Univerzita Hradec Králové

Fakulta informatiky a managementu

KIKM – Katedra informatiky a kvantitativních metod

WiFi lokalizace pomocí 802.11mc Fine Time Measurement / RTT

(WiFi – RTT)

Bakalářská práce

Autor: Jiří Jirásek
Studijní obor: Aplikovaná informatika

Vedoucí práce: Ing. Pavel Kříž, Ph.D.

Hradec Králové Duben 2020

Prohlášení:

Prohlašuji, že jsem bakalářskou/diplomovou práci zpracoval/zpracovala samostatně a s použitím uvedené literatury.

V Hradci Králové dne 29.4.2020 Jiří Jirásek

Poděkování:

Děkuji vedoucímu bakalářské práce Ing. Pavlu Křížovi, Ph.D. za metodické vedení práce a za pomoc při zpracování aplikace.

Anotace

Tato práce seznamuje čtenáře s tématy ohledně radio lokalizace. Dále pak dělení radio lokalizace jako například received signal strength indication, jeho použití a optimalizace, metoda time of flight a nejnovější technologie – fine time measurement. Všechna tato témata se věnují lokalizaci zařízení pomocí vysílání signálu a budou rozvedena detailněji, aby čtenáři poskytla klíčové znalosti, které budou rozvedeny detailněji v teoretické části a jsou nezbytné pro pochopení praktické části. Praktická část se věnuje pochopení volně přístupné aplikace vyvíjené společností Google pro mobilní zařízení s Android 9 a vyšší. Tato verze Androidu disponuje funkcí fine time measurement s podporou pro WiFi adaptér 802.11mc. Po detailnějším pochopení této aplikace se práce zaměřuje na její úpravu tak, aby se dala aplikace dále rozšířit o metody lokalizace, jako jsou například trilaterace, či multilaterace.

Klíčová slova: android, fine time measurement, lokalizace, round trip time, trilaterace, WiFi, 802.11mc

Annotation

Title: WiFi localization using 802.11mc Fine Time Measurement / RTT

This work introduces readers to the topics related to radio localization. Furthermore, the division of radio location such as received signal strength indication, its use and optimization, time of flight method and the latest technology – fine time measurement. All these topics are devoted to the localization of the device by means of signal transmission and will be elaborated in more detail in order to provide the reader with the key knowledge which will be elaborated in more detail in the theoretical part this knowledge will be necessary for understanding the practical part. The practical part is devoted to understanding the freely accessible application developed by Google for mobile devices with Android 9 and higher. This version of Android has a fine time measurement function with support for an 802.11mc WiFi adapter. After a more detailed understanding of this application, the work focuses on its modification so that the application can be further extended by localization methods, such as trilateration or multilateration.

Keywords: android, fine time measurement, localization, round trip time, trilateration, WiFi, 802.11mc

Seznam zkratek

AP – Access Point – přístupový bod

FTM – fine time measurement – technologie lokalizace

IEEE 802.11mc – standard podporující WiFi RTT

RSSI – Received Signal Strength Indication – indikátor síly signálu

RTT – Round-trip time – doba letu signálu

Obsah

Seznam obrázků

Seznam tabulek

1 Úvod

Každý z nás se setkal s pojmem GPS nebo jiným druhem lokalizace čehokoliv. Také každý z nás v dnešní době u sebe nosí nějaké mobilní zařízení, které je určitým způsobem využíváno k lokalizaci. Většinou je lokalizace spojena právě s GPS systémy, kdy hledáme konkrétní cestu z bodu A do bodu B. Pokud se ale dostaneme do prostředí, kde GPS signál není dostupný, tak lokalizace není možná. Tento problém nastává například v budovách nebo v uzavřených prostorech, kam GPS signál nemá šanci proniknout. Z tohoto důvodu vznikla myšlenka vytvořit vnitřní lokalizační systém. S tím však přichází otázka, jak něčeho takového docílit? Odpověď je celkem jednoduchá. K vnitřní lokalizace se použije zařízení, které má skoro každý z nás doma. Tímto zařízením je AP – přístupový bod. Pokud dokážeme určit vzdálenost mezi jedním AP a mobilním zařízením, tak pak již není problém pomocí známých metod, jako je trilaterace, multilaterace nebo jiného známého systému určovat polohu tohoto mobilního zařízení. Všechny tyto otázky byly vyřešeny s příchodem standardu IEEE 802.11mc, který disponoval technologií RTT (Round-trip time). Pokud máme AP, které má tuto funkcionalitu, a přidáme k tomu mobilní zařízení s Android 9 či IOS 12, pak máme vše potřebné k tomu, abychom dokázali vytvořit systém pro lokalizaci i ve vnitřních prostorách.

Praktická část je rozdělena do dvou částí. Nejprve pochopení Google Sample aplikace a poté testování možností FTM (fine time measurement) v této aplikaci, následně pak úprava a testování nově upravené aplikace. Pro praktickou část bude použit sample kód přímo od společnosti Google, který bude dále upravován, aby se aplikace rozšířila o další funkce, jako například trilaterace atd. Celý program je vytvořený v Android Studiu od Intellij.

2 Cíl práce

Cílem této bakalářské práce je seznámit čtenáře s problematikou lokalizace a jejího využití v reálném světě. První část práce bude zaměřena na teorii kolem lokalizace. Bude vysvětleno, co je to lokalizace, jaké jsou klíčové pojmy ohledně tématu lokalizace, jaké lokalizační metody se při lokalizaci využívají, jak tyto metody fungují.

V praktické části bude čtenář seznámen s aplikací vyvíjenou společností Google. Tato aplikace se zaměřuje na téma lokalizace, popisuje a umožňuje měřit vzdálenosti mobilního zařízení a AP za pomoci FTM RTT. Aplikace zde bude stručně vysvětlena, především způsob, jakým pracuje s daty.

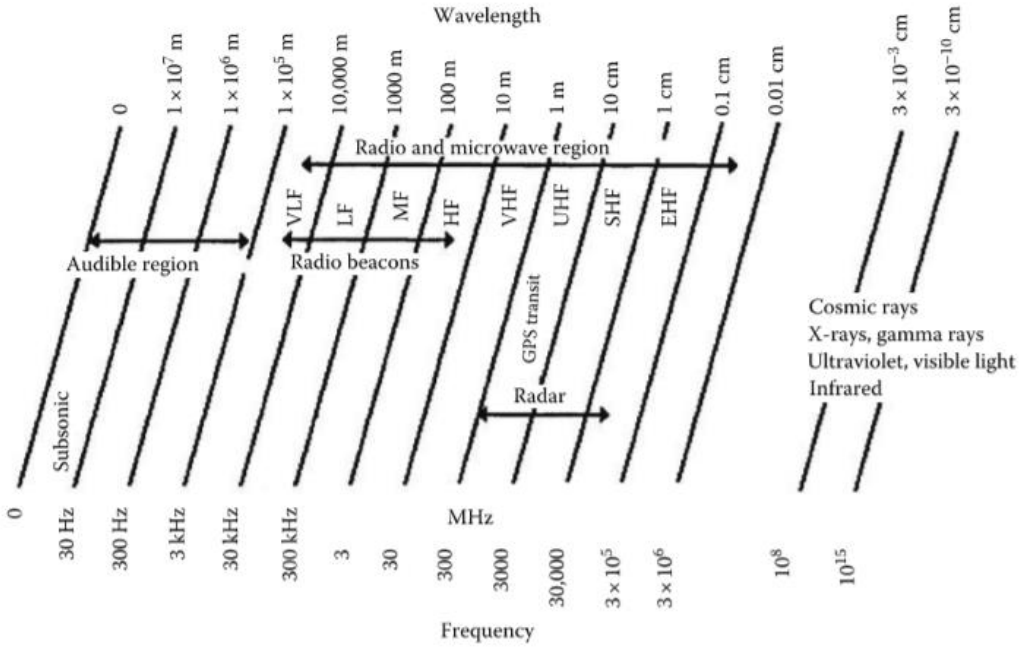
Posledním cílem této práce je otestovat tuto aplikaci a zanalyzovat její výstupy. Následně bude aplikace upravována takovým způsobem, aby byla schopna co nejpřesněji měřit pozici mobilního zařízení v souřadnicové síti, na které budou umístěny AP. Cílem práce je ukázat možnosti a nedostatky této technologie a také otestovat, zda Googlem deklarovaný chybový okruh 2 m je reálný, či výsledky nebudou v této toleranci.

3 Radio lokalizace

Radio lokalizace je proces hledání nějakého subjektu pomocí radiových vln. Je to technologie široce využívána vojenskými, dopravními a spotřebitelskými aplikacemi. Klíčovým prvkem jsou zde elektromagnetické radiové vlny. [1]

3.1 Elektromagnetické radiové vlny

Funkce elektromagnetického radiového vlnění v naší atmosféře je nezbytně důležitou součástí při tvorbě a navrhování různých satelitních zařízení, radarů až po jednoduchá každodenní zařízení, jako je mobilní telefon. [1]
Radiová vlna vytvořena v naší atmosféře putuje směrem ven od zdroje jejího vzniku. Při putování této vlny dochází k jejímu rušení, odrážení od různých povrchů, se kterými se setkává. K absorpci a rozptylu vln dochází díky excitaci elektronů uvnitř molekul v propagačním médiu. Dále chování vlny závisí na její frekvenci a vlnové délce. To znamená, že každá vlna se chová trošku jiným způsobem. Na následujícím obr. 1 je vidět, jak se dělí elektromagnetické vlny podle frekvence. Základní trojí dělení je na zvukové vlny (0- 3kHz), radiové vlny (5kHz – 300GHz) a viditelné světlo (300GHz a víc). Pro zjednodušení se radiové vlny dále dělí do osmi podkategorií a to VLF (velmi nízká frekvence <30kHz), LF (nízká frekvence 30 - 300kHz), MF (střední frekvence 300Khz-3MHz), VF (vysoká frekvence 3 - 30MHz), VHF (velmi vysoká frekvence 30 - 300MHz), UHF (ultra vysoká frekvence 300MHz – 3GHz), SHF (super vysoká frekvence 3 - 30GHz) a EHF (extrémně vysoká frekvence 30 - 300GHz). [1]



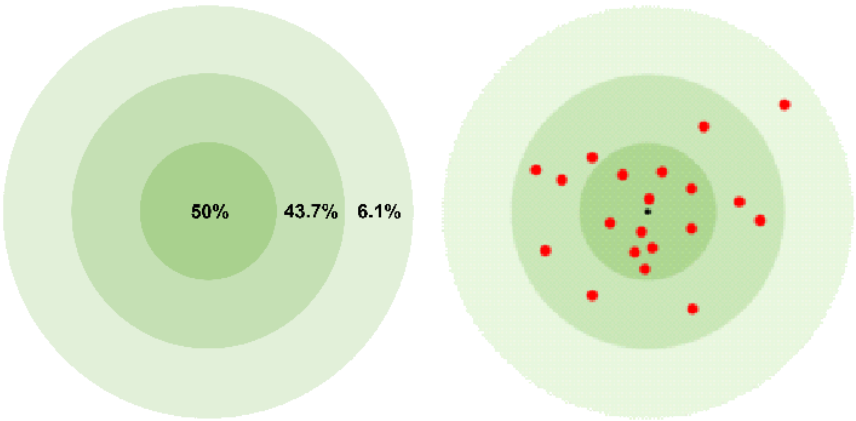
Obrázek 1: Rozdělení frekvenčních pásem [1]

3.2 Přesnost radio lokalizace

Existuje obrovská škála náhodných efektů, které ovlivňují přesnost určování polohy pomocí radiové vlny. Například atmosférická turbulence podél přenosové cesty, chyby v přijímačích či vysílačích, slábnutí signálu, odrazy atd. Z tohoto důvodu nedochází k určení přesné polohy, ale spíše k určení bodů, které jsou poblíž hledané polohy. Dvě běžně používané míry přesnosti jsou CEP (circular error probable) a RMS (root mean square). [1]

3.2.1 CEP (circular error probable)

CEP je metoda pravděpodobné kruhové chyby. Jedná se o kruh nakreslený v místě skutečné polohy. Pravděpodobnost kruhové chyby je definována jako poloměr kružnice, kde pravděpodobnost, že je hledaný bod uvnitř, je 50 %, jak je ukázáno na obr.2. Aby se dosáhlo sloučení různých testovacích informací, byly při výpočtu CEP rozsáhle studovány Bayesovské metody a vylepšené Bayesovské metody. Tyto metody by však mohly selhat, pokud v předcházejících informacích existuje neznámá systematická chyba. [1]



Obrázek 2 - Pravděpodobnost kruhové chyby

3.2.2 RMS (root mean square)

RMS hodnota sady hodnot (nebo průběžného tvaru vlny) je druhá odmocnina aritmetického průměru čtverců hodnot nebo druhá mocnina funkce, která definuje souvislý průběh. [1]

$$rms = \sqrt{\sum_{n=1}^{N} \frac{(E_n)^2}{N}}$$

Kde:

E je vzdálenost mezi skutečnou a odhadovanou polohou,

N je počet odhadnutých pozic. [1]

3.3 Druhy radio lokalizace

Setkáváme se s dvěma základními způsoby radio lokalizace, a to s aktivním a pasivním. [2]

V aktivním režimu se pro určení polohy používá snímání odrazů signálů od objektů. To tedy v praxi znamená, že hledaný subjekt odráží námi vyslaný signál a my zachytíme jeho odraz a tím získáme vzdálenost objektu a tím i jeho polohu. Aktivní režim může také naslouchat, takže pokud nějaký subjekt sám vysílá rádiové vlny, tak my je dokážeme zpracovat. Dalším případem je již zmíněný pasivní režim, což je vysokofrekvenční přijímač, který „poslouchá“ mikrovlnné a jiné

vysokofrekvenční přenosy. Může například určit přítomnost letadla, které vyzařuje vysokofrekvenční šum (aktivní radar, komunikace, rozptýlený únik RF), a může s lehkou nepřesností určit dráhu tohoto letadla. S více přijímači může určit polohu, dosah a rychlost letadla pomocí triangulace. [2]
Většinou se s termínem radio lokalizace setkáme v souvislosti s radarem. Radio lokalizace spočívá v tom, že vyšleme radiové vlny, které se odráží od okolí zpátky k nám, a tím nám předávají informace o poloze jednotlivých objektů kolem nás. Radar je hojně využívaný ve vojenském prostředí. [1]

Existuje mnoho metod pro radiovou lokalizaci:

- RSSI – received signal strength,
- TOA – time of arrival,
- DTOA – differential TOA,
- AOA – angle of arrival,
- GPS – global positioning systems,
- a další... [1].

3.3.1 RSSI – Received Signal Strength Indication

Technika určování vzdálenosti pomocí RSSI měří vzdálenost mezi uzly tak, že je opakovaně vysílaný radiový signál, u kterého je měřena síla, čím je objekt vzdálenější, tím je síla signálu nižší. Vztah mezi přijatou silou signálu (RSSI) a vzdáleností (d) je vyjádřen následující rovnicí, kde n představuje konstantu šíření signálu (koeficient šíření). A představuje sílu vyslaného signálu, když je vzdálená 1 m od odesílatele. [3]

$$RSSI = A - 10 * n * lg(d)$$

RSSI s použitím průměru hodnot

V reálném světě je hodnota RSSI ovlivňována okolním prostředím. Pro zlepšení RSSI je nezbytné, abychom nebrali pouze jednu naměřenou hodnotu, ale abychom zprůměrovali více měření do jedné hodnoty. V tomto případě se řídíme rovnicí, která vypadá následovně [3]:

$$RSSI = \frac{1}{m}\sum_{i=1}^{m} RSSI$$

Zavedení Gaussovského filtrovacího modelu

Ačkoliv je průměrovaná hodnota RSSI relativně přesná, můžeme ji dále optimalizovat pomocí Gaussovského filtrovacího modelu. Gaussovský filtrovací model pracuje s pravděpodobnostmi. Na zařízení přijde několik RSSI ze stejné pozice (n) a filtrovací model se rozhoduje, který RSSI je na základě předchozích měření pravděpodobný a který ne. Nevyhovující RSSI jsou zahazovány a ponechány jsou jen RSSI s vysokou pravděpodobností výskytu. Gaussova distribuční funkce vypadá následovně [3]:

$$f(x) = \frac{1}{\sqrt{2\pi\sigma}} e^{\frac{(x-u)^2}{2\sigma^2}};$$

$$u = \frac{1}{n}\sum_{i=1}^{1} x_i\,;$$

$$\sigma^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - u)^2\ ;$$

Abychom docílili maximálního využití potenciálu Gaussovského modelu, měla by se naše pravděpodobnostní funkce pohybovat někde mezi 0.6 a 1 tudíž by její přepis vypadal následovně [3]:

$$0.6 \leq f(x) \leq 1$$

Po úpravě základního vzorce a dosazení těchto veličin se dostaneme zhruba k tomuto rozsahu [3]:

$$0.15\sigma + u \leq x \leq 3.09\sigma + u$$

Kde pro dopočítání **σ** a **u** použijeme následující vzorce:

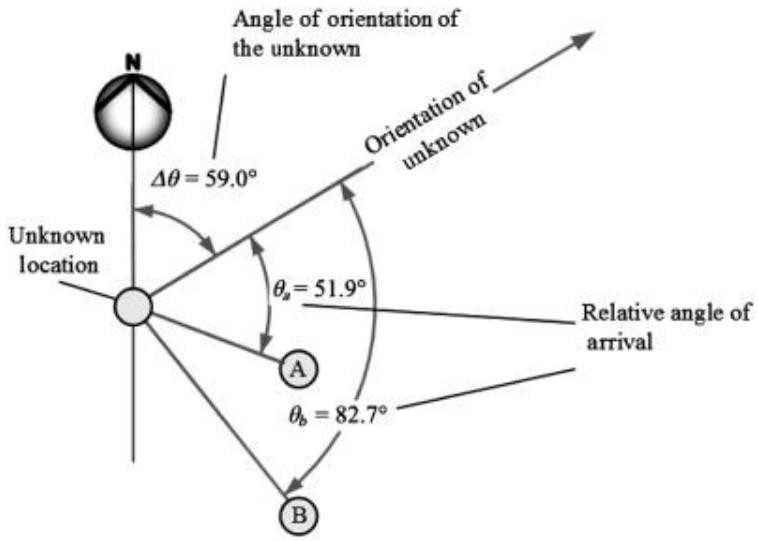
$$\sigma = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(RSSI_i - \frac{1}{n}\sum_{i=1}^{n}(RSSI_i)^2}$$

$$u = \frac{1}{n}\sum_{i=1}^{n} RSSI$$

Po zavedení Gaussovského filtrovacímu modelu dojde ke snížení zpracovávání nepravděpodobných hodnot a tím i ke zpřesnění výpočtu a v tom případě i ke zpřesnění lokalizace nějakého objektu. Proto je tato strategie hojně využívána v současných zařízeních. [3]

3.3.2 AOA – Angle of arrival

Úhel dopadu je dalším způsobem, jak získat přibližnou polohu nějakého objektu. Je to metoda pro určování směru šíření radiové vlny dopadající na soustavu antén, nebo určení z maximální síly dopadu signálu na anténu, která se otáčí. Úhel dopadu je měřen ve stupních a určuje se ve směru hodinových ručiček, přičemž 0° úhel, také nazývaný absolutní, vždy míří na sever. Kombinace úhlu dopadu signálu se známým umístěním více přijímačů poskytne přibližnou polohu, nebo alespoň směr vysílače. Na obr. 3 je vidět znázornění průběhu metody AOA. [4]



Obrázek 3 - Výpočet AOA [4]

3.3.3 TOA – Time of arrival

Jedná se o výpočet času letu radiového signálu od odesílatele k přijímači. Tento časový signál je označen časovým údajem a v přijímači je tento údaj použit pro výpočet vzdálenosti. Vysílač i přijímač by měly být synchronizovány, aby se zabránilo chybám způsobeným různým nastavení hodin na zařízení.

Synchronizace není potřeba pokud používáme RTT TOA. Vzdálenost je pak počítána tak, že obdržené TOA vynásobíme rychlostí [4]:

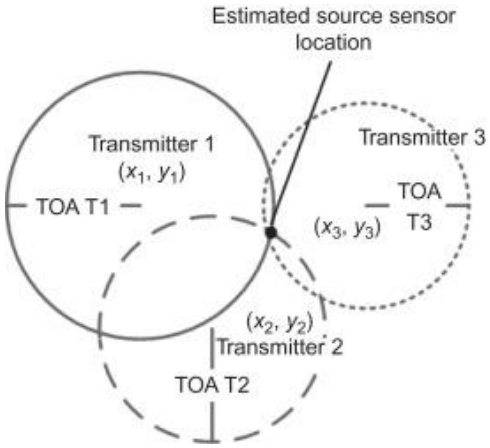
$$\Delta Distance = \Delta Time(t_r - t) \times Velocity$$

$\Delta Distance$: je námi hledaná vzdálenost [4]
$\Delta Time(t_r - t)$ je rozdíl mezi časem příjezdu do přijímače (t_r) a zdrojovým časem (t) [4]
$Velocity$: Rychlost světla [4]

S využitím Pythagorovy věty pro souřadnicový systém je vzdálenost mezi pozicí příjemce na souřadnicích (x_r, y_r) a zdrojovou pozicí (x, y,) dána [4]

$$Velocity \times (t_r - t) = \sqrt{(x_r - x)^2 + (y_r - y)^2}$$

Klíčovým prvkem při TOA je synchronizace času na všech kotevních zařízeních. Zařízení musí mít stejný časový údaj, aby poté výpočet polohy našeho zařízení byl správný. Toho lze dosáhnout pomocí protokolu IEEE 1588 – Precision time protocol. Obr. 4 znázorňuje spojení metody TOA s metodou výpočtu polohy pomocí trilaterace. [5]

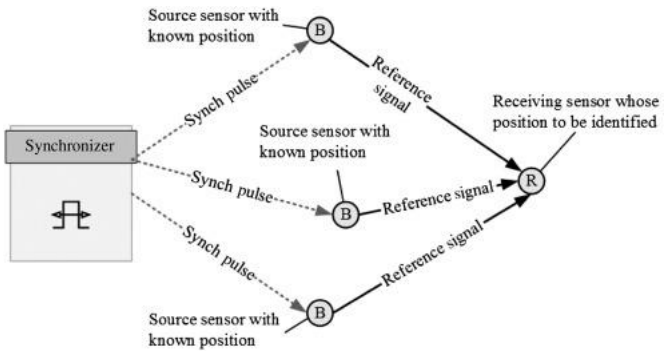


Obrázek 4 - TOA a výpočet pomocí trilaterace [5]

3.3.4 TDOA – Time Difference of Arrival

Jedná se o schéma, které řeší rozdíl času letu signálu od jednoho zdroje k tzv. kotvám či naopak. Kotva je pro nás přijímač, jehož poloha je nám dobře známá. Odesílatel vysílá radiové signály, zatímco přijímač zaznamenává čas jejich příchodu, jak je zobrazeno na obr. 5. Jakmile dokončíme měření signálu a získáme tak naměřená data, je možné vypočítat rozdíl v rozsahu mezi cílovým zařízením a kotvami. Z těchto dat poté složíme hyperbolické rovnice, kde po jejich vyřešení získáváme souřadnice cíle. [6]

Kritickým úkolem při lokalizaci založené na TDOA je přesně změřit TDOA mezi kterýmkoliv párem různých kotev. Obecně tato metoda potřebuje alespoň tři (čtyři) kotvy k lokalizaci cíle ve dvourozměrném (trojrozměrném) prostoru. Jako v předchozím případě je pro nás důležitá časová synchronizace všech zařízení. [6]



Obrázek 5 - Znázornění TDOA [6]

4 WiFi – Fine time measurement

Určování polohy se stalo běžnou součástí našich každodenních zařízení. Jelikož žijeme ve světě, kde máme přístupnou WiFi na každém kroku, tak si vývojáři položili otázku, proč pro lokalizaci nepoužívat právě WiFi. V současné době se WiFi lokalizace používá společně s GPS systémem v našich mobilních zařízeních. Avšak ačkoliv je tato technologie tak hojně využívána, co se týče pouze lokalizace ve vnitřních prostorách, stále se potýkáme s nepřesnými lokalizacemi a mnohými problémy. [7]

Tyto problémy začaly být řešeny v roce 2016 s příchodem standardu IEEE 802.11-016, kde bylo poprvé představeno FTM. Tato technologie umožňuje lokalizaci s metrovou přesností uvnitř budov. FTM určuje vzdálenost mezi dvěma zařízeními, např. mezi AP a smartphonem tak, že měří čas, který potřebuje radiový signál na cestu z jednoho zařízení na druhé. Před touto technologií se poloha určovala pomocí síly signálu (RSSI), tato technologie měla bohužel velmi nízkou přesnost, a proto se začala hledat vhodná alternativa, kterou se zatím stalo FTM. [7]

4.1 Technologie FTM

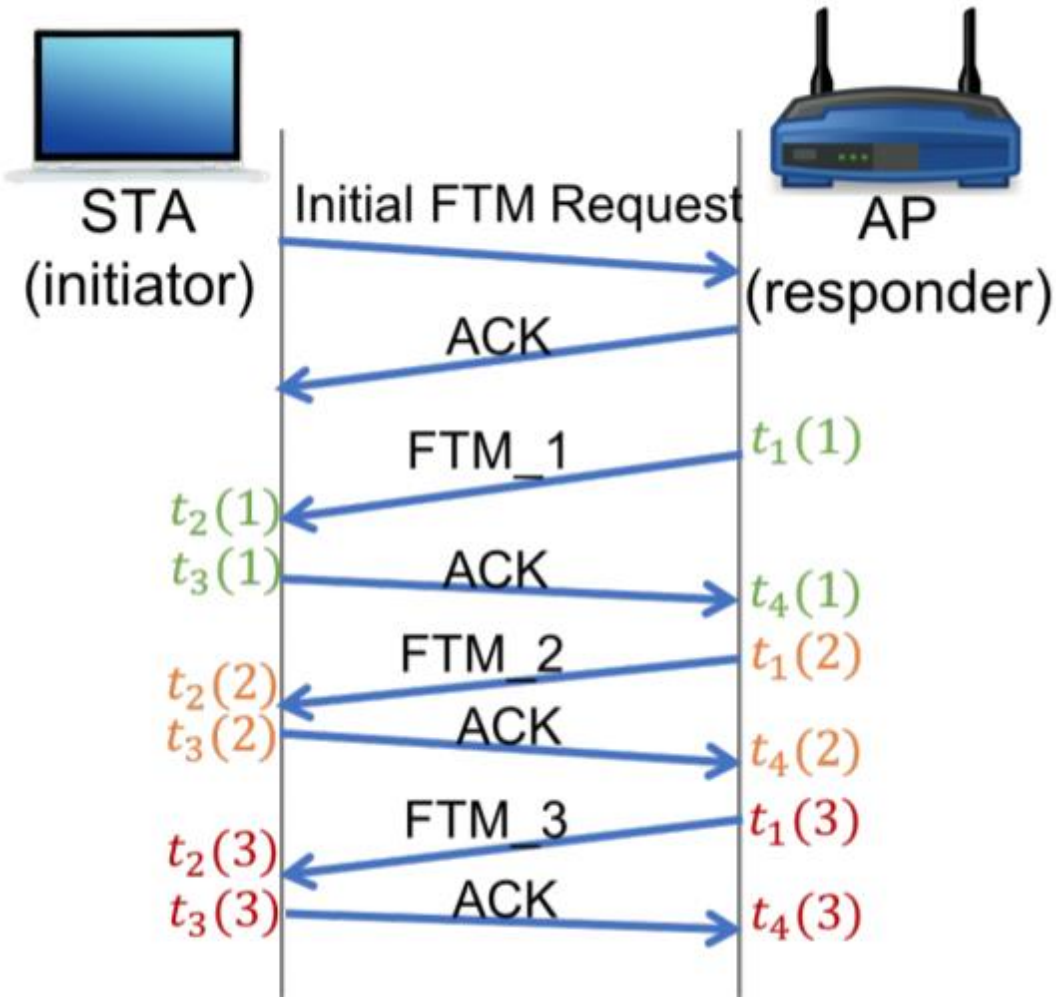
Obecně platí, že WiFi time of arrival odhaduje vzdálenost měřením doby zpáteční rychlosti signálu mezi stanicí a přístupovým bodem. TOA je lineárně závislá na dosahu. Když si představíme ideální prostředí, kde mezi vysílačem a přijímačem není žádná překážka, tak se dá očekávat, že měření vzdálenosti pomocí časové složky by mělo být dokonalé. [8]

Iniciátorem komunikace je stanice (STA), která započne proces tím, že vyšle požadavek FTM na přístupový bod (AP). AP se rozhodne, zda podporuje FTM a posílá odpověď, zda souhlasí, či zamítá měřící proces. V případě, že zařízení navážou spolupráci, začne AP vysílat FTM zprávy a čeká na odpověď ACK od STA. V tomto kroku se již začíná odhadovat RTT na základě časové značky přenosu zprávy FTM a odpovědi ACK. [8]

Obecně protokol vylučuje dobu zpracování STA tak, že od celkového RTT (t4-t1), které reprezentují odeslaní první FTM (t1), do momentu, kdy je přijat poslední ACK (t4), se odečtou mezihodnoty (t3-t2). Tento proces se opakuje pro všechny

FTM-ACK zprávy a finální RTT je vlastně průměrem těchto hodnot, které jsou vyslány v jednom komunikačním kroku. [8] Názorné zobrazení této komunikace je vidět na obr. 6.

$$RTT = \frac{1}{n}(\sum_{k=1}^{n} t_4(k) - \sum_{k=1}^{n} t_1(k)) - \frac{1}{n}(\sum_{k=1}^{n} t_3(k) - \sum_{k=1}^{n} t_2(k))$$

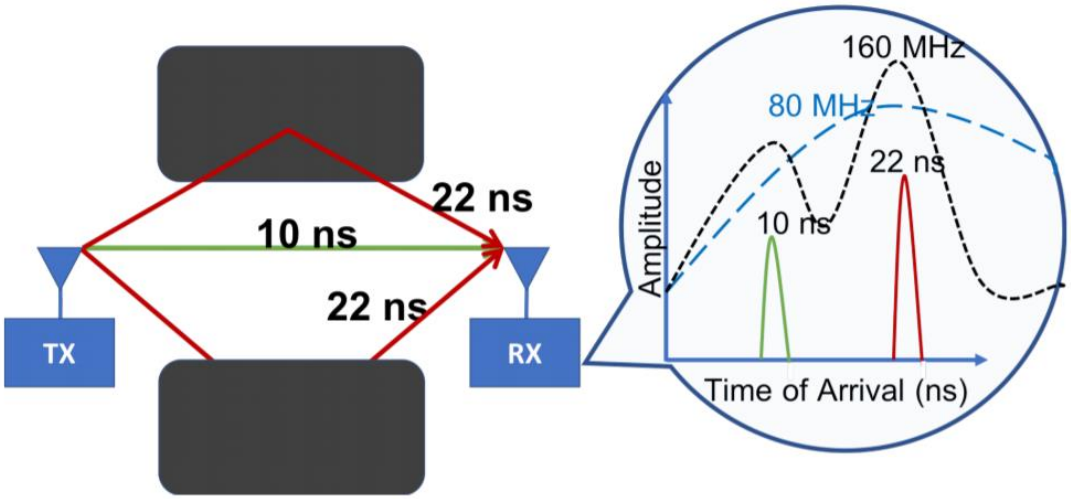


Obrázek 6 - Komunikace mezi AP a mobilním zařízením [8]

4.2 Problém odrazu – Multipath problem

Největším problémem všech měřících systémů je právě problém odrazu. Problém spočívá v tom, že vyslaný signál se odráží v prostředí do různých směrů a ve výsledku může dojít k tomu, že k přijímači přijde tento signál vícekrát z různých cest. Každý takto odražený signál pro nás má jiné výsledky, což může zkreslit

výsledek konečný. Proto je pro nás důležité rozlišovat RTT, které je založeno na přímé cestě, nebo na odrazu. Pokud se rozhodneme používat pouze přímé cesty, tak jednoduše vezmeme ten signál, který se k přijímači dostane jako první a ostatní zahodíme. Cesta s odrazy je poněkud horší, protože se musíme rozhodovat, které signály jsou přímé a ty musíme z měření vyloučit. Šíření signálu více cestami a odrazy ilustruje obr. 7. [8]



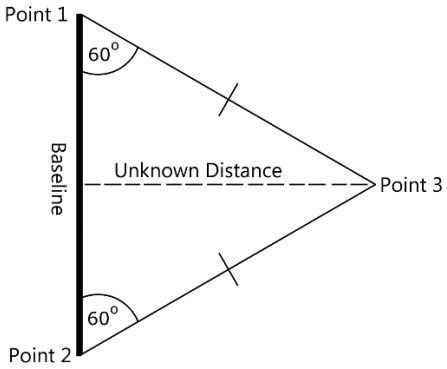
Obrázek 7 - Odraz signálu [8]

5 Metody určování polohy

5.1 Triangulace

Jedná se o metodu založenou na trigonometrii. Dále musíme znát alespoň dvě kotvy, podle kterých dopočítáme námi hledaný bod, tento proces je znázorněn na obr. 8. [9]

Typickými uživateli triangulace jsou zeměměřiči, kteří používají teodolity, což je nástroj pro měření úhlu. Vždy se postaví do jednoho známého bodu, nastaví úhel a vytvoří tak pomyslně trojúhelník, ze kterého se dá určit hledaná pozice. [9]



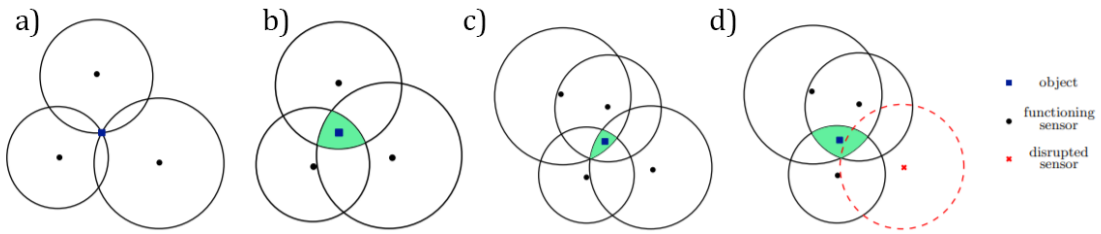
Obrázek 8 – Triangulace [9]

Jako každá metoda, tak ani triangulace nedokáže poskytovat zcela přesné výsledky, jelikož získané informace se signálů mohou být nepřesné nebo zkreslené z důvodu překážek atd. [9]

5.2 Trilaterace

Trilaterace, na rozdíl od triangulace pracuje se vzdálenostmi, nikoliv s úhly. Jedná se asi o nejpopulárnější matematické řešení pro lokalizaci konkrétního objektu. Princip trilaterace je v tom, že měříme vzdálenost objektu od nám známých třech statických objektů. Průnikem těchto tří kružnic získáme pozici námi hledaného objektu. (obr. 9a) Ovšem pozice objektu není úplně přesná, potýkáme se zde s rozptylem signálu, či blokováním signálu. V těchto případech dochází k nepřesným výpočtům a vzniká nám chyba, se kterou musíme počítat. (obr. 9b)

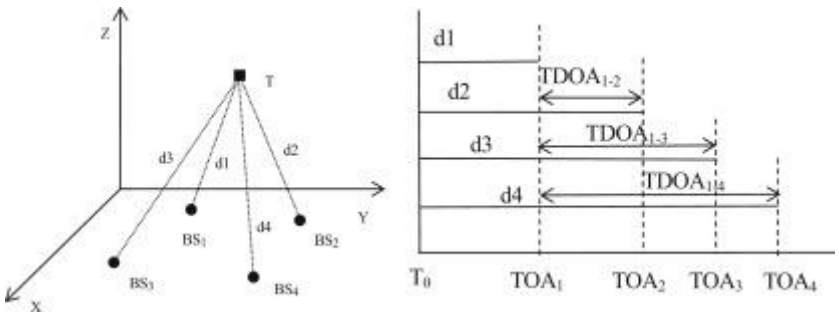
Nikdy nedokážeme vytvořit zcela dokonalé prostředí. Tudíž nám vzniká pouze přibližná oblast polohy objektu. Způsob, jakým dokážeme snížit chybu při výpočtu, je ten, kdy získáme další signál z dalšího zařízení, který nám zúží oblast, ve které se hledané zařízení nachází. (obr. 9c) Z toho nám vyplývá, že účinnost polohovacího systému nezávisí pouze na kvalitě použitých zařízení, ale také na pokrytí oblasti signály. Čím více signálů dokážeme získat, tím přesněji dokážeme vypočítat polohu námi hledaného zařízení. To ovšem přináší další problémy a to, jak určit hranici, kolik zařízení použít, aby systém byl co nejefektivnější a finančně co možná nejpřijatelnější. [10]



Obrázek 9 – trilaterace [10]

5.3 Multilaterace

Multilaterace je vlastně trilaterace, ale místo tří orientačních bodů máme alespoň čtyři tyto body. S přidáním dalšího takového bodu se přesnost určení naší pozice zpřesňuje, avšak výpočet je s každou takto přidanou hodnotou složitější. Multilaterace odhaduje stejně jako trilaterace umístění cíle. To vyžaduje, aby cíl aktivně vysílal elektromagnetické signály, které jsou pak přijímány mnoha senzory na zemi. Na základě TOA od kotev k zařízení je poté možnost určit polohu tohoto zařízení. Ovšem častějším metodou zpracování těchto signálů je TDOA znázorněna na obrázku 10. [15][16]



Obrázek 10 - Příklad multilaterace [16]

6 Zařízení schopné využívat technologii FTM

Abychom mohli využívat technologii FTM, je potřeba mít zařízení podporující IEEE 802.11mc.

6.1 IEEE 802.11mc

Jedná se o standard, který vylepšuje funkcionalitu RTT. RTT bylo vylepšeno o funkci FTM, který zvýšil rozsah časového razítka od 10ns do 100ps. Teoreticky by se díky tomuto vylepšení mohlo docílit v ideálním prostředí přesnosti z předchozích 3 m, které odpovídalo staré verzi RTT na pouhé 3 cm u verze nové. Avšak ideálního prostředí nelze reálně dosáhnout, tudíž se v reálném světě zlepšil stav do řádu několika metrů.[11]

6.2 Mobilní zařízení

Kompatibilní zařízení se standardem 802.11mc by měla být všechny zařízení, která disponují hardwarem podporujícím tuto technologii a také jsou vybavena systémy Android 9 a vyšší a IOS od verze 12. Prvním průkopníkem v OS mobilních zařízení s využíváním této technologie byl Google. Google s příchodem Android 9 Pie poprvé ukázal možnosti vylepšené RTT o FTM. První zařízení, které si mohli uživatelé sami otestovat možnosti RTT FTM, byly produkty Google Pixel (obr. 11), které dostaly aktualizaci na Android P jako první, tak jak už u Google bývá zvykem.

Obrázek 11 - Google Pixel 3a XL

6.3 AP

V současné době většina nasazených AP nedisponuje podporou standardu 802.11mc. Avšak některá zařízení se přeci jen najdou. Nejsnazším příkladem je Google WiFi (obr. 12). Google jako první představil, jaké možnosti tento standard umožňuje.

Obrázek 12 - Google WiFi

Další alternativou po Google WiFi je produkt společnosti Compulab. Celým názvem Compulab WiFi Indoor Location Device (WILD) (obr. 13). Toto zařízení se chlubí právě tím, že podporuje RTT. Na stránkách výrobce dokonce najdete jednu z dalších ukázkových aplikací volně ke stažení. Mezi přednosti tohoto zařízení patří předinstalované:

- Debian GNU/Linux,
- Intel AC8260 WiFi driver supporting 802.1 1mcFTM Responder mode,
- WILD utilities.

Obrázek 13 - Compulab WILD

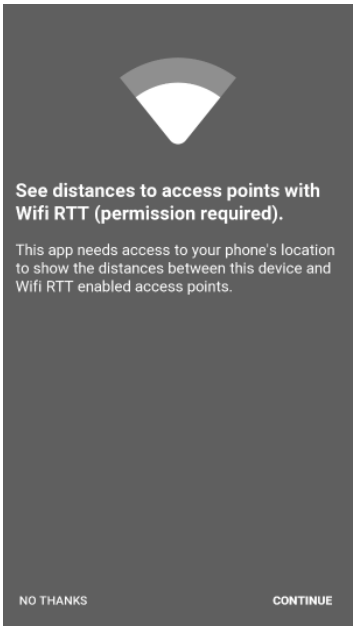
7 Google sample aplikace

Pro testování možnosti RTT s FTM byla použita Google sample aplikace přímo vytvořena pro RTT. Sample je volně ke stažení z Githubu. Aplikace je vytvořena čistě pro Android, proto je dobré zdrojové soubory otevřít ve vývojovém prostředí, které podporuje vývoj aplikací pro Android. Pro práci bylo použito Android studio od Intellij, jelikož je toto prostředí intuitivní a uživatelsky přívětivé.

Po otevření projektu a rozbalení všech položek menu je jasné, že je aplikace dělena na dvě hlavní části. Jednou z hlavních části je podsložka „res“, kde nalezneme spíše vizuální stránku aplikace, tudíž se jedná spíše o frontendové části aplikace. Druhou částí je podsložka „java“. V této podsložce se nachází naše backendové prvky, tudíž je zde vidno, jak aplikace funguje. Obě dvě tyto podsložky si dále rozebereme více do detailu, abychom pochopili, jakým způsobem tato aplikace funguje.

7.1 LocationPermissionRequestActivity

Jedná se o jednoduchou třídu, která se uživatele po prvním spuštění aplikace zeptá, zda aplikace může používat potřebné prvky pro zjišťování polohy (obr. 14). Pokud se uživatel rozhodne, že aplikaci nedovolí využívat tyto prvky, pak je aplikace ukončena, protože bez těchto prvků je tato aplikace nepoužitelná.



Obrázek 14 – Povolení sdílení polohy

Dotaz zjišťující, zda uživatel povolil využívání získávání polohy, se provede vždy při startu aplikace (viz obr. 15 a 16). Když je aplikace již nainstalována v telefonu a uživatel provedl povolení využívání sdílené polohy, tak si aplikace pamatuje to, že povolení jí bylo již uděleno, dále se již ptát nebude a rovnou vstoupí do aplikace.

```
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        // If permissions granted, we start the main activity (shut this activity down).
        if (ActivityCompat.checkSelfPermission( context: this, permission.ACCESS_FINE_LOCATION)
                == PackageManager.PERMISSION_GRANTED) {
            finish();
        }

        setContentView(R.layout.activity_location_permission_request);
    }

    public void onClickApprovePermissionRequest(View view) {
        Log.d(TAG,  msg: "onClickApprovePermissionRequest()");

        // On 23+ (M+) devices, fine location permission not granted. Request permission.
        ActivityCompat.requestPermissions(
                activity: this,
                new String[] {Manifest.permission.ACCESS_FINE_LOCATION},
                PERMISSION_REQUEST_FINE_LOCATION);
    }

    public void onClickDenyPermissionRequest(View view) {
        Log.d(TAG,  msg: "onClickDenyPermissionRequest()");
        finish();
    }
```

Obrázek 15 - Ukázka kódu 1

```
@Override
public void onRequestPermissionsResult(
        int requestCode, @NonNull String[] permissions, @NonNull int[] grantResults) {

    String permissionResult =
            "Request code: "
                    + requestCode
                    + ", Permissions: "
                    + permissions
                    + ", Results: "
                    + grantResults;
    Log.d(TAG,  msg: "onRequestPermissionsResult(): " + permissionResult);

    if (requestCode == PERMISSION_REQUEST_FINE_LOCATION) {
        // Close activity regardless of user's decision (decision picked up in main activity).
        finish();
    }
}
```

Obrázek 16 - Ukázka kódu 2

7.2 AcessPointRangingResultsActivity

Nejpodstatnější třídou je právě tato. Důvodem je, že právě tato třída zpracovává veškeré potřebné informace o AP. Uživatel vybere AP ze seznamu zařízení, které podporují RTT, a právě o tomto vybraném zařízení začne tato třída sbírat informace.

7.2.1 Ranging request

Metoda inicializuje Ranging request (obr. 17) a to tak, že se nejdříve zeptá, jestli uživatel povolil funkce získávání polohy, a poté sestaví ranging requests tím, že se vytvoří nová instance ranging requestu, na který se zavolá builder, kterému se předá AP ze mScanResults a na to celé se zavolá build. Poté stačí už jen zavolat funkci startRanging nad wifiManagerem a přidat mu potřebné parametry včetně ranging requestu a callbacku. Callback je třída, která zpracovává zpětná volání pro všechny RangingRequests a vydává nové RangingRequests v časovém intervalu, který se dá nastavit

```
private void startRangingRequest() {
    // Permission for fine location should already be granted via MainActivity (you can't get
    // to this class unless you already have permission. If they get to this class, then disable
    // fine location permission, we kick them back to main activity.
    if (ActivityCompat.checkSelfPermission( context: this, permission.ACCESS_FINE_LOCATION)
            != PackageManager.PERMISSION_GRANTED) {
        finish();
    }

    mNumberOfRangeRequests++;

    RangingRequest rangingRequest =
            new RangingRequest.Builder().addAccessPoint(mScanResult).build();

    mWifiRttManager.startRanging(
            rangingRequest, getApplication().getMainExecutor(), mRttRangingResultCallback);
}
```

Obrázek 17 - Ukázka kódu 3

7.2.2 Ranging results

Pokud se vyskytnout nějaké ranging results, které nejsou null, tak právě tato metoda získá data uložená v ranging result a přiřadí je do jednotlivých proměnných, které se poté dále propíšou do uživatelského rozhraní a vypíšou zde námi požadovaná data. Metoda je zobrazena na obrázku 18.

```
@Override
public void onRangingResults(@NonNull List<RangingResult> list) {
    Log.d(TAG,  msg: "onRangingResults(): " + list);

    // Because we are only requesting RangingResult for one access point (not multiple
    // access points), this will only ever be one. (Use loops when requesting RangingResults
    // for multiple access points.)
    if (list.size() == 1) {

        RangingResult rangingResult = list.get(0);

        if (mMAC.equals(rangingResult.getMacAddress().toString())) {

            if (rangingResult.getStatus() == RangingResult.STATUS_SUCCESS) {

                mNumberOfSuccessfulRangeRequests++;

                mRangeTextView.setText((rangingResult.getDistanceMm() / 1000f) + "");
                addDistanceToHistory(rangingResult.getDistanceMm());
                mRangeMeanTextView.setText((getDistanceMean() / 1000f) + "");

                mRangeSDTextView.setText(
                        (rangingResult.getDistanceStdDevMm() / 1000f) + "");
                addStandardDeviationOfDistanceToHistory(
                        rangingResult.getDistanceStdDevMm());
                mRangeSDMeanTextView.setText(
                        (getStandardDeviationOfDistanceMean() / 1000f) + "");

                mRssiTextView.setText(rangingResult.getRssi() + "");
                mSuccessesInBurstTextView.setText(
                        rangingResult.getNumSuccessfulMeasurements()
```

Obrázek 18 - Ukázka kódu 4

7.2.3 Ostatní metody

Zbylé metody v této třídě se zabývají dodatečnými výpočty pro různé druhy vzdálenosti, resetovaní dat atd. Některé tyto metody je možné vidět na obrázku 19.

```
private void resetData() {
    mSampleSize = Integer.parseInt(mSampleSizeEditText.getText().toString());

    mMillisecondsDelayBeforeNewRangingRequest =
            Integer.parseInt(
                    mMillisecondsDelayBeforeNewRangingRequestEditText.getText().toString());

    mNumberOfSuccessfulRangeRequests = 0;
    mNumberOfRangeRequests = 0;

    mStatisticRangeHistoryEndIndex = 0;
    mStatisticRangeHistory.clear();

    mStatisticRangeSDHistoryEndIndex = 0;
    mStatisticRangeSDHistory.clear();
}
```

Obrázek 19 - Ukázka kódu 5

7.3 MainActivity

Třída starající se o chod aplikace. Nejprve inicializuje všechny prvky grafického rozhraní. Po kliknutí na tlačítko se spustí metoda onClickFindDistancesToAccessPoints, která nalezne všechny AP, které podporují RTT. Pokud jsou některá taková zařízení nalezena, pak je právě tato třída zobrazí do uživatelského rozhraní. Poté už je jen na uživateli, které AP se rozhodne měřit.

7.3.1 onClickFindDistancesToAccessPoints

Po kliknutí na tlačítko se začnou scanovat dostupné AP zařízení v okolí. Ve spodní části aplikace se ukáže, kolik je dostupných AP v okolí a kolik z nich podporuje RTT. Ty, které podporují RTT, se poté ukáží v seznamu a bude možné je vybrat pro měření vzdálenosti. Metoda, která toto zajišťuje, je zobrazena na obrázku 20.

```
public void onClickFindDistancesToAccessPoints(View view) {
    if (mLocationPermissionApproved) {
        logToUi("Retrieving Access Points…");
        mWifiManager.startScan();

    } else {
        // On 23+ (M+) devices, fine location permission not granted. Request permission.
        Intent startIntent = new Intent( packageContext: this, LocationPermissionRequestActivity.class);
        startActivity(startIntent);
    }
}
```

Obrázek 20 - Ukázka kódu 6

7.3.2 find80211mcSupportedAccessPoints

Filtrovací metoda, která projde list všech AP zařízení a vybere z nich a uloží do nového listu pouze ta zařízení, která podporují 802.11mc standard. Metoda (obr. 21) je spouštěna v rámci metody onReceive, která bude vysvětlena v dalším odstavci.

```
private List<ScanResult> find80211mcSupportedAccessPoints(
        @NonNull List<ScanResult> originalList) {
    List<ScanResult> newList = new ArrayList<>();

    for (ScanResult scanResult : originalList) {

        if (scanResult.is80211mcResponder()) {
            newList.add(scanResult);
        }

        if (newList.size() >= RangingRequest.getMaxPeers()) {
            break;
        }
    }
    return newList;
}
```

Obrázek 21 - Ukázka kódu 7

7.3.3 onReceive

Pokud naše mobilní zařízení dostane nějakou odpověď, získá z wifiManagera list mScanResults. List scan result obsahuje seznam všech AP v okolí, které odpověděli na náš dotaz. V této metodě (obr. 22) se dále vyberou pouze jen ty zařízení, které podporují 802.11mc a dále vypíše do uživatelského rozhraní počet těchto AP.

```
@SuppressLint("MissingPermission")
public void onReceive(Context context, Intent intent) {

    List<ScanResult> scanResults = mWifiManager.getScanResults();

    if (scanResults != null) {

        if (mLocationPermissionApproved) {
            mAccessPointsSupporting80211mc = find80211mcSupportedAccessPoints(scanResults);

            mAdapter.swapData(mAccessPointsSupporting80211mc);

            logToUi(
                    scanResults.size()
                            + " APs discovered, "
                            + mAccessPointsSupporting80211mc.size()
                            + " RTT capable.");

        } else {
            // TODO (jewalker): Add Snackbar regarding permissions
            Log.d(TAG, msg: "Permissions not allowed.");
        }
    }
}
```

Obrázek 22 - Ukázka kódu 8

7.3.4 onScanResultItemClick

Pokud uživatel vybere nějaké AP, uživateli se otevře nové okno, kde budou detailnější informace o AP, a aplikace začne měřit veškeré údaje o zařízení. Tyto informace se začnou okamžitě zobrazovat v tomto nově otevřeném okně. Ukázku kódu najdeme na obrázku 23.

```
@Override
public void onScanResultItemClick(ScanResult scanResult) {
    Log.d(TAG,  msg: "onScanResultItemClick(): ssid: " + scanResult.SSID);

    Intent intent = new Intent( packageContext: this, AccessPointRangingResultsActivity.class);
    intent.putExtra(SCAN_RESULT_EXTRA, scanResult);
    startActivity(intent);
}
```

Obrázek 23 - Ukázka kódu 9

7.4 MyAdapter

MyAdapter se v podstatě stará o uživatelské rozhraní. Stará se listenery, které naslouchají, zda uživatel vybere nějaké AP ze seznamu, či kdy se mají začít hledat AP a další podobné funkce. Dále se stará o to, aby se data propisovala na správná místa viz obrázek 24.

```
public static class ViewHolderHeader extends RecyclerView.ViewHolder {
    public ViewHolderHeader(View view) { super(view); }
}

public class ViewHolderItem extends RecyclerView.ViewHolder implements View.OnClickListener {
    public TextView mSsidTextView;
    public TextView mBssidTextView;

    public ViewHolderItem(View view) {
        super(view);
        view.setOnClickListener(this);
        mSsidTextView = view.findViewById(R.id.ssid_text_view);
        mBssidTextView = view.findViewById(R.id.bssid_text_view);
    }

    @Override
    public void onClick(View view) {
        sScanResultClickListener.onScanResultItemClick(getItem(getAdapterPosition()));
    }
}
```

Obrázek 24 - Ukázka kódu 10

7.5 Celkový pohled na aplikaci

Aplikace je postavena tak, že se nejprve dotáže uživatele na povolení získávání polohy, pokud uživatel toto povolení udělí, pak je možné v aplikaci pokračovat. Na úvodní stránce aplikace se nachází jednoduché rozhraní (obr. 25), kde na horní části vidíme položky SSID a BSSID, pod které se nám poté budou vypisovat nascanované AP. V dolní části je button „SCAN WIFI“, po kliknutí na něj aplikace vyšle dotaz z telefonu na okolní AP a ptá se, která AP jsou k dispozici a která z nich podporují standard 802.11mc. Pokud se některá taková AP naleznou, pak jsou přidána jejich SSID a BSSID do seznamu. Nad tlačítkem se ukáže počet, kolik takových zařízení bylo nalezeno. Obrázek 24 znázorňuje, jak vypadá aplikace po načtení AP a vybrání právě chtěných AP, takových, co podporují RTT.

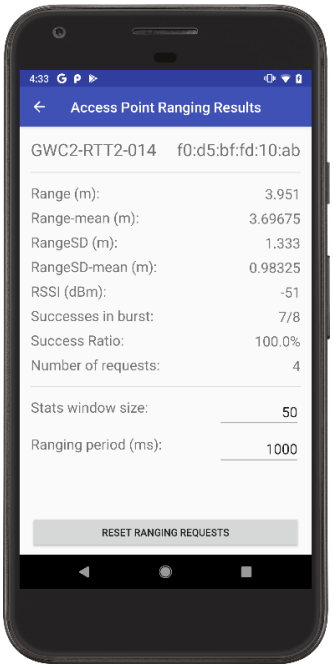
Obrázek 25 - Screenshot aplikace 1

Po kliknutí na některý řádek ze seznamu zařízení podporujících RTT se aplikace posune do dalšího okna, kde vidíme více informací o zařízení a vzápětí začne aplikace naslouchat tomuto zařízení a měřit veškeré údaje co RTT dokáže získat. Údaje, které aplikace vypisuje jsou:

- Range (m),
- Range-mean (m),

- RangeSD (m),
- RangeSD-mean (m),
- RSSI (dBm),
- Success in burst,
- Success ratio,
- Number of requests.

Pod těmito údaji se nacházejí nastavitelné pole. Nastavit můžeme nastavit například Ranging period který určuje po jaké době budou vysílány jednotlivé rangingRequesty. Na obrázku 26 je vidět, jak vypadá měření vybraného AP. Veškeré údaje jsou aktualizovány s každým dalším vyslaným požadavkem na měření.



Obrázek 26 - Screenshot aplikace 2

8 Úprava aplikace

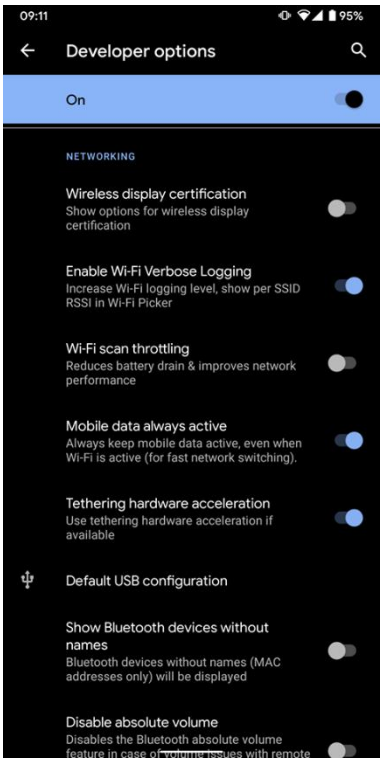
Po prozkoumání sample aplikace jsem se musel rozhodnout, jak tuto aplikaci začít upravovat, tak abych ji byl schopen rozšířit o další kroky.

8.1 Spuštění načítání AP, hned po spuštění aplikace

První změnou upravené aplikace bylo zakomponovat scanování okolních AP hned po startu aplikace. Prvním krokem bylo rozvázat funkci scanování s tlačítkem „SCAN WIFI“. Metoda, která pouští scanování je onClickFindDistancesToAccessPoints(mRecyclerView). Tato metoda se spouštěla po klepnutí na tlačítko. Tím, že byla tato metoda přesunuta do metody onResume(), bylo zajištěno, že se AP scanují pokaždé, kdy je aplikace spuštěna na popředí. To znamená, že pokud je aplikace shozena na pozadí, tak se aplikace přestane dotazovat na AP v okolí. V tento moment je aplikace totiž přepnuta do módu onPause(). Pokud aplikaci opět zpustíme, vrátí se aplikace do onResume() a spustí se znovu hledání zařízení v okolí.

8.1.1 Problém počtu dotazů na AP v okolí

Upravování aplikace postupně vedlo k problému, že aplikace nedokáže nalézt všechna AP v okolí ihned, ale že je zde zhruba 2minutová prodleva mezi jednotlivými požadavky na hledání. To je pro tuto aplikaci nepřípustné, takže se muselo přijít s nějakým alternativním řešením tohoto problému. Na stránkách Android developer se můžeme dočíst, že Android 9 má právě toto omezení, kdy zařízení může vyslat tento dotaz jednou za 2 minuty. Ovšem s aktualizací na Android 10 přišla možnost časové zpoždění trošku snížit, a to tak, že v developer options nalezneme pod networking položku WiFi scan throttling jak je zobrazeno na obrázku 27. Tato funkce je zde umístěna z důvodu spoření baterie a schválně omezuje dotazování tímto způsobem. Po vypnutí této funkce nám je umožněno scanovat zařízení 4krát za 2 minuty, což je jednou za 30 s, pokud chceme nějaké konstantní scanování. To už by se dalo považovat za použitelné, i když je to veliké omezení. Může totiž nastat několik případů, kdy by uživatel přešel z jedné části budovy do jiné a jeho mobilní zařízení by nedokázalo včasně zareagovat na příchod nového AP.



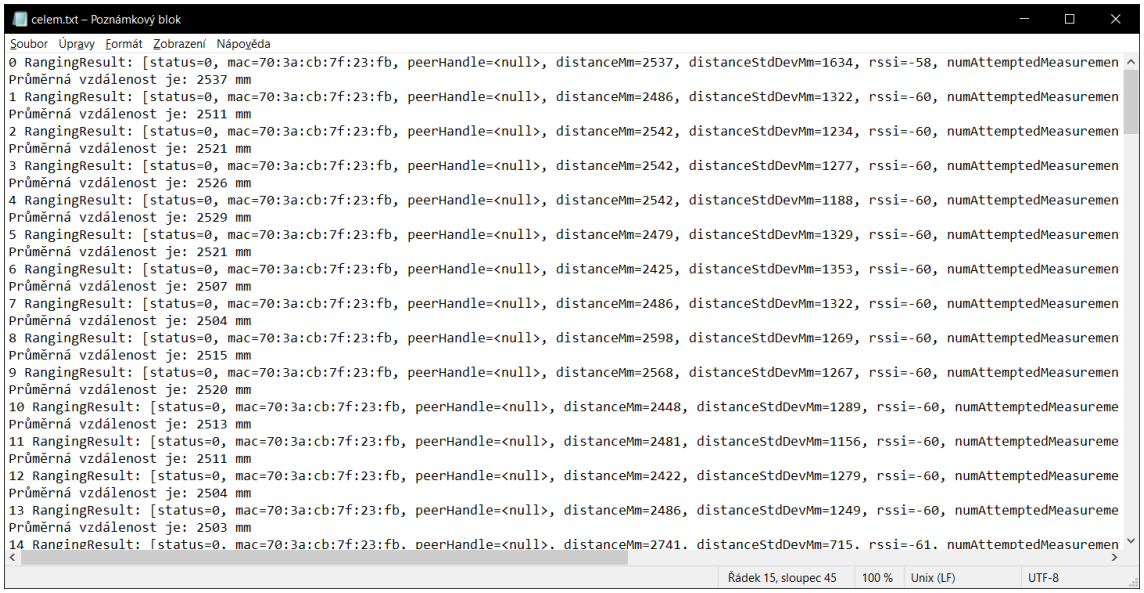
Obrázek 27 - Vypnutí WiFi scan throttling

8.2 Spuštění měření na všechny RTT AP v okolí

Dalším rozdílem v představě a ve skutečné aplikaci bylo to, že aplikace byla schopna měřit pouze jedno AP v daném čase a pro naši verzi aplikace potřebujeme měřit minimálně 3 zařízení současně. Aplikace sice načte všechna použitelná AP v okolí, ale uživatel si může vybrat pouze jedno, které se začne měřit. Z toho konceptu bylo upuštěno. V aplikaci byla zrušena možnost výběru jednoho AP.

8.3 Vypisování do .txt souboru

Aby byla možnost nějakým způsobem kontrolovat data z měření, bylo nutné vytvořit nějaký zápis do externího souboru. Pro tuto potřebu nám postačil jednoduchý filewriter. Do metody zadáme jako parametr list RangingResult a ty pak postupně vypíšeme. S každou naměřenou hodnotou se přidá řádek do textového souboru, kde jsou vidět konkrétní naměřené hodnoty vůči všem AP v okolí. Aby nám ovšem filewriter fungoval, musí uživatel aplikaci povolit přístup k souborům. Ukázka jak takový výpis do .txt souboru vypadá je ilustrován na obrázku 28.



Obrázek 28 – Zápis do .txt měření jednoho AP

8.4 Vytvoření vlastního scanResultu

ScanResult je pro aplikaci hlavním nosičem informace, v základu aplikace slouží scanResult k tomu, aby našel veškerá AP v okolí a vybral z nich poté ty, které podporují 802.11mcc. Avšak jelikož je umožněno scanování AP v okolí tímto způsobem pouze 4krát za 2minuty, nastává nám problémová situace, kdy aplikace nedokáže zareagovat na nově přidaná či odebraná AP. Proto se zde naskytla myšlenka, že když se nebudou zařízení scanovat, ale již budeme dopředu vědět, s jakými AP se uživatel může setkat, tak můžeme vytvořit námi vytvořené falešné scanResulty, které budeme schopni posílat stále, nehledě na to, zda jsou tyto AP v okolí a zda jsou již nascanované, či nikoliv. Ovšem toto řešení naráží na další problém, a to ten, že si sami v aplikaci nemůžeme jen tak vytvořit novou instanci scanResultu.

Abychom obešli tento problém, je nutné použít k vytvoření „falešného scanResultu“ tzv. reflexi. Za pomoci reflexe jsme schopni sami vytvořit. Implementace vytvoření takového scanResultu je znázorněna na obrázku 29.

```
    c3 = ScanResult.class.getDeclaredConstructor();
    c3.setAccessible(true);
    ScanResult r3 = (ScanResult) c1.newInstance();
    r3.SSID = "RTT test";
    r3.BSSID = "70:3a:cb:7f:dc:b4";
    r3.capabilities = "[WPA2-PSK-CCMP][RSN-PSK-CCMP][ESS]";
    r3.level = -50;
    r3.frequency = 5180;
    r3.centerFreq0 = 5210;
    r3.centerFreq1 = 0;
    r3.channelWidth = 2;
    @SuppressLint("SoonBlockedPrivateApi") Method method3 = r3.getClass().getDeclaredMethod( name: "setFlag", Long.TYPE);
    method3.setAccessible(true);
    method3.invoke(r3, ...objects: 2); // FLAG_80211mc_RESPONDER

    scanResults.add(r1);
    scanResults.add(r2);
    scanResults.add(r3);

} catch (NoSuchMethodException | IllegalAccessException | InstantiationException | InvocationTargetException e) {
    e.printStackTrace();
}
```

Obrázek 29 - Vytvoření instance scanResult

8.5 Přidání trilaterace

Asi nejzajímavější část aplikace by měla být trilaterace. Jak již bylo zmíněno v práci, trilaterace značí měření polohy zařízení vůči třem známým kotvám. Jako kotvy v tomto případě používáme 3 AP, které jsou umístěny v pevných bodech místnosti. Každé AP nám odpoví, v jaké vzdálenosti se od něj mobilní zařízení nachází, tím nám vzniknou „kružnice“ kolem každého AP. V průsečíku nebo nejblíže k průsečíku všech kružnic nám vznikne bod, ve kterém se mobilní zařízení nachází. Jelikož známe souřadnice [x, y] každého AP a známe poloměr (vzdálenost od AP k zařízení). Tak můžeme sestavit rovnice těchto kružnic. Když k těmto rovnicím přidáme element našeho zařízení, vznikají nám rovnice, které vypadají následovně. Hodnota (x, y) jsou pro nás hledané souřadnice mobilního zařízení a hodnoty (x_1,y_1), (x_2,y_2), (x_3,y_3) jsou souřadnice umístění jednotlivých AP.

$$(x - x_1)^2 + (y - y_1)^2 = r_1^2$$
$$(x - x_2)^2 + (y - y_2)^2 = r_2^2$$
$$(x - x_3)^2 + (y - y_3)^2 = r_3^2$$

Nyní můžeme rozšířit rovnice tak, že nám vznikne:

$$x^2 - 2x_1x + x_1^2 + y^2 - 2y_1y + y_1^2 = r_1^2$$
$$x^2 - 2x_2x + x_2^2 + y^2 - 2y_2y + y_2^2 = r_2^2$$
$$x^2 - 2x_3x + x_3^2 + y^2 - 2y_3y + y_š^2 = r_3^2$$

A odečteme druhou rovnici od první:

$$(-2x_1 + 2x_2)x + (-2y_1 + 2y_2)y = r_1^2 - r_2^2 - x_1^2 + x_2^2 - y_1^2 + y_2^2$$

Stejně odečteme třetí od druhé:

$$(-2x_2 + 2x_3)x + (-2y_2 + 2y_3)y = r_2^2 - r_3^2 - x_2^2 + x_3^2 - y_2^2 + y_3^2$$

V tomto kroku si pro zjednodušení přepíšeme tyto dvě rovnice pomocí hodnot A, B, C, D, E, F. To bude mít za následek, že předchozí dvě rovnice vypadají následovně:

$$A_x + B_y = C$$
$$D_x + E_y = F$$

Z tohoto již dokážeme dopočítat námi hledané souřadnice (x, y) tímto způsobem:

$$x = \frac{CE - FB}{EA - BD}$$
$$y = \frac{CD - AF}{BD - AE}$$

Tímto výpočtem získáme souřadnice (x, y) čili dokážeme určit polohu mobilního zařízení v místnosti.

8.5.1 Trilaterace v kódu

Nyní se podíváme, jak předchozí výpočet zakomponovat do naší aplikace. Pro tento účel byla vytvořena nová třída nazvána „Trilateration“. Tato třída bude mít na pevno nastaveny mac adresy třech známých AP, které budeme používat pro měření polohy mobilního zařízení. Pomocí rangingResultu, vloženého jako parametr metody getAP (obr. 30), vybereme pomocí MAC adresy jedno AP a vrátíme jeho pevně stanovenou pozici [x, y].

```
double[] getAP(RangingResult result) {
    double[] ap = new double[2];

    //Pozice AP v prostředí 1
    switch (Objects.requireNonNull(result.getMacAddress()).toString()) {
        case "70:3a:cb:7f:dc:1a":
            ap = new double[]{3, 0};
            break;
        case "70:3a:cb:7f:23:fb":
            ap = new double[]{3, 4};
            break;
        case "70:3a:cb:7f:dc:b4":
            ap = new double[]{0, 0};
            break;
    }
```

Obrázek 30 - Ukázka kódu 11

Metoda getPositionWithTrilateration je vlastně metodou, která vypočítá polohu mobilního zařízení. Jako parametr metoda dostane list rangingResultů. Prvním krokem tohoto výpočtu je určit vzdálenost jednotlivých AP od mobilního zařízení. Metoda nejdříve porovná MAC adresu z rangingResultu oproti pevně nastaveným MAC adresám v metodě getAP (obr. 31). Switch vrátí této metodě pozici měřeného AP. A přejde se k výpočtu trilaterace. Po získání těchto údajů metoda přikročí k dalšímu kroku. Tím krokem je samotné vypočítání výsledných souřadnic [x, y], tedy hledané pozice mobilního zařízení. Výpočet souřadnic v kódu je vyobrazen na obrázku 32.

```
double[] getPositionWithTrilateration(List<RangingResult> list) {
    double[] ap1 = getAP(list.get(0));
    double r1;
    if (list.get(0).getDistanceMm() > 0) {
        r1 = list.get(0).getDistanceMm() / 1000;
    } else {
        r1 = 0;
    }
    double[] ap2 = getAP(list.get(1));
    double r2;
    if (list.get(1).getDistanceMm() > 0) {
        r2 = list.get(1).getDistanceMm() / 1000;
    } else {
        r2 = 0;
    }
    double[] ap3 = getAP(list.get(2));
    double r3;
    if (list.get(2).getDistanceMm() > 0) {
        r3 = list.get(2).getDistanceMm() / 1000;
    } else {
        r3 = 0;
    }
```

Obrázek 31 - Ukázka kódu 12

```
double A = -2 * ap1[0] + 2 * ap2[0];
double B = -2 * ap1[1] + 2 * ap2[1];
double C = Math.pow(r1, 2) - Math.pow(r2, 2) - Math.pow(ap1[0], 2)
        + Math.pow(ap2[0], 2) - Math.pow(ap1[1], 2) + Math.pow(ap2[1], 2);
double D = -2 * ap2[0] + 2 * ap3[0];
double E = -2 * ap2[1] + 2 * ap3[1];
double F = Math.pow(r2, 2) - Math.pow(r3, 2) - Math.pow(ap2[0], 2)
        + Math.pow(ap3[0], 2) - Math.pow(ap2[1], 2) + Math.pow(ap3[1], 2);
xy[0] = (C * E - F * B) / (E * A - B * D);
xy[1] = (C * D - A * F) / (B * D - A * E);


return xy;
```

Obrázek 32 - Ukázka kódu 13

V rámci testování aplikace byla metoda spuštěna, pokud se v rangingResultu nacházely všechny 3 pevně nastavené AP a jejich rangingResulty. Pokud by alespoň jedno AP nedokázalo z nějakého důvodu poslat svůj rangingResult, tak by se metoda neprovedla. Při měření trilaterace využijeme znovu fileWriteru a zrovna můžeme zapisovat naměřené pozice do .txt souboru. Dále je v kódu zařízeno, aby se naměřená pozice ukazovala průběžně uživateli na display.

9 Testování aplikace

V této části práce se podíváme na to, jak byla upravovaná aplikace testována, tedy konkrétně v jakých podmínkách testování proběhlo a co vše mohlo nějakým způsobem ovlivnit výsledky. Nejprve se podíváme na vytvoření testovacího prostředí. Jedná se o prostředí, kde se testování odehrávalo, a jaké podmínky byly nastoleny.

9.1 Vytvoření testovacího prostředí

Testovací prostředí v prvním kroku byl pokoj o rozměrech 4,6m x 4m. Tato místnost byla rozdělena na čtverce. Tímto krokem se vytvořila souřadnicová síť, ze které bylo možné získat pozici pomocí souřadnic X, Y viz obrázek 33.

0,4	1,4	2,4	3,4	4,4
0,3	1,3	2,3	3,3	4,3
0,2	1,2	2,2	3,2	4,2
0,1	1,1	2,1	3,1	4,1
0,0	1,0	2,0	3,0	4,0

Obrázek 33 - Rozložení místnosti

9.2 Průběh testování

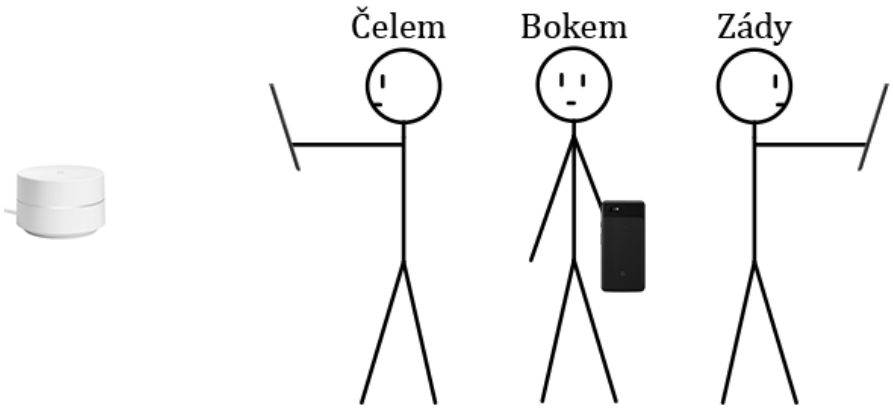
9.2.1 Testování měření vzdálenosti (testování FTM)

V této kapitole se podíváme na průběh měření vzdálenosti telefonu od AP v různých podmínkách. Pokusíme se nasimulovat různé podmínky prostředí a také pozice mobilního zařízení vůči AP a budeme zkoumat, jaký mají tyto podmínky vliv

na výsledné hodnoty. Cílem je zjistit, jak moc dokáže vliv prostředí ovlivnit přesnost měření.

Pro testování měření vzdálenosti je potřeba mít dvě věci. První věcí je telefon s OS Android 9 a vyšší. Pro tyto účely byl Univerzitou zapůjčen Google Pixel 2 a dále jelikož máme dostupný Google Pixel 3a XL, tak budeme měřit hodnoty na těchto dvou zařízení. Tím dalším prvkem je AP takové, které podporuje standard IEEE 802.11mc. Opět pro tyto účely Univerzita zapůjčila místní AP – Google WiFi.

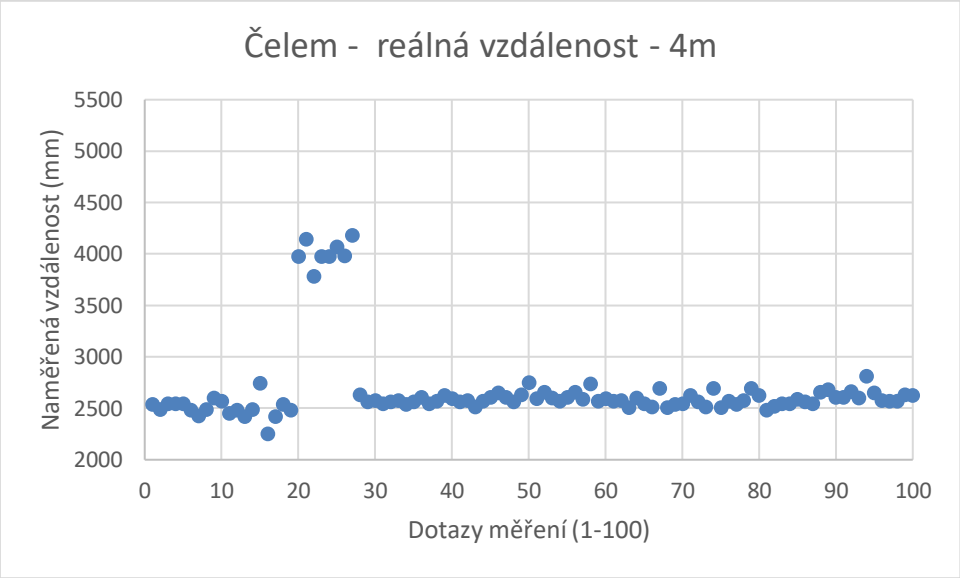
Pro testování možností RTT s FTM bylo vytvořeno testovací prostředí, kde byla naměřena vzdálenost 4 m od pozice AP. V této vzdálenosti byl umístěn subjekt s mobilním zařízením a vyslal směrem k AP 100 dotazů ve kterých byla zjišťována vzdálenost od AP. Tyto výsledky se ukládaly do .txt souboru v paměti telefonu, tak aby se později mohly zpracovat do grafů a mohly být porovnány mezi sebou. Po každém dotazu se vypočítala dosavadní průměrná hodnota, takže ve výpisu bylo vidět, jak se průměrná vzdálenost postupně ustaluje a přibližuje k nějaké hodnotě. Na konci souboru je viditelná průměrná hodnota ze všech 100 měření. Měření je rozděleno do dvou fází. První fáze je měření bez překážky v cestě mezi polohou telefonu a AP, poté proběhne měření stejného typu, ovšem v cestě bude zeď. Zde budeme zkoumat, jestli bude mít tato překážka nějaký větší vliv na měřená data, či nikoliv. Na obrázku 34 je ilustrováno, jakým způsobem bude subjekt s mobilním zařízení postaven vůči AP. Máme 3 základní polohy. Čelem, bokem a zády.



Obrázek 34 - Způsob postavení subjektu

Měření čelem k AP – čistý výhled

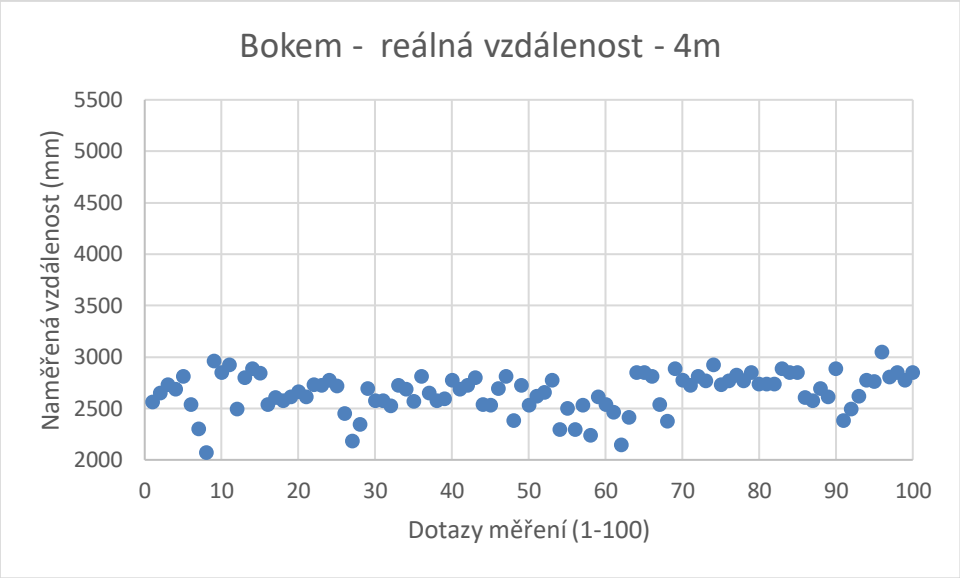
Při tomto měření byl subjekt postaven tak, aby s telefonem směřoval přímo naproti AP. V cestě nebyla přítomna žádná překážka, jen vzduch. V grafu je patrné, že už od začátku měření se držíme kolem hodnoty 2,5m. Poté přišel výkyv hodnot, které byly shodou okolností nejpřesnější, avšak podle mého názoru by se mohlo jednat o pakety odražené od zdi, které se náhodou dostaly nejblíže skutečnosti. Výsledná průměrná hodnota po 100 dotazech byla 2686mm tedy zhruba 2,7m, což je hodnota o 1,3 m kratší, než byla skutečnost. Výsledky si můžeme prohlédnout na obrázku 35.



Obrázek 35 - Graf měření čelem

Měření bokem k AP – čistý výhled

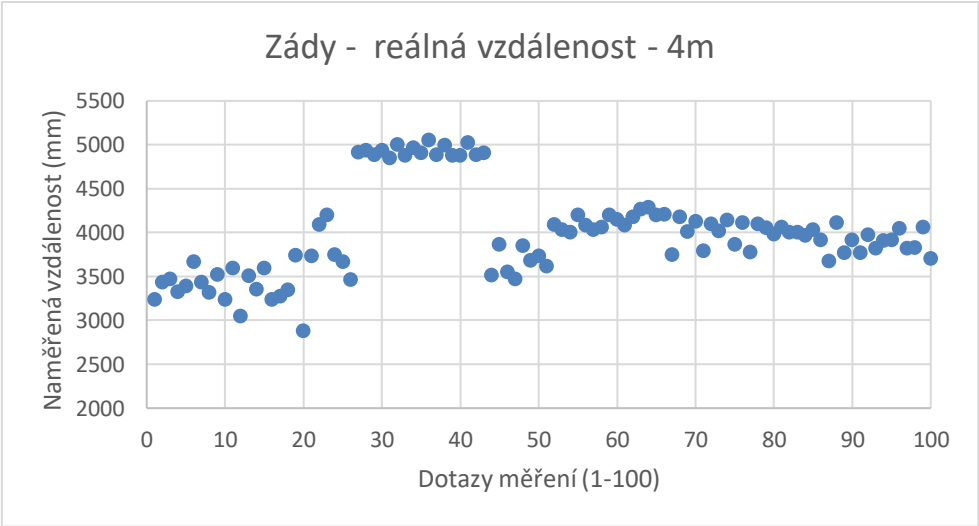
Toto měření dopadlo znatelně lépe, jak je vidno z grafu níže. V tomto grafu vidíme, že se hodnoty pohybovaly v celkem soustředěném rozpětí kolem hodnoty 2,5 m. Po spočítání průměru je výsledná hodnota dosti podobná předchozímu měření. Průměrná hodnota měření je 2659 mm, tudíž znovu jako v předchozím případě zhruba 2,7m, viz obr.36.



Obrázek 36 - Graf měření bokem

Měření zády k AP – čistý výhled

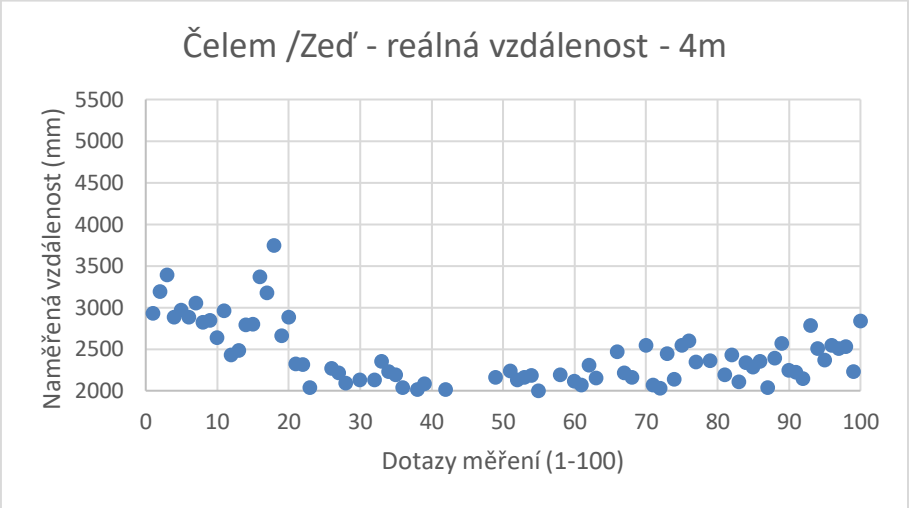
Největším překvapením bylo právě měření zády. Navzdory tomu, že zde opět došlo k odchýlení hodnot, a to zhruba k 5 m, tak se zbylé hodnoty pohybují kolem 4 m. Šokující je i výsledek tohoto měření, jelikož vyšlo jako nejpřesnější. Výsledná průměrná hodnota vzdálenosti vyšla neskutečných 4002 mm tedy 4 m. Výsledek tohoto měření (obr. 37) byl tak neočekávaný, že pro ověření se test opakoval několikrát vždy s podobným výsledkem.



Obrázek 37 - Graf měření zády

Měření čelem k AP – přes zeď

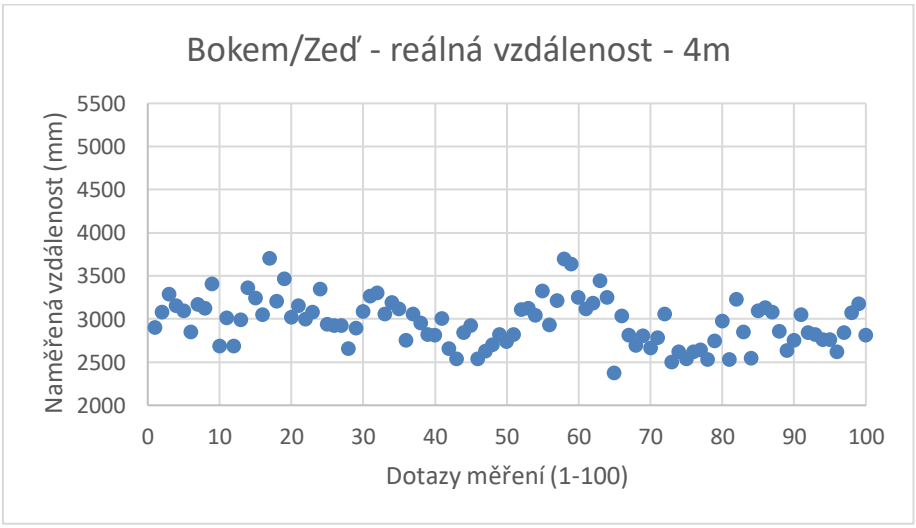
Hodnoty v grafu na obrázku 35 se pohybují někde mezi 2-4 m. Nejspíš jsou tyto velké skoky způsobeny právě odrazy od zdi, před kterou subjekt stojí. Výsledná průměrná hodnota po 100 měřeních vyšla 2318 mm, což je výsledek (obr. 38) ne o moc horší než výsledek při měření čelem bez překážky.



Obrázek 38 - Graf měření čelem přes zeď

Měření bokem k AP – přes zeď

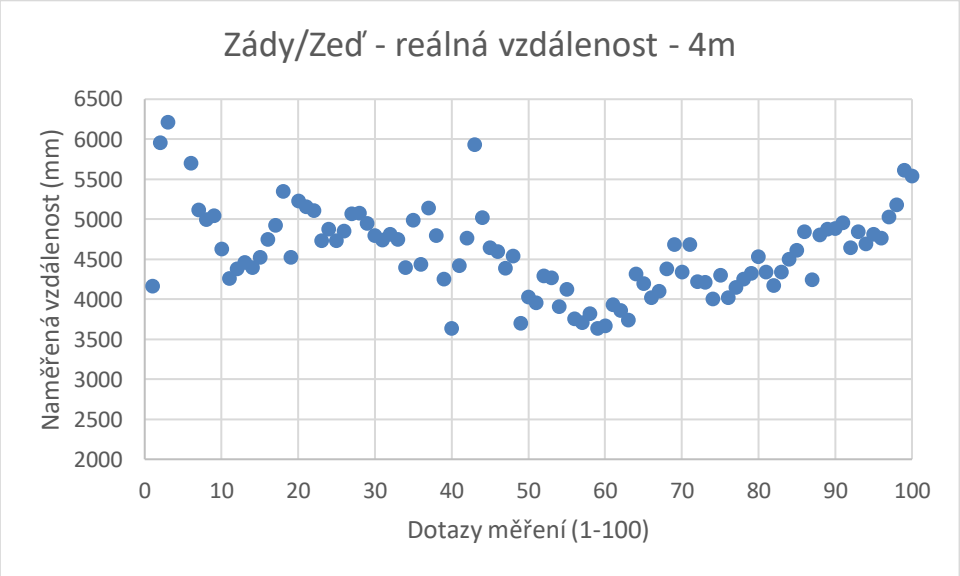
Z grafu (obr. 39) vidíme, že hodnoty se pohybují mezi 2,4 – 3,8 m, ovšem znovu se zde setkáváme s relativně velkým rozpětím.



Obrázek 39 - Graf měření bokem přes zeď

Měření zády k AP – přes zeď

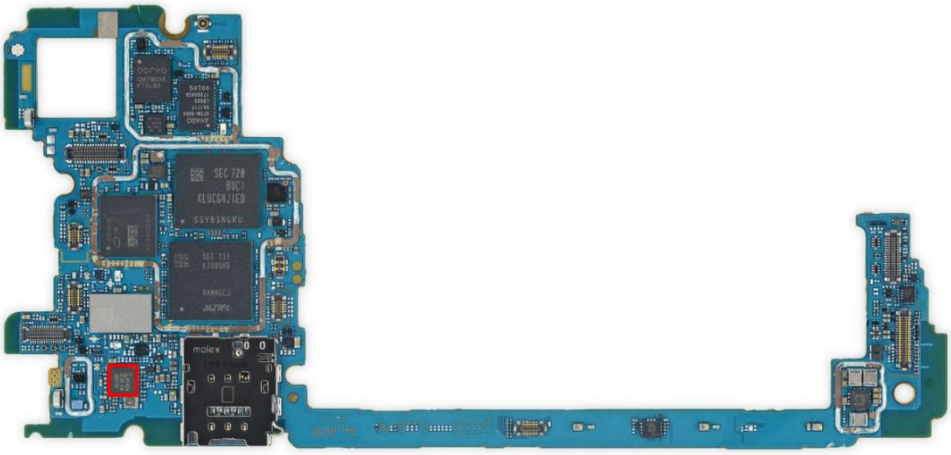
Toto měření dopadlo jako jediné s výsledkem vyšším, než byla reálná hodnota (obr. 40). Všechna ostatní měření prokázala chybu, která spíš směřovala k tomu, že byla vzdálenost nižší než skutečná.



Obrázek 40 - Graf měření zády přes zeď

Zhodnocení výsledků

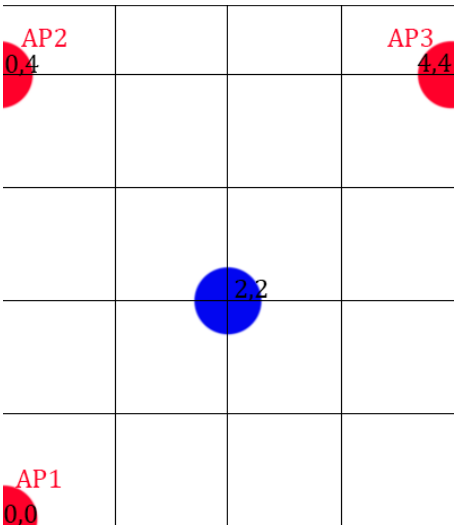
Jedním z ovlivňujících faktorů těchto měření bude rozhodně umístění antén v telefonu. Na obrázku 41 je vidět, kde jsou antény umístěny v telefonu, na kterém probíhaly testy. WiFi adaptér je označen červeně. Z výsledků je patrné, že postavení zdi do cesty mezi mobilním zařízením a AP vznikne větší rozptyl hodnot. Tento rozptyl nemá tendenci se s přibývajícím počtem měření ustalovat, tudíž hodnoty se zdají méně přesné než hodnoty, kde se zeď nenacházela v cestě. I když výsledky s překážkou dopadly o něco hůř, tak je stále můžeme ohodnotit pozitivně. Jelikož sám Google deklaroval, že přesnost této technologie je zhruba do dvou metrů, tak se všechny naše výsledky nacházejí právě v této chybové toleranci. Takže tuto technologii můžeme prohlásit za relativně funkční, i když je zde dále velký prostor pro zdokonalení.



Obrázek 41 - Umístění antén v telefonu

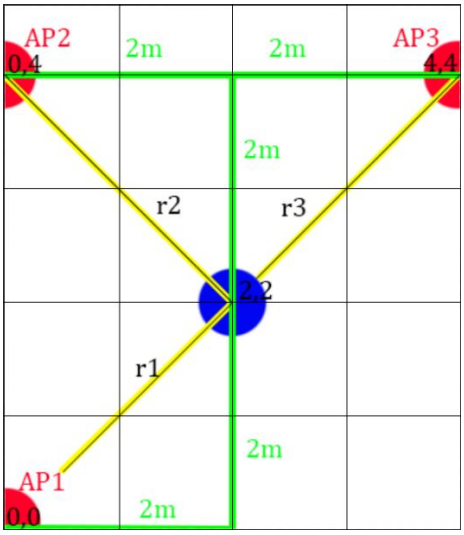
9.2.2 Testování trilaterace 1

V první části testování otestujeme algoritmus pomocí umělých dat, která budou představovat dokonalé prostředí s dokonalými daty. Jako testovací příklad byl vybrán příklad takový, kdy bude pozice telefonu na souřadnicích (2, 2), předem je jasné, že po proběhnutí metody pro vypočtení trilaterace by se měl algoritmus dostat právě k tomuto výsledku. Abychom byli schopni se dostat k takovémuto výsledku, musí algoritmus nejdříve znát, jaké hodnoty r1, r2 a r3 k tomuto výsledku povedou. Nejprve si určíme, jaký bude souřadnicový systém. Souřadnicový systém bude použit stejný, jako je znázorněn na obrázku 33. Dále umístíme AP a mobilní zařízení viz obr. 42.



Obrázek 42 - Rozmístění AP a mobilního zařízení

Nyní je potřeba pomocí Pythagorovy věty určit vzdálenosti od jednotlivých AP k telefonu (r1, r2, r3). Jak je vidět z obrázku 43, tak jelikož předem známe pozici subjektu, a víme, jak vzdáleny jsou od sebe AP, vznikají nám trojúhelníky, kde víme že všechny odvěsny jsou dlouhé 2m a středem našeho zájmu je čemu se rovnají přepony. V tomto příkladě je z obrázku 43 patrné, že všechny přepony (r1, r2, r3) budou stejně dlouhé. Ale to nám v tomto testovacím případě vůbec nevadí. Nám jde pouze o ověření, zda naše aplikace bude správně počítat.



Obrázek 43 - Počítání odvěsen

Pomocí vzorce Pythagorovy věty dopočítáme všechny potřebné vzdálenosti (r1, r2, r3).

$$a^2 + b^2 = c^2$$

$$c = \sqrt{a^2 + b^2}$$

$$r = \sqrt{2^2 + 2^2} = 2.82m$$

To pro nás znamená, že r1, r2, r3 jsou 2.82m, tyto hodnoty, když vložíme jako parametr do potřebných metod (obr. 44) , tak nám trilaterace spočítá výsledek.

```
//Testování trilaterace
Trilateration test = new Trilateration(2.82, 2.82, 2.82);
System.out.println("(" + test.getPosition()[0]+ " , " + test.getPosition()[1]);
```

Obrázek 44 - Parametry trilaterace

Tento kousek kódu nám do konzole vypíše výsledek (2, 2) což je námi očekávaný výsledek, a tudíž můžeme zhodnotit náš kód jako funkční.

9.2.3 Testování trilaterace v reálném prostředí

Bohužel v průběhu psaní této práce byla v ČR pandemie COVID-19 a vláda omezila pohyb osob jak ve škole, tak kdekoliv v ulicích. Z tohoto důvodu bylo nutno testování omezit pouze na dva byty. Jeden ve velkém panelovém domě a druhý v malém bytovém domě. Tyto stísněné prostory byly trochu omezující, ale i přesto se dala aplikace otestovat. Jelikož Google deklaroval přesnost zhruba na dva metry, tak by plocha jednoho bytu měla být dostačujícím prostorem. Dva byty byly zvoleny také z důvodu toho, že jeden z nich (dále testovací prostředí 1) se nacházel na takovém místě, kde bylo mnoho bytů pohromadě, a tudíž se zde nachází spoustu rušivých elementů, například v podobě ostatních AP zařízení. Na obrázku 45 je vidno kolik AP se nachází poblíž testovacího prostředí 1. Proto bylo zřízeno testovací prostředí 2, kde se v okolí nachází pouze pár bytových jednotek, a tak je zde menší element rušení. Tyto dvě testovací prostředí budou poté po jednotlivých měření mezi sebou porovnány a bude vyhodnoceno, jaký měl rušivý element vliv na měření naší trilaterace.

SSID	MAC Address	PHY Type	RSSI	Signal Quality	Average Signal...	Frequency	Channel	Information Size
WifiRTT1	70-3A-CB-7F-23-FB	802.11n/ac	-61	65	60.3	5.180	36	200
WifiRTT1	70-3A-CB-7F-23-FF	802.11g/n	-46	90	90.0	2.462	11	209
vendy	04-8D-38-CC-57-41	802.11g/n	-64	60	58.6	2.437	6	254
TP-LINK_E50E	18-A6-F7-7A-E5-0E	802.11g/n	-63	61	54.7	2.422	3	195
TP-LINK_Doma	90-F6-52-2F-52-EC	802.11g	-83	28	24.3	2.412	1	133
RTT test	70-3A-CB-7F-DC-1A	802.11n/ac	-39	99	98.8	5.180	36	200
RTT test	70-3A-CB-7F-DC-B4	802.11n/ac	-51	81	80.2	5.180	36	200
RTT test	70-3A-CB-7F-DC-B8	802.11g/n	-41	98	95.1	2.437	6	209
RTT test	70-3A-CB-7F-DC-1E	802.11g/n	-39	99	98.0	2.412	1	209
rawNET	04-8D-38-C9-63-26	802.11g/n	-82	30	29.2	2.437	6	255
MujO2Internet_3E6BD8	74-B5-7E-3E-6B-D8	802.11g/n	-72	46	44.3	2.462	11	442
Martin router king	18-31-BF-A2-EB-90	802.11g/n	-30	99	99.0	2.472	13	404
Helena	68-FF-7B-7D-0C-38	802.11g/n	-79	35	32.3	2.422	3	205
Hans	D8-5D-4C-DB-12-B8	802.11g	-87	21	22.0	2.462	11	61
DIRECT-0f-HP M377 LaserJet	0E-96-E6-98-63-0F	802.11g/n	-89	18	18.0	2.437	6	280
ASUS	F8-32-E4-45-A1-38	802.11g/n	-82	30	33.0	2.472	13	341
	72-3A-CB-7F-DC-19	802.11n/ac	-82	30	74.1	5.180	36	135
	72-3A-CB-7F-23-F9	802.11n/ac	-63	61	59.6	5.180	36	135
	72-3A-CB-7F-DC-B5	802.11n/ac	-51	81	80.4	5.180	36	135

Obrázek 45 - Počet AP testovací prostředí 1

V obou případech měření získáme z aplikace 100 dotazů na polohu mobilního zařízení. Poloha zařízení bude reprezentována jako body x a y na souřadnicové síti, kterou vytvoříme v obou testovacích prostředích. Do této sítě umístíme 3 AP, kde každé z těchto AP bude mít jasně danou svou pozici na souřadnicové síti. Poté umístíme na nějaký bod v síti mobilní zařízení, spustíme měření a výpočet jeho pozice a budou se průběžně ukládat data do souboru umístěném v paměti telefonu. Z tohoto souboru poté vyjmeme data a pomocí excelu vytvoříme grafy zobrazující hodnoty měření. Dále bude vytvořen průměr z naměřených hodnot a tato hodnota pro nás bude figurovat jako výsledek měření. Poté porovnáme výslednou průměrnou hodnotu s hodnotou reálné pozice zařízení a určíme zhruba jaké chyby se aplikace dopustila vůči skutečnosti.

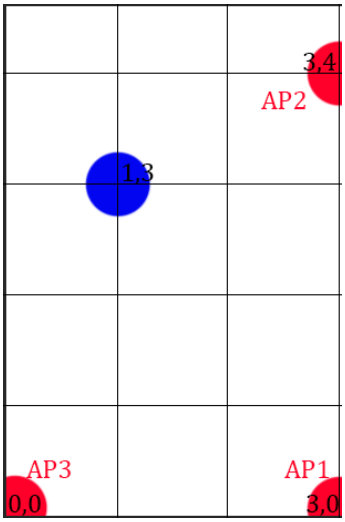
Testovací prostředí 1

Toto testovací prostředí je rozlohy 4x3m. Připravená souřadnicová síť je zobrazena na obrázcích 46, 47 a 49. Na těchto obrázcích je také vidět, jak byli rozmístěny jednotlivé AP a umístění telefonu pro 3 různá měření.

0,4	1,4	2,4	3,4
0,3	1,3	2,3	3,3
0,2	1,2	2,2	3,2
0,1	1,1	2,1	3,1
0,0	1,0	2,0	3,0

Obrázek 46 - Souřadnicová síť 1

První měření bude probíhat v sestavě, která je zobrazena na obrázku 47. Tento obrázek ukazuje, že mobilní zařízení bylo umístěno do souřadnicové sítě na pozici [1.0 , 3.0]. Z toho místa byla spuštěna aplikace a její měření trilaterace, přičemž každá vrácená hodnota se zapsala do souboru v paměti telefonu.



Obrázek 47 - Rozmístění AP a mobilního zařízení 1

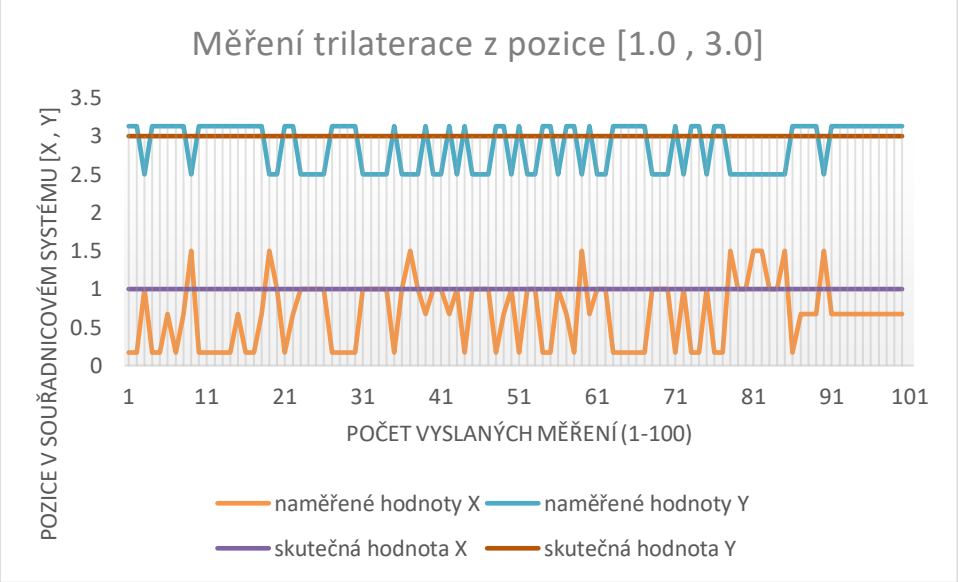
Soubor v telefonu byl poté zpracován v excelu, kdy byla rozparsována data, jak je ilustrováno v tabulce 1. Jak je vidět v tabulce, data v prvních 10 měřeních jsou konzistentní, avšak v průběhu všech 100 měření se data trošku liší. Tato data byla poté zpracována do bodového grafu viz obrázek 44.

Označení vyslaného požadavku	hodnota X	hodnota Y	skutečné X	skutečné y
1 Pozice mobilního zařízení:	1.5	2.125	1	3
2 Pozice mobilního zařízení:	1.5	2.125	1	3
3 Pozice mobilního zařízení:	1.5	2.125	1	3
4 Pozice mobilního zařízení:	1.5	2.125	1	3
5 Pozice mobilního zařízení:	1.5	2.125	1	3
6 Pozice mobilního zařízení:	1.5	2.125	1	3
7 Pozice mobilního zařízení:	1.5	2.125	1	3
8 Pozice mobilního zařízení:	1.5	2.125	1	3
9 Pozice mobilního zařízení:	1.5	2.125	1	3
10 Pozice mobilního zařízení:	1.5	2.125	1	3

Tabulka 1 - Ukázka dat pro zpracování v excelu

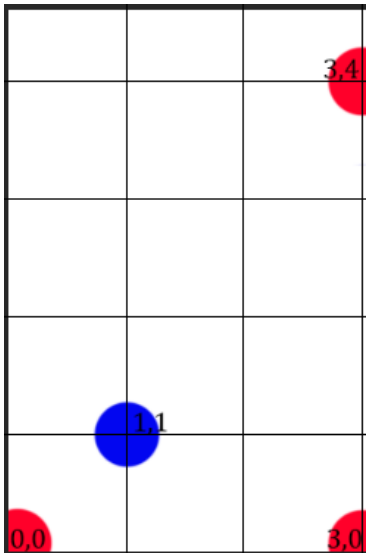
Na tomto grafu (obr. 48) vidíme na ose x, kolik bylo provedeno měření a na ose y jakou hodnotu pozice nám aplikace vrátila. Přímo v grafu vidíme 4 různé sledované údaje. Dva konstantní údaje jsou pro nás reálné pozice, kde bylo zařízení umístěno, jak vidíme, tak tato pozice se v průběhu měření nemění. Další dva údaje jsou již námi hledané hodnoty. Jedná se o hodnoty pozic, které nám byly

vyhodnoceny aplikací. Vidíme, že pro každý bod osy Y máme vždy 4 hodnoty na ose x. Tyto 4 hodnoty nám zobrazují, jaká byla v daném čase skutečná pozice [x, y] a za pomoci aplikace získaná pozice [x, y]. Po vyhodnocení grafu je patrné, že se k reálné pozici dostaneme zřídka, avšak chyba, které se aplikace dopouští, je maximálně do 1 metru. Jelikož Google sám avizoval, že přesnost této technologie je zhruba na 2 m, tak výsledky dopadly uspokojivě.

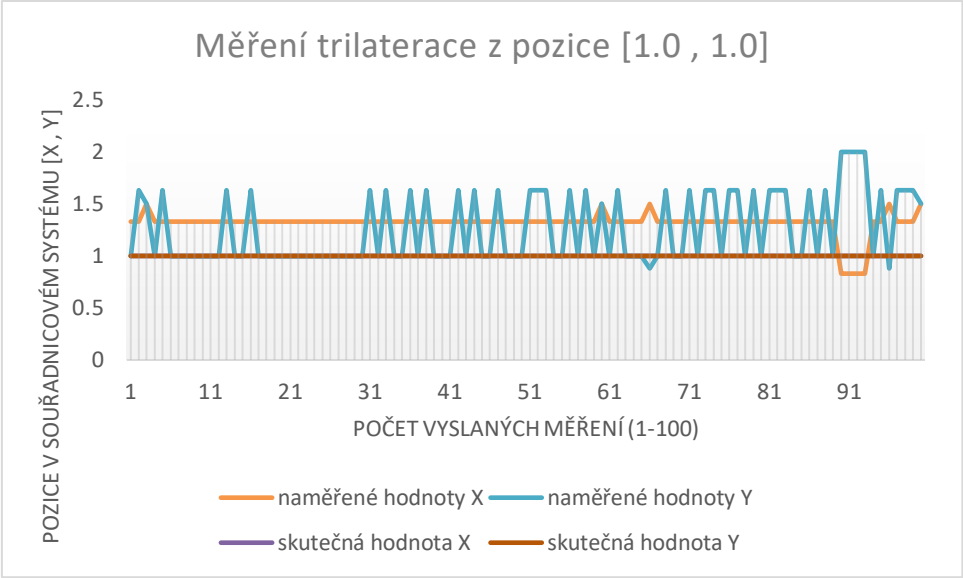


Obrázek 48 - Testovací prostředí 1 - měření 1

Druhé měření dopadlo dosti podobně jako měření předchozí (obr. 50). Místo, ze kterého bylo spuštěno měření, se nacházelo na souřadnicích [1.0, 1.0], jak je vidět na obrázku 44. Po zpracování měření nám vyšla průměrná pozice [1.32, 1,26], tudíž chyba měření byla opět v okruhu do 1 m.



Obrázek 49 - Rozmístění AP a mobilního zařízení 2



Obrázek 50 - Testovací prostředí 1 - měření 2

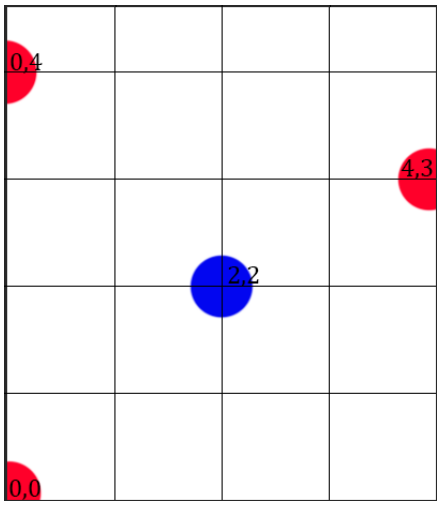
Testovací prostředí 2

Druhým prostředím bude bytová jednotka, která ovšem nemá tolik rušivých faktorů jako testovací prostředí jedna. Prvním rozdílem je počet aktivních AP v okolí. V tomto prostředí je zde pouze jedno zařízení AP (obr. 53), které by mohlo rušit naše testovací AP. Velikost souřadnicové sítě je 4x 4 m (obr. 51).

0,4	1,4	2,4	3,4	4,4
0,3	1,3	2,3	3,3	4,3
0,2	1,2	2,2	3,2	4,2
0,1	1,1	2,1	3,1	4,1
0,0	1,0	2,0	3,0	4,0

Obrázek 51 - Souřadnicová síť 2

Jednotlivá AP budou rozmístěna způsobem zobrazeným na obrázcích 52 a 55. Dále z těchto obrázku vidíme, jakým způsobem bude při jednotlivých měřeních umístěno mobilní zařízení, jehož polohu hodláme měřit.



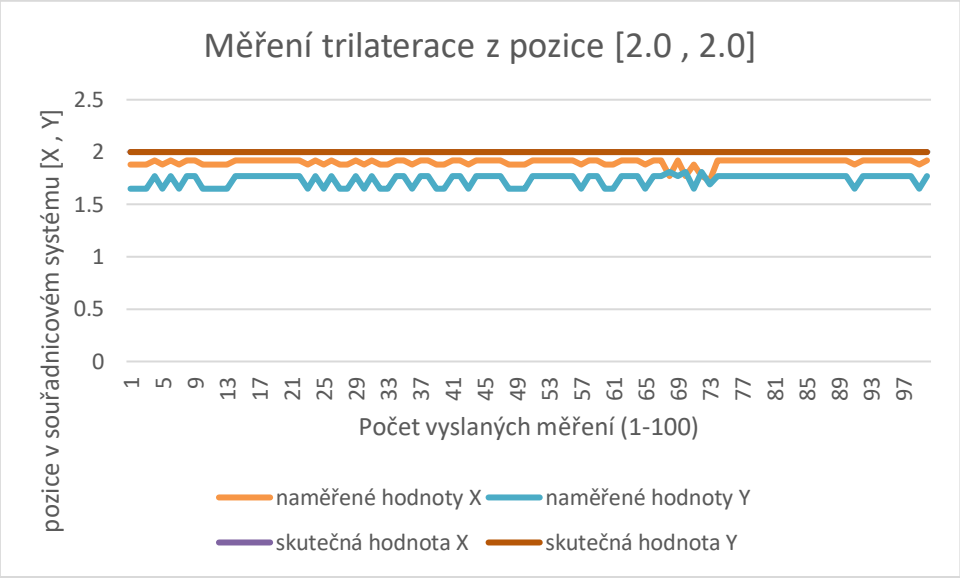
Obrázek 52- Rozmístění AP a mobilního zařízení 3

SSID	MAC Address	PHY Type	RSSI	Signal Quality	Average Signal...	Frequency	Channel	Information Size
WifiRTT1	70-3A-CB-7F-23-FF	802.11g/n	-48	86	90.2	2.462	11	209
WifiRTT1	70-3A-CB-7F-23-FB	802.11n/ac	-56	73	74.7	5.180	36	200
Souckova	BC-CF-4F-81-01-80	802.11g/n	-53	99	90.8	2.422	3	181
RTT test	70-3A-CB-7F-DC-B8	802.11g/n	-38	99	98.6	2.462	11	209

Obrázek 53 - Počet AP testovací prostředí 2

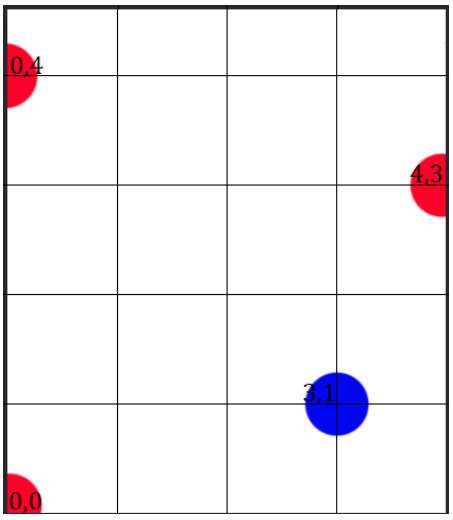
Při prvním měření v druhém testovacím prostředí je mobilní zařízení umístěno na pozici [2.0,2.0]. Po spuštění aplikace a získání všech potřebných dat byl znovu vypracován graf (obr. 54), podle kterého bude vyhodnocena kvalita měření.

Aplikace vyhodnotila průměrnou pozici zařízení jako [1.91, 1.74]. Tato poloha se znovu nachází v rozmezí do 0.5 metru.



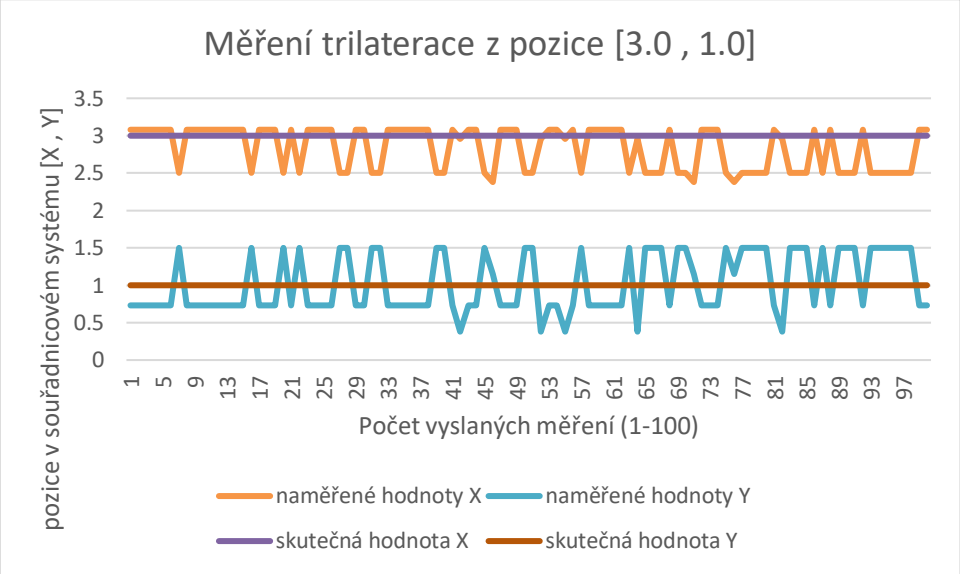
Obrázek 54 - Testovací prostředí 2 - měření 1

Druhé měření probíhalo stejným způsobem, jen umístění mobilního zařízení bylo na pozici [3.0,1.0](obr. 55). Aplikace vyhodnotila průměrnou pozici mobilního zařízení na [2.83,1.02] .



Obrázek 55 -Rozmístění AP a mobilního zařízení 4

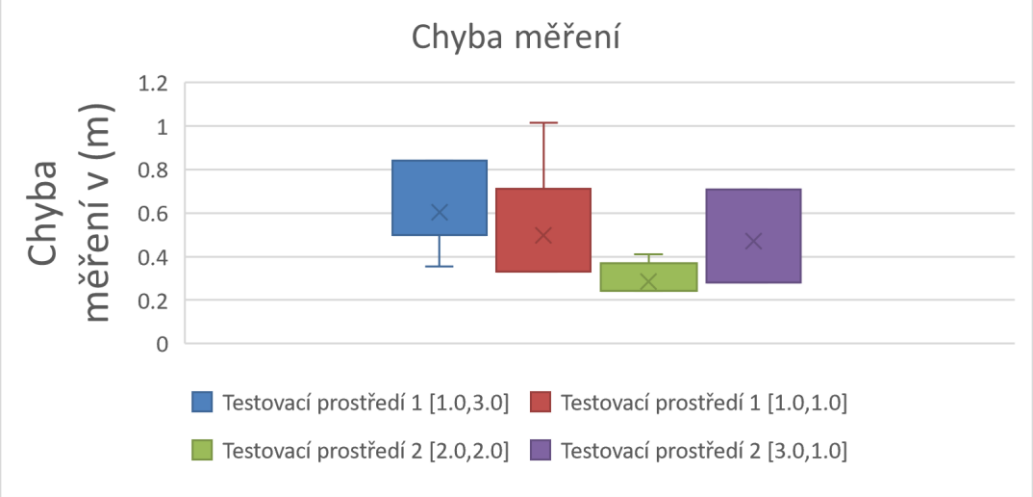
Toto měření mělo relativně velké odchylky. Jak je patrno z grafu (obr. 56), se hodnoty nedokázaly ustálit a blížit se k jedné hodnotě. Tuto chybu si vysvětlujeme tím, že za zdí vedle místa umístění mobilního zařízení se nachází další AP zařízení, které nejspíše odchýlilo naše rangingResulty.



Obrázek 56 - Testovací prostředí 2 - měření 2

10 Shrnutí výsledků

Pro otestování funkčnosti aplikace se sbírala data do textových souborů, které byly následně rozparsovány a otevřeny v excelu, kde se postupně jednotlivá data vyhodnocovala. Co se týče výsledků, aplikace vykazuje relativně stabilní výsledky. I když se práce odkazuje pouze na výsledky, které jsou uvedeny zde, tak každé měření probíhalo vícekrát a poté byl vybrán jeden náhodný příklad, který byl použit v této práci. Cílem práce bylo zjistit, zda technologie RTT skutečně funguje tak, jak je popisována, a zda dokáže měřit pozici (vzdálenost) s maximální chybovou odchylkou 2 m. Toto tvrzení můžeme po prozkoumání výsledků podpořit. Veškeré výsledky (průměr výsledků), které byly zpracovány, spadly právě do této chybové tolerance, jak je zobrazeno v grafu na obrázku 57. Tyto výsledky se zdají být až neuvěřitelně dobré, ale je možné, že dostáváme takto přesné výsledky právě z důvodu použití stísněného prostoru, kde se nachází minimum překážek, které by výsledky zkreslovaly.



Obrázek 57 - Graf ukazující rozmezí chyb při měření

11 Závěry a doporučení

Stanovené cíle práce se podařilo splnit. Práce nám odhalila, jaké jsou možnosti při lokalizaci nějakého subjektu, jakým způsobem lokalizace probíhá v reálném světě a jaké metody jsou k tomu využívány. Testovaná aplikace vyvíjená společností Google nám nastínila své možnosti, jakým způsobem pracuje s daty a co přesně probíhá v zařízení, které komunikuje s nějakým v okolí. Co se týče úpravy této aplikace, tak se povedlo aplikaci upravit do takového stádia, kdy uživatel po zapnutí aplikace povolí přístup ke sdílení polohy, následně se spustí okamžité vyhledávání všech AP v okolí. Z toho seznamu se poté vyberou pouze ty AP, které podporují námi testovanou technologii RTT. Tato zařízení se poté uživateli vypíšou v okně aplikace, kde se zobrazí SSID a BSSID jednotlivých zařízení. Poté má uživatel možnost pustit měření pomocí tlačítka „START RANGING“ po spuštění měření začne aplikace vysílat na již předem známá AP dotazy, které se následně zpracují. A uživateli se na obrazovce začne zobrazovat aktuální poloha zařízení vůči zmíněným AP.

Nyní se dostáváme k problémům, které nám testování přineslo. Jako každá aplikace ani tato není dokonalá. Některé naměřené hodnoty uvádějí extrémní výchylku, ale jelikož byla data zpracovávána formou průměrů, tak tyto extrémy v rámci vždy měřených 100 hodnot byly zanedbatelné. Jedním z klíčových problémů při testování aplikace byla vlastní ochrana Android systému. Každý Android systém se snaží svého uživatele nějakým způsobem chránit. První takovou překážkou bylo omezení scanování zařízení v okolí, kdy Android defaultně povoluje aplikaci vyslat takový požadavek pouze s prodlevou 2 minut. Tento problém mohl být řešen dvěma způsoby. Jedním z nich bylo zakázat WiFi scan throttling. Vypnutím této funkce se nám naskytla možnost scanovat zařízení v okolí 4x za dvě minuty tudíž cca jednou za 30 sec. Ovšem ani toto řešení nebylo úplně ideální, taková prodleva může způsobit mnoho problémů. Dalším řešení bylo vytvoření si vlastních výsledků scanování (scanResultů). Ovšem ani toto se neobešlo bez problémů. Jelikož Android brání uživateli vytvářet běžně takovéto scanResulty, musela být použita technika reflexe. Tato metoda ovšem není zrovna vítaná pro produkci takovýchto aplikací. Takže by nám v budoucnu tato technika bránila vydání aplikace v oficiálním Google

store. Ale pro naše potřeby to bylo dostačující. Jako další problém zde máme momentální omezení vlády ČR kvůli pandemii, protože bylo nutné testovat aplikaci v menších prostorech bytu. Rozhodně jedním z doporučení je testovat takovouto aplikaci ve větším prostředí, kde můžeme testovat různé náhodné situace a reakci aplikace. V neposlední řadě je jedno z doporučení pokusit se nalézt lepší metodu výpočtu. Pokud by se nalezla metoda, která dokáže lépe zpracovat odchylky výsledků a zpřesnit tak počítané údaje, tak by se teoreticky dal zpřesnit algoritmus počítání polohy. Dále by stálo za zvážení pokusit se o multilateraci. Bohužel z časových důvodů se na toto rozšíření nedostalo, ale určitě by stálo za to se nad tímto tématem zamyslet. Přidat možnost multilaterace a rozhodnout se, jakým způsobem řešit tuto problematiku, by bylo krásným tématem pro další práci.

12 Seznam použité literatury

[1] HALIT, Eren. *Radiolocation, Radio Navigation, and GPS Systems* [online]. [cit. 22.04.2020]. Dostupné z: https://www.researchgate.net/publication/264638301_Radiolocation_Radio_Navigation_and_GPS_Systems

[2] BINDNER, Patrick.*What are the advantages of a passive radar over an active radar?* [online]. [cit. 22.04.2020]. Dostupné z: https://www.quora.com/What-are-the-advantages-of-a-passive-radar-over-an-active-radar

[3] ZHOU Yang Dong, WEI Ming Xu, HAO Zhuang. *Research on ZigBee Indoor Technology Positioning Based on RSSI* [online]. [cit. 22.04.2020]. Dostupné z: https://www.sciencedirect.com/science/article/pii/S1877050919308294

[4] GAJAR, Manish J. *Mobile Sensors and Context-Aware Computing* [online]. [cit. 22.04.2020]. Dostupné z: https://www.sciencedirect.com/book/9780128016602/mobile-sensors-and-context-aware-computing

[5] VLADICA Sark, NEBOJSA Maletic, JESÚS Gutiérrez, ECKHARD Grass. *Performance Evaluation of a Time-of-Arrival Based Indoor Localization*. [online] [cit. 22.04.2020]. Dostupné z: https://www.researchgate.net/publication/339677733_Performance_Evaluation_of_a_Time-of-Arrival_Based_Indoor_Localization

[6] BIN Xu, GUODONG Sun, RAN Yu, ZHENG Yang. *High-Accuracy TDOA-Based Localization without Time Synchronization*. [online]. [cit. 22.04.2020]. Dostupné z: https://www.researchgate.net/publication/260358236_High-Accuracy_TDOA-Based_Localization_without_Time_Synchronization. 1558-2183.

[7] BURKE, Stephanie. *Wi-Fi CERTIFIED Location™ brings Wi-Fi® indoor positioning capabilities* [online]. [cit. 22.04.2020]. Dostupné z: https://wi-fi.org/news-events/newsroom/wi-fi-certified-location-brings-wi-fi-indoor-positioning-capabilities.

[8] MOHAMED Ibrahim, HANSI Liu, MINITHA Jawahar, VIET Nguyen. *Verification: Accuracy Evaluation of WiFi Fine Time Measurements on an Open Platform.* [online]. [cit. 22.04.2020]. Dostupné z: https://www.researchgate.net/publication/328327674_Verification_Accuracy_Evaluation_of_WiFi_Fine_Time_Measurements_on_an_Open_Platform

[9] *Trilateration vs Triangulation - How GPS Receivers Work* - GIS Geography. Home – GIS Geography [online]. Dostupné z: https://gisgeography.com/trilateration-triangulation-gps/

[10] KUN An, SIYANG Xie, YANFENG Ouyang. *Reliable sensor location for object positioning and surveillance via Trilateration.* [online]. [cit. 22.04.2020]. Dostupné z: https://www.sciencedirect.com/science/article/abs/pii/S0191261517310536.

[11] HORN Berthold K.P. *Indoor positioning using time of flight with respect to WiFi access points*. [online]. Dostupné z: http://people.csail.mit.edu/bkph/ftmrtt.

[12] *ScanResult | Android Developers. Android Developers* [online]. Dostupné z: https://developer.android.com/reference/android/net/wifi/ScanResult

[13] *android.net.wifi.ScanResult java code examples*. [online]. Dostupné z: https://www.codota.com/code/java/classes/android.net.wifi.ScanResult

[14] *wifi/java/android/net/wifi/ScanResult.java*. [online]. Dostupné z: https://android.googlesource.com/platform/frameworks/base/+/refs/heads/master/wifi/java/android/net/wifi/ScanResult.java

[15] FRISH Daniel, HANEBECK Uwe D. *Correction to: TDOA versus ATDOA for wide area multilateration systém*. [online]. Dostupné z: https://link.springer.com/article/10.1186/s13638-020-1656-1

[16] *Multilateration – an overview* [online]. [cit. 23.04.2020]. Dostupné z: https://www.sciencedirect.com/topics/engineering/multilateration

13 Přílohy

- Adresář WifiRTT_Aplikace
  - Zdrojové soubory aplikace
- Adresář Data_Měření_BP
  - bokem.txt – surová data z měření z polohy bokem
  - bokemZed.txt - surová data z měření z polohy bokem přes zeď
  - celem.txt - surová data z měření z polohy čelem
  - celemZed.txt - surová data z měření z polohy čelem přes zeď
  - zady.txt - surová data z měření z polohy zády
  - zadyZed.txt - surová data z měření z polohy zády přes zeď
  - trilateraceTest1_1.1.txt - surová data z měření polohy pomocí trilaterace z bodu [1.0, 1.0] . prostředí 1
  - trilateraceTest1_1.3.txt - surová data z měření polohy pomocí trilaterace z bodu [1.0, 3.0] – prostředí 1
  - trilateraceTest2_2.2.txt - surová data z měření polohy pomocí trilaterace z bodu [2.0, 2.0] . prostředí 2
  - trilateraceTest2_3.1.txt - surová data z měření polohy pomocí trilaterace z bodu [3.0, 1.0] . prostředí 2
  - mereni_vzdalenosti.xlsx – soubor, který obsahuje zpracovaná veškerá surová data, v souboru nalezneme všechny grafy a data, z nichž byly grafy vytvořen

Univerzita Hradec Králové
Fakulta informatiky a managementu

Zadání bakalářské práce

Autor: **Jiří Jirásek**

Studium: I1700092

Studijní program: B1802 Aplikovaná informatika

Studijní obor: Aplikovaná informatika

Název bakalářské práce: **WiFi lokalizace pomocí 802.11mc Fine Time Measurement / RTT**

Název bakalářské práce AJ: WiFi localization using 802.11mc Fine Time Measurement / RTT

Cíl, metody, literatura, předpoklady:

Cíle práce:

Navrhnout a implementovat aplikaci pro OS Android ver. 9 a vyšší, pomocí které bude prováděno měření vzdáleností k jednotlivým AP v rámci experimentů. Vyhodnocení experimentů (zaměření zařízení pomocí trilaterace, zjištění chyby).

Osnova:

1. Úvod

2. Cíle práce

3. Teoretická část

    Radio Lokalizace

    WiFi -Fine time measurement

4. Praktická část

    Návrh a implementace aplikace

    Experimentální měření a jejich vyhodnocení

5. Závěr

6. Použitá literatura a jiné zdroje

7. Přílohy

Ibrahim : Verification: Accuracy Evaluation of WiFi Fine Time Measurements on an Open Platform

Soltanaghaei : Multipath Triangulation: Decimeter-level WiFi Localization and Orientation with a Single Unaided Receiver

Horn : http://people.csail.mit.edu/bkph/ftmrtt

Garantující pracoviště: Katedra informatiky a kvantitativních metod, Fakulta informatiky a managementu

Vedoucí práce: Ing. Pavel Kříž, Ph.D.

Datum zadání závěrečné práce: 14.1.2018