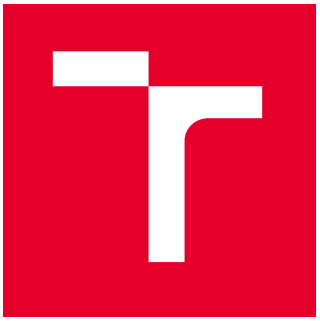


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA STROJNÍHO INŽENÝRSTVÍ

FACULTY OF MECHANICAL ENGINEERING

ÚSTAV MECHANIKY TĚLES, MECHATRONIKY A BIOMECHANIKY

INSTITUTE OF SOLID MECHANICS, MECHATRONICS AND BIOMECHANICS

NÁVRH MODELU ČTYŘNOHÉHO CHODÍCÍHO ROBOTU

DESIGN OF A FOUR-LEGGED WALKING ROBOT MODEL

BAKALÁŘSKÁ PRÁCE

BACHELOR'S THESIS

AUTOR PRÁCE — **Pavel - Jakub Novotný**

AUTHOR

VEDOUCÍ PRÁCE — **Ing. Martin Appel**

SUPERVISOR

BRNO 2023

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA STROJNÍHO INŽENÝRSTVÍ

Zadání bakalářské práce

Ústav:	Ústav mechaniky těles, mechatroniky a biomechaniky
Student:	**Pavel - Jakub Novotný**
Studijní program:	Mechatronika
Studijní obor:	bez specializace
Vedoucí práce:	**Ing. Martin Appel**
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 bakalářské práce:

Návrh modelu čtyřnohého chodícího robotu

Stručná charakteristika problematiky úkolu:

Práce se bude zabývat vytvořením modelu pro čtyřnohého robotu v programu Simulink 3D Animation. Zmíněný robot vznikl jako aktivita studentů v roce 2003. Dále na robotu pracovalo několik studentů v rámci diplomových a bakalářských prací. Motivací této práce je umožnit dalším studentům vyvíjet chodící algoritmy. Možnost si výsledný algoritmus vyzkoušet v simulaci by proces vývoje značně urychlil.

Cíle bakalářské práce:

1. Rešerše v oblasti chůze čtyřnohého robotu
2. Seznámit se se současným stavem robotu a souvisejícími DP a BP. Popsat stávající stav algoritmu chůze čtyřnohého robotu.
3. Vytvoření modelu pro čtyřnohý robot.
4. Vytvoření algoritmu chůze čtyřnohého robotu a otestovaní v simulaci.

Seznam doporučené literatury:

CORKE, Peter I. Robotics, vision and control: fundamental algorithms in MATLAB. Berlin: Springer, 2011. Springer tracts in advanced robotics, v. 73. ISBN 9783642201448.

GREPL, Robert. Kinematika a dynamika mechatronických systémů. Brno: Akademické nakladatelství CERM, 2007. ISBN 978-80-214-3530-8.

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

V Brně, dne

L. S.

prof. Ing. Jindřich Petruška, CSc. ředitel ústavu	doc. Ing. Jiří Hlinka, Ph.D. děkan fakulty

Abstrakt

Tato práce se zabývá rešerší v oblasti vývoje čtyřnohých robotů, vytvořením modelu robota v prostředí Simulink 3D Animation a návrhem algoritmu pro chůzi čtyřnohého robota.

Práce rozvíjí studentský projekt Jaromír z roku 2003 a to pomocí sestavení robota s vazbou parent - child, která je spolehlivější pro funkci robota. K vytvoření algoritmu chůze práce používá přímou kinematiku. Pro plynulost pohybu robota jsou natočení postupně inkrementována.

Cíle této práce byly rešerše v oblasti chůze čtyřnohého robota, vytvoření modelu čtyřnohého robota a vytvoření algoritmu chůze čtyřnohého robota a otestování v simulaci.

Summary

This thesis deals with research in the field of quadruped robot development, creation of robot model in Simulink 3D Animation environment and design of algorithm for quadruped robot walking.

This thesis develops the student project Jaromir from the year 2003 by building a robot with a parent-child bond that is more reliable for the robot's function. The thesis uses direct kinematics to develop the gait algorithm. The rotations are incrementally incremented for smooth robot motion.

The objectives of this thesis were to research in the area of quadruped robot gait, create a quadruped robot model, and create a quadruped robot gait algorithm and test it in simulation.

Klíčová slova

Simulink 3D Animation, F3D, čtyřnohý robot, modelování, Matlab

Keywords

Simulink 3D Animation, F3D, quadruped robot, modelling, Matlab

Bibliografická Citace

NOVOTNÝ, P. J. *Návrh modelu čtyřnohého chodícího robotu.* Brno: Vysoké učení technické v Brně, Fakulta strojního inženýrství, 2023. 37 s., Vedoucí diplomové práce: Ing. Martin Appel.

Prohlašuji, že jsem bakalářskou práci a téma Návrh modelu čtyřnohého chodícího robotu zpracoval sám na základě pokynů vedoucího bakalářské práce Ing. Martina Appela. Vycházel jsem při tom ze svých znalostí, odborných konzultací a literárních zdrojů.

Pavel-Jakub Novotný

Brno . . . .

Chtěl bych poděkovat vedoucímu práci Ing. Martinu Appelovi za cenné rady při řešení práce. Také bych chtěl poděkovat své rodině za podporu při studiu.

Pavel-Jakub Novotný

Obsah

1 Úvod **8**

2 Rešerše **9**
2.1 Vývoj a aplikace čtyřnohých robotů 9
2.2 Kinematika čtyřnohého robota 10
2.2.1 Popis anatomie robota 10
2.2.2 Přímá kinematika 10
2.3 Chůze čtyřnohých robotů 12
2.3.1 Oblast stability 12
2.3.2 Statická chůze 13
2.3.3 Dynamická chůze 13
2.4 Vývoj čtyřnohého robota Jaromír 13
2.4.1 Současný čtyřnohý robot Jaromír 14
2.5 Prostředí Simulink 3D animation 15

3 Model čtyřnohého robota **16**
3.1 Postup vytvoření modelu robota 16
3.1.1 Import modelu 19
3.1.2 Jednotky 20
3.1.3 Problémové funkce a proměnné 21

4 Algoritmus chůze **27**
4.1 Návrh algoritmu chůze čtyřnohého robota 27
4.1.1 Délka kroku 28
4.1.2 Výpočet těžiště 30
4.1.3 Chůze čtyřnohého robota 31

5 Závěr **35**

Reference **36**

1 Úvod

Od vytvoření prvního robota nabrala robotika velmi rychlý spád. Roboti se stávají většími, komplexnějšími a získávají další a další schopnosti. Jednou z důležitých vlastností mobilních robotů je možnost pohybu, ať už je to robot kolový nebo se pohybuje za pomoci noh. Jak to už to tak bývá, tak v přírodě bývá většina věcí vyřešena úsporně a elegantně a člověk se snaží tato věci napodobit. Jednou z takových věcí je chůze dvounohých ale i vícenohých tvorů.

Vývoj čtyřnohých robotů se v posledních letech značně zrychluje, je to zapříčiněno vzrůstajícím počtem aplikací, na které běžní kolový roboti nestačí. Jedná se zejména o úkoly ve složitých nerovných terénech, např. záchranné operace, vojenské akce atd. Čtyřnozí roboti získávají nové schopnosti a dostávají se do popředí.

Tato práce na téma Návrh modelu chodícího čtyřnohého robota, navazuje na řetězec jiných prací vytvořených při studentském projektu čtyřnohého robota Jaromíra. Projekt Jaromír je studentský projekt započatý v Mechlabu v roce 2003. Tento projekt byl postupně rozvíjen a vylepšován. Ambicí této práce je tedy vytvořit model čtyřnohého robota pro vývoj algoritmů chůze a jeho testování v simulaci.

Tato bakalářská práce se nejprve zabývá rešerší v oblasti čtyřnohých robotů a jejich chůze, přičemž dále navazuje vytvořením modelu čtyřnohého robota, který byl výrazně modernizován v této diplomové práci [1], v prostředí Simulink 3D Animation. Posledním bodem zadání je vytvoření algoritmu chůze a otestování jej v simulaci.

Práce se snaží zjistit zda je prostředí Simulink 3D Animation vhodné pro vytvoření komplexního modelu čtyřnohého robota a jeho následné otestování. Tato práce má pomoci usnadnit návrh algoritmů chůze pro čtyřnohé roboty, jelikož před aplikací na reálném hardwaru by tak mohly být otestovány v simulaci.

Tato práce má tři kapitoly. V kapitole 2 Rešerše je krátce zmíněn vývoj čtyřnohých robotů, popis kinematiky a způsoby jejich chůze. Dále je zde popsán vývoj projektu čtyřnohého robota Jaromír a krátký popis prostředí F3D v němž je tato práce realizována.

V kapitole 3 Model čtyřnohého robota je popsán postup vytváření modelu robota a jeho vkládání do prostředí. Je tu také popsán native formát F3D a problémy spojené s jeho vytvářením. Dále jsou zde zmíněny některé problémy, které byly objeveny při tvorbě modelu robota.

V následující kapitole 4 Algoritmus chůze je popsán algoritmus chůze čtyřnohého robota, který je spuštěn v simulaci. Také je zde popsán výpočet kroku a těžiště robota.

2 Rešerše

Čtyřnozí roboti jsou typem robotů, kteří mají čtyři nohy a jejichž konstrukce je inspirována přírodou. Tito roboti se konstrukcí podobají čtyřnohým zvířatům např. psům nebo kočkám. Na obrázku 2.1 je prototyp čtyřnohého robota od firmy Boston Dynamics.

Krom čtyřnohých robotů existují i vícenozí roboti. Tito roboti mají širokou škálu aplikací, včetně průzkumu terénu, záchranných misí, výzkumu, výroby, logistiky a dalších. Čtyřnozí roboti mají většinou nohy umístěny pod tělem, přičemž tyto jsou nezávisle řízeny pomocí elektroniky a softwaru. Je to z důvodu lepšího pohybu robota a snazšímu přizpůsobení se okolnímu terénu.

Obrázek 2.1: Čtyřnohý robot od firmy Boston Dynamics [2]

Čtyřnozí roboti mají své výhody oproti robotům s koly. Jednou z výhod je třeba lepší překonávání nerovností a pohyb po nezpevněných površích. Navíc, díky jejich schopnostem pohybovat se v terénu s nízkým třením, jsou méně náchylné k uvíznutí nebo zablokování, což může být např. při průzkumu terénu výhodou. Čtyřnozí roboti navíc umožňují různé typy pohybu, jako je například běh, plazení nebo chůze po schodech.

Na druhou stranu mají tyto stroje také své nevýhody. Jedná se třeba o vyšší náklady na údržbu a výrobu, pomalejší rychlosti pohybu a menší efektivitu chůze na hladkých a rovných površích. Jednou z jejich nevýhod v porovnání s roboty kolovými je relativně složitější řízení a ovládání chůze robota.

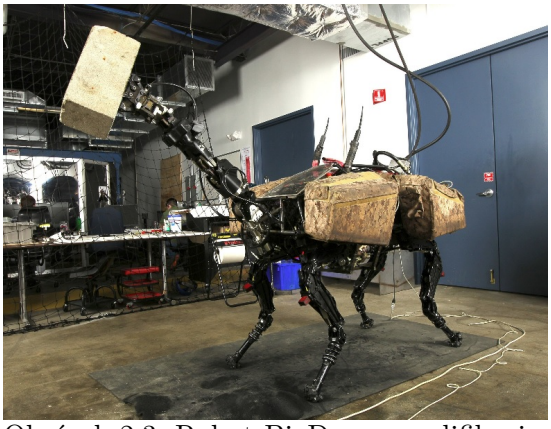
2.1 Vývoj a aplikace čtyřnohých robotů

Jejich vývoj se datuje až do 80. let 20. století. První generace robotů byla velká a neohrabaná a byla používána především pro vědecké účely. S postupným vývojem řídící a mikroprocesorové elektroniky došlo k zmenšení a odlehčení, čímž se posunula oblast jejich

využití od vědeckých projektů směrem ke komerčnímu využití.

Jednou z klíčových aplikací čtyřnohých robotů je průzkum a monitorování těžko přístupných nebo nebezpečných míst, jako jsou průmyslové havárie nebo radioaktivní zóny. Roboti jsou dálkově řízeni, čímž se omezují rizika pro lidské pracovníky. V současné době jsou tyto stroje stále vyvíjeny a zdokonalovány, aby mohly být využity v nových oblastech vědy a průmyslu, a poskytovaly tak nová řešení pro různé problémy (obrázky 2.2,2.3).

Obrázek 2.2: Podpůrná aplikace robota [3]



Obrázek 2.3: Robot BigDog v modifikaci s manipulátorem [4]

Na obrázku 2.2 je čtyřnohý robot LS3 (Legged Squad Support System) od firmy Boston Dynamics, testovaný armádou USA. Robot LS3 byl koncipován jako armádní robotický nosič, který unese 181 kg (400 liber). Pohon robota zajišťoval zážehový motor. Vývoj tohoto robota byl nakonec zrušen z důvodu nadměrného hluku, který by prozradil jednotku [3].

Na obrázku 2.3 je čtyřnohý robot BigDog, který byl vyvinut společností Boston Dynamics. Program robota BigDog byl nejprve tajný, po odtajnění získal několik modifikací. Jedna z posledních modifikací je, že robot obdržel mechanickou paži s nosností 300 kg. Tato modifikace byla zamýšlena jako podpůrná jednotka k ženijním operacím [4].

2.2 Kinematika čtyřnohého robota

2.2.1 Popis anatomie robota

Kinematika čtyřnohých robotů se zabývá pohybem čtyřnohých robotů a popisuje geometrické vlastnosti jejich pohybu. Čtyřnozí roboti se pohybují na základě změny polohy jejich nohou.

2.2.2 Přímá kinematika

Přímá neboli dopředná kinematika je obor robotiky, který se zabývá výpočtem pozice a orientace koncového efektoru robota na základě znalosti pohybu jednotlivých částí robota. Dopředná kinematika pracuje s matematickými vztahy, které popisují vztah mezi konkrétními parametry, pozicí a orientací koncového efektoru. Tyto matematické vztahy jsou z pravidla vyjádřeny pomocí matic transformace.

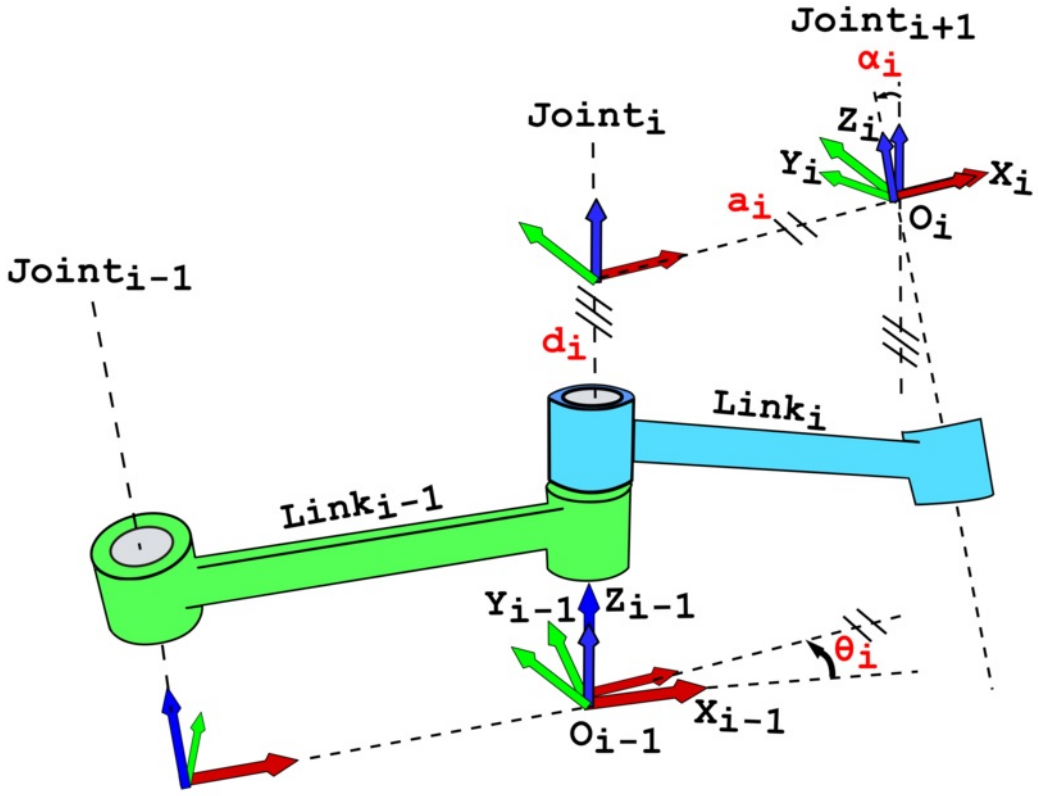
Přímá kinematika je důležitá pro plánování a řízení pohybu robota. Při plánování cesty robota je potřeba vědět, jaké hodnoty musí mít nastaveny na jednotlivých kloubech robota, aby se koncový efektor dostal na požadované místo.

DH parametry

Použití DH (Denavit-Hartenberg) parametrů je jeden ze způsobů popisu řetězců kinematických vazeb. DH parametry se používají zejména v robotice. Pro popis soustavy DH parametry stačí čtyři parametry, a to na rozdíl od dopředné kinematiky, kde jsou potřeba 3 posuny a 3 rotace tj. parametrů šest.

> "Manipulátor s N klouby je číslován od 1 do N a celkový počet částí manipulátoru je N+1, s částmi číslovanými od 0 do N. Kloub i spojuje část *i-1* s částí i a umožňuje jejich vzájemné posunutí. To znamená, že část i je připojena kloubem i k části *i+1*. Většinou se část 0 označuje jako základna manipulátoru a část N jako koncový efektor."[5]

Parametry a_i a α_i jsou konstantní, jedná se totiž o konstantní parametr systému. Parametr d_i je proměnná posuvné vazby, pokud se jedná o vazbu rotační, tak je parametr d_i konstantní. θ_i je proměnný parametr rotační vazby ale pro posuvnou vazbu je parametr konstantní.



Obrázek 2.4: Denavit-Hartenberg parametry [6]

kde

- a_i - vzdálenost mezi body O_i a O_i'
- d_i - vzdálenost mezi body O_i' a O_{i-1}
- α_i - úhel mezi osami z_{i-1} a z_i okolo osy x_i

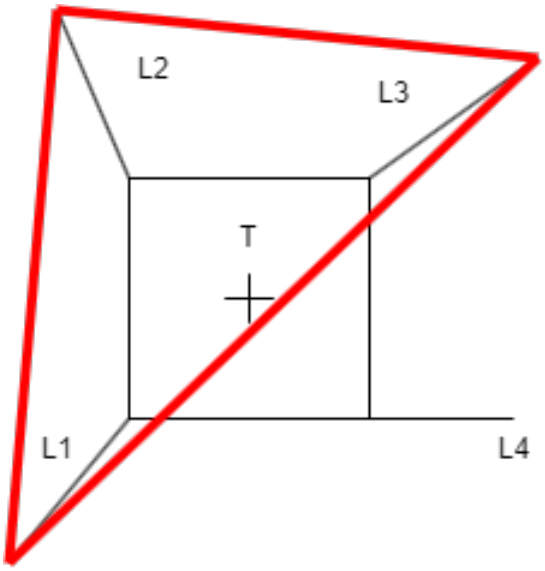
- θ_i - úhel mezi osami x_{i-1} a x_i okolo z_{i-1}
- $Joint_i$ - označení i-tého kloubu
- $Link_i$ - označení i-té části (prutu)

2.3 Chůze čtyřnohých robotů

2.3.1 Oblast stability

Oblast stability robota označuje pomyslný prostor jeho pohybu, který určuje zda je robot v daném místě stabilní. Z hlediska stability, můžeme roboty rozdělit do dvou základních kategorii a to na staticky stabilní a dynamicky stabilní.

Stabilita robota je klíčovým faktorem při návrhu a výrobě. Různé typy robotů jsou navrženy pro různé úkoly, a mají tak odlišné požadavky na stabilitu v závislosti na podmínkách, ve kterých budou pracovat. Mezi klíčové faktory, které ovlivňují stabilitu robota, patří hmotnost, geometrie těla, umístění těžiště robota a povrch terénu, kde má robot pohybovat. Tyto faktory musí být zohledněny při návrhu a vývoji robota, aby byla zajištěna jeho stabilita.



Obrázek 2.5: Oblast stability

Na obrázku 2.5 je jednoduché schéma čtyřnohého robota. Nohy robota jsou označeny L1, L2, L3 a L4. Těžiště robota je označeno písmenem T. Nohy L1, L2 a L3 mají určité natočení a jsou v kontaktu se zemí. Noha L4 se zrovna přesouvá na novou pozici, takže není v kontaktu se zemí. Červený trojúhelník označuje aktuální oblast stability, jehož vrcholy jsou v koncových efektorech jednotlivých noh.

Pokud se těžiště robota dostane mimo tuto oblast, robot začne padat. U statické chůze robot spadne, přičemž během dynamické je pád za určitých podmínek vykompenzován. Dynamická chůze je založena na tom, že robot přechází přes nestabilní oblast. Má-li robot odpovídající algoritmus chůze, tak mu krátké opuštění stabilní oblasti nevadí.

2.3.2 Statická chůze

U statické chůze se v určitý moment pohybuje pouze jedna noha. Při statické chůzi musí být čtyřnohý robot neustále v oblasti stability. Proto je nutná neustálá kontrola těžiště, zda neopustilo oblast stability. Pokud by se tak stalo, může to vést k pádu robota nebo k jiným nežádoucím důsledkům.

Statická chůze se moc nepodobá chůzi čtyřnohých živočichů, protože je trhaná a pomalá. Staticky stabilní roboti jsou často jednodušší na návrh a výrobu, ale jejich schopnosti přizpůsobit se okolním podmínkám jsou omezené. [7]

2.3.3 Dynamická chůze

Při dynamické chůzi těžiště robota opouští oblast stability. Je to kvůli tomu, že se robot pohybuje více částmi najednou, aby dosáhl co nejplynulejšího pohybu. Většinou přesouvá nohu na novou pozici a zároveň posouvá tělem. Právě roboti, kteří neustále padají nebo se jinak kymácí, se pokoušejí o dynamickou chůzi. Regulovat dynamickou chůzi vyžaduje mnohem složitější systémy. Na rozdíl od statické chůze se chůze dynamická v přírodě vyskytuje velmi často.

Dynamicky stabilní roboti jsou navrženi tak, aby udržovali rovnováhu i při pohybu. Tito roboti jsou schopni pohybu do různých směrů a přizpůsobení se různým podmínkám. Dynamicky stabilní roboti jsou složitější na návrh a výrobu, ale mají větší pohybovou flexibilitu a jsou schopni se pohybovat v náročných terénech. V odkazu je video robota s dynamickou chůzí [8].

2.4 Vývoj čtyřnohého robota Jaromír

Robot Jaromír je projekt zabývající se návrhem a řízením čtyřnohého chodícího robota [9]. První model, viz obrázek 2.6, byl ze sešroubovaných duralových. Pohyb byl zajištěn servopohony, které byly řízeny mikrokontrolerem Atmel.

Obrázek 2.6: První verze robota Jaromíra [1]

Michal Švehlák – DP – 2004 [10]
–Návrh konstrukce experimentálního kráčejícího robotu

Tato práce se zabývá návrhem konstrukce experimentálního čtyřnohého robota Jaromír [9] a jeho řízením. Pohyb nohou je zajištěn servopohony. Konstrukce je navržena tak, že je možné změnit rozsah jednotlivých servopohonů.

Miroslav Světlík – BP – 2006 [11]
–Kinematické řízení a virtualizace prototypu čtyřnohého kráčejícího robotu

Tato práce rozšířila experimentálního čtyřnohého robota Jaromír [9]. Byl přidán způsob ovládání robota analogovým ovladačem a způsob chůze v neregulárním terénu.

Marek Gogola – BP – 2007 [12]
– Konstrukce mechanické části servomechanismu

Tato práce se zabývá návrhem konstrukce mechanické části elektrického servopohonu pro kráčejícího robota. Cílem práce bylo minimalizovat hmotnost při dostatečné tuhosti a především zajištění požadované životnosti pohonu.

Jiří Konvičný – BP – 2007 [13]
–Návrh mechanické části servomechanismu
Václav Sztwiertnia – DP – 2007 [14]
–Využití mems akcelerometru pro stabilizaci chodícího robotu

Tato práce měla za cíl implementovat akcelerometry MEMS na robota a vytvoření potřebného řídícího softwaru. Pro jejich implementaci bylo nutné přepracovat řídící jednotku robota.

Daniel Youssef – BP – 2013 [15]
–Řídící jednotka čtyřnohého robotu na bázi mikrokontroléru PIC

Tato práce přidává ke konstrukci čtyřnohého robota Jaromíra [9] senzory a doplňuje ho o novou řídící jednotku.

2.4.1 Současný čtyřnohý robot Jaromír

Jan Králík - DP - 2018 [1]
- Kompletní přepracování robota Jaromíra

Bylo navrženo nové tělo robota, pro snazší úpravy bylo vyrobeno 3D tiskem, viz obrázek 3.1. Dále byla vytvořena kompletně nová deska, která byla rozšířena o sloty pro senzory. Robot obdržel senzory došlapu a byl vytvořen algoritmus chůze.

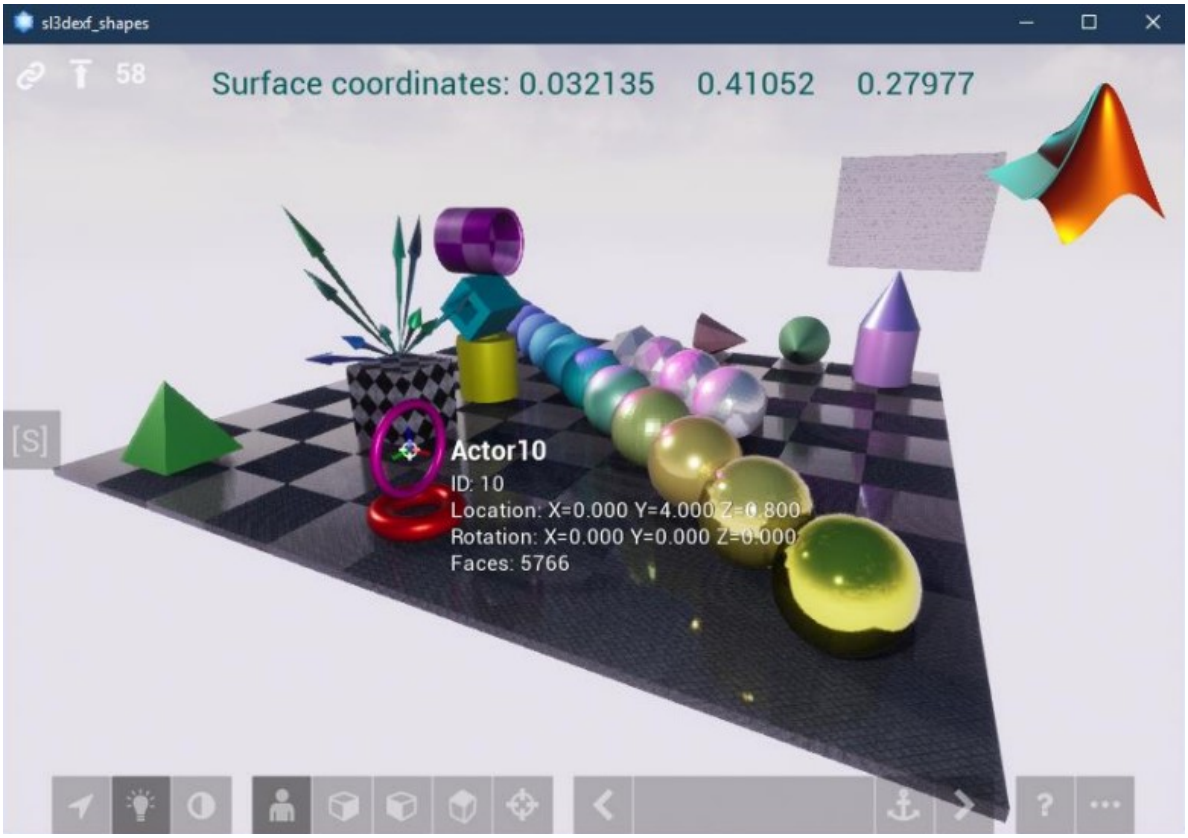
2.5 Prostředí Simulink 3D animation

Simulink 3D Animation je nástroj v prostředí Matlab a Matlab Simulink, který umožňuje vizualizaci a simulaci dynamických systémů jako roboti, stroje, letadla a dalších zařízení v 3D prostoru. Tento nástroj umožňuje uživatelům vytvářet interaktivní 3D animace a simulace, které dávají možnost detailního zkoumání systému v reálném čase.

Prohlížeč 3D Foundation Viewer je založen na herním enginu Unreal, který umožňuje používat jeho funkce z prostředí Matlab a Simulink. Pro detekci kolizí je použit fyzikální engine od Nvidie PhysX, který umožňuje nastavení různých fyzikálních vlastností.

Simulink 3D Animation umožňuje importovat CAD modely různých formátů (např. STL, F3D, VMRL, FBX. SDF) a virtuální scény, do kterých mohou být tyto modely umístěny. Uživatelé mohou pak vytvářet animace a simulace těchto modelů, které zahrnují různé fyzikální vlastnosti jako gravitace, tření, kolize atd. K dispozici je také knihovna tvarů pro různé typy objektů, jako koule, šipka, toroid, letadlo a další. Rovněž zde existuje i možnost nastavit vlastnosti objektů jako barva, průhlednost, textura atd.

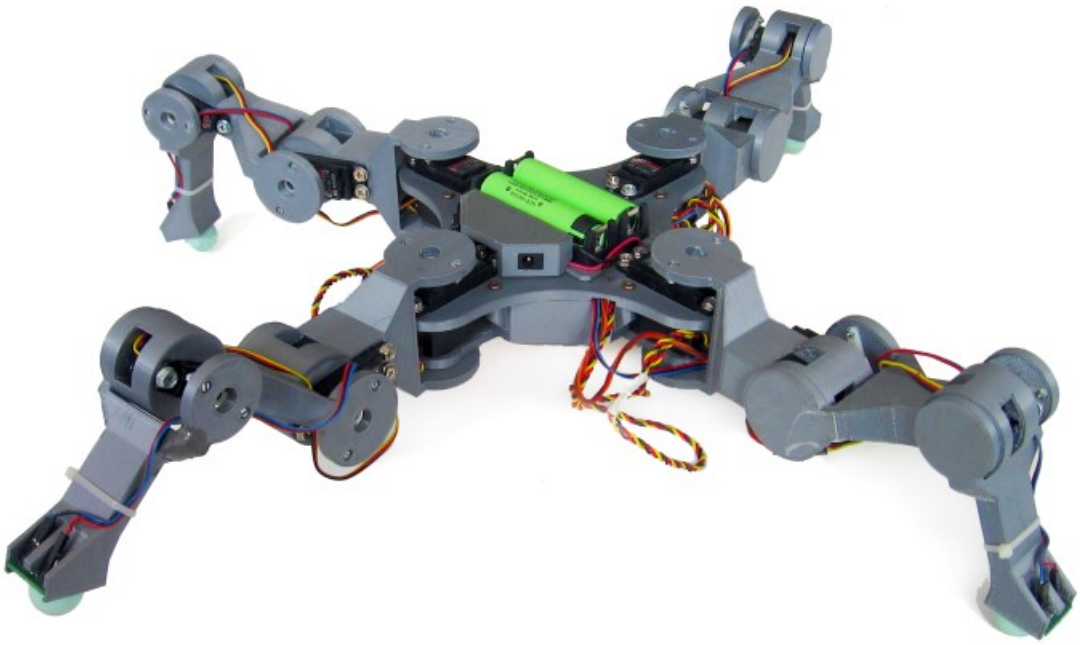
Simulink 3D Animation umožňuje uživatelům vytvářet scény s více kamerami a světly. Lze definovat animace, jako jsou rotace, posuvy nebo změny velikosti objektů. Dostupné jsou i interaktivní ovládací prvky jako jsou tlačítka nebo vyskakovací okna.



Obrázek 2.7: Vzorová ukázka prostředí F3D

3 Model čtyřnohého robota

Tato kapitola se zabývá vytvořením počítačového modelu čtyřnohého robota Jaromíra [9] v prostředí F3D a s tím spojenými problémy. V podkapitole 3.1 je popsán postup vytvoření a struktura modelu robota. Dále jsou zde popsány problémy, které se objevily při sestavování.



Obrázek 3.1: Vzhled reálného robota [1]

3.1 Postup vytvoření modelu robota

Jeden z bodů zadání této práce je vytvoření modelu čtyřnohého robota a jeho následné otestování v simulaci. V minulých letech byla vytvořena nová verze čtyřnohého robota Jaromír [1], která je dostupná v Mechlabu. Byla použita nejnovější verze modelu, aby kdokoliv, kdo vytvoří algoritmus chůze robota v simulaci, ho mohl otestovat i na reálném hardwaru.

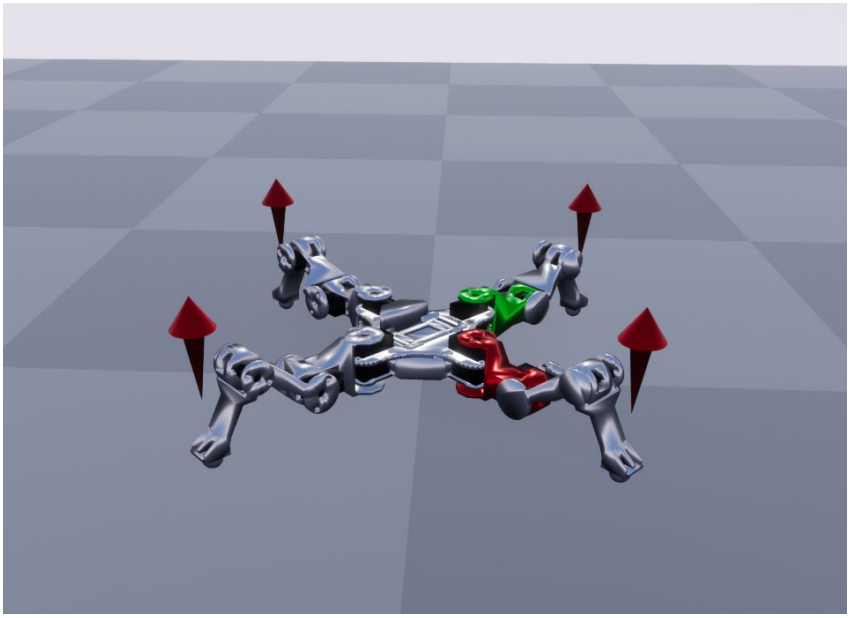
Po několika neúspěšných pokusech o vytvoření modelu čtyřnohého robota v prostředí F3D, se podařilo vytvořit částečně funkční model Robota. Avšak i tento exemplář má stále svá omezení viz. podkapitola 3.1.3. Struktura robota byla vytvořena tak, že každá následující část robota bude child částí předchozí (parent) viz. obrázek 3.3. Ukázka struktury kódu a vlastností modelu robota jsou k vidění níže.

```
Body = addChild(MW.Root,'Body');
load(Body, 'Telo_sestaveno.stl');
Body.Location = [0 0 3];
Body.Rotation = [0 0 45];
Body.Mass = 0.4;
Body.CenterOfMass = [0 0 1.65];
Body.Gravity = true;
Body.Collisions = true;
Body.Physics = true;
```

Každá část robota má svůj lokální souřadný systém. Souřadný systém rodiče (parent) je globálním souřadným systémem dítěte (child). Pro úplnost modelu byly domodelovány servopohony, které jsou u reálného robota významnými komponentami pro výpočet těžiště.

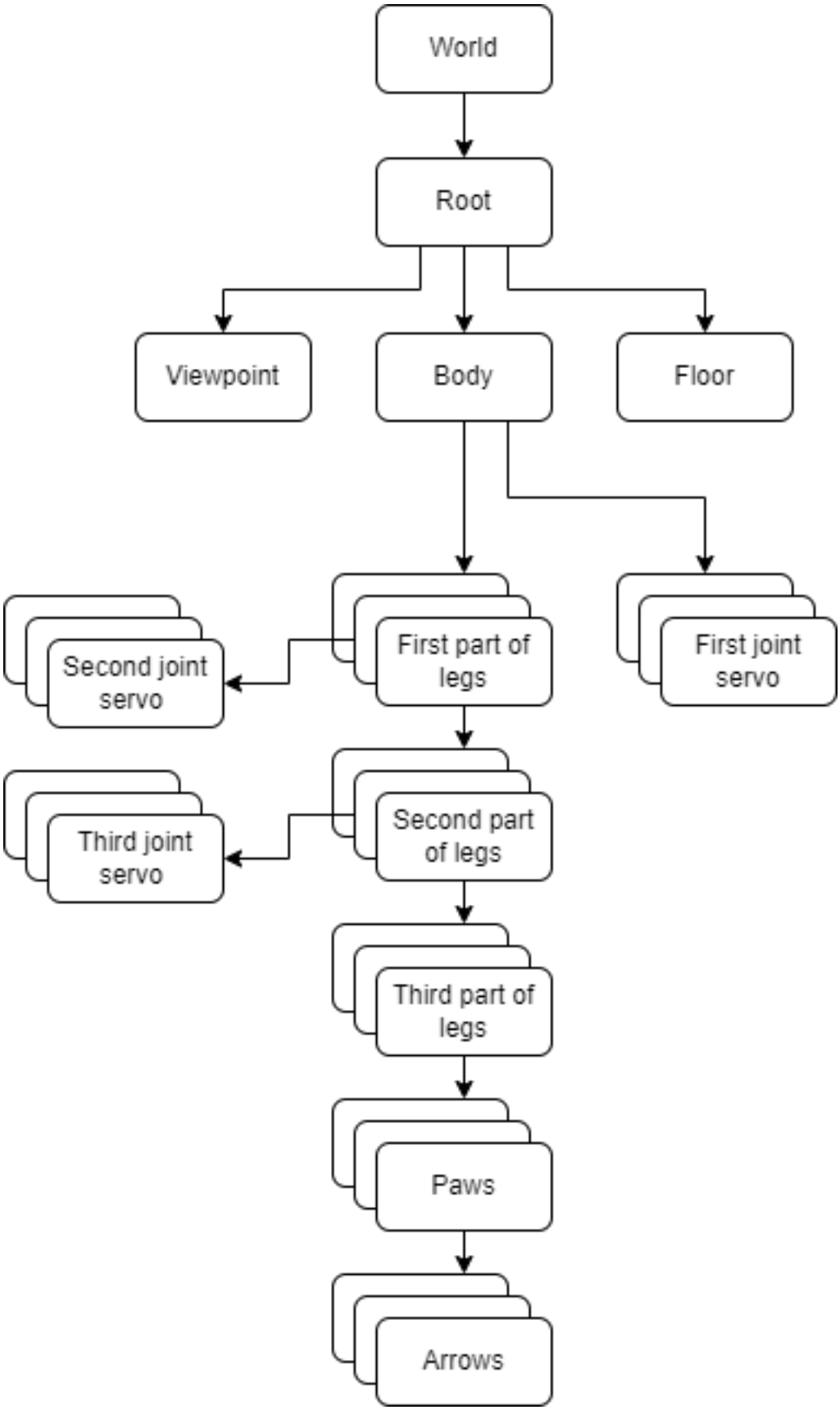
Vzhledem k tomu, že nohy jsou od sebe pootočeny. V případě druhé resp. čtvrté nohy je pootočení o 90° resp. o -90° a v případě třetí nohy je pootočení o 180° od nohy první. To znamená, že ke každému natočení nohy by muselo být připočteno příslušné natočení, aby se noha natočila dle požadavků.

Z důvodů jednoduššího ovládání byla vytvořena funkce `SetRotations`, která k zadanému natočení připočte zmíněné pootočení nohy. To znamená, že zadávaná natočení jsou pro každou nohu relativní. Pokud bude zadaná hodnota natočení 0°, což odpovídá výchozí pozici každé nohy, tak např. pro druhou nohu to bude znamenat absolutní natočení 90° okolo osy Z.



Obrázek 3.2: Model složeného robota

Do modelu robota byly přidány šipky (obrázek 3.2), pro znázornění interakce sil nohou robota se zemí. Šipka zmizí, když se noha robota zvedne a objeví se po zpětném dosednutí na zem.



Obrázek 3.3: Schéma struktury modelu robota

3.1.1 Import modelu

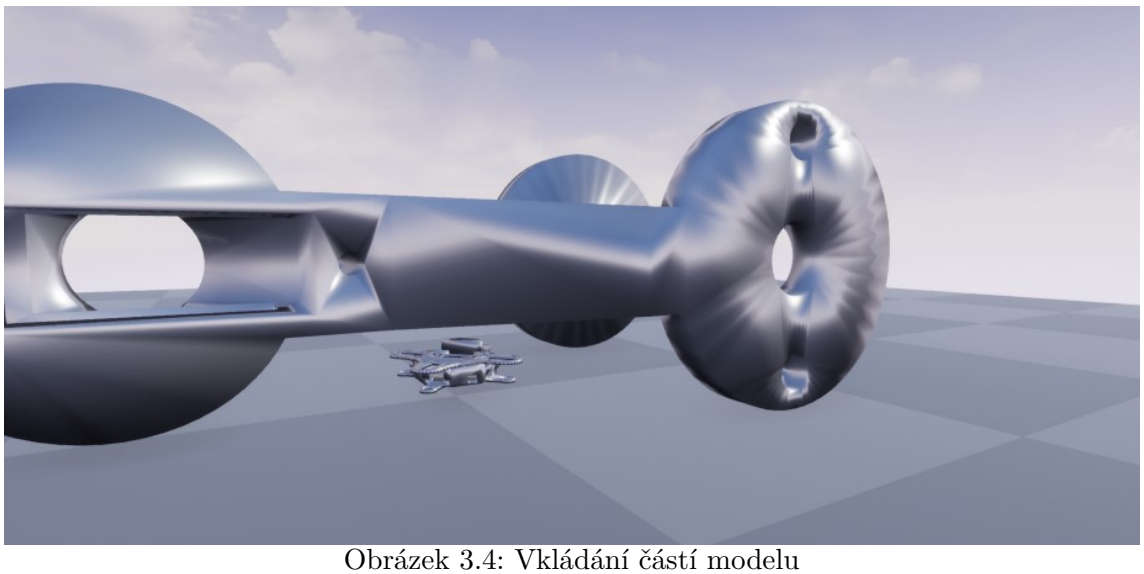
V F3D lze vložit model v zásadě dvěma způsoby. Za prvé, model je možné vytvořit pomocí předdefinovaných objektů, kterým stačí přiřadit výchozí parametry. Předdefinovány jsou různé základní objekty jako koule nebo krychle, ale také některé složitější jako letadlo. Tato možnost je v dokumentaci programu dobře popsána včetně syntaxe.

```
BasicGeo = addChild(MW.Root,'BasicGeo');
% V promenne BasicGeo vytvorim dite (child), ktery ma
%rodice (parent) MW.Root a nazvu jej BasicGeo
createShape(BasicGeo,'cylinder',[1 1 5]);
% Strukture BasicGeo priradim geometrii valce a rozmery [1 1 5]
```

Druhý způsob jak v F3D vytvořit model, je ho importovat z CADu. K vložení souboru se použije příkaz `load`. Podporované formáty souborů jsou např. X3D, WRL, FBX, STL a F3D. Pro import souboru z CADu se jako nejjednodušší ukázalo použití běžného formátu STL.

```
Body = addChild(MW.Root,'Body');
% V promenne Body vytvorim dite (child), ktery ma rodie (parent)
%MW.Root a bude se jmenovat Body
load(Body, 'Telo_sestaveno.stl');
% Do struktury Body nahraji geometrii
```

Při importu souboru z CADu je třeba si dát pozor na nastavení měřítka, protože sestavy ho můžou mít odlišné, než jednotlivé součásti (obrázek 3.4).



Obrázek 3.4: Vkládání částí modelu

Formát F3D

Soubor s příponou F3D je uváděn jako native, což znamená, že soubor v tomto formátu je nejlépe podporován. Problém nastává při pokusu o vytvoření tohoto typu souboru. Bylo zjištěno, že tento formát lze vytvořit v programu Fusion. Po jeho vytvoření a pokusu o nahrání do prostředí F3D je vygenerován error. Návod na vyřešení tohoto problému byl poskytnut jedním uživatelem na Mathworks fóru [16]. Totiž "správný"formát F3D lze vytvořit pouze Matlabem, jak je vidět na ukázce níže.

```
BasicGeo = addChild(MW.Root,'BasicGeo');
createShape(BasicGeo,'cylinder',[1 1 5]);


save(MW,'robot.f3d')     % Ulozim svet MW do souboru robot.f3d
```

Po vytvoření kódu generujícím model robota jsou všechny proměnné a data o importovaných tvarech uložena do souboru s příponou f3d. Vytvořený soubor je pak v Mainu načten příkazem `load` a je možné přistupovat k jednotlivým proměnným actorů a k proměnným jejich children.

3.1.2 Jednotky

Defaultně nastavené jednotky v F3D jsou metry a radiány. Změnit jednotky obvykle v těchto programech nebývá nic těžkého, avšak F3D je v tomto ohledu problematičtější. Jádrem problému je v příkazu `initialize`, pokud jsou jednotky nastaveny před inicializací, tak jsou resetovány (obrázek 3.5).

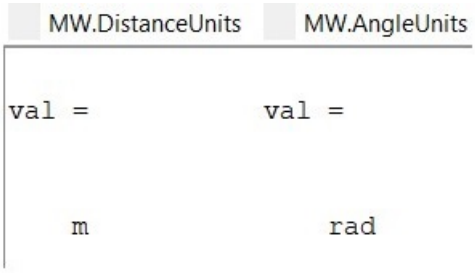
```
MW.AngleUnits = 'deg';
MW.DistanceUnits = 'mm';


initialize(MW, [0.1, 0.1, 0.04], [0 0 0]);  % inicializace ViewPointu
```

Problém lze obejít nastavením jednotek až po inicializaci. Nejspíš je to bug toolboxu, nejspíš bude co nejdříve odstraněn.

```
initialize(MW, [0.1, 0.1, 0.04], [0 0 0]);  % Spravne poradi zapisu


MW.AngleUnits = 'deg';
MW.DistanceUnits = 'mm';
```



Obrázek 3.5: Resetované jednotky

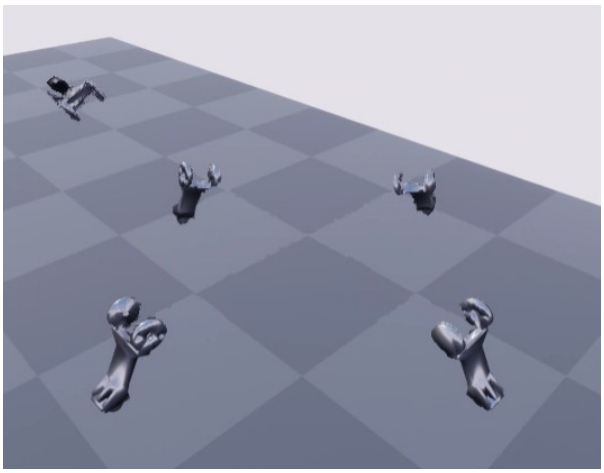
3.1.3 Problémové funkce a proměnné

Problémové funkce nejsou ty, které by nutně nefungovaly, do této kategorie jsou zahrnuty i funkce, které jsou nedostatečně popsané v dokumentaci programu. U nedostatečně popsaných funkcí nelze objektivně posoudit, zda nefungují nebo jen je špatně syntaxe.

Obrázek 3.6: Popis kinematické vazby v dokumentaci

Fyzika

Možnost zapnutí fyziky v tomto prostředí byla jednou ze stěžejních věcí při zadávání této bakalářské práce. Jak se později ukázalo, fyzika zde není plně funkční, resp. se jedná o herní fyziku. Fyzika je funkční u jednoduchých geometrických objektů, které nejsou spojeny vazbou parent-child. U složitějších geometrií se objekty rozlétnou nebo sebou nekontrolovatelně házejí (obrázek 3.7).



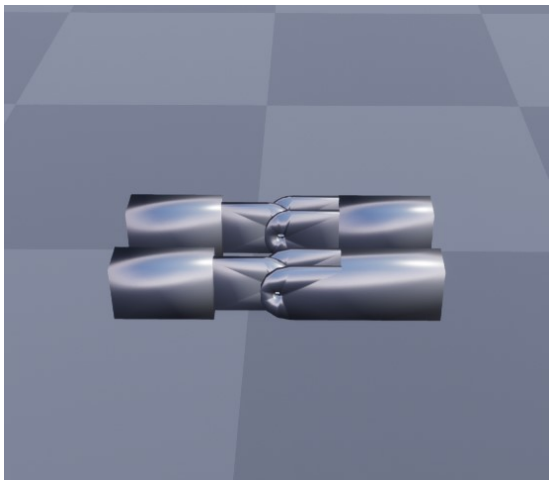
Obrázek 3.7: Zapnutí fyziky všech součástí

Vzhledem k tomu, že se jedná o herní fyziku, ve které se používá zjednodušení tj. konvexní obálky. To znamená, že se jedná o aproximaci povrchu, která nemusí být ve všech místech přesná. Konvexní obálka se používá právě u simulací a her pro urychlení výpočtu. Důsledkem aproximace by se mohly povrchy překrývat a tedy způsobovat zmíněnou

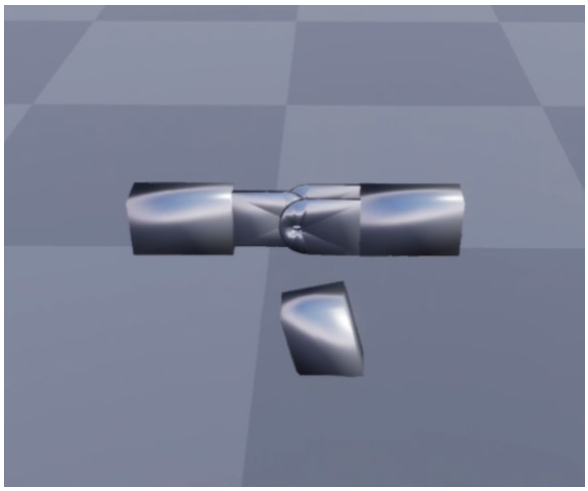
nestabilitu. Pro otestování funkčnosti fyziky byl vymyšlen jednoduchý kontrolní model, který zároveň vyzkouší kinematickou vazbu, vytvářenou příkazem `Joint`.

Test spočívá v tom, že jsou vytvořeny dva modely kloubů (obrázek 3.8). Jeden kloub je ze dvou částí, které jsou spojeny rotační kinematickou vazbou. Druhý kloub je sestaven z pěti kvádrů. Kvádry jsou spojeny fixní (vetknutí) kinematickou vazbou do stejných součástí, ze kterých je vytvořen první kloub. Tyto dvě součásti jsou navzájem spojeny rotační vazbou. Smyslem bylo vyzkoušet, zda se konvexní obálky nepřekrývají a nevytváří problém v působení fyziky.

Během testu byla spouštěna fyzika jednotlivých částí a bylo sledováno co nastane. Po spuštění fyziky jedné součásti prvního kloubu tato součást okamžitě propadla podlahou (obrázek 3.9).

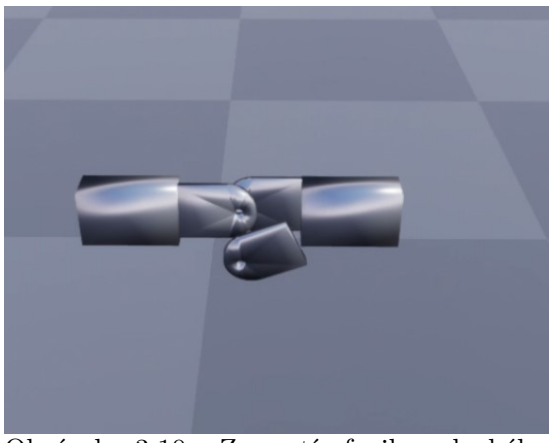


Obrázek 3.8: Začátek pokusu o kinematickou vazbu

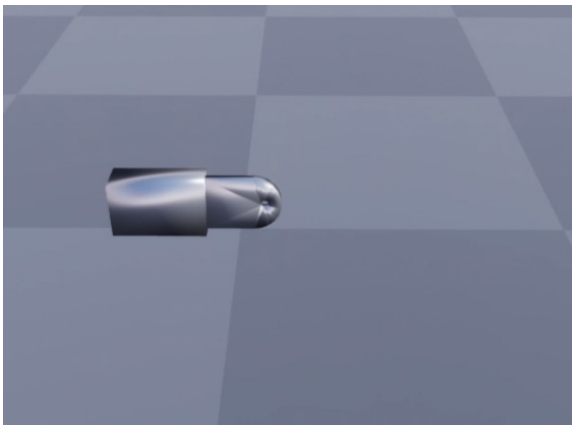


Obrázek 3.9: Zapnutí fyziky prvního kloubu

Při spouštění fyziky u druhého kloubu nejprve upadla jedna část, která byla přichycena fixní kinematickou vazbou (obrázek 3.10). Po zapnutí fyziky další části druhého kloubu zmizí polovina kloubu (obrázek 3.11).



Obrázek 3.10: Zapnutí fyziky druhého kloubu 1.část



Obrázek 3.11: Zapnutí fyziky druhého kloubu 2.část

Tento test neměl uspokojivé výsledky. Po konzultaci s vývojářem prostředí Ing. Luborem Zháňalem Ph.D., který se podílí na vývoji prostředí Simulink 3D Animation. Bylo

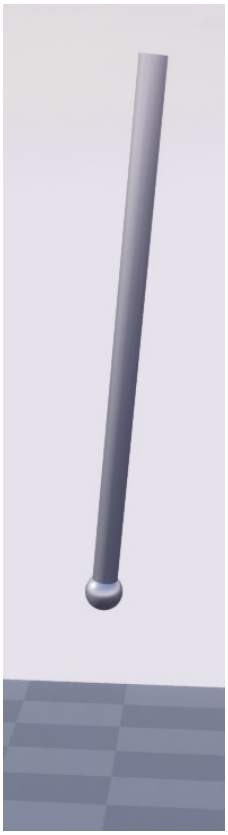
zjištěno, že fyzika funguje pouze na izolované části tělesa a nikoliv pro model jako celek. Také bylo od něj zjištěno, že funkce`Joint` nefunguje správně a její budoucnost je nejistá. Z tohoto důvodu by neměla být používána.

Spojení kloubů

Při hledání způsobu, jak sestavit robota, byla objevena funkce `Joint`. Vypadalo to, že podstatný problém jak "dát robota dohromady"je tímto vyřešen. Avšak se ukázalo, že nelze zvolit bod, kde budou působit vazby. Funkce je navíc v dokumentaci popsána poměrně nedostatečně, tím pádem nebylo možné jednoznačně určit, zda lze touto funkcí nastalý problém vyřešit. Na obrázku 3.6 je příklad popisu některých funkcí v dokumentaci programu.

Kinematickou vazbu se podařilo vytvořit jedině tak, že všem součástem v Inventoru byl umístěn počátek souřadného systému do požadovaného bodu rotace. Po konzultaci s Ing. Luborem Zháňalem Ph.D. byly kinematické vazby odstraněny. Důvod odstranění je na konci sekce Fyzika 3.1.3.

Kinematická vazba byla nahrazena vazbou parent-child, kde parent ovládá všechny následující child-komponenty. Na obrázku 3.12 je zkušební model vazby. Každý child má svůj lokální souřadný systém, ve kterém se pohybuje. Celková struktura parent-child robota je na obrázku 3.3.



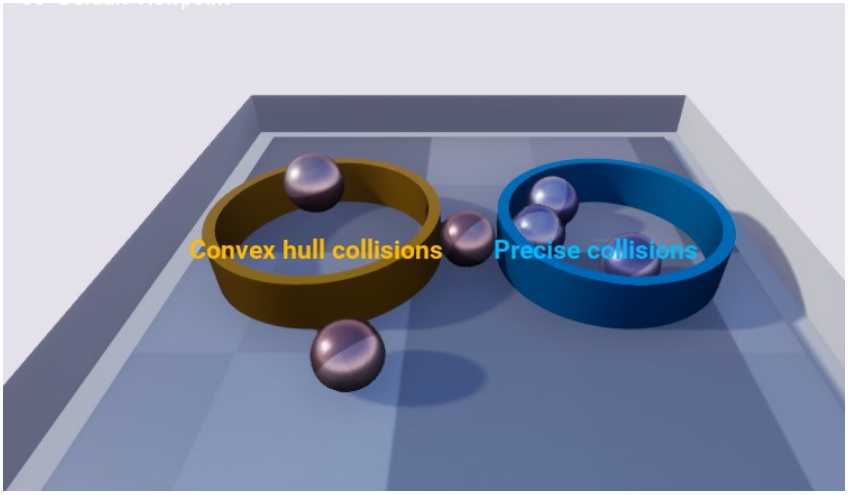
Obrázek 3.12: Vytvoření jednoduché rotační vazby parent-child

Kontakt

U každé soustavy se tělesa navzájem dotýkají. Není možné, aby se tělesa při kontaktu prolínala. Proto jsou v F3D implementovány funkce, které mají vytvořit realistický kontakt. Funkce jsou v zásadě dvě.

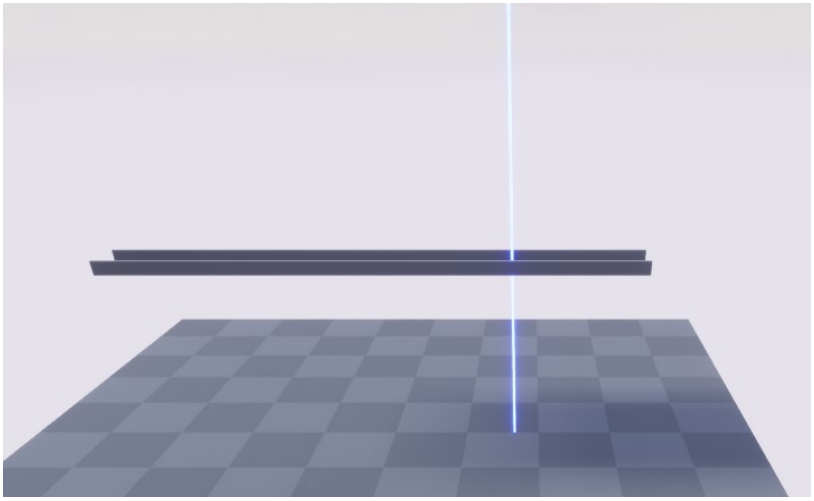
První je `PreciseContacts`, která vytvoří docela přesvědčivý kontakt mezi tělesy. Například odvalování kuliček kolem nějaké překážky 3.13. Bohužel tato funkce je neslučitelná se zapnutou fyzikou, což v některých případech vadit nemusí, ale zde je to vážný problém.

Druhou funkcí je `Contact`. Funkce Contact udělá konvexní obálku kolem tělesa, na které vyhodnocuje interakce s ostatními tělesy. Tento mód vizuálně nevypadá moc přirozeně, protože tělesa mohou "přeskočit"překážku jak je vidět na obrázku 3.13.

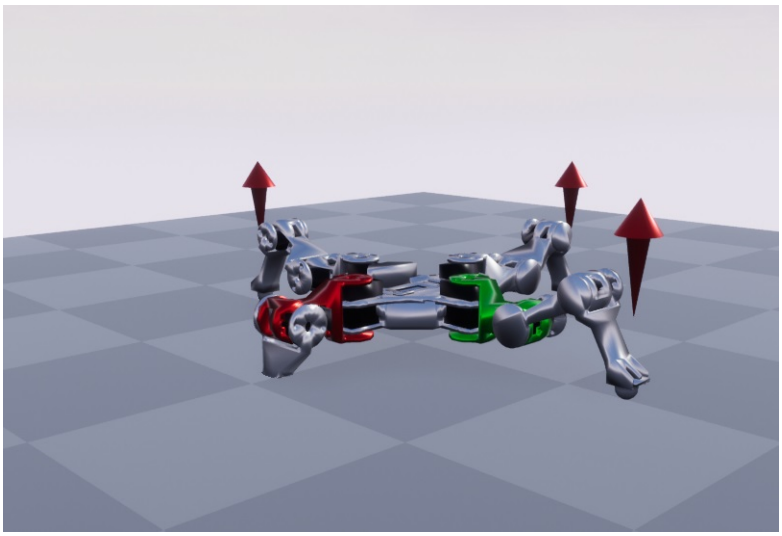


Obrázek 3.13: Porovnání módů kontaktu

Protože kontakt na modelu robota nefungoval správně, obrázek 3.15, byl vyzkoušen na jednodušším modelu. Model zahrnuje rampu, kterou je možné natáčet a kuličku, která je upuštěna z různých výšek. Model ukázal, že nezáleží na natočení rampy, ale spíš na výšce upuštění resp. na rychlosti kuličky (obrázek 3.14).



Obrázek 3.14: Jednoduché vyzkoušení kontaktu

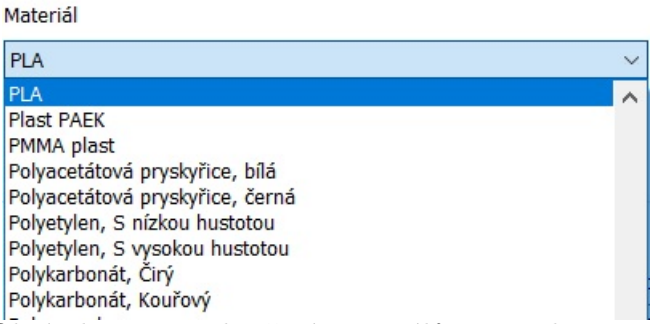


Obrázek 3.15: Nefunkční zero-crossing detection

Hmotnost

Hmotnost je nezbytný parametr pro výpočet kinematiky čtyřnohého robota, proto ji nelze nezmínit, když nefunguje jak má. Každý actor má ve své struktuře vlastnost `mass`, což na první pohled nevypadá nějak zvláštně. V dokumentaci je u proměnné Mass pouze uvedeno, že je v kilogramech, pořád nic neobvyklého.

Problém nastane při zápisu do proměnné, protože se do ní nic nepropíše. Navíc když spustíte simulaci, tak jako hmotnost je zapsané náhodné číslo. Snaha ovlivnit hmotnost změnou hustoty částí robota v Inventoru, se bohužel nesetkala s úspěchu. Jediné čím se podařilo pohnout hmotností bylo změnění měřítka, ale to moc nepomohlo.



Obrázek 3.16: Vyzkoušení materiálů s jinou hustotou

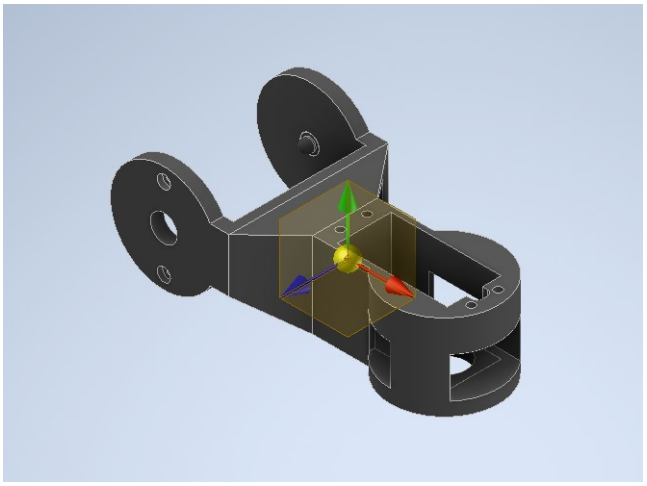
Těžiště

Těžiště má každý hmotný objekt a jeho znalost je nezbytným parametrem k výpočtu celkového těžiště soustavy nebo k výpočtu kinematiky. Proto bylo důkladně zjišťováno, jak funguje práce s těžištěm v prostředí F3D. Nejprve to vypadalo, že by si Matlab těžiště mohl dopočítat, ale to se ukázalo jako mylný předpoklad.

Tento předpoklad byl vyvrácen jednoduchým testem. Byl vytvořen objekt s jednoduchou geometrii. Jednoduchá geometrie je myšleno geometrie, kterou lze vytvořit v F3D

příkazem a parametry. Byl vytvořen válec, ale i ten měl defaultní hodnotu parametru `CenterOfMass` [0,0,0].

Souřadnice těžiště jednotlivých částí robota byly zjištěny v Inventoru 3.17 a do F3D byly vloženy.



Obrázek 3.17: Zjištění těžiště v Inventoru

Tření

Při vyvíjení algoritmu chůze je potřeba, aby krom pohybu nohou se pohybovalo i tělo. Tělo se většinou pohybuje, když nohy jsou pevně na zemi. V prostředí Simulink 3D Animation byl dělán posun těla a bylo potřeba, aby nohy stáli na místě. Byla zvyšována hodnota proměnné `Friction`, zajišťující tření, až do hodnoty 1, ale beze změny.

Bylo to otestováno na posledních částech nohou "packách"robota. Packám byla zapnuta fyzika a kontakt. Po nezdaru byl pokus rozšířen ještě na celé tělo robota, na které bylo aplikováno zatížení. Po několika pokusech s hodnotami tření v intervalu <0,1> nebyly pozorovány, žádné změny v chování jednotlivých částí.

4 Algoritmus chůze

V této kapitole je popsáno, jaké procesy nastanou při spuštění simulace. Dále je zde popsán způsob zjištění délky kroku a jeho výběr. Kapitola se také zmiňuje o výpočtu těžiště robota. V druhé části kapitoly je popsán jednoduchý algoritmus chůze čtyřnohého robota, který je testován v simulaci v programu F3D.

4.1 Návrh algoritmu chůze čtyřnohého robota

V kapitole 3 je popsáno, jakým způsobem byl sestaven model robota. Po několika vrstvách struktury parent - child je reference na koncové části tak dlouhá, že znepřehledňuje kód. Z tohoto důvodu byly v systémové funkci `UserData` vytvořeny struktury do kterých byly uloženy funkce pro snazší manipulaci.

Po načtení 3D Foundation Viewer (4.1 - načtení GUI) jsou načtena i dvě tlačítka. První je modré a je označeno GP (Go to position), po kliknutí na něj se zobrazí dialog, který umožní vložení požadovaných souřadnic, kam má robot dojít. Druhé tlačítko je označeno jako IR (Import rotations), po kliknutí zobrazí dialog umožňující vložení natočení jednotlivých kloubů. Toto tlačítko slouží pro vizuální kontrolu případných výpočtů.

Základní struktury, proměnné a meze rozsahů vytváří funkce `Init`, ze schématu 4.1, hned po spuštění programu. Při startu proběhne i funkce `Init kinematic`, která vypočte počáteční polohu těžiště robota. Ve schématu 4.1 je graficky znázorněno pořadí inicializačních kroků.

Obrázek 4.1: Schéma init programu

4.1.1 Délka kroku

Délka kroku čtyřnohého robota je důležitý parametr, který je podstatný pro stabilitu robota. Určuje totiž, jakým způsobem se robot může dostat přes překážky. Pokud je délka kroku malá, robot má vyšší stabilitu, ale jeho schopnost překračovat překážky se značně sníží. Při delším kroku se stabilita robota zmenšuje. Délka kroku se většinou určuje experimentálně nebo ze znalosti geometrie robota.

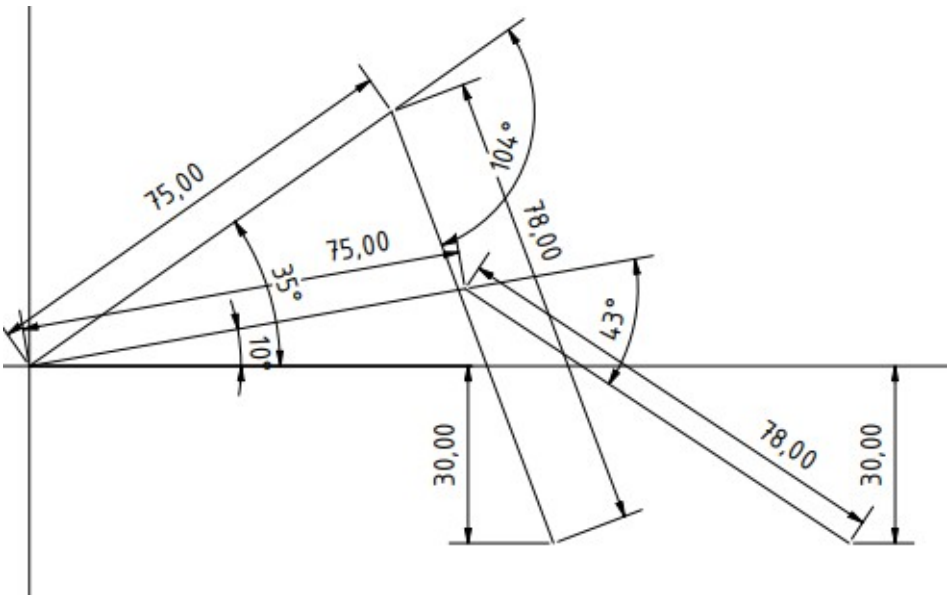
Délka kroku pro tento algoritmus bude odvozena z geometrie robota. Nejprve bude vypočten úhel natočení třetího kloubu v závislosti na druhém, za podmínky, že robot zůstane ve stejné výšce. Pro tento výpočet byla z geometrie odvozena rovnice 4.1. Správnost výpočtu byla ověřena nakreslením geometrie (obrázek 4.2), v programu Autodesk Inventor.

$$\beta = cos^{-1} \frac{Z_0 + l_2 \cdot sin(\alpha)}{l_3 + Z_p}, \tag{4.1}$$

kde

- β je úhel natočení třetího kloubu
- α je úhel natočení druhého kloubu
- Z_0 je výška těla robota nad zemí
- l_2 je délka druhého článku nohy
- l_3 je délka třetího článku nohy
- Z_p je výška koncového senzoru došlap

Na obrázku 4.2 je natočení druhého a třetího kloubu nohy s maximálním a minimálním úhlem, který je fyzicky možný, aby noha robota dosáhla na zem. Délka nohy v mezních stavech je důležitým parametrem, pro další výpočet délky kroku.



Obrázek 4.2: Ověření maximální délky kroku

Rozsah absolutního natočení druhého kloubu nohy je od -35°do -10°. Základní natočení druhého kloubu nohy je -30°, toto natočení odpovídá zadanému úhlu 0° v simulaci. Pro dosažení absolutního natočení z intervalu <-35°; -10°> je tedy nutné v simulaci zadat hodnotu z intervalu <-5°; 20°>.

Pro interval natočení druhého kloubu <-35°; -10°>, existuje odpovídající interval natočení třetího kloubu. Po dosazení minimálního a maximálního natočení druhého kloubu do rovnice 4.1 je spočten interval <78,4°; 54,67°> absolutního natočení třetího kloubu nohy robota.

Defaultní natočení třetího kloubu je 90°, proto je nutné před zapsáním hodnoty natočení do kloubu ji nejprve přepočítat (rovnice 4.2), aby výsledná hodnota byla tou požadovanou.

$$\beta_s = 90° - \beta_p \tag{4.2}$$

kde

- β_p je požadovaný úhel natočení třetího kloubu
- β_s je přepočtený úhel natočení třetího kloubu

Když jsou známy natočení druhého a třetího kloubu je možné spočítat aktuální délku nohy. K tomu je užita rovnice 4.3.

$$L_n = L_1 + L_2 \cdot cos(\alpha) + L_3 \cdot sin(\beta) \tag{4.3}$$

kde

- β je úhel natočení třetího kloubu
- α je úhel natočení druhého kloubu
- L_n je celková délka nohy
- L_1 je délka prvního článku nohy
- L_2 je délka druhého článku nohy
- L_3 je délka třetího článku nohy

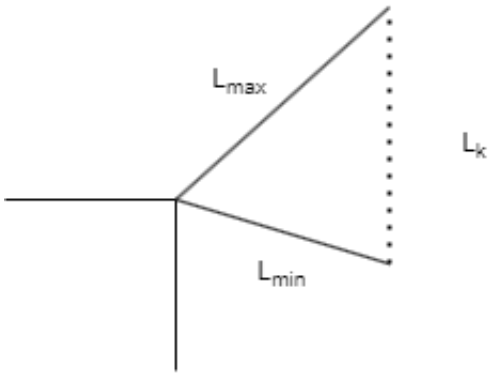
Z rovnice 4.3 pro výpočet délky nohy je vypočtena maximální a minimální délka. Po řadě pokusů byl vybrán rozsah pohybu prvního kloubu na 50°. Ze znalosti geometrie (obrázek 4.3) a kosinové věty (rovnice 4.4) je možné vypočítat vzdálenost L_k mezi maximálním a minimálním natažením nohy.

$$L_k = \sqrt{L_{max}^2 + L_{min}^2 - 2 \cdot L_{max} \cdot L_{min} \cdot cos(\gamma)} \tag{4.4}$$

kde

- γ je rozsah pohybu prvního kloubu

- L_max je maximální délka nohy
- L_min je minimální délka nohy
- L_k je délka kroku



Obrázek 4.3: Vzdálenost mezi koncovými efektory

Výběr kroku

Pokud je vzdálenost do cílové pozice větší než délka možného kroku nohy, je použit fixní krok. Tento typ kroku je nastaven jako nejoptimálnější varianta z fyzicky možného rozsahu. Pokud se robot blíží k cíli a požadovaná vzdálenost je menší než délka kroku, tak je zvolen proměnný krok.

U proměnného kroku je vzdálenost od cíle sledována v každé iteraci posunu těla. Pohyb je zastaven, když je rozdíl mezi aktuální a požadovanou vzdáleností polovina velikosti iterace. Velikost iterace závisí na zvolené délce kroku a počtu jeho dělení.

4.1.2 Výpočet těžiště

Pro stabilitu čtyřnohého robota je podstatné, aby se těžiště robota pohybovalo v oblasti stability (obrázek 2.3.1). Pokud robot oblast stability opustí, může spadnout a poškodit se.

Výpočet těžiště složitějších geometrii může být problém, proto bylo těžiště robota vypočteno po částech. U každé části robota je známo těžiště v jeho lokálním systému. Nejprve bude vypočteno těžiště každé nohy v souřadném systému celé nohy. Souřadný systém celé nohy je umístěn v prvním kloubu a je totožný se souřadným systémem prvního kloubu. Ze znalosti geometrie a pozice těžiště byly odvozeny rovnice 4.5, 4.6,4.7, 4.8, 4.9, pro těžiště každé části přepočtené do souřadného systému celé nohy.

$$CoM.X3(i) = (L_1 + L_2 \cdot cos(\phi_2(i)) + CoM_{J3}(1) \cdot sin(\phi_3(i)) \cdot cos(\phi_1(i))) \tag{4.5}$$

$$CoM.X3s(i) = (L_1 + L_2 \cdot cos(\phi_2(i)) \cdot cos(\phi_1(i))) \tag{4.6}$$

$$CoM.X2(i) = (L_1 + CoM_{J2}(1) \cdot cos(\phi_2(i)) \cdot cos(\phi_1(i))) \tag{4.7}$$

$$CoM.X2s(i) = L_1 \cdot cos(\phi_1(i)) \tag{4.8}$$

$$CoM.X1(i) = CoM_{J1}(1) \cdot cos(\phi_1(i) \tag{4.9}$$

kde

- ϕ_1, ϕ_2, ϕ_3 jsou aktuální natočení kloubů
- L_1, L_2, L_3 jsou délky článků nohy
- $CoM.X3(i)$ CoM je název struktury obsahující data, X3(i) znamená x-ová souřadnice třetí části i-té nohy
- $CoM_{J2}(1)$ je x-ová souřadnice těžiště druhé části

Následně jsou všechna dílčí těžiště přepočítána do souřadného systému robota. To znamená posun u 1. a 3. nohy o posun v ose X o délku $\pm$ LB a u 2. a 4. nohy ve směru osy Y o vzdálenost $\pm$ LB.

Pro získání těžiště celé nohy, je třeba udělat vážený průměr ze všech vypočtených těžišť (rovnice 4.10 a 4.11).

$$CoM.Xt(i) = \frac{\sum_{i=1}^{n} m_k \cdot X_i}{m_k} \tag{4.10}$$

$$CoM.Yt(i) = \frac{\sum_{i=1}^{n} m_k \cdot Y_i}{m_k} \tag{4.11}$$

kde

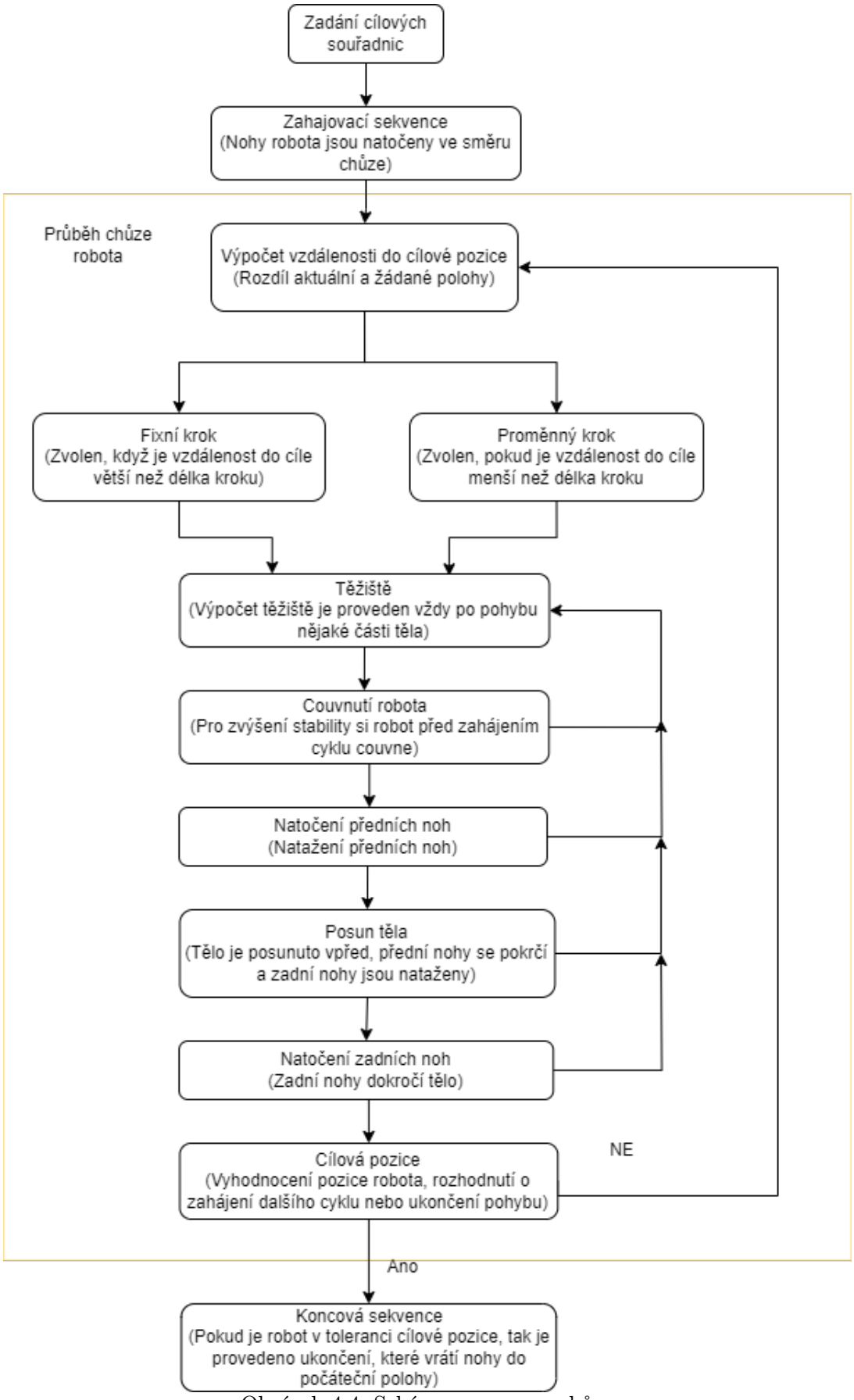
- $Xt(i), Yt(i)$ je x-ová a y-ová složka těžiště i-té nohy
- m_k jsou hmotnosti jednotlivých komponent
- X_i,Y_i jsou složky těžišť komponent nohy

Pro zjištění celkového těžiště stačí sečíst těžiště jednotlivých noh a těžiště těla robota, které má těžiště téměř ve středu.

4.1.3 Chůze čtyřnohého robota

Dle zadání byl vytvořen algoritmus chůze robota. Protože se jedná o simulaci jsou natočení okamžitá, aby se pohyb robota více podobal realitě, tak jsou natočení robota postupně inkrementována z aktuální hodnoty natočení do požadované hodnoty.

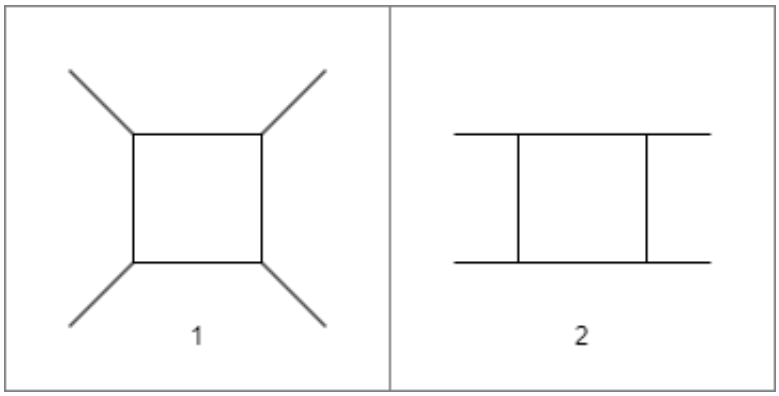
Celý program běží v jedné nadřazené smyčce `while`, která je přerušena až je robot na požadované pozici. V nadřazené smyčce `while` je `if` cyklus, který vyhodnocuje vyvolané změny. V jedné z podmínek, je podřazená `while` smyčka, se stavovým automatem se stavy `init`, `cyclic` a `exit`. Tato podřazená smyčka provádí požadované změny a je použita vícekrát během pohybu.



Obrázek 4.4: Schéma programu chůze

Zahajovací sekvence

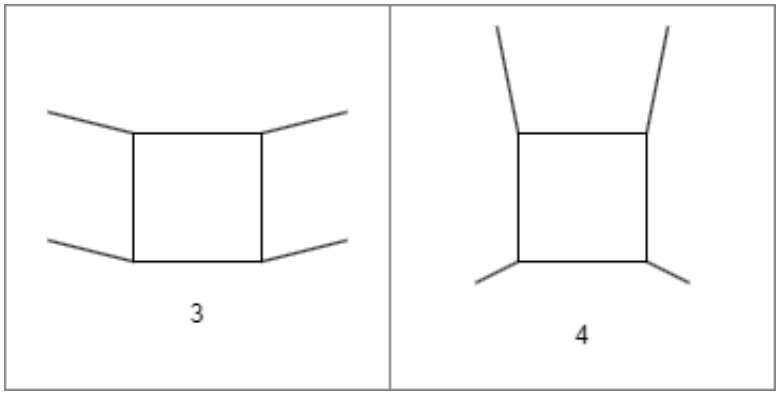
Robot je v simulaci v základní poloze (obrázek 4.5 - 1.část), což znamená, že všechny relativní natočení kloubů jsou nulová. Po zadání cílových souřadnic ve vizualizaci je určena vzdálenost cílové pozice. Z rozdílu požadované a aktuální pozice robot obdrží vzdálenost a směr požadované pozice. Kladný směr pohybu je určen ve směru os X a Y. Po zjištění směru pohybu robot zahájí přípravnou sekvenci, což je stav `init`, ve které se nohy natočí do počátečních pozic pro chůzi v požadovaném směru (obrázek 4.5 - 2.část). Na konci počátečního nastavení je aktualizována vzdálenost do požadovaného bodu.



Obrázek 4.5: Posun těla 1-2

Průběh chůze robota

Po počátečním natočení nohou robota nastane stav `cyclic`, který zajišťuje chůzi robota, průběh tohoto stavu je zobrazen na obrázku 4.6 a na obrázku 4.7. Úkony tohoto stavu jsou popsány na obrázku 4.4 jako průběh chůze robota.

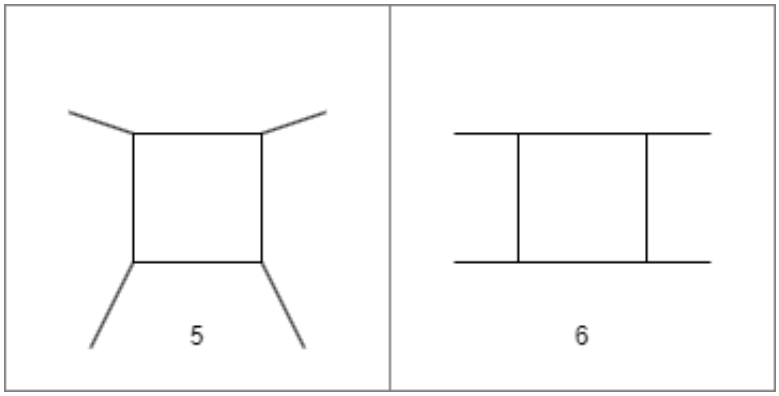


Obrázek 4.6: Posun těla 3-4

Na začátku každého cyklu je aktualizována vzdálenost k cílové pozici, která je následně porovnána se zadanou délkou kroku viz. obrázek 4.4. Za předpokladu, že je rozdíl vzdálenosti poloh větší než délka kroku je vybrán krok fixní, pokud je vzdálenost mezi body menší, tak je zvolen proměnný krok.

Po počáteční fázi je robot v poloze zobrazené na obrázku 4.5 - 2. část. Každý cyklus začíná malým posunem těla robota zpět (4.6 - 3.část), aby bylo těžiště posunuto hlouběji

do oblasti stability 2.3.1. Po dokončení tohoto posunu jsou nataženy přední nohy do svého maxima (4.6 - 4.část). Po každém pohybu končetiny následuje přepočet těžiště. Dále se vpřed posune tělo robota (obrázek 4.7 - 5.část), bez toho aniž by byly zdviženy nohy ze země. Poslední částí cyklu je dokročení zadních noh (obrázek 4.7 - 6.část), aby mohl nastat nový cyklus.

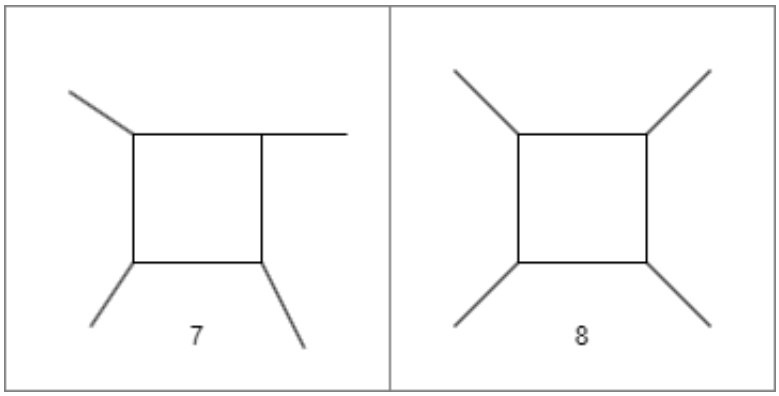


Obrázek 4.7: Posun těla 5-6

Na závěr cyklu kroku, je vypočtena vzdálenost do cílové pozice a je rozhodnuto, jestli nastane další cyklus nebo bude provedena koncová sekvence.

Koncová sekvence

Když je tělo robota v cílové pozici, tak je stav změněn na `exit`. V tomto stavu nastane konečné vyrovnání robota do počáteční pozice (obrázek 4.8). Po proběhnutí stavu `exit`, je přerušena podřazená `while` smyčka, která trvala po dobu chůze v daném směru. Pokud je požadovaná poloha shodná s aktuální polohou robota v obou osách, tak je pohyb vyhodnocen jako dokončený a je přerušena i nadřazená `while` smyčka.



Obrázek 4.8: Posun těla 7-8

5 Závěr

Cíle této práce byly rešerše v oblasti chůze čtyřnohého robota, vytvoření modelu čtyřnohého robota a vytvoření algoritmu chůze čtyřnohého robota a otestování v simulaci.

Byl vytvořen model čtyřnohého robota v prostředí Simulink 3D Animation. Byly použity modely dílů z poslední verze robota [1]. Při vytváření modelu bylo objeveno několik závažných problémů a nedostatků v použitém prostředí F3D.

Bylo zjištěno, že fyzika v tomto prostředí je funkční pouze u jednoduchých objektů, jako je třeba krychle nebo koule. U složitějších objektů není možné zapnutí fyziky, protože se model okamžitě rozpadne jak je vidět na obrázku 3.7.

Dalším z nedostatků F3D je, že není možné vytvořit libovolně kinematickou vazbu. Pro vytvoření kinematické vazby je v dokumentaci popsán příkaz `Joint`, ale tento příkaz nedokáže vždy vytvořit funkční kinematickou vazbu, jak lze vidět na obrázcích 3.8, 3.9, 3.10 a 3.11. Tento problém byl vyřešen použitím vazby parent - child.

U složitějších geometrií jako je čtyřnohý robot je problematický zejména kontakt s podložkou. Jako první byl vyzkoušen kontakt mezi robotem a podložkou, přičemž koncové senzory měly zapnutou fyziku a kontakt, ale i tak robot stále prošlapával.Druhým způsobem bylo použití `preciseContact`, který podle ukázkového příkladu vypadal velice realisticky 3.13, avšak ani tento kontakt nefungoval.

U některých problému se nedal jednoznačně určit zda se jedná o skutečný problém nebo jen o špatně užitou syntaxi, protože některé funkce neměly popis v dokumentaci programu, jako např. na obrázku 3.6.

Byl vytvořen jednoduchý (základní) algoritmus chůze pro otestování modelu robota a praktické ukázce prostředí F3D. Tento jednoduchý program slouží jako vzorová ukázka syntaxe.

Výše popsané nedostatky byly konzultovány s developerem prostředí F3D a řada problémů byla potvrzena. Z problémů a nedostatků, které tato práce zmínila vyplývá, že další rozšiřování simulací v prostředí Simulink 3D Animation je zbytečné. Minimálně do doby, než budou výše zmíněné problémy vyřešeny. V současné době tento program řeší danou problematiku nedostatečně, proto se nabízí využití jiných programů, například Simscape. Simscape je software od MathWorks, navržený pro simulaci fyzikálních komponent a pro různé další reálné simulace soustav.

Toto je jedna z prvních prací zabývající se tímto softwarem. V této práci byly odhaleny některé problémy prostředí F3D, které nyní mohou být vyřešeny. Poznatky této práce tedy mohou vést ke zlepšení tohoto softwaru.

Reference

[1] KRÁLÍK, Jan. *Implementace algoritmů kinematiky pro čtyřnohý chodící robot*. Vysoké učení technické v Brně. Fakulta strojního inženýrství, 2018.

[2] *Boston Dynamics' 'Android of robots' vision starts with launching 1000 robot dogs in 2019 | Packt Hub*. [B.r.]. Dostupné také z: https://hub.packtpub.com/boston-dynamics-android-of-robots-vision-starts-with-launching-1000-robot-dogs-in-2019/.

[3] *Robotický mezek v námořní pěchotě doběhal | ArmadniNoviny.cz*. [B.r.]. Dostupné také z: https://www.armadninoviny.cz/roboticky-mezek-v-namorni-pechote-dobehal.html.

[4] *» Robopes BigDog dostal ruku, s níž dokáže odhodit betonovou tvárnici*. [B.r.]. Dostupné také z: https://www.t3mag.cz/robopes-bigdog-dostal-ruku-s-niz-dokaze-odhodit-betonovou-tvarnici.

[5] CORKE, Peter. *Robotics, Vision and Control: Fundamental Algorithms in MATLAB® Second, Completely Revised, Extended and Updated Edition*. Sv. 118. Springer International Publishing AG, 2017. ISBN 9783319544120.

[6] *Denavit–Hartenberg parameters - Wikipedia*. [B.r.]. Dostupné také z: https://en.wikipedia.org/wiki/Denavit%E2%80%93Hartenberg_parameters.

[7] *Robotics/Types of Robots/Walkers - Wikibooks, open books for an open world*. [B.r.]. Dostupné také z: https://en.wikibooks.org/wiki/Robotics/Types_of_Robots/Walkers.

[8] *Atlas Gets a Grip | Boston Dynamics - YouTube*. [B.r.]. Dostupné také z: https://www.youtube.com/watch?v=-e1_QhJ1EhQ.

[9] *Čtyřnohý robot JaromírFourlegged walking robot Jaromír » MechLab*. [B.r.]. Dostupné také z: http://mechlab.fme.vutbr.cz/veda-a-vyzkum/ctyrnohy-robot-jaromir/.

[10] ŠVEHLÁK, Michal. *Návrh konstrukce experimentálního kráčejícího robotu*. 2004.

[11] SVĚTLÍK, Miroslav. *Kinematické řízení a vizualizace virtuálního prototypu čtyřnohého kráčejícího robotu*. 2006.

[12] GOGOLA, Marek. *KONSTRUKCE MECHANICKÉ ČÁSTI SERVOPOHONU*. 2007.

[13] KONVIČNÝ, Jiří. *Návrh mechanické části servomechanismu*. 2007.

[14] SZTWIERTNIA, Václav. *VYUŽITÍ MEMS AKCELEROMETRŮ PRO STABI- LIZACI CHODÍCÍHO ROBOTU*. 2007.

[15] *Řídicí jednotka čtyřnohého robotu na bázi mikrokontroléru PIC; Ing. Daniel Youssef (2012 - 61597) – VUT*. [B.r.]. Dostupné také z: `https://www.vut.cz/studenti/zav-prace/detail/61597`.

[16] *Creating a proper f3d file - MATLAB Answers - MATLAB Central*. [B.r.]. Dostupné také z: `https://www.mathworks.com/matlabcentral/answers/1799145-creating-a-proper-f3d-file`.