ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE

Fakulta Elektrotechnická
Katedra řídicí techniky

Kalmanova filtrace výstupních dat polohy ze systému UWB a jejich fúze s daty GPS sensoru.

The Kalman filtering of the UWB system positioning data output and their fusion with the GPS sensor data.

Bakalářská práce

Josef Krška

Praha 2018

Studijní program: Kybernetika a robotika
Studijní obor: Systémy a řízení

Vedoucí práce: prof. Ing. František Vejražka, CSc.

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Krška** Jméno: **Josef** Osobní číslo: **457187**

Fakulta/ústav: **Fakulta elektrotechnická**

Zadávající katedra/ústav: **Katedra řídicí techniky**

Studijní program: **Kybernetika a robotika**

Studijní obor: **Systémy a řízení**

II. ÚDAJE K BAKALÁŘSKÉ PRÁCI

Název bakalářské práce:

Kalmanova filtrace výstupních dat polohy ze systému UWB a jejich fúze s daty GPS sensoru.

Název bakalářské práce anglicky:

The Kalman filtering of the UWB system positioning data output and their fusion with the GPS sensor data.

Pokyny pro vypracování:

1. Analyzujte charakter výstupních údajů o poloze tzv. roveru či tagu v UWB systému typu DecaWave.
2. Zvažte zařazení systému do demonstrátoru vyvíjeného na katedře radioelektroniky s cílem pokrýt výpadky signálů GNSS. Toho můžete dosáhnout Kalmánovou filtrací, anebo použijete algoritmus spočívající v následujícím postupu: Pokud čidla GNSS zjistí nedostatečný družicový signál, a je pravděpodobné, že i data inerciálního systému degradovala s časem, spustí demonstrátor příjem UWB dat svým tagem. Předpokládá, že okolní prostor je pokryt signály tzv. kotev.
3. Ověřte tuto metodu a určete její kvantitativní parametry.
4. Ve spolupráci s vedoucím práce realizujte, bude-li to možné, experiment v povrchovém lomu vápence Čertovy schody.

Seznam doporučené literatury:

[1] Betz, J. W.: Global Navigation Satellite Systems, Signals, and Receivers. John Wiley, Hoboken 2016
[2] Bensky, A.: Wireless Positioning, Technologies and Applications. Artech House, Boston 2008

Jméno a pracoviště vedoucí(ho) bakalářské práce:

prof. Ing. František Vejražka, CSc., katedra radioelektroniky FEL

Jméno a pracoviště druhé(ho) vedoucí(ho) nebo konzultanta(ky) bakalářské práce:

Datum zadání bakalářské práce: **15.02.2018** Termín odevzdání bakalářské práce: **25.05.2018**

Platnost zadání bakalářské práce: **30.09.2019**

prof. Ing. František Vejražka, CSc. podpis vedoucí(ho) práce	prof. Ing. Michael Šebek, DrSc. podpis vedoucí(ho) ústavu/katedry	prof. Ing. Pavel Ripka, CSc. podpis děkana(ky)

III. PŘEVZETÍ ZADÁNÍ

Student bere na vědomí, že je povinen vypracovat bakalářskou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací. Seznam použité literatury, jiných pramenů a jmen konzultantů je třeba uvést v bakalářské práci.

. Datum převzetí zadání

Podpis studenta

Abstrakt

Hlavním zájmem této práce je navržení lokalizačního systému, který využívá radiové technologie UWB a který lokalizuje pomocí metody *Time Difference of Arrival* (TDoA). Nicméně aby byla lokalizace pomocí TDoA vůbec možná, je nutné, aby měla infrastruktura systému (kotvy) jednotný čas. Dále je popsáno použití Kalmanova filtru jako filtru výstupní polohy systému a také je uveden postup fúze dat GPS a UWB systému.

Klíčová slova

Levenberg-Marquardt, lokalizace, Kalman Filter, TDoA, UWB

Abstract

The main focus of this thesis is the design of a localization system with TDoA-based localization ability, which uses UWB technology. However, in order to make the TDoA localization work it is neccessary to synchronize the clocks of the system infrastructure (Anchors). Further, this thesis describes the usage of Kalman filter as a position filter and how to use data fusion to combine the GPS and UWB data.

Keywords

Levenberg-Marquardt, localization, Kalman Filter, TDoA, UWB

Poděkování

Tímto bych chtěl poděkovat vedoucímu této práce prof. Ing. Františku Vejražkovi, CSc. za cenné rady a připomínky týkající se této práce. Zároveň bych rád poděkoval i Ing. Václavu Navrátilovi, za jeho pomoc s teoretickou podporu. Firmě RCD Radiokomunikace bych tímto poděkoval za poskytnutí nezbytného hardware a za pomoc s provedenými měřeními. Také bych chtěl poděkovat své rodině nejen za psychickou podporu, bez které by tato práce nemohla vzniknout.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

V Praze dne 14. května 2018 .....

Podpis studenta

Obsah

1 Úvod **3**

2 Systém UWB **4**
2.1 Ultra-Wide Band 4
2.2 Využitelnost pro lokalizaci 6

3 Metody lokalizace **7**
3.1 TWR 9
3.2 TDoA 10

4 Implementace TDoA lokalizace **12**
4.1 Popis systému 12
4.1.1 Kotva 12
4.1.2 Tag 12
4.1.3 Výpočetní část 13
4.2 Synchronizace 13
4.2.1 Kalmanův filtr 14
4.2.2 Kalmanův filtr pro synchronizaci 17
4.3 Výpočet polohy 22
4.3.1 Levenbergova-Marquardtova metoda řešení nelinárních nejmenších čtverců . 22

5 Filtrace polohy Kalmanovým filtrem **26**
5.1 Implementace 27
5.2 Ukázka funkce filtru 29

6 Fúze dat z UWB a GPS systému **31**
6.1 Fúze dat Kalmanovým filtrem 32
6.1.1 Souřadné systémy a převody mezi nimi 32
6.1.2 Rozšíření filtru polohy 35
6.2 Měření na Letenské pláni 36

7 Závěr **41**

Seznam zkratek **42**

Vysvětlení vybraných pojmů **44**

Seznam obrázků **45**

Literatura **47**

1 Úvod

Již několik desítek let je velká poptávka po systémech umožňující lokalizaci objektu a následnou navigaci v dané lokalitě. Největším zájemcem o tuto technologii alespoň v počátcích tvořila armáda. Konkrétně armáda Spojených států amerických v šedesátých letech dvacátého století vytvořila první globální navigační satelitní systém (GNSS) jménem Transit [1]. Ten přijímací stanici dokázal lokalizovat na principu Dopplerova jevu. Navigační systém Transit se poté stal přímým předchůdcem dnešního nejrozšířenějšího navigačního systému GPS.

Dnes společně s GPS existují i další navigační systémy, a to ruský GLONASS, čínský BeiDou nebo evropský Galileo, který v době psaní této práce ještě není v plném provozu. Tyto systémy nabízí výborné výsledky (tj. přesnost několika metrů a méně) pro lokalizaci v prostorech, kde má lokalizované zařízení přímou viditelnost na satelity daného systému. Pokud však o přímou viditelnost přijdeme nebo se pohybujeme například v hustě zastavěné oblasti, tak se výsledky lokalizace pomocí zmíněných globálních systémů zhoršují.

V takových situacích mohou pomoci podpůrné lokalizační systémy, které doplní možnost lokalizace tam, kde je přesnost lokalizace GNSS nízká nebo kde není GNSS lokalizace dostupná vůbec (například ve vnitřních prostorech).

Hlavním zájmem této práce je navržení právě takového podpůrného systému, který rozšíří oblast, kde je možno se lokalizovat, a dále také vytvoření „nadstavby“ systému v podobě filtru polohy a fúze GNSS polohy a polohy z podpůrného systému k dosažení přesnějších výsledků.

Práce je rozdělena do několika tématických celků, kde v každé kapitole detailně popíšeme jednu z klíčových částí systému. V následující kapitole 2 popíšeme bezdrátovou technologii UWB, na které je postaven podpůrný lokalizační systém. Poté v kapitole 3 shrneme a popíšeme metody získání dat pro lokalizaci a více popíšeme zejména dvě metody a to ToA a TDoA. Kapitola 4 se pak zabývá konkrétním řešením, tedy vyvinutým UWB systémem, jakým způsobem lokalizuje uživatele a jak je poloha vypočítávána, což mimo jiné zahrnuje i použití Kalmanova filtru a Levenbergovy-Marquardtovy metody. Další část práce (kapitola 5) se zabývá vytvořením filtru polohy, který zjemní průběh odhadnutých poloh. V poslední kapitole 6 upravíme filtr vytvořený v kapitole 5, aby dokázal fúzovat polohy z obou systémů a na závěr této kapitoly popíšeme měření, které otestuje všechny části navrženého systému, a okomentujeme jeho výsledky.

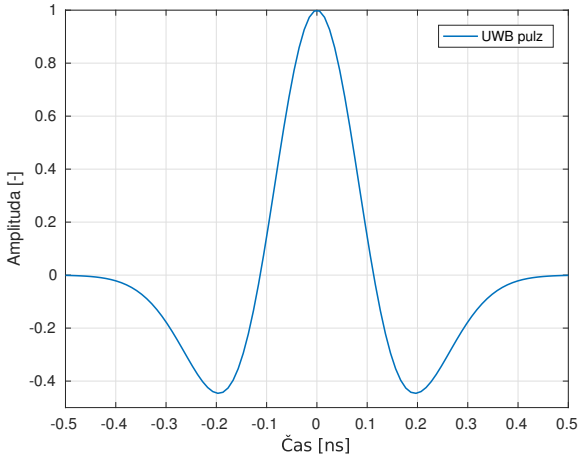
2 Systém UWB

Základním nástrojem této práce je systém UWB. Jeho prostřednictvím jsme schopni lokalizovat pohyblivá zařízení systému. Detailnější popis UWB systému je uveden v kapitole 4. V této kapitole krátce popíšeme a vysvětlíme pojem UWB a jaké výhody a nevýhody přináší jeho použití pro lokalizaci.

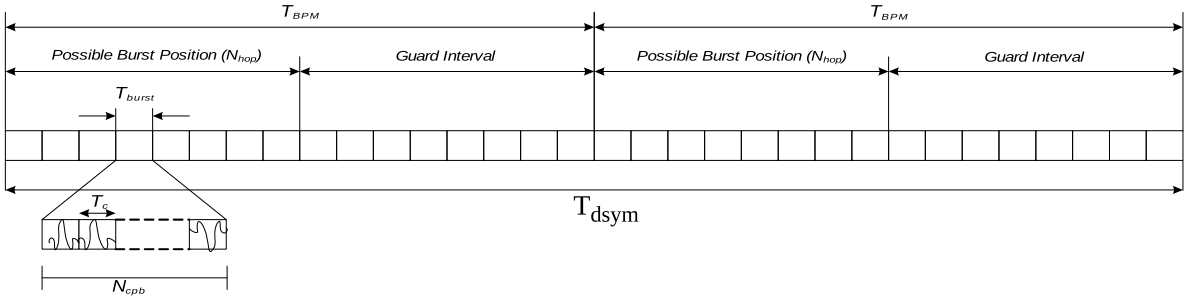
2.1 Ultra-Wide Band

Zkratkou UWB (Ultra-Wide Band, česky Ultra široké pásmo) se [2] označují rádiové technologie, které se vyznačují velkou šířkou pásma a nízkou spotřebou. Samozřejmě hlavním požadavkem, aby daná technologie mohla být takto označována, je splnění požadavků, které ukládá na její fyzickou (PHY) a linkovou (MAC) vrstvu (dle referenčního modelu ISO-OSI) standard IEEE-802.15.4.

Základním rysem UWB komunikace je její modulace. Na rozdíl od konvenčních modulací, jako je například ASK, QAM, či FSK, tato technologie nepoužívá žádnou nosnou vlnu k přenosu informace. Místo toho používá tzv. *Burst Position Modulation* (BPM) v kombinaci s *Binary Phase-Shift Keying* (BPSK) [3]. Základní informační jednotka v UWB komunikaci se nazývá *symbol*, která se přenese ve formě pulzů během stanoveného časového úseku T_{dsym} označovaného jako rámec neboli *frame*. Rámec je rozdělen na čtyři stejné intervaly, přičemž v každém intervalu se může nacházet nejvýše jeden impulz (či *burst* neboli dávka impulzů). Každý jeden symbol přenáší dva bity informace. První bit je definován pozicí pulzu v rámci (BPM) a druhý je definován polaritou pulzu (BPSK). Příklad UWB pulzu můžeme vidět na obrázku 2.1 a rozdělení rámce na intervaly je na obrázku 2.2. Díky své modulaci je UWB komunikaci také přezdíváno jako „impulzní“.



Obrázek 2.1: UWB impulz



Obrázek 2.2: Rozdělení časového rámce BPM-BPSK modulace [3]

Nyní pár slov ke struktuře přenášených dat. Na obrázku níže je zakreslen formát PHY datové jednotky (PPDU) definovaný standardem [3].

Obrázek 2.3: Struktura fyzického rámce

Aby rámec mohl být správně detekován, vysílá zařízení vysílající data před samotným rámcem tzv. *preambuli*, což je předem známá sekvence pulzů 1, −1 a 0 (normální pulz, invertovaný pulz, žádný pulz), které zde však nenesou žádnou další informaci. Pulzy jsou vysílány frekvencí označovanou jménem *Pulse Repetition Frequency* (PRF). Za ní je oddělovač začátku rámce SFD, který společně s preambulí tvoří synchronizační hlavičku SHR. Po něm následuje datová jednotka PPDU.

Standard [3] rozděluje UWB na dvě hlavní kategorie podle jejich PRF a to na *low rate* PRF UWB (LRP UWB) a *high rate* PRF UWB (HRP UWB). LRP UWB má průměrnou PRF nižší než 3 MHz a HRP UWB ji má v průměru vyšší než 3 MHz. Dále se však budeme zabývat pouze HRP UWB, protože do této kategorie spadá i použitý UWB systém.

Standard IEEE-802.15.4 rozděluje HRP UWB do tří frekvenčních pásem a to do

- subgigahertzového pásma od 249,6 MHz do 749,6 MHz s jedním kanálem (který tak zabírá celé pásmo),
- nižšího pásma od 3,1 GHz do 4,8 GHz se čtyřmi kanály
- a vyššího pásma od 6 GHz do 10,6 GHz s jedenácti kanály.

Pásma HRP UWB jsou tedy rozdělena celkem do 16 kanálů, které se označující čísly 0 až 15.

Jak bylo uvedeno dříve, technologie UWB se vyznačuje velkou šířkou pásma (odtud jméno Ultra-Wide Band). I šířka pásma je definovaná standardem a to tak, že pro UWB musí být buď absolutní šířka pásma BW větší než 500 MHz nebo relativní šířka pásma BW_{rel} větší než 0,2 [2]. Relativní šířku pásma vypočteme vzorcem

$$BW_{\text{rel}} = \frac{f_c}{BW}\,, \tag{2.1}$$

kde f_c je frekvence uprostřed daného kanálu (označována jako centrální). Zde šířku pásma BW chápeme jako rozsah frekvencí, pro které je útlum menší než 3 dB.

Díky tomu, že pásma HRP UWB zasahují i do bezlicenčních pásem, bylo nutné použití této technologie regulovat. Regulace se v tomto případě vztahují na vysílaný výkon [4], avšak každý stát určuje svou vlastní horní mez pro vysílaný výkon. Například pro pásmo 3 GHz až 10 GHz je nejčastěji průměrný (za 1 ms) maximální vyzářený výkon omezen na -41,3 dBm/MHz [4].

Pro větší přehlednost textu budeme v následujícím textu místo HRP UWB psát pouze UWB.

2.2 Využitelnost pro lokalizaci

Technologie UWB nám, díky své velké šířce pásma, umožňuje komunikovat s velkými přenosovými rychlostmi. Konkrétně to jsou rychlosti 110 kbit/s, 850 kbit/s, 6,81 Mbit/s a 27,24 Mbit/s. Toto se velice hodí, pokud chceme lokalizovat více zařízení současně. Jak bude popsáno v kapitole 4, lokalizovaná zařízení vysílají krátké zprávy a vysoká přenosová rychlost zde šetří kapacitu kanálu a zároveň tím poskytuje i prostor pro více lokalizovaných zařízení.

Dále se můžeme přímo ve standardu [3] dočíst, že zařízení splňující standard má vyšší odolnost proti vícecestnému šíření díky své BPM-BPSK modulaci. Jak bylo řečeno výše, rámec, ve kterém se přenáší pulz, je rozdělen na čtyři intervaly. Standard říká [3], že by se pulz resp. *burst* měl vyskytovat v prvním nebo ve třetím intervalu rámce, aby se tak potlačily pulzy resp. *bursty*, které vlivem vícecestného šíření či jiného rušení přijdou v druhém nebo čtvrtém intervalu (obrázek 2.2). Intervaly jedna a tři tak nesou označení *Possible burst intervals* a intervalům dva a čtyři se říká Ochranné intervaly (*Guard intervals*). Proto symbol přenáší pouze dva bity informace místo maximálních tří bitů (dva bity dle pozice pulzu a jeden dle jeho fáze). Tímto můžeme dosahovat vyšších přesností při lokalizaci, protože signály, které se k přijímači dostaly přes odrazy se budou (ne však vždy) ignorovat a tak se jim zabrání zhoršovat výsledky lokalizace. Zároveň nám toto umožňuje rozšířit oblast lokalizace z venkovních prostor i do interiéru, kde je typicky více odrazů.

V neposlední řadě je UWB vhodné pro lokalizaci také pro svou energetickou náročnost, která je díky vlastnostem UWB komunikace nízká (vysílání impulzů je „levné“). Což přináší velkou výhodu pro lokalizovaná uživatelská zařízení, která tak nemusí mít velikou a těžkou baterii, aby vydržela delší dobu bez nabíjení. Tím zároveň klesá i výrobní cena takového zařízení.

Zároveň s nízkou náročností a omezením vysílacího výkonu přichází i nevýhoda v podobě krátkého dosahu. Dosah se však dá zvýšit například vhodným upravením vysílaného výkonu v závislosti na délce zprávy tak, aby stále byla dodržena regulace, a nebo použitím kanálu s větší šířkou pásma [3].

Čip, který používáme v našem UWB systému je vyráběn společností DecaWave Ltd. a nese jméno DW1000. Ten nám dává schopnost komunikovat v kanálech $2, 3, 4, 5$ a 7 (část nižšího pásma a jeden kanál z vyššího) s rychlostmi 110 kbit/s, 850 kbit/s nebo 6,81 Mbit/s [5]. Čip umožňuje lokalizaci svou schopností opatřovat přijímané či vysílané zprávy časovou značkou s vysokým rozlišením 15,65 ps. Tímto nám umožňuje teoreticky dosáhnout přesnosti měření vzdálenosti/polohy až 4,7 mm.

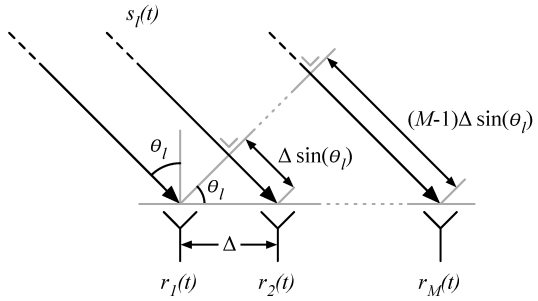
Konkrétní metody lokalizace budou uvedeny v kapitole 3.

3 Metody lokalizace

Existuje [2] mnoho způsobů lokalizace. Mezi nejznámější patří AoA (Angle of Arrival), RSS (Received Signal Strength), ToA (Time of Arrival) a TDoA (Time Difference of Arrival).

V lokalizačních systémech můžeme zařízení dělit dle jejich funkcí do dvou hlavních skupin a to na zařízení se známou polohou, neboli referenční, a na zařízení s neznámou polohou, neboli lokalizovaná či uživatelská. Referenční zařízení navíc bývají stacionární a (nejen) v použitém UWB systému se takovému zařízení říká kotva (*Anchor*). Podobně i uživatelské zařízení má své označení - *tag*.

Metoda zvaná „úhel příjmu signálu“ (AoA) vypočítává polohu z úhlů přijatého signálu získaných z několika zařízení. Ke zjištění úhlu se používá směrové antény, která se natočí do směru s nejsilnějším signálem, nebo seskupení několika stacionárních antén, kde se úhel určí podle rozdílů časů přijetí na jednotlivých anténách. Poté se každou kotvou, která přijala daný signál, vede přímka, která má vzhledem k $i$té kotvě takový úhel, pod jakým byl v $i$té kotvě přijat signál. Průsečík přímek pak značí polohu tagu. Tomuto způsobu výpočtu pozice bodu na základě úhlů se také říká triangulace. Pro určení 2D polohy tedy stačí pouze dvě měření úhlu, pro 3D jsou pak potřeba alespoň tři. Hlavní nevýhodou této metody je její citlivost na vícecestné šíření, jehož riziko je zejména při indoor lokalizaci veliké a proto je tato metoda nevhodná pro použití ve vnitřních prostorách.



Obrázek 3.1: Angle of Arrival se skupinou antén[1]

Další diskutovanou metodou je metoda „Síla přijatého signálu“ (RSS). Ta počítá polohu ze síly přijatého signálu, ze kterého lze přímo získat vzdálenost dvou zařízení. Při znalosti vysílaného výkonu a útlumu prostředí se vzdálenost d dá vypočítat ze vztahu [2]

$$P(d) = P_0 - 10n \log_{10}\left(\frac{d}{d_0}\right), \tag{3.1}$$

kde P je přijatý výkon, P_0 je výkon přijatý v referenční vzdálenosti d_0 a n je koeficient útlumu prostředí. Úloha určení pozice pak spočívá v prokládání kružnic, kde $i$tá kružnice má střed v $i$té kotvě a poloměr d_i, který odpovídá vypočtené vzdálenosti podle síly přijatého signálu

[1] http://www.mdpi.com/1424-8220/17/11/2522/htm

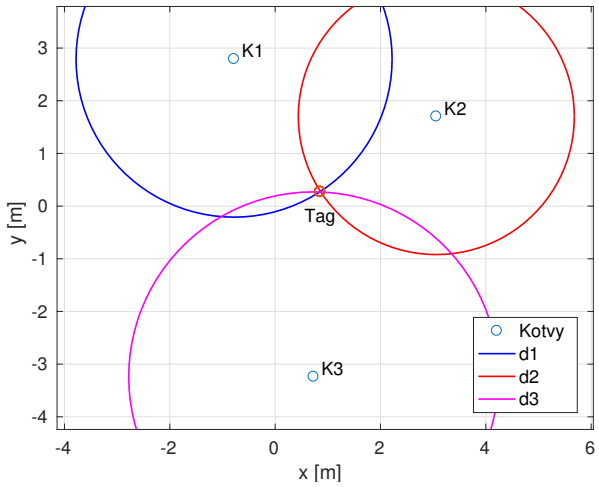
v i-té kotvě. Z tohoto lze vidět, že pro jednoznačné určení pozice ve 2D je potřeba alespoň tří měření a pro určení polohy ve 3D jsou požadována alespoň čtyři měření.

Metoda RSS spoléhá na správnou detekci signálu a správné změření přijatého výkonu a proto je velice citlivá na vícecestné šíření. Polohování pomocí této metody je také nepřesné zejména díky měření výkonu, které má tendenci oscilovat. Oscilace se při přepočtu na vzdálenost ještě zvětší vlivem nelineárního modelu šíření signálu popsaného rovnicí 3.1.

V ToA (Čas přijetí) lokalizaci se vzdálenost mezi tagem a kotvou určuje pomocí odhadnutí doby šíření signálu τ mezi tagem a kotvou. To můžeme zapsat rovnicí

$$\tau = t^A_{\mathrm{Rx}} - t^T_{\mathrm{Tx}} - \Delta\,, \tag{3.2}$$

kde t^A_{Rx} je čas přijetí signálu v kotvě v její časové doméně, t^T_{Tx} je čas vyslání signálu z tagu v jeho časové doméně a Δ je časový offset hodin obou zařízení, který je často neznámý. Notace je zde taková, že horní index symbolizuje časovou doménu, v jaké byl čas změřen. K odhadnutí τ je potřeba, aby všechna zařízení (tedy jak ta se známou polohou, tak i ta s neznámou polohou) měla buď synchronizované hodiny (pak $\Delta \approx 0$), což se v reálných aplikací zejména kvůli energetické náročnosti nevyplácí, nebo aby se offset hodin Δ zařízení eliminoval sadou měření. Konkrétní metoda eliminace offsetu je například TWR, která bude diskutována v následující části. Z odhadnutých dob šíření τ se vynásobením rychlostí šíření signálu c získá vzdálenost mezi zařízeními. Postup výpočtu polohy je pak podobný jako v RSS lokalizaci. Hledá tedy se průsečík několika kružnic.



Obrázek 3.2: Trilaterace

Metoda TDoA (Časový rozdíl přijetí) se oproti TWR nesnaží odhadnout přímo dobu šíření τ. Místo toho pracuje s rozdíly časů, v jakých byl signál z tagu přijat kotvami. Je proto potřeba snížit časový offset hodin kotev k 0. K tomu se tak nevyžaduje časové synchronizace celého systému, ale „pouze“ synchronizace hodin kotev. Poté je čas přijetí v $i$té kotvě závislý na vzdálenosti tagu k $i$té kotvě. Z naměřených časů přijetí se pak vypočtou rozdíly časů přijetí, pomocí nichž se určí neznámá poloha tagu. Na základě rozdílu časů přijetí z $i$té a $j$té kotvy tvoří množina bodů, kde se může nacházet tag, hyperbolu. Místo, kde se všechny hyperboly protínají, je pak místem, kde se nachází tag. Proto se této metodě také říká hyperbolická navigace. Zde je pro určení 2D polohy nutno získat alespoň dvě TDoA měření, resp. tři časy přijetí, a pro 3D polohu pak alespoň tři TDoA měření, resp. čtyři časy přijetí.

Z výše uvedených metod bylo pro náš UWB systém s čipem DW1000 nejvhodnější implementovat ToA [6] a TDoA [7], neboť nám čip poskytuje dobrou přesnost značkování zpráv a náročnost nutných výpočtů v zařízeních je nízká. Tyto dvě metody popíšeme podrobněji v následujícím textu.

	TWR	TDoA
Zvláštní nároky na implementaci	Žádné	Synchronizace kotev
Spotřeba energie tagů	Vysoká	Nízká
Počet uživatelů	<10/s	10-100+/s
Využití kapacity kanálu	Vysoké	Nízké

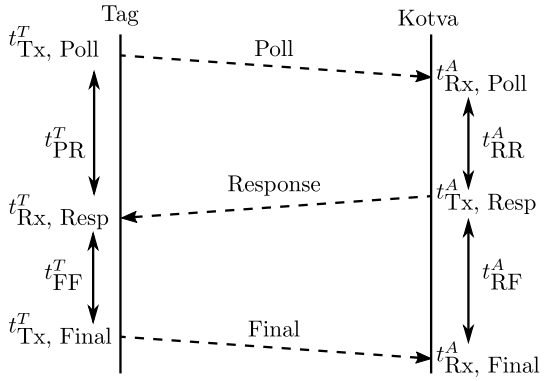
Tabulka 3.1: Porovnání TWR a TDoA

3.1 TWR

Metoda TWR, neboli Two-Way Ranging, je algoritmus [5], podle kterého si dvě zařízení s asynchronními hodinami vymění sérii zpráv, a pomocí časových značek přijetí/odeslání zpráv pak eliminují časový offset hodin. Tím se získá hodnota τ, která odpovídá době šíření signálu mezi zařízeními.

Existuje více druhů TWR, které se dělí na jednostranné (single-sided) a dvoustranné (double-sided). Blíže však popíšeme pouze double-sided TWR, ve kterém si zařízení vymění celkem tři zprávy. Tato metoda nabízí výhodný kompromis mezi rychlostí a přesností měřené vzdálenosti [5].

Na obrázku 3.3 vidíme grafické znázornění TWR měření. Postup TWR je následující.



Obrázek 3.3: Double Sided TWR

1. Měření zahajuje tag vysláním *Poll* zprávy a zaznamená si čas jejího vyslání $t^T_{\text{Tx, Poll}}$.
2. Kotva přijme *Poll* zprávu, uloží si čas přijetí $t^A_{\text{Rx, Poll}}$, odešle zprávu *Resp* a uloží čas odeslání zprávy $t^A_{\text{Tx, Resp}}$.
3. Tag přijme *Resp* zprávu v čase $t^T_{\text{Rx, Resp}}$ a do poslední *Final* zprávy kotvě zahrne časy $t^T_{\text{Tx, Poll}}$, $t^T_{\text{Rx, Resp}}$ a vypočtený $t^T_{\text{Tx, Final}}$.
4. Kotva si při přijetí poslední *Final* zprávy uloží čas $t^A_{\text{Rx, Final}}$ a přejde k odhadu τ.

Nyní popíšeme výpočet doby šíření τ. Hodiny tagu a kotvy nejsou během měření synchronizované a existuje mezi nimi nenulový offset

$$t^A = t^T - \Delta\,. \tag{3.3}$$

Pro odhad doby letu τ je nutné předpokládat, že po dobu měření vzdálenosti pomocí TWR je Δ konstantní. Odhad τ se získá na základě dob $t^T_{\text{PR}}, t^A_{\text{RF}}$, tedy dob než zařízení dostane

odpověď na svou zprávu (dle [5] označováno jako *Round Trip Delay*), a dob $t^A_{\mathrm{RR}}, t^T_{\mathrm{FF}}$ než zařízení vyšle odpověď na přijatou zprávu (dle [5] *Reply Delay*). Uvedeme příklad výpočtu Round Trip Delay pro tag mezi odesláním zprávy *Poll* a přijetím odpovědi *Resp*.

$$t^T_{\mathrm{PR}} = t^T_{\mathrm{Rx,\ Resp}} - t^T_{\mathrm{Tx,\ Poll}} \Rightarrow t^A_{\mathrm{PR}} = \left(t^A_{\mathrm{Rx,\ Resp}} - \Delta\right) - \left(t^A_{\mathrm{Tx,\ Poll}} - \Delta\right) = t^A_{\mathrm{Rx,\ Resp}} - t^A_{\mathrm{Tx,\ Poll}}\,. \tag{3.4}$$

Z rovnice 3.4 je vidět, že při konstantním Δ ho lze jednoduše eliminovat a získat tak časy ve stejné časové doméně. Analogicky se získají ostatní časy t^A_{RR}, t^T_{FF} a t^A_{RF}.

V získání odhadu τ se TWR dá dále dělit na symetrické (doby odeslání odpovědí t^A_{RR} a t^T_{FF} jsou stejné) a nesymetrické (doby odeslání odpovědí nejsou stejné). Způsoby se liší výpočtem a každý má své výhody a nevýhody [5].

Výhodou symetrického Double-sided TWR je, že výpočet τ obsahuje jen elementární operace sčítání a dělení čtyřmi (což v procesoru není náročná operace). Ovšem nevýhodou je, že se spoléhá na rovnost t^A_{RR} a t^T_{FF} a pokud nejsou stejné tak chyba v měřené vzdálenosti je pak úměrná rozdílu $t^A_{\mathrm{RR}} - t^T_{\mathrm{FF}}$. Výpočet τ se provede následovně [5]

$$\tau = \frac{\left(t^T_{\mathrm{PR}} - t^A_{\mathrm{RR}}\right) + \left(t^A_{\mathrm{RF}} - t^T_{\mathrm{FF}}\right)}{4}\,. \tag{3.5}$$

Velikou výhodou asymetrického Double-sided TWR je kompenzace jak offsetu hodin, tak i jejich driftu. Nevýhodou je pak náročnost výpočtu zahrnující jak dělení, tak i násobení, což může v mikrokontrolérech „trvat“ a zvyšovat spotřebu.

$$\tau = \frac{t^T_{\mathrm{PR}} t^A_{\mathrm{RF}} - t^A_{\mathrm{RR}} t^T_{\mathrm{FF}}}{t^T_{\mathrm{PR}} + t^A_{\mathrm{RF}} + t^A_{\mathrm{RR}} + t^T_{\mathrm{FF}}}\,. \tag{3.6}$$

Podrobné odvození toho vztahu můžeme najít v [8].

Jak bylo uvedeno výše, k získání jedné vzdálenosti je potřeba tří zpráv. Zároveň pro výpočet polohy ve 2D jsou potřeba alespoň tři vzdálenosti, tedy tag postupně musí provést TWR měření s každou kotvou a aby nedocházelo ke kolizím, nesmí během TWR měření probíhat na stejném kanále komunikace dalších zařízení. Z toho vyplývá vysoká časová náročnost TWR lokalizace a tedy i nízká frekvence určení polohy spolu s nízkým počtem současně běžících tagů.

Detailní popis aplikace této metody můžeme nalézt v [6].

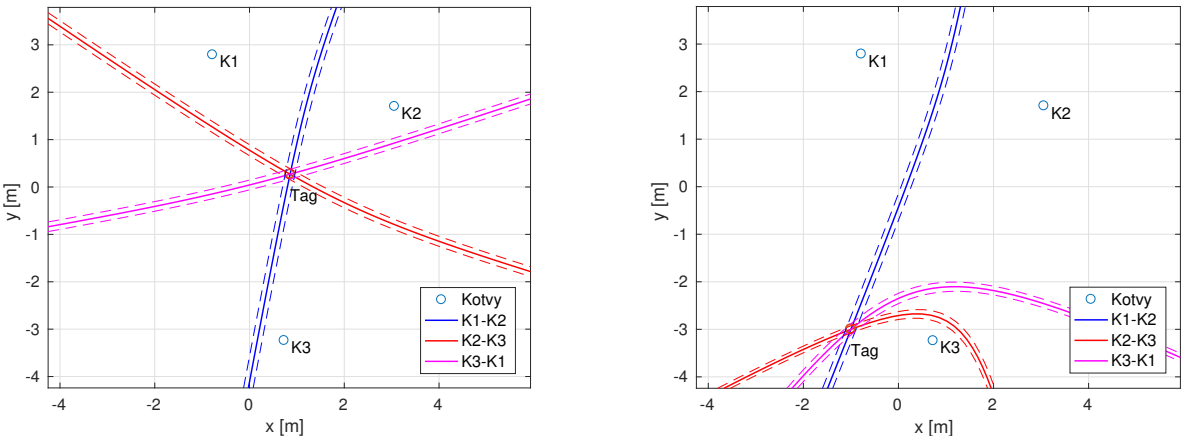
3.2 TDoA

Metoda Time Difference of Arrival předpokládá, že všechny kotvy v systému pracují ve stejné časové doméně M, tedy že mají synchronizované hodiny. Tag zde vysílá pouze tzv. *blink* zprávu, kterou přijímají všechny kotvy v dosahu signálu. Čas přijetí *blink* zprávy je v každé kotvě jiný, ale protože kotvy sdílejí jednu časovou doménu, tak je čas přijetí $t^M_{\mathrm{Rx,i}}$ $i$té kotvy zcela závislý jen na vzdálenosti tagu od $i$té kotvy.

Odečtením času přijetí v $j$té $t^M_{\mathrm{Rx,j}}$ od času z $i$té kotvy $t^M_{\mathrm{Rx,i}}$ (kombinace i a j volíme tak, aby vypočtené rozdíly byly lineárně nezávislé) získáme rozdíl času přijetí (odtud jméno metody)

$$\Delta\tau_{i,j} = t^M_{\mathrm{Rx,i}} - t^M_{\mathrm{Rx,j}}\,. \tag{3.7}$$

Čas $\Delta\tau_{i,j}$, resp. TDoA vzdálenost $d_{i,j} = c\Delta\tau_{i,j}$, kde c je rychlost šíření signálu, značí o kolik byl tag, v čase vyslání zprávy, dále či blíže k $j$té kotvě než ke kotvě $i$té. Protože je tento čas pro jedno měření konstantní, tvoří množina možných poloh tagu hyperbolu mezi $i$tou a $j$tou kotvou s ohnisky v daných kotvách.



(a) Lokalizace v místě s nízkým DOP

(b) Lokalizace v místě s vysokým DOP

Obrázek 3.4: Time Difference of Arrival, oblast okolo hyperbol značí nejistotu měření

Na rozdíl od metod jako je ToA a RSS se v TDoA neodhaduje doba šíření signálu, ze kterého se dále získá vzdálenost. Místo toho se k odhadu polohy použijí přímo naměřené TDoA vzdálenosti. Odhadnutá poloha tagu je pak v bodě, kde se protínají všechny hyperboly. Měření času však má své nejistoty a tak ve skutečných aplikacích je „průsečík“ spíše oblast než jediný bod (viz. obrázek 3.4a). Může se však stát, že se hyperboly protínají tzv. neostře, jak můžeme vidět na obrázku 3.4b. Pak se oblast, kde se může nacházet tag, výrazně zvětšuje. Toto bývá způsobeno například lokalizací tagu v oblasti s vysokým DOP (*Dillution Of Precision*, Zhoršení přesnosti). To samé platí i pro určení polohy hledáním průsečíku kružnic.

Výhodou, která vyplývá přímo z popisu metody, je rychlost získávání měření. Není zde potřeba, jako u TWR, aby každý tag oslovoval zprávou (zprávami) každou kotvu zvlášť (nemluvě o tom, že po dobu komunikace tagu a kotvy nemůže probíhat žádná další komunikace), ale stačí, aby tag poslal jedinou všesměrovou zprávu a tím získáme dostatek dat k jeho lokalizaci. To zásadně snižuje proudovou spotřebu tagu, protože vysílání je méně energeticky náročné než přijímání. Na druhou stranu je TDoA implementačně obtížné díky nutnosti synchronizovaných hodin ve všech kotvách.

Nicméně se po uvážení výhod, které TDoA přináší (nízká spotřeba tagu, vysoké množství tagů, efektivnější využití přenosového média) a které dokáží převážit nevýhody této metody (nutnost synchronizace), výsledků a omezení TWR metody (zdlouhavé získávání dat pro výpočet, nízký počet současně použitých tagů, vyšší spotřeba tagu) [6] jsme se při tvorbě použitého UWB systému rozhodli pokračovat v lokalizaci s využitím právě TDoA metody.

4 Implementace TDoA lokalizace

V předchozí kapitole jsme zmínili nejznámější metody pro získání dat potřebných k lokalizaci uživatelského zařízení a uvedli jsme důvody proč navrhovaný systém UWB lokalizuje s využitím TDoA.

Základem TDoA lokalizace je schopnost přesně synchronizovat stacionární lokalizační zařízení (kotvy), protože se od toho odvíjí schopnost přesně lokalizovat. Pro použitelnost ve vnitřních prostorách, kde je požadovaná přesnost lokalizace v řádech nejvýše desítek centimetrů, je přesnou synchronizací míněna maximálně nanosekundová odchylka hodin kotev. S postupem, který dále popíšeme, dosahujeme dokonce subnanosekundové přesnosti.

4.1 Popis systému

Než začneme se samotným popisem synchronizace, je velice vhodné blíže popsat systém, se kterým pracujeme.

Celý systém tvoří tři typy zařízení, kterými jsou kotvy, tagy a výpočetní část. Jejich jedinečné role popíšeme níže. Použitá zařízení byla vytvořena firmou RCD ve spolupráci s FEL ČVUT.

4.1.1 Kotva

Kotvy tvoří měřící infrastrukturu systému. V prostoru, kde se lokalizuje, jsou pevně umístěny a jejich polohy jsou známy. K UWB komunikaci je použit čip DW1000.

Role kotvy se dá dále rozdělit na roli master nebo slave. Kotva, která je slave, tvoří hlavní část systému pro získávání dat pro lokalizaci. Jejím hlavním úkolem je přijímání dvou typů zpráv, kde jedním typem jsou *blink* zprávy od tagů, které po přijetí odešle s potřebnými informacemi o zprávě výpočetnímu centru. Druhým typem jsou synchronizační zprávy od master kotvy, pomocí kterých se kotvy synchronizují.

Master kotva rozšiřuje funkčnost slave kotvy o synchronizaci skupiny. Periodickým všesměrovým vysíláním času master kotvy se slave kotvy sesynchronizují, aby byly schopny měřit čas ve stejné časové doméně (master doména M).

Vzhledem k tomu, že přijímání radiového signálu je mnohem náročnější na spotřebu než jeho vysílání a že kotvy posílají centru data o každé *blink* zprávě, je vhodné využít jejich stacionárního umístění. Například lze uvažovat o připojení kotev do elektrické rozvodné sítě a zároveň o přivedení strukturované kabeláže pro komunikaci.

4.1.2 Tag

Tagy jsou mobilní uživatelská zařízení, jejichž neznámou polohu se systém snaží s pomocí kotev určit. Na zařízení je kladen požadavek nízké spotřeby a malé váhy. Proto je jedinou úlohu tagu v TDoA systému periodicky (nemusí přesně periodicky) všesměrově vysílat krátké (typicky obsahují pouze identifikaci tagu) *blink* zprávy. K UWB komunikaci je podobně jako u kotev použit čip DW1000.

4.1.3 Výpočetní část

Poslední neméně důležitou částí systémů je výpočetní a řídící část.

Zde se shromažďují všechny zprávy, které kotvy posílají centru po přijetí *blink* zprávy. Tyto zprávy obsahují adresu kotvy, informaci o tagu, který *blink* zprávu vyslal a čas přijetí *blink* zprávy. Zprávy se zde seřadí a roztřídí podle tagu a vypočítá se z nich poloha daného tagu v daný čas.

Vypočtené polohy však mohou být nepřesné a mohou se mezi nimi vyskytovat i naprosto chybné polohy (to se děje, když se například snažíme lokalizovat tag, který se nachází v oblasti, kde se hyperboly protínají neostře). O způsobech filtrace a konkrétním filtru polohy, který zpřesní výsledky a zahodí zjevně špatné výsledky, se bude zabývat kapitola 5.

4.2 Synchronizace

Jak již bylo zmíněno výše, funkční požadavek, který způsobuje implementaci TDoA lokalizace obtížnou, je nutnost přesně synchronizovat měřící infrastrukturu systému.

K časovému značkování zpráv jsou využity interní hodiny čipu DW1000, které mají kmitočet 63,8976 GHz [5]. Frekvence hodin tak určuje rozlišení měření času, které je přibližně 15,65 ps. Hodiny používá jako čítací frekvenci interní 40bitový čítač modulů z čehož plyne, že čítač přeteče přibližně každých 17,2 s.

Ovšem žádný krystal (oscilátor, hodiny,...) není dokonalý a tak nemáme zaručeno, že bude oscilovat přesně na definované frekvenci. Společně s frekvencí je tak hlavním parametrem každého krystalu i přesnost. Z toho vidíme, že hodiny trpí nejen offsetem oproti hodinám v jiné kotvě (zařízení) Δ ale i chybou frekvence Δf. Offset hodin dvou kotev tak není v čase konstantní a z experimentálních měření jsme zjistili, že i tato rychlost změny offsetu se v čase mění. To je i díky proměnné frekvenci, která se mění především vlivem teploty čipu. Hodiny tedy trpí navíc i kolísáním frekvence $\Delta \dot{f}$. Pokud bychom zapsali časový průběh hodin kotvy A ve formě funkčního předpisu, dostali bychom

$$t^A(t) = \Delta + t\Delta f + \frac{1}{2}t^2\Delta \dot{f}\,. \tag{4.1}$$

Registr čítače v čipech zpřístupnil výrobce pouze ke čtení a díky tomu nelze synchronizovat přímo hodiny čipů. Synchronizace zde tak bude probíhat ve formě odhadu chyby hodin slave a master kotvy Δ^{MS}, který vystupuje jako jediná neznámá v následující rovnici

$$t^S_{Rx} - t^M_{Tx} = \frac{\parallel \boldsymbol{r}^S - \boldsymbol{r}^M \parallel}{c} + \Delta^{MS}\,, \tag{4.2}$$

kde t^M_{Tx} je čas vyslání synchronizační zprávy v časové doméně mastera, t^S_{Rx} je čas přijetí synchronizační zprávy v časové doméně slave kotvy a $\parallel \boldsymbol{r}^S - \boldsymbol{r}^M \parallel$ je vzdálenost mezi master a slave kotvou, ze které vydělením rychlostí šíření signálu c získáme dobu šíření signálu mezi master a slave kotvou.

Postupů synchronizace je několik. Všechny však spoléhají na přijímání periodických (v některých případech nemusí být nutně periodické) zpráv synchronizačního prvku (master kotvy). Při uvážení periodicity synchronizačních zpráv se nabízí synchronizovat a odhadnout chybu pomocí lineární aproximace mezi synchronizačními zprávami. Je také možnost k odhadu chyby použít PI regulátor nebo Kalmanův filtr. Zmíněné postupy jsou popsány a jejich možnosti diskutovány v [9], zároveň tento článek ukazuje, že nejlepších výsledků dosáhneme, pokud budeme odhadovat chybu hodin pomocí Kalmanova filtru.

4.2.1 Kalmanův filtr

Kalmanův filtr je algoritmus, který z naměřené směsi dat a šumu odhaduje skutečnou hodnotu měřené veličiny. Kromě skutečné hodnoty filtr odhaduje i její kovarianci, která je reprezentována maticí $\mathbf{P}$. Algoritmus je rekurzivní a tedy každý nový odhad bere „v úvahu“ předchozí odhady a je jimi ovlivňován.

Klasický Kalmanův filtr pro správnost odhadů předpokládá, že vývoj měřené veličiny je lineární a že měření jsou zatížena bílým gaussovským šumem. Pro systémy, které tyto předpoklady splňují, je Kalmanův filtr optimálním estimátorem [10].

Základem filtru je popsání procesu veličiny matematickým modelem, kterým se filtr snaží odhadnout její vývoj v čase. Na Kalmanův filtr se proto dá také pohlížet jako na pozorovatele stochastického systému. V lineárním případě a diskrétní formě se vývoj stavového vektoru filtru v diskrétním čase k dá zapsat pomocí následující lineární rovnice [10]

$$\boldsymbol{x}\left[k\right] = \mathbf{F}\boldsymbol{x}\left[k-1\right] + \mathbf{B}\boldsymbol{u}\left[k\right] + \boldsymbol{w}\left[k\right] , \tag{4.3}$$

kde vektor $\boldsymbol{x}$ je vektor n odhadovaných veličin neboli stavový vektor, $\mathbf{F}$ je matice přechodu mezi stavy (velikosti $n \times n$), $\boldsymbol{u} \in \mathbb{R}^m$ je vektor řídících vstupů ovlivňující odhadované veličiny, $\mathbf{B}$ je vstupní matice (velikosti $n \times m$) a $\boldsymbol{w} \sim N\left(0, \mathbf{Q}\right)$ je vícerozměrný gaussovský šum s nulovou střední hodnotou a rozptylem popsaným maticí $\mathbf{Q}$. Náhodný šum $\boldsymbol{w}$ se také nazývá procesním šumem a matice $\mathbf{Q}$ kovarianční maticí procesního šumu. Tyto veličiny zahrnují nepřesnosti, které jsou způsobeny nepřesností modelu. V mnohých případech nezasahuje do procesu žádný vstup a tak vstupní matice $\mathbf{B}$ bývá nulová. Tak je tomu i v našich aplikacích a proto nebudeme dále řídící vstupy $\boldsymbol{u}$ ani matici $\mathbf{B}$ uvažovat.

Rovnice 4.3 v ideálním případě odhaduje hodnotu měřené veličiny v následujícím časovém kroku k. To se dá zapsat jako

$$\boldsymbol{z}\left[k\right] = \mathbf{H}\boldsymbol{x}\left[k\right] + \boldsymbol{v}\left[k\right] , \tag{4.4}$$

kde $\boldsymbol{z}$ je vektor měření, $\mathbf{H}$ je matice pozorování, která převede stavy filtru na měřené veličiny a $\boldsymbol{v} \sim N\left(0, \mathbf{R}\right)$ je vícerozměrný náhodný šum s nulovou střední hodnotou a rozptylem $\mathbf{R}$.

Je však na místě podotknout, že matice $\mathbf{F}, \mathbf{B}, \mathbf{Q}, \mathbf{H}$ a $\mathbf{R}$ nemusí být časově nezávislé a mohou být v každém čase jiné. Pro jednoduchost zápisu budeme v této sekci uvažovat zmíněné matice jako časově neproměnné.

V běžné praxi není vždy zaručena linearita systému a ani normální rozložení šumu. Proto se vyvinula rozšíření a různé obměny Kalmanova filtru. Pro nelineární Kalmanův filtr mají rovnice 4.3 a 4.4 následující tvar

$$\boldsymbol{x}\left[k\right] = f\left(\boldsymbol{x}\left[k-1\right], \boldsymbol{u}\left[k\right]\right) + \boldsymbol{w} , \tag{4.5}$$

$$\boldsymbol{z}\left[k\right] = h\left(\boldsymbol{x}\left[k\right]\right) + \boldsymbol{v} , \tag{4.6}$$

kde funkce f a h jsou obecné nelineární funkce.

Pro použití v nelineárních systémech lze použít Rozšířeného Kalmanova filtru (EKF - Extended Kalman Filter), který používá linearizovaného modelu a linearizuje průběh v okolí předchozího odhadu $\boldsymbol{x}\left[k-1\right]$.

Když je matematický model silně nelineární, přestane i EKF pracovat správně. V takových případech je vhodné použít Unscented Kalman Filter (UKF), který pro odhad stavů používá tzv. Unscented transformaci [11]. Při řešení problému bezdrátové synchronizace si naštěstí vystačíme s diskrétním lineárním Kalmanovým filtrem.

Každý nový stav odhaduje Kalmanův filtr ve dvou krocích a to v predikčním a korekčním. V predikčním kroku se nový odhad stavového vektoru $\hat{\boldsymbol{x}}^{-}[k]$ (stříškou „ˆ“ značíme, že hodnota $\hat{x}$ je odhadem skutečné hodnoty veličiny x) vypočte na základě hodnot předchozího stavového vektoru $\hat{\boldsymbol{x}}[k-1]$. Využívá tedy tzv. *a priori* informace k odhadu nového stavu, neboli k odhadu vektoru v kroku k používá informací z kroků předchozích. Použití *a posteriori* dat v kroku k tedy znamená použití dat, která byla opravena měřením z kroku k. Veličiny, vypočtené pomocí *a priori* informací značíme s horním indexem „-“ a pro veličiny z *a posteriori* dat používáme horní index „+“. Značení *a posteriori* odhadu horním indexem „+“ a odhadu hodnoty veličiny stříškou „ˆ“ budeme používat jen v místech, kde by mohlo být značení nejednoznačné.

Po výpočtu nového odhadu se také přepočítá kovariance stavového vektoru $\mathbf{P}^{-}[k]$ z matice $\mathbf{P}[k-1]$. Rovnice pro predikční krok filtru mají následující tvar

$$\hat{\boldsymbol{x}}^{-}[k] = \mathbf{F}\hat{\boldsymbol{x}}[k-1]\,, \tag{4.7}$$

$$\mathbf{P}^{-}[k] = \mathbf{F}\mathbf{P}[k-1]\,\mathbf{F}^{\mathrm{T}} + \mathbf{Q}\,. \tag{4.8}$$

Nyní popíšeme rovnici 4.7 trochu blíže. Kovariance stavového vektoru $\mathbf{P}$ je definována jako [10]

$$\mathbf{P} = \mathrm{cov}\,(\boldsymbol{x}) = E\left[(\boldsymbol{x}-\hat{\boldsymbol{x}})\,(\boldsymbol{x}-\hat{\boldsymbol{x}})^{\mathrm{T}}\right]\,, \tag{4.9}$$

podobně pro $\mathbf{P}^{-}$

$$\mathbf{P}^{-} = E\left[(\boldsymbol{x}-\hat{\boldsymbol{x}}^{-})\,(\boldsymbol{x}-\hat{\boldsymbol{x}}^{-})^{\mathrm{T}}\right]\,, \tag{4.10}$$

kde $E[\xi]$ je střední hodnota náhodné veličiny ξ a $\mathrm{cov}\,(\xi) = E\left[(\xi - E[\xi])^2\right]$ je kovariance náhodné veličiny ξ. V rovnici 4.7 se pro predikci následujícího odhadu vynásobí předchozí odhad maticí $\mathbf{F}$. Tím zároveň dojde i k vynásobení kovariance stavů tou samou maticí a při uvážení linearity střední hodnoty resp. kovariance

$$E[aX] = aE[X]\,, \tag{4.11}$$

$$\mathrm{cov}\,(aX, bY) = E\left[(aX - E[aX])\,(bY - E[bY])\right] = abE\left[(X - E[X])\,(Y - E[Y])\right]\,, \tag{4.12}$$

je pak kovariance predikovaného stavu rovna

$$\mathbf{P}^{-} = \mathrm{cov}\,(\hat{\boldsymbol{x}}^{-}) = \mathrm{cov}\,(\mathbf{F}\hat{\boldsymbol{x}} + \boldsymbol{w}) = \mathrm{cov}\,(\mathbf{F}\hat{\boldsymbol{x}}) + \mathrm{cov}\,(\boldsymbol{w}) = \mathbf{F}\mathbf{P}\mathbf{F}^{\mathrm{T}} + \mathbf{Q}\,. \tag{4.13}$$

V rovnici 4.18 pro $\mathbf{P}^{-}$ se nachází i kovariance modelu $\mathbf{Q}$, čímž se do kovariance stavu zahrnují i nepřesnosti v predikci způsobené nepřesností modelu. Přičtením $\mathbf{Q}$ se $\mathbf{P}^{-}$ po každém predikčním kroku o něco navýší (samozřejmě pro $\mathbf{Q}$ nenulové) a tak se zvýší nejistota v predikovaném stavu.

Korekční krok používá naměřených dat $\boldsymbol{z}[k]$ v čase k, tedy *a posteriori* informací, pro výpočet Kalmanova zesílení $\mathbf{K}$ a tak i ke korekci odhadu $\hat{\boldsymbol{x}}^{-}[k]$.

$$\boldsymbol{y}[k] = \boldsymbol{z}[k] - \mathbf{H}\hat{\boldsymbol{x}}^{-}[k] \tag{4.14}$$

V rovnici 4.14 se $\boldsymbol{y}$ nazývá vektorem inovací nebo také residuálů (my se budeme držet prvního uvedeného termínu). Rovnice udává o kolik se nový odhad $\hat{\boldsymbol{x}}^{-}$ liší od reálného měření $\boldsymbol{z}$ a tedy v ideálním případě, kdy by matematický model reprezentovaný $\mathbf{F}$ perfektně popisoval

skutečný proces, by byl $\boldsymbol{y}$ roven nulovému vektoru. Podobně jako v predikčním kroku se i zde vypočte kovarianční matice inovace $\mathbf{S}$.

$$\mathbf{S}\left[k\right] = \mathbf{H}\mathbf{P}^{-}\left[k\right]\mathbf{H}^{\mathrm{T}} + \mathbf{R} \tag{4.15}$$

Zde matice $\mathbf{R}$ vyjadřuje nepřesnosti měřených dat, tedy na hlavní diagonále jsou variance měření a mimo diagonálu jsou kovariance mezi měřeními. Pokud jsou měření lineárně nezávislá, pak je matice $\mathbf{R}$ diagonální a na diagonále je rozptyl každé měřené veličiny.

Dále je vypočten Kalmanův zisk $\mathbf{K}$, který je dále použit k výpočtu *a posteriori* odhadu $\hat{\boldsymbol{x}}$.

$$\mathbf{K}\left[k\right] = \mathbf{P}^{-}\left[k\right]\mathbf{H}^{\mathrm{T}}\mathbf{S}^{-1}\left[k\right] \tag{4.16}$$

$$\hat{\boldsymbol{x}}\left[k\right] = \hat{\boldsymbol{x}}^{-}\left[k\right] + \mathbf{K}\left[k\right]\boldsymbol{y}\left[k\right] \tag{4.17}$$

Při dosazení za $\mathbf{S}$ z rovnice 4.15 do rovnice pro Kalmanovo zesílení 4.16 si můžeme všimnou, že mohou nastat dva extrémy. První situace nastává, když jsou prvky matice $\mathbf{R}$ blízké nebo dokonce rovny nule. To potom znamená, že získaná měření jsou velice přesná, Kalmanův zisk se blíží $\lim_{\mathbf{R}\to\mathbf{0}} \boldsymbol{K} = \mathbf{H}^{-1}$ a tedy se při výpočtu $\hat{\boldsymbol{x}}$ se více přihlíží na naměřené hodnoty a méně na $\hat{\boldsymbol{x}}^{-}$. V druhém případě, kdy si filtr „věří“ v odhadu stavu $\hat{\boldsymbol{x}}^{-}$, se k nule blíží prvky matice $\mathbf{P}^{-}$. S tím i Kalmanův zisk klesá k nule a pro výpočet $\hat{\boldsymbol{x}}$ se tak použije více $\hat{\boldsymbol{x}}^{-}$ a naopak se použije méně $\boldsymbol{z}$.

Posledním výpočtem, který se provede v korekčním kroku, je výpočet *a posteriori* kovarianční matice $\mathbf{P}\left[k\right]$ stavového vektoru $\hat{\boldsymbol{x}}\left[k\right]$.

$$\mathbf{P}\left[k\right] = \left(\mathbf{I} - \mathbf{K}\left[k\right]\mathbf{H}^{\mathrm{T}}\right)\mathbf{P}^{-}\left[k\right] , \tag{4.18}$$

kde $\mathbf{I}$ je jednotková matice.

Tímto je iterace Kalmanova filtru dokončena a při dalším opakovaní v čase $k+1$ se začne rovnicí 4.3, ale tentokrát s $\hat{\boldsymbol{x}}$ a $\mathbf{P}$ získanými z rovnic 4.17 a 4.18 v čase k.

Jak bylo psáno výše, Kalmanův filtr je rekurentní algoritmus a tak je pro něj potřeba stanovit počáteční podmínky a to jmenovitě $\hat{\boldsymbol{x}}\left[0\right]$ a $\mathbf{P}\left[0\right]$. Tyto počáteční podmínky ovlivňují zejména rychlost konvergence filtru ke správným odhadovaným hodnotám.

Také je třeba určit matice $\mathbf{R}$ a $\mathbf{Q}$. Kovarianční matice měření $\mathbf{R}$ se vztahuje ke zdrojům měření a nese na své diagonále informaci o rozptylech každé veličiny a mimo diagonálu kovariance mezi veličinami. V běžných aplikacích (kdy jsou měření lineárně nezávislá) nalézáme prvky $\mathbf{R}$ mimo diagonálu nulové. Pokud jde o časově neproměnnou matici $\mathbf{R}$, lze hodnoty určit poměrně jednoduše pomocí sady statických měření a statistických výpočtů, které určí střední hodnotu i směrodatnou odchylku šumu v datech. Pokud se matice v čase mění, tak ji je třeba v každém kroku dopočítávat.

Určení podoby kovarianční matice procesu $\mathbf{Q}$ už naneštěstí tak jednoduché není. Je to díky tomu, že (běžně) nedokážeme přímo pozorovat odhadovaný proces a tak určit odlišnosti. Také je možné, že použitý model sice neúplně odpovídá odhadovanému procesu, ale zároveň už lepší model třeba nedokážeme získat či implementovat. Pro určení hodnot matice $\mathbf{Q}$ vzniklo několik postupů a bylo na toto téma napsáno mnoho prací. Jedním z těch sofistikovanějších je například *Autocovariance Least-Square Method*, který je popsán v [12]. Příkladem méně sofistikované metody může být odladění hodnot pomocí datasheetu nebo metodou pokus-omyl a změnami $\mathbf{Q}$ dokud Kalmanův filtr nezačne odhadovat s dostatečnou přesností.

4.2.2 Kalmanův filtr pro synchronizaci

V této části budou popsány matice pro Kalmanův filtr, který bude odhadovat rozdíl hodin DW1000 čipů synchronizační master kotvy a slave kotvy. Ve zdroji [9] je filtr implementován jako dvoustavový, kde odhadovali jak offset hodin mezi slave a master kotvou Δ, tak chybu frekvence Δf. Avšak po řadě měření s dvoustavovým filtrem jsme pro zlepšení přesnosti synchronizace rozšířili Kalmanův filtr ještě o třetí stav a to o změnu frekvence $\Delta\dot{f}$. Dále budeme tedy uvádět matice filtru pro dvoustavový i pro třístavový KF a na konci sekce obě varianty porovnáme. Matice a vektory, které se budou vztahovat ke dvoustavovému KF označíme dolním indexem „2“, pro třístavový použijeme dolní index „3“,

Prvním, co můžeme ihned definovat, je stavový vektor $\boldsymbol{x}$. Pro dvoustavový filtr je vektor roven

$$\boldsymbol{x}_2 = \begin{pmatrix} x_0 \\ x_1 \end{pmatrix} = \begin{pmatrix} \Delta \\ \Delta f \end{pmatrix} \tag{4.19}$$

a pro třístavový

$$\boldsymbol{x}_3 = \begin{pmatrix} x_0 \\ x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} \Delta \\ \Delta f \\ \Delta\dot{f} \end{pmatrix} . \tag{4.20}$$

Při uvážení rovnice pro čas hodin kotvy 4.1 platí pro časový vývoj stavů následující rovnice

$$\Delta\left[k+1\right] = \Delta\left[k\right] + \Delta t\left[k\right]\Delta f\left[k\right] + \frac{1}{2}\Delta t^2\left[k\right]\Delta\dot{f}\left[k\right] , \tag{4.21}$$

kde $\Delta t\left[k\right]$ je doba mezi přijetí synchronizační zprávy v čase $k+1$ a v čase k. Z toho můžeme snadno vypozorovat matici $\mathbf{F}$ pro dvoustavový filtr

$$\mathbf{F}_2 = \begin{bmatrix} 1 & \Delta t\left[k\right] \\ 0 & 1 \end{bmatrix} , \tag{4.22}$$

resp. pro třístavový

$$\mathbf{F}_3 = \begin{bmatrix} 1 & \Delta t\left[k\right] & \frac{1}{2}\Delta t^2\left[k\right] \\ 0 & 1 & \Delta t\left[k\right] \\ 0 & 0 & 1 \end{bmatrix} . \tag{4.23}$$

Rovnice 4.2 umožňuje přímo vyjádřit rozdíl časových základen Δ mezi master a slave kotvou

$$\Delta = t^S_{Rx} - t^M_{Tx} - \frac{\parallel \boldsymbol{r}^S - \boldsymbol{r}^M \parallel}{c} \tag{4.24}$$

Když toto zkombinujeme s rovnicí 4.14 dostaneme převodní matici $\mathbf{H}$ pro dvoustavový filtr

$$\mathbf{H}_2 = \begin{bmatrix} 1 \\ 0 \end{bmatrix} , \tag{4.25}$$

resp. pro třístavový

$$\mathbf{H}_3 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} . \tag{4.26}$$

Nyní stačí vyčíslit matice $\mathbf{R}$ a $\mathbf{Q}$. Matice kovariance měření je pro obě formy filtru stejná. Díky to mu, že každou iteraci filtru provádíme pouze jedno měření je $\mathbf{R}$ pouze skalárem. Dále

také pro jednoduchost uvažujeme, že se hodnota $\mathbf{R}$ v čase nemění. Vyčíslení $\mathbf{R}$ je uvedeno v [9], kde je kovariance měření rovna $\mathbf{R} = \left[\sigma_v^2\right] = 3 \cdot 10^{-20}\ \mathrm{s}^2$. Ovšem toto zjednodušení může zhoršovat výsledky v případech, kdy je poměr síly přijaté synchronizační zprávy ku šumu nízký. V tu chvíli bude $\mathbf{R}$ různé od toho, které jsme uvedli výše, a to se negativně podepíše na přesnosti synchronizace a následně i na přesnosti lokalizace. Nicméně v aplikacích je předpokládáno, že rozmístění kotev i výběr master kotvy bude takový, aby tyto případy nastávaly minimálně. Potom předpoklad neměnného $\mathbf{R}$ neznehodnotí dostatek synchronizací, aby znemožnil lokalizování tagů.

Složitější je vyčíslení hodnot kovarianční matice procesu $\mathbf{Q}$. Matice vyjadřuje nejistoty zahrnuté v *a priori* odhadu $\hat{\boldsymbol{x}}^-$ a protože časový interval mezi jednotlivými odhady $\hat{\boldsymbol{x}}^-$ nemusí nutně být stále stejný (perioda synchronizace se může změnit nebo se synchronizační zpráva ztratí), musí tuto skutečnost odrážet i matice $\mathbf{Q}$ a nemůže být tak stálá v čase. Je tak vhodné násobit matici v čase k odpovídajícím $\Delta t\,[k]$

$$\mathbf{Q}\,[k] = \mathbf{Q}_0 \Delta t\,[k] \tag{4.27}$$

Jednotlivé prvky matice $\mathbf{Q}_0$ jsme odhadli na základě sady měření a použitého modelu procesu.

Na konec sekce uvedeme výpočet odhadu $\hat{\Delta}$ offsetu hodin Δ, který použijeme pro převedení času přijetí z časové domény slave kotvy do časové domény master kotvy. Nejprve označme Δt_m čas, který uplynul od poslední synchronizace do času přijetí *blink* zprávy

$$\Delta t_m = t_{\mathrm{Rx,\ blink}} - t_{\mathrm{Rx,\ sync}}\,. \tag{4.28}$$

Za čas Δt_m se offset Δ od posledního odhadu během synchronizace v čase vyvinul a tak pro jeho odhad v čase přijetí *blink* zprávy provedeme samotný predikční krok KF jen pro $\hat{\Delta}$ neboli $\hat{x}_0$.

$$\hat{x}_0\,(t_{\mathrm{Rx,\ blink}}) = \begin{bmatrix} 1 & \Delta t_m & \frac{1}{2}\Delta t_m^2 \end{bmatrix} \begin{pmatrix} \hat{x}_0 \\ \hat{x}_1 \\ \hat{x}_2 \end{pmatrix} \tag{4.29}$$

Společně se stavem $\hat{x}_0$ se však vyvinula i jeho kovariance, kterou vypočteme následovně

$$\sigma_{\hat{x}_0}^2 = \begin{bmatrix} 1 & \Delta t_m & \frac{1}{2}\Delta t_m^2 \end{bmatrix} \mathbf{P} \begin{bmatrix} 1 \\ \Delta t_m \\ \frac{1}{2}\Delta t_m^2 \end{bmatrix} + \Delta t_m Q_{\hat{x}_0}\,, \tag{4.30}$$

kde $Q_{\hat{x}_0}$ je prvek kovarianční matice procesu $\mathbf{Q}_0$, který odpovídá stavu $\hat{x}_0$.

4.2.2.1 Porovnání

V této části provedeme porovnání obou vytvořených verzí Kalmanova filtru. Tedy dvoustavový KF, který odhaduje časový offset hodin Δ^{MS} master a slave kotvy a rozdíl frekvence hodin obou zařízení Δf^{MS}, a třístavový KF, který navíc odhaduje i změnu frekvence $\Delta \dot{f}^{MS}$.

Přesnost filtrů je určována dle odchylky odhadnuté hodnoty a naměřené hodnoty a to podle rovnice pro odchylku offsetu hodin [7]

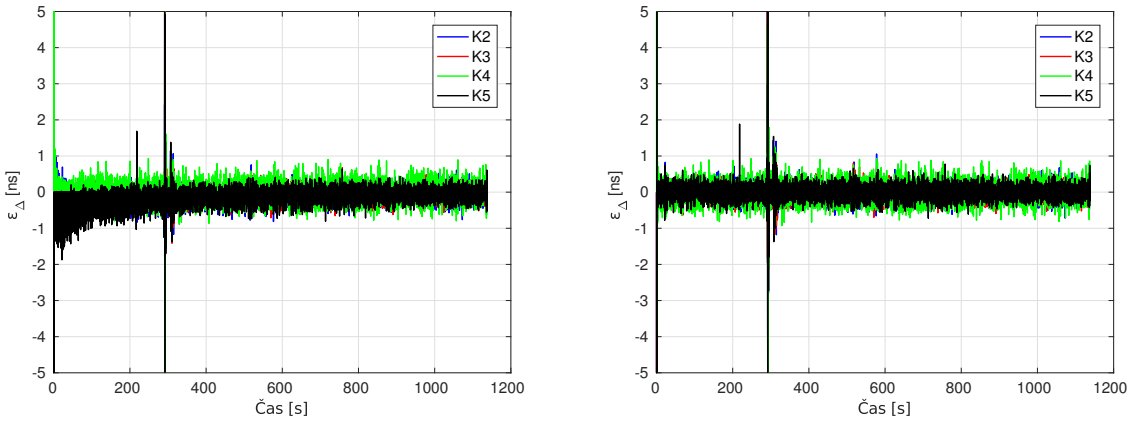
$$\varepsilon_\Delta\,[k] = \left(t_{Rx}^S\,[k] - t_{Tx}^M\,[k] - \frac{\parallel \boldsymbol{r}^S - \boldsymbol{r}^M \parallel}{c} \right) - \Delta\,[k]\ , \tag{4.31}$$

kde c je rychlost šíření signálu, a podle rovnice pro odchylku chyby frekvence

$$\varepsilon_{\Delta f}[k] = \left(t^S_{Rx}[k] - t^M_{Tx}[k] - \left(t^S_{Rx}[k-1] - t^M_{Tx}[k-1]\right)\right) - \Delta f[k] \ . \tag{4.32}$$

V rovnicích 4.31 a 4.32 značí $t^S_{Rx}[k]$ čas přijetí synchronizační zprávy slave kotvou v čase (zprávě) k a $t^M_{Tx}[k]$ čas vyslání synchronizační zprávy v čase (zprávě) k.

Měření pro porovnání proběhlo na testovacím polygonu v RCD. Synchronizovány byly čtyři kotvy s periodou 100 ms.



(a) KF 2 stavy

(b) KF 3 stavy

Obrázek 4.1: Odchylka odhadnutého offsetu hodin a jeho skutečné hodnoty

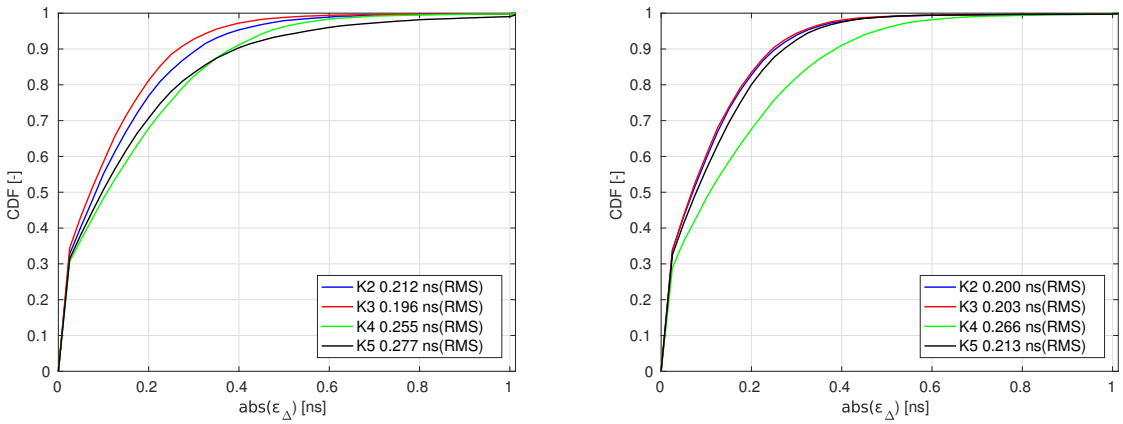
Na obrázku 4.1 můžeme vidět odchylku offsetu hodin pro jednotlivé kotvy, která byla vypočtena podle rovnice 4.31. Hodnota Δ je stěžejní pro synchronizaci a tak s přesnější synchronizací a přesnějšími odhady stavových veličin bude ε_Δ klesat k nule. Samozřejmě vlivem nepřesností typu špatného zaměření kotev, omezená přesnost aritmetiky, diskretizace spojitého děje a podobně, odchylka ε_Δ nikdy nedosáhne přesné nuly, ale spíš bude oscilovat kolem nuly.

Je vidět, že největší chybovosti synchronizace dosahuje krátce po spuštění kotev resp. synchronizace. To je způsobeno především filtrem, který během prvních několika iterací konverguje. Zvýšená chybovost je způsobena i zahříváním oscilátorů kotev, ale o tom napíšeme více později, kdy budeme diskutovat odhad rozdílu frekvence Δf.

Můžeme si všimnout, že třístavový filtr konverguje rychleji než dvoustavový. To je nejlépe vidět v porovnání obrázků 4.1a a 4.1b, kde na prvním zmiňovaném filtr konverguje přibližně 100 s, zatímco třístavový filtr potřebuje ke konvergenci řádově jednotky sekund.

Zavedením třetího stavu jsme také dosáhli zlepšení odhadu offsetu hodin pro kotvu K5. Na obrázku 4.1a je odhad offsetu pro kotvu K5 zatížen chybou, která souvisí s ohříváním oscilátoru a změnou jeho frekvence. Vlivem toho je $\varepsilon_\Delta[k]$ od počátku vychýlený a společně s ustalující se teplotou oscilátoru se odhad offsetu blíží skutečné hodnotě a odchylka se tak svou střední hodnotou blíží k nule. Oproti tomu třístavový filtr bere měnící se frekvenci v úvahu a proto odchylky nevykazují tak výrazné vychýlení.

Jaké přesnosti dosahujeme je více přehledné na následujících obrázcích 4.2, na kterých je kumulativní distribuční funkce absolutní chyby odhadu offsetu ε_Δ pro oba filtry.



(a) KF 2 stavy

(b) KF 3 stavy

Obrázek 4.2: Distribuční funkce odchylky odhadnutého a skutečného offsetu hodin

Z distribuční funkce na obrázku 4.2a je zřejmé, že u dvoustavového KF je z 90% chyba odhadu menší než 0,4 ns. Pro třístavový KF je z 90% chyba odhadu v kotvách na obrázku 4.2b dokonce menší než 0,3 ns. To však neplatí pro kotvu K4, která dosahuje opravdu dobrou shodou okolností podobných hodnot Δf a $\Delta \dot{f}$ jako master kotva. Toto bude více zřejmé na obrázcích 4.3a a 4.3b zachycujících odchylku frekvence Δf. Proto výsledky synchronizace kotvy K4 přidání třetího stavu KF výrazně nezlepší, ale ani nezhorší. Můžeme ale také pozorovat výrazné zlepšení přesnosti synchronizace u kotvy K5, jejíž RMS odchylka je menší o 0,064 ns.

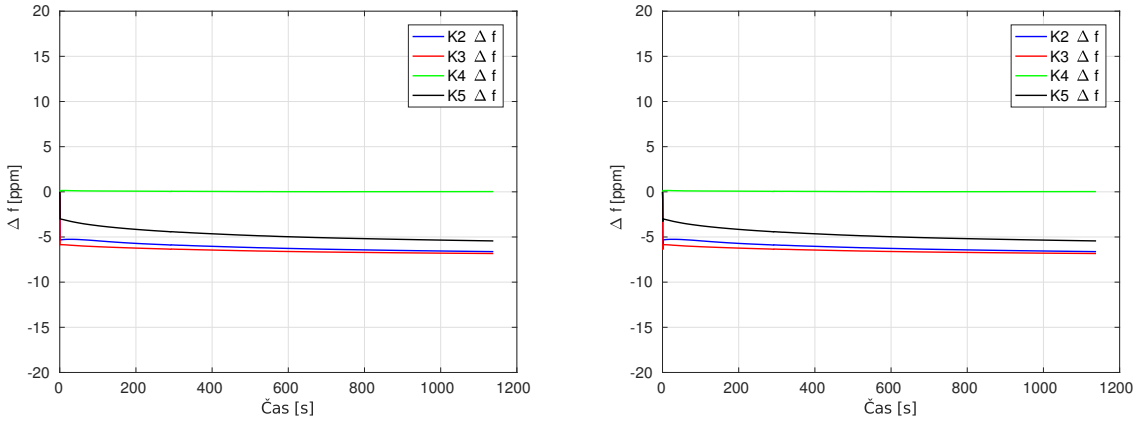
Hodnoty RMS pro odchylky jsou vypočteny podle vzorce

$$\mathrm{RMS} = \sqrt{\frac{1}{N}\sum_{k=0}^{N-1} |s\,[k]|^2}\,, \tag{4.33}$$

kde $s\,[k]$ je obecný signál vzorkovaný v diskrétním čase k.

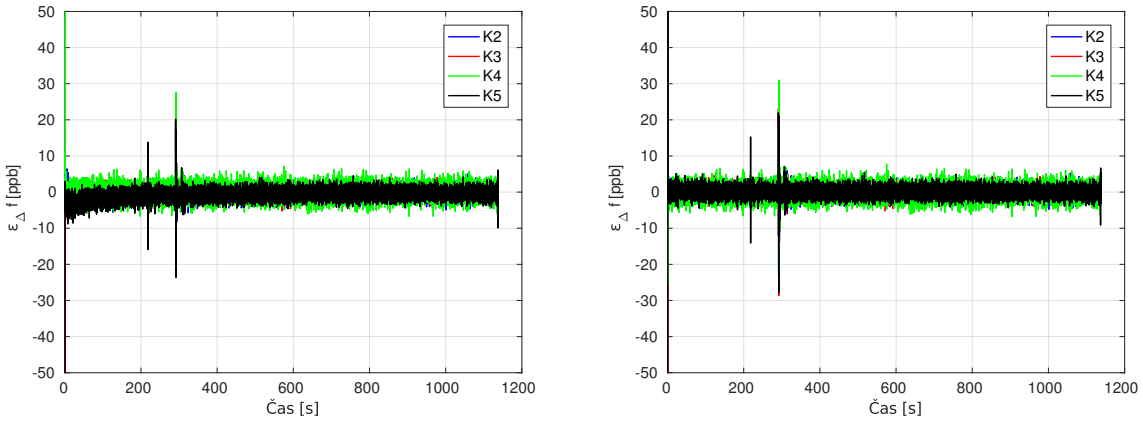
Obrázky 4.3 níže zachycují vývoj stavu filtru Δf, který popisuje rozdíl frekvence master a slave oscilátoru. Z nich můžeme vypozorovat vliv teploty na frekvenci oscilátorů. Než se hodnota Δf společně s teplotou oscilátorů ustálí (přibližně po deseti minutách chodu) je vidět, že se frekvence oscilátorů slave kotev oproti master kotvě skutečně mění. U kotvy K4 je také vidět, že odchylka její frekvence od master kotvy je minimální (ne však nulová). To poukazuje na to, že alespoň pro toto měření master kotva a kotva K4 vykazovaly stejné chyby oscilátorů.

Z obrázku 4.4a můžeme opět vidět, jak absence třetího stavu ovlivňuje rychlost konvergence odhadu chyby frekvence ve dvoustavovém filtru. Na druhém obrázku 4.4b také vidíme, že chyba frekvence ve třístavovém filtru se začne správně odhadovat již několik sekund po spuštění.



(a) KF 2 stavy (b) KF 3 stavy

Obrázek 4.3: Odhad chyby frekvence Δf



(a) KF 2 stavy (b) KF 3 stavy

Obrázek 4.4: Odchylka skutečné a odhadnuté chyby frekvence $\varepsilon_{\Delta f}$

Na závěr sekce uvádíme tabulku shrnující rozdíl ve výsledcích synchronizace vytvořených filtrů.

Kotva	KF2 [ps]	[cm]	KF3 [ps]	[cm]	Zlepšení pro KF3 [%]
K2	212	6,4	200	6	5,7 %
K3	196	5,9	203	6,1	−3,6 %
K4	255	7,6	266	8	−4,3 %
K5	277	8,3	213	6,4	23,1 %

Tabulka 4.1: Porovnání RMS hodnot odchylek offsetů obou filtrů

Už z výše uvedených grafů je vidět jisté zlepšení přesnosti synchronizace třístavového filtru oproti dvoustavovému filtru. To je díky tomu, že třetí stav popisuje změnu frekvence a tak filtr nepředpokládá neměnnou frekvenci. Dvoustavový filtr tak umožňuje pouze takové změny frekvence, které jsou v mezích směrodatné odchylky dané maticí **Q**. Čeho jsme tím dosáhli je zvýšení rychlosti konvergence k „přesným“ odhadům stavových veličin, zvýšení celkové přenosti synchronizace a potlačení nedokonalostí oscilátorů kotev.

Jak můžeme vidět v tabulce 4.1, tak třístavový filtr dokázal potlačit chybovost kotvy K5 a zároveň přidáním třetího stavu došlo ke zlepšení přesnosti synchronizace. Společně se snížením hodnoty RMS pro kotvy K2 a K5 došlo i k mírnému zvětšení chyby u kotev K3 a K4.

V dalším vývoji UWB systému jsme pokračovali se třístavovým filtrem, který dokáže zmírnit dopad měnící se teploty oscilátorů kotev a tím zvýšit odolnost synchronizace proti vlivu okolí.

4.3 Výpočet polohy

Lokalizace za pomocí TDoA se nazývá hyperbolickou navigací, neboť rozdíl v čase přijetí *blink* zprávy ve dvou kotvách je pro dané měření k konstantní a množina bodů, kde se mohl tag nacházet, tvoří hyperbolu. K výpočtu polohy je tedy nutné proložit hyperboly, získané z naměřených TDoA, a najít jejich průsečík, který je bodem, kde se nacházel tag v době vyslání lokalizační *blink* zprávy.

Pro každé z N měření časů přijetí v čase k dokážeme získat $N-1$ lineárně nezávislých rozdílů časů přijetí TDoA.

$$\mathrm{TDoA}_{i,j}\left[k\right] = t_{Rx,i}\left[k\right] - t_{Rx,j}\left[k\right] , \tag{4.34}$$

kde $i = \{1, 2, \ldots, N\}, j = \{1, 2, \ldots, N\}, i \neq j$ a $t_{Rx,i}$, resp. $t_{Rx,j}$ je čas přijetí *blink* zprávy v $i$té, resp. $j$té kotvě.

Pro hledání průsečíku hyperbol je však vhodnější počítat s TDoA vzdáleností $d_{i,j}\left[k\right]$, kterou získáme vynásobením příslušného $\mathrm{TDoA}_{i,j}\left[k\right]$ rychlostí šíření signálu c (rychlostí světla). V dalším textu budeme pro úsporu místa uvažovat, že se jedná o data získaná ve stejném čase k a tak budeme index vynechávat.

Je zřejmé, že $d_{i,j}$ musí být rovno

$$d_{i,j} = \| \boldsymbol{r}_i - \boldsymbol{r} \| - \| \boldsymbol{r}_j - \boldsymbol{r} \| , \tag{4.35}$$

kde $\boldsymbol{r}_i$, resp. $\boldsymbol{r}_j$ je polohový vektor $i$té, resp. $j$té kotvy a $\boldsymbol{r}$ je poloha tagu. Rovnice 4.35 je zároveň rovnicí hyperboly s ohnisky v $i$té a $j$té kotvě. Řešení $\boldsymbol{r}$ splňující rovnici 4.35 pro každou kombinaci $i, j : i \neq j$, kdy jsou rovnice lineárně nezávislé, je odhadem polohy tagu.

Na problém hledání průsečíku je tedy možno nahlížet jako na minimalizaci vzdálenosti bodu s polohovým vektorem $\boldsymbol{r}$ od každé naměřené hyperboly. Jinými slovy, hledání průsečíku několika hyperbol (obecně funkcí) se dá řešit jako problém nejmenších čtverců.

Tento problém je ovšem nelineární a tak je nutno zvolit správnou metodu řešení, která dokáže rychle a bezpečně konvergovat. My jsme se rozhodli pro použití Levenbergova-Marqaurdtova algoritmu.

4.3.1 Levenbergova-Marquardtova metoda řešení nelinárních nejmenších čtverců

Levenbergova-Marquardtova metoda byla prvně popsána již v roce 1944 v [13] a posléze v [14] a stala se, dá se říci, standardem pro řešení problému nelineárních nejmenších čtverců. Metoda

vznikla kombinací gradientního algoritmu největšího spádu a Gaussovy-Newtonovy metody, díky čemuž vyniká rychlou konvergencí a nízkým rizikem divergence.

Nejdříve definujme funkci, kterou chceme minimalizovat. Začneme přepsáním rovnice 4.35 do následujícího tvaru

$$\delta_{i,j}(\boldsymbol{r}) = d_{i,j} - (\parallel \boldsymbol{r}_i - \boldsymbol{r} \parallel - \parallel \boldsymbol{r}_j - \boldsymbol{r} \parallel) = d_{i,j} - \hat{d}_{i,j}(\boldsymbol{r}) \,, \tag{4.36}$$

kde $\delta_{i,j}(\boldsymbol{r})$ vyjadřuje o kolik se TDoA vzdálenost $\hat{d}_{i,j}(\boldsymbol{r})$ vypočtená v bodě s polohovým vektorem $\boldsymbol{r}$ liší od naměřené $d_{i,j}$. Vektorem $\boldsymbol{\delta}$ je pak myšlen souhrn všech lineárně nezávislých měření, který, bez újmy na obecnosti, definujeme jako sloupcový vektor

$$\boldsymbol{\delta} = \begin{pmatrix} \delta_{1,2} \\ \delta_{2,3} \\ \vdots \\ \delta_{N-1,N} \end{pmatrix} = \boldsymbol{d} - \hat{\boldsymbol{d}}(\boldsymbol{r}) \,. \tag{4.37}$$

Hodnoty v každém řádku $\boldsymbol{\delta}$ jsou rovny rozdílu naměřené $d_{i,j}$ a vypočtené TDoA vzdálenosti $\hat{d}_{i,j}$ mezi „sousedními“ kotvami (sousední ve smyslu indexů). S pomocí $\boldsymbol{\delta}$ definujme funkci $y(\boldsymbol{r})$, kterou budeme minimalizovat

$$y(\boldsymbol{r}) = \sum_{i=1}^{N-1} \delta_{i,i+1}^2(\boldsymbol{r}) = \parallel \boldsymbol{\delta}(\boldsymbol{r}) \parallel^2 \,. \tag{4.38}$$

Dále si rozdělme odhadovaný vektor $\boldsymbol{r}$, ukazující na odhadnutou polohu tagu, na predikční a korekční část

$$\boldsymbol{r} = \boldsymbol{r}_p + \boldsymbol{r}_c \,. \tag{4.39}$$

Predikční část je $\boldsymbol{r}$ vypočtené v předchozí iteraci algoritmu (nebo počáteční vektor jedná-li se o první iteraci). Výpočet korekční části bude popsán níže.

V úvodu této sekce jsme zmínili, že Levenbergova-Marquardtova metoda se skládá ze dvou minimalizačních algoritmů. Nyní tyto dva algoritmy krátce popíšeme společně s tím, jak je Levenbergova-Marquardtova metoda spojuje.

Gradientní metoda je iterativní metoda, která slouží k nalezení lokálního minima dané funkce $\boldsymbol{\delta}(\boldsymbol{r})$. Metoda je prvního řádu a tedy používá pouze prvního členu Taylorova rozvoje funkce a vyšší členy zanedbává. Pro určení směru $\boldsymbol{h}$ k minimu funkce je pak použita rovnice [15]

$$\boldsymbol{h} = \lambda \mathbf{J}^{\mathrm{T}}(\boldsymbol{r})\, \boldsymbol{\delta}(\boldsymbol{r}) \,, \tag{4.40}$$

pro $\lambda < 0$ a kde $\mathbf{J}$ je matice prvních derivací (Jakobián) funkce $\boldsymbol{\delta}$ podle $\boldsymbol{r}$. Směr $\boldsymbol{h}$ můžeme ztotožnit s korekcí $\boldsymbol{r}_c$. Metoda má nízké riziko divergence, ale pomalu konverguje.

Oproti tomu Gaussova-Newtonova metoda bere v úvahu i změnu spádu, neboli druhou derivaci, funkce $\boldsymbol{\delta}$. Rovnice pro hledání minima vychází z Taylorova rozvoje pro funkci $\boldsymbol{\delta}$. Směr k minimu funkce $\boldsymbol{h}$ (korekce $\boldsymbol{r}_c$) vypočítává Gaussova-Newtonova metoda následovně [15]

$$\left(\mathbf{J}^{\mathrm{T}}\mathbf{J}\right)\boldsymbol{h} = \mathbf{J}^{\mathrm{T}}\boldsymbol{\delta} \,. \tag{4.41}$$

Matice $\mathbf{J}^{\mathrm{T}}\mathbf{J}$ zde je přibližně rovna Hessiánu $\mathbf{H}$ [15] funkce $y(\boldsymbol{r})$. Tato metoda oproti gradientní metodě rychle konverguje ale s vysokým rizikem divergence.

Úplné odvození rovnic 4.40 a 4.41 je uvedeno v [15].

Dvě výše uvedené metody mají komplementární vlastnosti, což byla motivace pro vytvoření Levenbergovy-Marquardtovy metody (dále už jen jako L-M metoda) jakožto kombinaci předchozích metod. Proto L-M metoda rychle konverguje a diverguje pouze s malým rizikem. Kombinovaná rovnice pro L-M metodu se v literatuře [16] uvádí ve dvou verzích

$$\left(\mathbf{J}^{\mathrm{T}}\mathbf{J} + \lambda\mathbf{I}\right)\boldsymbol{h} = \mathbf{J}^{\mathrm{T}}\boldsymbol{\delta}\,, \tag{4.42}$$

$$\left(\mathbf{J}^{\mathrm{T}}\mathbf{J} + \lambda\mathrm{diag}\left(\mathbf{J}^{\mathrm{T}}\mathbf{J}\right)\right)\boldsymbol{h} = \mathbf{J}^{\mathrm{T}}\boldsymbol{\delta}\,, \tag{4.43}$$

kde matice $\mathbf{I}$ je jednotková a diag $\left(\mathbf{J}^{\mathrm{T}}\mathbf{J}\right)$ je matice, která má diagonálu rovnu diagonále matice matice $\mathbf{J}^{\mathrm{T}}\mathbf{J}$ a ostatní prvky má nulové. Rozdíl mezi rovnicemi 4.42 a 4.43 je v matici, která je násobena koeficientem metody λ. V rovnici 4.42 je λ násobena jednotková matice $\mathbf{I}$, což má za následek, že délka kroku metody směrem k hledanému minimu je neměnná. Druhá rovnice 4.43 nahrazuje jednotkovou matici diagonálou matice $\mathbf{J}^{\mathrm{T}}\mathbf{J}$ a díky tomu metoda délku kroku adaptivně nastavuje v závislosti na změně spádu funkce $y\left(\boldsymbol{r}\right)$.

Dále existují také vážené verze rovnic 4.42 a 4.43, které mají tvar

$$\left(\mathbf{J}^{\mathrm{T}}\mathbf{W}\mathbf{J} + \lambda\mathbf{I}\right)\boldsymbol{h} = \mathbf{J}^{\mathrm{T}}\mathbf{W}\boldsymbol{\delta}\,, \tag{4.44}$$

$$\left(\mathbf{J}^{\mathrm{T}}\mathbf{W}\mathbf{J} + \lambda\mathrm{diag}\left(\mathbf{J}^{\mathrm{T}}\mathbf{W}\mathbf{J}\right)\right)\boldsymbol{h} = \mathbf{J}^{\mathrm{T}}\mathbf{W}\boldsymbol{\delta}\,, \tag{4.45}$$

kde $\mathbf{W}$ je matice vah, která může být rovna inverzi kovarianční matice (pokud tyto kovariance známe) měření $\boldsymbol{\Phi}$ [15]

$$\mathbf{W} = \boldsymbol{\Phi}^{-1}\,. \tag{4.46}$$

V našem případě bude matice $\mathbf{W}$ rovna inverzi matice kovariance naměřených TDoA vzdáleností $\boldsymbol{\Phi_d}$, která má tvar [7]

$$\boldsymbol{\Phi_d} = \mathbf{M}\boldsymbol{\Phi}_{t_{Rx}}\mathbf{M}^{\mathrm{T}}c^2 = \boldsymbol{\Phi}_{\mathrm{TDoA}}c^2\,, \tag{4.47}$$

kde $\mathbf{M}$ je matice, která kombinuje jednotlivé časy přijetí pro vytvoření TDoA, $\boldsymbol{\Phi}_{t_{Rx}}$ je matice variancí každého času přijetí a c je rychlost šíření signálu.

Jak jsme uvedli výše, tvoříme TDoA měření odečtením času přijetí ze dvou „sousedních“ kotev, proto má matice $\mathbf{M}$ velikost $N-1\times N$ a tvar

$$\mathbf{M} = \begin{bmatrix} 1 & -1 & 0 & \dots & 0 & 0 \\ 0 & 1 & -1 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & 1 & -1 \end{bmatrix}. \tag{4.48}$$

Protože jsou jednotlivá měření času lineárně nezávislá, je $\boldsymbol{\Phi}_{t_{Rx}}$ diagonální

$$\boldsymbol{\Phi}_{t_{Rx}} = \begin{bmatrix} \sigma^2_{t_{Rx,1}} & 0 & \dots & 0 \\ 0 & \sigma^2_{t_{Rx,2}} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \sigma^2_{t_{Rx,N}} \end{bmatrix}, \tag{4.49}$$

kde $\sigma^2_{t_{Rx,i}}$ je kovariance měřeného času přijetí v $i$té kotvě, kterou dokážeme získat přímo ze synchronizačního algoritmu. Kovarianci vypočteme s využitím následujícího vztahu [7]

$$\sigma^2_{t_{Rx,i}} = \sigma^2_{\hat{x}_{0,i}} + \sigma^2_v\,, \tag{4.50}$$

kde $\sigma^2_{\tilde{x}_{0,i}}$ je kovariance offsetu hodin master a $i$té slave kotvy z rovnice 4.30 a $\sigma_v^2 = 3 \cdot 10^{-20}$ s^2 je variance měření času [17].

Koeficient λ nám umožňuje ovlivnit chování metody, kdy nastavením velkého λ se metoda chová spíše jako gradientní. Toho se využívá v prvních iteracích metody, kdy se chceme rychle přiblížit k řešení. Pro malé hodnoty λ se tedy metoda chová spíše jako Gaussova-Newtonova. Koeficient λ také měníme v závislosti na změně chyby na konci iterace. Pokud jsme v dané iteraci chybu zvětšili, zvětšíme i λ, abychom se rychleji dostali zpět do okolí řešení. Pokud se chyba dále zmenší, zmenšíme i λ [16].

Nyní naznačíme odvození matice $\mathbf{J}$. Matice $\mathbf{J}$ je jakobián vektoru $\boldsymbol{\delta}$ podle $\boldsymbol{r}$, tedy

$$\mathbf{J} = \left[\frac{\partial \boldsymbol{\delta}}{\partial \boldsymbol{r}}\right] = \begin{bmatrix} \frac{\partial \delta_{1,2}}{\partial \boldsymbol{r}} \\ \vdots \\ \frac{\partial \delta_{N-1,N}}{\partial \boldsymbol{r}} \end{bmatrix}. \tag{4.51}$$

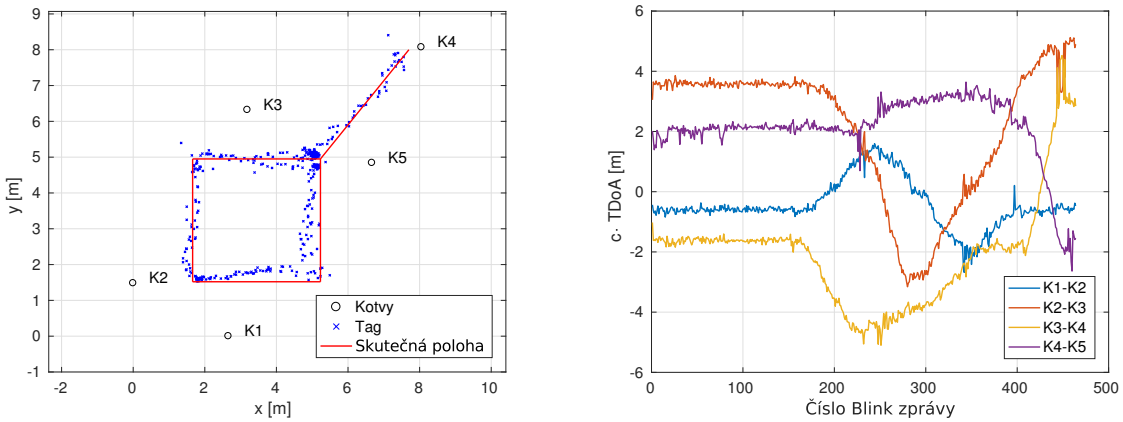
A protože parciální derivace $\frac{\partial \delta_{i,j}}{\partial \boldsymbol{r}}$ je rovna

$$\frac{\partial \delta_{i,j}}{\partial \boldsymbol{r}} = \frac{\partial}{\partial \boldsymbol{r}} \left(d_{i,j} - \sqrt{(\boldsymbol{r}_i - \boldsymbol{r})^2} + \sqrt{(\boldsymbol{r}_j - \boldsymbol{r})^2} \right) = \frac{\boldsymbol{r}_i - \boldsymbol{r}}{\parallel \boldsymbol{r}_i - \boldsymbol{r} \parallel} - \frac{\boldsymbol{r}_j - \boldsymbol{r}}{\parallel \boldsymbol{r}_j - \boldsymbol{r} \parallel} = \mathbf{1}_{i,r} - \mathbf{1}_{j,r}\,, \tag{4.52}$$

kde $\mathbf{1}_{i,r}$ je jednotkový vektor ukazující od $i$té kotvy k odhadované poloze tagu polohovým vektorem $\boldsymbol{r}$, tak má $\mathbf{J}$ tvar

$$\mathbf{J} = \begin{bmatrix} (\mathbf{1}_{0,r} - \mathbf{1}_{1,r})^{\mathrm{T}} \\ (\mathbf{1}_{1,r} - \mathbf{1}_{2,r})^{\mathrm{T}} \\ \vdots \\ (\mathbf{1}_{N-1,r} - \mathbf{1}_{N,r})^{\mathrm{T}} \end{bmatrix}. \tag{4.53}$$

Na závěr kapitoly uvedeme výsledek měření, ve kterém lokalizování tagu proběhlo podle postupu uvedeného v této kapitole.



(a) Vypočtené polohy

(b) Naměřená TDoA

Obrázek 4.5: Výpočet polohy L-M metodou

5 Filtrace polohy Kalmanovým filtrem

Levenbergovou-Marquardtovou metodou, popsanou v předchozí kapitole, jsme vypočetli polohy tagu. Nicméně jak můžeme vidět z obrázku 4.5, vypočtené polohy v některých místech nejsou shodné s těmi skutečnými. Můžeme si všimnout jistého rozptylu a také výskytu odhadů, které jsou zjevně chybné, označovaných jako tzv. *outliers*.

Příčiny vybraných chyb obsažených ve vypočtených polohách můžeme klasifikovat do následujících kategorií.

Omezená přesnost lokalizačního systému - Přesnost lokalizace, kdy kotvy mají přímou viditelnost na tag, značně závisí na přesnosti synchronizace jednotlivých kotev a přesnosti časového značkování zpráv a méně značně na numerické přesnosti minimalizačního algoritmu. Důsledkem je, že v případě, kdy se uživatel s tagem nehýbe, nejsou vypočtené polohy stacionární, ale mají tendenci se s jistým rozptylem „pohybovat“ okolo skutečné polohy. Tyto chyby jsou aditivní a náhodného charakteru, obvykle je lze popsat vícerozměrným normálním rozložením s nulovou střední hodnotou.

Geometrie kotev - Tento zdroj chyby souvisí se samotným umístěním kotev. Pro každou konfiguraci kotev lze v dané oblasti určit místa, kde bude lokalizace přesnější (nízké DOP - *Dillution of Precision*) a kde bude méně přesná (vyšší DOP). V místech s nízkým DOP se hyperboly protínají pod větším úhlem a tak je oblast nejistoty menší (viz obrázek 3.4a). V místech, kde je DOP naopak vyšší, se hyperboly protínají pod menším úhlem a oblast nejistoty je tak větší. Příkladem může být měření TDoA mezi kotvami A,B proti kotvě C, kde A a B jsou z pohledu kotvy C blízko k sobě.

Ztráta přímé viditelnosti - Ztráta přímé viditelnosti způsobí, že signál k přijímači cestuje jinou trasou než tou nejkratší a dostane se k němu pomocí odrazů od okolních objektů. Což tedy zanese do časů přijetí v kotvách chybu, neboť signál cestoval ke každé kotvě po delší trase, než by měl. Pak průsečíky hyperbol (pokud vůbec existují) se nacházejí daleko od skutečné polohy, čímž vznikne poloha typu *outlier*. Vlivem odrazu signál také ztratí část svého výkonu a tak se rychleji utlumí a bude mít menší dosah.

Rušení - Vliv rušení může mít za následek narušení UWB komunikace mezi zařízeními. Některé zprávy se tak v čipech vůbec nedetekují nebo se zahodí, protože přišly poškozené. Tím přijdeme o informaci potřebnou k lokalizaci a může dojít i k tomu, že celý jeden *blink* cyklus nebude pro lokalizaci použit z důvodu nedostatečného počtu měření (a to v lepším případě, v tom horším je sice dostatek měření k lokalizaci, ale vypočtená poloha je nejednoznačná). Takové momenty označujeme jako výpadek polohy. Rušení může být například elektromagnetické (cizí vysílání, interference zpráv) nebo také vlivem prostředí (kovové konstrukce v okolí zařízení).

Vliv chyb na lokalizaci dokáže z velké části potlačit filtrace odhadu polohy. To ovšem platí za předpokladu, že výskyt chyb není častý. Mezi nejjednodušší filtry patří klouzavý průměr, který počítá s několika posledními měřeními a ta mezi sebou zprůměruje. Tímto dokáže potlačit

vliv *outlierů* na lokalizaci a dokáže také potlačit šum způsobený konečnou přesností lokalizace a tím průběh polohy vyhladit. S čím se ale nevypořádá, jsou výpadky polohy. V takových chvílích jsou pak i filtrované polohy horší než nefiltrované a filtru trvá několik iterací než opět zkonverguje a začne vypočtené polohy filtrací zpřesňovat.

Filtrovat polohu můžeme také pomocí Kalmanova filtru, který jsme použili i pro synchronizaci hodin kotev. Filtr v sobě udržuje model pohybu a také znalost o varianci lokalizace a díky tomu dokáže potlačit *outliery* i šum lokalizace. Navíc dokáže snadno překonat i chvíle, kdy dojde k výpadku polohy.

Pro implementaci filtru polohy jsme ze zmíněných možností zvolili Kalmanův filtr a to pro jeho rychlost konvergence ke správným odhadům, jednoduchost implementace a výše zmíněným vlastnostem.

5.1 Implementace

Nyní odvodíme podobu matic pro KF polohy. K odvození použijeme znalostí a postupů popsaných v sekci 4.2.2.

Začněme s definicí stavového vektoru $\boldsymbol{x}$. Od filtru požadujeme, aby zpřesňoval polohu, která je ve formě kartézských souřadnic ve třech dimenzích. Proto bude stavový vektor $\boldsymbol{x}$ obsahovat složky polohy x, y a z. Navíc do stavového vektoru přidáme i rychlosti tagu podél souřadných os. Tím filtr „informujeme“, že filtrujeme polohu pohyblivého objektu a ne stacionárního.

$$\boldsymbol{x} = \begin{pmatrix} x \\ y \\ z \\ \dot{x} \\ \dot{y} \\ \dot{z} \end{pmatrix} \tag{5.1}$$

Pro jednoduchost implementace budeme předpokládat lineární model pohybu

$$\begin{aligned} x(t) &= \int_{t_0}^{t} \dot{x}\mathrm{d}t' + x(t_0) = \dot{x} \cdot (t - t_0) + x(t_0) \ , \\ y(t) &= \int_{t_0}^{t} \dot{y}\mathrm{d}t' + y(t_0) = \dot{y} \cdot (t - t_0) + y(t_0) \ , \\ z(t) &= \int_{t_0}^{t} \dot{z}\mathrm{d}t' + z(t_0) = \dot{z} \cdot (t - t_0) + z(t_0) \ , \end{aligned} \tag{5.2}$$

Pohyb však obecně není lineární a tak toto zjednodušení vnáší do procesu filtrace nepřesnosti. To znamená, že v případě, kdy bude tag vysílat *blink* zprávy s dlouhou prodlevou a uživatel se bude pohybovat po nelineární trase, tak přestane lineární model platit. Příkladem může být uživatelův pohyb po sinusové dráze, kdy tag vysílá *blink* zprávu se stejnou periodou s jakou uživatel projde jednu periodu sinusové dráhy. Pak by filtrované polohy tvořily přímku procházející spočtenými polohami. Kdyby však tag vysílal zprávy rychleji, pak by na kratších úsecích lineární model lépe aproximoval pohyb uživatele a vyfiltrované polohy by tak byly přesnější.

Z tohoto zjednodušení dále vyplývá, že filtr nebude přesně schopen sledovat rychlé změny směru pohybu uživatele. Toto téma bude blíže diskutováno v sekci 5.2 s výsledky filtrace.

Ze sady rovnic pro lineární pohyb 5.2 získáme po jejich diskretizaci sadu následujících rovnic

$$\begin{aligned} x\left[k+1\right] &= x\left[k\right] + \Delta t\left[k\right]\dot{x}\left[k\right] , \\ y\left[k+1\right] &= y\left[k\right] + \Delta t\left[k\right]\dot{y}\left[k\right] , \\ z\left[k+1\right] &= z\left[k\right] + \Delta t\left[k\right]\dot{z}\left[k\right] , \end{aligned} \tag{5.3}$$

ze kterých získáme matici modelu $\mathbf{F}$

$$\mathbf{F}\left[k\right] = \begin{bmatrix} 1 & 0 & 0 & \Delta t\left[k\right] & 0 & 0 \\ 0 & 1 & 0 & 0 & \Delta t\left[k\right] & 0 \\ 0 & 0 & 1 & 0 & 0 & \Delta t\left[k\right] \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} . \tag{5.4}$$

Nyní vysvětlíme, jakým způsobem se KF dokáže vypořádat s výpadkem polohy. V případě, že k výpadku dojde kvůli nedostatečnému množství měření, tak máme stále informaci o tom, kdy došlo k pokusu lokalizace tagu. Z toho dokážeme získat Δt a tak můžeme provést predikční krok KF a tím iteraci ukončit (nemáme k dispozici měření, které je potřebné v korekční části). Toto je vhodné, vypadne-li poloha ve chvíli, kdy uživatel s tagem drží stejný směr a rychlost pohybu a jedná se jen o dočasný výpadek. V takovém případě odhadované polohy do jisté míry odpovídají těm skutečným. Pochopitelně, pokud jedna z uvedených podmínek není splněna, pak odhadované polohy neodpovídají skutečnosti.

Dále budeme pokračovat vyjádřením převodní matice $\mathbf{H}$. Měřením je přímo vypočtená poloha tagu a při uvážení rovnice 4.14 a stavového vektoru $\boldsymbol{x}$ má matice jednoduchý tvar.

$$\mathbf{H} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix} . \tag{5.5}$$

Nakonec zbývá definovat matice pro kovarianci měření $\mathbf{R}$ a pro kovarianci procesu $\mathbf{Q}$. Kovarianci měření, v tomto případě vypočtených poloh, získáme snadno jako „mezi produkt“ při výpočtu polohy. Jako matici $\mathbf{R}$ použijeme kovarianci vypočtené polohy L-M metodou, kterou vypočteme jako inverzi matice $\mathbf{J}^{\mathrm{T}}\mathbf{W}\mathbf{J}$ [15]

$$\mathbf{R} = \left(\mathbf{J}^{\mathrm{T}}\mathbf{W}\mathbf{J}\right)^{-1} . \tag{5.6}$$

Matici $\mathbf{Q}$ nastavíme podle očekávané dynamiky lokalizovaného objektu. Typicky ji tak bude nutno měnit v závislosti na aplikaci. Například pokud bychom lokalizovali člověka, který se bude pohybovat chůzí, tak budou hodnoty matice $\mathbf{Q}$ menší, protože model bude při nízkých rychlostech více odpovídat skutečnosti. Pokud bychom lokalizovali vozidlo, tak budou hodnoty matice $\mathbf{Q}$ celkově větší, než u lokalizace pro člověka. Hodnoty matice však musí být dostatečně velké, aby zachytily nepřesnosti vnesené linearizací, šumem a rychlými změnami pohybu.

Co je však v tomto filtru jiné oproti filtru ze sekce 4.2.2, je způsob, jakým se vypočítává kovariance stavu $\mathbf{P}$. Rovnice 4.18 je totiž zjednodušené vyjádření *aposteriori* kovariance v pro optimální hodnotu Kalmanova zesílení $\mathbf{K}$, tedy když se $\mathbf{K}$ podle rovnice 4.16 rovná $\mathbf{P}^{-}\mathbf{H}^{\mathrm{T}}\mathbf{S}^{-1}$. Pak rovnice pro *a posteriori* kovarianci nabývá zjednodušeného tvaru 4.18. Pokud však $\mathbf{K}$ nenabývá své optimální hodnoty (vlivem například numerické nepřesnosti či šumu měření, který není gaussovský), je bezpečnější vypočítávat $\mathbf{P}$ níže uvedeným způsobem

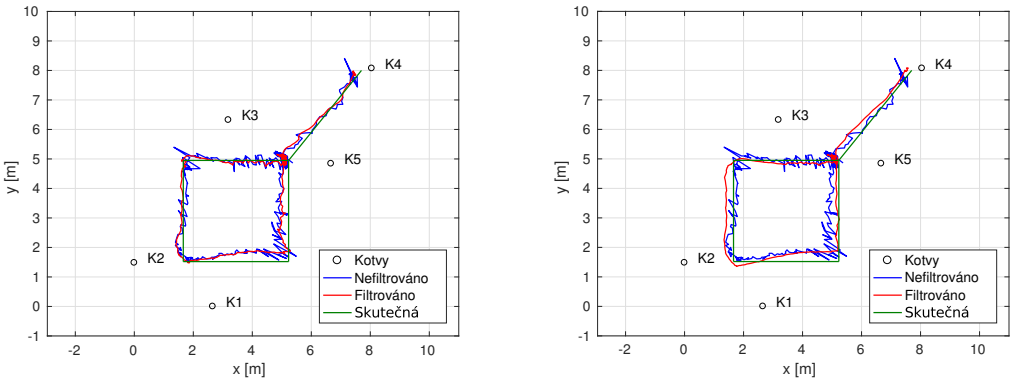
$$\mathbf{P} = \left(\mathbf{I} - \mathbf{K}\mathbf{H}\right)\mathbf{P}^{-}\left(\mathbf{I} - \mathbf{K}\mathbf{H}\right)^{\mathrm{T}} + \mathbf{K}\mathbf{R}\mathbf{K}^{\mathrm{T}} . \tag{5.7}$$

Toto nám zároveň umožňuje do nejistoty stavu dále zahrnout i nejistotu měření, ze kterého byl vypočten.

5.2 Ukázka funkce filtru

Pro ukázku funkce filtru jsme vybrali měření z měřícího polygonu firmy RCD. Stejné měření jsme použili i v sekci 4.3.1 zabývající se výpočtem polohy. Filtr jsme implementovali ve formě skriptu v programu MATLAB a dále ukážeme porovnání dvou verzí filtru, které se liší v matici $\mathbf{Q}$. První filtr má matici $\mathbf{Q}$ rovnu matici $\mathbf{Q}_0$, jejíž hodnoty jsou vyladěné pro požadovaný chod filtrace a kterou považujeme za výslednou. Druhý filtr pak má matici rovnu $\mathbf{Q} = \frac{1}{3}\mathbf{Q}_0$, což v porovnání znamená, že tento filtr bude více věřit predikovaným stavům z lineárního modelu a než první filtr.

V použitém měření se lokalizovalo za pomocí pěti kotev a tag zde byl připevněn na pohyblivý vozík ve výšce jednoho metru. Počáteční pozice uživatele s tagem byla na souřadnicích [4,5 m, 5 m] a jeho trajektorie byla ve tvaru obdélníku v protisměru hodinových ručiček. Po opsání úplného obdélníku se uživatel s tagem přemístil do rohu místnosti (na obrázcích 5.1 pravý horní roh), kde bylo měření ukončeno. Kvalitu filtrace a rozdíly filtrů budeme prezentovat pomocí porovnání vypočtené, filtrované a skutečné polohy a potom pomocí průběhů jednotlivých složek polohy. Bohužel je toto posouzení velice subjektivní, protože nemáme k dispozici časový průběh skutečné polohy, ale jen seznam bodů, kterými měl uživatel s tagem během měření projít.



(a) $\mathbf{Q} = \mathbf{Q}_0$ (b) $\mathbf{Q} = \frac{1}{3}\mathbf{Q}_0$

Obrázek 5.1: Porovnání vypočtené a filtrované polohy

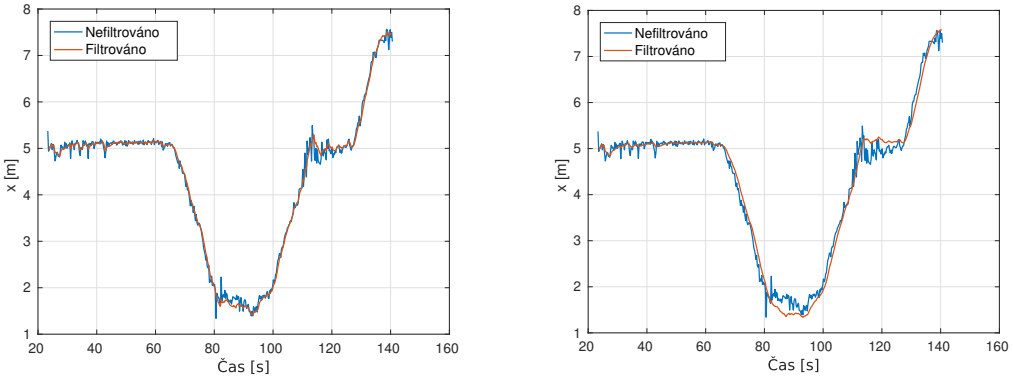
Na obrázku 5.1 vidíme porovnání filtrovaných a surových poloh. Na pravém obrázku 5.1b je vidět, že filtrovaná poloha nesedí na měřené poloze, ale spíš od ní „utíká“. To je více zřejmé v rozích obdélníku, kde filtr dle modelu předpokládá, že uživatel bude pokračovat ve stejném směru. Proto je vidět, že na chvíli filtrovaná poloha vybočí ze směru naměřené a chvíli trvá, než zase začne sledovat měření. Na rozdíl od toho první filtr sleduje měřenou polohu bez většího odbočení.

Je také zřejmé, že vypočtené polohy se v některých místech liší od skutečných offsetem (konec měření při přesunu do horního pravého rohu), či jiným tvarem trajektorie tagu (pohyb po spodní straně obdélníku). Zde může být na vině rušivý vliv prostředí (odrazy signálu,

zastínění kotvy, ...) a nebo také vybočení s vozíkem ze stanovené trasy (pak by však vyznačená „skutečná“ poloha nebyla pravdivá).

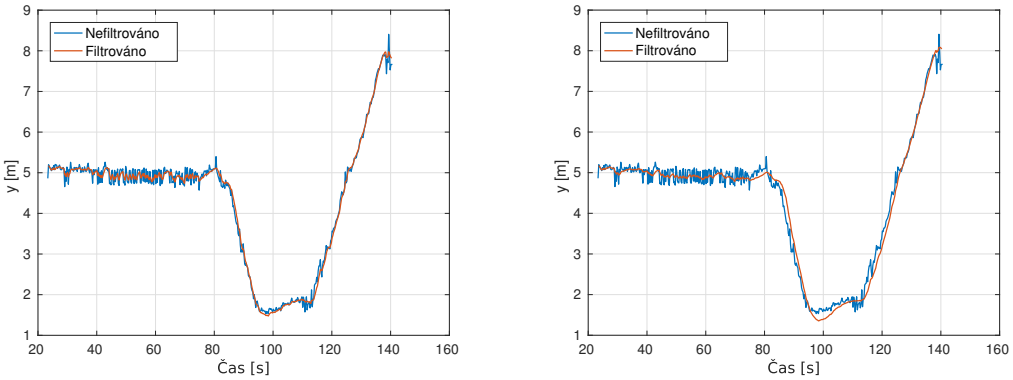
Pro větší názornost uvádíme níže dvě sady grafů. Každá sada odpovídá průběhu souřadnice x resp. y. Na průbězích je zřetelnější rozdíl filtrů a je i lépe vidět, kdy druhý filtr nesleduje rychlou změnu směru pohybu a dál pokračuje ve starém směru.

Sledování měřené polohy však není vše co od filtru požadujeme. Požadovanou vlastností filtru je také zmírnění rozptylu (nejen) stacionární polohy, který je způsoben přesností systému. Na obrázcích 5.1, 5.2 a 5.3 v časovém úseku $<23, 60>$ s je vidět, že polohu resp. jednotlivé složky polohy zbavuje šumu a filtrovaná poloha resp. jednotlivé složky jsou přibližně rovny střední hodnotě měřené polohy.



(a) $\mathbf{Q} = \mathbf{Q}_0$ (b) $\mathbf{Q} = \frac{1}{3}\mathbf{Q}_0$

Obrázek 5.2: Porovnání průběhu x-ové souřadnice



(a) $\mathbf{Q} = \mathbf{Q}_0$ (b) $\mathbf{Q} = \frac{1}{3}\mathbf{Q}_0$

Obrázek 5.3: Porovnání průběhu y-ové souřadnice

6 Fúze dat z UWB a GPS systému

Procesu datové fúze bylo v posledních letech věnováno mnoho pozornosti v širokém spektru oblastí. V principu datová fúze, nebo také senzorová fúze, shromažďuje a kombinuje data z více zdrojů s cílem získat přesnější anebo úplnější pohled na měřený proces. Hall a Llinas [18] definují datovou fúzi jako „*techniku kombinování dat z více senzorů a souvisejících informací z přidružených databází za účelem získání vyšší přesnosti a přesnějších závěrů, než jakých by bylo možno dosáhnout se samotným senzorem*“.

Senzorová fúze nám umožňuje získat přesnější měření za použití více senzorů a například tak zmenšit celkovou nejistotu měření. Zároveň nám datová fúze umožňuje efektivně kombinovat výhody různých senzorů a potlačit tak jejich nevýhody.

Příkladem budiž určování orientace v prostoru s využitím gyroskopu a akcelerometru. Gyroskop je senzor umožňující měřit úhlovou rychlost okolo os souřadného systému senzoru. Tedy prostou časovou integrací úhlových rychlostí lze získat orientaci senzoru. Ačkoliv je gyroskop vcelku přesný senzor, jeho velikou nevýhodou je, že (zvláště ty levné) trpí driftem vypočtených úhlů, který je třeba korigovat. Akcelerometr nám umožňuje měřit zrychlení, které působí na senzor podél os jeho souřadného systému. Pokud je senzor v klidu, dokážeme se znalostí velikosti místního gravitačního zrychlení vypočítat natočení senzoru [19]. Pokud se ale se senzorem začne hýbat, tak je určení orientace složité, protože nelze rozlišit mezi gravitačním zrychlením a zrychlením způsobeným pohybem. Akcelerometr má však oproti gyroskopu výhodu v tom, že netrpí driftem. Pokud tedy použijeme datovou fúzi na sloučení měřené orientace gyroskopem a akcelerometrem, získáme tím docela přesné určení orientace tělesa, na kterém jsou senzory umístěny, a také tím díky akcelerometru potlačíme negativní vliv driftu gyroskopu na určení orientace.

Nicméně krom zjevných výhod má datová fúze své nevýhody a určité limitace. Jednou z takových nevýhod je, že pokud spojujeme data z několika senzorů, které nefungují správně, nebudou pak správná ani data z nich a ani datová fúze nedokáže data vylepšit [20].

Pokud se vrátíme k naší problematice lokalizace pomocí UWB, tak můžeme zpřesnit data získaná z UWB systému pomocí dat získaných z GPS systému. Spojením dosáhneme i schopnosti lokalizovat tag v místech, kde je dostupná lokalizace pouze pomocí jednoho z jmenovaných systémů.

Nyní krátce ke způsobům senzorové fúze. Zdroj [21] nabízí jako možné způsoby implementace fúze rodinu metod známou jako Inferenční metody. Jednou z nich je klasická inference, jejímž základem je teorie pravděpodobnosti a která pracuje na základě testování hypotéz. Konkrétně vypočítává pravděpodobnost, s jakou by byla skutečně naměřená data naměřena, pokud by byla testovaná hypotéza pravdivá. Poddruhem klasické inference je také Bayesovská inference, která s využitím Bayesova vzorce pro podmíněnou pravděpodobnost bere v úvahu i *a prori* informaci.

Zmíněný zdroj popisuje i použití Kalmanova filtru. V případě použití KF pro senzorovou fúzi jsou dle zdroje rozeznávány dva případy. Jeden, kdy nejsme schopni měřit stavové veličiny přímo a měření je nutno na stavy přepočítat. V takovém typu fúze je pak matice přechodu stavů $\mathbf{F}$ rovna jednotkové matici a všechny systémové vstupy jsou ignorovány ($\mathbf{B} = \mathbf{0}$ nebo

$\boldsymbol{u} = \mathbf{0}$). V druhém případě máme možnost měřit přímo stavové veličiny, potom je převodní matice $\mathbf{H}$ rovna jednotkové matici nebo se jejím prostřednictvím měřená data lineárně zkombinují.

Z vyjmenovaných metod půjdeme opět cestou Kalmanova filtru, neboť stačí několik úprav již hotového filtru polohy, abychom jeho možnosti rozšířili o schopnost fúze odhadnutých poloh. Další výhodou je, že dokáže filtrovat, i když jeden zdroj dat vypadne. Čímž tedy přejde do „běžného módu“, kdy filtr nefúzuje data a v úvahu bude brát pouze ten zdroj měření, od kterého jsou data k dispozici.

V následujících sekcích budeme tedy diskutovat implementaci fúze pomocí KF a na konci sekce posoudíme výkonnost filtru na základě reálného měření.

6.1 Fúze dat Kalmanovým filtrem

Jak už bylo řečeno výše, Kalmanův filtr lze použít i pro úlohu fúze dat z více zdrojů. K tomu využijeme už implementovaný filtr polohy, který stačí rozšířit o další měření (rozšířit vektor měření).

Zdroji dat pro fúzi jsou myšleny systémy UWB a GPS. Před dalším postupem je ale nutné chvíli diskutovat o datech, která nám každý z těchto zdrojů poskytuje. Sice nám každý dává údaj o poloze stejného zařízení, ale problémem je, že polohy jsou v různých souřadných systémech.

Systém UWB poskytuje polohu v ENU (obecněji v lokálním) kartézském souřadném systému, neboť je snadné se v něm orientovat a polohy kotev se v něm dají rychle a pohodlně určit. Systém GPS pak poskytuje polohu v LLH geodetickém souřadném systému.

Je zřejmé, že změřené polohy je pro účely fúze nutné mezi jednotlivými souřadnými systémy převádět. V následující sekci popíšeme oba souřadné systémy a jak mezi nimi převést polohy.

6.1.1 Souřadné systémy a převody mezi nimi

Nejdříve si pár větami popišme každý souřadný systém, který budeme v této kapitole potřebovat. Jmenovitě to jsou systémy ENU (*East, North, Up*), ECEF (*Earth-Centered, Earth-Fixed*) a LLH (*Latitude, Longitude, Height*).

Tvar, který má naše planeta, je velice nepravidelný a je pro účely polohování vhodnější aproximovat zemské těleso mnohem jednodušším tvarem. Tím je elipsoid a pro popis polohy libovolného bodu na, vně či uvnitř elipsoidu byla vytvořena kartézská souřadná soustava ECEF. Jak název napovídá, její počátek je totožný s hmotným středem Země a také se soustava oproti Zemi nepohybuje (odtud *fixed*). Orientace soustavy je taková, že její xy rovina je shodná s rovinou rovníku, osa x je kolmá na nultý poledník, osa z prochází osou rotace planety směrem ke geografickému severnímu pólu a osa y je kolmá na obě dvě osy, tak aby dohromady tvořily pravotočivou soustavu.

Elipsoid, který aproximuje tvar Země, je nazýván referenčním a jeho střed je taktéž umístěn do hmotného středu Země. Také je orientován tak, aby jeho hlavní poloosa a byla rovnoběžná s osou souřadného systému x a vedlejší poloosa b měla shodný směr s osou z. V tabulce níže jsou uvedeny hlavní parametry referenčního elipsoidu WGS 84 [1, 22].

Parametr	
Hlavní poloosa a	6 378,137 km
Vedlejší poloosa b	6 356,7523142 km
Excentricita e	$\sqrt{0{,}00669437999014}$
Druhá excentricita e'	$\sqrt{0{,}00673949674228}$
Zploštění f	$(299{,}257223563)^{-1}$

Tabulka 6.1: Parametry referenčního elipsoidu

Excentricity e, e' a zploštění f můžeme také vypočítat za pomocí vzorců [22]

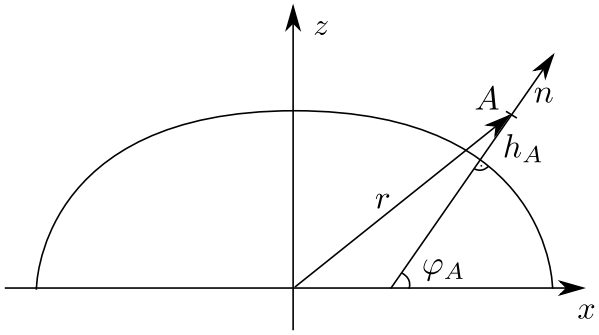
$$e = \sqrt{1 - \left(\frac{b}{a}\right)^2}, \tag{6.1}$$

$$e' = \sqrt{\left(\frac{a}{b}\right)^2 - 1} = \frac{a}{b}e, \tag{6.2}$$

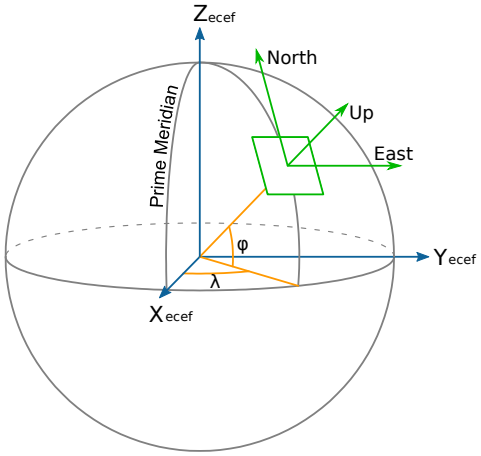
$$f = 1 - \frac{b}{a}. \tag{6.3}$$

Důvod, proč ECEF vůbec zmiňujeme, je prostý, jednak definuje základy pro souřadnou soustavu LLH a jednak nám umožňuje převádět polohy mezi LLH a ENU. Navíc přechod ECEF, ENU není náročný, jak ukážeme dále v této sekci.

Jako další soustavu popíšeme geodetický souřadný systém LLH (dále také označovaný jako *Latitude, Longitude, Altitude* (LLA) nebo *World Geodetic System 1984* (WGS 84)) [1]. Podobně jako ECEF má svůj počátek shodný s hmotným středem Země a vůči Zemi se nepohybuje. Poloha v LLH se sestává ze dvou úhlů λ a φ v radiánech a výšky h v kilometrech, kde úhel λ je zeměpisná šířka a φ je zeměpisná výška. Pro jejich definici uvažujme bod A, na který ukazuje jeho polohový vektor v soustavě ECEF $\boldsymbol{r}_{\text{ECEF}}$ a mající polohu v LLH $[\lambda_A, \varphi_A, h_A]$. Úhel λ_A je roven úhlu, který svírá $\boldsymbol{r}$, promítnutý do xy roviny, s osou x systému ECEF. Dále vytvořme normálu z bodu A kolmou k povrchu elipsoidu $\boldsymbol{n}$. Úhel φ_A je potom úhel, který svírá normála $\boldsymbol{n}$ a rovina xy. Nakonec výška h je vzdálenost bodu A od povrchu elipsoidu.



Obrázek 6.1: Souřadný systém LLH



Obrázek 6.2: Souřadné systémy ECEF, LLH a ENU[2]

Jako poslední popíšeme kartézský souřadný systém se zkratkou ENU. Jeho počátek je umístěn do zvoleného referenčního bodu a systém je orientován tak, aby jeho $x$ová osa mířila na východ (*East*), osa y na sever (*North*). Osa z je pak kolmá na obě osy a orientovaná tak, aby systém byl pravotočivý. Tedy směrem „nahoru“ (*Up*). Tento systém je velice vhodný pro lokalizaci v méně rozsáhlých prostorách, neboť rovina xy souřadného sytému tvoří tečnou rovinu k zemskému povrchu v počátku souřadného systému (referenčním bodě) a je tak snadné se v ENU orientovat. Na obrázku 6.2 můžeme vidět zjednodušené zakreslení zmíněných souřadných systémů

Nyní se všemi soustavami popsanými uvedeme, jak mezi nimi převádět.

Začneme tím jednodušším případem, kterým je převod mezi ENU a ECEF. Díky tomu, že jsou oba dva souřadné systémy kartézské, sestává se převod z pouhého posunutí počátku soustavy a následné rotace systému. Ovšem tato rotace je závislá na zeměpisné šířce λ a zeměpisné výšce φ, aby soustava ENU byla správně orientována. Následující rovnice popisuje převod polohového vektoru $\boldsymbol{r}$ z ECEF do ENU.

$$\boldsymbol{r}_{\mathrm{ENU}} = \mathbf{R}_{\mathrm{ENU}} \left(\boldsymbol{r}_{\mathrm{ECEF}} - \boldsymbol{O}_{\mathrm{ECEF}} \right) , \tag{6.4}$$

kde $\boldsymbol{O}_{\mathrm{ECEF}}$ je poloha počátku ENU v ECEF a $\mathbf{R}_{\mathrm{ENU}}$ je rotační matice, která má tvar [22]

$$\mathbf{R}_{\mathrm{ENU}} = \begin{bmatrix} -\sin\lambda & \cos\lambda & 0 \\ -\sin\varphi\cos\lambda & -\sin\varphi\sin\lambda & \cos\varphi \\ \cos\varphi\cos\lambda & \cos\varphi\sin\lambda & \sin\varphi \end{bmatrix} . \tag{6.5}$$

Dobrá vlastnost $\mathbf{R}_{\mathrm{ENU}}$ je, že bývá časově invariantní a tak i převod mezi ENU a ECEF zůstává lineární. K rovnici pro opačný převod se pak dostaneme snadno

$$\boldsymbol{r}_{\mathrm{ECEF}} = \mathbf{R}_{\mathrm{ENU}}^{-1} \boldsymbol{r}_{\mathrm{ENU}} + \boldsymbol{O}_{\mathrm{ECEF}} , \tag{6.6}$$

kde inverze matice $\mathbf{R}_{\mathrm{ENU}}$ je rovna $\mathbf{R}_{\mathrm{ENU}}^{-1} = \mathbf{R}_{\mathrm{ENU}}^{\mathrm{T}}$.

[2] http://www.dirsig.org/docs/new/coordinates.html

Dále ukážeme převod mezi LLH a ECEF. Začněme převodem z LLH do ECEF, který má tvar [1]

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} \frac{a\cos\lambda}{\sqrt{1+(1-e^2)\tan^2\varphi}} + h\cos\lambda\cos\varphi \\ \frac{a\sin\lambda}{\sqrt{1+(1-e^2)\tan^2\varphi}} + h\sin\lambda\cos\varphi \\ \frac{a\left(1-e^2\right)\sin\varphi}{\sqrt{1-e^2\sin^2\varphi}} + h\sin\varphi \end{pmatrix}, \tag{6.7}$$

kde a je délka hlavní poloosy a e je excentricita referenčního elipsoidu.

Převod z ECEF do LLH je díky nelineárním funkcím v rovnicích 6.7 podstatně složitější, protože neexistuje analytický předpis pro zeměpisnou výšku φ a výšku h. Pro určení φ a h vznikl iterativní Bowringův algoritmus [1], který je zde označen jako algoritmus 1. Zeměpisnou šířku λ převedeme takto [1]

$$\lambda = \begin{cases} \arctan\frac{y}{x} & , x \geq 0 \\ \arctan\frac{y}{x} + \pi & , x < 0 \wedge y \geq 0 \\ \arctan\frac{y}{x} - \pi & , x < 0 \wedge y < 0 \end{cases}. \tag{6.8}$$

V rovnici 6.8 záporné hodnoty λ odpovídají západním zeměpisným šířkám.

Algoritmus 1: Bowringova iterativní metoda určení φ a h

$p = \sqrt{x^2 + y^2}$ `/* Inicializace */`
$\tan u = \frac{z}{p}\frac{a}{b}$
while $\tan u$ *konverguje* **do**
  $\cos^2 u = \frac{1}{1+\tan^2 u}$
  $\sin^2 u = 1 - \cos^2 u$
  $\tan\varphi = \frac{z+e'^2 b\sin^3 u}{p-e^2 a\cos^3 u}$
  $\tan u = \frac{b}{a}\tan\varphi$
end
$\varphi = \arctan(\tan\varphi)$
$N = \frac{a}{\sqrt{1-e^2\sin^2\varphi}}$
$h = \begin{cases} \frac{p}{\cos\varphi} - N & , \varphi \neq \pm\frac{\pi}{2} \\ \frac{z}{\sin\varphi} - N - e^2 N & , \varphi \neq 0 \end{cases}$

6.1.2 Rozšíření filtru polohy

V této sekci popíšeme úpravy filtru polohy z předchozí kapitoly, které jsou nutné, aby zároveň dokázal filtrovat i fúzovat polohy ze systému UWB a GPS.

Stavový vektor $\boldsymbol{x}$ ponecháme beze změny. Filtr bude tedy i nadále filtrovat polohu v kartézských souřadnicích. Proto je nutné převést polohu z GPS, která je v LLH, do souřadného systému, který používá UWB. Výstupní polohu filtru pak můžeme dle potřeby nechat v kartézských souřadnicích nebo ji převést zpět do LLH.

Z toho tedy vyplývá, že jediné co je potřeba změnit jsou matice, které vystupují v rovnicích 4.14 (resp. 4.4) a 4.15. Tedy matice $\mathbf{H}$ a $\mathbf{R}$.

Přijímáme data o poloze ze dvou zdrojů, kde každý poskytuje polohu ve třech rozměrech. To znamená, že vektor měření $\boldsymbol{z}$ bude mít tvar

$$\boldsymbol{z} = \begin{pmatrix} \begin{pmatrix} x_{\mathrm{UWB}} & y_{\mathrm{UWB}} & z_{\mathrm{UWB}} \end{pmatrix}^{\mathrm{T}} \\ \begin{pmatrix} x_{\mathrm{GPS}} & y_{\mathrm{GPS}} & z_{\mathrm{GPS}} \end{pmatrix}^{\mathrm{T}} \end{pmatrix} \tag{6.9}$$

a že matice $\mathbf{H}$ bude následující

$$\mathbf{H} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}. \tag{6.10}$$

Také matice kovariancí měření se rozšíří

$$\mathbf{R} = \begin{bmatrix} \mathbf{R}_{\mathrm{UWB}} & \mathbf{0} \\ \mathbf{0} & \mathbf{R}_{\mathrm{GPS}} \end{bmatrix}, \tag{6.11}$$

kde $\mathbf{R}_{\mathrm{UWB}}$ je stejná matice jako v rovnici 5.6 a matici $\mathbf{R}_{\mathrm{GPS}}$ nám buď dodá systém GPS společně s každým odhadem polohy nebo ji budeme muset dopočítávat za chodu, neboť přesnost polohy bude závislá na poloze (DOP) a počtu satelitů použitých k lokalizaci.

6.2 Měření na Letenské pláni

V této části popíšeme závěrečné měření, kterým jsme si kladli za cíl vyhodnotit činnost datové fúze a jakých výsledků dosáhneme, pokud se například příjem GPS signálu zhorší (nebo vypadne).

Závěrečné měření jsme se tak rozhodli udělat na Letenské pláni, protože tam je k dispozici dostatek prostoru i pro rozsáhlejší měření a také je zde výborná viditelnost na družice GPS. Zároveň zde máme možnost dostat se i na místa, kde je špatná přímá viditelnost na družice GPS.

Za objekt, který jsme lokalizovali, jsme vybrali dron Bebop 2 od společnosti Parrot, který obsahuje GPS modul. Abychom dron mohli lokalizovat také pomocí UWB sítě, přidělali jsme k němu tag z našeho UWB systému.

Dále jsme vytvořili měřící UWB polygon sestávající se ze šesti kotev (5 slave a 1 master), které jsme umístili tak, aby nebyly všechny v jedné rovině a tím minimalizovali DOP ve všech třech osách. Dále jsme zaměřili kotvy K5 a K6. Kotvu K6 jsme použili jako referenční bod pro převod mezi soustavami LLH a ENU a zaměření K5 nám umožňuje přesnější převod mezi ENU a lokální souřadnou soustavou. Jako lokální souřadnou soustavu (dále jen LOC) jsme zvolili kartézský pravotočivý souřadný systém s počátkem shodným s počátkem referenčního ENU systému. Od referenčního ENU se systém LOC liší pouze pootočením v ose z o úhel α. Tato rotace je popsána maticí $\mathbf{R}_{\mathrm{LOC}}$ a převod mezi ENU a LOC popisuje rovnice 6.13.

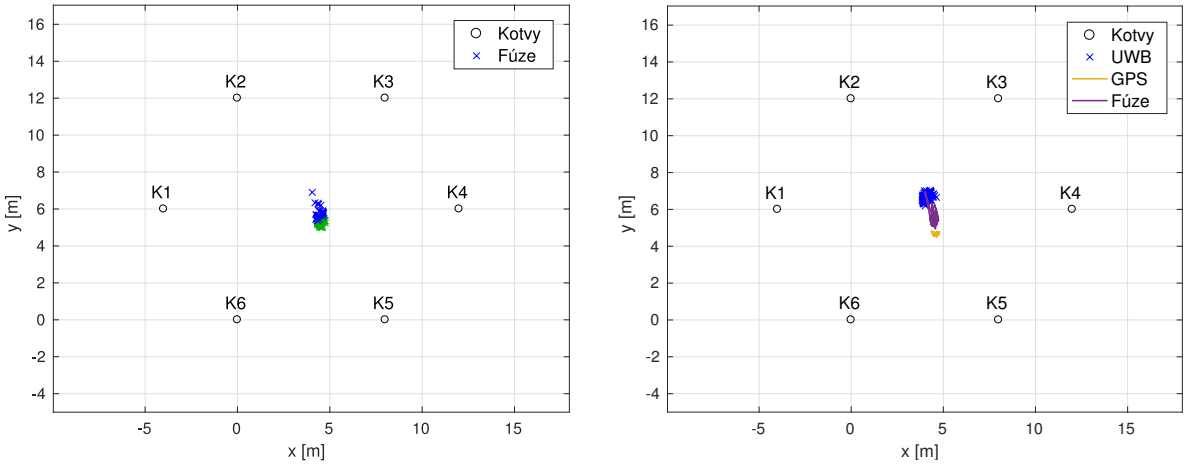
$$\mathbf{R}_{\mathrm{LOC}} = \begin{bmatrix} \cos\alpha & \sin\alpha & 0 \\ -\sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{6.12}$$

$$\boldsymbol{r}_{\mathrm{LOC}} = \mathbf{R}_{\mathrm{LOC}}\boldsymbol{r}_{\mathrm{ENU}} \tag{6.13}$$

Důvodem pro zavedení LOC je snazší zaměřování kotev vůči sobě a snazší orientace v prostoru polygonu.

Po připravení měřícího polygonu jsme provedli dvě měření. V prvním měření bylo s dronem pohybováno pouze v z-ové ose, tím zhodnotíme přesnost lokalizace UWB systému a jak se tato přesnost bude měnit v závislosti na výšce dronu. Během druhého měření jsme s dronem provedli průlet konstelací rychlostí do 6 $\mathrm{m{\cdot}s^{-1}}$ ve výšce přibližně 0,5 m. Začátek i konec trasy dronu byl tedy vně měřícího UWB polygonu. Takto jsme otestovali především schopnost lokalizace UWB mimo polygon.

Protože jsme měření provedli na místě, kde GPS přijímač neměl problémy s přímou viditelností a kde tedy byl signál silný, jsou polohovací data získaná z GPS velice přesná a můžeme je tedy považovat za skutečné polohy. Proto budeme výpadky a znehodnocení polohovacích dat simulovat v MATLAB skriptu prostým nepoužitím dat v daném časovém úseku a přidáním náhodného bílého šumu s rozptylem $(15\ \mathrm{cm})^2$ do získaných pozic. Odhadnuté polohy z UWB a polohy získané fúzí tak budeme porovnávat proti čistým GPS polohám. Pro větší přehlednost nebudeme do grafů zahrnovat polohy GPS systému, které jsme zatížili šumem.



(a) Poloha získaná datovou fúzí, červená - pouze GPS, modrá - pouze UWB, zelená - oba systémy

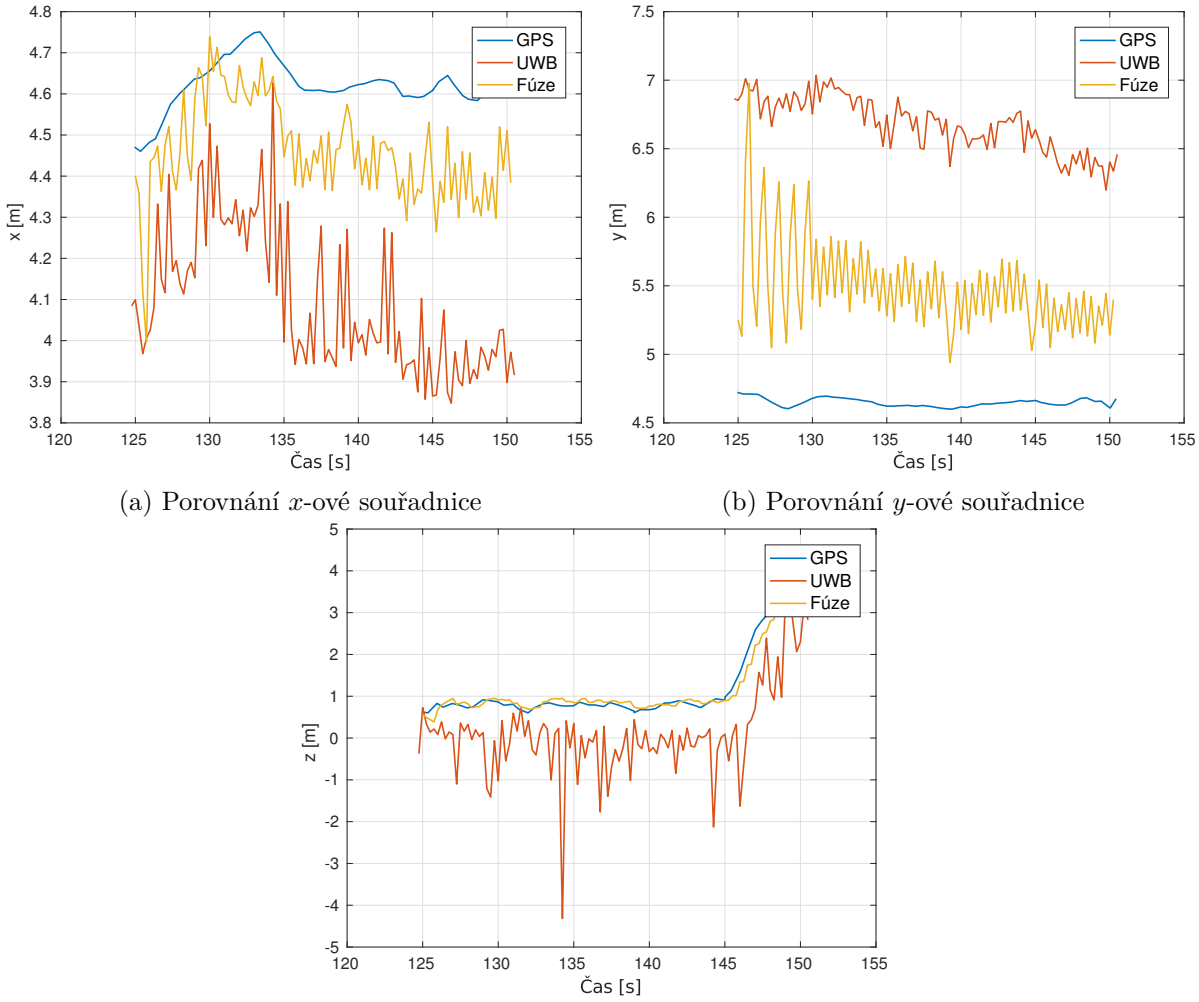
(b) Porovnání všech naměřených a vypočtených poloh

Obrázek 6.3: Statické měření

Na obrázku 6.3b můžeme vidět polohy získané z obou systémů společně s polohou získanou datovou fúzí. Hned z prvního pohledu je zřejmý offset UWB poloh oproti polohám z GPS. Příčin offsetu může být hned několik. Například špatné rozestavení kotev (tj. jiné, než se kterým počítáme), nesprávně změřená vzdálenost slave kotev od master kotvy nebo nesprávné zaměření referenční kotvy. Nejvíce pravděpodobná je však první jmenovaná příčina, neboť jsme UWB polygon rozestavěli s nedostatečně přesnými nástroji (použití pásma, které se prohýbá a natahuje, místo laserového měřiče vzdálenosti) a v časové tísni.

Pomineme-li však offset polohy, tak odhadnutá poloha tagu vychází skutečně „staticky“ (jistý rozptyl je stále přítomen) i pro UWB systém. Dále je zřejmé, že fúze dat z obou systémů pomohla offsetem zatíženým UWB polohám přiblížit se k těm skutečným a také potlačila jejich rozptyl. Samotný výsledek fúze můžeme vidět na obrázku 6.3a, kde je zároveň znázorněno,

jaká data byla použita pro výpočet dané polohy (tedy jestli poloha byla dostupná pouze z jednoho systému a jednalo se o filtr polohy - červená pro GPS a modrá pro UWB - nebo jestli byly dostupné oba systémy a data z nich se fúzovala - zelená).



(a) Porovnání x-ové souřadnice

(b) Porovnání y-ové souřadnice

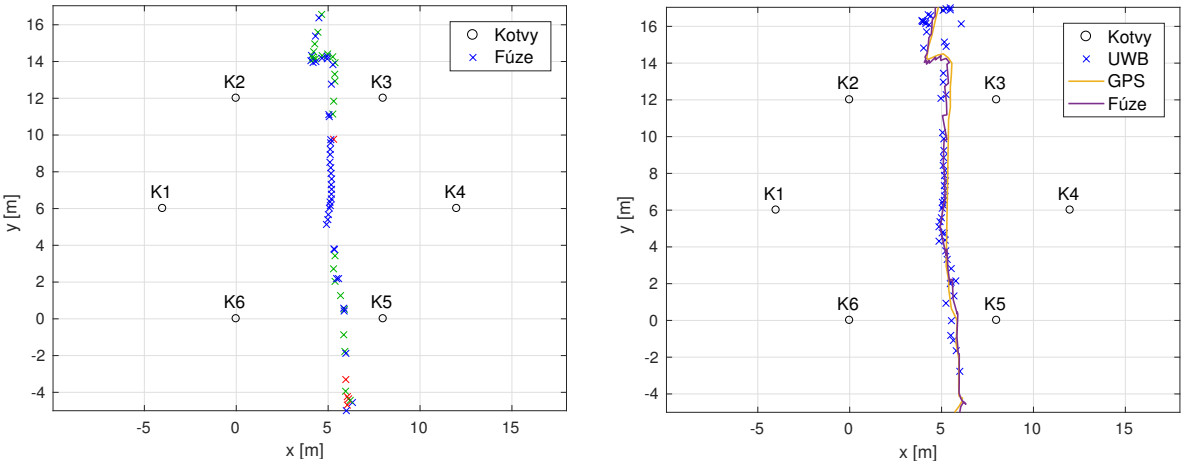
(c) Porovnání z-ové souřadnice

Obrázek 6.4: Porovnání souřadnic statického měření

Podrobnější náhled do výsledků fúze získáme z obrázků 6.4 výše. Zde je více patrné, jak jsme pomocí fúze zlepšili celkové výsledky a potlačili šum v datech. Poloha získaná fúzí je sice stále zatížená šumem, ale jeho rozptyl je nižší než ve vstupních datech. Konkrétně pro x-ovou souřadnici na obrázku 6.4a se podařilo rozptyl z UWB výrazně snížit. Dále pro y-ovou souřadnici na obrázku 6.4b se rozptyl pohybuje okolo $(20\ \text{cm})^2$.

Z obrázků výše také můžeme vypozorovat offset pro každou souřadnici. Popořadě pro x-ovou, y-ovou a z-ovou souřadnici jsou hodnoty offsetů přibližně 45 cm, 2 m a 1 m. To poukazuje na to, že jsme při rozmisťování kotev chybovali především v y-ové ose.

Přejděme nyní k následujícímu měření, které je tentokrát dynamické. Stejně jako v předchozím měření jsme i v tomto polohy z GPS zatížili bílým šumem s rozptylem $(15\ \text{cm})^2$. Navíc jsme zde simulovali výpadek GPS systému v době, kdy dron prolétal polygonem.



(a) Poloha získaná datovou fúzí, červená - pouze GPS, modrá - pouze UWB, zelená - oba systémy

(b) Porovnání všech naměřených a vypočtených poloh

Obrázek 6.5: Průlet dronu skrz polygon

Z obrázků 6.5 je vidět, že zde si UWB systém vedl mnohem lépe než v předchozím měření. Úsek, během kterého jsme simulovali výpadek GPS poloh, lze vidět na obrázku 6.5a jako řadu modrých (tedy jen z UWB systému) „fúzovaných“ poloh.

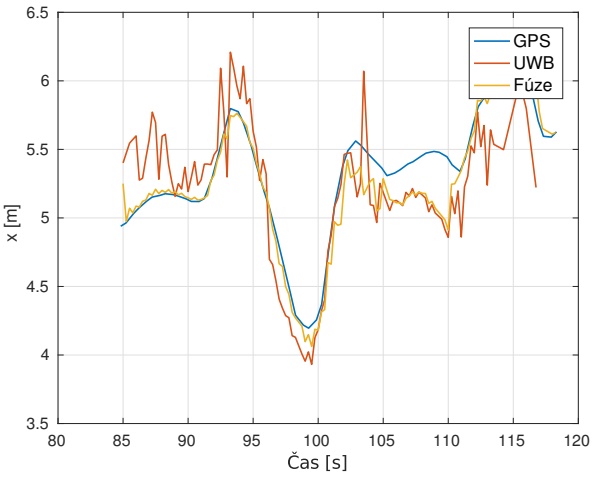
Dále můžeme vidět, že jsme byli schopni dron lokalizovat pomocí UWB, i když vyletěl mimo polygon, a co je důležité, je to, že horizontální souřadnice jsou i navzdory vyššímu DOP docela přesné.

Obrázky 6.6 ukazují, že offsety zůstávají stejné. To potvrzuje, že se jedná o systematickou chybu typu chybného zaměření kotvy či její nepřesné umístění.

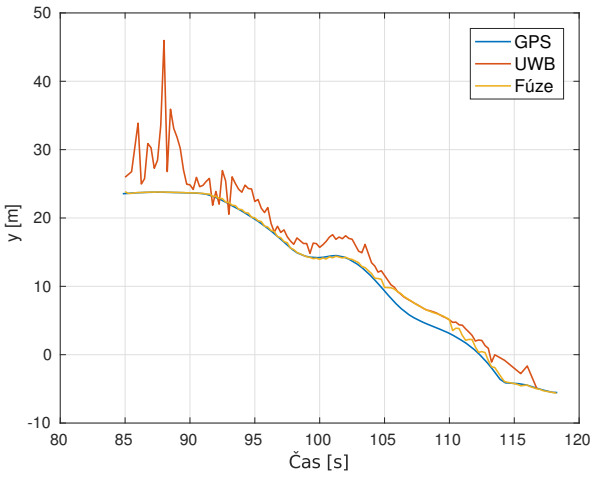
Značně horších výsledků zde na začátku měření dosahuje UWB lokalizace v z-ové ose. Důvodem je, že se v tu dobu tag nacházel mimo konstelaci, kde lokalizování ve třetí ose vycházelo nejednoznačně zejména díky vyššímu DOP, na které, ve spojení s umístěním kotev do špatné výšky, je lokalizace ve třetí ose citlivější. V momentě, kdy se dron dostal opět dovnitř polygonu, se chybovost snížila (společně s rozptylem, který je i tak stále velký). Opětovné zvýšení chybovosti můžeme pozorovat na konci měření, kdy dron vyletěl z polygonu (po čase 112 s). Nicméně polohy získané fúzí mají výšku opravenu pomocí dat z GPS.

Z naměřených a vypočtených dat jsme zjistili, že data UWB systému jsou zatížena systematickou chybou, která je nejspíše způsobena špatným rozmístěním kotev. Pravděpodobně ze stejného důvodu je ve vypočtených pozicích znatelný větší rozptyl, než jaký jsme zaznamenali v měření z předchozí kapitoly 5.

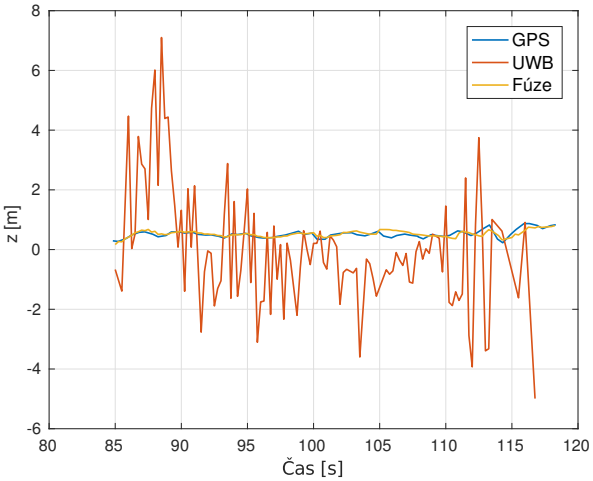
Šum a chyby v datech jsme však pomocí fúze dat z obou systémů byli schopni opravit a dosáhli jsme lepších výsledků, než kdybychom použili pouze polohy z UWB.



(a) Porovnání x-ové souřadnice



(b) Porovnání y-ové souřadnice



(c) Porovnání z-ové souřadnice

Obrázek 6.6: Porovnání jednotlivých souřadnic pro průlet dronu

7 Závěr

Cílem této práce bylo navrhnout a otestovat lokalizační systém postavený na technologii UWB a lokalizační metodě TDoA. Z toho vznikl koncept systému, který se skládá z pevně umístěných kotev, uživatelských mobilních tagů a výpočetního centra, kde pro UWB lokalizaci se používá čipu DW1000.

První překážkou tak bylo implementovat algoritmus, který umožní bezdrátově synchronizovat vnitřní hodiny kotev. Při hledání vhodného řešení jsme našli možnost synchronizovat pomocí PI regulátoru, či pomocí prosté lineární interpolace. Z nalezených metod dosahoval nejlepších výsledků Kalmanův filtr, který jsme pro naše potřeby upravili tak, aby byl schopen potlačit nedokonalosti, které mají oscilátory použitých čipů DW1000.

Dalším krokem byla implementace metody, se kterou odhadneme polohy uživatelských zařízení. Zvolena byla Levenbergova-Marquardtova metoda, která vyniká rychlou konvergencí a nízkým rizikem divergence. Pro větší přesnost odhadů byla použita vážená verze L-M metody, kde jsme k výpočtu vah použili variance odhadů offsetů hodin kotev a variance časů přijetí zprávy od tagu.

S fungujícím lokalizačním systémem jsme pokračovali v pokusu zlepšit přesnost lokalizace cestou filtru polohy. I pro tento účel jsme využili Kalmanova filtru, se kterým jsme skutečně v kapitole 5 dosáhli přesnějších výsledků a zároveň potlačili přítomný šum v datech i vliv „outlierů“.

Nakonec jsme upravili filtr polohy tak, aby jako vstupní data dokázal přijímat polohy jak ze systému UWB, tak i z GPS, a tím prováděl datovou fúzi. K tomu jsme provedli měření na Letné a zhodnotili výsledky měření.

Touto prací jsme tedy ověřili, že požadovaný systém lze vytvořit a že podle provedených měření lze lokalizovat tag s velikou přesností. Zároveň jsme díky experimentům zjistili informace o robustnosti schopnosti systému lokalizovat. Jak vyplývá z měření na Letné, je stále možné sledovat pohyb tagu, i když není zaměření a umístění kotev přesné. To je však za cenu snížení přesnosti a zanesení chyb do lokalizace, jako je například polohový offset či zvýšená chybovost v některých částech lokalizačního polygonu. Pro zlepšení přesnosti UWB lokalizace je tedy samozřejmě vhodné umístit kotvy tak, aby jejich polohy odpovídaly těm, které uvažuje výpočetní centrum.

Oblastí dalšího zájmu tedy bude zvýšení počtu uživatelů a zároveň s tím vylepšení a zrychlení algoritmů, aby byly stále provozuschopné. Dále provedeme další měření, které se bude zaměřovat na datovou fúzi, neboť provedené měření jsme provedli na místě, kde je lokalizace pomocí GPS velice přesná, a tak by bylo na místě fúzovat data i na místech, kde je GPS méně přesná.

Seznam zkratek

AoA	Angle of Arrival; Metoda určení polohy podle úhlu přijatého signálu.	*str. 7*
ASK	Amplitude-Shift Keying; Klíčování posuvem amplitudy.	*str. 4*
BPM	Burst Postion Modulation; Modulace pozicí impulzu.	*str. 4–6, 45*
BPSK	Binary Phase-Shift Keying; Binární klíčování fázovým posuvem.	*str. 4–6, 45*
DOP	Dilution of Precision; činitel udávající zhoršení přesnosti odhadu polohy vlivem geometrie konstelace.	*str. 11, 26, 36, 39, 45*
ECEF	Earth-Centered, Earth-Fixed; Kartézská pravotočivá souřadná soustava se středem shodným se středem Země, soustava se vůči Zemi nepohybuje.	*str. 32–35, 45*
EKF	Extended Kalman Filter; Rozšířený Kalmanův filtr.	*str. 14*
ENU	East-North-Up; Lokální souřadnicová soustava.	*str. 32–34, 36, 45*
FSK	Frequency-Shift Keying; Klíčování změnou frekvence.	*str. 4*
GNSS	Global Navigation Satellite System; Globální satelitní navigační systém.	*str. 3*
GPS	Global Positioning System; Globální navigační systém vyvinutý armádou USA.	*str. 3, 4, 31, 32, 35–39, 41, 45, 46*
HRP	High Rate Pulse repetition frequency; Vysoká opakovací frekvence pulzů.	*str. 5, 6*
KF	Kalman Filter; Kalmanův filtr.	*str. 17–21, 27, 28, 31, 32, 45*
LLA	Latitude, Longitude, Altitude; viz. LLH.	*str. 33*
LLH	Latitude, Longitude, Height; Globální geodetická souřadná soustava Země popisující polohu zeměpisnou výškou, zeměpisnou šířkou, výškou nad referenčním elipsoidem.	*str. 32–36, 45*
LOC	Lokální kartézská souřadná soustava.	*str. 36, 37*
LRP	Low Rate Pulse repetition frequency; Nízká opakovací frekvence pulzů.	*str. 5*
MAC	Media Access Control; Linková vrstva.	*str. 4*
MATLAB	MATrix LABoratory; Software pro maticový počet.	*str. 29, 37*
PHY	Physical; Fyzická/ý.	*str. 4, 5*
PI	Proportional-Integral controller; Regulátor složený z proporcionální a integrační části.	*str. 13, 41*

PPDU Physical Protocol Data Unit; Datová jednotka fyzické vrstvy. *str. 5*

PRF Pulse Repetition Frequency; Opakovací frekvence pulzů. *str. 5*

QAM Quadrature Amplitude Modulation; Kvadraturní Amplitudová Modulace. *str. 4*

RMS Root Mean Square; Kvadratický průměr. *str. 21, 22*

RSS Received Signal Strength; Síla přijatého signálu. *str. 7, 8, 11*

SFD Start-of-frame Delimiter; Oddělení začátku rámce. *str. 5*

SHR Synchronization Header; Synchronizační hlavička datového rámce. *str. 5*

TDoA Time Difference of Arrival; metoda určení polohy z rozdílů časů přijetí signálu, hyperbolická navigace. *str. 3, 4, 7–9, 11–13, 22–26, 41, 45*

ToA Time of Arrival; metoda určení polohy z času přijetí signálu. *str. 3, 7–9, 11*

TWR Two-Way Ranging; metoda měření vzdálenosti pomocí zařízení s asynchronními hodinami. *str. 8–11, 45*

UKF Unscented Kalman Filter; „Nezředěný“ Kalmanův filtr. *str. 14*

UWB Ultra-Wide Band; Standard širokopásmové komunikace umožňující přesná časová měření. *str. 3–7, 9, 11, 12, 22, 26, 31, 32, 35–39, 41, 45, 46*

WGS 84 World Geogetic System; Standard z roku 1984 definující globální souřadné soustavy ECEF a LLH a referenční elipsoid. *str. 32, 33*

Vysvětlení vybraných pojmů

Blink Krátká UWB zpráva, která je používána k určení polohy. *str. 10, 12, 13, 18, 22, 27*

FEL ČVUT Fakulta Elektrotechnická České Vysoké Učení Technické. *str. 12*

Kotva Nepohyblivý prvek UWB sítě, infrastruktura. *str. 4, 7–13, 17–22, 24–26, 29, 30, 32, 36–39, 41*

L-M Levenbergova-Marquardtova metoda - Metoda pro řešení nelineárních nejmenších čtverců, která rychle konverguje a má nízké riziko divergence.. *str. 24, 25, 28, 41, 45*

RCD RCD Radiokomunikace; společnost zabývající se radiotechnikou se sídlem ve Starém Hradišti. *str. 12, 19, 29*

Tag Pohyblivý prvek UWB sítě, uživatelské zařízení. *str. 7–13, 18, 22, 23, 25–29, 31, 36, 37, 39, 41*

Seznam obrázků

2.1 UWB impulz 4
2.2 Rozdělení časového rámce BPM-BPSK modulace [3] 5
2.3 Struktura fyzického rámce 5
3.1 Angle of Arrival se skupinou antén 7
3.2 Trilaterace 8
3.3 Double Sided TWR 9
3.4 Time Difference of Arrival, oblast okolo hyperbol značí nejistotu měření 11
a Lokalizace v místě s nízkým DOP 11
b Lokalizace v místě s vysokým DOP 11
4.1 Odchylka odhadnutého offsetu hodin a jeho skutečné hodnoty 19
a KF 2 stavy 19
b KF 3 stavy 19
4.2 Distribuční funkce odchylky odhadnutého a skutečného offsetu hodin 20
a KF 2 stavy 20
b KF 3 stavy 20
4.3 Odhad chyby frekvence Δf 21
a KF 2 stavy 21
b KF 3 stavy 21
4.4 Odchylka skutečné a odhadnuté chyby frekvence $\varepsilon_{\Delta f}$ 21
a KF 2 stavy 21
b KF 3 stavy 21
4.5 Výpočet polohy L-M metodou 25
a Vypočtené polohy 25
b Naměřená TDoA 25
5.1 Porovnání vypočtené a filtrované polohy 29
a $\mathbf{Q} = \mathbf{Q}_0$ 29
b $\mathbf{Q} = \frac{1}{3}\mathbf{Q}_0$ 29
5.2 Porovnání průběhu x-ové souřadnice 30
a $\mathbf{Q} = \mathbf{Q}_0$ 30
b $\mathbf{Q} = \frac{1}{3}\mathbf{Q}_0$ 30
5.3 Porovnání průběhu y-ové souřadnice 30
a $\mathbf{Q} = \mathbf{Q}_0$ 30
b $\mathbf{Q} = \frac{1}{3}\mathbf{Q}_0$ 30
6.1 Souřadný systém LLH 33
6.2 Souřadné systémy ECEF, LLH a ENU 34
6.3 Statické měření 37
a Poloha získaná datovou fúzí, červená - pouze GPS, modrá - pouze UWB, zelená - oba systémy 37
b Porovnání všech naměřených a vypočtených poloh 37
6.4 Porovnání souřadnic statického měření 38
a Porovnání x-ové souřadnice 38

b Porovnání y-ové souřadnice 38
c Porovnání z-ové souřadnice 38
6.5 Průlet dronu skrz polygon 39
a Poloha získaná datovou fúzí, červená - pouze GPS, modrá - pouze UWB, zelená - oba systémy 39
b Porovnání všech naměřených a vypočtených poloh 39
6.6 Porovnání jednotlivých souřadnic pro průlet dronu 40
a Porovnání x-ové souřadnice 40
b Porovnání y-ové souřadnice 40
c Porovnání z-ové souřadnice 40

Literatura

[1] Kaplan, Elliott D.; Hegarty, Christopher J.: *Understanding GPS: Principles and Applications*. Artech House mobile communications series, Artech House, druhé vydání, 2006, ISBN 978-1-58053-894-7.

[2] Sahinoglu, Zafer; Gezici, Sinan; Güvenc, Ismail: *Ultra-wideband Positioning Systems Theoretical Limits, Ranging Algorithms, and Protocols*. Cambridge University Press, 2008, ISBN 978-0-511-54105-6.

[3] IEEE Standard for Low-Rate Wireless Networks. *IEEE Std 802.15.4-2015 (Revision of IEEE Std 802.15.4-2011)*, Duben 2016, doi:10.1109/IEEESTD.2016.7460875.

[4] Decawave Ltd: UWB Worldwide Regulations. 2015, v. 1.0.
URL `<https://www.decawave.com/sites/default/files/apr001_uwb_worldwide_regulations_summary.pdf>`

[5] Decawave Ltd: DW1000 USER MANUAL. 2017, v. 2.12.
URL `<https://www.decawave.com/sites/default/files/resources/dw1000_user_manual_2.12.pdf>`

[6] Fiala, Zdeněk: UWB lokalizační systém. Interní zpráva č. TE01020186/CTU/2017/5, Centrum integrovaných družicových a pozemských navigačních technologií, 2017.

[7] Navrátil, Václav; Krška, Josef: Určování polohy UWB-TDOA: popis metod, výsledky a doporučení. Interní zpráva č. TE01020186/CTU/2017/8, Centrum integrovaných družicových a pozemských navigačních technologií, 2017.

[8] Neirynck, Dries; Luk, Eric; Mclaughlin, Michael: An Alternative Double-Sided Two-Way Ranging Method. V *2016 13th Workshop on Positioning, Navigation and Communications (WPNC)*, IEEE, Říjen 2016, doi:10.1109/WPNC.2016.7822844.

[9] McElroy, Ciaran; Neirynck, Dries; McLaughlin, Michael: Comparison of wireless clock synchronization algorithms for indoor location systems. V *2014 IEEE International Conference on Communications Workshops (ICC)*, IEEE, Červen 2014, ISBN 978-1-4799-4640-2, doi:10.1109/ICCW.2014.6881189.
URL `<http://ieeexplore.ieee.org/document/6881189/>`

[10] Welch, Greg; Bishop, Gary: An Introduction to the Kalman filter: SIGGRAPH 2001 Course 8. V *Computer Graphics*, Los Angeles, CA, USA (August 12-17): ACM Press, Addison-Wesley, Leden 2001.
URL `<https://sreal.ucf.edu/wp-content/uploads/2017/02/Welch2001.pdf>`

[11] Lindsey, Theodore S: *On the Kalman Filter and Its Variations*. Dizertační práce, University of Kansas, May 2014.
URL `<https://kuscholarworks.ku.edu/handle/1808/14535>`

[12] Odelson, Brian J.; Rajamani, Murali R.; Rawlings, James B.: A new autocovariance least-squares method for estimating noise covariances. *Automatica*, svazek 42, Únor 2006, ISSN 00051098, doi:10.1016/j.automatica.2005.09.006.

[13] Levenberg, Kenneth: A method for the solution of certain non-linear problems in least squares. *Quarterly of Applied Mathematics*, svazek 2, č. 2, 1944, ISSN 0033-569X, 1552-4485, doi:10.1090/qam/10666.

[14] Marquardt, Donald W.: An Algorithm for Least-Squares Estimation of Nonlinear Parameters. *Journal of the Society for Industrial and Applied Mathematics*, svazek 11, č. 2, Červen 1963, ISSN 0368-4245, doi:10.1137/0111030.

[15] Gavin, Henri P: The Levenberg-Marquardt method for nonlinear least squares curve-fitting problems. *Department of Civil and Environmental Engineering, Duke University*, Březen 2017.

[16] Dobeš, Josef; Žalud, Václav: *Moderní radiotechnika*. BEN - technická literatura, 2006, ISBN 80-7300-132-2.

[17] McElroy, C.; Neirynck, D.; McLaughlin, M.: Comparison of wireless clock synchronization algorithms for indoor location systems. V *2014 IEEE International Conference on Communications Workshops (ICC)*, Červen 2014, ISSN 2164-7038, doi:10.1109/ICCW.2014.6881189.

[18] Hall, D.L.; Llinas, J.: An introduction to multisensor data fusion. *Proceedings of the IEEE*, svazek 85, Leden 1997, ISSN 0018-9219, doi:10.1109/5.554205.

[19] Abyarjoo, Fatemeh; Barreto, Armando; Cofino, Jonathan; aj.: Implementing a sensor fusion algorithm for 3D orientation detection with inertial/magnetic sensors. V *Innovations and advances in computing, informatics, systems sciences, networking and engineering*, Springer, 2015.

[20] Fowler, C. A.: Comments on the Cost and Performance of Military Systems. *IEEE Transactions on Aerospace and Electronic Systems*, svazek AES-15, č. 1, Leden 1979, ISSN 0018-9251, doi:10.1109/TAES.1979.308790.

[21] Elmenreich, Wilfried: An Introduction to Sensor Fusion. Research Report 47/2001, Technische Universität Wien, Institut für Technische Informatik, Treitlstr. 1-3/182-1, 1040 Vienna, Austria, 2001.

[22] Navrátil, Václav: Určování polohy dálkoměrnou metodou. Algoritmy a jejich přesnost. Květen 2015.
URL `<https://dspace.cvut.cz/handle/10467/61736>`