Katedra informatiky
Přírodovědecká fakulta
Univerzita Palackého v Olomouci

DIPLOMOVÁ PRÁCE

Lokátor stanu pro nevidomé

2023

Bc. Marek Schiller

Vedoucí práce:
RNDr. Martin Trnečka, Ph.D.

Studijní program: Aplikovaná informatika,
Specializace: Vývoj software

Bibliografické údaje

Autor:	Bc. Marek Schiller
Název práce:	Lokátor stanu pro nevidomé
Typ práce:	diplomová práce
Pracoviště:	Katedra informatiky, Přírodovědecká fakulta, Univerzita Palackého v Olomouci
Rok obhajoby:	2023
Studijní program:	Aplikovaná informatika, Specializace: Vývoj software
Vedoucí práce:	RNDr. Martin Trnečka, Ph.D.
Počet stran:	61
Přílohy:	elektronická data v úložišti katedry informatiky
Jazyk práce:	český

Bibliographic info

Author:	Bc. Marek Schiller
Title:	Tent locator for the blind
Thesis type:	master thesis
Department:	Department of Computer Science, Faculty of Science, Palacký University Olomouc
Year of defense:	2023
Study program:	Applied Computer Science, Specialization: Software Development
Supervisor:	RNDr. Martin Trnečka, Ph.D.
Page count:	61
Supplements:	electronic data in the storage of department of computer science
Thesis language:	Czech

Anotace

V této práci si přiblížíme problematiku lokalizace stanu nevidomým. Jednotlivé lokalizační technologie budou porovnány. Porovnáme dostupná konkurenční řešení. Následně práce pojednává o implementaci mobilní aplikace a vytvoření příslušného hardware umožnující nevidovým nalezení stanu.

Synopsis

In this paper, we address the issue of tent localization for the blind. Different localization technologies will be compared. We will compare available competing solutions. Subsequently, the thesis discusses the implementation of a mobile application and the creation of suitable hardware enabling the blind to locate the tent.

Klíčová slova: lokalizace stanu; Bluetooth; BLE; UWB; ESP32; Flutter

Keywords: tent locator; Bluetooth; BLE; UWB; ESP32; Flutter

Především bych chtěl poděkovat panu RNDr. Martinu Trnečkovi, Ph.D. za podnětné rady, věcné připomínky a vstřícnost při konzultacích a vedení práce.

Odevzdáním tohoto textu jeho autor/ka místopřísežně prohlašuje, že celou práci včetně příloh vypracoval/a samostatně a za použití pouze zdrojů citovaných v textu práce a uvedených v seznamu literatury.

Obsah

Seznam obrázků

Seznam tabulek

Seznam zdrojových kódů

1 Úvod a motivace

Cílem této diplomové práce je návrh lokátoru, jenž by měl usnadnit nevidomému nalezení svého stanu. Podnětem k napsání práce byla možnost pomoci kamarádovi, který se mnou sdílel své špatné zkušenosti s podobnými pomůckami pro nevidomé, jež by měly pomoci lokalizovat stan. Dané pomůcky primárně nevyhovují z hlediska krátkého dosahu signálu, příliš hlasité či tiché signalizaci při hledání stanu, především však fungují na základě aplikací, které nejsou uzpůsobené nevidomým. Rozhodl jsem se navrhnout takovou kompenzační pomůcku, která by řešila výše uvedené problémy.

V teoretické části práce nastíním rozdíl mezi nativním a multiplatformním vývojem mobilních aplikací. Na základě průzkumu míry používání jsem popsal dva nejpoužívanější operační systémy iOS a Android. Stejně jsem vycházel u výběru dvou nejpoužívanějších frameworků. Třetí multiplatformní framework byl zvolen na základě mé osobní zkušenosti. Dále se zmiňuji o variantě – progresivní webová aplikace, to znamená webová aplikace, která napodobuje chování mobilní aplikace. V neposlední řadě jsem se věnoval problematice používání mobilního telefonu a mobilních aplikací nevidomými.

Další kapitola je věnována popisu a porovnání lokačních technologií. Porovnání mělo za cíl vybrat nejvhodnější technologii pro zamýšlený projekt. Většina porovnávaných lokačních technologií je samostatně nevyhovující, jelikož nedokáže určit směr, pouze vzdálenost. Z tohoto důvodu bylo nutné udělat průzkum a porovnat metody určující směr, kterým jsem se věnoval ve čtvrté kapitole.

Dále v práci uvádím přehled existujících produktů, které mají za úkol pomoci uživateli s nalezením věci (nejčastěji se jedná o pomůcky určené pro nalezení klíčů, peněženky, cestovních tašek).

V praktické části práce představím použité součástky, ze kterých jsem vytvořil prototyp kompenzační pomůcky. Dále se věnuji implementační části jednotlivých komponentů projektu. Mobilní aplikaci napsané ve Flutter, komunikující s ESP32 UWB Pro, která je hlavní částí kompenzační pomůcky. V jedné podkapitole navrhnu možné budoucí rozšíření pomůcky. Na závěr práce celý projekt otestuji a zhodnotím výsledek.

2 Vývoj mobilních aplikací

V této kapitole vás seznámím s úvodem do mobilních aplikací, nastíním rozdíl mezi nativním a multiplatformním vývojem mobilních aplikací. Na základě míry používaní jsou popsány dva nejpoužívanější operační systémy iOS a Android. Budou popsány jednotlivé frameworky umožnující vývoj mobilních aplikací. Dále se zmíním o variantě – progresivní webová aplikace, to znamená webová aplikace, která napodobuje chování mobilní aplikace. V neposlední řadě jsem se věnoval problematice používání mobilního telefonu a mobilních aplikací nevidomými uživateli.

Mobilní aplikace by měla uživateli co možná nejjednodušeji a nejrychleji vyřešit určitý požadavek, zjednodušit rutinní postupy a urychlit práci. Na začátku vývoje je velmi důležité rozumět daným požadavkům, které má aplikace řešit a pro jaké uživatele je určena. Od toho se odvíjí výběr vhodné technologie pro aplikaci. Je důležité, aby aplikace běžela plynule a měla co možná nejjednodušší a intuitivní uživatelské prostředí (UI). U aplikací určených pro širokou veřejnost je dobrý návrh UI klíčový a často rozhoduje o úspěšnosti aplikace na trhu. U aplikací určených pro specifickou skupinu lidí s určitým postižením má zásadní význam pro zajištění přístupnosti, použitelnosti a uspokojení potřeb těchto uživatelů. Správně navržené UI může výrazně zlepšit jejich uživatelský zážitek a umožnit jim plnohodnotné využívání mobilních aplikací.

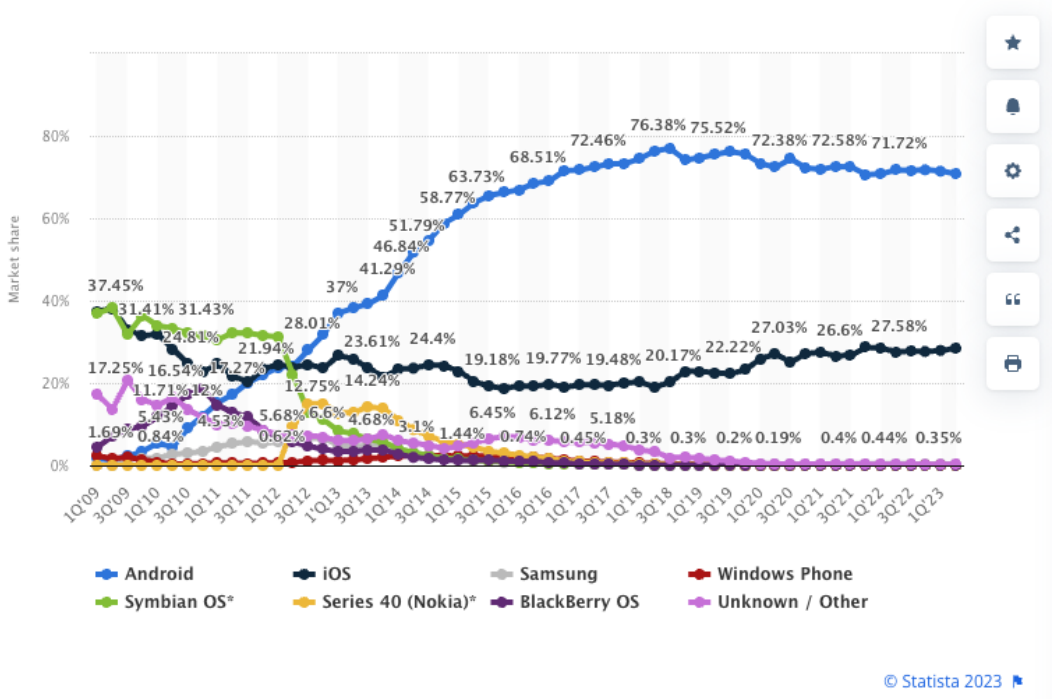
Specifické skupiny lidí, jako jsou lidé se zrakovým nebo sluchovým postižením, motorickými obtížemi nebo kognitivními omezeními, často čelí různým překážkám při používání běžných mobilních aplikací. Navržení přístupného UI je klíčovým faktorem, který jim umožňuje překonat tyto bariéry a aktivně se zapojit do digitálního světa mobilních aplikací.

Důkladná analýza potřeb a schopností cílové skupiny je nezbytná pro úspěšný návrh UI. Například pro uživatele se zrakovým postižením je důležité zahrnout funkce hlasového výstupu nebo technologie haptické zpětné vazby, které jim pomáhají přístupněji interagovat s aplikací. Pro uživatele se sluchovým postižením mohou být důležité vizuální indikátory a alternativní textové popisky pro zvuková upozornění.

Důsledný testovací proces s uživateli z cílové skupiny je nezbytný pro ověření přístupnosti a efektivnosti navrženého UI. Zpětná vazba od uživatelů je cenná pro další zlepšování a optimalizaci UI a celé aplikace.

V dnešní době se na trhu s mobilními telefony vyskytují v převážné míře dva operační systémy, jak je vidět na obrázku 1. iOS od firmy Apple, který je určen přímo pro hardware, je taktéž vyráběný firmou Apple. Android podporuje zařízení od vícero výrobců. Operační systém Android je open-source, což znamená, že jeho zdrojový kód je veřejně dostupný a může být upraven. Často si každý výrobce upraví svým požadavkům Nicméně tyto systémy existují ve mnoha různých verzích. Samotná zařízení se liší ve velikosti, funkcích a interních komponentech. To je právě to, co dělá vývoj mobilních aplikací složitým a zdůrazňuje důležitost správného výběru metody. Pro správné rozhodnutí je nezbytné se seznámit

s detaily různých stylů vývoje a frameworků.



Obrázek 1: Celosvětový podíl mobilních operačních systémů [1].

2.1 Nativní vývoj

Při této metodě je aplikace vytvářena odděleně pro každou platformu samostatně. To znamená, že je potřeba napsat stejný kód dvakrát v různých programovacích jazycích a použít odlišná vývojová prostředí a nástroje. Navíc je třeba vzít v úvahu, že Apple neumožňuje vývoj nativních aplikací na jiném operační systém než macOS.

Pokud by bylo cílem vytvořit aplikaci, která by měla běžet na obou nejpoužívanějších platformách (což je většinou předpokladem), společnost by se musela vybavit dvěma oddělenými vývojovými týmy, z nichž každý by se specializoval na jednu platformu. To by znamenalo zajištění potřebného odlišného vybavení pro každý tým. Tímto způsobem by vývoj trval nejen dlouho, ale byl by i nákladný.

Výsledná aplikace by měla být volně k dispozici uživatelům, pokud by se nejednalo o interní aplikaci firmy, která nechcete dát volně dostupnou aplikaci uživateli. To už záleží na povaze aplikace a politice firmy. Android a iOS používají své platformy k této distribuci. Pro iOS to je App Store, pro Android Play Store (dříve Google Play). Kvůli těmto platformám si firmy mohou udržet kontrolu nad aplikacemi, které před vydáním otestují a zkontrolují. Android není tak striktní a aplikace lze získat jednoduše i z jiného zdroje. Apple je v tomto ohledu velmi přísný, kontroluje každou aplikaci i aktualizaci, než je uvolní do App Store. Tato kontrola trvá i několik dní. V tomto procesu je hlavně ověřováno, zda produkt

skutečně splňuje to, co vývojář aplikace uvádí v popisu a zda splňuje všechny bezpečnostní normy. Tento přístup je vhodný pro specifické typy projektů:

- Projekty, kde není potřeba, aby výsledná aplikace běžela na obou platformách.
- Projekty, kde je kladen důraz na výkon aplikace, nebo je to z podstaty požadavku nutnost.

2.1.1 iOS

Aplikace pro iOS mohou být napsány pomocí jazyků Swift a Objective-C, iOS SDK (iOS Software Development Kit), dříve iPhone SDK, je balíček nástrojů umožňující vývoj aplikací. Nástroje umožňují vývojáři přistupovat k různým funkcím a službám iOS zařízení jako jsou hardwarové a softwarové atributy [14].

iOS SDK je k dispozici zdarma ke stažení pro uživatele počítačů Mac od společnosti Apple. Toto SDK není dostupné pro počítače s jiným operačním systémem. Vývojové prostředí Xcode je opět volně dostupné pro uživatele Macu. Společnost Apple si vybudovala svůj vlastní ekosystém, ve kterém se vzájemně integrují a optimalizují její produkty.

2.1.2 Android

Pro vývoj aplikací pro Android existují různé programovací jazyky jako Kotlin, Java a C++. Pomocí nástrojů Android SDK lze zkompilovat váš kód spolu s případnými daty a soubory se zdrojovým kódem do souboru APK (Android Application Package). Balíček pro Android, označovaný také jako soubor APK, je archivní soubor s příponou .apk, který obsahuje veškerý obsah aplikace pro Android, který je potřebný pro její správné fungování. Tento soubor je používán zařízeními s operačním systémem Android pro instalaci aplikace [13].

Pro vývoj pro platformu Android je nezbytné mít k dispozici Android SDK a vhodné vývojové prostředí. Podle dokumentace se doporučuje používat vývojové prostředí Android Studio. Výhodou je, že tyto nástroje jsou dostupné na jakémkoliv počítačovém operačním systému.

2.2 Multiplatformní vývoj

Multiplatformní vývoj je metoda, která umožňuje vývoj na více platformách naráz pomocí jednoho kódu a v některých případech se používá stejný kód. Každý framework toto umožňuje trochu jiným způsobem. Základní myšlenka spočívá v tom, že ve srovnání s nativními aplikacemi, které přímo komunikují s operačním systémem a zařízením, multiplatformní aplikace přidávají abstraktní mezivrstvu, nazývanou také nativní obálka. Tato mezivrstva zajišťuje, že aplikace na mobilním systému se tváří jako nativní, zatímco umožňuje kódu přistupovat ke všem periferiím zařízení, jako je například kamera, určování polohy, gyroskop a další. Mezi základní výhody patří:

- Opětovné využití kódu: Multiplatformní vývoj umožňuje sdílet a opětovně využívat kód mezi různými platformami. To znamená, že vývojáři nemusí psát a spravovat oddělený kód pro každou platformu, což šetří čas a snižuje nároky na údržbu.
- Snížení nákladů: Vývoj aplikace pro více platforem je nákladný proces, protože vyžaduje oddělené týmy a zdroje pro každou platformu. Multiplatformní přístup může snížit náklady, protože je potřeba pouze jedno vývojové prostředí a tým se může zaměřit na vytvoření a údržbu jednoho kódu pro všechny platformy.
- Rychlejší nasazení: Multiplatformní vývoj umožňuje rychlejší nasazení aplikace na trh, protože vývojáři mohou pracovat na všech platformách současně a aktualizace se aplikují na všechny verze aplikace najednou.

Mezi základní nevýhody patří:

- Omezení funkcionalit: Multiplatformní frameworky mohou mít omezenou podporu pro některé specifické funkcionality dané platformy. Pokud je aplikace závislá na funkcích, které nejsou dobře podporovány multiplatformním přístupem, může být obtížné je implementovat nebo dosáhnout stejné úrovně výkonu.
- Zpoždění aktualizací: Při multiplatformním vývoji mohou aktualizace a opravy chyb trvat déle, protože změny musí být provedeny na všech podporovaných platformách. To může vést k zpoždění dodání oprav uživatelům a potenciální ztrátě jejich spokojenosti.
- Nutnost kompromisů: Vzhledem k rozdílným platformovým specifikacím a omezením může být nutné dělat kompromisy ve funkcionalitách, designu nebo výkonu aplikace. To může ovlivnit celkovou kvalitu a uživatelský zážitek.
- Složitější ladění a testování: Testování a ladění aplikace na více platformách může být náročné a časově náročné. Různé platformy mohou mít odlišné chování a problémy, které je třeba řešit.
- Zvýšená velikost aplikace: Kvůli sdílení kódu a závislostem může být multiplatformní aplikace větší ve velikosti než aplikace vyvinutá pouze pro jednu platformu. To může zvýšit čas stahování a nároky na paměť zařízení.

2.2.1 React Native

React Native je open source framework, který umožňuje vývojářům vytvářet aplikace pro Android a iOS pomocí Reactu a nativních možností platformy. S využitím React Native je možné používat JavaScript k přístupu k rozhraním API dané platformy a také k popisu vzhledu a chování uživatelského rozhraní pomocí

komponent React. Tento framework poskytuje svazky opakovaně použitelného a vnořitelného kódu, které umožňují efektivní vývoj aplikací [16].

Důležitou vlastností React Native je, že kód je kompilován do nativního jazyka používaného danou platformou. Tímto přístupem je zajištěna vysoká výkonnost a přístup ke všem nativním funkcím a rozhraním platformy. Velkou výhodou je také možnost vytvoření vlastního modulu v jednom z nativních jazyků a jeho integrace do projektu, pokud žádná z poskytovaných knihoven neobsahuje potřebnou funkcionalitu. Tím je umožněno rozšíření a přizpůsobení aplikace dle konkrétních potřeb a požadavků.

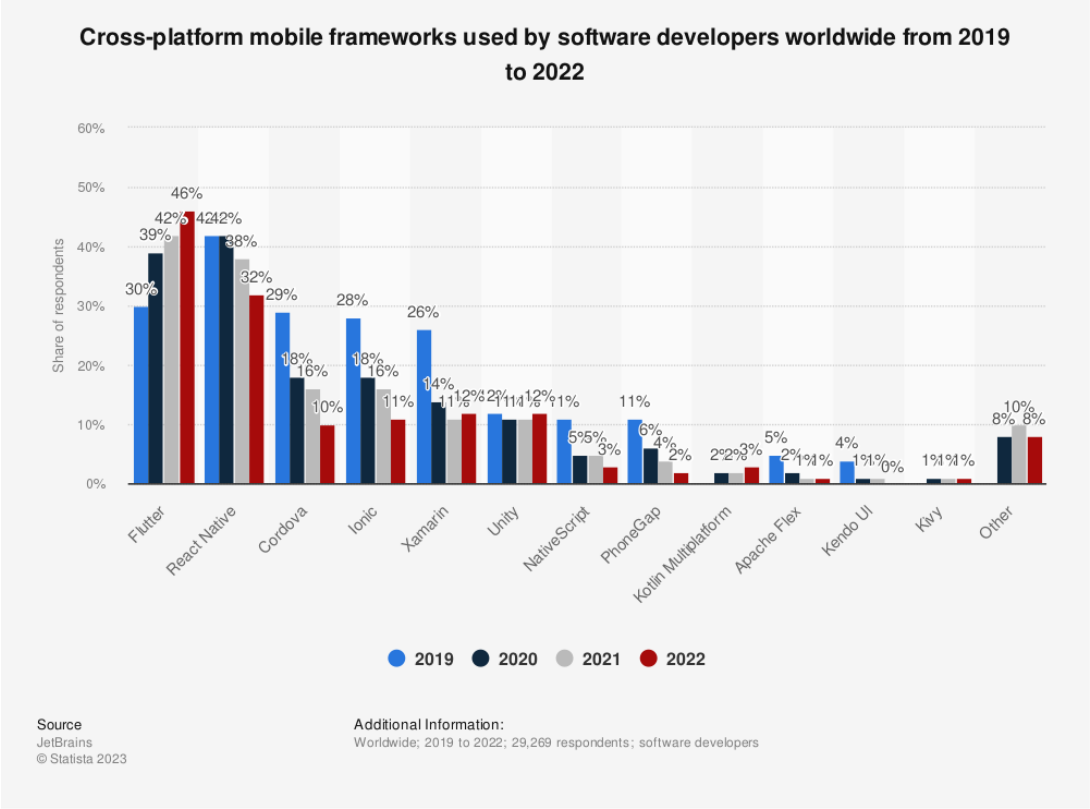
2.2.2 Xamarin

Xamarin je open-source platforma založená na rozhraní .NET, která umožňuje vývojářům vytvářet multiplatformní aplikace pro iOS, Android a Windows. V rámci Xamarinu je kód psán v programovacích jazycích C# F# a VB.NET. Jednou z výhod této platformy je možnost sdílení až 90 % kódu aplikace mezi různými platformami. To znamená, že vývojáři mohou psát veškerou logiku aplikace v jednom jazyce nebo znovu využít existující kód aplikace. Přitom však zůstává zachována 100% dostupnost nativního API daného operačního systému [15].

2.2.3 Flutter

Flutter je open-source framework vyvinutý společností Google, který umožňuje vytváření výkonných multiplatformních aplikací pro Android, iOS, web a další platformy. Flutter používájako svůj primární programovací jazyk Dart, který je kompilován do nativního kódu pro každou platformu. To znamená, že Flutter aplikace dosahují vysokého výkonu a mají přístup k nativním funkcím a API dané platformy. Programovací jazyk Dart byl vyvinut také společností Google jako součást projektu Flutter [18].

Flutter se vyznačuje svou vlastní grafickou knihovnou, což znamená, že nevyužívá nativních komponent a widgetů poskytovaných operačním systémem jako jiné frameworky. Namísto toho nabízí svou vlastní sadu widgetů a grafických prvků, které jsou vykreslovány pomocí vlastního enginu přímo na obrazovku. Tato vlastní grafická knihovna Flutteru se nazývá „Skia“. Skia je rychlý a výkonný 2D grafický engine, který zajišťuje rychlé vykreslování a animace. Flutter využívá Skia pro renderování svých widgetů. Za kvalitu framewoku mluví to, že je nejpoužívanejší ve své kategorii v letech 2021 – 2022, jak je vidět na obrázku 2. Na uvedeném snímku je také statistika používání dalších frameworků.



Obrázek 2: Mobilní frameworky používané vývojáři po celém světě v letech 2019 – 2022 [17].

2.3 Progresivní webové aplikace (PWA)

Progresivní webové aplikace jsou moderním přístupem k vývoji aplikací, který kombinuje sílu webových technologií s funkcemi nativních aplikací. PWA umožňují vývojářům vytvářet aplikace, které mohou běžet na různých zařízeních a platformách, jako jsou desktopové počítače, mobilní telefony a tablety. PWA poskytují pokročilé možnosti, jako je offline přístup k obsahu, notifikace, přístup k hardwarovým funkcím zařízení (například fotoaparátu či geolokaci) a mnoho dalšího. Díky svému progresivnímu charakteru se PWA mohou postupně zlepšovat a přizpůsobovat podle možností a podmínek zařízení, na kterých jsou spouštěny [19].

Důležitým stavebním kamenem této technologie jsou takzvané service workery. Service worker je skript, který běží na pozadí nezávisle na webové stránce a umožňuje funkcionalitu, která nepotřebuje webovou stránku nebo interakci uživatele. Service workery se zabývají procesy na pozadí, a také ukládají kód stažený z webového serveru do aplikační cache, čímž umožňují spuštění aplikace i bez přístupu k internetu. Tímto způsobem je možné poskytovat offline funkce, jako je ukládání dat, zpracování pozadavků na pozadí a další úkony bez přímého připojení k internetu.

Druhým důležitým prvkem jsou webové aplikační manifesty. Tyto manifesty

jsou soubory ve formátu JSON, které určují, jak se webová aplikace bude chovat a vypadat po „instalaci“ na zařízení. Manifesty poskytují informace o aplikaci, jako je název, ikona, výchozí URL adresa, nastavení orientace obrazovky, preferovaný jazyk a další metadatové informace. Tímto způsobem manifesty umožňují webovým aplikacím se prezentovat a chovat se podobně jako nativní aplikace po „instalaci“ na zařízení. Manifesty přispívají ke zlepšení uživatelského zážitku tím, že umožňují ikonu aplikace na domovské obrazovce, offline přístup a další funkcionality, které jsou charakteristické pro nativní aplikace [20].

2.4 Nevidomý a práce s mobilní aplikací

Přibývá uživatelů chytrých zařízení se zrakovou vadou. Zpřístupnit aplikace i pro tyto uživatele je stále více upřednostňováno. Nevidomí lidé využívají chytré telefony a speciální funkce, které jsou navrženy tak, aby jim poskytovaly přístupnost a usnadnily používání telefonu.

Jak na Androidu, tak na iOS platformně jsou zabudované takzvané čtečky již v systému. Na iOS mají nevidomí uživatelé k dispozici odečítač obrazovky VoiceOver, na Androidu pak odečítače TalkBack.

Mobilní zařízení se poslepu ovládají pomocí gest, která jsou jiná než standardní gesta. Případně se může k zařízení připojit externí zařízení, které lze použít jako vstupně-výstupní zařízení [21].

Nevidomí bez externího zařízení jsou odkázáni na dvě zpětné vazby:

1. Hlasová výstupní technologie: Hlasový výstup je klíčovou technologií pro nevidomé uživatele. Aplikace by měly poskytovat možnost přečíst textový obsah na obrazovce nahlas, a to včetně zpráv, navigačního textu, položek menu atd.

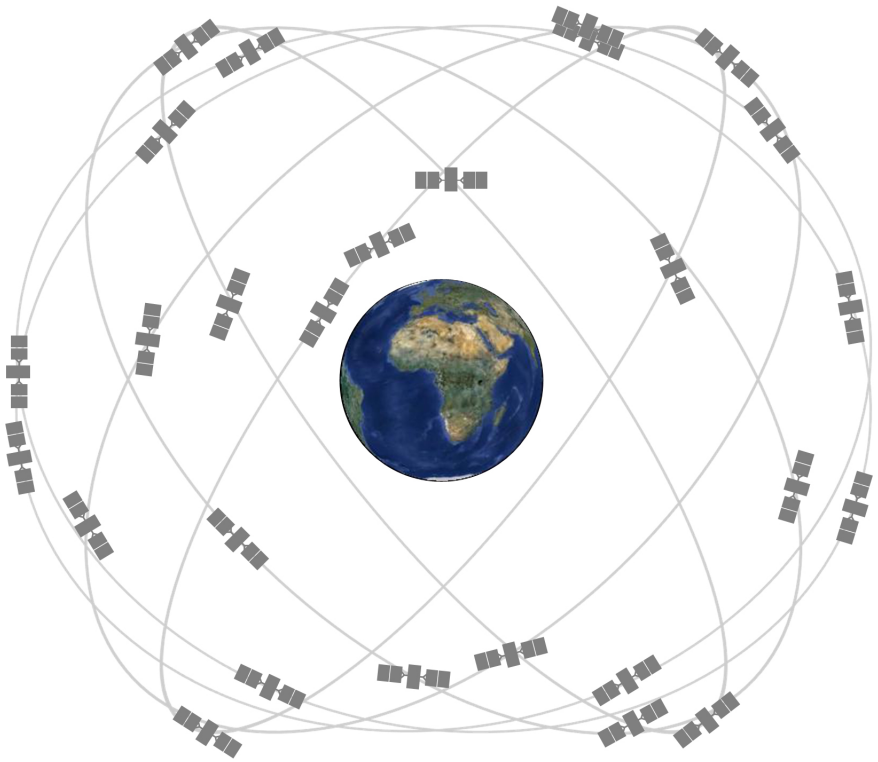
2. Hmatová zpětná vazba: Pro nevidomé uživatele je důležité poskytnout hmatovou zpětnou vazbu, která jim pomůže orientovat se po obrazovce a správně vybrat jednotlivé prvky.

3 Technologie pro lokalizační systémy

V této kapitole budou popsány technologie pro lokalizační systémy, ze kterých se vycházelo při návrhu řešení této diplomové práce.

3.1 GPS

V této podkapitole bude popsána technologie GPS (Global Positioning System), což je satelitní navigační systém používaný k určení pozemní polohy objektu. Systém GPS zahrnuje 31 satelitů rozmístěných ve vesmíru ve výšce 19 300 km nad mořem, ze kterých 24 jsou aktivní. Satelity jsou rovnoměrně rozmístěny a 4 z nich jsou viditelné z jakéhokoliv bodu zemského povrchu Země. Každý satelit má v sobě atomické hodiny pro určování přesného aktuálního času. Každý satelit GPS vysílá zprávu, která obsahuje aktuální polohu satelitu, oběžnou dráhu a přesný čas. Přijímač GPS kombinuje vysílaní ze 3 – 4 satelitů a vypočítává polohu pomoci metody triangulace. Aby zařízení GPS fungovalo správně, tak musí nejprve navázat spojení s požadovaným počtem satelitů, což může trvat od několika sekund do několika minut v závislosti na síle přijímače. Většina zařízení GPS také používá určitý typ ukládání polohy do paměti pro urychlení zjišťování GPS, protože díky zapamatování své předchozí polohy může zařízení rychleji určit, které satelity budou dostupné při příštím vyhledávání signálu GPS [2].

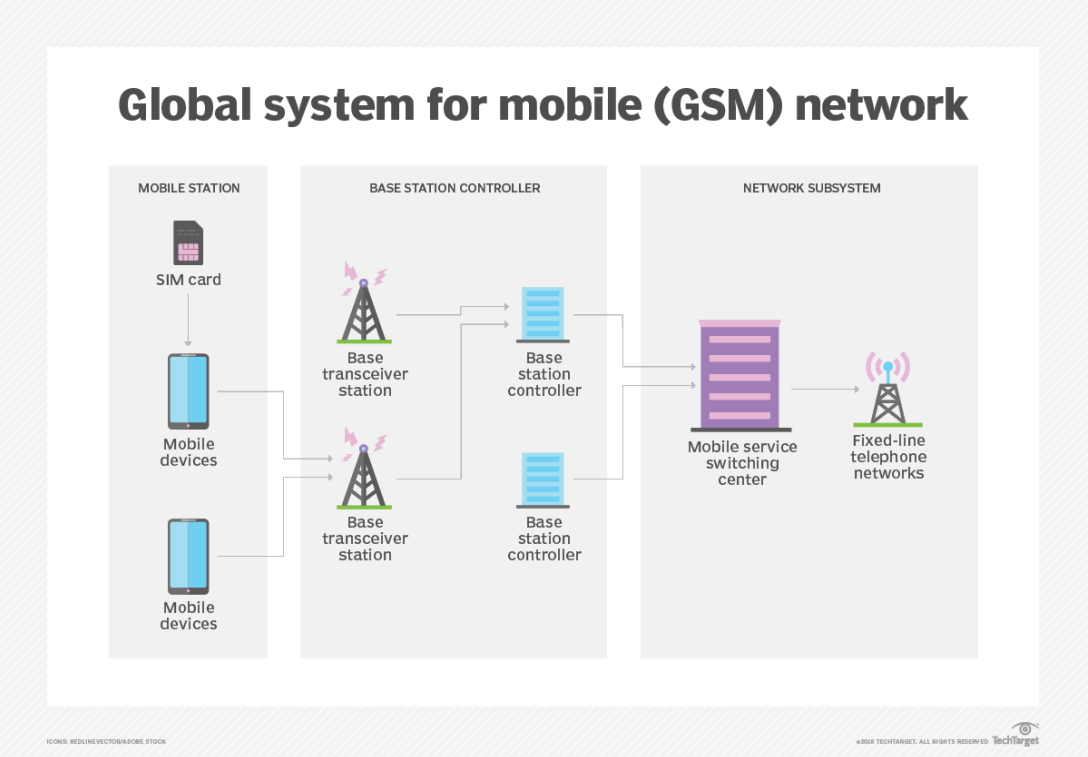


Obrázek 3: Rozšiřitelná 24-slotová satelitní konstelace podle GPS performance standardu [3].

Satelity GPS vysílají své signály s určitou přesností, avšak přesnost při přijímaní signálu závisí na dalších faktorech, jako jsou geometrie satelitů, blokování signálu, atmosférických podmínkách a kvalitě přijímače. V dnešní době mají chytré telefony s GPS přesnost s dosahem 5 metrů, tato přesnost se zhoršuje v blízkosti budov, mostů a stromů [4].

3.2 GSM

GSM (Global Systém for Mobile communication) je digitální mobilní síť, která používá variaci vícenásobného přístupu s časovým dělením. GSM pracuje ve frekvenčním pásmu 900 MHz nebo 1800 MHz a skládá se ze čtyř samostatných částí, které mezi sebou spolupracují: samotné mobilní zařízení, subsystém základové stanice, subsystém přepínání sítí a subsystém provozu a podpory. Mobilní zařízení se připojuje k síti prostřednictvím hardwaru a sim-karta poskytuje síti identifikační informace o uživateli. Subsystém základové stanice řídí provoz mezi mobilním telefonem a subsystémem přepínání sítí. Ten se skládá ze dvou hlavních částí, vysílací a přijímací stanice a řadiče stanice.



Obrázek 4: Schéma organizace sítě GSM [5].

Přestože byl GSM navržen jako bezpečný bezdrátový systém, stále může dojít k útokům. GSM používá autentizační opatření, které vyzve uživatele k poskytnutí platné odpovědí na otázku a předem sdílený klíč ve formě hesla. Mezi hlavní omezení GSM patří elektronické rušení, zpoždění šířky pásma, omezená rychlost přenosu dat a také opakovače pro zvýšení pokrytí [5].

Existuje několik metod lokalizace v mobilních sítích GSM, zejména pomocí ID buňky, předstihu načasování, vylepšenému pozorovanému časovému rozdílu, úhlu příjezdu (Angle of Arrival) a vylepšení globální identity buňky. Při použití metody úhlu příjezdu je možné identifikovat polohu s přesnosti kolem 300 metrů. Při použití metody vylepšené globální identity buňky je možné dosáhnout identifikace polohy s přesnosti 250 m pro venkov a 50 m v budově [6].

3.3 Wi-Fi

Wi-Fi je rodina bezdrátových síťových protokolů založených na rodině standardů IEEE 802.11, které umožňuje zařízením připojení k internetu a komunikaci mezi sebou bez potřeby kabelového spojení. Wi-Fi funguje pomocí využití radiových vln pro přenos a příjem dat mezi zařízeními. Využívá rádiové frekvenční spektrum v pásmu 2,4 GHz, 5 GHz a 6 GHz. Wi-Fi sítě se skládají ze dvou částí: bezdrátových přístupových bodů (routerů) a klientských zařízení. Přístupový bod funguje jako centrální uzel, který propojuje více klientů a vytváří síť. Každý přístupový bod má vlastní identifikátor sítě (SSID). Rádiová pásma Wi-Fi mají vysokou absorpci a nejlépe fungují při přímé viditelnosti. Celá řada překážek, zejména stěny, sloupy, stromy, domácí spotřebiče, může výrazně snížit dosah, i když pomáhají minimalizovat rušení mezi různými sítěmi v přeplněných prostředích. Dosah Wi-Fi komunikace za ideálních podmínek může být až 150 metrů s přesností v řádech několika metrů [9].

3.4 Bluetooth

Bluetooth je standard bezdrátové technologie krátkého dosahu, který se používá pro výměnu dat mezi zařízeními na krátké vzdálenosti. Bluetooth je postaven na radiotechnologii s krátkým dosahem, využívá rádiové vlny pro přenos dat v pásmech od 2,402 do 2,48 GHz. Nejčastější verze Bluetooth používaná v současnosti je Bluetooth Low Energy (BLE), která je optimalizována pro nízkou spotřebu energie a je vhodná pro připojení zařízení s omezenou bateriovou kapacitou. Používá se hlavně jako alternativa k drátovému připojení, k výměně souborů mezi zařízeními a mobilními telefony. Bluetooth používá metodu šířkového spektra, což umožňuje více zařízením komunikovat ve stejném časovém intervalu. Dosah Bluetooth se pohybuje obvykle v rozmezí 10 metrů, ale moderní verze Bluetooth mají dosah až 100 metrů. Skutečný dosah dosažený daným spojením bude záviset na kvalitě oboustranných zařízení, vzduchu a překážkách mezi těmito zařízeními. Bluetooth rozděluje přenášená data do paketů a přenáší každý paket na jednom ze 79 určených kanálů. Každý kanál má šířku pásma 1 MHz.

Bluetooth definuje několik vrstev architektury, které usnadňují komunikaci mezi zařízeními. Základními vrstvami jsou fyzická vrstva (Physical Layer) a vrstva ovládání přístupu ke středisku (Link Layer). Tyto vrstvy se starají o přenos dat a řízení komunikace mezi zařízeními. Nad nimi se nachází vrstvy, jako je L2CAP (Logical Link Control and Adaptation Protocol), která poskytuje protokoly pro

řízení přenosu dat, a nejvyšší je aplikační vrstva [8].

3.5 ZigBee

ZigBee je bezdrátová technologie, která byla založená na IEEE 802.15.4 a vyvinuta pro levné bezdrátové sítě M2M (Machine-to-Machine) a internet věcí (IoT) s nízkou spotřebou energie. Technologie definovaná specifikací ZigBee má být jednodušší a levnější než jiné bezdrátové osobní sítě (WPANs), jako například Bluetooth nebo Wi-Fi. Aplikace zahrnují bezdrátové spínače světel, monitoring energie a další spotřebitelská, lékařská a průmyslová zařízení, která vyžadují bezdrátový přenos dat s nízkou rychlostí na krátké vzdálenosti. Nízká spotřeba ZigBee omezuje přenosové vzdálenosti na 10-100 metrů přímé viditelnosti v závislosti na výkonu a charakteristikách prostředí.

Norma IEEE 802.15.4 specifikuje provoz v pásmech 2,4 až 2,4835 GHz (celosvětově), 902 až 928 MHz (Amerika a Austrálie) a 868 až 868,6 MHz (Evropa). Přenosová rychlost dat je 250 kbit/s v pásmu 2,4 GHz, 40 kbit/s v pásmu 915 MHz a 20 kbit/s v pásmu 868 MHz. Skutečná přenosová rychlost bude ale nižší než maximální specifikovaná přenosová rychlost kvůli režii paketů a zpoždění při zpracovaní dat [7].

3.6 RFID

RFID (Radio Frequency Identification – radiofrekvenční identifikace) je technologie bezdrátové komunikace, která využívá elektromagnetické nebo elektrostatické vazby v radiofrekvenčním spektru k identifikaci objektů nebo osob. Systém RFID se skládá ze tří základních komponent: antény, transceiveru a transpondéru. RFID čtečka je kombinací antény a transceiveru. Transpondér je elektronické zařízení umístěné v RFID štítku, které přijímá rádiové signály a odesílá zpět odpověď. Tento štítek obsahuje také energeticky nezávislou paměť pro ukládání a čtení dat [10].

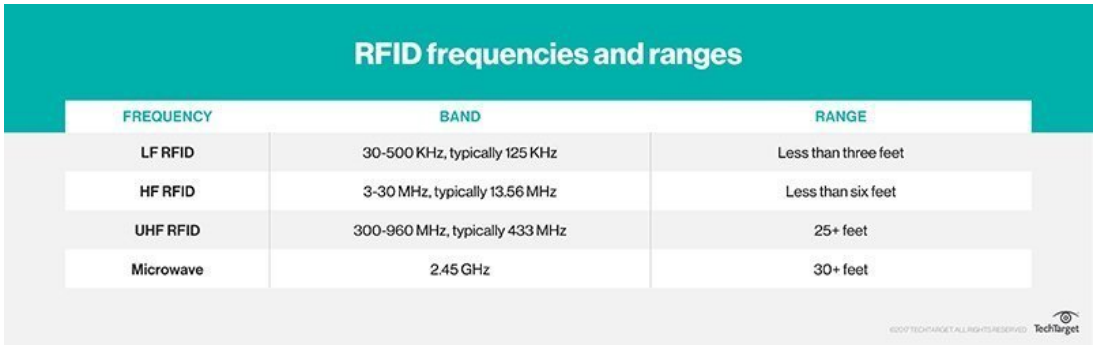
Existují dva hlavní typy transpondérů:

- Aktivní transpondéry: Tyto transpondéry mají vlastní zdroj energie a mohou přenášet data na větší vzdálenost.
- Pasivní transpondéry: Tyto transpondéry získávají energii pro přenos dat z elektromagnetického pole čtečky.

RFID systémy se dále dělí podle pracovní frekvence, která ovlivňuje dosah komunikace mezi čtečkou a štítkem:

- Nízkofrekvenční RFID systémy: Pracovní frekvence těchto systémů je obvykle 125 kHz, ale frekvenční pásmo se pohybuje v rozmezí 30 – 500 kHz. Pracovní vzdálenost je menší než 1 metr.

- Vysokofrekvenční RFID systémy: Pracovní frekvence těchto systémů je obvykle 13,56 MHz s frekvenčním pásmem 3 – 30 MHz. Pracovní vzdálenost je kolem 2 metrů.
- Ultra vysokofrekvenční RFID systémy: Pracovní frekvence těchto systémů je obvykle 433 MHz s frekvenčním pásmem 300 – 960 MHz. Pracovní vzdálenost je kolem 7 metrů.
- Mikrovlnné RFID systémy: Pracovní frekvence těchto systémů je 2,45 GHz. Pracovní vzdálenost je kolem 9 metrů.



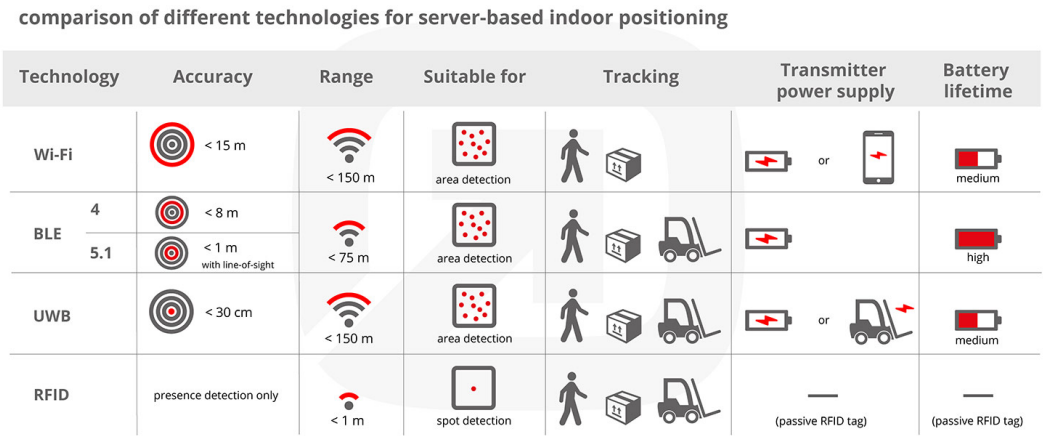
Obrázek 5: Porovnání rozsahu RFID s různými pracovními frekvencí [10].

Pro komunikaci mezi čtečkou a štítkem využívají RFID systémy rádiové vlny. Čtečka vysílá signál, který aktivuje štítek a způsobí odeslání dat zpět do čtečky. Po přijetí signálu čtečkou se vlna konvertuje na užitečná data.

3.7 UWB

Ultra-wideband (UWB) je rádiová technologie pro přenos informací přes širokou šířku pásma ($>$ 500 MHz). Tato technologie se stále více uplatňuje v různých odvětvích, včetně průmyslu, zdravotnictví, bezpečnosti a spotřební elektroniky a používá se ke sběru dat ze sensorů, měření přesné lokalizace a sledování pohybu v reálném čase. Významným rozdílem mezi tradičním radiovým přenosem a UWB radiovým přenosem je, že tradiční způsob přenosu informace využívá úrovňovou, frekvenční a/nebo fázovou modulaci sinusové nosné vlny. UWB využívá krátké a rychlé pulsy rádiových vln, což přináší výhodu nízké spotřeby energie. To je zvláště důležité pro zařízení s bateriemi, jako jsou chytré telefony, hodinky nebo senzory IoT. Jelikož UWB technologie využívá širokého frekvenčního pásma a krátkých impulzů, tak jí to umožňuje dobrou prostupnost skrze různé překážky, jako jsou stěny, budovy nebo stromy. Frekvenční rozsah UWB je 3,1 – 10,6 GHz s maximální datovou rychlosti 480 Mbit/s. Výhodou UWB technologie je také nízká náchylnost k rušení. Přesnost lokalizace UWB technologie je 0,1 – 0,5 m s dosahem 10 – 200 metrů v závislostí na typu aplikace [11].

Na obrázku 6 lze vidět porovnání rozsahu a přesnosti některých technologií, jako jsou Wi-Fi, Bluetooth, RFID a UWB.



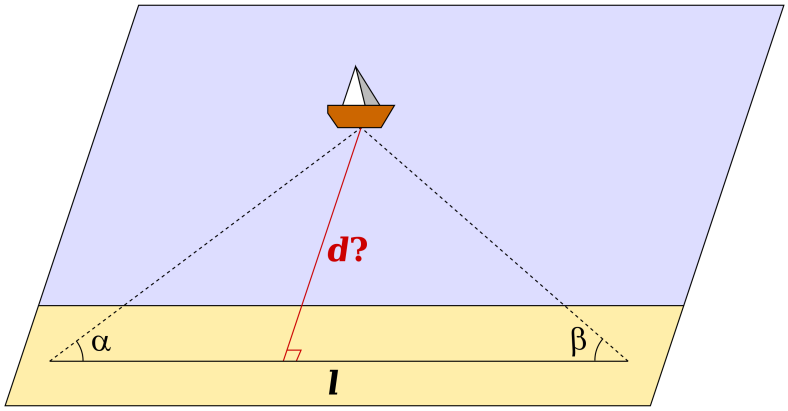
Obrázek 6: Porovnání rozsahu a přesnosti Wi-Fi, Bluetooth, RFID a UWB [12].

4 Metody určení směru

V této kapitole budou popsané metody určení směru, o kterých se uvažovalo nebo byli použitý v rámci realizace diplomové práce, zejména metoda triangulace, trilaterace, úhel příchodu signálu a síla přijatého signálu.

4.1 Triangulace

Triangulace je způsob zjišťování souřadnic a vzdáleností pomocí trigonometrického výpočtu. Sestrojí se pomyslný trojúhelník, jehož jedna strana je strana již známého jiného trojúhelníku s dvěma koncovými referenčními body, třetím bodem je místo, jehož souřadnice se zjišťují. Triangulace se nejčastěji užívá pro účely geodezie, metrologie, navigace a astronomie [39].



Obrázek 7: Příklad výpočtu vzdálenosti d na základě triangulace [39].

Na obrázku 8 lze vidět příklad triangulace, která se používá pro výpočet vzdálenosti a polohy lodi od pobřeží. Pokud jsou známe úhly α, β a vzdálenost l nebo souřadnice bodů A a B, tak je možné na začátku vypočítat třetí neznámý úhel θ, následně na základě sinové věty je možné vypočítat zbývající 2 délky stran trojúhelníku podle vzorce.

$$\frac{sin(\alpha)}{A} = \frac{sin(\beta)}{B} = \frac{sin(\theta)}{l}$$

Nakonec je možné vypočítat souřadnice lodi v bodě C a vzdálenost d podle vzorce.

$$sin(\alpha) = \frac{d}{B}$$

nebo

$$sin(\beta) = \frac{d}{A}$$

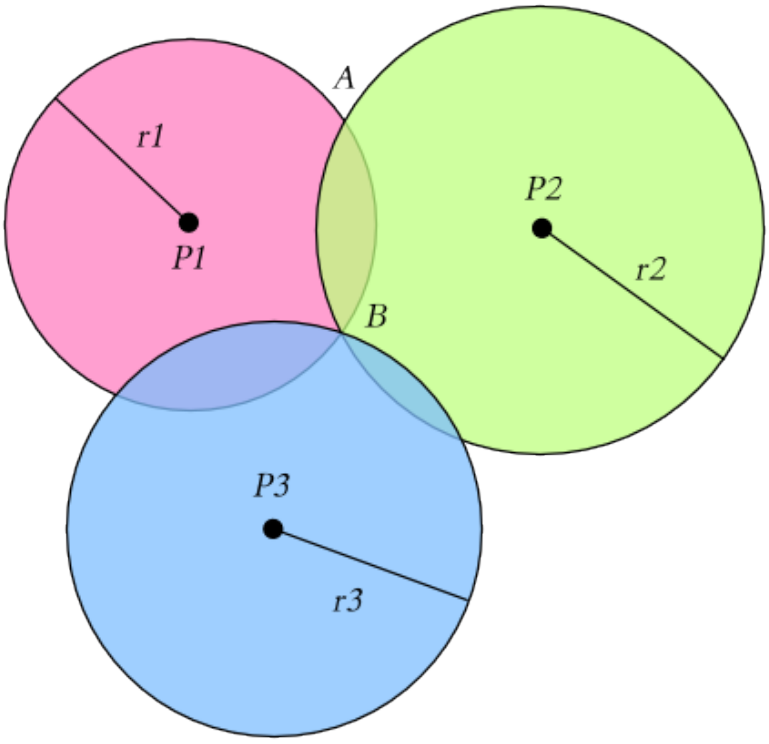
4.2 Trilatence

Trilaterace je metoda lokalizace, která na rozdíl od triangulace nepracuje s úhly, ale s délkami. Ty slouží k určení polohy objektu nebo zařízení v prostoru na základě znalosti vzdálenosti od tří nebo více referenčních bodů. Tato technika má široké uplatnění v různých oborech jako je navigace, geodezie, robotika, mobilní komunikace, bezdrátové senzorové sítě a dalších.

Trilaterace je založená na znalosti vzdálenosti od alespoň tří referenčních bodů. Pokud známe polohy tří bodů (A, B, C) v prostoru, potřebujeme znát vzdálenosti bodu D od těchto tří známých bodů, abychom určili polohu bodu D. Tyto vzdálenosti slouží jako poloměry kružnic se středy v bodech A, B a C. Tyto kružnice se protínají v jediném bodě D. Na základě tohoto principu lze určit polohu objektu ve dvourozměrném nebo trojrozměrném prostoru, ale většinou se pracuje s dvourozměrným prostorem [40].

V praxi k měření vzdálenosti se využívají signály, například rádiové, ultrazvukové nebo optické, které se šíří od referenčních bodů k objektu. Měření vzdálenosti od referenčních bodů není vždy přesné tak, aby se kružnice protínaly v jediném bodě. Místo toho se využívají kruhy a poloha bodu D se určí jako průsečík kruhů.

Princip spočívá ve vytvoření kružnic kolem známých bodů s poloměry o velikostech vzdáleností k hledanému bodu. Pokud jsou známé pouze dva body, dochází k průniku kružnic ve dvou místech. Z toho vyplývá, že k přesnému určení polohy hledaného bodu je za potřebí minimálně tří známých bodů.



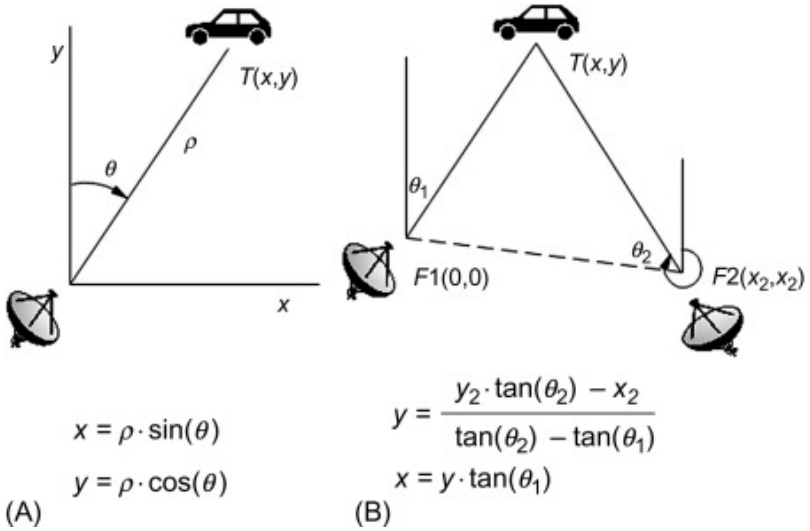
Obrázek 8: Princip metody trilaterace [40].

Mějme tři body $P1$, $P2$ a $P3$ na souřadnicích $(x1, x2)$, $(x2, y2)$ a $(x3, y3)$. Každý bod má kruh s poloměry $r1$, $r2$, $r3$. Poloměr je dán naměřenou nebo vypočítanou vzdáleností od hledaného bodu. Hledaný bod má souřadnice (x, y). Pro každou kružnici je určena rovnice a řešením soustavy tří kvadratických rovnic o dvou neznámých se hledá průsečík těchto kružnic.

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

4.3 Angle of Arrival (AoA)

Úhel příchodu signálu nebo Angel of Arrival (AoA) je směr, ze kterého je signál přijímán. Měření úhlu příchodu signálu lze provést tím, že se určí směr šíření rádiového signálu dopadajícího na anténní pole nebo se stanoví z maximální síly signálu během otáčení antény. Úhel příchodu signálu lze vypočítat měřením časového rozdílu příchodu (TDOA) mezi jednotlivými prvky pole. Toto měření TDOA se provádí tím, že se měří rozdíl přijaté fáze na každém prvku v anténním poli. To lze přirovnat k opačnému směrování paprsků. Při směrování paprsků je signál z každého prvku vážen tak, aby „nasměroval“ zisk anténního pole. U AoA je zpoždění příchodu na každém prvku měřeno přímo a převedeno na měření AoA. Pokud anténní pole má dva prvky vzdálené o polovinu vlnové délky přicházejícího rádiového signálu, a pokud je signál dopadající na pole přesně ve směru osy antény, tak signál dorazí na obě antény současně. To způsobí rozdíl fáze 0° mezi oběma prvky antény, což odpovídá AoA 0°. Pokud signál přijde na pole kolmo, pak bude změřen rozdíl fáze 180° mezi prvky, což odpovídá AoA 90°. Omezení přesnosti odhadu úhlu příchodu signálu v digitálních anténních polích souvisí s chybami ADC a DAC převodníků [41].



Obrázek 9: Princip metody úhlu příchodu signálu [42].

4.4 Síla přijatého signálu

Mimo dobu přenosu jsou další vlastnosti signálu, které lze použít pro určení vzdálenosti mezi objekty. Jednou z těchto vlastností je indikátor síly přijatého signálu (RSSI – Received Signal Strength Indication), což je hodnota, která udává množství elektrické energie, kterou zařízení přijímá ze vzdáleného zdroje signálu. Tato hodnota je udávána v dBm a může být vypočítaná následovně.

$$P_d = P_0 - 10 \cdot n \cdot \lg\left(\frac{d}{d_0}\right)$$

- $d[m]$ – vzdálenost od zařízení k vysílači.
- $d_0[m]$ – vzdálenost od zařízení k bodu, kde byla měřena síla signálu $P0$ zařízení, obvykle 1 m.
- $P_0[dBm]$ – síla signálu zařízení, která je měřená na jednotkovou vzdálenost d_0 od zařízení
- $n[-]$ – koeficient ztráty výkonu signálu při šíření v médiu(pro vzduch n=2)
- $P_d[dBm]$ – RSSI.

Nebo také podle vzorce.

$$RSSI[dBm] = RSCP[dBm] - \frac{Ec}{Io[dB]}$$

kde RSCP (Received Signal Code Power – výkon užitečného přijatého signálu) je souhrn radiofrekvenčních energie po korekcích, jako například odfiltrování šumu, a udává se v dBm. Ec/Io je poměr přijaté energie a energie přijatého šumu, udává se v dB. Tato metoda se používá především v telekomunikačních technologiích, jako jsou GSM, 5G, Wi-Fi a jiné. Síla přijatého signálu může být ovlivněna několika faktory, jako jsou vzdálenost mezi vysílačem a přijímačem, překážky v prostředí, například stěny budov, atmosférické podmínky, rušení od jiných zařízení atd [43].

5 Existující řešení

V této kapitole se zaměříme na porovnání existujících řešení pro lokaci předmětů.

5.1 AirTag

AirTag je lokátor od firmy Apple. Lze jej vidět na obrázku 10. AirTag je kompatibilní s výrobky firmy Apple, takže s další nejpoužívanější platformou Android nelze použít. Hlavní myšlenkou tohoto produktu je přidělat ho k předmětu, který se často hledá nebo ztrácí, například klíče.

AirTag používá pro nalezení tři způsoby [26]:

- Bluetooth pro vyhledání v blízkosti telefonu. Pokud je telefon připojený přes Bluetooth, můžeme si přehrát v AirTagu zvuk, pomocí kterého ho najdeme.
- Čip U1, jak jej označuje firma Apple, pracuje na principu UWB technologie. Pokud telefon má čip U1, tak telefon v blízkosti může ukázat směr a vzdálenost k AirTagu. Ovšem, jak funguje určení směru, není vysvětleno.
- Okolní zařízení. AirTag vysílá Bluetooth signál, který mohou detekovat ostatní zařízení a toto zařízení odešle polohu AirTagu. V podstatě zde Apple využívá síť používaných zařízení.

Obrázek 10: Apple AirTag [22].

5.2 Galaxy SmartTag

Galaxy SmartTag je lokátor od firmy Samsung. Vyráběné varianty jsou vidět na obrázku 11. Galaxy SmartTag je kompatibilní s více výrobci telefonů se systémem Android, ale mohou být nedostupné některé funkce. Některé pokročilé funkce jsou dostupné pouze na zařízeních Samsung. Galaxy SmartTag není podporován

pro iOS. Hlavní myšlenkou tohoto produktu je přidělat ho k předmětu, který se často hledá nebo ztrácí například klíče, peněženka nebo zavazadlo.

Galaxy SmartTag používá pro nalezení dva způsoby [23]:

- Bluetooth pro vyhledání v blízkosti telefonu. Pokud je telefon připojený přes Bluetooth můžeme si přehrát Galaxy SmartTag zvuk, pomocí kterého ho najdeme.
- Okolní zařízení Galaxy SmartTag vysílá Bluetooth signál, který mohou detekovat ostatní zařízení. Tato funkce využívá anonymního sdílení polohy mezi uživateli, což umožňuje lokalizaci ztraceného předmětu na základě signálů od ostatních zařízení SmartTag v okolí. V podstatě zde Samsung využívá síť svých zařízení.

Obrázek 11: Chytrý přívěsek Galaxy SmartTag ve čtyrech barvách [23].

5.3 Chipolo

Chipolo je firma vyrábějící celou řadu čipů/lokátorů v různých barvách a provedeních, jak je vidět na obrázku 12. Čipy od firmy Chipolo jsou kompatibilní jak s Android, tak i iOS.

Chipolo používá k nalezení dva způsoby [24]:

- Bluetooth pro vyhledání v blízkosti telefonu. Pokud je telefon připojený přes Bluetooth, můžeme si přehrát v čipu zvuk, pomocí kterého ho najdeme. Možné je to i naopak. Pokud máte čip, můžete stisknout tlačítko a mobil začne přehrávat zvuk.
- Okolní zařízení s aplikací Chipolo. Chipolo vysílá Bluetooth signál, který mohou detekovat ostatní zařízení s aplikací Chipolo. Zde si Chipolo vytváří síť uživatelů k lokaci mimo dosah Bluetooth.

Obrázek 12: Barevné provedení produktů od chipolo [24].

5.4 Tile

Tile je firma vyrábějící celou řadu čipů/lokátorů v různých barvách (různé barvy nejsou vidět na obrázku) a provedeních, jak je vidět na obrázku 13. Čipy od firmy Tile jsou kompatibilní jak s Android, tak i iOS.

Tile používá k nalezení dva způsoby [35]:

- Bluetooth pro vyhledání v blízkosti telefonu. Pokud je telefon připojený přes Bluetooth, můžeme si přehrát v čipu zvuk, pomocí kterého ho najdeme. Lze to i naopak. Pokud máte čip, můžete stisknout tlačítko a mobil začne přehrávat zvuk i v tichém režimu. Také je v aplikaci indikátor zobrazující přiblížení.
- Okolní zařízení s aplikací od Tile. Tile vysílá Bluetooth signál, který mohou detekovat ostatní zařízení s aplikací od Tile. Tile zdarma uchovává poslední polohu 24 hodin, pro třicetidenní historii je potřeba zaplatit premium členství. Zde si Tile vytváří síť uživatelů k lokaci mimo dosah Bluetooth.

Obrázek 13: Čtyři produkty od tile [25].

5.5 Motivace k návrhu

Existující řešení jsou nevyhovující primárně z hlediska dosahu a zvukové signalizace. Zvuková signalizace je tichá a je nemožné najít stan. Řešením by byla hlasitější signalizace, to je ovšem neohleduplné k ostatním například kempovací tábor, kde jiní lidé chtějí klid či spát. Takto hlasitá signalizace by je rušila.

Všechna existující řešení využívají spíše své vybudované sítě. Přes tyto vybudované sítě dosahují lokace na větší vzdálenosti, ale to není zaručeno bez okolních zařízení v sítí. Jenom AirTag s určitým typem telefonu dokáže určit vzdálenost a směr zařízení, ale tato vzdálenost je jen pár metrů. Navíc žádná z uvedených řešení nemají uzpůsobenou aplikaci pro nevidomé. Proto jsem chtěl navrhnout řešení, které by bylo bez zvukové signalizace kvůli ohleduplnosti. S aplikací přizpůsobenou pro nevidomé uživatele. Pokud je uživatel v dosahu, lze určit směr i vzdálenost.

6 Popis použitých HW prostředků

V této kapitole budou popsané HW prostředky, které byly využity pro realizaci této diplomové práce, zejména deska ESP32 UWB Pro, kterou lze vidět na obrázku 16, její součástí je řídicí jednotka ESP32-WROVER-B a UWB čip DWM1000, které komunikují mezi sebou prostřednictvím SPI sběrnice. Součástí HW řešení jsou také magnetometr QMC5883L a OLED display SSH1106.

V rámci této práce byl použít OLED display SSH1106 s úhlopříčkou 1,3“ a rozlišením 128 × 64 px. Displej se napájí 3,3 – 5V a komunikace s řídicí jednotkou probíhá pomocí I2C rozhraní [27].

6.1 Procesorové sériové sběrnice

V této podkapitole budou popsány procesorové sériové sběrnice SPI a I2C, které byly použity pro realizace této práce.

6.1.1 Sběrnice SPI

SPI (Seriál Peripheral Interface) je sériové komunikační rozhraní, které se používá pro komunikaci na krátkou vzdálenost, především ve vestavěných systémech. Prostřednictvím této sběrnice komunikují zařízení mezi sebou v plně duplexním režimu pomocí architektury master-slave, což znamená, že data mohou být přijímána a odesílána současně. Hlavní zařízení vytváří rámec pro čtení a zápis dat. Master zařízení může komunikovat s několika slave zařízeními, pomocí SS (Slave Select) vodiče, který slouží k označení konkrétního slave zařízení, které má být aktivováno pro komunikaci [33].

Sběrnice SPI specifikuje čtyři logické signály:

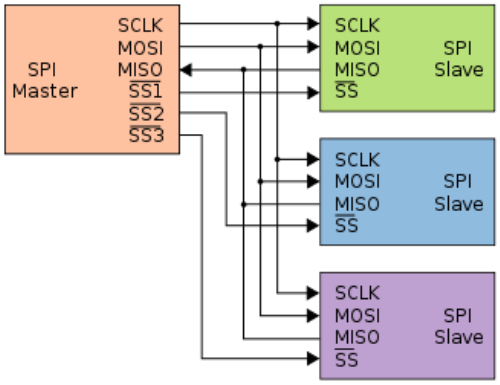
1. SCLK (Seriál Clock): Slouží jako synchronizační hodiny, které jsou generovány Master zařízením. Data se přenášejí s každým pulzem těchto hodin.

2. MOSI (Master Out Slave In): Master zařízení používá tento vodič k přenosu dat do Slave zařízení.

3. MISO (Master In Slave Out): Slave zařízení používá tento vodič k přenosu dat do Master zařízení.

4. SS (Slave Select): Slouží k označení konkrétního Slave zařízení, pro komunikaci.

Pro komunikaci mezi zařízeními nastaví Master na začátku logickou 0 na SS zařízení, se kterým chce komunikovat. Následně Master začne generovat hodinový signál SCLK a v té chvíli vyšlou obě zařízení svoje data. Délka vyslaných dat je buď 8 bitů, nebo 16 bitů.



Obrázek 14: SPI komunikace Master-Slave [33].

Vztah mezi hodinovým signálem a daty u sběrnice SPI se určuje dvěma konfiguračními bity, které se označuji jako CPOL a CPHA.

- CPOL = 0: Klidová úroveň hodinového signálu je logická 0.
- CPOL = 1: Klidová úroveň hodinového signálu je logická 1.
- CPHA = 0: Data jsou čtena při přechodu hodin z klidové úrovně do aktivní úrovně.
- CPHA = 1: Data jsou čtena při přechodu hodin z aktivní úrovně do klidové úrovně.

Tabulka módů definuje, jakou kombinace CPOL a CPHA sběrnice SPI má. Tyto módy často využívají některé logické analyzátory, datasheety součástek nebo programovací knihovny.

Tabulka 1: Tabulka módů sběrnice SPI pro „Microchip PIC“ / „ARM-based“ [33].

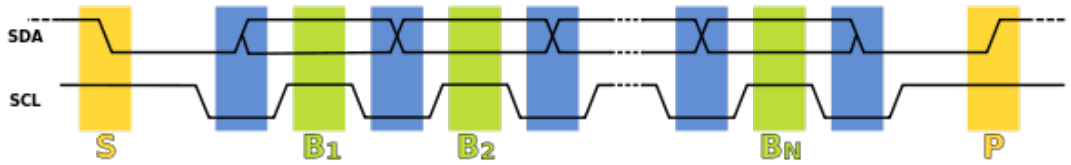
SPI mode	Clock polarity (CPHA)	Clock phase (CPOL/CKP)	Clock edge (CKE/NCPHA)
0	0	0	1
1	0	1	0
2	1	0	1
3	1	1	0

Tabulka 2: Tabulka módů sběrnice SPI pro ostatní MCU [33].

Mode	CPOL	CPHA
0	0	0
1	0	0
2	1	0
3	1	1

6.1.2 Sběrnice I2C

I2C (Inter-Integrated Circuit) je synchronní, multi-master sériová sběrnice, která se používá pro komunikaci mezi vestavěnými systémy na krátkou vzdálenost. Sběrnice I2C využívá architekturu Master-Slave, kde Master zařízení zahajuje a ukončuje komunikaci, generuje hodinový signál SCL. Slave zařízení reagují na příkazy Master zařízení. Na sběrnici je možné zapojit až 128 různých zařízení se dvěma vodiči. Jeden tvoří datový kanál SDA (Synchronous Data) pro přenos dat mezi zařízeními a druhý hodinový signál SCL (Synchronous Clock) pro synchronizaci datových přenosů. Maximální délka vodičů je dána jejich nejvyšší přípustnou kapacitou 400 pF. Rychlost komunikace na I2C sběrnici může mít několik módů, zejména: standardní mód s frekvenci 100 kHz, rychlý mód s frekvenci 400 kHz, nebo High-Speed mód s frekvenci 3,4 MHz [34].



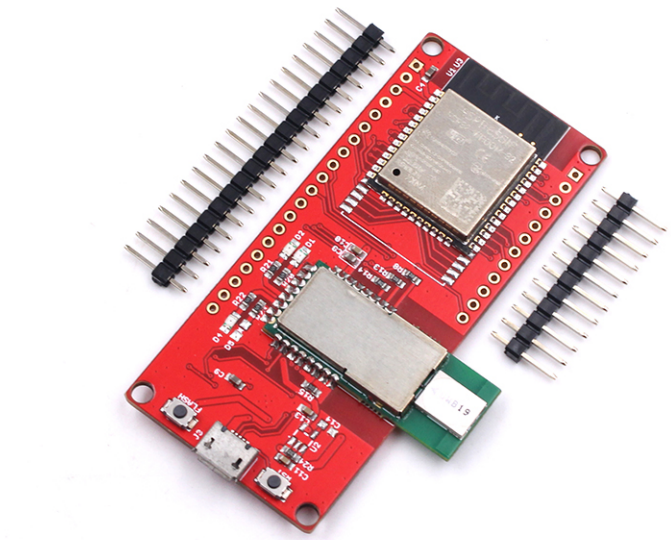
Obrázek 15: Časový diagram komunikace u sběrnice I2C [34].

Na obrázku 15 lze vidět časový diagram komunikace pomocí sběrnice I2C. Přenos dat se zahajuje pomocí START bitu (S), který generuje Master zařízení. Po dokončení přenosu je odeslán STOP bit (P), který uvolňuje datovou linku. STOP bit je zaslán tím, že se signál SDA změní na vysokou úroveň, zatímco signál SCL zůstává trvale vysoký. Každé Slave zařízení na sběrnici má unikátní 7-bitovou adresu, která mu umožnuje komunikaci s Master zařízením. Po zachycení podmínky START, porovnají všechny Slave zařízení svou adresu s adresou, kterou vyšle Master. Zjistí-li Slave zařízení shodu, tak musí přijetí adresy potvrdit bitem ACK, následně přijímá nebo vysílá data. Přenos dat se provádí po bytech, kde každý byte je potvrzován Slave zařízením pomocí bitu ACK.

6.2 Řídicí jednotka ESP32-WROVER-B

ESP32-WROVER-B je výkonný MCU od společnosti Espressif Systems, který má v sobě Wi-Fi, Bluetooth a BLE. MCU podporuje Bluetooth 4.2 až 5.0 a podporuje

jak klasické Bluetooth (BR/EDR), tak i Bluetooth Low Energy (BLE). Frekvence procesoru je nastavitelná od 80 MHz do 240 MHz. ESP32-WROVER má 4 MB integrované Flash paměti a 8 MB PSRAM. MCU umožňuje používání sériových komunikačních rozhraní, jmenovitě UART, SPI, I2C, I2S a také PWM a ADC, DAC, což umožňuje připojení senzorů, displejů, motorů a dalších zařízení. Pro napájení celé desky ESP32 UWB Pro je potřeba napětí 5 V, deska se napájí přes USB-A konektor. Pro napájení čipu ESP32-WROVER je potřeba napětí 3,0 – 3,6 V, které zajišťuje DC/DC převodník napětí na desce [28].



Obrázek 16: ESP32 UWB Pro (ESP32-WROVER a DWM1000) [29].

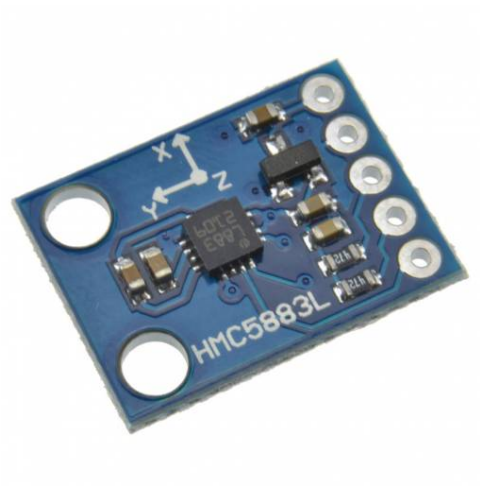
6.3 UWB čip DWM1000

DWM1000 je Ultra Wideband (UWB) čip od společnosti DecaWave. Tento čip integruje RF obvody, řízení napájení a obvody hodin v jednom modulu. Používá se především pro určování vzdálenosti nebo v lokalizačních systémech pomocí UWB technologie. DWM1000 pracuje ve frekvenčním rozsahu 3,5 – 6,5 GHz a podporuje přenos datových paketů s rychlosti 6,8 Mbps. Pro napájení tohoto čipu je potřeba napětí v rozmezí 2,8 – 3,6 V. DWM1000 má několik režimů spotřeby energie, včetně spánkového režimu pro sníženi spotřeby v nečinném stavu. Pro komunikaci s řídicí jednotkou se používá SPI rozhraní. Pomocí tohoto čipu lze dosáhnout za ideálních podmínek přesnosti lokalizace 10 cm s dosahem 200 metrů [30].

6.4 Magnetometr QMC5883L

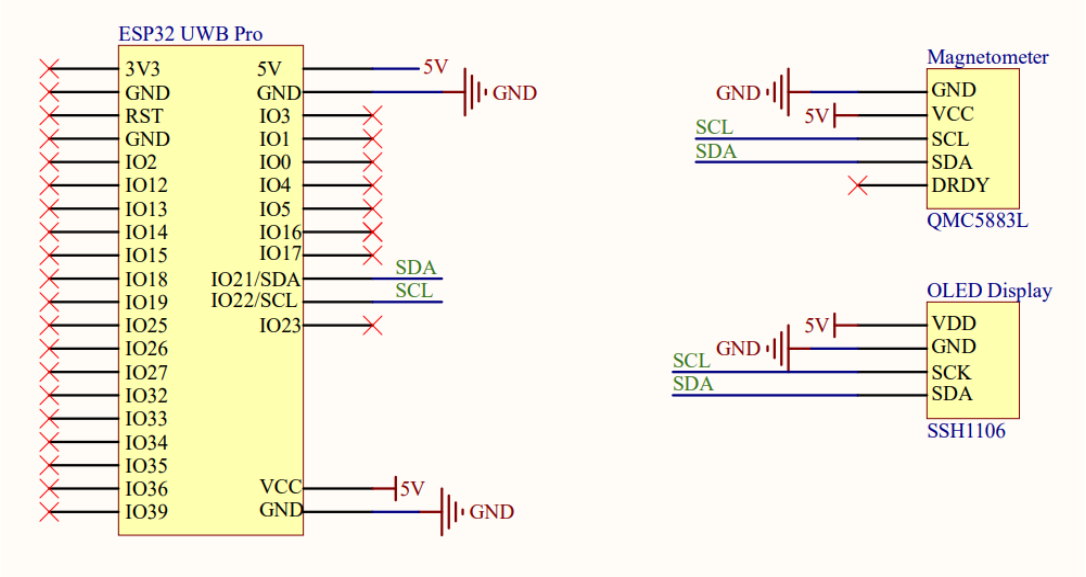
Pro splnění požadavků diplomové práce jsem se rozhodl použít magnetometr k měření magnetického pole, zejména magnetometr QMC5883L, který lze vidět na obrázku 17.

QMC5883L je snímač pro měření magnetického pole a je založen na principu Hallova jevu. Používá se k navigaci, ke sledování polohy a orientace, v robotech, mobilních a osobních zařízeních. Rozsah napájení snímače je 2,16 – 3,6 V. Magnetometr má nízkou spotřebu a má široký rozsah měření +- 8 Gaussů a poskytuje 16-bitová data pro každou osu (X, Y, Z), což vede k vyšší přesnosti měření. Maximální rychlost datového výstupu ze snímače je 200 Hz. Pro práci se snímačem je potřeba na začátku provést jeho kalibraci, aby se minimalizovaly chyby, které mohou vzniknout rušivými magnetickými poli. Komunikace s řídicí jednotkou je možná prostřednictvím I2C rozhraní [31].



Obrázek 17: Magnetometr QMC5883L [32].

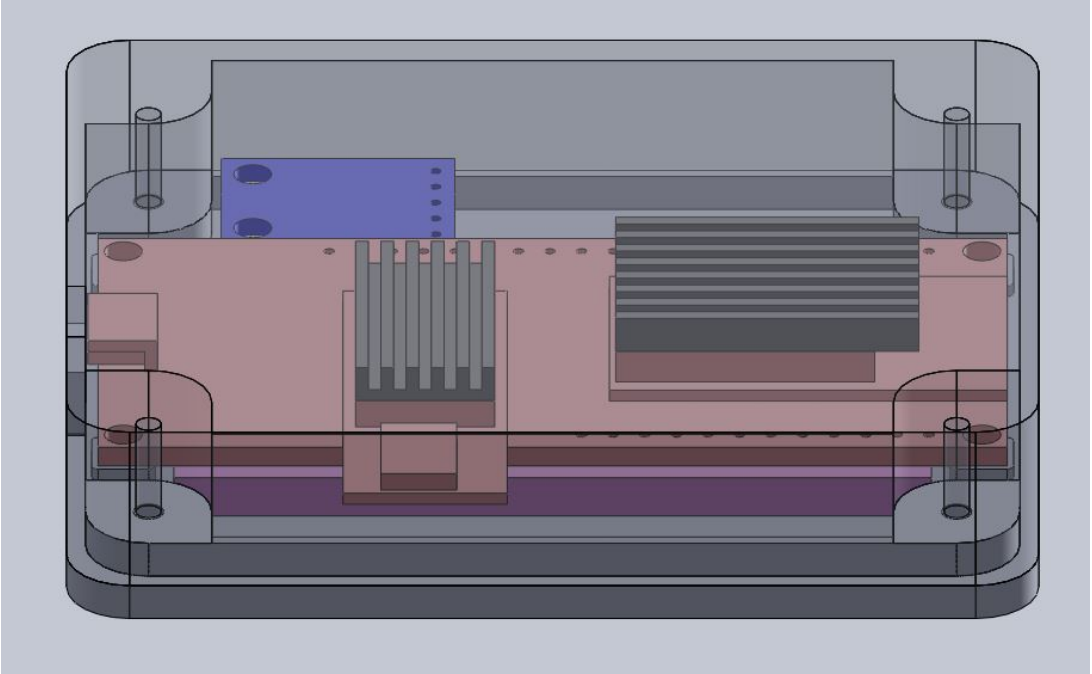
Na obrázku 18 lze vidět schéma zapojení magnetometru SSH1106 a OLED displeje QMC5883L do ESP32 UWB Pro pomocí I2C sběrnice.



Obrázek 18: Schéma zapojení magnetometru a OLED displeje do ESP32 UWB Pro.

6.5 Model 3D krabičky pro HW

V této podkapitole představím 3D návrh krabičky pro hardware lokátor. Návrh sestavené krabičky se nachází níže na obrázku 19. Dno a víčko krabičky by se vyrábělo na 3D tiskárně. V kusové či prototypové výrobě je 3D tisk mnohem rychlejší a levnější. V porovnání s jinou technologií, například technologií vstřikování polymeru, která by obnášela výrobu ocelové formy. Což je mnohonásobně dražší, zdlouhavější a případné chyby v návrhu nebo úpravy se řeší velmi obtížně.



Obrázek 19: Návrh krabičky pro hardware.

Popis jednotlivých komponent:

- Dno krabičky: Dno krabičky je šedou barvou.
- ESP32 UWB Pro: ESP32 UWB Pro je červenou barvou.
- Magnetometr: Magnetometr je modrou barvou.
- Chladiče: Chladiče pro UWB čip a procesor ESP32-WROVER-B jsou tmavě šedou barvou.
- Akumulátor: Akumulátor je fialovou barvou.
- Víčko krabičky: Víčko krabičky je průhledné.

Krabička, chladiče i akumulátor jsou pouze návrhem, jak by mohl vypadat budoucí lokátor ve finální podobě.

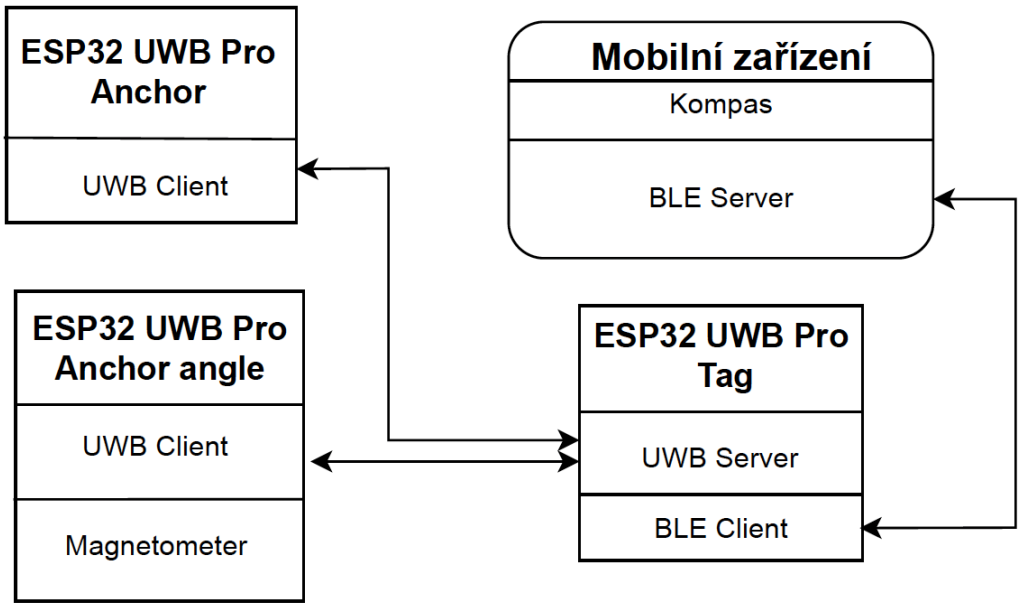
7 Implementace

V této kapitole popíši jednotlivé části kompenzační pomůcky, která umožní nevidomému uživateli nalézt stan a navigovat nevidomého uživatele pomocí mobilní aplikace a zařízení, které má u sebe. Toto zařízení označuji Tag a další dvě se nechají ve stanu s určitými pravidly, tyto pravidla budou popsána níže v kapitole. Dvě zařízení, které zůstávají ve stanu označuji Anchor a Anchor angle. Výhodou tohoto projektu je, že se uživatel neorientuje podle zvuku, který si přehraje, ale naviguje se podle instrukcí v aplikaci. Systém by měl navigovat na větší vzdálenosti a mobilní aplikace je uzpůsobena zpětné vazbě pro nevidomé uživatele.

Dále bude popsán algoritmus pro výpočet směru a vzdálenosti a jeho jednotlivé části. Z lokačních technologií, kterým jsem se věnoval v kapitole 3, jsem zvolil UWB, protože nejvíce splňuje požadavky na kompenzační pomůcku. A to zejména v těchto bodech.

- Přesnost: 0,1 – 0,5 m, mým cílem 0,5 m.
- Dosah: 10 – 200 m, mým cílem alespoň 50 m.

Na obrázku 20 je architektura systému. Mobilní zařízení má uživatel u sebe společně s ESP32 UWB Pro Tag. ESP32 UWB Pro Anchor a ESP32 UWB Pro Anchor angle se nechává ve stanu na vzdálenost jednoho metru nasměrovaných na sebe kvůli lokačnímu algoritmu.



Obrázek 20: Architektura systému.

Systém jsem nemohl navrhnout se třemi zařízeními, kde by na obrázku 20 chybělo jedno zařízení Anchor. Bez konstrukce vlastní desky plošných spojů

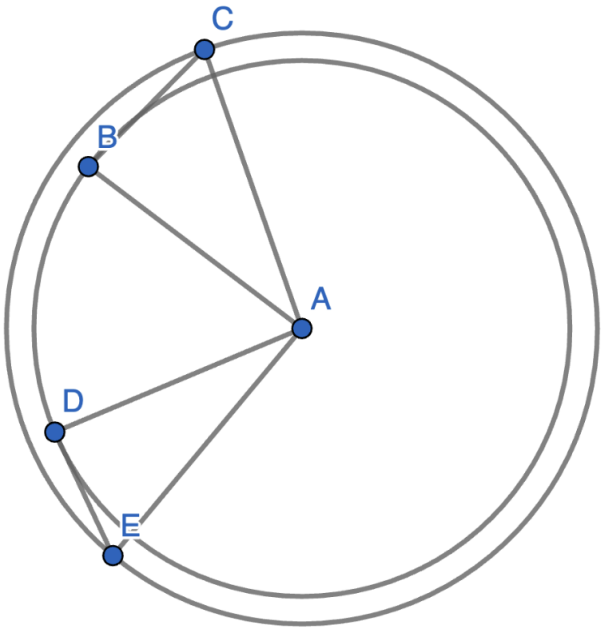
jsem nemohl vyzkoušet metodu Angle of Arrival, o které si myslím, že by mohla fungovat. Do konstrukce jsem se nepouštěl hlavně ze dvou důvodů:

- vysoká cena;
- časová náročnost.

7.1 Lokační algoritmus

V této podkapitole představím lokační algoritmus, jednotlivé jeho části a fungující podmínky. Pří návrhu tohoto algoritmu jsem se snažil vycházet z co nejméně informací ze senzorů, (nepřidávat do zařízení spoustu dalších senzorů kvůli informaci navíc) navrhnutého HW především, abych zachoval nízkou cenu celého systému.

Ze dvou zařízení dostávám jednu vzdálenost, což je pouze kružnice jedné vzdálenosti. Proto jsem přidal další zařízení na hledané místo s podmínkou, že od se musí být vzdáleny jeden metr. Kdyby zařízení ve stanu byly u sebe, tak bych nezískal informaci na víc. Vzdálenosti by byly blízko u sebe a splývaly by do jedné. Pokud budou dvě zařízení ve stanu alespoň metr od sebe dostanu dvě kružnice, ze kterých vypadá určení směru realistické, ale není to tak. Z těchto informací se dají spočítat všechny úhly v trojúhelníku, ale i přes tento vzniklý pomyslný trojúhelník se může pořád točit kolem bodu A, jak je znázorněno na obrázku 21. Informace o směru je stále neznáma. Trojúhelník ABC a ADE je stejný, ale na úplně jiném místě.



Obrázek 21: Situace s trojúhelníkem, tří vzdálenosti.

Úhly v trojúhelníku lze spočítat, když známe všechny strany pomocí Kosinové věty. Speciálním případem kosinové věty je Pythagorova věta, která však platí pouze pro pravoúhlý trojúhelník. Podle kosinové věty pro každý rovinný trojúhelník ABC s vnitřními úhly α, β a γ a stranami a, b a c platí kosinová věta:

$$a^2 = b^2 + c^2 - 2bc \cdot \cos(\alpha)$$
$$b^2 = a^2 + c^2 - 2ac \cdot \cos(\beta)$$
$$c^2 = a^2 + b^2 - 2ab \cdot \cos(\gamma)$$

Pokud známe délku všech stran a potřebujeme zjistit úhel, můžeme vzorce upravit na.

$$\cos(\alpha) = \frac{b^2 + c^2 - a^2}{2bc}$$
$$\cos(\beta) = \frac{a^2 + c^2 - b^2}{2ac}$$
$$\cos(\gamma) = \frac{a^2 + b^2 - c^2}{2ab}$$

```
1  num cosineTheoremAlfa(num a, num b, num c) {
2    var cosA = (pow(b, 2) + c * c - a * a) / (2 * b * c);
3    var radA = acos(cosA);
4    return radA / (pi / 180);
5  }
```

Zdrojový kód 1: Funkce počítající úhel na základě délek stran, podle kosinovy věty.

Trojúhelníková nerovnost je matematická věta: V každém trojúhelníku platí, že součet délek kterýkoliv dvou stran je vždy větší než délka třetí strany. Trojúhelníková nerovnost.

$$|a - b| \leq c \leq a + b$$

Trojúhelníkové nerovnosti se využívá při kontrole přijmutích dat z Tagu v mobilní aplikaci. Přijímaná data mohou být chybná nebo nepřesná, proto se kontroluje trojúhelníková nerovnost. Jestliže platí trojúhelníková nerovnost, může algoritmus pokračovat dál ve výpočtu úhlů. Pokud neplatí, vyhledá se nejmenší strana trojúhelníku. Až se najde, výpočet pokračuje tím, že zvětšuje nejmenší stranu a zkouší trojúhelníkovou nerovnost. Pokud nerovnost platí, pokračuje dál výpočet algoritmu, pokud se nepodaří najít v toleranci, kterou jsem empiricky zvolil na maximální hodnotu jeden metr s inkrementem deset centimetrů, výpočet končí a čeká se na nová data. Tato funkcionalita je vidět na zdrojovém kódu 2.

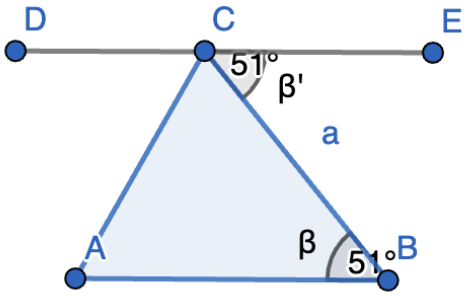
```
List<num> tryFindSolution(List<num> tri, int minIndex,
    [bool log = false, int tolerance = 10]) {
  if (log) print('hodnota na zacatku: ${tri[minIndex]}');
  for (int i = 1; i < tolerance + 1; i++) {
    tri[minIndex] = tri[minIndex] + 0.1;
    if (log) print('nova hodnota: ${tri[minIndex]}, inkrement:
        0.1');
    if (triangularInequality(tri[0], tri[1], tri[2])) {
      return tri;
    }
  }
  return [];
}
```

Zdrojový kód 2: Funkce hledající trojúhelník splňující trojúhelníkovou nerovnost.

Pro určení směru algoritmus používá na zařízení anchor angle magnetometr, který odesílá azimut zařízeních ve stanu do Tagu, který odesílá data do mobilní aplikace. V mobilní aplikaci se používá senzor magnetometru z telefonu, aby ze získávala nejsprávnější data, je potřeba mít mobilní telefon mít ve vodorovné pozici displejem nahoru. Stejná podmínka platí i na anchor angle ve stanu, aby data byla co nejpřesnější, musí být zařízení ve vodorovné pozici dál od kovových předmětů, které by mohly rušit magnetické pole, tudíž by byly výsledky špatné.

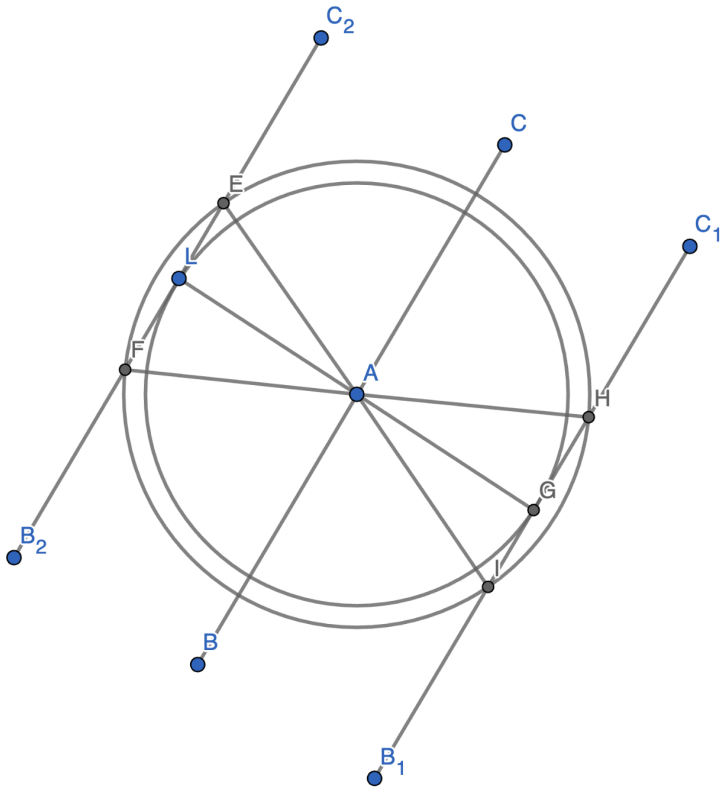
Když máme dvě vzdálenosti a k tomu údaje o azimutech, nastane situace na obrázku 23. Bod A je člověk hledající stan. Úsečky B_1C_1, B_2C_2 jsou znázorněny azimuty zařízeních ve stanu, které připadají v úvahu, aby byla splněna podmínka vzdáleností, které jdou získány pomocí UWB. Úsečka BC je azimut mobilního zařízení, tento azimut nemusí být rovnoběžný, jak je znázorněno na obrázku 23, ale tento úhel jde vypočítat. Z této situace již lze určit směr s padesáti procentní přesností. Potenciální cíl na obrázku představuje bod L nebo G.

Ve výpočtu rovnoběžného azimutu uživatele s Anchorama se využívá vlastnosti shodnosti úhlu. Trojúhelník ABC úhel β u vrcholu B tak úhel β' u vrcholu C a strany a je stejný. Znázorněno na obrázku 22.



Obrázek 22: Shodný vnější úhel.

Na obrázku 23 je znázorněna celá situace. Určit, zda bod L nebo G je správný bez další dodatečné informace, nejde. Algoritmus tuto situací řeší tak, že si kontroluje, zda se přibližuje či vzdaluje, pokud se přibližuje, zmenšuje se vzdálenost. Algoritmus zvolil správný směr, pokud se uživatel vzdaluje, vzdálenost do cíle se zvětšuje. Algoritmus zvolil špatný směr a je potřeba jít na druhou stranu.



Obrázek 23: Vzdálenosti se všemi daty.

Další situace, která nastává je, že algoritmus nerozezná, zda anchor angle je vlevo nebo vpravo. Abych tuto situaci více popsal na obrázku 23, řekněme, že správný cíl je bod L. Jde se od bodu A. Ale algoritmus nepozná, zda je situace přesně jako obrázku nebo jsou body L a E prohozeny. To způsobuje, že uživatel může cíl o kousek minout, ale jakmile se začne zase vzdalovat, algoritmus tuto situaci rozezná. Řekne uživateli, aby se otočil, v tuto chvíli jde správně do cíle. Tato situace jde řešit algoritmem, jak by se přibližoval, musel by sledovat, jestli se nepřibližuje k jednomu bodu více než ke druhému. Podle situace měnit výpočet. Toto z časových důvodů nebylo implementováno.

7.2 Implementace firmware

V této kapitole bych chtěl popsat možnosti vývojových prostředí pro firmware, jaké jsem použil a proč. V dalších podkapitolách jsou popsány implementační detaily jednotlivých firmware systému.

Arduino IDE (Integrated Development Environment) je oficiální softwarové vývojové prostředí poskytované společností Arduino. Jedná se o jednoduchý a uživatelsky přívětivý kódový editor, který umožňuje psaní, kompilaci a nahrávání kódu do desek Arduino. Arduino IDE podporuje programovací jazyk Arduino, který je založen na jazycích C a C++. Arduino IDE je nejjednodušší a nejvhodnější volbou pro začátečníky, kteří chtějí programovat desky Arduino. Obsahuje vestavěný správce knihoven a mnoho ukázkových kódů, které uživatelům pomáhají rychle začít s jejich projekty [37].

PlatformIO je open-source ekosystém pro vývoj v oblasti Internetu věcí (IoT). Podporuje širokou škálu vývojových platforem, včetně Arduino, ESP8266, ESP32, STM32 a mnoha dalších. PlatformIO není omezeno na konkrétní integrované vývojové prostředí (IDE) nebo kódový editor. Lze ho použít s různými populárními kódovými editory, jako je Visual Studio Code, Atom nebo CLion. PlatformIO kombinuje různé vývojové nástroje a knihovny, což umožňuje vývojářům pracovat na různých hardwarových platformách ve stejném prostředí [36].

Použil jsem PlatformIO rozšíření do VS Code z důvodů:

- PlatformIO je všestrannější se spojením VS Code, je zde spoustu klávesových zkratek, které zná programátor z jiných projektů.
- PlatformIO hlavně umožňuje mnohem lepší logické členění projektu pro HW, specifičtější nastavení MCU. Složitější nebo větší projekty si přes Arduino IDE neumím představit.

7.2.1 Tag

Tag je součást celého systému, kterou má uživatel u sebe. Na této součástce se vy-komunikovávají vzdálenosti z Anchorů a informace o azimutu. Tag je propojený s mobilní aplikací pomocí Bluetooth Low Energy (BLE). Metody zpráv BLE jsou:

- Notify: je jednosměrný způsob odesílání dat. Notify je vhodný pro nepřetržitý proud dat např ze senzorů.
- Indicate: je dvousměrný způsob komunikace. Je podobný jako notify s tím rozdílem, že se zpráva na přijímacím zařízení potvrzuje.
- Write: je dvousměrný způsob komunikace. Tento typ zpráv je vhodný pro situace, kdy je potřeba vyžádat nebo předat konkrétní informace nebo příkazy mezi zařízeními.

Tag posílá data do aplikace každých 600 milisekund, kde potom probíhá výpočet. Tag posílá data pomocí typu zprávy notify. Metoda notify je jednosměrný způsob odesílání dat ze zařízení. Když zařízení pošle notifikační zprávu, centrální zařízení je informován o nových datech.

Aby výpočet cíle byl co nejpřesnější, je třeba mít Tag s mobilem v jedné dlani, jestliže je mobil v jedné dlani a Tag v druhé, vzniká nepřesnost ve výpočtu a ztrácí se přesnost.

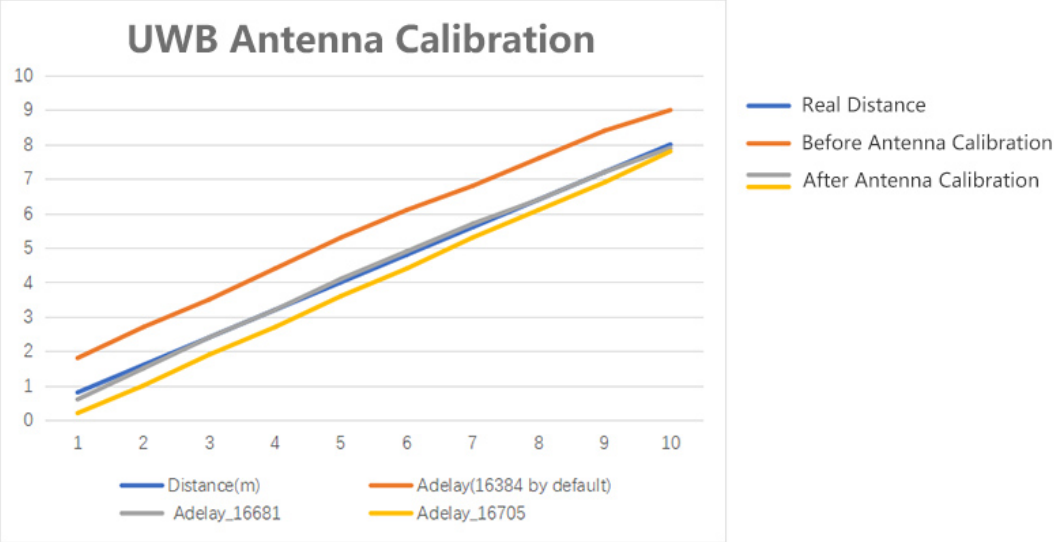
7.2.2 Anchor

Anchor je část systému, která má funkci UWB Clienta. Komunikuje s Tag, jehož funkcí je UWB server. Pro přesnější vzdálenost je potřeba udělat kalibraci antény. Zpoždění antény DW1000 je interní součástí čipu a není zahrnuto v TOF, ale je zahrnuto do zpoždění šíření od časového razítka vysílání do časového razítka příjmu zprávy [38].

$$tMeasured = tADTX + TOF + tADR$$

- ToF = Doba letu
- tMeasured = Časový rozdíl mezi vysíláním a přijímáním od časového razítka.
- tADTX = Zpoždění vysílací antény
- tADR = Receive antenna delay

Na obrázku 24 je porovnáni přesnosti po kalibraci. Tuto kalibraci jsem udělal na obou zařízeních Anchor.



Obrázek 24: Výsledek přesnosti po kalibraci [38].

7.2.3 Anchor angle

Anchor angle je část ze systému, která má funkci UWB Clienta a magnetometr. Komunikuje s Tag, který je UWB server. Aby bylo možné určit polohu, použil jsem magnetometr na zařízení. Odkud dostávám azimut nasměrování zařízeních ve stanu. Aby bylo možné posílat přes UWB čip další informaci, musel jsem modifikovat použitou knihovnu. Knihovna, kterou jsem upravil tak, že jsem přidal

posílání další zprávy vychází z `arduino-dw1000 library`, která nabízí základní funkce pro použití čipů/modulů DW1000 společnosti Decawave s Arduinem. Tuto knihovnu ještě upravila společnost Makerfabs, já jsem vycházel z upravené verze společnosti Makerfabs.

Použitý magnetometr QMC5883L je velmi citlivý na kovové předměty. U výběru magnetometru jsem hlavně z časových důvodu nevyzkoušel více typů a výrobců, abych našel nejvhodnější pro celý systém. Hlavní nevýhodou zde vidím velkou citlivost, i v klidovém stavu v téměř ideálních podmínkách má výsledný azimut velký rozptyl. Abych tento rozptyl, zmenšil průměruji naměřené hodnoty, po odeslání první hodnoty nastavím minulý průměr a resetuji počítadlo. Toto dělám z důvodu, aby po zapnutí a přemístění zařízení ve stanu adekvátně zareagovalo. Tato funkcionalita je vidět na zdrojovém kódu 3.

```
1  void compassLoop()
2  {
3     compass.read();
4     countAzimuth++;
5     azimuth = compass.getAzimuth();
6     azimuth = azimuth < 0 ? azimuth + 360 : azimuth;
7     akuAzimuth += azimuth;
8
9     if (currentMillis - startMillis >= period)
10    {
11       akuAzimuth = akuAzimuth / countAzimuth;
12       send_data(akuAzimuth);
13       countAzimuth = 1;
14       akuAzimuth = akuAzimuth;
15       Serial.print("Azimut ");
16       Serial.println(akuAzimuth);
17
18       startMillis = currentMillis;
19    }
20 }
```

Zdrojový kód 3: Funkce akumulující hodnoty z magnetometru, a odesílající každou vteřinu.

Pro zajištění přesnosti a spolehlivosti měření magnetického pole je kalibrace magnetometru nezbytnou součástí inicializačního procesu. Po zapnutí zařízení je magnetometr automaticky kalibrován pomocí získaných dat. Pro provedení kalibrace bylo provedeno několik měření magnetometru, přičemž každé měření bylo provedeno desetkrát. Následně byl vypočten průměr naměřených hodnot, který byl použit jako kalibrační faktor.

Tímto postupem je zajištěno, že uživatel nemusí provádět manuální kalibraci magnetometru. Namísto toho je kalibrace provedena automaticky při každém spuštění zařízení. Tato metoda poskytuje vysoce přesné výsledky měření a zajišťuje konzistentní výkon magnetometru bez nutnosti zásahu uživatele.

Díky automatické kalibraci je uživateli zajištěna jednoduchá a bezproblémová zkušenost při používání zařízení, aniž by bylo nutné řešit složitosti spojené s ruční kalibrací magnetometru.

```
1    compass.setCalibration(-912, 867, -916, 962, 0, 661);
```

Zdrojový kód 4: Kalibrace magnetometru s empirickými daty.

7.3 Mobilní aplikace

Při volbě, jestli aplikaci psát jako nativní, nebo multiplatformní, byla volba jednoznačná. Rozhodl jsme se ji naprogramovat jako multiplatformní, jelikož u aplikace jsem neočekával potřebu vysokého výkonu a chtěl jsem pokrýt co největší spektrum uživatelů, kterým by tento vzniklý produkt mohl pomoci. Z multiplatformních frameworků jsem si vybral Flutter, a to z důvodů:

- Je nejpoužívanější framework v letech 2021 a 2022 jak vyplývá z obrázku 2.
- Jazyk Dart není dynamicky typový.
- Dart byl navržen s důrazem na lepší výkon a optimalizace.
- Dart má syntaxi podobnou jiným jazykům jako Java nebo C#.

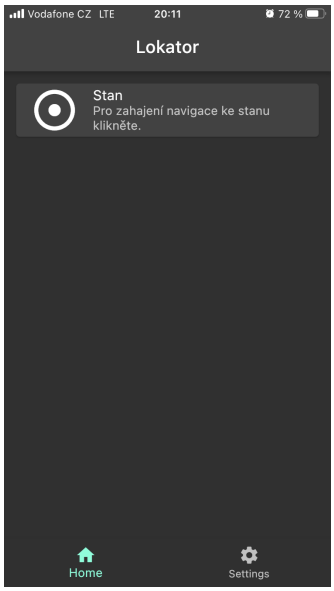
Mobilní aplikace je jednoduchá co se týče uživatelského prostředí, skládá se ze tří obrazovek, které jsou vidět níže. Aplikace nevyžaduje registraci, z důvodu co možná nejvíce ulehčit práci s aplikaci. Obrazovka po spuštění je vidět na 25. Kde se dá spustit navigace nebo přejít do nastavení.

Obrazovka s nastavením ve vidět na 26. Na této obrazovce lze nastavit:

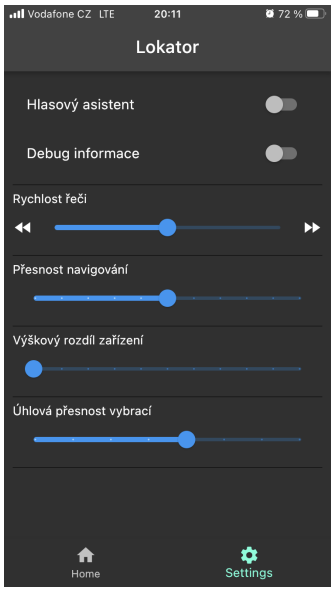
- Hlasový asistent: Který je buď zapnutý nebo vypnutý, ovládá hlasového asistenta aplikace.
- Debug informace: Jsou buď vypnuté nebo zapnuté, tyto dodatečné informace jsou zobrazeny na stránce s navigací, která je vidět na obrázku 27. Tyto informace jsou dobré při vývoji, zobrazují více informací k výpočtu a připojení jednotlivých Anchorů.
- Rychlost řeči: Nastaví rychlost řeči hlasového asistenta. Nevidomý často používají vyšší rychlost.
- Přesnost navigování: Nastaví, s jakou přesností chce, aby byl označen cíl. Minimální hodnota je 0,5 m nejvyšší 1,5 m, kroky jsou po 0,1 m.

- Výškový rozdíl zařízení: Nastaví, v jaké výšce jsou hledané Anchory. Toto je důležité pro přesnost navigování. UWB odesílá vzdálenost mezi zařízeními a je potřeba vzít v potaz i výškový rozdíl pro přesnost. Minimální hodnota je 0 což znamená že jsou ve stejné úrovni. Maximální hodnota je 2 m.
- Úhlová přesnost vibrací: Znamená, jaká odchylka při navigaci je v toleranci. Aplikace, pokud je nasměrována správným směrem, vibruje a tím uživatel ví, že má jít rovně. Ale při pohybu a s novými daty se často mění, proto je tu nastavení této tolerance. Možné hodnoty jsou 2° – 16°.

Obrazovka s navigací je vidět na obrázku 27. Zde je veliká šipka ukazující směr. Vzdálenost a úhel k cíli a stav připojení s Tag. Vzdálenost a směr říká hlasový asistent po ujití 1,9 m. Pro zopakování minulé instrukce slouží takzvaný „double tap“, stačí dvakrát poklepat a zopakuje se poslední instrukce. Pokud je hlasový asistent vypnutý navigace nemluví, ani nejde zopakovat poslední instrukci. Mobil vibruje, pokud je natočený správným směrem, takže zde je dobré mít mobil ve vertikální poloze obrazovkou vzhůru pro nejlepší výsledky.



Obrázek 25: Obrazovka aplikace po otevření.



Obrázek 26: Obrazovka aplikace s nastavením.

Obrázek 27: Obrazovka aplikace s navigací.

7.4 Možné rozšíření projektu

Mezi budoucí rozšíření projektu bych zařadil:

- Rozšíření knihovny pro UWB, aby bylo možné zjistit vzdálenost mezi Anchory. Přes Anchor Angle odeslat i tuto informaci do Tagu a z Tagu následně do mobilní aplikace. Toto rozšíření samozřejmě nese i svá rizika například když nedají se Anchory dostatečně od sebe, aby bylo možné

počítání směru a vzdálenosti. Související vylepšení je spíše mechanického rázu, aby bylo eliminováno špatné natočení Anchorů ve stanu, tak do navržené krabičky přidám mechanické „vysunutí“, které by hlídalo natočení a případně i minimální vzdálenost mezi Anchory. Toto by mohlo být řešeno pomocí teleskopického výsuvu.

- Přidat ke každé části systému akumulátor, který byl znázorněn v návrhu 3D krabičky na obrázku 19 fialovou barvou. A potřebnou implementaci ke správě akumulátoru. Aktuální stav je napájen z powerbanky. Proud při činném stavu jednoho zařízení: 0,25 – 0,3 A, napětí 5,05V. Spotřeba energie se počítá:

$$SpotřebaEnergie(Wh) = Proud(A) \cdot Napětí(V) \cdot Čas(h)$$

  Snad všechny dnešní powerbanky udávají svojí kapacitu v mAh. Pokud by jsme chtěli spočítat spotřebu, můžeme použít výpočet:

$$Q(mAh) = 1000 \cdot \frac{E(Wh)}{U(V)}$$

  Po výpočtu dostaneme spotřebu zaokrouhleně 300 mAh.

- Aplikace by si mohla ukládat GPS souřadnice pro dosažení většího dosahu. Zde je předpoklad, že se vychází ze stanu, můžeme si uložit pozici a ideální přesnost GPS 5 m zde nevadí, tato přesnost může být i horší. Když se přiblížilo do dosahu lokátoru, pak už se může použít UWB pro přesné navigování. Výhoda je, že by nevzrostla cena kompenzační pomůcky, protože GPS mají dnes již snad všechny mobilní telefony.

- Pokud by se do systému přidal GPS a celý systém rozšířil o komunikaci LoRa, tak by dosah mohl být v kilometrech, záleží na spoustě parametrech. Technologie LoRa nebyla uvedena v Teoretické části.

- Udělat průzkum možností filtrovat přijatá data, například při ztrátě spojení tak přicházejí nerelevantní data. Pro lepší výpočty by bylo vhodné filtrovat přijatá data. Jedna z možností by mohla být použít metodu Kalmanův filtr.

- Detekci Anchoru vlevo a vpravo. Při navigaci přidat detekci zařízení vlevo a vpravo. Algoritmus předpokládá nějaké rozdělení, ale toto není dané, uživatel může zařízení položit pokaždé jinak a jít z jiného směru, kdy se strany opět změní a stan se těsně mine. Podrobněji je tato situace popsána v kapitole 7.1.

- Přidat možnost do aplikace přidat více zařízeních pro lokaci dalších míst/-věcí, aktuální stav je jedna věc, stan s předem definovaným názvem BLE, po kliknutí na navigování se vyhledá název BLE tagu a na ten se připojí. Po prvním připojení si údaje mobilní aplikace uloží, aby příští připojení bylo rychlejší.

8 Testování

V této kapitole se budu věnovat testování celého systému a popíšu jednotlivé nedostatky, které jsem při testování zjistil a jak jsem tyto nedostatky případně řešil, jak vznikly.

Kde to bylo možné, tak vývoj probíhal na simulátorech, výhodou simulátorů je, že se může vyzkoušet spousta velikostí a jiných operačních systémů. Bohužel na simulátorech nejsou všechny funkce dostupné jako na reálném zařízení například Bluetooth, vibrace. Bluetooth není dostupné v simulátorech, a proto se musí vše spojené s Bluetooth testovat na reálném zařízení. Což pro důkladné otestování by bylo potřeba mít hodně zařízení ideálně s jinými verzi operačních systémů. Veškerý vývoj probíhal na mobilním telefonu, který má označení iPhone SE (1. generace) se systémem iOS verzí 15.7.3., což v době vývoje nebyl nejposlednější verze systému iOS.

8.1 Multiplatformní

K testování probíhalo na dalších dvou zařízeních:

- Huawei P20 Lite s verzí Androidu 9.1.0
- iPhone SE (3. generace) se verzí iOS verzí 16.4.

Aby bylo možné aplikaci nahrát do telefonu, je potřeba povolit si vývojářský režim, zde se postup může lišit u různých verzí operačních systému. Vývojový režim se zapne následovně:

- Android: V „Nastavení“ dále „systém“ „O telefonu“ „Číslo sestavení“ klikat, než se zobrazí zpráva jste vývojář.
- iOS: V „Nastavení“ dále „Soukromí a zabezpečení“ dále „Vývojový režim“ a zapnout.

U testování na reálném zařízení byl problém na Androidu s komunikací s Bluetooth Low Energy (BLE) z Tag. Použil jsem knihovnu `flutter_blue` verzi 0.8.0. Problém byl s nastavením Maximum Transmission Unit (MTU). Systém iOS nepovoluje změnu MTU, na iOS je nastavené MTU na 185 bajtů. Na Androidu je výchozí nastavení 23 bajtů. Maximální velikost MTU pro Android je 512 bajtů.

Kvůli zákazu na iOS měnit MTU a na Androidu potřeby zvětšit, protože z Tag posílám delší zprávy. Musel jsem použít platformově závislý kód, který je vidět na zdrojovém kódu 5. Na kódu je vidět if blok, kde se kontroluje stav připojení zařízení Bluetooth, pokud se změní na připojeno (connected) a pokud je platformou Android, požádá se o větší MTU. Pokud platforma není android, pokračuje se else větví dál bez požadavku na změnu MTU. Tento if je součástí událostního posluchače (event listener) pro stav zařízení Bluetooth.

```
if (state == BluetoothDeviceState.connected) {
      if (Platform.isAndroid) {
        // Android-specific code
        int mtuFirst = await dev.mtu.first;
        await dev.requestMtu(182);
        dev.mtu.listen((mtu) {
          if (mtu == 182) {
            _discoverServices(dev);
          }
        });
      } else {
        _discoverServices(dev);
      }
    }
```

Zdrojový kód 5: Platformově závislý kód pro vyjednání MTU pro BLE.

Při čtení hodnoty kompasu v mobilním telefonu bylo potřeba použít platformově závislý kód, který je vidět na zdrojovém kódu 6. Na platformě Android nemusí být kompas dostupný vůbec, telefon tento senzor nemá, nebo není zkalibrovaný a systém neposkytne data. Na zdrojovém kódu níže je vidět kontrola platformy a dostupnosti kompasu. Kontrolovat dostupnost i na platformě iOS by šlo také, ale nikdy se mi během testování nestalo, že by byl nedostupný. Na androidu se tato chyba stávala častěji. I s ohledem na to, že jsem aplikaci na platformě testoval dva dny, neboť jsem neměl déle k dispozici reálné zařízení.

```
    // Android-specific code
    if (Platform.isAndroid) {
      heading = event.heading;
      if (heading == null) {
        isError = true;
        errorMessage = compassUnavailable;
        FlutterBeep.beep(false);
        voiceService.speak(errorMessage);
      }
    } else {
      heading = event.heading;
    }
```

Zdrojový kód 6: Platformově závislý kód pro dostupnost kompasu.

8.2 Vzdálenost

U testování dosahu signálu jsem zjistil velký faktor ovlivňující vzdálenost a to výška zařízení, ve které vysílá, to znamená jestli jsou položena na betonové zemi, trávě nebo zavěšena v nějaké výšce.

Testovaná zařízení byla položena na zemi. V rámci testování a vývoje bylo zakoupeno existující řešení AirTag a Chipolo, proto byly zahrnuty také do měření. Výsledky měření jsou vidět v tabulce 3.

Tabulka 3: Tabulka dosahu lokátorů na zemi.

Vzdálenost	AirTag	Chipolo ONE	Lokator
5 m	ANO	ANO	ANO
10 m	ANO	ANO	ANO
15 m	ANO	ANO	ANO
20 m	ANO	ANO	ANO
25 m	ANO	ANO	ANO
30 m	NE	ANO	ANO
35 m	NE	ANO	NE
40 m	NE	ANO	NE
45 m	NE	NE	NE
50 m	NE	NE	NE
55 m	NE	NE	NE
60 m	NE	NE	NE
65 m	NE	NE	NE

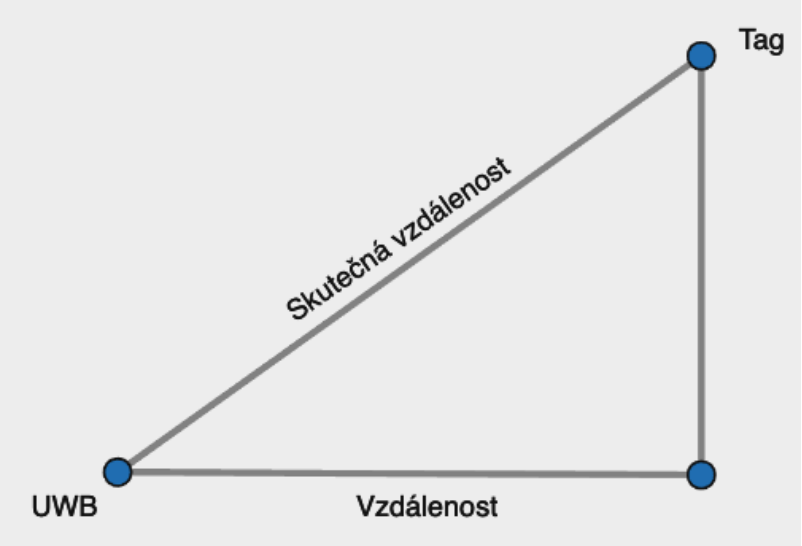
Další měření probíhalo ve výšce 46 cm nad zemí, výsledky jsou v tabulce 4, z tabulky vyplývá, že dosah lokátoru i AirTag ze zvýšil o polovinu.

Tabulka 4: Tabulka dosahu lokátorů nad zemí ve výšce 46 cm.

Vzdálenost	AirTag	Chipolo ONE	Lokator
5 m	ANO	ANO	ANO
10 m	ANO	ANO	ANO
15 m	ANO	ANO	ANO
20 m	ANO	ANO	ANO
25 m	ANO	ANO	ANO
30 m	ANO	ANO	ANO
35 m	ANO	ANO	ANO
40 m	ANO	ANO	ANO
45 m	ANO	ANO	ANO
50 m	ANO	ANO	ANO
55 m	NE	ANO	ANO
60 m	NE	ANO	ANO
65 m	NE	NE	NE

8.3 Přesnost

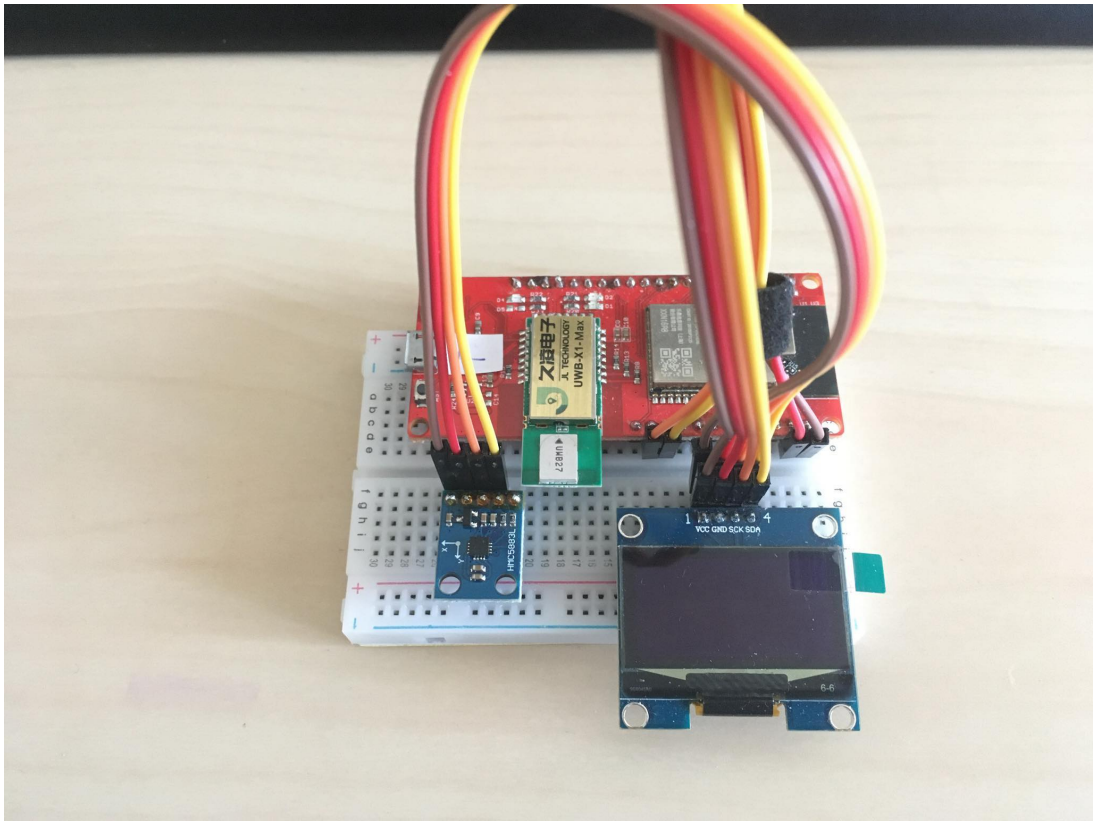
Při testování přesnosti jsem přišel na mnou opomenout věc a to, že jsem k lokalizaci přistupoval, že se odehrává v rovině a neuvědomoval si, že vzdálenosti, které dostávám, zahrnují i výškový rozdíl. Situaci je zobrazena na obrázku 28. Chyba byla v tom, že při přiblížení téměř splynou body UWB a TAG. Já jsem to tak uvažoval. Ale ve skutečnosti jsou někde položené Anchory, mobil s Tagem jsou v ruce a tato vzdálenost je skutečná strana označená „Skutečná vzdálenost“, jak je vidět na obrázku 28.



Obrázek 28: Nákres výškového rozdílu.

Proto jsem do nastavení aplikace přidal možnost nastavit tuto hodno pro dosažení lepší přesnosti navigování. Jak je vidět na obrázku 26.

Vysoká citlivost magnetometru. Použitý magnetometr je velice citlivý, stačí kovový předmět v blízkosti na deset centimetrů a vrácený azimut z magnetometru je úplně jiný (chybný) a v tuto chvíli je nemožné správné nalezení stanu. Už při testování se objevil problém s vysokou citlivostí, v testovacím sestavení v napájecím poli stačilo posunout o jednu zdířku a výsledky byli úplně jiné. Proto se při návrhu krabičky magnetometr umístil co nejdál od ostatních součástí. Prototyp v napájecím poli je vidět na obrázku 29. Na prototypu je připojený i OLED display, který sloužil především pro testovací účely.



Obrázek 29: Prototyp Anchor Angle v napájecím poli.

8.4 Interakce s uživatelem

Testujícím uživatelem byl můj nevidomý kamarád, sice byl sám, ale pro zpětnou vazbu dostačující. Testování proběhlo na zahradě za bytovým domem, kde bydlí, kde se cítil pohodlně a bezpečně.

Jednou z prvních výtek bylo, že by aplikace po otevření mohla rovnou navigovat, než aby se klikalo ještě na kartu stanu pro začátek navigace. Po konzultaci jsme dospěli k závěru, že pokud by se aplikace používala jenom k lokaci stanu je tento krok zbytečný, ale pokud by byla možnost to využít i na lokaci dalších věcí, je tento krok správný.

Hmatová zpětná vazba po chvilce užívání, říkal, že by bylo úžasné, kdyby aplikace uměla různé vibrace například pokud je daleko od cíle více jak 5 metrů, aby vibrace nebyly tak intenzivní jenom slabé, a při přiblížení k cíli vibrace byli intenzivnější. Tady bohužel záleží na mobilních telefonech, telefon, na kterém probíhal vývoj, umožňuje jeden druh vibrací. Tahle funkcionalita by šla doimplementovat, jenom by nebyla dostupná na všech zařízeních, proto jsem neměl ji možnost testovat.

U hlasová zpětné vazba při navigování by ocenil mít možnost nastavení četnosti opakování instrukcí, aby i uživatel měl možnost tohoto nastavení. Mě osobně překvapilo, jak rychle chodil, rychlost chůze byla srovnatelná s vidomým člověkem.

Celá instrukce je poměrně dlouhá například „Cíl před vámi 23 metrů 53 stupňů napravo“, když by se zkrátila, mohla by se i častěji opakovat, stačily by i kratší a jednodušší instrukce například „23 metrů daleko 53 stupňů napravo“.

Obě tyto zpětné vazby na interakci s uživatelem jsou věcné.

Závěr

Tato diplomová práce se zabývala návrhem lokátoru, usnadňující nevidomým nalezení stanu. Lokátor byl realizován jako mobilní multiplatformní aplikace s důrazem na použití cílovou skupinou. Za použití specializovaného hardwaru, který jsem navrhnul. Výsledné řešení by mělo umožňovat lokaci na vzdálenost alespoň padesáti metrů s přesností na půl metru.

V teoretické části této práce byly popsány rozdíly přístupů vývoje mobilních aplikací a jednotlivé technologie tohoto vývoje. Dále technologie používající se obecně k lokaci, aby mohla být vybrána nejvhodnější technologie pro splnění cílů. Potom byly představeny jednotlivé metody určení směru pro lokaci.

V praktické části byli popsány hardwarové prostředky, který byly vybrány a použity pro realizaci této práce, včetně schémat elektrických zapojení. Další kapitola se věnuje implementaci jednotlivým částem celého systému včetně lokačního algoritmu počítající lokaci cíle.

Výsledkem této práce je fungující prototyp, který dosahuje přesnosti půl metru při správném nastavení algoritmu a dodržení postupu používání. Dosah zařízení byl změřen na šedesát metrů při umístění ve výšce čtyřicet šest centimetrů.

V budoucnu bych chtěl v systému implementovat nedostatky nalezené během testování, který se nestihly z časových důvodů implementovat. A rozšířit celý projekt o vybrané důležité části, které jsem ve své práci uvedl.

Conclusions

This thesis dealt with the design of a locator to make it easier for the blind to find their tents. The locator was implemented as a mobile multi-platform application with an emphasis on use by the target group. Using specialized hardware that I designed. The resulting solution should allow location at a distance of at least fifty meters with half meter accuracy.

In the theoretical part of this thesis, the differences in approaches to mobile application development and the various technologies in this field were described. Furthermore, the technologies commonly used for location were explored to select the most appropriate technology to meet the objectives. Then, the different methods of determining the direction for location were presented.

In the practical part, the hardware resources that were selected and used for the realization of this work were described, including electrical wiring diagrams. The next chapter is devoted to the implementation of the different parts of the whole system, including the location algorithm computing the target location.

The result of this work is a working prototype that achieves half-meter accuracy when the algorithm is set up correctly and the usage procedure is followed. The range of the device was measured at sixty meters when placed at a height of forty-six centimeters.

In the future I would like to implement in the system the flaws found during testing, which could not be implemented due to time constraints. And to extend the whole project with selected important parts that I have mentioned in my thesis.

A Obsah elektronických dat

Na samotném konci textu práce je uveden stručný popis obsahu elektronických dat odevzdaných v systému katedry informatiky spolu s textem.

text/
: Adresář s textem práce ve formátu PDF, vytvořený s použitím závazného stylu KI PřF UP v Olomouci pro závěrečné práce, včetně všech (textových) příloh, a všechny soubory potřebné pro bezproblémové vytvoření PDF dokumentu textu (v ZIP archivu), tj. zdrojový text textu a příloh, vložené obrázky, apod.

src/mobile
: Kompletní zdrojový kód mobilní aplikace. Obsahující soubor `README.md` s detailními informacemi o sestavení daného adresáře.

src/tag
: Kompletní zdrojový kód pro Tag, část kompenzační pomůcky. Obsahující soubor `README.md` s detailními informacemi o sestavení daného adresáře.

src/anchor
: Kompletní zdrojový kód pro Anchor, část kompenzační pomůcky. Obsahující soubor `README.md` s detailními informacemi o sestavení daného adresáře.

src/anchor-angle
: Kompletní zdrojový kód pro Anchor Angle, část kompenzační pomůcky. Obsahující soubor `README.md` s detailními informacemi o sestavení daného adresáře.

src/lib
: Adresář obsahuje dva ZIP archivy s knihovnami pro UWB originální bez úprav a druhý ZIP archív s knihovnou mnou upravenou pro odesílání azimutu.

data
: Adresář s videí z testování vždy obsahují dvojici záznamů: displeje mobilu z aplikace a zaznamenaní situace v terénu.

README.txt
: Soubor obsahující informace o sestavení jednotlivých částí z celého projektu.

Literatura

[1] Statista. Mobile operating systems' market share worldwide from 1st quarter 2009 to 2nd quarter 2023 [online]. 2023 [cit. 2023-07-12]. Dostupné z: https://1url.cz/NuQjj

[2] TechTerms. GPS Definition [online]. 2023 [cit. 2023-07-12]. Dostupné z: https://techterms.com/definition/gps

[3] GPS.gov. Official U.S. government information about the Global Positioning System [online]. 2023 [cit. 2023-07-12]. Dostupné z: https://www.gps.gov/systems/gps/space/

[4] GPS.gov. Official U.S. government information about the Global Positioning System [online]. 2023 [cit. 2023-07-12]. Dostupné z: https://www.gps.gov/systems/gps/performance/accuracy/

[5] TechTarget. Global System for Mobile communication [online]. 2023 [cit. 2023-07-12]. Dostupné z: https://www.techtarget.com/searchmobilecomputing/definition/GSM

[6] CVUT. Základní lokalizační metody v GSM [online]. 2023 [cit. 2023-07-12]. Dostupné z: http://access.fel.cvut.cz/view.php?cisloclanku=2006022801

[7] Wikipedie. Zigbee [online]. 2023 [cit. 2023-07-12]. Dostupné z: https://en.wikipedia.org/wiki/Zigbee

[8] Wikipedie. Bluetooth [online]. 2023 [cit. 2023-07-12]. Dostupné z: https://en.wikipedia.org/wiki/Bluetooth

[9] Wikipedie. Wi-Fi [online]. 2023 [cit. 2023-07-12]. Dostupné z: https://en.wikipedia.org/wiki/Wi-Fi

[10] TechTarget. RFID (radio frequency identification) [online]. 2023 [cit. 2023-07-12]. Dostupné z: https://www.techtarget.com/iotagenda/definition/RFID-radio-frequency-identification

[11] Wikipedie. Ultra-wideband [online]. 2023 [cit. 2023-07-12]. Dostupné z: https://en.wikipedia.org/wiki/Ultra-wideband

[12] infsoft. Technologies for Server-Based Indoor Positioning Compared: Wi-Fi vs. BLE vs. UWB vs. RFID [online]. 2023 [cit. 2023-07-12]. Dostupné z: https://www.infsoft.com/blog/technologies-for-server-based-indoor-positioning-compared/

[13] Android Developers. Application fundamentals [online]. 2023. Dostupné z: https://developer.android.com/guide/components/fundamentalss

[14] Wikipedie. iOS SDK [online]. 2023 [cit. 2023-07-12]. Dostupné z: https://en.wikipedia.org/wiki/IOS_SDK

[15] Microsoft. What is Xamarin? [online]. 2023 [cit. 2023-07-12]. Dostupné z: https://learn.microsoft.com/en-us/xamarin/get-started/what-is-xamarin

[16] React Native. Core Components and Native Components [online]. 2023 [cit. 2023-07-12]. Dostupné z: https://reactnative.dev/docs/intro-react-native-components

[17] Statista. Cross-platform mobile frameworks used by software developers worldwide from 2019 to 2022 [online]. 2023 [cit. 2023-07-12]. Dostupné z: https://www.statista.com/statistics/869224/worldwide-software-developer-working-hours/

[18] Wikipedie. Flutter [online]. 2023 [cit. 2023-07-12]. Dostupné z: https://en.wikipedia.org/wiki/Flutter_(software)

[19] web.dev. What are Progressive Web Apps? [online]. 2023 [cit. 2023-07-12]. Dostupné z: https://web.dev/what-are-pwas/

[20] web.dev. Add a web app manifest [online]. 2023 [cit. 2023-07-12]. Dostupné z: https://web.dev/add-manifest/s

[21] poslepu.cz. Přístupnost mobilních aplikací pro nevidomé uživatele [online]. 2023 [cit. 2023-07-12]. Dostupné z: https://poslepu.cz/pristupnost-mobilnich-aplikaci-pro-nevidome-uzivatele/

[22] iStores. Apple AirTag [online]. 2023 [cit. 2023-07-12]. Dostupné z: https://www.istores.cz/produkt/apple-airtag-1-ks-2197395

[23] Samsung. Chytrý přívěsek Galaxy SmartTag [online]. 2023 [cit. 2023-07-12]. Dostupné z: https://www.samsung.com/cz/mobile-accessories/galaxy-smarttag-black-ei-t5300kmegeu/

[24] Chipolo. The feature-rich and colourful choice for everyone [online]. 2023 [cit. 2023-07-12]. Dostupné z: https://chipolo.net/en/products/category/chipolo

[25] Tile. Mate Essentials 4-Pack [online]. 2023 [cit. 2023-07-12]. Dostupné z: https://www.tile.com/product/mate-essentials-4-pack/

[26] Apple. AirTag [online]. 2023 [cit. 2023-07-12]. Dostupné z: https://www.apple.com/airtag/

[27] Allegro. OLED displej 1,3 Arduino I2C SH1106 modrý [online]. 2023 [cit. 2023-07-12]. Dostupné z: https://allegro.cz/nabidka/oled-displej-1-3-arduino-i2c-sh1106-modry-12169191276

[28] Espressif. ESP32WROVERB Datasheet [online]. 2023 [cit. 2023-07-12]. Dostupné z: https://www.espressif.com/sites/default/files/documentation/esp32-wrover-b_datasheet_en.pdf

[29] Makerfabs. ESP32 UWB Pro [online]. 2023 [cit. 2023-07-12]. Dostupné z: https://www.makerfabs.com/esp32-uwb-high-power-120m.html

[30] Decaware. Product Overview [online]. 2023 [cit. 2023-07-12]. Dostupné z: https://docs.ai-thinker.com/_media/uwb/docs/dwm1000-datasheet-1.pdf

[31] datasheet.lcsc.com. 3-Axis Magnetic Sensor QMC5883L [online]. 2023 [cit. 2023-07-12]. Dostupné z: https://datasheet.lcsc.com/szlcsc/QST-QMC5883L-TR_C192585.pdf

[32] Drátek. Kompas HMC5883L [online]. 2023 [cit. 2023-07-12]. Dostupné z: https://dratek.cz/arduino/1194-kompas-hmc5883l.html

[33] Wikipedie. Serial Peripheral Interface [online]. 2023 [cit. 2023-07-12]. Dostupné z: https://en.wikipedia.org/wiki/Serial_Peripheral_Interface

[34] Wikipedie. I2C [online]. 2023 [cit. 2023-07-12]. Dostupné z: https://cs.wikipedia.org/wiki/I%C2%B2C

[35] Tile. How Tile Works [online]. 2023 [cit. 2023-07-12]. Dostupné z: https://www.tile.com/how-it-works

[36] Platform IO. [online]. 2023 [cit. 2023-07-12]. Dostupné z: https://platformio.org/platformio-ide

[37] Arduino. [online]. 2023 [cit. 2023-07-12]. Dostupné z: https://www.arduino.cc/en/software

[38] Makerfabs. ESP32 UWB Antenna Delay Calibrating [online]. 2023 [cit. 2023-07-12]. Dostupné z: https://www.makerfabs.cc/article/esp32-uwb-antenna-delay-calibrating.html

[39] Wikipedie. Triangulace [online]. 2023 [cit. 2023-07-12]. Dostupné z: https://cs.wikipedia.org/wiki/Triangulace

[40] Wikipedie. Trilaterace [online]. 2023 [cit. 2023-07-12]. Dostupné z: https://cs.wikipedia.org/wiki/Trilaterace

[41] Wikipedie. Angle of arrival [online]. 2023 [cit. 2023-07-12]. Dostupné z: https://en.wikipedia.org/wiki/Angle_of_arrival

[42] ScienceDirect. Angle of arrival [online]. 2023 [cit. 2023-07-12]. Dostupné z: https://www.sciencedirect.com/topics/engineering/angle-of-arrival

[43] Wikipedie. Received Signal Strength Indication [online]. 2023 [cit. 2023-07-12]. Dostupné z: https://cs.wikipedia.org/wiki/Received_Signal_Strength_Indication