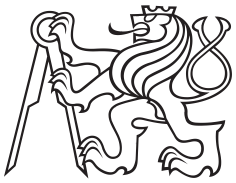
Bakalářská práce



České
vysoké
učení technické
v Praze

F3

Fakulta elektrotechnická

# Ovládání průmyslového robotu za pomoci systému pro virtuální realitu

Ondřej Mikoláš

Vedoucí: Ing. Vladimír Smutný, Ph.D.
Studijní program: Kybernetika a robotika
Květen 2023

# ZADÁNÍ BAKALÁŘSKÉ PRÁCE

## I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Mikoláš** Jméno: **Ondřej** Osobní číslo: **492319**

Fakulta/ústav: **Fakulta elektrotechnická**

Zadávající katedra/ústav: **Katedra kybernetiky**

Studijní program: **Kybernetika a robotika**

## II. ÚDAJE K BAKALÁŘSKÉ PRÁCI

Název bakalářské práce:

**Ovládání průmyslového robotu za pomoci systému pro virtuální realitu**

Název bakalářské práce anglicky:

**Industrial Robot Control Using Virtual Reality Tools**

Pokyny pro vypracování:

Cílem bakalářské práce je vytvořit ovládání pohybů robotu KUKA iiwa se sedmi stupni volnosti za pomoci ovladačů dostupných v systému HTC VIVE.
1. Seznamte se s robotem KUKA iiwa, jeho kinematikou a jeho začleněním do systému Robot Operating System (ROS).
2. Seznamte se se systémem pro virtuální realitu HTC VIVE a jeho drivery pro ROS.
3. Seznamte se se systémy pro interaktivní ovládání průmyslových manipulátorů.
4. Navrhněte, implementujte a vyzkoušejte online ovládání polohy chapadla robotu za pomoci ovladače HTC VIVE. Řešte problém rozdílné dynamiky pohybu ovladače rukou a pohybu chapadla robotem.
5. Pokuste se v rámci možností využít všech 7 os robotu pro zlepšení vlastností systému.
6. Využijte možností systému HTC VIVE pro zpětnou vazbu pro obsluhu.
7. Vyvinutý systém otestujte na skupině minimálně 5 osob, získejte od nich zpětnou vazbu a výsledky zhodnoťte.

Seznam doporučené literatury:

[1] Luigi Gammieri, Marco Schumann, Luigi Pelliccia, Giuseppe Di Gironimo, Philipp Klimant: Coupling of a Redundant Manipulator with a Virtual Reality Environment to Enhance Human-robot Cooperation, Procedia CIRP, Volume 62, 2017, Pages 618-623, ISSN 2212-8271, https://doi.org/10.1016/j.procir.2016.06.056
[2] Theodoros Togias, Christos Gkournelos, Panagiotis Angelakis, George Michalos, Sotiris Makris: Virtual reality environment for industrial robot control and path design, Procedia CIRP, Volume 100, 2021, Pages 133-138, ISSN 2212-8271, https://doi.org/10.1016/j.procir.2021.05.021.
[3] Alexis Koopmann, Kressa Fox, Phillip Raich, Jenna Webster: Virtual Reality Teleoperation Robot, ECE 3992 COMPUTER ENGINEERING SENIOR THESIS 26, DECEMBER 2020
[4] Teleoperating robots with virtual reality, https://robotics.mit.edu/teleoperating-robots-virtual-reality

Jméno a pracoviště vedoucí(ho) bakalářské práce:

**Ing. Vladimír Smutný, Ph.D. robotické vnímání CIIRC**

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

Datum zadání bakalářské práce: **31.01.2023** Termín odevzdání bakalářské práce: **26.05.2023**

Platnost zadání bakalářské práce: **22.09.2024**

| Ing. Vladimír Smutný, Ph.D. | prof. Ing. Tomáš Svoboda, Ph.D. | prof. Mgr. Petr Páta, Ph.D. |
|---|---|---|
| podpis vedoucí(ho) práce | podpis vedoucí(ho) ústavu/katedry | 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

# Poděkování

Rád bych poděkoval panu Ing. Vladimíru Smutnému, Ph.D. za trpělivost a veškerou pomoc, která umožnila vznik této bakalářské práce. Dále bych chtěl poděkovat své rodině a dalším blízkým za podporu během studia.

# Prohlášení

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

V Praze, 26. května 2023

# Abstrakt

Tato práce se zabývá vytvořením aplikace pro ovládání chapadla robota Kuka LBR iiwa 7 pomocí pohybů ovladače HTC Vive, který operátor drží v ruce. Návrhované ovládání je zaměřeno na jednoduchost, pochopitelnost a intuitivnost. Pro práci se systémem HTC Vive je použita knihovna Libsurvive a pro ovládání robota je použita knihovna Capek Robot Commander. Systémy jsou propojeny pomocí vícevláknového programování, kde každý systém funguje ve svém vlákně. Výsledné ovládání nabízí různá nastavení jako jsou módy, poměrová hodnota a připnutí na nejbližší ortogonální úhel. Pro snadnou konfiguraci byl vytvořen grafický panel využívající knihovnu Raylib. Práce také zahrnuje testování aplikace na skupině 5 osob, které vyplnily dotazník zaměřený na pochopení ovládání a intuitivnost aplikace. Výsledky ukazují, že aplikace se jeví jednoduchá a pochopitelná, nicméně její hlavní nedostatek je velké dopravní zpoždění, které znemožňuje rychlé ovládání robota.



**Klíčová slova:** HTC Vive, vzdálené ovládání, Kuka LBR iiwa

**Vedoucí:** Ing. Vladimír Smutný, Ph.D.



# Abstract

This work deals with the creation of an application for controlling the end effector of the Kuka LBR iiwa 7 robot using movements of the HTC Vive controller held by the operator. The proposed control is focused on simplicity, comprehensibility, and intuitiveness. The Libsurvive library is used for working with the HTC Vive system, and the Capek Robot Commander library is used for robot control. The systems are interconnected through multithreaded programming, with each system functioning in its own thread. The resulting control offers various settings such as modes, ratio values, and snapping to the nearest orthogonal angle. For easy configuration, a graphical panel utilizing the Raylib library has been created. The work also includes testing the application with a group of 5 individuals who filled out a questionnaire focused on understanding the control and the intuitiveness of the application. The results demonstrate that the application is rather easy to use and easy to understund, however the main drawback is high latency, that makes fast control of the robot impossible.



**Keywords:** HTC Vive, teleoperation, Kuka LBR iiwa

**Title translation:** Industrial Robot Control Using Virtual Reality Tools

# Obsah

# Obrázky



# Tabulky

# Kapitola 1
# Úvod

V dnešní době se roboti používají hlavně v provozu s opakujícími se činnostmi. Jsou tedy automatizované a nepotřebují lidského operátora.

Nicméně složitější úkoly již nemusí být jednoduché automatizovat, jelikož potřebují jistou úroveň pochopení zadaného úkolu. V těchto případech musí práci vykonat člověk. V případě, že je ale práce moc těžká nebo člověku nebezpečná, je lepší, když je vykonána robotem, který je řízený člověkem.

Jedním z nejčastějších způsobů řízení je kontrola jednotlivých kloubů robota zvlášť. Takto funguje například ovládání bagrů.

Dalším častým způsobem je kontrola pohybů robota v souřadné soustavě, tedy pohyby se zadávají jako poloha $x$,$y$ a $z$ a ta se přepočítá do poloh kloubů robota. Toto řízení může vypadat například jako mačkání tlačítek pro jednotlivé pohyby v $x$-ové, $y$-ové a $z$-ové ose, popřípadě i pro rotaci robota. Nicméně pohyby robota jsou omezeny jen na tyto pohyby. Pro složitější pohyby už musí operátor použít složitější formu ovládání.

Cílem této práce je vytvořit aplikaci pro ovládání robotického manipulátoru Kuka LBR iiwa 7 pomocí systému pro virtuální realitu HTC Vive. Cílem je pohybovat chapadlem robota podle pohybů ovladače, který operátor robota drží v ruce. Výsledné ovládání by mělo být jednoduché, pochopitelné a co nejintuitivnější. Dále se zabývá rozšířením ovládání o dodatečné uživatelské funkce. Vyvinuté ovládání se otestuje na skupině alespoň 5 lidí.

# Kapitola 2

# Aktuální stav technologií pro teleoperaci a využití systému virtuální reality v robotice

## 2.1 Roboticky asistované operace

Jedním z nejběžnějších případů vzdáleného ovládání robotů v současnosti je tzv. roboticky asistovaná chirurgie. Robotické operace jsou miniinvazivní zákroky, které umožňují chirurgický výkon jako při klasické operaci, ale z minimálního přístupu (minimálního otevření těla). Rekonvalescence a návrat do běžného života jsou po robotickém zákroku velmi rychlé na rozdíl od klasické operace. Nejznámějším systémem je robota Da Vinci. Robot Da Vinci můžete vidět na obrázku 2.1. Dle [3] jen v České republice provedl DaVinci několik tisíc operací od roku 2009.

**Obrázek 2.1:** Robot Da Vinci [1]

Robot Da Vinci má 4 pohyblivá ramena, která drží chirurgické nástroje a kamerku. Konec ramen se zavede do těla pacienta a chirurg ovládá ramena vzdáleně. Ramena mají 6 stupňů volnosti. Můžou se pohybovat všemi směry a umožňují 360 stupňový pohyb nástroje. Druhou částí robota je ovládací konzole. Chirurg u ní sedí, dívá se do průzoru, kde vidí 3D obraz snímaný kamerami uvnitř těla pacienta. Joystickem ovládá robotická ramena. Robotický systém naprosto eliminuje třes a vylepšuje pohyby chirurga.

## 2.2 Životu nebezpečné práce

Další běžné využití teleoperace je v situacích, kdy je potřeba vykonat člověku nebezpečná práce, například zneškodnění výbušnin. Nejčastěji se používá pásové vozítko s připevněnou robotickou paží jako na obrázku 2.2. Vozítko má za úkol dojet k podezřelému předmětu a pomocí robotické paže zneškodnit výbušninu. Celý robot se ovládá dálkově. Pohyby robotické paže se běžně ovládají pomocí joysticku.

**Obrázek 2.2:** Robot pro zneškodnění výbušnin [2]

Nicméně již existují řešení, která ovládání robotické paže zjednodušují. Například společnost REZ vyvinula systém [4], kdy operátor řídí paže na robotovi pomocí modelu této paže. Tedy operátor pohybuje s modelem robotické paže z bezpečné vzdálenosti a tím ovládá robotickou paži na vozítku. To jak se chová ovládaná paže vidí pomocí kamerového systému, jímž je vozítko vybaveno.

## 2.3 Využití virtuální reality v robotice

Systém virtuální reality se omezeně používá pro vzdálené ovládání průmyslových robotů robotů, nicméně ve většině případech začlenění virtuální reality do robotických systému se virtuální realita používá především jako pomůcka pro práci s virtuálním robotem ve virtuálním prostředí [9]. Operátor používá systém virtuální reality pro pohyb ve virtuálním pracovním prostředí a může ovládat virtuálního robota pomocí ovladačů virtuální reality. Práce ve virtuálním prostředí slouží například k zaučení nových pracovníků nebo k vyzkoušení chování virtuálního robota před testem s reálným hardwarem. Prací s virtuálním robotem lze předejít ublížení na zdraví operátora a možným škodám na reálném hardwaru v případě chyby programování robota.

# Kapitola 3
# Použité nástroje

## 3.1 ROS-Robot Operating System

Robot Operating System je open source soubor knihoven a nástrojů, které pomáhají vytvářet aplikace pro roboty. ROS má modulární architekturu, což zjednodušuje přidávání nových funkci a algoritmů.

## 3.2 Headset pro virtuální realitu HTC Vive

HTC Vive je headset pro virtuální realitu vyvinutý firmami HTC a Valve corporation. Skládá se z ovladačů a headsetu samotného, který si uživatel nasadí přes hlavu. Ovladače si uživatel vezme do rukou. Headset má 2 obrazovky, jednu pro každé oko a je schopný promítat 2 různé obrazy, uživatel může tedy vidět trojrozměrně. Ovladače slouží k interakci uživatele s virtuálním prostředím. Headset i ovladače mají na různých místech na sobě několik senzorů infračerveného světla. Pro zjištění polohy headsetu a ovladačů se používají majáky. Majáky promítají roviny infračerveného světla do prostoru, ve kterém se uživatel pohybuje. Tyto roviny světla jsou detekovány senzory na těle headsetu a ovladačů a následně je dopočítána jejich poloha a orientace v prostoru.

## 3.3 Knihovna libsurvive

Libsurvive [5] je soubor nástrojů a knihoven, které umožňují sledování polohy a orientace zařízení jako je HTC Vive. Libsurvive má k dispozici několik API [Aplikační interface], pro jednodušší použití. V této práci je využíváno API pro programovací jazyk C++. Kromě zjišťování polohy a orientace HTC Vive zařízeních umí Libsurvive také pracovat s ovládacími prvky na ovladači. Celý kód této knihovny je veřejně k dispozici.

## 3.4 Knihovna Raylib

Raylib [8] je open-source grafická knihovna pro C++ a další programovací jazyky. Tato knihovna poskytuje jednoduché rozhraní pro tvorbu 2D a 3D grafiky, zvukových efektů a interaktivních aplikací. Raylib využívá moderní technologie jako je OpenGL ES 2.0, Vulkan, stínování a podporuje platformy Windows, Linux, MacOS a Android. Jednou z hlavních výhod Raylibu je jeho snadná použitelnost a rychlost, což umožňuje rychlé prototypování a vývoj grafických aplikací.

## 3.5 Robot Kuka LBR iiwa 7 R800 CR

Robot KUKA LBR iiwa 7 R800 [7] je jeden z lehkých kobotů společnosti KUKA, který je specializovaný na jemnou montážní práci. Je to první sériově vyráběný citlivý a HRC-kompatibilní robot na světě. LBR znamená "Leichtbauroboter" (německy lehký robot), iiwa znamená "inteligentní průmyslový pracovní asistent". Robot LBR iiwa 7 R800 může spolupracovat s lidmi na vysoce citlivých úkolech v těsné spolupráci, což umožňuje nové aplikace a větší hospodárnost a efektivitu. Robot má nosnost 7 kilogramů a dosah 800 milimetrů.

# Kapitola 4

# Práce se systémem HTC Vive

## 4.1 Softwarové nástroje pro práci se systémem HTC Vive

### 4.1.1 OpenVR SDK

Pro práci se systém HTC Vive poskytuje společnost Valve oficiální sadu vývojových nástrojů OpenVR SDK [6]. OpenVR poskytuje možnost trackování zařízení HTC Vive. Také poskytuje integraci v několika herních enginech, pomocí kterých může vytvořit 3D virtuální prostředí a to zobrazovat v Headsetu HTC Vive.

OpenVR je robustní a spolehlivá sada nástrojů, nicméně pro svou funkci potřebuje další sadu nástrojů SteamVR a aplikaci Steam.

Systém HTC Vive je určen převážně pro herní průmysl a to se odráží v dokumentaci pro OpenVR. Ta se zaměřuje především na použití OpenVR pro vývoj her, to trochu komplikuje integraci OpenVR do přostředí pro práci s roboty, jako například ROS. Nicméně již existuje hotové řešení, které používá herní engine Unity a propojuje ROS se systémem HTC Vive. Jmenuje se ROS Reality [10] a využívá jak tracking zařízení HTC Vive, tak i promítání virtuálního prostředí do headsetu HTC Vive.

### 4.1.2 Libsurvive vs OpenVR SDK

Knihovna Libsurive na rozdíl od OpenVR poskytuje jen možnost trackování a interakce s ovládacími prvky ovladače HTC Vive. Díky tomu Libsurvive zabírá mnohem méně místa na disku oproti OpenVR a nepotřebuje žádné další nástroje nebo aplikace pro použití.

V mém případě jsem zvolil Libsurvive. Hlavním důvodem této volby byla jednodušší integrace do ROSu. Dále je výhodou to, že Libsurvive lze provozovat na mnohem více zařízeních oproti OpenVR, kvůli vyšším nárokům OpenVR na další sadu nástrojů a aplikaci Steam. Libsurvive je navíc mnohem kompaktnější knihovna, potřebuje jen zhruba 32 MB na disku. OpenVR potřebuje zhruba 1,5 GB místa na disku, a potřebuje navíc další softwarové nástroje. Knihovna libsurvive obsahovala mnoho chyb a občas byla nestabilní,

ale neustále se zvyšuje spolehlivost a stabilita této knihovny. V Únoru 2022 byla vydána verze 1.0, tato verze se zaměřovala hlavně na zlepšení stability.

## 4.2 Zpracování dat ze systému HTC Vive

### 4.2.1 Zpracování polohy

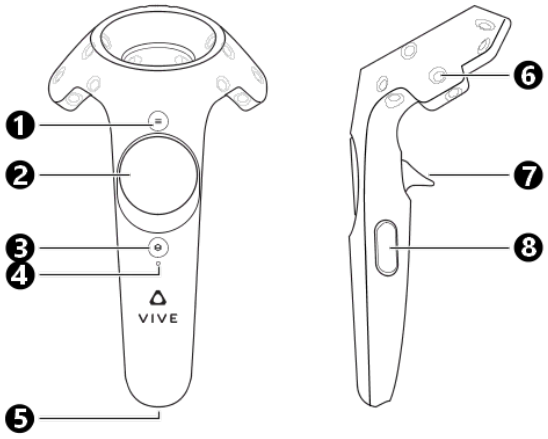
Knihovna Libsurvive poskytuje polohu ve formě vektoru $t = [x, y, z]^T$ a natočení v prostoru ve formě kvaternionu $q = [w, x, y, z]^T$. Použití kvaternionu je výhodou pro pozdější počítání s úhly, jelikož pro sečtení dvou orientaci ve formě kvaternionu stačí dané kvaterniony vynásobit pomocí násobení kvaternionů. Pro zjištění rozdílu orientace $q_{diff}$ mezi dvěma orientacemi vyjádřených kvaterniony $q_1$ a $q_2$ se počítá :

$$q_{diff} = q^{-1} \cdot q$$

, kde $q^{-1}$ značí inverzi kvaternionu $q$. $q^{-1}$ reprezentuje opačnou rotaci než $q$.

### 4.2.2 Zpracování interakce s ovládacími prvky

Na ovladači jsou k dispozici 3 tlačítka, jeden trigger, což je tlačítko, které umí snímat intenzitu (polohu) zmáčknutí a nakonec kruhový touchpad (dotyková plocha) spojený s tlačítkem.




| | |
|---|---|
| 1 | Tlačítko menu |
| 2 | Touchpad |
| 3 | Systémové tlačítko |
| 4 | Světélko stavu |
| 5 | Nabíjecí port |
| 6 | IR senzor |
| 7 | Trigger |
| 8 | Tlačítko stisku dlaně |

**Obrázek 4.1:** Periferie ovladače HTC Vive [12]

Knihovna Libsurvive poskytuje typ události přímo pro ovládací prvky ovladače. Pokud by někdo chtěl pracovat pouze s ovládacími prvky ovladače může ze všech přijatých událostí od vlákna Libsurvive vyfiltrovat pouze události související s ovládacími prvky. Každá událost poskytuje několik údajů o daném ovládacím prvku a o akci, která s ním byla vykonána. Bohužel, jsem nebyl schopen najít dokumentaci, která by popisovala souvislost mezi

| Akce | Tlačítko | e..type | b..id | a..count |
|---|---|---|---|---|
| změna míry stisku | trigger | 8 | 255 | 1 |
| úplné stisknutí | trigger | 3 | 0 | 0 |
| úplné povolení | trigger | 2 | 0 | 0 |
| zaznamenán dotyk | touchpad | 5 | 1 | 0 |
| dotyk zmizel | touchpad | 4 | 1 | 0 |
| změna polohy dotyku | touchpad | 8 | 255 | 2 |
| stisknutí tlačítka | touchpad | 3 | 1 | 0 |
| povolení tlačítka | touchpad | 2 | 1 | 0 |
| stisknutí tlačítka | systémové tlačítko | 3 | 3 | 0 |
| povolení tlačítka | systémové tlačítko | 2 | 3 | 0 |
| stisknutí tlačítka | tlačítko menu | 3 | 6 | 0 |
| povolení tlačítka | tlačítko menu | 2 | 6 | 0 |
| stisknutí tlačítka | tlačítko stisku dlaně | 3 | 7 | 0 |
| povolení tlačítka | tlačítko stisku dlaně | 2 | 7 | 0 |

**Tabulka 4.1:** Tabulka hodnot pro ovládací prvky ovladače HTC Vive

ovládacími prvky a poskytnutými údaji. Vytvořil jsem proto tabulku která popisuje všechny možné akce s ovládacími prvky ovladače HTC Vive.

Pro rozlišení různých akci s různými ovládacími prvky jsem použil 3 proměnné poskytované knihovnou Libsurvive: “event_type”, “button_id” a “axis_count”. Různé hodnoty těchto 3 proměnných popisují různé akce s různými ovládacími prvky. Výčet všech možných akcí s jednotlivými ovládacími prvky a jejich hodnoty jsou v tabulce 4.1:

Tyto údaje stačí pro detekci stisknutí tlačítka a povolení tlačítka. Pro detekci intenzity triggeru a dotyku na touchpadu je zapotřebí přečíst další 2 proměnné poskytované knihovnou Libsurvive a to “axis_ids” a “axis_val”. Proměnná “Axis_ids” říká na jakém indexu/indexech pole proměnných “axis_val” je uložená proměnná, kterou potřebuji. Tato proměnná má typ float. Pro trigger je jen jedna proměnná, která říká jak moc je trigger stisknutý (v jaké poloze je), může nabývat hodnot 0 až 1. Když má hodnotu 0, trigger není stisklý vůbec, při hodnotě 1 je trigger stisklý úplně. Pro touchpad jsou 2 hodnoty, které vyjadřují hodnoty x a y, určující bod dotyku na dotykové ploše touchpadu. Kde pro (x,y) = (0,0) platí, že dotyk je detekovaný uprostřed touchpadu.

# Kapitola 5

# Výpočet nové polohy a orientace pro robota

Pro výpočet nové polohy pro robota je nutné zpracovat údaje o poloze a orientaci ovladače HTC Vive. V této kapitole se porovnávají 2 různé metody, které pracují různými způsoby s daty o poloze a orientaci ovladače HTC Vive. Nakonec se předvede detailní výpočet nové polohy a orientace pro robota.

## 5.1 1. metoda

V první metodě se s konstantní periodou $t$ načítají aktuální data o poloze a orientaci ovladače. Vypočítává se rozdíl dvou po sobě načtených dat. Rozdíl je posun a rotace ovladače za jednu periodu $t$. Poloha a orientace ovladače načtená na začátku n-té periody je $C_n$, poloha a orientace ovladače načtená na konci periody je $C_{n+1}$. Posun a rotace $\Delta C_n$ mezi $C_n$ a $C_{n+1}$ se počítá:

$$\Delta C_n = C_{n+1} - C_n \tag{5.1}$$

Nová poloha a orientace robota $R_{n+1}$ se získá pomocí aktuální polohy a orientace robota $R_n$ a vypočteného rozdílu $\Delta C_n$:

$$R_{n+1} = R_n + \Delta C_n \tag{5.2}$$

Aby se robot hýbal jen když uživatel chce, musí uživatel držet stisklé tlačítko na ovladači. Když tlačítko nedrží, rozdíl $\Delta C_n$ se nahradí se nulovými údaji, robot se tedy nikam nepohne, jelikož $R_{n+1} = R_n$.

## 5.2 2. metoda

V této metodě se opět s konstantní periodou $t$ načítají aktuální data o poloze a orientaci ovladače. Když uživatel stiskne tlačítko na ovladači, uloží se aktuální poloha a orientace ovladače $C_1$ a aktuální poloha a orientace robota $R_1$. Po dobu stisknutí tlačítka se nová poloha pro robota počítá následujícím způsobem.

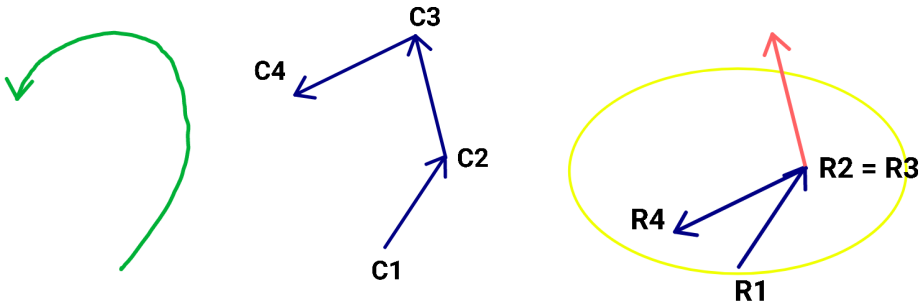
Poloha a orientace ovladače načtená na začátku n-té periody je $C_n$, poloha a orientace ovladače na konci periody je $C_{n+1}$. Při každém načtení polohy a orientace ovladače se vypočte rozdíl $\Delta C_n$ mezi počáteční polohou a orientací

$C_1$ a nově načtenými daty $C_n$. Rozdíl $\Delta C_n$ je posun a rotace z počáteční polohy ovladače do aktuální a počítá se takto:

$$\Delta C_n = C_{n+1} - C_1 \tag{5.3}$$

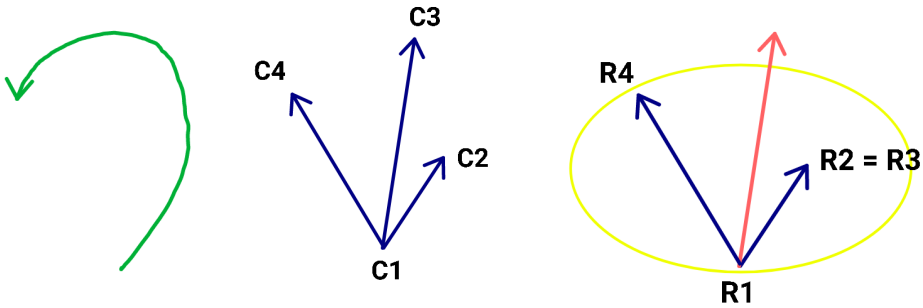
Nová poloha a orientace robota $R_{n+1}$ se získá pomocí počáteční polohy a orientace robota $R_1$ a vypočteného rozdílu $\Delta C_n$:

$$R_{n+1} = R1 + \Delta C_n \tag{5.4}$$




**(a) :** Pohyb ovladače **(b) :** Vypočtený pohyb **(c) :** Nové polohy pro robota

**Obrázek 5.1:** Výpočet pomocí 1. metody




**(a) :** Pohyb ovladače **(b) :** Vypočtený pohyb **(c) :** Nové polohy pro robota

**Obrázek 5.2:** Výpočet pomocí 2. metody

## 5.3 Porovnání metod

Průběh výpočtu nové polohy pro robota pomocí obou metod je znázorněn v rovině na obrázcích 5.1 a 5.2. Kde C1, C2, C3, C4 označují naměřenou polohu ovladače. R1, R2, R3, R4 je poloha robota po výpočtu nových souřadnic pro robota podle jednotlivých metod. Žlutý ovál značí obálku pracovního prostoru robota. To znamená, že robot nemůže dosáhnout vně tohoto oválu. červené šipky označují polohu, do níž se robot nemůže pohnout.

V první metodě každý nový výpočet polohy závisí na předchozím výpočtu. Díky nepřesné reprezentaci čísel v počítači se při každém vypočtení nové polohy může pomalu načítat odchylka. Ta může být velmi malá, ale po dostatečně dlouhé době se může chyba nasčítat a může být už nezanedbatelná. To by se projevilo například tím, že po přesunu ovladače do původního místa, kde pohyb začal, může být malý rozdíl mezi polohou robota a polohou ovladače. To samé se projevuje i ve výpočtu orientace. Tato chyba ale nijak neovlivňuje ovládání, protože si uživatel může polohu robota opravit, tak jak chce.

První metoda navíc nepočítá s tím, že by se robot mohl dostat mimo svůj pracovní prostor. Když se robot nemůže dostat do vypočtené polohy, tak zůstane na místě a čeká na další údaje o poloze. Vzniká tedy odchylka mezi polohou ovladače a polohou robota. Tato chyba je znázorněna na obrázku 5.1c.

Druhá metoda řeší teoreticky oba zmíněné problémy. V této práci používám tedy metodu druhou.

## 5.4 Detailní výpočet nové polohy a orientace pro robota

Výpočet zmíněný v sekci 5.2 pro výpočet nové polohy pro robota podle 2. metody lze rozepsat na výpočet polohy a orientace. Poloha je vyjádřena pomocí trojrozměrného vektoru a orientace je vyjádřena pomocí kvaternionu. Na začátku výpočtu nové polohy pro robota se uloží startovní poloha $P_{C_1}$ a orientace $q_{C_1}$ ovladače a startovní poloha $P_{R_1}$ a orientace $q_{R_1}$ robota.

Po dobu držení tlačítka na ovladači se každou n-tou periodu načítá aktuální poloha $P_{C_n}$ a orientace $q_{C_n}$ ovladače.

Vypočte se posun $\Delta P_{C_n}$ a rotace $\Delta q_{C_n}$ mezi startovní polohou a orientací ovladače a aktuální polohou a orientací ovladače.

$$\Delta P_{C_n} = P_{C_n} - P_{C_1} \tag{5.5}$$

$$\Delta q_{C_n} = q_{C_1}^{-1} \cdot q_{C_n} \tag{5.6}$$

Posun $\Delta P_{C_n}$ a rotace $\Delta q_{C_n}$ určují posunutí a rotaci startovní polohy a orientace robota. Tedy už lze získat nová poloha $P_{R_{n+1}}$ a orientace $q_{R_{n+1}}$ pro robota pomocí startovních údajů robota a údajů o posunutí a rotaci ovladače.

$$P_{R_{n+1}} = P_{R_1} + \Delta P_{C_n} \tag{5.7}$$

$$q_{R_{n+1}} = q_{R_1} \cdot \Delta q_{C_n} \tag{5.8}$$

# Kapitola 6

# Návrh aplikace

Aplikace by měla hýbat robotem podle pohybů ovladače HTC Vive. V případě této práce se pohybuje robotickým manipulátorem Kuka LBR iiwa. Tento robot je svými rozměry podobný lidské ruce. V tomto případě si lze představit možné využití v nebezpečných situacích pro člověka. Například při práci s nebezpečnými materiály, které mohou být výbušného charakteru,mohou být radioaktivní nebo mohou jakkoliv jinak ohrožovat člověka na zdraví. Jedním příkladem je tzv. "hot cell"[11] což je místnost s radioaktivním materiálem, ve které jsou chapadla robota uvnitř místnosti ovládaná vně místnosti.

Při použití robota výrazně většího než lidská ruka by se našlo využití například v průmyslu, kde je zapotřebí mnohem vetší síla než, které dosáhne člověk. Například ve stavebním nebo zemědělském průmyslu se využívají těžké stroje, které by mohly být ovládány systémem podobným této aplikaci.

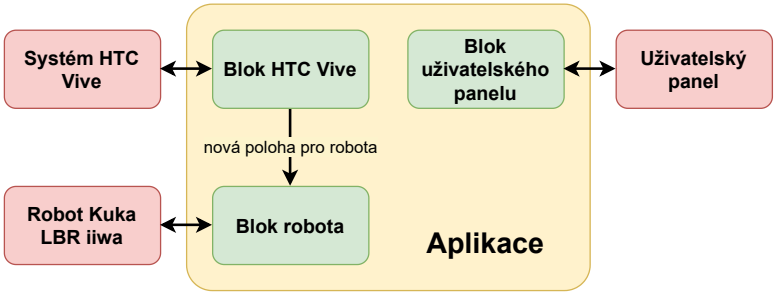
Další možností by bylo použití malého robota. Například pro práci s malými objekty. Jak již zaznělo v kapitole 2 ukázkovým příkladem je robot Da Vinci, který provádí neinvazivní minioperace v lidském těle.

## 6.1 Struktura aplikace

Aplikace pracuje se dvěma hlavními periferiemi, těmi je systém HTC Vive a robot Kuka LBR iiwa. Vedlejší periferií je uživatelský panel, který zobrazuje informace o aktuálním nastavení aplikace. Aplikaci jsem rozdělil do 3 bloků, které spolupracují a každý blok pracuje se svojí periferií. Bloky jsem pojmenoval:

- blok systému HTC Vive
- blok robota
- blok uživatelského panelu

Bloky a jejich periferie jsou zobrazeny na blokovém diagramu 6.1. Zelené bloky jsou bloky aplikace a červené bloky jsou periferie, se kterými aplikace pracuje.




**Obrázek 6.1:** Blokové schéma aplikace s periferiemi

### 6.1.1 Blok systému HTC Vive

Tento blok komunikuje se systémem HTC Vive. Zjišťuje polohy a orientace zařízení HTC Vive. Zpracovává interakce uživatele s ovládacími prvky na ovladači HTC Vive. Počítá novou polohu a orientaci robota a poskytuje ji bloku robota

### 6.1.2 Blok robota

Blok robota má za úkol pohybovat robotem pomocí dat o nové poloze, které mu poskytuje blok HTC Vive.

### 6.1.3 Blok uživatelského panelu

Uživatelský panel zobrazuje aktuální nastavení aplikace a stručně vysvětluje jak se aplikace ovládá.

## 6.2 Přidané funkce aplikace

Následující přidané funkce zjednodušují ovládání aplikace nebo nějak upravují výpočet nové polohy pro robota.

### 6.2.1 Škálování pohybů

Tato funkcionalita přidává možnost nastavení poměrové konstanty. Poměrová konstanta může nabývat hodnoty: 0.12, 0.25, 0.5, 1.0, 2.0. Když je poměrová konstanta rovna 1.0, tak se při posunu ovladače o 1 m, posune robot o 1 m. Při poměrové konstantě nastavené na hodnotu 0.5, se při posunu ovladače o 1 m, robot posune o 0.5 m.

Tohoto efektu lze docílit upravením mezivýpočtu pro posun $\Delta P_{C_n}$ 5.5 a rotaci $\Delta q_{C_n}$ 5.6 poměrovou konstantou. Pro upravení posunu stačí vynásobit jednotlivé složky vektoru $\Delta P_{C_n}$ poměrovou konstantou. Pro upravení rotace se hodí převést kvaternion na vyjádření osa a úhel, kde osa je normovaný vektor popisující osu rotace v prostoru a úhel udává úhel rotace. V tomto vyjádření stačí vynásobit úhel poměrovou konstantou a převést rotaci zpět na kvaternion.

### 6.2.2 Módy aplikace

Aplikace může pracovat ve 3 různých módech:

- Normal
- Rotation
- Translation

Mezi módy lze přepínat pomocí ovladače HTC Vive nebo pomocí klávesnice počítače.

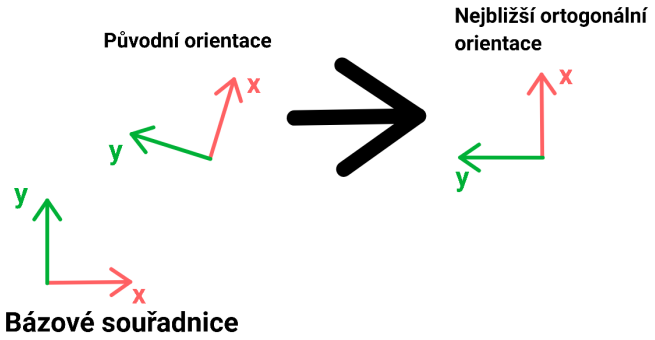
V módu Normal se posílá nová poloha a orientace pro robota tak, jak je vypočítána v sekci 5.4 a Upravená pomocí poměrové konstanty tak, jak je popsáno v 6.2.1.

V módu Rotation se se posílá jen nová orientace pro robota. Nová poloha pro robota $P_{R_{n+1}}$ 5.7 se nahradí startovní polohou robota $P_{R_1}$. Ve výsledku se poloha koncového bodu nemění, mění se jen orientace podle pohybů ovladače.

V módu Translation se posílá jen nová poloha pro robota. Nová orientace pro robota $q_{R_{n+1}}$ 5.8 se nahradí startovní orientací robota $q_{R_1}$.

### 6.2.3 Ortogonální připnutí orientace

Stisknutím systémového tlačítka lze změnit aktuální orientaci na nejbližší ortogonální orientaci. Ortogonální je myšleno vzhledem k souřadnicovým osám nastavených podle orientace headsetu HTC Vive. Buď je orientace ortogonální k souřadnicovým osám nebo je s nimi rovnoběžná. Jak ortogonální připnutí funguje je znázorněno v rovině na obrázku 6.2.




**Obrázek 6.2:** Znázornění ortogonálního připnutí v rovině

#### Nalezení nejbližší ortogonální orientace

V prostoru existuje 24 různých orientací, ke kterým se lze ortogonálně připnout.

Pro nalezení nejbližší ortogonální orientace k aktuální orientaci lze jen projít všech 24 zmíněných orientací a vybrat tu, která je nejblíž té aktuální.

Aby se orientace připla musí být nejbližší ortogonální orientace rotována maximálně 30° od aktuální, v opačném případě ovladač zavibruje.

### 6.2.4 Zpětná vazba uživateli

Uživatel dostává zpětnou vazbu prostřednictvím ovladače HTC Vive ve formě vibrací ovladače. Jsou dva případy, kdy může ovladač vibrovat:

- Překročení maximální rychlosti ovladače

  Při překročení maximální rychlosti ovladače ovladač zavibruje na jednu sekundu s frekvencí 200 Hz. Toto nastane jen, když se posílá nová poloha pro robota do vlákna robota. Překročení maximální rychlosti zapříčiní ukončení posílání nové polohy pro robota.

- Nová poloha robota je mimo možnosti robota

  Nová poloha pro robota je například už mimo pracovní prostor robota, jinak řečeno už tam robot nedosáhne. Další možností je, že nová poloha je v kolizi.

  Toto je značeno zavibrováním ovladače dokud uživatel drží tlačítko na ovladači pro pohyb a nová poloha pro robota je stále mimo možnosti reálného robota. Ovladač vibruje s frekvencí 1000 Hz.

- Ortogonální připnutí orientace se nezdařilo

  Pokud by při ortogonálním připnutí bylo všech 24 ortogonálních orientací vzdálených více než 30° od aktuální orientace robota, tak ovladač zavibruje na 1 sekundu s frekvencí 100 Hz.

# Kapitola 7

# Implementace aplikace

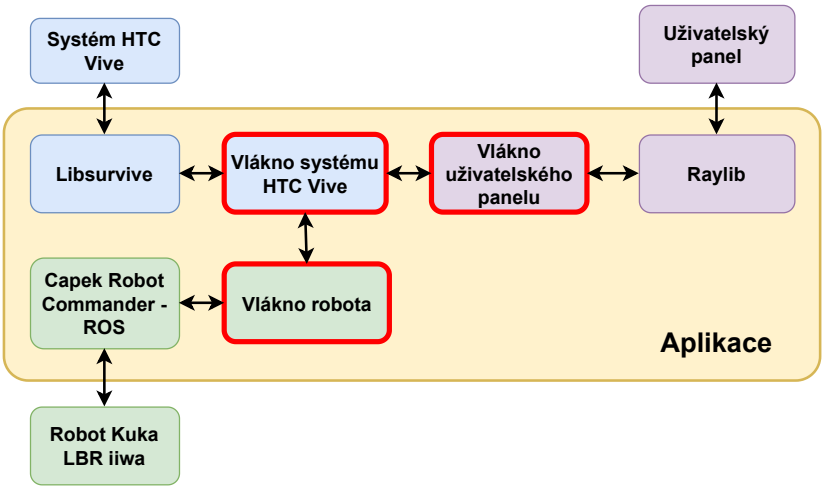
Jak je zmíněno v sekci 6.1, aplikace je rozdělena do 3 logických bloků. Bloky byly implementovány pomocí vícevláknového programování. Jsou tedy 3 vlákna:

- Vlákno systému HTC Vive
- Vlákno robota
- Vlákno uživatelského panelu (grafika)

Každé vlákno má přiřazené úkoly související s ovládáním robota, práce se systémem HTC Vive nebo správou uživatelského panelu. Komunikace mezi vlákny je provozována pomocí sdílené paměti, ke které mají přístup všechna vlákna.

Ve blokovém diagramu vláken 7.1 je zobrazeno, s jakými periferiemi vlákna pracují. Oproti diagramu 6.1 je zobrazeno také, které softwarové řešení každé vlákno používá pro práci s těmito periferiemi.

Červeně ohraničené části aplikace označují ty části, které byly vytvořené v rámci této práce.




**Obrázek 7.1:** Schéma vláken a jejich periferií

## 7.1 Vlákno robota

Toto vlákno zajišťuje připojení k robotu pomocí knihovny CRC [Capek Robot Commander]. Knihovna CRC používá ROS pro ovládání robota. Vlákno dále načítá aktuální data o poloze robota. Tyto data jsou k dispozici vláknu systému HTC Vive. Periodicky se aktualizuje nová poloha, ve které má robot být. Tato nová poloha je poskytována vláknem systému HTC Vive.

Pomocí knihovny CRC se počítá inverzní kinematika robota v dané poloze a následně se robot do dané polohy pohne. Pohyb robota ale blokuje běh vlákna. Při vykonání pohybu robota se tedy může nová poloha pro robota ve vláknu pro systém HTC Vive aktualizovat i několikrát. Po úspěšném pohybu robota se opět načte aktuální poloha, ve které má robot být. Následuje pohnutí robotem do nově načtené aktuální polohy. Tedy se opakovaně načítá nová poloha a poté se hýbe robotem do načtené polohy.

Jelikož pohnutí robotem zastaví na dobu pohybu funkčnost vlákna, určuje délka pohybu to, jak rychle se aktualizuje poloha. Platí tedy, že čím pomaleji se mění poloha, ve které má robot být, tím přesněji kopíruje robot trajektorii, která byla původně poslána vláknem ovladače HTC Vive. Během pohybu robota se přeskočí všechny polohy poskytnuty vláknem systému HTC Vive během tohoto pohybu. A nový pohyb robota se uskuteční do nejaktuálnější známé polohy. To způsobuje to, že při běhu aplikace se pohyby robota jeví přerušovaně. Tento nedostatek by bylo možné odstranit použitím složitějšího ovládání robota.

## 7.2 Vlákno uživatelského panelu

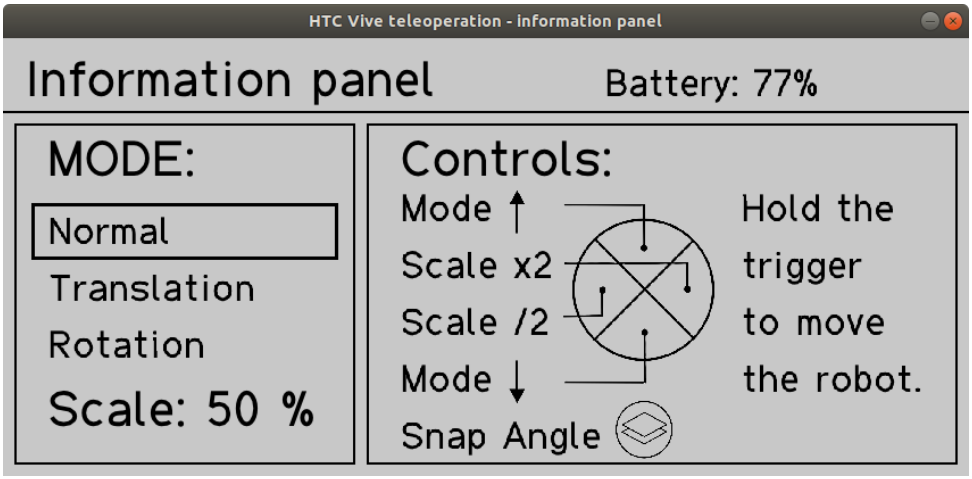
Uživatelský panel zobrazuje aktuální nastavení aplikace: mód, poměrová konstanta a nabití ovladače. Uživatelské rozhraní lze použít k nastavení aplikace, tedy ke změně módu a poměrové hodnoty pomocí šipek na klávesnici počítače. Toto vlákno používá grafickou knihovnu Raylib pro vytvoření grafického okna. Uživatelský panel je na obrázku 7.2.

Na levé straně se zobrazují informace o aktuálním nastavení aplikace, tedy mód, anglicky "mode", a poměrová konstanta, anglicky "scale".

Na pravé straně je vyobrazeno, jaké ovládací prvky na ovladači HTC Vive se používají k ovládání nastavení aplikace.

## 7.3 Vlákno systému HTC Vive

Toto vlákno používá knihovnu Libsurvive pro komunikaci se zařízeními systému HTC Vive. Zpracovává data o poloze jednoho ovladače, který uživatel drží v ruce a data o poloze headsetu. Zpracovává interakci uživatele s ovládacími prvky ovladače HTC Vive. Dále podle módu aplikace počítá novou polohu a orientaci pro robota a poskytuje tato data vláknu robota.




**Obrázek 7.2:** Grafický uživatelský panel

Poloha headsetu se používá jen pro nastavení orientace souřadného systému HTC Vive. HTC Vive používá jinak orientovanou soustavu souřadnic než robot, a proto se musí najít transformace mezi těmito soustavami. Aktuálně je nastaveno, že když je headset natočen tak, aby se díval směrem proti ose x v soustavě souřadnic robota, pak jsou souřadné soustavy orientovány stejně a směr pohybu ovladače je stejný jako směr pohybu robota.

Když je zjištěna transformace mezi souřadnými systémy robota a HTC Vive, může se načítat poloha a orientace ovladače. Při každém načtení dat polohy a orientace ovladače se hned data transformují do souřadné soustavy robota. To ulehčí následné zpracování dat, jelikož se s daty již může pracovat jen v souřadné soustavě robota.

### 7.3.1 Běh vlákna systému HTC Vive

Nejdříve vlákno HTC Vive spustí vlákno knihovny Libsurvive, která si sama inicializuje připojená zařízení systému HTC Vive a začne s nimi komunikovat.

Po úspěšném spuštění knihovny Libsurvive se aplikace pokusí zjistit, zda je připojen headset a ovladač. Poté, co najde obě zařízení, načte se poloha a orientace headsetu a uloží se jméno ovladače, který byl nalezen. Pod tímto jménem bude vlákno později žádat polohu ovladače. I když se později připojí druhý ovladač, aplikace bude stále pracovat pouze s tímto prvním nalezeným ovladačem. Pro výměnu ovladače se musí aplikace restartovat.

Následuje potvrzení o úspěšné inicializaci vlákna HTC Vive změnou příslušných proměnných ve sdílené paměti a čeká se na potvrzení o úspěšné inicializaci vlákna robota. Poté, co se potvrdí inicializace vlákna robota přejde se do hlavní smyčky vlákna. Smyčka se ukončí, když se zavře okno grafického rozhraní křížkem. V této smyčce se opakují tyto akce:

- aktualizace stavů ovládacích prvků,
- správa stavů bezpečnostního tlačítka (triggeru).

Tyto situace je nutno řešit co nejrychleji, proto nemají žádné časové omezení a kontrolují se co nejčastěji to jde. Jelikož jde o periferii, se kterou uživatel zachází přímo, měla by tedy mít co nejmenší odezvu.

Dále se s konstantní periodou opakují tyto akce:

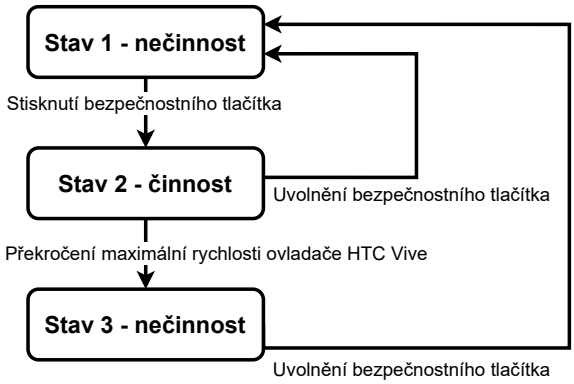
- zjištění aktuální polohy ovladače,
- výpočet nové polohy pro robota,
- zjištění aktuálního nabití ovladače.

### Aktualizace stavů ovládacích prvků

Pro detekci interakce uživatele s ovládacími prvky se načtou všechny události systému HTC Vive. Z těchto událostí se vyfiltrují jen události s ovládacími prvky. V aplikaci je pole, do kterého se ukládají všechny ovládací prvky a jejich stavy, ve smyslu stisklé/nestisklé případně údaj o daném tlačítku jako například poloha doteku na touchpadu. Pokaždé, co je zaznamenána akce s ovládacími prvky, toto pole se aktualizuje. Pro zjištění stavů ovládacích prvků se dále používá jen toto pole.

### Správa stavů bezpečnostního tlačítka (triggeru)

Trigger na ovladači se používá jako takzvané bezpečnostní tlačítko. Když uživatel úplně stiskne bezpečnostní tlačítko, aplikace přenáší novou polohu pro robota do vlákna robota na základě polohy ovladače HTC Vive a zvoleného módu aplikace. V aplikaci jsou 3 stavy, kontrolují zda se robot může pohybovat. Tyto stavy se mění podle stisknutí/uvolnění bezpečnostního tlačítka a překročením maximální rychlosti ovladače. Přepínání stavů je popsáno na diagramu 7.3




**Obrázek 7.3:** Diagram přepínání stavů bezpečnostního tlačítka

Když je aktivní první stav, tak vlákno HTC Vive neposílá žádnou novou pozici do vlákna robota. Z prvního stavu lze přejít do druhého stavu úplným stisknutím bezpečnostního tlačítka. Ve druhém stavu se posílají nová data o poloze a orientaci do vlákna robota. Při přechodu do tohoto stavu (v momentě

stisknutí tlačítka) se uloží aktuální data o poloze a orientaci robota a aktuální data o poloze a orientaci ovladače. Tato data se používají k výpočtu nové polohy a orientace pro robota. Ze druhého stavu jde přejít do prvního stavu uvolněním bezpečnostního tlačítka.

Při příliš vysoké rychlosti ovladače HTC Vive se přejde ze druhého stavu do třetího. Při přechodu ze druhého stavu do třetího ovladač zavibruje po dobu 1 sekundy, aby upozornil uživatele na změnu stavu. Ve třetím stavu se neodesílá žádná nová poloha a orientace do vlákna robota. Ve třetím stavu se čeká na uvolnění bezpečnostního tlačítka aby se mohlo přejít do prvního stavu.

### Zjištění aktuální polohy ovladače

Načítání polohy ovladače probíhá s periodou 10 ms. Nově načtená data o poloze a orientaci ovladače se ukládají do bufferu. Pro výpočet nové polohy a orientace pro robota se použije průměrná poloha a průměrná orientace z dat uložených v bufferu. To do jisté míry vyhladí trajektorii ovladače. V bufferu je uloženo 15 nejnovějších dat o poloze a orientaci. Experimentem bylo zjištěno, že velikost tohoto bufferu by měla být v řádech desítek uložených pozic. Když je velikost moc malá, následné průměrování dat z bufferu nemá příliš smysl a data se dostatečně nevyhladí, při velikosti bufferu v řádech 100 je už problém v tom, že vzniká příliš velké zpoždění. Již při 100 datech v bufferu je zpoždění 0.5 sekundy jen v načítání polohy a orientace ovladače. Velikost 15 je experimentálně ověřený kompromis. Při každém načtení nové polohy se vypočítá rozdíl s poslední hodnotou uloženou v bufferu. Z tohoto rozdílu se počítá rychlost ovladače. Může se stát, že aplikace chybně zaznamená rychlý pohyb ovladače. To může být způsobeno chybou v knihovně Libsurvive. Tato chyba se projevuje tím, že poloha zařízení HTC Vive je detekována špatně. Podle detekovaných údajů to vypadá tak, že poloha ovladače náhodně skáče někdy až o vzdálenosti kolem jednoho metru, i když reálný ovladač se ve skutečnosti ani nepohne.

Jedním možným řešením je vynucení kalibrace systému HTC Vive v knihovně Libsurvive. To lze provést vymazáním souboru "config.json"uloženého v adresáři "home/<user_name>/.config/libsurvive". Může se stát i to, že kalibrace problém nevyřeší a v tom případě se musí kalibrace opakovat, dokud chyba nezmizí. Aplikace při každém spuštění vymaže kalibrační soubor, aby vynutila novou kalibraci. Díky tomu se v běhu aplikace chyba objeví jen zřídka.

Tato chyba se může projevit i v případě, kdy je ovladač HTC Vive mimo svůj pracovní prostor. Ten je určen polohou a orientací majáků. V tomto případě lze problém vyřešit tím, že uživatel přemístí ovladač do jeho pracovního prostoru.

### Výpočet nové polohy a orientace robota

Výpočet probíhá tak jak je popsáno v sekci 5.4. Podle nastavení aplikace se před posláním do vlákna robota upraví tak jak je popsáno v sekci 6.2.

# Kapitola 8

# Experimenty

## 8.1 Dotazník - pochopení a intuitivita aplikace

### 8.1.1 Experimentální postup

Pro zjištění pochopitelnosti aplikace pro ovládání robota pomocí ovladače HTC Vive v praxi byl proveden experiment, který se zaměřoval na hodnocení srozumitelnosti a intuitivnosti ovládání robota pomocí Kuka Smartpadu a aplikace s využitím HTC Vive. Následující části popisují podrobnosti o provedení experimentu.

### 8.1.2 Výběr respondentů

Pro experiment byla vybrána skupina respondentů, která zahrnovala studenty v oboru strojírenství a robotiky.

### 8.1.3 Provedení experimentu

Respondenti měli příležitost vyzkoušet ovládání robota pomocí Kuka Smartpadu a aplikace s využitím HTC Vive. Pro obě ovládací metody byl připraven úkol. Respondenti měli dostatek času seznámit se s ovládacími rozhraními a úkol provést.

### 8.1.4 Dotazník

Pro sběr dat byl vytvořen dotazník obsahující otázky zaměřené na srozumitelnost, intuitivnost a obecné dojmy respondentů o ovládání robota pomocí Kuka Smartpadu a aplikace s využitím HTC Vive. Dotazník obsahoval škálové otázky, kde respondenti mohli vyjádřit svůj názor na ovládání na škále od 1 do 10, a otevřené otázky, které umožňovaly rozšířené komentáře a postřehy.

### 8.1.5 Výsledky dotazníku

Následují otázky a průměry odpovědí na otázky, na které bylo možné odpovědět na škále od 1 do 10. Kde 1 je špatné hodnocení a 10 je dobré hodnocení.

1. Jak byste zhodnotil(a) srozumitelnost ovládání pomocí Kuka Smartpadu na škále od 1 do 10?

   Průměrná odpověď: 7,2

2. Jak byste zhodnotil(a) srozumitelnost ovládání aplikace využívající HTC Vive na škále od 1 do 10?

   Průměrná odpověď: 9

3. Jak rychle si myslíte, že si budete schopen(a) opětovně vzpomenout na ovládání robota pomocí Kuka Smartpadu po uplynutí určité doby (např. týden), na škále od 1 do 10?

   Průměrná odpověď: 8,6

4. Jak rychle si myslíte, že si budete schopen(a) opětovně vzpomenout na ovládání aplikace s HTC Vive po uplynutí určité doby (např. týden), na škále od 1 do 10?

   Průměrná odpověď: 9,2

5. Jak byste zhodnotil(a) úroveň intuitivnosti ovládání robota pomocí Kuka Smartpadu na škále od 1 do 10?

   Průměrná odpověď: 7,8

6. Jak byste zhodnotil(a) úroveň intuitivnosti ovládání robota pomocí aplikace s HTC Vive na škále od 1 do 10?

   Průměrná odpověď: 8,6

Na základě průměrných hodnot lze zjistit, že respondenti hodnotili aplikaci s HTC Vive (9) více srozumitelnou než ovládání pomocí Kuka Smartpadu (7,2). Respondenti ohodnotili svou schopnost si opětovně vzpomenout na ovládání obou způsobů ovládání velmi podobně. Intuitivnost se respondentům více líbila u ovládání pomocí aplikace s HTC Vive (8,6) než u ovládání pomocí Kuka Smartpadu (7,8).

Dále byly položeny otevřené otázky:

1. Jaké vlastnosti ovládání robota pomocí aplikace s využitím HTC Vive se vám líbí nejvíce?

2. Co se vám na ovládání robota pomocí aplikace s využitím HTC Vive nelíbí nebo co byste na ní změnil(a)?

3. Jak si myslíte, že by mohlo být využití této formy ovládání robota na dálku s využitím systému jako je HTC Vive přínosné v praxi?

4. Máte nějaké další poznámky, komentáře nebo nápady ohledně ovládání robota pomocí Kuka Smartpadu nebo aplikace s využitím HTC Vive?

Na základě odpovědí na tyto otázky lze říct, že mezi oblíbenými vlastnostmi aplikace s HTC Vive je možnost ovládat jak posun, tak rotaci robota najednou

pomocí pohybu ruky. Respondenti však také zmínili některé negativní body, jako je zpoždění reakce robota, toto bylo zmíněno téměř všemi respondenty. Respondenti by dále ocenili přidání hystereze pro rotaci.

Co se týče využití této formy ovládání robota na dálku s využitím systému jako je HTC Vive, respondenti vidí potenciál v jednoduchosti ovládání, které je vhodné pro pracovníky bez technického vzdělání. Jako možné využití byla vícekrát vyjmenována práce ve stavebním průmyslu, konkrétně práce s těžkými materiály.

## 8.2 Sledování polohy zařízení HTC Vive

Systém HTC Vive sleduje polohu a orientaci zařízení pomocí 2 majáků, tak jak je zmíněno v sekci 3.2. Tyto majáky se musí umístit do prostoru, nejlépe každý do jednoho rohu pokoje a oba namířit doprostřed místnosti. To bohužel ve velké laboratoři, kde se systém testoval není úplně možné. Bylo vyzkoušeno, že když jsou oba majáky namířeny do stejného prostoru přesnost sledování polohy a orientace zařízení v tomto prostoru je velmi dobrá. Když se ale zařízení z tohoto prostoru vzdálí, přesnost se zhoršuje.

# Kapitola 9

# Závěr

Pro práci se systémem HTC Vive byla použita knihovna Libsurvive. Pro řízení robota byla použita knihovna Capek Robot Commander. Spojit tyto systémy se podařilo pomocí vícevláknového programování. Každý systém je provozován ve svém vlákně. Výsledné ovládání umí úspěšně pohybovat koncovým bodem robota pomocí ovladače HTC Vive. Ovládání má několik nastavení, jako módy, poměrovou hodnotu a připnutí na nejbližší ortogonální úhel. Aby byla aktuální nastavení k dispozici pro uživatele, rozhodl jsem se vytvořit pro aplikaci grafický panel. Grafický panel využívá knihovnu Raylib a má svoje vlastní vlákno.

Dále se pro aplikaci vytvořili dodatečné funkce, které umožňovali změnit podstatu ovládání nebo ovládání nějak upravili.

Aplikace byla vyzkoušena na skupině lidí, kteří následně vyplnili dotazník zaměřený na pochopení ovládání a intuitivnost ovládání aplikace.

Jak vyplývá z výsledků dotazníku, je zapotřebí snížit dlouhou odezvu ovládání. Výsledný pohyb robota je navíc sekaný, a to by se mohlo též vylepšit. Oba nedostatky by šlo do jisté míry vyřešit lepší regulací ovládání robota a použití složitějších softwarových nástrojů pro řízení robota. Například kontrolovat přímo rychlosti v jednotlivých kloubech robota.

# Literatura


[1] ker. Da Vinci Sistema Kirurgikoa [online]. Wikimedia Commons, 2023 [cit. 2023-05-14]. Dostupné z: https://commons.wikimedia.org/wiki/File:Da_Vinci_Sistema_Kirurgikoa.jpg.

[2] PvOberstein. IKOR T12 Caliber Bomb Disposal Robot [online]. Wikimedia Commons, 2023 [cit. 2023-05-14]. Dostupné z: https://commons.wikimedia.org/wiki/File:IKOR_T12_Caliber_Bomb_Disposal_Robot.jpg.

[3] Centrum robotické chirurgie. Nemocnice Na Homolce [online]. [cit. 2023-05-14]. Dostupné z: https://www.homolka.cz/nase-oddeleni/11635-specializovana-centra/11635-centrum-roboticke-chirurgie/

[4] Defusing a Bomb With a Joystick Isn't Easy, So Pittsburgh Company Creates Scale Robotic Arm. In: WESA [online]. 6. června 2017 [cit. 2023-05-14]. Dostupné z: https://www.wesa.fm/science-health-tech/2017-06-06/defusing-a-bomb-with-a-joystick-isnt-easy-so-pittsburgh-company-creates-scale-robotic-arm.

[5] Libsurvive [online]. [cit. 2023-05-14]. Dostupné z: https://github.com/cntools/libsurvive.

[6] OpenVR SDK [online]. [cit. 2023-05-14]. Dostupné z: https://partner.steamgames.com/doc/features/steamvr/openvr?l=czech.

[7] KUKA. LBR iiwa [online]. [cit. 2023-05-14]. Dostupné z: https://www.kuka.com/cs-cz/produkty,-slu%C5%BEby/robotick%C3%A9-syst%C3%A9my/pr%C5%AFmyslov%C3%A9-roboty/lbr%C2%A0iiwa.

[8] Raylib [online]. [cit. 2023-05-14]. Dostupné z: https://www.raylib.com/.

[9] Gammieri, L., Schumann, M., Pelliccia, L., Di Gironimo, G., & Klimant, P. (2017). Coupling of a Redundant Manipulator with a Virtual Reality Environment to Enhance Human-robot Cooperation. Procedia CIRP, 63, 181-186. DOI:10.1016/j.procir.2016.06.056

[10] Whitney, D., Rosen, E., Phillips, E., Konidaris, G., & Tellex, S. (2017). Comparing Robot Grasping Teleoperation across Desktop and Virtual Reality with ROS Reality. In Proceedings of the International Symposium on Robotics Research.

[11] Djedidi, A., Selliez-Vandernotte, C., & Malcolm, F. (n.d.). Operating gains achieved by a new generation of remotely-controlled manipulators. NEI Magazine. Dostupné z: https://www.neimagazine.com/features/featurehot-cell-robot-4483658/.

[12] HTC Vive. About the Controllers [online]. [cit. 2023-05-14]. Dostupné z: https://www.vive.com/us/support/vive/category_howto/about-the-controllers.html.

# Příloha A

## Seznam příloh

1. main.cpp: implementace vláken aplikace
2. pose_calc.h: hlavičkový soubor pro matematické operace s polohami
3. pose_calc.cpp: implementace matematických operací s polohami
4. vive.h: hlavičkový soubor pro interakci s ovladačem HTC Vive
5. vive.cpp: implementace interakce s ovladačem HTC Vive
6. headers.h: hlavičkový soubor importující používané knihovny
7. constants.h: hlavičkový soubor s konstantami aplikace
8. dotaznik.xlsx: výsledky dotazníku