ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE
FAKULTA STROJNÍ

Ústav mechaniky, biomechaniky a mechatroniky

BAKALÁŘSKÁ PRÁCE

Model a řízení kinematiky typu DOBOT

Vedoucí práce: Ing. Pavel Steinbauer, Ph.D.

Autor: Jan Hrnčíř

2021

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Hrnčíř** Jméno: **Jan** Osobní číslo: **483140**

Fakulta/ústav: **Fakulta strojní**

Zadávající katedra/ústav: **Ústav mechaniky, biomechaniky a mechatroniky**

Studijní program: **Teoretický základ strojního inženýrství**

Studijní obor: **bez oboru**

II. ÚDAJE K BAKALÁŘSKÉ PRÁCI

Název bakalářské práce:

Model a řízení kinematiky typu DOBOT

Název bakalářské práce anglicky:

Modelling and control of DOBOT kinematics

Pokyny pro vypracování:

- Seznamte se s užívanými kinematikami robotů a zejména s kinematikou typu DOBOT a základními metodami kalibrace mechanismů
- Pro existující robotickou ruku typu DOBOT:
  - Vytvořte přímou a inverzní kinematiku
  - Analyzujte pracovní prostor mechanismu
  - Vytvořte kinematické řízení na platformě Arduino v kartézských nebo cylindrických souřadnicích s komunikací přes sériový port RS232.
- Vytvořte MBS model v prostředí Matlab - Simulink - SimScape Multibody a ověřte dimenzování pohonů
- Dostupnými prostředky proveďte kalibraci daného DOBOTu a ověřte jeho přesnost

Seznam doporučené literatury:

- Valášek, M., Bauma, V., Šika, Z. Mechanika B, Nakladatelství ČVUT v Praze, 2006, Praha
- Firemní literatura Mathworks Matlab, Simulink, SimScape Multibody
- Hamrle, V. Kalibrovatelnost a její použití pro návrh paralelních kinematických struktur, dizertační práce, 2009, ČVUT v Praze
- Arduino - Tutorials [online]. Dostupné z: https://www.arduino.cc/en/Tutorial/HomePage. Citováno 2021/04

Jméno a pracoviště vedoucí(ho) bakalářské práce:

Ing. Pavel Steinbauer, Ph.D., odbor mechaniky a mechatroniky FS

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

Datum zadání bakalářské práce: **28.04.2021** Termín odevzdání bakalářské práce: **13.08.2021**

Platnost zadání bakalářské práce: _____

Ing. Pavel Steinbauer, Ph.D.	doc. Ing. Miroslav Španiel, CSc.	prof. Ing. Michael Valášek, DrSc.
podpis vedoucí(ho) práce	podpis vedoucí(ho) ústavu/katedry	podpis děkana(ky)

III. PŘEVZETÍ ZADÁNÍ

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

Datum převzetí zadání _____ Podpis studenta _____

Anotační list

Jméno autora:	Jan Hrnčíř
Název práce:	Model a řízení kinematiky typu DOBOT
Anglický název:	Modelling and control of DOBOT kinematics
Akademický rok:	2020/2021
Studijní program:	Teoretický základ strojního inženýrství
Studijní obor:	bez oboru
Ústav:	Ústav mechaniky, biomechaniky a mechatroniky Odbor mechaniky a mechatroniky
Vedoucí práce:	Ing. Pavel Steinbauer, Ph.D.
Rozsah práce:	61 stran 47 obrázků 9 tabulek
Klíčová slova:	kinematiky robotů, DOBOT, kinematické řízení, Arduino, metody kalibrace, modelování soustav mnoha těles
Key words:	robot kinematics, DOBOT, kinematics control, Arduino, calibration metods, MBS modeling

Abstrakt

Bakalářská práce se zaobírá řízením existující kinematiky typu Dobot. Je rozdělena do čtyř částí. V první části byla provedena rešerše používaných kinematických struktur a jejich rozdělení. Druhá část pojednává o zařazení Dobotu do tohoto rozdělení a dále je zde vyřešena kinematika Dobotu a jeho pracovní prostor. Ve třetí části je popsáno vytvoření řídícího programu, použitá elektronika, pohony, senzory a kalibrace Dobotu. Poslední část ověřuje dimenzování použitých pohonů pomocí MBS modelu Dobotu.

Abstract

The bachelor thesis deals with the control of existing Dobot kinematics. It is divided into four parts. In the first part, the research is made about the used kinematics structures and their partition. The second part discusses the inclusion of Dobot into the partition, also there is solved kinematics of Dobot, and their workspace. The third section describes the creation of the control program, the electronics used, actuators, sensors, and Dobot calibration. The last part verifies the dimensioning of the used actuators by using the MBS model of Dobot.

Čestné prohlášení

Prohlašuji, že jsem bakalářskou práci na téma: „Model a řízení kinematiky typu DOBOT" vypracoval samostatně s použitím odborné literatury a pramenů v práci uvedených.

V Praze dne: 30. 5. 2021

…….....

(podpis autora)

Poděkování

Rád bych poděkoval vedoucímu své bakalářské práce Ing. Pavlu Steinbauerovi, Ph.D. za cenné rady, věcné připomínky při konzultacích a vypracování bakalářské práce.

V neposlední řadě bych chtěl také poděkovat své rodině a blízkým za podporu po celou dobu studia.

Obsah

1 Úvod

Tato práce se zabývá popisem kinematiky robotické ruky typu Dobot a jejím řízením. Jedná se o hybridní kinematickou strukturu, která kombinuje výhody a nevýhody sériových a paralelních struktur. Vlastnostmi a využitím těchto struktur u průmyslových robotů se zabývá úvodní část práce.

Cílem této bakalářské práce je vytvoření řídícího programu na platformě Arduino, provedení kalibrace robotu a ověření jeho přesnosti. Dále si klade za cíl ověření dimenzování použitých motorů. Za tímto účelem bude vytvořen MBS model Dobotu pro určení potřebných hnacích momentů.

2 Roboty

Slovem robot pojmenoval Karel Čapek umělé bytosti v jeho díle R.U.R. v roce 1920. Od té doby se termín robot používá pro různá mechanická zařízení s určitou mírou autonomie. Je dobré upozornit, že Čapek používal životné skloňování podle vzoru pán, ale v technické terminologii se používá neživotné skloňování podle vzoru hrad. Existuje mnoho definic slova robot. Podle Robotics Institute of America (RIA) „robot je reprogramovatelný multifunkční manipulátor navržený pro přenášení materiálu, součástí, nástrojů, nebo specializovaných zařízení, pomocí variabilně programovaných pohybů k provádění různých úkolů“. [1][2]

Tato kapitola se bude zabývat rozdělením robotů, zejména pak používanými kinematickými strukturami robotů v průmyslu a jejich vlastnostmi.

2.1 Rozdělení robotů

Roboty můžeme rozdělit podle různých kritérií, například podle počtu stupňů volnosti, kinematické struktury, pracovního prostoru, použitých pohonů.

Dle počtu stupňů volnosti[1]

- univerzální – se 6 stupni volnosti
- redundantní – s více než 6 stupni volnosti
- deficitní – s méně než 6 stupni volnosti

Dle kinematické struktury[1]

- sériové
- paralelní
- hybridní (smíšené)

Dle pracovního prostoru[1]

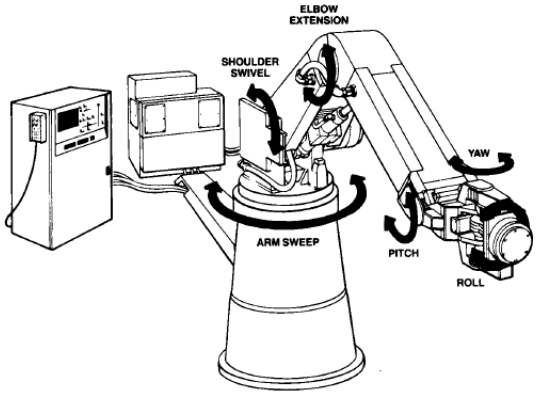
- kartézské
- cylindrické
- sférické
- angulární (úhlový)
- SCARA

Dle pohonů[1]

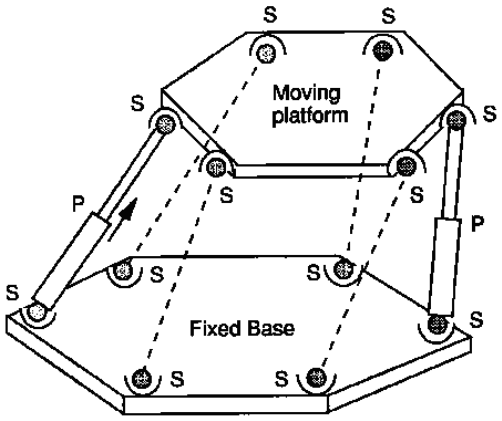
- elektrický
- hydraulický
- pneumatický

2.2 Kinematické struktury robotů

Jak je vidět v odst. 2.1 roboty lze rozdělit podle kinematické struktury na sériové (obr. 1), paralelní (obr. 2) a hybridní. Každá z těchto kinematik má své výhody a nevýhody, podle kterých se hodí k různým aplikacím. Příkladem smíšené kinematické struktury je kinematika typu DOBOT, kterou se bakalářská práce zabývá.



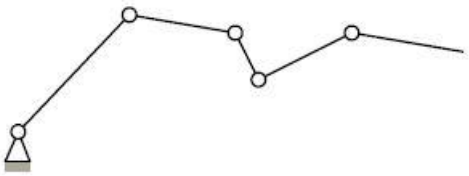
Obrázek 1 Sériová kinematika[2]



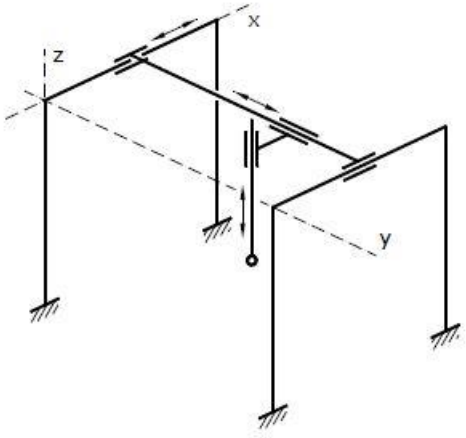
Obrázek 2 Paralelní kinematika[2]

2.2.1 Sériové roboty

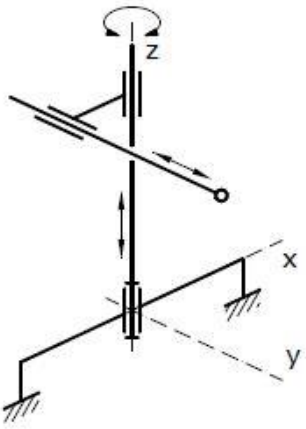
Jsou charakteristické, jak už název napovídá, sériovým spojením několika těles pomocí kinematických vazeb, typicky rotačními (R) a posuvnými (T) vazbami. Každá kinematická vazba náleží jednomu motoru. Jeden konec robotu je přichycen k rámu a druhý je volně polohovatelný v prostoru, tím vzniká otevřený kinematický řetězec (obr. 3). Volný konec se nazývá efektor a jeho poloha v prostoru je dána natočením a posunem kinematických vazeb. Kinematické dvojice můžeme za sebou různě uspořádat a tím vznikne výsledný pohyb efektoru, který vytvoří různé geometrie pracovního prostoru robotu. Podle takto vzniklých pracovních prostorů pojmenováváme kinematické struktury. V průmyslu nejvíce zastoupené kinematické struktury jsou (obr. 4 – 8) : kartézský (TTT), cylindrický (RTT), sférický (RRT), SCARA (RRT), angulární (RRR). V současnosti většina robotů má šest a méně stupňů volnosti, avšak první tři určují pozici efektoru (vytváří pracovní prostor) a zbylé stupně volnosti určují orientaci (natočení) efektoru.[1] [2]



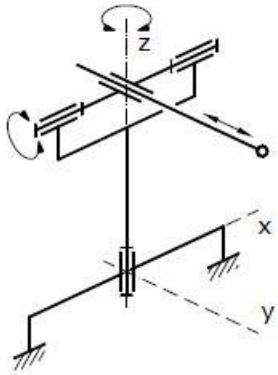
Obrázek 3 Otevřený kinematický řetězec[1]



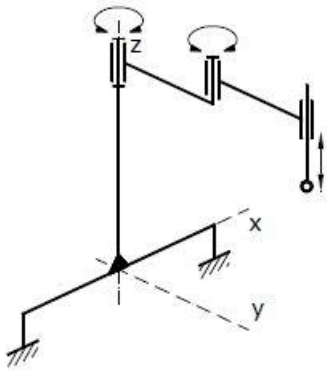
Obrázek 4 Kartézská struktura[1]



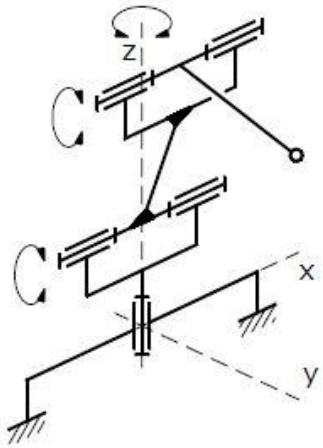
Obrázek 5 Cylindrická struktura[1]



Obrázek 6 Sférická struktura[1]



Obrázek 7 SCARA struktura[1]



Obrázek 8 Angulární struktura[1]

Velkými výhodami sériových robotů je velký pracovní prostor k zastavěné ploše, jednoduché řešení přímé kinematiky. Oproti tomu nevýhodami jsou nízká tuhost, menší přesnost (způsobená sčítáním chyb v jednotlivých kinematických vazbách), nižší rychlost, velká setrvačnost, předchozí člen musí nést všechny následující, tím roste jeho hmotnost, obtížné řešení inverzní kinematiky.[2]

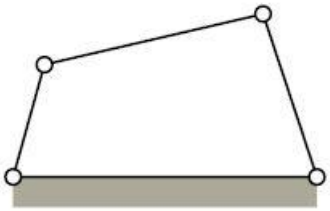
Právě sériové roboty se nejvíce využívají v automobilovém průmyslu (70%), kde vykonávají těžkou repetetivní práci v nebezpečném prostředí, další velký význam mají v paletizaci a obsluze balících linek. Nejčastější operace robotů v automobilovém průmyslu jsou: manipulace s těžkým materiálem, svařování, řezání, povlakování, broušení, leštění. Příklady průmyslových robotů jsou na obr. 9 a obr. 10. [3]

Obrázek 9 Angulární robot KUKA KR 360[4]

Obrázek 10 KUKA KR SCARA[4]

2.2.2 Paralelní roboty

Paralelní roboty se skládají z dvou a více uzavřených kinematických řetězců (příklad uzavřeného kinematického řetězce obr. 11). Efektor, v případě paralelních robotů většinou pohybující se platforma, je připojen k rámu nejméně dvěma nezávislými sériovými kinematickými řetězci, kterým říkáme nohy. Obvykle počet nohou odpovídá počtu stupňů volnosti a každá noha je ovládaná jedním motorem pevně přichyceném k rámu. To umožňuje velké zatížení oproti sériovým robotům, protože se zátěž rozdělí do jednotlivých noh a motorů. [2]



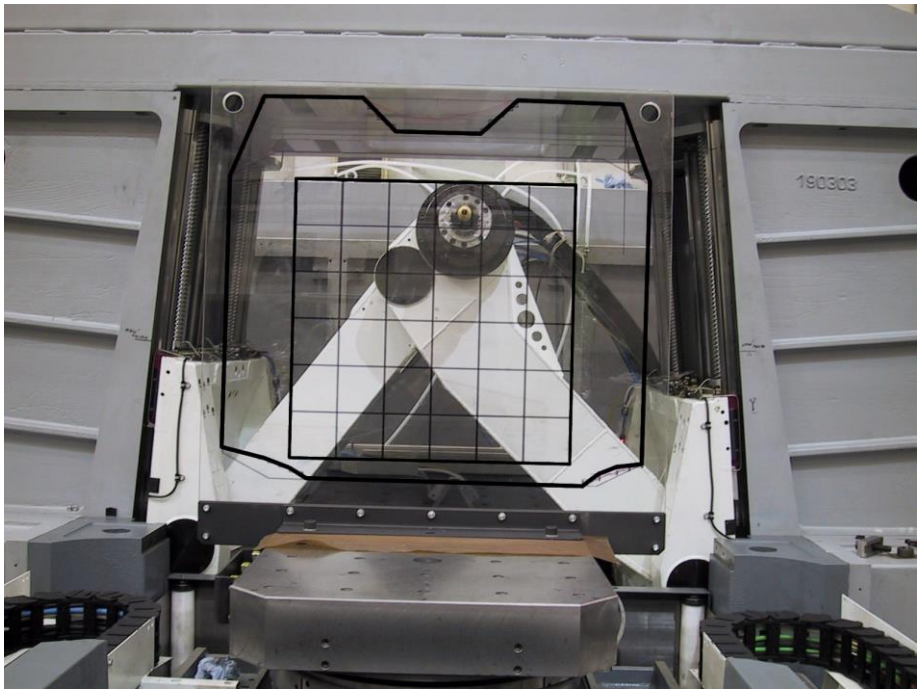
Obrázek 11 Uzavřený kinematický řetězec[1]

Mezi výhody paralelních robotů patří vysoká tuhost, vysoká přesnost polohování (chyby v kinematických vazbách se průměrují), malá setrvačnost, velká rychlost polohování, unesou velká zatížení. Nevýhodami jsou malý pracovní prostor k zastavěné ploše, obtížné řešení přímé kinematiky. [2][3]

Jako příklady paralelních robotů můžeme uvést:

- Stewartova platforma (obr. 2), která se používá u leteckých simulátorů
- Delta robot (obr. 12), pro svoji velkou rychlost využívaný k balení součástek
- Trijoint 900H (obr. 13), obráběcí stroj vyvinutý ve spolupráci KOVOSVIT MAS a.s. Sezimovo Ústí a Ústavu Mechaniky Fakulty strojní ČVUT v Praze[19]
- TRICEPT robot (obr. 14), průmyslový robot používaný ke svařování, řezání vodním paprskem nebo laserem[3]

Obrázek 12 Delta robot od firmy ABB IRB 360 FlexPicker™[5]



Obrázek 13 Trijoint 900H[19]

Obrázek 14 TRICEPT T606[6]

Vlastnost	Sériový robot	Paralelní robot
Pracovní prostor	velký	malý
Šíření chyb	sumace	průměr
Tuhost	nízká	vysoká
Setrvačnost	velká	malá
Přesnost	malá	vysoká

Tabulka 1 Srovnání vlastností seriových a paralelních robotů

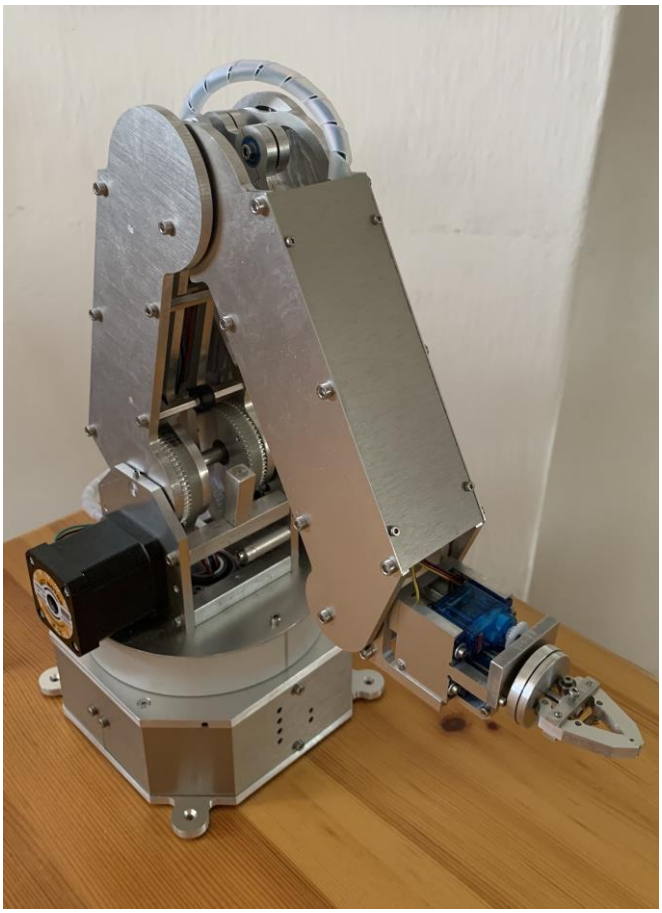
3 Kinematika typu DOBOT

Tato kapitola se zabývá analýzou kinematické struktury typu DOBOT, popsáním její kinematiky a zjištěním pracovního prostoru. Popsání kinematiky robotu a zjištění pracovního prostoru je důležité pro správnou funkci robotu.

3.1 Charakteristika kinematiky

Kinematika typu DOBOT je hybridního typu. Kombinuje tedy výhody a nevýhody paralelních a sériových struktur. Motory jsou umístěny na rámu a přes řemenový převod je poloha efektoru řízena pomocí paralelogramů, tímto je dána velká tuhost, přesnost a rychlost robotu. Nevýhodou je, že jednotlivá táhla paralelogramů se dostávají do kolize s konstrukcí robotu, což limituje jeho pracovní prostor.

Kinematika popsaná v této kapitole vychází z konstrukce firmy DOBOT. Robot se jmenuje DOBOT Magician a firma ho nabízí jako výukového robota automatizace, průmyslu 4.0 a robotiky. Podobnou konstrukci můžeme nalézt i u průmyslových robotů, které nabízejí firmy ABB a KUKA. Jsou využívány hlavně pro paletizaci.



Obrázek 15 Robot popisovaný v bakalářské práci

Obrázek 16 DOBOT Magician[7]

Obrázek 17 KUKA KR 40 PA[4]

Obrázek 18 ABB IRB 760[5]

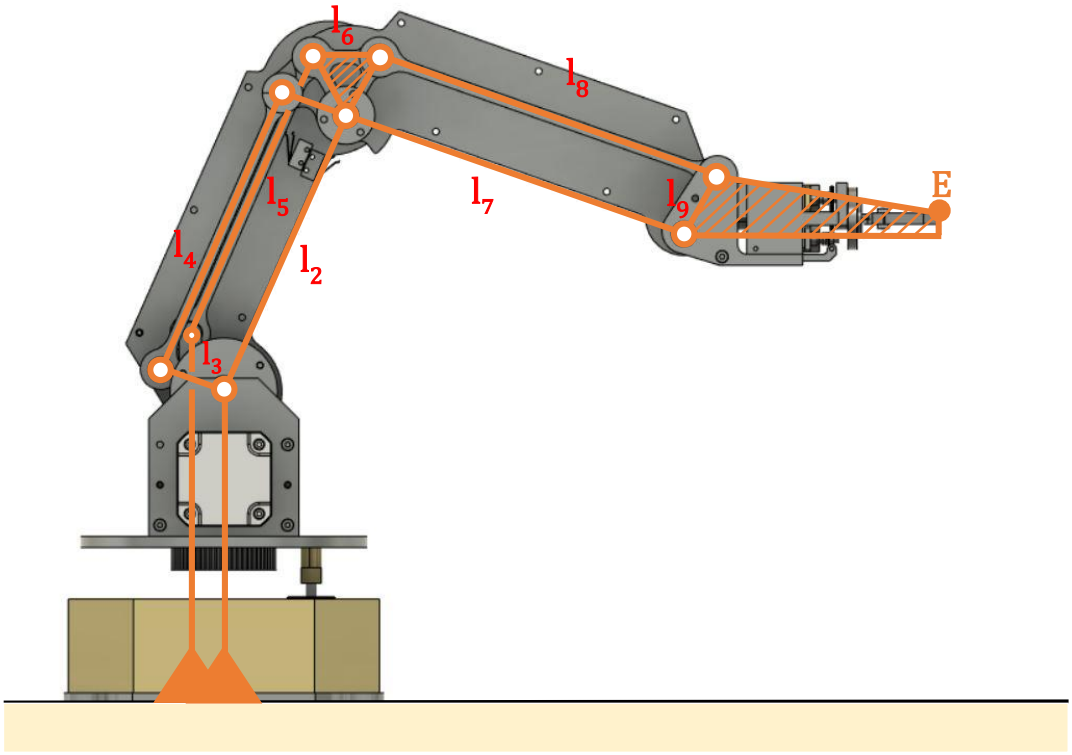
Pro analýzu mechanismů používáme zjednodušené kinematické schéma složené z ideálních tuhých těles a kinematických dvojic. Kinematickou strukturu DOBOTu můžeme modelovat jako rovinný mechanismus (obr. 19) se třemi stupni volnosti. V rovině počet stupňů volnosti n lze ověřit vzorcem, kde u je počet těles s rámem a r je počet rotačních kinematických dvojic.

$$n = 3 \cdot (u - 1) - 2 \cdot r = 3 \cdot (9 - 1) - 2 \cdot 11 = 2° \quad (1)$$

Třetí stupeň volnosti dostaneme při rotaci robotu kolem svislé osy. Jak je patrné z obr. 19 paralelogram složený z těles l_2, l_5 a l_6 udržuje pouze těleso l_9 v poloze rovnoběžné s rámem, proto ho v dalším popisu kinematiky nebudeme uvažovat. Rozměry jednotlivých těles a rozměry použité v odstavci 3.2 a 3.3 byly odměřeny z dostupného počítačového modelu k robotu, viz tabulka.

Těleso	Rozměr [mm]
l2, l4, l5	135
l3, l6, l9	30
l7, l8	160
z1o2	148
r3E	130
z3E	8,6

Tabulka 2 Rozměry robotu z počítačového modelu



Obrázek 19 Kinematické schéma DOBOTu

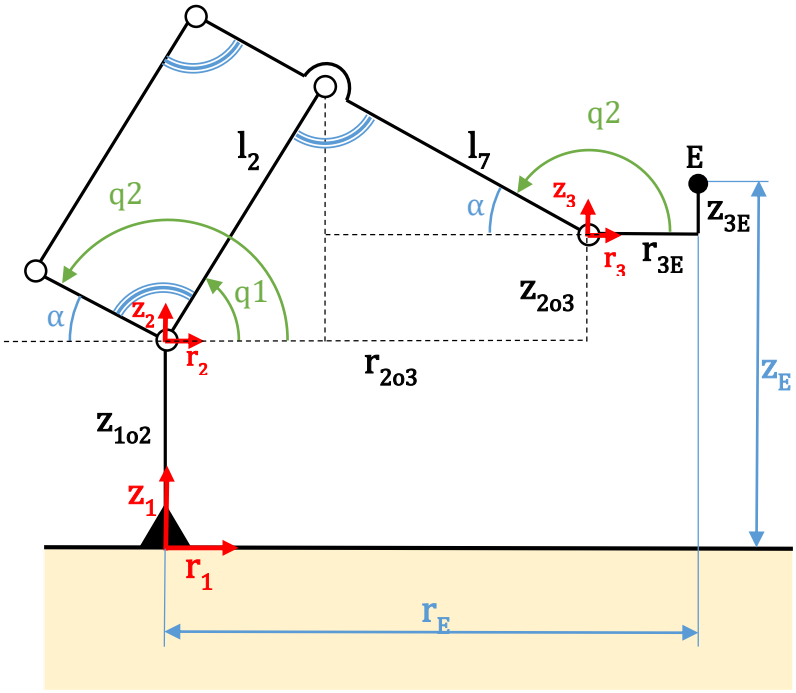
K popisu kinematiky robotu bude použita trigonometrická metoda, která spočívá v rozdělení kinematického schématu na trojúhelníky a za použití trigonometrických vztahů jako je kosinová či sinová věta nalezení vztahu mezi vstupními a výstupními souřadnicemi. Výhodou této metody je její intuitivnost. Naopak nevýhodou je, že nalezení řešení je možné jen pro některé jednodušší mechanismy. Pro složitější mechanismy se používá numerická iterační metoda známa jako vektorová metoda.[8]

Protože se jedná o rovinný mechanismus, který rotuje kolem svislé osy, byl zaveden cylindrický souřadnicový systém (r, z, φ). Souřadnice φ udává natočení robotu kolem osy z a tím přímo i natočení efektoru $\varphi_E = \varphi$, proto dále se při popisu přímé a inverzní kinematiky budeme zabývat jen souřadnicemi r a z.

3.2 Přímá kinematika

Při řešení přímé kinematiky hledáme vztahy pro určení polohy efektoru r_E, z_E při daných úhlových souřadnicích natočení q_1, q_2.

$$[r_E; z_E] = f(q_1; q_2) \tag{2}$$



Obrázek 20 Přímá kinematika

Na obrázku 20 byl zaveden souřadnicový systém spojený s rámem (r_1, z_1), souřadnicový systém v místě působení motorů (r_2, z_2), souřadnicový systém spojený s tělesem l_9 (r_3, z_3), na kterém je umístěn efektor, a pomocný úhel α. Výsledná poloha efektoru je dána průmětem těles l_2, l_7 do os souřadnicového systému (r_2, z_2) a posuvem mezi souřadnicovými systémy (r_1, z_1) a (r_2, z_2), (r_3, z_3) a efektorem. Můžeme tedy psát rovnice:

$$\begin{aligned} r_{\mathrm{E}} &= l_2 \cos(q_1) + l_7 \cos(\alpha) + r_{3\mathrm{E}} \\ z_{\mathrm{E}} &= l_2 \sin(q_1) - l_7 \sin(\alpha) + z_{3\mathrm{E}} + z_{1\mathrm{o}2} \end{aligned} \tag{3}$$

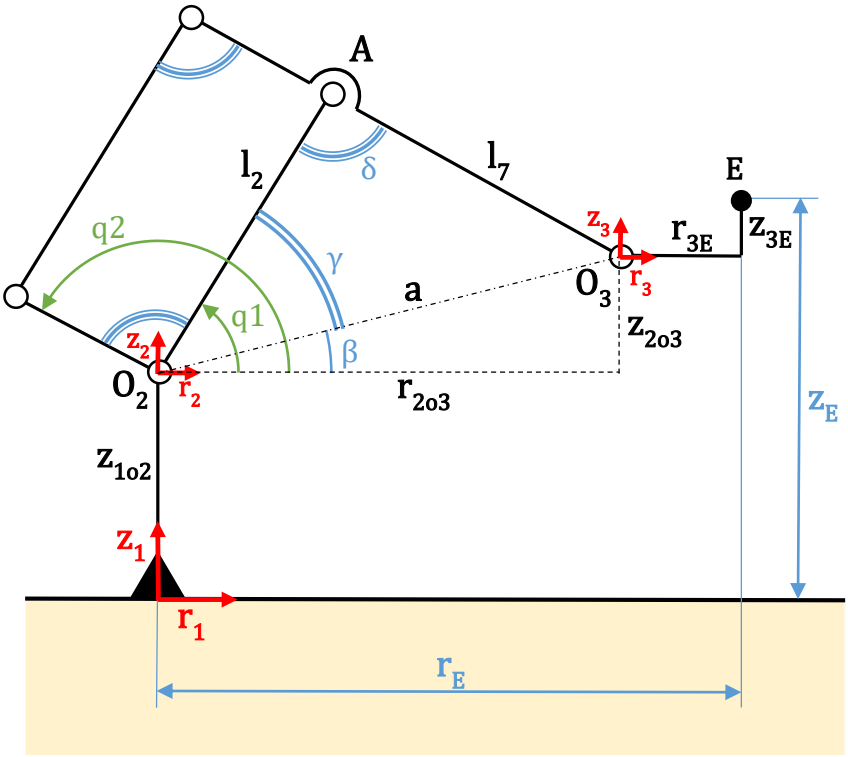
Při vyjádření úhlu α pomocí souřadnice q_2 ($\alpha = 180 - q_2$) a použití součtových vzorců, získáme výsledné rovnice pro popis přímé kinematiky.

$$\begin{aligned} r_{\mathrm{E}} &= l_2 \cos(q_1) - l_7 \cos(q_2) + r_{3\mathrm{E}} \\ z_{\mathrm{E}} &= l_2 \sin(q_1) - l_7 \sin(q_2) + z_{3\mathrm{E}} + z_{1\mathrm{o}2} \end{aligned} \tag{4}$$

3.3 Inverzní kinematika

Inverzní kinematika je opakem přímé kinematiky. Hledáme tedy vztah pro souřadnice natočení při známé poloze efektoru.

$$[q_1; q_2] = f(r_E; z_E) \tag{5}$$



Obrázek 21 Inverzní kinematika

Na obrázku 21 jsou zavedeny souřadnicové systémy jako v případě přímé kinematiky doplněné o pomocné úhly β, γ, δ a pomocnou úsečku *a*. Při odečtení posuvů mezi souřadnicovými systémy (r_1, z_1) a (r_2, z_2), (r_3, z_3) a efektorem od známé polohy efektoru získáme transformaci mezi souřadnicovými systémy (r_2, z_2) a (r_3, z_3).

$$\begin{aligned} r_{2o3} &= r_E - r_{3E} \\ z_{2o3} &= z_E - z_{3E} - z_{1o2} \end{aligned} \tag{6}$$

Ze získané transformace vypočteme z pravoúhlého trojúhelníku úhel β, pomocí Pythagorovy věty úsečku *a*, pomocí kosinové věty úhly γ a δ z trojúhelníku O_2AO_3.

$$a^2 = {r_{2o3}}^2 + {z_{2o3}}^2 \tag{7}$$

$$\beta = \operatorname{atan}\left(\frac{z_{2o3}}{r_{2o3}}\right) \tag{8}$$

$$\gamma = \operatorname{acos}\left(\frac{-l_7^2 + l_2^2 + a^2}{2 \cdot l_2 \cdot a}\right) \tag{9}$$

$$\delta = \operatorname{acos}\left(\frac{-a^2 + l_2^2 + l_7^2}{2 \cdot l_2 \cdot l_7}\right) \tag{10}$$

Z obrázku je zřejmé, že hledané souřadnice natočení jsou dány součtem vypočtených pomocných úhlů.

$$q_1 = \beta + \gamma \tag{11}$$

$$q_2 = q_1 + \delta \tag{12}$$

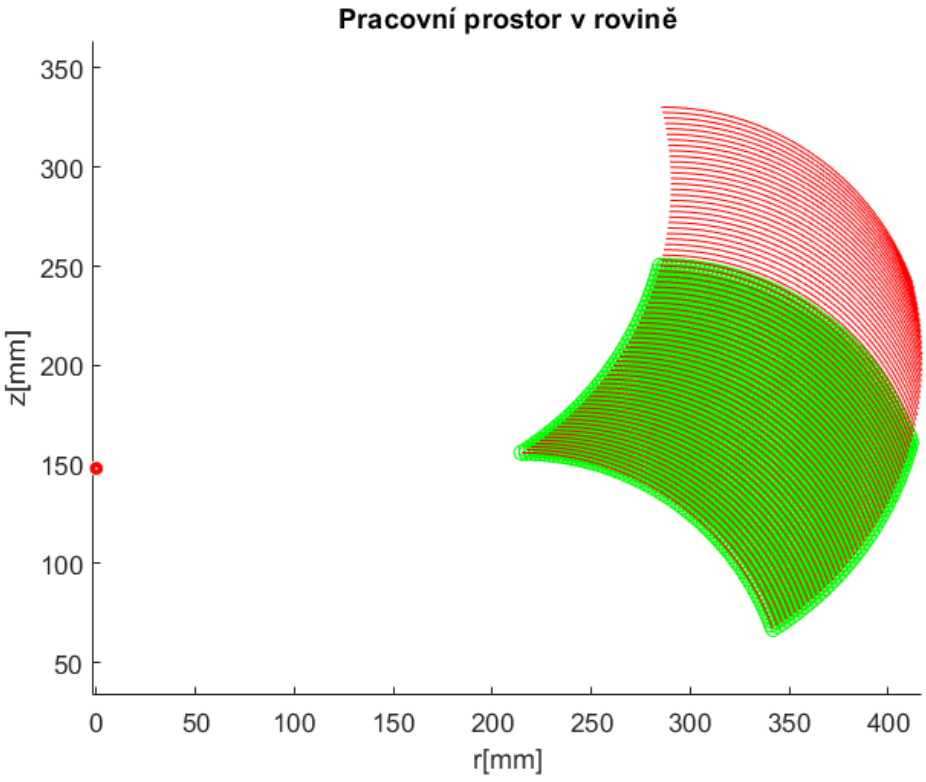
3.4 Pracovní prostor

Pro zjištění pracovního prostoru využijeme rovnic přímé kinematiky popsané výše a odměřených limitů robotu na reálném modelu. Jak již bylo řečeno, tělesa se dostávají do kolize s konstrukcí robotu, proto byly pro maximální natočení souřadnice q_2 odměřeny dvě hodnoty. Hodnota q_2=195° odpovídá maximálnímu natočení při q_1=90° a q_2=166° je maximální natočení při q_1=19°.

Souřadnice	Rozsah natočení [°]
$\mathbf{q_1}$	19 - 90
$\mathbf{q_2}$	122 – 166 (195)
$\boldsymbol{\varphi_E}$	0 - 300

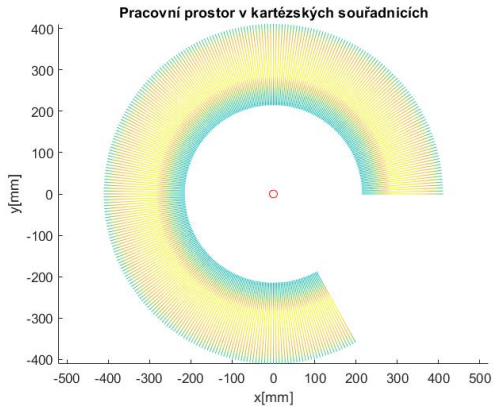
Tabulka 3 Rozsah natočení souřadnic

Výpočtem poloh efektoru z rovnic (4) pro zjištěné rozsahy získáme pracovní prostor. Na obrázku 22 je zobrazen zelenou barvou pracovní prostor, ve kterém nemůže dojít ke kolizi. Červenou barvou je zobrazen prostor, který by byl možný, kdyby mechanismus nebyl limitován svojí konstrukcí.

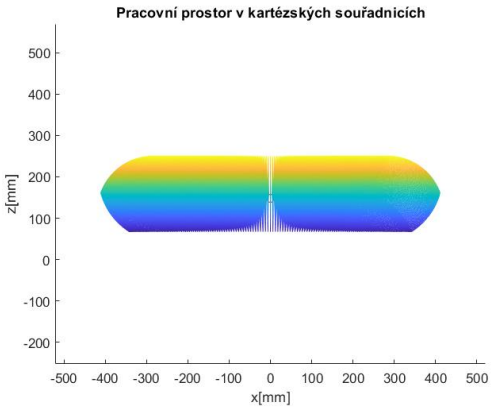


Obrázek 22 Pracovní prostor v rovině

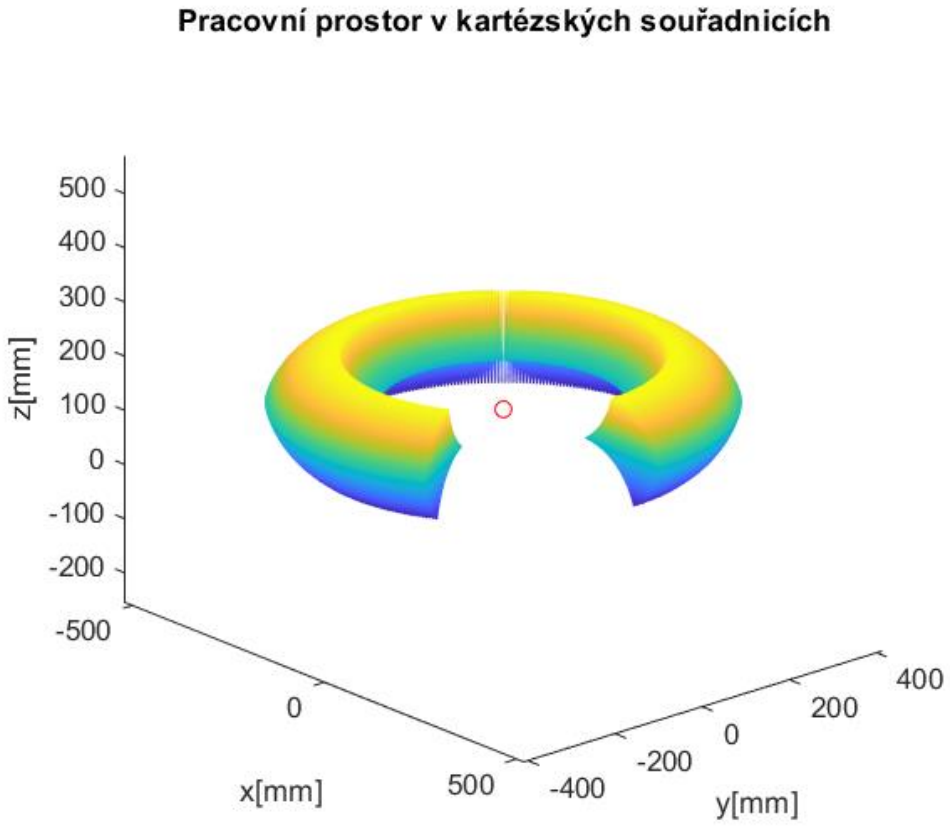
Při rotaci pracovního prostoru v rovině (obr. 22) kolem osy *z* v rozsahu souřadnice φ_E získáme pracovní prostor robotu v kartézských souřadnicích.



Obrázek 23 Pracovní prostor v rovině XY



Obrázek 24 Pracovní prostor v rovině XZ



Obrázek 25 Pracovní prostor v kartézských souřadnicích

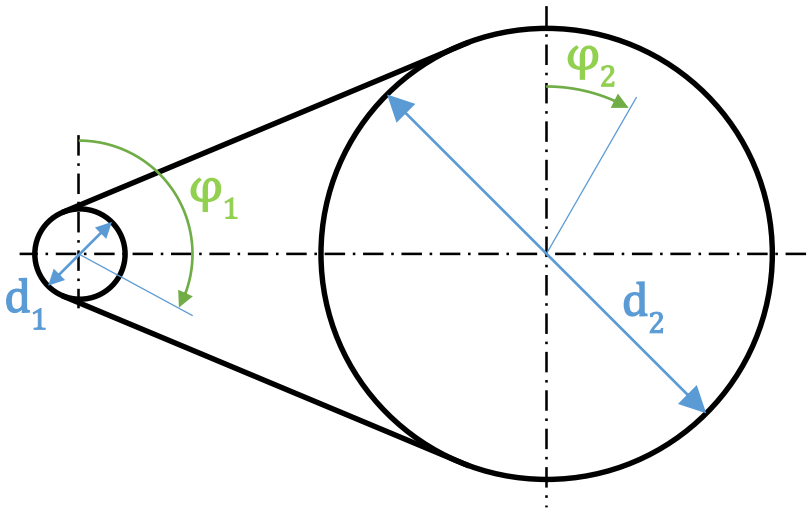
3.5 Převodový poměr

Úhlové souřadnice natočení q_1, q_2 a φ_E jsou ovládány motory třemi shodnými řemenovými převody, proto musíme určit převodový poměr, přes který budeme přepočítávat natočení motorů v závislosti na úhlových souřadnicích. S velkou řemenicí je spojeno ovládané těleso a malá řemenice je připojena na hřídel motoru.

Vyjdeme ze shodnosti kruhového oblouku. Kruhový oblouk daný pootočením malé řemenice o úhel φ_1 se rovná kruhovému oblouku velké řemenice, která se díky řemenu pootočí o úhel φ_2. Při zanedbání mechanických ztrát v řemenu a ložiscích platí rovnice 13, z které můžeme vypočítat převodový poměr i (rovnice 14).

$$r_1 \cdot \varphi_1 = r_2 \cdot \varphi_2 \rightarrow \varphi_1 = \frac{r_2}{r_1} \cdot \varphi_2 \tag{13}$$

$$i = \frac{r_2}{r_1} = \frac{d_2}{d_1} \tag{14}$$



Obrázek 26 Řemenový převod

4 Řízení Dobotu

Součástí této kapitoly je popis elektroniky, která byla použita k řízení robotu, popis řídícího programu, na závěr se zabývá základními metodami kalibrace robotů a provedením kalibrace Dobotu.

4.1 Použitá elektronika

4.1.1 Arduino

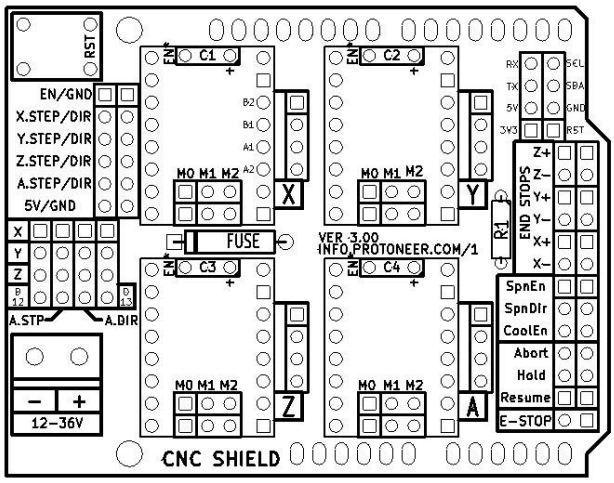
Pro řízení robotu byla použita vývojová deska Arduino UNO s mikrokontrolérem ATmega328. Deska má 14 digitálních vstupů/výstupů, 6 analogových vstupů, Flash paměť 32 KB, EEPROM 1 KB, USB konektor, napájecí konektor, resetovací tlačítko.

Obrázek 27 Arduino UNO[9]

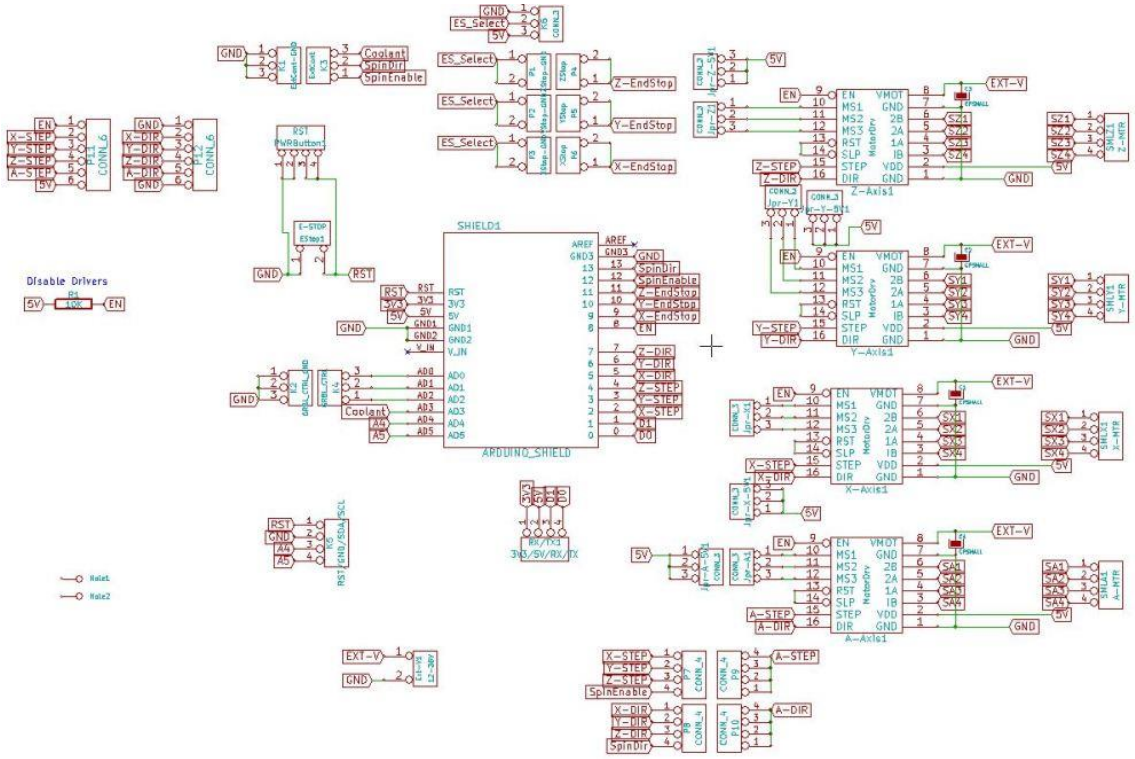
Arduino je open-source platforma. Pro svoji jednoduchost a přístupnost je vhodná pro začátečníky, ale je i dostatečně flexibilní pro pokročilé uživatele. Na stránkách Arduina jsou dostupné jednak výukové materiály tak i komunitní projekty pro inspiraci. Vývojové desky Arduina jsou programovatelné pomocí Arduino Software IDE, který obsahuje textový editor pro psaní kódu, textovou konzoli pro zobrazení chyb při kompilaci, sériový monitor pro komunikaci po sériové lince. [9]

4.1.2 CNC Shield V3

CNC Shield je rozšiřující modul pro Arduino UNO, který se připojuje k pinům základní desky. Jedná se o desku plošných spojů sloužící k jednoduchému připojení driverů krokových motorů, pro které jsou na Shieldu 4 sloty, a koncových spínačů k Arduinu. Na obrázcích je vidět schéma CNC Shieldu a jeho schématické zapojení.



Obrázek 28 CNC Shield V3[10]



Obrázek 29 Schématické zapojení CNC Shieldu[10]

4.1.3 Krokové motory

Krokový motor je stejnosměrný motor, jehož otáčka je rozdělená na daný počet kroků, daných konstrukcí motoru. Je složen ze statoru a rotoru. Stator je nepohyblivá část, na které jsou umístěny cívky kolem dokola rotoru. Rotor je tvořen hřídelí usazenou v kuličkových ložiskách a prstencem permanentních magnetů.

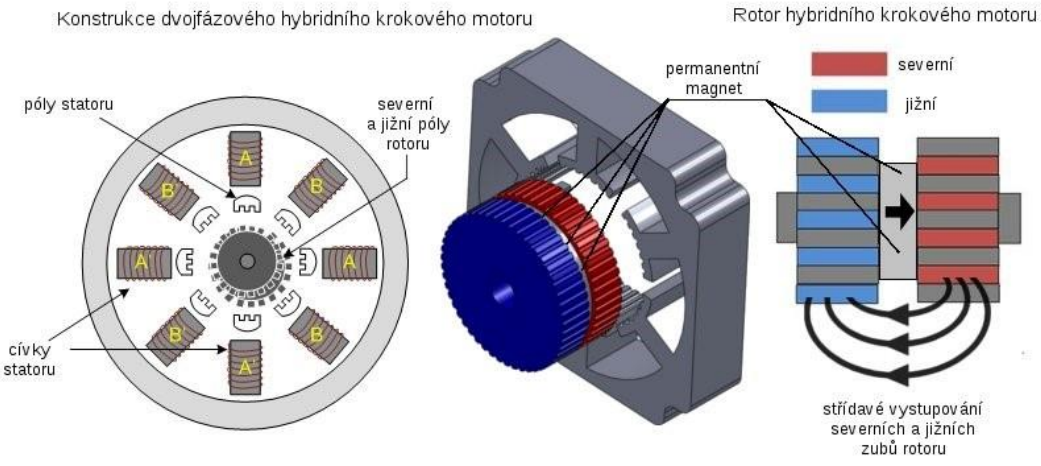
Princip krokového motoru je jednoduchý. Proud procházející cívkou statoru vytváří magnetické pole, které přitáhne opačný pól magnetu rotoru. Vhodným zapojováním cívek dosáhneme vytvoření rotujícího magnetického pole, které otáčí rotorem.[11]

Podle způsobu vinutí a způsobu řízení můžeme krokové motory rozdělit[11]:

1) **Unipolární** – Při unipolárním řízení prochází proud v jednom okamžiku právě jednou cívkou. Motor s tímto buzením má nejmenší odběr, ale také poskytuje nejmenší krouticí moment. Výhodou tohoto řešení je jednoduché zapojení řídící elektroniky - v podstatě stačí jeden tranzistor na každou cívku.
2) **Bipolární** – Při bipolárním řízení prochází proud vždy dvěma protilehlými cívkami. Ty jsou zapojené tak, že mají navzájem opačně orientované magnetické pole. Motor v tomto režimu poskytuje větší krouticí moment, ovšem za cenu vyšší spotřeby proudu.

Podle konstrukce můžeme rozdělit krokové motory[12]:

1) **Reluktanční** – (*reluktance = magnetický odpor*) motory využívají schopnosti magnetického obvodu natáčet se vždy do mechanické konfigurace s co nejmenší energií (co nejmenším magnetickým odporem). Vhodně tvarovaný magnetický obvod se zuby z magneticky vodivého materiálu se vždy natočí těmito zuby pod póly statoru s cívkami (zdroji magnetického toku), aby zvýšil magnetický tok a snížil magnetický odpor (reluktanci) celé konstrukce. Pouštěním proudu do cívek (nejjednodušeji zapínáním a vypínáním cívek) je potom možné s rotorem pohybovat.
2) **S permanentními magnety** – využívá magnetického pole vytvořeného permanentními magnety, v magnetickém obvodu se potom objevuje daleko větší magnetický tok a motor je daleko "živější". Severní a jižní póly pak vytvářejí různě směrované toky po celém motoru a vhodným elektrickým proudem se pak motor roztáčí v různém směru.
3) **Hybridní** – je kombinací obou předchozích. Hybridní krokový motor má vinutí ve statoru, permanentní magnet mezi částmi rotoru.



Obrázek 30 Konstrukce hybridního krokového motoru[12]

Mezi výhody krokových motorů patří snadnost řízení, umožňují velmi precizní polohování bez užití zpětné vazby o změně natočení, stačí totiž počítat kroky. Při malých otáčkách mají velký krouticí moment, který ale klesá s rychlostí otáčení. Disponují přídržným momentem při nulových otáčkách, který drží ovládanou zátěž ve stabilní poloze. Naopak největší nevýhodou krokového motoru je možnost ztráty kroku, pokud je překročena hodnota maximálního krouticího momentu motoru.[11][12]

Dobot je osazen třemi hybridními bipolárními motory. Vlastnosti motoru jsou uvedeny v tabulce.

Krokový motor	ACT MOTOR 17HS5415P1-X6
Přídržný moment	0,55 Nm
Úhel kroku	1,8°
Dovolený proud	1,5 A

Tabulka 4 Vlastnosti motoru [13]

4.1.4 Driver krokového motoru DRV8825

Driver je řídící elektronika krokového motoru. Pomocí logických pulzů na vstupu budí jednotlivá vinutí cívek krokového motoru v přesném pořadí.

DRV8825 je driver pro řízení bipolárního krokového motoru s maximálním proudem na cívce 2,2 A. Tento proud je nastavitelný potenciometrem a musí být nastaven podle maximálního proudu přípustného na motoru, aby nedošlo k jeho zničení. Při připojení motoru se na driveru změří hodnota napětí U mezi zemí a potenciometrem a podle rovnice (15), uváděné výrobcem, se vypočte trvalý proud I na cívce, který se potenciometrem zvýší na hodnotu maximálního proudu na motoru.

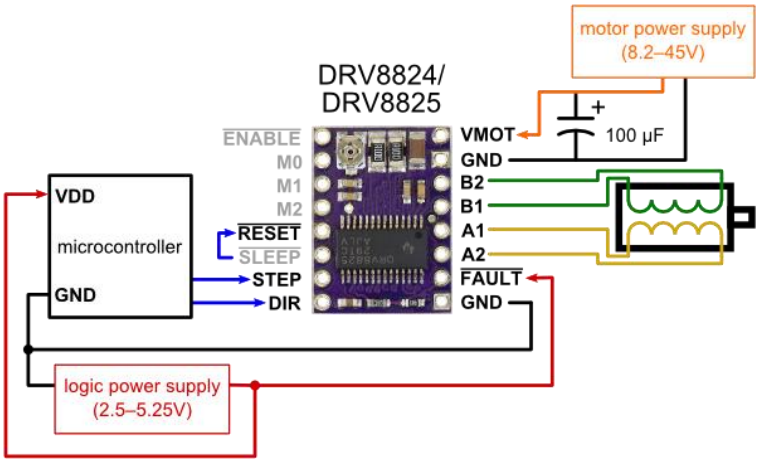
$$I = 2 \cdot U \tag{15}$$

Jak již bylo řečeno, krokové motory mají specifikován počet kroků na otáčku. Ten určuje velikost jednoho kroku, v našem případě 1,8° (plný krok). Driver DRV8825 umožňuje zvýšení rozlišení mikrokrokováním až na 1/32 plného kroku. Mikrokrokování způsobuje, že proudy na cívkách se mění po malých krocích (na jedné cívce proud roste a na druhé klesá) a to umožňuje rozdělení plného kroku na několik menších. Mikrokrokování se na driveru nastavuje pomocí logických vstupů (mode0, mode1, mode2) viz tabulka.

MODE 0	MODE 1	MODE 2	Rozlišení
Low	Low	Low	Plný krok
High	Low	Low	1/2 kroku
Low	High	Low	1/4 kroku
High	High	Low	1/8 kroku
Low	Low	High	1/16 kroku
High	Low	High	1/32 kroku

Tabulka 5 Nastavení mikrokrokování driveru DRV8825

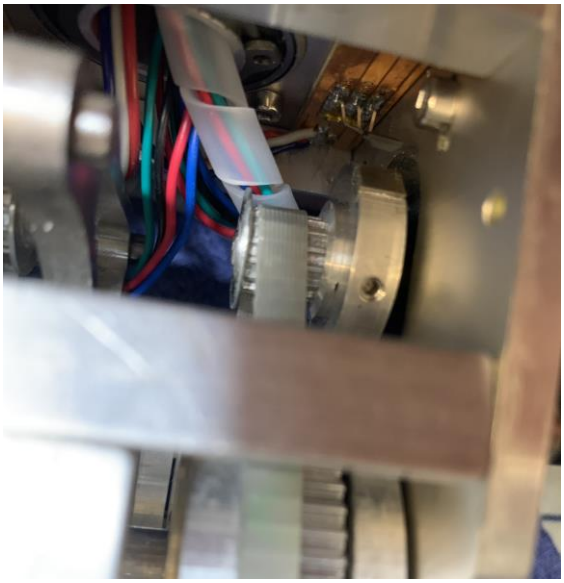
Driver vyžaduje zdroj napětí 8,2 – 45 V. Protože při připojování zdroje vznikají špičky napětí, které by zničily driver, připojuje se paralelně ke zdroji kondenzátor s kapacitou minimálně 47µF. Na obrázku 31 je vidět schéma zapojení driveru pro plný krok. Na pin STEP (z angličtiny krok) jsou přiváděny mikrokontrolérem (Arduinem) logické pulzy a na pin DIR (direction = z angličtiny směr) je přivedena buď logická hodnota 1 (HIGH) pro směr otáčení po směru hodinových ručiček nebo logická hodnota 0 (LOW) pro opačný směr otáčení. [14]



Obrázek 31 Schéma zapojení driveru pro plný krok[14]

4.1.5 Koncové snímače

Pro získání informace o koncové poloze robotu byly použity dva druhy snímačů mechanický spínač a Hallův senzor.



Obrázek 32 Hallův senzor

Obrázek 33 Mechanický spínač

4.1.5.1 Hallův senzor

Je tenká polovodičová destička typy P využívající Hallův jev. Destičkou prochází proud a v blízkosti magnetického pole, které způsobí přesun nosičů náboje (děr a elektronů) na opačné strany polovodiče, vzniká napříč destičkou malé napětí. Toto napětí můžeme měřit a zjišťovat tak přítomnost magnetického pole.[15]

Na robotu jsou Hallovy senzory umístěny u malých řemenic, které jsou připojeny na krokové motory a je na nich nalepen permanentní magnet tak, že když se robot pootočí do koncové polohy, dostane se magnet k Hallovu senzoru, na kterém vznikne napětí. Informace o přítomnosti magnetického pole je poté zpracována v řídícím programu.

4.1.5.2 Mechanický spínač

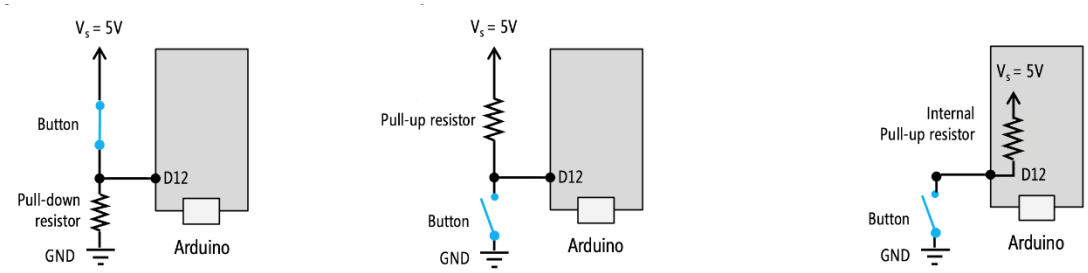
Je mechanická součástka, která spíná a rozpíná elektrický obvod. Pokud bychom ho zapojili pouze mezi zdroj 5V a vstupní snímací pin, v rozpojeném stavu spínače by se vodič připojený ke vstupnímu pinu choval jako anténa a docházelo by k elektromagnetickému rušení. Tomuto stavu říkáme, že pin „plave“ a vstupní hodnota je nepředvídatelná. Proto musíme tento vodič spojit se zemí přes rezistor. Kdybychom tak neudělali, došlo by ke zkratu a tím k zničení Arduina. Toto zapojení můžeme realizovat dvěma způsoby, a to v uspořádání s PULL - UP nebo s PULL - DOWN rezistorem. Názvy rezistorů jsou pojmenovány podle hodnoty na vstupním pinu ve stavu, kdy je spínač rozpojený.

Stav spínače	Vstupní pin, zapojení PULL-DOWN	Vstupní pin, zapojení PULL-UP
rozpojený	LOW	HIGH
sepnutý	HIGH	LOW

Tabulka 6 Logika zapojení Pull-Up a Pull-Down

Způsoby zapojení jsou na obrázku 34. Třetí případ zapojení na obrázku je stejný způsob zapojení jako s PULL-UP rezistorem s tím rozdílem, že nepotřebujeme externí rezistor, neboť je použit rezistor přímo na desce Arduina. Tento způsob zapojení je použit při řízení Dobotu. [16]

Jeden kontakt spínače se připojí k zemi a druhý k digitálnímu pinu, který je připojen přes vnitřní PULL-UP rezistor ke zdroji napětí. V rozpojeném stavu je digitální pin spojen přes PULL-UP rezistor ke zdroji napětí a je na něm měřená hodnota odpovídající logické 1 (HIGH). Při sepnutí spínače se uzavře obvod se zemí a hodnota na digitálním pinu se změní na logickou 0 (LOW).



Obrázek 34 Způsoby zapojení spínače[16]

4.2 Zapojení

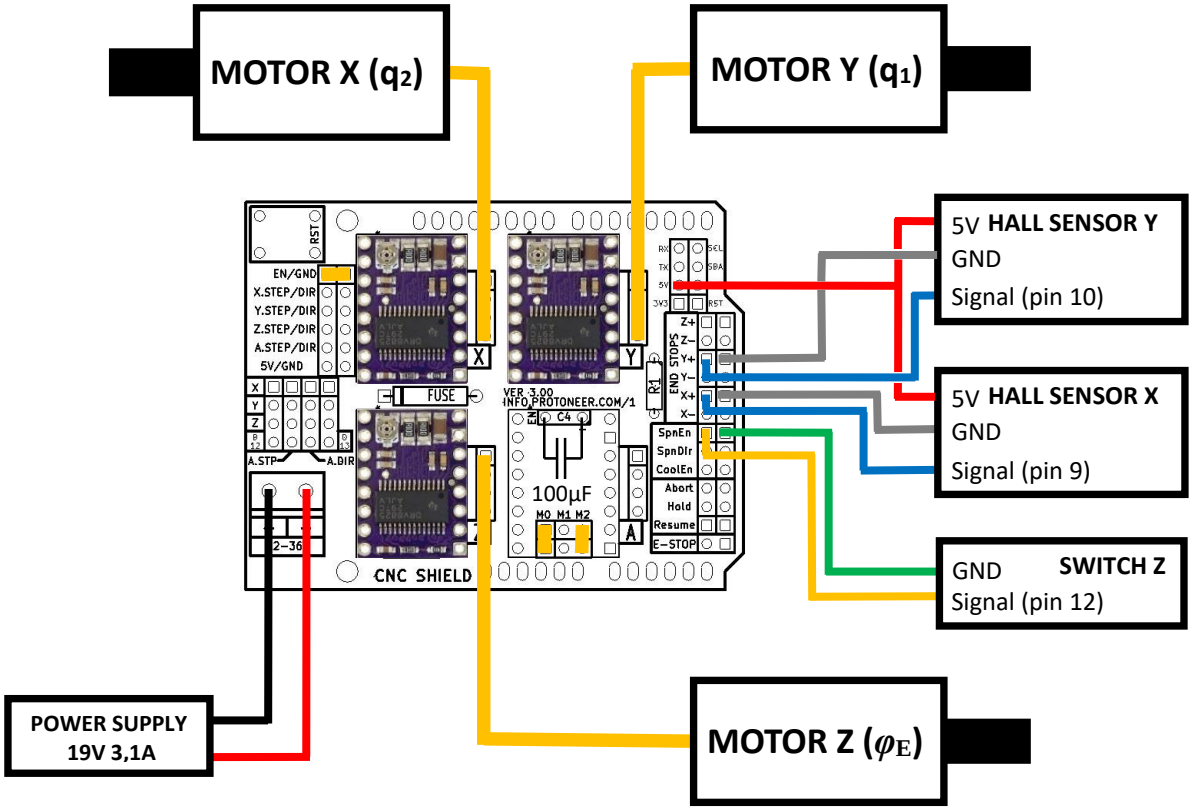
Výše popsané snímače a drivery, ke kterým jsou připojeny krokové motory, jsou připojeny přes CNC Shield k Arduinu dle schématu zapojení (obr. 35). Arduino je připojeno k počítači přes USB kabel, který zajišťuje logické napájení Arduina a pomocí převodníku USB-RS232, jež Arduino obsahuje, sériovou komunikaci. K CNC Shieldu je ještě připojen externí zdroj napájení, který napájí výkonovou část obvodu (vinutí krokových motorů). Na CNC Shieldu jsou již umístěny kondenzátory 100µF důležité pro drivery při připojování externího zdroje, jak je popsáno v odst. 4.1.4.

Pomocí jumperů je přivedena logická hodnota 1 (HIGH) na piny M0 a M2, tím jsou nastaveny (dle odst. 4.1.4) drivery krokových motorů pro větší rozlišení polohování v módu 1/32 plného kroku. To v kombinaci s motorem (200 kroků při plném kroku) dává 6400 kroků na jednu otáčku.

Protože na pin EN (enable) na CNC Shieldu je přes PULL-UP rezistor R1 přivedeno napětí 5V, je pin EN jumprem spojen se zemí, tím na pinu EN na driveru bude logická hodnota 0 (LOW) požadovaná pro aktivní stav driveru. Kdyby byla na pinu EN logická hodnota 1 (HIGH), driver by byl deaktivovaný a vstupní pulzy by byly ignorovány.

Krokové motory a k nim příslušné koncové snímače mají označení X Y Z, protože CNC Shield je určen pro kartézský souřadnicový systém. Toto značení je zachováno

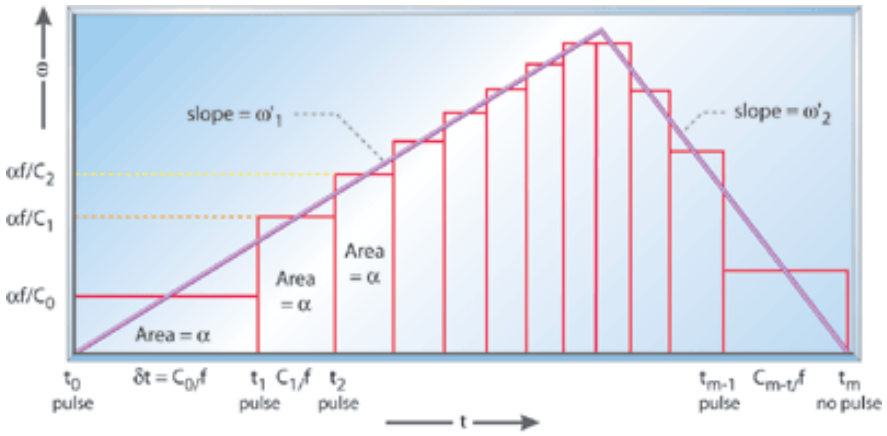
z důvodu snadné orientace při zapojování a kontrole vstupních/výstupních pinů podle schématu zapojení CNC Shieldu (odst. 4.1.2) při psaní programu.



Obrázek 35 Schéma zapojení

4.3 Program

Pro řízení robotu byly vytvořeny dva programy, pro přímé a inverzní řízení. Programy využívají knihovnu AccelStepper[17], která umožnuje řízení krokových motorů s driverem nezávisle na sobě. Pomocí příkazů setMaxSpeed(), setAccelleration(), moveTo() můžeme nastavit motoru jeho rychlost, zrychlení a absolutní polohu. Dále funkce run() krokuje motor, dokud není v požadované poloze. AccelStepper vypočítává čas kroku v mikrosekundách po každém kroku na základě nastavené rychlosti, zrychlení a počtu kroků, o které se má pootočit. Jak je vidět na obrázku 36, kde je znázorněna akcelerace a decelerace po rampě, zrychlení je simulováno zkracujícím se časovým intervalem jednotlivých kroků. Postup výpočtu rychlostního profilu je popsán v [18].



Obrázek 36 Rychlostní profil simulovaný délkou kroků (pulzů) [18]

4.3.1 Přímé řízení

Program tvořený v Arduino IDE se skládá z definice používaných proměnných, knihoven, funkce setup(), která po spuštění programu proběhne jednou a slouží k inicializaci proměnných, nastavení sériové komunikace, nastavení pinů jako vstupní resp. výstupní, a funkce loop(), jedná se o nekonečnou smyčku, která umožňuje změnu hodnot proměnných a je použita k aktivnímu řízení Arduina. [9]

Jako první je tedy nadefinována použitá knihovna AccelStepper, výstupní piny pro řízení krokových motorů, vstupní piny koncových snímačů, proměnné pro sériovou komunikaci, pozici motorů v krocích a jejich limitů, a převodový poměr zjištěný v následujícím odstavci 4.3.2 právě pomocí přímého řízení.

```
#include <AccelStepper.h>

#define motorInterfaceType 1
//Motor X
#define dirX 5
#define stepX 2
//Motor Y
#define dirY 6
#define stepY 3
//Motor Z
#define dirZ 7
#define stepZ 4

//Hall sensor
#define hallSensorX 9
#define hallSensorY 10
//switch Z sensor
#define zSwitch 12
//serial communication
String rxString = "";
String data[3];
//variables for position effector in STEPS
int stepperXPosition = 3500;
int stepperYPosition = -200;
int stepperZPosition = 1200;

const float phiToSteps = 75.959595; //(47*6400)/(11*360)

//Limits of robot in STEPS
//join q2
const int xMAX = 5500; //122°
const int xMIN = 2200; //166°
//join q1
const int yMAX = 5600; //19°
const int yMIN = 0;
//join phiE
const int zMAX = 13672; //180°
const int zMIN = 0;
```

Dále jsou vytvořeny objekty krokových motorů. První parametr udává, že krokový motor je řízen driverem pomocí pinů STEP a DIR, které jsou další dva parametry.

```
AccelStepper stepperX = AccelStepper(motorInterfaceType,stepX,dirX);
AccelStepper stepperY = AccelStepper(motorInterfaceType,stepY,dirY);
AccelStepper stepperZ = AccelStepper(motorInterfaceType,stepZ,dirZ);
```

Ve funkci setup() je nastavena sériová komunikace s rychlostí 9600 bitů za sekundu. Na sériový port je vypsána informace o očekávaných vstupech. Dále jsou nastaveny piny koncových spínačů jako vstupní a motorům je nastavena maximální rychlost v krocích za sekundu a zrychlení v krocích za sekundu na druhou. Parametrem INPUT_PULLUP je pin mechanického snímače nastaven jako vstupní a je připojen přes vnitřní PULL-UP rezistor se zdrojem podle způsobu zapojení popsaném v odst. 4.1.5.2. Ještě je u každého objektu krokového motoru nastavena minimální šířka pulzu v mikrosekundách, která je povolena driverem (tato hodnota je uvedena výrobcem[14]). Poslední příkaz spustí funkci homing().

```
void setup() {
  Serial.begin(9600);
  Serial.println("Hello! Please enter position (q2,q1,phiE)[deg]:");

  pinMode(hallSensorX,INPUT);
  pinMode(hallSensorY,INPUT);
  pinMode(zSwitch,INPUT_PULLUP);

  stepperX.setMaxSpeed(2000);
  stepperX.setAcceleration(2000);
  stepperX.setMinPulseWidth(2);
  stepperY.setMaxSpeed(2000);
  stepperY.setAcceleration(2000);
  stepperY.setMinPulseWidth(2);
  stepperZ.setMaxSpeed(2000);
  stepperZ.setAcceleration(2000);
  stepperZ.setMinPulseWidth(2);

  homing();
}
```

Protože při zapnutí robotu neznáme polohu natočení motorů, je nutné nejdříve motory inicializovat do výchozí pozice. K tomu slouží funkce homing(). Motory se inicializují postupně. Nejdříve se inicializuje motor ovládající souřadnici q_1, pak q_2 a nakonec φ_E. Postup je pro všechny motory stejný. Motor je roztočen a točí se, dokud není detekován snímačem v koncové poloze, kde mu je nastavena výchozí pozice. Následně je pootočen v opačném směru na pozici v pracovním prostoru.

```
 //Homing stepperY (q1)
  while(digitalRead(hallSensorY) == HIGH){
    stepperY.setSpeed(1800);
    stepperY.runSpeed();
  }
  stepperY.setCurrentPosition(0);
  delay(1000);
  stepperY.moveTo(-200);
  while(stepperY.currentPosition() != -200){
    stepperY.run();
  }
```

Pro přijmutí informace, do jaké polohy od výchozí pozice se mají motory otočit, je vytvořena funkce receiveData(). Veškeré informace o natočení ve stupních jsou přes sériový port přijaty jako jeden textový řetězec, který je pomocí znaku „čárky“ odseparován. Jednotlivá data jsou samostatně uložena do textového pole a vypsána po sériovém portu.

```
void receiveData(){
  if(Serial.available()>0){
    delay(2);
    rxString = Serial.readString();
    int stringStart = 0;
    int arrayIndex = 0;
    for (int i = 0; i  < rxString.length(); i++) {
            if (rxString.charAt(i) == ',') {
                data[arrayIndex] = ""; // deletes previous
                data[arrayIndex] = rxString.substring(stringStart, i);
                stringStart = (i + 1);
                arrayIndex++;
            }
    }
    Serial.print("q2[°]= ");
    Serial.print(data[0]);
    Serial.print("\tq1[°]= ");
    Serial.print(data[1]);
    Serial.print("\tq3[°]= ");
    Serial.println(data[2]);
```

Dále natočení ve stupních jsou přepočteny převodovým poměrem na počet kroků. Vypočtené kroky jsou zkontrolovány, zda jsou realizovatelné robotem, tedy zda jsou v krokových limitech motorů. Pokud ano jsou kroky uloženy do proměnných určených pro informaci o pozici motoru. Kroky motoru Y jsou uloženy se znaménkem mínus, protože motor je na robotu orientován tak, že jeho absolutní poloha natočení je určena proti směru hodinových ručiček. Ke krokům motoru Y je ještě připočteno 200 kroků, protože při tomto natočení byla souřadnice q_1 odměřena a odpovídá 90°. Když nejsou kroky v limitech, je vypsána informace, že zadané souřadnice jsou mimo pracovní prostor, nejsou realizovatelné a je vypsána aktuální pozice.

```
    //position in STEPS for check
    int x = phiToSteps * data[0].toFloat();
    int y = (phiToSteps * data[1].toFloat()) + 200;
    int z = phiToSteps * data[2].toFloat();
```

```
    if(x>=xMIN && x<=xMAX && y>=yMIN && y<=yMAX && z>=zMIN && z<=zMAX){
      stepperXPosition = x;
      stepperYPosition = -y;
      stepperZPosition = z;
      Serial.print("positionX[STEPS]= ");
      Serial.print(stepperXPosition);
      Serial.print("\tpositionY[STEPS]= ");
      Serial.print(stepperYPosition);
      Serial.print("\tpositionZ[STEPS]= ");
      Serial.println(stepperZPosition);
    }
    else{
      Serial.println("Coordinates are out of the workspace! They were
not realized! CURRENT POSITION:");
      Serial.print("positionX[STEPS]= ");
      Serial.print(stepperXPosition);
      Serial.print("\tpositionY[STEPS]= ");
      Serial.print(stepperYPosition);
      Serial.print("\tpositionZ[STEPS]= ");
      Serial.println(stepperZPosition);
    }
    Serial.println("\t---------------------");
  }
}
```

Nekonečná smyčka loop() obsahuje výše popsanou funkci receiveData(). Po získání informace o požadované pozici motorů, jsou tyto hodnoty předány motorům příkazem moveTo() a motory jsou příkazem run() krokovány, dokud nejsou v požadované pozici.

```
void loop() {
  receiveData();
  delay(1000);
  stepperX.moveTo(stepperXPosition);
  stepperY.moveTo(stepperYPosition);
  stepperZ.moveTo(stepperZPosition);
  while(stepperX.currentPosition() != stepperXPosition ||
stepperY.currentPosition() != stepperYPosition ||
stepperZ.currentPosition() != stepperZPosition){
    stepperX.run();
    stepperY.run();
    stepperZ.run();
  }
}
```

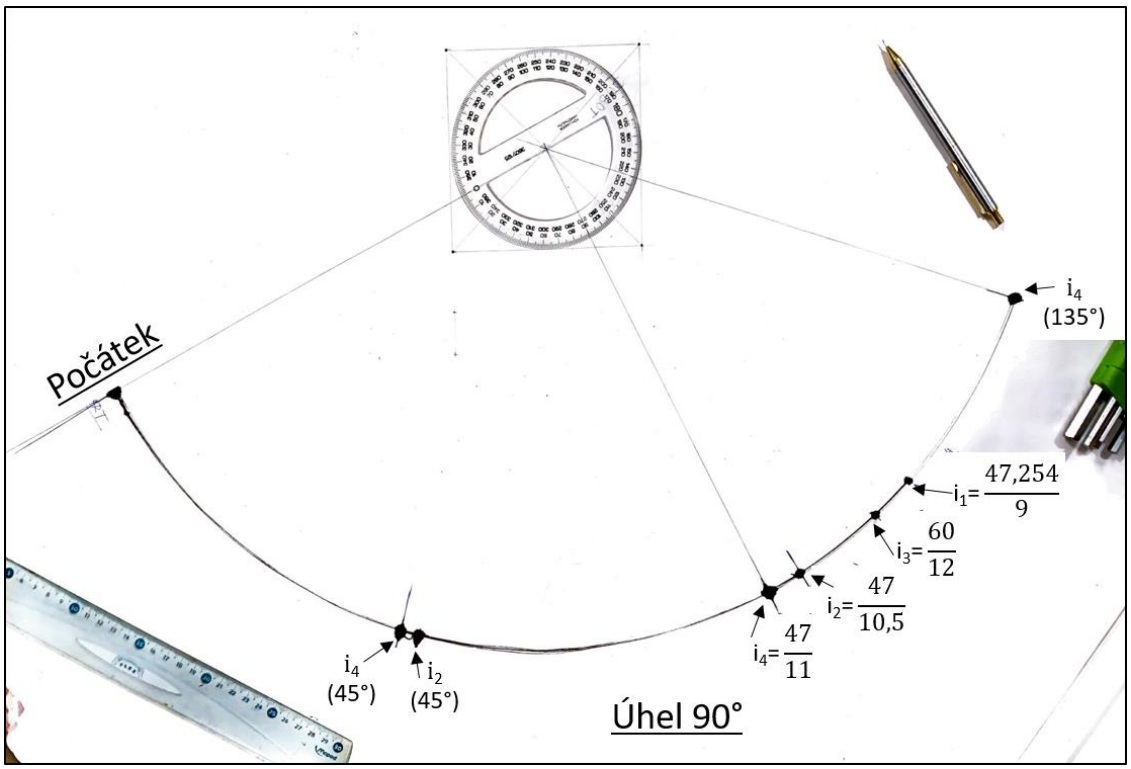
4.3.2 Validace převodového poměru

Protože poloha natočení motoru se v programu udává v krocích, musíme doplnit výpočet převodového poměru z odst. 3.5 o přepočet z úhlů na kroky ze znalosti, že jedna otáčka (360°) je rovna 6400 krokům.

$$i_{STEPS} = \frac{d_2}{d_1} \cdot \frac{6400}{360} \tag{16}$$

Byly vypočteny převodové poměry z rozměrů hlavových kružnic řemenic d_2 a d_1 odměřených z počítačového modelu (i_1), pomocí posuvného měřidla na reálném robotu (i_2), a byl také určen převodový poměr z počtu zubů řemenic (i_3). Robot byl přímým řízením s jednotlivými převodovými poměry natočen do polohy 90°. Nejblíže

skutečnému převodu byl převod s rozměry odměřenými přímo na reálném robotu. Tento převod byl dále technickým citem upraven na převod (i_4). Na obrázku 37 je vidět měření, na kterém byla ověřena správnost převodového poměru. V ose otáčení byla sestrojena kolmice k počáteční poloze, které náleží bod na kružnici získaný převodem i_4. Úhloměr je zde použit jako orientační stupnice.



Obrázek 37 Validace převodového poměru

4.3.3 Inverzní řízení

Program napsaný pro inverzní řízení funguje stejně jako již popsaný program pro přímé řízení s tím rozdílem, že ve funkci receiveData() se vypočítávají souřadnice natočení pomocí inverzní kinematiky (odst. 3.3) ze vstupních hodnot polohy efektoru. Pro úplnost je celý kód programu přiložen jako příloha práce.

```
//inverse kinematics
 float rE = data[0].toFloat();
 float zE = data[1].toFloat();
 float phiE = data[2].toFloat();
 float r2 = rE - r3E;
 float z2 = zE - z3E - z1o2;
 float a = sqrt(r2*r2 + z2*z2);
 double beta = atan(z2/r2);
 double gama = acos((-l_7*l_7+l_2*l_2+r2*r2+z2*z2)/(2*l_2*a));
 double delta = acos((-r2*r2-z2*z2+l_2*l_2+l_7*l_7)/(2*l_2*l_7));
 double q1 = beta + gama;
 double q2 = q1 + delta;
 Serial.print("q2[rad]= ");
 Serial.print(q2);
 Serial.print("\tq1[rad]= ");
 Serial.print(q1);
 Serial.print("\tphiE[deg]= ");
 Serial.println(phiE);
```

Ze znalosti odměřených souřadnic natočení v koncových polohách jsou vypočteny absolutní pozice motorů od koncové polohy v krocích. Následně je zkontrolována realizovatelnost požadované pozice a hodnoty jsou uloženy do proměnných tak, jak bylo popsáno u přímého řízení.

```
    //Home position in degrees
     float q1Home = 90;
     float q2Home = 195;
     Serial.print("\tq2Home[rad]= ");
     Serial.print(q2Home/180*M_PI);
     Serial.print("\tq1Home[rad]= ");
     Serial.println(q1Home/180*M_PI);
     //position in STEPS for check
     float x = phiRadToSteps * ((q2Home/180*M_PI) - q2);
     float y = (phiRadToSteps * ((q1Home/180*M_PI) - q1)) + 200;
     float z = phiToSteps * phiE;
```

Převodový poměr je zde doplněn o přepočet z radiánů na kroky, neboť goniometrické funkce vracejí hodnoty v radiánech. Pro rozměry robotu jsou zavedeny nové proměnné, aby byla možná jejich úprava po kalibraci.

```
const float phiRadToSteps = 4352.164262; // ((47*6400)/(11*2*PI))
//parameters of robot [mm]
const float l_2 = 135;
const float l_7 = 160;
const float z1o2 = 148;
const float r3E = 130;
const float z3E = 8.6;
```

Program je doplněn o možnost spuštění funkce homing(), vhodnou pro vrácení robotu do domácí pozice a aktualizace výchozí polohy, kdyby došlo ke ztrátě kroku a tím k desynchronizaci robotu. Program tedy na vstupu očekává čtyři hodnoty, polohu efektoru (r_E, z_E, φ_E) a čtvrtou hodnotou je stav, zda se má provést homing() funkce.

```
Serial.println("Hello! Please enter position
(rE[mm],zE[mm],phiE[deg],GoHome[1/0],):");
```

Pokud je tedy splněna následující podmínka, hodnoty pro novou polohu jsou ignorovány, robot provede homing() a je natočen do výchozí pozice.

```
   else if(homeCondition==1 && digitalRead(hallSensorX)==HIGH &&
digitalRead(hallSensorY)==HIGH && digitalRead(zSwitch)==HIGH){
      Serial.println("\tHoming is running...!");
      homing();
      stepperXPosition = 3500;
      stepperYPosition = -200;
      stepperZPosition = 1200;
      Serial.print("positionX[STEPS]= ");
      Serial.print(stepperXPosition);
      Serial.print("\tpositionY[STEPS]= ");
      Serial.print(stepperYPosition);
      Serial.print("\tpositionZ[STEPS]= ");
      Serial.println(stepperZPosition);
    }
```

4.4 Kalibrace

Přesná činnost robotu není možná bez znalosti jeho skutečných rozměrů a kinematických parametrů (v našem případě např. délky těles, převodový poměř). Tyto parametry se snažíme co nejpřesněji určit kalibrací. Podle přístupu nalezení přesných kinematických parametrů můžeme kalibraci rozdělit [19]:

1) **Přímá kalibrační metoda** – spočívá v odměření přesných rozměrů na externím měřícím zařízení. Tato metoda má několik nevýhod. Je velice časově náročná, protože musíme každý člen odměřit zvlášť. Vzájemné působení mechanismu komplikuje měření každého členu samostatně (pro kalibraci by musel být rozložen). Dále kinematická chyba se může projevit v každé poloze mechanismu vlivem působení gravitačních sil.
2) **Kinematická kalibrační metoda** – je založena na vypočtení kinematických rozměrů ze sestaveného mechanismu a všechny hledané parametry jsou vypočteny najednou. Efektor se postupně umísťuje do předem stanovených bodů v prostoru. Princip spočívá v minimálním rozdílu mezi polohou předem stanovených bodů a polohou vypočítanou pomocí kinematických vztahů mechanismu. Počtem poloh je dán počet rovnic. Z hlediska přesnosti výpočtu parametrů musí být výrazně vyšší počet rovnic, než je počet neznámých parametrů. Tím vzniká přeurčená soustava rovnic, která nejde řešit analyticky, a pro její výpočet je použita Newtonova iterační metoda upravená pro přeurčenou soustavu rovnic. Tato metoda je použita pro kalibraci Dobotu.

4.4.1 Postup kalibrace[19]

Základem je kinematická transformace mezi úhlovými souřadnicemi natočení a polohou efektoru popsaná v odst. 3.2, kde se ve vztazích vyskytují hledané rozměry robotu $\boldsymbol{d} = [l_2, l_7, r_{3E}, z_{3E}, z_{1o2}]$.

$$[r_E; z_E] = f(q_1; q_2; \boldsymbol{d}) \tag{17}$$

Polohování efektoru na kalibrační body $\boldsymbol{r}$ při současném měření natočení úhlových souřadnic dostaneme přeurčenou soustavu rovnic.

$$f(\boldsymbol{d}) = \mathbf{r} \tag{18}$$

$$\begin{bmatrix} l_2\cos(q_{11}) - l_7\cos(q_{21}) + r_{3\mathrm{E}} \\ l_2\sin(q_{11}) - l_7\sin(q_{21}) + z_{3\mathrm{E}} + z_{1o2} \\ l_2\cos(q_{12}) - l_7\cos(q_{22}) + r_{3\mathrm{E}} \\ l_2\sin(q_{12}) - l_7\sin(q_{22}) + z_{3\mathrm{E}} + z_{1o2} \\ l_2\cos(q_{13}) - l_7\cos(q_{23}) + r_{3\mathrm{E}} \\ l_2\sin(q_{13}) - l_7\sin(q_{23}) + z_{3\mathrm{E}} + z_{1o2} \\ l_2\cos(q_{14}) - l_7\cos(q_{24}) + r_{3\mathrm{E}} \\ l_2\sin(q_{14}) - l_7\sin(q_{24}) + z_{3\mathrm{E}} + z_{1o2} \\ l_2\cos(q_{15}) - l_7\cos(q_{25}) + r_{3\mathrm{E}} \\ l_2\sin(q_{15}) - l_7\sin(q_{25}) + z_{3\mathrm{E}} + z_{1o2} \\ l_2\cos(q_{16}) - l_7\cos(q_{26}) + r_{3\mathrm{E}} \\ l_2\sin(q_{16}) - l_7\sin(q_{26}) + z_{3\mathrm{E}} + z_{1o2} \\ l_2\cos(q_{17}) - l_7\cos(q_{27}) + r_{3\mathrm{E}} \\ l_2\sin(q_{17}) - l_7\sin(q_{27}) + z_{3\mathrm{E}} + z_{1o2} \\ l_2\cos(q_{18}) - l_7\cos(q_{28}) + r_{3\mathrm{E}} \\ l_2\sin(q_{18}) - l_7\sin(q_{28}) + z_{3\mathrm{E}} + z_{1o2} \\ l_2\cos(q_{19}) - l_7\cos(q_{29}) + r_{3\mathrm{E}} \\ l_2\sin(q_{19}) - l_7\sin(q_{29}) + z_{3\mathrm{E}} + z_{1o2} \\ l_2\cos(q_{110}) - l_7\cos(q_{210}) + r_{3\mathrm{E}} \\ l_2\sin(q_{110}) - l_7\sin(q_{210}) + z_{3\mathrm{E}} + z_{1o2} \\ l_2\cos(q_{111}) - l_7\cos(q_{211}) + r_{3\mathrm{E}} \\ l_2\sin(q_{111}) - l_7\sin(q_{211}) + z_{3\mathrm{E}} + z_{1o2} \\ l_2\cos(q_{112}) - l_7\cos(q_{212}) + r_{3\mathrm{E}} \\ l_2\sin(q_{112}) - l_7\sin(q_{212}) + z_{3\mathrm{E}} + z_{1o2} \\ l_2\cos(q_{113}) - l_7\cos(q_{213}) + r_{3\mathrm{E}} \\ l_2\sin(q_{113}) - l_7\sin(q_{213}) + z_{3\mathrm{E}} + z_{1o2} \\ l_2\cos(q_{114}) - l_7\cos(q_{214}) + r_{3\mathrm{E}} \\ l_2\sin(q_{114}) - l_7\sin(q_{214}) + z_{3\mathrm{E}} + z_{1o2} \end{bmatrix} = \begin{bmatrix} r_{\mathrm{E}_1} \\ z_{\mathrm{E}_1} \\ r_{\mathrm{E}_2} \\ z_{\mathrm{E}_2} \\ r_{\mathrm{E}_3} \\ z_{\mathrm{E}_3} \\ r_{\mathrm{E}_4} \\ z_{\mathrm{E}_4} \\ r_{\mathrm{E}_5} \\ z_{\mathrm{E}_5} \\ r_{\mathrm{E}_6} \\ z_{\mathrm{E}_6} \\ r_{\mathrm{E}_7} \\ z_{\mathrm{E}_7} \\ r_{\mathrm{E}_8} \\ z_{\mathrm{E}_8} \\ r_{\mathrm{E}_9} \\ z_{\mathrm{E}_9} \\ r_{\mathrm{E}_{10}} \\ z_{\mathrm{E}_{10}} \\ r_{\mathrm{E}_{11}} \\ z_{\mathrm{E}_{11}} \\ r_{\mathrm{E}_{12}} \\ z_{\mathrm{E}_{12}} \\ r_{\mathrm{E}_{13}} \\ z_{\mathrm{E}_{13}} \\ r_{\mathrm{E}_{14}} \\ z_{\mathrm{E}_{14}} \end{bmatrix} \tag{19}$$

Řešení nalezneme Newtonovou iterační metodou odvozenou Taylorovým rozvojem, kde $\boldsymbol{J}_{\mathbf{d}}$ je Jacobiho matice parciálních derivací podle kalibrovaných rozměrů $\boldsymbol{d}$, $\delta\boldsymbol{r}$ je vektor odchylek od kalibračních bodů.

$$\boldsymbol{J}_{\mathbf{d}} \cdot \delta\boldsymbol{d} = \mathbf{r} - \mathrm{f}(\mathbf{d}) = \delta\boldsymbol{r} \tag{20}$$

$$\boldsymbol{J}_{\mathbf{d}} = \begin{bmatrix} \cos(q_{11}) & -\cos(q_{21}) & 1 & 0 & 0 \\ \sin(q_{11}) & -\sin(q_{21}) & 0 & 1 & 1 \\ \cos(q_{12}) & -\cos(q_{22}) & 1 & 0 & 0 \\ \sin(q_{12}) & -\sin(q_{22}) & 0 & 1 & 1 \\ \cos(q_{13}) & -\cos(q_{23}) & 1 & 0 & 0 \\ \sin(q_{13}) & -\sin(q_{23}) & 0 & 1 & 1 \\ \cos(q_{14}) & -\cos(q_{24}) & 1 & 0 & 0 \\ \sin(q_{14}) & -\sin(q_{24}) & 0 & 1 & 1 \\ \cos(q_{15}) & -\cos(q_{25}) & 1 & 0 & 0 \\ \sin(q_{15}) & -\sin(q_{25}) & 0 & 1 & 1 \\ \cos(q_{16}) & -\cos(q_{26}) & 1 & 0 & 0 \\ \sin(q_{16}) & -\sin(q_{26}) & 0 & 1 & 1 \\ \cos(q_{17}) & -\cos(q_{27}) & 1 & 0 & 0 \\ \sin(q_{17}) & -\sin(q_{27}) & 0 & 1 & 1 \\ \cos(q_{18}) & -\cos(q_{28}) & 1 & 0 & 0 \\ \sin(q_{18}) & -\sin(q_{28}) & 0 & 1 & 1 \\ \cos(q_{19}) & -\cos(q_{29}) & 1 & 0 & 0 \\ \sin(q_{19}) & -\sin(q_{29}) & 0 & 1 & 1 \\ \cos(q_{110}) & -\cos(q_{210}) & 1 & 0 & 0 \\ \sin(q_{110}) & -\sin(q_{210}) & 0 & 1 & 1 \\ \cos(q_{111}) & -\cos(q_{211}) & 1 & 0 & 0 \\ \sin(q_{111}) & -\sin(q_{211}) & 0 & 1 & 1 \\ \cos(q_{112}) & -\cos(q_{212}) & 1 & 0 & 0 \\ \sin(q_{112}) & -\sin(q_{212}) & 0 & 1 & 1 \\ \cos(q_{113}) & -\cos(q_{213}) & 1 & 0 & 0 \\ \sin(q_{113}) & -\sin(q_{213}) & 0 & 1 & 1 \\ \cos(q_{114}) & -\cos(q_{214}) & 1 & 0 & 0 \\ \sin(q_{114}) & -\sin(q_{214}) & 0 & 1 & 1 \end{bmatrix} \tag{21}$$

V i-tém kroku Newtonovy metody jsou spočteny opravy rozměrů.

$$\delta \boldsymbol{d}_{\mathrm{i}} = (\boldsymbol{J}_{\mathbf{d}_i}^{\boldsymbol{T}} \cdot \boldsymbol{J}_{\mathbf{d}_i})^{-1} \cdot \boldsymbol{J}_{\mathbf{d}_i}^{\boldsymbol{T}} \cdot \delta \boldsymbol{r}_i \tag{22}$$

Nová hodnota rozměrů pro další iteraci se spočte.

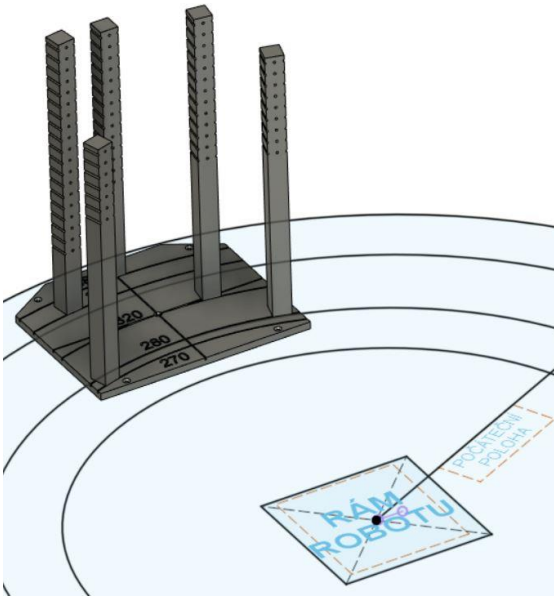
$$\boldsymbol{d}_{\mathrm{i+1}} = \boldsymbol{d}_{\mathrm{i}} + \delta \boldsymbol{d}_{\mathrm{i}} \tag{23}$$

Iterace pokračuje, dokud se odchylka zmenšuje.

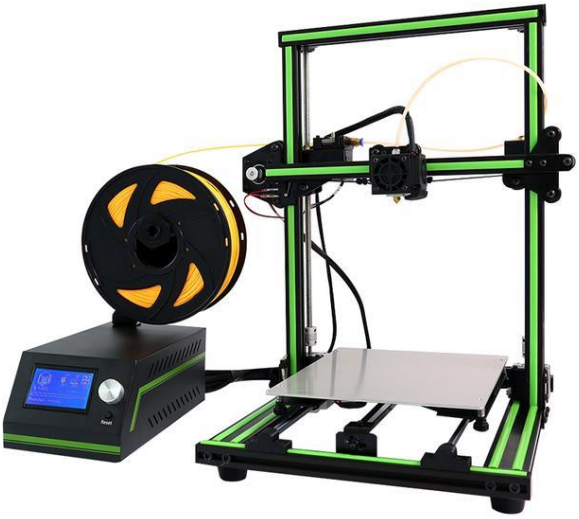
4.4.2 Kalibrační artefakt

Pro účel kalibrace byl vytvořen kalibrační artefakt v programu Autodesk Fusion 360. Artefakt je tvořen pěti sloupky rozmístěnými po základní ploše artefaktu tak, aby byl pokryt celý pracovní prostor robotu. Sloupky jsou od osy otáčení robotu ve vzdálenosti 270, 280, 320, 360 a 380 mm, a jsou od sebe pootočeny o 8°. Skrz sloupky jsou po 10 mm vytvořeny kruhové otvory s průměrem 2 mm, které slouží jako kalibrační body. Na základní ploše jsou vytvořeny otvory a drážky sloužící pro přesné umístění artefaktu v pracovním prostoru.

Artefakt byl vytisknut na 3D tiskárně Anet E10 s odhadnutou přesností 0,3 mm. Pro tisk byl použit materiál PLA. Tisk probíhal při teplotě trysky 210°C a teplotě podložky 60°C. Výška vrstvy byla 0,2 mm. Tisk trval přibližně 20 hodin a bylo spotřebováno 212g materiálu.



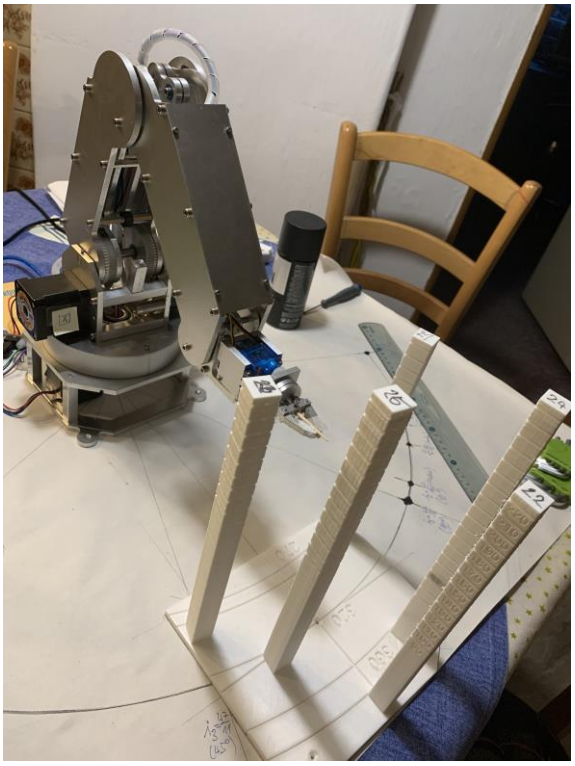
Obrázek 38 Model artefaktu ve Fusion360 s vyznačeným pracovním prostorem



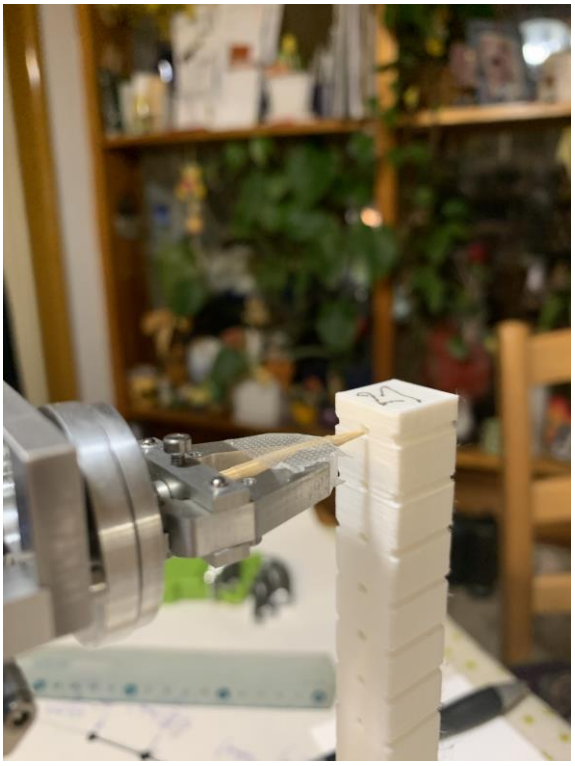
Obrázek 39 Tiskárna Anet E10[20]

4.4.3 Kalibrace Dobotu

Kalibrace Dobotu byla provedena pro nalezení jeho přesných rozměrů **d**. Jako hledaný parametr by mohl být i převodový poměr, ale ten byl určen a ověřen v odst. 4.3.2. Robot a kalibrační artefakt byly umístěny relativně přesně do pracovního prostoru (obr. 40). Efektor byl polohován inverzním řízením na vybraných 14 bodů na kalibračním artefaktu rozmístěných rovnoměrně po pracovním prostoru (obr. 41) a současně byly zaznamenávány úhlové souřadnice natočení. Pro zpracování hodnot byl napsán skript v programu MATLAB® podle postupu popsaného výše. Pro každý bod jsou dány dvě rovnice, do směru *r* a *z*, tím dostáváme 28 nelineárních rovnic. Numerická konvergence byla rychlá, již po první iteraci nedocházelo ke snížení odchylky. Získané zpřesněné rozměry (tab. 7) byly nahrány do programu inverzního řízení a polohování bylo provedeno znovu. Tabulka 8 ukazuje porovnání z obou měření.



Obrázek 40 Kalibrace Dobotu



Obrázek 41 Polohování efektoru

Rozměry robotu		
	z modelu	**po kalibraci**
l_2 [mm]	135	135,48527
l_7 [mm]	160	159,355547
r_{3E} [mm]	130	130,900837
z_{3E} [mm]	8,6	6,887047
z_{1o2} [mm]	148	146,287047

Tabulka 7 Porovnání kalibrovaných rozměrů

Kalibrace Dobotu									
Souřadnice r[mm]									
před kalibrací					po kalibraci				
kalibrační body		přímá kinematika	rozdíl	absolutní hodnota	kalibrační body		přímá kinematika	rozdíl	absolutní hodnota
1	270,000	271,036	-1,036	1,036	1	270,000	269,514	0,486	0,486
2	270,000	269,977	0,023	0,023	2	270,000	270,007	-0,007	0,007
3	280,000	278,977	1,023	1,023	3	280,000	278,992	1,008	1,008
4	280,000	279,005	0,995	0,995	4	280,000	278,977	1,023	1,023
5	280,000	279,973	0,027	0,027	5	280,000	279,985	0,015	0,015
6	320,000	317,982	2,018	2,018	6	320,000	319,015	0,985	0,985
7	320,000	317,995	2,005	2,005	7	320,000	318,994	1,006	1,006
8	320,000	318,993	1,007	1,007	8	320,000	319,498	0,502	0,502
9	320,000	319,980	0,020	0,020	9	320,000	319,495	0,505	0,505
10	320,000	319,986	0,014	0,014	10	320,000	319,987	0,013	0,013
11	380,000	379,011	0,989	0,989	11	380,000	378,998	1,002	1,002
12	380,000	379,500	0,500	0,500	12	380,000	379,008	0,992	0,992
13	380,000	379,485	0,515	0,515	13	380,000	379,999	0,001	0,001
14	380,000	380,007	-0,007	0,007	14	380,000	379,982	0,018	0,018
průměrná chyba				0,727	průměrná chyba				0,540
Souřadnice z[mm]									
před kalibrací					po kalibraci				
kalibrační body		přímá kinematika	rozdíl	absolutní hodnota	kalibrační body		přímá kinematika	rozdíl	absolutní hodnota
1	170,000	172,352	-2,352	2,352	1	170,000	170,022	-0,022	0,022
2	200,000	203,006	-3,006	3,006	2	200,000	200,029	-0,029	0,029
3	180,000	183,013	-3,013	3,013	3	180,000	180,530	-0,530	0,530
4	200,000	203,016	-3,016	3,016	4	200,000	200,508	-0,508	0,508
5	230,000	233,008	-3,008	3,008	5	230,000	230,510	-0,510	0,510
6	130,000	133,016	-3,016	3,016	6	130,000	130,514	-0,514	0,514
7	160,000	162,513	-2,513	2,513	7	160,000	160,504	-0,504	0,504
8	190,000	192,517	-2,517	2,517	8	190,000	190,029	-0,029	0,029
9	220,000	222,518	-2,518	2,518	9	220,000	220,037	-0,037	0,037
10	240,000	243,007	-3,007	3,007	10	240,000	240,034	-0,034	0,034
11	100,000	103,015	-3,015	3,015	11	100,000	100,026	-0,026	0,026
12	130,000	132,538	-2,538	2,538	12	130,000	130,030	-0,030	0,030
13	170,000	172,026	-2,026	2,026	13	170,000	169,516	0,484	0,484
14	210,000	212,031	-2,031	2,031	14	210,000	209,526	0,474	0,474
průměrná chyba				2,684	průměrná chyba				0,267

Tabulka 8 Kalibrace Dobotu

Z naměřených hodnot je zřejmé, že přesnost robotu se zlepšila. Výrazné zlepšení můžeme pozorovat na souřadnici *z,* kde průměrná chyba polohování před kalibrací byla 2,684 mm a po kalibraci 0,267 mm.

Provedená metodika kalibrace je běžně užívána v praxi. Důležité je podotknout, že přesnost výsledků kalibrace je však ovlivněna domácími podmínkami a použitými dostupnými přístroji (přesnost 3D tiskárny, vizuální polohování efektoru, přesnost umístění robotu a artefaktu v pracovním prostoru). I přes vzniklé nepřesnosti je vidět, jak kalibrace je důležitou součástí při řízení nejen robotů, ale i strojů kde je kladen důraz na velmi přesné polohování například u obráběcích strojů.

Za normálních podmínek by byl pro zpřesnění kalibrace použit přesnější kalibrační artefakt nebo přístroj pro 3D odměřování polohy LaserTraker, který určuje polohu reflektoru přichycenému k efektoru stroje s velikou přesností několika mikrometrů. Určení polohy je založeno na interferenci odraženého paprsku od reflektoru a měření výškového úhlu a natočení zdroje paprsku.

Obrázek 42 Kalibrace obráběcího stroje pomocí LASER TRACKERu[21]

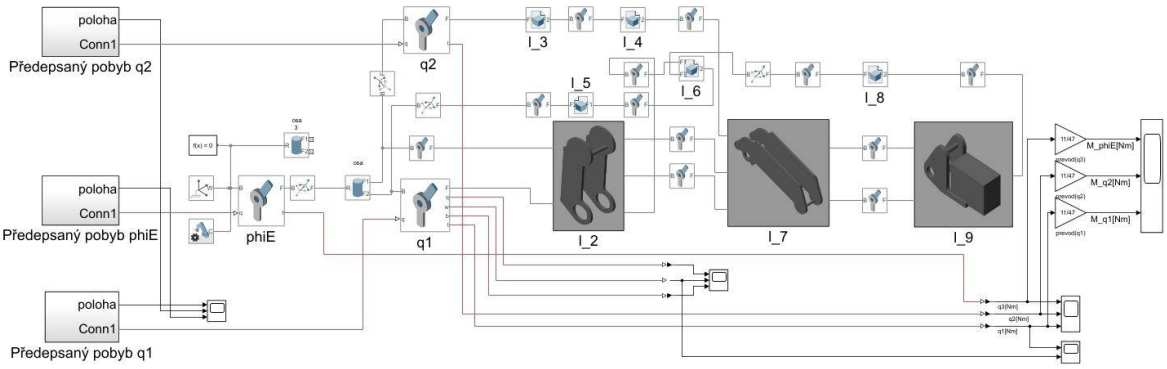
5 MBS model v prostředí MATLAB – Simulink – SimScape Multibody

SimScape Multibody™ (dále SM) je nadstavbou programu MATLAB® resp. Simulink® pro modelování a simulaci 3D mechanických soustav složených z mnoha těles (multibody simulation = MBS). Tato nadstavba byla využita pro sestavení MBS modelu Dobotu a ověření dimenzování motorů.

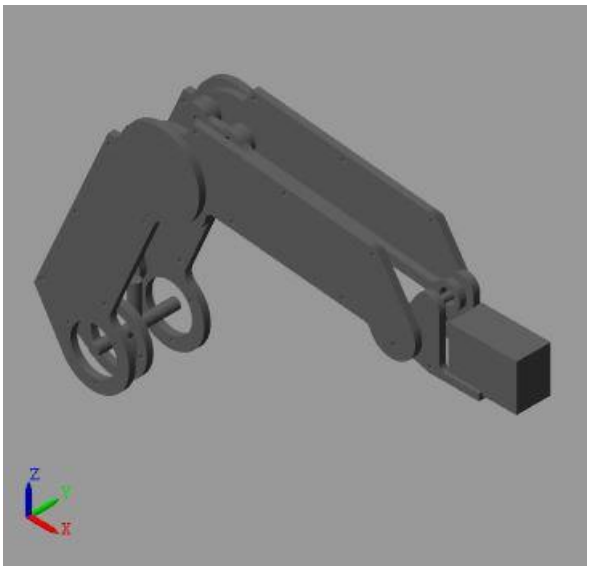
5.1 Tvorba modelu

Model v SM se sestavuje pomocí bloků reprezentující tělesa, kinematické vazby, silové prvky a senzory. Ze sestaveného modelu SM formuluje a řeší pohybové rovnice kompletního mechanického systému. Automaticky generovaná 3D animace umožnuje vizualizaci celého systému.[22]

Jednotlivá tělesa Dobotu byly do modelu naimportovány z počítačového modelu ve formátu STEP, byla jim nadefinována hustota $2\,800\,\frac{kg}{m^3}$, neboť je robot vyroben z duralu, a z geometrie daného tělesa si SM určil jeho momenty setrvačnosti. Spojení těles je modelováno rotačními vazbami.



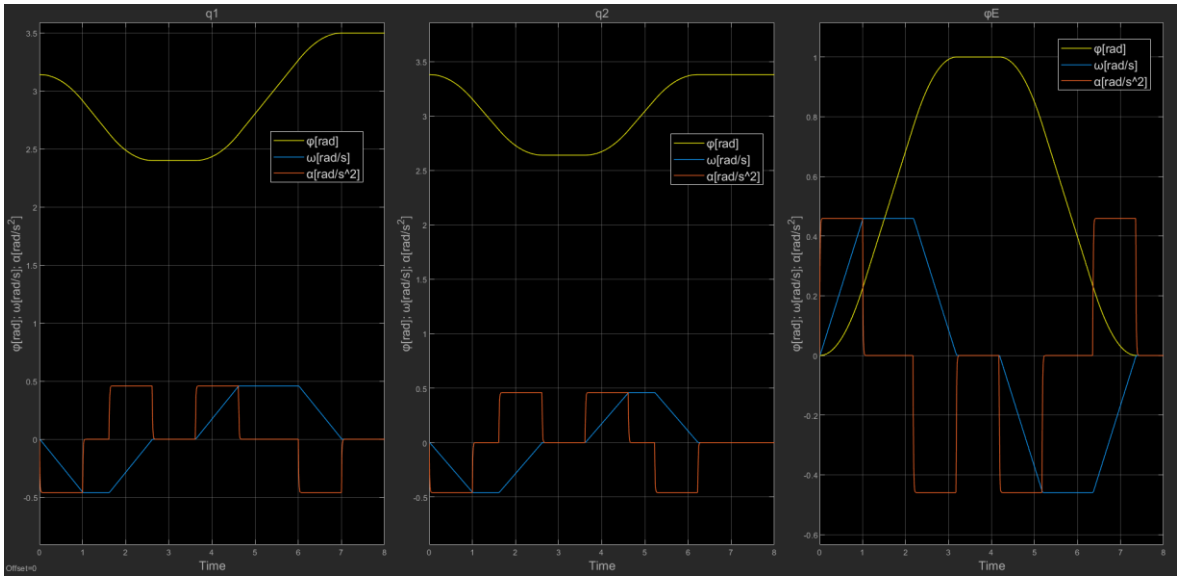
Obrázek 43 Simulink® blokové schéma



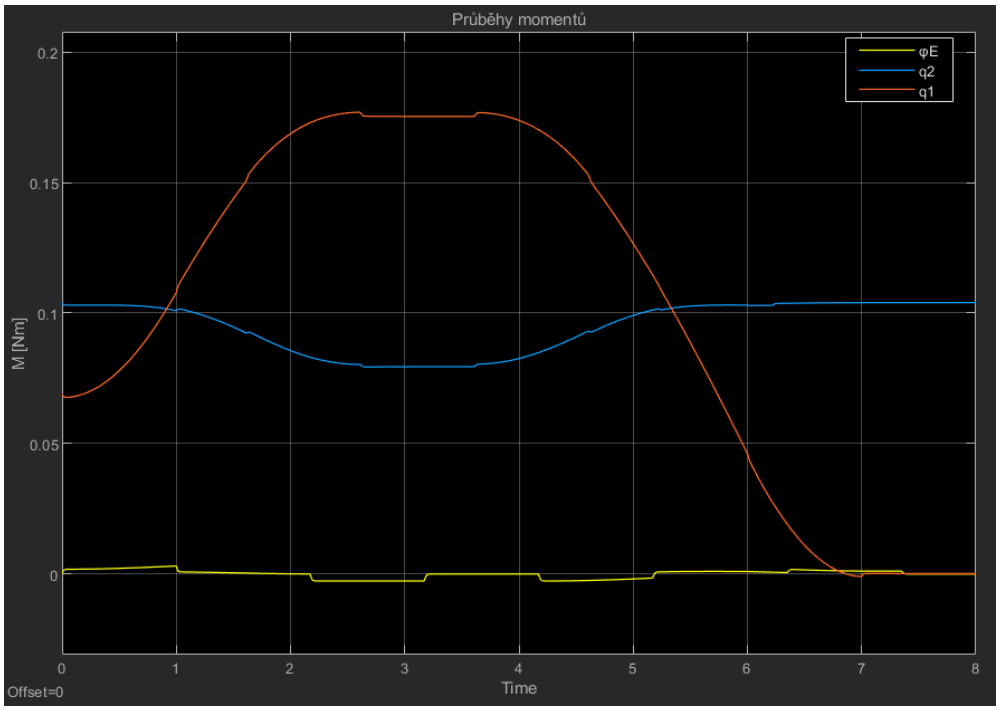
Obrázek 44 SimScape Multibody™ 3D vizualizace

5.2 Ověření dimenzování motorů

Rotačním vazbám byl předepsán pohyb odpovídající profilu na reálném robotu (pro rychlosti a zrychlení nastavené v programu). Pro zajištění předepsaného pohybu SM určí požadované momenty, tedy vyřeší inverzní dynamiku. Ze získaných průběhů momentů přepočtem přes převodový poměr získáme požadované momenty od krokových motorů.



Obrázek 45 Předepsané pohyby motorů



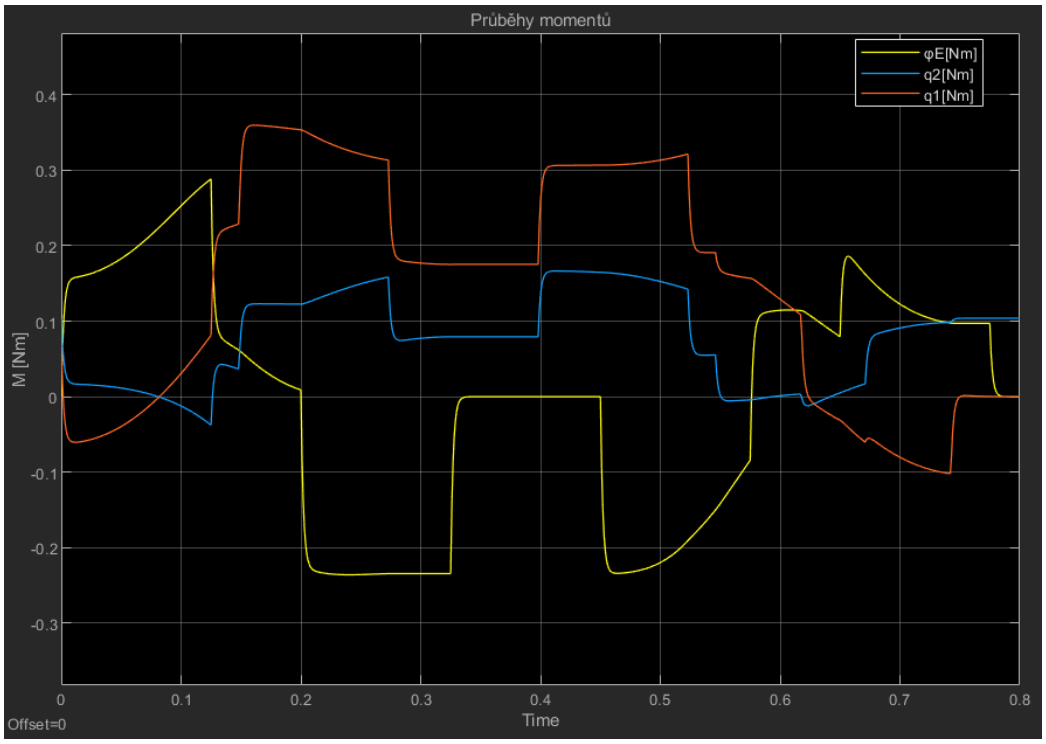
Obrázek 46 Průběhy momentů

Ze získaného průběhu momentů (obr. 46) vidíme, že maximální požadovaný moment je od motoru q_1 0,18 Nm. Z momentové charakteristiky motoru[13] je maximální dovolený moment při předepsané rychlosti 0,44 Nm, tudíž nedochází k přetížení a motory jsou správně dimenzovány.

Jelikož robot nevyužívá celý dostupný moment motoru, byla v simulaci pro stejný pohyb zvyšována rychlost a zrychlení tak, aby nebyly motory přetíženy. Tím byla nalezena maximální rychlost a zrychlení robotu při maximálním špičkovém momentu motoru q_1 0,36 Nm. Tato hodnota je s ohledem na bezpečnost a ztrátu kroku menší než maximální moment z momentové charakteristiky[13] při této rychlosti, který je 0,42 Nm.

Úhlová souřadnice	**Nastaveno v programu**		**Maximální**	
	ω [rad/s]	***α [rad/s²]***	**ω [rad/s]**	***α [rad/s²]***
q_1	0,4595	0,4595	5	40
q_2	0,4595	0,4595	5	40
φ_E	0,4595	0,4595	5	40

Tabulka 9 Úhlové rychlosti a zrychlení



Obrázek 47 Průběhy momentů při maximální rychlosti a zrychlení

6 Závěr

První část práce se zabývá průmyslovými roboty, které jsou dnes nedílnou součástí automatizace. Byla provedena rešerše používaných kinematických struktur robotů.

V druhé části je vyřešena kinematika existující robotické ruky s hybridní kinematickou strukturou typu Dobot, pro kterou je vytvořen program pro přímé a inverzní řízení na platformě Arduino. Program inverzního řízení umožňuje polohování robotu v cylindrickém souřadnicovém systému pomocí sériového rozhraní.

Další část práce se zabývá problematikou kalibrace strojů. Byla použita kinematická kalibrační metoda pro kalibraci Dobotu, po které došlo k výraznému zlepšení přesnosti polohování.

Poslední částí práce bylo vytvoření MBS modelu robotu, pomocí kterého bylo ověřeno dimenzování krokových motorů pro dané rychlosti a zrychlení motorů nastavené v řídícím programu. Dále byla zjištěna maximální přípustná rychlost a zrychlení, při nichž ještě nedojde k přetížení motorů.

Všechny cíle bakalářské práce byly splněny.

Literatura

[1] SKAŘUPA, Jiří. *Průmyslové roboty a manipulátory* [online]. Ostrava: Ediční středisko VŠB – TUO, 2007 [cit. 2021-04-13]. ISBN ISBN 978-80-248-1522-0. Dostupné z: http://www.elearn.vsb.cz/archivcd/FS/PRM/Text/Skripta_PRaM.pdf

[2] PANDILOV, Zoran a Vladimir DUKOVSKI. COMPARISON OF THE CHARACTERISTICS BETWEEN SERIAL AND PARALLEL ROBOTS. *ACTA TEHNICA CORVINIENSIS – Bulletin of Engineering* [online]. 2014 [cit. 2021-04-13]. ISSN 2067 – 3809. Dostupné z: http://acta.fih.upt.ro/pdf/2014-1/ACTA-2014-1-19.pdf

[3] MRÁZ, Petr. *Průmyslové roboty v praxi* [online]. ČVUT v Praze, Fakulta strojní, Ústav výrobních strojů a mechanismů [cit. 2021-04-17]. Dostupné z: https://docplayer.cz/6707543-Prumyslove-roboty-v-praxi.html

[4] *KUKA* [online]. [cit. 2021-04-18]. Dostupné z: https://www.kuka.com/cs-cz/

[5] *ABB* [online]. [cit. 2021-04-18]. Dostupné z: https://new.abb.com/cz

[6] *PKM TRICEPT* [online]. [cit. 2021-04-18]. Dostupné z: http://www.pkmtricept.com/index.php?id=en

[7] *DOBOT Magician* [online]. [cit. 2021-04-18]. Dostupné z: https://www.dobot.cc/dobot-magician/product-overview.html

[8] VALÁŠEK, Michael, Zbyněk ŠIKA a Václav BAUMA. *Mechanika B*. 2006. V Praze: Vydavatelství ČVUT, 2004. ISBN 80-010-2919-0.

[9] *Arduino* [online]. [cit. 2021-04-12]. Dostupné z: https://www.arduino.cc/

[10] *Arduino CNC Shield* [online]. [cit. 2021-5-1]. Dostupné z: https://blog.protoneer.co.nz/arduino-cnc-shield/

[11] ŘEZÁČ, Kamil. *Krokové motory* [online]. [cit. 2021-4-27]. Dostupné z: https://robotika.cz/articles/steppers/cs

[12] *ELUC: Krokový motor* [online]. [cit. 2021-4-27]. Dostupné z: https://eluc.kr-olomoucky.cz/verejne/lekce/809

[13] *Reichelt Elektronik: ACT 17HS5415P1X6 Stepper motor* [online]. [cit. 2021-5-7]. Dostupné z: https://www.reichelt.com/de/en/stepper-motor-4-pole-1-8-4-2-v-dc-act-17hs5415p1x6-p243412.html

[14] *DRV8825 Stepper Motor Driver Carrier, High Current* [online]. [cit. 2021-5-1]. Dostupné z: https://www.pololu.com/product/2133

[15] *Princip Hallova senzoru* [online]. [cit. 2021-5-2]. Dostupné z: https://e-konstrukter.cz/novinka/princip-hallova-senzoru

[16] *Physical Computing: Into to Arduino* [online]. [cit. 2021-5-6]. Dostupné z: https://makeabilitylab.github.io/physcomp/arduino/buttons.html#materials

[17] MCCAULEY, Mike. *AccelStepper* [online]. [cit. 2021-5-7]. Dostupné z: http://www.airspayce.com/mikem/arduino/AccelStepper/index.html

[18] AUSTIN, David. *Generate stepper-motor speed profiles in real time* [online]. [cit. 2021-5-7]. Dostupné z: http://web.archive.org/web/20140705143928/http://fab.cba.mit.edu/classes/MIT/961.09/projects/i0/Stepper_Motor_Speed_Profile.pdf

[19] HAMRLE, Vojtěch. *Kalibrovatelnost a její použití pro návrh paralelních kinematických struktur*. V Praze, 2009. Disertační práce. České vysoké učení technické, Fakulta strojní. Vedoucí práce Doc. Ing. Zbyněk Šika, Ph.D.

[20] *Anet 3D Printer* [online]. [cit. 2021-5-9]. Dostupné z: https://shop.anet3d.com/products/anet-e10-3d-printer

[21] *Haxagon: ETALON LASERTRACER-NG* [online]. [cit. 2021-5-15]. Dostupné z: https://www.hexagonmi.com/cs-CZ/products/machine-calibration-and-optimisation/etalon-lasertracer-ng

[22] *Firemní literatura Mathworks Matlab, Simulink, SimScape Multibody* [online]. [cit. 2021-5-20]. Dostupné z: https://www.mathworks.com/help/

Seznam obrázků

Seznam tabulek

Seznam elektronických příloh

1) Program přímého řízení
    - dobotSerialControl_X_Y_Z_forward.ino
2) Program inverzního řízení
    - dobotSerialControl_X_Y_Z_inverse_fcnHoming.ino
3) MBS model
    - MBS_model_DOBOT.slx
4) Skript pro zpracování hodnot kalibrace
    - kalibrace.m
5) Skripty pro nalezení pracovního prostoru
    - workspaceDobot2D.m
    - workspaceDobot3D.m
6) Model kalibračního artefaktu
    - kalibracniArtefakt.step

Seznam příloh práce

1) Program inverzního řízení

Program inverzního řízení

```
#include <AccelStepper.h>

#define led 13

#define motorInterfaceType 1
//Motor X
#define dirX 5
#define stepX 2
//Motor Y
#define dirY 6
#define stepY 3
//Motor Z
#define dirZ 7
#define stepZ 4

AccelStepper stepperX = AccelStepper(motorInterfaceType,stepX,dirX);
AccelStepper stepperY = AccelStepper(motorInterfaceType,stepY,dirY);
AccelStepper stepperZ = AccelStepper(motorInterfaceType,stepZ,dirZ);

//Hall sensor
#define hallSensorX 9
#define hallSensorY 10
//switch Z sensor
#define zSwitch 12
//serial communication
String rxString = "";
String data[4];

//parameters of robot [mm]
const float l_2 = 135.485270;
const float l_7 = 159.355547;
const float z1o2 = 146.287047;
const float r3E = 130.900837;
const float z3E = 6.887047;

//variables for position effector in STEPS
int stepperXPosition = 3500;
int stepperYPosition = -200;
int stepperZPosition = 1200;

const float phiRadToSteps = 4352.164262; // ((47*6400)/(11*2*PI))
const float phiToSteps = 75.959595; // ((47*6400)/(11*360))

//Limits of robot in STEPS
//q2
const int xMAX = 5500; //122°
const int xMIN = 2200; //166°
//q1
const int yMAX = 5600; //19°
const int yMIN = 0;
//phiE
const int zMAX = 13672; //180°
const int zMIN = 0;

void setup() {
  Serial.begin(9600);
  Serial.println("Hello! Please enter position
(rE[mm],zE[mm],phiE[deg],GoHome[1/0],):");

  pinMode(led,OUTPUT);
```

```
  digitalWrite(led,LOW);

  pinMode(hallSensorX,INPUT);
  pinMode(hallSensorY,INPUT);
  pinMode(zSwitch,INPUT_PULLUP);

  stepperX.setMaxSpeed(2000);
  stepperX.setAcceleration(2000);
  stepperX.setMinPulseWidth(2);
  stepperY.setMaxSpeed(2000);
  stepperY.setAcceleration(2000);
  stepperY.setMinPulseWidth(2);
  stepperZ.setMaxSpeed(2000);
  stepperZ.setAcceleration(2000);
  stepperZ.setMinPulseWidth(2);

  homing();
}

void loop() {
  receiveData();
  delay(1000);
  stepperX.moveTo(stepperXPosition);
  stepperY.moveTo(stepperYPosition);
  stepperZ.moveTo(stepperZPosition);
  while(stepperX.currentPosition() != stepperXPosition ||
stepperY.currentPosition() != stepperYPosition ||
stepperZ.currentPosition() != stepperZPosition){
    stepperX.run();
    stepperY.run();
    stepperZ.run();
  }
}

void receiveData(){
  if(Serial.available()>0){
    delay(2);
    rxString = Serial.readString();
    int stringStart = 0;
    int arrayIndex = 0;
    for (int i = 0; i  < rxString.length(); i++) {
            if (rxString.charAt(i) == ',') {
                data[arrayIndex] = ""; // deletes previous
                data[arrayIndex] = rxString.substring(stringStart, i);
                stringStart = (i + 1);
                arrayIndex++;
            }
    }
    Serial.print("rE[mm]= ");
    Serial.print(data[0]);
    Serial.print("\tzE[mm]= ");
    Serial.print(data[1]);
    Serial.print("\tphiE[°]= ");
    Serial.print(data[2]);
    Serial.print("\tGoHome[1/0]= ");
    Serial.println(data[3]);
    //GoHome condition
    int homeCondition = data[3].toInt();
    //inverse kinematics
    float rE = data[0].toFloat();
    float zE = data[1].toFloat();
```

```
    float phiE = data[2].toFloat();
    float r2 = rE - r3E;
    float z2 = zE - z3E - z1o2;
    float a = sqrt(r2*r2 + z2*z2);
    double beta = atan(z2/r2);
    double gama = acos((-l_7*l_7+l_2*l_2+r2*r2+z2*z2)/(2*l_2*a));
    double delta = acos((-r2*r2-z2*z2+l_2*l_2+l_7*l_7)/(2*l_2*l_7));
    double q1 = beta + gama;
    double q2 = q1 + delta;
    Serial.print("q2[rad]= ");
    Serial.print(q2);
    Serial.print("\tq1[rad]= ");
    Serial.print(q1);
    Serial.print("\tphiE[deg]= ");
    Serial.println(phiE);
    //Home position in degrees
    float q1Home = 90;
    float q2Home = 195;
    Serial.print("\tq2Home[rad]= ");
    Serial.print(q2Home/180*M_PI);
    Serial.print("\tq1Home[rad]= ");
    Serial.println(q1Home/180*M_PI);
    //position in STEPS for check
    float x = phiRadToSteps * ((q2Home/180*M_PI) - q2);
    float y = (phiRadToSteps * ((q1Home/180*M_PI) - q1)) + 200;
    float z = phiToSteps * phiE;
    Serial.print("x[STEPS]= ");
    Serial.print(x);
    Serial.print("\ty[STEPS]= ");
    Serial.print(y);
    Serial.print("\tz[STEPS]= ");
    Serial.println(z);
    if(homeCondition==0 && x>=xMIN && x<=xMAX && y>=yMIN && y<=yMAX &&
z>=zMIN && z<=zMAX){
      stepperXPosition = x;
      stepperYPosition = -y;
      stepperZPosition = z;
      Serial.print("positionX[STEPS]= ");
      Serial.print(stepperXPosition);
      Serial.print("\tpositionY[STEPS]= ");
      Serial.print(stepperYPosition);
      Serial.print("\tpositionZ[STEPS]= ");
      Serial.println(stepperZPosition);
    }
    else if(homeCondition==1 && digitalRead(hallSensorX)==HIGH &&
digitalRead(hallSensorY)==HIGH && digitalRead(zSwitch)==HIGH){
      Serial.println("\tHoming is running...!");
      homing();
      stepperXPosition = 3500;
      stepperYPosition = -200;
      stepperZPosition = 1200;
      Serial.print("positionX[STEPS]= ");
      Serial.print(stepperXPosition);
      Serial.print("\tpositionY[STEPS]= ");
      Serial.print(stepperYPosition);
      Serial.print("\tpositionZ[STEPS]= ");
      Serial.println(stepperZPosition);
    }
    else{
      Serial.println("!!Coordinates are out of the workspace! They
were not realized!! CURRENT POSITION:");
```

```
      Serial.print("positionX[STEPS]= ");
       Serial.print(stepperXPosition);
       Serial.print("\tpositionY[STEPS]= ");
       Serial.print(stepperYPosition);
       Serial.print("\tpositionZ[STEPS]= ");
       Serial.println(stepperZPosition);
     }
     Serial.println("\t----------------------");
   }
}

void homing(){
  //Homing stepperY
  while(digitalRead(hallSensorY) == HIGH){
    stepperY.setSpeed(1800);
    stepperY.runSpeed();
  }
  stepperY.setCurrentPosition(0);
  delay(1000);
  stepperY.moveTo(-200);
  while(stepperY.currentPosition() != -200){
    stepperY.run();
  }

  delay(1000);

  //Homing stepperX
  while(digitalRead(hallSensorX) == HIGH){
    stepperX.setSpeed(-1800);
    stepperX.runSpeed();
  }
  stepperX.setCurrentPosition(0);
  delay(1000);
  stepperX.moveTo(3500);
  while(stepperX.currentPosition() != 3500){
    stepperX.run();
  }

  delay(1000);

  //Homing stepperZ
  while(digitalRead(zSwitch) == HIGH){
    stepperZ.setSpeed(-2000);
    stepperZ.runSpeed();
  }
  stepperZ.setCurrentPosition(0);
  delay(1000);
  stepperZ.moveTo(1200);
  while(stepperZ.currentPosition() != 1200){
    stepperZ.run();
  }
}
```