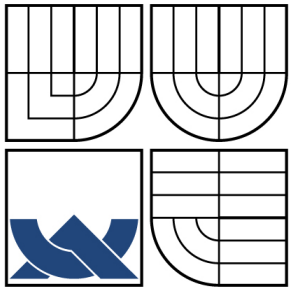
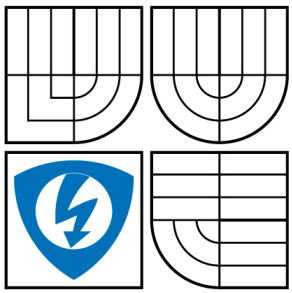


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

ÚSTAV AUTOMATIZACE A MĚŘICÍ TECHNIKY

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

DEPARTMENT OF CONTROL AND INSTRUMENTATION

ALGORITMY OPTIMÁLNÍHO ŘÍZENÍ POHONU SE STEJNOSMĚRNÝM MOTOREM

DC DRIVE OPTIMAL CONTROL

DIPLOMOVÁ PRÁCE

MASTER'S THESIS

AUTOR PRÁCE — Bc. DAVID REGENT

AUTHOR

VEDOUCÍ PRÁCE — doc. Ing. PAVEL VÁCLAVEK, Ph.D.

SUPERVISOR

BRNO 2008

VYSOKÉ UČENÍ
TECHNICKÉ V BRNĚ
Fakulta elektrotechniky
a komunikačních technologií
Ústav automatizace a měřicí techniky

Diplomová práce

magisterský navazující studijní obor
Kybernetika, automatizace a měření

Student: Regent David, Bc.
Ročník: 2

ID: 88522
Akademický rok: 2007/08

NÁZEV TÉMATU:

Algoritmy optimálního řízení pohonu se stejnosměrným motorem

POKYNY PRO VYPRACOVÁNÍ:

- Navrhněte algoritmus pro optimální řízení stejnosměrného servomotoru s ohledem na rychlost a energetickou náročnost regulace.
- Navržené algoritmy simulačně ověřte v prostředí Matlab-Simulink
- Algoritmy implementujte a ověřte na procesoru z rodiny Freescale 56F8300

DOPORUČENÁ LITERATURA:

Štecha, J.: Teorie dynamických systémů
Caha, Z., Černý,M.: Elektrické pohony
Šubrt, J.: Elektrické regulační pohony

Termín zadání: 3.12.2007

Termín odevzdání: 26.5.2008

Vedoucí projektu: doc. Ing. Pavel Václavek, Ph.D.

prof. Ing. Pavel Jura, CSc.
předseda oborové rady

UPOZORNĚNÍ:

Vysoké učení technické v Brně

Fakulta elektrotechniky a komunikačních technologií

Ústav automatizace a měřicí techniky

Algoritmy optimálního řízení se stejnosměrným motorem

Diplomová práce

Studijní obor:	Kybernetika, automatizace a měření
Autor práce:	Bc. David Regent
Vedoucí práce:	doc. Ing. Pavel Václavek, Ph.D.

Abstrakt:
Tato diplomová práce pojednává o problematice polohového řízení pohonu se stejnosměrným cize buzeným motorem. Veškeré simulace probíhaly v programovém prostředí Matlab-Simulink.
Práce obsahuje tvorbu matematického modelu motoru dvěma způsoby, pomocí simulačního schématu a S-funkce. Dále je zde provedena simulace obou modelů.
Dále se zabývá polohovým řízením pomocí kaskádní regulace.
V práci je zahrnuta teorie a návrh LQ regulátoru pomocí kvadratického kriteria. Simulace probíhala pro různá nastavení kvadratického kritéria.
Rozebírá teoretickou i praktickou ukázkou návrhu rekonstruktoru otáček. Je využit návrh pomocí Luenbergerova úplného rekonstruktoru.
Návrh regulátoru i rekonstruktoru je řešen na signálovém procesoru firmy Motorola (Freescale 56F8300). Programy jsou implementovány do programového prostředí CodeWarrioru.
Na závěr práce jsou shrnuty výsledky a porovnány simulace obou dvou regulátorů i rekonstruktoru.
Praktické výsledky odpovídají teoretickým předpokladům.

Klíčová slova:
cize buzený stejnosměrný motor, optimální řízení, LQ regulátor, rekonstruktor

Brno University of Technology

Faculty of Electrical Engineering and Communication

Department of Control, Measurement and Instrumentation

DC drive optimal control

Thesis

Specialisation of study: Cybernetics, Control and Measurement

Student: Bc. David Regent

Supervisor: doc. Ing. Pavel Václavek, Ph.D.

Abstract :

This master's thesis handles about problematic of positional control of the separate excitation DC machines. All simulations were progressed in programmatic environment Matlab-Simulink.
The work includes two kinds of a matematical model creation, with a simulation scheme and a S-function. There is also carried out simulations of both models.
The work deals with the positional control with help of a cascade control.
In document are encompass theory and plan of LQ controller with help of quadratic criterion. Simulation took place for dissimilar options quadratic criterion.
Theoretic and practice analysing plan state observer of revolutions. Luenberg's full state observer is exploit in plan.
Controller and state observer are adressed on signal processor from Motorola (Freescale 56F83000). Programmes are implement to programmatic enviroment CodeWarior.
At the end of the work are summarized the results of simulations and compared the two controllers and state obesrvers.
Practical results correspond to theoretical assumptions.

Key words:
separate excitation direct machine, optimal control, LQ controller, state observer

Bibliografická citace

REGENT, D. *Algoritmy optimálního řízení pohonu se stejnosměrným motorem* . Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2008. 71 s. Vedoucí diplomové práce doc. Ing. Pavel Václavek, Ph.D.

P r o h l á š e n í

„Prohlašuji, že svou diplomovou práci na téma „Algoritmy optimálního řízení pohonu se stejnosměrným motorem“ jsem vypracoval samostatně pod vedením vedoucího diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené diplomové práce dále prohlašuji, že v souvislosti s vytvořením této diplomové práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení § 152 trestního zákona č. 140/1961 Sb.“

V Brně dne : Podpis:

P o d ě k o v á n í

Děkuji tímto doc. Ing. Pavlu Václavkovi Ph.D. za cenné připomínky a rady při vypracování diplomové práce.

V Brně dne : Podpis:

OBSAH

SEZNAM OBRÁZKŮ

SEZNAM ZNAČEK A POUŽITÝCH SYMBOLŮ

Zkratka/Symbol	Jednotka	Popis
i_a	[A]	Proud kotvy
i_{an}	[A]	Normovaná hodnota proudu
i_{amax}	[A]	Maximální hodnota proudu
u_a	[V]	Napětí kotvy
u_{an}	[V]	Normovaná hodnota napětí
u_{amax}	[V]	Maximální hodnota napětí
i_b	[A]	Budící proud
u_b	[V]	Budící napětí
u_i	[V]	Vnitřní napětí
J	[kg·m^2]	Moment setrvačnosti
m_p	[N·m]	Pracovní moment
R_{ac}	[Ω]	Odpor kotvy
L_{ac}	[H]	Indukčnost kotvy
R_b	[Ω]	Odpor budícího obvodu
L_b	[H]	Indukčnost budícího obvodu
τ_{ac}	[s]	Elektromagnetická časová konstanta
τ_m	[s]	Mechanická časová konstanta
ω	[rad/s]	Úhlová rychlost
ω_n	[rad/s]	Normovaná

		hodnota rychlosti
ω_{max}	[rad/s]	Maximální hodnota rychlosti
e	[rad]	Úhlová odchylka
e_n	[rad]	Normovaná hodnota odchylky
e_{max}	[rad]	Maximální hodnota odchylky
A		Matice vazeb vnitřních stavů
B		Matice vazeb stavů na vstup
C		Matice vazeb výstupu na stavy
Q		Matice ovlivňující čas v kvadr. krit.
R		Matice ovlivňující energii v kvadr. krit.
P		Matice řešení Riccatiho rovnice
K		Vektor zpětnovazebních koeficientů
G		Matice nastavující rekonstruktor

1. ÚVOD

Elektromotor je stroj, měnící elektrickou energii na mechanickou práci. Většina elektromotorů pracuje na elektromagnetickém principu, ale existují i motory založené na jiných elektromechanických jevech. Základním principem, na němž jsou elektromagnetické motory založeny, je vzájemné silové působení vodičů, kterými protéká elektrický proud. Většina elektrických motorů je rotačních, ale existuje například i lineární elektromotor. V běžném rotačním motoru je umístěn rotor tak, aby vodiče rotoru a magnetické pole statoru vyvíjely točivý moment kolem osy rotoru. V rotačním motoru se rotující část (obvykle uvnitř) nazývá rotor a statická (pevná) část se nazývá stator.

Simulace je v matematice a kybernetice vědecká metoda, při které se zkoumají vlastnosti nějakého systému pomocí experimentů s jeho matematickým modelem. V minulosti se používaly pro modelování analogové počítače nebo simulační programovací jazyky. V posledním desetiletí se objevilo značné množství rozšířených verzí již existujících programů pro simulaci. Prvním krokem při počítačovém modelování je, že získáme nějakým způsobem matematický model zkoumaného systému. Matematický model musí vhodně charakterizovat závislost výstupů systému na jeho vstupech. Modely fyzikálních soustav jsou obvykle sestaveny jako soustavy několika diferenciálních rovnic. Rovnice modelu řešíme vhodnou numerickou metodou. Hlavní praktickou výhodou modelování je možnost pomocí pokusů a omylů vyřešit úlohy ověřit vlastnosti nákladných zařízení před jejich fyzickou realizací. K nejznámějším softwarovým programům patří například Matlab-Simulink pro modelování dynamických systémů.

Řízení je každé cílevědomé působení na řízený objekt s cílem dosáhnout předem daného stavu. Při řízení dynamických systémů je cílem, aby výstupní veličina co nejpřesněji sledovala průběh vstupní veličiny. Řídicí regulátor tedy musí zajistit sledování průběhu, což je obtížné vzhledem k časovému zpoždění, a kompenzovat poruchy. Vliv časového zpoždění lze zmenšit zavedením pomocné regulační veličiny.

V moderní teorii řízení jsou všechny požadavky na řízení shrnuty do kritéria kvality řízení a problém řízení je převeden na optimalizační problém minimalizace kritéria kvality řízení. Při tomto přístupu k řešení problému řízení existují dva zásadní problémy. Prvním problémem je vhodná volba kritéria kvality řízení, která by zahrnula všechny naše požadavky na kvalitu řízení. Druhým problémem je řešitelnost takto formulovaného optimalizačního problému. Nejvíce používaným kritériem kvality řízení je kvadratické kritérium, které pro lineární systémy vede na lineární zákon řízení.

Procesor je integrovaný obvod, který zajišťuje funkce počítače a je tak jeho základním kamenem. Výhody signálového procesoru jsou vysoká rychlost zpracování číslicových dat a velmi rychlé matematické operace a také schopnost zpracovávat velké objemy dat.

2. POHONY SE STEJNOSMĚRNÝMI MOTORY

2.1 POHONY SE STEJNOSMĚRNÝMI SÉRIOVÝMI MOTORY

Stejnosměrné sériové motory našly své využití v elektrické trakci, pro své výhody mechanické charakteristiky. V současné době se používají téměř výhradně ve spojení s polovodičovými měniči energie. Mechanický komutátor motoru se sběrným ústrojím je považován za určitý nedostatek tohoto motoru. Jsou snahy nahradit jej i v trakčních aplikačních pohonem střídavým. Ovšem stejnosměrný sériový motor zůstává v elektrické trakci stále dominující i přes snahu nahradit jej frekvenčně řízeným asynchronním motorem. Důvodem je, že podobného chování se střídavými pohony dosahujeme jen za cenu složitých regulačních a výkonových obvodů [1].

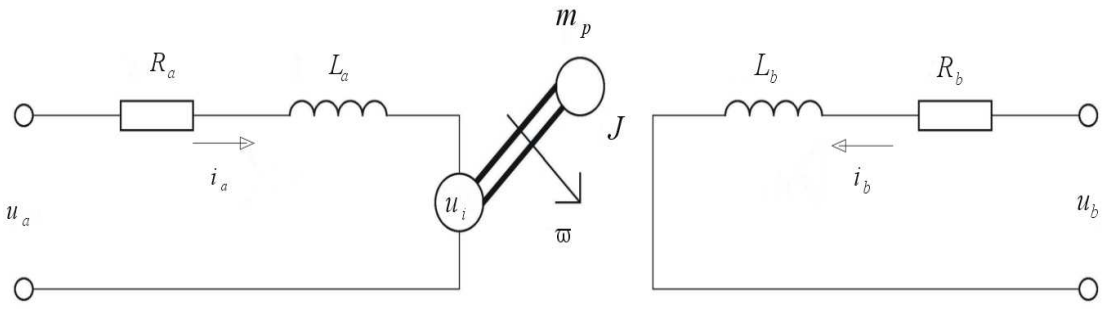
2.2 POHONY SE STEJNOSMĚRNÝMI CIZE BUZENÝMI MOTORY

Stejnosměrné cize buzené motory se používají téměř výhradně v regulačních pohonech pro nejrůznější aplikace ve spojení s polovodičovými měniči energie. Přes usilovnou snahu nahradit pohon se stejnosměrnými cize buzeným motorem ve spojení s polovodičovým měničem pohonem střídavým má tento pohon v oblasti regulačních pohonů stále dominantní postavení. Důvodem je celá řada jeho vlastností a relativně nízké pořizovací náklady. Jeho výhodou proti střídavým regulačním pohonům je jednoduché výkonové schéma a řízení měniče. Nezávislost řídicích vstupů budícího a kotevního vinutí motoru zjednodušuje návrh regulačních struktur a dovoluje dosáhnout snadné řiditelnosti pohonu v obou směrech otáčení ve všech pracovních režimech při širokém regulačním rozsahu [1]. Mechanický komutátor a sběrné ústrojí motoru však představuje nejslabší místo pohonu. Tento nedostatek společně s výkonovým omezením motoru vede ke snaze nahradit jej pohonem střídavým.

3. MODELOVÁNÍ STEJNOSMĚRNÉHO CIZE BUZENÉHO MOTORU

3.1 MATEMATICKÝ MODEL STEJNOSMĚRNÉHO CIZE BUZENÉHO MOTORU

Úplný matematický model stejnosměrného cize buzeného motoru by byl složitý. Proto se používají obvyklá zjednodušení. Zanedbává se rozptylový magnetický tok budícího vinutí, vliv reakce kotvy u kompenzovaných strojů, vzájemné transformační účinky jednotlivých vinutí, vliv vířivých proudů v magnetickém obvodu a úbytek napětí na kartáčcích. Vliv vířivých proudů se uplatňuje jen u větších motorů při rychlých změnách magnetického toku a vliv reakce kotvy jen u nekompenzovaných motorů. Za těchto předpokladů lze stejnosměrný motor podle Obr. 3.1 popsat rovnicemi:



Obr. 3.1: Obvodové schéma stejnosměrného cize buzeného motoru [1]

R_a, L_a … odpor a vinutí obvodu kotvy

R_b, L_b … odpor budícího vinutí a indukčnost budícího vinutí

i_a, u_a … proud a napětí obvodu kotvy

i_b, u_b … budící proud a napětí

u_i … vnitřní napětí

J … celkový moment setrvačnosti pohonu

m_p … pracovní moment

ω … úhlová rychlost pohonu

Pro elektrickou část pohonu lze podle II. Kirchhoffova zákona odvodit rovnici:

$$u_a = R_{ac}\, i_a + L_{ac} \frac{d\, i_a}{dt} + u_i\left(i_b, \omega\right) \qquad (3.1)$$

$$u_b = R_b\, i_b + L_b \frac{d\, i_b}{dt} \qquad (3.2)$$

$R_{ac,}\, L_{ac}$ … celkový činný odpor a indukčnost v obvodu kotvy

Pro mechanickou část pohonu lze napsat:

$$m_p = m - J \frac{d\, \omega}{dt} \qquad (3.3)$$

m … elektromagnetický moment motoru s magnetickým tokem závislým na proudu kotvy

Lze využít i dalších rovnic:

$$u_i = C\, \Phi\left(i_b\right)\omega \qquad (3.4)$$

$$m = C\, \Phi\left(i_a\right) i_a \qquad (3.5)$$

Po dosazení lze dostat pouze tři rovnice popisující stejnosměrný cize buzený motor:

$$u_a = R_{ac}\, i_a + L_{ac} \frac{d\, i_a}{dt} + C\, \Phi\left(i_b\right)\omega \qquad (3.6)$$

$$u_b = R_b\, i_b + L_b \frac{d\, i_b}{dt} \tag{3.7}$$

$$m_p = C\, \Phi\, (i_b) i_a - J \frac{d\, \omega}{dt} \tag{3.8}$$

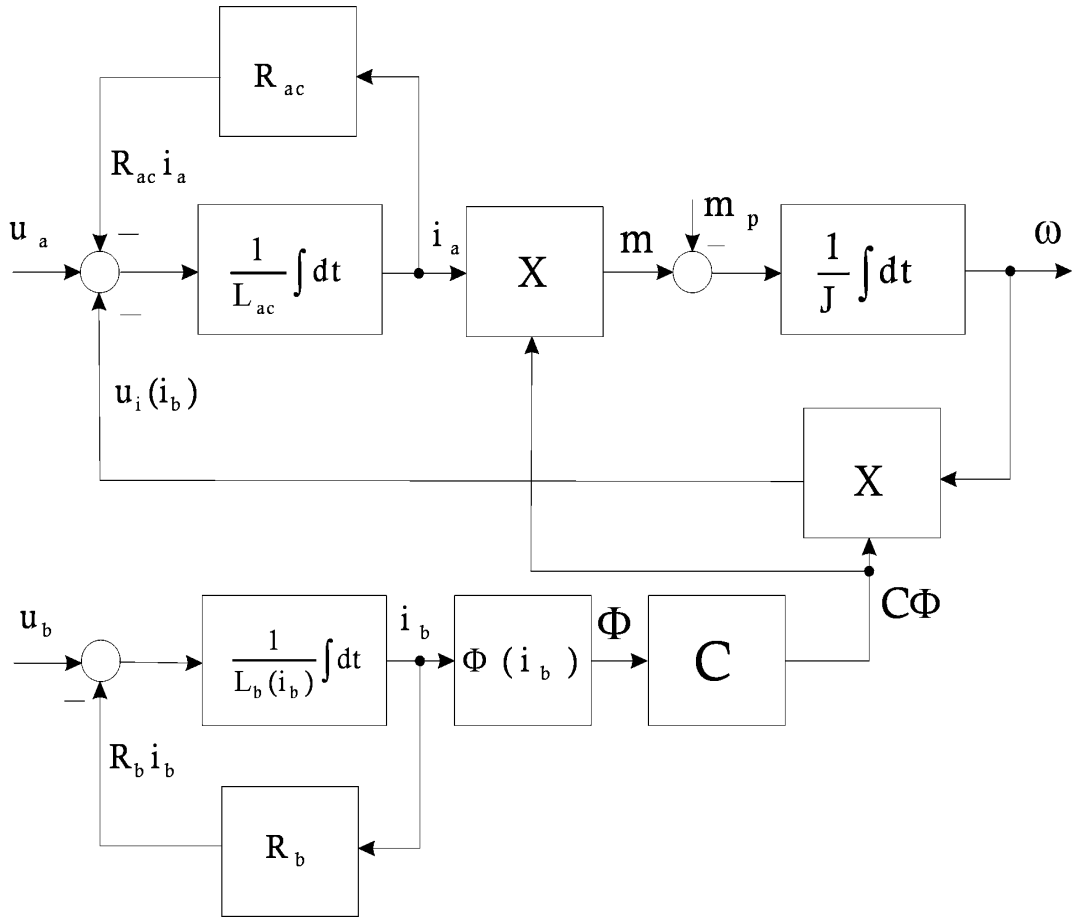
Rovnice (3.6) – (3.8) jsou matematickým modelem stejnosměrného cize buzeného motoru. Rovnice je výhodnější si přepsat do tvaru výpočtu derivací, z nichž lze sestavit blokové schéma stejnosměrného cize buzeného motoru:

$$\frac{d\, i_a}{dt} = \frac{u_a - R_{ac}\, i_a - C\, \Phi\, (i_b)\omega}{L_{ac}} \tag{3.9}$$

$$\frac{d\, i_b}{dt} = \frac{u_b - R_b\, i_b}{L_b} \tag{3.10}$$

$$\frac{d\, \omega}{dt} = \frac{C\, \Phi\, (i_b) i_a - m_p}{J} \tag{3.11}$$

Podle těchto rovnic lze sestavit blokové schéma, viz. Obr. 3.2:

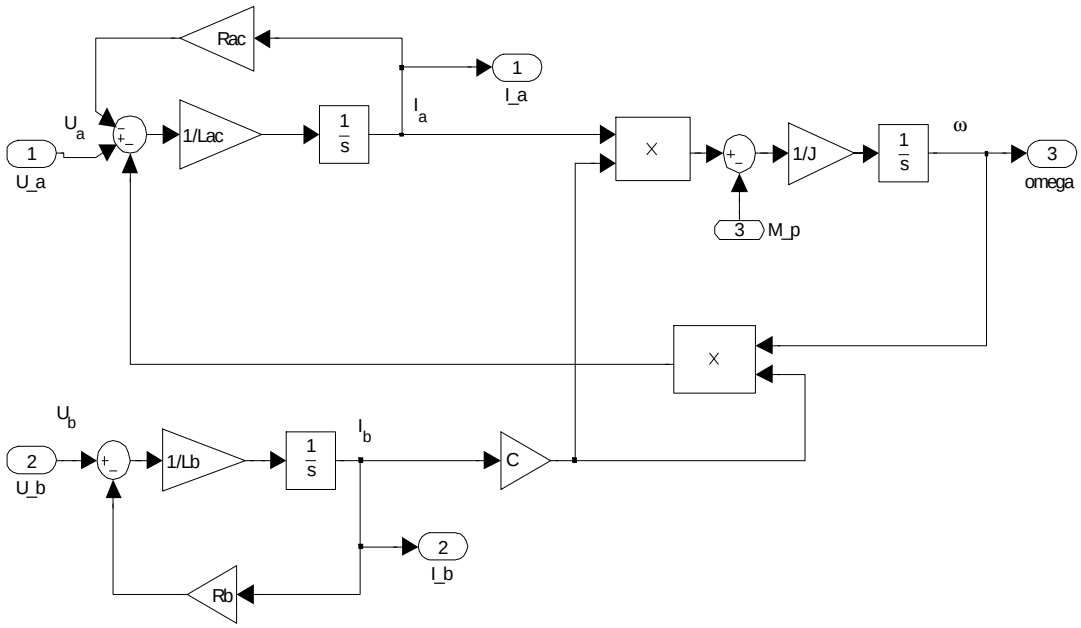


Obr. 3.2: Blokové schéma stejnosměrného cize buzeného motoru [1]

3.2 ŘEŠENÍ MATEMATICKÉHO MODELU STEJNOSMĚRNÉHO CIZE BUZENÉHO MOTORU

3.2.1 Využití Simulinku a tvorba modelu

Využilo se předešlých teoretických znalosti a blokové schéma z obrázku Obr. 3.2 bylo v Simulinku upraveno (viz. Obr. 3.3).



Obr 3.3: Blokové schéma stejnosměrného cize buzeného motoru v prostředí Simulink

Při modelování stejnosměrného cize buzeného motoru byl považován magnetizační obvod za lineární, tudíž využívám se využívalo jen konstanty *C* pro charakteristiku materiálu. Byl přidán výstup proudu i_a a i_b pro konečnou kontrolu a také pro S-funkci.

Blokové schéma je o něco složitější, ale stále lze považovat za docela jednoduché, ale pro lajka přece jen nepřehledné. Z celého schématu lze vytvořit subsystém, který bude mít stejnou funkčnost. Je potřeba jej dobře zamaskovat. Stejnosměrný cize buzený motor je jednoduchý model, proto byly při maskování pojmenovány vstupy a výstupy, zadefinovány proměnné, s kterými se počítalo a napsal stručný help. Dále se zde daly zadefinovat počáteční podmínky, této možnosti bylo využito až v pozdějších kapitolách.

V masce je uveden název, stručný popis bloku a lze zadat parametry R_{ac}, L_{ac}, R_b, L_b, C a J. Obyčejnému uživateli by tohle mělo postačit.

3.2.2 Využití Simulinku a S-funkce

Kromě blokového řešení v simulinku, lze stejnosměrný cize buzený motor také vyřešit pomocí využití rovnic a tzv. S-funkcí.

Využíval se předdefinovaný M-file „Level 1 M-file S-function Template“. Prvním krokem je pojmenovat vytvářenou funkci stejně jako celý výsledný M-file, pokud se tak nestane S-funkce nebude správně pracovat.

Pro stejnosměrný cize buzený motor bude základní deklarace vypadat následovně:

function [sys,x0,str,ts] = cizbuz(t,x,u,flag,Rac,Lac,Rb,Lb,C,J)

V inicializačním kroku byl nastaven vektor *sys*, aby obsahoval tři spojité stavové proměnné, tři vstupy, tři výstupy a počáteční podmínky nastavené do nuly. Postup byl následující, spočítány derivace, vráceno vektoru *sys*, zjištěny hodnoty výstupů a opět vráceno vektoru *sys*.

Posléze byl vytvořen potřebný M-file, využilo se opět služeb Simulinku. V bloku „S-function “ byl zadán název S-funkce a zadefinovány parametry. Bylo třeba přivést vstup jako vektor, obdobně i pro výstup.

Tímto by se dalo říct, že byla nastíněna problematika a způsob modelování pohonu se stejnosměrným cize buzeným motorem.

4. ŘÍZENÍ STEJNOSMĚRNÉHO CIZE BUZENÉHO MOTORU POMOCÍ KASKÁDNÍ REGULACE

4.1 ODVOZENÍ PŘENOSU ŘÍZENÍ

Je-li motor řízen změnou napětí na kotvě při ustáleném proudu I_b = konst., lze jej popsat obrazovým přenosem. K tomuto popisu je třeba rovnic (3.6) a (3.8), pouze byly přepíšeme pomocí Laplaceovy transformace:

$$u_a(p) = R_{ac} \cdot i_a(p) + p \cdot L_{ac} \cdot i_a(p) + C \cdot \Phi \cdot \omega(p) \quad (4.1)$$

$$m_p(p) = C \cdot \Phi \cdot i_a(p) - p \cdot J \cdot \varpi(p) \quad (4.2)$$

Obrazový přenos F(p) je definován jako podíl obrazu výstupu ku obrazu vstupu:

$$F(p) = \frac{\omega(p)}{u_a(p)} \quad (4.3)$$

Poruchovou veličinu m_p bylo třeba považovat za nulovou, poté pak lze upravit vzorce (4.1) a (4.2) a vhodně zobrazit přenos řízení stejnosměrného cize buzeného motoru. Nejdříve bylo vyjádřeno ω:

$$0 = C \cdot \Phi \cdot i_a(p) - p \cdot J \cdot \varpi(p) \Rightarrow$$
$$\Rightarrow \omega(p) = \frac{C \cdot \Phi \cdot i_a(p)}{p \cdot J} \quad (4.4)$$

Dále lze zjednodušit i druhý vzorec pro tvorbu přenosu řízení:

$$u_a(p) = i_a(p) \cdot (R_{ac} + p \cdot L_{ac}) + C \cdot \Phi \cdot \omega(p) =$$
$$= R_{ac} \cdot i_a(p) \cdot (1 + p \cdot \tau_{ac}) + \frac{(C \cdot \Phi)^2 \cdot i_a(p)}{p \cdot J} \tag{4.5}$$

Ve vzorci se objevuje elektromagnetická časová konstanta stejnosměrného cize buzeného motoru τ_{ac}, která byla dána:

$$\tau_{ac} = \frac{L_{ac}}{R_{ac}} \tag{4.6}$$

Nyní lze celkem jednoduše odvodit přenos řízení pro stejnosměrný cize buzený motor:

$$F(p) = \frac{\omega(p)}{u_a(p)} = \frac{\frac{C \cdot \Phi \cdot i_a(p)}{p \cdot J}}{R_{ac} \cdot i_a(p) \cdot (1 + p \cdot \tau_{ac}) + \frac{(C \cdot \Phi)^2 \cdot i_a(p)}{p \cdot J}} =$$
$$= \frac{\frac{C \cdot \Phi \cdot i_a(p)}{p \cdot J}}{\frac{R_{ac} \cdot i_a(p) \cdot p \cdot J \cdot (1 + p \cdot \tau_{ac}) + (C \cdot \Phi)^2 \cdot i_a(p)}{p \cdot J}} =$$
$$= \frac{\frac{C \cdot \Phi \cdot i_a(p)}{p \cdot J}}{\frac{i_a(p) \cdot (R_{ac} \cdot p \cdot J \cdot (1 + p \cdot \tau_{ac}) + (C \cdot \Phi)^2)}{p \cdot J}} =$$
$$= \frac{C \cdot \Phi}{R_{ac} \cdot p \cdot J \cdot (1 + p \cdot \tau_{ac}) + (C \cdot \Phi)^2} \cdot \frac{(C \cdot \Phi)^{-2}}{(C \cdot \Phi)^{-2}} =$$
$$= \frac{\frac{1}{C \cdot \Phi}}{\frac{R_{ac} \cdot p \cdot J}{(C \cdot \Phi)^2} + \frac{R_{ac} \cdot p^2 \cdot J \cdot \tau_{ac}}{(C \cdot \Phi)^2} + 1} = \frac{\frac{1}{C \cdot \Phi}}{p \cdot \tau_m + p^2 \cdot \tau_{ac} \cdot \tau_m + 1}$$

(4.7)

Zde se objevuje opět další časová konstanta stejnosměrného cize buzeného motoru τ_m, tzv. mechanická časová konstanta, která byla dána:

$$\tau_m = \frac{R_{ac} \cdot J}{(C \cdot \Phi)^2} \tag{4.8}$$

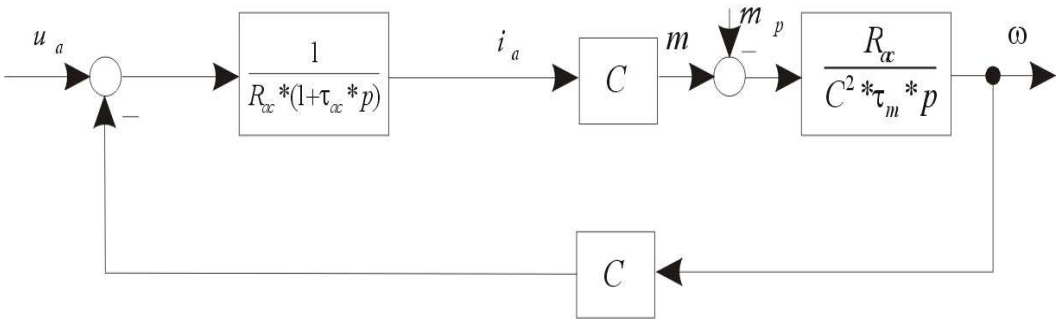
Z rovnice (4.2) lze vyjádřit obraz úhlové rychlosti $\omega(p)$:

$$\omega(p) = \frac{C \cdot \Phi \cdot i_a(p) - m_p(p)}{p \cdot J} = \frac{C \cdot \Phi \cdot i_a(p) - m_p(p)}{p \cdot \frac{\tau_m \cdot (C \cdot \Phi)^2}{R_{ac}}} = \frac{\left[C \cdot \Phi \cdot i_a(p) - m_p(p)\right] \cdot R_{ac}}{p \cdot \tau_m \cdot (C \cdot \Phi)^2} \tag{4.9}$$

Z rovnice (4.5) lze vyjádřit obraz proudu $i_a(p)$:

$$i_a(p) = \frac{u_a(p) - C \cdot \Phi \cdot \omega(p)}{R_{ac} \cdot (1 + p \cdot \tau_{ac})} \tag{4.10}$$

Předchozí rovnice byly vyjádřeny kvůli blokovému schématu stejnosměrného cize buzeného motoru, které vypadá následovně:



Obr. 4.1: Blokové schéma přenosu řízení [1]

4.2 KASKÁDNÍ ŘÍZENÍ A NÁVRHY REGULÁTORŮ

Při řízení příslušného motoru byla využita rozvětvená regulace s dvěma pomocnými regulovanými veličinami (proud kotvou motoru a otáčky). Základní regulovanou veličinou byla poloha osy motoru. Regulace proudu byla zároveň regulací momentu a spolu s regulací otáček slouží ke zvýšení stability a zlepšení dynamických vlastností polohového řízení.

Při návrhu regulátorů se využívá tvaru frekvenčních charakteristik otevřeného obvodu. S využitím Sisotoolu, lze s frekvenčními charakteristikami jednoduše pracovat a správně navrhnout příslušné regulátory. Přesnější popis návrhu je pospán v kapitole 4.2.2. Počínaje touto kapitolou se zde vyskytují již konkrétní přenosy, proto je uvedeno pro jaké hodnoty motoru byly regulátory navrhovány.

Parametry modelovaného stejnosměrného cize buzeného motoru:

R_{ac} = 0,8 Ω

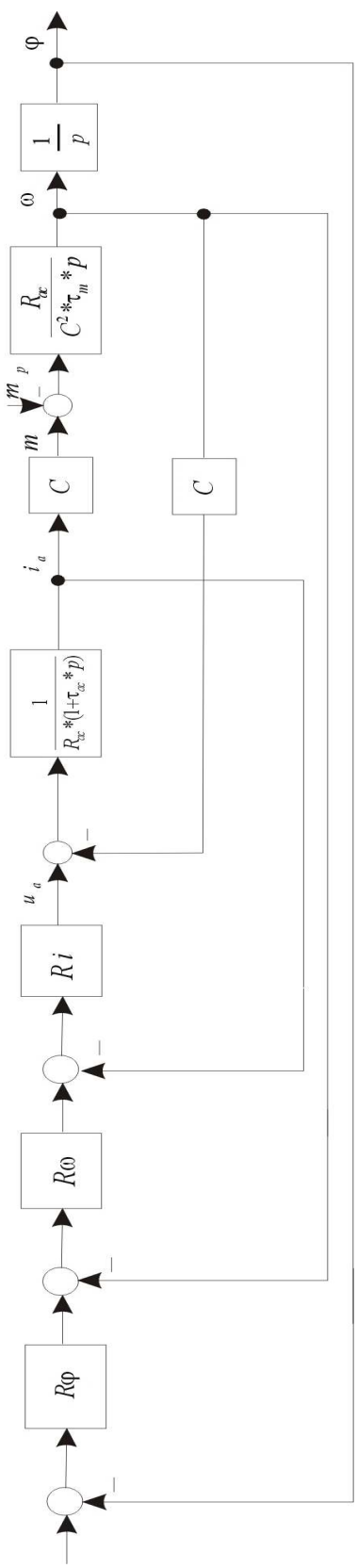
L_{ac} = 1 mH

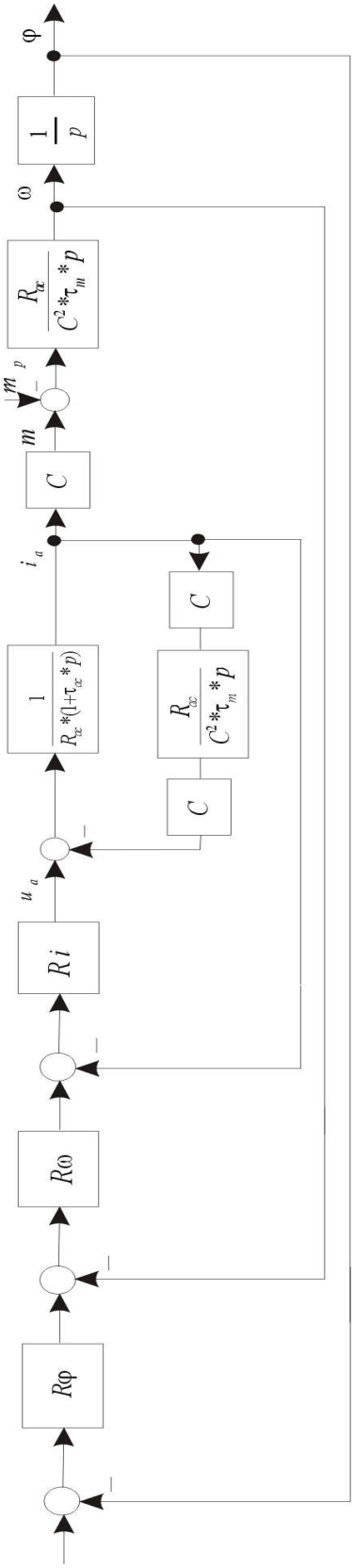
J = 0,0001 kg·m^2

C = 0,007

4.2.1 Kaskádní řízení

Na obrázku Obr. 4.2 je znázorněno schéma kaskádní regulace. V levé části lze vidět, že se zde objevuje jedna překřížená vazba, která musí být odstraněna. V pravé části již lze vidět, že překřížená vazby byla podle bakalářských znalostí odstraněna a již lze postupně navrhovat regulátory. Začíná se nejvnitřnější smyčkou a regulátorem proudu a postupuje se dále k vnějším smyčkám.


φ
1/p
ω
Rα/(C²*τm*p)
m
mp
C
C
ia
1/(Rα*(1+τα*p))
ua
Ri
Rω
Rφ


φ
1/p
ω
Rα/(C²*τm*p)
m
mp
C
ia
C
1/(Rα*(1+τα*p))
Rα/(C²*τm*p)
C
ua
Ri
Rω
Rφ

Obr. 4.2: Kaskádní regulace stejnosměrného cize buzeného motoru s původní překříženou vazbou (levá část) a její následné odstranění (pravá část)

4.2.2 Návrh prvního regulátoru (regulátor proudu) [3]

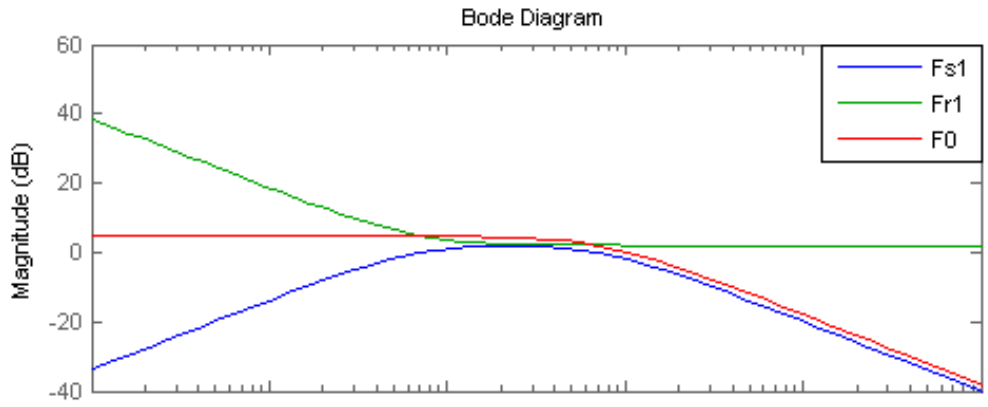
Z obrázku Obr. 4.2 byl spočítán přenos soustavy (podle blokové algebry), na kterou byl navržen regulátor proudu:

$$Fs_1 = \frac{F_1}{1 + F_1 \cdot F_2} = \frac{\dfrac{1}{R_{ac} \cdot (1 + p \cdot \tau_{ac})}}{1 + \dfrac{R}{p \cdot \tau_m} \cdot \dfrac{1}{R_{ac} \cdot (1 + p \cdot \tau_{ac})}} = \frac{\dfrac{1}{R_{ac} \cdot (1 + p \cdot \tau_{ac})}}{\dfrac{p \cdot \tau_m (1 + p \cdot \tau_{ac}) + 1}{p \cdot \tau_m \cdot (1 + p \cdot \tau_{ac})}} =$$

$$= \frac{p \cdot \tau_m}{R_{ac} \cdot p \cdot \tau_m \cdot (1 + \tau_{ac}) + R} = \frac{p \cdot \tau_m}{R \cdot \tau_m \cdot \tau_{ac} \cdot p^2 + R \cdot \tau_m \cdot p + R} =$$

$$= \frac{0{,}01633 \cdot p}{0{,}00001633 \cdot p^2 + 0{,}01306 \cdot p + 0{,}8} \qquad (4.11)$$

S využitím programu Matlab a příkazu *[z,p,k]=tf2zp(cit,jmen)* byly lehce zjištěny póly a nuly soustavy. Soustava má jednu nulu v počátku a dva póly v bodech –733,17 a –66,83. Z těchto poznatků byl zvolen regulátor typu PI. Regulátor na zadanou soustavu byl navrhován pomocí frekvenčních charakteristik. Základ této metody spočívá v tom, aby ω_F procházela 0 dB při sklonu –20 dB/dek. Proto byla nula regulátoru nastavena do prvního pólů soustavy. Přenos otevřené smyčky byl zobrazen v Sisotoolu a pouze bylo doladěno zesílení regulátoru, s ohledem odezvy na jednotkový skok pro přenos řízení a poruchy. Potřebné frekvenční charakteristiky jsou na obrázku Obr. 4.3, na kterém jsou zobrazeny charakteristiky soustavy, regulátoru a otevřené smyčky.



Obr. 4.3: Frekvenční charakteristiky první soustavy, regulátoru a otevřené smyčky

Zvolen PI regulátor s přenosem:

$$Fr_1 = \frac{85 \cdot (0{,}149 \cdot p + 1)}{p} \qquad (4.12)$$

4.2.3 Návrh druhého regulátoru (regulátor otáček) [3]

Opětovné využití obrázku Obr. 4.2 a spočítání přenosu soustavy (podle blokové algebry), na kterou byl navržen další regulátor, regulátor otáček:

$$Fs_2 = \frac{Fr_1 \cdot Fs_1}{1 + Fr_1 \cdot Fs_1} \cdot \frac{R_{ac}}{C \cdot \tau_m \cdot p} = \frac{\frac{1{,}267 \cdot p + 85}{p} \cdot \frac{p \cdot \tau_m}{R_{ac} \cdot \tau_m \cdot \tau_{ac} \cdot p^2 + R_{ac} \cdot \tau_m \cdot p + R_{ac}}}{1 + \frac{1{,}267 \cdot p + 85}{p} \cdot \frac{p \cdot \tau_m}{R_{ac} \cdot \tau_m \cdot \tau_{ac} \cdot p^2 + R_{ac} \cdot \tau_m \cdot p + R_{ac}}} \cdot$$

$$\cdot \frac{R_{ac}}{C \cdot \tau_m \cdot p} = \frac{\frac{\tau_m \cdot (1{,}267 \cdot p + 85)}{R_{ac} \cdot \tau_m \cdot \tau_{ac} \cdot p^2 + R_{ac} \cdot \tau_m \cdot p + R_{ac}}}{\frac{R \cdot \tau_m \cdot \tau_{ac} \cdot p^2 + R_{ac} \cdot \tau_m \cdot p + R_{ac} + \tau_m \cdot (1{,}267 \cdot p + 85)}{R_{ac} \cdot \tau_m \cdot \tau_{ac} \cdot p^2 + R_{ac} \cdot \tau_m \cdot p + R_{ac}}} \cdot \frac{R_{ac}}{C \cdot \tau_m \cdot p} =$$

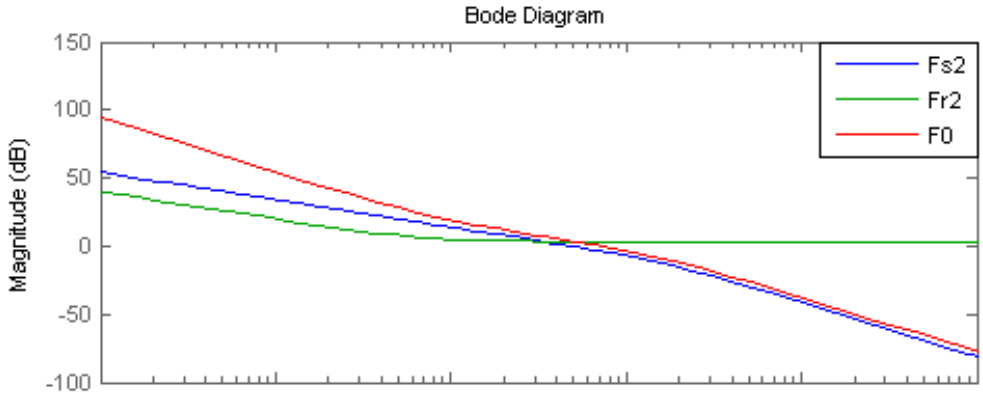
$$= \frac{\tau_m \cdot (1{,}267 \cdot p + 85)}{R_{ac} \cdot \tau_m \cdot \tau_{ac} \cdot p^2 + R_{ac} \cdot \tau_m \cdot p + R_{ac} + \tau_m \cdot (1{,}267 \cdot p + 85)} \cdot \frac{R_{ac}}{C \cdot \tau_m \cdot p} =$$

$$= \frac{\tau_m \cdot (1{,}267 \cdot p + 85)}{R_{ac} \cdot (\tau_m \cdot \tau_{ac} \cdot p^2 + \tau_m \cdot p + 1) + 1{,}267 \cdot p + 85} \cdot \frac{R_{ac}}{C \cdot \tau_m \cdot p} =$$

$$= \frac{(1{,}267 \cdot p + 85)}{C \cdot \tau_m \cdot \tau_{ac} \cdot p^3 + C \cdot p^2 \cdot (\tau_m + 1{,}267 \cdot \tau_m) + C \cdot p \cdot (1 + 85 \cdot \tau_m)} =$$

$$= \frac{1{,}267 \cdot p + 85}{0{,}000001429 \cdot p^3 + 0{,}002591 \cdot p^2 + 0{,}1671 \cdot p} \qquad (4.13)$$

Matlabem byly zjištěny póly a nuly soustavy. Soustava má dvě nuly, v počátku a -67,09, tři póly, v počátku a v bodech –1746,60 a –0,0670. Opět byl zvolen regulátor typu PI a stejný postup při návrhu jako v předchozím případě. Potřebné frekvenční charakteristiky jsou na obrázku Obr. 4.4, na kterém jsou zobrazeny charakteristiky soustavy, regulátoru a otevřené smyčky.



Obr. 4.4: Frekvenční charakteristiky druhé soustavy, regulátoru a otevřené smyčky

Zvolen PI regulátor s přenosem:

$$Fr_2 = \frac{100 \cdot (0{,}0149 \cdot p + 1)}{p} \qquad (4.14)$$

4.2.4 Návrh třetího regulátoru (regulátor polohy) [3]

Další využití obrázku Obr. 4.2 a spočítání přenosu soustavy (podle blokové algebry), na kterou byl navržen poslední regulátor, regulátor polohy:

$$Fs_3 = \frac{Fr_2 \cdot Fs_2}{1 + Fr_2 \cdot Fs_2} \cdot \frac{1}{p} =$$

$$= \frac{\frac{1{,}49 \cdot p + 100}{p} \cdot \frac{(1{,}267 \cdot p + 85)}{C \cdot \tau_m \cdot \tau_{ac} \cdot p^3 + C \cdot p^2 \cdot (\tau_m + 1{,}267 \cdot \tau_m) + C \cdot p \cdot (1 + 85 \cdot \tau_m)}}{1 + \frac{1{,}49 \cdot p + 100}{p} \cdot \frac{(1{,}267 \cdot p + 85)}{C \cdot \tau_m \cdot \tau_{ac} \cdot p^3 + C \cdot p^2 v (\tau_m + 1{,}267 \cdot \tau_m) + C \cdot p \cdot (1 + 85 \cdot \tau_m)}} \cdot \frac{1}{p} =$$

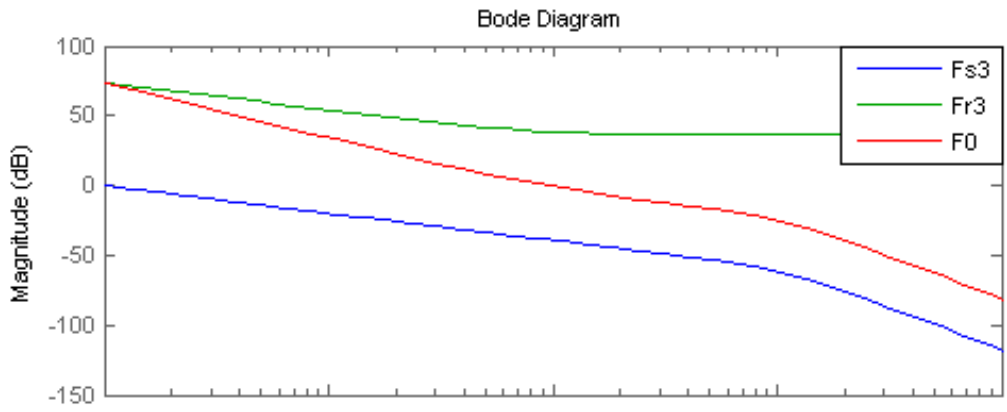
$$= \frac{\frac{1{,}49 \cdot p + 100}{p} \cdot \frac{(1{,}267 \cdot p + 85)}{C \cdot \tau_m \cdot \tau_{ac} \cdot p^3 + C \cdot p^2 \cdot (\tau_m + 1{,}267 \cdot \tau_m) + C \cdot p \cdot (1 + 85 \cdot \tau_m)}}{\frac{C \cdot \tau_m \cdot \tau_{ac} \cdot p^4 + C \cdot p^3 \cdot (\tau_m + 1{,}267 \cdot \tau_m) + C \cdot p^2 \cdot (1 + 85 \cdot \tau_m) + 1{,}888 \cdot p^2 + 253{,}3 \cdot p + 8500}{C \cdot \tau_m \cdot \tau_{ac} \cdot p^4 + C \cdot p^3 \cdot (\tau_m + 1{,}267 \cdot \tau_m) + C \cdot p^2 \cdot (1 + 85 \cdot \tau_m)}} \cdot \frac{1}{p} =$$

$$= \frac{1{,}888 \cdot p^2 + 253{,}3 \cdot p + 8500}{C \cdot \tau_m \cdot \tau_{ac} \cdot p^4 + 2{,}267 \cdot C \cdot \tau_m p^3 + p^2 \cdot (C + 85 \cdot C \cdot \tau_m + C \cdot 1{,}888) + 253{,}3 \cdot p + 8500} \cdot \frac{1}{p} =$$

$$= \frac{1{,}888 \cdot p^2 + 253{,}3 \cdot p + 8500}{C \cdot \tau_m \cdot \tau_{ac} \cdot p^5 + 2{,}267 \cdot C \cdot \tau_m p^4 + p^3 \cdot (C + 85 \cdot C \cdot \tau_m + 1{,}888) + 253{,}3 \cdot p^2 + 8500 \cdot p} =$$

$$= \frac{1{,}888 \cdot p^2 + 253{,}3 \cdot p + 8500}{0{,}000001429\, p^5 + 0{,}002591 \cdot p^4 + 2{,}055 \cdot p^3 + 253{,}3 \cdot p^2 + 8500 \cdot p} \qquad (4.15)$$

Matlabem byly zjištěny póly a nuly soustavy. Soustava má jednu dvojnásobnou nulu v 67,09 pět pólů, v počátku, jeden dvojnásobný v 1094,36 a další v bodech -73,73 a –67,38. Opět byl zvolen regulátor typu PI a stejný postup při návrhu jako v předchozích případech. Potřebné frekvenční charakteristiky jsou na obrázku Obr. 4.5, na kterém jsou zobrazeny charakteristiky soustavy, regulátoru a otevřené smyčky.



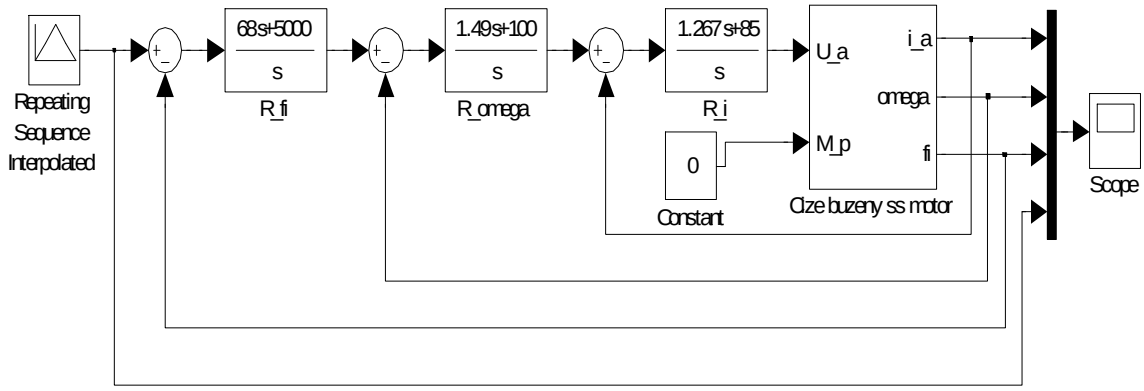
Obr. 4.5: Frekvenční charakteristiky třetí soustavy, regulátoru a otevřené smyčky

Zvolen PI regulátor s přenosem:

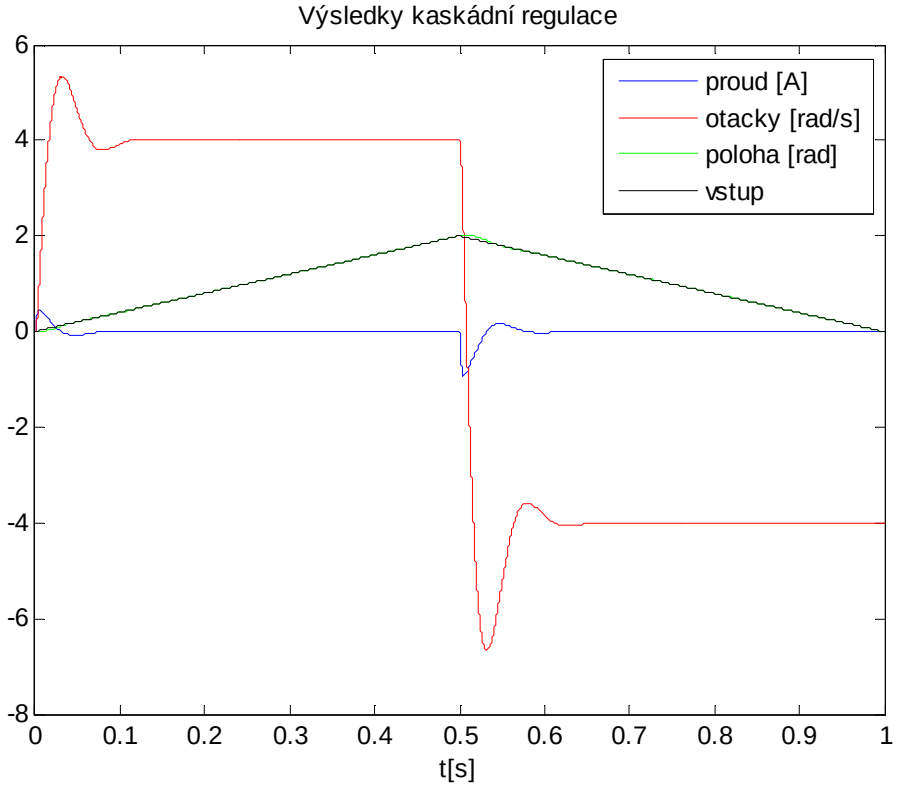
$$Fr_3 = \frac{5000 \cdot (0{,}0136 \cdot p + 1)}{p} \qquad (4.16)$$

5. PRAKTICKÁ UKÁZKA NAVRŽENÝCH REGULÁTORŮ A MATEMATICKÉHO MODELU STEJNOSMĚRNÉHO CIZE BUZENÉHO MOTORU

Do programu Simulink byl překreslen obrázek Obr. 4.2 (pravá část) a následně doplněny přenosy navržených regulátorů. Schéma v Simulinku lze vidět na obrázku Obr. 5.1. Na vstup byl přiřazen navržený trojúhelníkový signál s vrcholovou hodnotou 2 a periodou 1s. Na výstup připojen klasický osciloskop snímající všechny tři regulované veličiny (proud kotvy, otáčky a poloha) a také vzorový průběh, který by měl motor, co nejpřesněji sledovat (viz. Obr. 5.2).



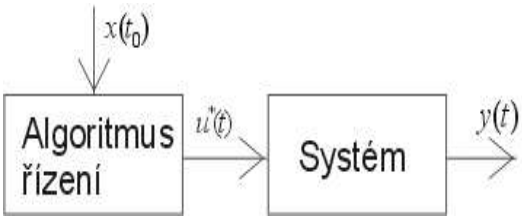
Obr. 5.1: Blokové schéma kaskádní regulace



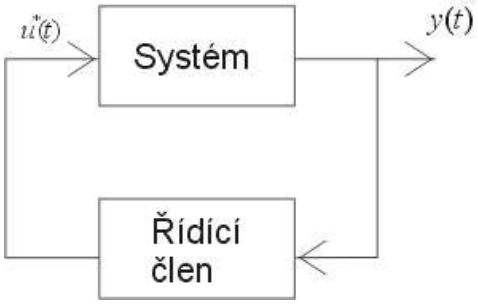
Obr. 5.2: Výsledek kaskádního řízení stejnosměrného cize buzeného motoru

6. DYNAMICKÁ OPTIMALIZACE

K optimálnímu průběhu přechodného děje vedou metody optimalizace parametrů, tedy metody, podle kterých se vypočítá optimální nastavení stávajících analogových, či číslicových řídicích členů. Budeme se zabývat metodami, které povedou na vytvoření struktury k optimálnímu chování podle zvoleného kritéria [4]. Víceméně existují pouze dva základní přístupy k řešení optimalizace struktury systému řízení, a to otevřený obvod (otevřená smyčka), viz. Obr. 6.1. Další možnou variantou je uzavřený obvod (zpětnovazební smyčka), viz. Obr. 6.2.



Obr. 6.1: Otevřený obvod



Obr. 6.2: Uzavřený obvod

První schéma je velice jednoduché, druhé zachovává výhodné vlastnosti se zpětnou vazbou při působení vnějších poruch. Návrh optimálního řídícího systému má tyto fáze [4]:

- stanovení matematického modelu
- určení kritéria řízení
- výpočet (určení řídící struktury algoritmu řízení)

- realizace řídícího systému
- diskuze výsledků

Stanovení matematického modelu je podobné s fází návrhu jakéhokoliv řídícího systému, další kroky mají u návrhu optimálního systému svá specifika. Vzhledem k povaze úlohy se využívá variačních metod pro vyřešení jak otevřené, tak uzavřené smyčky a příslušného řídícího algoritmu. Právě zavedení kriteria řízení je pro danou úlohu nutnou, ovšem ne postačující podmínkou jednoznačnosti řešení. Kromě případu jednoznačného řešení může nastat i případ, kdy řešení není jediné a také případ, kdy žádným řízením nelze dosáhnout požadovaného cíle. Je teda možné, že u dané úlohy není zaručena jednoznačnost ani existence řešení.

6.1 ZÁKLADNÍ ÚLOHA OPTIMÁLNÍHO ŘÍZENÍ

Je dán dynamický systém svými stavovými rovnicemi:

$$\begin{aligned} \dot{x} &= f(x,u,t) \\ y &= f(x,u,t) \end{aligned} \tag{6.1}$$

dále počáteční stav $x(t_0)$, koncový stav $x(t_1)$ a kriterium optimálního řízení

$$I = \Phi(x_1,t_1) + \int_{t_0}^{t_1} L(x,u,t)\,dt \tag{6.2}$$

Úkolem je najít takové optimální řízení $u^*(t)$ nebo $u^*(x)$, které:

- převádí systém z počátečního stavu do koncového stavu
- patří do třídy přípustných řízení
- minimalizuje kriterium optimálního řízení

Abychom mohli tuto úlohu vyřešit a realizovat, potřebujeme [4]:

1) znát rovnice systému
2) znát omezení kladená na řídící vektor
3) znát tvar a hodnoty v kritériu optimálního řízení
4) mít k dispozici matematické prostředky pro vyřešení úlohy

5) rozhodnout se pro otevřenou nebo uzavřenou optimalizační strukturu
6) realizovat toto řešení

Charakter koncového bodu určuje povahu úlohy a proto rozlišujeme [4], [6]:

1) úlohu s pevným koncem a pevným časem
2) úlohu s pevným koncem a volným časem
3) úlohu s volným koncem a pevným časem
4) úlohu s volným časem a cílovou množinou

Hlavním bodem optimálního řízení však stále zůstává volba kriteria. Jsou známy tyto standardní tvary kriterií pro typické úlohy v praxi:

1) Časově optimální systémy užívající kriteria (úloha s volným časem a pevným koncem)

$$I = \int_{t_0}^{t_1} dt \qquad (6.3)$$

2) Systémy s minimální spotřebou energie užívají kriteria

$$I = \int_{t_0}^{t_1} u^T(t)\, R\, u(t)\, dt \qquad (6.4)$$

R (r x r) … pozitivně definitní matice

Fyzikální smysl se velmi dobře ukáže u ss motoru s proudem jako akční veličinou.

3) Systémy s minimální spotřebou paliva mají kriterium obvykle ve tvaru

$$I = \int_{t_0}^{t_1} \sum c;u;(t)\, dt \qquad \text{nebo} \quad I = \int_{t_0}^{t_1} r\left|u(t)\right| dt \qquad (6.5)$$

4) Systémy s minimem čtverců odchylek nebo stavů

$$I = \int_{t_0}^{t_1} e^T(t)\, Q\, e(t)\, dt \qquad I = \int_{t_0}^{t_1} x^T(t) Q\, x(t)\, dt \qquad 6.6)$$

5) Systém optimální dle kvadratického kritéria

$$I = \Phi(x_1, t_1) + \frac{1}{2}\int_{t_0}^{t_1}\left[x^T(t)\,Q\,x(t) + u^T(t)\,R\,u(t)\right]dt \qquad (6.7)$$

Úloha hodnotí nejen minimální kvadratické odchylky, ale usiluje i o minimum energie. Jde o závažnou problematiku, kterou řeší lineární optimální systémy.

Základní úlohu optimálního řízení řeší:

- klasický variační počet Lagrangeův
- variační počet dle Hamiltona
- Pontrjaginův princip maxima a minima
- dynamické programování R. Bellmana
- některé další metody

Zatímco metody klasického variačního počtu jsou určeny pro řešení systémů s neomezeným vektorem řízení a stavu a jejich použití v inženýrských úlohách v důsledku toho je problematické, Pontrjaginův princip má neomezené použití právě v této oblasti. Pro praktickou realizaci je však jeho použití rovněž sporné, neboť vede buď na otevřenou smyčku, která je jako každé ovládání velmi citlivé na poruchy a změnu parametrů, anebo při uzavřené smyčce právě u časově optimálních systémů, pro jejichž řešení má ideální předpoklady, vede na složité nadplochy, které u systémů vyšších řádů je třeba realizovat vlastně jednoúčelovým počítačem. Bellmanovo programování je metoda přísně orientovaná na výpočetní techniku a zdá se, že se nehodí pro řešení řízení dynamických systému pro svoji složitost a zdlouhavost výpočtu. Řešení variační úlohy tedy sice není příliš obtížné po teoretické stránce, ale obtíže nastávají při realizaci řešení úlohy [4].

7. SYSTÉMY OPTIMÁLNÍ DLE KVADRATICKÉHO KRITERIA

Výsledkem řešení optimalizační úlohy byl obecně časový průběh optimálního vektoru řízení $x^*(t)$. Struktura takového optimálního řízení je struktura z obrázku Obr. 6.1. Jak již bylo řečeno tato struktura má značné nevýhody oproti uzavřené regulační smyčce. Je zpravidla možná převést algoritmus $u^*(t)$ na $u^*(x)$, tedy vytvořit uzavřenou strukturu z obrázku Obr. 6.2. Pak ovšem řídící člen s optimálním řídícím algoritmem zanáší do uzavřeného obvodu nelineární operace a původně lineární systém se sává nelineárním se všemi důsledky této struktury. Konečně ve fázi realizace optimálního řízení bylo nutné konstruovat složité jednoúčelové optimalizátory [4].

Pozornost se obrátila k tzv. LQ problému, tj. k systémům optimálním dle kvadratického kritéria. Tyto systémy vedou na lineární řízení a nazývají se proto též lineární optimální systémy.

Syntéza lineárního optimálního regulátoru [4], [6]:

Je dán systém

$$\begin{aligned} \dot{x} &= Ax + Bu \\ y &= Cx \end{aligned}, \qquad (7.1)$$

kde x(t) … stavový vektor (nx1)

u(t) … řídící vektor (rx1)

y(t) … výstupní vektor (mx1)

a kriterium

$$I = \frac{1}{2} x^T(t_1) \cdot Sx(t_1) + \frac{1}{2} \int_{t_0}^{t_1} \left[x^T(t) \cdot Qx(t) + u^T(t) \cdot Ru(t) dt \right] \qquad (7.2)$$

Matice *S*, *Q*, *R* jsou reálné, symetrické a je obvyklé je uvažovat jako diagonální

V čase t_0=0 je systém ve stavu $x(0)$

Čas t_1 je buď pevný a konečný nebo $t_1 \to \infty$

Pokud uvažujeme, že přechodný děj lineárního systému končí teoreticky v čase $t_1 \to \infty$, má úloha tuto povahu:

- systém musí být plně řiditelný
- systém musí být časově invariantní, tj. matice *A*, *B*, *C*, *Q*, *R* nezávisí na čase
- matice $S = 0$
- matice *R* je pozitivně definitní

Úlohu však můžeme stanovit také tak, že přechodný děj je konečný, tj. t_1 je pevný konečný čas. Pak má úloha jiný charakter:

- systém nemusí byt nutně plně řiditelný
- matice *A*, *B*, *C*, *Q*, *R* mohou mít časově proměnné koeficienty
- matice *R* musí být pozitivně definitní
- matice *S* a *Q* jsou pozitivně semidefinitní

Řešení úlohy:

Vzhledem k tomu, že jde o úlohu bez omezení a že jde o variační úlohu, kde funkcionál je závislý na vektorové funkci a navíc rovnice systému tvoří omezující podmínky, je vhodné řešit úlohu Hamiltonovým způsobem.

$$H = \frac{1}{2}(x^T Q x + u^T R u) + p^T \dot{x}$$
$$H(x,u,p) = \frac{1}{2}(x^T Q x + u^T R u) + p^T (Ax + Bu) \qquad (7.3)$$

Nutné podmínky optima jsou ve tvaru:

$$\frac{\partial H}{\partial u} = Ru + B^T p$$

z čehož plyne algoritmus optimálního řízení:

$$u(t) = -R^{-1} B^T p(t) \qquad (7.4)$$

Kovektor stavu p(t) vypočteme jako obvykle z Hamiltonova konjugovaného systému:

$$\dot{p} = -\frac{\partial H}{\partial x}$$
$$\dot{x} = \frac{\partial H}{\partial u}$$

(7.5),(7.6)

$$\dot{p} = -Qx - A^T p$$
$$\dot{x} = Ax + Bu$$

Tuto poslední rovnici lze napsat jako

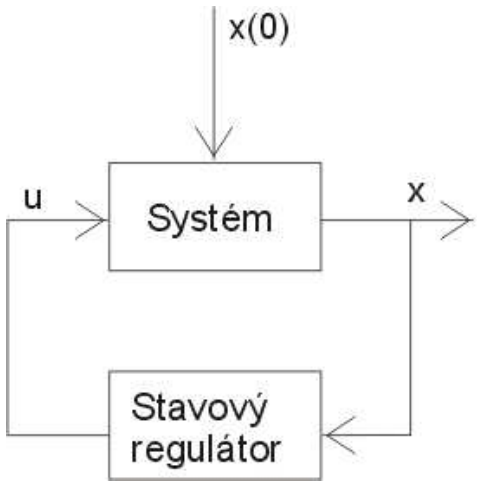
$$\dot{x} = Ax - BR^{-1}B^T p(t) \qquad (7.7)$$

Pro vyřešení Hamiltonova systému rovnic musíme znát strukturu řídícího obvodu. Jak jsme již řekli, nepůjde z pochopitelných důvodů o programování řízení, nýbrž o uzavřenou optimalizační strukturu. Využijeme proto stejné struktury, jako má obvod se stavovým regulátorem, viz. Obr. 7.1. Nyní nám tedy půjde o takové přiřazení, aby platilo *u=u(x)*. Z toho lze zvolit:

$$p(t) = P(t)x(t), \qquad (7.8)$$

kde P(t) je symetrická (nxn) matice. Existence záporné zpětné vazby je zajištěna již znaménkem ve výrazu (7.4). Pro určení Hamiltonova systému, tj. výpočtu *p(t)* derivujeme výraz (7.8) podle času:

$$\dot{p}(t) = \dot{P}(t)x(t) + P(t)\dot{x}(t) \qquad (7.9)$$



Obr. 7.1: Blokové schéma systému se stavovým regulátorem

Po dosazení rovnic (7.1) a (7.9)

$$\dot{p}(t) = \left\{\dot{P}(t) + P(t)A - P(t)BR^{-1}B^T P(t)\right\}x(t) \qquad (7.10)$$

Porovnáním levých a pravých stran výrazu (7.6) a (7.10) dostaneme:

$$(-Q - ATP(t))x(t) = (\dot{P}(t) + P(t)A - P(t)BR^{-1}B^T P(t))x(t) \qquad (7.11)$$

Z toho bezprostředně plyne rovnice:

$$\dot{P}(t) + P(t)A + A^T P(t) - P(t)BR^{-1}B^T P(t) + Q = 0 \qquad (7.12)$$

Tato vektorová nelineární rovnice 1. řádu se nazývá Riccatiho rovnice. Její řešení, které obecně nelze hledat analytickou cestou, je matice *P(t)*, tj. obecně matice s časově proměnnými prvky.

Optimální řízení je pak dáno výrazem:

$$u^*(t) = -R^{-1}B^T P(t)x(t), \qquad (7.13)$$

což je lineární vztah mezi vektorem řízení a vektorem stavu.

Zvláštního významu nabývá tento případ pro t_1 -> ∞, neboť v tom případě platí:

P(t)=konst. … symetrická, reálná, positivně definitní (nxn) matice koeficientů

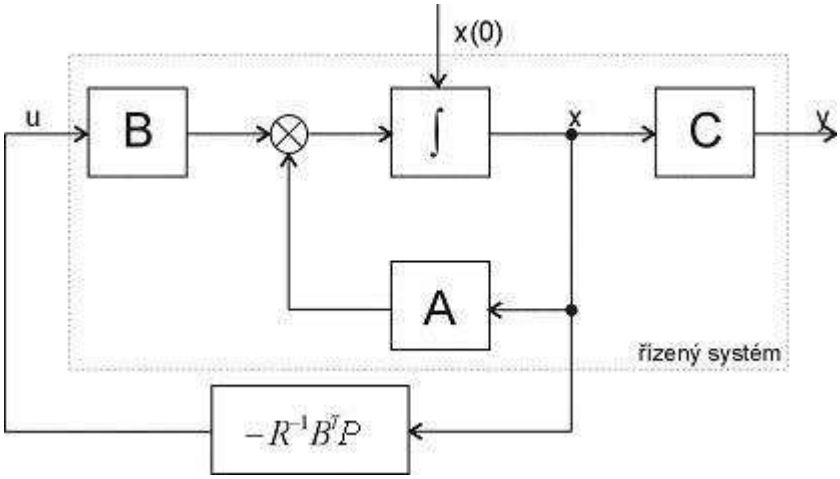
Rovnice (7.12) pak přechází na algebraickou nelineární vektorovou rovnici, jejíž řešení není sice o mnoho jednodušší, ale zato přejde na rovnici:

$$PA + A^T P - PBR^{-1}B^T P + Q = 0, \qquad (7.14)$$

která stanoví, že optimální stavový regulátor je tvořen proporcionálními vazbami obecně od všech stavových veličin, sečtenými na vstupu regulované soustavy. Jde tedy o stavový regulátor, jehož póly charakteristické rovnice nejsou volné, ale pevně určené Riccatiho rovnice. Tedy:

$$u^*(t) = -R^{-1}B^T Px(t) \qquad (7.15)$$

Z tohoto výrazu je dále zřejmé, proč musí být matice *R* positivně definitní. Strukturu celého řídícího systému lze vidět na obrázku Obr. 7.2.



Obr. 7.2: Celé schéma i s LQ regulátorem [4]

8. NÁVRH LQ REGULÁTORU

LG regulátor (L-linear system, Q-quadratic cost) je vlastně stavový regulátor, je potřeba zdůraznit, že stavový regulátor nelze považovat za optimální regulátor. Stavový regulátor by měl mít vlastnosti klasického regulačního zpětnovazebního obvodu [5].

Tvorba modelu motoru byla vybrána pomocí S-funkce, jelikož touto metodou je to o trošku přehlednější simulaci a zároveň je tu možnost lehkého nastavení počátečních podmínek pro odchylku a rychlost. Využíváme stejných rovnic pro výpočet derivace stavových proměnných jako v kapitole 3, pouze výpočet derivace polohy je změněn na výpočet derivace odchylky:

$$\frac{de}{dt} = -\omega \tag{8.1}$$

Tato změna je provedena kvůli tomu, že nebudeme motor řídit z počátku na nějakou žádanou hodnotu, ale přesně obráceně. Tudíž si nenulovou počáteční podmínkou nastavíme, jako odchylku od počátku a pomocí LQ regulátoru se budeme snažit motor řídit, tak aby se dostal do počátku, z čehož plyne, že odchylka bude nulová.

Pro možnost zadávání počátečních nenulových podmínek, stačí pouze u původní deklarace doplnit další dva parametry:

function [sys,x0,str,ts] = cizbuz(t,x,u,flag,Rac,Lac,Rb,Lb,C,J,rych,odch)

V masce motoru pak lze kromě nastavení parametru motoru, nastavit i tyto dvě počáteční podmínky, pro tuto úlohu bylo využito nastavení rychlost rovna nule a odchylka rovna pěti, dále lze nastavit i matice *R* a *Q* (vysvětleny dále v textu). Při správném navržení regulátoru, by tedy měla být hodnota odchylky v čase *t=0* rovna pěti a s narůstajícím časem se blížit k nule.

Samostatný návrh je podle kvadratického kriteria (7.2), kde matice *S* je nulová, matice *Q* je časová a matice *R* je energická. Matice byly zvoleny, podle předchozích řečených pravidel, viz. kapitola 7. Další důležitý vztah je rovnice (7.14) podle které byla vypočítána Riccatiho rovnice.

Matice *A* a *B*, se určí podle schématu Obr. 2.1, kde je ještě na výstup přidán integrátor pro získání polohy φ. Při konstantním buzení lze brát schéma bez stavu budícího proudu. Proto se následně vše trochu zjednoduší a máme opět pouze 3 stavy, z nichž první je proud i_a, druhý stav jsou otáčky ω a třetím stavem je odchylka e (viz. 8.2).

$$\begin{bmatrix} \dot{i}_a \\ \dot{\omega} \\ \dot{e} \end{bmatrix} = \begin{bmatrix} -\frac{R_{ac}}{L_{ac}} & -\frac{C}{L_{ac}} & 0 \\ \frac{C}{J} & 0 & 0 \\ 0 & -1 & 0 \end{bmatrix} \cdot \begin{bmatrix} i_a \\ \omega \\ e \end{bmatrix} + \begin{bmatrix} \frac{1}{L_{ac}} \\ 0 \\ 0 \end{bmatrix} \cdot [u_a] \qquad (8.2)$$

Po dosazení:

$$A = \begin{bmatrix} a_{11} & a_{12} & 0 \\ a_{21} & 0 & 0 \\ 0 & a_{32} & 0 \end{bmatrix}, \quad B = \begin{bmatrix} b_{11} \\ 0 \\ 0 \end{bmatrix}, \quad Q = \begin{bmatrix} q_{11} & 0 & 0 \\ 0 & q_{22} & 0 \\ 0 & 0 & q_{33} \end{bmatrix}, \quad R = [r_{11}]$$

$$P = \begin{bmatrix} p_{11} & p_{12} & p_{13} \\ p_{21} & p_{22} & p_{23} \\ p_{31} & p_{32} & p_{33} \end{bmatrix}$$

Nejdůležitějším bodem k úspěšnému dokončení této práce bylo vyřešní Riccatiho rovnice. První způsob jak vyřešit Riccatiho rovnici byl obecný výpočet prvků matice *P*, aby se tento projekt dal využít pro všechny typy stejnosměrných motorů s cizím buzením. Ovšem jak již bylo řečeno, řešení Riccatiho rovnice není jednoduché. I když bylo využito vlastností matice *P* (kapitola 7) a zjednodušena původní soustava devíti rovnic pro devět neznámých na konečnou soustavu šesti rovnic pro šest neznámých (8.2 - 8.7), nepodařilo se touto volbou Riccatiho rovnici vyřešit. Z těchto rovnic lze jednoznačně vyjádřit pouze parametr p_{13}, který se zároveň rovná parametru p_{31}. Bohužel další parametry nelze obecně vyjádřit.

$$(p_{11}\cdot a_{11}+p_{12}\cdot a_{21})+(p_{11}\cdot a_{11}+p_{12}\cdot a_{12})-\left(\frac{{p_{11}}^2\cdot {b_{11}}^2}{r_{11}}\right)+q_{11}=0$$

$$(p_{11}\cdot a_{12}+p_{13}\cdot a_{32})+(p_{12}\cdot a_{11}+p_{22}\cdot a_{21})-\left(\frac{{p_{11}}^2\cdot {b_{11}}^2}{r_{11}}\right)=0$$

$$(p_{31}\cdot a_{11}+p_{23}\cdot a_{21})-\left(\frac{p_{11}\cdot {b_{11}}^2\cdot p_{13}}{r_{11}}\right)=0$$

$$(p_{12}\cdot a_{12}+p_{22}\cdot a_{32})+(p_{12}\cdot a_{12}+p_{23}\cdot a_{32})-\left(\frac{{p_{12}}^2\cdot {b_{11}}^2}{r_{11}}\right)+q_{12}=0$$

$$(p_{13}\cdot a_{12}+p_{33}\cdot a_{32})-\left(\frac{p_{12}\cdot {b_{11}}^2\cdot p_{13}}{r_{11}}\right)=0$$

$$-\left(\left(\frac{{p_{13}}^2\cdot {b_{11}}^2}{r_{11}}\right)+q_{13}\right)=0$$

(8.2-8.7)

Další možností bylo využít algoritmu pro spojitý výpočet Riccatiho rovnice, takový algoritmus bohužel nebyl nalezen. Jedinou možností pro obecné řešení Riccatiho rovnice byl využit Kleinmanův algoritmus [7]. Matice systému *A* a *B* se teda musí upravit ze spojitýho tvaru do diskrétního, pomocí Eulerovy metody:

$$\dot{x}(t)=A\cdot x(t)+B\cdot u(t)$$
$$x(k+1)=x(k)+T_{vz}\cdot \dot{x}(k+1)$$
$$x(k+1)=x(k)+T_{vz}\cdot A\cdot x(k)+T_{vz}\cdot B\cdot u$$
$$x(k+1)=(I+T_{vz}\cdot A)\cdot x(k)+T_{vz}\cdot B\cdot u$$

T_{vz} … perioda vzorkování

Jelikož byla využita zrovna Eulerova metoda, bylo důležité zvolit periodu vzorkování dostatečně nízkou. Veškeré simulační výsledky byly pro zvolenou periodu vzorkování T_{vz} = 0,0001 s. Pro uživatelovu potřebu je možné periodu změnit v masce motoru.

Nyní již jsou matice v potřebném diskrétním tvaru a lze vypočítat matici pro řešení Riccatiho rovnice *P*. Vychází se ze známých matic *A*, *B*, *R*, *Q*

1. výpočet matice *S*

$$S = B \cdot R^{-1} \cdot B^T$$

2. výpočet matice *W*, kde *n* znamená počet stavů

$$W = \sum_{i=0}^{n-1} A^i \cdot S \cdot (A^T)^i$$

3. počáteční nastavení matice *P*

$$P_0 = (A^T)^n \cdot W^{-1} \cdot A^n$$

4. začátek smyčky pro výpočet matice *P*, ukončovací podmínku lze nastavit na počet určitých iterací nebo na rozdíl dvou po sobě následujících matic *P*

$$A_k = (I + S \cdot P_k)^{-1} \cdot A$$
$$P_{k+1} = A_k^T \cdot P_k \cdot A_k + A_k^T \cdot P_k \cdot S \cdot P_k \cdot A_k + Q$$

Výsledek tohoto algoritmu byl porovnán s příkazem „*dare*“, který také spočítá diskrétní řešení Riccatiho rovnice. Oba výsledky se shodovaly. Jedinou nevýhodou Kleinmanova algoritmu je velká závislost na volbě periody vzorkování. Při volbě periody vzorkování v této práci 0,0001 s cyklus musí proběhnout přibližně 150 000-krát než se dopočítá správného výsledku. Ovšem jak již bylo řečeno výše, perioda vzorkování musela býti volena tak nízko, z důvodu využití Eulerovy metody.

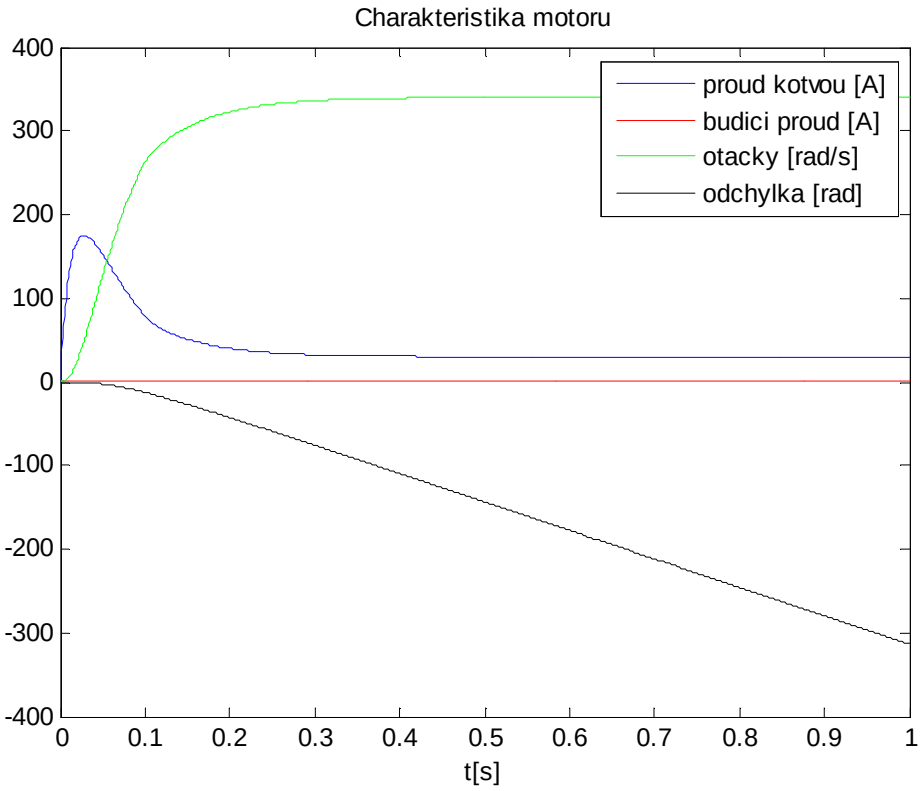
Posledním krokem je výpočet zpětnovazebních koeficientů:

$$K = (R + B^T \cdot P \cdot B)^{-1} \cdot B^T \cdot P \cdot A$$

Výsledkem je vektor tří zpětnovazebních koeficientů, pomocí kterých dosáhneme potřebného řízení odchylky polohy motoru na nulu. Praktické výsledky jsou shrnuty v následující kapitole.

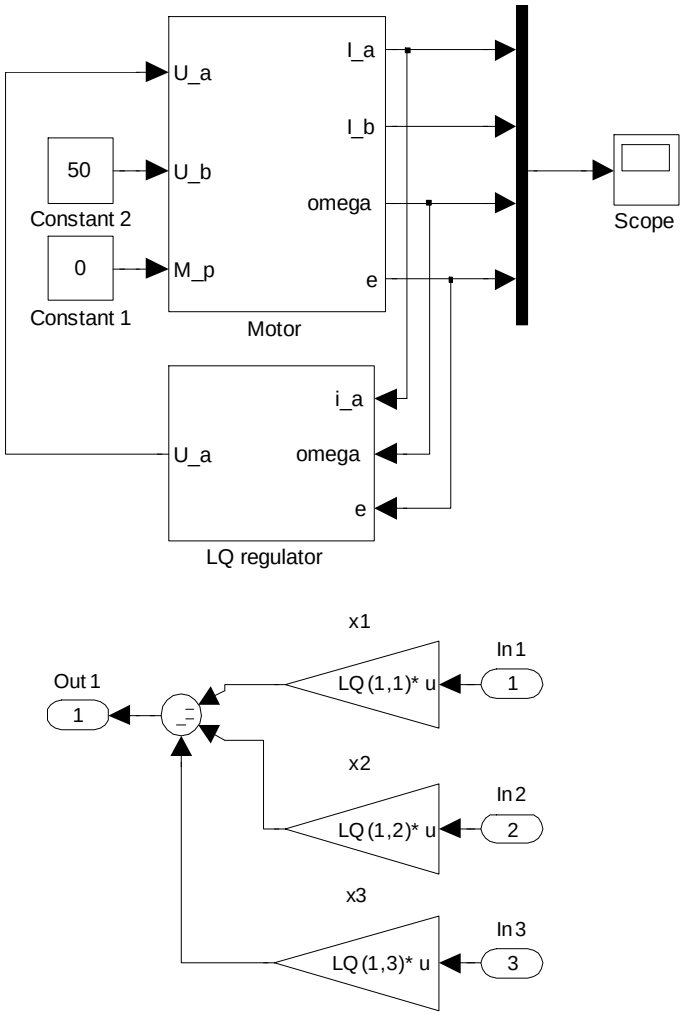
9. VÝSLEDKY NAVRŽENÉHO LQ REGULÁTORU (MATLAB)

V předchozí kapitole byl popsán návrh LQ regulátoru bez jakékoliv pomocné funkce, jako např. *„LQR“* či *„dare“*. V této kapitole budou shrnuty výsledky pro různá nastavení. Ovšem nejdříve na Obr. 9.1 lze vidět základní charakteristiky motoru. Začínáme od počátku, není zde žádné řízení, tudíž se odchylka stále zvětšuje.



Obr. 9.1: Základní charakteristiky motoru

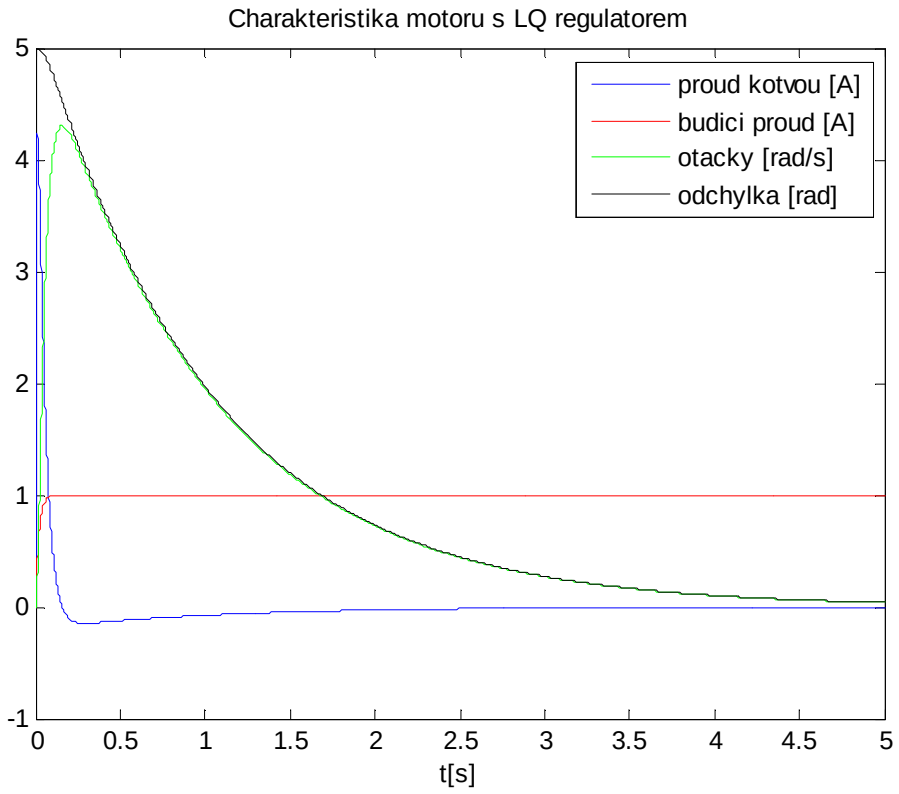
Další charakteristiky již budou se zapojeným LQ regulátorem, podle Obr. 7.2. Využité simulační schéma v programu Simulink, lze vidět na obrázku Obr. 9.2. Úkolem bylo vytvořit energetický regulátor, matice Q zůstane neměnná a matice R budu měněna, abych ukázal vliv působení této matice na celé řízení.



Obr. 9.2: Blokové schéma motoru s LQ regulátorem a LQ regulátoru

Pro první případ platí:

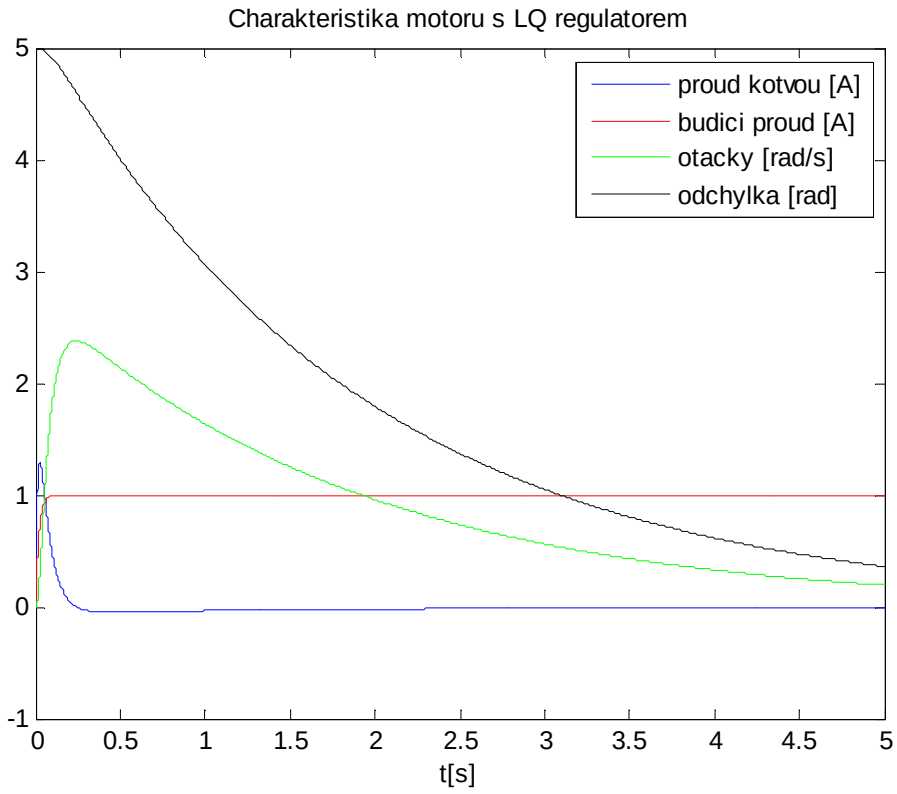
$$A=\begin{bmatrix} 0.99 & -0.005 & 0 \\ 0.25 & 1 & 0 \\ 0 & -0.0001 & 1 \end{bmatrix},\quad B=\begin{bmatrix} 0.01 \\ 0 \\ 0 \end{bmatrix},\quad Q=\begin{bmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 2 \end{bmatrix},\quad R=[0,2]$$



Obr. 9.3: Charakteristika motoru s LQ regulátorem a maticí R = 0,2

Pro druhý případ platí:

$$A=\begin{bmatrix} 0.99 & -0.005 & 0 \\ 0.25 & 1 & 0 \\ 0 & -0.0001 & 1 \end{bmatrix},\quad B=\begin{bmatrix} 0.01 \\ 0 \\ 0 \end{bmatrix},\quad Q=\begin{bmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 2 \end{bmatrix},\quad R=\begin{bmatrix} 20 \end{bmatrix}$$



Obr. 9.4: Charakteristika motoru s LQ regulátorem a maticí R = 20

Z obrázků Obr. 9.3 a Obr 9.4 plyne, že regulátor pracuje správně a odchylku reguluje na nulu. Dále je zde vidět i vliv matice *R* a je pravda, že ovlivňuje energii spotřebovanou při regulaci. V prvním případě proud kotvou překročil hodnotu 4 A a to mělo i za následek, že se odchylka ustálila dříve než v druhém případě, kde byl proud přibližně 2,5 A. Čím větší důraz byl dán na matici *R*, jako v druhém případě, tím bylo spotřebováno méně energie.

10. REKONSTRUKCE STAVU

Rekonstruktory stavu jsou vlastně modely řízeného procesu s vhodnou strukturou. Umožňují fyzicky získat odhady stavových veličin, potřebných pro generování řídícího vektoru. V principu známe rekonstruktory tohoto typu:

- sériové modely
- paralelní modely
- úplný Luenbergerův rekonstruktor
- Luenbergerův rekonstruktor redukovaného řádu

10.1 LUENBERGERŮV ÚPLNÝ REKONSTRUKTOR

V této kapitole je uvedeno zjednodušené matematické odvození úplného rekonstruktoru.

Je dán stacionární a plně pozorovatelný lineární systém n-tého řádu svými stavovými rovnicemi:

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t) \\ y(t) &= Cx(t) \end{aligned} \qquad (10.1)$$

Je žádána struktura, která v ustáleném stavu, ale zčásti i v přechodném ději velmi dobře rekonstruuje časový průběh všech n stavových veličin. K dispozici je řízení *u(t)* a výstup *y(t)*. Sestavíme tedy stavové rovnice rekonstruktoru, které jednak sledují strukturu systému a dále obsahují vliv vstupu a výstupu [4].

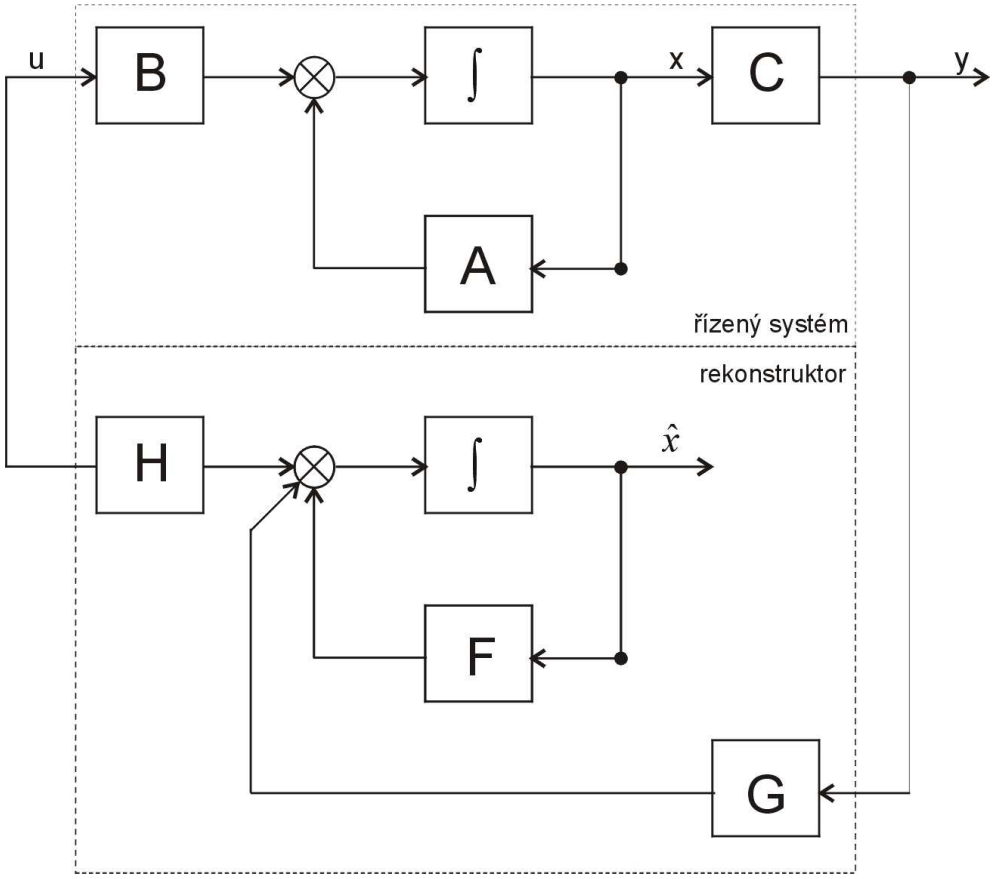
$$\dot{\hat{x}} = F\hat{x} + Hu + Gy$$

$\hat{x}$ … odhad stavu (nx1)

u … řízení (rx1)

y … vektor výstupu (mx1)

Pak celá struktura systému s rekonstruktorem vyjádřená obecnými vektorovými stavovými schématy je ukázána na Obr. 10.1.



Obr. 10.1: Blokové schéma Luenbergerova úplného rekonstruktoru [4]

Matice *F*, *H* a *G* zatím nejsou určeny. Dále požadujeme, aby minimálně v ustáleném stavu platilo:

$$x - \hat{x} = 0 \tag{10.2}$$

Z rovnic lze tedy psát:

$$\begin{aligned} \dot{x} - \dot{\hat{x}} &= Ax + Bu - F\hat{x} - Hu - Gy \\ \dot{x} - \dot{\hat{x}} &= Ax + Bu - F\hat{x} - Hu - GCx \\ \dot{x} - \dot{\hat{x}} &= F(x - \hat{x}) + (A - F - GC)x + (B - H)u \end{aligned} \tag{10.3}$$

Volbou matic F, G, H nyní můžeme zajistit, aby rovnice byla lineární homogenní diferenciální vektorovou rovnicí 1. řádu pro proměnné $(x - \hat{x})$. K tomu splníme podmínky:

$$\begin{aligned} B - H &= 0 \\ A - F - GC &= 0 \end{aligned} \tag{10.4}$$

Poté se rovnice (10.1) upraví na:

$$\hat{x} - \dot{\hat{x}} = F(x - \hat{x}) \tag{10.5}$$

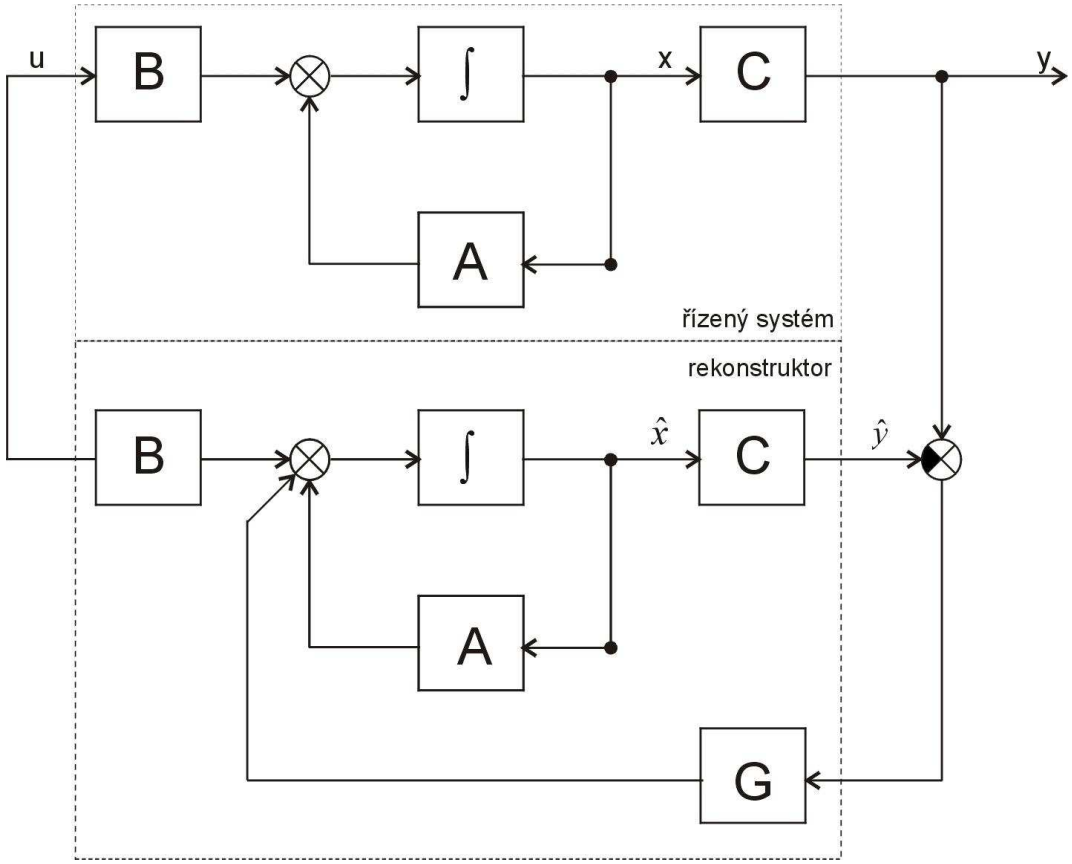
Z podmínek (10.4) vyplývá:

$$\begin{aligned} B &= H \\ F &= A - GC \end{aligned} \tag{10.6}$$

Matice F musí mít záporná vlastní čísla a taky splňovat předchozí podmínku. Tohoto dosáhneme správnou volbou zpětných vazeb v matici G. Rovnice (10.7) je výslednou rovnicí Leunbergerova úplného rekonstruktoru stavu.

$$\dot{\hat{x}} = (A - GC)\hat{x} + Bu + Gy \tag{10.7}$$

Podle ní lze upravit předchozí stavové schéma na nové stavové schéma, viz. Obr. 10.2. Statické vlastnosti rekonstruktoru jsou dány rovnicí (10.5) a dynamické se dají ovlivňovat maticí F, které je ovšem závislá na volbě matice G. Existuje možnost konstruovat, tzv. Luenbergerův rekonstruktor, redukovaného řádu, který rekonstruuje jen chybějící stavové veličiny.

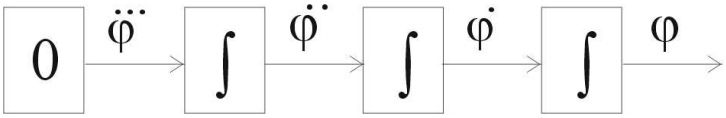


Obr. 10.2: Výsledné blokové schéma Luenbergerova úplného rekonstruktoru [4]

10.2 NÁVRH REKONSTRUKTORU

Skoro vše již bylo řečeno v předchozí kapitole. Schéma bylo zapojeno podle obrázku Obr. 10.2. Jediné, co bude zapotřebí je zvolit vhodně matici G, tak aby se odchylka e ($e = y - \hat{y}$), rovnala nule a rekonstruktor reagoval dostatečně rychle.

Další možnost rekonstrukce otáček byla pomocí jiného modelu motoru. Výstupem motoru byla poloha, tudíž lze její derivací získat rychlost a další derivací zrychlení. Použité schéma pro tento návrh lze vidět na obrázku Obr. 10.3.

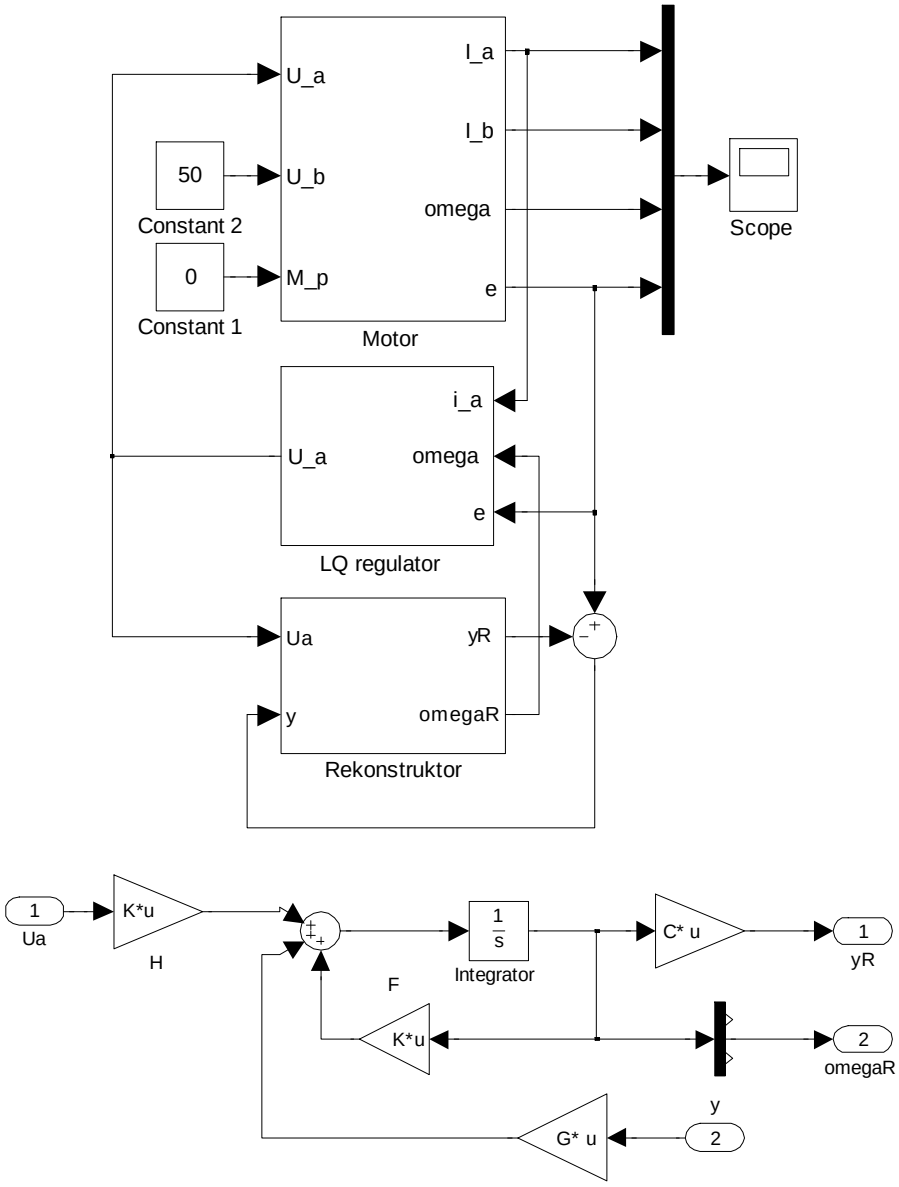


Obr. 10.3: Pomocné náhradní schéma motoru

Princip byl stejný jak u předchozího případu, jediný rozdíl je v použitém modelu, tudíž i matice systémy byly odlišné. Nastavila se matice *G* a poté tohle nastavení bylo aplikováno na skutečný model. Nevýhodou využití tohoto modelu může být nepřesnost derivace polohy. Pokud bude poloha změřena nepřesně a místo potřebné informace o poloze se nám zobrazí zašuměný signál, nastává problém, jelikož ani derivace polohy, tedy rychlost, nebude správná. K vyřešení tohoto problému je třeba využít filtr nebo přímo rekonstruktor. Pokud bude rekonstruktor nastaven, aby co nejrychleji sledoval průběh otáček, může nastat velmi podobný problém se šumem. Při zpomalení rekonstruktoru již tenhle problém nenastane.

11. VÝSLEDKY REKONTRUKTORU

K předchozímu schématu na obrázku Obr. 9.2 byl přidán blok pro rekonstruktor, z kterého byly získány otáčky pro LQ regulátor. Schéma je zobrazeno na obrázku Obr. 11.1.

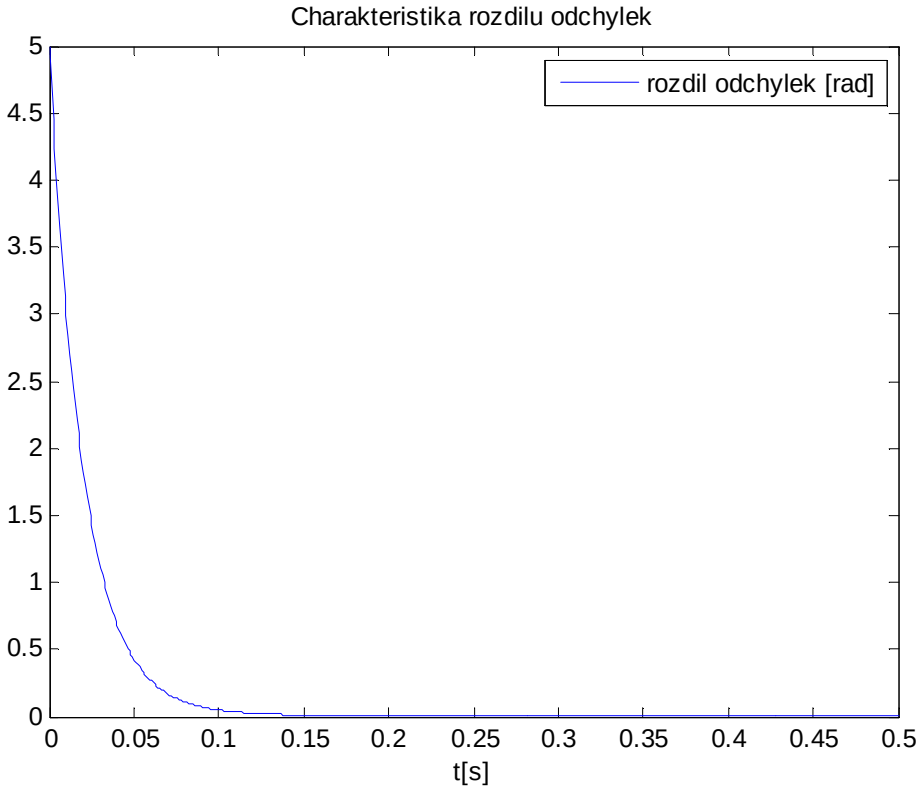


Obr. 11.1: Blokové schéma využívající LQ regulátoru i rekonstruktoru

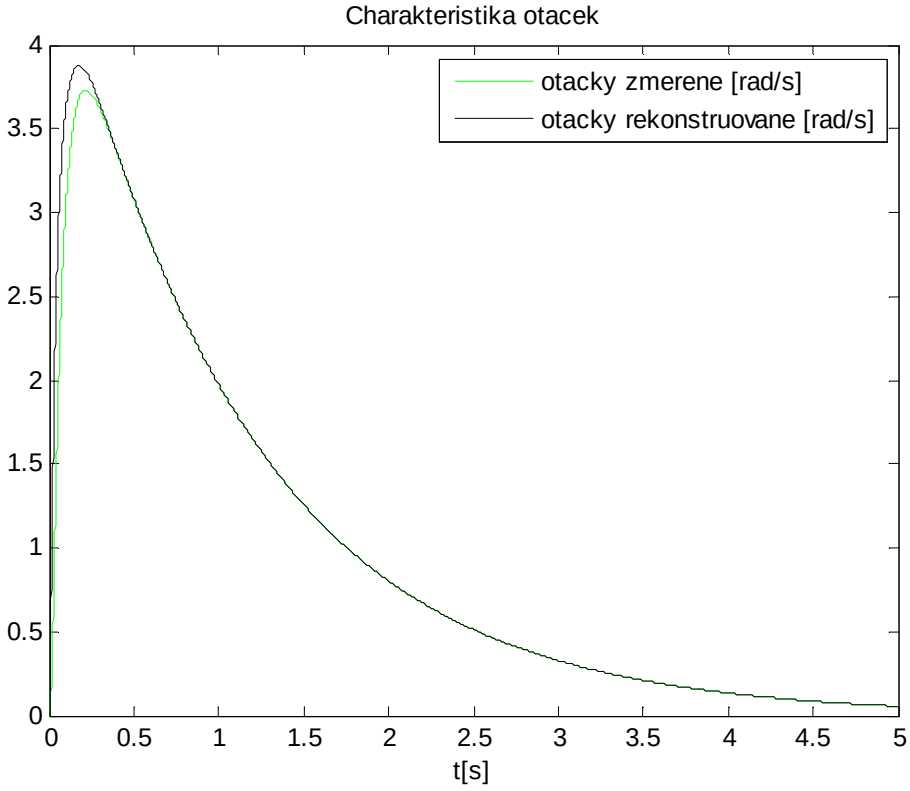
Po pár pokusech nastavení matice *G*, se asi jako nejlepší řešení ukázala tato volba:

$$G = \begin{bmatrix} 1 \\ 5 \\ 50 \end{bmatrix}$$

Při tomto nastavení byl rozdíl odchylek asi po 0,15 s roven nule (Obr. 11.2) a ve stejném časovém okamžiku se rekonstruované otáčky shodují s naměřenými otáčkami, jak lze vidět na Obr. 11.3.



Obr. 11.2: Detail rozdílu odchylek

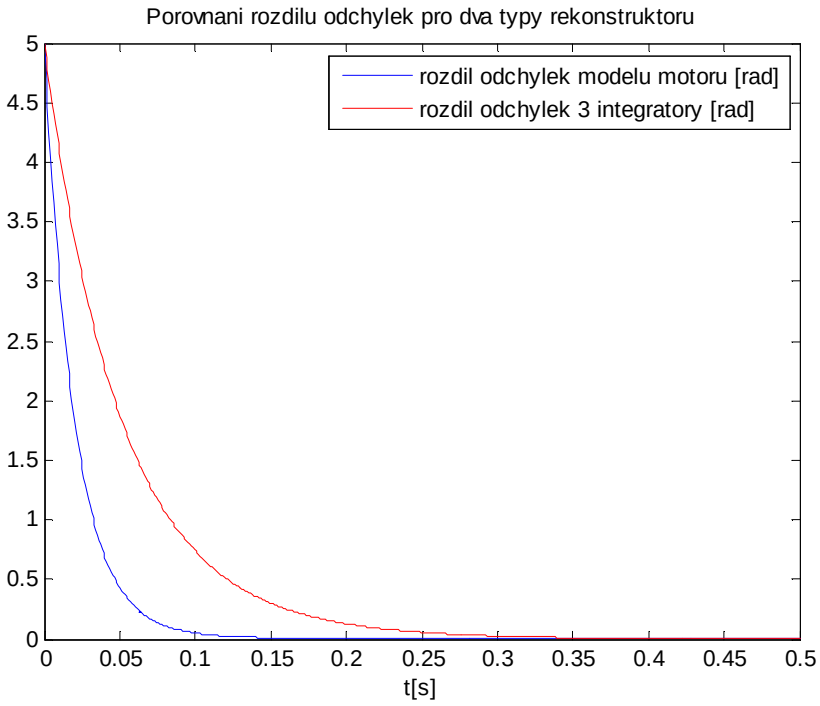


Obr. 11.3: Charakteristika změřených a rekonstruovaných otáček

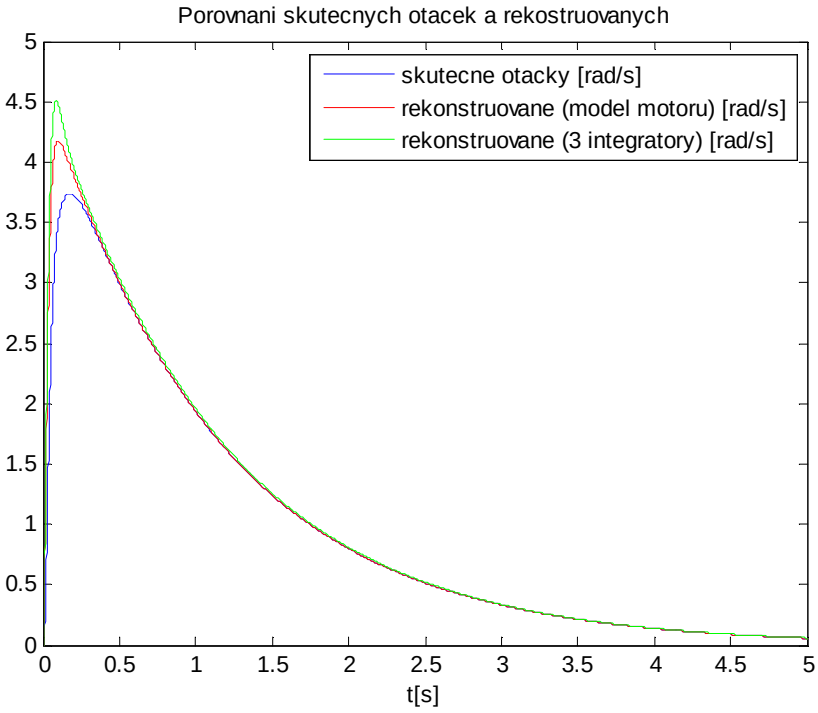
Při návrhu pomocí schématu na obrázku Obr. 11.3 se dosáhlo nastavení matice *G* ve tvaru:

$$G = \begin{bmatrix} 20 \\ 2 \\ 20 \end{bmatrix}$$

Na následujících obrázcích Obr. 11.4 a Obr. 11.5 lze vidět porovnání nastavení těchto dvou rekonstruktorů. Lze vidět, že přesnější návrh byl pomocí klasického matematického modelu motoru. Pomocí matematického modelu motoru byly rekonstruované otáčky podobnější skutečným oproti využití modelu tří integrátorů.

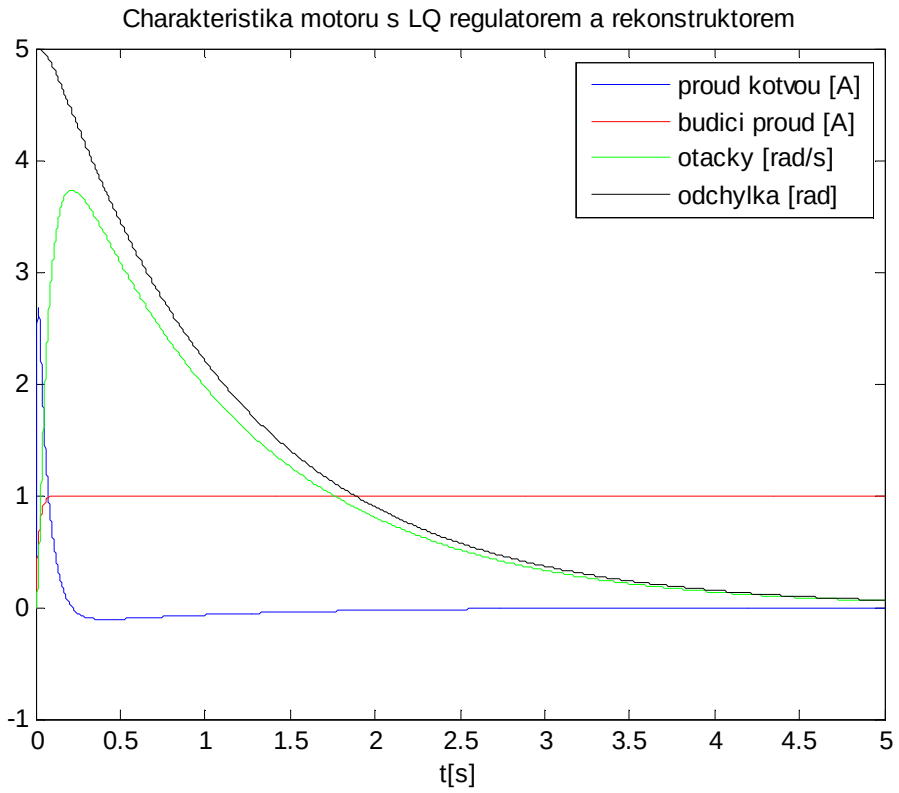


Obr. 11.4: Porovnání rozdílu odchylek dvou rekonstruktorů



Obr. 11.5: Srovnání skutečných a rekonstruovaných otáček dvou rekonstruktorů

Na závěr této kapitoly ukázka charakteristiky motoru s využitím LQ regulátoru i rekonstruktoru otáček zobrazená na obrázku Obr. 11.6. Bylo dosaženo podobného průběhu jako v případě měřitelného výstupu otáček, viz. Obr. 9.3 a Obr. 9.4. Úplně stejného průběhu nemohlo být dosaženo, jelikož simulace probíhala s nastavením matice $R = 2$.



Obr. 11.6: Charakteristika motoru s LQ regulátorem a rekonstruovanými otáčkami

12. APLIKACE LQ REGULÁTORU NA SIGNÁLOVÉM PROCESORU

V předchozích kapitolách byl k simulaci praktického návrhu využit pouze program Matlab-Simulink, v této kapitole bude popsán návrh vytvoření algoritmu pro signálový procesor, kterým bude řízen opět model motoru. Byl využit signálový procesor firmy Motorola, z rodiny Freescale 56F8300. Model motoru byl použit stejný jako v předchozích kapitolách, tedy vytvořen pomocí s-funkce a v prostředí Simulinku. Komunikace mezi těmito programy byla pomocí zapůjčené knihovny SFIO od vedoucího práce. Byl použit i jeden m-file, kterým byl naplněn hlavičkový soubor s parametry motoru, maximálními hodnotami proudu, otáček a odchylky a na závěr i hodnoty nastavující kvadratické kriterium, přesněji řečeno matice R a Q. Vše ostatní bylo řešeno již pouze procesorem, který vypočítal zpětnovazební konstanty a následné řízení motoru.

12.1 TVORBA PROGRAMU V CODEWARRIORU

Prvním úkolem byla implementace algoritmu pro výpočet Riccatiho rovnice a následný výpočet zpětnovazebních koeficientů, bez využití rekonstruktoru.

12.1.1 Riccatiho rovnice

Pro navržení LQ regulátoru je potřeba vyřešit Riccatiho rovnici. Původně bylo zamýšleno řešit jak spojitou tak i diskrétní Riccatiho rovnici, ale jelikož v kapitole 8 nebylo dosaženo postupu spojitého řešení ani další jiný spojitý algoritmus nebyl nalezen, je potřeba se spokojit s diskrétním postupem řešení. Pro tohle řešení byl použit již zmíněný Kleinmanův algoritmus.

V programovém prostředí Matlabu bylo vytvoření tohoto algoritmu celkem lehké, ovšem složitější už je v programovacím jazyce C, kde se musí vytvořit funkce pro násobení matic a jejich inverzi a další pomocné matice (S, W, P_0, A_k).

V procesoru je využita tzv. „fractional aritmetika“, což znamená, že veškeré vstupní, výstupní, spočítané hodnoty by měly býti v rozmezí intervalu <-1;1>. Proto

jsou potřeba různé normalizace, jednak vstupních, výstupních veličin a následně i mezivýpočtů. Normalizaci vstupních veličin docílíme pomocí nastavených maximálních hodnot.

Kleinmanův algoritmus je silně závislý na periodě vzorkování. Zároveň jde o náročný algoritmu, který by pro vytvoření ve fractional aritmetice byl složitý a časově náročný. Využila se tedy možnost knihovny pro počítání s floatovými typy.

12.1.2 Výpočet zpětnovazebních konstant

Z řešení Riccatiho rovnice již byly lehce spočítány potřebné zpětnovazební koeficienty. Jelikož se s těmito koeficienty bude počítat již ve fractional aritmetice, musí být v intervalu <-1;1>. Tato podmínka byla splněna pomocí následujících pár kroků. Ze všech třech prvků byl určen maximální prvek, dále byla vzata jeho absolutní hodnota, která byla dělena dvěma tolikrát, aby tato hodnota byla menší jak jedna. Je nutné si zapamatovat počet dělení.

12.1.3 Výpočet napětí kotvy

Posledním krokem k úspěšnému vyřešení scházelo vypočítat napětí kotvy motoru. Toto napětí bylo počítáno pomocí rovnice (12.3).

$$u_a = K_1 \cdot i_a + K_2 \cdot \omega + K_3 \cdot e$$
$$u_{an} \cdot u_{a\max} = K_1 \cdot i_{an} \cdot i_{a\max} + K_2 \cdot \omega_n \cdot \omega_{\max} + K_3 \cdot e_n \cdot e_{\max} \qquad (12.1\text{-}12.3)$$
$$u_{an} = K_1 \cdot i_{an} \cdot \frac{i_{a\max}}{u_{a\max}} + K_2 \cdot \omega_n \cdot \frac{\omega_{\max}}{u_{a\max}} + K_3 \cdot e_n \cdot \frac{e_{\max}}{u_{a\max}}$$

K_1, K_2, K_3 … zpětnovazební koeficienty

u_{an}, i_{an}, ω_n, e_n … normované veličiny

u_{amax}, i_{amax}, ω_{max}, e_{max} … maximální možné hodnoty veličin

Tímto vzorcem je zaručeno, že výsledná hodnota napětí kotvy nepřekročí hodnotu jedna a splní podmínku pro počítání ve fractional aritmetice.

12.1.4 Popis programu v CodeWarrioru

V programu byly vytvořeny 4 globální proměnné. Tři byly využity pro zpětnovazební koeficienty a poslední pro zapamatování počtu dělení největšího z nich a následné „shiftování“ výsledku.

Byla vytvořena funkce „riccati“, která vypočítá řešení Riccatiho rovnice, zpětnovazební koeficienty a počet kolikrát se dělilo dvěma. Současně zde byly vynásobeny zpětnovazební prvky s podílem maximálních příslušných hodnot. Tato část byla počítána pouze jednou, jelikož je během celého cyklu řízení stejná.

Scházelo vypočítat pouze výsledné napětí kotvy. Rovnice (12.3) se tudíž pozmění na rovnici (12.4). Tento výpočet byl zahrnut do funkce „vypocet“.

$$u_{an} = K_1 \cdot i_{an} + K_2 \cdot \omega_n + K_3 \cdot e_n \qquad (12.4)$$

$$K_1 \ldots K_1 \cdot \frac{i_{a\max}}{u_{a\max}}, \qquad K_2 \ldots K_2 \cdot \frac{\omega_{\max}}{u_{a\max}}, \qquad K_3 \ldots K_3 \cdot \frac{e_{\max}}{u_{a\max}}$$

12.1.5 Postup simulace

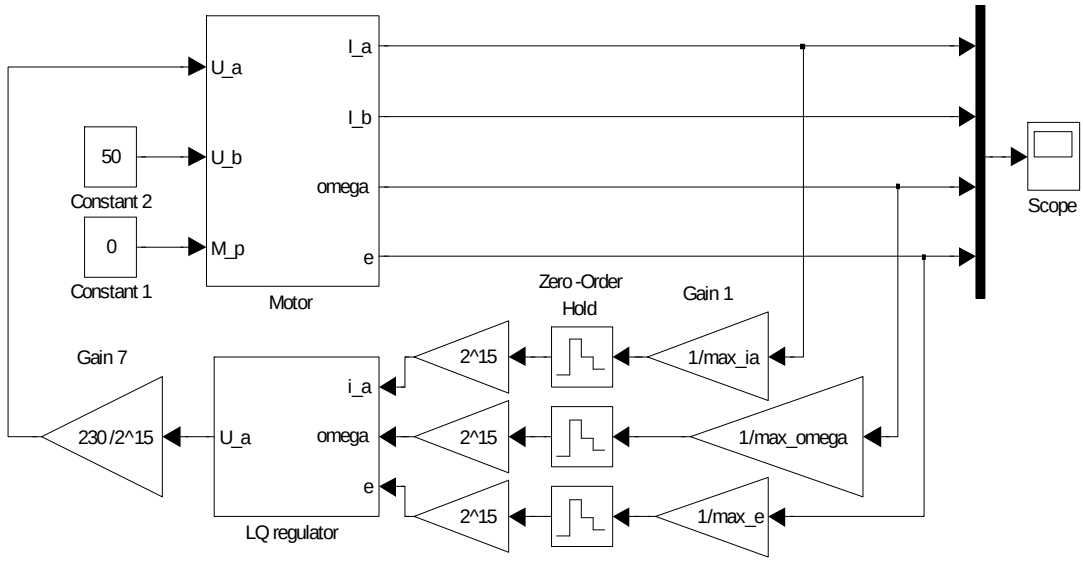
Ke správné simulaci jsou důležité tři soubory. Začalo se nastavením všech potřebných hodnot v modelovém schématu v Simulinku. V masce motoru se nastaví parametry motoru, počáteční podmínky rychlosti a odchylky, hodnoty matic R a Q, perioda vzorkování T_{vz}, maximální hodnoty proudu i_a, otáček ω a odchylky e. Je potřeba, aby byl zaškrtnut checkbox pro tvorbu hlavičkového souboru hodnot.

V dalším kroku se spustil m-file, který načetl všechny zadané hodnoty a vytvořil hlavičkový soubor důležitý pro program v CodeWarrioru.

Posledním krokem bylo spustit samotný program v CodeWarrioru a poté simulaci modelového schématu.

13. VÝSLEDKY NAVRŽENÉHO LQ REGULÁTORU (CODEWARRIOR)

K simulaci bylo využito níže uvedené schéma, které je zobrazeno v klasickém prostředí Simulink, pro správnou funkčnost je potřeba transformace bloku LQ regulátoru.



Obr. 13.1: Blokové schéma motoru s LQ regulátorem (CodeWarior)

V kapitole 9 byly ukázány výsledky pro různá nastavení matice *R*, v této kapitole bude ukázáno pouze jedno nastavení matice *R*.

Ostatní parametry:

R_{ac} = 1 Ω

L_{ac} = 1 mH

R_b = 50 Ω

L_b = 1 H

J = 0,02 kg·m^2

C = 0,5

T_{vz} = *0,0001 s*

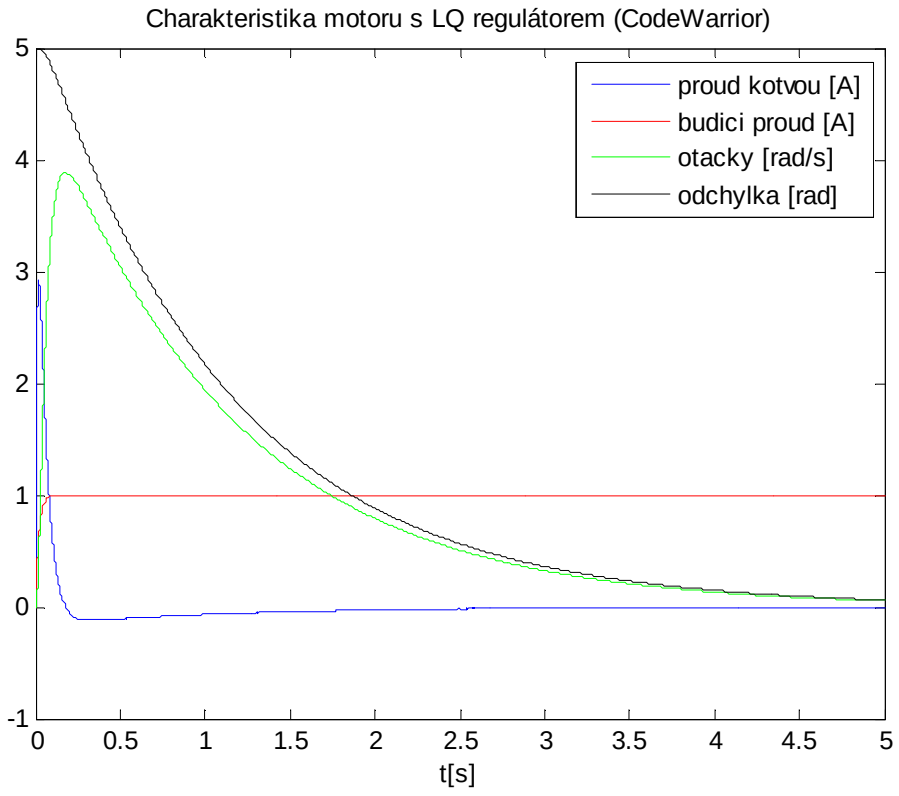
$$A=\begin{bmatrix} 0.99 & -0.005 & 0 \\ 0.25 & 1 & 0 \\ 0 & -0.0001 & 1 \end{bmatrix},\quad B=\begin{bmatrix} 0.01 \\ 0 \\ 0 \end{bmatrix},\quad Q=\begin{bmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 2 \end{bmatrix},\quad R=[2]$$

u_{amax} = 230 V

i_{amax} = 10 A

ω_{max} = 200 rad/s

e_{max} = 20 rad



Obr. 13.1: Charakteristika motoru s LQ regulátorem vypočítáním v programovém prostředí CodeWarior

Z obrázku Obr. 13.1 lze vidět, že průběh charakteristik odpovídá teoretickému předpokladu, tudíž se dá předpokládat, že se implementace algoritmu pro výpočet LQ regulátoru do programového prostředí CodeWarrior podařila.

14. APLIKACE REKONSTRUKTORU NA SIGNÁLOVÉM PROCESORU

V této kapitole se pojednává o rozšíření simulačního schématu a programu pro návrh LQ regulátoru. Motor byl opět simulován pomocí programového prostředí Matlab-Simulink a procesor vypočítával všechny parametry potřebné pro LQ regulátor a nyní i pro rekonstruktor.

14.1 TVORBA PROGRAMU V CODEWARRIORU

K původnímu programu byly vytvořeny nové dvě funkce, pro výpočet potřebných koeficientů a funkce pro výpočet samotného rekonstruktoru.

14.1.1 Výpočet koeficientů a rekonstruktoru

Vycházelo se z rovnic matematického modelu motoru (3.9), (3.11) a dále rovnice pro výpočet derivace odchylky (8.1) a z blokového schématu Luenbergerova rekonstruktoru na obrázku Obr. 10.2. Výsledné rovnice vypadaly následovně:

$$\frac{d\,i_a}{dt} = \frac{u_a - R_{ac}\cdot i_a - C\cdot\omega}{L_{ac}} + G_1\cdot(e-\hat{e}) \qquad (14.1)$$

$$\frac{d\,\omega}{dt} = \frac{C\,i_a - m_p}{J} + G_2\cdot(e-\hat{e}) \qquad (14.2)$$

$$\frac{de}{dt} = -\omega + G_3\cdot(e-\hat{e}) \qquad (14.3)$$

Tyto rovníce byly přepsány do diferenčního a přehlednějšího tvaru pro následnou normalizaci:

$$i_{an}\cdot i_{a\max} = \left(\frac{u_{an}\cdot u_{a\max} - R_{ac}\cdot i_{an}\cdot i_{a\max} - C\cdot\omega_n\cdot\omega_{\max}}{L_{ac}} + G_1\cdot y_n\cdot y_{\max}\right)\cdot T_{vz}$$

(14.4)

$$\omega_n \cdot \omega_{\max} = \left(\frac{C \cdot i_{an} \cdot i_{a\max}}{J} + G_2 \cdot y_n \cdot y_{\max}\right) \cdot T_{vz} \quad (14.5)$$

$$e_n \cdot e_{\max} = \left(-\omega_n \cdot \omega_{\max} + G_3 \cdot y_n \cdot y_{\max}\right) \cdot T_{vz} \quad (14.6)$$

Konečný normalizovaný tvar rovnic, využitých k implementaci rekonstruktoru do programu CodeWarior:

$$i_{an} = u_{an} \frac{u_{a\max} \cdot T_{vz}}{L_{ac} \cdot i_{a\max}} - i_{an} \frac{R_{ac} \cdot T_{vz}}{L_{ac}} - \omega_n \frac{C \cdot \omega_{\max} \cdot T_{vz}}{L_{ac} \cdot i_{a\max}} + y_n \frac{G_1 \cdot y_{\max}}{i_{a\max}} \quad (14.7)$$

$$\omega_n = i_{an} \cdot \frac{C \cdot i_{a\max} \cdot T_{vz}}{J \cdot \omega_{\max}} + y_n \cdot \frac{G_2 \cdot y_{\max} \cdot T_{vz}}{\omega_{\max}} \quad (14.8)$$

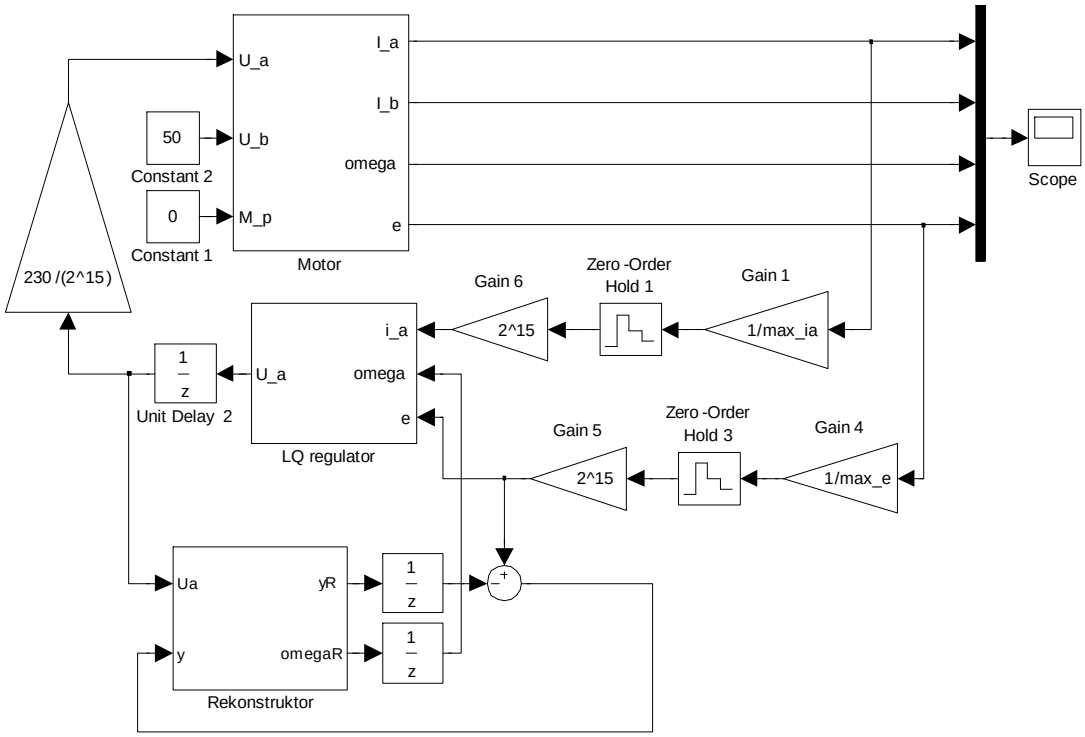
$$e_n = -\omega_n \cdot \frac{\omega_{\max} \cdot T_{vz}}{e_{\max}} + y_n \cdot \frac{G_3 \cdot y_{\max} \cdot T_{vz}}{e_{\max}} \quad (14.9)$$

Každý zlomek v těchto posledních třech rovnicích byl nahrazen koeficientem pro menší náročnost výpočtu, jelikož se tyto koeficienty během celého výpočtu nemění. Byly tedy počítány v inicializačním kroku podobně jako pro výpočet Riccatiho rovnice.

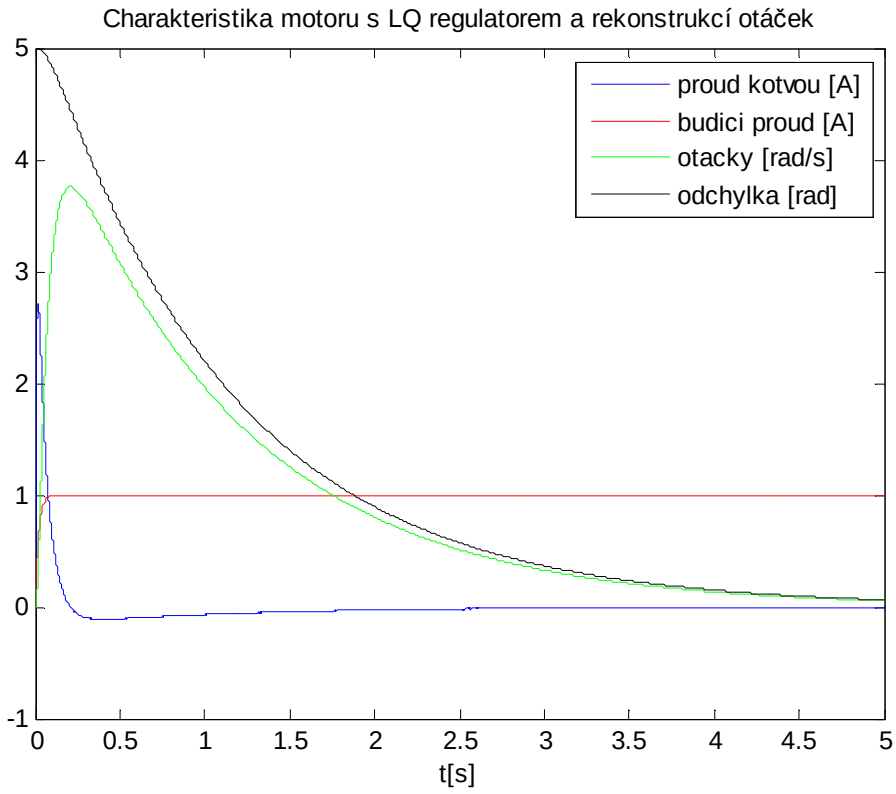
Těmito rovnicemi se spočítaly diference, které byly přičítány k příslušným stavovým proměnným proudu, otáček a odchylky.

15. VÝSLEDKY NAVRŽENÉHO REKONSTRUKTORU (CODEWARIOR)

K simulaci bylo využito níže uvedené schéma, které je zobrazeno v klasickém prostředí Simulink, pro správnou funkčnost je potřeba transformace bloku LQ regulátoru i rekonstruktoru.



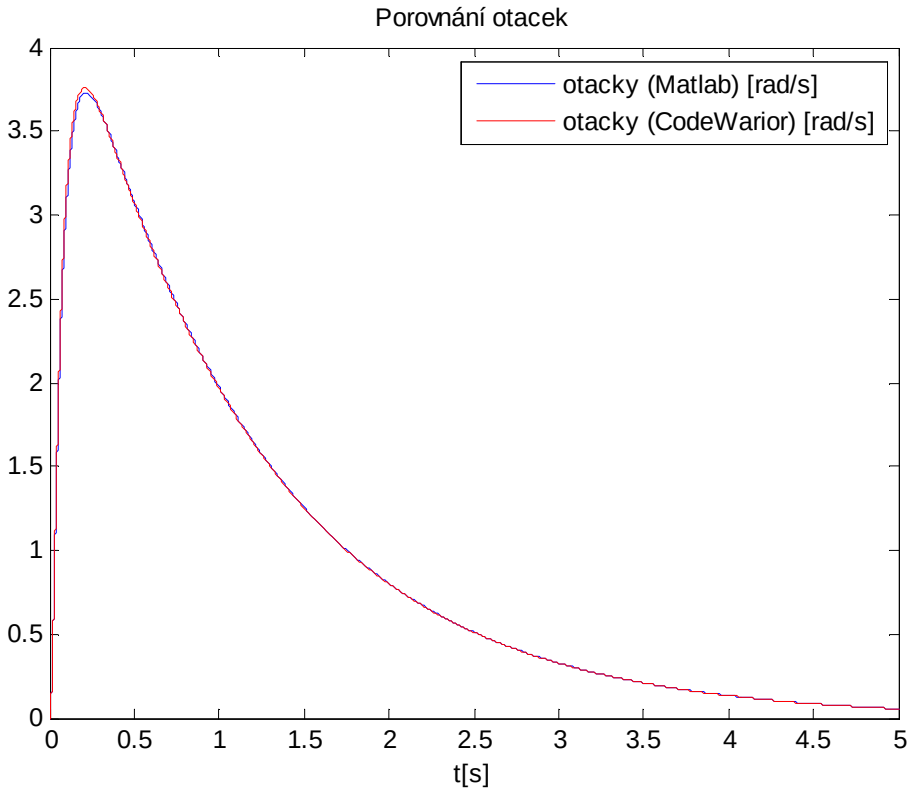
Obr. 15.1: Blokové schéma LQ regulátoru s rekonstruktorem (CodeWarior)



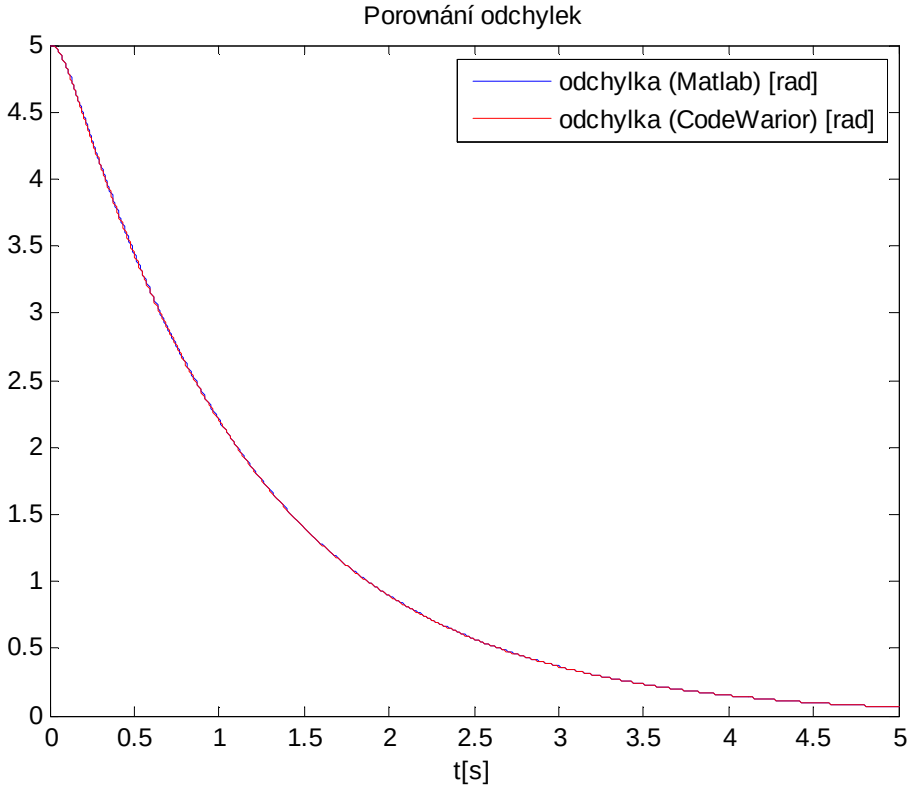
Obr. 15.2: Charakteristika motoru s LQ regulátorem a rekonstruktorem otáček naprogramovaných pomocí CodeWarioru

16. POROVNÁNÍ VÝSLEDKŮ MATLABU A CODEWARIORU

Na následujících dvou obrázcích (Obr. 16.1 a Obr. 16.2) lze vidět porovnání otáček a odchylek pro modely počítány v programu Matlab-Simulink a CodeWarior. Podle těchto obrázků lze usoudit, že se implementace algoritmu pro výpočet Riccatiho rovnice, LQ regulátoru i rekonstruktoru povedly.



Obr. 16.1: Porovnání otáček algoritmu vytvořených v programovém prostředí Matlab a CodeWarior



Obr. 16.2: Porovnání odchylek algoritmu vytvořených v programovém prostředí Matlab a CodeWarior

17. ZÁVĚR

Tato diplomová práce obsahuje teoretický rozbor matematického modelu, postupy pro tvorbu modelů k simulaci a návrh kaskádního řízení stejnosměrného cize buzeného motoru s využitím spojitých regulátorů. Dále teorii optimálního řízení, návrh LQ regulátoru a rekonstruktoru. LQ regulátor a rekonstruktor byly řešeny jak v prostředí Matlab-Simulink, tak v CodeWarioru.

Nejdříve byla potřeba navrhnout si model stejnosměrného cize buzeného motoru. Podle obvodových schémat byly vytvořeny rovnice pro elektrickou část a mechanickou část motoru a následně i bloková schémata. Využití Simulinku a tvorba modelu pomocí blokového schématu byla jednoduchá. Další možnost byla pomocí vytvoření S-funkce předdefinovaného vzoru. Při modelování se nevyskytl žádný výrazný problém a vše proběhlo bez komplikací.

Dále bylo úkolem seznámit se s problematikou řízení stejnosměrného cize buzeného motoru. Regulátory byly navrženy spojitě. Zde se již pár komplikací vyskytlo. Prvním byla překřížená vazba (viz. Obr. 4.2), která se sice dala odstranit, ale oproti původnímu předpokladu, ztížila výpočet přenosů určitých soustav. S využitím programu Matlab byly přenosy dosti nepřehledné a zbytečně vysokého řádu, proto jsem všechny přenosy počítal ručně. S možností krácení během výpočtu se snížily řády přenosů a už vypadaly přijatelněji. Bylo zapotřebí odstoupit od původního záměru navrhování, využívající opakování metody Ziegler-Nicholsova a přejít na metodu tvaru frekvenčních charakteristik otevřeného obvodu.

Z praktické ukázky na obrázku Obr. 5.2 lze soudit, že vytvoření modelu a navržení regulátorů pro stejnosměrný cize buzený motor se podařilo. Poloha motoru velmi přesně kopíruje vstupní průběh.

V následující části práce se zabývalo dalším způsobem polohového řízení stejnosměrného cize buzeného motoru. Jednalo se o kvadratické optimální řízení, jehož základem je kvadratické kriterium. Úkolem bylo navrhnout regulátor s ohledem na spotřebu energie při regulaci. Z obrázků Obr. 9.3 a Obr. 9.4 je vidět, že se tento návrh povedl a pomocí matice R si můžeme nastavit jak moc nám bude ovlivňovat spotřebovanou energii, v tomto případě se jedná o kotevní proud i_a. Při

tomhle návrhu se vyskytl asi nejzávažnější problém celé práce. Jednalo se o řešení Riccatiho rovnice. Jak již bylo zmíněno, zamýšlelo se řešení jak spojité tak diskrétní a pro různé parametry motorů. Spojitého řešení se nepodařilo dosáhnout a bohužel jiný algoritmus nebyl nalezen. Tudíž byla Riccatiho rovnice řešena pomocí diskrétního Kleinmanova algoritmu. Funkčnost tohoto algoritmu byla ověřena příkazem *„dare“*. Na obrázku Obr. 13.1 lze vidět průběh regulace spočítané v prostředí CodeWarioru a na obrázku Obr. 16.2 porovnání průběhu odchylek regulovaných v prostředí Matlab-Simulink a CodeWarior, průběhy se shodují.

Jednalo se o servomotor, kde vlastně výstupem je poloha φ (nebo spíše odchylka polohy e). Dalšími potřebnými veličinami jsou proud kotvou i_a, otáčky ω. Nabízí se možnost počítat otáčky přímo z derivace polohy, což ovšem při více zašuměném signálu může způsobit nepřesnost výpočtu. Proto byla zvolena možnost rekonstrukce otáček, k čemuž slouží např. Luenbergerův rekonstruktor otáček. Na obrázku Obr. 11.3, lze vidět porovnání skutečných a rekonstruovaných otáček. A na obrázku Obr. 16.1 porovnání rekonstrukce otáček pomocí Matlabu i CodeWarioru, oba průběhy se shodovaly. Návrh rekonstruktoru nebyl nějak zvláště obtížný, jednalo se spíše jen o správnou volbu matice G, aby rekonstruované otáčky, co nejlépe kopírovaly průběh naměřených otáček. Pokud je tohle splněno, lze jako vstup do regulátoru místo naměřených otáček použít právě tyhle vypočítané a na regulaci to nebude mít žádný vliv.

Posledním úkolem bylo implementovat algoritmy z Matlabu do signálového procesoru od Firmy Motorola. Největším rozdílem oproti práci v Matlabu byla tzv. *„fractional aritmetika“*, kdy bylo zapotřebí normalizovat veškeré veličiny do intervalu <-1;1>. Řešení Riccatiho rovnice je dosti náročné a normovat veškeré veličiny a výpočty by bylo dosti časově náročné, proto se využilo knihovny počítající s normální aritmetikou. Pomocí ní bylo vypočítáno řešení Riccatiho rovnice, což ovšem prodloužilo dobu výpočtu. Další výpočty byly realizovány ve zmíněné fractional aritmetice. K veškerým výpočtům byly využity, tzv. intrisické funkce.

Všechny výsledky modelování byly znázorněny v grafech a následně porovnávány podle teoretických předpokladů i mezi sebou. V poslední kapitole lze vidět, že se výsledky obou algoritmů shodují.

Závěrem bych ještě jednou rád poděkoval vedoucímu diplomové práce panu doc. Ing. Pavlu Václavkovi, Ph.D. za zapůjčení hardwarového i softwarového vybavení, odborné připomínky a cenné rady, které mi při zpracování pomohly a přispěly k její obsahové kvalitě.

18. SEZNAM LITERATURY

[1] Caha, Z., Černý, M.: *Elektrické pohony*, Ediční středisko ČVUT, Praha 1987

[2] Zezulka, F.: *Prostředky průmyslové automatizace*, skripta VUT, verze 4.11.2004

[3] Blaha, P., Vavřín, P.: *Řízení a regulace I*, skripta VUT, verze 7.11.2005

[4] Zezulka, F: *Teorie automatického řízení II*, SNTL – Nakladatelství technické literatury, Praha 1987

[5] Pivoňka, P.: *Optimalizace regulátorů*, skripta VUT, verze 7.12.2005

[6] Havlena, V., Štecha, J.: *Moderní teorie řízení*, Ediční středisko ČVUT, Praha 1999

[7] Kleinman, D.: *Stabilizing a Discrete, Constant, Linear System with Application to Iterative Methods for Solving the Riccati Equation*. IEEE Transactions on Automatic Control, 1974

[8] Freescale: *CodeWarrior™Development Studio for Freescale™ 56800/E*, 19 June 2006

[9] Herout, P.: Učebnice jazyka C, Kopp České Budějovice, 2005

PŘÍLOHA 1

OBSAH CD

Součástí diplomové práce je i kompaktní disk, na kterém se nachází samotná práce v elektronické podobě a veškeré programy a simulační schémata.

Složka „Kaskadni regulace“ obsahuje m-file (kaskadni_regulace.m) a model (kask_reg.mdl). Stačí spustit m-file, program se spustí, odsimuluje a zobrazí výsledky.

Složka „LQ regulator M“ obsahuje s-funkci (cizbuzLQ.m), m-file (Scizbuz_LQregulator.m) a model (Scizbuz_LQ.mdl. Stačí spustit m-file.

Složka „Rekonstruktor M“ obsahuje s-funkci (cizbuzLQ.m), m-file (Scizbuz_LQregulator_rekonstruktor.m) a model (Scizbuz_LQ_rek.mdl. Stačí spustit m-file.

Složka „LQ regulator CW“ obsahuje s-funkci (cizbuzLQ.m), m-file (Scizbuz_LQregulator_CW.m) a model (Scizbuz_LQ_CW.mdl) a další složku „CW“, kde se nachází příslušný projekt v CodeWarioru. Potřeba nastavit parametry motoru a další potřebné parametry k výpočtu, spustit m-file, projekt v CodeWarioru a jako poslední simulační schéma. Samotné nefunguje, potřeba dohrát knihovny.

Složka „Rekonstruktor CW“ obsahuje s-funkci (cizbuzLQ.m), m-file (Scizbuz_LQregulator_rekonstruktor_CW.m) a model (Scizbuz_LQ_rek_CW.mdl) a další složku „CW“, kde se nachází příslušný projekt v CodeWarioru. Potřeba nastavit parametry motoru a další potřebné parametry k výpočtu, spustit m-file, projekt v CodeWarioru a jako poslední simulační schéma. Samotné nefunguje, potřeba dohrát knihovny.

Složka „Porovnani M a CW“ obsahuje s-funkci (cizbuzLQ.m), m-file (Scizbuz_LQregulator_rekonstruktor_MxCW.m) a model (Scizbuz_LQ_rek_MxCW.mdl) a další složku „CW“, kde se nachází příslušný projekt v CodeWarioru. Potřeba nastavit parametry motoru a další potřebné parametry k výpočtu, spustit m-file, projekt v CodeWarioru a jako poslední simulační schéma. Samotné nefunguje, potřeba dohrát knihovny.