DIPLOMOVÁ PRÁCE

Bc. Viktor Vašina

Možnosti indoor geolokace mobilních zařízení

Katedra informatiky a výpočetní techniky

Vedoucí diplomové práce:	Ing. Ladislav Pešička
Studijní program:	Inženýrská informatika
Studijní obor:	Softwarové inženýrství

Plzeň 2017

Prohlašuji, že jsem tuto diplomovou práci vypracoval(a) samostatně a výhradně s použitím citovaných pramenů, literatury a dalších odborných zdrojů.

V ..... dne ..... Podpis autora

Název práce: Možnosti indoor geolokace mobilních zařízení

Autor: Bc. Viktor Vašina

Katedra: Katedra informatiky a výpočetní techniky

Vedoucí diplomové práce: Ing. Ladislav Pešička, Katedra informatiky a výpočetní techniky

Abstrakt: Tato práce je analýzou metod pro určování polohy mobilních zařízení uvnitř budov. Jednotlivé metody jsou popsány a porovnány z hlediska potřeby dodatečného softwaru nebo hardwaru, nákladů, doby nasazení, přesnosti a náročnosti realizace. Na základě této analýzy byla jako součást práce implementována aplikace umožňující určení mobilního zařízení uvnitř budovy a zobrazení této polohy na plánu budovy. Cílem práce není experimentálně ověřit všechny dostupné metody pro lokalizaci uvnitř budov, ale nalezení lokalizační metody, která bude dostatečně přesná a zároveň dostupná.

Klíčová slova: indoor lokalizace Bluetooth trilaterace bez GPS mobilní zařízení

Title: Posibilities of indoor geolocation of mobile devices

Author: Bc. Viktor Vašina

Department: Department of Computer Science and Engineering

Supervisor: Ing. Ladislav Pešička, Department of Computer Science and Engineering

Abstract: This thesis is an analysis of methods for determining the location of mobile devices inside buildings. The individual methods are described and compared to the need for additional software or hardware, cost, deployment time, accuracy, and performance. Based on this analysis, as a part of the thesis an application has been, that allows the determination of the mobile device inside the building and the display of this position on the building plan. The aim of the thesis is not to experimentally verify all available methods for localization within buildings, but to find a localization method that will be sufficiently accurate and available at the same time.

Keywords: indoor localization Bluetooth trilateration no GPS mobile devices

Děkuji svému vedoucímu práce Ing. Ladislavu Pešičkovi za cenné rady, věcné připomínky a vstřícnost při konzultacích a vypracování diplomové práce.

Obsah

Úvod

Cílem této práce je nastínit čtenáři možnosti lokace mobilních zařízení uvnitř budov nebo pod zemí, a na základě těchto poznatků navrhnout a implementovat aplikaci pro indoor navigaci, která bude dostatečně flexibilní pro nasazení uvnitř běžných budov.

V první části práce popisuje základní znalosti nutné pro pochopení lokalizačních metod – tedy jak může být poloha zařízení popsána a jak je tato informace reprezentována v mobilních zařízeních.

Na základně znalosti reprezentace polohy zařízení jsou dále popsány principy, na kterých jsou postaveny moderní lokační metody.

V další části jsou postupně analyzovány metody pro lokalizaci zařízení. U jednotlivých metod je popsán princip fungování, jejich přednosti a nedostatky.

Vybrané metody jsou využity v implementaci navigace pro platformu Android. Součástí řešení jsou pouze metody založené na hardwaru mobilních zařízení. Metody vyžadující specializovaný hardware pro uživatele nejsou zahrnuty z důvodů minimalizace nákladů a snahy o využití již existujících technologií.

Po přečtení práce čtenář získá znalosti potřebné pro pochopení indoor lokalizace zařízení. Práce popisuje již dobře známé metody i experimentální postupy včetně jejich úskalí.

1. Lokace zařízení

Nejdříve je nutné vymezit, jak bude pro účely této práce chápán pojem *lokace*. *Lokaci* lze chápat jako sloveso popisující proces určování polohy zařízení, nebo jako podstatné jméno vyjadřující pozici v prostoru. V této práci bude pojem *lokace* používán ve smyslu procesu určování polohy zařízení. Pro popis polohy bude využíván právě výraz *poloha* nebo *pozice*.

Určování polohy mobilních zařízení (a jejich uživatelů) je v dnešní době velice žádané, ať už kvůli navigaci v dopravě nebo kvůli reklamním účelům v nákupních střediscích. Tyto dva příklady se ovšem zásadně liší v samotném základu – významu pozice.

Pozice v prostoru je popsána souřadnicemi. Souřadnice popisují polohu vzhledem k nějakému počátku soustavy. Výše zmíněné příklady se zásadně liší právě v počátku soustavy. Základem dopravní navigace je pohyb mezi státy, městy atd., tím pádem pracuje s pozicí uživatele vzhledem k planetě Zemi. Počátkem soustavy je střed Země. Pro navigaci uvnitř malých prostor, jako jsou místnosti, budovy nebo jeskyně, je poloha vzhledem k Zemi nezajímavá. Jako počátek soustavy se tedy využívá nějaký význačný bod v místnosti, např. vchod nebo roh.

1.1 Zeměpisná poloha

Jedná se o pozici bodu v prostoru, která se vztahuje k planetě Zemi. Přesněji je pozice bodu popsána sférickými souřadnicemi, přičemž počátek soustavy je ve středu Země [1, 2].

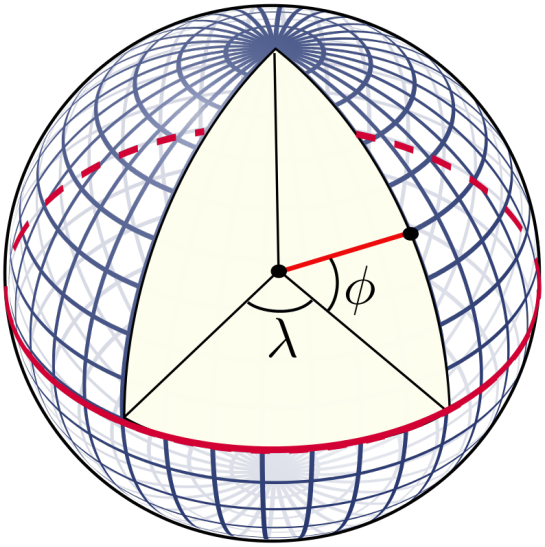
Zeměpisná délka (longitude) je úhel sevřený rovinou nultého poledníku a rovinou poledníku [1, 2], který protíná určovaný bod. Může nabývat hodnot z intervalu $x \in [0°, 180°]$ východní (od nultého poledníku směrem na východ) nebo západní (od nultého poledníku směrem na západ) délky. Spojení *východní, resp. západní délka* je dlouhé a používá se spíše v mluveném projevu. Pro zápis se využívá zkratka W (z ang. West pro západní délku) nebo E (z ang. East pro východní délku).

Zeměpisná šířka (latitude) je úhel mezi rovinou rovníku a spojnicí středu Země s hledaným bodem [1, 2]. Může nabývat hodnot z intervalu $x \in [0°, 90°]$ severní, resp. jižní šířky. I zde se využívá zkratek N (z ang. North pro severní šířku) nebo S (z ang. South pro jižní šířku).

Zeměpisná výška je údaj, který se běžně nepoužívá. Jedná se o vzdálenost hledaného bodu od středu Země, tedy počátku soustavy. Při používání zeměpisných souřadnic se bod automaticky uvažuje na povrchu planety (vzdálenost od počátku je tedy poloměr planety) nebo se upřesňuje pomocí nadmořské výšky.

Nadmořská výška je vzdálenost bodu od hladiny oceánu [2], ta tvoří pomyslnou nulovou výšku pro orientační účely. Je součástí zeměpisné výšky. Země-

pisnou výšku lze popsat i jako součet vzdáleností od počátku k hladině oceánu a od hladiny oceánu k určovanému bodu.



Obrázek 1.1: Zeměpisná šířka (ϕ) a zeměpisná délka (λ).

1.1.1 Reprezentace zeměpisné polohy

Pokud chceme informaci o zeměpisné poloze reprezentovat, narazíme na dvě hlediska:

- srozumitelnost pro člověka
- strojová přenositelnost

Pro člověka je nejčitelnější tzv. *DMS (degrees minutes seconds)* [3] zápis souřadnic. Např. zeměpisné souřadnice města Plzně:

$$49°44'18.3516''N$$

$$13°22'25.0932''E$$

Tento zápis je ovšem pro elektronickou přenositelnost komplikovaný. Proto se zde využívá formát *DDD* (degrees degrees degrees) [3], kdy je souřadnice převedena do dekadické soustavy:

$$49.738431(latitude)$$

$$13.373637(longitude)$$

Všimněme si, že zmizely písmena N a E identifikující severní šířku a východní délku. Pro další zjednodušení formátu souřadnic jsou jižní šířka a západní délka chápány v záporném smyslu. Město Plzeň leží v severovýchodním kvadrantu zeměpisných souřadnic, a proto jsou oba údaje kladné. Rio de Janeiro leží v opačném kvadrantu:

$$-22.906847(latitude)$$

$$-43.172896(longitude)$$

Systém zeměpisné polohy je vhodný pro navigaci na cestách či orientaci ve městě, tedy maximálně na úrovni budov a ulic. Ovšem pro určování polohy uvnitř budov je tento systém nevhodný. Pozice zařízení uvnitř místnosti bude vyžadovat vysokou přesnost (řádově jednotky metrů), protože třímetrová odchylka může činit rozdíl mezi dvěma místnostmi nebo obchody. Navíc poloha z hlediska Země je v takovém případě nepotřebná informace, a určování polohy by spíše komplikovala, protože by bylo nutné zaměřovat klíčové body místností, jako např. rohy, vzhledem k Zemi.

1.2 Indoor poloha

Pro navigaci uvnitř budov je vhodnější využít kartezskou soustavu souřadnic, tedy soustavu, jejíž osy jsou navzájem kolmé a protínají se v jediném bodě – počátku soustavy. Tento počátek lze umístit např. do středu, rohu nebo vchodu místnosti či budovy. Pozice bodu v místnosti je v tomto případě vztažena k objektu nikoliv k planetě Zemi. Díky tomu lze klíčové body budovy zaměřovat jen vzhledem k počátku soustavy, a na to stačí obyčejný metr nebo stavební plán budovy.

Pokud se jedná o jednu místnost či jednopatrovou budovu, je možné ze souřadnicového systému vypustit výškovou osu, a tak trojrozměrnou úlohu (3D) zjednodušit na dvourozměrnou (2D).

1.3 Druhy lokace

Lokace je tedy proces, který transformuje ukazatele z okolí na polohu. Obecně lze tedy zapsat lokaci jako:

$$L(\vec{o}) = \vec{p}$$

Lokaci lze tedy chápat jako funkci (L) vektoru faktorů okolí ($\vec{o}$ – poslední známá pozice, intenzita signálů WiFi, nadmořská výška, ...), jejíž výsledkem je pozice ($\vec{p}$), tedy vektor popisující pozici zařízení.

Lokaci lze rozdělovat dle různých hledisek:

- vstupního vektoru (intezita WiFi signálu, vektor magnetického pole, poslední známá pozice, ...)
- způsobu transformace vstupního vektoru na pozici (trilatera, triangulace, rozpoznávání obrazu, ...)
- výstupního vektoru pozice

Výsledná pozice může obsahovat různé informace. Za nutné *minimum* můžeme považovat horizontální souřadnice polohy $\vec{p}(x, y)$ pro indoor polohu nebo $//\vec{p}(latitude, longitude$ pro zeměpisnou polohu.

Pozice ovšem může obsahovat další informace jako nadmořskou výšku, číslo podlaží nebo např. azimut.

Pokud není nutné implementovat aplikaci typu navigace pro řidiče, vystačíme si z výše zmíněným minimem, které bude zohledněno při výběru lokačních algoritmů ve výsledné aplikaci.

Posledním faktorem ovlivňující výsledný systém je případné sdílení polohy. Lokace zařízení nemusí sloužit pouze pro potřeby uživatelů, ale i provozovatelů:

- přehled, kudy se zákazníci pohybují pro reklamní účely
- šetření energií – vypínání osvětlení nebo snižování teploty pro neobsazené prostory
- záchrané účely v případě ztracených dětí v obchodních centrech atp.

Toto hledisko nebude pro aplikaci uvažováno, protože se jedná o sdílení polohy přes aplikační server nebo internet, a to již není přímo součástí této práce. Možnost sdílení polohy tak ponechávám pro případné rozšíření výsledné aplikace.

2. Principy lokačních metod

Existuje několik způsobů jak lokalizovat zařízení v budově či mimo ni. Než budou jednotlivé technologie detailně analyzovány, je nutné pochopit, na jakých principech stojí.

2.1 Lokace orientačními body

Jedná se nejjednodušší způsob lokace, který všichni známe např. z informačních tabulí v nákupních střediscích. Velký bod na plánku nákupního centra označující aktuální polohu zákazníka.

Systém je tedy postavený na orientačních bodech, které jsou zaneseny v plánu místnosti nebo patra budovy. Pokud tyto body vidíme, znamená to, že jsme v jejich těsné blízkosti. V plánu jsou dále zaneseny body zájmu (dále POI – points of interest), jako např. obchody, eskalátory nebo východy. Díky tomu získáme snadno představu o svém okolí a kam se dále vydat.

Díky mobilním zařízením tyto plány nemusejí být veliké tabule uprostřed chodby. Plán může být součástí aplikace v mobilním zařízení a v budově mohou být rozesety jiné orientační body, které bude mobilní zařízení schopné detekovat prostřednictvím svých senzorů, fotoaparátu nebo bezdrátových technologií [37, 36].

2.2 Triangulační lokace

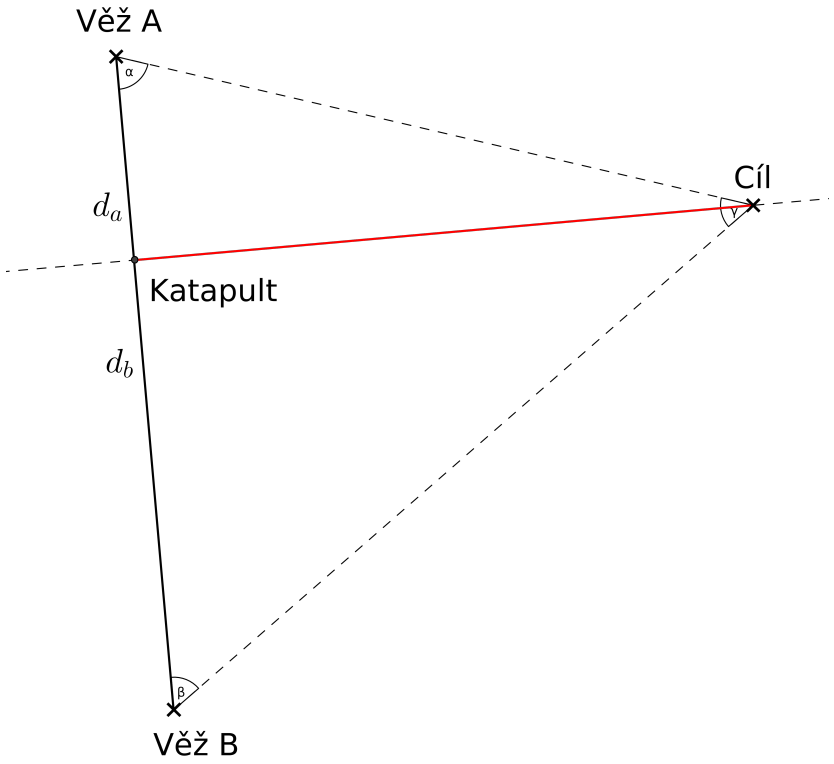
Triangulace [4] (často zaměňována s trilaterací) je postup pro měření vzdáleností, který se vyvinul v dobách, kdy bylo jednodušší změřit opticky úhly mezi body (např. pomocí teodolitu), než jejich vzdálenosti. Jak název napovídá, tato metoda využívá znalost všech tří vnitřních úhlů uvnitř trojúhelníku a délky jedné jeho strany.

Pokud jsou známy dva statické body (A a B) a vzdálenost mezi nimi $|AB|$ (základnou c), můžeme určit pozici třetího bodu C změřením vnitřních úhlů trojůhelníku ABC přilehlých k základně c. Jelikož součet vnitřních úhlů trojúhelníku je π rad, lze spočítat i velikost třetího úhlu. Následně využijeme sinovu větu pro výpočet vzdáleností $|AC|$ a $|BC|$.

Na obrázku (2.1) je znázorněna triangulace cíle za pomoci dvou věží, přičemž předpokládáme, že *katapult* je umístěn do paty výšky trojúhelníku.

Tento postup není vhodný pro lokaci na místech, kde se pohybuje více lidí. Vyžaduje aktivní referenční body, tj. body musí měřit úhel mezi jejich spojnicí a zaměřovaným bodem. To se děje optickou cestou a lokalizace desítek nebo stovek mobilních zařízení by byla velmi problematická.

Metody založené na triangulaci jsou použitelné pro specifičtější účely lokace, kde se lokalizuje jeden bod (nebo předem známý počet) v kontrolovaném prostředí



Obrázek 2.1: Triangulace cíle C pomocí dvou známých věží A, B a jejich spojnice c.

pomocí optického snímání. Do této skupiny patří např. mapování animovaných postav na herce ve speciálních kostýmech.

2.3 Trilaterační lokace

Trilaterace [5] je metoda mnohem mladší než triangulace. Důvodem je dřívější technologické omezení. Technika pro elektronické měření vzdálenosti se vyvinula až ve 20. století.

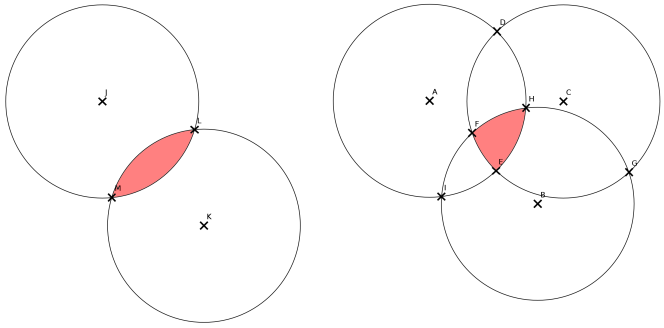
Tato metoda bývá často chybně označována jako triangulace. Princip je ale jiný. Trilaterace je založená na znalosti vzdálenosti od alespoň tří referenčních bodů. Známe-li pozice tří bodů (A, B, C) v prostoru, potřebujeme pro zaměření pozice (D) pouze znát vzdálenosti bodu D od známých tří bodů. Tyto vzdálenosti poslouží jako poloměry kružnic se středy v bodech A, B *a* C. Tyto kružnice se protnou v jediném bodě D.

To by byl ovšem ideální scénář. K měření vzdálenosti se využívají bezdrátové technologie (WiFi, Bluetooth nebo telefonní signál), což znamená, že měření vzdálenosti od referenčních bodů není tak precizní, aby se kružnice protnuly v jediném bodě. Místo toho se využívá kruhů a poloha bodu D se určí jako střed průniku kruhů.

Obrázek 2.2 popisuje trilateraci mobilního telefonu pomocí dvou atří vysílačů. Z obrázku je patrné, že pro případ kruhů stačí pro hrubé určení polohy pouze dva vysílače. Ovšem pokud by šlo o určení polohy pomocí kružnic, existovaly by dvě vyhovující polohy, proto je zde nutná další kružnice.

V oblasti lokace mobilních zařízení se využívají dva způsoby získání vzdálenosti:

- síla signálu [6]
- ToF (Time of Flight) - doba, za kterou se signál přenese z vysílače na přijímač [7]



Obrázek 2.2: Trilaterace pomocí signálů 3 vysílačů.

2.4 Lokace rozpoznáváním pohybu

Další možností lokace je určení přesné startovací polohy zařízení a následné stopování pohybu. Zařízení sleduje svůj vlastní pohyb z hlediska směru a vzdálenosti [11].

Ve spojení se startovací polohou je tedy možné určit, kudy se zařízení pohybovalo a tím jeho polohu. Základem této metody je akcelerometr, tedy senzor, který v reálném čase poskytuje informace o změnách zrychlení ve všech třech osách. Osy akcelerometru jsou spjaté se zařízením. Pokud tedy zařízení otočíme, bude poskytovat stejné hodnoty, jen na jiných osách. Akcelerometr je schopný poskytnout pouze informace o zrychlení v nějakém směru a o směru gravitace (gravitační zrychlení je v rámci budovy konstantní). To ovšem ke stopování pohybu nestačí. Akcelerometr není schopen zachytit rotační pohyb, tedy směrovou orientaci.

Informaci o směru pohybu je nutné získat jinou cestou např. spojením s trilaterací nebo zapojením gyroskopu či magnetometru.

2.5 Lokalizace pomocí otisků

Metody založené na otiscích (fingerprints) [10, 13] využívají toho, že konkrétní místo v budově má svůj otisk. Jako otisk může sloužit intenzita WiFi nebo Blu-

etooth signálů nebo např. echolokační otisk místnosti.

Otisky se ovšem musí fyzicky nasbírat a spojit s konkrétní polohou – metody tedy spadají do oblasti učení s učitelem.

2.6 Vizuální lokace

Do této kategorie spadají metody využívající fotoaparát mobilního zařízení.

První metoda se vlastně snaží napodobit základní orientaci člověka v prostoru pomocí orientačních bodů. „Pokud vidím tyto čtyři obchody a vstup na toalety, musím se nacházet v prvním patře blízko eskalátoru.“

Druhá metoda [11] možná již není tak zřejmá. Vychází z předpokladu, že uživatel drží své mobilní zařízení v takové poloze, že zabudovaný fotoaparát dokáže snímat alespoň část podlahy v blízkosti uživatele. Pokud bude známá výška a náklon fotoaparátu od země, lze z rozdílu snímků vypočítat, jak se telefon pohybuje (směr a rychlost).

2.7 Magnetická lokalizace

Některá mobilní zařízení jsou již vybavena 3D magnetometrem. Ten měří intenzitu magnetického pole ve třech osách vzhledem k mobilnímu zařízení.

Metoda využívá faktu, že samotné magnetické pole země je deformované interiérem staveb. Typicky kabely, jimiž protéká elektrický proud, generují vlastní magnetické pole. Tyto odchylky jsou unikátní a lze jich využít k určení míst, kterými mobilní zařízení prochází.

3. Analýza dostupných lokačních metod

Aplikací výše zmíněných lokalizačních principů je celá řada. Pro zvýšení přesnosti se typicky používá kombinace více metod.

Součástí charakteristiky lokalizační metody není pouze algoritmus a přesnost, spadají sem i další kritéria:

- potřebný hardware
- možnost využití stávající infrastruktury
- energetická náročnost (napájení hardware a výdrž baterie mobilního zařízení)
- cena komponent
- doba nutná pro nasazení (důležité např. pro algoritmy zahrnující učení s učitelem)

3.1 Družicové lokační systémy

Do této skupiny [21] patří systémy jako GPS (USA) [17], GLONASS (Rusko) [20], Beidou (Čína) [18] či GALILEO (EU) [19]. Jsou to systémy založené na trilateraci. Referenčními body jsou zde družice kroužící po orbitálních drahách kolem planety (vyjímku tvoří několik geostacionárních družic Beidou). Vzdálenost je získávána z časového rozdílu mezi odesláním a přijetím signálu, za předpokladu znalosti rychlosti šíření signálu.

Družice vysílají svá data [16] – konkrétně časovou značku vnitřních hodin, *efemeridy* a *almanach*.

Almanach – obsahuje údaje o celém systému, tj. detaily o všech družicích.

Efemeridy – odhady pozic jednotlivých družic v konkrétních okamžicích.

Pokud přijímač na Zemi zachytí časové značky z minimálně čtyř družic, je schopen díky rozdílu času od vyslání po příjem vypočíst vzdálenosti od družic. Trilaterací je dále možné zjistit polohu přijímače. Zde ovšem nebude pozice zařízení průnikem tří kruhů, nýbrž průnikem čtyř koulí.

Družice jsou tedy z hlediska uživatele pasivní, jednoduše stále vysílají své časové značky a upřesňující zprávy of *efemeridech* (každých 30 sekund). Přijímač těchto signálů musí vyčkat, až rozpozná signály alespoň od čtyř družic a poté může vypočítat svou polohu.

Při tzv. *studeném startu* (přijímač nemá k dispozici *efemeridy*, *almanach* ani svou aktuální pozici) musí zařízení vyčkat na stažení potřebných informací a následně vyčkat na signál od minimálně čtyř družic. To trvá dlouho (12 minut) z

důvodu přenosové rychlosti 50 bps. Tuto dobu mohou prodloužit i okolnosti jako výhled na oblohu a doba vysílání posledních *efemeridů* (interval je 30 sekund) [26]. V případě, že se přijímač nachází v centru města, kde není dobrý výhled na oblohu, může být tzv. *TTFF* (Time to First Fix – doba inicializace a získání polohy přijímače) více jak 12 minut.

V případě vlažného startu (zařízení zná svou přibližnou polohu s přesností kilometrů) je *TTFF* již rychlejší a limitem je zde pouze okolní terén (budovy, výhled na oblohu) a 30 sekundový interval mezi vysíláním *efemeridů*. Možnost rychle využít družicové systém tedy závisí na informacích, které má přijímač o své poloze, čase a případně almanachu družic. Tyto informace lze získat i z jiných zdrojů (i za cenu odchylky v datech) pomocí internetu.

Stažení dat [22, 23, 24, 25, 26] z družic 50 bps a v případě systému GALILEO 25 bps je kritická část. Tento problém je řešený dodatečnými asistenčními službami jako je např. *A-GPS*. Mobilní zařízení má díky WiFi nebo telefonnímu signálu k dispozici informaci o své přibližné poloze. S touto informací může kontaktovat přes internet servery asistenční služby a stáhnout si *almanach* a *efemeridy* potřebných družic ve své oblasti. Tato výměna je daleko rychlejší a než čekání na potřebné informace od družic, navíc funguje i v případě, že signál od družic není z nějakého důvodu dostupný. Díky tomu si může mobilní zařízení např. uvnitř budovy prostřednictvím WiFi aktualizovat *almanach a efemeridy* a po opuštění budovy bude *TTFF* daleko kratší.

Přehled zajímavých údajů nabízí tabulka 3.1. Nicméně je třeba podotknout, že se tyto údaje mění a nezřídka různé zdroje uvádí odlišné informace. Zejména počet družic a přesnost systému, přičemž se jedná o horizontální přesnost. Přesnost vertikální (nadmořská výška) je nižší a pohybuje se v násobcích horizontální přesnosti.

Za zmínku stojí pět družic systému Beidou, které nekrouží nad Zemí, ale jsou tzv. geostacionární, tedy vzhledem k Zemi se nepohybují [18].

Tabulka 3.1: Přehled údajů o nejznámějších satelitních pozičních systémech. [22, 23, 24, 25]

Údaj	*GPS*	*GLONASS*	*GALILEO*	*Beidou*
Přenos (bps)	50	50	25	50
Opakování efemeridů (s)	30	30	50	30
Aktualizace efemeridů (hod)	2	0.5	3	1
Počet družic	32	37	24	35
Přesnost (m)	3.5	4-7	4	10
Počet družic i přesnost systému se stále mění, některé zdroje se dokonce rozcházejí v údajích. U systémů je uveden zamýšlený počet družic, některé systémy jako GALILEO a Beidou teprve satelity nasazují.				

Problémem této metody je, že nelze využívat uvnitř budovy. Signál není dostupný uvnitř budov, navíc v oblastech s velkým počtem rušení v podobě elektromagnetického záření či budov, od kterých se signál odráží, dochází k dodatečným chybám a nepřesnosti určení polohy.

Tato varianta je tedy pro indoor navigaci nevhodná.

3.2 Trilaterace pomocí signálů

Záměrně volím slovo signál, protože k trilateraci zařízení lze využít více signálů – GSM, Bluetooth, WiFi a dokonce zvuk.

3.2.1 GSM Trilaterace

Nejjednodušší a nejméně přesné určení pozice telefonu je pomocí telefonních vysílačů (BTS – base transceiver station) [27]. Každý vysílač má svou unikátní identifikaci (ID). Pokud tedy telefon komunikuje s konkrétním vysílačem, je v kruhu, jehož středem je vysílač. Poloměr kruhu tvoří dosah vysílače. Metodu lze nadále zpřesnit díky znalosti poklesu intenzity signálu s rostoucí vzdáleností (klesá s rostoucím kvadrátem vzdálenosti). Dalším zpřesnění poskytuje dostupnost dalších vysílačů.

V GSM (z fr. Global Spécial Mobile) mobilních sítích funguje časový multiplex, který umožňuje aby na jedné frekvenci komunikovalo více zařízení. Každé zařízení má svůj časový slot na dané frekvenci. Aby toto mohlo fungovat, musí telefon i vysílač vysílat data s předstihem. Kvůli tomu musejí znát vzdálenost mezi sebou. Ta se v GSM určuje pomocí *Timing Advice*(TA), která rozděluje dosah vysílače na 64 pásem po 550 metrech.

I při možnosti využít informace z více dostupných vysílačů, se budeme stále pohybovat s přesností v řádech desítek nebo stovek metrů. Navíc např. u OS Android existuje v API metoda pro získání všech dostupných vysílačů, ovšem nemusí být podporována hardwarem. Díky tomu dostaneme často informaci pouze o vysílači, se kterým aktuálně komunikujeme a tím pádem i pouze jeho TA hodnotu.

3.2.2 WiFi trilaterace

WiFi [] je technologie pro bezdrátový přenost internetu skrze přístupové body (AP – access points). Funguje v nelicencovaných pásmech 2.4 a 5 GHz.

Lokace pomocí WiFi se dnes již běžně používá. Firmy jako Google [28] mají rozsáhlé databáze MAC adres jednotlivých AP a k nim přiřazená zeměpisné souřadnice. Díky tomu, pokud je na telefonu zapnutá WiFi, stačí když zjistí MAC adresy zařízení v dosahu a pošle dotaz prostřednictvím internetu. Polohu lze ještě zpřesnit pomocí intenzity signálu. Ovšem tímto způsobem se pohybujeme na granularitě budov, protože WiFi má dosah až desítky metrů a MAC adresa AP umístěného v kavárně nám pouze napoví, že se nacházíme v kavárně nebo její těsné blízkosti.

V nákupních střediscích a jiných veřejných budovách je dnes WiFi pokrytí již běžné. Na druhou stranu tím, že má WiFi poměrně velký dosah (i skrze podlahu), jsou AP umístěny tak, aby jich k pokrytí stačilo co nejméně. Což je vlastně proti

myšlence trilaterace. Navíc signál je ovlivněn zdmi, nábytkem, osobami a dokonce natočením telefonu vůči vysílači. S pomocí nynějšího vybavení WiFi v budovách získáme pro běžnou trilateraci přesnost při nejlepším na úrovni místností [8]. WiFi trilaterace by tedy byla použitelná, ale vyžadovala by daleko více AP v budově, což by bylo náročné finančně i energeticky.

Další možností je běžnou trilateraci spojit s dalšími metodami, které eliminují nedokonalosti WiFi signálu jako je např. *multipath* [8, 7] efekt. Což je jev, kdy se do zařízení dostane signál z vysílače nejen přímou cestou, ale také odrazem o stěny. Díky tomu na zařízení dorazí několik kopií stejné informace nebo spolu mohou tyto odrazy dokonce interferovat. Výsledkem je např. zkreslená síla signálu a tím i vzdálenost od vysílače.

Možným budoucím řešením jsou technologie jako *SpotFi* [8], *Chronos* [7] nebo *Widar* [9]. Ty využívají stávající WiFi standard a snaží se zvýšit jeho přesnost až na desítky centimetrů využitím dalších metod.

3.2.3 Bluetooth trilaterace

Nasazení Bluetooth pro trilateraci se v mnohém shoduje s WiFi. Dokonce operuje i ve stejném pásmu 2.4 GHz. Technologie Bluetooth [29, 30] byla vytvořena pro bezdrátovou komunikaci na krátké vzálenosti (cca 10 metrů bez překážek). Hardware potřebný ke komunikaci je tak menší a levější.

Starší verze Bluetooth (1, 2.1, 3) fungují na principu komunikace *master – slave* [29, 30], tedy jeden řídící uzel a ostatní uzly vedlejší. Vyhledávání Bluetooth zařízení v okolí navíc trvá dlouho (jednotky až desítky sekund). Tyto verze nejsou vhodné pro časté dotazy na intenzitu signálu.

Díky Bluetooth LE (Low Energy) [30] se ovšem situace změnila. Tento standard mj. umožňuje rozesílání krátkých periodických broadcastových zpráv, které může zachytit kterékoliv naslouchající zařízení v okolí. Díky těmto zprávám lze zjistit i intenzitu signálu a tím i vzdálenost. Při návrhu standardu bylo dbáno i na energetickou náročnost. Díky tomu tyto vysílače mohou rozesílat své zprávy a k napájení stačí běžné baterie, které postačí po dobu několika měsíců.

Menší dosah zařízení je svým způsobem výhodou. Není zde snaha o pokrytí prostoru minimem vysílačů, ba naopak. Díky nižší ceně a dlouhé výdrži baterií lze použít desítky takových vysílačů pro jediné patro budovy.

Bluetooth LE se tedy jeví jako vhodná technologie pro trilateraci.

3.2.4 Zvuková trilaterace

Stejně jako u Bluetooth nebo WiFi lze využít dobu cesty signálu k výpočtu vzdálenosti. V tomto případě neputuje signál rychlostí světla, ale jedná se o zvukové signály šířící se rychlostí zvuku.

Tato metoda je možná snadnější pro pochopení komplikací spjatých s rušením signálu, které se týká i Bluetooth a WiFi. I zde dochází k rušení prostřednictvím

útlumu zvuku kvůli interiéru a lidem v místnosti. Stejně tak se zde můžeme setkat s efektem *multipath* díky odrazům zvuku od stěn. Komplikací je i fyzické omezení zvukového rozsahu mikrofonu a reproduktoru telefonu.

Jejich rozsah je navíc omezen tím, že zvukové signály systému musejí být mimo slyšitelné spektrum lidského ucha.

Výhodou je, že mikrofon pro zachycení zvuku mají všechny telefony. V praxi jsem ovšem narazil na využití zvuku pouze pro měření vzdálenosti pomocí ultrasonických senzorů u autonomních zařízení jako jsou inteligentní vysavače.

3.3 Fingerprinting

Jak název napovídá, jde o metodu otisků. Otiskem se je v tomto případě nějaká informace unikátní pro konkrétní lokaci. Pro určování polohy mobilního zařízení lze využít jako otisk sílu signálů WiFi nebo Bluetooth [10, 6, 14] v daném místě. Otiskem tak bude vektor intenzit signálů v daném bodě.

Mobilní zařízení následně naměří intezitu signálů v aktuálním bodě a najde v databázi nejpodonější otisk, který bude svázán s konkrétní polohou.

Zde narážíme na několik problémů. Získané otisky svázané s pozicí jsou tzv. *učení s učitelem*, které musí být provedeno ručně. Tato data se musí sbírat znovu pro každou budovu a při zásadní změně interiéru bude potřeba otisky aktualizovat. Dalším úskalím je velikost takové databáze, která bude pomocí otisků hustě pokrývat mapovanou oblast. Velká databáze by nemohla být součástí aplikace. Otisky by se tedy musely získávat přes internet ze serveru, který by realizoval nalezení nejpodobnějšího otisku z celé databáze. To ovšem snižuje flexibilu řešení a zvyšuje náklady.

Dalším faktorem je počet vysílačů. Z popisuje metody je patrné, že čím více signálů bude v místě dostupných, tím lépe bude metoda fungovat. V případě WiFi je ale většinou snaha opačná – co nejmenším počtem vysílačů, pokrýt velký prostor.

Lokace založená pouze na metodě otisků by tedy nebyla dostatečně flexibilní, protože by vyžadovala sbírání otisků a následně internetově dostupnou službu, která by vyhodnocovala otisky zaslané od uživatelů.

Možný přínos vidím v nasazení otisků pro zpřesnění trilateračních metod. Při implementaci řešení by byly nasbírány otisky z míst jako jsou vchody do místností nebo rohy chodem. Tato malá databáze by pak mohla být využita jako další lokační faktor v trilateračních metodách.

Výše zmíněná metoda echolokačních otisků se ukázala využitelnou [13] na úrovni rozpoznávání místností. Stejnou službu ovšem poskytnou i levné Bluetooth majáky a to bez potřeby předem sbírat zvukové vzorky. Tato metoda je zde uvedena spíše pro přehled.

3.4 Lokace sledováním pohybu (dead reckoning)

Tato metoda vychází z myšlenky pohybu vzhledem k výchozímu bodu [11]. Pokud známe startovací pozici, stačí pouze sledovat následný pohyb zařízení vůči startovacímu bodu. K tomu se využívá senzoru akcelerometru. Ten zachycuje zrychlení ve všech třech osách mobilního zařízení. Znalost zrychlení nebo rychlosti však nestačí. Je nutné ještě zjistit směr pohybu. Tedy jak rychle a kterým směrem se zařízení pohybuje.

K určování směru se využívá gyroskop, který zachycuje úhlovou rychlost ve smyslu otáčení kolem všech tří os. K určení směru lze využít ještě magnetometr, který měří intezitu a směr magnetického pole v konkrétním místě.

3.4.1 Akcelerometr

Pro sledování zrychlení v nějakém směru lze využít tělesa na pružině. Příkladem může být závaží svisle zavěšené na pružině. Pokud dojde k pohybu aparatury ve svislém směru, délka pružiny se změní. Z této změny lze vypočítat i změnu zrychlení. Podobně by to fungovalo i ve zbylých dvou směrech. Takže závaží zavěšené na pružinách ve vše třech na sobě kolmých osách by byl 3D akcelerometr.

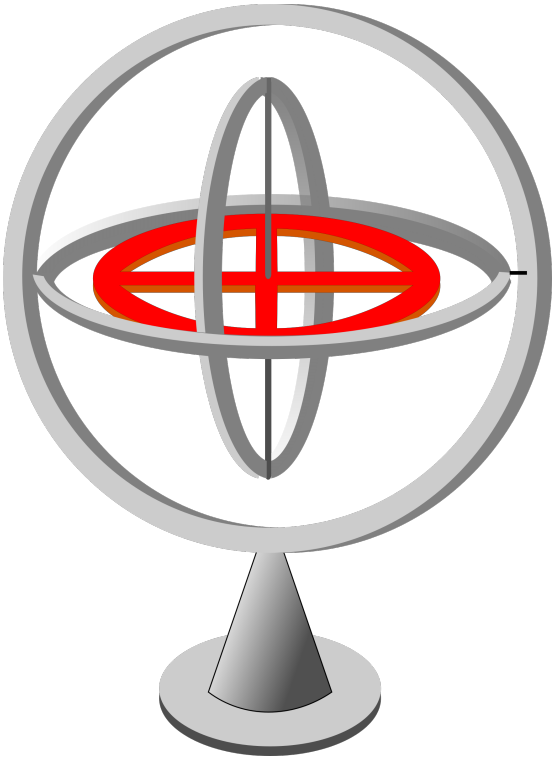
3.4.2 Gyroskop

Gyroskop je zařízení využívající gyroskopického efektu – pokud je hmotnost rotujícího tělesa soustředěna po jeho obvodu, má tendenci zachovat svou osu rotace. Na tomto principu funuje i obyčejná káča, která se při vysoké rotaci i v případě vychýlení navrací opět do kolmé polohy. Tento efekt je silnější s rostoucím poloměrem rotoru a rostoucí rychlostí.

Rotor gyroskopu je zasazen do dvou kruhových na sebe kolmých rámů (obrázek 3.1). V případě otočení zařízení se rámy pohybují se zařízením, zatímco rotor si zachovává svou polohu díky rotaci. Tímto vzniká odchylka mezi rotorem a rámy, ze které lze vypočíst změnu úhlové rychlosti.

Takto popsané senzory by byly příliš velké pro nasazení v mobilních zařízeních. Zde se uplatňují tzv. MEMS (microelectromechanical system) senzory využívající k měření piezoelektrického jevu (schopnost krystalu při deformaci generovat napětí). V akcelerometru zastupuje deformovaný krystal pružinu. MEMS gyroskop využívá místo rotace vibrací, protože vibrující těleso má tendenci zachovávat rovinu vibrací (normála roviny zde nahrazuje osu rotace). Vibrující těleso je z piezoelektrického materiálu a při rotaci na něj působí Coriolisova síla, čímž dochází k deformaci a měřitelné změně napětí.

Jako startovací pozici lze využít ruční určení polohy na mapě nebo třeba načtení QR kódu ze stěny. Bohužel, jak studie ukazují [11], tak lokace založená pouze na těchto senzorech vykazuje velké odchylky (hodnoty ze senzorů zařízení, které v klidu leží na stole nejsou stálé). Navíc tyto senzory mnoho zařízení nemá. U mobilních telefonů či tabletů lze za základ považovat akcelerometr (slouží ke zjištění



Obrázek 3.1: Rotační gyroskop.

pozice zařízení vzhledem k zemi a následné rotaci obrazovky), ovšem magnetometr a gyroskop již standardní vybavení nejsou.

Navigace pomocí sledování pohybu se v praxi využívá např. u hasičských sborů [32]. Součástí hasičské výstroje jsou tzv. *IMU (Inertial measurement unit)* jednotky. Jedná se o malá zařízení, která obsahují kvalitnější akcelerometr, gyroskop a magnetometr. S jejich pomocí lze sledovat pohyb hasičů v troskách.

Senzory nejsou běžným vybavením mobilních zařízení. Kvůli tomu by bylo nutné pořizovat IMU jednotky, které by byly spárované s telefonem nebo tabletem. Takové jednotky jsou cenově nákladné (aktuálně startují na cca 200 USD za kus [33, 34]). Lze nalézt i levnější alternativy [35], které ovšem poskytují pouze surová data ze senzorů. Díky odchylkám senzorů je nutné využívat další metody ke zpřesňování lokace a pravidelně synchronizovat přesnou polohu.

Tato metoda je vhodná pro případy, kdy není možné využít žádného hardwaru v okolí jako je WiFi, Bluetooth atd. Vhodným nasazením je umístění pohybových senzorů do hasičské výstroje. Při požárním zásahu nelze předpokládat na místě žádný pomocný hardware. Ovšem možnost znát polohu každého hasiče a případně ji mezi zasahujícími sdílet, může být značnou výhodou.

3.5 Magnetická navigace

Využívá faktu [31], že stavba díky kovům a elektrickému vedení deformuje přirozené magnetické pole Země. Spojitý magnetický otisk budovy (nebo velké množství nespojitých otisků) a počáteční poloha umožňují určování polohy sledováním

změn vektoru magnetického pole.

Tato metoda v podstatě spadá do oblasti fingerprintingu. Je ovšem zvláštní tím, že otisk je zde pouze trojrozměrný vektor magnetického pole. Tento vektor tak může být na mnoha místech stejný. Proto se využívá startovací polohy a porovnávají se vždy otisky z nejbližšího okolí poslední známe polohy.

Magnetický popis prostředí budovy navíc vyžaduje sběr dat, tedy učení s učitelem. Magnetometr navíc nemusí být součástí mobilního zařízení.

Nicméně již existují komerční systémy postavené na této metodě (www.indooratlas.com)
.

3.6 Resume

Bylo popsáno několik metod a pro přehlednost jsou výsledky shrnuty v tabulce 3.2.

Tabulka 3.2: Přehled lokačních metod

Metoda	Klady	Zápory
Družicové lokační systémy	• bez nutnosti instalace nového hardwaru pro zákazníka ani provozovatele • aplikace již existují	• nedostatečná přesnost pro indoor lokaci • nedostupnost signálu uvnitř budov
GSM trilaterace	• bez nutnosti instalace nového hardwaru pro zákazníka ani provozovatele	• nedostatečná přesnost pro indoor lokaci
WiFi trilaterace	• nevyžaduje speciální hardware pro uživatele ani provozovatele • známá problematika – dostupnost informací	• většina metod vyžaduje více AP • AP vyžadují stále napájení • z důvodu energetických nároků nevhodné např. pro jeskyně nebo památky • kolísání intenzity singálu a multipath efekt
		pokračování na další straně

Tabulka 3.2 – pokračování tabulky

Metoda	Klady	Zápory
Bluetooth Low Energy trilaterace	• uživatelé nepotřebují žádný speciální hardware • BLE majáky již existují • BLE majáky levnější než AP • již zkoumaná problematika – dostupnost informací • lze jednoduše sestrojit vlastní BLE majáky, které jsou ještě levnější • měsíce provozu na jedinou materii, malé rozměry – použitelné pro památky nebo jeskyně • BLE standard byl vytvářen i s ohledem na možnosti lokace zařízení • možnost lokační signál využít i k rozesílání dalších informací jako např. URL, kde lze nalézt informace o památce	• kolísání signálu a multipath efekt • BLE infrastruktura většinou není přítomna – nutná počáteční investice
Zvuková trilaterace	• bez nutnosti instalace nového hardwaru pru uživatele • levný hardware pro provozovatele, který ovšem není standardizovaný – nutné vyrobit	• méně zkoumaný problém než např. BLE trilaterace • více rušený signál než WiFi a BLE – krom multipath ruší i běžné zvuky

pokračování na další straně

Tabulka 3.2 – pokračování tabulky

Metoda	Klady	Zápory
Rozpoznávání pohybu	• již známá problematika – dostupnost informací • potřebný hardware standardizovaný • funguje pouze na straně uživatele – bez nutnosti napájení • nulové náklady pro provozovatele budovy nebo památky • není ovlivněno změnou nábytku v místnosti	• v zařízeních nejsou standardně všechny potřebné senzory (často jen akcelerometr) • senzory zašuměné – nutné data čistit • IMU jednotky nákladné
Fingerprinting	• schopnost pro otisk použít několik druhů signálu BLE, WiFi, zvuk atd. • již zkoumaná problematika – dostupnost informací • možnost využití již existující infrastruktury (WiFi, zvuk, magnetické pole)	• delší doba nasazení – je nutné učení s učitelem • kvůli práci s databází otisků by pro některé metody bylo nutné mít aplikační server, který by komunikoval se zařízeními
Magnetická lokace	• nevyžaduje ze strany provozovatele žádný hardware • rozvíjející se odvětví – dostupnost informací	• všechna zařízení nemusejí mít potřebný magnetometr • nutnost nasbírat magnetické otisky budovy • existující řešení často využívají aplikační servery, kde je poloha zařízení vyhodnocována – potřeba hardware nebo pronájem služby
Obrazová		

pokračování na další straně

Tabulka 3.2 – pokračování tabulky

Metoda	Klady	Zápory
lokace	• uživateli stačí pouze fotoaparát v mobilním zařízení • při obrazovém rozpoznávání pohybu není nutný žádný hardware ani na straně provozovatele objektu	• pro obrazové rozpoznávání okolí nutné učení s učitelem • závislé na světelných podmínkách – nevhodné pro místa s nedostatkem světla • pro rozpoznávání pohybu závislé i na vzhledu podlahy

Na základě této analýzy jsem se rozhodl pro výslednou aplikaci využít technologii BLE majáků, která umožňuje využití trilaterace, fingerprintingu, primitivní proximity lokace a navíc pomocí signálu rozesílat další informace jako URL, kde lze najít další informace o obchodu, exponátu atd.

Dalším kladem je nízká cena potřebného hardwaru, který lze díky nízké energetické náročnosti provozovat i v místech jako jsou např. jeskyně nebo chráněné památky bez ohledu na světelné podmínky. Pro tuto volbu hovoří i množství již existujících studií a aplikací, kterých lze využít jako vodítek při návrhu výsledné aplikace.

Díky malým rozměrům a mobilitě lze navíc, v případě BLE trilaterace, další majáky postupně doplňovat.

4. Bluetooth lokace

Do této skupiny patří hned několik řešení s využitím Bluetooth. Nejdříve je však nutné přiblížit Bluetooth technologii samotnou.

4.1 Bluetooth

Bluetooth [30] je standard bezdrátového přenosu mezi zařízeními. Aby mohla zařízení komunikovat musí se nejdříve spárovat. Tím se automaticky vytvoří tzv. *piconet*, což je místní bezdrátová síť. Bluetooth pracuje v nelicencovaném pásmu o frekvenci 2.4GHz.

Bluetooth je technologie nahrazující kabely pro komunikaci zařízení, které jsou u sebe blízko. Proto mají zařízení dosah typicky 10 metrů. Zařízení se dle rozsahu dělí do tří tříd (viz tabulka 4.1)

Zařízení Bluetooth o sobě poskytují jisté údaje:

- MAC adresu (48 bitů) - fyzickou adresu zařízení
- UUID (128 bitů) - identifikátor profilů poskytovaný zařízením nebo konkrétní aplikace, která je pro toto zařízení určena
- RSSI (Received Signal Strength Indication) - relativní síla přijatého signálu

Již z náhledu poskytnutých informací je patrné, že MAC adresa lze využít jako identifikace zařízení a údaj RSSI jako informaci pro určení vzdálenosti.

Tabulka 4.1: Třídy Bluetooth zařízení dle dosahu.

Třída	*Výkon (mW)*	*Dosah (m)*
Class 1	100	100
Class 2	2.5	50
Class 3	1	10

Bluetooth je zvláštní tím, že bylo navrženo pro komunikaci takřka libovolných zařízení mezi sebou. Podoba dat vyměňovaných mezi zařízeními i mezi jednotlivými vrstvami zásobníku Bluetooth se tedy může měnit v závislosti na tom, o jakou aplikace se jedná. Bluetooth tento fakt řeší zavedením tzv. *profilů*.

4.1.1 Bluetooth profily

Díky svojí obecnosti by plná implementace Bluetooth do všech zařízení byla náročná a zbytečná. Proto jsou zavedeny profily [30], které sdružují potřebné specifikace pro konkrétní služby. Při párování si zařízení mezi sebou vymění mj. i to, jaké profily podporují.

Jednotlivá zařízení tak obsahují pouze implementaci potřebnou pouze pro konkrétní profily a navíc Bluetooth *core* [29], což je společný standard pro všechna Bluetooth zařízení. Bluetooth *core* obsahuje protokoly

- RF - fyzická vrstva
- LC - linková vrstva (potvrzování, opětovné zasílání packetů)
- LM - link manager (vytváření, modifikace spojení zařízení)
- L2CAP - logická a adaptační vrstva
  - segmentace packetů
  - multiplexing packetů mezi více aplikací

Mezi známé profily patří:

A2DP slouží k bezdrátovému přenosu hudby ve stereo kvalitě.

AVRCP umožňuje nejen přenos hudby, ale také ovládání zařízení jako např. skok na další skladbu.

BPP profil umožňující bezdrátovou komunikaci s tiskárnami.

FTP profil umožňující přístup a práci se soubory na vzdáleném zařízení.

HFP je profil pro handsfree sady.

4.1.2 Verze Bluetooth

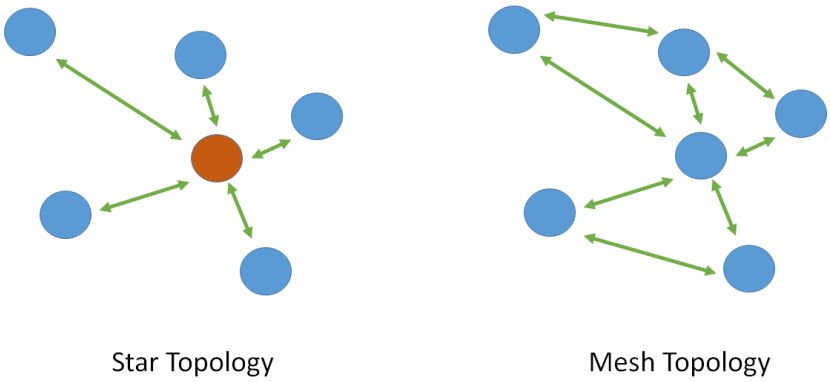
Dnes se již pracuje na specifikaci Bluetooth 5. V každé verzi [29] přišlo zlepšení v podobě zvýšení přenosové rychlosti nebo novém způsobu komunikace.

Bluetooth 2.1 vylepšilo spotřebu energie a využívá *EDR* (Enhanced Data Rate), což umožnilo zvýšení přenosové rychlosti z původních 1 Mbps na 2.1 Mbps. Další zlepšení byla ochrana proti *man-in-the-middle* útoku.

Bluetooth 3 umožnilo zrychlení přenosu až na teoreticky 24 Mbps, tím že pro větší objemy dat je schopné využít WiFi, které je přítomné na stejném zařízení.

Bluetooth 4 přineslo *BLE* (Bluetooth Low Energy) standard. Narozdíl od předchozích verzí není zpětně kompatibilní. *LE* standard byl vytvořen pro chytré hodinky, dálková ovládání, fitness monitory atp. Cílem je malá spotřeba pro zařízení, která vyžadují přenos malého objemu dat v pravidelných intervalech. Mimo tyto přenosy je Bluetooth ve spánkovém režimu a tím se šetří energie (na tužkové nebo knoflíkové baterie až měsíce provozu). Z popisu je jasné, že tento standard je nevhodným pro přenosy velkých objemů dat nebo streamování hudby, proto mají starší verze *3* a *2.1* stále své místo. Pro navigaci ovšem není nutné přenášet velké objemy dat, tím pádem je verze 4 vhodný kandidát. Dalším vylepšením je schopnost zařízení tvořit tzv. *mesh sítě*. Tedy komunikace mezi zařízeními již nutně nefunguje na principu centrálního uzlu (1 *master* a *slaves*), zařízení jsou schopna se spojit do sítě,

kde si mohou vyměňovat data i zařízení, která spolu nejsou přímo spojena – data pošlou k cíli prostřednictvím jiných uzlů (viz 4.1).

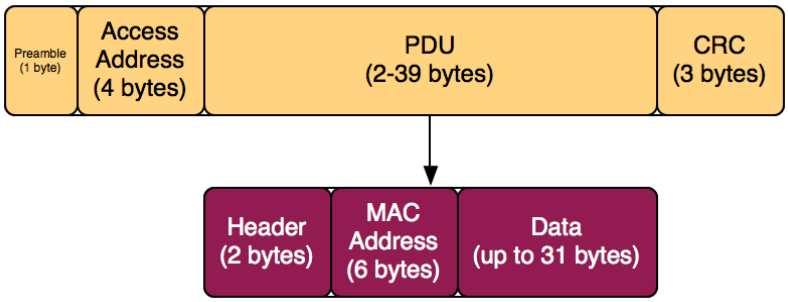


Obrázek 4.1: Topologie původního Bluetooth a mesh topologie umožněná BLE.

4.1.3 BLE Smart Beacons

Kromě spotřeby je velkým přínosem BLE možnost rozesílání broadcastových zpráv [37] – tedy pravidelných krátký zpráv, které může zachytit jakékoliv Bluetooth zařízení v dosahu, které podporuje BLE, bez nutnosti párování.

O rozesílání takových krátkých zpráv se starají tzv. *beacony* (majáky), které v pravidlených intervalech (typicky 100 ms) rozesílají krátké zprávy bez ohledu na to, zda někdo poslouchá. Formát packetů takových zpráv popisuje obrázek 4.2.

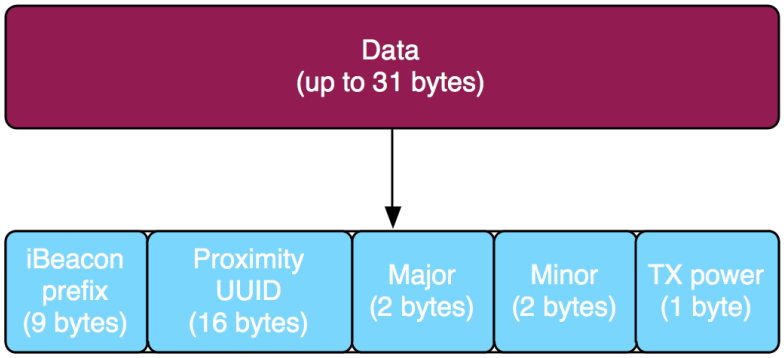


Obrázek 4.2: Struktura BLE packetu.

Jak je patrné, takto rozesílané zprávy jsou limitované na 31 bytů. Dodatečné informace jsou součástí aplikace, která čte signály od majáků, případně si je tato aplikace na základě přijatých údajů může stáhnout z internetu. Zprávy jsou tedy omezné a typicky využívané pro zasílání krátkých textových zpráv nebo číselných údajů. Pro uspořádání informací v *BLE* packetu se postupně vyvinuly formáty, z nichž nejznámější jsou *iBeacon*, *Eddystone* a *AltBeacon*. Existence těchto standardů umožňuje implementaci pokročilejších knihoven, které z nich vycházejí.

A iBeacon

Tento standard [37] byl vytvořen společnosti Apple v roce 2013. Jeho strukturu zachycuje obrázek 4.3. Jelikož standard popisuje pouze strukturu dat, je přístupný pro všechny platformy, tedy i pro Android.



Obrázek 4.3: Struktura iBeacon packetu.

Packet má 5 částí:

Prefix je fixní posloupnost 9 bytů. Podle ní lze poznat, zda se jedná o technologii *iBeacon*.

Proximity UUID je 16 bytů fungujících jako podpis majáků sloužící stejnému účelu. Typicky mobilní aplikace vyhledává majáky s konkrétním *UUID*. Majáky sloužící konkrétní aplikaci mají stejný podpis a tím se liší od ostatních. Proto na malém prostoru může koexistovat velké množství skupin majáků a aplikace se bude zajímat jen o ty správné.

Major jsou 2 byty označující podskupinu majáků v rámci stejného *UUID*.

Minor je dvojice bytů sloužící jako identifikace majáku v rámci podskupiny.

TX Power je jeden byt udávající sílu signálu ve vzdálenosti 1 metru od majáku.

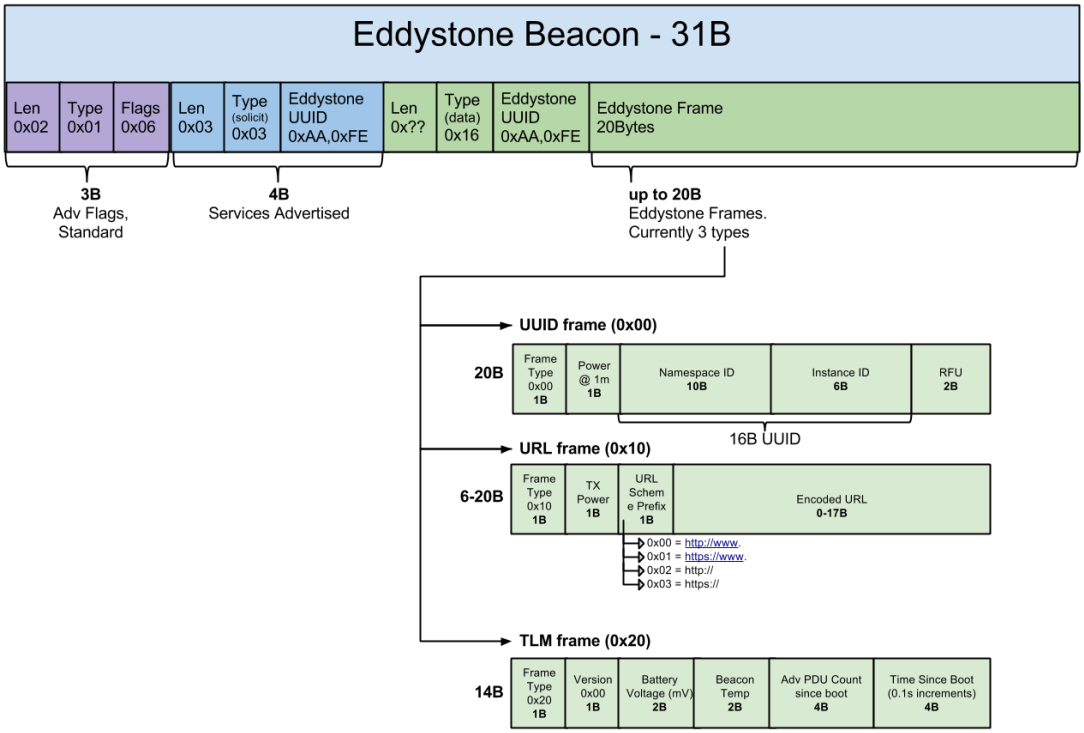
Pro snadnější pochopení uvedu příklad. Mějme síť kaváren. Pro tuto síť implementujeme aplikaci, která uživatele notifikuje o tom, že se nachází v blízkosti jedné z kaváren a zároveň prostřednictvím internetu stáhne dodatečné informace o otevírací době a aktuální nabídce.

- *Prefix* je fixní, v tento okamžik nás tedy nezajímá.
- *UUID* bude specifické pouze pro tuto aplikaci a všechny majáky na všech pobočkách jej budou mít shodné.
- *Major* bude identifikovat konkrétní kavárnu.
- *Minor* bude identifikovat jednotlivé majáky v rámci jedné kavárny, za předpokladu, že má např. více vchodů.

API využívající iBeacon standard je součástí vývojových nástrojů společnosti Apple pro mobilní technologie. Platforma Android nabízí tato API zatím jen prostřednitvím třetích stran.

B Eddystone

Je standard [36] vytvořený společností Google. Jeho strukturu popisuje obrázek 4.4. Stejně jako Apple zahrnuje *iBeacon* do svého SDK, je API pro *Eddystone* zahrnuto v Android SDK.



Obrázek 4.4: Struktura Eddystone packetu.

Jak je patrné na obrázku 4.4 *Eddystone* packet má několik podob v závislosti na typu. Ovšem všechny mají společný 11 bytů dlouhý prefix.

Eddystone UID je asi nejblíže *iBeacon* packetu.

Frame Type je 1 byte určující, typ packetu tedy *Eddystone UID*.

TX Power je 1 byte popisující sílu signálu ve vzdálenosti 0 metrů od majáku. Tím se liší od *iBeacon* formátu.

Namespace je 10 bytů sdružujících majáky do skupin.

Instance je 6 bytů identifikují majáky uvnitř skupiny.

RFU jsou 2 byty rezervované pro budoucí použití a zatím musejí být nastavené na 0.

Eddystone vyznává trochu jinou identifikaci majáků. Jsou zde pouze dvě úrovně hierarchie (*Namespace* a *Instance*) oproti *iBeacon* formátu, kde jsou úrovně 3. Na druhou stranu jsou tyto bloky větší. Eddystone tak dává více volnosti k vytváření hierarchie majáků.

Eddystone URL slouží k předávání URL adresy, kterou lze po příjetí klientským zařízením otevřít. Typicky to může být adresa s reklamním letákem nějakého řetězce obchodů nebo stránka dopravního podniku s odjezdy autobusů ze zastávky, které jste nejblíže.

Frame Type byte určující, že jde o *Eddystone URL* packet.

Tx Power je byte popisující sílu signálu majáku ve vzdálenosti 0 metrů.

URL Schema prefix slouží ke zkrácení URL – typický začátek adresy je zakódován číslem:

- 0x00 – http://www.
- 0x01 – https://www.
- 0x02 – http://
- 0x03 – https://

Encoded URL je až 17 bytů dlouhá komprimovaná URL.

Eddystone TLM slouží k přenosu servisních údajů o majáku jako je teplota, doba běhu, počet vyslaných zpráv či napětí baterie.

Frame Type je byte poskytující informaci o typu packetu.

Version verze telemetry packetu – počítá se s modifikací.

Battery Voltage jsou 2 byty udávající napětí baterie v mV.

Beacon Temp jsou 2 byty informující o teplotě majáku.

Adv PDU Count jsou 4 byty sloužící jako counter odeslaných informačních packetů.

Time Since Boot 4 byty informující o době běhu majáku. Counter přičítá po $0.1s$.

C AltBeacon

Tento standard [38] je dalším otevřeným standardem. Uvadím jej pouze pro úplnost, jelikož není tolik rozšířený.

Skupina stojící za tímto standardem ovšem vytvořila stejnojmennou knihovnu *AltBeacon*[39] pro platformu Android, která slouží k implementaci aplikací využívajících standardy *Eddystone, iBeacon* i *AltBeacon* (zdroj).

4.2 Síla signálu

V předchozích odstavcích byla často zmíněná síla nebo intenzita signálu. Pro další postup v práci je nutné popsat sílu signálu exaktně.

Síla signálu [41] se udává často v jednotkách *dbm*. Což je jednotka *decibel* vstažená na 1 miliwatt.

S jednotkou *db* se vetšina z nás setkala v souvislosti s měřením hlasitotsti zvuku. Tato jednotka je bezrozměrná stejně jak procenta a vyjadřuje poměr dvou veličin. Protože tento poměr může být často velmi malé číslo, využívá se ještě logaritmu.

$$L = 10 \log_{10} \frac{p_1^2}{p_0^2} = 20 \log_{10} \frac{p_1}{p_0}$$

Takto je popsána hlasitost zvuku. Pomocí druhé mocniny akustického tlaku (p_1) v poměru s nejnižší člověkem slyšitelnou hodnotou (p_0). Jako jmenovatel se využívá vždy referenční hodnota. Protože samotný logaritmus poměru vyjadřuje jednotku *bel*, násobí se výsledek ještě 10 - tím je získána jednotka *decibel*, která se využívá častěji.

Při dosazení výkonů (jednotka Watt) lze takto měřit intenzitu signálu – RSS (Received Signal Strength). Jako referenční hodnota se volí *miliwatt*.

$$RSS = 10 \log_{10} \frac{P_1}{1mW} [dBm]$$

4.2.1 Indikátor síly signálu

RSSI (received signal strength indicator) [40] je ukazatel síly signálu, který je k dispozici na straně příjemce skrze API. Bohužel jediným omezením je to, že se musí jednat o celé číslo o velikosti 1 byte a tím pádem různí výrobci hardwaru implementují tento ukazatel jiným způsobem. RSSI vlastně rozděluje přijatou sílu signálu na hladiny. Platí zde, že čím větší číslo hladiny, tím je signál silnější. Dalším negativem tohoto ukazatele je požadavek celého čísla, čímž dochází k jisté ztrátě přesnosti informace o síle signálu. Tento ukazatel původně nebyl zamýšlen pro tyto účely. Jeho primárním účelem bylo pouze rozlišit, zda se vysílač nachází v těsně blízkosti, poblíž nebo daleko.

BLE majáky vysílají mj. informaci o síle signálu v blízkosti 0, resp. 1 (T_x) metr. Pro výpočet vzdálenosti za pomocí RSSI je potřeba znát vzorec jakým výrobce hardwaru převádí přijatou sílu signálu na RSSI. Pokud tento vzorec známea známe i referenční sílu signálu T_x, je poslední proměnnou v rovnici právě vzdálenost zařízení.

Bohužel, postup výpočtu RSSI není vždy součástí dokumentace a to platí zejména pro levnější majáky nebo majáky, které si uživatel může sestrojit sám. V takovém případě je nutné provést několik kontrolních měření s jejichž pomocí lze provést aproximaci funkce popisující chování RSSI v závislosti na síle přijatého signálu.

4.3 Rušení BLE

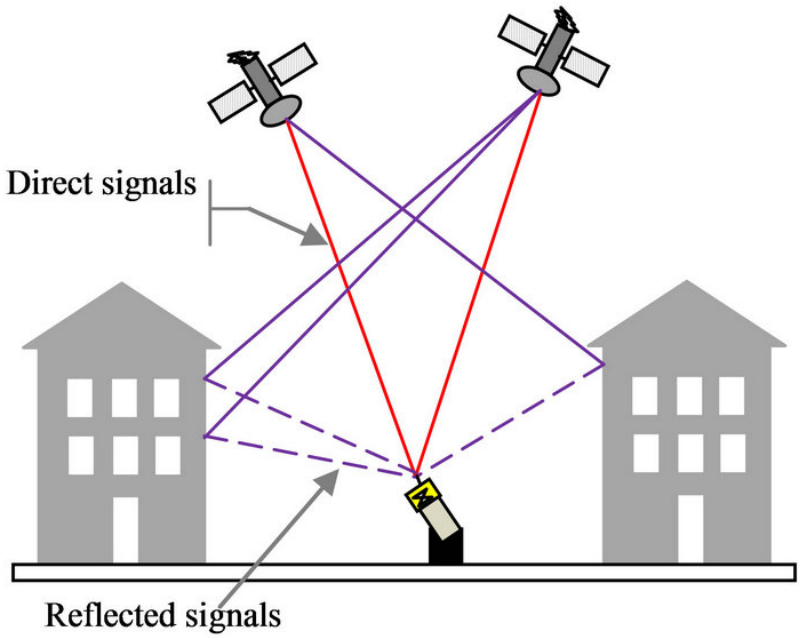
Bluetooth pracuje v pásmu 2.4GHz. Stejně jako u jiných signálů, zde dochází k rušení působením několika faktorů. Signál přirozeně slábne s rostoucí vzdáleností. Jeho pokles však způsobují i překážky v podobě nábytku, stěn i lidského těla. Naměřenou intenzitu signálu dokonce ovlivňuje rotace zařízení vzhledem k šíření vln. Posledním problematickým jevem, je tzv. *multipath* efekt.

Algoritmy využívající BLE musí tato rušení brát v úvahu a jejich dopady nějakým způsobem eliminovat.

4.3.1 Multipath efekt

Multipath efekt [42], jak název napovídá je jev, kdy se vyslaný signál dostanu k přijímači nejen přímou cestou, ale také pomocí odrazů od stěn. Problémem je, že přijímač v tu chvíli registruje signál přijatý v různých intanzitách a s časovým posunem. Vyvstává pak otázka, který je ten správný?

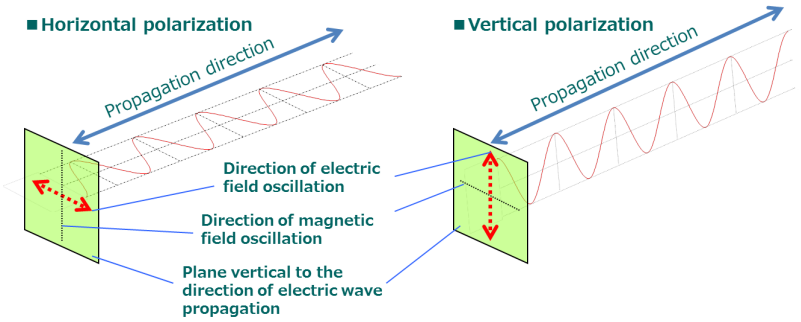
Nabízí se snadná odpověď – ten nejsilnější. Nejsilnější signál, by měl logicky projít k přijímači přímou cestou. Bohužel tomu tak není. Signál sice slábne při odrazech o překážky, ale takových odrazů se prostorem šíří několik. Ve chvíli přijmu signálu přijímačem se tyto odrazy mohou setkat a vzájemně se ovlivnit v závislosti na aktuální fázi vlny. Díky odrazům tak může vniknout signál, který se jeví silnější než signál, který dorazil k přijímači přímou cestou. Na obrázku 4.5 je nastíněn multipath efekt, který se netýká jen BLE, ale také WiFi či družicových signálů.



Obrázek 4.5: Multipath efekt.

4.3.2 Rotace přijímače

Signál BLE, WiFi nebo GSM je elektromagnetický. Jako takový je polarizován. Polarizace [42] signálu závisí na vysílající anténě. Pro jednoduchost uvažujme antény s vertikální nebo horizontální polarizací. Pokud anténa vysílá vertikálně polarizované vlnění (sinusoida vlnění se mění ve smyslu „nahoru a dolů“) je vhodné, aby přijímající anténa byla také vertikální. Tímto nedojde ke ztrátám díky polarizaci. Pokud ovšem bude přijímač polarizován horizontálně (bude očekávat sinusoidu měnící ze „zprava doleva“), bude intenzita přijatého signálu rovna 0. Tento problém bude patrnější z obrázku 4.6.



Obrázek 4.6: Popis šíření polarizovaného signálu.

Z toho je patrné, že rotace antény v mobilním zařízení vůči anténě vysílače má podstatný vliv na přijatou sílu signálu. Tento problém se řeší využitím několika, na sebe kolmých, vysílacích (resp. přijímacích) antén současně. Pokud jsou tedy vysílač a přijímač vzájemně pootočené o 90° je vše v pořádku. Při odchylce 45° již bude docházet k poloviční ztrátě.

Před samotnou implementací je nutné experimentálním měřením ověřit chování BLE signálu v budově. Jako testovací prostředí bylo zvoleno patro KIV.

5. Konstrukce BLE majáků

V předchozích kapitolách bylo analyzovány BLE vlastnosti pro využití v indoor lokaci, ale bylo čerpáno z externích zdrojů. Před implementací lokační aplikace je nutné chování BLE majáků experimentálně ověřit.

5.1 Konstrukce vlastních BLE majáků

Testů s komerčními majáky již proběhlo mnoho (zdroje). Pro tuto práci byla ovšem jedním z kritérií cena. Komerční majáky se pohybují kolem 25 USD za kus. Logickou otázkou bylo, zda je možné tuto cenu snížit zhotovením vlastních majáků.

Možné řešení se nabízí v oblasti DIY (Do It Yourself) projektů. Existují BLE knihovny pro platformu Arduino i Raspbery Pi, které by ve spojení s BLE modulem mohly fungovat jako BLE majáky. Nejlevnější Raspberry Pi Zero stojí 5 liber a od února roku 2017 je k dispozici Rasperry Pi Zero W, které je obohaceno o WiFi a Bluetooth 4.1. Samotná platforma Arduino Uno bez BLE periferie se pohybuje kolem 9 USD.

Jelikož maják je vlastně pouze vysílač (nepřijímá komunikaci od okolních zařízení), ukázalo se, že Arduino nebo Raspberry Pi není vůbec potřeba. BLE maják lze vyrobit i z obyčejného BLE modulu HM-10.

Modul HM-10 je BLE modul, který se využívá ve spojení např. s platformou Arduino pro získání BLE spojení. Ovšem modul sám o sobě poskytuje několik instrukcí pro nastavení a může fungovat jako BLE maják (standard iBeacon) samostatně. Cena modulu se pohybuje kolem 3 USD za kus. Nenalezl jsem levnější řešení a proto jsem se rozhodl využít tyto BLE majáky.

Náklady na konstrukci jednoho majáku popisuje tabulka (5.1). Zvýrazněné položky lze vyrobit např. pomocí 3D tisku součástek (viz CD `/3D_print`) a tím ušetřit. *FTDI* konektor se na na ceně modulu nepodílí, protože stačí jeden konektor na nastavené libovolného počtu modulů. Cena kabelů byla získána dělením, protože nejmenší balení bylo po 40 kusech. Ceny by samozřejmě, při objednání většího počtu kusů, ještě klesly.

5.1.1 Nastavení HM-10 modulu jako BLE majáku

Pro komunikaci s modulem je potřeba *FTDI* modul (převodník mezi USB a serial). Spojením(schema) FTDI a HM-10 za pomoci kabelů získáme možnost zasílat z PC instrukce HM-10 modulu a zároveň jej napájet.

Pro komunikaci s modulem je potřebné nainstalované Arduino IDE, které s sebou nese potřebné drivery pro komunikaci skrze FTDI.

Tabulka 5.1: Náklady na konstrukci jednoho majáku

Komponenta	*Cena*	*Zdroj*
BLE Modul HM-10	78.50 Kč (3.12 USD)	Aliexpress/Ebay
AAA battery holder	12.60 Kč (0.50 USD)	Aliexpress
2x kabely s koncovkou samec (pro snadné zapojení modulu)	1.8 Kč (0.07 USD)	Aliexpress/Ebay
3D printed case	12 Kč	viz CD `/3D_print`
3D printed battery holder	0 Kč	viz CD `/3D_print`
FTDI konektor	18.70 Kč (0.74 USD)	Aliexpress/Ebay

A Serial komunikace s HM-10 modulem

Pokud je HM-10 spojený s PC prostřednictvím FTDI, stačí spustit Arduino IDE. Následně nastavit v sekci *Tools Serial port*. Na OS GNU/Linux se využívá /dev/ttyUSB0, pro Windows to bude jeden z nabídnutých COMx portů. Následně lze již otevřít *Serial monitor*, což je terminál přímo pro komunikaci s HM-10 modulem. Okno je rozděleno na příkazový řádek a okno, kde se zobrazuje výstup z modulu.

Pro testování nastavení majáku jsem využil zařízení z tabulky 5.2. Modul se nyní nachází v běžném režimu a díky napájení, jej lze nalézt vidět jako dostupný v jednotlivých mobilních zařízeních. Párování bude ovšem modulem aktivně odmítnuto.

Tabulka 5.2: Zařízení a aplikace využité pro testování BLE majáků.

Zařízení	*Verze OS*
Elephone P8000	Android 6.0
Lenovo Tab2 A7-30HC	Android 4.4.2
Apple iPhone 6s Plus	iOS 10

Moduly HM-10 se vyskytují v několika provedeních, které se od sebe lehce liší (počtem nebo podobou instrukcí, ukončovacím znakem instrukce, možností updatovat firmware modulu). Všechna provedení mají ovšem řadu instrukcí společnou. Jednou z nich je instrukce *AT* na kterou by měl modul odpovědět *OK*.

Před odesláním první instrukce je ještě nastavit modulační rychlost (počet změn stavu za vteřinu) spojení – HM-10 vyžadují 9600 baud. Modulační rychlost lze nastavit v pravém dolním rohu *Serial monitoru*.

Prvním problémem může být znak ukončující instrukci. Pokud modul neodpovídá na tuto instrukci je pravděpodobně nastaven špatný ukončovací znak. Ten lze nastavit v *Serial monitoru* ve spodní liště. V mém případě to bylo spojení *Both NL & CR*. Doporučuji instrukci *AT* zaslat alespoň 2x, protože modul může být při delší neaktivitě uspán a první instrukce ho pouze probudí.

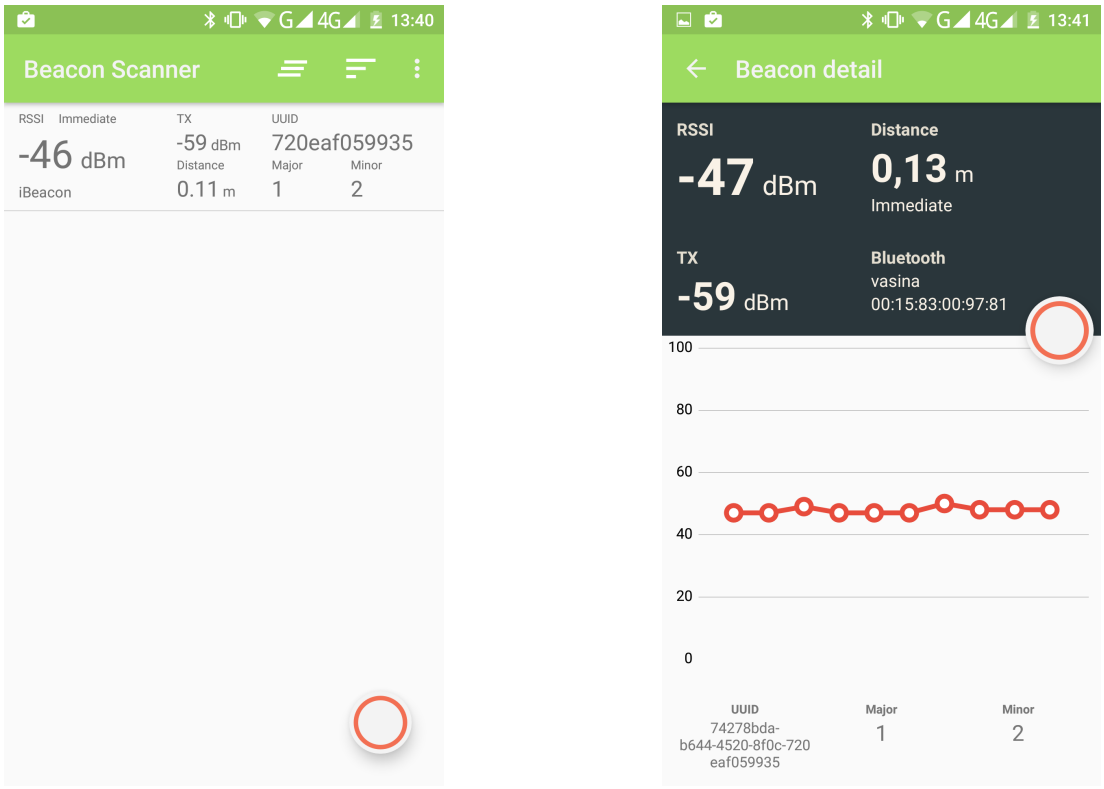
Další instrukcí je *AT+HELP*, na kterou modul odpoví seznamem všech podporovaných instrukcí a dodatečné informace o výrobci. Odpověď může vypadat např.

takto:

```
**********************************************************************
* Command             Description
* ------------------------------------------------------------------
* AT                  Check if the command terminal work normally
* AT+RESET            Software reboot
* AT+VERSION          Get firmware, bluetooth, HCI and LMP version
* AT+HELP             List all the commands
* AT+NAME             Get/Set local device name
* AT+PIN              Get/Set pin code for pairing
* AT+PASS             Get/Set pin code for pairing
* AT+BAUD             Get/Set baud rate
* AT+LADDR            Get local bluetooth address
* AT+ADDR             Get local bluetooth address
* AT+DEFAULT          Restore factory default
* AT+RENEW            Restore factory default
* AT+STATE            Get current state
* AT+PWRM             Get/Set power on mode(low power)
* AT+POWE             Get/Set RF transmit power
* AT+SLEEP            Sleep mode
* AT+ROLE             Get/Set current role.
* AT+PARI             Get/Set UART parity bit.
* AT+STOP             Get/Set UART stop bit.
* AT+START            System start working.
* AT+IMME             System wait for command when power on.
* AT+IBEA             Switch iBeacon mode.
* AT+IBE0             Set iBeacon UUID 0.
* AT+IBE1             Set iBeacon UUID 1.
* AT+IBE2             Set iBeacon UUID 2.
* AT+IBE3             Set iBeacon UUID 3.
* AT+MARJ             Set iBeacon MARJ .
* AT+MINO             Set iBeacon MINO .
* AT+MEA              Set iBeacon MEA .
* AT+NOTI             Notify connection event .
* AT+UUID             Get/Set system SERVER_UUID .
* AT+CHAR             Get/Set system CHAR_UUID .
* ------------------------------------------------------------------
* Note: (M) = The command support slave mode only.
* For more information, please visit http://www.bolutek.com
* Copyright@2013 www.bolutek.com. All rights reserved.
**********************************************************************
```

Další postup nastavení popisuje následující tabulka (5.3).

Zvýrazněné řádky obsahují instrukce, které umožňují úsporu energie, ale nejsou obsaženy ve všech variantách modulu *HM-10*. Přesněji modul *HM-10* je často zaměňován s jeho klonem *CC41-A* od firmy Bolutek, který je okem velmi špatně rozeznatelný, a tak uživatel zjistí odlišnosti až při zapojení modulu a zjištění chybějících instrukcí.



(a) Seznam majáků v dosahu.

(b) Detail majáku.

Obrázek 5.1: Náhledy iBeacon aplikace.

Na tento problém jsem narazil i já. Jelikož modul ovšem poskytoval požadovanou funkčnost iBeacon majáku a postrádal pouze možnost snížit energetické nároky, nebyl důvod jej měnit. Navíc rychlé dodání nových modulů by znamenalo je objednat z Evropy nebo Čech, kde se jejich cena pohybuje kolem 191 Kč za kus místo 2 USD při objednání v Číně.

Pokud máme modul stále připojený k PC, je tím pádem napájen. Dalším krokem je instalace aplikace pro monitorování okolních BLE majáků na mobilní zařízení. Pro platformu Android jsou to například aplikace *iBeacon* a *Locate*, které zvládají sledovat majáky *Eddystone* i *iBeacon*. Aplikace *Locate* dokonce umožňuje sledování majáků *AltBeacon* nebo z mobilního zařízení maják vytvořit. Zmíněné aplikace zároveň vypočítavají vzdálenost majáku dle RSSI a je tedy možné i kontrolovat rychlost odezvy a vypočtenou vzdálenost od majáku.

B Ověření funkčnosti majáku

Po nastavení majáku je ještě nutné ověřit, zda se maják chová jak je očekáváno. Za pomoci jedné ze zmíněných aplikací (*iBeacon, Locate*) lze scanovat okolí a hledat viditelné majáky. Já osobně jsem využil aplikaci *iBeacon*, protože poskytuje o majáku více informací (např. jméno majáku) a možnost sledovat v grafu vývoj RSSI sledovaného majáku v čase (viz obrázky 5.1a, 5.1b).

Pokud aplikace nevidí váš maják, s největší pravděpodobností nemáte na zaří-

zení povolené lokační služby. Mnoho uživatelů je má zakázané z důvodu šetření baterie nebo nechce, aby měly aplikace přístup k jeho poloze. Já se s tímto problémem setkal u svého tabletu. Lokační služby v telefonu nezahrnují pouze GPS a určování polohy pomocí dostupných WiFi. Toto nastavení umožňuje aplikacím využívat lokační API , které platforma na mobilním zařízení poskytuje. Pro platformy *Android* i *iOS* je, při vypnutí lokačních služeb, přístup do lokačního API odepřen. Technologie BLE majáků spadá právě do lokačního API. Pokud tedy není povolené, v aplikacích nebude maják vidět. Maják bude viditelný pouze v seznamu dostupných Bluetooth zařízení, ale nebude možné se s ním spárovat.

5.1.2 Časté problémy s moduly HM-10

Využití modulů HM-10 jako majáků může zkomplikovat několik drobností, které jsou však řešitelné.

Správný serial port

Při komunikaci s modulem v *Arduino IDE* je nutné nastavit správný serial port. Na OS GNU/Linux to bývá typicky */dev/ttyUSB0* a problém není. Na OS Windows existuje často více portů, které jsou označeny *COMx*, kde x je číslo.

Přenosová rychlost

Moduly vyžadují přenosovou rychlost 9600 baud, která je nastavitelná v pravém spodním rohu *Serial terminálu*.

Ukončovací znak instrukcí

Instrukce zasílané modulu musejí být ukončené speciálním znakem (často konec řádky). Tento znak může být však pro verze modulu jiný. Stejně jako přenosovou rychlost jej lze nastavit v *Serial terminálu*.

Velikost písmen v instrukcích

Základní podoba instrukcí pro moduly *HM-10* je tzv. upper case, tedy velká písmena. Některé verze mohou podporovat i jiné varianty. Pro jistotu tedy doporučuji používat rovnou instrukce v podobe *AT* místo *at*.

Klony modulu HM-10

Na trhu se objevují klony modulu *HM-10* a v internetovém obchodě je od originálu nerozeznáte. Liší se zejména tím, že postrádají některé instrukce vhodné pro úsporu energie majáku. Tyto klony lze však také využít jako majáky, pouze budou mít větší spotřebu.

Zakázané lokační služby

Na mobilních platformách jako je *Andoid* či *iOS* lze zakázat lokační služby. Tím se ovšem odepře aplikacím přístup k lokačním službám jejichž součástí je API pro BLE majáky. Díky tomu nebude pro aplikace maják viditelný. Aplikace pracující s majáky tedy vyžadují tyto služby povolené.

Napájení modulu

Modul vyžaduje napájení proudem s napětím v rozsahu zpravidla $3.3V$ až $6V$. Pro nižší hodnoty bude omezen výkon nebo modul nebude vůbec reagovat. Vyšší hodnoty mohou způsobit trvalé poškození modulu. Napětí $6V$ lze snadno docílit sériovým zapojením 4 kusů AA nebo AAA baterií.

Režim spánku

Pokud byla použita instrukce *AT-PWRM0*, je aktivní uspávání modulu při neaktivitě. To znamená, že modul je viditelný jako maják, ale není dostupný pro seriovou komunikaci. Je nutné ho probrat zasláním 80 náhodných znaků.

Pokud byl modul odpojen od napájení, může se stát, že při opětovném připojení vůbec nereaguje. V takovém případě stačí jej znovu odpojit a zapojit k napájení. Pokud tento krok nepomůže je modul pravděpodobně poškozen nebo jsou baterie vybité.

5.1.3 Další dostupné moduly

Pro výběr *HM-10* hovořilo několik skutečností:

- nízká cena
- dobře známý hardware (snadná dostupnost informací a firmwaru)
- již v základu podporuje iBeacon standard

Existují ovšem další způsoby, jak vytvořit BLE maják ať *iBeacon* či *Eddystone*.

- modul *RN4020*, který lze po upgradu firmwaru přeprogramovat pro vysílání *Eddystone URL* packetů
- *MCU CC2640* od Texas Instruments, kteří sami poskytují tutoriál, pro implementaci *Eddystone* majáku
- využití již hotových levných majáků ze sítí Aliexpress či Ebay (často využívajících modul *Nordic NRF51822*)
- spojení BLE vysílače s platformou Raspberry Pi nebo Arduino (dražší řešení nabízející plnou kontrolu na implementací majáku)

5.2 Experimentální měření

Než bylo možné přistoupit k samotné problematice lokace zařízení, bylo nutné otestovat chování majáků a BLE signálu v prostoru. Důležitý byl zejména vliv prostorových překážek a tvaru místnosti na vypočtenou vzdálenost od majáku.

5.2.1 Testovací aplikace

Pro účely testování byla implementována testovací aplikace, která byla nakonec rozvinuta v koncovou aplikaci. Pro pouhé monitorování chování signálu v prostředí lze aplikaci přepnout do vývojářského režimu (viz C), a zakázat všechny lokalizační algoritmy. Na plánu se pak budou zobrazovat pouze majáky s jejich dosahem.

Aplikace z QR kódu získá informace o majácích (pozice, major a minor označení). Majáky následně zanese do plánku na odpovídající souřadnice a zároveň do seznamu s detajly jednotlivých majáků.

Ve chvíli, kdy má aplikace potřebné informace o majácích, začne poslouchat BLE broadcast z okolí. Pokud zachytí zprávu od známého majáku, jsou informace o něm aktualizovány a je vypočtena vzdálenost od majáku.

Vzdálenost je ve stejný okamžik zanesena i do plánku, kde je zobrazena jako kruh, jehož střed je na pozici majáku. Vzdálenost od majáku je použita jako poloměr zmíněného kruhu.

V této fázi bylo zároveň testována funkčnost knihovem třetích stran:

Altbeacon

Známá knihovna pro práci s BLE majáky (standard iBeacon, Eddystone i Altbeacon). Zjednodušuje čtení dat z majáků a obsahuje již algoritmy pro výpočet vzdálenosti majáku na základě síly signálu (RSSI a TX).

ZXing

Oblíbená knihovna [43] pro práci s 1D a 2D čárovými kódy. Implementovaná v Javě (poskytuje integraci pro Android) a portovaná do dalších jazyků.

Subsampling Scale Image View

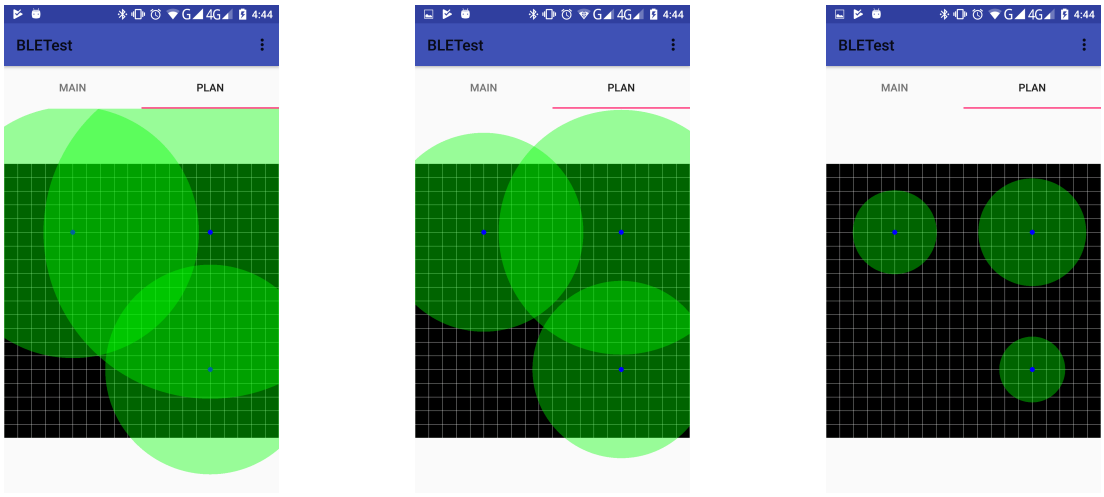
Jak název napovídá jedná se o prvek uživatelského rozhraní pro Android. Komponenta [44] byla původně vytvořena pro prohlížení a manipulaci s obrázky (změna velikosti a posouvání pomocí dotykových gest). Díky již implementovanému dotykovému ovládání a přepočtu souřadnic mezi obrazovkou a obrázkem, posloužila komponenta jako dobrý základ pro implementaci indoor mapového náhledu.

5.2.2 Vyhodnocení experimentu

Testy byly prováděny v prostorách KIV a budově Stafinu. Testovány byly jak prostory bez překážek, tak úzké chodby. Testovacích prostorách se objevovaly i překážky v podobě schodišť či skleněných stěn.

A Kolísání RSSI

První pokusy byly provedeny na podložce bez překážek. Výsledkem bylo sledování chování RSSI v klidném stavu. Mobilní zařízení bylo položeno v různých



(a) Ideální případ průniku.

(b) Průnik pouze dvojic majáků.

(c) Stav bez průsečíků.

Obrázek 5.2: Náhledy průniku signálů s testovací aplikace.

vzdálenostech (1, 2 a 3 metry) od majáku. V místnosti jsem zůstal pouze já, aby se odstínil vliv pohybujících se osob na kolísání signálu.

Za těchto podmínek RSSI kolísalo v rozmezí ±3 dbm. Toto kolísání ve spojení se znalostí výpočtu RSSI potvrzuje, že s rostoucí vzdáleností od majáku, roste chyba v určení vzdálenosti. V bezprostřední blízkosti majáku je odchylka 3 dbm v řádu centimetrů, ve vzdálenosti pěti metrů je již odchylka v řádu jednotek metrů tedy přes 20 %.

B Multipath efekt

Multipath efekt (viz sekce 4.3.1) byl patrný nejen na kolísání signálu, ale zejména v úzkých chodbách (2 metry široké), kde bylo na ploše 2x20 metrů nasazeno 5 majáků. Stávalo se, že maják vzdálený 20 metrů vykazoval nárazově vzdálenost 4 metry. Tedy signál cestující přímou cestou se např. sečetl s odraženým a jevil se silnější.

Tyto případy se objevovaly nárazově (jednalo se tedy o šum v informaci) a pokud jsem se s mobilním zařízením pohyboval, nebyly příliš patrné.

C Nedostatek průsečíků

Posledním problematický jev nastával v otevřených prostorech, kde na velké vzdálenosti bylo málo majáků. Tedy pokud se např. na chodbu 10x12 metrů umístily 4 majáky pouze do rohů. Díky velké vzdálenosti byla velká i chyba. Tím nastávala situace, kdy mobilní zařízení zachytilo všechny signály, ale jejich kruhy se na plánku neprotnuly. To je z logiky veci nesmysl, protože všechny majáky byly v dosahu. Náhled testování zachycují obrázky 5.2a, 5.2b, 5.2c.

Tento jev napověděl mnoho o dalším průběhu práce:

Kalibrace majáků

Pro výpočet vzdálenosti jsou klíčové hodnoty TX a RSSI. Hodnota TX je vysílaná majákem a představuje referenční RSSI naměřenou ve vzdálenosti 1 metr (pro *Eddystone* 0 metrů) od majáku. Pokud je tedy tato informace špatně nastavena, bude výsledná vzdálenost chybná. Majáky mají hodnoty TX pevně zvolenou nebo nastavitelnou pomocí instrukcí. V případě majáků použitých pro tuto práci jsou instrukce popsány v tabulce 5.3.

Změřit hodnotu RSSI v tak malé vzdálenosti je díky kolísání obtížné. Doporučuji pro kalibraci použínásledující postup:

1. Nastavit výrobcem doporučenou hodnotu TX, případně ponechat přednastavenou hodnotu.
2. Pomocí aplikace na mobilním zařízení zjistit vypočtenou vzdálenost při reálné vzdálenosti 1 metr a 3 metry. Díky kolísání intenzity signálu je potřeba v pozici chvíli setrvat a najít střed naměřených hodnot.
3. Pokud je vypočtená vzdálenost vyšší než reálná, je hodnota TX příliš vysoká a je třeba ji snížit. V opačném případě je nutné hodnotu zvýšit. Kalibrace pokračuje opět krokem 2.

Optimalizační algoritmy

V ideálním případě trilaterace se protnou kružnice v jediném bodě. Vlivem chyb se ovšem počítá s průnikem několika kruhů, přičemž hledaná pozice leží v oblasti průniku. Díky jevu, kdy se některé kruhy neprotnou nebo se dokonce neprotnou žádné kruhy, vyvstává problém. Algoritmy, které budou využívat průniky či průsečíky v těchto případech nejsou přímo použitelné.

Vhodnou alternativou by mohli být optimalizační algoritmy jako např. metoda nelineárních nejmenších čtverců nebo simplexová metoda. Tyto metody počítají s chybou naměřených hodnot a iterativně hledají řešení, pro které bude chyba nejmenší.

Vzdálenost majáku jako váha informace

Experiment ukázal, že chyba výpočtu vzdálenosti roste se vzdáleností od majáku. Při určování vzdálenosti by tedy mely bližší majáky mít větší váhu. Pri 2D trilateraci (výška ignorována) stačí pouze 3 signály. Je tedy vhodné využít pouze omezený počet nejbližších majáků a zbytek ignorovat, aby se do výpočtu nezanášely zbytečně velké chyby od vzdálených majáků.

Tabulka 5.3: Postup instrukcí pro nastavení *HM-10* jako BLE majáku.

Instrukce	*Popis*	*Odpověď*
AT+RENEW	Tovární nastavení	+RENEW OK
AT+RESET	Restartování modulu	+RESET OK
AT	Čekání na odpověď po restartu	OK
AT+MARJ0x0001	Nastavení iBeacon Major čísla na 0x0001 (hexadecimálně)	+MARJ=0x0001 OK
AT+MINO0x0002	Nastavení iBeacon Minor čísla na 0x0002 (hexadecimálně)	+MINO=0x0002 OK
AT+ADVI5	Nastavení vysílacího intervalu na 5 (546.25 millisekund).	+ADVI=5 OK
AT+NAMEvasina	Nastaveni jména modulu na *vasina*. Je vhodné unikátní pojmenování.	+NAME=vasina OK
AT+ADTY3	Nastavení modulu do režimu *non-connectable* pro úsporu energie. V tomto režimu nebude maják viditelný jako běžné zařízení, ale pouze jako maják.	+ADTY=3 OK
AT+IBEA1	Povolení režimu iBeacon	+IBEA=1 OK
AT+DELO2	Povolení pouze broadcastových zpráv pro úsporu energie	+DELO=2 OK
AT+PWRM0	Povolit uspání při neaktivitě. To umožňuje snížení spotřeby z 8 na 0.18 mA	+PWRM=0
AT+RESET	Restart s novým nastavením	+RESET OK

6. Trilaterační algoritmy

Pojem trilaterace zastřešuje metody, které jako vstup využívají pevně dané referenční body v prostoru a vzdálenosti od nich. Referenční body jsou využity jako středy kruhů, jejichž poloměr je právě vzdálenost mezi referenčním bodem a hledanou polohou.

Trilaterace je princip, který nijak nevymezuje způsob, jakým ze vstupních dat získat hledanou polohu. Algoritmy, které řeší tento problém, lze obecně rozdělit do tří skupin.

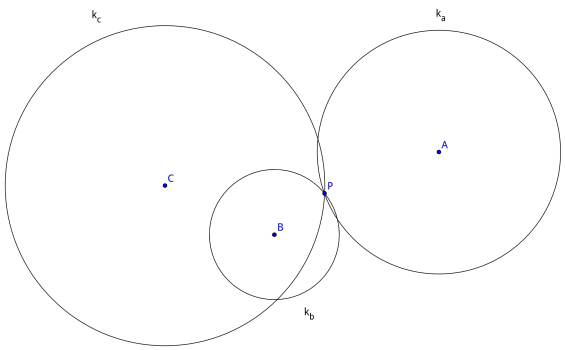
6.1 Exaktní rešení

Tato skupina obsahuje pouze jednu metodu a je uvedena pro úplnost. Vychází z předpokladu nulové chyby při měření vzdáleností od referenčních bodů. V takovém případě se musí protnout veškeré kruhy (resp. kružnice) v jediném bodě (viz obrázek 6.1).

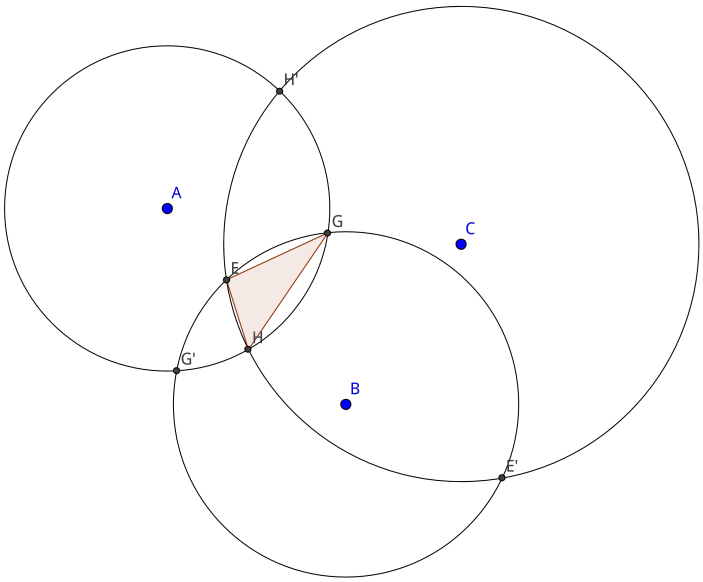
Jsou-li dány majáky $A = (a_x, a_y)$, $B = (b_x, b_y)$, $C = (c_x, c_y)$ a $D = (d_x, d_y)$, dále hledaná pozice $P = (p_x, p_y)$ a vzdálenosti majáků od hledané pozice a_r, b_r, c_r a d_r, lze souřadnice hledaného pozice P vypočítat pomocí soustavy rovnic kružnic (viz 6.1).

$$(p_x - a_x)^2 + (p_y - a_y)^2 - a_r = 0 \tag{6.1a}$$
$$(p_x - b_x)^2 + (p_y - b_y)^2 - b_r = 0 \tag{6.1b}$$
$$(p_x - c_x)^2 + (p_y - c_y)^2 - c_r = 0 \tag{6.1c}$$

Abychom mohli pozici jednoznačně určit, potřebujeme pro 2D trilateraci tři různé kružnice se třemi různými středy.



Obrázek 6.1: Průnik signálů v případě nulové chyby.



Obrázek 6.2: Ideální průnik signálů tří majáků s průsečíky E, E', G, G', H, H'.

6.2 Geometrické metody

Tyto metody [45] využívají také analytické geometrie. Ovšem již nevyužívají rovnice kružnic, ale nerovnice kruhů. Vychází tedy z předpokladu, že získaná vzdálenost od majáku obsahuje chybu. Počítá se tedy s chybou, díky které se kruhy překrývají, a průnikem tedy není jeden bod, ale plocha. Tuto plochu vyznačují průsečíky jednotlivých kružnic, které popisují vzdálenost od majáku.

6.2.1 Clusterové metody

V ideálním případě se všechny kruhy překrývají (viz obrázek 6.2). Ze tří majáků můžeme tedy získat až šest průsečíků. Pouhým okem je jasné, které tři průsečíky lemují kritickou oblast. Bohužel náhled pouhým okem nemáme při výpočtu k dispozici. Místo toho se využívá metoda clusteru.

Každý pár kružnic spolu vytváří dva průsečíky. Vypočteme-li vzájemné vzdálenosti průsečíků nenáležících stejným párům kružnic, stačí již jen vybrat nejbližší z těchto průsečíků. Z této dvojice vypočteme centroid, což je aritmetický průměr jejich jednotlivých souřadnic. Následně vybereme průsečík, který ještě nebyl použit a je centroidu nejblíže. Souřadnice centroidu vypočteme znovu, nyní aritmetickým průměrem všech tří průsečíků.

A Centroid clusteru

Při větším počtu kružnic lze proces aktualizace centroidu provádět opakovaně, dokud nebude počet zahrnutých průsečíků roven počtu majáků, resp. kružnic.

Aritmetickým průměrem souřadnic všech bodů clusteru získáme výslední centroid, což je i těžiště polygonu. Výhodou této metody je jednoduchost a možnost zvolit velikost clusteru, tedy možnost pro výpočet pozice využít více než tři majáky.

V tomto provedení, algoritmus nezohledňuje fakt, že chyba vypočtené vzdálenosti od majáku roste s s reálnou vzdáleností od majáku.

B Vážený centroid clusteru

Tento algoritmus je obdobný jako výpočet obyčejného centroidu. Liší se tím, že se k výpočtu vypočte vážený průměr. Váhy jednotlivých bodů musejí být nastaveny tak, aby reflektovali vlastnost rostoucí chyby s rostoucí vzdáleností od majáku. Váhy tedy mají podobu převrácené hodnoty vzdálenosti nebo převrácené hodnoty kvadrátu vzdálenosti.

C Body trojúhelníkového clusteru

Pokud bude mít cluster velikost 3, bude se jednat o trojúhelník. V něm lze nalézt několik význačných bodů [45]:

Fermatův bod

Fermatův bod P trojúhelníku ABC je takový bod, že $|AP| + |BP| + |CP|$ je minimální. Tedy součet vzdáleností vrcholů od bodu P je minimální.

Lemoinův bod

Lemoinův bod P trojúhelníku ABC je takový bod, že $|aP| + |bP| + |cP|$ je minimální. Tedy součet vzdáleností stěn od bodů P je minimální.

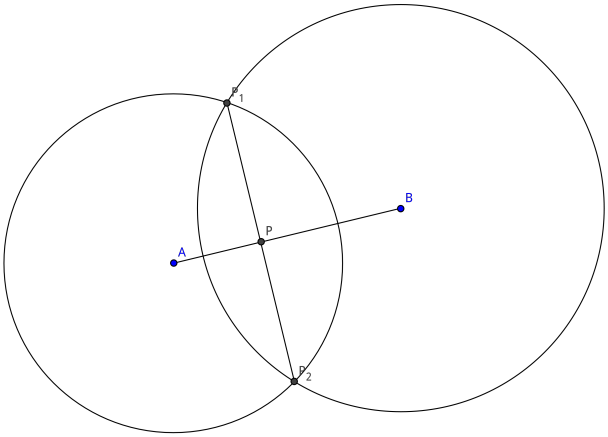
Střed kružnice opsané

Střed kružnice opsané je takový bod P trůjúhelníku ABC, že $|AP| = |BP| = |CP|$. Tedy vzdálenosti vrcholů trojúhelníku od bodu P jsou shodné.

6.2.2 Průnik dvou kruhů

Další jednoduchá metoda spočívající v nalezení průniku dvojice kružnic a následně nalezení průniku spojnice středů kružnic se spojnicí průsečíků (viz obrázek 6.3).

Metoda je také jednoduchá a postačí jí pouze dva kruhy. Opět má své místo pro porovnávací účely, případně pro nouzové situace, kdy budou dostupné pouze dva majáky.



Obrázek 6.3: Určení pozice (P) pomocí průniku pouze dvou signálů.

6.3 Optimalizační metody

Geometrické metody mají výhodu ve své jednoduchosti. Jejich slabinou je ovšem předpoklad dostatečného počtu průsečíků. Jak se při testování ukázalo, často může nastat situace, kdy nedojde vůbec k žádnému protnutí signálů díky multipath efektu, který signály zesílí.

V takové situaci je vhodné použít optimalizační algoritmy [45], které počítají s chybou výpočtu vzdálenosti od majáků. Na základě vstupních dat hledají takovou pozici, v níž bude součet chyb minimální. Modifikací rovnic 6.1 získáme:

$$(p_x - a_x)^2 + (p_y - a_y)^2 - a_r = a_\delta^2 \tag{6.2a}$$
$$(p_x - b_x)^2 + (p_y - b_y)^2 - b_r = a_\delta^2 \tag{6.2b}$$
$$(p_x - c_x)^2 + (p_y - c_y)^2 - c_r = a_\delta^2 \tag{6.2c}$$

Původní soustava uvažovala exaktní řešení a tedy nulovou chybu (proto byl vektor pravých stran nulový). Pokud je uvažována chyba, vektor pravých stran nebude nulový, ale bude obsahovat kvadráty chyb. Optimalizační metody budou právě tyto kvadráty minimalizovat.

6.3.1 Nelineární nejmenší čtverce

Minimalizace kvadrátů chyb ukazuje na jednu z nejznámějších metod – metodu nejmenších čtverců. Metoda nejmenších čtverců se využívá v případech jako je prokládání polynomu zadanými body. Využívá se v situacích, kdy by bylo náročné najít exaktní řešení, nebo pokud máme zadanou přeurčenou soustavu rovnic, tedy rovnic je více než proměnných. Cílem metody je nalezení takového řešení, pro které platí, že součet kvadrátů odchylek od zadaných bodů je minimální.

Pokud zapíšeme systém poskytnutých hodnot do matice jako $Ax = y$, představuje x vektor řešení a vetor y vektor pravých stran.

Rozdíl vypočteného řešení Ax a y představuje chybu, jejíž kvadrát se bude minimalizovat. Tedy $(Ax-y)^2 = min$. Způsob, jak nalézt vektor x, který minimalizuje celý výraz, je jednoduchý – je nutné postavit jeho derivaci nule. Je tedy nutné vyřešit $((Ax - y)^2)' = 0$.

Postup nalezení minima prostřednitvím derivace se již liší v závislosti na podobě aproximační polynomu. V případě trilaterace je problém ještě složitejší, protože se již nejedná o lineární aproximaci. V kvadrátu se zde objevují díky rovnicím kruhů obě proměnné x i y. Díky tomu úloha vede na tzv. Nonlinear Least Squares, tedy nelineární nejmenší čtverce.

K řešení tohoto problému se již nevyužívá výpočet exaktního řešení (kvůli složitosti), ale numerické iterativní metody, konkrétně Levenberg–Marquardtova metoda. Jedná se o kombinaci Gaussovy iterační metody a metody největšího spádu.

Metoda spadá již do pokročilejší matematiky a její detailní popis není předmětem této práce. Metoda je implementována mj. v Apache Math knihovně.

6.3.2 Nelder-Mead simplexová metoda

Simplexová metoda je další iterativní optimalizační metodou [46]. Tato metoda, na rozdílnarozdíl od nejmenších čtverců, nevyžaduje derivování. Místo toho postupně změnšuje polygon popisující prostor řešení.

Pokud hledáme řešení v prostoru R^n, je zapotřebí minimálně $n + 1$ počátečních bodů (simplex). Pro případ indoor lokalizace, kde je ignorována výšková osa, tedy potřebujeme trojici bodů. Jako počáteční body mohou posloužit například souřadnice tří nejbližších majáků.

Dalším požadavkem této metody je tzv. fitness funkce $f(x_n)$. Fitness funkce slouží pro ohodnocení kvality libovolného bodu. Díky tomu je možné tyto body mezi sebou porovnávat a vybírat nejlepší.

Celý algoritmus lze zapsat takto:

1. **Seřazení bodů**

   Seřadit všechny body dle výsledku fitness funce. Tedy porovnání na základě $f(x_1) \leq f(x_2) \cdots \leq f_{n+1}$.

2. **Výpočet centroidu**

   Vypočítat centroid x_0 jako aritmetický průměr souřadnic všech bodů kromě nejhoršího.

3. **Reflexe**

   Vypočítat reflexní bod jako $x_r = x_0 + \alpha(x_0 - x_{n+1})$, kde $\alpha > 0$. Pokud je reflexní bod x_r lepší než druhý nejhorší bod x_n, ale ne lepší než nejlepší bod x_1 (tj. $f(x_1) \leq f(x_r) < f(x_n)$), pak je nejhorší bod x_{n+1} nahrazen bodem x_r a pokračuje se bodem 1.

4. Expanze

Pokud je reflexní bod nejlepší ($f(x_r) < f(x_1)$), je vypočten expanzní bod $x_e = x_0 + \gamma(x_r - x_0)$, kde $\gamma > 1$. Nejhorší bod x_{n+1} je nahrazen lepším z dvojice x_r, x_e a pokračuje se bodem 1.

5. Kontrakce

Pokud platí $f(x_r) \geq f(x_n)$, je vypočten bod $x_c = x_0 + \rho(x_{n+1} - x_0)$, kde $0 < \rho \leq 0.5$. Pokud je nový bod lepší než nejhorší bod ($f(x_c) < f(x_{n+1})$) je nejhorší bod x_{n+1} nahrazen bodem x_c a pokračuje se bodem 1.

5. Kolaps

Posunutí všech bodů kromě nejlepšího (x_1) směrem k nejlepšímu bodu ($x_i = x_1 + \omega(x_i - x_1)$). Následně se opět pokračuje bodem 1.

Zastavovací podmínkou může být počet iterací nebo vzdálenost centroidů posledních dvou iterací. Je patrné, že simplex je vlastně polygon, který postupně mění tvar a posouvá se směrem k nejlepšímu řešení, kterým se nakonec stane nejlepší bod simplexu po ukončení algoritmu.

Opět se jedná o známou optimalizační metodu, jejíž implementaci lze nalézt v Apache Math knihovně.

7. Analýza

Cílem práce je navrhnout aplikaci pro indoor lokalizaci mobilního zařízení. Hlavními kritérii jsou:

- minimalizace nákladů
- odstranění potřeby dodatečného HW pro uživatele
- snížení požadavků na dodatečný HW pro zřizovatele
- náročnost implementace
- přesnost
- minimalizace doby nasazení
- snížení energetické náročnosti

Na základě kritérií (viz tabulka 3.2) byla vybrána metoda BLE trilaterace s konstrukcí vlastních majáků. Ta umožňuje lokalizaci mobilního zařízení, která od zákazníka vyžaduje pouze mobilní zařízení s technologií BLE. Od zřizovatele je nutné poskytnout BLE majáky a jejich souřadnice. Díky nízkým energetickým nárokům BLE technologie mohou být majáky napájeny z baterie. Majáky se tak vejdou do dlaně a jsou mobilní.

7.1 Využité lokační metody

Trilaterace popisuje pouze princip získání polohy na základě známých pozic a vzdáleností. Samotné vyřešení této úlohy lze realizovat několika způsoby (viz tabulka 7.1).

Pro testování aplikace budou implementovány a využity všechny algoritmy z tabulky 7.1.

7.2 Zobrazení

Při určování polohy je kromě výpočtu polohy podstatná také reprezentace výsledku. Výsledkem je vypočtená poloha bodu v prostoru, resp. v rovině. Tento soubor hodnot je pro člověka ovšem nedostatečně názorný. Body je zapotřebí zanést graficky do plánku místnosti nebo patra budovy. Promítnutí vypočtené pozice do plánu není triviální problém.

7.2.1 Metrický systém versus pixely

V reálném prostředí jsou souřadnice a vzdálenosti založené na metrickém systému. S tím počítají i trilaterační výpočty. Pro práci s obrázky a obrazovkou se ovšem používají bezrozměrné pixely.

Pro převod mezi reálným prostředím a souřadnicemi na obrazovce je tedy nutný údaj v jednotkách px/m, tedy počet pixelů na metr. Díky tomuto měřítku je pak možné např. vykreslit kružnici představující dosah signálu (*distance*). Poloměr této kružnice představuje vzdálenost od majáku v metrech. Pro vykreslení kružnice je ovšem zapotřebí poloměr kružnice v pixelech (*pixelRadius*) (viz vzorec 7.1).

$$pixelRadius = distance \cdot X[\frac{px}{m}] \tag{7.1}$$

7.2.2 Rozdílná orientace souřadných systémů

Dalším úskalím je fakt, že soustava souřadnic obrázků i obrazovky má počátek v levém horním rohu a roste směrem k pravému dolnímu rohu. Ovšem např. souřadný systém využívaný pro vykreslování grafů se chová jinak. Hodnota souřadnic roste směrem vpravo nahoru.

Nejjednodušším řešením tohoto problému je zohlednit jej už při vytváření souřádné sítě místnosti. Pokud je pro místnost využita soustava souřadná, která se chová stejně jakou soustava obrazovky, není následně nutný žádný přepočet uvnitř aplikace.

7.2.3 Rotace a zoom

Pro uživatelskou přívětivost je často požadováno, aby bylo možné plán oblasti zvětšovat či otáčet. V případě, že se celý plán vejde na obrazovku nebylo by nutné nic počítat, stačilo by pouze výše zmíněné měřítko $\frac{px}{m}$. V situaci, kdy se plánem začne otáčet nebo se zvětší a nevejde se do vymezeného prostoru na obrazovce, nastává problém. Souřadnice pixelů obrázku již nekorespondují se souřadnicemi obrazovky.

Situaci popíše jednoduchý příklad. Obrázek, který se vejde na obrazovku, má nulový bod v levém horním rohu stejně jako obrazovka. Ve chvíli, kdy se obrázek otočí o 180°, bude počátek souřadnic obrazovky stále v levém horním rohu. Počátek souřadnic obrázku však bude v pravém dolním rohu. Při zvětšování je problém ještě komplikovanější, protože z obrázku je vidět pouze omezený úsek, zbytek leží mimo obrazovku, tedy na záporných souřadnicích obrazovky.

Tento problém se řeší přepočtem souřadnic za pomoci matice. Koeficienty matice zachycují změnu (zvětšení, rotaci a posun) obrázku oproti originálu. Díky tomu lze prostým násobením souřadnic touto maticí získat souřadnice v obrázku a naopak. Funkce implementující operace s těmito maticemi jsou programovou výbavou vývojových platforem.

7.2.4 Přepočet souřadnic

Pro zobrazení vypočtené pozice či majáků na plánku je nutné využít všechny výše zmíněné principy.

1. Vytvořit plán v podobě obrázku.
2. Určit nulový bod na plánu i v prostoru.
3. Při určování reálných souřadnic majáků respektovat orientaci souřadného systému obrazovky.
4. Metrické souřadnice majáků nebo pozice přepočítat na pixelové souřadnice plánku pomocí $\frac{px}{m}$ měřítka.
5. Pixelové souřadnice plánu za pomoci transformační matice převést na pixelové souřadnice obrazovky.
6. Vykreslit bod.

7.3 Vstupní data

BLE triangulace využívá jako vstupní data intenzity signálů jednotlivých majáků. To je ovšem možné až ve chvíli, kdy jsou získána potřebná startovací data (minor a major hodnoty majáků spojené s jejich souřadnicemi).

7.3.1 Pozice majáků

Před zahájením trilaterace je nutné získat seznam pozic jednotlivých majáků. Majáky musejí být jednoznačně identifikovatelné a k tomuto identifikátoru je přidružena pozice.

Souřadnice majáku musí respektovat orientaci souřadného systému obrazovky. Souřadnice majáků vycházejí ze vzdáleností v reálném prostředí.

7.3.2 Pixely na metr

Pro možnost přepočtů mezi pixely a metry je nutné znát měřítko $\frac{px}{m}$ jak pro osu x, tak pro osu y. Toto měřítko lze získat přímo ze startovacích dat, nebo jej vypočíst. K výpočtu je potřeba znát počet metrů, který představují všechny pixely obrázku na výšku i na šířku. Měřítko tedy nemusí být v obou osách stejné.

7.3.3 Mapový podklad

Pro zobrazení pozice je velmi důležitý plán místnosti, patra atd. Mapový podklad není nic jiného, než obrázek reprezentující prostředí, na který je následně promítána aktuální pozice. Tento mapový podklad může být rovnou obsažen v

konkrétní implementaci aplikace. Další možností je nalezení plánu v úložišti nebo na internetu.

Pro upřesnění určování polohy je vhodné tento plán doplnit i o informace, které plochy plánu nejsou přístupné. V případě, že je výpočtem získána pozice, která leží v nepřístupné oblasti, bude vybrána nejbližší možná pozice.

Nedostupné plochy je možné popsat pomocí souřadnic obdélníkových ploch nebo pomocí barev. Popis ploch znamená větší objem vstupních dat, ale jednodušší testování validity pozice. Nalezení nejbližší validní pozice je však komplikované. Popis nedostupných oblastí pomocí barev je úspornější na vstupní data a lze také snáze určit nejbližší validní pozici.

Testování dostupnosti pozice na základě barev je prosté. Na vypočtených pixelových souřadnicích je zjištěna barva podkladu. V případě, že se jedná o zakázanou barvu, postupuje se od tohoto pixelu v kruzích a znovu se zjišťuje barva těchto pixelů. Ve chvíli, kdy je nalezen pixel označující dostupnou barvu, je pozice přesunuta na tento pixel.

Tabulka 7.1: Přehled trilateračních metod.

Metoda	*Klady*	*Zápory*
Průnik dvou kruhů	• snadná implementace • vhodný při nedostatečném počtu majáků	• vyšší nepřesnost • nezohledňuje nepřímou úměru mezi vzdáleností a přesností
Centroid clusteru	• snadná implementace • často používaný (dostatečný počet zdrojů) • lze pro výpočet využít více jak 3 majáky	• závislý na počtu průsečíků kružnic • nezohledňuje nepřímou úměru mezi vzdáleností a přesností
Vážený centroid clusteru	• snadná implementace • vhodný při nedostatečném počtu majáků • jednoduchá modifikace výpočtu běžného centroidu • zohledňuje nepřímou úměru mezi vzdáleností a přesností • lze využít více než tři majáky	• závislý na počtu průniků
Nelineární nejmenší čtverce	• implementováno v matematických knihovnách • není závislý na průsečících kružnic • pomocí vah lze zohlednit nepřímou úměru mezi přesností a vzdáleností	• iterativní algoritmus • výsledek může být doručován se zpožděním

8. Aplikace

Byla provedena analýza dostupných lokalizačních metod a dostupných technologií. Pro lokalizační jádro aplikace byla zvolena metoda trilaterace pomocí BLE majáků vlastní výroby, které ovšem respektují *iBeacon* standard.

Aplikace byla rozšířena o možnost určování pozice ve více patrech (viz sekce).

Pro implementaci aplikace byla zvolena platforma Android z důvodu:

- otevřenosti
- rozšířenosti mezi uživateli
- předchozích zkušeností autora s technologií
- široké škály dostupných knihoven
- velikosti programátorské komunity

8.1 Použité technologie

Vývoj pro platformu Android je možné realizovat několika způsoby:

Android Studio [50]

Vývojové studio podporované přímo společností Google. Využívá upravenou verzi studia InteliJ Idea pro implementaci v Javě s využitím Android SDK.

Xamarin [47]

Technologie umožňující vývoj multiplatformních aplikací (Android, iOS, Windows Phone a další) pomocí jazyka C#.

Apache Cordova [49]

Tato technologie využívá HTML5, CSS3 a Javascript pro implementaci multiplatformních mobilních aplikací. Tyto aplikace běží v kontejneru, který zajišťuje komunikaci s mobilním zařízením.

Z těchto variant vývoje byla vybrána první možnost, tedy implementace v Javě pomocí Android SDK. Důvodem byla zejména předchozí znalost technologie, velikost komunity, vyzrálosti technologie a dostupnost knihoven pro jazyk Java jako např. matematických knihoven *Apache*.

8.1.1 Knihovny třetích stran

Ve chvíli, kdy byla vybrána technologie, bylo nutné vyřešit základní otázky:

- Jak získat údaje z majáků a konvertovat je na vzdálenost?
- Jak z pozic a vzdáleností majáků vypočítat polohu zařízení?

- Jak uživatelsky přívětivě reprezentovat data?
- Jak získat počáteční informace jako např. souřadnice majáků?

Před implementací bylo nutné otázky zodpovědět. Odpovědi na otázky zároveň představovaly parametry pro hledání již existujících řešení. Výsledkem je využití knihoven, které jsou součástí systému repozitářů *Maven* i *Gradle*. Díky přítomnosti knihoven v repozitářích je možné mít k dispozici neustále aktuální verzi knihoven a zároveň je jejich integrace do aplikace jednodušší. Níže je u jednotlivých knihoven zmíněno, jak pomohly zodpovědět klíčové otázky aplikace.

A Altbeacon [39]

Altbeacon je knihovna, která zastřešuje práci z BLE majáky různých standardů (*iBeacon*, *Eddystone*, *Altbeacon*). Pro své mobilní aplikace ji využívají firmy(zdroj) jako např. McDonald's, KFC nebo Air France.

Za hlavní výhodu považuji fakt, že API této knihovny již obsahuje výpočet vzdálenosti majáku na základě RSSI a TX hodnot poskytnutých majákem. Tyto údaje jsou v základním nastavení poskytovány přibližně s frekvencí 1 Hz. Toto nastavení lze změnit, ale není to doporučeno kvůli spotřebě energie. Další problém hovořící proti zvýšení frekvence, je zkrácení periody vzorků. Pokud není během vzorkování přijata celá zpráva, je zahozena. Díky zvýšení frekvence se počet takto zahozených zpráv zvýší.

Jak knihovnu zaregistrovat v aplikaci lze nalézt přímo v ukázkovém zdrojovém kódu. Z hlediska implementace lokační aplikace je nejpodstatnější metoda `void didRangeBeaconsInRegion(Collection<Beacon>, Region)`. Tato metoda je volána knihovnou *Altbeacon* ve chvíli, kdy jsou v okolí nalezeny nějaké majáky. Jedná se o metodu rozhraní `RangeNotifier`.

Rozhraní `RangeNotifier` patří mezi návrhové vzory *Listener*. Implementace tohoto rozhraní jsou volány ve chvíli kdy je zaregistrován signál od majáků. Knihovna *Altbeacon* volá pouze rozhraní, tudíž implementaci posluchače je nutné dodat ze strany programátora.

B The Apache Commons Mathematics Library [48]

Tato knihovna byla využita pro implementaci trilaterace pomocí metody nelineárních nejmenších čtverců, konkrétně Levenberg–Marquardtovou metodou. Implementace této metody není triviální, proto bylo využito již existující knihovní funkce.

C Subsampling Scale Image View [44]

Pro zobrazení mapového podkladu bylo nutné vyřešit nejen vykreslení obrázku, ale i reakce uživetelského rozhraní a přepočet souřadnic. Možnost přiblížení obrázku pomocí dotykových gest je v mobilních aplikacích častý požadavek. Problém při zvětšování obrázku tkví v tom, že se mění i souřadnice, protože po zvětšení

či rotaci jsou již souřadnice obrázku a obrazovky jiné. Problematika souřadnic i reakcí na gesta je řešena pomocí modifikací této knihovny. Knihovna navíc nabízí možnost načítat velké obrázky na části pomocí dlaždic tak, jak je to známé např. z Google Maps. Tato funkcionalita tedy dává prostor budoucím vylepšením bez nutnosti velkých zásahů do zdrojového kódu aplikace.

D Picasso [51]

Pokud jsou v aplikaci využívány obrázky (v tomto případě mapové podklady), je nutné vyřešit jejich získání a zobrazení v aplikaci. Mapový podklad lze získat z internetu pomocí URL adresy, ze souboru v úložišti (SD karty), nebo z množiny obrázků, které jsou přímo součástí aplikace. Knihovna Picasso poskytuje API, které zjednodušuje získání obrázků z různých zdrojů včetně jejich případného cachování (dočasného ukládání v paměti místo opětovného stahování).

E Zxing [43]

Tato knihovna poskytuje možnost získávat textové informace z QR čárových kódů. Načítání QR kódů je v aplikaci využito pro počáteční získání informací o majácích a mapových podkladech.

8.2 Struktura aplikace

Na CD je přiložen celý projekt aplikace pro Android Studio, který obsahuje mnoho generovaných souborů, které nejsou pro rozbor důležité. Hlavní zdrojové soubory aplikace leží v adresáři `app/src/main/java/com/wikif/com/bletest` na CD, jehož strukturu popisuje příloha A.

Cílem práce není čtenáře seznámit s fungováním platformy Android SDK, proto není tato problematika popsána podrobněji.

8.2.1 Aktivity

Jedním ze základních stavebních prvků na platformě Android je *Aktivita*. Aktivita je kód pro jednu funkční obrazovku aplikace. Třída obsahuje zdrojový kód pro práci programu i správu aktuálního uživatelského rozhraní.

Třída `Settings` má na starost pouze zobrazení nastavení aplikace a uložení tohoto nastavení pro uchování nastavení mezi starty aplikace. Patří sem například aktivace jednotlivých trilateračních algoritmů nebo spuštění vývojářského režimu (viz příloha C).

Hlavní třídou aplikace je `MainActivity`, která je spuštěna jako hlavní aktivita aplikace při startu. Uživatelské rozhraní je popsáno v příloze C. Aktivita kromě UI spravuje:

- sledování BLE majáků
- načtení a perzistenci vstupních dat (mapový podklad, souřadnice majáků)
- spouštění vláken pro výpočet pozice zařízení
- běh vlákna pro pravidelnou kontrolu dostupnosti majáků
- vykreslení polohy zařízení a majáků na mapový podklad
- ve vývojářském režimu dále zpřístupňuje seznam s informacemi o majících

8.2.2 TrilaterationAlgorithms

Balík obsahuje mj. rozhraní trilateračního algoritmu, který aplikace očekává. Obsahuje jedinou metodu `getPosition(Point[], double distances)`. Tato metoda vrací objekt `Point`, který v sobě udržuje metrické souřadnice vypočtené polohy. Krom uživatelského rozhraní se všude v aplikaci chápou souřadnice jako souřadnice v metrech od počátku soustavy souřadné. Toto rozhraní v aplikaci implementují čtyři třídy:

SingleBeacon představující nejprimitivnější verzi pozicování – jako pozice jsou využity souřadnice nejbližšího majáku.

SimpleCentroid implementuje algoritmus, který z průniků tří kružnic nejbližších tří majáků získá tři průsečíky. Z těchto bodů je vypočten centroid – aritmetický průměr souřadnic. Tento algoritmus je použitelný pouze pokud se dosahy signálů překrývají.

WeightedCentroid představuje modifikace jednoduchého centroidu. Výsledné souřadnice jsou vypočteny jako vážený průměr tří průsečíků, přičemž váhy jsou stanoveny jako převrácené hodnoty kvadrátů vzdáleností majáků – nejbližší maják má tedy nejvyšší váhu.

NonLinearLeastSquares je implementace optimalizační metody nejmenších čtverců. Tato metoda není založená na geometrických průsečících a díky tomu funguje i ve stavu, kdy se dosahy signálů neprotínají.

8.2.3 Custom Views

Aplikace využívá dva upravené seznamy pro vývojářský režim aplikace. Jeden slouží pro zobrazení podrobností o majících načtených z konfigurace – aktuální vzdálenost, RSSI, TX, Major, Minor atd.

Druhý seznam slouží pro zobrazení číselných výsledků jednotlivý trilateračních algoritmů.

Posledním prvkem tohoto balíku je upravená třída `IndoorView`, která vznikla děděním třídy z knihovny *Subsampling Scale Image View*. Tato třída je pro aplikaci velmi důležitá, protože se stará o zobrazení mapového podkladu a následně majáků a pozice mobilního zařízení. Třída má oddělený seznam majáků i výsledných pozic zařízení, díky čemuž umí zobrazit více pozic najednou.

8.2.4 Utils

Jedná se o balík obsahující třídy a metody pro zjednodušení výpočtů s body (`Point`) a práci se soubor.

8.2.5 Entities

Tento balík obsahuje základní objekty, se kterými pracují jak aktivity, tak přímo trilaterační algoritmy. Patří sem:

Beacon

Objekt nesoucí informace o konkrétním majáku – informace odpovídající *iBeacon* standardu. Dále třída obsahuje informace o tom, jak dlouho nebyl signál majáku zachycen (aby mohl být označen za neaktivní).

IniDataBlock

Třída zapouzdřující vstupní data pro aplikaci, tedy pozice majáků, zdroje mapových podkladů. Aplikace umožňuje tato data získat skrze QR kód nebo ze souboru v mobilním zařízení. Předpokládá se formát Json (viz CD `/test_cases/json`), který je z QR kódu získán nebo přímo načten ze souboru. Data v Json formátu jsou následně přeložena a je vytvořena instance `InitDataBlock`. Je ovšem možné implementovat vlastní překlad a využít například data ve formátu XML.

Point

Jedná se o jednoduchou třídu představující bod v rovině (začátek soustavy souřadné, pozice majáku, pozice zařízení).

PaintedPoint

Jde o rozšíření třídy `Point`, které kromě souřadnic obsahuje navíc atribut typu `Paint`, který v prostředí Androidd představuje informaci o barvě a stylu výplně. Tyto upravené body jsou využívány v UI, kde je nutné znát nejen, pozici vykreslení bodu, ale také jakou barvou.

SignalCircle

Jak název napovídá, třída nese informaci o pozici majáku a jeho vzdálenosti.

8.3 Životní cyklus aplikace

Aplikace z hlediska určování pozice prochází několika stavy.

8.3.1 Získání vstupních dat

Pro určování pozice je nutné znát polohy a identifikace (Major a Minor) majáků, spolu s mapovým podkladem. V případě lokalizace na více patrech lze poskytnout

i více mapových podkladů. Pokud mapový podklad chybí je využita přednastavená síť v aplikaci. Jako formát vstupních dat byl zvolen Json, který lze získat ze souboru nebo pomocí QR kódu.

Pokud vstupní data obsahují URL obrázků mapových podkladů, jsou tyto údaje staženy prostřednictvím knihovny Picasso, která asynchronně stáhne data z internetu.

8.3.2 Zpracování dat z majáků

Majáky v pravidelných intervalech vysílají broadcastové zprávy. Ty jsou po zachycení zpracovány a nové hodnoty jsou promítnuty do seznamu majáků získaných ze vstupních dat.

Majákům, jejichž signál nebyl v tomto cyklu zachycen, je inkrementován počet promeškaných cyklů (po dosažení limitu 5 je maják označen jako neaktivní).

A Výpočet vzdálenosti majáku – ARMA filtr

Vzdálenost majáku je vypočtena pomocí knihovny *Altbeacon*. Není ovšem použita pouze aktuální hodnota. Kvůli kolísání intenzity signálu je použit primárně plovoucí průměr (o velikosti 20 vzorků). Ten ovšem způsobuje velkou prodlevu změny dat (přibližně 10 sekund). Pro určování polohy pohybujícího se člověka tedy není vhodný. Knihovna *Altbeacon* umožňuje přepnout na tzv. *ARMA filtr (Autoregressive–moving–avarage)*.

Jak název napovídá, je součástí algoritmu stále plovoucí průměr, nicméně za cenu případné větší chyby umožňuje rychlejší reakci na změny v hodnotách.

8.3.3 Výběr majáků

Aktualizované majáky jsou seřazeny vzestupně dle vzdálenosti, přičemž jsou vynechány neaktivní majáky. První maják v seznamu je tedy nejbližší a z tohoto majáku získáme informaci o aktuálním patře. Pokud se nově zjištěné patro liší od patra předchozího cyklu, je zobrazen nový mapový podklad a na něm pouze majáky z aktuálního patra. Majáky, které nejsou na aktuálním patře, jsou také ignorovány. Následně jsou vybrány tři nejbližší majáky (tj. 3 nejbližší aktivní majáky nacházející se na aktuálním patře). Tyto majáky jsou použity jako vstup trilateračních algoritmů.

Pokud je zvolen vývojářský režim, proběhnou všechny zvolené trilaterační algoritmy, a jsou zobrazeny, pokud jejich výsledkem není hodnota `null`. Pokud je vývojářský režim vypnutý je pro výpočet polohy použita metoda nejmenších čtverců, v krajním případě přítomnosti pouze jednoho aktivního majáku je použita metoda nejbližšího majáku.

8.3.4 Vývojářský režim

Pokud je tento režim povolen, je možné si v uživatelském rozhraní prohlédnout i aktuální hodnoty poskytované majáky a číselné údaje vypočtených pozic. Na mapě se zároveň zobrazí kruhy kolem majáků, které představují vypočtenou vzdálenost od majáku.

8.3.5 Vypnutí aplikace

Pokud je aplikace vypnuta, jsou pozice majáků a zdroje mapových podkladů uloženy (opět v podobě Json formátu), aby mohly být načteny při příštím spuštění.

8.3.6 Detekce offline režimu

Pokud jsou všechny majáky označeny jako neaktivní a není tedy z čeho určit poloha, pak se v UI objeví varování a na mapě jsou zobrazeny pouze pozice majáků. Ve chvíli, kdy je znovu zachycen maják, varování zmizí a aplikace běží dále dle výše popsaných bodů. Detekce tohoto stavu je zajištěna vláknem, které je periodicky spouštěno. K detekci offline stavu nebylo možné využít knihovní funkce, protože detekce nedostupných majáků je závislá na implementaci OS Android, který majáky označí za neaktivní až po 30 sekundách.

8.3.7 Asynchronní úkoly

Výpočet pozice a případné stahování mapových podkladů samozřejmě neprobíhá v hlavním vlákně UI, je k tomu využita konstrukce `AsyncTask`, která umožňuje časově náročné úkoly provést v jiném vlákně a do uživatelského rozhraní promítnou postup nebo rovnou výsledky akce.

9. Nasazení a testování aplikace

Bylo vytvořeno a testováno několik scénářů, které prověřily funkčnost a použitelnost aplikace. Scénáře jsou zároveň návodem, jak aplikaci nasadit a vytvořit potřebná vstupní data.

9.1 Příprava dat

Aplikace potřebuje pro svůj běh informace o majácích, které bude sledovat, a mapový podklad.

9.1.1 Nastavení majáků

Majáky je nutné nastavit tak, aby každý z nich měl unikátní kombinaci *Major* a *Minor* čísla. Následně je nutné nastavit hodnoty TX a sílu vysílaného signálu. Vše se provádí pomocí instrukcí popsaných v sekci 5.1.1.

9.1.2 Mapový podklad

Mapový podklad je nutný proto, aby zobrazená pozice měla pro uživatele nějaký smysl a mohl se orientovat v prostředí. Pro testování byl jako mapový podklad využit plán patra KIV v budově NTIS (k nalezení na CD `/test_cases/maps/`). Mapový podklad zároveň představuje plán pro jedno patro. Pro více pater bude zapotřebí více mapových podkladů.

A Výpočet měřítka

Pro korektní fungování je nutné aplikaci dodat měřítko mapového podkladu v jednotkách *pixel na metr*. Její výpočet není nijak složitý, postačí obyčejný metr a program malování v počítači.

Změříme nějakou konkrétní vzdálenost – v tomto případě šířku chodby mezi kancelářemi. Chodba je 1.8 metru široká. Pomocí grafického editoru zjistíme, kolik pixelů činí stejná vzdálenost na plánku patra. V tomto případě je to 44 pixelů. Prostým vydělením počtů pixelů reálnou vzdáleností získáme hodnotu 24.5 pixelů na metr (*pxpm*).

B Výpočet posunu počátku souřadnic

Zadávání souřadnic majáků probíhá pomocí souřadnic x (vzdálenost od levého okraje) a y (vzdálenost od horního okraje) v metrech. Na plánku je levý horní roh jasně patrný. V reálném prostoru ale tento údaj neexistuje a je nutné jej zvolit. Typicky to může být třeba vchod do budovy nebo dveře kabinetu. Tento

bod je počátkem soustavy souřadné pro reálné souřadnice. Aplikace potřebuje znát souřadnice tohoto bodu na plánku, díky čemuž zná posun (*offset*) soustavy obrázku a reálného prostředí vůči sobě.

C Výpočet souřadnic majáků

Souřadnice majáků se zadávají v metrech a jsou to souřadnice reálného prostředí. Jednou z možností je tedy měřit vzdálenosti od zvoleného počátku v prostoru.

Jednodušší varianta je s pomocí plánku budovy tyto souřadnice (vzdálenosti) vypočítat buď využitím měřítka plánku vůči reálnému světu (pokud je informace dostupná), nebo využít vypočteného měřítka *pxpm*. To samozřejmě násobením slouží k převodu metrů na pixely a dělením k převodu pixelů na metry.

D Identifikace majáku

Maják musí být jednoznačně identifikován kombinací *Major* a *Minor* čísel. Stačí již jen k těmto údajům přiřadit vypočtené souřadnice majáku. To spolu s měřítkem *pxpm* a posunem počátků souřadných soustav *offset* stačí pro zobrazení majáků a výpočet pozice zařízení. Pro lepší orientaci je vhodné dodat mapový podklad.

E Serializace vstupních dat

Nyní je nutné tato získaná data zapsat do formátu Json, se kterým aplikace umí pracovat. Je k tomu využit formát Json, jehož struktura je patrná nejlépe ze souboru `/test_cases/json/beaconsFloors.txt` na CD, který zachycuje úplnou konfiguraci pro více pater.

9.1.3 Instalace aplikace

Aplikace se na OS Android typicky instalují z internetového obchodu. Pro instalaci ze souboru je nutné mít v nastavení zařízení povolenou možnost instalace aplikací z neznámých zdrojů.

Aplikaci lze jednoduše překopírovat do úložiště telefonu a poté ji ve správci souboru najít a spustit.

Druhou možností je otevření složky `/src/BLETest` v Android Studiu. Po otevření se ihned spustí překlad a stahování knihoven, což v závislosti na internetovém připojení může trvat i 15 minut. Projekt lze spustit i z Android Studia, které nabídne možnost spustit aplikace v emulátoru nebo v mobilním zařízení, které je připojeno USB kabelem.

9.1.4 Řešení problémů

Aplikace ke svému fungování vyžaduje verzi Android 4.3 a vyšší z důvodu práce s BLE. Dále vyžaduje přístup k Bluetooth a nakonec je nutné mít povoleny lokační služby zařízení. Aplikace si tato oprávnění sama vyžádá.

A Načtení dat aplikací

Data lze načíst buď skrze QR kód nebo manuálně z úložiště zařízení.

Pro vytvoření takového souboru je možné editovat některý ze souborů na CD (`/test_cases/json/`) nebo využít přiložený grafický program (`/test_cases/data_generator/IndoorNavConfigurator.exe`), který umí vytvořit požadovaná data v Json formátu a zároveň generovat i QR kód s e vstupními daty.

O vytvoření QR kódu se může postarat přiložený program nebo lze vytvořená data v Json formátu kopírovat do jednoho z volně dostupných online generátorů.

9.2 Testovací scénáře

Pomocí těchto scénářů byla aplikace testována v prostorách fakulty. Mapové podklady i konfigurace majáků jsou dostupné na CD v sekcích `/test_cases/json` a `/test_cases/maps`.

9.2.1 Dlouhá úzká chodba

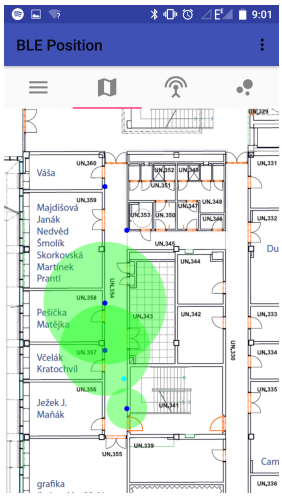
Pro tento test byla zvolena jedna z chodeb mezi kabinety (viz obrázky 9.1a, 9.1b, 9.1c). Pro tento test je využit soubor `/test_cases/json/beacons.txt`. Po jeho otevření aplikací dojde k získání údajů o majácích a zahájení stahování mapových podkladů z internetu. Pokud není připojení dostupné, je možné podklady nalézt manuálně v úložišti zařízení.

Cíl testu

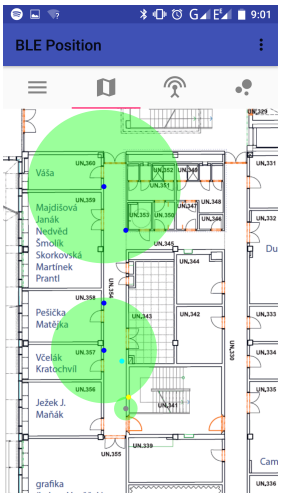
Cílem bylo získat srovnávací data pro další testy. Malý prostor s velkým počtem majáků tvoří ideální testovací podmínky.

Výsledky testu

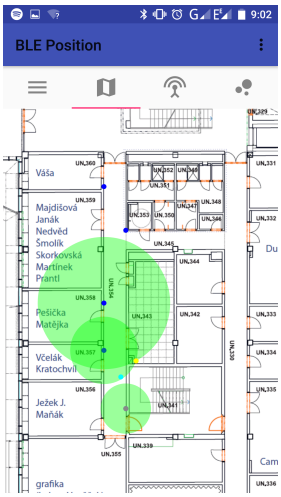
Záznam z testu zachycují obrázky (9.1a, 9.1b, 9.1c). Na úzké chodbě o rozměrech přibližně (1.8 x 21.6 metru) bylo umístěno pět majáků. Co se týče přesnosti vypočtené pozice, byl tento test nejúspěšnější. Ovšem díky tvaru chodby byl patrný i vliv multipath efektu, díky kterému se vzdálené majáky občas jevily daleko bližší.



(a) Běžný případ v úzké chodbě – neprotnou se všechny kruhy.



(b) Vliv multipath efektu – jako blízký byl vybrán vzdálený maják.



(c) Zde vážený průměr získal pozici již mimo chodbu.

Obrázek 9.1: Náhled testu v úzké chodbě *modré* body jsou majáky, *světle modrý* bod je pozice získaná metodou nejmenších čtverců, *žlutý* bod je vážený metody váženého centroidu.

9.2.2 Dlouhá úzká chodba s nízkým počtem majáků

Tento scénář vužívá jako vstupní konfiguraci majáků soubor `/test_cases/json/beaconsLess.t`
Mapový podklad je opět stažen z internetu.

Cíl testu

Smyslem testu je zjistit pokles přesnosti vypočtené pozice po odebrání poloviny majáků a tím i prodloužení jejich vzájemných vzdáleností.

Výsledky testu

Přesnost vypočtené polohy se zhoršila na přibližně 4 metry. Ovšem na chodbě o délce 21 metrů byly pouze tři majáky a i s touto nepřesností bylo snadné se orientovat viz (obrázek 9.2).

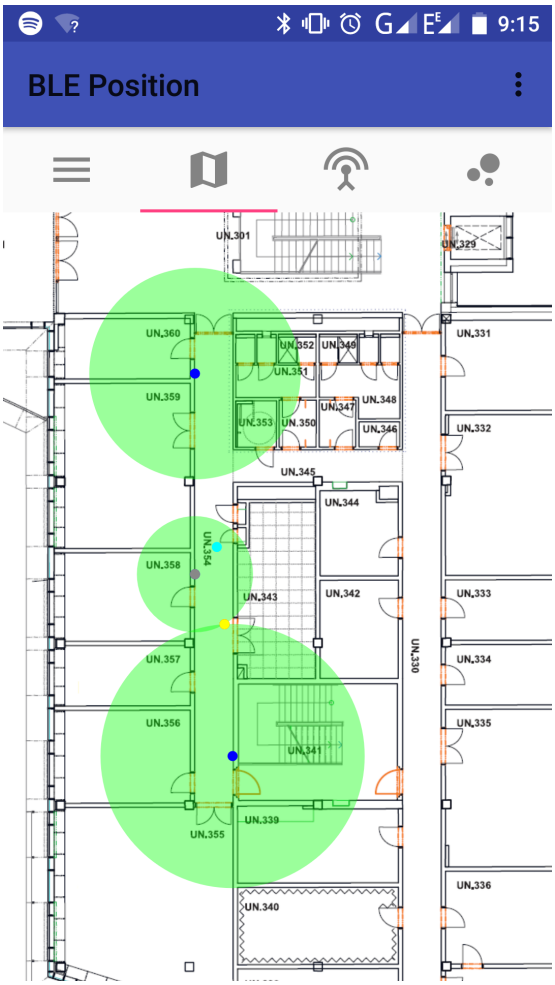
9.2.3 Chodba ve tvaru H

Tento scénář využívá jako vstupní konfiguraci majáků soubor `/test_cases/json/beaconsH.tx`
Mapový podklad je opět stažen z internetu. Z důvodu omezeného počtu majáků (5) bylo do spojovací chodby umístěn pouze jeden maják.

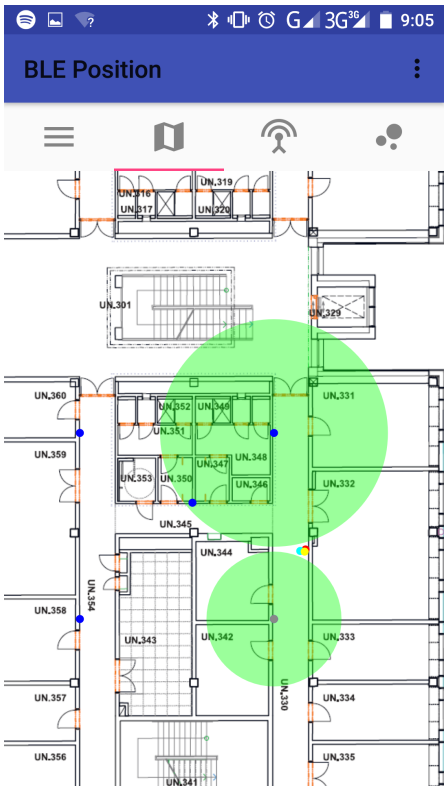
Cíl testu

Cílem testu bylo zjistit, jak se do vypočtené pozice promítne překážka v podobě rohu chodby. Dalším smyslem testu bylo otestovat, jak se zachová aplikace při výpadku všech majáků krom jediného.

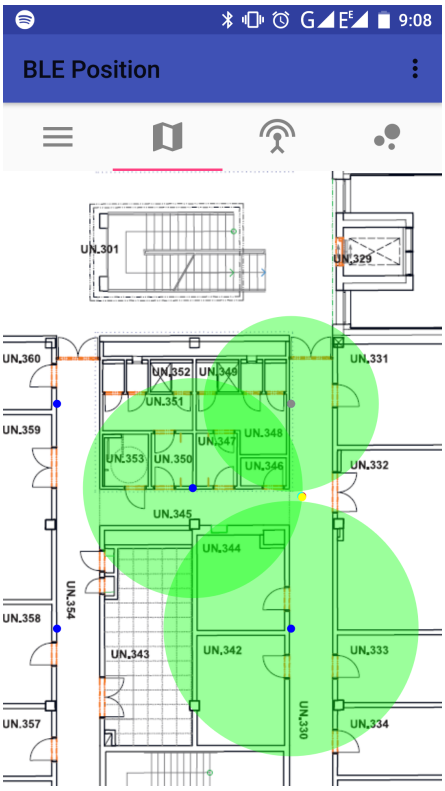
Výsledky testu



Obrázek 9.2: Šíření signálů v případě polovičního počtu majáků. *Modré* body jsou majáky, *světle modrý* bod je pozice nejmenších čtverců.

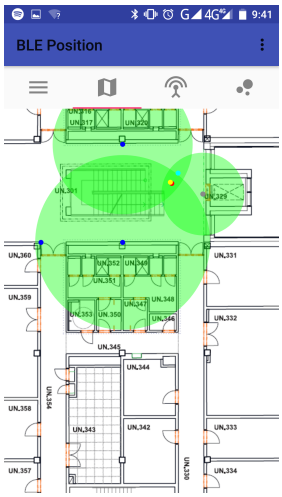


(a) Výpadek středového majáku kvůli rohu chodby.

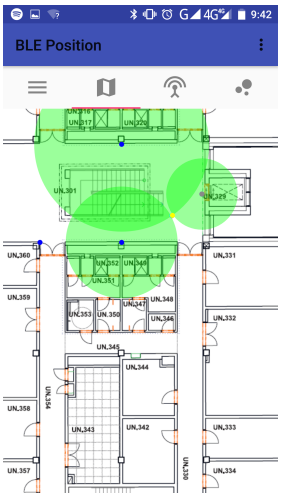


(b) Dostupný středový maják při čistém výhledu.

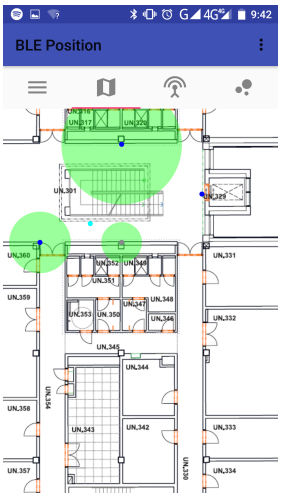
Obrázek 9.3: Testováhí H chodby. *Modré* body jsou majáky, *světle modrý* bod je pozice nejmenších čtverců.



(a) V blízkosti shodiště měla apliakce vysokou přesnost.



(b) V rozích místností byla přesnost daleko nižší kvůli absenci majáků.



(c) Vliv multipath efektu – apliakce detekuje 3 majáky, ale jejich vypočtený dosah se neprotnul.

Obrázek 9.4: Náhled testu v úzké chodbě. *Modré* body jsou majáky, *světle modrý* bod je pozice získaná metodou nejmenších čtverců, *žlutý* bod je metody váženého centroidu.

V bočních chodbách se dvěma majáky byla přesnost přibližně tři metry. Ve spojovací chodbě byl často dostupný právě pouze středový maják, protože ostatní byly odstíněny stěnami. Vliv stínění zachycují obrázky 9.3a a 9.3a. Ve chvíli, kdy byl dostupný pouze středový maják, přepnula se aplikace z algoritmu nejmenších čtverců na primitivní algoritmus nejbližšího majáku, jehož souřadnice byly označeny jako aktuální pozice.

Dále nastával stav, kdy byla vypočtená pozice umístěna do stěny nebo do místnosti za ní. Při pohybu nastává tento jev pouze dočasně, nicméně je patrné, že znatelným vylepšením aplikace by bylo nasazení nedostupných zón, kde by vypočtená poloha být nemohla.

9.2.4 Otevřený prostor se schodištěm

Test využívá konfigurace ze souboru na CD (`/test_cases/json/beaconsOpen.txt`). Mapový podklad si aplikace stáhne sama.

Cíl testu

Cílem testu je ověření chování v rozlehlejších prostorách jako například chodbě kolem schodiště. Zde je více prostoru a jedinou překážkou jsou schody uprostřed chodby.

Výsledku testu

Přesnost pozice kolísala mezi třemi až pěti metry v závislosti na části chodby. K dobrému pokrytí prostoru scházely alespoň dva majáky. V prostoru blízko

schodiště byla poloha velmi přesná, naopak nejhorší byla v rozích místnosti u výtahu, kde majáky nebyly (viz obrázky 9.4a, 9.4b, 9.4c). Nicméně i v tomto případě aplikace ve spojení s rozhledem na prostor poskytovala dostatek informací o poloze.

9.2.5 Test více pater

Tento test využívá vstupní soubor z CD `/test_cases/json/beaconsFloors.txt` a mapové podklady si opět stáhne z internetu.

Cíl testu

Cílem není přesnost určení pozice na jednotlivých patrech, ale naopak získání údajů o tom, jak rychle aplikace rozpozná změnu patra. Z obrázku je patrné, že majáky na patrech byly záměrně umístěny až za schody, aby zde byl i element překážky. Aplikace by měla na změnu patra reagovat změnou mapového podkladu a přenačtením majáků. Majáky z jiných pater by tedy neměly být viditelné.

Výsledky testu

Výsledky toho testu byly velmi uspokojivé. Aplikace rozpoznala změnu patra už během chůze po schodech, nebylo tedy nutné na patře na nic čekat. Mapový podklad byl také náležitě změněn. Vzhledem ke konfiguraci je pro každé patro využit stejný mapový podklad, liší se pouze tím, že na jednou z nich jsou uvedena jména vyučujících.

Závěr

Cílem práce bylo analyzovat dostupné metody pro určování polohy uvnitř budov a výběru vhodných metod na základě této analýzy. Závěry analýzy bylo nutné ověřit experimentálně, tedy implementací aplikace pro platformu Android.

Prvotní průzkum založený na již existujících pracích ukázal, že z mnoha dostupný metod jich má značná část vážné nedostatky (viz tabulka 3.2) v podobě přesnosti, potřeby speciálního hardwaru, vysokých nákladů na změnu nebo vybudování systému na straně poskytovatele.

Na základě provedené analýzy byla zvolena metoda trilaterace skrze BLE majáky. Pro tuto metodu nepotřebuje uživatel žádný dodatečný hardware, pouze svůj telefon či tablet, který disponuje technologií BLE. Metoda spoléhá na vybavení, kterým dnes disponuje drtivá většina mobilních zařízení na trhu – Bluetooth. Jiné metody vyžadují např. magnetometr nebo gyroskop, který se zejména na levnějších zařízeních nemusí vyskytovat.

Součástí implementace bylo i vytvoření vlastních BLE majáků z důvodu snížení nákladů. Komerční majáky se pohybují kolem hranice 50 USD. Maják sestrojený pro účely této práce stojí, včetně krytu vytištěného na 3D tiskárně (viz tabulka 5.1), 8 USD. Pokud by mohl být maják připojen k elektrické síti budovy a mohl být bez obalu (např. skrytý v obložení stropu), klesne jeho cena na 3 USD. Během implementace se ukázalo, že existují dvě provedení použitého BLE hardwaru. BLE vysílač, který byl použit pro tuto práci měl chudší instrukční sadu a po přepnutí na uspornější režim odebíral stále stejně energie, protože přebytek vyzářil v podobě tepla. I tento chudší klon však fungoval výborně a jediným problémem tedy byla snížená životnost baterií na 4 dny provozu.

Při testování majáků a aplikace se podařilo dosáhnout přesnosti 3 metrů v závislosti na rozložení majáků. Testování bylo prováděno v budově NTIS a díky této přesnosti bylo vždy možné bezpečně najít hledanou místnost. Aplikace byla schopna bezpečně rozpoznat i změnu patra v případě přesunu po schodech či opuštění výtahu.

Z testování vyplynulo, že pro přijatelnou přesnost 3 metrů je vhodné, aby od sebe majáky byly maximálně 7 metrů pokud na sebe mají přímý výhled. Pro lepší výsledky by tyto majáky neměly ležet na jedné přímce, čímž je možné rozpoznat pohyb pouze ve směru této přímky. Dle očekávání se nejlépe osvedčila metoda nelineárních nejmenších čtverců nejen kvůli přesnosti, ale díky robustnosti – metody založené na průnicích kružnic nefungují díky tomu, že se signály často nepřekrývají.

Aplikace je schopna jako mapový podklad využít výkres budovy ve formátu obrázku, není tedy nutné vytvářet žádné speciální plány.

Celé řešení ověřilo, že je možné vytvořit indoor navigaci bez nutnosti speciálního hardwaru pro uživatele a s nízkými náklady pro pořizovatele. Díky mobilitě majáků a rychlému vytvoření konfigurace je toto řešení možné nasadit např. i na krátkodobé akci v podobě festivalu nebo výstavy.

Prostor pro zlepšení vidím v implementaci zakázaných míst na mapě – tedy pomocí barev rozpoznat, že pozice není přistupná a přesunout pozici na nejbližší možnou. Za druhé přínosné řešení považuji implementaci minifikovaných URL adres jako jmen majáků, díky čemuž by aplikace mohla stahovat potřebná data ve chvíli přiblížení k majáku, např. ordinační hodiny doktorů na aktuálním patře v nemocnici.

Seznam použité literatury

[1] Margaret Rouse. Latitude and Longitude. http://whatis.techtarget.com/definition/latitude-and-longitude, červen 2015.
Čerpáno: 01.12.2016.

[2] Ordnance Survey. A guide to coordinate systems in Great Britain. http://www.ordnancesurvey.co.uk/docs/support/guide-coordinate-systems-great-britain.pdf, březen 2015.
Čerpáno: 17.1.2017.

[3] Google. Location. https://developer.android.com/reference/android/location/Location.html.
Čerpáno: 17.1.2017.

[4] Geodetické základy. http://gis.zcu.cz/studium/gen1/html/ch03.html.
Čerpáno: 17.1.2017.

[5] Trilateration Exercise. http://www.gps.gov/multimedia/tutorials/trilateration/.
Čerpáno: 17.1.2017.

[6] Anja Bekkelien. Bluetooth Indoor Positioning. https://pdfs.semanticscholar.org/1919/037541b4a84d0b5a6dd4d425c0891282ae8f.pdf, březen 2012.
Čerpáno: 17.1.2017.

[7] Deepak Vasisht, Swarun Kumar, and Dina Katabi. Decimeter-Level Localization with a Single WiFi Access Point. https://www.usenix.org/system/files/conference/nsdi16/nsdi16-paper-vasisht.pdf, březen 2016.
Čerpáno: 17.1.2017.

[8] Manikanta Kotaru, Kiran Joshi, Dinesh Bharadia, and Sachin Katti. SpotFi: Decimeter Level Localization Using WiFi. http://conferences.sigcomm.org/sigcomm/2015/pdf/papers/p269.pdf, srpen 2015.
Čerpáno: 17.1.2017.

[9] Pierluigi Gallo, Stefano Mangione, and Giampiero Tarantino. WIDAR: bistatic WI-fi Detection And Ranging for off-the-shelf devices. https://pdfs.semanticscholar.org/b064/f50f8ab1701964140ed94b0cacf0f4ce82cf.pdf, 2013.
Čerpáno: 17.1.2017.

[10] Lian Chen, Ling Pei, Heidi Kuusniemi, and Ruizhi Chen. Bayesian Fusion for Indoor Positioning Using Bluetooth Fingerprints. https://www.researchgate.net/publication/234026702_Bayesian_Fusion_for_Indoor_Positioning_Using_Bluetooth_Fingerprints, srpen 2012.
Čerpáno: 17.1.2017.

[11] Chris Hide, Terry Moore, and Tom Botterill. Low Cost Vision-Aided IMU for Pedestrian Navigation. https://pdfs.semanticscholar.org/3524/35c3794b0385c318c7a468f75e3bd7ceb66b.pdf, 2011.
Čerpáno: 17.1.2017.

[12] Fabian Hoflinger, Joachim Hoppe, Rui Zhang, Alexander Ens, Leonhard Reindl, Johannes Wendeberg, and Christian Schindelhauer. Acoustic Indoor-Localization System for Smart Phones. https://www.hindawi.com/journals/ijno/2015/694695/, 2015.
Čerpáno: 17.1.2017.

[13] Ruoxi Jia, Ming Jin, and Costas J. Spanos. SoundLoc: Acoustic Method for Indoor Localization without Infrastructure. https://arxiv.org/pdf/1407.4409v1.pdf, červenec 2014.
Čerpáno: 17.1.2017.

[14] Milan Herrera Vargas. INDOOR NAVIGATION USING BLUETOOTH LOW ENERGY (BLE) BEACONS. https://www.theseus.fi/bitstream/handle/10024/105619/Herrera%20Vargas_Milan.pdf?sequence=1, 2016.
Čerpáno: 17.1.2017.

[15] Current GLONASS and GPS status. https://www.glonass-iac.ru/en/GLONASS/index.php, prosinec 2016.
Čerpáno: 01.12.2016.

[16] GNSS Signal. http://www.navipedia.net/index.php/GNSS_signal, březen 2014.
Čerpáno: 23.01.2017.

[17] What is GPS? http://www.gps.gov/systems/gps/, září 2016.
Čerpáno: 01.12.2016.

[18] BeiDou General Introduction. http://www.navipedia.net/index.php/BeiDou_General_Introduction, září 2014.
Čerpáno: 01.12.2016.

[19] Galileo General Introduction. http://www.navipedia.net/index.php/GALILEO_General_Introduction, květen 2016.
Čerpáno: 01.12.2016.

[20] GLONASS History. https://www.glonass-iac.ru/en/guide/index.php.
Čerpáno: 23.01.2017.

[21] An intuitive approach to the GNSS positioning. http://www.navipedia.net/index.php/An_intuitive_approach_to_the_GNSS_positioning, květen 2014.
Čerpáno: 23.01.2017.

[22] GPS Signal Plan. http://www.navipedia.net/index.php/GPS_Signal_Plan, leden 2014.
Čerpáno: 23.01.2017.

[23] GLONASS Signal Plan. http://www.navipedia.net/index.php/GLONASS_Signal_Plan, červen 2012.
Čerpáno: 23.01.2017.

[24] GALILEO Signal Plan. http://www.navipedia.net/index.php/GALILEO_Signal_Plan, leden 2014.
Čerpáno: 23.01.2017.

[25] BeiDou Signal Plan. http://www.navipedia.net/index.php/BeiDou_Signal_Plan, květen 2014.
Čerpáno: 23.01.2017.

[26] Wentao Zhang and Yang Gao. Proposed GNSS Navigation Messages for Improved Performance. http://www.navipedia.net/index.php/BeiDou_Signal_Plan, říjen 2015.
Čerpáno: 24.01.2017.

[27] Viktor Vašina. GPS Framework na platformě Android. https://otik.uk.zcu.cz/bitstream/handle/11025/13496/bakalarka.pdf?sequence=1, květen 2014.
Čerpáno: 24.01.2017.

[28] The Google Maps Geolocation API. https://developers.google.com/maps/documentation/geolocation/intro.
Čerpáno: 24.01.2017.

[29] Bluetooth Core Specification. https://www.bluetooth.com/specifications/bluetooth-core-specification, 2016.

[30] Bluetooth Basics. https://learn.sparkfun.com/tutorials/bluetooth-basics, 2013.
Čerpáno: 01.12.2016.

[31] Greg Sterling. Magnetic Positioning. http://www.indooratlas.com/wp-content/uploads/2016/03/magnetic_positioning_opus_jun2014.pdf, červen 2014.
Čerpáno: 01.12.2016.

[32] J.-O. Nilsson, J. Rantakokko, P. Händel, I. Skog, M. Ohlsson, and K.V.S. Hari. Accurate Indoor Positioning of Firefighters using Dual Foot-mounted Inertial Sensors and Inter-agent Ranging. https://www.researchgate.net/publication/263022856_Accurate_indoor_positioning_of_firefighters_using_dual_foot-mounted_inertial_sensors_and_inter-agent_ranging, květen 2015.
Čerpáno: 01.12.2016.

[33] Zenshin Tech Shop. https://zenshin-tech.com/.
Čerpáno: 01.12.2016.

[34] Femtoduino - IMUduino. https://femto.io/products/imuduino.
Čerpáno: 01.12.2016.

[35] Texas Instruments. http://www.ti.com/tool/TIDC-CC2650STK-SENSORTAG.
Čerpáno: 01.12.2016.

[36] Eddystone. https://github.com/google/eddystone, 2016.
Čerpáno: 01.12.2016.

[37] WHAT IS IBEACON? A GUIDE TO BEACONS. http://www.ibeacon.com/what-is-ibeacon-a-guide-to-beacons/.
Čerpáno: 01.12.2016.

[38] AltBeacon. http://altbeacon.org/, 2015.
Čerpáno: 19.03.2017.

[39] Android Beacon Library. https://github.com/AltBeacon/android-beacon-library, březen 2017.
Čerpáno: 20.03.2017.

[40] Understanding RSSI. http://www.metageek.com/training/resources/understanding-rssi-2.html, 2005.
Čerpáno: 20.05.2017.

[41] Proč je síla signálu telefonu uváděna v záporných hodnotách a co vyjadřuje? https://www.svetandroida.cz/sila-signalu-201503/, březen 2015.
Čerpáno: 20.05.2017.

[42] An Analysis of the Accuracy of Bluetooth Low Energy for Indoor Positioning Applications. https://www.cl.cam.ac.uk/~rmf25/papers/BLE.pdf, září 2014.
Čerpáno: 20.05.2017.

[43] Library for 1D and 2D barcode scanning. https://github.com/zxing/zxing, 2017.
Čerpáno: 20.05.2017.

[44] Subsampling Scale Image View. https://github.com/davemorrissey/subsampling-scale-image-view, 2017.
Čerpáno: 20.05.2017.

[45] TRILATERATION-BASED LOCALIZATION ALGORITHM FOR ADS-B RADAR SYSTEMS. https://etda.libraries.psu.edu/files/final_submissions/8381, 2013.
Čerpáno: 20.05.2017.

[46] Nelder-Mead Method. http://www.jasoncantarella.com/downloads/NelderMeadProof.pdf, 2004.
Čerpáno: 20.05.2017.

[47] Xamarin. https://www.xamarin.com/, 2017.
Čerpáno: 20.06.2017.

[48] Apache Commons Math. http://commons.apache.org/proper/commons-math/, 2017.
Čerpáno: 20.06.2017.

[49] Apache Cordova. https://cordova.apache.org/, 2017.
Čerpáno: 20.06.2017.

[50] Android Studio. https://developer.android.com/studio/index.html, 2017.
Čerpáno: 20.06.2017.

[51] Picasso Library. http://square.github.io/picasso/, 2017.
Čerpáno: 20.06.2017.

Seznam obrázků

Seznam tabulek

Seznam použitých zkratek

2D	dvourozměrný
3D	trojrozměrný
API	Aplication Programming Interface
apod.	a podobně
atd.	a tak dále
atp.	a tak podobně
BLE	Bluetooth Low Energy
BTS	Base Transceiver Station vysílač/přijímač
CELL ID	unikátní identifikátor BTS
DIY	Do It Yourself
GPS	Global Positioning System
GSM	Groupe Spécial Mobile
JSON	JavaScript Object Notation
mj.	mimo jiné
MT	mobilní telefon
např.	například
obr.	obrázek
OS	operační systém
POI	Point Of Interest
popř.	popřípadě
TA	Timing Advance
QR	Quick Response
tj.	to jest
TS	Time Slot
tzv.	takzvaný
UI	user interface
URL	Uniform Resource Locator
USD	United States dollar

A. Přílohy

- bletest
  - MainActivity.java — hlavní aktivita aplikace
  - Settings.java — aktivita pro nastavení aplikace
  - customViews — upravené widgety
    - BeaconAdapter.java — widget položky seznamu majáků
    - IndoorView.java — zobrazení indoor mapy a bodů na ní
    - PaintedPointAdapter.java — widget položky seznamu algoritmů
  - entities — základní objekty pro trilaterační algoritmy a zobrazování
    - Point.java — bod v rovině
    - PaintedPoint.java — bod s vlastností barvy
    - SignalCircle.java — pozice majáku a jeho dosah
    - Beacon.java — BLE maják
    - InitDataBlock.java — contejner pro výchozí data aplikace
  - utils
    - PointOperations.java — metody pro práci s třídou Point
    - FileUtils.java — práce se soubory
  - trilaterationAlgorithms — trilaterační algoritmy
    - TrilaterationAlgorithm.java — rozhraní trilateračních algoritmů
    - SimpleCentroid.java — centroid průniků kružnic
    - WeightedCentroid.java — vážený centroid
    - SingleBeaconLocator.java — nejblížší maják
    - NonLinearLeastSquares.java — nejmenší čtverců

B. Adresářová struktura CD

- /
  - `src` ..... zdrojové soubory aplikace
    - `BLETest` ..... projekt pro Android Studio
  - `doc` ..... diplomová práce
    - `tex` ..... zdrojové soubory Tex
    - `thesis.pdf` ..... dokument diplomové práce
  - `test_cases` ..... data testovacích scénářů
    - `json` ..... configurační soubory majáků
    - `maps` ..... mapové podklady pro testy
    - `data_generator` ..... generátor configuraci v Json a QR
      - `IndoorNavConfigurator.exe` . program pro generování configurací
      - `IndoorNavConfigurator` ..... projekt generátor pro Visual Studio
  - `3D_print` ..... model obalu majáku pro 3D tisk
  - `poster` ..... postery

C. Uživatelská dokumentace

Tato sekce popisuje ovládání mobilní aplikace. Generování vstupních souborů a konfiguraci majáků lze nalézt v sekci 9.

C.1 Instalace aplikace

Aplikace není dostupná skrze obchod Google Play. je tedy nutné instalovat ji ze souboru. Stačí pouze zkopírovat aplikaci do mobilního zařízení pomocí USB kabelu. Soubor aplikace je dostupný na CD (`/test_cases/ble_navi.apk`).

Dále je nutné v nastavení telefonu otevřít sekci *Zabezpečení* a povolit instalaci aplikací z neznámých zdrojů. Nyní již stačí pouze najít soubor aplikace a otevřít jej.

C.2 Spuštění

Aplikace se při prvním psuštění zobrazí v běžném režimu. Který obsahuje dvě záložky:

- záložku pro získání informací o majácích a mapových podkladech
- záložku pro zobrazení mapy

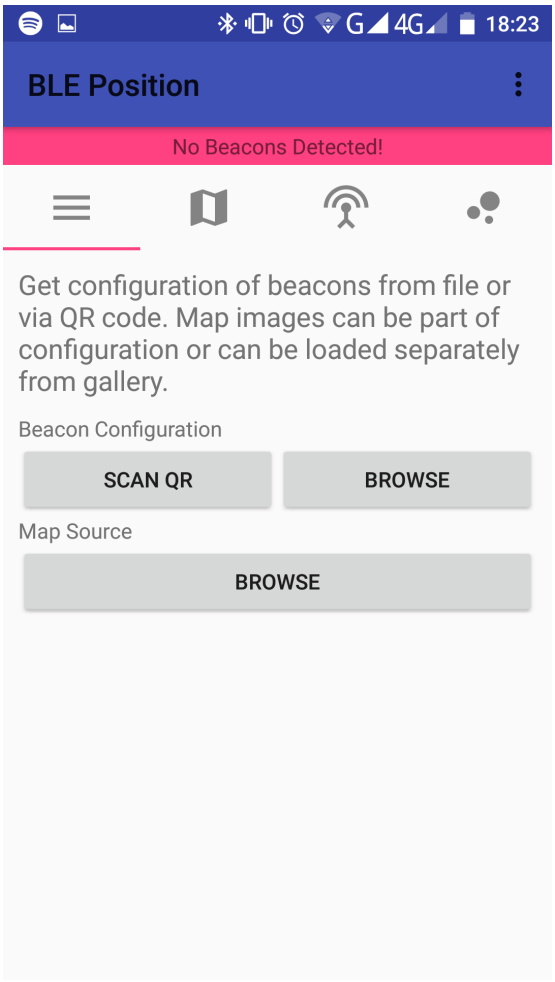
Vstupní data lze načíst scanováním QR kódu nebo ze souboru v úložišti. Pro výber souboru z úložiště se uživateli otevře správce souborů, pomocí kterého bude moci soubor vybrat.

Pokud nejsou informace o mapových podkladech součástí konfigurace (QR kódu nebo souboru), je možné mapový podklad manuálně vložit pomocí tlačítka v sekci *Map Sources* viz obrázek C.1.

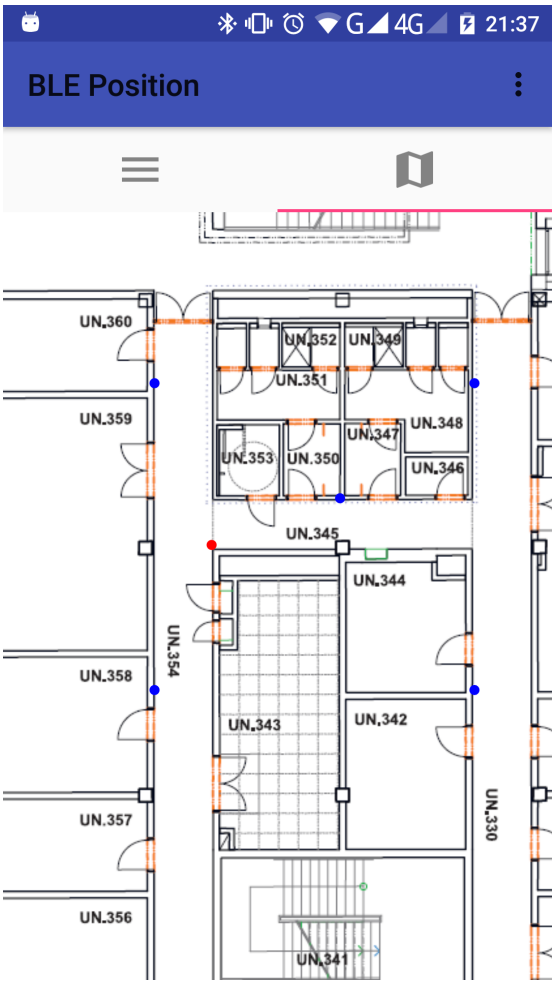
Po uspěšném získání informací o majácích bude v druhé záležce k dispozici mapa, která zobrazuje modrými body pozice majáků a červeným bodem vaši pozici (obr. C.2).

C.2.1 Offline režim

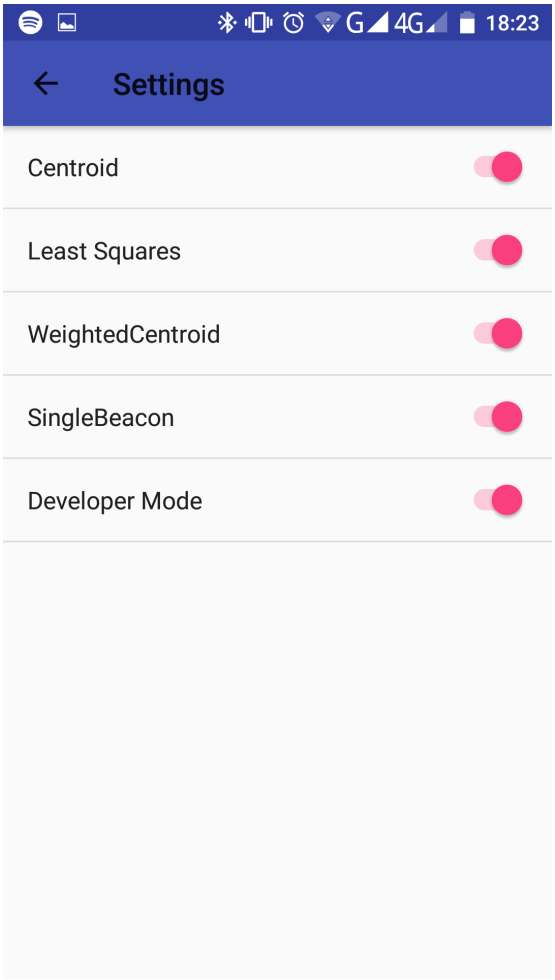
Pokud se žádný z majáků 5 sekund neohlásí, přepne se aplikace do offline režimu, což je patrné z varovného pruhu v horní části obrazovky (jako na obrázku C.1). V tu to chvíli můžete prohlížet mapu, ale vaše poloha není aktualizována. Aplikace se vráti zpět do běžného režimu ve chvíli, kdy zaregistruje signál od libovolného majáku.



Obrázek C.1: Menu aplikace bez detekovaných majáků.



Obrázek C.2: Zobrazení mapy v uživatelském režimu.



Obrázek C.3: Možnosti nastavení aplikace.

C.3 Nastavení

V pravém horním rohu aplikace je k dispozici kontextové menu, v které umožňuje otevřít nápovědu nebo nastavení aplikace (obr. C.3).

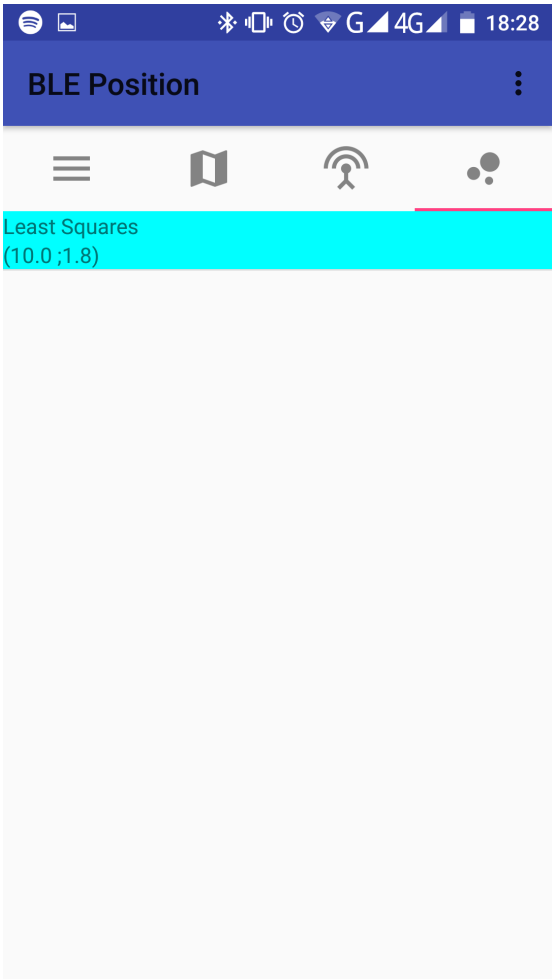
V nastavení aplikace je možné zapnout vývojářský režim (*developer mode*), který zpřístupní další funkce aplikace. Zároveň je zde možné vybrat z několika lokalizačních algoritmů, ovšem jejich nastavení je zohledněno poze ve vývojářském režimu.

C.4 Vývojářský režim

Tento režim zpřístupňuje uživateli další lokalizační algoritmy, které je možné využít pro testování vlastností BLE signálu. Na mapě jsou pomocí barevných bodů zobrazeny pozice získané různými algoritmy (obr. 9.4b, C.4).

Aplikace se ve vývojářském režimu rozšíří o dvě další záložky:

- sledování stavu majáků, tedy vzdálenost, jméno, síla signálu atd.



Obrázek C.4: Seznam použitých lokačních algoritmů.

- seznam vypočtených pozic – seznam je barevně rozlišen a barvy položek odpovídají barvám bodů na mapě

Barva položek seznamu majáků má svůj význam – dostupné majáky jsou zobrazeny zeleně. Ty které se déle jak 5 sekund neohlásily jsou červené a nejsou využívaný pro výpočet pozice.

Můžete si také povšimnout, že kolem majáků se na mapě vykreslují zelené kruhy. Tyto kruhy zobrazují vypočtenou vzdálenost od tohoto majáku. Menší poloměr tedy znamená, že jste majáku blíž. Takto se zobrazují poze tři nejbližší majáky, které jsou použité pro výpočet vaší pozice.

C.5 Více pater

Aplikace umožňuje lokalizace a zobrazení polohy napříč více patry. Jako mapový podklad je využit vždy plán aktuálního patra, na kterém jsou zobrazeny pouze majáky z aktuálního patra. Uživatel tedy zaznamená změnu, až když opustí patro.

Obrázek C.5: Seznam sledovaných majáků.

C.6 Ukládání posledního nastavení

Pro větší pohodlí si aplikace mezi spuštěními pamatuje pozice majáků a případně i informace o mapových podkladech. Aplikace si ukládá i nastavení povolených algoritmů a vývojářského režimu. Pokud tedy budete potřebovat určit pozici v budově, kde jste hledali pozici při posledním spuštění aplikace, není nutné znovu hledat soubor nebo QR kód.