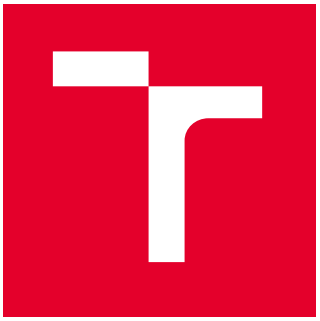



# VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

## FAKULTA STROJNÍHO INŽENÝRSTVÍ

FACULTY OF MECHANICAL ENGINEERING

## ÚSTAV MECHANIKY TĚLES, MECHATRONIKY A BIOMECHANIKY

INSTITUTE OF SOLID MECHANICS, MECHATRONICS AND BIOMECHANICS


# VIRTUÁLNÍ ZPROVOZNĚNÍ VÍCEOSÉHO MANIPULÁTORU

VIRTUAL COMMISSIONING OF MULTI-AXIS MANIPULATOR

**DIPLOMOVÁ PRÁCE**

MASTER'S THESIS


**AUTOR PRÁCE** — **Bc. Vojtěch Slavík**

AUTHOR

**VEDOUCÍ PRÁCE** — **Ing. Vojtěch Štěpánek, Ph.D.**

SUPERVISOR



**BRNO 2023**

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA STROJNÍHO INŽENÝRSTVÍ

# Zadání diplomové práce

| | |
|---|---|
| Ústav: | Ústav mechaniky těles, mechatroniky a biomechaniky |
| Student: | **Bc. Vojtěch Slavík** |
| Studijní program: | Mechatronika |
| Studijní obor: | bez specializace |
| Vedoucí práce: | **Ing. Vojtěch Štěpánek, Ph.D.** |
| Akademický rok: | 2022/23 |

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

## Virtuální zprovoznění víceosého manipulátoru

**Stručná charakteristika problematiky úkolu:**

Virtuální zprovoznění stroje je velmi účinný nástroj jak ušetřit čas i peníze a výrazně snížit riziko vzniku chyb během konstrukčního procesu. Jedním z nástrojů je nadstavba 3D CAD modeláře NX Mechatronics concept designer. Cílem diplomové práce je virtuální zprovoznění mechanismu pro obsluhu kulových segmentů (pixelů) a vytváření binárních obrazců. Součástí řešení je kromě zprovoznění modelu v prostředí MCD a sestavení řídicího PLC programu i výkres sestavy celého mechanismu a zhodnocení rizik spojených s jeho provozováním.

**Cíle diplomové práce:**

Rešerše z oblasti PLC řízení víceosých manipulátorů.
Tři koncepty řešení zadané úlohy a jejich multikriteriální zhodnocení.
Zpracování detailního 3D konstrukčního modelu navrženého řešení.
Návrh PLC řízení pro obsluhu požadované funkce.
Virtuální zprovoznění sestaveného modelu v prostředí NX MCD a vytvoření HMI.
Závěr, zhodnocení dosažených výsledků a doporučení pro praxi.

**Seznam doporučené literatury:**

MAREK, Jiří. Konstrukce CNC obráběcích strojů IV.0. Praha: MM publishing, 2018. MM speciál. ISBN ISBN978-80-906310-8-3.

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

SHIGLEY, Joseph Edward, Charles R MISCHKE a Richard G BUDYNAS, VLK, Miloš, ed. Konstruování strojních součástí. V Brně: VUTIUM, 2010, xxv, 1159 s. Překlady vysokoškolských učebnic. ISBN 978-80-214-2629-0.

Uživatelská příručka Mechatronics Concept Designer

Termín odevzdání diplomové práce je stanoven časovým plánem akademického roku 2022/23

V Brně, dne

L. S.

| | |
|---|---|
| prof. Ing. Jindřich Petruška, CSc.<br>ředitel ústavu | doc. Ing. Jiří Hlinka, Ph.D.<br>děkan fakulty |

## Abstrakt

Tato diplomová práce se zabývá návrhem a virtuálním zprovozněním víceosého manipulátoru, který bude schopen vytvářet obrazce pomocí kulových segmentů podle předlohy. Při návrhu je kladen důraz na vizuální vzhled a jednoduché ovládání manipulátoru. V práci je představen návrh modelu, metody řešení inverzní kinematiky, zprostředkování komunikace použitých softwarů a program pro řízení robota.

## Abstract

This thesis deals with the design and virtual commissioning of multi-axes manipulator, that will be able to create figures using spherical segments according to a template. In the design, emphasis is placed on the visual appearance and simple operation of the manipulator. The thesis presents the design of the model, methods for inverse kinematics solving, communication of used software and the robot control.

## Klíčová slova

PLC, robotický manipulátor, SCARA robot, Siemens NX, inverzní kinematika, Matlab, OPC UA, algoritmus manipulátoru, HMI, TwinCAT, zhodnocení rizik

## Keywords

PLC, robotic manipulator, SCARA robot, Siemens NX, inverse kinematics, Matlab, OPC UA, manipulator algorithm, HMI, TwinCAT, risk assessment

## Bibliografická citace

SLAVÍK, Vojtěch. *Virtuální zprovoznění víceosého manipulátoru.* Brno, 2023. Dostupné také z: https://www.vut.cz/studenti/zav-prace/detail/149557. Diplomová práce. Vysoké učení technické v Brně, Fakulta strojního inženýrství, Ústav mechaniky těles, mechatroniky a biomechaniky. Vedoucí práce Vojtěch Štěpánek.

## Čestné prohlášení

Prohlašuji, že jsem diplomovou práci na téma *Virtuální zprovoznění víceosého manipulátoru* vypracoval samostatně s použitím materiálů uvedených v seznamu literatury.

Vojtěch Slavík

V Brně . . . . . .

. . . . . .

## Poděkování

Tímto děkuji vedoucímu práce Ing. Vojtěchovi Štěpánkovi, Ph.D. a panu Ing. Daliborovi Červinkovi, Ph.D. za cenné rady, připomínky a věnovaný čas během zpracování této diplomové práce.

# Obsah

# Úvod

S pokrokem vývoje technologií, a zvláště elektrických strojů, se čím dál více zvyšuje důraz na automatizaci výrobních systémů. Pro automatizaci se ve velké míře využívají různé druhy robotů a manipulátorů, které jsou řízeny k plnění určitých funkcí. K ovládání těchto zařízení mohou kromě velkých a složitých počítačů sloužit i menší a cenově dostupnější řídící jednotky PLC. Většinou se pak v rámci výrobní linky jedná o použití více různých druhů robotů, protože každý z nich disponuje jistými výhodami a nevýhodami. Zvolení správného typu robota pro požadovanou aplikaci nemusí být vždy jednoduché a jednoznačné řešení. Aby se předešlo nesprávné volbě, je v dnešní době virtuální zprovoznění stroje často využívanou metodou. Tato metoda může ušetřit čas i finanční náklady, a především výrazně snížit riziko chyb během konstrukčního procesu.

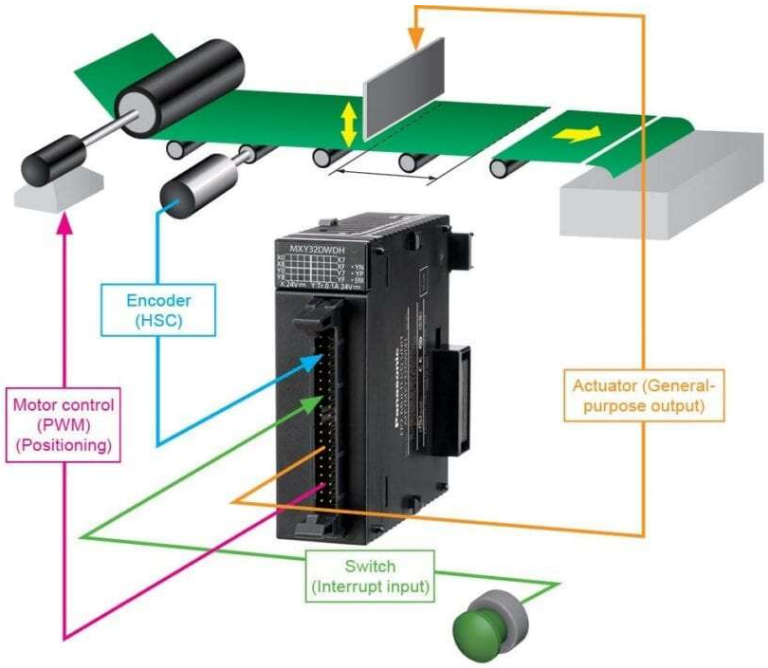
Tato práce se zabývá výběrem vhodné kinematiky manipulátoru, tvorbou modelu a řízením pro aplikaci vytváření binárních obrazců pomocí kulových segmentů. Na základě předlohy zadané uživatelem bude virtuálně zprovozněný model manipulátoru přenášet bílé a černé kuličky z podavače na pole tak, aby byl vytvořen obrazec.

Práce popisuje, jakým způsobem, a jaká byla zvolena kinematika víceosého manipulátoru. Dále se práce zabývá návrhem modelu a jednotlivých komponent zvoleného manipulátoru v 3D CAD modeláři. Protože při návrhu modelu hraje velkou roli i výkonnost elektrických pohonů, byla do práce zařazena i kapitola zabývající se jejich dimenzováním. Součástí práce je také řešení inverzní úlohy pro výpočet kloubových souřadnic a řízení modelu. Řízení probíhá pomocí vytvořeného stavového automatu a za pomocí zprostředkování komunikace mezi použitými softwary, pro kterou byl použit OPC UA server. V práci jsou vysvětleny postupy při návrhu modelu a výstupem práce je pak virtuální zprovoznění manipulátoru s ověřením funkčnosti řízení a použitelnosti vybrané kinematiky.

# 1 Současný stav poznání řešené problematiky

## 1.1 PLC

Programovatelné logické automaty (PLC) patří do skupiny počítačových zařízení a již dlouhou dobu se využívají především v průmyslových automatizacích k měření a regulaci různých aplikací a procesů. Řízení probíhá pomocí informací přicházejících ze senzorů, které sledují stav procesu, a na základě řídícího programu pak akční členy ovlivňují chování systému. [1]




Obrázek 2.1: PLC řízení [2]

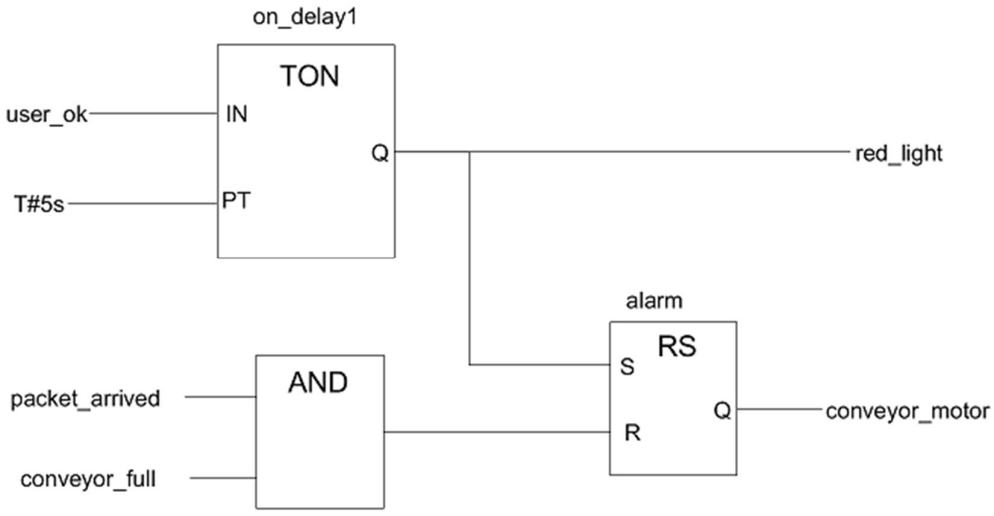
PLC s PC sdílí podobnou architekturu, která obsahuje základní desku, procesor, paměť a rozšiřující sloty. Oproti PC má PLC mikroprocesor spojený s pamětí a I/O (vstup/výstup) prostřednictvím paralelní adresové, datové a řídicí sběrnice. Pro ukládání dat a programů slouží pevná paměť, a místo monitoru, který je součástí PC, je pro rozhraní člověk-stroj využíván HMI panel, zobrazující stav procesu a stroje. Výhodou PC je současné vykonávání více úkonů, zatímco pro PLC se typické provádění úkonů popořadě (od první až do poslední instrukce). PLC jsou navrženy tak, aby co nejvíce usnadňovaly práci při instalaci a údržbě a jejich součástí je i programovací jazyk. [3]

Všechny PLC systémy se skládají ze základních bloků, které slouží k detekci příchozích informací, jejich zpracování a následného ovládání výstupu. Základní bloky se skládají z:

- zdrojové části – napájení jednotky a další modulů
- procesorové části – CPU, FPGA
- vstupní/výstupní části – poskytnutí rozhraní mezi provozními zařízeními a CPU
- komunikačního rozhraní – základní propojení s PC a dalšími PLC

Základní blok PLC může být dále rozšířen o libovolné další prvky jako jsou expanze vstupů a výstupů, komunikační sběrnice a speciální moduly jako jsou např. regulátory motorů. [1] [3]

Při programování PLC se využívají různé typy programovacích jazyků, od vysoko-úrovňových, které jsou grafické s velkými možnostmi strukturování, až po nízko-úrovňové, blízké strojovému jazyku. Obecně se PLC programují v následujících třech jazycích: žebříčkový, STL a funkční blokový diagram. Programovací software pak poskytuje různé typy bloků, ve kterých uživatel vytváří program. V závislosti na požadavcích procesu může být program strukturován do bloků, kterými jsou: organizační blok, funkce, datový blok a funkční blok. [4]




Obrázek 2.2: Diagram funkčních bloků [5]

Robotické manipulátory našli uplatnění především v automatizovaných dopravních linkách, jako jsou různé balírny apod., pro operaci pick and place. Řídicí systémy robotů jsou poměrně uzavřené systémy, které není možné upravit bez podpory od výrobce. Ve skutečnosti ale existuje několik otevřených architektur pro řízení robotů. Při použití těchto architektur mají výhodu PC, protože není potřeba složitých vývojů softwarů, díky vysoce-úrovňovým programovacím jazykům a dobře navrženému vývojovému prostředí. Na druhou stranu standardní PC nedisponuje požadovanou odolností v průmyslu. Různé ochranné prostředky a robustnější komponenty pak navyšují konečné náklady na PC řízeného robota.

PLC jsou řídicí zařízení, které hrají významnou roli právě kvůli jejich robustnosti a snadnému použití. Jejich výkon běžně nedosahuje výkonu PC, a proto jsou považovány za nízko-úrovňové systémy k řízení jednoduchých signálů. PLC programy jsou otevřenými systémy a je možné je rozšiřovat i bez původního zdrojového kódu. I přes tuto výhodu bylo původně řízení manipulátorů založené na PLC omezené na pouze jednoduché typy. Z toho důvodu muselo být řízení a programování systémů provozováno odděleně. Toto řešení však zahrnovalo řadu problémů, jako je znalost obou systémů nebo složitější hledání poruch. Moderní PLC však nabízí dostatečný výpočetní výkon i pro úlohy řešení kinematiky robota. [6]

## 1.2 Robotický manipulátor

Manipulátor neboli robotické rameno je přeprogramovatelné a multifunkční mechanické zařízení, které slouží k transportu předmětů nebo nástrojů prostřednictvím naprogramovaných pohybů. Jedná se o zařízení složené z několika spojených segmentů, na jejichž počtu závisí počet stupňů volnosti manipulátoru. U běžně používaných manipulátorů se počet stupňů volnosti pohybuje od dvou do šesti, a ve zvláštních případech i více. Počet stupňů volnosti je roven počtu nezávislých pohybů, které může soustava vykonávat a každý nezávislý pohyb je vybaven příslušným pohonem. [7]

Manipulátory jsou rozděleny do dvou základních tříd, kterými jsou sériové a paralelní manipulátory. Sériový manipulátor má otevřený kinematický řetězec, ve kterém je koncový efektor spojen se základnou pomocí členů zapojených do série kinematickými dvojicemi, jejichž počet definuje počet stupňů volnosti manipulátoru. [8]

Obrázek 2.3: Sériový kinematický řetězec [9]

Druhým typem je paralelní manipulátor, který se skládá z pohyblivé platformy spojené se základnou pomocí dvou a více kinematických řetězců. Na rozdíl od sériových manipulátorů, kde jsou pasivní kinematické dvojice, neboli dvojice bez aktivačního členu, výjimkou, jsou u paralelních manipulátorů pasivní kinematické dvojice jedním z hlavních rysů. [8]

Obrázek 2.4: Paralelní kinematický řetězec [10]

### Typy kinematických dvojic

V praxi se nejčastěji využívají sériové kinematické řetězce, které kombinují různé typy kinematických dvojic. Mezi ty nejpoužívanější patří rotační (R) a posuvné (T) kinematické dvojice. Ve zvláštních případech je možné narazit i např. na cylindrickou nebo sférickou. Řetězce se pak popisují symbolickým označením směrem od základny ke koncovému efektoru. Kombinace kinematických dvojic mohou být následovné [11]:

- TTT – řetězec je tvořen pouze posuvnými vazbami, příkladem je kartézský manipulátor
- RTT – první vazba řetězce je tvořena rotační vazbou, dále vazby posuvné
- RRT – dvě rotační a jedna posuvná vazba, typické uspořádání SCARA robota
- RRR – řetězec je tvořen pouze posuvnými vazbami

## 1.3 Inverzní kinematika

U přímé úlohy kinematiky o známém vektoru parametrů natočení $q$ hledáme stav koncového vektoru $X$. Formulace výpočtu přímé úlohy pak vypadá [12]:

$$X = f(q) \tag{2.1}$$

Oproti tomu u inverzní úlohy, ze známého vektoru $X$ kartézských souřadnic koncového efektoru a délek ramen $L1$ a $L2$ manipulátoru hledáme vektor natočení $q$. Zde přichází obtíže, jelikož funkce $f$ je nelineární a pomocí analytického řešení je pro složitější systémy neřešitelná. Výpočet inverze se dá formulovat rovnicí [12]:

$$q = f^{-1}(X) \tag{2.2}$$

Existuje více možných druhů technik a algoritmů pro řešení inverzní kinematiky jako jsou např. metody analytické, numerické nebo metody založené na optimalizaci. Volba dané metody pak závisí na konkrétních požadavcích a omezeních systému, kterými jsou počet stupňů volnosti, složitost kinematiky robota a požadovaná rychlost a přesnost řešení.

## 1.4 Problémy při použití inverzní kinematiky

### Redundance

Redundantním případem se rozumí stav, kdy k požadovanému pohybu manipulátoru není zapotřebí všech stupňů volnosti, kterými manipulátor disponuje. To znamená, že počet stupňů volnosti manipulátoru převyšuje počet požadovaných souřadnic. Typicky redundantní manipulátory, jsou manipulátory s více než 6 stupni volnosti (v prostoru požadováno 6 stupňů volnosti), kde počet nezávislých aktuátorů převyšuje maximální počet požadovaných stupňů volnosti v prostoru. Pokud je manipulátor redundantní, je možné dosáhnout požadované polohy a orientace koncového efektoru nekonečně mnoha způsoby. [13]


Obrázek 2.5: Redundantní manipulátor [14]

### Singularita

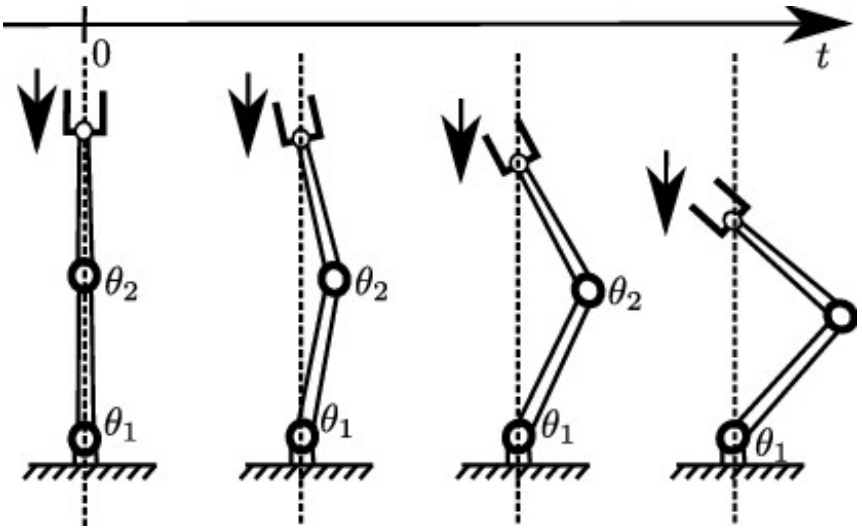
Singulárním stavem se rozumí specifická poloha koncového efektoru, která významně ovlivňuje jeho kinematické vlastnosti. Tuto skutečnost je nutné brát v potaz při řízení manipulátorů. V singulárních polohách dochází ke ztrátě hodnosti jakobiánu *J*:

$$rank(J) < \min(m, n)$$

Kde *m* a *n* je počet řádků a sloupců matice *J*. A ekvivalentně dostaneme:

$$\det(J) = 0$$

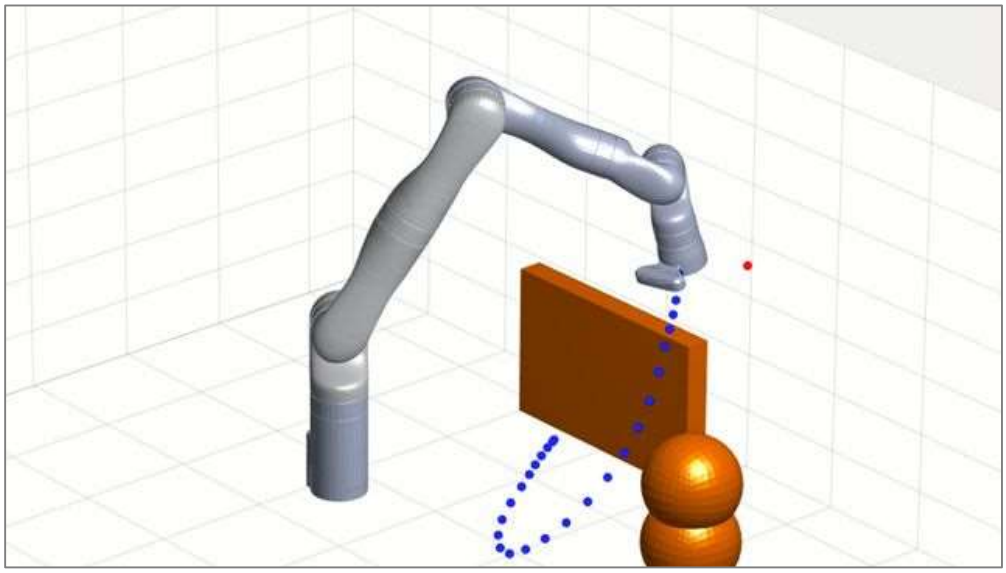
Což znamená, že existují nenulové rychlosti kloubových souřadnic manipulátoru, při kterých je výsledná hodnota koncového efektoru nulová, a tedy manipulátor ztrácí v těchto polohách lokálně 1 stupeň volnosti. I v polohách blízkých singulárnímu stavu, mohou malé rychlosti zobecněných souřadnic způsobit velké rychlosti kloubových souřadnic, a proto je nezbytné se těmto stavům vyhnout. [13]




Obrázek 2.6: Singulární směr a singulární poloha v počáteční konfiguraci (t = 0) [15]

## 1.5 Plánování trajektorie manipulátorů

Cílem plánování trajektorie je generování referenčních vstupů pro řídicí systém manipulátoru, zajišťující provedení požadovaného pohybu. Vstupem do plánovacího algoritmu je určitá geometrické dráha (množina bodů, kterými musí soustava projít na cestě z počáteční do koncové pozice) a kinematické a dynamické omezení. Výstupem je pak trajektorie (dráha + časová posloupnost bodů, ve kterých se bude soustava nacházet) kloubů nebo koncového efektoru vyjádřená jako sekvence polohy, rychlosti a zrychlení. Geometrická dráha bývá obvykle specifikována v operačním prostoru manipulátoru, protože úkony a překážky lze v tomto prostoru přirozeněji popsat. Plánování trajektorie v operačním prostoru se skládá z časového průběhu hodnot, které určují polohu a orientaci koncového efektoru. [16]



Obrázek 2.7: Plánování trajektorie [17]

## 1.6 Softwarové testování

Robotické systémy musí v průmyslu vykazovat vysokou spolehlivost. Velmi nákladnou částí je provádění změn po instalaci nebo uvedení stroje do provozu. Pomocí simulace je možné dopředu vidět všechny možné problémy a předejít jim. Proto je simulace pohybu manipulátorů nezbytnou součástí při návrhu např. robotické buňky. Simulace se využívá k určení optimální velikosti a umístění robota a je klíčovým nástrojem pro ověřování offline programování průmyslového manipulátoru. Téměř v každém případě je při volbě velikosti nejvhodnější vybrat toho nejmenšího možného robota, který bude zvládat vykonávání požadovaného úkonu. Menší roboti mají obecně vyšší zrychlení a pohybují se tedy vyšší rychlostí mezi dvěma body. Použití menších zařízení také bývá obecně přesnější a pořizovací náklady jsou nižší. Pomocí simulačních nástrojů může být vyhodnoceno umístění robota a zda zvolená velikost je dostačující k zajištění dosahu a natočení koncového efektoru pro určený úkol.

Simulace dále pomáhá oddělit vývoj softwaru a hardwaru a umožňuje vývojářům softwaru vytvářet nové vlastnosti a funkce vlastním zrychleným tempem. Vývojáři tedy nemusí čekat na dokončení celého produktu, ale mohou provádět testování už před nebo při jeho vývoji. Po provedení ověření funkčnosti manipulátoru může být program zkopírován do řídicí jednotky a provede se finální úprava programu tak, aby vyhovoval nainstalovaným komponentám. [18]

Obrázek 2.8: Softwarové testování [19]

# 2 Návrh víceosého manipulátoru

Téma této práce vzniklo za motivace návrhu a zprovoznění virtuálního víceosého manipulátoru pro danou funkci. Princip této funkce je poměrně jednoduchý. Na základě uživatelem vytvořeným či nahraným obrazcem v uživatelském prostředí začne manipulátor vytvářet obrazec dle předlohy. Tento obrazec má binární podobu a jeho tvorba spočívá v přeskládávání kulových segmentů ze zásobníků spirálovitého tvaru na prázdné pole o velikosti 10x10 pixelů. Jelikož pole má rozměry 10x10 pixelů, budou zásobníky kulových segmentů navrženy tak, aby obsahovaly 100 kuliček bílé i černé barvy. V uživatelském rozhraní je možné nalézt i další funkce, jako jsou vypnutí nebo zastavení manipulátoru, vyprázdnění pole či nahrání obrazce.

Práce je zajímavá zejména z hlediska využití víceosých manipulátorů a víceosých manipulátorů jako takových, ale také z hlediska propojení a užití více různých softwarů najednou, z nichž každý plní při návrhu svoji funkci. Jelikož se téma práce zabývá virtuálním zprovozněním manipulátoru, jsou tyto funkce především vizuální a výpočetní. Pro propojení softwarů bylo využito komunikačního protokolu OPC UA, který umožnuje u zařízení v průmyslu poměrně jednoduše realizovat komunikaci mezi zařízeními a tím i přenášet naměřená či vypočtená data. Jaké softwary, a jak byly využity, budou dále rozebrány v práci.

Manipulátory jsou zajímavé zejména pro svoji kinematiku, a tudíž pro výpočet jak přímé, tak inverzní kinematiky, které jsou jejich nedílnou součástí. Přímá kinematika se zabývá výpočtem kartézských souřadnic koncového efektoru ze zadaného natočení kloubových souřadnic, a její výpočet je jednodušší než v případě inverzní kinematiky. Jelikož v tomto projektu jsou známé koncové kartézské souřadnice a je nezbytné vypočíst kloubové souřadnice, bude využito výpočtu inverzní kinematiky. U většiny případů inverzní kinematiky neexistuje analytické řešení, a proto bude dále v práci popsáno numerické řešení této kinematiky.

V dnešní době je virtualizace zařízení stále častěji využívaná metoda pro zjištění funkčnosti, ale také vizuální stránky zařízení a její užití má poměrně vysoký dopad na snížení nákladů při realizaci stroje. Přestože má virtualizace zprovoznění manipulátoru vysoký potenciál v průmyslových aplikacích, tato práce je spíše její ukázkou. Z tohoto důvodu byl při návrhu a zprovoznění manipulátoru kladen důraz především na vizuální stránku.

## 2.1 Výběr vhodné kinematiky pro danou funkci

Aby bylo možné vytvořit model robota, je v první řadě nejdůležitější výběr správné kinematiky. Kinematiku je potřeba vybrat podle několika aspektů. Tím nejdůležitějším je vhodnost pro danou aplikaci. Další kritéria pak jsou složitost konstrukce a řešení, rychlost, přesnost, jelikož se jedná o virtuální provedení, tak svoji určitou roli hraje i zajímavost, a nakonec v malé míře i pořizovací náklady robota. Pro aplikaci skládání kuliček byly vybrány tři typy kinematik, které jsou v průmyslových aplikacích běžně využívány. Těmi jsou kartézský typ, delta a SCARA. Tyto kinematiky budou dále porovnány a bude vybrána nejvhodnější z nich.

### Kartézská kinematika

Jedním z nejjednodušších řešení, které se pro danou aplikaci manipulátoru nabízí je využití kartézského mechanismu. Kinematika kartézského manipulátoru operuje pouze na principu posuvných vazeb (lineární pohyb v osách X, Y a Z), což definuje především jeho konstrukci, ale i možnosti využití. Jedná se o sériovou kinematickou strukturu. Tedy takovou, kde pohyb v jedné vazbě není ovlivňován pohyby v dalších vazbách. Operační prostor tohoto tříosého

upořádání je dán velikostí lineárních vazeb a tvarem tohoto prostoru je hranol. Hlavní výhodou kartézských manipulátorů je vysoká přesnost, čehož je využíváno především u zařízení jako jsou CNC stroje a 3D tiskárny. Další jejich výhodou je díky možné robustnější konstrukci přesun hmotnějších těles. Jelikož v této práci není kladen důraz na zvýšenou přesnost, přenášené kulové segmenty nemají velkou hmotnost a konstrukce by nabývala zbytečně velkých rozměrů, byl tento typ kinematiky zavrhnut. Jedny z dalších možných typů kinematik, které připadají v úvahu jsou kinematika delta a SCARA. Ty nejsou tak přesné, ale za to dosahují vyšších operačních rychlostí. Jejich řešení nejsou tak jednoduchá, jelikož je za potřebí složitějších numerických výpočtů, avšak řešení jsou zajímavější. [20]

### Kinematika typu delta

Delta robot je paralelní robot, který se skládá z mobilní platformy propojené paralelními kinematickými řetězci (rameny) s pevnou základnou. Zároveň je delta robot dokonale paralelním robotem, jelikož počet jeho nohou je větší nebo roven počtu stupňů volnosti mobilní platformy. Každý z jeho ramen má svůj vlastní motor, který je umístěn na základně. Díky tomuto uspořádání mohou být motory umístěné na základně výkonnější než u jiných robotů, a v kombinaci s lehkými rameny dosahuje delta robot vysokých operačních rychlostí, kterým se ostatní roboty nemohou rovnat. Tento průmyslový robot bývá většinou namontován nad pracovní oblastí a je často využíván k přesunu předmětů na dopravnících ve výrobních linkách, kde je rychlost hlavní prioritou. Přední oblasti, ve kterých najdou delta roboti uplatnění jsou potravinářský a lékařský průmysl. [21]

### Kinematika typu SCARA

SCARA robot, neboli Selective Compliance Assembly Robot Arm, je typ sériového průmyslového robota, který se skládá ze dvou rotačních kinematických dvojic vytvářejících obecný rovinný pohyb v rovině X a Y a jedné lineární vazby v ose Z. Většina SCARA robotů disponuje ještě čtvrtým stupněm volnosti, kterým je rotace lineární vazby pro následné natáčení přenášeného segmentu, či natáčení manipulačního nástroje, umístěného na koncovém efektoru. Díky tomu je SCARA vhodná pro velké množství různých vertikálních montážních úkonů. Oproti předešlým typům robotů, u kterých jsou motory většinou umístěny na základně, bývá často jeden z motorů u SCARY umístěn i na samotném rameni, což způsobuje zatížení ramen, a proto jsou vyžadovány masivnější ložiska i motory o dostatečném točivém momentu. Z tohoto důvodu se motory umisťují co nejblíže základně, aby vznikala co nejmenší páka a robot mohl operovat i při maximálním natažení ramen. [22]

### Výběr mezi typem SCARA a typem delta kinematiky

Jelikož kartézské uspořádání bylo zavrženo z důvodu veliké konstrukce, pomalejší operační rychlosti než u typu SCARA či delta a v neposlední řadě i méně zajímavé kinematiky zavržen, zůstal ve výběru typ kinematiky SCARA a delta. Výběr mezi výhodami a nevýhodami těchto typů kinematik pro danou aplikaci nebyl zrovna jasný. Každý z těchto typů má svá pro i proti. Hlavním kladem delta robota je jeho vysoká rychlost, které se SCARA díky své konstrukci nemůže rovnat. Na druhé straně SCARA roboti jsou sice pomalejší, ale za to mají vyšší přesnost. U dané aplikace se přikláním na stranu vyšší rychlosti, jelikož vyšší přesnost v této úloze nehraje velkou roli. SCARA robot sice může disponovat čtvrtým stupněm volnosti, a tedy natáčením koncového efektoru. Avšak v této úloze, kde jsou přemisťovány kulové segmenty, by tento čtvrtý stupeň volnosti postrádal smysl. V případě, že není podmínkou delta robota

velice vysoká rychlost, a tudíž motory nemusí být vysoce výkonné, jsou finanční náklady na tyto roboty podobné a finanční prostředky nehrají při rozhodování příliš velkou roli. Z předešlých úvah vychází lépe delta robot, který má i dle mého názoru zajímavější kinematiku. Je ale nutností brát v potaz, že SCARA robot je určitě vhodnějším typem především pro sběr a pokládání předmětů do podavačů. A jelikož se v této úloze nacházejí spirálovité podavače kuliček, byl vybrán jako nejvhodnější typ kinematiky pro zadanou úlohu typ SCARA. [22], [23] Pro porovnání byla vytvořena tabulka s multikriteriálním zhodnocením.

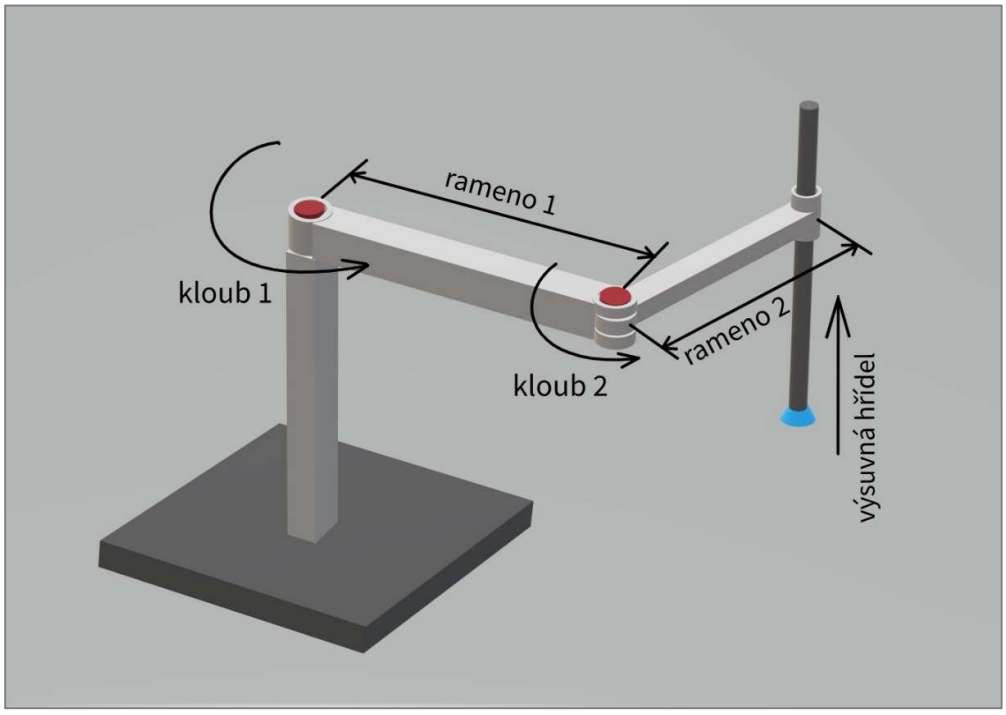
Tabulka 3.1: Multikriteriální zhodnocení kinematiky delta a SCARA

| **Kritéria** | **Váha [%]** | **delta** | | **SCARA** | |
|---|---|---|---|---|---|
| | | Hodnocení | Výsledek | Hodnocení | Výsledek |
| **Kuličkový podavač** | 40 | 40 | 16 | 100 | 40 |
| **Rychlost manipulátoru** | 20 | 100 | 20 | 70 | 14 |
| **Vizuální stránka** | 15 | 90 | 13,5 | 40 | 6 |
| **Operační Prostor** | 10 | 70 | 7 | 60 | 6 |
| **Přesnost manipulátoru** | 8 | 60 | 4,8 | 80 | 6,4 |
| **Hmotnost nákladu** | 5 | 20 | 1 | 60 | 3 |
| **Pořizovací náklady** | 2 | 80 | 1,6 | 90 | 1,8 |
| **Celkový výsledek** | 100 | | 64,2 | | 77,2 |

Pokud pomineme možnost čtvrtého stupně volnosti, a tedy natáčení koncového efektoru, existují tři hlavní typy konstrukce SCARA robotů. První z nich má lineární vazbu v podobě posuvu v ose Z umístěnou na základně. Hlavní výhodou této konstrukce je snížení zatížení ramen motorem zajišťujícím posuv v ose Z nacházejícím se na jednom z rameni a popř. i motorem otáčejícím druhým ramenem. Tento typ konstrukce je však objemnější a využívám spíše jako hobby pro různé ukázky. Druhý typ s umístěním lineární vazby na konci druhého ramene je naopak průmyslově využívaným typem konstrukce, jelikož nezabírá takové množství prostoru. Posledním z hlavních typů SCARA robotů je paralelní provedení druhého typu konstrukce. Dalo by se tedy říci, že se svými rameny a pohybem připomíná horizontálního delta robota. Výhodou této konstrukce je určitě rozložení zatížení do jeho ramen. V této práci jsem se rozhodl zvirtualizovat právě druhý typ konstrukce, který je v praxi používán nejčastěji a jedná se o klasický typ SCAR robota.

## 2.2 Návrh modelu SCARA robota

Na nosné podložce bude přimontována základová část robota, ke které budou připevněny dvě ramena vytvářející obecný rovinný pohyb. Na základové části se bude nacházet první rotační vazba, která bude umožňovat rotaci prvního ramene kolem vertikální osy Z a na konci druhého ramene se bude nacházet druhá rotační vazba sloužící k rotaci druhého ramene kolem ramene prvního. Samotné motory jsou z pohledu hmotnosti nejtěžší součástí a tím způsobují nárůst momentu setrvačnosti. Proto, aby byl moment setrvačnosti konstrukce co nejnižší a bylo tím co nejméně ovlivněno zrychlení ramen, bude třeba motory umístit buď do samotné základové části robota, nebo co nejblíže prvnímu rotačnímu kloubu. Na konci druhého ramene se pak bude nacházet třetí vazba, kterou bude lineární posuv v ose Z. Na jejím konci bude umístěna vakuová přísavka sloužící k úchopu kuliček a jejich přesunu. V závislosti na tomto návrhu bude dále modelován robot typu SCARA. K vytvoření celého modelu bude použit software 3D CAD modelář NX Siemens a pro následné zprovoznění virtualizaci pak jeho nadstavba NX Mechatronics concept designer.

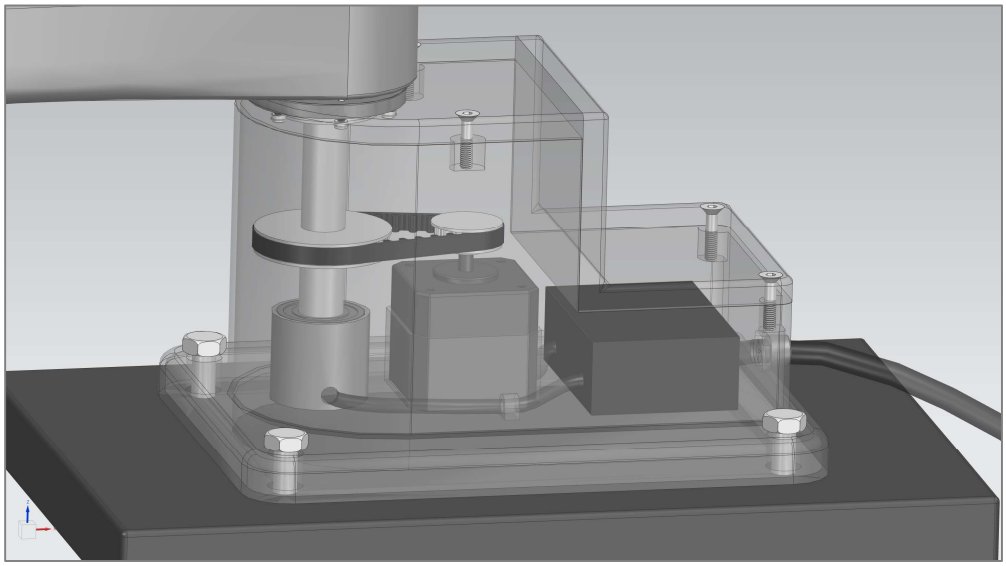



Obrázek 3.1: Návrh modelu SCARA robota

## 2.3 Rozbor modelovaných částí robota SCARA

### Základová část

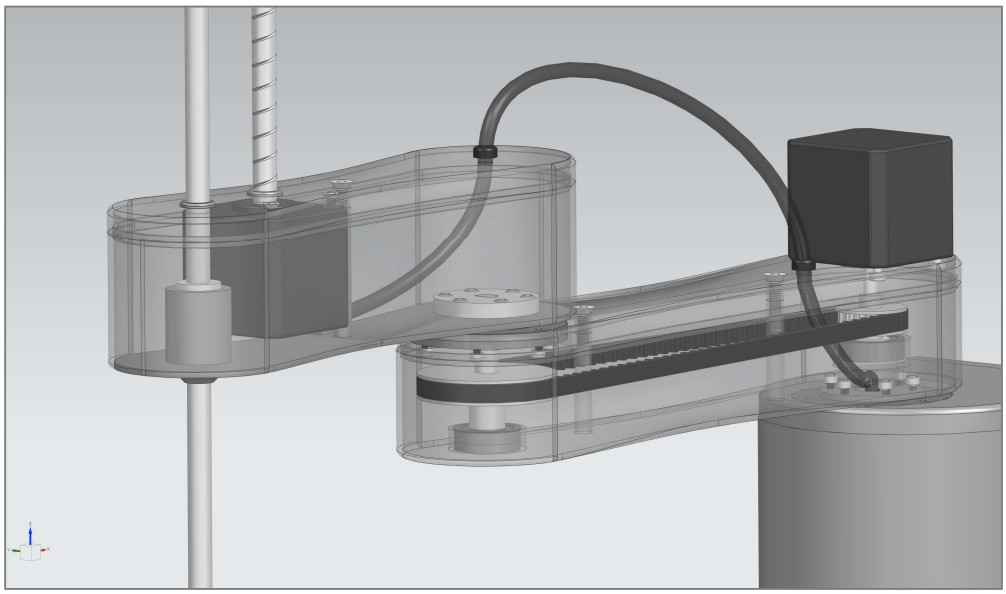
Základová část robota je navržena tak, aby ji bylo možné uchytit na podstavec či podložku. V modelu je tato část právě přichycena k podložce pomocí čtyř šroubů. V této části se nachází i jeden z motorů, který za pomoci řemenu a ozubených kol pohybuje prvním ramenem.



Obrázek 3.2: Základna manipulátoru

### Pohyblivá ramena

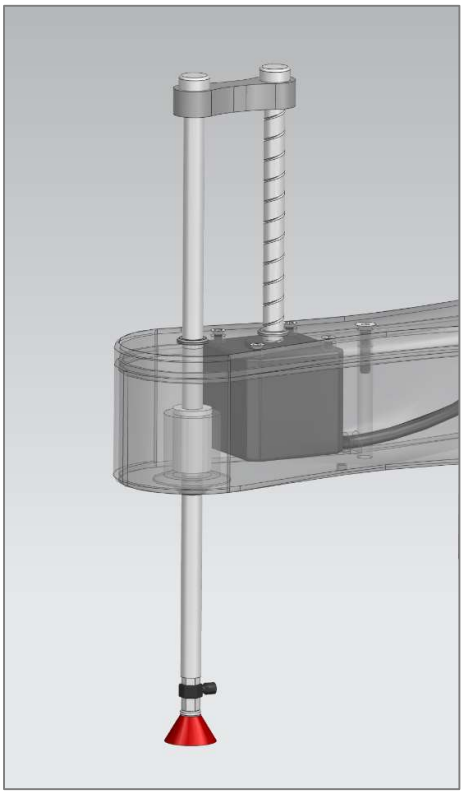
Významnou částí modelované konstrukce jsou dvě pohyblivá ramena. První z nich je přichyceno k základové části a zajištuje rotaci v ose rotace Z kolem základny. Na konci prvního ramene je pak uchyceno druhé rameno, které rotuje kolem ramene prvního. Tyto rotace jsou zajištěny křížovými válečkovými ložisky, která jsou právě pro aplikaci v otočných kloubech robotů určena. Vzdálenost prvního rotačního kloubu, který se nachází mezi základnou a prvním ramenem, od druhého, umístěného mezi prvním a druhým ramenem, je rovna 200 mm. A vzdálenost druhého rotačního kloubu od lineární vazby je 150 mm. Maximální možný obsluhovaný prostor tímto modelem je tedy kruh o poloměru 350 mm, což je pro danou aplikaci dostačující prostor.



Obrázek 3.3: Ramena manipulátoru

### Výsuvná hřídel

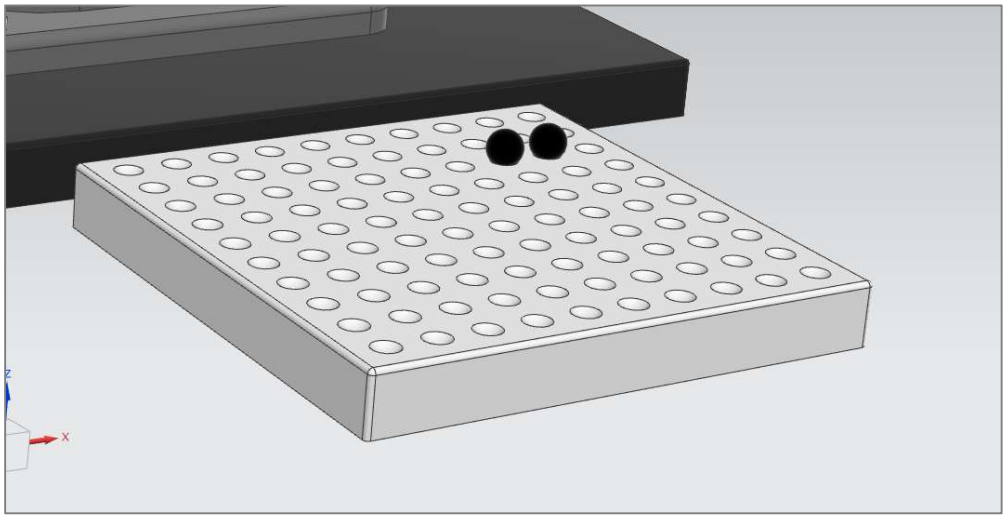
Třetím, a tedy posledním stupněm volnosti modelu robota je lineární vazba nacházející se na konci druhého ramene. Podobu této vazby tvoří výsuvná hřídel upevněná ve střední části lineárním ložiskem, které zajišťuje upevnění hřídele a její přímočarý pohyb v ose Z. Vrchní část hřídele je upevněna v jednom ze dvou otvorů pohyblivé součásti. Uvnitř druhého otvoru této součásti se nachází závit, který za pomoci rotující závitové hřídele uskutečňuje pohyb součásti společně s výsuvnou hřídelí v ose Z. Na spodní části hřídele se pak nachází podtlaková přísavka, která slouží právě k úchopu již zmíněných kuliček, pro jejich následný transport. V této části modelu se ovšem nenachází pneumatická hadička, jelikož by se jednalo při pohybu hřídele o součást měnící svůj tvar. Dosažení této funkce bylo zkoušeno, avšak po delší době se ukázalo, že není jisté, zda je toho vůbec možné dosáhnout. Jelikož by tato funkce měla i záporný dopad na rychlost simulace, bylo rozhodnuto pneumatickou hadičku odebrat.



Obrázek 3.4: Výsuvná hřídel

### Obrazcové pole

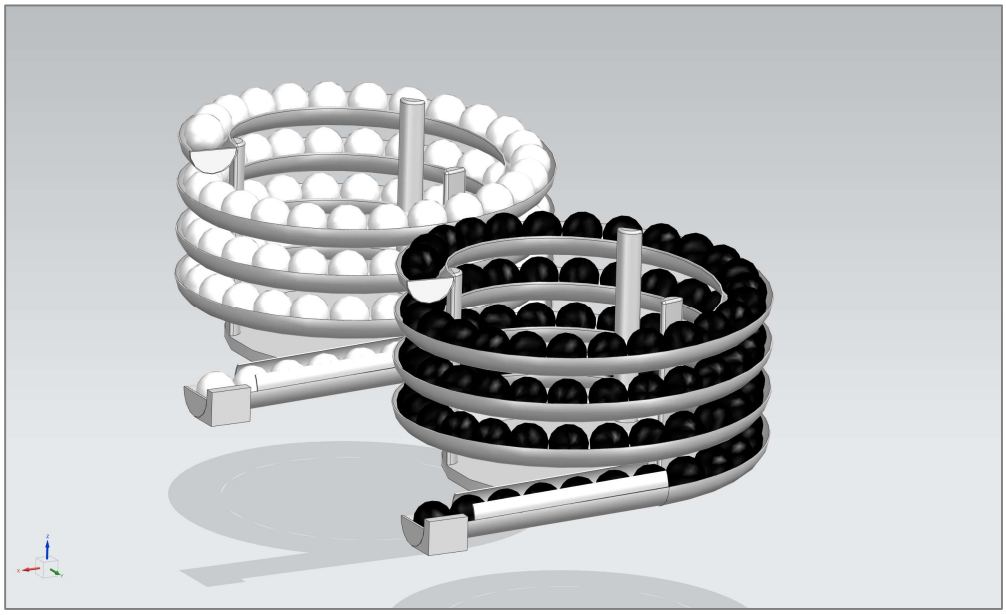
Vzhledem k množství kuliček bylo dohodnuto vytváření binárních obrazců o rozlišení 10x10 pixelů, protože i při tomto poměrně nízkém rozlišení je nutné, aby model obsahoval 100 kuliček od bílé i černé barvy. Kuličky byly modelovány s průměrem o velikosti 15 mm, to znamená, že při zaplnění celé řady deseti kuličkami by rozměry obrazce byly 150x150 mm. Mezi jednotlivé kuličky však byla přidána mezera 5 mm, a tudíž rozměry zaplněného pole vychází na 195x195 mm. Aby bylo možné kuličky na pole pokládat a následně nedošlo k jejich odvalení byly v obrazcovém poli vytvořeny vypouklé otvory, pomocí kterých je možné udržet kuličky na daném místě v poli.



Obrázek 3.5: Obrazcové pole

### Kuličkové zásobníky

Jak již bylo zmíněno, model obsahuje 100 kuliček od každé z barev. To znamená, že každý ze zásobníků musí mít dostatečný prostor pro umístění 100 kusů těchto segmentů. Nejdříve byly vyzkoušeny zásobníky tvaru dutého válce, ve kterém se nacházel otvor, ze kterého se po rampě odvalovaly kuličky. Toto řešení se však ukázalo jako nevhodné, jelikož docházelo k zasekávání kuliček před otvorem, a tudíž nemohly po rampě odkutálet. V závislosti na tomto nevhodném řešení byly následně navrženy zásobníky o tvaru spirály, jelikož u tohoto řešení nedochází právě k onomu zasekávání a kuličky se mohou odvalovat do té doby, než se zastaví o kuličky předchozí. Zásobníky byly navrženy tak, aby zabíraly co nejméně prostoru a vlezlo se do nich právě všech 100 kuliček, které do nich byly umístěny.



Obrázek 3.6: Kuličkové zásobníky

### Umístění krokových motorů

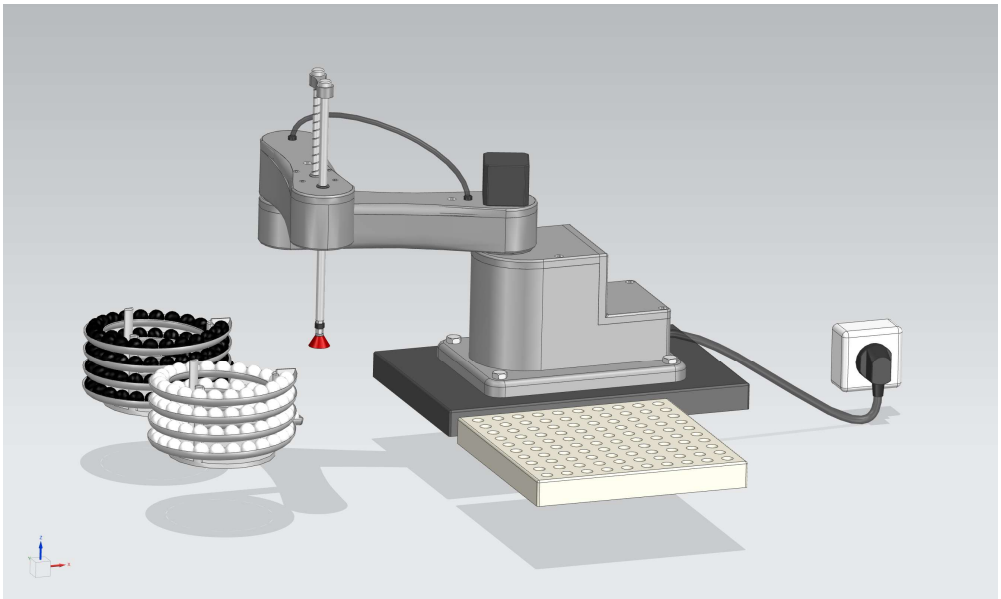
Při tvorbě modelu byl kladen důraz na to, aby byla ramena co nejlehčí a jejich těžiště byla co nejblíže bodu rotace prvního ramene, a proto bylo důležité zamyslet se na rozmístění jednotlivých krokových motorů, které mají vzhledem k použitým materiálům vysokou hmotnost. Pokud by ramena nebyla navrhována tak, aby jejich hmotnost byla co nejnižší a těžiště nebylo co nejblíže bodu rotace prvního ramene, narostl by moment setrvačnosti ramen. Při vyšším momentu setrvačnosti by bylo nutné použít výkonnější motory s vyšším kroutícím momentem, což by způsobilo nárůst odebírané elektrické energie, ale také vyšší pořizovací cenu robota. První motor byl tedy umístěn do základové části, který pomocí ozubených kol, řemenů a duté hřídele otáčí prvním ramenem. Druhý motor byl nejdříve umístěn také do základové části a pomocí hřídele a dvou ozubených kol nacházejících se v prvním rameni otáčel druhým ramenem. Při simulaci však bylo zjištěno, že tento návrh je chybný, jelikož při rotaci prvního ramene nedocházelo k reorientaci motoru pro pohon druhého ramene. To mělo za následek, že první ozubené kolo v rameni č. 1 nerotovalo kolem vertikální osy Z, zatímco druhé ozubené kolo, nacházející se na konci prvního ramene, rotovalo kolem osy Z s poloměrem 200 mm. Tato rotace způsobuje pohyb řemenu mezi těmito ozubenými koly a v konečném výsledku rotaci druhého ozubeného kola kolem osy Z procházející jeho středem. Pomocí simulace a následné úvaze bylo zjištěno, že by bylo nutné první ozubené kolo otáčející druhým ramenem reorientovat v závislosti na natočení prvního ramene. Pohyb jednotlivých komponent je kvůli simulaci řešen inverzně, to znamená, že se polohují ramena vůči sobě a základně a jejich pohyb je vazbami přenášen. Z důvodu rotace druhého ramene kolem prvního a následného převádění pohybu na ozubená kola, by v tomto případě simulace vyžadovala i samostatné ovládání natáčení prvního ozubeného kola, protože při nulovém natočení druhého ramene, se musí otáčet první ozubené kolo v rameni č 1. To by však zvyšovalo délku i náročnost programu. Proto bylo následně rozhodnuto umístit druhý krokový motor pohybující ramenem č. 2 na první rameno. Aby tento motor dále nezvyšoval moment setrvačnosti, byl jeho střed umístěn do osy rotace prvního ramene. Třetí motor už nevyžaduje takový kroutící moment, jelikož pohybuje výsuvnou hřídelí, a tedy i jeho velikost a hmotnost může být nižší než u předešlých dvou motorů. Jelikož by bylo konstrukčně výrazně složitější umístit tento motor do základny, popř. do prvního ramene, byla zvolena jednodušší cesta, umístit tento motor přímo do druhého ramene. Při tomto usazení bylo ale konstrukčně nutné vyřešit přívod kabelů pro napájení.

### Kabeláž

Další důležitou součástí při návrhu manipulátoru je i samotná kabeláž, aby bylo možné napájet motory a ovládat podtlakovou přísavku. Kabely jsou vyvedeny ze základové části, kde vychází nejdříve z driveru pro krokové motory, prochází úchytkou a dále dutou hřídelí až do prvního ramene. Tento způsob konstrukce byl navržen především z důvodu simulace, jelikož při vnějším natažení kabelů by je vzhledem ke konstrukci nebylo možné protáhnout v ose rotace prvního ramene a muselo by tedy docházet k jejich změně tvaru. Díky této konstrukci však první rameno může rotovat kolem osy Z o úhel 360°. Druhé rameno však rotovat o 360° kolem prvního, kvůli výsuvné hřídeli, nemůže, a zároveň je možné kabel protáhnout středem rotace druhého ramene. Proto byl kabel mezi rameny natažen vně konstrukce.

### Celkový navržený model

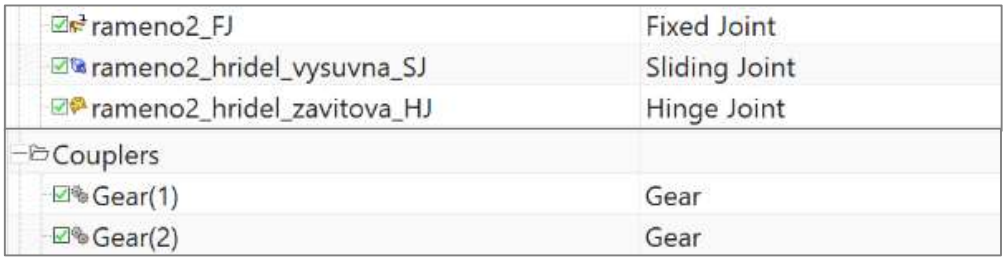
Na následujícím obrázku je možné vidět celý navržený model robotického manipulátoru typu SCARA, který se skládá ze všech výše uvedených komponent. Jednotlivým komponentům bylo nezbytné přiřadit dané vazební a aktivační členy, a nakonec zprostředkovat komunikaci s ostatními programy, pro řízení modelu. Po dokončení těchto kroků mohl být model virtuálně zprovozněn.



Obrázek 3.7: Celkový navržený model v NX

## 2.4 Vazební členy modelu

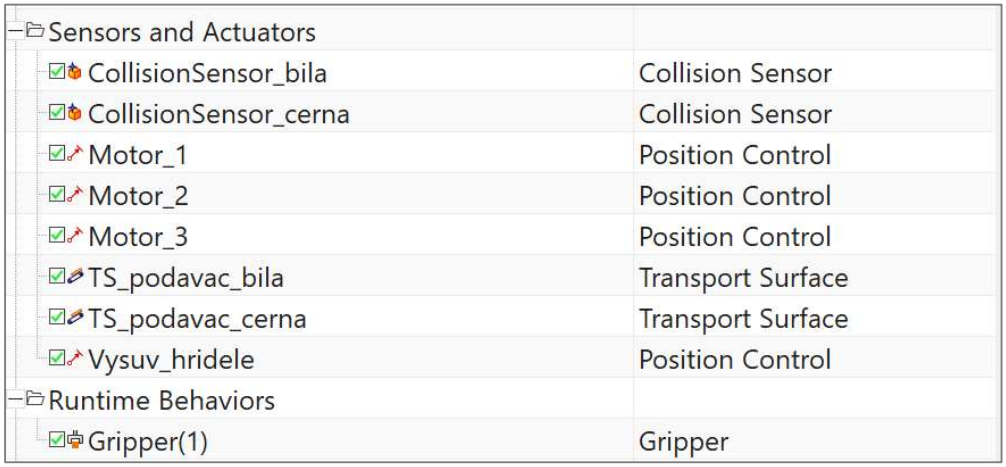
Po vymodelování veškerých potřebných částí manipulátoru a jejich sestavení v 3D CAD modeláři bylo dále nutné přiřadit součástem v nadstavbě Mechatronics Concept Designer dané vazby. Ty byly přiřazovány takovým způsobem, aby se virtuální model choval co nejpodobněji modelu reálnému. Pro součásti, které jsou vůči vlastnímu souřadnému systému nepohyblivé, nebo jsou fixně připevněny k jiným pohyblivým součástem, byla použita fixní vazba. Součástem rotujících kolem vlastní osy, kterými jsou ramena a ozubená kola, byla přiřazena rotační vazba a výsuvné hřídeli pohybující se lineárně v ose Z absolutního souřadného systému vazba lineární. Dále byly přidány převodové vazbové členy mezi ozubenými koly pohybující prvním ramenem, které se nachází v základně, a dvěma ozubenými koly pohybující druhým ramenem umístěné v prvním rameni. Převody byly z důvodu nižšího kroutícího momentu na hřídeli motoru navrženy do pomala, kde druhé ozubené kolo má vždy dvakrát větší poloměr, a proto je převodový poměr pro obě převodové vazby $i = 2$.




Obrázek 3.8: Použité vazební členy modelu

## 2.5 Akční členy modelu

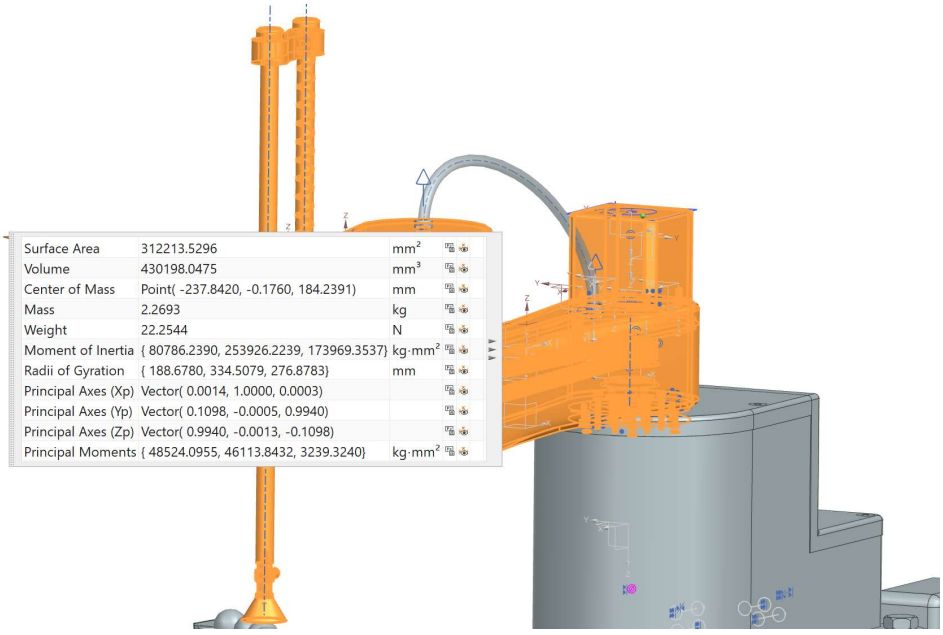
Aby bylo možné modelem pohybovat a ovládat ho pomocí napsaného programu, musely být určitým vazbám přiřazeny akční členy. Pro rotační vazbu mezi základnou a prvním ramenem a rotační vazbou mezi prvním a druhým ramenem byl použit člen pro ovládání jejich natočení ve stupních. V případě lineární vazby výsuvné hřídele pak posuvný pohyb v ose Z v milimetrech. Dále bylo žádoucí, aby rotovala i závitová hřídel, která slouží k přenosu rotace na lineární posuv výsuvné hřídele s podtlakovou přísavkou. Jelikož je ovládán pohyb výsuvné hřídele, musel by se její pohyb inverzně převádět na rotaci závitové hřídele. Toho však nebylo dosaženo a k ovládání její rotace byl proto použit také akční člen. Aby bylo možné kuličky uchopit a následně je přesunout, byl dané součásti představující podtlakovou přísavku přiřazen aktivátor, simulující její chování, tzv. suction cup. Dalším použitým členem byl transportní povrch. Při simulaci bylo zjištěno, že kuličky se v zásobníku nepohybují dle reálného modelu, protože při jejich pohybu docházelo k zasekávání v různých částech zásobníku i při úpravě jejich hmotnosti. U reálného modelu je však jisté, že by k tomuto nesprávnému pohybu nedocházelo a kuličky by v zásobníku plynule procházely. Pro úpravu jejich pohybu byl proto u zásobníku použit člen transportního povrchu, kde se jeho rychlost po ozkoušení nastavila na velmi nízkou hodnotu. Při jeho vysoké rychlosti docházelo u spodních kuliček, nacházejících se v zásobníku, ke kolizím, které způsobovaly jejich vypadávání. Aby bylo možné zjistit, zda kuličky v zásobníku dorazily na místo určené k jejich úchopu, byly do každého zásobníku přidány senzory, které jejich dokutálení kontrolují.




Obrázek 3.9: Akční členy modelu

## 2.6 Návrh kroutícího momentu krokových motorů

Moment setrvačnosti ramen je závislý na jejich vzájemném natočení. Z důvodu proměnného momentu setrvačnosti bude výsledný moment setrvačnosti počítán jako maximální možný, a tedy při plném natažení ramen. Moment setrvačnosti ramen byl vypočten pomocí NX modeláře a jeho velikost pro dimenzování prvního motoru v ose Z je rovna $J_0 = 173\,970\ \mathrm{kg \cdot mm^2}$. Při tomto výpočtu se neuvažovalo se zátěží, jelikož kuličky jsou malé a lehké. Softwarový výpočet je možné vidět na následujícím obrázku.




Obrázek 3.10: Softwarový výpočet momentu setrvačnosti

### První motor

Tento moment setrvačnosti je však stanoven k ose procházející těžištěm ramen. Při rotaci ramen v prvním rotačním kloubu je však pro výsledný moment setrvačnosti nutné použít Steinerovu větu, pomocí které je možné tento moment setrvačnosti přepočítat. [24]

$$J = J_T + m \cdot r^2 \tag{3.1}$$

Kde $J$ představuje moment setrvačnosti k ose rotace, $J_T$ moment setrvačnosti k ose procházející těžištěm, $m$ hmotnost tělesa a $r$ vzdálenost osy rotace od těžiště, která je v modelu posunuta o 60 mm od absolutního souřadného systému.

Pro první kloub po zaokrouhlení a dosazení vyjde moment setrvačnosti následovně:

$$J_1 = 0{,}174 + 2{,}27 \cdot 0{,}178^2 = 0{,}246\ \mathrm{kg \cdot m^2} \tag{3.2}$$

Dále je nutné připočíst moment setrvačnosti hřídele otáčející rameny, který byl opět zjištěn pomocí NX funkce, a jeho velikost je $J_{hříde} = 0{,}057\ \mathrm{kg \cdot m^2}$. Tyto vypočtené momenty setrvačnosti bylo třeba přepočítat na rotaci hřídele motoru. Do vzorce vstupuje převodový poměr, který je roven podílu rychlostí ozubených kol $i = \frac{\omega_1}{\omega_2} = 2$.

Vzorec pro výpočet redukovaného momentu setrvačnosti pak vypadá [25]:

$$J_{red} = (J_1 + J_{hřídel}) \cdot \frac{1}{i^2} \tag{3.3}$$

$$J_{red} = (0{,}246 + 0{,}057) \cdot \frac{1}{2^2} \cong 0{,}077 \text{ kg} \cdot \text{m}^2 \tag{3.4}$$

Když je vypočítána velikost redukovaného momentu setrvačnosti, bude proveden výpočet celkového momentu setrvačnosti [25]:

$$J_c = J_{red} + J_m \tag{3.5}$$

Kde $J_m$ je moment setrvačnosti rotoru, který vychází z tabulek. Po dosazení dostaneme celkový moment setrvačnosti:

$$J_c = 0{,}077 + 0{,}00004 = 0{,}077 \text{ kg} \cdot \text{m}^2 \tag{3.6}$$

Pokud známe celkový moment setrvačnosti a víme s jakým úhlovým zrychlením se má robot pohybovat, je možné provést výpočet dynamického momentu. Úhlové zrychlení se určí podle úhlu, kterého má segment dosáhnout za daný čas. V tomto případě je chtěno, aby se segment stihl otočit za jednu sekundu o 120°, a proto bude požadováno zrychlení $240° \cdot \text{s}^{-2}$. Dynamický moment se pak vypočte [25]:

$$M_d = J_c \cdot \varepsilon \tag{3.7}$$

$$M_d = 0{,}077 \cdot 240 \cdot \left(\frac{\pi}{180}\right) = 0{,}323 \text{ Nm} \tag{3.8}$$

Do výsledného momentu je nezbytné pak zahrnout jak moment dynamický, tak moment statický. Statický moment, ale nebylo možné určit a k zjištění jeho velikosti by musela být provedena měření na reálném modelu. Jelikož reálný model není k dispozici, byl statický moment odhadnut na 0,15 Nm. Celkový moment je pak po započítání účinnosti převodů (odhadnuté na $\eta = 0{,}95$) odhadnut na 0,5 Nm.

Z důvodu životnosti a správné funkce je potřeba předepsat kroutící moment motoru s minimálně 2 krát vyšší hodnotou, než je celkový moment. Proto by v tomto případě byl zvolen motor o kroutícím momentu 1,1 Nm.

### Druhý motor

Kroutící moment druhého motoru se vypočítá stejným způsobem, kde:

$$J_2 = 0{,}145 + 1{,}06 \cdot 0{,}107^2 = 0{,}157 \text{ kg} \cdot \text{m}^2 \tag{3.9}$$

$$J_{red} = (0{,}157 + 0{,}006) \cdot \frac{1}{2^2} \cong 0{,}041 \text{ kg} \cdot \text{m}^2 \tag{3.10}$$

$$J_c = 0{,}041 + 0{,}00002 = 0{,}041 \text{ kg} \cdot \text{m}^2 \tag{3.11}$$

Pro výpočet dynamického momentu bude bráno stejné zrychlení druhého ramene jako u ramene prvního a může se psát:

$$M_d = 0{,}041 \cdot 240 \cdot \left(\frac{\pi}{180}\right) = 0{,}172 \text{ Nm} \tag{3.12}$$

Po započítání účinnosti a statického momentu by bylo vhodným řešením zvolení kroutícího momentu motoru 0,6 Nm.

### Třetí motor

Pro výpočet kroutícího momentu třetího motoru bude třeba vzorce upravit, jelikož dochází k převodu úhlového natočení na posuv hřídele, a také zde bude potřeba započítat statický moment pro udržení výsuvné hřídele. Nejdříve se vypočte statický moment pomocí vzorce [26]:

$$M_s = \frac{p}{2\pi} \cdot (F_G + \mu \cdot m \cdot g) \tag{3.13}$$

Kde $p$ je stoupání závitu $p = 8$ mm, $F_G$ tíhová síla působící na výsuvnou hřídel, $\mu$ součinitel tření stanoven na hodnotu 0,2 a $m$ hmotnost výsuvné hřídele zjištěna pomocí NX o velikosti $m = 0{,}153$ kg, pokud budeme počítat hmotnost kuličky 10 g. Po dosazení se získá statický moment:

$$M_s = \frac{0{,}008}{2\pi} \cdot (0{,}153 \cdot 9{,}81 + 0{,}2 \cdot 0{,}153 \cdot 9{,}81) = 0{,}002 \text{ Nm} \tag{3.14}$$

Pro výpočet dynamického momentu je nejdříve nutné vypočítat redukovaný moment setrvačnosti a moment setrvačnosti rotoru motoru. Jelikož rotor motoru má velice nízkou hodnotu momentu setrvačnosti, bude se brát v potaz při výpočtu redukovaného momentu setrvačnosti pouze moment závitové hřídele $J_{hřídel\_závit} = 0{,}0083 \text{ km} \cdot \text{m}^2$, rychlost výsuvné hřídele $v = 0{,}05 \text{ m} \cdot \text{s}^{-1}$ a její hmotnost $m$ [25].

$$J_{red} = J_c = J_{hřídel\_závit} + m \cdot \frac{v^2}{\omega^2}$$

Při požadavku rychlosti výsuvu hřídele $v = 0{,}05 \text{ m} \cdot \text{s}^{-1}$, je potřeba, aby rychlost rotace závitové hřídele byla rovna $\omega = 6{,}25 \cdot 2\pi \text{ rad} \cdot \text{s}^{-1}$, neboli 6,25 otáček za sekundu. Druhý člen v rovnici pro výpočet redukovaného momentu kvůli zvoleným hodnotám nabývá minimálních hodnot a celkový moment setrvačnosti vyšel $J_c = 0{,}0083 \text{ kg} \cdot \text{m}^2$. Požadované úhlové zrychlení závitové hřídele $\varepsilon$ pak musí odpovídat dvojnásobku úhlové rychlosti $\omega$ a dynamický moment se vypočte:

$$M_d = J_c \cdot \varepsilon \tag{3.15}$$

$$M_d = 0{,}0083 \cdot 12{,}5\pi = 0{,}326 \text{ Nm} \tag{3.16}$$

Celkový výkon se pak spočte součtem momentů statického a dynamického:

$$M_c = 0{,}326 + 0{,}002 = 0{,}328 \text{ Nm} \tag{3.17}$$

Po započítání účinnosti a třecího momentu se z hodnoty výsledného momentu pro ovládání výsuvu hřídele určí jako vhodné řešení motor s kroutícím momentem 0,8 Nm.

Většina krokových motorů nabízí 200 kroků na otáčku, což by znamenalo odchylku $\pm$ 0,9°. Při použití převodovky o převodovém poměru 2 se odchylka sníží na polovinu, ale stále se je odchylka poměrně velká. Pro další snížení odchylky by bylo možné zvýšit převodový poměr, což ovlivňuje požadavky na rychlost rotace motoru. Vhodnějším způsobem snížení odchylky by bylo řízení motoru metodou half step a nižší.

# 3 Řešení inverzní úlohy kinematiky

U daného modelu jsou jednotlivé polohy znány, popř. je možné jejich dopočítání. Aby bylo možné manipulátor ovládat a pohybovat rameny tak, aby koncový efektor dosáhl požadované pozice, bylo třeba využít inverzní kinematiky. To znamená, že ze zadaných pozic [$x$, $y$] a délek ramen budou dopočítány zobecněné souřadnice [$q1$, $q2$] kloubového natočení. (kapitola 2.3)

## 3.1 Analytické řešení

Jedná se o jednu z jednodušších metod řešení inverzní kinematiky, založené na soustavě goniometrických rovnic. Tato metoda řešení inverzní kinematiky je však použitelná pouze u jednodušších typů manipulátorů, tzn. s nižším počtem stupňů volnosti. Jelikož se v této úloze jedná o rovinný RR manipulátor s posuvnou vazbou na jeho konci, lze říci, že se jedná právě o jednodušší typ manipulátoru, u kterého je možné tuto metodu použít.

Pokud máme ramena o délkách $L1$ a $L2$ a je znána poloha koncového efektoru [$x$, $y$] lze pomocí Pythagorovy věty dopočítat délku přepony $d$.

$$d^2 = x^2 + y^2 \tag{3.18}$$

Přepona $d$ tedy tvoří společně s odvěsnami o délkách $x$ a $y$ pravoúhlý trojúhelník. Zároveň ale přepona $d$ tvoří trojúhelník s odvěsnami $L1$ a $L2$. Úhel $q2$ mezi rameny není znán, ale je možné ho dopočítat pomocí kosinové věty, která po dosazení odvěsen zní:

$$d^2 = L1^2 + L2^2 - 2 \cdot L1 \cdot L2 \cdot \cos(q2) \tag{3.19}$$

Úhel natočení druhého ramene lze po upravení rovnice vypočíst následovně:

$$q2 = cos^{-1}\left(\frac{d^2 - L1^2 - L2^2}{2 \cdot L1 \cdot L2}\right) \tag{3.20}$$

Oproti natočení druhého ramene je úhel natočení prvního ramene o poznání složitější. Je třeba si rozdělit natočení přepony $d$ od osy na dva úhly. Úhel $q1$ mezi prvním ramenem a osou a úhel $\beta$ přepony $d$ od prvního ramene. Pomocí funkce tangent, kterou lze použít pro pravoúhlý trojúhelník, lze najít závislost mezi úhlem $q1$ a $\beta$ [27]:

$$\tan(q1 + \beta) = \frac{y}{x} \quad => \quad q1 = tan^{-1}\left(\frac{y}{x}\right) - \beta \tag{3.21}$$

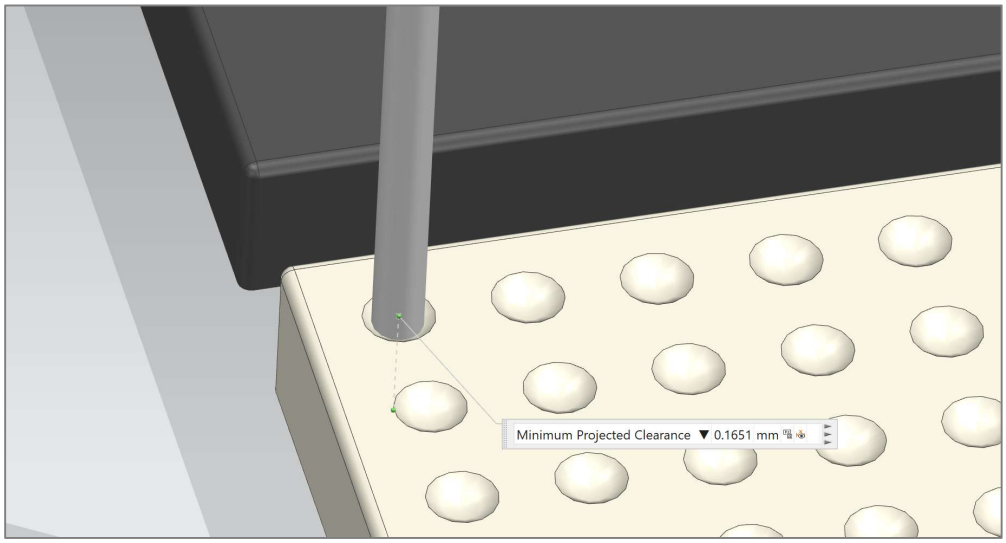
Kde úhel $\beta$ lze opět pomocí goniometrických funkcí a úhlu natočení $q2$ zapsat jako [27]:

$$\beta = tan^{-1}\left(\frac{L2 \cdot \sin(q2)}{L1 + L2 \cdot \cos(q2)}\right) \tag{3.22}$$

Po dosazení výpočtu úhlu $\beta$ do rovnice 3.21 získáme výsledný vztah pro výpočet kloubové souřadnice $q1$ [27]:

$$q1 = tan^{-1}\left(\frac{y}{x}\right) - tan^{-1}\left(\frac{L2 \cdot \sin(q2)}{L1 + L2 \cdot \cos(q2)}\right) \tag{3.23}$$

Bylo tedy vytvořeno analytické řešení, které bylo dosazením vypočtených hodnot otestováno na vytvořeném modelu a výsledek tohoto otestování je možné vidět na následujícím obrázku.




Obrázek 3.11: Odchylka analytického řešení

Na obrázku č. 3.11 je možné vidět dosazení vypočtených kloubových souřadnic do pozičního řízení rotačních vazeb ramen v modelu. Při dosazení se však největší výsledná odchylka koncového efektoru od středu výřezu určeného pro položení kuličky nacházela v ose Y a její hodnota činila 0,165 mm, v ose X pak 0,127 mm. Podobné hodnoty odchylek bylo možné pozorovat i pro následující polohy. Jelikož tato metoda počítá kloubové souřadnice pomocí goniometrických funkcí, vzniká odchylka především zaokrouhlovací chybou, která je v mezích přijatelnosti přesného řešení. Ověřením bylo zjištěno, že analytická metoda pro jednoduché mechanismy je přesná a pro danou aplikaci je to vhodné řešení. Na druhou stranu je toto řešení méně často užívanou metodou a je omezeno složitostí mechanismů. Z tohoto důvodu, bylo dále vytvořeno i numerické, neboli iterační řešení, které je však výpočetně složitější, což vyžaduje vyšší výpočetní výkon.

## 3.2 Numerické řešení

Jedná se o všeobecně využívanou metodu řešení inverzní kinematiky. Protože je funkce *f* nelineární (kapitola 2.3), je potřeba ji linearizovat a k tomu se využije jacobián. Jacobiho matice je totiž matice parciálních derivací vektorové funkce:

$$J = \begin{bmatrix} \frac{\partial f1}{\partial q1} & \frac{\partial f1}{\partial q2} \\ \frac{\partial f2}{\partial q1} & \frac{\partial f2}{\partial q2} \end{bmatrix}$$

To znamená ze determinant jacobiho matice, neboli jacobián, udává míru změny v daném místě. Pro přímou úlohu se bude rovnice za použití jacobiánu formulovat:

$$\Delta X = J \cdot \Delta q \tag{3.24}$$

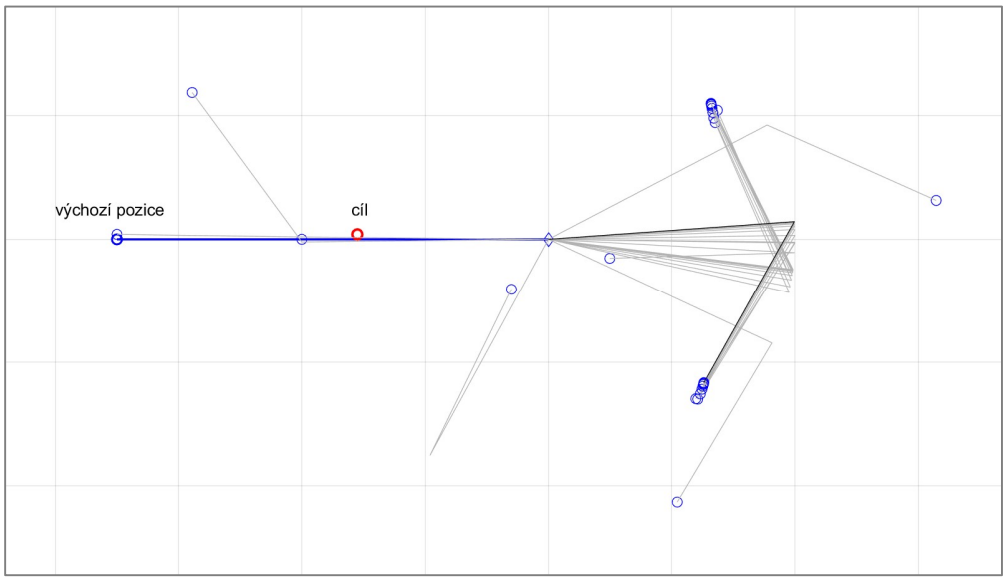
A pokud tedy chceme zapsat rovnici ve tvaru pro výpočet inverní úlohy, bude vypadat následovně:

$$\Delta q = J^{-1} \cdot \Delta X \tag{3.25}$$

Po dosazení rozdílu souřadnic koncového efektoru za $\Delta X$ získáme změnu stavu systému $\Delta q$. Zde může nastat problém s příliš velkou hodnotou $\Delta X$ (rozdílem požadovaného $x_{cíl}$ a aktuálního $x_k$ vektoru). Jakobián totiž závisí na aktuálním stavu sytému a výpočet má tedy smysl jen pro malé rozdíly souřadnic koncového efektoru. V opačném případě by to mohlo vést ke špatné konvergenci řešení. Aby bylo tomuto problému zabráněno, je nutné omezovat maximální natočení $\Delta q$. Jedna z metod, jak toho docílit, je snížení hodnoty $\Delta X$ při obdržení příliš velkého $\Delta q$. Aby se došlo k rozumnému výsledku, je potřeba tento výpočet opakovat. Iterační proces se bude realizovat [28]:

$$q_{k+1} = q_k + J^{-1} \cdot (x_{cíl} - x_k) \tag{3.26}$$

U metod, které využívají inverze jacobiánu ale může dojít k problémum se singularitou, neboli když determinant jacobiho matice vyjde $\det(J) = 0$, protože neexistuje jeho inverze. V blízkosti singulární polohy se malé rychlosti v kartézských souřadnicích mohou projevit jako velké rychlosti v kloubových souřadnicích a algoritmus vede na špatné hodnoty. Pokud tak nastane, bude se manipulátor nacházet v tzv. singulárním stavu, což má za následek snížení jeho pohyblivosti (kapitola 2.4).




Obrázek 3.12: Singulární stav

Existuje více způsobů, jak se vyhnout singulárnímu stavu. Prvním způsobem je předejití tomuto stavu, který však nemusí být vždy možný. Druhým způsobem je použití metody Levenberg-marquardt, neboli metody nejmenších čtverců s tlumením DLS. Samotná pseudoinverze řeší problém singulárních poloh lépe, protože pomaleji roste do extrémů, ale singulární polohy vyřešit nedokáže. Metoda DLS však při správném zvolení konstanty tlumení $\lambda$ ovlivňující rychlost konvergence dokáže vyřešit i singulární polohy. Vypočet inverzní úlohy pomocí pseudoinverze:

$$\Delta q = J^{\dagger} \cdot \Delta X \tag{3.27}$$

Kde $J^{\dagger}$ se zapíše jako [28]:

$$J^{\dagger} = J^{T}(J \cdot J^{T})^{-1} \tag{3.28}$$

A výsledný vztah pro výpočet inverzní úlohy pomocí metody DLS vypadá [29]:

$$\Delta q = J^{T}(J \cdot J^{T} + \lambda^{2})^{-1} \cdot \Delta X \tag{3.29}$$

Další možnou a na modelu vyzkoušenou metodou je Gaussova eliminace. Gaussova eliminační metoda je metoda řešení soustavy lineárních rovnic algebraických rovnic vedoucí k správnému výsledku v konečně mnoha krocích. Je založena na rozkladu matic jako součin dvou dalších matic, kde první matice má v dolním trojúhelníkovém tvaru na hlavní diagovále číslo jedna a druhá v horním tvora nenulové hodnoty. [30] Pomocí této metody lze vypočítat i inverzi matice, a tedy i inverzní řešení kinematiky. V Matlabu je pro Gaussovu eliminaci funkce, která hledá přibližné řešení, aby se vyhnulo extrémům. Její zápis v Matlabu pak vypadá:

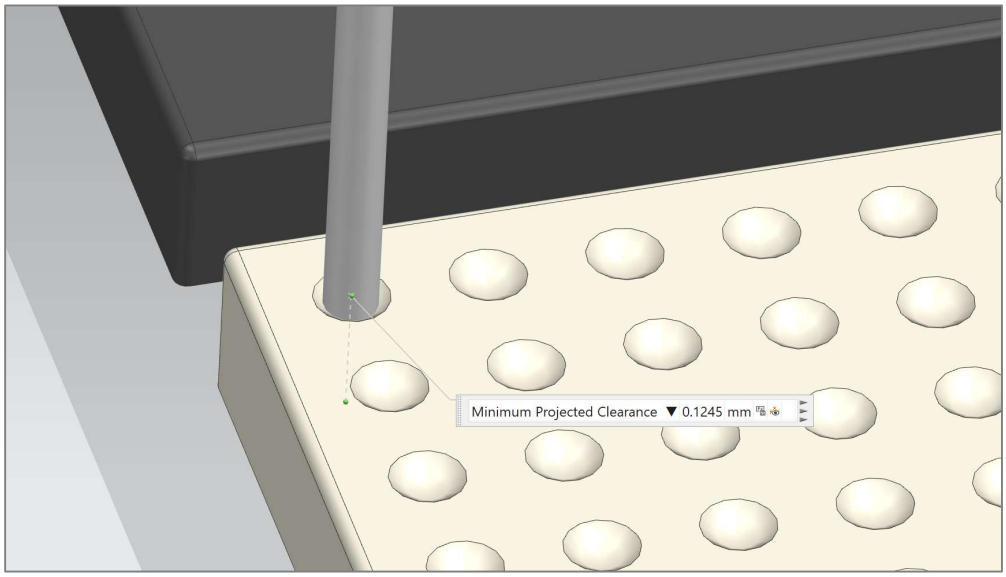
$$\Delta q = inv(J)\backslash\Delta X \tag{3.30}$$

Pro ověření obou metod byly prováděny zkušební výpočty a došlo se k závěru, že obě metody jsou schopné řešit problém se singulárními stavy a proto při výsledném výběru hrál roli výpočetní čas a počet iterací. Zde se ukázalo, že Gaussova eliminační metoda je s výpočetním časem 0,05 s a 425 iteracemi mnohem rychlejší než metoda DLS s výpočetním časem 0,17 s a 3150 iteracemi. Při využití numerické metody pro výpočet kloubových souřadnic je tedy vzhledem k výpočetnímu času Gaussova eliminace vhodnějším řešením.

```
Q = [-pi;0];                    % výchozí natočení
Xcil = [fp1,-fp2]';             % zápis cílových souřadnic
[X, Xend] = fkine(Q,L);         % Výpočet polohy koncového efektoru
err = norm(Xend-Xcil);          % výpočet velikosti chyby

while err > 1e-4                % pokud je chyba větší než 0,1 mm

    J = jacobian(Q, L);         % výpočet parciálních derivací
    dX = Xcil-Xend;             % velikost rozdílu koncového efektoru
    dQ = inv(J) \ dX;           % výpočet změny kloubových souřadnic
    dQ = J'*pinv(J*J' + lambda^2) * dX;  % metoda Levemberg-Mrkvardt
    Q = Q + dQ;                 % výpočet celkového natočení kloubů
    [X, Xend] = fkine(Q,L);     % úprava polohy koncového efektoru
    err = norm(Xend-Xcil);      % úprava velikosti chyby

end                             % opakování při nesplnění podmínky
```

Obrázek 3.13: Numerické řešení

Na obrázku č. 3.13 je možné vidět použitý algoritmus pro výpočet inverzní úlohy. Výsledky vypočtené tímto algoritmem byly dále otestovány na vytvořeném modelu robota, a tím bylo provedeno ověření, zda toto řešení je možné použít.




Obrázek 3.14: Odchylka numerického řešení

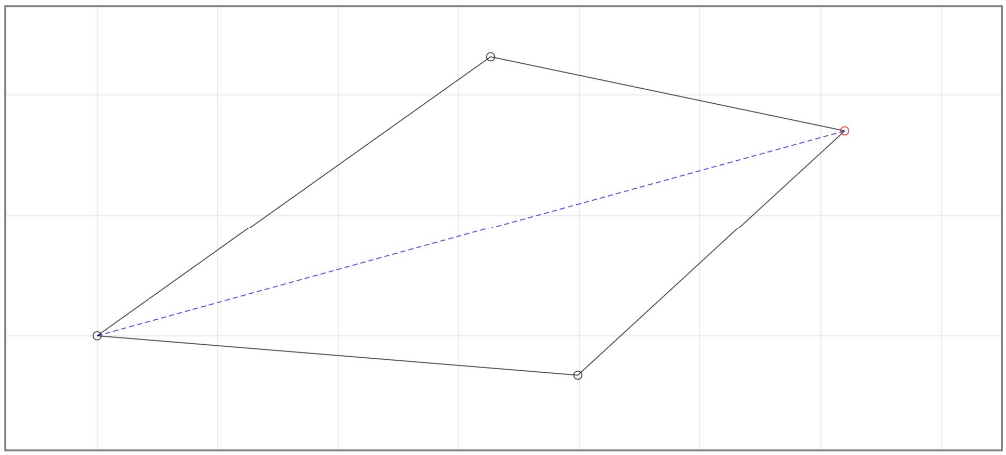
Na obrázku č. 3.14 lze pozorovat polohu koncového efektoru v rovině XY po zapsání vypočtených kloubových souřadnic do modelu. Na první pohled je téměř nerozeznatelné, které z výsledků řešení je přesnější. Po zobrazení odchylky lze však vidět, že výsledky vypočtené numerickou metodou vyšly o minimální rozdíl lépe s největší odchylkou v ose X 0,1245 mm. Stojí ovšem za povšimnutí, že výpočetně má celková výsledná odchylka vycházet méně než 0,1 mm, avšak při simulaci byla zjištěna odchylka větší. Tento rozdíl byl nejspíše způsoben především zatížením ramen a tudíž jejich ohybem a krutem, což ovlivnilo polohu výsledného efektoru. Z malé části mohlo ovlivnit nepřesnost i samotné zaokrouhlování vypočtených kloubových souřadnic.

### Výběr řešení inverzní kinematiky

Ve výsledném rozhodování tedy přesnost nehrála roli, jelikož vycházela u obou řešení podobně. Naopak důležitou roli při výběru řešení hrála metoda výpočtu inverzní úlohy a výpočetní čas. Analytická metoda dle otestování je určitě vhodným řešením pro danou aplikaci s výpočetním časem 0,034 s, ale na druhou stranu se jedná o méně zajímavou metodu, použitelnou pouze pro jednoduché systémy. Oproti tomu numerická metoda je sice také vhodným řešením, ale nabízí zajímavější postup výpočtu. Pokud se zaměříme na délku výpočetního času, je analytické řešení pro jednoduché mechanismy časově méně nákladné, protože numerické řešení realizujeme iteračním procesem, který vyžaduje dostatečný výpočetní výkon. Při použití Gaussovy eliminační metody však výpočetní čas 0,052 s není tolik rozdílný oproti analytickému řešení. Rychlost výpočtu byla ozkoušena na modelu, a ukázalo se, že program výpočetně zvládá numerickou metodu pro realizaci pohybu robota. Proto bylo dále v práci použito, především kvůli zajímavosti, právě numerické řešení.

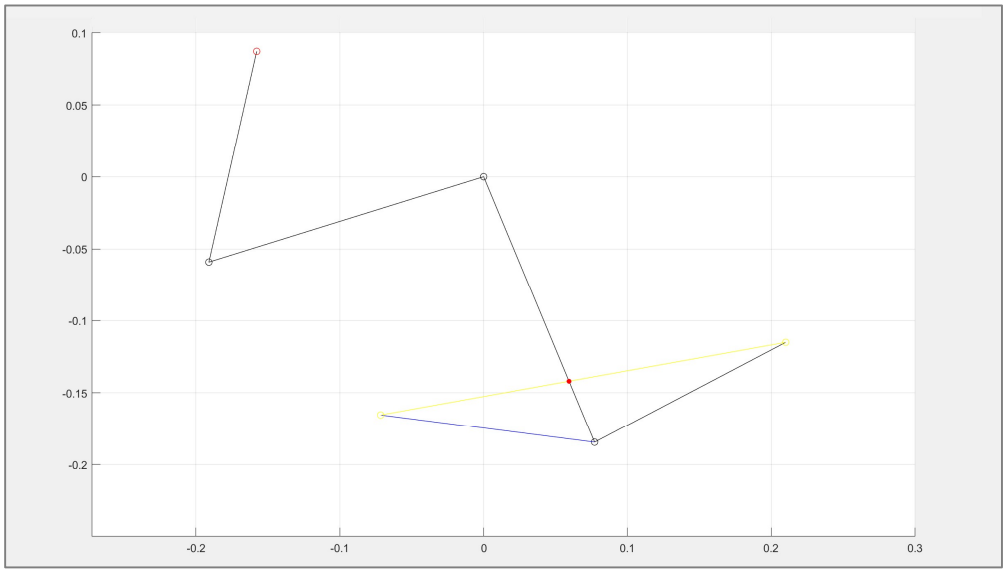
## 3.3 Polohování ramen

Pohyb ramen se realizuje zápisem vypočtených hodnot do aktivačních členů daných vazeb. Tyto aktivátory nabízejí určité možnosti pro změnu pozice, které nabízejí, jakým způsobem chceme dostat pohyblivou součást z bodu A do bodu B. U rotačních vazeb je možné vybrat ze čtyř možností. Nejkratší cestou, rotací po směru hodinových ručiček, pohybem proti směru hodinových ručiček a nakonec sledování více otáček. Jelikož je možné zvolit pouze jeden typ, byla zvolena možnost nejkratšího přesunu. Při tomto nastavení však dochází ke skutečnosti, že v několika případech docházelo k rotaci druhého ramene přes první rameno, neboli druhé rameno procházelo úhlem 180°. To znamenalo, že koncový efektor narazil při pohybu do prvního ramene, což je situce, které se bylo třeba vyhnout, protože by mohlo dojít k destrukci mechanismu. U těchto RR manipulátorů je ovšem známo, že kromě případů, kdy dochází k plnému natažení ramen, existují právě dvě správná řešení. U těchto dvou řešení jsou natočení obou ramen stejná, ale jsou zrcadlově obrácená vzhledem k ose procházející jejich přeponou, jako je možné vidět na obrázku č. 3.15.



Obrázek 3.15: Víceznačnost řešení

To znamená, že pokud program vypočítá kloubové souřadnice tak, že při nejkratší cestě z aktuální pozice, do pozice nastavené, dojde ke kolizi koncového efektoru a prvního ramene, je možné zrcadlově prohodit kloubové souřadnice a tím změnit nejkratší dráhu pohybu druhého ramene. V závislosti na této úvaze byl vytvořen algoritmus, pomocí kterého bude této kolizi zapříčiněno. Základem tohoto algoritmu je porovnávání aktuálního natočení ramen s vypočítaným natočením v pozici, do které se má manipulátor dostat. Při tomto porovnávání se hledá intersekční bod mezi pozicemi koncových efektorů a prvním ramenem. Pokud bod intersekce není nalezen jsou ponechána výpočtená natočení kloubů a k žádným změnám nedochází. Je-li však bod intersekce nalezen, je pomocí výpočtů úhlů natočení získáno zrcadlové natočení ramen, kterých bude dále využito.




Obrázek 3.16: Řešení problému křížení ramen

Na obrázku č. 3.16 je možné vidět případ pozic ramen, které jsou zobrazena černou barvou, u kterých by při pohybu manipulátoru nejkratší cestou došlo ke kolizi s koncovým efektorem. Jak lze na obrázku pozorovat, natočení druhého ramene z výchozí pozice (na obrázku vlevo nahoře znázorněno černou barvou) bylo umístěno na konec prvního ramene v poloze, kde má manipulátor skončit (na obrázku vpravo dole, opět označeno černou barvou) a je označeno modrou barvou. Díky této úpravě mohla být následně provedena intersekce mezi koncovými efektory (na obrázku žlutou barvou) a prvním ramenem. Při existenci intersekce se zobrazí v místě překřížení červený bod. Pomocí tohoto algoritmu byl vyřešen problém s křížením ramen.

# 4 Zprostředkování softwarové komunikace

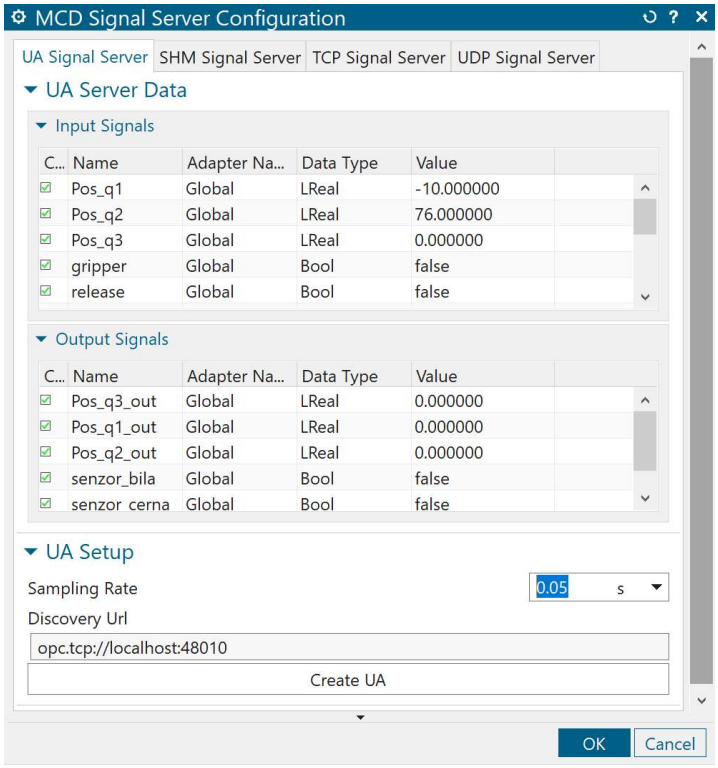
Při tvorbě této práce byly využito tří softwarů, kde každý plnil danou funkci. Prvním softwarem je NX Siemens, který sloužil k tvorbě 3D modelu SCARA robota a jeho nadstavba Mechatronics Concept Designer je použita k simulaci modelu. Druhým užitým softwarem je Matlab, který byl využit z důvodu složitějších výpočtů při tvorbě programu, protože nabízí velké množství různých výpočetních funkcí, což velmi usnadnilo práci. A posledním využitým softwarem byl Beckhoff TwinCAT, který sloužil při tvorbě programovatelného logického automatu, neboli PLC. Všechny tři softwary je třeba využívat současně, a to tak, aby spolu mohly navzájem komunikovat. K tomu bylo využito metody vytvoření OPC UA serveru.

### OPC UA server

Jedná se o v dnešní době velmi využívaný způsob komunice v průmyslových procesech, který umožňuje na výrobci nezávislou výměnu dat mezi zařízeními a systémy. Komunikační protokol OPC UA je novější verze protokolu OPC DA, která je oproti OPC DA nezávislý na platformě, protože je založený na obecně užívaných komunikačních standardech jako je TCP/IP, HTTP a SOAP. Podstata OPC UA je založená na komunikaci mezi OPC UA klientem a OPC UA serverem a podporuje mapování, navazování spojení, zabezpečení komunikace a strukturu zasílanych dat. [31]

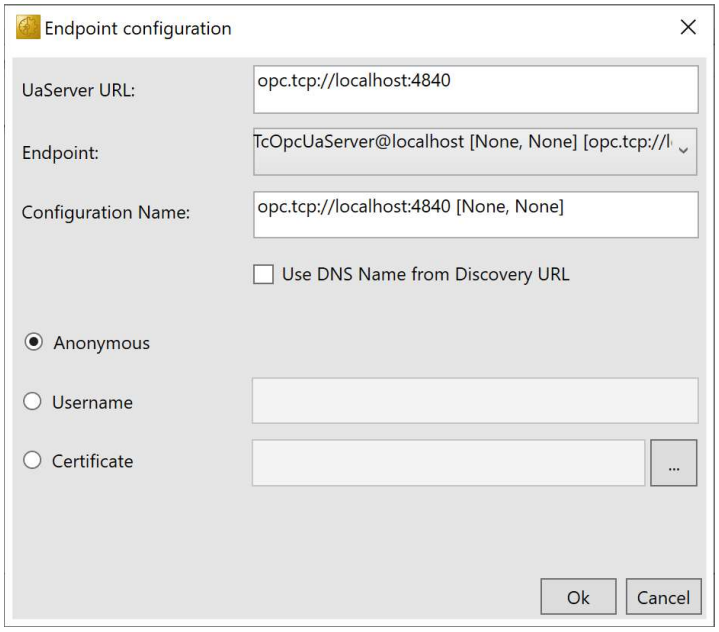
## 4.1 Vytvoření OPC UA serveru

Tou jednodušší cestou bylo vytvoření OPC UA serveru přímo v aplikaci NX Siements a následně k serveru připojit ostatní softwary. Tento postup se zdál být nejdříve úspěšný, avšak po krátké době přestal OPC UA server vytvořený v aplikaci NX fungovat. Proto musela být zvolena jiná cesta pro dosažení komunikace.




Obrázek 3.17: Vytvoření OPC UA serveru v NX

Pomocí nainstalované aplikace UA Expert bylo zkoušeno připojení k vytvořenému serveru, bohužel bez úspěchu. Po velkém množství pokusů o připojení byl tento postup zavržen a bylo třeba vytvořit OPC UA server poněkud složitější cestou a to v aplikaci pro tvorbu programovatelných automatů TwinCAT. K vytvoření OPC UA serveru v softwaru TwinCAT bylo zapotřebí nainstalovat toolbox, pomocí kterého bylo následně možné OPC server vytvořit. Při vytváření OPC serveru byla nastavena URL adresa serveru opc.tcp://localhost s portem 4840 a bezpečnost přístupu se nastavila na šifrovací typ None/None, při kterém nevznikaly žádné další komplikace.




Obrázek 3.18: Vytvoření OPC UA serveru v TwinCATu

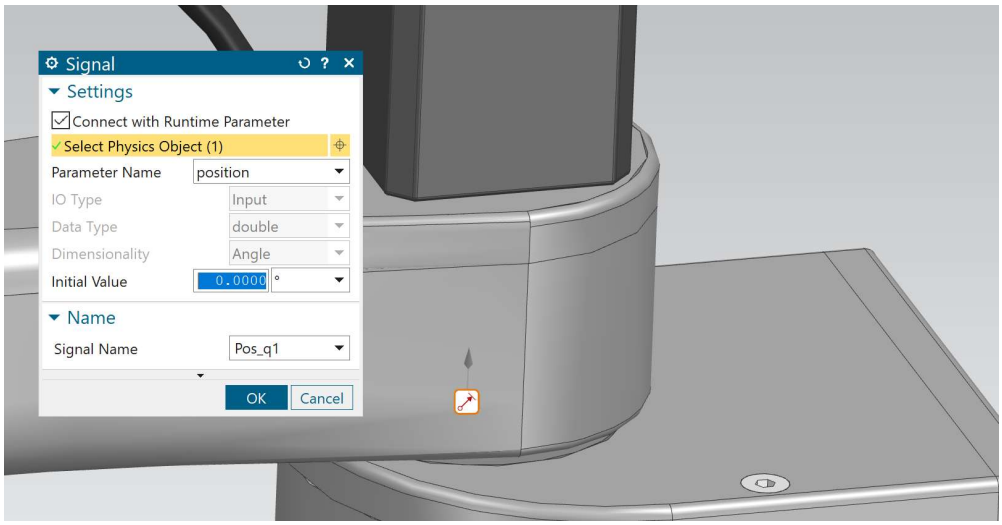
## 4.2 Mapování signálů

Při tvorbě programu v aplikaci TwinCAT bylo zapotřebí, aby TwinCAT komunikoval s oběma dalšími softwary a byla přenášena požadovaná data. Na následujícím obrázku je tedy možné vidět, jakým způsobem a jaká data byla namapována na výstup a na vstup. Při mapování proměnné na výstup se používá adresování *AT %Q** a pokud chceme namapovat proměnnou na vstup, použije se adresace *AT %I**. Dále je možné na obrázku vidět jaké proměnné vstupují do Matlabu, a které z Matlabu vystupují. To samé lze pozorovat i pro aplikaci NX.

```
// Matlab inputs
fp1     AT %Q*  : LREAL := 0;                              // pozice x
fp2     AT %Q*  : LREAL := 0;                              // pozice y
mStart  AT %Q*  : BOOL  := TRUE;                           // vypnutí výpočetu v Matlabu
nCycle  AT %Q*  : UDINT := 0;                              // cyklus
nPos    AT %Q*  : UDINT := 0;                              // pozice ramen
aPole   AT %Q*  : ARRAY [0..9] OF ARRAY [0..9] OF BOOL;    // matice obrazce
sImage  AT %Q*  : STRING(15);                              // jméno obrázku
bImg    AT %Q*  : BOOL := FALSE;                           // nahrát obrázek

// Matlab outputs
fq1     AT %I*  : LREAL := 0;                              // kloubová souřadnice 1
fq2     AT %I*  : LREAL := 0;                              // kloubová souřadnice 2
aDiff   AT %I*  : BOOL := 0;                               // změna matice
nColumn AT %I*  : INT;                                     // změněný sloupec
nRow    AT %I*  : INT;                                     // změněný řádek
aImgMat AT %I*  : ARRAY [0..9] OF ARRAY [0..9] OF BOOL;    // matice načteného obrazce

// NX inputs
fQq1    AT %Q*  : LREAL;                                   // zápis kloubové souřadnice 1
fQq2    AT %Q*  : LREAL;                                   // zápis kloubové souřadnice 2
fQq3    AT %Q*  : LREAL := 0;                              // zápis axiální pozice
gripper AT %Q*  : BOOL;                                    // zápis úchopu kuličky
release AT %Q*  : BOOL;                                    // zápis uvolnění kuličky

// NX outputs
fQq1_in      AT %I* : LREAL;                               // čtení kloubové souřadnice 1
fQq2_in      AT %I* : LREAL;                               // čtení kloubové souřadnice 2
fQq3_in      AT %I* : LREAL;                               // čtení axiální pozice
bsenzorCerna AT %I* : BOOL;                                // čtení senzoru černé kuličky
bsenzorBila  AT %I* : BOOL;                                // čtení senzoru bílé kuličky
```

Obrázek 3.19: Mapování signálů v TwinCATu

V softwaru NX bylo kvůli namapování nejdříve nutné vytvořit signály akčních členů, které měly být ovládány. Při tvorbě signálů bylo nutné nastavit runtime propojení, zda má být signál výstupem nebo vstupem a dále bylo nastavitelné upravení jednotek, kde byly pro rotaci nastaveny jednotky ve stupních a pro posuv jednotky v milimetrech, počáteční hodnotu a vybrat akční člen, do kterého budou hodnoty zapisovány. Poté byla nastavena URL adresa OPC UA serveru a obnovovací čas, jehož hodnota byla nastavena na 0,05 s.




Obrázek 3.20: Vytvoření signálu

Na obrázku č. 3.21 lze vidět namapování jednotlivých signálů, kde se v druhém sloupci nacházejí signály vytvořené v NX a ve čtvrtém sloupci pak proměnné vytvořené v TwinCATu. První sloupec zobrazuje jejich vzájemné propojení a třetí sloupec pomocí šipek ukazuje zda se jedná o vstup nebo výstup z NX.

| Connection Name | MCD Signal Name | Direction | External Signal Name | Owner Component |
|---|---|---|---|---|
| OPC UA.opc.tcp://localhost:4840 | | | | |
| Global_Pos_q1_fQq1 | Pos_q1 | ← | PLC1.GVL.fQq1 | |
| Global_Pos_q2_fQq2 | Pos_q2 | ← | PLC1.GVL.fQq2 | |
| Global_Pos_q3_fQq3 | Pos_q3 | ← | PLC1.GVL.fQq3 | |
| Global_gripper_gripper | gripper | ← | PLC1.GVL.gripper | |
| Global_release_release | release | ← | PLC1.GVL.release | |
| Global_Pos_q3_out_fQq3_in | Pos_q3_out | → | PLC1.GVL.fQq3_in | |
| Global_Pos_q1_out_fQq1_in | Pos_q1_out | → | PLC1.GVL.fQq1_in | |
| Global_Pos_q2_out_fQq2_in | Pos_q2_out | → | PLC1.GVL.fQq2_in | |

Obrázek 3.21: Mapování signálů v NX

V Matlabu nebylo připojení realizováno jako u ostatních programů, ale pomocí kódu. Aby bylo možné připojení k OPC UA serveru realizovat, bylo na začátku nutné nainstalovat OPC toolbox. Poté už mohl být napsán kód, kde bylo třeba do několika funkcí zapsat URL adresu OPC serveru, nastavit použitý typ zabezpečení a potom už jen připojit. U signálů bylo nutné signál nejdřit pomocí funkce *opcuanode* inicializovat, aby bylo poté možné u vstupních signálu přečíst hodnotu proměnné a výstupních hodnotu zapsat. Pro přečtení se použila funkce *readValue* a pro zápis *writeValue*. Na obrázku č. 3.22 je pouze ukázka mapování signálu. Vstupních i výstupních proměnných bylo použito více a je možné je najít v obrázku č. 3.19.

```
uaClient = opc.ua.Client('opc.tcp://localhost:4840');       % nastavení OPC UA serveru
setSecurityModel(uaClient,'None');                          % nastavení zabezpečení
connect(uaClient);                                          % připojení k OPC UA serveru

fp1 = opcuanode(4,'GVL.fp1');                               % inicializace vstupní proměnné
fp1 = readValue(uaClient,fp1);                              % přečtení hodnoty proměnné
fq1 = opcuanode(4,'GVL.fq1');                               % inicializace výstupní proměnné
writeValue(uaClient, fq1, q1);                              % zápis hodnoty proměnné
```

Obrázek 3.22: Mapování signálů v Matlabu

# 5 Virtuální zprovoznění

V této fázi práce je tedy k dispozici vytvořený model robota v aplikaci NX, napsaný algoritmus pro výpočet inverzní kinematiky s hlídáním správné rotace druhého ramene a zajištěná komunikace mezi softwary. Jelikož jsou splněny všechny předpoklady pro tvorbu programu, přichází na řadu jeho psaní v aplikaci TwinCAT.

TwinCAT je software určený k vytváření programovatelných stavových automatů. Použití tohoto typu programování je právě velmi vhodné pro různé funkce robotů a automatizací. Společnost Beckhoff také nabízí velké množství nejrůznějších toolboxů, které slouží např. právě k různým druhům propojení nebo usnadnění práce s programováním robotů. Jeden z nabízených toolboxů je např. i motion toolbox, který počítá kinematické transformace pro velké množství kinematických typů. K výpočtu inverzní kinematiky v této práci měla právě tato nadstavba původně posloužit. Při použití této nadstavby, je pro výpočet kinematických transformací nejdříve nutné nadefinovat osy souřadného systému stroje MCS a osy souřadného systému aktuátorů ACS. Dále se pak nastaví všechny potřebné parametry a funkce *FB_KinConfigGroup* následně vypočítá kinematické transformace.


Obrázek 3.23: Funkční blok pro výpočet kinematiky [32]

Z důvodu zajímavosti a vyzkoušení propojení komunikace s Matlabem, který nabízí velké množství výpočetních funkcí a simulací, bylo toto řešení výpočtu kinematických transformaci po domluvě rozhodnuto ponechat stranou.

## 5.1 Základní algoritmus pro tvorbu obrazců

Dále bude v práci popsáno, jak a jakým způsobem byl vytvořen program pro danou funkci, jak funguje a jaké nabízí možnosti. Jak již bylo zmíněno, pro ovládání robotů je v dnešní době PLC běžně využívanou metodou. Z tohoto důvodu je modelovaný robot řízen právě pomocí logického stavového automatu.

### Proměnné

Nejdříve bych se zastavil u vytváření proměnných. Velká část proměnných, které program využívá se nachází v tzv. listu globálních proměnných. Ty jsou v něm nadefinované

s příslušnými datovými typy a slouží k usnadnění práce v rámci celého projektu. Tyto proměnné je možné vidět například na obrázku 3.19, a jsou používány právě pro komunikaci mezi softwary pomocí OPC UA serveru. Dále se v tomto listu globálních proměnných nachází i známé pozice kartézského souřadného systému, které jsou využívány k výpočtu kloubových souřadnic pomocí inverzní úlohy. A součástí tohoto listu jsou i proměnné, které jsou namapovány v HMI. Takto vytvořené proměnné už poté není nutné ve funkcích definovat, ale je potřeba kromě jejich názvu zapsat i jejich cestu. Na příkladu kloubové souřadnice to pak vypadá takto: *gvl.fq1*. Pokud se nadefinují proměnné v dané funkci, stačí při jejich použití napsat pouze jejich název.

### Timer

V napsaném programu je možné narazit na tzv. timery, které počítají čas. Timery nabízejí více možností nastavení, podle toho, co je zrovna potřeba počítat. V tomto programu je potřeba sledovat čas, pro vyčkání modelu v dané pozici a po uběhnutí definovaného času pokračovat v pohybu. Příkladem je třeba podtlaková přísavka, pomocí které je možné uchopení kuliček realizovat. Je nutné, aby koncový efektor zůstal určitou dobu v pozici úchopu kuličky. V opačném případě by k úchopu kuličky nedošlo. Timery jsou nadefinovány v listu proměnných a dále pomocí přípony *.PT* definujeme čas, který musí uplynout, aby byla vytvořena náběžná hrana, která je zjišťována pomocí koncovky *.Q*. Timer je mimo přísavky použit i pro výpočet inverzní kinematiky při startu a nebo k nahrání obrázku, které je popsáno v kapitole 3.11. Byly tedy popsány proměnné a timery, a je možné přejít na vytvořené algoritmy.

### Rozhodovací algoritmus mezi bílou a černou kuličkou

Nejdříve bylo nutné zamyslet se nad způsobem, jakým bude řešeno zjišťování jednotlivých stavů v obrazcovém poli. Uživatel totiž zadává, jak má výsledný obrazec vypadat, a v závislosti na tomto zadaní bude robot skládat kuličky. Protože má výsledný obrazec velikost rozlišení 10x10 pixelů, nabízí se vhodné řešení pomocí matice o dvou stavech. Jelikož se v modelu nacházejí pouze bílé a černé kuličky a jedná se tedy o vytváření binárního obrazce, budou jednotlivé stavy v matici 0 nebo 1. To znamená že se nadefinovala matice o rozměrech 10x10 datového typu BOOL: *abarvy AT %Q* : ARRAY [0..9] OF ARRAY [0..9] OF BOOL;*, která je mimo jiné, kvůli dále popsané funkci, i vstupní proměnou v Matlabu. Na základě této úvahy se vytvořil rozhodovací algoritmus mezi bílou a černou kuličkou, který je možné vidět na následujícím obrázku.

```
IF gvl.abarvy[matRadek][matSloupec] = 1 THEN
  gvl.fp1 := gvl.fcernaSebratX;  // zápis souřadnice X do Matlabu
  gvl.fp2 := gvl.fcernaSebratY;  // zápis souřadnice Y do Matlabu
  bBarva := TRUE;                // zápis barvy
ELSE
  gvl.fp1 := gvl.fbilaSebratX;
  gvl.fp2 := gvl.fbilaSebratY;
  bBarva := FALSE;
END_IF
```

Obrázek 3.24: Rozhodovací algoritmus mezi bílou a černou kuličkou

### Přesun ramen

V této kapitole bych se rád pozastavil u výpočtu kloubových souřadnic pomocí inverzní úlohy v Matlabu. Bylo nutné upravit program v Matlabu takovým způsobem, aby při změnách pozic

kartézského systému, vypočítal nové natočení kloubů. Základem této úpravy je *while* smyčka, která běží tak dlouho, dokud nebude výpočet vypnut. Do této while smyčky bylo potřeba vhodně umístit rozhodovací algoritmus, který rozhoduje, zda došlo ke změně pozic. K tomu bylo potřeba uložení již vypočítaných pozic. Zda nastal menší problém, protože je nezbytné měnit kloubové souřadnice i při změně polohy pouze jedné z kartézských souřadnic. Při zápisu nových pozic v TwinCATu, který probíhá postupně, však docházelo v některých případech k výpočtu kloubových souřadnic pouze pro jednu změněnou kartézskou souřadnici. Proto bylo potřeba tuto skutečnost ošetřit. Z důvodu již napsaného většího množství programu byl tento problém ošetřen méně vhodným řešením, a to, že při změně některé z pozic program v Matlabu na chvíli počká, a poté si hodnoty opět ověří. Vhodnější řešení by určitě bylo zapsat hodnoty do matice a tu nastavit jako vstup v Matlabu. Každopádně bylo ověřeno, že tento způsob ošetření nemá na simulaci negativní vliv.

Výpočet kloubových souřadnic tedy probíhá tím způsobem, že jakmile přepíšeme pozice kartézských souřadnic *fp1* a *fp2*, vypočte Matlab kloubové souřadnice a zapíše je do proměnných *fq1 fq2*. A pokud přichází na řadu pohyb s rameny, jednoduše se zapíší hodnoty vypočtených kloubových souřadnic do proměnných *fQq1 a fQq2*, které jsou vstupními signály v NX MCD.

```
gvl.fQq1 := gvl.fq1;  // zápis 1. kloubové souřadnice do NX
gvl.fQq2 := gvl.fq2;  // zápis 2. kloubové souřadnice do NX
```

Obrázek 3.25: Zápis kloubových souřadnic do NX

### Posuv výsuvné hřídele s přísavkou

Aby nedocházelo k různým kolizím s kuličkami je výsun a zásun hřídele realizován ve chvíli, kdy ramena dorazí do žádané pozice. K tomu slouží výstupní signály z NX MCD, kterými jsou senzory natočení ramen a výsunu hřídele. Tyto signály vstupují do projektu v TwinCATu a je pomocí nich možné trasovat natočení ramen. Pokud se žádaná hodnota rovná s hodnotou aktuálního natočení kloubů může být výsuvná hřídel vysunuta či zasunuta. Jelikož vstupní hodnoty ze senzorů natočení jsou nadefinovány datovým typem *LREAL,* je potřeba je z důvodu porovnávání zaokrouhlit. Její posuv je pak realizován stejným způsobem jako pohyb ramen.

```
gvl.fQq3 := gvl.fpneuPoloz;          // zápis pozice výsunu do NX

fpos3 := LREAL_TO_INT(gvl.fQq3_in);// zaokrouhlení na jednotky

IF fpos3 = gvl.fQq3 THEN
  nState := 180;
END_IF
```

Obrázek 3.26: Ovládání výsuvné hřídele

### Ovládání podtlakové přísavky

V této fázi je vysvětleno, jakým způsobem program zjistí, kterou kuličku uchopit, jak je realizován pohyb ramen a následně vysunutí hřídele. Na řadu tedy přichází úchop kuličky. Ten je ovládán pomocí výstupního signálu datového typu *BOOL*. Pokud má být proveden úchop, je potřeba nastavit u výstupní proměnné *gripper* hodnotu 1 a u proměnné *release* hodnotu 0. Pokud chceme kuličku pustit, nastaví se hodnoty obráceně. Tento signál vstupuje do NX MCD

a provádí úchop či puštění. Jak bylo zmíněno v kapitole s timery, je potřeba, aby přísavka zůstala určitou dobu dané pozici a úchop nebo puštění kuličky bylo realizováno. V případě úchopu by kulička zůstala na své pozici a v případě puštění by byla kulička puštěna na nesprávném místě.

```
TON_grip(IN := TRUE);     // spuštění timeru
gvl.gripper := TRUE;      // zapnutí úchopu
gvl.release := FALSE;     // vypnutí uvolnění

IF TON_grip.Q THEN        // po uplynutí času
  nState := 190;
  TON_grip(IN := FALSE); // reset timeru
END_IF
```

Obrázek 3.27: Ovládání podtlakové přísavky

### Přepočet pozic

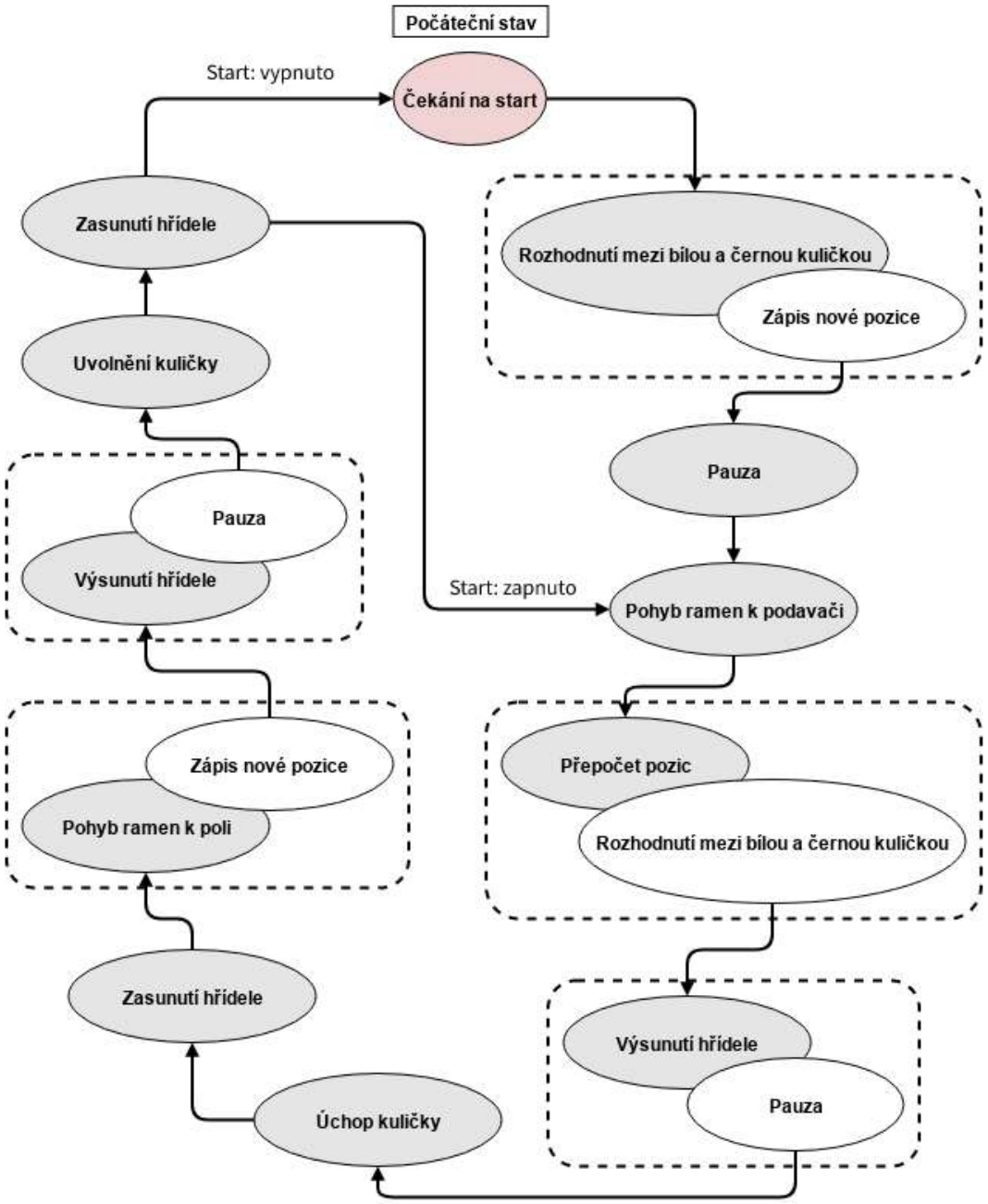
Po úchopu či puštění kuličky dojde k zásunu hřídele a ramena se mohou přesunout na další pozici. Aby mohlo přesunu dojít, bylo potřeba přepočítat pozice kartézských souřadnic. Tento přepočet je nejvhodnější provést co nejdříve po zápisu vypočtených kloubových souřadnic do NX MCD, protože numerické řešení inverzní kinematiky je časově náročnější, a je potřeba, aby Matlab tento výpočet včas provedl a výsledek zapsal do proměnných v TwinCATu. Přepočet kartézských souřadnic se provádí na základě následujícího řádku a sloupce v matici stavů *abarvy*. Manipulátor je totiž naprogramován tak, aby skládal jednotlivé kuličky po řádcích. To znamená, že se začíná v prvním řádku a sloupci (v kódu [0,0]) a pokračuje na druhý sloupec. K souřadnici *x* se pak připočte hodnota 0,02, což je rozdíl vzdáleností mezi vedlejšími středy otvorů v obrazcovém poli, určených pro položení kuliček. Pokud je hodnota sloupce vyšší než 10, přejde se na druhý řádek a první sloupec. V závislosti na tom je pak do souřadnice x zapsána výchozí pozice a k souřadnici *y* se připočítá hodnota 0,02.

```
IF matSloupec = 10 THEN
  gvl.fstartPosX := gvl.fstartNewPosX;     // zápis výchozí pozice x
  matSloupec := 0;                         // úprava sloupce
  matRadek := matRadek + 1;                // úprava řádku
  gvl.fstartPosY := gvl.fstartPosY + 0.02; // úprava pozice y
ELSE
  gvl.fstartPosX := gvl.fstartPosX + 0.02; // úprava pozice x
END_IF

END_IF
```

Obrázek 3.28: Přepočet matice a kartézských souřadnic

### Stavový automat

V předchozích odstavcích byly popsány hlavní části kódu a myšlenek při vytváření základního stavového automatu. Pod pojmem stavový automat je možné si představit přehledný programovací model, u kterého se jednotlivé činnosti rozdělí do jednotlivých stavů, ve kterých se může nacházet, a po splnění určitých podmínek mezi nimi může přecházet. Pro znázornění bylo vytvořeno schéma použitého stavového automatu, které je možné vidět na obrázku č. 3.29. V tomto případě je jedná pouze o základní schéma, které nenabízí žádné další funkce.




Obrázek 3.29: Diagram základního stavového automatu

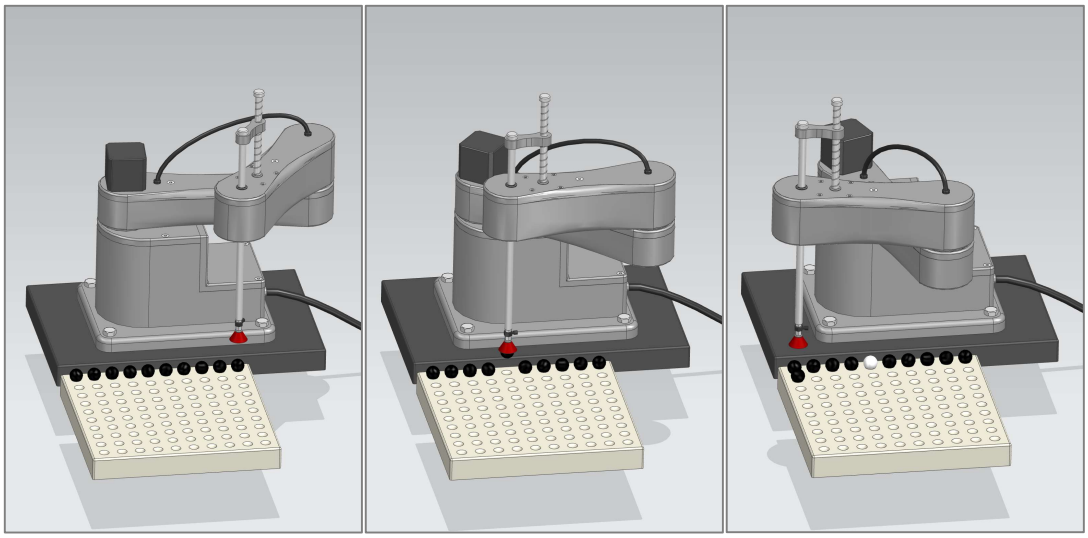
## 5.2 Rozšíření algoritmu

Výše zmíněný algoritmus nabízí pouze základní funkci, kterou je skládání kuliček na připravené pole, a tím se vytváří binární obrazec. Při tvorbě se však nabízely další možnosti, jak model upravit, aby nabízel více funkčních možností pro úpravu obrazce. Mezi rozšířené funkce patří výměna kuliček, odebírání kuliček, nahrání vlastního obrazce, a nakonec i funkce pro zastavení modelu robota. Všechny tyto funkce mohou být užitečnými, a proto byly do algoritmu přidány.

### Výměna kuliček

První funkcí, o kterou byl rozšířen základní algoritmus byla výměna kuliček. Může se stát, že uživatel bude z různých důvodů požadovat drobnou změnu obrazce, ale klidně i přeskládání obrazce celého. Z tohoto důvodu byla tato funkce přidána a její řešení je následovné.

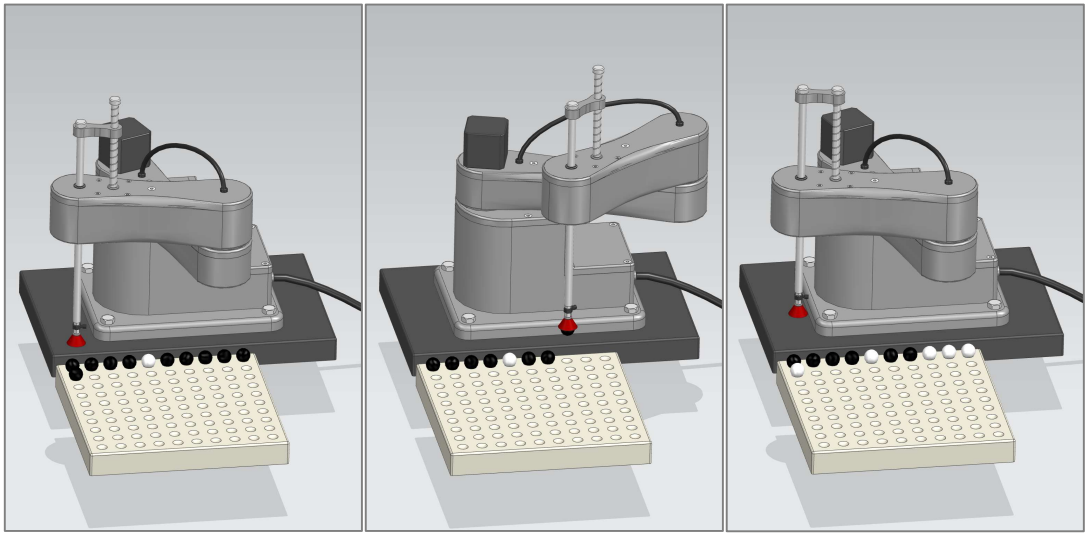
Nejdříve bylo nutné zařídit, aby program rozpoznal, že došlo ke změně. To je možné provést porovnáním matic zadávaného obrazce a obrazce vytvořeného. Ovšem bylo zjištěno, že v softwaru TwinCAT porovnávání matic není možné a bylo by tedy nutné pomocí *for* smyčky porovnávat v každém cyklu všechny stavy matic. Toto řešení by však zvyšovalo výpočetní čas, a proto přišlo v úvahu nahrávat matice do Matlabu a porovnávat je v něm. Matlab se pro podobné výpočty hodí více, a tak mohlo být zároveň v Matlabu vytvořeno i hledání změněného řádku a sloupce pomocí *for* smyčky. Výsledkem tohoto výpočtu je pak hodnota 1 nebo 0 typu *BOOL*, zda došlo ke změně a zároveň hodnota řádku a sloupce změny. U hodnoty řádku a sloupce se nabízely dvě možnosti, kterými jsou výměna kuliček od konce, nebo od začátku obrazcového pole. Jelikož ani jedna možnost nezaručí vždy rychlejší provedení, byl vybrán zápis řádku a sloupce nejblíže výchozí pozici. Následně jsou pomocí hodnot řádku a sloupce změny vypočítány kartézské souřadnice, zapíše se hodnota barvy kuličky a automat přejde do stavů pro výměnu kuliček. Program poté přechází mezi základním algoritmem a algoritmem rozšířeným o výměnu kuliček tak dlouho, dokud nejsou vyměněny všechny požadované kuličky. Až poté se začne pokračovat ve skládání.



Obrázek 3.30: Provedení výměny kuliček

### Odebírání kuliček

Druhou možností, jak upravit program a tím rozšířit možnosti tvorby obrazce, bylo zavedení funkce pro odebíraní kuliček. Pokud uživatel bude požadovat vyprázdnění pole poslouží k tomu právě tato funkce. Samozřejmě je také možné odebrat pouze určité množství kuliček. V prvním kroku bylo nutné vyřešit samotné odebírání a v druhém kroku pak vyřešit kolidování s výměnou kuliček neboli aby se ukončila funkce výměny a nastalo odebírání. Stavový automat plní základní funkci, a tedy skládání kuliček na obrazcové pole do té doby, dokud se nespustí funkce vyskládání pole. Poté přejde automat do stavů pro odběr kuliček. Nejdříve je potřeba přepočítat kartézské souřadnice a řádek a sloupec obrazcové matice. Dále mohou být zahájeny přesuny ramen, posun výsuvné hřídele a ovládání podtlakové přísavky. Odebírání kuliček je realizováno ve směru od poslední položené kuličky směrem k první, v opačném případě by to nedávalo smysl, zvláště pokud má být odebráno pouze posledních pár kuliček. Tato funkce je prováděna tak dlouho, dokud je v uživatelském prostředí zapnuto vyskládání pole a po vypnutí opět začne skládání. V případě, že nedojde k vypnutí funkce vyskládání, budou odebrány veškeré kuličky z pole, ramena skončí ve výchozí pozici, vypne se funkce vyskládání pole a automat bude čekat na opětovný start. Po rozchození této funkce bylo potřeba vyřešit kolidování s výměnou kuliček. Pokud uživatel změní část obrazce dojde k výměně kuliček. Avšak po zapnutí vyskládání pole uživatelem není chtěno, aby dále probíhala výměna, ale aby se odstartovalo odebírání. To bylo potřeba ošetřit pomocí dalších stavů a přepočtů. Na obrázku č. 3.31 je možné vidět odebrání čtyř černých kuliček, které byly později nahrazeny bílými.



Obrázek 3.31: Provedení odebírání kuliček

### Nahrání obrázku

Další možností, jak usnadnit práci je nahrání libovolného obrázku do uživatelského prostředí. To slouží k tomu, aby uživatel nemusel pokaždé zdlouhavě zadávat obrazec, který má být vytvořen, zvláště pokud má k dispozici určitou předlohu. K této funkci bylo opět využito možností Matlabu. Do Matlabu lze totiž poměrně jednoduše nahrát obrázek a ten dále upravovat. Pokud je k dispozici obrázek ve formátu, který lze bez problému nahrát do Matlabu (v tomto případě obrázek VUT loga), je potřeba mu nejdříve upravit rozměry. Protože se má vytvořit obrazec o rozměrech 10x10 pixelů, bude i nahraný obrázek upraven na tuto velikost pomocí funkce *resizedImage*. Tím je získán barevný obrázek o změněných rozměrech. Protože

je však nutné získat pro zápis obrázku do TwinCATu matici o stavech 0 nebo 1, musí být obrázek v Matlabu převeden na binární obrazec neboli matici o hodnotách 0 nebo 1. K tomu bylo využito *for* smyčky, díky které je do jednotlivých pozic matice zapsána hodnota 0 nebo 1. Protože jsou data barevného obrázku trojrozměrná, kde první dva rozměry udávají rozlišení a třetí rozměr barvu, bylo nutné vytvořit rozhodovací algoritmus, který přiřadí dané hodnotě barev 0 nebo 1. Také se může stát, že uživatel chce nahrát černobílý obrázek, který má pouze dva rozměry. Pro tento případ musel být rozhodovací algoritmus rozšířen a je možné ho vidět na následujícím obrázku.

```
for pix_c = 1:10                % sloupec
    for pix_r = 1:10            % řádek
        if sizeSize == [1,3]    % barevný obrázek
            % porovnání jednotlivých hodnot RGB
            if resizedImage(pix_r,pix_c,1) > 135 && ...
               resizedImage(pix_r,pix_c,2) > 128 || ...
               resizedImage(pix_r,pix_c,3) > 150
                imMat(i) = 1;   % zápis hodnoty 1 do matice
                i = i+1;        % posun pozice v matici
            else
                imMat(i) = 0;
                i = i+1;
            end
        else                    % černobílý obrázek
            % porovnání hodnot
            if resizedImage(pix_r,pix_c,1) > 120
                imMat(i) = 1;
                i = i+1;
            else
                imMat(i) = 0;
                i = i+1;
            end
        end
    end
end
```

Obrázek 3.32: Tvorba černobílého obrázku

Výsledek této úpravy je možné vidět na obrázku č. 3.33. Výsledek se může zdát mírně nepřesný, což je způsobeno úpravou barevného obrázku pro získání vhodných rozměrů. Rozměry původního obrázku jsou totiž obdélníkového tvaru (554x810 pixelů). Výsledný obrázek má ale čtvercové rozměry (10x10 pixelů).

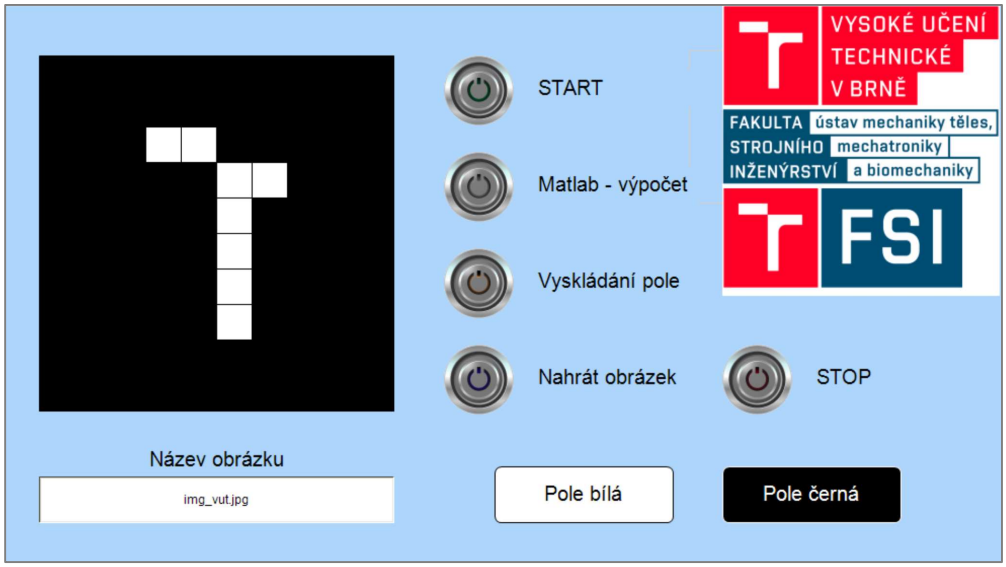
Obrázek 3.33: Porovnání předlohy a vytvořeného obrázku [33]

### Zastavení manipulátoru

Posledním rozšířením stavového automatu bylo přidání funkce pro zastavení robota. Pokud je vypnuto tlačítko startu, dokončí se právě rozpracovaný cyklus, ale robot není zastaven. Např. v případě nehody nebo kolize je nezbytné, aby došlo k co nejrychlejšímu zastavení. Proto byla do programu přidána i tato možnost. Největším a zároveň neřešitelným problémem je při použití této funkce časová prodleva mezi vyslaným signálem z TwinCATu a přijatým signálem v NX. Vzhledem k výpočetnímu výkonu byl obnovovací čas vstupních a výstupních signálu v NX nastaven na 0,05 s. Z tohoto důvodu nebylo možné realizovat zastavení manipulátoru pomocí pozice, protože signál o pozici byl dva krát zpožděný. Nejdříve při vstupu do TwinCATu, a poté při vstupu do NX. Proto bylo zastavení modelu řešeno způsobem nastavení nulové rychlosti aktuátorů. I zde se projevuje časová odchylka, každopádně manipulátor je zastaven a nevrací se zpět na pozici, ve které se už nacházel. Při tomto způsobu řešení je však pomocí simulace vidět, že při předepsání nulové rychlosti aktuátorům dojde k mírnému rozhození ramen. Při pokusu o mírnější zastavení pomocí postupného snižování rychlosti se kvůli časové odezvě také nedostalo lepším výsledkům, a proto byl ponechán původní kód. Tato funkce však slouží pouze k nucenému zastavení, ke kterému by mohlo dojít např. při ohrožení na zdraví.

## 5.3 HMI

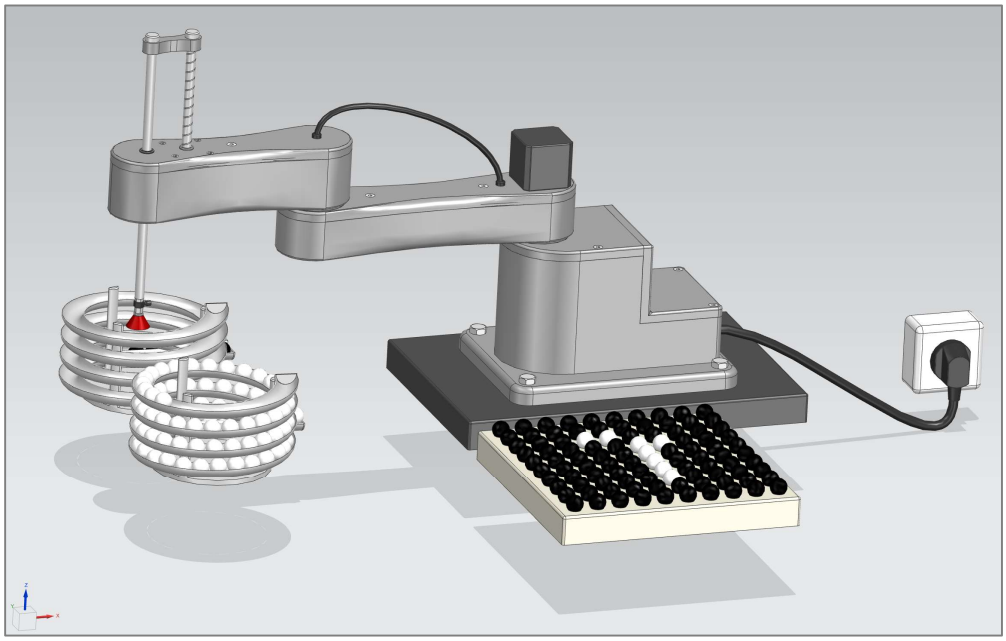
HMI je rozhraní mezi člověkem a strojem a umožňuje řízení různých automatizačních zařízení a zobrazení a předání informací o stavu ovládaného zařízení. Společně s PLC pak tvoří systémy OPLC. Protože je nutností v této práci řídit zařízení uživatelsky vhodným řešením, bylo pro tuto funkci vybráno HMI. To bylo v této práci vytvořeno ve vizualizačním okně, které nenabízí takové prostředky pro tvorbu HMI jako samotná nadstavba HMI, avšak pro tuto práci bylo dostačující. Nejdříve bylo nutné vytvořit pole pro zadání obrazce, které se skládá ze sta segmentů o stavech 0 nebo 1, které jsou zapisovány do obrazcové matice, a kde je každý segment namapován na příslušný řádek a sloupec této matice. Dále byly přidány přepínače pro odstartování modelu a také vypnutí výpočtu v Matlabu. Pro usnadnění zadávání obrazce byla vytvořena tlačítka *Pole bílá* a *Pole černá*, která převedou veškeré segmenty do stejného stavu (0 nebo 1). Nakonec byly přidány přepínače, pro funkce rozšířeného automatu. Těmi jsou *Vyskládání pole*, *Nahrát obrázek* a *STOP*. Součástí nahrání obrázku je pak i textové pole *Název obrázku*, do kterého je potřeba zapsat název obrázku i s formátem, který má být nahrán. Toto vytvořené HMI se nachází na obrázku č. 3.34.




Obrázek 3.34: Vytvořené HMI pro ovládání modelu

## 5.4 Výsledný obrazec

Na obrázku č. 3.35 lze vidět model robota i s vytvořeným obrazcem, podle předlohy nahraného obrázku VUT loga. Vzhledem k výpočetnímu výkonu použitého hardwaru, trvala simulace 25 minut a 53 sekund, což by neodpovídalo reálnému modelu manipulátoru. Pomocí kódu v Matlabu byl vypočten čas potřebný k přesunům ramen, který vyšel 172 sekund. Dále byl vypočten čas pro výsuv hřídele, vycházející 336 sekund, a nakonec se připočetlo 100 sekund pro ovládání hřídele. Celkový čas reálného modelu pro vytvoření obrazce na obrázku 3.35 tedy po sečtení vychází 10 minut a 8 sekund. Tento výsledný čas se však liší v závislosti na vytvářeném obrazci.



Obrázek 3.35: Vytvoření výsledného obrazce

# 6 Zhodnocení rizik spojených s provozem

Jelikož se tato práce zabývá virtuálním zprovozněním víceosého manipulátoru, je prakticky zbytečné přidávat do modelu určité nástroje pro snížení rizik spojených s provozem zařízení. Protože je však zhodnocení rizik jedním z cílů práce, budou v této kapitole popsána rizika a jejich zamezení, které by bylo třeba řešit u reálného modelu robota.

Při návrhu a konstrukci elektrických strojů je nutné dodržovat určité zásady pro zajištění bezpečnosti tak, že strojní zařízení musí být navrženo a konstruováno tak, aby při provozu ani údržbě nedocházelo k vystavení riziku osob. K opatřením před rizikem pak slouží různé normy a směrnice, podle kterých se musí výrobce zařízení řídit. U všech probíhajících procesů dochází k rizikům, která mohou být zanedbatelná až nepřijatelná. Aby se předešlo úrazu, je potřeba rizika analyzovat a snižovat jejich úroveň, nebo je úplně odstraňovat. Nejčastějším zdrojem úrazů je podle analýz nedostatečné odhadnutí rizika, což může mít v mnoha případech i špatný vliv na psychiku pracovníků. K takovému riziku dochází např. při mechanickém pohybu částí strojního zařízení nebo při práci s elektrickým zařízením, kde může dojít k úrazu elektrickým proudem nebo požárem z důvodu poškození nebo závady na elektrickém zařízení stroje. Jedním z dalších možných rizik, se kterým je dobré počítat, je i vliv externích vlivů nebo kybernetický útok. Kybernetické útoky jsou stále častějšími hrozbami podniků, kde způsobují především finanční ztráty, každopádně není vyloučeno, že napadené zařízení nebo jeho řídicí systém nemůže ovlivnit bezpečnost při práci.

Všechna možná rizika spojená s konstrukcí strojního zařízení je potřeba identifikovat (pro celý životní cyklus) a vytipovat hlavní nebezpečné prostory stroje. Poté se odhadne velikost rizika jak před přijetím preventivního opatření, tak po přijetí. Odhadnutá rizika se zhodnotí, zda splňují určitou míru akceptovatelnosti, kterou je předem stanovená hranice přijatelnosti rizika. Pokud je riziko akceptovatelné, přejde se k dalšímu významnému nebezpečí, a pokud není akceptovatelné, provede se návrh opatření, které riziko sníží. [34] Dále budou popsána rizika, která by mohla vést k úrazu při použití reálného modelovaného zařízení.

### Riziko úrazu elektrickým proudem

Protože zařízení by bylo elektrickým strojem, kde je hlavním zdrojem elektrická energie, bylo by nutné provést opatření k snížení rizika úrazu elektrickým proudem. Hlavní součástem zařízení byl předepsán materiál hliník, který je vodivým kovem, a proto první nabízející se ochranou by byla ochrana pospojováním neboli uzemněním neživých částí, v tomto případě konstrukce. Další ochranou proti riziku úrazu elektrickým proudem by měla být vhodná izolace a v případě potřeby i použití krytí, které by v případě činnosti stroje v okolí vodních zdrojů mělo být dimenzováno proti vniknutí vody. Zařízení by mělo obsahovat všechny potřebné upozornění a správně by mělo být i revidováno.

### Riziko úrazu pohybem ramen

Při integraci robotů je potřeba důsledně zvážit rizika úrazu způsobená jejich pohybem. Je nutné vzít v potaz jejich rychlost, dosah, použité materiály při výrobě nebo způsob obsluhy. U rizikovějších zařízení, kde nedochází ke sdílení pracovního prostoru robota s člověkem se pro ochranu před nebezpečím používá např. ochranná klec, která má zamezit vniknutí těles nebo částí lidského těla do pracovního prostoru stroje. Pokud má být pracovní prostor robota sdílen se člověkem, je potřeba použít jiné typy ochran, jako jsou ruční ovládání, monitorované zastavení nebo omezení síly a napájení. [35] Navrhnutý model zařízení je poměrně malý a poměrně lehký, což velmi snižuje riziko tímto úrazem. Protože však manipulátor nemusí sdílet pracovní prostor s člověkem, použila by se pro zajištění bezpečnosti např. ochranná konstrukce.

# 7 Zhodnocení výsledků a doporučení pro praxi

V předchozích kapitolách byly popsány veškeré kroky, kterých bylo nutné dosáhnout pro vytvoření funkčního modelu víceosého manipulátoru pro aplikaci tvorby obrazce. V průmyslu je virtualizace zařízení běžně užívanou metodou pro odhalení chyb a problému před samotnou instalací. Ověřování pomocí virtualizace bylo prováděno takovým způsobem, aby byly případně odhaleny všechny nedostatky, což znamená, že byl model manipulátoru testován pro všechny možně dostupné funkce. Samotný algoritmus pro skládání kuliček nebylo složité ověřit, protože nemohlo docházet k více kolizím. Po přidání rozšiřujících funkcí muselo být ale provedeno testování pro vícero možných vzniků chyby. Mezi ty patří například křížení funkcí, kterými jsou: skládání kuliček, výměna kuliček a odebírání kuliček. Spouštěním těchto funkcí v různých okamžicích a v různých pozicích byly odhaleny nedostatky řízení, které byly opraveny. Při výsledné simulaci byla ověřena správnost řízení, které je možné nahrát do řídicí jednotky reálného manipulátoru. Před uvedením manipulátoru do praxe, by však bylo nejdříve nutné provést úpravu komponent a proměnných v programu.

# Závěr

Tato diplomová práce se zabývá virtuálním zprovozněním víceosého manipulátoru, který pomocí kulových segmentů vytváří binární obrazce. Nejdříve bylo potřeba zvolit vhodný typ kinematiky, který bude schopný dobře vykonávat danou funkci. Protože k této funkci postačí pouze 3-osý manipulátor, rozhodovalo se mezi kartézským typem kinematiky, typem delta a SCARA. Po multikriteriálním zhodnocení vyšel jako nejvhodnější typ kinematiky SCARA. Tento typ kinematiky byl dále navržen a vymodelován v 3D CAD modeláři NX siemens a v jeho nadstavbě MCD pak byly přidány vazební a aktivační členy. Součástí modelu je také i návrh a dimenzování krokových motorů.

Dále bylo nutné vyřešit výpočet inverzní kinematiky, protože jsou znány kartézské souřadnice a pro pohyb ramen je potřebné znát kloubové souřadnice. Nejdříve měl být výpočet inverzní kinematiky prováděn v softwaru TwinCAT, který byl použit pro vytvoření PLC. Protože kód pro výpočet inverzní kinematiky jsem chtěl vytvořit sám a zároveň vyzkoušet komunikaci Matlabu s TwinCATem pomocí OPC UA serveru, byl vytvořen skript pro výpočet inverzní kinematiky v Matlabu. Bylo vyzkoušeno, jak analytické, tak numerické řešení inverzní úlohy, kde především z důvodu zajímavosti a četnosti užití v praxi bylo použito numerické řešení.

Aby bylo možné vytvořit stavový automat v softwaru TwinCAT, musela být zprostředkována komunikace mezi použitými softwary. K této komunikaci bylo využito OPC UA serveru, který byl vytvořen pomocí nadstavby v TwinCATu a NX i Matlab se k tomuto serveru připojili. Pro předávání dat musely být na vstup či výstup namapovány jednotlivé signály, které v NX zprostředkovávají pohyb a řízení manipulátoru, v Matlabu především výpočet kloubových souřadnic a TwinCAT veškeré úkony řídí.

Po zprostředkování komunikace bylo dalším krokem vytvoření stavového automatu v softwaru TwinCAT, kde byl vytvořen základní algoritmus pro skládání kuliček a vytváření obrazce. Tento základní algoritmus byl dále rozšířen o další funkce, jako je výměna kuliček, odebírání kuliček, zastavení manipulátoru a nahrání předlohy obrazce. Pro ovládání modelu pak bylo v TwinCATu ve vizualizační časti vytvořeno HMI.

Ve finální fázi bylo prováděno testování řízení, zda program funguje správně a nedochází ke kolidování různých funkcí. Ověření se provádělo zapínáním a vypínáním různých kombinací funkcí, a to v různých pozicích, ve kterých se manipulátor nacházel.

Výsledkem této práce je tedy funkční a jednoduše ovladatelný model manipulátoru, který slouží k vytváření binárních obrazců. Virtuální zprovoznění manipulátoru bylo otestováno a kinematika typu SCARA se projevila jako vhodné řešení. Proto je možné práci využít pro řízení reálného modelu manipulátoru v praxi, což bylo hlavní myšlenkou této diplomové práce.

# Seznam použité literatury


[1] Co se skrývá pod označením PLC ? | Automatizace.HW.cz. Automatizace.HW.cz | Elektronika v automatizaci [online]. Dostupné z: https://automatizace.hw.cz/co-se-skryva-pod-oznacenim-plc

[2] Multifunkční řízení v jediném modulu PLC [online]. 2016 [cit. 2023-05-22]. Dostupné z: https://www.volty.cz/2016/10/03/multifunkcni-rizeni-v-jedinem-modulu-plc/

[3] ScienceDirect. ScienceDirect [online]. Copyright © [cit. 21.05.2023]. Dostupné z: https://www.sciencedirect.com/science/article/pii/S1364032116000551

[4] Programmable Logic Controllers (PLC) and its Programming – IJERT. IJERT – International Journal of Engineering Research & Technology [online]. Copyright © [cit. 21.05.2023]. Dostupné z: https://www.ijert.org/programmable-logic-controllers-plc-and-its-programming

[5] Function Block Diagram. In: PLC Programming according to IEC 61131-3 [online]. [cit. 2023-05-22]. Dostupné z:

[6] Walsh Medical Media | Open Access Journals [online]. Copyright © [cit. 21.05.2023]. Dostupné z: https://www.walshmedicalmedia.com/open-access/robot-manipulator-control-using-plc-with-position-based-and-image-based-algorithm-2090-4908-1000154.pdf

[7] What is a Robotic Manipulator? - Robots Done Right. Robots Done Right - Used Robot Sales [online]. Dostupné z: https://robotsdoneright.com/Articles/what-is-a-robotic-manipulator.html

[8] ScienceDirect. ScienceDirect [online]. Copyright © [cit. 22.05.2023]. Dostupné z: https://www.sciencedirect.com/science/article/pii/B978032395348100034X

[9] [online]. In: . [cit. 2023-05-22]. Dostupné z: https://www.freepik.com/premium-photo/orange-robot-arm-manipulator_8058994.htm

[10] Delta 3 robot [online]. In: . [cit. 2023-05-22]. Dostupné z: https://www.se.com/ww/en/product/VRKP2S0RNC00000/delta-3-robot-p2-34-axis-15-kg-permissible-load-0800-mm-working-envelop/

[11] [online]. Copyright © [cit. 22.05.2023]. Dostupné z: http://www.elearn.vsb.cz/archivcd/FS/PRM/Text/Skripta_PRaM.pdf

[12] Moved Permanently [online]. Dostupné z: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=973374

[13] CORE – Aggregating the world's open access research papers [online]. Copyright © [cit. 22.05.2023]. Dostupné z: https://core.ac.uk/download/295583387.pdf

[14] Movement of the KUKA LBR iiwa at the redundancy circle. In: RedRobCo: Redundant Robot Control [online]. [cit. 2023-05-22]. Dostupné z: https://www.joanneum.at/en/robotics/reference-projects/finalized-projects/redrobco-redundant-robot-control

[15] Regularization of the inverse positioning problem of revolute joint manipulators [online]. Budapest, Hungary, May 2015 [cit. 2023-05-22]. Dostupné z: https://www.researchgate.net/publication/277182473_Regularization_of_the_inverse_positioning_problem_of_revolute_joint_manipulators

[16] Gasparetto, A., Boscariol, P., Lanzutti, A. et al. Trajectory Planning in Robotics. Math.Comput.Sci. 6, 269–279 (2012). Dostupné z: https://doi.org/10.1007/s11786-012-0123-8

[17] [online]. In: . [cit. 2023-05-22]. Dostupné z: https://www.mathworks.com/products/robotics.html

[18] Robot Simulation Software: Everything You Need to Know - Visual Components. Visual Components - 3D manufacturing simulation software [online]. Copyright © Visual Components 2023 [cit. 21.05.2023]. Dostupné z: https://www.visualcomponents.com/resources/blog/robot-simulation-software-everything-you-need-to-know/

[19] Robotics simulation before solution execution [online]. [cit. 2023-05-22]. Dostupné z: https://unity.com/solutions/automotive-transportation-manufacturing/robotics

[20] What is a Cartesian robot?. Linear Motion Tips [online]. Dostupné z: https://www.linearmotiontips.com/what-is-a-cartesian-robot/

[21] Delta robot - Wikipedia. [online]. Dostupné z: https://en.wikipedia.org/wiki/Delta_robot

[22] SCARA Robots: Guide to The Most Versatile and Sought After Robot | #HowToRobot. Connecting Industrial Robot Suppliers & Buyers – #HowToRobot [online]. Dostupné z: https://howtorobot.com/expert-insight/scara-robots

[23] Delta nebo SCARA robot: Rozdíly, které musíte znát – FactoryAutomation.cz. FactoryAutomation.cz – Magazín o průmyslové automatizaci a robotice [online]. Copyright © 2014 [cit. 02.05.2023]. Dostupné z: https://factoryautomation.cz/delta-nebo-scara-robot-rozdily-ktere-musite-znat/

[24] Steinerova věta – Wikipedie. [online]. Dostupné z: https://cs.wikipedia.org/wiki/Steinerova_v%C4%9Bta

[25] KOLÁČNÝ, Josef. Elektrické pohony. Brno, datum vydání není známo.

[26] Load Torque Calculation Formula | Miki Pulley . [online]. Copyright © Miki Pulley Co., Ltd All Rights Reserved. [cit. 16.05.2023]. Dostupné z: https://www.mikipulley.co.jp/EN/Services/Tech_data/tech26.html

[27] 5.1 Inverse Kinematics - YouTube. YouTube [online]. Copyright © 2023 Google LLC [cit. 23.05.2023]. Dostupné z: https://www.youtube.com/watch?v=RH3iAmMsolo&t=203s&ab_channel=Woolfrey

[28] GREPL, Robert. Kinematika a dynamika mechatronických systémů [skriptum]. Olomučany: AKADEMICKÉ NAKLADATELSTVÍ CERM, s.r.o. Brno, 2007 [cit. 2023-05- 08]. ISBN 978-80-218-3530-8.

[29] ResearchGate | Find and share research [online]. Dostupné z: https://www.researchgate.net/publication/220494116_Selectively_Damped_Least_Squares_for_Inverse_Kinematics

[30] Gaussova eliminační metoda – Wikipedie. [online]. Dostupné z: https://cs.wikipedia.org/wiki/Gaussova_elimina%C4%8Dn%C3%AD_metoda

[31] Průmyslová komunikace OPC UA - 1.díl - popis protokolu | Automatizace.HW.cz. Automatizace.HW.cz | Elektronika v automatizaci [online]. Dostupné z: https://automatizace.hw.cz/prumyslova-komunikace-opc-ua-1dil-popis-protokolu.html

[32] FB_KinConfigGroup [online]. In: . [cit. 2023-05-22]. Dostupné z: https://infosys.beckhoff.com/english.php?content=../content/1033/tf5110-tf5113_tc3_kinematic_transformation/1955711627.html&id=

[33] [online]. In: . 2015 [cit. 2023-05-22]. Dostupné z: https://www.designportal.cz/vut-v-brne-rebranduje-tenke-linie-strida-pismeno-t/

[34] STAVBAAPROVOZCNC | UVSSR | FSI | VUT. [online]. Dostupné z: https://www.os.fme.vutbr.cz/StavbaAProvoz/ev-1/Kapitola8/AspektyBezpecnosti

[35] Jak bezpečně začlenit koboty na průmyslové pracoviště | Automatizace.HW.cz. Automatizace.HW.cz | Elektronika v automatizaci [online]. Dostupné z: https://automatizace.hw.cz/jak-bezpecne-zaclenit-koboty-na-prumyslove-pracoviste.html

# Seznam použitých zkratek a symbolů

| | |
|---|---|
| PLC | Programmable Logic Controller |
| CAD | Computer Aided Design |
| delta | typ paralelního manipulátoru (název podle shodného tvaru se symbolem Δ) |
| SCARA | Selective Compliance Articulated Robot Arm |
| DLS | damped least-squares |
| OPC | Open Platform Communications |
| UA | Unified Architecture |
| DA | Data Access |
| STL | Standard Template Library |
| CPU | Central Processing Unit |
| FPGA | Field Programmable Gate Arrays |
| TCP | Transmission Control Protocol |
| IP | Internet Protocol |
| HTTP | Hyper Text Transfer Protocol |
| SOAP | Simple Object Access Protocol |
| URL | Uniform Resource Locator |
| NX | CAD program |
| MCD | Mechatronics Concept Designer |
| HMI | Human Machine Interface |
| i | převodový poměr |
| m | hmotnost |
| r | vzdálenost těžiště od osy rotace |
| $J_x$ | moment setrvačnosti |
| $M_s$ | statický moment |
| $M_d$ | dynamický moment |
| η | účinnost |
| ω | úhlová rychlost |
| ε | úhlové zrychlení |
| g | tíhové zrychlení |
| $F_G$ | tíhová síla |
| p | stoupaní závitu |
| μ | součinitel tření |

| | |
|---|---|
| v | rychlost výsuvné hřídele |
| RR | rotace, rotace |
| f | funkce |
| q | vektor kloubových souřadnic |
| X | vektor kartézských souřadnic |
| L | délka ramene |
| d | přepona |
| β | úhel mezi ramenem a přeponou |
| J | jakobián |
| $J^{-1}$ | inverze jakobiánu |
| $J^{\dagger}$ | pseudoinverze jakobiánu |
| $\lambda$ | součinitel tlumení |
| det() | determinant |

# Seznam příloh

Elektronické přílohy:

- Řešení inverzní úlohy a zpracování obrázku – složka: *SCARA_Matlab*
- Model robota SCARA – složka: *SCARA_NX*
- Řízení modelu – složka: *SCARA_TwinCAT*