UNIVERZITA PARDUBICE
Fakulta elektrotechniky a informatiky

Programová podpora pro práci s robotickým ramenem z prostředí MATLAB

Bc. Dominik Varga

Diplomová práce
2020

Univerzita Pardubice
Fakulta elektrotechniky a informatiky
Akademický rok: 2019/2020

ZADÁNÍ DIPLOMOVÉ PRÁCE
(projektu, uměleckého díla, uměleckého výkonu)

Jméno a příjmení:	**Bc. Dominik Varga**
Osobní číslo:	**I18234**
Studijní program:	**N2612 Elektrotechnika a informatika**
Studijní obor:	**Řízení procesů**
Téma práce:	**Programová podpora pro práci s robotickým ramenem z prostředí MATLAB**
Zadávající katedra:	**Katedra řízení procesů**

Zásady pro vypracování

Cíl: Vytvořit SW v pro ovládání robotického ramene Universal Robots UR3 z prostředí MATLAB pomocí příkazů zasílaných na server robota. SW umožní zasílat příkazy pro ovládání robota pomocí TCP/IP komunikace.
Obsah teoretické části: Robotická ramena, kategorie pro různé aplikační oblasti, způsoby programování, kolaborativnost, kognitivní robotika.
Obsah implementační části: Popis architektury systému Universal Robots, programování, možnosti ovládání robota z počítače, návrh ukázkových aplikací, dokumentace, ověření funkčnosti, zhodnocení.

Rozsah pracovní zprávy: **60**
Rozsah grafických prací:
Forma zpracování diplomové práce: **tištěná/elektronická**

Seznam doporučené literatury:

HAVELKA, Martin a Veronika STOFFOVÁ. Robotika – stavba a programování robotů. Olomouc: Univerzita Palackého v Olomouci, 2017. ISBN 978-80-244-5194-7.
ZÁDA, Václav. Robotika: matematické aspekty analýzy a řízení. Liberec: Technická univerzita v Liberci, 2012. ISBN 978-80-7372-882-3.
HAVEL, Ivan M. Robotika: Úvod do teorie kognitivních robotů. Praha: Státní nakladatelství technické literatury, 1980. Teoretická knižnice inženýra.

Vedoucí diplomové práce: **Ing. Daniel Honc, Ph.D.**
Katedra řízení procesů

Datum zadání diplomové práce: **7. listopadu 2019**
Termín odevzdání diplomové práce: **15. května 2020**

L.S.

Ing. Zdeněk Němec, Ph.D.
děkan

Ing. Daniel Honc, Ph.D.
vedoucí katedry

V Pardubicích dne 7. listopadu 2019

Prohlášení autora

Prohlašuji:

Tuto práci jsem vypracoval samostatně. Veškeré literární prameny a informace, které jsem v práci využil, jsou uvedeny v seznamu použité literatury.

Byl jsem seznámen s tím, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, zejména se skutečností, že Univerzita Pardubice má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 autorského zákona, a s tím, že pokud dojde k užití této práce mnou nebo bude poskytnuta licence o užití jinému subjektu, je Univerzita Pardubice oprávněna ode mne požadovat přiměřený příspěvek na úhradu nákladů, které na vytvoření díla vynaložila, a to podle okolností až do jejich skutečné výše.

Beru na vědomí, že v souladu s § 47b zákona č. 111/1998 Sb., o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších předpisů, a směrnicí Univerzity Pardubice č. 7/2019 Pravidla pro odevzdávání, zveřejňování a formální úpravu závěrečných prací, ve znění pozdějších dodatků, bude práce zveřejněna prostřednictvím Digitální knihovny Univerzity Pardubice.

V Pardubicích dne 11. 05. 2020

Bc. Dominik Varga

Poděkování

Tímto bych rád poděloval vedoucímu diplomové práce Ing. Danielu Honcovi, Ph.D. za vedení práce, ochotu, rady a připomínky ke zpracování. Dále bych rád poděkoval rodině a přítelkyni za podporu po celou dobu studia. Diplomová práce byla finančně podpořena projektem SGS 2020 Fakulty elektrotechniky a informatiky Univerzity Pardubice.

V Pardubicích dne 11. 05. 2020

Bc. Dominik Varga

ANOTACE

Cílem této diplomové práce je vytvořit software v prostředí MATLAB, umožňující zasílat příkazy robotickému ramenu UR3 a tím poskytnout možnost ovládat robotické rameno vzdáleně z počítače. Softwarové řešení je založeno na vytvoření síťového TCP/IP soketu a využití síťové architektury klient – server. Řešení využívá klientského rozhraní ramene UR3 pro práci v reálném čase, kterému je přiřazena role serveru a příkazy jsou odesílány z programu MATLAB, který zastává roli klienta. Navržené řešení umožní vyčítat informace o aktuálním stavu robota, ovládat robotické rameno a také vytvářet program mimo standardní prostředí pro programování robotického ramene UR3.

KLÍČOVÁ SLOVA

MATLAB, vzdálené ovládání, robotické rameno, UR3, TCP/IP, Universal Robots

TITLE

SW SUPPORT FOR ROBOTIC ARM IN MATLAB ENVIRONMENT

ANNOTATION

The objective of this diploma thesis is creating a software in the MATLAB environment, which will enable to send commands to robotic arm UR3 and allows to control functions of the robotic arm remotely from the computer. The software solution is based on the creation of a TCP/IP network socket and utilization of the client-server network architecture. The solution uses the UR3 arm client's interface for real-time operation. Robot acts as a server and listens to the commands send from MATLAB, which acts as a client. The final solution will allow to read information's about the current state of the robot, to control various functions of the robotic arm and also to create a program outside standard robot programming environment for the robotic arm UR3.

KEYWORDS

MATLAB, Remote Control, Robotic Arm, UR3, TCP/IP, Universal Robots

OBSAH

SEZNAM ZKRATEK A ZNAČEK

DoF	Degrees of Freedom
Cobot	Collaborative Robot
SCARA	Selective Compliant Articulated Robot Arm
TCP	Tool Center Point
TCP/IP	Transmission Control Protocol/Internet Protocol
DNS	Domain Name System
RTDE	Real Time Data Exchange

SEZNAM SYMBOLŮ PROMĚNNÝCH VELIČIN A FUNKCÍ

x	souřadnice osy x, mm
y	souřadnice osy y, mm
z	souřadnice osy z, mm
Rx	souřadnice úhlu natočení v ose x, rad
Ry	souřadnice úhlu natočení v ose y, rad
Rz	souřadnice úhlu natočení v ose z, rad

SEZNAM ILUSTRACÍ

SEZNAM TABULEK

ÚVOD

Robotika je moderní vědní obor zabývající se návrhem, tvorbou a aplikací mechanických strojů, lépe řečeno robotů, do činností ulehčujících lidem každodenní život. V průmyslovém světě 21. století se konkrétně jedná o téma průmyslové robotiky, jež se zabývá aplikací robotů do výrobních procesů či způsobů automatizace těchto procesů. Implementací robotů dochází k vylepšení a zefektivnění výroby. Robotika je tedy klíčovým prvkem či nástrojem automatizace a často bývá spojována s průmyslovou revolucí 4.0, jenž je v současné době v plném proudu.

Cílem této diplomové práce je vytvořit software, který umožní vzdáleně ovládat a monitorovat kolaborativní průmyslové robotické rameno Universal Robots UR3 prostřednictvím počítače. Software vytvořený v Matlabu poskytne uživateli jednoduchou nabídku operací, které lze s ramenem provádět. Jedná se zejména o čtení informací potřebných k ovládání ramene a zasílání příkazů za účelem provádění pohybových operací a zásahů. Program tedy umožní provádět základní operace s ramenem vzdáleně z počítače a nabídne tak jinou alternativu řízení než prostřednictvím standardního programovacího prostředí robota.

1 TEORETICKÁ ČÁST

1.1 AUTOMATIZACE A ROBOTIZACE ANEB ČLOVĚK A STROJ

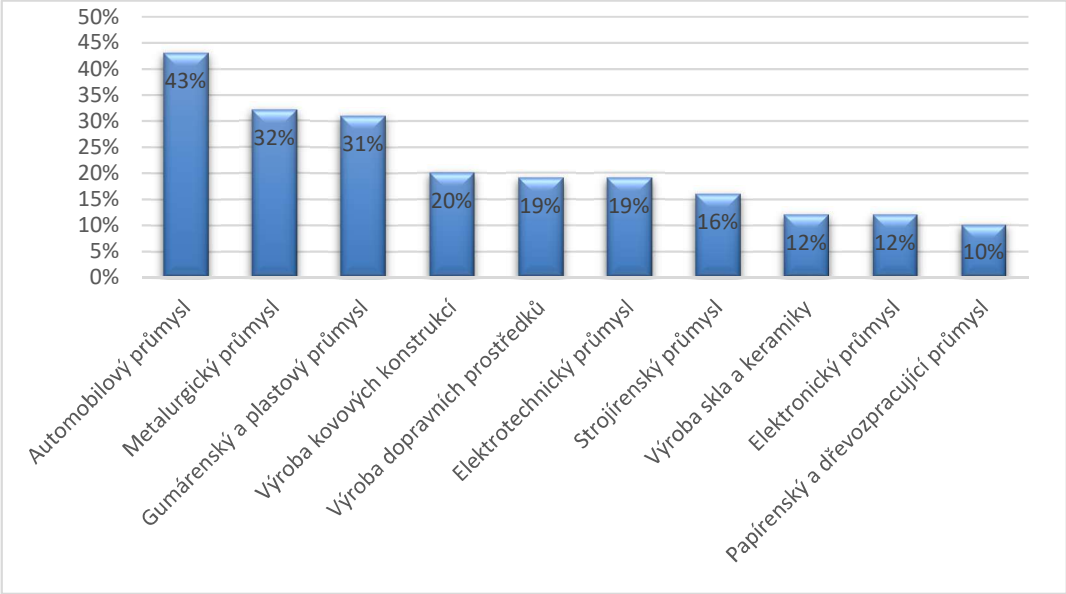
Náš život je rychlý a plný technologických vymožeností, bez kterých bychom si naše každodenní počínání už asi nedovedli ani představit. Již od pradávna se člověk snažil ulehčit si práci tím, že používal nejrůznější nástroje, které se náročným vývojem snažil, pokud možno, co nejvíce zdokonalit. Takovým vývojem vznikly speciální nástroje, lépe řečeno spíš stroje, které mohou částečně pomáhat a usnadňovat lidskou práci nebo ji dokonce provádět samostatně bez pomoci člověka.

Jedním z těchto velmi oblíbených nástrojů dnešní doby jsou roboty, které se v průmyslu čím dál tím častěji implementují do výrobního procesu a takové implementaci se odborně říká robotizace. Pod tímto pojmem se v průmyslu rozumí takový proces, při němž se lidský faktor nahrazuje strojem, lépe řečeno průmyslovým robotem. Robotizace je tedy velmi užitečný nástroj dnešní průmyslové automatizace, který slouží k částečné nebo úplné automatizaci výrobních procesů. Průmyslové roboty a robotizace obecně je dneska velmi oblíbené a často řešené téma, neboť úzce souvisí s tzv. průmyslovou revolucí 4.0 (Duchoslav, 2017).

1.1.1 Robotická ramena v průmyslu

Robotická ramena, někdy též označovaná jako průmyslové roboty, průmyslové manipulátory či kolaborativní servisní roboty, jsou mocný a účinný nástroj dnešního průmyslového světa, který značnou mírou ovlivňuje kvalitu výroby daného výrobku. Není se tedy čemu divit, že téměř každá větší továrna, která něco vyrábí či zpracovává, se snaží roboty implementovat do svého výrobního procesu, i za cenu větších pořizovacích nákladů. Investice do již zmíněné robotizace může být klíčová a při správném použití pro danou oblast může poskytnout velkou konkurenční výhodu, a tudíž i větší zisk.

Důkazem jsou údaje vydané českým statistickým úřadem. Tyto údaje ukazují, že průmyslové či kolaborativní roboty používalo za rok 2018 16 % firem napříč celou Českou republikou, přičemž největší zastoupení mají velké podniky. Velmi zajímavý je také pohled na procentuální zastoupení robotů v jednotlivých průmyslových odvětvích, kde jednoznačně vede automobilový průmysl, viz obrázek 1.1 (Zbíralová, 2019).



Obrázek 1.1 – Procentuální zastoupení robotů v průmyslových odvětvích

1.1.2 Definice robotického manipulátoru

Na otázku, co to vlastně robotický manipulátor či robotické rameno je, lze nalézt mnoho různých odpovědí. Například norma ISO 8373:2012 definuje robotické rameno jako automaticky ovládaný a přeprogramovatelný manipulátor, který lze programovat ve třech nebo více osách a je určený převážně pro průmyslovou automatizaci. Podobně znějících definic lze najít mnoho, ale robotické rameno si lze jednoduše představit jako mechanickou strukturu, která za pomocí elektrických motorů, které se chovají jako servomotory, tuto strukturu mění.

Hlavním úkolem takové struktury je, pokud možno, co nejpřesněji simulovat, tedy napodobovat a do značné míry tak nahrazovat chování lidské paže. Tím je možné na některých místech a v určitých aplikacích provést již zmiňovanou robotizaci a redukovat tak lidský faktor. Takové nahrazení člověka robotem přináší nespočet výhod, neboť robot může neúnavně pracovat desítky hodin denně a může provádět i takové činnosti, které člověk nemůže, jako například práci v nebezpečném prostředí (Gupta, 2017).

1.2 KATEGORIZACE ROBOTICKÝCH MANIPULÁTORŮ

Z předchozího je zřejmé, že průmyslové roboty jsou velmi užitečný a mocný nástroj k dosažení efektivity, přesnosti, kvality a mnoho dalšího, ale jak již bylo zmíněno, je velmi důležitý správný výběr pro danou oblast. Je tedy obtížné poznat, který robot je pro daný obor, či konkrétní činnost, ten správný, a proto jsou roboty používané v průmyslu nějakým způsobem

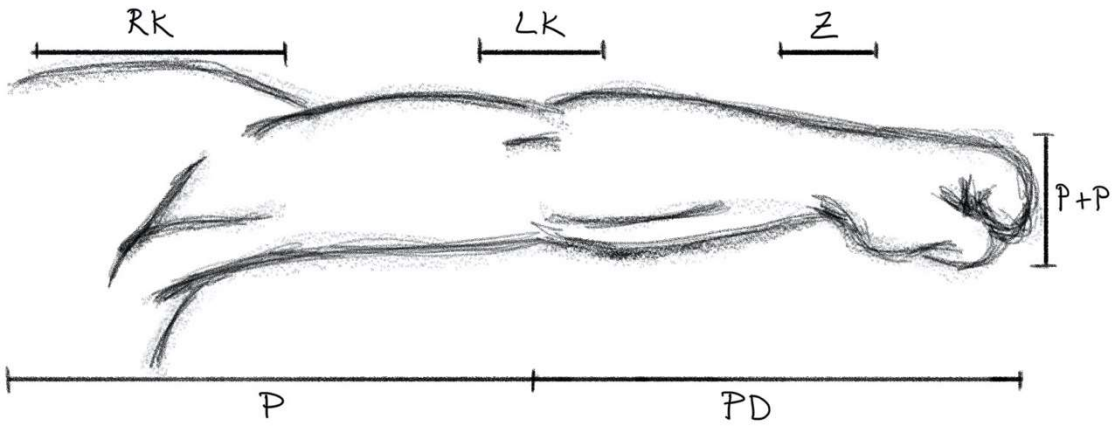
kategorizovány dle různých hledisek do různých kategorií. Tím jsou definovány značné rozdíly mezi jednotlivými typy robotických ramen a takové rozdělení je velmi důležité, neboť usnadňuje výběr konkrétního robotického ramene pro danou činnost. Způsobů, jakými lze robotická ramena kategorizovat, je spousta, ale zde budou uvedeny jenom základní jednoduché typy, které jasně poukazují na odlišnosti mezi jednotlivými typy manipulátorů. Základní dva způsoby kategorizace robotických manipulátorů jsou:

- kategorizace podle počtu stupňů volnosti,
- kategorizace dle typu pohybu a pracovního prostoru robota.

1.2.1 Kategorizace manipulátorů podle počtu stupňů volnosti

Nejzákladnějším a velmi oblíbeným a jasným způsobem, jakým lze robotická ramena klasifikovat, je podle počtu stupňů volnosti (DoF = **D**egree **o**f **F**reedom). Tento způsob klasifikace je v dnešní době velmi oblíbený, protože tato informace udává pohybové vlastnosti ramene a je tak klíčová při správné volbě robota. V jednoduchosti si lze představit, že stupně volnosti definují, v kolika směrech se mohou jednotlivé části ramene pohybovat nezávisle na sobě, a rovněž je tímto způsobem jednoduše definován stav soustavy. Každým použitým kloubem je do systému zaveden jeden stupeň volnosti.

Jak už bylo v pododdíle 1.1.1 psáno, úkolem každého robotického ramene je, pokud možno, co nejpřesněji do jisté míry napodobit chování lidské horní končetiny. Pro zajímavost je na obrázku 1.2 ukázán jednoduchý model lidské horní končetiny, která má od ramenního kloubu po konce prstů celkem 27 stupňů volnosti, popsaných v tabulce1.1 (Král, 2016; Dale, 2009)



Obrázek 1.2 – Lidská horní končetina

Tabulka 1.1 – Počet stupňů volnosti horní končetiny

Části horní končetiny	Počet stupňů volnosti
Ramenní kloub (RK)	2°
Paže (P)	1°
Loketní kloub (LK)	1°
Předloktí + dlaň (PD)	1°
Zápěstí (Z)	2°
4 Prsty + palec (P+P)	$4 \cdot 4 + 3 + 1 = 20^{\circ}$

Pokud má tedy robotické rameno volně manipulovat s nějakým tělesem, musí mít v manipulačním prostoru alespoň tolik stupňů volnosti, jako má dané těleso. Pohybem ramene je myšlena translace neboli posuvný pohyb, či rotace neboli otáčení ramene. Podle počtu DoF lze robotické manipulátory jednoduše dělit na:

- deficitní robotický manipulátor,
- redundantní robotický manipulátor,
- univerzální robotický manipulátor.

Deficitní robotický manipulátor

V praxi se často vyskytuje situace, kdy je potřeba manipulovat s nějakým objektem například pouze v rovině a není tak potřeba nasazovat robota s příliš mnoho stupni volnosti. V takovém případě je možné použít deficitní robotický manipulátor, což je označení takového typu manipulátoru, které disponují méně než šesti DoF. Deficitní manipulátory se tedy hodí zejména tam, kde není nutný složitý pohyb ani manipulační schopnosti robota. Při správném výběru a použití deficitního robota tam, kde stačí omezené množství stupňů volnosti, lze značně ušetřit, neboť není potřeba pořizovat složitější typy robotů (Skařupa, 2007).

Redundantní robotický manipulátor

Redundantní robotické manipulátory jsou přímým opakem manipulátorů deficitních, a vyznačují se větším počtem stupňů volnosti, než je v daném manipulačním prostoru nutné. Do této speciální skupiny patří manipulátory se sedmi DoF. Roboty s osmi a více stupni volnosti se v průmyslových aplikacích objevují spíše ojediněle, ale občas si své uplatnění nacházejí. Typickým představitelem této skupiny jsou manipulátory s hadovitým tvarem, které se hodí zejména tam, kde je zapotřebí větších manipulačních schopností (Skařupa, 2007).

Univerzální robotický manipulátor

Aby robotický manipulátor měl možnost dosáhnout do libovolné polohy a orientace v trojrozměrném prostoru, musí disponovat alespoň šesti stupni volnosti. Přesně takovou dovedností disponují univerzální manipulátory, které mají právě šest kloubů, a tudíž i šest stupňů volnosti. Univerzální manipulátor tímto způsobem jednoznačně vymezuje polohu a orientaci manipulujícího objektu v pravoúhlém kartézském souřadném systému pomocí šestice číselných údajů. Tento typ manipulátorů se značně podobá lidské horní končetině a v průmyslu se objevuje velmi často, neboť je všestranný a lze jej použít pro mnoho činností a úkonů (Skařupa, 2007).

Je patrné, že robotická ramena mají vůči horní lidské končetině svoje limity a omezení, ale v průmyslu se většinou jedná o jednoduché manipulační činnosti, pro které jsou schopnosti robotických manipulátorů dostatečné. Aby bylo možné s robotickým ramenem dosáhnout stejného počtu stupňů volnosti (ne však citu a obratnosti) jako s lidskou končetinou, je zapotřebí použít takový koncový efektor, který nahrazuje tvarem i funkcionalitou lidskou ruku, která podle tabulky 1.1 sama disponuje 20 stupni volnosti.

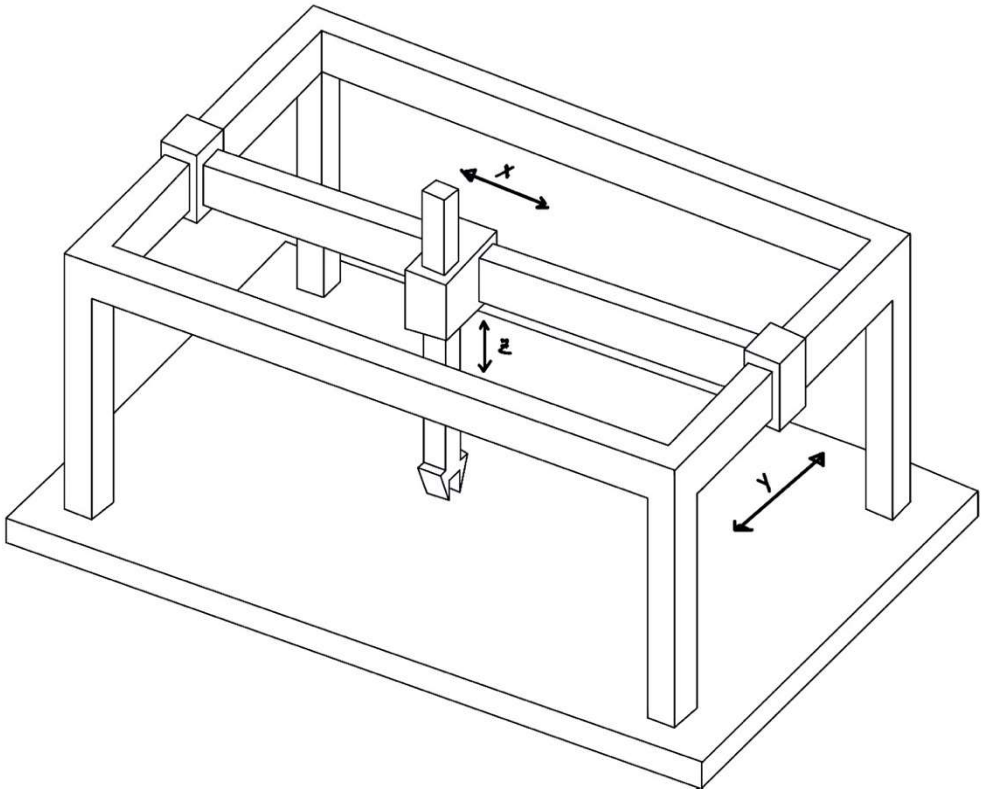
1.2.2 Kategorizace manipulátorů dle typu pohybu a pracovního prostoru

Při pořizování robotického manipulátoru je klíčové, v jakém prostředí bude pracovat a také jaký pohyb bude vykonávat. Dalším velmi užitečným způsobem, jak lze manipulátory od sebe odlišit, je podle typu vykonávaného pohybu neboli kinematiky daného manipulátoru. Tento způsob klasifikace je trochu jednodušší než v předchozím případě, neboť se bere zřetel hlavně na typ pohybu a tvar pracovního prostoru robota a zájemce o robotický manipulátor tak nemusí znát definici stupňů volnosti. U tohoto typu dělení se tedy nesleduje počet DoF, i když s ním úzce souvisí, neboť jak již bylo zmíněno, stupně volnosti definují pohybové vlastnosti manipulátoru. Jednoduchá klasifikace typů manipulátorů dle vykonaného typu pohybu je následující:

- kartézský manipulátor,
- cylindrický manipulátor,
- sférický manipulátor,
- angulární manipulátor.

Kartézský manipulátor – TTT

Kartézský robotický manipulátor znázorněný na obrázku 1.3 je typ, jehož osy jsou tři protínající se přímky x, y, z. U tohoto typu nedochází ke změně orientace objektu, dochází pouze k přesunu v kartézském souřadném systému. Kartézský manipulátor tedy nabízí pouze základní třírozměrný translační pohyb a dle předchozího způsobu klasifikace tento typ robota spadá do skupiny deficitních manipulátorů, neboť disponuje pouze třemi stupni volnosti, což v případě vhodného využití tohoto robota není překážkou. Pracovním prostorem kartézského manipulátoru je hranol. Tento robot se výborně hodí zejména pro montážní práce, kde je hlavním úkolem něco překládat neboli pro takzvané operace „pick and place – vezmi a polož“. Velikou výhodou tohoto manipulátoru je mimo jiné snadné přímé programování, ale nevýhodou je například to, že robot může manipulovat pouze s objekty před sebou.

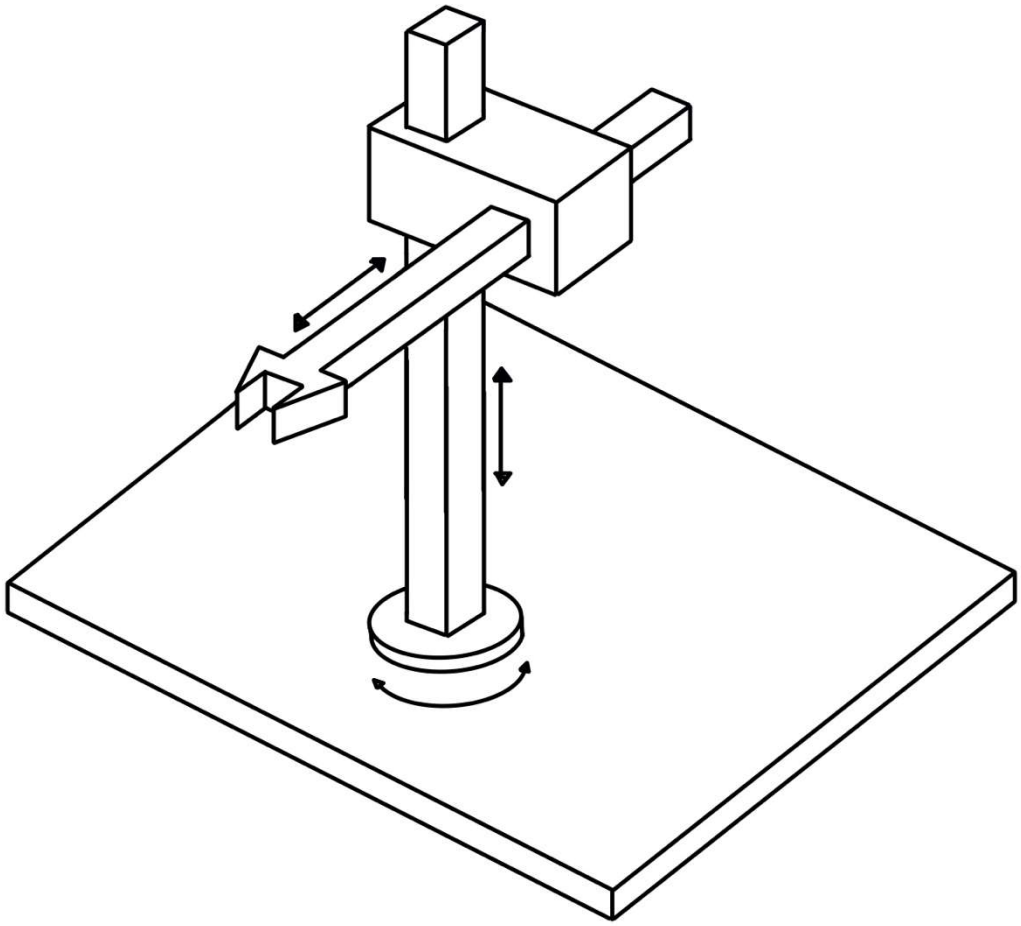


Obrázek 1.3 – Kartézský robotický manipulátor

Cylindrický manipulátor – RTT

Cylindrický robotický manipulátor, jehož vyobrazení lze vidět na obrázku 1.4, je typ manipulátoru disponující jedním rotačním pohybem a dvěma translačními posuvnými pohyby. Na rozdíl od předchozího typu má cylindrický manipulátor jeden rotační kloub, čímž může docházet ke změně orientace manipulujícího objektu a tím umožnuje lepší manipulaci

s objektem, neboť může dosáhnout všude kolem sebe. Polohování robota se provádí v cylindrickém válcovém souřadném systému a pracovním prostorem tohoto robota je válcový prstenec. Podobně jako kartézský manipulátor se snadně programuje a skvěle se hodí zejména pro montážní operace. Velkou nevýhodou tohoto manipulátoru je potřeba velkého manipulačního prostoru k manévrování a nemožnost dosáhnout nad sebe. V praxi se velmi často objevuje modifikace cylindrického manipulátoru nazývaná jako montážní manipulátor SCARA. V porovnání s předchozí metodou klasifikace robotických manipulátoru spadá stejně, jako kartézský manipulátor, do skupiny robotů deficitních.

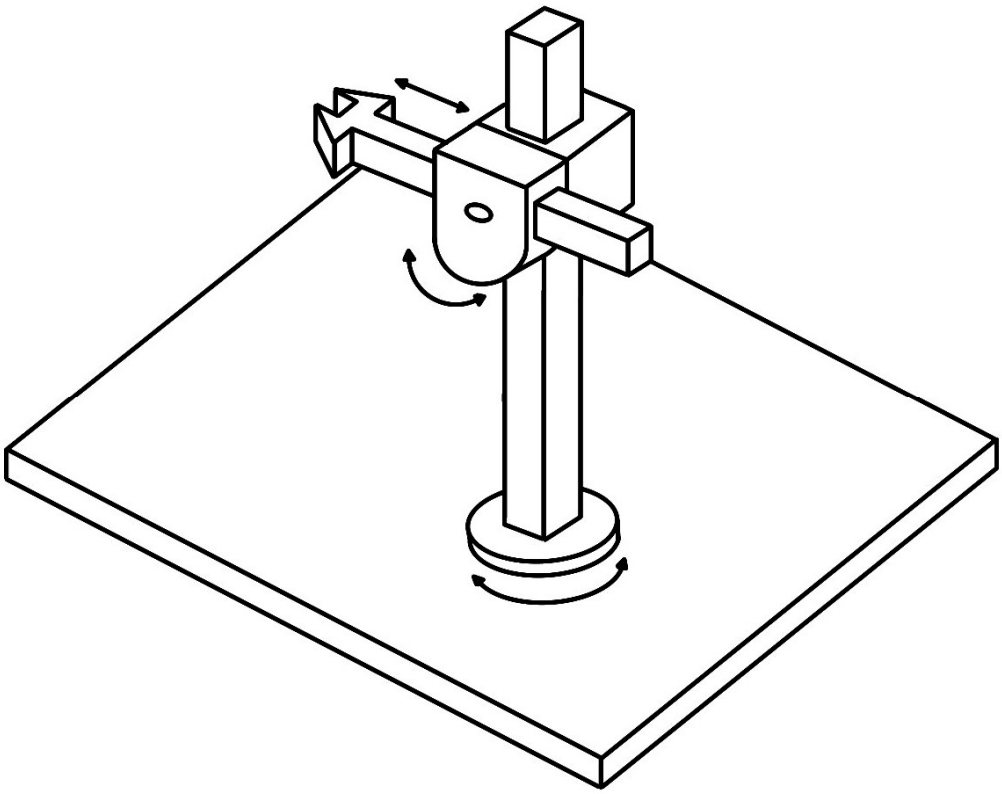


Obrázek 1.4 – Cylindrický robotický manipulátor

Sférický manipulátor – RRT

Stejně jako v případě cylindrického manipulátoru, tak i u sférického typu robota (obrázek 1.5), dochází ke změně orientace s manipulujícím objektem. Sférický manipulátor ale disponuje dvěma rotačními pohyby a jedním translačním posuvným pohybem. Dva rozměry sférického manipulátoru jsou tedy úhly a třetí rozměr je lineární. Jak již z názvu vyplývá, polohování tohoto typu manipulátoru se provádí ve sférické neboli kulové soustavě souřadnic

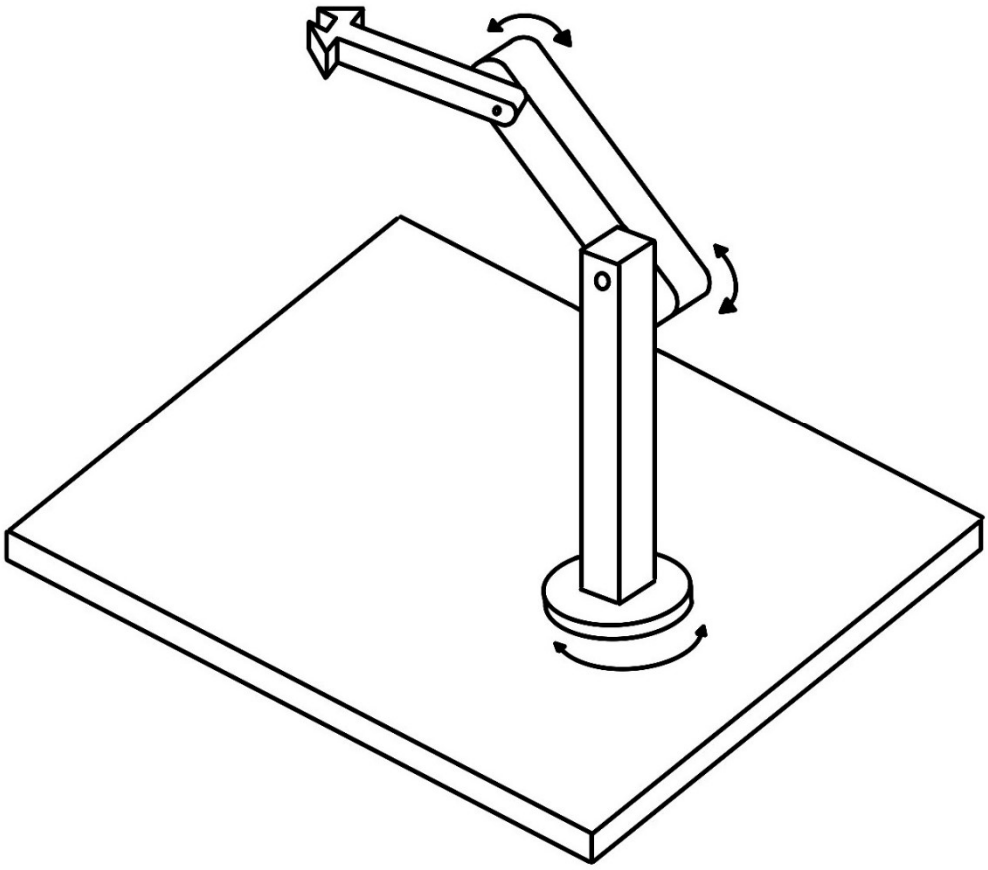
a pracovní prostor tohoto robota je kulový. Sférický manipulátor se výborně hodí zejména pro svařovací operace, například pro obloukové či bodové svařování. Velkou nevýhodou tohoto manipulátoru je složitější koordinace, vizualizace a programování.



Obrázek 1.5 – Sférický robotický manipulátor

Angulární manipulátor – RRR

Angulární robotické rameno, někdy též známo pod označením jako kloubový robot, znázorněný na obrázku 1.6, je typ manipulátoru, jenž se vyznačuje třemi rotačními pohyby. Všechny tři rozměry robota jsou tedy úhly natočení kloubů. V praxi se angulární manipulátor používá nejčastěji, neboť kinematické rozložení tohoto manipulátoru je velmi podobné lidské paži. Angulární manipulátor ve svém pracovním prostoru dosáhne na objekt z každé strany, a to mu umožnuje nejlepší manipulovatelnost s objektem. Tento typ se vyznačuje velmi dobrými dynamickými vlastnostmi, mechanickou flexibilitou a skvěle se hodí při montážních operacích, jako jsou svařování, lakování apod. Pracovní prostor manipulátoru je kulový vrchlík a oproti výše zmiňovaným manipulátorům je menší. Velkou nevýhodou tohoto manipulátoru je přesnost, která klesá s délkou ramene a také složitější ovládání a programování.



Obrázek 1.6 – Angulární robotický manipulátor

1.3 PROGRAMOVÁNÍ ROBOTICKÝCH RAMEN

Předchozí oddíly pojednávaly o tom, co jsou robotická ramena a k čemu všemu je lze využít. Z definice, viz pododdíl 1.1.1, však vyplývá, že robotický manipulátor je přeprogramovatelný mechanický stroj. Je tedy jasné, že aby robotická ramena mohla vykonávat svou činnost, potřebují od uživatele program, který definuje, co a jakým způsobem budou vykonávat. Na základě uloženého programu pak ovládá řídicí jednotka činnost robota. Způsobů, jak manipulátory programovat je vícero, ale mezi základní typy programovacích metod patří online a off-line programování.

1.3.1 Online programování

V dnešní době je nejvíce rozšířenou volbou metoda online programování, kdy obsluha tvoří program robota přímo na pracovišti prostřednictvím takzvaného „teach-pendantu“ jehož použití je znázorněno na obrázku 1.7. Teach pendant je kompaktní, dnes již téměř vždy plně dotykové zařízení, které nabízí mnoho důležitých funkcí, jako například ovládání pohybu

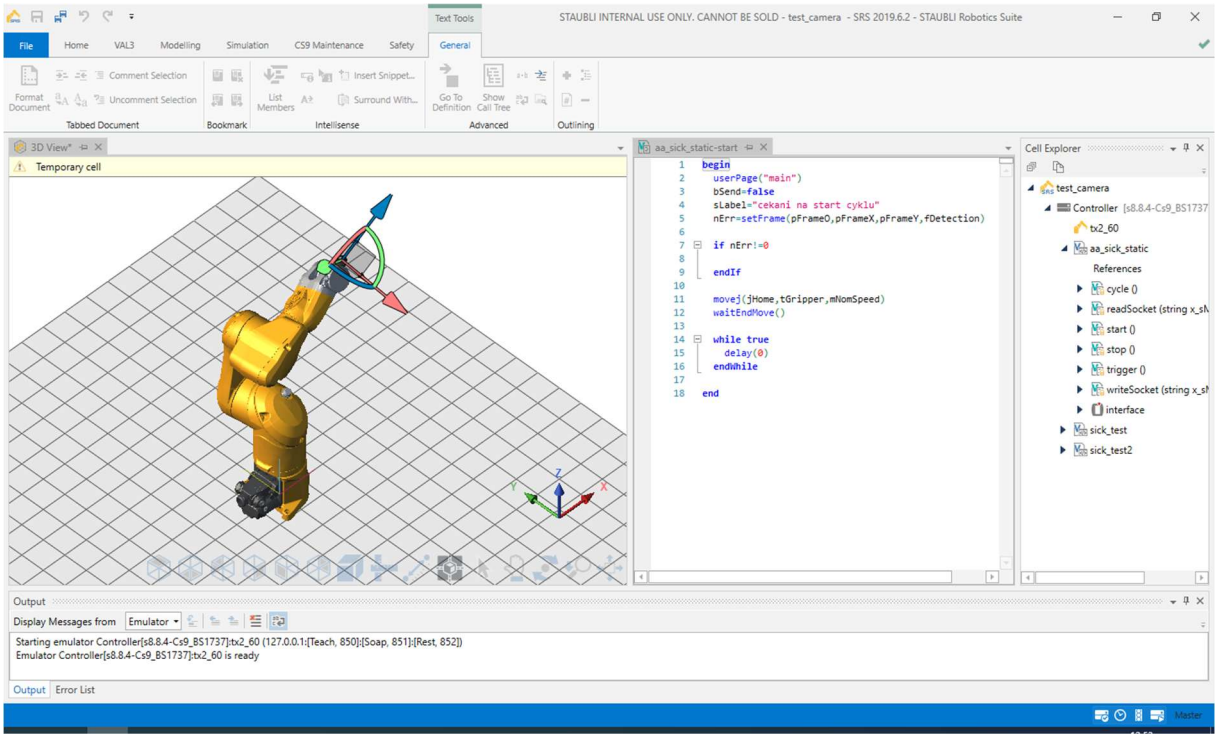
robota, možnost nouzového zastavení manipulátoru v případě kolize, a mnoho dalších funkcí. Teach pendant lze rovněž využít k ovládání a přímému programování v programovacím jazyce robotického manipulátoru. Velkou výhodou teach pendantu je jeho kompaktnost a malé rozměry zařízení, čímž je umožněno ovládání a programování v těsné blízkosti robota, a tudíž i okamžitá kontrola funkcionality a správnosti programu a tím i bezchybnost provozu manipulátoru (Kolíbal, 2016; Gupta, 2017).

Obrázek 1.7 – Online programování pomocí teach pendantu (KUKA.SystemSoftware, 2019)

1.3.2 Off-line programování

Další alternativu dnešního programování robotických manipulátorů principiálně zobrazuje obrázek 1.8 a tato metoda se nazývá off-line programování. Tento způsob programování je díky velké dostupnosti silné výpočetní techniky velmi oblíbený. Tato metoda programování je založená na softwaru pro běžný počítač, který nabízí virtuální prostředí pro psaní a tvorbu programu v programovacím jazyce robota. Prostředí by mělo být, pokud možno, vizuálně i funkčně téměř totožné jako to, které nabízí teach pendant robota a ideálně by mělo poskytnout stejné možnosti a funkce. Metoda off-line programování nabízí řadu výhod, mezi které patří zejména možnost vytvářet program z pohodlí domova či kanceláře bez nutnosti být

fyzicky u robotického manipulátoru. Program si tak lze připravit předem, zkopírovat na přenosné zařízení a následně přenést do teach pendantu manipulátoru. Rovněž velikou výhodou tohoto typu programování je možnost off-line simulace, čímž lze jednoduše ověřit správnost napsaného programu a simulovat chování manipulátoru. Metoda se tedy výborně hodí zejména pro výzkum, optimalizaci a pro testování koncepcí na zařízení (Kolíbal, 2016; Gupta, 2017).



Obrázek 1.8 – Off-line programování a simulace

1.3.3 Interaktivní metoda programování

Interaktivní metoda programování je moderní způsob tvorby programu pro manipulátor založený na vzájemné interakci robota a člověka. Jedná se vlastně o vylepšenou verzi online programování, která umožnuje operátorovi přímou interakci s manipulátorem a s využitím teach pendantu lze manipulátor učit a tvořit tak program. Velmi oblíbenou funkcí této metody je volný pohyb robotického ramene rukou do zvolené polohy a následné zaznamenání této polohy do proměnné v programu. Tím je usnadněna manipulace s ramenem, ale také se zjednodušuje programování, neboť operátor nemusí tolik přemýšlet, kterým kloubem ramene pohnout, či které souřadnice změnit, aby se dostal do cílové polohy. Nutno ovšem zmínit, že tato metoda je možná pouze v případě použití bezpečnostních prvků nebo s použitím speciálních kolaborativních manipulátorů, o nichž bude zmínka dále v práci.

1.3.4 Programovací jazyk manipulátorů

Z předchozího je jasné, že každý robotický manipulátor se programuje pomocí nějakého programovacího jazyku. Téměř každý výrobce průmyslových robotů si vyvíjí svůj vlastní programovací jazyk, který je optimalizovaný pro daný typ manipulátoru. Jedná se tedy o nejjednodušší a nejlepší volbu při programování manipulátoru, neboť je použit jeho vlastní programovací jazyk. Tabulka 1.2 zobrazuje některé výrobce manipulátoru s označením teach pendantu, řídicího systému a nadřazený programovací jazyk.

Tabulka 1.2 – Druhy programovacích jazyků

Manipulátor	Řídicí systém	Programovací jazyk	Teach pendant
Universal Robots	CB3.1	URSript	PolyScope
ABB	IRC5	RAPID	FlexPendant
Stäubli	CS9	VAL 3	SP2
KUKA	KUKA KR C4	KRL	KUKA smartPAD
FANUC	J-30*i*B	Karel	iPendant Touch

1.3.5 Programování ramene pomocí programovacích editorů

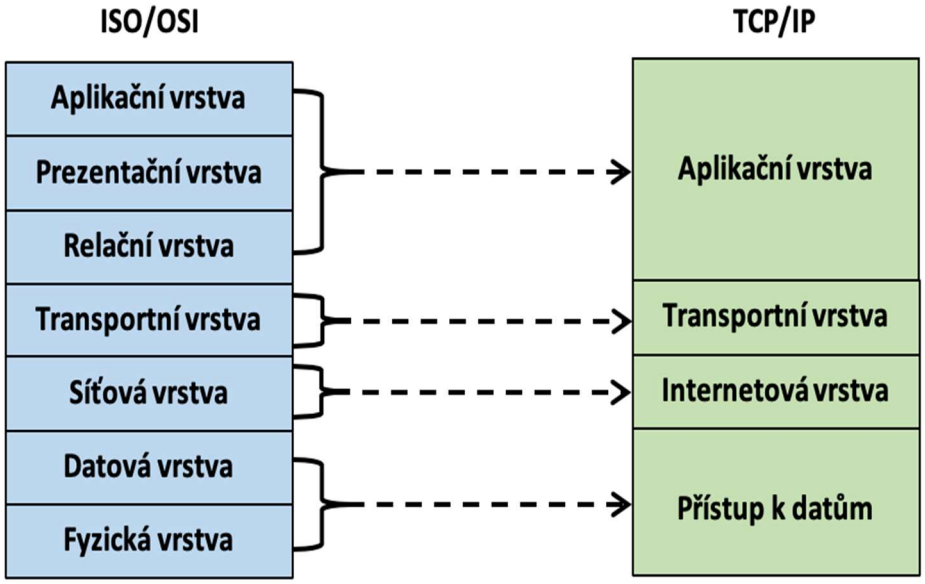
Klasické metody programování robotických ramen již byly zmíněny, ale většina manipulátorů nabízí ještě možnost ovládat a programovat rameno pomocí externích programovacích editorů. Tato možnost umožnuje vytvářet program nebo zasílat příkazy manipulátoru například z prostředí Matlab, Python atd. Tento způsob programování je založený na principu zasílání příkazů v jazyce manipulátoru pomocí síťového soketu. Principiálně se však jedná o vytvoření síťového soketu a nejčastěji je využit síťový komunikační protokol TCP/IP.

Síťový komunikační protokol TCP/IP

Komunikační protokol TCP/IP, viz obrázek 1.9, vychází z ISO/OSI modelu, a jedná se o množinu definovaných protokolů, přičemž jak již z názvu vyplývá, dominantními, ne však jedinými, protokoly jsou TCP (**T**ransmission **C**ontrol **P**rotocol) a IP (**I**nternet **P**rotocol). Síťové komunikační protokoly si lze jednoduše představit jako souhrn nějak standardizovaných pravidel, které se zařízení zavazují dodržovat za účelem možnosti použít či dlouhodobě

využívat nějaké služby. Dle obrázku 1.9 je vidět, že na rozdíl od ISO/OSI modelu, TCP/IP protokol předpokládá existenci pouze těchto čtyř vrstev:

- Aplikační vrstva – Tato vrstva zajišťuje přenos a srozumitelnost zpráv a využívá nejrůznější služby, mezi které patří například FTP (**F**ile **T**ransport **P**rotocol), který slouží k přenosu souborů ze vzdálených disků.
- Transportní vrstva – Transportní vrstva obsahující protokoly TCP a UDP slouží k navázání a ukončení spojení, ale také k zaručení celistvosti zprávy.
- Internetová vrstva – Na této vrstvě pracuje mimo jiné protokol IP, který se stará o rychlé doručení datagramů ke koncovému adresátovi.
- Vrstva pro přístup k datům – Zde dochází k přenosu rámců mezi dvěma propojenými počítači.



Obrázek 1.9 – ISO/OSI model a protokol TCP/IP

Síťový protokol TCP/IP je určen především ke komunikaci v heterogenních sítích, tedy v takových sítích, které mohou propojovat různorodé výpočetní systémy v různých částech datové sítě, a také mohou být používána různá komunikační prostředí. K adresaci v síti TCP/IP se používají tři typy adres, a to:

- MAC adresa – Adresa pevně přidělená každému adaptéru. Skládá se z části udávající informaci o výrobci a části udávající informace o daném adaptéru.

- IP adresa – Tato adresa v 32bitovém tvaru slouží k přesné identifikaci zařízení, které mají být předána data.
- Doménová adresa – Tato adresa je vlastně IP adresa, která odpovídá konkrétnímu webu a překlad IP adresy na doménová jména zajišťuje DNS (**D**omain **N**ame **S**ystem).

Za účelem jednoduché komunikace v TCP/IP síti je velmi často využíván model klient/server. Tato architektura od sebe odděluje klienta, který žádá o služby jiného uživatele zvaného server. Jednoduše si to lze představit tak, že server naslouchá na nějakém portu, a klient na tento port odesílá své požadavky (Beneš, 2014; Horák, 2011).

1.4 KOLABORATIVNOST

Ve dvou se to lépe táhne, toto tvrzení lze aplikovat snad ve všech oblastech života a vystihuje určitou míru spolupráce při nějaké činnosti. Spolupráci neboli kolaboraci lze definovat jako činnost s někým dalším, za účelem lehčího dosažení společných cílů. Tato myšlenka dala vzniknout takzvaným kolaborativním robotům neboli cobotům.

Coboti jsou speciální typ průmyslového manipulátoru, který umožnuje přímou interakci neboli spolupráci s člověkem v rámci definovaného kolaborativního pracovního prostoru. Takové spojení sil umožňuje ještě efektivnější pracovní výkon, neboť robot vyniká v přesných jednoduchých opakovaných činnostech, kdežto lidé disponují jedinečnými kognitivními schopnosti, které umožnují přizpůsobit úkoly náročným dynamickým změnám. Nutno ovšem dodat, že aby byla tato spolupráce efektivní, je nutné optimálně sdílet práci s manipulátorem, a to není vždy možné u každé činnosti.

1.4.1 Kolaborativní manipulátor a pracovní prostor

Aby tedy mohl robotický manipulátor spadat do speciální kategorie kolaborativních manipulátorů, musí disponovat mnoha různými bezpečnostními prvky a senzory a splňovat různé bezpečnostní normy. Právě tyto prvky, jako například proudové čidlo snímající náraz, jsou ty nejdůležitější části, které odlišují kolaborativní manipulátor od klasického průmyslového manipulátoru. Kolaborativní robot pracuje v definovaném kolaborativním pracovním prostoru, což je takový zabezpečený prostor, kde může člověk volně spolupracovat s manipulátorem během provozu, a na rozdíl od klasického průmyslového manipulátoru zde

nejsou zapotřebí ochranné klece či jiné omezení. Obrázek 1.10 znázorňuje možný způsob spolupráce cobota s člověkem (Maurtua, 2017).

Obrázek 1.10 – Kolaborace člověka s robotem (HOW YOU CAN USE COLLABORATIVE ROBOTS, 2019)

1.4.2 Bezpečnost kolaborativních robotů

Jak již bylo zmíněno, kolaborativní manipulátor musí splňovat různé normy, které definují, za jakých podmínek může do této skupiny manipulátor spadat. Jednou z norem zabývající se problematikou bezpečnosti cobotů je ISO/TS 15066, která definuje výkonová omezení a nejrůznější limitní hodnoty silového působení na člověka tak, aby se zabránilo zranění. Norma ISO/TS 15066, která vznikla v roce 2016, je doplňující norma k ISO 10218-1 a ISO 10218-2, definuje možnosti a rizika kolaborativní spolupráce s robotem a vymezuje takové limity, při kterých je spolupráce ještě bezpečná. Jinak řečeno, cílem je vytvořit takový manipulátor, aby v případě, že dojde ke kontaktu člověka s manipulátorem, nedošlo k lidskému zranění (Vojáček, 2019).

1.4.3 Modely kolaborativních robotů

Kolaborativní manipulátory jsou díky spolupráci s člověkem opravdu všestranné a díky normám a bezpečnostním prvkům dosti bezpečné stroje, které se v průmyslu objevují čím dál

tím více a není se čemu divit. Kolaborativní manipulátory se výborně hodí i na jemné manipulační úkony a dost často se používají při podávání mechanických dílu přímo člověku do ruky. Pro zajímavost a srovnání je zde tabulka 1.3, která porovnává podobná kolaborativní robotická ramena některých známých výrobců, které lze v dnešní době na trhu sehnat.

Tabulka 1.3 – Srovnání kolaborativních manipulátorů

Výrobce	Model	Počet os	Nosnost	Opakovatelnost pozice	Dosah	Hmotnost
Universal Robots	UR3	6	3 Kg	$\pm 0{,}1$ mm	500 mm	11 Kg
ABB	YuMi IRB 14000	7	0,5 Kg	$0{,}02$ mm	500 mm	38 Kg
Kuka	LBR iiwa 7 R 8000	7	7 Kg	$\pm 0{,}1$ mm	800 mm	24 Kg
Fanuc	CR-4iA	6	4 Kg	$\pm 0{,}01$ mm	550 mm	48 Kg

1.4.4 Kognitivní robotika

Z předchozího je zřejmé, že roboty a manipulátory jsou skvělými pomocníky při řešení běžných úkonů pro lidské potřeby v našem dynamickém světě. Roboty či coboty jsou však pořád jenom stroje ovládané člověkem, které poslouchají svůj program, a ne vždy dovedou řešit úkony tak optimálně, jak by je řešili lidé. Člověk vnímá svět kolem sebe, disponuje tedy kognitivními schopnostmi, pomocí kterých dokáže řešit i složité problémy a přizpůsobovat jim svoji činnost. Robot, který by se dokázal chovat tímto způsobem, by dokázal řešit problémy podobným způsobem jako člověk, a mohl tak ještě efektivněji pomáhat lidem. Touto problematikou se zabývá kognitivní robotika (Hlaváč, nedatováno).

Pod pojmem kognitivní robotiky si lze představit vědní obor, který kombinuje znalosti adaptivní robotiky, umělé inteligence a kognitivní vědy. Výsledkem této vědní disciplíny je vývoj speciálního typu robotů, jež se vyznačují určitou mírou inteligence, za účelem lepší, kvalitnější, a hlavně bezpečnější interakce s člověkem. Kognitivního robota si tedy lze jednoduše představit jako speciální typ kolaborativního robota, který nejenomže dokáže pracovat v těsné blízkosti člověka, ale také vnímá své okolí. Vnímáním svého okolí může robot

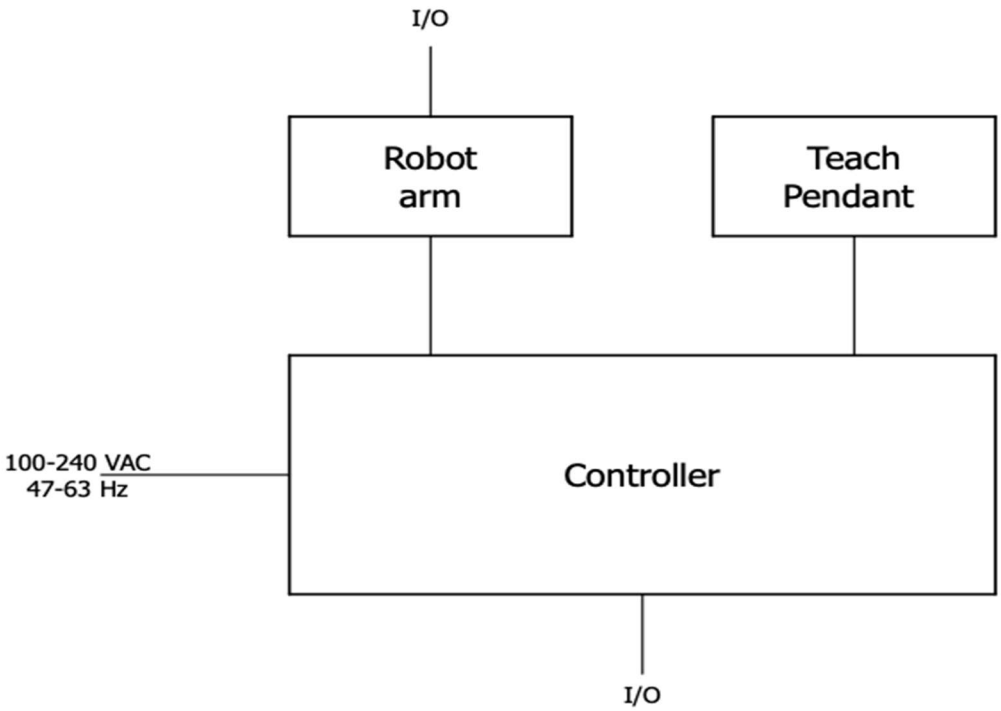
sledovat aktuální dění a na základě toho efektivně plánovat svoji činnost a také předvídat takové události, které ještě nenastaly. Na základě těchto událostí se robot učí, a z výsledné interakce si rovněž bere ponaučení.

Klíčovou vlastností kognitivních robotů, jsou jejich prediktivní schopnosti, které robotovi umožnují z množiny řešení v dané chvíli nalézt to optimální pro určitou činnost. Rovněž velmi slibná disciplína sloužící k učení a rozvoji těchto robotů jsou neuronové sítě. Kognitivní robotika není v průmyslu zatím skoro vůbec využívána, neboť se jedná o velmi složitý vědní obor, využívající umělé inteligence a složitých algoritmů, který vyžaduje ještě dlouhý vývoj (Hlaváč, nedatováno).

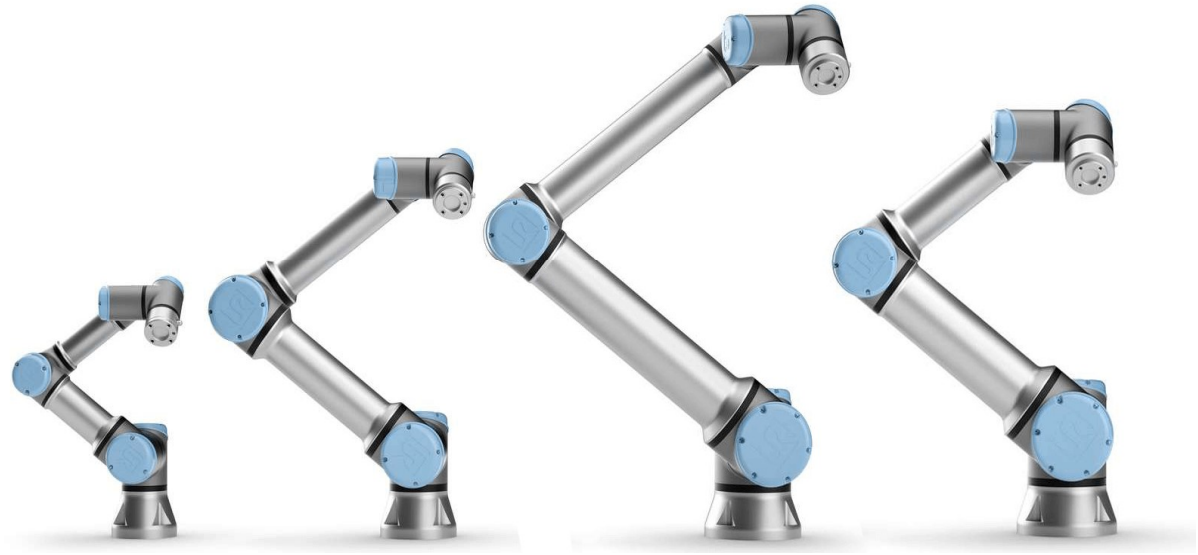
2 IMPLEMENTAČNÍ ČÁST

2.1 UNIVERSAL ROBOTS

Universal Robots je dánská společnost vyrábějící flexibilní univerzální kolaborativní průmyslové roboty neboli zkráceně coboty, o kterých byla zmínka v teoretické části práce. Jak lze vidět na obrázku 2.2, společnost má ve své nabídce celkem čtyři typy univerzálních šestiosých cobotů, které nesou označení ur3, ur5, ur10 a ur16. Číselné označení těchto cobotů vypovídá o nosnosti, a tudíž udává, jak těžkým předmětem jsou schopny manipulátory manipulovat. Rozměrově se jedná především o menší roboty určené k manipulaci spíše lehčích předmětů, ale zato se jedná o plně kolaborativní bezpečné univerzální roboty, které mají výborné pohybové vlastnosti a hodí se i pro náročnější úkony. Každé robotické pracoviště od Universal robots, jakožto i ostatních společností, by měla obsahovat řídicí jednotku, robotické rameno a teach pendant tak, jak je znázorněno na obrázku 2.1 (Universal Robots, 2020).



Obrázek 2.1 – Součásti robotického pracoviště (Universal Robots, 2019)

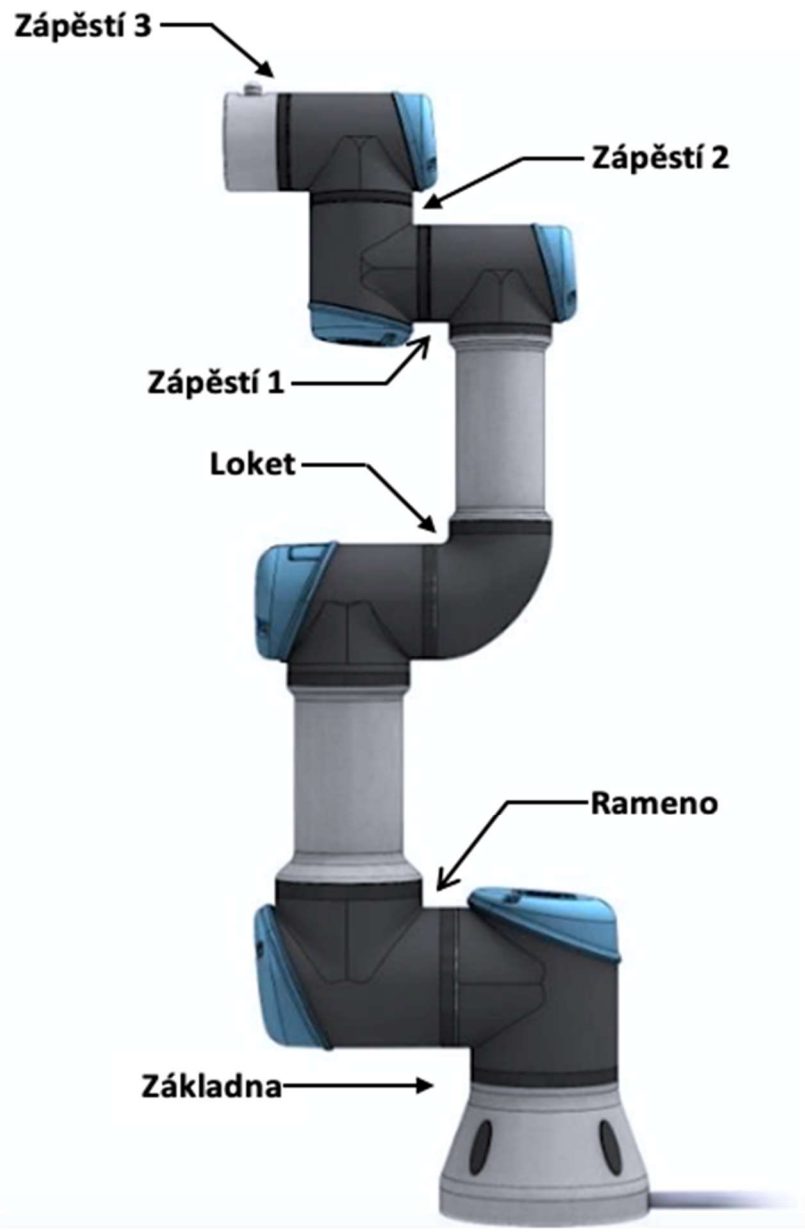


Obrázek 2.2 – Produktová nabídka společnosti Universal Robots (Universal Robots, 2020)

2.1.1 Kolaborativní robotické rameno UR3

Robotický manipulátor UR3 je nejmenším zástupcem společnosti Universal Robots a jedná se o šestiosý plně kolaborativní manipulátor. Již z názvu společnosti je zřejmé, že manipulátor spadá do skupiny univerzálních robotů a je vybavený šesticí rotačních kloubů, a tudíž šesti stupni volnosti, což umožnuje manipulátoru bezproblémový pohyb v prostoru. Jak již bylo zmíněno, rameno UR3, viz obrázek 2.3, se skládá z šesti kloubů, tedy základny, ramene, loktu a tří zápěstí, jejichž rotační možnosti a rychlosti jsou uvedeny v tabulce 2.1. V tabulce si lze rovněž všimnout, že rameno UR3 dokáže neomezeně rotovat s přírubou nástroje, což je schopnost, která u manipulátorů spadajících do této kategorie nebývá vždy samozřejmostí, ale výborně se hodí například u šroubovacích operací.

Celková hmotnost manipulátoru UR3 je 11 kg a dovede manipulovat s objekty, vážícími do 3 kg. Montáž ramene je libovolná a díky relativně nízké hmotnosti lze robota montovat i na pracovní stůl, což může být někdy výhodné. Pracovní dosah ramene je 500 mm od kloubu základny a při montáži je nutné dbát na válcový prostor nad a pod základnou manipulátoru. Opakovatelnost polohy je u UR3 $\pm 0{,}1$ mm a tento údaj je při výběru robota důležitý, neboť udává odchylku mezi polohami při opakovaném pohybu jedné polohy (Universal Robots, 2020).



Obrázek 2.3 – Klouby ramene UR3 (Universal Robots, 2019)

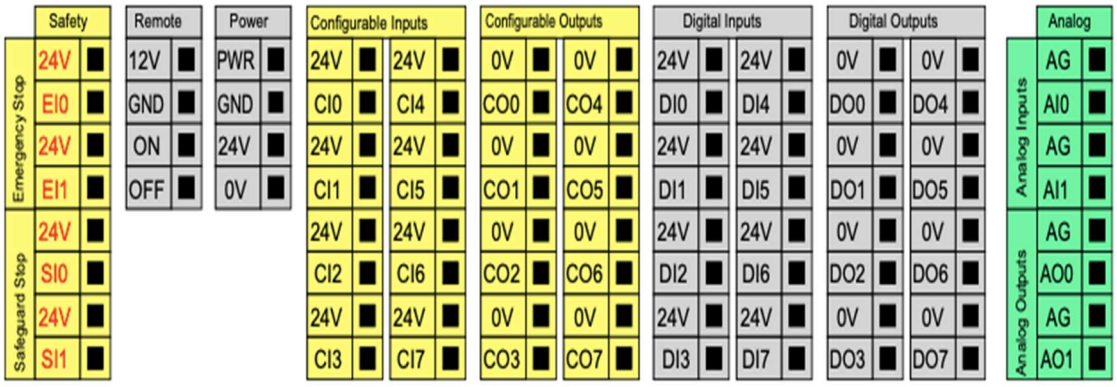
Tabulka 2.1 – Kloubové možnosti otáčení a rychlosti

Umístění kloubu	Rozsah otáčení		Rychlost kloubů
	Minimální	Maximální	
Základna	-360°	360°	$\pm 180°s^{-1}$
Rameno	-360°	360°	$\pm 180°s^{-1}$
Loket	-360°	360°	$\pm 180°s^{-1}$
Zápěstí1, 2	-360°	360°	$\pm 360°s^{-1}$
Zápěstí 3 (příruba nástroje)	Neomezené		$\pm 360°s^{-1}$

2.1.2 Řídicí kontrolér CB 3.1

Řídicí kontrolér robotických ramen společnosti Universal Robots je znázorněný na obrázku 2.4 a nese označení CB 3.1. Jedná se o kompaktní skříň, jež slouží k napájení robotického ramene, teach pendantu, připojení digitálních/analogových vstupů a výstupů, které lze využít k připojení například PLC automatů a jiných zařízení. Obrázek 2.4 rovněž zobrazuje teach pendant manipulátoru zvaný PolyScope. Řídicí kontrolér CB 3.1 disponuje vícero digitálními/analogovými vstupy a výstupy, které jsou barevně rozlišeny podle toho, o jaký vstup či výstup se jedná, viz obrázek 2.5. Zelená barva indikuje analogové vstupy a výstupy, šedou barvou jsou označeny víceúčelové digitální vstupy a výstupy, které lze napájet z vnitřního zdroje 24 V. Žlutou barvou s černým textem jsou označeny konfigurovatelné vstupy a výstupy, které lze programovat a žlutá barva s červeným textem pak označuje bezpečnostní „safety" signály, určené například pro nouzové zastavení robota. Konfigurovatelným vstupům a výstupům lze v GUI přiřadit úlohu bezpečnostních (Universal Robots, 2020).

Obrázek 2.5 – Řídicí jednotka robota UR3 (Universal Robots, 2020)



Obrázek 2.4 – Digitální, analogové vstupy a výstupy (Universal Robots, 2019)

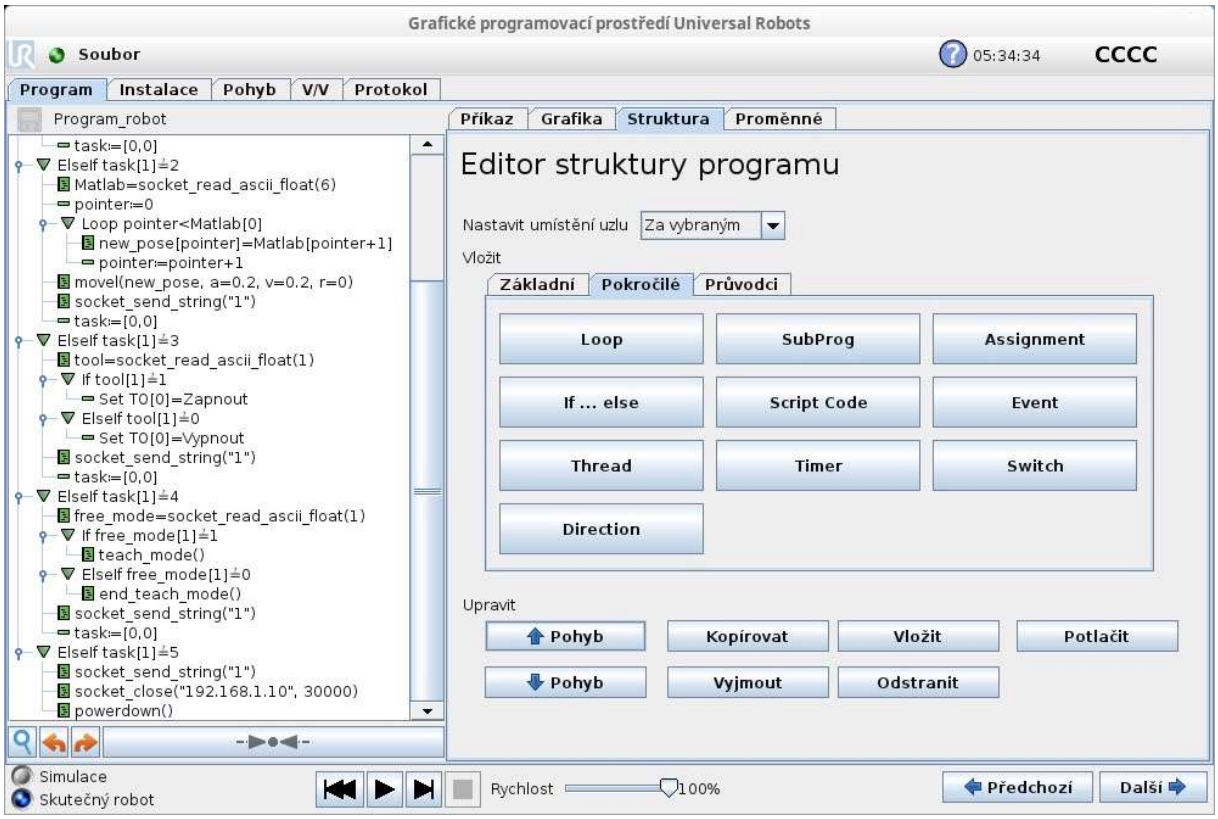
2.1.3 Programovací jazyk a PolyScope

Robotické manipulátory od Universal Robots jsou ovládány a programovány přímo z teach pendantu, který po spuštění uživateli nabídne programové prostředí zvané PolyScope, jehož úvodní obrazovka je znázorněná na obrázku 2.6. Toto programové grafické uživatelské prostředí neboli GUI lze jednoduše obsluhovat na plně dotykové dvanáctipalcové obrazovce připojené k ovládací jednotce robota. PolyScope je všestranný nástroj, jenž mimo programování a ovládání pohybu ramene slouží také k nastavení nejrůznějších funkcí robota, mezi které patří mimo jiné i změna jazyku, nastavení sítě či aktualizaci systémového softwaru. V prostředí PolyScope lze rovněž konfigurovat vstupně výstupní zařízení, zobrazit protokol aktuálního stavu robota či měnit a vizualizovat pohyb ramene a mnoho dalšího. Zajímavou funkcí robotů Universal Robots je režim volného pohybu ramene, který odbrzdí klouby ramene a umožnuje tak pohybovat ramenem volně rukou. Tento režim lze mimo jiné zapnout v pohybové záložce v prostředí PolyScope.

Obrázek 2.6 – Úvodní obrazovka prostředí PolyScope

Kolaborativní robotické rameno UR3, jak už bylo naznačeno v tabulce 1.2, při své funkci využívá svůj vlastní programovací jazyk zvaný URScript. Tento skriptovací programovací jazyk je určen k programování všech robotů Universal Robots, který obsahuje proměnné, datové typy a příkazy sloužící k tvorbě programu a řízení robotického ramene.

Při snaze vytvořit program pro rameno má uživatel na výběr z několika možností. První z možností již byla zmíněna a jedná se o nejjednodušší variantu, kterou je tvorba programu přímo v teach pendantu robota. Obrázek 2.6 ukazuje možnost volby „Naprogramovat robota“ na úvodní obrazovce grafického uživatelského prostředí PolyScope. Po zvolení této volby se uživateli zobrazí okno pro přímou tvorbu programu pomocí příkazů a funkcí jazyka URScript, viz obrázek 2.7.



Obrázek 2.7 – Možnost tvorby programu v PolyScope

Za účelem tvorby programu pro manipulátor lze využít jakéhokoliv textového editoru, ale je nutné dodržet syntaxi URScript. Takto napsaný program lze jednoduše přenést na externí zařízení a nahrát přímo do teach pendantu robota. Rovněž lze využít metody off-line programování, o nichž byla zmínka v pododdílu 1.3.2. Off-line programování je v dnešní době velmi oblíbená metoda, neboť většina výrobců robotických manipulátorů má svůj vlastní software pro off-line tvorbu programů a simulací. V případě manipulátoru společnosti

Universal Robots, se jedná o software s názvem URSim, který bude popsán v pododdílu 2.1.4. Další velice elegantní způsob ovládání ramene a tvorby programu je zasílání příkazu URScript vzdáleně z počítače. Tento způsob programování bude rovněž popsán dále v práci.

2.1.4 OFF-LINE SIMULÁTOR URSim

Jak již tedy bylo zmíněno, manipulátory společnosti Universal Robots využívají software zvaný URSim, jenž je, mimo jiné, určený k tvorbě programů off-line a k simulaci při plánovaní pohybů. URSim je volně dostupný software, který lze zadarmo stáhnout z oficiálních stránek společnosti Universal Robots a jednou z jeho hlavních předností je, že se vlastně jedná o totožné prostředí, jaké zná uživatel z teach pendantu robota. Tím je docíleno, že se uživatel nemusí přizpůsobovat novému prostředí, ale může na počítači pracovat tak, jak je zvyklý z teach pendantu. Po spuštění simulátoru se tedy uživateli nabídne úvodní obrazovka grafického uživatelského prostředí PolyScope, která je vyobrazená na obrázku 2.6.

Pomocí programu URSim lze využívat téměř veškerých funkcí, jako při použití teach pendantu a reálného manipulátoru včetně tvorby programu, jenž je ukládán ve formátu „soubor.urp“. Následně lze hotový program zkopírovat na přenosné médium a v případě potřeby nahrát soubor do teach pendantu robota. URSim rovněž umožnuje off-line ověření správnosti programu pomocí simulátoru, což je velmi užitečný nástroj, neboť lze tímto způsobem například odladit chyby programu pohodlně v počítači.

2.1.5 Instalace URSim

Již byla zmínka o užitečnosti a velkých výhodách softwaru URSim, ale při použití uživatel zjistí, že program je dostupný pouze pro operační systémy Linux. Pro využití lze tedy program nainstalovat na jakoukoliv distribuci Linuxu, nebo lze využít softwarů pro virtualizaci operačního systému a z oficiálních stránek Universl Robots stáhnout právě tuto verzi. Pro zaručení plynulého a bezchybného fungování programu se však doporučuje instalovat URSim přímo na operační systém Linux. Instalace samotného simulátoru není nijak náročná a postup spočívá v dekompresi staženého souboru a zadání těchto tří příkazů v terminálu:

Příkazy v terminálu

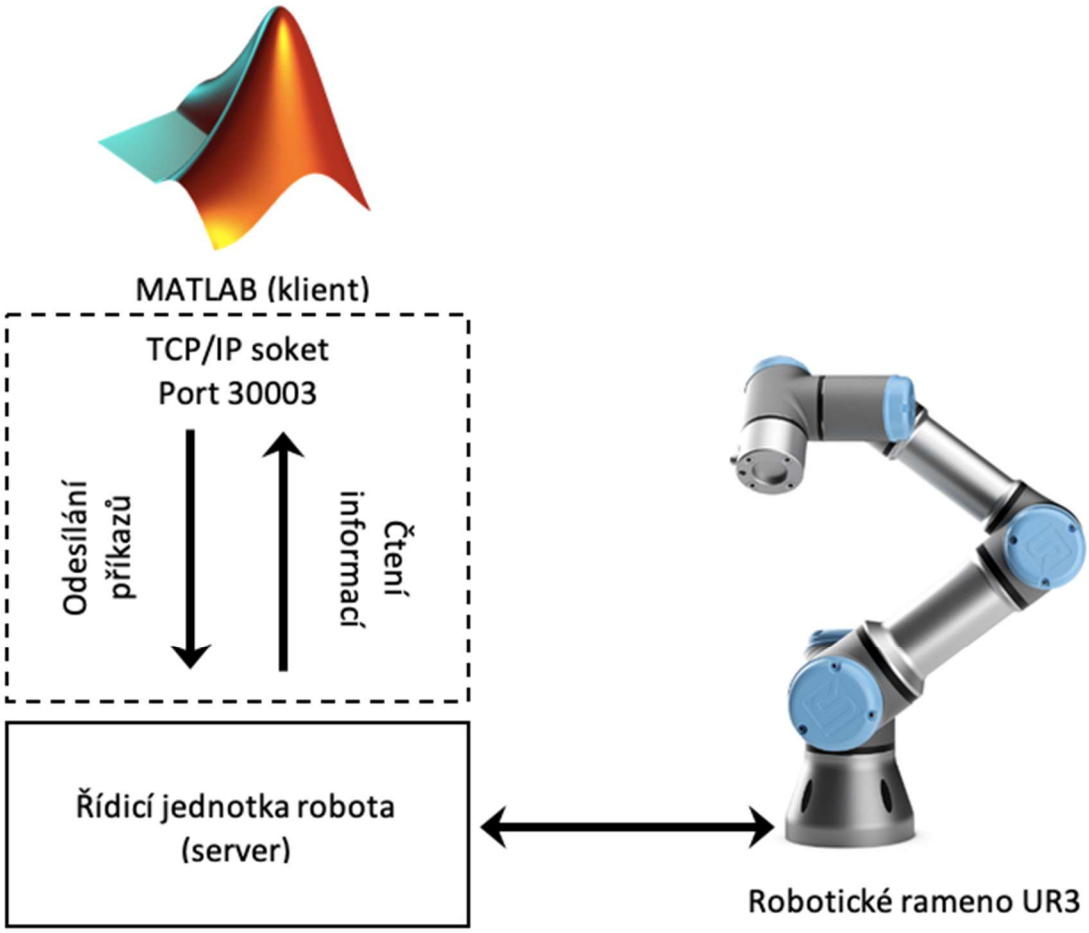
```
cd ursim-3.12.0.9
./install.sh
./start-ursim.sh
```

Pro správnou funkci takto zadaných příkazů je zapotřebí, aby se dekomprimovaný adresář ursim-3.12.0.9 nacházel v domovském adresáři. Následně je prvním příkazem docíleno přepnutí do adresáře obsahující instalační soubor a druhým příkazem je spuštěna instalace programu. Zadání třetího příkazu do terminálu umožnuje program spustit. Rovněž lze vytvořit odkaz na ploše a spouštět program jednodušeji.

2.2 PROGRAMOVÁNÍ RAMENE Z POČÍTAČE

Robotické rameno UR3 lze ovládat a programovat i vzdáleně z počítače pomocí síťového soketu. Tento způsob programování nabízí nesčetně výhod, neboť lze z jednoho hostitelského počítače ovládat či programovat rameno, a zároveň využívat pokročilé počítačové programy za účelem plánování trasy, simulace a další možnosti počítače, jako například internetu. Jelikož má být robot řízen z počítače vzdáleně a dochází tak k částečné eliminaci teach pendantu a PolyScopu, je zapotřebí, aby uživatel dokázal od robota získat užitečné informace, které jsou nezbytné k řízení. Jedná se zejména o aktuální polohy kloubů, stav nástroje apod. Tyto informace robot umí posílat, ale je nutné je dekódovat a vhodně využít, princip bude vysvětlen v následujících pododdílech.

Princip vzdáleného programování z počítače znázorňuje obrázek 2.8, v podstatě se jedná o vytvoření síťového soketu a následně o navázání oboustranného komunikačního toku informací mezi počítačem a řídicí jednotkou robota. Nestačí však informace od manipulátoru pouze získávat, ale také je třeba informace neboli příkazy robotu zasílat tak, aby je uměl číst a byl schopen provádět požadavky uživatele. Je tedy velmi důležité, aby uživatel znal a ovládal syntaxi jazyka URScript a věděl jaké chování robota lze od daných zasílaných příkazů očekávat. Jak již bylo zmíněno, vzdálené ovládání ramene a komunikační tok informací je realizováno pomocí síťového soketu a komunikačních protokolů TCP/IP.



Obrázek 2.8 – Princip vzdáleného ovládání ramene z prostředí MATLAB

2.2.1 Klientské rozhraní robota

Aby bylo možné ovládat robotické rameno tak, jak je znázorněno na obrázku 2.8, je zapotřebí, jak již bylo zmíněno v úvodu tohoto oddílu, zajistit obousměrný tok informací. Robotické rameno UR3 a jeho řídicí systém nabízí za tímto účelem několik klientských rozhraní dostupných na různých portech. Tato klientská rozhraní jsou určená pro vzdálenou interakci se softwary třetích stran, jako například Matlab, Python a jiné. Rameno UR3 má k dispozici celkem čtyři klientské rozhraní a to:

- Primární klient – Klientské rozhraní dostupné na portu 30001 určené k přenosu informativních dat o stavu robota a dalších zpráv. Tento klient dokáže přijímat a aktualizovat data s frekvencí 10 Hz. To znamená, že příkazy URScript odesílané robotovy pomocí primárního klienta dokáže přijímat, každých 100 ms tj. 0,1 s.

- Sekundární klient – Sekundární klientské rozhraní dostupné na portu 30002 je stejně jako v případě primárního rozhraní určeno k přenosu dat informujících o stavu robota, ale nikoli zpráv. Přenosová frekvence sekundárního rozhraní je stejná jako u primárního klienta 10 Hz a obě rozhraní umožnují vzdálené ovládání ramene prostřednictvím příkazů URScriptu.
- Real-time klient – Klientské rozhraní pracující v reálném čase je dostupné na portu 30003 a chováním je velmi podobné primárnímu a sekundárnímu rozhraní. Velkým rozdílem je vyšší frekvence, s níž umožnuje klient přijímat příkazy a měnit stav robota. Rozhraní reálného času disponuje frekvencí 125 Hz, tedy aktualizace stavu robota je pravidelně každých 8 ms, tj. 0,008 s. Tento klient na rozdíl od prvních dvou umožnuje navíc ještě odesílat některé důležité informace poskytované řídicí jednotkou robota a toho lze dále využívat, viz pododdíl 2.2.3.
- Klient RTDE – Klientské rozhraní RTDE dostupné na portu číslo 30004 slouží k výměně dat v reálném čase. Vlastně se jedná o velice podobné rozhraní jako v případě reálného času, ale RTDE umožnuje navíc i synchronizovat externí aplikace s řadičem, aniž by došlo k narušení vlastností řadiče. Tento klient rovněž umožnuje interakci s ovladači nebo ovládání vstupů a výstupy robota. RTDE klient pracuje stejně jako klient reálného času s frekvencí 125 Hz.

2.2.2 Rozhraní pro práci v reálném čase

Z předchozího pododdílu 2.2.2 je zřejmé, že řídicí systém robota má k dispozici celkem čtyři klientská rozhraní určená pro vzdálenou interakci s robotem. Softwarové řešení vzdáleného ovládání ramene z počítače popsané dále využívá rozhraní určené pro práci v reálném čase, neboť tento klient se na tuto úlohu výborně hodí a disponuje velkými výhodami. Klientské rozhraní pracující v reálném čase, jak již bylo zmíněno, je dostupné na portu 30003 a pracuje s frekvencí 125 Hz, což umožňuje zasílání URScript příkazů do řídicího systému každých 8 ms.

Řídicí systém navíc prostřednictvím tohoto klienta s frekvencí 125 Hz pravidelně posílá nejrůznější informace o aktuálním stavu robota, což je výhoda oproti primárnímu či sekundárnímu klientu a lze toho patřičně využít pro vzdálené řízení z počítače. Mezi pravidelně poskytované informace od řídicího systému robota jsou například aktuální pozice TCP, úhly natočení jednotlivých kloubů robota, teplota jednotlivých kloubů ve stupních Celsia a mnoho

dalších užitečných informací. Tabulka 2.2 zobrazuje některé důležité informace, které lze od řídicího systému dostat a vyčíst prostřednictvím soketu.

Tabulka 2.2 – Informace poskytované real-time klientem o aktuálním stavu robota

Význam zprávy	Datový typ	Počet očekávaných hodnot	Velikost zprávy v bajtech	Řádky
Velikost obdržené zprávy v bajtech	Integer	1	4	
Aktuální natočení kloubů	Double	6	48	32–37
Aktuální rychlosti kloubů	Double	6	48	38-43
Aktuální kartézské souřadnice TCP	Double	6	48	56-61
Aktuální stav digitálních vstupů	Double	1	8	86
Teplota jednotlivých kloubů	Double	6	48	87-92
Elektrické napětí kloubů	Double	6	48	125-130
Aktuální stav digitálních výstupů	Double	1	8	131
Bezpečnostní status (Safety Status)	Double	1	8	139

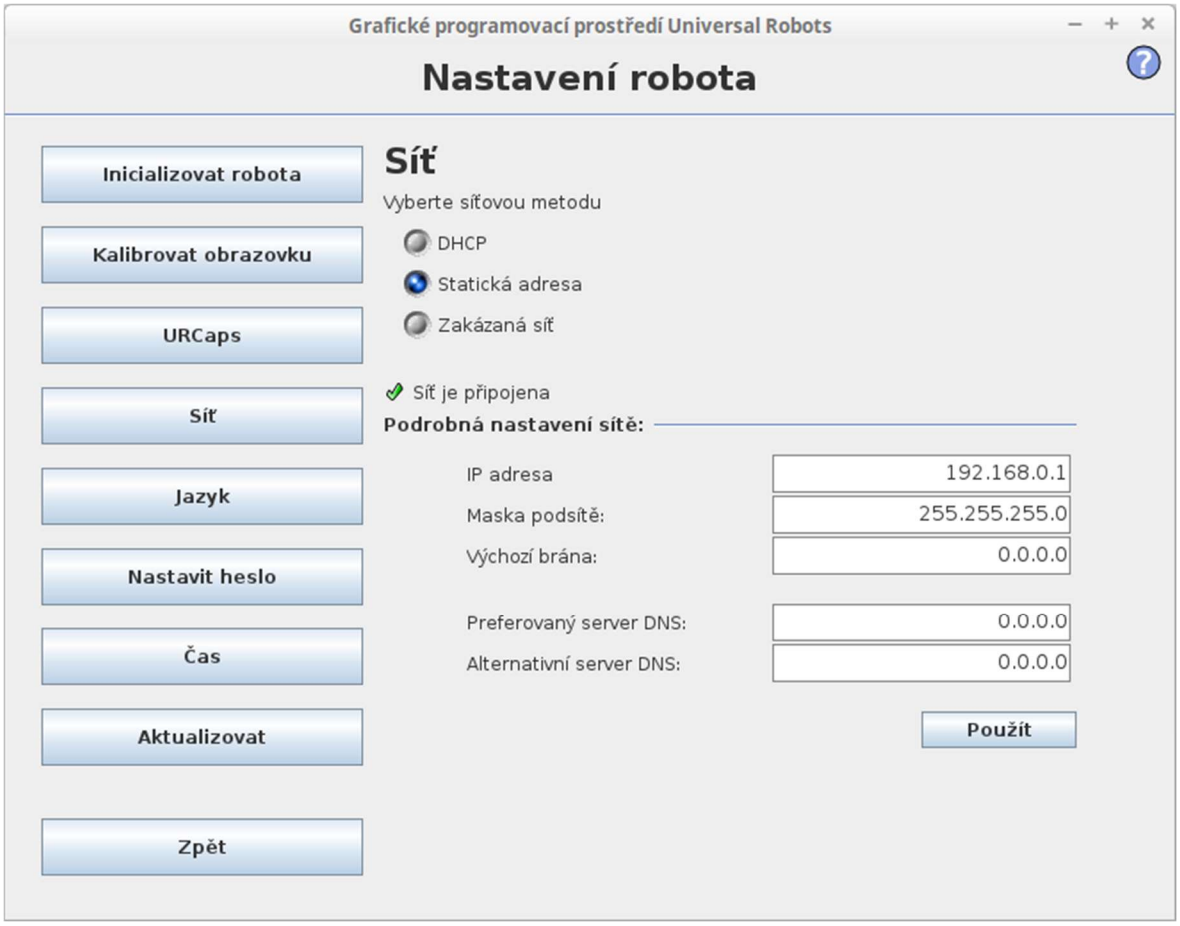
Tabulka 2.2 výše obsahuje jenom některé ze základních informací a celkem lze takto od řídicího systému prostřednictvím soketu pravidelně každých 8 ms vyčíst 1116 bajtů a obdržet tak celkem 140 hodnot, které lze následně v programu MATLAB ukládat a dále zpracovat. Způsob výčtu informací ze soketu a následná úprava v programu MATLAB bude dále v práci ukázána v oddílu 2.3.

2.3 SOFTWAROVÉ ŘEŠENÍ

Softwarové řešení pro vzdálené ovládání robotického ramene, které je níže popsáno, využívá způsobů a možností robota popsaných v implementační kapitole výše. Za účelem komunikace mezi hostitelským počítačem a řídicím kontrolérem ramene UR3 je využito síťového soketu pomocí protokolu TCP/IP a architektury klient/server. Pro zasílání příkazů a čtení informací od ramene je využit klient určený pro práci v reálném čase, kvůli výhodám a možnostem zmíněných v pododdíle 2.2.3.

2.3.1 Komunikace ramene s počítačem

Prvním a velmi důležitým krokem je vytvoření síťové komunikace mezi ramenem UR3 a počítačem, lépe řečeno Matlabem. Zde je zapotřebí zjistit, jakou IP adresu má robot přidělenou, popřípadě nemá-li žádnou adresu nastavenou, pak je potřeba ji nastavit. Toho lze docílit v PolyScope výběrem možnosti „nastavení robota“ a „síť“, viz obrázek 2.9.



Obrázek 2.9 – Síťová nabídka PolyScope

Následně je potřeba přepnout počítač do stejné sítě, ve které se robot nachází. Má-li robot například IP adresu 192.168.1.1 a masku 255.255.255.0 pak by počítač měl mít IP adresu například 192.168.1.2 a masku 255.255.255.0. Příkaz pro vytvoření soketu v programu Matlab je následující:

Kód v Matlabu

```
Robot_IP = '192.168.1.1';
Port = 30003;
socket = tcpip(Robot_IP, Port, 'NetworkRole', 'client');
fopen(socket);
fclose(socket);
```

Příkaz `tcpip` v Matlabu umožnuje vytvořit objekt TCP/IP komunikace, jenž je následně uložený do proměnné `socket`. Parametry příkazu `tcpip` jsou IP adresa robota, číslo portu, které jak již bylo zmíněno, patří klientu pracujícího v reálném čase. V parametru příkazu lze rovněž jednoduše nastavit počítači roli klienta, jenž bude moct posílat příkazy a přijímat informace od řídicí jednotky robota, který zastává funkci serveru. Jelikož je objekt TCP/IP komunikace uložený v proměnné `socket`, lze viz čtvrtý a pátý příkaz soket otevřít a v případě potřeby zase zavřít.

2.3.2 Princip vyčítání informací ze soketu

Existuje-li aktivní síťové spojení mezi řídicí jednotkou robota a počítačem, lze zahájit čtení informací od klienta reálného času dostupném na portu 30003. Jelikož uživatel přesně ví, jak se klient reálného času chová a také kolik informací posílá, viz pododdíl 2.2.3, lze upravit kód v Matlabu tak, aby bylo přesně definováno, kolik bajtů se očekává. Předchozí příkaz v Matlabu lze upravit takto:

Kód v Matlabu

```
socket = tcpip(Robot_IP, Port, 'NetworkRole', 'client',...
'InputBufferSize', 1116*5, 'BytesAvailableFcnCount', 1116, ...
'BytesAvailableFcnMode', 'byte', 'BytesAvailableFcn', @cti);
fopen(socket);
```

Nyní je kromě IP adresy, portu klienta a role, také definována velikost vstupní vyrovnávací paměti v bajtech, počet bajtů, které musí být ve vyrovnávací paměti, aby se vygenerovala daná událost a také volání funkce, kde je definováno, co se má provést po obdržení daného počtu bajtů. Volaná funkce má následující tvar:

Kód v Matlabu pro vyčítání informací ze soketu

```
function cti(socket, event)
delka = fread(socket, 1, 'Uint32');
zprava = fread(socket, 139, 'double');
if delka ~= 1116
    disp('Porušení synchronizace zpráv')
end
socket.UserData.zrprava = zprava;
```

Vstupními parametrem funkce `cti` je objekt TCP/IP komunikace. Následně je vytvořená proměnná `delka`, do které je pomocí příkazu `fread` vyčtena první příchozí informace ze soketu. Parametry příkazu `fread` se vyplní podle tabulky 2.2, ze které vyplývá, že se očekává jedna hodnota datového typu integer. Hodnota uložená v proměnné `delka` by ideálně měla být rovna hodnotě 1116, což poukazuje na přijetí 1116 bajtů informací. Dle pododdílu 2.2.3 je v těchto 1116 bajtech uchováno 140 zpráv, a jelikož první už je přečtena, v další proměnné zvané `zprava` je ze soketu získáno a uloženo zbylých 139 zpráv datového typu double. Následně je pomocí jednoduché podmínky ověřeno, zda bylo obdrženo 1116 bajtů a v případě že ne, je uživatel informován o přerušení synchronizace. Tímto způsobem je umožněno přijímat a aktualizovat různé informace od řídicí jednotky robota, a to pravidelně každých 8 ms.

2.3.3 Zasílání příkazů

Za předpokladu, že má uživatel k dispozici potřebné informace o aktuálním stavu ramene, čehož je docíleno způsobem popsaným v pododdílu 2.3.2, lze využít syntaxe URScript a zasílat manipulátoru nejrůznější příkazy vzdáleně. Princip vzdáleného odesílání příkazů je velmi jednoduchý a v podstatě se jedná o prosté zasílání textového řetězce po síťovém spojení tak, aby robot obdržel příkaz, kterému rozumí, a tudíž ho je schopen provést. Odesílané příkazy přijímá a zpracovává řídicí jednotka robota, lépe řečeno klient určený pro práci v reálném čase

aktivní na portu 30003 každých 8 ms. Způsob odesílání příkazů a ovládání ramene z Matlabu je následující:

Kód v Matlabu pro zasílání příkazů

```
fprintf(socket, 'Příkaz_1 v URScriptu');
pause(0,5);
fprintf(socket, 'Příkaz_2 v URScriptu');
```

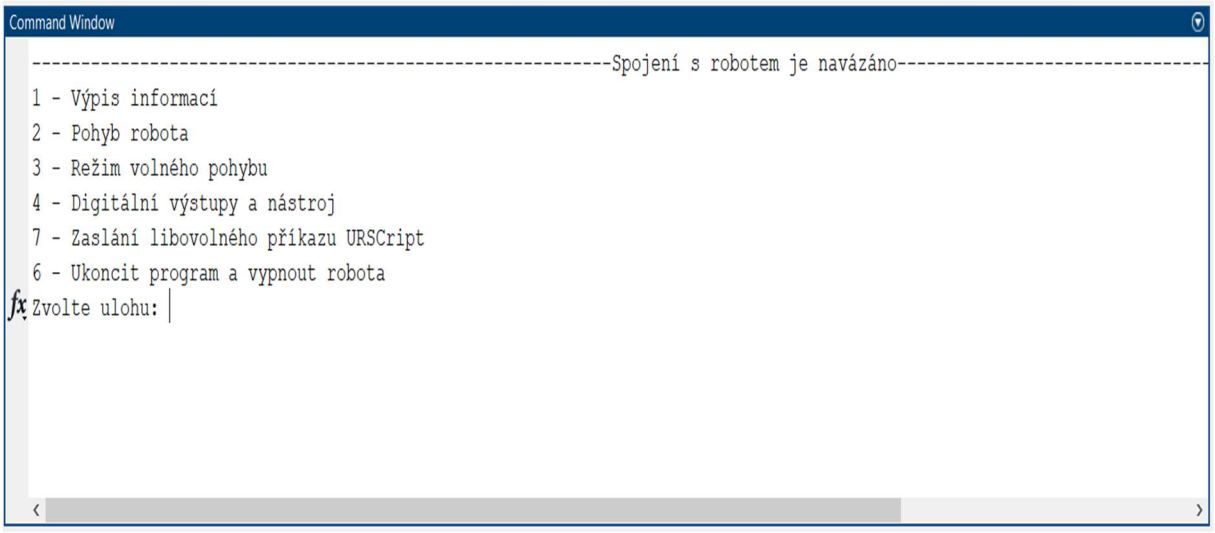
Již na první pohled je zřejmé, že se jedná o mnohem jednoduší úkol než v předchozím případě, kdy bylo třeba ze soketu informace číst. Tento jednoduchý příkaz pouze zapisuje textový řetězec do objektu TCP/IP, ale pro tuto úlohu je tento způsob dostačující. Odešle-li operátor tímto způsobem příkaz pro nějakou interakci s ramenem, řídicí jednotka tento příkaz přijme a za předpokladu, že je syntakticky v pořádku, je příkaz zpracován a manipulátor provede svůj úkol. Klient pracující v reálném čase má však jednu obrovskou nevýhodu a tou je, že nedává zpětnou informaci o stavu provedení příkazu. Operátor tak není o provedení, dokončení či úspěšnosti příkazu nijak informován. Zde je možné, a většinou i nezbytné, využít řešení popsané v předchozím pododdílu 2.3.2 a každých 8 ms vyčítat soket pro zjištění aktuálního stavu manipulátoru.

Z omezení klienta určeného pro práci v reálném čase vyplývá, že uživatel, jenž chce příkazy manipulátoru zasílat, si musí uvědomit, že ne každý příkaz má smysl tímto způsobem posílat. Příkaz, jenž nemá smysl manipulátoru zasílat, je takový, který slouží k získání nějaké informace, jako je tomu například u „get_actual_tcp()“. Kdyby operátor soketem poslal řídicí jednotce manipulátoru tento příkaz, nedostal by požadovanou návratovou odpověď tohoto příkazu, ale pouze již zmíněných 1116 bajtů informací. Tyto přijaté informace sice shodou okolností obsahují i návratovou informaci výše uvedeného příkazu, ale jak již bylo zmíněno, řídicí jednotka robota posílá tyto informace pravidelně a nezávisle na odesílaných příkazech s frekvencí 125 Hz. Je tedy zbytečné například tento typ příkazů soketem posílat, neboť to nemá smysl.

2.4 UKÁZKOVÉ APLIKACE

V oddílu 2.3 byl vysvětlen princip fungování soketové komunikace mezi počítačem a robotickým ramenem UR3, a to včetně čtení příchozích informací od řídicí jednotky robota, ale také zasílání příkazu URScriptu z počítače. Tento oddíl, respektive pododdíly obsahují ukázky

konkrétních aplikací v prostředí Matlab, a to jak pro vyčítání konkrétních informací, jako například úhlů natočení kloubů, kartézských souřadnic TCP apod., ale také způsob zasílání některých příkazů pro robota. Pro lepší představu je vytvořena jednoduchá aplikace, která běží v cyklu a nabízí uživateli na výběr ze základních funkcionalit. Čtenáři jsou zde ukázány obrázky zachycující obrazovku Matlabu při používání této aplikace. Program, jak lze vidět na obrázku 2.10, nabídne po spuštění uživateli různé interakce s ramenem.



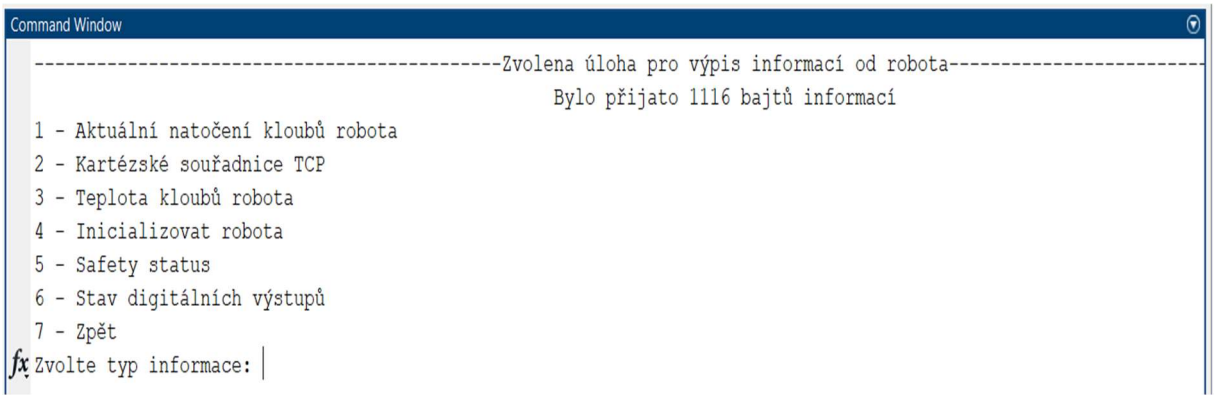
Obrázek 2.10 – Úvodní nabídka programu

2.4.1 Čtení a zpracování informací

Za účelem vyčítání informací ze soketu je vždy využita funkce `cti`, která je popsána a ukázána v pododdílu 2.3.2. Uvedeny a popsány jsou zde pouze informace popsané v tabulce 2.2, neboť se jedná o výběr z těch nejdůležitějších informací, které lze soketem přijmout a zpracovávat. Po spuštění aplikace v Matlabu a výběru první možnosti se zobrazí nabídka výpisu nejrůznějších informací, které jsou ze soketu vyčítány, viz obrázek 2.11.

Velikost obdržené zprávy

Jak již bylo zmíněno a uvedeno v tabulce 2.2 a také při popisu funkce `cti`, první přijatý řádek ze soketu obsahuje informaci o tom, kolik bajtů informací bylo přijato. Jak již bylo uvedeno dříve, je zřejmé, že by vždy mělo být přijato 1116 bajtů, a to bez ohledu na posílané příkazy. Informace o velikosti zprávy lze získat pomocí funkce `cti` ukázané v pododdílu 2.3.2 a důkaz o přijetí správného počtu bajtů informací ukazuje obrázek 2.11.



Obrázek 2.11 – Nabídka výpisu informací

Aktuální natočení kloubů ramene

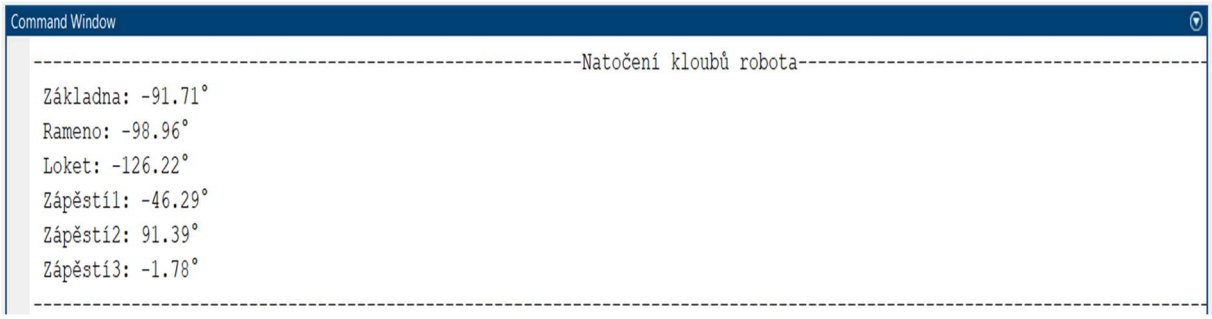
Informace o aktuálním natočení kloubů je u angulárního typu manipulátorů jednou z nejdůležitějších informací, které je potřeba sledovat. Aktuální úhlové natočení kloubů je rovněž důležité pro kloubový pohyb ramene, jenž je považován za nejúčinnější a nejrychlejší pohyb používaný při plánování pohybů manipulátoru v kloubových souřadnicích. Za normálních okolností by pro získání této informace mohl operátor využít například příkaz jazyka URScript „get_actual_joints_positions()“, který by jako návratovou informaci vrátil úhly natočení jednotlivých kloubů v radiánech. Jak ale již bylo zmíněno, řídicí jednotka manipulátoru na tento příkaz odeslaný soketem nebude z našeho pohledu nijak reagovat, a tudíž se operátor o této informaci nic nedoví. Možné řešení již bylo zmíněno v pododdílu 2.3.3 a skript pro výčet kloubových proměnných v prostředí Matlabu je následující:

Kód v Matlabu pro výčet aktuálních kloubových proměnných

```
Uhly_kloubu = (socket.UserData.zprava(32:37))*180/pi;
disp('Natočení kloubů ramene je: );
fprintf('Základna: %1.2f°\n Rameno: %1.2f°\n Loket: %1.2f°\n
Zápěstí1: %1.2f°\n Zápěstí2: %1.2f°\n Zápěstí3: %1.2f°\n',
Uhly_kloubu (1), Uhly_kloubu (2), Uhly_kloubu (3), Uhly_kloubu
(4), Uhly_kloubu (5), Uhly_kloubu (6));
```

Z použitého kódu výše je zřejmé, že pro výčet kloubových proměnných se volá `UserData.zprava` a konkrétně se čte řádek 32 až 37, viz tabulka 2.2. Vyčtené informace se uloží do proměnné `Uhly_kloubu`, jenž obsahuje šest číselných hodnot neboli šest kloubových úhlů v radiánech. Následně je pro lepší představu každá hodnota násobena podílem

$180/\pi$ a převedená na úhly ve stupních. Další řádky kódu slouží pro přehledný výpis dané informace na obrazovku uživatele, viz obrázek 2.12.



Obrázek 2.12 – Výpis kloubových proměnných

Aktuální kartézské souřadnice TCP

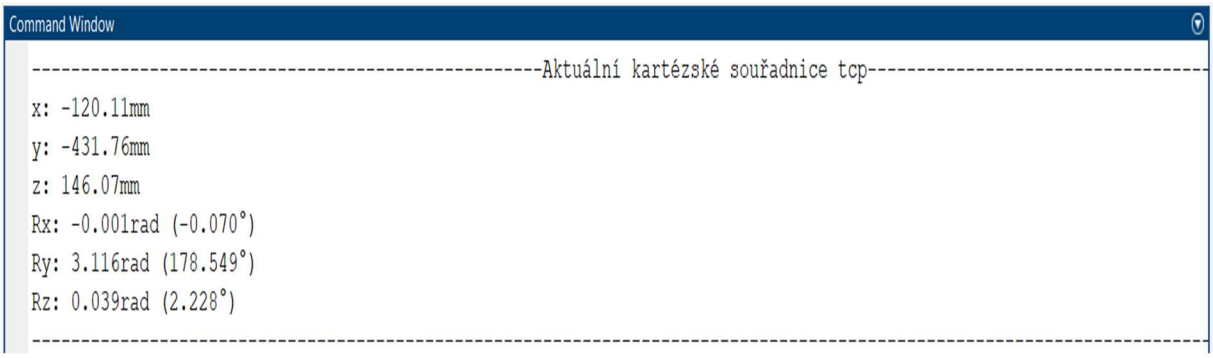
Středový bod nástroje anglicky **T**ool **C**enter **P**oint je další velmi užitečná informace sloužící k ovládání pohybu ramene. Tato informace udává trojici translačních a trojici rotačních kartézských souřadnic určujících přesně střed nástroje. Této informace, lépe řečeno těchto souřadnic je často využíváno při lineárním typu pohybu manipulátoru, jenž se v prostoru nástroje pohybuje lineárně. Způsob získání této informace od řídicí jednotky robota lze použitím skriptu takto:

Kód v Matlabu pro získání kartézských souřadnic TCP

```
tcp = (socket.UserData.zprava(56:61);
pause(0.5);
disp('Aktuální kartézské souřadnice tcp jsou: ');
fprintf('x:  %1.2fmm\ny:  %1.2fmm\nz:  %1.2fmm\nRx:  %1.2frad
(%1.2f°)\nRy:  %1.2frad  (%1.2f°)\nRz:  %1.2frad  (%1.2f°)\n',
tcp(1)*1000, tcp(2)*1000, tcp(3)*1000, tcp(4), tcp(4)*180/pi,
tcp(5), tcp(5)*180/pi, tcp(6), tcp(6)*180/pi);
```

Podobně, jako tomu bylo u kloubových souřadnic, tak i v tomto případě se informace čte stejným způsobem, ale dle tabulky 2.2 se nyní jedná řádky 56 až 61. Obdržený výsledek je uložen do proměnné `tcp` a před vypsáním souřadnic na obrazovku jsou hodnoty pro lepší představu upraveny. V případě translačních souřadnic se převedou hodnoty na milimetry, a v případě rotačních souřadnic jsou úhly natočení pro lepší představu zobrazeny v radiánech i

ve stupních. Výsledkem je přehledný výpis jednotlivých souřadnic středu nástroje x, y, z, Rx, Ry, Rz, viz obrázek 2.13.



Obrázek 2.13 – Výpis kartézských souřadnic TCP

Stav digitálních výstupů

Další nezbytností při ovládání ramene je možnost číst stav digitálních výstupů a v případě nutnosti také měnit jejich stav. Řídicí jednotka robota má k dispozici deset digitálních výstupů, přičemž digitální výstupy č. 8 a 9 jsou výstupy určeny pro nástroj. Informace o aktuálním stavu digitálních výstupů je řídicí jednotkou rovněž poskytována a dle tabulky 2.2 se jedná o řádek 131. Princip je založený na čtení číselné hodnoty ze soketu, která poukazuje na daný aktivní digitální výstup. Každý digitální výstup má v případě, že je aktivní, svou vlastní číselnou hodnotu. V případě, že je aktivních vícero výstupů, jedná se o součet těchto hodnot. Číselné hodnoty pro aktivní digitální výstupy znázorňuje tabulka 2.3. V případě, že přečtená číselná hodnota je rovna nule, pak to znamená, že není aktivní ani jeden digitální výstup.

Tabulka 2.3 – Číselné hodnoty pro aktivní digitální výstupy

Číslo digitálního výstupu	Číselná hodnota
0	1
1	2
2	4
3	8
4	16
5	32
6	64
7	128
8 (nástroj 0)	65536
9 (nástroj 1)	131072

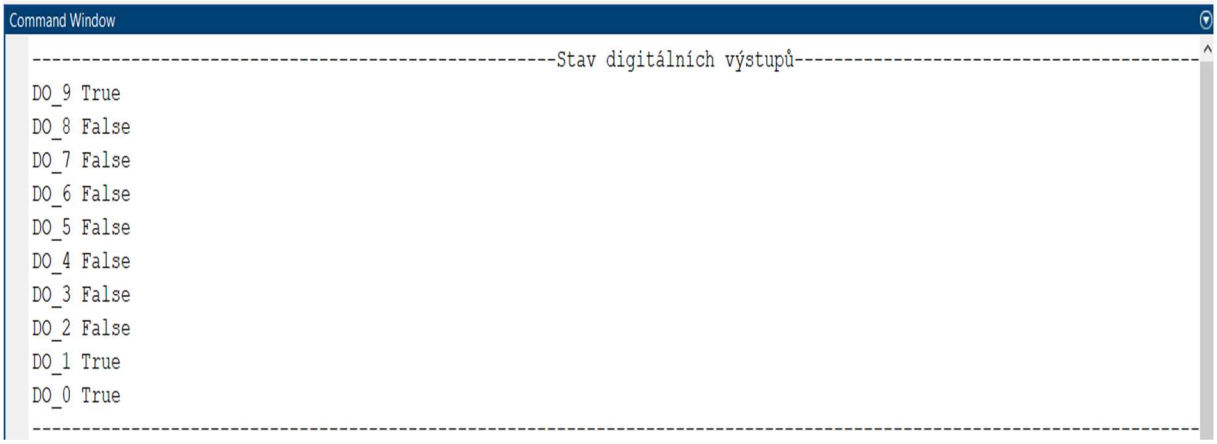
Způsob získání číselné hodnoty aktivního digitálního výstupu a zpracování vzdáleně z programu Matlab je následující. Pro zjednodušení je zde uveden postup pouze pro čtyři digitální výstupy, neboť postup pro zbylé výstupy je stejný a je zbytečné uvádět všechny.

Kód v Matlabu pro vyčítání digitálních výstupů

```
function vystupy(socket)
disp('Stav digitálních výstupu')
vystup = socket.UserData.zprava(131);
pause(0.1);
mozne_stavy = [1 2 4 8 16 32 64 128 65536 131072];
stav = vystup/mozne_stavy(10);
% DO_9/ digital_tool_out(1)
if stav >= 1
    disp('DO_9 True');
else
    disp('DO_9 False');
end
% DO_8 digital_tool_out(0)
stav = rem(vystup, mozne_stavy(10))/mozne_stavy(9);
if stav >= 1
    disp('DO_8 True');
else
    disp('DO_8 False');
end
% DO_1
stav = rem(vystup, mozne_stavy(3))/mozne_stavy(2);
if stav >= 1
    disp('DO_1 True');
else
    disp('DO_1 False');
end
% DO_0
stav = rem(vystup, mozne_stavy(2))/mozne_stavy(1);
if stav >= 1
```

```
    disp('DO_0 True');
else
    disp('DO_0 False');
end
end
```

Funkce v Matlabu výše je volána vždy, když chce uživatel znát aktuální stav digitálních výstupů. Stejně jako v předchozích dvou případech je nejprve ze soketu přečtena informace, která, jak již bylo zmíněno v tabulce 2.2, je dostupná na řádku 139. Číselná hodnota je uložena do proměnné `vystup` a pomocí dělení, zbytku po dělení a jednoduchých podmínek jsou vyhodnoceny aktivní a neaktivní výstupy. Výsledkem je výpis aktivních či neaktivních digitálních výstupů na obrazovku uživatele tak, jak zobrazuje obrázek 2.14.



Obrázek 2.14 – Výčet digitálních výstupů robota

Bezpečnostní režim robota

Mezi další velmi užitečné informace, které jsou v aplikaci čteny, patří aktuální bezpečnostní status, jenž vypovídá o stavu robota v rámci bezpečnosti. Řídicí jednotka robota má definováno deset různých bezpečnostních stavů, z nichž každý stav má podobně, jako tomu bylo v případě výstupů, svůj číselný kód, který je čtený a na základě jeho hodnoty je provedeno vyhodnocení. Číselné hodnoty pro dané stavy jsou uvedeny v tabulce 2.4.

Tabulka 2.4 – Číselné hodnoty pro bezpečnostní stavy

Bezpečnostní stav	Číselná hodnota stavu
Nedefinovaný bezp. stav	11
Selhání bezpečnostního stavu	9
Porušení bezp. stavu	8
Nouzové zastavení robota	7
Nouzové zastavení systému	6
Bezpečnostní zastavení	5
Režim obnovy bezp. stavu	4
Ochranné zastavení	3
Snížený bezpečnostní stav	2
Normální bezpečnostní stav	1

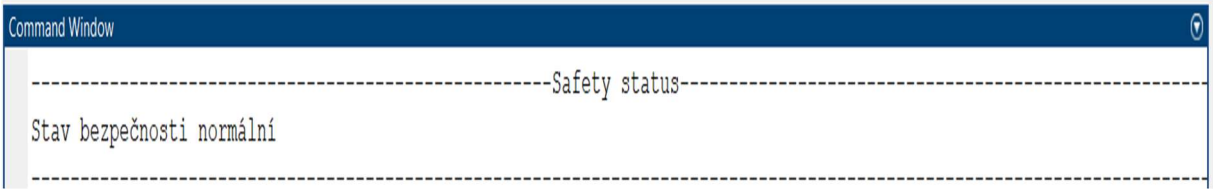
Výčet příslušné číselné hodnoty ze soketu a následné porovnání za účelem výběru aktuálního bezpečnostního stavu je následující:

Kód v Matlabu pro zjištění aktuálního bezpečnostního stavu

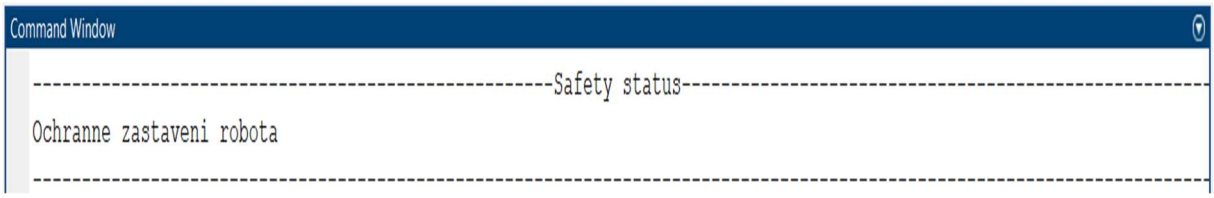
```
function safety_status(socket)
clc;
disp('Bezpečnostní status');
stav = (socket.UserData.zprava(139));
pause(0.5);
switch stav
case 1
    disp('Stav bezpečnosti normální');
case 2
    disp('Snížený bezpečnostní stav');
case 3
    disp('Ochranné zastavení robota');
case 4
    disp('Bezpečnostní status obnoven');
case 5
    disp('Ochranné zastavení');
case 6
```

```
    disp('Nouzové zastavení systému');
case 7
    disp('Nouzové zastavení robota');
case 8
    disp('Porušení bezpečnostního stavu');
case 9
    disp('Chyba bezpečnostního stavu');
case 11
    disp('Nedefinovaný bezpečnostní stav');
end
end
```

Opět se jedná o samostatnou funkci, která je volána z hlavního programu. Vstupním parametrem funkce je objekt soketu, aby bylo možné vyčíst příslušnou číselnou hodnotu. Z kódu je vidět, že příslušná informace se nachází na řádku 139. Přečtená hodnota se uloží do proměnné `stav` a pomocí porovnávací funkce `switch` je vybrán správný bezpečnostní stav. Aktuální stav je následně vypsán uživateli na obrazovku tak, jak je ukázáno na obrázku 2.15 a 2.16. První obrázek reprezentuje normální bezpečnostní stav a obrázek 2.16 pak stav ochranného zastavení robota, ke kterému dochází například při singularitě pohybu.



Obrázek 2.16 – Čtení bezpečnostního statusu při normálním stavu



Obrázek 2.15 – Čtení bezpečnostního statusu při stavu nouzového zastavení

Stav ramene

Velice podobnou funkcí jako v předchozím případě je funkce s možností přečíst aktuální stav ramene. Tato funkce dává informaci o tom, co rameno právě dělá, tedy jestli je

napájení vypnuto, zapnuto, jestli robot není v režimu bootování apod. Podobně jako v předchozím případě, tak i tato funkce má definované číselné hodnoty pro každý stav, viz tabulka 2.5.

Tabulka 2.5 – Číselné hodnoty pro stav ramene

Stav ramene	Číselná hodnota stavu
Chybí kontrolér	-1
Odpojeno	0
Robot čeká na potvrzení bezpečnosti	1
Bootování	2
Napájení vypnuto	3
Napájení zapnuto	4
Robot nečinný	5
Backdrive	6
Robot v provozu	7
Aktualizace firmwaru	8

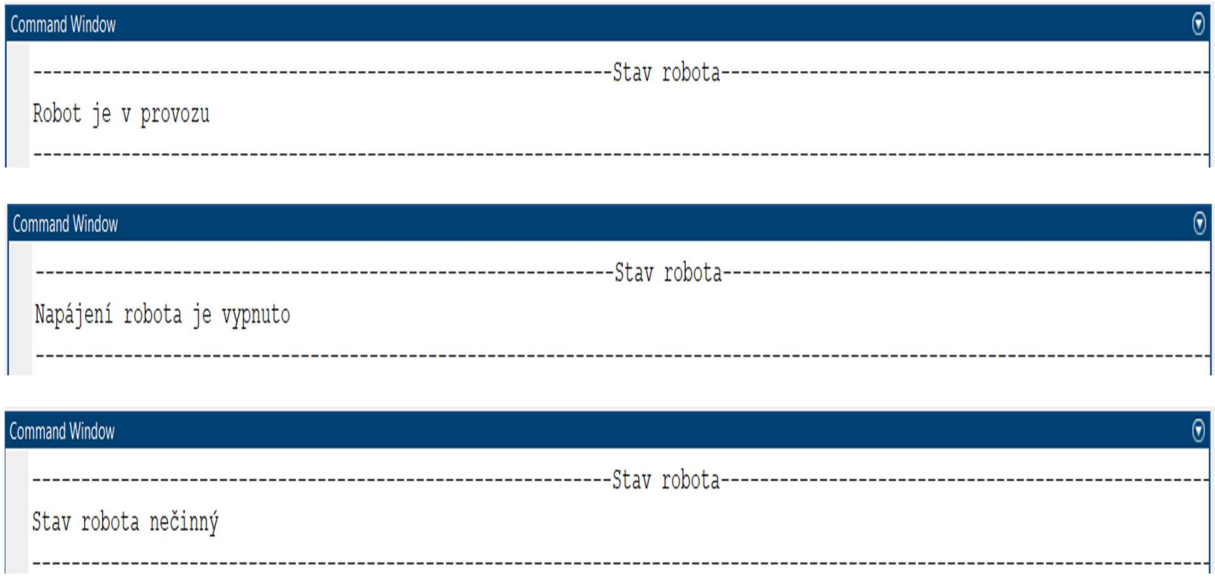
Získání příslušné číselné hodnoty je opět provedeno pomocí funkce v Matlabu, která je z hlavního programu v případě potřeby volána a kód je následující:

Kód v Matlabu pro zjištění aktuálního stavu ramene

```
Function Stav_ramene(socket)
disp('Inicializace robota');
stav = (socket.UserData.zprava(95));
switch stav
case -1
    disp('Chybí kontrolér');
case 0
    disp('Robot je odpojen');
case 1
    disp('Robot čeká na potvrzení bezpečnosti');
case 2
    disp('Robot bootuje');
case 3
```

```
        disp('Napájení robota je vypnuto');
    case 4
        disp('Napájení robota je zapnuto');
    case 5
        disp('Stav robota je nečinný');
    case 6
        disp('Backdrive');
    case 7
        disp('Robot je v provozu');
    case 8
        disp('Robot aktualizuje firmware');
    end
end
```

Vstupním parametrem funkce je opět objekt TCP/IP komunikace a vyčtený parametr je uložený a vyhodnocený pomocí funkce `switch`. Po vyhodnocení informace je výsledek, tedy aktuální stav ramene, vypsán na obrazovku uživatele. Obrázek 2.17 níže ukazuje tři různé stavy ramene.



Obrázek 2.17 – Možné stavy ramene

Teplota kloubů

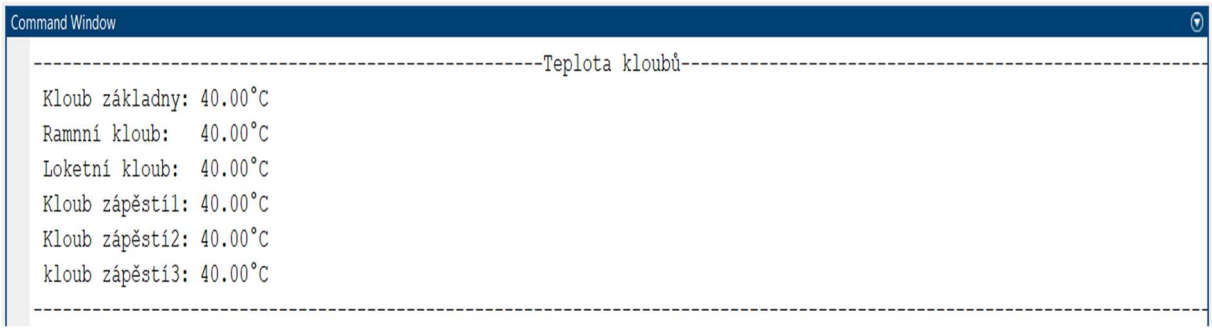
Poslední funkcionalitou hlavního programu, která ještě nebyla popsána, je informace o teplotě kloubů. Tato informace je dostupná na řádcích 87 až 92. Teplota kloubů by se běžně

měla pohybovat okolo 40 °C. Při vysoké rychlosti pohybu jsou klouby více namáhány a teplota může být vyšší. Kód v Matlabu pro zjištění aktuální teploty jednotlivých kloubů je následující:

Kód v Matlabu po zjištění aktuální teploty kloubů ramene

```
clc;
teplota = (socket.UserData.zprava(87:92));
pause(0.5);
disp('Teplota kloubů');
fprintf('Kloub základny: %1.2f°C\n Ramenní kloub: %1.2f°C\n...
Loketní kloub: %1.2f°C\n Kloub zápěstí1: %1.2f°C\n ...
Kloub zapesti2: %1.2f°C\n kloub zapesti3: %1.2f°C\n'...
, teplota (1), teplota (2), teplota (3), teplota (4),...
teplota (5), teplota (6));
```

Vyčítání aktuální teploty kloubů je velmi jednoduché a vlastně stačí přečíst informaci ze soketu z příslušných řádků a vypsat tuto informaci ve správném formátu uživateli na obrazovku. Výsledek takto napsaného kódu je přehled jednotlivých kloubů a dané teploty, viz obrázek 2.18.



Obrázek 2.18 – Teplota kloubů

2.4.2 Zasílání různých příkazu

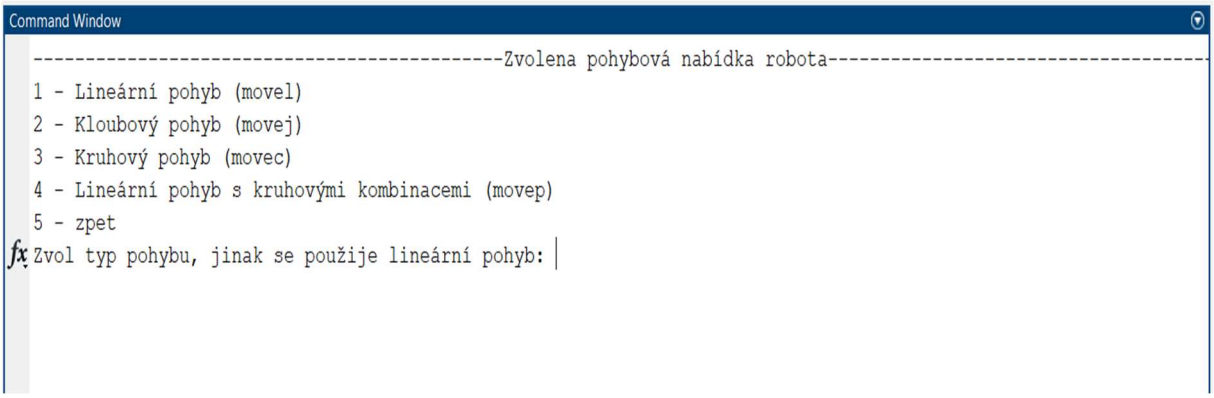
Způsob zjišťování informací byl na několika ukázkách znázorněn v pododdílu 2.4.1, tento pododdíl bude pojednávat o způsobu odesílání příkazů řídicí jednotce robota. Nutno připomenout, že při odesílání příkazů je důležité dodržet syntaxi jednotlivých příkazů, jinak nebudou zasílané příkazy provedeny. Dále je velmi důležité, aby uživatel věděl, co může od

zaslaného a prováděného příkazu čekat, protože v krajním případě může dojít k poškození zařízení nebo nežádoucímu chování.

V tomto pododdílu čtenář nalezne popis kódů a obrázky zachycující obrazovky Matlabu pro příklady zasílání příkazů pro funkcionality ukázkového programu uvedené na obrázku 2.10. Jedná se zejména o pohyb ramene, změnu stavu digitálních výstupů, aktivaci, resp. deaktivaci režimu volného pohybu a možnosti vypnutí řídicí jednotky a robota.

Pohyb ramene

Ovládání pohybu robotického ramene je jednou ze základních a asi nejdůležitějších operací, které se s ramenem provádějí. Každý nebo alespoň většina robotických manipulátorů má na výběr hned z několika typů pohybů. Většinou se jedná o lineární, kloubový, kruhový či jiný typ pohybu, z nichž každý typ má své výhody a nevýhody a hodí se k něčemu jinému. Je výhodné znát všechny typy pohybů, kterými se robot může pohybovat a v případě potřeby je v programu různě kombinovat. Rameno UR3 má na výběr lineární, kloubový, kruhový a smíšený pohyb, a každý z těchto pohybů má svůj příkaz v jazyce URScrip, který lze ramenu vzdáleně poslat. Níže jsou popsány způsoby ovládání lineárního a kloubového pohybu, neboť jsou tyto dva typy pohybů nejčastěji používány. Obrázek 2.19 tedy ukazuje pohybovou nabídku, dostupnou z Matlabu.



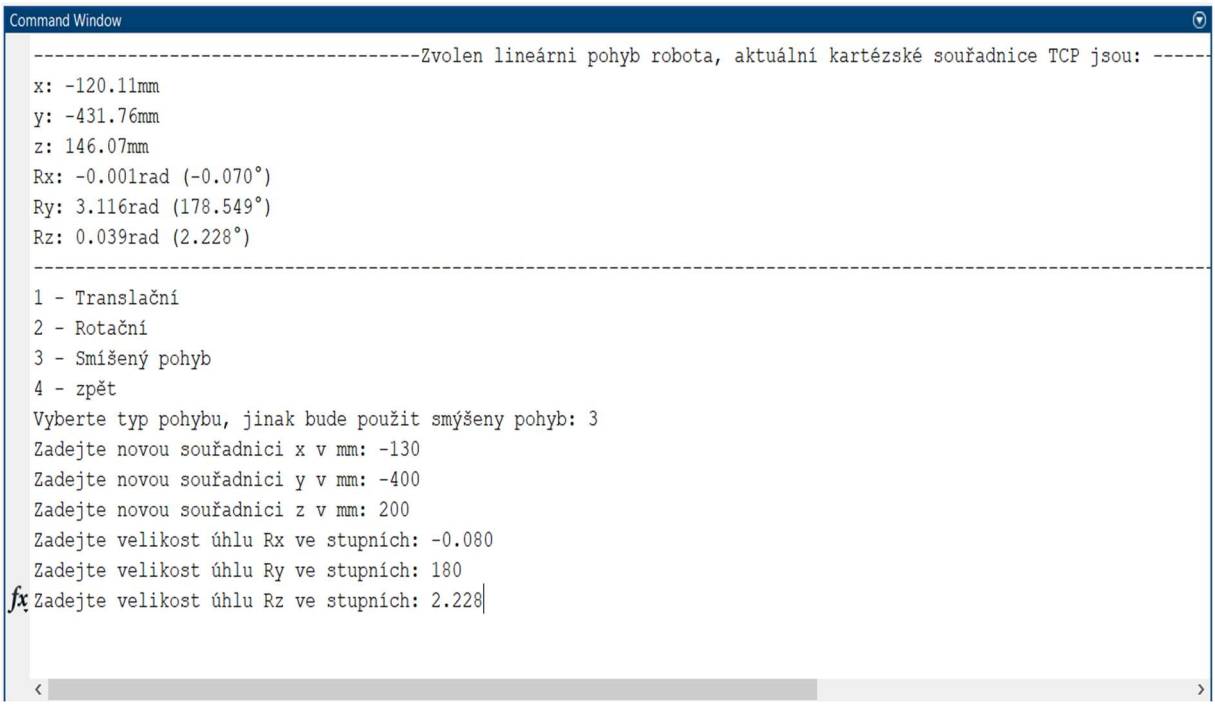
Obrázek 2.19 – Pohybová nabídka ramene

Obrázek 2.19 ukazuje, že je uživatel vyzván ke zvolení jednoho z nabízených typů pohybu. Po zvolení je uživateli zobrazená nová nabídka dle zvoleného pohybu. Jelikož ale při pohybu nezáleží pouze na tom, kam se má robot dostat, ale i na tom odkud se robot do cíle má pohnout, jsou uživateli podle zvoleného typu pohybu vypsané aktuální kloubové či kartézské souřadnice. Kód pro vypsání aktuálních kartézských souřadnic již byl popsán, takže je zde vynechán. Kód pro odeslání požadavku k lineárnímu pohybu je následující:

Kód v Matlabu pro lineární pohyb

```
tcp = (socket.UserData.zprava(56:61));
fprintf('1 - Translační \n2 - Rotační \n3 - Smíšeny pohyb \n4
- zpět\n');
Slozka_pohybu = input('Vyberte typ souřadnic, jež chcete
změnit, jinak se použije smísený: ');
if Slozka_pohybu == 1
    x = input('Zadejte novou souřadnici x v mm: ') * 0.001;
    y = input('Zadejte novou souřadnici y v mm: ') * 0.001;
    z = input('Zadejte novou souřadnici z v mm: ') * 0.001;
    Rx = tcp(4);
    Ry = tcp(5);
    Rz = tcp(6);
elseif Slozka_pohybu == 2
    x = tcp(1);
    y = tcp(2);
    z = tcp(3);
    Rx = input('Zadejte velikost úhlu Rx ve stupních: ')*pi/180;
    Ry = input('Zadejte velikost úhlu Ry ve stupních: ')*pi/180;
    Rz = input('Zadejte velikost úhlu Rz ve stupních: ')*pi/180;
else
    x = input('Zadejte novou souřadnici x v mm: ') * 0.001;
    y = input('Zadejte novou souřadnici y v mm: ') * 0.001;
    z = input('Zadejte novou souřadnici z v mm: ') * 0.001;
    Rx = input('Zadejte velikost úhlu Rx ve stupních: ')*pi/180;
    Ry = input('Zadejte velikost úhlu Ry ve stupních: ')*pi/180;
    Rz = input('Zadejte velikost úhlu Rz ve stupních: ')*pi/180;
end
fprintf(socket,['movel(p[',num2str(x),',',num2str(y),','...
,num2str(z),',',num2str(Rx),',',num2str(Ry),','...
,num2str(Rz), '], a=1.4', ', v=1.05',')\n']);
pause(5);
```

Zvolením lineárního typu pohybu jsou uživateli vypsány kartézské souřadnice TCP a může zvolit, zda chce pohyb translační, rotační nebo smíšený. Výběrem je ovlivněna možnost měnit souřadnice, například translační složka pohybu využívá pouze souřadnice x, y, z, kdežto pro rotaci kolem nějaké osy je potřeba zadat velikost úhlu natočení ve stupních. Zadané souřadnice jsou uloženy do příslušných proměnných a převedeny na jednotky, které jsou vhodné pro odeslání. Poslední příkaz slouží k odeslání pohybového příkazu ve správném tvaru robotovi. Jedná se o modifikovaný příkaz „movel(p[x, y, z, Rx, Ry, Rz], a=1,4, v = 1,05)“ tak, aby proměnné obsahující souřadnice byly zadávány uživatelem. Výsledek kódu popsaného výše znázorňuje obrázek 2.20.

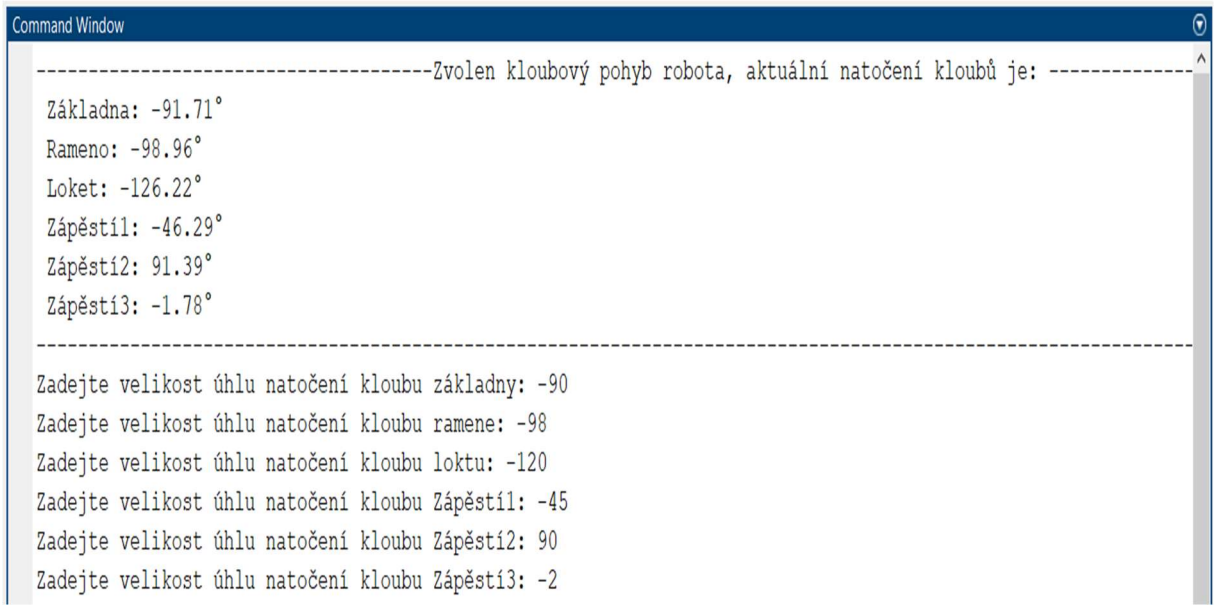


Obrázek 2.20 – Lineární pohyb robota

Zvolením druhé pohybové možnosti, tedy kloubového pohybu, jsou uživateli na obrazovku zobrazeny aktuální kloubové souřadnice základny, ramene, loktu, zápěstí 1 až zápěstí 3. Následně je uživatel vyzván k zadání nových úhlů natočení ve stupních a hodnoty jsou uloženy do příslušných proměnných. V dalším kroku jsou převedeny na radiány a příkaz je odeslán robotovi. V tomto případě je využit příkaz „movej([základna, rameno, loket, zápěstí 1, zápěstí2, zápěstí3], a = 1,4, v = 1,05)“. Výsledek kódu popsaného níže je znázorněn na obrázku 2.21 a kód pro kloubový pohyb je následující:

Kód v Matlabu pro kloubový pohyb

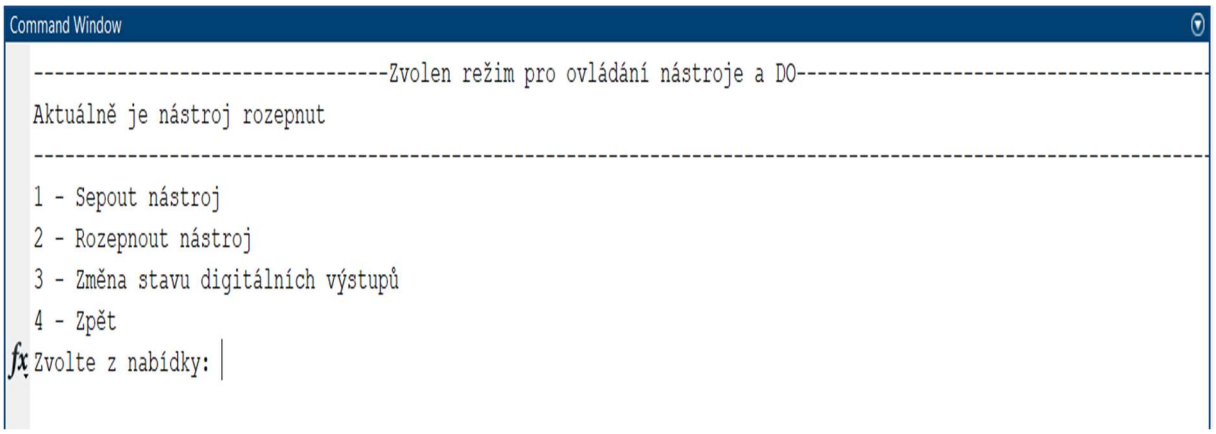
```
fprintf('Zvolen kloubový pohyb robota, aktuální natočení kloubů je: \n');
joints = (socket.UserData.zprava(32:37))*180/pi;
pause(0.1)
fprintf(' Základna: %1.2f°\n Rameno: %1.2f°\n Loket: %1.2f°\n Zápěstí1: %1.2f°\n Zápěstí2: %1.2f°\n Zápěstí3: %1.2f°\n', joints(1), joints(2), joints(3), joints(4), joints(5), joints(6));
základna = input('Zadejte velikost úhlu natočení kloubu základny: ') * pi/180;
rameno = input('Zadejte velikost úhlu natočení kloubu ramene: ')*pi/180;
loket = input('Zadejte velikost úhlu natočení kloubu loktu: ')*pi/180;
zapesti1 = input('Zadejte velikost úhlu natočení kloubu Zápěstí1: ')*pi/180;
zapesti2 = input('Zadejte velikost úhlu natočení kloubu Zápěstí2: ')*pi/180;
zapesti3 = input('Zadejte velikost úhlu natočení kloubu Zápěstí3: ')*pi/180;
fprintf(socket,
['movej([',num2str(zakladna),',',num2str(rameno),',',num2str(loket),',',num2str(zapesti1),',',num2str(zapesti2),','...
,num2str(zapesti3),'],a=',num2str(1.4),',v=',num2str(1.05),')\n']);
pause(5);
```



Obrázek 2.21 – Kloubový pohyb robota

Změna stavu digitálních výstupů robota

Z pododdílu 2.4.1 je patrné, jak zjistit aktuální stav digitálních výstupů, ale to samozřejmě nestačí a někdy je také potřeba změnit jejich stav. Nejčastěji se u manipulátorů mění stav digitálního výstupu, do kterého je zapojený nástroj robota. Z tabulky 2.3 je patrné, že u manipulátorů UR3 se jedná o digitální výstupy 8 a 9. V testovacím programu, viz obrázek 2.10, je pod volbou číslo čtyři možnost měnit stav digitálních výstupů a nástroje. Zvolením této možnosti, se uživateli zobrazí následující nabídka (obrázek 2.22).



Obrázek 2.22 – Nabídka pro ovládání digitálních výstupů

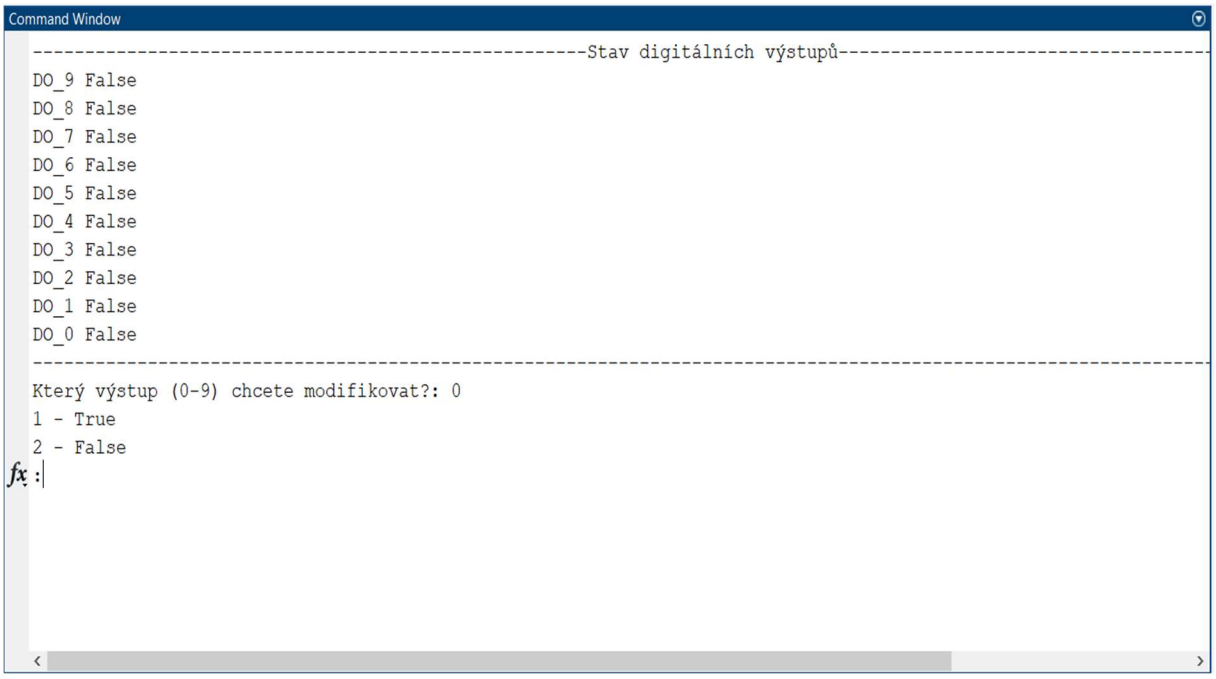
První dvě volby odešlou příkaz pro sepnutí či rozepnutí nástroje. Třetí možnost nabídne uživateli další nabídku, kde si vybere, u kterého z digitálních výstupů (0 až 9) chce změnit stav

a následně se příkaz s požadavkem odešle robotovi. Výběrem třetí možnosti, tedy modifikace stavu digitálních výstupů, se uživateli ukáže nabídka ukázána na obrázku 2.23. Kód pro výběr požadavku a odeslání příkazu je následující:

Kód v Matlabu pro interakci s DO

```
clc;
fprintf('Zvolen režim pro ovládání nástroje a DO \n');
Stav_DO = socket.UserData.zprava(131);
pause(0.5);
if  Stav_DO >= 65536 && Stav_DO <= 65791 || Stav_DO >=196608...
&& Stav_DO <= 196863
    disp('Aktuálně je nástroj sepnut');
else
    disp('Aktuálně je nástroj rozepnut');
end
fprintf('1 - Sepnout nástroj \n2 - Rozepnout nástroj ...
\n3 - Změna stavu digitálních výstupů \n4 - Zpět\n');
Nastroj = input('Zvolte z nabídky: ');
if Nastroj == 1
    clc;
    fprintf(socket, 'set_tool_digital_out(0, True)');
    pause(0.1);
elseif Nastroj == 2
    clc;
    fprintf(socket, 'set_tool_digital_out(0, False)\n');
    pause(0.1);
elseif Nastroj == 3
    clc;
    fprintf('Aktuální stav digitálních výstupů \n');
    vystupy(socket);
    zvol = input('Který výstup (0-9) chcete modifikovat? ');
    akce = input('1 - True \n2 - False \n:');
    pause(0.01);
if akce == 1
```

```
    fprintf(socket,['set_digital_out(',num2str(zvol),',...
    ,'True',')\n']);
else
    fprintf(socket,['set_digital_out(',num2str(zvol),',...
    ,'False',')\n']);
end
```



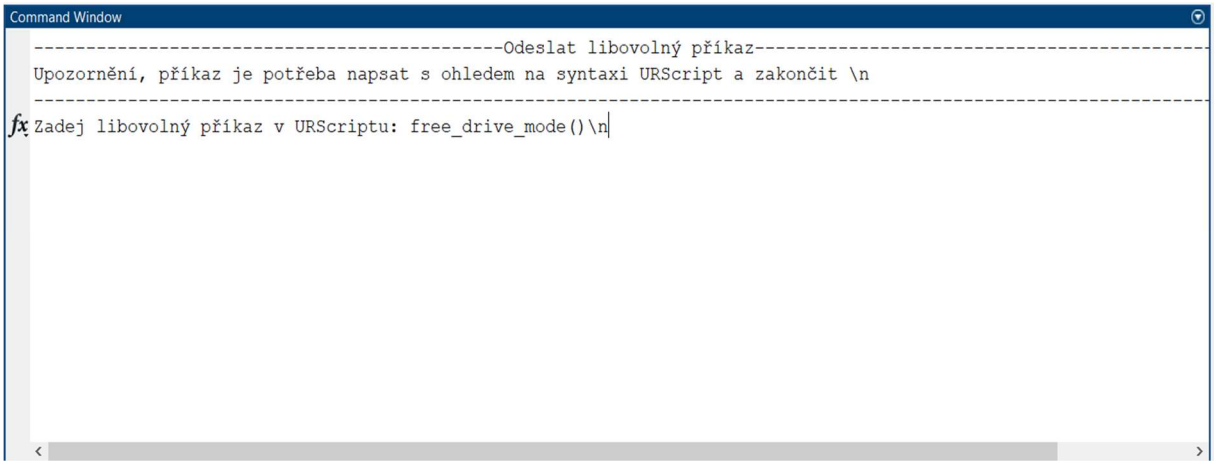
Obrázek 2.23 – Modifikace stavu digitálních výstupů

Zaslání libovolného příkazu

V případě, že uživatel chce poslat robotovi libovolný jiný příkaz než ty, které jsou zautomatizovány v programu, lze využít možnost číslo 5, viz obrázek 2.10. Tato možnost nabídne uživateli zadat libovolný příkaz URScriptu, avšak nutno připomenout, že aby byl příkaz odeslán a proveden, musí být dodržena syntaxe. A jak již bylo zmíněno, je potřeba brát v úvahu také to, že ne každý typ příkazu má smysl posílat. Výsledek kódu tedy průběh zadávání libovolného příkazu znázorňuje obrázek 2.24. Kód pro možnost zaslání různých příkazů je následující:

Kód v Matlabu pro zaslání libovolného příkazu

```
clc;
disp('Odeslat libovolný příkaz');
disp('Upozornění, příkaz je potřeba napsat s ohledem na syntaxi
URScript a zakončit \n');
prikaz = input('Zadejte libovolný příkaz v URScriptu: ', 's');
fprintf(socket, prikaz);
pause(1);
```



Obrázek 2.24 – Průběh zadávání příkazu

Vypnutí ramene

Poslední možností programu je vzdálené vypnutí robotického ramene a ukončení programu v Matlabu. Zde se jedná o jednoduchý proces odeslání příkazu pro vypnutí ramene a pomocí příkazu `break` v Matlabu ukončit cyklus programu. Kód pro vzdálené vypnutí ramene je následující:

Kód v Matlabu pro vypnutí robota na dálku

```
disp('Program bude ukončen a robot se vypne');
fprintf(socket, 'powerdown()\n');
pause(1);
clc;
break;
```

2.5 TESTOVÁNÍ REÁLNÉ APLIKACE

Předchozí pododdíly pojednávaly o způsobu ovládání ramene UR3 vzdáleně z počítače. Jelikož hlavním úkolem průmyslových manipulátorů jsou hlavně pohybové úlohy, kde se nejčastěji manipuluje s nějakými objekty, jsou zde ukázány tři jednoduché úlohy, prezentující způsob ovládání pro reálnou aplikaci. Při těchto úkonech se často využívá velké přesnosti i při vysokých rychlostech pohybu.

2.5.1 Paletizace

V první ukázkové úloze se jedná o přesun dvanácti objektů válcového tvaru o rozměrech 40 x 20 mm do příslušných otvorů v paletě 200 x 150 mm. Tato úloha je jedním z typů, při kterých lze vidět vysokou přesnost manipulátoru, a to i při použití vysoké rychlosti pohybu. Na obrázku 2.25 je znázorněn výchozí stav, na kterém je možné vidět prázdnou paletu a objekty ve výchozí poloze.

Na začátku je zapotřebí znát kartézské souřadnice všech objektů a příslušných otvorů v paletě. Za tímto účelem je například možné využít již zmiňovaný režim volného pohybu, a přesunout TCP nástroje manipulátoru nad jednotlivé objekty i otvory ručně a souřadnice si nechat vypsat v Matlabu způsobem, který byl již popsán. Zná-li uživatel jednotlivé souřadnice, může využít aplikace popsané v oddílu 2.4 a postupně přesunout každý objekt samostatně, nebo může napsat pohybové příkazy v Matlabu následovně:

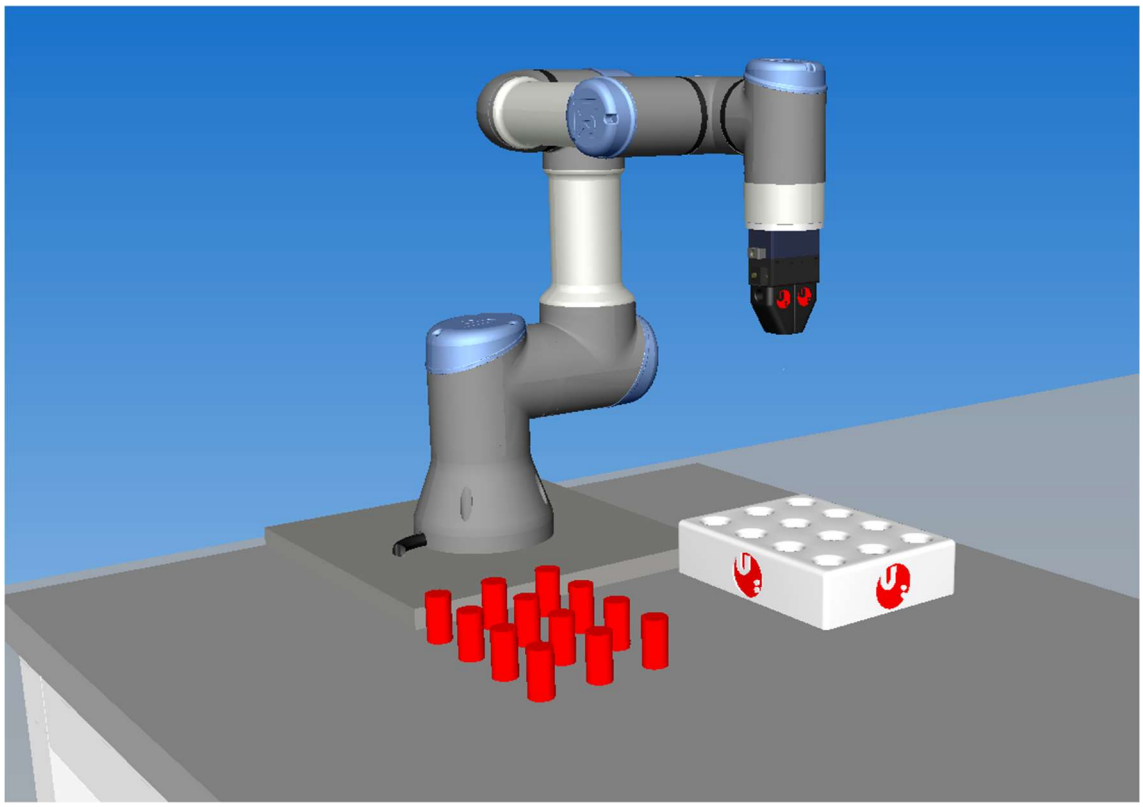
Kód v Matlabu pro ovládání úlohy paletizace

```
fprintf(socket,  ('movel(p[val1_x,  val1_y,  val1_z,  val1_Rx,
val1_Ry, val1_Rz], a = 1.2, v = 0.25)\n'));
pause(3);
fprintf(socket, 'set_tool_digital_out(0, False)\n');
pause(1);
fprintf(socket, ('movel(p[val1_x, val1_y, val1_z-z‘, val1_Rx,
val1_Ry, val1_Rz], a = 1.2, v = 0.25)\n'));
pause(3);
fprintf(socket, 'set_tool_digital_out(0, True)\n');
pause(0.3);
fprintf(socket, ('movel(p[val1_x, val1_y, val1_z+z‘, val1_Rx,
val1_Ry, val1_Rz], a = 1.2, v = 0.25)\n'));
```

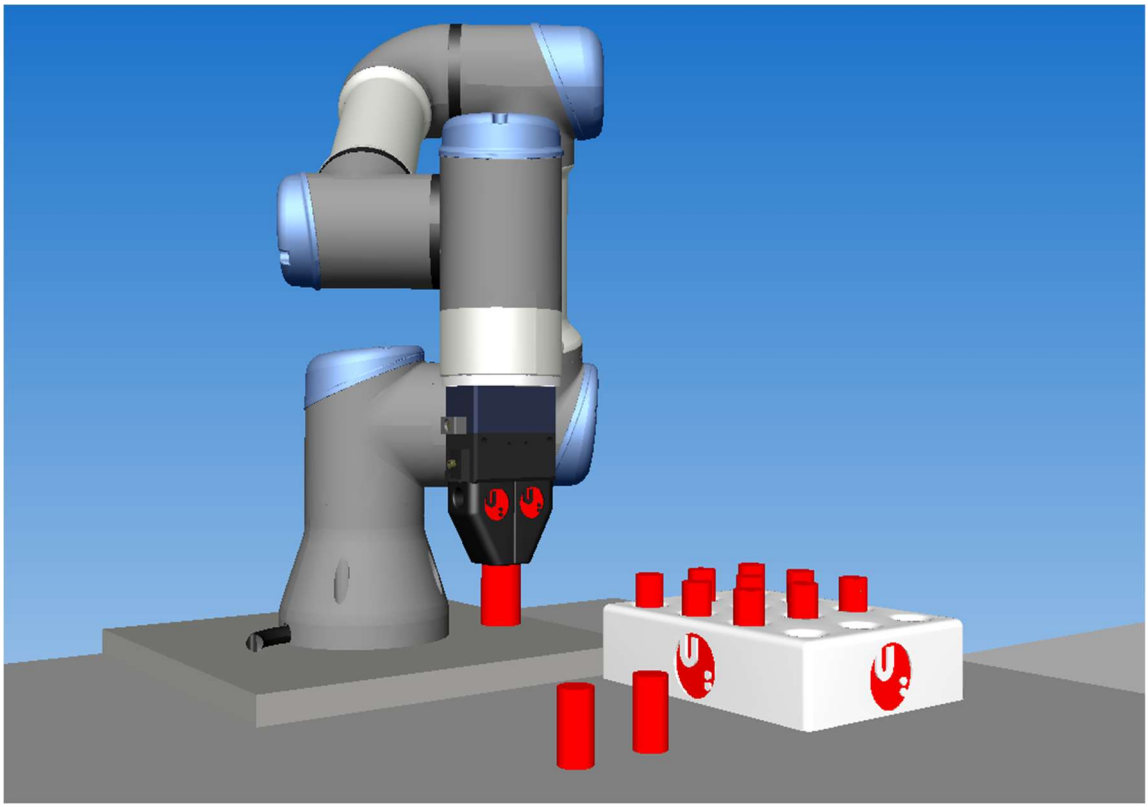
```
pause(3);
fprintf(socket,  ('movel(p[pal_x1,  pal_y1,  pal_z1,  pal_Rx1,
pal_Ry1, pal_Rz1], a = 1.2, v = 0.25)\n'));
pause(3)
fprintf(socket, ('movel(p[pal_x1, pal_y1, pal_z1-z``, pal_Rx1,
pal_Ry1, pal_Rz1], a = 1.2, v = 0.25)\n'));
pause(3)
fprintf(socket, 'set_tool_digital_out(0, False)\n');
pause(0.3);
fprintf(socket, ('movel(p[pal_x1, pal_y1, pal_z1+z``, pal_Rx1,
pal_Ry1, pal_Rz1], a = 1.2, v = 0.25)\n'));
```

Prvním příkazem je docíleno lineárního pohybu s TCP nad první váleček. Následně se rozepne nástroj a upraví se souřadnice z o hodnotu z‘, aby nástroj mohl uchopit váleček. Následuje příkaz pro sepnutí nástroje a opětovná změna v ose z o z‘, pro zvednutí válečku.

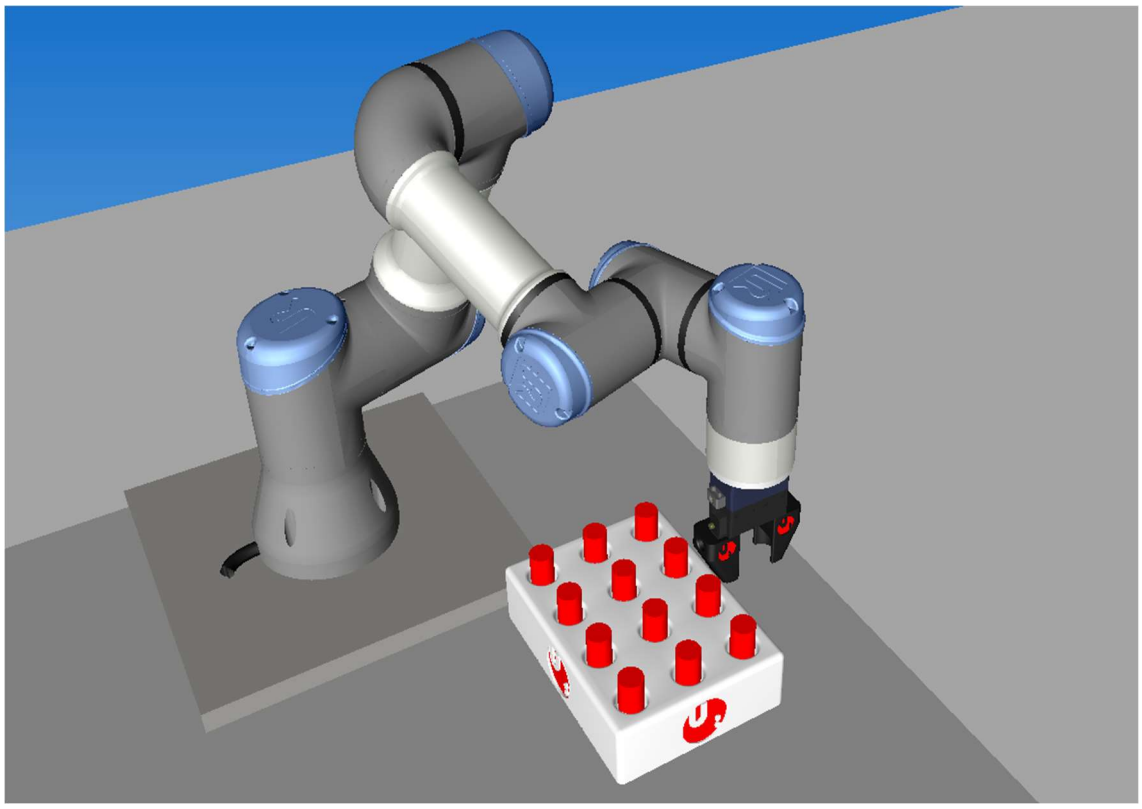
Další řádky kódu jsou obdobné, nejprve je rameno lineárně přesunuto nad otvor, poté je v ose z váleček vložen do palety, kde ho rameno pustí. Pro představu je zobrazena pouze část kódu pro přesun jednoho válečku, neboť postup přesunu je stejný a mění se pouze souřadnice pro uchopení a odkládání válečků. Pohybové úkony manipulátoru jsou znázorněny na obrázcích 2.25 až 2.27.



Obrázek 2.25 – Výchozí stav válečků



Obrázek 2.26 – Přesun válečků



Obrázek 2.27 – Cílové pozice válečků

2.5.2 Stavění pyramidy

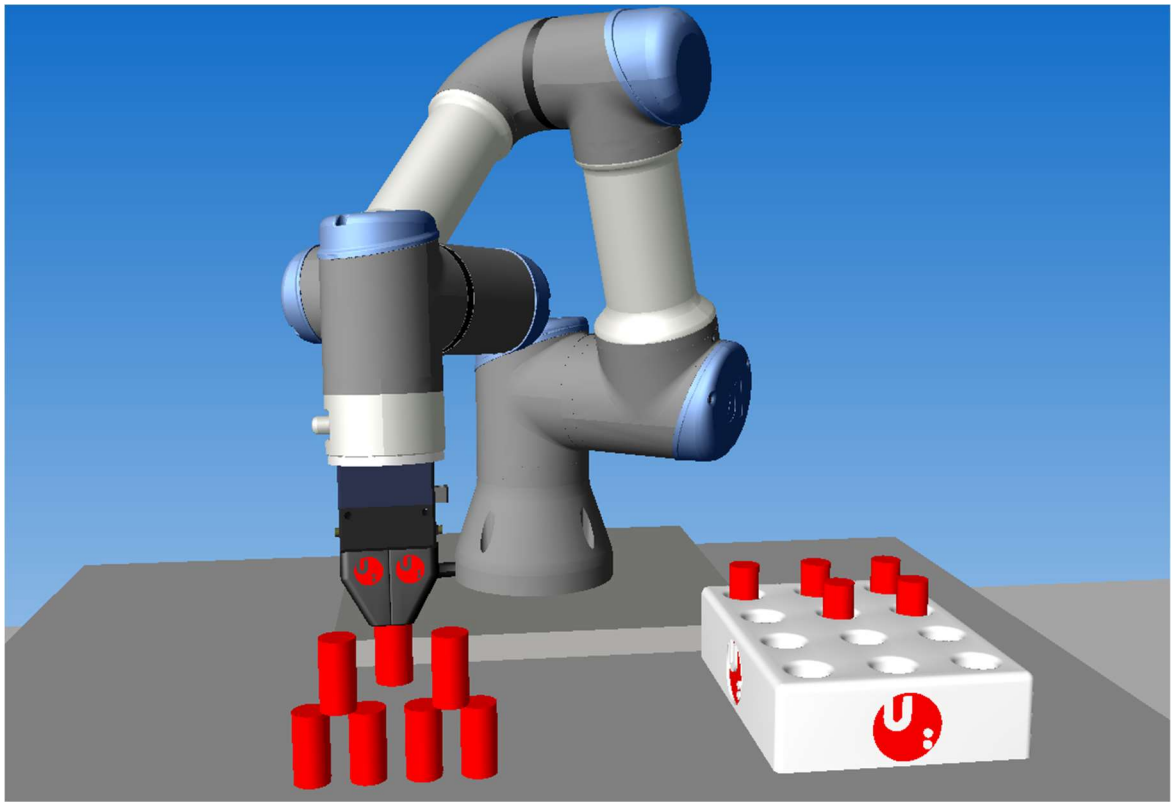
Druhou úlohou prezentující výbornou manipulační schopnost ramene je stavění válcových objektů na sebe (stavění pyramidy). Tentokrát se vychází z předchozího cílového stavu znázorněného na obrázku 2.27. Souřadnice objektů jsou známy a pohybové úkony manipulátoru jsou obdobné. Válečky jsou stavěny s 15mm rozestupy z důvodu zamezení kolize již položených válečků a nástroje při pokládání. Ovládání pohybové úlohy z Matlabu je následující:

Kód v Matlabu pro ovládání úlohy stavění pyramidy

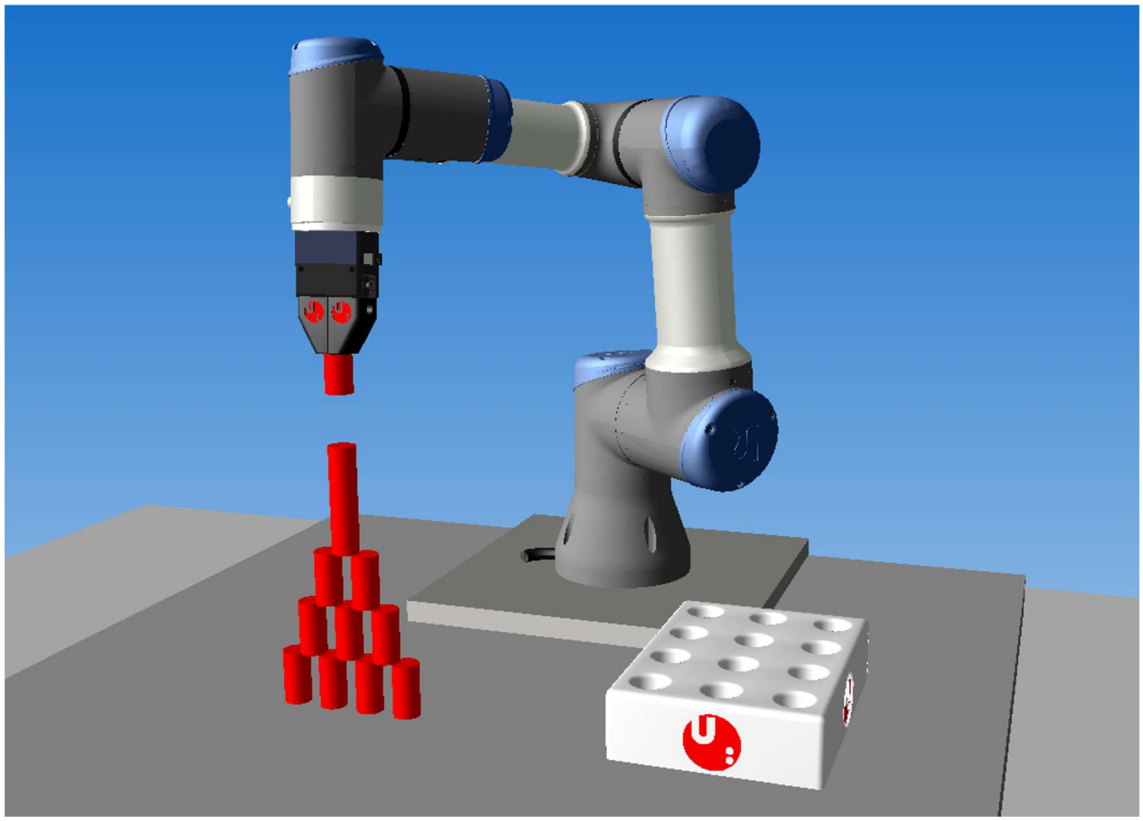
```
fprintf(socket, ('movel(p[pal_x1, pal_y1, pal_z1, pal_Rx1,
pal_Ry1, pal_Rz1], a = 1.2, v = 0.25)\n'));
pause(3);
fprintf(socket, 'set_tool_digital_out(0, False)\n');
pause(1);
```

```
fprintf(socket, ('movel(p[pal_x1, pal_y1, pal_z1-z`, pal_Rx1,
pal_Ry1, pal_Rz1], a = 1.2, v = 0.25)\n'));
pause(3);
fprintf(socket, 'set_tool_digital_out(0, True)\n');
pause(0.3);
fprintf(socket, ('movel(p[pal_x1, pal_y1, pal_z1+z`, pal_Rx1,
pal_Ry1, pal_Rz1], a = 1.2, v = 0.25)\n'));
pause(3);
fprintf(socket,  ('movel(p[pyr_x1,  pyr_y1,  pyr_z1,  pyr_Rx1,
pyr_Ry1, pyr_Rz1], a = 1.2, v = 0.25)\n'));
pause(3)
fprintf(socket, ('movel(p[pyr_x1, pyr_y1, pyr_z1-z``, pyr_Rx1,
pyr_Ry1, pyr_Rz1], a = 1.2, v = 0.25)\n'));
pause(3)
fprintf(socket, 'set_tool_digital_out(0, False)\n');
pause(0.3);
fprintf(socket, ('movel(p[pyr_x1, pyr_y1, pyr_z1+z``, pyr_Rx1,
pyr_Ry1, pyr_Rz1], a = 1.2, v = 0.25)\n'));
```

Opět se jedná o využití pohybových příkazů a ovládání digitálního výstupu nástroje. Tentokrát se přesouvají objekty z palety na předem definované kartézské pozice a vhodným umístěním se z nich staví pyramida. Výsledek a mezikrok této úlohy znázorňují obrázky 2.28 a 2.29.



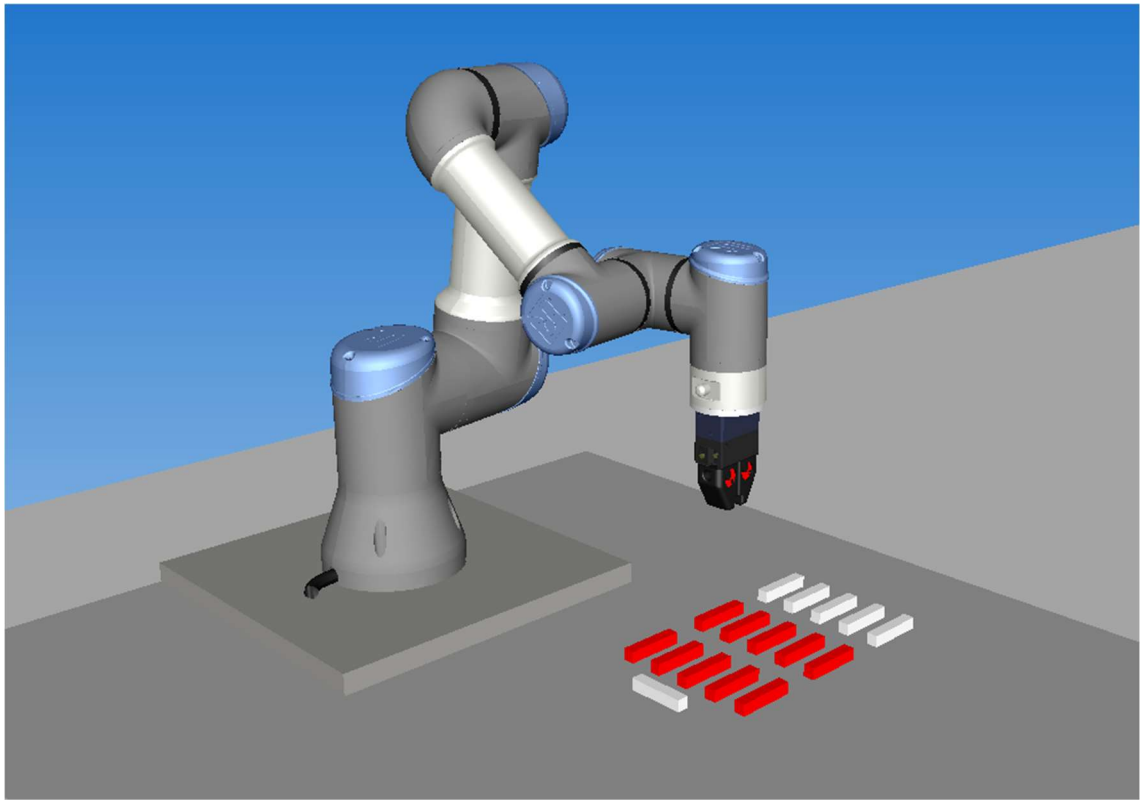
Obrázek 2.28 – Stavění pyramidy



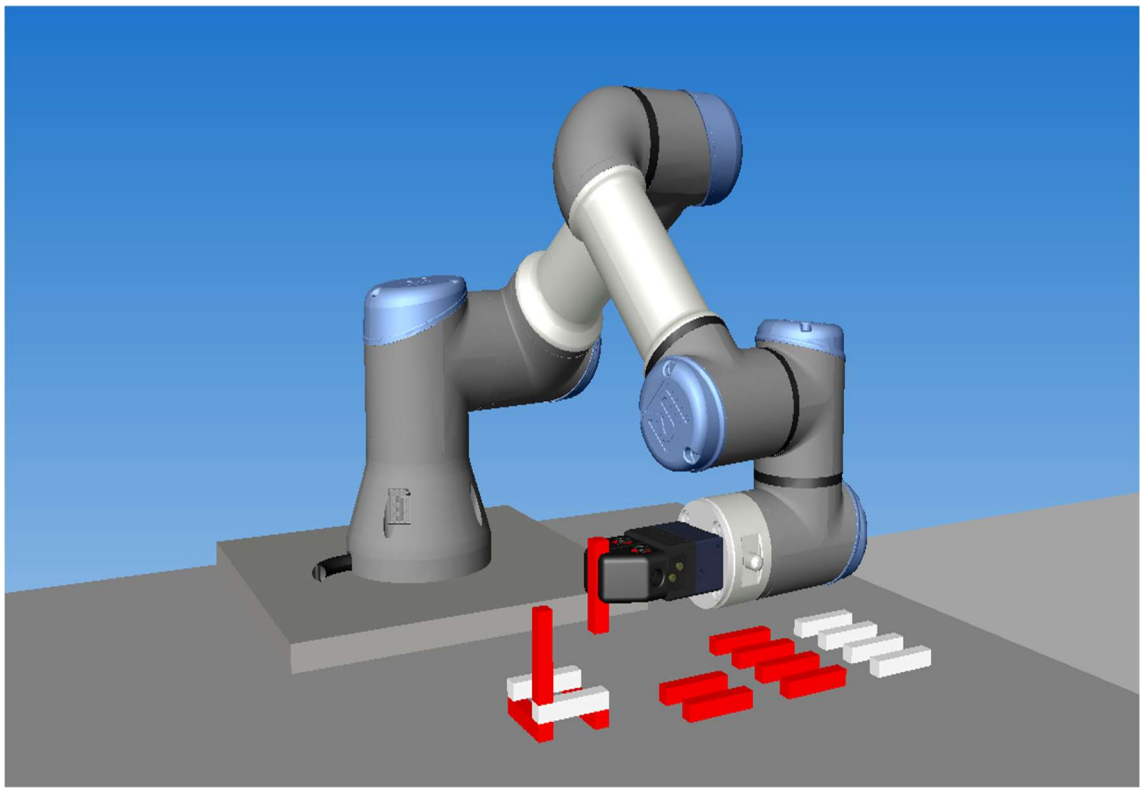
Obrázek 2.29 – Finální pyramida

2.5.3 Stavění z kostek

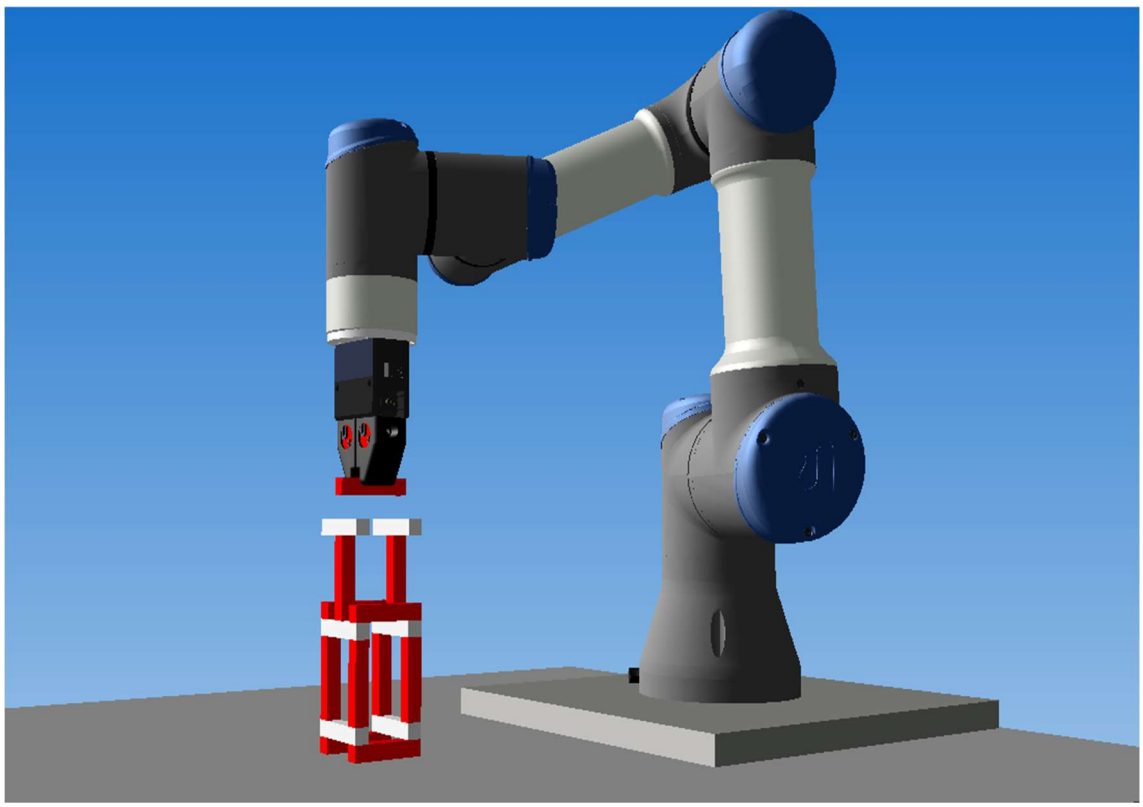
Ve třetí pohybové úloze má rameno za úkol manipulovat s objekty kvádrového tvaru o rozměrech 10 x 10 x 50 mm, které jsou opět umístěny ve své výchozí pozici o známých souřadnicích, viz obrázek 2.30. Úkolem robota je postupným přemístěním objektů na definované pozice vytvořit jednoduchou stavbu. U této úlohy je kladen větší důraz na přesnou a citlivou manipulaci s objekty a ve větší míře se využívá rotace nástroje, viz obrázek 2.31. Kód v Matlabu s pohybovými příkazy je téměř analogický jako v předchozích dvou případech, ale dochází zde také ke změně rotačních souřadnic Rx, Ry a Rz při pokládání některých objektů na výšku. Na obrázku 2.32 je znázorněna finální stavba z kvádrů.



Obrázek 2.30 – Výchozí pozice kvádrů



Obrázek 2.31 – Pokládání kvádru s příslušnou rotací



Obrázek 2.32 – Finální stavba

ZÁVĚR

Diplomová práce je věnována problematice robotických manipulátorů a tvorbě řídicího softwaru v prostředí Matlab, který umožňuje vzdálené ovládání robotického ramene UR3 společnosti Universal Robots.

Navržené řešení pro ovládání ramene je realizováno prostřednictvím TCP/IP síťové komunikace na bázi klient-server a umožňuje komunikovat s ramenem periodicky každých 8 ms. Robotická ramena Universal Robots mají pro účely programování a ovládání své grafické uživatelské rozhraní, pomocí kterého lze rameno programovat a ovládat. Vzdálené ovládání z Matlabu přináší řadu výhod, neboť lze z jednoho prostředí ovládat rameno a také souběžně využívat další možnosti, kterými jsou komunikace s periferními zařízeními, výpočetní výkon, maticové operace, optimalizační funkce, knihovny pro počítačové vidění a strojové učení a podobně. Jistou nevýhodu je omezená frekvence pro řízení rychlých dějů a pohybů, kde tento způsob ovládaní nezajistí plynulost pohybu, tak jak tomu bývá při použití standardních prostředků.

Funkčnost SW pro vzdálené ovládání byla postupně odzkoušena na reálnem zařízení a pomocí simulací testovacích aplikacích uvedených v oddílu 2.4 a také při realizaci jednoduchých úloh v oddílu 2.5.

Předmětem dalšího výzkumu za účelem vylepšení dosavadního řešení by mohlo být vytvoření grafického uživatelského rozhraní pro pohodlnější a přehlednější ovládání ramene a tvorbu programu. Rovněž by se dalo uvažovat o využití jiného klienta, pro zasílání příkazů a čtení informací, nebo také poskytnou uživateli možnost volby klienta.

LITERATURA

Universal Robots [online]. Universal Robots A/S., ©2019 [cit. 2019-10-09]. Dostupné z: https://www.universal-robots.com

ZBÍRALOVÁ, K. ©2014-2019. Automatizace v ČR pohledem statistik: Jakému odvětví vládnou roboty? *Faktory Automation: Magazín o průmyslové automatizaci a robotice* [online]. Czech Republic: FANUC Czech, 26. června 2019 [cit. 2019-10-13]. Dostupné z: https://factoryautomation.cz/automatizace-v-cr-pohledem-statistik-jakemu-odvetvi-vladnou-roboty/

KRÁL, M. ©2019. Problematika hodnocení a užití ruky v pracovním procesu. *Práce a mzda* [online]. Praha 3: Wolters Kluwer ČR, 8. 12. 2016 [cit. 2019-10-19]. Dostupné z: https://www.praceamzda.cz/clanky/problematika-hodnoceni-uziti-ruky-v-pracovnim-procesu

DALE, R. P. a W. F. STEPHEN, 2009. Industrial Process Control Systems. *Industrial Process Control Systems (2nd Edition)* [online]. 2nd. Indian: Fairmont Press [cit. 2019-10-21]. ISBN 978-1-62870-181-4. Dostupné z: https://app.knovel.com/web/toc.v/cid:kpIPCSE001/viewerType:toc//root_slug:industrial-process-control/url_slug:degrees-of-freedom?b-q=Degrees%20of%20freedom%20human&b-subscription=true&b-group-by=true&b-sort-on=default&b-content-type=all_references&include_synonyms=no&issue_id=kt00BYUYO8&hierarchy=

GUPTA, A. K., S. K. ARORA a J. R. WESTCOTT, 2017. *Industrial Automation and Robotics* [online]. Dulles, Virginia: Mercury Learning and Information [cit. 2019-10-26]. ISBN 978-1-5231-0916-6. Dostupné z: https://app.knovel.com/hotlink/toc/id:kpIAR00001/industrial-automation/industrial-automation

KUKA.SystemSoftware. In: *KUKA* [online]. Augsburg: KUKA Aktiengesellschaft, ©2019 [cit. 2019-10-29]. Dostupné z: https://www.kuka.com/cs-cz/produkty,-služby/robotické-systémy/software/systémový-software/kuka_systemsoftware

DUCHOSLAV, P. 5 věcí, které je potřeba zvážit při automatizaci průmyslovými roboty. *Faktory Automation: Magazín o průmyslové automatizaci a robotice* [online]. Czech Republic: FANUC Czech, ©2014-2019, 8. ledna 2017 [cit. 2019-10-30]. Dostupné z:

https://factoryautomation.cz/5-veci-ktere-je-potreba-zvazit-pri-automatizaci-prumyslovymi-roboty/

ABB RobotStudio software extension – Remote Laser Welding Pack. In: *Veletrhy Brno* [online]. Brno: Veletrhy Brno, ©2019 [cit. 2019-10-31]. Dostupné z: https://www.bvv.cz/en/msv/msv-gold-medal/2018/entered-exhibits2/08-softwarova-nastavba-k-abb-robotstudiu-remote-la/

HOW YOU CAN USE COLLABORATIVE ROBOTS. In: *Van Meter* [online]. Iowa: Van Meter, ©2019, 9 April 2019 [cit. 2019-11-07]. Dostupné z: https://www.vanmeterinc.com/blog/how-you-can-use-collaborative-robots.html

UNIVERSAL ROBOTS. In: *MSI TEC* [online]. Colorado: MSI Tec, ©2018 [cit. 2019-11-11]. Dostupné z: https://www.msitec.com/robotics/manufacturers/universal-robots/

Overview of client interfaces - 21744. In: *Universal Robots* [online]. Denmark: Universal Robots A/S., ©2019 [cit. 2019-11-11]. Dostupné z: https://www.universal-robots.com/how-tos-and-faqs/how-to/ur-how-tos/overview-of-client-interfaces-21744/

CHLEBNÝ, J., P. BENEŠ, J. LANGER, J. KRÁL a M. MARTINÁSKOVÁ. *Automatizace a automatizační technika: prostředky automatizační techniky*. 5., rozš. a aktualiz. vyd. Brno: Computer Press, 2014, s. 155-164. ISBN 978-80-251-3747-5.

UNIVERSAL ROBOTS: Uživatelská příručka – UR3/CB3 Překlad originálního návodu. Verze 3.12. ©2019. Dostupné také z: https://s3-eu-west-1.amazonaws.com/ur-support-site/61803/99255_UR3_User_Manual_cs_Global.pdf

Universal Robots. In: *ATN* [online]. Oppach: ATN HÖLZEL, ©2020 [cit. 2020-01-31]. Dostupné z: http://atngmbh.com/en/universal-robots/

HORÁK, J. Základní pojmy síťového softwaru. HORÁK, J. a M. KERŠLÁGER. *Počítačové sítě pro začínající správce*. 5., aktualiz. vyd. Brno: Computer Press, 2011, s. 67-76. ISBN 978-80-251-3176-3.

KOLÍBAL, Z. *Roboty a robotizované výrobní technologie*. Brno: Vysoké učení technické v Brně – nakladatelství VUTIUM, 2016. ISBN 978-80-214-4828-5.

VOJÁČEK, A. Problematika bezpečnosti kolaborativních robotů – ISO/TS 15066. *Automatizace.hw.cz: rady a poslední novinky z oboru* [online]. Praha 4: HW server, ©1997-2014, 17. října 2019 [cit. 2020-04-27]. Dostupné z:

https://automatizace.hw.cz/problematika-bezpecnosti-kolaborativnich-robotu-isots-15066.html

NOF, S. Y. *Handbook of industrial robotics*. New York: Wiley, 1985. ISBN 04-718-9684-5.

SKAŘUPA, J. *Průmyslové roboty a manipulátory* [online], 2007. Ostrava: Vysoká škola báňská – Technická univerzita [cit. 2020-04-01]. ISBN 978-80-248-1522-0. Dostupné z: http://www.elearn.vsb.cz/archivcd/FS/PRM/Text/Skripta_PRaM.pdf

MAURTUA, I., A. IBARGUREN, J. KILDAL, L. SUSPERREGI a B. SIERRA. Human–robot collaboration in industrial applications: Safety, interaction and trust. *International Journal of Advanced Robotic Systems* [online]. July 7, 2017, 14(4) [cit. 2020-04-27]. DOI: 10.1177/1729881417716010. ISSN 1729-8814. Dostupné z: https://journals.sagepub.com/doi/pdf/10.1177/1729881417716010

HLAVÁČ, V. *Kognitivní robotika* [přednáška]. Praha: Fakulta elektrotechnická ČVUT, katedra kybernetiky, Centrum strojového vnímání. Dostupné z: http://people.ciirc.cvut.cz/~hlavac/TeachPresCz/51Robotika/71KognitivniRobotika.pdf

PŘÍLOHY

A – CD

Příloha A

Příloha k diplomové práci

Programová podpora pro práci s robotickým ramenem z prostředí MATLAB

Bc. Dominik Varga

CD

Obsah