



**FAKULTA APLIKOVANÝCH VĚD  
ZÁPADOČESKÉ UNIVERZITY  
V PLZNI**

**KATEDRA  
KYBERNETIKY**

## **Bakalářská práce**

# **Sledování pohybu pracovního nástroje a replikace pohybu robotem**

Filip Kejval



FAKULTA APLIKOVANÝCH VĚD  
ZÁPADOČESKÉ UNIVERZITY  
V PLZNI

KATEDRA  
KYBERNETIKY

## **Bakalářská práce**

# **Sledování pohybu pracovního nástroje a replikace pohybu robotem**

Filip Kejval

**Vedoucí práce**

Ing. Martin Švejda

© Filip Kejval, 2024.

Všechna práva vyhrazena. Žádná část tohoto dokumentu nesmí být reprodukována ani rozšiřována jakoukoli formou, elektronicky či mechanicky, fotokopírováním, nahráváním nebo jiným způsobem, nebo uložena v systému pro ukládání a vyhledávání informací bez písemného souhlasu držitelů autorských práv.

**Citace v seznamu literatury:**

KEJVAL, Filip. *Sledování pohybu pracovního nástroje a replikace pohybu robotem*. Plzeň, 2024. Bakalářská práce. Západočeská univerzita v Plzni, Fakulta aplikovaných věd, Katedra kybernetiky. Vedoucí práce Ing. Martin Švejda.

ZÁPADOČESKÁ UNIVERZITA V PLZNI

Fakulta aplikovaných věd  
Akademický rok: 2023/2024

# ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(projektu, uměleckého díla, uměleckého výkonu)

Jméno a příjmení: **Filip KEJVAL**  
Osobní číslo: **A21B0380P**  
Studijní program: **B0714A150005 Kybernetika a řídicí technika**  
Specializace: **Automatické řízení a robotika**  
Téma práce: **Sledování pohybu pracovního nástroje a replikace pohybu robotem**  
Zadávací katedra: **Katedra kybernetiky**

## Zásady pro vypracování

1. Seznamte se s problematikou sledování pohybu pracovních nástrojů (možné metody, výhody, nevýhody).
2. Prozkoumejte možnost využití systému HTC Vive Tracker jako senzoru pro snímání pohybu.
3. Implementujte vhodné rozhraní do řídicího systému REXYGEN.
4. Analyzujte možné pracovní módy trackeru (přenos nezpracovaných dat, nastavení filtračních mechanismů).
5. Navrhněte postupy pro zpracování měřených dat (např. interní přímo v HTC Trackeru, externí – vlastní filtrace, apod.) a postupy porovnejte.
6. Jako referenční měřicí systém (generátor referenční trajektorie) vyberte vhodný průmyslový robot, navrhněte a realizujte experiment.

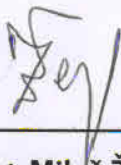
Rozsah bakalářské práce: **30-40 stránek A4**  
Rozsah grafických prací:  
Forma zpracování bakalářské práce: **tištěná/elektronická**

Seznam doporučené literatury:

1. W. Khalil, E. Dombre: Modelling, Identification and Control of Robots
2. L. Sciavicco, B. Siciliano: Modelling and Control of Robot Manipulators
3. Přednášky k předmětu URM

Vedoucí bakalářské práce: **Ing. Martin Švejda, Ph.D.**  
Katedra kybernetiky

Datum zadání bakalářské práce: **17. října 2023**  
Termín odevzdání bakalářské práce: **20. května 2024**



**Doc. Ing. Miloš Železný, Ph.D.**  
děkan



**Doc. Dr. Ing. Vlasta Radová**  
vedoucí katedry

# Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně a výhradně s použitím citovaných pramenů, literatury a dalších odborných zdrojů. Tato práce nebyla využita k získání jiného nebo stejného akademického titulu.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona v platném znění, zejména skutečnost, že Západočeská univerzita v Plzni má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 autorského zákona.

Ve Veselé u Rokycan dne 13. května 2024

.....

Filip Kejval

V textu jsou použity názvy produktů, technologií, služeb, aplikací, společností apod., které mohou být ochrannými známkami nebo registrovanými ochrannými známkami příslušných vlastníků.

## Abstrakt

Tato práce se zabývá možností využití komerčně dostupného systému virtuální reality HTC Vive a rozšířeného Kálmánova filtru pro přesné a spolehlivé on-line sledování libovolného pracovního nástroje či objektu v průmyslových řešeních a následnou replikací zaznamenané dráhy robotickým manipulátorem. Je zde popsán proces získání dat, jejich zpracování a následné porovnání sledovacích schopností s reálným robotickým systémem Staubli TX40.

## Abstract

This work deals with the possibility of using the commercially available HTC Vive system and the extended Kálmán filter for accurate and reliable online tracking of any work tool or object in industrial applications and subsequent replication of the recorded path by a robotic manipulator. The process of data acquisition, processing and subsequent comparison of tracking capabilities is then compared with the real Staubli TX40 robotic system.

## Klíčová slova

rozšířený Kálmánův filtr • filtrace dat • HTC Vive • REXYGEN • intuitivní programování robotů • záznam trajektorie

## Poděkování

Tímto bych rád poděkoval vedoucímu bakalářské práce, panu Ing. Martinu Švejdovi Ph.D za jeho podporu, vedení a cenné rady, které vedly k vypracování této práce. Dále bych rád poděkoval panu Ing. Ondřeji Severovi za opětovanou pomoc při technických potížích a panu Ing. Jindřichu Liškovi Ph.D za rady při zpracování výsledných dat.



# Obsah

<b>1</b>	<b>Úvod</b>	<b>3</b>
<b>2</b>	<b>Extrakce dat</b>	<b>5</b>
2.1	Hardware a software . . . . .	5
2.1.1	Deska UPBOARD . . . . .	5
2.1.2	Řídící systém REXYGEN . . . . .	6
2.2	Sběr dat ze senzorů . . . . .	6
2.3	Přenos dat . . . . .	7
<b>3</b>	<b>Matematický model tělesa</b>	<b>8</b>
3.1	Stavová rovnice . . . . .	9
3.2	Výstupní rovnice . . . . .	11
<b>4</b>	<b>Rozšířený Kálmánův filtr</b>	<b>14</b>
4.1	Princip EKF . . . . .	14
4.2	Realizace v systému REXYGEN . . . . .	16
4.3	Odhad šumů . . . . .	17
<b>5</b>	<b>Testování navrženého filtru</b>	<b>19</b>
5.1	Zarovnání souřadných systémů . . . . .	20
5.2	Testy . . . . .	21
5.2.1	Lineární pohyby bez změny orientace . . . . .	23
5.2.2	Lineární pohyby bez změny translace . . . . .	25
5.2.3	Obecné lineární pohyby . . . . .	26
5.2.4	Obecné kloubové pohyby . . . . .	29
<b>6</b>	<b>Závěr</b>	<b>32</b>
<b>A</b>	<b>První příloha</b>	<b>34</b>
A.1	Doplňkové grafy k lineárnímu pohybu bez změny rotace . . . . .	34
A.2	Doplňkové grafy k lineárnímu pohybu bez změny translace . . . . .	36

<b>Bibliografie</b>	<b>37</b>
<b>Seznam obrázků</b>	<b>38</b>
<b>Seznam tabulek</b>	<b>40</b>

Technologie virtuální reality je v dnešní době rozšířený a dostupný systém, který je využíván v mnoha oblastech. Tyto systémy jsou však často uzavřené a neumožňují přímý přístup k informacím z dostupných senzorů. V této práci je použito zařízení HTC Vive, pro které existuje externí knihovna Libsurvive, která umožňuje tyto data zpřístupnit ve zdrojovém formátu. Zároveň je tento systém v porovnání s průmyslovými alternativami levný, nedosahuje však dostatečné přesnosti v oblasti robotiky [Bor+18]. Kombinací systému HTC Vive a knihovny Libsurvive lze pak zaznamenávat hrubá data ze senzorů a následně je filtrovat za účelem zvýšení přesnosti. Toto řešení umožňuje mimo jiné intuitivní programování průmyslových robotů bez nutnosti hlubší znalosti této oblasti.

Operátor výroby může například manuálně provést trajektorii (rukou nebo pracovním nástrojem), která se zaznamená sledovacím systémem a následně se použije jako vzor trajektorie pro robotický manipulátor. Manipulátor tak lze jednoduše přeprogramovat na nové účely. Tím se lze zjednodušit návrh trajektorie manipulátoru v těch místech, kde není potřeba zajistit absolutní přesnost.

Dosavadní řešení mají však limitace v dosažené přesnosti sledování [Šve+22]. Tato práce prozkoumává možnost přesunutí filtračního algoritmu do prostředí REXYGEN za účelem zvýšení přesnosti sledování a přidání modularity pro libovolné aplikace. Jako filtrační algoritmus je zvolen rozšířený Kálmánův filtr, jelikož jeho algoritmus je implementován v použitém řídicím systému.

V první části je popsán problém extrakce hrubých dat z měřicí jednotky (**HTC Vive tracker + HTC Base station**) a jejich přenos do řídicího systému REXYGEN. Je zde rozveden způsob přístupu k těmto datům a následné zpracování pro spolehlivý přenos.

Dále je popsán a vytvořen matematický model tělesa pohybujícího se v prostoru se šesti stupni volnosti. Kromě obvyklých stavových veličin jako je poloha, rychlost a rotace je model rozšířen o offset (integrační chybu) úhlové rychlosti. Model je realizován v systému REXYGEN.

V další části je tento model použit na realizaci rozšířeného Kálmánova filtru. Je zde vysvětlen princip filtračního algoritmu a problematika odhadu stavového a

výstupního šumu.

Nakonec je finální filtr otestován pomocí průmyslového manipulátoru. Důraz je kladen hlavně na přesnost sledování během rychlých změn pohybu a porovnání filtrace s již dostupnými metodami.

# Extrakce dat

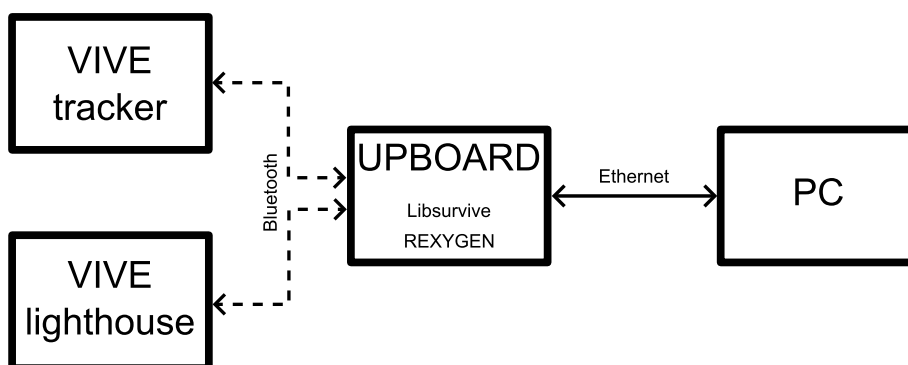
## 2

Prvním úkolem bylo získat přístup k hrubým informacím o poloze a orientaci sledovaného trackeru, zpracovat je a vhodným komunikačním prostředkem je posílat do řídicího systému.

## 2.1 Hardware a software

### 2.1.1 Deska UPBOARD

Jako výpočetní jednotka pro sběr dat a následnou filtraci byla zvolena deska **UPBOARD**. Deska disponuje procesorem INTEL, 4 GB RAM, 32GB interního uložení a standardními I/O porty pro komunikaci s vnějšími zařízeními. Jako operační systém byl použit linux Debian 11 Bullseye. Operační systémy linux umožňují spuštění procesů s velmi malou periodou, což je pro účely sběru dat a přesné filtrace nutné. Komunikace s deskou byla realizována vzdáleným přístupem pomocí ethernet portu a protokolu ssh. Bylo tak možné desku programovat a ladit na dálku v grafickém prostředí, zvolený operační systém totiž podporuje interakci pouze přes konzolový terminál. Přenos informací z trackeru a světelných stanic do výpočetní jednotky je zajištěn bezdrátovým bluetooth připojením integrovaným v systému HTC Vive. Komunikační schéma je zobrazeno na Obrázku (2.1)



Obrázek 2.1: Schéma komunikace

## 2.1.2 Řídící systém REXYGEN

REXYGEN je nástroj sloužící k nastavení a implementaci řídicích aplikací pro stroje a procesy. Je vhodný pro vývoj algoritmů pro automatické řízení, měření, regulaci, vestavěné systémy, modelování, simulaci, robotiku, mechatroniku, průmyslovou komunikaci, diagnostiku a další aplikace. Poskytuje možnost grafického programování řídicích algoritmů pomocí funkčních bloků a sekvenčních funkčních diagramů. Obsahuje nástroje pro snadnou tvorbu uživatelských rozhraní typu člověk-stroj (HMI). Podporuje širokou škálu komunikačních protokolů, včetně OPC UA, Modbus, EtherCAT a průmyslového IoT. Systém nabízí jednotné vývojové prostředí pro všechny cílové platformy s operačním systémem Linux nebo Windows. Pro uživatele se znalostí systému Matlab-Simulink je tento program snadno použitelný téměř ihned [24b].

## 2.2 Sběr dat ze senzorů

Pro přístup k pozici a rotaci sledovaného tělesa byla použita knihovna **Libsurvive**. Jedná se o open-source balíček knihoven a nástrojů poskytující možnost přístupu k datům přímo ze senzorů trackeru. Knihovna je napsaná v jazyku C a je dostupná na platformě github [22].

Po nainstalování knihovny lze příkazem `./bin/survive-cli` spustit skript, který automaticky zkalibruje tracker a začne ho sledovat. Jednoduchou vizualizaci lze spustit příkazem `./bin/survive-websocketd`. Jelikož je knihovna v základu nastavena pro použití se SteamVR, bude nejprve nutné získat přístup k samotným pozičním datům. K tomu lze použít různé obalovací komponenty dostupné v programovacích jazycích *C*, *C#* a *Python*. Pro aplikaci byla zvolena obalovací třída v jazyce *Python*, jelikož umožňuje jednoduché posílání dat (viz kapitola 2.3) a řídicí systém REXYGEN podporuje spouštění uživatelského *Python* kódu.

Základ kódu tvoří obalovací komponenta `full-example.py`, do které byly následně přidány potřebné funkcionality. Nejprve se v programu inicializuje a spustí kontext C-čkového programu příkazem `pysurvive.init(config)`, kde `config` reprezentuje volitelné konfigurační parametry. Tímto se na pozadí spustí prázdný program bez další funkcionality. Aby bylo možné přistupovat k požadovaným datům, je nutné nadefinovat funkce `pose_fn()` respektive `imu_fn()`, které umožňují získávat poziční respektive IMU (z anglického *inertial measurement unit*) data ze senzorů. Tyto funkce je pak nutné nainstalovat příkazem `pysurvive.install\_<jménoFunkceVC>\_fn(ctx, <jménoPythonFunkce>)` do C-čkového programu běžícího na pozadí. Nutno podotknout, že knihovna nabízí různé další funkcionality, které lze tímto způsobem nainstalovat (např. reakce na

stisknutí tlačítka na trackeru nebo informace o rychlosti trackeru). Pro naši aplikaci byly však použity pouze výše uvedené funkce.

Program v tomto nastavení ale nevrací hrubá data, ale už zpracovaná interním Kálmánovým filtrem. Je proto nutné během inicializace do proměnné *config* přidat parametr `--use-kalman 0`. Tímto se vypne vnitřní filtrace a program bude vracet nezpracovaná data. Do proměnné *config* je vhodné navíc přidat `--force-calibrate`, aby se tracker pokaždé zkalibroval na novou nulovou pozici při spuštění programu. Také lze tímto způsobem nastavovat například vnitřní parametry integrovaného Kálmánova filtru nebo prahy citlivosti senzorů. List všech nastavitelných parametrů lze získat pomocí příkazu `./bin/survive-cli -h`

## 2.3 Přenos dat

Program běží v nekonečném cyklu, ve kterém se dotazuje, zda nejsou k dispozici nová data. Pokud ano, zavolá se příslušná nainstalovaná obslužná funkce. Uvnitř každé funkce je tak k dispozici aktuální vektor dat z příslušného senzoru. Například ve funkci `imu_fn()` dostáváme vektor o délce 6-ti prvků, který se skládá ze zrychlení v osách *x*, *y*, *z* a úhlové rychlosti kolem těchto os.

Získaný vektor je dále rozšířen o identifikační číslo, čas vygenerování vzorku v nanosekundách a řetězec znaků, který popisuje typ uložených dat. Takto rozšířený vektor je pak zkomprimován a uložen do binární proměnné pomocí knihovny *struct* a příkazu `struct.pack()`. Minimalizuje se tak velikost posílaných paketů a jednoznačně se oddělí data mezi sebou. Pro samotný přenos dat do řídicího systému byl zvolen komunikační protokol UDP, jelikož stačí pouze jednosměrná komunikace a je nutné zajistit co nejrychlejší odesílání naměřených dat. Zkomprimovaná binární proměnná je nakonec lokálně poslána přes otevřený socket na danou adresu a port.

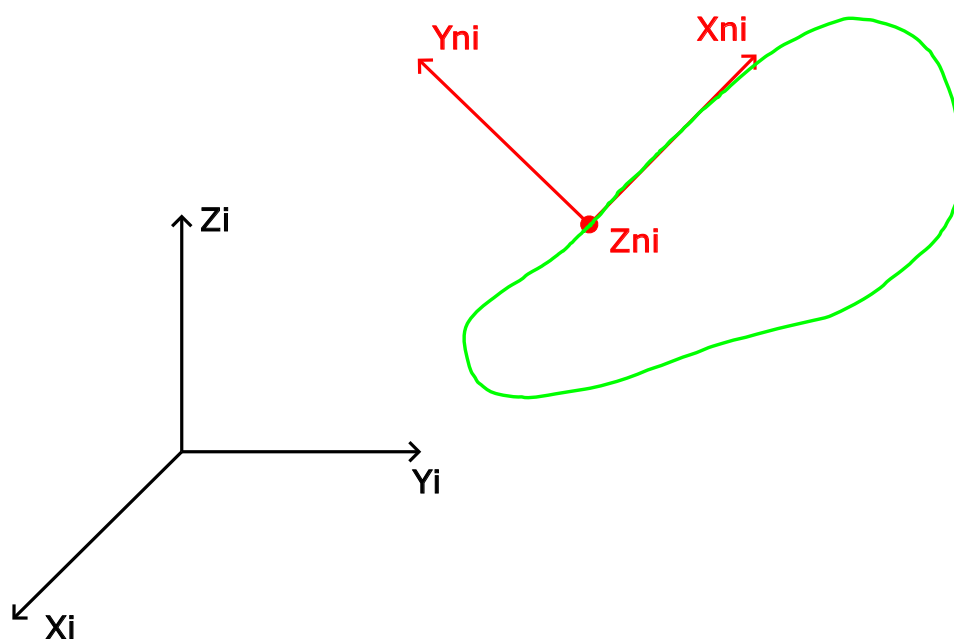
Pakety jsou přijímány v řídicím systému uvnitř bloku PYTHON, který umožňuje spouštět uživatelský kód. Task pro příjem dat se spouští s periodou *3 ms* a je umístěn první ve frontě. Uvnitř bloku se během inicializace otevře druhý socket se stejnou adresou a portem a nastaví se jako neblokující. Toto je důležité pro plynulý chod systému, protože jinak by se socket během každé periody zablokoval a čekal by, než obdrží nový packet. Po získání packetu je binární soubor dekomprimován, roztríděn na data z IMU a světelných senzorů a zapsán na výstup bloku.

Nutno podotknout, že IMU generuje nové vzorky přibližně dvakrát rychleji než optické senzory a je proto potřeba pro správný chod filtrace vhodně signalizovat dostupnost nového optického měření. V každé periodě programu se vyhodnocuje podmínka, zda jsou k dispozici nová optická data na zpracování. Pokud je tato podmínka splněna, zapíše se na výstup bloku logická hodnota 1 signalizující dostupnost nového měření. V opačném případě se zapíše logická hodnota 0 a filtrační algoritmus v tomto kroku pouze predikuje hodnoty na základě vnitřního modelu.

# Matematický model tělesa

## 3

Aby bylo možné použít Kálmánův filtr na odhad stavů, je nejprve nutné vytvořit matematický model letícího tělesa v prostoru. Na Obrázku (3.1) jsou zdefinovány jednotlivé souřadné systémy, inerciální systém HTC base stanic  $F_I$  (černě) a neinerciální systém letícího tělesa (trackeru)  $F_{NI}$  (červeně), který se pohybuje po neznámé trajektorii (zeleně). V následujících částech budou sestaveny stavové a výstupní rovnice systému, na základě kterých bude implementován model v systému REXYGEN.



Obrázek 3.1: Souřadné systémy



## 3.1 Stavová rovnice

Celý stavový model bude vyjádřen v inerciálním souřadném systému. IMU jednotka vrací data v neinerciálním souřadném systému, je proto nutné správně transformovat vstupní data.

Rovnice budou sestaveny na základě dvou dostupných vektorů měření, úhlové rychlosti  $\omega_M^{NI}$  a translačního zrychlení  $\mathbf{a}_M^{NI}$ . Tato data jsou vyjádřena vzhledem k neinerciálnímu souřadnému systému a jsou uvažována jako vstup do systému. Hledaná nelineární stavová funkce systému bude mít následující tvar:

$$\dot{\mathbf{x}} = f(\mathbf{X}(t), \omega_M^{NI}(t), \mathbf{a}_M^{NI}(t), \mathbf{N}_\omega(t), \mathbf{N}_a(t)), \quad (3.1)$$

kde  $\mathbf{x}(t)$  je vektor stavů a  $\mathbf{N}_\omega(t)$ ,  $\mathbf{N}_a(t)$  jsou šumy gyroskopu a akcelerometru.

Stavy tělesa jsou zvoleny následovně:

$$\mathbf{x} = \begin{bmatrix} s_x \\ s_y \\ s_z \\ v_x \\ v_y \\ v_z \\ q_1 \\ q_2 \\ q_3 \\ q_4 \\ \omega_{ofstX} \\ \omega_{ofstY} \\ \omega_{ofstZ} \end{bmatrix}, \quad (3.2)$$

kde  $s_{x,y,z}$  je translační poloha,  $v_{x,y,z}$  je translační rychlost,  $q_{1,2,3,4}$  jsou složky kvaternionu reprezentujícího rotaci z inerciálního do neinerciálního systému a  $\omega_{ofstX,Y,Z}$  je offset (bias) gyroskopu. Každý gyroskop má posun měřené hodnoty od reálné o nějakou konstantu a je potřeba to zahrnout do modelu.

Nejprve bude přepočítán jednotkový kvaternion na matici rotace  $R_{NI}^I$  z inerciálního souřadného systému (referenční s.s. base stanice) do neinerciálního systému trackeru (aktuální rotace). Tato matice bude dále použita na transformaci vstupních dat.

$$R_{NI}^I = \begin{bmatrix} 2q_1^2 + 2q_2^2 - 1 & -2q_1q_4 + 2q_2q_3 & 2q_1q_3 + 2q_2q_4 \\ 2q_1q_4 + 2q_2q_3 & 2q_1^2 + 2q_3^2 - 1 & -2q_1q_2 + 2q_3q_4 \\ -2q_1q_3 + 2q_2q_4 & 2q_1q_2 + 2q_3q_4 & 2q_1^2 + 2q_4^2 - 1 \end{bmatrix} \quad (3.3)$$

Použitím propagace kvaternionu bude vyjádřena časová změna jednotlivých složek vektoru  $\mathbf{q}^I$ . Je nutné zde zohlednit skutečnost, že měření  $\omega_M$  je vyjádřeno v

neinerciálním systému a je nutné ho transformovat.

$$\dot{\mathbf{q}}^I = \frac{1}{2} \mathbf{q}^I(t) A(R_{NI}^I \omega_M^{NI}) \quad (3.4)$$

$$\begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \\ \dot{q}_3 \\ \dot{q}_4 \end{bmatrix} = \frac{1}{2} \begin{bmatrix} q_1(t) \\ q_2(t) \\ q_3(t) \\ q_4(t) \end{bmatrix} \begin{bmatrix} 0 & -\omega_x(t) & -\omega_y(t) & -\omega_z(t) \\ \omega_x(t) & 0 & \omega_z(t) & -\omega_y(t) \\ \omega_y(t) & -\omega_z(t) & 0 & \omega_x(t) \\ \omega_z(t) & \omega_y(t) & -\omega_x(t) & 0 \end{bmatrix} \quad (3.5)$$

$$\begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \\ \dot{q}_3 \\ \dot{q}_4 \end{bmatrix} = \begin{bmatrix} -0.5q_2(t)\omega_x(t) - 0.5q_3(t)\omega_y(t) - 0.5q_4(t)\omega_z(t) \\ 0.5q_1(t)\omega_x(t) - 0.5q_4(t)\omega_y(t) + 0.5q_3(t)\omega_z(t) \\ 0.5q_4(t)\omega_x(t) + 0.5q_1(t)\omega_y(t) - 0.5q_2(t)\omega_z(t) \\ -0.5q_3(t)\omega_x(t) + 0.5q_2(t)\omega_y(t) + 0.5q_1(t)\omega_z(t) \end{bmatrix} \quad (3.6)$$

Úhlová rychlost  $\omega_{x,y,z}$  obsahuje následující složky

$$\omega_x = \omega_{Mx} + \omega_{ofstX} + N_{\omega X} \quad (3.7)$$

$$\omega_y = \omega_{My} + \omega_{ofstY} + N_{\omega Y} \quad (3.8)$$

$$\omega_z = \omega_{Mz} + \omega_{ofstZ} + N_{\omega Z} \quad (3.9)$$

Dále bude vyjádřena derivace translační rychlosti. Zde je opět nutno použít výše odvozenou transformační matici  $R_{NI}^I$  a aplikovat jí na měřený vektor zrychlení  $\mathbf{a}_M$ .

$$\dot{\mathbf{v}}^I = R_{NI}^I \mathbf{a}_M^{NI} - G^I \quad (3.10)$$

$$\begin{bmatrix} \dot{v}_x \\ \dot{v}_y \\ \dot{v}_z \end{bmatrix} = \begin{bmatrix} (2q_1^2 + 2q_2^2 - 1)a_{Mx} + (-2q_1q_4 + 2q_2q_3)a_{My} + (2q_1q_3 + 2q_2q_4)a_{Mz} \\ (2q_1q_4 + 2q_2q_3)a_{Mx} + (2q_1^2 + 2q_3^2 - 1)a_{My} + (-2q_1q_2 + 2q_3q_4)a_{Mz} \\ (-2q_1q_3 + 2q_2q_4)a_{Mx} + (2q_1q_2 + 2q_3q_4)a_{My} + (2q_1^2 + 2q_4^2 - 1)a_{Mz} + g_0 \end{bmatrix}, \quad (3.11)$$

kde  $G^I$  je vektor gravitačního zrychlení v inerciálním souřadném systému

$$G^I = \begin{bmatrix} 0 \\ 0 \\ g_0 \end{bmatrix}. \quad (3.12)$$

Offset gyroskopu  $\omega_{ofst}$  je obecně konstanta a nebude se měnit.

$$\begin{bmatrix} \dot{\omega}_{ofstX} \\ \dot{\omega}_{ofstY} \\ \dot{\omega}_{ofstZ} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \quad (3.13)$$

Časové změny polohy jsou rovny aktuální rychlosti trackeru. Rovnice budou ve tvaru

$$\dot{\mathbf{s}} = \mathbf{v} \quad (3.14)$$

$$\begin{bmatrix} \dot{s}_x \\ \dot{s}_y \\ \dot{s}_z \end{bmatrix} = \begin{bmatrix} v_x \\ v_y \\ v_z \end{bmatrix}. \quad (3.15)$$

Nyní jsou odvozeny všechny potřebné vztahy pro vyjádření stavové rovnice (3.2). Vyjádřená stavová rovnice (3.16) bude složena postupně z (3.15),(3.11),(3.6) a (3.13).

$$\dot{\mathbf{x}} = \begin{bmatrix} v_x \\ v_y \\ v_z \\ (2q_1^2 + 2q_2^2 - 1)a_{Mx} + (-2q_1q_4 + 2q_2q_3)a_{My} + (2q_1q_3 + 2q_2q_4)a_{Mz} \\ (2q_1q_4 + 2q_2q_3)a_{Mx} + (2q_1^2 + 2q_3^2 - 1)a_{My} + (-2q_1q_2 + 2q_3q_4)a_{Mz} \\ (-2q_1q_3 + 2q_2q_4)a_{Mx} + (2q_1q_2 + 2q_3q_4)a_{My} + (2q_1^2 + 2q_4^2 - 1)a_{Mz} + g_0 \\ -0.5q_2(t)\omega_x(t) - 0.5q_3(t)\omega_y(t) - 0.5q_4(t)\omega_z(t) \\ 0.5q_1(t)\omega_x(t) - 0.5q_4(t)\omega_y(t) + 0.5q_3(t)\omega_z(t) \\ 0.5q_4(t)\omega_x(t) + 0.5q_1(t)\omega_y(t) - 0.5q_2(t)\omega_z(t) \\ -0.5q_3(t)\omega_x(t) + 0.5q_2(t)\omega_y(t) + 0.5q_1(t)\omega_z(t) \\ 0 \\ 0 \\ 0 \end{bmatrix} \quad (3.16)$$

## 3.2 Výstupní rovnice

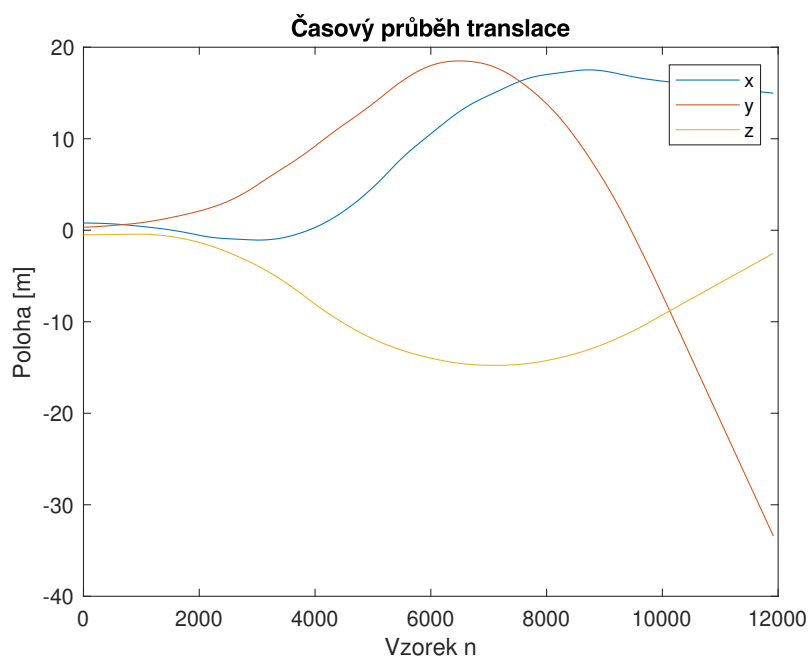
Výstupní rovnice je uvažována ve tvaru

$$\mathbf{y} = \mathbf{h}(\mathbf{x}(t)) + \mathbf{v}(t), \quad (3.17)$$

kde  $\mathbf{v}(t)$  je vektor aditivního výstupního šumu. Jelikož je celý stavový popis uvažován v inerciálním systému, výstupní rovnice bude obsahovat polohu a orientaci sledovaného trackeru bez dalších transformací. Rovnice bude samozřejmě zatížena výstupním aditivním šumem.

$$\mathbf{y} = \begin{bmatrix} s_x + N_x \\ s_y + N_y \\ s_z + N_z \\ q_1 + N_{q1} \\ q_2 + N_{q2} \\ q_3 + N_{q3} \\ q_4 + N_{q4} \end{bmatrix} \quad (3.18)$$

Pro ověření funkčnosti byl model vytvořen v prostředí REXYGEN použitím standardních bloků pro práci s vektory a maticemi. Bylo také nutné přidat normalizaci kvaternionu, jelikož během každého kroku se vlivem výpočtů mění velikost kvaternionu. Na obrázku (3.2) je zobrazena výstupní poloha namodelovaného trackeru v reakci na změny měřeného zrychlení fiktivní IMU jednotkou bez vlivu šumu.



Obrázek 3.2: Průběh translace

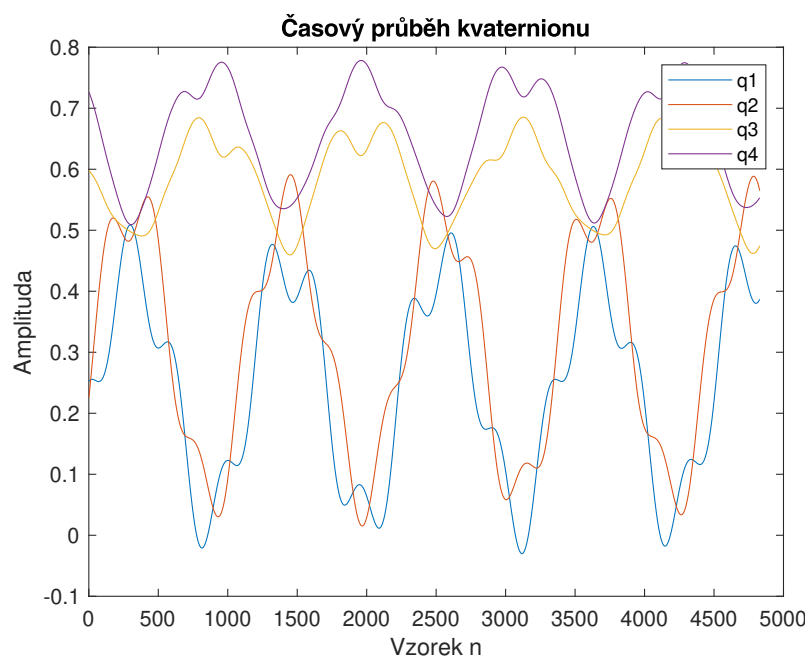
Vstupní hodnoty zde byly zvoleny různě velká konstantní zrychlení. Model neuvazuje ztráty vlivem odporu vzduchu a tím pádem poloha postupně diverguje, jedná se tedy o nestabilní subsystém. Rotace byla testována podobným způsobem. Jako vstup byly použity signály

$$\omega_{Mx} = 0.3 \sin t + 10$$

$$\omega_{My} = 0.7 \sin 0.3t - 25$$

$$\omega_{Mz} = 0.5 \sin t.$$

Časový průběh hodnot kvaternionu je na obrázku (3.3). Tento subsystém je díky normalizaci kvaternionu stabilní.



Obrázek 3.3: Průběh rotace

# Rozšířený Kálmánův filtr

## 4

Reálná data jsou často zatížena nepřesnostmi měřících instrumentů a náhodným šumem. Pro zvýšení přesnosti lze použít Kálmánův filtr (KF). Tato metoda filtrace používá dynamický model systému pro predikci neznámého vnitřního stavu a následnou korekci tohoto odhadu pomocí měření výstupní veličiny. Lze tak dosáhnout vyšší přesnosti než při použití pouze jednoho měření.

Základní KF ale uvažuje lineární model systému. U sledování objektu v 3-D prostoru se ale obecně jedná o nelineární systém, je tedy nutné použít rozšířený Kálmánův filtr (EKF, z anglického *extended kalman filter*), který umožňuje filtrovat data i pro nelineární systémy.

## 4.1 Princip EKF

V kapitole 3 byl odvozen nelineární model zatížený šumem ve tvaru

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t), \boldsymbol{\omega}(t)) \quad (4.1)$$

$$\mathbf{y} = \mathbf{h}(\mathbf{x}(t)) + \mathbf{v}, \quad (4.2)$$

kde  $\mathbf{x}(t) \in \mathbb{R}^n$  je vektor vnitřních stavů,  $\mathbf{u}(t) \in \mathbb{R}^d$  je vstup do systému z  $d$  senzorů,  $\boldsymbol{\omega}(t) \in \mathbb{R}^n$  je stavový stochastický šum,  $\mathbf{y} \in \mathbb{R}^m$  je vektor měření a  $\mathbf{v} \in \mathbb{R}^m$  je stochastický šum měření.

Základní způsob vyjádření vlivu šumu na systém je využití předpokladu, že šum působí na systém aditivně. Pro výše popsaný systém toto ale neplatí, jelikož stavová rovnice (4.1) je mimo jiné funkcí neznámého stavového šumu  $\boldsymbol{\omega}(t)$ . Převod na popis s aditivním šumem je popsán níže. Výstupní rovnice 4.2 představuje pozorovatelné stavy systému, které zle přímo měřit externím senzorem. Funkce  $\mathbf{h}(\cdot)$  tedy mapuje vnitřní stavy systému na měřitelné výstupy. Tato rovnice je zatížena šumem měření  $\mathbf{v}$ , který je ale v tomto případě aditivní.

EKF představuje v jiném slova smyslu prediktor stavu, tedy algoritmus, který na základě vstupu  $\mathbf{u}(t)$ , modelu systému  $\mathbf{f}(\cdot)$  a měření  $\mathbf{y}(t)$  odhaduje vnitřní stav

$\hat{\mathbf{x}}(t)$ . EKF nepracuje se stavovým popisem systému, ale s chybou odhadu  $\delta\mathbf{x}(t)$

$$\delta\mathbf{x}(t) = \mathbf{x}(t) - \hat{\mathbf{x}}(t), \quad (4.3)$$

kde  $\delta\mathbf{x}(t)$  je vektor chyby odhadu stavů. Obecně lze říci, že pro správné vysledování vnitřního stavu se musí  $\delta\mathbf{x}(t)$  blížit nule. Lineární model chyb odhadu stavu je získán použitím Taylorova rozvoje na rovnice (4.1)(4.2) a zanedbáním vyšších řádů. Zároveň jsou v tomto kroku separovány stavové šумы.

$$\delta\dot{\mathbf{x}} = F(t)\delta\mathbf{x}(t) + \mathbf{w} \quad (4.4)$$

$$\delta\mathbf{y} = H(t)\delta\mathbf{x}(t) + \mathbf{v}, \quad (4.5)$$

kde

$$F(t)_{n \times n} = \frac{\partial \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t))}{\partial \mathbf{x}} \quad (4.6)$$

$$H(t)_{m \times n} = \frac{\partial \mathbf{h}(\mathbf{x}(t))}{\partial \mathbf{x}}. \quad (4.7)$$

EKF předpokládá šумы  $\mathbf{w}(t)$  a  $\mathbf{v}(t)$  z normálního rozdělení s nulovou střední hodnotou a kovariančními maticemi  $Q \in \mathbb{R}^{n \times n}$  a  $R \in \mathbb{R}^{m \times m}$ . Kovarianční matice  $Q$  je však v této práci proměnná v závislosti na aktuální rotaci a orientaci tělesa. Je tedy potřeba vyjádřit novou kovarianční matici  $Q_{\bar{\mathbf{w}}}(k)$ . Nejprve bude zavedena nová náhodná veličina  $\bar{\mathbf{w}}$  reprezentující proměnlivý šum

$$\bar{\mathbf{w}} = L_k(\mathbf{x})\mathbf{w}. \quad (4.8)$$

Jedná se stále o bílý šum s nulovou střední hodnotou. Dále bude určena jeho kovariance  $Q_{\bar{\mathbf{w}}}(k)$

$$\begin{aligned} \text{COV}[\bar{\mathbf{w}}] &= E[(\bar{\mathbf{w}} - E[\bar{\mathbf{w}}])(\bar{\mathbf{w}} - E[\bar{\mathbf{w}}])^T] \\ \text{COV}[\bar{\mathbf{w}}] &= E[(L_k(\mathbf{x})\mathbf{w} - E[L_k(\mathbf{x})\mathbf{w}])(L_k(\mathbf{x})\mathbf{w} - E[L_k(\mathbf{x})\mathbf{w}])^T] \\ \text{COV}[\bar{\mathbf{w}}] &= E[(L_k(\mathbf{x})\mathbf{w}\mathbf{w}^t L_k(\mathbf{x})^T] \\ \text{COV}[\bar{\mathbf{w}}] &= L_k(\mathbf{x})E[\mathbf{w}\mathbf{w}^t]L_k(\mathbf{x})^T \\ \text{COV}[\bar{\mathbf{w}}] &= L_k(\mathbf{x})\text{COV}[\mathbf{w}]L_k(\mathbf{x})^T \\ \text{COV}[\bar{\mathbf{w}}] &= L_k(\mathbf{x})QL_k(\mathbf{x})^T \\ Q_{\bar{\mathbf{w}}}(k) &= L_k(\mathbf{x})QL_k(\mathbf{x})^T, \end{aligned} \quad (4.9)$$

kde  $Q$  je konstantní kovariance stavových šumů  $\mathbf{w}$  a  $L_k(\mathbf{x})$  se vypočítá jako

$$L_k = \left[ \frac{\partial \mathbf{f}(\mathbf{x}(k), \mathbf{u}(k), \mathbf{w}(k))}{\partial \mathbf{u}_1} \quad \frac{\partial \mathbf{f}(\mathbf{x}(k), \mathbf{u}(k), \mathbf{w}(k))}{\partial \mathbf{u}_2} \right], \quad (4.10)$$

kde  $\mathbf{u}_1, \mathbf{u}_2$  jsou vstupy  $\mathbf{a}_M$  a  $\omega$ . Matice  $Q$  tedy reprezentuje kovarianci v kroku  $k = 0$ . Výsledná stavová rovnice bude mít tedy tvar

$$\delta \dot{\mathbf{x}} = F(t) \delta \mathbf{x}(t) + \bar{\mathbf{w}} \quad (4.11)$$

$$\bar{\mathbf{w}} \in N(0, Q_{\bar{w}}). \quad (4.12)$$

Dále bude definována matice  $P \in \mathbb{R}^{n \times n}$  jako

$$P(t) = \delta \mathbf{x}(t) \delta \mathbf{x}(t)^T, \quad (4.13)$$

představující kovarianční matici aposteriori odhadu přesnosti stavu.

Samotný algoritmus EKF pak pracuje ve dvou krocích, kroku predikce a kroku opravy. V kroku predikce se vypočítá odhadovaný stav na základě stavové funkce (4.1)

$$\mathbf{x}_{k+1|k} = f(\mathbf{x}_k, \mathbf{u}_k, \mathbf{w}_k) \quad (4.14)$$

$$P_{k+1|k} = F_k P_k F_k^T + G_k Q G_k^T, \quad (4.15)$$

kde  $\mathbf{x}_{k+1|k}$  je predikovaný stav v následujícím kroku a  $P_{k+1|k}$  predikovaná kovariance odhadu. Během kroku opravy se zkombinuje predikovaná hodnota s naměřenou a vypočítá se korekce stavu na základě Kálmánova zesílení  $K$ .

$$K_{k+1} = P_{k+1|k} H_k^T (H_k P_{k+1|k} H_k^T + R_k)^{-1} \quad (4.16)$$

$$\mathbf{x}_{k+1|k+1} = \mathbf{x}_{k+1|k} + K_k (\mathbf{y}_k - H_k \mathbf{x}_{k+1|k}) \quad (4.17)$$

$$P_{k+1|k+1} = P_{k+1|k} - K_k H_k P_{k+1|k}, \quad (4.18)$$

kde  $\mathbf{x}_{k+1|k+1}$  je opravený odhad stavu,  $P_{k+1|k+1}$  je opravená kovariance odhadu a  $\mathbf{y}_k$  je dostupný vektor měření. Optimální zesílení  $K_k$  minimalizuje odchylku  $\delta \mathbf{x}_k$  [Ati19].

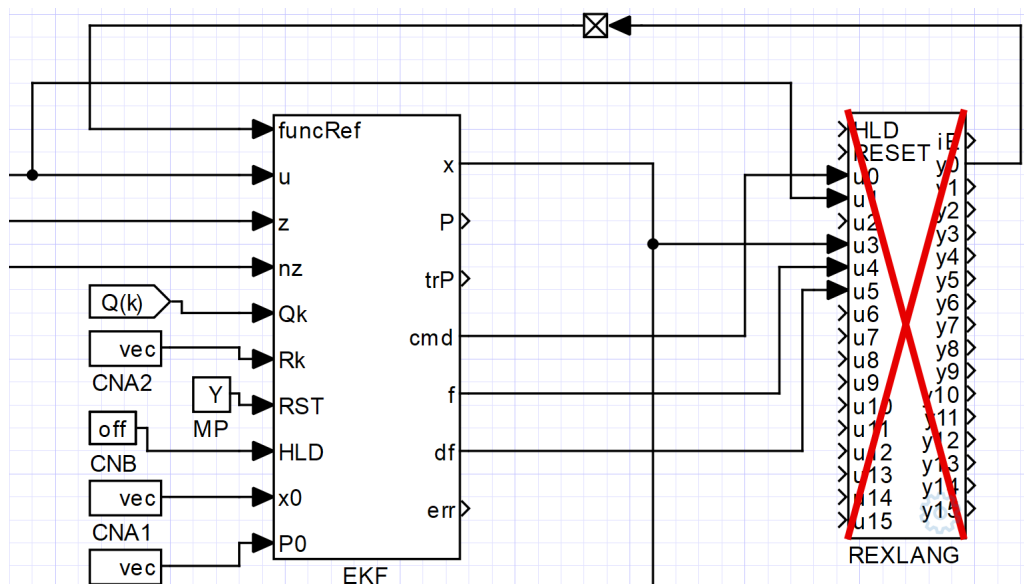
## 4.2 Realizace v systému REXYGEN

Pro realizaci filtru byl v řídicím systému vytvořen nový Task, který byl ve spouštěcí frontě umístěn za Task zajišťující příjem dat. Celková perioda běžícího programu je  $3ms$ , z toho se  $1ms$  přijímají data a během  $2ms$  se tato data filtrují a zpracovávají.

Uvnitř Tasku je pak zapojen blok EKF, který realizuje algoritmus Rozšířeného Kálmánova filtru. Jako numerická metoda integrace byla zvolena Adams-Bashforthova metoda druhého řádu. Konkrétní implementace je popsána v dokumentaci [24a].

Spojité diferenciální rovnice systému jsou napsány uvnitř kooperačního bloku REXLANG. Oba bloky je nutné správně pospojovat, aby byla zaručena funkčnost algoritmu (Obrázek 4.1). Krok filtrace je proveden vždy když je v daném okamžiku spuštění bloku hodnota  $n_z > 0$ . Toto signalizuje dostupný vektor optického měření na vstupu  $z$ , který je následně použit pro opravu odhadu stavu a jeho kovariance.





Obrázek 4.1: Schéma zapojení EKF a REXLANG

Uvnitř bloku REXLANG se nejprve načtou reference na vstupní respektive stavový vektor ze vstupů  $u_1$  respektive  $u_3$  příkazem `GetInArrDouble(<čí sloVstupu>, řádek, vektor)`; Po načtení stavového vektoru se navíc provede normalizace načteného kvaternionu  $q$ . Dále je v kódu přepínací příkaz `switch`, který v závislosti na stavu vstupu  $u_0$  přepíná mezi výpočtem stavové ( $f(x, u)$ ,  $df(x, u)$ ) a výstupní ( $h(x)$ ,  $dh(x)$ ) rovnice podle požadavku z EKF bloku. Tyto rovnice odpovídají rovnicím (3.16)(3.18) a reference na jejich hodnotu je zapisována na vstupy  $u_4$  a  $u_5$  příkazem `SetInArrValue(<čí sloVstupu>, řádek, vektor, hodnota)`; Tímto způsobem jsou předkládány hodnoty funkcí pro algoritmus EKF bloku.

## 4.3 Odhad šumů

Kálmánův filtr poskytuje optimální predikci neznámého stavu pouze za předpokladu, že šumy působící na systém jsou náhodné veličiny z Gaussova rozložení s nulovou střední hodnotou a známými kovariančními maticemi

$$\boldsymbol{w} \in N(0, Q) \quad (4.19)$$

$$\boldsymbol{v} \in N(0, R). \quad (4.20)$$

V praxi je ale velmi obtížné tyto šumy přesně odhadnout. Pro účely této práce byly kovarianční matice výstupního a stavového šumu  $R$  a  $B$  zvoleny iteračně pomocí experimentů. Přihlíženo bylo hlavně na rychlost vysledování, minimalizaci překmitu

a dostatečné vyhlazení výstupních dat

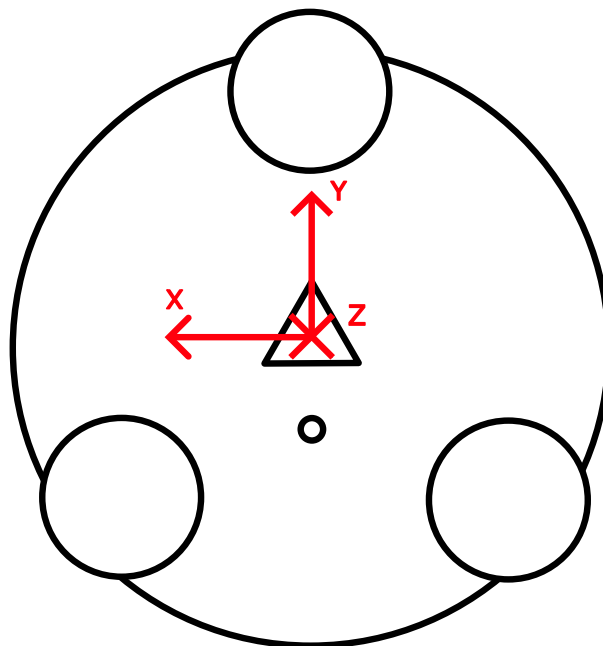
$$R = \begin{bmatrix} 0.001 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.001 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.001 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.01 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.01 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.01 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.01 \end{bmatrix}, \quad (4.21)$$

$$Q = \begin{bmatrix} 0.1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.1 \end{bmatrix}. \quad (4.22)$$

# Testování navrženého filtru

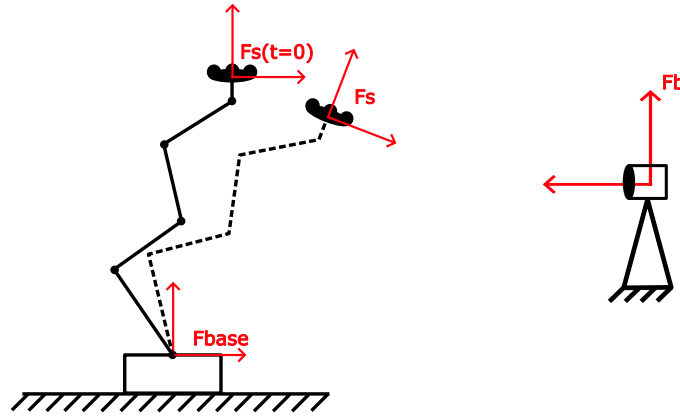
## 5

Pro úlohu testování byl zvolen robotický manipulátor **Staubli TX40** řízený systémem **REXYGEN**. Tracker byl připevněn na přírubu robotu pomocí úchytu vytisknutém na 3D tiskárně a pro optické snímání byly použity dvě **HTC Vive** base stanice ve vzdálenosti přibližně  $2m$  od manipulátoru. Celková výška úchytu je  $88mm$ , bylo tedy nutné přidat offset pracovního nástroje v řídicím systému manipulátoru a přenastavit souřadný systém koncového efektoru tak, aby byl stejný se souřadným systémem trackeru (Obrázek (5.1)).



Obrázek 5.1: Souřadný systém HTC Trackeru

## 5.1 Zarovnání souřadných systémů



Obrázek 5.2: Souřadné systémy během experimentu, vlevo robot, vpravo HTC base stanice

Manipulátor vrací transformaci od paty robotu do koncové příruby, tedy transformaci  $T_S^{base}$ . Data získaná z trackeru jsou ale vztažena k jedné HTC Base stanici, která je navíc vnitřním algoritmem zvolena náhodně. Aby bylo možné měření porovnat, musí tracker vracet stejnou transformaci jako manipulátor, tedy  ${}^*T_S^{base}$ . Tato transformace se musí odvodit jen ze známého měření trackeru.

Z trackeru je získána transformace  $T_S^b$  z neznámého souřadného systému  $b$  do aktuální pozice  $S$ . Tato transformace může být zapsána a upravena jako

$$T_S^b(t) = T_{S(t=0)}^b T_S^{S(t=0)}(t) \quad (5.1)$$

$$T_S^{S(t=0)}(t) = (T_{S(t=0)}^b)^{-1} T_S^b(t) \quad (5.2)$$

kde  $T_{S(t=0)}^b$  je konstantní transformace z  $b$  do inicializační pozice v čase  $t = 0$  a  $T_S^{S(t=0)}(t)$  je transformace z inicializační pozice do  $S$ . Transformaci  $T_{S(t=0)}^b$  lze odměřit z počátečních dat před začátkem experimentu a následně spočítat její inverzi, kterou bude pak násobena každá nově přijatá transformace. Získáme tak transformaci vzhledem k novému souřadnému systému  $F_{S(t=0)}$ , která odpovídá pozici a orientaci, kde byl systém inicializován (5.3).

$$T_S^{S(t=0)}(t=0) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (5.3)$$

Hledanou transformaci  ${}^*T_S^{base}(t)$  lze zapsat jako

$${}^*T_S^{base}(t) = T_{S(t=0)}^{base} T_S^{S(t=0)}(t) \quad (5.4)$$

kde  $T_{S(t=0)}^{base}$  je konstantní transformace z paty robotu do koncového efektoru v čase  $t = 0$ . Ta je vyjádřena z prvního vzorku pozičních a rotačních dat z manipulátoru a převedena na homogenní transformační matici [Šve17].

Změna referenčního souřadného systému je realizována kalibračním algoritmem uvnitř bloku *rot\_pos\_calibration*. Po spuštění kalibrace se nejprve 10 vteřin snímá a průměruje transformace  $T_{S(t=0)}^b$ . Tato matice se uloží a všechny následující naměřené transformace jsou násobeny inverzí této matice. Jakýkoliv další pohyb trackeru je tak vztažen k souřadnému systému, kde byla kalibrace spuštěna.

Nakonec je tato transformace vynásobena druhou konstantní transformací  $T_{S(t=0)}^{base}$  a tím se získá požadovaná transformace  $*T_S^{base}$ . Tato operace je provedena až při zpracovávání dat, aby byl kalibrační algoritmus obecně použitelný v libovolné aplikaci požadující zkalibrování měřicího instrumentu.

Nyní jsou k dispozici dvě transformace  $T_S^{base}$  a  $*T_S^{base}$ , které vyjadřují stejnou transformaci, můžou být tedy použity na otestování přesnosti filtrace.

## 5.2 Testy

Během testování byl porovnáván interní Kálmánův filtr z knihovny Libsurvive s Kálmánovým filtrem vytvořeným v prostředí REXYGEN (4.2). Přesnost sledování byla otestována na devíti trajektoriích

1. Lineární pohyby se zastavením na 2s, bez změny orientace ( $v = 50mm/s$ ,  $a = 100mm/s^2$ )
2. Lineární pohyby se zastavením na 2s, bez změny orientace ( $v = 300mm/s$ ,  $a = 800mm/s^2$ )
3. Lineární pohyby s poly napojením ( $r = 30mm$ ), bez změny orientace ( $v = 300mm/s$ ,  $a = 800mm/s^2$ )
4. Lineární pohyby se zastavením na 2s, bez změny translace ( $\omega = 10deg/s$ ,  $\dot{\omega} = 180deg/s^2$ )
5. Lineární pohyby se zastavením na 2s, bez změny translace ( $\omega = 100deg/s$ ,  $\dot{\omega} = 180deg/s^2$ )
6. Lineární pohyby bez zastavení ( $v = 50mm/s$ ,  $a = 100mm/s^2$ ,  $\omega = 10deg/s$ ,  $\dot{\omega} = 180deg/s^2$ )
7. Lineární pohyby s poly napojením ( $r = 30mm$ ,  $v = 250mm/s$ ,  $a = 800mm/s^2$ ,  $\omega = 100deg/s$ ,  $\dot{\omega} = 1080deg/s^2$ )
8. Kloubové pohyby s poly napojením ( $\omega = 60deg/s$ ,  $\dot{\omega} = 720deg/s^2$ )

9. Kloubové pohyby s poly napojením ( $\omega = 120deg/s$ ,  $\dot{\omega} = 720deg/s^2$ )

Všechny pokusy byly provedeny v laboratorním prostředí. Před každým pokusem byla provedena kalibrace popsaná v (5.1), aby byl minimalizován efekt unášení gyroskopu. Následně byla vykonána předem vytvořená trajektorie robotickým manipulátorem a po ukončení pohybu byla data z obou měřidel uložena. Tento postup byl opakován pro všechny zadané trajektorie. Naměřené data bylo následně nutné znormalizovat, jelikož obě měřidla mají různé periody vzorkování.

$$T_{tracker} = 0.003s$$

$$T_{rob} = 0.02s$$

Pro účely porovnání bylo také nutné srovnat počet vzorků dat z obou měřidel, byla proto provedena interpolace dat z manipulátoru na stejný počet vzorků, jako měly data z trackeru. Na naměřená data z trackeru nebyly použity žádné zpracovací algoritmy, aby bylo zajištěno co nejvěrohodnější porovnání mezi oběma filtry.

Pro vyhodnocení přesnosti sledování a porovnání obou algoritmů bylo použito zprůměrované IAE kritérium

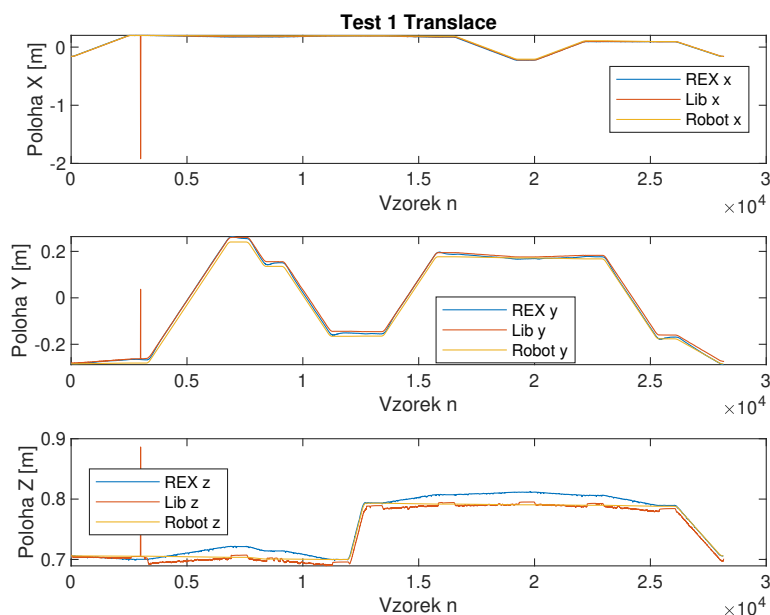
$$IAE = \frac{\sum_{i=1}^n |ROB_i - HTC_i|}{n} [m] \quad (5.5)$$

kde  $ROB_i$  jsou data v  $i$ -tém vzorku získaná z manipulátoru,  $HTC_i$  jsou data v  $i$ -tém vzorku odměřená z trackeru a  $n$  je počet vzorků. Chyba se počítá vždy pro celý experiment v jednotlivých osách jak pro translaci, tak pro orientaci.

Výše uvedené testy lze rozdělit do čtyř kategorií, lineární pohyby bez změny orientace, lineární pohyby bez změny translace, obecné lineární pohyby a obecné kloubové pohyby. Během experimentů bylo testováno unášení gyroskopu během zastavení pohybu a přesnost sledování při pozvolných a ostrých změnách translačního a rotačního pohybu. V grafech je porovnávána přesná hodnota souřadnice z manipulátoru *Robot* s odměřenou hodnotou z trackeru pro navržený algoritmus *REX* a interním algoritmem *Lib*.

## 5.2.1 Lineární pohyby bez změny orientace

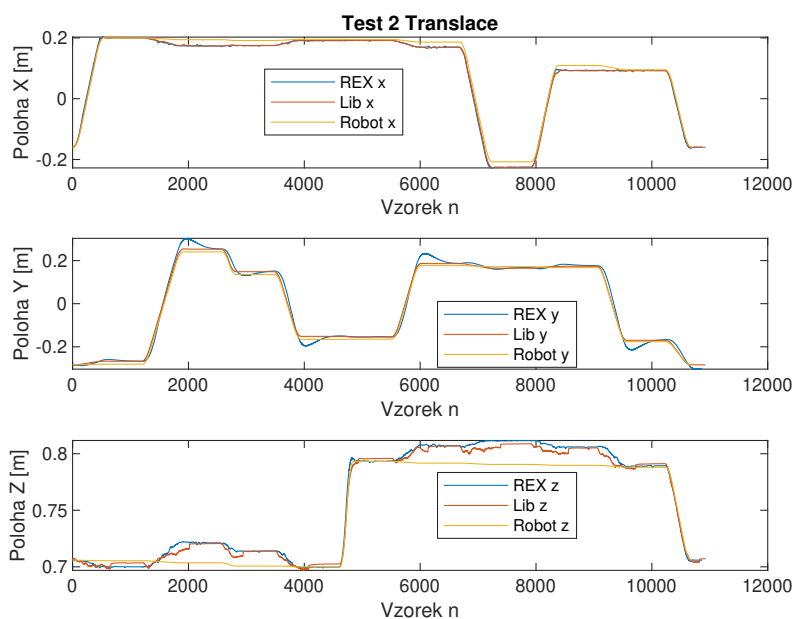
Výsledky přesnosti sledování první trajektorie interním respektive navrženým filtrem jsou zobrazeny na obrázku 5.3.



Obrázek 5.3: Porovnání přesnosti sledování, první trajektorie, Translace,  $IAE_{Lib1-T} = 0.008$ ,  $IAE_{REX1-T} = 0.005$

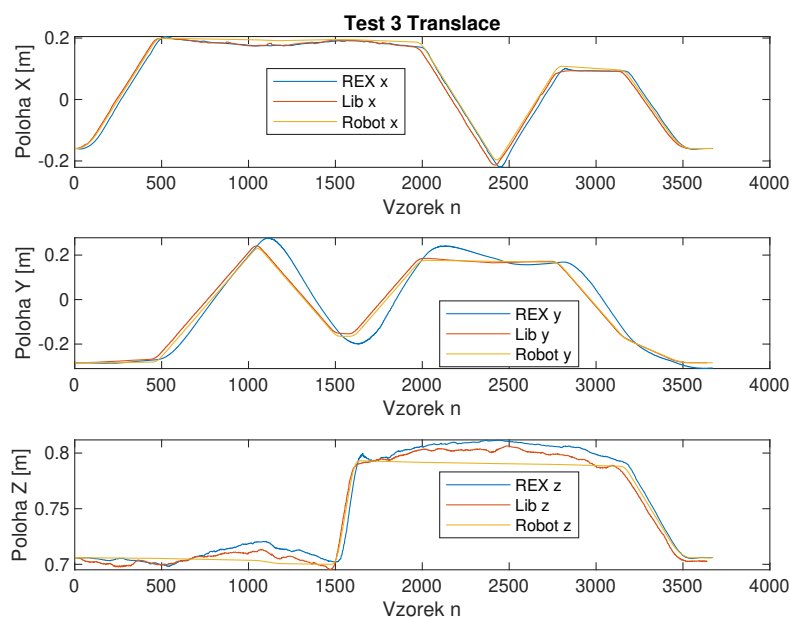
Interní filtr ztratil ve vzorku  $n = 2996$  optická data, což vyvolalo vysokou špičku v měřené poloze. Pokud by stejná situace nastala během používání navrženého filtru, algoritmus by přestal předkládat data z optických senzorů do výpočtu a používal by pouze predikci z matematického modelu. Docílilo by se tak vyhlazení špičky. Výraznější unášení gyroskopu bylo pozorováno v Z-ové složce navrženého filtru, maximální chyba od referenčního měření byla  $e = 0.021m$ . Zároveň byly u navrženého filtru pozorovány nepatrné překmity v Y-ové složce u ostrých přechodů.

Výsledky druhého experimentů jsou na obrázku 5.4. Opět lze pozorovat unášení gyroskopů v Z-ové složce translace, tentokrát u obou algoritmů. Vlivem zvýšení rychlosti pohybu manipulátoru narostly překmity v Y-ové složce navrženého filtru. Pohyby se zastavením tedy vyvolají v Y-ové překmit.



Obrázek 5.4: Porovnání přesnosti sledování, druhá trajektorie, Translace,  $IAE_{Lib2-T} = 0.009$ ,  $IAE_{REX2-T} = 0.005$

Poslední experiment přidal hladké napojení mezi lineárními pohyby o poloměru  $R = 30mm$ . Výsledky jsou zobrazeny za obrázku 5.5. Rychlá změna směru pohybu bez čekání způsobila u navrženého filtru opět značný překmit v Y-ové složce a měřená hodnota se během celého experimentu téměř neustálila na referenci.



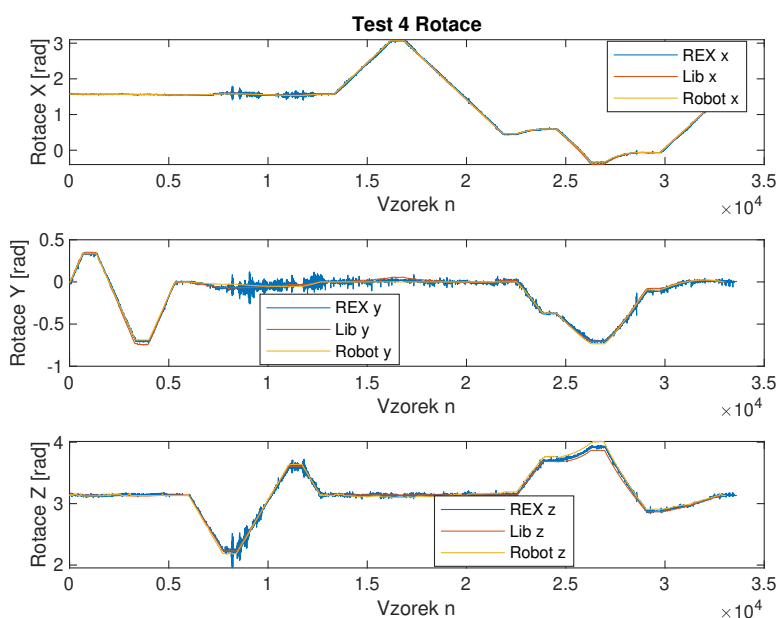
Obrázek 5.5: Porovnání přesnosti sledování, třetí trajektorie, Translace,  $IAE_{Lib3-T} = 0.008$ ,  $IAE_{REX3-T} = 0.0003$



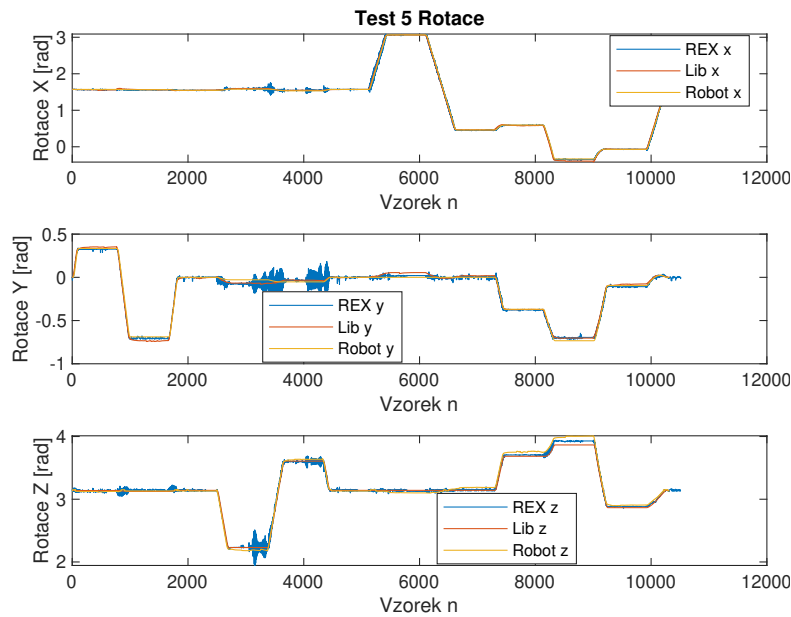
Z grafů lze usoudit, že si filtry zachovávají schopnost sledovat daný objekt i při vyšších translačních rychlostech. Výjimkou je Y-ová osa navrženého algoritmu, kde pro tyto druhy pohybu vzniká během rychlejšího pohybu vysoký překmit. Interní filtr pak dosáhl v těchto experimentech lepší výsledky než navržený ve smyslu zvoleného kritéria. Naměřená rotace byla ve všech případech téměř shodná s referenční hodnotou, odchylky obou filtračních algoritmů byly v rozsahu setin radiánu. Tyto grafy jsou k dispozici k nahlédnutí v příloze A.1.

## 5.2.2 Lineární pohyby bez změny translace

Ve čtvrtém a pátém experimentu byl tracker rotován kolem počátku souřadného systému. Výsledky jsou zobrazeny na obrázku 5.6 a 5.7.



Obrázek 5.6: Porovnání přesnosti sledování, čtvrtá trajektorie, Rotace,  $IAE_{REX4-R} = 0.006$ ,  $IAE_{Lib4-R} = 0.015$

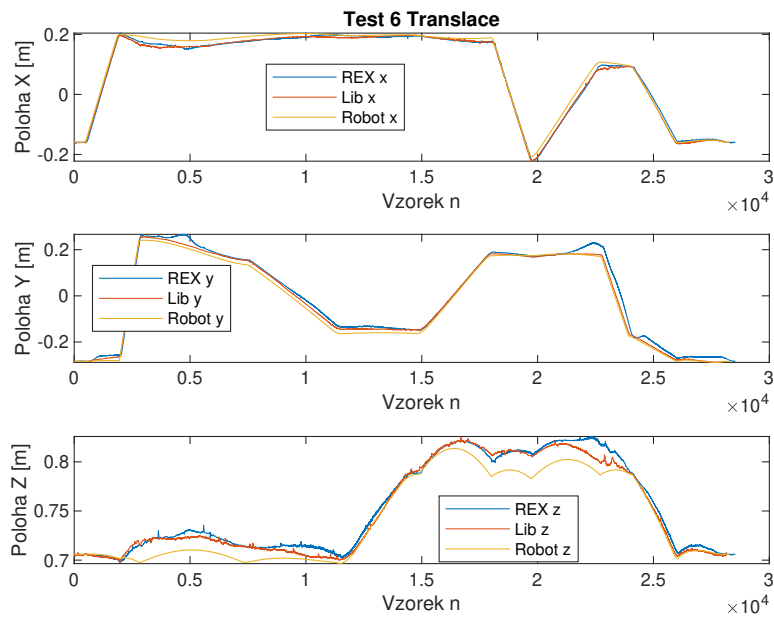


Obrázek 5.7: Porovnání přesnosti sledování, pátá trajektorie, Rotace,  $IAE_{REX5-R} = 0.003$ ,  $IAE_{Lib5-R} = 0.019$

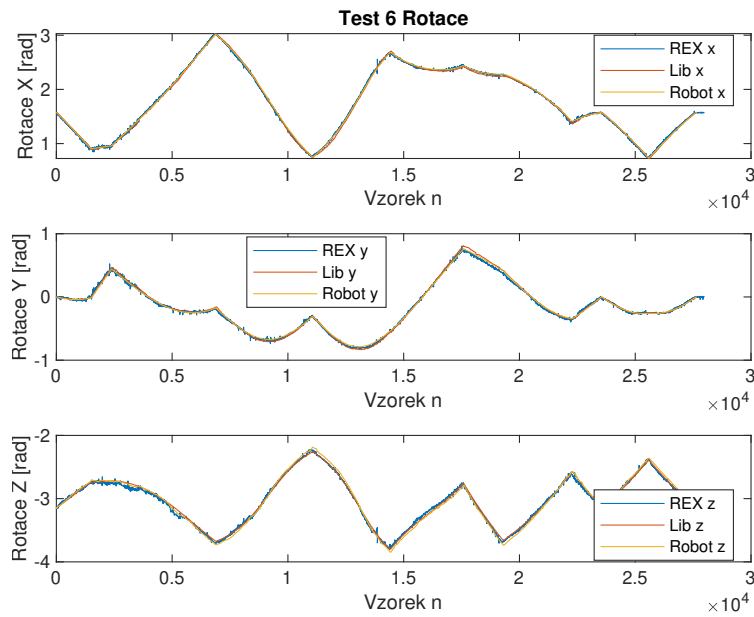
Odměřená rotace navrženého filtru na rozdíl od translace téměř neobsahuje překmity, místy je ale více zašuměná (Obr.5.7, osa Y,  $n=3500$ ). Přesnost sledování mezi oběma filtry je velmi podobná, ve smyslu zvoleného kritéria však navržený filtr dosáhl lepších výsledků. Doplnkové grafy translace jsou v příloze A.2.

### 5.2.3 Obecné lineární pohyby

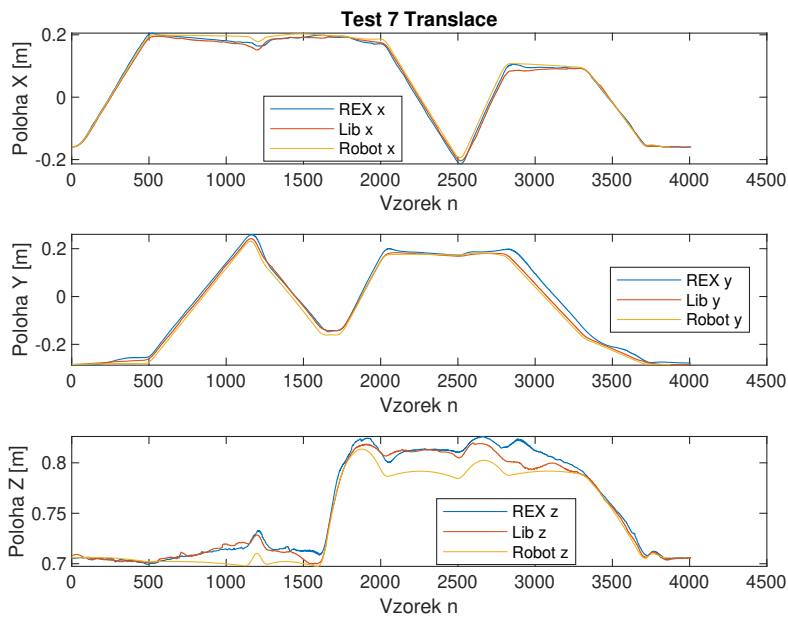
V šestém a sedmém experimentu byla zkombinována translace a rotace do obecného lineárního pohybu manipulátoru. Mezi těmito experimenty lze u obou algoritmů pozorovat zhoršení kvality sledování translace ve smyslu zvoleného kritéria. Překmity Y-ové složky navrženého filtru se zde však snížily oproti experimentu č. 3(5.5). V Z-ové složce translace se oba filtry opět odchyľují od reference během malých pohybů manipulátoru. Špatné sledování malých a pomalých translačních pohybů je tedy opakujícím se trendem. U navrženého filtru lze také pozorovat zhoršení sledování rotace oproti experimentům č. 4 a 5 (5.6,5.7). Výsledky pokusů jsou zobrazeny na obrázku 5.8, 5.9, 5.10, 5.11.



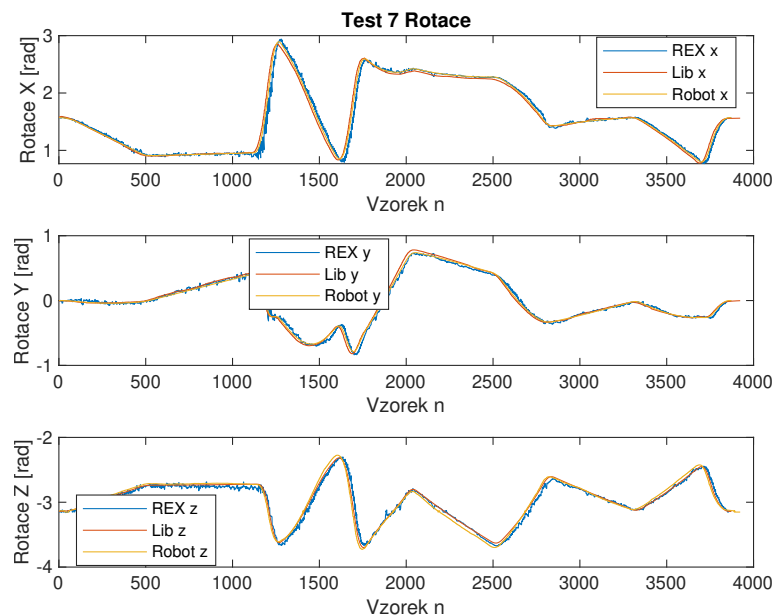
Obrázek 5.8: Porovnání přesnosti sledování, šestá trajektorie, Translace,  $IAE_{REX6-T} = 0.023$ ,  $IAE_{Lib6-T} = 0.008$



Obrázek 5.9: Porovnání přesnosti sledování, šestá trajektorie, Rotace,  $IAE_{REX6-R} = 0.013$ ,  $IAE_{Lib6-R} = 0.015$



Obrázek 5.10: Porovnání přesnosti sledování, sedmá trajektorie, Translace,  $IAE_{REX7-T} = 0.025$ ,  $IAE_{Lib7-T} = 0.007$

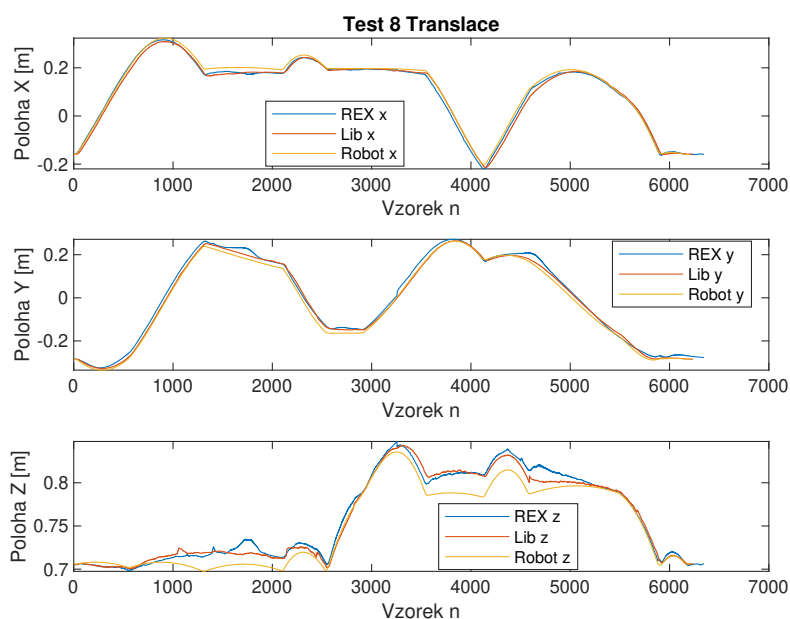


Obrázek 5.11: Porovnání přesnosti sledování, sedmá trajektorie, Rotace,  $IAE_{REX7-R} = 0.018$ ,  $IAE_{Lib7-R} = 0.006$

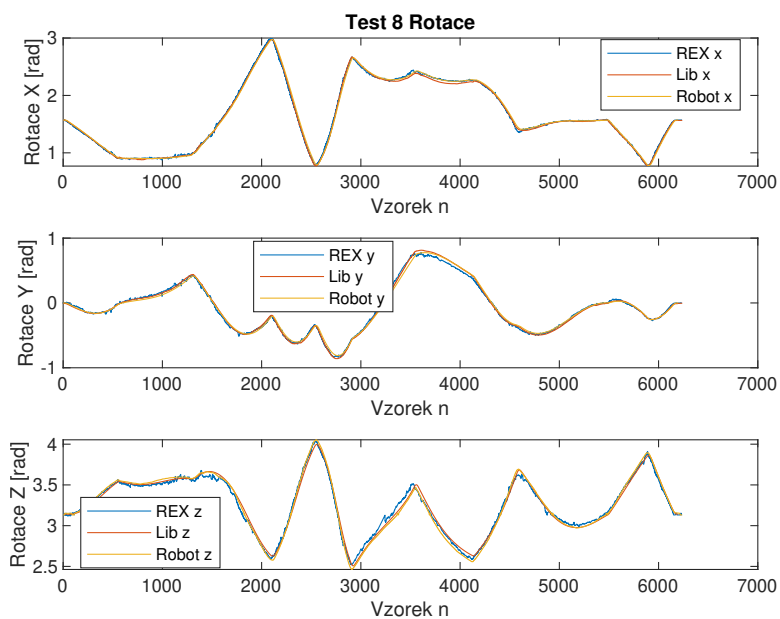
Z provedených testů vychází, že navržený filtr překoná interní filtr ve smyslu zvoleného kritéria pouze v jednom případě (5.9). V ostatních případech vlivem překmitů a šumu nedokáže navržený filtr dosáhnout lepších výsledků než interní.

## 5.2.4 Obecné kloubové pohyby

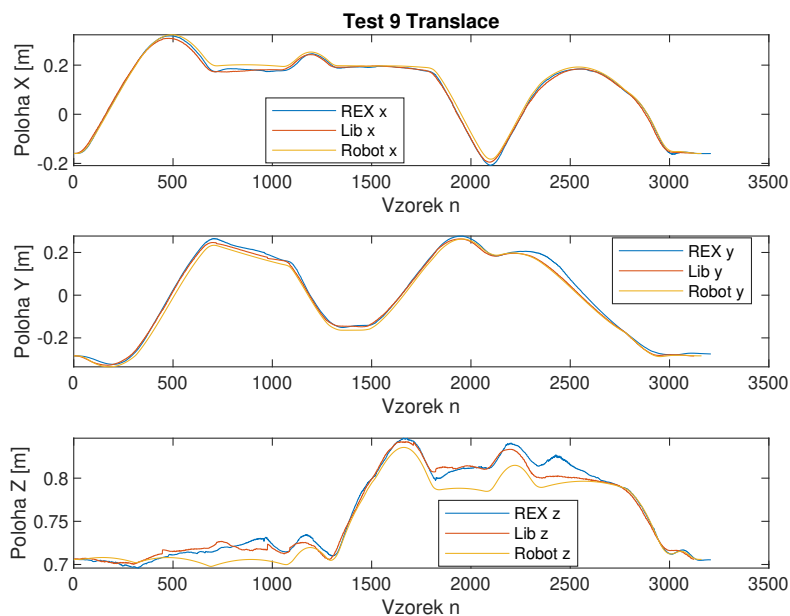
Poslední experimenty obsahovaly pouze rotaci v kloubech manipulátoru, výsledná trajektorie má tedy mnohem více změn vzhledem k souřadnicím  $XYZ$ . Domněnka o horším sledování reference ve smyslu zvoleného kritéria při rychlejších pohybech se zde potvrdila, jelikož výsledná kritéria jsou vyšší než u předešlých experimentů s pomalým pohybem. Z translačních dat experimentu č.8 a 9 lze na ose  $Z$  vidět výraznější odchylku od reference. Zaznamenané trajektorie translace a rotace jsou na obrázku 5.12, 5.13, 5.14 a 5.15.



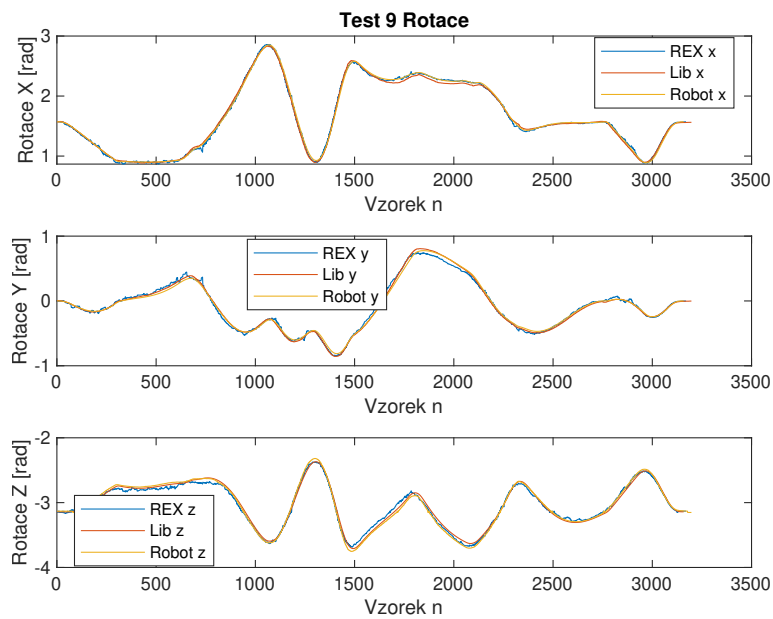
Obrázek 5.12: Porovnání přesnosti sledování, osmá trajektorie, Translance,  $IAE_{REX8-T} = 0.021$ ,  $IAE_{Lib8-T} = 0.008$



Obrázek 5.13: Porovnání přesnosti sledování, osmá trajektorie, Rotace,  $IAE_{REX8-R} = 0.002$ ,  $IAE_{Lib8-R} = 0.014$



Obrázek 5.14: Porovnání přesnosti sledování, devátá trajektorie, Translace,  $IAE_{REX9-T} = 0.020$ ,  $IAE_{Lib9-T} = 0.006$



Obrázek 5.15: Porovnání přesnosti sledování, devátá trajektorie, Rotace,  $IAE_{REX9-R} = 0.009$ ,  $IAE_{Lib9-R} = 0.015$

I když se celková hodnota kritéria obou filtrů pro rychlejší pohyby zvýšila, interní filtr dosahoval pro rotaci nižších hodnot kritéria než navržený filtr. Průměrná chyba sledování rotace během experimentu č. 8 byla pouze 0.002rad. Ve sledování translace dosáhl však interní filtr opět lepších výsledků. Soupis všech zjištěných kritérií je sepsán v tabulce 6.2.

Po implementaci filtračního algoritmu, zhodnocení výsledků z kapitoly 5.2 a porovnání hodnot kritérií z tabulky 6.2 bylo přistoupeno k závěru, že navržený filtrační algoritmus s aktuálním nastavením kovariančních matic šumů nedokáže oproti internímu filtru knihovny **Libsurvive** zlepšit přesnost on-line sledování translace pracovního nástroje ve smyslu minimalizace zvoleného kritéria. Hlavním důvodem nižší přesnosti je obtížné nastavení kovariančních matic stavových a výstupních šumů uvnitř Kálmánova filtru, které způsobují neoptimální filtraci a tím pádem horší výsledky. Navržený filtr však dosáhl v osmi z devíti experimentů lepších výsledků při sledování orientace než interní filtr.

Dále byl porovnán test č. 7 s testem *TEST\_1\_5* z [Šve+22] (Tabulka 6.1). Tyto testy byly vybrány, jelikož se u obou jedná o lineární pohyby s přibližně stejnými rychlostmi a zrychleními. Z porovnání lze usoudit, že filtr navržený v této práci dosáhl lepší přesnosti ve sledování rotace, průměrná chyba sledování translace pro navržený filtr však byla o  $0.021\text{ m}$  vyšší.

-	Průměrná chyba translace [m]	Průměrná chyba rotace [rad]
Navržený filtr	0.025	0.018
TEST_1_5	0.004	0.021

Tabulka 6.1: Porovnání přesnosti vybraných testů

Navržený filtr ale obsahuje detekci dostupnosti optického měření a tedy dokáže lépe odstranit chybné měření a špičky v signále. Při výpadku optického měření zle pomocí vnitřního modelu predikovat budoucí polohu a například tím zabránit kolizi.



Č. experimentu	REX trans [m]	Lib trans [m]	REX rot [rad]	Lib rot [rad]
1	0.008	0.005	0.003	0.008
2	0.009	0.005	0.004	0.004
3	0.008	0.0003	0.003	0.004
4	0.004	0.002	0.006	0.015
5	0.006	0.001	0.003	0.019
6	0.023	0.008	0.013	0.015
7	0.025	0.007	0.018	0.006
8	0.021	0.008	0.002	0.014
9	0.020	0.006	0.009	0.015

Tabulka 6.2: Výsledné hodnoty IAE kritérií pro jednotlivé experimenty

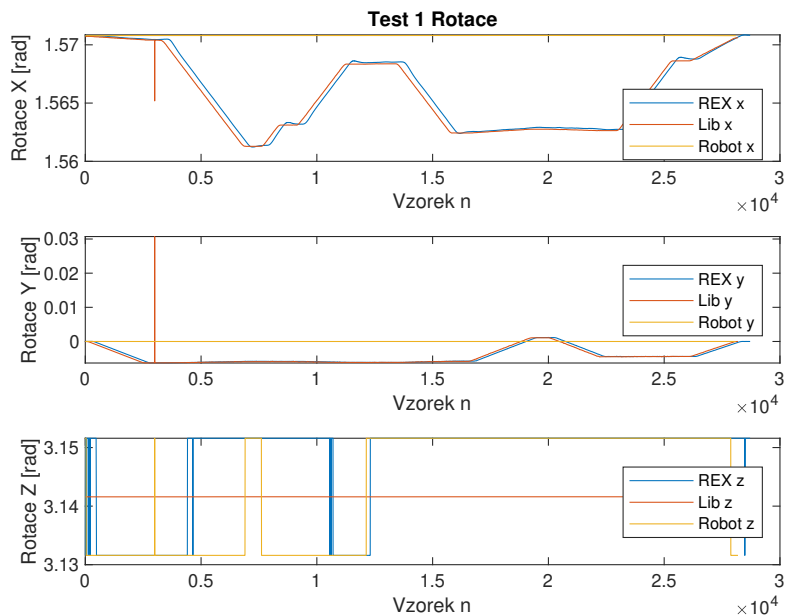
Další výhodou takto navrženého algoritmu je implementace většiny postupů do prostředí **REXYGEN**. Algoritmus tak poskytuje plnou modularitu pro budoucí používání v ostatních aplikacích vyžadující sledování pracovního nástroje nebo bodu v prostoru (např. sledování pohybu dronu v místnosti). Zároveň použitím přenosového protokolu UDP lze přímo a rychle přistupovat k naměřeným datům. Dodatečná přesnost a robustnost sledování by pak mohla být dosažena přidáním více HTC base stanic nebo použitím nekauzálních filtrů na získaná data pro off-line aplikace.

Veškerý vytvořený kód použitý v této práci je dostupný na [Kej].

# První příloha

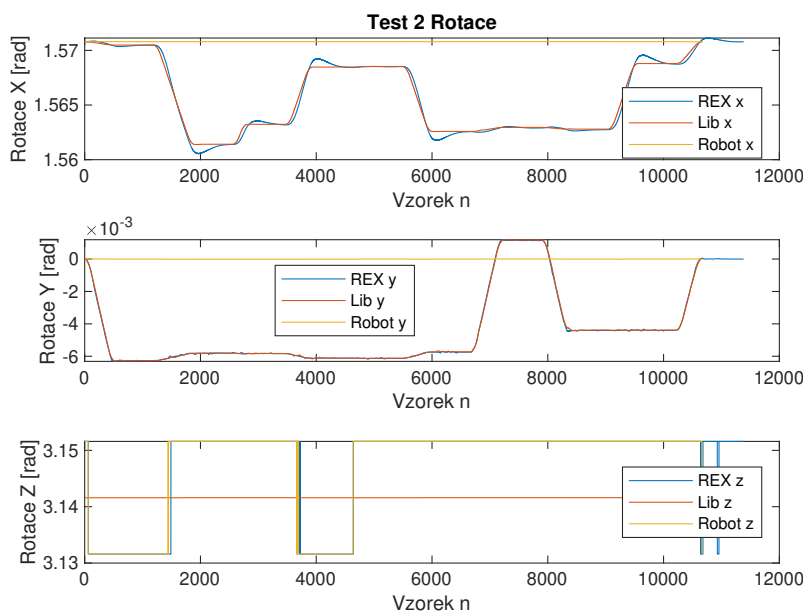
# A

## A.1 Doplnkové grafy k lineárnímu pohybu bez změny rotace

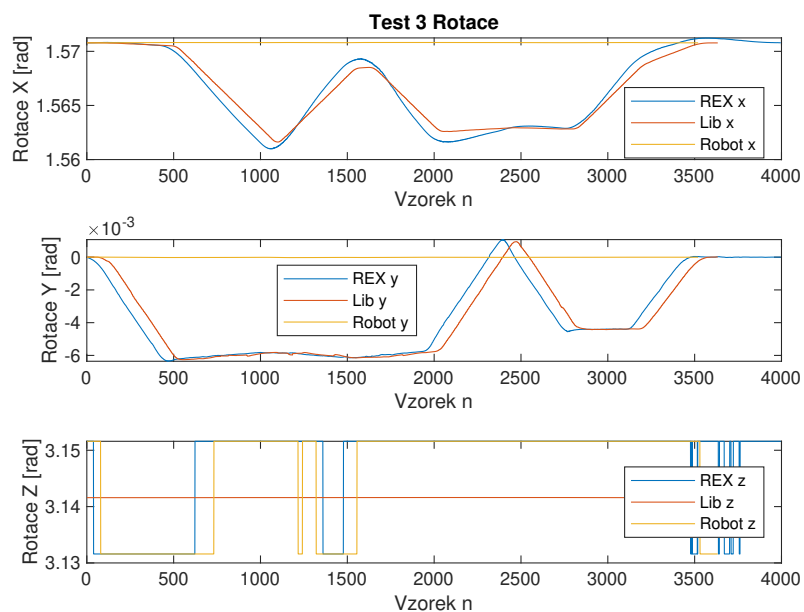


Obrázek A.1: Porovnání přesnosti sledování, první trajektorie, Rotace

### A.1 Doplnkové grafy k lineárnímu pohybu bez změny rotace

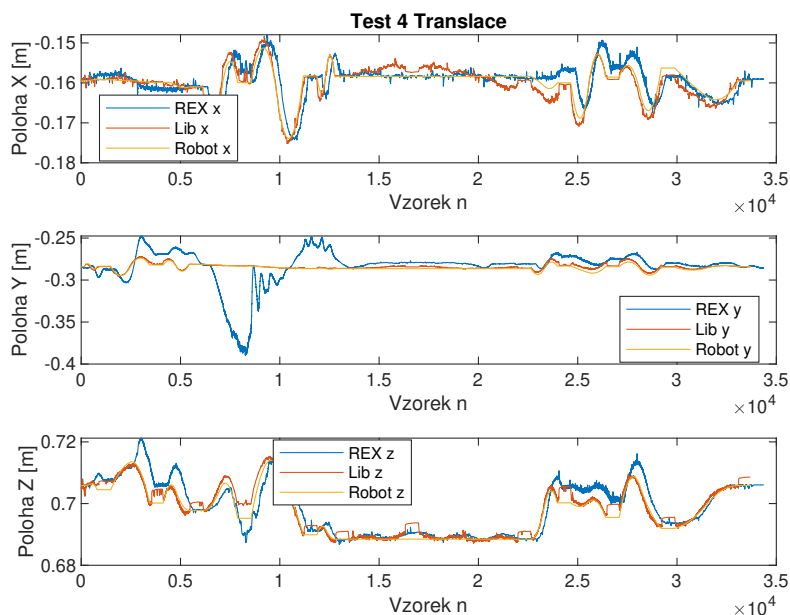


Obrázek A.2: Porovnání přesnosti sledování, druhá trajektorie, Rotace

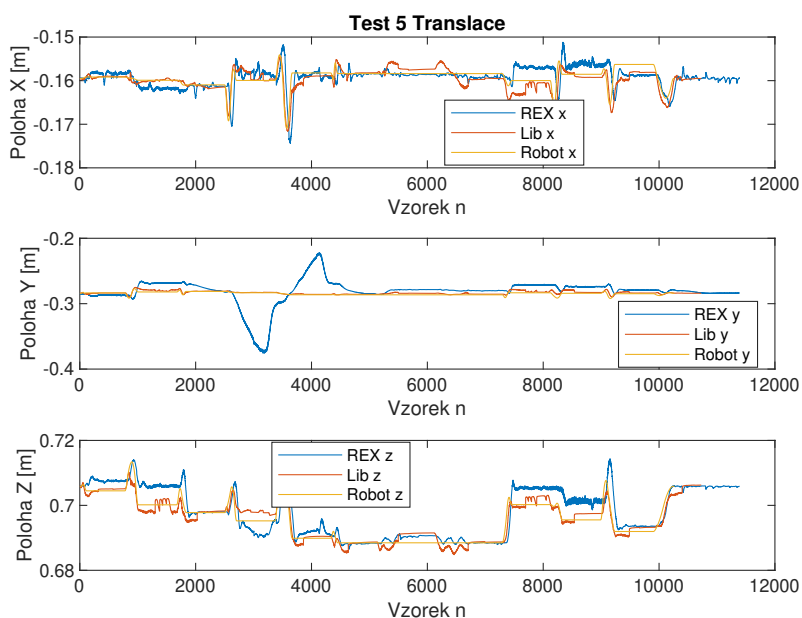


Obrázek A.3: Porovnání přesnosti sledování, třetí trajektorie, Rotace

## A.2 Doplňkové grafy k lineárnímu pohybu bez změny translace



Obrázek A.4: Porovnání přesnosti sledování, čtvrtá trajektorie, Translace



Obrázek A.5: Porovnání přesnosti sledování, pátá trajektorie, Translace

# Bibliografie

- [Ati19] ATIA, Mohamed. Design and simulation of sensor fusion using symbolic engines. *Mathematical and Computer Modelling of Dynamical Systems*. 2019, roč. 25, č. 1, s. 40–62. Dostupné z DOI: 10.1080/13873954.2019.1566266.
- [Bor+18] BORGES, Miguel; SYMINGTON, Andrew; COLTIN, Brian; SMITH, Trey; VENTURA, Rodrigo. HTC Vive: Analysis and Accuracy Improvement. In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2018, s. 2610–2615. Dostupné z DOI: 10.1109/IROS.2018.8593707.
- [Kej] KEJVAL, Filip. *Zdrojové kódy k bakalářské práci*. [B.r.]. Dostupné také z: [https://drive.google.com/drive/folders/1Pktv2Y1aMCsoRnnq2-1jAeHuY6R0tNq2?usp=drive\\_link](https://drive.google.com/drive/folders/1Pktv2Y1aMCsoRnnq2-1jAeHuY6R0tNq2?usp=drive_link).
- [22] *Libsurvive*. 2022. Dostupné také z: <https://github.com/cntools/libsurvive>.
- [24a] *Rexygen documentation*. 2024. Dostupné také z: <https://www.rexygen.com/documentation-and-support/>.
- [24b] *Rexygen website*. 2024. Dostupné také z: <https://www.rexygen.com/>.
- [Šve17] ŠVEJDA, Martin. *Reprezentace obecného pohybu v robotice*. 2017. Dostupné také z: [https://home.zcu.cz/~msvejda/URM/prednasky/prednaska\\_2/prezentace.pdf](https://home.zcu.cz/~msvejda/URM/prednasky/prednaska_2/prezentace.pdf).
- [Šve+22] ŠVEJDA, Martin; GOUBEJ, Martin; JÁGER, Arnold; REITINGER, Jan; SEVERA, Ondřej. Affordable Motion Tracking System for Intuitive Programming of Industrial Robots. *Sensors*. 2022, roč. 22, č. 13. ISSN 1424-8220. Dostupné z DOI: 10.3390/s22134962.

# Seznam obrázků

2.1	Schéma komunikace . . . . .	5
3.1	Souřadné systémy . . . . .	8
3.2	Průběh translace . . . . .	12
3.3	Průběh rotace . . . . .	13
4.1	Schéma zapojení EKF a REXLANG . . . . .	17
5.1	Souřadný systém HTC Trackeru . . . . .	19
5.2	Souřadné systémy během experimentu, vlevo robot, vpravo HTC base stanice . . . . .	20
5.3	Porovnání přesnosti sledování, první trajektorie, Translace, $IAE_{Lib1-T} = 0.008$ , $IAE_{REX1-T} = 0.005$ . . . . .	23
5.4	Porovnání přesnosti sledování, druhá trajektorie, Translace, $IAE_{Lib2-T} = 0.009$ , $IAE_{REX2-T} = 0.005$ . . . . .	24
5.5	Porovnání přesnosti sledování, třetí trajektorie, Translace, $IAE_{Lib3-T} = 0.008$ , $IAE_{REX3-T} = 0.0003$ . . . . .	24
5.6	Porovnání přesnosti sledování, čtvrtá trajektorie, Rotace, $IAE_{REX4-R} = 0.006$ , $IAE_{Lib4-R} = 0.015$ . . . . .	25
5.7	Porovnání přesnosti sledování, pátá trajektorie, Rotace, $IAE_{REX5-R} = 0.003$ , $IAE_{Lib5-R} = 0.019$ . . . . .	26
5.8	Porovnání přesnosti sledování, šestá trajektorie, Translace, $IAE_{REX6-T} = 0.023$ , $IAE_{Lib6-T} = 0.008$ . . . . .	27
5.9	Porovnání přesnosti sledování, šestá trajektorie, Rotace, $IAE_{REX6-R} = 0.013$ , $IAE_{Lib6-R} = 0.015$ . . . . .	27
5.10	Porovnání přesnosti sledování, sedmá trajektorie, Translace, $IAE_{REX7-T} = 0.025$ , $IAE_{Lib7-T} = 0.007$ . . . . .	28
5.11	Porovnání přesnosti sledování, sedmá trajektorie, Rotace, $IAE_{REX7-R} = 0.018$ , $IAE_{Lib7-R} = 0.006$ . . . . .	28
5.12	Porovnání přesnosti sledování, osmá trajektorie, Translace, $IAE_{REX8-T} = 0.021$ , $IAE_{Lib8-T} = 0.008$ . . . . .	29

---

5.13	Porovnání přesnosti sledování, osmá trajektorie, Rotace, $IAE_{REX8-R} = 0.002$ , $IAE_{Lib8-R} = 0.014$ . . . . .	30
5.14	Porovnání přesnosti sledování, devátá trajektorie, Translace, $IAE_{REX9-T} = 0.020$ , $IAE_{Lib9-T} = 0.006$ . . . . .	30
5.15	Porovnání přesnosti sledování, devátá trajektorie, Rotace, $IAE_{REX9-R} = 0.009$ , $IAE_{Lib9-R} = 0.015$ . . . . .	31
A.1	Porovnání přesnosti sledování, první trajektorie, Rotace . . . . .	34
A.2	Porovnání přesnosti sledování, druhá trajektorie, Rotace . . . . .	35
A.3	Porovnání přesnosti sledování, třetí trajektorie, Rotace . . . . .	35
A.4	Porovnání přesnosti sledování, čtvrtá trajektorie, Translace . . . . .	36
A.5	Porovnání přesnosti sledování, pátá trajektorie, Translace . . . . .	36

# Seznam tabulek

6.1	Porovnání přesnosti vybraných testů . . . . .	32
6.2	Výsledné hodnoty IAE kritérií pro jednotlivé experimenty . . . . .	33



