



ΕΘΝΙΚΟ ΚΑΙ ΚΑΠΟΔΙΣΤΡΙΑΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ

**ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ**

ΠΡΟΓΡΑΜΜΑ ΜΕΤΑΠΤΥΧΙΑΚΩΝ ΣΠΟΥΔΩΝ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

**Αποδοτική ανίχνευση πτώσης σε Android με χρήση τεχνικών
κατωφλίου και εντοπισμό ζώνης εσωτερικού χώρου**

**Άγγελος Γ. Βαλσαμής
Ελευθέριος Π. Σκανδαλέλλης**

***Επιβλέπουσα:* Ιζαμπώ Καράλη, Επίκουρος Καθηγήτης**

ΑΘΗΝΑ

ΙΟΥΛΙΟΣ 2017

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Αποδοτική ανίχνευση πτώσης σε Android με χρήση τεχνικών κατωφλίου και εντοπισμό ζώνης εσωτερικού χώρου

Άγγελος Γ. Βαλσαμής

A.M.: M1432

Ελευθέριος Π. Σκανδαλέλλης

A.M.: M1312

ΕΠΙΒΛΕΠΟΥΣΑ **Ιζαμπώ Καράλη**, Επίκουρος Καθηγήτης

ΕΞΕΤΑΣΤΙΚΗ ΕΠΙΤΡΟΠΗ: **Ιζαμπώ Καράλη**, Επίκουρος Καθηγήτης
Χατζηευθυμιάδης Ευστάθιος, Αναπληρωτής Καθηγήτης

Ιούλιος 2017

ΠΕΡΙΛΗΨΗ

Στην παρούσα εργασία έχουμε αντιμετωπίσει το πρόβλημα της ανίχνευσης πτώσης μέσω τεχνολογίας Android σε κινητές συσκευές. Το πρόβλημα αφορά κυρίως τους ηλικιωμένους, αφού με βάση τα στατιστικά, ένα συντριπτικό ποσοστό επισκέψεων σε νοσοκομεία για άτομα ηλικίας 65 και άνω, είναι λόγω κάποιας πτώσης. Για το συγκεκριμένο πρόβλημα έχουν χρησιμοποιηθεί στο παρελθόν διάφορες τεχνολογίες, όπως κάμερες, αισθητήρες χώρου, τεχνολογίες υπέρυθρων κτλ. Η έλευση όμως των “έξυπνων κινητών τηλεφώνων” (smartphones) και των φορετών συσκευών (wearables, smartwatches), καθώς και οι αισθητήρες που διαθέτουν αυτές οι συσκευές, έχουν φέρει μια επανάσταση στον χώρο της ανίχνευσης πτώσης, αφού είναι πιο φθηνές λύσεις και με χρήση κατάλληλων τεχνικών, όπως οι τεχνικές κατωφλίου και οι τεχνικές μηχανικής μάθησης, δίνουν το ίδιο αξιόπιστα αποτελέσματα και βοηθάνε με πολλούς επιπλέον τρόπους (κλήση έκτακτης ανάγκης, ειδοποίηση υπευθύνου κτλ.).

Στα πλαίσια της διπλωματικής μας έχουμε αναπτύξει μια εφαρμογή σε Android, η οποία χρησιμοποιεί τους ενσωματωμένους αισθητήρες που υπάρχουν σε αυτές τις συσκευές, και πιο συγκεκριμένα το επιταχυνσιόμετρο (accelerometer), για να συλλέξει δεδομένα. Με την προϋπόθεση ότι ο ηλικιωμένος έχει τη συσκευή πάνω του κατά τη διάρκεια της καθημερινότητας στο χώρο του, μπορούμε με βάση αυτά τα δεδομένα να ανιχνεύσουμε πτώσεις, μέσω ενός αλγορίθμου που χρησιμοποιεί τεχνικές κατωφλίου (threshold-based techniques) και εκτελείται σε απομακρυσμένο εξυπηρετητή (server). Στη συνέχεια, όταν εντοπιστεί κάποια πτώση, στέλνεται ειδοποίηση (notification) σε κάποιον υπεύθυνο για τον ηλικιωμένο, ώστε να λάβει δράση.

Συμπληρώνοντας τον αλγόριθμο ανίχνευσης πτώσης, έχουμε αναπτύξει και μία τεχνική ανίχνευσης ζώνης σε εσωτερικό χώρο. Πιο συγκεκριμένα, στην αρχική ρύθμιση της εφαρμογής, ο υπεύθυνος έχει τη δυνατότητα να επιλέξει κάποιες ζώνες στο χώρο του ηλικιωμένου που θεωρεί επικίνδυνες και αντίστοιχα κάποιες που θεωρούνται ασφαλείς. Μια ζώνη ορίζεται από τις τιμές της ισχύος σήματος που λαμβάνει η κινητή συσκευή από όλα τα ασύρματα δίκτυα (wi-fi) στο χώρο. Έτσι, αφού καθοριστούν οι ζώνες, ο αλγόριθμος υπολογίζει σε συγκεκριμένα χρονικά διαστήματα τη θέση του ηλικιωμένου, ώστε να γνωρίζουμε αν βρίσκεται κοντά σε κάποια ασφαλή ή επικίνδυνη ζώνη. Αυτή η πληροφορία χρησιμοποιείται για να κάνουμε τον αλγόριθμο ανίχνευσης πτώσης πιο “ευαίσθητο” σε περιοχές υψηλής επικινδυνότητας και λιγότερο “ευαίσθητο” σε περιοχές χαμηλής επικινδυνότητας. Επίσης η πληροφορία για τη ζώνη αποστέλλεται στον υπεύθυνο κατά την ειδοποίηση, ώστε να ενημερωθεί σε ποιο σημείο συνέβη η πτώση.

Τέλος, μέσω του API αναγνώρισης δραστηριότητας (activity recognition) που προσφέρει η Google, έχουμε προσπαθήσει να συμπεριλάβουμε στην ειδοποίηση πληροφορία για τον πιθανό τρόπο που συνέβη η πτώση (κατά τη διάρκεια δραστηριότητας ή από κρεβάτι/καρέκλα) καθώς και για το αν ο ηλικιωμένος έχει ή όχι τις αισθήσεις του μετά την πτώση.

ΘΕΜΑΤΙΚΗ ΠΕΡΙΟΧΗ: Αναγνώριση Δραστηριότητας

ΛΕΞΕΙΣ ΚΛΕΙΔΙΑ: Τεχνική κατωφλίου, Αισθητήρες κινητής συσκευής, Ασύρματα δίκτυα, Σήμα δικτύου, Αλγόριθμος fingerprint

ABSTRACT

In this thesis, we have tackled the problem of fall detection, using Android technology on mobile devices. This problem concerns mostly the elderly, since, based on statistics, a huge percentage of hospitalization for people aged 65 and above, is due to some kind of a fall. A wide variety of technologies have been used to tackle this issue in the past, including cameras, room sensors, infrared technologies etc. The advent of “smart devices”, like smartphones, wearables and smartwatches, along with the embedded sensors that those devices have, has revolutionized the field of fall detection, since they are cheaper solutions and using certain techniques, like thresholding and machine learning techniques, they produce reliable results and can help in many other ways (emergency call, guardian notification etc).

In the context of this thesis, we have developed an Android app that uses the embedded sensors, which are present in all those devices, and more specifically the accelerometer, to collect data. Assuming that the elderly has the device on him/her during his/her day-to-day routine, we can use those data to detect falls, through an algorithm that uses threshold-based techniques and runs on a remote server. Then, when a fall is detected, a notification is sent to someone responsible for the elderly, in order to take action.

In addition to the fall detection algorithm, we have also developed a zone detection technique for interior spaces. More specifically, in the initial configuration of the application, the guardian of the elderly has the option of determining some zones in the elderly living space that he considers dangerous and, accordingly, some that are considered safe. A zone is defined by the signal strength values received by the mobile device from all wireless networks (wi-fi) in the room. Thus, once the zones have been defined, the algorithm calculates at certain intervals the position of the elderly, so that we know if s(he) is near a safe or dangerous zone. This information is used to make the fall detection algorithm more "sensitive" to high-risk areas and less "sensitive" to low-risk areas. Also, the zone information is sent to the guardian of the elderly, in order to inform him/her where the fall has occurred.

Finally, through the Google Activity Recognition API, we have tried to include information in the notification about the possible way that the fall happened (during light activity or from bed/chair), as well as whether the elderly has her senses or not after the fall.

SUBJECT AREA: Activity Recognition

KEYWORDS: Thresholding technique, Mobile device sensors, Wireless networks, Network Signal, Fingerprint algorithm

ΕΥΧΑΡΙΣΤΙΕΣ

Για την εκπόνηση της παρούσας διπλωματικής οφείλουμε θερμές ευχαριστίες στην καθηγήτρια μας κα. Ιζαμπώ Καράλη για την πολύτιμη καθοδήγηση και τα επικοινωνιακά της σχόλια και παρατηρήσεις.

Επίσης θα θέλαμε να ευχαριστήσουμε τον καθηγητή μας κ. Χατζηευθυμιάδη, ο οποίος ήταν μέλος της εξεταστικής επιτροπής, για την ανάγνωση της εργασίας μας και τα πολύτιμα σχόλια του.

Τέλος, δε θα μπορούσαμε να ξεχάσουμε τους γονείς μας, για τη στήριξη που μας έχουν προσφέρει όλα αυτά τα χρόνια και για όλες τις θυσίες που έχουν κάνει ώστε να φτάσουμε σε αυτό το σημείο.

ΠΕΡΙΕΧΟΜΕΝΑ

ΠΡΟΛΟΓΟΣ	12
1. ΕΙΣΑΓΩΓΗ	13
2. ΥΠΑΡΧΟΥΣΕΣ ΜΕΘΟΔΟΛΟΓΙΕΣ ΚΑΙ ΤΕΧΝΙΚΕΣ	15
2.1. Αλγόριθμοι ανίχνευσης πτώσης.....	15
2.1.1. Τεχνικές κατωφλίου	15
2.1.2. Τεχνικές μηχανικής μάθησης	17
2.2. Εφαρμογές ανίχνευσης πτώσης.....	19
2.2.1. Εφαρμογές κινητών	20
2.2.2. Τεχνολογία φορετών συσκευών (wearables)	26
2.3. Αλγόριθμοι και τεχνολογίες ανίχνευσης ζώνης εσωτερικού χώρου.....	29
3. ΠΕΡΙΓΡΑΦΗ ΣΥΣΤΗΜΑΤΟΣ	34
3.1. Περιγραφή Αλγορίθμων	35
3.1.1. Αλγόριθμος ανίχνευση πτώσης	35
3.1.2. Αλγόριθμος ανίχνευσης ζώνης εσωτερικού χώρου	39
3.2. Server	44
3.2.1. Spring Framework	44
3.2.2. RESTful Web Services	45
3.2.3 Swagger API Framework	47
3.2.4. Hibernate ORM Framework	49
3.2.5. Gradle	50
3.2.6. Οργάνωση βάσης	50
3.3. Εφαρμογή Android	53
3.3.1. Android Activities.....	53
3.3.2. Android Services	58
3.3.3. Αναγνώριση δραστηριότητας χρήστη (Google ActivityRecognition API)	59
3.3.4. Ειδοποιήσεις Android (Firebase Cloud Messaging)	60
3.3.5. Επικοινωνία με RESTful Server (Square Retrofit).....	61
4. ΑΞΙΟΛΟΓΗΣΗ ΣΥΣΤΗΜΑΤΟΣ	64

4.1. Περιβάλλον.....	64
4.2. Παράμετροι.....	65
4.3. Μετρήσεις	67
5. ΣΥΜΠΕΡΑΣΜΑΤΑ	73
ΠΙΝΑΚΑΣ ΟΡΟΛΟΓΙΑΣ	74
ΣΥΝΤΜΗΣΕΙΣ – ΑΡΚΤΙΚΟΛΕΞΑ - ΑΚΡΩΝΥΜΑ.....	75
ΑΝΑΦΟΡΕΣ	76

ΚΑΤΑΛΟΓΟΣ ΣΧΗΜΑΤΩΝ

Σχήμα 1: Γενική αρχιτεκτονική συστήματος.....	σελ. 35
Σχήμα 2: Διάγραμμα ροής αλγορίθμου ανίχνευσης πτώσης	σελ. 38
Σχήμα 3: Διάγραμμα ροής online διαδικασίας της ανίχνευσης ζώνης εσωτερικού χώρου	σελ. 43
Σχήμα 4: Διάγραμμα οντοτήτων συσχετίσεων (EER diagram) της βάσης μας.....	σελ. 53

ΚΑΤΑΛΟΓΟΣ ΕΙΚΟΝΩΝ

Εικόνα 1: Τυπική μεταβολή επιτάχυνσης κατά τη διάρκεια πτώσης	σελ. 16
Εικόνα 2: Διαδικασία κατασκευής, αξιολόγησης και χρήσης μοντέλου	σελ. 19
Εικόνα 3: Όρια που χρησιμοποιήθηκαν στο iFall.....	σελ. 21
Εικόνα 4: Ορισμός των αξόνων του PerFallD	σελ. 22
Εικόνα 5: Μέθοδος lateration	σελ. 30
Εικόνα 6: Μέθοδος angulation	σελ. 31
Εικόνα 7: Μέθοδος εγγύτητας.....	σελ. 32
Εικόνα 8: Ο χώρος διεξαγωγής του πειράματος του [18].....	σελ. 33
Εικόνα 9: Ο υπεύθυνος εισάγει μια περιοχή κατά την offline διαδικασία.....	σελ. 41
Εικόνα 10: Τεκμηρίωση API μέσω Swagger	σελ. 48
Εικόνα 11: Αυτόματη μετατροπή κλάσης σε MySQL πίνακα μέσω Hibernate	σελ. 49
Εικόνα 12: AuthenticationActivity - LoginFragment.....	σελ. 54
Εικόνα 13: AuthenticationActivity - SignupFragment	σελ. 55
Εικόνα 14: MainActivity.....	σελ. 56
Εικόνα 15: MainActivity - πλευρική μπάρα	σελ. 57
Εικόνα 16: ZoneActivity - εμφάνιση ζωνών χρήστη	σελ. 57
Εικόνα 17: Αρχιτεκτονική του Firebase Cloud Messaging	σελ. 61
Εικόνα 18: Δείγμα κλάσης διεπαφής SimpleApi	σελ. 62
Εικόνα 19: Κάτοψη του σπιτιού όπου έγιναν τα πειράματα	σελ. 64
Εικόνα 20: Η κάτοψη του σπιτιού με χρώματα σε κάθε σημείο.....	σελ. 69

ΚΑΤΑΛΟΓΟΣ ΠΙΝΑΚΩΝ

Πίνακας 1: Προσαρμογή των παραμέτρων του E-FalD με βάση την ηλικία, το φύλο και το BMI.....	σελ. 24
Πίνακας 2: Είδη πτώσεων ανάλογα με την αντίστοιχη μεταβολή της κατάστασης του χρήστη.....	σελ. 39
Πίνακας 3: Endpoint και τύπος REST υπηρεσιών.	σελ. 45
Πίνακας 4: Παράμετροι εισόδου/εξόδου και περιγραφή REST υπηρεσιών.....	σελ. 46
Πίνακας 5: Απεικόνιση των ζωνών που έχουν οριστεί στο χώρο.....	σελ. 67
Πίνακας 6: Απεικόνιση των μετρήσεων στα 50 σημεία του χώρου.	σελ. 68
Πίνακας 7: Αποτελέσματα από μετρήσεις ADLs.	σελ. 70
Πίνακας 8: Αποτελέσματα από μετρήσεις πτώσεων.....	σελ. 71

ΠΡΟΛΟΓΟΣ

Η παρούσα εργασία διενεργήθηκε στα πλαίσια του Προγράμματος Μεταπτυχιακών Σπουδών του τμήματος Πληροφορικής και Τηλεπικοινωνιών, της σχολής Θετικών επιστημών του Εθνικού και Καποδιστριακού Πανεπιστημίου Αθηνών. Ο τόπος διεξαγωγής της εργασίας ήταν η Αθήνα.

1. ΕΙΣΑΓΩΓΗ

Οι πτώσεις είναι ένα σημαντικό πρόβλημα σε θέματα υγείας στη σημερινή εποχή, αφού είναι η κύρια αιτία τραυματισμού και εισαγωγής σε νοσοκομεία στις τάξεις των ηλικιωμένων. Το 2010, το 71.2% των επειγόντων επισκέψεων σε νοσοκομείο για άτομα ηλικιών 65 και πάνω σχετιζόταν με πτώσεις, ενώ για τον συνολικό πληθυσμό το ποσοστό αυτό είναι 35.7% [1].

Αυτό έχει προκαλέσει μια στροφή ενδιαφέροντος σε εφαρμογές ανίχνευσης πτώσης που μπορούν με αποδοτικό τρόπο να αναγνωρίσουν πότε συμβαίνει μια πτώση και να ενημερώνουν κάποιον υπεύθυνο ώστε να επέμβει. Οι εφαρμογές αυτές, τα τελευταία χρόνια, εκμεταλλεύονται κυρίως τις τεχνολογίες των “έξυπνων συσκευών”, όπως smartphones και smartwatches, και χρησιμοποιώντας τους αισθητήρες που διαθέτουν οι συσκευές αυτές, έχουν αναπτυχθεί ολοκληρωμένες λύσεις για να βοηθήσουν στην έγκαιρη ειδοποίηση του υπεύθυνου. Οι αλγόριθμοι που έχουν χρησιμοποιηθεί στις εφαρμογές αυτές χωρίζονται σε δύο ευρείες κατηγορίες: 1) τις τεχνικές κατωφλίου, οι οποίες χρησιμοποιούν κάποια κατώφλια σε συγκεκριμένες μετρήσεις και εντοπίζουν κάποια πτώση όταν τα προκαθορισμένα αυτά κατώφλια ξεπεραστούν και 2) τις τεχνικές μηχανικής μάθησης, όπου χρησιμοποιούνται γνωστοί αλγόριθμοι μάθησης πάνω σε δεδομένα που έχουν συλλεχθεί για την εκπαίδευση του μοντέλου, και στη συνέχεια εφαρμόζονται αλγόριθμοι αξιολόγησης, για την κατηγοριοποίηση νέων δεδομένων σε πτώσεις ή άλλες δραστηριότητες τις καθημερινής ζωής.

Στην παρούσα διπλωματική παρουσιάζουμε μια ολοκληρωμένη και ρεαλιστική λύση για το πρόβλημα της ανίχνευσης πτώσης. Έχουμε αναπτύξει μια εφαρμογή android, η οποία μπορεί να εκτελεστεί σε κινητές (smartphones) ή φορετές (wearables) συσκευές και λειτουργεί σε συνδυασμό με έναν εξυπηρετητή (server). Η εφαρμογή εκτελείται σε διαφορετική μορφή στις συσκευές του υπεύθυνου και του ηλικιωμένου. Στη συσκευή του ηλικιωμένου συλλέγονται απλά δεδομένα από αισθητήρες της συσκευής, όπως το επιταχυνσιόμετρο, και αποστέλλονται στον server, στον οποίο εκτελείται ο αλγόριθμος ανίχνευσης πτώσης ώστε να εντοπίσει κάποια ενδεχομένως επικίνδυνη κατάσταση για τον ηλικιωμένο. Ο υπεύθυνος από την άλλη μπορεί να κάνει διάφορες ρυθμίσεις και να ελέγχει ουσιαστικά το πότε επιθυμεί να στέλνονται δεδομένα από την εφαρμογή του ηλικιωμένου. Όταν εντοπιστεί κάποια ενδεχόμενη επικίνδυνη κατάσταση, στέλνεται ειδοποίηση (notification) στην εφαρμογή του υπεύθυνου, ώστε να μπορεί να επέμβει.

Αυτό που προσφέρει η εφαρμογή μας επιπλέον σε σχέση με άλλες εργασίες πάνω σε ανίχνευση πτώσης είναι ότι συμμετέχει κι άλλος ένας αλγόριθμος που τρέχει στον server με βάση δεδομένα που έχουν ληφθεί από το κινητό, για την ανίχνευση της ζώνης στην οποία βρίσκεται ο ηλικιωμένος χρήστης. Συγκεκριμένα ο υπεύθυνος μπορεί μέσω της εφαρμογής του να καθορίσει επικίνδυνες και ασφαλείς ζώνες στο χώρο του ηλικιωμένου μέσω δειγματοληψίας, σε κάθε περιοχή, της ισχύος σήματος που δέχεται η android συσκευή από ασύρματα δίκτυα (wi-fi) σε αυτή την περιοχή. Στη συνέχεια, με βάση την εικόνα που έχουμε κάθε στιγμή για την ισχύ σήματος από όλα τα wi-fi στο σημείο που βρίσκεται ο ηλικιωμένος, υπολογίζουμε μέσω μιας τεχνικής αποτύπωσης σήματος (signal fingerprinting) τη ζώνη στην οποία είναι πιο “κοντά”, η οποία θεωρούμε ότι είναι και η ζώνη στην οποία βρίσκεται. Την πληροφορία της επικίνδυνης/ασφαλούς ζώνης τη χρησιμοποιούμε για να κάνουμε περισσότερο/λιγότερο ευαίσθητο τον αλγόριθμο αντίστοιχα, εφόσον σε επικίνδυνες ζώνες θέλουμε να είμαστε σίγουροι ότι δεν θα χάσουμε

κάποια πτώση, ακόμα και αν αυτό σημαίνει κάποιες ψευδώς θετικές (false positive) ενδείξεις.

Επιπλέον, έχουμε χρησιμοποιήσει τη διεπαφή (API) αναγνώρισης δραστηριότητας που μας προσφέρει η Google, ώστε να αποθηκεύουμε στη βάση πληροφορία σχετικά με το αν ο ηλικιωμένος είναι ακίνητος ή πραγματοποιεί κάποια δραστηριότητα. Με βάση αυτή την πληροφορία, βγάζουμε και κάποια επιπλέον συμπεράσματα σε μια πιθανή πτώση, όπως το αν ο ηλικιωμένος έχει ή όχι τις αισθήσεις του και το αν έπεσε ενώ έκανε κάποια δραστηριότητα ή από κάποια στατική θέση (από κρεβάτι/καρέκλα). Αυτά τα πρόσθετα συμπεράσματα καταγράφονται στην ειδοποίηση (notification) που φτάνει στο κινητό του υπεύθυνου χρήστη.

Η συνέχεια της εργασίας είναι οργανωμένη ως εξής:

- Στην ενότητα 2 παρουσιάζουμε τις διάφορες τεχνικές και αλγορίθμους που έχουν χρησιμοποιηθεί για το πρόβλημα της ανίχνευσης πτώσης και παρουσιάζουμε εφαρμογές που έχουν γίνει τόσο σε smartphones όσο και σε wearables. Τέλος περιγράφουμε σχετικές εργασίες για ανίχνευση ζώνης εσωτερικού χώρου.
- Στην ενότητα 3, παρουσιάζουμε τους αλγόριθμους που χρησιμοποιήσαμε στην εργασία τόσο για ανίχνευση πτώσης, όσο και ανίχνευση ζώνης εσωτερικού χώρου. Στη συνέχεια αναλύουμε τη δομή του server και της βάσης μας, καθώς και τις βασικές λειτουργίες της εφαρμογής android που υλοποιήσαμε.
- Στην ενότητα 4 περιγράφουμε την αξιολόγηση που κάναμε για να μετρήσουμε την αποδοτικότητα των δύο αλγορίθμων μας.
- Τέλος, στην ενότητα 5 κάνουμε μια σύνοψη της εργασίας μας και συζητάμε ενδιαφέροντα μελλοντικά βήματα πάνω στο αντικείμενο.

2. ΥΠΑΡΧΟΥΣΕΣ ΜΕΘΟΔΟΛΟΓΙΕΣ ΚΑΙ ΤΕΧΝΙΚΕΣ

Σε αυτή την ενότητα θα μελετήσουμε τους διάφορους αλγορίθμους που χρησιμοποιούνται για την ανίχνευση πτώσεων, καθώς και διάφορα συστήματα και εφαρμογές που έχουν αναπτυχθεί στο χώρο. Επίσης θα δούμε και κάποιες εργασίες που έχουν γίνει πάνω στο δεύτερο κομμάτι της εργασίας μας, που είναι η ανίχνευση ζώνης εσωτερικού χώρου. Συγκεκριμένα στην ενότητα 2.1 θα συζητήσουμε για τις δύο μεγάλες κατηγορίες αλγορίθμων που έχουν αναπτυχθεί για ανίχνευση πτώσεων με κινητές συσκευές, στην ενότητα 2.2 θα δούμε εφαρμογές που έχουν αναπτυχθεί για κινητά, φορητές συσκευές και διάφορες άλλες προσεγγίσεις, και τέλος στην ενότητα 2.3 θα μελετήσουμε διάφορες τεχνικές ανίχνευσης ζώνης εσωτερικού χώρου.

2.1 Αλγόριθμοι ανίχνευσης πτώσης

Οι αλγόριθμοι και οι τεχνικές που έχουν χρησιμοποιηθεί για να αντιμετωπίσουν το πρόβλημα της ανίχνευσης πτώσης με χρήση κινητών συσκευών, χωρίζονται σε δύο μεγάλες κατηγορίες: τις τεχνικές κατωφλίου (threshold-based) και τις τεχνικές μηχανικής μάθησης (machine learning) [2]. Η πιο ευρέως χρησιμοποιούμενη μέθοδος είναι η πρώτη, στην οποία χρησιμοποιούνται κάποια κατώφλια (thresholds) από συγκεκριμένες μετρήσεις που λαμβάνονται από αισθητήρες όπως το 3-αξονικό επιταχυνσιόμετρο, για να ανιχνευθεί μία πτώση. Στις τεχνικές μηχανικής μάθησης, συλλέγονται κάποια δεδομένα από δραστηριότητες της καθημερινής ζωής (activities of daily living - ADL) και στη συνέχεια χρησιμοποιείται ένας αλγόριθμος μάθησης ώστε να γίνει εκπαίδευση πάνω σε αυτά τα δεδομένα. Στη συνέχεια χρησιμοποιείται ένας αλγόριθμος αξιολόγησης για να κατηγοριοποιήσει τα δεδομένα ως φυσιολογικό ADL (περπάτημα, τρέξιμο κλπ) ή ως πτώση. Στις επόμενες ενότητες, θα συζητήσουμε σε περισσότερο βάθος τους μηχανισμούς που χρησιμοποιούνται σε κάθε μία από αυτές τις οικογένειες μεθόδων.

2.1.1 Τεχνικές κατωφλίου

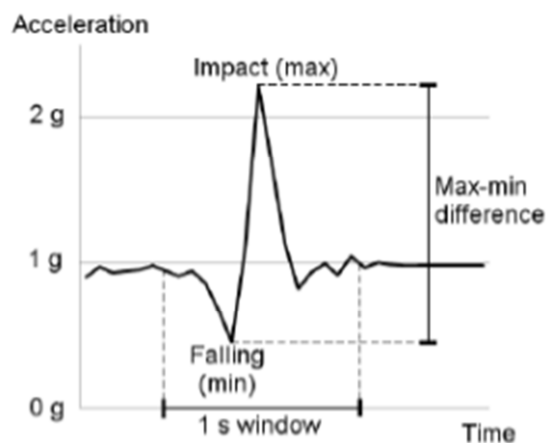
Όπως αναφέραμε, σε αυτή την οικογένεια αλγορίθμων ανίχνευσης πτώσης, χρησιμοποιούνται δεδομένα που συλλέγονται από έναν (ή περισσότερους) από τους αισθητήρες μίας κινητής συσκευής. Στις περισσότερες περιπτώσεις, ο αισθητήρας που χρησιμοποιείται είναι το επιταχυνσιόμετρο, αλλά μπορεί να χρησιμοποιηθούν και άλλοι, όπως το μαγνητόμετρο και το γυροσκόπιο.

Οι τιμές του επιταχυνσιόμετρου έχουν τρεις συνιστώσες, που αναπαριστούν τη μέτρηση της επιτάχυνσης στους άξονες x, y και z αντίστοιχα (A_x , A_y , A_z). Η κύρια μέτρηση που χρησιμοποιείται για την ανίχνευση πτώσης ονομάζεται διάνυσμα επιτάχυνσης ή SMV (Signal Magnitude Vector) και το οποίο ισούται με το εξής:

$$SMV = \sqrt{A_x^2 + A_y^2 + A_z^2}$$

Το πως θα χρησιμοποιηθεί η παραπάνω μέτρηση για να ανιχνευθεί μία πτώση, μπορεί να εξηγηθεί μελετώντας το πως ορίζεται μια πτώση. Σε γενικές γραμμές μια πτώση μπορεί να περιγραφεί ως μια απότομη και ακούσια κίνηση από όρθια θέση σε μία ανακλινόμενη και σχεδόν οριζόντια θέση [3]. Παρ' όλα αυτά μια πτώση δεν μπορεί να εντοπιστεί εύκολα μόνο από αυτό τον ορισμό, καθώς μπορεί εύκολα να υπάρξει σύγχυση με άλλα ADLs. Οπότε αυτό που γίνεται συνήθως στις τεχνικές κατωφλίου είναι το "σπάσιμο" μίας πτώσης σε διακριτές φάσεις, που πρέπει να εντοπιστούν σε μία πτώση, ώστε να την κατηγοριοποιήσουν με μεγαλύτερη ακρίβεια. Διακρίνουμε λοιπόν τις εξής τέσσερις φάσεις:

1. Φάση προ-πτώσης, κατά την οποία πραγματοποιούνται κλασικές ADLs. Εδώ μπορεί να υπάρχουν κάποιες δραστηριότητες, όπως το να κάτσει κάποιος ή να ξαπλώσει γρήγορα, το οποίο μοιάζει με πτώση, αλλά πρέπει να αποφευχθεί η λανθασμένη κατηγοριοποίηση του.
2. Φάση ελεύθερης πτώσης, η οποία είναι μια σύντομη φάση (300-600 ms), κατά την οποία το ανθρώπινο σώμα κινείται προς το έδαφος. Αυτή η φάση μπορεί να αναγνωριστεί, διότι η τιμή του SMV θα είναι σχεδόν ίση με μηδέν και μικρότερη από 0.6g (όπου g η σταθερά της βαρύτητας ίση με 9.81 m/s²).
3. Φάση πρόσκρουσης, όπου το σώμα χτυπάει το έδαφος και παρατηρείται μία απότομη αιχμή στο SMV, της τάξης των 1.8g και πάνω (συνήθως κοντά στα 3g). Σε κάποιες περιπτώσεις, μετά την αιχμή της αρχικής πρόσκρουσης, μπορεί να υπάρξουν κάποιες μικρότερες αιχμές για κάποιο ελάχιστο χρονικό διάστημα.
4. Φάση μετά-πτώσης, κατά την οποία το σώμα μπορεί να βρίσκεται σταθερό και ξαπλωμένο στο πάτωμα. Αυτή η φάση μπορεί να διαφέρει από πτώση σε πτώση, καθώς σε μία πτώση που δεν ήταν τόσο σοβαρή, ο ασθενής μπορεί να ανακάμψει γρήγορα και να συνεχιστούν τα ADLs. Σε περίπτωση σοβαρής πτώσης όμως μπορεί λόγω σοβαρού τραυματισμού ή απώλειας αισθήσεων να παραμείνει σε αυτή την οριζόντια θέση για αρκετό διάστημα.



Εικόνα 1: Τυπική μεταβολή επιτάχυνσης κατά τη διάρκεια πτώσης

Υπάρχουν κάποιες κατηγορίες πτώσεων (πχ πτώση από σκάλα) που μπορεί να αποκλίνουν από αυτές τις φάσεις αλλά ο γενικός κανόνας για τις περισσότερες πτώσεις

βασίζεται στην ανάλυση αυτών των φάσεων. Όσες περισσότερες από αυτές τις φάσεις μελετά ο εκάστοτε αλγόριθμος, τόσο καλύτερα είναι και τα αποτελέσματα του.

Κάποιες τεχνικές χρησιμοποιούν επίσης (συνήθως σε συνεργασία με το επιταχυνσιόμετρο) και μετρήσεις από κάποιον άλλο αισθητήρα (μαγνητόμετρο, γυροσκόπιο) για να υπολογίσουν τη διεύθυνση του ατόμου πριν και μετά από μια πιθανή πτώση. Η τεχνική που ακολουθείται συνήθως είναι ο υπολογισμός της γωνίας μεταξύ της επιτάχυνσης στον άξονα που “δείχνει” προς τα κάτω, δηλαδή προς την κατεύθυνση της βαρύτητας της γης και του διανύσματος επιτάχυνσης. Αν δηλαδή ο άξονας ο οποίος δείχνει προς τα κάτω είναι ο z, τότε η γωνία (φ) υπολογίζεται ως εξής:

$$\cos\varphi = \frac{A_z}{\sqrt{A_x^2 + A_y^2 + A_z^2}}$$

Σύμφωνα με το [4], ένας άνθρωπος είναι σε όρθια θέση όταν $-35^\circ < \varphi < 35^\circ$. Οπότε με βάση αυτό, όταν βρεθεί από το επιταχυνσιόμετρο μία μέτρηση που δείχνει πρόσκρουση και στη συνέχεια η θέση του ατόμου δεν είναι όρθια με βάση τη γωνία φ , τότε μπορούμε να πούμε με μεγαλύτερη σιγουριά ότι έχουμε πτώση. Όλα αυτά βέβαια προϋποθέτουν ότι η συσκευή θα βρίσκεται πάνω στο ελεγχόμενο σώμα σε όρθια θέση (πχ στη ζώνη ή στην τσέπη).

Οι τεχνικές κατωφλίου προτιμώνται γενικά στη βιβλιογραφία για την αντιμετώπιση του προβλήματος ανίχνευσης πτώσης, αφού είναι πιο φθηνές και εύκολες στην κατανόηση από τις τεχνικές μηχανικής μάθησης, δίνοντας παράλληλα έναν ικανοποιητικό βαθμό ακρίβειας. Επιπλέον, στις τεχνικές κατωφλίου ο αλγόριθμος μπορεί να τρέξει απευθείας στη συσκευή, χωρίς μεσολάβηση κάποιου server, ενώ στις τεχνικές μηχανικής μάθησης είναι απαραίτητη η ύπαρξη βάσης και μοντέλου.

2.1.2 Τεχνικές μηχανικής μάθησης

Οι τεχνικές βασισμένες σε αλγόριθμους μηχανικής μάθησης είναι η άλλη μεγάλη κατηγορία τεχνικών αναγνώρισης πτώσης.

Οι τεχνικές μηχανικής μάθησης για αναγνώριση πτώσης έχουν ως βασικά χαρακτηριστικά, σύμφωνα με το [5], τα στάδια της εκπαίδευσης ενός μοντέλου που να αναγνωρίζει πτώσεις και της χρήσης του σε νέα δεδομένα. Για την κατασκευή ενός μοντέλου, απαιτείται η συλλογή των ανεπεξέργαστων δεδομένων και ενδεχομένως μια προ επεξεργασία πάνω στα δεδομένα πριν αυτά δοθούν στον αλγόριθμο κατασκευής μοντέλου. Τέλος, μετά τη κατασκευή του μοντέλου, εφαρμόζονται σε αυτό κάποιες τεχνικές για την αξιολόγηση του, ως προς την ακρίβεια, την ανάκληση και άλλα χαρακτηριστικά.

Κατά την **συλλογή** των δεδομένων, συλλέγονται σχετικές πληροφορίες από τους αισθητήρες του κινητού όπως ταχύτητα, συντεταγμένες επιτάχυνσης, γωνιακή ταχύτητα, περίοδοι αδράνειας κλπ. Οι πιο δημοφιλείς αισθητήρες στη βιβλιογραφία, αναφορικά με λύσεις του προβλήματος από κινητά, είναι το επιταχυνσιόμετρο, το γυροσκόπιο και το βαρόμετρο.

Κατά την **προ-επεξεργασία** των δεδομένων, τα δεδομένα περνούν από μια σειρά μετασχηματισμών ώστε να καθαριστεί ο θόρυβος, όσο αυτό είναι δυνατό, αλλά ταυτόχρονα να διατηρηθεί η βασική πληροφορία. Οι μετασχηματισμοί προ επεξεργασίας

περιλαμβάνουν, μεταξύ άλλων, την ομογενοποίηση, την ομαλοποίηση, την ενίσχυση σήματος και την διακριτοποίηση συνεχών μεταβλητών.

Εκτός από τους μετασχηματισμούς που μπορεί να εφαρμοστούν στα δεδομένα, μπορεί να γίνει και μια επιλογή, μεταξύ των διαφορετικών δεδομένων εισόδου, για την χρησιμοποίηση αυτών που είναι πιο σχετικά με την αναγνώριση πτώσεων. Έτσι, ο αριθμός των δεδομένων εισαγωγής για τη κατασκευή του μοντέλου, και κατ' επέκταση η πολυπλοκότητα κατασκευής και εφαρμογής του μοντέλου μπορεί να μειωθεί, χωρίς αυτό να έχει επίπτωση στην αποδοτικότητά του. Επιπροσθέτως, μπορεί να βοηθήσει στην κατασκευή ενός πιο γενικού μοντέλου. Αυτή η επιλογή των πιο αντιπροσωπευτικών δεδομένων εισαγωγής, μπορεί να γίνει με κάποιο αλγόριθμο αυτόματης απόδοσης σειράς κατάταξης (ranking methods), ή μπορεί να είναι ενσωματωμένη στον αλγόριθμο μηχανικής μάθησης που θα χρησιμοποιηθεί στο επόμενο στάδιο.

Στο στάδιο της **εκπαίδευσης** του μοντέλου ανίχνευσης πτώσεων, δεδομένα που έχουν συλλεχθεί και μετασχηματιστεί κατάλληλα χρησιμοποιούνται ως είσοδος σε αλγόριθμους ταξινόμησης για την κατασκευή των μοντέλων. Είναι εμφανές, πως στη συγκεκριμένη οικογένεια τεχνικών, είναι αναγκαίο ένα αρχικό σύνολο δεδομένων, επισημασμένο με την πληροφορία της πτώσης ή μη, ώστε να μπορεί να εκπαιδευτεί το μοντέλο.

Οι αλγόριθμοι κατηγοριοποίησης που έχουν χρησιμοποιηθεί για το συγκεκριμένο πρόβλημα περιλαμβάνουν τεχνικές βασισμένες σε κανόνες και αλγόριθμους μηχανικής μάθησης όπως τα νευρωνικά δίκτυα και νευρώνες perceptron, δέντρα αποφάσεων, Naive Bayes, μοντέλα Hidden Markov, αλγόριθμους κοντινότερου γείτονα, εκμάθηση βασισμένη στα παραδείγματα, συστήματα ασαφούς λογικής, Gaussian μοντέλα, Μηχανές Διανυσμάτων Υποστήριξης (Support Vector Machines) και πολλά ακόμα.

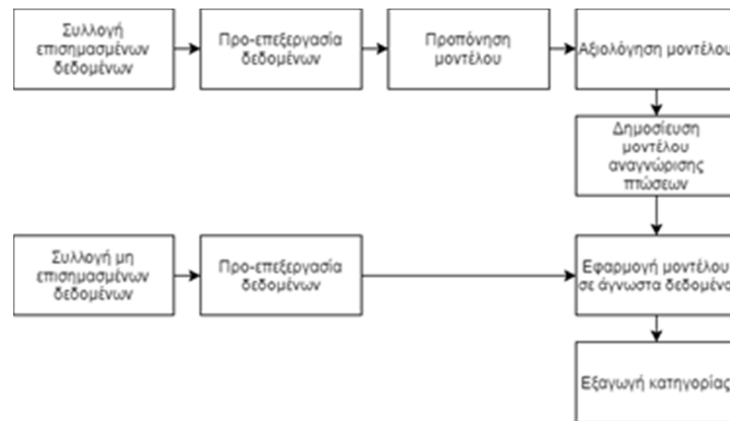
Πριν τη χρήση του μοντέλου, χρειάζεται συνήθως, το τελευταίο να **αξιολογηθεί** ως προς κάποια βασικά χαρακτηριστικά του. Για το σκοπό αυτό, κάποια από τα επισημασμένα δεδομένα, που είδαμε στο στάδιο κατασκευής μοντέλου, δε τα χρησιμοποιούμε για τη κατασκευή του μοντέλου. Αντίθετα, τα φυλάμε για το στάδιο της αξιολόγησης όπου τα εισαγάγουμε ως άγνωστα δεδομένα στο μοντέλο και καταγράφουμε την απόδοση του τελευταίου.

Οι βασικές μετρικές που χρησιμοποιούνται για την αξιολόγηση ενός μοντέλου, είναι η συνολική ορθότητα (accuracy), η ακρίβεια (precision), η ανάκληση (recall), η εξειδίκευση (specificity) και ο δείκτης false-positive. Το F-score είναι μια μετρική που συνδυάζει την ακρίβεια και την ανάκληση του μοντέλου. Όλες οι παραπάνω μετρικές βασίζονται στο πίνακα σύγχυσης που περιέχει τις περιπτώσεις:

- Εγγραφές της κλάσης «Πτώση» που προβλέπονται ως κλάση «Πτώση» (True Positive)
- Εγγραφές της κλάσης «Πτώση» που προβλέπονται ως κλάση «Μη πτώση» (False Negative)
- Εγγραφές της κλάσης «Μη Πτώση» που προβλέπονται ως κλάση «Πτώση» (False Positive)
- Εγγραφές της κλάσης «Μη Πτώση» που προβλέπονται ως κλάση «Μη πτώση» (True Negative)

Οι καμπύλες λειτουργικού χαρακτηριστικού δέκτη (ROC) χρησιμοποιούνται επίσης ως ένα εργαλείο αξιολόγησης.

Στο στάδιο της **χρήσης** του μοντέλου, δεδομένα που δεν έχουν επισημανθεί με την πληροφορία της πτώσης ή μη, μπορούν να δοθούν ως είσοδος στο μοντέλο το οποίο μπορεί να κατηγοριοποιήσει την ενέργεια. Κάποια μοντέλα επιτρέπουν την εσαεί εκμάθηση με τα καινούργια δεδομένα που εισέρχονται κάθε φορά στο σύστημα.



Εικόνα 2: Διαδικασία κατασκευής, αξιολόγησης και χρήσης μοντέλου

Στην Εικόνα 2 παρουσιάζεται ο κύκλος ζωής μιας τεχνικής μηχανικής μάθησης όπως τον παρουσιάσαμε παραπάνω.

Οι αρχιτεκτονικές που εκμεταλλεύονται τεχνικές μηχανικής μάθησης συνήθως απαιτούν υψηλό υπολογιστικό κόστος, μαζική ανάλυση δεδομένων, πρόσβαση σε βάσεις δεδομένων και μακρές περιόδους εκπαίδευσης όπου ο αλγόριθμος ταξινόμησης πρέπει να παραμετροποιηθεί ή να προσαρμοστεί στα ίχνη κίνησης ενός συνόλου πειραματικών χρηστών.

Πάρα ταύτα, με την αύξηση της χωρητικότητας σε μπαταρία, των νεότερων, υψηλής τεχνολογίας κινητών τηλεφώνων, οι συγκεκριμένες τεχνικές προσελκύουν ολοένα και περισσότερο ερευνητικό ενδιαφέρον, από τη στιγμή που υπό προϋποθέσεις (ποιοτικά και πολλά δεδομένα για εκπαίδευση μοντέλου), μπορούν να δώσουν καλύτερα αποτελέσματα από τις τεχνικές κατωφλίου [6].

2.2 Εφαρμογές ανίχνευσης πτώσης

Αφού μελετήσαμε τους αλγόριθμους που χρησιμοποιούνται από τα διάφορα συστήματα ανίχνευσης πτώσεων, σε αυτή την ενότητα θα εξετάσουμε τις σημαντικότερες εφαρμογές που έχουν αναπτυχθεί σε αυτό τον τομέα.

Μια σημαντική προϋπόθεση κατά την ανάπτυξη ενός συστήματος ανίχνευσης πτώσεων είναι ότι πρέπει η ανάπτυξη να γίνει για ένα σύστημα το οποίο θα συνδέεται άμεσα με το χρήστη και το οποίο -ιδανικά- ο χρήστης θα έχει συνεχώς πάνω του. Παλιότερα η ανίχνευση πτώσεων γινόταν με συστήματα που χρησιμοποιούσαν δικό τους υλικό και λογισμικό (κάμερες, αισθητήρες περιβάλλοντος, αισθητήρες πίεσης, τεχνολογίες

υπέρυθρων κλπ.), το οποίο ο χρήστης θα έπρεπε να προμηθευτεί και ουσιαστικά ήταν εξειδικευμένο για αυτή την εφαρμογή. Με αυτόν τον τρόπο αυξανόταν και το κόστος της χρήσης ενός τέτοιου συστήματος, καθώς και ο χρόνος εγκατάστασης, εκπαίδευσης και προσαρμογής, και επίσης υπήρχαν ζητήματα καταπάτησης της ιδιωτικότητας του (ηλικιωμένου) χρήστη. Όλα αυτά απέτρεπαν γενικά τη χρήση τέτοιου είδους συστημάτων και την περιορίζαν σε λίγους, πιο ευκατάστατους συνήθως χρήστες [7].

Σήμερα ωστόσο, η ανίχνευση πτώσης μπορεί να γίνει χωρίς όλα τα παραπάνω κόστη, μέσω μιας συσκευής που κατέχουν δισεκατομμύρια άνθρωποι σε όλο τον κόσμο, και δεν είναι άλλη από τα “έξυπνα κινητά τηλέφωνα” (smartphones), τα οποία διαθέτουν ως επί το πλείστον τους απαραίτητους αισθητήρες για να μπορέσει να λειτουργήσει μια εφαρμογή ανίχνευσης πτώσεων. Εκτός από αυτό, μπορεί να χρησιμοποιηθεί το υπάρχον υλικό και λογισμικό των smartphones και για πολλές άλλες λειτουργίες (κλήση έκτακτης ανάγκης, απόκριση με οθόνη αφής, φωνητικές εντολές κλπ.). Προσφάτως, εκτός από τα smartphones έχουν έρθει στην επιφάνεια και χρησιμοποιούνται αρκετά και τα “έξυπνα ρολόγια” (smartwatches) ή γενικά φορετές συσκευές (wearables) που συνδέονται με το smartphone, τα οποία ενδεχομένως να φανούν ακόμα πιο χρήσιμα στο μέλλον για τέτοιου είδους εφαρμογές, καθώς είναι μια συσκευή, που είναι παράλληλα και αξεσουάρ, το οποίο ο χρήστης έχει συνέχεια πάνω του, σε αντίθεση με το κινητό, το οποίο μπορεί να μην το έχει κάποιες φορές.

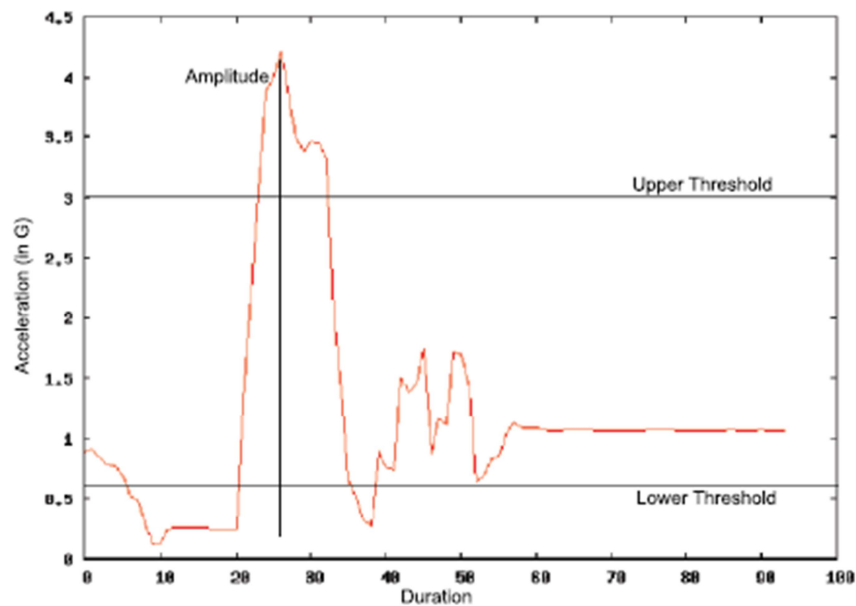
Στις επόμενες υποενότητες θα δούμε εφαρμογές που έχουν αναπτυχθεί τόσο για κινητά όσο και για φορετές συσκευές.

2.2.1 Εφαρμογές κινητών

Παρακάτω θα αναλύσουμε κάποιες εφαρμογές ανίχνευσης πτώσεων που έχουν αναπτυχθεί για κινητά τηλέφωνα:

- **iFall**

Το iFall [7] είναι μία εφαρμογή για Android smartphones, η οποία χρησιμοποιεί τεχνικές κατωφλίου για ανίχνευση πτώσης. Ουσιαστικά η πτώση χωρίζεται στις φάσεις που έχουμε περιγράψει στο 2.1.1 και οι συγγραφείς ελέγχουν κάποιες προϋποθέσεις για να εξαλείψουν όσο γίνεται τα ψευδώς θετικά (false positive) αποτελέσματα. Ουσιαστικά, το πρώτο σημάδι που δείχνει ότι έχουμε πιθανή πτώση είναι το εύρος (amplitude) της επιτάχυνσης, δηλαδή του διανύσματος SMV. Αν αυτό περάσει μέσα σε μικρό χρονικό διάστημα κάτω από το χαμηλό όριο (lower threshold) και στη συνέχεια πάνω από το υψηλό όριο (upper threshold) τότε έχουμε ένα σημάδι πτώσης (Εικόνα 3)



Εικόνα 3: Όρια που χρησιμοποιήθηκαν στο iFall

Το παραπάνω ως ένδειξη δεν είναι όμως αρκετό για να πει κάποιος με σιγουριά ότι υπάρχει πτώση καθώς μπορεί να εμφανιστεί και σε άλλα ADLs. Έτσι οι συγγραφείς κάνουν την υπόθεση ότι μία πτώση μπορεί να αρχίσει μόνο από όρθια θέση και να καταλήξει σε οριζόντια, οπότε η διαφορά στην θέση πριν και μετά είναι περίπου 90° , οπότε αυτή είναι η δεύτερη παράμετρος που χρησιμοποιείται από το iFall. Τέλος, για να εξαλείψουν τα false positives, ορίζουν ένα διάστημα μετά την ενδεχόμενη πτώση, κατά το οποίο μπορεί να γίνει ανάκαμψη, για παράδειγμα σε περίπτωση που έπεσε απλά το κινητό ή η πτώση δεν ήταν σοβαρή. Αν σε αυτό το διάστημα επιστρέψει το κινητό σε όρθια θέση, ο αλγόριθμος ξεκινάει από την αρχή. Σε διαφορετική περίπτωση έρχεται αρχικά μια ειδοποίηση στο κινητό του χρήστη (ηλικιωμένου) την οποία καλείται να κλείσει. Αν δεν την κλείσει μέσα σε κάποιο χρονικό διάστημα, τότε μόνο στέλνεται ειδοποίηση στην επαφή έκτακτης ανάγκης για να βοηθήσει.

Επίσης, το iFall δίνει την δυνατότητα στον χρήστη να προσαρμόσει την ευαισθησία (sensitivity) της εφαρμογής, αυξάνοντας ή μειώνοντας τα πάνω και κάτω όρια ανάλογα το πόσο ευαίσθητη θέλει να είναι η εφαρμογή σε πιθανές πτώσεις. Άλλη μία βελτιστοποίηση που έχει υλοποιηθεί είναι η δυναμική μεταβολή του άνω ορίου ανάλογα με το σε ποιο σημείο του σώματος βρίσκεται η συσκευή, για παράδειγμα αν βρίσκεται στο χέρι, που είναι ένα σημείο που έχει μεγαλύτερες επιταχύνσεις, τότε το άνω όριο θα πρέπει να αυξηθεί, ενώ αν βρίσκεται στον κορμό, πρέπει να μειωθεί. Ανάλογη βελτιστοποίηση έχει πραγματοποιηθεί για την δυναμική αλλαγή της αρχικής θέσης, ανάλογα αν το κινητό βρίσκεται κάθετα ή οριζόντια πάνω στο σώμα. Τέλος, επειδή κάποιες κινήσεις όπως το να απαντήσει κάποιος ή να τερματίσει μια κλήση, οδηγούν σε κινήσεις του κινητού που μπορεί να συγχυστούν με πτώσεις, οι συγγραφείς έχουν κάνει κάποιες βελτιστοποιήσεις, έτσι ώστε όταν το κινητό είναι σε κλήση, να αποφεύγονται τα false positives.

Τέλος, πρέπει να αναφέρουμε ότι οι συγγραφείς δεν μας παραθέτουν κάποια αξιολόγηση για το σύστημα τους.

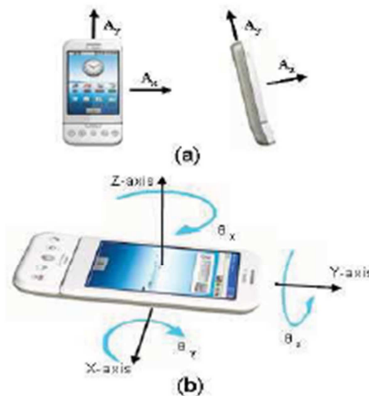
- **PerFallD**

Το PerFallD [8] χρησιμοποιεί επίσης την τεχνολογία Android για την ανίχνευση πτώσεων. Ο αλγόριθμος που έχει αναπτυχθεί και όλες οι λειτουργίες της εφαρμογής τρέχουν στην πλευρά του κινητού, χωρίς τη χρήση κάποιου εξυπηρετητή. Αρχικά δημιουργείται ένα προφίλ για τον χρήστη ανάλογα με τις ανάγκες του, το οποίο περιλαμβάνει βασικές ρυθμίσεις, όπως τη λίστα επαφών έκτακτης ανάγκης, καθώς και τη συχνότητα δειγματοληψίας δεδομένων σε διάφορα βασικά σενάρια. Στη συνέχεια αρχίζει να τρέχει το κυρίως πρόγραμμα, ως υπηρεσία παρασκηνίου (background service) και όταν εντοπιστεί πτώση γίνεται ενεργοποίηση ενός συναγερμού και ενός χρονοδιακόπτη, ο οποίος αν δεν απενεργοποιηθεί από το χρήστη, τότε στέλνεται ειδοποίηση στην επαφή έκτακτης ανάγκης που έχει καθορίσει. Για όλη αυτή τη διαδικασία, οι συγγραφείς αναφέρουν ότι έχουν παρθεί μέτρα ώστε η κατανάλωση της μπαταρίας και γενικά των πόρων του κινητού να είναι η ελάχιστη δυνατή.

Για τον αλγόριθμο έχουν χρησιμοποιηθεί δύο μετρικές. Η πρώτη είναι το SMV που έχουμε αναφέρει στο 2.1.1 και η δεύτερη είναι η παρακάτω, η οποία χρησιμοποιεί τις γωνίες κύλισης, ανύψωσης, στροφής (roll, pitch, yaw) ή αλλιώς γωνίες Euler (θ_x , θ_y και θ_z αντίστοιχα), οι οποίες υπολογίζονται από το γυροσκόπιο του κινητού:

$$|A_v| = |A_x * \sin(\theta_z) + A_y * \sin(\theta_y) - A_z * \cos(\theta_y) * \cos(\theta_z)|$$

όπου οι x,y,z άξονες ορίζονται όπως στην Εικόνα 4.



Εικόνα 4: Ορισμός των αξόνων του PerFallD

Τόσο οι τιμές του SMV όσο και του A_v συγκρίνονται με κάποια όρια που έχουν ορίσει οι συγγραφείς και αν και τα δύο ξεπερνούν τα συγκεκριμένα κατώφλια τότε ανιχνεύεται μία πτώση.

Για την αξιολόγηση του συστήματος τους, οι συγγραφείς δοκίμασαν να συλλέξουν δεδομένα από πολλά είδη πτώσεων, καθώς και από διάφορα άλλα ADLs, τα οποία εκτέλεσαν νεαροί εθελοντές. Η απόδοση του συστήματος μετρήθηκε ως ποσοστό των ψευδώς θετικών (false positives) και των ψευδώς αρνητικών (false negatives) μέσα στο σύνολο των πτώσεων. Το συμπέρασμα που προέκυψε από τα πειράματά τους είναι ότι η ιδανική θέση πάνω στο σώμα για το κινητό ώστε να μειωθούν τα false positives/negatives

είναι η μέση, καθώς σε αυτή τη θέση τα false negatives ήταν στο 2.67% και τα false positives στο 8.7%.

- **Fall Detection Using Single Tri-Axial Accelerometer [9]**

Στη συγκεκριμένη εργασία, οι συγγραφείς χρησιμοποιούν τον αισθητήρα του επιταχυνσιόμετρου από Windows Phone για την ανίχνευση πτώσεων με βάση τον ακόλουθο αλγόριθμο:

- Καταρχήν αποθηκεύουν διανύσματα επιτάχυνσης με ρυθμό δειγματοληψίας 100Hz.
- Κάθε δευτερόλεπτο εφαρμόζεται στα δεδομένα ένα “φίλτρο υψηλής διέλευσης” (A_h) καθώς και ένα “φίλτρο χαμηλής διέλευσης” (A_l). Στο πρώτο, οι τιμές αντιστοιχούν στην επιτάχυνση λόγω κίνησης του χρήστη, ενώ στο δεύτερο στην επιτάχυνση λόγω βαρύτητας.
- Στη συνέχεια υπολογίζεται η “ L^1 νόρμα του φίλτρου υψηλής διέλευσης” ως εξής:

$$|A_h|_1 = \sum_{i=0}^{99} |A_{hx_i}| + |A_{hy_i}| + |A_{hz_i}|$$

- Αν η τιμή αυτή είναι πάνω από ένα (εμπειρικά) προκαθορισμένο όριο (E_h) σημαίνει ότι ο χρήστης είτε κάνει κάποια έντονη δραστηριότητα, όπως το να τρέχει ή να πηδάει, είτε ότι έχει πέσει.
- Αν το παραπάνω ισχύει, τότε μετά από 3 δευτερόλεπτα υπολογίζεται η διεύθυνση της συσκευής με βάση τον παρακάτω τύπο (υποθέτοντας ότι ο y είναι ο κάθετος άξονας):

$$\theta_i = \cos^{-1}\left(\frac{A_{ly_i}^2}{\sqrt{A_{lx_i}^2 + A_{ly_i}^2 + A_{lz_i}^2}}\right)$$

- Με βάση τις 100 διαφορετικές τιμές του θ χρησιμοποιείται ένας αλγόριθμος ψηφοφορίας, όπου όλα τα θ_i συγκρίνονται με ένα κατώφλι θ_h και αν είναι μεγαλύτερα παίρνουν μία νίκη, αλλιώς μία ήττα. Αν οι νίκες είναι περισσότερες από τις ήττες τότε σημαίνει ότι ο χρήστης έχει πέσει.
- Αν η πτώση είναι επιπόλαια ώστε ο χρήστης να ανακάμπτει αμέσως, τότε για να μην ανοίξει ο συναγερμός χωρίς να χρειάζεται, στα επόμενα 5 δευτερόλεπτα, οι δύο παραπάνω μετρικές επαναυπολογίζονται και αν το $|A_h|_1$ ξεπερνάει ένα όριο και το θ_i δείχνει ότι ο χρήστης σηκώθηκε, τότε ο συναγερμός δεν ξεκινάει.
- Στο τελευταίο στάδιο η πτώση θεωρείται σίγουρη, αλλά οι συγγραφείς υπολογίζουν και την περίπτωση να έπεσε καταλάθος το τηλέφωνο. Για να “πιάσουν” αυτές τις περιπτώσεις δίνονται 5 δευτερόλεπτα στο χρήστη να ακυρώσει το συναγερμό, πριν στείλει τις GPS συντεταγμένες και το

προκαθορισμένο μήνυμα στους δικούς του.

Με βάση την (περιορισμένη) αξιολόγηση που μας παρουσιάζουν οι συγγραφείς, φαίνεται ο αλγόριθμος να έχει καλή ακρίβεια με κάποια false positives και false negatives σε ορισμένες περιπτώσεις.

- **E-FallD**

Σε αυτή τη δημοσίευση [10], οι συγγραφείς έχουν αναπτύξει άλλη μία android εφαρμογή, η οποία χρησιμοποιεί μια κλασική μεθοδολογία με όρια και η κύρια συνεισφορά της είναι ότι τα όρια αυτά προσαρμόζονται με βάση το βάρος, το ύψος, το φύλο και την ηλικία του χρήστη.

Αρχικά ορίζουν κάποιες σταθερές αρχικές τιμές: ο χρόνος “παραθύρου” $T_{win} = 1,5s$, το πάνω όριο $Th_{max} = 1.75g$ και το κάτω όριο $Th_{min} = 0.75g$. Από αυτά, τα T_{win} και Th_{max} προσαρμόζονται με βάση τα χαρακτηριστικά του χρήστη. Το βάρος και το ύψος συνδυάζονται σε μία μετρική που είναι το BMI (δείκτης μάζας σώματος και το οποίο είναι ίσο με:

$$BMI = \text{mass}(\text{kg}) / \text{height}^2 (\text{m})$$

Τα όρια λοιπόν προσαρμόζονται με βάση τον πίνακα παρακάτω:

Πίνακας 1: Προσαρμογή των παραμέτρων του E-FallD με βάση την ηλικία, το φύλο και το BMI

	age		sex	
	≥ 60	< 60	M	F
Th_{max}		+0.05g		-0.05g
T_{win}		+0.1s	+0.1s	
	BMI			
	< 20	20-25	25-30	> 30
Th_{max}	-0.05g		+0.05g	+0.1g
T_{win}	-0.1s		+0.1s	+0.2s

Ο αλγόριθμος που χρησιμοποιείται είναι ο ακόλουθος: Αρχικά μετρίεται το διάνυσμα επιτάχυνσης μέσα σε παράθυρα T_{win} δευτερολέπτων με 50% επικαλύψεις. Για κάθε δείγμα υπολογίζεται η μέγιστη και η ελάχιστη επιτάχυνση (A_{max} και A_{min}) καθώς και οι χρόνοι που αυτές υπολογίστηκαν (T_{max} και T_{min}). Η δραστηριότητα καταγράφεται ως πτώση όταν $A_{max} > Th_{max}$ και $A_{min} < Th_{min}$ και $T_{min} < T_{max}$, αλλιώς καταγράφεται ως άλλο ADL.

Με βάση την αξιολόγηση με πειράματα σε χρήστες διαφορετικών χαρακτηριστικών -μία φορά με την χρήση των χαρακτηριστικών και μία χωρίς- διαπιστώθηκε ότι τα ποσοστά επιτυχίας εύρεσης πτώσης ήταν κατά λίγο καλύτερα με τη χρήση των χαρακτηριστικών του ατόμου σε σχέση με τον κλασικό αλγόριθμο, καθώς υπήρχαν λιγότερα false positives και false negatives.

- **Σύστημα Albert et al.**

Οι συγγραφείς του [11] εφάρμοσαν τεχνικές μηχανικής μάθησης για αναγνώριση πτώσεων αλλά και για την αυτόματη ταξινόμηση του τύπου της πτώσης.

Πιο συγκεκριμένα, ζητήθηκε από 15 άτομα να προσομοιώσουν τέσσερις διαφορετικούς τύπους πτώσεων - αριστερά και δεξιά, προς τα εμπρός και προς τα πίσω ενώ συγκέντρωναν μετρήσεις επιταχυνσιόμετρου από τα κινητά τους τηλέφωνα. Εννέα ακόμα άτομα φορούσαν τις κινητές συσκευές για δέκα ημέρες, για να παρέχουν δεδομένα καθημερινών δραστηριοτήτων προς σύγκριση με τις προσομοιωμένες πτώσεις. Με αυτό τον τρόπο μπόρεσαν να αξιολογήσουν αν το σύστημα τους έχει τη δυνατότητα να αναγνωρίσει τις πραγματικές πτώσεις από δραστηριότητες της καθημερινής ζωής που μοιάζουν με πτώσεις (fall-like δραστηριότητες), όπως είναι η απότομο κάθισμα σε καρέκλα ή κρεβάτι.

Για την ανίχνευση πτώσεων χρησιμοποιήθηκαν πέντε διαφορετικοί αλγόριθμοι ταξινόμησης: Μηχανές Διανυσμάτων Υποστήριξης (SVMs), sparse multinomial logistic regression (SMLR), Naive Bayes, δέντρα αποφάσεων, και τον αλγόριθμο των κοντινότερων γειτόνων. Οι αξιολόγηση των αλγόριθμων έγινε με τη μέθοδο της 10-διασταυρωμένης επικύρωσης (10-cross validation). Από τους παραπάνω αλγόριθμους, τα SVMs και το SMLR ήταν σε θέση να αναγνωρίσουν τις πτώσεις από τις καθημερινές δραστηριότητες με 98% ακρίβεια και τις διαφορετικές πτώσεις μεταξύ τους με ακρίβεια 99%.

Η μεγαλύτερη συνεισφορά αυτής της εργασίας ήταν η επαλήθευση πως μπορεί να γίνει εκμετάλλευση ενός μεγάλου αριθμού εξαγομένων χαρακτηριστικών από τους αισθητήρες εύκολα με τη χρήση αλγόριθμων μηχανικής μάθησης. Αντί να αναλύονται τα δεδομένα χειροκίνητα σχετικά με το πιο χαρακτηριστικό περιγράφει καλύτερα ένα είδος πτώσης ή μια καθημερινή δραστηριότητα, οι αλγόριθμοι μηχανικής μάθησης καλούνται να συνδυάσουν, μεταβάλλοντας βάρη, τα διάφορα χαρακτηριστικά με τον τρόπο που αυτά θεωρούν αποδοτικό.

Στους περιορισμούς της προσέγγισης αξιολόγησης του συστήματος, συγκαταλέγονται η θέση του κινητού κατά τη διάρκεια των πειραμάτων καθώς και η αβεβαιότητα για τη γενικότητα των fall-like δραστηριοτήτων. Αναφορικά με τη θέση του κινητού, αυτό ήταν σε σταθερή θέση πάνω στον φορέα κάτι το οποίο δεν συμβαδίζει με τον τρόπο που οι άνθρωποι μεταφέρουν τα κινητά τους τηλέφωνα κάθε μέρα. Όσον αφορά την αβεβαιότητα σχετικά με τη γενικότητα των καθημερινών δραστηριοτήτων, οι συγγραφείς αναφέρουν πως δεν είναι σε θέση να αξιολογήσουν αν τα δεδομένα που έλαβαν από τα εννέα άτομα περιέχουν όλες τις περιπτώσεις fall-like δραστηριοτήτων. Ο δείκτης των false-positives θα πρέπει να εξεταστεί εξονυχιστικά ώστε να επιβεβαιωθεί η γενική ακρίβεια της εφαρμογής

- **Σύστημα Abbatea et al.**

Το [12] αποτελεί ένα υβριδικό σύστημα τεχνικής κατωφλίου και τεχνικής μηχανικής μάθησης.

Παρότι για την ανίχνευση πτώσης χρησιμοποιεί τη κλασική μεθοδολογία με τα όρια που περιγράφηκε στο 2.1.1, στη συνέχεια το αποτέλεσμα φιλτράρεται μέσω ενός μοντέλου που

κατηγοριοποιεί την ενέργεια σε πτώση ή σε κάποια από τις δραστηριότητες της καθημερινής ζωής που θα μπορούσαν να παρερμηνευτούν ως πτώσεις από τη τεχνική κατωφλίων. Πιο συγκεκριμένα, το μοντέλο διαχωρίζει την ενέργεια σε περπάτημα/τρέξιμο, κάθισμα/ξάπλωμα, ακούσιο χτύπημα του κινητού, ή πτώση.

Ως μοντέλο χρησιμοποίησαν ένα νευρωνικό δίκτυο το οποίο εκπαιδεύτηκε με δεδομένα 86 δραστηριοτήτων από 7 εθελοντές. Το σύστημα τους κατάφερε να ανιχνεύσει όλες τις πτώσεις/μη πτώσεις επιτυχώς. Αντίθετα, όταν εφάρμοσαν στα δεδομένα την μεθοδολογία του iFall που περιγράψαμε παραπάνω τα αποτελέσματα ήταν 81.4% συνολική ορθότητα συστήματος, με 81.8% ευαισθησία (true positive) και 81.0% εξειδίκευση (true negative).

Μια ακόμα συνεισφορά αυτής της εργασίας είναι τα συμπεράσματα που εξήγαγαν από συνέντευξη που πήραν από μια ομάδα 10 ηλικιωμένων. Συγκεκριμένα, συμπεράναν πως η χρηστικότητα παίζει βασικό ρόλο στην υιοθέτηση ενός συστήματος ανίχνευσης πτώσης. Το να αναγκάσουν τους χρήστες να κρατούν το κινητό στη ζώνη τους σε συγκεκριμένη κλίση, όπως αρχικά σχεδίαζαν, θα είχε ως αποτέλεσμα την δυσαρέσκεια μεγάλου μέρους των χρηστών τους. Έτσι, αποφάσισαν και υλοποίησαν μια μικρή εξωτερική μονάδα ανίχνευσης που μπορεί να βρίσκεται στη ζώνη του ηλικιωμένου και να επικοινωνεί τα δεδομένα που συλλέγει στο κινητό του μέσω Bluetooth. Ο ηλικιωμένος πλέον έχει την ελευθερία να χρησιμοποιεί το κινητό όποια στιγμή το επιθυμεί και να την τοποθετεί όπου τον βολεύει.

2.2.2 Τεχνολογία φορετών συσκευών (wearables)

Τα wearables ή έξυπνα ρολόγια (smartwatches) είναι μια σχετικά νέα τεχνολογία που χρησιμοποιεί το Android και μπορεί να εκτελέσει βασικές λειτουργίες κατά κύριο λόγο σε συνδυασμό με Android smartphone, με το οποίο είναι “δεμένο” και επικοινωνεί με κάποιον (ασύρματο συνήθως) τρόπο (bluetooth, wi-fi κτλ). Η συγκεκριμένη τεχνολογία έχει την προοπτική να βοηθήσει σημαντικά στην ανίχνευση πτώσης, καθώς προσφέρει αρκετά πλεονεκτήματα σε σχέση με το κινητό και ήδη έχουν γίνει κάποιες προσπάθειες να χρησιμοποιηθεί στον τομέα αυτό.

Ένα πλεονέκτημα που προσφέρουν τα smartwatches είναι η δυνατότητα να βρίσκονται πάνω στον ηλικιωμένο καθ’ όλη τη διάρκεια της ημέρας, ακόμα και σε περιστάσεις όπως το μπάνιο, σε αντίθεση με το κινητό, το οποίο μπορεί να μην το έχει κάποιος πάνω του σε όλες τις περιστάσεις και είναι επίσης πιο εύκολο να το ξεχάσει. Το ότι το smartwatch είναι παράλληλα και ρολόι ενθαρρύνει τους ηλικιωμένους να το φοράνε και έτσι η αναγνώριση της δραστηριότητας και η ανίχνευση πτώσης γίνεται συνεχώς, χωρίς να υπάρχει κίνδυνος να πέσει ο ηλικιωμένος ενώ δεν έχει πάνω του τη συσκευή και να μην ενημερωθεί έτσι κανένας για την πτώση του. Επίσης, επειδή το smartwatch μπορεί να πραγματοποιήσει και κλήσεις, είναι πιο εύκολο για τον ηλικιωμένο να επικοινωνήσει άμεσα με κάποιον όταν είναι σε κίνδυνο.

Ένα μειονέκτημα που ίσως να είναι σημαντικό είναι πως το smartwatch βρίσκεται στον καρπό, που είναι ένα σημείο με συνεχή κίνηση, ίσως και απότομη κάποιες φορές σε σχέση με μέρη του σώματος όπως η μέση που μένουν συνήθως πιο ακίνητα. Αυτό μπορεί να έχει ως αποτέλεσμα κάποιες ψευδώς θετικές ενδείξεις πτώσης, όμως μπορεί να δίνεται εύκολα η δυνατότητα στον ηλικιωμένο να κλείνει το “συναγερμό” της συσκευής σε περίπτωση που

δεν έχει συμβεί στα αλήθεια κάποια πτώση, και να μην πηγαίνει έτσι ενημέρωση στον υπεύθυνο.

Τέλος, θα πρέπει να αναφέρουμε ότι μία πολύ σημαντική λειτουργία που μπορεί να πραγματοποιηθεί με χρήση smartwatch είναι η μέτρηση του παλμού του ηλικιωμένου, αφού η πλειοψηφία των συσκευών αυτών διαθέτει αισθητήρα μέτρησης παλμού. Έτσι, μπορεί και πάλι να ειδοποιηθεί κάποιος υπεύθυνος όταν οι μετρήσεις δείχνουν πως κάτι δεν πάει καλά.

Προς το παρόν, δεν έχουν γίνει πάρα πολλές εργασίες με χρήση smartwatches, λόγω του ότι είναι πολύ καινούρια τεχνολογία. Παρακάτω θα αναφέρουμε μερικές εφαρμογές που έχουν κάνει κάποιες πρώτες προσπάθειες σε αυτή την κατεύθυνση:

- **Συνδυασμός smartphone και smartwatch για ανίχνευση πτώσης**

Στη συγκεκριμένη εργασία [13], οι συγγραφείς έχουν χρησιμοποιήσει ταυτόχρονα και τις δύο συσκευές για ανίχνευση πτώσης, εκμεταλλευόμενοι μια ποικιλία αλγορίθμων που τρέχουν παράλληλα και στο ρολόι και στο κινητό. Αυτό έχει γίνει για να εξαλείψουν πιθανά false positives που μπορούν να προέρχονται από τη χρήση μόνο του smartwatch λόγω απότομων κινήσεων στα χέρια, καθώς και για μεγαλύτερο βαθμό σιγουριάς, εφόσον δύο διαφορετικές συσκευές πάνω στον ηλικιωμένο εντοπίζουν πτώση.

Έχει αναπτυχθεί μία android εφαρμογή, η οποία τρέχει και στο τηλέφωνο και στο ρολόι και καταγράφει τις κινήσεις του ηλικιωμένου μέσω του επιταχυνσιόμετρου και του γυροσκοπίου, χρησιμοποιώντας βασικές τεχνικές κατωφλίου για εντοπισμό πτώσεων. Έχουν χρησιμοποιηθεί τέσσερις βασικοί αλγόριθμοι, από τους οποίους μπορεί να επιλεγεί ποιος θα χρησιμοποιηθεί κάθε φορά. Ο πρώτος συγκρίνει τη μέτρηση του SMV με κάποιο όριο, ο δεύτερος χρησιμοποιεί τη μέθοδο δείκτη πτώσης FI_i (Fall Index) όπου:

$$FI_i = \sqrt{\sum_{k=x,y,z} \sum_{i=19}^i (A_{k_i} - A_{k_{i-1}})^2}$$

Το FI_i χρησιμοποιεί τις 20 τελευταίες μετρήσεις επιτάχυνσης και συγκρίνει αυτή την τιμή με κάποιο όριο για να αποφασίσει για μια πτώση. Ο τρίτος αλγόριθμος που χρησιμοποιούν είναι παρόμοιος με αυτόν του perFallD και ο τέταρτος με το iFall.

Αν το ρολόι εντοπίσει κάποια πτώση, τότε μεταδίδεται ένα μήνυμα από το ρολόι στο κινητό. Για την επικοινωνία αυτή χρησιμοποιείται η τεχνολογία Bluetooth. Όταν το κινητό λάβει αυτό το μήνυμα προσπαθεί να εντοπίσει ένα συμβάν πτώσης σε ένα πολύ μικρό χρονικό διάστημα πριν ή μετά από την ώρα που έλαβε το μήνυμα (πχ ένα δευτερόλεπτο). Αν εντοπιστεί πτώση και από το κινητό, αρχίζει ο συναγερμός, τον οποίο αν ο χρήστης δεν κλείσει σε 20 δευτερόλεπτα, γίνεται κλήση ή στέλνεται μήνυμα στην επαφή έκτακτης ανάγκης.

Για την αξιολόγηση έχει χρησιμοποιηθεί μια έξυπνη τακτική, στην οποία μετρίεται ξεχωριστά το ποσοστό επιτυχίας ανίχνευσης πτώσης για το smartphone και για το smartwatch, καθώς και το ποσοστό επιτυχίας του συνδυασμού των δύο συσκευών. Στα πειράματα το smartphone τοποθετείται στην τσέπη, αφού λόγω της ύπαρξης και δεύτερης συσκευής οι συγγραφείς κρίνουν ότι δε χρειάζεται το κινητό να τοποθετηθεί σε “άβολο” για τον ασθενή σημείο όπως η μέση. Οι μετρικές που χρησιμοποιούνται είναι η ευαισθησία (sensitivity) και η εξειδίκευση (specificity). Η πρώτη δείχνει ουσιαστικά τον αριθμό των θετικών ενδείξεων πτώσης που κατηγοριοποιούνται σωστά ως θετικές και η δεύτερη τον αριθμό των αρνητικών ενδείξεων πτώσης που κατηγοριοποιούνται σωστά ως αρνητικές. Όπως αναφέρουν οι συγγραφείς, η χρήση και των δύο συσκευών ταυτόχρονα αυξάνει την εξειδίκευση της ανίχνευσης πτώσης κατά 5-15% για τους τέσσερις βασικούς αλγόριθμους που έχουν χρησιμοποιήσει, αλλά παράλληλα μειώνει κατά λίγο ποσοστό την ευαισθησία του.

- **Αναγνώριση πτώσης λαμβάνοντας υπόψη την υπολειπόμενη κίνηση (residual movement)**

Στη συγκεκριμένη εργασία [14] οι συγγραφείς έχουν χρησιμοποιήσει μόνο smartwatch για την ανίχνευση πτώσης. Συγκεκριμένα συλλέγουν δεδομένα από το επιταχυνσιόμετρο του smartwatch και χρησιμοποιούν φίλτρα για να εντοπίσουν κάποια πρότυπα (patterns) που υποδεικνύουν πτώση. Γίνεται χρήση τεχνικών κατωφλίου λαμβάνοντας υπόψη και την υπολειπόμενη κίνηση του χρήστη μετά την πτώση. Στη συνέχεια ο χρήστης μπορεί να επιλέξει τον τρόπο που προτιμά για να ειδοποιηθεί ο υπεύθυνος (κλήση, SMS, e-mail).

Στην αρχή, οι συγγραφείς μας εξηγούν ότι έχουν επιλέξει ένα χρονικό παράθυρο 6 δευτερολέπτων, μέσα στο οποίο μπορεί να εντοπιστεί ένα πρότυπο πτώσης. Αυτός ο αριθμός προέκυψε από δοκιμές, ώστε να περιλαμβάνει όλους τους πιθανούς τύπους πτώσεων. Στη συνέχεια μας εξηγούν πως ορίζουν ένα πρότυπο πτώσης. Για να εξεταστεί μία δραστηριότητα ως πτώση πρέπει: 1) Το SMV διάνυσμα επιτάχυνσης να ξεπερνάει το όριο των 18 m/sec^2 και 2) Μετά από ένα συγκεκριμένο χρονικό διάστημα το SMV να ξεπερνάει τα 7 m/sec^2 . Το χρονικό διάστημα αυτό ορίζεται ως υπολειπόμενη κίνηση από τους συγγραφείς και ανάλογα με το προφίλ του κάθε χρήστη μπορεί να πάρει τρεις τιμές (100, 300 ή 500ms). Αν οι 2 παραπάνω συνθήκες επαληθεύονται μέσα στο παράθυρο των 6 δευτερολέπτων, τότε υπάρχει υποψία πτώσης. Το τελευταίο βήμα για την ανίχνευση μιας πτώσης είναι ένας μετρητής που χρησιμοποιείται και αυξάνεται κάθε φορά που οι δύο συνθήκες ικανοποιούνται. Αν ο μετρητής είναι μεταξύ 1 και 14, τότε εντοπίζεται πτώση. Αν είναι μεγαλύτερος ή ίσος του 14, τότε οι συγγραφείς θεωρούν ότι είναι λόγω του ότι εκτελείται κάποια άλλη δραστηριότητα. Αν ο μετρητής είναι μικρότερος του 1, τότε θεωρείται ότι απλά έγινε κάποια απότομη κίνηση του καρπού και έτσι οι δύο συνθήκες δεν ικανοποιήθηκαν.

Για την αξιολόγηση του συστήματος τους πραγματοποίησαν μια σειρά πειραμάτων σε εθελοντές που πραγματοποιούσαν απλές δραστηριότητες καθημερινής ζωής (ADLs), καθώς και μια σειρά από πτώσεις σε στρώμα. Τα ποσοστά που πέτυχαν σε αληθώς θετικές ενδείξεις (ευαισθησία) πτώσεων ήταν 92.18% και το ποσοστό αληθώς αρνητικών ενδείξεων (εξειδίκευση) σε ADLs ήταν 87.29%.

2.3 Αλγόριθμοι και τεχνολογίες ανίχνευσης ζώνης εσωτερικού χώρου

Στα πλαίσια κατασκευής μιας έξυπνης εφαρμογής, αποφασίσαμε να βρούμε ένα τρόπο προσέγγισης της σχετικής θέσης του ηλικιωμένου στο χώρο του. Αυτό θα μας επιτρέψει να παραμετροποιήσουμε τις μεταβλητές κατωφλίων ανίχνευσης πτώσης, λαμβάνοντας υπόψιν την επικινδυνότητα της ζώνης στην οποία βρίσκεται ο ηλικιωμένος αλλά και αργότερα η θέση του ηλικιωμένου να μπορεί να επικοινωνηθεί στον υπεύθυνο σε περίπτωση πτώσης. Για τον λόγο αυτό ανατρέξαμε στη βιβλιογραφία για εντοπισμό αλγορίθμων και τεχνολογιών ανίχνευσης σχετικής θέσης σε εσωτερικό χώρο.

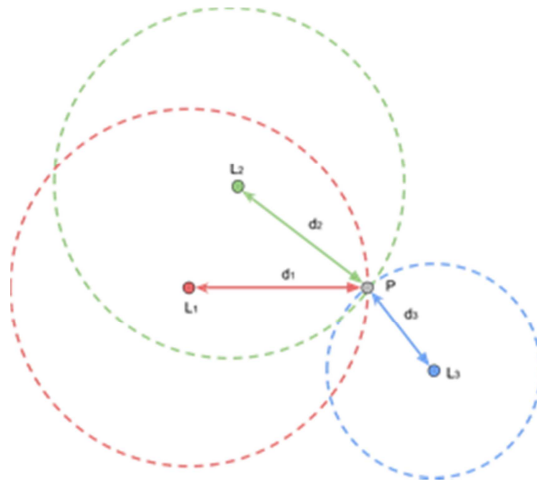
Τα συστήματα χωροθέτησης σε εξωτερικούς χώρους έχουν γνωρίσει μεγάλη ανάπτυξη τα τελευταία χρόνια και έχουν σχεδόν τελειοποιηθεί. Με τη χρήση του Global Positioning System (GPS) η ανίχνευση της τοποθεσίας του χρήστη έχει γίνει αρκετά εύκολη διαδικασία, με τη χρήση των δορυφόρων. Έτσι, συναντάμε πολύ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

Πάρα ταύτα, ενώ η χωροθέτηση σε εξωτερικούς χώρους είναι πλέον απλή διαδικασία, αποτελεί ακόμα και σήμερα πρόκληση να υλοποιηθούν αποτελεσματικά συστήματα για εσωτερική χωροθέτηση μέσα σε κτήρια. 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.

Οι τεχνικές εσωτερικής χωροθέτησης έχουν ως χαρακτηριστικά τις μετρικές, τις μεθόδους και τις τεχνολογίες που εκμεταλλεύονται.

Μια μέθοδος εσωτερικής χωροθέτησης είναι η χωροθέτηση με βάση την απόσταση (**lateration**) [Εικόνα 5]. Σύμφωνα με αυτή τη μέθοδο, αν είναι γνωστή η θέση τουλάχιστον τριών σημείων στο χώρο και η απόσταση τους από ένα τέταρτο σημείο στο χώρο, τότε μπορεί να υπολογιστεί η θέση αυτού του τέταρτου σημείου. Πολλές μετρικές προσπαθούν να υπολογίσουν προσεγγιστικά την απόσταση μεταξύ των σημείων αναφοράς και του άγνωστου (κινητού) σημείου. Κάποιες από αυτές περιλαμβάνουν:

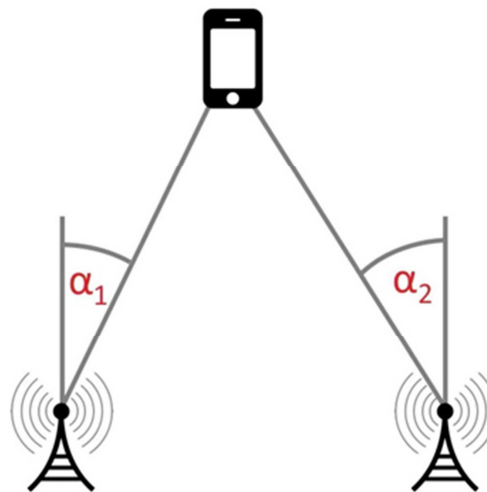
- Την **ισχύ του λαμβανόμενου σήματος**, σύμφωνα με την οποία, γνωρίζοντας την αρχική ισχύ εκπομπής του σήματος και τη τελική και χρησιμοποιώντας κατάλληλο αλγόριθμο υπολογισμού απώλειας για εσωτερικό χώρο, μπορούμε να έχουμε μια αρκετά καλή προσέγγιση για την απόσταση μεταξύ πομπού και δέκτη.
- Το **χρόνο άφιξης του σήματος**. Είναι φανερό πως η μέθοδος αυτή χρειάζεται τον ακριβή χρόνο εκπομπής και λήψης του σήματος. Έτσι είναι απαραίτητος ο χρονικός συγχρονισμός κάθε πομπού και κάθε δέκτη ώστε να αποφευχθεί η συμφόρηση. Κάτι τέτοιο είναι πολύ δύσκολο, ειδικά για μικρές αποστάσεις.



Εικόνα 5: Μέθοδος lateration

- Τη **διαφορά του χρόνου άφιξης σημάτων**, η οποία υπολογίζει τη διαφορά μεταξύ πολλαπλών σημάτων. Πιο συγκεκριμένα, διαφορετικοί πομποί, συγχρονισμένοι μεταξύ τους, εκπέμπουν ένα σήμα, και ο κινητός κόμβος τα συλλέγει και υπολογίζει τη συγκεκριμένη μετρική. Αντίστροφα, μπορεί ο κινητός κόμβος, να γίνει πομπός μεταδίδοντας ένα σήμα το οποίο λαμβάνεται από διαφορετικούς δέκτες (τους πομπούς του προηγούμενου παραδείγματος), οι οποίοι είναι πλέον υπεύθυνοι να υπολογίσουν τη διαφορά μεταξύ δυο σημάτων. Και στις δύο περιπτώσεις, απαιτείται συγχρονισμός μεταξύ των σταθερών κόμβων. Για τον υπολογισμό της θέσης του κινητού κόμβου απαιτείται η μέτρηση τουλάχιστον δύο διαφορών χρόνου άφιξης.
- Τη **διαφορά φάσης μεταξύ των σημάτων άφιξης**, σύμφωνα με την οποία, οι πομποί μεταδίδουν κύματα ίδιας συχνότητας την ίδια χρονική στιγμή, και ο δέκτης με βάση τη διαφορά φάσης των λαμβανόμενων σημάτων μπορεί να υπολογίσει την σχετική απόσταση που έχει από το κάθε πομπό, και τελικά τη θέση του στο χώρο.

Μια άλλη μέθοδος εσωτερικής χωροθέτησης είναι αυτή που βασίζεται στη γωνία (**angulation**) [Εικόνα 6]. Σύμφωνα με αυτή τη μέθοδο, γνωρίζοντας τη γωνία με την οποία έρχεται ένα σήμα σε τουλάχιστον δύο δέκτες, μπορεί να υπολογιστεί γεωμετρικά η θέση εκπομπής του. Ο μόνος τρόπος που μπορεί να υλοποιηθεί αυτή η μέθοδος είναι η ύπαρξη τουλάχιστον δύο συγχρονισμένων δεκτών στο χώρο. Απαιτείται να έχουν ειδικό εξοπλισμό που να μπορεί να μετράει τη **κλίση του λαμβανόμενου σήματος** (angle of arrival) από το πομπό, τη θέση του οποίου επιθυμούμε να υπολογίσουμε.



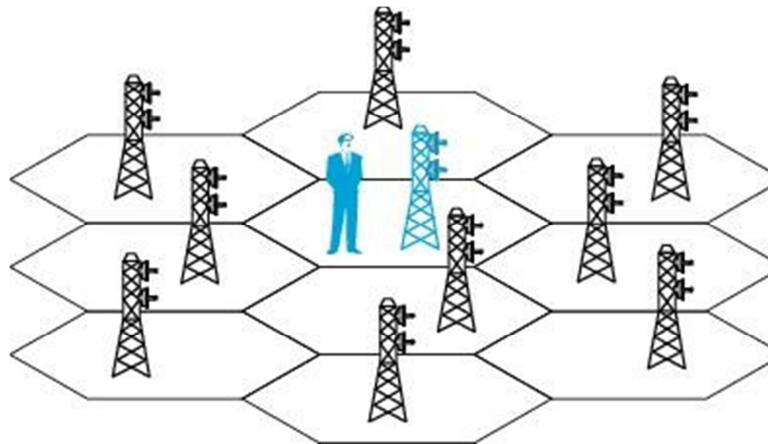
Εικόνα 6: Μέθοδος angulation

Εφαρμογές εσωτερικής χωροθέτησης κάνουν επίσης χρήση της **fingerprint** μεθόδου. Η fingerprint μέθοδος διακρίνεται από τρία κυρίως στάδια:

1. Το στάδιο συλλογής δεδομένων εκπαίδευσης, κατά το οποίο συλλέγονται χαρακτηριστικά δεδομένα από σημεία στο χώρο.
2. Προαιρετικά, το στάδιο της εκπαίδευσης ενός μοντέλου, όπου τα δεδομένα του προηγούμενου σταδίου, μαζί με κάποιον αλγόριθμο χρησιμοποιούνται για τη κατασκευή ενός μοντέλου.
3. Το στάδιο όπου το μοντέλο που έχει κατασκευαστεί ή ένας αλγόριθμος που συμπεριλαμβάνει τα δεδομένα του πρώτου σταδίου δίνει το αποτέλεσμα για τα άγνωστα δεδομένα.

Η μετρική που συνήθως συλλέγεται κατά το στάδιο συλλογής των δεδομένων, είναι η **λαμβανόμενη ισχύς σήματος**. Έτσι ο πίνακας με τη λαμβανόμενη ισχύ σημάτων από διαφορετικούς πομπούς για ένα σημείο, αποτελεί το αναγνωριστικό του σημείου αυτού. Αργότερα ο πίνακας με τη λαμβανόμενη ισχύ σημάτων του άγνωστου σημείου επιχειρείται να αντιστοιχηθεί με το πιο κοντινό γνωστό σημείο, είτε με την εφαρμογή του μοντέλου είτε με κάποιο αλγόριθμο συσταδοποίησης που δεν απαιτεί την κατασκευή μοντέλου αλλά εφαρμόζεται στα ανεπεξέργαστα δεδομένα. Το πρόβλημα ανάγεται σε πρόβλημα κατηγοριοποίησης αφού πλέον δεν ανιχνεύουμε ακριβή θέση. Σε κάποιες περιπτώσεις χρησιμοποιείται η πληροφορία της λαμβανόμενης ισχύος σήματος σε συνεργασία με κάποια άλλη μετρική, όπως το χρόνο άφιξης του σήματος [15] ή τη διαφορά του χρόνου άφιξης [16] για να βελτιώσει τα αποτελέσματα της fingerprint μεθόδου.

Οι αλγόριθμοι που μπορούν να χρησιμοποιηθούν για τη κατασκευή του μοντέλου ή για τη συσταδοποίηση των δεδομένων ποικίλουν. Περιλαμβάνουν κυρίως τον αλγόριθμο κοντινότερου γείτονα [17] και K-κοντινότερου γείτονα [18], τον Bayesian αλγόριθμο συγχώνευσης [19], τα νευρωνικά δίκτυα [15],[20], τις Μηχανές Διανυσμάτων Υποστήριξης (SVMs) [21] αλλά και πολλές παραλλαγές των παραπάνω.



Εικόνα 7: Μέθοδος εγγύτητας

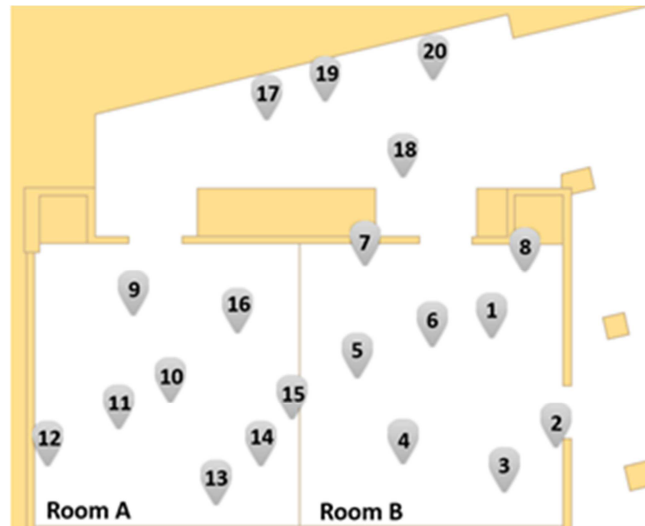
Μια ακόμα μέθοδος εσωτερικής χωροθέτησης είναι αυτή της **εγγύτητας** [Εικόνα 7]. Κι αυτή η μέθοδος, όπως άλλωστε και η fingerprint, παρέχει σχετική πληροφορία θέσης. Η μέθοδος της εγγύτητας χρησιμοποιεί την μετρική του κυττάρου προέλευσης. Συνήθως, βασίζεται σε ένα πυκνό πλέγμα κεραιών, καθένα από τα οποία έχει μια γνωστή θέση. Όταν ανιχνεύεται ένας κινητό από μια κεραία τότε η σχετική του θέση είναι η θέση της κεραίας. Όταν περισσότερες από μία κεραιές ανιχνεύουν τον κινητό, θεωρείται ότι σχετική του θέση είναι η θέση της κεραίας που λαμβάνει το ισχυρότερο σήμα. Η αποδοτικότητα αυτής της μεθόδου βασίζεται στον αριθμό των κεραιών που είναι διαθέσιμοι και ως εκ τούτου δεν αποτελεί ρεαλιστική λύση όταν ο χώρος στον οποίο εφαρμόζεται η τεχνική της εσωτερικής χωροθέτησης είναι πολύ μεγάλος.

Οι παραπάνω μέθοδοι εκμεταλλεύονται μετρικές από την εκπομπή και λήψη σημάτων για να προσεγγίσουν τη θέση του κινητού στο χώρο. Οι τεχνολογίες σημάτων χωρίζονται σε δυο βασικές κατηγορίες: τεχνολογίες που απαιτούν εξειδικευμένη υποδομή και τεχνολογίες που χρησιμοποιούνται ήδη στη καθημερινότητα μας και ως αποτέλεσμα δε απαιτούν επιπλέον κόστος για αγορά και συντήρηση επιπλέον συσκευών.

Οι προσεγγίσεις χωρίς υποδομή έχουν επικεντρωθεί κυρίως στη εκμετάλλευση –καναλιών Wi-Fi [22],[23], FM συχνοτήτων [24], του Παγκόσμιου Συστήματος Κινητών Επικοινωνιών (GSM) [25] και ηχητικών σημάτων [26]. Οι προσεγγίσεις που βασίζονται σε εξειδικευμένη υποδομή επικεντρώνονται στην ανάπτυξη προσαρμοσμένων συσκευών, όπως ραδιοσυχνοτήτων (RFID) [27], υπέρυθρων [28], υπέρηχων [29], Bluetooth [30], LED φωτισμού [31].

Παρόλο που έχουν προταθεί εκατοντάδες διαφορετικοί συνδυασμοί μεθόδων, τεχνικών και τεχνολογιών στη βιβλιογραφία, το πρόβλημα της εσωτερικής θέσης παραμένει άλυτο. Η ερευνητική κοινότητα δεν έχει συγκλίνει σε μια ενιαία, ευρέως αποδεκτή λύση που μπορεί να επιτύχει την επιθυμητή ακρίβεια με το απαιτούμενο κόστος.

Σύμφωνα με τους συγγραφείς του [32] αυτό οφείλεται στο γεγονός πως η αξιολόγηση των συστημάτων εσωτερικής χωροθέτησης δεν είναι ομοιόμορφη αλλά αξιολογούνται σε ένα εξαιρετικά ελεγχόμενο περιβάλλον κάθε φορά. Αυτό έχει σαν αποτέλεσμα να μην μπορούν να συγκριθούν τα διαφορετικά συστήματα μεταξύ τους.



Εικόνα 8: Ο χώρος διεξαγωγής του πειράματος του [18]

Για να μπορέσουν να συγκρίνουν με αντικειμενικά κριτήρια διαφορετικές εφαρμογές εσωτερικής χωροθέτησης διοργάνωσαν ένα διήμερο διαγωνισμό στον οποίο 22 συστήματα εφάρμοσαν τις τεχνικές τους στο ίδιο, άγνωστο για όλα τα συστήματα, χώρο. Την πρώτη μέρα του διαγωνισμού, οι ομάδες κλήθηκαν να εγκαταστήσουν τα συστήματά τους στο χώρο να πάρουν μετρήσεις αν τις χρειάζονται και πιθανώς να φτιάξουν κάποια μοντέλα, αν αυτό ήταν αναγκαίο για το σύστημά τους. Τη δεύτερη μέρα τα συστήματα αξιολογήθηκαν σε 20 σημεία στο χώρο. Ο χώρος του πειράματος και τα 20 άγνωστα για τα συστήματα σημεία φαίνονται στη Εικόνα 8.

Ένα από τα σημαντικά συμπεράσματα αυτής της έρευνας είναι ότι επιβεβαίωσε την ύπαρξη «εύκολων» και «δύσκολων» σημείων, σημείων δηλαδή που τα περισσότερα συστήματα, ακόμα και τα αποδοτικότερα, πέτυχαν (πχ. σημεία 6,9,10) ή απέτυχαν (πχ. σημεία 1,2,7) να υπολογίσουν σωστά. Αυτό δείχνει ότι η επιλογή των σημείων δοκιμής είναι ζωτικής σημασίας και αποκαλύπτει τη δυσκολία της αντικειμενική αξιολόγησης συστημάτων εσωτερικής θέσης. Αξίζει να αναφερθεί, επίσης, πως το καλύτερο σύστημα εσωτερικής χωροθέτησης βασισμένο σε εξειδικευμένη υποδομή (τεχνολογία RFID 2.4GHz, μέθοδος διαφοράς φάσης [33]) πέτυχε το μισό λάθος από το καλύτερο σύστημα εσωτερικής χωροθέτησης βασισμένο σε καθημερινή τεχνολογία (τεχνολογία WiFi, μέθοδος Fingerprinting, Bayesian filtering αλγόριθμος [34]).

3. ΠΕΡΙΓΡΑΦΗ ΣΥΣΤΗΜΑΤΟΣ

Στα πλαίσια της παρούσας διπλωματικής αναπτύξαμε μια εφαρμογή Android καθώς και έναν εξυπηρετητή.

Η εφαρμογή Android μπορεί να εκτελεστεί σε κινητές (smartphones) ή φορετές (wearables) συσκευές. Χωρίζεται σε δύο τμήματα: την εφαρμογή που βλέπει ο υπεύθυνος και την εφαρμογή που βλέπει ο ηλικιωμένος. Αυτές μοιράζονται το ίδιο εκτελέσιμο .apk και ο διαχωρισμός τους γίνεται με βάση το διαφορετικό ρόλο χρήστη. Στην εφαρμογή του ηλικιωμένου γίνεται μόνο η συλλογή δεδομένων από τους αισθητήρες της συσκευής και η αποστολή τους στον εξυπηρετητή. Στην εφαρμογή του υπεύθυνου μπορούν να γίνουν διάφορες ρυθμίσεις, όπως το πότε θα στέλνονται δεδομένα από την εφαρμογή του ηλικιωμένου, ή η ρύθμιση των ζωνών στο χώρο του ηλικιωμένου.

Επιπροσθέτως, για την επικοινωνία μεταξύ των δύο τύπων εφαρμογών, ήταν αναγκαία η ανάπτυξη ενός εξυπηρετητή. Σε αυτόν υπάρχει η λογική του αλγορίθμου ανίχνευσης πτώσης και του αλγορίθμου ανίχνευσης ζώνης εσωτερικού χώρου. Αποφασίσαμε η λογική των αλγορίθμων να βρίσκεται στη πλευρά του εξυπηρετητή για να πετύχουμε μεγαλύτερη αυτονομία στην εφαρμογή Android. Για την επικοινωνία μεταξύ των δύο υπο-εφαρμογών Android, γίνεται χρήση μιας βάσης δεδομένων και ενός μηχανισμού αποστολής ειδοποιήσεων.

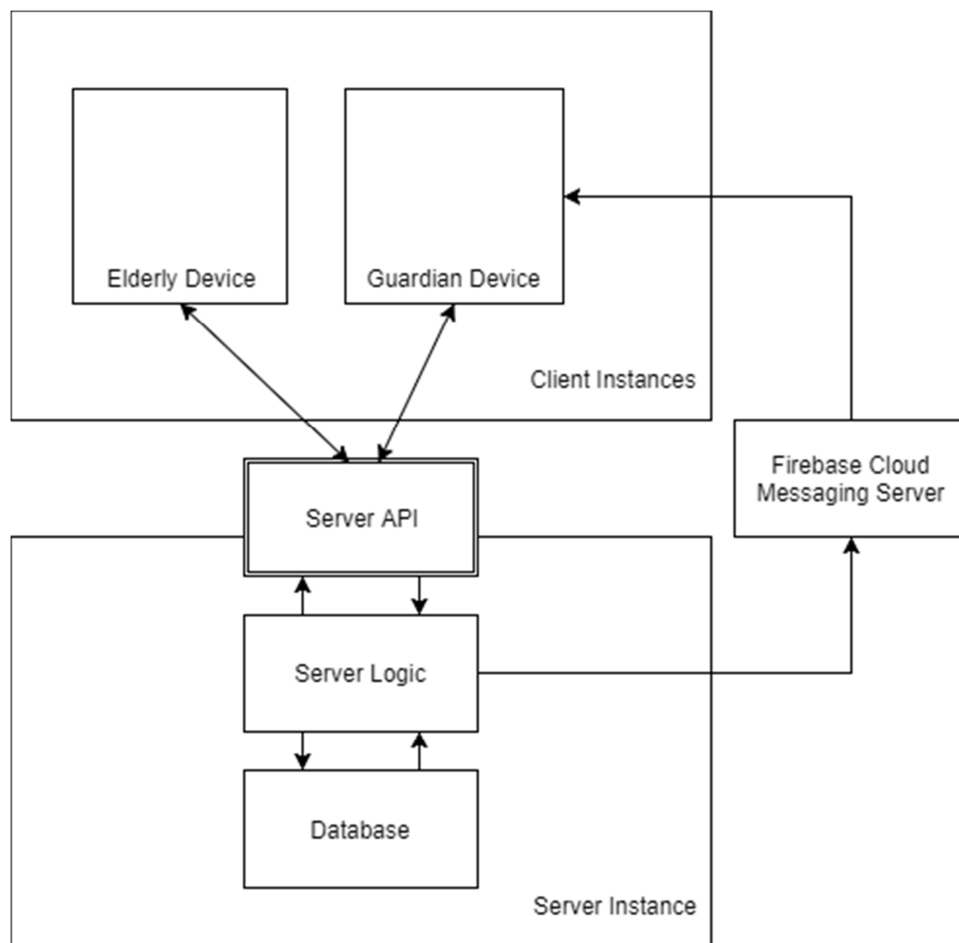
Βασισμένοι στην ανάγκη ανάπτυξης ενός συστήματος επεκτάσιμου και αποδοτικού, αποφασίσαμε να αναπτύξουμε ένα σύστημα σε τρία διαφορετικά λογικά επίπεδα (3-layer architecture):

- την επιχειρησιακή λογική στο server
- την βάση δεδομένων
- την εφαρμογή πελάτη

Για τον server χρησιμοποιήσαμε Tomcat 7 μέσω του Spring Boot, η βάση δεδομένων μας είναι MySQL 5.7 ενώ για την εφαρμογή πελάτη, native Android SdkVersion 25. Για την επικοινωνία μεταξύ πελάτη και Server αναπτύξαμε REST υπηρεσίες ιστού, τις οποίες καταναλώνουν οι εφαρμογές πελάτη με τη βοήθεια της βιβλιοθήκης Retrofit. Η ανάγκη αποστολής ειδοποιήσεων στη συσκευή του υπεύθυνου για κάποια πιθανή πτώση του ηλικιωμένου λύθηκε με τη χρήση του Firebase Cloud Messaging.

Το Σχήμα 1 παρουσιάζει τη γενική αρχιτεκτονική του συστήματος μας.

Στα παρακάτω υποκεφάλαια θα αναλύσουμε με περισσότερη λεπτομέρεια τους αλγόριθμους ανίχνευσης πτώσης και ζώνης εσωτερικού χώρου που αναπτύξαμε, τις τεχνολογίες που χρησιμοποιήσαμε στη μεριά του server και την Android εφαρμογή (client) τόσο στην μεριά του ηλικιωμένου όσο και στη μεριά του υπεύθυνου χρήστη.



Σχήμα 1: Γενική αρχιτεκτονική συστήματος

3.1. Περιγραφή Αλγορίθμων

Στις παρακάτω ενότητες θα περιγράψουμε τους αλγορίθμους που εφαρμόσαμε τόσο στην ανίχνευση πτώσης όσο και στην ανίχνευση ζώνης εσωτερικού χώρου, καθώς και το πως συνδυάζονται τα δύο αυτά συστατικά για να επιτευχθεί το καλύτερο δυνατό αποτέλεσμα.

3.1.1 Αλγόριθμος ανίχνευσης πτώσης

Για τον αλγόριθμο ανίχνευσης πτώσης έχουν χρησιμοποιηθεί βασικές μεθοδολογίες από τη βιβλιογραφία, που περιγράψαμε στην Παράγραφο 2.1, με την προσθήκη δικών μας ιδεών, ώστε να ενισχυθεί ακόμα περισσότερο το αποτέλεσμα. Συγκεκριμένα χρησιμοποιούμε την τεχνική κατωφλίου για να ξεχωρίσουμε τις τρεις βασικές φάσεις (ελεύθερη πτώση, πρόσκρουση, μετά-πτώση) με βάση συγκεκριμένα όρια στην τιμή της επιτάχυνσης που λαμβάνουμε από τις μετρήσεις του επιταχυνσιόμετρου. Η συνεισφορά του αλγόριθμου μας έγκειται στη χρησιμοποίηση της τρέχουσας ζώνης για την μεταβολή

των ορίων, καθώς και στην αποστολή περισσότερης πληροφορίας στον υπεύθυνο του ηλικιωμένου, όπως η ζώνη που πιστεύουμε ότι έγινε η πτώση, ο τρόπος που πιστεύουμε ότι έγινε (από δραστηριότητα ή πτώση από κρεβάτι/καρέκλα), καθώς και το αν έχει ή όχι τις αισθήσεις του ο ηλικιωμένος.

Ο συγκεκριμένος αλγόριθμος “τρέχει” στην πλευρά του server κάθε ένα λεπτό, και χρησιμοποιεί σαν είσοδο τα δεδομένα του επιταχυνσιόμετρου, όπως απεικονίζονται στον πίνακα `accelerometer_stats` της βάσης μας. Σε κάθε εκτέλεση χρησιμοποιούνται τα δεδομένα των `LAST_X_MINUTES` (στην περίπτωση μας `LAST_X_MINUTES=2`) λεπτών, και έτσι ο υπεύθυνος χρήστης θα λάβει την ειδοποίηση δύο φορές σαν υπενθύμιση. Αρχικά, γίνεται μια κλήση ώστε να βρούμε την τελευταία ζώνη που έχει καταγραφεί στον πίνακα `user_in_zone`, ώστε να δούμε αν ο χρήστης βρίσκεται σε επικίνδυνη, ασφαλή ή “ουδέτερη” ζώνη. Με βάση το αποτέλεσμα αυτής της αναζήτησης, αυξάνουμε ή μειώνουμε την “ευαισθησία” του αλγορίθμου σε πτώσεις (θα περιγραφεί παρακάτω αναλυτικότερα). Επίσης πριν ξεκινήσει να τρέχει ο αλγόριθμος, διαγράφουμε από τον πίνακα `accelerometer_stats` όλα τα παλιά δεδομένα, που έχουν χρησιμοποιηθεί ήδη για ανίχνευση πτώσης και δεν χρειαζόμαστε πια.

Σε κάθε εκτέλεση του αλγορίθμου, αρχίζουμε να ψάχνουμε τα δεδομένα μας, υπολογίζοντας κάθε φορά το διάνυσμα επιτάχυνσης $A = \sqrt{Ax^2 + Ay^2 + Az^2}$. Αν σε οποιοδήποτε σημείο, βρούμε το A να είναι μικρότερο από μια σταθερά `LOWER_THRESHOLD` ($0,5 \cdot g$ στην περίπτωση μας) τότε θεωρούμε ότι έχουμε εντοπίσει την πρώτη φάση της πτώσης (φάση ελεύθερης πτώσης όπως έχουμε περιγράψει παραπάνω), κρατάμε την τιμή του A (A_{fall}) και συνεχίζουμε ψάχνοντας την φάση πρόσκρουσης. Διαφορετικά αν δεν βρούμε πουθενά τέτοια τιμή, θεωρούμε ότι δεν υπάρχει καμία πτώση σε αυτό το διάστημα που ψάχνουμε (περίπτωση 0).

Αφού λοιπόν έχουμε βρει μια τιμή μικρότερη του `LOWER_THRESHOLD`, κρατάμε την τιμή αυτή και ψάχνουμε στα δεδομένα του επιταχυνσιόμετρου για τα επόμενα `MAX_TIME_AFTER_FREE_FALL` δευτερόλεπτα (στην περίπτωση μας `MAX_TIME_AFTER_FREE_FALL = 3`). Αν λοιπόν για τις επόμενες τιμές που παίρνουμε από τον πίνακα `accelerometer_stats`, βρούμε μέσα στα 3 επόμενα δευτερόλεπτα μια τιμή για το A (A_{impact}), της οποίας η διαφορά από το A_{fall} είναι μεγαλύτερη μίας μέγιστης διαφοράς (`MAX_DIFFERENCE`), τότε θεωρούμε ότι έχουμε σε αυτό το σημείο την φάση πρόσκρουσης. Όπως περιγράψαμε παραπάνω, ανάλογα με τη ζώνη που βρίσκεται ο χρήστης, η τιμή του `MAX_DIFFERENCE` προσαρμόζεται ανάλογα έτσι ώστε να είναι λιγότερο ή περισσότερο ευαίσθητος ο αλγόριθμος σε πτώσεις. Στην περίπτωση μας, η προκαθορισμένη (default) τιμή για το `MAX_DIFFERENCE` είναι $2,5g$. Αυτή η τιμή παραμένει έτσι όταν ο χρήστης δεν βρίσκεται ούτε σε επικίνδυνη ούτε σε ασφαλή ζώνη. Όταν ο χρήστης βρίσκεται σε επικίνδυνη ζώνη, η τιμή μειώνεται κατά $0,2g$, οπότε πέφτει στα $2,3g$ και ο αλγόριθμος βρίσκει έτσι πιο εύκολα πτώσεις. Είναι πιθανό με αυτή τη ρύθμιση να αυξηθούν τα false positives, αλλά αυτό είναι σκόπιμο εφόσον η ζώνη είναι επικίνδυνη και δεν θέλουμε να ρισκάρουμε να χάσουμε κάποια πτώση. Αντίστοιχα όταν ο χρήστης είναι σε ασφαλή ζώνη, η τιμή αυξάνεται κατά $0,2g$ και πάει έτσι στα $2,7g$, οπότε και ο αλγόριθμος βρίσκει πιο δύσκολα πτώσεις. Όταν λοιπόν με αυτές τις ρυθμίσεις βρούμε πρόσκρουση (περίπτωση 1), κρατάμε και το χρόνο (timestamp) στον οποίο συνέβη η πιθανή πτώση.

Στη συνέχεια, για να σιγουρευτούμε ότι έχουμε πτώση, ψάχνουμε την φάση μετά-πτώσης. Για να το κάνουμε αυτό, ελέγχουμε την τιμή του διανύσματος επιτάχυνσης για τα επόμενα `MAX_TIME_AFTER_IMPACT` δευτερόλεπτα (5 δευτερόλεπτα στην περίπτωση μας). Για να

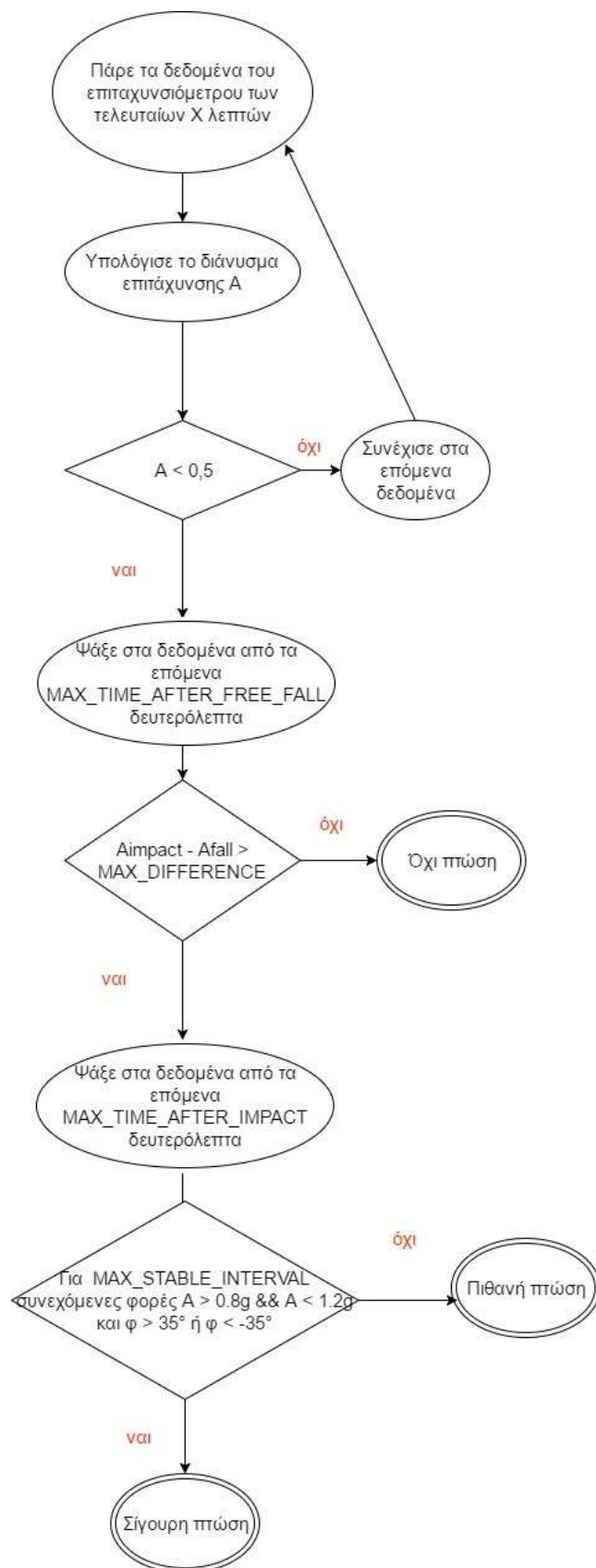
έχουμε πτώση θεωρούμε ότι πρέπει για MAX_STABLE_INTERVAL (5 στην περίπτωση μας) συνεχόμενες τιμές του πίνακα να ισχύουν οι παρακάτω δύο συνθήκες:

- Η τιμή του διανύσματος επιτάχυνσης να είναι μεταξύ 0,8g και 1,2g, που σημαίνει ότι ο χρήστης είναι (σχεδόν) ακίνητος.
- Η γωνία φ , όπου $\cos\varphi = \frac{Ay}{Ax^2 + Ay^2 + Az^2}$ και y ο άξονας που δείχνει προς το έδαφος να είναι $\varphi < -35^\circ$ ή $\varphi > 35^\circ$. Αυτό προκύπτει αφού όπως είπαμε παραπάνω στην ενότητα 2.1.1 θεωρούμε ότι κάποιος είναι σε όρθια θέση όταν η γωνία φ με το έδαφος είναι μεταξύ 35 και -35 μοιρών.

Όταν και οι δύο παραπάνω συνθήκες ικανοποιούνται για παραπάνω από MAX_STABLE_INTERVAL συνεχόμενες φορές, τότε θεωρούμε ότι έχουμε σίγουρη πτώση (περίπτωση 2).

Άρα, στο τέλος μίας εκτέλεσης του αλγορίθμου μπορούμε να έχουμε τρεις περιπτώσεις, τις οποίες ξεχωρίζουμε θέτοντας την αντίστοιχη τιμή κάθε περίπτωσης σε μία μεταβλητή:

- Περίπτωση 0, όπου σίγουρα δεν έχουμε πτώση (no fall).
- Περίπτωση 1, όπου έχουμε πρόσκρουση ή πολύ μεγάλη απότομη επιτάχυνση για οποιοδήποτε λόγο (possible fall).
- Περίπτωση 2, όπου ισχύουν όλα όσα ισχύουν και στην περίπτωση 1, αλλά έχουμε και φάση μετά-πτώσης (sure fall).



Σχήμα 2: Διάγραμμα ροής αλγορίθμου ανίχνευσης πτώσης

Στη συνέχεια με βάση το αποτέλεσμα του αλγόριθμου, στέλνουμε στον υπεύθυνο του ηλικιωμένου την αντίστοιχη ειδοποίηση (notification), όπου περιέχει πληροφορίες για το πόσο πιθανό είναι να έχουμε όντως πτώση, σε ποια ζώνη εντοπίστηκε η πτώση και πως ακριβώς συνέβη. Αυτό γίνεται ως εξής:

- Ξεκινάμε μετά από καθορισμένο χρόνο από την εκτέλεση του αλγορίθμου (πχ 1 λεπτό) ένα καινούριο Thread. Με αυτόν τον τρόπο, υπάρχει μεν μια μικρή καθυστέρηση στην ειδοποίηση του υπεύθυνου, αλλά δίνεται μια πιο ακριβής εικόνα της πτώσης.
- Βρίσκουμε την τελευταία ζώνη που έχει υπολογιστεί μετά το timestamp πρόσκρουσης. Η ζώνη αυτή θα σταλεί στον υπεύθυνο χρήστη μέσω του notification.
- Βρίσκουμε την τελευταία δραστηριότητα πριν και την πρώτη μετά την πρόσκρουση από τη βάση μέσω του Google Activity Recognition API (βλέπε 3.3.3). Με βάση αυτές τις δύο τιμές, φτιάχνουμε ανάλογα το μήνυμα που θα σταλεί στον υπεύθυνο με βάση τον παρακάτω πίνακα:

Πίνακας 2: Είδη πτώσεων ανάλογα με την αντίστοιχη μεταβολή της κατάστασης του χρήστη.

		Μετάβαση καταστάσεων			
		όχι ακίνητος -> ακίνητος	ακίνητος -> ακίνητος	όχι ακίνητος -> όχι ακίνητος	ακίνητος -> όχι ακίνητος
Είδη πτώσεων	χωρίς αισθήσεις	X	X		
	με αισθήσεις			X	X
	από καρέκλα/κρεβάτι		X		X
	από δραστηριότητα	X		X	

Έτσι στο τελικό notification που στέλνεται στον υπεύθυνο αναφέρεται η σιγουριά της πτώσης (sure/possible fall), η ζώνη στην οποία έγινε η πτώση, ο πιθανός τρόπος πτώσης του ηλικιωμένου (από καρέκλα/κρεβάτι ή από δραστηριότητα), καθώς και το αν θεωρούμε πως έχει τις αισθήσεις του ή όχι.

3.1.2 Αλγόριθμος ανίχνευσης ζώνης εσωτερικού χώρου

Όπως αναφέραμε σε προηγούμενο κεφάλαιο, ερευνήσαμε τρόπους προσδιορισμού της σχετικής θέσης του ηλικιωμένου στο χώρο ώστε να μπορούμε να προσδιορίσουμε εάν αυτός είναι σε περιοχή υψηλής (όπως μπάνιο ή χολ) ή χαμηλής επικινδυνότητας (όπως καθιστικό).

Αυτή την πληροφορία μπορούμε να χρησιμοποιήσουμε αργότερα για να στείλουμε μια πιο περιγραφική ειδοποίηση στον υπεύθυνο, όπου θα τον ενημερώνουμε για την πιθανή ζώνη πτώσης, αλλά και στο ίδιο τον αλγόριθμο της αναγνώρισης πτώσης, με σκοπό να κάνουμε τον αλγόριθμο πιο ευαίσθητο στα σημεία υψηλής επικινδυνότητας. Ο λόγος αυτής της απόφασης είναι πως προτιμούμε να αναγνωρίσει ο αλγόριθμος κάποια false positives, παρά να αγνοήσει κάποιες ενδείξεις του επιταχυνσιόμετρου, που είναι πολύ κοντά στα όρια, ειδικά στις επικίνδυνες ζώνες.

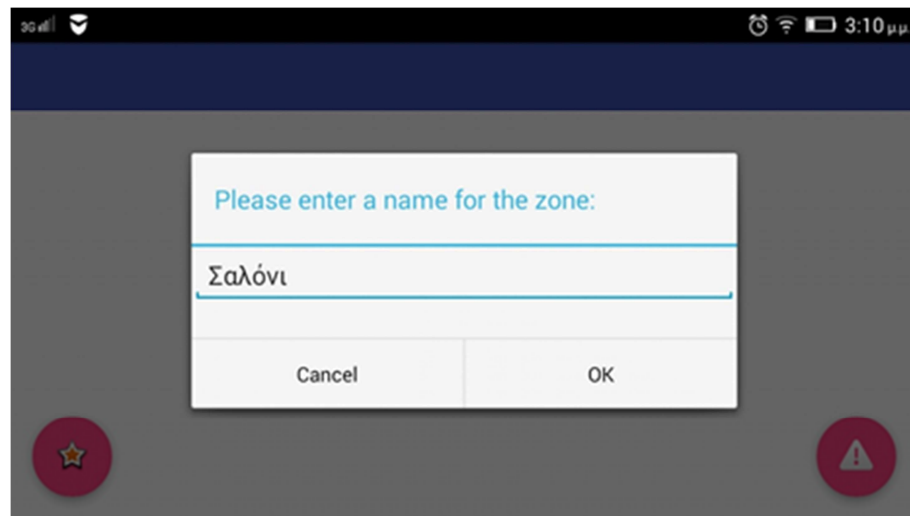
Για την ανίχνευση ζώνης εσωτερικού χώρου εφαρμόσαμε μια παραλλαγή του fingerprint αλγόριθμου με βάση την λαμβανόμενη ισχύ Wi-Fi σήματος. Οι λόγοι που χρησιμοποιήσαμε

αυτή τη τεχνική σε αντίθεση με τις εναλλακτικές επιλογές που παρουσιάσαμε στο κεφάλαιο 2.3 είναι γιατί η χρήση αυτής της τεχνικής έχει παρουσιάσει καλύτερα αποτελέσματα (συγκριτικά με τεχνικές που κάνουν χρήση της γωνίας ή του χρόνου άφιξης του σήματος), αλλά και γιατί είχαμε έτοιμη την υποδομή (βάση και επικοινωνία με αυτή) και τις αναγκαίες τεχνολογίες από την υλοποίηση του αλγόριθμου ανίχνευσης πτώσης.

Οι αλγόριθμοι fingerprint, όπως αναφέραμε σε προηγούμενο κεφάλαιο διακρίνονται από δύο φάσεις:

- 1) Την offline φάση που περιλαμβάνει:
 - a) Τη φάση συλλογής δεδομένων μέτρησης της ισχύος των σημάτων, όπου χρησιμοποιείται η fingerprint τεχνική για τον υπολογισμό και την αποθήκευση γνωστών θέσεων στο χώρο ώστε αργότερα να χρησιμοποιηθούν από το σύστημα για την κατασκευή ενός εμπειρικού μοντέλου ή μοντέλου ακρίβειας.
 - b) Προαιρετικά, τη φάση φιλτραρίσματος των δεδομένων που συλλέχθηκαν στο στάδιο συλλογής, για την μείωση του θορύβου στα δεδομένα ώστε το μοντέλο να έχει καλύτερη απόδοση.
- 2) Την online φάση, στην οποία τα δεδομένα μέτρησης της ισχύος των σημάτων σε άγνωστα σημεία συγκρίνονται με τα δεδομένα που έχουν συλλεχθεί κατά τη διάρκεια της offline φάσης με σκοπό την ανίχνευση της ζώνης στην οποία βρίσκεται η συσκευή.

Κατά την *offline* διαδικασία, καλείται ο υπεύθυνος να μαρκάρει τις περιοχές χαμηλής και υψηλής επικινδυνότητας, χρησιμοποιώντας το κινητό του στο χώρο του ηλικιωμένου. Ο υπεύθυνος επιλέγει αρχικά το κουμπί εισαγωγής επικίνδυνης ή ακίνδυνης ζώνης, και στην συνέχεια ένα χαρακτηριστικό όνομα για αυτή την περιοχή (όπως «σαλόνι», «τουαλέτα») [Εικόνα 9]. Στη συνέχεια το κινητό καλείται για SCAN_TIMES φορές, 10 στη περίπτωση μας, να πάρει μετρήσεις από το σημείο με την ισχύ των γειτονικών σημάτων. Αυτή τη λίστα την προωθεί στον server. Ο χρήστης καλείται να μένει όσο το δυνατό ακίνητος, ή να αφήσει το κινητό σε ένα σταθερό σημείο κατά τη διάρκεια των μετρήσεων, ώστε να μην επηρεαστούν οι μετρήσεις από τυχαίες κινήσεις του. Όταν τελειώσει με την επισήμανση όλων των ζωνών επικινδυνότητας, θεωρούμε ότι ολοκληρώνεται η offline διαδικασία.



Εικόνα 9: Ο υπεύθυνος εισάγει μια περιοχή κατά την offline διαδικασία

Η φάση *φιλτραρίσματος*, στη περίπτωση μας, λαμβάνει χώρα τη στιγμή που τα δεδομένα φτάνουν στον server και πριν από την αποθήκευση τους στη βάση. Για την ακρίβεια, για κάθε λίστα διαδοχικών μετρήσεων της ισχύος των σημάτων σε ένα σημείο:

1. Υπολογίζεται η μέση τιμή της ισχύος για κάθε σήμα ως η πιο αντιπροσωπευτική μέτρηση.
2. Φιλτράρονται μέσες τιμές έτσι ώστε $\text{Signalx} > -80\text{dB}$. Ο λόγος που φιλτράρουμε τέτοιες μετρήσεις είναι γιατί Wi-Fi πομποί που είναι σαφώς μακριά από το σημείο δεν αποτελούν τους βασικούς αντιπροσώπους του σημείου και μπορούν μάλιστα να επηρεάσουν την ανίχνευση άγνωστης ζώνης αρνητικά με το να εξαφανίζονται σε κάποιες μετρήσεις από τη λίστα (από τη στιγμή που είναι στο όριο εύρους).

Το αποτέλεσμα του φιλτραρίσματος που περιγράφηκε παραπάνω αποθηκεύεται στη βάση και συγκεκριμένα στον πίνακα `wifi_in_zone`.

Κατά την online φάση, έχουμε πλέον το κινητό του ηλικιωμένου και θέλουμε να ερευνήσουμε τη ζώνη στην οποία βρίσκεται κάθε φορά. Ο αλγόριθμος αυτός ξεκινάει παράλληλα με τον αλγόριθμο ανίχνευσης πτώσης και ολοκληρώνεται όταν ο υπεύθυνος ζητήσει να ολοκληρωθεί η περίοδος ανίχνευσης.

Σε κάθε εκτέλεση του αλγορίθμου έχουμε μια σειρά από βήματα:

- Συλλέγεται η ισχύς των σημάτων στο σημείο που βρίσκεται ο ηλικιωμένος, με τον ίδιο τρόπο που περιγράφηκε και για την offline φάση και στέλνονται στο server για επεξεργασία.
- Ο server υπολογίζει τους μέσους όρους των σημάτων για το συγκεκριμένο σημείο και φιλτράρει τη πληροφορία με ανάλογο με την offline φάση τρόπο, ώστε να κρατήσει μόνο τα αντιπροσωπευτικά για τη συγκεκριμένη ζώνη δεδομένα.
- Για κάθε μέτρηση ισχύος πομπού, αναζητούνται οι κοντινότερες τιμές ισχύος στη βάση για αυτό το Wi-Fi πομπό
 - ο Αν δεν υπάρχουν τιμές στη βάση που να είναι σχετικά κοντινές

(NEIGHBOR_THR = 5dB), τότε η τιμή του συγκεκριμένου πομπού δίνει πόντο σε μια άγνωστη ζώνη ("unknown") που δεν έχει επισημανθεί από τον υπεύθυνο.

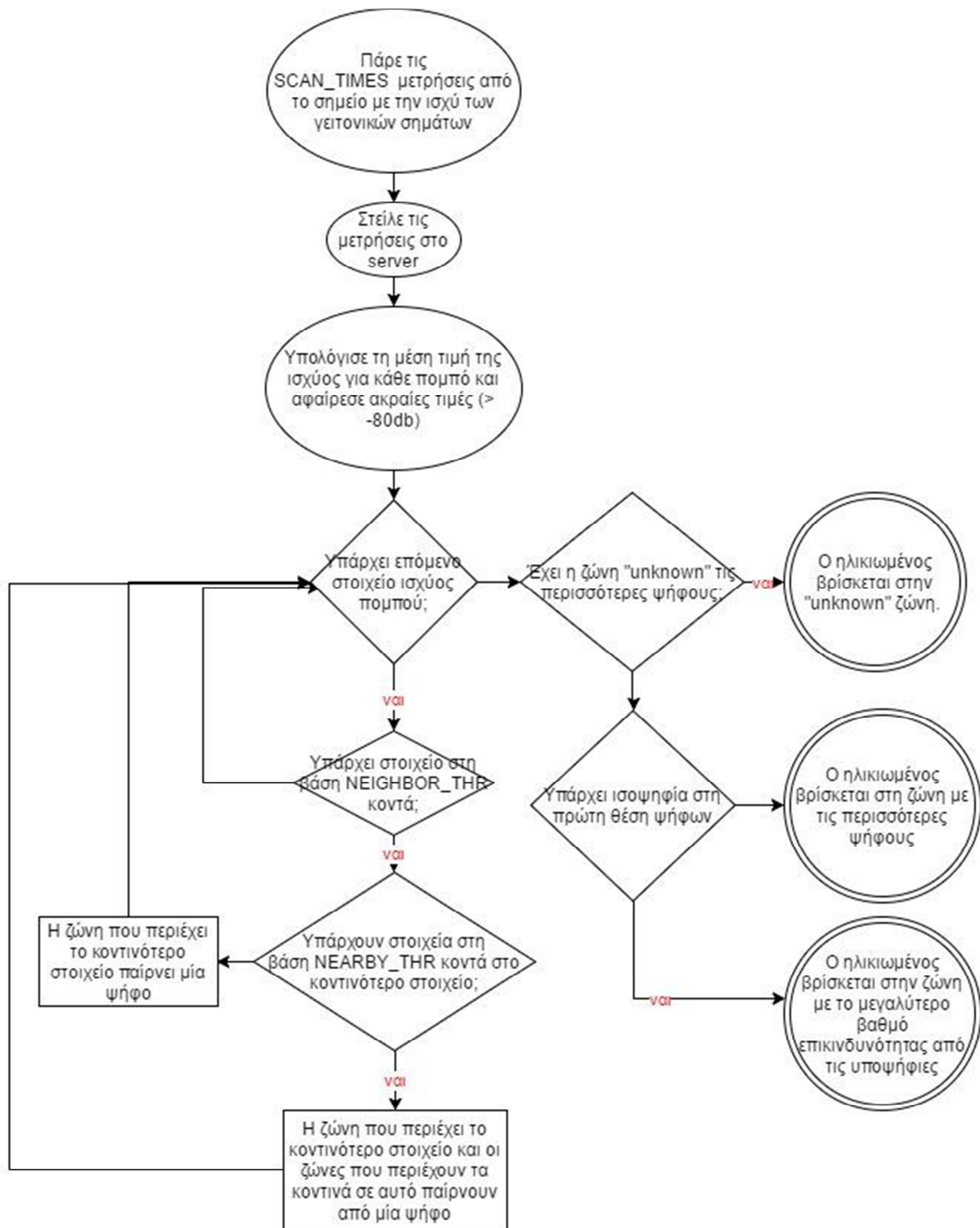
- ο Αν υπάρχει μία τιμή που να είναι σχετικά κοντινή στη βάση και είναι μοναδική τότε παίρνει έναν πόντο.
- ο Αν υπάρχουν παραπάνω από μια τιμές που να είναι σχετικά κοντινές, τότε παίρνουν πόντους εκτός από την απόλυτα κοντινή και αυτές που είναι πολύ κοντά σε αυτή (NEARBY_THR = 1.5dB).
- Τελικά, η ζώνη που έχει συγκεντρώσει τους περισσότερους πόντους θεωρούμε πως είναι αυτή στην οποία βρίσκεται ο ηλικιωμένος. Αν αυτή η ζώνη έχει χαρακτηριστεί ως ζώνη υψηλής επικινδυνότητας, τότε κατά την ανίχνευση πτώσης θα ληφθεί υπόψιν. Αν η άγνωστη ζώνη έχει τους περισσότερες πόντους, τότε ο ηλικιωμένος είναι έξω από τις ζώνες της offline φάσης και δεν επηρεάζεται η ευαισθησία του αλγόριθμου πτώσης. Αν δύο ή παραπάνω ζώνες έρθουν ισόπαλες ως υποψήφιος από την παραπάνω διαδικασία, τότε επιλέγουμε τη ζώνη υψηλότερης επικινδυνότητας από αυτές ως υποψήφια θέση του ηλικιωμένου. Η διαδικασία της online φάσης, παρουσιάζεται και στο Σχήμα 3.

Η πληροφορία που εξάγουμε από τον παραπάνω αλγόριθμο αποθηκεύεται στη βάση, στον πίνακα `user_in_zone` για μελλοντική χρήση.

Αυτή η παραλλαγή των k-κοντινότερων γειτόνων (k-NN) μας επιτρέπει να δίνουμε βάρη στους απόλυτα κοντινούς γείτονες, αλλά να μην αγνοούμε γείτονες που είναι πολύ κοντά στον επικρατέστερο, ακόμα και αν ξεπερνούν τη μεταβλητή K. Επίσης, μας επιτρέπει να θέσουμε ένα όριο ώστε να διαχωρίζουμε και τη περίπτωση όπου ο ηλικιωμένος δε βρίσκεται σε καμία από τις θέσεις που έχει επισημάνει ο υπεύθυνος κατά την offline φάση.

Παρά τα πολλά πλεονεκτήματα αυτής της μεθόδου, όπως είναι το χαμηλό κόστος και η αναφερόμενη υψηλή ακρίβεια, υπάρχουν και κάποιοι περιορισμοί άξιοι αναφοράς.

Αρχικά, οι αλγόριθμοι Wi-Fi Fingerprint απαιτούν την ύπαρξη ακριβών υποστηρικτικών αλγόριθμων, όπως άλλωστε και αυτός που υλοποιήσαμε εμείς. Για να αντιμετωπίσουμε αυτό το περιορισμό, επιλέξαμε το ακριβό κομμάτι του αλγορίθμου να γίνεται στη πλευρά του server, έτσι ώστε να μην επιβαρύνεται η κινητή συσκευή και να μην επηρεάζεται η αυτονομία της τελευταίας.



Σχήμα 3: Διάγραμμα ροής online διαδικασίας της ανίχνευσης ζώνης εσωτερικού χώρου

Ένας άλλος περιορισμός, είναι πως ο υπεύθυνος και ο ηλικιωμένος δεν έχουν υπό τον έλεγχο τους τους διαφορετικούς Wi-Fi πομπούς. Έτσι, μπορεί σε βάθος χρόνου, αυτοί οι

πομποί να αλλάζουν θέση, να κλείσουν, ή γενικά να υπάρξουν περιβαλλοντικές αλλαγές που θα επηρεάσουν την λαμβανόμενη ισχύ των πομπών σε κάποιο σημείο. Για να προσπαθήσουμε να περιορίσουμε αυτό το πρόβλημα, επιλέξαμε να χρησιμοποιήσουμε την παραλλαγή του k-NN έτσι ώστε να μην υπάρξει σημαντική απόκλιση από τις πραγματικές ζώνες με την αλλαγή της θέσης λίγων Wi-Fi. Σε κάθε περίπτωση όμως, θα πρέπει να ενθαρρύνεται ο υπεύθυνος να ανανεώνει τις τιμές των ζωνών σε τακτά χρονικά διαστήματα.

Τέλος, είναι φανερό πως η αποδοτικότητα της παραπάνω μεθόδου είναι στενά συνδεδεμένη με τον αριθμό των διαθέσιμων Wi-Fi μιας περιοχής. Ένα διαμέρισμα μεγαλούπολης θα αναγνωρίζει δεκάδες πομπούς για κάθε ζώνη, ενώ ένα σπίτι σε χωρίο μπορεί να έχει δυο με τρία Wi-Fi διαθέσιμα. Αυτός είναι ένας περιορισμός που δε μπορεί να λυθεί εύκολα από τη στιγμή που επιλέξαμε να χρησιμοποιήσουμε τη συγκεκριμένη τεχνολογία.

3.2 Server

Για τον εξυπηρετητή (server) του συστήματος μας ακολουθήσαμε κάποιες βέλτιστες τακτικές και τις υλοποιήσαμε με μοντέρνες τεχνικές και τεχνολογίες με κύριο στόχο τη διατήρηση υψηλής επεκτασιμότητας και ως προς τη προσθήκη νέων λειτουργιών αλλά και ως προς τη εξυπηρέτηση περισσότερων πελατών (client). Στις παρακάτω υποενότητες θα εξετάσουμε τις τεχνολογίες που χρησιμοποιήσαμε στον server και θα κάνουμε μια επισκόπηση της βάσης που διατηρούμε και η οποία συνυπάρχει με τον server.

3.2.1 Spring framework

Το Spring framework [35] είναι ένα πλαίσιο με το οποίο μπορούν να συνυπάρχουν και να επικοινωνούν αποδοτικά διαφορετικές εφαρμογές γραμμένες σε Java. Είναι υπεύθυνο για τη διαχείριση του κύκλου ζωής για τις διαφορετικές εφαρμογές και λειτουργεί ως μεσάζοντας μεταξύ τους. Οι εφαρμογές που μπορούν να γίνουν εύκολα διαχειρίσιμες μέσω του Spring framework περιλαμβάνουν λογισμικά για υλοποίηση model-view-controller μοντέλων αρχιτεκτονικής (MVC frameworks), λογισμικά για χαρτογράφηση αντικειμένων της Java σε πίνακες σχεσιακής βάσης (ORM frameworks) και ακόμα, λογισμικά απαραίτητα για ένα σύγχρονο εξυπηρετητή όπως λογισμικά για εργασίες που αφορούν σε αποστολή ηλεκτρονικών μηνυμάτων ή εξυπηρέτησης αιτημάτων μέσω χρονοδρομολόγησης. Αξίζει να αναφέρουμε ότι είναι ελεύθερο προς χρήση με βάσει την έκδοση άδειας χρήσης 2.0 της Apache.

Στα πλαίσια της παρούσας εργασίας χρησιμοποιήσαμε το Spring Boot [36], μία εφαρμογή γραμμένη πάνω στο Spring framework με σκοπό να απλοποιήσει τη ρύθμιση των διαφορετικών εφαρμογών που φιλοξενεί το Spring framework. Αυτή η απλοποίηση επιτυγχάνεται με την προ-ρύθμιση και προ-διαμόρφωση των διαφορετικών τεχνολογιών/εφαρμογών με προκαθορισμένες τιμές. Επίσης περιλαμβάνει ενσωματωμένο εξυπηρετητή εφαρμογών (Tomcat, Jetty, Undertow κτλ.). Ο χρήστης του Spring boot έχει

αργότερα την δυνατότητα να αλλάξει κάποιες από αυτές τις ρυθμίσεις ανάλογα με τις απαιτήσεις του συστήματος που αναπτύσσει.

3.2.2 RESTful Web Services

Για την επικοινωνία μεταξύ του εξυπηρετητή και των εφαρμογών πελατών σε Android επιλέξαμε Representational State Transfer (RESTful) [37] διαδικτυακές υπηρεσίες. Οι RESTful υπηρεσίες, μια δεκαετία μετά την πρώτη τους εμφάνιση, έχουν αποδειχθεί ως η πιο σημαντική τεχνολογία σε εφαρμογές διαδικτύου. Δεν είναι τυχαίο πως όλες οι γλώσσες ανάπτυξης εφαρμογών έχουν πλέον υποστήριξη για τη ανάπτυξη REST υπηρεσιών. Το Spring boot επιτρέπει τον ορισμό RESTful υπηρεσιών με κατάλληλη χρήση υπομνηματισμού (annotation) στο κώδικα ανάπτυξης (π.χ. @RestController, @RequestMapping κ.α.).

Η REST αρχιτεκτονική επιτρέπει ελάχιστο βαθμό σύζευξης μεταξύ πελάτη και εξυπηρετητή, με αποτέλεσμα η εξέλιξη του συστήματος να μπορεί πιο εύκολα να απορροφήθει από τους πελάτες του συστήματος. Αυτή η αρχιτεκτονική προσέγγιση είναι άλλωστε και ο βασικός λόγος εξάπλωσης των συγκεκριμένων υπηρεσιών, καθώς οι άλλες προσεγγίσεις απαιτούν μεγάλο βαθμό σύζευξης μεταξύ πελάτη και εξυπηρετητή, με αποτέλεσμα, ακόμα και μικρές αλλαγές να απαιτούν σημαντική προσπάθεια στη διαδικασία του συγχρονισμού.

Παρακάτω αναφέρουμε επιγραμματικά τις διαφορετικές REST υπηρεσίες που φιλοξενεί ο εξυπηρετητής μας:

Πίνακας 3: Endpoint και τύπος REST υπηρεσιών

API	Τύπος Υπηρεσίας	Endpoint
Authentication	POST	authenticate/login
	POST	authenticate/register
DataCollection	POST	data/sendDataPacket
	GET	data/shouldRun/{userId}
	POST	data/startDataCollection
	POST	data/stopDataCollection
Zone	GET	zone/dangerZones/{userId}

	GET	zone/safeZones/{userId}
	POST	zone/getZone
	POST	zone/isInDangerZone
	POST	zone/registerZone
ActivityRecognition	POST	activityrecognition/{userId}

Για καθένα από τα διαφορετικά endpoint παραθέτουμε τις παραμέτρους εισόδου/εξόδου και μια σύντομη περιγραφή της λειτουργίας τους παρακάτω:

Πίνακας 4: Παράμετροι εισόδου/εξόδου και περιγραφή REST υπηρεσιών

Endpoint	Παράμετροι	Περιγραφή υπηρεσίας
authenticate/login	username – string password – string token – string Επιστρέφει: response SimpleResponse	Είσοδος στο σύστημα με credentials username και password. Το token είναι το FCM διακριτικό εγγραφής (βλέπε 3.3.4).
authenticate/register	user - User class Επιστρέφει: response SimpleResponse	Εγγραφή στο σύστημα του χρήστη user.
data/sendDataPacket	dataPacket DataPacket – { List<AccelerometerStats>, User } - DataPacket class Επιστρέφει: response SimpleResponse	Είσοδος στο σύστημα του πακέτου accelerometerStats από το χρήστη user.
data/shouldRun/{userId}	userId – string Επιστρέφει: response SimpleResponse	Έλεγχος για την έναρξη της ανίχνευσης πτώσης από το χρήστη με userId.
data/startDataCollection	user – User class Επιστρέφει: response SimpleResponse	Ο χρήστης user δηλώνει την επιθυμία έναρξης ανίχνευσης πτώσης για το συνδεδεμένο με αυτόν

		ηλικιωμένο
data/stopDataCollection	user – User class Επιστρέφει: response SimpleResponse	Ο χρήστης user δηλώνει την επιθυμία διακοπής της ανίχνευσης πτώσης για το συνδεδεμένο με αυτόν ηλικιωμένο
zone/dangerZones/{userId}	userId – string Επιστρέφει: zones List<Zones>	Επιστροφή των επικίνδυνων ζωνών του χρήστη με userId.
zone/safeZones/{userId}	userId – string Επιστρέφει: zones List<Zones>	Επιστροφή των ασφαλών ζωνών του χρήστη με userId.
zone/getZone	wifiList - List<Wifi> zone – Zone Επιστρέφει: zoneId – string	Έλεγχος για την τωρινή ζώνη του χρήστη user με βάση τη wifiList που συλλέχθηκε.
zone/registerZone	wifiList - List<Wifi> zone – Zone Επιστρέφει: response SimpleResponse	Εγγράφει τη ζώνη zone που περιγράφεται από τη wifiList ως ασφαλής ή επικίνδυνη (ανάλογα με την επισήμανση που έχει δωθεί στο αντικείμενο zone).
activityrecognition/{userId}	recognized Activity - RecognizedActivity { ... user – User } class Επιστρέφει: response SimpleResponse	Αποθηκεύει την τωρινή δραστηριότητα του χρήστη user (βλέπε 3.3.3)

3.2.3 Swagger API Framework

Στην εργασία μας χρησιμοποιήσαμε επίσης το Swagger API framework [38], ένα ελεύθερο λογισμικό βάσει της άδειας χρήσης Apache 2.0.

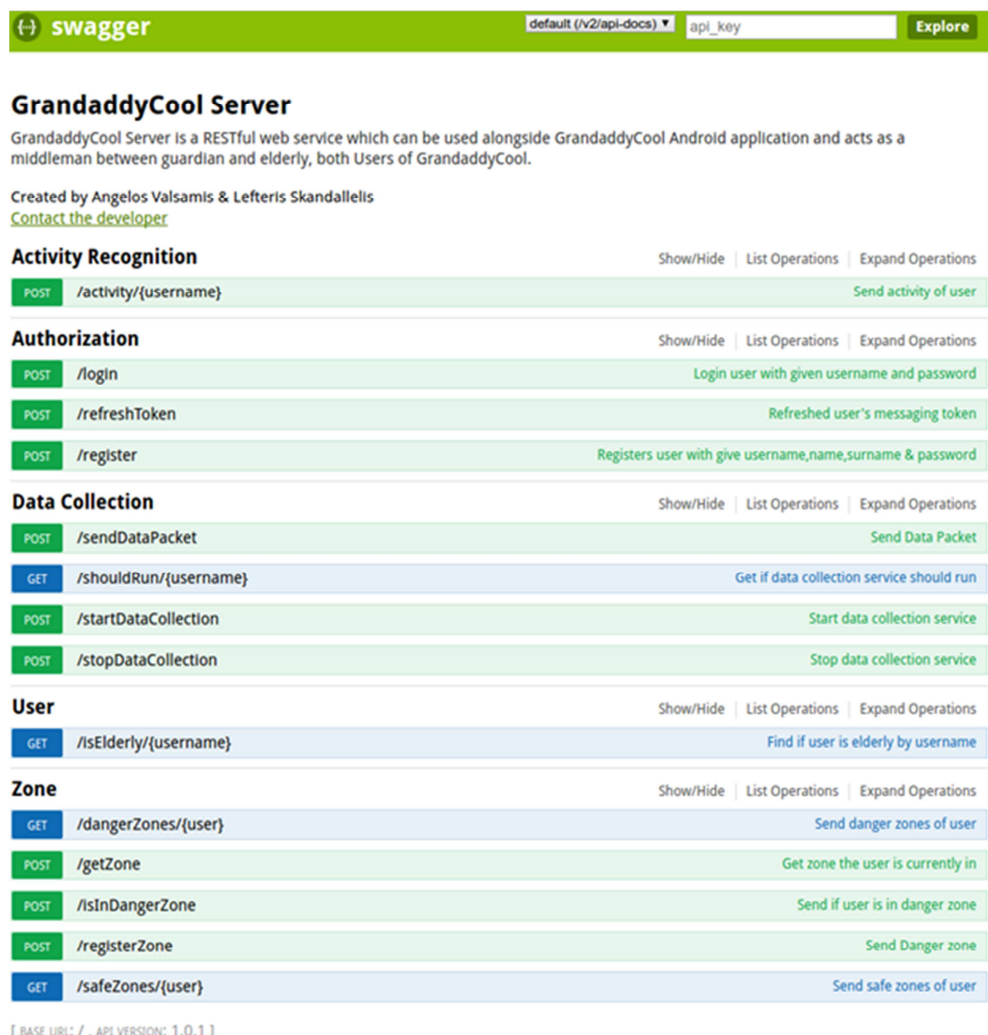
Το Swagger επιτρέπει τη περιγραφή των APIs με τρόπο που να είναι κατανοητός από διάφορα εργαλεία του Swagger αλλά και τρίτων. Τα βασικά εργαλεία που προσφέρει το Swagger είναι παραγωγή διαδραστικού API τεκμηρίωσης (documentation), αυτόματη

παραγωγή βιβλιοθηκών πελάτη (client libraries) σε πληθώρα γλωσσών αλλά και παραγωγή κώδικα σκελετού για *testing*.

Οι βασικές παράμετροι που πρέπει να αποσαφηνιστούν από το προγραμματιστή για την παραγωγή του Swagger specification αρχείου είναι:

- Καταγραφή των διαφορετικών λειτουργιών που υποστηρίζει το σύστημα
- Καταγραφή των παραμέτρων που χρειάζονται ως είσοδο για τη κάθε λειτουργία αλλά και την έξοδο της κάθε λειτουργίας
- Περιγραφή τυχόν απαιτήσεων authorization που χρειάζεται το σύστημα
- Διάφορες μετα-πληροφορίες, όπως όροι, άδειες και τρόπους επικοινωνίας με τον προγραμματιστή.

Παρότι το Swagger API framework προσφέρει πολλά περισσότερα από μία διαδραστική τεκμηρίωση, στα πλαίσια της εργασίας μας, χρειάστηκε να εκμεταλλευτούμε μονάχα αυτό το χαρακτηριστικό. Στην Εικόνα 10, παρουσιάζουμε το documentation που παράχθηκε με βάση τις διαθέσιμες διαδικτυακές υπηρεσίες του εξυπηρετητή μας:



Εικόνα 10: Τεκμηρίωση API μέσω Swagger

3.2.4 Hibernate ORM Framework

Το Hibernate ORM [39] είναι ένα εργαλείο για χαρτογράφηση αντικειμένων της Java σε πίνακες σχεσιακής βάσης (ORM framework) το οποίο κυκλοφορεί με την LGPL v2.1 άδεια χρήσης.

Το κύριο χαρακτηριστικό του Hibernate είναι η χαρτογράφηση από κλάσεις Java σε πίνακες βάσεων δεδομένων και η αντιστοίχιση τύπων δεδομένων Java σε τύπους δεδομένων SQL. Η λειτουργία Hibernate παρέχει εργαλεία για αυτόματη ή ημιαυτόματη δημιουργία ερωτημάτων SQL.

Η διαχείριση της επικοινωνίας του εξυπηρετητή με την σχεσιακή βάση είτε για την αποθήκευση των δεδομένων, είτε για την ανάκτηση τους με τη παραγωγή κατάλληλων ερωτημάτων ανακουφίζει τον προγραμματιστή από μια επίπονη διαδικασία μετατροπής

```
@Entity
@Table(name = "zone")
public class Zone {

    @Id
    @GeneratedValue(strategy = GenerationType.AUTO)
    private Long zoneId;

    private String friendlyName;

    @ManyToOne
    @JoinColumn(name = "userId")
    private User user;

    private Integer isSafe;

    //Constructors and getters/setters

}
```

```
'CREATE TABLE `zone`
(`zone_id` bigint(20) NOT NULL AUTO_INCREMENT,
`friendly_name` varchar(255) DEFAULT NULL,
`is_safe` int(11) DEFAULT NULL,
`user_id` bigint(20) DEFAULT NULL,
PRIMARY KEY (`zone_id`),
KEY `FK1mlqb6n2hcrrmyjlyo5qn6vep` (`user_id`),
CONSTRAINT `FK1mlqb6n2hcrrmyjlyo5qn6vep`
FOREIGN KEY (`user_id`)
REFERENCES `user` (`user_id`)
)
ENGINE=InnoDB DEFAULT CHARSET=latin1'
```

Εικόνα 11: Αυτόματη μετατροπή κλάσης σε MySQL πίνακα μέσω Hibernate

των αντικειμένων που χρησιμοποιεί στη Java σε πίνακες με χρήση SQL. Έτσι η χρήση του Hibernate μας επέτρεψε να αποφύγουμε την δημιουργία μεγάλου όγκου κώδικα (και συχνά επαναλαμβανόμενου) και να επικεντρωθούμε στην βασική επιχειρησιακή λογική της εφαρμογής και όχι με την επιρρεπή σε σφάλματα σύνταξη της SQL.

Στην Εικόνα 11 παρουσιάζουμε πώς το Hibernate μετατρέπει αυτόματα μια κατάλληλα υπομνηματισμένη κλάση Java σε MySQL πίνακα.

Παρόλο που το Hibernate Framework δεν είναι η μόνη ORM λύση, έχει γίνει πολύ ιάσημο κατά το πρόσφατο παρελθόν, λόγω της μεγάλης ποικιλίας λειτουργιών που προσφέρει σε σύγκριση με τους ανταγωνιστές του.

3.2.5 Gradle

Στα πλαίσια της εργασίας μας, χρησιμοποιήσαμε τα εργαλεία κατασκευής Gradle [40] και για την εφαρμογή εξυπηρετητή αλλά και για την εφαρμογή πελάτη.

Τα εργαλεία κατασκευής είναι προγράμματα που αυτοματοποιούν τη δημιουργία εκτελέσιμων εφαρμογών από τον πηγαίο κώδικα (π.χ. .apk για τις εφαρμογές Android). Η αυτοματοποίηση της κατασκευής είναι η πράξη της συγγραφής ή αυτοματοποίησης μιας ευρείας ποικιλίας εργασιών που κάνουν οι προγραμματιστές λογισμικού στις καθημερινές τους δραστηριότητες και περιλαμβάνουν:

- Λήψη εξαρτήσεων (συνήθως βιβλιοθήκες τρίτων).
- Μεταγλώττιση πηγαίου κώδικα σε δυαδικό κώδικα.
- Συσκευασία του δυαδικού κώδικα και δυνατότητα συμπίεσης του.
- Εκτέλεση δοκιμών για εξακρίβωση ορθής λειτουργικότητας εφαρμογής.
- Ανάπτυξη σε συστήματα παραγωγής.

Το Gradle είναι ένα σύστημα αυτοματισμού ανοιχτού κώδικα (Apache License, έκδοση 2.0) που βασίζεται στις ιδέες του Apache Ant και του Apache Maven και σκοπό έχει τη παροχή ενός εύκολου και ταυτόχρονα εύκολα προσαρμόσιμου τρόπου περιγραφής του συνόλου των εργαλείων που απαιτούνται για τη κατασκευή και δημοσίευση της εφαρμογής. Αυτό το επιτυγχάνει με την χρήση αρχείων που συντάσσονται με τη βοήθεια της γλώσσας Groovy, αντί της μορφής XML που χρησιμοποιεί το Apache Maven, για να δηλώσει τη διαμόρφωση της εφαρμογής. Το Gradle χρησιμοποιεί ένα κατευθυνόμενο ακυκλικό γράφημα για να καθορίσει τη σειρά με την οποία μπορούν να εκτελεστούν οι εργασίες.

3.2.6 Οργάνωση βάσης

Μαζί με τον server μας, διατηρούμε και μία MySQL βάση, την οποία έχουμε οργανώσει στους παρακάτω πίνακες:

- Πίνακας **user**

Σε αυτόν τον πίνακα, αποθηκεύουμε τους χρήστες της εφαρμογής μας. Κατά τη φάση της εγγραφής του χρήστη, εγγράφουμε και τον ηλικιωμένο και τον υπεύθυνο του, χρησιμοποιώντας διαφορετικά αναγνωριστικά (id) χρήστη, και τους συνδέουμε χρησιμοποιώντας τον πίνακα `elderly_responsible` (περιγράφεται παρακάτω). Για κάθε χρήστη κρατάμε ένα μοναδικό αναγνωριστικό, το όνομα, επώνυμο, όνομα χρήστη (username) και κωδικό, καθώς και μια συμβολοσειρά (token), το οποίο παράγεται μέσω Google API και είναι μοναδικό για κάθε συσκευή, και χρησιμοποιείται για να στείλουμε την ειδοποίηση (notification) από τον server στην εφαρμογή του υπεύθυνου για τον ηλικιωμένο.

- Πίνακας **elderly_responsible**

Όπως περιγράψαμε παραπάνω, αυτός ο πίνακας κρατάει μόνο τα αναγνωριστικά του ηλικιωμένου και του υπεύθυνου του, έτσι ώστε να τους κρατάμε σε ζευγάρια.

- Πίνακας **accelerometer_stats**

Αυτός ο πίνακας περιέχει τα δεδομένα, που έρχονται από τον αισθητήρα επιταχυνσιόμετρο του κινητού, και τα οποία είναι απαραίτητα για να τρέξει ο αλγόριθμος ανίχνευσης πτώσης. Σε αυτόν έχουμε ένα μοναδικό αναγνωριστικό για κάθε εγγραφή, μαζί με τις τιμές επιτάχυνσης στους άξονες X, Y και Z, όπως ακριβώς τις έχει μετρήσει το κινητό, καθώς και το αναγνωριστικό χρήστη (ηλικιωμένου) από το κινητό του οποίου ήρθε η συγκεκριμένη μέτρηση.

- Πίνακας **wifi**

Σε αυτόν τον πίνακα αποθηκεύουμε τα δεδομένα από τα ασύρματα δίκτυα, τα οποία έχει εντοπίσει η εφαρμογή, κατά τη φάση της προσθήκης από τον υπεύθυνο του ηλικιωμένου των επικίνδυνων και ασφαλών ζωνών. Κρατάμε ένα αναγνωριστικό, την διεύθυνση mac και το όνομα του κάθε ασύρματου δικτύου.

- Πίνακας **zone**

Αυτός ο πίνακας περιέχει τα δεδομένα για τις ζώνες κάθε χρήστη. Κάθε ζώνη έχει ένα αναγνωριστικό, ένα όνομα το οποίο ο χρήστης έχει θέσει όταν πρόσθεσε την ζώνη, μία μεταβλητή που δείχνει αν η ζώνη είναι ασφαλής ή επικίνδυνη και το αναγνωριστικό χρήστη (υπεύθυνου) στον οποίο ανήκει η ζώνη.

- Πίνακας **wifi_in_zone**

Σε αυτό τον πίνακα γίνεται η σύνδεση μεταξύ wifi και ζώνης στην οποία ανήκουν, καθώς και την ισχύ σήματος που λαμβάνει η android συσκευή από το συγκεκριμένο wifi σε αυτή τη ζώνη. Έτσι ένα wifi μπορεί να υπάρχει σε πολλές ζώνες, αλλά σε κάθε μία έχει διαφορετική τιμή ισχύος σήματος.

- Πίνακας **user_in_zone**

Σε αυτό τον πίνακα, αποθηκεύουμε πληροφορίες για την τρέχουσα ζώνη στην οποία βρίσκεται ο ηλικιωμένος. Για κάθε εγγραφή κρατάμε ένα μοναδικό αναγνωριστικό, την τρέχουσα ημερομηνία/ώρα (timestamp), το αναγνωριστικό του (ηλικιωμένου) χρήστη και το αναγνωριστικό της ζώνης.

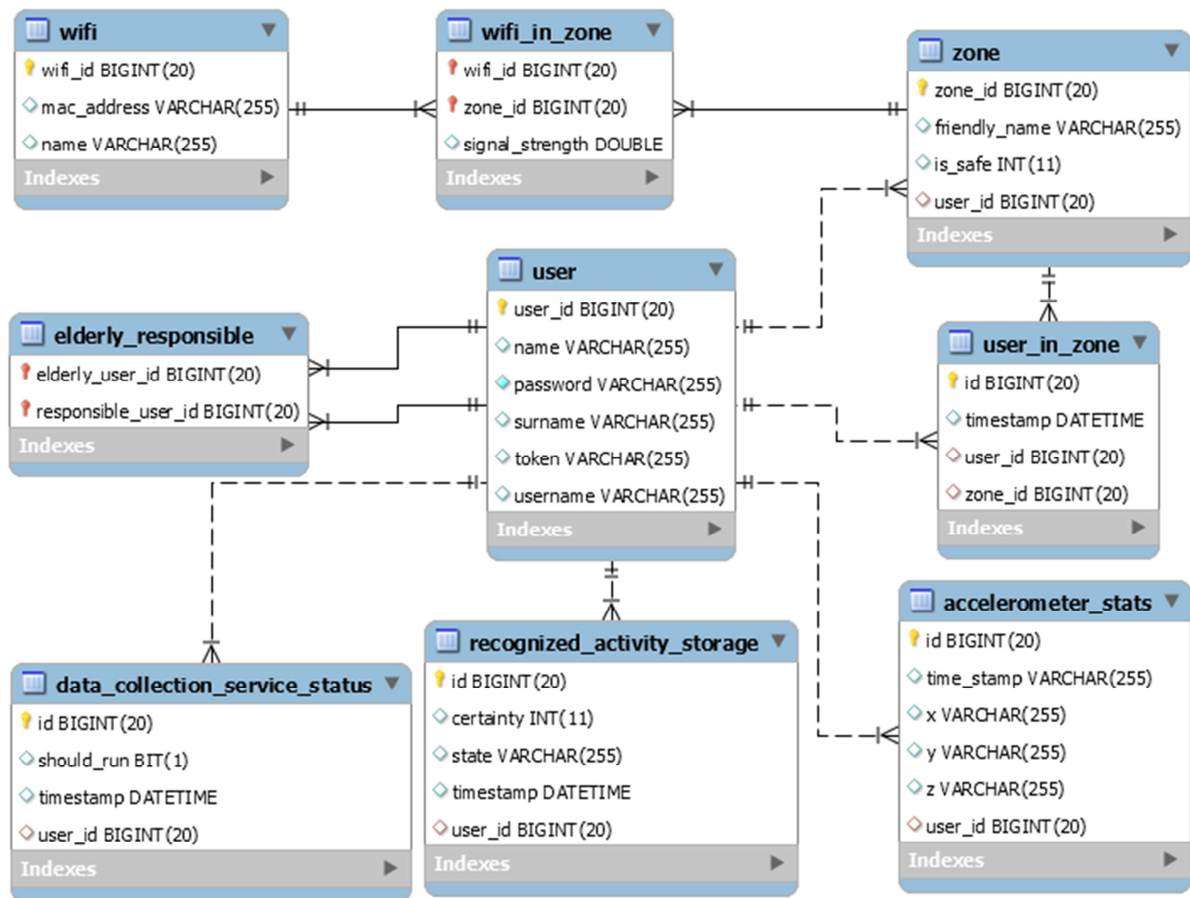
- Πίνακας **data_collection_service_status**

Αυτός ο πίνακας περιέχει την πληροφορία που χρειάζεται τόσο ο server όσο και ο (android) client, για να αποφασίσουν αν οι διάφορες λειτουργίες (συλλογή και αποστολή δεδομένων από τους αισθητήρες, ανίχνευση πτώσης, ανίχνευση ζώνης) πρέπει να τρέξουν τη συγκεκριμένη χρονική στιγμή. Κάθε εγγραφή περιέχει ένα αναγνωριστικό, το timestamp, το αναγνωριστικό του (ηλικιωμένου) χρήστη, και μία Boolean μεταβλητή που καθορίζει αν οι διάφορες λειτουργίες πρέπει να τρέξουν. Πάντα κοιτάμε την πιο πρόσφατη εγγραφή για έναν συγκεκριμένο χρήστη, για να πάρουμε αυτή την απόφαση.

- Πίνακας **recognized_activity_storage**

Τέλος, σε αυτό τον πίνακα, αποθηκεύουμε την πληροφορία που λαμβάνουμε από το API της Google σχετικά με τις τελευταίες δραστηριότητες που έκανε ο ηλικιωμένος. Οι εγγραφές περιέχουν ένα αναγνωριστικό, την δραστηριότητα που βρέθηκε (still, tilting, foot κτλ), το ποσοστό βεβαιότητας για την συγκεκριμένη δραστηριότητα, το timestamp και το αναγνωριστικό του (ηλικιωμένου) χρήστη.

Παρακάτω, στο Σχήμα 4, παραθέτουμε ένα διάγραμμα της βάσης, αυτόματα παραγόμενο με το εργαλείο MySQL Workbench 6.3 CE:



Σχήμα 4: Διάγραμμα οντοτήτων συσχετίσεων (EER diagram) της βάσης μας

3.3 Εφαρμογή Android

Για το τμήμα του πελάτη (client) της εφαρμογής μας, έχουμε υλοποιήσει μια εφαρμογή σε Android, η οποία αποτελείται κατά κύριο λόγο από κάποια Activities και Services, τα οποία θα δούμε αναλυτικότερα παρακάτω. Επίσης, έχουμε χρησιμοποιήσει το API που προσφέρει η Google για αναγνώριση δραστηριότητας. Η επικοινωνία με τον RESTful server γίνεται μέσω Retrofit, καθώς επίσης χρησιμοποιείται και λογική με ειδοποιήσεις (notifications) μέσω του Firebase Cloud Messaging για την ειδοποίηση του υπεύθυνου του ηλικιωμένου για πιθανή πτώση.

3.3.1 Android Activities

Το σκεπτικό για την εφαρμογή μας είναι ότι θα είναι διαφορετική για τον ηλικιωμένο και για τον υπεύθυνο του. Συγκεκριμένα, στην πλευρά του κινητού του ηλικιωμένου δεν χρειάζεται να υπάρχει γραφικό περιβάλλον (UI), καθώς θεωρούμε πως ο ηλικιωμένος δεν γνωρίζει ή δεν είναι σε θέση να χρησιμοποιήσει την εφαρμογή για κάποιο σκοπό. Έτσι η αρχική ρύθμιση της εφαρμογής γίνεται από τον υπεύθυνο, ο οποίος έχει τη δυνατότητα να ανοίγει

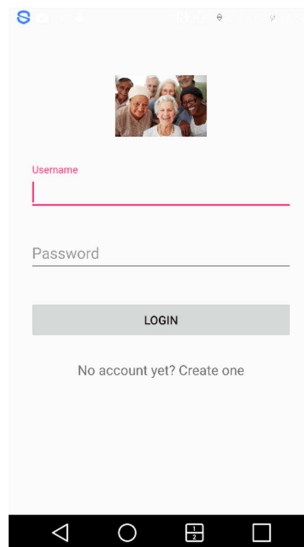
και να κλείνει απομακρυσμένα την υπηρεσία συλλογής δεδομένων, να προσθέτει ζώνες κτλ. Τα android activities που έχουμε χρησιμοποιήσει για το UI της εφαρμογής είναι τα παρακάτω:

- **AuthenticationActivity**

Το συγκεκριμένο Activity είναι το πρώτο που βλέπει ο χρήστης όταν ξεκινάει την εφαρμογή και με το οποίο μπορεί να πραγματοποιήσει τις λειτουργίες της σύνδεσης (login) και της εγγραφής (register). Προγραμματιστικά, το Activity χωρίζεται σε δύο Fragment, τα οποία είναι συστατικά του Android, που χρησιμοποιούνται για να απεικονίσουν μέρος ενός Activity προς το χρήστη και να δημιουργήσουν έτσι ένα UI πολλών παραθύρων κάτω από ένα κοινό Activity.

Αρχικά, το Activity ελέγχει αν ο χρήστης έχει κάνει ήδη login μία φορά στην εφαρμογή. Αυτό γίνεται ελέγχοντας τα SharedPreferences, τα οποία είναι άλλος ένας μηχανισμός του Android, ο οποίος κρατάει δεδομένα της εφαρμογής μόνιμα στη συσκευή ώστε να χρησιμοποιούνται σε κάθε χρήση της. Έτσι όταν ο χρήστης έχει ήδη κάνει login μία φορά, έχει αποθηκευτεί το όνομα χρήστη (username) του στα SharedPreferences και δεν χρειάζεται να ξαναπεράσει την διαδικασία του login.

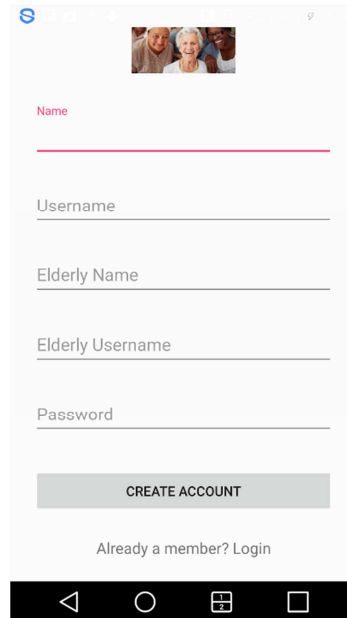
Αν δε βρεθεί όνομα χρήστη στα SharedPreferences, τότε ο χρήστης μεταφέρεται αυτόματα στο LoginFragment, όπου του ζητείται το όνομα χρήστη και ο κωδικός που έχει ορίσει για να συνδεθεί στην εφαρμογή. Αφού δώσει τα στοιχεία αυτά μπορεί να πατήσει το κουμπί “Login”, ώστε να μπει στην κυρίως εφαρμογή.



Εικόνα 12: AuthenticationActivity - LoginFragment

Κάτω από το Login υπάρχει επιλογή που μπορεί να πατήσει ο χρήστης, ώστε αν δεν έχει λογαριασμό να τον δημιουργήσει. Αν ο χρήστης πατήσει στη συγκεκριμένη επιλογή, θα μεταφερθεί στο SignupFragment, όπου μπορεί να πραγματοποιήσει την εγγραφή του στην υπηρεσία. Στο σημείο αυτό, ο υπεύθυνος του ηλικιωμένου, δηλώνει το όνομα του και ένα αναγνωριστικό για τον ίδιο, καθώς και το όνομα του ηλικιωμένου και ένα αναγνωριστικό για τον ηλικιωμένο. Τέλος επιλέγει έναν κωδικό, ο οποίος είναι κοινός και για τους δύο

χρήστες. Αφού ορίσει αυτά και πατήσει στην επιλογή “Create Account”, γίνεται κλήση προς τον server για να αποθηκεύσει στη βάση και τους δύο χρήστες, καθώς και τη σύνδεση μεταξύ τους.



Εικόνα 13: AuthenticationActivity - SignupFragment

Μετά από αυτό, το UI επιστρέφει στο AuthenticationActivity που με τη σειρά του στέλνει το χρήστη στο LoginFragment για να πραγματοποιήσει πλέον είσοδο. Όταν ο χρήστης πραγματοποιήσει είσοδο γίνονται τα εξής: αρχικά καλείται ο server στην login μέθοδο ώστε να μπει στη βάση το token της συσκευής του χρήστη από την οποία έγινε το login. Αυτό θα χρησιμοποιηθεί, ώστε να μπορέσει να πάει το notification στη συσκευή του υπεύθυνου χρήστη (βλέπε ενότητα 3.3.4). Στη συνέχεια αποθηκεύεται το username του χρήστη (είτε υπεύθυνου είτε ηλικιωμένου) στα SharedPreferences της εφαρμογής, ώστε να μη χρειαστεί να κάνει είσοδο ξανά και τέλος ο χρήστης μεταφέρεται πλέον στο MainActivity (αν είναι ο υπεύθυνος) ή στο DataCollectionActivity (αν είναι ο ηλικιωμένος).

- **DataCollectionActivity**

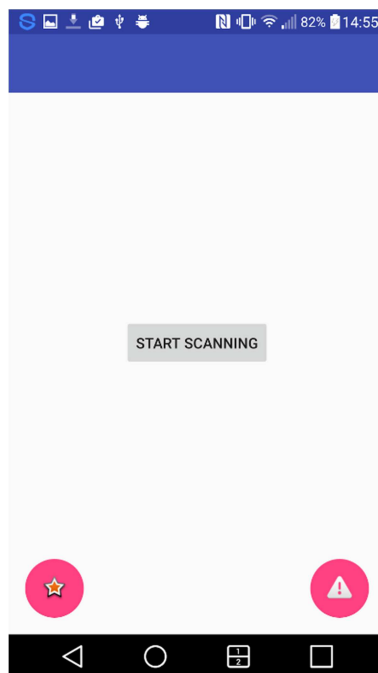
Για το συγκεκριμένο Activity δεν έχουμε να πούμε πολλά, καθώς όπως είπαμε ο σκοπός μας ήταν ο ηλικιωμένος να μην έχει UI. Οπότε το συγκεκριμένο Activity το μόνο που κάνει είναι να ξεκινάει τα δύο Service που τρέχουν στην εφαρμογή του ηλικιωμένου (βλέπε ενότητα 3.3.2) για την συλλογή δεδομένων και την ανίχνευση δραστηριότητας. Στη συνέχεια κάνουμε finish() και το Activity κλείνει αυτόματα, οπότε ο χρήστης μεταφέρεται στο γραφικό του κινητού του, έξω από την εφαρμογή και τα services τρέχουν στο παρασκήνιο. Τέλος να σημειώσουμε ότι με τη βοήθεια ενός BroadcastReceiver, που ονομάζουμε StartDataCollectionServiceAtBootReceiver, το συγκεκριμένο Activity ξεκινάει να εκτελείται αυτόματα, κάθε φορά που η συσκευή για οποιονδήποτε λόγο επανεκκινηθεί. Έτσι αν η συσκευή του ηλικιωμένου κλείσει, το μόνο που χρειάζεται είναι να την ανοίξει ξανά, και η εφαρμογή θα τρέξει αυτόματα, χωρίς να χρειαστεί κάποια επιπλέον ενέργεια από τον ηλικιωμένο.

- **MainActivity**

Το συγκεκριμένο Activity είναι το κύριο UI που παρέχεται στον υπεύθυνο του ηλικιωμένου και με το οποίο μπορεί να πραγματοποιήσει διάφορες λειτουργίες.

Αρχικά, στη μέση της σελίδας υπάρχει το κουμπί για την λειτουργία της έναρξης ή παύσης της αποστολής δεδομένων από την εφαρμογή του ηλικιωμένου στη βάση (Start scanning/Stop scanning). Όταν ο χρήστης επιλέξει Start scanning τότε μπαίνει μια εγγραφή στον πίνακα `data_collection_service_status`, με το boolean `should_run` να παίρνει την τιμή `true`. Αυτό στη συνέχεια χρησιμοποιείται από το Service της εφαρμογής του ηλικιωμένου για να αρχίσει να στέλνει δεδομένα από το επιταχυνσιόμετρο, καθώς και δεδομένα για την τρέχουσα ζώνη του ηλικιωμένου. Αντίστοιχα όταν επιλεγθεί το Stop scanning, το `should_run` γίνεται `false` και σταματάει η αποστολή των δεδομένων.

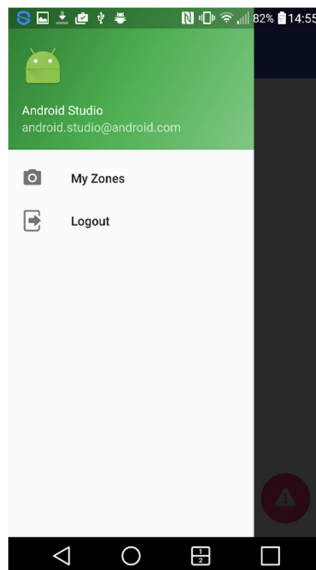
Στο ίδιο Activity, υπάρχουν επίσης δύο κουμπιά κάτω δεξιά και κάτω αριστερά, τα οποία χρησιμοποιούνται κατά την πρώτη συνήθως χρήση της εφαρμογής, για να ρυθμίσει ο χρήστης τις ζώνες στο χώρο του ηλικιωμένου που θεωρεί επικίνδυνες ή ασφαλείς. Πατώντας το κάτω δεξιά κουμπί ορίζει μια επικίνδυνη ζώνη στην οποία δίνει ένα όνομα. Μόλις επιβεβαιώσει το όνομα, τότε η εφαρμογή συλλέγει για 20 δευτερόλεπτα δεδομένα από τα ασύρματα δίκτυα του χώρου και αποθηκεύει μέσες τιμές για την ισχύ σήματος που λαμβάνει το κινητό για κάθε ένα από αυτά τα ασύρματα δίκτυα. Κατόπιν κάνει κλήση στο server με αυτές τις τιμές ώστε να δημιουργηθεί η ζώνη και στη βάση. Αντίστοιχα με το αριστερό κουμπί γίνεται η ίδια διαδικασία για τη δημιουργία ασφαλούς ζώνης.



Εικόνα 14: MainActivity

Επιπλέον, στο `mainActivity` υπάρχει η “κρυφή” επιλογή της πλευρικής μπάρας, όπου αν ο χρήστης κάνει κύλιση (slide) από αριστερά προς τα δεξιά, ανοίγει ένα πλευρικό μενού, από το οποίο μπορεί να επιλέξει “My zones” ή “Logout”. Στη δεύτερη επιλογή, προφανώς, σβήνεται από το `SharedPreferences` το `username` του χρήστη και μεταφέρεται στο

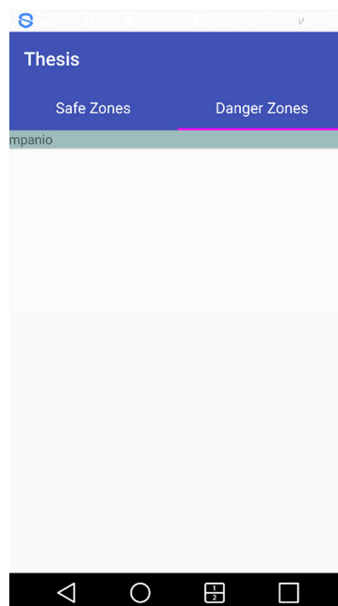
AuthenticationActivity. Όταν επιλέξει το “My zones” μεταφέρεται στο τελευταίο Activity που θα περιγράψουμε, το ZoneActivity.



Εικόνα 15: MainActivity - πλευρική μπάρα

- **ZoneActivity**

Στο συγκεκριμένο Activity, ο χρήστης μπορεί να δει τις ζώνες που έχει αποθηκεύσει και το αναγνωριστικό (id) της κάθε μία από αυτές όπως της έχει δοθεί και είναι αποθηκευμένο στη βάση. Σε αυτή τη φάση δεν υπάρχει κάποια άλλη λειτουργικότητα, αλλά θα μπορούσε στο μέλλον να προστεθεί κουμπί ώστε ο χρήστης να μπορεί να διαγράψει μια ζώνη, είτε να αλλάξει την αποθηκευμένη τιμή της, από ασφαλή σε επικίνδυνη και αντιστρόφως.



Εικόνα 16: ZoneActivity - εμφάνιση ζωνών χρήστη

3.3.2 Android Services

Στην εφαρμογή μας, έχουμε χρησιμοποιήσει μια σειρά από Android Services, από τα οποία τα περισσότερα λειτουργούν σαν daemons και τρέχουν στο παρασκήνιο επιτελώντας τις βασικές λειτουργίες:

- **MyFirebaseInstanceIdService**

Το συγκεκριμένο Service χρησιμοποιεί ένα έτοιμο API της Google για να δώσει στην κάθε συσκευή ένα μοναδικό αναγνωριστικό (token), το οποίο αποθηκεύεται στον server και δίνει τη δυνατότητα αποστολής ειδοποιήσεων στην συγκεκριμένη συσκευή (θα αναλυθεί περισσότερο στην ενότητα 3.3.4).

DataCollectionService

Το συγκεκριμένο Service είναι το ένα από τα δύο που ξεκινάει η DataCollectionActivity, οπότε τρέχει στην εφαρμογή του ηλικιωμένου, και είναι εκείνο που συλλέγει τα δεδομένα από το επιταχυνσιόμετρο στο κινητό και στη συνέχεια τα στέλνει στον server.

Συγκεκριμένα στην μέθοδο onStartCommand, η οποία εκτελείται κάθε φορά που τρέχει το Service, ξεκινάει ένα Thread, το οποίο τρέχει κάθε 15 δευτερόλεπτα, και αυτό που κάνει είναι να διαβάζει μέσω του server την τιμή της μεταβλητής should_run από τον πίνακα data_collection_service_status για το πιο πρόσφατο timestamp. Αν η τιμή είναι true, τότε καλεί την μέθοδο collectData, η οποία αρχικοποιεί έναν Listener ώστε να διαβάζει και να αποθηκεύει τις τιμές του επιταχυνσιόμετρου κάθε φορά που αλλάζουν. Οι τιμές αυτές στέλνονται στον server σε πακέτα των 20. Αν η τιμή της μεταβλητής είναι false, τότε αφαιρεί τους Listeners ώστε να σταματάει η μετάδοση δεδομένων. Με αυτό τον τρόπο βελτιώνεται η απόδοση και μειώνεται ο ρυθμός κατανάλωσης της μπαταρίας όταν η μετάδοση δεδομένων είναι κλειστή.

- **ZoneDetectionService**

Το συγκεκριμένο Service τρέχει αυτόματα μέσω AlarmManager κάθε ένα λεπτό, όταν έχει ξεκινήσει η μετάδοση δεδομένων. Το service αυτό σε κάθε εκτέλεση ανιχνεύει τα ασύρματα δίκτυα (wi-fis) της περιοχής για 20 δευτερόλεπτα και παίρνει τους μέσους όρους των τιμών της ισχύος σήματος που λαμβάνει το κινητό για κάθε ένα από αυτά. Στη συνέχεια στέλνει τα wifis και την τιμή που υπολόγισε για το καθένα στον server, ο οποίος στη μέθοδο getZone συγκρίνει τις τιμές που έλαβε με τις ζώνες που έχει ήδη αποθηκευμένες στη βάση, ώστε να υπολογίσει την τρέχουσα θέση του κινητού (βλέπε ενότητα 3.1.2).

- **RecognitionService**

Είναι το δεύτερο από τα Service που ξεκινάει η DataCollectionActivity και εσωτερικά καλεί το DetectedActivitiesIntentService, το οποίο εμπεριέχει τη λογική για την αναγνώριση δραστηριότητας του χρήστη, όπως παρέχεται από το API της Google. Η συγκεκριμένη λειτουργικότητα θα αναπτυχθεί πιο αναλυτικά στην επόμενη ενότητα.

3.3.3 Αναγνώριση δραστηριότητας χρήστη (Google ActivityRecognition API)

Για την ενημέρωση του χρήστη της δραστηριότητας του ηλικιωμένου την στιγμή κατά την οποία αναγνωρίστηκε η πτώση κάναμε χρήση του ActivityRecognition API της Google [41].

Το ActivityRecognitionAPI είναι μια διεπαφή που μας επιτρέπει να αναγνωρίσουμε την δραστηριότητα του χρήστη μια δεδομένη χρονική στιγμή, χωρίς την ταλαιπωρία της συλλογής ανεπεξέργαστων δεδομένων από αισθητήρες και της εκτέλεσης πολύπλοκων αναλύσεων για να καταλήξουμε σε ένα συμπέρασμα, όπως κάνουμε για την σαφώς πιο δύσκολα αναγνωρίσιμη, πτώση.

Το API καλείτε να κάνει αυτή τη επίπονη διαδικασία για μας. Η συσκευή μπαίνει σε λειτουργία σε διαφορετικά χρονικά διαστήματα και συλλέγονται δείγματα από τους χαμηλής κατανάλωσης αισθητήρες του κινητού. Αυτό έχει σαν αποτέλεσμα να κρατά τη κατανάλωση της μπαταρίας του κινητού στο ελάχιστο δυνατό. Τα δεδομένα που συλλέγει δίνονται σε κάποιο εσωτερικό μοντέλο του API το οποίο επιστρέφει τελικά την ανιχνευόμενη δραστηριότητα μαζί με την εμπιστοσύνη του αποτελέσματος. Οι δραστηριότητες που μπορεί να αναγνωρίσει είναι οι εξής:

- IN_VEHICLE : Η κινητή συσκευή είναι σε κάποιο όχημα, όπως αυτοκίνητο
- ON_BICYCLE: Ο χρήστης της κινητής συσκευής χρησιμοποιεί ποδήλατο
- ON_FOOT: Ο χρήστης της κινητής συσκευής τρέχει ή περπατάει
- RUNNING: Ο χρήστης της κινητής συσκευής τρέχει
- WALKING: Ο χρήστης της κινητής συσκευής περπατάει
- STILL: Η συσκευή είναι ακίνητη
- TILTING: Η κλίση της κινητής συσκευής σε σχέση με τη βαρύτητα μεταβλήθηκε σημαντικά
- UNKNOWN: Αδυναμία αναγνώρισης της δραστηριότητας του χρήστη

Όταν το API έχει αναγνωρίσει το κινητό ως 'STILL' για ένα μακρό χρονικό διάστημα, τότε μπορεί να σταματήσει τη λειτουργία του, μέχρις ότου το κινητό ξεκινήσει να κινείται ξανά. Με αυτό επιτυγχάνει περαιτέρω διατήρηση πόρων.

Όπως προαναφέραμε, μαζί με την δραστηριότητα που αναγνωρίζει ο αλγόριθμος του Activity Recognition, επιστρέφει και ένα ποσοστό το οποίο δηλώνει την βεβαιότητα για την δεδομένη δραστηριότητα. Όταν αυτή ξεπερνάει το 75%, θεωρείται σχεδόν βέβαιη η δραστηριότητα, και σύμφωνα με την τεκμηρίωση του API, τότε μόνο μπορούμε να δράσουμε με βάση αυτή τη πληροφορία.

Στη δική μας περίπτωση, οι δραστηριότητες που μας ενδιέφεραν να συγκεντρώσουμε είναι το αν η συσκευή είναι ακίνητη ή όχι, παρά ταύτα αποθηκεύουμε όλες τις δραστηριότητες που ξεπερνούν το 75% στη βάση. Στη συνέχεια, με βάση την δραστηριότητα που έχει αναγνωρίσει το API πριν την πιθανή πτώση και τη δραστηριότητα που αναγνωρίζει μετά από τη πιθανή πτώση, στο διάστημα τους ενός λεπτού που καθυστερούμε την ενημέρωση του υπεύθυνου, μπορούμε να κάνουμε μια πρόβλεψη για τον τρόπο με τον οποίο έπεσε ο ηλικιωμένος και για την κατάσταση στην οποία βρίσκεται μετά τη πτώση (από καρέκλα-κρεβάτι/εν κινήσει, αναίσθητος/με αισθήσεις). Αυτή την πληροφορία την μεταβιβάζουμε και στον υπεύθυνο.

Αξίζει εδώ να αναφέρουμε πως αν το ActivityRecognition API δεν μας έχει δώσει πληροφορία δραστηριότητας για κάποιο διάστημα πριν την πτώση, ή κάποια πληροφορία για το διάστημα τους ενός λεπτού μετά τη πτώση παραλείπουμε να στείλουμε προβλέψεις, από την στιγμή που δεν έχουμε τις απαραίτητες πληροφορίες για την εξαγωγή τους. Η αναγνώριση της δραστηριότητας από το ActivityRecognitionAPI δεν γίνεται ανά τακτά χρονικά διαστήματα. Για να πετύχει χαμηλή κατανάλωση μπαταρίας, συλλέγει και εξετάζει κάποια δείγματα από τους αισθητήρες του κινητού και δεν κάνει συνεχή χρήση των τελευταίων. Ακόμα και αν ζητήσουμε την γρηγορότερη δυνατή ενημέρωση μας για μεταβολή δραστηριότητας (μέσω της παραμέτρου 'detectionIntervalMillis'), για να δώσει βεβαιότητα >75% σε κάποια δραστηριότητα χρειάζεται να πάρει αρκετά δείγματα κατά τη διάρκεια της δραστηριότητας. Έτσι, αν για παράδειγμα ο φορέας της συσκευής περπατάει για ένα λεπτό στο χώρο, το API μπορεί να μας ενημερώσει για αυτή τη δραστηριότητα όταν περάσει μισό λεπτό. Από τη στιγμή που η ενημέρωση δραστηριότητας είναι μια προαιρετική εργασία για τη συγκεκριμένη εφαρμογή, ο παραπάνω περιορισμός είναι ανεκτός.

3.3.4 Ειδοποιήσεις Android (Firebase Cloud Messaging)

Για την αποστολή της ειδοποίησης πτώσης στο κινητό του υπεύθυνου χρησιμοποιήσαμε το Firebase Cloud Messaging (FCM), της Google [42].

Η πλατφόρμα μηνυμάτων Firebase Cloud Messaging (γνωστή και ως Google Cloud Messaging) είναι μια δωρεάν υπηρεσία ειδοποίησης για κινητά από την Google, η οποία επιτρέπει στους προγραμματιστές εφαρμογών να στέλνουν ειδοποιήσεις από τους FCM server στους χρήστες τους. Τα δεδομένα των ειδοποιήσεων μεταφέρονται από το server της εφαρμογής ή από το γραφικό περιβάλλον ειδοποιήσεων του FCM στους FCM servers και από εκεί στα κινητά των χρηστών. Η αρχιτεκτονική αυτή παρουσιάζεται και στην Εικόνα 17.

Για να επιτευχθεί αυτή η έμμεση επικοινωνία είναι αναγκαία η πιστοποίηση και του server αλλά και των διαφορετικών παραληπτών των ειδοποιήσεων.

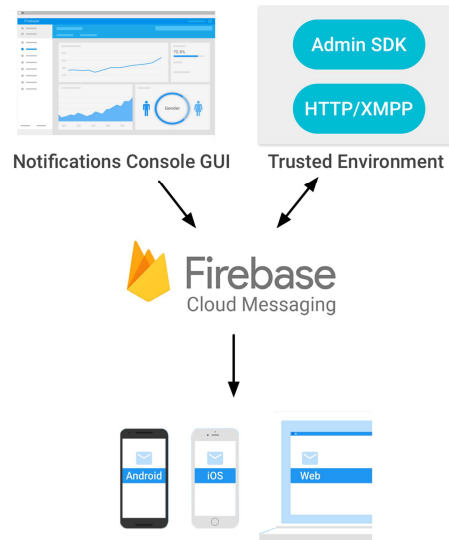
Σχετικά με τη πιστοποίηση των παραληπτών, κατά την αρχική εκκίνηση της Android εφαρμογής, το FCM δημιουργεί ένα διακριτικό εγγραφής. Αυτό το διακριτικό στέλνεται στο server για χρήση στην αποστολή ειδοποιήσεων. Σύμφωνα με τη τεκμηρίωση του εργαλείου, το διακριτικό μπορεί να ανανεωθεί σε διαφορετικές περιπτώσεις όπως όταν:

- Ο χρήστης διαγράψει και εγκαταστήσει εκ νέου την εφαρμογή
- Ο χρήστης καθαρίσει τα δεδομένα της εφαρμογής
- Η εφαρμογή αποκατασταθεί σε μια νέα συσκευή
- Η εφαρμογή διαγράψει το αναγνωριστικό

Για τον λόγο αυτό επιλέγουμε να στέλνουμε την τελευταία έκδοσή του διακριτικού, κάθε φορά που ο χρήστη συνδέεται στην Android εφαρμογή.

Για την πιστοποίηση του server, είναι αναγκαία η δημιουργία ενός κλειδιού ελέγχου ταυτότητας το οποίο προσθέτουμε στο server και το χρησιμοποιούμε σε κάθε κλήση προς

τους FCM servers. Για να πάρουμε αυτό το κλειδί πρέπει να εγγράψουμε την εφαρμογή μας μέσω του προγραμματιστικού λογαριασμού μας στη κονσόλα του Firebase.



Εικόνα 17 – Αρχιτεκτονική του Firebase Cloud Messaging

Το FCM συνήθως στέλνει τα μηνύματα αμέσως μετά την αποστολή τους από το server της εφαρμογής. Ωστόσο, αυτό δεν είναι πάντα δυνατό. Για παράδειγμα, η συσκευή παραλήπτη μπορεί να έχει απενεργοποιηθεί ή να είναι εκτός σύνδεσης. Επίσης, το FCM ενδέχεται να καθυστερήσει σκόπιμα κάποια μηνύματα για να εμποδίσει μια εφαρμογή να καταναλώσει υπερβολικούς πόρους και να επηρεάσει αρνητικά τη διάρκεια ζωής της μπαταρίας.

Όταν συμβεί αυτό, το FCM αποθηκεύει το μήνυμα και το παραδίδει το συντομότερο δυνατό. Ενώ αυτό είναι επιτρεπτό σε κάποιες εφαρμογές, στη δική μας περίπτωση, όπου το μήνυμα είναι εξαιρετικά σημαντικό να φτάσει έγκαιρα στο παραλήπτη του, είναι αναγκαίο να βρεθούν εναλλακτικές λύσεις.

Αποτρέπουμε την περίπτωση της καθυστέρησης των μηνυμάτων με το να τα επισημάνουμε ως υψηλής προτεραιότητας. Το FCM επιχειρεί να αποστείλει αμέσως αυτά τα μηνύματα, ενεργοποιώντας την συσκευή σε περίπτωση που αυτή είναι σε κατάσταση 'ύπνου'. Για την περίπτωση όπου το κινητό του υπεύθυνου είναι εκτός σύνδεσης ή είναι απενεργοποιημένο μπορούμε μελλοντικά να εισαγάγουμε εναλλακτικούς τρόπους επικοινωνίας, όπως την αποστολή αυτόματου μηνύματος ή την δυνατότητα εισαγωγής μιας δεύτερης συσκευής για επικοινωνία σε περίπτωση έκτακτης ανάγκης.

3.3.5 Επικοινωνία με RESTful Server (Square Retrofit)

Για την επικοινωνία της εφαρμογής Android με τον Server μας χρησιμοποιήσαμε τη βιβλιοθήκη Retrofit [43]. Η Retrofit της Square είναι μια βιβλιοθήκη-πελάτη REST υπηρεσιών ανοιχτού κώδικα που χρησιμοποιείται σε Android και Java για να δημιουργήσει HTTP αιτήματα αλλά και να επεξεργαστεί αποτελέσματα αιτημάτων από ένα REST API.

Για την επικοινωνία Android εφαρμογής με web services έχουν δημιουργηθεί κι άλλες βιβλιοθήκες όπως η Volley [44] ή η RestTemplate [45], ωστόσο προτιμήσαμε τη Retrofit (έκδοση 1.9) λόγω προηγούμενης εμπειρίας με τη βιβλιοθήκη.

Για την χρήση της βιβλιοθήκης χρειάζονται κυρίως τρεις βασικές κλάσεις:

1. Μια κλάση διεπαφής που ορίζει τις λειτουργίες HTTP (SimpleApi.java στη περίπτωση μας)

Κάθε μέθοδος της διεπαφής αντιπροσωπεύει μια πιθανή κλήση API. Πρέπει να έχει έναν σχολιασμό HTTP (GET, POST, κλπ.) για να καθορίσει τον τύπο αίτησης και τη σχετική διεύθυνση URL. Η τιμή επιστροφής μπορεί να περιβάλλει την απάντηση σε ένα αντικείμενο Callback με τον τύπο του αναμενόμενου αποτελέσματος.

```
public interface SimpleApi {  
    String BASE_URL = "http://192.168.1.74:8080/";  
  
    @POST(Constants.URL_REGISTER)  
    void registerUser(@Body User user, Callback < SimpleResponse > cb);  
  
    @GET(Constants.URL_SAFE_ZONES)  
    List < Zone > getSafeZones(@Path("user") String userId);  
}
```

Εικόνα 18 – Δείγμα κλάσης διεπαφής SimpleApi

Στην Εικόνα 18 βλέπουμε ένα κομμάτι της διεπαφής SimpleApi, και συγκεκριμένα πώς ορίζουμε την λειτουργία POST εγγραφής χρήστη και τη λειτουργία GET λήψης ασφαλών ζωνών χρήστη. Το Callback στην περίπτωση της λειτουργίας εγγραφής χρήστη, υποδηλώνει ότι η μέθοδος μπορεί να εκτελεστεί ασύγχρονα από το κύριο νήμα ενώ στη περίπτωση της *getSafeZone*, η κλήση δεν είναι ασύγχρονη και για αυτό η μέθοδος θα πρέπει να καλείται μόνο από νήματα που λειτουργούν στο παρασκήνιο έτσι ώστε να μη υπάρχουν κολλήματα στο κύριο νήμα και κατ' επέκταση χαμηλή εμπειρία χρήστη.

2. Μια κλάση παραγωγής RestAdapter (RestApiDispenser στη περίπτωση μας)

Αυτή η κλάση ακολουθεί το πρότυπο εργοστασίου (Factory Pattern) και είναι ουσιαστικά ο τρόπος χρήσης του ίδιου RestAdapter στα διαφορετικά Activities, Fragments ή Services που μπορεί να το χρειαστούν. Σε αυτό το σημείο μπορεί να προστεθούν και οι διαφορετικοί Retrofit μετατροπείς.

Οι Retrofit μετατροπείς είναι μια συμφωνία μεταξύ της Android εφαρμογής και του server για τη μορφή με την οποία θα μεταφέρουν τα δεδομένα. Οι δύο πλευρές μπορούν να συμφωνήσουν ότι για την επικοινωνία τους, η μορφή της μεταφοράς δεδομένων θα είναι JSON (μέσω των βιβλιοθηκών Gson, Jackson ή Moshi) ή XML (μέσω της βιβλιοθήκης SimpleXML). Ο τρόπος που έχει χτιστεί αυτός ο μηχανισμός είναι αρκετά επεκτάσιμος με αποτέλεσμα να μπορούν να

προστεθούν εύκολα και άλλοι μετατροπείς στο μέλλον.

3. Οι κλάση/κλάσεις Java που αναπαριστούν τα αντικείμενα που δίνονται ως παράμετροι ή που επιστρέφουν οι διαφορετικές REST λειτουργίες

Για τη σωστή λειτουργία των Retrofit μετατροπών, είναι απαραίτητη η ύπαρξη απλών κλάσεων Java (Plain Old Java Object, POJO) που αναπαριστούν τα αντικείμενα που παραλαμβάνει ή αποστέλλει το RestAPI.

Αυτά, στη περίπτωση μας, τα παίρνουμε κατ' ευθείαν από τον ορισμό τους στη πλευρά του server. Αν ή ανάπτυξη του server και της εφαρμογής γινόταν από διαφορετικούς φορείς, ο πιο συνηθισμένος τρόπος επικοινωνίας αυτών των κλάσεων είναι με τη δημιουργία μιας RestAPI βιβλιοθήκης πελάτη από τη μεριά του server. Μάλιστα, αν ο server χρησιμοποιεί τα εργαλεία του Swagger Framework, μπορεί να αυτοματοποιήσει αυτή τη διαδικασία με τη χρήση του SwaggerCodegen [46] που δημιουργεί RestAPI βιβλιοθήκες πελάτη για πλειάδα διαφορετικών γλωσσών προγραμματισμού.

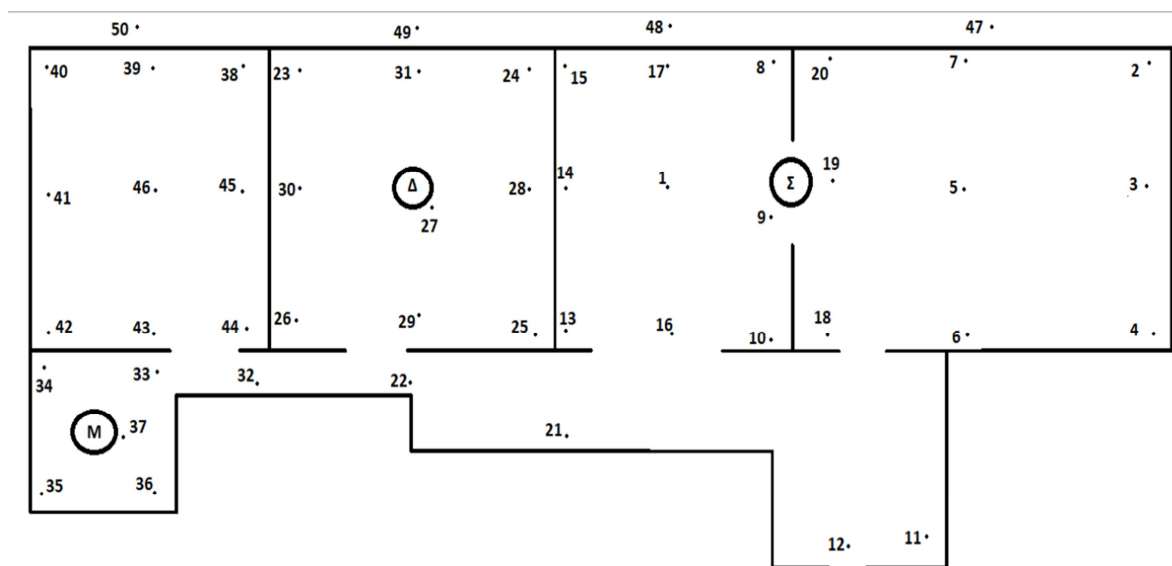
4. ΑΞΙΟΛΟΓΗΣΗ ΣΥΣΤΗΜΑΤΟΣ

Η αξιολόγηση του συστήματος μας χωρίζεται σε δύο τμήματα: στην αξιολόγηση του αλγορίθμου ανίχνευσης πτώσης και στην αξιολόγηση του αλγορίθμου ανίχνευσης ζώνης εσωτερικού χώρου. Αξίζει να σημειώσουμε ότι τα πειράματα πραγματοποιήθηκαν σε κινητό LG G4 με έκδοση Android 6.0. Παρακάτω θα δούμε το περιβάλλον, τις παραμέτρους και τις μετρήσεις που έγιναν, για την αξιολόγηση καθενός από τους δύο αλγορίθμους.

4.1 Περιβάλλον

Το περιβάλλον στο οποίο πραγματοποιήθηκαν τα πειράματα είναι το σπίτι του ενός εκ των δύο συγγραφέων της διπλωματικής, ένας χώρος περίπου 65 τ.μ. με έξι βασικούς χώρους/δωμάτια.

Ειδικότερα για την αξιολόγηση του αλγορίθμου ανίχνευσης ζώνης, δημιουργήθηκαν στο χώρο τρεις βασικές ζώνες, οι οποίες συμβολίζονται με Σ (σαλόνι), η οποία ορίζεται ως επικίνδυνη ζώνη και περιλαμβάνει ουσιαστικά δύο δωμάτια που χωρίζονται με μία πόρτα, Δ (δωμάτιο), όπου ορίζεται ως ασφαλής ζώνη και Μ (μπάνιο), όπου ορίζεται ως επικίνδυνη ζώνη. Στη συνέχεια ορίστηκαν 50 σημεία διεσπαρμένα σε όλο το χώρο του σπιτιού (Εικόνα 19) και σε κάθε σημείο εκτελέσαμε τον αλγόριθμο ανίχνευσης ζώνης για να μας δώσει την εκτίμηση για τη ζώνη στην οποία βρισκόμαστε.



Εικόνα 19: Κάτοψη του σπιτιού όπου έγιναν τα πειράματα

Τα πειράματα για τον αλγόριθμο ανίχνευσης πτώσης εκτελέστηκαν στον ίδιο χώρο από τους συγγραφείς της διπλωματικής. Τα διάφορα είδη πτώσεων εξετάστηκαν πάνω σε στρώμα, καθώς δεν ήταν εφικτό οι συνθήκες να είναι πλήρως ρεαλιστικές για ευνόητους λόγους.

4.2 Παράμετροι

Για τους δύο αλγορίθμους έχουν χρησιμοποιηθεί κάποιες παράμετροι, στις οποίες έχουμε δώσει μέσω πειραματισμού τις κατάλληλες τιμές ώστε να δίνουν τα καλύτερα δυνατά αποτελέσματα. Αυτές θα περιγράψουμε συνοπτικά παρακάτω:

- Αλγόριθμος ανίχνευσης ζώνης εσωτερικού χώρου:
 - **SCAN_TIMES**: Η συγκεκριμένη παράμετρος χρησιμοποιείται τόσο στην αρχική ρύθμιση των ζωνών όσο και στον αλγόριθμο ανίχνευσης ζώνης, και δείχνει τον αριθμό των επαναλήψεων κατά τις οποίες λαμβάνονται τιμές από την ισχύ σήματος των ασύρματων δικτύων της περιοχής προς το κινητό. Έχουμε δώσει στην παράμετρο την τιμή 10, δηλαδή σε κάθε μέτρηση για ένα σημείο λαμβάνουμε 10 συνεχόμενες μετρήσεις και παίρνουμε το μέσο όρο, για να έχουμε έτσι μεγαλύτερη ακρίβεια στη μέτρηση.
 - **SCAN_INTERVAL**: Η τιμή της συγκεκριμένης μεταβλητής είναι 2000 και δείχνει το χρονικό διάστημα σε milliseconds, ανάμεσα σε κάθε δειγματοληψία, από τις 10 που περιγράψαμε παραπάνω. Έτσι με βάση τις τιμές των δύο αυτών παραμέτρων, χρειάζεται 20 δευτερόλεπτα για κάθε εκτέλεση του αλγορίθμου ανίχνευσης ζώνης, καθώς και για την αρχική ρύθμιση των ζωνών.
 - **MAX_DB_NOISE**: Η παράμετρος αυτή χρησιμοποιείται για να φιλτράρουμε τα ασύρματα δίκτυα εκείνα, τα οποία μπορούν να θεωρηθούν ως “θόρυβος”, αφού βρίσκονται πολύ μακριά από το σημείο δειγματοληψίας. Η τιμή της ορίζεται σε -80dB.
 - **MAX_DISTANCE_FROM_ZONE**: Η συγκεκριμένη παράμετρος έχει τιμή 5dB και χρησιμοποιείται για να δώσουμε πόντους στη ζώνη “unknown”. Αν λοιπόν η πιο κοντινή ισχύς σήματος για ένα wi-fi στο σημείο δειγματοληψίας από κάποια από τις γνωστές ζώνες είναι μεγαλύτερη των 5dB, τότε θεωρούμε πως το κινητό βρίσκεται μακριά από όλες τις καταχωρημένες ζώνες, και δίνουμε έναν πόντο για το συγκεκριμένο wi-fi στη ζώνη “unknown”.
 - **MAX_DISTANCE_FOR_ADDITIONAL**: Τέλος, η συγκεκριμένη παράμετρος έχει την τιμή 1.5dB και χρησιμοποιείται για να δώσουμε πόντους και σε άλλες ζώνες, όταν μια μέτρηση δεν είναι ξεκάθαρη υπέρ μίας συγκεκριμένης ζώνης. Αν για παράδειγμα έχουμε για ένα wi-fi στο σημείο που βρισκόμαστε μια μέτρηση 50dB και το συγκεκριμένο wi-fi στην ζώνη A είχε 50.5 dB και στη ζώνη B είχε 51.5dB, τότε θα πρέπει να πάρουν πόντο και οι 2 ζώνες, αφού δεν είναι ξεκάθαρο από τη μέτρηση αν πρόκειται για τη μία ή την άλλη.

- Αλγόριθμος ανίχνευσης πτώσης
 - ο **LOWER_THRESHOLD**: Χρησιμοποιείται για την ανίχνευση της φάσης ελεύθερης πτώσης (pre-fall) και έχει τιμή $0.5 \cdot g$ (όπου $g=9.81$ η επιτάχυνση της βαρύτητας). Όταν βρεθεί επιτάχυνση κάτω από αυτό το όριο, τότε ο αλγόριθμος αναζητά τη φάση πρόσκρουσης.
 - ο **MAX_DIFFERENCE**: Η συγκεκριμένη παράμετρος μπορεί να έχει τιμή $2.5 \cdot g$, η οποία είναι η προκαθορισμένη (default) τιμή, καθώς και η τιμή όταν η ζώνη είναι η “unknown”. Σε περίπτωση που έχει υπολογιστεί ότι η συσκευή βρίσκεται σε επικίνδυνη ζώνη, το MAX_DIFFERENCE έχει τιμή $2.3 \cdot g$, ενώ σε ζώνη χαμηλού κινδύνου η τιμή ανεβαίνει στο $2.7 \cdot g$.
 - ο **MAX_TIME_AFTER_FREE_FALL**: Η τιμή αυτής της παραμέτρου (3 στην περίπτωση μας) είναι σε δευτερόλεπτα και δείχνει το χρόνο μέσα στον οποίο ο αλγόριθμος ψάνει να βρει πρόσκρουση, αφού εντοπιστεί η φάση ελεύθερης πτώσης.
 - ο **MAX_TIME_AFTER_IMPACT**: Παρόμοια με την προηγούμενη, η συγκεκριμένη παράμετρος έχει οριστεί στα 5 δευτερόλεπτα, και καθορίζει τον χρόνο μετά την πρόσκρουση, στον οποίο εξετάζουμε αν υπάρχει ανάκαμψη ή όχι του ηλικιωμένου.
 - ο **MAX_STABLE_INTERVAL**: Η συγκεκριμένη παράμετρος έχει τιμή 5 στην περίπτωση μας και καθορίζει το πόσες φορές πρέπει να ικανοποιούνται οι δύο συνθήκες που περιγράφονται στην ενότητα 3.1.1, ώστε να αποφασίσει ο αλγόριθμος για σίγουρη ή πιθανή πτώση.
 - ο **LAST_X_MINUTES**: Καθορίζει το χρόνο (σε λεπτά) μέσα στον οποίο λαμβάνονται δεδομένα για κάθε εκτέλεση του αλγορίθμου ανίχνευσης πτώσης, στην περίπτωση μας στα 2 λεπτά. Ουσιαστικά ο αλγόριθμος τρέχει κάθε λεπτό και κάθε φορά χρησιμοποιεί τα δεδομένα των τελευταίων 2 λεπτών, έτσι ώστε να ειδοποιήσει το χρήστη και μια δεύτερη φορά, στην χειρότερη περίπτωση που “έχασε” για οποιοδήποτε λόγο την αρχική ειδοποίηση.
 - ο **DEGREES_THRESHOLD**: Καθορίζει το πάνω και κάτω όριο για την γωνία φ , η οποία όπως έχουμε περιγράψει στο 3.1 θα πρέπει να μην είναι μεταξύ -35 και 35 μοιρών για να αποφασίσουμε κάποια πτώση. Έτσι η παράμετρος έχει την τιμή 35.
 - ο **ACCELERATION_THRESHOLD_AFTER_FALL**: Τέλος, δίνουμε στη συγκεκριμένη παράμετρο την τιμή 0.2 για να ελέγχουμε μετά από την πρόσκρουση αν η επιτάχυνση είναι μεταξύ $1 + \text{ACCELERATION_THRESHOLD_AFTER_FALL}$ και $1 - \text{ACCELERATION_THRESHOLD_AFTER_FALL}$, δηλαδή μεταξύ 0.8 και 1.2 m/sec^2 .

4.3 Μετρήσεις

Παρακάτω περιγράφουμε τις μετρήσεις και τα αποτελέσματα που πήραμε με βάση τα πειράματα για κάθε έναν από τους δύο αλγόριθμους.

- Αλγόριθμος ανίχνευσης ζώνης εσωτερικού χώρου

Για τον αλγόριθμο ανίχνευσης ζώνης χρησιμοποιήσαμε ένα επιπλέον κουμπί στην εφαρμογή ώστε να μπορούμε να πάρουμε μέτρηση για ζώνη όποτε θελήσουμε και προχωρήσαμε όπως περιγράψαμε παραπάνω στην λήψη μετρήσεων σε κάθε ένα από τα 50 σημεία που ορίσαμε στο χώρο του σπιτιού.

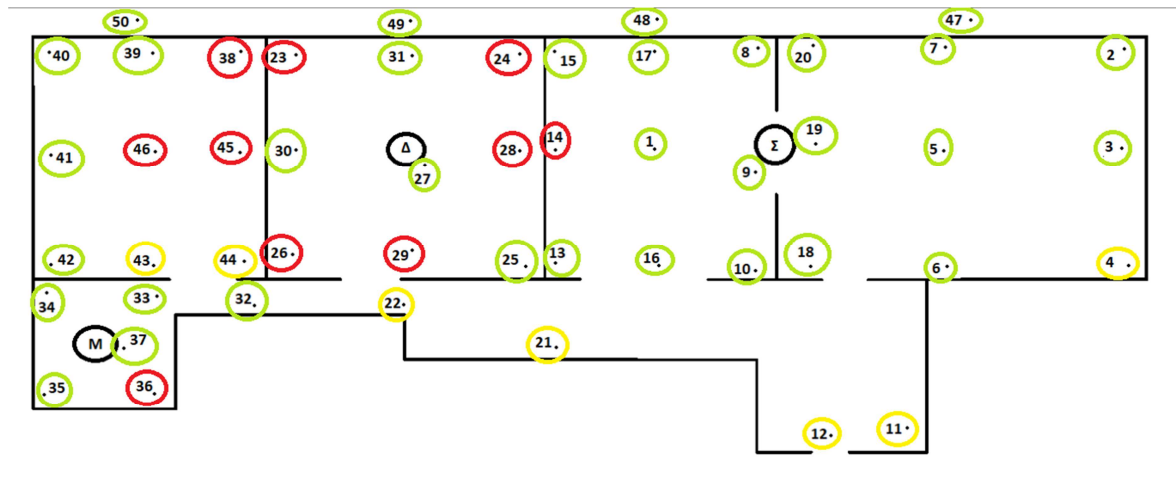
Παρακάτω, στο Πίνακα 6, βλέπουμε την εικόνα που προέκυψε από τις μετρήσεις για όλα τα σημεία, με το ποσοστό επιτυχούς αναγνώρισης ζώνης να είναι στο 76.7% αν θεωρήσουμε αδιάφορες τις μετρήσεις που έχουμε σημειώσει με κίτρινο. Οι συγκεκριμένες μετρήσεις δε μπορούν να καταχωρηθούν ως λανθασμένες, αφού η ζώνη που έχουν καταδείξει δεν είναι απόλυτα λανθασμένη, αλλά ούτε και απόλυτα σωστή. Για παράδειγμα στις ζώνες 43, 44 έχει δοθεί η τιμή M (μπάνιο) από τον αλγόριθμο, ενώ τα σημεία βρίσκονται στην είσοδο της κουζίνας, είναι όμως πολύ κοντά στο μπάνιο, οπότε “σωστά” ο αλγόριθμος τα έχει κατηγοριοποιήσει σε αυτή τη ζώνη αντί για τη ζώνη “unknown”. Με την ίδια λογική και τα σημεία 11,12,21 και 22 έχουν πάρει τιμή της ζώνης στην οποία βρίσκονται κοντά, ενώ αντίθετα το σημείο 4 βρίσκεται στο σαλόνι αλλά σε αρκετά μακρινή απόσταση από το σημείο ορισμού της ζώνης, οπότε έχει πάρει την τιμή “unknown”. Αν θεωρήσουμε τις 7 αυτές περιπτώσεις ως σωστές, τότε το ποσοστό επιτυχίας του αλγορίθμου ανεβαίνει στο 80%.

Πίνακας 5: Απεικόνιση των ζωνών που έχουν οριστεί στο χώρο

zone_id	friendly_name	is_safe	user_id
1	saloni	0	1
2	mpanio	0	1
3	domatio	1	1

Πίνακας 6: Απεικόνιση των μετρήσεων στα 50 σημεία του χώρου

id	timestamp	user_id	zone_id
1	6/4/2017 16:26	1	1
2	6/4/2017 16:27	1	1
3	6/4/2017 16:28	1	1
4	6/4/2017 16:28	1	NULL
5	6/4/2017 16:29	1	1
6	6/4/2017 16:30	1	1
7	6/4/2017 16:30	1	1
8	6/4/2017 16:31	1	1
9	6/4/2017 16:32	1	1
10	6/4/2017 16:33	1	1
11	6/4/2017 16:34	1	1
12	6/4/2017 16:35	1	1
13	6/4/2017 16:36	1	1
14	6/4/2017 16:38	1	3
15	6/4/2017 16:39	1	1
16	6/4/2017 16:40	1	1
17	6/4/2017 16:42	1	1
18	6/4/2017 16:55	1	1
19	6/4/2017 16:55	1	1
20	6/4/2017 16:56	1	1
21	6/4/2017 16:57	1	1
22	6/4/2017 16:58	1	3
23	6/4/2017 16:59	1	2
24	6/4/2017 16:59	1	1
25	6/4/2017 17:00	1	3
26	6/4/2017 17:00	1	1
27	6/4/2017 17:02	1	3
28	6/4/2017 17:02	1	1
29	6/4/2017 17:04	1	1
30	6/4/2017 17:04	1	3
31	6/4/2017 17:05	1	3
32	6/4/2017 17:08	1	NULL
33	6/4/2017 17:09	1	2
34	6/4/2017 17:09	1	2
35	6/4/2017 17:10	1	2
36	6/4/2017 17:10	1	NULL
37	6/4/2017 17:11	1	2
38	6/4/2017 17:16	1	2
39	6/4/2017 17:16	1	NULL
40	6/4/2017 17:16	1	NULL
41	6/4/2017 17:17	1	NULL
42	6/4/2017 17:17	1	NULL
43	6/4/2017 17:18	1	2
44	6/4/2017 17:18	1	2
45	6/4/2017 17:19	1	2
46	6/4/2017 17:19	1	2
47	6/4/2017 17:48	1	NULL
48	6/4/2017 17:48	1	NULL
49	6/4/2017 17:49	1	NULL
50	6/4/2017 17:49	1	NULL



Εικόνα 20: Η κάτοψη του σπιτιού με χρώματα σε κάθε σημείο

Παρατηρούμε ότι οι 10 λανθασμένες μετρήσεις είναι κυρίως σε σημεία που βρίσκονται ενδιάμεσα σε δύο ζώνες, όπως τα 14, 24, 28 ενώ στα υπόλοιπα η μέτρηση για κάποιον λόγο ήταν ασυνεπής σε σχέση με τις αρχικές μετρήσεις. Αυτό στις περισσότερες περιπτώσεις μπορεί να εξηγηθεί, καθώς με βάση τον αλγόριθμο έχουμε δώσει “προτεραιότητα” στις επικίνδυνες ζώνες και έτσι σε πιθανή ισοψηφία μεταξύ μίας ασφαλούς και μίας επικίνδυνης ζώνης, ο αλγόριθμος αποφασίζει υπέρ της επικίνδυνης. Αυτό γίνεται επειδή, όπως έχουμε εξηγήσει παραπάνω, θέλουμε όταν δεν υπάρχει σιγουριά από τον αλγόριθμο για το αν ο ηλικιωμένος βρίσκεται σε επικίνδυνη ή ασφαλή ζώνη, να θεωρούμε ότι βρίσκεται σε επικίνδυνη, έτσι ώστε να μη χάσουμε κάποια πτώση. Έτσι βλέπουμε ότι από τις 10 λανθασμένες μετρήσεις οι μισές είναι στο δωμάτιο, το οποίο είναι ασφαλής ζώνη και έτσι έχασε όλες τις ισοψηφίες με τους άλλους χώρους. Παρακάτω παραθέτουμε για κάθε μία από τις “κόκκινες” μετρήσεις μας, το πως κατανεμήθηκαν οι πόντοι στις ζώνες για τα συγκεκριμένα σημεία.

Σημείο 14: {saloni=2, domatio=5, mpanio=4}

Σημείο 23: {saloni=1, domatio=3, mpanio=4, unknown=2}

Σημείο 24: {saloni=4, domatio=4, mpanio=1, unknown=4}

Σημείο 26: {saloni=3, domatio=3, mpanio=2, unknown=1}

Σημείο 28: {saloni=3, domatio=2, mpanio=2, unknown=3}

Σημείο 29: {saloni=5, domatio=2, mpanio=3}

Σημείο 36: {mpanio=2, unknown=3}

Σημείο 38: {domatio=4, mpanio=4, unknown=3}

Σημείο 45: {saloni=1, domatio=1, mpanio=4, unknown=4}

Σημείο 46: {saloni=1, domatio=1, mpanio=4, unknown=4}

Παρατηρούμε ότι στα σημεία 24, 26, 38, 45, 46 το λανθασμένο αποτέλεσμα προκύπτει από τις ισοβαθμίες, όπως περιγράψαμε παραπάνω. Στο σημείο 14, έχει προκύψει λάθος, καθώς το σημείο βρίσκεται στα όρια μεταξύ σαλονιού και δωματίου, και μάλλον πιο κοντά

στο δωμάτιο, οπότε το δωμάτιο πήρε περισσότερους πόντους. Τα υπόλοιπα 4 σημεία έχουν δοθεί λανθασμένα περισσότεροι πόντοι σε μία ζώνη, το οποίο μπορούμε να το αποδώσουμε σε στιγμιαία λανθασμένη εικόνα στις ισχύς σήματος που έλαβε το κινητό.

- Αλγόριθμος ανίχνευσης πτώσης

Για τον αλγόριθμο ανίχνευσης πτώσης, εκτελέστηκαν 15 τεχνητές πτώσεις σε στρώμα όπως περιγράψαμε παραπάνω και συγκεκριμένα 5 από κάθε είδος πτώσης (πτώση προς τα εμπρός, προς τα πίσω και πλάγια πτώση. Επιπλέον εκτελέστηκαν 30 ενέργειες που μοιάζουν με πτώσεις καθώς και απλές δραστηριότητες της καθημερινής ζωής, συγκεκριμένα 10 απότομα καθίσματα στην καρέκλα, 10 απότομα ξαπλώματα στο κρεβάτι και 10 φορές απλό περπάτημα μέσα στο σπίτι. Να σημειώσουμε ότι όλες οι μετρήσεις πραγματοποιήθηκαν στο σαλόνι (επικίνδυνη ζώνη) έτσι ώστε να δούμε πως ανταποκρίνεται ο αλγόριθμος μας στην πιο ευαίσθητη περίπτωση.

Στους παρακάτω πίνακες παραθέτουμε τα αποτελέσματα των πειραμάτων μας:

Πίνακας 7: Αποτελέσματα από μετρήσεις ADLs

Δραστηριότητα	Επιτάχυνση ελεύθερης πτώσης (g)	Επιτάχυνση Πρόσκρουσης (g)	Πτώση
Κάθισμα #01	0.24	2.04	Δε βρέθηκε
Κάθισμα #02	0.33	2.25	Δε βρέθηκε
Κάθισμα #03	> 0.5	NA	Δε βρέθηκε
Κάθισμα #04	0.19	2.77	Βρέθηκε
Κάθισμα #05	> 0.5	NA	Δε βρέθηκε
Κάθισμα #06	0.38	1.99	Δε βρέθηκε
Κάθισμα #07	> 0.5	NA	Δε βρέθηκε
Κάθισμα #08	> 0.5	NA	Δε βρέθηκε
Κάθισμα #09	0.34	1.81	Δε βρέθηκε
Κάθισμα #10	0.41	2.59	Βρέθηκε
Ξάπλωμα #01	0.43	2.00	Δε βρέθηκε
Ξάπλωμα #02	0.30	2.69	Βρέθηκε
Ξάπλωμα #03	0.23	2.63	Βρέθηκε
Ξάπλωμα #04	0.42	1.60	Δε βρέθηκε
Ξάπλωμα #05	0.35	1.92	Δε βρέθηκε
Ξάπλωμα #06	0.28	2.25	Δε βρέθηκε
Ξάπλωμα #07	0.33	1.98	Δε βρέθηκε
Ξάπλωμα #08	0.45	2.12	Δε βρέθηκε
Ξάπλωμα #09	0.19	2.71	Βρέθηκε
Ξάπλωμα #10	> 0.5	NA	Δε βρέθηκε
Περπάτημα #01	0.47	0.91	Δε βρέθηκε

Περπάτημα #02	> 0.5	NA	Δε βρέθηκε
Περπάτημα #03	0.39	1.34	Δε βρέθηκε
Περπάτημα #04	0.44	1.69	Δε βρέθηκε
Περπάτημα #05	0.45	1.93	Δε βρέθηκε
Περπάτημα #06	0.47	1.70	Δε βρέθηκε
Περπάτημα #07	> 0.5	NA	Δε βρέθηκε
Περπάτημα #08	0.38	1.59	Δε βρέθηκε
Περπάτημα #09	0.47	1.50	Δε βρέθηκε
Περπάτημα #10	0.45	1.05	Δε βρέθηκε

Πίνακας 8: Αποτελέσματα από μετρήσεις πτώσεων

Είδη πτώσεων	Επιτάχυνση ελεύθερης πτώσης (g)	Επιτάχυνση Πρόσκρουσης (g)	Πτώση
Πλάγια #1	0.46	3.76	Βρέθηκε
Πλάγια #2	0.38	2.8	Βρέθηκε
Πλάγια #3	0.49	3.48	Βρέθηκε
Πλάγια #4	0.44	3.3	Βρέθηκε
Πλάγια #5	0.48	4.19	Βρέθηκε
Προς τα πίσω #1	0.45	2.96	Βρέθηκε
Προς τα πίσω #2	0.23	5.23	Βρέθηκε
Προς τα πίσω #3	0.49	3.45	Βρέθηκε
Προς τα πίσω #4	0.23	2.38	Βρέθηκε
Προς τα πίσω #5	0.42	1.87	Δε βρέθηκε
Προς τα εμπρός #1	0.25	2.45	Βρέθηκε
Προς τα εμπρός #2	0.49	2.70	Βρέθηκε
Προς τα εμπρός #3	0.48	5.25	Βρέθηκε
Προς τα εμπρός #4	0.27	3.15	Βρέθηκε
Προς τα εμπρός #5	0.44	3.02	Βρέθηκε

Η τιμή επιτάχυνση ελεύθερης πτώσης παραπάνω δείχνει την πρώτη τιμή που εντοπίστηκε κάτω από το όριο του 0.5*g που έχουμε καθορίσει, και είχε ως αποτέλεσμα ο αλγόριθμος να περάσει στη φάση που προσπαθεί να εντοπίσει πρόσκρουση. Στις στήλες που έχουμε ως τιμή “>0.5” σημαίνει ότι οι μετρήσεις δεν έπεσαν ποτέ κάτω από το όριο, οπότε δεν εξετάστηκαν οι επόμενες φάσεις. Η τιμή επιτάχυνση πρόσκρουσης δείχνει την τιμή της επιτάχυνσης όταν το σώμα χτύπησε στο έδαφος (είτε στην καρέκλα/κρεβάτι στην περίπτωση των υπόλοιπων δραστηριοτήτων). Τέλος, η στήλη πτώση δείχνει αν ο αλγόριθμος εντόπισε ή όχι πτώση σε κάθε ένα από τα πειράματα.

Παρατηρούμε, ότι ο αλγόριθμος εντόπισε σωστά σχεδόν όλες τις πτώσεις, ενώ στις δραστηριότητες της καθημερινής ζωής είχε κάποιες ψευδώς θετικές ενδείξεις στις περιπτώσεις του απότομου καθίσματος και ξαπλώματος, οι οποίες μοιάζουν περισσότερο στις πτώσεις. Θα πρέπει όμως να σημειωθεί ότι ακόμα και οι μετρήσεις που βγήκαν ψευδώς θετικές είναι στο όριο, αφού αν π.χ τα πειράματα γίνονταν σε ασφαλή ζώνη, οι

τρεις από τις πέντε δε θα ήταν πάνω από το όριο του 2.7 που έχουμε για αυτές τις ζώνες και έτσι δεν θα κατηγοριοποιούνταν ως πτώσεις. Οι υπόλοιπες δύο είναι κι αυτές λίγο πάνω από το όριο (2.71, 2.77). Οπότε σε γενικές γραμμές τα αποτελέσματα του αλγορίθμου θεωρούνται άκρως ικανοποιητικά γι αυτό το μικρό δείγμα το οποίο έχουμε πάρει.

Πιο συγκεκριμένα, χρησιμοποιούμε ως μετρικές αξιολόγησης, τους δείκτες ευαισθησίας (sensitivity) και εξειδίκευσης (specificity), οι οποίες χρησιμοποιούνται σε όλες σχεδόν τις αντίστοιχες εφαρμογές.

Ο δείκτης της ευαισθησίας (TPR) μας δείχνει το ποσοστό των θετικών ενδείξεων, που σωστά κατηγοριοποιήθηκαν ως θετικές και υπολογίζεται αν διαιρέσουμε τον αριθμό των αληθώς θετικών ενδείξεων (TP) με το σύνολο των θετικών ενδείξεων, δηλαδή το άθροισμα των αληθώς θετικών και των ψευδώς αρνητικών (FN):

$$TPR = \frac{TP}{TP + FN} = \frac{14}{14 + 1} = 93.3\%$$

Ο δείκτης της ευαισθησίας (SPC) μας δείχνει το ποσοστό των αρνητικών ενδείξεων, που σωστά κατηγοριοποιήθηκαν ως αρνητικές και υπολογίζεται αν διαιρέσουμε τον αριθμό των αληθώς αρνητικών μετρήσεων με (TN) με το σύνολο των αρνητικών ενδείξεων, δηλαδή το άθροισμα των αληθώς αρνητικών και των ψευδώς θετικών (FP):

$$SPC = \frac{TN}{TN + FP} = \frac{25}{25 + 5} = 83.3\%$$

5. ΣΥΜΠΕΡΑΣΜΑΤΑ

Στην παρούσα εργασία παρουσιάσαμε μια ολοκληρωμένη λύση για το πρόβλημα της ανίχνευσης πτώσης, χρησιμοποιώντας παράλληλα και έναν αλγόριθμο ανίχνευσης ζώνης εσωτερικού χώρου, με τη βοήθεια του οποίου προσαρμόζεται η ευαισθησία του πρώτου αλγορίθμου και δίνονται επιπλέον πληροφορίες για τη ζώνη στην οποία πιθανότατα συνέβη η πτώση. Τέλος επιχειρήσαμε να δώσουμε πληροφορίες για τον τρόπο που συνέβη η πτώση, εκμεταλλευόμενοι το API της Google για αναγνώριση δραστηριότητας. Από την αξιολόγηση του συστήματος, διαπιστώσαμε ότι ο αλγόριθμος μας ανιχνεύει πτώσεις σε ικανοποιητικό βαθμό, καθώς έχει και αρκετά καλά ποσοστά στην ανίχνευση της ζώνης στην οποία βρίσκεται η συσκευή μέσα στον εσωτερικό χώρο.

Ένας περιορισμός που παρατηρήσαμε κατά τη διάρκεια της αξιολόγησης σχετικά με την ανίχνευση ζώνης, είναι η αδυναμία του αλγορίθμου σε σημεία που είναι σε άκρο μίας ζώνης και κοντά σε μια άλλη. Αυτό επηρεάζει τον αλγόριθμο ειδικά αν η μία ζώνη είναι ασφαλής και η άλλη επικίνδυνη, αφού μπορεί να επικρατήσει η ασφαλής ζώνη και να μη βρεθεί τελικά σωστό αποτέλεσμα. Επίσης υπάρχει περίπτωση κάποια σημεία του χώρου να “μοιάζουν” σαν μέτρηση σε περισσότερες από μία ζώνες και να προκύψει έτσι σύγχυση στο τελικό αποτέλεσμα. Τέλος κάποιοι ασύρματοι πομποί δικτύου είναι πιθανόν να μην είναι σταθεροί στο ίδιο σημείο και να βρίσκονται σε διαφορετική θέση τη στιγμή της αρχικής ρύθμισης των ζωνών και κατά την μετέπειτα χρήση της εφαρμογής. Για αυτό το λόγο θα πρέπει οι ζώνες να ανανεώνονται ανά τακτά διαστήματα από τον υπεύθυνο χρήστη.

Σχετικά με τον αλγόριθμο ανίχνευσης πτώσης, υπάρχουν σίγουρα περιθώρια επιπλέον βελτίωσης ώστε να ξεχωρίζονται κάποιες περιπτώσεις που μοιάζουν σαν πρότυπο (pattern) με την πτώση, όπως αυτές που περιγράψαμε στην ενότητα 4.3. Αυτό θα μπορούσε ίσως να γίνει και με χρήση άλλων αισθητήρων της συσκευής εκτός από το επιταχυνσιόμετρο, καθώς και με τη χρήση άλλου είδους αλγορίθμου, είτε με τεχνικές κατωφλίου, είτε με τεχνικές μηχανικής μάθησης. Επίσης, δεν έχουμε διαχειριστεί περιπτώσεις, όπως η απλή πτώση της συσκευής, χωρίς πτώση του χρήστη, “την οποία το σύστημα θα πρέπει να διαχειρίζεται διαφορετικά. Θα μπορούσε να υλοποιηθεί κάποιος μηχανισμός για τον ηλικιωμένο χρήστη ώστε να αποτρέπει την αποστολή ειδοποίησης σε τέτοιες περιπτώσεις. Τέλος θα πρέπει να εξεταστεί και η κατανάλωση μπαταρίας και για τους δύο αλγορίθμους, καθώς είναι πολύ σημαντικό να μην χρειάζεται πολύ συχνή φόρτιση, εφόσον ο χρήστης πρέπει να έχει τη συσκευή συνεχώς πάνω του.

Σε μελλοντικά βήματα, θα μπορούσε να χρησιμοποιηθεί η τεχνολογία του smartwatch που περιγράψαμε στη βιβλιογραφία και η οποία είναι ένας σημαντικός και νέος στον χώρο παράγοντας που αναπτύσσεται συνεχώς. Η τεχνολογία αυτή θα μπορούσε να χρησιμοποιηθεί με πολλούς παραπάνω τρόπους από το κινητό, όπως π.χ. με τη λειτουργία μέτρησης παλμών, την οποία διαθέτουν οι περισσότερες συσκευές τέτοιου είδους. Τέλος, θα μπορούσε να διερευνηθεί και η ανίχνευση πτώσης σε εξωτερικό χώρο, π.χ. με την δυνατότητα αποστολής στίγματος από το σημείο που συνέβη και άμεσης κλήσης για βοήθεια.

ΠΙΝΑΚΑΣ ΟΡΟΛΟΓΙΑΣ

Ξενόγλωσσος όρος	Ελληνικός Όρος
Smartphone	Έξυπνο κινητό τηλέφωνο
Wearable	Φορετή συσκευή
Accelerometer	Επιταχυνσιόμετρο
Threshold-based Techniques	Τεχνικές κατωφλίου
Server	Εξυπηρετητής
Notification	Ειδοποίηση
Wi-Fi	Ασύρματο δίκτυο
Activity Recognition	Αναγνώριση δραστηριότητας
Ranking Method	Αλγόριθμος αυτόματης απόδοσης σειράς κατάταξης
Accuracy	Ορθότητα
Precision	Ακρίβεια
Recall	Ανάκληση
Sensitivity	Ευαισθησία
Specificity	Εξειδίκευση
Amplitude	Εύρος
Background service	Υπηρεσία παρασκηνίου
Roll	Κύλιση
Pitch	Ανύψωση
Yaw	Στροφή
Cross Validation	Διασταυρωμένη επικύρωση
Residual movement	Υπολειπόμενη κίνηση
Patterns	Πρότυπα
Lateration	Απόσταση
Angulation	Γωνία
Angle of arrival	Κλίση του λαμβανόμενου σήματος
Default	Προκαθορισμένος
Client application	Εφαρμογή πελάτη
Annotation	Υπόμνημα
Documentation	Τεκμηρίωση
Client library	Βιβλιοθήκη πελάτη
Login	Σύνδεση
Register	Εγγραφή
Slide	Κύλιση
Factory Pattern	Πρότυπο εργοστασίου
Framework	Πλαίσιο

ΣΥΝΤΜΗΣΕΙΣ – ΑΡΚΤΙΚΟΛΕΞΑ – ΑΚΡΩΝΥΜΙΑ

POJO	Plain Old Java Object
UI	User Interface
EER	Enhanced Entity–Relationship model
REST	Representational State Transfer
ORM	Object-relational mapping
RFID	Radio Frequency Identification
k-NN	k-Nearest Neighbors
MVC	Model View Controller model
GSM	Global System for Mobile communications
SVM	Support Vector Machine
ROC	Receiver operating characteristic
ADL	Activities of daily living
SMV	Signal Magnitude Vector
BMI	Body Mass Index
SMLR	Sparse Multinomial Logistic Regression
FI	Fall Index
GPS	Global Positioning System
API	Application Programming Interface
FCM	Firebase Cloud Messaging
XML	eXtensible Markup Language
JSON	JavaScript Object Notation

ΑΝΑΦΟΡΕΣ

- [1] Causes of Injuries Treated in the Emergency Department, USA, 2010
<http://www.hcup-us.ahrq.gov/reports/statbriefs/sb156.pdf>, Accessed October 10, 2016
- [2] Lim D, Park C, Kim N. H, Kim S-H, Yu YS. Fall-detection algorithm using 3-axis acceleration: combination with simple threshold and hidden Markov model. J Appl Math. 2014; 2014:1–8. doi:10.1155/2014/896030.
- [3] Casilari, E.; Luque, R.; Morón, M.-J. Analysis of android device-based solutions for fall detection. Sensors 2015, 15, 17827–17894.
- [4] S. Kozina, H. Gjoreski, M. Gams, and M. Lustrek, "Efficient activity recognition and fall detection using accelerometers, in Evaluating AAL Systems Through Competitive Benchmarking, Springer, 2013, pp. 13-23.
- [5] Delahoz, Y.S.; Labrador, M.A. Survey on Fall Detection and Fall Prevention Using Wearable and External Sensors. Sensors 2014, 14, 19806-19842.
- [6] Igual R., Medrano C, Plazalgual I., "Challenges, issues and trends in fall detection systems", BioMedical Engineering OnLine 2013 12:66.
- [7] Sposaro F, Tyson G: iFall: an Android application for fall monitoring and response. Conf Proc IEEE Eng Med Biol Soc 2009, 2009: 6119-6122.
- [8] J. Dai, X. Bai, Z. Yang, Z. Shen, and D. Xuan, PerFallD: A Pervasive Fall Detection System Using Mobile Phones, In Proc. 8th IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshops), pp. 292-297, Germany, 2010.
- [9] Kazi, S.B., Sikander, S., Yousafzai, S., Mazhar, S.: Fall detection using single tri-axial accelerometer. In: ASEE 2014 Zone I Conference (2014)
- [10] Y. Cao, Y. Yang, and W. Liu, "E-fall: A fall detection system using android-based smartphone," in Fuzzy Systems and Knowledge Discovery (FSKD), 2012 9th International Conference on. IEEE, 2012, pp. 1509–1513.
- [11] Albert MV, Kording K, Herrmann M, Jayaraman A (2012) Fall Classification by Machine Learning Using Mobile Phones. PLoS ONE 7(5): e36556. doi:10.1371/journal.pone.0036556
- [12] Stefano Abbate, Marco Avvenuti, Francesco Bonatesta, Guglielmo Cola, Paolo Corsini, and Alessio Vecchio. 2012. A smartphone-based fall detection system. Pervasive Mob. Comput. 8, 6 (December 2012), 883-899. DOI=<http://dx.doi.org/10.1016/j.pmcj.2012.08.003>
- [13] E. Casilari and M. A. Oviedo-Jiménez, "Automatic fall detection system based on the combined use of a smartphone and a smartwatch," PLoS One, vol. 10, no. 11, p. e0140929, 2015.
- [14] P. Kostopoulos, T. Nunes, K. Salvi, M. Deriaz, and J. Torrent, "Increased fall detection accuracy in an accelerometer-based algorithm considering residual movement," in International Conference on Pattern Recognition Applications and Methods, 2015.
- [15] Lei Yu, Mohamed Laaraiedh, Stephane Avrillon, Bernard Uguen. Fingerprinting localization based on neural networks and ultra-wideband signals. Signal Processing and Information Technology (ISSPIT), 2011 IEEE International Symposium on, Dec 2011, Bilbao, Spain. pp.184 -189, 2011, <10.1109/ISSPIT.2011.6151557>.
- [16] Ben-Moshe, H. Levi, P. Shamil, and A. Dvir. Indoor Positioning using Time of Flight Fingerprinting of Ultrasonic Signals. In Proceedings of the 2013 International Conference on Indoor Positioning and Indoor Navigation (IPIN2013), 2013.

- [17] Bahl, Paramvir, and Venkata N. Padmanabhan. "RADAR: An in-building RF-based user location and tracking system." INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE. Vol. 2. IEEE, 2000.
- [18] Gong Chen, Qiang Liu, Yunkai Wei, Qin Yu, "An efficient indoor location system in WLAN based on Database Partition and Euclidean Distance-Weighted Pearson Correlation Coefficient", Computer and Communications (ICCC) 2016 2nd IEEE International Conference on, pp. 1736-1741, 2016.
- [19] Chen, L.; Pei, L.; Kuusniemi, H.; Chen, Y.; Kröger, T.; Chen, R. Bayesian fusion for indoor positioning using bluetooth fingerprints. *Wirel. Pers. Commun.* 2012, 67, 1–11.
- [20] E. Mok and B. K. S. Cheung, "An improved neural network training algorithm for Wi-Fi fingerprinting positioning," *ISPRS International Journal of Geo-Information*, vol. 2, no. 3, pp. 854– 868, 2013.
- [21] Abdou, A.S.; Aziem, M.A.; Aboshosha, A. An efficient indoor localization system based on Affinity Propagation and Support Vector Regression. In *Proceedings of the 2016 Sixth International Conference on Digital Information Processing and Communications (ICDIPC)*, Beirut, Lebanon, 21–23 April 2016
- [22] Yang, Chouchang, and Huai-Rong Shao. "WiFi-based indoor positioning." *IEEE Communications Magazine* 53.3 (2015): 150-157.
- [23] He Wang, Souvik Sen, Ahmed Elgohary, Moustafa Farid, Moustafa Youssef, and Romit Roy Choudhury. 2012. No need to war-drive: unsupervised indoor localization. In *Proceedings of the 10th international conference on Mobile systems, applications, and services (MobiSys '12)*. ACM, New York, NY, USA, 197-210. DOI=<http://dx.doi.org/10.1145/2307636.2307655>
- [24] Yin Chen, Dimitrios Lymberopoulos, Jie Liu, and Bodhi Priyantha. 2012. FM-based indoor localization. In *Proceedings of the 10th international conference on Mobile systems, applications, and services (MobiSys '12)*. ACM, New York, NY, USA, 169-182. DOI=<http://dx.doi.org/10.1145/2307636.2307653>
- [25] Varshavsky, Alex, et al. "GSM indoor localization." *Pervasive and Mobile Computing* 3.6 (2007): 698-720.
- [26] Stephen P. Tarzia, Peter A. Dinda, Robert P. Dick, and Gokhan Memik. 2011. Indoor localization without infrastructure using the acoustic background spectrum. In *Proceedings of the 9th international conference on Mobile systems, applications, and services (MobiSys '11)*. ACM, New York, NY, USA, 155-168. DOI=<http://dx.doi.org/10.1145/1999995.2000011>
- [27] Zou, Han, et al. "An RFID indoor positioning system by using weighted path loss and extreme learning machine." *Cyber-Physical Systems, Networks, and Applications (CPSNA)*, 2013 IEEE 1st International Conference on. IEEE, 2013.
- [28] Yucel, Hikmet, et al. "Development of indoor positioning system with ultrasonic and infrared signals." *Innovations in Intelligent Systems and Applications (INISTA)*, 2012 International Symposium on. IEEE, 2012.
- [29] Medina, Carlos, José Carlos Segura, and Angel De la Torre. "Ultrasound indoor positioning system based on a low-power wireless sensor network providing sub-centimeter accuracy." *Sensors* 13.3 (2013): 3501-3526.
- [30] Lin, Xin-Yu, et al. "A mobile indoor positioning system based on iBeacon technology." *Engineering in Medicine and Biology Society (EMBC)*, 2015 37th Annual International Conference of the IEEE. IEEE, 2015.
- [31] L. Li, P. Hu, C. Peng, G. Shen, and F. Zhao. Epsilon: A Visible Light Based Positioning System. In *11th USENIX Symposium on Networked Systems Design and Implementation (NSDI 14)*, pages 331–343, Seattle, WA, Apr. 2014. USENIX Association.
- [32] Dimitrios Lymberopoulos, Jie Liu, Xue Yang, Romit Roy Choudhury, Vlado Handziski, and Souvik Sen. 2015. A realistic evaluation and comparison of indoor location technologies: experiences and lessons learned. In *Proceedings of the 14th International Conference on Information Processing in Sensor Networks (IPSN '15)*. ACM, New York, NY, USA, 178-189 <http://dx.doi.org/10.1145/2737095.2737726>

- [33] C. Beder and M. Klepal. Fingerprinting based localisation revisited - a rigorous approach for comparing RSSI measurements coping with missed access points and differing antenna attenuations. In 2012 International Conference on Indoor Positioning and Indoor Navigation (IPIN), 2012.
- [34] R. Reimann, A. Bestmann, and M. Ernst. Locating Technology for AAL Applications with Direction Finding and Distance Measurement by Narrow Bandwidth Phase Analysis. In Evaluating AAL Systems Through Competitive Benchmarking, volume 362 of Communications in Computer and Information Science, pages 52–62. Springer Berlin Heidelberg, 2013.
- [35] <https://projects.spring.io/spring-framework/>, τελευταία πρόσβαση 4/2017
- [36] <https://projects.spring.io/spring-boot/>, τελευταία πρόσβαση 4/2017
- [37] <https://tools.ietf.org/html/rfc6690>, τελευταία πρόσβαση 4/2017
- [38] <http://swagger.io/>, τελευταία πρόσβαση 4/2017
- [39] <http://hibernate.org/orm/>, τελευταία πρόσβαση 4/2017
- [40] <https://gradle.org/>, τελευταία πρόσβαση 4/2017
- [41] <https://developers.google.com/android/reference/com/google/android/gms/location/ActivityRecognitionApi>, accessed 18/6/2017
- [42] <https://firebase.google.com/docs/cloud-messaging>, accessed 18/6/2017
- [43] <http://square.github.io/retrofit/>, accessed 18/6/2017
- [44] <https://github.com/google/volley>, accessed 25/6/2017
- [45] <https://docs.spring.io/spring/docs/current/javadoc-api/org.springframework.web.client/RestTemplate.html>, accessed 25/6/2017
- [46] <https://github.com/swagger-api/swagger-codegen>