UNIVERZITA KOMENSKÉHO V BRATISLAVE
FAKULTA MATEMATIKY, FYZIKY A INFORMATIKY

LOKALIZÁCIA POMOCOU WIFI SIGNÁLU

Bakalárska práca

2013

Andrej Krajči

UNIVERZITA KOMENSKÉHO V BRATISLAVE
FAKULTA MATEMATIKY, FYZIKY A INFORMATIKY

LOKALIZÁCIA POMOCOU WIFI SIGNÁLU

Bakalárska práca

Študijný program: Informatika
Študijný odbor: 2508 Informatika
Školiace pracovisko: Katedra informatiky
Školiteľ: RNDr. Michal Forišek PhD.

Bratislava, 2013

Andrej Krajči

Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

ZADANIE ZÁVEREČNEJ PRÁCE

Meno a priezvisko študenta: Andrej Krajči
Študijný program: informatika (Jednoodborové štúdium, bakalársky I. st., denná forma)
Študijný odbor: 9.2.1. informatika
Typ záverečnej práce: bakalárska
Jazyk záverečnej práce: slovenský

Názov: Lokalizácia pomocou WiFi signálu

Cieľ: Cieľom práce je preskúmať možné metódy lokalizácie používateľa mobilného zariadenia pomocou intenzity WiFi signálov a vyhodnotiť tieto metódy z hľadiska dosiahnuteľnej presnosti. Súčasťou práce by malo byť zozbieranie dát o zmene intenzity WiFi signálu v rámci nejakej lokality a následná analýza týchto dát s cieľom identifikovania vhodných metód lokalizácie. Výstupmi práce by mali byť samotný algoritmus lokalizácie a jeho implementácia v podobe mobilnej aplikácie na platformu Android.

Vedúci: RNDr. Michal Forišek, PhD.
Katedra: FMFI.KI - Katedra informatiky
Vedúci katedry: doc. RNDr. Daniel Olejár, PhD.

Dátum zadania: 29.10.2012

Dátum schválenia: 30.10.2012 doc. RNDr. Daniel Olejár, PhD.
garant študijného programu

.....
študent

.....
vedúci práce

Poďakovanie

Chcem sa poďakovať vedúcemu práce RNDr. Michalovi Forišekovi, PhD., za cenné rady, odbornú pomoc a konzultácie, ktoré mi poskytol pri vypracovaní bakalárskej práce. Osobitné poďakovanie patrí mojej rodine a priateľom za podporu.

Abstrakt

Táto práca sa venuje problému lokalizácie zariadenia pomocou WiFi signálu. V práci popisujeme existujúce riešenia lokalizácie využívajúce WiFi signál, rôzne prístupy ku spracovávaniu dostupných dát z prijímača a ich použitie v lokalizácií. Ďalej sa zaoberáme šírením signálu v uzavretých priestoroch, problémami s tým spojenými a následným dopadom na algoritmy počítajúce lokáciu. Výsledkom je aplikácia na mobilný operačný systém Android, v ktorej sú použité dva rôzne algoritmy na vypočítanie polohy zariadenia. V algoritmoch používam lineárnu interpoláciu, barycentrické koordináty a Delaunayovu trianguláciu na dopočítanie údajov pre presnú lokalizáciu. V práci navrhujem viaceré rozšírenia a pokračovania aplikácie.

Kľúčové slová: Lokalizácia, WiFi, Delaunayova triangulácia, Android, Barycentrické koordináty, lineárna interpolácia, šírenie WiFi signálu

Abstract

This bachelor thesis addresses the problem of locating devices using WiFi signal. In this bachelor thesis we describe the existing solutions for localization using WiFi signal, different approaches to the processing of data from WiFi receiver and their usage in the process of localization. In this thesis we are describing the spreading of a WiFi signal in confined areas, related problems and the impact on algorithms estimating the location of a given device. The result of our research is an application for Android mobile operating system, which uses two different algorithms for estimating the location of a given device. The algorithms use linear interpolation, barycentric coordinates and Delaunay triangulation to calculate all necessary data for an accurate localization. This thesis proposes a number of possible extensions and continuations for the application.

Keywords: Localization, WiFi, Delaunay triangulation, Android, Barycentric coordinates, linear interpolation, WiFi signal spreading

Obsah

Úvod

Časy, keď sme sa snažili určiť našu polohu v meste pomocou mapy a nášho orientačného zmyslu už pomaly doznievajú. V dnešnej dobe už majú aj najlacnejšie mobilné telefóny zabudovaný GPS prijímač, ktorý dokáže držiteľa takéhoto zariadenia lokalizovať na mape sveta s presnosťou pár metrov v okamihu pár sekúnd. Toto všetko je pravdaže možné iba za predpokladu, že GPS prijímač má nerušený prístup ku satelitom, pomocou ktorých je určovanie pozície vykonávané. Túto podmienku nie je ťažké splniť pri jazde autom po diaľnici, pri prechádzke mestom alebo pri iných činnostiach, pri ktorých zariadenie nemá zaclonený „výhľad“ na oblohu, na ktorej sa nachádzajú satelity GPS systému.Kvalita GPS signálu však rapídne klesá v uzavretých priestoroch, čím robí lokalizáciu v budove pomocou GPS signálu takpovediac nemožnou.

Jedným z možných riešení nedostatočnej kvality GPS signálu v uzavretých priestoroch sú takzvané GPS opakovače. Tieto zariadenia prijímajú GPS signál pomocou antény umiestnenej na budove a následne vysielajú zachytený signál do vybraného priestoru, čím zvyšujú intenzitu signálu dostupného v krytých budovách. (ROGER-GPS Oy). Výhodou tohto riešenia je používanie už existujúcej technológie, čím je koncový užívateľ odbremenený od inštalovania aplikácií tretích strán. Nevýhodou je jeho finančná náročnosť a nepraktickosť na účely presnej lokalizácie, nakoľko každý takýto opakovač vysiela do svojho okolia informáciu iba o svojej polohe. Aby sme dosiahli požadovanú funkcionalitu lokalizácie, bolo by potrebné mať takýto opakovač nasadený po celom interiéri v pravidelných intervaloch. Aj po splnení tejto podmienky by však presnosť bola čisto diskrétna – zariadenie na lokalizáciu by svoju polohu vždy videlo na mieste najbližšej antény, takže počet možných pozícií v budove by bol priamo úmerný počtu nasadených GPS opakovačov. Signál z GPS opakovačov by sa naviac medzi sebou navzájom rušil, čo by ešte viac znepresňovalo výpočet lokácie zariadenia.

Jednoznačne lacnejším a praktickejším riešením je teda použitie signálu WiFi na lokalizáciu v uzavretých priestoroch, kde je intenzita GPS signálu nedostatočná na presné určenie pozície a nasadenie GPS opakovačov nie je žiadúce. Výhodou tohto riešenia je, že na lokalizáciu pomocou WiFi signálu sa dajú použiť už existujúce WiFi Access Pointy,

čím sa znižuje hardvérová investícia na implementáciu. Užívateľ dokonca nemusí mať ani prístupové údaje k jednotlivým dostupným prístupovým bodom, nakoľko sa dajú na lokalizáciu využívať voľne dostupné údaje o WiFi sieťach, ktoré sú vysielané každým prístupovým bodom (za predpokladu, že WiFi sieť nemá explicitne nastavené skrytie svojho SSID, čím sa stáva pre zariadenie nedetekovateľnou).

Tejto technológií sa už začínajú venovať aj väčšie spoločnosti ako napríklad Google, ktorá už sprístupnila verejnosti rozhranie, pomocou ktorého sa dá užívateľ na základe dostupných WiFi prístupových bodov a iných, bližšie nešpecifikovaných údajov, lokalizovať s presnosťou približne 200 metrov. Rozhranie využíva centrálnu databázu bodov, ktorým bola pridelená geografická lokácia. (DSL.sk, 2008)

V tejto práci sa budeme venovať výhodám a nevýhodám implementácie lokalizovania najmä v uzavretých priestoroch pomocou WiFi signálu z dostupných prístupových bodov. Nadobudnuté znalosti chceme následne použiť v ukážkovej aplikácii pre mobilný operačný systém Android. Táto aplikácia by mala dokázať po predchádzajúcej kalibrácií vedieť presne lokalizovať užívateľa v uzavretej budove a byť náhradou za lokalizáciu pomocou GPS signálu práve v takýchto podmienkach.

1. Dostupné riešenia

Nakoľko presnosť GPS signálu dosť kolíše v zastavaných oblastiach a vnútrajšie priestory budov robia prijímanie GPS signálu nemožným, viacero spoločností sa už začalo zaoberať myšlienkou využitia WiFi signálu na spresnenie lokalizácie zariadenia, či už v budovách alebo zastavaných priestranstvách.

1.1. A-GPS (Assisted GPS)

Jedným z najrozšírenejších riešení je v dnešnej dobe Assisted GPS, ktoré sa nachádza vo väčšine moderných smartfónov. Toto riešenie stále využíva na lokalizáciu signál z GPS satelitov, používa mobilnú sieť na spresnenie a zrýchlenie lokalizácie zariadenia v nepriaznivých podmienkach. Dostupnosť mobilnej siete a pripojenia na internet sa v tomto prípade využíva viacerými spôsobmi:

a) Zariadenie si stiahne dáta o pozícií satelitov zo serveru dostupného na internete (AGPS server), čo je často markantne rýchlejšie ako si sťahovať tieto údaje z GPS satelitov.
b) Zariadenie si vie zistiť svoju približnú polohu pomocou informácií dostupných z BTS (base transciever station, vysielač mobilnej siete) a zoznamu dostupných WiFi sietí.

Vďaka týmto dvom spôsobom je chytenie správneho signálu GPS mnohokrát rýchlejšie ako v prípade samotného GPS bez asistencie.

1.2. Apple iOS

Spoločnosť Apple začala vo svojom operačnom systéme iOS ponúkať vývojárom API (application programming interface, rozhranie pre programovanie aplikácií) na prístup ku lokalizačným službám, v ktorom kombinuje viacero spôsobov lokalizácie do jedného API, do ktorého programátor iba zadá požadovanú presnosť. Ak je požadovaná presnosť pár kilometrov, ako napríklad v prípade aplikácií na predpoveď počasia, na vypočítanie

lokácie sa použijú iba údaje o dostupných WiFi sieťach, ich lokáciach a údaje od vysielačov mobilného signálu. V opačnom prípade sa použijú aj informácie od GPS a GLONASS satelitov. Ku programátorom sa následne dostanú iba údaje o odhadovanej lokácií a presnosti tohto odhadu. Na určenie lokácie sú však použité dáta z GPS a GLONASS satelitov, informácií z BTS od operátorov mobilných sietí a dostupných WiFi sietí. V prípade, že užívateľ nechytá dostatočný GPS/GLONASS signál, na určenie pozície sa použijú údaje o WiFi sieťach z centrálnej databázy, ktorá je budovaná užívateľmi zariadení s operačným systémom iOS. Ak má totiž užívateľ zariadenia s iOS zapnuté lokalizačné služby, jeho zariadenie periodicky odosiela informácie o jeho polohe a dostupných WiFi sieťach, ktoré sú následne použité na budovanie centrálnej databázy o WiFi hotspotocha pozícií vysielačov mobilnej siete. (Apple, 2012)

1.3. Google Gears / Android

Spoločnosť Google vo svojom operačnom systéme Android zvolila opačný prístup. Programátori si môžu vybrať, ktorý z dvoch dostupných systémov použijú:

- GPS_PROVIDER používa na určenie lokácie GPS
- NETWORK_PROVIDER používa na určenie lokácie dostupné WiFi siete, údaje z vysielačov mobilných sietí.

Ak chce užívateľ zistiť svoju pozíciu pomocou dostupných WiFi sietí, Google použije svoju centrálnu databázu WiFi sietí, ktorá je budovaná užívateľmi a Google autami, ktoré zbierajú údaje do Google Street View. Na základe verejne dostupných informácií vieme, že spoločnosť Google pri prejazde mestami týmito špeciálne upravenými autami zbierala celé packety komunikácie cez WiFi, z ktorých následne extrahovala MAC adresu, intenzitu signálu a pomocou GPS zariadenia vo vozidle pridelila presnú lokáciu danému meraniu (Google, 2010).

Ako som už spomenul v úvode tejto práce, Google poskytuje nepriamy prístup ku týmto údajom aj vo svojej službe Google Gears, kde spomína odhadovanú presnosť 200m pri lokalizovaní zariadenia. Programátori môžu s pomocou tejto služby lokalizovať všetky zariadenia pristupujúce na ich server, a to aj zariadenia, ktoré by sa inak lokalizovať nedali, ako napríklad notebooky, ktoré nemajú integrovaný GPS/GLONASS prijímač. (DSL.sk, 2008)

2. Kalibrácia systému

Hlavným základom dobrej a hlavne presnej lokalizácie pomocou WiFi signálu je kvalitná kalibrácia. Počas výskumu sme dospeli ku dvom odlišným algoritmom, pomocou ktorých sa dá lokalizácia pomocou WiFi signálu uskutočniť, oba algoritmy však nedokázali spoľahlivo a presne pracovať bez dostatočnej kalibrácie pred samotným meraním.

Počas kalibrácie si systém zaznačí užívateľovu pozíciu na mape, ktorú mu užívateľ povie. Pre tento bod si potom daná aplikácia uloží do svojej internej databázy všetky dostupné WiFi prístupové body. Pre tento účel sa do databázy ukladajú štyri údaje, ktoré vysiela každý WiFi prístupový bod verejne, bez nutnosti znalosti prístupových údajov ku sieti. Tieto údaje sú nasledovné:

i) SSID – Service Set Identification
ii) BSSID – Basic Service Set Identification
iii) Sila signálu v dB.
iv) Pozícia bodu (údaj vložený užívateľom)

Každá WiFi sieť vysiela aj SSID aj BSSID. SSID je zväčša názov siete v ľahko čitateľnom tvare, ako napr. „Eduroam“ alebo „FMFI UK“. Toto SSID môže byť rovnaké aj pre viacero prístupových bodov, najmä ak sú tieto prístupové body spolu prepojené do jednej siete, ako je to napríklad v prípade WiFi sietí „Eduroam“ alebo „FMFI UK“. BSSID na druhej strane slúži ako unikátny identifikátor pre každý jeden vysielač. Nakoľko by nemali mať dva vysielače rovnaké BSSID, tento údaj je zväčša reprezentovaný MAC adresou zariadenia.

2.1. Charakteristika šírenia WiFi signálu

Jedným z hlavných problémov lokalizácie iba pomocou WiFi signálu je jeho nerovnomerné šírenie v priestore. Hoci WiFi signál prechádza aj cez steny a iné prekážky, každá takáto prekážka signál výrazne oslabuje, čo vo viacerých prípadoch nepriaznivo ovplyvňuje presnosť lokalizácie. Ďalším zavádzajúcim faktorom je odrážanie signálu od

rôznych povrchov. V bežnom živote sa nám môže kľudne stať, že vo vzdialenejšom bode od vysielača bude signál silnejší ako v bode nachádzajúcim sa bližšie, nakoľko sa signál od prístupových bodov nešíri priamočiaro, ale môže sa odrážať od okolitých objektov. Na ilustráciu problému som pomocou miernej úpravy algoritmu na interpoláciu, popísaného v ďalších kapitolách, vygeneroval tepelné mapy intenzity signálu.

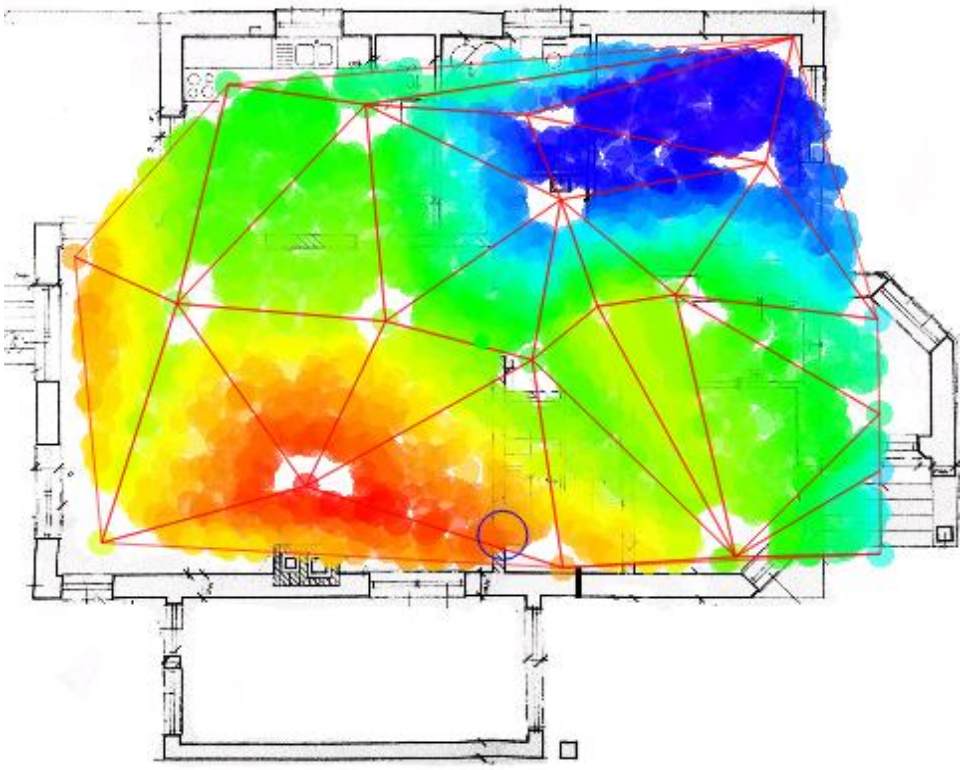

Diagram 1 - Tepelná mapa intenzity signálu v uzavretých priestoroch. Červená farba značí najsilnejší signál, modrá najslabší

Ako možno vidieť na Diagrame 1, WiFi signál z prístupového bodu umiestneného v ľavej spodnej časti mapy (najčervenší región) sa nešíri priamočiaro, čo je dobre ilustrované na pravom hornom rohu tepelnej mapy, kde signál rapídne stratí na intenzite po styku s múrom, čím vzniká modrý štvorcový útvar – izba. Ďalej vidno na mape pár miest, kde signál vzďaľujúc sa od zdroja strácal na intenzite, no po pár metroch opäť zosilnel, čo bolo pravdepodobne zapríčinené odrazom signálu od iných objektov.

Z týchto dôvodov nie je lokalizácia pomocou WiFi signálu v uzavretých priestoroch rovnako presná ako iné spôsoby určovania lokácie v otvorených

priestranstvách, nakoľko tieto metódy (ako napríklad GPS alebo GLONASS) nie sú tak veľmi ovplyvňované štruktúrou, zastavaním alebo zaľudnením priestranstva.

3. Lokalizácia

Spôsobov, ktorými sa dá uskutočniť lokalizácia zariadenia pomocou WiFi signálu je viacero, v tejto práci sme sa rozhodli podrobne popísať a analyzovať nasledovné tri algoritmy:

- Lokalizácia zariadenia na základe odhadnutia vzdialenosti od vysielača pomocou intenzity signálu siete
- nájdenie najpodobnejšieho bodu s následnou interpoláciou hodnôt bodov v okolí
- nájdenie najpodobnejšieho bodu vo vopred interpolovanej mape meraní

Prvý algoritmus používa databázu prístupových bodov a ich geografickú lokáciu. Nakoľko môže jeden vysielač vysielať viacero WiFi sietí, v databáze je pre každý prístupový bod uložený zoznam sietí, ktoré vysiela.

Druhý a tretí algoritmus využíva databázu kalibračných bodov, ktorá bola popísaná v predchádzajúcej kapitole.

3.1. Lokalizácia zariadenia na základe odhadnutia vzdialenosti od vysielača pomocou intenzity signálu siete

Ako prvý algoritmus lokalizácie budeme popisovať trianguláciu medzi prístupovými bodmi pomocou intenzity ich sietí. Zariadenie pri výpočtoch nepoužíva databázu kalibračných bodov, ale iba zoznam jednotlivých prístupových bodov, sietí ktoré vysielajú a ich geografickú lokáciu na mape. Na podobnom princípe funguje lokalizácia mobilných telefónov, ktorú používajú mobilní operátor v núdzových situáciách, ale aj napríklad pri sledovaní pohybu daného zariadenia / osoby.

3.1.1. Fungovanie algoritmu

Hľadané mobilné zariadenie si na základe intenzity GSM signálu z okolitých vysielačov vypočíta odhadovanú vzdialenosť od každého prijímača. Nakoľko má systém

(mobilný operátor) údaje o geografickej lokácii každého vysielača, vie si pozíciu mobilného telefónu ľahko dopočítať. Takáto lokalizácia sa dnes používa v napríklad v núdzových situáciách, alebo napríklad pri sledovaní pohybu nejakej osoby.

V prípade WiFi vysielačov sa algoritmus od toho použitého pri GSM triangulácií moc nelíši. Mobilný telefón si zistí intenzitu dostupných WiFi sietí a na základe svojej internej databázy prístupových bodov má údaje aj o ich pozícií v budove. Následne sa algoritmus snaží nájsť taký bod, ktorý spĺňa všetky podmienky – odhadované vzdialenosti od jednotlivých prístupových bodov. Plocha na mape, ktorá tieto podmienky spĺňa, by mala byť pozícia hľadaného zariadenia. Takýto výpočet pomocou kružníc je ilustrovaný v Diagrame 2. Jednotlivé kružnice predstavujú úroveň, na ktorej by sa zariadenie mohlo nachádzať, ak signál prístupového bodu má danú intenzitu. Zelenou farbou je zvýraznená plocha, na ktorej by sa zariadenie malo nachádzať, nakoľko leží vo všetkých kruhoch.

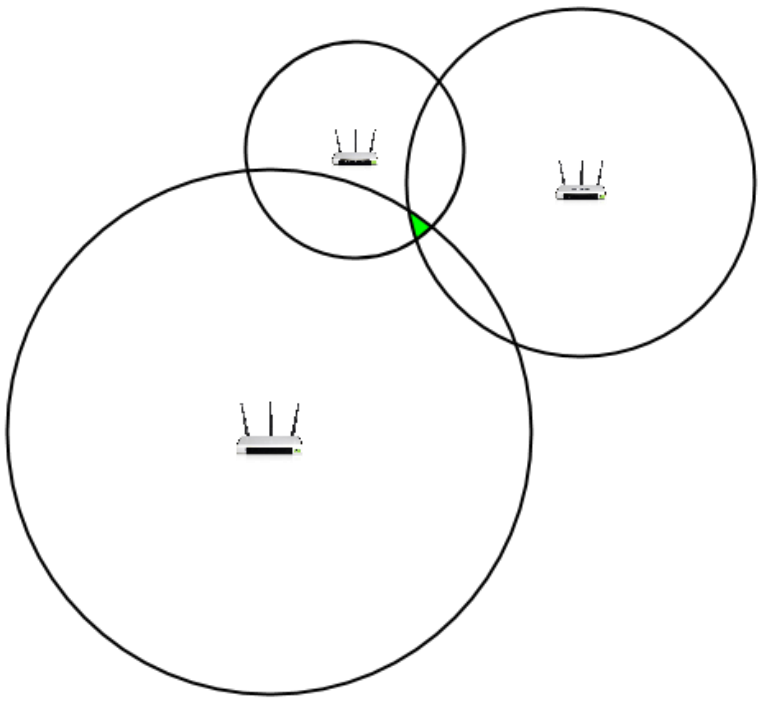

Diagram 2 – Odhadnutie vzdialenosti od vysielačov v otvorenom priestranstve

3.1.2. Výhody a nevýhody algoritmu

Tento algoritmus na lokalizáciu pomocou WiFi signálu by fungoval relatívne presne v otvorených priestranstvách, čoho dôkazom je aj dodnes využívaná jeho implementácia pri GSM triangulácii. V uzavretých priestoroch sa však signál nešíri

rovnomerne (ako bolo demonštrované v predchádzajúcej kapitole), ale jeho šírenie je ovplyvnené rôznymi faktormi, ako napríklad zaľudnenie priestorov, steny, dvere, kovové objekty a iné. Z tohto dôvodu je priamočiara triangulácia pomocou rátania vzdialenosti od vysielačov nepoužiteľná, čo je ilustrované na Diagrame 3.

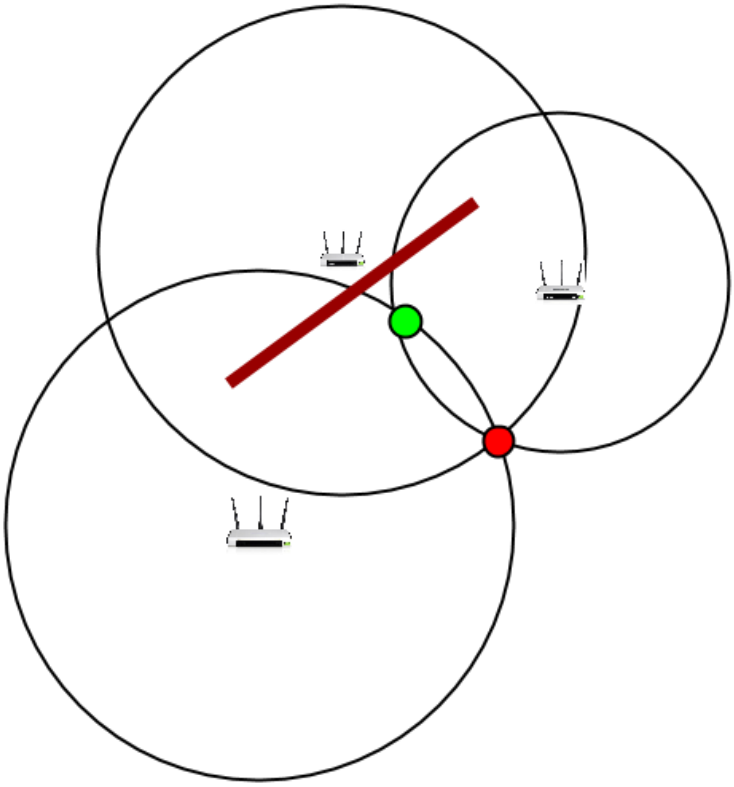

Diagram 3 - Odhadnutie vzdialenosti od vysielačov s prekážkou

Ako ilustračný príklad sme sa rozhodli vložiť prekážku (stenu) medzi jeden z vysielačov a zariadenie, ktoré chceme lokalizovať. Nakoľko má signál šíriaci sa z vysielaču v ceste prekážku, nameraná intenzita na zariadení bude menšia ako v prípade bez prekážky, hoci sa vysielače ani lokalizované zariadenie nepohlo (pozícia zariadenia je v bode označenom zelenou). Algoritmus teda odhadne vzdialenosť od prístupového bodu väčšiu ako v skutočnosti je, čím sa vypočítaná možná pozícia zariadenia presunie do bodu označenom červenou, kde sa kružnice vzdialeností od vysielačov pretínajú.

Jednou z výhod tohto algoritmu je jeho ľahká implementácia a nezávislosť od zariadenia, na ktorom prebieha výpočet na lokalizáciu užívateľa, nakoľko ku fungovaniu algoritmu stačí iba zoznam aktívnych prístupových bodov a ich lokácia. Tento algoritmus je navyše odolný voči zmenám v rozložení prístupových bodov, nakoľko stačí

administrátorovi siete vydať nový zoznam prístupových bodov a ich geografických pozícií. Tento prístup je v porovnaní s databázou kalibračných bodov značne jednoduchší, keďže pri akejkoľvek zmene v konštelácií prístupových bodov je odporúčané celú databázu nanovo namerať.

Hoci je tento algoritmus ľahko využiteľný v otvorených priestranstvách, z dôvodov vymenovaných vyššie je nevhodný na použitie v uzavretých priestranstvách, čím sa stáva nepoužiteľným v aplikácii na lokalizáciu zariadení v budovách pomocou WiFi signálu.

3.2. Lokalizácia zariadenie pomocou nájdenia najpodobnejšieho kalibračného bodu

Jedným z algoritmov, ktoré sú implementované v ukážkovej aplikácii pre operačný systém Android, je lokalizácia pomocou nájdenia najpodobnejšieho reálne nameraného bodu pri kalibrácií a následná interpolácia bodov v okolí tohto referenčného bodu, za účelom získania presnejšieho odhadu lokácie zariadenia. Zariadenie má na rozdiel od prvého algoritmu uloženú iba databázu kalibračných bodov, ktorá bola popísaná v predošlej kapitole. Algoritmus o pozícií jednotlivých prístupových bodov nevie nič a nepoužíva ich pri výpočtoch.

3.2.1. Fungovanie algoritmu

Hľadané mobilné zariadenie si pri požiadavke o výpočet lokácie urobí meranie na aktuálnej pozícii, ktoré následne porovná so všetkými bodmi merania, ktoré má uložené v kalibračnej databáze. Porovnanie bodov funguje nasledovne:

- Ak sa prístupový bod nachádza iba na aktuálnej pozícii a v kalibračnom bode sa prístupový bod s danou MAC adresou (BSSID prístupového bodu) nenachádza, algoritmus s daným prístupovým bodom počíta ako s hraničným prípadom detekovateľnosti, takže odchýlka v intenzite signálov sietí sa vyhodnotí ako maximálna možná, a to konkrétne 95 dBm. Väčšina smartfónov totiž nevie detekovať WiFi prístupové body s intenzitou signálu menšou ako -95dBm, preto aj v tomto algoritme budem považovať -95dBm ako hraničnú hodnotu

intenzity signálu, kedy je sieť detekovaná zariadením. Úroveň, za ktorou už zariadenie danú sieť nevie nájsť, sa pravdaže líši od zariadenia k zariadeniu a je ovplyvňované citlivosťou prijímacieho modulu WiFi v zariadení a aj samotnou konštrukciou zariadenia, na ktorom je aplikácia spustená.

- Ak sa prístupový bod nachádza iba v kalibračnom bode a zariadenie ju na aktuálnej pozícii zariadenia nechytá, odchýlka sa rovnako ako v predchádzajúcom prípade zaznačí ako maximálna možná, a to 95 dBm.
- Ak sa prístupový bod nachádza aj v kalibračnom bode a aj na aktuálnej pozícii zariadenia, algoritmus vypočíta rozdiel v intenzite z dostupných meraní.

Po prejdení a následnom porovnaní všetkých prístupových bodov, ktoré zariadenie chytá na aktuálnej pozícií a v kalibračnom bode, sa vypočíta priemerná odchýlka v intenzite signálov ako priemer všetkých odchýlok pre dané miesto na mape. Ukážkové porovnanie jedného bodu je v tabuľke nižšie:

Aktuálna pozícia	**Rozdiel v intenzite**	**Kalibračný bod**
Eduroam	10 dBm	Eduroam
FMFI	15 dBm	FMFI
NovaWifi	95 dBm	-----
-----	95 dBm	ZrusenaWifi

Celkový rozdiel v intenzite signálov je 215 dBm, pričom priemerná odchýlka je 53.75 dBm. Spoločné prístupové body pre aktuálnu pozíciu a kalibračný bod sú siete *FMFI* a *Eduroam*, sieť *NovaWifi* je nameraná iba na aktuálnej pozícii zariadenia a sieť *ZrusenaWifi* sa nachádzala v kalibračnom bode, no v meraní na aktuálnej pozícii chýbala.

Po preiterovaní všetkých kalibračných bodov a vypočítaní priemernej odchýlky v intenzitách signálov nimi vysielaných algoritmus vyberie bod, ktorý má priemernú odchýlku najmenšiu, čo je v tomto prípade považované za bod, ku ktorému by malo byť zariadenie vzdialenostne najbližšie. Následne sa rozdelí okolie tohto bodu na štyri kvadranty (severný, východný, západný a južný), pričom z každého kvadrantu sa vyberie práve jeden bod, ktorý sa nachádza najbližšie ku referenčnému bodu, ako je ilustrované na Diagrame 4.

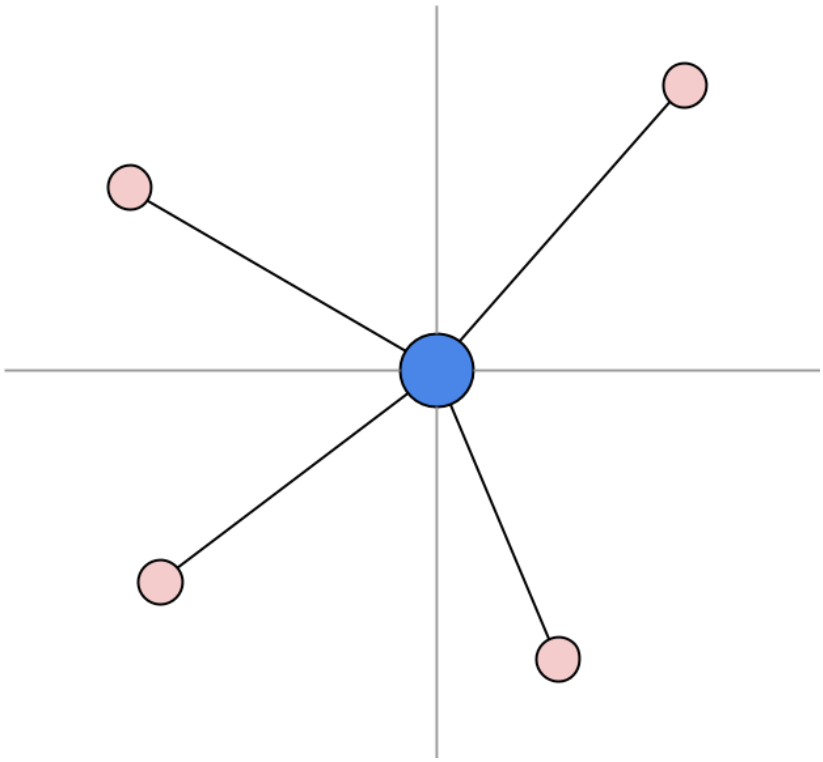

Diagram 4 – Rozdelenie okolia bodu na kvadranty a výber najbližších bodov

Po vybratí týchto štyroch bodov začne algoritmus generovať body na spojniciach (čierna farba) medzi referenčným bodom (modrá farba) a najbližšími bodmi v kvadrantoch (ružová farba). Tieto body, vytvorené interpolovaním, predstavujú odhadovanú intenzitu signálov jednotlivých prístupových bodov na danom mieste. Zámerom rozdelenia priestoru na kvadranty, výberu štyroch bodov a následnej interpolácie je zistiť, či nie je zhoda medzi aktuálnym bodom a interpolovanou hodnotou po spriemerovaní lepšia ako v modrom, referenčnom bode. Algoritmus sa preto vráti do prvej fázy a urobí porovnanie medzi meraním na aktuálnej pozícii zariadenia a interpolovanými bodmi. Z týchto bodov sa následne vyberie bod, ktorý má zo všetkých bodov najmenšiu priemernú odchýlku v intenzitách signálov s porovnaním s bodom, ktorý bol získaný meraním na aktuálnej pozícii zariadenia. Týmto spôsobom sa môže ešte viac zvýšiť presnosť odhadnutia pozície zariadenia.

3.2.2. Výhody a nevýhody algoritmu

Hlavnou výhodou tohto algoritmu je jeho funkčnosť a presnosť aj v uzavretých priestranstvách. Predchádzajúci algoritmus (triangulácia na základe odhadnutia vzdialenosti od vysielača pomocou intenzity signálu siete) síce fungoval vynikajúco

v otvorených priestranstvách, v budovách a zastavaných priestoroch však jeho presnosť rapídne klesla. Pomocou tohto algoritmu a dobrej databázy kalibračných bodov vieme dosiahnuť vysokú presnosť, v dokonalých podmienkach dokonca menej než na meter.

Hlavnou nevýhodou tohto algoritmu je jeho závislosť na databáze kalibračných bodov a jej aktuálnosti. Ak sa náhodou stane, že niektorý z prístupových bodov je vypnutý alebo zrušený, algoritmus to nemá ako vedieť a všetky porovnania bodov budú týmto faktom ovplyvnené, čím sa zníži celková presnosť určovania pozície zariadenia. Ďalším podobným problémom sú mobilné prístupové body, ktoré si ľudia vytvárajú pomocou mobilných telefónov, aby si urobili súkromnú WiFi sieť pre svoj notebook alebo tablet. Tieto prístupové body sú prítomné v meraniach pri vytváraní databázy kalibračných bodov a aj pri samotnom určovaní pozície. Nakoľko sú tieto prístupové body na nerozoznanie od klasických prístupových bodov, nedajú sa automaticky vyfiltrovať z databázy kalibračných bodov a ani pri uskutočňovaní merania na aktuálnej pozícii zariadenia. Dopad na výsledky porovnávania bodov je ilustrovaný v tabuľke nižšie:

Aktuálna pozícia	**Rozdiel v intenzite**	**Kalibračný bod**
Eduroam	10 dBm	Eduroam
FMFI	15 dBm	FMFI
NovaWifi	95 dBm	-----
-----	95 dBm	ZrusenaWifi
HTC MobileHotspot	95 dBm	-----
-----	95 dBm	Samsung MobileHotspot

Ako vidno v tabuľke, celková odchýlka je tento krát 405 dBm, čo zhoršuje priemernú odchýlku intenzít signálov prístupových bodov na 67.5 dBm. V predchádzajúcom prípade s rovnakými prístupovými bodmi bola táto odchýlka iba 53.75 dBm, čo predstavuje zhoršenie o 13.75 dBm len kvôli prístupovým bodom vytvorených študentmi za účelom lepšieho prístupu na internet. Vo všeobecnosti je však tento algoritmus pomerne presný v otvorených aj uzavretých priestranstvách, čo ho činí vhodným pre použitie v mobilnej aplikácii pre operačný systém Android, pomocou ktorej budeme testovať presnosti jednotlivých algoritmov v reálnom živote.

3.3. Lokalizovanie nájdením najzhodnejšieho bodu na vopred interpolovanej mape meraní

Tretím, a zároveň najkomplexnejším algoritmom, ktorý budem v mojej bakalárskej práci popisovať, je lokalizovanie zariadenia nájdením najzhodnejšieho bodu na vopred interpolovanej mape meraní. Zariadenie pri tomto algoritme pracuje s databázou kalibračných bodov, ku ktorým pri každej zmene vygeneruje chýbajúce body v mriežke interpoláciou pomocou barycentrických koordinátov.

3.3.1. Fungovanie algoritmu

Rovnako ako v predchádzajúcom algoritme, zariadenie si pri požiadavke o lokalizáciu zariadenia urobí meranie na aktuálnej pozícii zariadenia, ktoré následne porovnáva s bodmi v kalibračnej databáze. Algoritmus sa snaží vylepšiť výpočet lokácie pomocou automaticky vygenerovaných interpolovaných bodov, podobne ako v druhom algoritme (lokalizácia pomocou nájdenia najpodobnejšieho bodu a následnej interpolácie po spojniciach medzi bodmi v kvadrantoch). Na rozdiel od tohto algoritmu však systém generuje interpolované body už pri pridávaní kalibračných bodov a systematicky sa snaží dosiahnuť mriežku, aby existoval aspoň jeden bod na každých pár metrov plochy, či už kalibračný alebo interpolovaný.

Pri tomto algoritme sa pri každej požiadavke o vygenerovanie mriežky interpolovaných bodov postupuje nasledovne:

- Z databázy kalibračných bodov sa vyberú všetky body, ktoré sa algoritmus snaží pospájať do trojuholníkov čo najvhodnejším spôsobom pre následnú interpoláciu, k čomu využíva Delaunayovu trianguláciu, ktorá je podrobnejšie popísaná neskôr.
- Algoritmus následne spustí nad každým trojuholníkom procedúru, ktorá interpoláciou dopočíta hodnoty bodov (dostupné prístupové body a intenzity ich signálov) vnútri trojuholníka, k čomu využíva barycentrické koordináty.
- Na záver hľadania pozície sa spustí vyhľadávanie nad databázou bodov z kalibračnej databázy a interpolovaných bodov, ktoré vráti práve jeden bod, ktorý má najlepší index zhody s meraním na aktuálnej pozícii zariadenia. Index

zhody je určovaný rovnako ako v predchádzajúcom algoritme, a to porovnávaním vysielaných prístupových bodov a ich intenzít.

3.3.2. Delaunayova triangulácia

Jedným z problémov tohto algoritmu je správne rozdelenie celej databázy kalibračných bodov na trojuholníky tak, aby vzniknuté trojuholníky mali uhly približne rovnakej veľkosti. Na základe môjho výskumu na túto tému som dospel k rozhodnutiu, že najvhodnejším spôsobom, ako množinu bodov z kalibračnej databázy rozdeliť na takéto trojuholníky, je práve Delaunayova triangulácia. Výhodou Delaunayovej triangulácie je fakt, že práve Delaynayova triangulácia maximalizuje minimálny vnútorný uhol vo všetkých trojuholníkoch, tvorených bodmi z kalibračnej databázy (de Berg, et al., 2008).

Delaunayova triangulácia τ je také rozdelenie množiny bodov P v rovine alebo priestore vtedy a práve vtedy, keď opísaná kružnica ľubovoľného trojuholníka z τ neobsahuje bod z množiny P vo svojom vnútre (de Berg, et al., 2008). Takáto triangulácia aj s opísanými kružnicami je ilustrovaná na Diagrame 5.

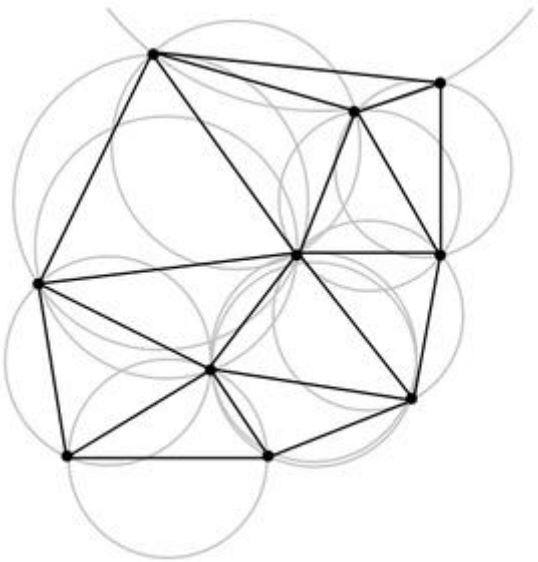

Diagram 5 - Delaunayova triangulácia bodov P v rovnice aj s opísanými kružnicami jednotlivých trojuholníkov[1]

[1] Zdroj obrázku: http://upload.wikimedia.org/wikipedia/commons/thumb/c/c9/Delaunay_circumcircles.png/280px-Delaunay_circumcircles.png

Množina bodov *P* je tvorená bodmi z kalibračnej databázy meraní, ktoré pre účely triangulácie obsahujú svoju pozíciu v rovine. Táto pozícia bude použitá pri samotnej triangulácii ako súradnice *x* a *y*. Informácie o prístupových bodoch a o intenzite signálov nimi vysielaných, dostupných na danej lokácií, sú následne použité v ďalšej časti algoritmu, kde sa umelo dopočítajú body vnútri trojuholníkov pomocou barycentrických koordinátov. Tieto umelo vypočítané body sú následne zrovnoprávnené s bodmi v kalibračnej databáze, aby sa dali neskôr použiť na porovnanie s meraním na aktuálnej pozícii hľadaného zariadenia pre účely nájdenia najzhodnejšieho bodu.

3.3.3. Barycentrické koordináty a interpolácia

Ak sa algoritmu podarí korektne rozdeliť databázu kalibračných bodov na trojuholníky, prichádza na rad samotné počítanie hodnôt bodov obsiahnutých v každom trojuholníku. Účelom tohto dopočítania je zistenie intenzity signálu vysielaného prístupovými bodmi na každom mieste nachádzajúcom sa vnútri plochy, ktorá je tvorená bodmi z kalibračnej databázy a tým zväčšiť množinu bodov, s ktorými sa porovnáva meranie na aktuálnej pozícii zariadenia. Na interpolovanie náhodného bodu v trojuholníku sa použijú barycentrické súradnice, pomocou ktorých sa vypočítajú váhy vrcholov, z ktorých sa následne dopočíta finálna hodnota intenzity jednotlivých sietí bodu.

Najjednoduchší spôsob, ktorým sa dajú vypočítať barycentrické súradnice *(u, v, w)* pre bod *P* v trojuholníku *ABC*, sú nasledujúce tri rovnice:

$$u = \frac{[PBC]}{[ABC]}$$

$$v = \frac{[APC]}{[ABC]}$$

$$w = \frac{[ABP]}{[ABC]}$$

V hore uvedených rovniciach sa pod [*ABC*] myslí plocha trojuholníka *ABC*, rovnako ako pre [*PBC*], [*APC*] a [*ABP*] sa myslí plocha trojuholníkov tvorená danými vrcholmi. Ukážkový trojuholník a jeho rozdelenie na tri podtrojuholníky s centrálnym bodom *P* je ilustrované na Diagrame 6.

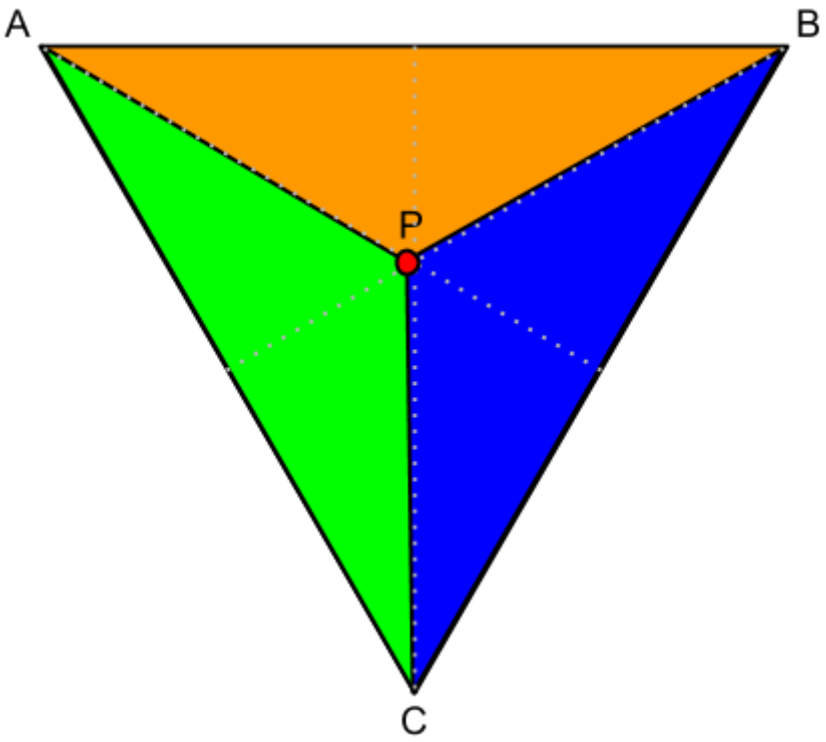

Diagram 6 – Trojuholník ABC s centrálnym bodom P a jeho rozdelenie na tri podtrojuholníky

Nakoľko je bod *P* položený v ťažisku trojuholníka *ABC* (ťažnice sú na diagrame reprezentované šedou prerušovanou čiarou), podtrojuholníky *ABP*, *APC* a *PBC* sú rovnakých veľkostí. Výsledné barycentrické súradnice bodu *P* sú teda $(\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$. Tieto barycentrické koordináty sa následne použijú vo váženom priemere intenzít signálov ako váhy jednotlivých vrcholov, z čoho vzíde finálna interpolovaná hodnota signálu pre prístupový bod v danom bode *P*. Takýmto spôsobom prejde algoritmus všetky trojuholníky, aby vytvoril dostatok bodov pre účely lokalizácie zariadenia.

Po vytvorení dostatočného množstva bodov sa spustí vyhľadávanie nad množinou interpolovaných a kalibračných bodov, v ktorom sa hľadá bod s najmenšou priemernou odchýlkou intenzít prístupových bodov. Tento postup je podrobnejšie opísaný v predchádzajúcom algoritme, no v tomto prípade sa však zároveň vyhľadáva aj vo všetkých interpolovaných bodoch. Algoritmus následne vráti ako odhadovanú lokáciu zariadenia bod, ktorého priemerná odchýlka od merania na aktuálnej pozícii je najmenšia zo všetkých porovnaní.

3.3.4. Výhody a nevýhody algoritmu

Hlavnou výhodou tohto algoritmu je jeho rýchlosť pri samotnom vyhľadávaní lokácie užívateľa. Nakoľko sú všetky body už vopred vygenerované a ich hodnoty interpolované z trojuholníkov, jediné čo musí algoritmus urobiť, je spustiť vyhľadávanie nad databázou všetkých bodov (aj kalibračných aj interpolovaných) a vrátiť bod s najmenšou odchýlkou. Predchádzajúci algoritmus musel pri samotnom hľadaní ešte dodatočne interpolovať spojnice medzi bodmi, čo predstavovalo dodatočnú záťaž na procesor mobilného telefónu a spomaľovalo určenie lokácie.

Interpolovanie bodov už pri pridávaní kalibračných bodov je zároveň aj jednou z nevýhod tohto algoritmu, nakoľko po akejkoľvek zmene v databáze kalibračných bodov je potrebné nanovo vygenerovať všetky interpolované body. Pre koncového užívateľa je však výhodnejšie chvíľu počkať po kalibrácií a následne mať rýchle lokalizovanie zariadenia, než mať rýchlu kalibráciu a pomalé lokalizovanie zariadenia.

Algoritmus je rovnako ako predchádzajúci algoritmus náchylný na ovplyvnenie meraní súkromnými mobilnými prístupovými bodmi, ktoré nie sú stabilné – menia svoju polohu a ich životnosť je zväčša len pár hodín. Tento problém je podrobnejšie popísaný vo výhodách a nevýhodách predchádzajúceho algoritmu.

Ďalšou z nevýhod tohto a zároveň aj predošlého algoritmu je jeho nepresnosť, ak sa aktuálna pozícia zariadenia nachádza mimo plochy tvorenej kalibračnými a interpolovanými bodmi. V tomto prípade totiž neexistujú interpolované body v okolí pozície zariadenia, nakoľko sa algoritmom interpolujú iba hodnoty v trojuholníkoch, ktoré sú tvorené vrcholmi kalibračných bodov. Ukážková situácia je znázornená na Diagrame 7:

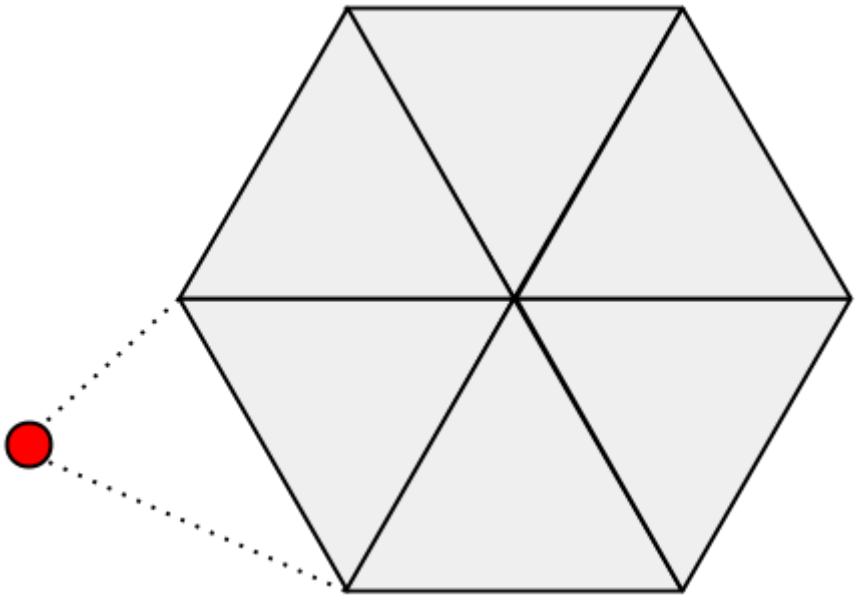

Diagram 7 - Zariadenie mimo interpolovanej mriežky

Ako vidno na diagrame, reálna lokácia smartfónu (na diagrame červený bod) sa nachádza mimo interpolovanej plochy (na diagrame šedou farbou). Nakoľko algoritmus nevie, že sa zariadenie nachádza mimo kalibračných a interpolovaných bodov, samotné hľadanie prebieha bežným spôsobom. Toto však môže spôsobiť, že lokácia užívateľa nebude detekovaná presne, resp. algoritmom odhadnutá poloha bude ležať v interpolovanej ploche, takže presnosť lokalizácie v metroch sa bude v najlepšom prípade rovnať vzdialenosti medzi pozíciou zariadenia a najbližším kalibračným alebo interpolovaným bodom, ak práve tento bod má najmenšiu priemernú odchýlku s meraním na aktuálnej pozícii zariadenia.

Ak sa však zariadenie nachádza vnútri plochy tvorenej kalibračnými a interpolovanými bodmi, presnosť tohto algoritmu je teoreticky vyššia ako u predchádzajúcich dvoch algoritmoch. V predchádzajúcom algoritme sa totiž interpolovali iba hodnoty intenzít signálov prístupových bodov v okolí referenčného bodu, ktorý sa pri prvotnom preiterovaní cez databázu kalibračných bodov najviac podobal na meranie na aktuálnej pozícií. Takto môže vzniknúť situácia, že si systém vyberie kalibračný bod, ktorý sa na meranie na aktuálnej pozícií podobá menej, ako bod získaný interpoláciou na druhom konci budovy, kde sa aj zariadenie nachádza. Táto interpolácia však neprebehne, keďže sa interpolujú iba hodnoty v okolí referenčného bodu. Nakoľko si tretí algoritmus už vopred vygeneruje všetky body z trojuholníkov pomocou barycentrických koordinátov a následnej

interpolácie, algoritmus sa pri porovnávaní vždy pozerá na všetky body, čím sa vyhneme situácií, kedy sa nevyinterpoluje bod z inej časti mapy. Vo všeobecnosti je tento algoritmus presný pri určovaní polohy lokalizovaného zariadenia, preto bude použitý aj v ukážkovej aplikácií na mobilný operačný systém Android, v ktorej budeme tieto algoritmy porovnávať.

4. Implementácia

Poznatky, ku ktorým sme dospeli pri našom výskume o možnostiach lokalizovania zariadenia s pomocou WiFi signálu, sme sa rozhodli využiť v ukážkovej aplikácií na mobilný operačný systém Android. Implementáciu jednotlivých funkcionalít v jazyku Java budem podrobne popisovať v tejto kapitole.

4.1. UML diagram aplikácie

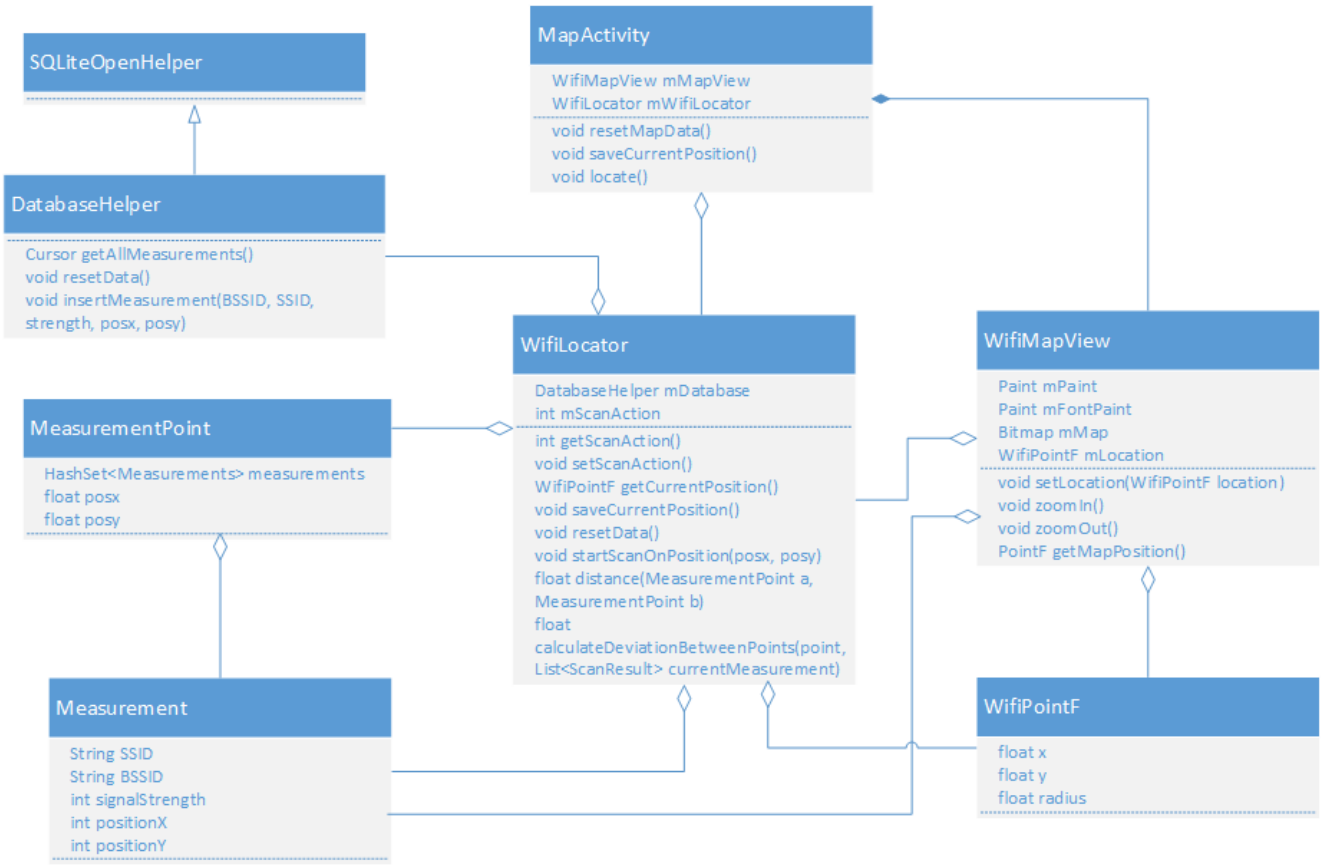

Diagram 8 – UML diagram

Aplikácia je tvorená na základe Model-View Controller prístupu (MVC). V aplikácii je jediný View, ktorý sa volá `WifiMapView` a obsahuje mapu priestorov, v ktorých sa zariadenie nachádza, spolu s ovládacími prvkami (lokalizuj zariadenie, pridaj

bod do kalibračnej databázy, vymaž databázu kalibračných a interpolovaných bodov). Tento View je ovládaný controllerom `MapActivity`, ktorý reaguje na ovládacie prvky vo `WifiMapView` a volá príslušné metódy a procedúry ostatných tried.

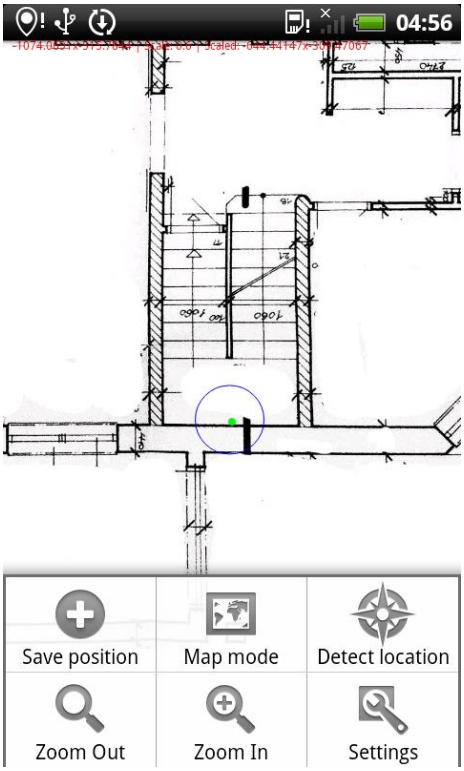

Diagram 9 - Grafické rozhranie aplikácie na operačnom systéme Android

Kalibračná databáza bodov je spravovaná triedou `DatabaseHelper`, ktoré dedí triedu `SQLiteOpenHelper`. Ako sa dá vytušiť už z názvu triedy, dáta sú uložené v SQLite databáze. K databáze ako takej ma priamy prístup iba `DatabaseHelper`, ostatné triedy môžu k databáze pristupovať iba cez funkcie a procedúry tejto triedy.

Na pomoc a reprezentáciu dát sú v aplikácii použité tri dátové štruktúry vytvorené pre tento účel:

- **Measurement**
    - Trieda `Measurement` nemá žiadne procedúry ani metódy, slúži na reprezentáciu stavu jednotlivých prístupových bodov, nakoľko obsahuje polia ako SSID, BSSID, sila signálu a pozícia merania.

- **MeasurementPoint**
    - Trieda `MeasurementPoint` združuje všetky prístupové body (uložené ako inštancia triedy `Measurement`), ktoré boli namerané pri meraní. Tieto body sú uložené v dátovej štruktúre HashSet. Trieda `MeasurementPoint` obsahuje aj pozíciu tohto merania.
- **WifiPointF**
    - Trieda `WifiPointF` sa používa najmä vo View-i `WifiMapView` na určenie lokácie zariadenia, nakoľko táto trieda má okrem pozície aj premennú `radius`, ktorým sa určuje odhadovaná presnosť výpočtu lokácie. Táto informácia je následne prezentovaná užívateľovi ako bod na mape (pozícia) s kružnicou, ktorá reprezentuje odhadovaný rádius lokácie zariadenia.

Trieda `WifiLocator` obsahuje algoritmy na lokalizáciu zariadenia pomocou WiFi signálu, ktoré sú popísané v predchádzajúcej kapitole. Trieda obsahuje referenciu na inštanciu triedy `DatabaseHelper`, pomocou ktorej pristupuje ku databáze kalibračných bodov. Trieda rovnako obsahuje atribút `mScanAction`, v ktorom je uložená aktivita, ktorú chce užívateľ vykonať (uloženie nového kalibračného bodu, lokalizovanie zariadenia), nakoľko po vyžiadaní informácie o dostupných prístupových bodoch systému trvá pár sekúnd, kým v druhom vlákne dokončí scan okolitých WiFi sietí. Trieda má funkciu na vypočítanie vzdialenosti medzi dvoma bodmi – `float distance(MeasurementPoint a, MeasurementPoint b)` a funkciu na vypočítanie priemernej odchýlky intenzít signálov prístupových bodov na aktuálnej lokácií a daného bodu z kalibračnej databázy.

4.2. Meranie intenzity signálu prístupových bodov

Základným kameňom celej aplikácie je meranie dostupných prístupových bodov, intenzita signálov nimi vysielaných sietí a ich pozícia na mape. V operačnom systéme Android sa na tento účel používa trieda `WifiManager`, ktorú som v mojej aplikácií implementoval nasledovným spôsobom:

```
// Instanciovanie triedy, starajúcej sa o lokalizáciu
mWifiLocator =new WifiLocator(this);

IntentFilter i =new IntentFilter(WifiManager.SCAN_RESULTS_AVAILABLE_ACTION);
```

```
// Listener na broadcast spravy, ktore vysiela WifiManager ked ukonci SCAN
registerReceiver(new BroadcastReceiver(){
    public void onReceive(Context c, Intent i){

        // MAP_SCAN_ACTION je rovna ulozeniu, takze ulozime bod
        if(mWifiLocator.getScanAction()== WifiLocator.MAP_SCAN_ACTION_SAVE) {

            // Ulozime bod do kalibracnej databazy
            mWifiLocator.saveCurrentPosition();
            mMapView.setMeasurementPoints(mWifiLocator.getMeasurements());
            // Refreshneme mapu, aby sa na nej zobrazil namerany kalibracny bod
            mMapView.invalidate();
            mWifiLocator.setScanAction(-1);
        }
        elseif(mWifiLocator.getScanAction()==WifiLocator.MAP_SCAN_ACTION_LOCATE){

            Log.i("AppMessages","Scan for locate done...");
            WifiPointF location = mWifiLocator.getCurrentPosition();

            // Ak je lokacia NULL, nepodarilo sa zariadenie lokalizovat
            if(location ==null) {
                return;
            }

            // Nastavime lokaciu na mape na terajsie miesto
            mMapView.setLocation(location);
            mWifiLocator.setScanAction(-1);
        }
    }
}, i);
```

Ak sa niektorá trieda zaregistruje ako `BroadcastReceiver`, je schopná prijímať všetky broadcast správy od triedy `WifiManager`. V okamihu, keď trieda `WifiManager` úspešne ukončí meranie dostupných prístupových bodov a intenzít ich signálov na aktuálnej pozícii zariadenia (zväčša behom pár sekúnd), vyšle broadcast správu všetkým triedam, ktoré sa predtým zaregistrovali ako `BroadcastReceiver`. Táto správa je následne spracovaná v dvoch prípadoch:

1. Chceme pridať nový bod do kalibračnej databázy. Meranie sa teda uloží spolu s pozíciou na mape, ktorú si manuálne určí užívateľ v aplikácií.
2. Chceme lokalizovať zariadenie v priestore. Meranie sa teda použije v príslušnom algoritme, ktorý si užívateľ zvolí (lokalizácia zariadenia pomocou nájdenia najzhodnejšieho bodu z kalibračných bodov a následná interpolácia bodov v kvadrante alebo sa použije algoritmus na nájdenie najzhodnejšieho bodu z množiny kalibračných bodov a interpolovaných bodov získaných pomocou Delaunayovej triangulácie a barycentrických koordinátov.

4.3. Výpočet priemernej zhody medzi kalibračným bodom a aktuálnym meraním

Oba algoritmy na výpočet lokácie užívateľa potrebujú k úspešnému výpočtu počítať priemernú odchýlku v intenzitách signálov prístupových bodov dostupných na aktuálnej pozícii zariadenia, v porovnaní s kalibračnými bodmi z databázy. Tento výpočet je implementovaný v triede `WifiLocator`, ktorá obsahuje oba algoritmy na určovanie lokácie zariadenia. Funkcia má dva argumenty: `MeasurementPoint point`, kalibračný bod, ktorý ideme porovnávať a `List<ScanResult> currentMeasurement`, ktorý je sprostredkovaný priamo od triedy `WifiManager`, ktorá sa v operačnom systéme Android stará o prácu s WiFi prístupovými bodmi a intenzitami ich signálov. Implementácia tejto funkcie na výpočet priemernej odchýlky vyzerá nasledovne:

```
publicfloat calculateDeviationBetweenPoints(MeasurementPoint point,
List<ScanResult> currentMeasurement){

    float signalDifferenceSum =0;
    int signalDifferenceItems =0;

    // iterujeme vsetky wifi, ktore sme na danom mieste nechytali a teraz chytame
    search:
    for(ScanResult s : currentMeasurement){

        for(Measurement m : point.measurements)
            if(s.BSSID.equals(m.BSSID))
                continue search;

        float signalDifference = Math.max(0, s.level +95);
        ++signalDifferenceItems;


        signalDifferenceSum += signalDifference;
    }

    // iterujeme vsetky namerane wifi v danom mieste
    for(Measurement m : point.measurements){

        float signalDifference = Math.max(0, m.signalStrength +95);


        /*
         * Pozrieme ci mame danu wifi v terajsom merani
         * matchujeme BSSID, kedze jedna wifi ma zhodne SSID ale
         * BSSID je unikatne pre kazde AP
         */
        for(ScanResult s : currentMeasurement){
            if(s.BSSID.equals(m.BSSID)){
                signalDifference = Math.abs(s.level - m.signalStrength);
                break;
            }
        }

        signalDifferenceSum += signalDifference;
        ++signalDifferenceItems;
```

```
    }

    float deviation = signalDifferenceSum / signalDifferenceItems;

    return deviation;
}
```

Ako možno vidieť na implementácii tejto funkcie, algoritmus najprv preiteruje cez všetky prístupové body, ktoré sa v kalibračnom bode nenachádzajú, no nachádzajú sa v meraní uskutočnenom na aktuálnej pozícii zariadenia. Všetky tieto prístupové body označí ako nedostupné a počíta s hodnotou signálu -95 dBm, z ktorej následne vypočíta odchýlku.

Algoritmus následne uskutoční rovnaký postup s intenzitami prístupových bodov, ktoré aktuálne nechytáme, ale v kalibračnom bode sa nachádzajú. Týmto bodom rovnako pridelí intenzitu -95 dBm a vypočíta odchýlku, ktorú pridá do priemeru.

Ako posledný krok algoritmus zarátava body, ktoré sa nachádzajú aj v kalibračnom bode aj v aktuálnom meraní. Odchýlka sa vypočíta ako rozdiel v intenzitách signálov. Následne sa vypočíta nevážený priemer zo všetkých vypočítaných deviácií, ktorý je vrátený ako výsledná hodnota funkcie.

4.4. Algoritmus lokalizácie zariadenia pomocou nájdenia najpodobnejšieho kalibračného bodu

Nakoľko je algoritmus lokalizácie zariadenia pomocou nájdenia najpodobnejšieho kalibračného bodu pomerne rozsiahly, v tejto časti budem analyzovať a ukazovať úryvky iba dôležitých častí kódu.

Algoritmus v prvom kroku získa všetky doteraz vytvorené kalibračné body od triedy `DatabaseHandler`, ktoré vloží do interného poľa, použitého na dočasné výpočty. Následne vyberie bod zo všetkých kalibračných bodov, ktorý má najmenšiu odchýlku v porovnaní s meraním na aktuálnej pozícii.

```
MeasurementPoint bestPoint =null;
float bestDeviation =999999.0f;

/*
 * Preiterujeme vsetky body a pozrieme si zhodu, vyberieme si ten
najlepsi,
 * od ktoreho potom budeme robit interpolovane ciarocky
 *
```

```
 * jednotlive wifi priemerujeme vazenym priemerom
 */
for(MeasurementPoint point : measurementPoints.values()){
    Log.i("wifilocator", point.posx+"x"+point.posy);

    float deviation = calculateDeviationBetweenPoints(point,
currentMeasurement);
    if(deviation < bestDeviation){
        bestPoint = point;
        bestDeviation = deviation;
    }

    Log.i("wifilocator","=== AVERAGE DEVIATION: "+deviation);
}

Log.i("Locator","Best point position:
"+bestPoint.posx+"x"+bestPoint.posy);
```

Algoritmus v tomto kroku vypočíta priemernú odchýlku medzi jednotlivými bodmi, pomocou funkcie `calculateDeviationBetweenPoints`, ktorá je popísaná vyššie. Následne najlepší bod vyberie a použije ho v nasledujúcej časti, v ktorej algoritmus vyberá štyri najbližšie body z jednotlivých kvadrantov v okolí bodu s najmenšou priemernou odchýlkou:

```
/*
 * selectneme 4 body
 *
 *    A          B
 *
 *       tu som
 *
 *    C          D
 *
 * vyinterpolujeme ciary medzi mojou pozicou a kazdym bodom.
 * nasledne sa pozrieme ci sa nemame po nejakej priamke posunut
 */
MeasurementPoint[] interpolArray =new MeasurementPoint[4];

for(MeasurementPoint point : measurementPoints.values()){
    // vlavo hore
    if(point.posx < bestPoint.posx && point.posy > bestPoint.posy){
        if(interpolArray[0]==null)
            interpolArray[0]= point;
        elseif(distance(bestPoint, point)< distance(bestPoint, interpolArray[0]))
            interpolArray[0]= point;
    }
    // vpravo hore
    elseif(point.posx > bestPoint.posx && point.posy > bestPoint.posy){
        if(interpolArray[1]==null)
            interpolArray[1]= point;
        elseif(distance(bestPoint, point)< distance(bestPoint, interpolArray[1]))
            interpolArray[1]= point;
    }
    // vlavo dole
    elseif(point.posx < bestPoint.posx && point.posy < bestPoint.posy){
        if(interpolArray[2]==null)
            interpolArray[2]= point;
        elseif(distance(bestPoint, point)< distance(bestPoint, interpolArray[2]))
            interpolArray[2]= point;
```

```
    }
    // vpravo dole
    elseif(point.posx > bestPoint.posx && point.posy < bestPoint.posy){
        if(interpolArray[3]==null)
            interpolArray[3]= point;
        elseif(distance(bestPoint, point)< distance(bestPoint, interpolArray[3]))
            interpolArray[3]= point;
    }
}
```

Algoritmus následne pristúpi ku samotnej interpolácií bodov na spojniciach medzi bodmi z kvadrantov a referenčným bodom. Ak sa náhodou interpolovaný bod zhoduje s meraním na aktuálnej pozícii viac ako referenčný bod, ako pozícia zariadenia sa označí práve tento interpolovaný bod:

```
for(int i =0; i < interpolArray.length; i++){
        MeasurementPoint point = interpolArray[i];

        if(point ==null)
                continue;

        if(point.posx == bestPoint.posx && point.posy == bestPoint.posy)
                continue;

        float distance = distance(point, bestPoint);

        int steps = Math.max((int)distance /5,20);

        /*
         * Preinterpolujeme priamku medzi aktualnou poziciou a bestPointom
         * a porovname zhodu bodov
         */
        for(int j =0; j < steps; j++){
                float completion =(float)j / steps;

                MeasurementPoint interpolatedPoint =new MeasurementPoint();
                interpolatedPoint.posx = bestPoint.posx +(point.posx -
bestPoint.posx)*completion;
                interpolatedPoint.posy = bestPoint.posy +(point.posy -
bestPoint.posy)*completion;

                // iterujeme vsetky wifi, ktore sme na koncovom bode nechytali a v
bestPointe hej
                search:
                for(Measurement n : bestPoint.measurements){
                        for(Measurement m : point.measurements)
                                if(n.BSSID.equals(m.BSSID))
                                        continue search;

                        float signalStrength =((n.signalStrength +95)*completion)-95;

                        // pridame
                        if(signalStrength >-93){
                                Measurement bodicek =new Measurement(n.BSSID,
n.SSID,(int)signalStrength,(int)interpolatedPoint.posx,(int)interpolatedPoint.posy);
                                interpolatedPoint.measurements.add(bodicek);
                        }
                }

                // iterujeme vsetky namerane wifi v koncovom bode
                for(Measurement m : point.measurements){
                        float signalStrength =((m.signalStrength +95)* completion)-95;
                        /*
                         * Pozrieme ci mame danu wifi v terajsom merani
                         * matchujeme opat BSSID
                         */
                        for(Measurement n : bestPoint.measurements){
                                if(n.BSSID.equals(m.BSSID)){
                                        signalStrength = n.signalStrength +(m.signalStrength -
n.signalStrength)* completion;
```

```
                                        break;
                                }
                        }

                        if(signalStrength >-93){
                                Measurement bodicek =new Measurement(m.BSSID,
m.SSID,(int)signalStrength,(int)interpolatedPoint.posx,(int)interpolatedPoint.posy);
                                interpolatedPoint.measurements.add(bodicek);
                        }
                }

                float deviation = calculateDeviationBetweenPoints(interpolatedPoint,
currentMeasurement);

                if(deviation < bestInterpolatedDeviation){
                        bestInterpolatedPoint = interpolatedPoint;
                        bestInterpolatedDeviation = deviation;
                }
        }
}
```

Ako posledný krok algoritmu je samotné vyhodnotenie a vrátenie odhadovanej lokácie používateľa:

```
if(bestInterpolatedPoint ==null)
        return new WifiPointF(bestPoint.posx, bestPoint.posy);

return new WifiPointF(bestInterpolatedPoint.posx, bestInterpolatedPoint.posy);
```

4.5. Algoritmus lokalizácie nájdením najzhodnejšieho bodu na vopred interpolovanej mape meraní

Druhým algoritmom na vypočítanie pozície zariadenia je algoritmus lokalizácie nájdením bodu na vopred interpolovanej mape meraní, ktorý sme taktiež opisovali v predchádzajúcej kapitole. Nakoľko aplikácia slúži ako dôkaz o funkčnosti týchto dvoch algoritmov, do algoritmu sme neimplementovali priebežné ukladanie vyinterpolovaných bodov do tabuľky, ale body sú generované interpoláciou pri každom vyhľadávaní. Takýmto spôsobom sa dajú ľahko porovnať rýchlosti nájdenia lokácie oboch algoritmov, nakoľko oba začínajú z rovnakého stavu – majú dostupnú iba databázu s kalibračnými bodmi. Nakoľko je algoritmus komplexnejší a obsahuje viacero procedúr a funkcií, opisovať budem iba významné časti algoritmu.

Prvý krok algoritmu je zhodný s predchádzajúcim algoritmom, keď si systém najprv rozdelí všetky merania do bodov podľa ich pozície. Tieto body následne vloží do algoritmu, ktorý ich usporiada do triangulácie podľa Delaunayovho algoritmu. Nakoľko je implementácia Delaunayovej triangulácie mimo rozsahu témy tejto bakalárskej práce,

použili som už existujúcu implementáciu v jazyku Java (Google Project, 2010). Rozdelenie množiny bodov na trojuholníky som vizualizoval aj v samotnej aplikácii, čo je ilustrované na Diagrame 10:

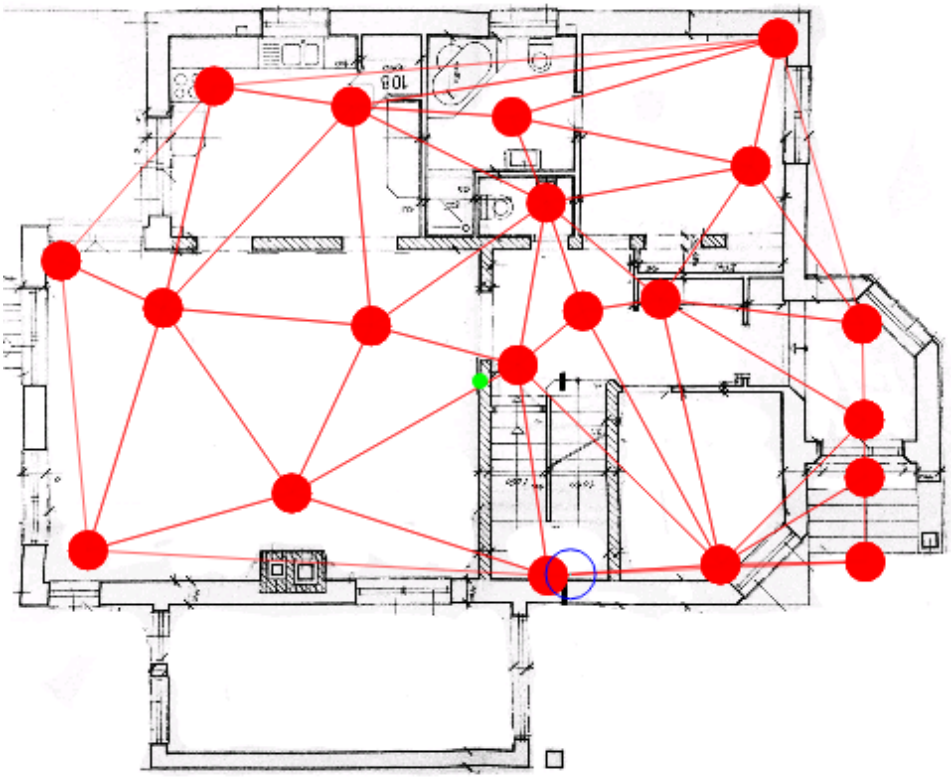

Diagram 10 - Delaunayova triangulácia z množiny kalibračných bodov

Algoritmus následne preiteruje cez všetky trojuholníky a spustí procedúru `interpolateCenter` na interpoláciu ťažiska trojuholníka pomocou barycentrických koordinátov:

```
Triangle_dt curr =null;
Iterator<Triangle_dt> it = mDelaunayTriangulation.trianglesIterator();
while(it.hasNext()){
	curr = it.next();

	if(curr.isHalfplane())
		continue;

	interpolateCenter(curr.p1(), curr.p2(), curr.p3());
}
```

Procedúra `interpolateCenter` vypočíta pozíciu ťažiska v trojuholníku, vypočíta obsah trojuholníka a jeho podtrojuholníkov, tvorených pôvodnými bodmi trojuholníka

a jeho ťažiskom. Na základe týchto údajov sa použitím rovníc z predchádzajúcej kapitoly vypočítajú barycentrické súradnice ťažiska, čím môžeme pristúpiť ku samotnej interpolácii dostupných prístupových bodov a intenzít ich signálov v ťažisku trojuholníka. Implementácia procedúry `interpolateCenter(Point_dt a, Point_dt b, Point_dt c)` je nasledovná:

```
double px =(a.x()+b.x()+c.x())/3;
double py =(a.y()+b.y()+c.y())/3;

Point_dt p =new Point_dt(px, py);

// Vypocitame plochy
double abc = area(a, b, c);
double abp = area(a, b, p);
double apc = area(a, p, c);
double pbc = area(p, b, c);

// Vypocitame vahy
double u = pbc / abc;
double v = apc / abc;
double w = abp / abc;

MeasurementPoint point =new MeasurementPoint();

MeasurementPoint atmp =
mInterpolatedMeasurementPoints.get((int)a.x()+"x"+(int)a.y());
MeasurementPoint btmp =
mInterpolatedMeasurementPoints.get((int)b.x()+"x"+(int)b.y());
MeasurementPoint ctmp =
mInterpolatedMeasurementPoints.get((int)c.x()+"x"+(int)c.y());

MeasurementPoint amp = preparePointForInterpolation(atmp, btmp, ctmp);
MeasurementPoint bmp = preparePointForInterpolation(btmp, atmp, ctmp);
MeasurementPoint cmp = preparePointForInterpolation(ctmp, atmp, btmp);

// Kontrola spravnosti: vsetky body by mali mat rovnaky pocet APciek
if(!(amp.measurements.size()==bmp.measurements.size()&&
amp.measurements.size()==cmp.measurements.size())){
       System.out.println("Oops");
       return;
}
```

Funkcia `preparePointForInterpolation` sa postará o to, aby všetky body mali v meraniach rovnaké prístupové body. Ak náhodou jeden z vrcholov niektorý prístupový bod nenameral pri kalibrácií, tento prístupový bod je doňho umelo pridaný s hraničnou hodnotou nameranteľnosti -95 dBm. Samotná interpolácia je implementovaná nasledovne:

```
for(Measurement m : amp.measurements){

       int aStrength = m.signalStrength;
       int bStrength =999;
       int cStrength =999;

       for(Measurement tmp : bmp.measurements){
              if(tmp.BSSID.equals(m.BSSID)){
```

```
                        bStrength = tmp.signalStrength;
                        break;
                }
        }

        for(Measurement tmp : cmp.measurements){
                if(tmp.BSSID.equals(m.BSSID)){
                        cStrength = tmp.signalStrength;
                        break;
                }
        }

        int signalStrength =(int)((u*aStrength+v*bStrength+w*cStrength)/(u+v+w));
        Measurement interpolatedMeasurement =new Measurement(m.BSSID, m.SSID,
signalStrength,(int)px,(int)py,true);

        point.measurements.add(interpolatedMeasurement);
        mInterpolatedMeasurements.add(interpolatedMeasurement);
}
```

Ak je náhodou niektorá plocha z podtrojuholníkov *ABP*, *APC* alebo *PBC* priveľká, procedúra `interpolateCenter` sa spustí ešte raz pre daný podtrojuholník, čím sa zaručí dostatočne hustá mriežka interpolovaných bodov:

```
if(abp >500)
        interpolateCenter(a, b, p);
if(apc >500)
        interpolateCenter(a, p, c);
if(pbc >500)
        interpolateCenter(p, b, c);
```

Samotný algoritmus na lokalizáciu po vyinterpolovaní všetkých potrebných bodov pokračuje obdobne ako predchádzajúci algoritmus – hľadá najzhodnejší bod z databázy v porovnaní s meraním na aktuálnej pozícií zariadenia. Tentokrát však do množiny bodov, s ktorými meranie porovnáva, zahrnie aj všetky vyinterpolované body:

```
MeasurementPoint bestPoint =null;
float bestDeviation =999999.0f;

/*
 * Preiterujeme vsetky body a pozrieme si zhodu, vyberieme si ten
najlepsi.
 */
for(MeasurementPoint point : mInterpolatedMeasurementPoints.values()){
        float deviation = calculateDeviationBetweenPoints(point,
currentMeasurement);
        if(deviation < bestDeviation){
                bestPoint = point;
                bestDeviation = deviation;
        }

        Log.i("wifilocator","=== AVERAGE DEVIATION: "+deviation);
}
```

```
Log.i("Locator","Best point position:
"+bestPoint.posx+"x"+bestPoint.posy+", deviation: "+bestDeviation);

return new WifiPointF(bestPoint.posx, bestPoint.posy, bestDeviation*10);
```

4.6. Generovanie tepelnej mapy intenzity WiFi signálu

Pre lepšiu vizualizáciu šírenia sa WiFi signálu v priestore sme sa rozhodli interpolované body, vytvorené druhým algoritmom, zobrazovať nad mapou objektu v ktorom sa zariadenia nachádza, ako tepelnú mapu. Pomocou miernej úpravy vykresľovania kalibračných bodov na mapu sme vytvorili nasledovnú procedúru, ktorá vykreslí interpolované body na mapu a pridelí im farbu podľa intenzity ich signálu. Implementácia tejto procedúry je nasledovná:

```
for(Measurement m : mMeasurementPoints){
    if(m.BSSID.equals(mostCommonWifi)){
        float range =(float)highestValueWifi - lowestValueWifi;
        float perc  =(highestValueWifi - m.signalStrength)/range;

        float[] hsv ={250*perc,1,1};

        Paint p =new Paint();
        p.setColor(Color.HSVToColor(250, hsv));
        canvas.drawCircle((mX+m.positionX)*mScale,(mY+m.positionY)*mScale,7, p);
    }
}
```

Farba je v tomto prípade reprezentovaná vo formáte HSV – farebný tón, sýtosť farby a svetelnosť, nakoľko je veľmi ľahko realizovateľný prechod medzi jednotlivými teplotami. Pri zmene triednej premennej `mMeasurementPoints` sa zavolá procedúra, ktorá zo všetkých meraní vyberie najviac objavujúci sa prístupový bod, ktorý je na reprezentáciu šírenia signálu najvhodnejší, nakoľko je na najväčšej ploche. Výsledkom takejto vizualizácie je nasledovná tepelná mapa:

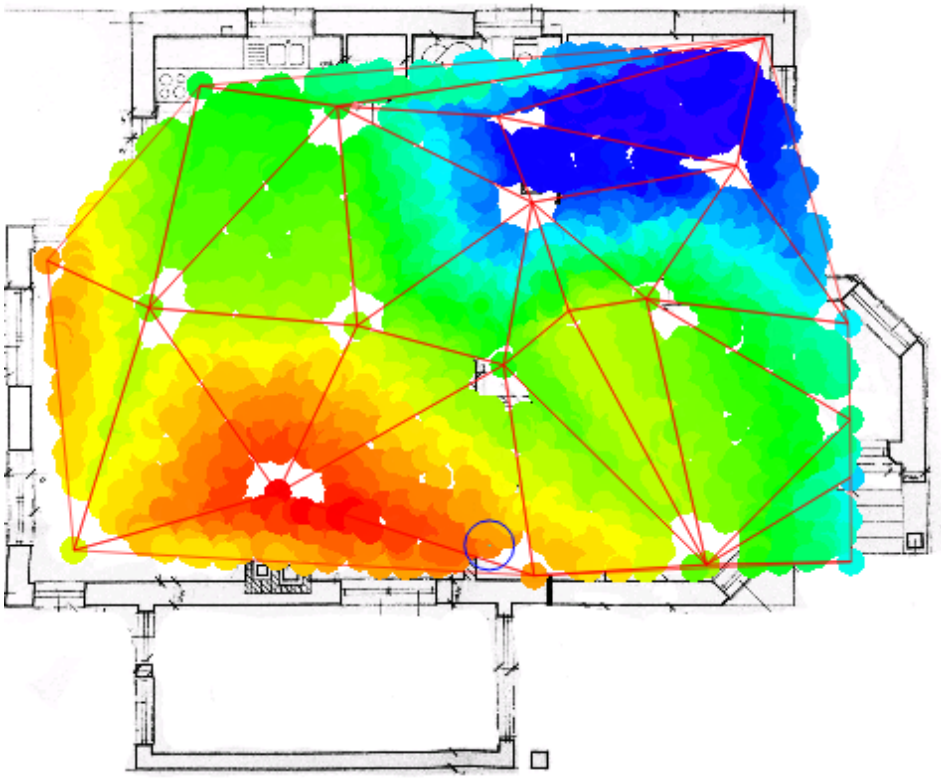

Diagram 11 - Interpolovaná tepelná mapa aj s Delaunayovými trojuholníkmi

5. Presnosť algoritmov

Nakoľko bolo cieľom tejto bakalárskej práce preskúmať možnosti a presnosť lokalizácie zariadenia pomocou WiFi signálu, výslednú aplikáciu na mobilný operačný systém Android sme otestovali v reálnom prostredí. Aplikácia bola testovaná v dvoch budovách, ktoré mali rozdielny tvar, zastavanie aj ľudnatosť. Podmienky počas testovania boli nasledovné:

1) Prvá budova
    - štvorcový tvar budovy
    - testovaná plocha približne 180 m^2
    - 19 kalibračných bodov (9.5 m^2 na kalibračný bod)
    - nízka ľudnatosť
2) Druhá budova
    - podlhovastý tvar budovy
    - testovaná plocha približne 1700 m^2
    - 20 kalibračných bodov (85 m^2 na kalibračný bod)
    - Stredná ľudnatosť

Pri testovaní aplikácie v nasledovných dvoch budovách sme dospeli k viacerým prekvapivým záverom. Jedným z nich bolo, že aplikácia dosahovala v druhej budove prekvapivo vysokú presnosť, nakoľko miestami detekovala lokáciu zariadenia až na meter presne. Druhá budova bola totiž FMFI Univerzity Komenského, prvé poschodie matematického pavilónu, čo je jedna dlhá chodba. Kalibračné body boli v jednej rovine, nakoľko kalibrácia aj detekovanie lokácie boli vykonávané na chodbe. Aplikácia dosahovala následne vysokú presnosť, keď aj interpolované body súhlasili so stavom a intenzitou signálu na danom mieste. Priemerná presnosť bola 2.6 metra, najhoršie určenie lokácie bolo 8 metrov, najlepšie 0.5 metera. Nakoľko boli kalibračné body v rovine a signál sa dobre šíril, presnosť oboch algoritmov bola zhodná.

V druhej budove boli výsledky horšie, no stále pomerne presné. Kalibračné body boli usporiadané do mriežky, v každej miestnosti sa nachádzal aspoň jeden kalibračný bod, okolo budovy boli rozmiestnené štyri kalibračné body, aby interpolovaná plocha pokrývala celú budovu. Priemerná presnosť pri určovaní pozície bola 3.3 metra, najhoršie určenie

pozície bolo 7 metrov použitím algoritmu hľadajúceho najzhodnejší bod a interpoláciou spojníc, najlepšie určenie lokácie bolo 0.5 metra pri algoritme hľadania najzhodnejšieho bodu interpolovaním. Algoritmus určenia lokácie zariadenia nájdením najzhodnejšieho bodu na vopred interpolovanej mape meraní mal v tomto prípade väčšiu presnosť ako algoritmus nájdením najzhodnejšieho kalibračného bodu a následnej interpolácie spojníc s okolitými bodmi. Príčinou lepšej presnosti bola zrejme dostupnosť interpolovaných hodnôt na inak neznámych miestach. Hodnoty jednotlivých meraní v obidvoch budovách boli nasledovné:

Meranie / Presnosť	**FMFI**		**Druhá budova**	
	Best Point	Interpolate	Best Point	Interpolate
1.	0,5 m	1 m	6 m	4 m
2:	2 m	3 m	4 m	3 m
3.	3 m	2 m	3 m	2 m
4.	0,5 m	0,5 m	3 m	4 m
5.	8 m	2 m	7 m	2 m
6.	3 m	5 m	1 m	0,5 m

Každé meranie bolo vykonané na inej lokácii v každej z budov, na každej lokácii sa vykonala lokalizácia zariadenia pomocou obidvoch algoritmov. Výsledné hodnoty ukazujú, že algoritmus interpolate, ktorý využíva interpolované body je síce pomalší, ale jeho presnosť je vyššia, nakoľko v obidvoch budovách priemerná hodnota presnosti určenia lokácie dosiahla hodnotu 2,41 metra. Rýchlejší algoritmus bestpoint, ktorý po nájdení najzhodnejšieho kalibračného bodu interpoluje spojnice je stále dostatočne presný, nakoľko jeho priemerná presnosť v obidvoch budovách dosiahla hodnotu 3,41 metra.

Oba algoritmy dokázali presne určiť pozíciu zariadenia v uzavretých a zaľudnených priestoroch, preto môžeme prehlásiť, že sa dá použiť signál WiFi na lokalizáciu zariadenia aj v uzavretých priestoroch s lepšou presnosťou ako pri GPS / GLONASS.

6. Možné vylepšenia aplikácie

Presnosť algoritmov na lokalizovanie pomocou WiFi signálu, použitých v ukážkovej Android aplikácii, síce dosahujú uspokojivé hodnoty, pri svojom výskume som však dospel ku viacerým vylepšeniam, ktoré by sa dali v budúcnosti implementovať za účelom vylepšenia aplikácie. Jednotlivé riešenia budem popisovať v tejto kapitole.

6.1. Neustále sledovanie pozície zariadenia

Presnosť sledovania aplikácie je jednou z najkrehkejších častí aplikácie. Stačí, aby sa signál odrážal mierne odlišne ako pri kalibrácií a algoritmus už nebude môcť spoľahlivo a presne lokalizovať zariadenie v budove. Tomuto by sa dalo predísť tak, že by aplikácia bežala v pozadí a pasívne by snímala pohyby užívateľa. Nakoľko by pohyb musel byť plynulý, algoritmus by vedel zobrať do úvahy pri výpočtoch možné pohyby užívateľa a veľmi vzdialené časti objektu, v ktorom sa zariadenie s užívateľom nachádza, by automaticky vyhodnocoval ako chybu výpočtu.

Nevýhodou takéhoto prístupu by bola zvýšená spotreba zariadenia, čo by mohlo mať negatívny dopad na používanie aplikácie bežnými užívateľmi.

6.2. Vylepšenie algoritmu počítajúceho zhodu medzi bodmi

Algoritmus na počítanie zhody medzi kalibračnými, interpolovanými a meranými bodmi, ktorý sme popisovali v predchádzajúcich kapitolách síce funguje, jeho presnosť a korektnosť je však diskutabilná. Algoritmus vie síce pomerne presne určiť zhodu medzi bodmi vypočítaním priemernej odchýlky v intenzitách signálov reprezentovaných v dBm, tento prístup však nie je úplne presný, nakoľko táto jednotka je logaritmická. Tento algoritmus taktiež neberie do úvahy rozdiel v citlivosti prijímačov, čím robí databázu kalibračných bodov ťažko použiteľnou na viacerých zariadeniach bez zmien v hodnotách.

Jedným z možných riešení by bolo napríklad použitie upraveného koeficientu korelácie. Tento koeficient by sa vypočítal ako kosínus uhla medzi priamkami tvorenými hodnotami jednotlivých prístupových bodov. Problém v tomto prístupe je však ten, že v prípade znížených intenzít všetkých prístupových bodov v jednom meraní zostáva kosínus uhla rovnaký ako v prípade intenzít oboch meraní na rovnakej úrovni, nakoľko sa priamka iba posunie nižšie po osi *y*. Zariadenie sa však môže v prípade znížených intenzít signálov všetkých prístupových bodov nachádzať od bodu ďalej, čo by mal algoritmus vyhodnotiť ako menšiu zhoda bodu s meraním. Kosínus uhla medzi týmito dvoma porovnaniami intenzít by však zostal rovnaký, čo by negatívne ovplyvňovalo presnosť určenia zhody merania s kalibračnými a interpolovanými bodmi, čo by následne negatívne ovplyvnilo presnosť výpočtu pozície zariadenia. Spomínané problémy by sa však dali odstrániť vhodnou úpravou vzorca a algoritmu, ktorý zhodu meraní počíta.

6.3. Vektorové mapy a trojrozmerné priestory

Oba algoritmy aktuálne pracujú iba s kalibračnými bodmi v rovine a ostatné ovplyvňujúce faktory, ako napríklad steny alebo poschodia, neberú pri výpočtoch vôbec do úvahy. Aplikácia by sa však dala preprogramovať tak, aby ako mapu objektu brala mapu vo vektorovom formáte a nie ako bitmapu. Takýmto prístupom by algoritmus vedel vyhodnotiť, ktoré časti objektu sú steny alebo iné rušivé prekážky a následne by podľa toho upravoval proces interpolácie hodnôt v priestore, ktoré neboli namerané pri kalibrácii.

Algoritmus by mohol rovnako zohľadňovať aj tretí rozmer, nie iba súradnicu dĺžky a šírky. Takýmto spôsobom by vedel pri interpolácii využívať dáta aj z iných poschodí, čím by sa mohla presniť interpolácia a tým pádom aj celková presnosť lokalizácie zariadenia.

6.4. Navigácia

Jedným z možných ďalších smerovaní vývoju tejto aplikácie je práve navigácia. Ako ukážkovú situáciu použijem prvákov, nových študentov, ktorých práve prijali na Fakultu matematiky, fyziky a informatiky Univerzity Komenského. Nakoľko sú títo študenti zväčša noví a ešte nezorientovaní v priestoroch školy, často nevedia nájsť miestnosť, v ktorej im bude pokračovať výuka. Nakoľko už existujúca Android aplikácia

dokáže úspešne lokalizovať zariadenie v uzavretom priestranstve, nič nebráni tomu, aby sa dorobila databáza miestností, ktorá bude obsahovať názov miestnosti a jej lokáciu na mape.

Ak by sa implementovala funkcionalita vektorových máp, ktorú som popisoval ako jedno z možných vylepšení aplikácie, relatívne jednoducho by sa dala urobiť navigácia, ktorá by dokázala užívateľa Android aplikácie navigovať priamo do ním zvolenej miestnosti, čím by sa možno znížili neskoré príchody najmä na začiatku semestra a mnoho stratených študentov na chodbách FMFI UK. Nad vektorovými mapami by sa dal totiž jednoducho spustiť algoritmus na nájdenie najkratšej cesty, nakoľko vieme jasne definovať steny a iné prekážky, čo by mohlo byť následne vizualizované v samotnej aplikácii.

Takto upravená aplikácia by mohla byť podľa môjho názoru často využívaná najmä prvákmi bakalárskeho štúdia a hosťami univerzity, nakoľko by priamo vo vestibule školy mohol byť QR kód s odkazom na stiahnutie tejto aplikácie, takže samotná distribúcia aplikácie takéhoto typu by nebola problém.

Jediným problematickým faktorom tohto riešenia je totiž už viackrát diskutovaná presnosť kalibračných bodov, keď sa prenesú na zariadenie s inou konštrukciou a citlivosťou prijímaču. Aplikácia má totiž relatívne vysokú presnosť určovania polohy s databázou kalibračných bodov vytvorenou priamo zariadením, prenesené body z centrálnej databázy by však mohli spôsobiť problémy a urobiť lokalizáciu zariadenia nepresným.

Záver

V dnešnej dobe si už málokto vie predstaviť život bez určovania lokácie pomocou GPS / GLONASS. Sú však prípady, keď je signál od satelitov nedostatočný, ako napríklad v uzavretých priestoroch. V takýchto prípadoch sa treba poobzerať po alternatívnych spôsoboch lokalizácie, ktoré sa dajú využiť aj v podmienkach, v ktorých sa signál nešíri priamočiaro a rovnomerne.

Jedným z takýchto prístupoch k lokalizácii zariadenia je využitie údajov z WiFi prístupových bodov. Tieto údaje sú voľne dostupné všetkým zariadeniam s WiFi prijímacím modulom. V práci sme sa venovali využitiu údajov ako BSSID siete a intenzita signálu v rôznych vzdialenostiach od vysielačov a rozoberali sme rôzne algoritmy, pomocou ktorých sa dá robiť výpočet lokácie zariadenia. Vybrané dva algoritmy boli následne implementované v ukážkovej aplikácii na mobilný operačný systém Android. Aplikácia vedela po predošlej kalibrácii pomerne presne určiť pozíciu zariadenia v zastavaných priestoroch, čím sa potvrdila možnosť využitia signálu z WiFi prístupových bodov ako jedna z možností lokalizácie zariadenia v sťažených podmienkach. Kalibrácia aplikácie spočívala v nameraní hodnôt intenzít signálov dostupných prístupových bodov a následnom uložení nameraných hodnôt spolu s geografickou lokáciou merania do internej databázy aplikácie.

Výhodou lokalizovania pomocou WiFi signálu v porovnaní s inými prístupmi je možnosť využiť už existujúcu sieť prístupových bodov, nakoľko všetky zariadenia majú prístup ku nami potrebným údajom aj bez znalosti prístupových údajov ku sieťam. Vďaka tomuto faktu je cenová investícia na lokalizáciu zariadení vnútri budovy fakticky nulová, čo je jedným v hlavných benefitov tohto prístupu.

Ďalšou výhodou WiFi lokalizácie je jej presnosť v uzavretých a zastavaných priestoroch. Pri testovaní algoritmov vo vytvorenej Android aplikácii v dvoch odlišných budovách pre dosiahnutie objektívnejších výsledkov sme dokázali, že presnosť lokalizácie na báze intenzít WiFi signálov má dostatočnú presnosť na použitie pri lokalizovaní v budovách, v niektorých prípadoch bolo dokonca určenie pozície zariadenie presné na meter.

Výsledkom práce bola aplikácia s dvoma funkčnými algoritmami, ktoré sa dajú použiť v reálnom živote na lokalizáciu zariadení v zastavaných priestoroch, kde lokalizácia pomocou GPS / GLONASS neprichádza pre zlý signál do úvahy. Lokalizácia zariadení pomocou WiFi signálu má svoju budúcnosť, čo dokazuje aj záujem zo strany veľkých spoločností ako Apple či Google, ktoré už dnes začínajú používať informácie z WiFi prístupových bodov na poskytnutie rýchlejšej a hlavne presnejšej lokalizácie zariadenia.

Zoznam použitej literatúry

Apple. 2012. iOS 5: Understanding Location Services. [Online] Apple, Október 22, 2012. [Cited: Máj 5, 2013.] http://support.apple.com/kb/HT4995.

—. 2012. Staying on Track with Location Services. [Online] 2012. [Cited: Máj 5, 2013.] https://developer.apple.com/videos/wwdc/2012/.

de Berg, Mark, et al. 2008. *Computational Geometry: Algorithms and Applications.* s.l. : Springer-Verlag, 2008. 3540779736.

DSL.sk. 2008. Gears rozšírené o detekciu polohy pomocou WiFi. *DSL.sk*. [Online] 22. 10 2008. [Dátum: 15. 3 2013.] http://www.dsl.sk/article.php?id=6504&hot=2.

Google. 2010. Google submission WiFi collection. [Online] Apríl 27, 2010. [Cited: Máj 5, 2013.] http://static.googleusercontent.com/external_content/untrusted_dlcp/www.google.com/sk//googleblogs/pdfs/google_submission_dpas_wifi_collection.pdf.

Google Project. 2010. Java Delaunay Triangulation. *JDT*. [Online] Január 1, 2010. [Cited: Máj 8, 2013.] https://code.google.com/p/jdt/downloads/list.

Jaborník, Ondrej. 2006. *Triangulácia implicitne definovanej plochy*. [Diplomová práca] Bratislava : s.n., 2006.

ROGER-GPS Oy. GPS Repeater FAQ. [Online] [Dátum: 1. Január 2013.] http://www.gps-repeating.com/index.php?option=com_easyfaq&Itemid=6.

Tanenbaum, Andrew S. 2003. *Computer Networks (4th Edition)*. s.l. : Prentice Hall, 2003. 0-13-066102-3.