Dvojité inverzní kyvadlo

Bc. Jakub Trefil

Diplomová práce
2020

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky

Univerzita Tomáše Bati ve Zlíně

Fakulta aplikované informatiky

Ústav automatizace a řídicí techniky

Akademický rok: 2019/2020

ZADÁNÍ DIPLOMOVÉ PRÁCE

(projektu, uměleckého díla, uměleckého výkonu)

Jméno a příjmení:	**Bc. Jakub Trefil**
Osobní číslo:	**A17235**
Studijní program:	**N3902 Inženýrská informatika**
Studijní obor:	**Automatické řízení a informatika**
Forma studia:	**Prezenční**
Téma práce:	**Dvojité inverzní kyvadlo**

Zásady pro vypracování

1. Stručně popište charakteristické vlastnosti nestabilních systémů a porovnejte problematiku regulace stabilních a nestabilních systémů.
2. Popište základní charakteristiky obecného inverzního kyvadla a dvojitého inverzního kyvadla.
3. Na základě literární rešerše uveďte způsoby používané pro regulaci inverzního kyvadla a dvojitého inverzního kyvadla.
4. Navrhněte a realizujte reálné dvojité inverzní kyvadlo. Je možné využít systém Amira PS600 Inverted Pendulum, který je dostupný v Laboratoři reálných systémů FAI.
5. Vytvořte simulační model dvojitého inverzního kyvadla, který bude využitelný při návrhu regulátoru pro reálnou soustavu dvojitého inverzního kyvadla. Srovnejte vlastnosti simulačního modelu a reálné soustavy.
6. Navrhnete regulátor pro dvojité inverzní kyvadlo. Ověřte regulátor na simulačním modelu a na reálné soustavě. Srovnejte obdržené výsledky.

Rozsah diplomové práce:
Rozsah příloh:
Forma zpracování diplomové práce: **tištěná/elektronická**

Seznam doporučené literatury:

1. BOBÁL, Vladimír. *Identifikace systémů*. Zlín: Univerzita Tomáše Bati ve Zlíně, 2009, 128 s. ISBN 978-80-7318-888-7.
2. CHEN, Xianmin, Rongrong YU, Kang HUANG, Shengchao ZHEN, Hao SUN a Ke SHAO. Linear motor driven double inverted pendulum: A novel mechanical design as a testbed for control algorithms. *Simulation Modelling Practice and Theory*. 2018, **81**, 31-50. DOI: 10.1016/j.simpat.2017.11.009. ISSN 1569190X. Dostupné také z: https://linkinghub.elsevier.com/retrieve/pii/S1569190X17301703
3. FORMAL'SKIJ, Aleksandr Moisejevič. *Stabilisation and motion control of unstable objects*. Berlin: De Gruyter, [2015], xvi, 236. De Gruyter studies in mathematical physics. ISBN 978-3-11-037582-4.
4. LAW, Averill M. *Simulation modeling and analysis*. Fifth edition. New York: McGraw-Hill Education, 2015. McGraw-Hill international editions. ISBN 978-1-259-25438-3.
5. *PS600 Laboratory Experiment Inverted Pendulum*. Duisburg: Amira, 2000, 351 s.
6. ŘEZNÍČEK, Bohuslav. *Regulace laboratorního modelu inverzního kyvadla*. Zlín: Univerzita Tomáše Bati ve Zlíně, 2008, 66 s. Dostupné také z: http://hdl.handle.net/10563/5441.

7. XIN, Xin. Analysis of the Energy Based Swing-up Control for a Double Pendulum on a Cart. *IFAC Proceedings Volumes* [online]. 2008, 41(2), 4828-4833 [cit. 2018-10-10]. DOI: 10.3182/20080706-5-KR-1001.00811. ISSN 14746670. Dostupné z: http://linkinghub.elsevier.com/retrieve/pii/S1474667016397063

Vedoucí diplomové práce: **Ing. Petr Chalupa, Ph.D.**
Ústav řízení procesů

Datum zadání diplomové práce: **15. prosince 2019**
Termín odevzdání diplomové práce: **15. května 2020**

UNIVERZITA TOMÁŠE BATI VE ZLÍNĚ
Fakulta aplikované informatiky
L.S.

doc. Mgr. Milan Adámek, Ph.D.
děkan

prof. Ing. Vladimír Vašek, CSc.
ředitel ústavu

Ve Zlíně dne 20. prosince 2019

Prohlašuji, že

- beru na vědomí, že odevzdáním diplomové 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 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 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 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 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 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 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 12.8.2020

Jakub Trefil, v.r.
podpis diplomanta

ABSTRAKT

Diplomová práce řeší problematiku řízení dvojitého inverzního kyvadla na vozíku. Cílem bylo vytvoření simulačních modelů, na kterých je možné ověřit různé způsoby regulace. Ke stabilizaci se využívají klasické metody, jako jsou optimální řízení nebo prediktivní řízení. Pro manévr vyšvihnutí kyvadla z klidu do inverzní polohy byla zvolena netradiční technika strojového učení. Teoretická část je zaměřena na seznámení se s problémem dvojitého inverzního kyvadla a využitelných metod pro jeho řízení. Praktická část zachycuje vývoj reálného modelu a potřebných simulací. Výsledkem práce je funkční řízení reálné soustavy a další simulační programy.

Klíčová slova: dvojité inverzní kyvadlo na vozíku, LQR, prediktivní řízení, swing-up, zpětnovazební učení, diferenciální evoluce, Real-Time Simulink

ABSTRACT

This thesis is concerned with the issue of controlling a double inverted pendulum on a cart. The goal is to create simulation models, which verify various methods of regulation. For stabilization purpose, we use classical methods such as optimal control or model predictive control. A non-traditional machine learning technique was chosen to swing the pendulum from rest to the inverted position. The theoretical part focuses on getting familiar with the double inverted pendulum and usable control methods. The practical part shows the development of a real model and the necessary simulation models. The result of this thesis is a functional control of the real system and additional simulation programs.

Keywords: double inverted pendulum on a cart, LQR, Model Predictive Control, swing-up, Reinforcement Learning, Differential Evolution, Real-Time Simulink

Rád bych poděkoval vedoucímu práce Ing. Petru Chalupovi, Ph.D. Dále bych chtěl poděkovat rodině, která mě během dlouhých studií neustále podporovala. Mimořádné poděkování patří mému otci, který se podílel na tvorbě reálného modelu kyvadla.

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

OBSAH

ÚVOD

Inverzní kyvadlo a jeho varianty jsou v hledáčku teorie řízení už od roku 1960. Postupem času se z něj stal nástroj pro ověřování různých druhů regulace a s rozvojem výpočetní techniky se do popředí dostávají stále složitější algoritmy.

Při řízení kyvadla se zaměřujeme na dvě rozdílné oblasti. První z nich je stabilizace kyvadla v jedné z nestabilních poloh za použití vybraného regulátoru. Druhá část poté tvoří tzv. swing-up manévr, kdy se snažíme kyvadlo převést ze spodní stabilní polohy do inverzní nestabilní polohy a zde ho stabilizovat. Některé techniky bohužel dodnes zůstávají neověřené na reálném modelu a existují pouze jejich simulace pro ideální podmínky.

Dvojité inverzní kyvadlo na vozíku je sytém se třemi stupni volnosti, který obsahuje tři nestabilní polohy a je typickým představitelem nelineárního systému. Zároveň se jedná o podaktuovaný systém, kde tři stupně volnosti jsou řízeny pouze jedním akčním zásahem. Obvykle se jedná o sílu působící na vozík. Pochopení řízení podaktuovaných systémů je velmi důležité s ohledem na jeho využití v oblastech pro kráčející roboty, kvadrokoptéry, ponorky nebo kosmické lodě.

Cílem této práce je vytvoření reálné soustavy dvojitého inverzního kyvadla, na které bude ověřena stabilizace a swing-up. Pro swing-up je zvolena metoda strojového učení, která na reálném modelu kyvadla ještě nebyla ověřena.

Teoretická část práce se zabývá odvozením pohybových rovnic, které jsou potřeba k vytvoření simulačního modelu, a dále jejich linearizací pro návrh regulátoru. Jsou zde stručně popsány vybrané možnosti stabilizace a swing-up manévru.

Praktická část zachycuje tvorbu reálného modelu dvojitého inverzního kyvadla. Je zde ukázka vybraných mechanických částí a přehled zvolené elektroniky. Dále popisuje tvorbu simulačních modelů a skriptů. Na reálné soustavě je poté ověřena schopnost stabilizace a swing-up manévr.

I. TEORETICKÁ ČÁST

1 STABILITA SYSTÉMU

1.1 Stabilní a nestabilní systémy

Mějme systém lineárních diferenciálních rovnic (ODR) a jejich řešení:

$$\dot{x}(t) = \boldsymbol{A}x(t), \quad x \in R^m \tag{1.1}$$

$$x(t) = e^{At}x(0) \tag{1.2}$$

x(0) představuje vektor počátečních podmínek.

Pokud budeme uvažovat matici **A** jako diagonální, prvky na této diagonále nazveme vlastní čísla λ a řešení systému rovnic bude:

$$x(t) = \begin{pmatrix} e^{\lambda_1 t} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & e^{\lambda_n t} \end{pmatrix} x(0) \tag{1.3}$$

Pro obecnou matici **A** existují takové transformace využívající vlastních vektorů, které umožňují přechod na diagonální tvar a zase zpátky. Podrobnou teorii lze nalézt např. v [1].

Základem tohoto systému jsou stavy, které na sobě nejsou nijak závislé. Každý stav má svou dynamiku, která je řízena příslušným vlastním číslem. Každé vlastní číslo obsahuje reálnou a imaginární část. Vlastní čísla matice **A** představují v teorii řízení póly systému.

$$\lambda = a + ib \tag{1.4}$$

S využitím Eulerova vzorce:

$$e^{i\varphi} = \cos\varphi + i\sin\varphi \tag{1.5}$$

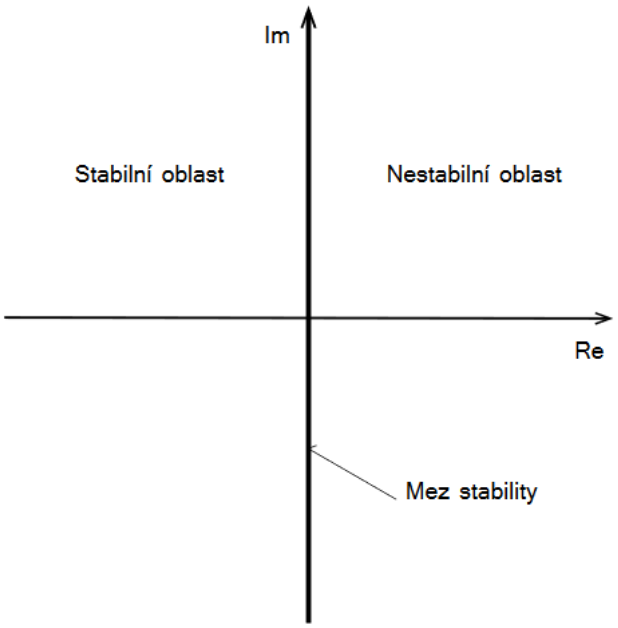
Dostáváme:

$$\begin{gathered} x(t) = e^{\lambda t}x(0) = e^{(a+ib)t}\,x(0) = e^{at}e^{ibt}\,x(0) \\ x(t) = e^{at}(\cos bt + i\sin bt)x(0) \end{gathered} \tag{1.6}$$

Z hlediska stability nás zajímá, jak se bude chovat funkce *x(t)* pro $t \to \infty$. Člen obsahující harmonické funkce bude vždy nabývat omezených hodnot a *x(0)* ovlivňuje pouze počáteční hodnotu. Hlavní roli zde hraje parametr *a*.

Pro $a > 0$ dostáváme rostoucí exponenciálu. Pro $a < 0$ dostáváme exponenciálu jdoucí k nule. Oscilace průběhu závisí na komplexní složce.

Z předešlého můžeme vyvozovat závěry o stabilitě. Pokud matice **A** obsahuje vlastní čísla a všechna vlastní čísla mají zápornou reálnou část, všechny stavy půjdou z libovolné počáteční podmínky k nule a systém je stabilní. Obsahuje-li matice **A** vlastní číslo s kladnou reálnou částí, jeden ze stavů je nestabilní a tím i celý systém. V teorii řízení se poté setkáváme s rozdělením komplexní roviny na stabilní a nestabilní oblast, které jsou rozděleny mezí stability (pouze imaginární část vlastního čísla – netlumené kmity).

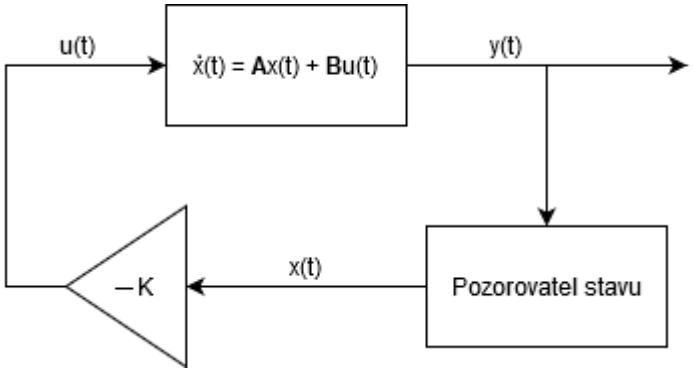


Obrázek 1 – Komplexní rovina stability spojitých systémů, převzato z [2]

1.2 Stabilizace pomocí teorie řízení

V teorii řízení si systémy doplňujeme o akční zásah, díky kterému jsme schopni měnit stavové veličiny. Předmětem návrhu regulátoru je poté využití zpětné vazby tak, aby došlo ke stabilizaci systému s požadovanými vlastnostmi. Ve stavovém řízení fungují techniky pozorovatele stavu, které umožňují odhadovat hodnoty všech stavových veličin (i neměřených) z měřených hodnot. Pro návrh řízení pak můžeme použít:

$$\begin{aligned} \dot{x}(t) &= \boldsymbol{A}x(t) + \boldsymbol{B}u(t), \quad x \in R^m, u \in R^n \\ y(t) &= \boldsymbol{C}x(t), \quad y \in R^p \end{aligned} \tag{1.7}$$



Obrázek 2 – Jednoduchý stabilizační stavový regulátor

Obrázek 2 ukazuje jednoduchý zpětnovazební obvod, pro který platí:

$$u(t) = -Kx(t)$$

$$\dot{x}(t) = \boldsymbol{A}x(t) + \boldsymbol{B}(-Kx(t)) \tag{1.8}$$

$$\dot{x}(t) = (\boldsymbol{A} - \boldsymbol{B}K)x(t)$$

Návrh regulátoru tedy spočívá v nalezení takového vektoru *K*, který přesune vlastní čísla matice *(**A** - **B**K)* do záporné části komplexní roviny. Zároveň je zde zohledněno požadované chování výsledného regulačního obvodu a možnosti akčního členu. U stabilních systémů tedy posouváme vlastní čísla v záporné části komplexní roviny tak, abychom dostali žádané chování systému. U nestabilních systému se snažíme přesunout vlastní čísla z nestabilní oblasti do stabilní.

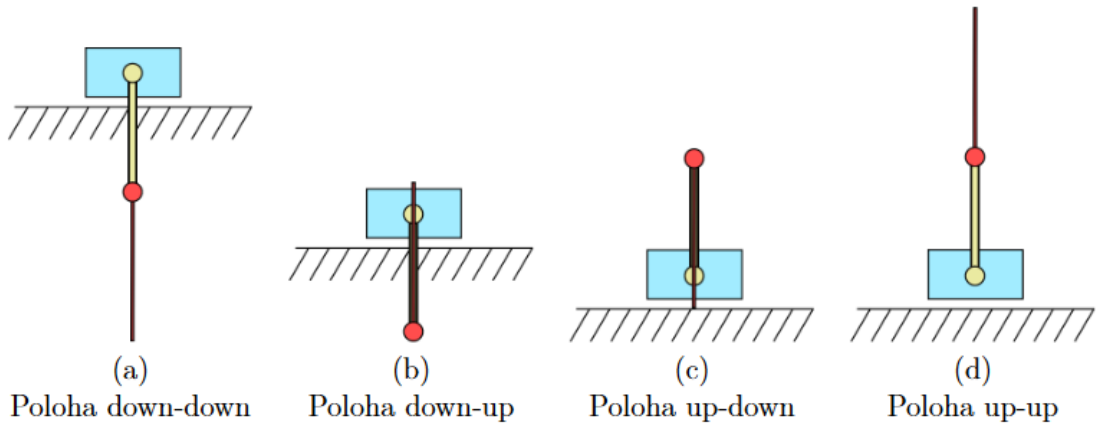
2 DVOJITÉ INVERZNÍ KYVADLO

Inverzní kyvadla se obecně považují za klasické systémy v teorii řízení, na kterých lze testovat různé metody regulace pro svoji vysokou nestabilitu, nelinearitu a vysoký řád systému (čím více ramen kyvadla, tím větší složitost). Zároveň obsahuje více stupňů volnosti než akčních zásahů (*underactuated system*). [3]

Nejjednodušší verzí je jednoduché kyvadlo na vozíku, které obsahuje dva stupně volnosti a jeden akční zásah. V mnoha případech se kyvadla zjednodušují do tvaru, kdy celá hmotnost kyvadla je umístěna na konci a je zavěšena na nehmotném závěsu. Pro rostoucí počet ramen kyvadla poté úměrně roste počet pohybových rovnic potřebných k popsání dynamiky systému. U jednoduchého kyvadla na vozíku se jedná o dvě rovnice a s každým dalším ramenem se počet rovnic zvedá o jednu.

V našem případě se bude jednat o dvojité inverzní kyvadlo, které se skládá ze dvou ramen a vozíku. Ramena budeme považovat za hmotná, s příslušným momentem setrvačnosti kolem těžiště. Jediná vstupní veličina je zde síla, která působí na vozík. Pohyb vozíku poté vyvolává pohyb ramen. Soustava je svázána pohybovými rovnicemi, které jsou odvozeny v kapitole 2.1.

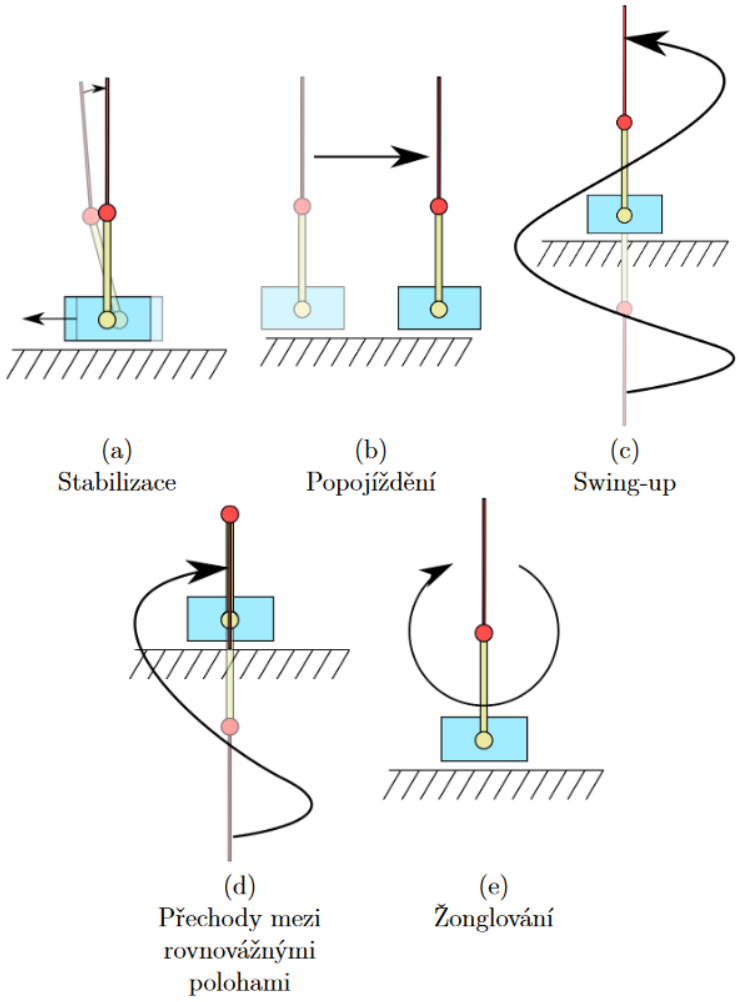
Dvojité inverzní kyvadlo obsahuje jednu stabilní rovnovážnou polohu Down/Down a tři nestabilní Up/Up, Up/Down a Down/Up.



Obrázek 3 – Rovnovážné polohy dvojitého kyvadla, převzato z [4]

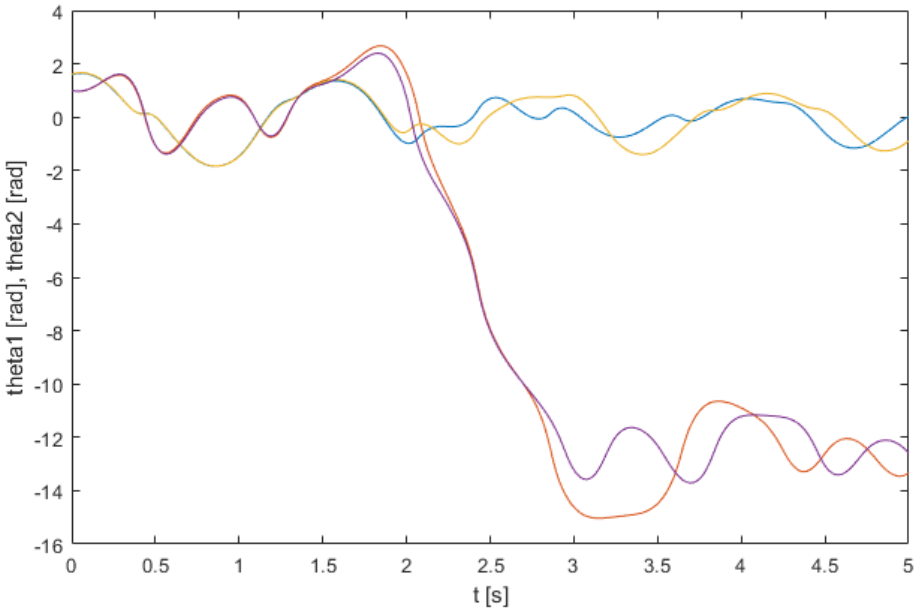
Cílem teorie řízení je nalézt takové regulátory, které soustavu stabilizují v nestabilních polohách. Další možností regulace je přechod mezi jednotlivými stavy. Nejznámější manévr je z polohy Down/Down do Up/Up, který se nazývá swing-up. Další pohyby, které lze vykonávat je popojíždění vozíku a žonglování.

Jedná se o nelineární systém a pro návrh regulátorů je třeba provést linearizaci kolem nestabilních poloh.



Obrázek 4 – Různé pohyby kyvadla, převzato z [4]

Dvojité inverzní kyvadlo je zároveň považováno za chaotický systém. Malá odchylka v počátečních podmínkách může vést na úplně odlišné trajektorie ve stavovém prostoru. Následující obrázek zachycuje změnu trajektorie při dvou pokusech. Počáteční poloha *theta1* byla v druhém pokusu změněna o tisícinu radiánu.



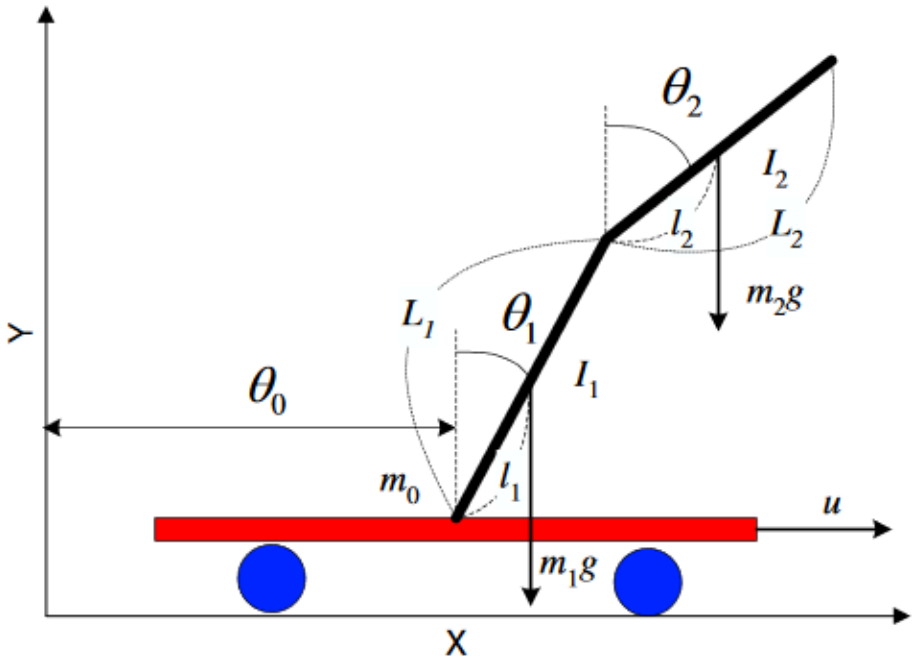
Obrázek 5 – Chaotické chování kyvadla

2.1 Dynamika dvojitého kyvadla

Při odvozování dynamiky budeme vycházet z Lagrangeových rovnic II. druhu.

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_r}\right) - \frac{\partial L}{\partial q_r} = Q_r$$
$$L = W_k - W_p \tag{2.1}$$

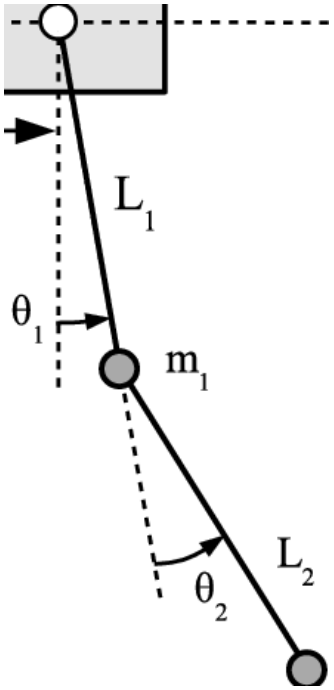
Kde q_r představuje zobecněné souřadnice, Q_r je zobecněná síla, což je průmět síly F do tečného směru k zobecněné souřadnici q_r. L je Lagrangián, jedná se o rozdíl kinetické a potenciální energie.



Obrázek 6 – Schéma dvojitého inverzního kyvadla, převzato z [6]

- θ_0 [m] - poloha vozíku (měřená hodnota), v našem případě značíme *x*
- θ_1 [rad] - úhel natočení prvního ramene (měřená hodnota)
- θ_2 [rad] - úhel natočení druhého ramene (měřená hodnota)
- L_1, L_2 [m] - délky ramen, v našem případě l_1, l_2
- l_1, l_2 [m] - vzdálenost k těžišti, v našem případě l_{1T}, l_{2T}
- M, m_1, m_2 [kg] - hmotnosti vozíku a ramen
- I_1, I_2 $[kg \cdot m^2]$ - momenty setrvačnosti kolem těžišť ramen
- g $[m \cdot s^{-2}]$ - tíhové zrychlení

V našem případě je pro odvození použito podobného modelu s tím rozdílem, že kladný úhel natočení θ_1 se počítá proti směru hodinových ručiček v poloze Down/Down a kladný úhel θ_2 je uvažován proti směru hodinových ručiček vzhledem k prvnímu rameni. Dále budeme značit T_1 jako těžiště prvního ramene a T_2 jako těžiště druhého ramene a M těžiště vozíku.



Obrázek 7 – Úhly natočení pro náš případ

Jako zobecněné souřadnice považujeme $x, \theta_{1,2}$. Globální polohy:

$$x_M = x$$
$$y_M = 0 \tag{2.2}$$

$$x_{T1} = x_M + l_{1T} \sin\theta_1$$
$$y_{T1} = -l_{1T} \cos\theta_1 \tag{2.3}$$

$$x_{T2} = x_M + l_1 \sin\theta_1 + l_{2T} \sin(\theta_1 + \theta_2)$$
$$y_{T2} = -l_1 \cos\theta_1 - l_{2T} \cos(\theta_1 + \theta_2) \tag{2.4}$$

Pro kinetické energie:

$$W_{kM} = \frac{1}{2} M {x_M}^2 = \frac{1}{2} M \dot{x}^2 \tag{2.5}$$

$$W_{kT1} = \frac{1}{2} m_1 (\dot{x_{T1}} + \dot{y_{T1}})^2 + \frac{1}{2} I_1 {\dot{\theta}_1}^2 \tag{2.6}$$

$$W_{kT2} = \frac{1}{2} m_2 (\dot{x_{T2}} + \dot{y_{T2}})^2 + \frac{1}{2} I_2 {\dot{\theta}_2}^2 \tag{2.7}$$

Pro potenciální energie:

$$W_{pM} = Mgy_M = 0 \tag{2.8}$$

$$W_{pT1} = m_1 g y_{T1} = -m_1 g l_{1T} \cos\theta_1 \tag{2.9}$$

$$W_{pT2} = m_2 g y_{T2} = -m_2 g (l_1 \cos\theta_1 - l_{2T} \cos(\theta_1 + \theta_2)) \tag{2.10}$$

Lagrangián pak bude:

$$L = W_{kM} + W_{kT1} + W_{kT2} - \left(W_{pM} + W_{pT1} + W_{pT2}\right) \tag{2.11}$$

Pohybové rovnice dostáváme řešením:

$$\begin{aligned} \frac{d}{dt}\left(\frac{\partial L}{\partial \dot{x}}\right) - \frac{\partial L}{\partial x} &= F_x - b_1 \dot{x} \\ \frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\theta_1}}\right) - \frac{\partial L}{\partial \theta_1} &= -b_2 \dot{\theta}_1 \\ \frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\theta_2}}\right) - \frac{\partial L}{\partial \theta_2} &= -b_3 \dot{\theta}_2 \end{aligned} \tag{2.12}$$

Výsledný tvar rovnic:

$$\begin{aligned} &\ddot{x}(M + m_1 + m_2) + \ddot{\theta_1} \cos\theta_1 (m_1 l_{1T} + m_2 l_1) - \dot{\theta_1}^2 \sin\theta_1 (m_1 l_{1T} + m_2 l_1) + \\ &m_2 l_{2T} \cos(\theta_1 + \theta_2)\left(\ddot{\theta_1} + \ddot{\theta_2}\right) - m_2 l_{2T} \sin(\theta_1 + \theta_2)\left(\dot{\theta_1} + \dot{\theta_2}\right)^2 = F_x - b_1 \dot{x} \end{aligned} \tag{2.13}$$

$$\begin{aligned} &\ddot{x}(m_1 l_{1T} \cos\theta_1 + m_2 l_1 \cos\theta_1 + m_2 l_{2T} \cos(\theta_1 + \theta_2)) + \ddot{\theta_1}\left(m_1 {l_{1T}}^2 + I_1 + \right. \\ &m_2 {l_1}^2 + m_2 {l_{2T}}^2 + 2 m_2 l_1 l_{2T} \cos\theta_2 + I_2\left.\right) + \ddot{\theta_2}\left(m_2 {l_{2T}}^2 + m_2 l_1 l_{2T} \cos\theta_2 + \right. \\ &I_2\left.\right) + \dot{\theta_1}\left(-2 m_2 l_1 l_{2T} \dot{\theta_2} \sin\theta_2\right) - m_2 l_1 l_{2T} {\dot{\theta_2}}^2 \sin\theta_2 + m_1 g l_{1T} \sin\theta_1 + \\ &m_2 g l_1 \sin\theta_1 + m_2 g l_{2T} \sin(\theta_1 + \theta_2) = -b_2 \dot{\theta}_1 \end{aligned} \tag{2.14}$$

$$\begin{aligned} &\ddot{x}(m_2 l_{2T} \cos(\theta_1 + \theta_2)) + \ddot{\theta_1}\left(m_2 {l_{2T}}^2 + m_2 l_1 l_{2T} \cos\theta_2 + I_2\right) + \ddot{\theta_2}\left(I_2 + \right. \\ &m_2 {l_{2T}}^2\left.\right) + m_2 l_1 l_{2T} {\dot{\theta_1}}^2 \sin\theta_2 + m_2 g l_{2T} \sin(\theta_1 + \theta_2) = -b_3 \dot{\theta}_2 \end{aligned} \tag{2.15}$$

2.2 Linearizace pohybových rovnic

Linearizace systému je potřebná pro návrh regulátoru.

Při linearizaci rovnic budeme vycházet z následujících zjednodušení:

$$\begin{aligned} \sin\theta &\approx \theta \; pro \; \theta \; blízké \; nule \\ \sin\theta &\approx (-\theta) \; pro \; \theta \; blízké \; \pi \\ \cos\theta &\approx 1 \; pro \; \theta \; blízké \; nule \\ \cos\theta &\approx -1 \; pro \; \theta \; blízké \; \pi \end{aligned} \tag{2.16}$$

Zároveň zde zavedeme konstanty, které nám pomůžou v kompaktnějším zápise.

$$\begin{aligned} k_1 &= M + m_1 + m_2 \\ k_2 &= m_1 {l_{1T}}^2 + I_1 + m_2 {l_1}^2 + m_2 {l_{2T}}^2 + 2m_2 l_1 l_{2T} + I_2 \\ k_3 &= I_2 + m_2 {l_{2T}}^2 \\ k_4 &= m_1 l_{1T} + m_2 l_1 + m_2 l_{2T} \\ k_5 &= m_2 {l_{2T}}^2 + m_2 l_1 l_{2T} + I_2 \\ k_6 &= m_1 g l_{1T} + m_2 g l_1 + m_2 g l_{2T} \\ k_7 &= m_2 g l_{2T} \\ k_8 &= m_2 l_{2T} \\ k_9 &= 2m_2 l_1 l_{2T} \end{aligned} \tag{2.17}$$

2.2.1 Poloha Down/Down

V této poloze linearizujeme pro úhly $\theta_1 = 0\ rad$, $\theta_2 = 0\ rad$.

$$\begin{aligned} \sin\theta_1 &\approx \theta_1 \\ \cos\theta_1 &\approx 1 \\ \cos\theta_2 &\approx 1 \\ \dot{\theta_{1,2}}^2 &\approx 0 \\ \cos(\theta_1 + \theta_2) &\approx 1 \end{aligned} \tag{2.18}$$

Linearizované rovnice:

$$\ddot{x}(M + m_1 + m_2) + \ddot{\theta}_1(m_1 l_{1T} + m_2 l_1) + m_2 l_{2T}(\ddot{\theta}_1 + \ddot{\theta}_2) = F_x - b_1 \dot{x} \tag{2.19}$$

$$\begin{aligned}&\ddot{x}(m_1 l_{1T} + m_2 l_1 + m_2 l_{2T}) + \ddot{\theta}_1\left(m_1 {l_{1T}}^2 + I_1 + m_2 {l_1}^2 + m_2 {l_{2T}}^2 + \right.\\ &\left.2m_2 l_1 l_{2T} + I_2\right) + \ddot{\theta}_2\left(m_2 {l_{2T}}^2 + m_2 l_1 l_{2T} + I_2\right) + m_1 g l_{1T}\theta_1 + m_2 g l_1 \theta_1 + \\ &m_2 g l_{2T}(\theta_1 + \theta_2) = -b_2 \dot{\theta}_1\end{aligned} \tag{2.20}$$

$$\begin{aligned}&\ddot{x}(m_2 l_{2T}) + \ddot{\theta}_1\left(m_2 {l_{2T}}^2 + m_2 l_1 l_{2T} + I_2\right) + \ddot{\theta}_2\left(I_2 + m_2 {l_{2T}}^2\right) + \\ &m_2 g l_{2T}(\theta_1 + \theta_2) = -b_3 \dot{\theta}_2\end{aligned} \tag{2.21}$$

Rovnice můžeme zapsat do maticového tvaru s využitím konstant (2.17).

$$\begin{bmatrix} k_1 & k_4 & k_8 \\ k_4 & k_2 & k_5 \\ k_8 & k_5 & k_3 \end{bmatrix} \begin{bmatrix} \ddot{x} \\ \ddot{\theta}_1 \\ \ddot{\theta}_2 \end{bmatrix} = \begin{bmatrix} 0 & -b_1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -k_6 & -b_2 & -k_7 & 0 \\ 0 & 0 & -k_7 & 0 & -k_7 & -b_3 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ \theta_1 \\ \dot{\theta}_1 \\ \theta_2 \\ \dot{\theta}_2 \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} F_x \tag{2.22}$$

Označíme:

$$\boldsymbol{K_1} \begin{bmatrix} \ddot{x} \\ \ddot{\theta}_1 \\ \ddot{\theta}_2 \end{bmatrix} = \boldsymbol{K_2} \begin{bmatrix} x \\ \dot{x} \\ \theta_1 \\ \dot{\theta}_1 \\ \theta_2 \\ \dot{\theta}_2 \end{bmatrix} + \boldsymbol{K_3} F_x \tag{2.23}$$

Pro druhé derivace bude platit (vynásobení inverzní maticí ke K_1 zleva):

$$\begin{bmatrix} \ddot{x} \\ \ddot{\theta}_1 \\ \ddot{\theta}_2 \end{bmatrix} = \boldsymbol{K_1}^{-1} \boldsymbol{K_2} \begin{bmatrix} x \\ \dot{x} \\ \theta_1 \\ \dot{\theta}_1 \\ \theta_2 \\ \dot{\theta}_2 \end{bmatrix} + \boldsymbol{K_1}^{-1} \boldsymbol{K_3} F_x = \begin{bmatrix} f_1 \\ f_2 \\ f_3 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ \theta_1 \\ \dot{\theta}_1 \\ \theta_2 \\ \dot{\theta}_2 \end{bmatrix} + \begin{bmatrix} g_1 \\ g_2 \\ g_3 \end{bmatrix} F_x \tag{2.24}$$

Kde $f_{1,2,3}$ jsou řádkové vektory a $g_{1,2,3}$ skaláry.

Linearizovaný systém poté můžeme zapsat jako:

$$\frac{d}{dt} \begin{bmatrix} x \\ \dot{x} \\ \theta_1 \\ \dot{\theta}_1 \\ \theta_2 \\ \dot{\theta}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ & & & f_1 & & \\ 0 & 0 & 0 & 1 & 0 & 0 \\ & & & f_2 & & \\ 0 & 0 & 0 & 0 & 0 & 1 \\ & & & f_3 & & \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ \theta_1 \\ \dot{\theta}_1 \\ \theta_2 \\ \dot{\theta}_2 \end{bmatrix} + \begin{bmatrix} 0 \\ g_1 \\ 0 \\ g_2 \\ 0 \\ g_3 \end{bmatrix} F_x \tag{2.25}$$

Pro rovnici výstupu:

$$y = \boldsymbol{C}\begin{bmatrix} x \\ \dot{x} \\ \theta_1 \\ \dot{\theta_1} \\ \theta_2 \\ \dot{\theta_2} \end{bmatrix} + \mathbf{0}F_x \tag{2.26}$$

Kde $\boldsymbol{C}$ je jednotková matice, jsou měřeny polohy, rychlosti budou odvozeny numerickou derivací.

2.2.2 Poloha Up/Up

Úhly $\theta_1 = \pi$ *rad,* $\theta_2 = 0$ *rad.*

$$\begin{gathered} \sin\theta_1 \approx -\theta_1 \\ \cos\theta_1 \approx -1 \\ \cos\theta_2 \approx 1 \\ \sin\theta_2 \approx \theta_2 \\ \dot{\theta_{1,2}}^2 \approx 0 \\ \cos(\theta_1 + \theta_2) \approx -1 \\ \sin(\theta_1 + \theta_2) \approx -(\theta_1 + \theta_2) \end{gathered} \tag{2.27}$$

Linearizované rovnice v maticovém tvaru:

$$\begin{bmatrix} k_1 & -k_4 & -k_8 \\ -k_4 & k_2 & k_5 \\ -k_8 & k_5 & k_3 \end{bmatrix}\begin{bmatrix} \ddot{x} \\ \ddot{\theta_1} \\ \ddot{\theta_2} \end{bmatrix} = \begin{bmatrix} 0 & -b_1 & 0 & 0 & 0 & 0 \\ 0 & 0 & k_6 & -b_2 & k_7 & 0 \\ 0 & 0 & k_7 & 0 & k_7 & -b_3 \end{bmatrix}\begin{bmatrix} x \\ \dot{x} \\ \theta_1 \\ \dot{\theta_1} \\ \theta_2 \\ \dot{\theta_2} \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}F_x \tag{2.28}$$

Další postup je stejný jako v předchozím případě. V praktické části se o výpočty stará vytvořený skript podle výše odvozeného. Podobně lze linearizovat i polohy Up/Down a Down/Up.

3 METODY PRO REGULACI INVERZNÍHO KYVADLA

Po tom, co jsme obdrželi linearizovaný tvar pohybových rovnic, můžeme napsat lineární časově invariantní stavový model systému:

$$\begin{aligned} \dot{x}(t) &= \boldsymbol{A}x(t) + \boldsymbol{B}u(t) \\ y(t) &= \boldsymbol{C}x(t) + \boldsymbol{D}u(t) \end{aligned} \tag{3.1}$$

Kde *x* představuje stavy systému, *u* je vstup do systému (působící síla na vozík), *y* je výstup systému, ***A*** je matice systému, ***B*** je vstupní váhová matice (matice řízení), ***C*** je výstupní váhová matice a ***D*** je přímovazební (feedforward) matice.

3.1 Stabilizace inverzního kyvadla

3.1.1 Pole Placement

Metoda Pole Placement, označovaná jako PP nebo Přiřazení pólů je nejjednodušší z uvedených metod. Zároveň se jedná o nejméně účinnou metodu z několika důvodů. Přiřazení pólů se snaží uvažovanému systému vnutit chování pomocí posunutí vlastních čísel (pólů) matice systému do stabilní oblasti podle rovnice (1.8). Systém musí mít kontrolovatelné všechny stavy, aby mohlo být požadované chování systému vnuceno. Dále předpokládáme, že máme k dispozici hodnoty všech stavů (celý stavový vektor).

Pro nalezení vhodného chování systému je potřeba množství experimentů. K nalezení výsledných hodnot regulátoru je možno využít např. Ackermannovy formule. [7,8]

3.1.2 LQR a LQG

LQR (Linear-Quadratic-Regulator) a LQG (Linear-Quadratic-Gaussian) jsou dva přístupy návrhu regulátoru, které mají systematický způsob určování žádaného chování systému. Způsob návrhu regulátoru se pro oba případy nijak neliší. Rozdíl je pouze v použití pozorovatele stavu. U LQR lze použít libovolného pozorovatele. U LQG se bavíme pouze o Kalmanově filtru - optimální pozorovatel stavu. Pokud tedy spojíme optimální regulátor LQR a optimálního pozorovatele KF dostáváme řízení LQG.

Nevýhodou LQG může být malá (až žádná) robustnost výsledného regulovaného obvodu. John C. Doyle ve svém známém článku „*Guaranteed Margins for LQG Regulators*“ v abstraktu napsal: „*There are none.*“.

Návrh LQR regulátoru spočívá v nalezení matice **K**:

$$u(t) = -\boldsymbol{K}x(t) \tag{3.2}$$

Matice **K** musí být taková, která minimalizuje kvadratické kritérium:

$$J = \int_0^\infty (x^T(t)\boldsymbol{Q}x(t) + u^T(t)\boldsymbol{R}u(t))dt \tag{3.3}$$

Kde $\boldsymbol{Q}$ je váhová matice stavů a $\boldsymbol{R}$ je váhová matice vstupů, tyto matice nám slouží k určení důležitosti jednotlivých stavů a případného omezení akčního zásahu. Podrobné odvození není předmětem této práce, ale řešení vede na tzv. Riccatiho maticovou rovnici:

$$\begin{gathered} \boldsymbol{PA} + \boldsymbol{A}^T\boldsymbol{P} - \boldsymbol{PBR}^{-1}\boldsymbol{B}^T\boldsymbol{P} + \boldsymbol{Q} = \boldsymbol{0} \\ \boldsymbol{K} = \boldsymbol{R}^{-1}\boldsymbol{B}^T\boldsymbol{P} \end{gathered} \tag{3.4}$$

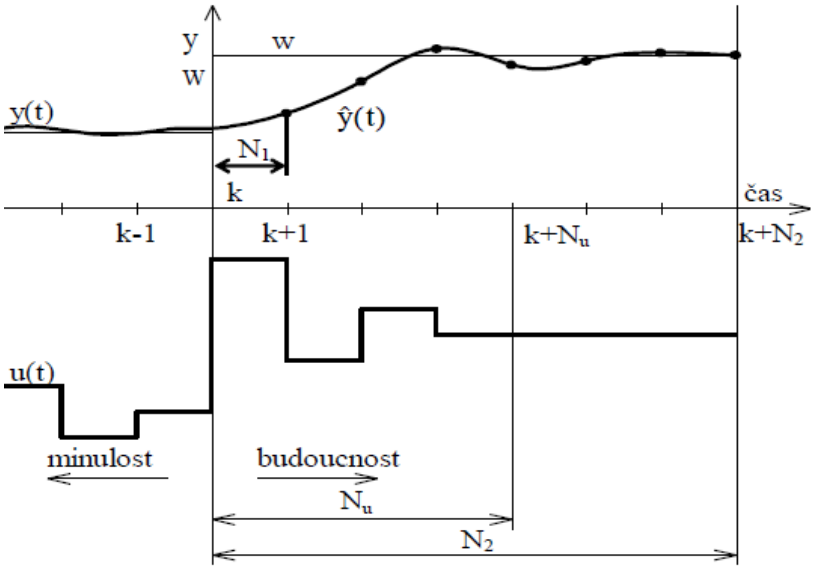
Kde **P** představuje řešení Riccatiho rovnice. Postup odvození lze nalézt v [8].

Váhové matice se obvykle volí jako diagonální a někdy je potřeba několika iterací k nalezení vhodného řešení. [6, 9]

3.1.3 Prediktivní řízení

Prediktivní řízení, nebo také MPC (Model Predictive Control) vychází z diskrétních modelů systémů. MPC využívá znalosti dynamiky řízeného systému, která slouží k optimalizaci budoucích akčních zásahů pro dosažení žádané hodnoty. Počítá tedy predikci budoucích výstupních hodnot s ohledem na dynamiku systému a akční zásah. Podobně jako u LQR zde probíhá minimalizace vhodného kritéria.

Celé řízení pak spočívá v tom, že je predikováno chování do budoucna na základě několika akčních zásahů, ale pro řízení se použije jen první zásah a celý proces se opakuje. Následující obrázek zachycuje průběh MPC. N_1 (minimální horizont) představuje počet přeskočených kroků výstupní veličiny při optimalizaci. Obvykle zůstává na hodnotě 1 (bez přeskakování). Vhodnou volbou N_1 můžeme zamezit problémům u systémů s dopravním zpožděním a také u neminimálně fázových systémů. N_2 (maximální horizont) představuje hranici, do které počítáme budoucí predikce výstupu. Hodnota N_2 by měla pokrývat téměř celý přechodový děj. N_u (řídicí horizont) ovlivňuje počet akčních zásahů, pro které se celý pochod optimalizuje. N_u nám také určuje výpočtovou náročnost MPC. [10]



Obrázek 8 – Průběh a parametry prediktivního řízení, převzato z [10]

Kvadratické kritérium pro diskrétní systém:

$$J = (\hat{y} - w)^T(\hat{y} - w) + \lambda\tilde{u}^T\tilde{u} \tag{3.5}$$

$$\hat{y}^T = [\hat{y}(k+1), \hat{y}(k+2), \dots \hat{y}(k+N_2)] \tag{3.6}$$

$$\tilde{u}^T = [\Delta u(k), \Delta u(k+1), \dots \Delta u(k+N_u)] \tag{3.7}$$

$$w^T = [w(k+1), w(k+2), \dots w(k+N_2)] \tag{3.8}$$

Kde $\hat{y}$ je vektor predikovaných výstupů, $\tilde{u}$ je vektor změn akčního zásahu, w je vektor budoucích hodnot žádané veličiny.

Pokud neznáme budoucí trajektorii žádané veličiny, volí se konstantní od současné hodnoty.

Odvození prediktoru je možno nalézt v [10]. Aplikace pro řízení dvojitého kyvadla pak v [11].

3.1.4 Ostatní možnosti stabilizace

Ke stabilizaci lze použít další metody, které v některých případech vycházejí z předcházejících kapitol nebo naopak volí úplně jiný přístup. Řízení spoléhající na jinou strategii je použití neuronových sítí [6]. NN (Neural Networks) je možno použít i pro řízení nelineárních systému. Nevýhodou bývá delší doba návrhu regulátoru a případného učení neuronové sítě pro danou úlohu. Zároveň se jedná o tzv. „černou skříňku“. Při poruše nelze jednoduše najít příčina problému a nemůžeme analyticky ověřit vlastnosti systému.

Další oblastí, která se dostává do popředí je použití strojového učení (ML – Machine Learning, technika RL - Reinforcement Learning). Jedná se o způsob, kdy se snažíme naučit agenta strojového učení volit správné akční zásahy s ohledem na předem naprogramovaný systém odměn. Více o strojovém učení je rozebráno v kapitole 3.2.2.

Některé metody pak spojují různé techniky, např. neuronové sítě a fuzzy řízení [12] nebo se jedná o známý algoritmus, ale parametry pro výpočet regulátoru jsou hledány např. evolučními technikami nebo jinými optimalizačními algoritmy (LQR a rojení částic (PSO)) [13].

3.2 Swing-up

Klasický manévr, který spadá do oblasti zájmu teorie řízení. Obsahuje několik složitostí, se kterými se musí navrhnuté techniky vypořádat. U dvojitého inverzního kyvadla se jedná o soustavu velmi citlivou na nepřesnosti v modelu. Zároveň je to systém nelineární a i když jsme schopni ho linearizovat kolem rovnovážných stavů, pořád při manévru prochází velmi nelineární oblastí.

Některé metody fungují pouze v simulacích a neexistuje jejich ověření implementací na reálné modely.

Klasickým swing-up technikám zde není věnována velká část, neboť v praktické části je realizován za pomoci strojového učení.

3.2.1 Klasické metody

Nelineární prediktivní řízení - nebylo donedávna aplikováno pro svou velkou výpočtovou náročnost a vysokou nelinearitu kyvadla. NMPC (nelineární MPC) musí v reálném čase řešit tzv. two-point boundary value problem (řešení diferenciálních rovnic musí splňovat počáteční a koncové podmínky). Využitím NMPC pro swing-up se zabývá [14].

Řízení energie – metody řízení energie nespočívají v řízení stavových veličin, ale řízení celkové mechanické energie systému. Žádaná hodnota celkové energie systému je potenciální energie systému v poloze Up/Up. Regulátor je poté nastaven tak, aby do systému postupně dodával energii podle zvoleného přístupu. Jednu z možných metod nabízí [17].

Feedforward a Feedback - v tomto způsobu se jedná o nalezení takové časové posloupnosti vstupu, která systém převede z dolní do inverzní polohy. Způsobu, který využívá předem určenou trajektorii k dosažení cíle bez zpětné vazby, se nazývá Feedforward (dopředné)

řízení. Takový způsob řízení ovšem není vhodný pro reálné soustavy. Hraje zde roli nepřesná identifikace (model) systému, působení vnějších vlivů a poruch. Proto se v takovém případě zavádí ještě Feedback (zpětná vazba), která má za úkol hlídat (a kompenzovat) průběh celého děje. Máme tedy naplánovanou trajektorii výpočtem a zavádíme zpětnovazební řízení k tomu, abychom se od této trajektorie nevzdálili. Aplikací těchto technik nalezneme v [15, 16].

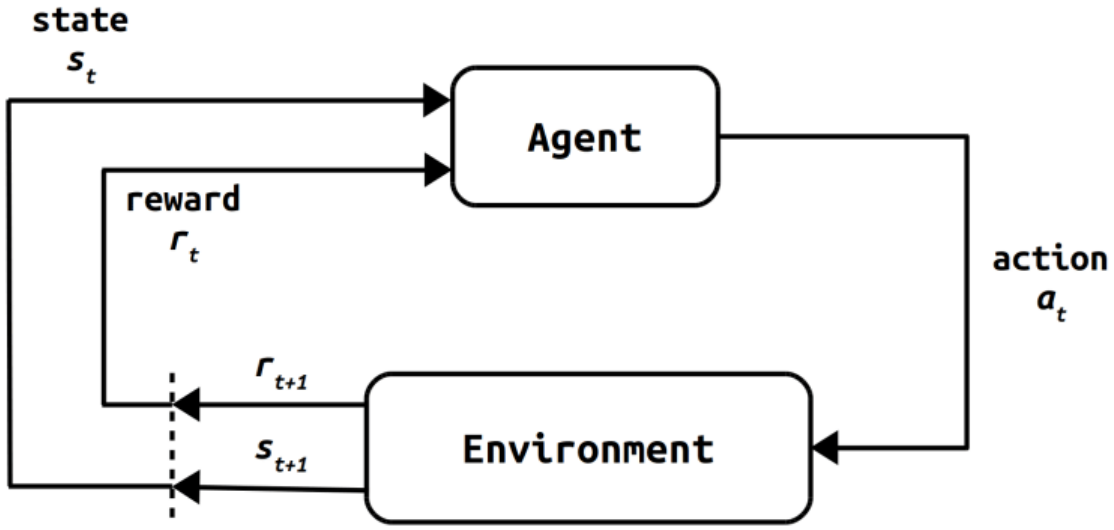
3.2.2 Strojové učení - Reinforcement Learning

Jedním z nových oborů rozmáhajících se ve všech odvětvích je strojové učení. Při využití strojového učení u swing-up manévru se budeme bavit o konkrétní technice, kterou je Reinforcement Learning (RL). V češtině toto označení nemá jednoznačný překlad, ale mohli bychom se bavit o „posíleném“ nebo „zpětnovazebním“ učení. V této kapitole se ovšem budu snažit o překlad ostatních termínů. Následující část bude popisovat základy, které byly potřebné k vytvoření swing-up řešení. Bližší povídání o RL lze nalézt v [18] a také Matlab dokumentaci, která byla využita pro vytváření simulací.

Jedná se o techniku, která využívá odměnu a trest za plnění určitého úkolu. Setkáváme se zde s několika pojmy:

- Agent (Agent) – část programu, která prozkoumává, interaguje a učí se od prostředí
- Akce (Action) – působení agenta na prostředí, prostředí působením této akce mění své stavy, okamžitá hodnota stavů produkuje odměnu za danou akci, z této znalosti agent upravuje své chování pro zisk větší odměny v budoucnu, tímto procesem se učí, akce může být ze spojitého intervalu nebo diskrétní množiny
- Prostředí (Enviroment) – mění svůj stav na základě působení agenta, výstupem prostředí je jeho stav a odměna, označuje se tak vše, co není samotným agentem (viz. *Obrázek 9*)
- Pozorování (Observation) – pozorovaný stav prostředí
- Politika (Policy) – součást agenta, která z pozorování prostředí rozhoduje o akci, která bude vykonána, často je využito neuronových sítí – z několika (i tisíce) vstupů (stavů) generuje výstupy (akční zásahy)
- Algoritmus zpětnovazebního učení (RL algorithm) – upravuje politiku na základě vykonané akce, pozorování stavu prostředí a odměny získané z prostředí, cílem RL je využití RL algoritmu tak, aby postupem času změnil politiku do podoby, kdy nehledě na stav prostředí agent vždy zvolí nejvýhodnější akci

- Odměna (Reward) – okamžitá hodnota odměny pro stav, ve kterém se systém nachází, měříme tak úspěch nebo neúspěch agenta
- Hodnota (Value) – celková odměna, kterou může agent očekávat z daného stavu do budoucna, oproti odměně udává dlouhodobou efektivitu místo okamžité, uplatňuje se zde sleva
- Sleva (Discount) – snižování budoucích odměn při počítání hodnoty (Value), bere v úvahu nejistotu modelu/prostředí, předpovídané odměny v budoucnu se zde nemusí nacházet, proto je jim přikládán menší a menší význam
- Prozkoumávání (Exploration) – agent upřednostňuje prozkoumávání odměny pro stavy, ve kterých ještě nebyl, je zde možnost nalezení nových stavů s velkou odměnou, ale také nevyužití již nabytých znalostí o prostředí
- Zneužití (Exploitation) – agent využívá již prozkoumané stavy, které nesou velké odměny, ale málo prozkoumává další možné řešení



Obrázek 9 – Schéma agenta a prostředí v RL, převzato z [18]

Agent nemusí vědět nic o prostředí a přes pozorování a odměnu se přesto naučí nejvýhodnější chování. Funkce odměny může být libovolná funkce, která motivuje agenta k požadovanému chování. Tato funkce může být při experimentech několikrát upravena tak, aby výsledný systém splňoval naše požadavky. Při návrhu odměny se setkáváme s tím, že agent nalezne nejvhodnější chování podle dané funkce, ale toto chování vůbec nesplňuje požadavky dané úlohy. Pokud necháme ve funkci zkratku (neošetříme určitý vzorec chování), RL algoritmus ji využije ke svému prospěchu. Často se jedná o „schované“ chování, které při počátečním návrhu nejsme schopni ověřit.

Síla využití neuronových sítí spočívá v jejich univerzálnosti aproximovat libovolnou funkci. Při správné volbě struktury sítě lze modelovat jakoukoliv vstupně/výstupní závislost. Musíme volit dostatečně složitou strukturu pro daný problém, ale ne příliš velkou, aby se problém nestal moc komplexním na vyřešení.

V našem případě bylo využito RL učení typu Actor-Critic. Actor (neuronová síť) se snaží zvolit nejlepší akční zásah s ohledem na stav, ve kterém se prostředí nachází. Critic (druhá neuronová síť) poté odhaduje očekávanou hodnotu (value) na základě stavu prostředí a akce, kterou zvolil Actor. Critic se tedy učí odhadovat očekávanou hodnotu na základě zpětné vazby z prostředí. Actor se poté učí na základě odezvy (doporučení) od Critic. Jedná se o optimalizační proces se dvěma neuronovými sítěmi.

Nevýhodou takového řešení je, že se jedná o tzv. „černou skříňku“. Nelze se podívat na důvod volby toho či onoho akčního zásahu. Nelze také jednoduše měnit naučené chování. Pokud bychom chtěli změnit chování, je třeba přepracovat prostředí a reward funkci, a následně provádět další učení. Tento proces může být velmi zdlouhavý. Dále je potřeba mít velmi přesný model prostředí, protože při nesprávném chování na reálné soustavě nelze jednoduše změnit parametry agenta podle požadavků. Zároveň nelze analyticky otestovat řešení na stabilitu nebo robustnost.

Řešením pro zvýšení robustnosti je variace parametrů prostředí při učení agenta v každé epizodě. Průběžným rozmítáním parametrů způsobíme to, že se agent nenaučí jedno chování pro dané prostředí, ale robustní chování pokrývající celou řadu kombinací parametrů. Aby zůstal problém řešitelný, musíme zvolit vhodné rozmítání, které je agent ještě schopný aproximovat v jedné síti.

II. PRAKTICKÁ ČÁST

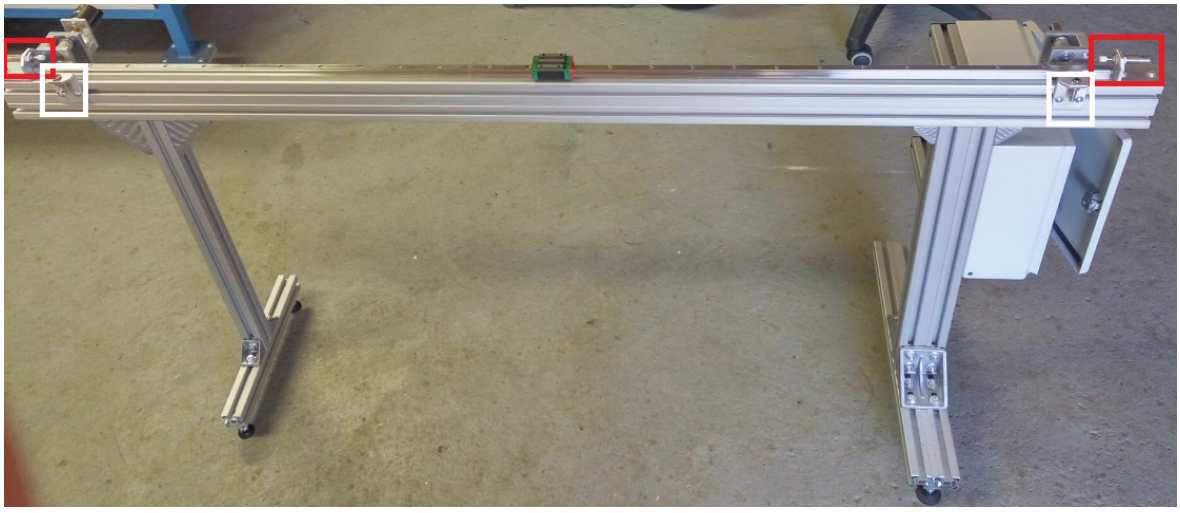
4 REÁLNÁ SOUSTAVA DVOJITÉHO KYVADLA

Sestava kyvadla byla vytvářena od úplného základu. Model vychází z klasického uspořádání dvojitého inverzního kyvadla na vozíku, který se pohybuje po kolejnici. Celý návrh zahrnoval - původní náčrty ramen, vozíku, umístění snímačů a jejich propojení, vytváření výkresové dokumentace k vyráběným dílům, 3D modelování a 3D tisk vlastních dílů, výběr a propojení elektroniky, montáž všech dílů, vytváření bezpečného napájení silnoproudem, návrh a realizace řízení pomoci Real-Time v Simulinku. Podrobnější popis vybraných částí obsahují následující podkapitoly.

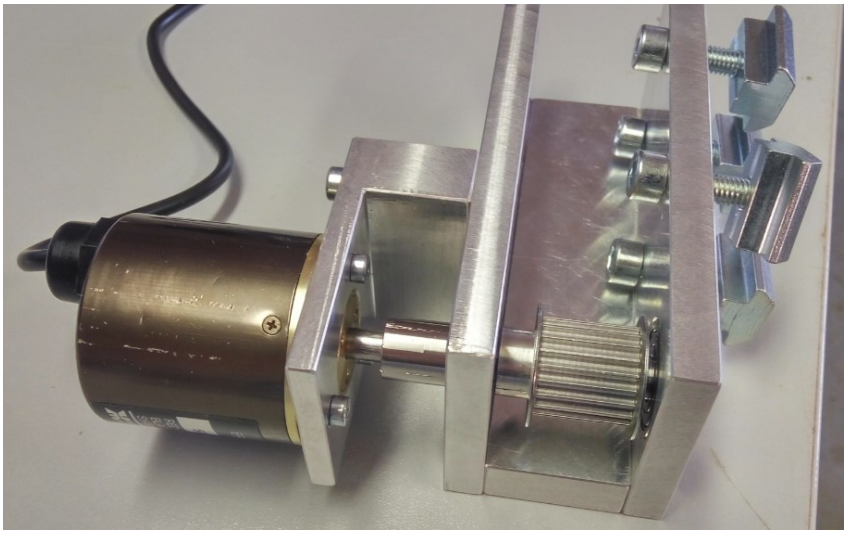
4.1 Mechanická konstrukce

4.1.1 Rám

Rám je vytvořen z hliníkových profilů Bosh (40x80 a 40x40 + spojovací materiál), délka kolejnice je 155 cm, výška od podlahy ke kolejnici 78 cm. Na koncích kolejnice jsou umístěny tlumiče, pro případ poruchových stavů (*Obrázek 10* - červeně označené). Před tlumiči jsou také indukční koncové snímače (*Obrázek 10* – bíle označené). Pokud na snímač přijede vozík (okraj vozíku), vypne se motor. Vzdálenost koncových snímačů se během vývoje měnila a finální délka mezi nimi je 133 cm. Efektivní délka, po které se může vozík pohybovat je ještě zkrácena o jeho šířku (12 cm).

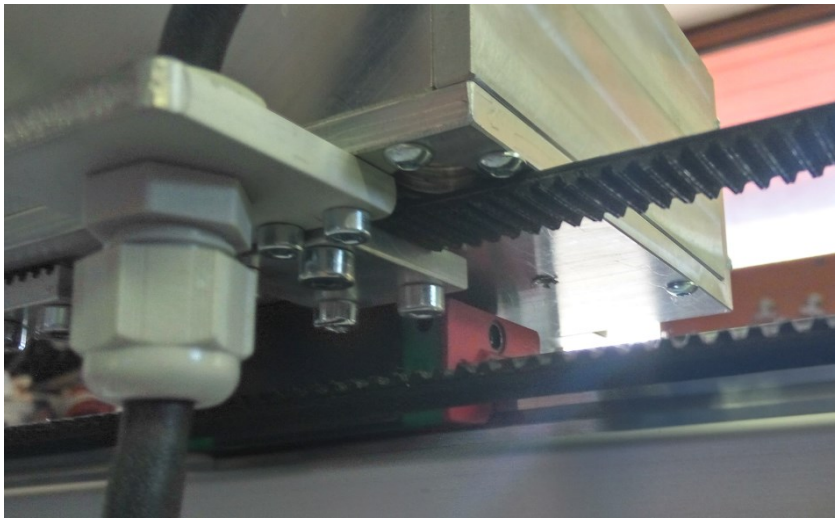


Obrázek 10 – Rám s kolejnicí, enkodérem, rozvaděčem, tlumiči a koncovými snímači



Obrázek 11 – Uchycení enkodéru a řemenice

Motor a enkodér jsou napojeny přes hřídele na řemenice o průměru 20 mm. Řemen je napojen na vozík ze spodní strany a na krajích prochází přes řemenice motoru a enkodéru. Typ řemene je RPP 3-15 mm.



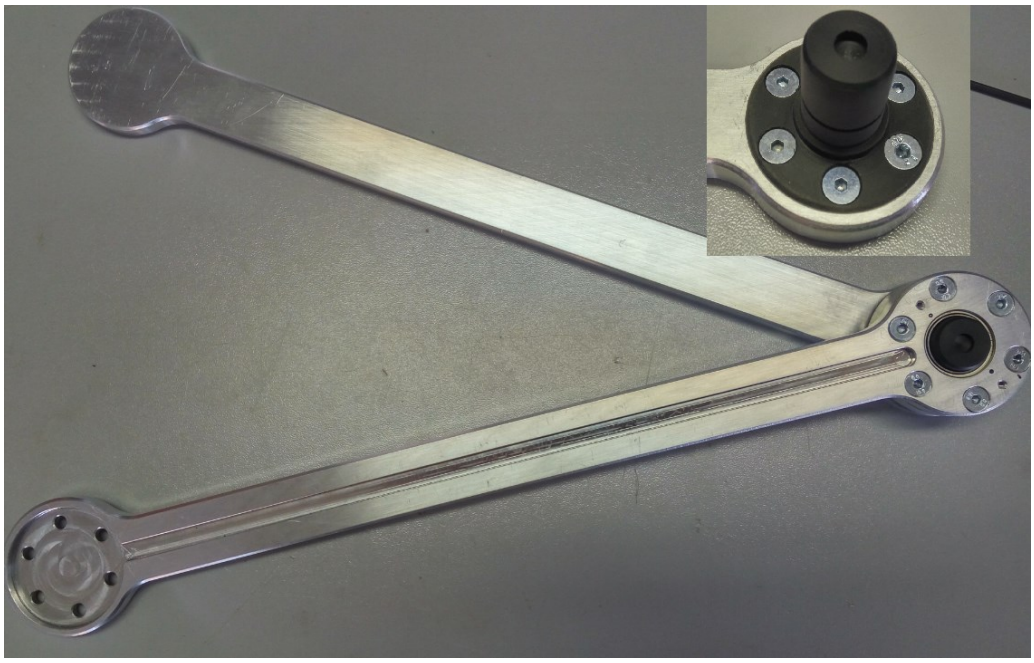
Obrázek 12 – Uchycení řemenu, spodní strana vozíku

4.1.2 Kyvadlo

Ramena kyvadla jsou vyrobena z hliníku, obě mají celkovou délku 346 mm a vzdálenost mezi středy otáčení obou ramen je 300 mm. Tloušťka ramen je 8 mm. Přesný technický výkres v příloze č. 1 a 2.

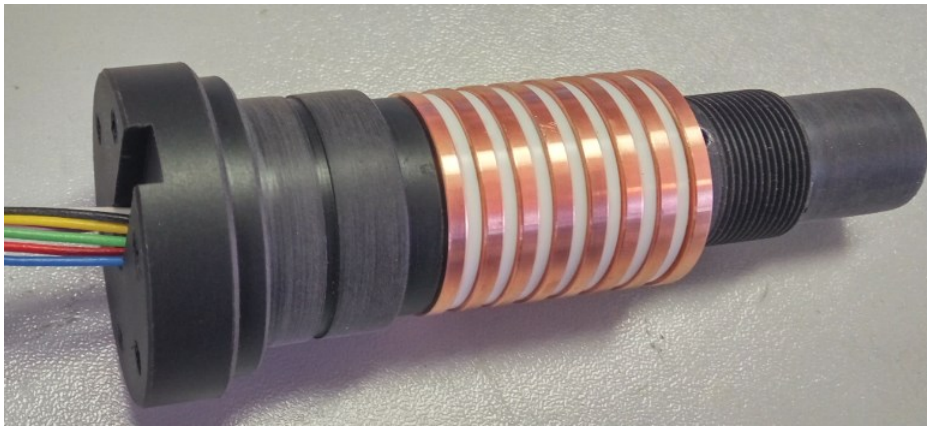
Spojení ramen je provedeno za pomoci plastového dílu s hřídelí na prvním rameni, na kterou je nasazeno druhé rameno přes ložisko a zajištěno pojistným kroužkem o průměru 20 mm.

První rameno obsahuje vybrání, které slouží k uložení vodičů pro komunikaci se snímačem natočení druhého ramene.



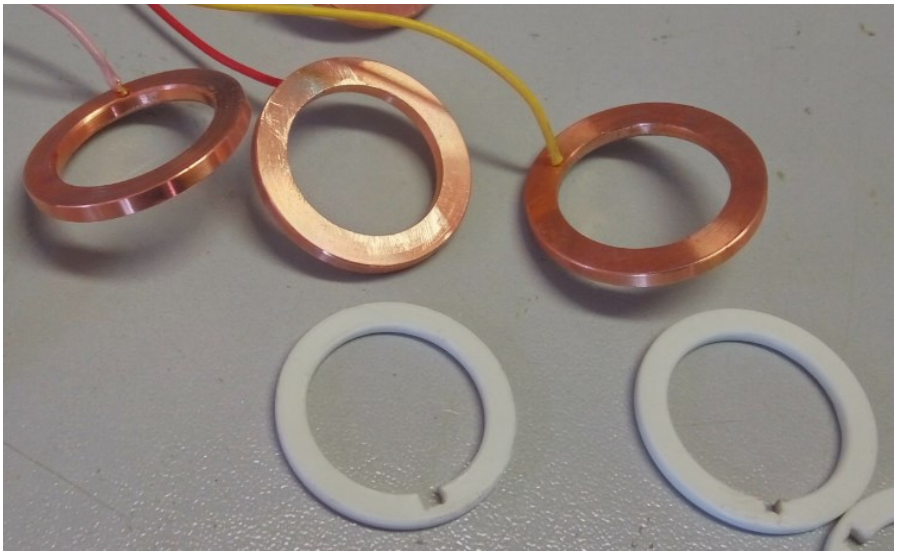
Obrázek 13 – Ramena kyvadla a detail na hřídel, na konci hřídele díra pro magnet

Největším problémem u návrhu dvojitého kyvadla bylo vyřešení kabeláže senzorů tak, aby neovlivňovaly dynamiku systému. Z tohoto důvodu bylo vymyšleno řešení, které umožňuje ramenům volně se otáčet bez jakéhokoliv omezení. Slouží k tomu vytvořené jádro (hřídel, materiál PA - Polyamid), které je pevně spojeno s prvním ramenem a jeho středem jsou vedeny vodiče k snímači úhlu natočení druhého ramene. Technický výkres v příloze č. 3.



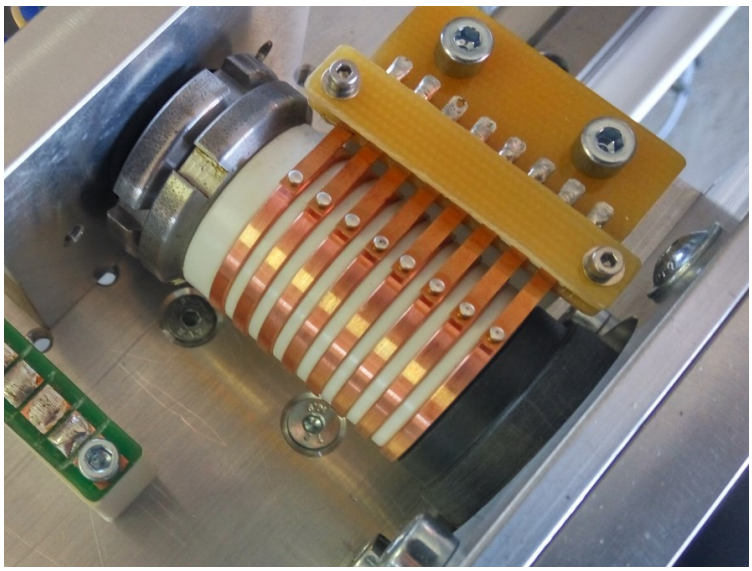
Obrázek 14 – Vytvořené jádro s vodiči

Na jádru jsou vyvedeny měděné kroužky, na které jsou jednotlivé vodiče napájeny a od sebe jsou odizolovány pomocí plastových kroužků (materiál PA). Jádro je uchyceno na vozíku pomoci dvou ložisek.



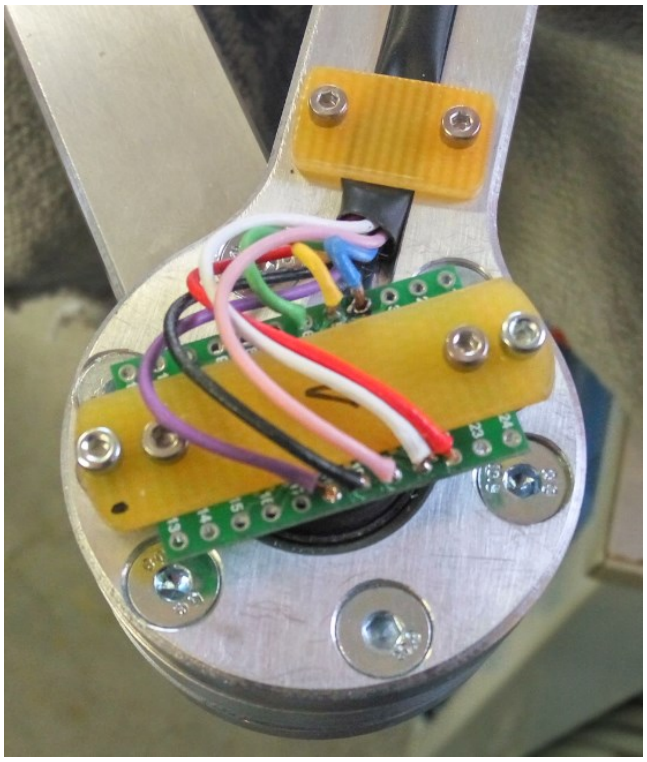
Obrázek 15 – Ukázka měděných a plastových kroužků s napájenými vodiči

Na měděné kroužky jádra doléhají (a jsou přitlačeny) měděné plíšky, které mají postříbřený kontakt a při rotaci jádra (prvního ramene) umožňují elektrické spojení se senzorem. Plíšky jsou přitlačeny tak, aby zbytečně nezvyšovaly koeficienty tření, ale dostatečně na to, aby při rotaci kyvadla nedocházelo k výpadkům signálu. K uchycení plíšků na vozík byla vytvořena a vytisknuta (3D tisk) vlastní součástka.



Obrázek 16 – Detail pro jádro a přitlačené plíšky

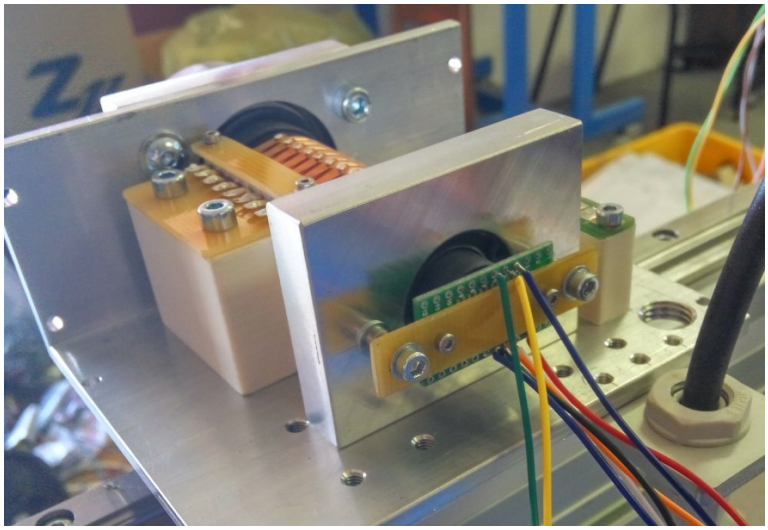
Musela být také vytvořena deska plošných spojů, na kterou byl snímač napájen. Na desku se poté pájely jednotlivé vodiče. Jelikož se jedná o snímač na bázi Hallova jevu, musela být mezi snímačem a magnetem (nalepen na konci hřídele) vytvořena mezera 1 - 2 mm. Více o snímači je uvedeno v kapitole 4.2.2.



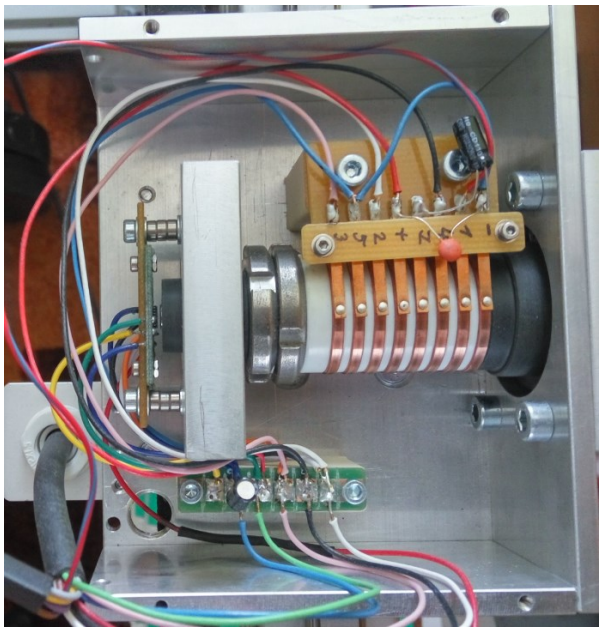
Obrázek 17 – Uchycení senzoru pro druhé rameno

4.1.3 Vozík

Vozík slouží k uchycení jádra ve dvou ložiscích, dále k upevnění snímače a umožňuje spojit příslušné vodiče tak, aby se přes vedený kabel dalo komunikovat s řídicí elektronikou. Na konci jádra je magnet, díky kterému snímač měří úhel natočení prvního ramena. Ze spodní strany je upevněn řemen, který přenáší sílu motoru na vozík (viz *Obrázek 12*). Vozík se pohybuje po kolejnicovém lineárním vedení HIWIN HG15.



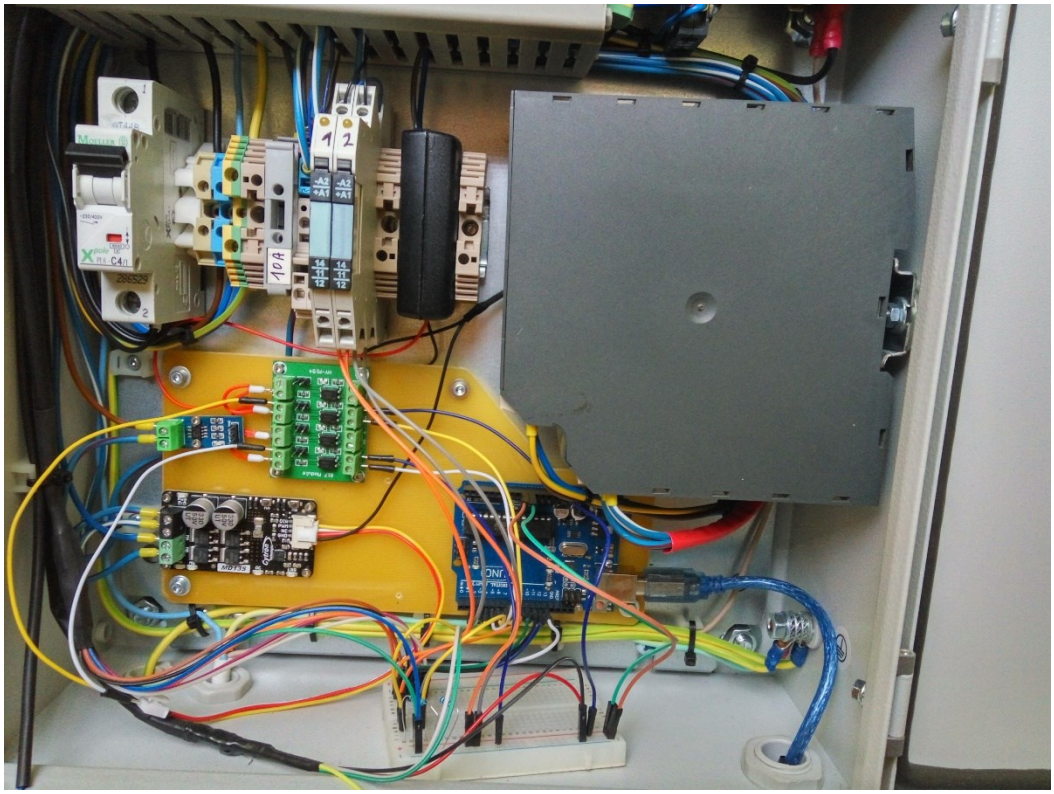
Obrázek 18 – Vozík bez krytu a bez propojení vodičů



Obrázek 19 – Propojení vodičů na vozíku

4.1.4 Rozvaděč

Na rozvaděči se nachází hlavní vypínač a signalizace stavu zařízení (pod napětím/vypnuto). Je zde umístěna veškerá elektronika a jediným výstupem z rozvaděče je USB kabel, který slouží k propojení Arduina a PC. Podrobné schémata zapojení se nachází v kapitole 4.2.9.



Obrázek 20 – Rozvržení elektrických částí v rozvaděči

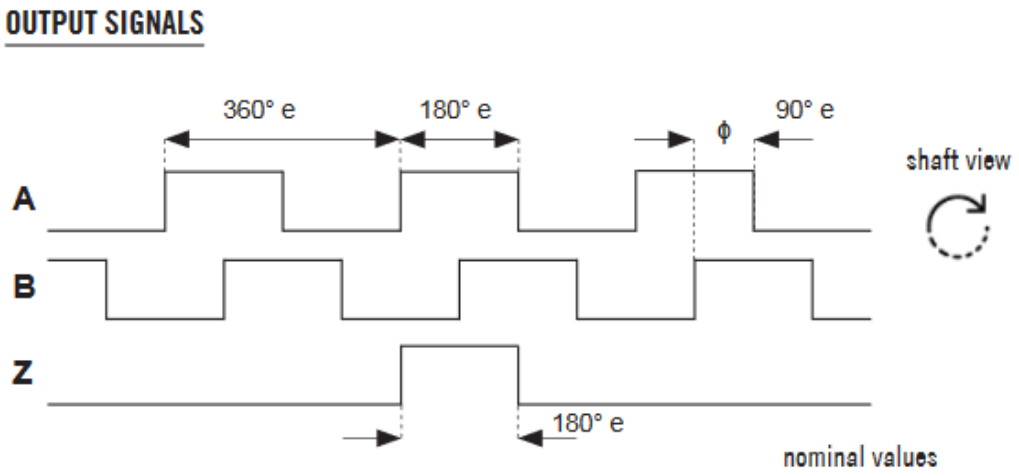
4.2 Elektronické součásti

4.2.1 Enkodér

Pro měření polohy vozíku byl použit starší inkrementální rotační enkodér od firmy Matsushita (od roku 2008 Panasonic). Rozlišení enkodéru je 500 PPR (Pulses Per Revolution, pulzy na otáčku). Zapojením signálů A, B tak získáváme 2000 CPR (Counts Per Revolution, čtení na otáčku). Enkodér je napájen z Arduina pomocí 5 Vdc a GND, signály A, B jsou připojeny na piny 2 a 3. Ze znalosti průměru řemenice a CPR dostáváme, že jeden pulz zde představuje přibližně 0,03 mm.

Obrázek 21 – Použitý rotační enkodér

Základní části optického enkodéru jsou optický vysílač (např. LED dioda), optický přijímač (fototranzistor, fotodioda) a neprůhledná kruhová mřížka s průhlednými otvory. Mechanickým otáčením hřídele enkodéru se otáčí i kruhová mřížka, přes jejíž průhledné části proniká elektromagnetické záření vysílače a na přijímači generuje pulzy. V našem případě se jedná o dvojici vysílačů a přijímačů, které generují 2 signály A, B. Díky tomu je možno určovat i směr otáčení.



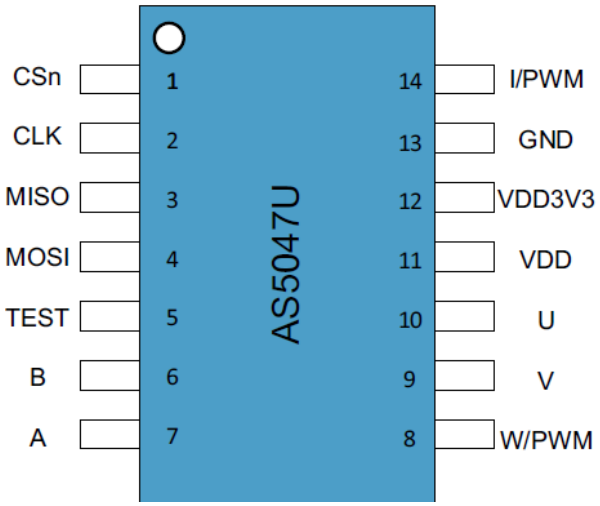
Obrázek 22 – Signály rotačního enkodéru [19]

4.2.2 Snímače úhlu

Jako snímače úhlu natočení ramen byly původně uvažovány mechanické rotační enkodéry (absolutní nebo inkrementální). Po průzkumu možností na trhu se ukázalo, že tyto snímače

jsou velmi rozměrné nebo mají nedostatečné rozlišení pro přesné měření. Enkodér splňující všechny požadavky by byl zase velmi drahý. Proto byla zvolena varianta s magnetickými rotačními senzory využívající Hallova jevu, které byly mnohem menší a levnější.

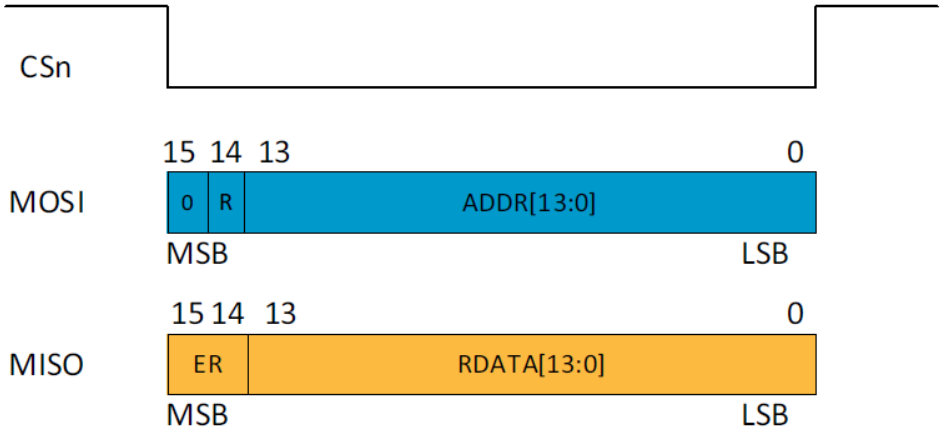
Typ senzoru byl vybrán AS5047U s 14-ti bitovým A/D převodníkem. Jedná se o absolutní senzor a rozděluje 360° na 16384 dílů. Dokáže rozlišit 0,022°. Data ze senzoru lze získávat buď rozhraním SPI nebo čítáním pulzů signálu ABI nebo PWM. V našem případě byla zvolena komunikace SPI.



Obrázek 23 – Schéma AS5047U [20]

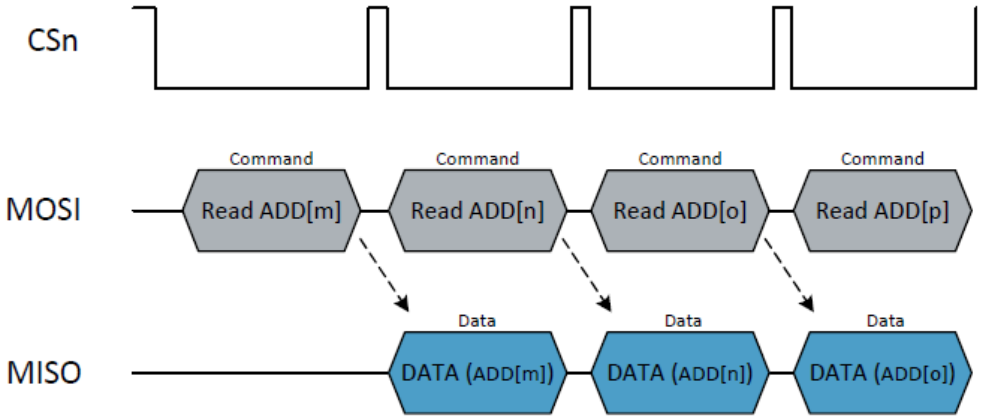
Senzor obsahuje pole Hallových sond, jejichž signál je zesílen a prochází filtrací. Poté dochází k převodu analogového signálu na digitální a tyto data jsou zpracovány ve výpočetní jednotce CORDIC (coordinate rotation digital computer) pro výpočet přesného úhlu a síly magnetického pole. Snímač automaticky kompenzuje vlivy teploty a odlišnosti ve vzdálenosti magnetu.

SPI komunikace umožňuje číst čistá data (16 bitů rámec) nebo využít CRC (24 nebo 32 bitů rámec). Pro naši úlohu je využito SPI s délkou rámce 16 bitů, které vypadají následovně.

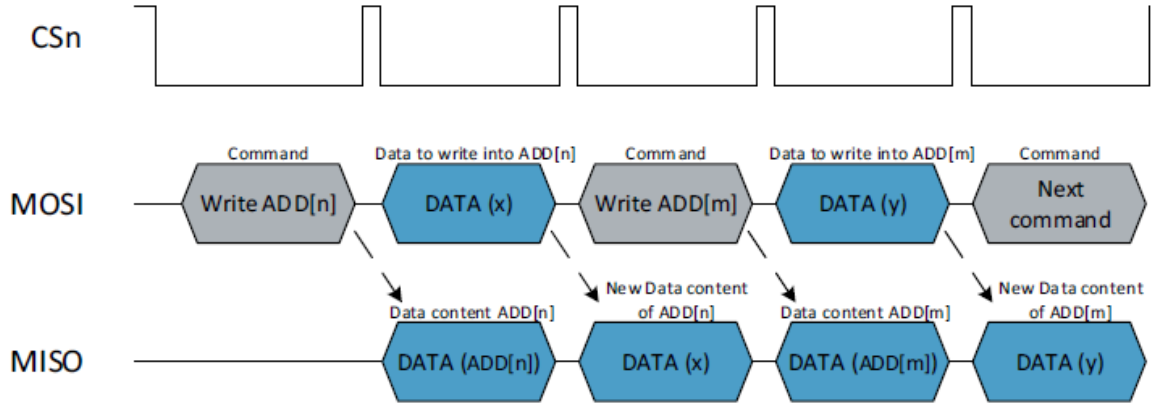


Obrázek 24 – SPI rámce [20]

CSn označuje signál sloužící k výběru senzoru. Oba senzory sdílejí komunikační sběrnici, ale každý má vlastní CSn pin. Rámec vyslaný k senzoru (příkazový, MOSI) obsahuje na bitech 0 - 13 adresu registru, se kterým chceme pracovat. Na bitu 14 se nachází požadavek ke čtení/zápisu (1 – čtení, 0 – zápis) a na hodnotě bitu 15 nezáleží. Rámec, který přijímáme (datový, MISO) obsahuje na bitech 0 - 13 data, a bity 14 a 15 slouží k informaci o chybě nebo varování (14 – error, 15 – warning).



Obrázek 25 – SPI Read (čtení) [20]



Obrázek 26 – SPI Write (zápis) [20]

Ve vytvořeném programu se používají hlavně následující registry:

Tabulka 1 – Vybrané paměťové registry [20]

Adresa registru	Název	Původní hodnota registru	Popis
0x0001	ERRFL	0x0000	Registr chyb
0x3FFE	ANGLEUNC	0x0000	Hodnota úhlu bez dynamické kompenzace
0x3FFF	ANGLECOM	0x0000	Hodnota úhlu s dynamickou kompenzací

V registru ERRFL představují jednotlivé bity různé chybové stavy (např. chybná vzdálenost magnetu, chyba v komunikaci, neplatný příkaz, CRC chyba). V registrech ANGLEUNC a ANGLECOM obsahují bity 0 - 13 přímou (absolutní) hodnotu úhlu natočení. Další registry, které zde nejsou uvedeny, soužily hlavně k původní diagnostice při raných fázích vývoje. Všechny dostupné registry a další informace o senzorech lze nalézt v dokumentaci na stránkách výrobce [20].

Jako magnety slouží neodymové válce průměru 6 mm a výšky 2,5 mm magnetované kolmo na osu. Přesný typ magnetu je VMM4H-N35H.

4.2.3 Indukční koncové snímače

Jako koncové snímače byly použity indukční snímače firmy SICK, typ IME08-04NPSZT0K, které jsou propojeny konektorem M8 (3 vodiče) přes relé [21]. Obvod se

snímači je na zdroji 24 Vdc a relé rozepíná napětí Arduina 5 Vdc. K detekci jsou využity piny 4 a 6 s použitím pull-down rezistorů.

4.2.4 Motor

Původně byl pro testování zvolen stejnosměrný uhlíkový motor z akumulátorové vrtačky. Toto řešení se ukázalo býti schůdným i pro výslednou sestavu, ale určitě to není řešení ideální. Zde výsledná reálná soustava trpí na kvalitě nejvíce a v budoucnu by to byla hlavní oblast pro vylepšení.

Tabulka 2 – Získané parametry DC motoru

Parametr	**Hodnota**
Otáčky [min^{-1}]	0-400 nebo 0-1500
Maximální točivý moment [Nm]	9,5
Proud bez zátěže [A]	2,2
Proud v zátěži [A]	8

Z otáček motoru a průměru řemenice lze přibližně vypočítat maximální rychlost vozíku:

$$o = \pi d = 3{,}14 \cdot 0{,}02 = 0{,}0628\, m$$

$$v_{max} = \frac{1500}{60} \cdot o = 1{,}57\, m/s$$

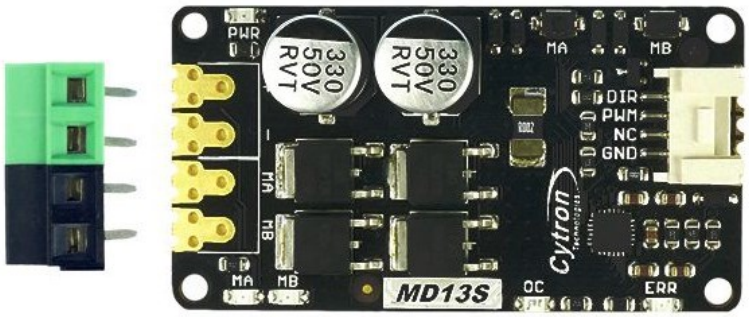
4.2.5 Řídicí jednotka motoru

Jako první byl použit H-můstek IBT-2 se kterým byly během testování velké problémy. Následně byl nahrazen jednotkou MD13S. Řídicí jednotka slouží k PWM řízení uhlíkových motorů do jmenovitého proudu 13 A s proudovými špičkami 30 A (do 10 sekund). Podporuje 6-30 Vdc napájení motoru. Na vstupu jsou 4 piny:

Tabulka 3 – Popis pinů MD13S [22]

Značení pinu	**Popis**
GND	Logická zem
NC	Not Connected – Nepřipojuje se
PWM	Vstup PWM pro řízení

DIR	Směr otáčení motoru



Obrázek 27 – Řídicí jednotka MD13S [22]

4.2.6 Optočlen

Řídicí jednotka je od Arduina oddělena ještě modulem, který obsahuje 4 kanály s optočleny PC817. Využity jsou dva kanály, jeden pro PWM a druhý pro DIR. Modul podporuje napětí 3,6 – 24 Vdc na vstupu a 3,6 – 30 Vdc na výstupu.

4.2.7 Arduino

Zpracování signálu na nejnižší úrovni zprostředkovává Arduino Uno Rev3. Obecně Arduino vyhodnocuje signál z enkodéru, dvou snímačů úhlu a posílá je po sériové lince (USB) do PC. Dále kontroluje koncové snímače a provádí případné zastavení motoru. PC pracuje v Real-Time režimu Simulinku a posílá vypočtené hodnoty akčního zásahu zpět do Arduina, kde je prováděno řízení motoru na úrovni PWM. Arduino je napájeno přes USB.

Tabulka 4 – Parametry vývojové desky Arduino Uno [23]

Parametr	**Hodnota**
Pracovní napětí	5 Vdc
Vstupní napětí (doporučeno)	7-12 Vdc
Vstupní napětí (mezní hodnoty)	6-20 Vdc
Digitální I/O piny	14 (z toho 6 PWM)
Počet analogových vstupů	6
Maximální proudové zatížení na 1 pin	20 mA

Frekvence hodin	16 MHz
Paměť pro program	32 kB
Paměť pro proměnné	2 kB

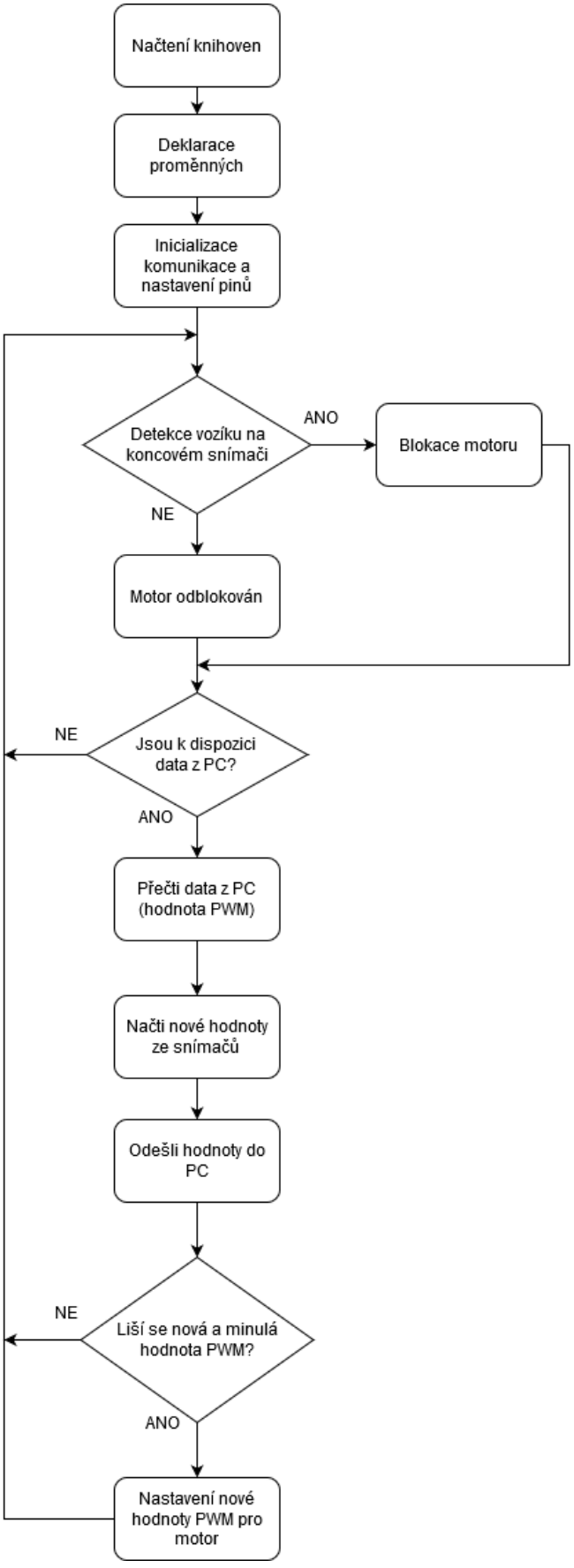
Program Arduina využívá knihoven *Encoder.h* k čítání pulzů enkodéru a *SPI.h* ke komunikaci se senzory.

Pro PWM řízení motoru lze využít funkci *analogWrite()*, která pracuje s hodnotami 0 – 255. Abychom pomocí jedné *unsigned* (bez znaménka) proměnné z PC řídili motor oběma směry, byla jako neutrální poloha zvoleno číslo 255. Pro čísla menší než 255 se motor otáčí proti směru hodinových ručiček (vozík směrem doleva) a pro čísla větší než 255 na opačnou stranu.

Za zmínku stojí komunikace se snímači úhlu, kde se přes SPI posílají data po jednotlivých bytech. SPI rámec (viz *Obrázek 24*) o velikosti 16 bitů tak musel být rozdělen na dvě části – *lowByte* a *highByte,* které se posílají za sebou. To samé platí pro vyčítání dat ze senzoru.

Část řízení reálné soustavy týkající se Simulinku je rozebírána dále v kapitole 6.1.

Zjednodušený vývojový diagram je zachycen na následujícím obrázku.



Obrázek 28 – Vývojový diagram programu Arduina

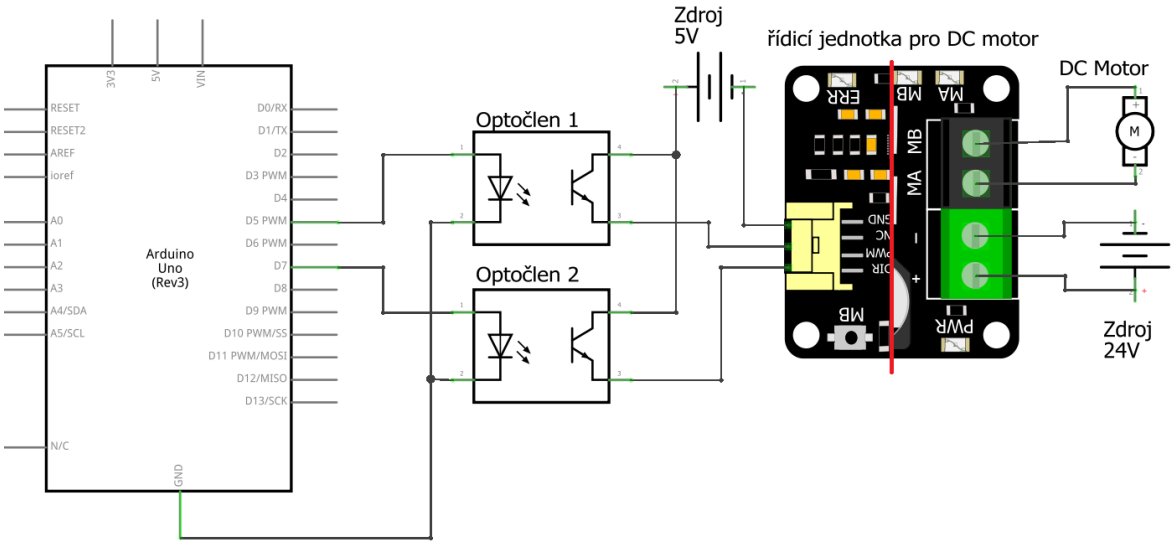
4.2.8 Zdroj 24 V a 5 V

Jako napájecí zdroj pro motor byl použit SIEMENS SITOP PSU100L. Jedná se o stabilizovaný stejnosměrný zdroj s napětím 24 V a výstupním proudem až 10 A. Zdroj je napájen ze sítě 230 Vac. Slouží také k napájení indukčních snímačů. [24]

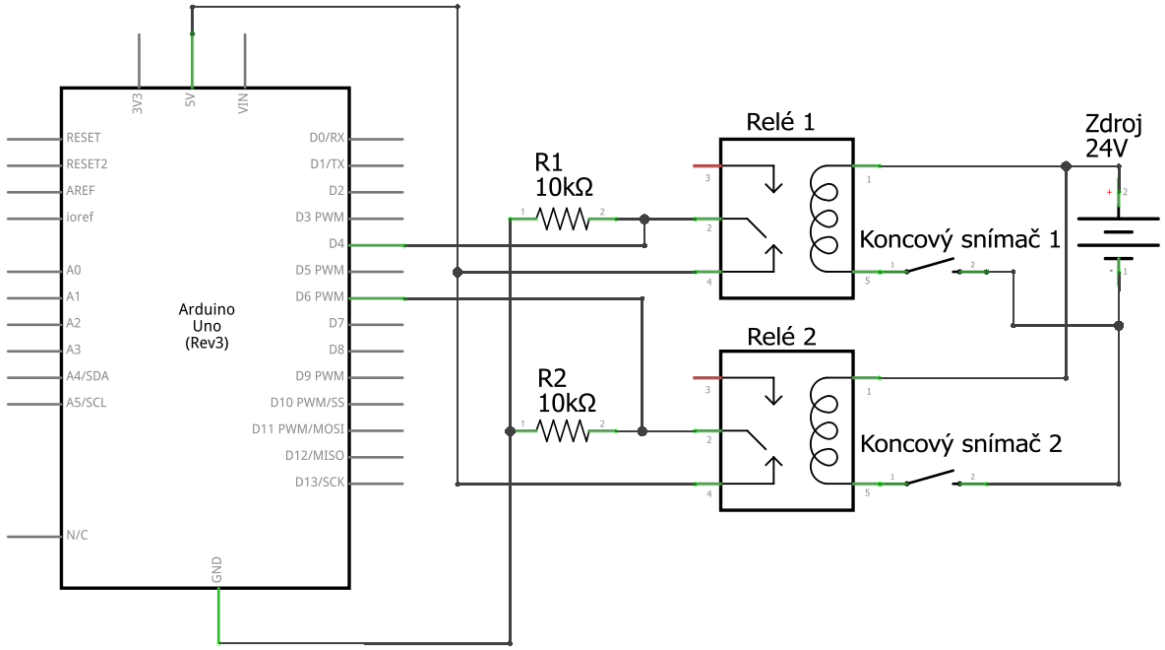
Druhý zdroj s napětím na výstupu 5 Vdc, maximálním proudem 0,5 A, je určen k napájení galvanicky oddělené části řídicího obvodu obsahující optočleny a řídicí jednotku motoru.

4.2.9 Schéma zapojení

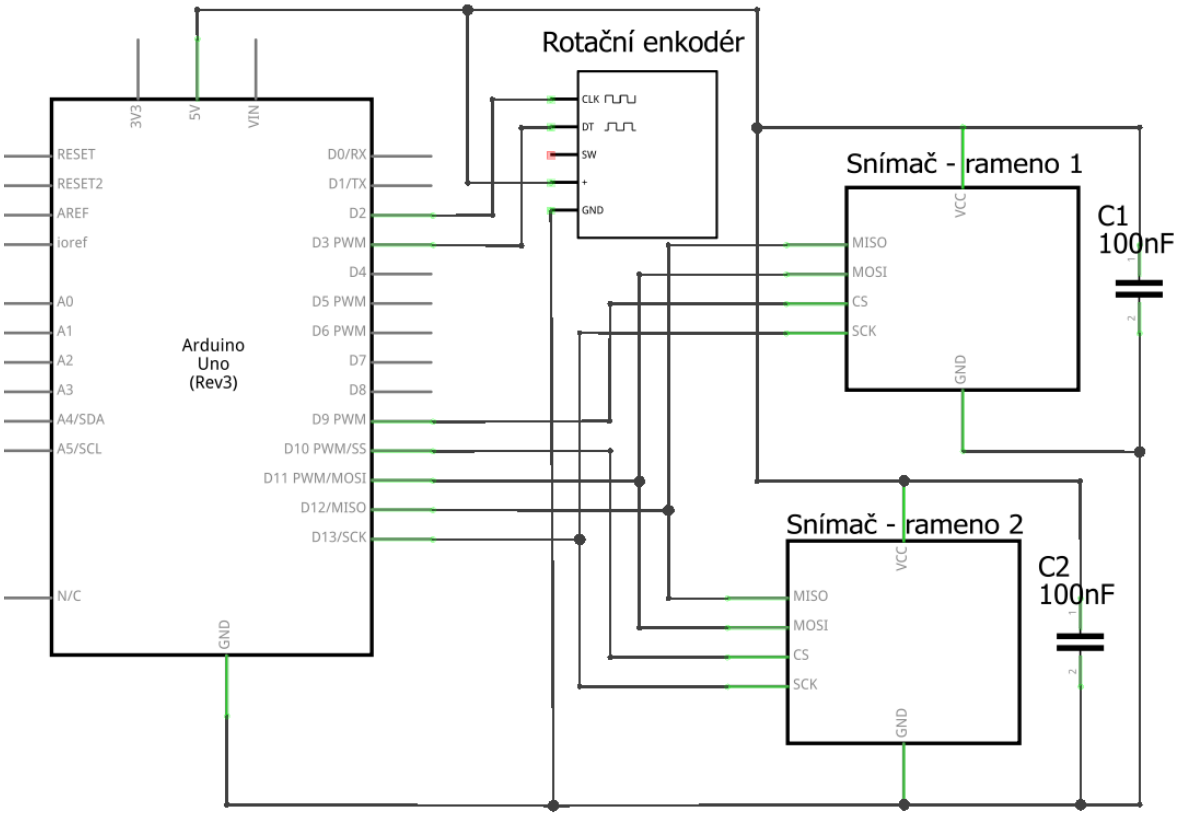
Celkové zapojení je rozděleno do tří menších schémat z důvodu přehlednosti. První schéma zobrazuje část řídicí (Arduino → motor). Druhé schéma znázorňuje část vstupů (snímače → Arduino) a ve třetím schématu je zapojení koncových snímačů.



Obrázek 29 – Schéma elektrického zapojení řídicí části



Obrázek 30 – Schéma elektrického zapojení indukčních koncových snímačů



Obrázek 31 – Schéma elektrického zapojení snímačů úhlu a enkodéru

4.3 Parametry soustavy získané měřením

Během konstrukce reálného kyvadla byly měřeny důležité parametry, které definují dynamiku systému. Naměřené hodnoty byly poté využity jako výchozí odhad při dodatečné identifikaci systému.

Hmotnosti vozíku a ramen byly zjištěny pomoci digitální stolní váhy. Délky ramen a vzdálenosti středů otáčení byly určeny z technické dokumentace pro výrobu ramen. Vzdálenosti od osy otáčení k těžišti byly určeny experimentem s břitvou. Břitva byla umístěna do svěráku a na ní probíhalo vyvažování ramene. Po úspěšném vyvážení byla vyznačena ryska a vzdálenost byla odečtena posuvným měřidlem.

Nakonec proběhl výpočet momentů setrvačností obou ramen. Rameno bylo zjednodušeno na obdélník o rozměru 0,345x0,02 m. Moment setrvačnosti obdélníku kolem středu:

$$I = \frac{1}{12} m(a^2 + b^2) \tag{4.1}$$

$$I_1 = \frac{1}{12} m_1(a^2 + b^2) = \frac{0{,}378}{12}(0{,}345^2 + 0{,}02^2) = 0{,}0038\ kg \cdot m^2 \tag{4.2}$$

$$I_2 = \frac{1}{12} m_2(a^2 + b^2) = \frac{0{,}171}{12}(0{,}345^2 + 0{,}02^2) = 0{,}0017\ kg \cdot m^2 \tag{4.3}$$

Tabulka 5 – Parametry získané měřením

Parametr	**Hodnota**
M [kg] – hmotnost vozíku	0,5420
m1 [kg] – hmotnost ramene 1	0,3780
m2 [kg] – hmotnost ramene 2	0,1710
l1 [m] – vzdálenost mezi osami otáčení	0,3000
l1_c, l2_c [m] – celkové délky ramen	0,3450
l_{1t} [m] – vzdálenost k těžišti rameno 1	0,1350
l2t [m] – vzdálenost k těžišti rameno 2	0,1705
I1 [$kg \cdot m^2$] – moment setrvačnosti kolem těžiště – rameno 1	0,0038
I2 [$kg \cdot m^2$] – moment setrvačnosti kolem těžiště – rameno 2	0,0017

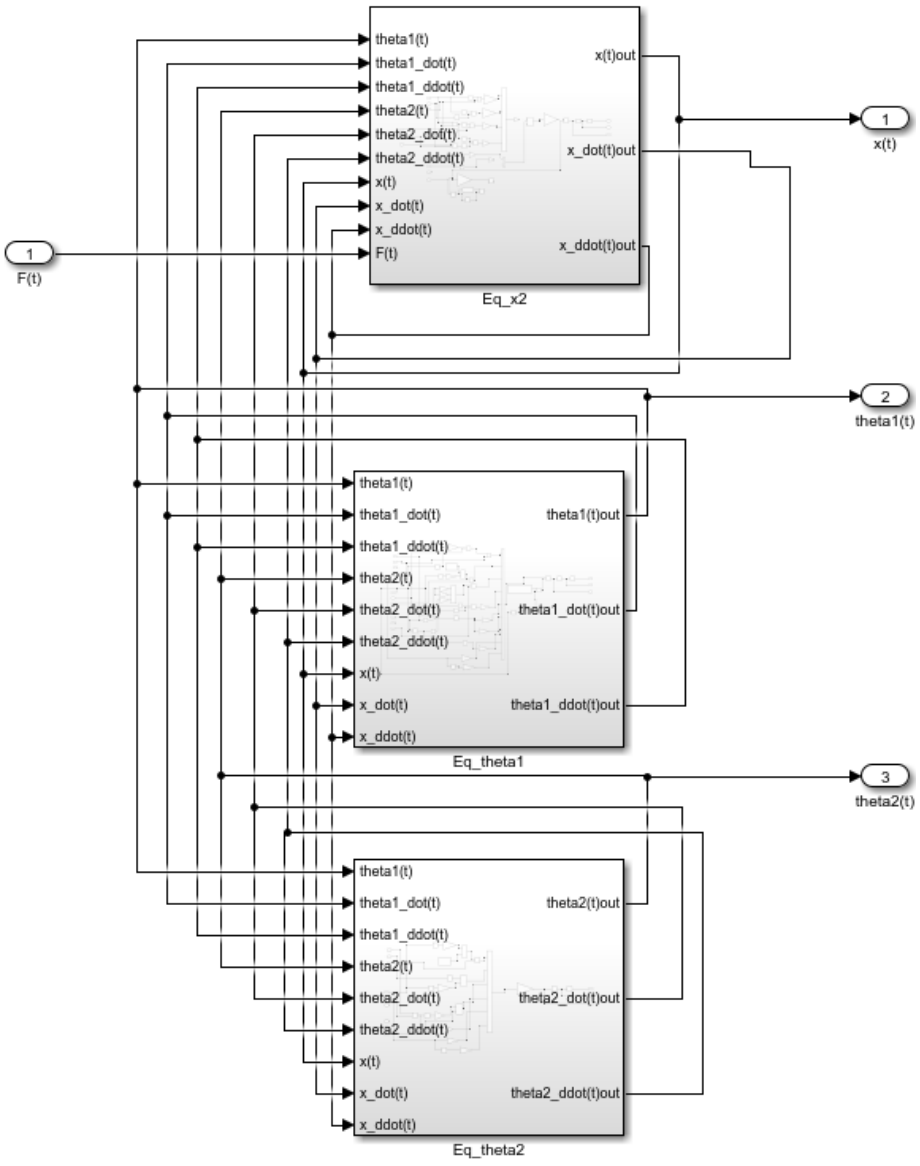
5 SIMULAČNÍ MODELY

Všechny modely byly vytvořeny v programu Matlab/Simulink.

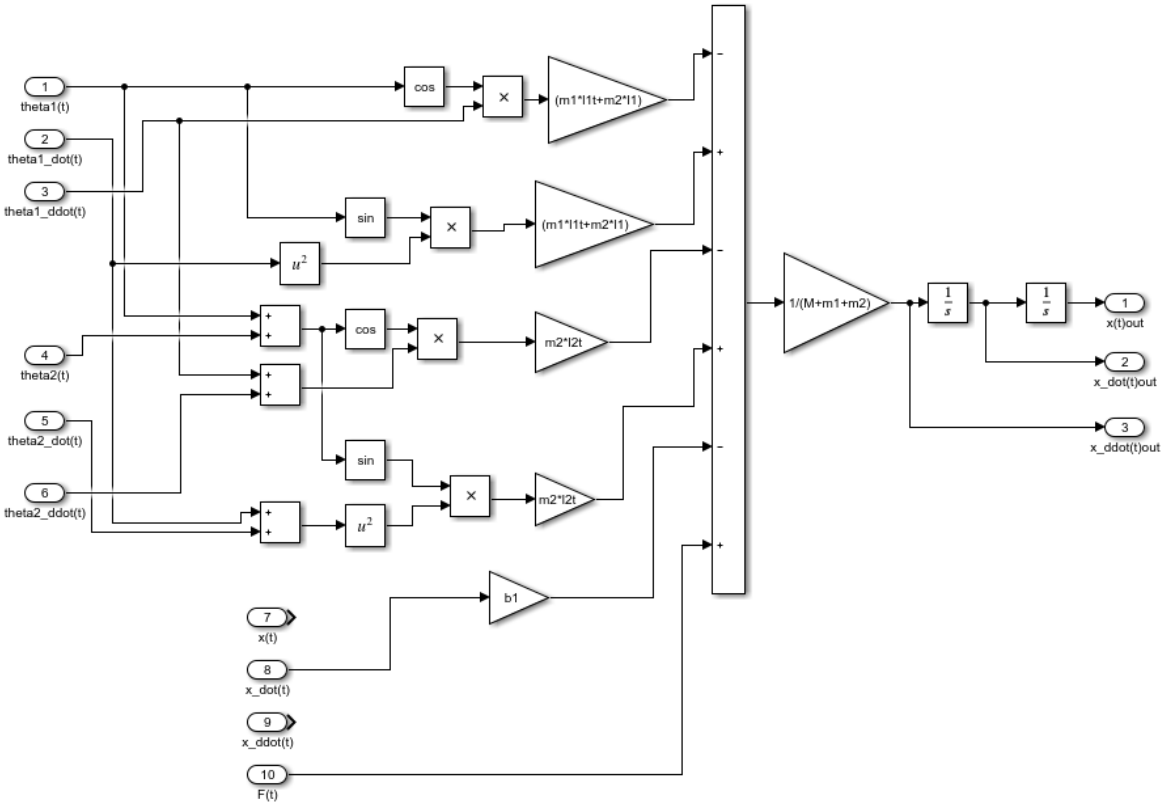
5.1 Model dvojitého inverzního kyvadla

Model kyvadla byl vytvořen na základě pohybových rovnic odvozených v kapitole 2.1.

Základem jsou 3 podsystémy, mezi kterými jsou propojeny výstupní signály na vstupy všech bloků. Každý podsystém poté vyjadřuje jednu diferenciální rovnici a výstupem jsou signály – poloha vozíku *x(t)*, úhel natočení $\theta_1(t)$ a $\theta_2(t)$. Výstupy lze rozšířit o první a druhé derivace těchto signálů (značené _dot a _ddot (double dot)). Vstupem do tohoto systému je síla *F(t)*.



Obrázek 32 – Model kyvadla



Obrázek 33 – Ukázka prvního podsystému Eq_x

5.2 Pomocné funkce

Ještě než budou rozebrány samotné skripty pro návrh a ověření regulátorů, je třeba se zmínit o pomocných funkcích.

Funkce *init(x_0_in,x_dot_0_in,theta1_0_in,theta1_dot_0_in,theta2_0_in,theta2_dot_0_in)* je volána pro inicializaci parametrů kyvadla (vytváří hodnoty do *workspace* se kterými lze poté pracovat). Jejím vstupem jsou počáteční hodnoty stavových veličin. Například *init(0,0,pi,0,0,0)* slouží k inicializaci parametrů kyvadla a nastavení počátečních podmínek simulace do polohy Up/Up (kyvadlo vzhůru, nulová pozice v ose x a všechny rychlosti nulové).

Funkce *[A,B,C,D] = linearizace(sw,parameters)* pracuje s parametry kyvadla a na základě zvoleného rovnovážného bodu dělá linearizaci. Funkce vrací čtveřici matic stavového popisu, které jsou poté využity k návrhu regulátoru. Parametr *sw* slouží k určení rovnovážného bodu, kolem kterého chceme systém linearizovat (*sw*: 1 – pozice Down/Down; 2 – Up/Up; 3 – Up/Down, 4 – Down/Up).

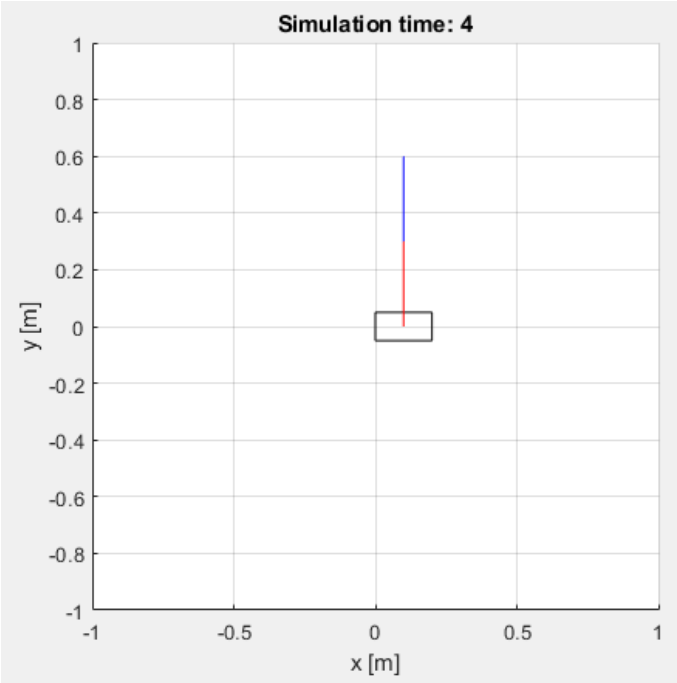
U návrhu regulátoru je zde dodatečná funkce *swHandle()*, která slouží pouze k automatickému určení úhlů θ_1, θ_2 pro inicializaci podle zvolené polohy.

Skript *regKrit.m* má za úkol výpočet hodnotící funkce regulačního pochodu. Jedná se o běžnou kvadratickou chybu. Pro *N* vzorků signálů $y, w, u = [N \times 1]$:

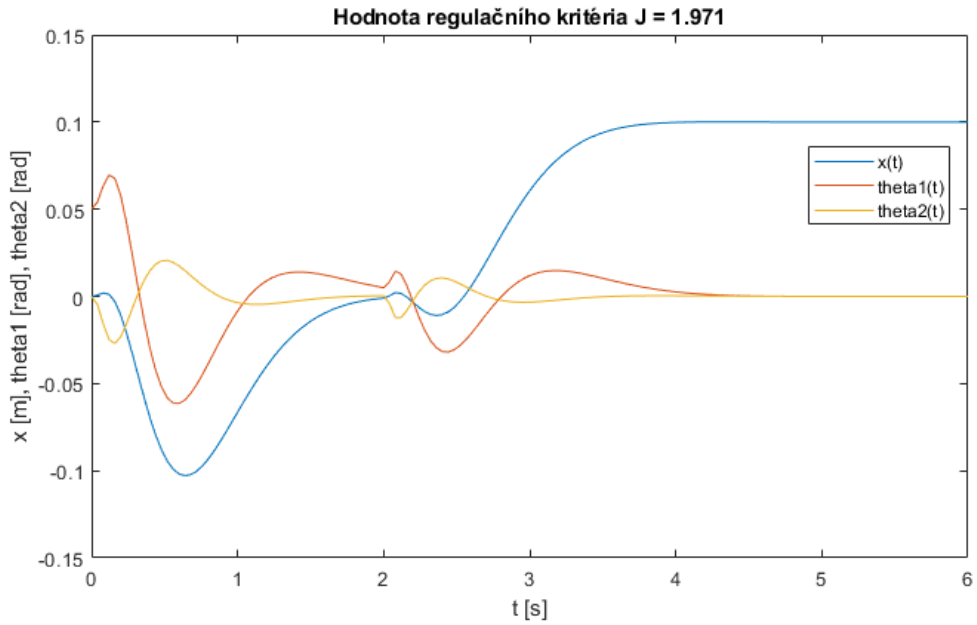
$$J = P_e(w - y)'(w - y) + P_u u'u \tag{5.1}$$

Kde P_e je penalizační konstanta regulační odchylky a P_u je penalizace akčního zásahu. Při porovnávání jednotlivých regulačních pochodů je zvolena hodnota $P_e = 1$ a $P_u = 0.001$.

Skript *AnimaceRegulace.m* slouží ke grafickému vykreslení průběhu regulace. Jedná se o několik grafů vykreslených s určitým zpožděním tak, aby vytvářel dojem animace. Skript se automaticky spouští po ukončení simulace a ze získaných hodnot v simulaci je průběh zobrazen. Kyvadlo s vozíkem má stejné poměry velikostí jako reálná soustava. Zároveň je zde vykreslen druhý graf, který ukazuje regulační pochod a hodnotu regulačního kritéria. Hodnoty v grafu regulačního pochodu ukazují odchylku od zvoleného rovnovážného stavu, nikoliv momentální hodnotu z modelu kyvadla. Animace byla nastavena tak, aby v ní čas běžel stejně rychle jako v reálu. Bohužel zde závisí na výkonu počítače, na kterém se animace spouští. Grafika tak může působit zpomaleně nebo zrychleně. Nad animací se zobrazuje také momentální čas simulace, podle kterého se lze řídit.



Obrázek 34 – Ukázka okna s animací



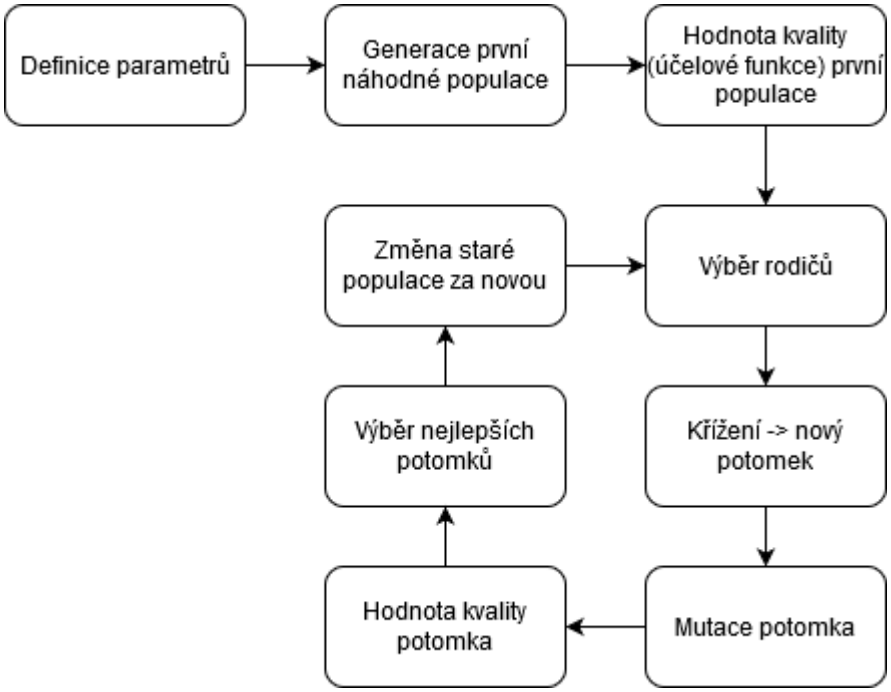
Obrázek 35 – Ukázka regulačního pochodu a hodnoty kritéria J

5.3 Identifikace parametrů pro návrh regulátorů

Při konstrukci reálného modelu byly naměřeny a vypočítány parametry kyvadla (*Tabulka 5*). Tyto parametry slouží jako základ pro numerické metody, díky kterým přesněji identifikujeme chování soustavy. Celá identifikace se skládá ze dvou částí. První část slouží k identifikaci dynamiky samotných ramen (*IdentifikaceEvoluceSystem.m*, lze upravit i pro celou soustavu s motorem). Druhá část identifikuje motor (*IdentifikaceEvoluceMotor.m)*.

5.3.1 Diferenciální evoluce

Jako numerický řešitel pro optimalizaci parametrů byla zvolena evoluční technika, přesněji diferenciální evoluce (DE). Jedná se o stochastický algoritmus, který vychází z oboru biologie a genetiky. Tato práce nemá za cíl rozebírat teorii evolučních algoritmů, ale následující část bude popisovat základy využité při tvorbě vlastního programu.



Obrázek 36 – Obecný diagram genetických algoritmů

Všechny vytvořené skripty DE pracují na stejném principu, ale liší se v počtu optimalizovaných parametrů a tvaru účelové funkce (kritérium).

Parametry DE jsou voleny:

Tabulka 6 – Hodnoty parametrů DE

Parametr	**Interval**	**Popis**
NP	15 - 40	Velikost populace
F	0,7 – 0,9	Mutační konstanta
CR	0,7 – 0,9	Práh křížení
G	50 - 500	Počet generací / kol šlechtění

Nejprve se inicializuje první náhodná generace. Jedinci zde představují jedno možné řešení daného problému. Všechny algoritmy vycházely z určitého odhadu, proto zde parametry (geny, argumenty) v jedinci vyjadřují násobek počátečního odhadu. Například původní hmotnost vozíku 0,542 kg je zde reprezentována možnou hodnotou jedince 0,9 - 1,1, což znamená počáteční odhad 0,488 – 0,596 kg. Jedinci jsou zde uloženi jako matice [NP x počet genů + 1], kde řádek matice představuje jednoho jedince a každý sloupec matice je jeden

gen (argument). Poslední sloupec matice je vyhrazen pro hodnotu účelové funkce daného jedince.

Algoritmus po inicializaci náhodné populace vypočítá hodnoty účelové funkce za pomocí vytvořených simulací. Simulace poskytuje data z vytvořeného modelu kyvadla s ohledem na parametry ověřovaného jedince a účelová funkce počítá kvadratickou chybu mezi simulačními daty a naměřenými daty na reálné soustavě. Cílem je minimalizace účelové funkce.

Generacemi se postupuje následovně: Algoritmus prochází v každé generaci jedince postupně od prvního k poslednímu, než je ukončen dosažením poslední generace. Výpočet lze ukončit i požadovanou hodnotou účelové funkce (*Kritérium* < *KritériumMin*). Noví jedinci jsou počítáni takto: k vybranému jedinci (aktivní/aktuální jedinec) jsou vybrání tři náhodní jedinci a proběhne výpočet vektoru podle:

$$\overrightarrow{J_{novy}} = (\overrightarrow{J_1} - \overrightarrow{J_2})F + \overrightarrow{J_3} \tag{5.2}$$

Kde $J_{1,2,3}$ jsou náhodní jedinci (vektory) a F – mutační konstanta (skalár)

Nově vzniklý vektor se také označuje jako mutace nebo šumový vektor. Nakonec přichází křížení. Pro každý gen je vygenerováno náhodné reálné číslo od 0 do 1. Pokud je náhodné číslo větší než práh křížení *CR*, je pro Trial (pokusného) jedince použit gen z šumového vektoru, v opačném případě se přenáší gen současného jedince. U Trial jedince poté dochází k výpočtu účelové funkce. Pokud má Trial jedinec lepší (v našem případě menší) účelovou funkci, nahrazuje současného jedince v populaci. Algoritmus takto pokračuje přes další jedince.

Výsledkem je průběžné snižování hodnoty účelové funkce. Výsledná populace (nebo vybraní jedinci) je po dostatečném počtu generací blízko optimálního řešení dané úlohy. Velkým problémem je časová náročnost řešení pro větší počet jedinců a generací (pro každý krok se spouští simulace k získání dat).

Vytvořené skripty zobrazují graficky průběh nejlepšího jedince pro každou generaci a lze tak sledovat vývoj identifikace.

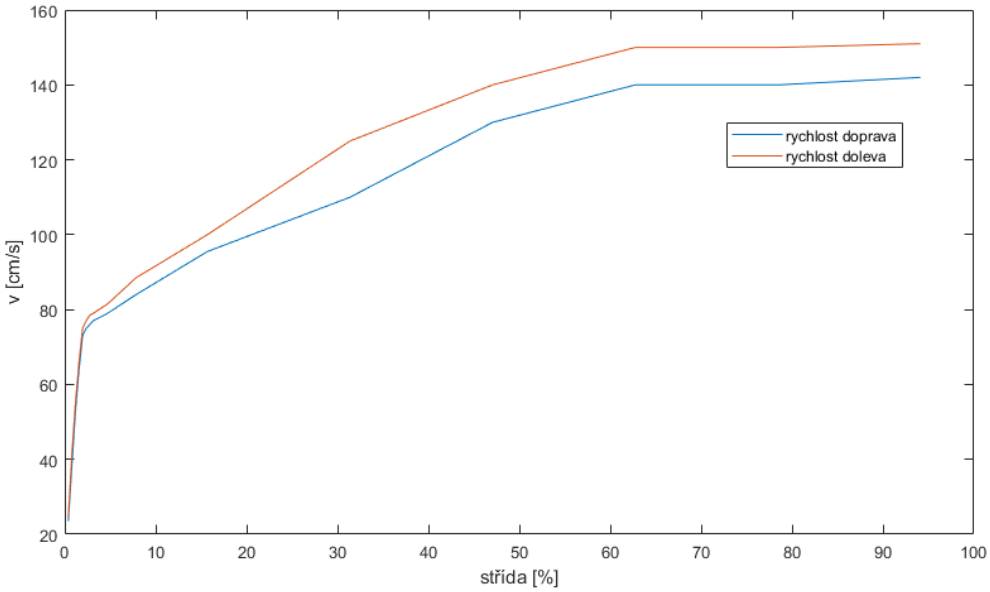
5.3.2 Soustava motoru

Práce s motorem byla nejvíce problémová na celé reálné soustavě. Nejprve probíhaly pokusy o řízení motoru za pomoci snímání proudu motorem. Řešení bylo velmi nestabilní, hlavně

z důvodu nekvalitního motoru i proudového senzoru. Přešlo se tedy na jiný přístup – uvažování motoru jako zdroje rychlosti a poté zrychlení. Pohon je natolik silný, že dynamika celé soustavy na jeho pohyby nemá téměř žádný vliv.

Motor je řízen pomocí střídy PWM z Arduina. Arduino hodnotou 0 – 255 na PWM výstupu reprezentuje rozsah střídy 0 – 100 %. Byl tedy navržen přenos, který má jako vstup střídu a na výstupu rychlost / zrychlení vozíku.

Aby bylo možné využít takové řízení, musela by být závislost rychlosti na střídě lineární. Proběhlo proto měření, ze kterého plyne následující charakteristika:



Obrázek 37 – Závislost rychlosti vozíku na střídě PWM řízení

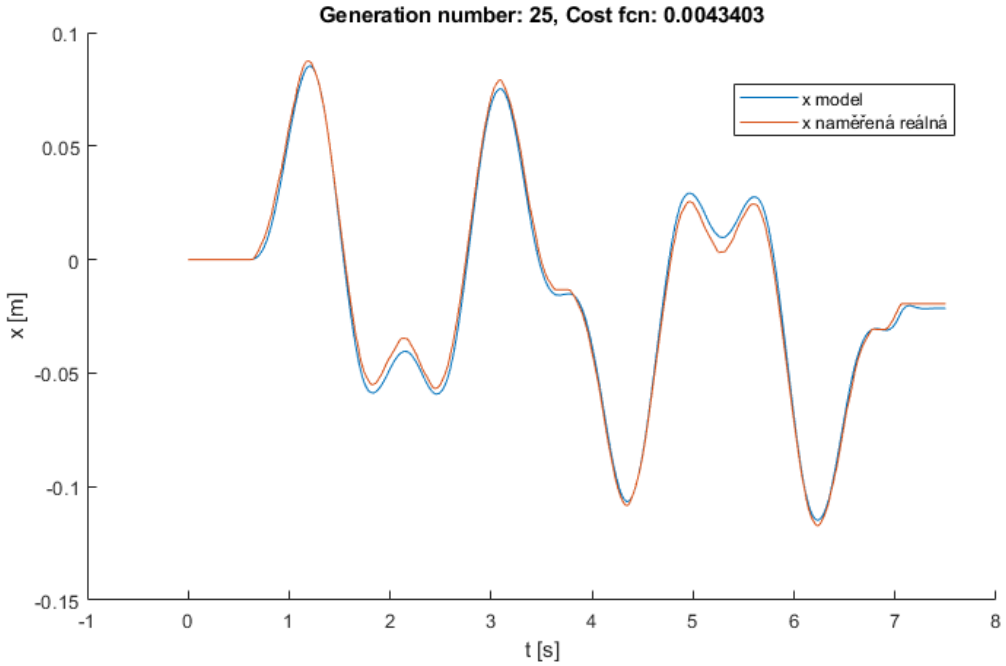
Charakteristika byla lineární v oblasti blízké nulové střídě (0 – 2 %), což by poskytovalo pouze 10 rozdílných hodnot pro řízení (5 hodnot pro každý směr). Další lineární část se nacházela už v oblasti, kde rychlosti byly moc vysoké na to, aby mohly být využity k jemné korekci při balancování (od 75 cm/s). Od 60 % střídy se už rychlost nezvyšovala. Další vlastnost, která vyplynula z charakteristiky, byla závislost rychlosti na směru otáčení motoru. Tento rozdíl v rychlostech pro oba směry byl následně kompenzován při identifikaci systému v simulačním bloku kyvadla (reprezentující první pohybovou rovnici).

Bylo využito skutečnosti, že perioda vzorkování v Simulinku je 0,02 s. Za tuto dobu je Arduino schopno několika desítek cyklů. Proto bylo manuálně vytvořeno nové „mapování“ PWM. Perioda 0,02 s byla rozdělena na šest menších částí. V každé z těchto částí Arduino

posílá na motor hodnotu PWM zvlášť. Není tedy třeba se omezit na jednu hodnotu PWM obdrženou z PC po celou dobu periody, ale lze provádět úpravu přímo v cyklu Arduina.

Celý rozsah PWM odesílaný z PC byl omezen na 52 hodnot pro jeden směr. V Arduinu poté probíhá mapování těchto hodnot tak, aby bylo dosaženo lineární charakteristiky. Výsledkem je lineární rozsah rychlostí od 0 do 140 cm/s s krokem 2,5 cm/s.

Pro identifikaci motoru byla využita vygenerovaná posloupnost střídy jako vstup (součin funkcí sinus) a na reálné soustavě se měřila poloha a rychlost vozíku. Hodnoty poté sloužily k výpočtu účelové funkce v optimalizačním algoritmu.



Obrázek 38 – Identifikovaná soustava motoru, výstup z DE

Dynamika motoru byla identifikována jako přenos druhého řádu:

$$G_m(s) = \frac{0{,}0286}{0{,}0018s^2 + 0{,}042s + 1}$$

Pro použití motoru jako zdroje zrychlení vozíku pak:

$$G_m(s) = \frac{0{,}0286s}{0{,}0018s^2 + 0{,}042s + 1}$$

Po úspěšné linearizaci motoru a vytvoření přenosu proběhla úprava první pohybové rovnice (zanedbání dynamiky, zrychlení = vstup do systému). Změna byla poté aplikována i v simulačních schématech a algoritmech pro linearizaci. Celý systém tak lze popsat spojením

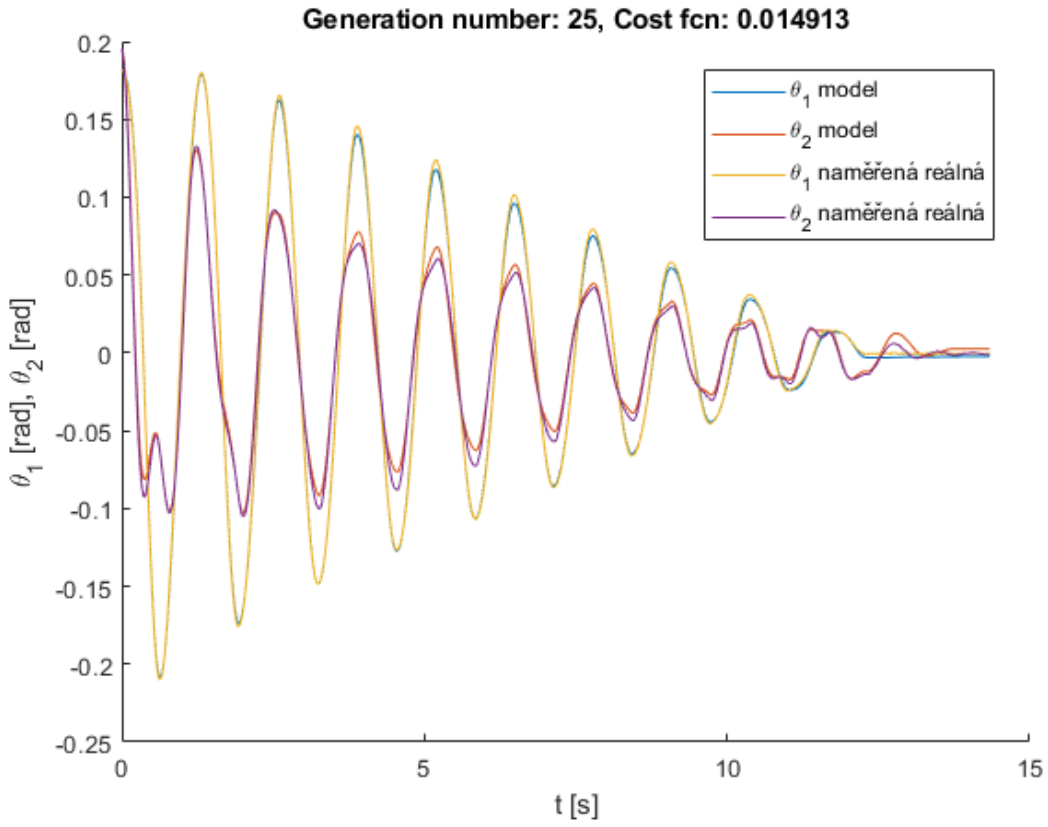
systému motoru a systému kyvadla do série (výstup motoru je zrychlení → vstup do soustavy kyvadla).

5.3.3 Parametry kyvadla po identifikaci

Soustava byla doplněna o součinitel suchého tření ramen pro lepší výsledek identifikace. Porovnání modelu a reálné soustavy je vidět níže.

Tabulka 7 – Porovnání měřených a nově identifikovaných parametrů

Parametr	**Původní hodnota**	**Hodnota po identifikaci**
M [kg] – hmotnost vozíku	0,5420	0,5420
m1 [kg] – hmotnost ramene 1	0,3780	0,4075
m2 [kg] – hmotnost ramene 2	0,1710	0,1716
l1 [m] – vzdálenost mezi osami otáčení	0,3000	0,3000
l1_c, l2_c [m] – celkové délky ramen	0,3450	0,3450
l1t [m] – vzdálenost k těžišti rameno 1	0,1350	0,1306
l2t [m] – vzdálenost k těžišti rameno 2	0,1705	0,1752
I1 [kg · m^2] – moment setrvačnosti kolem těžiště – rameno 1	0,0038	0,0069
I2 [kg · m^2] – moment setrvačnosti kolem težiště – rameno 2	0,0017	0,0015
b1 [N · $s \cdot m^{-1}$] – koeficient viskózního tření, rameno 1	0	0,0012
b2 [N · $s \cdot m^{-1}$] – koeficient viskózního tření, rameno 2	0	0,0017
c1 [N · $s \cdot m^{-1}$] – koeficient suchého tření, rameno 1	0	0,0084
c2 [N · $s \cdot m^{-1}$] – koeficient suchého tření, rameno 2	0	0,0007



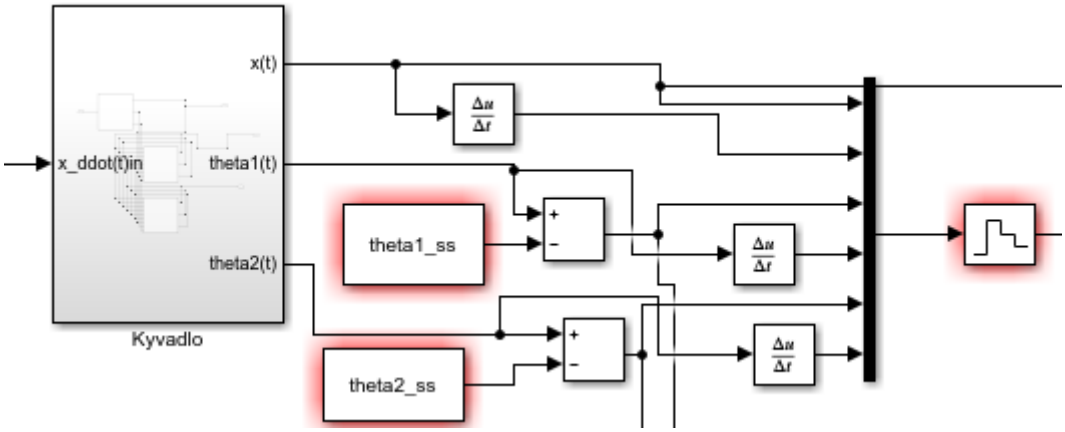
Obrázek 39 – Identifikace ramen kyvadla a porovnání s naměřeným průběhem

5.4 Výsledky regulace - simulace

Byly vytvořeny 3 skripty pro každý druh regulace – PP (*polePlacementRizeni.m*), LQG (*LQGrizeni.m*) a MPC (*MPCrizeni.m*). Všechny skripty jsou postaveny na stejném principu. V každém skriptu se nejprve pouští *pathFind.m*. Jedná se o přidání cesty k modelům a funkcím, které se nachází v jiné složce, aby je bylo možné spustit z těchto skriptů. Následuje volání funkce *init()* a *linearizace()*. V těchto funkcích si volíme, jaká je počáteční hodnota pro stavový vektor, probíhá načtení proměnných do *workspace* a systém je linearizován kolem zvolené polohy. Lze nastavit i periodu vzorkování *Ts*. Dále probíhá výpočet regulátoru na základě zvoleného druhu za pomoci matic systému, které jsou převedeny ze spojité oblasti do diskrétní. Je zde možno nastavit požadovanou hodnotu polohy vozíku pro ověření regulátoru a celkovou dobu simulace. Po spuštění skriptu se vše provede automaticky a je zobrazena animace regulace a výsledný regulační pochod.

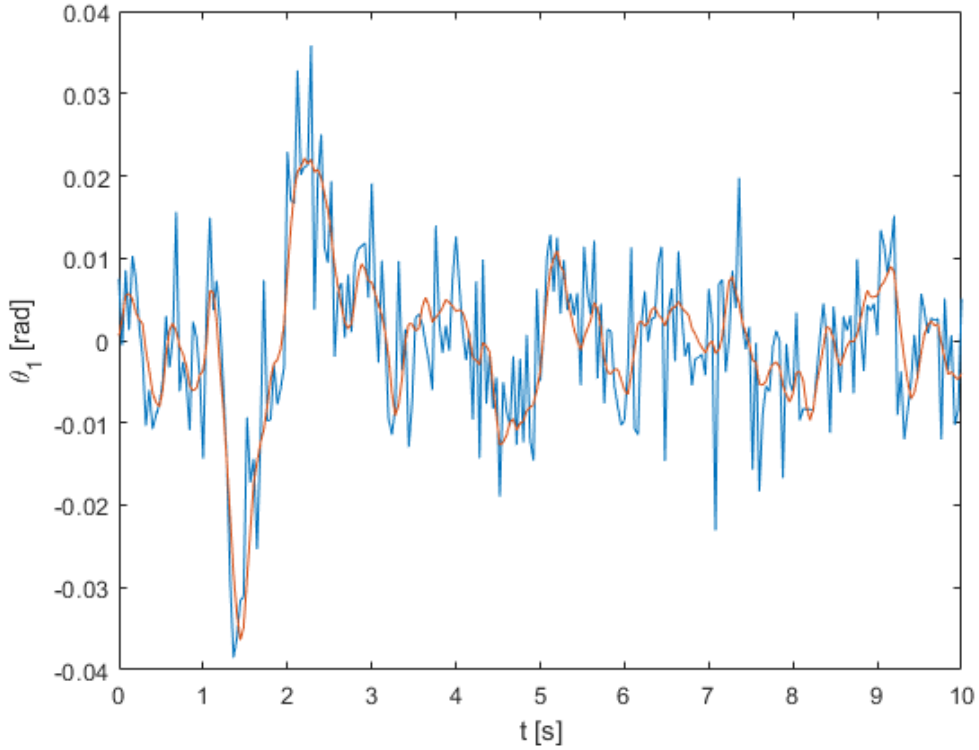
Pro simulaci byla vytvořena dvě schémata – jedno společné pro LQG a PP (*LQG_PP_rizeniModel.slx*, využívají stejný princip regulace) a druhé schéma pro MPC (*MPC_rizeniModel.slx*).

V obou schématech se nachází část, která slouží k přepočtu úhlu na nulovou hodnotu (*theta1_ss a theta2_ss*) – nezáleží, v jaké pozici máme kyvadlo (Up/Up, Down/Up apod.), od úhlů je odečtena rovnovážná poloha a do Kalmanova filtru vstupují odchylky od této rovnovážné polohy. K určení rychlostí zde slouží numerická derivace a všechny hodnoty jsou vzorkovány s požadovanou periodou *Ts*.



Obrázek 40 – Část schématu pro určení rovnovážné polohy a derivace

V simulacích se nachází Kalmanův filtr (KF). Jedná se o optimálního pozorovatele stavu. Jeho vstupem jsou měřené stavy systému a akční zásah. Na základě matic systému pak tento filtr odhaduje celý stavový vektor. Bere se zde také ohled na působení šumu. KF lze ladit maticemi *Q* (kovarianční matice šumu procesu) a *R* (kovarianční matice šumu měření). V zásadě reprezentují míru nejistoty pro měření a modelovaný systém. Pokud máme měření téměř bez šumu a nepřesný model – volíme malé hodnoty *R* a větší hodnoty *Q*. Naopak pro dobrý model a nepřesné měření se šumem volíme malé hodnoty *Q* a větší *R*. Obecně je volba *Q* a *R* kompromisem pro danou úlohu.



Obrázek 41 – Průběh θ_1 se šumem (modrá) a výstup z KF (oranžová)

5.4.1 Pole - placement / Přiřazení pólů

Nejjednodušším způsobem regulace je metoda přiřazení pólů. Umístění pólů bylo experimentální.

Hodnoty pólů, které vedly na stabilní pochod, jsou:

poly1 = [-15;-3.5;-11.2;-3;-12;-2.5; -0.005;-0.5]

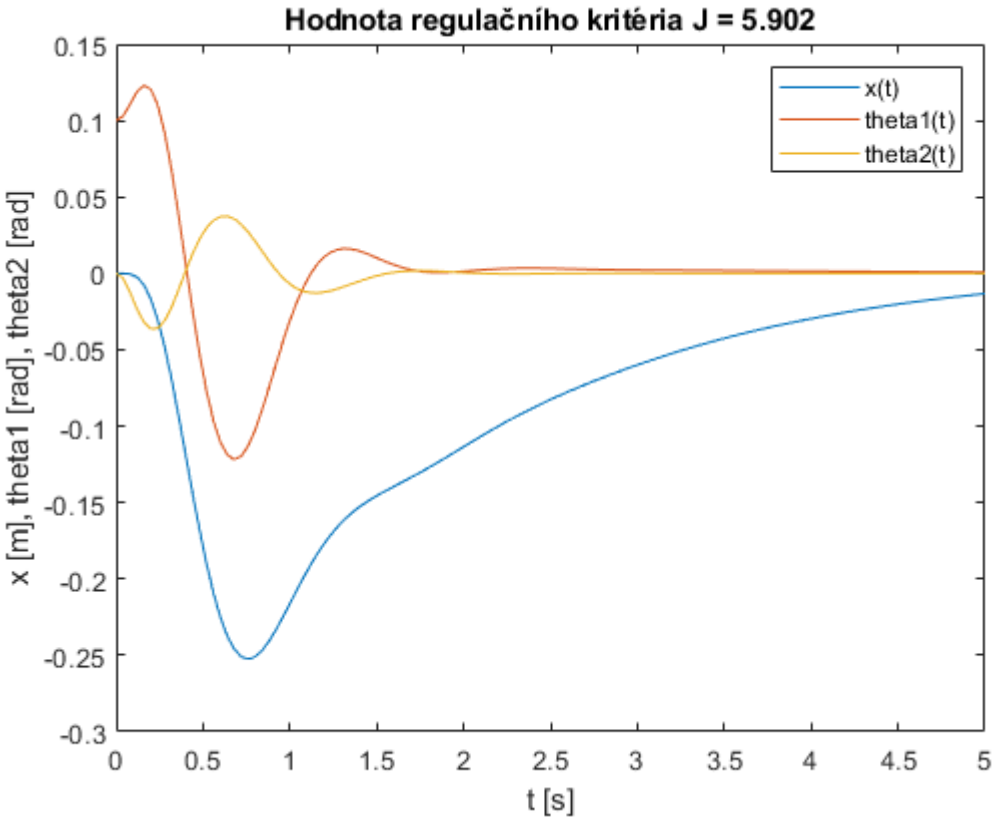
Dále byly hodnoty pólů zvětšeny a průběhy jsou porovnány graficky.

poly2 = [-18;-4.5;-12.2;-4;-13;-3.5; -0.005;-0.5];

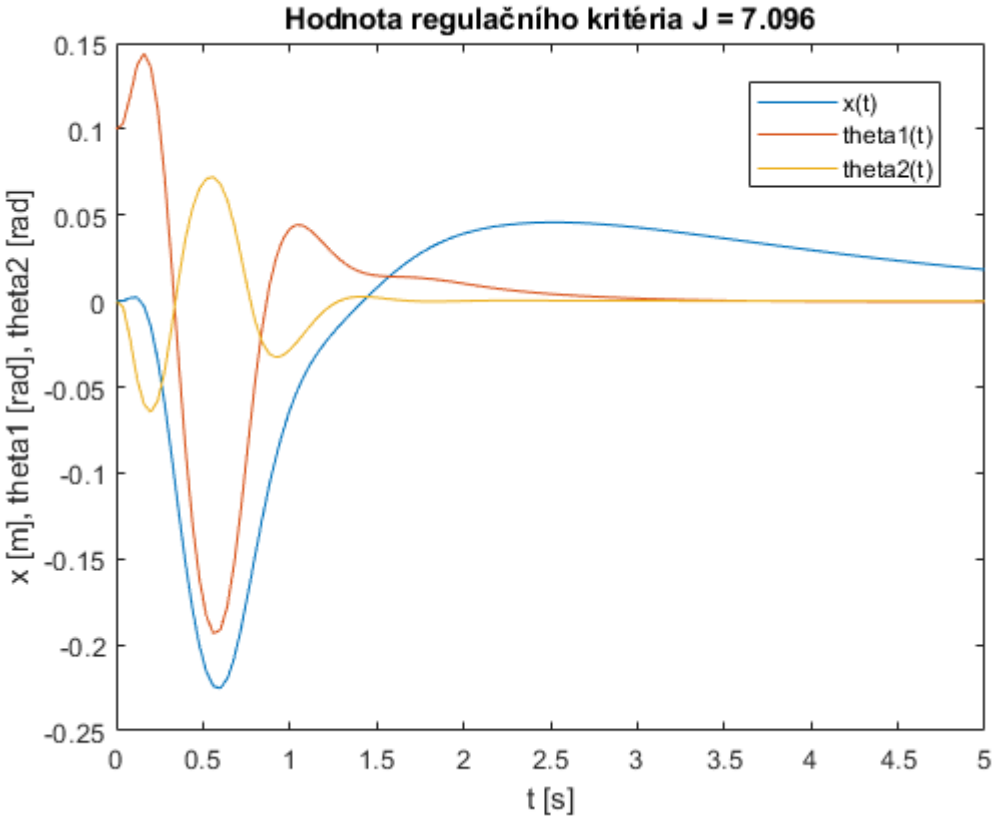
Z porovnání a dalších provedených experimentů je jasné, že nelze jednoduše určit póly tak, aby regulační pochod splňoval předem stanovené požadavky.

Ve všech příkladech, které následují (i pro LQG a MPC) je inicializace určena odchylkou 0,1 radiánů od rovnovážné polohy, ostatní stavové veličiny jsou nulové:

init(0,0,pi+0.1,0,0.00,0)



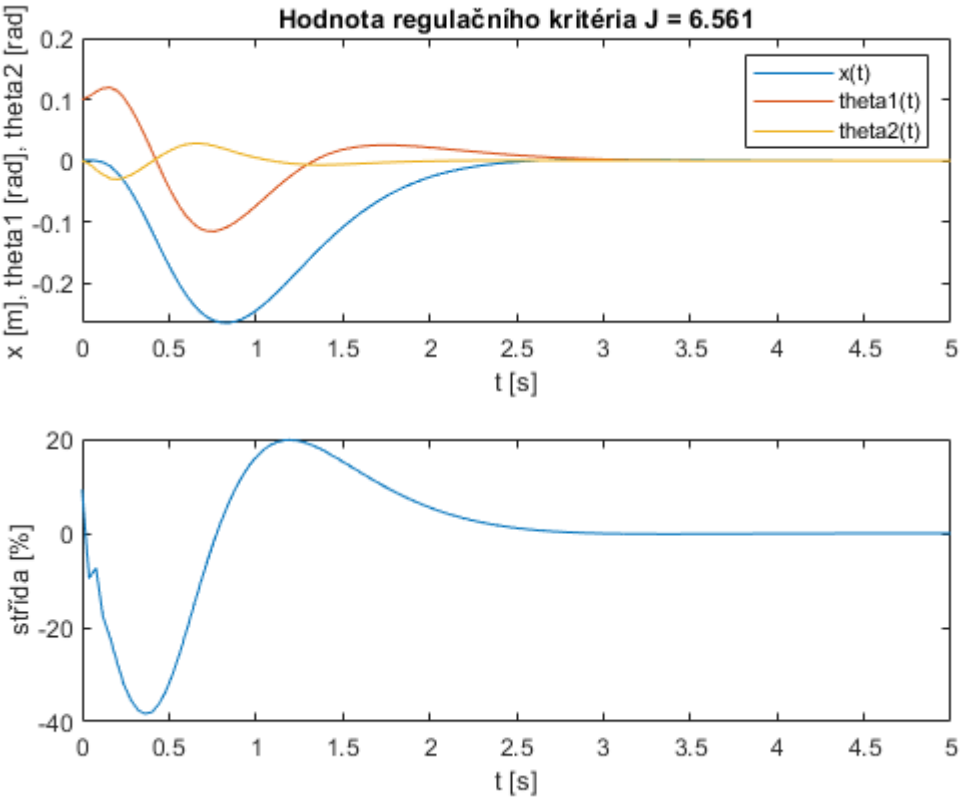
Obrázek 42 – Regulační pochod pro přiřazení pólů 1



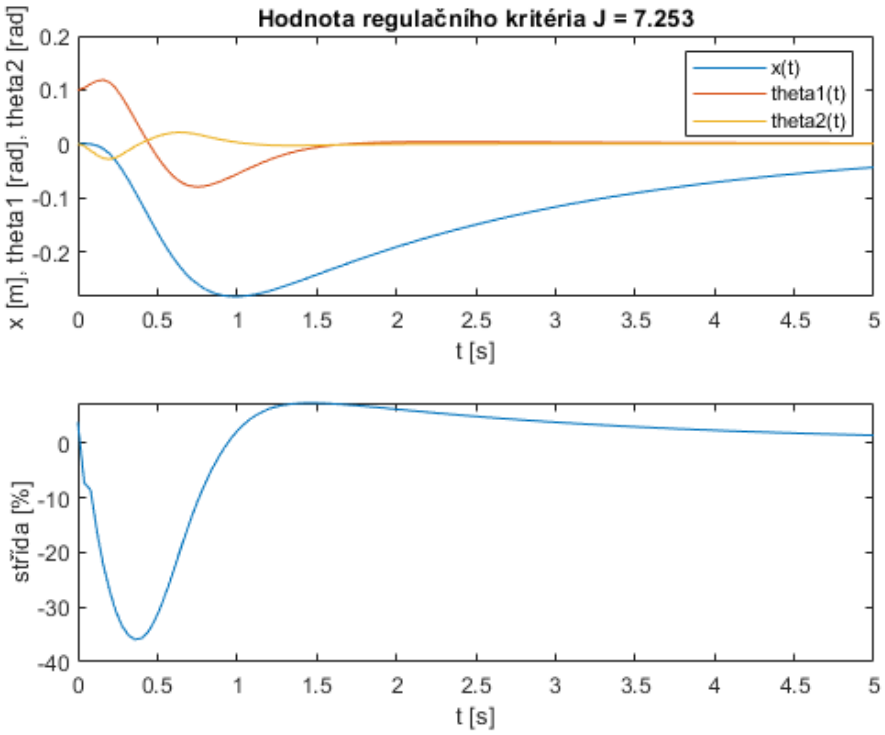
Obrázek 43 – Regulační pochod pro přiřazení pólů 2

5.4.2 LQG (Linear-Quadratic-Gaussian)

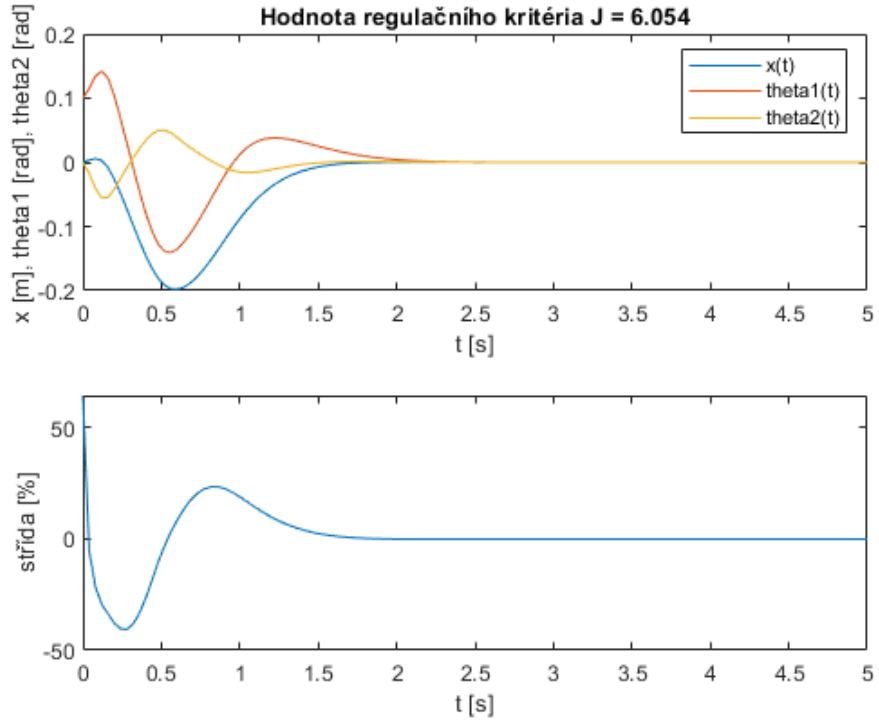
U LQG je jednodušší upravit parametry regulátoru podle potřeby. Slouží k tomu váhová matice Q a penalizace akčního zásahu Rp. Každý prvek matice Q zastupuje jeden stav stavového vektoru. V našem případě $Q = [x, x_{dot}, \theta_1, \theta_{1dot}, \theta_2, \theta_{2dot}, k_1, k_2]$, kde k_1 a k_2 představují stavy vzniklé sériovým spojením systému motoru a kyvadla. Pro výpočet regulátoru je použita funkce Matlabu *dlqr()*, která slouží k určení parametrů diskrétního LQR regulátoru podle matic systému, váhové matice a penalizace akčního zásahu.



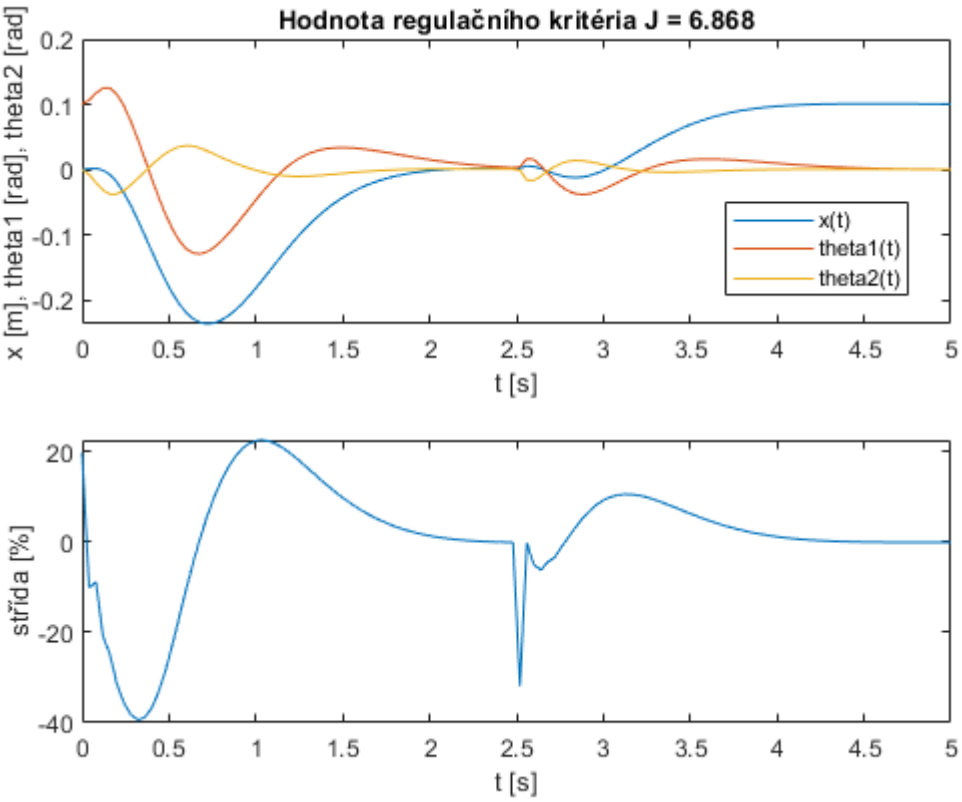
Obrázek 44 – Regulační pochod LQG, Q= diag([3,0.1,5,0.1,5,0.1,0,0]), Rp = 0.0001



Obrázek 45 - Regulační pochod LQG, Q= diag([3,0.1,5,0.1,5,0.1,0,0]), Rp = 0.01, zvýšená penalizace akčního zásahu



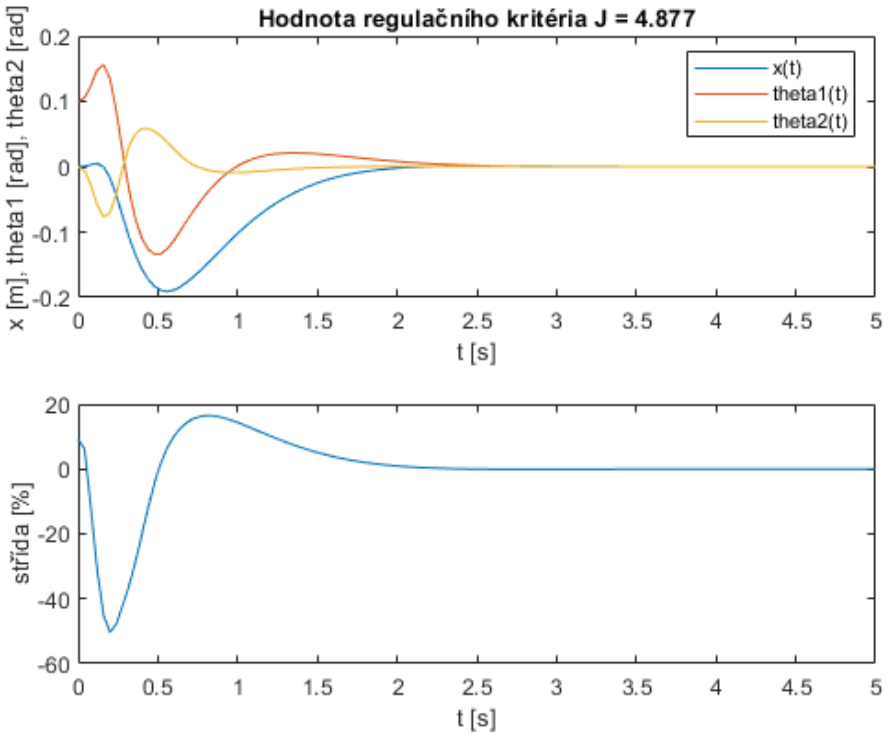
Obrázek 46 - Regulační pochod LQG, Q= diag([50,0.001,10,0.05,10,0.05,0,0]), Rp = 0.000001, agresivní pochod



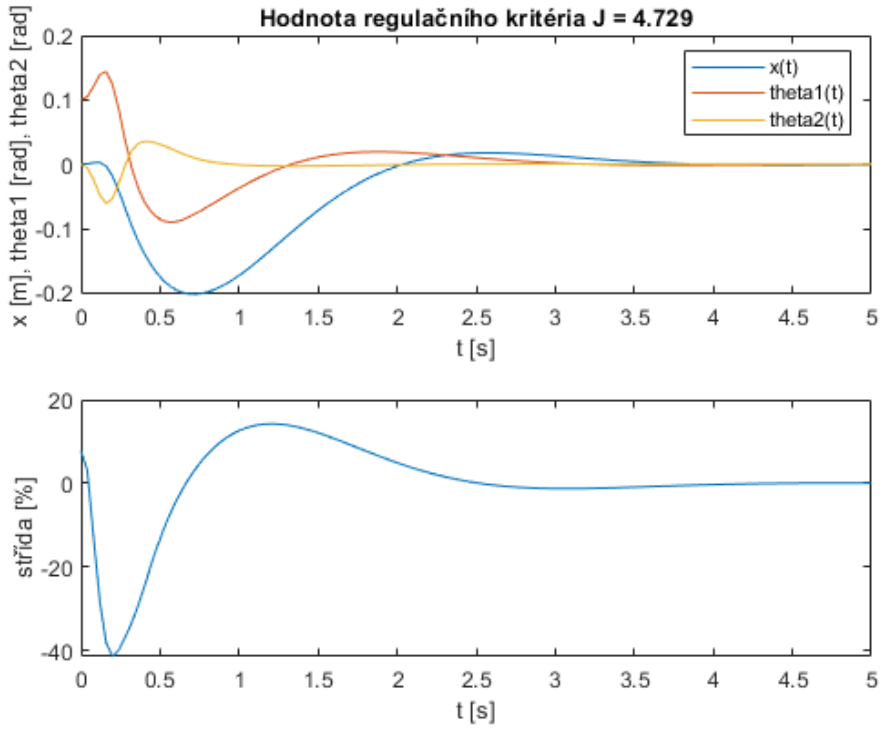
Obrázek 47 – Regulační pochod LQG, Q = diag([5,0.1,3,0.05,3,0.05,0,0]), Rp = 0.00001; změna žádané polohy vozíku v čase 2,5 s na 0,1 m

5.4.3 Prediktivní regulátor

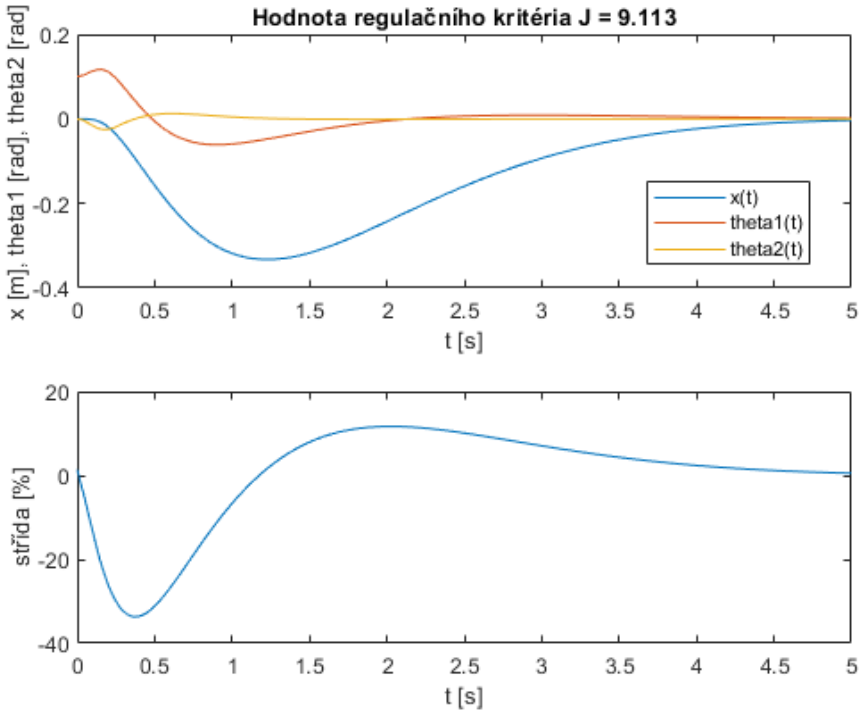
U prediktivního řízení byl naprogramován vlastní výpočet regulátoru a jeho implementace v simulaci podle teorie z kapitoly 3.1.3, přesněji [10]. Na základě simulací bylo zjištěno, že nejnižší hranice pro počet budoucích predikcí N_2 je 40 kroků. Pod touto hranicí nemusí být regulace stabilní. U počtu predikcí akčního zásahu N_u je vhodné volit alespoň 8 kroků. Při menším počtu predikcí akčního zásahu nastává zhoršení kvality regulace (regulace ale zůstává stabilní i pro $N_u = 4$).



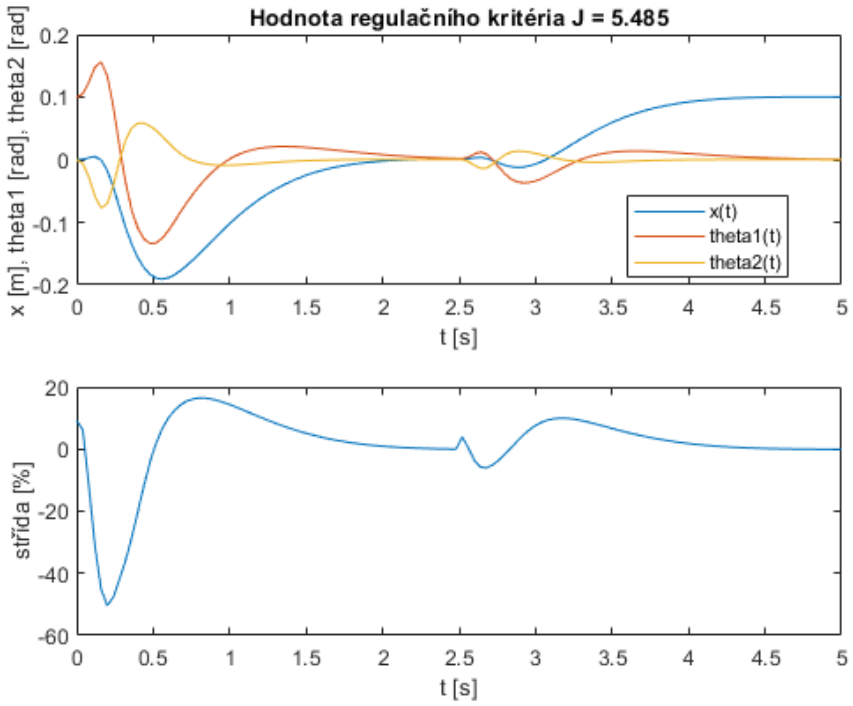
Obrázek 48 - Regulační pochod MPC, Q = [5 0.1 10 0.01 10 0.01 0 0], lambdau = 0.01, N2 = 60, N1 = 1, Nu = 20



Obrázek 49 - Regulační pochod MPC, Q = [5 0.1 10 0.01 10 0.01 0 0], lambdau = 0.01, N2 = 60, N1 = 1, Nu = 8



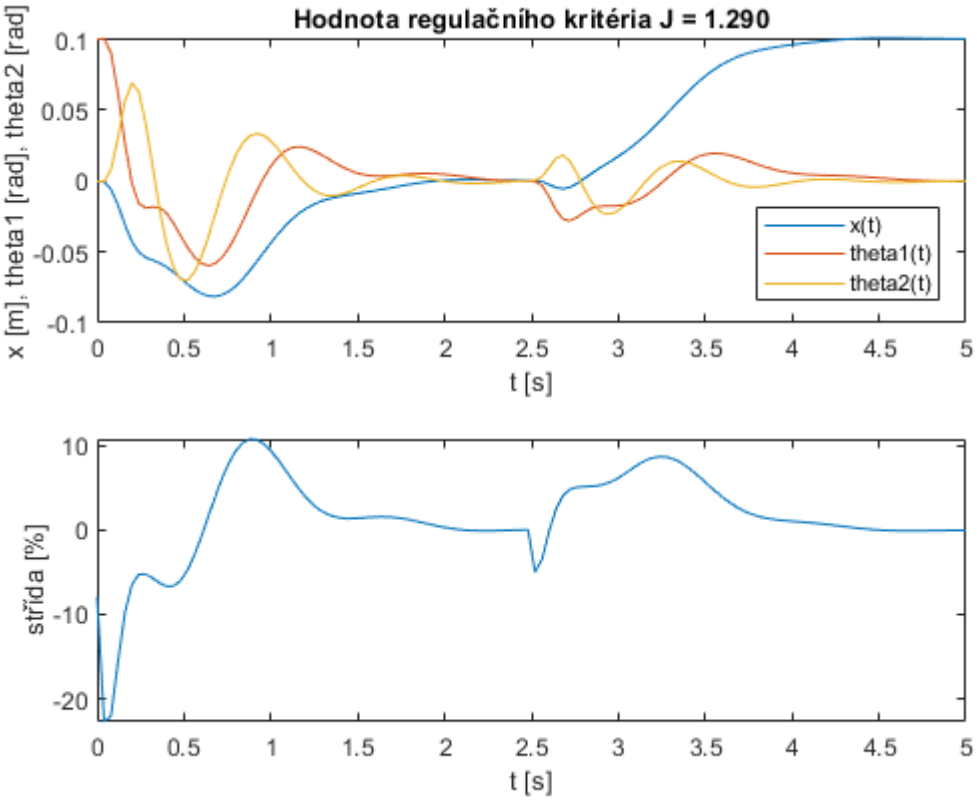
Obrázek 50 - Regulační pochod MPC, Q = [10 1 3 2 3 2 0 0], lambdau = 0.2, N2 = 60, N1 = 1, Nu = 20, pomalý regulační pochod



Obrázek 51 - Regulační pochod MPC, Q = [5 0.1 10 0.01 10 0.01 0 0], lambdau = 0.2, N2 = 60, N1 = 1, Nu = 20, změna žádané polohy vozíku v čase 2,5 s na 0,1 m

5.4.4 Ostatní rovnovážné polohy

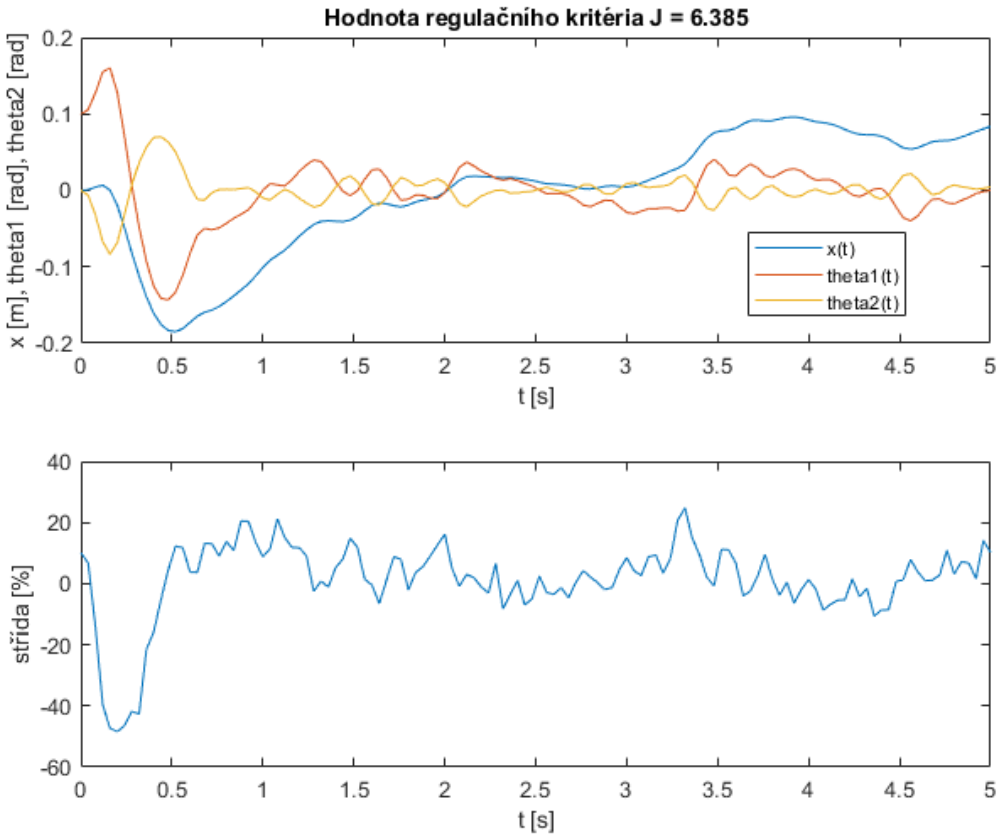
Ve vytvořených skriptech je na výběr z poloh Down/Down, Up/Up, Up/Down, Down/Up. Lze tedy vyzkoušet regulaci pro všechny 4 (pro každou se počítá nový regulátor). Animace zobrazuje korektní pozice ramen pro všechny polohy.



Obrázek 52 – Ukázka regulačního pochodu MPC pro polohu ramen Up/Down, změna žádané polohy vozíku v čase 2,5 s na 0,1 m

5.4.5 Zavedení šumu měření a suchého tření do simulace

Regulátory lze testovat i se zavedením šumu pro měřené veličiny. Změna se musí provádět v simulačním schématu. Zavedením šumu působící na měřené veličiny dostáváme následující chování.

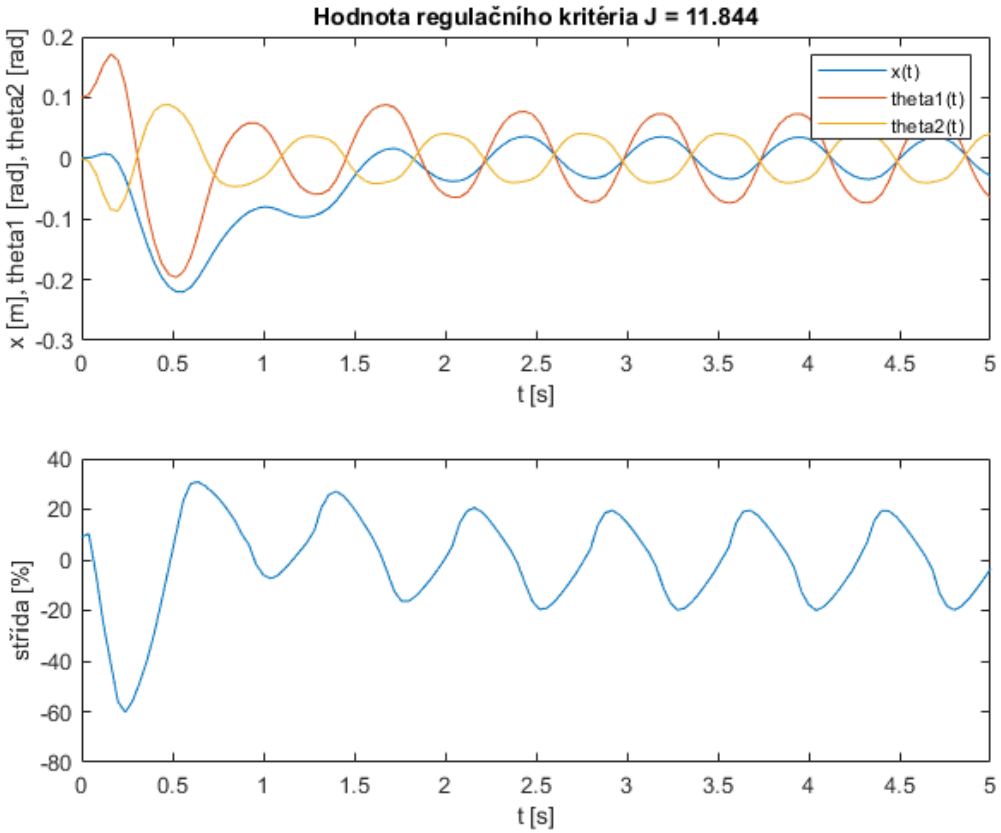


Obrázek 53 - Ukázka regulačního pochodu MPC, působení šumu na měřené veličiny, změna žádané polohy vozíku v čase 2,5 s na 0,1 m

Zajímavé chování poté nastává při zavedení suchého tření ramen do simulace – musí se nastavit v simulaci v bloku kyvadla, podsystémy *Eq_theta1* a *Eq_theta2*.

Obrázek 54 – Část schématu pro zavedení suchého tření

Regulace se dostává do tzv. limitního cyklu. Systém nikdy nedojde do ustáleného stavu, ale bude ho pořád „obíhat“ ve stavovém prostoru.



Obrázek 55 - Ukázka regulačního pochodu MPC, zavedení suchého tření

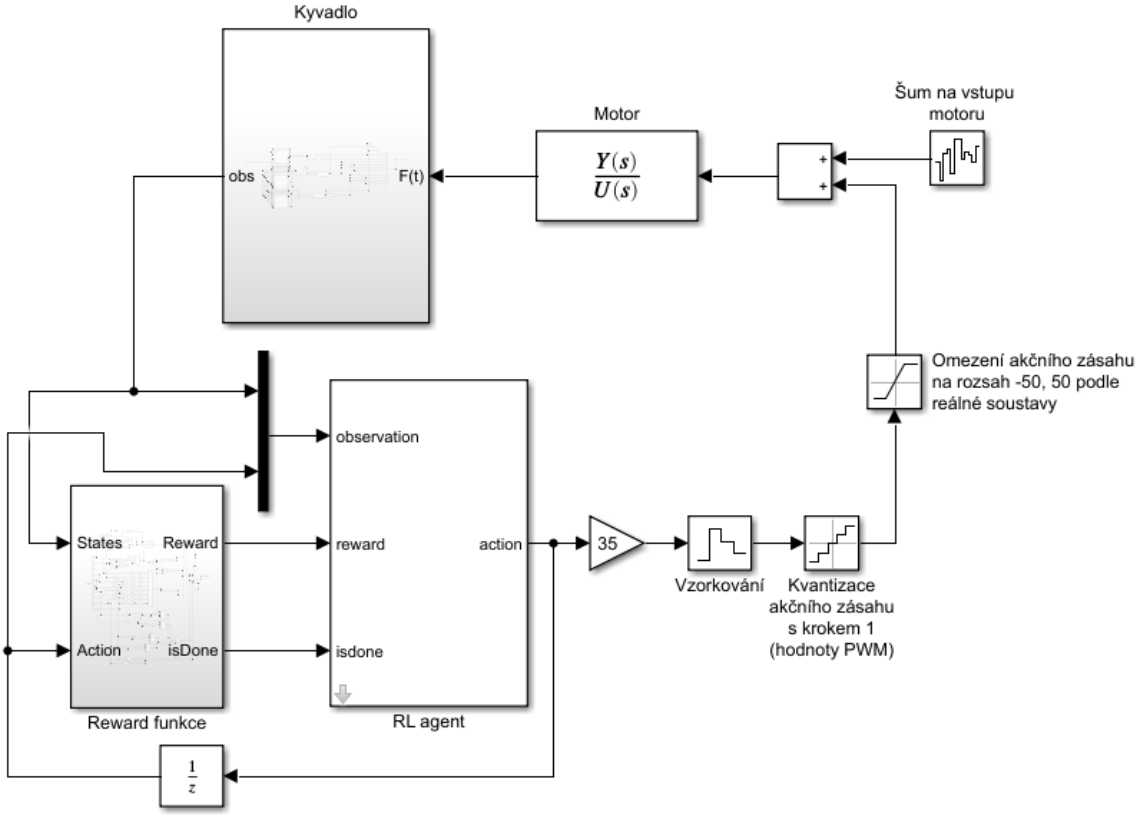
Ke kompenzaci cyklického chování existují teorie o přídavných kompenzačních regulátorech [25, 26]. Ty však v této práci využity nebyly.

5.5 Simulace – Swing-up

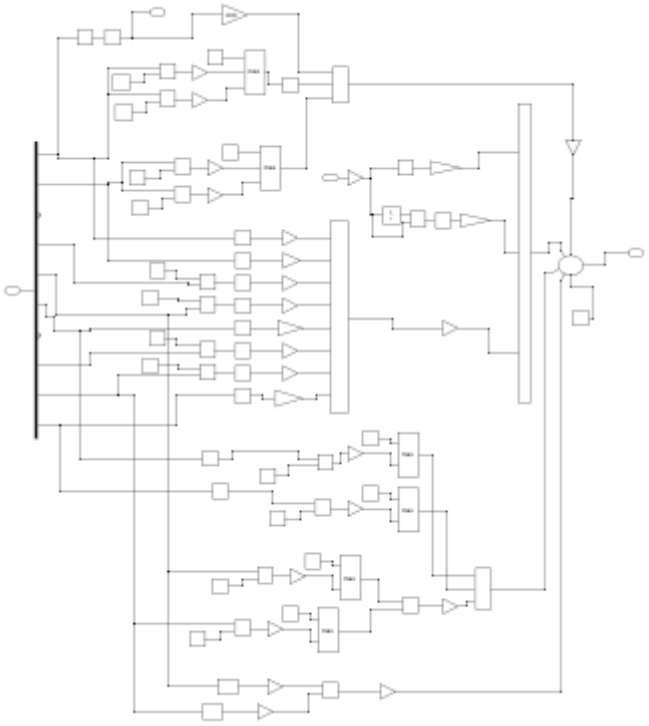
K provedení swing-up bylo využito strojového učení s využitím agenta TD3. Celý proces aplikace této metody se skládal z několika iterací s velkou časovou náročností. Nejprve byly testovány jednoduché modely kyvadla bez tření, vycházelo se z jednoduchých hodnotících funkcí, používaly se jednoduché neuronové sítě pro Actor a Critic. Byly použity funkce Matlabu z Reinforcement Learning Toolboxu.

Postupně se přicházelo s takovými parametry, které dávaly lepší a lepší výsledky. Swing-up byl na původním modelu nakonec úspěšný a přešlo se na složitější model (blíže reálné soustavě). Do modelu bylo zavedeno suché tření, výstupní hodnoty modelu byly vzorkovány s periodou 0,02 s a kvantovány podle reálného rozlišení snímačů - kvantizační krok úhlů natočení $2pi/(2^{14})$ [rad] a rotačního enkodéru 1/33365 [m]. Na vstup motoru byl přidán náhodný šum.

Reward funkce musela být několikrát upravena. Výsledné části systému strojového učení vypadají následovně:



Obrázek 56 – Simulační schéma pro učení agenta



Obrázek 57 – Pohled na strukturu Reward funkce

U Reward funkce byly postupně doplňovány části, které dodatečně zlepšovaly kvalitu učení agenta. Celá funkce se dostala do strukturálně složité podoby, kterou je možno si blíže prohlédnout ve schématu v přiložených souborech. Základní části lze ovšem popsat následovně:

Penalizace agenta pro: odchylku vozíku od nulové polohy, překročení rychlosti 1,2 m/s, velký akční zásah, velká diference akčního zásahu (rozdíl oproti minulé hodnotě)

Odměna agenta pro: polohu kyvadla blízkou Up/Up (v rozsahu 0,3 rad), nízké úhlové rychlosti ramen v blízkosti Up/Up, vzdálenost od polohy Down/Down, pasivní odměna v každé periodě vzorkování

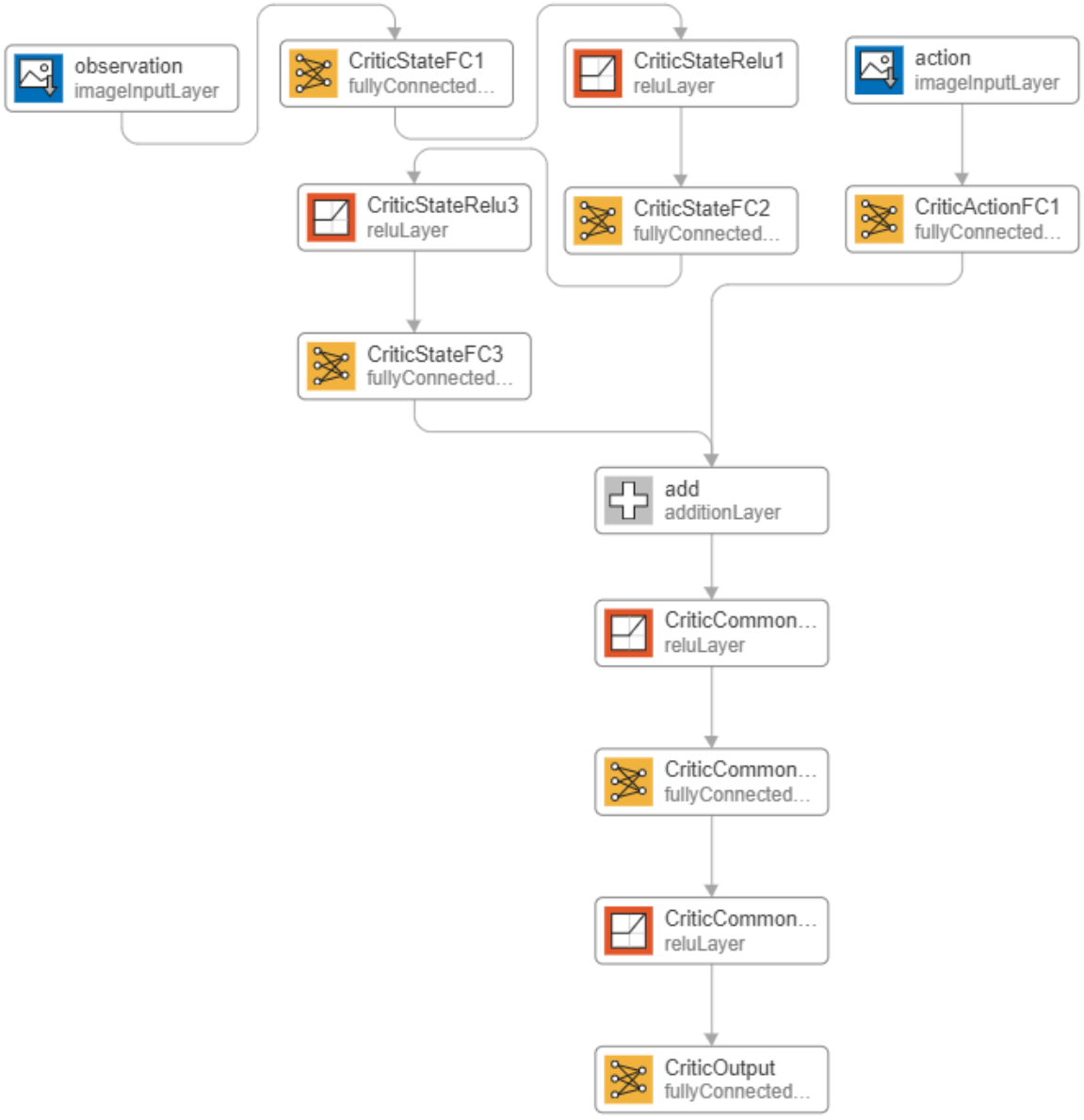
Všechny penalizace a odměny jsou spojité funkce – čím větší odchylka, tím větší penalizace a čím blíže jsme požadované hodnotě – tím větší odměna.

Epizoda je ukončena, pokud poloha vozíku překročí 0,6 m (reálné omezení ze soustavy) nebo uplyne zvolený čas simulace.

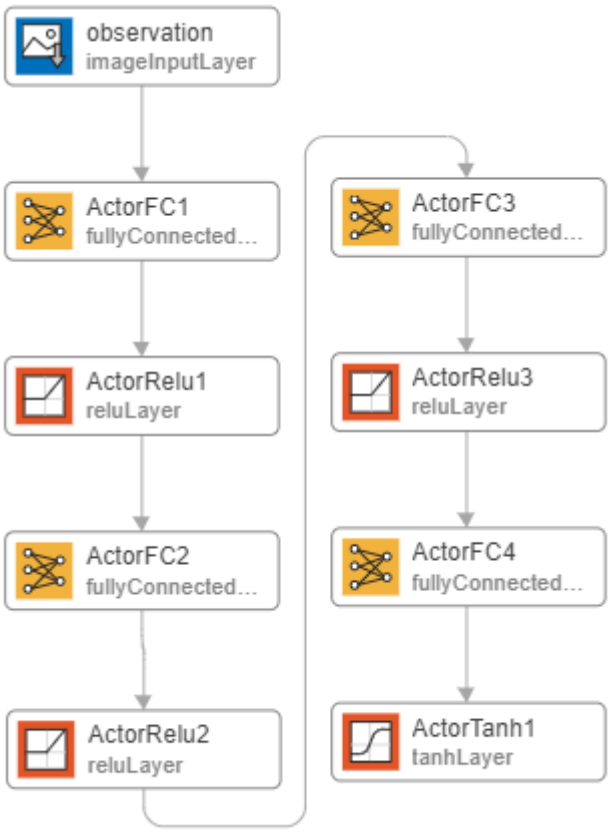
Vstupy agenta jsou: poloha vozíku a jeho derivace, absolutní úhly natočení obou ramen, jejich sinus, cosinus, derivace sinu úhlů a předchozí akční zásah: *x, x_dot, theta1wraped,*

theta1sin, theta1cos, theta1sin_dot, theta2wraped, theta2sin, theta2cos, theta2sin_dot, u(k-1)

Výstupem agenta je poté akční zásah v rozsahu *<-1, 1>*, který je násoben konstantou 35. Tato konstanta plyne z reálné soustavy, kde je akční zásah sice omezen až hodnotou 52, ale v našem případě chceme omezit maximální rychlosti vozíku při swing-up (s ohledem na možnou nepředvídatelnost chování).



Obrázek 58 – Architektura neuronové sítě Critic



Obrázek 59 – Architektura neuronové sítě Actor

Počet neuronů vrstev *FC* je v našem případě 300, vrstva *CriticCommon* má 15 neuronů, jenom vrstva *ActorFC3* má 30 neuronů. Ve vytvořeném skriptu je možno měnit velikosti těchto vrstev a při znalosti práce s neuronovými sítěmi v Matlabu i architekturu. Zvyšování počtu neuronů má za následek delší dobu učení, menší počet může způsobit nedostatečnou aproximaci reward funkce a problém se stane neřešitelným.

Pro optimalizaci vah NN je použit SGDM (Stochastic Gradient Descent with Momentum) se základními parametry z Matlabu.

Tabulka 8 – Parametry pro optimalizaci vah NN

Parametr	**Hodnota**
LearnRate Critic	0,005
LearnRate Actor	0,001
GradientThreshold	1
L2RegularizationFactor	0,0001

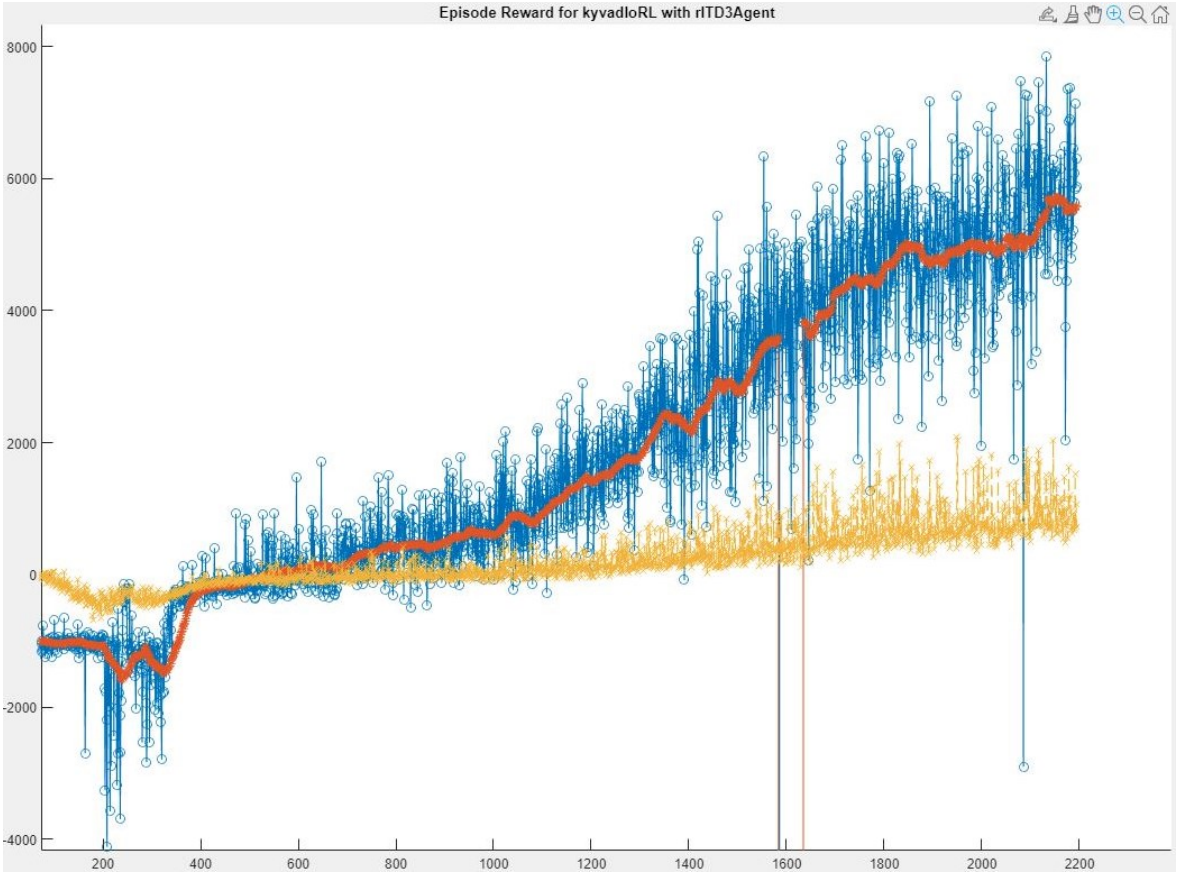
Tabulka 9 – Parametry agenta, pokud zde není některý z parametrů uvedený, je ponecháno základní nastavení z Matlabu

Parametr	**Hodnota**
MiniBatchSize	64
NumStepsToLookAhead	8
ExperienceBufferLength	60000
DiscountFactor	0,96
ExplorationModel Variance	0,2
ExplorationModel VarianceDecayRate	0
TargetPolicySmoothModel Variance	0,2
TargetPolicySmoothModel VarianceDecayRate	0

Učení probíhá do té doby, než je ukončeno zadaným maximálním počtem epizod nebo uživatelem (vidíme, že se výsledky už delší dobu nezlepšují). Během učení se automaticky ukládají agenti, kteří dosáhli na požadovanou hodnotu Reward funkce. V našem případě byla určena hodnota 32000, která vede na uložení desítek použitelných agentů. Toto číslo by mohlo být zvýšeno o pár tisíc, pokud bychom chtěli ukládat jen velmi dobré agenty. Ukládají se do složky *savedAgents*.

Při testování a vylepšování celého procesu byly vygenerovány stovky agentů. Vybraní agenti byli testováni nejprve v simulaci pro několik swing-up pokusů s působením náhodných poruch. Nejlepší z nich byli poté implementováni do simulace pro reálnou soustavu a nakonec byl vybrán jeden, který fungoval jak pro simulaci, tak pro reálnou soustavu. V přiloženém souboru se nachází pouze finální agent, protože celá složka s agenty měla téměř 5 GB a taková příloha by byla nevhodná.

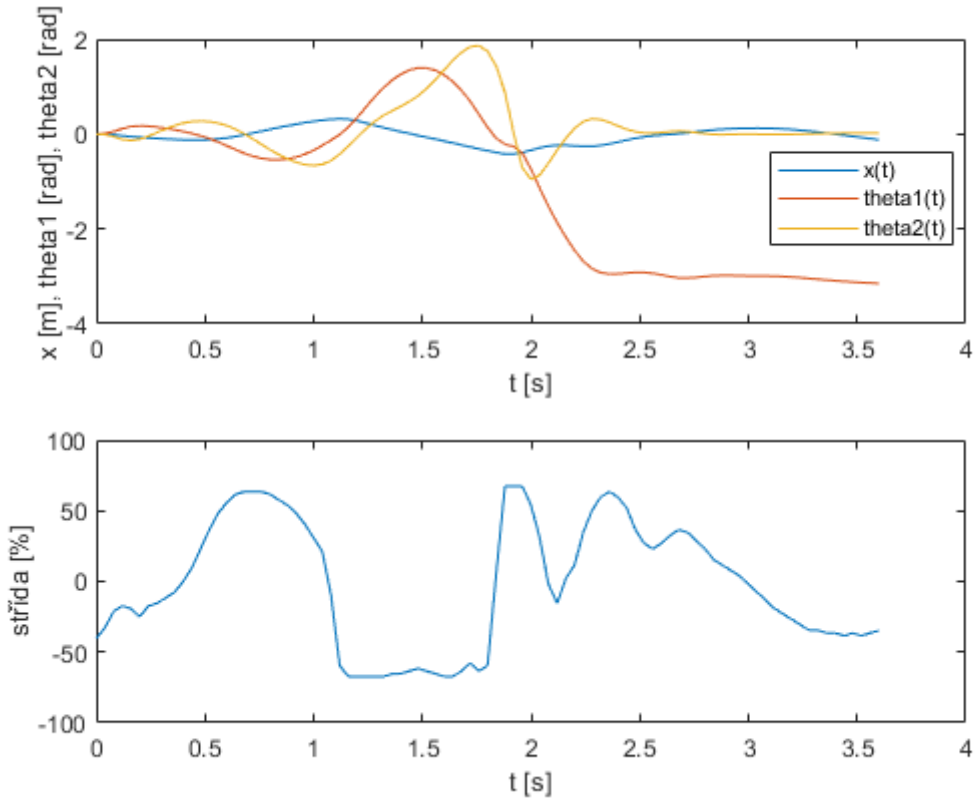
Následující graf zobrazuje možný průběh učení. S ohledem na stochastičnost celého procesu je pravděpodobné, že s každým opakováním dostaneme odlišný průběh. Zároveň se jedná o graf, který byl pořízen před finální verzí učení a slouží spíše jako ukázka.



Obrázek 60 – Průběh odměny pro jednotlivé epizody, graf získaný z prostředí Matlab – RL episode manager, osa x představuje číslo epizody, osa y pak získanou odměnu (reward)

Strojové učení má zajímavý vývoj chování – v časném stádiu učení dochází pouze k rozhoupávání kyvadla, později se dostává do fáze, kdy ramena periodicky „krouží“ kolem vozíku a nakonec se snaží zastavit ramena v poloze Up/Up. Celková doba procesu učení závisí na výkonu CPU, může trvat až deset hodin. Jako výpočetní jednotku lze použít i GPU, ale Matlab podporuje jen vybrané typy. Z vlastního experimentování jsem nepozoroval značnou změnu při využití GPU místo CPU. Průběhy jednotlivých epizod lze sledovat v simulaci v bloku kyvadla.

Byly vytvořeny 2 skripty - *uceni_TD3.m* a *swingUpSimulace.m*. Jak z názvu vypovídá, první ze skriptů slouží k učení nových agentů úplně od začátku. Druhý skript využívá finálního agenta a ukazuje swing-up jako graf, a také jako animaci. Na reálné soustavě je poté využito přepnutí swing-up na navržený stabilizující regulátor po dosažení polohy Up/Up.



Obrázek 61 – Průběh swing-up v simulaci

6 REGULACE NA REÁLNÉ SOUSTAVĚ

Při implementaci řízení pro reálnou soustavu nastalo několik problému. Vytvořená simulace nepracovala správně při využití Kalmanova filtru. Byla proto upravena tak, aby ho nebylo potřeba. S tímto omezením se nedaly nasadit regulátory, které byly vypočítány v předchozích kapitolách. K něčemu nám ovšem pořád byly – sloužily jako základ k již známým metodám diferenciální evoluce. Proto byl vytvořen další skript pro identifikaci *IdentifikaceEvoluceLQR.m*, který využívá vypočtený regulátor z LQG jako počáteční odhad. V jeho okolí se poté hledají stabilizující řešení pro zvolenou polohu.

Ve skriptu je možnost nastavení počátečního stavu, počáteční odchylky ramen, nastavení váhové matice (podobně jako u LQG) a penalizace akčního zásahu. Dále je zde volba doby simulace. Při počátečním hledání je vhodné volit krátkou dobu (5 s) a s hledáním lepšího a lepšího řešení tuto dobu zvětšovat. Ostatní parametry se týkají samotné evoluce viz kapitola 5.3.1.

Simulace, ve které probíhá hledání regulátoru, využívá co nejpřesnější model – stejně jako u strojového učení je zde tření, kvantizace, šum na výstupu a následná filtrace.

6.1 Simulace pro řízení reálné soustavy

6.1.1 Inicializační skript a přepočet hodnot ze senzorů

Inicializace potřebných parametrů pro všechny simulační schémata (viz níže) se nachází v jednom skriptu (*parametryRealnaSoustava.m*). Opět zde existují stavy (*state*), které představují rovnovážné polohy. Obsahují ale navíc možnost *swing-up*. Ve všech stavech je určena maximální možná odchylka od rovnovážné polohy (*th1limit, th2limit*), než se regulátor vypne („spadnutí“ kyvadla z určitých mezí má za následek vypnutí regulace, předejde se tak nežádoucímu chování soustavy). Dále jsou zde hodnoty pro regulátor, které byly získány v identifikaci.

Jelikož snímače úhlů natočení měří absolutní polohu a magnety nejsou přesně uloženy tak, aby v poloze Down/Down měřily senzory 0/0, musela být provedena úprava hodnot. Při regulaci přepočítáváme hodnoty snímačů tak, aby zvolená rovnovážná poloha představovala hodnotu 0/0 a zároveň musí obsahovat kladné i záporné odchylky v hodnotách radiánů (snímače posílají data pouze v kladných číslech 0 až $(2^{14}-1)$). Proto byla při měření na reálné soustavě odečtena dolní poloha kyvadel, která dávala hodnoty senzorů 12555 a 12540.

S využitím této znalosti byly dopočítány všechny ostatní rovnovážné polohy a celkový interval 0 – (2^{14} - 1) převeden na nový interval -π až π. Převod měřené hodnoty na odchylku v radiánech pak v simulaci obstarává tabulka, kde indexem do této tabulky je přijatá hodnota ze snímače a hodnotou v tabulce je požadovaná odchylka od rovnovážné polohy. Dále je zde vypočítaný krok enkodéru – metrem byly přesně odměřeny dva body od krajní polohy (enkodér v nule) a z enkodéru se určily hodnoty pro tyto body. Podílem naměřené vzdálenosti a hodnoty na enkodéru se vypočítal kvantizační krok.

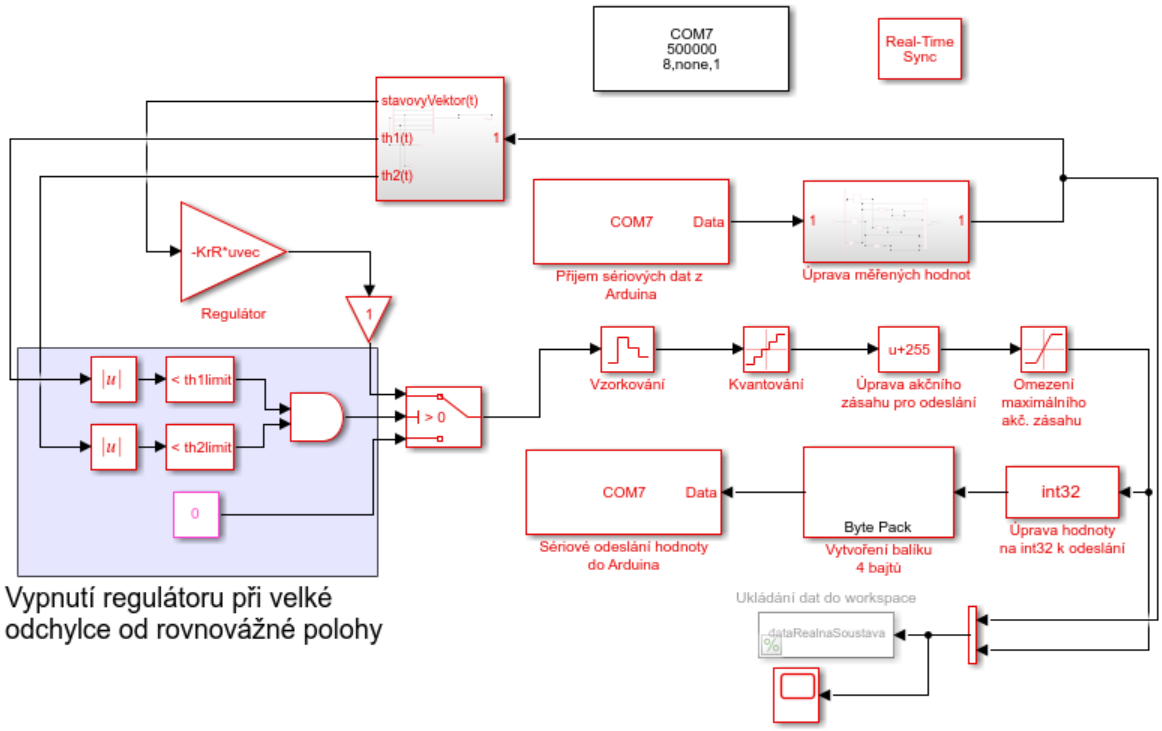
Poslední parametry ve skriptu se vztahují k inicializaci polohy vozíku – možnost nastavení rychlosti vozíku při nalézání středu dráhy a vzdálenost středu od koncového snímače.

6.1.2 Simulační modely

V Simulinku je využíváno dostupného rozšíření *Simulink Support Package for Arduino Hardware*.

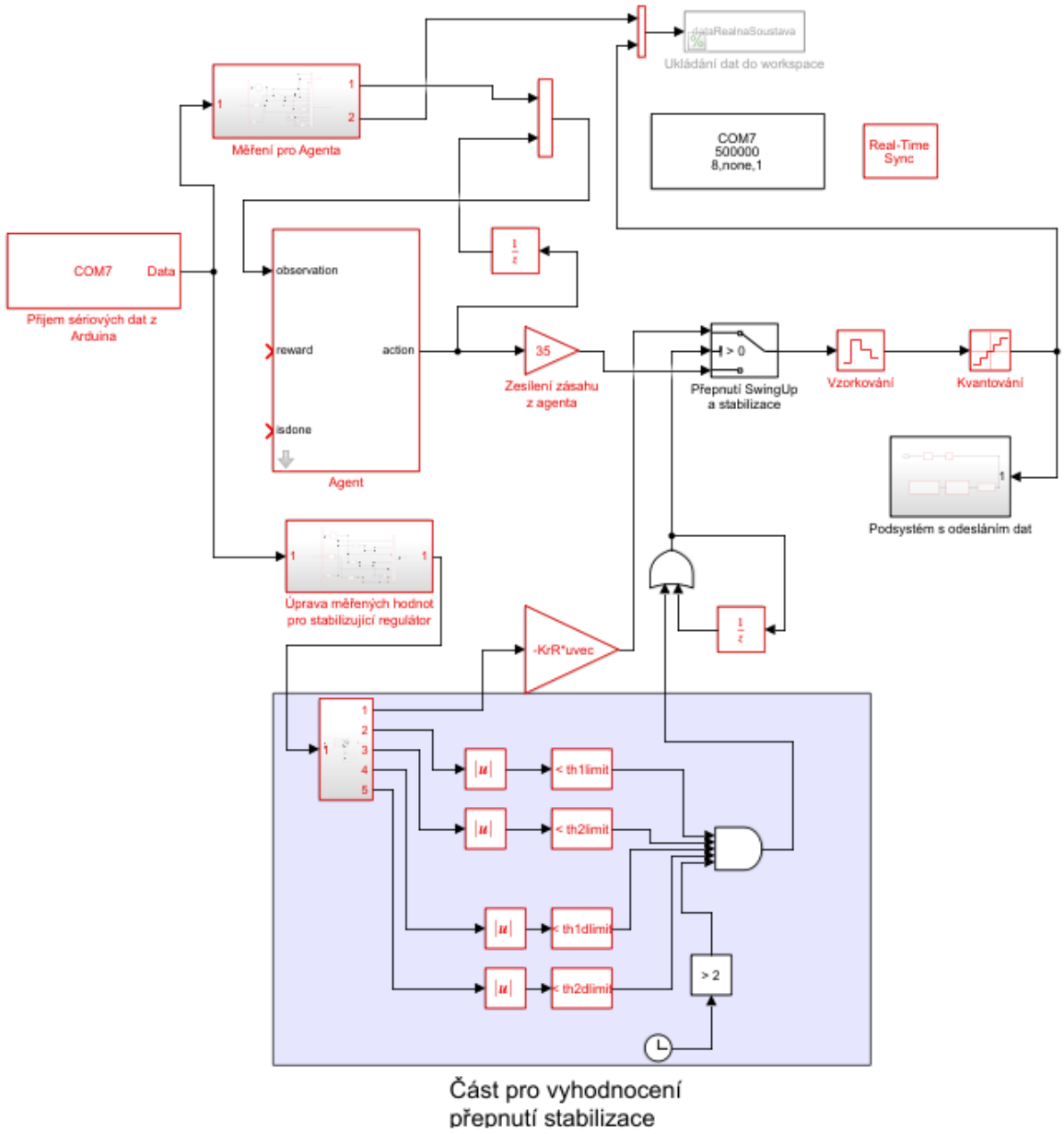
Pro práci s reálnou soustavou byly vytvořeny tři modely. Prvním z nich je *initPozice.slx*, který slouží k nalezení středu dráhy. Vozík se po třech vteřinách rozjede na levou stranu, než dojede na koncový snímač. Při kontaktu se nuluje hodnota na enkodéru a vozík se rozjíždí na druhou stranu. Podle hodnoty enkodéru je naměřena přesná vzdálenost od koncového snímače a vozík se zastaví v požadovaném bodě.

Druhý model *stabilizace.slx* se využívá pro stabilizaci kyvadla ve zvoleném rovnovážném bodu.



Obrázek 62 – Model pro stabilizaci kyvadla

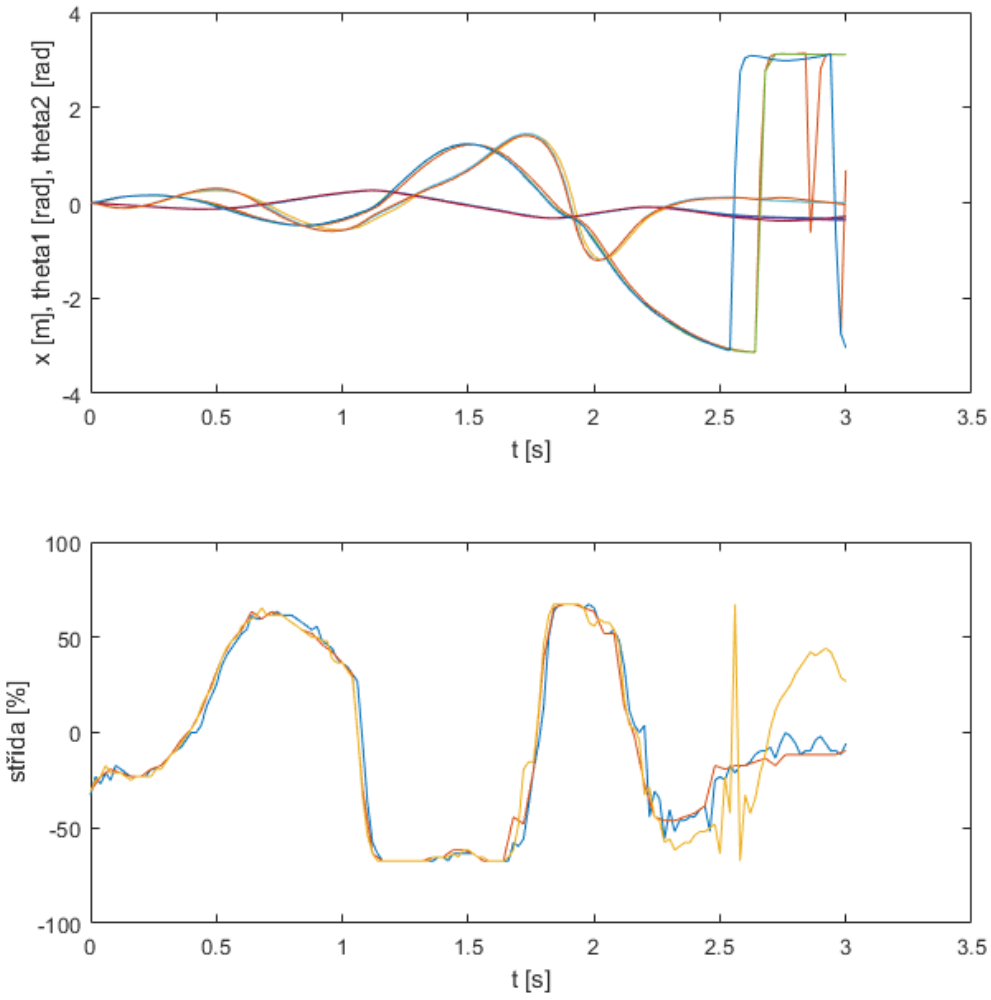
Třetí model *swingUp.slx* pracuje s přepínáním strojového učení (agenta) a normálního regulátoru pro stabilizaci (Up/Up). Experimentálně byly určeny hodnoty, při kterých se regulace přepíná: $\theta_1 < 0{,}32$ rad, $\theta_2 < 0{,}15$ rad, $\theta_{1\_dot} < 1{,}2$ rad/s, $\theta_{2\_dot} < 1{,}2$ rad/s



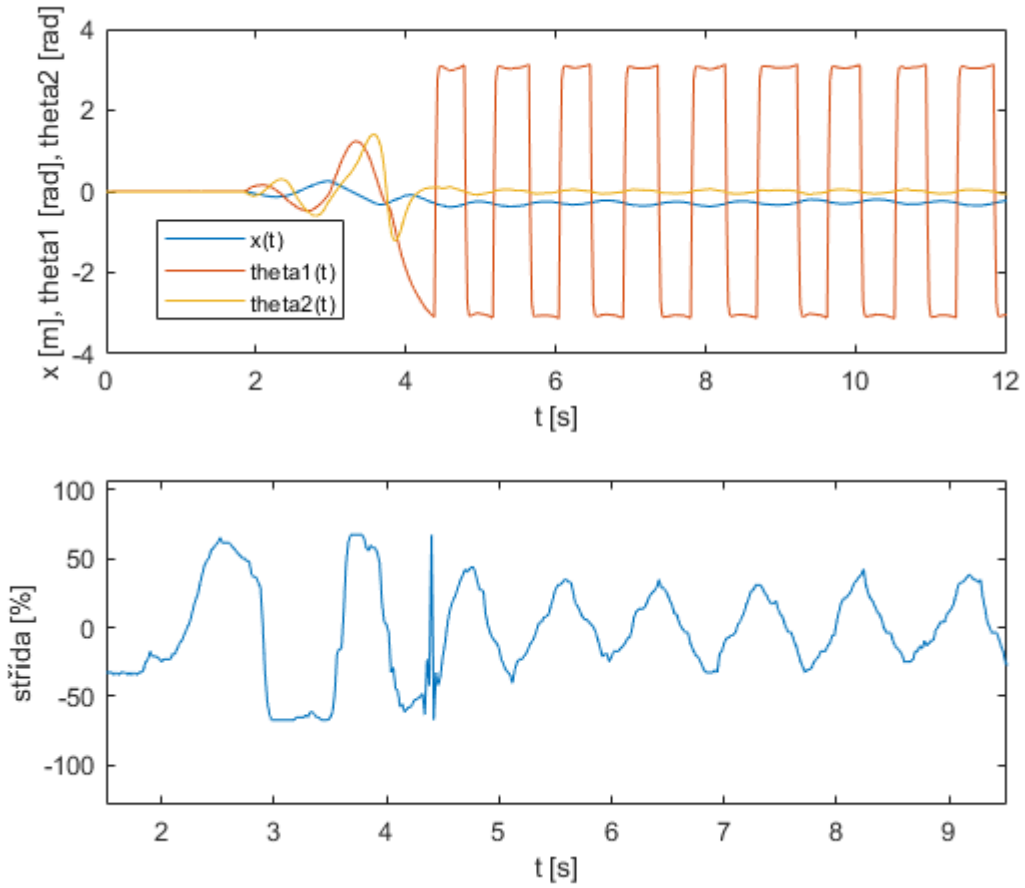
Obrázek 63 – Model pro swing-up a následnou stabilizaci

6.1.3 Swing-up a stabilizace

Následující graf zobrazuje tři různá měření pro swing-up. Jednotlivé průběhy jsou si velmi podobné. Zároveň se blíží i datům získaným v předchozích simulacích. Jelikož se v reálné simulaci začíná z polohy Down/Down pro snímače úhlu natočení, hodnoty v horní poloze Up/Up se kvůli absolutnímu snímání úhlu mění z -π na π. Proto je v grafickém vyjádření vidět „skoky“ mezi těmito hodnotami po dokončení swing-up. V obrázku (*Obrázek 65*) pak můžeme vidět swing-up i s následnou stabilizací. V čase přibližně 4,5 s je ze střídy patrné přepnutí regulátoru.



Obrázek 64 – Tři různé průběhy swing-up na reálné soustavě



Obrázek 65 – Průběh swing-up a následná stabilizace

V elektronické příloze se nachází jak programy, tak videa stabilizace a swing-up manévru.

ZÁVĚR

Cílem práce bylo vytvoření reálného modelu dvojitého inverzního kyvadla a pomocných modelů v programu Matlab/Simulink pro otestování řízení. Simulační část proběhla bez problémů. Na reálné soustavě se vyskytly komplikace, které byly průběžně řešeny.

Komplikace vznikaly použitím nekvalitního motoru, který by byl při dalších úpravách nahrazen. Dále byl problém se snímáním úhlu natočení, kdy při rychlých pohybech u swing-up manévru občas docházelo k impulsovému rušení a tím i nepřesnému provedení manévru. Byly otestovány různé druhy filtrů, které nakonec zlepšily kvalitu snímaných dat.

Velkou časovou náročnost vyžadovala část při vytváření simulací pro strojové učení. Bylo prováděno mnoho pokusů a iterací, než bylo nalezeno vhodné řešení. Problém byl v nedostatečné dokumentaci k danému problému. Strojové učení by mohlo být rozšířeno tak, že by zároveň dokázalo po swing-upu stabilizovat kyvadlo a nedocházelo by k přepínání na stabilizující regulátor.

Reálná soustava byla řízena Arduinem a výpočet akčního zásahu probíhal v Real-Time Simulinku běžícím na PC. Zapojení bylo testováno celou dobu na jednom zařízení a není potvrzena kompatibilita s ostatními PC. Problém by vyřešila implementace celého řídicího programu přímo do mikropočítače. Proveditelnost řešení by musela být blíže zkoumána.

Další vylepšení by mohla přinést úprava zpětnovazebního učení. Při správném nastavení by mělo být možné přecházet z libovolné nestabilní polohy do jiné nestabilní polohy v jednom algoritmu. S touto myšlenkou bylo experimentováno pouze simulačně. Zároveň by šlo aplikovat kompenzační regulátory pro tření.

Práce spojovala více oborů do jedné komplexní úlohy. Jednalo se o vytváření technické dokumentace pro výrobu, mechanickou montáž, modelování a 3D tisk součástek, návrh vhodných snímačů, identifikaci systému a využití poznatků z teorie řízení, strojového učení a nakonec i programování.

Výsledkem je soustava umožňující stabilizaci v různých nestabilních polohách a funkční swing-up manévr. Bylo dokázáno, že využití zpětnovazebního učení pro návrh swing-up je možné.

SEZNAM POUŽITÉ LITERATURY

[1] Brunton, S., & Kutz, J. (2019). *Data-Driven Science and Engineering: Machine Learning, Dynamical Systems, and Control*. Cambridge: Cambridge University Press. doi:10.1017/9781108380690

[2] JELEN, Michal. *Analýza stability lineárních systémů*. Brno, 2012. Bakalářská práce. Vysoké učení technické v Brně. Fakulta elektrotechniky a komunikačních technologií.

[3] CHEN, Xianmin, Rongrong YU, Kang HUANG, Shengchao ZHEN, Hao SUN a Ke SHAO. Linear motor driven double inverted pendulum: A novel mechanical design as a testbed for control algorithms. *Simulation Modelling Practice and Theory* [online]. 2018, **81**, 31-50 [cit. 2020-07-10]. DOI: 10.1016/j.simpat.2017.11.009. ISSN 1569190X. Dostupné z: https://linkinghub.elsevier.com/retrieve/pii/S1569190X17301703

[4] SLABÝ, Vít. *Návrh a realizace demonstračního modelu dvojitého kyvadla*. Brno, 2018. Diplomová práce. Vysoké učení technické v Brně. Fakulta strojního inženýrství.

[5] LANGER, Jiří a Jiří PODOLSKÝ. *Studijní text k přednášce:* Teoretická mechanika [online]. Matematicko – fyzikální fakulta, Univerzita Karlova v Praze, 2013 [cit. 2020-07-10]. Dostupné z: http://utf.mff.cuni.cz/vyuka/OFY003/TEXTY/lagrange.pdf

[6] BOGDANOV, Alexander. Optimal Control of a Double Inverted Pendulum on a Cart [online]. USA, 2004 [cit. 2020-07-10]. Dostupné z:https://www.semanticscholar.org/paper/Optimal-Control-of-a-Double-Inverted-Pendulum-on-a-Bogdanov/5638d1ff5eac6f7bf50c22aebcaf27c772a65f79

[7] SHAH, Nita a Mahesh YEOLEKAR. Pole Placement Approach for Controlling Double Inverted Pendumum. *Global Journal of Researches in Engineering*. 2013. ISSN 2249-4596. Dostupné z: https://www.researchgate.net/publication/239949494_Pole_placement_approach_for_controlling_double_inverted_pendulum

[8] OGATA, Katsuhiko. *Modern Control Engineering*. 4th ed. Upper Saddle River,NJ: Prentice-Hall, 2002. ISBN 978-0130609076.

[9] XU, Bowen, You LYU a Stephen Andrew GADSDEN. Estimation And Control Of A Double-Inverted Pendulum. In: *Progress in Canadian Mechanical Engineering* [online]. York University Libraries, 2018, 2018-05-30, [cit. 2020-07-15]. DOI: 10.25071/10315/35250. ISBN 9781773550237. Dostupné z:

http://hdl.handle.net/10315/35250

[10] BOBÁL, Vladimír. *Adaptivní a prediktivní řízení*. Zlín, 2007. Skripta. Univerzita Tomáše Bati ve Zlíně.

[11] Camacho, E. F., Bordons, C. Model Predictive Control. Second Edition. Advanced Textbooks in Control and Signal Processing. London, Springer-Verlag, 2004. ISBN 978-0-85729-398-5

[12] ZHANG, Su Ying, Ying WANG, Jie LIU a Xiao Xue ZHAO. Adaptive Neural Network Fuzzy Control Plane Double Inverted Pendulum. *Advanced Materials Research* [online]. 2013, **765-767**, 2004-2007 [cit. 2020-07-15]. DOI: 10.4028/www.scientific.net/AMR.765-767.2004. ISSN 1662-8985. Dostupné z: https://www.scientific.net/AMR.765-767.2004

[13] XIONG, Xin a Zhou WAN. The simulation of double inverted pendulum control based on particle swarm optimization LQR algorithm. In: *2010 IEEE International Conference on Software Engineering and Service Sciences* [online]. IEEE, 2010, 2010, s. 253-256 [cit. 2020-07-20]. DOI: 10.1109/ICSESS.2010.5552427. ISBN 978-1-4244-6054-0. Dostupné z: http://ieeexplore.ieee.org/document/5552427/

[14] JUNG, Sooyong a John T. WEN. Nonlinear Model Predictive Control for the Swing-Up of a Rotary Inverted Pendulum. *Journal of Dynamic Systems, Measurement, and Control* [online]. 2004, **126**(3), 666-673 [cit. 2020-07-21]. DOI: 10.1115/1.1789541. ISSN 0022 0434. Dostupné z: https://www.researchgate.net/publication/228732856_Nonli near_Model_Predictive_Contol_for_the_Swing-Up_of_a_Rotary_Inverted_Pendulum

[15] WU, Yun Feng, Zhu MING a Ke Chang FU. Feedforward and Feedback Control of an Inverted Pendulum. *Advanced Materials Research* [online]. 2011, **328-330**, 2194-2197 [cit. 2020-07-21]. DOI: 10.4028/www.scientific.net/AMR.328-330.2194. ISSN 1662-8985.

[16] GRAICHEN, Knut, Michael TREUER a Michael ZEITZ. Swing-up of the double pendulum on a cart by feedforward and feedback control with experimental validation. *Automatica* [online]. 2007, **43**(1), 63-71 [cit. 2020-07-21]. DOI: 10.1016/j.automatica.2006.07.023. ISSN 00051098. Dostupné z: https://linkinghub.elsevier.com/retrieve/pii/S0005109806003050

[17] XIN, Xin. Analysis of the Energy Based Swing-up Control for a Double Pendulum on a Cart. *IFAC Proceedings Volumes* [online]. 2008, **41**(2), 4828-4833 [cit. 2020-07-21]. DOI: 10.3182/20080706-5-KR-1001.00811. ISSN 14746670. Dostupné z: https://linkinghub.elsevier.com/retrieve/pii/S1474667016397063

[18] SUTTON, S. Richard a G. Andrew BARTO. *Reinforcement learning*. London: MIT Press, 2018. ISBN 978026203924.

[19] Enkodér ER – 40 Matsushita. *Eltra.it* [online]. [cit. 2020-07-21]. Dostupné z: http://www.eltra.it/products-eltra-encoder-for-motion-control/rotary-incremental-encoders/ottici-albero-sporgenti-en-gb/el-er-40-abchinx-en-gb//download/1069

[20] AS5047U. *ams.com* [online]. [cit. 2020-07-21]. Dostupné z: https://ams.com/as5047u#tab/features

[21] Indukční snímač SICK IME08. *sick.com* [online]. [cit. 2020-07-21]. Dostupné z: https://www.sick.com/cz/cs/indukcni-snimace/indukcni-snimace/ime/ime08 04npszt0k/p/p228419

[22] Motor driver 13A 6-30V DC. *cytron.io* [online]. [cit. 2020-07-21]. Dostupné z: https://www.cytron.io/p-13amp-6v-30v-dc-motor-driver

[23] Arduino Uno. *wikipedia.org* [online]. [cit. 2020-07-21]. Dostupné z: https://cs.wikipedia.org/wiki/Arduino_Uno

[24] Zdroj 6EP1334-1LB00. *siemens.com* [online]. [cit. 2020-07-22]. Dostupné z: https://mall.industry.siemens.com/mall/en/WW/Catalog/Product/6EP1334-1LB00

[25] CAMPBELL, Sue Ann, Stephanie CRAWFORD a Kirsten MORRIS. Friction and the Inverted Pendulum Stabilization Problem. *Journal of Dynamic Systems, Measurement, and Control* [online]. 2008, **130**(5) [cit. 2020-07-22]. DOI: 10.1115/1.2957631. ISSN 0022 0434. Dostupné z: https://www.researchgate.net/publication/228871763_Friction_and_the_Inverted_Pendulum_Stabilization_Problem

[26] NEJADFARD, Atabak, M. J. YAZDANPANAH a Iraj HASSANZADEH. Friction compensation of double inverted pendulum on a cart using locally linear neuro-fuzzy model. *Neural Computing and Applications* [online]. 2013, **22**(2), 337-347 [cit. 2020-07-22]. DOI: 10.1007/s00521-011-0686-3. ISSN 0941-0643. Dostupné z: http://link.springer.com/10.1007/s00521-011-0686-3

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

A	Ampér
A/D	Analogově/Digitální
CLK	Clock – synchronizační hodiny
CORDIC	Coordinate rotation digital computer
CPR	Counts Per Revolution – čtení na otáčku
CPU	Central Processing Unit – centrální procesorová jednotka
CRC	Cyclic redundancy check - Cyklický redundantní součet
DE	Diferenciální evoluce
GND	Ground - uzemnění
GPU	Graphics Processing Unit - grafický procesor
Hz	Hertz
I/O	Input/Output – Vstupní/Výstupní
KF	Kalmanův filtr
LED	Light-Emitting Diode - světelná dioda
LQG	Linear-quadratic-Gaussian
LQR	Linear-quadratic regulator
MISO	Multi Input Single Output – Více vstupů, jeden výstup
MOSI	Multi Output Single Input – Více výstupů, jeden vstup
MPC	Model Predictive Control – prediktivní řízení
NMPC	Nonlinear Model Predictive Control – nelineární prediktivní řízení
NN	Neural Network – neuronová síť
ODR	Obyčejná diferenciální rovnice
PA	Polyamid – druh plastu
PP	Pole Placement – umístění pólů

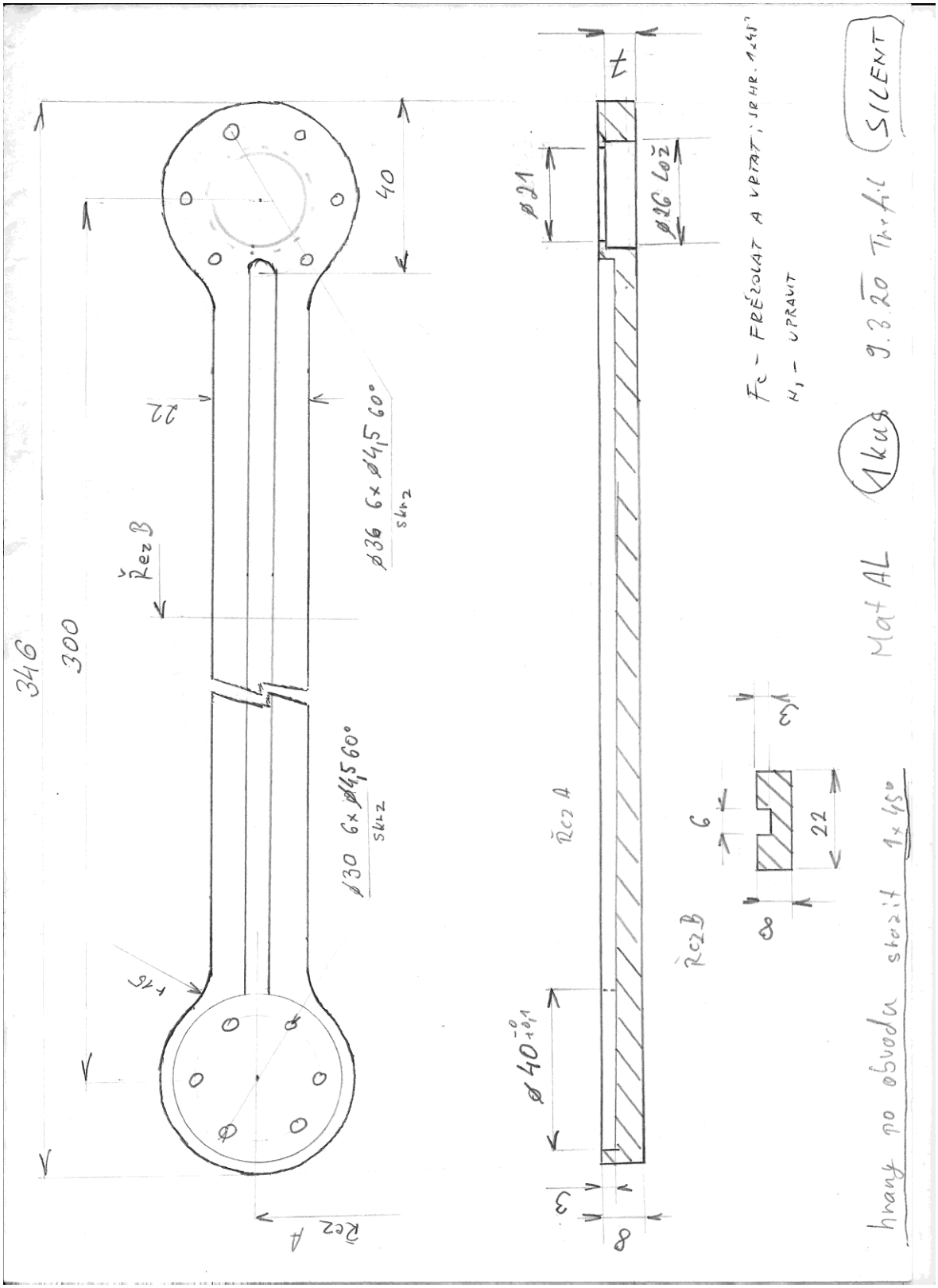
PPR	Pulses Per Revolution – pulzy na otočku
PSO	Particle Swarm Optimization – optimalizace rojením částic
PWM	Pulse Width Modulation - Pulzně šířková modulace
RL	Reinforcement Learning – zpětnovazební učení
SGDM	Stochastic Gradient Descent with Momentum
SPI	Serial Peripheral Interface - sériové periferní rozhraní
T_s	Sample Time – perioda vzorkování
USB	Universal Serial Bus - univerzální sériová sběrnice
Vac	Střídavé napětí
Vdc	Stejnosměrné napětí

SEZNAM OBRÁZKŮ

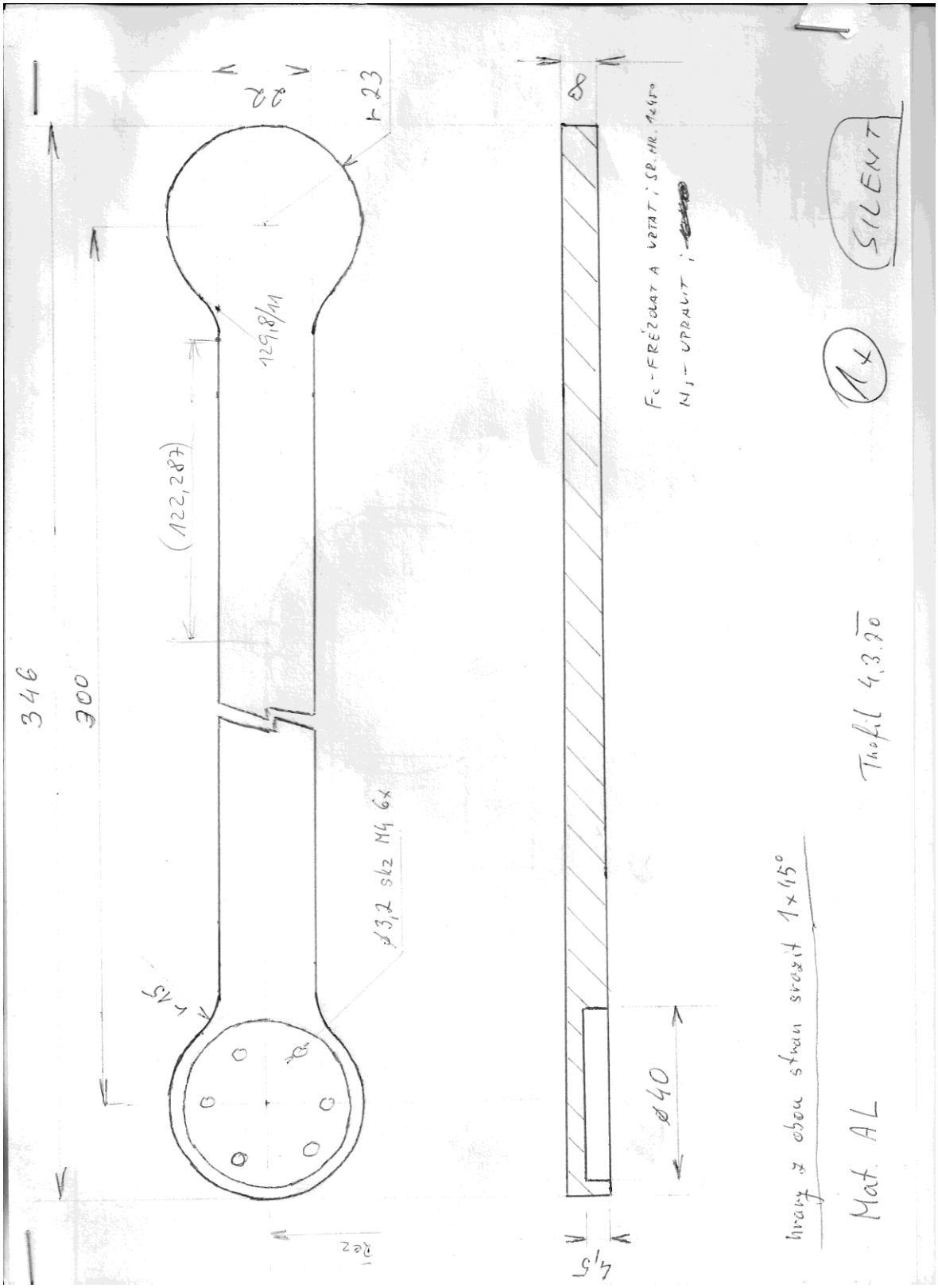
SEZNAM TABULEK

SEZNAM PŘÍLOH

PŘÍLOHA P I: TECHNICKÝ VÝKRES RAMENO 1



PŘÍLOHA P II: TECHNICKÝ VÝKRES RAMENO 2



PŘÍLOHA P III: TECHNICKÝ VÝKRES JÁDRO