησιμοποιώντας την τεχνολογία ασύρματης πρόσβασης στο διαδίκτυο (Wi-fi).

Αρχικά, γίνεται μια παρουσίαση της σχετικής βιβλιογραφίας, και, πιο συγκεκριμένα των τεχνικών υπολογισμού θέσης που έχουν χρησιμοποιηθεί και των διαφόρων συστημάτων που έχουν αναπτυχθεί με βάση αυτές, καθώς και τα πλεονεκτήματα και περιορισμούς της κάθε τεχνολογίας.

Στη συνέχεια περιγράφονται οι αλγόριθμοι που χρησιμοποιήσαμε για την ανάπτυξη της android εφαρμογής μας, γίνεται μια σύντομη αναφορά στα βασικά συστατικά του android που εκμεταλλευτήκαμε, και αναλύονται τα πειράματα που εκτελέσαμε καθώς και τα αποτελέσματά τους.

Τέλος, γίνεται μια παρουσίαση των βασικών συμπερασμάτων που καταλήξαμε μέσα από τα πειράματα, καταγράφονται οι βασικοί περιορισμοί καθώς και οι παραδοχές που ήταν απαραίτητες και γίνεται αναφορά στα επόμενα βήματα που μπορούν να γίνουν στο συγκεκριμένο τομέα.

ΘΕΜΑΤΙΚΗ ΠΕΡΙΟΧΗ: Εσωτερική Χωροθέτηση

ΛΕΞΕΙΣ ΚΛΕΙΔΙΑ: λαμβανόμενη ισχύς σήματος, σημεία πρόσβασης, ρυθμός ανανέωσης, τεχνολογία wlan,trilateration

ABSTRACT

The purpose of this thesis is to list the techniques and systems used for positioning and more specifically indoor positioning, i.e. to identify the user's location within a particular interior space (museum, mall, hospital and so on). In addition, we try to implement such a system on android, using the Wi-fi technology.

Initially, there is a presentation of the related work, and more specifically the techniques for calculating position that have been used and the various systems that have been developed based on those, as well as the advantages and limitations of each technology.

After that, we describe the algorithms we used for the development of our android application, we briefly describe the main components of the android technology that we exploited, and analyze the experiments performed and their results.

Finally, we present the main conclusions reached from experiments, record the main limitations and assumptions that were necessary, and make reference to the next steps that can be taken in this area.

SUBJECT AREA: Indoor Positioning

KEYWORDS: received signal strength, access points, refresh rate, wlan technology, trilateration

ΕΥΧΑΡΙΣΤΙΕΣ

Θερμές ευχαριστίες οφείλουμε στην καθηγήτρια μας κα. Νάνσυ Αλωνιστιώτη για την πρόταση της να ασχοληθούμε με το συγκεκριμένο θέμα.

Θα θέλαμε επίσης να ευχαριστήσουμε τους κ. Κωνσταντίνο Χατζηκοκολάκη και κ. Σωκράτη Μπαρμπουνάκη, για την διακριτική αλλά καθοριστική τους καθοδήγηση, καθώς και για την άψογη συνεργασία μας κατά τη διάρκεια αυτής της πτυχιακής.

Τέλος, δε μπορούμε να αφήσουμε έξω από τις ευχαριστίες, τις οικογένειες μας και να μην αναγνωρίσουμε τις θυσίες και την υποστήριξη τους όλα αυτά τα χρόνια.

ΠΕΡΙΕΧΟΜΕΝΑ

1. ΕΙΣΑΓΩΓΗ

Όπως όλοι γνωρίζουμε, τα συστήματα χωροθέτησης (positioning) σε εξωτερικούς χώρους έχουν γνωρίσει μεγάλη ανάπτυξη τα τελευταία χρόνια και έχουν σχεδόν τελειοποιηθεί. Με τη χρήση του Global Positioning System (GPS) η ανίχνευση της τοποθεσίας του χρήστη έχει γίνει αρκετά εύκολη διαδικασία, με τη βοήθεια των δορυφόρων. Έτσι, συστήματα GPS συναντάμε πολύ συχνά είτε ενσωματωμένα σε αυτοκίνητα, είτε σε ξεχωριστές συσκευές, και εδώ και μερικά χρόνια έχουν ενσωματωθεί και στα κινητά τηλέφωνα νέας τεχνολογίας (smartphones).

Παραταύτα, ενώ η χωροθέτηση σε εξωτερικούς χώρους είναι πλέον απλή διαδικασία, αποτελεί ακόμα και σήμερα πρόκληση να υλοποιηθούν αποτελεσματικά συστήματα για εσωτερική χωροθέτηση (Indoor Positioning) μέσα σε κτήρια, όπως εμπορικά καταστήματα, νοσοκομεία, μουσεία, αρχαιολογικούς χώρους και πολλά άλλα

[1] [2] Η δυσκολία συνίσταται στο ότι το GPS δεν μπορεί να αναπτυχθεί για χρήση σε εσωτερικούς χώρους, διότι δεν υπάρχει η δυνατότητα οπτικής επαφής μεταξύ των δεκτών και των δορυφόρων σε ένα εσωτερικό περιβάλλον. Οπότε, για να λυθεί το συγκεκριμένο πρόβλημα γίνεται χρήση άλλων εναλλακτικών τεχνικών χωροθέτησης (κεφάλαιο 2).

Η εσωτερική χωροθέτηση στα κτήρια που αναφέραμε είναι αρκετά σημαντική στη σύγχρονη εποχή, αφού θα μας επιτρέπει πλέον να μπορούμε να εντοπίσουμε αυτό που αναζητάμε μέσα στο εκάστοτε κτήριο εύκολα και γρήγορα, χωρίς να χρειάζεται να κάνουμε περιττές διαδρομές, για παράδειγμα όταν ψάχνουμε ένα συγκεκριμένο κατάστημα μέσα σε ένα μεγάλο εμπορικό κέντρο. Επιπλέον είναι αρκετά πρακτική η χρήση της εσωτερικής χωροθέτησης σε περιβάλλοντα πλούσια σε πληροφορίες, όπως μουσεία ή αρχαιολογικούς χώρους, έτσι ώστε να παρέχεται στον χρήστη-επισκέπτη προσωποποιημένη πληροφορία με βάση τα ενδιαφέροντα του.

Στην παρούσα πτυχιακή καλούμαστε να αντιμετωπίσουμε το πρόβλημα της εσωτερικής χωροθέτησης και συγκεκριμένα της εύρεσης της σχετικής θέσης του χρήστη σε ένα εμπορικό κέντρο (mall), βασιζόμενοι στην τεχνολογία ασύρματης πρόσβασης στο διαδίκτυο (Wi-fi) και εργαζόμενοι πάνω σε λογισμικό android, το οποίο υπάρχει πλέον στην μεγάλη πλειοψηφία των κινητών συσκευών.

Πιο συγκεκριμένα, ο στόχος της android εφαρμογής μας είναι η διαπίστωση του πότε ο χρήστης της συσκευής android βρίσκεται σταματημένος μπροστά από τη βιτρίνα κάποιου καταστήματος μέσα στο mall, έτσι ώστε να λαμβάνει στο κινητό του ειδοποίηση ότι βρίσκεται στο συγκεκριμένο κατάστημα, καθώς και, αν θελήσει, να λάβει τις προσφορές αυτού του καταστήματος.

Το πρόβλημα αυτό αντιμετωπίστηκε με χρήση της τεχνολογίας Wi-Fi και ενός δικού μας αλγορίθμου, που βαδίζει στα χνάρια της τεχνικής του trilateration [3] . Επίσης, το δεύτερο σκέλος του αλγορίθμου περιλαμβάνει μια τεχνική έτσι ώστε ο ρυθμός ανανέωσης (refresh rate), δηλαδή το πόσο συχνά το κινητό εκτελεί σάρωση για σημεία πρόσβασης (Access Points-APs), να μην είναι σταθερός, αλλά να μεταβάλλεται ανάλογα αν ο χρήστης κινείται μέσα στο χώρο ή όχι.

Στις παρακάτω ενότητες της πτυχιακής θα ακολουθήσουν τα εξής:

- Στο κεφάλαιο 2 θα κάνουμε μια παρουσίαση των τεχνολογιών και των συστημάτων για indoor positioning που υπάρχουν στη βιβλιογραφία εώς τώρα.
- Στο κεφάλαιο 3 θα κάνουμε μια περιγραφή του αλγορίθμου μας, καταρχήν του πως γίνεται η χωροθέτηση και κατά δεύτερον πως γίνεται η προσαρμογή του ρυθμού ανανέωσης.
- Στο κεφάλαιο 4 θα περιγράψουμε με περισσότερες λεπτομέρειες τις λειτουργίες του Android που χρησιμοποιήσαμε στην ανάπτυξη της εφαρμογής μας.
- Στο κεφάλαιο 5 θα παρουσιάσουμε τα αποτελέσματα που πήραμε από δύο πειράματα που εκτελέσαμε ώστε να δοκιμάσουμε την εφαρμογή μας.
- Στο κεφάλαιο 6 θα κάνουμε μια σύνοψη και παρουσίαση συμπερασμάτων.

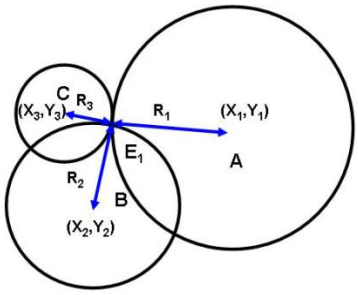
2. ΣΧΕΤΙΚΗ ΒΙΒΛΙΟΓΡΑΦΙΑ

Εξαιτίας της πολυπλοκότητας της *εσωτερικής χωροθέτησης* αλλά και των διαφορετικών σε απαιτήσεις εφαρμογών που την απαιτούν, πολλές τεχνικές έχουν χρησιμοποιηθεί για την ανάπτυξη διαφορετικών συστημάτων. Στο κεφάλαιο 2.1 θα παρουσιάσουμε συνοπτικά τις τεχνικές υπολογισμού θέσης καθώς και τις μεθόδους που τις χρησιμοποιούν έτσι ώστε να γίνει κατανοητός ο τρόπος λειτουργίας των συστημάτων. Στο κεφάλαιο 2.2 θα παρουσιάσουμε διάφορα συστήματα κατανεμημένα με βάση τις τεχνολογίες που χρησιμοποιούν. Τέλος, στο κεφάλαιο 2.3 θα παρουσιάσουμε συγκεντρωτικό πίνακα των τεχνολογιών που περιγράφηκαν, σε συνδυασμό με τα πλεονεκτήματα/μειονεκτήματα της κάθε μίας.

2.1 Τεχνικές και Μέθοδοι Χωροθέτησης

Trilateration

Σύμφωνα με την τεχνική του trilateration, γνωρίζοντας τις γεωγραφικές (ή σχετικές) θέσεις τριών σημείων στο χώρο, μπορούμε να βρούμε τη γεωγραφική (ή σχετική) θέση ενός τέταρτου σημείου αν ξέρουμε επιπλέον τις αποστάσεις μεταξύ των τριών σημείων και του τέταρτου [Εικόνα 2-1-1]. Αυτό γίνεται δυνατό, με την επίλυση του συστήματος των τριών κύκλων με κέντρα τα τρία γνωστά σημεία και ακτίνες τις αποστάσεις τους από το ζητούμενο σημείο. Αξίζει να αναφερθεί πως το trilateration είναι υποπερίπτωση της τεχνικής του multilateration, που χρησιμοποιείται όταν έχουμε παραπάνω από τρία σημεία αναφοράς.



Εικόνα 2-1-1 Τεχνική του trilateration: Γνωρίζοντας τις θέσεις τριών σημείων αναφοράς μπορούμε να υπολογίσουμε τη θέση του τέταρτου σημείου στο χώρο.

Αρκετές μέθοδοι χρησιμοποιούν την τεχνική του trilateration για να βρουν τη θέση ενός αντικειμένου. Κάθε μία έχει συγκεκριμένα πλεονεκτήματα αλλά και περιορισμούς:

- ***Time of Arrival (TOA)***

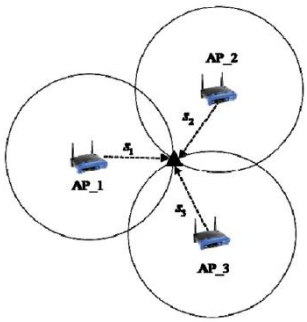
Η TOA χρησιμοποιεί τον χρόνο μεταξύ της εκπομπής σήματος του πομπού και της λήψης αυτού από το δέκτη για να υπολογίσει τη θέση του δέκτη. Πιο συγκεκριμένα, χρησιμοποιώντας την [Εξίσωση 1], όπου u η ταχύτητα (σταθερή), t ο χρόνος και r η απόσταση, αρκεί να βρούμε το χρόνο για να υπολογίσουμε την απόσταση. Στη συνέχεια, μόλις έχουμε τις αποστάσεις μεταξύ τριών ή

περισσοτέρων σημείων και του δέκτη μπορούμε να προχωρήσουμε στην εφαρμογή της τεχνικής του trilateration [Εικόνα 2-1-2].

$$r = u * t$$

Εξίσωση 1: Υπολογισμός απόστασης σε μεθόδους TOA

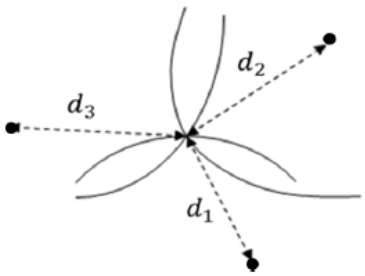
Είναι φανερό πως η μέθοδος αύτη χρειάζεται τον ακριβή χρόνο εκπομπής και λήψης του σήματος. Έτσι είναι απαραίτητος ο συγχρονισμός κάθε πομπού καθώς και κάθε δέκτη ώστε να αποφευχθεί η συμφόρηση. Κάτι τέτοιο είναι πολύ δύσκολο, ειδικά για μικρές αποστάσεις, και η αποτυχία του μπορεί να προκαλέσει λάθη της τάξης των εκατοντάδων μέτρων.



Εικόνα 2-1-2 Μέθοδος του χρόνου άφιξης.

- ***Time Difference of Arrival (TDOA)***

Η μέθοδος αυτή ασχολείται με την διαφορά του χρόνου άφιξης μεταξύ σημάτων παρά με το χρόνο άφιξης ενός. Σύμφωνα με τη μέθοδο, πρέπει να υπολογιστεί η διαφορά του χρόνου άφιξης για κάθε ζευγάρι δεκτών. Κάθε διαφορά τοποθετεί το πομπό σε μια υπερβολή [Εικόνα 2-1-3]. Αρκούν δύο υπερβολές, ώστε να βρούμε τη θέση του πομπού (το κοινό σημείο των υπερβολών). Ο αριθμός των διαθέσιμων δεκτών είναι ανάλογος της αξιοπιστίας της προβλεπόμενης θέσης.

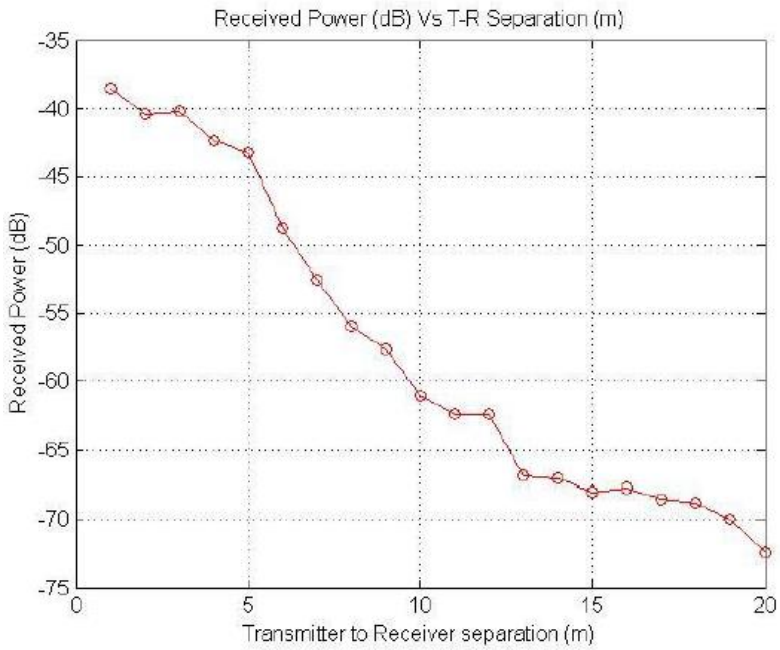


Εικόνα 2-1-3 Μέθοδος της διαφοράς του χρόνου άφιξης μεταξύ σημάτων.

Σε αντίθεση με την *TOA*, η *TDOA* απαιτεί μονάχα το συγχρονισμό των δεκτών, καθιστώντας την πιο εύκολα εφαρμόσιμη όταν οι δέκτες είναι κοντά μεταξύ τους. Το μεγάλο μειονέκτημα είναι το κόστος της καθώς χρειάζεται μεγάλη υποδομή με πολλούς δέκτες για ικανοποιητική ακρίβεια.

- ***Received Signal Strength (RSS)***

Για να μπορέσουμε να καταλάβουμε πλήρως τη συγκεκριμένη μέθοδο θα πρέπει να μελετήσουμε τη σχέση μεταξύ λαμβανόμενου σήματος και απόστασης πομπού. Στο [4] [4] βλέπουμε μια τέτοια προσπάθεια [Εικόνα 2-1-4]. Παρατηρούμε πως με την αύξηση της απόστασης μεταξύ πομπού και δέκτη το λαμβανόμενο σήμα εξασθενεί. Αυτό οφείλεται στη φυσική εξασθένηση του σήματος καθώς αυτό μεταδίδεται σε ευθεία γραμμή στο χώρο, γνωστή και ως απώλεια μετάδοσης. Γνωρίζοντας την αρχική ισχύ εκπομπής του σήματος και τη τελική και χρησιμοποιώντας κατάλληλο αλγόριθμο υπολογισμού απώλειας για εσωτερικό χώρο, μπορούμε να έχουμε μια αρκετά καλή προσέγγιση για την απόσταση μεταξύ πομπού και δέκτη. Στη συνέχεια με τις αποστάσεις που συλλέγουμε μπορούμε να εφαρμόσουμε το trilateration για να υπολογίσουμε τη θέση του δέκτη.



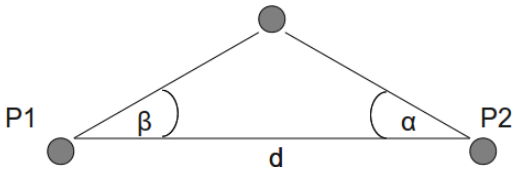
Εικόνα 2-1-4 Σχέση μεταξύ RSS και απόστασης [4] **.**

Αρκετούς περιορισμούς παρουσιάζει και αυτή η μέθοδος. Λόγω της πολυπλοκότητας των εσωτερικών χώρων, όπως η ύπαρξη τοίχων, ανελκυστήρων και επίπλων, περιβαλλοντικές συνθήκες όπως υψηλές θερμοκρασίες ή υγρασία αλλά ακόμα και έντονη ανθρώπινη δραστηριότητα, έχουν επίδραση στην απώλεια μετάδοσης. Οι αλγόριθμοι υπολογισμού απώλειας για εσωτερικούς χώρους προσπαθούν να συμπεριλάβουν αυτούς τους παράγοντες, ωστόσο είναι τόσο μεγάλη η πολυπλοκότητα που αρκετές φορές δεν είναι ικανοί να παράγουν ικανοποιητικά αποτελέσματα. Η κύρια πρόκληση

της συγκεκριμένης μεθόδου είναι ακριβώς η υψηλή εξάρτησή της από τις αλλαγές του περιβάλλοντος.

Triangulation

Η τεχνική του triangulation, σε αντίθεση με αυτή του trilateration , δεν ασχολείται με αποστάσεις, αλλά με γωνίες. Πιο συγκεκριμένα, δοθέντων δύο σημείων και γνωρίζοντας (ή υπολογίζοντας) την μεταξύ τους απόσταση, μπορούμε να βρούμε τη θέση ενός τρίτου σημείου αν γνωρίζουμε επιπλέον τις γωνίες μεταξύ των δύο σημείων και του τρίτου.

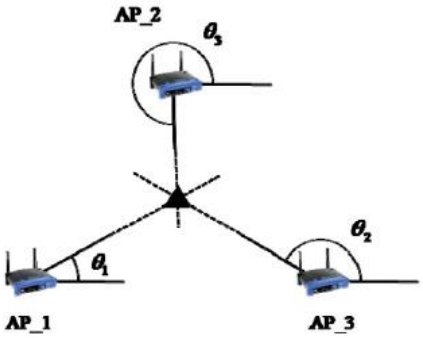


Εικόνα 2-1-5 Τεχνική του triangulation: Γνωρίζοντας τις θέσεις δύο σημείων αναφοράς μπορούμε να υπολογίσουμε τη θέση του τρίτου σημείου στο χώρο.

Συγκριτικά με τη μέθοδο του trilateration, έχει το πλεονέκτημα πως χρειάζονται μόνο δύο σημεία για να μπορέσουμε να υπολογίσουμε τη θέση του άγνωστου σημείου στο χώρο. Ωστόσο, όταν το σημείο προς ανίχνευση βρίσκεται μακριά, μικρά αναπόφευκτα λάθη στους υπολογισμούς των γωνιών, μπορούν να προκαλέσουν σοβαρά λάθη στην ακρίβεια αυτής της τεχνικής.

- ***Angle of Arrival (AoA)***

Η Angle of Arrival απαιτεί τουλάχιστον δύο δέκτες σήματος. Ο προς-ανίχνευση πομπός, εκπέμπει το σήμα του το οποίο καταλήγει στους δέκτες. Αυτοί ανιχνεύουν τη γωνία με την οποία ήρθε το σήμα και στη συνέχεια με την τεχνική του Triangulation μπορούν να υπολογίσουν τη θέση του πομπού.



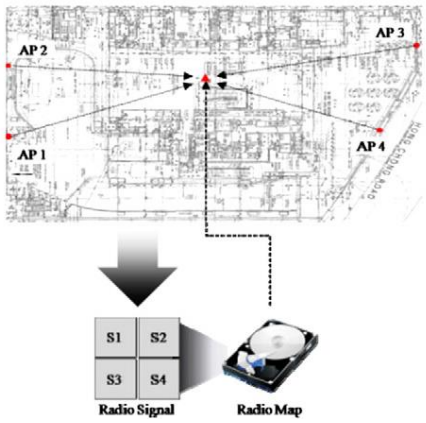
Εικόνα 2-1-6 Μέθοδος της γωνίας άφιξης.

Για την επιτυχή ανίχνευση της γωνίας άφιξης από τους δέκτες, κάθε δέκτης πρέπει να έχει επιπλέον εξοπλισμό. Το συγκεκριμένο υλικό είναι πολύπλοκό και ο εφοδιασμός κάθε δέκτη με αυτό είναι αρκετά δαπανηρός.

Fingerprinting

Η τεχνική του fingerprint είναι αρκετά διαδεδομένη τα τελευταία χρόνια, παρά το μεγάλο της κόστος, καθώς είναι σε θέση να δώσει εξαιρετικά αποτελέσματα στα συστήματα εσωτερικής χωροθέτησης που την χρησιμοποιούν.

Στην τεχνική αυτή έχουμε δύο φάσεις, την offline και την online. Στην offline φάση, συλλέγουμε χρήσιμα στοιχεία από διαφορετικά σημεία του χώρου τα οποία αποθηκεύουμε σε μια βάση [Εικόνα 2-1-7]. Αργότερα, στην online φάση, τα στοιχεία που συλλέγονται από το προς ανίχνευση αντικείμενο, συγκρίνονται με τα στοιχεία της βάσης και επιλέγεται το πιο κοντινό στοιχείο από τη βάση δεδομένων. Η θέση στην οποία βρέθηκε το στοιχείο αυτό θεωρείται αρκετά ασφαλής προσέγγιση της θέσης του προς ανίχνευση αντικειμένου. Πολλές φορές, για την αποφυγή λάθους πρόβλεψης λόγω στοιχείων αποθηκευμένων στη βάση με μεγάλο θόρυβο, είναι αναγκαία είτε η χρήση αλγορίθμων κατά την σύγκριση στην online φάση (όπως των k-κοντινότερων γειτόνων)[5] [5] , είτε το φιλτράρισμα των στοιχείων στην offline φάση.



Εικόνα 2-1-7 Δείγμα της offline φάσης. Οι χρήσιμες πληροφορίες από τα τέσσερα APs αποθηκεύονται στη βάση δεδομένων.

Κάποια από τα πλεονεκτήματα αυτής της τεχνικής είναι πως δεν απαιτεί κάποια επιπλέον υποδομή καθώς χρησιμοποιεί την υπάρχουσα. Επιπλέον, με την προϋπόθεση πως έχουμε αρκετές πληροφορίες από την offline φάση, η ακρίβεια είναι, στη γενική περίπτωση, σαφώς βελτιωμένη από τις προαναφερθείσες τεχνικές. Παρ' όλα αυτά, δεν στερείται μειονεκτημάτων. Το σημαντικότερο είναι το πρόβλημα της αμφιβολίας (*ambiguity problem*). Η τεχνική δεν μπορεί να ξεχωρίσει αποτελεσματικά σημεία τα οποία έχουν συλλέξει ίδιες (ή παρόμοιες) πληροφορίες κατά την offline φάση. Μπορούμε να μετριάσουμε αυτό το πρόβλημα είτε με την εισαγωγή όσο το δυνατό περισσότερων access point ώστε να μπορούμε να διακρίνουμε περισσότερες αλλαγές στα διαφορετικά σημεία, είτε χρησιμοποιώντας πληροφορίες όπως την ταχύτητα ή το ιστορικό από τις προηγούμενες θέσεις του προς ανίχνευση αντικειμένου, ώστε να είμαστε σε θέση να αποκλείσουμε ορισμένες υποψήφιες θέσεις της βάσης δεδομένων.

- ***Signal Fingerprinting***

Η μέθοδος του Signal Fingerprint αποθηκεύει κατά την offline φάση ένα δισδιάστατο πίνακα με τις τιμές της δύναμης των σημάτων που λαμβάνει από τα γειτονικά APs. Αργότερα, κατά την online φάση, οι τιμές των σημάτων από τον προς ανίχνευση δέκτη συγκρίνονται με τη βάση για να βρεθεί το πιο «κοντινό» σημείο. Πολλές παραλλαγές αυτής της κλασσικής μεθόδου έχουν επιχειρηθεί από ερευνητές [6] [7] [8] .

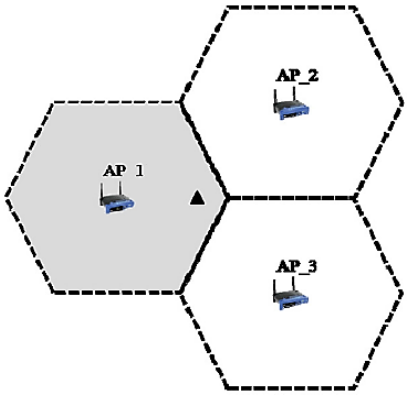
Proximity sensing

Η τεχνική του proximity sensing δεν είναι σε θέση να δώσει προσέγγιση της ακριβής θέσης του προς-ανίχνευση αντικειμένου αλλά απλώς μια προσέγγιση του χώρου μέσα στο οποίο βρίσκεται αυτό. Μερικές εφαρμογές δύναται να απαιτούν μόνο αυτό από τη μέθοδο εσωτερικής χωροθέτησης.

Ο τρόπος με τον οποίο επιτυγχάνεται η προσέγγιση αυτή είναι αρκετά απλός. Απαιτείται η τοποθέτηση δεκτών στο χώρο στον οποίο η εφαρμογή χρειάζεται την εσωτερική χωροθέτηση. Όταν κάποιος δέκτης λάβει μήνυμα, αυτό συνεπάγεται πως ο πομπός βρίσκεται στο εύρος από το οποίο μπορεί να λαμβάνει μηνύματα ο δέκτης. Τα όρια αυτού του εύρους είναι μια καλή προσέγγιση του χώρου στον οποίο βρίσκεται ο πομπός.

- ***Cell of Origin (COO)***

Η μέθοδος Cell of Origin είναι η πιο γνωστή μέθοδος για proximity sensing. Σύμφωνα με αυτή, το Access Point με το οποίο συνδέεται το κινητό προς ανίχνευση επιλέγεται ως το πιο «κοντινό» σε αυτό. Το κινητό τοποθετείται μέσα στο εύρος λήψης του συγκεκριμένου Access Point. Η ακρίβεια αυτής της μεθόδου εξαρτάται άμεσα από το εύρος και τον πληθικό αριθμό των Access Point· όσο μικρότερο το εύρος τόσο πιο ακριβής η προσέγγιση και όσο μεγαλύτερος ο αριθμός των Access Point τόσο πιο μεγάλη κάλυψη του χώρου και άρα πιο αξιόπιστα αποτελέσματα.



Εικόνα 2-1-8 Μέθοδος Cell-of-Origin.

Ο βασικός περιορισμός του Cell of Origin είναι ακριβώς η εξάρτησή του από τον αριθμό των σταθμών λήψης. Στην επαρχία ένα κελί μπορεί καλύπτει χιλιόμετρα, ενώ σε αστικές περιοχές μόλις μερικά μέτρα. Επιπλέον, σύμφωνα με μετρήσεις του [9] το κινητό-πομπόςσε 43% των περιπτώσεων δεν

κατάφερε να συνδεθεί με το κοντινότερο γεωγραφικά σταθμό, με αποτέλεσμα να μην υφίσταται ως μέγιστο σφάλμα το εύρος του κοντινότερου σταθμού.

Vision Analysis

Η τεχνική του Visual Analysis εξάγει πληροφορίες για την θέση από εικόνες. Μια η περισσότερες κάμερες τοποθετούνται προσεκτικά στο χώρο προς ανίχνευση για να μπορούν να καλύπτουν όλη την περιοχή. Φωτογραφίες αναλύονται περιοδικά ώστε να ανιχνευθούν σε αυτές οι επιθυμητοί στόχοι (αντικείμενα ή άνθρωποι). Στη συνέχεια με τη βοήθεια προ-αποθηκευμένης πληροφορίας για το χώρο, μπορεί να υπολογιστεί η θέση του στόχου.

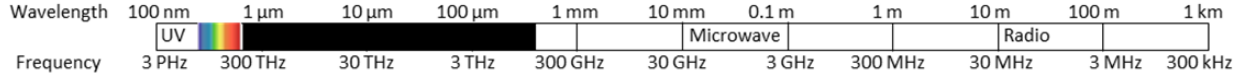
Με την παραπάνω τεχνική, ο χρήστης δεν είναι υποχρεωμένος να κρατάει πάνω του οποιαδήποτε συσκευή. Επίσης, επιπλέον πληροφορίες μπορούν να εξαχθούν εκτός από τη θέση -του στόχου, όπως η ανθρώπινη δραστηριότητα ή η γωνία του αντικείμενου προς ανίχνευση, που μπορούν να είναι βοηθητικές ή αναγκαίες σε πληθώρα εφαρμογών.

2.2 Τεχνολογίες και Συστήματα Χωροθέτησης

Infrared (IR)-Based συστήματα

Τα συστήματα που χρησιμοποιούν υπέρυθρες ακτίνες για εσωτερική χωροθέτηση είναι αρκετά διαδεδομένα καθώς η συγκεκριμένη τεχνολογία συναντάται σε πληθώρα δημοφιλών ενσύρματων ή ασύρματων συσκευών όπως τηλεοράσεις, κινητά τηλέφωνα και εκτυπωτές, δημιουργώντας κατ' επέκταση μεγαλύτερο κίνητρο στην έρευνα συστημάτων που να εκμεταλλεύονται την υπάρχουσα υποδομή.

Το μήκος κύματός τους κυμαίνεται από το 1 χιλιοστό έως τα 700 νάνο-μέτρα, όπου ξεκινά το ορατό φάσμα. [Εικόνα 2-2-1]. Έτσι, αποφεύγονται τα διαφορά προβλήματα που εμφανίζονται στα συστήματα που χρησιμοποιούν ορατά στον άνθρωπο κύματα, τα οποία θα αναλύσουμε παρακάτω. Επιπλέον, τα συστήματα που χρησιμοποιούν υπέρυθρες ακτίνες είναι σε γενικές γραμμές ανθεκτικά στα σφάλματα με μεγάλη ακρίβεια στις προβλέψεις τους.



Εικόνα 2-2-1 Το μήκος κύματος και η συχνότητα εκπομπής των υπέρυθρων ακτινών.

Στον αντίποδα, αν η εφαρμογή μας θέλει ακριβή χωροθέτηση, θα πρέπει να υπάρχει άμεση οπτική επαφή μεταξύ πομπού και δέκτη καθώς οι ακτίνες δε μπορούν να περάσουν τοίχους ή πολλαπλά εμπόδια. Σαν αποτέλεσμα, τα συστήματα αυτά είναι εφαρμόσιμα μόνο σε εφαρμογές που χρειάζονται ανίχνευση μερικών μέτρων καθώς κάθε συσκευή ανίχνευσης έχει εύρος το δωμάτιο στο οποίο έχει τοποθετηθεί. Επίσης, δυνατές πηγές φωτός, όπως οι ηλιακές ακτίνες, μπορούν να εμποδίσουν την εξαγωγή σωστών μετρήσεων. Τέλος, παρά το γεγονός πως η απαιτούμενη υποδομή είναι απλή και εύκολη στη τοποθέτηση και συντήρηση, είναι αρκετά ακριβή.

Τα συστήματα που εκμεταλλεύονται τις υπέρυθρες ακτίνες μπορούν να χωριστούν σε δύο κατηγορίες: τα συστήματα που απαιτούν συσκευές-πομπούς και

αυτά που εκμεταλλεύονται τη φυσική ή τεχνητή εκπομπή υπέρυθρης ακτινοβολίας. Με τη περιγραφή των συστημάτων που ακολουθούν θα προσπαθήσουμε να αποσαφηνίσουμε τις κύριες διαφορές των δύο καπηγοριών.

- ***Active Badge***

Το Active Badge [10] [10] είναι από τα πρώτα συστήματα εσωτερικής χωροθέτησης που εκμεταλλεύονται τις υπέρυθρες ακτίνες και ανήκει στη κατηγορία των συστημάτων που απαιτούν την ύπαρξη συσκευών εκπομπής. Αναπτύχθηκε στις αρχές του 1990 στα AT&T εργαστήρια του Cambridge.

Μια μικρή συσκευή εκπομπής θα πρέπει να είναι στη κατοχή καθενός ατόμου προς ανίχνευση. Αυτή εκπέμπει κάθε δεκαπέντε δευτερόλεπτα ένα μοναδικό για κάθε χρήστη υπέρυθρο σήμα. Σε κάθε δωμάτιο του κτιρίου έχουν τοποθετηθεί δέκτες οι οποίοι ανιχνεύουν αυτές τις εκπομπές και με τη βοήθεια της COO τεχνικής μπορούν να δώσουν μια καλή προσέγγιση για το δωμάτιο στο οποίο βρίσκεται ο πομπός και κατ' επέκταση ο κομιστής του.

Η ακρίβεια του συγκεκριμένου συστήματος είναι παρόμοια με τα όρια εκπομπής υπέρυθρου σήματος, που είναι έξι μέτρα. Οι πομποί είναι μικροί σε μέγεθος, εύκολοι στη χρήση και με μπαταρίες που μπορούν να λειτουργούν μέχρι και ένα χρόνο κάτι πολύ βολικό για τους χρήστες. Επιπλέον, θέματα μυστικότητας δύσκολα μπορούν να υπάρξουν από τη στιγμή που ο χρήστης οικειοθελώς έχει προμηθευτεί τη συσκευή εκπομπής. Ωστόσο, η μικρή ισχύ των υπέρυθρων κυμάτων μπορεί να αναγκάσει στη τοποθέτηση πολλών δεκτών σε μεγάλα δωμάτια, για να μπορεί να υπάρχει κάλυψη σε όλους του χώρους. Η αργή εκπομπή των πομπών (15 δευτερόλεπτα), καθιστά το σύστημα αδύνατο για εφαρμογές που απαιτούν ζωντανή κάλυψη των χώρων και αναφορά σε πραγματικό χρόνο.

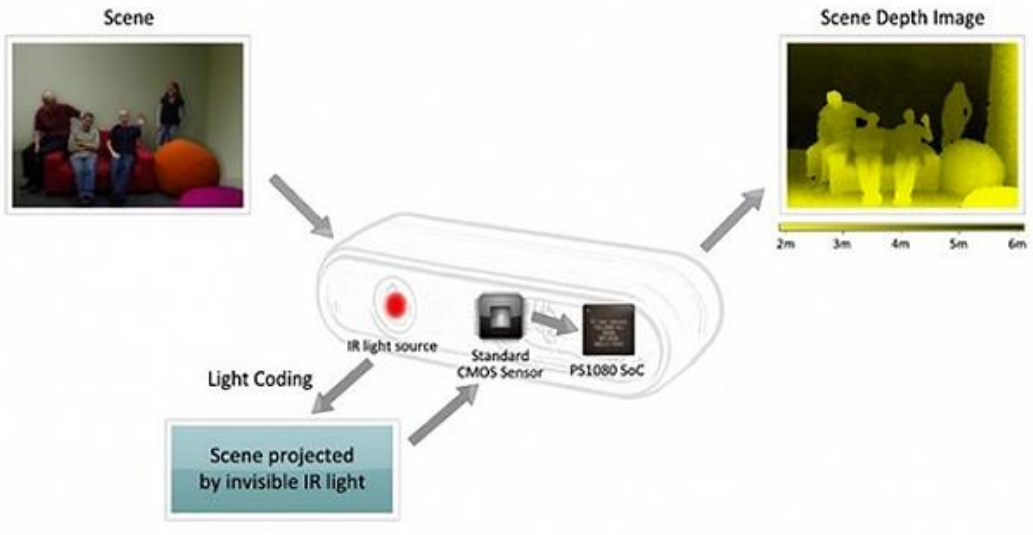
- ***Kinect***

Το Kinect (Project Natal) [11] [12] είναι μια συσκευή ανίχνευσης που χρησιμοποιείται στη κονσόλα παιχνιδιών της Microsoft, XBox. Αναπτύχθηκε με τη βοήθεια της Rare, θυγατρικής εταιρίας της Microsoft, και της PrimeSense και έγινε διαθέσιμο στο κοινό στα τέλη του 2010. Ανήκει στη δεύτερη κατηγορία συστημάτων υπέρυθρων, και πιο συγκεκριμένα χρησιμοποιεί τεχνητή εκπομπή ακτινών.

Με τη εκπομπή της υπέρυθρης ακτίνας και την ταυτόχρονη ανίχνευση της από το CMOS αισθητήρα μπορεί να μετρηθεί ο χρόνος που χρειάστηκε για να φτάσει η ακτίνα σε εμπόδιο (τεχνική χρόνου άφιξης – TOA) κι έτσι να τοποθετηθούν τα προς ανίχνευση άτομα στο χώρο.

Η ακρίβεια του συστήματος έχει μετρηθεί 1cm σε 2 μέτρα απόσταση από τη συσκευή. Τα όρια ανίχνευσης της συσκευής είναι τα 3.5 μέτρα. Παρά την εξαιρετική ακρίβεια στις μετρήσεις το σύστημα δεν μπορεί να αποφύγει τους περιορισμούς της τεχνολογίας, όπως τα προβλήματα σε περιπτώσεις

δυνατών πηγών φωτός στο ίδιο δωμάτιο, παρά τα φίλτρα που χρησιμοποιεί για να τα περιορίσει.

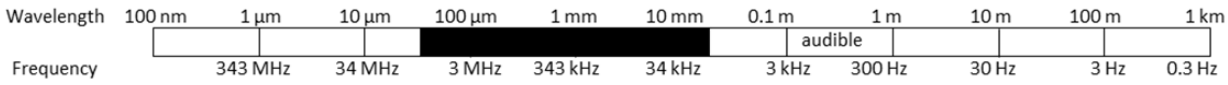


Εικόνα 2-2-2 Τρόπος λειτουργίας του Kinect.

Ultra Sound-Based συστήματα

Μια άλλη τεχνολογία που έχει χρησιμοποιηθεί σε μεγάλο βαθμό σε συστήματα εσωτερικής χωροθέτησης είναι αυτή των υπέρηχων. Οι υπέρηχοι χρησιμοποιούνται από τις νυχτερίδες για να μπορούν να βρίσκουν το δρόμο τους κάτι το οποίο ενέπνευσε πολλούς στο σχεδιασμό συστημάτων για την εξυπηρέτηση των ανθρώπων πλέον.

Πρόκειται για μια σχετικά φθηνή τεχνολογία που συνεπάγεται και σε ανάπτυξη συστημάτων μικρού κόστους. Όμως, υποφέρει από πολλούς περιορισμούς. Η ακρίβεια στη πρόβλεψη της ακριβής θέσης είναι χειρότερη από αυτή των υπέρυθρων ακτινών. Το χειρότερο ίσως μειονέκτημα είναι πως τα περισσότερα συστήματα που χρησιμοποιούν υπέρηχους είναι επιρρεπή στο θόρυβό. Μικρές παρεμβολές μπορούν να επηρεάσουν σε μεγάλο βαθμό τα αποτελέσματα.



Εικόνα 2-2-3 Το μήκος κύματος και η συχνότητα εκπομπής των υπέρηχων.

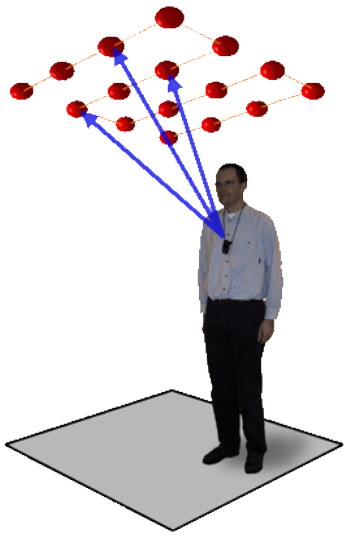
Τα συστήματα υπέρηχων διακρίνονται σε δυο κατηγορίες: τα συστήματα ενεργών συσκευών, και τα συστήματα παθητικών συσκευών.

Τα πρώτα απαιτούν από κάθε προς ανίχνευση συσκευή την εκπομπή υπέρηχων. Το μειονέκτημα αυτής της προσέγγισης είναι πως σε περίπτωση πολλών συσκευών σε ένα δωμάτιο μπορούμε να έχουμε παρεμβολές που μπορούν να μειώσουν δραματικά την ακρίβεια. Ο συγχρονισμός των συσκευών είναι εφικτός, όμως σε μια τέτοια περίπτωση έχουμε μείωση του ρυθμού μετρήσεων και άρα της αποδοτικότητας του συστήματος.

Τα συστήματα παθητικών συσκευών βασίζονται σε πομπούς τοποθετημένους μόνιμα στο χώρο που μεταδίδουν κύματα στους προς ανίχνευση δέκτες. Αυτοί λαμβάνουν αυτά τα σήματα και στη συνέχεια με συγκεκριμένες τεχνικές χωροθέτησης μπορούν να βρουν προσεγγιστικά τη θέση τους. Το πλεονέκτημα αυτής της προσέγγισης είναι η εξασφάλιση της μυστικότητας από πλευράς δέκτη αφού μόνο αυτός έχει τις πληροφορίες που χρειάζονται για τον υπολογισμό της θέσης του. Επιπροσθέτως, η αύξηση των δεκτών σε ένα χώρο δεν επηρεάζει αρνητικά την ακρίβεια του συστήματος εν αντιθέσει με τη προσέγγιση των ενεργών συσκευών.

- ***Bat System***

Το Bat System [13] αναπτύχθηκε από ερευνητές του AT&T στο Cambridge και είναι μια προσπάθεια βελτίωσης του συστήματος Active Badge. Συγκεκριμένα, το τελευταίο δεν ήταν σε θέση να δώσει 3D πληροφορία καθώς και πληροφορία για την γωνία του προς ανίχνευση στοιχείου, με αποτέλεσμα την ανάγκη ανάπτυξης ενός συστήματος για να καλύψει αυτές τις ανάγκες. Ανήκει στη κατηγορία των συστημάτων ενεργών συσκευών.



Εικόνα 2-2-4 Τρόπος λειτουργίας Bat System

Για τον υπολογισμό της ζητούμενης θέσης, το Bat System χρησιμοποιεί τη τεχνική του trilateration. Ένας παλμός εκπέμπεται από το πομπό ο οποίος καταλήγει στους δέκτες οι οποίοι είναι τοποθετημένοι σε γνωστά σημεία στο ταβάνι του δωματίου. Γνωρίζοντας το χρόνο άφιξης του σήματος σε κάθε δέκτη, καθώς και τη ταχύτητα του ήχου στον αέρα μπορούμε εύκολα να υπολογίσουμε την απόσταση μεταξύ πομπού και δέκτη. Θα χρειαστούμε τρεις μετρήσεις πριν μπορέσουμε να χρησιμοποιήσουμε τη τεχνική του trilateration. Κάθε παραπάνω μέτρηση είναι σε θέση να βελτιώσει τη πρόβλεψη χρησιμοποιώντας συγκεκριμένο αλγόριθμο. Συγχρόνως μπορούμε να υπολογίσουμε και μια προσέγγιση για τη γωνία της προς ανίχνευση συσκευής με βάση το RSS στους διάφορους δέκτες.

Σύμφωνα με τα πειράματα του [13]720 δέκτες σε 1000m^2 μπορούν να ανιχνεύσουν μέχρι και 75 συσκευές το δευτερόλεπτο με ακρίβεια περίπου 3cm στις τρεις διαστάσεις. Οι πομποί είναι μικροί και βολικοί για τους χρήστες καθώς δε χρειάζονται συνεχή φόρτιση. Όμως, η ακρίβεια αυτού του συστήματος εξαρτάται από τα εμπόδια που βρίσκονται μεταξύ πομπού και δέκτη, καθώς αλλοιώνει τις μετρήσεις. Επίσης, η τοποθέτηση τόσο μεγάλου αριθμού δεκτών στο ταβάνι μπορεί να είναι χρονοβόρα και πολύπλοκη.

- ***Cricket***

Το Cricket [14][15] είναι ένα σύστημα που χρησιμοποιεί παθητικές συσκευές. Αναπτύχθηκε από ομάδα του MIT και η βασική του ιδέα έχει χρησιμοποιηθεί σε πολλές μεταγενέστερες εφαρμογές.

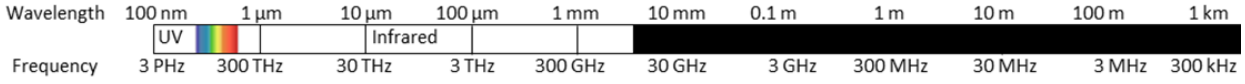
Σε αυτή τη περίπτωση έχουμε πομπούς στο ταβάνι των δωματίων οι οποίοι στέλνουν το μήνυμά τους στο κινητό-δέκτη. Αυτός ακολουθώντας την αντίστροφη διαδικασία από αυτή του Bat System, υπολογίζει τις αποστάσεις από τους πομπούς και στη συνέχεια με τη τεχνική του triangulation προσεγγίζει τη θέση του. Στη περίπτωση όπου δεν ληφθούν αρκετά σήματα υπέρηχων, υπάρχει εναλλακτικός τρόπος εσωτερικής χωροθέτησης στον οποίο χρησιμοποιούνται ραδιοκύματα τα οποία εκπέμπονται από τους ίδιους πομπούς.

Το Cricket μπορεί να προσφέρει προσέγγιση με απόκλιση απόστασης 10cm και γωνίας 3 μοιρών. Δεν απαιτεί μεγάλο αριθμό πομπών αντίθετα με το μεγάλο αριθμό δεκτών που χρειάζεται το Bat System. Έτσι, είναι περισσότερο εφικτή λύση όταν είναι αναγκαία η κάλυψη μεγάλων χώρων. Ωστόσο, οι δέκτες επιβαρύνονται αρκετά καθώς πρέπει να υπολογίζουν θέσεις βάσει και των υπέρηχων και των ραδιοκυμάτων με αποτέλεσμα να έχουν μεγάλες ανάγκες ενέργειας και τη συχνή αλλαγή μπαταριών.

Radio Frequency (RF) συστήματα

Ένα από τα μεγαλύτερα πλεονεκτήματα που παρουσιάζουν στο σύνολο τους οι τεχνολογίες που θα αναφέρουμε παρακάτω και κάνουν χρήση των ραδιοκυμάτων για αποτελεσματική εσωτερική χωροθέτηση είναι η ικανότητα των συγκεκριμένων κυμάτων να διαπερνούν τοίχους και άλλες «δύσκολες» επιφάνειες, ικανότητα που μέχρι τώρα δεν έχουμε ξαναδεί από τα προαναφερθέντα σήματα. Σαν επακόλουθο έχουμε την δυνατότητα ανίχνευσης μεγαλύτερου χώρου με την ίδια υποδομή. Επιπλέον, συστήματα εκμεταλλευόμενα τα ραδιοκύματα μπορούν να κάνουν χρήση της υπάρχουσας υποδομής.

A) Radio Frequency Identification (RFID)



Εικόνα 2-2-5 Το μήκος κύματος και η συχνότητα εκπομπής των RFID.

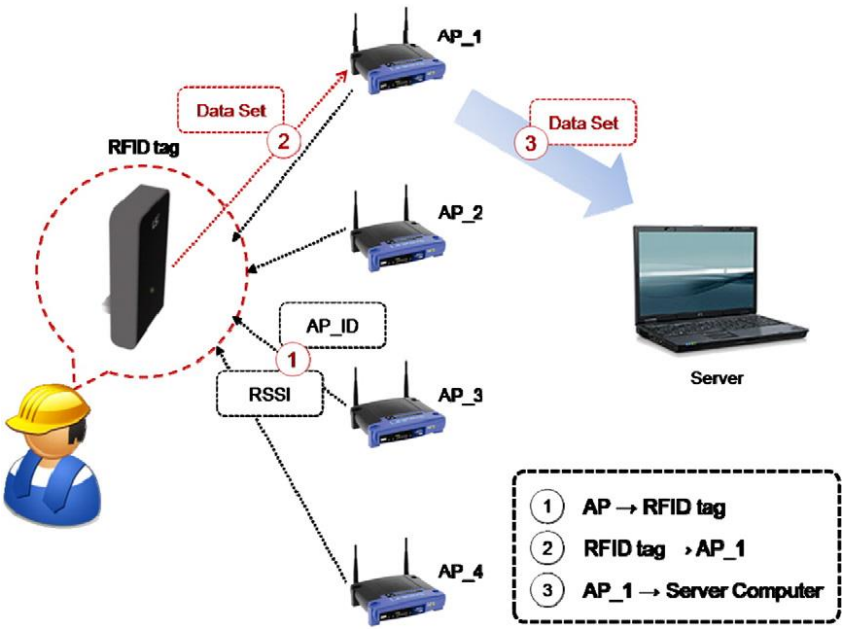
Η τεχνολογία του RFID χρησιμοποιείται σε περίπλοκους εσωτερικούς χώρους όπως νοσοκομεία, γραφεία ή πολυκαταστήματα. Είναι σε θέση να ανιχνεύσει με επιτυχία τη θέση συγκεκριμένου ατόμου ή αντικειμένου με τη βοήθεια συσκευής που κρατάει ο τελευταίος. Αυτή η συσκευή μπορεί είτε να εκπέμπει είτε να δέχεται ραδιοκύματα, χωρίζοντας έτσι και αυτή τη τεχνολογία σε δύο κατηγορίες: τα συστήματα ενεργών και παθητικών RFID. Τα συστήματα ενεργών RFID είναι πιο ακριβά από αυτά των παθητικών καθώς τα τελευταία δεν έχουν ανάγκη μπαταρίας. Η κάλυψη των ενεργών όμως είναι πολύ μεγαλύτερη

- ***Labor tracking System***

Στο [16] [16] μπορούμε να δούμε ένα σύστημα το οποίο αναπτύχθηκε για της ανάγκες εσωτερικής χωροθέτησης σε εργοτάξιο του Guangzhou. Πιο συγκεκριμένα, ήταν αναγκαία η ανίχνευση και η διαρκής αναφορά για τη θέση σημαντικών εργαλείων, μηχανών και οχημάτων. Η χρήση τεχνολογίας ενεργών συσκευών RFID κρίθηκε απαραίτητη.

Χρησιμοποιείται η μέθοδος του fingerprint για τη αποθήκευση των τιμών της δύναμης του σήματος σε διαφορετικά σημεία του χώρου. [Εικόνα 2-2-6] Με τη χρήση συγκεκριμένων αλγόριθμων γίνεται προσπάθεια φιλτραρίσματος των αποθηκευμένων τιμών για το περιορισμό του θορύβου. Αργότερα στη online φάση του fingerprint οι νέες τιμές συλλέγονται και συγκρίνονται με τις αποθηκευμένες για να εξαχθεί η πιο καλή προσέγγιση.

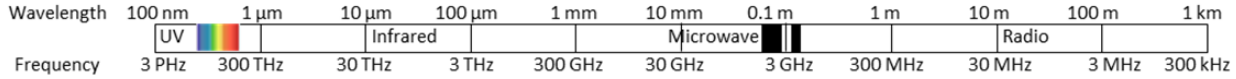
Από την εφαρμογή του συστήματος οι συγγραφείς αναφέρουν πως πέτυχε ακρίβεια 5 μέτρων κάτω από πολύ σκληρές συνθήκες όπως υψηλές θερμοκρασίες και υγρασία, αποδεικνύοντας την αξιοπιστία των συστημάτων εκπομπής ραδιοκυμάτων σε αντίξοες περιπτώσεις.



Εικόνα 2-2-6 Ενεργό RFID για ανίχνευση αντικειμένων [16]

B) WLAN-Based συστήματα

Η τεχνολογία WLAN (IEEE 802.11) είναι από τις πιο διαδεδομένες καθώς συστήματα εσωτερικής χωροθέτησης που την εκμεταλλεύονται έχουν χρησιμοποιηθεί σε πολλούς δημόσιους χώρους. Αυτό οφείλεται κυρίως στο γεγονός πως μπορούν να χρησιμοποιούν την υπάρχουσα υποδομή από τη στιγμή μάλιστα όπου πολυάριθμα ασύρματα σημεία πρόσβασης WLAN κατακλύζουν τις περισσότερες αστικές περιοχές. Η απόσταση των 20 έως 50 μέτρων που είναι το εύρος των περισσότερων WLAN είναι σαφώς μεγαλύτερο από αυτά των Bluetooth ή RFID κάνοντας την ακόμα πιο ελκυστική επιλογή.



Εικόνα 2-2-7 Το μήκος κύματος και η συχνότητα εκπομπής του WLAN.

Οι μεγάλες προκλήσεις που καλούνται να αντιμετωπίσουν οι ερευνητές όσον αφορά τους περιορισμούς αυτής της τεχνολογίας είναι η μείωση της ακρίβειας λόγω διαφορετικών παραγόντων όπως η καθυστέρηση σημάτων λόγω εμποδίων μεταξύ πομπού και δέκτη, η θέση και η κίνηση του ατόμου προς ανίχνευση και η επικάλυψη των ασύρματων σημείων πρόσβασης. Όλοι αυτοί οι παράγοντες, κάνουν σχεδόν αδύνατη τη χρήση τεχνικών όπως του ToA, TDoA και AoA, αφήνοντας μόνο τις τεχνικές του fingerprinting και του RSS ως τις πιο αξιόπιστες. Ακόμα και αυτές, όμως, στη γενική περίπτωση, δε μπορούν να δώσουν μεγαλύτερη ακρίβεια από 3-5 μέτρα [17]

- ***Radar***

Το Radar [18] είναι η πρώτη απόπειρα για σύστημα εσωτερικής χωροθέτησης με βάση τη τεχνολογία WLAN και τις τεχνικές του RSS και του fingerprinting. Αναπτύχθηκε από ερευνητική ομάδα της Microsoft και θεωρείται από τους πρωτοπόρους της ιδέας χρήσης του fingerprinting στην εσωτερική χωροθέτηση. Η βασική διαφορά από τη κλασσική περιγραφή του αλγόριθμου του fingerprinting που έχουμε αναφέρει παραπάνω είναι η χρήση των κ-κοντινότερων γειτόνων στην online φάση.

Στα πειράματα του παρατηρούμε ακρίβεια 5 μέτρων χρησιμοποιώντας 3 access points σε χώρο 1000 m^2 . Κάποια από τα συμπεράσματα στα οποία καταλήγουν οι συγγραφείς είναι πως η απόδοση του συστήματος εξαρτάται από τον αριθμό των στοιχείων που έχουν συλλεχθεί κατά την offline φάση, αλλά και από την κατεύθυνση και ταχύτητα του χρήστη προς ανίχνευση.

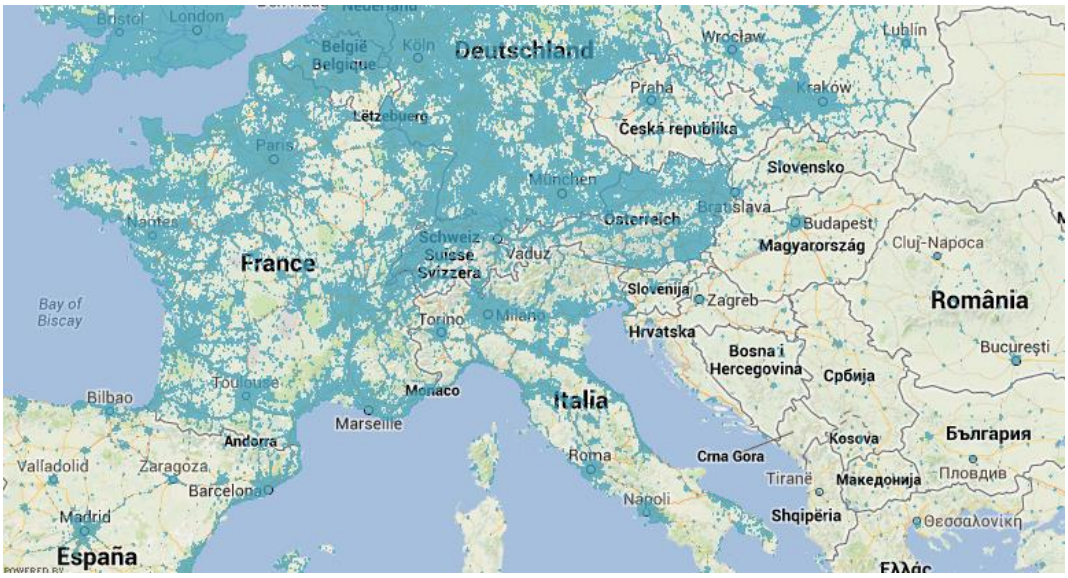
- ***Skyhook***

Το σύστημα Skyhook [19] είναι μια εμπορική λύση στην εσωτερική χωροθέτηση που έχει χρησιμοποιηθεί κατά καιρούς από πολλούς κατασκευαστές σύγχρονων συσκευών όπως η Apple και η Samsung.

Κάνει χρήση της signal fingerprint μεθόδου που έχουμε αναλύσει σε προηγούμενο κεφάλαιο. Το κύριο πρόβλημα είναι αυτό της κάλυψης των χώρων προς ανίχνευση με την συλλογή αρκετών πληροφοριών για την offline φάση του αλγορίθμου. Αυτό επιτυγχάνεται με την διατήρηση και τον εμπλουτισμό μιας παγκόσμιας βάσης δικτύων Wi-Fi. Αυτή η βάση κρατάει αυτή τη στιγμή πληροφορίες από 700 εκατομμύρια δίκτυα Wi-Fi στο 70% των αστικών περιοχών της Βόρειας Αμερικής, Ευρώπης, Αυστραλίας και Ασίας. Στη [Εικόνα 2-2-8] μπορούμε να δούμε ένα κομμάτι αυτού του χάρτη για τη κάλυψη στη κεντρική Ευρώπη.

Παρατηρούμε πως από τι στιγμή όπου η offline φάση γίνεται ανεξάρτητα από το χρόνο που αυτή θα ζητηθεί, ο χρήστης δεν επιβαρύνεται με το χρόνο που αυτή απαιτεί. Σύμφωνα με τις δοκιμές απόδοσης του συστήματος παρατηρείται ακρίβεια 10 μέτρων χρησιμοποιώντας μια υβριδική εκδοχή του συστήματος με χρήση Wi-Fi positioning σε εσωτερικούς χώρους και GPS σε εξωτερικούς.

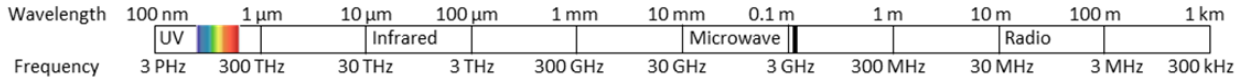
Στον αντίποδα, δεν έχει δοθεί η απαραίτητη προσοχή σε θέματα ασφάλειας. Εικονικά δίκτυα Wi-Fi με Mac διευθύνσεις που έχουν αποθηκευτεί στο σύστημα μπορούν να παραμορφώσουν τα αποτελέσματα υπολογίζοντας θέση απόκλισης χιλιάδων χιλιομέτρων. Αυτό μπορεί να είναι ανώδυνο σε κάποιες περιπτώσεις ωστόσο όταν πρόκειται για συστήματα ασφάλειας μπορεί να είναι καταστροφικό.



Εικόνα 2-2-8 Γεωγραφική κάλυψη των Wi-Fi δικτύων στη κεντρική Ευρώπη.

C) Bluetooth-Based συστήματα

Η τεχνολογία του Bluetooth (IEEE 802.15.1) έκανε την εμφάνιση της το 1994 και πλέον διευθύνεται από τη Bluetooth Special Interest Group (SIG). Σχεδιάστηκε με σκοπό την ελαχιστοποίηση της καταναλισκόμενης ενέργειας και του κόστους των δεκτών. Η περιοχή κάλυψης των Bluetooth συσκευών είναι κοντά στα 10 μέτρα, σαφώς μικρότερη από αυτή των WLAN, και με τη σειρά της επηρεάζεται σημαντικά από το περιβάλλων χώρο.



Εικόνα 2-2-9 Το μήκος κύματος και η συχνότητα εκπομπής του Bluetooth.

Μια συσκευή μπορεί να επικοινωνήσει ταυτόχρονα με άλλες 7 με την τεχνολογία αυτή. Η παρέμβαση του χρήστη δεν είναι αναγκαία για την εγκαθίδρυση μιας τέτοιας επικοινωνίας. Αξίζει να σημειωθεί, επίσης, πως έχει προβλεφθεί η περίπτωση παρεμβολής μεταξύ των συσκευών. Για την αποτροπή αυτής της πιθανότητας γίνεται χρήση μιας τεχνικής στην οποία κάθε συσκευή εναλλάσσει την εκπομπή της ανάμεσα σε 79 πιθανές συχνότητες, 1600 φορές το δευτερόλεπτο.

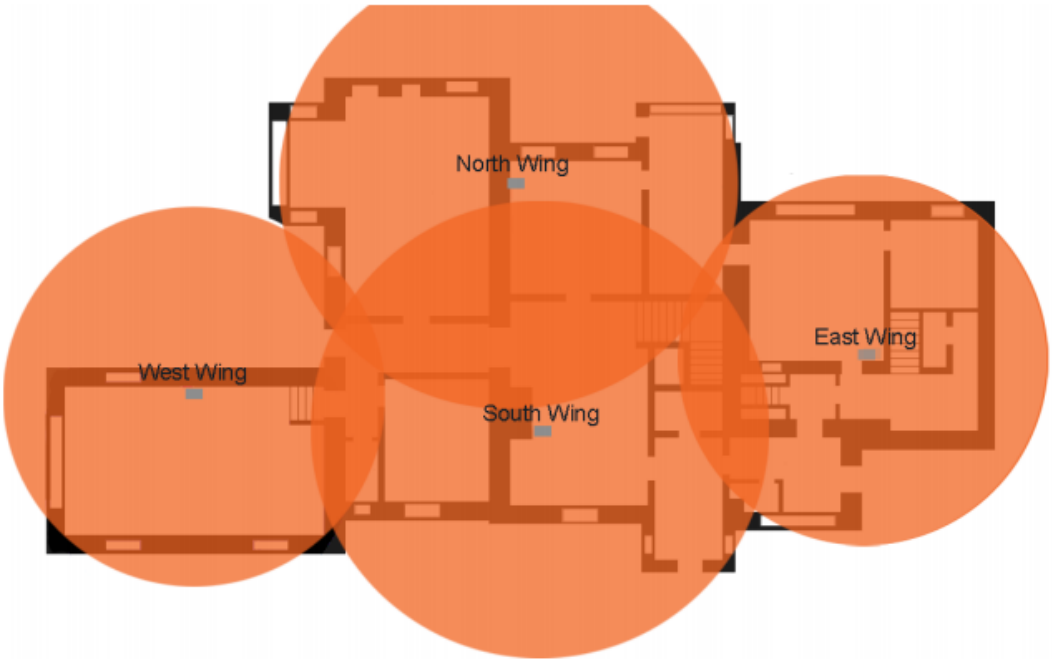
Ο κύριος περιορισμός για τα συστήματα που εκμεταλλεύονται την τεχνολογία του Bluetooth είναι πως μπορούν να ανιχνεύουν με καθυστέρηση τουλάχιστον 10 δευτερολέπτων. Αυτός είναι ο χρόνος που απαιτεί το Bluetooth για να ανανεώσει τη λίστα των διαθέσιμών συσκευών για επικοινωνία. Αυτό, παρόλο που ανήκει στο πρωτόκολλο του Bluetooth που προσφέρει την ασφάλεια στις επικοινωνίες, καθιστά τη συγκεκριμένη τεχνολογία μη αρεστή σε περιπτώσεις όπου χρειάζεται ανίχνευση σε πραγματικό χρόνο.

- ***ZONITH***

Το σύστημα εσωτερικής χωροθέτησης Zonith της Teldio [20] [21] είναι ένα σύστημα που βασίζεται στο Bluetooth για την εξασφάλιση ασφάλειας σε επικίνδυνους εργασιακούς χώρους. Πιο συγκεκριμένα, είναι σε θέση να στείλει προειδοποιητικό σήμα με τη θέση κάποιου εργάτη ο οποίος μένει ακίνητος για μεγάλο χρονικό διάστημα ή να κάνει γνωστή τη θέση προσωπικού ασφαλείας ο οποίος βρίσκεται σε κίνδυνο, χωρίς ο τελευταίος να πρέπει να αναφέρει τη τοποθεσία του, αλλά απλώς πατώντας ειδικό κουμπί στον ασύρματο.

Το Zonith χρησιμοποιεί την μέθοδο του Cell of Origin (COO). Πομποί τοποθετούνται στα κέντρα των δωματίων του κτιρίου και μπορούν να επικοινωνούν με τις συσκευές Bluetooth που βρίσκονται σε αυτά. Οι πομποί θα πρέπει να τοποθετηθούν στη σωστή θέση και η δύναμη του σήματος τους να ρυθμιστεί με τέτοιο τρόπο ώστε να μην υπάρχουν σημεία χωρίς κάλυψη. Ένας ενδιαφέρον μηχανισμός, που μπορεί να ενεργοποιηθεί κατά βούληση, είναι η ανίχνευση και η ικανότητα του συστήματος να τοποθετήσει το χρήστη στα σύνορα μεταξύ δύο ή παραπάνω πομπών. Όπως βλέπουμε στην εικόνα

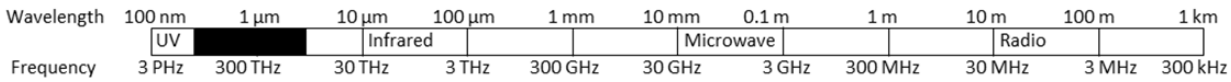
2-2-10 οι 4 πομποί μπορούν να χωρίσουν το κτίριο σε 11 πεδία. Έτσι όταν μια συσκευή Bluetooth «βλέπει» και τη συσκευή του *North Wing*, και του *West Wing*, μπορεί να τοποθετηθεί στα κοινά σημεία τους.



Εικόνα 2-2-10 Παράδειγμα τοποθέτησης πομπών Bluetooth που χωρίζουν το κτίριο σε 4 ή 11 πεδία. [20]

Δεν είναι αναγκαία η εγκατάσταση κάποιου λογισμικού από μεριάς των ανιχνεύσιμων συσκευών, παρά μόνο η εγγραφή τους στο σύστημα παρακολούθησης. Η ακρίβεια του συστήματος εξαρτάται άμεσα από το εύρος κάθε τοποθετημένου πομπού και κατ' επέκταση είναι ανάλογη του αριθμού των πομπών που χρησιμοποιούνται. Η ύπαρξη πολλών συσκευών Bluetooth σε ένα δωμάτιο μπορεί να επιβαρύνει την συχνότητα ανανέωσης των θέσεων λόγω της καθυστέρησης ανίχνευσης διαθέσιμων συσκευών.

Vision-Based συστήματα



Εικόνα 2-2-11 Το μήκος κύματος και η συχνότητα εκπομπής της κάμερας.

Τα συστήματα αυτά βασίζονται σε μια ή περισσότερες κάμερες για τον εντοπισμό και τον υπολογισμό της θέσης του προς ανίχνευση αντικειμένου. Αποκτούν μεγάλη δημοτικότητα καθώς μπορούν να εξυπηρετήσουν μεγάλο εύρος εφαρμογών με

εξαιρετική ακρίβεια. Τα συγκεκριμένα συστήματα έχουν επίσης το πλεονέκτημα της εξαγωγής επιπλέον πληροφοριών από τις εικόνες εκτός της θέσης που μπορούν να φανούν πολύτιμες σε συγκεκριμένες εφαρμογές.

Τα πιο πολλά vision-based συστήματα βασίζονται στη σύγκριση εικόνων που έχουν τραβηχτεί σε ζωντανό χρόνο με μια σειρά εικόνων που έχουν τραβηχτεί σε περασμένο χρόνο. Με τη σύγκριση αυτών μπορεί να ανιχνευθεί η κίνηση και να υπολογισθεί η θέση των κινητών. Άλλες τεχνικές είναι η σύγκριση με τρισδιάστατο μοντέλο του χώρου, η χρήση συγκεκριμένων σημείων αναφοράς όπως γραμμωτών κωδικών ή ομόκεντρων δαχτυλιδιών, για το περιορισμό εξωτερικών παραγόντων που επηρεάζουν τις κάμερες, όπως ηλιοφάνεια, αλλά και η χρήση αναφοράς από αντικείμενα που φωτίζονται τεχνητά. Επίσης, έχουν κατά καιρούς προταθεί συστήματα τα οποία δε χρησιμοποιούν κάποια αναφορά αλλά προσπαθούν σε πραγματικό χρόνο να υπολογίσουν την απόσταση. Αυτό απαιτεί στη γενική περίπτωση μεγάλη ακρίβεια και κατ ’επέκταση οι κάμερες που είναι υψηλού κόστους.

Ο μεγαλύτερος περιορισμός αυτής της τεχνολογίας είναι η αδυναμία σε θέματα ιδιωτικότητας. Κανένας δε μπορεί να εγγυηθεί πως το άτομο που παρακολουθείται το επιθυμεί. Επίσης, τα συστήματα επηρεάζονται πολύ από εξωτερικού παράγοντες, όπως πηγές φωτός ή από τις καιρικές συνθήκες. Τέλος, η ανίχνευση πολλών ατόμων ταυτόχρονα στον ίδιο χώρο, συνεχίζει να είναι δύσκολη, απαιτώντας μεγάλη υπολογιστική ισχύ.

- ***Kim & Jun System***

  Το σύστημα που προτείνουν οι Kim και Jun (2008) [22]**Error! Reference source not found.**, χρησιμοποίει την σύγκριση με παλαιότερες εικόνες για την χωροθέτηση. Οι κάμερες που απαιτεί είναι χαμηλού κόστους και μπορούν να εξυπηρετήσουν μεγάλο χώρο αν τοποθετηθούν σε σωστό σημείο

  Κατασκευάζεται μια βάση δεδομένων με εικόνες από τη τοποθεσία προς ανίχνευση, από την οποία μπορούν να εξαχθούν πληροφορίες σχετικά με τη θέση των σταθερών αντικειμένων και τις αποστάσεις μεταξύ αυτών. Η ακρίβεια και η αποδοτικότητα επιτυγχάνεται με τη χρήση μιας σειράς από τεχνικές της τεχνολογίας, όπως την εκμετάλλευση τοπογραφικών πληροφοριών.

  Με την πειραματική διαδικασία του [22], βρίσκεται πως το σύστημα πετυχαίνει την χωροθέτηση σε 89% των περιπτώσεων, ποσοστό ενθαρρυντικό αν αναλογιστούμε και το χαμηλό κόστος για τις κάμερες.

Άλλα συστήματα

Παραπάνω προσπαθήσαμε να περιγράψουμε τις πιο δημοφιλείς προσεγγίσεις για την ανάπτυξη συστημάτων εσωτερικής χωροθέτησης. Ωστόσο, κατά καιρούς έχουν προταθεί και συστήματα που εκμεταλλεύονται εναλλακτικές τεχνολογίες. Οι περιορισμοί αυτών των τεχνολογιών όμως τα καθιστούν τις περισσότερες φορές απρόσιτα για γενική χρήση.

Η τεχνολογία υπερ-ευρείας ζώνης (**Ultra Wideband** – UWB) ήρθε στο προσκήνιο στις αρχές τις προηγούμενης δεκαετίας ως μια πολλά υποσχόμενη λύση για

ασύρματα συστήματα με υψηλούς ρυθμούς δεδομένων, ασφαλούς επικοινωνίας, χαμηλή κατανάλωση, χαμηλή πολυπλοκότητα και με μικρό κόστος υλοποίησης, με ικανότητες για εντοπισμό θέσεως και εξαιρετικά χαμηλές παρεμβολές σε άλλα ασύρματα συστήματα. Η συνήθης ακρίβεια των συστημάτων που κάνουν χρήση αυτής της τεχνολογίας είναι μεταξύ εκατοστών και μέτρου, ενώ το εύρος των πομπών είναι μέχρι 50 μέτρα. Παρά την εξαιρετική ακρίβεια των συστημάτων αυτών, κανένα δεν έχει διατεθεί στο ευρύ κοινό παρά μόνο περιστασιακά σε βιομηχανικές εφαρμογές. Ο κύριος λόγος για αυτό είναι η απουσία υπάρχουσας υποδομής.

Τα συστήματα βασιζόμενα στον ήχο που αντιλαμβάνεται ο άνθρωπος (**Audible Sound Based Systems**) χρησιμοποιούν μεθόδους όπως το TOA και το Trilateration για την επιτυχή εσωτερική χωροθέτηση. Οι βασικοί περιορισμοί αυτών των συστημάτων που τα περιορίζουν σε ειδικές εφαρμογές είναι η αναπόφευκτη παρεμβολή από ήχους του περιβάλλοντα χώρου, η περιορισμένη ικανότητα του ήχου να διαπερνά εμπόδια καθώς και η πιθανή ενόχληση των ανθρώπων που βρίσκονται στο κτίριο από τον ήχο των πομπών.

Η τεχνολογία του **GNSS** (τεχνολογία που χρησιμοποιεί το γνωστό Global Positioning System) είναι η μόνη η οποία μπορεί να έχει παγκόσμια κάλυψη και να μην εξαρτάται από την τοπική υποδομή. Αυτοί οι λόγοι καθιστούν σαφές το λόγο που ένα επιτυχημένο σύστημα βασισμένο στη τεχνολογία αυτή θα ήταν πραγματικό άλμα στην έρευνα της εσωτερικής χωροθέτησης. Όμως, οι εσωτερικοί χώροι δεν μπορούν να καλυφθούν εύκολα από αυτή τη τεχνολογία καθώς απαιτείται οπτική επαφή μεταξύ των δορυφόρων και του δέκτη. Πολλές ερευνητικές προσπάθειες έχουν γίνει με συστήματα που χρησιμοποιούν υπερ-ευαίσθητους δέκτες GNSS ή συστήματα που εκμεταλλεύονται προηγούμενες μετρήσεις για να προβλέψουν επόμενες όταν δεν υπάρχει επικοινωνία με τους δορυφόρους [23] [24]. Τα αποτελέσματα όμως δεν είναι ενθαρρυντικά μέχρι στιγμής.

2.3 Πίνακας Τεχνολογιών

Ακολουθεί πίνακας με τις τεχνολογίες που έχουν χρησιμοποιηθεί για ανάπτυξη συστημάτων εσωτερικής χωροθέτησης, τα τυπικά εύρη ακρίβειας και κάλυψης που συναντάμε καθώς και τους βασικούς περιορισμούς αλλά και πλεονεκτήματα της καθεμίας.

Table 2-1 Τεχνολογίες για εσωτερική χωροθέτηση

Τεχνολογία	Τυπική Ακρίβεια	Τυπικό Εύρος	Πλεονεκτήματα	Περιορισμοί
Infrared	cm-m	1-5(m)	χρήση υπάρχουσας υποδομής – εξαιρετική ακρίβεια	ανάγκη οπτικής επαφής – μικρό εύρος
Ultrasound	cm	2-10(m)	εξαιρετική ακρίβεια – φθηνά συστήματα	επηρεάζονται από θόρυβο/θερμοκρασία – γρήγορη εξασθένηση (μικρό εύρος)
RFID (Radio Frequency)	dm-m	2-50(m)	ικανοποιητικό εύρος (τα RF διαπερνούν επιφάνειες)	εξαρτάται από τη πυκνότητα των RFID tag
WLAN (Radio Frequency)	m	20-50(m)	χρήση υπάρχουσας υποδομής – πολύ φθηνά συστήματα - ικανοποιητικό εύρος	επηρεάζονται πολύ από εμπόδια μεταξύ πομπού/δέκτη και από την επικάλυψη των ασύρματων σημείων πρόσβασης.
Bluetooth (Radio Frequency)	m	5-10(m)	χρήση υπάρχουσας υποδομής – υψηλή ασφάλεια, χαμηλό κόστος και ενέργεια	καθυστέρηση ανανέωσης θέσης (>10 δευτερόλεπτα)
Vision	mm-dm	5-10(m)	εξαιρετική ακρίβεια – εξαγωγή επιπλέον πληροφορίας από θέση	αδυναμία σε θέματα ιδιωτικότητας – αδυναμία στην ανίχνευση πολλών ατόμων ταυτόχρονα
Ultra Wideband	cm-dm	1-50(m)	εξαιρετική ακρίβεια, χαμηλή κατανάλωση, χαμηλή πολυπλοκότητα	δεν υπάρχει έτοιμη υποδομή
Audible Sound	cm-m	5-10(m)	χρήση υπάρχουσας υποδομής - φθηνά συστήματα	επηρεάζεται από ήχους του περιβάλλων χώρου – ακουόμενος ήχος: «ενοχλητικά» συστήματα
GNSS	>10m	θεωρητικά παγκόσμια κάλυψη	η τεχνολογία είναι παγκόσμιας κάλυψης και ανεξαρτήτου υποδομών	αδυναμία αντιμετώπισης προβλημάτων από την έλλειψη οπτικής επαφής με τους δορυφόρους

3. ΠΕΡΙΓΡΑΦΗ ΑΛΓΟΡΙΘΜΟΥ

Σε αυτό το κεφάλαιο θα παρουσιάσουμε τους δύο αλγόριθμους που υλοποιήσαμε στην παρούσα πτυχιακή. Πιο συγκεκριμένα στο κεφάλαιο 3.1 περιγράφεται ο αλγόριθμος που χρησιμοποιήσαμε για να επιτύχουμε τη χωροθέτηση μέσα στο εμπορικό κέντρο (Positioning Method). Στο κεφάλαιο 3.2 γίνεται αναφορά στον αλγόριθμο που εξετάζουμε το ανά πόσο χρονικό διάστημα θα σκανάρει το κινητό για τα διαθέσιμα APs, καθώς και κάθε πότε θα υπολογίζει την θέση-κατάστημα που βρισκόμαστε.

3.1 Positioning Method

Ο αλγόριθμος που χρησιμοποιήσαμε για να λύσουμε το πρόβλημα της εσωτερικής χωροθέτησης (Indoor Positioning) χρησιμοποιεί την τεχνολογία Wi-fi, εκτελώντας παράλληλα μια παραλλαγή της μεθόδου του trilateration. Το ζητούμενο που θέλουμε από τον συγκεκριμένο αλγόριθμο είναι να εντοπίσει αν ο χρήστης βρίσκεται μπροστά από τη βιτρίνα κάποιου καταστήματος μέσα σε ένα εμπορικό κέντρο (mall) καθώς και ποιό κατάστημα είναι αυτό.

Καταρχάς για τη σωστή λειτουργία του αλγορίθμου μας χρειάζεται μία όσο το δυνατόν πιο ακριβής χαρτογράφηση (mapping) του χώρου, δηλαδή του εμπορικού κέντρου στον οποίο θα γίνει η εκτέλεση της εφαρμογής μας. Στα πλαίσια της παρούσας πτυχιακής έχουμε εργαστεί στο καρτεσιανό επίπεδο, δηλαδή χρησιμοποιώντας μόνο το ζεύγος συντεταγμένων (x,y), με την υπόθεση δηλαδή ότι είμαστε στο ίδιο επίπεδο z. Αυτό που χρειάζεται η εφαρμογή είναι να έχουμε χαρτογραφήσει εκ των προτέρων τις θέσεις των APs μέσα στα διάφορα καταστήματα, καθώς και τα όρια της βιτρίνας του κάθε καταστήματος. Ορίζουμε λοιπόν ως προϋπόθεση το πρώτο AP να είναι στο σημείο (0,0) , και από κει και πέρα παίρνουμε μετρήσεις για να ορίσουμε κατάλληλα τις θέσεις των υπόλοιπων APs, καθώς και των ορίων των καταστημάτων. Επίσης για κάθε AP παίρνουμε την Mac Address του καθώς και την ισχύ εκπομπής (Tx-Power).

Έτσι τελικά φτιάχνουμε μια λίστα (λίστα Α) που περιλαμβάνει για κάθε ένα AP τα εξής στοιχεία:

1. Mac Address
2. Tx-Power (σε dBm)
3. Συντεταγμένες στο δισδιάστατο χώρο όπου βρίσκεται το AP.
4. Συντεταγμένες στο δισδιάστατο χώρο του αριστερού ορίου της βιτρίνας του καταστήματος στο οποίο βρίσκεται το συγκεκριμένο AP.
5. Συντεταγμένες στο δισδιάστατο χώρο του δεξιού ορίου της βιτρίνας του καταστήματος στο οποίο βρίσκεται το συγκεκριμένο AP.

Στη συνέχεια έχοντας τα στοιχεία αυτά μπορούμε να βρούμε μια αρκετά καλή προσέγγιση της θέσης που βρίσκεται το κινητό του χρήστη ως εξής:

Το κινητο «σκανάρει» τα διαθέσιμα Access Points και δημιουργείται μια λίστα (λίστα Β) από αυτά, η οποία συγκρίνεται με τη λίστα Α που φτιάξαμε πριν κατά τη διάρκεια της χαρτογράφησης. Στην λίστα Β καταχωρούμε, εκτός από τα ονόματα και τις Mac Addresses των APs, και τα επίπεδα των σημάτων που λαμβάνει από κάθε AP η συσκευή (σε dBm).

Χρησιμοποιώντας τον τύπο [29]:

Ploss= 32.4 + 20log10d + 20log10f

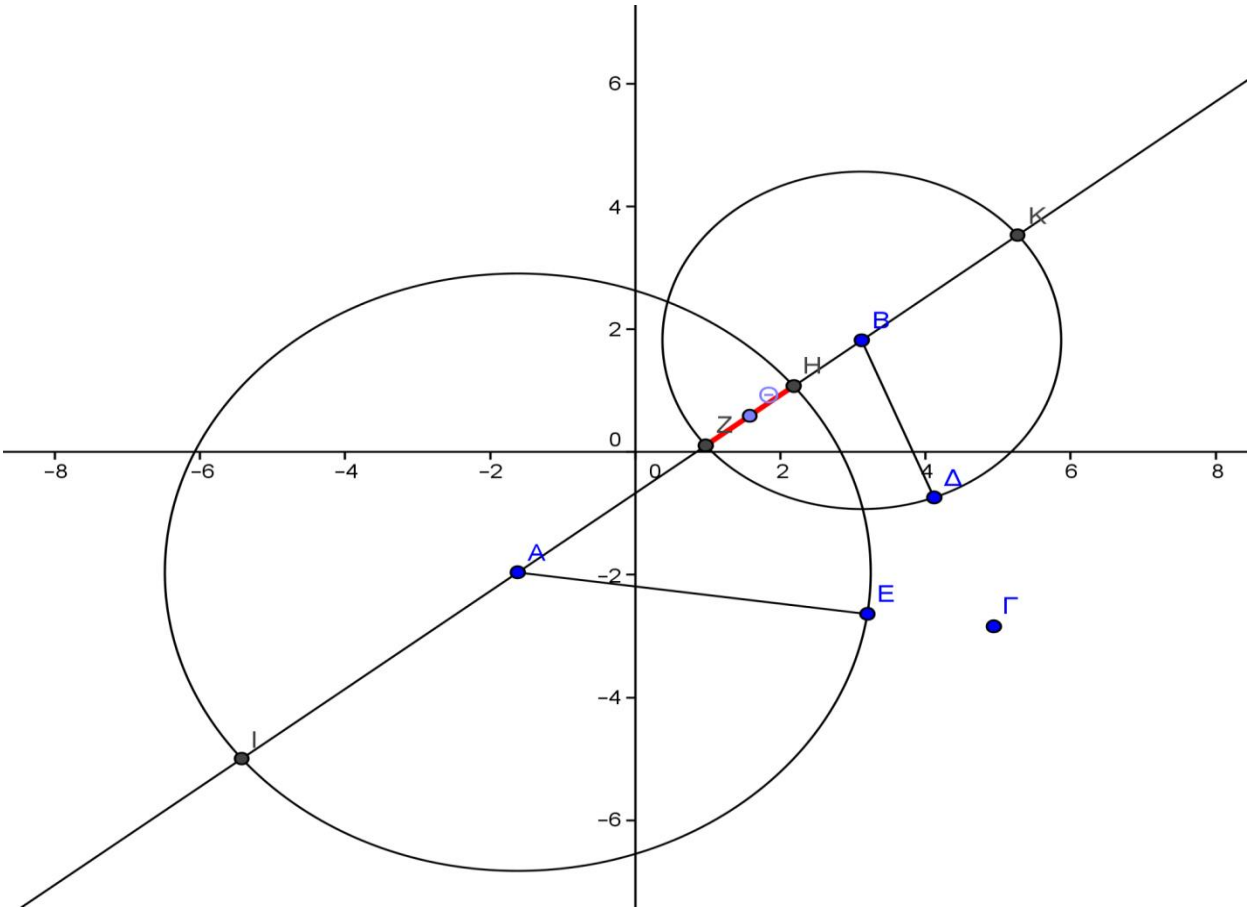
όπου Ploss είναι η απώλεια μετάδοσης (ισχύς εκπομπής μείον την ισχύ με την οποία έφτασε το σήμα στη συσκευή του χρήστη), και f η συχνότητα εκπομπής του δρομολογητή, βρίσκουμε το d, την απόσταση δηλαδή σε χιλιόμετρα του σημείου που βρισκόμαστε από το κάθε AP.

Έτσι φτιάχνουμε μια άλλη λίστα (λίστα Γ) που περιλαμβάνει τα APs που βρήκαμε ότι ανήκουν και στη λίστα Α και στη λίστα Β και για το κάθε στοιχείο-AP της λίστας έχουμε πλέον και την –κατα προσέγγιση- απόσταση μας από αυτό.

Σε αυτό το σημείο αρχίζει το κύριο κομμάτι του αλγορίθμου μας [Εικόνα 3.1.1]: Αρχικά παίρνουμε τα δύο πρώτα στοιχεία της λίστας Γ, τα οποία είναι τα 2 APs στα οποία είμαστε θεωρητικά πιο κοντά με βάση τα επίπεδα των σημάτων που έλαβε το κινητό από αυτά. Χρησιμοποιώντας τις συντεταγμένες των APs, που μας είναι γνωστές, ως κέντρα(σημεία Α και Β) και τις αποστάσεις που υπολογίσαμε(ΑΕ και ΒΔ) από το σημείο που βρισκόμαστε(σημείο Γ), ως ακτίνες αντίστοιχα, σχηματίζουμε δύο κύκλους.

Μέχρι αυτό το σημείο η μέθοδος θυμίζει τη μέθοδο του trilateration, αφού, σύμφωνα με αυτήν, αν παίρναμε και ένα τρίτο AP και φέρναμε τον αντίστοιχο κύκλο, οι τρεις κύκλοι θα τέμνονταν σε ένα κοινό και για τους τρεις σημείο, το οποίο θα ήταν ακριβώς το σημείο που βρισκόμαστε. Όμως αυτό θα συνέβαινε αν οι αποστάσεις ήταν απόλυτα ακριβείς. Στην περίπτωση μας αυτό δε μπορεί να συμβεί, λόγω του ότι τα signal levels που αντιλαμβάνεται το κινητό δεν έχουν υψηλή αξιοπιστία και μπορεί να μεταβάλλονται ακόμα και αν βρισκόμαστε στο ίδιο σημείο, οπότε η απόσταση που υπολογίζουμε από τον τύπο είναι κατά προσέγγιση. Έτσι είναι πολύ πιθανό να μην υπάρχει καν σημείο όπου να τέμνονται και οι τρεις κύκλοι. Επομένως η παραπάνω μέθοδος πρέπει να προσαρμοστεί κατάλληλα.

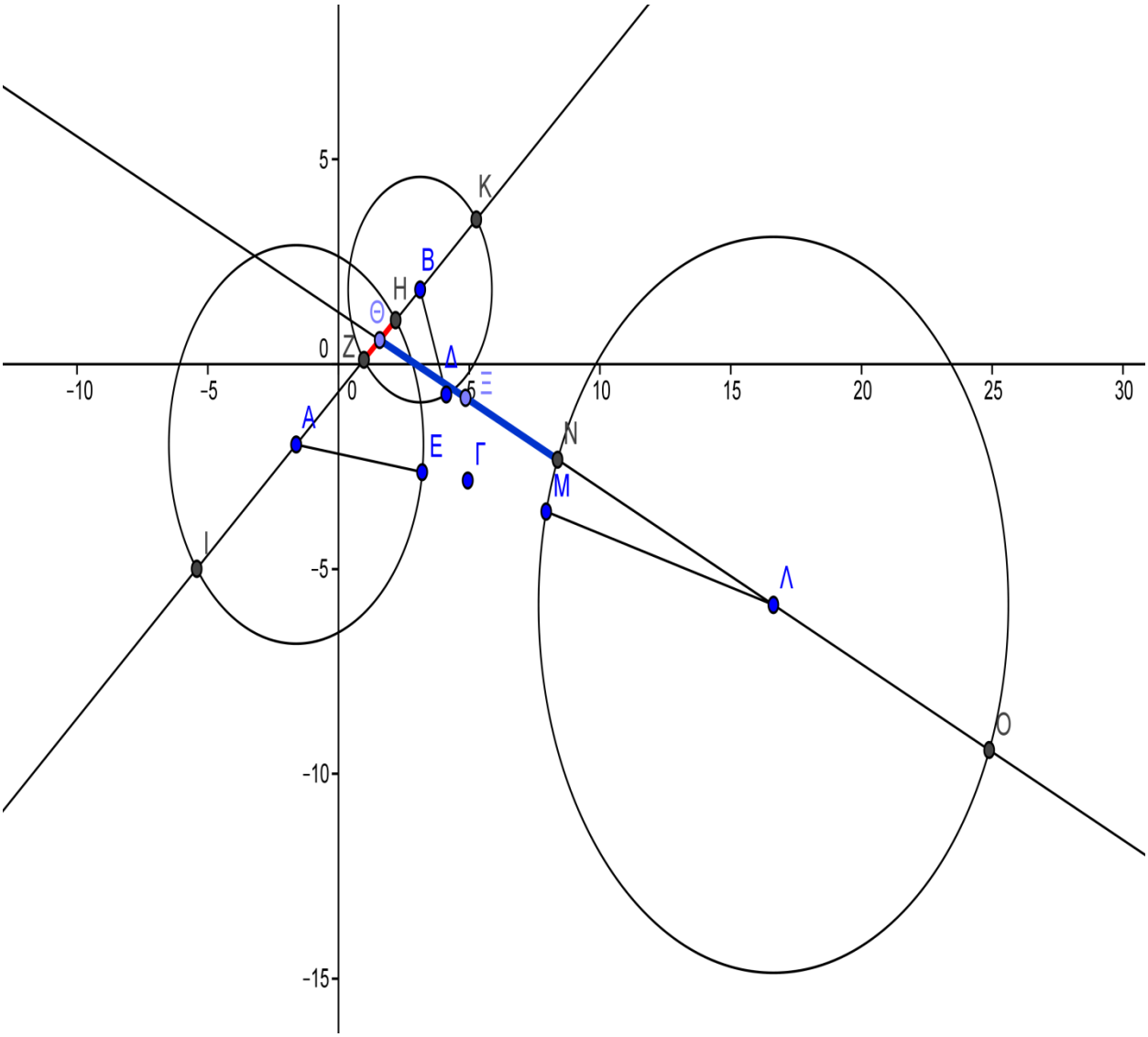
Η στρατηγική λοιπόν που ακολουθούμε είναι η εξής: Καταρχήν φέρνουμε την ευθεία που περνάει από τα κέντρα των δύο κύκλων και τέμνει τον κάθε κύκλο σε δύο σημεία(Ι,Η και Ζ,Κ αντίστοιχα). Κατόπιν ψάχνουμε να βρούμε τα δύο κοντινότερα σημεία των κύκλων που ανήκουν στην ευθεία(Ζ και Η). Αφού τα βρούμε, παίρνουμε το ευθύγραμμο τμήμα που ενώνει τα δύο αυτά σημεία (συμβολίζεται με κόκκινο στο σχήμα μας) και βρίσκουμε το μέσο του (σημείο Θ). Το σημείο αυτό είναι η αρχική μας προσέγγιση ως προς το σημείο που βρισκόμαστε.



Εικόνα 3.1.1: Εύρεση πρώτης προσέγγισης (σημείο Θ)

Στη συνέχεια [Εικόνα 3.1.2] παίρνουμε το επόμενο στοιχείο από τη λίστα Γ (αν υπάρχει) προκειμένου να βρούμε μια καλύτερη προσέγγιση ως εξής: Παίρνουμε κατά όμοιο τρόπο τον κύκλο με κέντρο τις συντεταγμένες του AP (σημείο Λ) και ακτίνα την απόσταση μας από το AP αυτό(ΛΜ) και αυτή τη φορά φέρνουμε την ευθεία που περνάει από το κέντρο του κύκλου και από το σημείο της αρχικής μας προσέγγισης. Βρίσκουμε το σημείο που η ευθεία τέμνει τον κύκλο και είναι το κοντινότερο στο σημείο της αρχικής προσέγγισης (σημείο Ν) και παίρνουμε ως δεύτερη προσέγγιση το μέσο (Ξ) του ευθύγραμμου τμήματος (ΝΘ), το οποίο ενώνει αυτό το σημείο με το σημείο αρχικής προσέγγισης (με μπλε γραμμή στο σχήμα).

Ο αλγόριθμος συνεχίζει για όσα ακόμα στοιχεία-APs έχουμε στη λίστα Γ με όμοιο τρόπο, χρησιμοποιώντας δηλαδή κάθε φορά το τρέχων σημείο προσέγγισης και τον κύκλο που αντιστοιχεί στο επόμενο AP της λίστας. Αφού τα στοιχεία της λίστας Γ τελειώσουν έχουμε πλέον ένα τελικό σημείο το οποίο είναι η καλύτερή μας προσέγγιση. Αυτό που χρειάζεται δηλαδή για να έχουμε κάποια προσέγγιση είναι τουλάχιστον δύο APs στη λίστα Γ.

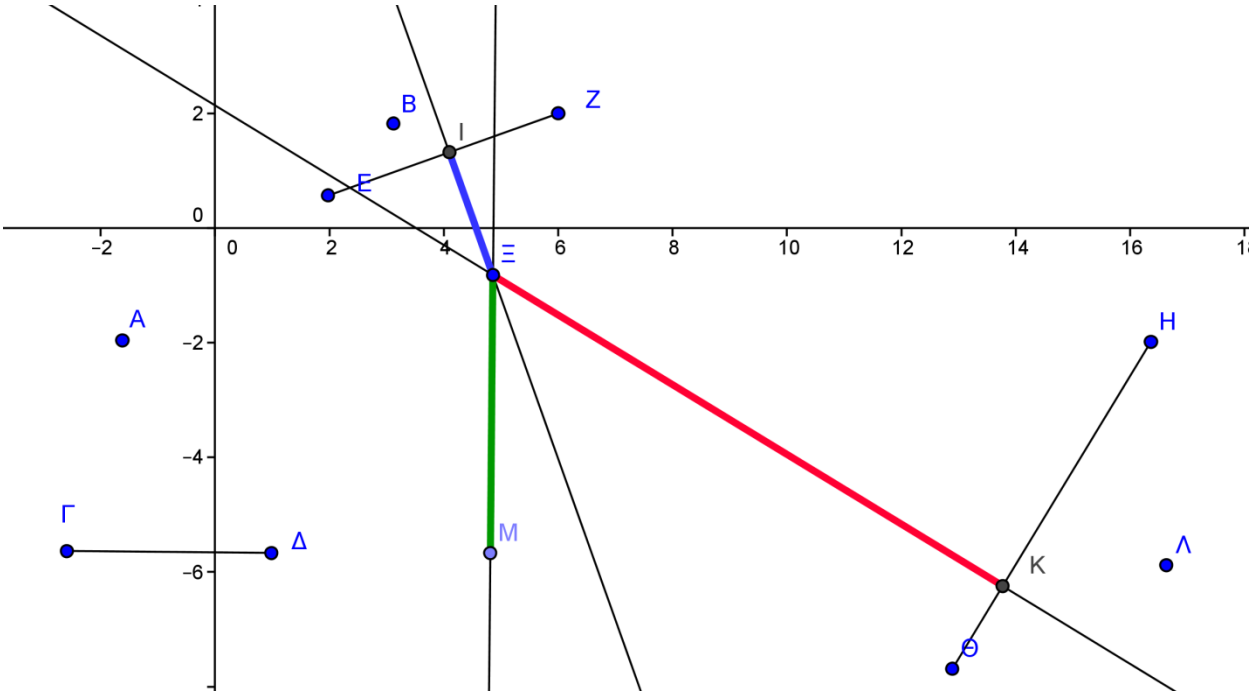


Εικόνα 3.1.2: Εύρεση δεύτερης προσέγγισης (σημείο Ξ)

Με βάση το τελικό αυτό σημείο βρίσκουμε το κατάστημα το οποίο είναι το πλησιέστερο ως εξής: Για κάθε ΑΡ από τη λίστα Γ παίρνουμε τα σημεία που είναι τα όρια του καταστήματος που βρίσκεται το συγκεκριμένο ΑΡ και φέρνουμε την ευθεία που συνδέει αυτά τα όρια. Κατόπιν από το τελικό μας σημείο φέρνουμε κάθετη στην ευθεία αυτή και βλέπουμε αν το σημείο που τέμνει η κάθετη την ευθεία ανήκει στο ευθύγραμμο τμήμα που βρίσκεται μεταξύ των ορίων του καταστήματος. Αν δεν ανήκει αγνοούμε αυτό το κατάστημα και αν ανήκει βρίσκουμε την απόσταση, το μήκος δηλαδή της καθέτου. Η μικρότερη απόσταση που θα βρούμε είναι το κατάστημα στο οποίο είμαστε μπροστά από τη βιτρίνα του, οπότε και αυτό θα στείλει τις προσφορές του στην android συσκευή του χρήστη. Αν καμία από τις καθέτους δεν τέμνει μέσα στα όρια των ευθυγράμμων τμημάτων μας, τότε συμπαιρένουμε ότι ο χρήστης έχει σταματήσει, αλλά δε βρίσκεται μπροστά από κανένα κατάστημα.

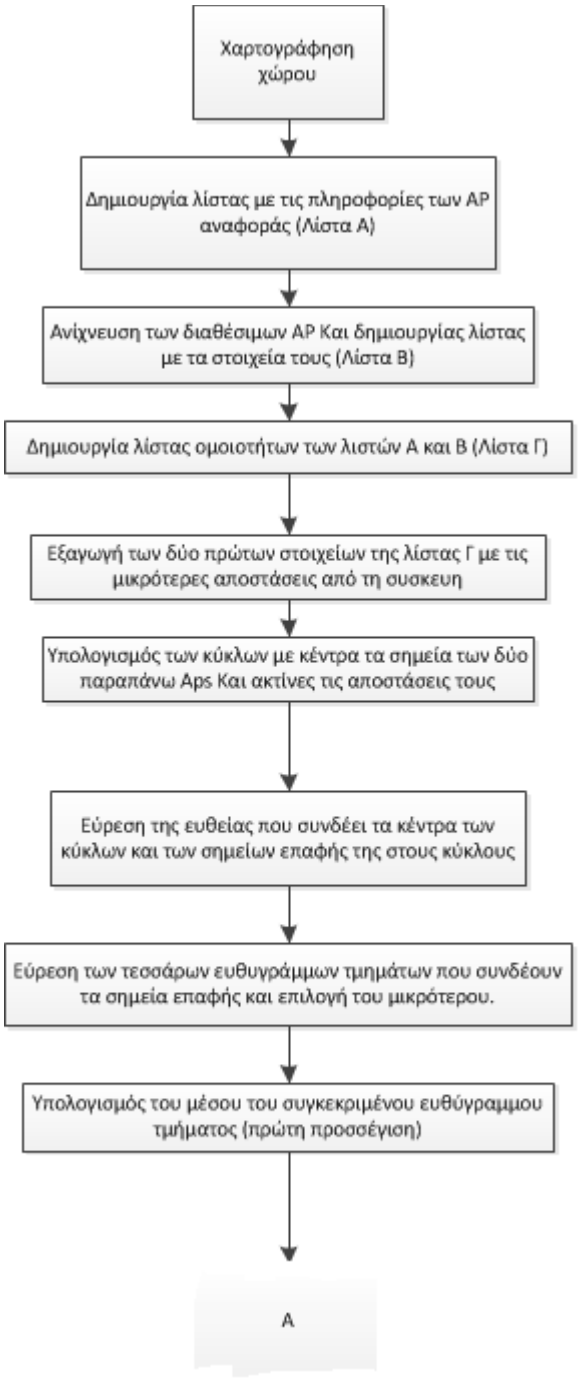
Για παράδειγμα στην εικόνα 3.1.3, τα καταστηματά μας (ή καλύτερα τα σημεία που βρίσκονται τα APs των καταστημάτων) είναι το Α, που έχει βιτρίνα το ευθύγραμμο τμήμα ΓΔ, το Β με βιτρίνα το ΕΖ και το Λ με βιτρίνα το ΘΗ. Έχοντας το σημείο Ξ που βρήκαμε πριν φέρνουμε τις κάθετες στα τρία αυτά ευθύγραμμα τμήματα και βλέπουμε ότι μόνο στα ΕΖ και ΘΗ, η κάθετος τα τέμνει μέσα στα όρια τους (σημεία Ι και Κ

αντίστοιχα) και όχι στην προέκταση τους (σημείο Μ στην προέκταση του ΓΔ). Άρα το ΓΔ απορρίπτεται και μεταξύ των ΕΖ και ΘΗ επιλέγουμε το ΕΖ γιατί η απόσταση ΞΙ είναι μικρότερη της ΞΚ. Άρα συμπεραίνουμε ότι βρισκόμαστε μπροστά από τη βιτρίνα ΕΖ δηλαδή στο κατάστημα Β.

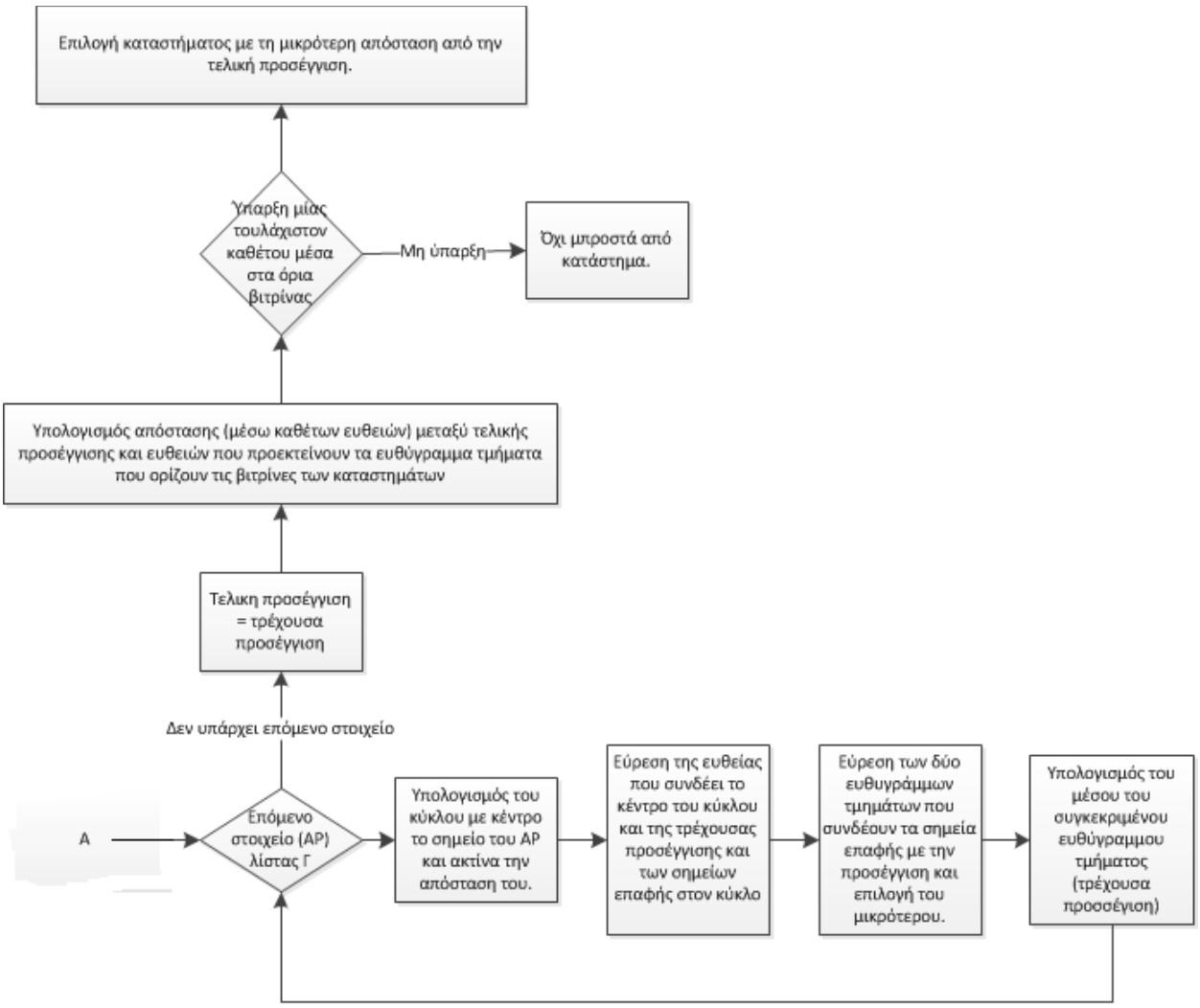


Εικόνα 3.1.3: Εύρεση πλησιέστερου καταστήματος

Στις εικόνες 3.1.4α και 3.1.4β παρακάτω παρουσιάζεται ένα flow diagram με τα βήματα του αλγορίθμου που περιγράψαμε παραπάνω:



Εικόνα 3.1.4α Βήματα αλγορίθμου (flow diagram)



Εικόνα 3.1.4β Βήματα αλγορίθμου (flow diagram)

3.2 Refresh Rate Adaptation

Στο δεύτερο κομμάτι του αλγορίθμου κληθήκαμε να εξετάσουμε το ανά πόσο χρονικό διάστημα θα σκανάρει το κινητό για τα διαθέσιμα APs, καθώς και κάθε πότε θα υπολογίζει την θέση-κατάστημα που βρισκόμαστε. Προφανώς αν αυτά τα δύο γίνονται συνεχώς αυτό θα έχει επίπτωση στην κατανάλωση της μπαταρίας, οπότε φτιάξαμε έναν αρκετά αποδοτικό αλγόριθμο για να περιορίσουμε αυτό το φαινόμενο.

Η βασική ιδέα γύρω από αυτόν τον αλγόριθμο είναι το να ανιχνεύσουμε με κάποιο τρόπο αν ο χρήστης, και κατά συνέπεια η android συσκευή, κινείται ή παραμένει

σε ακινησία. Στην πρώτη περίπτωση δεν θέλουμε να γίνεται ανίχνευση θέσης, αφού ο χρήστης δεν έχει σταματήσει σε κάποια βιτρίνα, συνεπώς δεν πρέπει να λάβει προσφορές από κανένα κατάστημα. Στην περίπτωση όμως που παραμένει ακίνητος για κάποιο χρονικό διάστημα μας ενδιαφέρει να βρούμε σε ποιο σημείο βρίσκεται, και, αν βρίσκεται μπροστά από τη βιτρίνα κάποιου καταστήματος να λάβει ειδοποίηση ότι βρίσκεται εκεί και κατά συνέπεια να λάβει προσφορές από αυτό το κατάστημα.

Ο προβληματισμός μας λοιπόν καταρχήν ήταν στο πως θα μπορέσουμε να ανιχνεύσουμε αν ο χρήστης κινείται. Υπήρχε μία σκέψη να χρησιμοποιήσουμε το Accelerometer του Android, δηλαδή να χρησιμοποιήσουμε στην ουσία τους αισθητήρες της συσκευής για τον εντοπισμό κίνησης, ταχύτητας ή επιτάχυνσης, αλλά τα αποτελέσματα της βιβλιογραφίας δεν ήταν ενθαρρυντικά, καθώς η συγκεκριμένη τεχνολογία δεν παρέχει την απαραίτητη ακρίβεια [25].

Η ανίχνευση λοιπόν της κίνησης στον αλγόριθμο μας γίνεται με έμμεσο τρόπο και μέσω των αποτελεσμάτων της σάρωσης της συσκευής για ασύρματα δίκτυα. Παίρνοντας δύο διαδοχικές σαρώσεις μέσα σε κάποιο αρχικό χρονικό διάστημα, έστω 5 δευτερόλεπτα, μπορούμε να δούμε τις αλλαγές στα επίπεδα λήψης σήματος από τα διάφορα APs.

Ορίζουμε αρχικά έναν μετρητή αλλαγών που θα μας δείξει τελικά πόσο διαφορετική είναι η εικόνα των δύο σαρώσεων. Ως αλλαγή ορίζουμε μια μεταβολή του επιπέδου λήψης σήματος από ένα AP της τάξης των 5 dBm προς τα πάνω ή προς τα κάτω. Για κάθε AP λοιπόν που παρατηρούμε τέτοιου είδους μεταβολή αυξάνουμε τον μετρητή αλλαγών κατά μία μονάδα. Αυτό το συν ή πλην 5 dBm είναι αναγκαίο, καθώς ακόμα και όταν η συσκευή είναι ακίνητη, λόγω διαφόρων παραγόντων θορύβου μπορεί να μας επιστρέψει διαφορετική μέτρηση για την ισχύ που έλαβε από κάποιο AP. Βάζουμε λοιπόν ως όριο τα 5 dBm αφού σπάνια η αλλαγή λόγω θορύβου στο ίδιο σημείο είναι πάνω από αυτό το όριο.

Υπάρχει και η περίπτωση κάποιο AP να εξαφανίζεται εντελώς από την λίστα των APs που επιστρέφει η επόμενη σάρωση, ή κάποιο καινούριο που δεν υπήρχε στην προηγούμενη σάρωση να εμφανίζεται. Σε αυτή την περίπτωση προσθέτουμε μισή μονάδα στο μετρητή αλλαγών για κάθε εμφάνιση ή εξαφάνιση κάποιου AP από τη λίστα. Επίσης, έχουμε βάλει έναν περιορισμό ώστε να ασχολούμαστε κάθε φορά με APs με επίπεδα λαμβανόμενης ισχύος μεγαλύτερα του -90, αφού αυτά με μικρότερη ισχύ μπορούν να έρχονται και να φεύγουν συνεχώς, λογίζονται δηλαδή ως «θόρυβος».

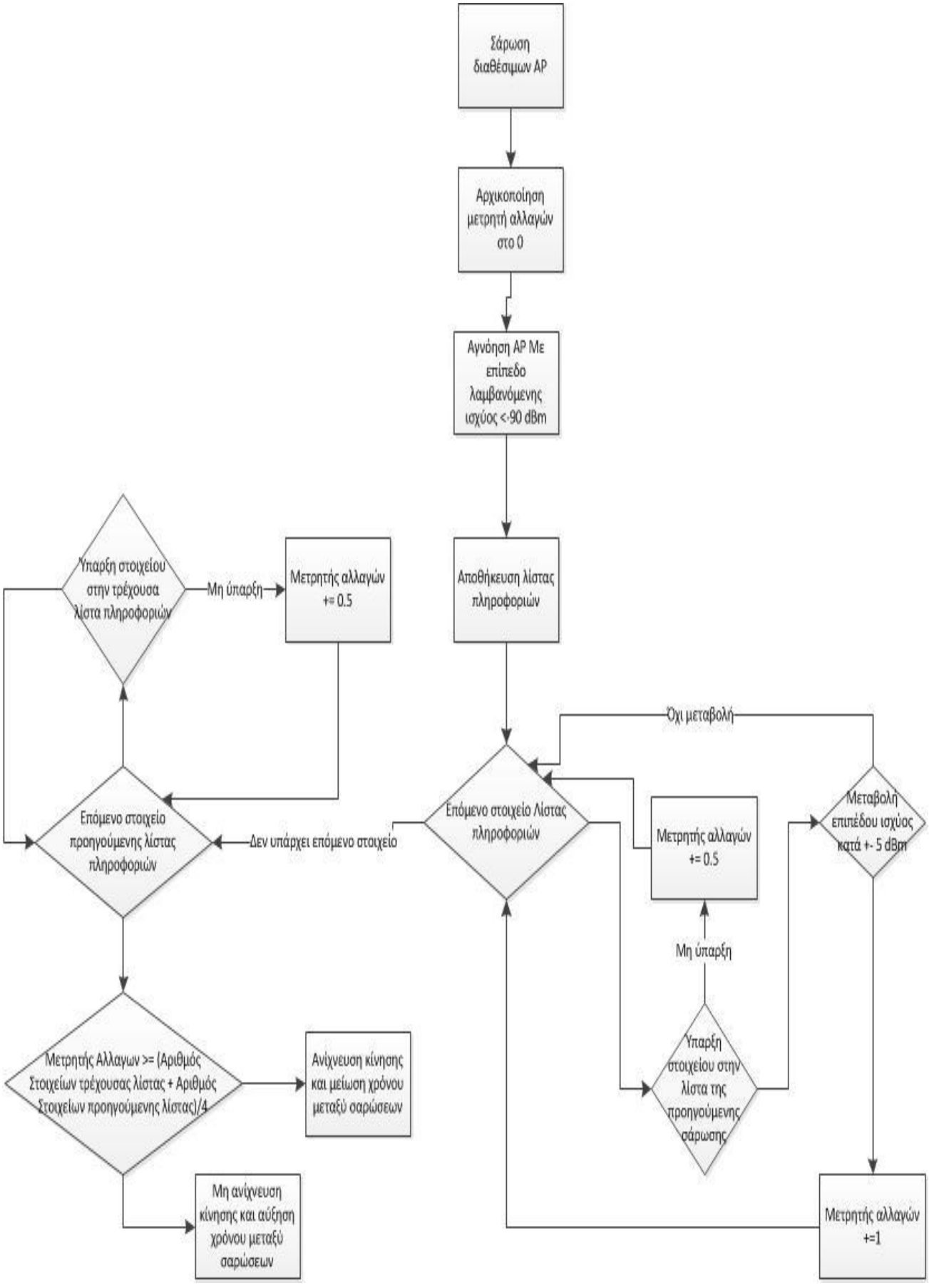
Αφού γίνουν αυτά τα βήματα έχουμε μία συγκεκριμένη τιμή στον μετρητή αλλαγών, την οποία θα χρησιμοποιήσουμε για να δούμε αν κινηθήκαμε ή όχι, δηλαδή αν θα αυξήσουμε ή θα μειώσουμε το χρονικό διάστημα μέχρι την επόμενη σάρωση. Μετά από αρκετές δοκιμές και πειράματα καταλήξαμε ότι το όριο του μετρητή αλλαγών, πάνω από το οποίο συμπεραίνουμε κίνηση είναι το (i+j)/4, οπου i και j συμβολίζουν τα μεγέθη των λιστών που μας επέστρεψε η προηγούμενη και η τρέχουσα σάρωση αντίστοιχα. Δηλαδή αν στην πρώτη σάρωση είχαμε 6 APs και στην δεύτερη είχαμε 5, τότε, αν η τιμή του μετρητή αλλαγών είναι μεγαλύτερη ή ίση του πάνω ορίου του (5+6)/4, δηλαδή του 3, συμπεραίνουμε ότι έχουμε κινηθεί.

Όταν, με βάση τον παραπάνω αλγόριθμο, βρούμε ότι δεν έχουμε κινηθεί τότε το διάστημα μέχρι την επόμενη σάρωση αυξάνεται κατά ένα δευτερόλεπτο. Αν αντίθετα ανιχνεύσουμε κίνηση το διάστημα αυτό μειώνεται στο αρχικό (5 δευτερόλεπτα). Αν τέλος είμαστε ήδη στο αρχικό διάστημα και βρούμε κίνηση μειώνεται ακόμα περισσότερο το διάστημα μέχρι την επόμενη σάρωση. Ο χρόνος αυτός μπορεί να

κυμαίνεται μέσα σε ένα κάτω και πάνω όριο που έχουμε βάλει, δηλαδή δεν μπορεί να πάει κάτω από 2 δευτερόλεπτα, ούτε πάνω από 10.

Τέλος, για το κάθε πότε θα γίνεται ο υπολογισμός της θέσης μας, όπως είπαμε εξαρτάται από το αν είμαστε ακίνητοι και για πόσο χρόνο. Με βάση λοιπόν τον αλγόριθμο μας, αν μετά από τρεις διαδοχικές σαρώσεις δεν έχει ανιχνευτεί κίνηση τότε συμπεραίνουμε ότι ο χρήστης είναι αρκετό χρόνο ακίνητος για να θεωρήσουμε ότι είναι μπροστά από κάποιο κατάστημα, άρα τότε υπολογίζουμε τη θέση του για να βρούμε ποιό κατάστημα είναι αυτό, ή αν είναι σταματημένος σε κάποιο άλλο σημείο εκτός των καταστημάτων.

Στην εικόνα 3.2.1 παρακάτω παρουσιάζεται ένα flow diagram με τα βήματα του αλγορίθμου που περιγράψαμε παραπάνω.



Εικόνα 3.2.1 Βήματα αλγορίθμου Refresh Rate Adaptation (Flow diagram)

Στην εικόνα 3.2.2 φαίνεται ένα παράδειγμα εκτέλεσης της εφαρμογής μας (στο οποίο η συσκευή έμενε ακίνητη στο ίδιο σημείο) για να δείξουμε τον παραπάνω αλγόριθμο:

Στο πρώτο τμήμα της εικόνας (πάνω αριστερά) βλέπουμε ότι έχουμε αρχικά 2 APs και ξεκινάμε με αρχικό sleeptime 5 δευτερόλεπτα (5000msec). Στο επόμενο στιγμιότυπο (πάνω δεξιά) βλέπουμε ότι το sleeptime έχει αυξηθεί σε 6 δευτερόλεπτα, γιατί ο μετρητής αλλαγών (Changes) είναι 0, αφού οι τιμές των σημάτων από τα 2 APs έχουν αλλάξει κατά 3 και 1 dBm αντίστοιχα. Το threshold μας δείχνει από ποια τιμή του μετρητή αλλαγών και πάνω θα συμπεραίναμε κίνηση. Στο τρίτο στιγμιότυπο (κάτω αριστερά) ο μετρητής αλλαγών γίνεται 0.5 αφού προστίθεται ένα νέο AP και το threshold γίνεται 2 ((3+2)/4), ενώ το sleeptime αυξάνεται στο 7000. Τέλος στο τελευταίο στιγμιότυπο (κάτω δεξιά), εφόσον τρεις συνεχόμενες φορές ο μετρητής αλλαγών δεν ξεπέρασε το threshold, υπολογίζεται και η θέση του κινητού, και αν ο χρήστης βρίσκεται μπροστά από κατάστημα θα λάβει εκείνη τη στιγμή τις προσφορές. Αν μετά από το sleeptime των 8 δευτερολέπτων συνεχίσει να μην κινείται απλά θα αυξηθεί ακόμα περισσότερο το sleeptime ενώ όταν αρχίσει να κινείται θα πέσει στο αρχικό των 5 δευτερολέπτων, μέχρι να διαπιστωθεί εκ νέου ακινησία για 3 συνεχόμενα sleeptime.

Εικόνα 3.2.2: Παράδειγμα εκτέλεσης εφαρμογής

4. ΛΕΙΤΟΥΡΓΙΕΣ ANDROID

Το Android είναι ένα λογισμικό ανοιχτού κώδικα για κινητά τηλέφωνα που δημιουργήθηκε από την Google και την OHA (Open Handset Alliance). Σήμερα υπάρχει εγκατεστημένο στην πλειοψηφία των κινητών τηλεφώνων και άλλων κινητών συσκευών, πράγμα που καθιστά το Android μια σημαντική πλατφόρμα για προγραμματιστές εφαρμογών.

Πρόκειται για μια ολοκληρωμένη πλατφόρμα που διαθέτει ένα λειτουργικό σύστημα βασισμένο στο Linux για τη διαχείριση των συσκευών, της μνήμης και των διεργασιών. Μερικά από τα πιο αξιόλογα χαρακτηριστικά του android που το καθιστούν ένα πολύ ενδιαφέρον περιβάλλον ανάπτυξης εφαρμογών είναι:

- Δεν υπάρχουν ειδικά κόστη για άδειες, διανομή και ανάπτυξη λογισμικού.
- Πρόσβαση σε Wi-Fi.
- GSM, EDGE και 3G δίκτυα για τηλεφωνία ή μεταφορά δεδομένων, που δίνουν τη δυνατότητα εισερχόμενων και εξερχόμενων κλήσεων και μηνυμάτων SMS, ή την αποστολή και λήψη δεδομένων μέσω δικτύων κινητής τηλεφωνίας
- Ολοκληρωμένο περιβάλλον προγραμματισμού εφαρμογών για υπηρεσίες με βάση την τοποθεσία, όπως το GPS.
- Πλήρες υλικό ελέγχου πολυμέσων, συμπεριλαμβανομένης της αναπαραγωγής και εγγραφής με την κάμερα και μικρόφωνο.
- Διεπαφές προγραμματισμού εφαρμογών με τη χρήση του υλικού του αισθητήρα, συμπεριλαμβανομένων του επιταχυνσιόμετρου και της πυξίδας.
- Βιβλιοθήκες για τη χρήση του Bluetooth για τη μεταφορά δεδομένων από χρήστη σε χρήστη.
- Ανταλλαγή και μεταφορά μηνυμάτων για την επικοινωνία μεταξύ διαδικασιών.
- Κοινόχρηστες αποθήκες δεδομένων.
- Εφαρμογές και διαδικασίες που τρέχουν στο παρασκήνιο.
- Ένα ολοκληρωμένο πρόγραμμα περιήγησης ανοιχτού κώδικα βασισμένο στο HTML5WebKit.
- Πλήρης υποστήριξη για εφαρμογές που ενσωματώνουν ελέγχους χάρτη, ως μέρος της διεπαφής χρήστη τους.
- Γραφικά βελτιστοποιημένα για κινητά, βασισμένα στην OpenGL.
- Παρέχει μια ελαφριά σχεσιακή βάση δεδομένων για κάθε εφαρμογή που χρησιμοποιεί SQLite, από την οποία οι εφαρμογές μπορούν να επωφεληθούν για την αποθήκευση δεδομένων με ασφάλεια και αποτελεσματικά.
- Βιβλιοθήκες πολυμέσων για την αναπαραγωγή και την καταγραφή μιας ποικιλίας από μορφές ήχου, βίντεο ή εικόνας.
- Ένα πλαίσιο εφαρμογής που ενθαρρύνει την επαναχρησιμοποίηση των συνιστωσών της εφαρμογής και την αντικατάσταση των ιθαγενών εφαρμογών (native apps).

Αξίζει να αναφέρουμε σε αυτό το σημείο πως αν ο αναγνώστης ενδιαφέρεται να γνωρίσει και να πειραματιστεί με το συγκεκριμένο λογισμικό, μπορεί να βρει πλειάδα πληροφοριών στο διαδίκτυο, ενώ αξιόλογα βιβλία όπως τα [26] [27] [28] μπορούν να βοηθήσουν αυτή την προσπάθεια σαν βιβλία αναφοράς. Σε κάθε περίπτωση, η συνεχής εξάσκηση είναι ο καλύτερος σύμμαχος σε ένα τέτοιο εγχείρημα και δε μπορεί να αντικατασταθεί.

Παρακάτω θα γίνει αναφορά στα πιο σημαντικά στοιχεία του Android τα οποία χρησιμοποιήθηκαν στην εργασία μας. Πιο συγκεκριμένα, στο 4.1 θα γνωρίσουμε τα *Activities*, σημαντικές κλάσεις για την οπτικοποίηση της πληροφορίας. Στο 4.2 θα περιγράψουμε συνοπτικά τους τρόπους με τους οποίους μπορούμε να έχουμε εργασίες να τρέχουν στο παρασκήνιο. Στο 4.3 θα παρουσιάσουμε το *WifiManager*, βιβλιοθήκη του Android με λειτουργίες για την ανίχνευση των δικτύων του περιβάλλοντα χώρου.

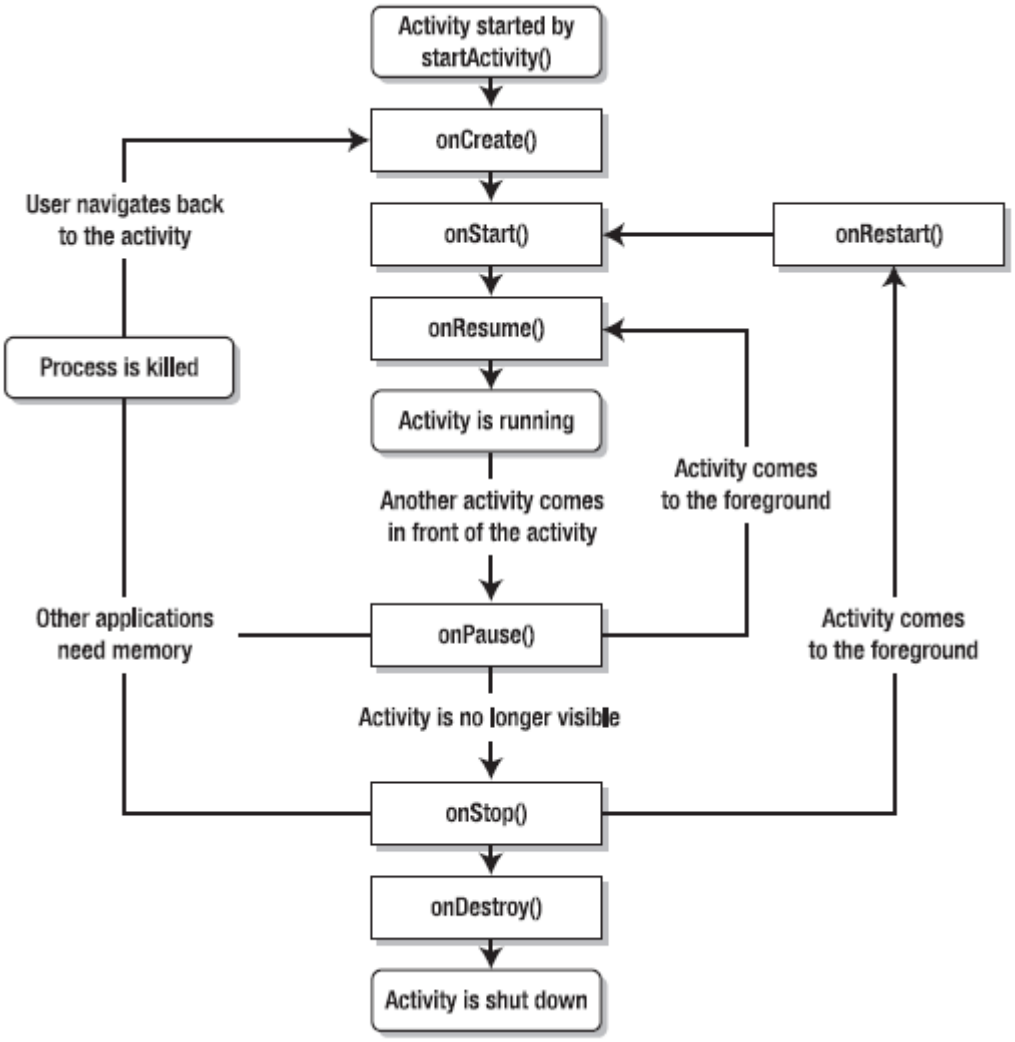
4.1 Activities

Η δραστηριότητα (***Activity***) αποτελεί το στρώμα παρουσίασης της εφαρμογής, έτσι ώστε ο χρήστης να μπορεί να αλληλεπιδρά με αυτή. Για παράδειγμα, η εφαρμογή για τις επαφές του Android περιλαμβάνει μια δραστηριότητα για την είσοδο μιας νέας επαφής, η εφαρμογή τηλεφώνου περιλαμβάνει μια δραστηριότητα για την κλήση ενός αριθμού τηλεφώνου και η εφαρμογή υπολογιστή τσέπης περιλαμβάνει μια δραστηριότητα για την εκτέλεση βασικών υπολογισμών.

Παρά το γεγονός ότι μια εφαρμογή μπορεί να περιλαμβάνει μια ενιαία δραστηριότητα, είναι πιο χαρακτηριστικό για τις εφαρμογές να συμπεριλαμβάνουν πολλαπλές δραστηριότητες. Για παράδειγμα, η εφαρμογή υπολογιστή τσέπης περιλαμβάνει επίσης μια δραστηριότητα «επιστημονικό panel», που επιτρέπει στο χρήστη να υπολογίζει τετραγωνικές ρίζες, τριγωνομετρία, και την εκτέλεση άλλων προηγμένων μαθηματικών πράξεων.

Κάθε δραστηριότητα έχει το δικό της κύκλο ζωής [Εικόνα 4.1.1]. Ακολουθεί συνοπτική περιγραφή κάθε σταδίου:

- onCreate(): Η onCreate καλείται όταν η δραστηριότητα πρωτοξεκινά. Μπορεί να χρησιμοποιηθεί για αρχικοποίηση της διεπαφής χρήστη. Παίρνει μια παράμετρο null ή πληροφορίες κατάστασης που έχουν αποθηκευτεί από τη μέθοδο onSaveInsanceState().
- onStart(): Στην onStart η δραστηριότητα είναι έτοιμη για εμφάνιση στο χρήστη.
- onResume(): Καλείται όταν η δραστηριότητα είναι έτοιμη να αλληλεπιδράσει με το χρήστη.
- onPause(): Εκτελείται όταν η δραστηριότητα είναι έτοιμη να πάει στο παρασκήνιο, συνήθως επειδή μια άλλη δραστηριότητα ξεκινά. Σε αυτή τη μέθοδο πρέπει να έχει προβλέψει ο προγραμματιστής για την αποθήκευση όσων δεδομένων χρειάζεται η εφαρμογή.
- onStop(): Καλείται όταν η δραστηριότητα δεν είναι ορατή πλέον στο χρήστη. Η χρονική στιγμή κλήσης της είναι μη ντετερμινιστική καθώς εξαρτάται από το λειτουργικό σύστημα και όχι από κάποια ενέργεια του χρήστη.
- onDestroy(): Καλείται όταν η δραστηριότητα καταστρέφεται. Η χρονική στιγμή κλήσης της είναι μη ντετερμινιστική καθώς εξαρτάται από την ανάγκη του λειτουργικού συστήματος για επιπλέον πόρους και όχι από κάποια ενέργεια του χρήστη.



Εικόνα 4.1.1 Κύκλος ζωής ενός Activity.

4.2 Background Processes

Service – IntentService

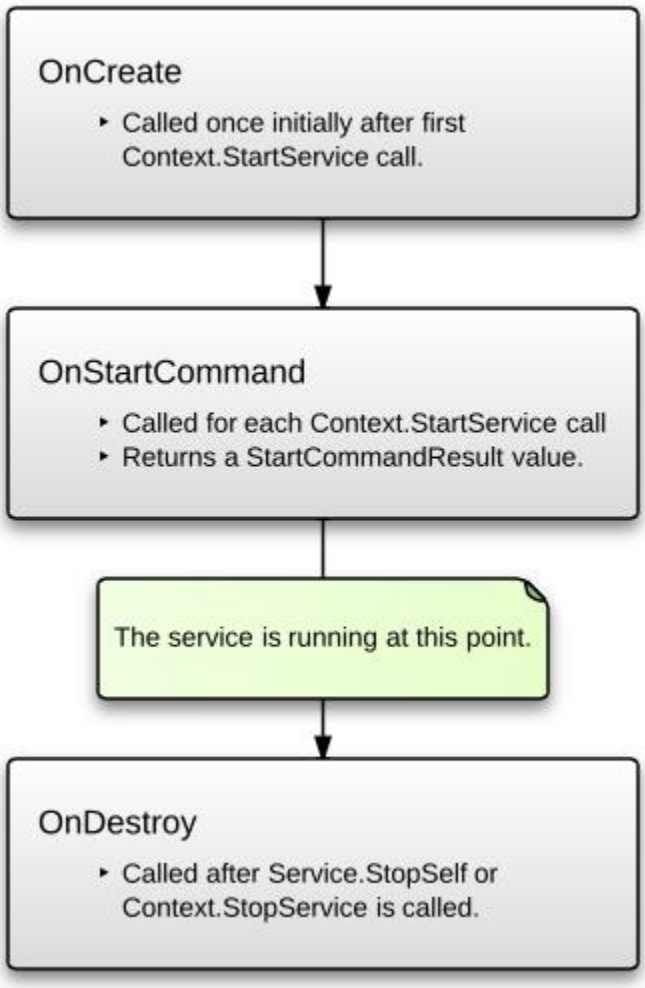
Τα ***Services*** είναι διαδικασίες οι οποίες τρέχουν στο παρασκήνιο με μικρή παρέμβαση από το χρήστη. Ενώ είναι δυνατό να σχεδιαστούν ώστε να είναι απολύτως αόρατα από το χρήστη, γενικά θεωρείται καλύτερο να μπορεί ο τελευταίος να εκτελεί βασικές λειτουργίες που αφορούν τη διαδικασία καλείται να φέρει σε πέρας το service, όπως η έναρξη, παύση η τερματισμό αυτής.

Τα Services έχουν και αυτά με τη σειρά τους ένα κύκλο ζωής, ο οποίος μάλιστα διαφέρει από αυτόν του activity που τα δημιούργησε [Εικόνα 4.2.1]. Αυτό τους επιτρέπει να είναι αυτόνομα και να μπορούν να συνεχίζουν την εργασία τους ακόμα και μετά τον τερματισμό της δραστηριότητας που τα δημιούργησε.

Τα βασικά στάδια ζωής ενός Service έχουν ως εξής:

- onCreate(): Η onCreate καλείται από το σύστημα όταν το service δημιουργείται για πρώτη φορά.

- onStartCommand(Intent intent, int flags, int startId): Η onStartCommand καλείται κάθε φορά που κάποιος πελάτης καλεί το Service με την startService(Intent), παρέχοντας συγχρόνως την αρχική πληροφορία καθώς και ένα μοναδικό αριθμό για να δηλώσει την έναρξη του Service. Αν δεν έχει δημιουργηθεί το Service θα προηγηθεί η κλήση της onCreate αυτόματα. Επιστρέφει ένα στατικό int που δηλώνει αν πρόκειται για service που πρέπει να μένει ενεργό για συγκεκριμένο χρονικό διάστημα (START_STICKY) ή για service που μπορεί να παύει και να συνεχίζει την λειτουργία του σεβόμενο τις ανάγκες των άλλων διαδικασιών του συστήματος για πόρους (START_NOT_STICKY). Ένα παράδειγμα START_STICKY service είναι η εκτέλεση ενός μουσικού κομματιού. Αυτό δε μπορεί να σταματήσει καθώς επικοινωνεί άμεσα με το χρήστη. Ένα παράδειγμα START_NOT_STICKY service είναι αυτό της συλλογής πληροφορίας από απομακρυσμένο server. Αυτή η διαδικασία μπορεί να σταματήσει σε περίπτωση που οι πόροι είναι άμεσα αναγκαίοι από το κινητό, και να συνεχίσει τη λειτουργία του από το σημείο που είχε σταματήσει αργότερα.
- onDestroy(): Η onDestroy καλείται όταν κληθεί η StopService() ή η StopSelf() εσωτερικά του Service, και χρησιμεύει για την καταστροφή του Service.



Εικόνα 4.2.1 Κύκλος ζωής ενός Service.

Το βασικό πλεονέκτημα των services έναντι των άλλων τρόπων εκτέλεσης διαδικασιών στο παρασκήνιο, είναι η υψηλή προτεραιότητα που τους δίνεται. Ο μόνος λόγος για να τερματίσει το Android ένα Service είναι για να δώσει πόρους για κάποιο activity στο προσκήνιο καθώς είναι το μόνα που έχουν μεγαλύτερη προτεραιότητα. Ακόμα και αν κάτι τέτοιο χρειαστεί, υπάρχουν μηχανισμοί για επανεκκίνηση του service και διατήρηση της προόδου του όταν ελευθερωθούν εκ νέου οι πόροι. Επίσης, όταν κάποιο Service κρίνεται εξαιρετικά απαραίτητο, συνήθως όταν επικοινωνεί συχνά με το χρήστη, υπάρχουν η μηχανισμοί για την αύξηση της προτεραιότητας του, κάνοντας το ισόπιμο με τα ενεργά Activities. Αυτό ασφαλίζει την ακεραιότητα του εκτός από ακραίες καταστάσεις, ωστόσο μειώνει το χρόνο εκμετάλλευσης των διαθέσιμων πόρων.

Ο βασικός περιορισμός αυτών είναι πως ακόμα και αν λειτουργούν στο παρασκήνιο, τρέχουν στο βασικό thread της εφαρμογής. Αυτό έχει ως αποτέλεσμα να το επιβαρύνουν με κίνδυνο να το μπλοκάρουν και έτσι να μην αποτελούν καλή λύση αν πρόκειται να λειτουργήσουν για μεγάλο χρονικό διάστημα.

Για την αντιμετώπιση αυτού του περιορισμού κρίθηκε απαραίτητος ο σχεδιασμός του ***IntentService***, μιας κλάσης που ενθυλακώνει την βασική κλάση Service. Υποστηρίζει τη δημιουργία ξεχωριστού νήματος για να μην επιβαρύνεται το βασικό νήμα με όλο το φόρτο της απαιτούμενης εργασίας.

Ακριβώς επειδή αυτά έχουν σχεδιασθεί για να λειτουργούν για μεγάλα χρονικά διαστήματα, η επικοινωνία με το χρήστη είναι συνήθως περιορισμένη. Αν αυτό κριθεί απαραίτητο, υπάρχουν μηχανισμοί όπως τα handlers για να το πραγματοποιήσουν.

Τα IntentServices δεν υποστηρίζουν παράλληλη εργασία. Αυτό οφείλεται στο σχεδιασμό τους, αφού όταν υπάρχουν πολλαπλές κλήσεις του IntentService, αυτό της τοποθετεί σε μια ουρά και τις εξυπηρετεί σειριακά. Έτσι, αν σε κάποια εφαρμογή χρειαστεί να χρησιμοποιήσουμε παραλληλία θα πρέπει αυτό να γίνει μέσω επέκτασης της βασικής κλάσης.

Στη γενική περίπτωση είναι πιο απλό στη χρήση σε σχέση με το Service και βολικό σε αρκετά μεγάλο φάσμα εφαρμογών.

Thread – AsyncTask

Ένας ακόμα τρόπος για εκτέλεση λειτουργιών στο παρασκήνιο είναι με τη χρήση νήματος (***Thread***) που δημιουργείται απευθείας από το activity. Χρησιμοποιούνται συνήθως για την ολοκλήρωση διαδικασιών που αν έτρεχαν στο βασικό νήμα θα μπορούσαν να προκαλέσουν «κολλήματα» και γενικά χαμηλής ποιότητας εμπειρία για το χρήστη.

Με τη δημιουργία του thread μπορούμε να περάσουμε τις αρχικές πληροφορίες και για την περαιτέρω επικοινωνία μεταξύ activity και thread να χρησιμοποιήσουμε handlers. Έτσι, μπορεί να στέλνει περιοδικά πληροφορίες το νήμα εργάτης στο βασικό για την παρουσίαση τους στο χρήστη, ή να στείλει τα βασικό νήμα στον εργάτη πληροφορία για την παύση ή το τερματισμό του.

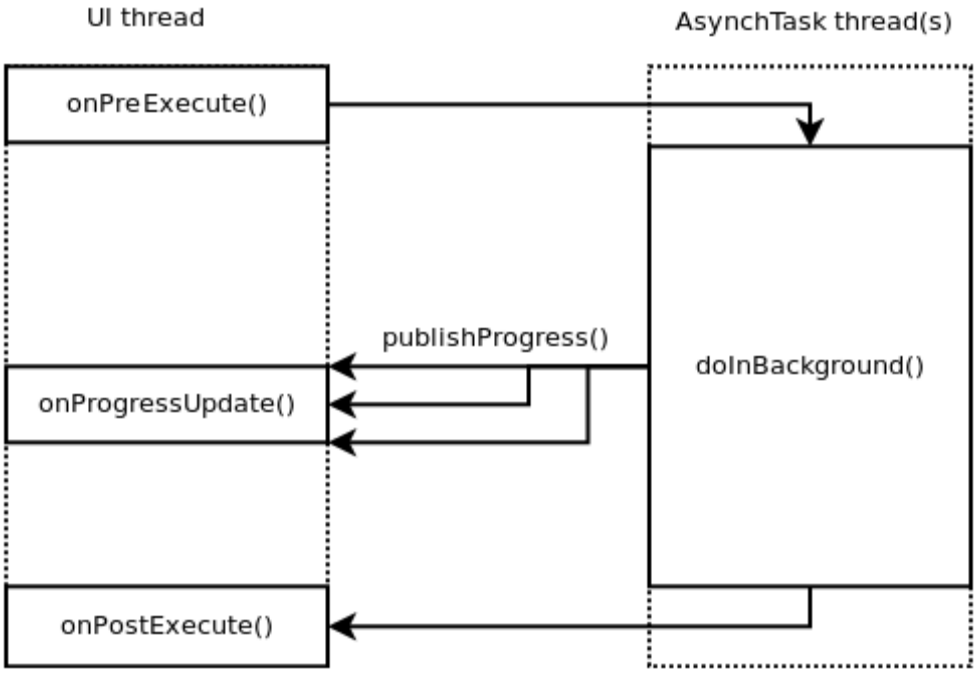
Συγκριτικά με το Service, το thread δεν έχει τόσο μεγάλη προτεραιότητα όσον αφορά τη σημαντικότητα του. Έτσι σε ακραία περίπτωση υπάρχουν πιο πολλές πιθανότητες να επιλεχθεί αυτό για να τερματίσει σε σχέση με κάποιο service αφού βρίσκονται σε διαφορετικά επίπεδα. Στον αντίποδα, η σχεδίαση πολύ-νηματικής εφαρμογής γίνεται πολύ πιο εύκολη από την σχεδίαση με τη χρήση service. Επίσης, είναι πιο οικεία προσέγγιση για κάποιον ο οποίος είναι εξοικειωμένος με τον προγραμματισμό σε Java, αφού έχει ελάχιστες διαφορές από τον προγραμματισμό ενός thread σε Java εφαρμογή.

Το ***AsyncTask*** είναι μια κλάση του Android που ενθυλακώνει τη λειτουργικότητα ενός thread και χρησιμοποιείται γιατί κάνει πιο εύκολο το προγραμματισμό και πιο ευανάγνωστο το κώδικα.

Εγγυάται thread-safe λειτουργίες και σε αντίθεση με τα services συνίσταται σε λειτουργίες του παρασκηνίου που κρατάνε σχετικά μικρό χρονικό διάστημα (σαν εμπειρικό κανόνα δε πρέπει να ξεπερνά ο κύκλος ζωής τους, το κύκλο ζωής των activity που τα δημιούργησε). Περιέχει λειτουργίες επικοινωνίας με το βασικό thread για αναφορά των αποτελεσμάτων χωρίς να είναι αναγκαία η χρήση handlers όπως τα απλά threads.

Τα στάδια του κύκλου ζωής ενός AsyncTask [Εικόνα 4.2.2] έχουν ως εξής:

- onPreExecute(): Καλείται από το βασικό thread πριν την εκτέλεση της ζητούμενης λειτουργίας. Χρησιμοποιείται συνήθως για να προετοιμάσει για την έναρξη όπως την εμφάνιση μιας μπάρας προόδου στο χρήστη.
- doInBackground(Params..): Καλείται από το AsyncTask νήμα αμέσως μετά το τέλος του onPreExecute(). Αποτελεί τη καρδιά του κύκλου ζωής αφού εδώ γίνεται η υπολογιστική πρόοδος της διαδικασίας. Σε αυτό το σημείο μπορεί να γίνει η κλήση της publishProgress(Progress...) η οποία δίνει πληροφορίες στην onProgressUpdate(Progress...), που θα αναλύσουμε παρακάτω.
- onProgressUpdate(Progress...): Καλείται από το βασικό νήμα και παρουσιάζει την πρόοδο του AsyncTask όπως αυτή δοθεί από την publishProgress(Progress...).
- onPostExecute(Result): αυτή η συνάρτηση καλείται μόλις ο υπολογισμός στο παρασκήνιο ολοκληρωθεί. Η παράμετρος Result αποτελεί το τελικό αποτέλεσμα των υπολογισμών.



Εικόνα 4.2.2 Κύκλος ζωής ενός AsynchTask thread και επικοινωνία με το Main Thread.

4.3 WifiManager

Το ***WifiManager*** είναι μια κλάση που χρησιμοποιήσαμε κατά κόρον στην εργασία μας. Η συγκεκριμένη κλάση παρέχει το βασικό API για τη διαχείριση των Wifi συνδέσεων του περιβάλλοντα χώρου. Μερικές από τις δυνατότητες του είναι:

- Παροχή λίστας με τα ρυθμισμένα δίκτυα. Αυτή η λίστα μπορεί να παρουσιαστεί αλλά και να ανανεωθεί με την επεξεργασία των επιμέρους δικτύων.
- Παροχή λίστας με το τρέχον ενεργό δίκτυο. Η ανανέωση αυτής της λίστας μπορεί να γίνεται δυναμικά, ανάλογα με τις απαιτήσεις της εκάστοτε εφαρμογής.
- Παροχή λίστας με σαρώσεις των σημείων πρόσβασης του περιβάλλοντα χώρου. Επίσης επιτρέπεται η συλλογή επιμέρους πληροφορίας, όπως η ισχύς σήματος των δικτύων κάτι που είναι ζωτικής σημασίας για το αλγόριθμο εσωτερικής χωροθέτησης που έχουμε σχεδιάσει.
- Καθορισμός ονομάτων διαφόρων ενεργειών οι οποίες μεταδίδονται από κάθε είδους αλλαγή σε μια Wi-Fi κατάσταση.

5. ΠΕΙΡΑΜΑΤΙΚΑ ΑΠΟΤΕΛΕΣΜΑΤΑ

Προκειμένου να επαληθεύσουμε την ορθότητα της εφαρμογής μας, ρυθμίσαμε ένα πραγματικό περιβάλλον μικρής κλίμακας. Σε αυτό το πλαίσιο τοποθετήσαμε φορητά σημεία πρόσβασης μέσα σε δωμάτια που υποθέσαμε ότι αντιστοιχούν σε καταστήματα του εμπορικού κέντρου, και πήραμε μετρήσεις κινούμενοι σε ένα χώρο, που είχε το ρόλο του διαδρόμου μέσα στο εμπορικό κέντρο. Η εφαρμογή δοκιμάστηκε σε κινητό Samsung Galaxy Ace με έκδοση Android 2.2.1.

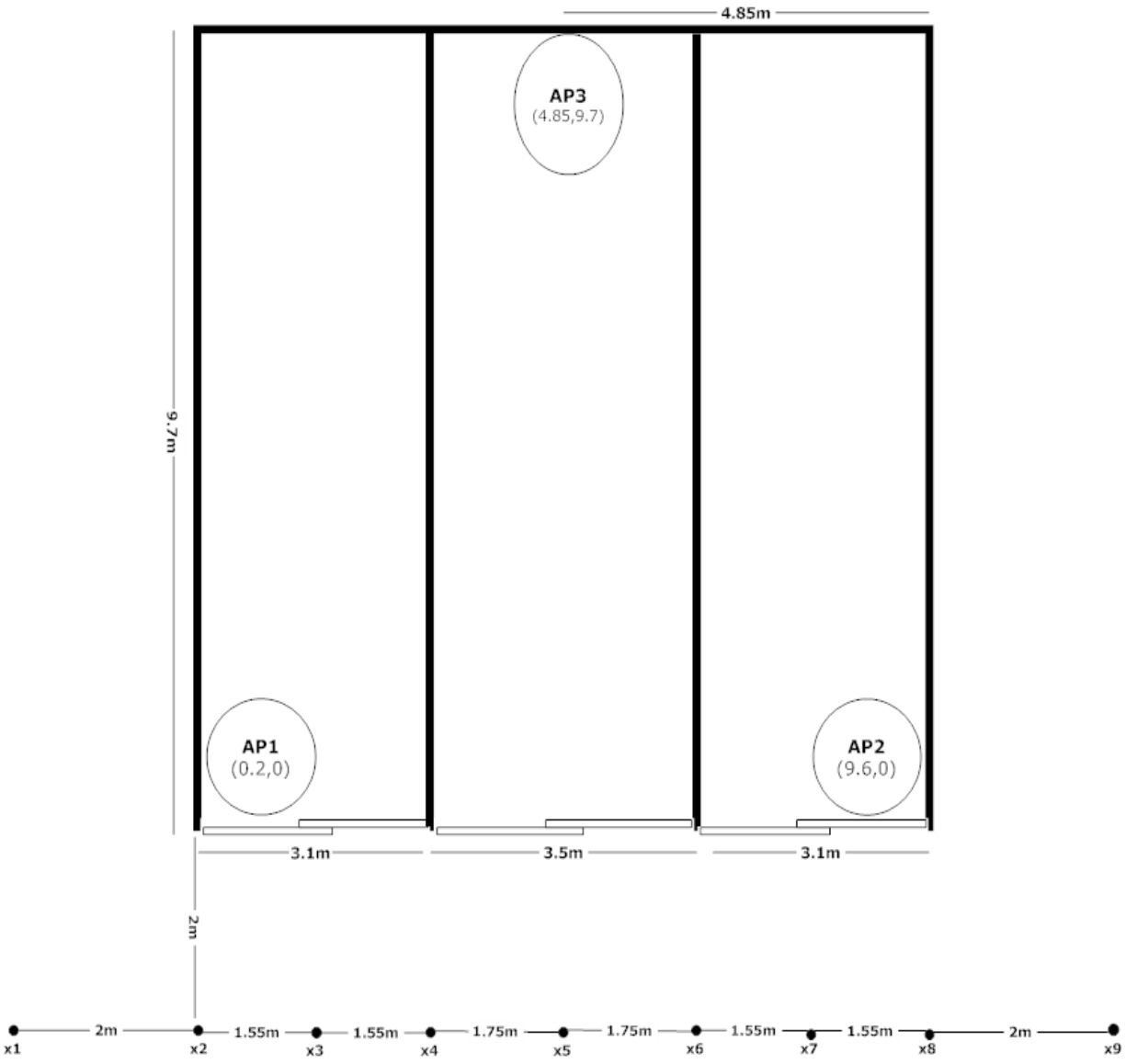
5.1 Περιβάλλον

Το πείραμα μας πραγματοποιήθηκε στο Τμήμα Πληροφορικής και Τηλεπικοινωνιών του Εθνικού Καποδιστριακού Πανεπιστημίου Αθηνών και συγκεκριμένα στην αίθουσα Υ22. Στη συγκεκριμένη αίθουσα οι συνθήκες μπορούν να παρομοιαστούν, σε κάποιο βαθμό, με αυτές ενός εμπορικού καταστήματος, αφού υπάρχουν καταρχήν ξεχωριστά δωμάτια, που χωρίζονται με τζάμια, τα οποία θα μπορούσαν να χαρακτηριστούν ως οι βιτρίνες των διαφόρων καταστημάτων.

Η αίθουσα χωρίστηκε σε τρία δωμάτια-καταστήματα, στα οποία τοποθετήθηκαν οι δρομολογητές μας. Όπως φαίνεται στην εικόνα 5.1 παρακάτω το AP1, γνωστό και ως GYN, τοποθετήθηκε στο σημείο (0.2,0), άρα ουσιαστικά έχουμε πάρει ως σημείο (0,0) στη χαρτογράφηση του χώρου το μπροστινό αριστερότερο σημείο της αίθουσας. Αντίστοιχα το AP2, γνωστο και ως PIK, έχει τοποθετηθεί κοντά στο μπροστινό δεξιότερο σημείο της αίθουσας (9.6,0) και το AP3 (MAK) έχει τοποθετηθεί στο κέντρο τις αίθουσας και στο βάθος της (4.85,9.7).

Ο διάδρομος που εκτελέσαμε τα πειράματα ήταν στα 2 μέτρα έξω από την αίθουσα πάνω σε μια ευθεία ουσιαστικά γραμμή, που δείχνει περίπου πως θα προχωρούσε ο πελάτης από το ένα κατάστημα προς το άλλο μέσα στο εμπορικό κέντρο. Ανάλογα έχουμε πάρει τις μετρήσεις μας στα σημεία x1,x2,…,x9 όπως αυτά φαίνονται στην εικόνα 5.1.1.

Αξίζει να σημειωθεί ότι άλλος ένας λόγος που οι συνθήκες στην συγκεκριμένη αίθουσα είναι ανάλογες ενός εμπορικού καταστήματος είναι η συνεχής παρουσία και κίνηση κόσμου μέσα στην αίθουσα, όπου εργάζονται κατά κύριο λόγο οι μεταπτυχιακοί και ερευνητές του τμήματος. Έτσι υπάρχει σε μόνιμη βάση παρουσία κόσμου αλλά και κινητών συσκευών, υπολογιστών και άλλων συσκευών που μπορούν να δημιουργήσουν παρεμβολές στις μετρήσεις. Όπως θα δούμε λοιπόν και στις μετρήσεις παρακάτω αυτό οδηγεί στο να δημιουργούνται σε ορισμένες περιπτώσεις λάθη, λόγω θορύβου, και οι μετρήσεις της ισχύος των σημάτων που λαμβάνονται από τα σημεία πρόσβασης στο ίδιο σημείο να αλλάζουν σημαντικά από τη μία στιγμή στην άλλη.



Εικόνα 5.1.1: Κάτοψη χώρου διενέργειας πειραμάτων

5.2 Παράμετροι

Κατά τη διεξαγωγή των πειραμάτων χρησιμοποιήθηκαν ορισμένες παράμετροι, στις οποίες προσπαθήσαμε να δώσουμε τις κατάλληλες τιμές ώστε να βελτιστοποιήσουμε τα αποτελέσματα που δίνει η εφαρμογή. Ορισμένες από αυτές έχουν αναφερθεί και στο κεφάλαιο 3, αλλά τις αναφέρουμε και εδώ όλες μαζί συγκεντρωτικά.

- Καταρχήν έχουμε τρείς παραμέτρους, τις maxsleeptime, minsleeptime και initsleeptime, οι οποίες αναφέρονται στο μέγιστο, ελάχιστο και αρχικό χρόνο αναμονής μεταξύ δύο σαρώσεων για σημεία πρόσβασης αντίστοιχα. Το initsleeptime έχει οριστεί στα 5 δευτερόλεπτα (5000msec) και όσο αυτό αλλάζει με τον τρόπο που περιγράψαμε στο 3β, δε μπορεί να ξεπεράσει το maxsleeptime, που ορίζουμε στα 10 δευτερόλεπτα, ούτε και να πέσει κάτω από το minsleeptime, που είναι ορισμένο στα 2 δευτερόλεπτα.
- Έχουμε επίσης την παράμετρο levelthreshold, η οποία, με βάση τον αλγόριθμό μας για το πότε εντοπίζουμε αλλαγή (περιγράφτηκε επίσης στο 3β), είναι το όριο

μεταξύ δύο διαδοχικών σαρώσεων, κατά το οποίο όταν αλλάξει κάποια λαμβανόμενη ισχύς από κάποιο σημείο πρόσβασης, αυξάνεται ο μετρητής αλλαγών. Αυτό έχει οριστεί σε 5 και αυτό σημαίνει ότι αν αλλάξει κάποια τιμή λαμβανόμενης ισχύος από κάποιο σημείο πρόσβασης κατά +- 5 dBm, τότε αυξάνεται ο μετρητής αλλαγών κατά ένα.

- Η παράμετρος constant, που έχει οριστεί σε 4, χρησιμοποιείται στον έλεγχο για το αν έχουμε κίνηση ή όχι με βάση το μετρητή αλλαγών. Όπως έχουμε περιγράψει παραπάνω το όριο του μετρητή αλλαγών, πάνω από το οποίο συμπεραίνουμε κίνηση είναι το (i+j)/constant, οπου i και j συμβολίζουν τα μεγέθη των λιστών που μας επέστρεψε η προηγούμενη και η τρέχουσα σάρωση αντίστοιχα. Η τιμή 4 προέκυψε μετά από αρκετές δοκιμές καθώς αν μειωθεί ανιχνεύεται κίνηση πολύ δύσκολα, αφού το όριο μεγαλώνει, ενώ αν αυξηθεί, ανιχνεύεται κίνηση συνήθως χωρίς να έχουμε μετακινηθεί ουσιαστικά.
- Η παράμετρος maxcount, που έχει οριστεί σε 3, αντιστοιχεί στο μέγιστο αριθμό επαναλήψεων μέχρι να γίνει ο υπολογισμός της θέσης. Δηλαδή αν μεταξύ τριών διαδοχικών σαρώσεων δεν έχει διαπιστωθεί κίνηση, τότε στην 4η σάρωση θα γίνει και ο υπολογισμός της θέσης που έχει σταματήσει ο χρήστης.
- Ως παράμετροι θα περαστούν και τα γνωστά σε εμάς APs, και συγκεκριμένα θα δημιουργηθεί η λίστα A (βλέπε 3α), η οποία αναφέρεται στον κώδικα ως knownaps και στην οποία αποθηκεύονται οι mac διευθύνσεις (BSSID) των σημείων πρόσβασης, οι συντεταγμένες των ίδιων των APs, καθώς και του αριστερού και δεξιού ορίου της βιτρίνας του καταστήματος στο οποίο βρίσκεται το καθένα, και τέλος την ισχύ εκπομπής για κάθε AP, το οποίο θα σχολιάσουμε παρακάτω.
- Η ισχύς εκπομπής των δρομολογητών, χρησιμοποιείται στον τύπο του [29] για τον υπολογισμό της απώλειας μετάδοσης (path loss) και μέσω αυτής της απόστασης μας από το εκάστοτε AP. Με κατάλληλες μετρήσεις πάνω στους τρεις δρομολογητές μας βρήκαμε ότι η ισχύς εκπομπής είναι περίπου στο -16 για όλους. Έτσι στο πείραμα μας τα tx1, tx2, tx3 είναι μεταξύ τους ίσα και ισούνται με -16.
- Τέλος το constant2, που ισούται με 32.4 είναι η σταθερά που χρησιμοποιείται στον τύπο που βρήκαμε και χρησιμοποιήσαμε από το [29]**Error! Reference source not found.**.

5.3 Μετρήσεις

Παρακάτω, στην εικόνα 5.3.1 φαίνονται τα αποτελέσματα των μετρήσεων που πήραμε στις συγκεκριμένες θέσεις x1,x2,…,x9 που φαίνονται στην εικόνα 5.1.1 με τις παραμέτρους που περιγράψαμε παραπάνω. Για κάθε μία από τις 9 θέσεις πήραμε δύο μετρήσεις. Με πράσινο χρώμα φαίνονται οι σωστές μετρήσεις, με κόκκινο οι λανθασμένες και με κίτρινο αυτές που είναι «λανθασμένα σωστές».

- Στο σημείο x1 (-2,-2) η πρώτη μέτρηση μας έδωσε το σημείο (-9.96,-1), το οποίο θα μας εμφανίσει σωστά ότι δεν είμαστε έξω από κανένα κατάστημα, αλλά γενικά δεν είναι τόσο καλή μέτρηση και αυτό οφείλεται σε μεγάλο θόρυβο στις τιμές της ισχύος των λαμβανόμενων σημάτων από τα σημεία πρόσβασης. Η δεύτερη μέτρηση πάντως δίνει πιο σωστό αποτέλεσμα (-0.65,2.52)
- Στο σημείο x2 (0,-2) ουσιαστικά στεκόμαστε στο αριστερότερο όριο της βιτρίνας του πρώτου καταστήματος, οπότε είτε μας εμφανίσει ότι βρισκόμαστε έξω από κανένα κατάστημα, είτε ότι είμαστε έξω από το πρώτο δεν υπάρχει πρόβλημα. Η πρώτη μέτρηση (0.63,-0.63) μας λέει ότι βρισκόμαστε έξω από το πρώτο κατάστημα, οπότε είναι αποδεκτή, η δεύτερη όμως μας δείχνει ότι βρισκόμαστε έξω από το δεύτερο κατάστημα οπότε είναι λανθασμένη.
- Στο σημείο x3 (1.55,-2) οι 2 μετρήσεις μας είναι σωστές, καθώς βρισκόμαστε ακριβώς στο κέντρο της βιτρίνας του πρώτου καταστήματος και αυτό ακριβώς θα δείξει και η εφαρμογή μας και τις δύο φορές.
- Στο σημείο x4 (3.1,-2) βρισκόμαστε στο όριο των βιτρίνων πρώτου και δεύτερου καταστήματος και μας εμφανίζει και στις δύο μετρήσεις ότι βρισκόμαστε έξω από το δεύτερο κατάστημα, που είναι αποδεκτό.
- Στο σημείο x5 (4.85,-2) βρισκόμαστε στο κέντρο της βιτρίνας του δεύτερου καταστήματος και εδώ έχουμε λάθος στην πρώτη μέτρηση,αφού μας βγάζει οριακά αριστερά από το δεύτερο κατάστημα και έξω από τη βιτρίνα του πρώτου, ενώ η δεύτερη μέτρηση είναι σωστή.
- Στο σημείο x6 (6.6,-2) βρισκόμαστε στο όριο των βιτρίνων δεύτερου και τρίτου καταστήματος και μας εμφανίζει και στις δύο μετρήσεις ότι βρισκόμαστε έξω από το δεύτερο κατάστημα, που είναι αποδεκτό.
- Στο σημείο x7 (8.15,-2) είμαστε στο κέντρο της βιτρίνας του τρίτου καταστήματος και έχουμε και τις 2 μετρήσεις να μας δείχνουν σωστά ότι είμαστε εκεί.
- Στο σημείο x8 (9.7,-2) είμαστε στο δεξί όριο της βιτρίνας του τρίτου καταστήματος και οι μετρήσεις μας δείχνουν αρκετά μακριά και έξω από κανένα κατάστημα, πράγμα όμως αδιάφορο.
- Τέλος στο σημείο x9 (11.7,-2) είμαστε 2 μέτρα δεξιά από τη βιτρίνα του τρίτου καταστήματος και σωστά οι μετρήσεις μας δείχνουν ότι δεν είμαστε έξω από κάποιο κατάστημα.

	Position		Received Signal Strength			Results	
Point	x axis	y axis	AP1 GYN	AP2 PIK	AP3 MAK	Estimated x	Estimated y
x1	-2	-2	-77	-80	-82	-9,96	-1
			-67	-78	-75	-0,65	2,52
x2	0	-2	-56	-75	-77	0,63	-0,63
			-59	-68	-76	3,49	-0,45
x3	1,55	-2	-61	-74	-79	1,04	-2,14
			-61	-72	-74	2,74	0,62
x4	3,1	-2	-61	-68	-68	3,98	2,53
			-62	-71	-74	3,15	0,58
x5	4,85	-2	-66	-73	-68	2,86	3,7
			-61	-67	-67	4,24	2,57
x6	6,6	-2	-72	-70	-71	6,37	3,59
			-71	-67	-69	5,6	4,25
x7	8,15	-2	-75	-65	-79	8,49	-2,18
			-74	-63	-75	7,52	0,22
x8	9,7	-2	-80	-61	-80	13,09	-3,73
			-81	-59	-76	13,24	1
x9	11,7	-2	-78	-60	-78	11,32	-1,86
			-79	-58	-80	13,61	-1,53

Εικόνα 5.3.1: Πίνακας αποτελεσμάτων

5.4 Συμπεράσματα

Σε γενικές γραμμές, τα αποτελέσματα του παραπάνω πειράματος ήταν αρκετά ικανοποιητικά. Υπάρχει βέβαια το ενδεχόμενο αρκετές φορές οι μετρήσεις να μην είναι τόσο ακριβείς και γι αυτό ευθύνονται οι τιμές που παίρνει το κινητό σε ότι αφορά την λαμβανόμενη ισχύ από τα διάφορα APs. Αυτό γίνεται εμφανές, αν κοιτάξουμε τις δύο ξεχωριστές μετρήσεις που έχουμε πάρει σε κάθε σημείο για τα διάφορα σημεία.

Για παράδειγμα, στο x1, στο οποίο έχουμε μια λάθος μέτρηση, παρατηρούμε ότι στο AP1 έχουμε την πρώτη φορά -77 και την δεύτερη, που είναι η σωστή, -67. Ανάλογες διακυμάνσεις μικρότερου μεγέθους έχουμε και στα AP2 και AP3 και γι αυτό τα αποτελέσματα διαφέρουν, με το δεύτερο να είναι πολύ πιο κοντά στην πραγματικότητα. Αυτές οι διακυμάνσεις, είναι πολύ πιθανό να οφείλονται σε παρεμβολές, όπως κινήσεις ανθρώπων μέσα στο χώρο, σήματα άλλων κινητών και άλλων συσκευών, τα οποία δημιουργούν θόρυβο

Παρομοίως, στο σημείο x6 βλέπουμε πολύ μικρότερες διακυμάνσεις των τιμών του λαμβανόμενου σήματος και στα 3 APs, οπότε τα σημεία που παίρνουμε ως αποτέλεσμα είναι κοντά το ένα στο άλλο και μας δίνουν σωστές μετρήσεις.

Το παραπάνω πρόβλημα, της μειωμένης ακρίβειας των αποτελεσμάτων εξαιτίας διαφορετικών παραγόντων όπως η καθυστέρηση σήματος λόγω εμποδίων μεταξύ πομπού και δέκτη, αποτελεί μείζον πρόβλημα των περισσότερων προσπαθειών

ανάπτυξης συστήματος εσωτερικής χωροθέτησης με χρήση Wlan τεχνολογίας, όπως έχει αναφερθεί άλλωστε και στο δεύτερο κεφάλαιο. Παρόπι έχει περιγραφεί και αναλυθεί από πολυάριθμες πηγές [30] , οι λύσεις που έχουν προτάθει δεν είναι καθολικές και μπορούν να υλοποιηθούν μόνο υπό προϋποθέσεις. Συνοπτικά αναφέρουμε πως ενώ αρκετοί συγγραφεί προτείνουν την δημιουργία ενός χάρτη σημάτων όπου θα μπορούν να καταγραφούν οι ιδιατερότητες του κτιρίου [31] , κάτι τέτοιο μπορεί να αποδειχθεί μη αποτελεσματικό σε περιβάλλοντα με συχνές δυναμικές αλλαγές, όπως είναι ένα νοσοκομείο ή ένα εμπορικό κέντρο στη περίπτωσή μας.

6. ΕΠΙΛΟΓΟΣ

6.1 Γενικά συμπεράσματα

Υλοποιώντας την συγκεκριμένη εφαρμογή και ψάχνοντας τρόπους να επιτευχθεί όσο γίνεται μεγαλύτερη ακρίβεια, συνειδητοποιήσαμε ότι είναι αρκετά δύσκολος ο υπολογισμός της ακριβούς θέσης, ειδικά όταν στο χώρο που γίνεται η μέτρηση έχουμε μεγάλη κίνηση ανθρώπων και πολλές συσκευές που δημιουργούν παρεμβολές και θόρυβο.

Λόγω των παρεμβολών αυτών, οι μετρήσεις της λαμβανόμενης ισχύος από τα διάφορα AP στο κινητό είναι πολλές φορές λανθασμένες και αυτό επηρεάζει και τα δύο κομμάτια της εφαρμογής μας: το refresh rate αφού όταν είμαστε στο ίδιο σημείο μπορεί να λάβουμε πολύ διαφορετική ισχύ από ένα ή περισσότερα AP και αυτό λανθασμένα να οδηγήσει στο συμπέρασμα ότι έχουμε κίνηση, και τον υπολογισμό της θέσης αφού, όπως εξηγήσαμε παραπάνω, είναι πολύ βασικό να έχουμε τις σωστές μετρήσεις λαμβανόμενης ισχύος ώστε να βρούμε σωστή θέση.

Εφόσον όμως στην εφαρμογή μας ασχολούμαστε με βιτρίνες, δεν μας ενδιαφέρει τόσο πολύ να βρούμε ακριβώς το σημείο που βρισκόμαστε, αλλά να υπάρχει μια συγκεκριμένη ακρίβεια, ώστε αν είμαστε για παράδειγμα μπροστά από κάποιο κατάστημα, να υπολογίζεται μια θέση που είναι μέσα στα όρια της βιτρίνας του συγκεκριμένου καταστήματος. Αυτό το κομμάτι σε μεγάλο βαθμό έχει επιτευχθεί.

6.2 Παραδοχές-περιορισμοί

Όπως έχουμε αναφέρει, υπάρχουν ορισμένοι περιορισμοί και παραδοχές που έχουμε κάνει στην εφαρμογή μας, ώστε να μπορέσει να λειτουργήσει σωστά.

Καταρχήν, πρέπει πάντα να γίνεται χαρτογράφηση του χώρου στον οποίο θέλουμε να τρέξουμε την εφαρμογή, και για κάθε AP να δίνεται η διεύθυνση mac του, οι συντεταγμένες του στον δισδιάστατο χώρο, καθώς και οι συντεταγμένες του καταστήματος μέσα στο οποίο βρίσκεται.

Επίσης, ένα σημαντικό ζήτημα για την ακρίβεια των αποτελεσμάτων είναι το πόσο μεγάλο θα είναι το sleeptime, δηλαδή ο χρόνος μεταξύ των σαρώσεων για ανίχνευση κίνησης. Αν αυτό είναι για παράδειγμα πολύ μεγάλο, τότε μπορεί να παραλείψουμε ότι κάποιος χρήστης είχε σταματήσει για έναν αρκετό χρόνο μπροστά από ένα κατάστημα, αφού μέχρι να γίνει η επόμενη σάρωση μπορεί να έχει μετακινηθεί και να έχει φύγει από αυτό. Πρέπει λοιπόν να προσαρμοστούν κατάλληλα οι παράμετροι maxsleeptime και maxcount, έτσι ώστε να μην περνάει πολύ σημαντικός χρόνος μέχρι να αποφασίσει η εφαρμογή αν ο χρήστης βρίσκεται μπροστά από κάποιο κατάστημα ή όχι.

Ένας ακόμη περιορισμός που αναφέραμε παραπάνω είναι αυτός των παρεμβολών λόγω κίνησης, οπότε μπορούμε να πούμε ότι όσο περισσότερος κόσμος και γενικά εμπόδια υπάρχουν στο χώρο που τρέχουμε την εφαρμογή, τόσο περισσότερα τα λάθη.

6.3 Επόμενα βήματα

Ένα βήμα που θα μπορούσε να βοηθήσει πολύ στη βελτίωση της εφαρμογής μας είναι να ερευνηθούν μέθοδοι και τεχνικές ώστε να μπορέσει να περιοριστεί το φαινόμενο των διακυμάνσεων στις τιμές των λαμβανόμενων ισχύων, και να παίρνουμε μετρήσεις που είναι πιο κοντά στην πραγματικότητα, ώστε να βρίσκουμε πιο ακριβή θέση. Εφαρμόζοντας κάποιες από τις λύσεις που έχουν προταθεί κατά καιρούς [31] θα μπορούσε να επιτευχθεί κάτι τέτοιο, θα πρέπει όμως να είμαστε αρκετά προσεκτικοί στην υιοθέτηση κάποιας λύσης καθώς θα πρέπει να μπορεί να προσαρμοστεί στις ειδικές συνθήκες της συγκεκριμένης εφαρμογής.

ΠΙΝΑΚΑΣ ΟΡΟΛΟΓΙΑΣ

Ξενόγλωσσος όρος	**Ελληνικός Όρος**
Positioning	χωροθέτηση
indoor positioning	εσωτερική χωροθέτηση
Wi-fi	τεχνολογία ασύρματης πρόσβασης στο διαδίκτυο
refresh rate	ρυθμός ανανέωσης
access points	σημεία πρόσβασης
ambiguity problem	προβλημα αμφιβολίας
Audible Sound Based Systems	συστήματα βασιζόμενα στον ήχο
Ultra Wideband	τεχνολογία υπερ-ευρείας ζώνης
Mapping	χαρτογράφηση
tx-power	ισχύς εκπομπής
native applications	ιθαγενείς εφαρμογές
Activity	δραστηριότητα
Thread	νήμα
path loss	απώλεια μετάδοσης

ΣΥΝΤΜΗΣΕΙΣ – ΑΡΚΤΙΚΟΛΕΞΑ – ΑΚΡΩΝΥΜΙΑ

GPS	Global Positioning System
AP	Access Point
TOA	Time of Arrival
TDOA	Time Difference of Arrival
RSS	Received Signal Strength
AoA	Angle of Arrival
COO	Cell of Origin
RF	Radio Frequency
RFID	Radio Frequency Identification
GNSS	Global Navigation Satellite System
OHA	Open Handset Alliance

ΠΑΡΑΡΤΗΜΑ Ι

Κώδικας Υλοποίησης

Ο κώδικας είναι χωρισμένος σε 3 πακέτα (packages) και σε 5 αρχεία .java. Παρακάτω παρατίθενται τα περιεχόμενα του καθενός.

Package android.ip:

- **MainActivity.java**

```
package android.ip;


import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.ip.R;
import java.util.logging.Level;
import java.util.logging.Logger;


public class MainActivity extends Activity{


	private static String TAG = "MainActivity";


	@Override
	protected void onCreate(Bundle savedInstanceState) {
		// TODO Auto-generated method stub


		super.onCreate(savedInstanceState);
	    setContentView(R.layout.main);
```

```
            //  Button b1=(Button)findViewById(R)

            // m = Memory.getSingleInstance(getApplicationContext());

                final StackTraceElement[] ste = Thread.currentThread().getStackTrace();

                Log.i(TAG, "Called " + ste[2].getMethodName()+  " of
"+this.getClass().getName());

        }


        @Override

        protected void onDestroy() {

                super.onDestroy();

                final StackTraceElement[] ste = Thread.currentThread().getStackTrace();

                Log.i(TAG, "Called " + ste[2].getMethodName()+  " of
"+this.getClass().getName());

        }




        @Override

        protected void onPause() {

                // TODO Auto-generated method stub

                super.onPause();

                final StackTraceElement[] ste = Thread.currentThread().getStackTrace();

                Log.i(TAG, "Called " + ste[2].getMethodName()+  " of
"+this.getClass().getName());

        }


        @Override

        protected void onResume() {

                // TODO Auto-generated method stub

                super.onResume();

                final StackTraceElement[] ste = Thread.currentThread().getStackTrace();
```

```
		Log.i(TAG, "Called " + ste[2].getMethodName()+  " of
"+this.getClass().getName());

	}


	@Override

	protected void onStart() {

		// TODO Auto-generated method stub

		super.onStart();

		final StackTraceElement[] ste = Thread.currentThread().getStackTrace();

		Log.i(TAG, "Called " + ste[2].getMethodName()+  " of
"+this.getClass().getName());

	}


	@Override

	protected void onStop() {

		// TODO Auto-generated method stub

		super.onStop();

		final StackTraceElement[] ste = Thread.currentThread().getStackTrace();

		Log.i(TAG, "Called " + ste[2].getMethodName()+  " of
"+this.getClass().getName());

	}



    public void scanAPs(View view)

    {

      Class ourClass = null;

      try {

            ourClass =
Class.forName(getResources().getText(R.string.CorePackage)  +".ScanAPs");

                  Intent ourIntent  = new Intent(this,ourClass);

                  ourIntent.setFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
```

```
                    startActivity(ourIntent);
                }
            catch (ClassNotFoundException ex)
            {
                Logger.getLogger(MainActivity.class.getName()).log(Level.SEVERE, null, ex);
            }
        }
    }
```

- **ScanAPs.java**

```
package android.ip;

import android.app.Activity;
import android.ip.tasks.APScanner;
import android.os.Bundle;
import android.os.Handler;
import android.os.Message;
import android.util.Log;
import android.view.View;
import android.widget.TextView;
import android.widget.Toast;
import android.ip.memory.Memory;
import java.util.List;
import java.util.logging.Level;
import java.util.logging.Logger;

public class ScanAPs extends Activity{

    private static String TAG = "GeneralInfo";
    private TextView testview;
```

```
private Boolean buttonPressed =false;
private APScanner wt;
private Thread t;
private Memory m;
private final Handler handler = new Handler() {
        @Override
        public void handleMessage(Message msg) {
                Bundle b = msg.getData();
                List ssid = b.getStringArrayList("ssid");
          List level = b.getIntegerArrayList("level");
          List bssid = b.getStringArrayList("bssid");
          List distances = b.getStringArrayList("distances");

          long sleeptime = b.getLong("sleeptime");

          int newaps = b.getInt("newaps");
          int aps = b.getInt("aps");
          double changes = b.getDouble("changes");
          int threshold = b.getInt("threshold");
          int count = b.getInt("count");
          double magicx=b.getDouble("magicx");
          double magicy=b.getDouble("magicy");
          double newmagicx=b.getDouble("newmagicx");
          double newmagicy=b.getDouble("newmagicy");
          double distfromA=b.getDouble("distfromA");
          double distfromB=b.getDouble("distfromB");
          double dist1=b.getDouble("dist1");
          double dist2=b.getDouble("dist2");
          double distancefromshop=b.getDouble("distancefromshop");
          double distancefromAP=b.getDouble("distancefromAP");
```

```
double magicx2=b.getDouble("magicx2");
double magicy2=b.getDouble("magicy2");
String closestshop=b.getString("closestshop");
String bla=b.getString("msg");

double x1=b.getDouble("x1");
double x2=b.getDouble("x2");
double x3=b.getDouble("x3");
double x4=b.getDouble("x4");
double y1=b.getDouble("y1");
double y2=b.getDouble("y2");
double y3=b.getDouble("y3");
double y4=b.getDouble("y4");
if(ssid!=null && level!=null)
{
   testview.getEditableText().clear();
   for(int i=0; i<ssid.size(); i++)
   {
      //testview.append(ssid.get(i)+"  "+bssid.get(i)+" "+level.get(i));
      //testview.append("\n");
   }
   testview.append(Long.toString(sleeptime));
   testview.append("\nAPS:  " +aps+"\n"+"NewAPS:
"+newaps+"\nChanges:  "+changes+"\nThreshold:  "+ threshold);
   testview.append("\nAP with the best signal level in the area is
"+bssid.get(0)+"-"+ssid.get(0));

   //testview.append("\n"+dist1+"\n"+dist2);
   //testview.append("\n"+bla+"  circles");
   //testview.append("\nold  lon "+magicx);
```

```
                    //testview.append("\nold  lat "+magicy);

                    //testview.append("\nnew  lon "+newmagicx);

                    //testview.append("\nnew  lat "+newmagicy);

                    //testview.append("\ndist  from A "+distfromA);

                    //testview.append("\ndist  from B "+distfromB);

                   /* if(closestAP==null)

                    {

                       testview.append("\nYou  are now in the area of: "+ssid.get(0)+"
(based on signal levels)");

                       testview.append("\nDistance  measured from the AP of the shop is
"+distancefromAP+" meters");

                    }

                    else

                    {

                       testview.append("\n(triangulation)You  are now in the area of:
"+closestAP+" (based on triangulation  technique)");

                       testview.append("\nDistance  measured from the shop is
"+distancefromshop+ " meters");

                    }*/

                    //testview.append("\n"+x1+","+y1);

                    //testview.append("\n"+x2+","+y2);

                    //testview.append("\n"+x3+","+y3);

                    //testview.append("\n"+x4+","+y4);

                    //if(magicx2!=0.0 && magicy2!=0.0)

                    testview.append("\n");

                    for(int i=0; i<distances.size(); i++)

                    {

                      //testview.append("dist:  "+distances.get(i));

                      //testview.append("\n");


                    }
```

```
                    testview.append("\nYour  coordinates
are:\nx:"+magicx2+"\ny:"+magicy2);
                    if(closestshop!=null)
                    testview.append("\nYou  are standing before the shop: "+closestshop);
                    testview.append("\n"+count);
                }

                String message = b.getString("message");
                if(message!=null)
                {
                    Toast.makeText(getApplicationContext(),  message, 7).show();
                    m.getT().interrupt();
                    if(m.getT()!=null)
                    {
                        try
                        {
                            m.getT().join();
                        }
                        catch (InterruptedException  ex)
                        {

Logger.getLogger(ScanAPs.class.getName()).log(Level.SEVERE,  null, ex);
                        }
                    }
                    buttonPressed = false;
                    m.setT(null);
                    m.setWt(null);
                }
            }
        };
```

```
@Override
protected void onCreate(Bundle savedInstanceState) {
        // TODO Auto-generated method stub
        super.onCreate(savedInstanceState);
        setContentView(R.layout.scan_aps);
    testview = (TextView)findViewById(R.id.testview);
    m = Memory.getSingleInstance(null);
}



@Override
protected void onDestroy() {
        super.onDestroy();
        final StackTraceElement[] ste = Thread.currentThread().getStackTrace();
        Log.i(TAG, "Called " + ste[2].getMethodName()+  " of
"+this.getClass().getName());
}

@Override
protected void onPause() {
        // TODO Auto-generated method stub
        super.onPause();
        final StackTraceElement[] ste = Thread.currentThread().getStackTrace();
        Log.i(TAG, "Called " + ste[2].getMethodName()+  " of
"+this.getClass().getName());
}

@Override
```

```
	protected void onResume() {
		// TODO Auto-generated method stub
		super.onResume();
		final StackTraceElement[] ste = Thread.currentThread().getStackTrace();
		Log.i(TAG, "Called " + ste[2].getMethodName()+  " of
"+this.getClass().getName());
	}

	@Override
	protected void onStart()
      {
		// TODO Auto-generated method stub
		super.onStart();
	    m = Memory.getSingleInstance(null);
	    t = m.getT();
	    wt = m.getWt();
	    if (t!=null)
	    {
	       wt.setHandler(handler);
	       buttonPressed=true;
	    }

		final StackTraceElement[] ste = Thread.currentThread().getStackTrace();
		Log.i(TAG, "Called " + ste[2].getMethodName()+  " of
"+this.getClass().getName());
	}

	@Override
	protected void onStop() {
		// TODO Auto-generated method stub
```

```
            super.onStop();
        if (t!=null)
        {
            wt.setHandler(null);
        }

            final StackTraceElement[] ste = Thread.currentThread().getStackTrace();
            Log.i(TAG, "Called " + ste[2].getMethodName()+  " of
"+this.getClass().getName());
    }

    public void scanAPs(View v)
    {

        switch(v.getId())
        {
            case(R.id.startScanning):
                if(buttonPressed==true)
                {
                    Toast.makeText(getApplicationContext(),  "Already scanning...",
5).show();
                    break;
                }
                buttonPressed=true;

                wt = new APScanner(handler,this);
                t = new Thread(wt);
                t.start();
```

```
                m.setT(t);
                m.setWt(wt);
                break;
              case(R.id.stopScanning):

                if(buttonPressed==false)
                {
                    Toast.makeText(getApplicationContext(), "You have not started
scanning...", 5).show();
                    break;
                }
                buttonPressed=false;
            t.interrupt();
                m.setT(null);
                m.setWt(null);
                testview.getEditableText().clear();

                  if(t!=null)
                {
                   try
                   {
                       t.join();
                   } catch (InterruptedException ex)
                   {
                       Logger.getLogger(ScanAPs.class.getName()).log(Level.SEVERE,
null, ex);
                   }
                }
          }
      }
```

```
}
```

Package android.ip.memory

- **APMemory.java**

```
package android.ip.memory;


public class APMemory
{
    private String SSID;
    private String BSSID;
    private int level;
    private double distance;
    private double tx_power;
    private double x;
    private double y;
    private double s1x1;
    private double s1x2;
    private double s1y1;
    private double s1y2;

    public APMemory(String BSSID, double tx_power, double x, double y, double s1x1, double s1y1,double s1x2, double s1y2) {
        this.BSSID = BSSID;
        this.tx_power = tx_power;
        this.x = x;
```

```
        this.y = y;
        this.s1x2 = s1x2;
        this.s1y1 = s1y1;
        this.s1y2 = s1y2;
        this.s1x1 = s1x1;
    }

    public APMemory(String SSID, String BSSID, int level, double x, double y, double
distance) {
        this.SSID = SSID;
        this.BSSID = BSSID;
        this.level = level;
        this.distance = distance;
        this.x = x;
        this.y = y;
    }




    public double getDistance() {
        return distance;
    }

    public double getS1x1() {
        return s1x1;
    }

    public double getS1x2() {
        return s1x2;
    }
```

```
public double getS1y1() {
    return s1y1;
}

public double getS1y2() {
    return s1y2;
}

public void setDistance(double distance) {
    this.distance = distance;
}

public void setS1x1(double s1x1) {
    this.s1x1 = s1x1;
}

public void setS1x2(double s1x2) {
    this.s1x2 = s1x2;
}

public void setS1y1(double s1y1) {
    this.s1y1 = s1y1;
}

public void setS1y2(double s1y2) {
    this.s1y2 = s1y2;
}
```

```
public APMemory(String SSID, String BSSID, int level) {
    this.SSID = SSID;
    this.BSSID = BSSID;
    this.level = level;
}

public void setTx_power(int tx_power) {
    this.tx_power = tx_power;
}

public void setX(double x) {
    this.x = x;
}

public void setY(double y) {
    this.y = y;
}

public double getTx_power() {
    return tx_power;
}

public double getX() {
    return x;
}
```

```
public double getY() {
    return y;
}

public String getSSID() {
    return SSID;
}

public String getBSSID() {
    return BSSID;
}

public int getLevel() {
    return level;
}

public void setSSID(String SSID) {
    this.SSID = SSID;
}

public void setBSSID(String BSSID) {
    this.BSSID = BSSID;
}

public void setLevel(int level) {
    this.level = level;
}
```

```
}
```

- **Memory.java**

```
package android.ip.memory;


import android.content.Context;
import android.ip.tasks.APScanner;


public class Memory {

    private static Memory singleInstance;
    private APScanner wt;
    private Thread t;

    private Memory() {}

    public static Memory getSingleInstance(Context  context) {
        if (singleInstance  == null) {
            singleInstance  = new Memory();
        }
        return singleInstance;
    }
```

```
    public APScanner getWt() {
        return wt;
    }

    public Thread getT() {
        return t;
    }

    public void setWt(APScanner wt) {
        this.wt = wt;
    }

    public void setT(Thread t) {
        this.t = t;
    }


}
```

Package android.ip.tasks

- **APScanner.java**

```
package android.ip.tasks;


import android.content.Context;
import android.ip.memory.APMemory;
import java.util.ArrayList;
import android.net.wifi.ScanResult;
import android.net.wifi.WifiManager;
```

```
import android.os.Bundle;
import android.os.Handler;
import android.os.Message;
import android.util.Log;
import java.util.Collections;
import java.util.Comparator;
import java.util.List;


public class APScanner implements Runnable{
    private Message msg;
    private Bundle b;
    private Handler handler;
    private Context context;

    private static long maxsleeptime=10000;
    private static long minsleeptime=2000;
    private static long initsleeptime=5000;

    private static int levelthreshold  = 5;
    private static double constant = 4.0;
    private static int maxcount=3;

    //private static String BSSID1="00:1d:7e:bc:6f:73";
    //private static String BSSID2="00:12:17:7a:bf:57";
    //private static String BSSID3="00:12:17:70:d7:08";
    //double tx2= -21;
    //double tx1= -23;
    //double tx3= -24;
    private static String BSSID1="00:24:17:22:cb:97";
    private static String BSSID2="62:07:26:56:79:e3";
    private static String BSSID3="38:22:9d:c1:04:65";
    double tx2= -16;
    double tx1= -16;
    double tx3= -16;
```

```
//private static double tx=-14.47158;
private static double constant2=32.4;


private long sleeptime = initsleeptime;

public APScanner (Handler h, Context c){
    this.handler = h;
    this.context = c;
}


            @Override
            public void run()
            {
                int f1=0,f2=0,f3=0;
                double x = 0,y = 0,newx1 = 0,newy1 = 0,newx2=0,newy2=0,newx=0,newy=0;
                String ssid1 = null,ssid2 = null;
                List <APMemory> aps = new ArrayList();
                double changes;
                int count=0;

                List <APMemory> knownaps = new ArrayList();
                knownaps.add(new APMemory(BSSID3, tx3, 0.1,0,0,0,3.1,0));
                knownaps.add(new APMemory(BSSID2, tx2, 9.6,0,6.6,0,9.7,0));
                knownaps.add(new APMemory(BSSID1, tx1, 9.7/2,8.8,3.1,0,6.6,0));
                //knownaps.add(new APMemory(BSSID4, tx, -5,-15,-6,-18,-4,-18));
                //knownaps.add(new APMemory(BSSID5, tx, 5,-15,4,-18,6,-18));
                //knownaps.add(new APMemory(BSSID6, tx, 10,-20,8,-22,12,-22));
```

```
while(!Thread.currentThread().isInterrupted())
{
   List <APMemory> newaps = new ArrayList();
   changes=0;

   String connectivity_context  = context.WIFI_SERVICE;

   final WifiManager wifi =
(WifiManager)context.getSystemService(connectivity_context);
   boolean wifiEnabled = wifi.isWifiEnabled();
   if (!wifiEnabled)
   {
      //wifi.setWifiEnabled(true);
      try
      {
         msg = new Message();
         b = new Bundle();
         b.putString("message", "Please enable Wi-Fi...");
         msg.setData(b);
         if (handler !=null)
         {
            handler.sendMessage(msg);
         }
         Thread.sleep(2000);
      }
      catch (InterruptedException  ex)
      {
         Thread.currentThread().interrupt();
      }
   }
   else
   {
   wifi.startScan();
   List <ScanResult>  s;
```

```
s=wifi.getScanResults();
Collections.sort(s, new Comparator<ScanResult>() {
@Override
public int compare(ScanResult lhs, ScanResult rhs) {
   return (lhs.level >rhs.level ? -1 : (lhs.level==rhs.level ? 0 : 1));
}
});

msg = new Message();
b = new Bundle();

APMemory ap;
List <String> ssid = new ArrayList();
List <String> bssid = new ArrayList();
List <Integer> level = new ArrayList();
double r1 = 0;
double r2 = 0;
double r3 = 0;
List <APMemory> toCompare = new ArrayList();
List <String> distances = new ArrayList();


if (!aps.isEmpty())
{
   for(int i=0; i<s.size(); i++)
   {
      if(s.get(i).level>-100)
      {


         ssid.add(s.get(i).SSID);
         level.add(s.get(i).level);
         bssid.add(s.get(i).BSSID);
         ap=new APMemory(s.get(i).SSID,s.get(i).BSSID,s.get(i).level);
```

```
                    newaps.add(ap);
                    APMemory tmp=containsBSSID(aps,s.get(i).BSSID);
                    if(tmp==null)
                    {
                        changes+=0.5;
                    }
                    else
                    {
                        if((tmp.getLevel()-s.get(i).level>=levelthreshold)          ||
(tmp.getLevel()-s.get(i).level<=-levelthreshold))
                        {
                            changes++;
                        }
                    }
                }
            }
            for(int i=0; i<aps.size(); i++)
            {
                APMemory tmp=containsBSSID(newaps,aps.get(i).getBSSID());
                if(tmp==null)
                {
                    changes+=0.5;
                }
            }

        }
        else
        {
            for(int i=0; i<s.size(); i++)
            {
                if(s.get(i).level>-100)
                {

                    ssid.add(s.get(i).SSID);
                    level.add(s.get(i).level);
                    bssid.add(s.get(i).BSSID);
```

```
                ap=new APMemory(s.get(i).SSID,s.get(i).BSSID,s.get(i).level);
                aps.add(ap);


            }
        }
    }
```

```
 x = newx;
 y = newy;
```

```
int i=aps.size();
if (!newaps.isEmpty())
{
    int j=newaps.size();

        //if(!aps.get(0).getBSSID().equals(newaps.get(0).getBSSID()))
        //    sleeptime-=1000;
    if(changes>=(int)(Math.ceil((i+j)/constant)))
    {
        if (sleeptime > initsleeptime)
            sleeptime = initsleeptime;
        else
            sleeptime-=1000;
        count=0;
    }
    else
    {
        sleeptime+=1000;
        count++;
```

```
                }

                if (sleeptime>maxsleeptime)
                    sleeptime = maxsleeptime;
                if (sleeptime<minsleeptime)
                    sleeptime = minsleeptime;
            }

            if(count==maxcount)
            {
            for(i=0; i<s.size(); i++)
            {
                for(int j=0; j<knownaps.size(); j++)
                {
                    if (s.get(i).BSSID.equals(knownaps.get(j).getBSSID()))
                    {
                        int f=s.get(i).frequency;
                        r1=Math.pow(10,((knownaps.get(j).getTx_power()-
s.get(i).level-constant2-20*Math.log10(f))/20))*1000;
                        distances.add(knownaps.get(j).getSSID()+"  "+r1+" ");
                        APMemory apToCompare = new
APMemory(s.get(i).SSID,s.get(i).BSSID, s.get(i).level,knownaps.get(j).getX(),
knownaps.get(j).getY(), r1);
                        toCompare.add(apToCompare);
                        Collections.sort(toCompare, new Comparator<APMemory>() {
                        public int compare(APMemory lhs, APMemory rhs) {
                            return (lhs.getLevel() >rhs.getLevel() ? -1 :
(lhs.getLevel()==rhs.getLevel() ? 0 : 1));
            }
            });



                    }
                }
            }
```

```
                double []init_point = compute_init_point(toCompare.get(0), toCompare.get(1),b);

                for (i=2; i<toCompare.size(); i++)
                {
                    double[] optimized_point = optimize_point(init_point, toCompare.get(i));
                    init_point = optimized_point;
                }
                ////////////////////////////////////////////
                double mindist= -1;
                String closestShop = null;

                for (i=0; i<toCompare.size(); i++)
                {
                  for(int j=0; j<knownaps.size();  j++)
                  {
                     if (toCompare.get(i).getBSSID().equals(knownaps.get(j).getBSSID()))
                     {
                         double distance = distancefromshop(knownaps.get(j), init_point);
                         Log.i("distance",distance+"");
                         if (distance > mindist)
                         {
                            mindist = distance;
                            closestShop = toCompare.get(i).getBSSID();
                         }
                     }
                  }
                }
                 if (mindist==-1)
                 {
                    b.putString("closestshop","none");
                 }
                 else
                 {
```

```
                b.putString("closestshop",closestShop);
                b.putDouble("mindistance", mindist);
            }
            ////////////////////////////////////////////////////////
            b.putDouble("magicx2",init_point[0]);
            b.putDouble("magicy2",init_point[1]);

           if(r1<r2)
              b.putDouble("distancefromAP",r1);
           else
              b.putDouble("distancefromAP",r2);



           }
           b.putInt("count",count);
           b.putStringArrayList("ssid", (ArrayList<String>) ssid);
           b.putStringArrayList("distances", (ArrayList<String>) distances);
           b.putStringArrayList("bssid", (ArrayList<String>) bssid);
           b.putIntegerArrayList("level", (ArrayList<Integer>) level);
           b.putLong("sleeptime", sleeptime);
           b.putInt("aps", aps.size());
           b.putInt("newaps", newaps.size());
           b.putDouble("changes", changes);
          b.putInt("threshold",                                    (int)
Math.ceil(((aps.size()+newaps.size())/constant)));



           msg.setData(b);

           if (handler !=null)
           {
              handler.sendMessage(msg);
           }
           if (!newaps.isEmpty())
           {
```

```
                    int j;
                    aps=new ArrayList();
                    for (j=0; j<newaps.size();  j++)
                    {
                        aps.add(newaps.get(j));
                    }
                }
                try
                {
                    Thread.sleep(sleeptime);

                } catch (InterruptedException  ex)
                {
                     Thread.currentThread().interrupt();
                }
              }
            }
        }

    public void setHandler(Handler  handler)
    {
        this.handler  = handler;
    }
    public static APMemory containsBSSID(List<APMemory>  list, String bssid) {
    for (APMemory object : list) {
        if (object.getBSSID().equals(bssid))  {
            return  object;
        }
    }
    return  null;
}

 static  double[] compute_init_point(APMemory  ap1, APMemory ap2, Bundle  b)
    {
        double χ1,χ2,χ3,χ4,ψ1,ψ2,ψ3,ψ4,magicx  = 0,magicy = 0;
```

```
double x1 = ap1.getX();
double x2 = ap2.getX();
double y1 = ap1.getY();
double y2 = ap2.getY();
double r1 = ap1.getDistance();
double r2 = ap2.getDistance();

double [] first_circle =solve_system(x1,x2,y1,y2,r1);

double[] second_circle = solve_system(x2,x1,y2,y1,r2);

χ1=first_circle[0];
χ2=first_circle[1];
ψ1=first_circle[2];
ψ2=first_circle[3];
χ3=second_circle[0];
χ4=second_circle[1];
ψ3=second_circle[2];
ψ4=second_circle[3];



double dist1=Math.sqrt(Math.pow(χ3-χ1,2)+Math.pow(ψ3-ψ1,2));
double dist2=Math.sqrt(Math.pow(χ4-χ1,2)+Math.pow(ψ4-ψ1,2));
double dist3=Math.sqrt(Math.pow(χ3-χ2,2)+Math.pow(ψ3-ψ2,2));
double dist4=Math.sqrt(Math.pow(χ4-χ2,2)+Math.pow(ψ4-ψ2,2));

//Log.i("TTTT",  dist1+" "+dist2+" "+dist3+" "+dist4);
double dist=Math.min(Math.min(dist1,dist2),Math.min(dist3,  dist4));
if(dist==dist1)
{
   magicx=(χ1+χ3)/2;
   magicy=(ψ1+ψ3)/2;
}
else if(dist==dist2)
{
```

```
    magicx=(χ1+χ4)/2;
    magicy=(ψ1+ψ4)/2;
}
else if(dist==dist3)
{
    magicx=(χ2+χ3)/2;
    magicy=(ψ2+ψ3)/2;
}
else if(dist==dist4)
{
    magicx=(χ2+χ4)/2;
    magicy=(ψ2+ψ4)/2;
}

double[] initpoint= new double[2];
initpoint[0]=magicx;
initpoint[1]=magicy;
/*Log.i("FSD1", ap1.getBSSID());
Log.i("FSD1",ap1.getDistance()+"");
Log.i("FSD1", ap1.getLevel()+"");
Log.i("FSD123", ap1.getX()+"");
Log.i("FSD122", ap1.getY()+"");
Log.i("FSD2", ap2.getBSSID());
Log.i("FSD2", ap2.getDistance()+"");
Log.i("FSD2", ap2.getLevel()+"");
Log.i("FSD223", ap2.getX()+"");
Log.i("FSD222", ap2.getY()+"");*/
b.putDouble("x1", χ1);
b.putDouble("x2", χ2);
b.putDouble("y1", ψ1);
b.putDouble("y2", ψ2);
b.putDouble("x3", χ3);
b.putDouble("x4", χ4);
b.putDouble("y3", ψ3);
b.putDouble("y4", ψ4);
```

```
    return initpoint;
}

static double[] optimize_point(double[] point, APMemory newap){

    double χ1,χ2,ψ1,ψ2,magicx = 0,magicy = 0;

    double[] optimizedpoint= new double[2];
    double x2=point[0];
    double y2=point[1];
    double x1= newap.getX();
    double y1= newap.getY();
    double r1= newap.getDistance();



    //System.out.println("x1:  "+x1+" y1: "+y1+" x2: "+x2+" y2: "+y2);
    double[] circle = solve_system(x1,x2,y1,y2,r1);

    χ1=circle[0];
    χ2=circle[1];
    ψ1=circle[2];
    ψ2=circle[3];
    //System.out.println("χ1:  "+χ1+" ψ1: "+ψ1+" χ2: "+χ2+" ψ2: "+ψ2);

    double dist1=Math.sqrt(Math.pow(x2-χ1,2)+Math.pow(y2-ψ1,2));
    double dist2=Math.sqrt(Math.pow(x2-χ2,2)+Math.pow(y2-ψ2,2));
    //System.out.println("dist1  " + dist1);
    //System.out.println("dist2  "+ dist2);
    double dist=Math.min(dist1,dist2);
    if(dist==dist1)
    {
        magicx=(χ1+x2)/2;
        magicy=(ψ1+y2)/2;
    }
```

```
        else if(dist==dist2)
        {
            magicx=(χ2+x2)/2;
            magicy=(ψ2+y2)/2;
         }
        optimizedpoint[0]= magicx;
        optimizedpoint[1]= magicy;
       return optimizedpoint;


    }


   static  public double[] solve_system(double x1, double x2, double y1, double y2, double r1){
              double χ1 = 0,χ2 = 0,χ3 = 0,χ4 = 0,ψ1 = 0,ψ2 = 0,ψ3 = 0,ψ4 = 0,magicx = 0,magicy = 0;
              double alpha, beta;
                      double α1, β1, γ1;
        if(x2-x1==0)
        {
            alpha = 0;
            beta = x1;
            α1=1;
            β1=-(2*y1);
            γ1=Math.pow(y1,  2)-Math.pow(r1,2);
            double disc = Math.pow(β1,2)  - (4*α1*γ1);
            χ1 =beta;
            χ2 =beta;
            ψ1 = ( -β1 + Math.sqrt(disc))  / (2.0 * α1);
            ψ2 =  ( -β1 - Math.sqrt(disc)) / (2.0 * α1);



        }
        else if (y2-y1  ==0)
        {
            alpha = y1;
            beta = 0;
```

```
        α1=1;
        β1=-(2*x1);
        γ1=Math.pow(x1, 2)-Math.pow(r1,2);
        double disc = Math.pow(β1,2) - (4*α1*γ1);
        χ1 = ( -β1 + Math.sqrt(disc)) / (2.0 * α1);
        χ2 = ( -β1 - Math.sqrt(disc)) / (2.0 * α1);
        ψ1 = alpha;
        ψ2 =alpha;
    }
    else
    {
        alpha=(y2-y1)/(x2-x1);
        beta=-(alpha*x1)+y1;
        α1=1+Math.pow(alpha, 2);
        β1=-(2*x1)+(2*(-y1+beta)*alpha);
        γ1=Math.pow(x1, 2)+Math.pow(-y1+beta, 2)-Math.pow(r1,2);
        double disc = Math.pow(β1,2) - (4*α1*γ1);
        //Log.i("DISC",disc+" ");

        χ1 = ( -β1 + Math.sqrt(disc)) / (2.0 * α1);
        χ2 = ( -β1 - Math.sqrt(disc)) / (2.0 * α1);
        ψ1 = alpha*χ1+beta;
        ψ2 = alpha*χ2+beta;

    }
    double[] coords= new double[4];
    coords[0] = χ1;
    coords[1] = χ2;
    coords[2] = ψ1;
    coords[3] = ψ2;

    return coords;
}

public double distancefromshop(APMemory ap1, double[] optimizedpoint)
```

```
{
    double x1 = ap1.getS1x1();
    double x2 = ap1.getS1x2();
    double y1 = ap1.getS1y1();
    double y2 = ap1.getS1y2();
    double pointx = optimizedpoint[0];
    double pointy = optimizedpoint[1];
    double px = x2-x1;
    double py = y2-y1;

    float norm = (float) (px*px + py*py);

    double u =  ((pointx - x1) * px + (pointy - y1) * py) / norm;
    if(u<0 || u>1)
        return -1;
    else
    {
        double x = x1 + u * px;
        double y = y1 + u * py;
        px=pointx-x;
        py=pointy-y;
        double  dist = Math.sqrt(px*px + py*py);
        return dist;
    }

 }
}
```

ΑΝΑΦΟΡΕΣ

[1] Pateli A., Giaglis G.M, Fouskas K., Kourouthanassis P., Tsamakos A., “On the Potential Use of Mobile Positioning Technologies in Indoor Environments”, In the Proceedings of 15th Bled Electronic Commerce Conference -e-Reality: Constructing the e-Economy, Bled, Slovenia June 17 - 19, 2002, pp. 421-427.

[2] Rainer Mautz, “Indoor Positioning Technologies”, Habilitation Thesis, Institute of Geodesy and Photogrammetry, Department of Civil, Environmental and Geomatic Engineering, ETH Zurich, February 2012, pp. 11-14.

[3] Jirapat Sangthong, Prakaidao Dokpikul, and Sathaporn Promwong, “Indoor Positioning Based on IEEE 802.15.4a Standard Using Trilateration Technique and UWB Signal”, Progress In Electromagnetics Research Symposium Proceedings, KL, MALAYSIA, March 27-30, 2012,pp. 473-474

[4] S.Thummalapalli, “Wi-Fi Indoor Positioning”, Master Report, IDE 1254, September 2012, pp. 5-7, Figure 3.

[5] Landu Jiang, “A Wlan Fingerprinting Based Indoor Localization Technique”, Lincoln, Nebraska, July, 2012.

[6] Kamol Kaemarungsi, Prashant Krishnamurthy, “Modeling of Indoor Positioning Systems Based on Location Fingerprinting”, IEEE, 2004.

[7] Dimitris Milioris, George Tzagkarakis, Artemis Papakonstantinoua, Maria Papadopouli, Panagiotis Tsakalides, “Low-dimensional signal-strength fingerprint-based positioning in wireless LANs”, December 2011.

[8] Chenshu Wu, Zheng Yang, Yunhao Liu, Wei Xi, “WILL: Wireless Indoor Localization Without Site Survey”, Transactions on parallel and distributed systems, Vol X., No. X. , February 2012.

[9] E. Trevisani, A. Vitaletti, “Cell-id location technique, limits and benefits: an experimental study,” in Mobile Computing Systems and Applications, 2004. WMCSA 2004, Sixth IEEE Workshop, 2004, pp. 51 – 60.

[10] R. Want, A. Hopper, V. Falcao, J. Gibbons, "The Active Badge Location System", ACM Trans. Information Systems, vol. 10, no. 1, January 1992, pp. 91-102.

[11] http://www.xbox.com/en-US/kinect, τελευταία περιήγηση 1/10/2013

[12] http://www.primesense.com/casestudies/kinect/, τελευταία περιήγηση 1/10/2013

[13]http://www.cl.cam.ac.uk/research/dtg/attarchive/bat/, τελευταία περιήγηση 1/10/2013

[14] Priyantha, N.B. "The Cricket Indoor Location System", PhD Thesis, Massachusetts Institute of Technology.

[15] http://cricket.csail.mit.edu/, τελευταία περιήγηση 1/10/2013

[16] Sunkyu Woo, Seongsu Jeong, Esmond Mok, Linyuan Xia, Changsu Choi, Muwook Pyeon, Joon Heo.
"Application of Wi-Fi-based indoor positioning system for labor tracking at construction sites: A case study in Guangzhou MTR", May 2010

[17] Rainer Mautz, "Indoor Positioning Technologies", Habilitation Thesis, Institute of Geodesy and Photogrammetry, Department of Civil, Environmental and Geomatic Engineering, ETH Zurich, February 2012, p. 64.

[18] Bahl, P. and Padmanabhan, "Radar: An In-Building RF-based User Location and Tracking System", Proceedings of INFOCOM 2000, IEEE Conference on Computer Communications, 2000, Tel Aviv, Israel.

[19] http://www.skyhookwireless.com/, τελευταία περιήγηση 1/10/2013

[20] http://www.zonith.com/, τελευταία περιήγηση 1/10/2013

[21] ZONITH Indoor Positioning Module white paper version 2.1a.

[22] Kim, J. and Jun, H.S., "Vision-Based Location Positioning using Augmented Reality forIndoor Navigation", IEEE Transaction on Consumer Electronics, vol. 54, no. 3, October 2008, pp. 954–962,

[23] Robert Watson, Gerard Lachapelle, Richard Klukas, Seppo Turunen, Samuli Pietila, Ismo Halivaara, "Investigating GPS Signals Indoors with Extreme High-Sensitivity Detection Techniques", Nokia Corporation, Tampere, Finland, February 2006.

[24] Anja Bekkelien, Michel Deriaz, "Hybrid Positioning Framework for Mobile Devices", Institute of Services Science, University of Geneva, Switzerland, 2012.

[25] Eladio Martin, Oriol Vinyals, Gerald Friedland, Ruzena Bajcsy, Precise Indoor Localization Using Smart Phones, International conference on Multimedia, 2010, pp. 787-790.

[26] Satya Komatineni, Dave MacLean, and Sayed Y. Hashimi, "Pro Android 3", ISBN-13 (pbk): 978-1-4302-3222-3, 2011.

[27] Dave Smith, Jeff Friesen, "Android Recipes: A Problem-Solution Approach", ISBN-13: 978-1430246145, December 5, 2012.

[28] Mark Murphy, "Beginning Android 2", ISBN-13: 978-1430226291, March 19, 2010.

[29] David B. Green, M. S. Obaidat, "An Accurate Line of Sight Propagation Performance Model for Ad-Hoc 802.11 Wireless LAN (WLAN) Devices", SRI International and Monmouth University, 2002.

[30] Masahiro Takashima, Dapeng Zhao, Kentaro Yanagihara, Kiyoshi Fukui, Shigeru

Fuku-naga, Shinsuke Hara, and Ken-Ichi Kitayama. "Location estimation using received signal power and maximum likelihood estimation in wireless sensor networks". Electronics and Communications in Japan, Part I: Communications,v.90, n.12, December, 2007, pp 62-72.

[31] K. Pahlavan, Xinrong Li, and J. P. Makela. "Indoor geolocation science and technology. Communications Magazine", IEEE, pp112-118, 2002.

[32] Yiming Ji, Saad Biaz, Santosh Pandey, Prathima Agrawal. "Ariadne: a dynamic indoor signal map construction and localization system". In MobiSys '06: Proceedings of the 4th international conference on Mobile systems, applications and services, New York, NY, USA, 2006, pp. 151-164.