VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA STROJNÍHO INŽENÝRSTVÍ
FACULTY OF MECHANICAL ENGINEERING

ÚSTAV MECHANIKY TĚLES, MECHATRONIKY A BIOMECHANIKY
INSTITUTE OF SOLID MECHANICS, MECHATRONICS AND BIOMECHANICS

KNIHOVNA PRO OVLÁDÁNÍ KOLABORATIVNÍHO ROBOTA UR5E Z PROSTŘEDÍ MATLAB/SIMULINK
COLLABORATIVE ROBOT UR5E INTERFACE LIBRARY FOR MATLAB/SIMULINK

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — **Adam Sladký**
AUTHOR

VEDOUCÍ PRÁCE — **Ing. Martin Brablc**
SUPERVISOR

BRNO 2021

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ FAKULTA STROJNÍHO INŽENÝRSTVÍ

Zadání bakalářské práce

Ústav:	Ústav mechaniky těles, mechatroniky a biomechaniky
Student:	**Adam Sladký**
Studijní program:	Aplikované vědy v inženýrství
Studijní obor:	Mechatronika
Vedoucí práce:	**Ing. Martin Brablc**
Akademický rok:	2020/21

Ředitel ústavu Vám v souladu se zákonem č.111/1998 o vysokých školách a se Studijním a zkušebním řádem VUT v Brně určuje následující téma bakalářské práce:

Knihovna pro ovládání kolaborativního robota UR5e z prostředí Matlab/Simulink

Stručná charakteristika problematiky úkolu:

Mechatronická laboratoř nově získala pro výzkumné a výukové účely kolaborativního robota UR5e. Výrobce poskytuje vlastní uživatelské rozhraní, ale podporuje také ovládání robota z jiných programovacích jazyků. Smyslem této práce je vyvinout knihovnu základních funkcí pro polohování robota pro Matlab/Simulink tak, aby bylo možné ho jednoduše a bezpečně použít při výuce a výzkumu.

Cíle bakalářské práce:

1) Proveďte rešerši v oblasti dostupných nástrojů a metod pro ovládání a modelování sériových manipulátorů.
2) Realizujte knihovnu funkcí pro Matlab, která umožní pokročilému uživateli ovládání robota z tohoto programovacího jazyka. Bude tedy obsahovat základní funkce pro navádění polohy koncového bodu robotického ramena i pro ovládání dostupných gripperů. Knihovna bude obsahovat také simulátor manipulátoru.
3) Na základě knihovny realizované v předchozím bodě vytvořte její studentskou verzi, která zajistí bezpečnost uživatele i robota a umožní její jednoduché použití ve výuce. Předpokládá se omezení rychlosti pohybu i pracovního prostoru robota.
4) S využitím studentské verze knihovny realizujte demonstrační úlohy kopírující zadání používaná při výuce kinematiky sériových manipulátorů.

Seznam doporučené literatury:

MURRAY, Richard M. A Mathematical Introduction to Robotic Manipulation, CRC Press, 1994, ISBN 9780849379819.

NELLES, Oliver. Nonlinear system identification: from classical approaches to neural networks and fuzzy models. Berlin: Springer, 2011. ISBN 978-364-2086-748.

LJUNG, Lennart. System identification: theory for the user. 2nd ed. Upper Saddle River, NJ: Prentice Hall PTR, 1999. ISBN 978-0136566953.

VALÁŠEK, Michael. Mechatronika. Dot. 1. vyd. Praha: České vysoké učení technické, 1996. ISBN 80-010-1276-X.

NOSKIEVIČ, Petr. Modelování a identifikace systémů. Ostrava: Montanex, 1999. ISBN 80-722-50-0-2.

Termín odevzdání bakalářské práce je stanoven časovým plánem akademického roku 2020/21

V Brně, dne

L. S.

prof. Ing. Jindřich Petruška, CSc. ředitel ústavu	doc. Ing. Jaroslav Katolický, Ph.D. děkan fakulty

Abstrakt

Tato práce se zabývá postupem vytvoření knihovny funkcí, která umožní snadné ovládání robota UR5e a jeho interaktivního modelu z prostředí Matlab. Na její tvorbu byla použita metoda objektového programování. Funkce knihovny byly navrženy odděleně pro módy ovládání robota a modelu. Součástí jsou i funkce pro ovládání gripperu RG2 a detekci kolize zabraňující srážce robota se sebou samým. Výsledkem je knihovna navržená tak, že je možné robota snadno využít při měřeních, práci na projektech i jako pomůcku ve výuce pouhým zařazením jeho funkcí do existujícího skriptu zpracovávajícího data nebo provádějícího jiné výpočty. Na závěr jsou uvedeny tři výukové úlohy představující studentská zadání, na kterých je ukázáno použití knihovny.

Abstract

This thesis describes the process of creating a function library which will enable easy control of a UR5e robot and its model using the Matlab environment. It was made using object based programming. Library functions were designed for two separate modes, each controlling either the robot or the model. Functions for RG2 gripper control and for collision detection, which prevents the robot from colliding with itself, are also part of the library. The resulting library is designed in a way that allows the robot to be easily used for measuring, project work or as a teaching tool by simply adding its functions to already existing data processing scripts or those performing other calculations. Finally, three example tasks are presented to demonstrate the application of this library.

Klíčová slova

Matlab, knihovna funkcí, UR5e, RG2, Robotics System Toolbox, kolaborativní robot

Keywords

Matlab, function library, UR5e, RG2, Robotics System Toolbox, collaborative robot

Bibliografická citace

SLADKÝ, Adam. *Knihovna pro ovládání kolaborativního robota UR5e z prostředí Matlab/Simulink*. Brno, 2021. Bakalářská práce. Vysoké učení technické v Brně, Fakulta strojního inženýrství, Ústav mechaniky těles, mechatroniky a biomechaniky. Vedoucí práce Ing. Martin Brablc.

Čestné prohlášení

Prohlašuji, že jsem bakalářskou práci na téma „Knihovna pro ovládání kolaborativního robota UR5e z prostředí Matlab/Simulink“ vypracoval samostatně s použitím odborné literatury a pramenů uvedených v seznamu, který tvoří přílohu této práce.

21. května 2021

.....

Adam Sladký

Poděkování

Tímto děkuji panu Ing. Martinu Brablcovi za cenné připomínky a rady při vypracování této bakalářské práce.

Obsah:

1 Úvod

Robotizace pracovních úkonů není v dnešní době ničím neobvyklým v mnoha odvětvích průmyslu. Spolehlivost a rychlost práce spojená s neomezenou pracovní dobou již není doménou pouze automobilek a jiných velkých továren, ale stává se dostupným řešením i pro malé firmy. Velkou měrou k tomu přispívají kolaborativní roboti, kteří jsou v posledních dvou dekádách na vzestupu a těší se velké oblibě. Nejčastěji nahrazují lidské pracovníky v úkonech, které nejsou náročné na sílu, ale jsou velmi repetitivní. Příkladem je odebírání výrobků z pásů, skládání na palety nebo balení do krabic. U člověka mohou vzniknout častým opakováním stejného pohybu dlouhodobé následky, zatímco robot může pracovat celý den nepřetržitě v podmínkách jako je neklimatizovaná místnost a s pominutím pořizovací ceny i za nižší náklady. Proto je velmi často využíván ve firmách, které si nemohou dovolit velké průmyslové roboty, ale chtějí rozšířit automatizaci výroby.

Kromě komerčního sektoru bývají také stále častěji použiti jako asistenti v laboratořích, například pro 3D skeny objektů, jako polohovatelné držáky nebo při manipulaci vyžadující velkou přesnost nebo přesnou trasu pohybu při každém opakování. Rostoucí zájem firem o absolventy se znalostmi robotiky z nich dělají také výbornou pomůcku pro výuku, například kinematiky nebo programování. Možnost názorně ukázat určení parametrů robotů nebo vyzkoušet programování jednoduchého manipulátoru je pro studenty přínosným rozšířením teoretické výuky.

Většina kolaborativních robotů je dodávána s vlastním uživatelským prostředím. Inženýrské i studentské projekty a experimenty ale často používají jako nástroj k programování Matlab a využívat dvě navzájem nepropojená prostředí je nepraktické. Bylo tedy potřeba vytvořit knihovnu umožňující integraci ovládání robota přímo do jeho prostředí. Aby bylo použití bezpečné pro studenty, součástí musí být také verze s bezpečnostními omezeními. I za tímto účelem je důležité mít k dispozici model robota umožňující testování programů.

Na následujících stranách budou nejdříve představeni kolaborativní roboti zastoupení robotem UR5e od Universal Robots s připojitelným gripperem OnRobot RG2 a toolbox umožňující tvorbu modelů. V navazující praktické části bude jako první popsán postup tvorby modelu robota UR5e od vytvoření základní struktury až po konečnou vizualizaci včetně gripperu. Následovat bude vysvětlení animace pohybu modelu a způsob ověřování kolize. Dále bude představena struktura řídícího objektu knihovny, jeho parametry a popis jednotlivých funkcí spolu s krátkou ukázkou programování.

V závěru práce budou jako příklad použití knihovny vyřešeny tři ukázkové úlohy navržené jako zadání úkolů pro studenty.

2 Rešerše

Přestože roboti jsou využívaní již několik desítek let a někteří z nich se už dávno dostali na Měsíc, jejich menší a bezpečnější verze, nazvané podle spolupracování s lidmi kolaborativní, jsou vynálezem starým jen asi čtvrt století. Jejich univerzální použití, kompatibilita s širokou řadou nástrojů a snadné zařazení do výrobního procesu z nich dělá atraktivní možnost automatizace průmyslu. V této práci budou zastoupeni modelem UR5e od Universal Robots s připojeným gripperem OnRobot RG2.

Jelikož programování není snadno představitelná věc, je dobré mít k dispozici i model robota, k čemuž slouží například rozšíření Matlabu nazvané Robotics System Toolbox.

2.1 Kolaborativní roboti

Kolaborativní robot, také zvaný kobot, je robot navržený pro operaci v blízkosti nebo přímo v prostoru sdíleném s člověkem. To je rozdíl oproti tradičním robotům, kteří bývají odděleni od lidských pracovníků klecemi, zábranami a jinými bezpečnostními prostředky. Podle Mezinárodní federace robotiky (IFR) existuje pět úrovní spolupráce mezi robotem a člověkem:

- Buňka – pracovní prostory jsou odděleny zábranou
- Koexistence – oddělené pracovní prostory, ale bez zábrany
- Sekvenční kolaborace – sdílený pracovní prostor a sekvenční spolupráce na úkolu
- Kooperace – současná práce na úloze
- Responsivní kolaborace – robot je schopen v reálném čase reagovat na člověka

V současnosti je nejčastějším případem koexistence nebo sekvenční kolaborace. [1]

2.1.1 Historie

V roce 1995 firma General Motors vypsala grant na výzkum způsobů, jak umožnit bezpečnou spolupráci robotů a lidí při výrobě automobilů. O rok později tak profesoři Northwesternské univerzity J. Edward Colgate a Michael Peshkin představili prvního kobota. Ten zaručoval bezpečnost tím, že sám náklad pouze podpíral a veškerý pohyb vykonával člověk. Počítač ale řídil instalaci součástí řízením směru pohybu tak, aby se minimalizovaly chyby. Colgate a Peshkin si v roce 1997 svého kobota patentovali a založili firmu Cobotics, která vyráběla roboty patřící do kategorie responsivní kolaborace pro použití v automobilovém průmyslu. Ta působila až do roku 2003, kdy byla odkoupena společností Stanley Assembly Technologies. Úspěchu kolaborativních robotů si brzy všimly další firmy, které je vyrábí dodnes, mezi nimi KUKA, FANUC, ABB nebo Universal Robots. [2]

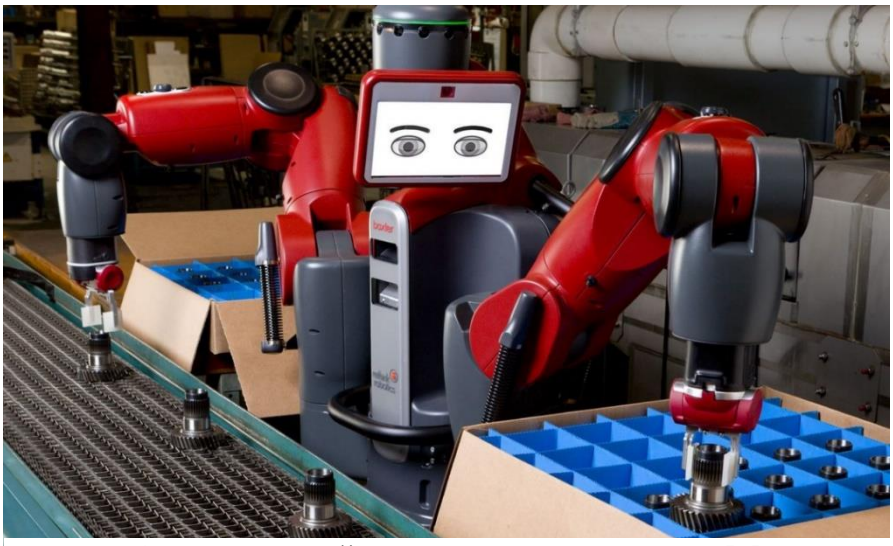
2.1.2 Výhody a nevýhody kolaborativních robotů

Primární výhodou je možnost spolupráce s člověkem bez dalších bezpečnostních opatření. Koboty mohou být vybaveny senzory, které v přítomnosti osob robota zpomalí nebo zastaví, dokud se pracovní prostor neuvolní. Další ochranou je použití motorů se snímáním momentu. To znamená, že pokud překročí vypočtenou hodnotu, robot se preventivně zablokuje, čímž dojde k omezení následků případného nárazu. Také mívají zaoblené hrany a jsou vyrobeny z lehkých materiálů. To umožňuje snadné přesouvání mezi pracovišti, což je další velkou výhodou hlavně pro malé firmy, které si nemohou dovolit výrobu plně automatizovat velkými průmyslovými roboty. Kobot ale může být snadno zařazen přímo do výrobního procesu, a to jen na malé části pracoviště sdílené s lidmi. V případě potřeby také může snadno změnit vykonávaný úkol díky jednoduchému softwarovému prostředí. Při programování nám pomáhá možnost vytvořit program v učícím módu. V něm se robot ručně provede požadovanými úkony, které si zapamatuje a zopakuje. Pro samotné pracovníky pak použití robotů znamená ulehčení od repetitivních úkolů, jako je přemisťování dílů nebo kontrola jakosti. Je prokázáno, že člověk nedokáže dělat dané úkoly stejně pečlivě po celou pracovní dobu.

Nevýhodou kobotů oproti klasickým robotům je jejich nižší nosnost a maximální síla, jakou může robot působit. Tyto problémy jsou důsledkem bezpečnostních omezení pro spolupráci kobotů s lidmi. Kobot také nemůže vykonávat takové úlohy, u kterých se vyžaduje absolutní přesnost nebo vysoká rychlost, a pracovat s netříděnými nebo nepravidelnými díly. [3]

2.1.3 Příklady použití kolaborativních robotů

Typickou aplikací robotů jsou úlohy typu Pick and Place, tedy přemisťování objektů z jednoho místa na jiné. Jedná se hlavně o balení, skládání, nakládání a odebírání malých výrobků z pásů, K daným činnostem je často nutné i použití vizuálních senzorů k synchronizaci s pásem. Kvůli opakování stejného pohybu dochází u lidí často ke zdravotním problémům, jednotvárnost úkonů pak také snižuje jejich pozornost při práci, což vede často k chybám. Tomu ale lze jednoduše předejít, proto bývají tyto úlohy mezi prvními, které jsou ve firmách automatizovány.



Obrázek 2.1: Příklad Pick and Place úlohy [4]

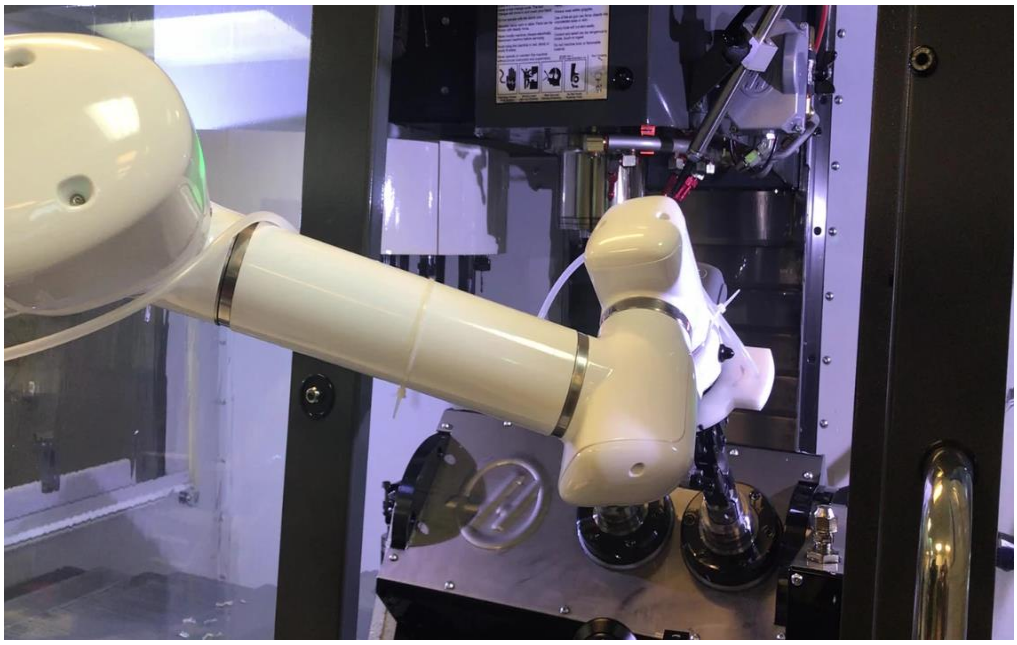
Dalším častým využitím kobotů je kontrole kvality. Jako nástavec jsou pro tento typ činnosti použity kamery s vysokým rozlišením, pomocí kterých se porovnává dokončený výrobek s počítačovým modelem. Protokol o kontrole také může být snadno uložen pro pozdější revizi nebo dohledání konkrétních kusů při reklamacích. Kontrola kvality pomocí robotů bývá přesnější než při kontrole lidmi, což vede k vyšší kvalitě produkce.

Obrázek 2.2: Kamerový nástavec [5]

Třetím nejčastějším použitím jsou výrobní procesy, jako je svařování, pájení nebo lepení. Zaškolení pracovníka, který by dokázal například opakovaně stejně dokonale nanášet lepidlo, by trvalo v lepším případě týdny. Kobot může být snadno naprogramován pomocí ručního navádění a program pak může být zkopírován na ostatní jednotky vykonávající stejný úkol. Dráha pohybu i jeho výsledek bude díky tomu pokaždé stejný, ať už jde o množství lepidla nebo tvar svaru.

Obrázek 2.3: Kobot aplikující lepidlo [6]

Dále mohou být tito roboti využiti k obsluze obráběcích strojů, 3D tiskáren nebo vstřikovacích zařízení, kde slouží například k vyměňování nástrojů, doplňování materiálu nebo odebírání výrobků. Také uvolní lidskou obsluhu, která by jinak strávila dohromady několik hodin pouze čekáním na práci stroje. Lidská práce tak může být využita pro náročnější úkoly.



Obrázek 2.4: Obsluha CNC pomocí kobota [7]

Poslední možnou aplikací kobotů jsou dokončovací operace jako je broušení nebo leštění. Díky snímačům se dokáže robot po povrchu výrobku pohybovat se stálou rychlostí a přítlakem, takže lze docílit konzistentní drsnosti po celém povrchu, včetně zaoblených nebo ostrých hran.

Obrázek 2.5: Broušení pomocí kobota [8]

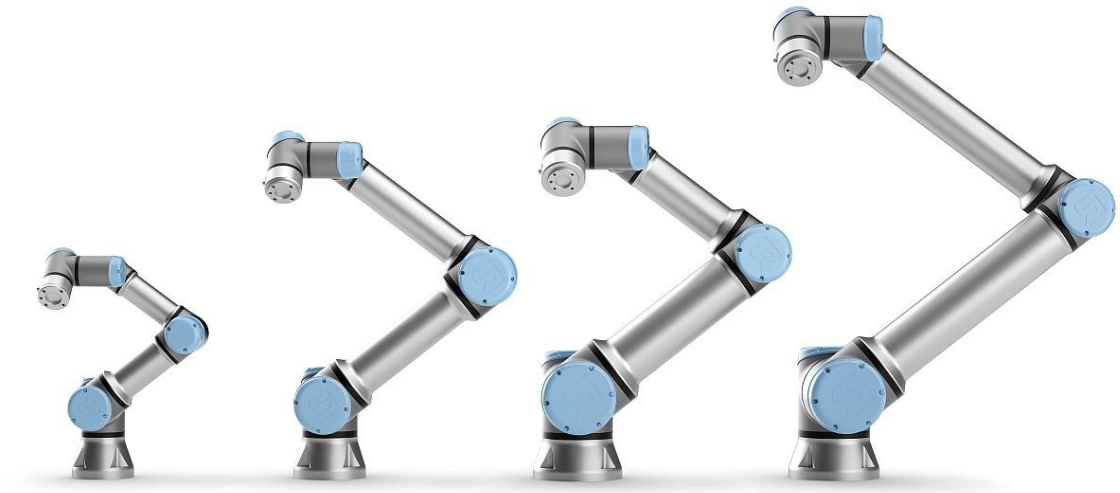
2.2 Universal Robots UR5e

Firma Universal Robots je jeden z největších výrobců kolaborativních robotů na světě. Jejich roboti řady e-Series jsou založeni na myšlence poskytnout dostupnou automatizaci i pro nejmenší podnikatele. Hlavními přednostmi jsou snadná instalace díky lehké a kompaktní konstrukci, bezproblémové programování a rozsáhlé možnosti přizpůsobení díky kompatibilitě s nástavci různých výrobců, jmenovitě např. OnRobot nebo Robotiq. V nabídce jsou čtyři verze odlišující se velikostí a maximálním nákladem, podle kterého jsou označeni.

Model UR5e je střední cesta nabízející kompromis mezi velikostí a nosností. Jedná se o robota se šesti klouby, třemi na rameni a třemi pro rotaci nástroje. Robot tedy disponuje šesti stupni volnosti. Základní technické údaje jsou uvedeny v tabulce 1: [9]

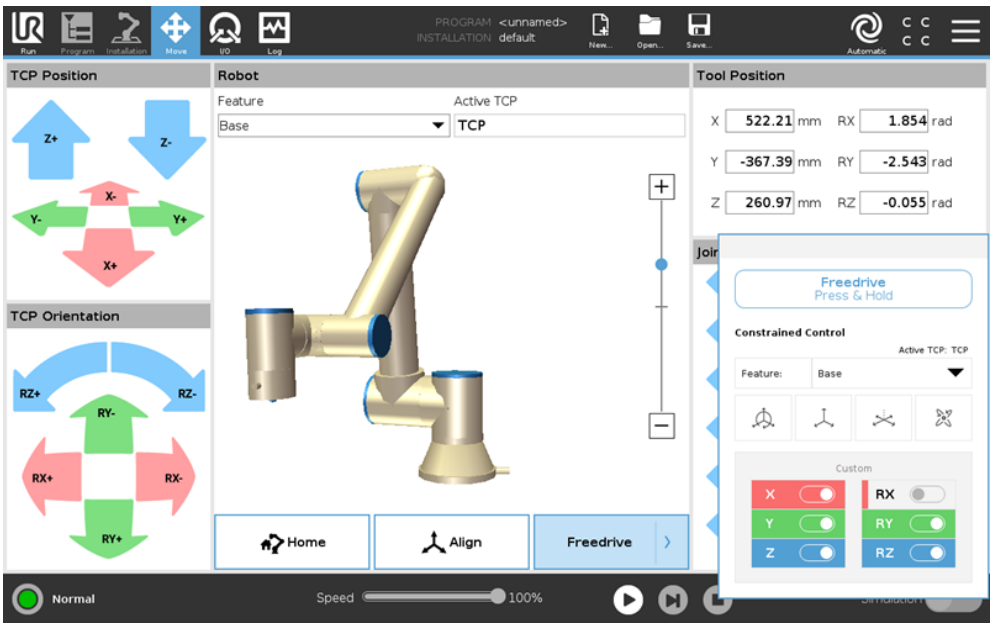
Tabulka 1: Specifikace UR5e

Maximální dosah	850 mm
Nosnost	5 kg
Rozsah otočení kloubů	± 360°
Rozsah snímače momentu	0-10 Nm
Přesnost snímače momentu	0,30 Nm
Maximální rychlost pohybu nástroje	1 m/s
Přesnost pohybu nástroje	0,03 mm

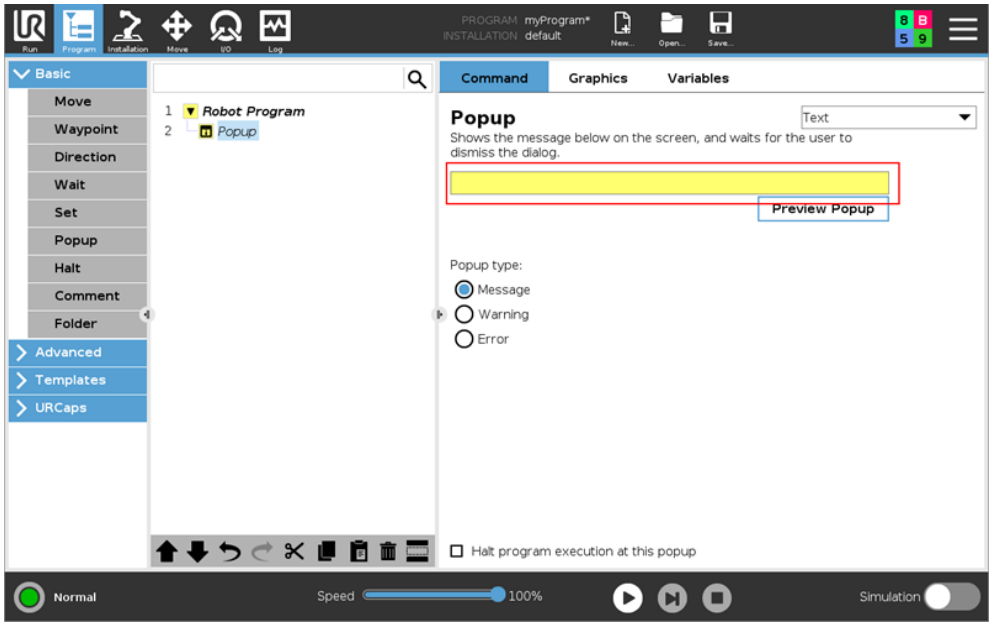


Obrázek 2.6: UR5e (druhý zleva) a ostatní modely e-Series [10]

Ke každému z robotů patří také Teach Pendant, tedy ovládací panel sloužící k programování robota skrze přehledné uživatelské rozhraní. Dané rozhraní nabízí tři módy: automatický, ve kterém lze pouze spouštět lokálně uložené programy, dálkové ovládání, ve kterém jsou příkazy posílány skrz datový kabel, a manuální, přístupný s heslem, ve kterém lze upravovat nastavení robota nebo vytvářet nové programy. Dále je možné pomocí panelu sledovat v reálném čase polohu koncového bodu a jednotlivých kloubů robota, manipulovat s nimi nebo aktivovat Freedrive. Koncepce Freedrive umožňuje pohybovat robotem ručně, k čemuž pak slouží i tlačítko na zádech panelu. Pokud by v programu nastala chyba, obsahuje také protokol přijatých příkazů a změn v nastavení.



Obrázek 2.7: Uživatelské rozhraní ovládání UR5e [11]



Obrázek 2.8: Uživatelské rozhraní programování UR5e [12]

2.2.1 Komunikace s PC

Komunikace robota s počítačem probíhá pomocí protokolu TCP/IP a připojení přes ethernetový kabel, při kterém je PC nastaven jako server a robot jako klient. Rozhraní UART poskytuje několik portů, z nichž jsou v knihovně využity tyto:

- Port 29999 – Dashboard Server, slouží ke spouštění programů, zapínání a odbrzdění, obnovovací frekvence 10 Hz
- Port 30001 – Primary Interface, slouží k monitorování stavu robota, obnovovací frekvence 10 Hz
- Port 30003 – Real-time, také zvaný Matlab Interface, slouží k ovládání robota přímými příkazy bez nutnosti programu nahraného v paměti, proto vyšší obnovovací frekvence 500 Hz

Příkazy se z Matlabu odesílají pomocí funkce *fprintf*, musejí být ve formátu string, obsahovat příkaz jazyka URScript v syntaxi podle manuálu výrobce a být ukončené znakem \n. Příklad odeslání příkazu k zastavení robota by tedy vypadal takto:

```
port30003 = tcpip(10.0.0.2, 30003);          % vytvoření komunikace
fopen(port30003);                            % otevření portu
fprintf(port30003, 'stopl(0.05)\n');         % v závorce brzdné zrychlení
```

Základní skripty umožňující komunikaci s Matlabem byly převzaty z GitHubu. [13]

2.3 OnRobot RG2 – gripper

OnRobot je jednou z firem vyrábějících nástroje kompatibilní s roboty UR5e. Mezi ně patří například nástavce pro šroubování, broušení nebo vakuové a prstové grippery (uchopovače). Model RG2 je základním dvouprstým gripperem. Prsty jsou pro lepší úchop pogumované. S robotem je gripper spojen pomocí kabelu, který slouží k přenosu signálů, ale i pro napájení. Kabel je také připojen ke konektoru na nástavci, jež s gripperem přímo otáčí, takže nemůže dojít k zamotání a přetržení kabelu. K ovládání uchopovače slouží funkce nahrané do paměti ovládacího panelu, ty jsou specifické pro jednotlivé grippery. Ovládací funkce se spouštějí skrze software URCap, který je dodávaný spolu s gripperem. URCap se spouští jako podprogram uživatelského rozhraní robota, do kterého se při každém spuštění musí načíst. Protože se při dálkovém ovládání z Matlabu uživatelské rozhraní nepoužívá, nelze tyto funkce použít stejným způsobem jako nativní funkce robota. Základní technické údaje RG2 jsou v tabulce 2: [14]

Tabulka 2: Základní technické údaje gripperu RG2

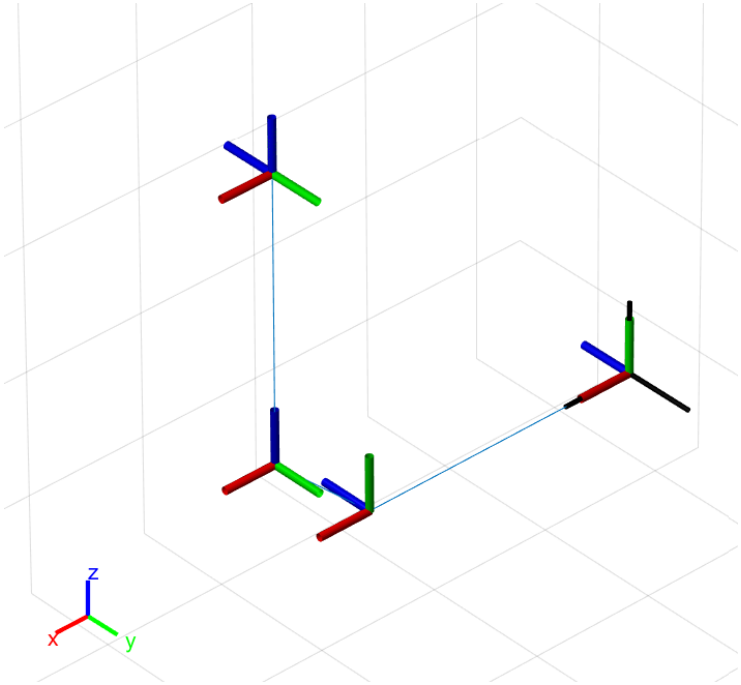
Maximální náklad rovnoběžně s prsty	2 kg
Maximální náklad kolmo na prsty	5 kg
Maximální rozevření prstů	110 mm
Přesnost pohybu prstů	0.1 mm
Maximální síla stisku	40 N

Obrázek 2.9: Gripper OnRobot RG2 [15]

2.4 Robotics System Toolbox

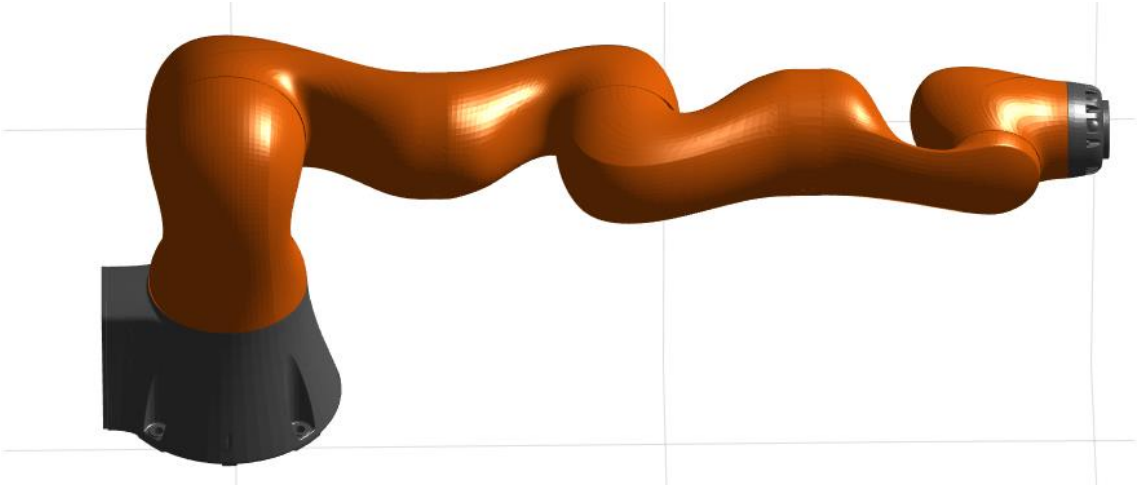
Pro vytváření modelů robotických systémů v prostředí Matlabu se využívá Robotics System Toolbox, vyvinutý panem Peterem Corkem. Tento systém obsahuje nástroje a algoritmy pro navrhování, simulaci a testování robotických ramen a mobilních nebo humanoidních robotů. Pro ukázku jsou jeho součástí modely několika komerčních robotů. Také lze načíst externí modely ve formátu URDF (Unified Robot Description Format).

Tvorba modelu probíhá tak, že se pro každou část robota vytvoří objekt typu Rigid Body, tedy tuhé těleso, kterému se přiřadí kloub. Ten může být buď fixní, který slouží pouze k propojování, nebo pohyblivý. Pohybovat se pak kloub může buď lineárně, nebo rotačně. Typ a počet pohyblivých kloubů udávají stupně volnosti konečného modelu. Jednotlivé části se pak spojí do jednoho objektu Rigid Body Tree, tedy stromu tuhých těles, tak, že první část se pevně připojí k výchozímu bodu představujícímu svět a ostatní se postupně připojují jedna na druhou, takže každá část je ve stromě pro předchozí část Child a pro navazující Parent. Pro návaznosti jednotlivých souřadnicových systémů se používají takzvané Denavit-Hartenbergovy parametry, což jsou čtyři údaje popisující posuv a natočení lokálního souřadnicového systému vůči jinému, v tomto případě vůči lokálnímu systému předchozí části.

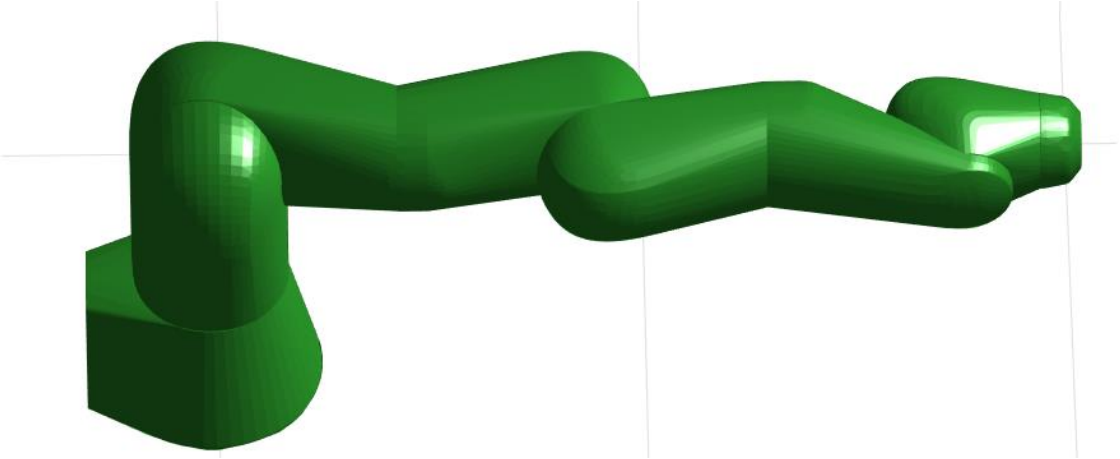


Obrázek 2.10: Rigid Body Tree model robota Puma 1

Pokud jsou k dispozici, lze ke každé části přiřadit 3D CAD model ve formátu STL. Ten může sloužit buď pro vizualizaci, nebo jako kolizní objekt, popřípadě obojí. Pro kolizi lze také použít primitivní objekty, jako jsou válce nebo krychle vygenerované pomocí k tomu určených funkcí. Po přiřazení může však být nutné upravit orientaci modelu vůči souřadnicovému systému. To provádíme pomocí transformační matice.



Obrázek 2.11: Vizualizovaný 3D model KUKA LBR iiwa 14



Obrázek 2.12: Kolizní model KUKA LBR iiwa 14

Pro práci s hotovým modelem pak toolbox obsahuje funkce pro počítání dopředné a inverzní kinematiky, vytváření trajektorie a nově také výpočty vzdáleností jednotlivých částí a detekci případné kolize.

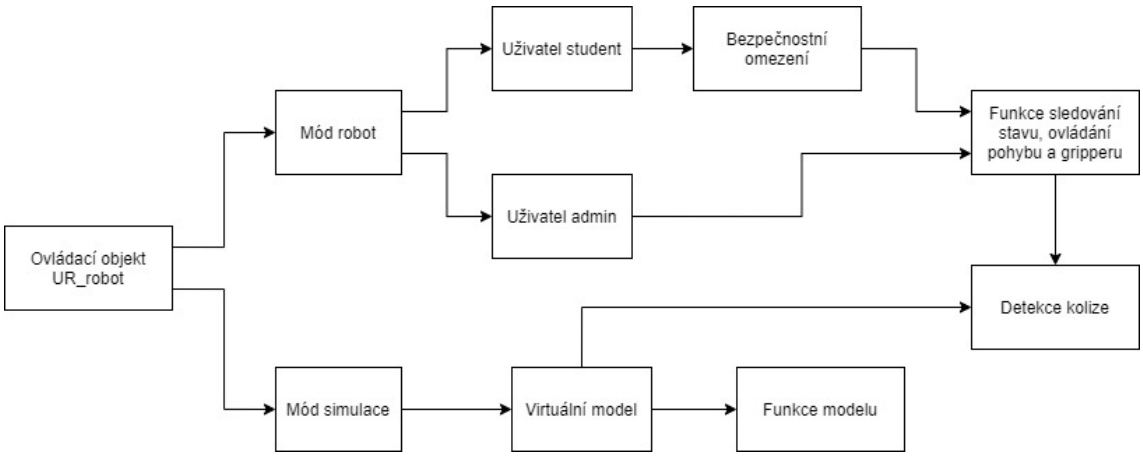
3 Knihovna

Ovládání robota by mělo být snadné a intuitivní, aby mohl být rychle naprogramován a použit v mnoha různých aplikacích, a zároveň integrované do Matlabu. Z tohoto důvodu bylo zvoleno programování pomocí objektu. Ten má přehledně rozdělené a snadno přístupné parametry, určující například rychlost či zrychlení, a jasně dané metody spustitelné přes tečkovou syntaxi, které lze v případě potřeby rozšířit o nové. Funkce *set* a *get* navíc umožňují snadné omezení parametrů a konstrukční funkce snadnou inicializaci.

Knihovna byla vystavěna tak, aby měla dvě hlavní části. Každá z nich patří k jednomu z nastavitelných módů lišících se použitím.

Do první patří funkce umožňující ovládání reálného robota pomocí komunikace s počítačem. Zahrnuje tedy funkce zadávající příkazy k pohybu nebo ovládání gripperu. Stejně tak do ní patří funkce sloužící k zapínání nebo odblokování a ke sledování aktuálního stavu robota. Tato část je dále rozdělena podle uživatele na administrátorskou a studentskou verzi. V případě použití knihovny jako student jsou nastavena omezení sloužící k zajištění bezpečnosti při použití ve výuce.

Mód simulace obsahuje virtuální model robota vytvořený v prostředí Matlabu, ke kterému patří funkce kopírující ty obsažené v první části. Hlavním účelem modelu je testování kódu, knihovna byla tedy navržena tak, aby bylo možné volně přepínat mezi módy při zachování stejného kódu. Model samotný je také využíván funkcí detekující kolizi při ovládání reálného robota.



Obrázek 3.1: Struktura knihovny

3.1 Matlab model

Celý model se skládá ze dvou hlavních částí, z robota a gripperu. Částmi robota jsou základna, širší a užší rameno a tří díly zápěstí. První tři primárně ovládají pohyb koncového bodu v prostoru, zatímco zápěstí určuje orientaci nástroje. Gripper je pak možné připojit k ramenu ve třech různých polohách, kdy každá poloha má vlastní model.

Pro rozpohybování modelu se používá řešič inverzní kinematiky. Pomocí něj lze pak vytvořit funkce zobrazující animace napodobující pohyb reálného robota. Ty bylo nutné napsat tak, aby generovaly trajektorie koncového bodu, a to v kloubových i kartézských souřadnicích.

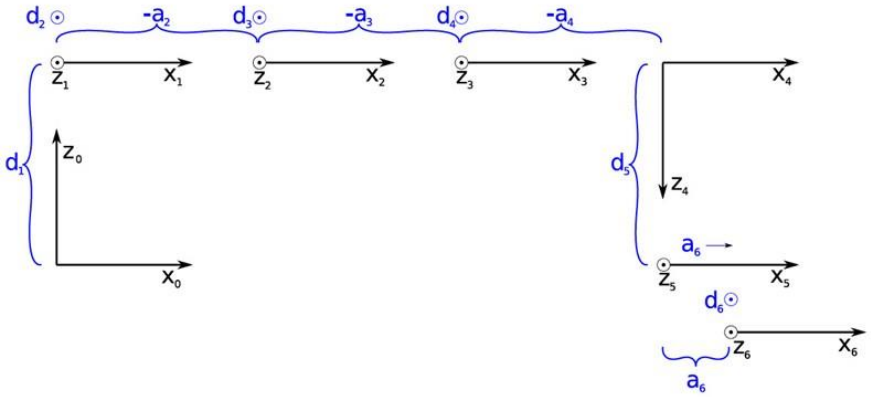
Jelikož robot sám o sobě nedokáže předpovídat srážku a časté nárazy by snižovaly jeho životnost, bylo pomocí modelu vytvořeno řešení funkce detekce kolize. To by mělo srážce zabránit včasným zastavením robota v případě kritického přiblížení jeho částí.

3.1.1 Základní model UR5e

Při tvorbě modelu robota UR5e byly použity Denavit-Hartenbergovy parametry vypsané v tabulce 3 dostupné od výrobce, kde *a* je posuv podél osy x, *d* podél osy z, θ je rotace kolem osy z a α rotace kolem osy x. [16]

Tabulka 3: Denavit-Hartenbergovy parametry UR5e

Číslo kloubu	θ [rad]	*a* [m]	*d* [m]	α [rad]
1	0	0	0,1625	$\pi/2$
2	0	-0,425	0	0
3	0	-0,3922	0	0
4	0	0	0,1333	$\pi/2$
5	0	0	0,0997	$-\pi/2$
6	0	0	0,0996	0



Obrázek 3.2: Schéma souřadnicových systémů UR5e

Nejdříve byla vytvořena Rigid Body Tree struktura se základnou, ke které byla přiřazena deska jako kolizní objekt. K té byly následně pomocí D-H parametrů postupně připojeny souřadnicové systémy všech částí robota, které byly doplněny o vizuální a kolizní modely stažené z GitLabu. [17]

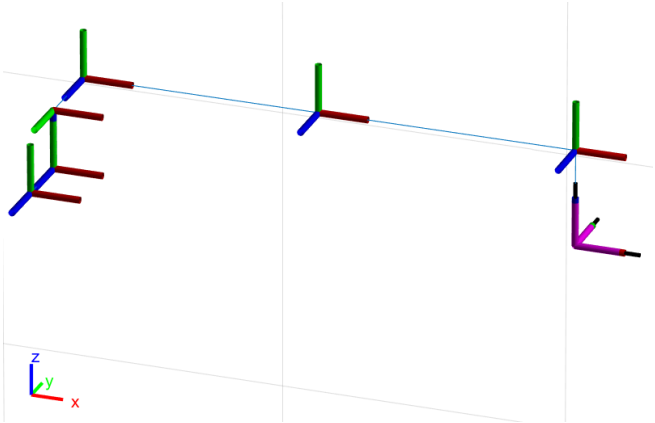
Ukázka vytvoření a připojení ramene robota k základně:

```
shoulder_link = rigidBody('shoulder_link');
shoulder_pan_joint = rigidBodyJoint('shoulder_pan_joint', 'revolute');
setFixedTransform(shoulder_pan_joint, [0, pi/2, 0.1625, 0],  'dh');
shoulder_link.Joint = shoulder_pan_joint;
tf_mesh_shoulder = [1 0 0 0; 0 0 1 0; 0 1 0 0; 0 0 0 1];
tf_mesh_shoulder_col = [1 0 0 0; 0 0 1 0.002; 0 1 0 0; 0 0 0 1];
addVisual(shoulder_link,"Mesh","shoulder_vis.stl",tf_mesh_shoulder)
addCollision(shoulder_link,"Mesh","shoulder_col.stl",tf_mesh_shoulder_col);
addBody(robot, shoulder_link, 'base');
```

Struktura celého modelu vypadá takto:

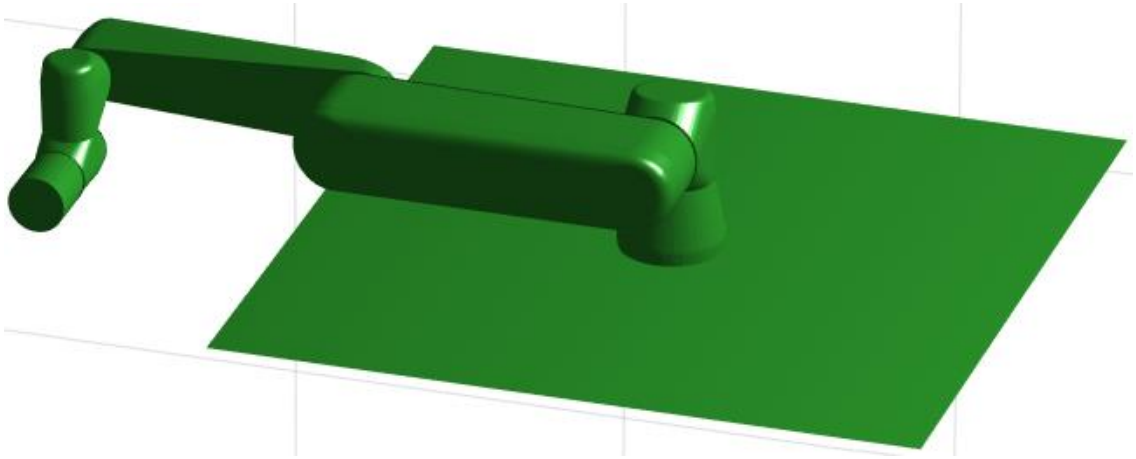
Tabulka 4: Struktura modelu UR5e

Název	world	base	shoulder	upper arm	forearm	wrist1	wrist2	wrist3
Kloub č.			1	2	3	4	5	6
Natočení			±2π	±2π	±2π	±2π	±2π	±2π



Obrázek 3.3: Rigid Body Tree model robota UR5e

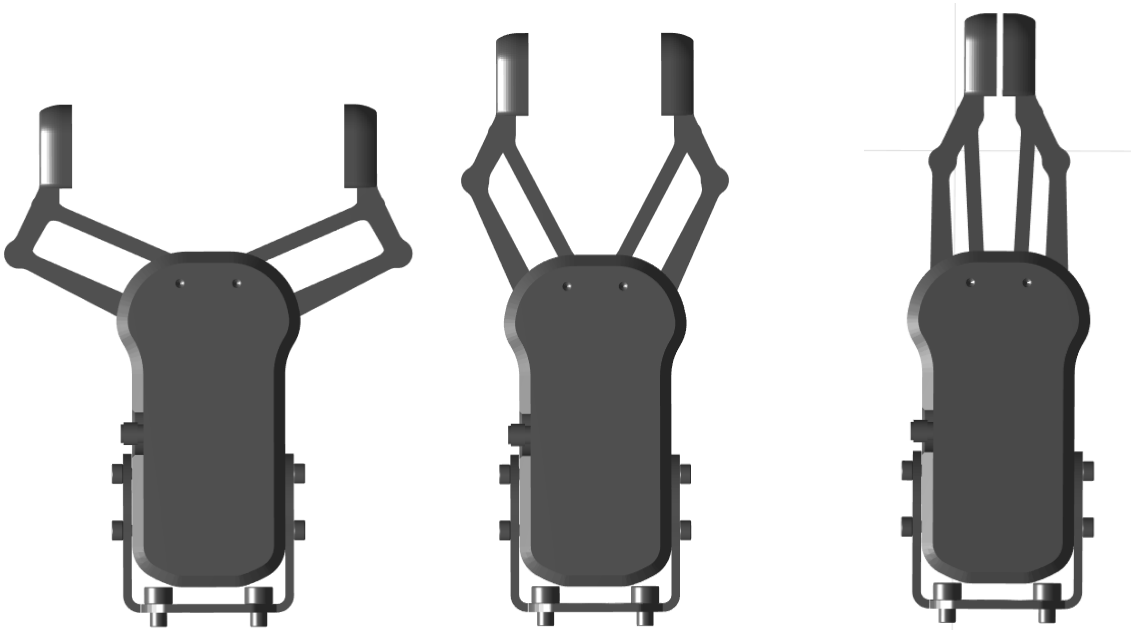
Obrázek 3.4: Vizualizovaný model model robota UR5e



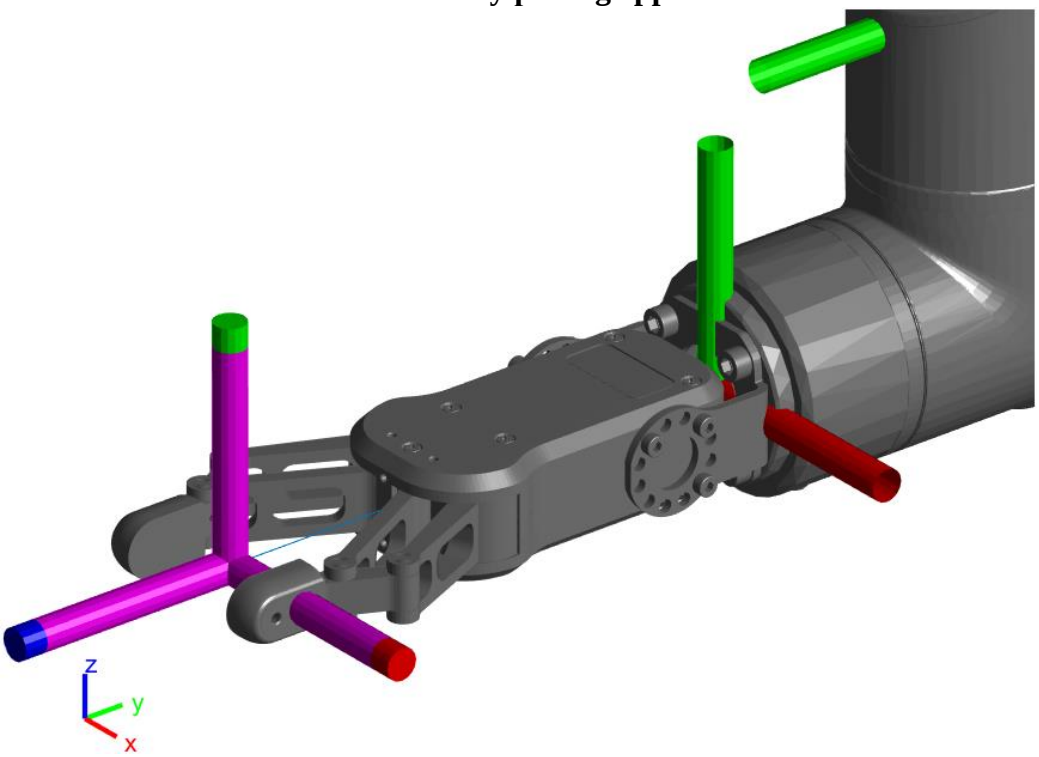
Obrázek 3.5: Kolizní model robota UR5e

3.1.2 Model gripperu RG2

Model gripperu byl opět stažen z GitHub úložiště. Protože se ale jedná pouze o 3D model, který je nepohyblivý, musel by být pro každé rozevření prstů vytvořený zvlášť. Pro potřeby našeho modelu byly tedy pomocí programu Autodesk Meshmixer vytvořeny dvě modifikace modelu tak, aby existovaly alespoň tři polohy prstů: zavřené, polootevřené a otevřené. [18]



Obrázek 3.6: Polohy prstů gripperu



Obrázek 3.7: Zápěstí s gripperem a vyznačenými souřadnicovými systémy

3.1.3 Plánování trajektorií modelu

Nejdříve je potřeba pomocí funkce z toolboxu vytvořit řešič inverzní kinematiky. K tvorbě samotné trajektorie slouží funkce *trapveltraj,* která interpoluje mezi body tak, aby vznikl trapézovitý průběh rychlosti. V našem případě je nastavena tím způsobem, že vytvoří dvacet mezikroků pro vzdálené body a čtyři mezikroky, pokud jsou body blízko u sebe, aby se ušetřil výpočetní čas. Zrychlovat a zpomalovat bude koncový bod vždy po jednu čtvrtinu pohybu. Pro animaci modelu je potřeba znát konfigurace kloubů v bodech získaných interpolací, k čemuž slouží právě řešič inverzní kinematiky. Získat je můžeme dvěma způsoby, jejichž výsledkem budou různé tvary trajektorie.

První možností je pohyb v kloubovém souřadnicovém systému. Ten je výpočetně jednodušší, protože konfigurace výchozího bodu je známá, takže se inverzní kinematika spočítá pouze v konečném bodě trajektorie. Výsledné konfigurace pouze vložíme do funkce *trapveltraj*, jejímž výsledkem jsou přímo interpolované konfigurace ve všech mezibodech. Protože pohyb probíhá po celou dobu současným otáčením všemi klouby a to pouze v jednom směru, výsledný pohyb koncového bodu probíhá po křivkách.

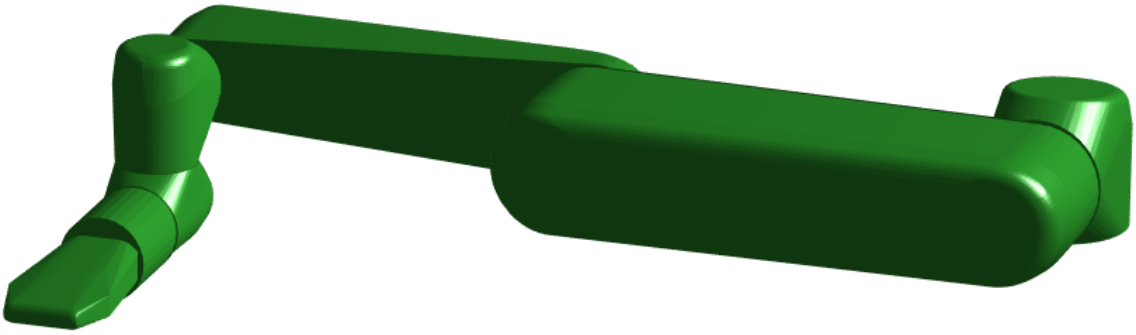
Pokud potřebujeme pohyb po přímce, řešíme trajektorii v kartézském souřadnicovém systému. V tomto případě jsou vstupem do funkce *trapveltraj* pouze souřadnice počátečního a koncového bodu na jednotlivých osách, výsledkem tedy budou pouze interpolované hodnoty [x, y, z]. Pro dosažení žádané pozice včetně rotací kolem os musíme využít ještě funkci *rottraj,* která je také součástí toolboxu. Po převedení vektorů souřadnic z funkce *trapveltraj* na transformační matice a kvaternionů z funkce *rottraj* na matice rotační je mezi sebou vynásobíme a inverzní kinematiku počítáme pro matice výsledné. Pohyb pak bude probíhat po přímce procházející bodem výchozím a žádaným. Jelikož k výpočtu inverzní kinematiky dochází pro každý mezibod zvlášť, výpočetní doba je úměrně delší.

Na rozdíl od kloubového pohybu může nastat situace, že řešič konverguje ke konfiguraci, která nenavazuje na předchozí, v důsledku čehož při animaci dojde k nereálnému přeskočení modelu do následující polohy. Příkladem takové situace jsou singularity, což jsou místa, ve kterých má inverzní kinematika velmi mnoho řešení nebo ve kterých při velké změně natočení kloubů dojde pouze k malému posuvu koncového bodu. Takové místo se nachází například nad ukotvením robota. K přeskočení animace by došlo také, pokud by některý z kloubů byl v limitní poloze, ale pohyb by měl pokračovat. V takovém případě se však po vzoru skutečného robota funkce zastaví a do konzole se vypíše chyba.

Samotná animace pohybu pak probíhá tak, že se v cyklu postupně prochází jednotlivé konfigurace a ty se vykreslují funkcí *show*. Pomocí ní se dá robot zobrazit s vizuálními modely, kolizními modely nebo pouze souřadnicovými systémy, případně kombinací.

3.1.4 Detekce kolize

Novou funkcí Robotics System Toolboxu je možnost pomocí kolizních objektů počítat vzdálenost mezi jednotlivými částmi modelu a zda mezi nimi dochází ke kolizi. K tomu slouží funkce *checkCollision*. Výstupem je jedna proměnná, která je buď 1 nebo 0 podle toho, jestli ke kolizi dochází, a čtvercová matice o počtu řádků a sloupců odpovídajícím počtu částí robota. Základní rovina je vždy v posledním řádku a sloupci. Pole v průsečíku řádku a sloupce dvou částí obsahuje vypočtenou vzdálenost mezi těmito částmi v metrech. U těch, které na sebe navazují, se výpočet neprovádí, výsledkem bude Inf. Pokud dojde mezi částmi ke kolizi, výsledkem v poli bude NaN, výpočet se zastaví a do zbývajících polí se doplní Inf. Funkce je nastavena tak, aby robota zastavila, pokud je vzdálenost menší než 1 cm. Při testování však docházelo ve většině případů k tomu, že podle výpočtů byly některé části robota v kolizi, přestože viditelně být nemohly, nejčastěji mezi širším ramenem a ukotvením robota, deskou základny nebo posledním článkem zápěstí. Prvním návrhem řešení bylo nahradit kolizní modely těchto objektů válci. Toto řešení zlepšilo přesnost pouze u zápěstí. Další možností tedy bylo kolizní objekty základny a ukotvení odstranit.



Obrázek 3.8: Upravený kolizní model s gripperem

Po této úpravě již nedochází k chybné detekci, pokud jsou gripper a rameno od sebe vzdáleny přibližně 10 cm a více. Při kratších vzdálenostech se může stát, že robot bude zastaven předčasně, naopak před jistou kolizí by měl zastavit téměř vždy, při testování však nastal i případ, že nedošlo k detekci kolize, která nastala. Funkci je tedy možné používat experimentálně nebo pro ukázku, nemůžeme se na ni ale stoprocentně spoléhat.

Na následující straně jsou dvě ukázky výpočtu v náhodných konfiguracích. V první nedošlo k žádné kolizi, výpočet tedy proběhl až do konce. Pro části bez kolizního modelu a části na sebe navazující je výsledek správně Inf. Z tabulky 5 vidíme, že nejblíže je podle výpočtu poslední část zápěstí k první, což je dáno konstrukcí, a prostřední část zápěstí k užšímu ramenu. Vypočtená vzdálenost jsou přibližně 4 cm, což odpovídá zobrazení na obrázku 3.8.

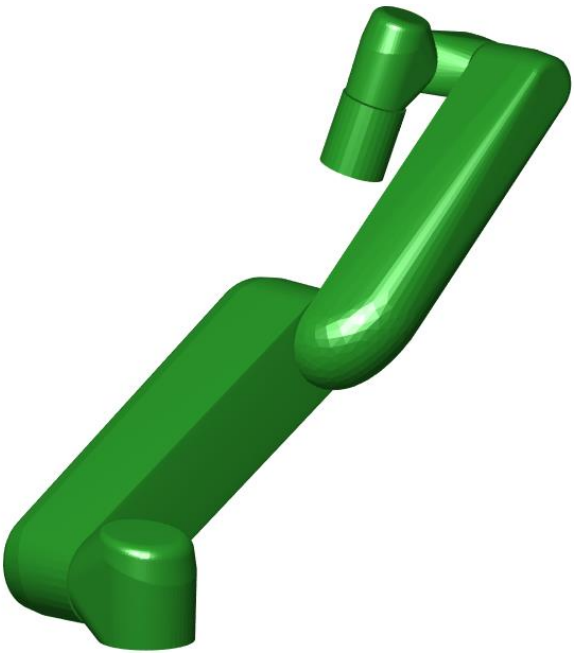
V druhé ukázce je vidět, že funkce vždy provede výpočty jedné části postupně pro všechny navazující, než pokračuje k další. Funkce tedy proběhla v pořádku, než došla k výpočtu vzdálenosti mezi užším ramenem a poslední částí zápěstí. Tam detekovala kolizi, do pole tedy doplnila NaN a do polí zbývajících Inf, jak je vidět v tabulce 6. Tato konfigurace je zobrazena na obrázku 3.9. Konec robota viditelně zasahuje do užšího ramene, kolize tedy byla detekována správně.

Tabulka 5: Výsledek výpočtu vzdáleností – ukázka 1

Část	base	shoulder	upper arm	forearm	wrist1	wrist2	wrist3	world
base	Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf
shoulder	Inf	Inf	Inf	0.2877	0.6824	0.6257	0.5536	Inf
upper arm	Inf	Inf	Inf	Inf	0.2979	0.2497	0.1634	Inf
forearm	Inf	0.2877	Inf	Inf	Inf	0.0379	0.0503	Inf
wrist1	Inf	0.6824	0.2979	Inf	Inf	Inf	0.0181	Inf
wrist2	Inf	0.6257	0.2497	0.0379	Inf	Inf	Inf	Inf
wrist3	Inf	0.5536	0.1634	0.0503	0.0181	Inf	Inf	Inf
world	Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf

Tabulka 6: Výsledek výpočtu vzdáleností – ukázka 2

Část	base	shoulder	upper arm	forearm	wrist1	wrist2	wrist3	world
base	Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf
shoulder	Inf	Inf	Inf	0.2209	0.2441	0.1991	0.2027	Inf
upper arm	Inf	Inf	Inf	Inf	0.2478	0.1873	0.2194	Inf
forearm	Inf	0.2209	Inf	Inf	Inf	0.0144	NaN	Inf
wrist1	Inf	0.2441	0.2478	Inf	Inf	Inf	Inf	Inf
wrist2	Inf	0.1991	0.1873	0.0144	Inf	Inf	Inf	Inf
wrist3	Inf	0.2027	0.2194	NaN	Inf	Inf	Inf	Inf
world	Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf



Obrázek 3.9: Ukázková konfigurace 1

Obrázek 3.10: Ukázková konfigurace 2

3.2 Ovládací objekt

Samotná knihovna funkcí je přístupná pomocí ovládacího objektu. Jeho hlavní parametry jsou nastaveny při inicializaci. Primárně je to mód ovládání, kde je možností buď robot nebo simulace. V případě zvolení módu robot je možné pokračovat jako uživatel student, který je nastaven automaticky, nebo jako administrátor. Pro přístup jako uživatel admin je potřeba zadat heslo uložené v souboru.

3.2.1 Parametry ovládacího objektu

Výpis parametrů objektu je v tabulce 7 a 8:

Tabulka 7: Veřejné parametry

user	Uživatel admin/student
mode	Módy robot/sim
password	Uložené heslo
sim_pose	Aktuální souřadnice koncového bodu modelu
sim_config	Aktuální konfigurace kloubů modelu
sim_freedrive	Proměnná pro interaktivní model
a_joint	Zrychlení kloubů v rad/s^2
a_tool	Zrychlení koncového bodu v m/s^2
v_joint	Maximální rychlost kloubů v rad/s
v_tool	Maximální rychlost koncového bodu m/s
ip_UR	Adresa robota pro připojení
tcp_data	Uložené sety dat nákladů
n_tcp_data	Číslo aktivního nákladu
target_pose	Aktuální cílová pozice
collision	Kolize on/off

Tabulka 8: Soukromé parametry

s1	TCP/IP port 29999
s2	TCP/IP port 30003
s3	TCP/IP port 30001
pose	Aktuální pozice robota
config	Aktuální konfigurace robota
force	Aktuální síly kloubů
active_tcp	Data aktivního nákladu
freedrive_status	Učící mód zapnutý/vypnutý

Objekt je navržen tak, že hlavní ovládací funkce mají stejnou syntaxi pro model i pro použití s robotem, takže je možné kód napsaný a vyzkoušený v simulaci používat jen s drobnými změnami i s robotem. Dvě uživatelské verze, tedy studentská a administrátorská, se liší omezeními, která mají zajišťovat bezpečnost použití ve výuce. Jedná se o omezení maximálního zrychlení a rychlosti koncového bodu i otáčení kloubů, jmenovitě na 5 cm/s^2 a 5 cm/s pro koncový bod a 20 stupňů/s^2 a 10 stupňů/s pro klouby.

Student také nemá oprávnění vypnout detekci kolize. Pro plný přístup je nutné zadat do příslušného parametru heslo, a to buď při inicializaci uživatelem admin, nebo lze objekt inicializovat jako student, vložit heslo do parametru a přepnout uživatele. Aby nemohl uživatel bez oprávnění tyto omezení odstranit, jinak změnit kód nebo zjistit heslo, celá knihovna je zašifrována pomocí *pcode*.

Výchozí IP adresa robota je nastavena na 10.1.1.2, pokud je v nastavení robota jiná, je potřeba spustit manuální režim na ovládacím panelu a změnit ji v nastavení robota, nebo lze objekt inicializovat v režimu simulace, změnit parametr adresy a přepnout objekt do módu robota.

Parametr *tcp_data* je struktura, v níž jsou uložená data o nákladu. Těmi jsou:

- mass – hmotnost nákladu, výchozí 0 kg
- CoG – poloha těžiště nákladu vůči lokálnímu souřadnicovému systému ve formátu [x, y, z], výchozí [0, 0, 0]
- pose – posuv souřadnicového systému nákladu vůči souřadnicím koncového bodu ve formátu [x, y, z, rot_x, rot_y, rot_z], výchozí [0, 0, 0, 0, 0, 0]
- str – string obsahující název setu dat pro snadnější identifikaci

Pokud dojde ke změně, nová data se automaticky nahrají do robota. Je důležité nastavit náklad pouze z Matlabu nebo na ovládacím panelu, jinak dojde ke konfliktu. K přepínání mezi jednotlivými uloženými náklady slouží parametr *n_tcp_data,* aktuálně nahraná data nákladu jsou uložena v parametru *active_tcp*. *Target_pose* slouží k uchování aktuální cílové pozice pro případ pozastavení a opětovného navázání pohybu robota.

3.2.2 Ovládání gripperu

Jak bylo popsáno v rešerši, gripper nelze ovládat stejným způsobem jako samotného robota. Jelikož je ale ovládání nástroje při používání robota nutností, bylo nutné přistoupit k méně praktickému, ale funkčnímu řešení. Využilo se možnosti pomocí dálkového ovládání nahrát do uživatelského prostředí ovládacího panelu program uložený v paměti a ten následně spustit. Byly tedy vytvořeny tři programy, každý obsahující pouze příkaz k rozevření gripperu a to na hodnoty 2, 50 a 100 mm, které odpovídají zavření, polovičnímu a úplnému otevření stejně jako v modelu. Pokud by bylo potřeba jiné rozevření prstů, lze jednoduše vytvořit nový program s žádanou hodnotou. Aby se program stihl načíst a prsty se stihly rozevřít, je do funkcí zařazena půlvteřinová prodleva.

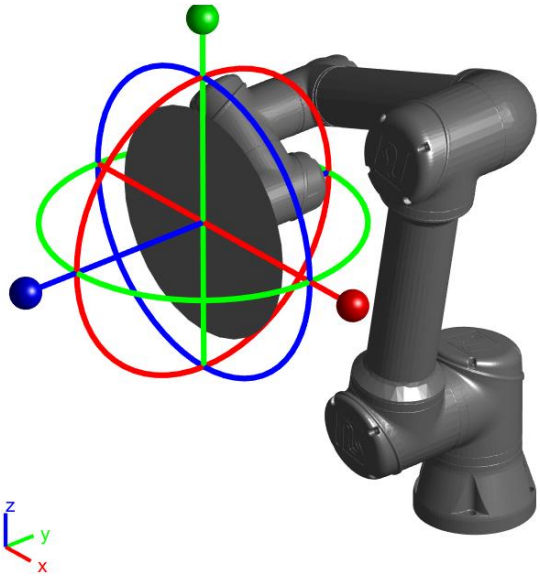
3.2.3 Seznam a popis funkcí

- UR_robot(mode, user, password) – inicializace objektu
  - user a password – nepovinné vstupy
- freedrive_on – zapne učící mód
  - v módu simulace otevře interaktivní model robota
- freedrive_off – vypne učící mód
  - v módu simulace uloží poslední pozici a konfiguraci, zůstane zobrazena
- refresh_freedrive_status – načtení stavu učícího módu zap/vyp
- add_gripper – přidá do modelu gripper v zavřené poloze (pouze simulace)
- clear_gripper – odstraní z modelu gripper (pouze simulace)
- open_gripper – otevře gripper do maximální polohy 100 mm
- close_gripper – zavře gripper
- half_gripper – otevře gripper na poloviční rozevření prstů 50 mm
- power_on – spustí a odbrzdí robota (pouze robot)
- power_off – vypne robota (pouze robot)
- refresh_status – vrátí pozici koncového bodu, konfiguraci a síly v kloubech
  - [pose, config, force] – pouze robot
- stop – zastaví robota (pouze robot)
- resume – robot bude pokračovat v pohybu (pouze robot)
- unlock – spustí robota ze stavu preventivního zastavení (pouze robot)
- movel(pose) – pohyb po přímce do polohy pose
  - pose = [x, y, z, rot_x, rot_y, rot_z]
- movej(pose) – pohyb v kloubovém s. s. do polohy pose
  - pose = [x, y, z, rot_x, rot_y, rot_z]
- movel_tcp(offset) – posuv o offset po přímce
  - offset = [x, y, z, rot_x, rot_y, rot_z]
- movej_tcp(offset) – posuv o offset v kloubovém s. s.
  - offset = [x, y, z, rot_x, rot_y, rot_z]
- move_joint(joint_num, angle) – otočí kloubem joint_num o angle
  - joint_num – číslo kloubu
  - angle – úhel ve stupních
- set_config(config) – přesune robota do žádané konfigurace
  - config = [a; b; c; d; e; f], rozmezí <-2π,2π>
  - pomocí pohybu v kloubových souřadnicích

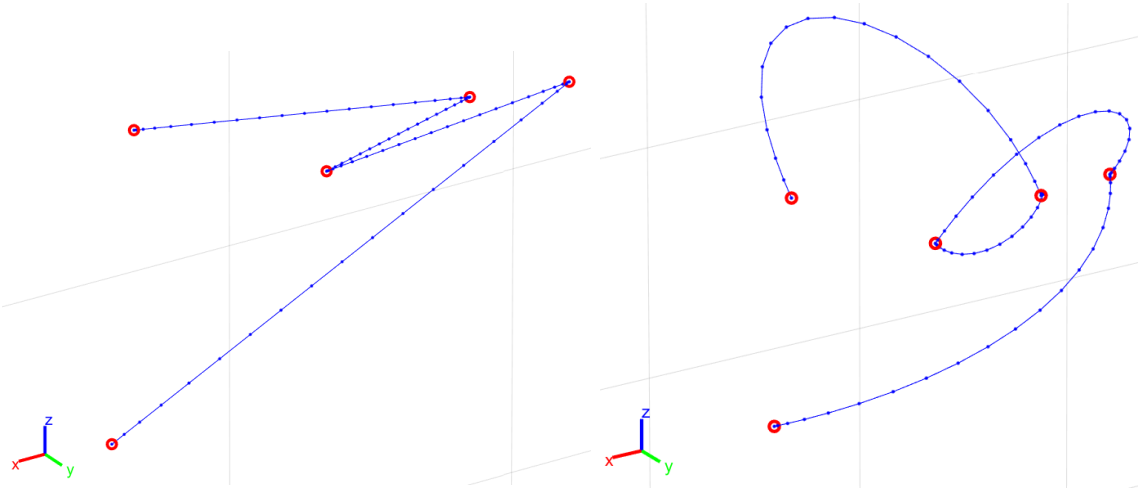
Při pohybu v Matlabu běží cyklus srovnávající snímanou aktuální polohu s žádanou, pokud se liší o 0,1 mm nebo 1 stupeň, cyklus se opustí a půl vteřiny se čeká před přijetím dalšího příkazu. Je-li povolena detekce kolize, zároveň se v modelu kontroluje vzdálenost částí způsobem popsaným v kapitole 3.1. Pokud je třeba změnit probíhající pohyb, musí se cyklus ukončit přes konzoli.

3.2.4 Ukázka programování

```
password = load('password');                    % načtení hesla
robot = UR_robot('robot', 'admin',password);    % inicializace objektu
robot.add_gripper;                              % do modelu kolize
robot.move_joint(2,-70);                        % otočení kloubem
robot.movej([x; y; z; rot_x; rot_y; rot_z]);    % kloubový pohyb
robot.half_gripper;                             % poloviční otevření
robot.movel([x; y; z; rot_x; rot_y; rot_z]);    % kartézský pohyb
robot.user = 'student';                         % přepnutí uživatele
robot.mode = 'sim';                             % přepnutí módu
robot.freedrive_on;                             % učící mód
```



Obrázek 3.11: Model v učícím módu



Obrázek 3.12: Srovnání trajektorií pomocí příkazů movel a movej

4 Ukázkové úlohy

Jako ukázka reálného použití knihovny byly navrženy tři jednoduché studentské úlohy zaměřené vždy na jednu oblast výuky, jmenovitě na kinematiku, programování a návrh regulátorů. Dostupné jsou ve formě skriptů společně s knihovnou. Součástí každé úlohy je i ukázkové řešení.

4.1 Kinematická úloha

Tato úloha je zaměřena na počítání dopředné kinematiky robotických ramen, konkrétně robota UR5e za použití D-H parametrů. Cílem je ukázat způsob tvorby transformačních matic a co představují ve vztahu souřadnicových systémů.

Zadání

Určete Denavit-Hartenbergovy parametry robota UR5e. Postup najdete v knize Robot Modeling and Control. [19]

Výsledné hodnoty porovnejte s hodnotami od výrobce. Dále pomocí nich vytvořte funkci, jež bude mít jako vstup sloupcový vektor šesti úhlů pro jednotlivé klouby a jako výstup transformační matici koncového bodu do globálního souřadnicového systému. Tuto matici poté porovnejte s transformační maticí vytvořenou funkcí *getTransform* a modelu z knihovny.

Řešení

Funkci navrhneme tak, že pomocí D-H parametrů, úhlů a rovnice 1 postupně vytvoříme šest matic, které mezi sebou následně vynásobíme.

$$\begin{bmatrix} \cos\theta_i & -\sin\theta_i \cos\alpha_{i,i+1} & \sin\theta_i \sin\alpha_{i,i+1} & a_{i,i+1}\cos\theta_i \\ \sin\theta_i & \cos\theta_i \cos\alpha_{i,i+1} & -\cos\theta_i \sin\alpha_{i,i+1} & a_{i,i+1}\sin\theta_i \\ 0 & \sin\alpha_{i,i+1} & \cos\alpha_{i,i+1} & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (1)$$

Výsledkem bude transformační matice obsahující vektor určující pozici koncového bodu v globálním souřadnicovém systému a rotační matici popisující rotace kolem jeho os. Tuto matici srovnáme s maticí získanou z funkce *getTransform* tímto skriptem:

```
robot = UR_robot('sim');
transform_matrix = getTransform(robot.model, konfigurace, 'wrist3_link');
pose = vase_funkce(konfigurace);
pose == transform_matrix;
```

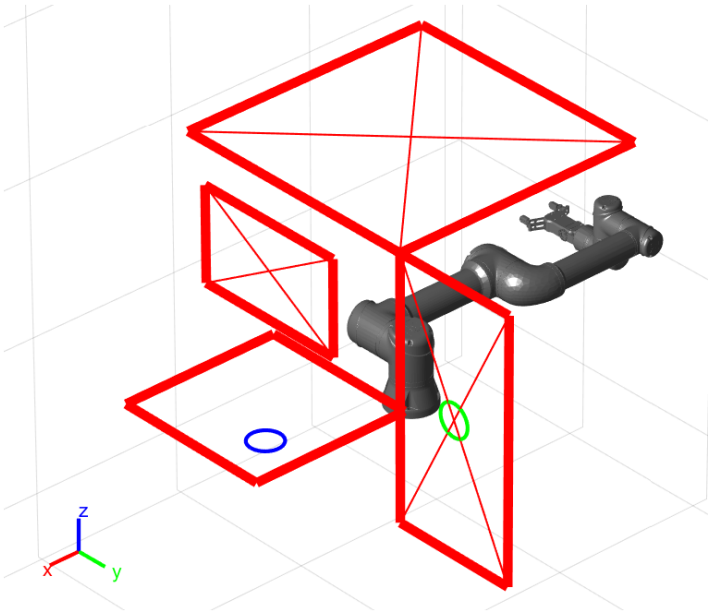
Pokud se všechny pozice matic rovnají, funkce výpočtu dopředné kinematiky je správná.

4.2 Úloha na manipulaci

Cílem této úlohy je nastínit programování robotického ramene přenášejícího objekt z jednoho bodu do druhého tak, aby se zároveň vyhnulo všem překážkám, což je úloha často kolaborativními roboty vykonávaná.

Zadání

Funkcí *prekazky* si zobrazte překážky v prostoru robota, značené červeně, a cílové body značené barevnými kruhy. Proškrtnuté překážky jsou stěny, ostatní jsou pouze linie. Vaším úkolem je přesunout konec robota s připojeným otevřeným gripperem do modrého kruhu, tam gripper napůl zavřít a následně jej přesunout do zeleného kruhu tak, aby žádná část robota neprošla překážkou.

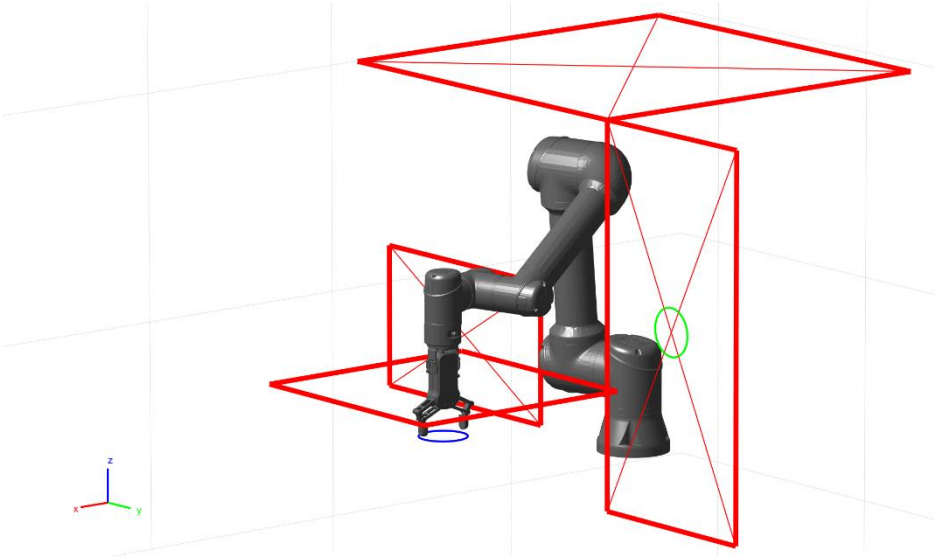


Obrázek 4.1: Výchozí stav úlohy

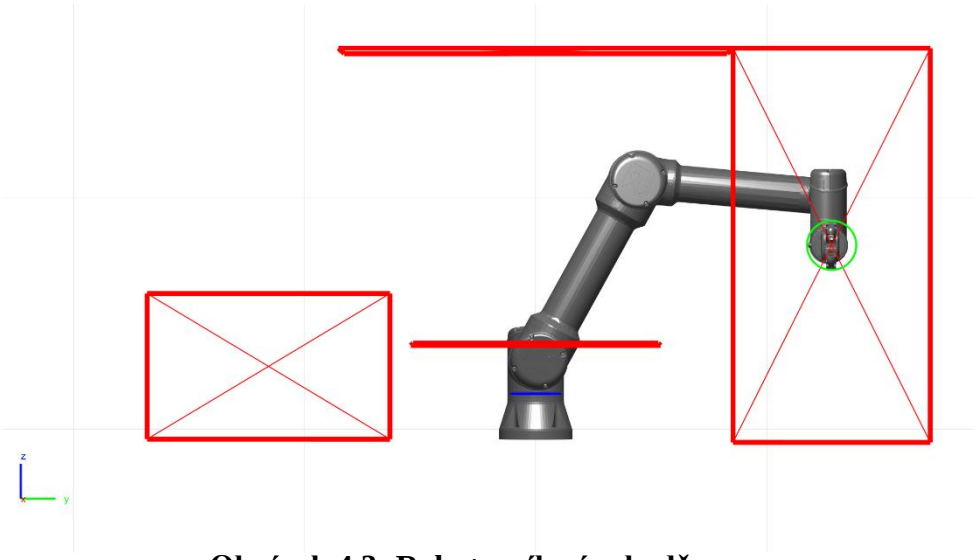
Řešení

Tento kód je pouze jedno z možných řešení:

```
robot = UR_robot('sim');
robot.add_gripper();
robot.open_gripper();
figure = prekazky(robot);
robot.move_joint(3,-130);
robot.move_joint(2,-70);
robot.movej([0.5;0;0.4;-pi;0;0]);
robot.movel([0.5;0;0.1;-pi;0;0]);
robot.half_gripper;
robot.movel([0.5;0;0.4;-pi;0;0]);
robot.move_joint(1,-270);
robot.movej([0.2;0.6;0.4;0;pi/2;0]);
robot.movel([0.4;0.6;0.4;0;pi/2;0]);
```



Obrázek 4.2: Robot v prvním bodě



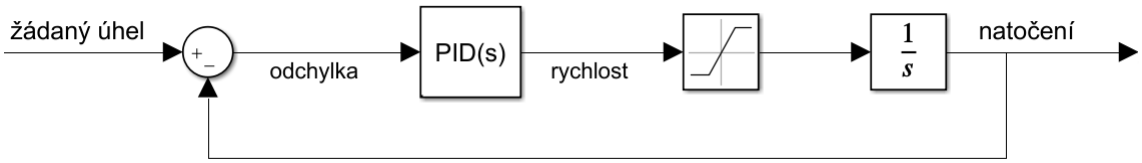
Obrázek 4.3: Robot v cílovém bodě

4.3 Regulační úloha

V této úloze mají studenti za úkol navrhnout regulátor natočení jednoho z kloubů robotického ramene. Cílem je na příkladu ukázat vliv jednotlivých složek PID regulátoru na výsledný průběh.

Zadání

Navrhněte konstanty P, I, D složek regulátoru. Ty pak společně s žádaným a současným úhlem zadejte do funkce *sim_regulátor* pro zobrazení průběhu natočení kloubu a vizualizaci pohybu robota. Jelikož se jedná o diskrétní pohyb, regulátor ve funkci na základě odchylky žádaného a současného úhlu nastaví velikost kroku představující rychlost za sekundu. Vstupem regulátoru je tedy žádaný úhel, regulovanou veličinou je natočení třetího kloubu, v modelu nazvaném *forearm*, a akční veličinou je krok pohybu, který je omezen na 10 stupňů za sekundu. Kvůli stabilitě by hodnoty neměly být větší než 1.



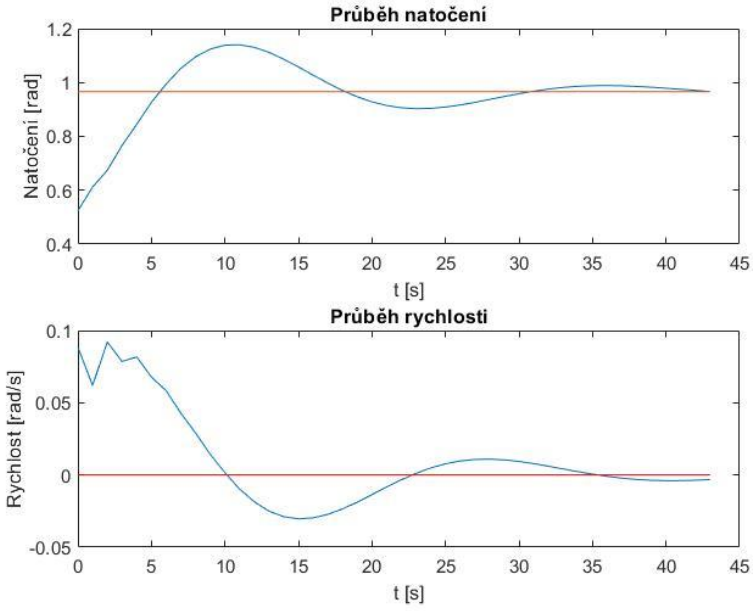
Obrázek 4.4: Schéma regulační úlohy

Řešení

Pro ukázku byly koeficienty nastaveny jako 0,2 pro proporcionální, 0,1 pro integrační a 0,5 pro derivační složku. Výsledný průběh natočení je vidět na obrázku 4.6.

```
sim_regulator(0.2,0.1,0.5,2.46pi/8,pi/6);     % parametry [K_P,K_I,K_D, úhel]
```

Obrázek 4.5: Regulovaný kloub



Obrázek 4.6: Regulovaný průběh natočení a rychlosti

5 Závěr

V práci byly nejdříve představeni kolaborativní roboti, model UR5e a gripper RG2 společně s Robotics System Toolboxem. Následně byl vysvětlen postup tvorby virtuálního modelu robota i gripperu a generování trajektorií.

Další část byla věnována tvorbě modelu. Nejdříve byl vysvětlen postup vytvoření základní struktury podle Denavit-Hartenbergových parametrů, ke které se následně připojily 3D modely pro vizualizaci a kolizní objekty. Dále byl popsán způsob generování trajektorií v kartézském a kloubovém systému a jejich využití k vytvoření animací napodobujících pohyb robota. Následovalo vysvětlení způsobu detekce kolize. Funkce k tomu sloužící jsou novou součástí toolboxu a nekonvergují vždy ke správnému výsledku. Pokud je detekce zapnuta, robot by měl být schopný včas zastavit a zabránit tak srážce. Na tuto funkci se aktuálně nelze stoprocentně spoléhat, je tedy experimentální součástí knihovny.

Na model navazoval popis základní struktury knihovny a vysvětlení použití programování pomocí objektu. Následně byly popsány jeho jednotlivé parametry a pro ně platící omezení nastavená ve studentské verzi.

Ovládání gripperu bylo zprvu výzvou, protože se nedařilo zprovoznit komunikaci mezi počítačem a gripperem. Jak bylo předpokládáno, důvodem bylo nahrávání funkcí gripperu do uživatelského prostředí robota, které ale není při dálkovém ovládání přímo využíváno. Tento problém byl nakonec vyřešen nepříliš elegantním, ale snadno přizpůsobitelným a hlavně funkčním řešením. Tím bylo vytvoření programů obsahujících pouze otevření nebo zavření gripperu v uživatelském prostředí robota a jejich uložení do paměti ovládacího panelu. Pokud chce uživatel gripper otevřít, příslušný kód se pouze nahraje z paměti a přehraje pomocí uživatelského prostředí.

V další části práce následoval popis jednotlivých funkcí obsažených v knihovně včetně jejich způsobu zadávání, který je identický pro použití v simulaci i s robotem. Pro příklad byla uvedena i krátká ukázka programování.

Na závěr byly pro názornost představeny tři příklady použití knihovny k řešení úloh představujících výukové zadání.

Navázáním na tuto práci by mohlo být vytvoření komunikace mezi robotem a počítačem mimo místní síť, aby mohl být ovládán například přes internet. Dále rozšíření seznamu funkcí nebo implementace dalších gripperů a lepší funkce pro jejich ovládání podle potřeby aktuálního projektu nebo experimentu. Užitečným rozšířením knihovny by také bylo vytvoření vlastního řešení pro detekci kolize, které by bylo přesnější a spolehlivější, a rozšíření modelu o dynamické parametry.

Obrázek 5.1: Konečný model UR5e s gripperem

6 Zdroje

[1] Demystifying Collaborative Industrial Robots 2018 [online], [rev. 2019-10], [cit. 2021-4-25]. Dostupné z: https://ifr.org/papers

[2] PITTMAN, Kagan. A History of Collaborative Robots: From Intelligent Lift Assists to Cobots. In: *engineering.com* [online]. 2016-10-28 [cit. 2021-4-25]. Dostupné z: https://www.engineering.com/story/a-history-of-collaborative-robots-from-intelligent-lift-assists-to-cobots

[3] FANUC. Kolaborativní průmyslové roboty mýtů zbavené. In: *vseoprumyslu.cz* [online]. [cit. 2021-4-25]. Dostupné z: https://www.vseoprumyslu.cz/inspirace/firemni-novinky/kolaborativni-prumyslove-roboty-zbavene-mytu.html

[4] Robots with high degrees of freedom face barriers to adoption [online]. In: *cobottrends.com*. 2019-10-2 [cit. 2021-5-15]. Dostupné z: https://www.cobottrends.com/robots-with-high-degrees-of-freedom-face-barriers-to-adoption/

[5] Cobots for the Win [online]. In: *qualitymag.com*. 2021-5-5 [cit. 2021-5-15]. Dostupné z: https://www.qualitymag.com/articles/96491-cobots-for-the-win

[6] Arsenal Robotic Dispensing System. In: *olympus-controls.com* [online]. 2020-5-13 [cit. 2021-5-15]. Dostupné z: https://olympus-controls.com/2020/05/13/arsenal-robotic-dispensing-system/

[7] CROWE, Steve. How a CNC Machine Shop Maximizes Productivity with Cobots. In: *Cobottrends.com* [online]. 2020-6-4 [cit. 2021-5-15]. Dostupné z: https://www.cobottrends.com/cnc-machine-shop-productive-robotics/

[8] OWEN-HILL, Alex. 7 Companies That Are Using Polishing and Buffing Robots. In: *blog.robotiq.com* [online]. 2019-7-18 [cit. 2021-5-15]. Dostupné z: https://blog.robotiq.com/7-companies-that-are-using-polishing-and-buffing-robots

[9] Universal Robots e-Series. 2021 [online]. [cit. 2021-4-27]. Dostupné z: https://www.universal-robots.com/e-series/

[10] In: *robots.ieee.org* [online]. [cit. 2021-5-15]. Dostupné z: https://robots.ieee.org/robots/universal/

[11] In: *universal-robots.com*. 2021-5-14 [online]. [cit. 2021-5-15]. Dostupné z: https://www.universal-robots.com/articles/ur/release-notes/release-note-software-version-58xx/

[12] In: *thinkbotsolutions.com* [online]. [cit. 2021-5-15]. Dostupné z: https://thinkbotsolutions.com/products/universal-robots-ur5e

[13] ZHU, Shijie 2020. Universal Robots Matlab Interface [online]. Dostupné z: https://github.com/zhusj16/Universal-Robots-Matlab-interface

[14] OnRobot RG2. 2021 [online]. [cit. 2021-4-27]. Dostupné z: https://onrobot.com/en/products/rg2-gripper

[15] In: *allied-automation.com* [online]. [cit. 2021-5-15]. Dostupné z: https://www.allied-automation.com/partners/onrobot/rg2-gripper/

[16] DH Parameters for Calculations of Kinematics and Dynamics [online]. [rev. 2021-2-23], [cit. 2021-5-2]. Dostupné z: https://www.universal-robots.com/articles/ur/application-installation/dh-parameters-for-calculations-of-kinematics-and-dynamics/

[17] NIEWINSKI, Dave 2019. UR5e meshes [online], [cit. 2021-5-2. Dostupné z: https://git-ce.rwth-aachen.de/henrik.hose/universal_robot/-/tree/kinetic-devel/ur_e_description/meshes/ur5e

[18] JOTAWAR, Sharath 2017. RG2 gripper [online], [cit. 2021-5-2]. Dostupné z: https://github.com/zhusj16/Universal-Robots-Matlab-interface

[19] Chapter 3: Forward Kinematics: The Denavit-Hartenberg Convention. SPONG, Mark W., Seth HUTCHINSON a M. VIDYASAGAR. *Robot Modeling and Control, 2nd Edition.* Wiley, 2020, s. 71-102. ISBN 9781119523994.

7 Seznam obrázků