VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INTELIGENTNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INTELLIGENT SYSTEMS

ŘÍZENÍ POHYBU ROBOTA TYPU HEXAPOD

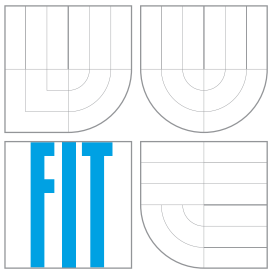
DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR

Bc. MAREK ŽÁK

BRNO 2015

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INTELIGENTNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INTELLIGENT SYSTEMS

ŘÍZENÍ POHYBU ROBOTA TYPU HEXAPOD

HEXAPOD ROBOT MOVEMENT CONTROL

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR

Bc. MAREK ŽÁK

VEDOUCÍ PRÁCE
SUPERVISOR

Ing. JAROSLAV ROZMAN, Ph.D.

BRNO 2015

Abstrakt

V této práci je popsána problematika kráčivých robotů, jejich rozdělení, řízení a konstrukce. Jsou popsány nejznámější pohybové algoritmy a jejich grafická reprezentace. Dále jsou uvedeny příklady existujících kráčivých robotů. V práci jsou také popsány změny na robotu hexapod vlastní konstrukce, jeho hardwarové a softwarové vybavení. Robot je řízen z grafického uživatelského rozhraní, které zobrazuje data ze všech senzorů, vizualizuje pozice končetin a umožňuje tvorbu uživatelských pohybových algoritmů a jejich následnou simulaci.

Abstract

This thesis discusses walking robots issues, their classification, management and construction. There are listed the most famous motion algorithms and their graphical representation. Examples of existing walking robots are also mentioned in this thesis. There are also described modifications of hexapod robot, its hardware and software. The robot is controlled through graphical user interface, which displays data from all sensors, visualises positions of all legs and allows the creation of user defined gaits and its simulations.

Klíčová slova

Hexapod, kráčející robot, kráčivé podvozky, algoritmy chůze, konstrukce hexapodu.

Keywords

Hexapod, walking robot, walking chassises, walking gaits, hexapod construction.

Citace

Marek Žák: Řízení pohybu robota typu hexapod, diplomová práce, Brno, FIT VUT v Brně, 2015

Řízení pohybu robota typu hexapod

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně pod vedením inženýra Jaroslava Rozmana, Ph.D.

.....
Marek Žák
26. května 2015

Poděkování

Chtěl bych tímto poděkovat svému vedoucímu, Ing. Jaroslavu Rozmanovi, Ph.D., který mi poskytl odbornou pomoc během tvorby diplomové práce.

Obsah

Úvod

Robotika je v dnešní době velmi aktuálním odvětvím průmyslu. Lidé se snaží najít nová řešení, která by umožnila přenechat fyzicky a časově náročné či pravidelně se opakující pracovní činnosti strojům. Jedná se především o manipulaci s těžkými břemeny za pomoci robotických paží nebo o automatizovanou výrobní linku. Roboty jsou konstruovány na pomoc člověku při práci v nedostupných nebo nebezpečných místech. Své uplatnění nacházejí v odvětvích jako je průmysl, lékařství či vojenství.

Jsou hledány různé technologie, které by nahradily pohybový aparát člověka, a které by se dokázaly vyrovnat nebo alespoň přiblížit organickým kostrám. Ať už se jedná o robotické protézy, které by nahradily amputované končetiny či orgány, jako například mechanické srdce, vždy je velice komplikované nahradit část organickou částí technologickou.

Pro zajištění vysoké mobility strojů v nedostupných terénech jsou konstruovány robotické podvozky, které používají ke svému pohybu kola či pásy nebo jsou inspirovány přírodou. Jedná se například o robotické podvozky, které pro svůj pohyb využívají kráčivých končetin, případně se pohybují valivým nebo plazivým pohybem podobně jako zvířata.

Mne zaujaly přírodou inspirované robotické podvozky, které se dají použít v nerovném, členitém terénu, kam se kolové či pásové podvozky nehodí. Jedním z typů takových podvozků je hexapod, tedy podvozek složený z těla a šesti končetin. Podvozek typu hexapod se pohybuje za pomoci chůze. Výhodou oproti kolovým a pásovým podvozkům je možnost pokračovat v pohybu při poškození až dvou končetin podvozku.

Existuje ovšem celá řada dalších kráčivých podvozků, které postupně hledají své uplatnění. Značnou část oblasti kráčivých podvozků zabírají systémy jejich řízení a ovládání. Stále častěji se objevují výzkumy na inteligentní řízení kráčivých podvozků. Jsou hledány nové metody, které by umožnily kráčivým robotům samostatnou existenci bez nutnosti zásahu člověka. V poslední době je také častým tématem inteligence robotů, tedy schopnost reagovat na změny v prostředí a hledat optimální způsob řešení problémů. U kráčivých robotů jde tato inteligence ještě o něco dále. Je snaha vytvořit roboty, které se budou schopny samy učit chodit, k čemuž se využívají například neuronové sítě.

Tato práce se věnuje kráčivým podvozkům a jejich vlastnostem se zaměřením na hexapody. V první kapitole se nachází stručný úvod ke kolovým, pásovým a kráčivým podvozkům a jejich srovnání. Druhá kapitola podrobněji rozebírá vlastnosti kráčivých podvozků se zaměřením na hexapody, jejich možná rozdělení, inspiraci v přírodě, jejich konstrukci a řízení. Dále jsou nastíněny některé pohybové algoritmy a jejich grafické reprezentace. Kapitola se také krátce věnuje řízení kráčivých podvozků užitím neuronových sítí. Závěrem jsou uvedeny některé příklady existujících kráčivých robotů. Kapitola třetí popisuje hexapod vlastní konstrukce a několik úprav, vedoucích ke zlepšení vlastností robotu a k rozšíření senzorického systému. Čtvrtá kapitola se věnuje hardwarovému a softwarovému vybavení robotu. Popsány jsou vlastnosti jednotlivých komponent a senzorů a způsob jejich použití. Dále jsou rozebrány jednotlivé prvky uživatelského rozhraní jako je vizualizace pozic

končetin robotu nebo tvorba uživatelských pohybových algoritmů a jejich simulace či přímé řízení robotu. Pátá kapitola pojednává o rozšířeních a vylepšeních, která byla realizována nad rámec zadání práce. Šestá a poslední kapitola se věnuje testování systému robotu, senzorickému systému, softwarových prvků a pohybových vlastností robotu.

Výsledný podvozek slouží k testování a ověřovaní algoritmů a hypotéz, které se zabývají systémy šestinohých kráčivých robotů. Tato platforma disponuje velkým množstvím senzorů, které umožňují pohyb robotu i ve velice členitém terénu. Robot je možné monitorovat a řídit pomocí uživatelského rozhraní, které zobrazuje data ze všech senzorů, vizualizuje pohyb končetin robotu a umožňuje tvorbu uživatelských pohybových algoritmů a jejich simulaci.

Kapitola 1

Podvozky

Proč má smysl studovat kráčivé podvozky? Nevystačíme si s podvozky kolovými a pásovými a jejich modifikacemi? Aby bylo takovéto otázky možné zodpovědět, musíme se nejprve na jednotlivé podvozky podívat ze všech stran. Musíme zvážit jejich výhody i nevýhody a uvážit i extrémní situace, do nichž se daný podvozek může dostat.

1.1 Kolové podvozky

Kolové podvozky využívají kolo pro transformaci rotačního pohybu v pohyb posuvný. Tyto podvozky mohou disponovat jedním nebo více koly, jsou relativně jednoduché a umožňují nám dosahovat vysokých rychlostí pohybu. To ovšem platí jen v ideálních terénních podmínkách, kdy je velikost nerovností na vozovce vůči velikosti kola malá nebo zanedbatelná. Běžné kolové podvozky navíc mohou uvíznout v měkkém nebo kluzkém terénu, případně zůstat viset na rámu podvozku.

1.2 Pásové podvozky

Pásové podvozky svým použitím navazují na podvozky kolové. Ty mají na kolech nasazeny pásy, které se mohou lišit podle terénu, ve kterém se mají pohybovat. Rychlost pásových podvozků je nižší než podvozků kolových, dosahují však lepších výsledků v členitějším prostředí, kde kolové podvozky selhávají, zejména protože styčná plocha s vozovkou je u pásových podvozků výrazně větší.

Obrázek 1.1: Kolový podvozek.[1]

Obrázek 1.2: Pásový podvozek.[2]

1.3 Kráčivé podvozky

Kráčivé podvozky se od obou předchozích příkladů liší prakticky ve všech aspektech. Podvozek není poháněn koly, pro svůj pohyb využívá přesouvání svých končetin - chůzi. Tento způsob pohybu skýtá nespočet výhod. Například kráčivé podvozky překonají velké terénní nerovnosti až do výšky zdvihu končetiny. Mohou překonat díry v terénu, do kterých by kolové či pásové podvozky zapadly. Dokáží dokonce pokračovat i při ztrátě nebo poškození některé z končetin a jsou jedním z nejuniverzálnějších podvozků. Tělo podvozku zůstává neustále v jedné rovině, což je vhodné pro umístění různých senzorů [35].

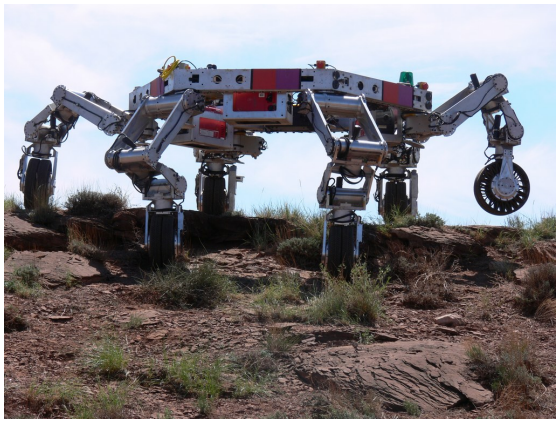
Na druhé straně použití kráčivého podvozku přináší mnoho komplikací. Konstrukce podvozku je výrazně složitější než u jiných typů. Kráčivé podvozky jsou energeticky náročnější, protože spotřebovávají energii neustále, i když se nepohybují, a jejich řízení vyžaduje složitější systémy.

Dále můžeme nalézt podvozky, které kombinují výhody různých typů podvozků, například kolových a kráčivých. Vždy ovšem dominuje jeden z přístupu a druhý pouze podvozek vylepšuje, zvyšuje jeho schopnost pohybu. Příkladem může být kráčivý bagr, který má kola připevněna k hydraulickým končetinám a může tak měnit jejich polohu. Nedokáže však chodit v pravém slova smyslu.

Jiným příkladem může být robot The ATHLETE Rover (The All-Terrain Hex-Limbed Extra-Terrestrial Explorer) [25] vyrobený NASA. Jedná se o šestinohou platformu se šesti stupni volnosti na končetinu. Motor v každém kloubu má výkon 745 W. Každá z končetin je dále zakončena kolem. První generace robotu byla vyrobena v roce 2005 a vážila 850 kg. Nosnost robotu je 300 kg. V rovném nebo málo členitém terénu robot využívá pohybu na kolech, v členitějším terénu může kráčet a překonávat překážky. Ke končetinám je navíc možné připojit pracovní nástroje jako vrták nebo klepeto a manipulovat s okolními předměty nebo vrtat do země.

Tato práce se dále bude věnovat kráčivým podvozkům, zvláště pak hexapodům, jejich modifikacím a řízení.

Obrázek 1.3: Kráčivý bagr. Díky svým končetinám může pracovat i v členitém terénu.[3]



Obrázek 1.4: The ATHLETE Rover, robot vyrobený NASA. Robot kombinuje výhody kolových a kráčivých podvozků. [25]

[1] Staženo z http://www.avtorinok.ru/photo/Komatsu_WA500_pic_46584.jpg, dne 4. 12. 2014.
[2] Staženo z http://www.dts-as.cz/obr_miru_4.jpg, dne 4. 12. 2014.
[3] Staženo z http://media.machinerypark.com/offer/images/c8/02/1280-960/db81ba3bf826526784.jpg, dne 4. 12. 2014.

Kapitola 2

Kráčivé podvozky

Tato kapitola se podrobněji zabývá kráčivými podvozky. Jsou zde nastíněna základní dělení podvozků a inspirace pro jejich výrobu. Kapitola dále popisuje konstrukci kráčivých podvozků a končetin. Jsou zde také uvedeny kinematické vlastnosti podvozků a způsoby reprezentace pohybových algoritmů.

2.1 Rozdělení kráčivých podvozků

Kráčivé podvozky můžeme dělit podle počtu končetin, podle použitého pohonu, podle počtu kloubů nebo podle pohybové techniky. Jednotlivé varianty lze libovolně kombinovat, čímž je možné dosáhnout mnoha konfigurací kráčivých podvozků.

2.1.1 Podle počtu končetin

Počet končetin kráčivého podvozku ovlivňuje stabilitu a pracovní prostor robotu. Rozlišují se dva základní typy biologicky inspirovaných kráčivých podvozků [24]. Nejvýraznější rozdíl těchto dvou typů je způsob pohybu končetin ve vztahu k tělu.

První typ je podobný lidem, ptákům nebo šelmám. Pohyb končetiny probíhá kolem horizontální osy. Druhý typ je podobný hmyzu, kdy pohyb končetiny probíhá kolem vertikální osy. Tento způsob poskytuje větší míru stability.

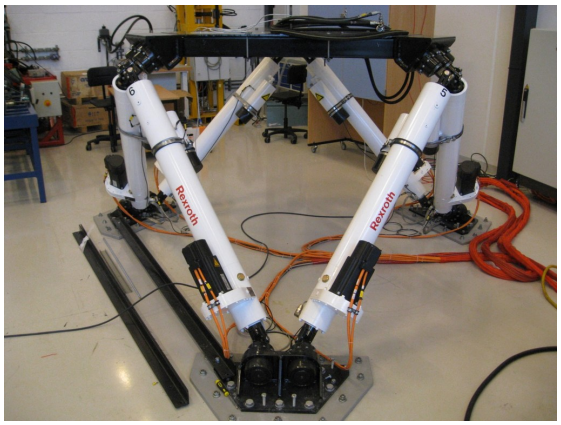
Počet končetin kráčivých podvozků se pohybuje v rozmezí 1 až 9. Vyšší počet končetin je samozřejmě možný, nicméně zbytečný. Speciálním typem potom jsou podvozky bez nohou, které se pohybují plazivým pohybem.

2.1.2 Podle použitého pohonu

Pohon podvozku může být elektrický nebo hydraulický. Elektrický pohon je konstrukčně jednodušší a sestává se z elektromotorů, servomotorů, případně krokových motorů. Elektrický pohon je rychlejší a umožňuje rychlejší reakci a dynamickou chůzi.

Naproti tomu hydraulický pohon je výkonnější a pomalejší. Jeho využití je spíše pro velké roboty, u kterých se předpokládá nutnost manipulace s těžkými předměty. Obecně použití hydraulických pohonů u podvozků je spíše experimentální, lze ho najít například u některých robotů společnosti Boston Dynamics. Jiným příkladem užití hydraulických podvozků je Stewartova platforma. Jedná se o zařízení, které se obvykle skládá z šesti hydraulických končetin, které jsou v párech připevněny k plošině. Nastavením jednotlivých

končetin můžeme dosáhnout libovolného náklonu roviny. Stewartovu platformu zobrazuje Obrázek 2.1. Zařízení se používá například v leteckých simulátorech (viz Obrázek 2.2).



Obrázek 2.1: Stewartova platforma využívající hydraulické motory.[1]

Obrázek 2.2: Praktické využití Stewartovy platformy v leteckém simulátoru.[2]

2.1.3 Podle počtu kloubů

Pro konstrukci kráčivého podvozku je třeba alespoň dva stupně volnosti – jeden pro pohyb končetiny nahoru a dolů, druhý pro pohyb končetiny dopředu a dozadu. Nicméně pro dobře fungující podvozek je třeba alespoň tři stupně volnosti, protože končetiny se pohybují po kružnici a při pohybu těla dopředu by docházelo k prokluzování mezi končetinou a terénem, což lze kompenzovat třetím kloubem. Více než tři stupně volnosti jsou zbytečné pro chůzi jako takovou, mohou být ovšem využity za jiným účelem, například pro manipulaci s předměty.

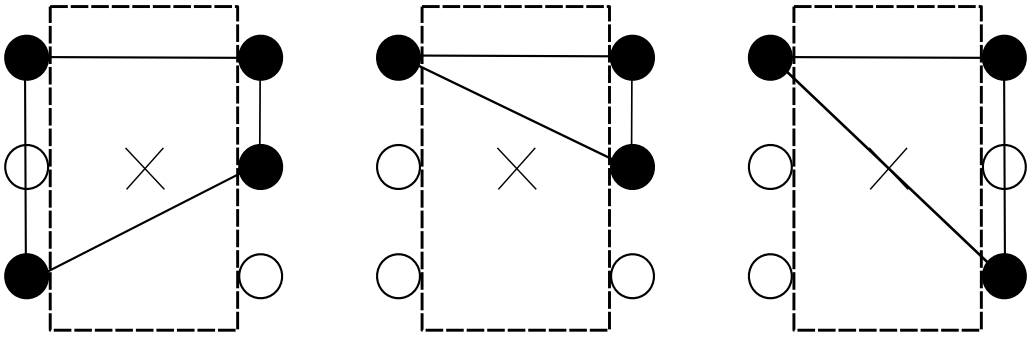
2.1.4 Podle pohybové techniky

Pohyb kráčivých podvozků lze rozdělit na staticky stabilní a dynamicky stabilní [24]. Statická stabilita vyjadřuje schopnost podvozku zůstat ve stabilní poloze v každém okamžiku pohybu. Dynamicky stabilní podvozek se v určité fázi pohybu nachází v nerovnováze – balancuje nebo padá. Použitá pohybová technika je úzce spjata s počtem končetin. Podvozky se třemi (v jistých případech i čtyřmi) a méně končetinami se nedokáží pohybovat, aniž by se neocitly ve stavu, kdy musejí balancovat. Jedná se tedy o dynamický pohyb, kdy se rovnováha podvozku ovlivňuje pohybem končetin. Některé podvozky se dokonce pohybují díky neustálému padání a následnému nastavování volných končetin, které pádu zamezí. Příkladem je chůze dvounohých robotů, kdy při přesunu končetiny na další pozici robot balancuje na jedné končetině.

Naproti tomu statický pohyb můžeme najít u robotů se čtyřmi a více končetinami, kdy je možné provádět krok (posun končetin(y) na novou pozici) a podvozek jako celek je neustále v rovnováze. Příkladem může být šestinohý robot, který vždy stabilně stojí na třech končetinách a zbylé tři končetiny přesouvá na nové pozice.

Jednotlivé situace, do kterých se může podvozek dostat popisuje Obrázek 2.3. Staticky stabilní pozice, nestabilní pozice a kriticky stabilní pozice.

[1] Staženo z http://www.norcowe.no/bilder//Stewart%20platform.jpg, dne 7. 1. 2015.
[2] Staženo z http://theflyingengineer.files.wordpress.com/2011/11/sims04.jpg, dne 7. 1. 2015.



(a) Staticky stabilní (b) Staticky nestabilní (c) Kriticky stabilní

Obrázek 2.3: Obrázek uvádí možné situace stability, do niž se podvozek může dostat. Obrázek vychází z [24].

2.2 Inspirace kráčivých podvozků

Nelze si nevšimnout, že konstrukce kráčivých podvozků nápadně připomíná některé zástupce z říše zvířat, především pak koně a šelmy v případě čtyřnohých robotů, členovce v případě šestinohých (brouci) a osminohých (pavouci) robotů. Skutečně to není náhoda, lidé se při konstrukci kráčivých podvozků inspirují v přírodě, kdy evoluce již řadu problémů vyřešila za nás. Proporce jednotlivých končetin a jejich přesné umístění na těle jsou výsledkem dlouhodobého procesu.

Bohužel v technice lidé stále ještě zaostávají, a proto není možné konstrukci těla živočichů přesně napodobovat. Například pohonné jednotky nedosahují ani zdaleka takového výkonu jako svaly živočichů s ohledem na jejich hmotnost. I přesto je ale dobré se z pozorování a studia živočichů poučit a získané informace využít při konstrukci mechanických kráčivých podvozků.

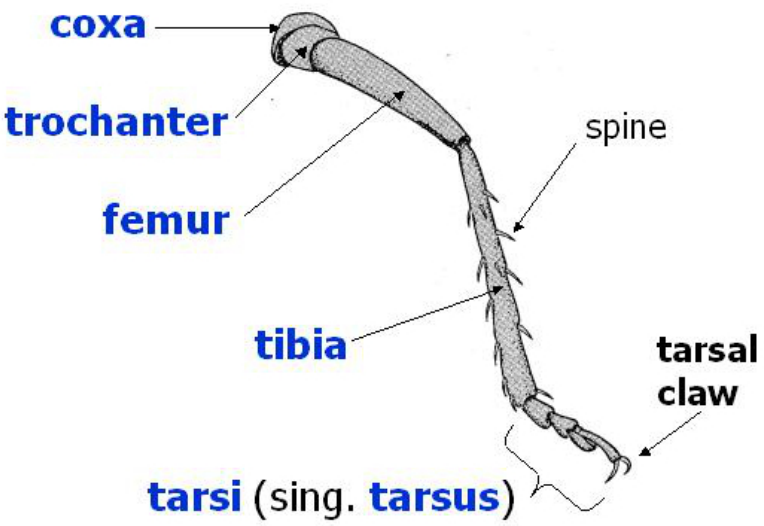
Při srovnávání mechanických robotů s živočichy je obvykle kladen důraz na dvě vlastnosti: rychlost a obratnost [17]. Gepard se dokáže pohybovat na čtyřech končetinách rychlostí mezi 112 a 120 km/h. [36]. Americký šváb se dokáže pohybovat rychlostí 5,4 km/h na šesti končetinách. [20]. Takové srovnání ovšem není příliš relevantní. Proto se pro měření a srovnávání rychlosti různě velkých živočichů používá jednotka délka těla za sekundu. Ta mnohem lépe vystihuje rychlost pohybu živočicha vzhledem k jeho velikosti. Rázem již gepard není tak rychlý, pouze 20 až 30 délek těla za sekundu. Naproti tomu šváb dosahuje rychlosti 50 až 70 délek těla za sekundu. Rekord pak drží svižník [37], který se dokáže pohybovat rychlostí 9 km/h, což odpovídá 250 délek těla za sekundu.

Srovnávání obratnosti je dosti problematické, protože nic jako jednotka obratnosti neexistuje. Přesto je zřejmé, že mnozí živočichové jsou velice obratní. Obyčejný brouk dlouhý 1 cm je schopen kráčet po 1 mm silné dřevěné tyčince, na jejím konci se obrátit a jít zpět, aniž by spadl. Některá klíšťata jsou dokonce schopna překonat "propast"stejně širokou jako je délka jejich těla.

Roboty takové rychlosti nebo obratnosti zatím nedosahují. Jeden z nejrychlejších robotů je Cheetah [10] (viz Obrázek 2.12), který v laboratoři překonal rychlost 46 km/h. Ve volném terénu byl odzkoušen v roce 2013 a dosáhl rychlosti přes 25 km/h.

Konstrukce končetin často vychází ze skutečných živočichů. I proto se v odborné zahra-

ční literatuře užívají jejich původní názvy (viz Obrázek 2.4).



Obrázek 2.4: Schéma končetiny hmyzu. Končetina je připojena k tělu kyčelním kloubem (coxa). Následuje stehenní kost (femur) napojená na kost holenní (tibia). Konstrukce robotické končetiny je obdobná.[3]

2.3 Konstrukce kráčivých podvozků

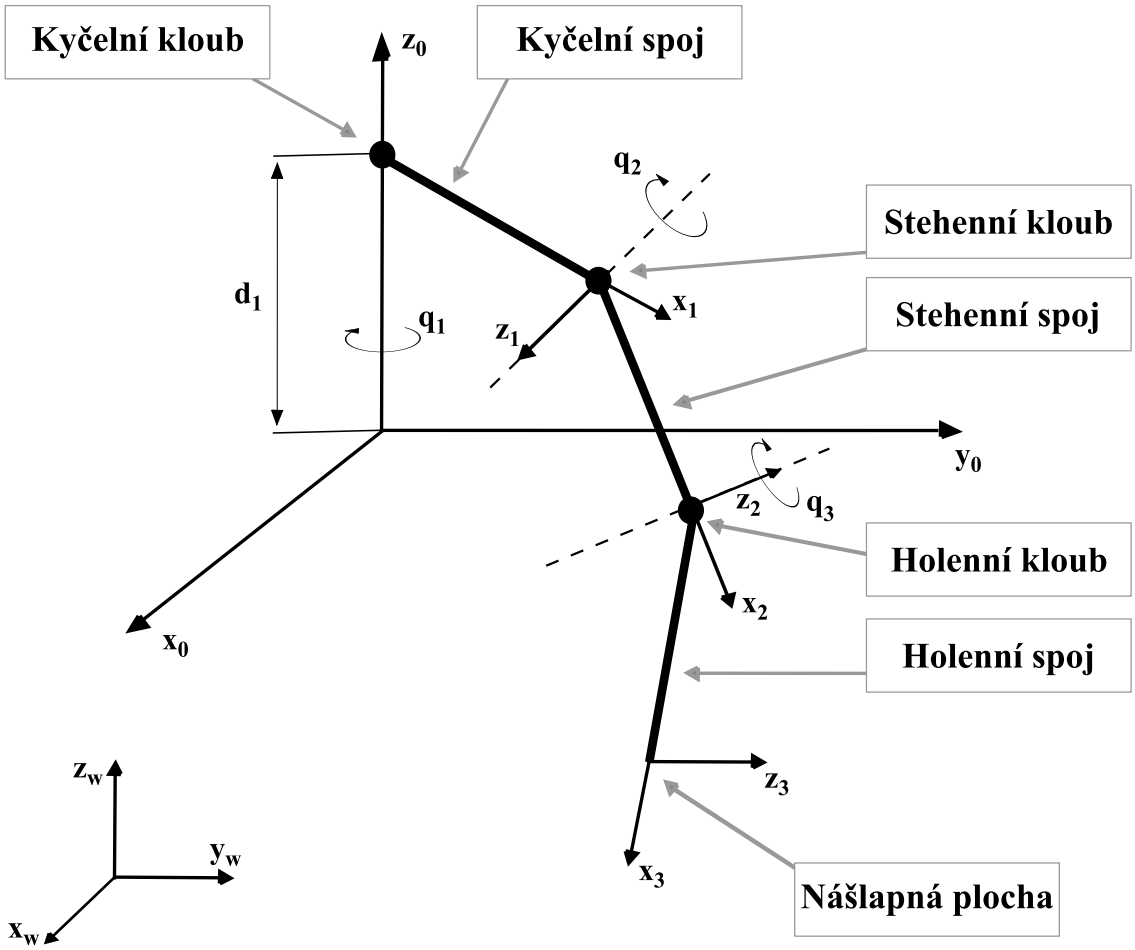
Nejběžnější typy kráčivých podvozků lze rozdělit podle tvaru podvozku do dvou základních skupin: obdélníkové, které mají shodný počet končetin na každé straně podvozku a hexagonální (případně kruhové), které mají končetiny rovnoměrně rozloženy po obvodu celého podvozku.

2.4 Konstrukce končetin

Při návrhu kráčivých podvozků hraje významnou roli návrh končetiny jako takové. Její tvar, počet stupňů volnosti nebo rozměry mají vliv na následný pohyb podvozku. Je důležité vybrat končetinu, která umožní maximální rozsah pohybu, a která nebude klást zbytečná omezení při chůzi. Pohyb končetiny je limitován jejími fyzickými a fyzikálními omezeními, především pohyblivostí jednotlivých kloubů.

Končetinu můžeme rozdělit na tři části [16, 31]. Za prvé je to část připojená k tělu robotu nazývaná coxa (kyčelní spoj). Coxa je k tělu připojena kyčelním kloubem a je využívána k rotaci končetiny směrem dopředu a dozadu. Dále je to femur (stehenní spoj), který je přímo připojen pomocí stehenního kloubu ke kyčli. Stehenní část končetiny slouží k zvedání končetiny nahoru a dolů. V neposlední řadě je to tibia (holenní spoj). Ten je připojen k stehenní kosti pomocí holenního kloubu a slouží k vyrovnávání chůze. Ke konci holenní kosti je připojena nášlapná plocha končetiny. Konstrukci končetiny zachycuje Obrázek 2.5.

[3] Staženo z http://www.insectsexplained.com/img/0304.JPG, dne 8. 1. 2015.



Obrázek 2.5: Schéma robotické končetiny. Podobně jako končetina živočicha se skládá ze tří hlavních části: kyčle, stehna a holeně. Jednotlivé části jsou navzájem spojeny klouby. Obrázek vychází z [24].

2.5 Kinematika kráčivých podvozků

Pro popis kinematických vlastností bude použita končetina se třemi stupni volnosti, jejíž schéma je uvedeno na Obrázku 2.5. Přímý geometrický model pro každou končetinu je vztah mezi fixním rámcem $O_W(X_W, Y_W, Z_W)$ a mezi pohyblivým rámcem nášlapné plochy končetiny $O_i(x_i, y_i, z_i)$, kde $i = 1, 2, 3$ [24]. Jednotlivé rámce jsou označeny na Obrázku 2.5. Propojení jednotlivých kloubů odpovídá Denavit-Hartenbergově algoritmu pro přímé geometrické modelování. Model končetiny je tvořen klouby a spoji, které se nazývají coxa (kyčelní spoj), femur (stehenní spoj) a tibia (holenní spoj).

Rámec končetiny začíná propojením 0, což je bod, kde je končetina připojena k tělu robotu. Spojení 1 potom odpovídá kyčelnímu spoji, spojení 2 odpovídá stehennímu spoji a spojení 3 odpovídá holennímu spoji. Končetiny jsou rozmístěny symetricky kolem osy ve směru pohybu (v tomto případě kolem osy y). Obecná forma transformační matice ze spoje i do spoje $i - 1$ s použitím Denavit-Hartengbergových parametrů vypadá následovně:

$$T_i^{i-1} = \begin{bmatrix} cos\theta_i & -sin\theta_i cos\alpha_i & sin\theta_i sin\alpha_i & a_i cos\theta_i \\ sin\theta_i & cos\theta_i cos\alpha_i & -cos\theta_i sin\alpha_i & a_i sin\theta_i \\ 0 & sin\alpha_i & cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.1}$$

Transformační matice se skládá ze série transformací:

1. Posun d_i podle osy z_{i-1},
2. Otočit θ_i kolem osy z_{i-1},
3. Posun a_i podle osy x_{i-1},
4. Otočit α_i kolem osy x_{i-1}.

Celková transformace je získána vzájemným vynásobením tří transformačních matic:

$$T^{kycel}_{naslapna\_plocha} = T^{kycel}_{stehno} T^{stehno}_{holen} T^{holen}_{naslapna\_plocha} \tag{2.2}$$

kde

$$T^{kycel}_{stehno} = \begin{pmatrix} cos\theta_1 & 0 & sin\theta_1 & L_1 cos\theta_1 \\ sin\theta_1 & 0 & -cos\theta_1 & L_1 sin\theta_1 \\ 0 & 1 & 0 & d_1 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{2.3}$$

$$T^{stehno}_{holen} = \begin{pmatrix} cos\theta_2 & sin\theta_2 & 0 & L_2 cos\theta_2 \\ sin\theta_2 & -cos\theta_2 & 0 & L_2 sin\theta_2 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{2.4}$$

$$T^{holen}_{naslapna\_plocha} = \begin{pmatrix} cos\theta_3 & -sin\theta_2 & 0 & L_3 cos\theta_2 \\ sin\theta_3 & cos\theta_3 & 0 & L_3 sin\theta_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{2.5}$$

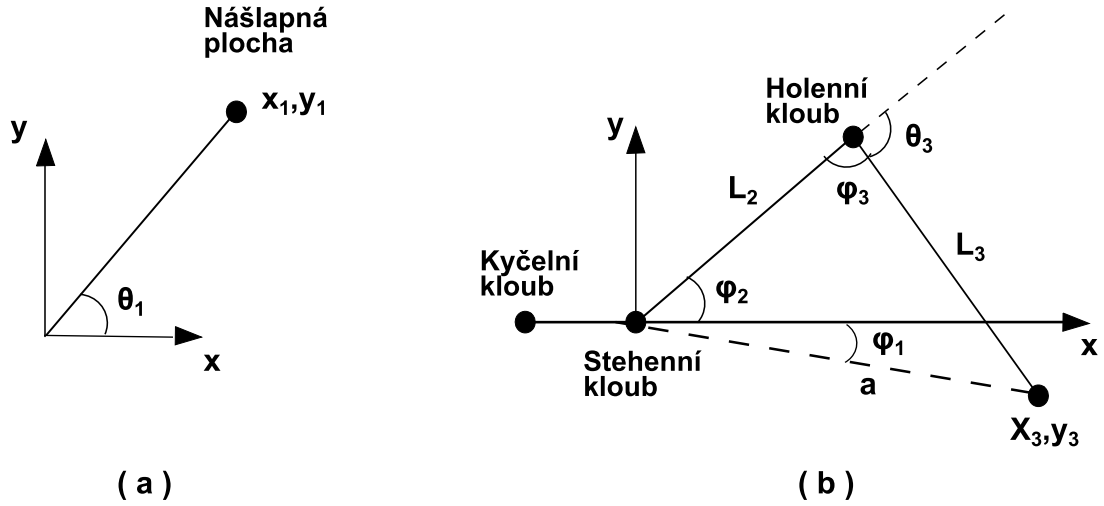
Užitím rovnice 2.2 a uvážením 2.3, 2.4 a 2.5 je možné vyjádřit souřadnice nášlapné plochy končetiny následovně:

$$\begin{aligned} x &= cos\theta_1(L_1 + L_2 cos\theta_2 + L_3 cos(\theta_2 - \theta_3)) \\ y &= sin\theta_1(L_1 + L_2 cos\theta_2 + L_3 cos(\theta_2 - \theta_3)) \\ z &= d_1 L_2 sin\theta_2 + L_3 sin(\theta_2 - \theta_3) \end{aligned} \tag{2.6}$$

kde:
d_1 je vzdálenost od země ke kyčelnímu kloubu,
L_i jsou délky jednotlivých spojů končetiny.

2.5.1 Nepřímá kinematika

V předchozí části je uveden popis geometrického modelu, který spojuje pozice jednotlivých kloubů a orientaci koncového rámce. Aby bylo možné robot efektivně řídit, je třeba nastavovat úhly na jednotlivých motorech, nikoli z úhlů počítat koncovou polohu končetiny. K tomu slouží inverzní kinematika, které řeší problém nalezení cílových hodnot pro správné natočení motorů. Cílem je nalézt tři proměnné $\theta_1, \theta_2, \theta_3$, do kterých se mají motory nastavit, aby bylo dosaženo požadované polohy končetiny.



Obrázek 2.6: Schéma končetiny. Obrázek (a) zobrazuje vztah končetiny k tělu robotu. Obrázek (b) zobrazuje závislosti stehna a holeně končetiny. Obrázek vychází z [24].

Použitím rovnice 2.6 a uvážením následujících omezení: všechny klouby rotují pouze kolem jedné osy, stehenní a holenní kloub rotují kolem stejné osy, fyzická omezení jednotlivých končetin a kloubů, je možné určit úhel jednotlivých kloubů.

Úhel pro kyčelní kloub můžeme získat, jak ukazuje Obrázek 2.6 následovně:

$$\theta_1 = atan2(y_1, x_1) \tag{2.7}$$

Pro zjištění zbývajících dvou úhlů lze použít geometrický přístup. Pro zjednodušení se aplikuje následující transformace na požadované souřadnice nášlapného bodu končetiny:

$$T^{stehno}_{kycel} = \begin{pmatrix} (T^{kycel}_{stehno})^T & -(T^{kycel}_{stehno})^T \cdot d^{kycel}_{stehno} \\ 0 & 1 \end{pmatrix} \tag{2.8}$$

Úhel φ_2, který reprezentuje natočení stehenního kloubu, je možné vyčíst přímo z trojúhelníku:

$$\theta_2 = \varphi_2 \tag{2.9}$$

Úhel φ_1 je úhel mezi osou x přímkou a a je možné ho spočítat následovně:

$$\varphi_1 = atan2(y_3, x_3) \tag{2.10}$$

kde x_3 a y_3 jsou požadované souřadnice nášlapné plochy končetiny. Úhel stehenního kloubu lze získat následovně:

$$\theta_2 = acos(\frac{L_2^2 + a^2 - L_3^2}{2L_2a}) + atan2(y_3, x_3) \tag{2.11}$$

Užitím Kosinovy věty je možné vyjádřit úhel φ_3 následovně:

$$\varphi_3 = acos(\frac{L_2^2 + L_3^2 - a^2}{2L_2L_3}) \tag{2.12}$$

Z Obrázku 2.6 je vidět, že úhel θ_3 lze nalézt následovně:

$$\theta_3 = \pi - \varphi_3 \tag{2.13}$$

2.6 Reprezentace pohybových algoritmů

Při popisování pohybových algoritmů kráčivých robotů je třeba pokrýt několik informací. Za prvé je to jednoznačná identifikace končetiny, které se pohyb bude týkat. Za druhé je to potřeba identifikovat kloub dané končetiny. Dále je třeba určit polohu, do které se daný kloub nastavuje. Za třetí je to čas, ve kterém se má nastavení končetiny provést. Čas je brán relativně k předchozímu pohybu, protože není známa rychlost pohybu jednotlivých končetin. Často se graf zjednodušuje a uvažuje se pouze pohyb vpřed a vzad. Taková reprezentace je dostačující pro reprezentaci jednotlivých algoritmů. Pro následující reprezentace bude použit šestinohý robot.

2.6.1 Reprezentace sekvencí textových označení

Pohybový algoritmus je možné reprezentovat sekvencí textových označení daných končetin spolu s časem a pozicí. Zápis může vypadat následovně:

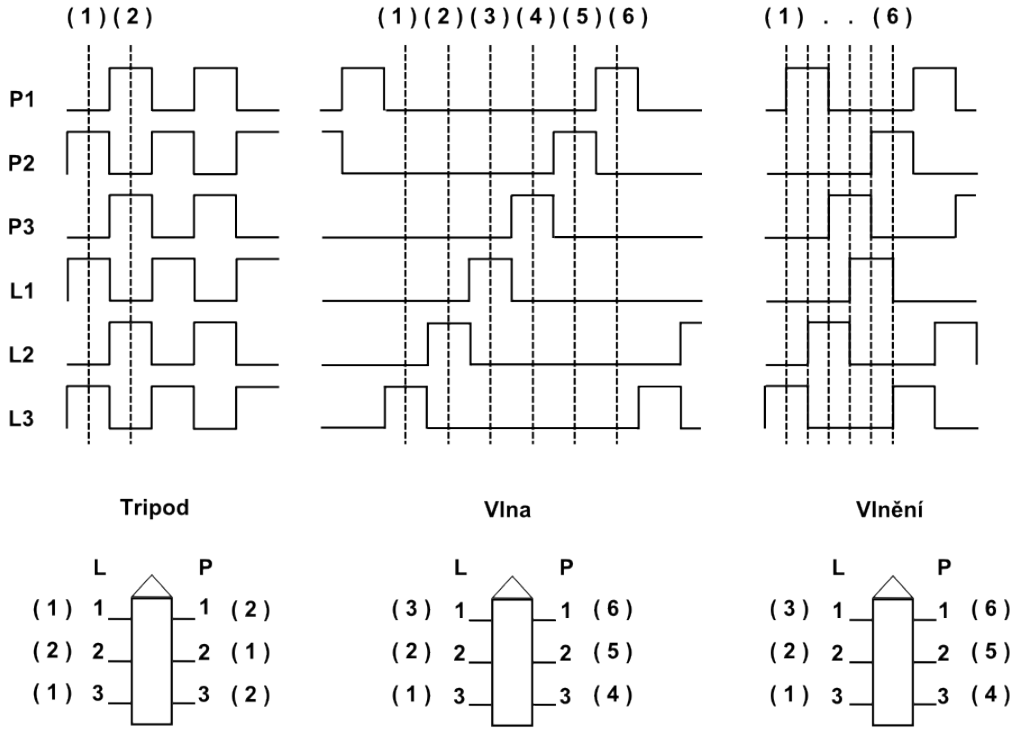
- 1, 3, 5 nahoru
- 1, 3, 5 dopředu; 2, 4, 6 dozadu

- 1, 3, 5 dolů
- 2, 4, 6 nahoru
- 1, 3, 5 dozadu; 2, 4, 6 dopředu
- 2, 4, 6 dolů

Tato reprezentace je velmi neurčitá, a proto se nepoužívá.

2.6.2 Reprezentace grafem

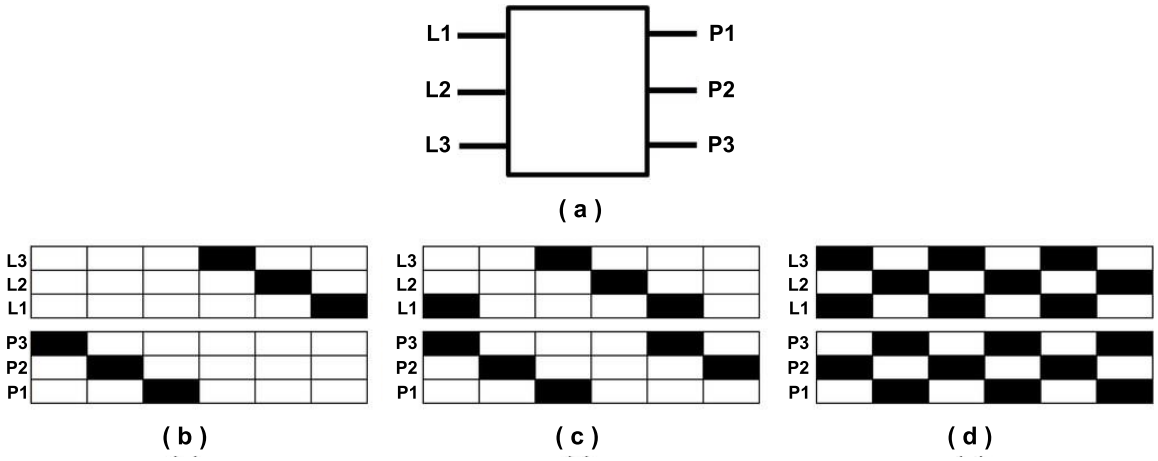
Další způsob je reprezentace pomocí grafu [33]. Každý kloub každé končetiny má vlastní část grafu a přechod mezi horní a dolní mezí reprezentuje pohyb končetiny dopředu resp. dozadu. Osa x reprezentuje čas, osa y reprezentuje jednotlivé končetiny. Příklad uvádí Obrázek 2.7.



Obrázek 2.7: Reprezentace pohybových algoritmů grafem. Obrázek zachycuje časování pohybu končetin u tří nejznámějších kráčivých algoritmů: tripod, vlna a vlnění.[4]

Další možností reprezentace je tabulka. Černá pole udávají posun dané končetiny směrem vpřed. Bílá pak znamenají zachování předchozího stavu (žádnou akci). Příklad je uveden na Obrázku 2.8.

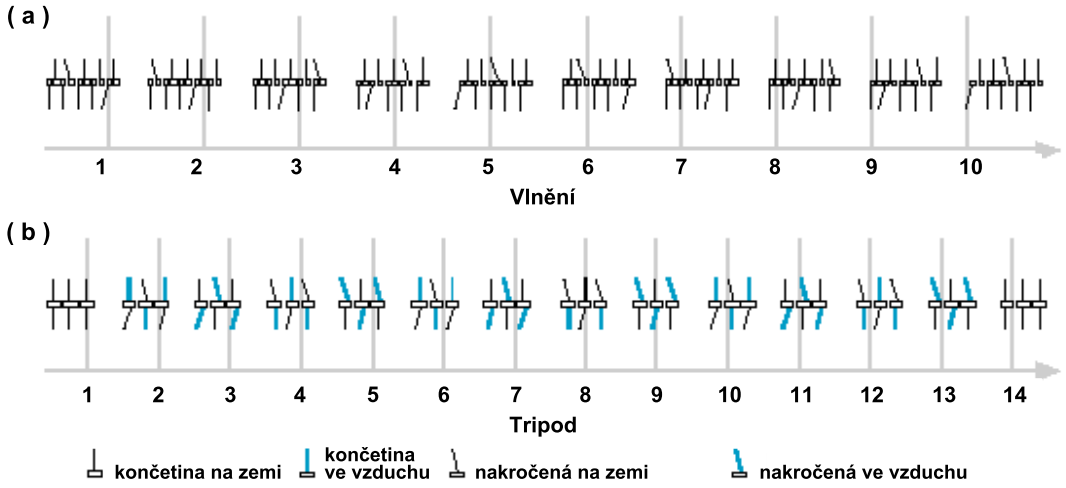
[4] Obrázek inspirován http://www.oricomtech.com/projects/cynthia2.gif dne 8. 12. 2014.



Obrázek 2.8: Reprezentace grafem. Jiná varianta reprezentace pohybového algoritmu grafem [33].

2.6.3 Reprezentace schématem

Jednotlivé pohybové algoritmy je také možné reprezentovat pomocí jednoduchých schémat robotu. V takovém případě zachycujeme stav končetin v jednotlivých časových okamžicích. Příklad uvádí Obrázek 2.9.



Obrázek 2.9: Reprezentace schématem. Obrázek (a) uvádí pohybový algoritmus vlnění, Obrázek (b) uvádí algoritmus tripoda.[5]

2.7 Pohybové algoritmy hexapodu

Jak už bylo řečeno, hexapod a kráčivé podvozky obecně používají pro svůj pohyb chůzi. Za chůzi je možné považovat cyklické kladení končetin požadovaným směrem a přenášení váhy podvozku. Při chůzi se musí alespoň jedna končetina neustále dotýkat země.

[5] Staženo z http://www.mindcreators.com/Images/LM_InsectGaits.gif, dne 8. 12. 2014.

U člověka je tento způsob pohybu poměrně snadno popsatelný, protože člověk disponuje pouze dvěma končetinami. Pokud je tedy jedna noha nakročena dopředu, musí následovat druhá končetina, protože první končetina krok již udělala.

Podobně je tomu u robotů. Významný rozdíl je ale v počtu končetin. V případě robotu hexapod je k dispozici hned šest končetin. Existuje tedy velké množství kombinací, jak je možné klást končetiny (viz Kapitola 2.7.4).

2.7.1 Tripod

Při použití pohybového algoritmu tripod [38] jsou končetiny rozděleny do dvou skupin: pravá přední, pravá zadní a levá střední jsou v jedné skupině, ostatní končetiny jsou v druhé skupině. Následně se jedna skupina končetin zvedne, nakročí dopředu a položí. Poté se zvedne druhá skupina končetin a nakračuje dopředu. Současně se první skupina končetin posouvá dozadu. Nakonec se druhá skupina položí. Je vidět, že obě skupiny vykonávají stejný pohyb, jen jsou navzájem posunuté o půl periody. Algoritmus je zobrazen na Obrázku 2.7, Obrázku 2.8 (d) a Obrázku 2.9 (b).

Tento algoritmus je jeden z nejrychlejších. Na druhou stranu je také nejméně stabilní, protože robot vždy spočívá pouze na třech končetinách, někdy dokonce vychýlených z osy těla, což narušuje těžiště, a může dojít k pádu robotu. Navíc motory, které zvedají středové končetiny nahoru a dolů, jsou velice namáhány, protože nesou v jeden okamžik polovinu váhy celého robotu.

2.7.2 Wave (Vlna)

Algoritmus vlny [38] spočívá v postupném nastavování končetin směrem vpřed – vždy se hýbe pouze jedna končetina. Po nastavení všech končetin dojde k posunu směrem vpřed všemi končetinami současně (končetiny se pohybují směrem vzad).

Tento algoritmus je velice stabilní, protože robot vždy spočívá na pěti a více končetinách. Z toho důvodu je také velice pomalý a jeho využití je vhodné například v terénních nerovnostech. Algoritmus je uveden na Obrázku 2.7.

2.7.3 Ripple (Vlnění)

Algoritmus vlnění [33] vychází z pohybu skutečných živočichů, například brouků. Každá končetina neustále vykonává ten samý pohyb: zvedne se nahoru, nakročí směrem dopředu, položí na zem a udělá krok směrem dopředu (končetina se pohybuje dozadu).

V tomto algoritmu je velice důležitá dynamika jednotlivých částí kroku. Pohyb končetiny směrem vzad (tedy krok dopředu) určuje rychlost pohybu. Ostatní pohyby (nahoru, dolů a nakročení vpřed) by měly probíhat maximální možnou rychlostí. Díky tomu je tento algoritmus velice stabilní, protože doba, kdy se končetina nedotýká země je minimalizována maximální rychlostí přesunu končetiny. Pohyb robotu je velice plynulý. Algoritmus je uveden na Obrázku 2.7.

2.7.4 Další algoritmy

Jiným způsobem pohybu kráčivého podvozku může být kráčení stranou. To je možné díky třetímu stupni volnosti. Použití algoritmu je vhodné, pokud změna směru nevyžaduje zatočení.

Další možnost pohybu kráčivého podvozku je rotace. Při ní se robot otáčí kolem své osy díky synchronizovaným úkrokům vpřed a vzad.

Počet kráčivých algoritmů N kráčivých robotů závisí na počtu jejich končetin. Ten je možné vyjádřit následovně:

$$N = (2K - 1)! \tag{2.14}$$

Je vidět, že počet různých algoritmů rychle roste s počtem končetin (pro robot se šesti končetinami existuje 11! = 39 916 800 možných sekvencí pohybu), nicméně mnohé z nich není možné v praxi použít, protože nevedou k pohybu nebo způsobují nestabilitu a pády robotu [32].

2.7.5 Překonávání překážek

Kráčivé podvozky jsou vhodné pro pohyb v prostředí s velkými terénními nerovnostmi. Plynulého pohybu přes překážky lze dosáhnout všemi uvedenými algoritmy, nicméně je vhodné zvolit některý ze stabilnějších, jako je vlna nebo vlnění.

2.7.6 Výběr vhodného algoritmu

Při výběru vhodného algoritmu je nejdůležitější znát členitost terénu, v kterém se bude robot pohybovat. Dále je třeba zvážit maximální nosnost robotu s ohledem na výkon jeho motorů. Například při použití algoritmu tripoda je v daný okamžik polovina váhy robotu (včetně nákladu) pouze na jedné končetině. Motory musejí unést i takovéto zatížení, jinak není možné algoritmus tripoda použít.

Jednotlivé algoritmy je možné střídat i během provozu robotu na základě aktuální hmotnosti a členitosti okolního terénu. Je ovšem důležité zajistit, aby během výměny algoritmu nedošlo k pádu robotu.

2.8 Řízení hexapodu neuronovou sítí

Jednoduché pohybové algoritmy kráčivých robotů si je schopen programátor představit a implementovat. Ovšem počet kombinací všech možných poloh končetin a situací, v nichž se může robot vyskytnout, je obrovské. Proto se stále častěji objevují výzkumy a projekty, které se zabývají různými způsoby generování pohybových algoritmů.

Jedním z možných řešení je řídit pohyb robotu pomocí neuronové sítě [28, 38]. Cílem je nalézt takovou neuronovou síť, která je schopna generovat sekvenci signálů, které by dokázaly ovládat jednotlivé končetiny způsobem, jenž by utvářel neměnný pohyb vpřed podél osy těla robotu. Řízení pomocí neuronové sítě má oproti řízení konečným automatem nebo řádky kódu jazyka C výhodu při provádění změn. Malá změna ve struktuře sítě vede k malé změně chování robotu.

Pří výstavbě neuronové sítě se objevuje několik zásadních problémů. Není znám žádný algoritmus, který dokáže určit velikost a strukturu neuronové sítě, která dokáže reprezentovat požadované chování – pohybový algoritmus. Malá síť s malým počtem propojení nemusí být schopna reprezentovat požadovaný algoritmus, velká, plně propojená síť zase vyžaduje velké množství trénovacích vektorů a její učení trvá dlouho. Dalším problém spojeným s neuronovými sítěmi je tvorba trénovací množiny. Čím větší je neuronová síť, tím více trénovacích vektorů je třeba.

Výhodou neuronových sítí je, že dokáží během procesu trénování nalézt vhodný model, který nemusí odpovídat skutečnému systému. Dále zde existuje také možnost distribuce výpočtů na více systémů.

Randall D. Beer vyvinul rekurentní neuronovou síť, založenou na studiu amerického švába [9]. Tato síť byla ovšem ručně nastavena, aby dosahovala požadovaných výsledků. Přestože se jedná o velký úspěch, chybí zde automatické učení a proces obsahuje příliš mnoho lidské intervence.

Při tvorbě neuronové sítě je vhodné nejprve naučit "chodit" pouze jednu končetinu, a následně takto natrénovanou síť rozkopírovat, vzájemně propojit a trénovat jen spoje mezi končetinami.

2.8.1 Generování trénovací množiny

Generování trénovací množiny je relativně obtížné, protože člověk nemá osobní zkušenost s chůzí po šesti končetinách. Navíc vytváření trénovací množiny je časově náročné. Možné je například vytvořit model robotu, jehož klouby jsou osazeny potenciometry, a poté na takovém modelu ručně simulovat pohyb a zaznamenávat jej.

Jiným způsobem generování trénovacích dat je simulace. Výhodou simulace je, že není třeba ji dělat krok po kroku, probíhá do jisté míry automaticky. Je ovšem třeba vytvořit model skutečného světa, což je finančně i časově náročné. Některé faktory se modelují velice obtížně, jiné není možné zachytit vůbec [35].

2.8.2 Trénování neuronové sítě

Pro trénování neuronové sítě je možné použít standardní metody, jako je například back-propagation. Další způsob určení vah jednotlivých neuronů je možnost použití genetických algoritmů [28]. Náhodně je vygenerována populace několika desítek chromozomů, kde každý chromozom reprezentuje strukturu neuronové sítě. Následně je daná síť spuštěna a vygeneruje několik stovek pulzů pro servomotory. Jednotlivé sekvence jsou ohodnoceny fitness funkcí, která má tři základní parametry: pohyb dopředu, počet zvednutí končetiny nahoru a odpor. Pohyb dopředu odpovídá pohybu, kdy byla končetina na zemi. Počet zvednutí končetiny je penalizace, protože jen spotřebovává energii a snižuje efekt dopředného pohybu. Odpor je penalizace, která vzniká, pokud je končetina na zemi a je nastavena do nejzadnější pozice. V takovém případě totiž při pohybu dopředu pouze zpomaluje.

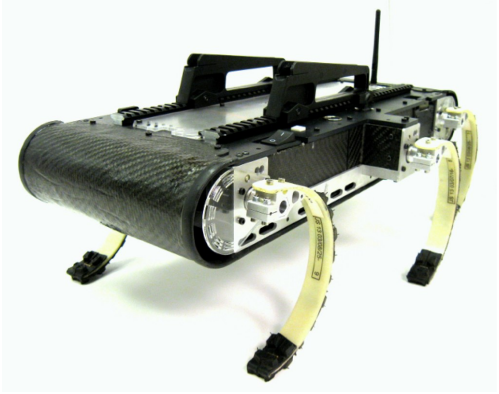
2.9 Příklady existujících kráčivých robotů

Jedním z největších výrobců kráčivých robotů je společnost Boston Dynamic [11]. Její roboty představují nejvyspělejší řešení v oboru.

2.9.1 RHex

RHex [12] je vysoce mobilní šestinohý robot. Každá končetina je řízená nezávisle, což robotu umožňuje překonávat terénní nerovnosti. Tělo robotu je uzavřené a může operovat ve vlhkém či bahnitém prostředí. RHex je vybaven kamerami vpředu i vzadu a je dálkově řiditelný na vzdálenost až 700 metrů. Jeho konstrukce umožňuje pokračovat v činnosti i při převrácení robotu na záda. Konstrukce je k dispozici od více různých tvůrců převážně z akademických kruhů.

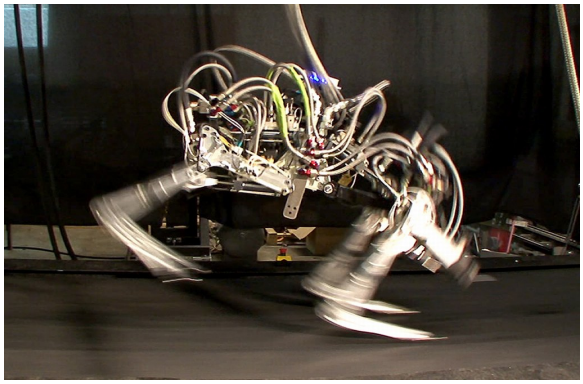
Obrázek 2.10: Boston Dynamics RHex.[6]



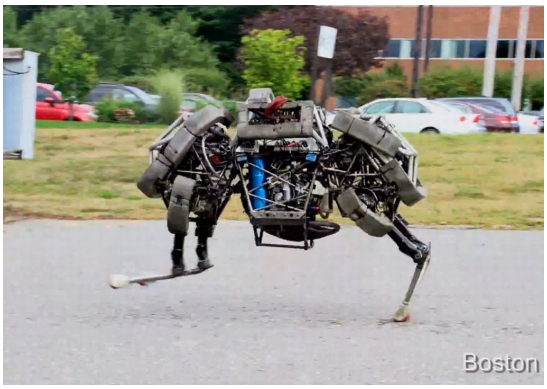
Obrázek 2.11: Stejný typ robotu z univerzity v Pennsylvanii.[7]

2.9.2 Cheetah

Cheetah [10] je doposud nejrychlejší robot, který má končetiny. Robot překoná rychlost 46 km/h na běžícím páse. Robot má kloubová záda a při běhu se ohýbá sem a tam jako skutečná zvířata a je napájen pomocí externího hydraulického čerpadla. Další verze robotu bude WildCat, která je navržena pro provoz ve volném terénu.



Obrázek 2.12: Cheetah.[8]



Obrázek 2.13: WildCat.[9]

2.9.3 BigDog

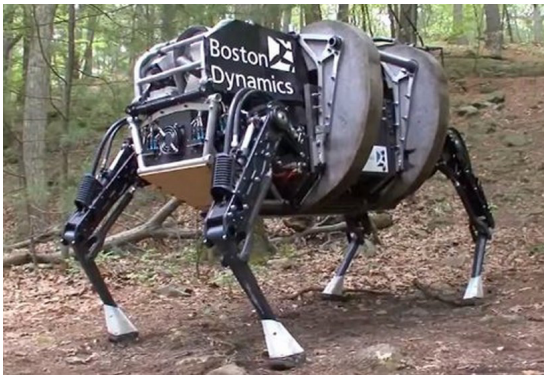
BigDog [13] je čtyřnohý robot sestavený do těžkého terénu poháněný akčním hydraulickým systémem. Velikost odpovídá většímu psu a váží přibližně 110 kg. Robot je vybaven stereo kamerou, lidarem a dalšími senzory, které mapují okolí a pomáhají stabilizaci robotu. BigDog se může pohybovat 6,5 km/h, unese 150 kg nákladu a dokáže vystoupat svah o sklonu 35 ° nebo se pohybovat po suti.

[6] Staženo z http://www.bostondynamics.com/img/RHex-BostonDynamics1.jpg, dne 15. 12. 2014.
[7] Staženo z http://kodlab.seas.upenn.edu/uploads/Kod/X-RHex_tech.jpg, dne 15. 12. 2014.
[8] Staženo z http://www.bostondynamics.com/img/cheetah_reduced.jpg, dne 5. 12. 2014.
[9] Staženo z http://3.bp.blogspot.com/-fsZEBjH2_so/UlHNNiIqY-I/AAAAAAAAFL8/ulWDZ1abcfI/s1600/run.jpg, dne 5. 12. 2014.

2.9.4 LS3

LS3 [14] je čtyřnohý robot vyvinutý z robotu BigDog. Jeho velikost odpovídá přibližné velikosti koně. Robot je vybaven stereo kamerami a senzory, které mu umožňují sledovat člověka. Robot unese náklad o hmotnosti 180 kg, je schopen ujít až 30 km a dosahuje rychlosti 10 km/h. V roce 2012 byl robot představen ve venkovním prostředí. Jeho chůze je dynamická, vždy přesouvá protilehlé končetiny.

Obrázek 2.14: BigDog.[10]



Obrázek 2.15: LS3, nástupce BigDog.[11]

2.9.5 Další roboty

Společnost Boston Dynamic však není jedinou společností zabývající se kráčivými roboty. Například firma Trossen Robotics [34] také vyrábí kráčivé roboty. Jedná se o plastové modely čtyřnohých, šestinohých a osminohých robotů. Každá končetina disponuje třemi stupni volnosti. Roboty jsou poháněny servomotory Dynamixel AX-12A nebo AX-18A. Tyto servomotory jsou speciálně upraveny pro použití v robotice. Jejich pouzdro obsahuje více úchytů, než obyčejné modelářské servomotory, jsou výkonnější a jsou ovládány sériovou linkou. To je výhodné, neboť není třeba speciální kabel ke každému servomotoru a jednotlivé servomotory je možné řetězit.

(a) PhantomX AX Quadruped Mark II.

(b) PhantomX AX Hexapod Mark II.

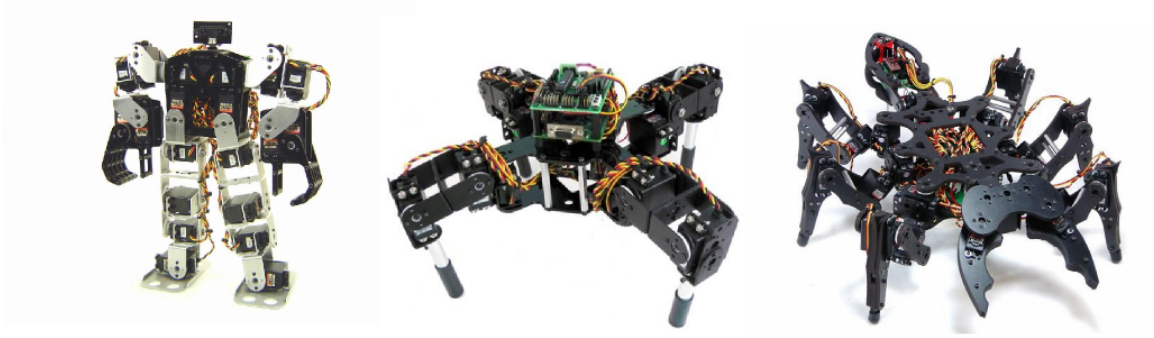
(c) PhantomX Octopod Edge Kit.

Obrázek 2.16: Podvozky Trossen Robotics. [12], [13], [14]

[10] Staženo z http://www.bostondynamics.com/img/BigDog_Snow.png, dne 5. 12. 2014.

[11] Staženo z http://www.blogcdn.com/www.engadget.com/media/2012/09/boston-dynamics-alphadog-ls3-darpa-demo.jpg, dne 5. 12. 2014.

Další společností zabývající se modely kráčivých robotů je společnost Lynxmotion [23]. Na rozdíl od předchozího případu nabízí také dvounohý robot, který disponuje 22 stupni volnosti. Dále jsou k dispozici standardní čtyřnohé a šestinohé verze robotů se třemi stupni volnosti na končetinu. Roboty využívají k pohybu modelářské servomotory firmy Hitec.



(a) Biped Pete. (b) SQ3U. (c) A-Pod.

Obrázek 2.17: Podvozky Lynxmotion. [15], [16], [17]

[12] Staženo z http://learn.trossenrobotics.com/images/tutorials/phantomXQuadruped/phantomXQuadrupedMain.jpg, dne 17. 12. 2014.

[13] Staženo z http://www.trossenrobotics.com/resize/images/PImages/KIT-RK-PXC-HEX-AX-mk2-f.jpg?bw=1000&bh=1000, dne 17. 12. 2014.

[14] Staženo z http://forums.trossenrobotics.com/attachment.php?attachmentid=4675&d=1368193485, dne 17. 12. 2014.

[15] Staženo z http://www.lynxmotion.com/images/hi-res/irnman4.jpg, dne 17. 12. 2014.

[16] Staženo z http://www.robotshop.com/media/catalog/product/cache/1/image/515x515/9df78eab33525d08d6e5fb8d27136e95/l/y/lynxmotion-sq3u-symmetric-quadruped-walking-robot_1.jpg, dne 18. 5. 2015.

[17] Staženo z http://www.lynxmotion.com/images/jpg/apod02.jpg, dne 17. 12. 2014.

Kapitola 3

Úpravy robotu

V rámci předchozí práce byl navržen a sestaven prototyp šestinohého robotu. Jelikož některé parametry navrženého robotu nebyly dostatečné, byl robot pro účely této práce upraven a vylepšen. Ke změnám došlo u některých servomotorů, regulátorů napětí a u řídící jednotky. Dále byly přidány enkodéry do servomotorů a nášlapné snímače na konce končetin.

3.1 Konstrukce vlastního robotu

V rámci práce byl vytvořen robot, který slouží pro ověřování jednotlivých hypotéz a k experimentovaní s kráčivými podvozky (viz Obrázek A.1). Jedná se o šestinohý robot s obdélníkovým tvarem těla (hexapod), jehož každá končetina disponuje třemi stupni volnosti. Konstrukce robotu je z hliníkových profilů, které poskytují dostatečnou pevnost a jsou snadno dostupné. Za pohonnou jednotku byly zvoleny servomotory. Řídicí jednotkou je mikrokontrolér Atmega2560 od společnosti Atmel. Tento čip je integrován na vývojovou desku Arduino Mega 2560. Pro řízení servomotorů je použit servo řadič, který umožňuje ovládat až 34 motorů. Robot dále nese několik ultrazvukových sonarů, které slouží k detekci překážek v okolí, LCD displej, který zobrazuje informace o stavu robotu, a Xbee modul, který slouží ke komunikaci s řídícím počítačem po bezdrátové sériové lince. Robot je napájen jedním tříčlánkovým LiPo akumulátorem 11,1 V. Toto napětí je posléze usměrněno regulátory napětí, protože servomotory vyžadují 6 V napájení a elektronika vyžaduje 5 V napájení. Robot je podrobněji popsán v práci Návrh a konstrukce šestinohého mobilního robotu [21].

3.2 Servomotory

Došlo k výměně středních servomotorů, které slouží ke zvedání končetiny nahoru a dolů. Důvodem výměny byl nedostatečný výkon servomotorů, který se projevoval především při použití pohybového algoritmu trioda, kdy jedna samotná končetina neunesla hmotnost robotu a docházelo k nestabilitě při chůzi. Původní servomotor HS-5485HB [18] měl kroutivý moment 6,4 kg/cm a karbonové převody. Nový servomotor HS-5645MG [19] má téměř dvojnásobný kroutivý moment – 12,1 kg/cm a kovové převody.

3.3 Regulátory napětí

Dále došlo k výměně napěťových regulátorů. V původní verzi robotu byly použity lineární regulátory, které vykazují vysokou ztrátovost při velkých rozdílech mezi vstupním a výstupním napětím. Navíc byly regulátory umístěny v nepájivém poli, které zabíralo příliš mnoho prostoru na těle robotu a hrozilo náhodné uvolnění kabelů. Původní lineární regulátor L78S06 dosahoval účinnosti 57 %, nový spínaný regulátor LM2596 má účinnost 92 %. Navíc jeho výstupní napětí je nastavitelné.

3.4 Raspberry Pi

Ke změnám došlo také v řídící jednotce, která byla rozšířena o modul Raspberry Pi [30], model B+, který má výrazně vyšší výkon než mikrokontrolér a disponuje real-time operačním systémem. Pro komunikaci s řídícím počítačem slouží technologie Wi-Fi, která umožňuje vyšší přenosové rychlosti než Xbee.

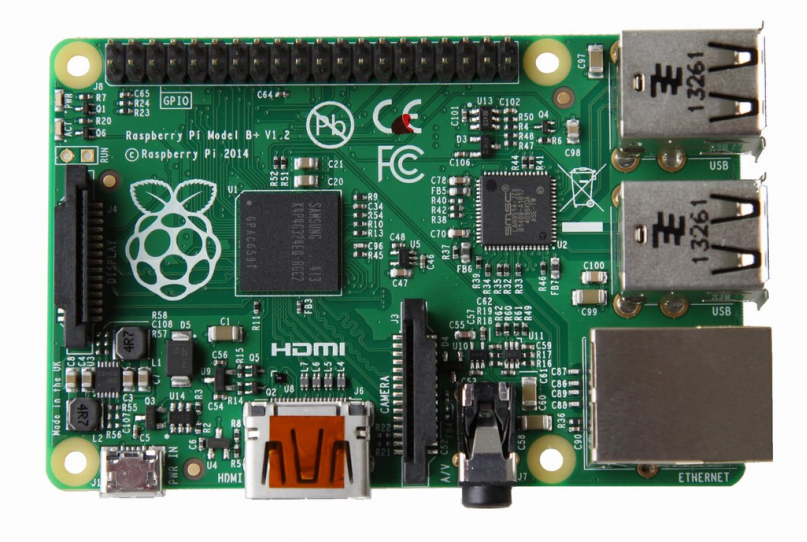
Raspberry Pi je miniaturní počítač o velikosti kreditní karty, do kterého je možné připojit standardní monitor, klávesnici a myš. Deska Raspberry Pi je vybavena ARM procesorem s pracovní frekvencí 700 MHz, pamětí RAM o velikosti 256 resp. 512 MB sdílenou s GPU, grafickým čipem, výstupem na monitor s podporou různých rozlišení, audio výstupem, konektorem síťového rozhraní RJ45 Ethernet 10/100, slotem pro paměťovou kartu microSD a několika USB konektory. Dále je Raspberry Pi vybaveno konektorem pro kameru a GPIO piny, které jsou vyvedeny přímo z procesoru.

Raspberry Pi je k dostání v několika různých modelech: A, A+, B, B+. Jednotlivé modely se liší především svojí vybaveností, velikostí paměti RAM a samozřejmě cenou. Modely A a A+ navíc nemají síťové rozhraní Ethernet, zato mají poloviční spotřebu. Podrobnější srovnání jednotlivých modelů uvádí Tabulka 3.1.

Pro provoz Raspberry Pi je třeba operační systém. Všechny nabízené systémy jsou na bázi linuxového jádra a uživatel má na výběr hned z několika distribucí, v závislosti na požadovaných vlastnostech a zkušenostech uživatele.

	Model A	**Model A+**	**Model B**	**Model B+**
Paměť [MB]:	256	256	512	512
USB 2.0:	1	1	2	4
Úložiště:	SD	MicroSD	SD	MicroSD
Síť:	-	-	10/100 Mbit/s Ethernet	10/100 Mbit/s Ethernet
Nízkoúrovňové periferie:	8 GPIO + UART, I^2C, SPI, I^2S	17 GPIO	8 GPIO + UART, I^2C, SPI, I^2S	17 GPIO
Spotřeba [mA]:	300 (1,5 W)	200 (1 W)	700 (3,5 W)	600 (3 W)

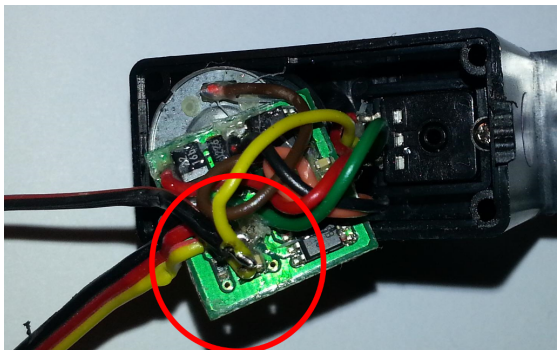
Tabulka 3.1: Srovnání modelů Raspberry Pi.



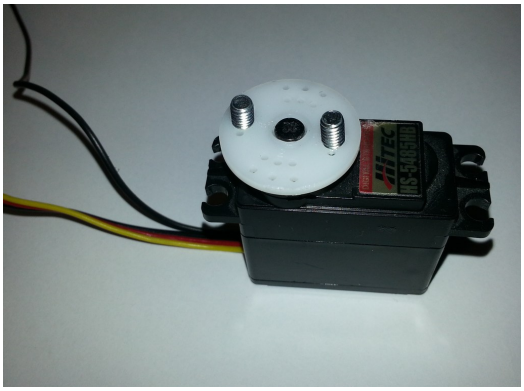
Obrázek 3.1: Raspberry Pi model B+.[1]

3.5 Enkodéry servomotorů

Dále byly přidány enkodéry do všech servomotorů, aby bylo možné sledovat aktuální pozice všech kloubů robotu. Přidání enkodéru do servomotoru není jednoduché, optické enkodéry vyžadují velký zásah do pouzdra servomotoru, navíc jejich výroba je náročná. Cíleného efektu bylo dosaženo připojením kabelu na jezdce potenciometru, který je uvnitř servomotoru. Hřídel potenciometru je pevně spjata s hřídelí servomotoru. Na základě zpětné vazby potenciometru se určuje, zda servomotor dorazil do zvolené pozice (zda vstupní signál odpovídá signálu generovanému na základě natočení potenciometru). Připojením na jezdce potenciometru bylo získáno napětí, které je následně měřeno A/D převodníkem na mikrokontroléru. Ze získané hodnoty je možné určit aktuální polohu hřídele servomotoru.



(a) Připojení k potenciometru.



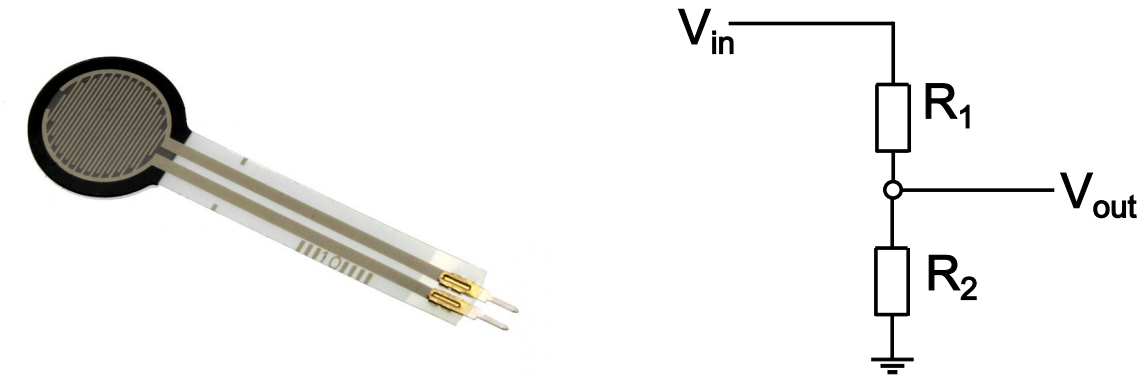
(b) Servomotor s enkodérem.

Obrázek 3.2: Enkodér servomotoru. Připojením kabelu na jezdce potenciometru uvnitř servomotoru je možné získat informaci o aktuální poloze hřídele motoru.

[1] Staženo z https://www.raspberrypi.org/wp-content/uploads/2014/07/rsz_b-.jpg, dne 9. 1. 2015.

3.6 Tlako-citlivé rezistory

Dále byly přidány tlako-citlivé rezistory na konce končetin. Doposud nebylo možné provozovat robot v členitém terénu, končetiny se vždy nastavovaly do úrovně země. Díky tlako-citlivým rezistorům je možné získat informaci o překážce pod končetinou, která není v úrovni země. Data z rezistorů lze také použít k automatické kalibraci jednotlivých končetin robotu.



(a) Tlako-citlivý rezistor.[2]

(b) Dělič napětí.

Obrázek 3.3: Tlakový senzor. Pomocí tlako-citlivého rezistoru a rezistoru 10 KΩ je možné vytvořit dělič napětí. Následně postačí měřit analogové napětí n děliči a určit tak míru přitlaku na tlako-citlivém rezistoru.

3.7 Další senzory

Rozšířen byl také senzorický systém. Bylo přidáno více ultrazvukových sonarů, aby bylo možné monitorovat širší okolí robotu.

[2] Staženo z https://content.solarbotics.com/products/photos/712473272609715702b1be0d48924c32/lrg/50803_-_img_6116.jpg, dne 10. 5. 2015.

Kapitola 4

Hardware a software robotu

Součástí projektu je také kontrolní software pro jednotlivé komponenty. Za prvé je to uživatelské rozhraní, které je vytvořené v jazyce C++ s využitím knihovny Qt. Program slouží k monitorování a řízení robotu, umožňuje vizualizovat aktuální pozice končetin a poskytuje nástroj pro vytváření vlastních pohybových algoritmů.

Za druhé je to program pro Raspberry Pi, který je taktéž vytvořen v jazyce C++ s využitím knihovny Qt. K dispozici jsou dvě verze tohoto programu. První verze programu obsahuje grafické uživatelské rozhraní, které zobrazuje základní informace o stavu robotu. Program také disponuje ladícím oknem, které zobrazuje jednotlivé příchozí a odchozí zprávy. Druhá verze programu je v provedení konzolové aplikace, která redukuje náročnost na výkon systému a je vhodná pro jednovláknové procesory, jako má právě Raspberry Pi.

Za třetí je to program pro mikrokontrolér, který obsluhuje jednotlivé periferie robotu jako například sonary, LCD displej nebo motory. Dále zajišťuje vykonávání příkazů, které jsou uživatelem požadovány. Také odesílá informace o stavu robotu a data ze senzorů.

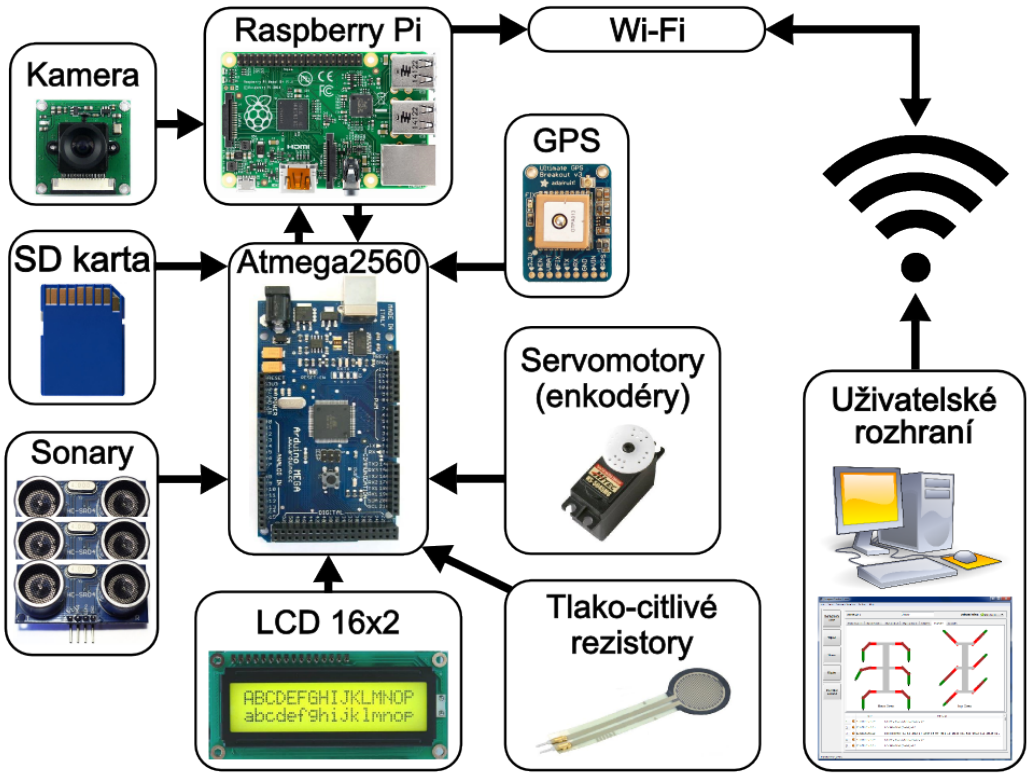
4.1 Elektronické komponenty robotu

Robot je vybaven řadou prvků, které slouží k pohybu či získávání dat z prostředí. V první řadě jsou to pohonné jednotky – servomotory, které slouží k pohybu končetin. Dále jsou to sonary, které umožňují detekovat překážky pomocí ultrazvukových vln. Každá končetina je vybavena snímačem země, který umí rozeznat došlápnutí končetiny. Dále je to LCD displej, který zobrazuje základní informace o robotu, GPS modul, který udává aktuální polohu robota, a SD karta, která slouží k ukládání konfiguračních souborů a logů. Robot je také vybaven senzorem stavu baterie.

Vše je řízeno z mikrokontroléru Atmega 2560, který je integrován na vývojovou desku Arduino. Na piny mikrokontroléru jsou připojeny jednotlivé periferie. Mikrokontrolér je dále připojen pomocí sériové linky k Raspberry Pi. K Raspberry Pi je dále připojena kamera a Wi-Fi modul. Schéma jednotlivých komponent a jejich propojení je uvedeno na Obrázku 4.1.

4.1.1 Servomotory

Servomotory se starají o pohyb jednotlivých končetin. Každá končetina disponuje třemi servomotory. První motor slouží k rotaci končetiny vpřed a vzad, druhý motor umožňuje pohyb končetiny nahoru a dolů, a třetí motor je využíván k vyhlazování chůze a při chůzi stranou.



Obrázek 4.1: Elektronické schéma robotu. Srdcem je mikrokontrolér Atmel Atmega 2560, který je integrován na vývojovou desku Arduino. K mikrokontroléru jsou připojeny periferie jako LCD displej, SD karta, sonary, servomotory a enkodéry. Mikrokontrolér je připojen k Raspberry Pi pomocí sériové linky. K Raspberry Pi je připojena také kamera a Wi-Fi modul. Robot je řízen z uživatelského rozhraní, které běží na řídícím počítači.

Robot je poháněn dvěma typy servomotorů. První typ servomotoru je HS-5485HB [18], který je využit pro rotační a kotníkový kloub končetin. Druhý typ servomotoru je HS-5645MG [19], který je výkonnější a je použit pro rotační kloub končetin. Základní parametry servomotorů jsou uvedeny v Tabulce 4.1.

4.1.2 Sonary

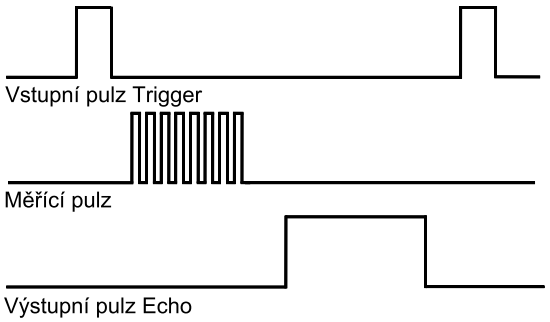
Robot je vybaven ultrazvukovými dálkoměry – sonary. Jedná se o typ HC-SR04 [15]. Tento typ dálkoměru je schopen měřit vzdálenost v rozmezí 2 - 400 cm s rozlišením 0,3 cm. Vyzařovací úhel sonaru je 30 °. Sonar je napájen 5 V a jeho maximální spotřeba činí 15 mA.

Sonar je připojen k digitálním pinům mikrokontroléru. Pin 'Trig' je připojen na digitální výstup a pin 'Echo' je připojen na digitální pin, který podporuje externí přerušení.

	HS-5485HB	HS-5645MG
Typ motoru:	3-pólový motor	3-pólový motor
Typ ložiska:	horní kuličkové	dvojité kuličkové
Rychlost (4,8/6,0V) [s/60°]:	0,20 / 0,17	0,23 / 0,18
Točivý moment (4,8/6,0V) [kg.cm]:	5,2 / 6,4	10,3 / 12,1
Rozměry [mm]:	39,88 x 19,81 x 37,85	40,39 x 19,56 x 37,59
Váha [g]:	45,08	59,82

Tabulka 4.1: Parametry použitých servomotorů. První servomotor HS-5485HB je méně výkonný a je použit pro rotační a kotníkový kloub. Druhý servomotor HS-5645MG je výkonnější a je osazen ve zvedacích kloubech končetiny.

(a) Ultrazvukový dálkoměr HC-SR04. Senzor disponuje čtyřmi piny. Piny 'Vcc' a 'GND' jsou pro připojení napájení 5 V. Pin 'Trig' slouží k vystavení signálu, který zajistí vyslání ultrazvukových vln do okolí. Na pinu 'Echo' se následně objeví pulz, jehož délka odpovídá době, která uplynula od vyslání vln.[1]



(b) Schéma měření vzdálenosti sonarem. V první části je na pin 'Trig' vystaven signál, který způsobí vyslání několika ultrazvukových vln do okolí. Tyto vlny se v prostředí odrazí od objektů a vrátí se zpět do přijímače sonaru. Na základě času uplynulého od vyslání vln a rychlosti zvuku je možné spočítat vzdálenost od nejbližší překážky.

Obrázek 4.2: Sonar HC-SR04 a schéma měření vzdálenosti.

4.1.3 Snímače země

Pokud má být robot schopen efektivního pohybu v členitém terénu, musí mít možnost detekovat překážky při došlapování na zem. Jednou z možností je snímat obraz okolí pomocí kamery a detekovat překážky z obrazu. Nevýhodou tohoto přístupu je, že rozpoznávání obrazu není jednoznačné a je třeba odhadovat rychlost a směr pohybu robotu vzhledem k nasnímanému obrazu. Řešením tohoto problému je použití senzorů, které jsou schopny detekovat kontakt s objekty při došlapu končetiny.

Nejjednodušší způsob realizace těchto senzorů je umístění taktilních spínačů na konce končetin. Výhodou je snadná realizace a nízká cena, postačí jednoduchý spínač. Ten ovšem poskytuje pouze dva stavy – sepnuto nebo rozepnuto.

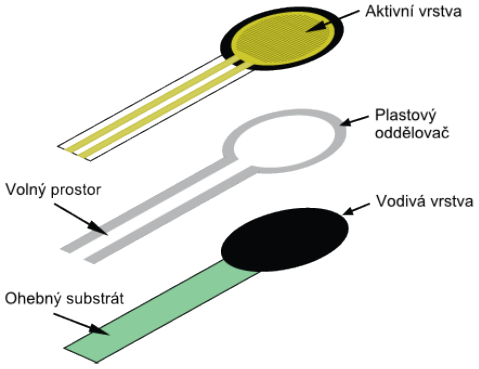
Vhodnějším řešením je užití senzoru, který umožňuje detekovat více různých stavů. Tím

[1] Staženo z http://buildbot.com.br/blog/wp-content/uploads/2015/01/Modulo-HC-SR04-Pinagem.jpg, dne 13. 5. 2015.

je například tlako-citlivý rezistor [2]. Jedná se o obyčejný polovodičový prvek, který mění svůj odpor v závislosti na tlaku na měřící plochu senzoru. Složení tlako-citlivého rezistoru ukazuje Obrázek 4.3a.

Tlako-citlivý rezistor je zapojen jako dělič napětí spolu s 10 KΩ rezistorem. Následně je možné získat hodnotu napětí na děliči pomocí A/D převodníku na mikrokontroléru. Výslednou hodnotu je možné využít jednak k určení překážky při došlapu končetiny, ale také ke kalibraci servomotorů. Rozložení váhy robotu není kvůli nepřesnostem v konstrukci rovnoměrné. To je třeba kompenzovat pomocí servomotorů. Tlako-citlivé rezistory poskytují hodnotu aktuálního zatížení každé končetiny. Je třeba nalézt takový stav, kdy budou všechny končetiny rovnoměrně zatíženy.

Tlako-citlivé rezistory jsou vestavěny do končetiny, aby se snížilo riziko jejich poškození. Do vnějšího profilu končetiny je vložen užší profil, který je opatřen šrouby na obou koncích. Do profilu jsou vyvrtány otvory oválného tvaru. Tento menší profil je uchycen pomocí dvou šroubů k vnějšímu profilu končetiny. Díky elipsovitým otvorům se může vnitřní profil posouvat nahoru a dolů. Při pohybu nahoru dochází ke kontaktu s tlako-citlivým rezistorem. Konstrukci snímače uvádí Obrázek 4.3b.



(a) Složení tlako-citlivého rezistoru. Ten se skládá ze tří vrstev. Horní vrstva je potištěná polovodičem a obsahuje konektory. Prostřední vrstva je plastový oddělovač, který vytváří prostor pro působení síly. Spodní vrstva je vodivá a podílí se s polovodičovou vrstvou na změně odporu senzoru.[2]

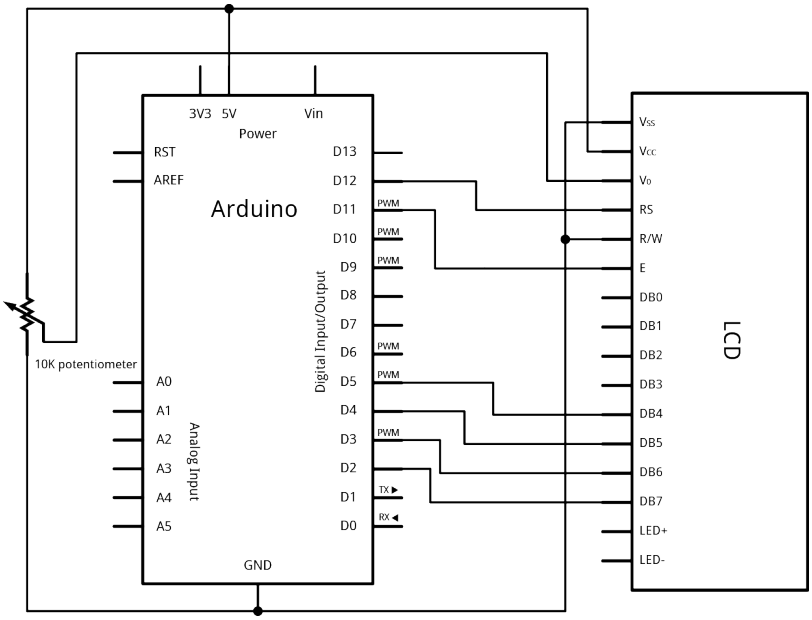
(b) Integrace snímačů země do konstrukce robotu. Vnitřní profil je zakončen šrouby po obou stranách a je pohyblivý nahoru a dolů. Ve směru nahoru se šroub opírá o tlako-citlivý rezistor. Tím dochází k vytvoření tlaku na senzoru, který je detekován a změřen mikrokontrolérem.

Obrázek 4.3: Složení tlako-citlivého rezistoru a integrace snímačů země do konstrukce.

4.1.4 LCD displej

Robot je vybaven LCD displejem, který zobrazuje základní informace o robotu. Použitý LCD displej má dva řádky po šestnácti znacích. Displej je připojen k mikrokontroléru pomocí 4-bitového módu. Řadič displeje je kompatibilní s řadičem Hitachi HD44780, pro který existuje řada knihoven. Zapojení displeje k mikrokontroléru znázorňuje Obrázek 4.4.

[2] Staženo z http://sensorwiki.org/lib/exe/fetch.php/sensors/fsr_diagram.png?w=400&h=&cache=cache, dne 14. 5. 2015.



Obrázek 4.4: Připojení LCD displeje k mikrokontroléru. Je použit 4-bitový mód. Displej vyžaduje také připojení pinu kontrastu na 10 KΩ potenciometr a případně pinů napájení podsvícení displeje.[3]

4.1.5 GPS senzor

Pro určení polohy robotu slouží GPS senzor [1]. Ten umožňuje navigaci ve venkovních prostorech. Senzor je připojen k mikrokontroléru pomocí sériové linky, po které posílá informace například o aktuální poloze či nadmořské výšce.

4.1.6 SD karta

Do výbavy robotu patří také SD karta, která slouží k ukládání konfiguračních souborů a záznamů o běhu systému. SD karta resp. její adaptér je připojen pomocí SPI sběrnice přímo k mikrokontroléru. Při startu robotu jsou načteny hodnoty z konfiguračních souborů. Adaptér je připojen na digitální piny 50 až 53. Více uvádí [6].

4.2 Kontrolní uživatelské rozhraní

Aby bylo možné robot snadno a efektivně ovládat, musí mít operátor neustálý přehled o datech ze senzorů a z řídícího systému robotu. K tomu slouží grafické uživatelské rozhraní (GUI), které umožňuje robot ovládat z libovolného počítače, zobrazuje data ze senzorů robotu, vizualizuje aktuální pozici robotu a umožňuje návrh vlastních pohybových algoritmů. Rozhraní také zobrazuje ladící informace z mikrokontroléru a zprávy zasílané mezi řídícím počítačem a robotem.

Grafické uživatelské rozhraní je naprogramováno v jazyce C++ s využitím knihovny Qt. Server běží ve vlastním vlákně, protože přijímá velké množství informací (každých 30 ms se aktualizuje pozice všech končetin, dále jsou zasílány informace ze senzorů dvakrát za sekundu a jiné). Po spuštění programu je vytvořena instance serveru v daném vlákně a je ak-

[3] Staženo z http://www.arduino.cc/en/uploads/Tutorial/LCD_schem.png, dne 14. 5. 2015.

tivováno broadcastové vysílání, které umožňuje klientům získat IP adresu serveru v dané síti.

Po spuštění programu jsou aktualizovány výchozí hodnoty z dat v konfiguračním souboru. V konfiguračním souboru je možné nastavit některé výchozí hodnoty uživatelského rozhraní, jak uvádí Tabulka 4.2.

Příkaz konf. souboru	**Popis**	**Výchozí hodnota**
broadcastPortIn	vstupní port broadcastoveho vysílání	45454
broadcastPortOut	výstupní port broadcastového vysílání	54545
serverPort	port serveru	9999
logging	zaznamenávání zpráv do souboru	**true** / false
showDebugMessages	zobrazení ladících zpráv	**true** / false
showInformMessages	zobrazení informačních zpráv	**true** / false
showWarningMessages	zobrazení varovných zpráv	**true** / false
showErrorMessages	zobrazení chybových zpráv	**true** / false
showAcceptMessages	zobrazení úspěšných zpráv	**true** / false

Tabulka 4.2: Příkazy konfiguračního souboru uživatelského rozhraní. Každý příkaz musí být uveden samostatně na začátku řádku ve tvaru "command=value", kde 'command' je název příkazu, který je konfigurován a 'value' je hodnota nastavená tomuto příkazu. Komentáře v konfiguračního souboru musí být uvozeny znakem '#'.

Konfigurační soubor musí být pojmenován jako "config.txt"a umístěn v adresáři programu. Každý z příkazů musí být na začátku samostatného řádku souboru ve tvaru "command=value", kde 'command' je název příkazu, který je konfigurován a 'value' je hodnota nastavená tomuto příkazu (viz Tabulka 4.2). Komentáře v konfiguračního souboru musí být uvozeny znakem '#'. Ty jsou pak při nahrávání souboru přeskočeny.

4.2.1 Komunikační protokol

Přenos dat mezi robotem a řídícím počítačem je realizován textovým protokolem. Obrázek 4.5 ukazuje, jaké zprávy jsou zasílány v průběhu počáteční komunikace mezi serverem (uživatelské rozhraní) a klientem (robot). Klient po připojení k serveru žádá o přidělení identifikátoru, který následně používá ke komunikaci se serverem. Během prvotní fáze komunikace jsou klientem odeslány také základní údaje o typu a případném jménu robotu.

Po dokončení inicializační fáze je klient připojen k serveru a může být zahájena příslušná komunikace. Klient zasílá údaje ze senzorů a aktuální pozice končetin. Součástí zprávy je také identifikátor, který byl klientovi přidělen serverem v počátcích komunikace. Ten slouží k identifikaci zpráv od jednotlivých klientů. Následně je možné zobrazovat pouze data od vybraného klienta a přepínat mezi jednotlivými roboty. Zprávy od klienta jsou ve tvaru "id:TYPE:MESSAGE", kde 'id' je identifikátor klienta přidělený v prvotní komunikaci se serverem (viz Obrázek 4.5), 'TYPE' je typ zprávy, který udává, jakou akci má server provést, a 'MESSAGE' je tělo zprávy, které obsahuje doplňující informace o dané akci nebo jiná data. Přehled příkazů, které umí klient přijmout, popisuje Tabulka 4.3.

Server zasílá příkazy, které byly zadány uživatelem v ovládacím rozhraní. Pro každý příkaz uživatele je vygenerována zpráva ve tvaru "CMD:TYPE:MESSAGE", kde 'TYPE'

Příkaz	Akce	Ukázka
NEW	Uloží identifikátor	NEW:1
TYPE	Odpoví typem robotu	TYPE
NAME	Nastaví nové jméno	NAME
REMOVED	Ukončí spojení a zastaví MCU	REMOVED
CMD:SERVOS	Přepošle zprávu do MCU	CMD:SERVOS:0=30
CMD:GAIT	Uloží a pošle do MCU pohybový alg.	CMD:GAIT:1
CMD:MCU	Přepošle jako příkaz do MCU	CMD:MCU:command
CMD:DEFAULTS	Pošle výchozí hodnoty motorů do MCU	CMD:DEFAULTS:0=5

Tabulka 4.3: Příkazy, které umí klient přijmout, jejich význam a formát. Každý z příkazů musí být ukončen speciální ukončovací sekvencí tvaru "--end--".

je typ akce, kterou má klient vykonat a 'MESSAGE' je tělo zprávy, které obsahuje doplňující informace o dané akci. Schéma zpráv zobrazuje Obrázek 4.6. Přehled příkazů, které umí server přijmout popisuje Tabulka 4.4.

Příkaz	Akce	Ukázka
NEW	Zašle nový identifikátor	:NEW
CONFIRM ID	Potvrdí ID, žádá o typ	1:CONFIRM ID
TYPE	Uloží typ k robotu	1:TYPE:Hexapod
NAME	Uloží jméno k robotu	1:NAME:Jmeno
SONARS	Nastaví hodnotu sonaru v UI	1:SONARS:1=25,2=12
FSR	Nastaví hodnotu nášlapných snímačů v UI	1:FSR:0=456,15=0
ENCODERS	Nastaví hodnotu enkodérů v UI	1:ENCODERS:1=47
SERVOS	Nastaví posuvníky a model robotu v UI	1:SERVOS:0=0,17=18
DEBUG	Zobrazí ladící zprávu v UI	1:DEBUG:Zprava
ERROR	Zobrazí chybovou zprávu v UI	1:ERROR:Zprava
ACCU	Nastaví hodnotu napětí akumulátoru v UI	1:ACCU:11,83
WIFI	Nastaví hodnotu síly signalu Wi-Fi v UI	1:WIFI:98
ELECTRONICS	Nastaví hodnotu napětí elektroniky v UI	1:ELECTRONICS:4,9

Tabulka 4.4: Příkazy, které umí server přijmout, jejich význam a formát. Každý z příkazů musí být ukončen speciální ukončovací sekvencí tvaru "--end--".

Klient následně komunikuje s mikrokontrolérem a přeposílá data ze senzorů na server a příkazy serveru do mikrokontroléru. Přehled příkazů, které umí mikrokontrolér přijmout popisuje Tabulka 4.5. Každý z příkazů musí být ukončen speciální ukončovací sekvencí tvaru "--end--".

Komunikace mezi serverem a klientem probíhá pomocí TCP a UDP schránek. Při navazování inicializační spojení jsou použity TCP schránky, protože je třeba zajistit spolehlivý

přenos všech dat. Naopak při zasílání poloh končetin a dat ze senzorů je důležité zachování rychlosti a občasné výpadky dat nejsou problém, protože hodnoty se aktualizují každých 30 ms.

Příkaz	**Akce**	**Ukázka**
HexapodStart	Zahájí zasílání dat ze senzorů	HexapodStart
HexapodEnd	Ukončí zasílání dat ze senzorů	HexapodEnd
SERVOS	Nastaví servomotory do zadané pozice	SERVOS:0=5,17=180
FORWARD	Pohyb vpřed	FORWARD
BACKWARD	Pohyb vzad	BACKWARD
RIGHT	Pohyb vpravo	RIGHT
LEFT	Pohyb vlevo	LEFT
STOP	Zastaví pohyb	STOP
PACK	Sbalí končetiny	PACK
UNPACK	Rozbalí končetiny	UNPACK
DEFAULTS	Nastaví výchozí hodnoty servomotorů	DEFAULTS
RESET	Nastaví servomotory do výchozích pozic	RESET
TripodGait	Nastaví pohybový algoritmus tripod	TripodGait
WaveGait	Nastaví pohybový algoritmus vlna	WaveGait
RippleGait	Nastaví pohybový algoritmus vlnění	RippleGait
NoneGait	Nastaví pohybový algoritmus na žádný	NoneGait

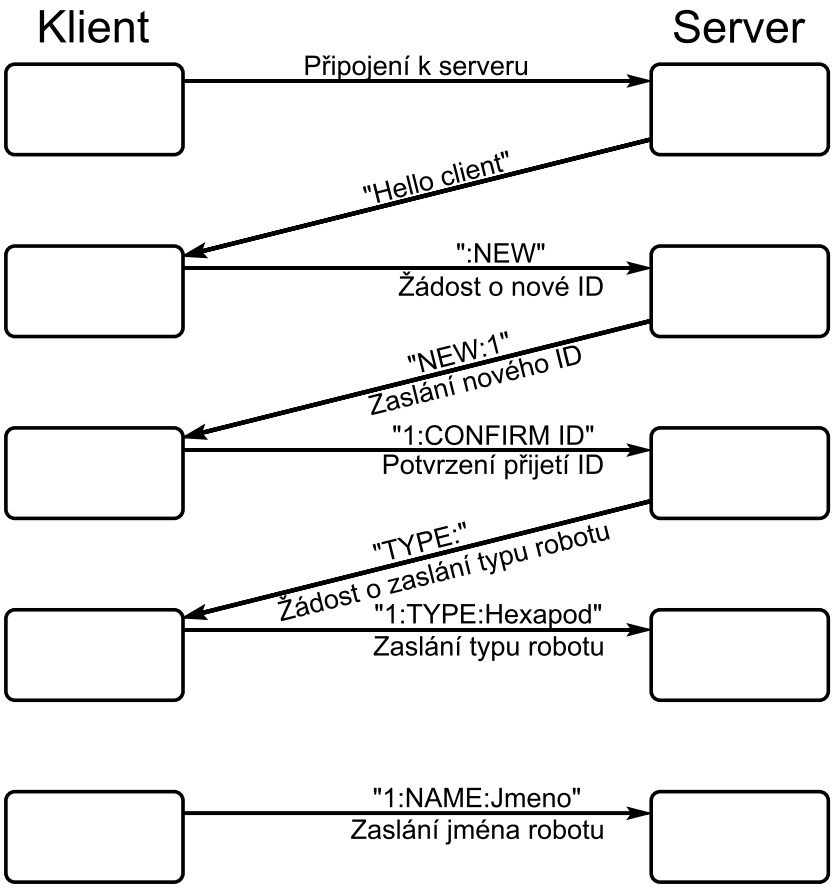
Tabulka 4.5: Příkazy, které umí mikrokontrolér přijmout, jejich význam a formát. Každý z příkazů musí být ukončen speciální ukončovací sekvencí tvaru "--end--".

4.2.2 Monitorování robotu

Součástí uživatelského rozhraní jsou také prostředky, které zobrazují data ze senzorů robotu jako jsou sonary, enkodéry, tlako-citlivé rezistory, síla signálu Wi-Fi na robotu, napětí akumulátoru, napětí na 5 V okruhu elektronických komponent a GPS souřadnice robotu. K dispozici je také informace o aktuálních pozicích servomotorů ve stupních a grafická reprezentace aktuálních poloh končetin robotu. Data jsou zasílána z mikrokontroléru dvakrát za sekundu, aktuální pozice končetin potom každých 30 ms. Data jsou zasílána pomocí UDP schránek, které zajišťují rychlé doručení dat.

Uživatelské rozhraní poskytuje také okno zpráv, kde jsou zobrazovány informace o běhu systému a příchozí zprávy od klienta. Okno podporuje zprávy několika typů: ladící, varovné, chybové, informační a úspěšné. Monitorovací obrazovka společně s oknem zpráv je zobrazena na Obrázku 4.7a.

Uživatelské rozhraní umožňuje také vizualizovat aktuální pozice končetin robotu. Data jsou odesílána z robotu každých 30 ms. Díky tomu vzniká animace pohybu končetin robotu se 33 snímky za sekundu. Rozhraní nabízí dva různé pohledy na robot. První pohled je zezadu a jsou zde vidět pohyby druhého a třetího kloubu jednotlivých končetin ve směru

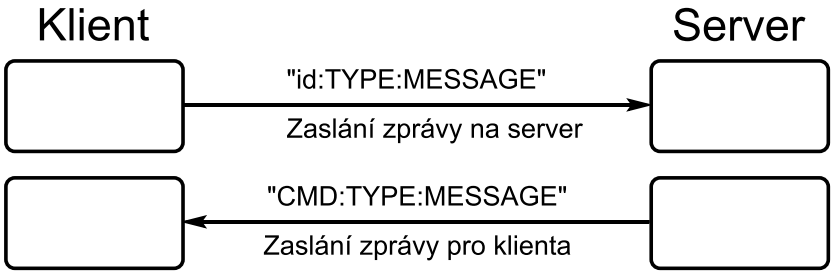


Obrázek 4.5: Schéma komunikace během připojování robotu k serveru uživatelského rozhraní. Klient (robot) získá IP adresu serveru z broadcastového vysílání a zahájí připojování k serveru (program uživatelského rozhraní), který zasílá uvítací zprávu "Hello client". Následně klient žádá o přidělení nového identifikátoru zasláním příkazu ":NEW". Pokud má server volnou kapacitu, vygeneruje nový identifikátor a zašle jej klientovi zprávu "NEW:1", kde '1' je nový identifikátor. Klient potvrdí příjem identifikátoru zasláním zprávy "CONFIRM ID". Po obdržení potvrzení od klienta žádá server o zaslání typu robotu prostřednictvím zprávy "TYPE:". Klient odpovídá zprávou obsahující typ robotu "1:TYPE:Hexapod", kde '1' je identifikátor klienta. Volitelně může klient zaslat také jméno robotu užitím zprávy "1:NAME:Jmeno".

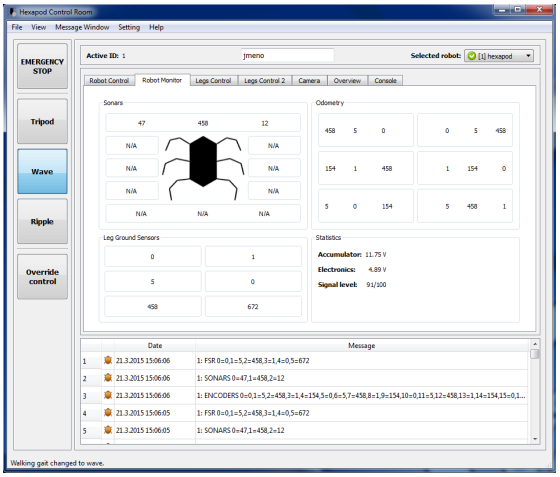
nahoru a dolů. Druhý pohled je ze shora, kdy je viditelný pohyb prvního kloubu končetiny ve směru dopředu a dozadu. Vizualizaci lze také využít při tvorbě uživatelských pohybových pro testovací účely (více viz Kapitola 4.2.4).

4.2.3 Ovládání robotu

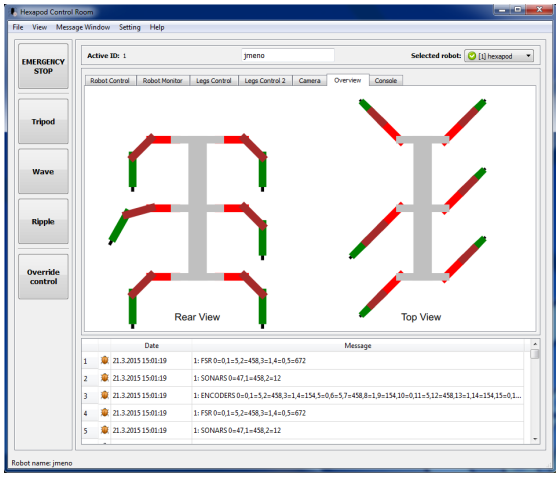
Z uživatelského rozhraní je možné robot nejen monitorovat, ale také ovládat. Nejprve je třeba převzít ovládání robotu pomocí tlačítka v levé části uživatelského rozhraní. V záložce řízení robotu (viz Obrázek 4.8a) je možné určovat základní směry pohybu robotu, jako je pohyb vpřed, vzad, vlevo, vpravo a rotace vpravo a vlevo. Dále je možné zvolit výšku těla robotu od země. Některá tlačítka je také možné ovládat pomocí klávesových zkratek.



Obrázek 4.6: Schéma komunikace při zasílání příkazů. V případě, že klient (robot) zasílá data na server (uživatelské rozhraní), je vytvořena zpráva ve tvaru "id:TYPE:MESSAGE", kde 'id' je identifikátor klienta přidělený v prvotní komunikaci se serverem (viz Obrázek 4.5), 'TYPE' je typ zprávy, který udává, jakou akci má server provést, a 'MESSAGE' je tělo zprávy, které obsahuje doplňující informace o dané akci nebo jiná data. V případě, kdy server zasílá zprávu klientovi, je vygenerována zpráva ve tvaru "CMD:TYPE:MESSAGE", kde 'TYPE' je typ akce, kterou má klient vykonat, a 'MESSAGE' je tělo zprávy, které obsahuje doplňující informace o dané akci.



(a) Monitorovací obrazovka v uživatelském rozhraní. K dispozici jsou informace ze sonarů, tlako-citlivých rezistorů, enkodérů, stav baterie, síla Wi-Fi signálu a GPS souřadnice robotu.



(b) Vizualizece pohybu končetin v uživatelském rozhraní. Robot je zobrazován ze dvou různých pohledů – shora a zezadu. Každý z pohledů zachycuje jiný pohyb končetin.

Obrázek 4.7: Monitorovací obrazovka a vizualizece pohybu končetin v uživatelském rozhraní.

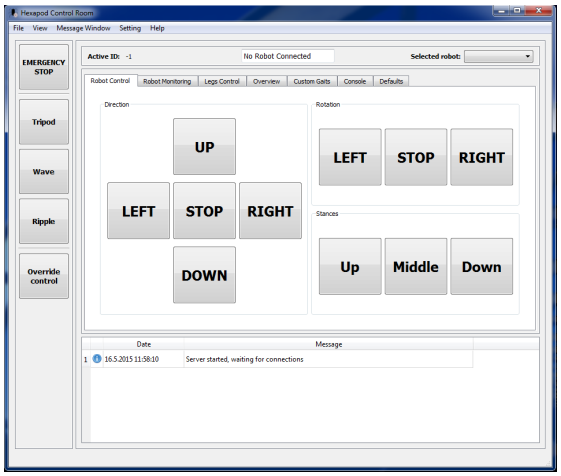
Další ze záložek uživatelského rozhraní poskytuje osmnáct posuvníků. Každý posuvník ovládá jeden servomotor. Po nastavení nové hodnoty je vygenerován požadavek na změnu polohy příslušného servomotoru, který je odeslán do robotu. Každý ze servomotorů je možné nastavit do pozice v rozsahu 0 ° až 180 ° s rozlišením jednoho stupně.

4.2.4 Vytváření vlastních pohybových algoritmů

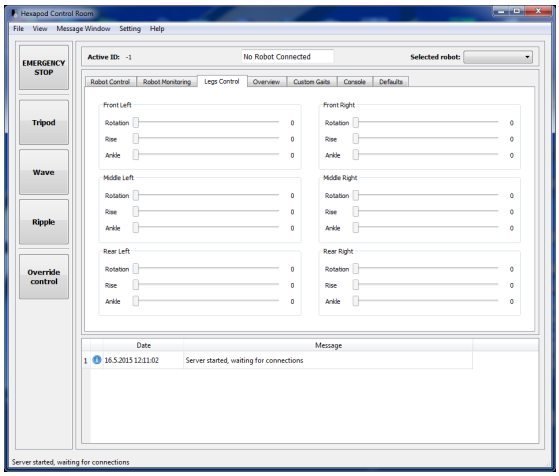
Pokud uživateli nevyhovují již předdefinované pohybové algoritmy, je možné využít další z nástrojů, které uživatelské rozhraní poskytuje – generátor uživatelských pohybových al-

goritmů. V tomto nástroji je možné přidat několik kroků nového pohybového algoritmu, nastavit čas, ve kterém se má daný krok provést, zvolit končetiny a směr jejich pohybu.

Následně je z vytvořených kroků vygenerována sestava, kterou je možné simulovat v uživatelském rozhraní. Průběh simulace je také vizualizován v záložce s modelem robotu (viz Obrázek 4.7b). Simulaci je také možné provádět přímo na robotu, kdy simulace přímo řídí jednotlivé servomotory robotu.



(a) Ovládací obrazovka robotu. Pomocí tlačítek lze robot ovládat a určovat směr jeho pohybu. V pravé části je možná rotace robotu vpravo či vlevo a dokonce možnost nastavení výšky těla robotu od země. Některá z tlačítek lze také obsluhovat pomocí přiřazených klávesových zkratek.

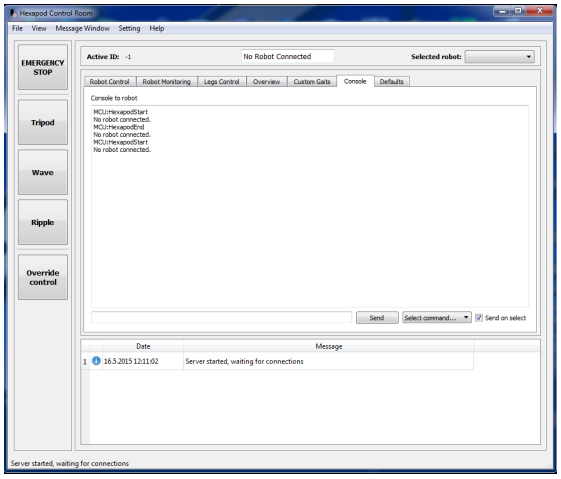


(b) Pomocí posuvníků je možné ovládat každý z 18 servomotorů s rozlišením jednoho stupně. Při změně hodnoty posuvníku je ihned vygenerován požadavek na změnu polohy daného servomotoru. Ten je následně zaslán robotu a proveden. Posuvníky ukazují také aktuální pozice servomotorů podle zaslaných zpráv od robotu.

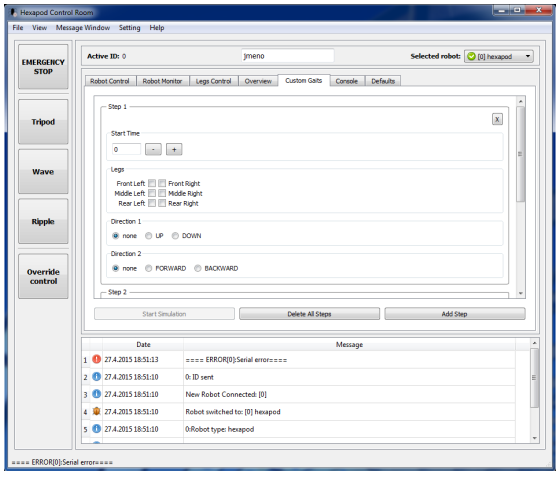
Obrázek 4.8: Ovládací obrazovka a obrazovka posuvníku jednotlivých servomotorů v uživatelském rozhraní.

4.3 Knihovna pro komunikaci s klientem

Součástí softwaru je knihovna napsaná v C++ s využitím knihovny Qt, která slouží ke komunikaci s klientem (robotem). Knihovna umožňuje vytvořit spojení uživatelského programu s klientem a volitelně také inicializovat prvotní komunikaci mezi klientem a uživatelským programem (viz Obrázek 4.5). Následně stačí v uživatelském programu definovat pouze jeden slot pro příjem dat z klienta. Knihovna poskytuje také rozhraní pro zasílání dat klientu. Kód 4.1 a kód 4.2 ukazuje možné použití knihovny v uživatelském programu. Nejprve je vytvořen nový objekt serveru, následně je připojen signál serveru ke slotu, který zajišťuje čtení dat v uživatelské aplikaci, dále je nastaven příznak 'SetNegotiateConnection' na 'true', což signalizuje knihovně, že se má provést také inicializace spojení (výměna údajů o typu robota podle Obrázku 4.5). Také je zde vidět slot uživatelského programu, který využívá funkci knihovny pro deserializaci příchozích dat. Knihovna podporuje pouze jednoho klienta a následující příkazy: "NEW", "CONFIRM ID", "TYPE"a "NAME".



(a) Uživatelské rozhraní nabízí konzoli, pomocí které je možné přímo zasílat příkazy do robotu. Příkazy mohou být pouze pro Raspberry Pi nebo dokonce přímo pro mikrokontrolér. V rozhraní je nastaveno několik předdefinovaných příkazů pro základní akce robotu. Je možné také nastavit odeslání ihned po výběru příkazu.



(b) Záložka uživatelského rozhraní, která umožňuje tvorbu uživatelských pohybových algoritmů. Je možné přidat několik kroků a nastavit směr pohybu vybraným končetinám. Následně je možná simulace v záložce vizualizující končetiny robotu nebo je možné robot přímo řídit daným algoritmem.

Obrázek 4.9: Obrazovka konzole robotu a obrazovka tvorby uživatelských pohybových algoritmů v uživatelském rozhraní.

```
1  MainWindow::MainWindow(QWidget *parent) : QMainWindow(parent)
2  {
3     // vytvoření nového objektu
4     s = new HexapodServerLib(this);
5
6     // připojení signálu knihovny na slot, který zajišťuje čtení dat
7     connect(s, SIGNAL(readyRead(QString)), this, SLOT(readServerData(QString)));
8
9     // knihovna zařídí také počáteční inicializaci spojení
10    s->SetNegotiateConnection(true);
11
12    // spuštění serveru, který čeká na připojení klienta (robotu)
13    s->serverStart();
14 }
15
16 void MainWindow::readServerData(QString message)
17 {
18     qDebug() << "SLOT:" << s->ParseMessage(message);
19 }
```

Kód 4.1: Připojení uživatelské aplikace k serveru pomocí knihovny.

```
#include "hexapodserverlib.h"

/**
 * @brief The MainWindow class
 */
class MainWindow : public QMainWindow
{
    Q_OBJECT

public:
    explicit MainWindow(QWidget *parent = 0);
    ~MainWindow();

private slots:
   // slot pro čtení dat ze serveru v uživatelské aplikaci
    void readServerData(QString message);

private:
   // objekt serveru
    HexapodServerLib *s;
};
```

Kód 4.2: Připojení uživatelské aplikace k serveru pomocí knihovny - hlavičkový soubor.

4.4 Program pro Raspberry Pi

Druhou velkou částí software je program vytvořený pro Raspberry Pi [30]. Ten je napsán v jazyce C++ s užitím knihovny Qt. K dispozici jsou dvě mutace tohoto programu. První z nich je aplikace s uživatelským rozhraním, která zobrazuje základní informace o stavu robotu a obsahuje okno zpráv, ve kterém je zaznamenávána komunikace, kterou Raspberry Pi obdrží jak od řídícího počítače, tak od mikrokontroléru. Druhou mutací programu je obyčejná konzolová aplikace, která cílí na vysoký výkon programu a je vhodná pro finální nasazení na Raspberry Pi, protože pro svůj běh vyžaduje pouze jedno vlákno, které se může plně věnovat komunikaci, a nemusí aktualizovat grafické rozhraní aplikace.

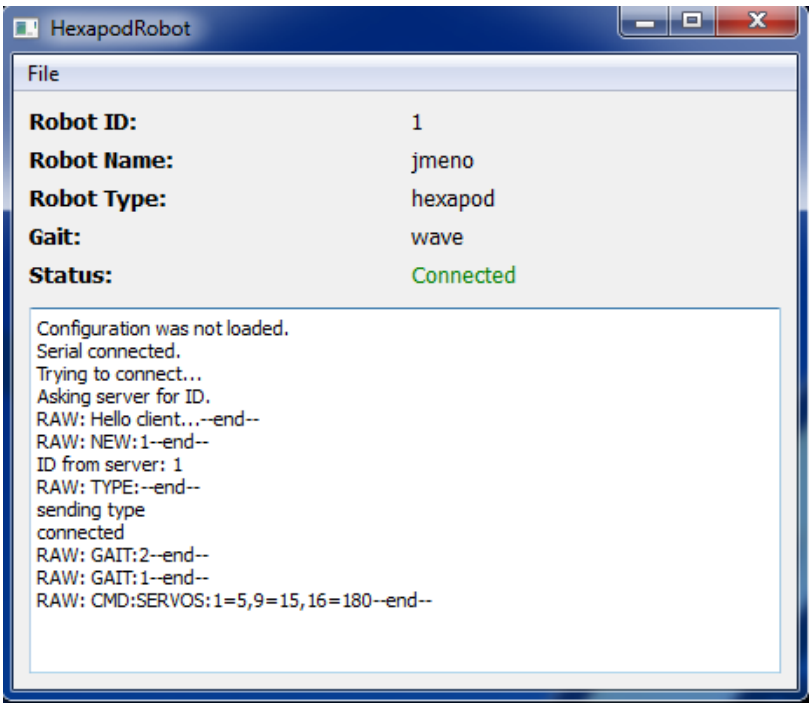
Aplikace zpracovává data ze sériové linky, na níž je připojen mikrokontrolér, a data ze sítě, která zasílá program uživatelského rozhraní. Nastavení některých parametrů aplikace je možná pomocí konfiguračního souboru. Ten musí mít název "config.txt"a musí být umístěn v adresáři aplikace. Parametry, které je možné pomocí konfiguračního souboru nastavit, popisuje Tabulka 4.6.

4.5 Program pro mikrokontrolér

Poslední, ale neméně důležitou, částí je program pro mikrokontrolér Atmega2560 [7], který je integrovaný na vývojovou desku Arduino Mega 2560 [3]. Ten ovládá většinu periferií jako jsou servomotory a senzory a komunikuje po sériové lince s Raspberry Pi. Pro práci s mikrokontrolérem je použito funkcí platformy Arduino.

Příkaz konf. souboru	Popis	Výchozí hodnota
broadcastPortIn	vstupní port broadcastového vysílání	45454
broadcastPortOut	výstupní port broadcastového vysílání	54545
serverPort	port serveru	9999
logging	zaznamenávání zpráv do souboru	**true** / false
comPortName	nazev sériového portu	COM1

Tabulka 4.6: Příkazy konfiguračního souboru pro Raspberry Pi. Každý příkaz musí být uveden na samostatně na začátku řádku ve tvaru "command=value", kde 'command' je název příkazu, který je konfigurován, a 'value' je hodnota nastavená tomuto příkazu. Komentáře v konfiguračního souboru musí být uvozeny znakem '#'.



Obrázek 4.10: Grafická verze programu pro Raspberry Pi. Aplikace zobrazuje základní informace o stavu robotu jako je identifikátor, jméno a typ robotu, zvolený pohybový algoritmus a stav připojení k serveru. Ve spodní části aplikace je okno zpráv, které zobrazuje jednotlivé zprávy komunikace.

4.5.1 Ovládání servomotorů

Robot je vybaven osmnácti servomotory typu HS-5485HB a HS-5645MG, třemi na každé končetině. Jedná se o standardní modelářské servomotory, které jsou řízeny pulzem o šířce od 1 ms do 2 ms s periodou 20 ms, kde 1 ms odpovídá pozici 0 ° a 2 ms odpovídají pozici 180 °. Oba typy servomotorů jsou digitální, a proto je možné tyto motory programovat – měnit některé jejich parametry jako je směr rotace nebo rychlost pohybu servomotoru.

Vygenerovat na pinech mikrokontroléru signál s šířkou pulzu 1 ms je možné několika způsoby. Prvním z nich je vytvoření programu, který nastavuje na výstup digitálního

pinu příslušnou hodnotu a zbytek času aktivně čeká. Ukázku takového přístupu poskytuje Kód 4.3. Tento způsob řízení je sice funkční, ovšem v praxi nepoužitelný, protože mikrokontrolér aktivně čeká, a proto nemůže provádět další akce či obsluhovat jiné periferie.

```
// číslo pinu, na kterém bude signál generován
int servoPin = 1;

// inicializace
void setup()
{
   // nastavení pinu jako výstupní
   pinMode(servoPin, OUTPUT);
}

// nekonečná smyčka
void loop()
{
   // nastavení úrovně na 1
   digitalWrite(servoPin, HIGH);

   // aktivní čekání 1 ms
   delay(1);

   // nastavení úrovně na 0
   digitalWrite(servoPin, LOW);

   // aktivní čekání 20 ms
   delay(19);
}
```

Kód 4.3: Použití aktivního čekání pro generování řídících pulzů servomotoru.

Proto je třeba použít neblokující způsob, který umožní provádění také jiných akcí. Řešením může být použití hardwarového řadiče, který umí řídit více servomotorů současně, a přitom neblokuje program mikrokontroléru, protože se jedná o samostatný hardwarový prvek. Řadiče jsou obvykle obsluhovány pomocí příkazů, které jsou jim zasílány pomocí sběrnice. Tou je nejčastěji sériová linka nebo I_2C sběrnice.

Další možností je použití časovače mikrokontroléru. Jedná se o prostředek, který umožňuje provádění nějaké operace jednou za specifikovaný časový interval. Použití časovače popisuje Kód 4.4. Takto inicializovaný časovač generuje signál s periodou 20 ms na výstupní pin mikrokontroléru. Je také možné nastavit časovač tak, aby generoval přerušení a následně nastavit požadovanou změnu na výstupním pinu. Pro řízení motoru v tomto projektu byla použita knihovna VarSpeedServo [26], která používá právě časovače mikrokontroléru a umí také řídit rychlost pohybu servomotorů.

4.5.2 Měření vzdálenosti

Pro detekci překážek robot disponuje ultrazvukovými dálkoměry – sonary. Pro změření vzdálenosti od první zachycené překážky je třeba aktivovat měření sonaru. To je provedeno zasláním krátkého pulzu na pin 'Trig'. Sonar následně vyšle několik ultrazvukových vln. K získání vzdálenosti je třeba změřit délku pulzu, který je vygenerován na pinu `'Echo'` po zahájení měření. Délku tohoto pulzu je možné určit několika způsoby.

```
// inicializace
void setup()
{
   // frekvence = 50 Hz ~ perioda 20 ms
   IRC1 = 20000;

   // prescaler 8, zdroj hodiny
   TCCR1B = _BV(CS11) | _BV(WGM13);

   // nastavení přičítání a nulování počítadla
   TCCR1A = _BV(COM1A1) | _BV(COM1B1);

   // počáteční hodnota počítadla
   OCR1A = 50;

   // výstupní pin
   DDRD = _BV(PD4);
}

// nekonečná smyčka
void loop()
{
}
```

Kód 4.4: Použití časovače mikrokontroléru pro generování řídících pulzů servomotoru.

První možností je užití funkce `pulseIn` [4], která dokáže určit délku pulzu. Nevýhodou je, že tato funkce je blokující. Pokud je objekt daleko od sonaru, dochází k blokování hlavní smyčky programu, což je nežádoucí. Délku tohoto čekání lze zkrátit nastavením třetího volitelného parametru funkce `pulseIn` – `timeout`, který definuje maximální dobu, po kterou má funkce čekat na konec měřeného pulzu. Stále ovšem dochází k blokování hlavní smyčky.

Řešením je užití externího přerušení, které je vygenerováno, pokud dojde k předdefinované události na zadaném pinu mikrokontroléru. Ukázku uvádí Kód 4.5. Pomocí funkce `attachInterrupt` je navázáno spuštění funkce `sonar`, kdykoli dojde ke změně z 0 na 1 na zadaném pinu. Dojde k uložení aktuálního času mikrokontroléru, změně stavu a k nastavení přerušení tak, aby reagovalo při změně z 1 na 0. Při dalším spuštění funkce `sonar` je z uplynulého času spočítána vzdálenost od nejbližší překážky.

Samotný výpočet vzdálenosti vychází z rychlosti zvuku ve vzduchu. Ta je 343.2 m/s v suchém vzduchu při teplotě 20 °C. Pokud se zvuková vlna vrátí za 853 µm, pak urazila přibližně 2 metry. Je ovšem třeba vzít v úvahu cestu tam i zpět, protože se jedná o odraženou vlnu. Vzdálenost od překážky je tedy 1 metr. Z tohoto vztahu je možné odvodit konstantu 58, kterou stačí následně podělit výsledný čas v mikrosekundách, a získat vzdálenost v centimetrech. Mezi jednotlivými měřeními by měl být rozestup alespoň 50 ms, protože jinak může docházet k chybnému měření kvůli přeslechům senzoru. Tento časový rozestup je vhodné dodržet i v případě, kdy jsou sonary vzájemně orientovány opačným směrem.

```
// echo pin, pin musí podporovat externí přerušení
int interruptPin = 0;

// stav měření
int state = 0;
long start = 0;

// inicializace
void setup()
{
   // počáteční nastavení přerušení na echo pin, funkci 'sonar' a náběžnou
       hranu (0 -> 1)
   attachInterrupt(interruptPin, sonar, RISING);
}

// funkce volaná při přerušení
void sonar()
{
   if (state == 0)
   {
      attachInterrupt(interruptPin, sonar, FALLING);
      state = 1;
      start = micros();

   }
   else if (state == 1)
   {
      state = 0;

      // vzdálenost v cm
      int distance = (micros() - start) / 58;
   }
}
```

Kód 4.5: Použití externího přerušení mikrokontroléru pro měření vzdálenosti sonarem.

4.5.3 Tlako-citlivé rezistory

Tlako-citlivé rezistory jsou použity k implementaci snímačů země. Ty jsou schopny detekovat překážku pod končetinu při jejím pokládání končetiny a zastavit pohyb servomotorů. Senzory jsou připojeny k mikrokontroléru pomocí napěťového děliče. Díky tomu je možné využít A/D převodník mikrokontroléru a měřit napětí na rezistorech. Na základě naměřeného napětí se určí tlak na senzor a vyhodnotí se, zda je v cestě překážka či nikoli.

Výstup ze snímačů země je také použit pro kalibraci servomotorů, kdy je hledána taková pozice servomotorů, při níž jsou jednotlivé končetiny rovnoměrně zatíženy. Rovnoměrné rozložení váhy robotu na všechny končetiny je důležité, aby nedocházelo k přetěžování některého ze servomotorů.

4.5.4 Ostatní periferie

Pro ovládání zbývajících periferií robotu jsou použity knihovny, které poskytuje platforma Arduino.

LCD displej

LCD displej osazený na robotu je kompatibilní s řadičem Hitachi HD44780, a je tedy možné pro jeho ovládání použít knihovnu, která je standardním prvkem platformy Arduino. Po inicializaci displeje je možné zapisovat znaky na požadované pozice displeje. Ukázku použití displeje demonstruje Kód 4.6.

```
// LCD displej
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

// inicializace
void setup()
{
   // nastavení velikosti displeje (16 znaků, 2 řádky)
   lcd.begin(16, 2);

   // nastavení kurzoru pro zápis na displej
   lcd.setCursor(0, 3);

   // zápis na displej
   lcd.print("Hello, world!");
}
```

Kód 4.6: Použití LCD displeje na platformě Arduino.

SD karta

SD karta slouží k ukládání konfiguračních a záznamových souborů robotu. Pro její ovládání je použita SD [5]. Použití knihovny demonstruje Kód 4.7.

```
// inicializace
void setup()
{
    // inicializace SD karty
    pinMode(52, OUTPUT);
    pinMode(53, OUTPUT);
    if (!SD.begin(53))
    {
      // chyba
    }

    // otevři soubor
   cFile = SD.open("soubor.txt");
    if (cFile)
    {
        while (cFile.available())
        {
         // načti znak ze souboru
         c = cFile.read();
      }
    }
}
```

Kód 4.7: Použití knihovny pro práci s SD kartou.

GPS modul

Robot je vybaven také GPS modulem. Ten je k mikrokontroléru připojen pomocí sériové linky, po jejíž inicializaci je možné číst data, která modul zasílá. Tato data obsahují informace o aktuální poloze, času, nadmořské výšce apod.

4.5.5 Řízení programu

Při práci s mikrokontrolérem je důležité, aby nedocházelo k aktivnímu čekání. Pokud totiž hlavní smyčka obsahuje prodlevu, v níž procesor jen čeká, může procesor nezpracovat některé impulzy, které periferní zařízení zasílají. Existuje několik způsobů, jak realizovat určité události v konkrétním čase, případně tyto události pravidelně opakovat.

Prvním přístupem je měření času, kdy se v programu vytvoří virtuální časomíra, které je porovnávána se skutečným časem mikrokontroléru. Pokud hodnota časomíry přesáhne skutečný čas, je provedena daná událost. Výhodou této varianty je snadná implementace. Nevýhodou je jistá míra nepřesnosti.

Jinou variantou je užití časovače. Tento přístup je zvláště vhodný pro periodicky se opakující akce, kdy časovač vždy po určitém čase vygeneruje přerušení, které spustí předdefinovanou rutinu, která provede příslušné akce. Tento postup je přesnější oproti předchozí variantě. Je možné tyto přístupy také kombinovat, kdy k přerušení dochází pravidelně a v programu jsou vytvořena počítadla, která provedou akci vždy, například každé desáté přerušení časovače.

Pro řízení programu robotu je použito obou těchto způsobů. Časovač pomocí přerušení signalizuje, kdy je třeba odeslat aktuální polohu všech končetin. U dat ze senzorů není přesnost tolik kritická a jsou použita počítadla a následné porovnání s časem mikrokontroléru.

Řízení pohybových algoritmů

Pohybový algoritmus je sekvence pohybů jednotlivých končetin. Tyto pohyby lze rozložit na změny natočení servomotorů. Tyto pohyby musí být vykonány v přesně stanovený čas, aby následný pohyb končetin odpovídal stanovenému pohybovému algoritmu. K docílení takového chování je použit algoritmus, který vychází z algoritmu next-event [29]. Nejprve jsou jednotlivé pohyby uloženy do datové struktury – kalendáře – společně s údajem o času, kdy se má pohyb provést, a také s cílovou pozicí servomotoru. Kalendář je následně setříděn podle aktivačního času jednotlivých událostí. Následně je vždy vybrán první prvek (prvek s nejnižším aktivačním časem) z kalendáře a je okamžitě proveden. Současně dochází k posunu ukazatele na druhý prvek. Na rozdíl od next-event algoritmu jsou prvky v kalendáři ponechávány, protože po dokončení poslední události z kalendáře se pokračuje opět od začátku – prvním prvkem. To vyplývá z cykličnosti pohybových algoritmů. Kód 4.8 ukazuje, jak algoritmus funguje.

```
// inicializace kalendáře a vložení jednotlivých pohybů
calendar.Init();

// setřídění kalendáře
calendar.Sort();
,
// dokud se má robot pohybovat
while (isMoving)
{
   // vyber aktivní pohyb z kalendáře
   Item i = calendar.GetActiveItem();

   // pokud je aktuální čas větší roven času události
   if (time >= i.Time)
   {
      // provedení pohybu
      Execute(i);

      // posun ukazatel na další prvek
      calendar.Next;
   }
}
```

Kód 4.8: Modifikovaný next-event algoritmus, který řídí provádění pohybových algoritmů při chůzi robotu.

Kapitola 5

Rozšíření

V rámci projektu bylo identifikováno několik rozšíření nad rámec zadání, která by zlepšila parametry robotu nebo uživatelského rozhraní. Některá z těchto rozšíření byla analyzována a implementována, jiná jsou teprve v návrhu a jejich implementace je plánována v budoucnu.

5.1 Implementovaná rozšíření

Tato kapitola popisuje rozšíření, která byla implementována nad rámec zadání. Tato rozšíření jsou již plně funkční a připravena k použití. Jedná se o vizualizaci aktuálních pozic jednotlivých končetin robotu, nástroj pro tvorbu uživatelských pohybových algoritmů nebo snímače země, které jsou schopny detekovat překážku při pokládání končetiny.

5.1.1 Vizualizace pozic končetin v uživatelském rozhraní

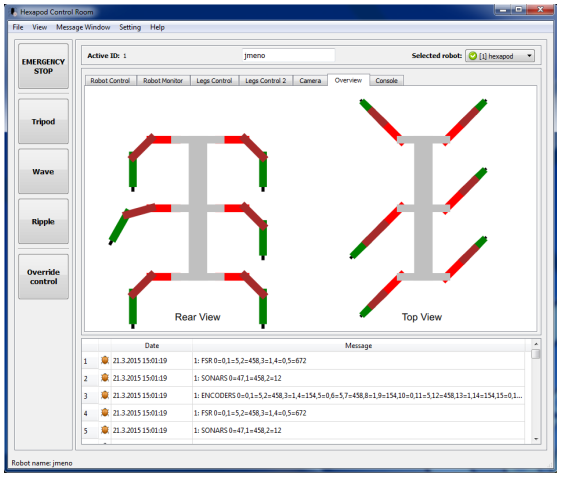
V rámci tohoto rozšíření byla do uživatelského rozhraní přidána nová záložka, která zobrazuje aktuální pozice všech končetin robotu, a to ze dvou pohledů: ze shora, kdy je zobrazován pohyb končetiny vpřed a vzad, a zezadu, kdy je vizualizován pohyb krajních dvou stupňů volnosti. Záložka uživatelského rozhraní je zobrazena na Obrázku 5.1a.

Pro dosažení plynulého pohybu končetin v okně uživatelského rozhraní je třeba aktualizovat pozici alespoň 25 krát za sekundu. Proto jsou aktuální pozice všech končetin odesílány z mikrokontroléru každých 30 ms.

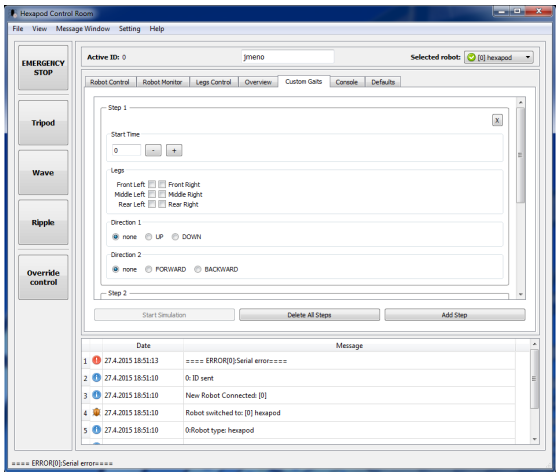
Zpráva z mikrokontroléru je odeslána prostřednictví Raspberry Pi do uživatelského rozhraní, kde dochází k jejich zpracování a aktualizaci QML objektu, který reprezentuje jednotlivé končetiny.

5.1.2 Knihovna serveru pro komunikaci s robotem

Součástí softwaru je knihovna napsaná v C++ s využitím knihovny Qt, která slouží ke komunikaci s klientem (robotem). Knihovna umožňuje vytvořit spojení uživatelského programu s klientem a volitelně také inicializovat prvotní komunikaci mezi klientem a uživatelským programem. Knihovna podporuje pouze jednoho klienta a následující příkazy: "NEW", "CONFIRM ID", "TYPE"a "NAME". Knihovnu je vhodné použít při implementaci uživatelského programu, který řídí a monitoruje robot.



(a) Záložka uživatelského rozhraní, která zobrazuje aktuální pozice končetin robotu. Pozice končetin jsou aktualizovány 30 krát za sekundu, což zajišťuje plynulost pohybu.



(b) Záložka uživatelského rozhraní, která umožňuje tvorbu uživatelských pohybových algoritmů. Je možné přidat několik kroků a nastavit pohyb vybraným končetinám.

Obrázek 5.1: Záložky uživatelského rozhraní, které rozšiřují možnosti programu.

5.1.3 Nástroj pro tvorbu vlastních pohybových algoritmů

Uživatelské rozhraní bylo také rozšířeno o nástroj pro tvorbu uživatelských pohybových algoritmů. Je možné přidat několik kroků pohybového algoritmu, nastavit čas zahájení daného kroku, jaké končetiny se budou pohybovat a jakým směrem. Následně je vygenerován seznam pohybů pro jednotlivé servomotory. Nově vytvořený pohybový algoritmus je možné pouze simulovat v uživatelském rozhraní, kdy je pozice končetin vizualizována v dané záložce programu (viz Obrázek 5.1a), nebo pomocí těchto pohybů přímo robot ovládat. Obrázek 5.1b zachycuje rozhraní nástroje pro tvorbu uživatelských pohybových algoritmů.

5.1.4 Snímače země

Robot v základní verzi není schopen chůze v členitém terénu, protože jeho končetiny se vždy vracejí do původní (nulové) pozice. Pro odstranění této nevýhody byly do končetin robota integrovány snímače země, které jsou schopny detekovat překážku během pokládání končetiny. Jakmile je překážka detekována, je pozastaven pohyb končetiny, která tak setrvá na překážce, a nedochází k nepřirozenému naklánění robotu. Senzory navíc zajišťují stabilní pozici těla vůči okolí, což je vhodné pro umístění některých senzorů, které jsou citlivé na otřesy nebo změnu horizontu.

Snímač je sestaven z tlako-citlivého rezistoru, který mění svůj odpor s tlakem. Senzor je součástí děliče napětí a pomocí A/D převodníku mikrokontroléru je možné získat aktuální hodnotu přítlaku. Konstrukce končetiny je uvedena na Obrázku 5.2.

5.1.5 Kamera

Robot je vybaven také kamerou, která je připojená k Raspberry Pi. Kamera je schopná snímat okolí ve Full HD kvalitě rychlostí 30 snímků za sekundu. Aktuálně jsou data z kamery k dispozici pouze na Raspberry Pi, ale do budoucna je plánován streaming do uživatelského

Obrázek 5.2: Konstrukce snímače země. Snímač země se skládá z tlako-citlivého rezistoru, který je integrován do končetiny robotu, aby nedošlo k mechanickému poškození senzoru. Končetina disponuje nášlapnou částí, která je pohyblivá v řádu desetin milimetru nahoru a dolů. To zlepšuje výsledky měření, protože je možné dosáhnout nulové hodnoty a neovlivňuje pohyb robotu.

rozhraní. Také bude možné data zpracovávat na řídícím počítači, který disponuje vyšším výkonem. Raspberry Pi umožňuje nad videem pouze základní operace.

5.1.6 Řízení pomocí modelářské RC soupravy

Robot je možné ovládat pomocí analogové nebo digitální RC modelářské soupravy (viz Obrázek 5.3b). Robot je vybaven RC přijímačem, který generuje na výstupních pinech stejný signál, jakým je řízen servomotor, tedy obdélníkový signál s délkou pulzu od 1 ms do 2 ms s periodou 20 ms. Délku pulzu měří mikrokontrolér pomocí externích přerušení, kdy je měřen čas mezi náběžnou a sestupnou hranou pulzu. Tento čas je následně převeden na odpovídající informaci o požadovaném směru pohybu.

5.1.7 3D model robotu

V rámci práce byl také vytvořen 3D model robotu v prostředí Autodesk Inventor 2015 [8]. Jedná se o model 1:1 s reálným robotem, který slouží k experimentování ve virtuálním prostředí. Navíc je model možné použít k výrobě jednotlivých komponent například pomocí 3D tisku nebo obrábění. Výhodou by byla vyšší míra přesnosti jednotlivých komponent. Výsledný 3D model robotu je uveden na Obrázku 5.4.

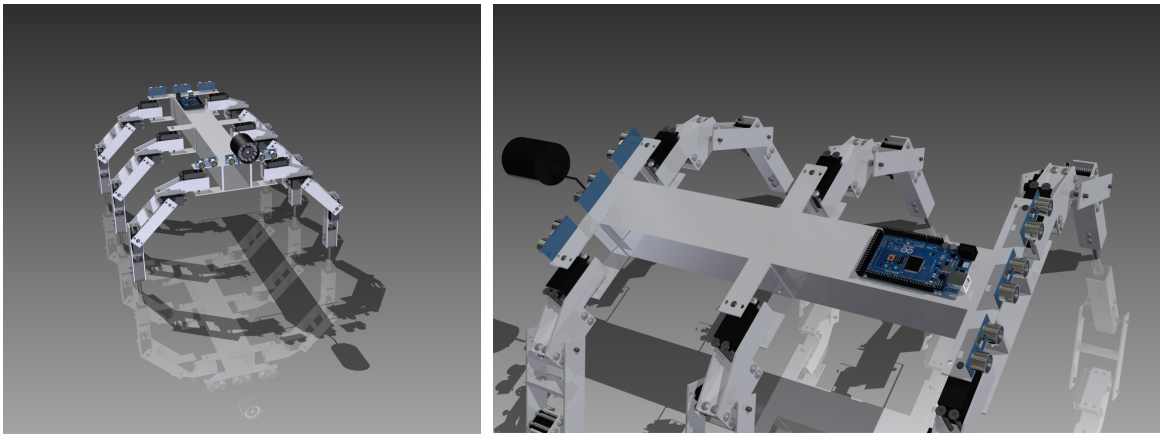
5.2 Plánovaná rozšíření

Robot není zdaleka dokončený a existuje mnoho oblastí, kde je možné něco vylepšit nebo rozšířit. Jedná se například o streamování obrazu z kamery do uživatelského rozhraní, napojení robotu na Robotický operační systém nebo rozšíření inteligentních vlastností robotu.

(a) Raspberry Pi kamera, která je připojena do desky Raspbbery Pi.[1]

(b) Modelářský RC vysílač. Pomocí něj je mimo jiné robot možné ovládat.[2]

Obrázek 5.3: Raspberry Pi kamera a modelářský RC vysílač.



Obrázek 5.4: Výsledné rendery 3D modelu robotu.

5.2.1 Streamování obrazu z kamery do uživatelského rozhraní

Kamera na robotu prozatím ukládá a poskytuje video pouze pro Raspberry Pi. V budoucnu by bylo vhodné video streamovat prostřednictvím Wi-Fi do uživatelského rozhraní, které by jej umožňovalo zobrazovat. Také by bylo možné nad videem provádět například analýzu obrazu. To lze využít při vyhledávání překážek nebo cesty.

5.2.2 Integrace na Robotický operační systém

Robotický operační systém (ROS) [27] je univerzální nástroj, který poskytuje řadu knihoven a nástrojů pro řízení robotů. Jedno z možných rozšíření může být implementace platformy pro šestinohý robot pro ROS.

5.2.3 Rozšíření senzorického systému

Robot prozatím není schopen detekovat například hloubku schodu nebo překážku před končetinou. Nabízí se navýšení počtu senzorů, které by tyto překážky byly schopny detekovat. Na výběr jsou například taktilní senzory nebo infračervené nebo ultrazvukové dálkoměry.

Robot je schopen detekovat překážky nebo zastavit pohyb končetiny, pokud při došlápnutí na zem narazí na překážku. Je zde ale prostor pro obecnější řízení robotu jako je užití neuronových sítí při učení pohybových algoritmů nebo při zotavování při ztrátě končetiny.

5.2.4 Jednotná deska plošných spojů

Robot je vybaven řadou elektronických komponent, které zajišťují jeho fungování. Řada těchto komponent je umístěna na nepájivém poli, což není nejlepší řešení. Proto by bylo dobré navrhnout a vyrobit desku plošných spojů, která by integrovala všechny komponenty a poskytovala přípojky pro motory, enkodéry, snímače země a další senzory.

[1] Staženo z http://smart-prototyping.com/image/cache/data/2_components/Raspberry%20Pi/100940%20Raspberry%20Pi%20camera%20module%205MP%20CF5647CM-V1/1-750x750.jpg, dne 13. 5. 2015.

[2] Staženo z http://www.modelarina.cz/images/rc-soupravy/futaba-fast-t7c.jpg, dne 13. 5. 2015.

Kapitola 6

Testování robotu

6.1 Testování senzorického systému

Robot disponuje řadou senzorů, které detekují překážky, určují polohu robotu nebo měří stav akumulátoru. Každý z těchto senzorů byl podroben řadě testů, které ověřovali funkčnost, správnost a přesnost měření. Testovány byly ultrazvukové dálkoměry, nášlapné snímače, enkodéry a měřiče napětí.

6.1.1 Sonary

Při testování sonarů byla ověřována zvláště přesnost měření. Dále byly testovány vlivy materiálů v okolí na odraženou vlnu a přesnost měření. Senzor byl umístěn na hraně otevřeného prostoru, aby nedocházelo k odrazům od země, a následně byly měřeny vzdálenosti 5 cm, 15 cm, 50 cm a 100 cm. Každé měření spočívalo v umístění překážky do vyzařovacího pole senzoru a následném měřícím cyklu. Vždy bylo provedeno deset měření v rámci jedné vteřiny, výsledné hodnoty těchto deseti měření byly průměrovány a vyhodnocována byla až průměrná hodnota. Program, který obsluhuje sonar, přeskakuje hodnoty, které jsou menší nebo rovny nule, a hodnoty větší než 400 cm.

Výsledky měření se od měřené hodnoty odchylovaly maximálně o ± 1.4 cm. V jednom případě, kdy byla překážka tvořená hustým textilním závěsem, došlo pravděpodobně k úplnému pohlcení zvukových vln a měření selhalo. V ostatních případech probíhalo měření podle očekávání.

6.1.2 Nášlapné snímače

Při testování nášlapných snímačů byl robot umístěn na stojan, jeho končetiny byly ve vzduchu, a prováděl algoritmus vlny. Následně byly pod končetiny umisťovány překážky a bylo sledováno, kdy dojde k zastavení pohybu končetiny při pokládání směrem dolů. Zároveň byla hledána nejvhodnější prahová hodnota pro zastavení pohybu končetiny robotu.

Během testování pracovaly snímače země bez potíží, vždy došlo k zastavení končetiny na dané překážce. Aktuálně nastavená prahová hodnota pro zastavení končetiny je 400 z 1024. Tato hodnota je ovšem pouze výchozí, je následně měněna automatickou kalibrací.

6.1.3 Enkodéry

Při testování enkodérů bylo zjišťováno, zda určitá poloha servomotoru vždy odpovídá stejné hodnotě, kterou naměřily enkodéry. Dále bylo ověřováno, zda enkodéry vracejí různé hod-

noty po celém rozsahu pohybu servomotorů. Testování probíhalo na každém servomotoru zvlášť, kdy servomotor prováděl pohyb mezi krajními polohami a byly zaznamenávány hodnoty z enkodérů. Následně byly jednotlivé průchody mezi sebou porovnávány.

Z testů vyplývá, že enkodéry sice pokrývají celý rozsah pohybu servomotorů, ale nejsou schopny naměřit přesně stejné výsledky v daných polohách. Při měření docházelo k odchylkám ±8 dílků stupnice z celkového rozsahu 1024. Tato odchylka není fatální, a pro určení přibližné polohy servomotoru, potažmo končetiny, je akceptovatelná. Pro eliminaci odchylky je použita tolerance ±10. To stále umožňuje určovat polohu ve stovce různých pozic, což odpovídá rozlišení přibližně 1,5 °.

6.1.4 Měření napětí na akumulátoru

Pro měření hodnoty na akumulátoru je použit dělič napětí, který je připojen na A/D převodník mikrokontroléru. Testování tak mělo za úkol určit okrajové hodnoty. Byla hledána maximální hodnota, při níž je indikován stav akumulátoru jako "Plně nabito", a hodnota, při níž již není možné robot používat, protože by mohlo dojít k podvybití akumulátoru. Hodnoty byly určeny z experimentálního měření akumulátoru.

6.2 Testování softwaru robotu

Testování jednotlivých softwarových komponent probíhalo nejprve odděleně, kdy byly testovány funkcionality, které nevyžadují příchozí resp. odchozí komunikaci, následně byla komunikace simulována pomocí terminálu a nakonec komponenty komunikovaly mezi sebou a byl sledován průběh jejich komunikace a chování.

V uživatelském rozhraní byly testovány ovládací prvky a jejich obsluha, dále pak správnost hodnot získaných z uživatelského rozhraní a správnost formátu zpráv. Následně byly testovány reakce na zasílání jednotlivých zpráv protokolu a jejich propagace do uživatelského rozhraní.

Programy pro Raspberry Pi a mikrokontrolér byly testovány obdobným způsobem jako uživatelské rozhraní. Následně byly všechny komponenty spuštěny současně. Sledována byla především správnost používání protokolu a formáty příchozích a odchozích dat. Software byl také podroben zátěžovým testům, kdy bylo připojeno několik robotů a uživatelské rozhraní zobrazovalo dané údaje. Provedeny byly také dlouhodobé testy, kdy byl systém spuštěn po dobu několika hodin.

Během testování nedošlo k žádnému neočekávanému chování. Systém se jeví jako stabilní.

6.3 Testování pohybového aparátu

Jelikož robot postavený v rámci této práce je mobilní robot, jednou z hlavních částí testování je ověření schopností jeho pohybu. Testována byla opakovatelnost pohybu robotu, pohyb v členitém terénu a jednotlivé pohybové algoritmy. Z některých testů jsou k dispozici videa dostupná z [22].

Jedním z testů byl pohyb robotu po různých površích, s různými nášlapnými plochami, při využití různých pohybových algoritmů. Jako nášlapné plochy byly testovány šrouby a šrouby opatřené papírovou páskou. Testování probíhalo na dřevěné podlaze a koberci s využitím pohybových algoritmů tripod a vlna. Podrobný rozpis jednotlivých testů uvádí Tabulka 6.1.

ID	Povrch	Nášlapné plochy	Pohybový algoritmus
1	koberec	šroub	vlna
2	koberec	šroub	tripod
3	koberec	šroub s páskou	vlna
4	koberec	šroub s páskou	tripod
5	dřevěná podlaha	šroub	vlna
6	dřevěná podlaha	šroub	tripod
7	dřevěná podlaha	šroub s páskou	vlna
8	dřevěná podlaha	šroub s páskou	tripod

Tabulka 6.1: Testování pohybu robotu na různých površích s různými nášlapnými plochami.

Během pohybu robotu byl sledován průběh pohybu a stabilita systému. Monitorovány byly také nežádoucí jevy, jako je podkluzování nášlapných ploch nebo nestabilita a pády robotu. Z výsledků jednotlivých testů lze říci, že robot při všech osmi konfiguracích úspěšně prošel stanovenou trasu. Výjimkou byl testovací případ číslo 6, kdy robot na dřevěné podlaze při použití pohybového algoritmu tripod značně podjížděl a při jednom z kroků dokonce upadl. Bylo také potvrzeno, že páska na koncích šroubů snižuje prokluzy mezi nášlapnou plochou a povrchem.

6.3.1 Opakovatelnost pohybu

Mobilní robot musí být schopen nejen pohybu, ale musí být také schopen tento pohyb co nejpřesněji zopakovat. Testování probíhalo s využitím pohybového algoritmu vlna, kdy robot z vyznačené startovní pozice provedl 13 cyklů tohoto algoritmu (jeden cyklus odpovídá postupnému nastavení všech šesti končetin a následnému pohybu těla směrem vpřed), po jejichž provedení byly zaznačeny cílové pozice všech šesti končetin. Postupně bylo provedeno deset testů. Výsledné pozice končetin zobrazuje Obrázek 6.1 a Tabulka 6.2. Směrodatná odchylka jednotlivých testů činí pouze 21,72 mm v ose x a 14,45 mm v ose y.

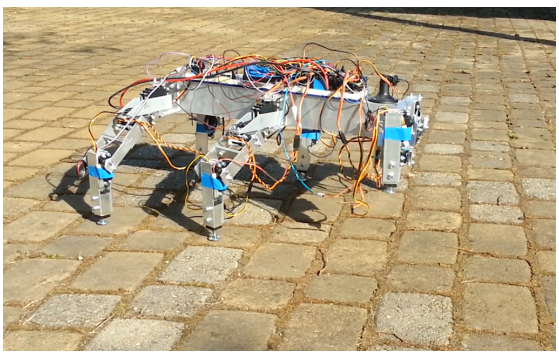
6.3.2 Pohyb v členitém terénu

Při konstrukci byl robot vybaven řadou senzorů, které mu umožňují pohyb v členitém terénu. Jsou to zejména snímače země, které detekují překážku při pokládání končetiny. Testování probíhalo v uměle vytvořeném terénu, kdy překážky byly tvořeny kameny a kusy dřeva. Robot měl za úkol projít stanovenou trasu. V průběhu testů byla sledována činnost nášlapných snímačů a jejich vliv na pohyb robotu. Obrázek 6.2 ukazuje robot během jednoho z testů. Robot průchod trasou absolvoval bez větších obtíží. Jediným problémem, který se v jednom z testu objevil, byl postupný pokles těla robotu. To je způsobeno typem řízení snímačů země a jejich vliv na pohyb končetiny při pokládání. Jako jedno z možných řešení je implementace kontroly poklesu těla a následný zdvih na všech končetinách současně, pokud to situace dovolí. Prozatím není implementováno.

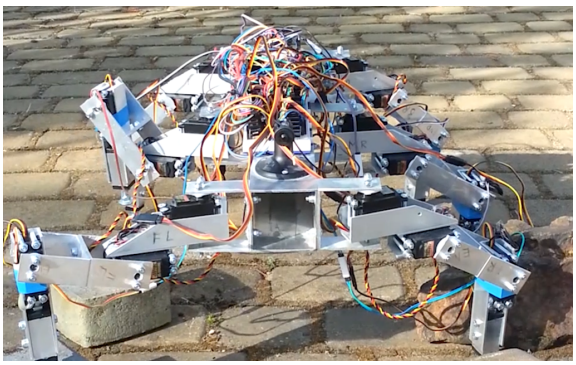
Obrázek 6.1: Výsledné pozice pravé přední končetiny v testu opakovatelnosti pohybu. Každá z nálepek odpovídá koncové pozici pravé přední končetiny při jednom z testů. Z obrázku je vidět, že vzdálenosti mezi jednotlivými nálepkami jsou minimální. Směrodatná odchylka jednotlivých testů činí pouze 21,72 mm v ose x a 14,45 mm v ose y.

ID	Přední pravá končetina [x, y]	Vzdálenost od startu [cm]
1	45, 61	81,0
2	68, 49	82,0
3	65, 79	79,0
4	89, 70	80,0
5	30, 36	83,4
6	34, 72	79,8
7	25, 60	81,0
8	29, 47	82,3
9	61, 68	80,2
10	79, 85	78,5

Tabulka 6.2: Výsledek testů opakovatelnosti pohybu pro pravou přední končetinu.



(a) Pohyb robotu ve venkovním prostředí s použití pohybového algoritmu vlna.



(b) Robot během jednoho z testů v členitém terénu překonává překážky.

Obrázek 6.2: Pohyb robotu ve venkovním prostředí a členitém terénu.

Závěr

Cílem této práce bylo zmapovat problematiku kráčivých podvozků, jejich vlastnosti, konstrukci, způsoby řízení i pohybu a srovnat je s aktuálně užívanými podvozky. Dále pak vytvořit šestinohý robot a jeho řídicí program. Robot byl také rozšířen o řadu vylepšení. Součástí projektu je uživatelské rozhraní, které umožňuje monitorovat a ovládat robot.

V rámci projektu byla nastudována teorie kráčivých podvozků se zaměřením na šestinohé roboty – hexapody. V první kapitole jsou porovnány kolové, pásové a kráčivé podvozky. Druhá kapitola se věnuje vlastnostem kráčivých podvozků, jejich rozdělením, konstrukcí a řízením. Dále jsou uvedeny nejznámější pohybové algoritmy a jejich možné grafické reprezentace, zmiňuje se také o řízení kráčivých podvozků neuronovým sítěmi. Závěrem kapitoly je uvedeno několik příkladů existujících kráčivých robotů. Třetí a čtvrtá kapitola se věnuje kráčivému robotu hexapod vlastní konstrukce, jeho úpravám a softwaru. V páté kapitole jsou popsána jednotlivá rozšíření, která byla implementována nad rámec zadaní práce. Poslední kapitola pojednává o testování robotu.

Robot, který vznikl v rámci této práce, slouží jako platforma pro testování pohybových algoritmů a jiných mechanizmů, které jsou typické pro šestinohé podvozky. Platforma je snadno rozšiřitelná a její pořizovací cena je oproti komerčním verzím poloviční. Podvozek je oproti komerčně dostupným modelům doplněn řadou senzorů, jako jsou ultrazvukové sonary, které umožňují detekci překážek v okolí, nebo nášlapné snímače, díky kterým se robot může pohybovat i v členitém terénu. Robot je schopen pohybu pomocí pohybových algoritmů tripod, vlna, vlnění a rotace.

Robot je možné řídit a monitorovat z uživatelského rozhraní, které mimo jiné vizualizuje pohyb končetin robotu, zobrazuje data ze všech senzorů a poskytuje nástroj pro tvorbu a následnou simulaci uživatelských pohybových algoritmů. Uživatel může také využít připravenou knihovnu a vytvořit vlastní ovládací rozhraní bez nutnosti implementace komunikačního protokolu robotu.

Práce byla prezentována na Studentské konferenci inovací, technologií a vědy v IT Excel@FIT 2015, kde získala 1. místo v kategorii Cena akademické obce: Technologická úroveň. Práce se také umístila ve třech dalších kategoriích – Cena průmyslu: Obchodní potenciál (4. místo), Cena veřejnosti: Výborný nápad (3. místo), Cena veřejnosti: Společenský přínos (5. místo) a získala cenu zlatého sponzora Honeywell.

Literatura

[1] Adafruit: Adafruit Ultimate GPS Breakout - 66 channel w/10 Hz updates [Version 3]. https://www.adafruit.com/products/746, 2014, [Online; navštíveno 15-05-2015].

[2] Adafruit: Round Force-Sensitive Resistor (FSR). https://www.adafruit.com/products/166, 2014, [Online; navštíveno 15-05-2015].

[3] Arduino: Arduino - ArduinoBoardMega2560. http://arduino.cc/en/Main/ArduinoBoardMega2560, 2015, [Online; navštíveno 21-03-2015].

[4] Arduino: Arduino - PulseIn. http://arduino.cc/en/Reference/pulseIn, 2015, [Online; navštíveno 21-03-2015].

[5] Arduino: Arduino - SD. http://www.arduino.cc/en/Reference/SD, 2015, [Online; navštíveno 17-05-2015].

[6] Arduino: Arduino - SPI. http://www.arduino.cc/en/Reference/SPI, 2015, [Online; navštíveno 15-05-2015].

[7] Atmel: Atmega2560. http://www.atmel.com/Images/Atmel-2549-8-bit-AVR-Microcontroller-ATmega640-1280-1281-2560-2561_datasheet.pdf, 2015, [Online; navštíveno 21-03-2015].

[8] Autodesk Inc.: 3D CAD Software — Inventor 3D CAD — Autodesk. http://www.autodesk.com/products/inventor/overview, 2015, [Online; navštíveno 13-05-2015].

[9] Beer, R. D.: *Intelligence As Adaptive Behavior: An Experiment in Computational Neuroethology*. San Diego, CA, USA: Academic Press Professional, Inc., 1990, ISBN 0-12-084730-2.

[10] Boston Dynamics: Dedicated to the Science and Art of How Things Move. http://www.bostondynamics.com/robot_cheetah.html, 2015, [Online; navštíveno 10-01-2015].

[11] Boston Dynamics: Dedicated to the Science and Art of How Things Move. http://www.bostondynamics.com/, 2015, [Online; navštíveno 10-01-2015].

[12] Boston Dynamics: Dedicated to the Science and Art of How Things Move. http://www.bostondynamics.com/robot_rhex.html, 2015, [Online; navštíveno 10-01-2015].

[13] Boston Dynamics: Dedicated to the Science and Art of How Things Move. http://www.bostondynamics.com/robot_bigdog.html, 2015, [Online; navštíveno 10-01-2015].

[14] Boston Dynamics: Dedicated to the Science and Art of How Things Move. http://www.bostondynamics.com/robot_ls3.html, 2015, [Online; navštíveno 10-01-2015].

[15] Cytron Technologies Sdn. Bhd.: User's Manual V1.0. https://docs.google.com/document/d/1Y-yZnNhMYy7rwhAgyL_pfa39RsB-x2qR4vP8saG73rE/edit, 2014, [Online; navštíveno 13-05-2015].

[16] Ding, X.; Wang, Z.; Rovetta, A.; aj.: Locomotion analysis of hexapod robot. *Climbing and Walking Robots*, 2010: s. 291–310.

[17] Habib, M.: *Bioinspiration and robotics: walking and climbing robots*. Wien: I-Tech Education and Publ, 2007, ISBN 978-3-902613-15-8.

[18] HITEC RCD USA, Inc.: HS-5485HB Standard Karbonite Digital Sport Servo. HS-5485HBStandardKarboniteDigitalSportServo|HITECRCDUSA, 2014, [Online; navštíveno 15-05-2015].

[19] HITEC RCD USA, Inc.: HS-5645MG High Torque, Metal Gear Digital Sport Servo. http://hitecrcd.com/products/servos/sport-servos/digital-sport-servos/hs-5645mg-high-torque-metal-gear-servo/product, 2014, [Online; navštíveno 15-05-2015].

[20] Jacobs, S., Sr. Extension Associate: American cockroach — Entomology — Penn State University. http://ento.psu.edu/extension/factsheets/american-cockroaches, 2013, [Online; navštíveno 12-01-2015].

[21] Žák, M.: *Návrh a konstrukce šestinohého mobilního robotu*. bakalářská práce, FIT VUT v Brně, 2013.

[22] Žák, M.: Hexapod Robot Project. http://hexapod.marekzak.cz/, 2015, [Online; navštíveno 18-05-2015].

[23] Lynxmotion: Lynxmotion Robot Kits. http://www.lynxmotion.com/, 2015, [Online; navštíveno 10-01-2015].

[24] Manoiu-Olaru, S.; Nitulescu, M.; Stoian, V.: Hexapod robot. Mathematical support for modeling and control. In *System Theory, Control, and Computing (ICSTCC), 2011 15th International Conference on*, Oct 2011, s. 1–6.

[25] NASA: All-Terrain Hex-Limbed Extra-Terrestrial Explorer. http://athlete.jpl.nasa.gov/, 2009, [Online; navštíveno 09-01-2015].

[26] NETLab Toolkit Group: VarSpeedServo. https://github.com/netlabtoolkit/VarSpeedServo, 2013, [Online; navštíveno 21-03-2015].

[27] Open Source Robotics Foundation: ROS.org — Powering the world's robots. http://www.ros.org/, 2015, [Online; navštíveno 13-05-2015].

[28] Parker, G.; Lee, Z.: Evolving neural networks for hexapod leg controllers. In *Intelligent Robots and Systems, 2003. (IROS 2003). Proceedings. 2003 IEEE/RSJ International Conference on*, ročník 2, Oct 2003, s. 1376–1381 vol.2.

[29] Peringer, P.: SNT – Simulační nástroje a techniky. https://www.fit.vutbr.cz/study/courses/SNT/public/Prednasky/SNT.pdf, 2015, [Online; navštíveno 15-05-2015].

[30] Raspberry Pi Foundation: Raspberry Pi. http://www.raspberrypi.org/, 2014, [Online; navštíveno 09-01-2015].

[31] Saranlı, U.: *Dynamic locomotion with a hexapod robot*. Dizertační práce, The University of Michigan, 2002.

[32] Siegwart, R.: *Introduction to autonomous mobile robots*. Cambridge, Mass: MIT Press, 2004, ISBN 0-262-19502-x.

[33] Tedeschi, F.; Carbone, G.: Design Issues for Hexapod Walking Robots. *Robotics*, ročník 3, č. 2, 2014: s. 181–206.

[34] Trossen Robotics: Trossen Robotics. http://www.trossenrobotics.com/, 2015, [Online; navštíveno 10-01-2015].

[35] Vallidis, N.: A Hexapod Robot and Novel Training Approach for Artificial Neural Networks. 2008.

[36] Wikipedia: Cheetah — Wikipedia, The Free Encyclopedia. http://en.wikipedia.org/w/index.php?title=Cheetah&oldid=640731497, 2015, [Online; navštíveno 08-01-2015].

[37] Wikipedia: Tiger beetle — Wikipedia, The Free Encyclopedia. http://en.wikipedia.org/w/index.php?title=Tiger%20beetle&oldid=641449375, 2015, [Online; navštíveno 09-01-2015].

[38] Zheng, Y.-F.: *Recent trends in mobile robots*. Singapore River Edge, N.J: World Scientific, 1993, ISBN 981-02-1511-8.

Seznam obrázků

Seznam tabulek

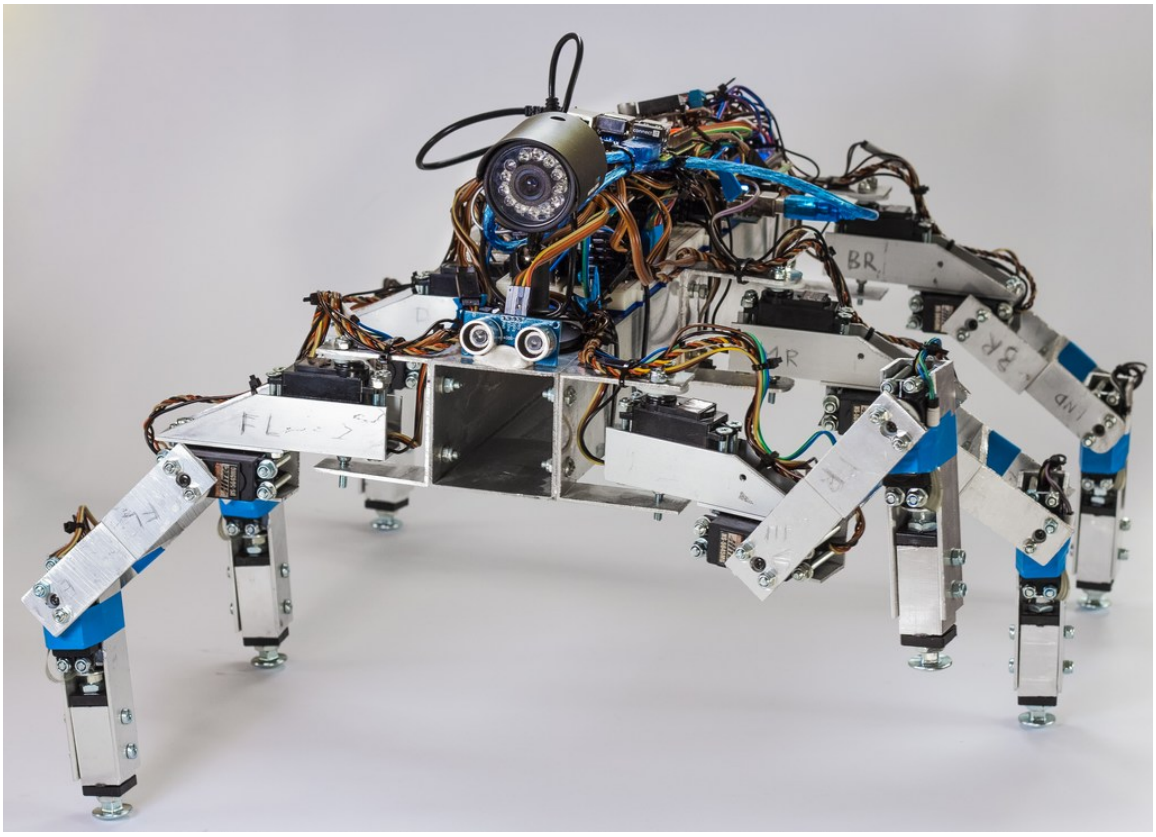
Seznam kódů

Seznam rovnic

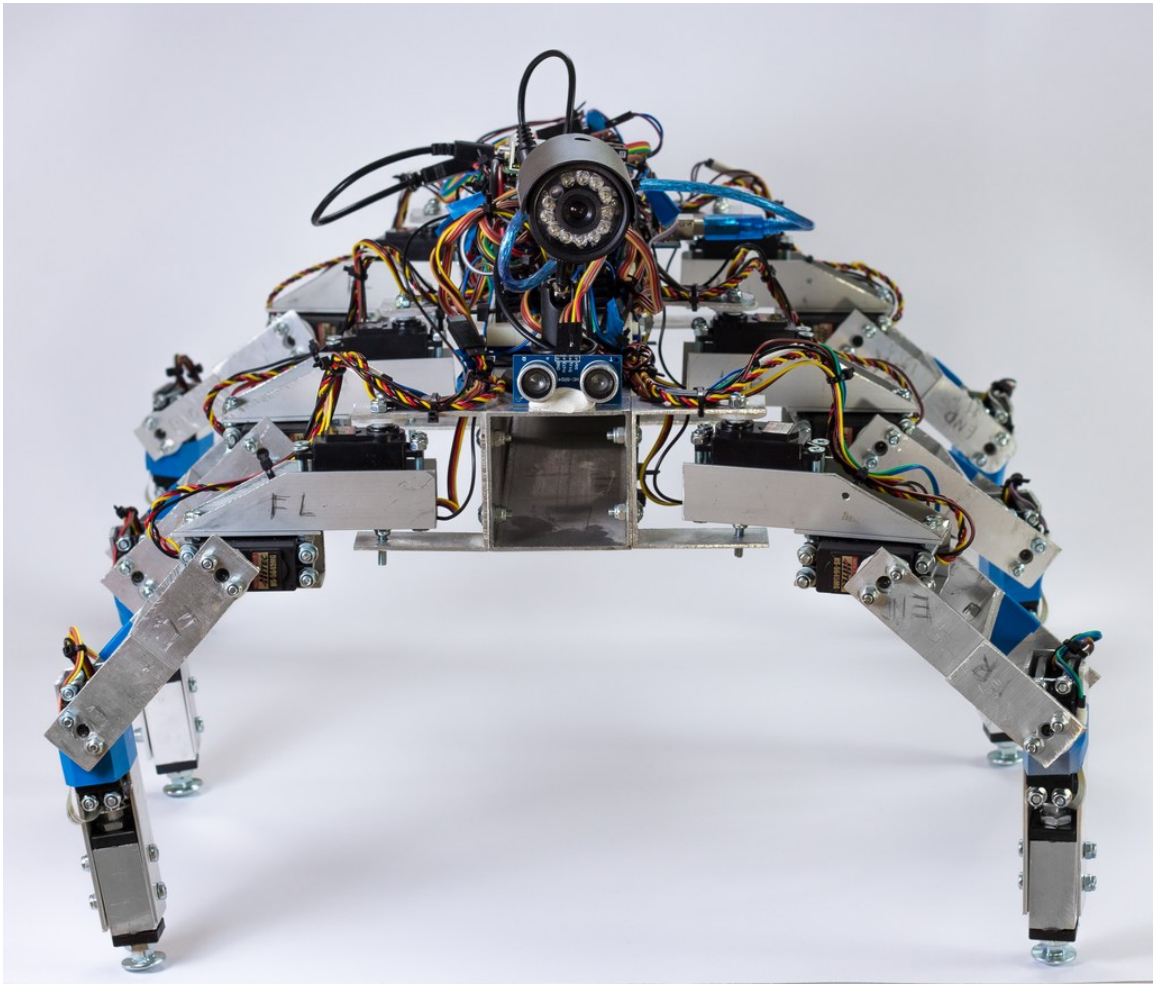
Seznam příloh

Příloha A

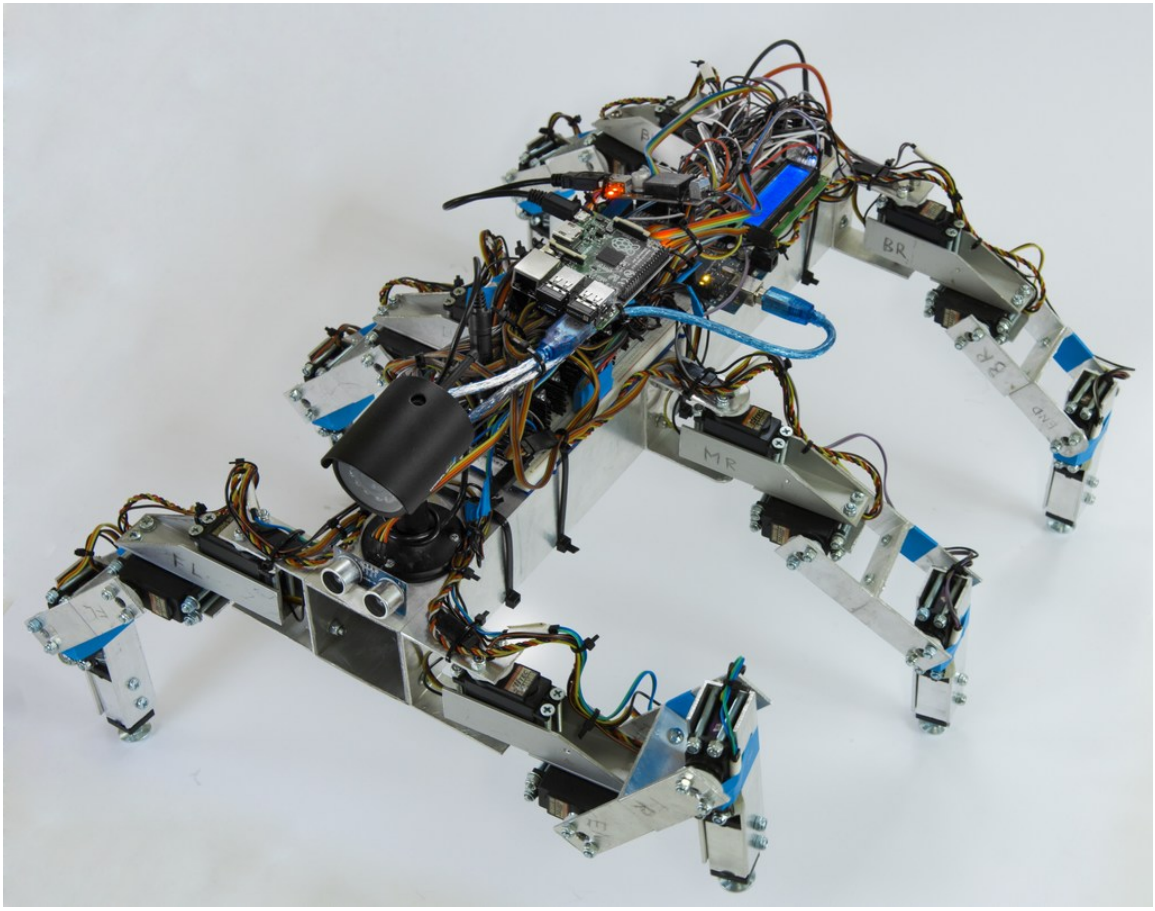
Hexapod vlastní konstrukce



Obrázek A.1: Robot hexapod vlastní konstrukce, pohled z boku.



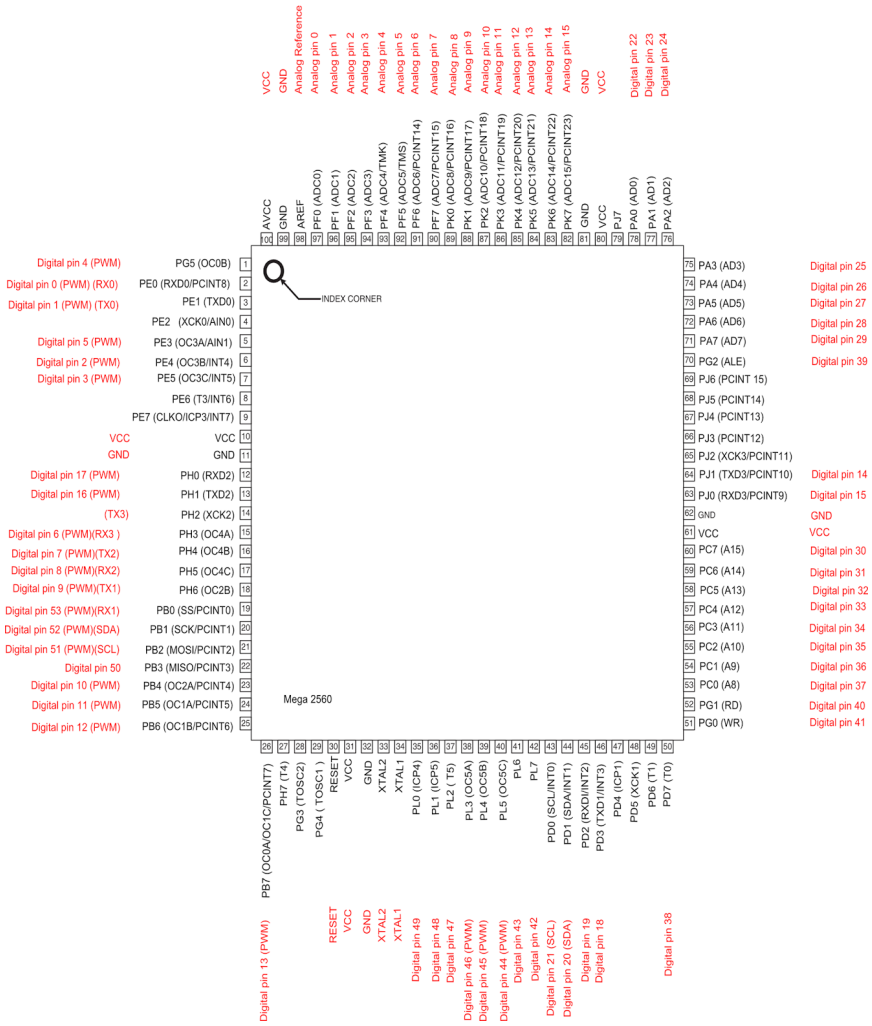
Obrázek A.2: Robot hexapod vlastní konstrukce, pohled zepředu.



Obrázek A.3: Robot hexapod vlastní konstrukce, pohled shora.

Příloha B

Rozložení vývodů mikroprocesoru Atmega2560



Obrázek B.1: Rozložení vývodů mikroprocesoru Atmega2560 a jejich namapování na vývojovou desku Arduino Mega 2560.[1]

Příloha C

Obsah CD

Media

Složka obsahuje mediální soubory k práci.

Videa

Složka obsahuje videa z testování robotu.

CAD

Složka obsahuje zdrojové soubory 3D modelu robotu pro Autodesk Inventor.

Obrazky

Složka obsahuje veškeré obrazové materiály k práci včetně zdrojových SVG souborů. Složka dále obsahuje fotodokumentaci robotu a jeho dílů.

Prace

Složka obsahuje práci ve formátu PDF.

Latex

Složka obsahuje zdrojové soubory latexu včetně obrázků.

Software

Složka obsahuje veškerý software, který je nezbytný pro provoz robotu.

Hexapod Control Room - PC

Složka obsahuje spustitelnou verzi programu 'Hexapod Control Room', který slouží k řízení a monitorování robotu. K dispozici jsou také zdrojové kódy aplikace.

[1] Staženo z http://1.bp.blogspot.com/-d_OkcysZNSU/UkxOd7QFsQI/AAAAAAAAElY/yX1rVFIIuzQ/s1600/PinMap2560.png, dne 8. 5. 2015.

Hexapod Robot - Raspberry Pi

Složka obsahuje spustitelnou verzi programu 'Hexapod Robot', který je určen pro desku Raspberry Pi umístěnou na robotu. K dispozici jsou také zdrojové kódy aplikace.

MCU program - Atmega2560

Složka obsahuje zdrojové kódy programu, který je určen pro mikrokontrolér Atmega2560 na desce Arduino Mega 2560. Program slouží k řízení servomotorů a senzorů a zasílá data na sériovou linku. Je zde umístěna také nejnovější verze vývojového prostředí Arduino IDE, pomocí kterého je možné program nahrát do mikrokontroléru.

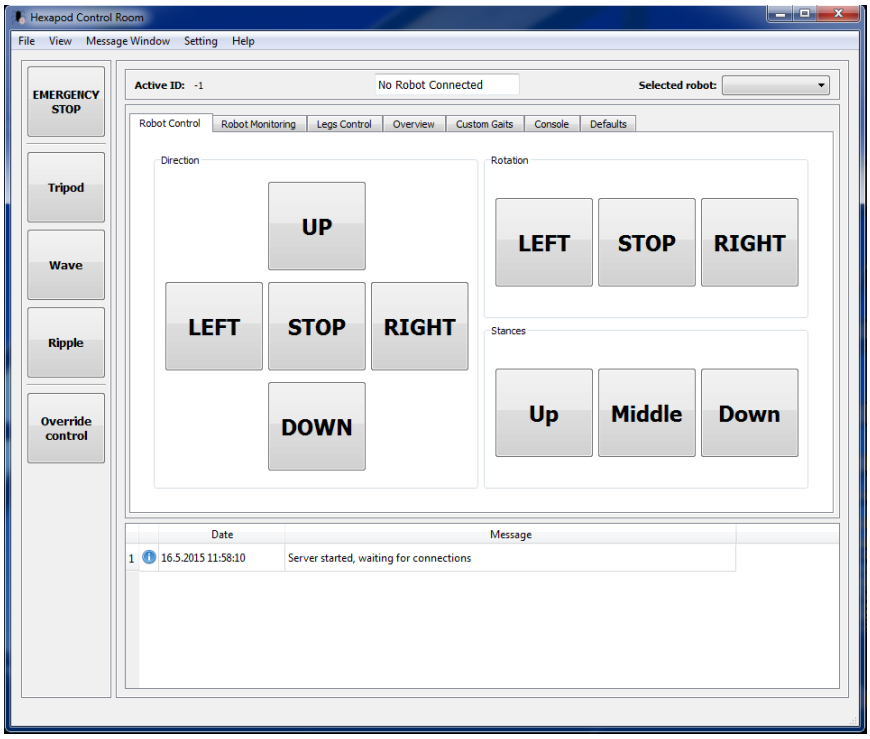
Příloha D

Plakát

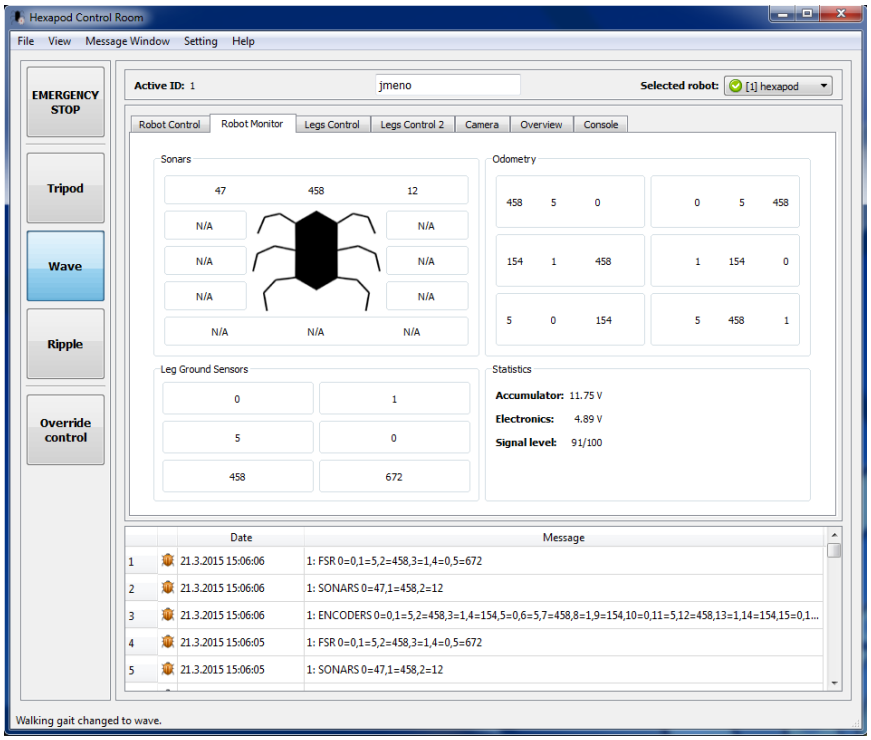
Obrázek D.1: Plakát.

Příloha E

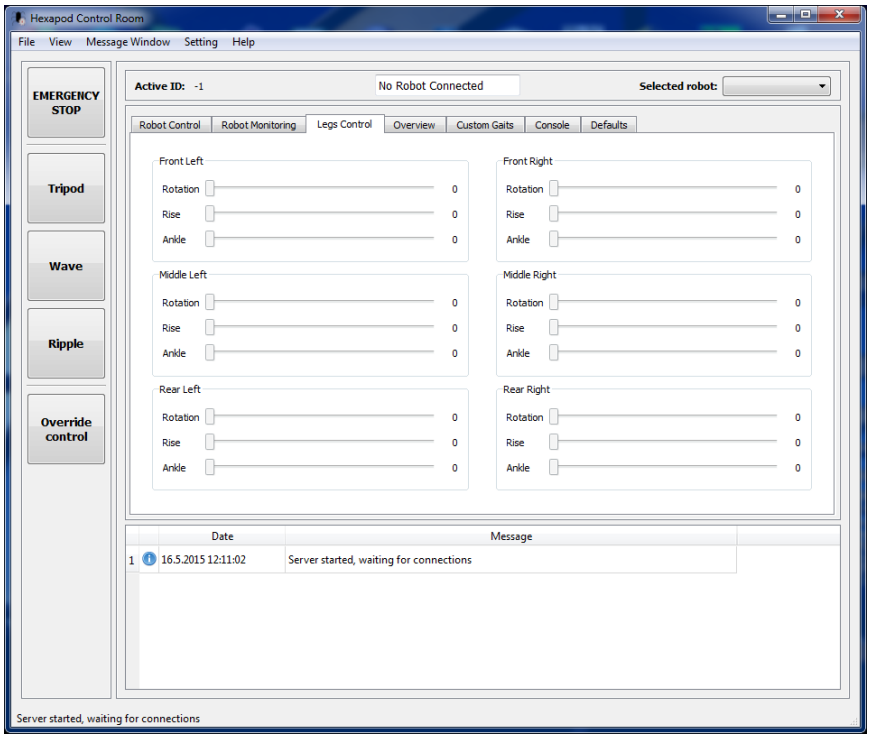
Záložky uživatelského rozhraní



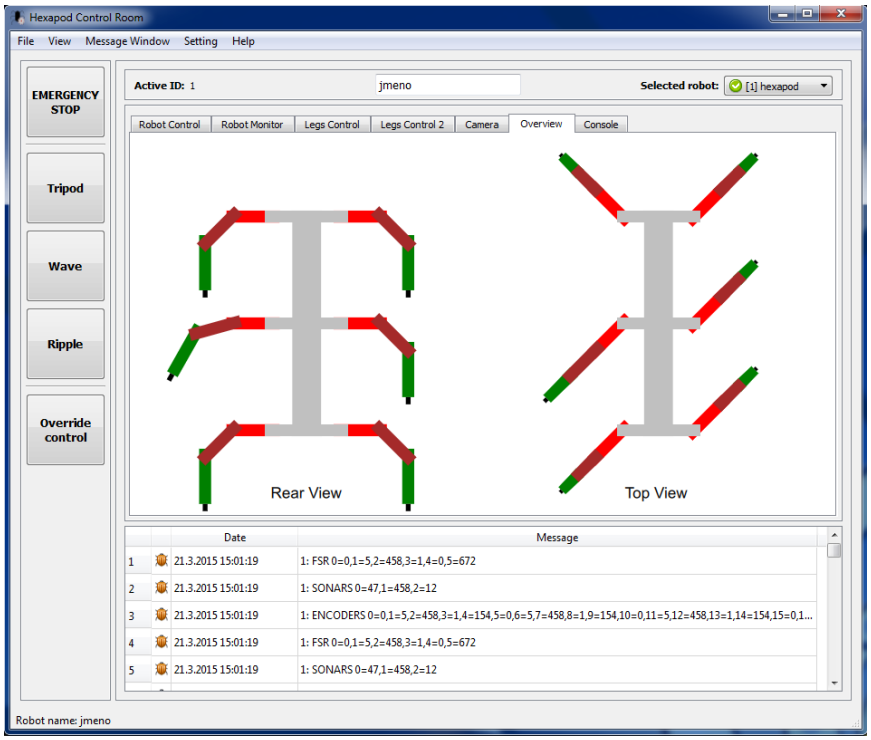
Obrázek E.1: Záložka řízení pohybu robotu.



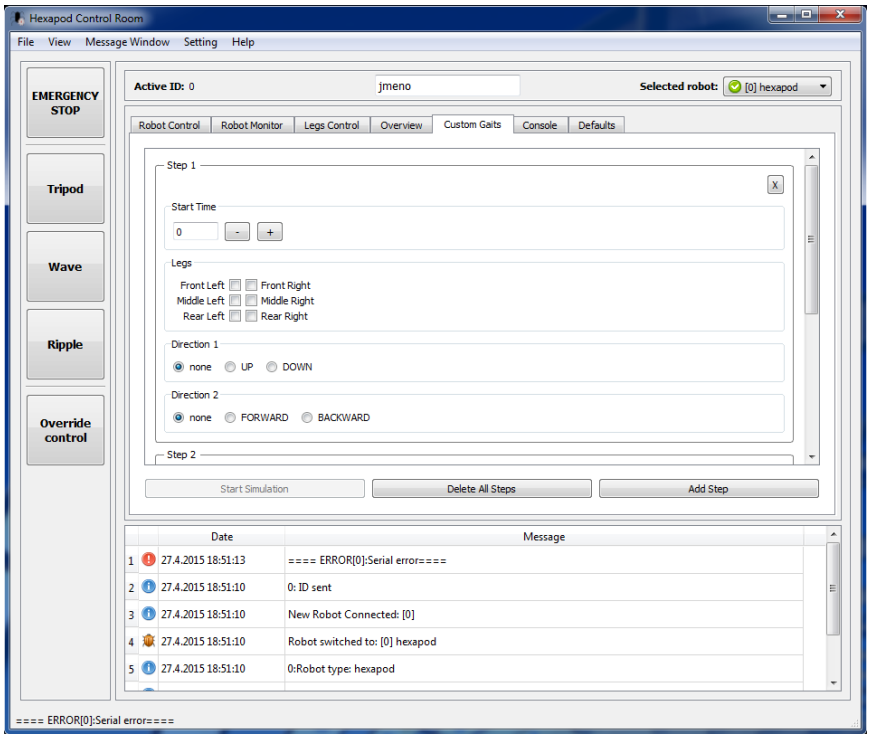
Obrázek E.2: Monitorování dat ze senzorů robotu.



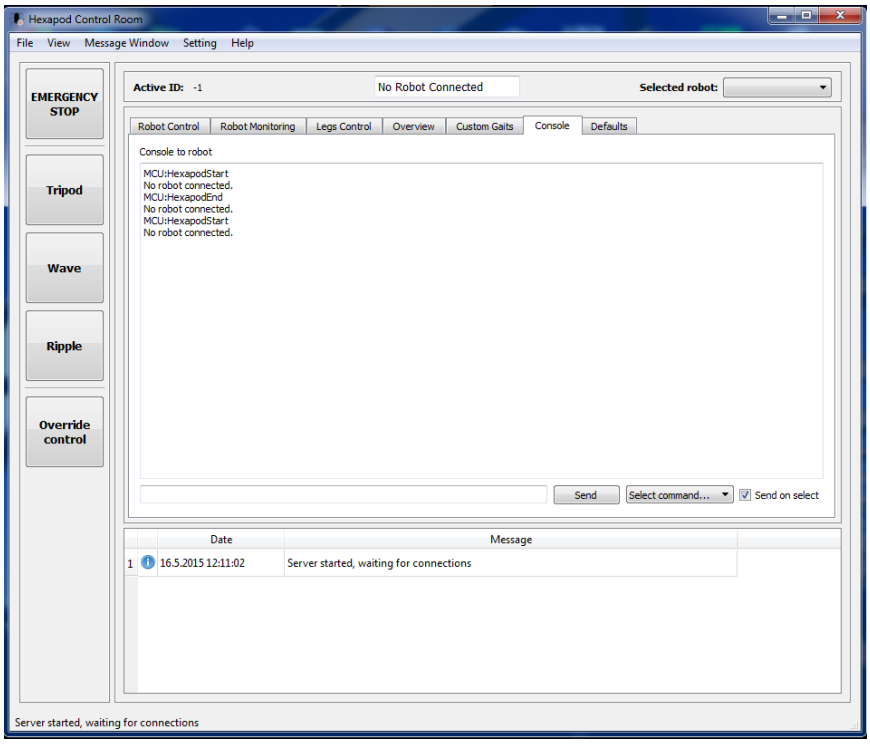
Obrázek E.3: Záložka ovládání jednotlivých servomotorů robotu.



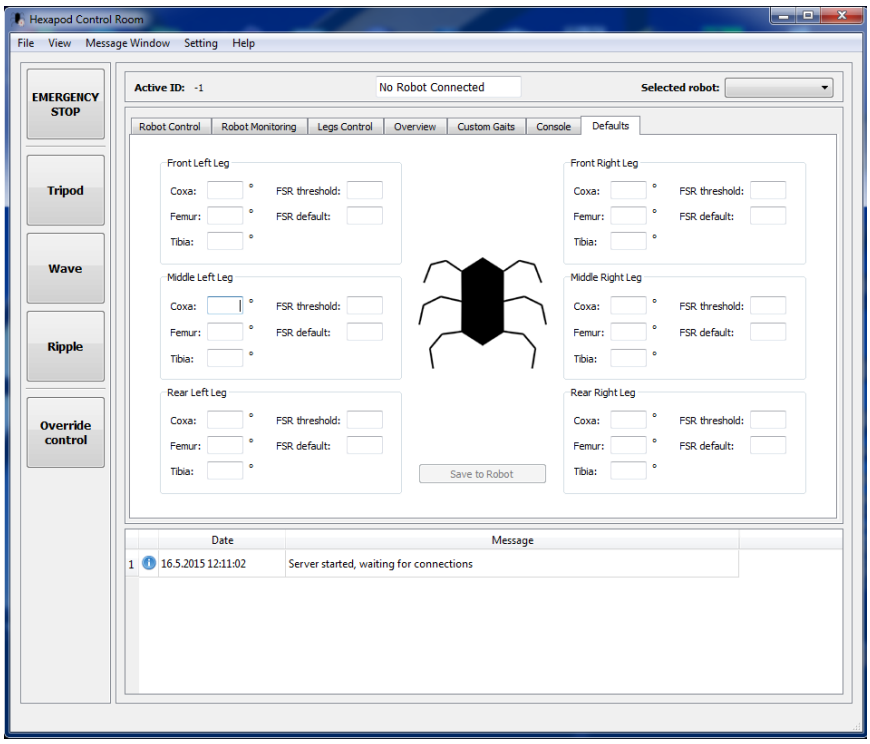
Obrázek E.4: Záložka vizualizace pozic jednotlivých končetin robotu.



Obrázek E.5: Záložka pro tvorbu uživatelských pohybových algoritmů.



Obrázek E.6: Záložka konzole umožňuje zasílání příkazů do robotu.



Obrázek E.7: Záložka výchozích hodnot pro jednotlivé servomotory robotu.