Modelování a řízení 2-kolového nestabilního transportéru Inteco

Bc. Zdeněk Charous

Diplomová práce
2021

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky
Ústav automatizace a řídicí techniky

Akademický rok: 2020/2021

ZADÁNÍ DIPLOMOVÉ PRÁCE

(projektu, uměleckého díla, uměleckého výkonu)

Jméno a příjmení:	**Bc. Zdeněk Charous**
Osobní číslo:	**A19383**
Studijní program:	**N3902 Inženýrská informatika**
Studijní obor:	**Automatické řízení a informatika**
Forma studia:	**Kombinovaná**
Téma práce:	**Modelování a řízení 2-kolového nestabilního transportéru Inteco**
Téma práce anglicky:	**Modelling and Control of the Two-Wheeled Unstable Transporter Inteco**

Zásady pro vypracování

1. Vypracujte literární rešerši na dané téma.
2. Důkladně se seznamte s výukovým modelem 2-kolového nestabilního transportéru od firmy Inteco a proveďte jeho zprovoznění.
3. Navrhněte zjednodušený matematický model tohoto zařízení a implementujte ho do prostředí MATLAB/Simulink.
4. Navrhněte vhodné způsoby řízení tohoto systému a porovnejte odezvy simulačního modelu s reálnými experimenty.
5. K danému transportéru vytvořte několik typových úloh pro studenty kurzu Řízení reálných procesů a také pro propagační účely fakulty.

Forma zpracování diplomové práce: **Tištěná/elektronická**

Seznam doporučené literatury:

1. *Two-Wheeled Unstable Transporter User's Manual*. Krakow: Inteco Ltd, 2017.
2. KNAPIK, D., K. KOLEK, M. ROSOL a A. TURNAU. Autonomous, reconfigurable mobile vehicle with rapid control prototyping functionality. *IFAC-PapersOnLine*. 2019, 52(8), 13-18.
3. BOBÁL, V. *Identifikace systémů*. Zlín: UTB ve Zlíně, Fakulta aplikované informatiky, 2009.
4. DOSTÁL, P.; MATUŠŮ, R. *Stavová a algebraická teorie řízení*. Zlín: UTB ve Zlíně, Fakulta aplikované informatiky, 2010.
5. BALÁTĚ, J. *Automatické řízení*. Praha: BEN – technická literatura, 2003.
6. WELLSTEAD, P. *Introduction to Physical Modelling*. London: Academic Press Ltd, 2000.
7. SKOGESTAD, S.; POSTLETHWAITE, I. *Multivariable Feedback Control: Analysis and Design*. Chichester: Wiley, 2005.

Vedoucí diplomové práce: **doc. Ing. František Gazdoš, Ph.D.**
Ústav řízení procesů

Datum zadání diplomové práce: **15. ledna 2021**
Termín odevzdání diplomové práce: **17. května 2021**

doc. Mgr. Milan Adámek, Ph.D. v.r.
děkan

UNIVERZITA TOMÁŠE BATI VE ZLÍNĚ
Fakulta aplikované informatiky
DĚKAN

prof. Ing. Vladimír Vašek, CSc. v.r.
ředitel ústavu

Ve Zlíně dne 15. ledna 2021

Jméno, příjmení: Zdeněk Charous
Název bakalářské/diplomové práce: Modelování a řízení 2-kolového nestabilního transportéru Inteco
Prohlašuji, že

- beru na vědomí, že odevzdáním diplomové/bakalářské práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb. o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších právních předpisů, bez ohledu na výsledek obhajoby;
- beru na vědomí, že diplomová/bakalářská práce bude uložena v elektronické podobě v univerzitním informačním systému dostupná k prezenčnímu nahlédnutí, že jeden výtisk diplomové/bakalářské práce bude uložen v příruční knihovně Fakulty aplikované informatiky Univerzity Tomáše Bati ve Zlíně a jeden výtisk bude uložen u vedoucího práce;
- byl/a jsem seznámen/a s tím, že na moji diplomovou/bakalářskou práci se plně vztahuje zákon č. 121/2000 Sb. o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších právních předpisů, zejm. § 35 odst. 3;
- beru na vědomí, že podle § 60 odst. 1 autorského zákona má UTB ve Zlíně právo na uzavření licenční smlouvy o užití školního díla v rozsahu § 12 odst. 4 autorského zákona;
- beru na vědomí, že podle § 60 odst. 2 a 3 autorského zákona mohu užít své dílo – diplomovou/bakalářskou práci nebo poskytnout licenci k jejímu využití jen připouští-li tak licenční smlouva uzavřená mezi mnou a Univerzitou Tomáše Bati ve Zlíně s tím, že vyrovnání případného přiměřeného příspěvku na úhradu nákladů, které byly Univerzitou Tomáše Bati ve Zlíně na vytvoření díla vynaloženy (až do jejich skutečné výše) bude rovněž předmětem této licenční smlouvy;
- beru na vědomí, že pokud bylo k vypracování diplomové/bakalářské práce využito softwaru poskytnutého Univerzitou Tomáše Bati ve Zlíně nebo jinými subjekty pouze ke studijním a výzkumným účelům (tedy pouze k nekomerčnímu využití), nelze výsledky diplomové/bakalářské práce využít ke komerčním účelům;
- beru na vědomí, že pokud je výstupem diplomové/bakalářské práce jakýkoliv softwarový produkt, považují se za součást práce rovněž i zdrojové kódy, popř. soubory, ze kterých se projekt skládá. Neodevzdání této součásti může být důvodem k neobhájení práce.

Prohlašuji,

- že jsem na diplomové/bakalářské práci pracoval samostatně a použitou literaturu jsem citoval. V případě publikace výsledků budu uveden jako spoluautor.
- že odevzdaná verze diplomové práce a verze elektronická nahraná do IS/STAG jsou totožné.

Ve Zlíně, dne 4.5.2021

Zdeněk Charous v. r.
podpis diplomanta

ABSTRAKT

Práce se zabývá dvoukolovými nestabilními systémy, které jsou založené na principu inverzního kyvadla. Obsahuje obecný popis těchto systémů a shrnuje důležité literární zdroje zabývající se danou problematikou. Dále se zabývá vytvořením matematického modelu konkrétního robota Inteco a jeho implementací do prostředí Matlab-Simulink. Navržené způsoby řízení jsou nejprve simulačně ověřeny a potom aplikovány na reálný systém. Konkrétně se jedná o PID, LQ a MPC regulátory. Závěr práce je věnován zhodnocení jednotlivých regulátorů na základě dosažených výsledků při řešení několika typových úloh souvisejících se stabilizací robota a sledováním definované trajektorie.

Klíčová slova: dvoukolový robot, nestabilní systém, inverzní kyvadlo, matematický model, LQ řízení, MPC řízení, Inteco, Matlab, Simulink, nepřímý návrh regulátoru

ABSTRACT

This diploma thesis deals with two–wheeled unstable models, which are based on the principle of inverse pendulum. It contains a general description of these systems and summarizes important literary sources dealing with this issue. It also deals with the development of a mathematical model for the specific Inteco robot and its implementation in the Matlab-Simulink environment. The proposed control methods are first verified by simulation means and then applied to the real Inteco system. More specifically, PID, LQ and MPC controllers were tested. The conclusion of the work is devoted to the evaluation of individual controllers based on the results achieved in solving several defined tasks related to stabilization and trajectory tracking.

Keywords: two–wheeled robot, unstable system, inverse pendulum, mathematical model, LQ control, MPC control, Inteco, Matlab, Simulink, Model-Based Controller Design

Tímto bych chtěl poděkovat panu doc. Ing. Františkovi Gazdošovi, Ph.D. za vedení mé diplomové práce, cenné rady a připomínky. Dále pak panu doc. Ing. Markovi Kubalčíkovi, Ph.D. za rady a čas věnovaný konzultacím.

Prohlašuji, že odevzdaná verze bakalářské/diplomové práce a verze elektronická nahraná do IS/STAG jsou totožné.

OBSAH

ÚVOD

Tématem této diplomové práce je modelování a řízení dvoukolového nestabilního robota vyrobeného firmou Inteco. Jedná se o výukový model určený k praktickému ověření znalostí získaných při teoretickém studiu automatizace se zaměřením na nestabilní systémy z oblasti mechatroniky.

Teoretická část této práce zahrnuje literární rešerši, která shrnuje problematiku systémů založených na principu inverzního kyvadla v průřezu celou historií – od prvních jednoduchých experimentálních modelů až po moderní autonomní systémy. Dále teoretická část popisuje jednotlivé konstrukční části dvoukolových robotů a specifické požadavky na jejich parametry. Navazuje popis konkrétního systému Inteco, včetně detailního popisu zprovoznění a práce s robotem. Dále se věnuji matematickému popisu systému s cílem vytvořit a implementovat matematický model použitelný pro simulace při návrhu zákona řízení, a to jednak v nelineární verzi a také linearizované verzi.

V praktické části je popsána implementace matematického modelu do prostředí Matlab Simulink a ověření platnosti matematického modelu pomocí experimentů na reálném systému. Dále zde navrhuji několik možností řízení robota, konkrétně PID, LQ, a MPC. Pro návrh používám převážně nástroje systému Matlab.

Závěrem se věnuji porovnání navrhnutých regulačních struktur, a to provedením několika experimentů, které lze použít jako typové úlohy pro budoucí studenty a uživatele robota Inteco.

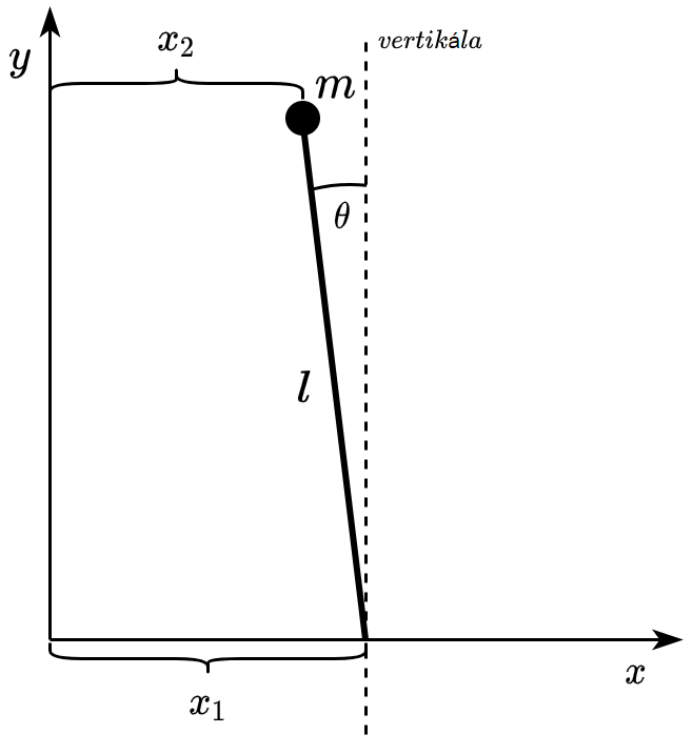
I. TEORETICKÁ ČÁST

1 LITERÁRNÍ REŠERŠE A OBECNÝ POPIS DVOUKOLOVÉHO ROBOTA

Následující kapitola by měla sloužit jako úvod do problematiky dvoukolových robotů. První část kapitoly je věnována literární rešerši. Zde jsou shrnuty jak poznatky z oblasti matematického modelování a možnosti řízení dvoukolových robotů, tak i zajímavé projekty z této oblasti. Následuje obecný popis dvoukolového robota. Zde jsou podrobně popsány jednotlivé části robota a jejich význam.

1.1 Literární rešerše

Jako první bych rád zmínil práci „The Mechanical Seal“, jejímž autorem je James Kerr Roberge. Práce byla vydána roku 1960 a jako vůbec první popisuje návrh a realizaci experimentálního modelu inverzního kyvadla a jeho stabilizaci pomocí zpětnovazebního řízení.



Obrázek 1 Inverzní kyvadlo

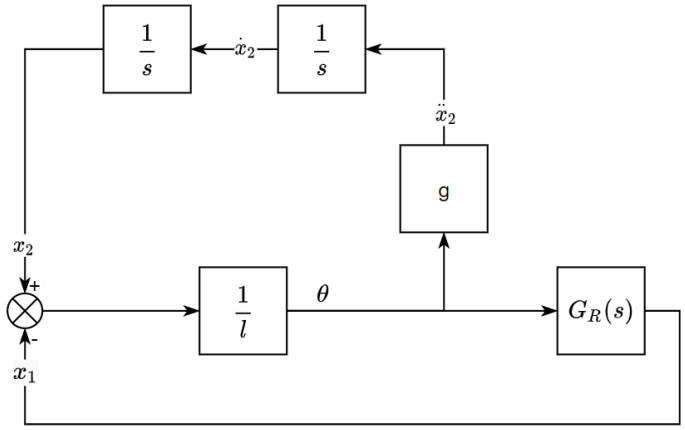
Autor používá jednoduchý matematický model inverzního kyvadla linearizovaný pro malé úhly vychýlení z vertikální polohy vycházející z rovnice (1).

$$sin\theta \approx \theta = \frac{x_2 - x_1}{l} \tag{1}$$

Za předpokladu ideálně tuhého nehmotného ramene o délce *l* potom popisuje sílu působící na hmotu *m* rovnicí (2).

$$F_m = m\ddot{x}_2 = mg\ sin\theta - B\dot{x}_2 \tag{2}$$

Autor se snaží dle blokového schématu (obrázek 2), při dočasném zanedbání tlumení (koeficient tlumení *B* považujeme za nulový) nalézt takovou přenosovou funkci regulátoru $G_r(s)$ aby dosáhl stabilizace systému, což ověřuje použitím Routhova kritéria stability na přenosovou funkci uzavřeného regulačního obvodu.

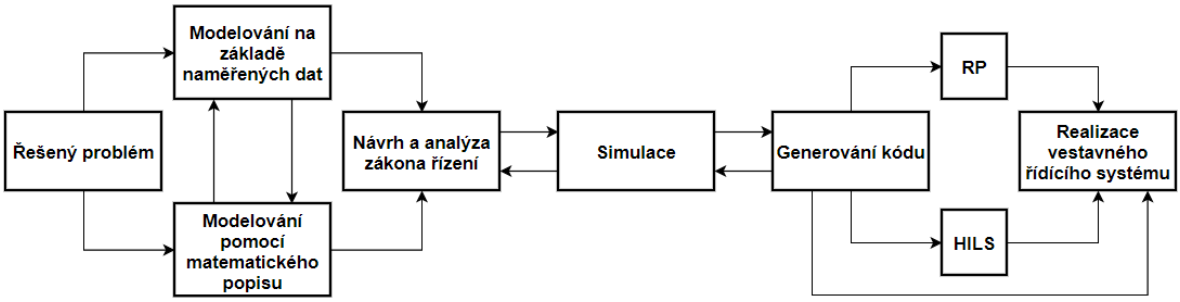


Obrázek 2 Blokové schéma regulátoru

V práci je detailně popsána realizace celého zařízení, ale především zpětnovazebního regulátoru, tvořeného převážně diskrétními součástkami. Potřeba dvou integrátorů je vyřešena pomocí dvojice tachodynam. Nastavení proměnných parametrů je prováděno potenciometry. Výslednou stabilitu celého zařízení autor posuzuje dle Nyquistova kritéria stability a Bodeho diagramů. Při testování zařízení dokáže inverzní kyvadlo stabilizovat a pokud na kyvadlo nepůsobí vnější poruchy osciluje kolem žádané hodnoty s frekvencí 1 rad/s. Dále autor uvádí, že při správném nastavení parametru zesílení zpětné vazby je rozkmit stabilizační platformy možno udržet pod hranicí 15 centimetrů. Linearizace systému je platná pouze pro malé úhly vychýlení a systém je tedy schopen eliminovat poruchu maximálně 3° až 4° a stabilizace systému při takové poruše trvá přibližně 30 sekund.

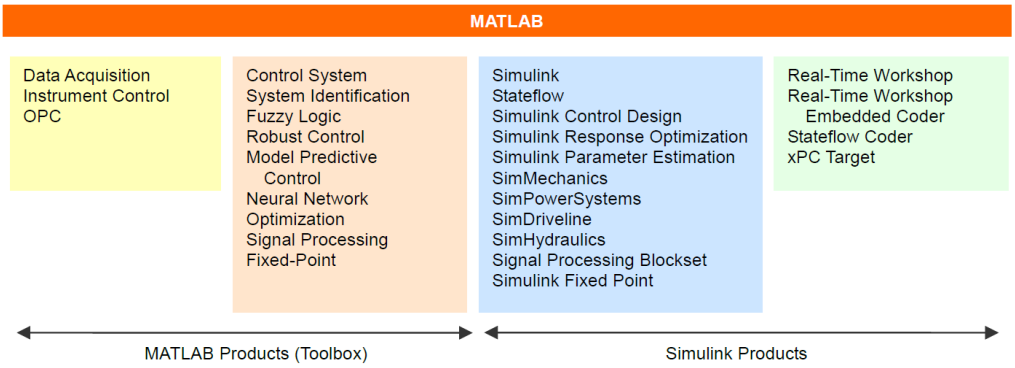
Dosažené výsledky jsou v porovnání z dnešními systémy spíše podprůměrné. Avšak s přihlédnutím k omezeným technickým zdrojům tehdejší doby se jednalo o průlomovou práci nejen po teoretické, ale i po praktické stránce. Autor touto prací položil základní kámen pro další výzkum a vývoj v oblasti regulace nestabilních systémů, založených na principu inverzního kyvadla, ve kterém sám dále pokračoval. [1]

Moderní metody návrhu zákona řízení obvykle vyžadují kvalitní matematický model řízeného procesu. Práce od autora Yorihisa Yamamota [2] se zabývá návrhem zákona řízení dvoukolového robota NXTway-GS technikou MBD (Model Based Design). Jedná se o techniku, kdy je k problematice návrhu zákona řízení přistupováno dle postupu, jehož vývojový diagram vidíme na obrázku 3.



Obrázek 3 Vývojový diagram MBD

Je zřejmé, že stěžejní částí tohoto postupu je vytvoření modelu řešeného problému. Další postup je implementace modelu do prostředí umožňující návrh, analýzu a simulaci zákona řízení. K těmto účelům autor používá systém Matlab, což je dnes jeden z nejkomplexnějších systémů pro tyto účely [15].



Obrázek 4 Softwarové moduly Matlab MBD [2]

V práci je dále uveden postup vytvoření matematického modelu systému pomocí Lagrangeovské formulace mechaniky. Autor používá k popisu Lagrangeův tvar Newtonových pohybových rovnic a zobecněné souřadnice.

Lagrangeovu formu pohybové rovnice můžeme obecně zapsat jako

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_r}\right) - \frac{\partial L}{\partial q_r} = F_r \tag{3}$$

kde

L je Lagrangeova funkce (tzv. kinetický potenciál)

q_r jsou zobecněné souřadnice systému pro $r = 1{,}2 \cdots n$ stupňů volnosti systému

F_r je zobecněná síla spojená se zobecněnou souřadnicí q_r

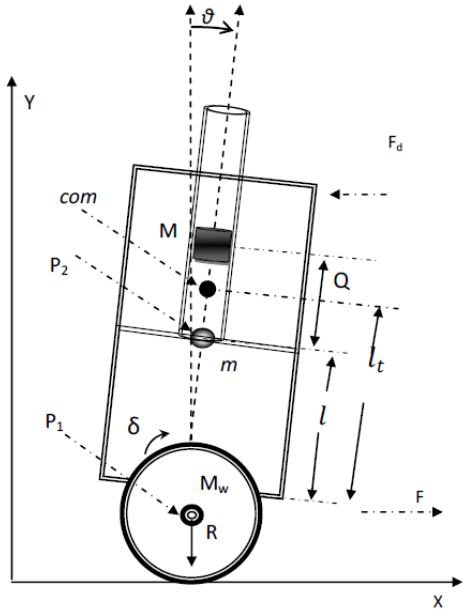
Takto lze získat n diferenciálních rovnic druhého řádu, kde zobecněné souřadnice můžeme chápat jako stavové veličiny (závislé proměnné) a čas t jako nezávislou proměnnou. Zobecněné souřadnice jsou v tomto případě měřitelné parametry určující stav systému [4]. Tento přístup k vytvoření matematického modelu uvádějí i další zdroje [3][6][7][8].

Při vytváření matematického modelu lze k přesnému určení parametrů závislých na mechanické konfiguraci robota použít CAD software. Tento přístup je založen na přesném vymodelování jednotlivých mechanických částí a specifikaci materiálu, resp. hustoty materiálu ze kterého jsou vyrobeny. CAD software je potom schopný určit hmotnosti a setrvačné síly jednotlivých částí i funkčních celků nebo vzdálenost těžiště od osy otáčení kol. Tento přístup byl úspěšně aplikován při návrhu řízení robota „Joe“ vyvinutého ve Švýcarském federálním technologickém institutu v Lausanne [5].

Čím přesněji jsou výše uvedené parametry určeny, tím přesnější je i výsledný matematický model systému. Zejména určení polohy těžiště je u nehomogenní struktury, jakou je skelet robota osazený jednotlivými komponentami, problematické. Jedná se však o stěžejní parametr významně ovlivňující výsledné dynamické chování robota.

Tohoto faktu využil Omar Khalil při návrhu dvoukolového robota s proměnnou polohou těžiště. Do standardní koncepce bylo přidáno závaží, jehož poloha je řízená pomocí aktuátoru viz obrázek 5. Počet diferenciálních rovnic popisující systém se nezmění, protože pohybující se závaží nelze považovat za další stupeň volnosti systému v globální

soustavě souřadnic. Jedná se pouze o řiditelný parametr, kterým lze měnit dynamiku systému s cílem zlepšení výkonu navrhovaného řídicího algoritmu [6].



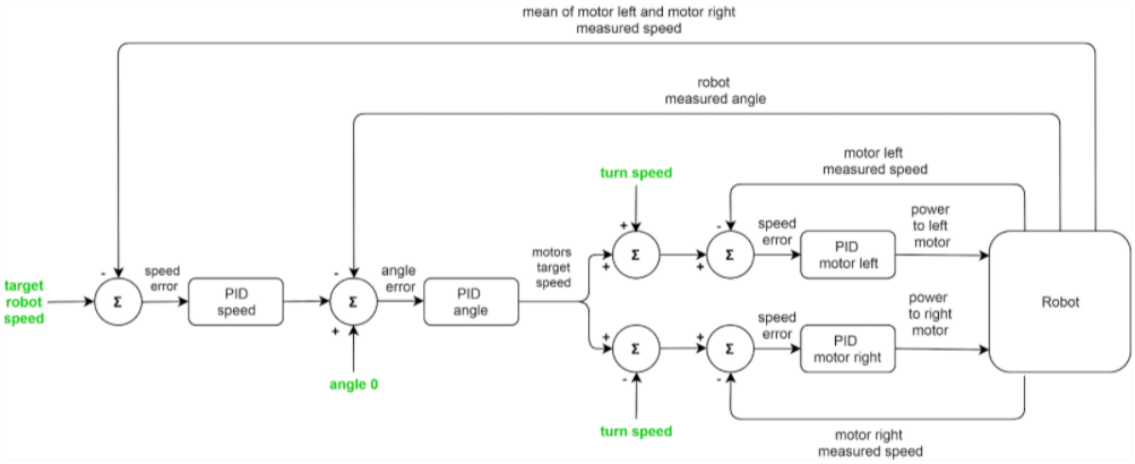
Obrázek 5 Dvoukolový robot
s proměnnou polohou těžiště [6]

Nyní shrneme možnosti návrhu zákona řízení na základě vytvořeného matematického modelu. Použití tohoto přístupu je výhodné, protože simulace na matematickém modelu jsou daleko rychlejší a bezpečnější než experimenty na reálném systému. Je však nutné si uvědomit, že matematický model je pouze aproximací popisovaného systému a nikdy nepopisuje reálný systém dokonale. Na to je třeba pamatovat při návrhu zákona řízení a před jeho aplikací provést nejenom simulace na matematickém modelu, ale také testy na reálném systému.

K řízení nelineárního systému můžeme přistupovat dvěma způsoby. První možností je linearizace matematického modelu systému a nasazení metod lineárních regulace. Úspěšné použití toho postupu uvádí zdroje [1], [2], [5], [7], [9], [10]. Matematický model je linearizován v okolí pracovního bodu, který je přirozeně ve všech zdrojích volen jako požadovaný stav systému, a tím je vzpřímená pozice robota.

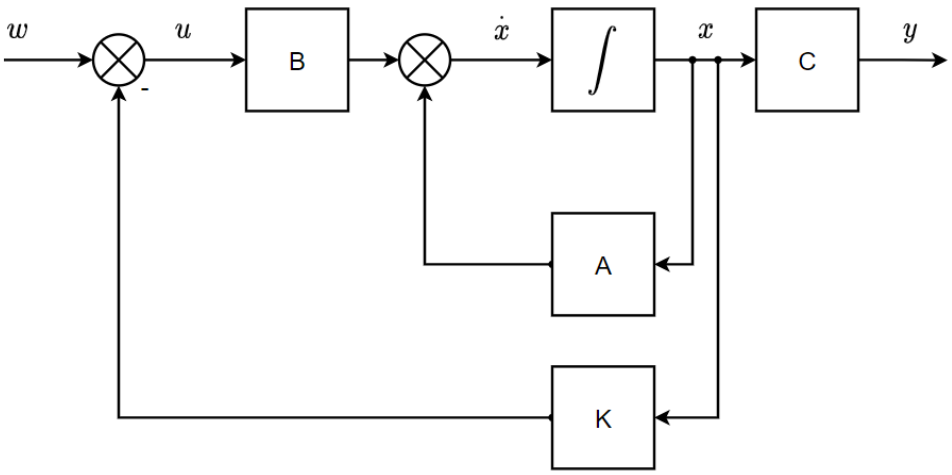
Co se týče jednotlivých typů regulátorů, velmi často zdroje uvádějí použití regulátoru typu PID, jehož struktura je poměrně jednoduchá a při správném seřízení dokáží zajistit poměrně kvalitní a robustní regulaci. V jednoduchých projektech je PID regulátor používán jako dostačující prostředek pro stabilizaci robota ve vzpřímené poloze [9][11]. Jiné zdroje jej uvádějí jako základní, jednoduchý regulátor, se kterým jsou potom

porovnávány vlastnosti jiných sofistikovanějších regulátorů. Dále je zde možnost spojit více PID regulátorů do kaskády za účelem řízení polohy robota v prostoru [12][14].



Obrázek 6 Kaskádní PID struktura řízení dvoukolového robota [12]

V případě složitějších systémů je při vysokých požadavcích na kvalitu regulace vhodné použít řízení ve stavovém prostoru. Implementace stavového řízení výrazně zvyšuje kvalitu regulace, lze ji nasadit i na složité, nelineární systémy, u nichž konvenční teorie řízení neposkytuje požadované výsledky. V porovnání s konvenčními regulátory však vyžaduje jisté teoretické znalosti z oblasti stavového řízení. Samotný návrh stavového regulátoru potom může být proveden metodami jako přiřazení pólů nebo nejčastěji metodou lineární kvadratické regulace (LQR). Metoda LQR spočívá v nalezení vhodného vektoru zesílení *K* viz obrázek 7. [13]



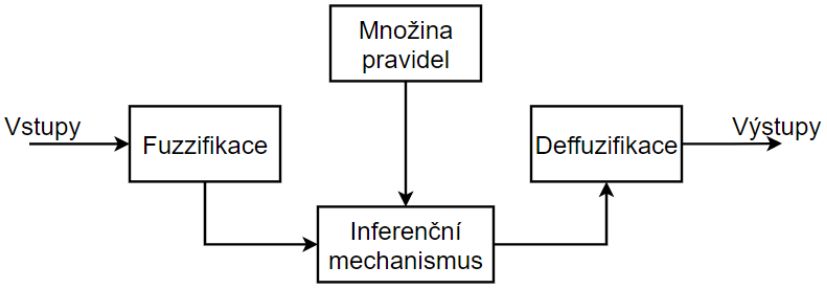
Obrázek 7 Regulační obvod LQR

Tato metoda je velmi často implementována ve spojení s Kalmanovým filtrem, který plní úlohu rekonstruktoru stavů a zároveň i filtru hodnot získaných z IMU jednotky viz kapitola 0. Rekonstruktor stavů je schopný určovat aktuální hodnoty stavů systému a jeho použití je nutné, protože stavy systému většinou nejsou přímo měřitelné, a často je toto měření zatíženo chybou. [13]

Výhodou použití linearizovaného modelu pří návrhu zákona řízení je možnost použití efektivních metod teorie lineární regulace. Nevýhodou je, že navrhnutý regulační obvod je zpravidla funkční pouze v okolí bodu linearizace (pracovní bod).

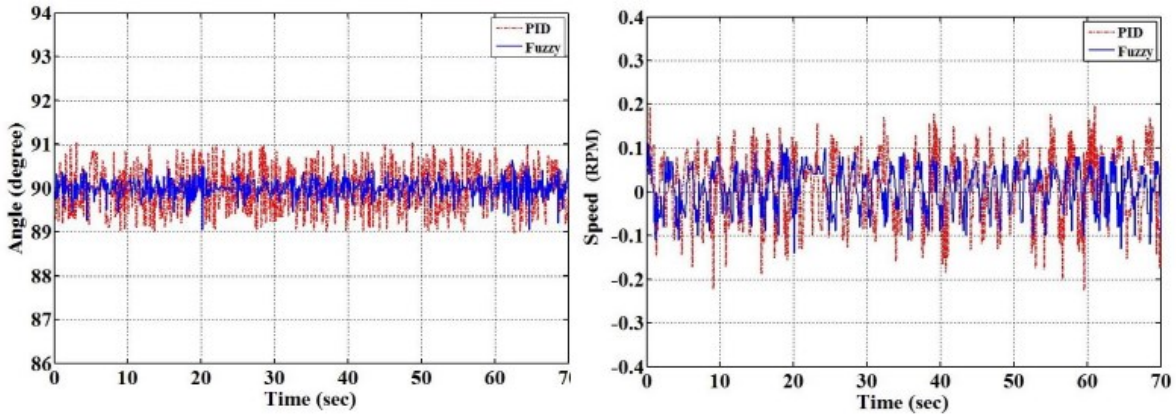
Druhou možností, jak řídit nelineární systém je použití některé z nelineárních regulačních metod. Ve spojitosti s řízením dvoukolového robota můžeme nejčastěji narazit na použití fuzzy regulátoru nebo SMC (Sliding Mode Control) regulátoru.

Fuzzy regulátory jsou zvláštní skupinou regulátorů, které pracují na principu bližším lidské logice. Jednoduše řečeno fuzzy regulátory obsahují množinu pravidel, jakým způsobem reagovat na dané vstupy. Reakcí dle daných pravidel zde myslíme nastavení výstupů takovým způsobem, aby bylo dosaženo požadovaného chování regulovaného systému. Vstupy, v našem případě číselného charakteru, je nutné před zpracováním fuzzy regulátorem upravit do formátu kompatibilního s množinou pravidel, resp. inferenčním mechanismem viz obrázek 8. Tento proces nazýváme fuzzifikace. Poté inferenční mechanismus zpracuje vstupní data a výstup této operace je opět nutné upravit do formátu použitelného k akčnímu zásahu do systému. Tomuto procesu říkáme defuzzifikace. [16]



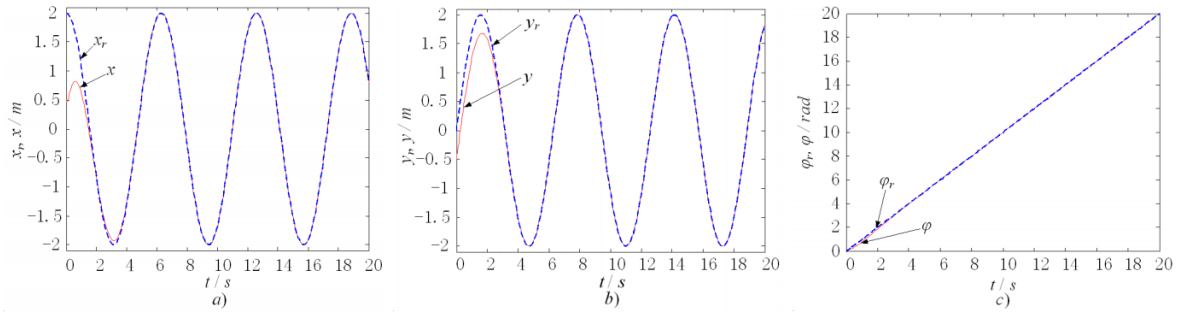
Obrázek 8 Blokové schéma fuzzy regulátoru

Výsledky úspěšného použití fuzzy regulátoru k řízení dvoukolového robota můžeme pozorovat na obrázku 9. Autor zde pro porovnání uvádí i výsledky dosažené PID regulátorem [6].



Obrázek 9 Porovnání fuzzy a PID regulátoru [6]

SMC regulátor neboli regulátor v klouzavém režimu je nelineární regulační algoritmus pracující na principu vysoké frekvence přepínání vstupu (akční veličiny). Základní SMC regulátor přepíná pouze mezi minimální a maximální hodnotou akční veličiny. Přepínání je řízeno tak, aby stav systému vždy směřoval k tzv. přepínací nadploše. Výhodou těchto regulátorů je jednoduchost jejich návrhu a realizace a také robustnost. Nevýhody potom vyplývají z vysoké frekvence přepínaní vstupu, což může mít za následek vybuzení rezonancí systému na vysokých frekvencích nebo rychlé opotřebování mechanických částí systému. Avšak při správném návrhu lze tyto nevýhody potlačit a dosáhnout velmi dobrých výsledků regulace viz obrázek 10. Z obrázku je patrné, že referenční trajektorií při experimentu byla kružnice a poloha robota (x, y, φ) dosáhne požadované trajektorie (x_r, y_r, φ_r) do 3 sekund a poté již velmi přesně sleduje referenční signál. [17][18][28]



Obrázek 10 Výsledky při řízení polohy robota pomocí SMC regulátoru [28]

V souvislosti s konstrukcí dvoukolových robotů je dobré zmínit některé inovativní přístupy. Poslední část této kapitoly je tedy věnována již existujícím prototypům dvoukolových robotů

Mezi nejzajímavější projekty v osobní přepravě nepochybně patří hybridní robotické křeslo iBot vyvíjené firmou DEKA ve spolupráci s firmou Toyota. Mezi hlavní přednosti tohoto robota patří velmi dobrá prostupnost terénem (nerovnosti, schody). [19]

Obrázek 11 Robot iBot

Jako další zástupce je zde robot Ascento, za jehož vývojem stojí švýcarský tým z výzkumného ústavu ETH v Curychu. Na rovném povrchu se robot Ascento chová velmi podobně jako klasicky konstruovaný dvoukolový robot. Narozdíl od klasické konstrukce, nejsou kola spojena se skeletem robota pevně, ale pomocí ramen se dvěma stupni volnosti. Tato konstrukční úprava umožňuje robotovi pohyb v náročném terénu, a dokonce mu dává schopnost přeskakovat překážky. [20]

Obrázek 12 Robot Ascento

Jako třetího zástupce uvedeme robota typu „ballbot“. Zde je dvojice cylindrických kol nahrazena jediným sférickým kolem. K řízení pohybu by teoreticky stačily dva motory, v praxi se však používají tři nebo čtyři motory. Použitím sférického kola získá robot ještě lepší manévrovatelnost než dvoukolový robot. Jedním z nejpropracovanějších exemplářů je

ballbot vyvinutý v institutu robotiky CMU (Carnegie Mellon University). Jedná se o servisního robota vybaveného velmi pokročilým senzorickým systémem. Je schopný interakce se svým okolím pomocí robotických paží a dokáže manipulovat i těžkými objekty. [21]

Obrázek 13 Robot CMU ballbot

1.2 Obecný popis dvoukolového robota

Jedná se o dnes již notoricky známou konstrukci, kterou většina z nás bude znát díky populárním dopravním prostředkům jako je Segway nebo Hoverboard. Samozřejmě existuje mnoho dalších zařízení založených na principu inverzního kyvadla. Menší verze těchto robotů slouží většinou ke studijním účelům, jako názorné modely umožňující uživatelům zkoumat chování a možnosti řízení nestabilních nelineárních systémů. Dále mohou tito roboti sloužit jako inteligentní podvozky nesoucí jiná technická zařízení jako robotický manipulátor nebo pokročilejší senzorické systémy.

Společným rysem těchto zařízení je skelet robota, jehož těžiště je v ideálním případě umístěno přesně nad osu otáčení dvou, většinou separátně poháněných kol. V literatuře se taková struktura označuje jako diferenciální podvozek, protože orientace podvozku se řídí pomocí rozdílných otáček jednotlivých motorů.

1.2.1 Skelet

Slouží jako nosná konstrukce. Spojuje dvojici kol, a jsou na něm nainstalovány potřebné senzory a řídicí elektronika. Při volbě materiálu, kterým je skelet tvořen nejsme nijak omezeni, avšak v praxi se osvědčilo použití lehkých materiálů jako lepenka, dřevo, plast,

slitiny lehkých kovů a jejich kombinace. Důležité je, aby se skelet při vyvažovacích manévrech co nejméně deformoval. Deformace mohou vést ke změnám polohy těžiště, a navíc zde mohou vznikat parazitní setrvačné síly a vibrace, které mohou nepříznivě ovlivnit senzorický systém vyhodnocující aktuální stav robota, což se může projevit např. jako oscilace robota okolo žádané klidové polohy.

Tvar skeletu je symetrický podél osy společné pro obě kola. Důvodem je snaha umístit těžiště skeletu co možná nejpřesněji nad osu symetrie, aby byl robot ve své pracovní, a tedy vzpřímené poloze co nejlépe vyvážený. Na vyvážení robota je dobré myslet i při osazování skeletu již zmíněnými součástmi, a to zejména těmi s podstatně vyšší hmotností jako jsou například akumulátory. Poloha těžiště, respektive jeho pozice na prostorové ose rovnoběžné s vektorem gravitačního zrychlení je významným parametrem ovlivňujícím celkovou dynamiku robota [6]. Výše položené těžiště znamená vyšší moment setrvačnosti, což odpovídá menšímu úhlovému zrychlení vůči ose otáčení. To znamená, že regulátor má více času zareagovat na případné poruchy systému, tedy výchylky robota z rovnovážné polohy.

1.2.2 Pohon kol

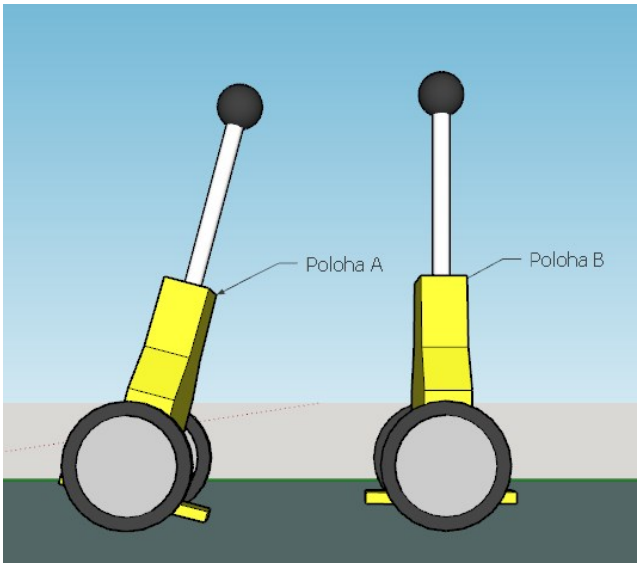
Jako akční členy poháněj ící kola se používají různé druhy elektromotorů. Volba určitého druhu motoru se vždy odvíjí od požadavků dané aplikace. V případě dvoukolového robota jsou podstatné parametry:

- točivý moment

- otáčky motoru

- hmotnost a rozměry motoru

Točivý moment musí být dostatečný, aby byl motor schopen regulačních zásahů, které se projeví na změně náklonu či pozice robota. Nároky na kroutící moment a otáčky motoru budou přímo úměrné velikosti odchylky robota z pracovní *polohy B* – viz obrázek níže. To znamená, že čím více bude robot vychýlen ze vzpřímené polohy, tím větší kroutící moment a vyšší rychlost bude potřeba ke stabilizaci robota. Je tedy vhodné instalovaný výkon motorů dimenzovat tak, aby bylo možné robota uvést z počáteční stabilní *polohy A* do pracovní *polohy B* viz obrázek 14.



Obrázek 14 Základní polohy robota

Tento manévr se často označuje anglickým termínem Swing-Up, a setkat se sním můžeme zejména ve spojitosti se systémy založenými na principu inverzního kyvadla. *Polohu A* lze považovat za maximální možnou výchylku robota vůči podložce, danou konstrukčním návrhem. Nároky na osazené motory a vlastně i samotný regulátor lze tedy při Swing-up manévru považovat za maximální požadavky. Tento manévr však většina dvoukolových robotů nezvládne, ať již z důvodu nedostatečného výkonu motorů nebo linearizace systému při návrhu regulátoru, a tedy jednou z podmínek k zahájení balancování robota je počáteční poloha v blízkém okolí pracovního bodu (vzpřímená nestabilní pozice).

Nejčastěji v realizacích dvoukolových robotů nalezneme stejnosměrný kartáčový nebo bezkartáčový motor, méně často potom krokový motor.

Stejnosměrné kartáčové motory jsou levné a velmi jednoduše se ovládají. Otáčky lze regulovat změnou vstupního napětí a reverzaci motoru docílíme přepólováním. V nízkých otáčkách dosahují vysokého točivého momentu. Účinnost je na dnešní standardy průměrná a pohybuje se od 75 % do 80 %. Hlavní nevýhody těchto motorů jsou způsobeny komutátorem. V mezerách na komutátoru, kde dochází k přepínání rotorových vinutí vznikají elektrické výboje, čímž je generován elektromagnetický šum, který může způsobit rušení v jiných částech systému. Další nevýhodou je mechanické opotřebovávání kartáčů, a tedy nutnost údržby motoru. [38]

Stejnosměrné bezkartáčové motory jsou složitější a obvykle dražší než motory kartáčové. Díky absenci komutátoru a kartáčů však produkují daleko méně elektromagnetického

šumu, jsou tiché a nenáročné na údržbu. Komutace je zde prováděna elektronicky, což ale zvyšuje cenu motoru a také komplikuje jeho řízení. [38]

Krokové motory pracují odlišně než předchozí dva zmíněné motory. Pohybují se nespojitě o přesně daný úhel – krok. Maximální točivý moment jsou schopny dodávat od velmi nízkých otáček až po řádově tisíce RPM. S rostoucími otáčkami však točivý moment klesá a při větší zátěži se zvyšuje riziko vynechání kroku. Dalšími nevýhodami je vysoká spotřeba, zahřívání motoru nezávisle na zátěži a poměrně hlučný provoz. [38]

Momentová charakteristika krokového motoru je pro účely řízení dvoukolového robota vhodnější než charakteristika stejnosměrných motorů. Avšak použitím převodovky v kombinaci se stejnosměrným kartáčovým či bezkartáčovým motorem, lze získat volbou vhodného převodového poměru požadovanou momentovou charakteristiku také.

Osobně bych volil kombinaci stejnosměrného motoru a převodovky. Celá sestava je při daném výkonu menší než krokový motor se srovnatelnými parametry. Řízení je jednodušší než u krokového motoru a průběh kroutícího momentu je především v nízkých otáčkách hladší a nevznikají tak nežádoucí vibrace, které negativně ovlivňují výsledky měření IMU jednotky. Navíc nabídka hotových sestav (motor a převodovka) je široká a díky velké poptávce je pořizovací cena poměrně nízká.

1.2.3 Kola

U dvoukolových robotů se používají výhradně cylindrická kola. Při volbě pláště, respektive povrchu kola, se snažíme volit materiál a případně dezén tak, abychom dosáhli co největší hodnoty součinitele smykového tření mezi kolem a terénem. V případě, že je hodnota součinitele smykového tření nízká, dochází k prokluzu kola. Takové chování potom výrazným způsobem komplikuje řízení robota. To je způsobeno tím, že určitá část energie akčního zásahu se při ztrátě adheze (prokluzu kola) přemění v teplo. Při přenosu točivého momentu kola na podélný pohyb, můžeme vždy pozorovat určitý rozdíl mezi obvodovou rychlostí kola a jeho podélnou rychlostí. Tento rozdíl rychlostí označujeme jako skluz. Závislost adhezního poměru na skluzu potom označujeme jako skluzovou charakteristiku. Vrchol této charakteristiky je dán koeficientem adheze. Při překročení tohoto vrcholu se hodnota skluzu rychle zvyšuje a tečná síla ve styku kola s podložkou klesá a dochází k prokluzu. Tomuto stavu se snažíme předejít volbou vhodného materiálu a dezénu pláště kola vzhledem k předpokládanému povrchu a podmínkám, ve kterých bude

robot pracovat. U většiny robotů, jak laboratorních, tak určených pro osobní přepravu je nejčastěji voleným materiálem pláště pryž a liší se pouze tvarem dezénu. [4] [14]

Materiál použitý na plášť kola určuje schopnost tlumit vibrace, vznikající pohybem robota po podložce. Přenos těchto vibrací skrze kola a skelet robota, až k inerciální měřicí jednotce způsobuje nežádoucí zkreslení signálu.

Dalším důležitým parametrem kola je jeho průměr. Jelikož kolo zde plní i funkci převodovky, kdy podélná rychlost kola je přímo úměrná otáčkám motoru a průměru kola (v ideálním případě):

$$v_p = \omega_{kola} \cdot r_{kola} \tag{4}$$

a výsledná tečná síla, kterou působí kolo na podložku je dána poměrem točivého momentu motoru ku průměru kola:

$$F_T = \frac{M_m}{r_{kola}} \tag{5}$$

1.2.4 Senzorický systém a zpracování signálů

Senzorický systém dvoukolových robotů můžeme rozdělit na několik částí. První a nejdůležitější částí je senzorický systém, který vyhodnocuje náklon, zrychlení a orientaci robota. Tato část senzorického systému je často označována jako inerciální měřicí jednotka nebo anglickým termínem IMU (*Inertial Measurement Unit*), což je obecně zařízení složené často z více inerciálních senzorů sloužící k určení relativní polohy objektu. Nejčastěji používané senzory pro určení náklonu dvoukolového robota jsou uvedeny dále.

Akcelerometr

Je zařízení schopné převést statické i dynamické zrychlení na měřitelný elektrický signál. Akcelerometr je tedy schopný změřit statické gravitační zrychlení. Na základě této informace je pak pomocí výpočtů, možné určit náklon robota od vertikály. Měření zrychlení je prováděno vždy v dané prostorové ose. Dnešním standardem jsou akcelerometry schopné měřit ve všech třech prostorových osách (tzv. 3D akcelerometry). Problémem je, že výstupní signál akcelerometru je zkreslen vysokofrekvenčním šumem. Tento šum je způsoben již zmíněnými nežádoucími vibracemi. Řešením může být použití

filtru, který odfiltruje vysokofrekvenční složky signálu (více dále v sekci Zpracování signálů). [23]

Gyroskop

Je zařízení schopné měřit úhlovou rychlost rotačního pohybu na základě gyroskopického efektu. Ze znalosti vztahu pro úhlovou rychlost:

$$\omega = \frac{d\varphi}{dt} \tag{6}$$

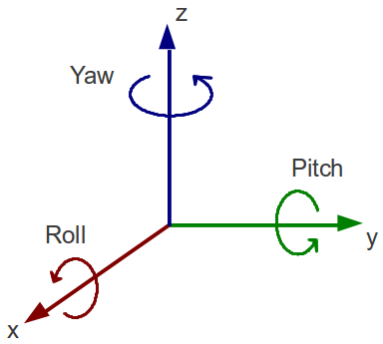
je zřejmé, že pro určení deviačního úhlu je nutné použít integrál:

$$\varphi = \int_{t_0}^{t_1} \omega \, dt \tag{7}$$

A tedy hodnota deviačního úhlu v časovém okamžiku t_1 je vztažená k referenčnímu bodu, kterým je počáteční úhel v okamžiku t_0. Z toho vyplývá, že gyroskop nám poskytuje pouze informaci o změně úhlu, od počátku měření. Výstupní signál je opět zkreslen šumem, což při integraci způsobuje narůstající chybu úměrnou délce měření danou vztahem $(t_1 - t_0)$. [14]

Magnetometr

Je zařízení schopné měřit intenzitu a směr magnetického pole. Toho lze využít při měření magnetického pole země za účelem navigace. V inerciálních měřicích jednotkách magnetometr často doplňuje akcelerometr a gyroskop. Důvodem je, že akcelerometr a gyroskop dosahují podprůměrných výsledků při určení rotace kolem osy Z (*yaw*), v porovnání s přesností určení zbývajících úhlů natočení (*roll* a *pitch*).



Obrázek 15 Rotace okolo prostorových os

Použití magnetometru k určení úhlu natočení zejména okolo osy Z je výhodné hned z několika důvodů. Magnetické pole země můžeme v dlouhodobém horizontu považovat za neměnné. Navíc měření magnetometrem není zatíženo chybou vzrůstající s dobou měření tak jako je tomu u gyroskopu a akcelerometru. Použitím magnetometru tedy získáme referenci pro měření úhlu natočení okolo osy Z (touto referencí může být například poloha severního magnetického pólu). Použití magnetometru v IMU zlepší orientaci robota v prostoru, ale pro úkony jako je stabilizace a základní pohyby v prostoru (dopředu/dozadu) není jeho použití nezbytné. [22]

Dále existují i jiné přístupy založené na měření vzdálenosti určité části robota od podložky, po které se robot pohybuje. K měření vzdálenosti je např. použit infračervený senzor. Při změně náklonu robota se změní i vzdálenost snímače od podložky. Na základě naměřené vzdálenosti je výpočtem určen úhel vychýlení robota. Tento přístup předpokládá velmi přesnou znalost geometrie robota a lze ho použít pouze ke stabilizaci robota. [24]

Další, již volitelnou částí senzorického systému robota mohou být senzory, které slouží ke zlepšení orientace v prostoru. Pokud se má robot nejen udržet ve stabilní poloze, ale i řízeně pohybovat v prostoru, je nutná další zpětná vazba poskytující informaci o jeho poloze. K tomuto účelu se nejčastěji využívají rotační enkodéry nainstalované na kola robota. Díky informaci o úhlové poloze každého kola, resp. o dráze, kterou každé kolo urazilo, jsme schopni přesněji určit relativní polohu robota v prostoru. Zde je však nutno podotknout, že reálná pozice a pozice určená na základě dat z enkodérů se může lišit z důvodu neočekávaných poruch (např. ztráta trakce kol). To lze částečně vyřešit použitím GPS modulu. Zde se přesnost zaměření pozice pohybuje v řádech desítek centimetrů.

Poslední částí, také úzce spjatou s orientací a pohybem v prostoru jsou senzory schopné získávat informace o okolí robota. Mohou to být například informace o překážkách a jejich vzdálenosti. Velmi často se k těmto účelům používají ultrazvukové senzory, zejména pro jejich jednoduchost a malou pořizovací cenu. U profesionálních robotů se potom můžeme setkat s LIDAR senzory, nebo kamerovými systémy s pokročilým zpracováním obrazové informace známé jak strojové vidění. [20] [21]

Provedení IMU jednotek

V inerciálních měřicích jednotkách dvoukolových robotů se nejčastěji setkáme s kombinací akcelerometru a gyroskopu. Někdy se můžeme setkat s označením 6DoF IMU. Termín 6DoF zde však neoznačuje 6 stupňů volnosti, které má hmotné těleso

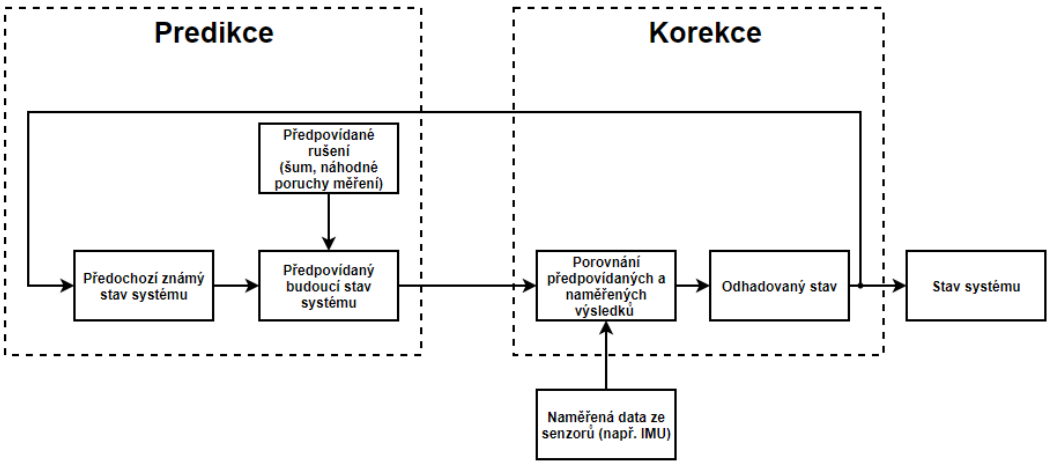
v prostoru. Jedná se o součet stupňů volnosti, které jsou schopné jednotlivé senzory tvořící IMU měřit. Tedy kombinace tříosého gyroskopu a tříosého akcelerometru tvoří právě 6DoF inerciální měřicí jednotku.

Jako nejběžnějšího zástupce můžeme uvést běžně dostupnou IMU s označením MPU-6050. Obsahuje 3D akcelerometr a 3D gyroskop se separátně nastavitelnou citlivostí pro jednotlivé snímače. Dále obsahuje výpočetní jednotku DMP (*Digital Motion Procesor*), která dokáže zpracovat údaje ze senzorů a poskytuje informaci o úhlech natočení ve volitelném formátu jako matice rotace, Eulerových úhlů, kvaternionu či surových dat pomocí I2C sběrnice. [30]

Jako zástupce 9DoF IMU uvedeme integrovaný obvod MPU-9250. Tento čip v sobě skrývá 3D akcelerometr, 3D gyroskop a 3D magnetometr. Opět obsahuje DMP pro preprocessing dat ze snímačů a komunikuje pomocí I2C nebo SPI sběrnice. [31]

Zpracování signálů z IMU jednotky

Jak již bylo uvedeno, signály získané ze senzorů jsou vždy zatížené chybou vlivem rušení. Rušení může být elektromagnetické způsobené vnitřními nebo vnějšími zdroji rušení. Dále za rušení můžeme považovat i mechanické vibrace zkreslující měření IMU jednotkou. Jelikož na základě těchto signálů budeme odhadovat stav systému, je vhodné tyto signály zpracovat a pokusit se tak získat co možná nejpřesnější odhad stavu systému. K tomuto účelu se velmi často používá algoritmus známý jako Kalmanova filtrace. Použití Kalmanova filtru nám umožní nejen filtrovat naměřená data, ale také využít redundance informací ze snímačů ke zpřesnění výsledného odhadu stavu systému. [27]



Obrázek 16 Blokové schéma Kalmanova filtru

Kalmanova filtrace je poměrně složitý, ale užitečný matematický algoritmus pro filtraci signálů v časové oblasti. Skládá se obecně ze dvou částí, a to predikce a filtrace (estimace). Zjednodušený princip je takový, že na základě minulých hodnot signálů (stavů systému) algoritmus odhaduje budoucí hodnoty a tyto pak porovnává se skutečně naměřenými hodnotami. Rozdíl predikovaných a naměřených hodnot potom v dalším kroku použije ke zpřesnění predikce. Praktické použití tento algoritmus našel především jako estimátor vnitřních stavů stochastických systémů, kde jsou měřené signály výrazně zkresleny rušením a náhodnými poruchami měření.

Výhodou Kalmanova filtru je, že dokáže filtrovat signály v časové oblasti téměř stejně kvalitně jako filtrační algoritmy pracující ve frekvenční oblasti. Není tak potřeba transformace vstupního signálu z časové do frekvenční oblasti použitím Fourierovy transformace, což podstatně šetří výpočetní výkon řídicí jednotky. Další výhodou je, že není potřeba znalost informací o vstupním signálu ani o rušení, které na něj působí. [13][27]

1.2.5 Řídicí jednotka

Je elektronický obvod, schopný řídit přímo či nepřímo svými výstupy dané zařízení, na základě vstupních dat zpracovaných pomocí předem definovaného programu. V případě dvoukolového robota řídicí jednotka zpracovává data získaná senzorickým systémem. Na základě těchto dat a požadavků na chování robota potom program řídicí jednotky nastavuje výstupy, pomocí kterých řídí dvojici motorů. Jako řídicí jednotky dvoukolových robotů se nejčastěji používají jednočipové nebo jednodeskové počítače.

Jednočipové počítače

Většina amatérských realizací dvoukolových robotů používá jako řídicí jednotku jednočipový počítač. Výpočetní výkon dnešních mikrokontrolerů je pro účely stabilizace a základní orientace v prostoru více než dostatečný. Mezi další výhody patří malé rozměry, nízká energetická náročnost nebo malá pořizovací cena. Usnadnění práce s mikrokontrolerem přináší použití již hotové vývojové platformy. Tyto vývojové platformy nám ulehčí práci s mikrokontrolerem tím způsobem, že nemusíme řešit napájení, ošetření vstupů a výstupů, komunikaci nebo složité vytváření a nahrávání programu. Nejznámější vývojová platforma *Arduino* má k dispozici vlastní vývojové prostředí a mikrokontroler tak lze programovat ve vyšším programovacím jazyce *Wiring* a používat již připravené knihovny funkcí. [14]

Jednodeskové počítače

Jednodeskový počítač je vysoce integrovaný elektronický obvod, obsahující vše potřebné, aby mohl plnit funkci plnohodnotného počítače. Na rozdíl od jednočipových počítačů tedy poskytuje mnohonásobně vyšší výpočetní výkon a větší kapacitu pamětí. Komunikace s dalšími zařízeními je vyřešena u většiny dostupných modelů přítomností ethernet portu, WiFi, Bluetooth a USB portu. Jako zástupce a velmi často používaný model můžeme uvést počítač *Raspberry Pi*.

Použití jednodeskového počítače jako řídicí jednotky dvoukolového robota má tedy význam, pokud je potřeba vysoký výpočetní výkon. Tato potřeba může vzniknout použitím náročnějších algoritmů řízení, výměnou informací s nadřazeným systémem nebo při zpracování větších objemů dat, například zpracování obrazu. [25]

Běžně používané operační systémy jednodeskových počítačů nejsou vhodné pro použití v aplikacích, kde je nutná rychlá, resp. garantovaná odezva systému.

Jelikož stabilizace dvoukolového robota je náročná na rychlé reakce řízení, je nutné použít operační systém ze skupiny systémů *RTOS* (*Real Time Operating System*). Tyto systémy jsou navrženy tak aby byly schopné reagovat na vnější podměty v předem stanoveném čase. To znamená, že předem známe maximální zpoždění reakce systému na daný podmět. [26]

1.2.6 Zhodnocení konstrukce a možnosti jejího využití

Mezi hlavní výhody této konstrukce patří její jednoduchost. Použitím dvojice motorů a kol získáme kompaktní konstrukci s velmi dobrou manévrovatelností. Tato konstrukce se však nehodí pro pohyb v náročnějších terénních podmínkách. Mezi další nevýhody patří nestabilita v pracovním bodě, a tedy nutnost neustálých akčních zásahů. Z toho plyne, že robot nedokáže ve vzpřímené poloze stát zcela nehybně, ale osciluje kolem žádané polohy.

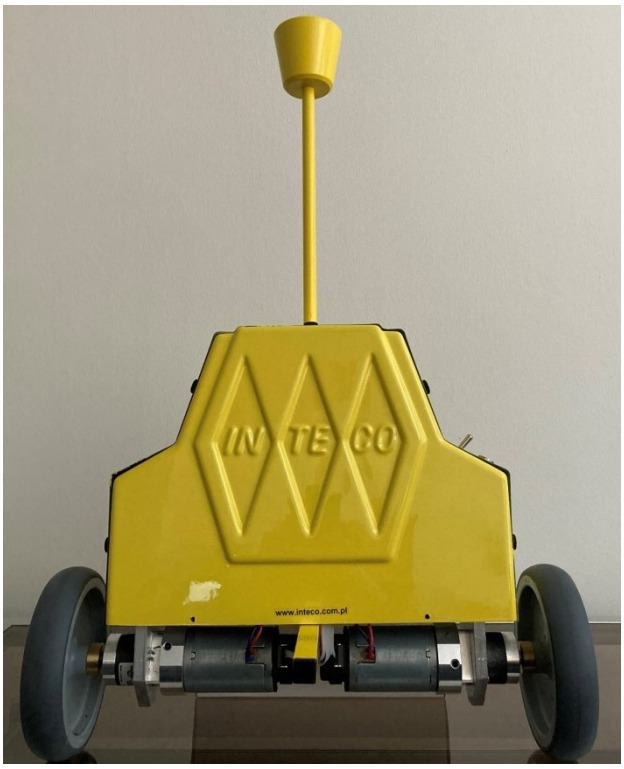
Možnosti využití dvoukolových robotů jsou široké a zasahují do mnoha odvětví. Ať už jsou to laboratorní modely určené pro výuku a experimenty, nebo roboti sloužící k osobní přepravě. Dále mohou dvoukoloví roboti sloužit jako podvozky pro jiná technická zařízení. V těchto aplikacích potom účel robota určují právě tato instalovaná zařízení. Ať už jsou to např. průzkumní roboti jako MegaScout [29] sloužící k průzkumu pro člověka potencionálně nebezpečných oblastí, nebo servisní roboti schopní interagovat se svým okolím.

2 POPIS A ZPROVOZNĚNÍ DVOUKOLOVÉHO ROBOTA INTECO

V této kapitole bude popsán dvoukolový robot vyrobený firmou Inteco [3]. Tento robot je primárně určen pro laboratorní a edukační účely. Následuje popis postupu instalace potřebného softwaru a samotného zprovoznění robota.

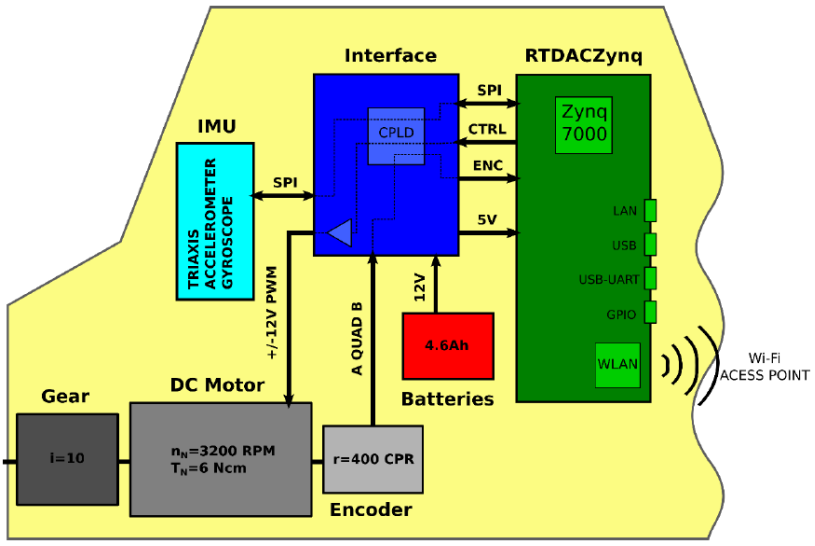
2.1 Dvoukolový robot Inteco

Dvoukolový robot Inteco je zobrazen na obrázku 17. Jedná se o autonomní bateriový systém tzn., že robot ke své funkci nepotřebuje žádný externí ovladač ani zdroj energie a dokáže fungovat bez vnějších zásahů člověka.



Obrázek 17 Dvoukolový robot Inteco

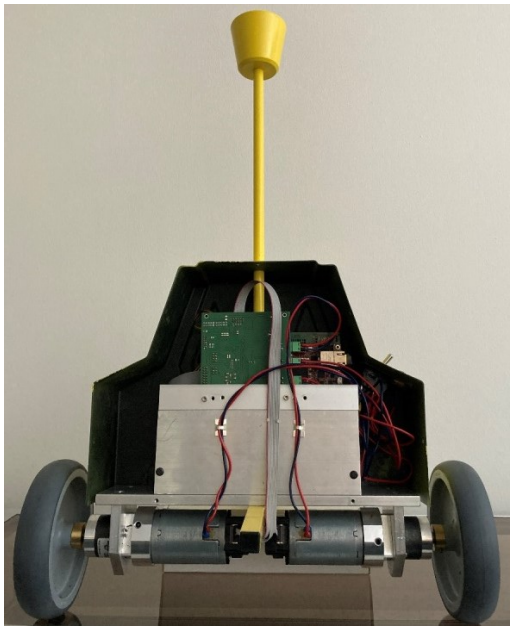
Robot Inteco se skládá ze základních částí viz obrázek 18, které byly popsány v kapitole 1.2 a zde budou uvedeny pouze upřesňující údaje o jednotlivých komponentách.



Obrázek 18 Blokové schéma HW robota Inteco [3]

Skelet

Samotný skelet je tvořen z několika částí vyrobených z lehké hliníkové slitiny viz obrázek 19. Na této konstrukci jsou namontovány všechny komponenty robota a z důvodu ochrany před poškozením jsou kryté plastovými kryty. Poloha těžiště je posunuta směrem od osy otáčení kol přídavnou zátěží.



Obrázek 19 Osazený skelet robota Inteco

Pohon kol

Pohon kol je realizován dvojicí stejnosměrných elektromotorů kombinovaných s planetovou převodovkou s převodovým poměrem 1:10 viz obrázek 20. Maximální otáčky

motoru jsou 3200 RPM a je schopný poskytnout točivý moment 6 Ncm. Řízení motorů je realizováno pulsní šířkovou modulací (PWM).

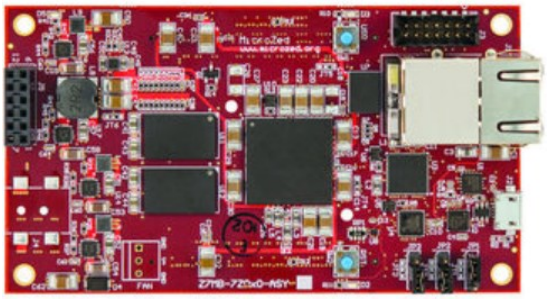
Obrázek 20 Instalovaný DC motor s převodovkou a enkodérem

Senzorický systém

Robot Inteco je vybaven IMU jednotkou, která obsahuje 3D akcelerometr a 3D gyroskop. Senzory poskytují naměřená data pomocí rozhraní SPI. Senzorický systém robota je dále doplněn od dva inkrementální rotační enkodéry nainstalované na hřídele motorů. Tyto enkodéry mají rozlišení 400 pulsů na otáčku a poskytují nám informaci o úhlové poloze, resp. úhlové rychlosti jednotlivých kol a umožňují tak přesnější řízení polohy robota v prostoru.

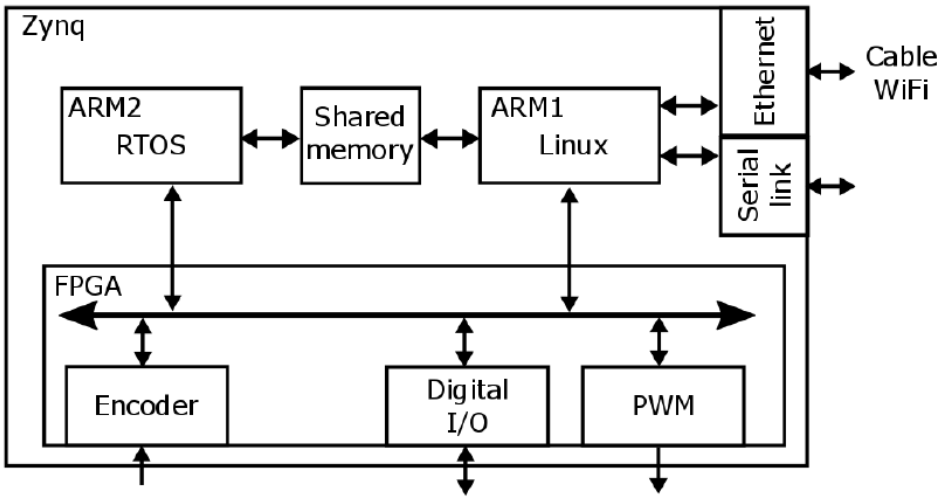
Řídicí jednotka

Řídicí jednotka je složena z několika částí. První část je tvořena vývojovou deskou *MicroZed* založenou na *Xiling Zyng-7000 All Programmable SoC* a obsahuje dva procesory *ARM A9*. Deska *MicroZed* (obrázek 21) zajišťuje běh RT aplikace, komunikaci s externím počítačem pomocí WiFi AP, odkud jsou načítány řídicí algoritmy a vizualizována naměřená data. Deska může fungovat samostatně, nebo osazená na speciální rozšiřující kartě jako *SoM*.[32]



Obrázek 21 Řídicí deska *MicroZed* [32]

Rozšiřující karta RT-DAC v případě robota Inteco obsahuje čip CPLD a zajišťuje funkci digitálních vstupů a výstupů, zpracování signálů z enkodérů a generování PWM signálu pro řízení motorů.



Obrázek 22 Blokové schéma řídicí jednotky [3]

Na obrázku 22 můžeme vidět blokové schéma řídicí jednotky robota Inteco. Dvojice ARM procesorů pracuje v asymetrickém multiprocesorovém módu (ASMP). Na procesoru ARM1 běží operační systém na bázi Linuxu, konkrétně PetaLinux verze 2016.3. Zde jsou zpracovávány úlohy, u kterých není rychlost odezvy systému kritická. Na procesoru ARM2 běží systém reálného času (RTOS). Zde jsou v reálném čase vykonávány řídicí algoritmy vygenerované kompilátorem ze Simulinkových schémat. [3]

2.2 Zprovoznění robota Inteco

V této podkapitole bude detailně popsána procedura zprovoznění robota Inteco. V první části jsou popsány minimální požadavky pro zprovoznění a práci s robotem. Druhá část popisuje instalaci potřebného softwaru. Poslední část této podkapitoly se věnuje prvnímu spuštění a kontrole správné funkce instalovaného softwaru a samotného robota.

2.2.1 Minimální požadavky

Hardwarové požadavky

- Dvoukolový robot Inteco
- Napájecí adaptér *H-Tronic AL 800* (součást dodávky firmou Inteco)
- Osobní počítač disponující technologií WLAN

Softwarové požadavky

- Operační systém Windows (podporované verze jsou Windows 7, Windows 8, Windows 10)
- Matlab (podporované verze jsou R2018a, R2020a)
- Inteco UnTrans software (součást dodávky firmou Inteco)

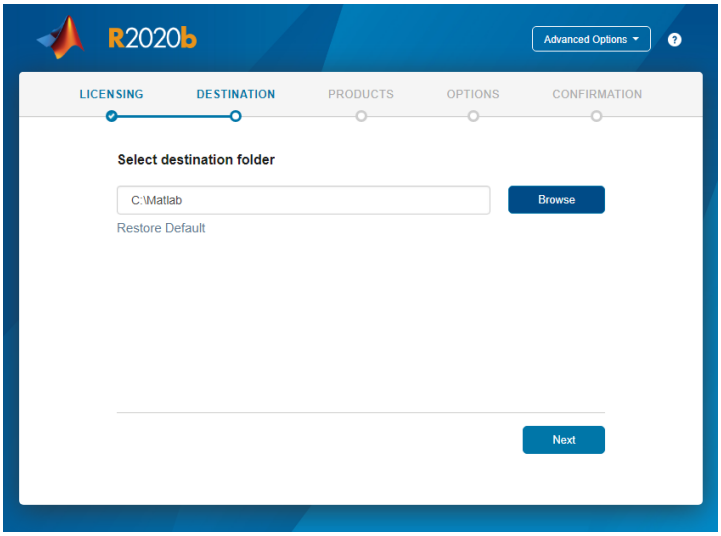
2.2.2 Instalace softwaru

Veškeré následující úkony byly provedeny na dvou odlišných zařízeních z důvodu odhalení případných problémů při instalaci potřeného softwaru:

- Laptop
  - Procesor AMD
  - Operační systém Windows 10
- Desktop
  - Procesor Intel
  - Operační systém Windows 10

Instalace systému Matlab

Software je dostupný na stránkách výrobce MathWorks. V případě potřeby zde nalezneme i starší verze systému Maltab. Po stažení a spuštění instalačního balíčku postupujeme dle pokynů průvodce instalací. **V instalačním kroku, kde volíme cílový adresář pro instalaci je nezbytně nutné zvolit umístění adresáře tak, aby se nenacházel v adresáři Program Files** viz obrázek 23.



Obrázek 23 Volba výchozího adresáře pro instalaci

Tento krok je nezbytně nutný pro následující instalaci softwaru Inteco. Důvodem je problém s přístupovými právy do adresáře Program Files v operačním systému Windows. V následujícím kroku vybereme požadované části systému, které mají být nainstalovány:

- Simulink

Je grafické rozhraní systému Matlab, sloužící k modelování, simulování a analýze systémů. Systémy jsou zde vytvářeny primárně pomocí funkčních bloků, jako blokové diagramy. Základní knihovny funkčních bloků lze doplnit o další rozšíření třetích stran, či vytvářet vlastní funkční bloky.

- Matlab Coder

Tento toolbox dokáže generovat kód v jazyce C/C++ ze zdrojového Matlab kódu. Vygenerovaný kód lze dále použít na různých hardwarových platformách od desktopových aplikací až po vestavné systémy.

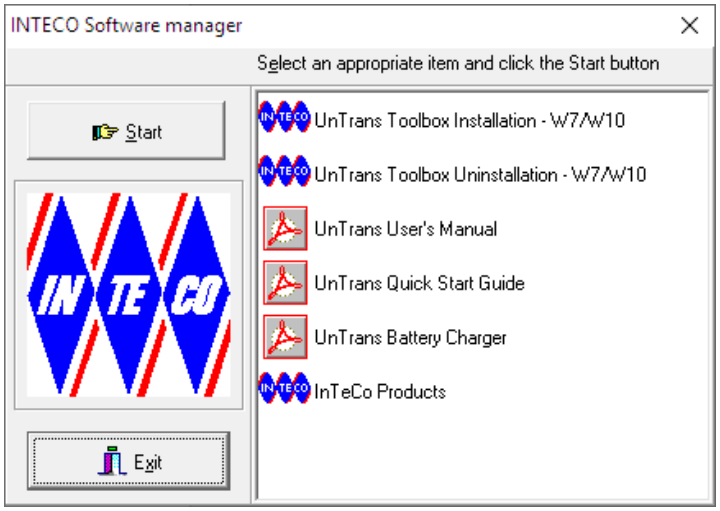
- Simulink Coder

Je toolbox určený k převodu schémat vytvořených v prostředí Simulink do kódu v jazyce C/C++. Vygenerovaný kód lze použít pro aplikace v reálném čase. Dále lze vygenerovaný kód spustit na cílovém zařízení a ladit nebo sledovat průběhy signálů pomocí Simulinku. Tento toolbox vyžaduje ke své funkci toolbox Matlab Coder a nelze ho nainstalovat samostatně.

Dokončíme instalaci systému Matlab dle pokynů průvodce instalací.

Instalace softwaru Inteco

Instalaci zahájíme spuštěním aplikace s názvem MANAGER.exe, která se nachází v kořenovém adresáři instalačního balíčku dodaného firmou Inteco. Doporučuji tuto aplikaci spustit jako správce a ujistit se, že není spuštěný systém Matlab. Po spuštění aplikace se zobrazí jednoduchý rozcestník viz obrázek 24.



Obrázek 24 Instalační aplikace Inteco

Pro instalaci zvolíme položku „UnTrans Toolbox Installation – W7/W10“ a potvrdíme tlačítkem Start. Dále pokračujeme dle pokynů průvodce instalací až ke kroku, kde volíme verzi a umístění systému Matlab viz obrázek 25.

Obrázek 25 Instalace softwaru Inteco

Verze operačního systému je detekována automaticky. Dále zvolíme verzi a umístění kořenového adresáře systému Matlab viz *Instalace systému Matlab*. Kliknutím na tlačítko další proběhne instalace potřebných souborů včetně instalace Linaro cross-compileru [33]. Po dokončení instalace je vhodné restartovat operační systém.

V původní dodávce od firmy Inteco byl dodán software určený pro Matlab R2018. Ve verzi Matlab R2020 tento software nefunguje. Ve spolupráci s firmou Inteco vznikl nový software fungující ve Matlab R2020. Součástí přílohy této práce jsou obě verze softwaru. U verze softwaru pro Matlab R2020 bylo dále nutné upravit některé části Simulinkových DEMO schémat viz 2.2.3 První spuštění a test.

2.2.3 První spuštění a test

Před prvním spuštěním jsem provedl důkladnou kontrolu hardwarového stavu robota.

Bylo zkontrolováno:

- mechanický stav robota
- kontrola zapojení vnitřních obvodů a svorkovnic
- kontrola stavu akumulátoru
- kontrola funkce napájecího adaptéru

Jediným zjištěným problémem byla nulová kapacita akumulátoru. Provedl jsem tedy výměnu akumulátoru.

Spuštění robota

Spuštění robota se provádí pomocí hlavního vypínače viz obrázek 26. Spuštění je indikováno kontrolní LED diodou viz *Signalizace*. Pokud se spuštění nepodaří postupujeme dle kapitoly 2.2.4.

Obrázek 26 Hlavní vypínač

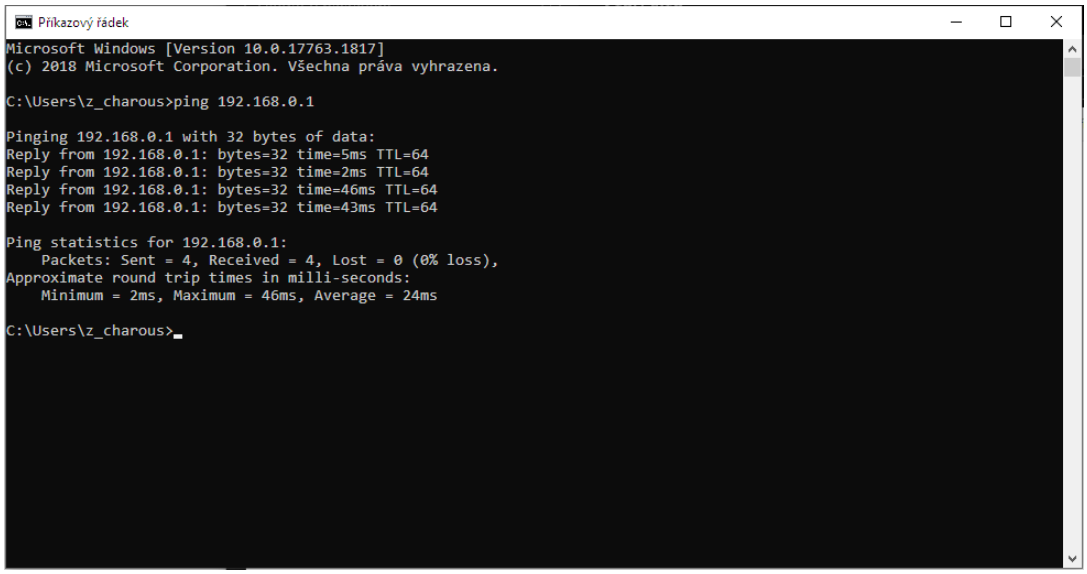
Připojení k systému UnTrans

Prvním krokem po úspěšném spuštění je připojení externího počítače k systému UnTrans. K tomu slouží bezdrátový přístupový bod (Wifi Access Point). Po spuštění systému je potřeba počkat, než se dokončí spouštěcí procedury systému UnTrans (cca 1 min) a je možné se k systému připojit. K přístupovému bodu se připojíme jako ke standardní WiFi síti pomocí následujících údajů:

- Název sítě (SSID): InTeCo_UnTrans
- Heslo: UnTrans7

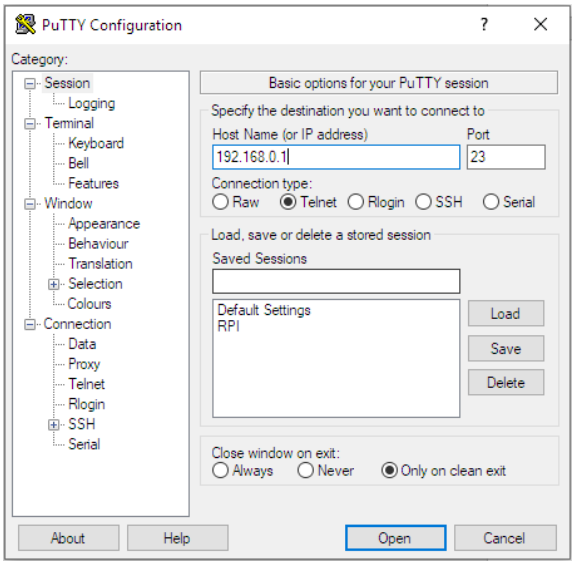
Adresa zařízení UnTrans (IP) je ve výchozím nastavení 192.168.0.1. Provedeme kontrolu připojení zadáním příkazu „ping 192.168.0.1“ do příkazové řádky systému Windows, kterou spustíme stisknutím klávesové zkratky „Windows + R“ a zadáním příkazu „cmd“.

Pokud je spojení se systémem UnTrans v pořádku, měl by výstup příkazové řádky vypadat jako na obrázku 27.



Obrázek 27 Kontrola připojení k systému UnTrans

Dále je potřeba zkontrolovat, zda je možné do systému UnTrans přistupovat. To lze provést například pomocí aplikace Putty [34]. Pomocí aplikace (obrázek 28) se připojíme k systému UnTrans komunikačním protokolem telnet.

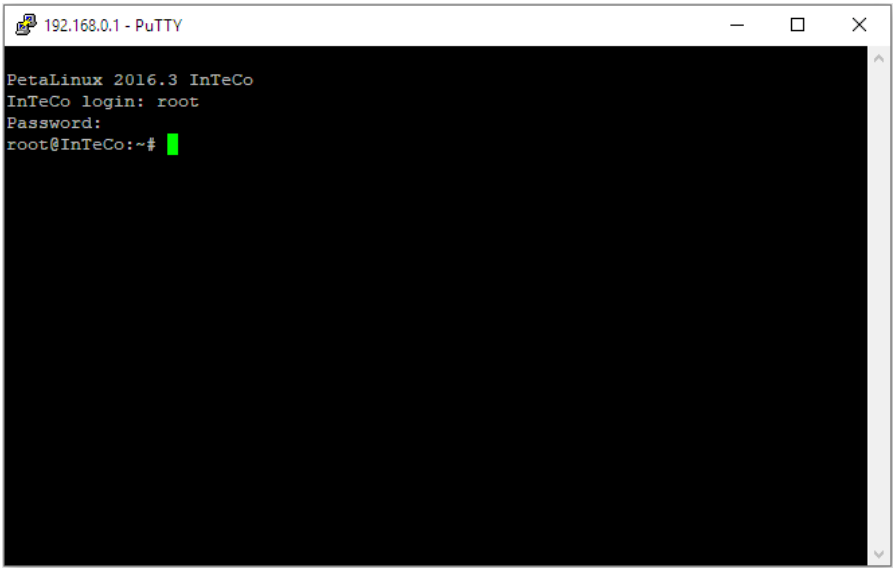


Obrázek 28 Aplikace Putty

Přihlašovací údaje jsou následující:

- Uživatel (login): root
- Heslo (password): root

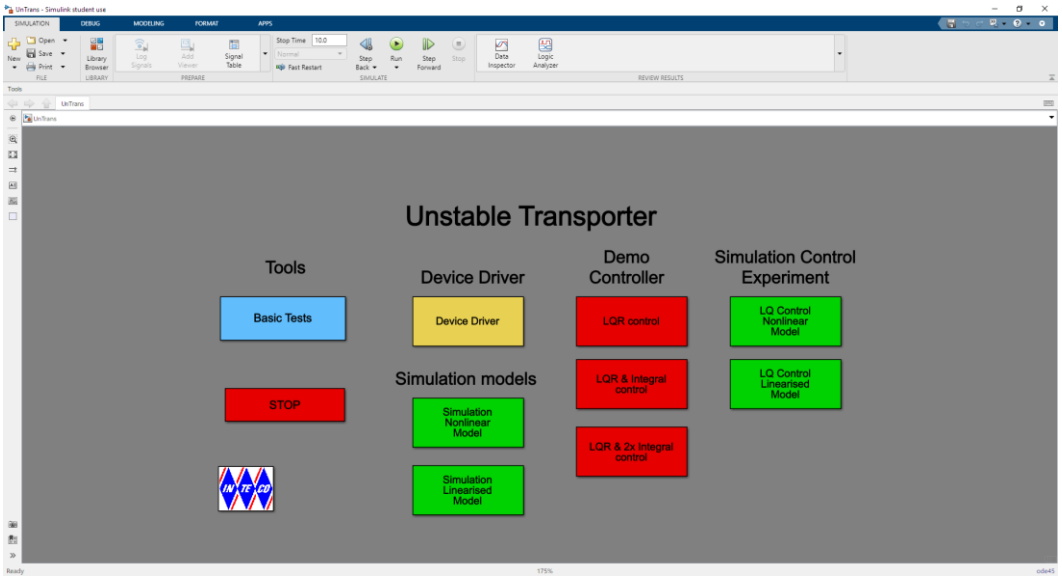
Po úspěšném přihlášení viz obrázek 29 můžeme spojení ukončit a přistoupit k dalšímu kroku.



Obrázek 29 Spojení se systémem UnTrans

Spuštění aplikace UnTrans v systému Matlab

Spuštění řídicí aplikace v systému Matlab provedeme zadáním příkazu „UnTrans“ do příkazového okna. Pokud instalace proběhla v pořádku, mělo by se zobrazit hlavní okno řídicí aplikace UnTrans viz obrázek 30.

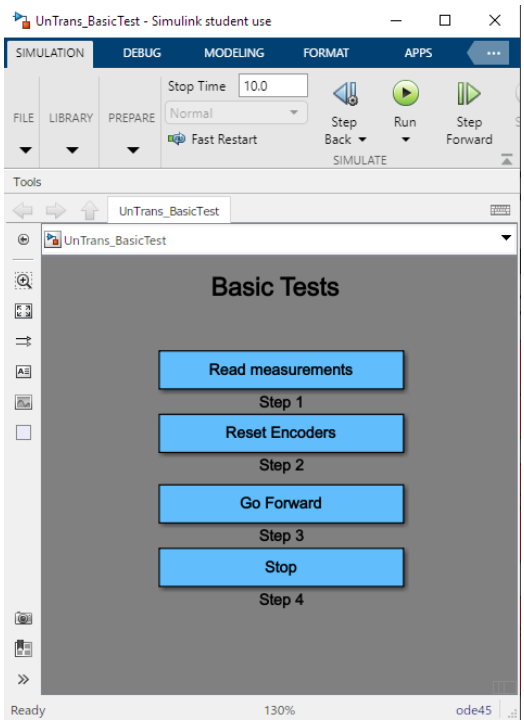


Obrázek 30 UnTrans aplikace

Hlavní okno je rozděleno do několika základních sekcí. Následuje popis jednotlivých položek a jejich využití.

Sekce Tools - Basic Tests

Při prvním spuštění bude objektem našeho zájmu především tato položka. Nabídka obsahuje procedury pro ověření správné funkce všech důležitých částí robota viz obrázek 31.



Obrázek 31 Sekce Basic Tests

Read measurements

Spuštění této procedury vypíše do hlavního okna systému Matlab hodnoty z jednotlivých senzorů robota viz obrázek 32.

Obrázek 32 Test snímačů robota Inteco

Reset Encoders

Pomocí této položky lze vynulovat čítače, do kterých se zapisují údaje z inkrementálních enkodérů.

Go Forward

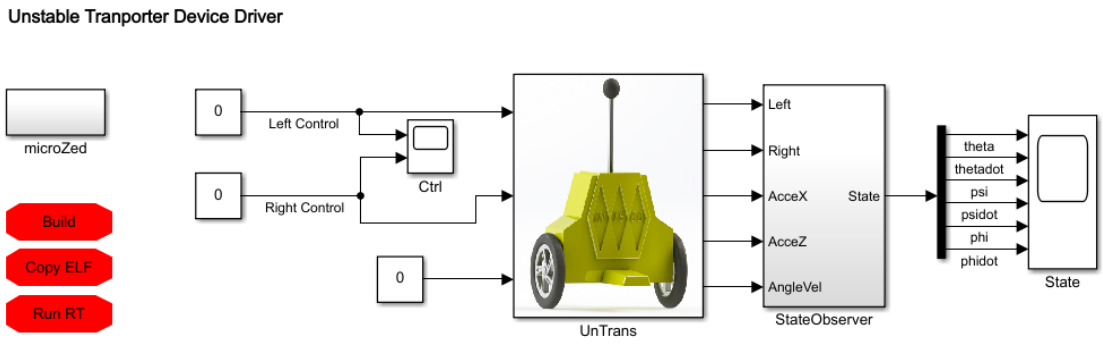
Tato položka slouží k ověření funkce motorů robota. Před jejím použitím doporučuji robota umístit tak, aby kola nebyla v kontaktu s podložkou ani jinými objekty. Po spuštění této procedury se motory začnou okamžitě otáčet. Vypnout je lze použitím poslední položky *Stop* nebo vypnutím robota hlavním vypínačem.

Stop

Tato položka slouží k zastavení otáčení motorů.

Device driver

Tato položka otevře Simulinkové schéma obsahující základní strukturu pro řízení robota Inteco viz obrázek 33. Jedná se o funkční šablonu, kterou doporučuji použít jako výchozí schéma pro návrh vlastního řídicího algoritmu.



Obrázek 33 UnTrans Device Driver

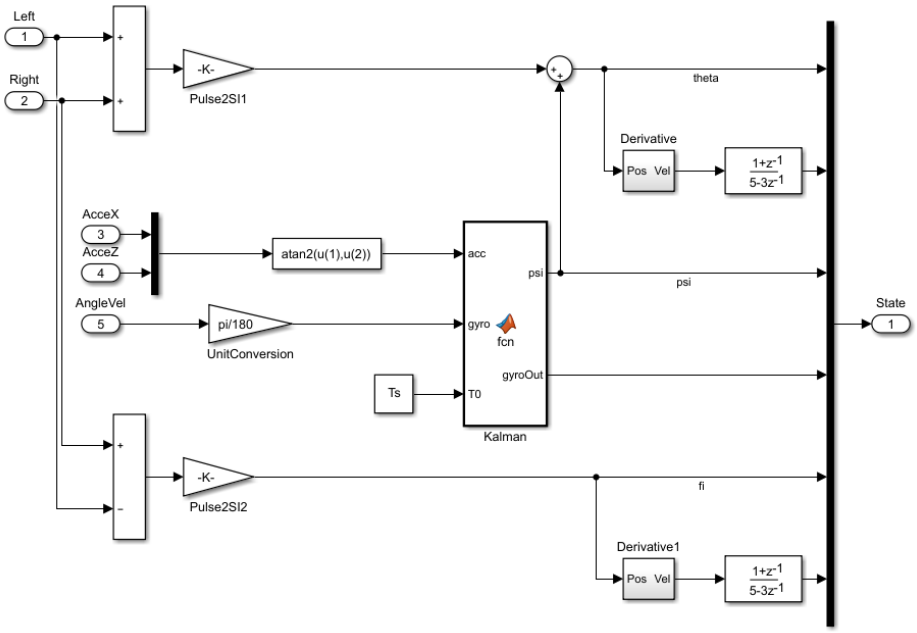
Toto schéma obsahuje dva důležité základní bloky:

- Blok *UnTrans*

  Zprostředkovává komunikaci s řídicí deskou. Přijímá signály ze senzorů a odesílá PWM signály, které řídí jednotlivé motory.

- Blok *State Observer*

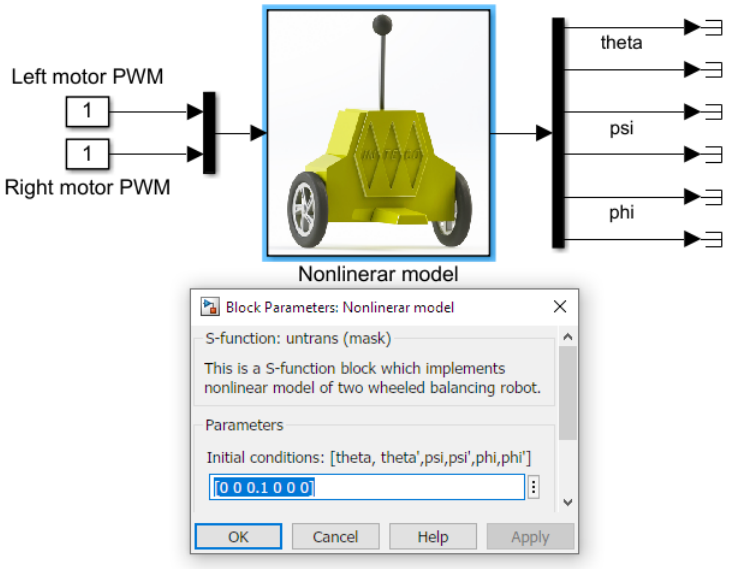
  Výstupem tohoto bloku jsou odhady aktuálních stavů systému. Vnitřní struktura bloku je na obrázku 34. Stavy systému jsou odhadovány pomocí tzv. Kalmanova filtru viz kapitola *1.2.4*.



Obrázek 34 State Observer

Simulation models

Tato nabídka obsahuje nelineární a linearizovaný model systému. Nabídka funkčního bloku umožňuje nastavit počáteční podmínky systému viz obrázek 35.

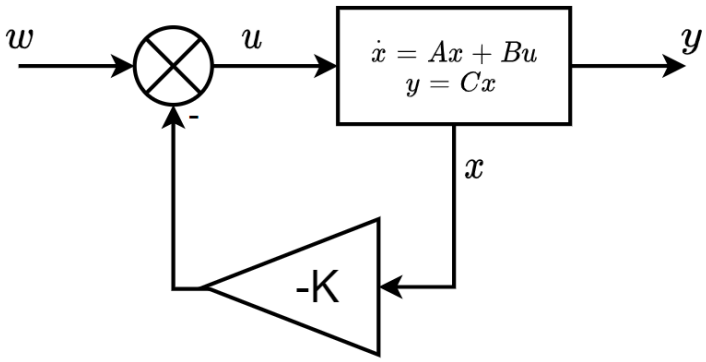


Obrázek 35 Nelineární model UnTrans

Demo controller

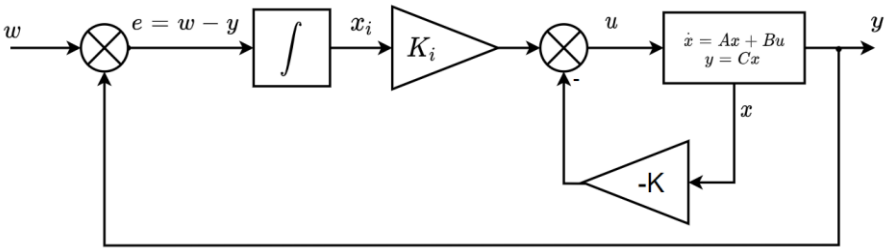
Nabídka obsahuje funkční ukázkové příklady regulačních algoritmů. Ve všech třech případech se jedná o lineární kvadratické stavové regulátory.

V prvním případě se jedná o klasický LQR regulační obvod se strukturou viz obrázek 36.



Obrázek 36 Struktura regulačního obvodu pro LQ řízení

Následují dva regulační obvody jsou typu LQI tedy lineární kvadratické integrační regulátory, někdy také nazývané „LQ servo“. Základní strukturu můžeme vidět na obrázku 37.

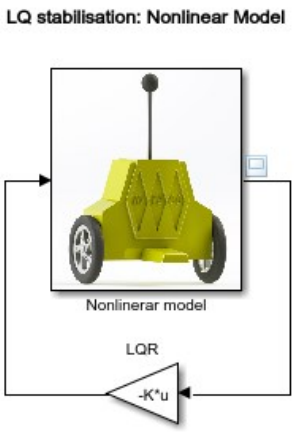


Obrázek 37 Typický LQI regulační obvod

Rozdíl mezi LQ a LQI regulátorem je ten, že zatímco LQ regulátor získává informace pouze o stavech systému, LQI regulátor pracuje nejen se stavy systému, ale také s jeho výstupy a navíc obsahuje integrátor. Díky tomu je LQI regulátor schopen pracovat tak, že výstup systému sleduje např. konstantní referenční signál. Pokud se jedná o MIMO systém, jako v našem případě, potom každá sledovaná výstupní veličina má vlastní integrátor.

Simulation Control Experiment

Poslední nabídka obsahuje dvě položky. Jedná se o simulační experimenty LQ regulátoru nasazeného na matematický model robota Inteco za účelem stabilizace robota ve vzpřímené poloze. Tento regulátor je aplikován jak na linearizovaný, tak na nelineární matematický model robota viz obrázek 38.



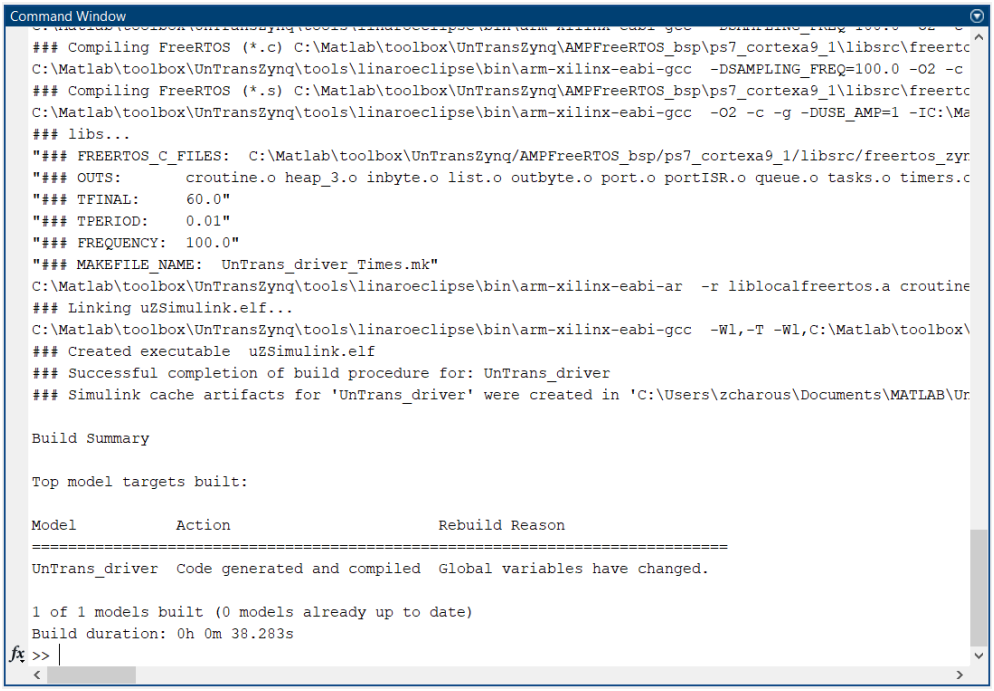
Obrázek 38 Schéma simulačního experimentu

Nahrání a spuštění RT aplikace

Pokud všechny procedury z nabídky *Basic Tests* jsou spustitelné a proběhnou korektně, je možné přistoupit k nahrání a spuštění RT aplikace. Jako první doporučuji nahrát a spustit RT aplikaci *Device Driver* (obrázek 33). Celý proces nahrávání aplikace lze provést pomocí tří tlačítek, které nalezneme jak ve schématu *Device Driver,* tak ve všech schématech nabídky *DEMO Controler*.

Obrázek 39 Tlačítka pro nahrávání RT aplikace

Build tlačítko slouží k vytvoření instalačního balíčku nebo spustitelné aplikace na základě zdrojového kódu, což je v tomto případě schéma vytvořené v prostředí Simulink. K tomuto procesu je nutné použít výhradně tlačítko Build (obrázek 39). Toto je zdůrazněno, protože systém Simulink obsahuje také tlačítko Build, ale jeho použití skončí chybou. Pokud vše proběhne správně, v hlavním okně systému Matlab by se měla zobrazit zpráva shrnující průběh procesu viz obrázek 40.

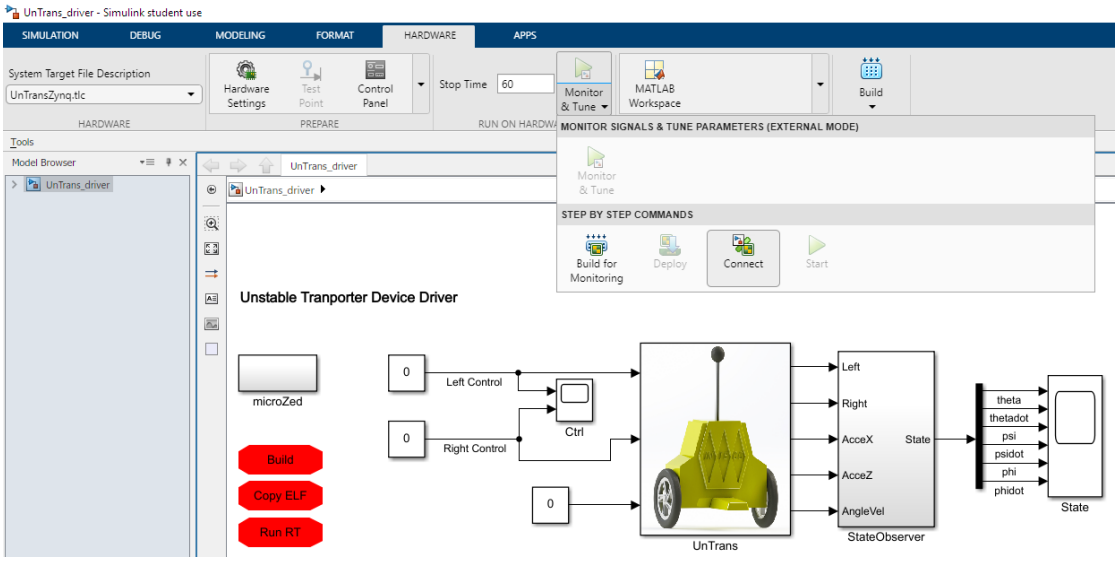


Obrázek 40 Úspěšný build RT aplikace

Copy ELF tlačítko slouží k nahrání připravené RT aplikace do systému UnTrans. Úspěšné dokončení operace je indikováno hlášením v systému Simulink a také LED signalizací.

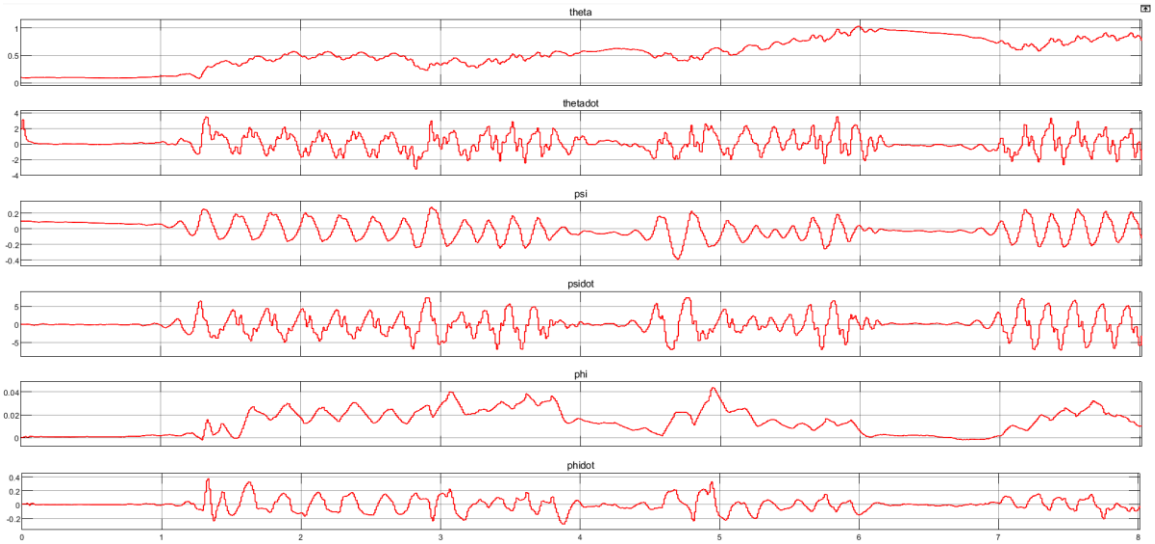
Run RT tlačítko slouží ke spuštění RT aplikace nahrané do systému UnTrans. Úspěšné spuštění je indikováno hlášením v systému Simulink a také LED signalizací. Tím se RT aplikace v systému UnTrans spustí a přepne do stavu „připravena“.

Dalším krokem je připojení systému Simulink k systému UnTrans. Toto spojení potom umožní přepnout RT aplikaci do stavu „běží“, sledovat a zaznamenávat průběhy jednotlivých signálů v regulačním obvodu nebo měnit proměnné parametry. Připojení systému Simulink provedeme tlačítkem *Connect* v nabídce *Monitor&Tune*. Po úspěšném spojení je možné přepnout aplikaci do stavu „běží“ pomocí tlačítka *Start*, které se také nachází v nabídce *Monitor&Tune*. RT aplikace potom běží po dobu určenou parametrem *Stop Time* v systému Simulink. Po uběhnutí této doby se RT aplikace vypne. Pokud RT aplikace udržovala robota ve vzpřímené poloze, potom se po jejím vypnutí robot nekontrolovaně přesune do nejbližší stabilní polohy (mechanický doraz). Všechny popisované ovládací prvky můžeme vidět na obrázku 41.



Obrázek 41 Spuštění RT aplikace v prostředí Simulink

Po úspěšném spuštění RT aplikace *Device Driver* doporučuji znovu vyzkoušet ovládání jednotlivých motorů pomocí nastavení konstant *Left Control* a *Right Control* (v intervalu 0 až 1). Zde je opět vhodné, aby kola nebyla v kontaktu s podložkou ani jinými objekty. Dále doporučuji zkontrolovat správnou funkci IMU jednotky a bloku *State Observer* pomocí osciloskopu (blok *State* v uvedeném schématu výše). Při tomto testu je nutné robotem pohybovat manuálně, nejlépe rotací okolo osy otáčení kol a okolo vertikály. Výstup může vypadat jako na obrázku 42 níže.



Obrázek 42 Výstup bloku State Observer

Signalizace

Robot je vybaven signalizační LED diodou (obrázek 43). Jednotlivé barvy LED diody indikují aktuální stav robota:

- Stav „*Neznámý*“ je indikován bílou barvou LED diody. V tomto stavu se systém UnTrans nachází po zapnutí robota nebo po restartu ARM2 procesoru.
- Stav „*Připraven*“ je indikován modrou barvou LED diody. V tomto stavu se systém UnTrans nachází po úspěšném nahrání RT aplikace. RT aplikace je připravena a systém čeká na příkaz k jejímu spuštění z externího počítače, tj. stisknutí tlačítka *Run* v prostředí Simulink.
- Stav „*Běží*“ je indikován červenou barvou LED diody. V tomto stavu je systém řízen nahranou RT aplikací a v prostředí Simulink je možné nastavovat proměnné parametry nebo sledovat průběh signálů.
- Stav „*Ukončen*“ je indikován zelenou barvou LED diody. RT aplikace je ukončena. Systém čeká na nahrání a spuštění RT aplikace.

Obrázek 43 Signalizační LED dioda

2.2.4 Řešení možných problémů

Problém: Systém Inteco nelze zapnout.

Projevy: Systém Inteco nelze zapnout hlavním vypínačem.

Řešení: Je nutné zkontrolovat stav baterií, popřípadě provést jejich nabití pomocí nabíjecího adaptéru.

Problém: Připojení k systému Inteco není stabilní.

Projevy: Systém Inteco se často odpojuje od řídicího počítače, což má za následek nekorektní chování systému Matlab.

Řešení: Doporučuji vypnout možnost automatického připojení pro všechny bezdrátové sítě v dosahu řídicího počítače kromě sítě systému Inteco. Pokud dojde v průběhu práce s robotem k neočekávanému krátkému výpadku spojení se systémem Inteco, potom doporučuji restart systému Inteco a také systému Matlab.

Problém: Chování systému Inteco neodpovídá předpokladům, získaných na základě simulací nebo po předchozích experimentech.

Projevy problému: Systém nelze stabilizovat dříve odladěným regulačním obvodem při srovnatelných provozních podmínkách. Při stabilizaci systému se objevují silné vibrace. Robot se otáčí okolo své vertikální osy rotace, aniž by tato rotace byla požadována.

Řešení: Nutná kontrola mechanického stavu robota. Zejména dotažení šroubů kol. Dále kontrola dotažení šroubů krytů robota a kontrola dotažení tyče unášející horní závaží.

Problém: Instalované motory se netočí.

Projevy problému: Systém Inteco nelze řídit z důvodu nulové odezvy instalovaných akčních členů.

Řešení: Je nutné dobít baterie systému Inteco. Stav nabití baterií není do systému Inteco implementován, takže při jakémkoliv nekorektním chování systému Inteco doporučuji jako první krok řešení dobít baterie.

3 MATEMATICKÝ MODEL ROBOTA

V této kapitole bude popsán postup vytvoření matematického modelu robota Inteco. První část kapitoly popisuje zvolený postup při tvorbě matematického modelu v obecném tvaru. Následuje popis vytvoření nelineárního matematického modelu robota Inteco. Konec kapitoly je věnován linearizaci odvozeného nelineárního modelu.

3.1 Postup při vytváření matematického modelu

Samotný postup vytváření matematického modelu procesu můžeme rozdělit do několika dílčích kroků:

- Vytvoření matematického popisu procesu na základě fyzikálních zákonů
- Zjednodušení modelu a jeho úprava do vhodné formy
- Vývoj simulačního modelu, resp. implementace matematického modelu do prostředí vhodného pro simulace a další práci s modelem
- Vyladění parametrů modelu (identifikace reálného procesu)
- Praktické ověření modelu

K vytvoření matematického popisu procesu lze přistoupit různými způsoby. Newtonovy pohybové rovnice sice umožňují popsat kompletní mechanický pohyb, avšak z matematického hlediska tato metoda vede na velmi složitý problém. Jisté zjednodušení tohoto problému přináší již zmíněný přístup tzv. Lagrangeovská formulace mechaniky. Tento přístup používá k popisu Lagrangeův tvar Newtonových pohybových rovnic. V této formulaci se k popisu systému používají zobecněné souřadnice. Postup je potom následující:

- Zvolíme zobecněné souřadnice

$$\begin{matrix} q_1 \\ \vdots \\ q_r \end{matrix} \tag{8}$$

- Vyjádříme celkovou kinetickou energii T a celkovou potenciální energii U jako sumu kinetických a potenciálních energií jednotlivých částí systému pomocí zvolených zobecněných souřadnic a jejich derivací.

- Pomocí celkové kinetické a potenciální energie určíme Lagrangeovu funkci („lagrangián“)

$$L(q, \dot{q}) = T(q, \dot{q}) - U(q) \tag{9}$$

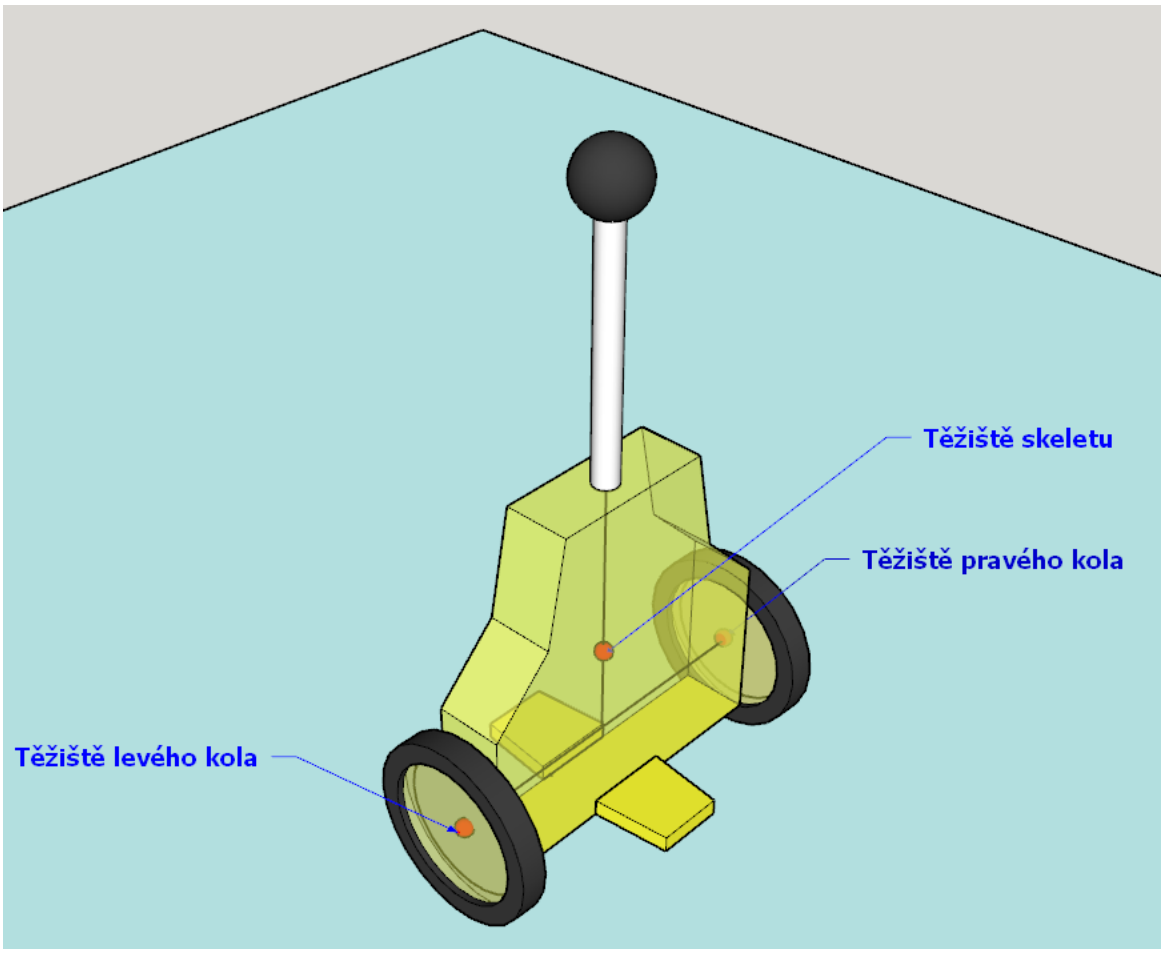
- Identifikujeme a popíšeme zobecněné síly F_r, popřípadě i disipační síly systému $D(\dot{q})$, tedy ztráty.
- Dosadíme Lagrangeovu funkci do Lagrangeovy rovnice II. druhu

$$\frac{d}{dt}\left(\frac{\partial}{\partial \dot{q}} L(q, \dot{q})\right) - \frac{\partial}{\partial q} L(q, \dot{q}) = -\frac{\partial}{\partial \dot{q}} D(\dot{q}) + F_r \tag{10}$$

- Takto získáme pohybové rovnice systému (popis dynamiky systému). Celkový počet těchto rovnic je roven počtu zobecněných souřadnic.

3.2 Matematický model robota Inteco

Samotný robot, tedy jeho hmotné části jsou při procesu matematického modelování charakterizovány hmotnostmi, těžišti a momenty setrvačností těchto částí viz obrázek 44.

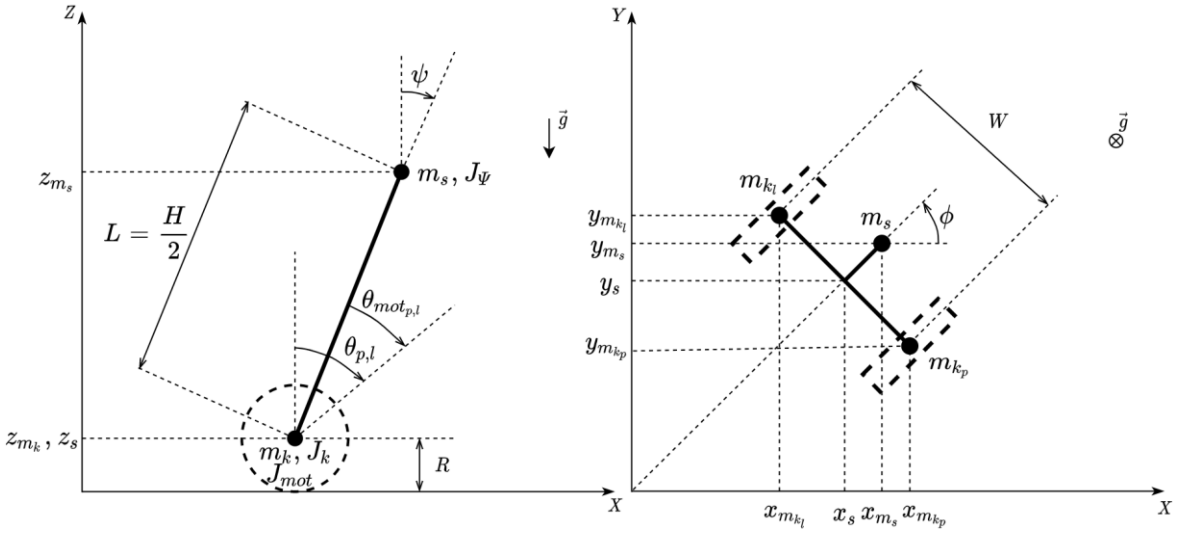


Obrázek 44 Těžiště jednotlivých mechanických částí robota

Zobecněné souřadnice byly zvoleny následovně:

- Souřadnice θ - úhlová poloha kol (pravé, levé), resp. úhlová rychlost jako derivace této souřadnice podle času
- Souřadnice ψ - odklon robota od vertikály (tzv. *pitch*)
- Souřadnice ϕ – úhel rotace robota okolo vertikály (tzv. *yaw*)

Samozřejmě existuje několik dalších možností výběru zobecněných souřadnic. Po prostudování dostupných zdrojů jsem zvolil právě výše popsané zobecněné souřadnice. Tento způsob volby je názorný, dobře pochopitelný a výrazně zjednodušuje výsledné rovnice, a to bez ztráty přesnosti modelu.



Obrázek 45 Popis hmotných částí robota v kartézském souřadném systému

Legenda k obrázku 45:

m_k, J_k… hmotnost a moment setrvačnosti kola robota

m_s, J_ψ…hmotnost a moment setrvačnosti skeletu robota

J_{mot}…moment setrvačnosti motoru a převodovky s ohledem na převodový poměr

z_{m_k}… vzdálenost těžiště kola od podložky

z_{m_s}…vzdálenost těžiště robota od podložky

R…poloměr kola

L…vzdálenost těžiště skeletu od osy otáčení kol

W…rozteč kol

g…tíhové zrychlení

$x_{m_{k_l}}, y_{m_{k_l}}$…souřadnice levého kola

$x_{m_{k_p}}, y_{m_{k_p}}$…souřadnice pravého kola

x_{m_s}, y_{m_s}…souřadnice skeletu robota

x_s, y_s…souřadnice středu robota, resp. vertikální osy souměrnosti

Nyní vyjádříme celkovou kinetickou a potenciální energii systému. Pomocí zobecněných souřadnic a na základě zvoleného souřadného systému viz obrázek 45, můžeme popsat hmoty (těžiště) charakterizující jednotlivé části robota. Vztahy mezi jednotlivými proměnnými lze zapsat následujícími rovnicemi:

$$\theta = \frac{1}{2}(\theta_l + \theta_r) \tag{11}$$

$$(\theta_r, \theta_l) = \left(\theta + \frac{W}{R}\phi, \theta - \frac{W}{2R}\phi\right) \tag{12}$$

$$\Phi = \frac{R}{W}(\theta_r - \theta_l) \tag{13}$$

$$\dot{x} = R\dot{\theta}\cos\phi \tag{14}$$

$$\dot{y} = R\dot{\theta}\sin\phi \tag{15}$$

$$(x_s, y_s, z_s) = R\,\theta\cos\phi\,, R\,\theta\sin\phi\,, R \tag{16}$$

$$\left(x_{m_s}, y_{m_s}, z_{m_s}\right) = (x_s + L\sin\psi\cos\phi\,, y_s + L\sin\psi\sin\phi\,, z_{m_k} + L\cos\psi) \tag{17}$$

$$\left(x_{m_{k_p}}, y_{m_{k_p}}, z_{m_k}\right) = \left(x_s + \frac{W}{2}\sin\phi\,, y_s - \frac{W}{2}\cos\phi\,, z_{m_k}\right) \tag{18}$$

$$\left(x_{m_{k_l}}, y_{m_{k_l}}, z_{m_k}\right) = \left(x_s - \frac{W}{2}\sin\phi\,, y_s + \frac{W}{2}\cos\phi\,, z_{m_k}\right) \tag{19}$$

Dále vyjádříme translační kinetickou energii T_t, rotační kinetickou energii T_r a potenciální energii U:

$$T_t = \frac{1}{2}m_k\left(\left(\dot{x}_{m_{k_p}}{}^2 + \dot{y}_{m_{k_p}}{}^2 + \dot{z}_{m_k}{}^2\right) + \left(\dot{x}_{m_{k_l}}{}^2 + \dot{y}_{m_{k_l}}{}^2 + \dot{z}_{m_k}{}^2\right)\right) + \frac{1}{2}m_s\left(\dot{x}_{m_s}{}^2 + \dot{y}_{m_s}{}^2 + \dot{z}_{m_s}{}^2\right) \tag{20}$$

$$T_r = \frac{1}{2}J_k\left(\left(\dot{\theta}_p{}^2\right) + \left(\dot{\theta}_l{}^2\right)\right) + \frac{1}{2}J_\psi\dot{\psi}^2 + \frac{1}{2}J_\phi\dot{\phi}^2 + \frac{1}{2}J_m\left(\left(\dot{\theta}_p - \dot{\psi}\right)^2 + \left(\dot{\theta}_l - \dot{\psi}\right)^2\right) \tag{21}$$

$$U = 2\left(m_k g z_{m_k}\right) + m_s g z_{m_s} \tag{22}$$

Lagrangeova funkce podle rovnice (9) bude mít tvar:

$$L = T_t + T_r - U \tag{23}$$

Lagrangeovy rovnice pro zobecněné síly podle rovnice (10), při zanedbání ztrát budou:

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\theta}}\right) - \frac{\partial L}{\partial \theta} = F_\theta \tag{24}$$

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\psi}}\right) - \frac{\partial L}{\partial \psi} = F_\psi \tag{25}$$

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\phi}}\right) - \frac{\partial L}{\partial \phi} = F_\phi \tag{26}$$

Derivací rovnic (24), (25), (26) dostáváme pohybové rovnice robota:

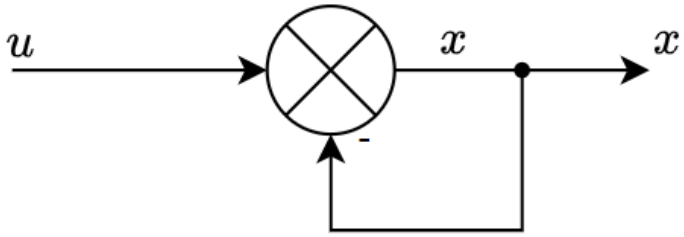
$$\left((2m_k + m_s)R^2 + 2J_k + 2J_{mot}\right)\ddot{\theta} + (m_s RL\cos\psi - 2J_{mot})\ddot{\psi} - m_s RL\psi^2\sin\psi = F_\theta \tag{27}$$

$$\left(m_s L^2 + J_\psi + 2J_{mot}\right)\ddot{\psi} + (m_s RL\cos\psi - 2J_{mot})\ddot{\theta} - m_s L^2\dot{\phi}^2\sin\psi\cos\psi - m_s gL\sin\psi = F_\psi \tag{28}$$

$$\left(\frac{1}{2}m_k W^2 + m_s L^2\sin^2\psi + \frac{W^2}{2R^2}(J_w + J_{mot}) + J_\phi\right)\ddot{\phi} + 2m_s L^2\dot{\phi}\dot{\psi}\sin\psi\cos\psi = F_\phi \tag{29}$$

V rovnicích (27), (28) se vyskytují derivace druhého řádu u dvou proměnných. Jelikož budeme matematický model implementovat do prostředí Simulink, je nutné tyto rovnice upravit. Důvodem je, že rovnice v tomto tvaru nelze řešit numericky z důvodu algebraické smyčky. Pokud rovnice obsahuje algebraickou smyčku, nelze vygenerovat model. Přestože prostředí Simulink obsahuje řešitele algebraické smyčky, není zaručeno, že bude schopen problém vyřešit a pokud ano, simulace bude probíhat výrazně pomaleji z důvodu vytížení

systému řešením algebraické smyčky, a to zejména pro vypočet prvního kroku. Jednoduchý příklad algebraické smyčky můžeme vidět na obrázku 46. Na první pohled je patrné, že výpočet hodnoty výstupu $x = u - x$ je závislý na informaci o jeho hodnotě a tím vzniká již zmíněná algebraická smyčka a výpočetní problém. [35]



Obrázek 46 Příklad jednoduché algebraické smyčky

K řešení tohoto problému aplikujeme na rovnice (27), (28) Cramerovo pravidlo a zavedeme substituci:

$$a\ddot{\theta} + b\ddot{\psi} = F_\theta + c \tag{30}$$

$$d\ddot{\psi} + b\ddot{\theta} = F_\psi + e \tag{31}$$

kde

$$a = (2m_k + m_s)R^2 + 2J_k + 2J_{mot} \tag{32}$$

$$b = m_s RL\cos\psi - 2J_{mot} \tag{33}$$

$$c = m_s RL\psi^2 \sin\psi \tag{34}$$

$$d = m_s L^2 + J_\psi + 2J_{mot} \tag{35}$$

$$e = m_s L^2 \dot{\phi}^2 \sin\psi\cos\psi + m_s gL\sin\psi \tag{36}$$

A po použití Cramerova pravidla dostáváme:

$$\ddot{\theta} = \frac{d(F_\theta + c) - b(F_\psi + e)}{ad - b^2} \tag{37}$$

$$\ddot{\psi} = \frac{a\left(F_\psi + e\right) - b(F_\theta + c)}{ad - b^2} \tag{38}$$

$$\ddot{\phi} = \frac{F_\phi - 2m_s L^2 \dot{\phi}\dot{\psi} \sin\psi \cos\psi}{\frac{1}{2}m_k W^2 + m_s L^2 \sin^2\psi + \frac{W^2}{2R^2}\left(J_k + J_{mot}\right) + J_\phi} \tag{39}$$

Dále je potřeba definovat zobecněné síly s přihlédnutím k použitému zdroji těchto sil. V našem případě se jedná o stejnosměrný motor. Stejnosměrný motor můžeme popsat obecně rovnicí (40)

$$u_d(t) - \omega k_\omega - R_k\, i_{vst}(t) - L_k \frac{d}{dt} i_{vst}(t) = 0 \tag{40}$$

kde

u_d…vstupní napětí motoru

i_{vst}…vstupní proud do motoru

R_k…ohmický odpor vinutí motoru

L_k…vlastní indukčnost vinutí motoru

k_ω…rychlostní konstanta motoru (velikost BEMF v závislosti na otáčkách rotoru)

ω…úhlová rychlost rotoru

Po odeznění přechodového děje můžeme zanedbat vlastní indukčnost motoru a rovnici (40) zapsat jako:

$$i_{vst} = \frac{u_{d_{p,l}} + \omega k_\omega}{R_k} \tag{41}$$

Vztahy pro zobecněné síly zapíšeme následovně:

$$\left(F_\theta, F_\psi, F_\phi\right) = \left(\frac{1}{2}\left(F_p + F_l\right), F_\psi, \frac{R}{W}\left(F_p - F_l\right)\right) \tag{42}$$

$$F_p = nk_t i_{vst_p} + f_m(\dot{\psi} - \dot{\theta}_p) - f_w\dot{\theta}_p \tag{43}$$

$$F_l = nk_t i_{vst_l} + f_m(\dot{\psi} - \dot{\theta}_l) - f_w\dot{\theta}_l \tag{44}$$

$$F_\psi = -nk_t i_{vst_l} - nk_t i_{vst_p} - f_m(\dot{\psi} - \dot{\theta}_l) - f_m(\dot{\psi} - \dot{\theta}_p) \tag{45}$$

Řízení motorů robota probíhá pomocí PWM modulace, což je způsob, pomocí kterého můžeme řídit vstupní napětí motorů. Dosazením rovnic (41), (43), (44), (45) do rovnice (42) dostaneme vztahy pro zobecněné síly řízené vstupními napětími motorů:

$$F_\theta = \frac{k_t}{R_m}\left(u_{d_p} + u_{d_l}\right) + 2\left(\frac{k_t k_\omega}{R_m} + f_m\right)(\dot{\psi} - \dot{\theta}) \tag{46}$$

$$F_\psi = -\frac{k_t}{R_m}\left(u_{d_p} + u_{d_l}\right) + 2\left(\frac{k_t k_\omega}{R_m} + f_m\right)(\dot{\psi} - \dot{\theta}) \tag{47}$$

$$F_\phi = -\frac{k_t}{R_m}\frac{W}{2R}\left(u_{d_p} - u_{d_l}\right) - \frac{W^2}{2R^2}\left(\frac{k_t k_\omega}{R_m} + f_m\right)\dot{\phi} \tag{48}$$

kde

$u_{d_{p,l}}$…vstupní napětí levého a pravého motoru

R_m…ohmický odpor vinutí motoru

k_t… elektromechanická transformační konstanta motoru

k_ω…rychlostní konstanta motoru (velikost BEMF v závislosti na otáčkách rotoru)

f_m… koeficient tření mezi robotem a motorem

Pro další práci s matematickým modelem shrneme dosavadní poznatky a označíme jednotlivé stavové a vstupní proměnné následovně:

$$\begin{bmatrix} x_1 = \theta = \dfrac{\theta_p + \theta_l}{2} \\ x_2 = \dot{\theta} \\ x_3 = \psi \\ x_4 = \dot{\psi} \\ x_5 = \phi \\ x_6 = \dot{\phi} \\ u_1 = u_p \\ u_2 = u_l \end{bmatrix} \tag{49}$$

Stavové rovnice systému potom můžeme zapsat jako:

$$\begin{aligned} \dot{\boldsymbol{x}}_{\mathbf{1}} &= x_2 \\ \dot{\boldsymbol{x}}_{\mathbf{2}} &= \frac{d\left(F_{x_1} + c\right) - b(F_{x_3} + e)}{ad - b^2} \\ \dot{\boldsymbol{x}}_{\mathbf{3}} &= x_4 \\ \dot{\boldsymbol{x}}_{\mathbf{4}} &= \frac{a\left(F_{x_3} + e\right) - b(F_{x_1} + c)}{ad - b^2} \\ \dot{\boldsymbol{x}}_{\mathbf{5}} &= x_6 \\ \dot{\boldsymbol{x}}_{\mathbf{6}} &= \frac{F_{x_5} - 2m_s L^2 \dot{\phi}\dot{\psi} \sin\psi \cos\psi}{\frac{1}{2} m_k W^2 + m_s L^2 \sin^2\psi + \frac{W^2}{2R^2}\left(J_k + J_{mot}\right) + J_\phi} \end{aligned} \tag{50}$$

kde

$a = (2m_k + m_s)R^2 + 2J_k + 2J_{mot}$

$b = m_s RL \cos\psi - 2J_{mot}$

$c = m_s RL\psi^2 \sin\psi$

$d = m_s L^2 + J_\psi + 2J_{mot}$

$e = m_s L^2 \dot{\phi}^2 \sin\psi \cos\psi + m_s gL \sin\psi$

$F_{x_1} = \frac{k_t}{R_m}\left(u_{d_p} + u_{d_l}\right) + 2\left(\frac{k_t k_\omega}{R_m} + f_m\right)(x_4 - x_2)$

$F_{x_3} = -\frac{k_t}{R_m}\left(u_{d_p} + u_{d_l}\right) + 2\left(\frac{k_t k_\omega}{R_m} + f_m\right)(x_4 - x_2)$

$F_{x_5} = -\frac{k_t}{R_m}\frac{W}{2R}\left(u_{d_p} - u_{d_l}\right) - \frac{W^2}{2R^2}\left(\frac{k_t k_\omega}{R_m} + f_m\right)x_6$

A vektor vstupů systému (což jsou vstupní napětí motorů):

$$u = [u_{d_p} \quad u_{d_l}] \tag{51}$$

Výsledkem výše popsaného postupu je nelineární matematický model daného dvoukolového robota Inteco. Tento model lze použít k návrhu nelineárního zákona řízení, nebo ho linearizovat v okolí zvoleného pracovního bodu a použít k návrhu lineárních metod řízení.

3.3 Linearizace matematického modelu robota Inteco

Linearizace bude provedena v pracovním bodě, který můžeme považovat za „*ekvilibrium*“. Tímto termínem můžeme označit stav, resp. polohu, kdy je robot dokonale vyvážen ve vzpřímené poloze. V takto idealizovaném případě se dynamika vnitřních stavů stává nulovou a stejně tak jsou nulové i vstupy systému. Tento stav vyjádříme pomocí vektoru stavů a vstupů (49) jako:

$$\boldsymbol{x_0} = [x_1 \quad x_2 \quad x_3 \quad x_4 \quad x_5 \quad x_6]^T = [0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0]^T \tag{52}$$

$$\boldsymbol{u_0} = [u_{d_p} \quad u_{d_l}] = [0\ 0] \tag{53}$$

Obecný nelineární matematický model ve vektorovém tvaru lze psát jako:

$$\begin{aligned} \frac{d\boldsymbol{x}}{dt} &= \boldsymbol{f}(t, \boldsymbol{x}, \boldsymbol{u}) \\ \boldsymbol{y} &= \boldsymbol{g}(t, \boldsymbol{x}, \boldsymbol{u}) \end{aligned} \tag{54}$$

kde

$\boldsymbol{u}$…vektor vstupních veličin

$\boldsymbol{y}$… vektor výstupních veličin

$\boldsymbol{x}$… vektor stavových veličin

$\boldsymbol{f}, \boldsymbol{g}$…obecně nelineární vektorové funkce

Linearizace v definovaném pracovním bodě x_0, u_0, viz (52), (53) je potom popsána takto:

$$\begin{aligned}\frac{d\boldsymbol{x}}{dt} &= \boldsymbol{f}(\boldsymbol{x_0}, \boldsymbol{u_0}) + \left.\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}}\right|_{x_0,u_0} (\boldsymbol{x} - \boldsymbol{x_0}) + \left.\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{u}}\right|_{x_0,u_0} (\boldsymbol{u} - \boldsymbol{u_0}) = \\ &= \boldsymbol{f}(\boldsymbol{x_0}, \boldsymbol{u_0}) + \boldsymbol{A}(\boldsymbol{x} - \boldsymbol{x_0}) + \boldsymbol{B}(\boldsymbol{u} - \boldsymbol{u_0})\end{aligned} \tag{55}$$

$$\begin{aligned}\boldsymbol{y} &= \boldsymbol{g}(\boldsymbol{x_0}, \boldsymbol{u_0}) + \left.\frac{\partial \boldsymbol{g}}{\partial \boldsymbol{x}}\right|_{x_0,u_0} (\boldsymbol{x} - \boldsymbol{x_0}) + \left.\frac{\partial \boldsymbol{g}}{\partial \boldsymbol{u}}\right|_{x_0,u_0} (\boldsymbol{u} - \boldsymbol{u_0}) = \\ &= \boldsymbol{g}(\boldsymbol{x_0}, \boldsymbol{u_0}) + \boldsymbol{C}(\boldsymbol{x} - \boldsymbol{x_0}) + \boldsymbol{D}(\boldsymbol{u} - \boldsymbol{u_0})\end{aligned} \tag{56}$$

Absolutní člen v rovnici (55) bude nulový při splnění podmínky, že zvolený pracovní bod odpovídá tzv. ustálenému stavu systému („*ekvilibriu*“). Absolutní člen rovnice (56) lze převést na levou stranu rovnice a dále pracovat s odchylkou výstupu od hodnoty $\boldsymbol{y_0} = \boldsymbol{y}(0) = \boldsymbol{g}(\boldsymbol{x_0}, \boldsymbol{u_0})$. Je-li tedy linearizace provedena v pracovním bodě, kdy je systém v ustáleném stavu je výsledkem obecný linearizovaný stavový model v odchylkovém tvaru:

$$\begin{aligned}\frac{d\boldsymbol{x}}{dt} &= \boldsymbol{A}\Delta\boldsymbol{x} + \boldsymbol{B}\Delta\boldsymbol{u} \\ \Delta\boldsymbol{y} &= \boldsymbol{C}\Delta\boldsymbol{x} + \boldsymbol{D}\Delta\boldsymbol{u}\end{aligned} \tag{57}$$

kde

$$\begin{aligned}\Delta\boldsymbol{x} &= (\boldsymbol{x} - \boldsymbol{x_0}) \\ \Delta\boldsymbol{u} &= (\boldsymbol{u} - \boldsymbol{u_0}) \\ \Delta\boldsymbol{y} &= \boldsymbol{y} - \boldsymbol{g}(\boldsymbol{x_0}, \boldsymbol{u_0}) = \boldsymbol{y} - \boldsymbol{y_0}\end{aligned} \tag{58}$$

jsou odchylky stavů, vstupů a výstupů systému a matice $\boldsymbol{A}$, $\boldsymbol{B}$, $\boldsymbol{C}$ a $\boldsymbol{D}$ jsou jacobiány příslušných vektorových funkcí. V tomto případě budou však matice $\boldsymbol{C}$ a $\boldsymbol{D}$ jednoduššího tvaru, konkrétně $\boldsymbol{C}$ bude jednotková matice (stavy systému jsou zároveň i výstupy) a $\boldsymbol{D}$ bude nulová matice.

K samotnému výpočtu lze použít například funkci *linmod* v systému Matlab. Po vyčíslení rovnic (55) a (56) v definovaném pracovním bodě a dosazením parametrů robota z tabulky 1 dostaneme linearizovaný matematický model robota Inteco jako:

$$\dot{\boldsymbol{x}}(t) = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & -1{,}094 & -29{,}14 & 0{,}4733 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0{,}248 & 29{,}16 & -0{,}1401 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & -2{,}635 \end{bmatrix} \Delta\boldsymbol{x}(t) + \begin{bmatrix} 0 & 0 \\ 109{,}4 & 109{,}4 \\ 0 & 0 \\ -32{,}37 & -32{,}37 \\ 0 & 0 \\ -89{,}87 & 89{,}87 \end{bmatrix} \Delta\boldsymbol{u}(t)$$

$$\Delta\boldsymbol{y} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \Delta\boldsymbol{x}(t) \tag{59}$$

Tabulka 1 Parametry robota Inteco

Parametr	**Jednotky**	**Popis**
$m_k = 0{,}32$	kg	Váha kola
$D_k = 0{,}15$	m	Průměr kola
$J_k = mR^2$	kgm^2	Moment setrvačnosti kola
$J_k = 0{,}0013$	kgm^2	Identifikovaný moment setrvačnosti kola
$m_s = 5{,}41$	kg	Váha skeletu robota
$W = 0{,}4$	m	Rozteč kol robota
$L = 0{,}102$	m	Vzdálenost těžiště skeletu od osy otáčení kol
$J_\psi = (ML^2)/3$	kgm^2	Odhadnutý moment setrvačnosti náklonu skeletu
$J_\psi = 0{,}104$	kgm^2	Identifikovaný moment setrvačnosti náklonu skeletu
$J_\varphi = 0{,}0484$	kgm^2	Moment setrvačnosti robota vůči ose rotace kol
$J_{mot} = 0{,}00119$	kgm^2	Identifikovaný moment setrvačnosti motoru a převodovky s ohledem na převodový poměr
$R_m = 1$	Ω	Ohmický odpor vinutí stejnosměrného motoru
$k_t = 0{,}025$	Nm/A	Elektromechanická transformační konstanta motoru
$k_\omega = 0{,}025$	Vs/rad	BEMF (rychlostní) konstanta motoru
$f_m = 0{,}00024$	-	Identifikovaný koeficient tření mezi robotem a motorem

Podklady k vypracování kapitoly 3 byly čerpány ze zdrojů [2], [3], [4], [7], [35], [36], [37] [38].

Parametry uvedené v tabulce 1 byly převzaty od výrobce robota Inteco [3].

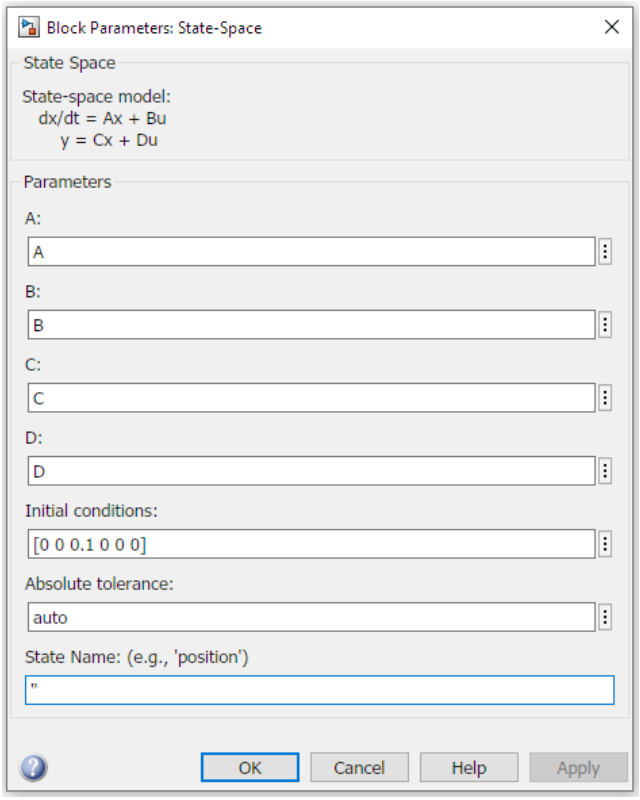
II. PRAKTICKÁ ČÁST

4 IMPLEMENTACE MATEMATICKÉHO MODELU ROBOTA INTECO DO PROSTŘEDÍ MATLAB-SIMULINK

Tato kapitola se bude věnovat implementaci vytvořených matematických modelů robota Inteco do prostředí Matlab Simulink. Závěr kapitoly popisuje proces ověření platnosti matematického modelu.

4.1 Linearizovaný model

Implementace linearizovaného modelu do prostředí Simulink je poměrně jednoduchá. K tomuto účelu lze použít funkční blok z knihovny pro práci se spojitými systémy s názvem *State-Space*. Jedná se o funkční blok, kterým lze realizovat stavový popis lineárního systému. Vstupní matice byly vytvořeny pomocí skriptu (viz příloha P I).

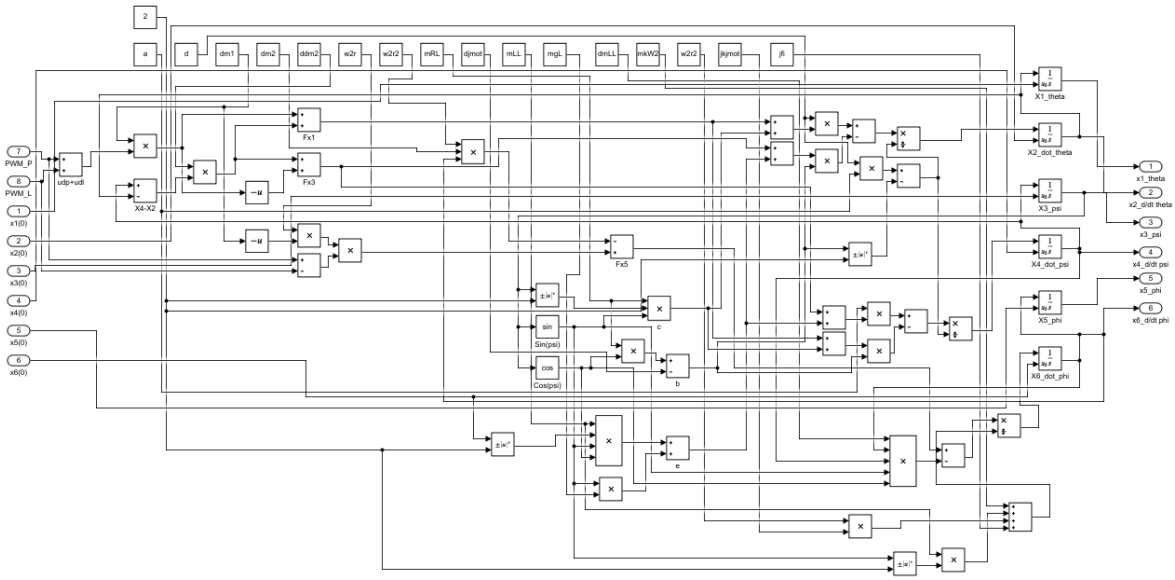


Obrázek 47 Vstupy funkčního bloku *State Space*

Jako vstupy použijeme linearizované rovnice (59). Dále jako počáteční podmínky zvolíme např. blízké okolí bodu linearizace. Vstupem do bloku je vektor vstupů $\boldsymbol{u}$ (53) a výstupem je v našem případě vektor stavů $\boldsymbol{x}$ (52).

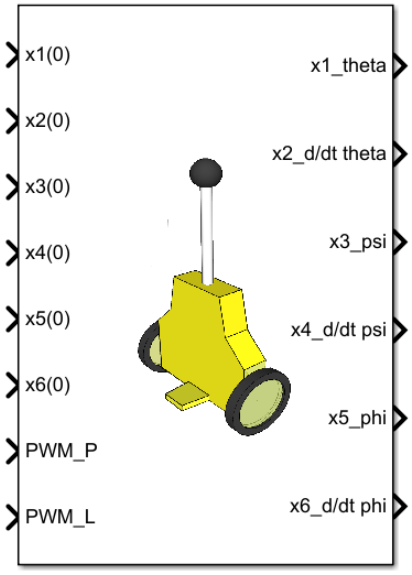
4.2 Nelineární model

Implementace nelineárního matematického modelu je komplikovanější než u lineárního modelu. Do prostředí Simulink je potřeba implementovat rovnice (49), (50), (51). K tomuto účelu je možné použít několika prostředků, které prostředí Simulink nabízí. První možností je použití funkčního bloku *S-function*, který interpretuje funkci (množinu rovnic) popisující vztahy mezi vstupy, stavy a výstupy systému. Funkce může být zapsána v jazyce Matlab, C, C++ nebo Fortran. Osobně jsem zvolil druhou možnost a tou je implementace rovnic do Simulinku pomocí standardních bloků jako jsou konstanta, matematické funkce, integrátor atd. Tuto možnost považuji za názornější, ale méně přehlednou. Určité zjednodušení může přinést zápis často se opakujících výpočtů a částí rovnic do Matlab skriptu (viz příloha P I) nebo použití sub-systémů. Takto připravené části rovnic pak lze již jednoduše (po spuštění skriptu) využít v prostředí Simulink.



Obrázek 48 Implementace nelineárního modelu do prostředí Simulink

Celé schéma, prezentované na obrázku 48, potom bylo převedeno jako subsystém do výsledného funkčního bloku nelineárního matematického modelu robota Inteco viz obrázek 49 níže. Jako vstupy bloku zde figurují počáteční podmínky a vstupní napětí motorů. Výstupy jsou shodné se stavy systému viz rovnice (49) resp. obrázek 45.



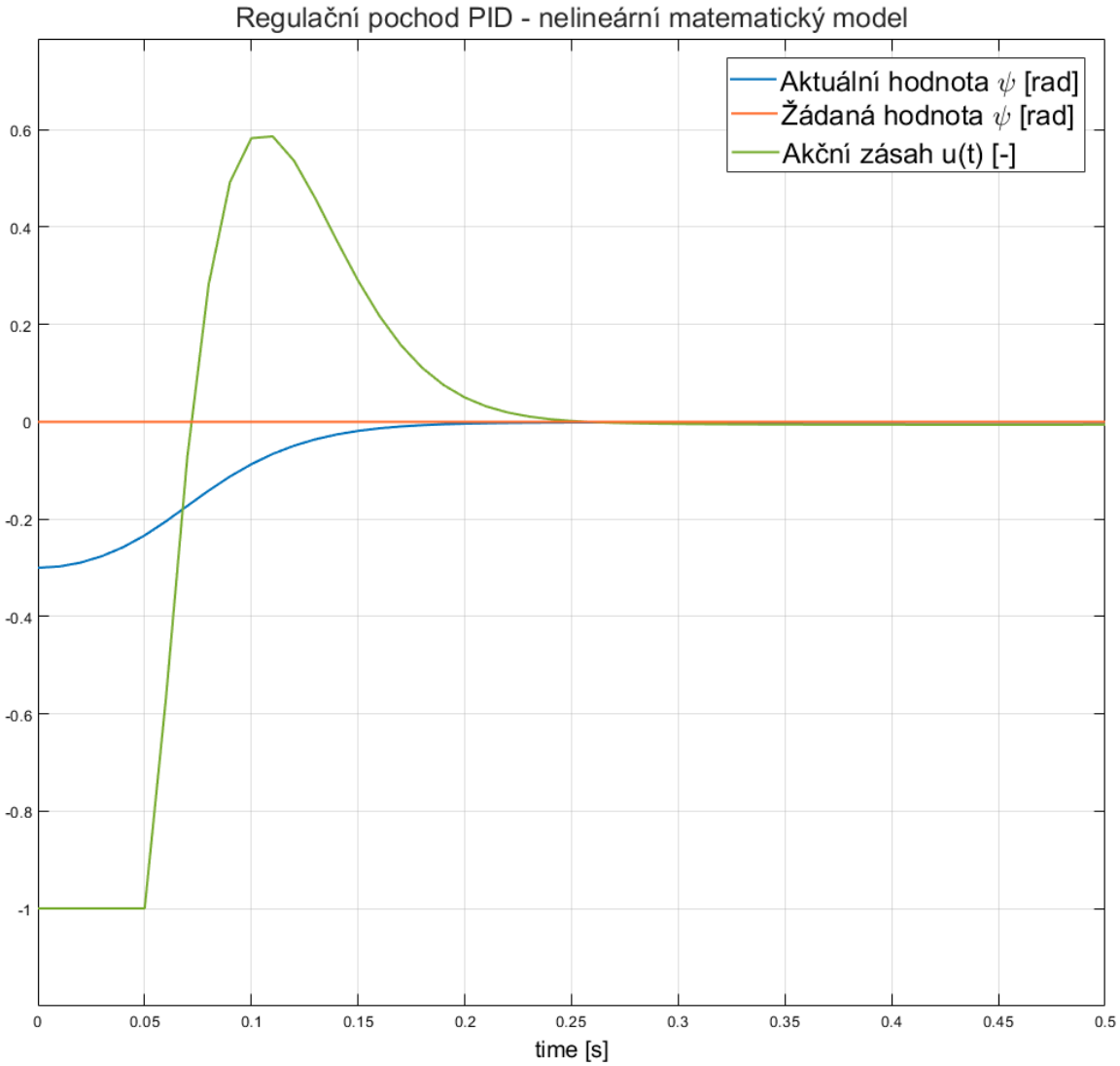
Obrázek 49 Nelineární model – funkční blok

4.3 Ověření simulačního modelu

Ověření simulačního modelu bylo provedeno porovnáním odezvy reálného systému a matematického modelu na stejný podmět. Jako podmínky tohoto ověření jsem zvolil reakci systému na regulační zásah PID regulátorem, navrženým na základě matematického modelu při daných počátečních podmínkách:

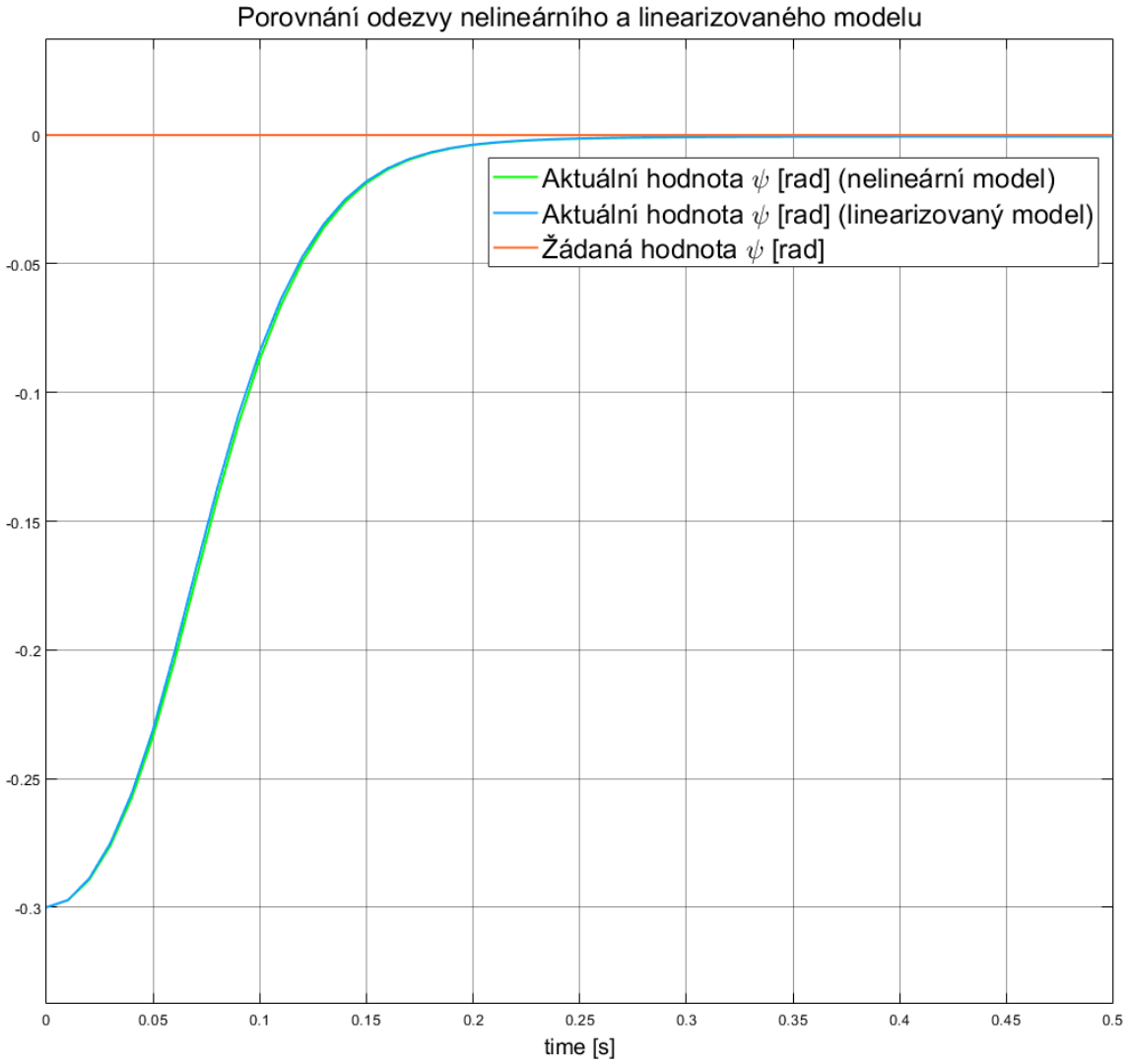
$$\begin{aligned} x_1 &= 0 \\ x_2 &= 0 \\ x_3 &= -0{,}1 \\ x_4 &= 0 \\ x_5 &= 0 \\ x_6 &= 0 \end{aligned} \tag{60}$$

Návrh regulátoru je detailně popsán v kapitole 5.1. Regulační pochod pro matematický model je na obrázku 50.



Obrázek 50 Regulační pochod PID – nelineární matematický model

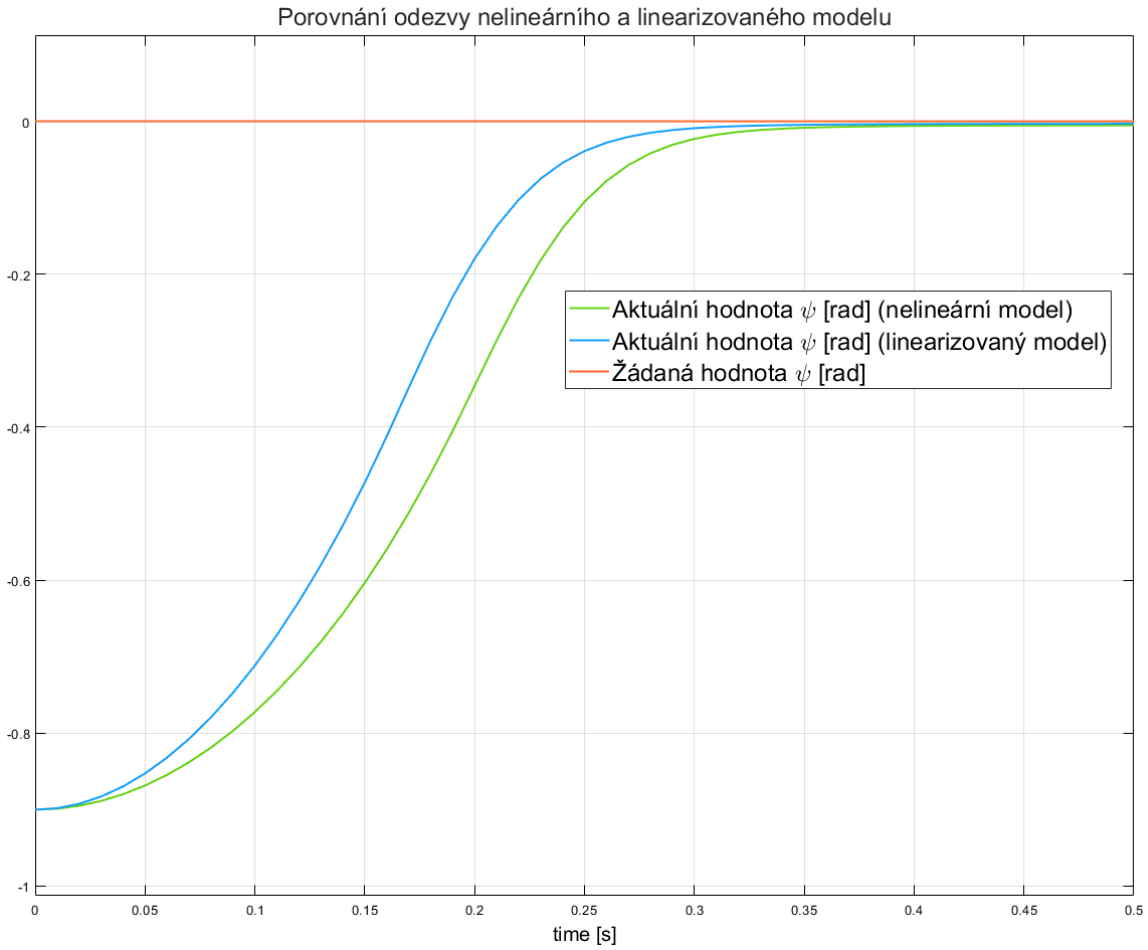
Dále bylo provedeno porovnání odezvy nelineárního a linearizovaného modelu na akční zásah regulátoru. Na obrázku 51 můžeme vidět, že odezva modelů je téměř totožná. Rozdíly v odezvě modelů se začnou projevovat, pokud se budou počáteční podmínky více lišit od pracovního bodu, ve kterém byl model linearizován (52).



Obrázek 51 Porovnání odezvy nelineárního a linearizovaného matematického modelu

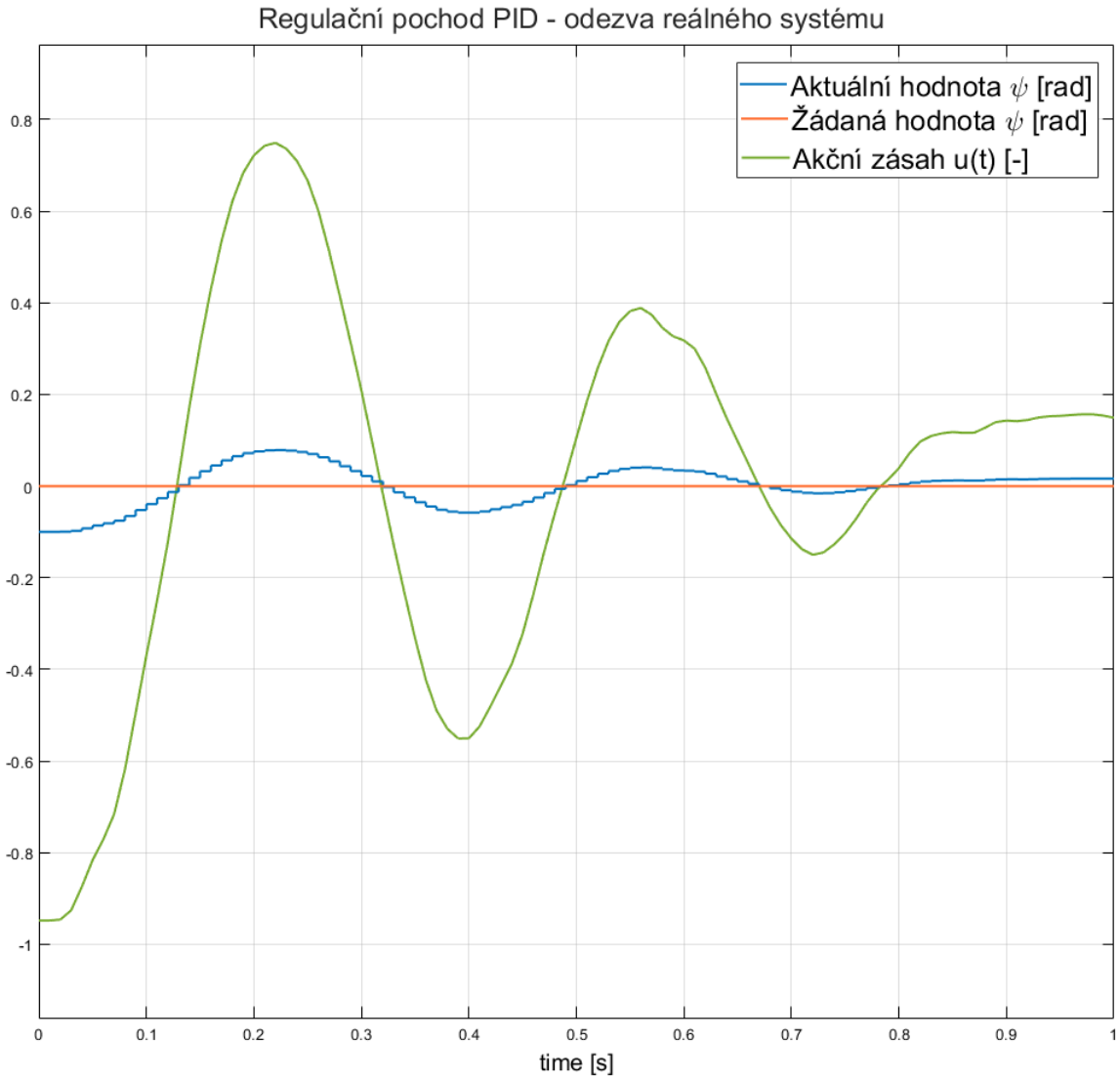
Příklad takového chování můžeme vidět na obrázku 52. V simulaci byly použity počáteční podmínky:

$$\begin{aligned} x_1 &= 0 \\ x_2 &= 0 \\ x_3 &= -0{,}9 \\ x_4 &= 0 \\ x_5 &= 0 \\ x_6 &= 0 \end{aligned} \tag{61}$$



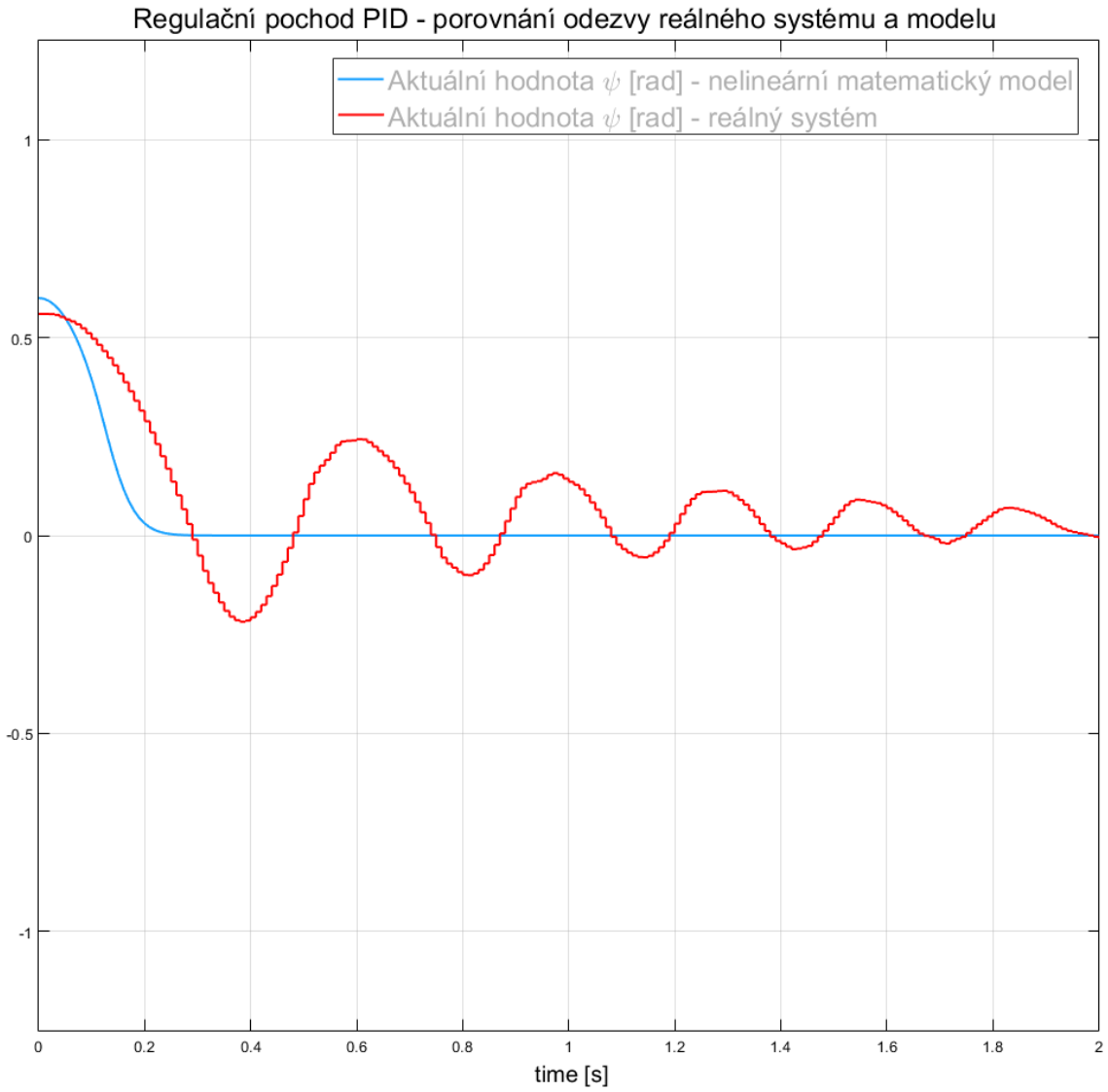
Obrázek 52 Porovnání odezvy matematických modelů pro větší odchylku z pracovního bodu

Dále byl proveden experiment, při kterém byl navržený PID regulátor použit k řízení reálného systému. Výsledky experimentu můžeme vidět na obrázku 53.



Obrázek 53 Regulační pochod reálného systému

Porovnáním výsledků provedených experimentů s výsledky simulace lze usoudit, že odvozené matematické modely popisují reálný systém dostatečně přesně pro účely návrhu zákona řízení. Jako důkaz tohoto tvrzení lze uvézt fakt, že PID regulátor navržený přístupem „*Model-Based Design*“ (tj. nepřímý návrh regulátoru na základě modelu systému) velmi dobře řídil reálný systém bez nutnosti úprav parametrů regulátoru. Pro lepší představu o přesnosti matematického modelu porovnáme pouze průběh regulované veličiny viz obrázek 54. Je patrné, že model nepopisuje dokonale dynamiku reálného systému, což můžeme přisoudit nedostatečně přesně identifikovaným parametrům systému a zanedbaní vnějších vlivů, které mají na chování reálného systému významný vliv. V průběhu práce s robotem jsem jako nejvíce kritický parametr shledal povrch, po kterém se robot pohybuje, resp. velikost koeficientu smykového tření mezi kolem a podložkou.



Obrázek 54 Porovnání odezvy reálného systému a matematického modelu

5 NÁVRH ZPŮSOBŮ ŘÍZENÍ SYSTÉMU

V této kapitole bude popsán návrh několika regulátorů určených pro řízení robota Inteco. Návrh obecně probíhal obecným přístupem *Model-Based design* (MBD) za pomoci systému Matlab a jeho toolboxů. Přístup MBD využívá matematický model systému již od počátku návrhu zákona řízení. Pomocí simulací lze systém analyzovat, navrhnout vhodné způsoby řízení a výsledný regulační obvod bezpečně otestovat. Dále tato metoda poskytuje možnost jednoduché implementace na cílovou platformu. Správná implementace řídicích funkcí je potom ověřena pomocí tzv. *software-in-the-loop* a *processor-in-the-loop* simulací.

5.1 PID regulátor

Jako první regulátor jsem zvolil notoricky známý PID regulátor, a to zejména pro jeho jednoduchost. Regulátor PID je tvořen třemi složkami:

- **Proporcionální složka** zesiluje regulační odchylku. Příliš vysoká hodnota může způsobit rozkmitání systému což může vést k nestabilnímu chování regulovaného systému.
- **Integrační složka** integruje regulační odchylku. Pomocí této složky lze dosáhnout nulové regulační odchylky. Příliš vysoká hodnota však může způsobit překmity regulované veličiny.
- **Derivační složka** umožňuje regulátoru vyhodnotit a reagovat na rychlost změny regulační odchylky. Při správném nastavení začne regulátor reagovat na změny regulační odchylky s určitým předstihem. Toto chování je žádoucí především u rychlých soustav s malými časovými konstantami. Velikost této složky je však nutno volit opatrně s přihlédnutím na skutečnost, že zesiluje šum.

Systém Simulink obsahuje přímo funkční blok PID regulátoru ve tvaru:

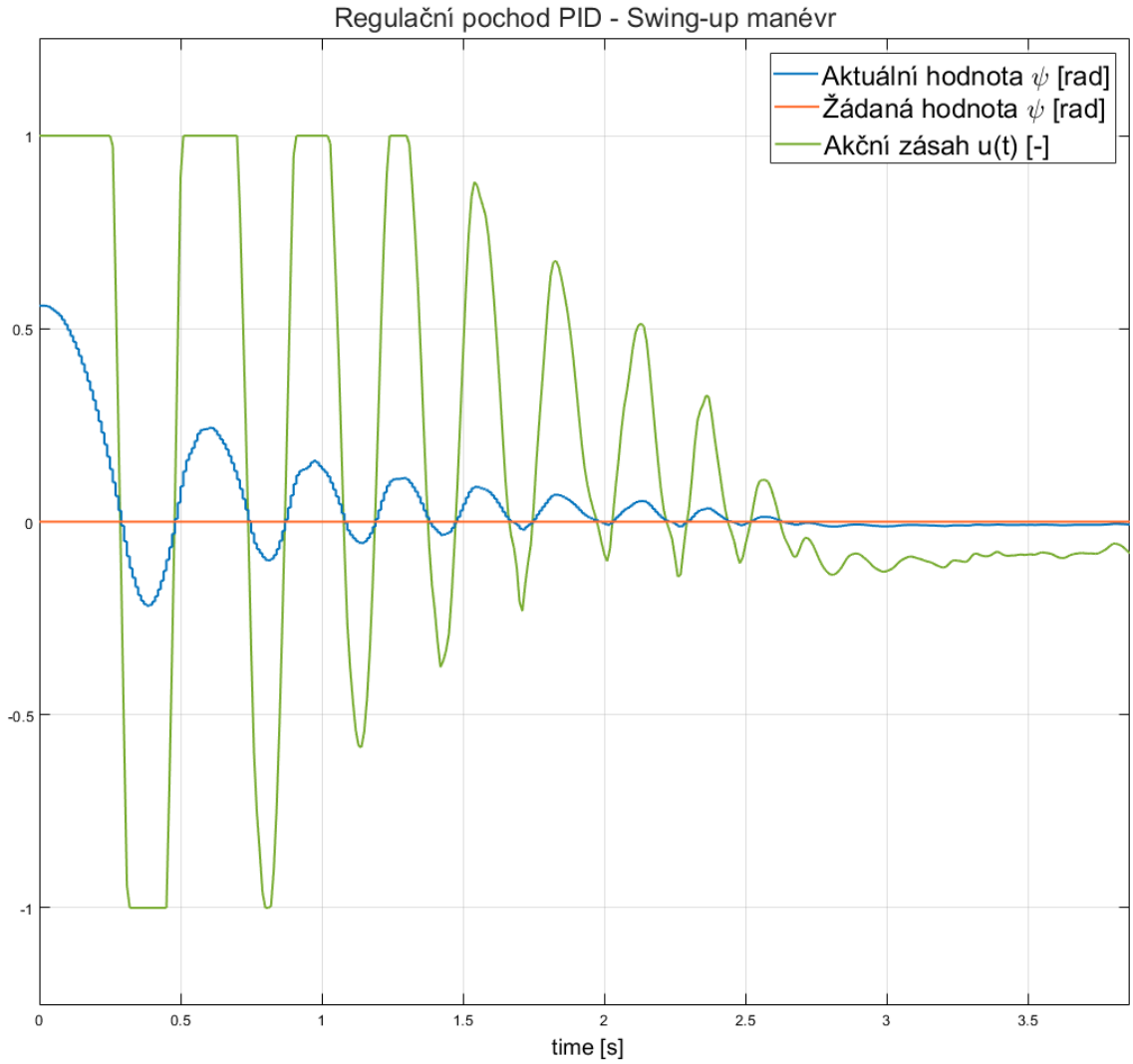
$$u(t) = P + I\frac{1}{s} + D\frac{N}{1 + N\frac{1}{s}} \tag{62}$$

kde P, I, D jsou jednotlivé složky a N je koeficient ovlivňující filtraci derivační složky. Nastavení PID regulátoru je poměrně intuitivní postup a díky možnosti simulace na matematickém modelu nehrozí nebezpečí způsobené nežádoucím chováním reálného

systému. Při návrhu regulátoru jsem použil heuristický postup velmi podobný metodě Ziegler – Nichols, na jejímž základě jsem navrhnul přibližné nastavení regulátoru. Doladění do finální podoby jsem potom provedl na základě vlastních zkušeností a ověřováním pomocí simulace. Jednotlivé složky PID regulátoru jsem nakonec odladil následovně:

$$\begin{aligned} P &= 9{,}5 \\ I &= 0.01 \\ D &= 0.53 \end{aligned} \tag{63}$$

Chování systému řízeného tímto regulátorem již bylo představeno v kapitole 4.3. Regulátor je však schopen nejen stabilizovat robota při malých úhlech vychýlení, ale dokáže jej stabilizovat i z počáteční klidové polohy a provést tzv. swing-up manévr, tj. uvést robota z klidové do vzpřímené polohy. Při tomto manévru se naplno projeví dynamika celého systému. Proto jsem experiment provedl několikrát pro obě možné počáteční klidové polohy robota.

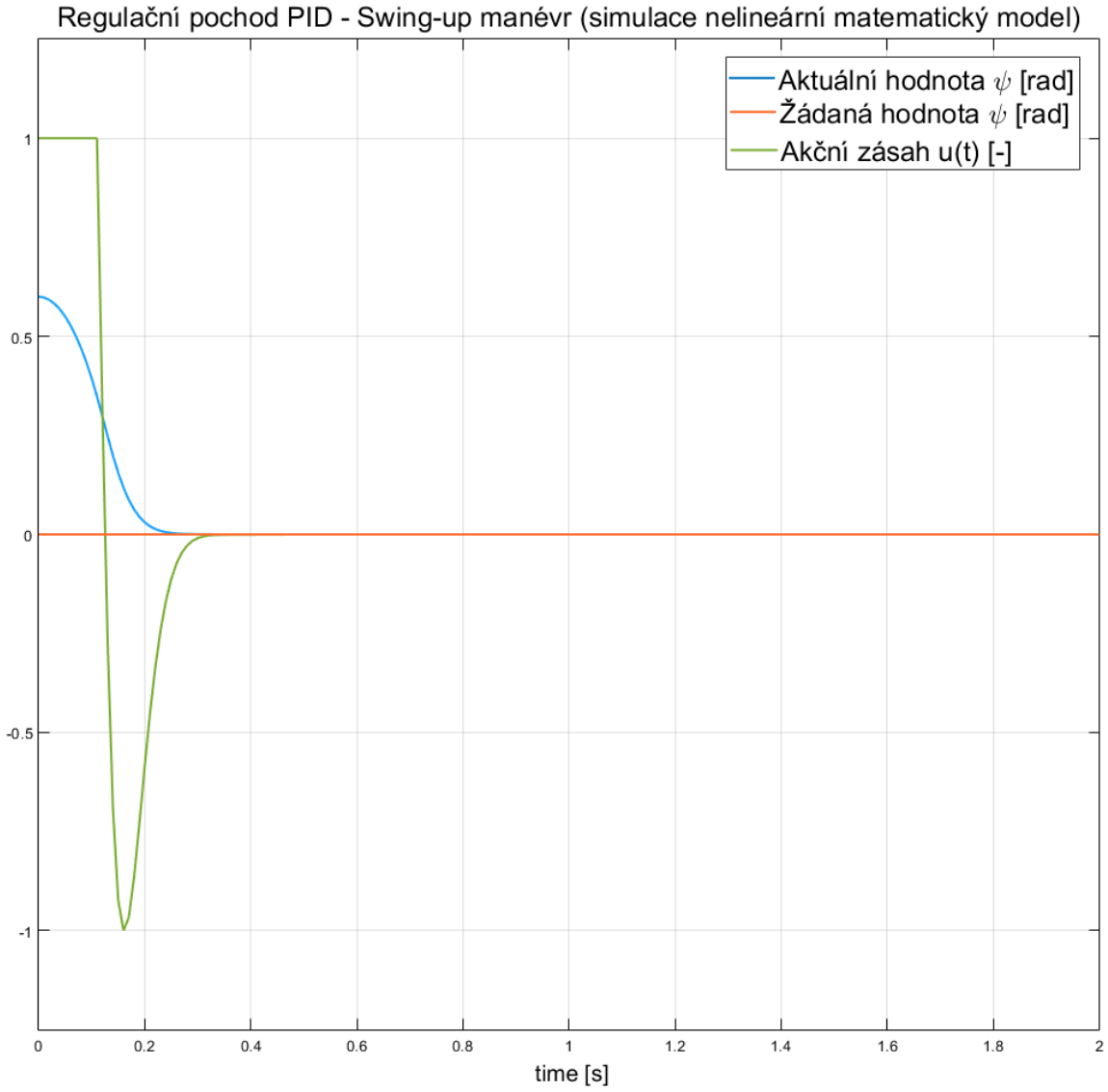


Obrázek 55 Swing-up manévr PID

Z průběhu na obrázku 55 lze vyvodit několik závěrů. Jako první si můžeme všimnout překmitů žádané hodnoty. Takové chování regulačního obvodu většinou indikuje špatné nastavení jednotlivých složek PID regulátoru. V tomto případě by bylo vhodné snížit hodnotu integrační složky, a naopak zvýšit hodnotu složky derivační. To bohužel nebylo možné z důvodu rušení, které proniká do signálu z IMU jednotky. Při zvýšení hodnoty derivační složky se regulační obvod rozkmitával i při použití filtrace. Toto chování ještě umocňuje fakt, že torzní tuhost celého modelu není dostatečná a do systému tak vstupují další poruchy a nelinearity způsobené pružností skeletu modelu. Dále je z průběhu regulované veličiny patrná asymetrie vůči ose x. Tento jev jsem po několika experimentech diagnostikoval jako nedokonalé rozložení hmotnosti robota vůči osám symetrie. Toto potvrzuje i fakt, že pokud byla žádaná hodnota náklonu nastavena na nulovou hodnotu,

měl robot stále tendenci pomalu popojíždět. Toto chování jsem potlačil nastavením žádané hodnoty na nenulovou hodnotu v řádech setin radiánu.

Pro porovnání na můžeme vidět stejnou situaci simulovanou na nelineárním matematickém modelu.



Obrázek 56 Simulace Swing-up manévru

5.2 LQ regulátor

Jedná se zde o stavovou regulaci. Návrh vychází ze stavového popisu systému a výsledný regulátor určuje akční zásah na základě aktuálních stavů systému. Pro získání odhadu stavů systému se zde opět používá Kalmanův filtr. V našem případě se jedná o úlohu optimálního řízení statickým stavovým regulátorem na nekonečném časovém horizontu

[36]. Uvedené řešení je platné pro časově invariantní systémy popsané stavovým popisem (57), kdy matice systému $\boldsymbol{A}$, matice řízení $\boldsymbol{B}$, matice vazeb výstupu na stav $\boldsymbol{C}$ jsou konstantní. Účelová funkce je potom obecně ve tvaru:

$$J = \frac{1}{2}\int_0^{\infty} [\boldsymbol{x}^T(t)\,\boldsymbol{M}\boldsymbol{x}(t) + \boldsymbol{u}^T(t)\,\boldsymbol{R}\boldsymbol{u}(t)]dt \tag{64}$$

kde $\boldsymbol{M}$ a $\boldsymbol{R}$ jsou konstantní váhové matice, pomocí kterých lze ovlivňovat chování regulátoru a konstantní tedy budou i prvky výsledné matice $\boldsymbol{K}$, která je určena řešením algebraické maticové rovnice Riccatiho typu:

$$-\boldsymbol{M} - \boldsymbol{A}^{\boldsymbol{T}}\boldsymbol{K} - \boldsymbol{K}\boldsymbol{A} + \boldsymbol{K}\boldsymbol{S}\boldsymbol{K} = 0 \tag{65}$$

kde

$$\boldsymbol{S} = \boldsymbol{B}\boldsymbol{R}^{-\boldsymbol{1}}\boldsymbol{B}^{\boldsymbol{T}} \tag{66}$$

a výsledný zákon řízení je potom dán vztahem:

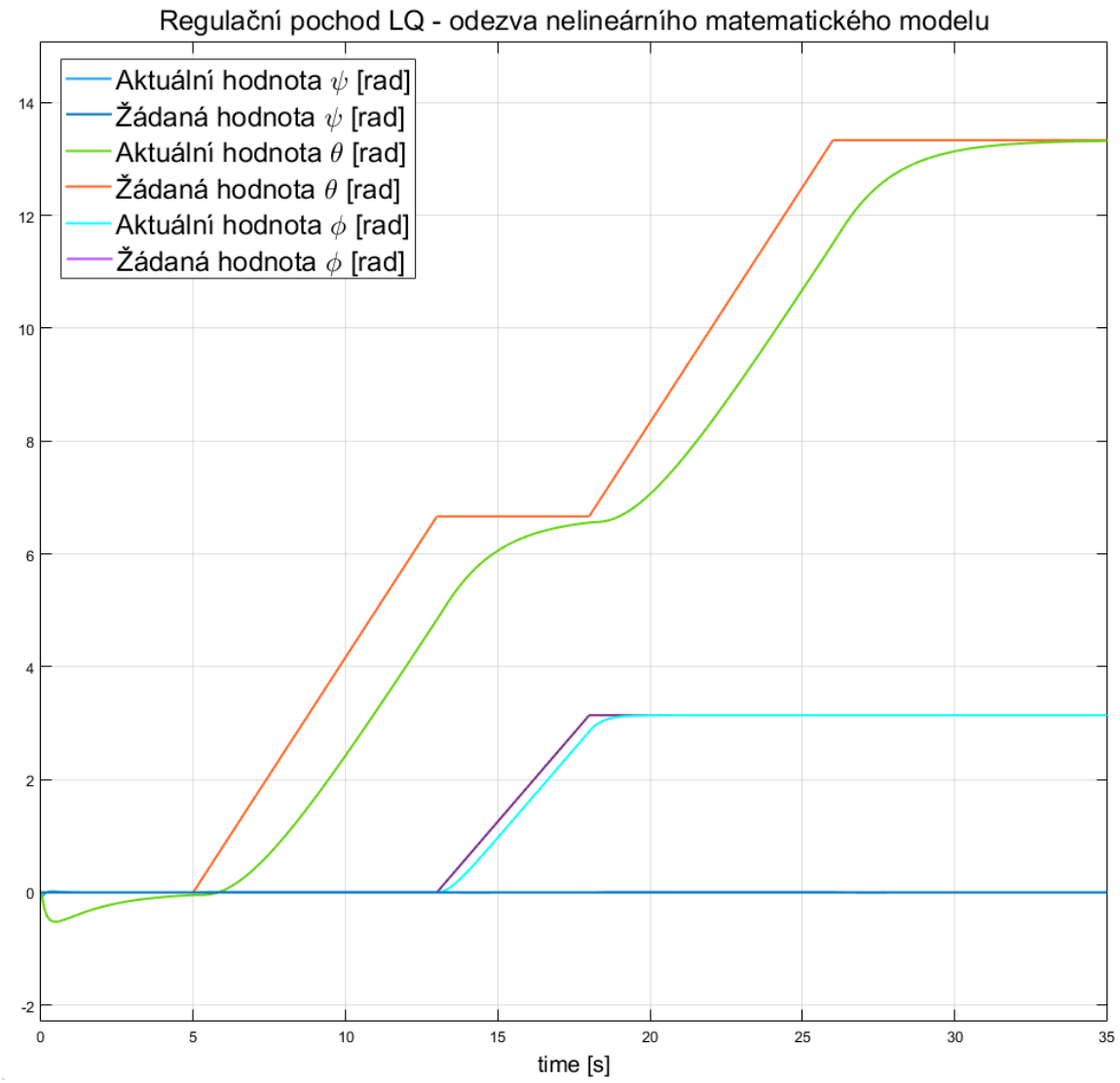
$$\boldsymbol{u}(t) = -\boldsymbol{R}^{-\boldsymbol{1}}\boldsymbol{B}^{\boldsymbol{T}}\boldsymbol{K}\boldsymbol{x}(t) \tag{67}$$

Samotnou implementaci LQ řízení jsem provedl pomocí systému Matlab a jeho knihovny *Control System Toolbox*. K tomuto účelu lze použít např. funkci *lqr*, kde vstupní parametry funkce jsou:

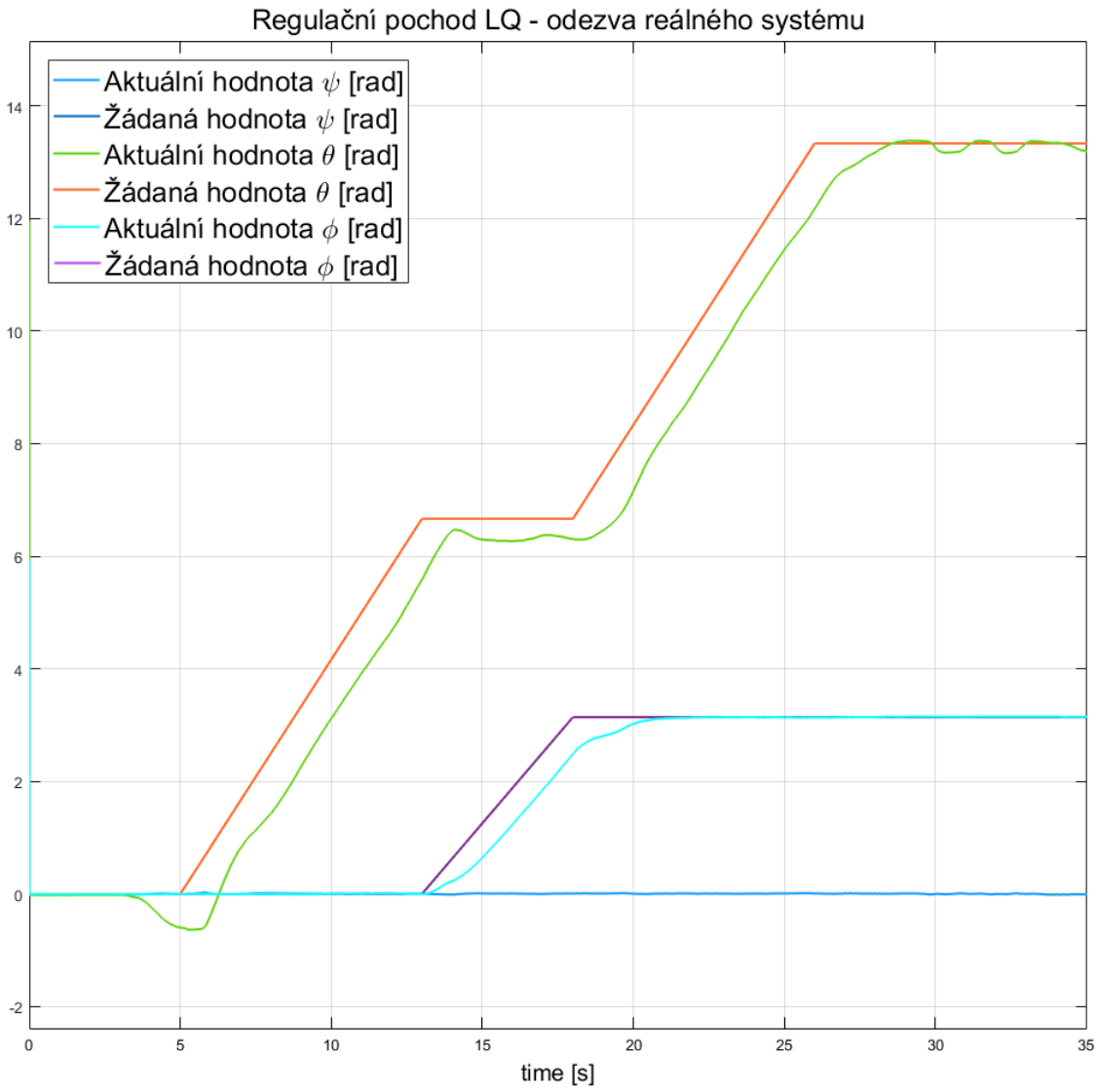
- Stavový popis LTI systému
- Váhová matice stavů $\boldsymbol{M}$
- Váhová matice vstupů $\boldsymbol{R}$

Výstupem uevdené funkce je jednak řešení Riccatiho rovnice $\boldsymbol{K}$, výsledný stavový regulátor a také póly takto navržené zpětné vazby. Složitější částí návrhu je volba vhodných váhových matic $\boldsymbol{M}$ a $\boldsymbol{R}$. V počátcích návrhu je možné tyto matice volit jako jednotkové. Matice $\boldsymbol{M}$ určuje váhy jednotlivých stavů, což znamená, že čím vyšší váhu určité proměnné nastavíme tím větší důraz je kladen na minimalizaci daného stavu při výpočtu účelové funkce (64). Jednoduše řečeno, pokud požadujeme, aby regulátor co

nejrychleji dosáhl žádanou hodnotu stavu a co nejlépe sledoval žádaný průběh tohoto stavu nastavíme příslušnou váhu v matici ***M*** na vysokou hodnotu. Je však nutné si uvědomit, že se stále jedná o optimalizační problém, a tedy vysoká priorita (váha) jednoho stavu znamená nižší prioritu ostatních stavů systému. Tento problém se naplno projeví, pokud se snažíme řídit stavy, které jsou navzájem závislé. V případě robota Inteco se jedná o polohu a úhel náklonu robota. K tomu, aby se robot udržel ve vzpřímené poloze je totiž potřeba změna jeho pozice. Je tedy nutné určit priority pro daný systém a najít vhodný kompromis pro nastavení jednotlivých prvků váhové matice ***M***. V případě robota Inteco je prioritou stabilizace robota ve vzpřímené pozici. Požadavkem však je i řízení robota v prostoru. Váhová matice ***M*** tedy byla nastavena s přihlédnutím k těmto požadavkům. Matice ***R*** potom omezuje nároky na akční zásah za cenu pomalejší regulace. Čím vyšší jsou hodnoty prvků matice ***R***, tím více je akční zásah omezen. Vliv této váhové matice se nejvíce projevil při řízení reálného systému, kdy vysoká hodnota akčního zásahu většinou vedla k prokluzu kol robota a ztrátě stability celého systému. Samotný návrh v podobě Matlab skriptu včetně hodnot váhových matic je součástí této práce jako příloha P I. Regulátor jsem opět navrhoval přístupem MBD a dosažené výsledky nejdříve ověřoval simulačně. Simulovaný regulační pochod je zobrazen na obrázku 57. Průběh reálné regulace pomocí LQ regulátoru je zobrazen na obrázku 58. Po porovnání obou průběhů lze konstatovat, že regulace na reálném systému probíhala dle očekávání. Dále se opět potvrdila vhodnost vytvořeného matematického modelu systému. Pro upřesnění – experiment spočíval ve stabilizaci robota a jeho následném přemístění o danou vzdálenost vyjádřenou úhlovou polohou kol v radiánech. Po uražení dané vzdálenosti se robot otočil o 180° a přesunul se zpět o stejnou vzdálenost na startovní pozici. Shoda mezi startovní a návratovou pozicí se pohybovala v jednotkách centimetrů. Experimentálně jsem však ověřil, že se tato odchylka zvětšuje úměrně s uraženou vzdáleností.



Obrázek 57 Simulace LQ regulačního pochodu

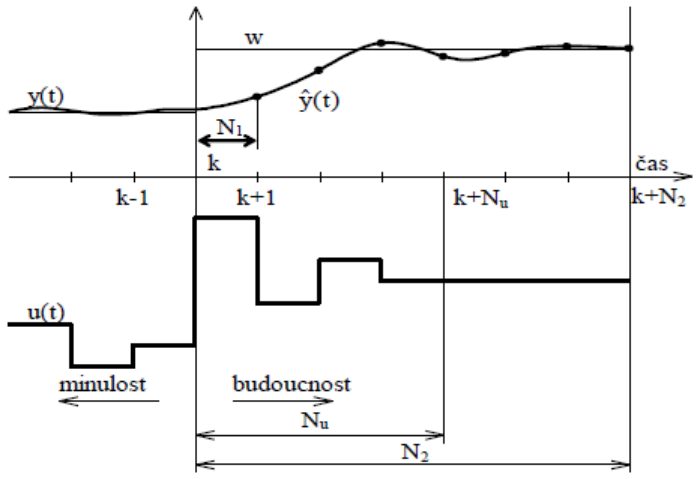


Obrázek 58 LQ regulační pochod – reálný systém

5.3 MPC regulátor

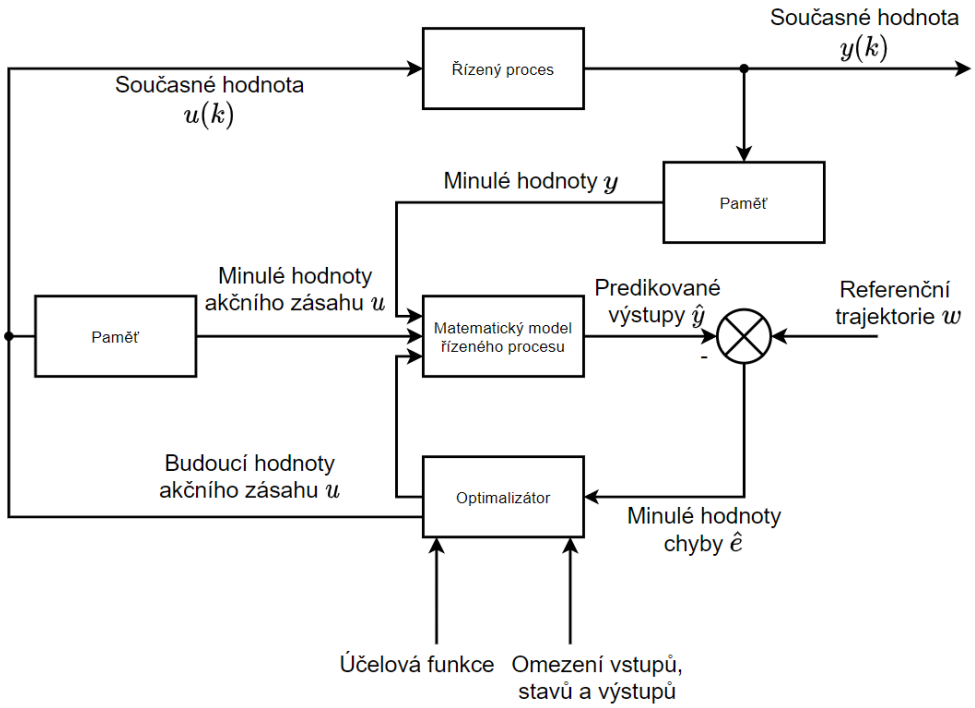
Jedná se o moderní metodu řízení založenou na predikcích N budoucích hodnot výstupů systému $\hat{\boldsymbol{y}}$ (horizont predikce). Tyto predikce jsou určovány na základě matematického modelu řízeného procesu a je zde tedy požadavek, aby byl model schopen dostatečně přesně zachytit dynamiku reálného systému. Dále je nutno podotknout, že metoda je ve své podstatě založena na použití diskrétního modelu systému. Referenční trajektorie žádané hodnoty $\boldsymbol{w}$ je předem určená a díky tomu je prediktivní regulátor schopen s předstihem reagovat na změny žádané hodnoty. Trajektorie budoucích akčních zásahů, resp. přírůstků $\Delta\boldsymbol{u}$ je potom určena řešením optimalizačního problému, jehož součástí je vhodná, většinou kvadratická účelová funkce a omezení. Lepší představu o problému můžeme získat pomocí obrázku 59. Zde vidíme, že N_1 a N_2 určují velikost horizontu predikce a N_u je řídící

horizont určující počet prvků vektoru budoucích akčních zásahů, určených tak aby budoucí výstupy $\hat{\boldsymbol{y}}$ co nejlépe sledovaly referenční trajektorii $\boldsymbol{w}$. Řídící horizont N_u přirozeně volíme stejný nebo kratší, než je délka horizontu predikce N. Přestože je v každém kroku vypočtena celá trajektorie akčních zásahů $\Delta\boldsymbol{u}$, k řízení je použit pouze první člen této posloupnosti a v dalším kroku se celý postup výpočtu opakuje. Je třeba si uvědomit, že na reálný systém působí poruchy a také se zde projeví nepřesnost matematického modelu. Tyto vlivy však nejsou zahrnuty do výpočtu trajektorie akčních zásahů $\Delta\boldsymbol{u}$, a tedy použití více prvků takto určené posloupnosti může vést k nekorektnímu chování řízeného systému. Proto se v každém kroku výpočet trajektorie akčních zásahů $\Delta\boldsymbol{u}$ opakuje na základě aktuálních hodnot stavů systému. Tento postup je známý jako *strategie pohyblivého horizontu*. [40]



Obrázek 59 Princip prediktivního řízení [40]

Principiální blokové schéma prediktivního regulátoru můžeme vidět na obrázku 60.



Obrázek 60 Blokové schéma prediktivního regulátoru [40]

Z obrázku 60 je patrné, že k implementaci prediktivního řízení je zapotřebí určit predikce výstupů systému $\hat{\boldsymbol{y}}$, tedy výstupy matematického modelu. Výstup matematického modelu lze určit jako součet volné a nucené odezvy modelu. Volná odezva modelu $\boldsymbol{y_0}$ je dána pouze minulými hodnotami akčních zásahů a výstupů modelu a předpokládá se, že budoucí přírůstky akčního zásahu budou nulové. Nucená odezva $\boldsymbol{y_n}$ potom zohledňuje nenulové budoucí přírůstky akčního zásahu. Budeme uvažovat diskrétní stavový popis systému ve tvaru:

$$\begin{aligned} \boldsymbol{x}(k+1) &= \boldsymbol{A}\boldsymbol{x}(k) + \boldsymbol{B}\boldsymbol{u}(k) \\ \boldsymbol{y}(k) &= \boldsymbol{C}\boldsymbol{x}(k) \end{aligned} \tag{68}$$

Dále je nutné pro zajištění nulové regulační odchylky model systému doplnit o integrátor, což provedeme tak, že vstupem do modelu systému nebude akční zásah $\boldsymbol{u}(k)$, ale pouze jeho přírůstek $\Delta\boldsymbol{u}(k)$:

$$\boldsymbol{u}(k) = \Delta\boldsymbol{u}(k) + \boldsymbol{u}(k-1) \tag{69}$$

Definujeme nový stavový vektor:

$$\overline{\boldsymbol{x}}(k) = \begin{bmatrix} \boldsymbol{x}(k) \\ \boldsymbol{u}(k-1) \end{bmatrix} \tag{70}$$

Z rovnice (68) potom pro krok $(k+1)$ dostaneme rovnici:

$$\begin{gathered} \boldsymbol{x}(k+1) = \boldsymbol{A}\boldsymbol{x}(k) + \boldsymbol{B}\big(\boldsymbol{u}(k-1) + \Delta\boldsymbol{u}(k)\big) \\ \boldsymbol{u}(k) = \boldsymbol{u}(k-1) + \Delta\boldsymbol{u}(k) \end{gathered} \tag{71}$$

A výsledná rovnice stavu s doplněným integrátorem bude:

$$\overline{\boldsymbol{x}}(k+1) = \overline{\boldsymbol{A}}\overline{\boldsymbol{x}}(k) + \overline{\boldsymbol{B}}\Delta\boldsymbol{u}(k) \tag{72}$$

A rovnice výstupu bude mít tvar:

$$\boldsymbol{y}(k) = \overline{\boldsymbol{C}}\overline{\boldsymbol{x}}(k) \tag{73}$$

Vektor predikcí výstupu potom bude dán jako:

$$\begin{gathered} \hat{\boldsymbol{y}}(k+1) = \overline{\boldsymbol{C}}\overline{\boldsymbol{x}}(k+1) = \overline{\boldsymbol{C}}(\overline{\boldsymbol{A}}\overline{\boldsymbol{x}}(k) + \overline{\boldsymbol{B}}\Delta\boldsymbol{u}(k)) \\ \hat{\boldsymbol{y}}(k+2) = \overline{\boldsymbol{C}}\overline{\boldsymbol{x}}(k+2) = \overline{\boldsymbol{C}}(\overline{\boldsymbol{A}}\overline{\boldsymbol{x}}(k+1) + \overline{\boldsymbol{B}}\Delta\boldsymbol{u}(k+1)) \\ \vdots \\ \hat{\boldsymbol{y}}(k+N_2) = \overline{\boldsymbol{C}}\overline{\boldsymbol{x}}(k+N_2) = \overline{\boldsymbol{C}}(\overline{\boldsymbol{A}}\overline{\boldsymbol{x}}(k+N_2-1) + \overline{\boldsymbol{B}}\Delta\boldsymbol{u}(k+N_2-1)) \end{gathered} \tag{74}$$

kde $\bar{x}$ jsou predikce stavů modelu systému dané jako:

$$\begin{gathered} \overline{\boldsymbol{x}}(k+1) = \overline{\boldsymbol{A}}\overline{\boldsymbol{x}}(k) + \overline{\boldsymbol{B}}\Delta\boldsymbol{u}(k) \\ \overline{\boldsymbol{x}}(k+2) = \overline{\boldsymbol{A}}\overline{\boldsymbol{x}}(k+1) + \overline{\boldsymbol{B}}\Delta\boldsymbol{u}(k+1) = \\ = \overline{\boldsymbol{A}}^2\overline{\boldsymbol{x}}(k) + \overline{\boldsymbol{A}}\overline{\boldsymbol{B}}\Delta\boldsymbol{u}(k) + \overline{\boldsymbol{B}}\Delta\boldsymbol{u}(k+1) \\ \overline{\boldsymbol{x}}(k+3) = \overline{\boldsymbol{A}}\overline{\boldsymbol{x}}(k+2) + \overline{\boldsymbol{B}}\Delta\boldsymbol{u}(k+2) = \\ = \overline{\boldsymbol{A}}^3\overline{\boldsymbol{x}}(k) + \overline{\boldsymbol{A}}^2\overline{\boldsymbol{B}}\Delta\boldsymbol{u}(k) + \overline{\boldsymbol{A}}\overline{\boldsymbol{B}}\Delta\boldsymbol{u}(k+1) + \overline{\boldsymbol{B}}\Delta\boldsymbol{u}(k+2) \\ \vdots \\ \overline{\boldsymbol{x}}(k+N_2) = \overline{\boldsymbol{A}}^{N_2}\overline{\boldsymbol{x}}(k) + \overline{\boldsymbol{A}}^{N_2-1}\overline{\boldsymbol{B}}\Delta\boldsymbol{u}(k) + \cdots + \overline{\boldsymbol{B}}\Delta\boldsymbol{u}(k+N_2-1) \end{gathered} \tag{75}$$

Spojením predikcí do vektorového tvaru dostaneme vektor predikcí výstupu:

$$\hat{\boldsymbol{y}} = \boldsymbol{G}\tilde{\boldsymbol{u}} + \boldsymbol{y_0} \tag{76}$$

kde $\boldsymbol{G}\tilde{\boldsymbol{u}}$ je nucená odezva modelu a $\boldsymbol{y_0}$ je volná odezva systému. Samotná matice $\boldsymbol{G}$ je tzv. jacobián modelu a má tvar:

$$\boldsymbol{G} = \overline{\boldsymbol{C}}\begin{bmatrix} \overline{\boldsymbol{B}} & 0 & \cdots & \cdots & 0 \\ \overline{\boldsymbol{A}}\overline{\boldsymbol{B}} & \overline{\boldsymbol{B}} & 0 & \cdots & 0 \\ \overline{\boldsymbol{A}}^2\overline{\boldsymbol{B}} & \overline{\boldsymbol{A}}\overline{\boldsymbol{B}} & \overline{\boldsymbol{B}} & \cdots & \vdots \\ \vdots & \vdots & \vdots & \overline{\boldsymbol{B}} & 0 \\ \overline{\boldsymbol{A}}^{N_2-1}\overline{\boldsymbol{B}} & \overline{\boldsymbol{A}}^{N_2-2}\overline{\boldsymbol{B}} & \cdots & \cdots & \overline{\boldsymbol{B}} \end{bmatrix} \tag{77}$$

a vektor volné odezvy má tvar:

$$\boldsymbol{y_0} = \overline{\boldsymbol{C}}\begin{bmatrix} \overline{\boldsymbol{A}} \\ \overline{\boldsymbol{A}}^2 \\ \vdots \\ \overline{\boldsymbol{A}}^{N_2} \end{bmatrix}\overline{\boldsymbol{x}}(k) \tag{78}$$

Predikce výstupu $\hat{\boldsymbol{y}}$ je potom:

$$\hat{\boldsymbol{y}} = \boldsymbol{G}\tilde{\boldsymbol{u}} + \boldsymbol{y_0} = \overline{\boldsymbol{C}}\begin{bmatrix} \overline{\boldsymbol{B}} & 0 & \cdots & \cdots & 0 \\ \overline{\boldsymbol{A}}\overline{\boldsymbol{B}} & \overline{\boldsymbol{B}} & 0 & \cdots & 0 \\ \overline{\boldsymbol{A}}^2\overline{\boldsymbol{B}} & \overline{\boldsymbol{A}}\overline{\boldsymbol{B}} & \overline{\boldsymbol{B}} & \cdots & \vdots \\ \vdots & \vdots & \vdots & \overline{\boldsymbol{B}} & 0 \\ \overline{\boldsymbol{A}}^{N_2-1}\overline{\boldsymbol{B}} & \overline{\boldsymbol{A}}^{N_2-2}\overline{\boldsymbol{B}} & \cdots & \cdots & \overline{\boldsymbol{B}} \end{bmatrix}\tilde{\boldsymbol{u}} + \overline{\boldsymbol{C}}\begin{bmatrix} \overline{\boldsymbol{A}} \\ \overline{\boldsymbol{A}}^2 \\ \vdots \\ \overline{\boldsymbol{A}}^{N_2} \end{bmatrix}\overline{\boldsymbol{x}}(k) \tag{79}$$

Nyní je potřeba určit budoucí akční zásahy tak, aby byla minimalizována účelová funkce (80):

$$J = E\left\{\sum_{i=N_1}^{N_2}\delta(i)[\hat{\boldsymbol{y}}(k+i) - \boldsymbol{w}(k+i)]^2 + \sum_{i=N_1}^{N_u}\lambda(i)[\Delta\boldsymbol{u}(k+i-1)]^2\right\} \tag{80}$$

kde $\delta(i)$ a $\lambda(i)$ jsou volitelné parametry, pomocí kterých můžeme ovlivňovat chování řízeného systému.

Převedeme účelovou funkci (80) na maticový tvar:

$$J = \delta\big((\hat{\boldsymbol{y}} - \boldsymbol{w})^T(\hat{\boldsymbol{y}} - \boldsymbol{w})\big) + \lambda\tilde{\boldsymbol{u}}^T\tilde{\boldsymbol{u}} \tag{81}$$

Do rovnice (81) dosadíme rovnici (76) a dostaneme:

$$J = \delta\big((\boldsymbol{G}\tilde{\boldsymbol{u}} + \boldsymbol{y_0} - \boldsymbol{w})^T(\boldsymbol{G}\tilde{\boldsymbol{u}} + \boldsymbol{y_0} - \boldsymbol{w})\big) + \lambda\tilde{\boldsymbol{u}}^T\tilde{\boldsymbol{u}} \tag{82}$$

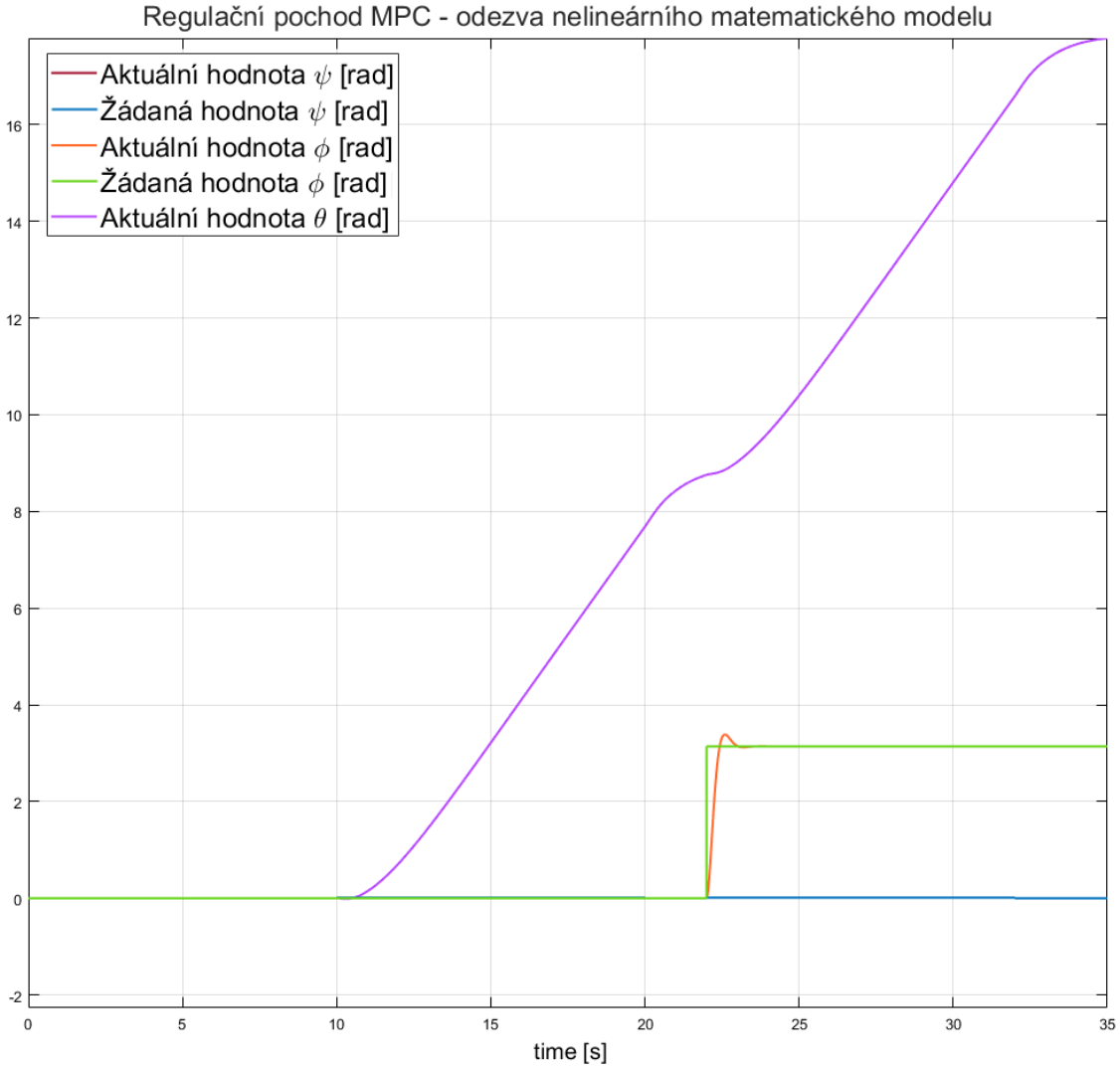
Realizaci tohoto regulátoru lze provést naprogramováním prediktoru (79) do Matlab skriptu. Následnou optimalizaci, tedy hledání minima účelové funkce (80) lze jako úlohu

kvadratického programování vyřešit pomocí funkce *quadprog.* Další možností je použití speciální knihovny – *Model Predictive Control toolbox* systému Matlab. Výhodou tohoto toolboxu je, že obsahuje funkční prediktor a optimalizátor a poskytuje tak jednoduchý způsob implementace MPC regulátoru. Pomocí funkce *mpc* po zadání vstupních parametrů, kterými jsou:

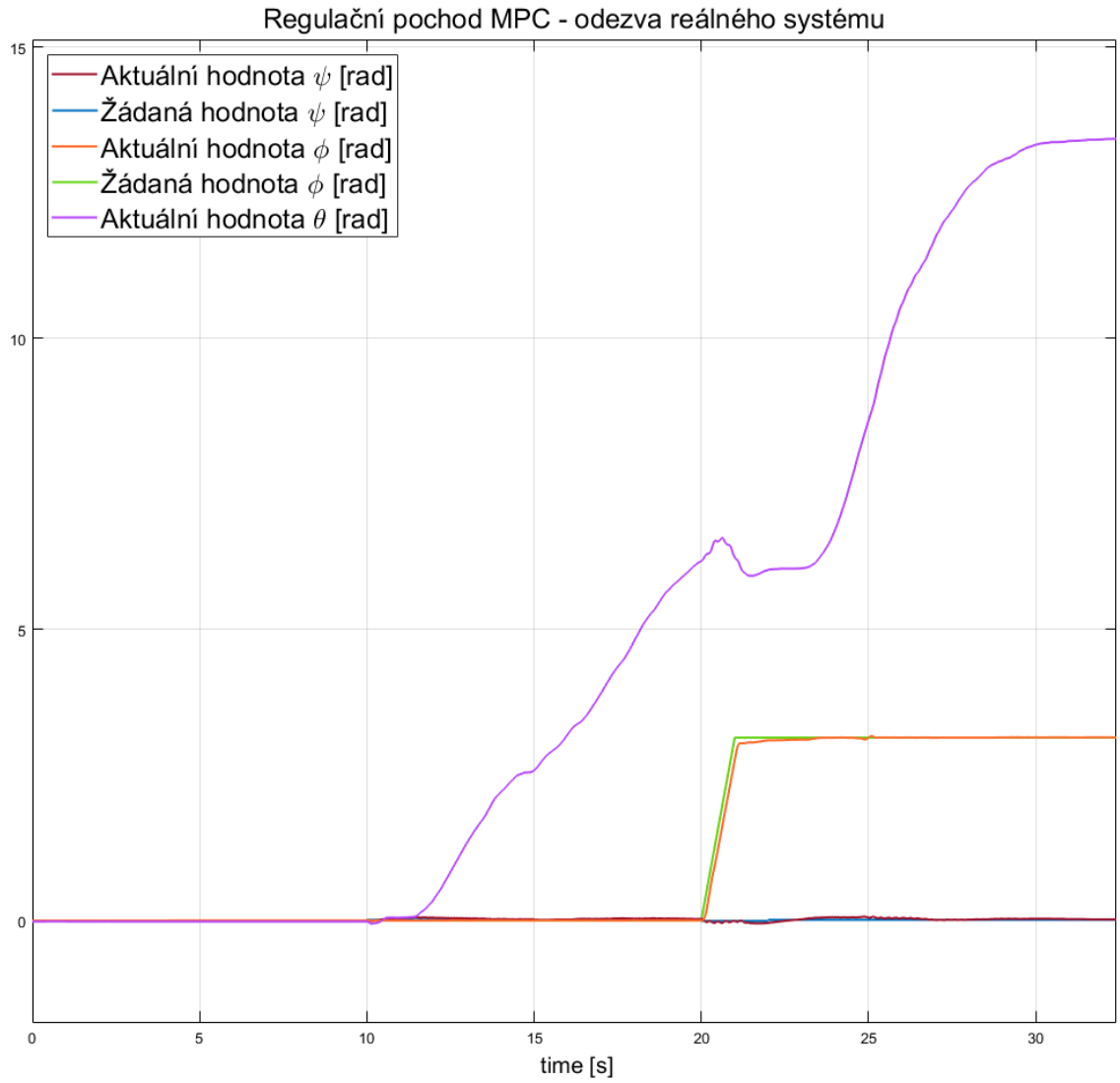
- diskrétní stavový popis systému nebo spojitý stavový popis a vzorkovací perioda
- horizont predikce
- řídicí horizont
- parametry δ a λ, které můžeme chápat jako ekvivalent váhových matic $\boldsymbol{M}$ a $\boldsymbol{R}$ funkce *lqr*
- parametry určující omezení vstupů, stavů a výstupů
- měřitelné poruchy a náhodné poruchy (šum)

získáme *mpcobject,* který lze prostřednictvím funkčního bloku *MPC Controller* systému Simulink použít dále jako regulátor systému. Podobně jako u LQ regulátoru je třeba určit váhové matice. Hlavní důraz je opět kladen na stabilizaci robota ve vzpřímené poloze. V této části návrhu se projevil již zmíněný problém řízení vzájemně závislých stavů systému. Nepodařilo se mi nalézt vhodnou váhovou matici tak aby výsledný regulátor současně robota stabilizoval ve vzpřímené poloze a zároveň byl schopen řídit jeho pozici. Na druhou stranu, při nastavení priority na stabilizaci robota regulátor stabilizoval robota velmi dobře a při poruše (vychýlení robota ze vzpřímené polohy rukou) regulátor reagoval velmi rychle akčním zásahem působícím proti vyvolané změně a výsledkem bylo, že robota dokázal udržet téměř na stejné pozici, na které se nacházel v okamžiku začátku působení poruchy. Dále regulátor dosahoval velmi dobrých výsledků při řízení rotace okolo vertikály. Konkrétní nastavení parametrů MPC regulátoru je součástí této práce jako příloha P I. Rozhodl jsem se alespoň částečně vyřešit problém řízení pohybu robota pomocí jednoduché úvahy, kterou lze použít i u kaskádního PID regulátoru [12]. Hlavní myšlenkou je, že pokud chceme, aby se robot pohyboval vpřed, je možné toho dosáhnout nastavením nenulové žádané hodnoty úhlu odklonu od vertikály. A jelikož navržený MPC regulátor je schopný velmi dobře řídit tento stav systému, je tak možné provést podobný experiment jako v kapitole 5.2. Je však nutné si uvědomit, že v tomto případě není žádanou hodnotou vzdálenost a z tohoto pohledu se tedy jedná spíše o ovládání než řízení. Na

druhou stranu čím větší odchylku od vertikály nastavíme, tím rychleji se robot bude pohybovat, takže tímto způsobem můžeme ovládat i rychlost pohybu robota. Tuto teorii jsem ověřil jak simulačně (obrázek 61), tak experimentem na reálném systému (obrázek 62). Navíc informaci o poloze robota máme k dispozici, takže se jako možná varianta řešení nabízí kombinace MPC regulátoru s jeho uvedenými výhodami a PID regulátoru řídícího odklon robota od vertikály na základě požadované polohy.



Obrázek 61 Simulace MPC regulačního pochodu



Obrázek 62 MPC regulační pochod – reálný systém

6 TYPOVÉ ÚLOHY PRO PRÁCI S TRANSPORTÉREM INTECO A POROVNÁNÍ NAVRŽENÝCH REGULÁTORŮ

6.1 Stabilizace robota

Úkolem je navrhnout regulační obvod schopný stabilizovat robota ve vzpřímené poloze. Jedná se o jednoduchou úlohu sloužící především k seznámení s transportérem Inteco. K řešení této úlohy doporučuji použít PID regulátor z několika důvodů:

- Není třeba vytvářet složitá bloková schémata v Simulinku
- Návrh PID regulátoru je poměrně jednoduchý a rychlý
- V případě nevhodného chování regulátoru lze většinou jednoduše nalézt a odstranit příčinu
- Parametry PID regulátoru je možné měnit přímo v průběhu experimentu na reálném systému bez nutnosti opakované kompilace celé aplikace

Zejména poslední bod je důležitý, protože si uživatel může vyzkoušet ladit parametry PID regulátoru online a lépe tak pochopit význam jednotlivých složek PID. Uživatel také získá lepší představu o chování a vlastnostech transportéru Inteco (dynamika akčních členů, citlivost IMU jednotky, přilnavost kol atd.). Tyto zkušenosti budou užitečné při pozdějším návrhu složitějších regulátorů.

Metodu návrhu PID regulátoru lze volit libovolně. Při návrhu metodou Ziegler–Nichols je použití simulace na matematickém modelu důrazně doporučeno zejména při určování kritického zesílení. Podobně při použití jiných metod návrhu PID regulátoru důrazně doporučuji výsledný regulátor simulačně ověřit pomocí přiloženého simulačního modelu.

Porovnání výsledků navržených regulátorů:

Nejlepších výsledků v této úloze dosahoval MPC a LQ regulátor. MPC regulátor dokázal nejrychleji reagovat na velké poruchy. Na menší poruchy reagoval také dobře, ale z důvodů popsaných v kapitole 5.3 nedokázal udržovat polohu robota v přijatelných mezích. LQ regulátor podával jen nepatrně horší výsledky v reakci na větší poruchy, což se projevilo změnou pozice robota. Na druhou stranu však na vychýlení z polohy reagoval a dokázal robota po odeznění poruchy uvést na původní pozici. PID regulátor podával v porovnání s výše uvedenými regulátory průměrné výsledky, avšak jeho návrh a implementace byly časově nejméně náročné.

Kvalitu regulačního pochodu lze posuzovat např. podle následujících kritérií [41]:

- Kritérium trvalé regulační odchylky

  Podle tohoto kritéria hodnotíme průběh řízeného výstupu systému na nekonečném časovém horizontu. Uvažujeme-li stabilní regulační obvod, potom např. při skokové změně žádané hodnoty nám toto kritérium udává, jak přesně se regulovaná veličina přiblíží žádané hodnotě, resp. jak velká bude regulační odchylka *e* v čase $t \to \infty$. Trvalá regulační odchylka (TRO) je definována jako:

$$TRO = \lim_{t \to \infty} e(t) \tag{83}$$

- Kritérium maximálního přeregulování

  Toto kritérium hodnotí velikost překmitu řízeného výstupu y_M (v první fázi řízení) nad mez danou hodnotou žádané veličiny *w*. Maximální přeregulování se zpravidla vyhodnocuje v procentech a je definované jako:

$$\sigma[\%] = \frac{y_M}{w} \tag{84}$$

- Kritérium podle doby regulace

  Za dobu regulace můžeme považovat čas, za který řízený výstup *y* dosáhne pásma v okolí žádané hodnoty *w* a oblast tohoto pásma $(\pm\delta)$ již neopustí. Potom platí:

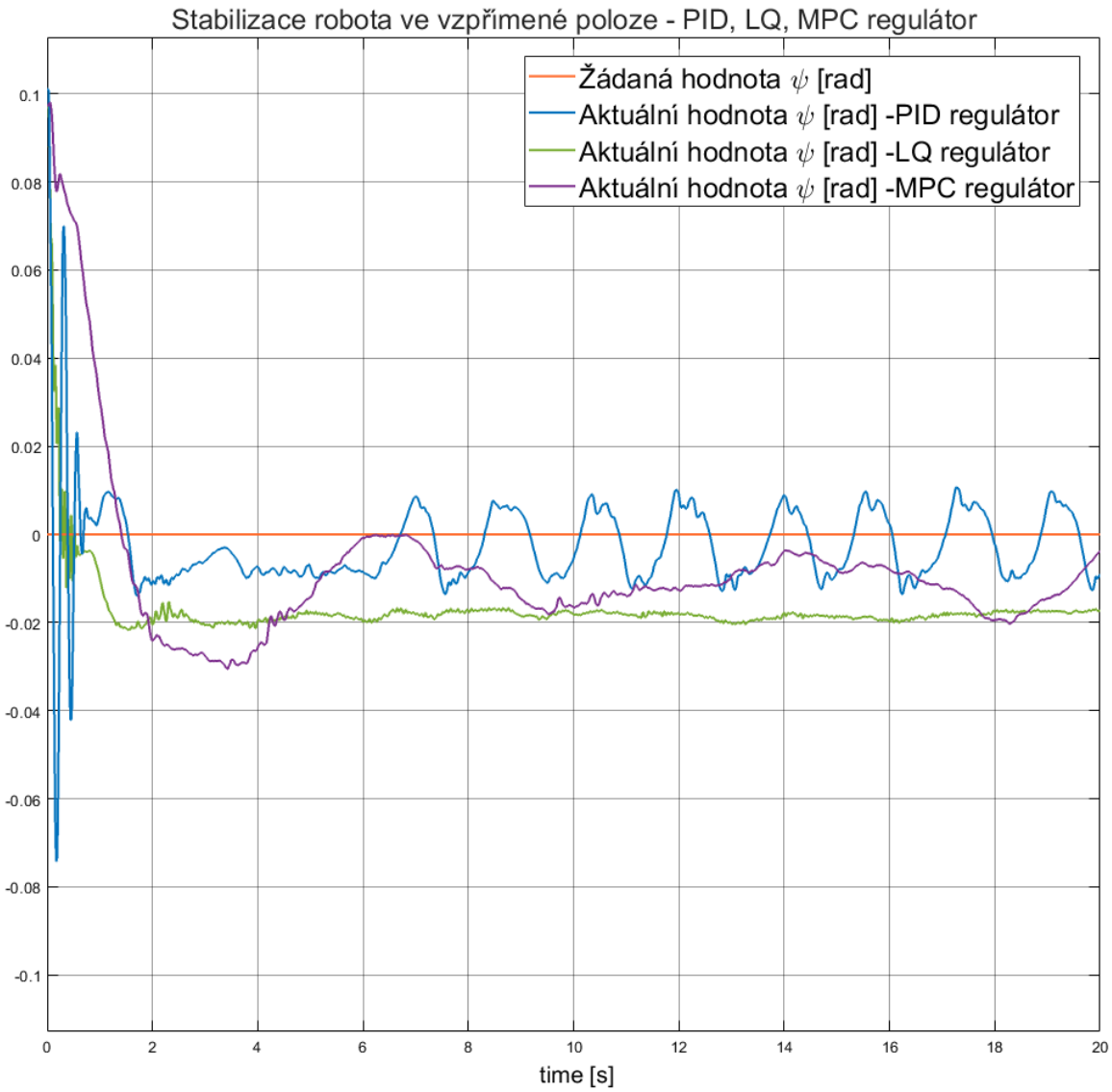
$$t > t_p : w - \delta \leq y(t) \leq w + \delta \tag{85}$$

- Kritérium podle tlumení regulačního pochodu

  Toto kritérium lze použít pro hodnocení periodických regulačních pochodů za předpokladu, že řízený výstup *y* má charakter tlumených kmitů. Kritérium zkoumá dvě za sebou následující amplitudy A_1 a A_2 tlumených kmitů a hodnotí koeficient tlumení ξ podle vztahu:

$$\xi = \frac{A_1 - A_2}{A_1} \; ; 0 < \xi < 1 \tag{86}$$

Porovnání regulačního pochodu jednotlivých regulátorů můžeme vidět na obrázku 63. Dle výše uvedených kritérií můžeme dojít k několika závěrům. Všechny tři navržené regulátory pracují s nějakou trvalou regulační odchylkou. Podle kritéria maximálního přeregulování nejlepších výsledků dosáhl LQ regulátor s hodnotou překmitu (podkmit) $y_m = -0{,}021\,rad$ a nejhorších výsledků dosáhl PID regulátor s hodnotou překmitu $y_m = -0{,}074\,rad$. Podle kritéria hodnotícího dobu regulace při zvoleném pásmu $\delta = \pm 0{,}02$ nejlepších výsledků dosáhl PID regulátor s časem regulace $t_p = 0{,}6\,s$. Následuje LQ regulátor s časem regulace $t_p = 4{,}12\,s$ a MPC regulátor s časem regulace $t_p = 4{,}36\,s$. Dle kritéria hodnotícího tlumení regulačního pochodu dosáhl nejlepších výsledků LQ regulátor následovaný MPC regulátorem. PID regulátor z pohledu tohoto kritéria dosahoval nejhorších výsledků, protože výstup systému měl charakter netlumených kmitů.



Obrázek 63 Porovnání regulačního pochodu PID, LQ, MPC regulátoru

6.2 Stabilizace robota a pohyb vpřed

Úkolem je navrhnout regulační obvod schopný stabilizovat robota ve vzpřímené poloze a poté robota uvést do pohybu vpřed. Jedná se o nepatrně složitější úlohu, ke které může uživatel přistoupit různými způsoby. Nejsnadnějším řešením je použít regulátor navržený v kapitole 6.1 a pohyb vpřed vyvolat nenulovou žádanou hodnotou úhlu ψ. Dalším možným řešením je návrh složitějšího regulátoru. V této části práce s modelem bych doporučil návrh LQ regulátoru, viz kapitola 5.2. Uživatel se může pro začátek soustředit na nastavení vhodných hodnot váhové matice $\boldsymbol{M}$ pro stavy ψ a θ a váhové matice $\boldsymbol{R}$ pro vstupy systému. Návrh LQ regulátoru lze provést v systému Matlab buď výpočtem pomocí rovnic (64), (65), (66), (67) nebo přímo pomocí funkce *lqr*. K návrhu je potřeba lineární stavový popis systému. Uživatel se může pokusit o linearizaci nelineárního matematického modelu (50)

například pomocí nástroje Model Linearizer, nebo použít linearizovaný matematický model (59).

Porovnání výsledků navržených regulátorů:

Požadavky této úlohy splnily všechny navrhované regulátory, jelikož zadání nespecifikuje, zda má být pohyb vpřed řízený.

6.3 Stabilizace robota a pohyb po zadané trajektorii

Úkolem je navrhnout regulační obvod schopný stabilizovat robota ve vzpřímené poloze a poté robota přemístit po předem dané trajektorii. Je tedy nutné navrhnout regulátor schopný ovlivňovat chování robota ve všech třech zobecněných souřadnicích, pomocí kterých byl vytvářen matematický model v kapitole 3.2. Z uvedených algoritmů řízení, dle mých zkušeností, je nejlepší volbou pro řešení této úlohy LQ regulátor. Výhodou je, pokud uživatel již použil LQ regulátor k řešení úlohy 6.2. Návrh regulátoru pak opět spočívá v nalezení vhodných váhových matic stavu a akčního zásahu $\boldsymbol{M}$ a $\boldsymbol{R}$. Při řešení této úlohy je však složitější nalézt vhodný kompromis pro nastavení váhové matice $\boldsymbol{M}$ a do účelové funkce (64) je nutné zahrnout minimalizaci nejen stavů jednotlivých zobecněných souřadnic, ale také jejich derivací podle času.

Porovnání výsledků navržených regulátorů:

V této úloze jednoznačně nejlepších výsledků dosáhl LQ regulátor, protože jako jediný navržený regulátor dokázal řídit všechny požadované stavy systému (viz obrázek 58). Dále byl proveden podobný experiment s MPC regulátorem (viz obrázek 62), a z výsledků lze usoudit, že za cenu použití složitějšího regulačního obvodu (viz kapitola 5.3) by MPC regulátor mohl dosáhnout minimálně srovnatelných výsledků jako LQ regulátor.

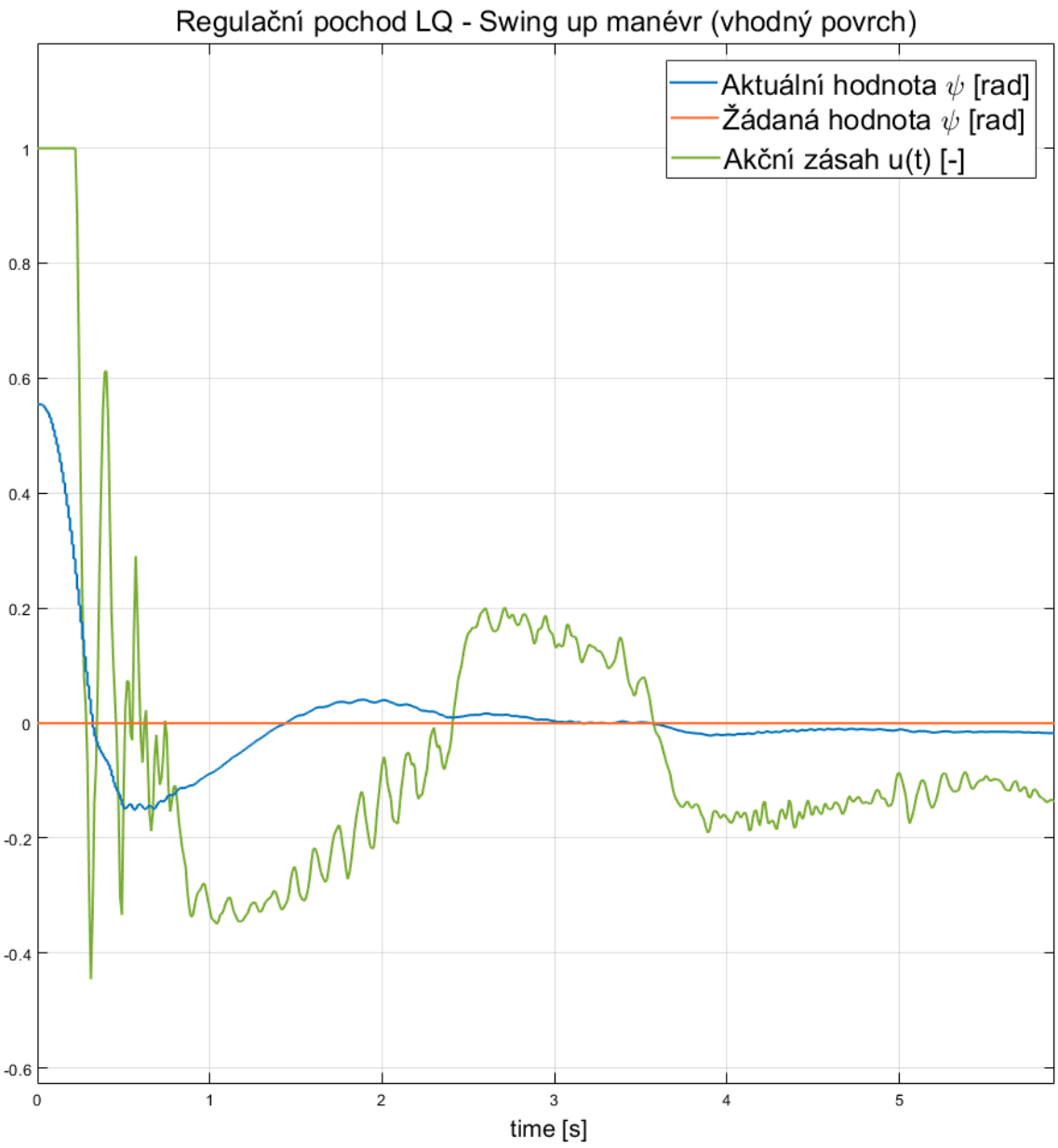
6.4 Provedení Swing-up manévru

Úkolem je navrhnout regulační obvod schopný uvést robota z klidové polohy do vzpřímené polohy a v této poloze jej udržet. Cílem této úlohy je ukázat závislost chování systému, resp. regulačního obvodu na počátečních a provozních podmínkách. Uživatel si při řešení této úlohy může ověřit robustnost již navržených regulátorů, popřípadě se přesvědčit o vlivu provozních podmínek na dosažené výsledky řízení. Při řešení této úlohy doporučuji první experimenty provádět na povrchu, kde lze očekávat vysoký součinitel

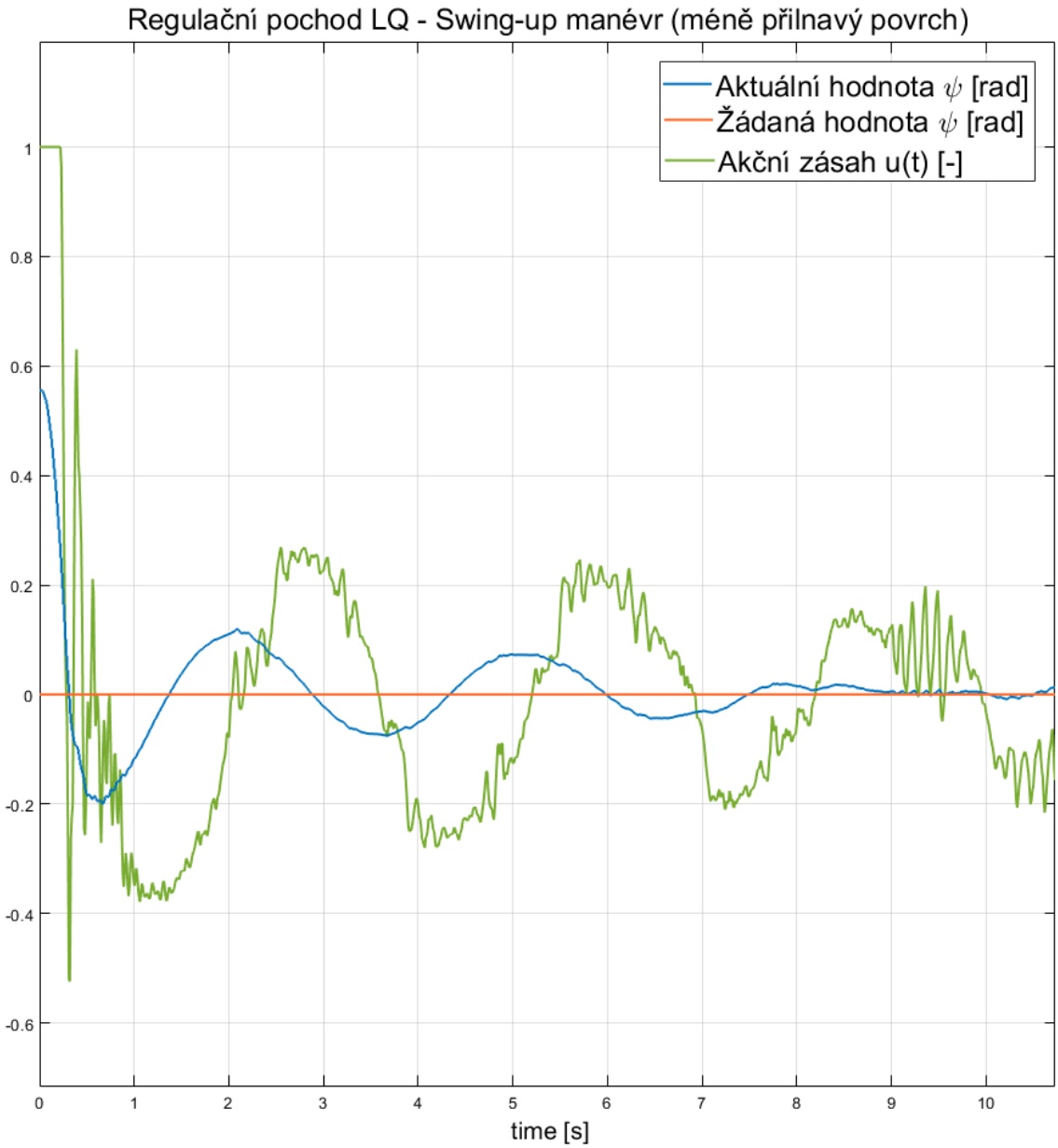
tření mezi kolem a povrchem. Osobně jsem k tomuto účelu použil pěnovou PVC podložku. Pokud se regulátoru podaří robota uvést do žádané polohy a v této poloze jej stabilizovat, může uživatel přikročit k provedení experimentu na méně přilnavém povrchu.

Porovnání výsledků navržených regulátorů:

Úlohu se podařilo splnit pomocí LQ a PID regulátoru. MPC regulátor bohužel nebyl schopen robota uvést do požadované polohy s žádným z testovaných nastavení. Průběh experimentu s PID regulátorem je na obrázku 55. Z průběhu je patrné tlumené rozkmitání systému a uspokojivě stabilizovat robota ve vzpřímené poloze dokáže regulátor za cca 3 sekundy. Těchto výsledků však regulátor dosahoval pouze na již zmíněné pěnové podložce. Na laminátové podlaze již nebyl PID regulátor schopen robota uvést do požadované polohy. LQ regulátor opět podával nejlepší výsledky a průběh experimentu (na pěnové podložce) je na obrázku 64. Za těchto podmínek dokázal robota uspokojivě stabilizovat již za 2,5 sekundy. Navíc nároky na akční zásah jsou v porovnání s PID regulátorem menší. Na méně přilnavém povrchu je robot uspokojivě stabilizován za 8 sekund viz obrázek 65.



Obrázek 64 Swing-up manévr LQ regulátor (vhodný povrch)



Obrázek 65 Swing-up manévr LQ regulátor (méně přilnavý povrch)

ZÁVĚR

Hlavním cílem diplomové práce bylo zprovoznění a návrh řízení dvoukolového robota Inteco. Za tímto účelem jsem prostudoval dostupné literární zdroje, z nichž byla vypracována rešerše a obecný popis konstrukčního řešení dvoukolových robotů. Na základě získaných poznatků jsem zvolil vhodný způsob pro vytvoření matematického modelu systému.

Důkladně jsem se seznámil s robotem Inteco a v průběhu práce s robotem jsem odhalil a odstranil několik hardwarových i softwarových závad. Mezi ty méně závažné se řadil nefunkční akumulátor nebo nekompatibilní software. Diagnostika a odstranění těchto závad byly poměrně jednoduché. Jako závažnější problémy bych zařadil opakované uvolňování mechanických částí robota, celkově nízkou torzní tuhost modelu nebo časté náhodné pády systému Matlab v průběhu experimentů. Po diagnostikování uvolněných mechanických částí robota jsem provedl kontrolu a dotažení všech šroubových spojů a na vhodná místa jsem vložil materiál tlumící vibrace. Tato opatření znatelně zlepšila fungování celého systému. Pády systému Matlab se mi nepovedlo úplně odstranit, pouze snížit jejich četnost.

Pro modelování systému jsem použil Lagrangeovskou formulaci mechaniky, pomocí které je systém popisován prostřednictvím zvolených zobecněných souřadnic. Většina praktické části této práce vznikla v systému Matlab. Vytvořený matematický model byl implementován do grafického simulačního rozhraní Simulink a v průběhu další práce se systémem Inteco se potvrdila jeho použitelnost.

Syntéza jednotlivých regulátorů byla provedena technikou *Model-Based desing* pomocí knihoven *Control Systém Toolbox* a *Model Predictive Control Toolbox* dostupných v systému Matlab. Funkčnost navržených regulátorů byla experimentálně ověřena pomocí série úloh. Na základě dosažených výsledků jsem provedl porovnání navržených regulátorů a dospěl k následujícím závěrům.

Za nejvhodnější regulátor (ze všech otestovaných) bych označil stavový LQ regulátor z několika důvodů. Jako jediný dokázal uspokojivě řídit všechny stavy systému. Dále dokázal provést Swing-up manévr, a to i na méně přilnavém povrchu. Z hlediska složitosti návrhu se LQ regulátor řadí pomyslně doprostřed mezi nejjednodušší fixní PID regulátor a nejsložitější MPC regulátor.

MPC regulátor podával velmi dobré výsledky v oblasti stabilizace robota ve vzpřímené poloze a řízení rotace robota okolo vertikály. Nepodařilo se mi nalézt kompromis pro nastavení regulátoru takovým způsobem, aby byl schopen uspokojivě stabilizovat robota a zároveň kontrolovat jeho polohu. Nicméně na velké poruchy regulátor reagoval tak rychle, že se porucha na změně polohy robota téměř neprojevila.

PID regulátor (naladěný heuristickým postupem) dosahoval uspokojivých, ale v porovnání s dalšími regulátory nejhorších výsledků. Byl schopen stabilizovat robota ve vzpřímené poloze. Na poruchy však reagoval velkou změnou polohy robota. Na rozdíl od MPC regulátoru však byl schopný provést Swing-up manévr.

Úlohy použité k ověření regulátorů mají postupně se zvyšující náročnost a byly navrhovány za účelem využití v praktické výuce, případně pro propagační účely fakulty. Stejně tak celá diplomová práce je koncipována jako pomocný materiál pro práci s transportérem Inteco.

Pro budoucí práci s transportérem Inteco bych navrhnul několik doporučení. Velmi vhodné by bylo implementovat indikaci stavu nabití baterií. V průběhu práce s transportérem Inteco byly právě vybité baterie nejčastější příčinou problémů jako nepředvídatelné chování transportéru nebo nestabilní spojení s řídicím počítačem. Dále bych doporučil pravidelnou kontrolu mechanického stavu transportéru, zejména pak dotažení šroubových spojů. Celkově hodnotím transportér od firmy Inteco jako velmi kvalitní výukový model. Práce s modelem z počátku nebyla zcela intuitivní a uživatelsky přívětivá, ale po důkladnějším seznámení se specifiky systému bylo možné realizovat všechny požadované úkony. Věřím, že moje práce pomůže budoucím studentům a uživatelům rychleji překonat tyto počáteční nesnáze a umožní jim intenzivněji se věnovat možnostem, které tento výukový model nabízí.

SEZNAM POUŽITÉ LITERATURY

[1] ROBERGE, James Kerr. The Mechanical Seal. Massachusetts, USA, 1960. Massachusettský technologický institut.

[2] Yorihisa Yamamoto, NXTway-GS Model-Based Design - Control of self-balancing two-wheeled robot built with LEGO Mindstorms NXT, 2009

[3] INTECO LTD. Two-wheeled Unstable Transporter: User's Manual [online]. Krakow, 2017 [cit. 2021-03-20].

[4] ÚŘEDNÍČEK, Zdeněk. Robotika. Zlín: Univerzita Tomáše Bati ve Zlíně, 2012. ISBN 978-80-7454-223-7.

[5] Grasser, Felix & D'Arrigo, Aldo & Colombi, Silvio & Rufer, Alfred. (2002). JOE: A mobile, inverted pendulum. Industrial Electronics, IEEE Transactions on. 49. 107 - 114. 10.1109/41.982254.

[6] SAYIDMARIE, Omar Khalil. Design and real-time control of a new structure of two-wheeled robot. Sheffield, 2016. Dissertation. Department of Automatic Control and Systems Engineering The University of Sheffield.

[7] RODRÍGUEZ, Diego López. Controlador basado en LQ Singular y Modos Deslizantes de quinta generación para un transportador inestable. Ciudad de México, 2016. Thesis. Universidad Nacional Autónoma de México.

[8] PRABHAKAR, SELVAPERUMAL, PUGAZHENTHI, UMAMAHESWARI a ELAMURUNGAN. Online optimization based model predictive control on two wheel Segway system. Karur, India, 2020. Article.

[9] KADLEC, Tomáš. KONSTRUKCE A ŘÍZENÍ NESTABILNÍHO PODVOZKU MOBILNÍHO ROBOTU. Pardubice, 2014. Diplomové práce. UNIVERZITA PARDUBICE Fakulta elektrotechniky a informatiky.

[10] JIMÉNEZ, Fabián, Ilber RUGE a Andrés JIMÉNEZ. Modeling and Control of a Two Wheeled Auto-Balancing Robot: a didactic platform for control engineering education. Colombia, 2014. Article. Universidad de los Llanos - Faculty of Sciences and Engineering.

[11] Balanduino [online]. Denmark: TKJ Electronics [cit. 2021-03-20]. Dostupné z: http://balanduino.tkjelectronics.dk/

[12] Fresh look at self-balancing robot algorithm [online]. Dominik Nowak, 2018 [cit. 2021-03-20]. Dostupné z: https://medium.com/husarion-blog/fresh-look-at-self-balancing-robot-algorithm-d50d41711d58

[13] VÍTEČKOVÁ, Miluše a Antonín VÍTEČEK. STAVOVÉ ŘÍZENÍ [online]. Ostrava: Technická universita Ostrava, Fakulta strojní, Katedra automatizační techniky a řízení, 2016 [cit. 2021-03-20]. ISBN 978-80-248-3900-4. Dostupné z: http://books.fs.vsb.cz/ZRMS/stavove-rizeni.pdf

[14] HESS, Lukáš. NÁVRH DVOUKOLOVÉHO AUTONOMNÍHO ROBOTA. Brno, 2013. Diplomové práce. VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ, FAKULTA STROJNÍHO INŽENÝRSTVÍ, ÚSTAV AUTOMATIZACE A INFORMATIKY.

[15] Model Based Design: Matlab [online]. Praha: Humusoft [cit. 2021-03-20]. Dostupné z: https://www.humusoft.cz/matlab/mbd/

[16] ŠKRABÁNEK, Pavel. Teorie fuzzy množin a její aplikace [online]. Pardubice: Univerzita Pardubice, Fakulta elektrotechniky a informatiky, 2014 [cit. 2021-03-20]. ISBN 978-80-7395-875-6. Dostupné z: https://www.researchgate.net/profile/Pavel-Skrabanek/publication/309010508_Teorie_fuzzy_mnozin_a_jeji_aplikace/links/57fdcca508ae49db47554278/Teorie-fuzzy-mnozin-a-jeji-aplikace.pdf

[17] KREJČÍ, Alois. POKROČILÉ TECHNIKY ŘÍZENÍ POHYBU PRO MECHATRONICKÉ APLIKACE. Plzeň, 2012. Diplomová práce. Západočeská univerzita v Plzni, Fakulta aplikovaných věd, Katedra kybernetiky.

[18] MATTAPALLIL, Jim Joy. Self Balancing Two Wheel Mobile Robot Using Sliding Mode Control. Kerala, 2017. Article. Mar Baselios College of Engineering and Technology,Thiruvananthapuram,Kerala, India.

[19] VOGEL, Bob. The iBot is Back. [online]. July 2019 [cit. 2021-03-20]. Dostupné z: https://www.newmobility.com/2019/07/the-ibot-is-back/

[20] Klemm, Victor & Morra, Alessandro & Salzmann, Ciro & Tschopp, Florian & Bodie, Karen & Gulich, Lionel & Kung, Nicola & Mannhart, Dominik & Pfister, Corentin & Vierneisel, Marcus & Weber, Florian & Deuber, Robin & Siegwart, Roland. (2019). Ascento: A Two-Wheeled Jumping Robot. 7515-7521. 10.1109/ICRA.2019.8793792.

[21] Dynamically-Stable Mobile Robots in Human Environments [online]. 2013 September [cit. 2021-03-20]. Dostupné z: http://www.msl.ri.cmu.edu/projects/ballbot/

[22] Co je a k čemu slouží magnetometr? [online]. 2008-07-24 [cit. 2021-5-11]. Dostupné z: http://www.enviweb.cz/71131

[23] Jak funguje a k čemu slouží akcelerometr? [online]. 2020-10-09 [cit. 2021-5-11]. Dostupné z: https://www.tme.eu/cz/news/library-articles/page/22568/jak-funguje-a-k-cemu-slouzi-akcelerometr/

[24] VEJLUPEK, Josef. MOTION CONTROL OF TWO-WHEEL ROBOT. Brno, 2008. Diplomová práce. FAKULTA STROJNÍHO INŽENÝRSTVÍ ÚSTAV MECHANIKY TĚLES, MECHATRONIKY A BIOMECHANIKY. Vedoucí práce Ing. PAVEL HOUŠKA, Ph.D.

[25] ČERMÁK, David. NÁVRH JEDNODESKOVÉHO ŘÍDÍCÍHO SYSTÉMU PRO MODEL VOZIDLA POHYBUJÍCÍHO SE V AUTONOMNÍM KONVOJI. Brno, 2013. Bakalářská práce. FAKULTA STROJNÍHO INŽENÝRSTVÍ ÚSTAV AUTOMATIZACE A INFORMATIKY. Vedoucí práce ING. STANISLAV VĚCHET PH.D.

[26] Přehled operačních systémů reálného času. AT&P journal [online]. 2005, 3 [cit. 2021-03-04]. Dostupné z: https://www.atpjournal.sk/buxus/docs/atp-2005-10-51.pdf

[27] Co je to Kalmanova filtrace? Automatizace.hw.cz [online]. [cit. 2021-03-05]. Dostupné z: https://automatizace.hw.cz//clanek/2007042901

[28] Yue, Ming & Sun, Wei & Hu, Ping. (2011). Sliding mode robust control for two-wheeled mobile robot with lower center of gravity. International Journal of Innovative Computing, Information and Control. 7.

[29] MegaScout [online]. Minnesota: University of Minnesota, 2003 [cit. 2021-03-21]. Dostupné z: http://distrob.cs.umn.edu/megascout.php

[30] MPU-6000 and MPU-6050 Product Specification Revision 3.4 [online]. InvenSense [cit. 2021-03-21]. Dostupné z: https://invensense.tdk.com/wp-content/uploads/2015/02/MPU-6000-Datasheet1.pdf

[31] MPU-9250 Product Specification Revision 1.1 [online]. InvenSense [cit. 2021-03-21]. Dostupné z: https://invensense.tdk.com/wp-content/uploads/2015/02/PS-MPU-9250A-01-v1.1.pdf

[32] MicroZed 7010 SOM - Zynq SoC based, C-grade [online]. AVNET [cit. 2021-03-22]. Dostupné z: https://www.avnet.com/shop/us/products/avnet-engineering-services/aes-z7mb-7z010-som-g-rev-g-3074457345641351169?INTCMP=AMER-ECOMM-AVNET-BOARDS-AVT-INT-WEB-MICROZED-PRODUCTS-08282021

[33] Linaro Cross Compiler [online]. [cit. 2021-03-23]. Dostupné z: https://www.linaro.org/downloads/

[34] Putty [online]. [cit. 2021-03-23]. Dostupné z: https://www.putty.org/

[35] MATHWORKS. Algebraic Loop Concepts [online]. [cit. 2021-04-04]. Dostupné z: https://www.mathworks.com/help/simulink/ug/algebraic-loops.html

[36] DOSTÁL, P.; MATUŠŮ, R. Stavová a algebraická teorie řízení. Zlín: UTB ve Zlíně, Fakulta aplikované informatiky, 2010

[37] DUŠEK, František. Identifikace a linearizace v Simulinku [online]. Katedra řízení procesů a výpočetní techniky, FCHT, Univerzita Pardubice [cit. 2021-5-11]. Dostupné z: https://www2.humusoft.cz/www/papers/tcp01/001_dusek.pdf

[38] ÚŘEDNÍČEK, Zdeněk. Elektromechanické akční členy. Zlín: Univerzita Tomáše Bati ve Zlíně, 2009. ISBN 978-80-7318-835-1.

[39] Model Predictive Control. 2nd ed. London: Springer, 2004. ISBN 1852336943.

[40] BOBÁL, Vladimír. Adaptivní a prediktivní řízení [online]. Zlín: UTB Fakulta aplikované informatiky, 2007 [cit. 2021-5-2].

[41] DOSTÁL, Petr a František GAZDOŠ. ŘÍZENÍ TECHNOLOGICKÝCH PROCESŮ [online]. Zlín: Univerzita Tomáše Bati ve Zlíně Fakulta aplikované informatiky Ústav řízení procesů, 2006 [cit. 2021-5-12]. ISBN 80-7318-465-6.

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

3D	3 - dimension
ARM	Advanced RISC Machine
BEMF	Back Electromotive Force
CAD	Computer Aided Design
CMU	Carnegie Mellon University
CPLD	Complex Programmable Logic Devices
CTRL	Control
DC	Direct Current
DMP	Digital Motion Procesor
DoF	Degree of Freedom
FPGA	Field Programmable Gate Array
GPIO	General Purpose Input Output
GPS	Global Positioning System
HILS	Hardware in the Loop Simulation
IMU	Inertial Measurement Unit
LAN	Local Area Network
LED	Light Emitting Diode
LIDAR	Light Detection And Ranging
LQ	Linear Quadratic (control)
LQI	Linear Quadratic Integral (control)
LQR	Linear Quadratic Regulator
MBD	Model-Based design
MIMO	Multiple-input multiple-output
MPC	Model Predictive Control
PID	Proportional-Integral-Derivative (PID) control

PWM	Pulse Width Modulation
RP	Real Process
RPM	Rotation per minute
RT	Real Time
RT-DAC	Real Time-Digital to Analog Converter
RTOS	Real-time operating system
SMC	Sliding Mode Control
SoC	System on a chip
SoM	System on module
SPI	Serial Peripheral Interface
USB	Universal Serial Bus
USB-UART	Universal Serial Bus - Universal Asynchronous Receiver Transmitter
Wi-Fi	Wireless Fidelity
WLAN	Wireless Local Area Network

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

SEZNAM PŘÍLOH

PŘÍLOHA P I: MATLAB SKRIPT PRO IMPLEMENTACI LINEARIZOVANÉHO A NELINEÁRNÍHO MODELU, NÁVRH LQ A MPC REGULÁTORU

```
%Konstanty a pomocné proměnné pro tvorbu nelin. mat. modelu
%pohon
ujm = 12;
jmot = 0.00119;
fm = 0.00024;
rm = 1;
kt = 0.025;
kw = 0.025;
n = 10;
%kolo
mk = 0.32;
jk = 0.0013;
R = 0.075;
fw = 0.01;
%skelet
ms = 5.41;
L = 0.102;
jpsi = 0.1656;
jfi = 0.028;
W = 0.4;
g = 9.81;
%pomocné proměnné pro nelin. mat. model
a = (2*mk + ms)*R^2 + 2*jk + 2*jmot;
d = ms*L^2 + jpsi + 2*jmot;
dm1 = (ujm * n * kt) / rm;
dm2 = ((n * kt * kw) / rm) + fm;
ddm2 = 2*(((n * kt * kw) / rm) + fm);
w2r2 = W^2 / (2 * R^2);
w2r= W /(2*R);
mRL = ms * R * L;
mLL = ms * L^2;
mgL = ms * g * L;
djmot = 2 * jmot;
dmLL = 2 * (ms * L^2);
jkjmot = jk + jmot;
mkW2 = 0.5 * (mk * W^2);
%Matice linearizovaného mat.modelu
A = [0 1 0 0 0 0;0 -0.4733 -29.1464 0.4733 0 0;0 0 0 1 0 0;0 0.1401
29.1651 -0.1401 0 0;0 0 0 0 0 1;0 0 0 0 0 -2.6349];
B = [0 0;109.4 109.4;0 0;-32.38 -32.38;0 0;-89.87 89.87];
C = eye(6);
D = zeros(6,2);
%Návrh LQ regulátoru
M = [6 0 0 0 0 0;0 20 0 0 0 0;0 0 10000 0 0 0;0 0 0 0.1 0 0;0 0 0 0 150
0;0 0 0 0 0 30];
R= [600 0;0 600];
sys = ss(A,B,C,D);
[K,S,e] = lqr(A,B,M,R);
%Návrh MPC regulátoru
Ts = 0.01;
sysd = c2d(sys,Ts);
sysd = setmpcsignals(sysd,'MV',1,'MV',2);
MPCobj = mpc(sysd);
MPCobj.MV(1).Min = -1;
```

```
MPCobj.MV(1).Max = 1;
MPCobj.MV(2).Min = -1;
MPCobj.MV(2).Max = 1;
MPCobj.ControlHorizon = 1;
MPCobj.PredictionHorizon = 30;
MPCobj.Weights.ManipulatedVariables(1) = 0;
MPCobj.Weights.ManipulatedVariables(2) = 0;
MPCobj.Weights.ManipulatedVariablesRate(1) = 0.1;
MPCobj.Weights.ManipulatedVariablesRate(2) = 0.1;
MPCobj.Weights.OutputVariables(1) = 0;
MPCobj.Weights.OutputVariables(2) = 0;
MPCobj.Weights.OutputVariables(3) = 500;
MPCobj.Weights.OutputVariables(4) = 0.1;
MPCobj.Weights.OutputVariables(5) = 10;
MPCobj.Weights.OutputVariables(6) = 0.1;
```

PŘÍLOHA P II: ADRESÁŘOVÁ STRUKTURA PŘILOŽENÝCH SOUBORŮ