České vysoké učení technické v Praze
Fakulta elektrotechnická

Fúze dat v demonstrátoru zpracování družicových signálů a 'opportunity' signálů

Bakalářská práce

Jan Povolný

Praha, květen 2018

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

Prohlášení

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

V Praze 24. května 2018

.....

Jan Povolný

Poděkování

Chtěl bych poděkovat svému vedoucímu, panu prof. Vejražkovi, a panu Ing. Navrátilovi za pomoc a cenné připomínky. Také bych chtěl poděkovat i svým spolužákům za důležité rady při psaní bakalářské práce.

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Povolný** Jméno: **Jan** Osobní číslo: **425548**

Fakulta/ústav: **Fakulta elektrotechnická**

Zadávající katedra/ústav: **Katedra mikroelektroniky**

Studijní program: **Komunikace, multimédia a elektronika**

Studijní obor: **Aplikovaná elektronika**

II. ÚDAJE K BAKALÁŘSKÉ PRÁCI

Název bakalářské práce:

Fúze dat v demonstrátoru zpracování družicových signálů a 'opportunity ' signálů

Název bakalářské práce anglicky:

Fusion of GNSS Signals with Opportunity Signals in the Demonstrator

Pokyny pro vypracování:

Zabývejte se možnostmi sloučení dat od několika družicových navigačních systémů a jejich podporou tzv.?opportunity? signály. Navrhněte vhodný algoritmus a simulujte jeho činnost v jazyce MatLab.
Písemná zpráva: do 50 stran.

Seznam doporučené literatury:

[1] Parkincon, Brad (editor): Global Positioning Systém: Theory and applications. Vol. II. Progress in Astronautics and Aeronautics. Washington, D.C., str. 290 ? 305.
[2] Chun Yang: Signals of Opportunity for Positioning. ION Cal Sec Meeting. Torrance, Sep, 2011.
[3] Grejner-Brzezinska, D.: Referát na IAIN World Meeting, Praha, OCT 2015.

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: **09.02.2017** Termín odevzdání bakalářské práce: **25.05.2018**

Platnost zadání bakalářské práce: **10.09.2018**

prof. Ing. František Vejražka, CSc. podpis vedoucí(ho) práce	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

Práce se zaměřuje na fúzi dat o poloze a pohybu z různých zdrojů. Jednak jde o fúzi více zdrojů měření polohy, např. z více GNSS systémů, nebo využití takzvaných „opportunity“ signálů, které nejsou primárně určeny pro měření polohy.

Dále je navržen model pohybu, který aproximuje pohyb sledovaného objektu, a Kalmanův filtr, který se snaží zmenšit chyby měření polohy pomocí predikce následujícího stavu s použitím daného modelu pohybu. Zde vstupují do procesu „opportunity“ signály ve formě časově proměnných veličin pohybu měřených pohybovými senzory.

Cílem je také určit polohu i v případě, že hlavní zdroj určování polohy, obvykle nějaký GNSS systém, má výpadek signálu nebo nějakou poruchu, a to právě za pomoci výše zmíněných metod.

Navržený algoritmus je simulován v prostředí MATLAB a ověřován na testovacích datech určených pro post-processing, zachycených při dřívějším experimentu s navigačním demonstrátorem. Vstupní data jsou polohy z přijímače GPS a naměřené veličiny z pohybových senzorů. Je zde měřeno zrychlení 3-osým akcelerometrem a úhlová rychlost 3-osým gyroskopem.

Abstract

This thesis is dealing with a fusion of position and motion data from different sources. Firstly, there is the fusion of different sources of the position measurement, e.g. different GNSS systems, or utilization of “signals of opportunity”, which are not designated for position measurement.

Secondly, a motion model was designed to approximate the motion of the navigated object and a Kalman filter for reducing the error of the position measurement by predicting the next state using the given motion model. Here the “signals of opportunity” are input into the process as time-varying motion quantities measured by the motion sensors.

Another goal is to determine the position even if the main positioning source, usually a GNSS system, has a signal error or other failure, using the above-mentioned methods.

The designed algorithm is simulated in the MATLAB environment and verified on a test dataset for post-processing that was measured in an earlier experiment with a navigation demonstrator. The input data are positions from a GPS receiver and the measured quantities from the motion sensors. Acceleration from a 3-axis accelerometer and angular velocity from a 3-axis gyroscope are measured.

Obsah

Prohlášení ..... - 2 -
Poděkování ..... - 3 -
Abstrakt ..... - 5 -
Abstract ..... - 5 -
Úvod ..... - 8 -
1 Popis pohybu ..... - 9 -
1.1 Rychlost ..... - 9 -
1.2 Poloha ..... - 9 -
1.3 Rotace ..... - 9 -
1.3.1 Základní teorie ..... - 9 -
1.3.2 Převod mezi tělesovou a pevnou soustavou: $\boldsymbol{x}' \rightarrow \boldsymbol{x}$ ..... - 12 -
1.3.3 Odhad matice přechodu z rychlosti ..... - 13 -
1.4 Zrychlení ..... - 15 -
2 Model pohybu ..... - 16 -
2.1 Predikce rychlosti ..... - 16 -
2.2 Predikce polohy ..... - 16 -
2.3 Rotace – aktualizace matice přechodu ..... - 16 -
2.4 Převod zrychlení z tělesové do pevné soustavy ..... - 17 -
3 Fúze dat ..... - 18 -
3.1 Fúze dat více zdrojů polohy ..... - 18 -
3.2 Kalmanův filtr ..... - 18 -
3.2.1 Základní definice ..... - 18 -
3.2.2 Predikce ..... - 20 -
3.2.3 Korekce – filtrace ..... - 20 -
3.2.4 Průběh algoritmu ..... - 21 -
3.3 Filtr polohy ..... - 22 -
3.3.1 Inicializace ..... - 22 -
3.3.2 Predikce ..... - 23 -
3.3.3 Korekce – filtrace ..... - 24 -
3.3.4 Korekce matice přechodu z rychlosti ..... - 25 -
3.3.5 Numerická korekce matice rotace ..... - 25 -
3.3.6 Kompenzace systematických chyb senzorů ..... - 26 -
4 Testování algoritmu ..... - 31 -
4.1 Vizualizace testovaných dat ..... - 33 -
4.2 Chyby v určení polohy přijímačem GPS ..... - 33 -
4.3 Výsledný odhad trasy ..... - 36 -

Závěr ..... - 40 -
Reference ..... - 41 -
Přílohy ..... - 42 -
A. Filtr polohy v MATLABu ..... - 42 -
B. Ostatní přílohy ..... - 45 -

Úvod

Poloha se dnes měří nejčastěji pomocí družicové navigace – nějakého ze systémů GNSS. Nejznámější a nejrozšířenější je dnes americký systém GPS, nicméně existují už i další funkční systémy, např. ruský GLONASS a jiné. Na měření polohy pomocí nějakého ze systémů GNSS dnes spoléhá velké množství dopravních systémů i různých služeb, a stává se tak prakticky nutnou součástí dnešní společnosti.

Každé měření je ovšem zatíženo chybami, které zde mohou znemožnit určení polohy s požadovanou přesností. Dalším problémem jsou výpadky družicového signálu, které po určitý časový interval znemožňují určení polohy úplně. Tyto problémy jsou v družicové navigaci poměrně časté, zvláště v nějakém náročnějším prostředí, a mohou být velkou překážkou pro aplikaci. V městském prostředí mohou být tyto problémy způsobeny například odrazem signálu od budov, což má za následek vícecestné šíření signálu. Dále může být příjem signálu ztížen zastíněním výhledu stromy nebo jinou vegetací.

Jedním z možných řešení těchto problémů je použití více zdrojů informace o poloze a pohybu v prostoru. Jednak to mohou být jiné družicové systémy, např. při použití více GNSS systémů paralelně, nebo např. signály, které nejsou určeny přímo pro určování polohy – takzvané „opportunity“ signály. Důležité je, aby jednotlivé signály na sobě byly navzájem nezávislé. To nám do systému přináší větší množství informace a jsme pak lépe schopni zmenšit chyby měření.

V našem případě je družicová navigace doplněna o senzory pohybu. Navržený model pohybu aproximuje pohyb měřeného objektu a Kalmanův filtr vytváří predikci následujícího stavu na základě tohoto modelu. Predikci následně porovnává s dalším měřením a vyhodnocuje nový odhad polohy včetně odhadu jeho chyby.

1 Popis pohybu

Popis pohybu je v čase spojitý. Při vytváření modelu pohybu v kapitole 2 je tento popis diskretizován v čase a funkce některých veličin jsou aproximovány pro numerické výpočty. Pohyb je popisován v pevné inerciální soustavě, zatímco zrychlení $\boldsymbol{a}$ objektu je většinou měřeno v rotující (neinerciální) soustavě tělesa. Před dosazením do následujících vztahů je proto potřeba souřadnice zrychlení nejprve převést do pevné soustavy podle kapitoly 1.3.2.

1.1 Rychlost

Rychlost objektu v čase $\boldsymbol{v}(t)$ určíme jako

$$\boldsymbol{v}(t) = \boldsymbol{v}(0) + \int_0^t \boldsymbol{a}(t)\, dt\,, \tag{1.1}$$

kde $\boldsymbol{a}(t)$ je jeho zrychlení v čase.

1.2 Poloha

Polohu objektu v čase popíšeme funkcí

$$\boldsymbol{r}(t) = \boldsymbol{r}(0) + \int_0^t \boldsymbol{v}(t) dt\,, \tag{1.2}$$

do které dosadíme (1.1) za rychlost a po úpravách dostaneme:

$$\boldsymbol{r}(0) + \int_0^t \left(\boldsymbol{v}(0) + \int_0^t \boldsymbol{a}(t)\, dt\right) dt =$$

$$\boldsymbol{r}(0) + \int_0^t \boldsymbol{v}(0)\, dt + \iint_0^t \boldsymbol{a}(t)\, dt^2 = \boldsymbol{r}(0) + \boldsymbol{v}(0) \cdot t + \iint_0^t \boldsymbol{a}(t)\, dt^2\,. \tag{1.3}$$

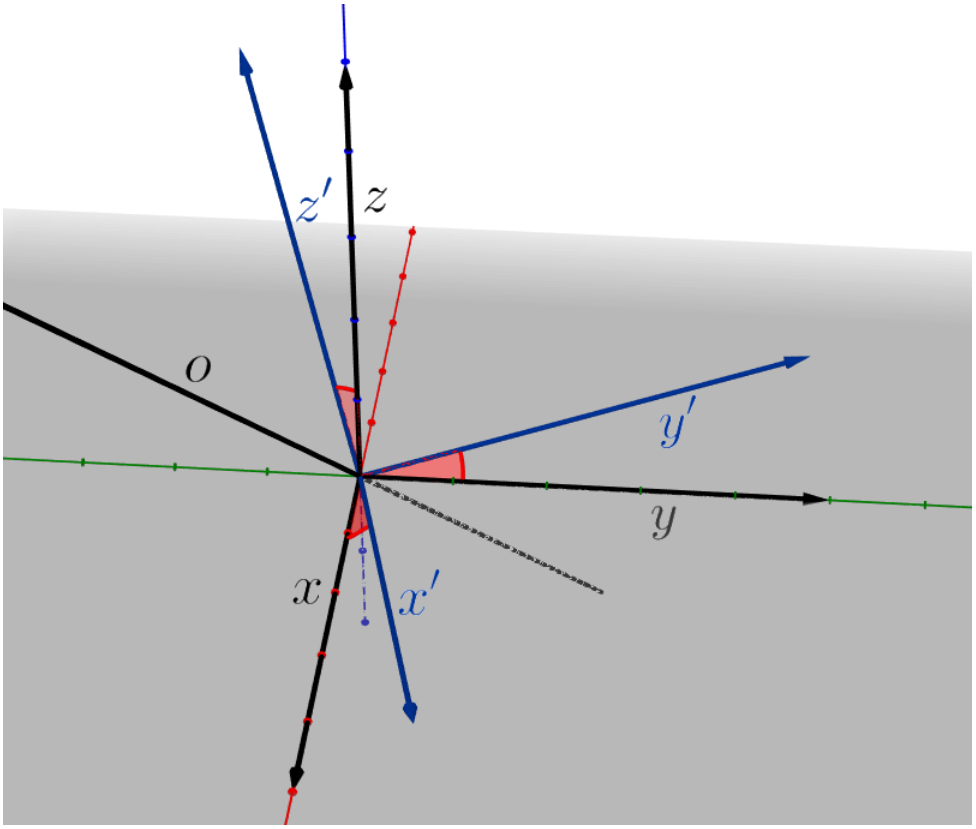
Funkce polohy je tedy tvaru

$$\boldsymbol{r}(t) = \boldsymbol{r}(0) + \boldsymbol{v}(0) \cdot t + \iint_0^t \boldsymbol{a}(t)\, dt^2\,. \tag{1.4}$$

1.3 Rotace

1.3.1 Základní teorie

Objekt sledujeme v nějaké pevné inerciální soustavě s bází $\mathcal{X}$, může být spojená například se středem Země (ECEF) nebo s nějakým vztažným bodem na povrchu Země (ENU, NED). S objektem je pak pevně spojena rotující (neinerciální) soustava tělesa s bází $\mathcal{X}'$. Když se pozorovaný objekt začne otáčet, ať už to bude zatáčení auta nebo jeho náklon kolem nějaké osy v prostoru, například když začne stoupat do kopce, jeho tělesová vztažná soustava se bude otáčet v prostoru spolu s ním vůči pevné soustavě $\mathcal{X}$. Vzhledem k tomu, že popis pohybu děláme v pevné soustavě s bází $\mathcal{X}$, ale zrychlení a úhlovou rychlost objektu měříme většinou akcelerometrem a gyroskopem v rotující soustavě tělesa $\mathcal{X}'$, budeme potřebovat vztah pro převod mezi těmito dvěma soustavami. Pro následující úvahy předpokládáme, že obě soustavy mají společný počátek souřadnic. Na obrázku 1.1 je znázorněna rotace soustavy tělesa $\mathcal{X}'$ vůči pevné inerciální soustavě $\mathcal{X}$, kolem osy o.



Obrázek 1.1: Rotace soustavy tělesa $\mathcal{X}'$ vůči pevné soustavě $\mathcal{X}$, kolem osy o.

Abychom tedy zjistily, jak se těleso natáčí a jaké je jeho zrychlení v pevné soustavě, musíme znát převod souřadnic z báze $\mathcal{X}'$ do $\mathcal{X}$. Pro to se hodí výpočet pomocí matice přechodu od $\mathcal{X}$ k $\mathcal{X}'$

$$\mathbb{A}_{\mathcal{X}}^{\mathcal{X}'} = ((\boldsymbol{x}_1')_{\mathcal{X}}\ (\boldsymbol{x}_2')_{\mathcal{X}}\ \dots), \tag{1.5}$$

kde $(\boldsymbol{x}_i')_{\mathcal{X}}$ jsou souřadnice i-tého bázového vektoru $\mathcal{X}'$ v bázi $\mathcal{X}$. Pomocí matice přechodu $\mathbb{A}_{\mathcal{X}}^{\mathcal{X}'}$ pak můžeme jednoduše převést souřadnice libovolného vektoru $\boldsymbol{x}$ ze soustavy tělesa do pevné soustavy:

$$(\boldsymbol{x})_{\mathcal{X}} = \mathbb{A}_{\mathcal{X}}^{\mathcal{X}'} \cdot (\boldsymbol{x})_{\mathcal{X}'}, \tag{1.6}$$

kde $(\boldsymbol{x})_{\mathcal{X}'}$ jsou souřadnice $\boldsymbol{x}$ v bázi $\mathcal{X}'$ a $(\boldsymbol{x})_{\mathcal{X}}$ souřadnice v bázi $\mathcal{X}$. Je dobré upozornit, že matice přechodu od $\mathcal{X}$ k $\mathcal{X}'$ převádí souřadnice vektoru přesně naopak, než bychom čekali z jejího názvu, tedy z báze $\mathcal{X}'$ do $\mathcal{X}$, což může být velice matoucí.

Nejvíce nás zajímá, jak se změní matice přechodu, když se tělesová soustava otočí kolem nějaké osy o nějaký úhel, abychom potom opět byli schopni převést souřadnice vektorů z $\mathcal{X}'$ do $\mathcal{X}$. V další části budeme uvažovat prostor dimenze 3 a báze $\mathcal{X} = (\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z})$, $\mathcal{X}' = (\boldsymbol{x}', \boldsymbol{y}', \boldsymbol{z}')$. Nejdříve musíme popsat vektor úhlové rychlosti. Správně to je pseudovektor, ale to na situaci nic nemění. Znamená to pouze, že není invariantní např. vůči zrcadlení. Ovšem je invariantní vůči rotaci kolem osy, a především to nás zajímá. Prakticky to znamená, že při rotaci kolem osy dané např. maticí R, se bude transformovat podle vztahu

$$\boldsymbol{\omega}' = R \cdot \boldsymbol{\omega}, \tag{1.7}$$

zatímco např. při zrcadlení daném maticí Z by to bylo

$$\boldsymbol{\omega}' = -Z \cdot \boldsymbol{\omega}. \tag{1.8}$$

Vektor úhlové rychlosti je tedy

$$\boldsymbol{\omega}(t) = \begin{pmatrix} \omega_x(t) \\ \omega_y(t) \\ \omega_z(t) \end{pmatrix}. \tag{1.9}$$

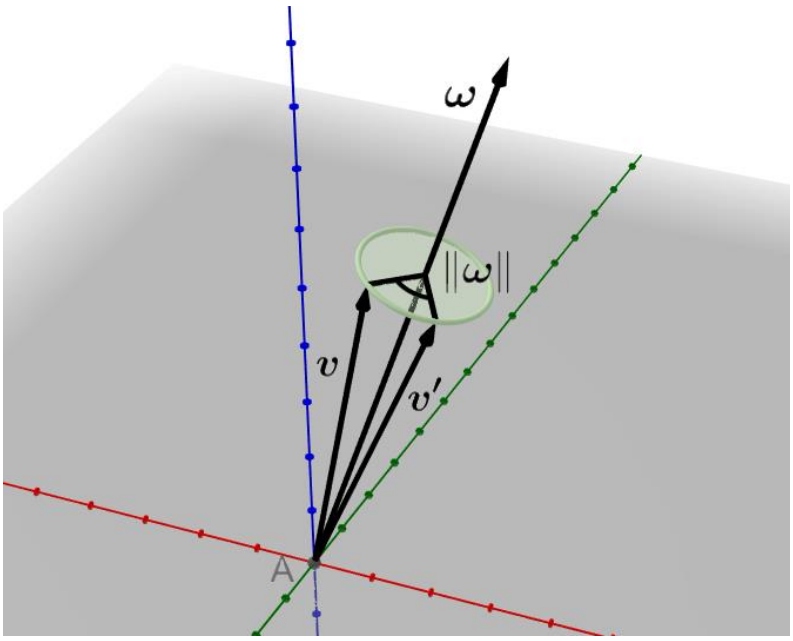
Tento vektor nám určuje oba parametry rotace, jednak osu rotace, a jednak velikost úhlové rychlosti kolem této osy. Vektor osy rotace je přímo $\boldsymbol{\omega}$, budeme ho ovšem používat normovaný – jednotkový, osa rotace pak bude

$$\boldsymbol{o}(t) = \frac{\boldsymbol{\omega}(t)}{\|\boldsymbol{\omega}(t)\|} \tag{1.10}$$

a velikost úhlové rychlosti kolem této osy

$$\omega(t) = \|\boldsymbol{\omega}(t)\|. \tag{1.11}$$

Význam vektoru úhlové rychlosti je znázorněn na obrázku 1.2, kde se vektor $\boldsymbol{v}$ otáčí s úhlovou rychlostí $\boldsymbol{\omega}$. Jestliže bude $\boldsymbol{\omega} = \overrightarrow{\boldsymbol{konst}}$., tak se za 1 sekundu $\boldsymbol{v}$ otočí kolem osy dané úhlovou rychlostí $\boldsymbol{\omega}$ o úhel $\|\boldsymbol{\omega}\|$.



Obrázek 1.2: Rotace vektoru $\boldsymbol{v}$ daná úhlovou rychlostí $\boldsymbol{\omega}$.

Jestliže bude mít vektor úhlové rychlosti na nějakém časovém intervalu konstantní směr $\Rightarrow \boldsymbol{o}(t) = \overrightarrow{\boldsymbol{konst}}$., pak můžeme pro úhel otočení psát

$$\varphi(t) = \int_0^t \omega(t)\, dt = \int_0^t \|\boldsymbol{\omega}(t)\| dt\,. \tag{1.12}$$

Dále budeme potřebovat otočit vektor kolem nějaké osy o určitý úhel. Pro to se hodí Rodriguesova věta, kterou použijeme v maticové formě. Chceme-li otočit vektor $\boldsymbol{a}$ o úhel φ (v kladném směru) kolem osy $\boldsymbol{o}$, použijeme matici [1]:

$$\mathbb{R}_{\varphi,\boldsymbol{o}} = \begin{pmatrix} c + o_x^2(1-c) & o_x o_y(1-c) - o_z s & o_x o_z(1-c) + o_y s \\ o_y o_x(1-c) + o_z s & c + o_y^2(1-c) & o_y o_z(1-c) - o_x s \\ o_z o_x(1-c) - o_y s & o_z o_y(1-c) + o_x s & c + o_z^2(1-c) \end{pmatrix}, (1.13)$$

kde $c = \cos\varphi$, $s = \sin\varphi$ a $\boldsymbol{o} = \begin{pmatrix} o_x \\ o_y \\ o_z \end{pmatrix}$. Kladný směr otáčení poznáme tak, že když se díváme proti vektoru osy $\boldsymbol{o}$ (ukazuje šipkou na nás), tak kladný směr bude proti směru hodinových ručiček. Nový otočený vektor $\boldsymbol{a}'$ tedy bude

$$\boldsymbol{a}' = \mathbb{R}_{\varphi,\boldsymbol{o}} \cdot \boldsymbol{a}. \quad (1.14)$$

1.3.2 Převod mezi tělesovou a pevnou soustavou: $\mathcal{X}' \to \mathcal{X}$

Řekněme, že na začátku bude tělesová soustava shodná s pevnou ($\mathcal{X}' = \mathcal{X}$). Když má objekt úhlovou rychlost $\boldsymbol{\omega}'(t)$ v tělesové soustavě (v tento moment i v pevné) a konstantní osu rotace v čase (směr $\boldsymbol{\omega}'(t)$), tak můžeme podle (1.12) spočítat úhel rotace za nějaký časový úsek $\varphi(t)$. Tělesová soustava se nám tedy za tento časový úsek otočí v pevné soustavě o úhel $\varphi(t)$. Jestliže chceme převést souřadnice nějakého vektoru z pevné soustavy do otočené o úhel α, musíme ho otočit o opačný úhel ⇒ dosadit při výpočtu matice rotace (1.13) úhel $-\alpha$. My ale naopak potřebujeme převést souřadnice z rotující tělesové soustavy do pevné, budeme proto při výpočtu matice rotace do (1.13) za úhel dosazovat přímo $\varphi(t)$. Matice přechodu mezi bázemi (od $\mathcal{X}$ k $\mathcal{X}'$) tedy teď po otočení tělesové soustavy bude

$$\mathbb{A}_{\mathcal{X}}^{\mathcal{X}'} = \mathbb{R}_{\varphi,\boldsymbol{o}}. \quad (1.15)$$

Matici rotace spočítáme podle (1.13), kde úhel rotace bude $\varphi(t)$ a osa rotace $\frac{\boldsymbol{\omega}'(t)}{\|\boldsymbol{\omega}'(t)\|}$. Když se objekt v dalším časovém úseku bude dál otáčet nějakou novou úhlovou rychlostí $\boldsymbol{\omega}'(t)$ v tělesové soustavě (opět s konst. osou rotace), tak opět můžeme vypočítat úhel rotace $\varphi(t)$ za tento časový úsek podle (1.12) a soustava se nám otočí o tento úhel do $\mathcal{X}''$. Podle (1.13) opět spočítáme novou matici rotace $\mathbb{R}_{\varphi,\boldsymbol{o}}$ (s novou osou a úhlem), a když teď po 2 otočeních budeme chtít pro vektor $\boldsymbol{a}$ z tělesové soustavy $\mathcal{X}''$ spočítat souřadnice v pevné soustavě $\mathcal{X}$, musíme spočítat novou matici přechodu mezi bázemi (od $\mathcal{X}$ k $\mathcal{X}''$). Souřadnice vektoru $\boldsymbol{a}$ nejdříve převedeme ze soustavy $\mathcal{X}''$ do $\mathcal{X}'$

$$(\boldsymbol{a})_{\mathcal{X}'} = \mathbb{R}_{\varphi,\boldsymbol{o}} \cdot (\boldsymbol{a})_{\mathcal{X}''}. \quad (1.16)$$

V soustavě $\mathcal{X}'$ už ale známe matici přechodu $\mathbb{A}_{\mathcal{X}}^{\mathcal{X}'}$, kterou jsme spočítali z předešlé rotace. Můžeme tedy souřadnice vektoru $(\boldsymbol{a})_{\mathcal{X}'}$ dál převést do pevné soustavy

$$(\boldsymbol{a})_{\mathcal{X}} = \mathbb{A}_{\mathcal{X}}^{\mathcal{X}'} \cdot (\boldsymbol{a})_{\mathcal{X}'}. \quad (1.17)$$

Dosadíme z (1.16) za $(\boldsymbol{a})_{\mathcal{X}'}$ a dostaneme

$$(\boldsymbol{a})_{\mathcal{X}} = \mathbb{A}_{\mathcal{X}}^{\mathcal{X}'} \cdot \mathbb{R}_{\varphi,\boldsymbol{o}} \cdot (\boldsymbol{a})_{\mathcal{X}''}. \quad (1.18)$$

Vzhledem k tomu, že tento vztah nám říká, jak převést souřadnice vektoru z báze $\mathcal{X}''$ do $\mathcal{X}$, tak můžeme rovnou psát, že matice přechodu mezi těmito bázemi je:

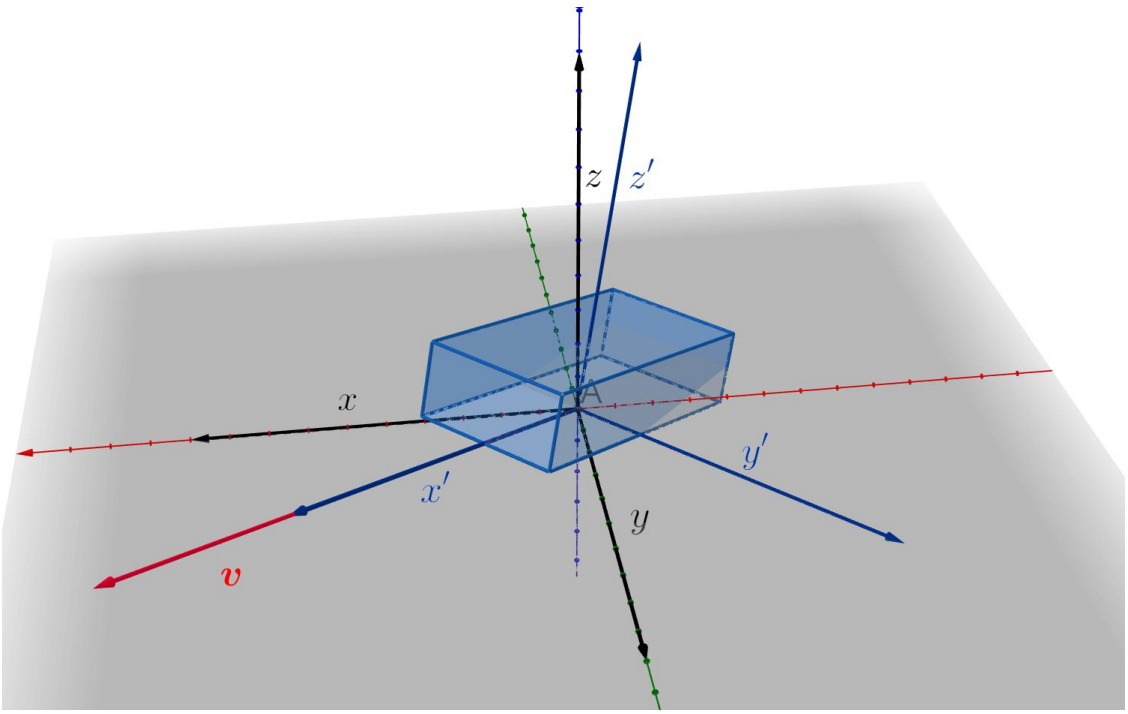
$$\mathbb{A}_{\mathcal{X}}^{\mathcal{X}''} = \mathbb{A}_{\mathcal{X}}^{\mathcal{X}'} \cdot \mathbb{R}_{\varphi,\boldsymbol{o}}. \quad (1.19)$$

Dostali jsme tedy v podstatě rekurzivní vztah, jak z matice přechodu od $\mathcal{X}$ k $\mathcal{X}'$ spočítat matici přechodu od $\mathcal{X}$ k $\mathcal{X}''$. Důležitá podmínka ovšem je, aby osa rotace byla v jednotlivých časových intervalech konstantní (neměnila směr), jinak bychom rotaci nemohli počítat pomocí matice $\mathbb{R}_{\varphi,\boldsymbol{o}}$.

Výhoda tohoto postupu je, že pro výpočet matice rotace podle (1.13) používáme úhlovou rychlost v souřadnicích rotující tělesové soustavy, to znamená přímo v souřadnicích, ve kterých ji měříme. Není tedy potřeba souřadnice úhlové rychlosti nijak převádět, což značně zrychluje algoritmus.

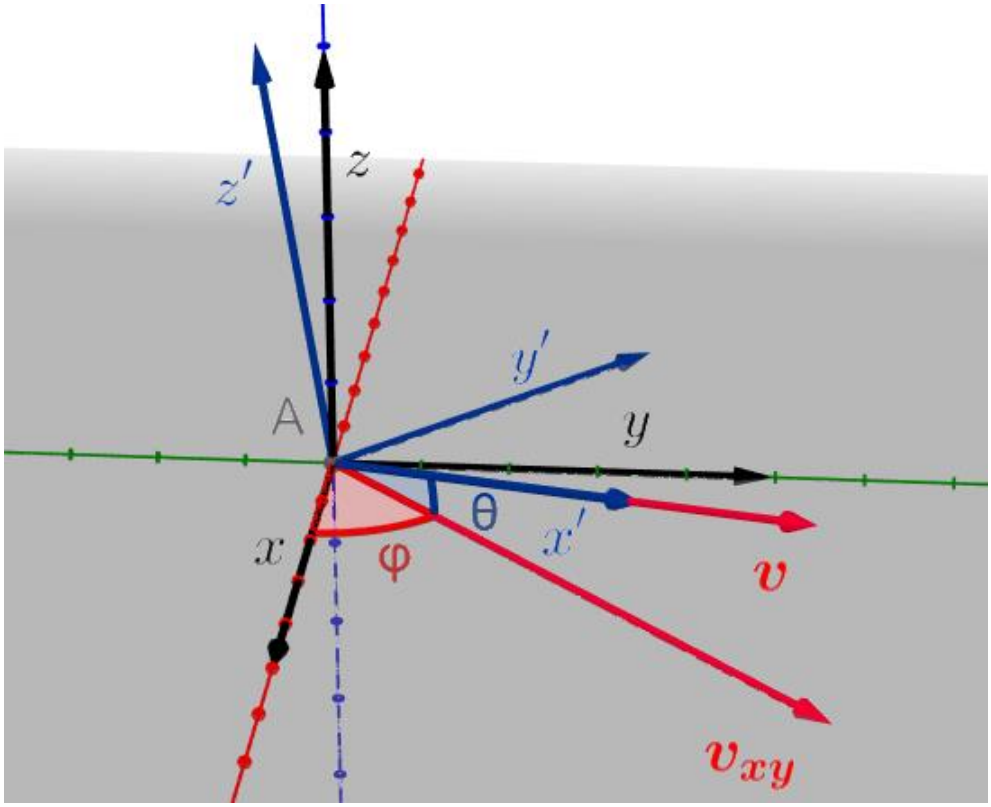
1.3.3 Odhad matice přechodu z rychlosti

Když se sledovaný objekt otáčí, můžeme matici přechodu pro převod souřadnic z tělesové do pevné soustavy počítat iterativně podle (1.19) nebo ji můžeme také odhadnout z aktuální rychlosti. Vyjdeme z toho, že pohybové senzory jsou ve sledovaném objektu umístěny tak, aby osa x' tělesové soustavy spojené pevně s objektem směřovala dopředu ve směru pohybu. Budeme-li dále předpokládat, že osa y' je rovnoběžná s povrchem Země (neboli s rovinou xy pevné soustavy $\mathcal{X}$), pak osa x' bude mít směr rychlosti objektu $\boldsymbol{v}$ a zbylé 2 osy určíme z předpokladu a z podmínky kolmosti os. Zanedbáme tím rotaci objektu kolem vlastní osy, což v některých případech můžeme docela dobře předpokládat. Situace je znázorněna na obrázku 1.3, kde je červeně vektor rychlosti objektu $\boldsymbol{v}$ ve stejném směru jako osa jeho tělesové soustavy x'.



Obrázek 1.3: Natočení tělesové soustavy $\mathcal{X}'$ vůči pevné $\mathcal{X}$.

Potom můžeme brát přechod mezi soustavami $\mathcal{X}$ a $\mathcal{X}'$ jako dvě složené rotace. Soustava $\mathcal{X}$ se nejdříve otočí kolem osy z o úhel φ, a poté kolem osy y' (už otočená osa y) o úhel θ. Situace je zobrazena na obrázku 1.4. Úhly rotací φ a θ můžeme určit z vektoru rychlosti $\boldsymbol{v}$, na obrázku 1.4 vidíme, že se jedná o úhly mezi rychlostí $\boldsymbol{v}$ a rovinou xy (úhel θ) a mezi průmětem rychlosti $\boldsymbol{v}$ do roviny xy ($\boldsymbol{v}_{xy}$) a osou x.



Obrázek 1.4: Určení přechodu mezi soustavami ze směru rychlosti.

Pro určení matic těchto rotací nepotřebujeme určovat přímo úhly φ a θ, stačí nám jejich sin( ) a cos( ). Pro úhel mezi vektorem rychlosti $\boldsymbol{v}$ a rovinou xy je to

$$sin\,\theta = \frac{v_z}{\|\boldsymbol{v}\|}, \tag{1.20}$$

$$cos\,\theta = \frac{\|\boldsymbol{v}_{xy}\|}{\|\boldsymbol{v}\|} = \frac{\sqrt{v_x^2 + v_y^2}}{\|\boldsymbol{v}\|}, \tag{1.21}$$

kde $\boldsymbol{v}_{xy}$ je průmět rychlosti $\boldsymbol{v}$ do roviny xy. Pro úhel mezi průmětem $\boldsymbol{v}_{xy}$ a osou x pak platí

$$sin\,\varphi = \frac{v_y}{\|\boldsymbol{v}_{xy}\|} = \frac{v_y}{\sqrt{v_x^2 + v_y^2}}, \tag{1.22}$$

$$cos\,\varphi = \frac{v_x}{\|\boldsymbol{v}_{xy}\|} = \frac{v_x}{\sqrt{v_x^2 + v_y^2}}. \tag{1.23}$$

Rotace kolem osy z o úhel φ pak bude dána maticí

$$\mathbb{R}_{\varphi,z} = \begin{pmatrix} cos\,\varphi & -sin\,\varphi & 0 \\ sin\,\varphi & cos\,\varphi & 0 \\ 0 & 0 & 1 \end{pmatrix}, \tag{1.24}$$

kolem osy y' o úhel θ to bude

$$\mathbb{R}_{\theta,y'} = \begin{pmatrix} cos\,\theta & 0 & -sin\,\theta \\ 0 & 1 & 0 \\ sin\,\theta & 0 & cos\,\theta \end{pmatrix}. \tag{1.25}$$

Výsledná matice přechodu pro převod souřadnic mezi soustavami $\mathcal{X}$ a $\mathcal{X}'$ se určí prostým složením těchto rotací – vynásobením jejich matic:

$$\mathbb{A}_{\mathcal{X}}^{\mathcal{X}'} = \mathbb{R}_{\varphi,z} \cdot \mathbb{R}_{\theta,y'}. \tag{1.26}$$

Potom už můžeme převádět souřadnice vektorů z $\mathcal{X}'$ do $\mathcal{X}$ podle (1.6).

1.4 Zrychlení

Jestliže má objekt v tělesové soustavě zrychlení $(\boldsymbol{a})_{\mathcal{X}'}$, tak podle kapitoly 1.3.2 zjistíme matici přechodu a podle (1.6) převedeme souřadnice vektoru zrychlení do pevné soustavy:

$$(\boldsymbol{a})_{\mathcal{X}} = \mathbb{A}_{\mathcal{X}}^{\mathcal{X}'} \cdot (\boldsymbol{a})_{\mathcal{X}'}. \tag{1.27}$$

Potom už jsme schopni ze zrychlení spočítat rychlost a z rychlosti a zrychlení polohu podle (1.1) a (1.4).

2 Model pohybu

Náš model pohybu používá popis pohybu podle kapitoly 1. Pro numerické výpočty je ovšem třeba popis diskretizovat v čase a některé funkce aproximovat. *Tímto se dopouštíme diskretizační chyby, která obecně roste s časovým krokem*, a časový krok budeme tedy chtít co nejmenší. Veličinu x v čase t_k budeme značit

$$x_k := x(t_k). \tag{2.1}$$

Časový krok mezi dvěma okamžiky pak

$$\Delta t_k := t_{k+1} - t_k. \tag{2.2}$$

2.1 Predikce rychlosti

Chceme odhadnout rychlost objektu v dalším časovém kroku. Pro zjednodušení budeme předpokládat, že zrychlení je během jednoho časového kroku konstantní, což pro malé časové kroky (časté měření akcelerometrem) bude dostatečně přesné. Potom můžeme (1.1) přepsat jako

$$\boldsymbol{v}_{k+1} = \boldsymbol{v}_k + \boldsymbol{a}_k \Delta t_k, \tag{2.3}$$

kde $\boldsymbol{v}_{k+1}$ je rychlost v následujícím $(k+1)$-ním okamžiku a $\boldsymbol{v}_k$, resp. $\boldsymbol{a}_k$ jsou rychlost, resp. zrychlení v aktuálním k-tém okamžiku. Zde je důležité si uvědomit, že do vztahu (2.3) musíme dosadit zrychlení objektu $\boldsymbol{a}_k$ v pevné soustavě $\mathcal{X}$, ve které měříme i rychlost objektu. Většinou totiž měříme zrychlení v tělesové soustavě $\mathcal{X}'$, která je pevně spojená s objektem a zpravidla se vůči $\mathcal{X}$ různě natáčí. Převod zrychlení z tělesové do pevné soustavy je vysvětlen v kapitole 2.4.

2.2 Predikce polohy

Pořád předpokládáme konstantní zrychlení během jednoho časového intervalu, polohu v dalším časovém kroku pak odhadneme podle (1.4)

$$\boldsymbol{r}_{k+1} = \boldsymbol{r}_k + \boldsymbol{v}_k \Delta t_k + \frac{1}{2}\boldsymbol{a}_k \Delta t_k^2\,, \tag{2.4}$$

kde $\boldsymbol{r}_{k+1}$ je poloha v následujícím $(k+1)$-ním okamžiku a $\boldsymbol{r}_k$, $\boldsymbol{v}_k$ a $\boldsymbol{a}_k$ jsou poloha, rychlost a zrychlení v aktuálním k-tém okamžiku. Pro dostatečně malé časové kroky je pak možné napsat

$$\boldsymbol{r}_{k+1} \approx \boldsymbol{r}_k + \boldsymbol{v}_k \Delta t_k, \tag{2.5}$$

jestliže bude $\boldsymbol{v}_k \Delta t_k \gg \frac{1}{2}\boldsymbol{a}_k \Delta t_k^2$. I zde platí stejně jako v předchozí části, že zrychlení $\boldsymbol{a}_k$ musíme brát v pevné soustavě $\mathcal{X}$.

2.3 Rotace – aktualizace matice přechodu

Když se objekt bude otáčet kolem nějaké osy (viz. Kapitola 1.3), bude se spolu s ním natáčet i tělesová soustava $\mathcal{X}'$, která je s ním spojená, vůči pevné soustavě $\mathcal{X}$. Může to být např. zatáčení vozidla nebo stoupání do kopce apod. Během tohoto otáčení se bude postupně měnit i matice přechodu mezi pevnou a tělesovou soustavou $\mathbb{A}_{\mathcal{X}}^{\mathcal{X}'}$. Pro zjednodušení výpočtu budeme předpokládat konstantní úhlovou rychlost objektu v jednom časovém intervalu. Osa rotace pak bude podle (1.10)

$$\boldsymbol{o}_k = \frac{\boldsymbol{\omega}_k}{\|\boldsymbol{\omega}_k\|}. \tag{2.6}$$

Úhel rotace podle (1.12)

$$\varphi_k = \|\boldsymbol{\omega}_k\| \Delta t_k, \tag{2.7}$$

kde $\boldsymbol{\omega}_k$ je úhlová rychlost objektu v aktuálním k-tém časovém okamžiku. Z toho podle (1.13) spočítáme matici rotace v k-tém časovém kroku $\mathbb{R}^{(k)}_{\boldsymbol{\varphi},\boldsymbol{o}}$, kterou následně dosadíme do (1.19) a dostaneme odhad matice přechodu do dalšího časového kroku

$$\mathbb{A}^{\mathcal{X}'_{k+1}}_{\mathcal{X}} = \mathbb{A}^{\mathcal{X}'_k}_{\mathcal{X}} \cdot \mathbb{R}^{(k)}_{\boldsymbol{\varphi},\boldsymbol{o}}, \tag{2.8}$$

kde $\mathbb{A}^{\mathcal{X}'_{k+1}}_{\mathcal{X}}$ je matice přechodu v následujícím $(k+1)$-ním okamžiku a $\mathbb{A}^{\mathcal{X}'_k}_{\mathcal{X}}$ je matice přechodu v aktuálním k-tém okamžiku.

2.4 Převod zrychlení z tělesové do pevné soustavy

Když známe matici přechodu, můžeme převést souřadnice zrychlení z tělesové soustavy do pevné podle (1.6):

$$(\boldsymbol{a}_k)_{\mathcal{X}} = \mathbb{A}^{\mathcal{X}'_k}_{\mathcal{X}} \cdot (\boldsymbol{a}_k)_{\mathcal{X}'}. \tag{2.9}$$

Pak už můžeme spočítat predikci rychlosti a polohy podle (2.3) a (2.4).

3 Fúze dat

3.1 Fúze dat více zdrojů polohy

Jestliže máme k dispozici více zdrojů polohy, které měří současně, můžeme jejich měření sloučit do jednoho přesnějšího odhadu polohy. Změříme-li veličinu X vícekrát, nejlepší výsledný odhad její hodnoty bude [2]

$$\hat{x} = \frac{\sum_i \frac{\bar{x}_i}{\sigma_i^2}}{\sum_i \frac{1}{\sigma_i^2}}, \tag{3.1}$$

kde $\bar{x}_i$ jsou střední hodnoty jednotlivých měření a σ_i^2 jsou jejich rozptyly. Je to v podstatě vážený průměr všech měření, kde váhou je převrácená hodnota jejich rozptylu. Je potřeba ovšem dodat, že to platí pouze za předpokladu, že jednotlivá měření nejsou zatížena systematickými chybami. Jinými slovy, že chyba měření je náhodná a má nulovou střední hodnotu. Pro odhad polohy

$$\boldsymbol{r} = \begin{pmatrix} x \\ y \\ z \end{pmatrix} \tag{3.2}$$

z více zdrojů můžeme vztah (3.1) aplikovat ve všech 3 dimenzích a spočítáme tak $\hat{x}, \hat{y}, \hat{z}$. Pokud máme k dispozici rozptyly polohy zvlášť v jednotlivých směrech

$$\begin{pmatrix} \sigma_x^2 \\ \sigma_y^2 \\ \sigma_z^2 \end{pmatrix}, \tag{3.3}$$

spočítáme odhad pro každý směr zvlášť s příslušnými rozptyly. Předpokládáme zde, že jednotlivé směry jsou nezávislé, což odpovídá diagonální kovarianční matici vektoru $\boldsymbol{r}$:

$$P = \begin{pmatrix} \sigma_x^2 & 0 & 0 \\ 0 & \sigma_y^2 & 0 \\ 0 & 0 & \sigma_z^2 \end{pmatrix}. \tag{3.4}$$

3.2 Kalmanův filtr

3.2.1 Základní definice

Pro fúzi dat z měření polohy, zrychlení, úhlové rychlosti a dalších veličin pohybu můžeme použít Kalmanův filtr. Kalmanův filtr je algoritmus, který nám říká, *jak nejlépe odhadnout hodnotu nějaké měřené veličiny, která se v čase mění lineárně* [3] (nebo se alespoň její průběh dá v jednotlivých časových krocích linearizovat). Algoritmus zde popsaný je tzv. „open-loop“ verze Kalmanova filtru. Složky stavového vektoru

$$\boldsymbol{x} = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} \tag{3.5}$$

x_i jsou naše veličiny, jejichž hodnotu chceme odhadnout. Chyby našeho odhadu představuje jeho kovarianční matice

$$P_{ij} = cov(x_i, x_j), \tag{3.6}$$

což můžeme rozepsat v čitelnější podobě:

$$P = \begin{pmatrix} var(x_1) & cov(x_1, x_2) & \cdots & cov(x_1, x_n) \\ cov(x_2, x_1) & var(x_2) & \cdots & cov(x_2, x_n) \\ \vdots & \vdots & \ddots & \vdots \\ cov(x_n, x_1) & cov(x_n, x_2) & \cdots & var(x_n) \end{pmatrix}. \tag{3.7}$$

Diagonální prvky kovarianční matice jsou rozptyly odhadovaných veličin $var(x_i) = \sigma^2(x_i)$, prvky mimo diagonálu jsou kovariance jednotlivých veličin, které představují závislost mezi nimi. Jestliže budou měřené veličiny na sobě nezávislé, pak jejich kovariance budou nulové a matice bude mít diagonální tvar

$$P = \begin{pmatrix} \sigma^2(x_1) & 0 & \cdots & 0 \\ 0 & \sigma^2(x_2) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \sigma^2(x_n) \end{pmatrix}. \tag{3.8}$$

Kovarianční matice je tedy čtvercová rozměru $n \times n$ (pro n-rozměrný vektor) a musí být vždy symetrická podle diagonály. Složky vektoru měření

$$\boldsymbol{y} = \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix} \tag{3.9}$$

y_i jsou přímo měřené veličiny, což nemusí být veličiny, které chceme odhadovat. V případě nepřímého měření mezi nimi bude vztah

$$\boldsymbol{y} = h(\boldsymbol{x}). \tag{3.10}$$

Například, když budeme chtít odhadnout polohu (to bude naše $\boldsymbol{x}$), tak často nebudeme měřit přímo polohu, ale například vzdálenosti od různých objektů – družic, kotev, atd. (to bude naše $\boldsymbol{y}$). Z těchto vzdáleností bychom pak museli dopočítat polohu pomocí h^{-1}. Jestliže bude h lineární (když ne, tak ji můžeme zkusit linearizovat), můžeme ji nahradit násobením maticí:

$$\boldsymbol{y} = H \cdot \boldsymbol{x} \tag{3.11}$$

kde matici H nazýváme maticí měření. V případě, že měříme přímo veličiny, které chceme odhadovat, bude

$$H = \mathbb{I}, \tag{3.12}$$

h tedy bude identita a H jednotková matice. Potom bude platit

$$\boldsymbol{y} = \boldsymbol{x}. \tag{3.13}$$

Chyby přímo měřených veličin představuje kovarianční matice vektoru měření $\boldsymbol{y}$

$$R_{ij} = cov(y_i, y_j), \tag{3.14}$$

Což můžeme opět rozepsat jako

$$R = \begin{pmatrix} var(y_1) & cov(y_1, y_2) & \cdots & cov(y_1, y_n) \\ cov(y_2, y_1) & var(y_2) & \cdots & cov(y_2, y_n) \\ \vdots & \vdots & \ddots & \vdots \\ cov(y_n, y_1) & cov(y_n, y_2) & \cdots & var(y_n) \end{pmatrix}. \quad (3.15)$$

Jestliže budou jednotlivé veličiny navzájem nezávislé, budou jejich kovariance opět nulové a matice bude diagonální:

$$R = \begin{pmatrix} \sigma^2(y_1) & 0 & \cdots & 0 \\ 0 & \sigma^2(y_2) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \sigma^2(y_n) \end{pmatrix}. \quad (3.16)$$

3.2.2 Predikce

Písmeno k v dolním indexu značí v následující části časový index. Jestliže se odhadované veličiny ve stavovém vektoru $\boldsymbol{x}$ lineárně mění v čase, můžeme na základě současného stavu udělat predikci do dalšího časového kroku

$$\tilde{\boldsymbol{x}}_{k+1} = \Phi_k \cdot \hat{\boldsymbol{x}}_k, \quad (3.17)$$

kde vlnovka $\tilde{\boldsymbol{x}}_{k+1}$ reprezentuje predikci stavu do dalšího kroku a střecha $\hat{\boldsymbol{x}}_k$ filtrovaný odhad stavu v současném kroku. Matice Φ_k reprezentuje lineární změnu odhadovaných veličin v aktuálním časovém kroku. Nejjednodušší odhad chyby této predikce je potom dán její kovarianční maticí [4]

$$\tilde{P}_{k+1} = \Phi_k \hat{P}_k \Phi_k^T + Q_k, \quad (3.18)$$

kde $\hat{P}_k$ je kovarianční matice filtrovaného odhadu stavu v současném kroku a Q_k je kovarianční matice, která reprezentuje příspěvek chyby v jednom kroku, v důsledku šumu systému. Součástí tohoto šumu je i to, že neznáme přesně Φ_k.

3.2.3 Korekce – filtrace

Nejdříve musíme spočítat matici Kalmanova zisku, ta nám určuje, o kolik více věříme novému měření než predikci z předchozího stavu na základě porovnání jejich rozptylů, resp. kovariančních matic. Matici Kalmanova zisku spočítáme jako [4]

$$K_k = \tilde{P}_k H_k^T \left(H_k \tilde{P}_k H_k^T + R_k\right)^{-1}, \quad (3.19)$$

kde $(\ldots)^{-1}$ značí inverzní matici, $\tilde{P}_k$ je kovarianční matice predikce současného stavu a R_k je kovarianční matice aktuálního měření. Když máme Kalmanův zisk, můžeme sloučit predikci a nové měření a výsledný filtrovaný odhad stavu bude [4]

$$\hat{\boldsymbol{x}}_k = \tilde{\boldsymbol{x}}_k + K_k(\boldsymbol{y}_k - H_k \tilde{\boldsymbol{x}}_k), \quad (3.20)$$

kde $\tilde{\boldsymbol{x}}_k$ je predikce současného stavu a $\boldsymbol{y}_k$ je aktuální měření. Výsledná chyba našeho filtrovaného odhadu stavu bude dána kovarianční maticí [4]

$$\hat{P}_k = (\mathbb{I} - K_k H_k)\tilde{P}_k, \quad (3.21)$$

toto už je ovšem zjednodušená verze původního tvaru [3]

$$\hat{P}_k = (\mathbb{I} - K_k H_k)\tilde{P}_k(\mathbb{I} - K_k H_k)^T + K_k R_k K_k^T, \quad (3.22)$$

Joseph v [3] uvádí, že toto zjednodušení je špatné, ačkoliv je formálně správně, protože je numericky nestabilní. Podle něho i malá chyba v určení Kalmanova zisku (3.19) vede při použití vztahu (3.21) pro kovarianci filtrovaného odhadu stavu k velikým chybám, zatímco vztah (3.22) by měl dávat i v případě nepřesně určeného Kalmanova zisku rozumné výsledky. V experimentu, kde jsme algoritmus testovali na reálných datech, jsme ovšem nezaznamenali žádnou pozorovatelnou změnu ve výsledcích při použití (3.21) nebo (3.22).

3.2.3.1 *Přímé měření → $H = I$*

Pokud budeme měřit přímo stavové veličiny, a tedy $H = \mathbb{I}$, potom se předchozí vztahy výrazně zjednoduší. Matice kalmanova zisku (3.19) se zjednoduší na

$$K_k = \tilde{P}_k\left(\tilde{P}_k + R_k\right)^{-1}. \tag{3.23}$$

Výsledný filtrovaný odhad stavu (3.20) bude

$$\hat{\boldsymbol{x}}_k = \tilde{\boldsymbol{x}}_k + K_k(\boldsymbol{y}_k - \tilde{\boldsymbol{x}}_k). \tag{3.24}$$

Jeho kovarianční matice (3.21) se zjednoduší na

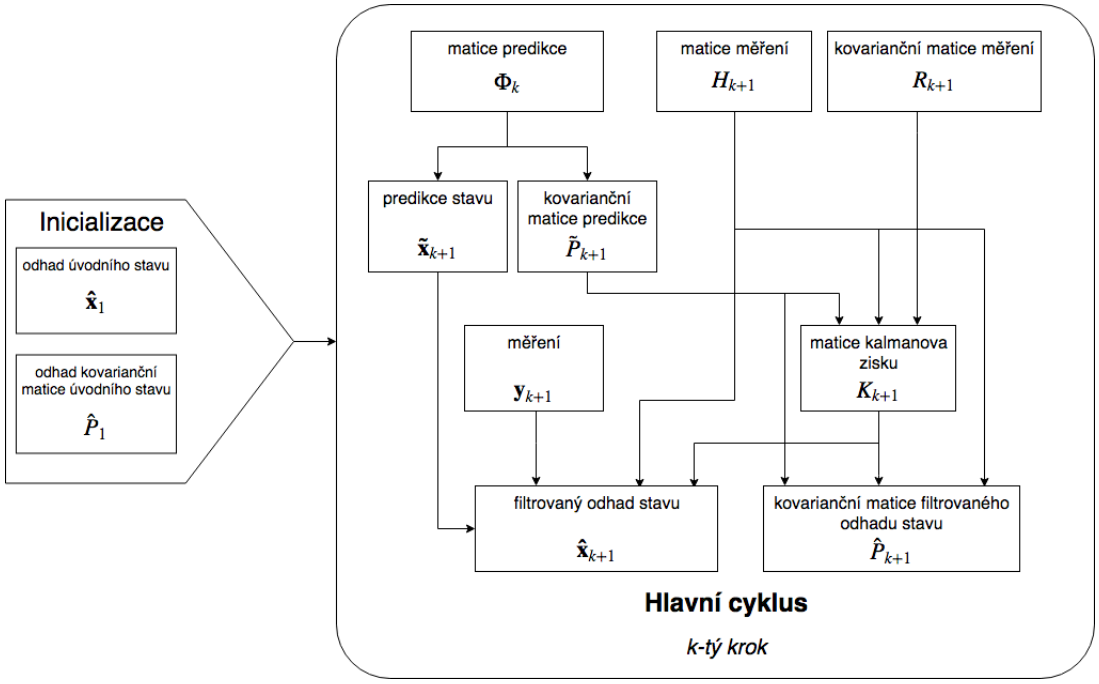
$$\hat{P}_k = (\mathbb{I} - K_k)\tilde{P}_k, \tag{3.25}$$

případně při použití numericky stabilního vztahu (3.22) na

$$\hat{P}_k = (\mathbb{I} - K_k)\tilde{P}_k(\mathbb{I} - K_k)^T + K_k R_k K_k^T. \tag{3.26}$$

3.2.4 Průběh algoritmu

Algoritmus Kalmanova filtru tedy začíná nějakým prvotním odhadem stavu a jeho chyb (kovarianční matice), a poté opakuje cyklus 1. predikce, 2. korekce – filtrace, jehož kroky jsou popsány v kapitolách 3.2.2 a 3.2.3. Následující schéma na obrázku 3.1 znázorňuje postup výpočtu.



Obrázek 3.1: Schéma algoritmu Kalmanova filtru.

3.3 Filtr polohy

3.3.1 Inicializace

Určení polohy z GNSS systémů může často selhat nebo může být zatížené velkou chybou. Důvodů může být mnoho, např. nedostatečná úroveň signálu, odraz signálu od budov – vícecestné šíření, vegetace nebo různých jiných objektů. Pro lepší odhad polohy – filtrování chyb a pro případ, kdy měření polohy z GNSS systému není možné vůbec, potřebujeme přidat nějakou další informaci do systému, která by nám umožnila dělat predikce následujících stavů. Proto spojujeme data z GNSS systému s daty z pohybových senzorů (zde 3-osý akcelerometr a gyroskop) skrze Kalmanův filtr popsaný v kapitole 3.2. Predikci polohy a rychlosti jsme potom schopni spočítat, díky měření zrychlení a úhlové rychlosti, podle modelu pohybu v kapitole 2. První věc, kterou při návrhu Kalmanova filtru musíme udělat, je zvolit si vhodný stavový vektor. Ve stavovém vektoru jsou veličiny, jejichž hodnotu chceme odhadovat (které chceme filtrovat). Zde ho volíme složený z polohy a rychlosti se souřadnicemi v pevné soustavě $\mathcal{X}$ (např. ENU)

$$\hat{\boldsymbol{x}} = \begin{pmatrix} \hat{\boldsymbol{r}} \\ \hat{\boldsymbol{v}} \end{pmatrix}, \tag{3.27}$$

což je rozepsané po složkách

$$\hat{\boldsymbol{x}} = \begin{pmatrix} \hat{r}^x \\ \hat{r}^y \\ \hat{r}^z \\ \hat{v}^x \\ \hat{v}^y \\ \hat{v}^z \end{pmatrix}. \tag{3.28}$$

Složky píšeme v horním indexu, abychom odlišili časový dolní index. Střechy znázorňují odhad dané veličiny. Polohu a rychlost určujeme ze systému GNSS a vektor měření bude

$$\boldsymbol{y} = \begin{pmatrix} \boldsymbol{r} \\ \boldsymbol{v} \end{pmatrix}. \tag{3.29}$$

Měříme tedy přímo stavové veličiny, takže naše matice měření bude identita $H = \mathbb{I}$. Inicializaci provedeme odhadem prvního stavu přímo měřenou polohou a rychlostí ze systému GNSS

$$\hat{\boldsymbol{x}}_1 = \begin{pmatrix} \boldsymbol{r}_1 \\ \boldsymbol{v}_1 \end{pmatrix}, \tag{3.30}$$

kovarianční matice prvního stavu bude diagonální

$$\hat{P}_1 = \begin{pmatrix} \sigma^2(r_1^x) & & & & & \\ & \sigma^2(r_1^y) & & & & \\ & & \sigma^2(r_1^z) & & & \\ & & & \sigma^2(v_1^x) & & \\ & & & & \sigma^2(v_1^y) & \\ & & & & & \sigma^2(v_1^z) \end{pmatrix}, \tag{3.31}$$

předpokládáme, že pohyb a rychlost jsou v jednotlivých směrech a mezi sebou nezávislé. Z prvotního odhadu rychlosti $\boldsymbol{v}_1$ (3.30) určíme matici přechodu $\mathbb{A}_{\mathcal{X}}^{\mathcal{X}_1'}$ pro převod souřadnic mezi tělesovou soustavou $\mathcal{X}'$ a pevnou $\mathcal{X}$ podle (1.26) v kapitole 1.3.3.

3.3.2 Predikce

Vytváříme predikci následujícího stavu, tedy polohy a rychlosti. Budeme postupovat způsobem popsaným v kapitolách 2.2 a 2.1, ovšem pro použití v Kalmanově filtru musíme vztahy přepsat do maticové formy. Je to zejména z důvodu výpočtu jednotlivých kovariančních matic, které představují chyby našich odhadů a vzájemné závislosti těchto odhadů. Nejprve ze všeho musíme převést změřené zrychlení z tělesové soustavy $\mathcal{X}'$ do pevné $\mathcal{X}$. Použijeme matici přechodu $\mathbb{A}_{\mathcal{X}}^{\mathcal{X}_k'}$ a podle (2.9) dostaneme zrychlení v pevné soustavě

$$(\boldsymbol{a}_k)_{\mathcal{X}} = \mathbb{A}_{\mathcal{X}}^{\mathcal{X}_k'} \cdot (\boldsymbol{a}_k)_{\mathcal{X}'}. \tag{3.32}$$

Akcelerometr měří kromě zrychlení objektu i tíhové zrychlení g*, po převodu zrychlení do pevné soustavy je tedy potřeba toto kompenzovat a tíhové zrychlení odečíst*

$$\left(\boldsymbol{a}_k^{komp}\right)_{\mathcal{X}} = (\boldsymbol{a}_k)_{\mathcal{X}} - \begin{pmatrix} 0 \\ 0 \\ g \end{pmatrix}. \tag{3.33}$$

Měření zrychlení akcelerometrem je zatíženo různými chybami. Pokud budeme předpokládat, že tyto chyby jsou náhodné s nulovou střední hodnotou a rozptylem $\sigma_{\boldsymbol{a}}^2$, pak zrychlení $(\boldsymbol{a}_k)_{\mathcal{X}'}$ v tělesové soustavě bude mít kovarianční matici

$$\sigma_{\boldsymbol{a}}^2 I = \begin{pmatrix} \sigma_{\boldsymbol{a}}^2 & 0 & 0 \\ 0 & \sigma_{\boldsymbol{a}}^2 & 0 \\ 0 & 0 & \sigma_{\boldsymbol{a}}^2 \end{pmatrix}. \tag{3.34}$$

Když zrychlení převedeme do jiné soustavy podle (2.9), násobíme jeho vektor maticí, při tom se ovšem změní i jeho kovarianční matice, takže zrychlení v pevné soustavě $(\boldsymbol{a}_k)_{\mathcal{X}}$ bude mít kovarianční matici [4]

$$\mathbb{A}_{\mathcal{X}}^{\mathcal{X}_k'} \cdot \sigma_{\boldsymbol{a}}^2 \mathbb{I} \cdot \left(\mathbb{A}_{\mathcal{X}}^{\mathcal{X}_k'}\right)^T. \tag{3.35}$$

Predikce následujícího stavu bude

$$\widetilde{\boldsymbol{x}}_{k+1} = \Phi_k \cdot \widehat{\boldsymbol{x}}_k + \Delta \boldsymbol{v}_k, \tag{3.36}$$

kde $\widehat{\boldsymbol{x}}_k$ je odhad současného stavu, $\Delta \boldsymbol{v}_k$ je vektor změny rychlosti a Φ_k je matice, která reprezentuje lineární změnu stavu v rámci jednoho časového kroku. V našem případě je tato lineární změna stavu pohyb objektu, a Φ_k je tedy jakási „matice pohybu“. Vektor změny rychlosti se spočítá ze zrychlení podle (2.3)

$$\Delta \boldsymbol{v}_k = \Delta t_k \cdot \begin{pmatrix} 0 \\ 0 \\ 0 \\ a_k^x \\ a_k^y \\ a_k^z \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ a_k^x \Delta t_k \\ a_k^y \Delta t_k \\ a_k^z \Delta t_k \end{pmatrix}, \tag{3.37}$$

kde Δt_k je délka aktuálního časového kroku a a_k^x, a_k^y a a_k^z jsou složky zrychlení v pevné soustavě $\mathcal{X}$ spočítané pomocí (2.9) a kompenzované vůči tíhovému zrychlení podle (3.33). Matice reprezentující pohyb se pak spočítá podle (2.5)

$$\Phi_k = \begin{pmatrix} 1 & & & \Delta t_k & & \\ & 1 & & & \Delta t_k & \\ & & 1 & & & \Delta t_k \\ & & & 1 & & \\ & & & & 1 & \\ & & & & & 1 \end{pmatrix}. \tag{3.38}$$

Kovarianční matici predikce následujícího stavu (3.36) odhadneme podle (3.18)

$$\tilde{P}_{k+1} = \Phi_k \hat{P}_k \Phi_k^T + Q_k, \tag{3.39}$$

kde $\tilde{P}_{k+1}$ je kovarianční matice predikce následujícího stavu, $\hat{P}_k$ je kovarianční matice odhadu současného stavu, Φ_k je matice reprezentující pohyb objektu a Q_k zde reprezentuje chyby změny rychlosti $\Delta \boldsymbol{v}_k$, které vycházejí z kovarianční matice zrychlení měřeného akcelerometrem (3.35)

$$Q_k = \begin{pmatrix} 0 & & & \\ & 0 & & \\ & & 0 & \\ & & & \mathbb{A}_{\mathcal{X}}^{\mathcal{X}_k'} \cdot \sigma_{\boldsymbol{a}}^2 \mathbb{I} \cdot \left(\mathbb{A}_{\mathcal{X}}^{\mathcal{X}_k'}\right)^T \end{pmatrix}. \tag{3.40}$$

3.3.2.1 Aktualizace matice přechodu

Z naměřené úhlové rychlosti objektu $\boldsymbol{\omega}_k$ podle kapitoly 2.3, určíme nový odhad matice přechodu pro následující časový krok. Po spočítání matice rotace $\mathbb{R}_{\boldsymbol{\varphi},\boldsymbol{o}}^{(k)}$ aktualizujeme matici přechodu podle (2.8)

$$\mathbb{A}_{\mathcal{X}}^{\mathcal{X}_{k+1}'} = \mathbb{A}_{\mathcal{X}}^{\mathcal{X}_k'} \cdot \mathbb{R}_{\boldsymbol{\varphi},\boldsymbol{o}}^{(k)}. \tag{3.41}$$

3.3.3 Korekce – filtrace

V tomto kroku porovnáme predikci současného stavu s jeho naměřenými hodnotami. V našem případě měříme přímo stavové veličiny – polohu a rychlost. Matice měření bude tedy identita $H = \mathbb{I}$. Nejdříve tedy spočítáme matici Kalmanova zisku podle (3.23)

$$K_k = \tilde{P}_k \left(\tilde{P}_k + R_k\right)^{-1}, \tag{3.42}$$

kde $(\ldots)^{-1}$ značí inverzní matici, $\tilde{P}_k$ je kovarianční matice predikce současného stavu a R_k je kovarianční matice současného měření. Filtrovaný odhad stavu pak bude podle (3.24)

$$\hat{\boldsymbol{x}}_k = \tilde{\boldsymbol{x}}_k + K_k(\boldsymbol{y}_k - \tilde{\boldsymbol{x}}_k), \tag{3.43}$$

kde $\tilde{\boldsymbol{x}}_k$ je predikce současného stavu a $\boldsymbol{y}_k$ je jeho měření, v našem případě

$$\boldsymbol{y}_k = \begin{pmatrix} \boldsymbol{r}_k \\ \boldsymbol{v}_k \end{pmatrix}. \tag{3.44}$$

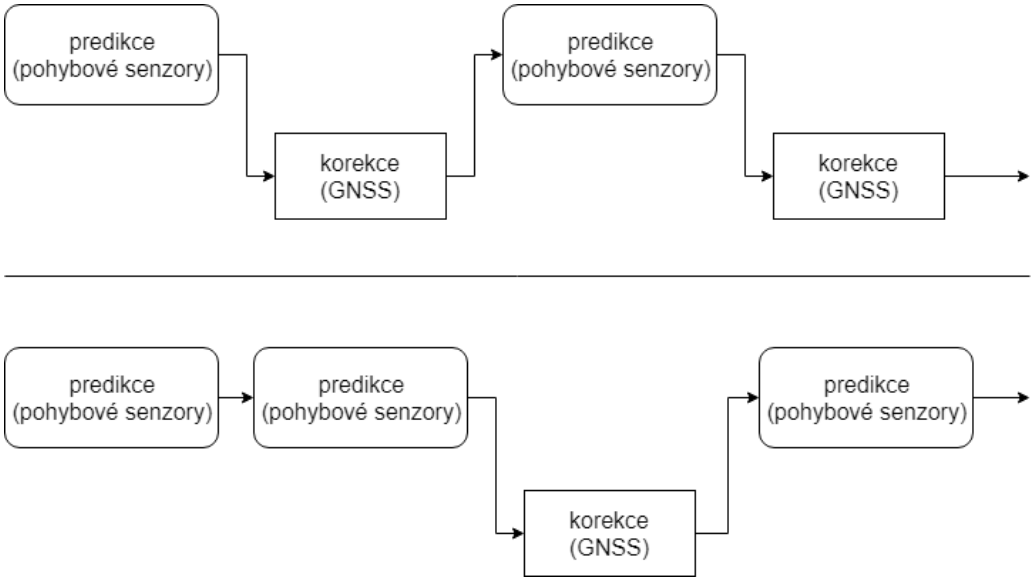
Chyba výsledného filtrovaného stavu bude dána kovarianční maticí podle (3.25)

$$\hat{P}_k = (\mathbb{I} - K_k)\tilde{P}_k, \tag{3.45}$$

případně můžeme použít numericky stabilnějšího vztahu (3.26), to se však při testování algoritmu neprojevilo jako nutné.

Při reálném použití Kalmanova filtru nemusí po každé predikci nutně přijít krok korekce, ten totiž přichází, jen pokud máme k dispozici nová naměřená data, tedy nový vektor měření $\boldsymbol{y}$. Často

můžeme naopak několikrát za sebou opakovat krok predikce, dokud nedostaneme nová naměřená data. V reálném případě například GNSS přijímač bude měřit jednotlivé polohy podstatně pomaleji, než jsou schopny měřit pohybové senzory. Na obrázku 3.2 jsou vidět dva z možných průběhů Kalmanova filtru, v prvním průběhu přichází po každé predikci z dat pohybových senzorů měření a korekce pomocí systému GNSS. Ve druhém průběhu je měření ze systému GNSS dvakrát pomalejší než z pohybových senzorů, a krok predikce se tedy opakuje dvakrát za sebou, než přijde měření a korekce z GNSS. Je ovšem důležité zdůraznit, že při opakovaném vytváření predikce bez následné korekce se nám s každým krokem zvětšuje její chyba podle vztahu (3.39).



Obrázek 3.2: Možné průběhy algoritmu Kalmanova filtru.

3.3.4 Korekce matice přechodu z rychlosti

V kroku korekce, kdy započítáváme nové měření ze systému GNSS, provádíme také korekci matice přechodu $\mathbb{A}_{\mathcal{X}}^{\mathcal{X}'_{k+1}}$ pomocí filtrovaného odhadu rychlosti. Podle kapitoly 1.3.3 odhadneme z rychlosti $\hat{\boldsymbol{v}}_k$ matici přechodu a nazveme ji $\mathbb{A}_{k+1}^{\boldsymbol{v}}$. Výsledný odhad matice přechodu spočítáme jako vážený průměr

$$\widehat{\mathbb{A}}_{\mathcal{X}}^{\mathcal{X}'_{k+1}} = \frac{1}{1+m}\left(\mathbb{A}_{\mathcal{X}}^{\mathcal{X}'_{k+1}} + m \cdot \mathbb{A}_{k+1}^{\boldsymbol{v}}\right), \tag{3.46}$$

kde m je váha se kterou započítáváme odhad matice přechodu z rychlosti. Po takto provedeném odhadu je možné, že matice přechodu už nebude čistě rotační, proto je potřeba ji opravit podle následující kapitoly 3.3.5.

3.3.5 Numerická korekce matice rotace

Každá matice rotace $\mathbb{R}$ musí splňovat určitá kritéria. Matice rotace je ortogonální matice a její determinant $\det \mathbb{R} = 1$. Ortogonalita matice znamená

$$\mathbb{R}^T\mathbb{R} = \mathbb{R}\mathbb{R}^T = \mathbb{I}, \tag{3.47}$$

z čehož plyne vztah pro inverzní matici

$$\mathbb{R}^{-1} = \mathbb{R}^T, \tag{3.48}$$

přičemž inverzní matice $\mathbb{R}^{-1}$ k matici rotace $\mathbb{R}$ představuje „inverzní rotaci“. Jestliže je $\mathbb{R}$ matice rotace kolem nějaké osy o úhel α, tak $\mathbb{R}^{-1}$ je matice rotace kolem stejné osy o úhel $-\alpha$.

Naše matice přechodu $\mathbb{A}_{\mathcal{X}}^{\mathcal{X}'}$, pro převod souřadnic mezi tělesovou a pevnou soustavou, je postupně složená z různých rotací (viz v kapitole 2.3 vztah (2.8)), jak se objekt v čase různě natáčí, a tudíž naše matice přechodu $\mathbb{A}_{\mathcal{X}}^{\mathcal{X}'}$ je také maticí nějaké rotace. To znamená, že i matice $\mathbb{A}_{\mathcal{X}}^{\mathcal{X}'}$ musí splňovat kritéria pro matice rotace a v každém kroku tedy musí platit

$$det\, \mathbb{A}_{\mathcal{X}}^{\mathcal{X}'} = 1. \tag{3.49}$$

Vlivem různých numerických chyb, jak postupně skládáme jednotlivé rotace násobením jejich matic, se ovšem postupně může stát, že determinant matice přechodu nebude 1. V takovém případě už matice přechodu nebude představovat pouze rotaci, ale bude nám měnit i velikost převáděných vektorů. Zrychlení, které pomocí matice přechodu převádíme z tělesové do pevné soustavy, bychom tak vlastně převodem soustavně zvětšovali nebo zmenšovali a do systému bychom tak zavedli systematickou chybu.

Provedeme proto korekci matice přechodu podle následující metody [1], která opravuje matici, aby byla čistě rotační. Mějme matici

$$\mathbb{R} = \begin{pmatrix} R_{11} & R_{12} & R_{13} \\ R_{21} & R_{22} & R_{23} \\ R_{31} & R_{32} & R_{33} \end{pmatrix}, \tag{3.50}$$

jejíž sloupce jsou vektory $\boldsymbol{r}_1, \boldsymbol{r}_2, \boldsymbol{r}_3$

$$\boldsymbol{r}_1 = \begin{pmatrix} R_{11} \\ R_{21} \\ R_{31} \end{pmatrix}, \boldsymbol{r}_2 = \begin{pmatrix} R_{12} \\ R_{22} \\ R_{32} \end{pmatrix}, \boldsymbol{r}_3 = \begin{pmatrix} R_{13} \\ R_{23} \\ R_{33} \end{pmatrix}. \tag{3.51}$$

Nejdříve spočítáme odchylky od ortogonality jednotlivých sloupečků matice

$$e_{\boldsymbol{r}_1\boldsymbol{r}_2} = \boldsymbol{r}_1 \cdot \boldsymbol{r}_2, \tag{3.52}$$

$$e_{\boldsymbol{r}_2\boldsymbol{r}_3} = \boldsymbol{r}_2 \cdot \boldsymbol{r}_3, \tag{3.53}$$

$$e_{\boldsymbol{r}_1\boldsymbol{r}_3} = \boldsymbol{r}_1 \cdot \boldsymbol{r}_3, \tag{3.54}$$

poté sloupce ortogonalizujeme

$$\boldsymbol{r}_1' = \boldsymbol{r}_1 - \frac{e_{\boldsymbol{r}_1\boldsymbol{r}_2}}{2}\boldsymbol{r}_2 - \frac{e_{\boldsymbol{r}_1\boldsymbol{r}_3}}{2}\boldsymbol{r}_3, \tag{3.55}$$

$$\boldsymbol{r}_2' = \boldsymbol{r}_2 - \frac{e_{\boldsymbol{r}_1\boldsymbol{r}_2}}{2}\boldsymbol{r}_1 - \frac{e_{\boldsymbol{r}_2\boldsymbol{r}_3}}{2}\boldsymbol{r}_3, \tag{3.56}$$

$$\boldsymbol{r}_3' = \boldsymbol{r}_3 - \frac{e_{\boldsymbol{r}_1\boldsymbol{r}_3}}{2}\boldsymbol{r}_1 - \frac{e_{\boldsymbol{r}_2\boldsymbol{r}_3}}{2}\boldsymbol{r}_2. \tag{3.57}$$

A pak už jen normalizujeme jednotlivé sloupce a dostaneme korigovanou matici rotace

$$\mathbb{R}' = \begin{pmatrix} \dfrac{\boldsymbol{r}_1'}{\|\boldsymbol{r}_1'\|} & \dfrac{\boldsymbol{r}_2'}{\|\boldsymbol{r}_2'\|} & \dfrac{\boldsymbol{r}_3'}{\|\boldsymbol{r}_3'\|} \end{pmatrix}. \tag{3.58}$$

3.3.6 Kompenzace systematických chyb senzorů

Pohybové senzory měří vždy s nějakou chybou. Předpokládáme, že tato chyba je složena z náhodné chyby s nulovou střední hodnotou a chyby systematické. Náhodná chyba nepředstavuje tak velký problém, protože právě díky její nulové střední hodnotě je filtr schopen ji docela dobře potlačit. Co ovšem už tak jednoduše odfiltrovat nelze, to jsou systematické chyby jednotlivých senzorů. Jedná se o

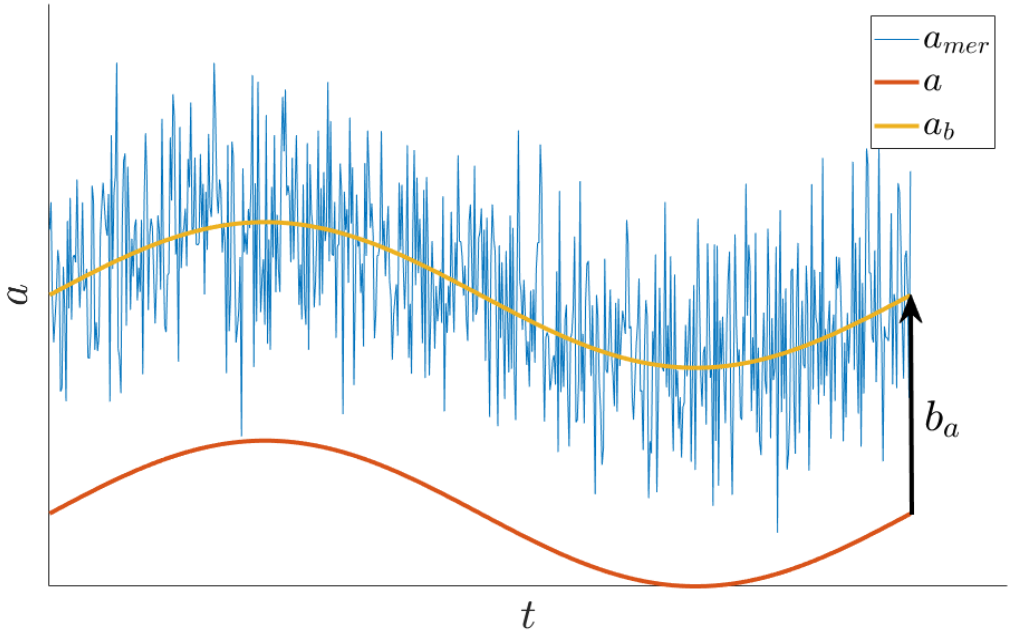
konstantní posunutí měřené hodnoty od skutečné. Ve skutečnosti je situace ještě o něco složitější, protože systematická chyba nemusí být po celý čas konstantní, ale postupně se může měnit. Pouze na nějakém dostatečně krátkém časovém intervalu můžeme brát systematickou chybu jako konstantu.

Měřená úhlová rychlost z gyroskopu se ve filtru postupně sčítá (integruje) na úhel rotace, což postupně mění matici přechodu mezi tělesovou a pevnou soustavou (viz. kapitola 2.3). Stejně tak se ve filtru postupně sčítá (integruje) zrychlení měřené akcelerometrem na rychlost a posléze na polohu (viz. kapitoly 2.1 a 2.2). Pokud tedy budeme měřit úhlovou rychlost, resp. zrychlení se systematickou chybou, tato se bude také postupně sčítat (integrovat), což způsobí s časem rostoucí chybu v určení matice přechodu, resp. rychlosti a polohy při predikci.

Měřené zrychlení tedy podle předpokladů bude

$$\boldsymbol{a}_{\text{měř}} = \boldsymbol{a} + \boldsymbol{w}_a + \boldsymbol{b}_a, \tag{3.59}$$

kde $\boldsymbol{a}_{\text{měř}}$ je zrychlení měřené, $\boldsymbol{a}$ je zrychlení skutečné, $\boldsymbol{w}_a$ je náhodná chyba (bílý šum) s nulovou střední hodnotou a $\boldsymbol{b}_a$ je systematická chyba akcelerometru. Chyby měření akcelerometru jsou znázorněny na obrázku 3.3, kde je modře měřené zrychlení a_{mer}, červeně skutečné zrychlení a a žlutě posunuté zrychlení a_b o systematickou chybu b_a.



Obrázek 3.3: Chyby akcelerometru při měření zrychlení.

Ve skutečnosti se pak ještě systematická chyba může v čase měnit, takže bude funkcí času

$$\boldsymbol{b}_a = \boldsymbol{b}_a(t), \tag{3.60}$$

předpokládáme ovšem, že tato změna je pomalá, takže na dostatečně krátkém časovém intervalu se dá aproximovat konstantním vektorem $\boldsymbol{b}_a$. Jestliže budeme podle (2.3) ze zrychlení postupně dělat predikce rychlosti, dostaneme po n krocích predikci rychlosti

$$\tilde{\boldsymbol{v}}_{k+n} = \hat{\boldsymbol{v}}_k + \sum_{i=k}^{k+n-1} \boldsymbol{a}_i \Delta t_i \,. \tag{3.61}$$

Po dosazení (3.59) za zrychlení $\boldsymbol{a}_i$ vyjde

$$\tilde{\boldsymbol{v}}_{k+n} = \hat{\boldsymbol{v}}_k + \sum_{i=k}^{k+n-1} (\boldsymbol{a}_i + \boldsymbol{w}_a + \boldsymbol{b}_a)\Delta t_i$$

$$= \hat{\boldsymbol{v}}_k + \sum_{i=k}^{k+n-1} \boldsymbol{a}_i \Delta t_i + \sum_{i=k}^{k+n-1} \boldsymbol{w}_a \Delta t_i + \sum_{i=k}^{k+n-1} \boldsymbol{b}_a \Delta t_i \,. \tag{3.62}$$

Jestliže budou časové kroky stejné $\Delta t_i = \Delta t$, tak se výraz dále zjednoduší na

$$\tilde{\boldsymbol{v}}_{k+n} = \hat{\boldsymbol{v}}_k + \Delta t \sum_{i=k}^{k+n-1} \boldsymbol{a}_i + \Delta t \sum_{i=k}^{k+n-1} \boldsymbol{w}_a + \Delta t \sum_{i=k}^{k+n-1} \boldsymbol{b}_a \,, \tag{3.63}$$

kde první suma přes zrychlení $\boldsymbol{a}_i$ představuje skutečnou změnu rychlosti (je v ní ovšem zahrnuta chyba diskretizace). U náhodné chyby předpokládáme její nulovou střední hodnotu, a dá se tedy předpokládat, že druhá suma přes $\boldsymbol{w}_a$ se s rostoucím n bude blížit nule. V poslední sumě sčítáme konstantní systematickou chybu $\boldsymbol{b}_a$

$$\Delta t \sum_{i=k}^{k+n-1} \boldsymbol{b}_a = \boldsymbol{b}_a \Delta t \cdot n, \tag{3.64}$$

chyba predikce $\tilde{\mathbf{v}}_{k+n}$ vlivem systematické chyby akcelerometru tedy bude lineárně narůstat s časem.

Systematické chyby akcelerometru a gyroskopu můžeme určit například, když senzory stojí v klidu, tedy když se sledovaný objekt nepohybuje. V takovém případě by gyroskop měl měřit nulovou hodnotu úhlové rychlosti a akcelerometr pouze tíhové zrychlení g. Nejprve je tedy potřeba zjistit, kdy se sledovaný objekt nepohybuje. Ve filtru je sledováno časové okno posledních n měření rychlosti z GNSS přijímače a zjišťuje se, jestli objekt během tohoto časového okna stál na místě, k tomu jsou použita dvě kritéria, která musí platit současně:

$$\left\| \frac{1}{n} \sum_{i=k-n+1}^{k} \boldsymbol{v}_i \right\| < v_0, \tag{3.65}$$

$$\left\| \begin{pmatrix} \sigma_n^2(v_x) \\ \sigma_n^2(v_x) \\ \sigma_n^2(v_z) \end{pmatrix} \right\| < \sigma_0, \tag{3.66}$$

kde první kritérium říká, že norma z průměrné rychlosti v testovaném časovém okně má být menší než v_0, které si volíme. V druhém kritériu se spočítají rozptyly jednotlivých složek rychlosti v daném časovém okně a norma jejich vektoru musí být menší než σ_0, které si opět volíme. Když v daném časovém okně měřená rychlost z přijímače GNSS vyhoví oběma kritériím, tak filtr vyhodnotí, že objekt stojí. V takovém případě se spočítá průměrné zrychlení a úhlová rychlost v daném časovém okně

$$\bar{\boldsymbol{a}}_n = \frac{1}{n} \sum_{i=k-n+1}^{k} \boldsymbol{a}_i \,, \tag{3.67}$$

$$\bar{\boldsymbol{\omega}}_n = \frac{1}{n} \sum_{i=k-n+1}^{k} \boldsymbol{\omega}_i \,, \tag{3.68}$$

kde n je délka časového okna. Od průměrného zrychlení musíme stejně jako v (3.33) odečíst tíhové zrychlení g ve třetí složce

$$\overline{\boldsymbol{a}}_n^{komp} = \overline{\boldsymbol{a}}_n - \begin{pmatrix} 0 \\ 0 \\ g \end{pmatrix}. \tag{3.69}$$

Podle předpokladu má měřené zrychlení a obdobně i úhlová rychlost tvar (3.59), který se skládá ze skutečné hodnoty, náhodné chyby a systematické chyby. Když dosadíme tento tvar do výpočtu průměrného zrychlení (3.67), dostaneme

$$\begin{aligned} \overline{\boldsymbol{a}}_n &= \frac{1}{n} \sum_{i=k-n+1}^{k} (\boldsymbol{a}_i + \boldsymbol{w}_a + \boldsymbol{b}_a) \\ &= \frac{1}{n} \sum_{i=k-n+1}^{k} \boldsymbol{a}_i + \frac{1}{n} \sum_{i=k-n+1}^{k} \boldsymbol{w}_a + \frac{1}{n} \sum_{i=k-n+1}^{k} \boldsymbol{b}_a . \end{aligned} \tag{3.70}$$

U náhodné chyby předpokládáme nulovou střední hodnotu, takže prostřední suma přes náhodnou chybu bude pro dostatečně vysoké n malá. Pro dostatečně dlouhé časové okno, kdy sledovaný objekt stojí, tedy průměrné zrychlení aproximujeme

$$\overline{\boldsymbol{a}}_n \approx \frac{1}{n} \sum_{i=k-n+1}^{k} \boldsymbol{a}_i + \frac{1}{n} \sum_{i=k-n+1}^{k} \boldsymbol{b}_a . \tag{3.71}$$

Předpokládáme, že v rámci jednoho časového okna je systematická chyba akcelerometru $\boldsymbol{b}_a$ konstantní. Když výraz v (3.71) upravíme, dostaneme

$$\frac{1}{n} \sum_{i=k-n+1}^{k} \boldsymbol{a}_i + \frac{\boldsymbol{b}_a n}{n} = \frac{1}{n} \sum_{i=k-n+1}^{k} \boldsymbol{a}_i + \boldsymbol{b}_a. \tag{3.72}$$

Dostali jsme tedy aproximovaný vztah pro průměrné zrychlení

$$\overline{\boldsymbol{a}}_n \approx \frac{1}{n} \sum_{i=k-n+1}^{k} \boldsymbol{a}_i + \boldsymbol{b}_a, \tag{3.73}$$

protože ovšem předpokládáme, že v tomto časovém okně objekt stojí na místě, skutečná zrychlení $\boldsymbol{a}_i$ budou nulová, a tedy i suma přes ně bude nulová. Průměrné zrychlení, které jsme spočítali v (3.69), tak považujeme přímo za systematickou chybu akcelerometru

$$\boldsymbol{b}_a \approx \overline{\boldsymbol{a}}_n^{komp}. \tag{3.74}$$

Obdobně postupujeme i pro úhlovou rychlost a její průměrnou hodnotu v daném časovém okně považujeme za systematickou chybu gyroskopu

$$\boldsymbol{b}_\omega \approx \overline{\boldsymbol{\omega}}_n. \tag{3.75}$$

Systematické chyby jsou pak ještě filtrovány váženým průměrem s jejich předchozím odhadem

$$\widehat{\boldsymbol{b}}_a[l] = (1-m) \cdot \widehat{\boldsymbol{b}}_a[l-1] + m \cdot \boldsymbol{b}_a[l], \tag{3.76}$$

kde $\widehat{\boldsymbol{b}}_a[l]$ značí filtrovaný odhad systematické chyby akcelerometru po l-té kalibraci, $\boldsymbol{b}_a[l]$ je určeno v l-té kalibraci podle (3.74) a m je váha, se kterou započítáváme nově určené $\boldsymbol{b}_a[l]$. Pro váhu musí

platit $m \in \langle 0,1 \rangle$ a ve filtru ji nastavujeme tím větší, čím je starší poslední kalibrace. Stejně filtrujeme i odhad systematické chyby gyroskopu $\boldsymbol{b}_\omega$. Když máme odhadnuté systematické chyby akcelerometru a gyroskopu, tak do jejich další kalibrace (než objekt na dostatečně dlouhou dobu znovu zastaví) kompenzujeme měřené zrychlení a úhlovou rychlost prostým odečtením těchto systematických chyb

$$\widehat{\boldsymbol{a}}_k = \boldsymbol{a}_k - \widehat{\boldsymbol{b}}_a, \tag{3.77}$$

$$\widehat{\boldsymbol{\omega}}_k = \boldsymbol{\omega}_k - \widehat{\boldsymbol{b}}_\omega, \tag{3.78}$$

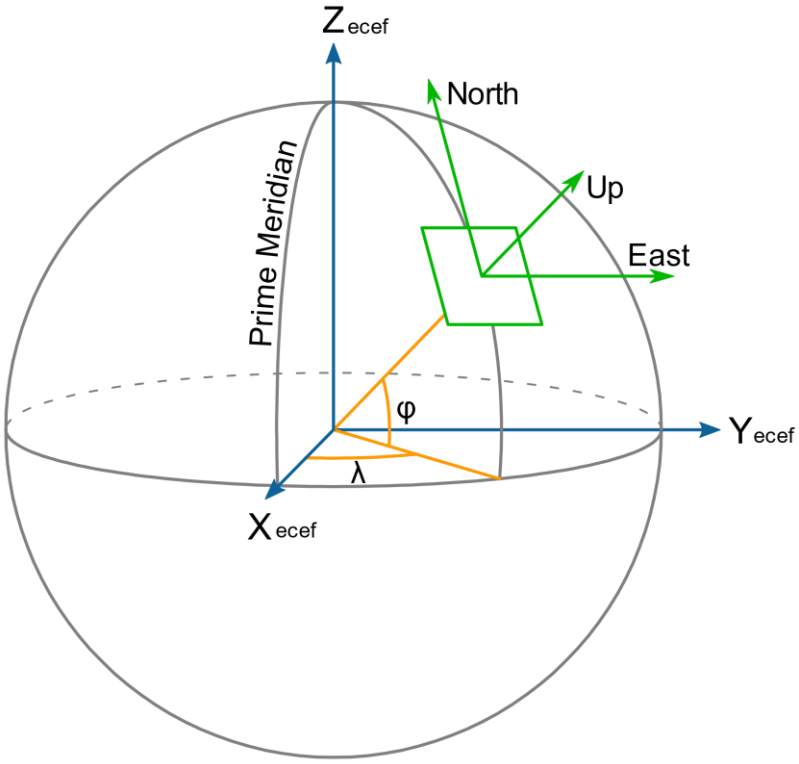
kde $\boldsymbol{\omega}_k$ a $\boldsymbol{a}_k$ jsou měřená úhlová rychlost a zrychlení a $\widehat{\boldsymbol{\omega}}_k$, $\widehat{\boldsymbol{a}}_k$ jsou nové kompenzované odhady těchto veličin, $\widehat{\boldsymbol{b}}_a$ a $\widehat{\boldsymbol{b}}_\omega$ jsou pak filtrované odhady systematických chyb akcelerometru a gyroskopu podle (3.76).

4 Testování algoritmu

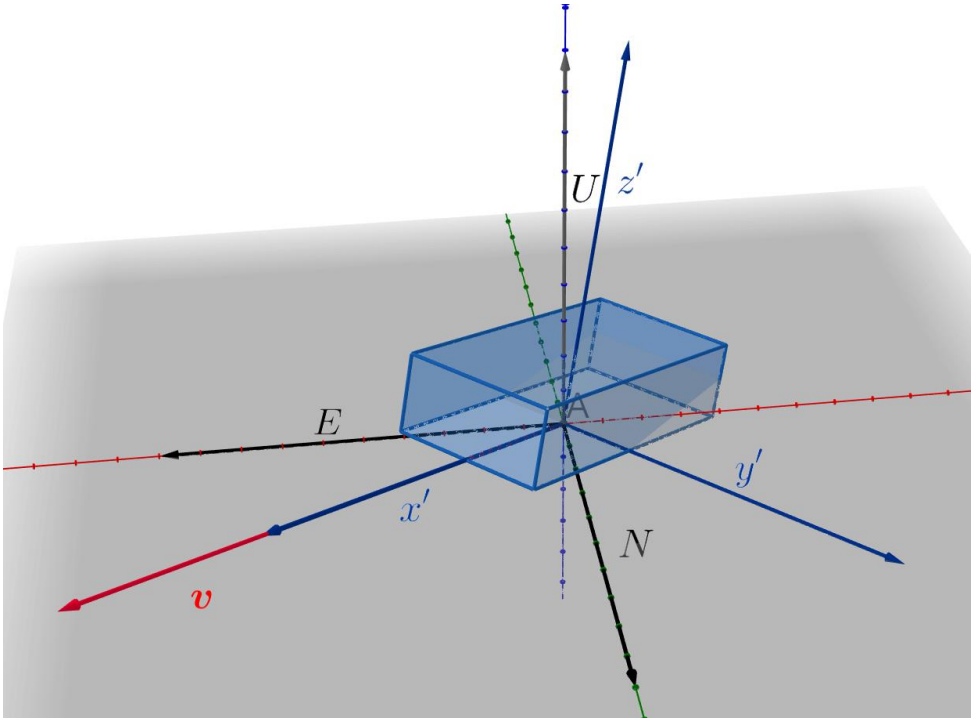
Navržený algoritmus, popsaný v kapitole 3.3, implementujeme v prostředí MATLAB. Na testování používáme data pro post-processing zachycená při dřívějším experimentu s navigačním demonstrátorem. Data byla naměřena při jízdě autem po Praze. Při experimentu byly naměřeny polohy a rychlosti z GPS přijímače Novatel OEM-628 (L1+L2) a pomocí upraveného programu RTKLIB 2.4.2 byl proveden RTK odhad polohy. Dále byly naměřeny zrychlení a úhlové rychlosti pomocí pohybového MEMS senzoru Silicon Sensing DMU10, který se skládá z 3-osého akcelerometru a 3-osého gyroskopu.

Frekvence měření polohy a rychlosti z přijímače GPS byla $10\ Hz$, což znamená časový krok $100\ ms$ mezi jednotlivými měřenými body. Reálné časové kroky se pak občas lišily, především při výpadku signálu přijímače dosahovaly až jednotek sekund. Pohybové senzory (akcelerometr a gyroskop) měřily s frekvencí $100\ Hz$, tedy 10krát rychleji než GPS přijímač, a časový krok mezi jejich měřeními byl tedy $10\ ms$. Mezi dvěma korekcemi z přijímače GPS tedy proběhlo většinou 10 predikcí pomocí pohybových senzorů.

Osy akcelerometru a gyroskopu (x', y', z') jsou nastaveny ve stejném směru, obě veličiny jsou měřeny v kartézské tělesové soustavě $\mathcal{X}'$, která je pevně spojena s autem. Osa x' je nasměrovaná ve směru pohybu auta (dopředu), osa y' míří doprava a osa z' dolů. Ve filtru pak používáme osy ve směrech dopředu, doleva a nahoru, takže stačí prohodit znaménka u směrů y' a z'. Polohy a rychlosti, které vstupují do filtru polohy, jsou v pevné kartézské soustavě ENU (East-North-Up – viz obrázek 4.1), kterou zde považujeme za přibližně inerciální. Vzájemná orientace obou soustav je vidět na obrázku 4.2.



Obrázek 4.1: Soustava souřadnic ENU (East-North-Up) [5].



Obrázek 4.2: Vzájemná orientace pevné soustavy ENU a tělesové $\mathcal{X}'$.

Soustavu, kterou jsme v minulých kapitolách značili jako $\mathcal{X}$, tedy zvolíme ENU a bude vztažena k prvnímu bodu trasy, který je počátkem této soustavy. Směry soustavy ENU jsou E – východ, N – sever, U – nahoru.

Testovaná trasa prochází centrem Prahy, od nábřeží Vltavy až do Dejvic do areálu FEL ČVUT, viz obrázek 4.3.

Obrázek 4.3: Testovací trasa centrem Prahy.

4.1 Vizualizace testovaných dat

Testovaná data, naměřené a filtrované polohy, jsou zobrazena v programu Google Earth Pro, který je volně dostupný ke stažení. Pro grafické zobrazení polohových dat v programu Google Earth je nutné vytvořit z dat soubor ve formátu KML. K tomu nám slouží volně dostupný toolbox pro MATLAB „Google Earth Toolbox“ [6]. Z toolboxu využíváme hlavně následující funkce:

- *ge_plot()*
- *ge_point()*
- *ge_output()*

Funkce *ge_plot()* má na vstupu polohy v souřadnicích WGS84, tedy zeměpisnou délku a šířku. Tato funkce nám vytvoří textový řetězec, který předáme na vstup do funkce *ge_output()* a ta nám vytvoří soubor KML, který už můžeme otevřít v programu Google Earth. Funkce *ge_plot()* dělá lomenou čáru postupně přes všechny body na vstupu, takže je dobře vidět časový průběh jednotlivých měřených bodů. Můžeme ještě také použít funkci *ge_point()*, která má stejně jako *ge_plot()* na vstupu polohy v souřadnicích WGS84, ale vytváří pouze značky v jednotlivých měřených bodech. I tato funkce má na výstupu textový řetězec, ze kterého nám funkce *ge_output()* udělá soubor KML. V toolboxu je pak i mnoho dalších užitečných funkcí, například pro grafické zobrazování vektorů (např. rychlosti, zrychlení atd.) jako šipek v mapě a dalších. Ve zdrojovém kódu 4.1 je příklad vytvoření trasy ve formátu KML z jednotlivých poloh.

```
%% export do Google Earthu
earthString = ge_plot(PolohyWGS_K(2,:), PolohyWGS_K(1,:), 'lineWidth', 2, ...
        'lineColor', sprintf('FF00%s%s', dec2hex(40, 2), dec2hex(100, 2)));
ge_output('cesta.kml', earthString);
```

Zdrojový kód 4.1: Vytvoření trasy ve formátu KML pomocí Google Earth Toolboxu.

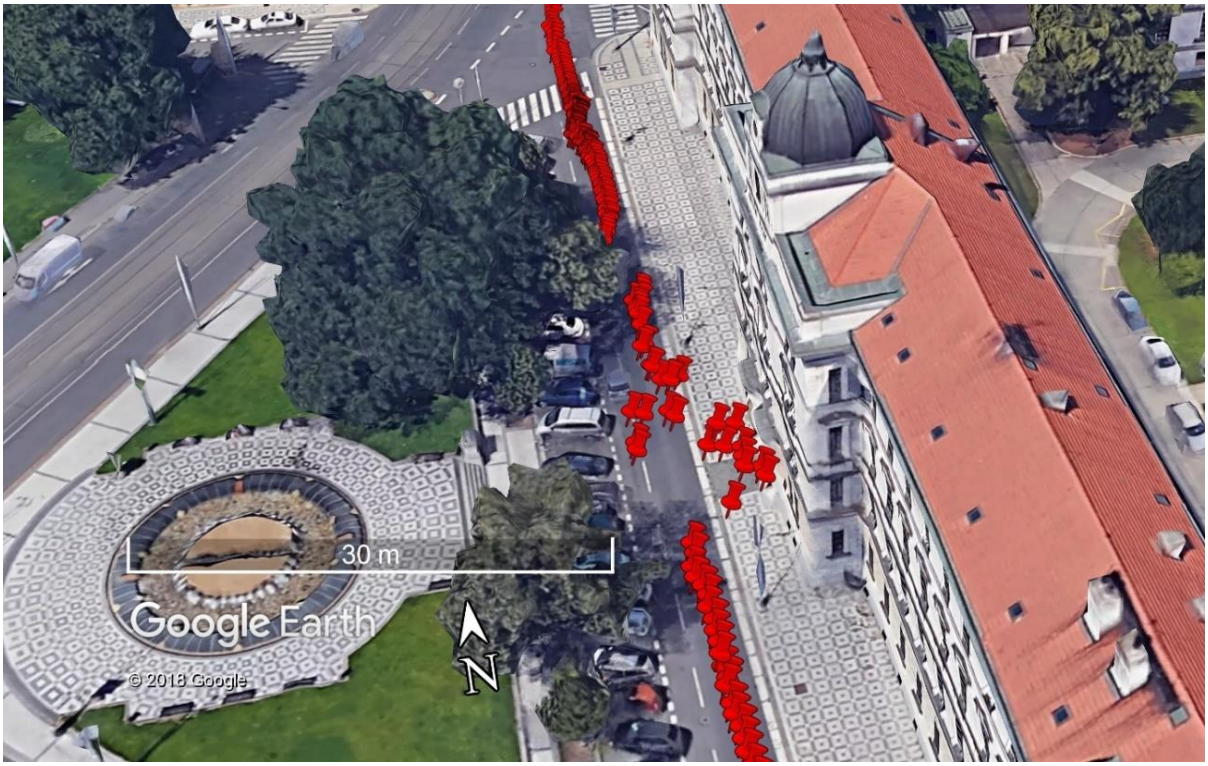
Funkce *ge_plot()* má dva povinné vstupní argumenty, a to jsou vektory obsahující zeměpisné délky a šířky jednotlivých bodů. Další argumenty funkce jsou volitelné a mění různé grafické parametry pro vykreslení trasy.

4.2 Chyby v určení polohy přijímačem GPS

V naměřených polohách přijímačem GPS se vyskytovalo několik druhů chyb. První a asi nejčastější chyba byla pouze nepřesnost určené polohy v podobě menších odchylek. V centru města to může být způsobeno například odrazem signálu od budov, který způsobuje mnohacestné šíření signálu, případně i zakrytí výhledu vegetací – různými stromy apod. Tyto chyby jsou dobře vidět na obrázcích 4.4 a 4.5. Na následujících obrázcích jsou červenými ikonkami znázorněny polohy měřené přijímačem GPS.

Dalším typem chyby je úplný výpadek přijímače GPS na určitou dobu v důsledku špatného signálu. Po tuto dobu tedy z přijímače GPS nedostáváme žádná data o poloze ani rychlosti a na obrázku 4.6 to vidíme jako mezery mezi řadami naměřených bodů.

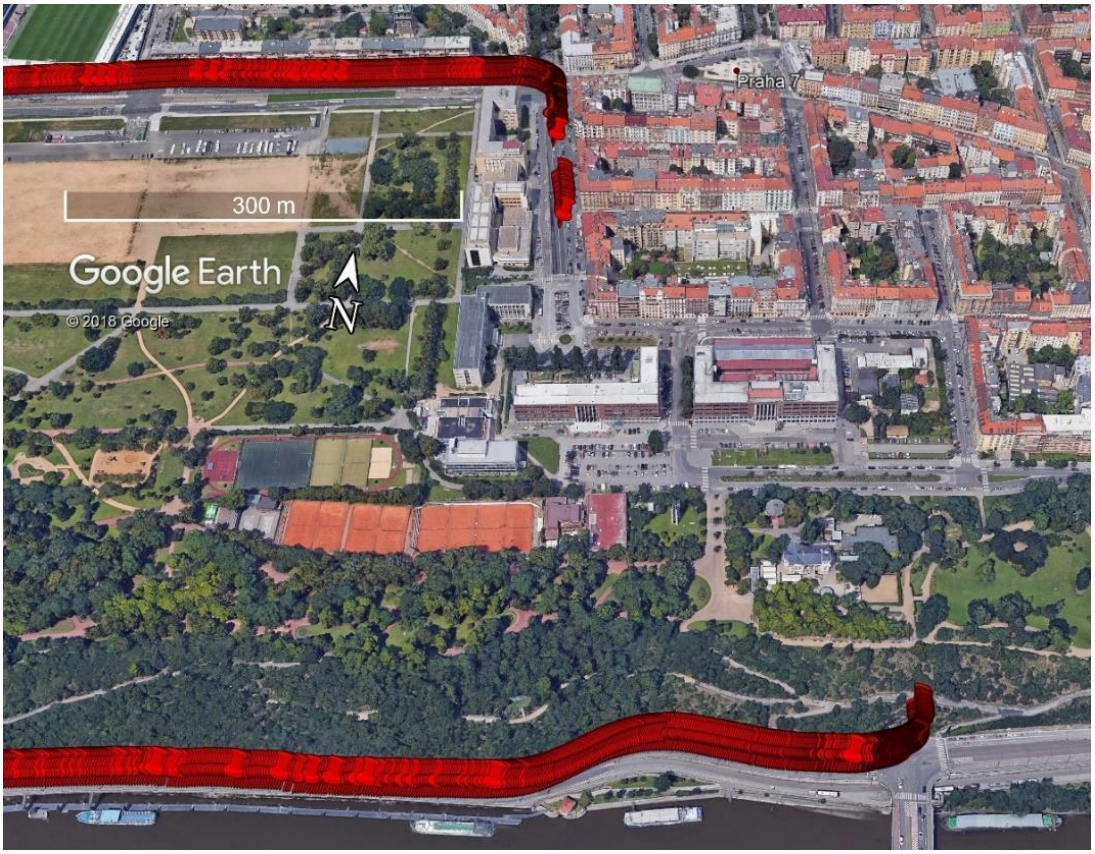
Zvláštním typem chyby je průjezd tunelem, kde přijímač GPS nemá žádný signál a my tak nemůžeme měřit polohu ani rychlost. Nejedná se v podstatě o chybu, ale přesto nám zde absence měření může vadit. Příklad vidíme na průjezdu Letenským tunelem na obrázku 4.7.



Obrázek 4.4: Nepřesné určení polohy přijímačem GPS.

Obrázek 4.5: Nepřesné určení polohy přijímačem GPS.

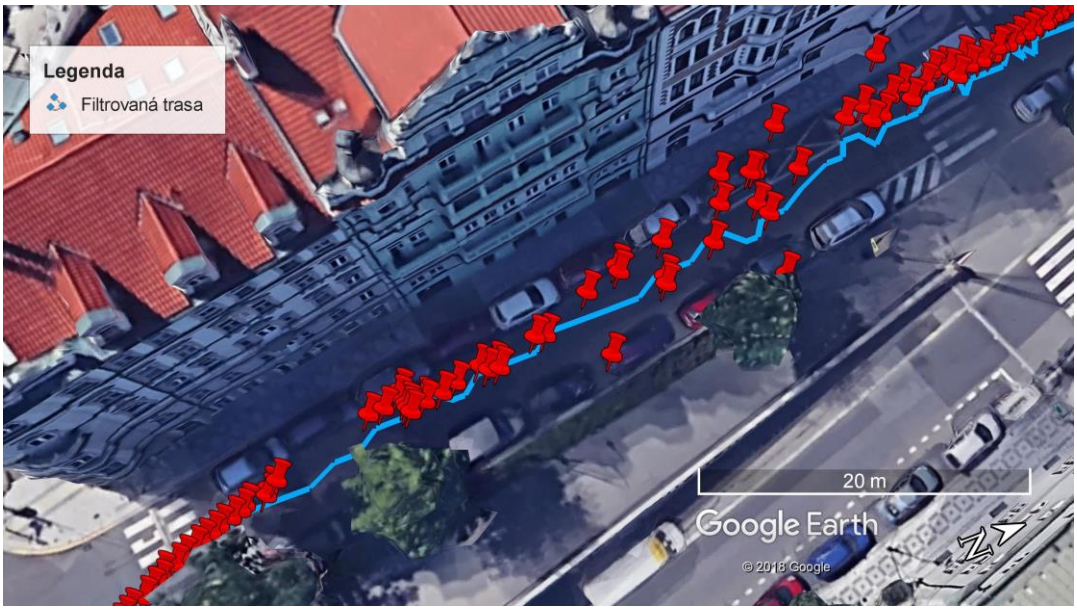
Obrázek 4.6: Výpadek signálu přijímače GPS.



Obrázek 4.7: Průjezd Letenským tunelem. Polohy jsou naměřené z přijímače GPS.

4.3 Výsledný odhad trasy

Výsledný odhad trasy, na který používáme filtr polohy popsaný v kapitole 3.3, se snaží jednak odfiltrovat menší odchylky měření polohy přijímačem GPS, což se daří docela dobře při správném odhadu rozptylu měřené polohy z přijímače, jak je vidět např. na obrázku 4.8, kde červené body představují naměřené polohy z přijímače GPS a modrá čára je filtrovaná trasa.



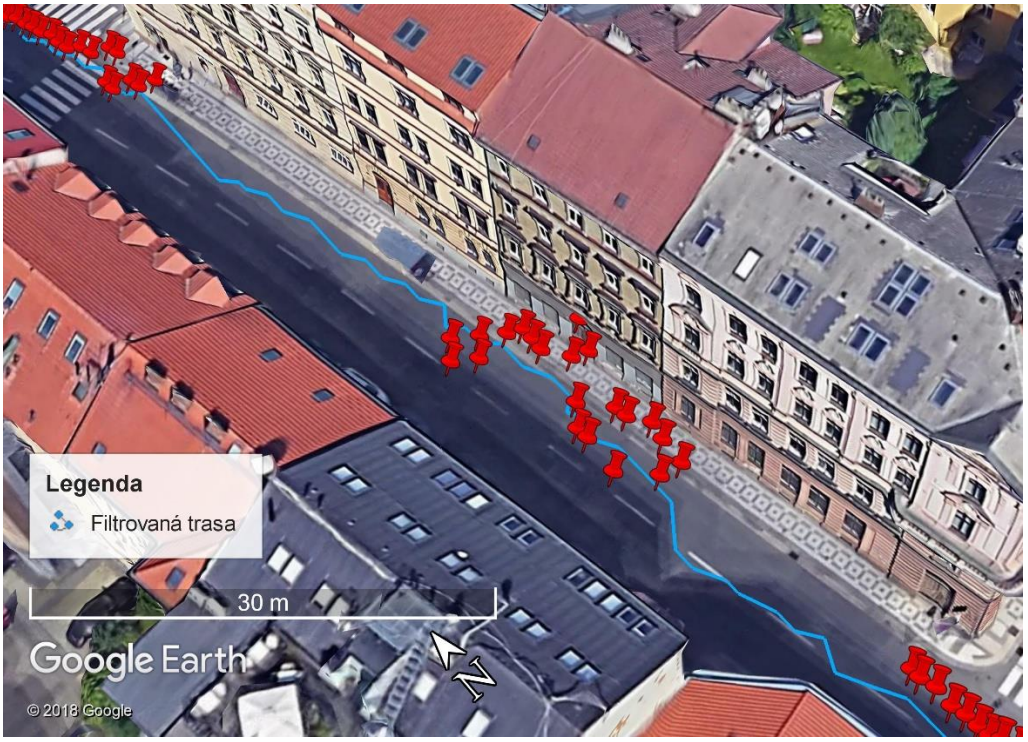
Obrázek 4.8: Korekce menších odchylek přijímače GPS pomocí filtru polohy.

Naproti tomu při špatně určeném rozptylu polohy přijímačem GPS se i filtrovaná trasa odchýlí od skutečnosti, což je vidět na obrázku 4.9.

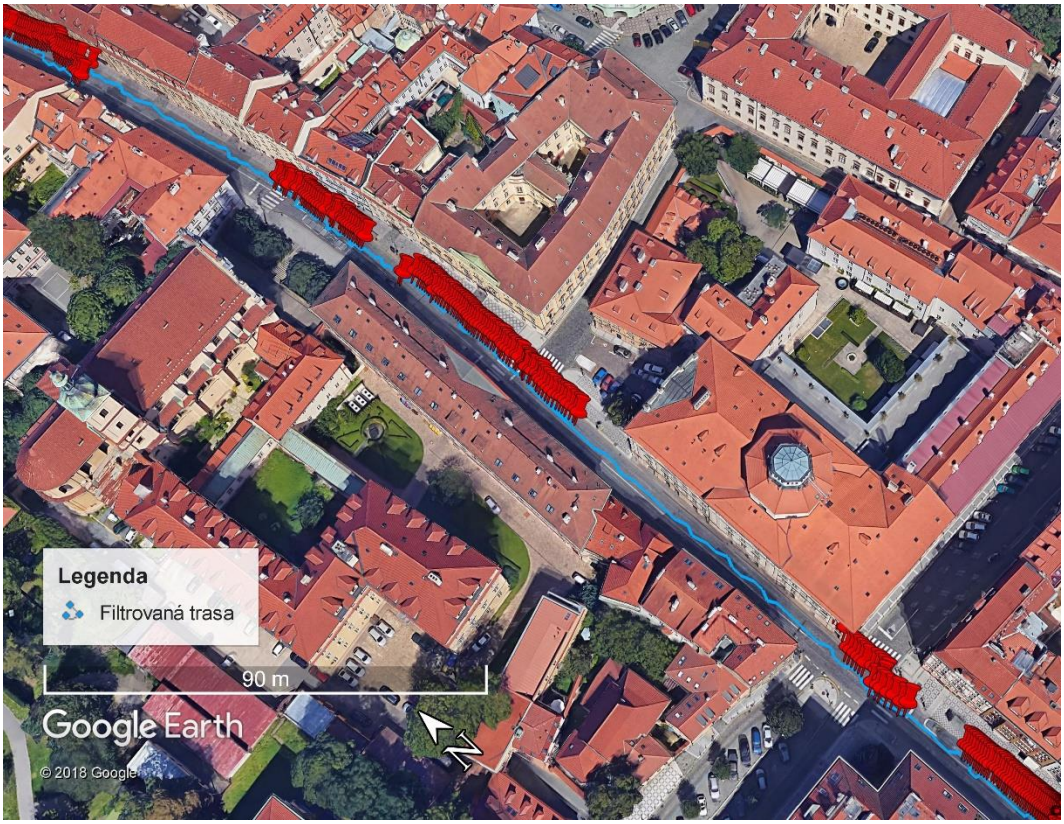
Obrázek 4.9: Odchylka korekce při špatně určeném rozptylu polohy z přijímače GPS.

V případě, kdy přijímač GPS po nějakou dobu neměří polohu vůbec, filtr odhaduje polohu dál pomocí predikce polohy a rychlosti, kterou počítá z předchozích stavů a měření akcelerometru a gyroskopu podle kapitoly 3.3.2. Příklady odhadů polohy při ztrátě signálu GPS přijímače jsou na obrázcích 4.10, 4.11 a 4.12, kde červenými značkami jsou znázorněny polohy měřené přijímačem GPS a modrou čárou filtrovaný odhad trajektorie.

Obrázek 4.10: Odhad polohy při výpadku signálu přijímače GPS.

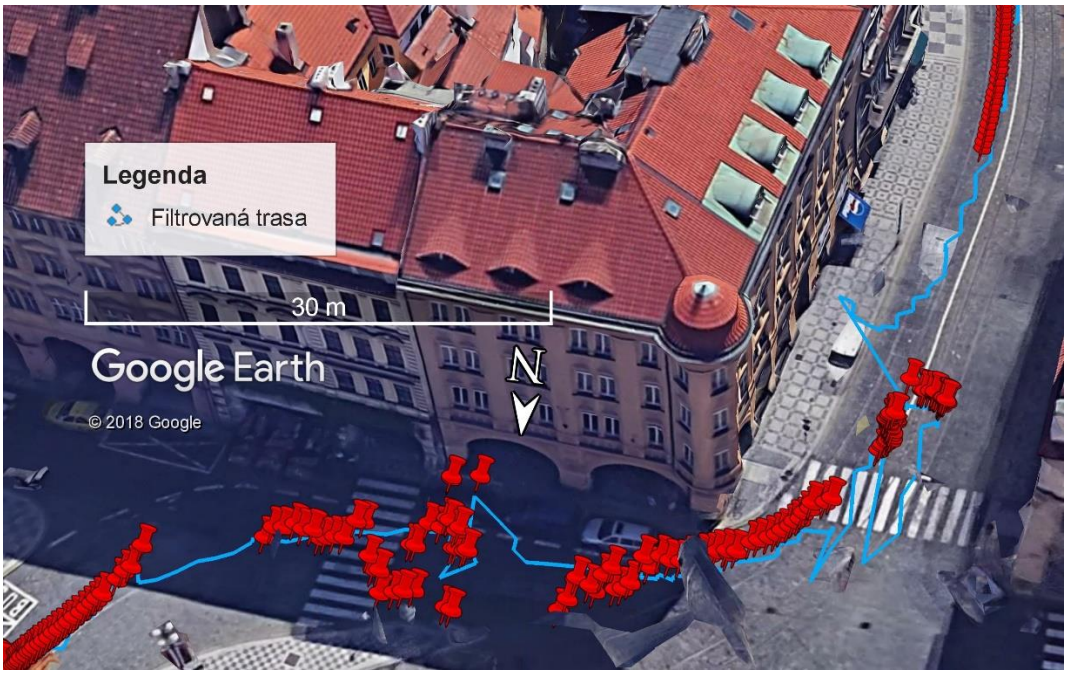


Obrázek 4.11: Odhad polohy při výpadku signálu přijímače GPS.



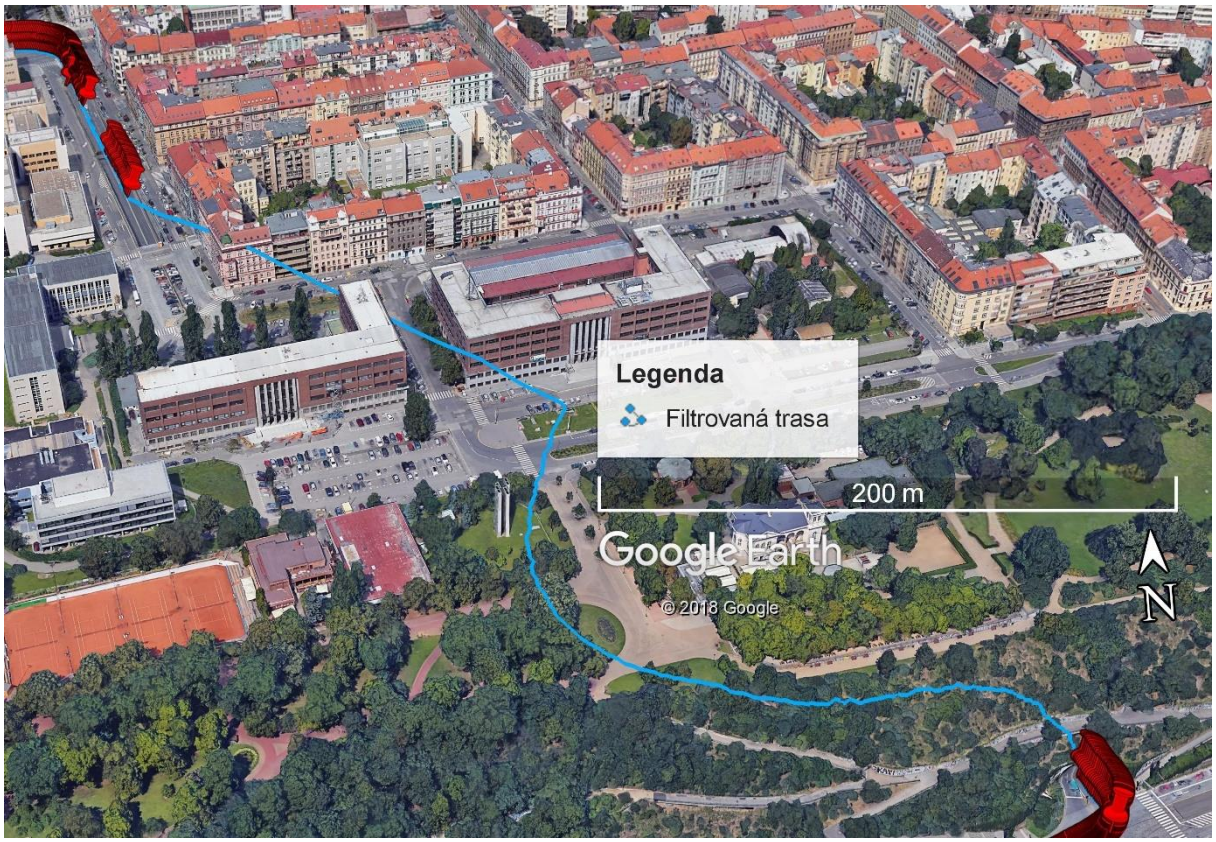
Obrázek 4.12: Odhad polohy při výpadku signálu přijímače GPS.

V některých případech se trajektorie odhadovaných poloh filtrem postupně chybně stáčí, jak je vidět na obrázku 4.13.



Obrázek 4.13: Stáčení trajektorie odhadnuté filtrem.

Speciální případ je odhad trajektorie při průjezdu Letenským tunelem, kde přijímač GPS hned na začátku ztratil signál a nemáme z něho tak žádné naměřené polohy. Výsledná filtrovaná trajektorie je vidět na obrázku 4.14, kde červené značky značí měřené polohy přijímačem GPS a modrá čára filtrovanou trajektorii. Jak je z obrázku vidět, výsledný odhad polohy v tunelu, vytvořený jako predikce pomocí pohybových senzorů, není moc přesný, což může být dané např. špatně provedenou poslední kalibrací akcelerometru a gyroskopu.



Obrázek 4.14: Filtrovaný odhad trajektorie při průjezdu Letenským tunelem.

Závěr

V práci je navržen algoritmus fúze dat několika měření polohy z různých zdrojů, např. různých systémů GNSS, a algoritmus pro podporu družicové navigace pomocí pohybových senzorů. Algoritmus pro podporu družicové navigace (filtr polohy) je postaven na základě „open-loop“ verze Kalmanova filtru. Vytváří predikce polohy pomocí pohybových senzorů (akcelerometru a gyroskopu) a provádí jejich korekci pomocí měřené polohy a rychlosti z přijímače GNSS.

Algoritmus pro podporu družicové navigace byl otestován na datech určených pro post-processing zachycených při dřívějším experimentu s navigačním demonstrátorem. Data pochází z jízdy autem po Praze. Polohy a rychlosti byly měřené GPS přijímačem Novatel OEM-628 (L1+L2) a pomocí upraveného programu RTKLIB 2.4.2 byl proveden RTK odhad polohy. Dále bylo měřeno zrychlení a úhlové rychlosti pomocí pohybového MEMS senzoru Silicon Sensing DMU10, který se skládá z 3-osého akcelerometru a 3-osého gyroskopu. Filtr je navržen a simulován tak, že používá pouze ta data, která by v daném časovém kroku měl k dispozici i v případném reálném provozu. Znamená to, že predikce a korekce jdou postupně za sebou podle pořadí, v jakém byly měřeny jednotlivé veličiny přijímačem GPS a pohybovými senzory.

Algoritmus opravil menší odchylky při měření polohy, pokud byl dobře odhadnutý rozptyl naměřených poloh z přijímače GPS. Pokud byl rozptyl odhadnut špatně, algoritmus se více přiklonil k nepřesně měřené poloze z přijímače GPS. Filtrovaný odhad trasy pak celkem dobře doplnil místa, kde přijímač GPS neměl dostatečný signál, a neměřil polohu ani rychlost vůbec. Občas se ovšem v takových případech trajektorie predikce, vytvářené pomocí měřeného zrychlení a úhlové rychlosti z akcelerometru a gyroskopu, začala postupně chybně stáčet ze skutečné trasy. Stejná chyba nastala při dlouhém výpadku přijímače GPS během průjezdu Letenským tunelem. Toto mohlo být způsobeno špatným odhadem systematických chyb akcelerometru a gyroskopu při jejich poslední kalibraci, případně špatnou korekcí matice přechodu pomocí rychlosti těsně před vjezdem do tunelu.

Navržený filtr by bylo vhodné dále testovat a ladit, a to především na určitých typických scénářích, kde nastávají hlavní potenciální problémy. Jedná se například o různé průjezdy zatáček s velkou změnou směru pohybu, případně dlouhé výpadky přijímače družicové navigace (např. při průjezdu tunelem). Filtr by pak bylo vhodné dále doplnit o detekce některých chyb, které se při testování projevovaly. Např. by bylo dobré detekovat špatně odhadnuté rozptyly polohy přijímačem GPS, případně špatně odhadnuté systematické chyby akcelerometru a gyroskopu. Pozornost je pak třeba věnovat ještě algoritmu pro odhad matice přechodu, která nám slouží především pro převod měřeného zrychlení, protože její špatný odhad vnáší chyby i do tohoto zrychlení, a tím pak i do predikce polohy a rychlosti.

Reference

[1] A. Janota, V. Šimák, D. Nemec a J. Hrbček, „Improving the Precision and Speed of Euler Angles Computation from Low-Cost Rotation Sensor Data,“ *Sensors,* pp. 7016-7039, 23 Březen 2015.

[2] M. Červenka, „Zpracování fyzikálních měření,“ 23 Leden 2013. [Online]. Available: http://fyzika.feld.cvut.cz/~cervenka/vyuka/zprfm/zpracdat.pdf. [Přístup získán 28 Březen 2018].

[3] P. D. Joseph, „The one dimensional Kalman Filter,“ [Online]. Available: www.eeng.dcu.ie/~ee502/Kalman_1Lessons_0_to_4.doc. [Přístup získán 28 Březen 2018].

[4] P. D. Groves, Principles of GNSS, Inertial, and Multisensor Integrated Navigation Systems, Artech House, 2008.

[5] Wikimedia, „Wikimedia,“ 18 Únor 2010. [Online]. Available: https://upload.wikimedia.org/wikipedia/commons/7/73/ECEF_ENU_Longitude_Latitude_relationships.svg. [Přístup získán 7 Květen 2018].

[6] S. L. D. Jurriaan H. Spaaks, „Google Earth Toolbox,“ 2 Únor 2012. [Online]. Available: https://www.mathworks.com/matlabcentral/fileexchange/12954-google-earth-toolbox. [Přístup získán 7 Květen 2018].

Přílohy

A. Filtr polohy v MATLABu

```
function [ PolohyENU_K, RychlostiK, ZrychleniENU] =
kalmanFiltr( PolohyENU, RozptylyENU, id, RychlostiENU, faktorRozRych, ...
    ZrychleniXYZ, rozptylAkcel, UhlovyRychlosti, casKroky, pruchod )
%   Filtr filtruje polohy pomoci predikce, kterou pocita z aktualni rychlosti,
%   zrychleni a uhlove rychlosti. Zrychleni a uhlova rychlost jsou
%   v soustave XYZ, ktera je spojena s vozidlem. Polohy a rychlosti jsou v
%   soustave ENU.
%
%   X - dopredu ve smeru jizdy (aktualni rychlosti)
%   Y - doleva
%   Z - nahoru
%
%   Vstupy:
%   PolohyENU - matice 3 x N -> vstupni polohy naskladane vedle sebe
%   RozptylyENU - matice 3 x N -> rozptyly vstupnich poloh vedle sebe
%   id - vektor N x 1 -> index, oznacuje poradi mereni akcelerometru a
%   gyroskopu
%   RychlostiENU - matice 3 x N -> vstupni rychlosti naskladane vedle sebe
%   faktorRozRych -> ladici konst. pro rozptyl rychlosti
%   ZrychleniXYZ - matice 3 x M -> vstupni zrychleni naskladane vedle sebe
%   rozptylAkcel -> ladici konst. pro rozptyl zrychleni
%   UhlovyRychlosti - matice 3 x M -> vstupni uhlove rychlosti naskladane
%   vedle sebe
%   casKroky - vektor M-1 x 1 -> casove kroky mezi jednotlivymi merenimi
%   akcelerometru a gyroskopu (mezi dvema sousednimi indexy 'id')
%   pruchod -> pomocny index, neni defaultne pouzit
%
%   Vystupy:
%   PolohyENU_K - matice 3 x N -> vystupni polohy naskladane vedle sebe
%   RychlostiK - matice 3 x N -> vystupni rychlosti naskladane vedle sebe
%   ZrychleniENU - matice 3 x M -> vystupni zrychleni v soustave ENU
%   naskladane vedle sebe

%% konstanty
g = 9.80665; % tihove zrychleni g

%% definice promennych
RychlostiK = zeros(3, length(RychlostiENU));
RychlostiK(:, 1) = RychlostiENU(:, 1);

PolohyENU_K = zeros(3, 2e4);
PolohyENU_K(:, 1) = PolohyENU(:, 1);
pocetPoloh = 1;

ZrychleniENU = zeros(3, length(ZrychleniXYZ));

%% pocatecni nastaveni
% promenny Kalmanova filtru
stav = [PolohyENU(:, 1); RychlostiENU(:, 1)];
kovarianceStavu = diag([RozptylyENU(:, 1); faktorRozRych*RozptylyENU(:, 1)]);

% zjistuje pocatecni orientaci objektu - zanedbava vlastni rotaci
% pocatecni odhad je proveden ze smeru vektoru rychlosti
cosFi = RychlostiK(1, 1) / norm(RychlostiK(1:2, 1));
sinFi = RychlostiK(2, 1) / norm(RychlostiK(1:2, 1));
cosTheta = norm(RychlostiK(1:2, 1)) / norm(RychlostiK(:, 1));
```

```
sinTheta = RychlostiK(3, 1) / norm(RychlostiK(:, 1));
RotZ = [cosFi -sinFi 0; sinFi cosFi 0; 0 0 1];
RotY = [cosTheta 0 -sinTheta; 0 1 0; sinTheta 0 cosTheta];
% matice prechodu z rotujici baze XYZ spojeny s vozidlem do ENU
maticePrechodu = RotZ * RotY;

% kalibrace gyra podle zacatku, kdy auto stoji
offsetGyro = mean(UhlovyRychlosti(:, 1:2000), 2);
UhlovyRychlosti = UhlovyRychlosti - offsetGyro;
offsetAkcel = mean(ZrychleniXYZ(:, 1:2000), 2);
offsetAkcel(3) = offsetAkcel(3) - g;
ZrychleniXYZ = ZrychleniXYZ - offsetAkcel;
stariKalibrace = 0;

%% hlavni cyklus - krokuje po jednotlivych merenich z GPS
for k = 1 : length(PolohyENU)-1
    % vnitrni cyklus - krokuje po jednotlivych merenich akcelerometru a
    % gyroskopu
    for m = id(k) : id(k+1)-1
        stariKalibrace = stariKalibrace + 1;
        % rotace zrychleni: XYZ -> ENU
        zrychleniENU = maticePrechodu * ZrychleniXYZ(:, m);
        kovarianceZrychleni = maticePrechodu * (rozptylAkcel*eye(3)) * ...
maticePrechodu';
        % korekce gravitace
        zrychleniENU(3) = zrychleniENU(3) - g;
        % uklada vystupni zrychleni
        ZrychleniENU(:, m) = zrychleniENU;

        %% predikce stavu a jeho chyby
        maticePredikce = eye(6);
        maticePredikce(1:3, 4:6) = eye(3) * casKroky(m);
        zmenaRychlosti = [0; 0; 0; zrychleniENU] * casKroky(m);

        stav = maticePredikce * stav + zmenaRychlosti;
        kovarianceStavu = maticePredikce * kovarianceStavu * maticePredikce';
        kovarianceStavu(4:6, 4:6) = kovarianceStavu(4:6, 4:6) + ...
casKroky(m)^2 * kovarianceZrychleni;

        % ukladam jen kazdou 20. polohu z predikce
        if mod(m - id(k) + 1, 20) == 0
            pocetPoloh = pocetPoloh + 1;
            PolohyENU_K(:, pocetPoloh) = stav(1:3);
        end

        %% urceni matice prechodu XYZ -> ENU
        uhlovaRychlost = UhlovyRychlosti(:, m);
        % parametry rotace z uhlove rychlosti
        uhelRotace = norm(uhlovaRychlost) * casKroky(m);
        uhlovaRychlost = uhlovaRychlost / norm(uhlovaRychlost);
        cosRot = cos(uhelRotace);
        sinRot = sin(uhelRotace);

        % urceni rotace baze mezi dvema kroky - z uhlove rychlosti
        rotaceBaze = [
            cosRot + uhlovaRychlost(1)^2*(1-cosRot), ...
```

```
            uhlovaRychlost(1)*uhlovaRychlost(2)*(1-cosRot) - ...
uhlovaRychlost(3)*sinRot, ...
            uhlovaRychlost(1)*uhlovaRychlost(3)*(1-cosRot) + ...
uhlovaRychlost(2)*sinRot;
            uhlovaRychlost(1)*uhlovaRychlost(2)*(1-cosRot) + ...
uhlovaRychlost(3)*sinRot, ...
            cosRot + uhlovaRychlost(2)^2*(1-cosRot), ...
            uhlovaRychlost(2)*uhlovaRychlost(3)*(1-cosRot) - ...
uhlovaRychlost(1)*sinRot;
            uhlovaRychlost(1)*uhlovaRychlost(3)*(1-cosRot) - ...
uhlovaRychlost(2)*sinRot, ...
            uhlovaRychlost(2)*uhlovaRychlost(3)*(1-cosRot) + ...
uhlovaRychlost(1)*sinRot, ...
            cosRot + uhlovaRychlost(3)^2*(1-cosRot)
            ];

        % update matice prechodu do dalsiho kroku z rotace baze
        maticePrechodu = maticePrechodu * rotaceBaze;
    end

    %% korekce stavu
    kovarianceMereni = diag([RozptylyENU(:, k+1); faktorRozRych*RozptylyENU(:,
k+1)]);
    kalmanZisk = kovarianceStavu * inv(kovarianceStavu + kovarianceMereni);
    mereni = [PolohyENU(:, k+1); RychlostiENU(:, k+1)];

    stav = stav + kalmanZisk * (mereni - stav);

    % prej numericky nestabilni
    kovarianceStavu = (eye(6) - kalmanZisk) * kovarianceStavu;

    pocetPoloh = pocetPoloh + 1;
    PolohyENU_K(:, pocetPoloh) = stav(1:3);
    RychlostiK(:, k+1) = stav(4:6);

    %% prubezna kalibrace akcelerometru a gyroskopu
    if k > 500
        if norm(mean(RychlostiENU(:, k-9:k+1), 2)) < 0.01
            if norm(var(RychlostiENU(:, k-9:k+1), 0, 2)) < 0.005
                if stariKalibrace > 1e4
                    stariKalibrace = 1e4; % kdyz je stari kalibrace 1e4, tak
uz zustava stejny
                end
                offsetGyro = mean(UhlovyRychlosti(:, id(k-9):id(k+1)), 2);
                UhlovyRychlosti(:, id(k+1):end) = UhlovyRychlosti(:,
id(k+1):end) - stariKalibrace/1e4*offsetGyro;
                offsetAkcel = mean(ZrychleniXYZ(:, id(k-9):id(k+1)), 2);
                offsetAkcel(3) = offsetAkcel(3) - g;
                ZrychleniXYZ(:, id(k+1):end) = ZrychleniXYZ(:, id(k+1):end) -
stariKalibrace/1e4*offsetAkcel;
                stariKalibrace = 0;
            end
        end
    end
```

```
    %% filtr matice prechodu XYZ -> ENU
    % zjistuje orientaci objektu - zanedbava vlastni rotaci
    % odhad je proveden ze smeru vektoru rychlosti
    cosFi = RychlostiK(1, k+1) / norm(RychlostiK(1:2, k+1));
    sinFi = RychlostiK(2, k+1) / norm(RychlostiK(1:2, k+1));
    cosTheta = norm(RychlostiK(1:2, k+1)) / norm(RychlostiK(:, k+1));
    sinTheta = RychlostiK(3, k+1) / norm(RychlostiK(:, k+1));
    RotZ = [cosFi -sinFi 0; sinFi cosFi 0; 0 0 1];
    RotY = [cosTheta 0 -sinTheta; 0 1 0; sinTheta 0 cosTheta];
    % matice prechodu z rotujici baze XYZ spojeny s vozidlem do ENU na
    % zaklade vektoru rychlosti
    RozptylV_K = kovarianceStavu(4:6, 4:6);

    if mod(k, 10) == 0 && norm(RozptylV_K) < 1
        maticePrechodu = RotZ*RotY; % pri kazdy 10. korekci se resetuje matice
prechodu
    else
        % vazeny prumer matice prechodu z uhlove rychlosti a z vektoru
        % rychlosti
        maticePrechodu = (maticePrechodu + RotZ*RotY*norm(RozptylV_K)*0.008) /
(norm(RozptylV_K)*0.008 + 1);

        %% normovavni matice -> aby byla ciste rotacni
        e_xy = maticePrechodu(:, 1)' * maticePrechodu(:, 2);
        e_yz = maticePrechodu(:, 2)' * maticePrechodu(:, 3);
        e_zx = maticePrechodu(:, 3)' * maticePrechodu(:, 1);

        X = maticePrechodu(:, 1) - e_xy/2 * maticePrechodu(:, 2)...
            - e_zx/2 * maticePrechodu(:, 3);
        Y = maticePrechodu(:, 2) - e_xy/2 * maticePrechodu(:, 1)...
            - e_yz/2 * maticePrechodu(:, 3);
        Z = maticePrechodu(:, 3) - e_zx/2 * maticePrechodu(:, 1)...
            - e_yz/2 * maticePrechodu(:, 2);

        maticePrechodu = [X/norm(X) Y/norm(Y) Z/norm(Z)];
    end
end

PolohyENU_K = PolohyENU_K(:, 1:pocetPoloh);
```

Zdrojový kód A.1: Filtr polohy jako funkce v MATLABu.

B. Ostatní přílohy

Zbylé přílohy jsou soubory na přiloženém CD

- kalmanFiltr.m – funkce v MATLABu, uvedená zde v sekci příloh A
- bodyM.kml – měřené polohy z přijímače GPS ve formátu KML pro Google Earth
- cesta.kml – filtrovaný odhad trasy ve formátu KML pro Google Earth