ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE

FAKULTA STROJNÍ

Ústav výrobních strojů a zařízení

Diplomová práce

Návrh řízení pro malý průmyslový robot

2021 **Bc. Mykyta Klymoshenko**

ZADÁNÍ DIPLOMOVÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Klymoshenko** Jméno: **Mykyta** Osobní číslo: **453568**

Fakulta/ústav: **Fakulta strojní**

Zadávající katedra/ústav: **Ústav výrobních strojů a zařízení**

Studijní program: **Průmysl 4.0**

Studijní obor: **bez oboru**

II. ÚDAJE K DIPLOMOVÉ PRÁCI

Název diplomové práce:

Návrh řízení pro malý průmyslový robot

Název diplomové práce anglicky:

Control design for a small industrial robot

Pokyny pro vypracování:

Popis tématu: Cílem práce je vytvoření SW pro řízení sériového 6-osého robotu, který by měl umožnit pohyb robotu dle zadání v Kartézském souřadnicovém systému. Řešení bude navrženo pro malý laboratorní robot vycházející z open-source dokumentace; Osnova práce: 1) Rešerše možných řešení, 2) Návrh struktury řízení (včetně výpočtu inverzní kinematiky a ošetření singulárních stavů), 3) Implementace do real-time, 4) Simulační testování; Rozsah grafické části: schéma řízení robotu, grafy vybraných průběhů polohy koncového bodu robotu a natočení jednotlivých kloubů; Rozsah textové části: 60-80 stran;

Seznam doporučené literatury:

STEJSKAL, V. a M. VALÁŠEK. Kinematics and Dynamics of Machinery. New York: Marcel Dekker, 1996. ISBN 0824797310;
RAGHAVAN, M. a B. ROTH. Inverse Kinematics of the General 6R Manipulator and Related Linkages. The American Society of Mechanical Engineers, 1993, č. 115, s. 502-08. ISSN 1050-0472;

Jméno a pracoviště vedoucí(ho) diplomové práce:

Ing. Jiří Švéda, Ph.D., ústav výrobních strojů a zařízení FS

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

Ing. Štěpán Chládek, Ph.D., ústav výrobních strojů a zařízení FS

Datum zadání diplomové práce: **29.04.2021** Termín odevzdání diplomové práce: **25.07.2021**

Platnost zadání diplomové práce: **30.09.2021**

Ing. Jiří Švéda, Ph.D. podpis vedoucí(ho) práce	Ing. Matěj Sulitka, Ph.D. podpis vedoucí(ho) ústavu/katedry	prof. Ing. Michael Valášek, DrSc. podpis děkana(ky)

III. PŘEVZETÍ ZADÁNÍ

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

.	
Datum převzetí zadání	Podpis studenta

© ČVUT v Praze, Design: ČVUT v Praze, VIC

Prohlášení

Prohlašuji, že jsem svou bakalářskou práci vypracoval samostatně a že jsem uvedl v přiloženém seznamu veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací, vydaným ČVUT v Praze 1. 7. 2009.

Nemám závažný důvod proti užití tohoto školního díla ve smyslu § 60 Zákona č.121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon).

V Praze dne 18.7.2021 ……

podpis

Poděkování

Rad bych poděkoval vedoucímu své diplomové práce Ing. Jiřímu Švédovi, Ph.D. za vedení a cenné připomínky při vypracování celé práce. Dále bych rad poděkoval konzultantovi Ing. Štěpánovi Chládkovi, Ph.D. za jeho čas, vstřícnost při konzultacích a pomoc během vypracování praktické části práce.

Také bych rad poděkoval rodině za podporu během celé doby studia.

Anotace

Autor:	Bc. Mykyta Klymoshenko
Název DP:	Návrh řízení pro malý průmyslový robot
Rozsah práce:	69 str., 61 obr., 2 tab.
Školní rok vyhotovení:	2020/21
Škola:	ČVUT v Praze, Fakulta strojní
Ústav:	Ú12135 – Ústav výrobních strojů a zařízení
Vedoucí diplomové práce:	Ing. Jiří Švéda, Ph.D.
Konzultant:	Ing. Štěpán Chládek, Ph.D.
Zadavatel:	ČVUT FS
Využití:	Řízení malého průmyslového robotu s využitím PLC od společnosti Beckhoff.
Klíčová slova:	Řešení kinematiky robotu, dopředná kinematická úloha, inverzní kinematická úloha, analytické řešení, numerické řešení, C++, real-time.
Anotace:	Diplomová práce se zabývá vytvořením SW nástroje pro řízení sériového šestiosého robotu se zahrnutím dopředné a inverzní kinematické úlohy. Řízení robotu je realizováno v prostředí TwinCAT s využitím programovacího jazyku C++ pro výpočet v reálném čase. Řešení je vytvořeno pro malý laboratorní robot vycházející z open source dokumentace.

Annotation

Author:	Bc. Mykyta Klymoshenko
Title of master dissertation:	Control design for a small industrial robot
Extent:	69 p., 61 fig., 2 tab.
Academic year:	2020/21
University:	CTU in Prague – Faculty of Mechanical Engineering
Department:	Ú12135 – Department of Production Machines
Supervisor:	Ing. Jiří Švéda, Ph.D.
Consultant:	Ing. Štěpán Chládek, Ph.D.
Submitter of the Theme:	ČVUT FS
Application:	Control of a small industrial robot using a PLC from Beckhoff
Key words:	Robot kinematics solution, forward kinematic problem, inverse kinematic problem, analytical solution, numerical solution, C ++, real-time
Annotation:	The diploma thesis deals with the creation of a SW tool for calculating the forward and inverse kinematics of a serial six-axis robot. Robot control is written in the TwinCAT environment using the C++ programming language for real-time calculation. The solution is created for a small laboratory robot based on open source documentation.

Obsah

1 Úvod ... 8
2 Stav problematiky ... 9
2.1 Průmyslové roboty ... 9
2.1.1 Souřadnicové systémy robotu ... 18
2.2 Řídicí systém ... 19
2.2.1 Programování robotů ... 24
3 Cíle diplomové práce ... 26
4 Návrh struktury řízení ... 27
4.1 Kinematika robotu ... 27
4.1.1 Dopředná kinematická úloha ... 27
4.1.2 Inverzní kinematická úloha ... 29
4.2 Singulární polohy robotu ... 40
4.3 Metoda výpočtu kinematiky ... 41
4.4 Schéma řízení ... 48
5 Implementace do real-time ... 50
5.1 NC úloha ... 50
5.2 Modul pro uživatelskou transformaci ... 52
5.3 PLC projekt ... 54
5.4 Výpočet inverzní kinematiky v C++ ... 57
6 Simulační testování ... 60
7 Závěr ... 62
Seznam použité literatury ... 63
Seznam obrázků ... 66
Seznam tabulek ... 68
Seznam příloh ... 69

1 Úvod

V dnešní době, která spadá pod tzv. 4. průmyslovou revoluci, jsou stále aktuální termíny automatizace a robotizace. Každý rok jsou na výrobní průmyslové podniky kladeny vyšší nároky na kvalitu, rychlost výroby, nižší pořizovací náklady a další faktory. Aby tyto nároky splňovaly, využívají v některých případech už i plně automatizovaná pracoviště. Součástí takových pracovišť jsou téměř vždy roboty.

Mechanické části robotu jsou v dnešní době téměř na hraně optimalizace a poskytují pouze malý potenciál pro zdokonalení. Toto však neplatí pro neviditelnou část každého robotu – software. Právě software pro ovládaní robotu je místem, kde se neustále dají vyvíjet nové funkce a zdokonalovat ty staré, čímž se dají splňovat stále rostoucí požadavky.

Jednou z nejdůležitějších funkcí, které provádí SW pro řízení, je kinematická transformace. Každý výrobce robotů nebo dodavatel SW pro řízení robotu řeší tyto funkce jinak. Jsou složité a většinou nejsou univerzální, což znamená, že nejde použít jiný typ robotu, ve smyslu kinematické konfigurace, se stejnou kinematickou transformací. Překonfigurováni je pak náročné a může trvat dlouho.

Kromě velkých průmyslových robotů s rozvojem alternativních způsobů výroby, jako je například 3D tisk, jsou čím dál tím víc používány roboty malé. Tyto roboty jsou nejvíce vhodné pro různé druhy vzdělávání. Vzdělávání a školení pracovníků údržby v továrnách je podmínkou pro splnění velkých nároků na výrobu. Mimo jiné se takové roboty používají i ve školách, kde si každý může zkusit naprogramovat robot a svůj program pak na místě vyzkoušet. Jednou z výhod takových robotů jsou nízké pořizovací náklady, a to díky značnému poklesu nákladů na mechanickou část.

Tato diplomová práce se bude zabývat tvorbou univerzálního SW nástroje pro výpočet kinematické transformace pro malý průmyslový robot, řízený PLC od společnosti Beckhoff.

2 Stav problematiky

2.1 Průmyslové roboty

Dobře formulovaný pojem „robot“ najdeme ve knize [1]: *„Robotem rozumíme počítačem řízený integrovaný systém, schopný autonomní a cílově orientované interakce s reálným prostředím v souladu s instrukcemi od člověka.“*.

Upřesnit se tento pojem ještě dá dalšími podmínkami, jako je např. komunikace s člověkem v umělém nebo přirozeném jazyce nebo schopnost vnímat a rozpoznávat prostředí atd. Obecně se ale dá říct, že robot je složité zařízení, které určitým způsobem napodobuje člověka a je schopné lidské schopnosti nejen rozšířit, ale i v některých případech nahradit.

Základní rozdělení robotů podle zaměření [2]:

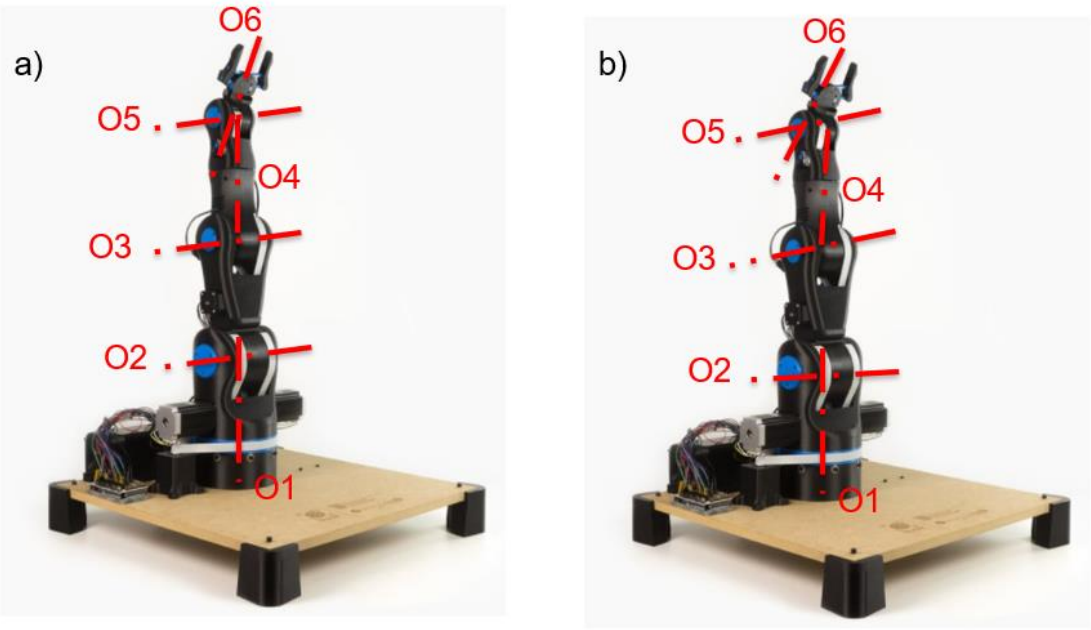
- Průmyslové roboty (Obr. 1 – a).
- Specializované, domácí a servisní roboty (Obr. 1 – b).
- Humanoidní roboty (Obr. 1 – c).

Obr. 1: Rozdělení robotů podle zaměření [2]: a) průmyslový robot, b) servisní robot, c) humanoidní robot

Pro pochopení celého obsahu práce je důležité uvést, že všechny roboty, které výrobce robotů dodávají, jsou tzv. **reálné** (zachybovaný nominální). Výpočetní metody pro výpočet dopředné a inverzní kinematické metody se ale počítají s robotem **ideálním**(nominální). Rozdíly mezi robotem reálným a ideálním nejdou poznat pouhým okem, tzn. že budou vypadat úplně stejně. Rozdíly však spočívají v odlišných geometrických rozměrech robotu. Jedná se o rozdíly v milimetrech a menší. Geometrické

rozměry jsou alternativně označovány jako kinematické parametry a patří mezi ně například délky ramen robotu a vzájemné orientace jednotlivých ramen robotu.

Na Obr. 2 jsou schematicky znázorněné robot ideální a robot reální, kde u reálného robotu je zřetelně vidět, že osy jednotlivých rotačních kloubů nejsou paralelní, a ani poslední tři osy O4, O5 a O6 se neprotínají ve stejném bodě, což je například nezbytnou podmínkou pro analytické řešení inverzní kinematiky.



Obr. 2: Schematické zobrazení rozdílů mezi robotem reálným (b) a ideálním (a).

Definice průmyslového robotu je stanovená dle [3] a zní: *„Průmyslový robot je autonomní programovatelný víceúčelový manipulátor s třemi nebo více osami, který může být stacionární nebo mobilní a je určen pro použití v průmyslu“*. V dnešní době je už těžce představitelný například automobilový průmysl bez používání průmyslových robotů.

Dle kinematiky se dají průmyslové roboty pak rozdělit na roboty s kinematikou:

- Sériovou (Obr. 1 – a).
- Paralelní (Obr. 3).

Obr. 3: Paralelní robot spol. ABB[4]

Nejznámější výrobci průmyslových robotů na českém trhu jsou:

1. ABB.
2. KUKA.
3. FANUK.

Výrobce open source robotů:

1. BCN3D.
2. NIRYO.

ABB

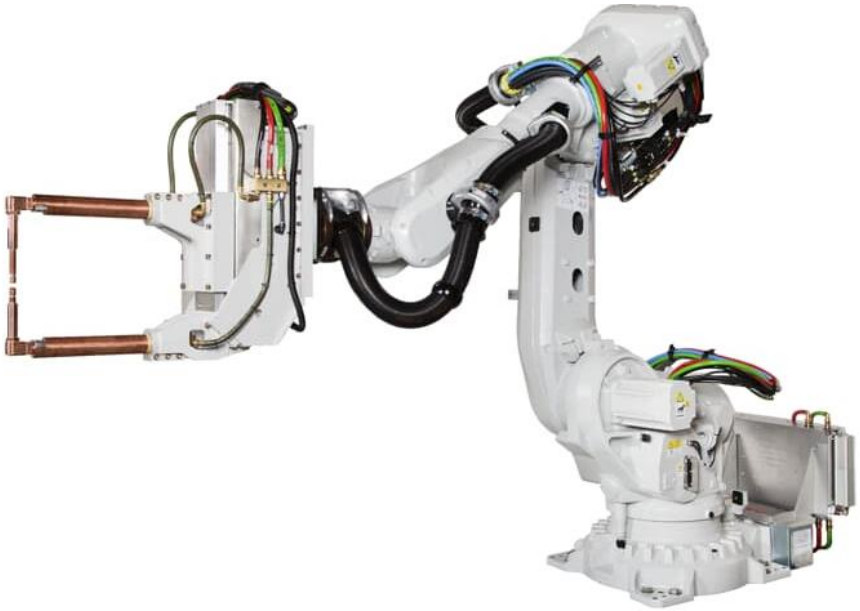
Společnost ABB je jeden z největších dodavatelů výrobků, služeb a řešení pro digitální průmysl. Mezi hlavní čtyři oblasti, kterými se zabývají, patří elektrotechnika, průmyslová automatizace, pohony a robotická automatizace.

Tato část společnosti ABB má rozsáhlou nabídku. Prodává jak roboty samotné, průmyslové i malé kolaborativní, tak i celé řešení problematiky dle poptávky zákazníka. Typickým příkladem použití robotů v průmyslu je robot pro manipulaci s těžkým materiálem. Pro takové manipulace ABB nabízí robota ze skupiny Power Robot IRB 6640, který je vidět na Obr. 4. [4]

Obr. 4: Robot IRB 6640 [4]

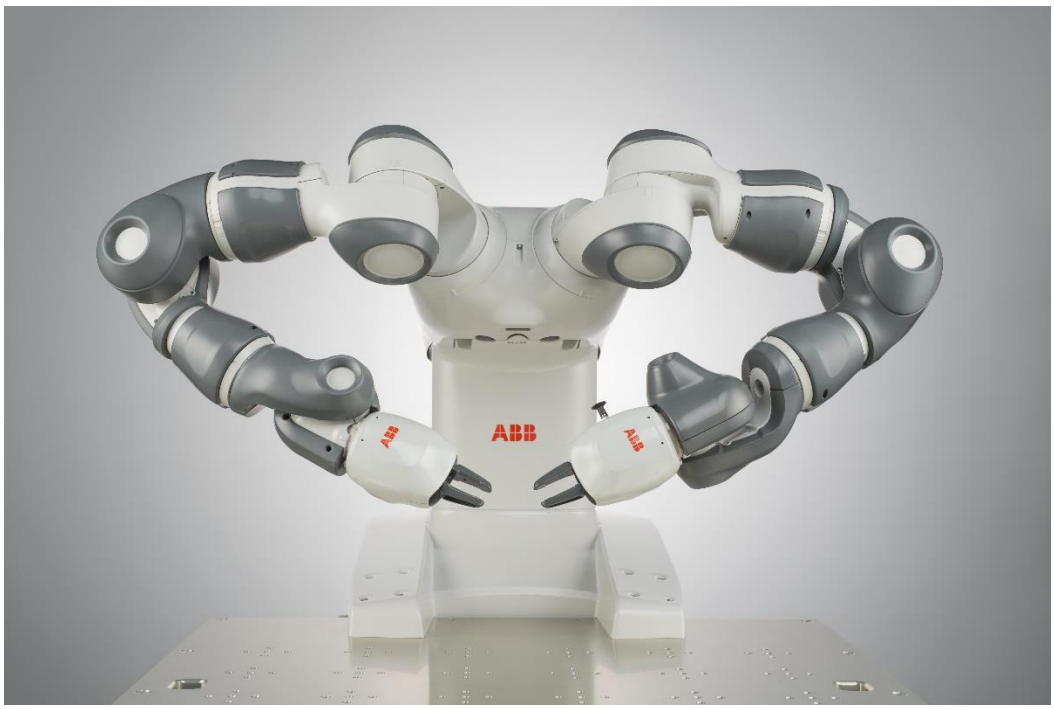
Dalším příkladem jsou svařovací roboty, které se dají použít jak na obloukové, tak i bodové svařování. Na Obr. 5 je vidět robot IRB 1520ID, který se dá použít na svařování obloukem a robot IRB 6620 se svařovací hlavou pro bodové svařování na Obr. 6. [4]

Obr. 5: Robot IRB 1520ID [4]



Obr. 6: Robot IRB 6620 se svařovací hlavou [4]

Od roku 2015 mají v nabídce i malého průmyslového kolaborativního robota YuMi (Obr. 7). Využívá se především při montáži drobných součástek a dokáže pracovat s člověkem. [4]



Obr. 7: Kolaborativní robot YuMi [4]

KUKA

Dalším známým výrobcem průmyslových robotů je společnost KUKA. Stejně jako ABB se zabývá poskytováním inteligentních automatizačních řešení v odvětvích jako automotive, elektronika, potravinářství, průmysl plastů a mnoho dalších. [5]

Příkladem manipulačního robota od společnosti KUKA muže být třeba robot řady KR QUANTEC – KR 120 R2700-2 (Obr. 8). [5]

Obr. 8: Robot KR 120 R2700-2 [5]

Mají také roboty kolaborativní, které se dají použít jak v práci spolu s člověkem, tak i bez, například u měření, anebo u balení, a ještě v zajímavém designu (Obr. 9). [5]

Obr. 9: Kolaborativní robot LBR Iiwa 7 [5]

FANUC

Společnost FANUC je především známý výrobce obráběcích strojů, ale v neposlední řadě je dodavatelem i průmyslových robotů. Výrobky této společnosti můžeme potkat relativně ve všech odvětvích průmyslu od automobilového až po zdravotnictví a hodinářství. [6]

Stejně jako ABB a KUKA mají různé třídy robotů pro různé aplikace.

Například pro manipulaci s těžkými objekty, jako jsou třeba naskládané palety, se dá použít robot řady R-2000 s nosností až 270 kg (Obr. 10). [6]

Obr. 10: Robot R-2000 iC [6]

V časti kolaborativních robotů mají docela velký výběr dle požadavků zákazníka. Jedním z novějších je robot CRX – 10iA/L, který je vidět na Obr. 11. [6]

Obr. 11: Robot CRX - 10iA/L [6]

BCN3D

Je to španělská společnost, která je zaměřena na výrobu 3D tištěných komponentů. Mají vyvinutou vlastní 3D tiskárnu, software, ale také malého testovacího robota.[7]

Robot představuje pětiosou robotickou ruku, která je řízena řídicí jednotkou (Obr. 12). Společnost BCN3D pro tento účel používá komponenty Arduino. V případě této práce bude použita řídicí jednotka od firmy Beckhoff. Další specifikace budou popsány v praktické časti práce.

Obr. 12: Robot BCN3D MOVEO [7]

NIRYO

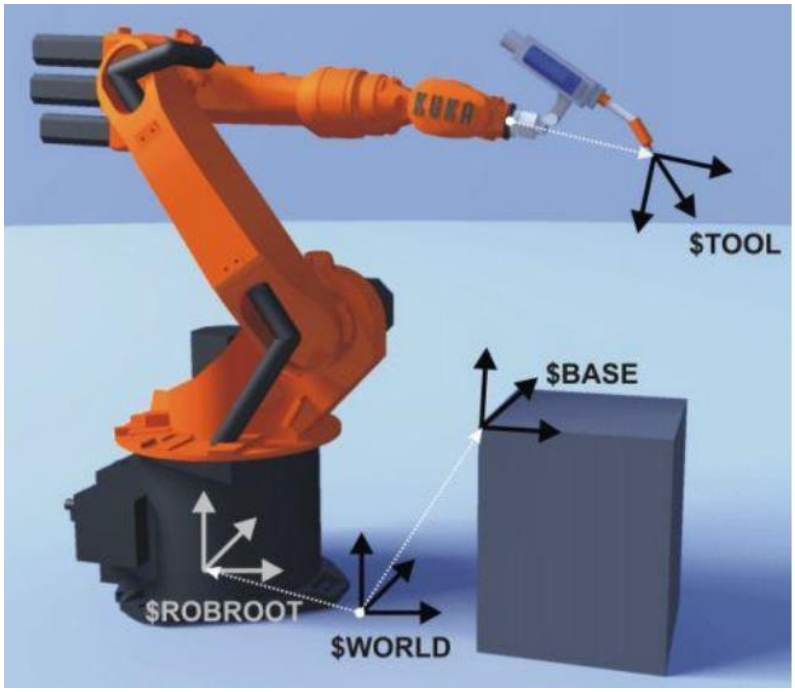
Dalším výrobcem a dodavatelem malých výukových robotů je společnost Niryo. Stejně jako firma BCN3D vyvinula robota (Obr. 13), který se dá skoro celý vytisknout na 3D tiskárně, výjimky jsou elektronické součástky robotu (motory, delší ramena atd.). Dodávají přímo ke svému robotu i software, který lze zadarmo stáhnout ze stránek společnosti. [8]

Pro řízení se dají použít řídicí jednotky od Arduino anebo Raspbbery Pi, avšak jsou i další možnosti.

Obr. 13: Robot Nyrio One [8]

2.1.1 Souřadnicové systémy robotu

Pro popis polohy tělesa v prostoru se používají souřadnicové systémy (ss) tělesa a základní souřadnicový systém, vůči němuž se těleso pohybuje. Stejně tak se pro přesný popis polohy robota v prostoru používá několik ss, které jsou vidět na Obr. 14.



Obr. 14: Souřadnicové systémy robota KUKA [9]

WORLD – je pevně definovaný světový kartézský souřadný systém robota. V základním nastavení je umístěn v patě robota. Je základním pro ss ROBROOT a BASE. [9]

ROBROOT – je kartézský souřadný systém robota, který určuje jeho pozici vůči světovému ss a je referenční pro mechanické konstrukce robota. Vždycky je umístěn v patě robota. [9]

BASE – je kartézský souřadný systém. Umístění řeší programátor. Je vztažený k ss WORLD a určuje polohu pracovního prostoru. [9]

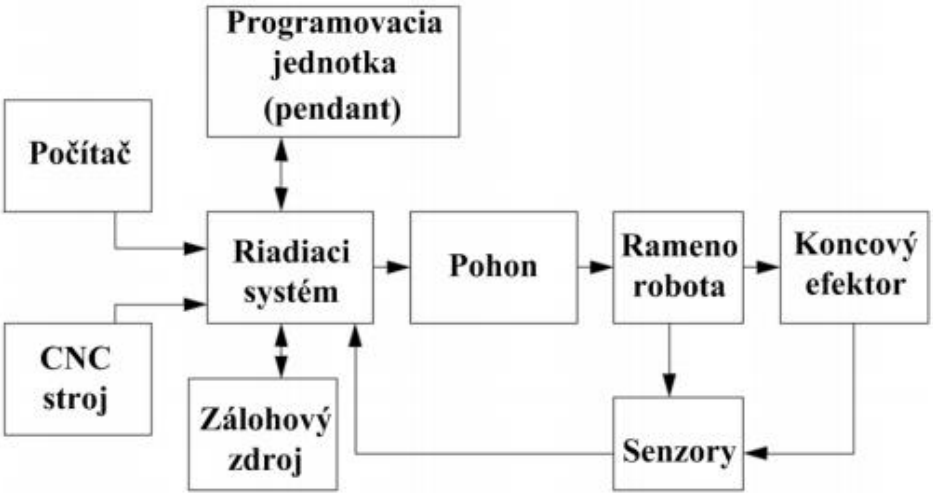
TOOL – je kartézský souřadný sytém, který je umístěn ve střením bodě nástroje (TCP – tool center point). Je definován programátorem a je vztažený ke ss BASE. [9]

2.2 Řídicí systém

Obecně se řídicím systémem (ŘS) myslí programovatelné zařízení, které je dle uživatelem připraveného programu schopné řídit určitý technologický proces. Velikost a složitost řídicího systému vychází z požadované funkčnosti procesu. Informace o stavu procesu či zařízení se dostává přes vhodně zvolené senzory, které jsou připojené k jeho vstupu. Počet vstupů a výstupů, které musí ŘS zpracovávat, má přímý vliv na jeho

složitost. Typickým příkladem jednoduchého řídicího systému muže být řídicí jednotka Arduino.

Řídicí systém zejména průmyslových robotů není tak jednoduchý. Obsahuje nejen řídicí jednotku, a i další prvky, které pomáhají robota ovládat a komunikovat s dalšími prvky procesu. Jednoduché blokové schéma je vidět na Obr. 15



Obr. 15: Blokové schéma řídicího systému průmyslového robota [10]

ABB

Příkladem nového ŘS od firmy ABB je IRC5 (Obr. 16), který je pátou generací řídicího systému robotů.

Skládá se ze dvou modulů [11]:

- Řídicí modul (Control module), který obsahuje všechno, co se týká řízení (např. hlavní počítač) a provozuje veškerý SW nutný na ovládání robota.
- Pohybový modul (Drive module), který obsahuje veškerou elektroniku, která ovládá motory.

Pro konfiguraci a programování robotů ABB využívá technický nástroj RobotStudio. Pomocí RobotStudio se dají programovat jak virtuální jednotky na PC, tak i „živé“ roboty ve fabrikách a dílnách. Programuje se pomocí editoru jazyka RAPID. To je programovací jazyk přímo od společnosti ABB [11].

Součástí systému IRC5 je také obslužní jednotka FlexPendant, která je vidět na Obr. 16 na rozvaděči. Slouží především k provádění úkonů při obsluze robotů a představuje malý samostatný počítač [11].

Obr. 16: Řídicí systém IRC5 [11]

KUKA

Řídicí systém KR C4 od firmy KUKA je vidět na Obr. 17. Je to poměrně nový ŘS, který v sobě zahrnuje čtyři nadstavby: Robot Control, PLC Control, Motion Control a Safety Control, které společně sdílí datovou základnu a běží na bázi Windows.

Kromě samotného PC, který má funkci vytváření programů, řízení sekvencí, komunikaci s dalšími kontroléry, důležitou součástí ŘS jsou tzv. KSP (KUKA Servo Packs), která řídí motory robotů. Je tedy jasně vidět, že většina ŘS průmyslových robotů je víceméně stejná, liší se jenom výrobce a občas způsob řízení. [9]

Softwarem pro offline simulace a konfigurace je KUKA SimPro. Pro programování se používá program Work Visual. Pro online programování a vlastní řízení robotů se používá KUKA.SystemSoftware (KSS), v případě ŘS KR C4 jde o verze KSS 8.6. [9]

Dalším prvkem ŘS KR C4 je smartPAD, který plní stejné funkce jako FlexPendant u řídicích systémů ABB. [9]

Obr. 17: Řídicí systém KR C4 [9]

FANUC

Příkladem řídicího systému od společnosti FANUC je R30iB Plus (Obr. 18). Jednou výhodou, kterou má řídicí systém R30iB proti jiným ŘS, které používají systém Windows (KUKA), je to, že běží na vlastním patentovaném SW od společnosti FANUC, což zvyšuje jeho bezpečnost. [6]

Stejně jako předchozí ŘS má R30iB kromě řídicí jednotky s počítačem prvek pro řízení servopohonu a iPendant pro online programování a servisní služby.

Softwarem pro tvoření programu offline a simulaci pohybu robotu je Roboguide. [6]

Obr. 18: Řídicí systém FANUC R30iB Plus [6]

Řídicí systém Beckhoff

Příkladem řídicího systému jsou i systémy běžící na bázi průmyslového počítače od společnosti Beckhoff. Existují různé varianty provedení, od výkonově nejmenších, jako jsou třeba embedded PC, až po průmyslové servery s výkonným procesorem s více jádry [12].

Jsou to průmyslové PC, na kterých běží software pro řízení PLC. Hlavní výhodou řídicích systémů společnosti Beckhoff je, že jsou tzv. hard-real-time, což znamená, že pracovní cyklus PLC je nastavován uživatelem a nemění se za běhu programu, ale ani při změně délky programu, což umožňuje používání těchto řídicích systémů v náročných a přesných aplikacích [12].

Jako software, který se používá pro programování PLC a který řídí real-time, se nazývá TwinCAT. Ve verzi TwinCAT 3 lze programovat všemi jazyky, která definuje norma pro programování PLC IEC 61131-3 [12].

Jedná se o tyto jazyky:

- IL – instruction list,
- ST – structured text,
- LD – ladder diagram,
- FBD – function block diagram,
- SFC – sequential function chart.

Výhodou programování ve TwinCAT 3 je to, že pokud máte zkušenosti například s jinými programovací jazyky, jako jsou C++ nebo JAVA, máte možnost vytvářet real-time aplikace použitím svých znalostí těchto jazyků [12].

Jako komunikační sběrnici používají PLC od Beckhoff – EtherCAT, ale díky připojovacím modulům lze integrovat i jiné komunikační sběrnice, jako jsou například PROFIBUS nebo PROFINET.

2.2.1 Programování robotů

Robot vykonává úlohy dle dopředu vloženého programu. Program je definován jako posloupnost příkazů, které vedou k vykonání požadované úlohy. Programování robotů potom není nic jiného než tvorba programů pro robota. Programování se dělí na [10]:

1. On-line programování – programování robota bezprostředně u robota přes jednotku „pendant“.
2. Off-line programování – programování na počítači, kdekoliv.

2.2.1.1 On-line programování

On-line programování se uskutečňuje přímo před robotem (Obr. 19). Manuálním řízením z programovatelného panelu programátor navádí robota na požadovanou polohu a po nastavení správné orientace koncového efektoru tento bod ukládá. Programátor zadává také parametry těchto bodů, které pak mají vliv na výsledný pohyb robota. Poté následuje programování logické části řízení, například otevírání a zavírání servokleští při bodovém svařování. [10]

Obr. 19: On-line programování robotu [5]

Za hlavní výhodou on-line programování bych považoval možnost svůj program hned (nebo už v průběhu) otestovat za reálných podmínek provozu.

Nevýhodou tohoto typu programování je především jeho pomalejší rychlost, která omezuje výrobní kapacity výrobních linek, ve kterých je daný robot použit.

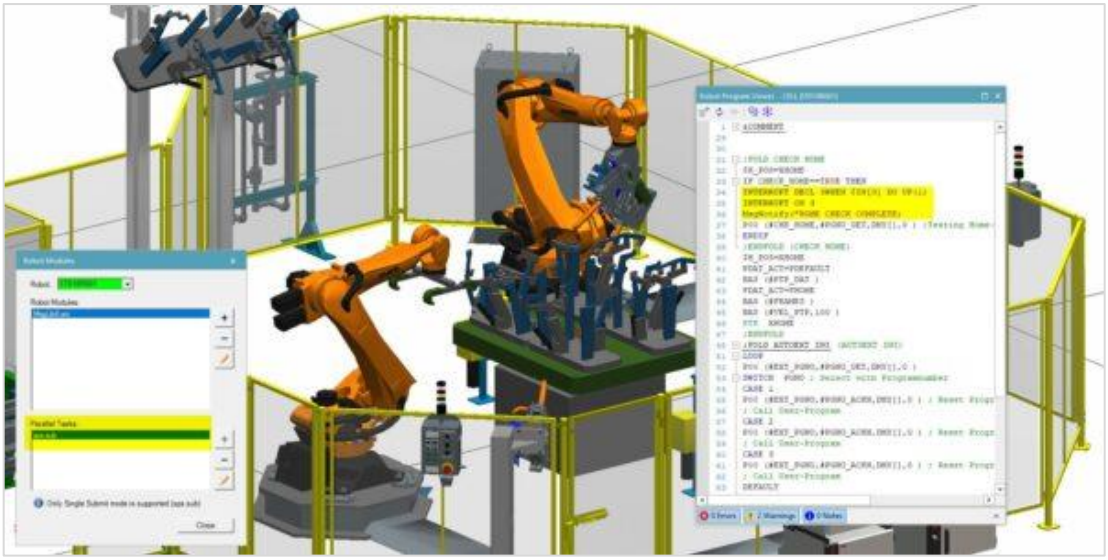
2.2.1.2 Off-line programování

Off-line programování je typ programování robotu, který nepotřebuje fyzickou přítomnost robotu nebo řídicího systému. Je to v dnešní době nejrozšířenější druh programování robotu. Program před tím, než se nahraje na řídicí systém reálného robota, je, většinou, otestován v programovacím prostředí. [10]

Na základě digitálního dvojčete robota a počítačového modelu buňky nebo linky se naprogramují robotické dráhy, které následně budou nasimulované v programovacím prostředí. Čím je lepší program, tím víc budou vygenerované dráhy odpovídat reálnému pohybu robota. V dnešní době takové programy nejen pomáhají správně nastavit pohyby robotů, ale i optimalizovat manipulační cykly. Je to především díky tomu, že umějí provádět simulace v reálném čase. Samozřejmě, že se to souvisí s dalšími náklady, a to jak na program, tak i na člověka, který umí v tomto prostředí programovat. [10]

Posledním krokem off-line programování je programování on-line. Robotický program vygenerovaný programovacím (simulačním) prostředím se nahraje na řídicí systém robota a v režimu on-line programátor projede celý program ještě jednou za snížené rychlosti. Zkontroluje se tím ještě jednou, zda nenastávají například kolizní stavy. [10]

Nejlepším příkladem jak simulačního, tak i programovacího prostředí je SW od společnosti Siemens – Tecnomatix Process Simulate (Obr. 20).



Obr. 20: Tecnomatix Process Simulate[5]

3 Cíle diplomové práce

Hlavním cílem diplomové práce je vytvoření a implementace SW nástroje pro počítání inverzní kinematiky šestiosého sériového robotu, který umožní pohyb robotu v kartézském souřadnicovým systému. Součástí je také připravení PLC projektu a nastavení NC úlohy pro simulační testování.

Pro splnění hlavního cíle byla práce rozdělena na následující dílčí úlohy:

- Návrh metody výpočtu inverzní kinematiky pro reálného robota.
- Ověření navrhnuté metody v SW MATLAB.
- Implementace kódu z MATLAB do C++.
- Implementace do real-time aplikaci v prostředí TwinCAT.
- Simulační testování.

4 Návrh struktury řízení

Tato část se zabývá návrhem struktury řízení robotu. Součástí návrhu řízení je metoda výpočtu inverzní kinematické úlohy. Navržená metoda byla nejdříve napsána a otestována v prostředí MATLAB (verze R2020a), následně byla přepsána do jazyku C++ s použitím Visual Studio 2019 a implementována pro řízení robotu v reálném čase v programovacím prostředí pro programování PLC společnosti Beckhoff – TwinCAT3.

Další částí diplomové práce bylo vytvoření a konfigurace NC úlohy a PLC projektu, který bude použit pro simulační testování. NC úloha odpovídá za nastavení jednotlivých os robotu a také za přepočet mezi souřadnicemi kartézskými a kloubovými. V PLC části bude provedena veškerá logika, která se týká jak řízení jednotlivých os, tak i vytvoření a ovládání inverzní a dopředné kinematiky.

4.1 Kinematika robotu

Tato kapitola se bude zabývat teorií výpočtu kinematické úlohy robotu, a to jak dopředné, tak i inverzní. Na ni bude navazovat návrh vlastní metody a ověření.

Obecně je úlohou kinematiky popsat pohyb bez ohledu na síly, které tento pohyb způsobují. Na začátku probereme úlohu a problémy kinematiky dopředné, pak i inverzní.

4.1.1 Dopředná kinematická úloha

Úlohou dopředné kinematické úlohy je určit polohu a orientaci koncového efektoru v závislosti na hodnotách kloubových souřadnic robotu. Jinými slovy je to transformace z kloubových souřadnic do souřadnic pracovního prostoru. To se matematicky dá popsat jako [13]:

$$\vec{q} = T(\vec{q}) \tag{1}$$

, kde je $\vec{q}$ - vektor zobecněných souřadnic, T- transformační matice.

Pro popis transformačních matic v této práci se bude používat Denavit-Hartenbergova konvence (dále jen DH). DH konvenci je možné použít pouze pro otevřené kinematické

řetězce, které tvoří posloupnost těles spojených klouby, kde poslední kloub je odlišný od prvního. Podle této konvence lze každou homogenní transformace A_i popsat pomocí čtyř základních transformací:

- Rot_{z,θ_i}- rotace kolem osy z_i o úhel θ.
- $Trans_{z,d_i}$ – translace mezi osy x_i a x_{i-1} o vzdálenost d_i.
- $Trans_{x,a_i}$ – translace mezi osy z_i a z_{i-1} o vzdálenost a_i.
- Rot_{z,θ_i}- rotace kolem osy x_i o úhel α.

$$T_i = Rot_{z,\theta_i} \cdot Trans_{z,d_i} \cdot Trans_{x,a_i} \cdot Rot_{x,\alpha_i} =$$

$$= \begin{bmatrix} c_{\theta_i} & -s_{\theta_i} & 0 & 0 \\ s_{\theta_i} & c_{\theta_i} & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 & a_i \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & c_{\alpha_i} & -s_{\alpha_i} & 0 \\ 0 & s_{\alpha_i} & c_{\alpha_i} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} c_{\theta_i} & -s_{\theta_i}c_{\alpha_i} & s_{\theta_i}s_{\alpha_i} & a_i c_{\theta_i} \\ s_{\theta_i} & c_{\theta_i}c_{\alpha_i} & -c_{\theta_i}s_{\alpha_i} & a_i s_{\theta_i} \\ 0 & s_{\alpha_i} & c_{\alpha_i} & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2}$$

, kde θ_i- natočení osy x_i vůči ose x_{i-1} kolem osy z_{i-1}, d_i- vzdalenost mezi x_i a x_{i-1} podél z_{i-1}, a_i- vzdalenost mezi z_i a z_{i-1} podél x_i a α_i- natočení osy z_{i-1} vůči z_i kolem osy x_i.

Proměnným $\theta_i d_i a_i \alpha_i$ se říká DH parametry a jsou přidružené ramenu i a kloubu i. V případě kinematiky robotu parametry a_i a α_i jsou konstanty a jeden z parametru d_i nebo θ_i se mění v závislosti na typu kloubu – posuvný (θ_i je konstanta) nebo otočný (d_i je konstanta) [16].

Pokud označíme neznámý parametr za q_i, dostaneme:

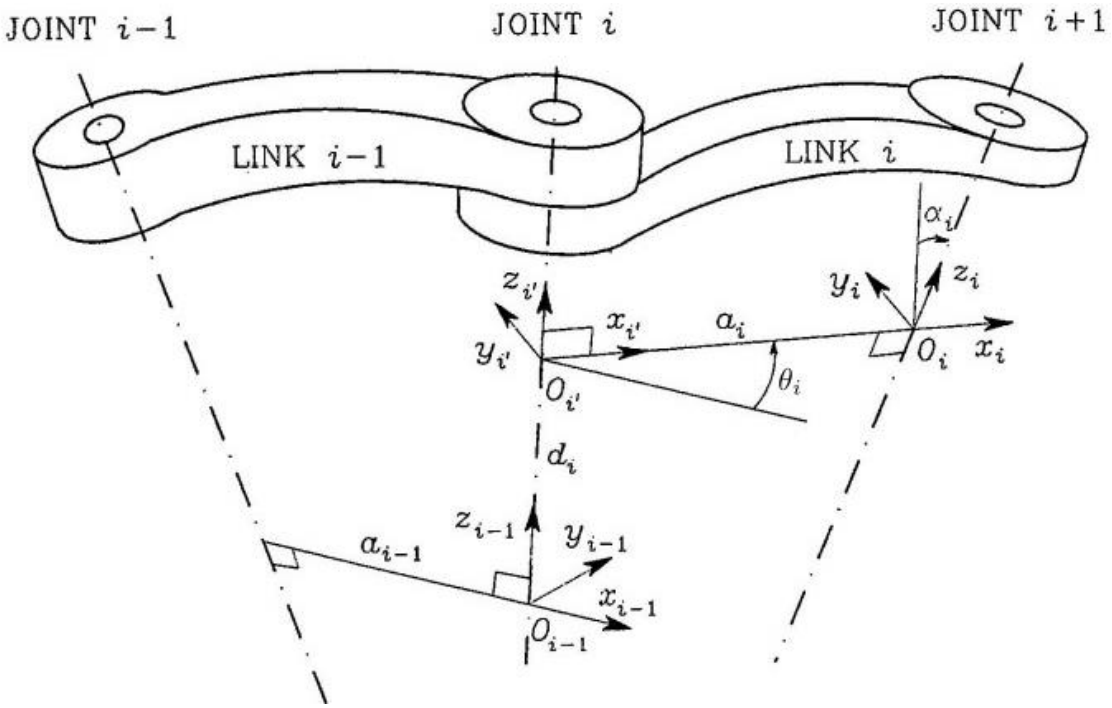
$$x^0 = T_1^0(q_1)T_2^1(q_2)T_3^2(q_3) \dots T_n^{n-1}(q_n)x^n \tag{3}$$

, kde x^0 – souřadnice hledaného bodu vůči základnímu souřadnicovému systému.

Pro každou hodnotu vektoru $q = (q_1, q_2, q_3, \dots, q_n)$ můžeme vypočítat souřadnice hledaného bodu vůči základnímu souřadnicovému systému ze zadaných souřadnic v systému, ve kterém se tento bod nachází.

Pro otevřený řetězec bude dopředná kinematická úloha vždy řešitelná analyticky [13].

Na Obr. 21 je vidět, jak se podle DH konvenci určují souřadnicové systémy kinematických členů robotů.



Obr. 21: DH princip zavadění souřadnicových systémů [15]

Jsou ale případy, kdy použít DH konvenci jednoznačně nemůžeme. Existují ale i další konvence, které tyto případy zahrnují. V této práci si však s DH konvencí vystačíme.

4.1.2 Inverzní kinematická úloha

Tato kapitola bude zaměřena na úlohu, která je opačná než ta předchozí. Ve většině případů se při pohybu s robotem chceme dostat na nějakou polohu v prostoru, což znamená, že jediné, co známe, jsou souřadnice koncového bodu v základním souřadném systému. Úlohou inverzní kinematiky je tím pádem najít všechny neznámé parametry kloubů pro dosažení zadané polohy efektorem. To se dá matematicky vyjádřit rovnicí [13]:

$$T(q) = T_1^0(q_1)T_2^1(q_2)T_3^2(q_3) \dots T_n^{n-1}(q_n) \tag{4}$$

, kde T je daná transformační matice, která určuje polohu bodu v základním souřadným systému a q_i- neznámé parametry [13].

Úloha inverzní kinematiky může být poměrně složitá a může zahrnovat výpočet soustavy nelineárních rovnic. Nejde ji vždy vyřešit analyticky, oproti dopředné kinematické úloze, a řešení může nabývat nekonečný počet hodnot, především kolem tzv. singulárních bodů. Inverzní kinematická úloha pro robot se šesti stupně volnosti řeší soustavu šesti nelineárních rovnic o šesti neznámých, což jak už bylo řečeno, muže být komplikované.

4.1.2.1 Metody řešení inverzní kinematické úlohy

Inverzní kinematickou úlohu lze řešit několika způsoby. Ty jsou dále závislé například na rychlosti počítání, anebo na univerzálnosti. U robotů jsou požadavky na rychlost počítání vyšší než na univerzálnost, protože se musí pohybovat v tzv. real-time (reálném čase).

Obecně lze metody řešení rozdělit na metody analytické a numerické. Numerické metody se dále skládají z metod optimalizačních a aproximačních. Každá z metod má svoje výhody a nevýhody. Tato práce se nejvíce bude zajímat o metody analytické, protože právě tyto metody budou použity při psaní programu v praktické části práce.

Analytické metody

Analytickou metodu výpočtu inverzní kinematiky pro šestiosého robota lze použít za podmínky, že je struktura řešitelná. O řešitelnosti rozhoduje kinematické uspořádání os robotů.

Podmínky řešitelnosti [13]:

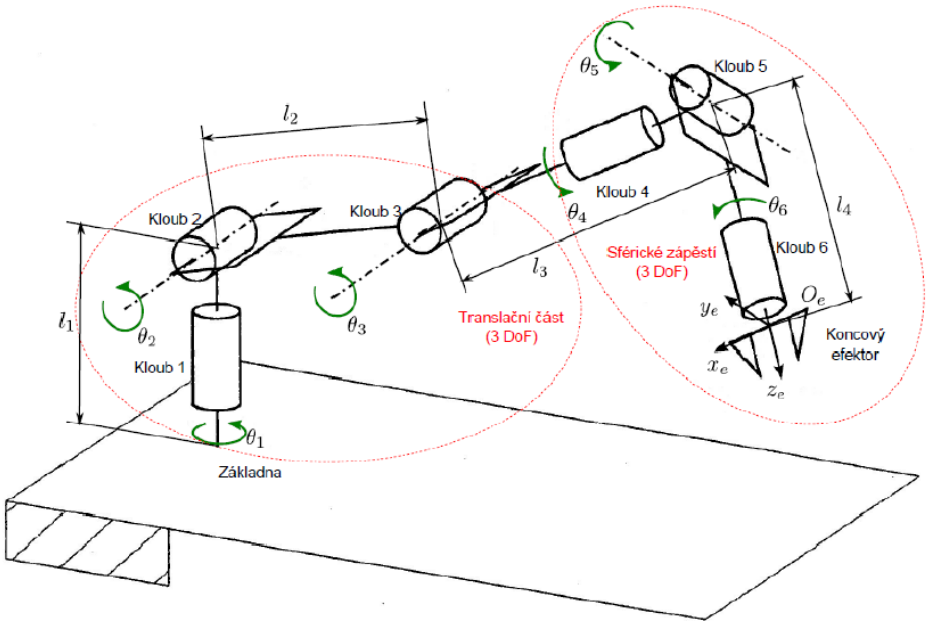
1. Pro robota se šesti stupni volnosti tři po sobě jdoucí rotační klouby mají osy protínající se v jednom bodě nezávisle na pohybu.
2. Pro robota se šesti stupni volnosti tři po sobě jdoucí osy musí být rovnoběžné nezávisle na pohybu.

Příkladem řešitelné struktury robotu je i například robot BCN3D MOVEO (Obr. 12).

Obecný postup u analytické metody neexistuje. Navíc se muže stát, že k řešení se ani nedostaneme. Právě proto se při modelování robotických struktur hned uvažují podmínky řešitelnosti. Tím se ale omezuje univerzálnost této metody. Na druhou stranu za předpokladu, že nebylo použité žádné zjednodušení, se dostaneme do poměrně přesného řešení dané úlohy [14].

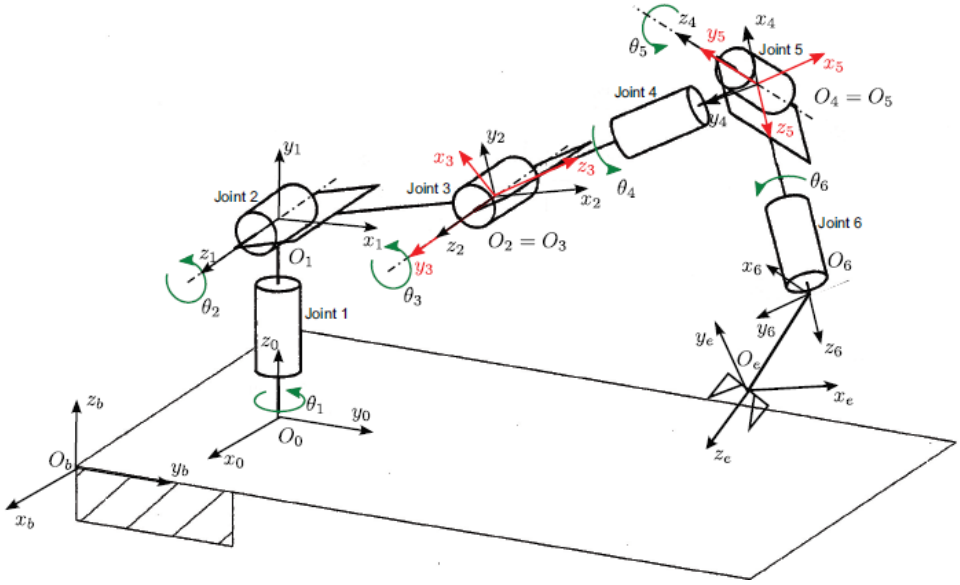
Příklad použití analytické metody pro řešitelnou strukturu šestiosého robota:

Jak je vidět z Obr. 22, máme řešitelnou strukturu šestiosého robota z první podmínky řešitelnosti, tzn. že osy třech po sobě jdoucích rotačních kloubů se protínají ve stejném bodě, a to nezávislé na pohybu. Jsou to osy kloubu 4, 5 a 6.



Obr. 22: Řešitelná struktura robotu se šesti stupni volnosti [15]

Dále zavedeme souřadnicové systémy podle DH úmluvy (Obr. 23) a tím zjistíme potřebné konstanty (Tab. 1).



Obr. 23: Zavedení souřadnicových systému dle DH úmluvy [15]

Kloub	d	a	α	θ
1	l_1	0	π/2	θ_1
2	0	l_2	0	θ_2
3	0	0	π/2	θ_3
4	l_3	0	-π/2	θ_4
5	0	0	π/2	θ_5
6	l_4	0	0	θ_6

Tab. 1: DH parametry pro zvolenou strukturu robotu

Úhly θ_1 $a\check{z}$ θ_6 jsou ty inverzní kinematickou úlohou hledané. Je důležité zmínit, že toto jsou obecné DH parametry, které platí pro každého robota se strukturou, jak je na Obr. 23. Lišit se budou pouze hodnotami.

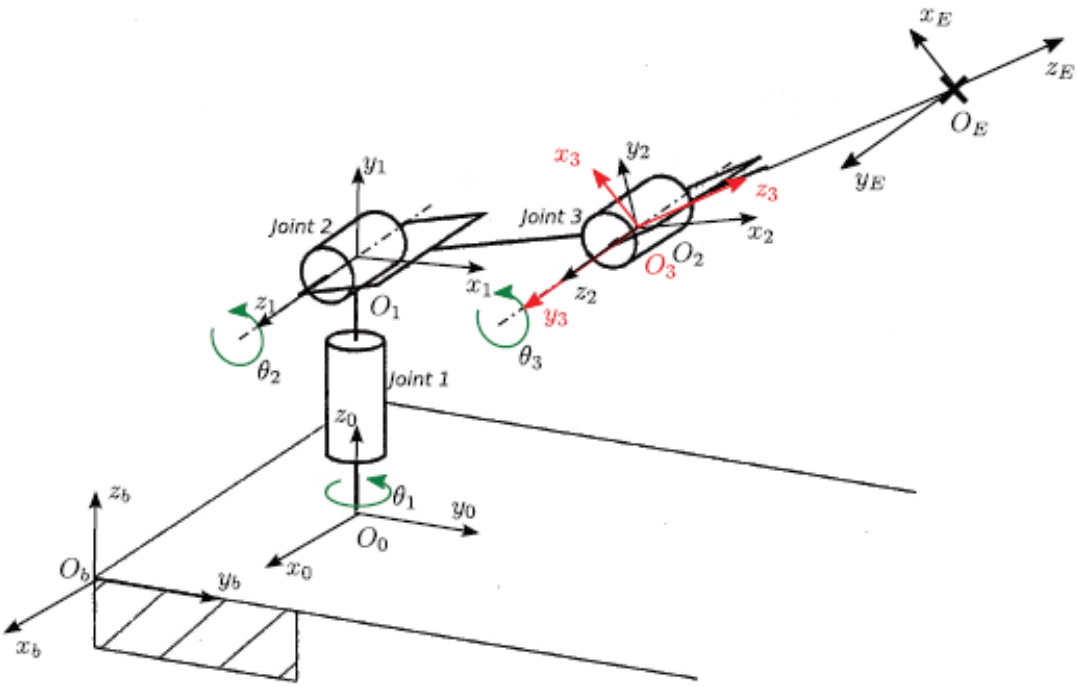
Po získání DH parametru se provede kinematická dekompozice robotu na tzv. část translační a sférické zapěsti, které jsou vyznačené na Obr. 22. Jde to provést právě kvůli zvolené struktuře robotu. Tím, že byla provedena dekompozice, je složitá úloha rozdělená do dvou menších a skoro triviálních úloh (Obr. 24, Obr. 26). Pořád ale jde o řešení soustavy nelineárních rovnic.

Translační část

Souřadnice koncového bodu translační části robota lze vypočítat ze znalosti transformační matice souřadnicového systému č. 6 - T_6^1 a délky ramena l_4:

$$X_E = O_4^0 = O_5^0 = O_6^0 - z_6^0 \cdot l_4 \tag{5}$$

, kde z_6^0 – třetí sloupec transformační matici T_6^1.



Obr. 24: Translační část úlohy [15]

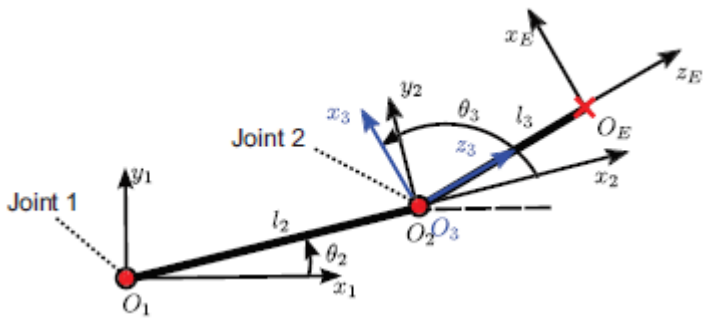
V této pozici je dobře vidět problém, který nastává u robotů se sériovou kinematikou – singulární polohy. Tato problematika je důkladněji popsaná na konci části.

Kvůli tomu, že nastává tzv. singulární stav, budou řešení pro úhel θ_1 dvě:

$$\theta_1 = tan^{-1}\frac{O_{E_y}^0}{O_{E_x}^0} \; a \; \theta_1 = tan^{-1}\frac{O_{E_y}^0}{O_{E_x}^0} + \pi \tag{6}$$

, kde $O_{E_y}^0$ – ypsilonová souřadnice středu ss O_E vůči nultému ss robota, analogicky platí pro $O_{E_x}^0$.

Tím, že jsme našli řešení pro θ_1, převádíme naší dekomponovanou sestavu na ještě jednodušší sestavu planárního robota, kterou schematicky lze nakreslit, jak je vidět na Obr. 25.



Obr. 25. Schematické uspořádání planárního robotu [15]

Dále je potřeba najít souřadnice středu O_E vůči prvnímu souřadnicovému systému.

$$O_E^0 = T_1^0 \cdot O_E^1 \tag{7}$$

, pak

$$O_E^1 = (T_1^0)^{-1} \cdot O_5^0 \tag{8}$$

Z toho získáme xsové a ypsilonové souřadnice bodu O_E a následovně je vyjádříme přes geometrické funkce z Obr. 25. Dostaneme:

$$O_{E_x}^1 = a = l_2 \cos\theta_2 + l_3 \cos(\theta_3 + \theta_2) \tag{9}$$

$$O_{E_y}^1 = b = l_2 \sin\theta_2 - l_3 \cos(\theta_3 + \theta_2) \tag{10}$$

Pomocí součtových goniometrických vzorců se dostaneme k řešení pro úhly θ_2 a θ_3 ve tvaru:

$$\theta_2 = \tan^{-1}\frac{\sin\theta_2}{\cos\theta_2} \tag{11}$$

$$\theta_3 = \tan^{-1}\frac{\sin\theta_3}{\cos\theta_3} \tag{12}$$

Existence inverzní kinematické úlohy můžeme definovat po jednoduchých úpravách takto:

$$l_2 - l_3 \leq \|O_E^1\| \leq l_2 + l_3 \tag{13}$$

Z toho plyne, že celkový počet řešení pro translační část je 4 (dva pro úhel θ_1 a další dva z existence inverzní kinematické úlohy).

Sférické zapěsti

Zobecněné souřadnice pro sférické zapěstí (Obr. 26) jdou zapsat ve tvaru:

$$X_{sf} = R_6^3 \tag{14}$$

, kde R_6^3 je matice směrových kosinu z transformační matice T_6^3. Matici T_6^3 získáme pomocí transformačních matic T_3^0 a T_6^0 , a to následujícím způsobem:

$$T_6^0 = T_3^0 \cdot T_6^3 \tag{15}$$

, pak

$$T_6^3 = (T_3^0)^{-1} \cdot T_6^0 \tag{16}$$

T_3^0 známe z výpočtu části translační a T_6^0 známe ze zadání (vstupní hodnoty inverzní kinematické úlohy).

Podle DH úmluvy matici T_6^3 lze rozepsat do tvaru:

$$T_6^3 = \begin{bmatrix} c\theta_4 c\theta_5 c\theta_6 - s\theta_4 s\theta_6 & -c\theta_4 c\theta_5 c\theta_6 - s\theta_4 c\theta_6 & c\theta_4 s\theta_5 & O_{6_x}^3 \\ s\theta_4 c\theta_5 + c\theta_4 s\theta_6 & -s\theta_4 c\theta_5 s\theta_6 + c\theta_4 c\theta_6 & s\theta_4 s\theta_5 & O_{6_y}^3 \\ -s\theta_5 c\theta_6 & s\theta_5 s\theta_6 & c\theta_5 & O_{6_z}^3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{17}$$

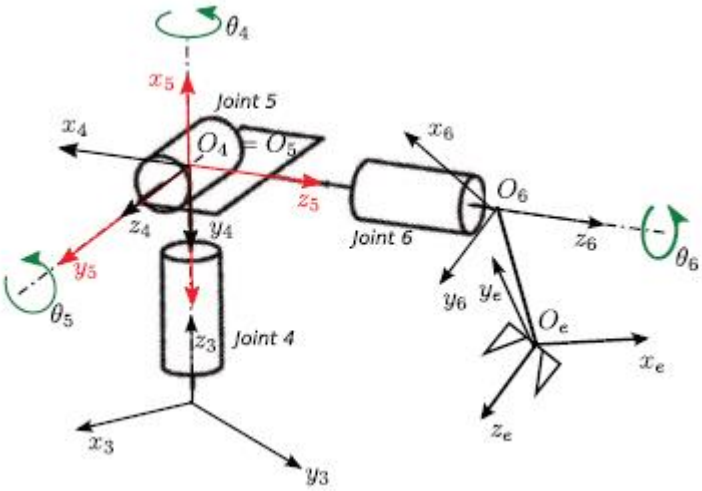
, kde O_6^3 jsou souřadnice středu ss č. 6 vůči ss č. 3.

Znovu pomocí goniometrických vzorků dostáváme řešení pro úhly θ_4, θ_5 a θ_6 ve tvaru:

$$\theta_4 = tan^{-1} \frac{sin\,\theta_4}{cos\,\theta_4}$$

$$\theta_5 = tan^{-1}\frac{sin\,\theta_5}{cos\,\theta_5}$$

$$\theta_6 = tan^{-1}\frac{sin\,\theta_6}{cos\,\theta_6}$$



Obr. 26: Sférické zapěsti [15]

Podrobné řešení je rozebrané ve [15].

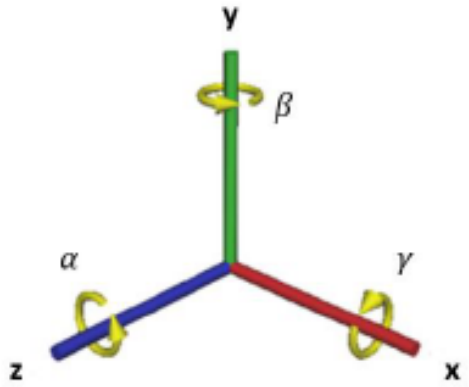
Numerické metody

Jak už bylo naznačeno dříve, inverzní kinematická metoda není nic jiného než řešení soustavy nelineárních rovnic. To znamená, že k výpočtu můžeme používat kromě analytických metod také známé metody numerické. Tyto metody můžou být použity pro různé kinematické struktury robotů. Nevýhodou je ale to, že k přesnému řešení se můžeme buď nedostat vůbec, anebo se dostat, ale po delší době. Také je třeba dávat pozor na stabilitu a na podmínky, za kterých konkrétní metoda konverguje. Přesnost numerických metod jde zadávat, nastavením malé přesnosti si můžeme zvětšit rychlost konvergování řešení, ale přijdeme o přesnost polohy koncového bodů.

Z obecně známých numerických metod výpočtu jsou například Newtonova aproximační metoda, anebo metoda gradientní.

Prvním krokem, který musíme provést, než se do nějaké z metod pustíme, je vyjádřit soustavu rovnic, kterou budeme řešit.

Pro popis polohy koncového bodu robotu použijeme translace a rotace v kartézském souřadnicovém systému robotu { $x_0, y_0, z_0,\ O_0$ } pomocí šesti parametrů $[x, y, z, \alpha, \beta, \gamma]$, kde část $[x, y, z]$ odpovídá translaci a $[\alpha, \beta, \gamma]$ rotací vůči základnímu (nultému) souřadnicovému systému robotu. Úhel $\alpha -$ odpovídá rotaci kolem osy Z, $\beta -$ kolem osy Y a $\gamma -$ kolem osy X (viz Obr. 27).



Obr. 27: Rotace kolem základního ss robotu

Dále transformační matici, se kterou jsme se už setkali v analytickém řešení inverzní kinematické úlohy, lze obecně zapsat ve tvaru:

$$T = \begin{bmatrix} R & P \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & x \\ r_{21} & r_{22} & r_{23} & y \\ r_{31} & r_{32} & r_{33} & z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

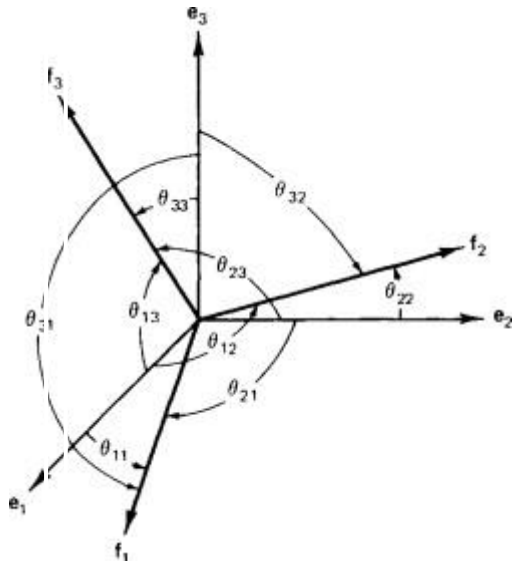
, kde matice R je matice směrových kosínů popisující natočení následujícího ss vůči předešlému, a vektor P je vektor translace.

První tři rovnice pro angulárního robot se šesti stupně volnosti dostaneme tak, že z transformační matici T_6^0, kterou jsme dostáli postupným přenásobením matic T_1^0 až T_6^5 podle DH úmluvy, vezmeme prvky, které odpovídají translačním prvkům x, y, z a položíme je rovně stejným prvkům ze vstupní maticí. Tyto tři rovnice se dají zapsat ve tvaru:

1. $T_{6_{DH}}^0(1,4) = T_{vstup}(1,4) - translace\ podel\ X.$
2. $T_{6_{DH}}^0(2,4) = T_{vstup}(2,4) - translace\ podel\ Y.$
3. $T_{6_{DH}}^0(3,4) = T_{vstup}(3,4) - translace\ podel\ Z.$

Další tři rovnice vycházejí z matice směrových kosínů R. V této části je důležitý vybrat tři prvky z matice směrových kosínů, které jednoznačně popisují rotace jednoho ss vůči druhému.

Z Obr. 28 je vidět, že pro jednoznačný popis můžeme použít diagonální prvky matice směrových kosínů.



Obr. 28: Interpolace matice směrových kosínů

To znamená, že další tři rovnice můžeme zapsat ve tvaru:

4. $T^0_{6_{DH}}(1,1) = T_{vstup}(1,1)$.
5. $T^0_{6_{DH}}(2,2) = T_{vstup}(2,2)$.
6. $T^0_{6_{DH}}(3,3) = T_{vstup}(3,3)$.

Teď už máme šest rovnic o šesti neznámých (jednotlivá natočení kloubu θ_1 až θ_6) a můžeme na to aplikovat známé numerické metody, například Newtnovou aproximační metodu.

Newtonova aproximační metoda

Newtonova aproximační metoda je metodou iterační (krokovou) a řeší soustavu nelineárních rovnic ve tvaru:

$$F(\vec{X}) = 0 \tag{18}$$

, kde $F(\vec{X}) = [\, f_1(q_1),\, f_2(q_2), \dots,\, f_n(q_n)]^T$, převedením na soustavu lineárních rovnic pomocí Taylorova rozvoje, která vypadá takto:

$$F(\vec{X}) + J(\vec{X}) \cdot \Delta = 0 \tag{19}$$

, kde $F(\vec{X}) = [\, g_1(x_1),\, g_2(x_2), \dots,\, g_n(x_n)]^T$, $J(\vec{X}) -$ tzv. Jacobiho matice a Δ −přírůstek.

Jednou z podmínek jedinečného řešení soustavy je regularita Jacobiho matice, tzn.:

$$J(\vec{X}) = \begin{pmatrix} \frac{\partial f_1}{\partial q_1} & \cdots & \frac{\partial f_1}{\partial q_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_n}{\partial q_1} & \cdots & \frac{\partial f_n}{\partial q_n} \end{pmatrix} \tag{20}$$

, pak $det\, J \neq 0$.

Pokud je Jacobiho matice regulární, znamená, že má poslední soustava jediné řešení Δ^k (pří k ≥ 0) a lze definovat $x^{(k+1)} = x^{(k)} + \Delta^{(k)}$ (21), kde $\Delta^{(k)} -$ je řešením soustavy lineárních rovnic $F(\vec{X}) + J(\vec{X}) \cdot \Delta = 0$.

Obecný postup řešení pro první iterace:

1. k = 0
2. Zvolíme počáteční aproximace $X^{(0)}$ dostatečně blízko přesnému řešení X^*
3. Spočítáme $F(X^{(0)})$ a $J(X^{(0)})$
4. Vyřešíme soustavu $F(X^{(0)}) + J(X^{(0)}) \cdot \Delta^{(0)} = 0$ pro $\Delta^{(0)}$
5. $X^{(1)} = X^{(0)} + \Delta^{(0)}$

Pokud je splněno $\|X^{(0)} - X^*\| < \delta$, označíme $X^{(0)}$ za řešení soustavy, pokud splněno není, uděláme to samé pro k = 1, a tak dále, dokud nebude splněna podmínka přesnosti.

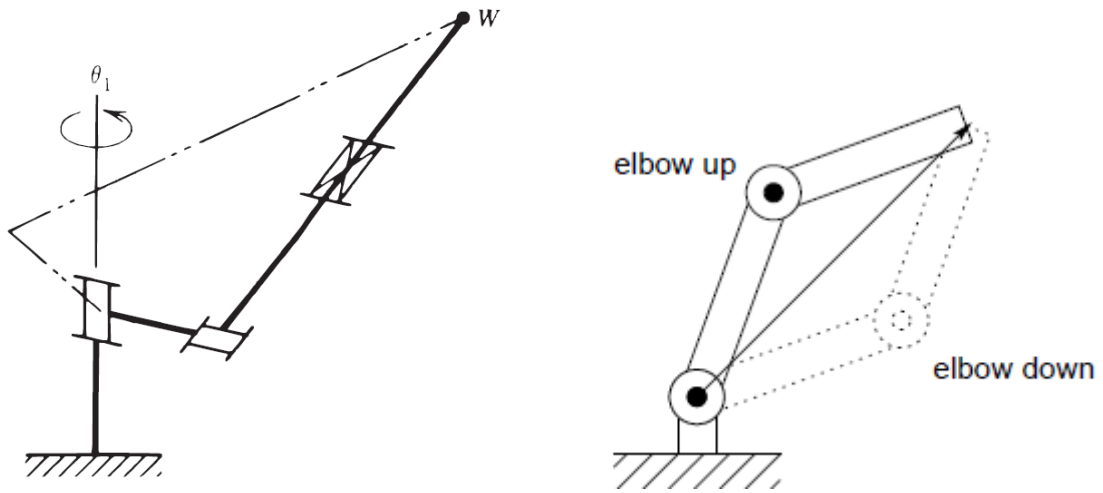
Přesnost výsledků je určovaná parametrem δ, který udává, jak blízko přesného řešení X^* je třeba volit počáteční aproximaci $X^{(0)}$:

$$\|X^{(0)} - X^*\| < \delta \tag{22}$$

Detailněji postup lze najít v [17].

4.2 Singulární polohy robotu

Jedná se o takové polohy koncového bodu robotu nebo efektoru, do kterých se můžeme dostat více způsoby. To znamená, že existují takové body v prostoru, pro které platí víc (může být i nekonečně mnoho) řešení inverzní kinematické úlohy, což v značné míře komplikuje řízení robotu a klade další požadavky na metodu počítání. Zjednodušené příklady singulární polohy manipulátoru jsou vidět na Obr. 29.



Obr. 29: Příklady singulární polohy manipulátoru [13]

Pro vyšetřování singulárních poloh robota se používá Jacobiho matice (J), o které se už mluvilo v předchozí kapitole.

Obecně soustava nelineárních rovnic pro řešení inverzní kinematické úlohy vypadá tímto způsobem:

$$\begin{cases} f_1(q_1, \dots, q_6) = 0; \\ f_2(q_1, \dots, q_6) = 0; \\ \dots \\ f_6(q_1, \dots, q_6) = 0; \end{cases} \tag{23}$$

Jacobiho matice je potom matice prvních parciálních derivací funkcí podle neznámých q_1 až q_6 a odpovídá matici (7). Dále singulární polohu sériového manipulátoru (robotu) můžeme chápat jako polohu koncového bodu, ve které Jacobiho matice ztrácí hodnost – rank (J) < min (m, n), kde m a n jsou stejné a reprezentují počet sloupců a řádků Jacobiho matice [18].

Při uvažování, které řešení máme použít, je potřeba se zaměřit na tyto body [13]:

1. Zda jsou vypočítané hodnoty úhlů natočení kloubu dosažitelné pro konstrukci robotu. Každá osa má svoje limitní úhly natočení, které fyzicky nelze přesáhnout.
2. Jakou cestou do singulárního bodu najíždíme. Pro plynulou trajektorii a chod robotu je důležité plynule měnit konfigurace robotu.
3. Jakým směrem budeme pokračovat ze singulárního bodu dál.
4. Jestli je operační prostor omezený, což nám odebere několik řešení, při kterých robot do požadovaného prostoru ani nevejde. Příkladem může být montáž přední palubky automobilu do karoserie, kde je robot zcela omezen prostorem karoserie.

Podle [16] je důležité určit singulární polohy z těchto důvodů:

- Singulární polohy představují konfigurace robotu, ze kterých jsou určité směry pohybu nedosažitelné.
- V singularitách omezené rychlosti koncového efektoru mohou korespondovat s nekonečnými rychlostmi kloubů.
- V singularitách omezené síly koncového efektoru mohou korespondovat z nekonečnými silami ve kloubech.
- Singularity jsou často odpovídají hraničním bodům pracovního prostoru robota.
- Singularity odpovídají bodům v pracovním prostoru, které mohou být nedosažitelné při malých odchylkách parametrů, jako je délka, offset atd.
- Blíže singularit neexistuje jednoznačné řešení pro inverzní kinematickou úlohu. V tomto případě řešení muže nabývat nekonečně mnoho hodnot, anebo nemusí vůbec existovat.

4.3 Metoda výpočtu kinematiky

Jak už bylo řečeno v předchozí kapitole, existují dvě kinematické úlohy – dopředná a inverzní. Dopředná kinematická úloha řeší polohu a orientace koncového bodu robotu v závislosti na kloubových souřadnicích jednotlivých kloubů (úhly natočení os). Je jednoduchá a vždy řešitelná analytickým způsobem. Inverzní kinematická úloha řeší problém opačný, a v závislosti na kinematické struktuře muže být pro analytické řešení hodně složitá. Kromě toho existují různé zdroje chyb, které vnáší do výsledků výpočtu

nepřesnosti, a to jak u úlohy dopředné, tak i u inverzní. Jedním z takových zdrojů můžou být geometrické chyby robotu. Vlivem takových chyb se robot ideální stává robotem reálným (zachybovaný nominální), což se musí uvažovat při výpočtu.

Při návrhu metody byl uvažován robot reálný, což znamená, že obsahuje geometrické chyby. Navržená metoda používá oba popsané způsoby výpočtu, a to tak, že se analytickou metodou nejdříve spočítají úhly natočení os **ideálního** robotu. Tyto úhly jsou použité jako počáteční podmínky numerické metody řešení inverzní kinematiky robota **reálného**. Díky dobře zvoleným počátečním podmínkám z analytické metody(řešení inverzní kinematické úlohy reálného robotu je blízké řešení ideálního robotu) je možné numerické řešení provést v reálném čase, tzn. stačí jen pár iterací.

Dopředná kinematika se pak používá pro programování robotu ve kloubových souřadnicích, inverzní – pro klasické programování v kartézském souřadnicovém systému.

Dopředná kinematika

Dopředná kinematická úloha je vždy řešitelná analyticky. Z matematického hlediska se při výpočtu dopřední kinematiky jedná jen o násobení transformačních matic ve správném pořadí. Pokud je použita Denavit-Hartenbergova konvence, transformační matice vždycky bude ve stejném tvaru:

$$T_i^{i-1} = \begin{bmatrix} c_{\theta_i} & -s_{\theta_i}c_{\alpha_i} & s_{\theta_i}s_{\alpha_i} & a_i c_{\theta_i} \\ s_{\theta_i} & c_{\theta_i}c_{\alpha_i} & -c_{\theta_i}s_{\alpha_i} & a_i s_{\theta_i} \\ 0 & s_{\alpha_i} & c_{\alpha_i} & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

, kde T_i^{i-1} je transformační matice mezi ss i a $i-1$ a θ, α, a, d jsou DH parametry robotu.

Výpočet spočívá ve vytvoření transformačních matic dle DH konvence a na postupném násobení.

```
%**** Obecna definice transformacni matice ****

T1 = [ cos(fi1) -sin(fi1)*cos(alp1) sin(fi1)*sin(alp1) a1*cos(fi1);...
T2 = [ cos(fi2) -sin(fi2)*cos(alp2) sin(fi2)*sin(alp2) a2*cos(fi2);...
T3 = [ cos(fi3) -sin(fi3)*cos(alp3) sin(fi3)*sin(alp3) a3*cos(fi3);...
T4 = [ cos(fi4) -sin(fi4)*cos(alp4) sin(fi4)*sin(alp4) a4*cos(fi4);...
T5 = [ cos(fi5) -sin(fi5)*cos(alp5) sin(fi5)*sin(alp5) a5*cos(fi5);...
T6 = [ cos(fi6) -sin(fi6)*cos(alp6) sin(fi6)*sin(alp6) a6*cos(fi6);...

%T(1) - transofrmacni matice T01, T(2) - tm T12 atd.

T16 = T1*T2*T3*T4*T5*T6;
```

Obr. 30:Výpočet dopředné kinematiky

Výsledkem je také transformační matice, která popisuje polohu a orientace koncového souřadnicového systému robotu, kde je koncový bod robotu začátkem posledního ss. Poslední sloupec matice reprezentuje X, Y a Z souřadnice koncového bodu. Na Obr. 31 vyznačená matice 3x3 odpovídá matici směrových kosínů, která reprezentuje natočení souřadnicového systému kolem základního (nultého) ss robotu.

```
T16 =

A, B, C <-- [ 0.5902   0.2250   0.7753 ]   [ 0.3048 ] --> X
            [-0.7898  -0.0378   0.6122 ]   [ 0.3685 ] --> Y
            [ 0.1670  -0.9736   0.1554 ]   [ 0.3338 ] --> Z
                  0         0        0       1.0000
```

Obr. 31: Transformační matice koncového bodu šestiosého robotu

Inverzní kinematika

Existuje několik metod řešení inverzní kinematické úlohy. Jsou to metody analytické a numerické. Detailněji byly tyto metody rozebrány v kapitole č. 4.1.2. V této části bude rozebrána navržená metoda, která zahrnuje oba přístupy.

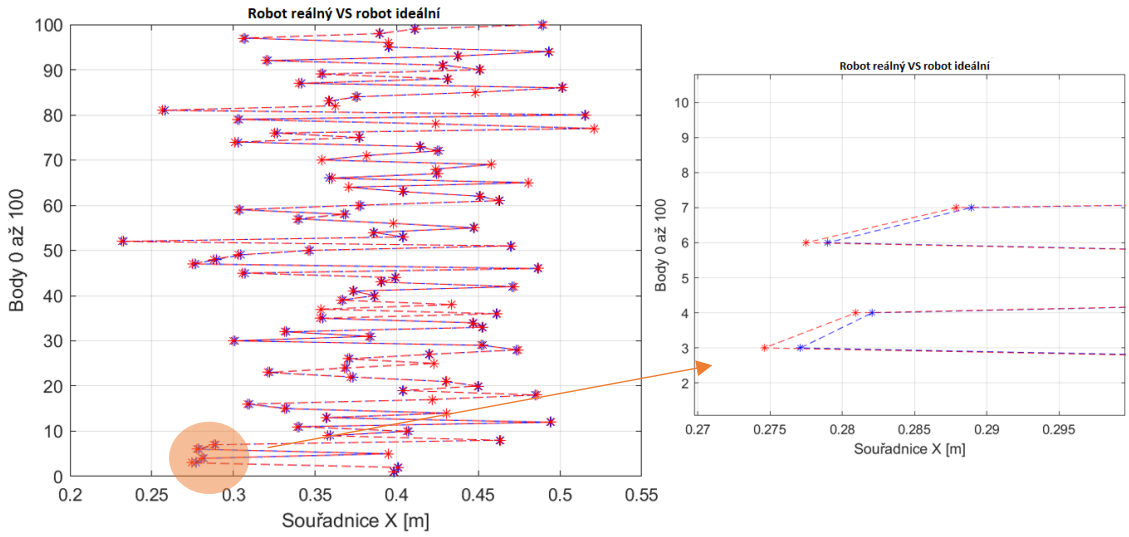
Jak už bylo naznačeno dříve, úlohou diplomové práce bylo vytvořit SW nástroj pro řízení sériového šestiosého robotu, včetně výpočtu inverzní kinematiky. Inverzní kinematika se bude počítat s robotem reálným. Výsledkem inverzní kinematické úlohy jsou úhly natočení jednotlivých kloubu (os) robotu.

Pro testovací účely byl vytvořen simulační model robotu s DH parametry a jejich offsety, které reprezentují geometrické chyby, které jsou vidět na Obr. 32. Dále náhodným způsobem byly 100krát zvoleny úhly natočení kloubu J1 až J6 a dopřednou kinematikou dopočítané koncové polohy robotu ideálního a reálného (parametry + offset). Na Obr. 33 je pak vidět rozdíl v souřadnicích X koncového bodu robotu ideálního a reálného.

```
%**** DH Konstanty kloubu ****

d1 = sym(0.15,'d'); %[m]
d2 = 0; %[m]
d3 = 0; %[m]
d4 = sym(0.15,'d'); %[m]
d5 = 0; %[m]
d6 = sym(0.1,'d'); %[m]

a1 = 0; %[m]
a2 = sym(0.3,'d'); %[m]
a3 = 0; %[m]
a4 = 0; %[m]
a5 = 0; %[m]
a6 = 0; %[m]

alp1 = sym(1.57,'d'); %[rad]
alp2 = 0; %[rad]
alp3 = sym(1.57,'d'); %[rad]
alp4 = sym(-1.57,'d'); %[rad]
alp5 = sym(1.57,'d'); %[rad]
alp6 = 0; %[rad]
```

```
%******** Offsets *******
d1_ofs = 0.000254; %[m]
d4_ofs = 0.0001; %[m]
d6_ofs = 0.000354; %[m]
a2_ofs = - 0.000283; %[m]
alp1_ofs = 0.002; %[rad]
alp2_ofs = 0.003; %[rad]
alp3_ofs = 0.004; %[rad]
alp4_ofs = 0.0052; %[rad]
alp5_ofs = 0.01; %[rad]
alp6_ofs = 0.012; %[rad]
```

Obr. 32: Model simulačního robotu

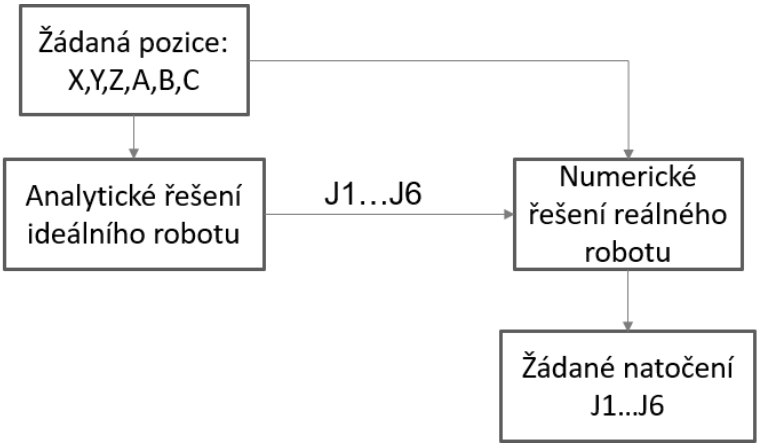


Obr. 33: Porovnání výsledku mezi robotem reálným a ideálním

Jak je vidět z grafu, výsledná chyba polohování v ose X je v hodně případech více než milimetr. Takové osy jsou tři, což nám výrazně mění koncovou polohu robotu, kam jsme se chtěli při programování dostat.

Navržená metoda počítání inverzní kinematiky je funkční jak pro robota ideálního, tak i pro robota reálného. Cíle je dosaženo použitím obou metod – analytické i numerické.

Základním zadáním pro metodu, která počítá inverzní kinematickou úlohu reálného robotu, je zjistit takové natočení kloubu (os) robotu J1 až J6, aby se reálný robot dostal do stejné polohy jako robot ideální. Způsob, jak navržená metoda zvládá tento problém, je zobrazeno na Obr. 34. Toto schéma představuje kompletní řešení, které bylo použito.



Obr. 34: Schéma počítání inverzní kinematiky navrženou metodou

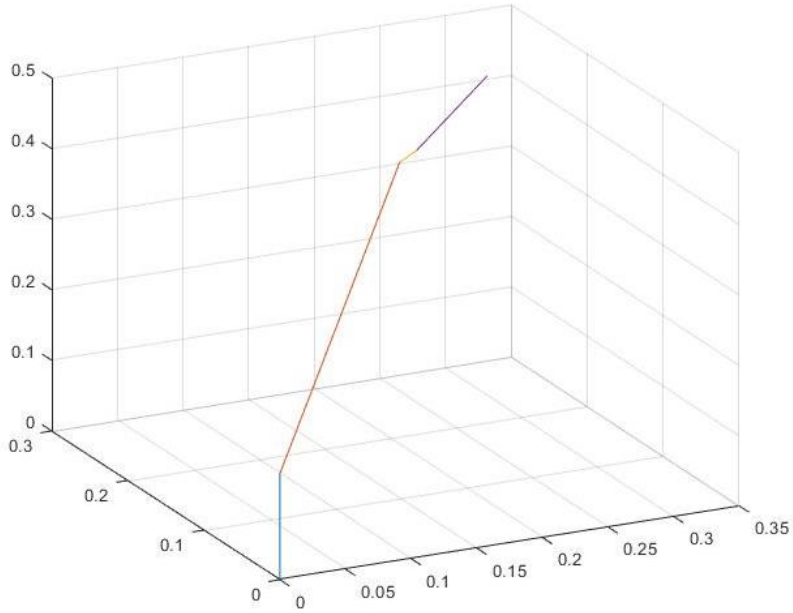
Vstupem do úlohy inverzní kinematiky je **žádaná poloha** koncového bodu robotu. Představuje souřadnice koncového bodu **X, Y, Z** (Obr. 35) a natočení kolem základního souřadnicového systému robotu **A, B, C**. Pro zjednodušení při testování metody byla vstupem transformační matice T16, která byla vygenerována dopřednou kinematikou s pevně stanovenými úhly J1 až J6 (Obr. 36). Dopřednou kinematikou se mimo jiné ověřilo, že výsledný bod je pro zvolený robot dosažitelný.

X → 0.31530750793997970849125571654879
Y → 0.27169095980922802370766747119 12
Z → 0.42875163616753043406017547351084
1

Obr. 35: X, Y a Z souřadnice vstupní matici T16

```
fi1 = 0.7791; %[rad]
fi2 = 0.8720; %[rad]
fi3 = 0.6099; %[rad]
fi4 = 0.3907; %[rad]
fi5 = 0.8358; %[rad]
fi6 = 0.5226; %[rad]
```

Obr. 36: Pevně stanovené úhly pro generování vstupní matice T16



Obr. 37: Schematické zobrazení robotu pro dané natočení os

Analytickou metodou jsou dále spočítané úhly natočení kloubu **ideálního** robotu (Obr. 38). Jsou to úhly, při kterých ideální robot dosáhne žádané polohy. Tyto úhly použijeme jako odhad nulté iterace u metody numerické.

```
teta1 :
0.7792
teta3 :
0.6098
teta2 :
0.8720
teta5 :
0.8358
teta6 :
0.5227
teta4 :
0.3911
```

Obr. 38: Analytickou metodou spočítané úhly ideálního robotu

Dosadíme-li analytickou metodou spočítané úhly do dopředné kinematiky reálného(zachybovaného ideálního) robota, zjistíme polohu koncového bodu reálného robotu Obr. 39. Je vidět, že v takovém případě není dosažena přesná(žádaná) poloha (Obr. 35). Přesná poloha je pak dosažená použitím natočení kloubů z numerického řešení.

```
X ⟶ 0.31425190946552235192523020964101
Y ⟶ 0.27330991235566374952661060162622
Z ⟶  0.42936472859143317583654410189 17
                                      1
```

Obr. 39: X, Y a Z souřadnice reálného robotu

Numerickým řešením následovně dopočítáme úhly natočení kloubu robota **reálného**. Výsledné úhly jsou úhly hledané (Obr. 40).

nultý odhad

```
phi0 =
    0.7793    0.8720    0.6099    0.3912    0.8359    0.5228
phiSolved_V2 =
    0.7802    0.8744    0.6082    0.4262    0.8322    0.4870
```

Úhly natočení os reálného robotu

Obr. 40: Numerické řešení výsledných úhlů reálného robotu

Při návrhu metody byla jako numerický řešič použita funkce *.fsolve( )* v MATLAB, jako algoritmus počítání byl použit levenberg-marquardt.

Iteration	Func-count	f(x)	Norm of step	First-order optimality	Trust-region radius
0	7	0.000685562		0.0271	1
1	14	3.36122e-07	0.0506563	0.000493	1
2	21	1.14598e-14	0.000650008	7.51e-08	1

Obr. 41: Informace z numerického řešiče

Dle Obr. 41 je vidět, že mezi výsledným řešením a nultým odhadem je dostatečně málo iterací, což nám umožnuje použití navrhnuté metody v real-time aplikacích.

Pro ověření správnosti metody byly výsledné úhly dosazené do dopředné kinematiky reálného robota, výsledkem pak je transformační matice (Obr. 42), která byla na začátku použita jako vstup do inverzní kinematické úlohy. Tím jsme ověřili to, že se **reálný** robot

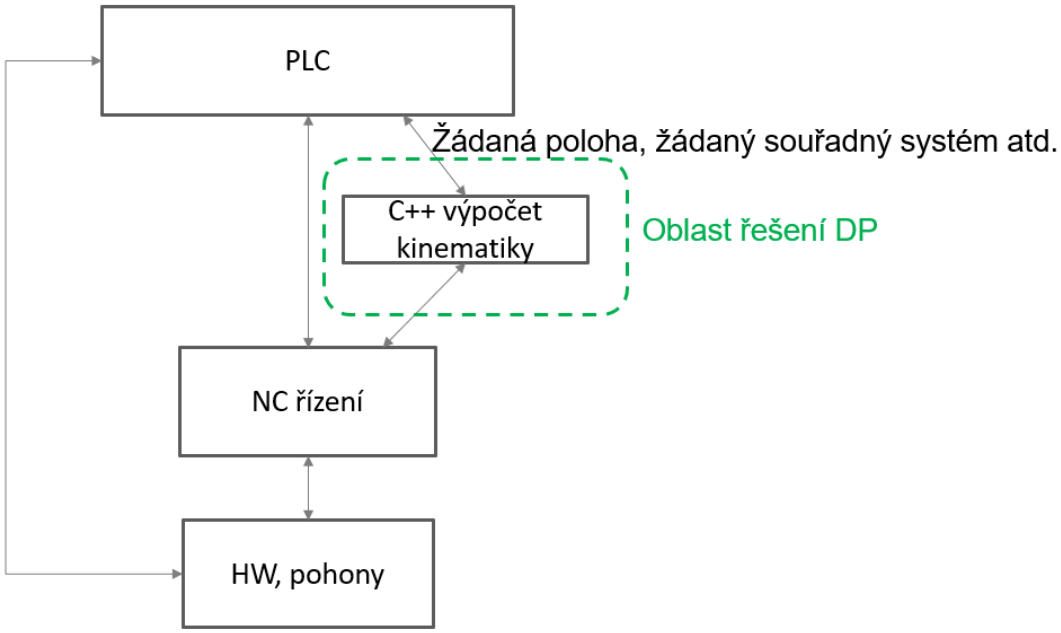
při výsledných uhlech dostane do stejného bodu jako robot **ideální** při počátečních, pevně zvolených uhlech.

X → 0.31530499882455923569083170253 74
Y → 0.27168810772062083611368349947967
Z → 0.42876256040465995475787285887356
1

Obr. 42: X, Y a Z souřadnice transformační matice reálného robotu

4.4 Schéma řízení

Pro simulační testování vycházíme z toho, že robota bude řídit PLC od společnosti Beckhoff. Kromě PLC řízení, které ovládá samotné PLC, je součástí i NC řízení, ve kterém bude aplikována inverzní a dopředná kinematika. Kompletní schéma je pak vidět na Obr. 43.



Obr. 43: Schéma řízení

PLC a NC řízení jsou spojené prostřednictvím PLC tagů, které jsou nalinkované na každou osu, která byla vytvořena v NC úloze. Pohony robotu pak jde ovládat jak z NC úlohy (přímo zadávat povely ose), tak i z PLC. C++ modul, který počítá inverzní

a dopřednou kinematiku, se nachází v části NC, ale také pracuje s daty, která PLC a NC řízení používají.

5 Implementace do real-time

V této části je popsán postup implementace navrhnutého způsobu počítání inverzní a dopředné kinematiky, včetně NC úlohy, PLC projektu a C++ modulu pro uživatelskou kinematickou transformaci. Implementace byla provedena ve SW pro programování PLC od společnosti Beckhoff – TwinCAT3. Tento SW je propojený z Visual Studio 2019 pro programování C++ části.

SW, který je, byl použit:

- TwinCAT (verze 3) build 4024.17.
- Visual Studio 2019 (důležitý je nejdříve nainstalovat TwinCAT a pak při instalaci VS nainstalovat doplněk pro TwinCAT).

Balíčky/knihovny:

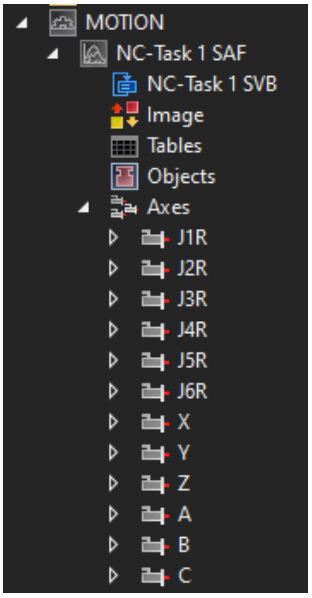
- Balíček TwinCAT3 Advanced Motion Pack.
- Knihovna TC3 TF511x L4.

Po instalaci všeho SW a balíčku je potřeba ve VS založit prázdný TwinCAT XAE projekt.

5.1 NC úloha

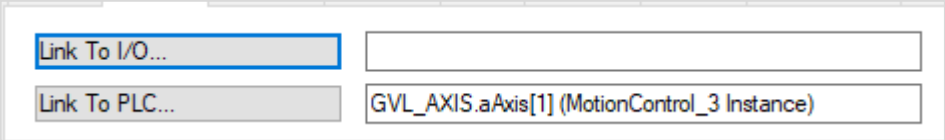
NC úloha je samostatná část projektu, která odpovídá za nastavení jednotlivých os robota, a to jak přímo ovládaných (fyzické motory), tak i programovatelných (virtuální osy X, Y, Z, A, B, C). Tyto osy je potřeba vytvořit a nastavit dle aktuálního použití.

Ve záložce *MOTION* se vytvoří *NC-Task*, následovně se do složky *Axes* se přidají potřebné osy (Obr. 44). Pro řízení sériového šestiosého robotu bude vytvořeno celkem dvanáct os – J1 až J6 (fyzické) a X, Y, Z, A, B, C (programovatelné).



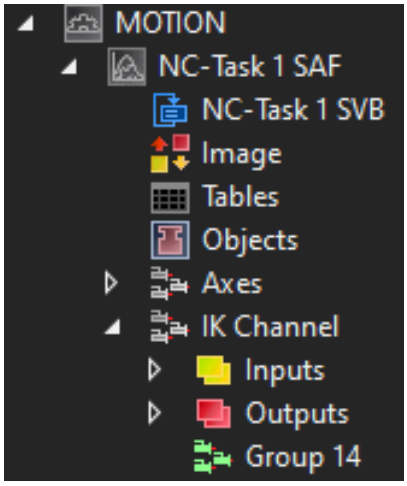
Obr. 44: Vytváření os pro řízení robotu

Dále je potřeba projit nastavením os a nastavit je dle aktuálních požadavků. Každá osa má být nalinkována na PLC prostřednictvím PLC tagu. PLC tagy potřebné pro linkování se vytvářejí v PLC projektu.



Obr. 45: Linkování osy na PLC tag

Pro kinematickou transformaci do záložky *NC-Task* se přidá speciální *Channel*, ve kterém se budou počítat dopředná a inverzní kinematika. Po přidání má záložka *MOTION* vypadat, jak je ukázáno na Obr. 46. Do *Group 14* se na konci přidá C++ modul s navrhnutou a implementovanou metodou.



Obr. 46: Přidání Channel pro počítání kinematických transformací

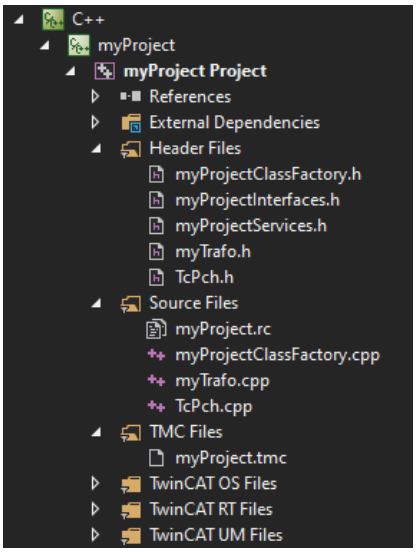
5.2 Modul pro uživatelskou transformaci

Dalším krokem je přidání C++ modulu, ve kterém bude navržená metoda implementována. Detailní postup lze najít ve [21]. V této části bude uveden postup zkrácený.

Postup pro vytvoření modulu pro externí kinematickou transformaci:

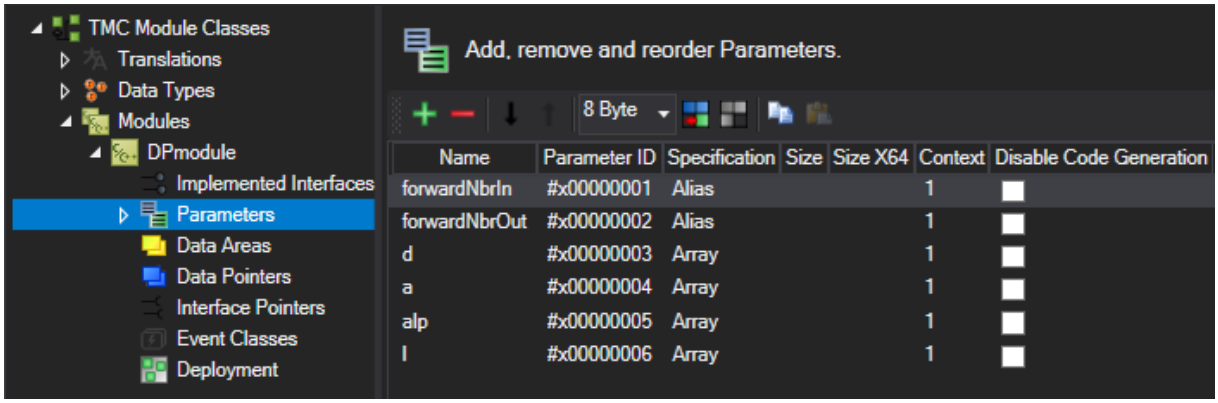
1. Přidat TwinCAT Drive Project ve záložce *C++*.
2. Jako modul zvolit TwinCAT Module Class.

Po přidání projektu a modulu má záložka *C++* vypadat jako na Obr. 46.



Obr. 47: Vytváření C++ projektu

3. V .tmc souboru v záložce *Implemented interfaces* přidat *ITcNcTrafo* interface.
4. Nakonfigurovat *Parameters* dle Obr. 48.



Obr. 48: Nakonfigurované parametry

forwardNbrIn je počet jointových os robotu a *forwardNbrOut* počet virtuálních os a jsou typu INT, pak *d, a, alp, l* jsou DH konstanty typu array of LREAL.

5. Spustit *TMC Code Generator*

Po automatickém generování potřebných souborů a části kódu v souboru *<NazevModulu>.cpp* vzniknou funkce pro implementaci uživatelské kinematické transformace (viz Obr. 49).

```
///<AutoGeneratedContent id="ImplementationOf_ITcNcTrafo">
HRESULT CmyTrafo::Forward(TcNcTrafoParameter* p)
{

    HRESULT hr = E_NOTIMPL;
    return hr;
}

HRESULT CmyTrafo::Backward(TcNcTrafoParameter* p)
{

    HRESULT hr = E_NOTIMPL;
    return hr;
}
```

Obr. 49: Funkce pro implementace dopředné a inverzní kinematiky

Důležitou částí kódu je přidání inicializace počtu virtuálních a jointových os. Dělá se to v autogenerované funkci *GetDimension( )*. Funkční způsob inicializací je na Obr. 49.

```
HRESULT CDPmodule::GetDimensions(ULONG* pFwdInput, ULONG* pFwdOutput)
{
    HRESULT hr = S_OK;

    if (pFwdInput && pFwdOutput)
    {
        *pFwdInput = m_forwardNbrIn;
        *pFwdOutput = m_forwardNbrOut;
    }
    else
    {
        hr = E_POINTER;
    }

    return hr;
}
///</AutoGeneratedContent>
```

Obr. 50: Inicializace jointových a virtuálních os

5.3 PLC projekt

Pro řízení PLC, kromě jiného, je vždycky potřeba ho naprogramovat. Tato část práce se bude zabývat programováním řídicího PLC, které bylo použito při simulačním testování, které se dá i v budoucnu použít pro řízení reálného robotu.

PLC program se od klasických programů liší tím, že se provádí v neustále smyčce, která trvá v řadech desítek milisekund. To je při programování PLC nutné uvažovat. V každém cyklu smyčky se PLC program načítá vstupy, provádí celý program a zapíše hodnoty do výstupů. Při simulačním testování v této práce bylo použité tzv. softwarové PLC. To znamená, že operační systém počítače, kromě svých vlastních funkcí, vykonává i PLC úlohy. Největší výhodou takového typu PLC je ten, že není potřeba mít PLC fyzické, a tím pádem se dá otestovat celý program, než se bude používat v reálných aplikacích. Tím se dá odhalit spousta chyb, které by mohly vést ke škodě.

Celý PLC program se dá najít ve Příloze 1.

Seznam použitých PLC knihoven potřebných pro programování řízení os:

- Tc2_MC.
- Tc2_NCI.
- Tc2_NcKinematicTransformation.

- Tc2_PlcInterpolation.
- Tc2_Standart.
- Tc2_Utilities.
- Tc3_Module.

Klasicky se PLC projekt skládá z POU (Program Organization Unit). POU muže být jak celý program, nebo jenom jednou funkcí. V teoretické části práce už byly popsané jazyky, pomocí kterých se dají PLC programovat. Ve své diplomové práci používám jazyk ST (Structured Text). Všechny podprogramy a funkce se volají ve hlavním bloku s názvem MAIN. Na Obr. 51 je vidět, jaké programy se v MAIN volám ve svým projektu já. Další části každého PLC programu jsou GVL (Global Variable List) – je to seznam deklarace globálních proměnných. Zvykem je do takových seznamu dávat věci, které patří k sobě a tvoří nějakou skupinu, jak je to vidět na Obr. 52 pro GVL_AxisControl.

```
1  // Axis control block
2  PRG_AXIS();
3
4  // Kinemtaic control block
5  PRG_KIN();
6
7  // Visulization
8  PRG_VISU();
```

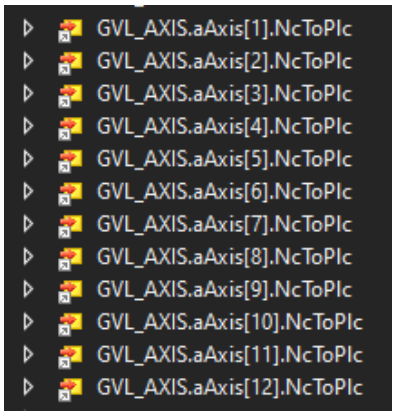
Obr. 51: MAIN smyčka PLC projektu

Scope	Name	Address	Data type	Initialization	Comment
VAR_GLOBAL	**bEnableAxis**		BOOL	FALSE	Enable vsech os
VAR_GLOBAL	**fOverride**		LREAL	25	Overridejednotlivych os
VAR_GLOBAL	**bStopAxis**		ARRAY[1..AXIS_COUNT] OF BOOL		Stop jednotlivych os
VAR_GLOBAL	**bResetAxis**		BOOL	FALSE	Resetovani vsech os
VAR_GLOBAL	**stAxisParameters**		ARRAY[1..AXIS_COUNT] OF S_MovePara		Parametry pohybu jednotlivyvh os
VAR_GLOBAL	**bAxisState**		ARRAY[1..AXIS_COUNT] OF BOOL		Stav jednotlivych os
VAR_GLOBAL	**bStartAxisMove**		ARRAY[1..AXIS_COUNT] OF BOOL		Start pohybu jednotlivych os

Obr. 52: GVL_AxisControl

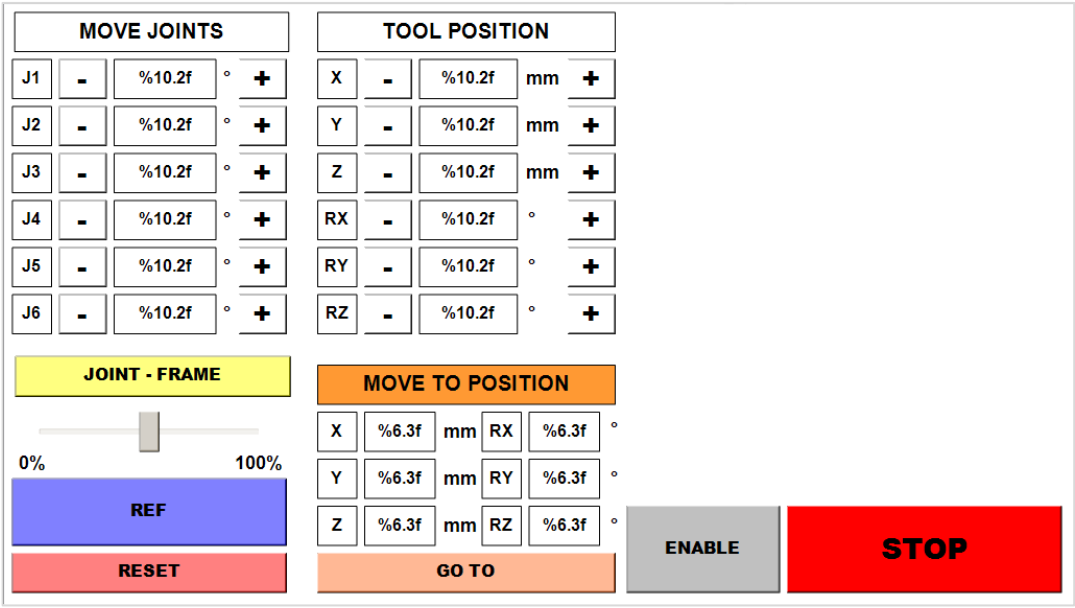
Blok *PRG_AXIS( )* odpovídá za nastavení a status jednotlivých os, vytvořených v NC úloze, *PRG_KIN( )* je pak blok, ve kterém se vytváří kinematická skupina pro transformace, *PRG_VISU( )* odpovídá za HMI (Human Machine Interface) pro jednodušší ovládání.

Pro linkování na NC osy bylo vytvořeno 12 proměnných typu *AXIS_REF*. Tyto proměnné se pak nalinkují na jednotlivé NC osy. Příklad vstupů do PLC z NC úlohy je pak vidět na Obr. 53.



Obr. 53: Vstupy os do PLC z NC úlohy

Ovládání řízení robotu je realizováno přes HMI, který stejně jako PLC jde nasimulovat při testování. Ovládací panel pro testování řízení robotu je vidět na Obr. 54.



Obr. 54: HMI panel pro řízení robotu

Tlačítkem *ENABLE* se aktivují jednotlivé osy J1 až J6, po aktivaci jsou připravené pro pohyb. Pro vrácení robotu do referenční pozice se použije tlačítko *REF*. Po zmačknutí se robot rozjede do v PLC projektu předem stanovené pozici. V části *MOVE JOINTS* jde ovládat každou osu zvlášť – pohyb ve kloubových souřadnicích. Při pohybu robota ve kloubových souřadnicích se používá dopřední kinematika. Inverzní kinematika se

používá při pohybu a programování robotu v kartézských souřadnicích X, Y, Z, která je v části *TOOL POSITION* a *MOVE TO POSITION*. Tlačítko *JOINT-FRAME* se používá pro přepínání mezi kloubovým a kartézským pohybem robotu. Tlačítko *STOP* osy deaktivuje, a tím se deaktivuje i jakákoliv aktivita os, pokud jsou v pohybu – pohyb přeruší.

5.4 Výpočet inverzní kinematiky v C++

Výpočet inverzní kinematiky v C++ byl rozdělen do dvou dílčích úkolů. Prvním úkolem bylo navrhnutou metodu, ověřenou v MATLAB, implementovat v jazyce C++. Druhým úkolem pak bylo aplikovat tento kód do real-time aplikací v prostředí TwinCAT3.

První verze kódu pro výpočet inverzní kinematiky v C++ byla navrhnuta bez uvažování omezení, které vnáší pojem real-time a prostředí TwinCAT pro programování v C++. Detailně jsou tato omezení popsaná v [21]. Pro provádění maticových operaci a pro numerické řešení inverzní kinematiky reálného robotu byly použity externí opensource knihovny. Maticové operace byly řešené pomocí knihovny Eigen a pro numerické řešení byla použita knihovna ALGlib. Výsledný kód je plně funkční a počítá dopřednou a inverzní kinematiku správně, což bylo ověřeno porovnáním výsledku C++ a MATLAB. Nevýhodou ale takového způsobu řešení je to, že tyto pojmenované knihovny nepodporují real-time, jinými slovy jsou pomalé. Výsledkem je funkční kód, který ale nejde použít pro testování a aplikace do řízení robotu ve TwinCAT3. Tuto verzi lze najít v Příloze 2.

Jelikož nešlo používat externí knihovny, byly všechny maticové operace přepsané a ověřené pomocí klasického programování bez použití externí knihoven, které by se měly pak do TwinCAT vkládat. Ve výsledku do C++ bez využití externích knihoven a s použitím pouze klasického C++ byly implementované dopředná kinematika a analytické řešení ideálního robotu. Tato druhá verze je dále aplikována do prostředí TwinCAT a je bezchybně zkompilována. Tato verze je součástí PLC projektu a Přílohy 1.

Verze 1

Analytická a numerická části inverzní kinematiky ve verzi 1 jsou udělány jako metody třídy *Robot*. Objekt třídy jako parametry očekává: DH konstanty $double\ d[\,], a[\,], alp[\,], l[\,]$ a geometrické chyby $double\ offsets[\,]$. Vytvoření objektu a příklad konstant a offsetů je vidět na Obr. 55.

```
// DH constants
double d_id[]{ 0.15, 0, 0, 0.15, 0, 0.1 };
double a_id[]{ 0, 0.3, 0, 0, 0, 0 };
double alp_id[]{ 1.57, 0, 1.57, -1.57, 1.57, 0 };
double l_id[]{ 0.15, 0.3, 0.15, 0.1 };

//Offsets
double offsets[]{ 0.000254 ,0.0001 ,0.000354 ,-0.000283 ,0.002 ,0.003 ,0.004 ,0.0052 ,0.01 ,0.012 };

//Robot class obj inicialization
Robot BCN3D(d_id, a_id, alp_id, l_id, offsets);
```

Obr. 55: Příklad inicializace objektu třídy Robot

Analytická část inverzní kinematiky se pak volá pomocí metody *.inverseKinematic_A( )*, kde jako parametr metoda očekává matici T16 (poloha v prostoru, kam se chceme dostat). Výstupem je pak pole hodnot natočení kloubu ideálního robotu, které jsou dále vstupem do numerické části.

Numerická část *.inverseKinematic_N( )*, jak už bylo řečeno, má jako parametr pole šesti hodnot natočení kloubu z části analytické a vstupní matice T16. Výstupem je pole natočení kloubu robota reálného.

Příklad volání metod je uvedený na Obr. 56.

```
//Analitic solution IK
phi_A_BCN3D = BCN3D.inverseKinematic_A(Tstart);
//Numeric solution IK
phi_N_BCN3D = BCN3D.inverseKinematic_N(Tstart, phi_A_BCN3D);
```

Obr. 56: Příklad použití metod třídy pro výpočet inverzní kinematiky

Důležitou částí numerického řešení je vygenerovat soustavu šesti nelineárních rovnic, které chceme v numerické části řešit. V kapitole č. 4 bylo detailně popsáno, jak tyto rovnice sestavit. Tyto rovnice nejsou triviální. Proto je pro jejich generování lepší použít nějaký nástroj. Pro generování rovnic v této práci byl jako nástroj použit MATLAB. Níže je popsán krátký návod na generování potřebných rovnic.

Důležité je si uvědomit – aby správně fungovala navržená metoda, musejí se rovnice generovat pro robota reálného, tzn. po přičtení offsetu.

Návod na sestavení rovnic pro numerickou metodu:

1. Sestavení modelu robotu (Obr. 32) včetně chyb (offsetu).
2. Sestavení matic T1 až T6 (Obr. 30).
3. Před matici doplnit řádek, jak je to na Obr. 57.
4. Spustit skript (viz. Příloha 3).

Výsledkem je matice T16 v symbolickém zápisu, odkud se dá pomocí jednouchého příkazu dostat na jednotlivé složky matice a doplnit výslednou soustavu rovnic.

```
syms fi1 fi2 fi3 fi4 fi5 fi6

   %**** Obecna definice transformacni matice ****

 T1 = [ cos(fi1) -sin(fi1)*cos(alp1) sin(fi1)*sin(alp1)
 T2 = [ cos(fi2) -sin(fi2)*cos(alp2) sin(fi2)*sin(alp2)
 T3 = [ cos(fi3) -sin(fi3)*cos(alp3) sin(fi3)*sin(alp3)
 T4 = [ cos(fi4) -sin(fi4)*cos(alp4) sin(fi4)*sin(alp4)
 T5 = [ cos(fi5) -sin(fi5)*cos(alp5) sin(fi5)*sin(alp5)
 T6 = [ cos(fi6) -sin(fi6)*cos(alp6) sin(fi6)*sin(alp6)

%T(1) - transofrmacni matice T01, T(2) - tm T12 atd.

T16 = T1*T2*T3*T4*T5*T6
```

Obr. 57: Část kódů pro generování matice T16 ve symbolickém tvaru

Verze 2

Ve verzi č. 2 je analytická část představená jednou funkcí *.InverseKin_A( )*. V této verzi je vstupem do analytické části matice T16, která byla vypočítaná v části dopředné kinematiky. Kromě matice budou vstupovat i DH parametry ideálního robotu. Vektor J je vstupní/výstupní vektor, do kterého se na konci výpočtu zapíšou výsledné hodnoty úhlů natočení kloubu robotu. Tato metoda je schopná řídit robota v kartézském ss, ale pouze při pohybu v osách X, Y a Z. Nejdříve je nutné se dopřednou kinematikou dostat na nějakou polohu v prostoru a pak následovně pomocí inverzní kinematiky ovládat robot.

Příklad použití funkce je na Obr. 58.

```
InverseKin_A(d, a, alp, l, T_global ,J);
```

Obr. 58: Volání funkci pro výpočet inverzní kinematiky

6 Simulační testování

Simulační testování se provádí pomocí softwarového PLC a HMI.

Pro několik bodů v prostoru byly navrhnutou metodou spočítané výsledné úhly natočení kloubu robotu, a to nejdříve v MATLAB, teprve potom v real-time systému TwinCAT. Porovnání hodnot je vidět v Tab. 2.

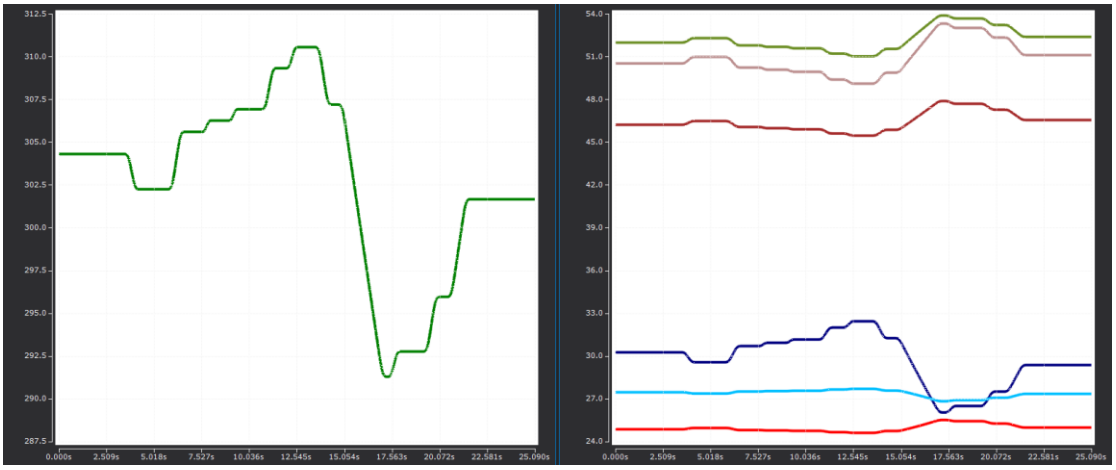
1. X = 316.94mm, Y = 270.56mm, Z = 426.82mm, Rx = 38.78°, Ry = -16.08°, Rz = 34.85°
2. X = 314.7mm, Y = 268.89mm, Z = 421.92mm, Rx = 38.78°, Ry = -16.08°, Rz = 34.85°
3. X = 310.59mm, Y = 263.65mm, Z = 416.11mm, Rx = 38.78°, Ry = -16.08°, Rz = 34.85°

Bod č.	SW	Úhly natočení kloubů(J1 až J6)					
1	MATLAB	44.73	50.12	34.86	24.46	47.7	27.92
	TwinCAT	44.7	50	34.8	24.42	47.6	27.9
2	MATLAB	44.78	50.42	31.81	23.53	50.24	29.34
	TwinCAT	44.77	50.50	31.76	23.58	49.94	29.20
3	MATLAB	44.64	51.63	26.80	22.25	53.69	31.39
	TwinCAT	44.63	51.71	26.75	22.29	53.38	31.25

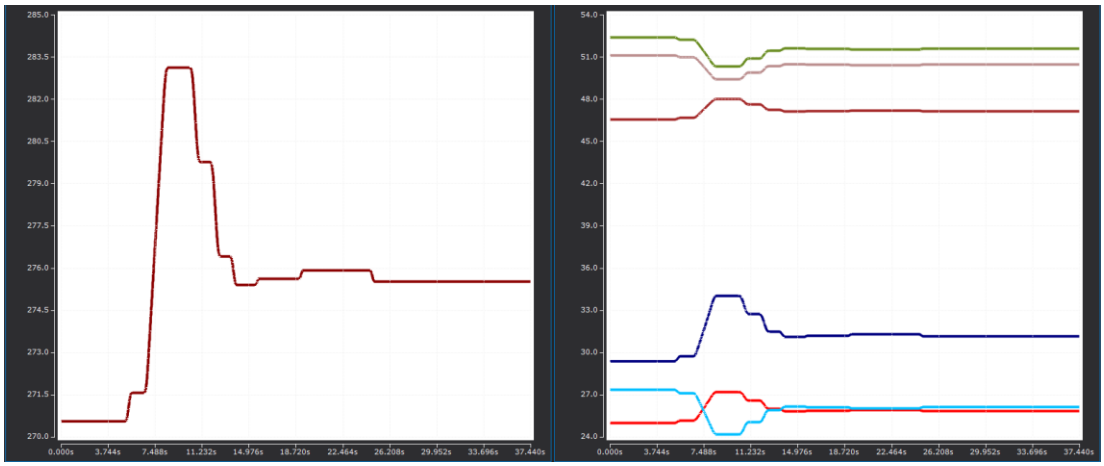
Tab. 2: Porovnání výsledků MATLAB a TwinCAT

Na obrazcích Obr. 59 až Obr. 61 jsou vidět grafy závislosti aktuální pozice jednotlivých os robotu na čase.

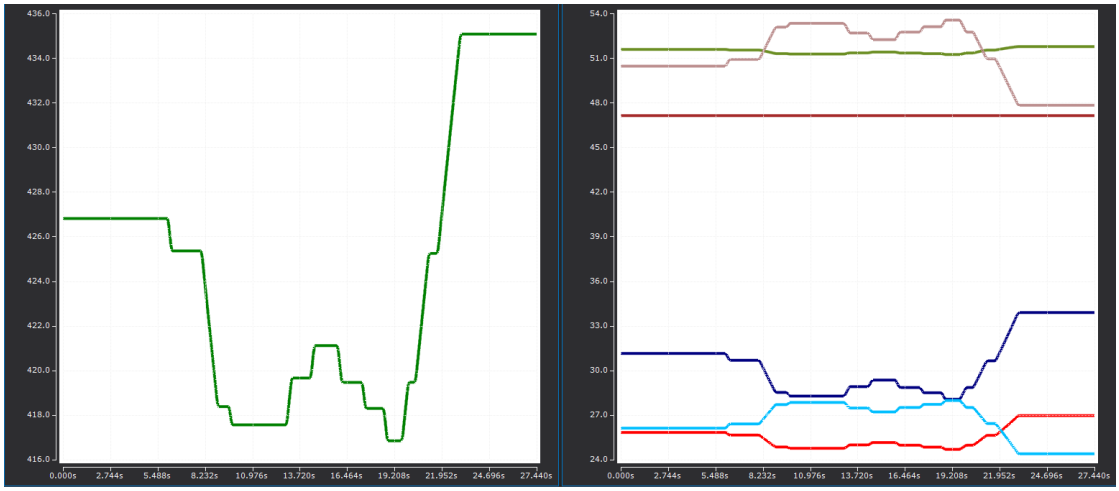
Graf na Obr. 59 znázorňuje, jak se při změně aktuální polohy souřadnice X koncového bodu robotu mění kloubové souřadnice J1 až J6. Další grafy zobrazují stejné informace ale pro osy Y a Z, kde změna aktuální polohy virtuálních os, je vždycky vlevo a kloubových os J1 až J6 vpravo. Svislá osa je vždycky aktuální poloha v [mm] u virtuálních os, u os kloubových ve stupních a vodorovná osa je čas v [s].



Obr. 59: Graf závislosti aktuální polohy osy X a os J1 až J6



Obr. 60: Graf závislosti aktuální polohy osy Y a os J1 až J6



Obr. 61:Graf závislosti aktuální polohy osy Z a os J1 až J6

7 Závěr

Cílem diplomové práce bylo vytvoření a implementace SW nástroje pro počítání inverzní kinematiky sériového šestiosého robotu, který umožní pohyb robotu v kartézském souřadnicovém systému. Práce byla rozdělena do dílčích úloh a při jejich vypracování byly použity SW MATLAB, Visual Studio a TwinCAT3.

Po provedení rešerší v části oblasti robotiky byla navržena metoda pro výpočet inverzní kinematiky pro reálného robota. V této metodě je použit jak analytický, tak i numerický způsob výpočtu. Ověření dané metody úspěšně proběhlo v SW MATLAB. Dále následovala implementace do C++. Pomocí SW Visual Studio byl kód z MATLAB úspěšně implementován do klasického C++ s použitím externích knihoven, které byly využity pro práci s maticemi a řešení numerické části inverzní kinematiky.

Do real-time aplikací v prostředí TwinCAT byly implementovány dopředná kinematika a pouze analytická část inverzní kinematiky, což znamená, že je aplikovatelná jen pro robota ideálního. Numerická část nebyla aplikována z důvodu omezení, které klade TwinCAT na C++ programování modulu. Pro dokončení implementace metody počítání inverzní kinematiky pro robota reálného je potřeba doprogramovat numerický řešič soustavy nelineárních rovnic, a to s ohledem na všechna omezení.

Simulační testování proběhlo pro implementovanou část metody a ukazuje dobré výsledky. Pro porovnání jsou v části testování představeny hodnoty z MATLAB.

Seznam použité literatury

[1] HAVEL, Ivan M. Robotika: úvod do teorie kognitivních robotů. 1. vyd.[online] Praha: SNTL, 1980. 279, [1] s. Teoretická knižnice inženýra.[cit. 1.6.2021] Dostupné z: https://digilib.phil.muni.cz/bitstream/handle/11222.digilib/106655/B_Philosophica_30-1983-1_21.pdf?sequence=1

[2] POCHYLÝ, Aleš. Ústav výrobních strojů systémů a robotiky FSi VUT v Brně. Úvod do průmyslové robotiky. [online] Brno, 2012. [cit. 1.6.2021] Dostupné z: https://docplayer.cz/5871734-Uvod-do-prumyslove-robotiky-ales-pochyly-co-je-to-robot-robotizovane-pracoviste-znalosti-v-robotice-roboticky-system.html

[3] ISO 8373:2012. *Robots and robotic devices – Vocabulary*. Technical Committee ISO/TC 184., 2012[online] [cit. 1.6.2021] Dostupné z: https://www.iso.org/obp/ui/#iso:std:iso:8373:ed-2:v1:en

[4] *ASEA Brown Boveri* [online]. ABB: © Copyright 2021 ABB [cit. 1.6.2021]. Dostupné z: https://new.abb.com/cz

[5] *Keller und Knappich Augsburg* [online]. KUKA: © KUKA AG 2021 [cit. 1.6.2021]. Dostupné z: https://www.kuka.com/cs-cz

[6] *FANUC* [online]. FANUC: [cit. 1.6.2021]. Dostupné z: https://www.fanuc.eu/cz/en

[7] *BCN3D Technologies, Inc* [online]. BCN3D: 2021. BCN3D Technologies, Inc. All Rights Reserved. [cit. 1.6.2021]. Dostupné z: https://www.bcn3d.com/

[8] *NIRYO* [online]. NIRYO: © Copyright 2016–2021 [cit. 1.6.2021]. Dostupné z: https://niryo.com/

[9] *KUKA ROBOTER GMBH. KUKA Operating and Programming Instructions: For End User*, KSS V5.2, V5.3, V5.4. 1.1. [online] Augsburg, 2006, 124 s. [cit. 1.6.2021]. Dostupné z:

http://www.wtech.com.tw/public/download/manual/kuka/krc4/KUKA%20KSS-8.3-Programming-Manual-for-SI.pdf

[10] VITRALAB: *Příručka. Automatizační a robotická technika*.[online]. Leonardo dá Vinci Programme LLP/LDV/TOI/2009/SK/93100530. [cit. 1.6.2021]. Dostupné z: http://www.sjf.tuke.sk/vitralab/files/vystupy/prirucka_cz_final.pdf

[11] *Návod k použití IRC5 s jednotkou FlexPendant RobotWare 6.0*.[online] ABB AB Robotics Products Se-721 68 Västerås Švédsko [cit. 1.6.2021]. Dostupné z: https://robotum.cz/wp-content/uploads/2019/10/IRC5sFPU.pdf

[12] *Control Engineering.* [online] Copyright © 2007-2021. [cit. 1.6.2021] Dostupné z: http://www.controlengcesko.com/hlavni-menu/artykuly/artykul/article/zaciname-s-beckhoffem/

[13] VLADIMÍR, Smutný. *Přímá a inverzní kinematika otevřených kinematických řetězců.*[online] Český institut informatiky, robotiky a kybernetiky (CIIRC), ČVUT. [cit. 1.6.2021]. Dostupné z: http://cmp.felk.cvut.cz/cmp/courses/ROB/roblec/serial-notecz.pdf

[14] MOSTÝN, V. *Mechatronika* [online]. [cit. 1.6.2021]. Dostupné z: http://robot.vsb.cz/elekskripta/mechatronika.pdf

[15] ŠVEJDA, Martin. *Kinematika manipulátorů*. [online] Technická zpráva. Katedra kybernetiky, Fakulta aplikovaných věd, Západočeská univerzita v Plzni [cit. 1.6.2021]. Dostupné z: http://home.zcu.cz/~msvejda/_publications/2011/1_kinematikaManipulatoru.pdf

[16] SPONG, M. W., HUTCHINSON, S. A., & VIDYASAGAR, M. *Robot modeling and control*. [online]. IEEE Control Systems, 26(6), 113-115. [cit. 1.6.2021]. Dostupné z: https://doi.org/10.1109/MCS.2006.252815

[17] BENDA, Josef a ČERNÁ, Růžena. *Numerická matematika – doplňkové skriptum.* [online] Praha ČVUT, 2008. ISBN 978-80-01-04037-9. [cit. 1.6.2021].

[18] ŠVEJDA, Martin. *Optimalizace robotických architektur*. [online] Plzeň, 2016. Disertační práce. Fakulta aplikovaných věd. Katedra kybernetiky. [cit. 1.6.2021].

Dostupné z: http://home.zcu.cz/~msvejda/_publications/2016/4_SvejdaMartin_thesis_2016_06_14.pdf

[19] STEJSKAL, V. a M. VALÁŠEK. Kinematics and Dynamics of Machinery. New York: Marcel Dekker, 1996. ISBN 0824797310.

[20] RAGHAVAN, M. a B. ROTH. Inverse Kinematics of the General 6R Manipulator and Related Linkages. The American Society of Mechanical Engineers, 1993, č. 115, s. 502-08. ISSN 1050-0472.

[21] Beckhoff information system [online]. Dostupné z: http://infosys.beckhoff.com

Seznam obrázků

Obr. 1: Rozdělení robotů podle zaměření [2]: a) průmyslový robot, b) servisní robot, c) humanoidní robot 9
Obr. 2: Schematické zobrazení rozdílů mezi robotem reálným (b) a ideálním (a)..... 10
Obr. 3: Paralelní robot spol. ABB[4]..... 11
Obr. 4: Robot IRB 6640 [4]..... 12
Obr. 5: Robot IRB 1520ID [4] 12
Obr. 6: Robot IRB 6620 se svařovací hlavou [4] 13
Obr. 7: Kolaborativní robot YuMi [4]..... 13
Obr. 8: Robot KR 120 R2700-2 [5]..... 14
Obr. 9: Kolaborativní robot LBR Iiwa 7 [5]..... 15
Obr. 10: Robot R-2000 iC [6]..... 16
Obr. 11: Robot CRX - 10iA/L [6] 16
Obr. 12: Robot BCN3D MOVEO [7]..... 17
Obr. 13: Robot Nyrio One [8] 18
Obr. 14: Souřadnicové systémy robota KUKA [9] 19
Obr. 15: Blokové schéma řídicího systému průmyslového robota [10] 20
Obr. 16: Řídicí systém IRC5 [11]..... 21
Obr. 17: Řídicí systém KR C4 [9] 22
Obr. 18: Řídicí systém FANUC R30iB Plus [6] 22
Obr. 19: On-line programování robotu [5] 24
Obr. 20: Tecnomatix Process Simulate[5]..... 25
Obr. 21: DH princip zavadění souřadnicových systémů [15] 29
Obr. 22: Řešitelná struktura robotu se šesti stupni volnosti [15]..... 31
Obr. 23: Zavedení souřadnicových systému dle DH úmluvy [15] 32
Obr. 24: Translační část úlohy [15] 33
Obr. 25. Schematické uspořádání planárního robotu [15]..... 34
Obr. 26: Sférické zapěsti [15]..... 36
Obr. 27: Rotace kolem základního ss robotu..... 37
Obr. 28: Interpolace matice směrových kosínů 38
Obr. 29: Příklady singulární polohy manipulátoru [13] 40
Obr. 30:Výpočet dopředné kinematiky 43

Obr. 31: Transformační matice koncového bodu šestiosého robotu 43
Obr. 32: Model simulačního robotu 44
Obr. 33: Porovnání výsledku mezi robotem reálným a ideálním 44
Obr. 34: Schéma počítání inverzní kinematiky navrženou metodou..... 45
Obr. 35: X, Y a Z souřadnice vstupní matici T16 45
Obr. 36: Pevně stanovené úhly pro generování vstupní matice T16 46
Obr. 37: Schematické zobrazení robotu pro dané natočení os 46
Obr. 38: Analytickou metodou spočítané úhly ideálního robotu..... 46
Obr. 39: X, Y a Z souřadnice reálného robotu 47
Obr. 40: Numerické řešení výsledných úhlů reálného robotu 47
Obr. 41: Informace z numerického řešiče 47
Obr. 42: X, Y a Z souřadnice transformační matice reálného robotu..... 48
Obr. 43: Schéma řízení 48
Obr. 44: Vytváření os pro řízení robotu 51
Obr. 45: Linkování osy na PLC tag..... 51
Obr. 46: Přidání Channel pro počítání kinematických transformací 52
Obr. 47: Vytváření C++ projektu 52
Obr. 48: Nakonfigurované parametry..... 53
Obr. 49: Funkce pro implementace dopředné a inverzní kinematiky..... 53
Obr. 50: Inicializace jointových a virtuálních os..... 54
Obr. 51: MAIN smyčka PLC projektu 55
Obr. 52: GVL_AxisControl..... 55
Obr. 53: Vstupy os do PLC z NC úlohy..... 56
Obr. 54: HMI panel pro řízení robotu..... 56
Obr. 55: Příklad inicializace objektu třídy Robot..... 58
Obr. 56: Příklad použití metod třídy pro výpočet inverzní kinematiky..... 58
Obr. 57: Část kódů pro generování matice T16 ve symbolickém tvaru 59
Obr. 58: Volání funkci pro výpočet inverzní kinematiky..... 59
Obr. 59: Graf závislosti aktuální polohy osy X a os J1 až J6 61
Obr. 60: Graf závislosti aktuální polohy osy Y a os J1 až J6 61
Obr. 61: Graf závislosti aktuální polohy osy Z a os J1 až J6 61

Seznam tabulek

Tab. 1: DH parametry pro zvolenou strukturu robotu 32
Tab. 2: Porovnání výsledků MATLAB a TwinCAT 60

Seznam příloh

Elektronické přílohy:

Příloha č. 1: DP_Klymoshenko.tnzp – PLC projekt a C++ modul pro kinematickou transformace

Příloha č. 2: Kinematika.zip – Verze 1 kódu v C++

Příloha č. 3: DP_MATLAB.rar – Ověření metody v MATLAB