České vysoké učení technické v Praze
Fakulta elektrotechnická

DIPLOMOVÁ PRÁCE

Implementace adaptivního řízení pohonu řezacího stroje

Praha, 2007 **Lukáš Synovec**

Katedra řídicí techniky ***Školní rok:*** 2004/2005

ZADÁNÍ DIPLOMOVÉ PRÁCE

Student: Lukáš Synovec

Obor: Technická kybernetika

Název tématu: Implementace adaptivního řízení pohonu řezacího stroje

Zásady pro vypracování:

1. Nastudujte dokumentaci k řídicímu systému.
2. Navrhněte řízení pohonu stroje.
3. Implementujte a odzkoušejte navržené řízení.

Seznam odborné literatury: Dodá vedoucí práce.

Vedoucí diplomové práce: Ing. Přemysl Šůcha

Termín zadání diplomové práce: letní semestr 2004/2005

Termín odevzdání diplomové práce: květen 2006

prof. Ing. Michael Šebek, DrSc.
vedoucí katedry

L.S.

v z.z.n.

prof. Ing. Vladimír Kučera, DrSc.
děkan

V Praze dne 28.06.2005

Poděkování

Úvodem mé práce bych rád poděkoval zejména ing. Miloslavu Žižkovi, který mi pomohl nejen mnoha radami a podněty, ale také poskytl potřebné zázemí pro realizaci experimentů a nezbytnou pomoc při implementaci algoritmů pro DSP procesor. Také děkuji doc. Janu Johnovi, neboť poznatky získané při konzultaci s ním mi byly užitečné během celé práce.

Můj největší dík patří školiteli ing. Přemyslu Šůchovi, který mi byl, kdykoliv bylo třeba, vždy ochotně připraven pomoci. Jsem mu vděčný nejen za příkladné vedení, ale i za trpělivý a lidský přístup po celou dobu vzniku práce.

V neposlední řadě patří můj dík rodičům a mé ženě Janě, bez jejichž podopory a trpělivosti bych si nemohl představit ani dokončení této práce, ani předcházející studium.

Abstrakt

Obsahem této práce je návrh algoritmu, který rozšíří řídící systém CNC řezacího stroje pro tvarové dělení ocelových plechů plazmou nebo kyslíkem o automatické seřízení PSD regulátorů servopohonů. Součástí práce je také implementace navrženého algoritmu v jazyce C pro řídící systém stroje s operačním systémem Linux. Použitá metoda automatického seřízení regulátoru je založena na frekvenční metodě syntézy, která zaručuje zvolenou fázovou bezpečnost přenosu otevřené smyčky a tím požadované vlastnosti přenosu uzavřené smyčky. Syntéza regulátoru vychází z přenosu systému získaného identifikací metodou nejmenších čtverců pomocí ARX modelu. Úspešnost implementované metody je demonstrována simulacemi i experimenty na řídícím systému s reálným servopohonem.

Abstract

Auto-tuning PSD controller has been proposed and applied to the servo controller of CNC-controlled machines for oxygen shape-cutting of sheet metal and for cutting with the aid of the plasma technologies, in this thesis. The algorithm implementation in the C programming language for Linux system, which is used for driving the machine, has been included in this work too. The proposed method results from frequency synthesis and achieves prescribed phase and gain margin and thus guarantees quality of the closed loop transfer functions. Controller synthesis is based on plant transfer function obtained using least square method model identification using ARX model. The effectiveness of the proposed method is shown through simulations and experiments on the control system with real servo unit.

Obsah

Seznam obrázků

Seznam tabulek

Kapitola 1

Úvod

CNC *(Computer Numerical Control)* řízené stroje pro tvarové řezání plechů se používají k výrobě polotovarů, nazývaných výpalky, vyřezáváním z plechů podle CAD technických výkresů z plechů až do tloušťky 250mm. Pro dělení materiálu je využívano řezání kyslíkem, nebo pro menší tloušťky materiálu plazmové technologie. Řezací stroje jsou nejčastěji portálové konstrukce, kdy je pohyb hořáku zajišťován servopohonem v příčné a podélné souřadnicové ose. Obrázek stroje této konstrukce, který je v sortimentu firmy Vanad 2000 a.s. `<http://www.vanad.com/CZ>` je na obrázku 1.1.

Obrázek 1.1: Řezací stroj portálové kostrukce

Na kvalitu výpalku jsou kladeny vzhledem k technologii řezání mimořádně vysoké nároky, zejména co se týče rozměrové přesnosti, kolmosti hran a struktury povrchu řezu.

Vzhledem k vysoké ceně materiálu a případného dalšího obrábění je snaha o maximální využití této provozně levné technologie přirozená. Tento trend ale klade vysoké nároky na přesnost řízení pohybu hořáku. Posuv musí být dostatečně jemný, přesný a měl by současně zajišťovat po celou dobu konstantní rychlost pohybu bez kmitání. Je zřejmé, že tyto podmínky jsou fyzikálně nesplnitelné už pro jakýkoliv výpalek s vrcholem na obrysu, kde dochází ke skokové změně rychlosti pohybu. Příklad ne příliš dobrého povrchu řezu výpalku z důvodu nerovnoměrné rychlosti pohybu nebo špatné regulace výšky hořáku vidíme na obrázku 1.2.

Z důvodu požadavku jemnosti a kostantní rychlosti nelze používat pro pohon stroje krokových motorů. Používají se stejnosměrné servopohony, které spolu s inkrementálními snímači pracují v uzavřené servosmyčce. Vznikají tím ale vysoké nároky na kvalitu regulace servopohonů neboť, jak je vidět třeba na uvedeném příkladu vrcholů na obrysu, má dynamika uzavřené regulační smyčky servopohonu zásadní vliv na výsledný tvar a povrch řezu.

Obrázek 1.2: Fotografie povrchu řezu výpalku

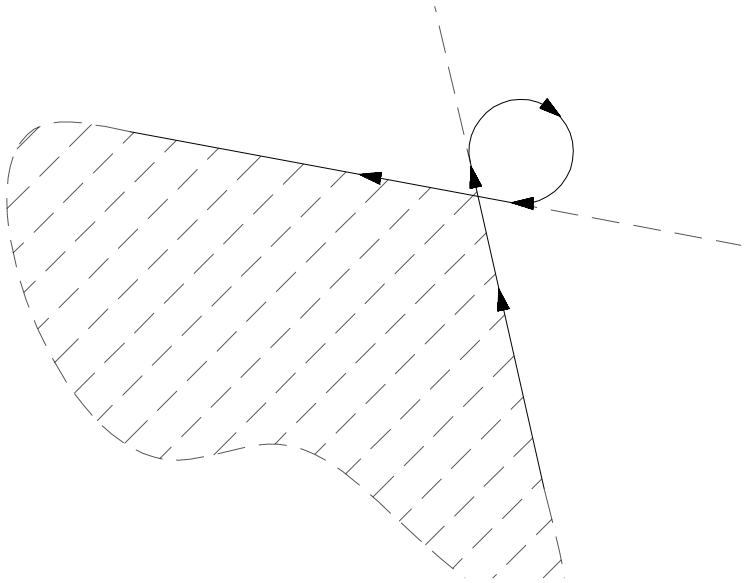
Původně byl řídící systém, na kterém jsme testovali nový adaptivní algoritmus, vybaven P regulátorem, ručně nastaveným technikem při uvádění stroje do provozu. Další úpravy nastavení po dobu životnosti mechanických čátí stroje nebyly prováděny. Zesílení

regulátoru bylo často nastavováno pro jistotu menší, aby se předešlo případným problémům se stabilitou uzavřené smyčky. To je samozřejmě na úkor dynamiky, takže například zmiňovaný problém vrcholů je často řešen úpravou tvaru obrysu za cenu vyšší spotřeby materiálu, jak je vidět na obrázku 1.3. Přestože by bylo pro přesnost regulace výhodné využití PID regulátoru, ukázalo se, že ruční nastavení PID regulátoru servopohonů je v praxi nerealizovatelné.

Hlavním cílem této práce tedy je zajistit automatické nastavení diskrétního regulátoru PID, v literatuře často označovaného PSD *(proporcionálně sumačně diferenční)*, při zachování stávajícího řídícího systému stroje.

I když přináší zachování klasické PID regulace omezení ve výběru adaptivních algoritmů, byla k jejímu zachování důvodem zejména distribuovaná struktura a výpočetní výkon řídícího systému a také ten fakt, že technikům zůstane možnost ručního nastavení metodami, s kterými mají zkušenosti. Nemůžeme ale například použít lineárně kvadratického (LQ) přístupu k syntéze řízení, který předpokládá stavovou zpětnou vazbu, přestože je například v (Neuhauser, 2004) tato metoda v podobném případě použita. O této práci se ještě podrobněji zmíníme spolu s dalšími na konci úvodu.

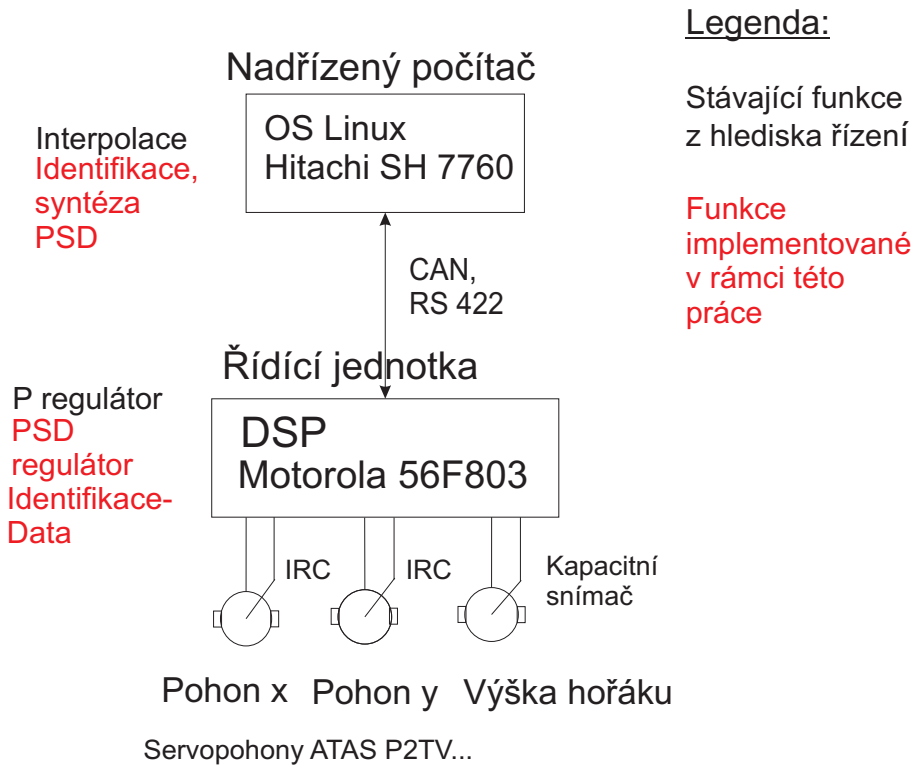
V dále se budeme věnovat právě struktuře řídícího systému, protože je důležitým faktorem pro volbu algoritmu jak vlastního regulátoru tak i algoritmu vyšší úrovně, který bude zajišťovat nastavení a úpravu jeho parametrů, případně diagnostiku stroje. Dále zmíníme také zajímavé práce zabývající se využitím adaptivních algoritmů.



Obrázek 1.3: Způsob vyřezávání vrcholu při nevyhovující dynamice servopohonu

1.1 Řídící systém CNC řezacího stroje

Na obrázku 1.4 je schematicky znázorněná struktura řídícího systému stroje. Z obrázku je zřejmé, že činnost stroje zajišťují dva procesory. Řízení servopohonů v obou osách a servopohonu zdvihu hořáku zajišťuje řídící jednotka se signálovým procesorem Motorola DSP56F803, který generuje napětí pro servopohony pomocí *pulsně šířkových modulátorů* (PWM). Koncové stupně PWM jsou umístěny na desce společně s DSP *(Digital Signal Processing)* procesorem. Ten je rovněž vybaven obvody pro zpracování signálu z inkrementálního snímače umístěného na hřídeli motoru. Z hlediska řízení by měl zajišťovat DSP procesor vykonávání časově kritických operací. V navrhovaném řešení je to jednak výpočet akčního zásahu PSD regulátoru, jednak získání dat pro identifikaci v otevřené smyčce. Výpočet akčního zásahu PSD regulátoru i sběr dat pro identifikaci by měla provádět řídící jednotka s konstantní vzorkovací periodou Ts. Žádné další složitější operace není možné v DSP procesoru řídící jednotky vykonávat.



Obrázek 1.4: Uspořádání řídícího systému řezacího stroje

Zbývající úlohy řízení, kterými jsou identifikace modelu servomechanismu a syntéza regulátoru, musí tedy být implementovány v nadřízeném počítači. Vzhledem k tomu, že nadřízený počítač není vybaven real-time operačním systémem, musí probíhat identifi-

kace a syntéza regulátoru bez nároků na zpracování v reálném čase. V současné době je nadřízený počítač tvořen takzvaným *vestavěným systémem* v anglické terminologii nazývaným *embedded system* tedy jednodeskovým počítačem určeným pro specielní účely, v našem případě pro řízení stroje. Je vybaven prosesorem Hitachi SH7760 a pro tyto účely upraveným operačním systémem Linux, rovněž někdy označovaným anglickým termínem Embedded Linux.

Nadřízený počítač vykonává v současné době zejména interpolaci. Při té dochází ke zpracování zadaných výkresů definujících obrys výpalku do sekvencí kroků rozložených pro pohony v jednotlivých osách. Cílem řízení pohonů je tedy co nejvěrněji sledovat polohu generovanou interpolátorem.

Na obrázku 1.4 jsou heslovitě uvedeny funkce důležité z hlediska řízení vykonávané jednotlivými procesory. Černou barvou jsou vypsány stávající funkce a červenou barvou ty, které jsou implementovány v rámci této práce.

Na reálném stroji řídící jednotka s DSP procesorem řídí více servopohonů, což komplikuje komunikaci. Protože cílem této práce není detailní popis fungování řídícího systému pálicího stroje, omezíme se při popisu komunikace mezi nadřízeným počítačem a řídící jednotkou (viz kapitola 3) pouze na jeden servopohon. Důvodem je také to, že pro usnadnění ladění software byl místo průmyslového počítače s procesorem SH7760 použit osobní počítač (PC) rovněž s operačním systémem Linux. Komunikaci, která probíhá po sběrnici CAN nebo RS422, tak nahradil sériový port RS232. Z hlediska implementace a ladění řízení tato změna nemá vliv.

1.2 Adaptivní systémy-rozdělení

V této kapitole uvedeme použité řešení do kontextu poměrně široké problematiky adaptivního řízení. Pokusíme se definovat pojem adaptivního řízení a stručnou klasifikaci adaptivních řídících systémů (AS). Vyjdeme z podrobného přehledu a terminologie uvedené v monografii (Bobál et al., 1999). V této monografii je několik definic adaptivního systému. Citujme například tuto obecnou definici:

Adaptivní systém měří určité ukazatele chování daného nastavitelného systému pomocí jeho vstupů, stavů nebo výstupů. Na základě porovnání těchto měření ukazatelů a množiny požadovaných ukazatelů, modifikuje parametry nebo strukturu nastavitelného obvodu nebo generuje pomocný vstup, tak aby se požadované ukazatele chování udržovaly na hodnotách

co nejbližších žádaným.

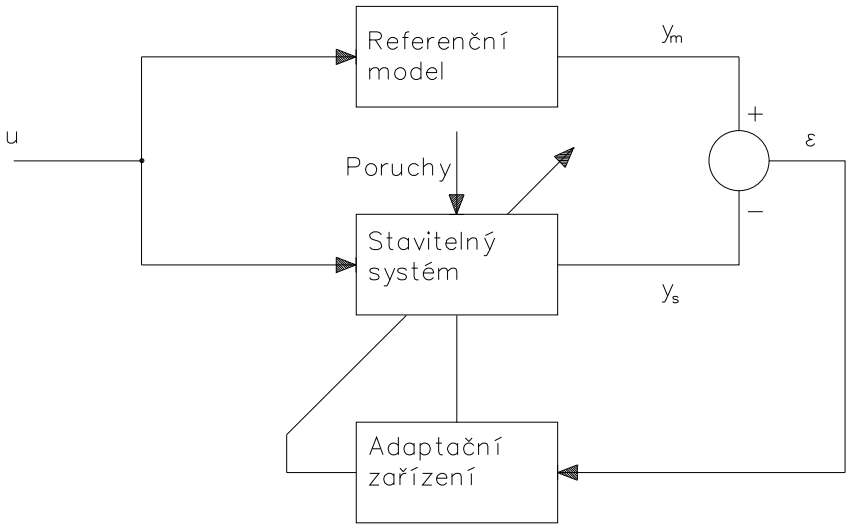
Podle způsobu, jak definované vlastnosti adaptivního systému dosahujeme, můžeme AS rozdělit do těchto základních skupin:

- Adaptivní systémy založené na heuristickém přístupu
- Samočinně se nastavující regulátory *(STC, Self Tuning Controller)*
- Adaptivní systémy s referenčním modelem *(MRAS, Model Reference Adaptive Systems)*

Adaptivní systémy založené na heuristickém přístupu nepoužívají identifikaci modelu systému. Nastavení regulátoru vychází z vyhodnocování průběhu regulované veličiny. Často jsou využívány pro seřízení PID regulátoru.

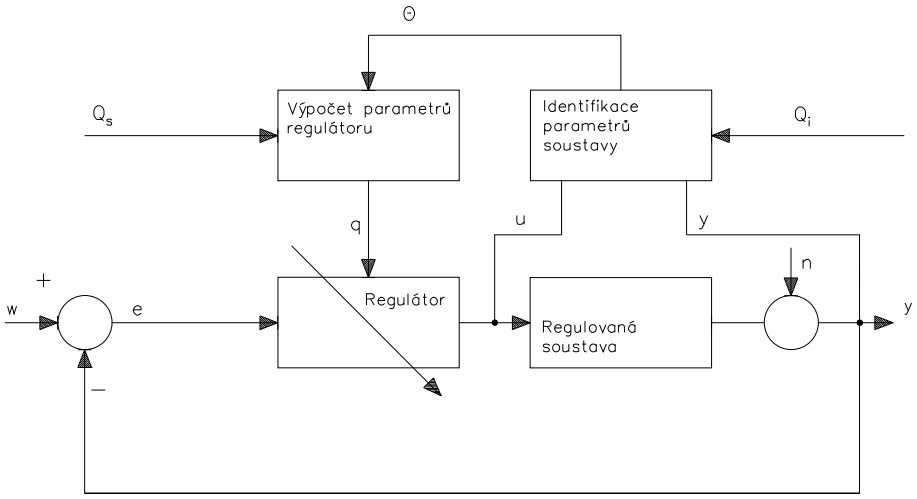
Příkladem AS založeného na heuristickém přístupu, který je blízký našemu problému, je regulátor, který může zařadit do zpětné vazby nelinearitu typu relé a rozkmitat tak systém na kritické frekvenci. Z poměru amplitud 1.harmonické vstupního obdelníkového signálu a amplitdy regulační odchylky se určí kritické zesílení a pro syntézu se použije Ziegler-Nicholsova metoda.

Princip AS s referenčním modelem je nejlépe patrný z blokového schéma na obrázku 1.5. Výstupem referenčního modelu je žádaný výstup y_m případně žádaný stavový vektor x_m. Adaptací hledáme takové parametry stavitelného systému, aby se výstup stavitelného systému co nejvíce blížil referenčnímu. Úkolem adaptace je tedy minimalizovat adaptační odchylku ϵ nebo odchylku stavového vektoru systému od stavového vektoru referenčního.



Obrázek 1.5: Blokové schéma systému s referenčním modelem

Samočinně se nastavující regulátory (STC Self-Tuning Controllers) jsou obvykle, narozdíl od předchozích případů, založeny na průběžné identifikaci modelu soustavy. Takto získaného modelu je následně využito k syntéze parametrů regulátoru. Blokové schéma samočinně se nastavujícího regulátoru je vidět na obrázku 1.6. Na obrázku vidíme tzv. *explicitní STC*. Symboly pro označení vstupů a výstupů bloků mají tento význam: Q_i je kriteriem identifikace, Q_s je kriteriem syntézy, q jsou parametry regulátoru, Θ parametry modelu, n neměřitelné poruchy, y regulovaná veličina, w žádaná hodnota a u akční veličina. Pokud jsou parametry regulátoru určeny přímo bez přepočtu přes parametry modelu Θ, nazýváme *STC implicitní*. Podobného principu lze využít i pro automatické nastavování parametrů regulátoru označované v (Bobál et al., 1999) pojmem *auto-tuning* (ATC, Auto-Tuning Controller). U automatického nastavování parametrů je oddělena seřizovací fáze, která probíhá opakovaně, od vlastní regulace. Ta je prováděna s pevnými parametry. Příkladem automatického nastavení parametrů byl i jednoduchý regulátor uvedený pro ilustraci u heuristických metod adaptivního řízení, využívající po nalezení kritické frekvence a zesílení empirické Ziegler-Nicholsovy metody (John, 2003) určení parametrů regulátoru. Velmi podrobně se problematice auto-tuningu věnuje monografie (Leva et al., 2003). V té můžeme najít definici systému automatického ladění v anglické terminologii obvykle nazývaného autotuner a najít ještě podrobnější rozdělení pojmů automatického ladění a adaptivních regulátorů.



Obrázek 1.6: Blokové schéma samočinně se nastavujícího regulátoru

V adaptivním řízení je nejčastěji používána průběžná (rekurzivní) identifikace, při které jsou odhadovány parametru regresního ARX nebo ARMAX modelu (Havlena a Štecha, 2000). K odhadu je používána metoda nejmenších čtverců. U rekurzivní identifikace jsou data pro identifikaci získávána při uzavřené regulační smyčce. To ale přináší problém, protože přesnost regulace a přesnost identifikace jsou faktory působící proti sobě. Z důvodu stále stejné žádané hodnoty nebo malé regulační ochylky dané přesnou regulací se objevuje lineární závislost v datech a klesá přesnost identifikovaného modelu. Tento problém se snaží řešit teorie duálního řízení, která se snaží o optimální sledování požadované veličiny a která současně klade požadavek na průběh akční veličiny tak, aby byl systém vždy dostatečně vybuzen pro účely identifikace.

Protože řídící systém stroje nemá pro adaptivní algoritmy založené na rekurzívní identifikaci dostatečný výpočetní výkon, ani vhodnou strukturu, bylo nakonec v této práci využito jednorázové identifikace. Využili jsme obdobného principu, jako mají explicitní samočinně se nastavující regulátory, ale identifikaci, která vychází z dat získaných měřením v otevřené smyčce. V terminologii používané v (Bobál et al., 1999) se jedná o automaticky nastavovaný regulátor (ATC). Seřízení regulátoru tak lze provádět před každým řezáním provedením jednoduchého pohybu strojem, při kterém vybudíme servomechanismus v co nejširším frekvenčním rozsahu a získáme tak vhodná data pro identifikaci. Jak již bylo zmíněno výše, pro regulaci bylo využito diskrétní varianty PID (PSD) regulátoru.

Při správném nastavení PSD regulátor vykazuje ve spojení se servomechanismem

velmi dobré řídící účinky a současně řízení zůstává přístupné technikům, kteří mají s tímto typem regulátoru zkušenosti. Hlavní výhodou je ale snadná implementace algoritmu PSD regulátoru, což je důležité vzhledem k omezeným výpočetním možnostem DSP procesoru řídící jednotky servopohonů. Algoritmus regulátoru bude podrobněji popsán v kapitole 3.2.3.

Pro syntézu parametrů regulátoru byla využita frekvenční metoda syntézy, která využívá souvislosti přenosu uzavřené a otevřené smyčky regulačního obvodu. Volbou vhodné fázové bezpečnosti je zaručena robustnost regulačního obvodu vůči nepřesnostem modelu a současně je dána kvalita regulace uzavřené smyčky. Podrobněji se budeme frekvenční metodě syntézy věnovat v kapitole 2.2.

Alternativně byla v rámci této práce ověřena metoda založená na algebraickém výpočtu stabilní oblasti parametrů regulátoru a následné numerické optimalizaci. Tímto způsobem se podařilo najít pro implementaci vhodný algoritmus nalezení P regulátoru minimalizujícího lineární regulační plochu. Algoritmus byl s využitím prostředí Matlab a Maple odzkoušen se stejným modelem servomechanismu, jaký byl použit u frekvenčních metod. Výsledky jsou pro srovnání uvedeny v příloze A. Pro praktický návrh PSD regulátoru se tato metoda ukázala příliš komplikovaná.

Výhodou našeho přístupu je, že máme již před zahájením řezání k dispozici přesný model soustavy, který umožní od začátku přesné seřízení regulátoru. Rovněž dochází k přizpůsobení nastavení regulátoru v případě změn parametrů stroje z různých technologických důvodů, jakými mohou být např. opotřebení stroje, změna provozní teploty, úpravy a seřizování mechanických částí stroje. Vzdali jsme se naopak výhod samočinného nastavení založeného na rekurzivní identifikaci, které v podstatě zavádí zpětnou vazbu vyšší úrovně. Ta umožňuje průběžnou změnou parametrů regulátoru kompenzovat například nelineární chování řízené soustavy nebo zlepšovat regulaci při výskytu nestacionárních poruch.

1.3 Související práce

Tato kapitola je věnována několika pracem, které ilustrují možnosti adaptivní regulace a mají nějaký vzah k zadanému problému nebo použitému řešení.

Zajímavým příkladem implictního STC, ve spojení s klasickým PID regulátorem je (Fujinaka et al., 2000). Samočinné seřízení parametrů regulátoru je zajištěno neu-

ronovou sítí. Autoři ověřili tento přístup simulacemi na regulaci nelineárního systému tvořeného dvojitým inverzním kyvadlem.

Druhým případem adaptivních regulátorů využívajícím algoritmů umělé inteligence, v anglické terminologii nazývaných Soft-Computing, je práce, (Sundareswaran a Begum, 2004) ve které je pro automatické seřízení regulátoru výkonu synchronního generátoru využito genetického algoritmu. Výsledky metody jsou prezentovány simulacemi pomocí linearizovaného modelu synchronního stroje.

Příkladem adaptivního LQ řízení je diplomová práce (Neuhauser, 2004), ve které autor pro vylepšení dynamických vlastností zátěžových strojů nahradil klasickou PID regulaci samočinně se nastavujícím regulátorem využívajícím rekurzivní identifikace a syntézy lineárně kvadratického (LQ) řízení. Řešení bylo implementováno v Matlabu a ověřeno i na reálném zařízení. Přestože byl při experimentech k dispozici mnohem výkonější počítač (Intel Pentium 4 2.4 GHz, RAM 256 MB), než je k dispozici v řídícím systému řezacího stroje, ukázal se výpočetní výkon omezením. Společnou motivací s naším problémem bylo odstranění nutnosti ručního nastavení regulátoru. Zátěžové stroje, respektive zkoušené vzorky, ale narozdíl od řezacího stroje vykazují více nelineární chování. Samočinné nastavení se zde ukázalo přínosem, bylo však bohužel srovnáno pouze s přibližně nastavenou PID regulací. Řešení v uvedené práci umožnovalo i režim automatického nastavení parametrů regulátoru.

V článku (Prokop a Korbel, 2006) je popsán ATC regulátor s identifikací využívající nelinearitu typu relé ve zpětné vazbě. Jedná se o zdokonalený postup využívající principu uvedeného výše v příkladu u adaptivních regulátorů s heuristickým přístupem. Identifikace je v této práci popsána pro symetrické a nesymetrické relé s hysterezí nebo bez. Autorům se tak podařilo odhadovat i přenosy složitější než prvního řádu. Rovněž syntéza regulátoru je oproti využití Zieglerovy-Nicholsovy metody zdokonalena. Autoři využívají algebraického přístupu s parametrizací stabilizujících regulátorů. Přístup k volbě parametru a tím k volbě konkrétního regulátoru je ale empirický a je odhadován podle dominantní časové konstanty systému.

Příspěvek (Leva et al., 2003) je věnovaný seznámení s problematikou automatického ladění PID regulátoru. Je zde uveden souhrn používaných metod pro automatické ladění regulátoru používaných v praxi i těch, které jsou v současnosti předmětem výzkumu. Obsahuje i stručný popis praktických realizací automatického nastavení regulátorů různých firem. Z uvedené literatury se jedná o příspěvek poskytující nejucelenější pohled na danou problematiku.

1.4 Přínos práce

Přínosem této práce je, že nabízí řešení, které umožnuje efektivní implementaci na distribuovaném systému, kde je výpočet akčního zásahu prováděn na jiném procesoru než algoritmus automatického nastavení. Toto řešení je numericky podstatně méně náročné než samočinně se nastavující regulátory jako v předchozí kapitole uvedený příklad adaptivního LQ řízení, ale současně umožnuje dosažení lepších výsledků než jednoduché systémy ATC založené na heuristickém přístupu nebo identifikaci pomocí relé ve zpětné vazbě, které se v praxi často používají. Na rozdíl od metod založených na umělé inteligenci využívajících genetických algoritmů nebo neuronových sítí je ale seřízení regulátoru teoreticky podložené. Díky tomu lze volbou parametrů identifikačního průběhu a fázové bezpečnosti přizpůsobit průběh identifikace a syntézy řízenému pohonu a částečně kompenzovat chyby, které vznikají jeho nelineárním chováním nebo nepřesným měřením.

1.5 Struktura dokumentu

Úvod práce byl věnován seznámení s problematikou CNC řezacích strojů a stručnému shrnutí používaných přístupů k adaptivnímu řízení. Cílem úvodu bylo ale především zdůvodnění volby navrženého algoritmu, který byl v této práci implementován. V teoretické části se podrobněji seznámíme s konkrétním postupem identifikace modelu systému a metody syntézy regulátoru tak, jak byl implementován.

V kapitole 3 se pak budeme zabývat vlastní implementací, která tvoří nejdůležitější část práce. V této kapitole se zaměříme na výběr použitých matematických knihoven, který doplníme opět stručným přehledem dostupných matematických násrojů vhodných pro implementace algoritmů pro řízení. Bude popsáno rozčlenění řešení na jednotlivé části a popsána jejich funkce. Zmíníme se také o použitém algoritmu PSD regulátoru.

V kapitole 4 se pokusíme několika experimenty s reálnými daty demonstrovat chování implementovaného řešení z hlediska praktických požadavků.

Kapitola 2

Teoretická část

Tato kapitola je věnována stručnému teoretickému popisu řešení. V první podkapitole bude popsán algoritmus jednorázové identifikace metodou nejmenších čtverců. Protože pro úspěšnou identifikaci je důležité odhadovat správný řád modelu, zaměříme na začátku kapitoly na tento problém. Ukazuje se, že v praktickém případě není volba přístupu k odhadu řádu modelu jednoznačná. Vrátíme se proto ještě k této problematice v kapitole 4, kde využijeme místo matematického modelováni, využívající apriorní znalosti systému, analýzu naměřených dat. Dále je tato kapitola věnována stručnému teoretickému vysvětlení implementovaných algoritmů s četnými odkazy na podrobnější odvození v literatuře. To se týká i druhé podkapitoly, která je zaměřena na popis syntézy regulátoru.

2.1 Identifikace

Prvním problémem identifikace je určení řádu modelu. Optimální řád modelu můžeme hledat tak, že budeme postupně zvyšovat řád modelu a sledovat konvergenci součtu kvadrátů odchylek dat od modelu (Trnka, 2006) pro n dat (x_i, y_i) od modelu $Y(\theta_i, x_i)$. Pro konstantní počet vzorků n můžeme využít parametru χ^2 daného vztahem

$$\chi^2 = \sum_i (y_i - Y(\theta_i, x_i))^2 \tag{2.1}$$

a nebo pomocí matematického modelování mechanismu. Věnujme se nejprve druhému přístupu, který později v kapitole 4 ověříme na reálných datech pomocí první metody.

Stavové rovnice servomotoru z cizím buzením zatíženého momentem setrvačnosti včetně vlastní setrvačnosti rotoru $J[kg\,m^2\,s^{-1}]$ a konstantním zátěžovým momentem

$M_z[Nm]$ můžeme zapsat například těmito diferenciálními rovnicemi

$$\begin{aligned} \frac{d\ i(t)}{dt} &= -\frac{R}{L}i(t) - \frac{k_e}{L}\omega(t) + \frac{1}{L}u(t) \\ \frac{d\ \omega(t)}{dt} &= -\frac{k_m}{J}i(t) - \frac{b}{J}\omega(t) - \frac{M_z}{J} \\ \frac{d\ \phi(t)}{dt} &= \omega(t) \end{aligned} \tag{2.2}$$

kde i [A] je proud motoru, $\omega[s^{-1}]$ jsou otáčky motoru, $\phi[rad]$ je úhel natočení hřídele, $u[V]$ je vstupní napětí motoru,$R[\Omega]$ rezistivita vinutí $L[H]$ idukčnost vinutí motoru, $b[k \cdot g \cdot m^2 \cdot s^{-2}]$ konstanta tření motoru a mechanismu, $k_e[s \cdot V^{-1}]$ elektrická konstanta motoru a $k_m[k \cdot g \cdot m^2 \cdot s^{-2}]$ mechanická konstanta motoru.

Ze stavových rovnic odvodíme teoretický přenos mezi vstupním napětím $u(t)$ a úhlem natočení hřídele motoru $\phi(t)$ servopohonu, který bude astatický třetího řádu

$$S(s) = \frac{\frac{k_m}{LJ}}{s^3 + (\frac{R}{L} + \frac{d}{J})s^2 + (\frac{Rb + k_e k_m}{LJ})s} \ . \tag{2.3}$$

Identifikací ale odhadujeme přenos diskrétní. Pokud zdiskretizujeme obecný přenos ve tvaru podle (2.3) například metodou ZOH dostaneme přenos v z-transformaci

$$S(z) = \frac{b_2 z^2 + b_1 z + b_0}{z^3 + a_2 z^2 + a_1 z + a_0} \ . \tag{2.4}$$

Neznámé parametry přenosu systému najdeme dále metodou nejmenších čtverců. Podrobné odvození jednorázové identifikace metodou nejmenších čtverců můžeme najít v (Štecha, 2004).

Definujeme vektor neznámých parametrů $\boldsymbol{\Theta} = [a_2\ a_1\ a_0\ b_2\ b_1\ b_0]'$ a řešíme soustavu lineárních rovnic kterou můžeme zapsat maticově

$$\boldsymbol{y} = \boldsymbol{Z^T}\boldsymbol{\Theta} + \boldsymbol{e} \ . \tag{2.5}$$

Matice soustavy má v našem případě tvar

$$\boldsymbol{Z^T} = \begin{pmatrix} -y(k-1) & \cdots & -y(k-3) & u(k-1) & \cdots & u(k-3) \\ -y(k) & \cdots & -y(k-2) & u(k) & \cdots & u(k-2) \\ \vdots & & \vdots & \vdots & & \vdots \\ -y(k+m-2) & \cdots & -y(k+m-4) & u(k+m-2) & \cdots & u(k+m-4) \end{pmatrix}$$

$\boldsymbol{y} = [y(k)\ y(k-1)\ \cdots\ y(k+m-1)]'$ je vektor naměřených hodnot a

$\boldsymbol{e} = [e(k)\ e(k-1)\ \cdots\ e(k+m-1)]'$ je vektor chyb.

Pro řešení soustavy ve smyslu nejmenších čtverců je možné použít metod pseudoinverze, QR rozklad nebo dekompozicí matice podle singulárních čísel (SVD rozklad). Metody řešení jsou popsány rovněž v (Štecha, 2004). V našem případě bylo řešení z numerického hlediska nejvýhodnější s použitím SVD rozkladu, a to jak u výpočtů v Matlabu tak při implementaci funkce identifikace v jazyce C. Vyšší výpočetní náročnost SVD rozkladu oproti dalším dvěma metodám při daném rozměru matice $\boldsymbol{Z^T}$ nezpůsobovala problémy.

Podrobněji se ještě k implementaci identifikace vrátíme v kapitole 3.

2.2 Frekvenční metoda

Frekvenční metoda návrhu regulátoru je podrobně popsána v (John, 2003). Zdrojem informací byla také elektronická pomůcka, (John, 2006) kde je možné také získat demonstrační program frekvenční syntézy pro Matlab `frpid` použitý ke kontrole mezivýsledků výpočtu. Zde bude uvedena pouze základní myšlenka metody, a dále se zaměříme na detaily jejího využití k seřízení PSD regulátoru servomechanizmu.

Princip metody spočívá v úpravě frekvenční charakteristiky otevřené smyčky regulačního obvodu tak, abychom splnili ukazatele kvality regulace uzavřené smyčky, kterými jsou šířka propustného pásma a maximální rezonanční převýšení přenosu uzavřené smyčky. Mezi frekvenčním přenosem otevřené smyčky $G(j\omega) = S(j\omega)R(j\omega)$, kde $S(j\omega)$ je přenos soustavy, $R(j\omega)$ je přenos regulátoru a ω úhlová frekvence a přenosem uzavřené regulační smyčky $F(j\omega)$ je jednoznačný vztah

$$F(j\omega) = \frac{G(j\omega)}{1 - G(j\omega)} \ . \tag{2.6}$$

Chceme-li tedy zaručit vlastnosti uzavřené smyčky, můžeme toho dosáhnout úpravou frekvenční charakteristiky přenosu otevřené smyčky vhodným korekčním členem – regulátorem. Ten v našem případě obsahuje proporcionální, derivační a integrační člen (PID). Budeme-li předpokládat přenos tohoto regulátoru v Laplaceově transformaci ve tvaru

$$R(s) = \frac{r_1(s + \omega_D)(s + \omega_I)}{s} \ , \tag{2.7}$$

pak je regulátor dán třemi parametry – frekvencí derivační ω_D, integrační ω_I, a konstantou r_1. Zvolíme-li si vhodný poměr složek $\frac{\omega_D}{\omega_I}$, určíme parametry ω_D a r_1 tak, abychom

upravili frekvenční charakteristiku a dosáhli zvolené fázové bezpečnosti. Ta říká, o jak velké zpoždění (fázi) si můžeme dovolit zpozdit vstupní signál na frekvenci ω_{PM}, pro kterou má přenos $G(s)$[1] jednotkové zesílení, než se fáze obrátí na -180° a zpětnovazební systém se tak dostane na hranici stability. Tím současně díky souvislosti $G(j\omega)$ a $F(j\omega)$ zaručíme odpovídající kvalitu regulace, tedy maximální šířku přenášeného pásma s minimální hodnotou maximálního rezonančního převýšení.

2.2.1 Použití metody na identifikovaný model servomechanismu

Identifikací metodou nejmenších čtverců popsanou v předchzí kapitole získáme diskrétní přenos servomechanizmu v z-transformaci ve tvaru (2.4).

Pro syntézu regulátoru frekvenčními metodami potřebujeme vypočítat frekvenční charakteristiky systému a regulátoru. Z toho důvodu je pro výpočet frekvenční charakteristiky systému třeba zavést transformaci

$$z = e^{j\omega T_s}; \omega \in \left\langle 0, \frac{\pi}{T_s} \right\rangle , \tag{2.8}$$

kde T_s je vzorkovací perioda diskrétního systému.

Protože je ale předpokládán i diskrétní regulátor, je třeba, zejména při pomalejším vzorkování, ještě uvažovat zpoždění odpovídající vzorkovací periodě T_s, které vzniká časem potřebným pro výpočet regulačního zásahu. To můžeme formálně přidat do systému vynásobením přenosu v z-transformaci (2.4) výrazem z^{-1}, který reprezentuje zpoždění jeden krok.

Nyní je možné aproximovat frekvenční charakteristiku systému výpočtem hodnot přenosu systému po transformaci (2.8) pro hodnoty ω v intervalu

$$\omega \in \left\langle a, \frac{\pi}{T_s} \right\rangle$$

omezeném shora Nyquistovou frekvencí. Spodní hranici intervalu danou parametrem a zvolíme tak, abychom jednak vyloučili prohledávání frevenční charakteristiky v místech, kde identifikovný model už nemůže být přesný a současně kde by vypočtený regulátor neměl fyzikálně smysl. Máme-li průběh frekvenční charakteristiku systému, tedy průběh modulu a fáze přenosu systému v závislosti na úhlovém kmitočtu ω, spočívá návrh regulátoru v přizpůsobení parametrů regulátoru tak, abychom dosáhli zvolené fazové bezpečnosti. Toho pro PID regulátor dosáhneme tímto postupem:

[1] V textu je používána v přenosové funkci substituce $s = j\omega$ Podrobněji v (Horáček, 2001) s.137

Zvolíme Derivační frekvenci regulátoru ω_D tak, aby byla shodná se zatím neznámou frekvencí ω_{PM}, na níž budeme měřit fázovou bezpečnost $\Delta\varphi$ výsledného přenosu otevřené smyčky$G(s)$. Na této frekvenci zvýší PD člen fázi soustavy o 45°.

Frekvenci ω_I zvolíme takovou, aby se vliv integrační korekce na frekvenci $\omega_D = \omega_{PM}$ málo projevil, v našem případě zvolíme například $\omega_D = 10\omega_I$. Musíme tedy zohlednit úhel, o který se sníží fáze otevřené smyčky na frekvenci ω_D vlivem integrační složky. V tomto případě bude úbytek fáze $arctg(\omega_I/\omega_D) = 5,7°$.

Hodnotu najdeme tedy na frekvenční charakteristice soustavy v bodě s fází

$$\begin{aligned} arg(S(j\omega_D)) &= -180 + \Delta\varphi - arg(R(j\omega_D)) \ , \\ arg(R(j\omega_D)) &= +45° - 5,7° \ , \end{aligned} \tag{2.9}$$

kde $\Delta\varphi$ je zvolená fázová bezpečnost, $arg(C),\ C \subset \mathcal{C}$ je fáze komplexního čísla. Poslední parametr regulátoru r_1 určíme z podmínky

$$|G(j\omega_D)| = |S(j\omega_D) \cdot R(j\omega_D)| = |S(j\omega_D)| \cdot \sqrt{2} \cdot r_1 \cdot \omega_D = 1 \ ,$$

kde $|C|\ ,\ C \subset \mathcal{C}$ je modul komplexního čísla. Předchozím postupem jsme získali parametry $\omega_I,\ \omega_D$ a r_1 spojitého regulátoru.

Protože byl pro implementaci PSD regulátoru v driveru využit polohový algoritmus regulátoru (Pivoňka, 2003), je třeba ještě upravit výsledek na konstanty $T_D,\ T_I$ a K, kde T_D je derivační konstanta T_I integrační konstanta a K zesílelní PID regulátoru. Spojitý idealizovaný PID můžeme zapsat v tomto základním tvaru:

$$u(t) = K\left(e(t) + \frac{1}{T_I}\int_0^t e(\tau) + T_D\frac{de(t)}{dt}\right) \ . \tag{2.10}$$

Tomu odpovídá přenos PID v Laplaceově transformaci

$$R(s) = K(1 + \frac{1}{T_I s} + T_D s) \ . \tag{2.11}$$

Konstanty regulátoru s přenosem ve tvaru (2.11) vypočteme takto

$$\begin{aligned} K &= r_1\left(\omega_D + \omega_I\right) \ , \\ T_I &= \frac{\omega_D + \omega_I}{\omega_D\omega_I} \ , \\ T_D &= \frac{1}{\omega_D + \omega_I} \ . \end{aligned} \tag{2.12}$$

Abychom získali diskrétní ekvivalent PID regulátoru, musíme diskretizovat derivační a integrační složku rovnice (2.10). Nejjednodušší možností je nahradit derivaci diferencí 1.řádu. Použitá metoda diskretizace se nazývá dopředná obdélníková metoda. Další metody nejdeme v (Bobál et al., 1999) nebo (Pivoňka, 2003).V této práci vyjdeme z předpokladu, že pro krátkou vzorkovací periodu T_S, vzhledem k dominantní časové konstantě systému, kterou se budeme snažit zaručit není použitá metoda diskretizace důležitá a použijeme zmíněnou obdelníkovou metodu. Derivaci tedy aproximujeme diferencí

$$\frac{de(t)}{dt} \approx \frac{e(k) - e(k-1)}{T_s} \tag{2.13}$$

a integrál aproximujeme sumací

$$\int_0^t e(\tau) \approx T_s \sum_{i=1}^{k} e(i) \ . \tag{2.14}$$

S použitím těchto aproximací má rovnice PSD regulátoru tvar

$$u(k) = K \left(e(k) + \frac{T_s}{T_I} \sum_{i=1}^{k} e(i) + \frac{T_D}{T_s} e(k) - e(k-1) \right) \ . \tag{2.15}$$

Z této rovnice vychází polohový algoritmus regulátoru PSD (Pivoňka, 2003), použitý v této práci. Algoritmus bude podrobněji popsán v kapitole 3. Na závěr syntézy regulátoru jsou tedy ještě konstanty přepočteny tak aby je bylo možné přímo využít při výpočtu akčního zásahu v algoritmu. Výstupem jsou tedy konstanty K, $\frac{T_s}{T_I}$ *a* $\frac{T_D}{T_s}$. Je třeba ale ověřit, jestli pro dané parametry konstant regulátoruT_I, T_D *a* K a vzorkovací frekvenci T_S je dynamika PSD regulátoru obdobná PID. Této problematice je věnována zejména pozornost v (Bobál et al., 1999). Zde uvedeme podmínku upravenou na tvar vhodný k ověření konstant použitého polohového algoritmu PSD

$$\frac{T_s}{T_I} \ < \ \frac{T_D}{T_s} \tag{2.16}$$

a v kapitole 4 ověříme jejich platnost pro zvolenou periodu $T_s = 0,002s$.

Kapitola 3

Implementační část

V popisu implementace algoritmu seřízení regulátoru stroje je těžiště celé práce. Problematice implementace se nevyhneme při žádném pokusu o využití teoreticky zvládnutých metod řízení v průmyslovém provozu. Programování v prostředí Matlab je snadné, prostředí poskytuje formou Toolboxů připravené nástroje z celé řady oblastí. Výsledné řešení ale nevyhoví náročnému průmyslovému prostředí, kde je často využíváno z důvodů spolehlivosti a odolnosti vnějším vlivům méně výkonných počítačů, které díky tomu nepotřebují nucené chlazení a pracují v širším teplotním rozsahu. V laboratorních podmínkách je naopak využíváno většinou bežných osobních počítačů PC, jak tomu bylo například i v případě adaptivního řízení zátěžových strojů v (Neuhauser, 2004), což dovolí i menší efektivitu implementace algoritmů řízení. Pro cílovou praktickou aplikaci je tedy vždy nakonec nutné převedení výpočtu do nízkoúrovňového jazyka jakým je zejména jazyk C.

Velkou výhodou jazyka C je přenositelnost zdrojového kódu na různé platformy. Toho bylo využito i v tomto případě, kdy byl software vyvíjen a laděn na počítači PC. Řešení je pak snadno přenositelné na cílovou platformu, v našem případě průmyslový počítač s operačním systémem Linux.

Přestože bylo snahou vybrat z teorie řízení co nejjednodušší postup, kterým lze samočinné seřízení regulátoru spolehlivě zajistit, neobejde se výpočet bez náročnějších matematických výpočtů. Jedná se zejména o výpočty spojené s sestavením a řešením soustavy lineárních rovnic (2.5). V tomto případě, kdy pracujeme při identifikaci s 2000 vzorky, bude potřebný rozměr matice $\boldsymbol{Z^T}$ roven 2000×6.

Frekvenční metoda syntézy je založena na výpočtu frekvenčních charakteristik. Z hlediska implementace bylo třeba se vyrovnat s transformací (2.8) potřebnou pro výpočet frekvenční charakteristiky identifikovaného systému. Znamená to efektivní výpočet hodnoty komplexní exponenciely, a následně pro účely dosazení do přenosu (2.4) vyčíslit

hodnotu polynomu v oboru komplexních čísel.

V příloze je ještě zmíněna polynomiální metoda syntézy, která byla při volbě metody syntézy rovněž vyzkoušena, a je uvedena pro srovnání. Ta by vyžadovala pro implementaci nalezení stabilních oblastí zejména vyčíslení kořenů polynomů. Numericky výhodné řešení tohoto problému je sestavení "Companion matrix" s následným hledáním vlastních čísel této matice. Fibonachiho metoda použitá pro nalezení optima v rámci stabilní oblasti se ukázala přes svou jednoduchost v jednorozměrném případě velmi efektivní a s její implementací by nebyl problém.

V další kapitole se budeme věnovat výběru a popisu vhodné matematické knihovny v jazyce C, která by umožnovala řešení výše vyjmenovaných matematických problémů.

3.1 Volba a popis použitých matematických knihoven

Mimo podpory výše uvedených matematických výpočtů, je třeba aby knihovny byly volně dostupné pod nekomerční licencí a byly pro vyšší efektivitu psané v jazyce C.

Poměrně podrobný přehled matematických knihoven lineární algebry je uveden v (Halmo, 2004). Ze zde uvedených knihoven přicházejí v úvahu tyto:

- BLAS *(Basic Linear Algebra Subprograms)* `<http://www.netlib.org/blas/>`
- LAPACK *(Linear Algebra PACKage)* `<http://www.netlib.org/lapack/>`
- ATLAS *(Automatically Tuned Linear Algebra Software)* `<http://math-atlas.sourceforge.net/>`

BLAS (Basic Linear Algebra Subprograms) je knihovna napsaná v jazyce Fortran77. Knihovna BLAS je členěna na tři úrovně.

- Level 1 vektorové operace
- Level 2 operace matice-vektor
- Level 3 operace matice-matice

Knihovna BLAS neobsahuje nástroje vyšší lineární algebry potřebné pro řešení metody nejmenších čtverců, je ale zajímavá tím, že díky svému značnému rozšíření poskytuje

standartní interface pro výpočty nižší úrově vetšiny matematických knihoven. K dispozici je také verze v jazyce C, která byla vygenerována automatickým převodem z Fortranu (CBLAS).

Knihovna LAPACK (Anderson et al., 1999) je rovněž napsaná v jazyce Fortran77. Poskytuje funkce pro řešení soustav lineárních rovnic, problému nejmenších čtverců a maticové rozklady. Pro svůj běh potřebuje knihovnu BLAS. Využívá ji mnoho komerčních i nekomerčních programových produktů (např. Matlab, Octave, Scilab). Stejně jako v případě BLASu, existuje automaticky generovaná verze v jazyce C (CLAPACK)

Knihovna ATLAS (Whaley a Petitet, 2005) v současnosti, poskytuje C a Fortran77 interface k implementaci knihovny BLAS optimalizované pro různé platformy. Obsahuje také některé funkce z knihovny LAPACK. Knihovnu ATLAS využívájí pro výpočty nižší úrovně například produkty Matalb od verze 6 a Maple od verze 7. Její význam spočívá především v automatické optimalizaci kódu podle cílové platformy.

Nejvýhodnější knihovnou pro implementaci algoritmu v této práci se nakonec ukázala knihovna GSL-GNU Scientific Library (Galassi et al., 2006). S GSL knihovnou se seznámíme v další kapitole.

3.1.1 Popis knihovny GSL-GNU Scientific Library

GSL, jejíž koncepce vznikla v roce 1996 zásluhou Dr M. Galassiho a Dr J. Theilera z národní laboratoře v Los Alamos, narozdíl od knihovny LAPACK pokrývá mnohem širší oblast matematiky než jen lineární algebru. Knihovna je zaměřena na operační systémy odvezené od Unixu, lze ji používat v C i C++ programech. Předkompilovaná je součástí některých Linuxových distribucí, jako je například Debian Linux, není objektová a je psána v ANSI C (C89). To je v našem případě výhodné z hlediska rychlosti výsledného kódu a usnadňuje to pozdejší kompilaci programu pro procesor řídícího systému. Přetěžování funkcí, které by bylo výhodné pro přehledný zápis funkcí pro různé datové typy, je nahrazeno jednotným způsobem modifikace jména funkce podle datového typu, pro který je určena. Například funkce fn z imaginárního modulu `gsl_foo` pro typ `double` se volá:

```
gsl_foo_fn
```

Stejná funkce například pro typ `long double` se volá:

```
gsl_foo_long_double_fn
```

Knihovna GSL potřebuje pro svou činnost knihovnu CBLAS a poskytuje interface k jejím funkcím. GSL může být pro lepší výkon slinkována také s knihovnou ATLAS a jejím CBLAS iterface.

Informace o projektu najdeme na domovských internetových stránkách `<http://www.gnu.org/software/gsl/>`.

Na této adrese je také možné získat literauru (Galassi et al., 2006).

3.2 Popis činnosti a struktury programu

Jak bylo uvedeno v kapitole 1.1 věnované struktuře řídícího systému, program je rozdělen na část pracující v reálném čase, která je implementována na DSP procesoru příslušného serva a nadřízenou část, která je prováděna v nadřízeném počítači s operačním systémem Linux. Za účelem odladění algoritmu je pro komunikaci použito sériové linky a jako nadřízený počítač je použito osobního počítače PC. Cílovou platformou je ale průmyslový počítač popsaný v kapitole 1.1.

Algoritmus seřízení regulátoru je prováděn v následujících krocích:

1. Data popisující paramery a průběh napětí přivedeného na servomotor pro identifikaci jsou vygenerována v nadřízeném počítači

2. Před identifikací jsou tato data zaslána řídící jednotce, v takovém formátu, aby bylo možné je uložit do interní paměti DSP procesoru o velikosti 256 Byte.

3. V průběhu identifikace pro každý vzorek čte řídící jednotka servomotoru z interní paměti hodnoty napětí, nastavuje PWM modulátor a snímá hodnotu úhlu natočení hřídele servomotoru. Protože nemá driver pro uložení dat dostatek paměti, odesílá je hned nadřízenému počítači.

4. Po ukončení identifikačního pohybu servopohonu je v nadřízeném počítači je provedena identifikace a syntéza regulátoru

5. Zesílení kanálů regulátoru jsou přeneseny do řídící jednotky, kde je implementován algoritmus PSD regulátoru.

Program v nadřízeném počítači je rozdělen do tří nezávislých částí. Jedná se o programy: `ident`, `model_ls` a `fr_PSD`. Mezivýsledky jednotlivých programů jsou ukládány

do souborů tak, aby byla možná jejich kontrola a popřípadě další analýza například pomocí Matlabu. Program je samozřejmě možné spustit celý najednou dávkovým příkazem. V tabulce 3.1 jsou přehledně znázorněny vstupy a výstupy jednotlivých částí programu. V dalších podkapitolách popíšeme podrobněji funkci jednotlivých programů.

příkaz	`ident`	`model_ls`	`fr_PSD`
parametry	`tty V H`	`n <phidata.txt`	$\Delta\varphi$ $\Delta\omega$ `<model.dat`
vstupní soubor	`idata.bin`	`phidata.txt`	`model.dat`
výstupní soubor	`phidata.txt`	`model.dat`	`-`

Tabulka 3.1: Členění programu

3.2.1 Implementace identifikace

Obecným požadavkem na volbu průběhu napětí během identifikace z hlediska systému je vybudit identifikovaný systém v co nejširším frekvenčním spektru, aby se frekvenční charakteristika identifikovaného modelu co nejvíce blížila reálnému systému. Dále je nutné zohlednit vliv nelinearit a nepřekračovat limitní hodnoty parametrů z hlediska zatížení identifikovaného systému.

Z hlediska implementace bylo třeba signál definovat tak, aby jej bylo možné realizovat řídící jednotkou servopohonu. Příklad průběhu, který se pro řešení osvedčil je znázorněn na obrázku 3.1. Je definován ustálenou hodnotou V a hodnotou skoku H který se od ustálené hodnoty odečítá. Průběh může mít až 4000 vzorků, které jsou uloženy v binárním souboru, ve kterém každý bit odpovídá jednomu vzorku. Význam nastavení daného bitu n je znázorněn tabulce 3.2.

bit	hodnota PWM
n=1	V
n=0	V-H

Tabulka 3.2: Definice průběhu napětí

Jednotlivé bity jsou uloženy v binárním souboru `data.bin` po 32bitech jako čísla typu integer. Obsah souboru byl vygenerován programem v prostředí Matlab, jako obdélníkový průběh s náhodně volenou dobou trvání jednotlivých skoků. V dalších podkapitolách nejprve popíšeme způsob použití programů, `ident` a `ls_model` které identifikaci provádějí.

Na závěr se budeme věnovat problémům společným celé identifikaci jako je volba průběhu napětí při identifikaci, možnosti jeho úpravy a periodě vzorkování použité při identifikaci.

3.2.1.1 Snímání dat pro identifikaci modelu servopohonu

Na obrázku 3.1 vidíme průběh napětí na motoru, v hodnotách ukádaných do registru modulátoru PWM v průběhu identifikace. Rozsah vstupních hodnot PWM je v intervalu $< -16000, 16000 >$. Průběh je definovám ustálenou hodnotou, která je v tomto případě $V = 10000$ a hodnotou skoku $H = 5000$, který se od ustálené hodnoty odečítá. Průběh napětí při identifikaci program `ident` negeneruje pokaždé jiný, ale používá průběh definovný v binárním souboru `idata.bin`. Program s pomocí tohoto souboru a zadaných parametrů V a H vytvoří hlavičku.

Odesláním hlavičky řídící jednotce servopohonu zahajuje nadřízený počítač identifikaci. DSP procesor řídící jednotky uloží průběh do paměti a zajistí rozběh po vhodné rampě na rychlost V. Rozběh je realizován v otevřené smyčce lineárním zvyšováním napětí. Strmost nárůstu je volena s ohledem na mechanické zatížení servopohonu. Díky tomu že je rozběh prováděn v otevřené smyčce, není třeba před prvním laděním regulátoru žádné znalosti o dynamice regulovaného pohonu. Po rozběhu program v nadřízeném počítači přijímá od řídící jednotky data s přírůstky natočení rotoru serva a po skončení je uloží do souboru `phidata.txt`.

3.2.1.2 Identifikace modelu servopohonu

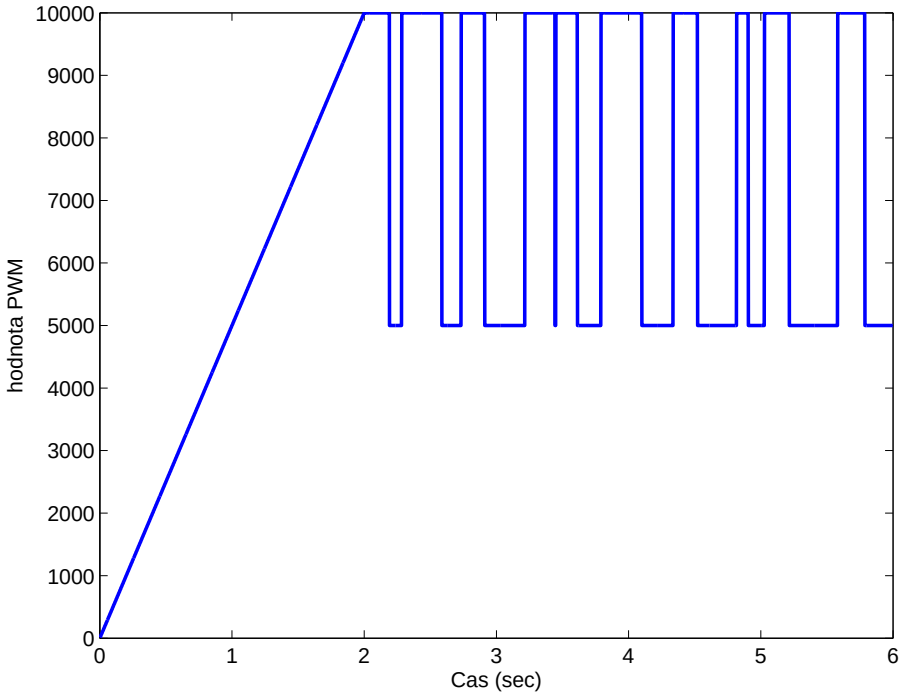
Program `model_ls` je implementací identifikace modelu servopohonu metodou nejmenších čtverců posané v kapitole 2.1. Pro hledání n parametrů obecného lineárního modelu metodou nejmenších čtverců poskytují knihovny GSL přímo funkci, která využívá pro řešení rovnice (2.5) SVD rozkladu. Prototyp funkce je v hlavičkovém souboru `gsl/gsl_multifit.h`. Hlavním úkolem programu je tedy sestavení matic $\boldsymbol{X}$ *a* $\boldsymbol{y}$ [1] které tvoří vstupní parametry funkce

`gsl_multifit_linear_svd (X, y, tol ,&rank, c, cov,&chisq, work)`.

Další parametry funkce mají tento význam:

- `tol` - Vstupní parametr, který definuje maximální poměr singulárníních čísel matice $\boldsymbol{S}$. Je-li $\frac{s_i}{s_0} < tol$ dojde k redukci hodnosti matice.

[1] V manuálu GSL a programu je použito toto značení $\boldsymbol{\Theta} \equiv \boldsymbol{c}, \boldsymbol{Z^T} \equiv \boldsymbol{X}$



Obrázek 3.1: Průběh napětí motoru při identifikaci

- `rank` - Parametr vrací hodnost matice soustavy, tedy počet parametrů řešení (řád modelu).
- `cov` - Parametr vrací kovarianční matici parametrů modelu. Kovarianční matici lze využít k výpočtu standartní odchylky modelu v daném bodě. Pro identifikaci není využita.
- `chisq` χ^2 - Je definován rovnicí 2.1 kapitole 2.1.

Výsledný vektor parametrů sestavený v pořadí $\boldsymbol{c} = [a_2\ a_1\ a_0\ b_1\ b_0]'$ a hodnota χ^2 vypočtená podle rovnice (2.1) je pro další zpracování uložen programem do souboru `model.dat`.

3.2.1.3 Průběh napětí při identifikaci

Protože program `ident` umožnuje změnit průběh napětí tak, aby byl výpočet parametrů modelu pomocí SVD rozkladu numericky stabilní, je důležitá volba parametru `tol`, protože tím můžeme zabránit syntéze regulátoru pomocí nepřesného modelu. Hodnota použitá v programu $tol = 10^{-9}$ byla určena experimentálně podle nejhoršího poměru singulárních čísel matice $\boldsymbol{S}$ za situace, kdy výpočet modelu s předpokládaným řádem

probíhal ještě s dostatečnou přesností. Přestože ani během dalších experimentů nebylo třeba identifikační průběh měnit, diskutujme ještě stav, kdy je efektivní hodnost soustavy menší a získáme model s menším počtem parametrů. Signalizuje to situaci, kdy se snažíme odhadovat složitý model s nedostatečnou informací obsaženou v nasnímaných datech. I když by bylo teoreticky možné pracovat s při syntéze s jednodušším modelem, je tato situace v implementovaném řešení považována za chybnou.

V této situaci je vhodné změnit parametry programu `ident`, které umožňují nastavit napěťové úrovně identifikačního signálu. Další možností je změna identifikačního průběhu. Nejsnáze toho lze docílit změnou obsahu souboru `idata.bin`. Ten byl pro účely experimentů provedených v rámci této práce vygenerován pomocí programu v prostředí Matlab výše popsaným způsobem. Není proto problém vygenerovat pro případ reálných aplikací s výrazně odlišnou dynamikou servopohonů jiný soubor s identifikačními průběhem.

Vzhledem k tomu že soubor `idata.bin` používaný během experimentů vyžíval jen polovinu z maximální možné délky identifikačního průběhu, která je 4000 vzorků a je omezena pamětí řídící jednotky, lze předpokládat, že nebude problém ani v reálné aplikaci přispůsobit identifikační signál vlastnostem konkrétního pohonu.

I když to není principielně nutné, je celý algoritmus automatického nastavení navržen tak, že předpokládá stejnou periodu vzorkování jak pro identifikaci tak pro syntézu a vlastní regulaci. Jiné vzorkování při identifikaci by bylo možné za předpokladu převzorkování modelu získaného pro následnou syntézu. Převzorkování modelu bylo využito k simulaci vlivu pomalejšího vzorkování na kvalitu regulace v kapitole 4 provedenou v prostředí Matlab. Pro využití algoritmu automatického nastavení ale odlišné vzorkování nemá význam a změna identifikačního průběhu pomocí úpravy souboru idata.bin je plně dostačující. S danou vzorkovací periodu $T_s = 0.002s$ a nejvetším počtem 4000 vzorků je maximální doba identifikace 8 sekund. Tato doba je vhodná pro technickou realizaci vzhledem k rychlostem pojezdů řezacího stroje, kde bude potřeba zajistit, aby byl pohyb uskutečněn v rámci pracovního rozsahu stroje, a snímat data pro obě osy pohybu popřípadě i pro servopohon výšky hořáku, což lze uskutečnit postupným opakováním identifikace pro každý pohon zvlášť.

3.2.2 Syntéza regulátoru

Syntéza regulátoru implementována programem `fr_PSD`. Ten je narozdíl od přechozích komplikovanější a je proto rozdělen na dvě části. Modul `fr_char` poskytuje externí funkci s těmito parametry:

```
extern void bode(gsl_vector *mag, gsl_vector *phase, gsl_vector *omega,
const gsl_vector *num,const gsl_vector *den,const double Ts,
const double omega_min, const int steps)
```

Tato funkce je využívána programem `fr_PSD` pro výpočet frekvenční charakteristiky diskrétního systému s přenosem, jehož koeficienty čitatele jsou uloženy ve vektoru `num` a jmenovatele `den`. Funkce vrací hodnoty amlitudy ve vektoru `mag` a fáze `phase`. Odpovídající frekvence jsou uloženy ve vektoru `omega`. Charakteristika je vypočítána pro počet bodů uložených v parametru `steps` v rozsahu od `omega_min` do Nyquistovy frekvence π/T_s. Vzorkovací perioda T_s je předána v parametru `Ts`. Frekvenční charakteristika je počítána v logaritmických souřadnicích. Pro konkrétní data se ukazuje, že parametry regulátoru jsou vypočteny s dostatečnou přesností už při frekvenční charakteristice dané 64 body. Z hlediska trvání výpočtu ale není problém použít daleko jemnější dělení. V programu `fr_PSD` je proto využíváno dělení 512 bodů.

Protože je během syntézy frekvenční metodou třeba výpočet frekvenční odezvy několikrát, byl modul `fr_char` pojat obecněji. Díky většímu počtu parametrů je možné jej použít nejen pro výpočet frekvenční odezvy diskrétního systému v libovolném počtu kroků, ale umožnuje i výpočet frekvenční odezvy spojitého systému pro jednu danou frekvenci. To je potřebné k výpočtu korekcí daných derivační a integrační složkou regulátoru, který uvažujeme při návrhu jako spojitý.

Knihovny GSL nabízejí funkci vyčíslení hodnoty polynomu pouze v oboru reálných čísel. Pro vyšší efektivitu výpočtu frekvenční odezvy ve funkci `freqresp`, která je součástí modulu `fr_char` bylo vyčíslení hodnoty polynomů čitatele a jmenovatele implementováno tímto způsobem.

Vektor s koeficienty polynomu byl převeden také do oboru komplexních čísel. Zavedli jsme vektor s mocninami dané hodnoty proměnné stejné délky jako vektor s koeficienty. Následně byl polynom vyčíslen skalárním násobením s vektorem `znum, zden` s mocninami daného bodu. Algoritmus tak lze zapsat pomocí knihovny GSL velmi kompaktním zápisem.

Vypsaný fragment kódu počítá frekvenční odezvu v bodech připravených ve vektoru `s`. Pro každý prvek `s` spočte vektory `znum` s délkou danou stupněm čitatele `nn` a `zden` s délkou `nd`. Dále jsou vektory skalárně vynásobeny a výsledky vyděleny.

```
for (i=0; i<steps; i++)
 {
 z=gsl_vector_complex_get(s,i);
```

```
 for (j=0;j<nn;j++) //nn stupen jmenovatele
  gsl_vector_complex_set(znum,(nn-1)-j,gsl_complex_pow_real(z,j));
 for (j=0;j<nd;j++) //nd stupen citatele
  gsl_vector_complex_set(zden,(nd-1)-j,gsl_complex_pow_real(z,j));
 gsl_blas_zdotu(numc,znum,&numval);
 gsl_blas_zdotu(denc,zden,&denval);
 if (GSL_REAL(denval)==0 && GSL_IMAG(denval)==0)
  printf("singularity in frequence response");
 gsl_vector_set(mag,i,gsl_complex_abs(gsl_complex_div(numval,denval)));
 gsl_vector_set(phase,i,gsl_complex_arg(gsl_complex_div(numval,denval)));
}
```

Tento algoritmus byl uveden jednak proto, že je dobrým příkladem využití GSL knihoven a jejich BLAS interface, ale také proto, že se jedná z hlediska výpočetní náročnosti syntézy o klíčové místo. Jak bylo uvedeno výše, bylo prakticky ověřeno, že rychlost výpočtu frekvenční charakteristiky je pro daný systém a požadovanou přesnost tímto způsobem naprosto vyhovující.

Program `fr_PSD` s využitím modulu `fr_char` provádí syntézu PID postupem uvedeným v kapitole 2.2 pro model soustavy daný souborem `model.dat` a zadaný parametry požadované fázové bezpečnosti $\Delta\varphi$ a poměrem složek $\Delta\omega = \omega_D/\omega_I$. Vypočtené konstanty PID jsou ještě přepočteny pro algoritmus polohového PSD regulátoru tak, aby v programu pro DSP procesor nebylo třeba násobit a dělit. Pro řídící jednotku jsou tedy připraveny konstanty K, $\frac{T_s}{T_I}$ a $\frac{T_D}{T_s}$.

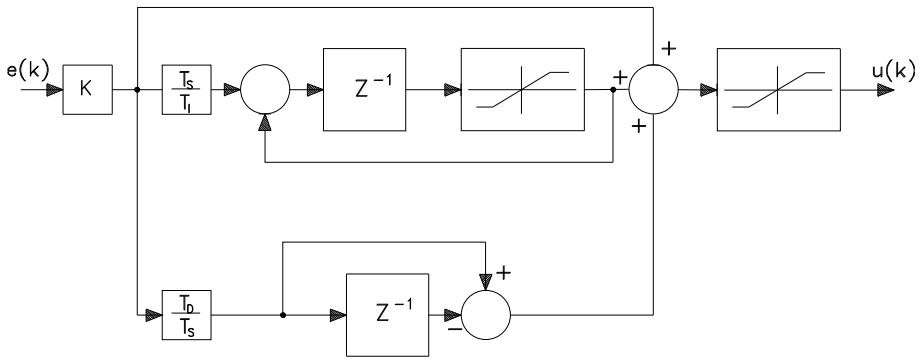
3.2.3 Algoritmus PSD regulátoru

Na obrázku 3.2 vidíme stavové schéma polohového regulátoru PSD (Pivoňka, 2003). Schéma vychází z rovnice (2.15), je ale doplněno o omezení sumační složky jako opatření proti windupu (John, 2003). Výstupní akční veličina je rovněž omezena rozsahem PWM. Stavové schéma lze zapsat algoritmem uvedeným rovněž v (Pivoňka, 2003).

Algoritmus PSD regulátoru byl naprogramován ve vývojovém prostředí pro DSP procesor Metrowerks v jazyce C. Pro vyšší efektivitu jsou výpočty prováděny v pevné řádové čárce s přesností integer 16 bitů.

```
{s[2] = Výstup_z_procesu;
s[1] = W - s[2];
```

```
s[3] = K*s[1];
s[4] = s[5];
if (s[4] > umax ) s[4] = umax ;
if (s[4] < umin) s[4] = umin ;
s[5] = T/TI *s[3] + s[4];
s[6] = s[3] + s[4] + TD/T*(s[3] -s[7]);
s[7] = s[3];
if (s[6] > umax ) s[6] = umax ;
if (s[6] < umin ) s[6] = umin ;
Akce_do_procesu(s[6]);}
```



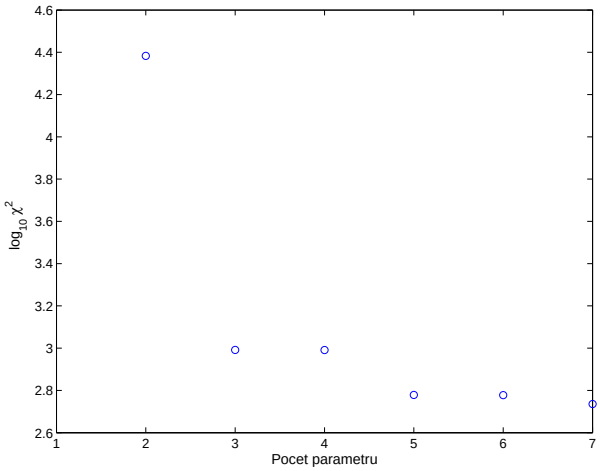
Obrázek 3.2: Schéma polohového regulátoru PSD

Kapitola 4

Simulace, experimenty

4.1 Identifikace - určení řádu modelu

V kapitole 2.1 byl pomocí fyzikálního modelování servomechanizmu určen předpokládaný spojitý přenos modelu. Jeho diskretizací jsme odhadli teoreticky potřebný počet parametrů, které je třeba získat pomocí identifikace. Protože se už ve fázi testování algoritmu identifikace na syntetických datech ukázalo, že koeficient nejvyšší mocniny čitatele diskrétního přenosu má zanedbatelný význam a navíc jeho hodnota je numericky nestabilní, bylo provedeno určení řádu modelu ještě tak, že jsme s reálnými daty získanými měřením servopohonu provedli opakovaně identifikaci pro vzrůstající počet parametrů a sledovali konvergenci parametru χ^2 vypočteného podle vztahu (2.1). Hodnoty χ^2 jsou pro 2-7 parametrů modelu vyneseny v grafu na obrázku 4.1.

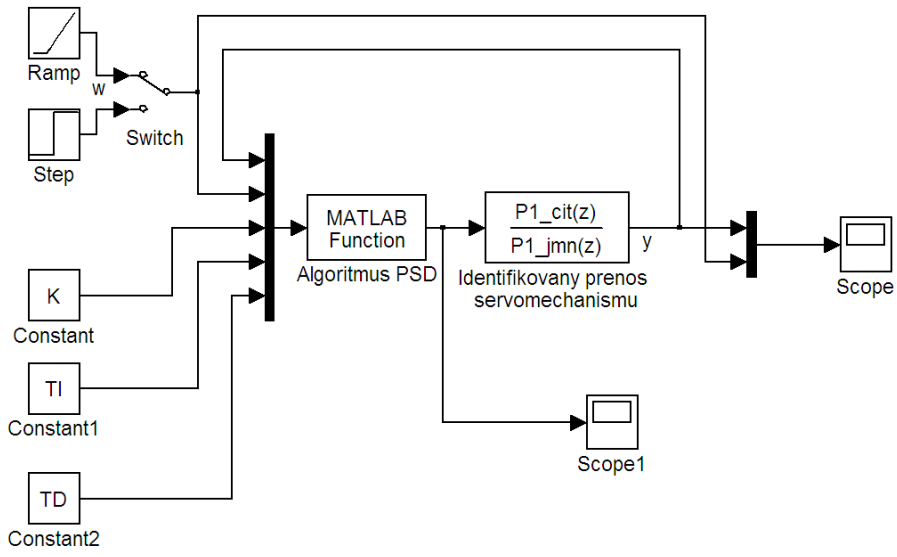


Obrázek 4.1: Závislost logaritmu χ^2 na počtu parametrů modelu

V souladu s pozorováním se potvrdilo, že pro systém reprezentovaný naměřenými daty je optimální řád modelu 5. Proto se v kapitole 3, kde je popsána implementace identifikace, programem `ls_model` předpokládá vektor parametrů ve tvaru $\boldsymbol{c} = [a_2\ a_1\ a_0\ b_1\ b_0]'$, a parametr b_2 je vynechán.

4.2 Simulace algoritmu PSD v Simulinku

Pro ověření výsledků syntézy regulátoru, se zvoleným algoritmem PSD regulátoru bylo využito možnosti Simulinku volat zvolenou externí funkci napsanou v Matalbu. Simulační schéma vidíme na obrázku 4.2. V bloku MATLAB function je volán s vzorkovací periodou $T_s = 0.002s$ polohový algoritmus PSD regulátoru, který byl uveden v podkapitole 3.2.3. Vstupními parametry funkce jsou navržené konstanty regulátoru a žádaná hodnota. Akční veličina vstupuje do systému simulovaného identifikovaným modelem. Ten je ale bez dopravního zpoždění, které je přidáno do systému za účelem syntézy. Tato simulace byla pro srovnání provedena i s jednoduššími typy PD a P regulátoru, navrženými rovněž frekvenční metodou syntézy se stejnou fázovou bezpečností.



Obrázek 4.2: Simulační schéma pro algoritmus PSD regulátoru

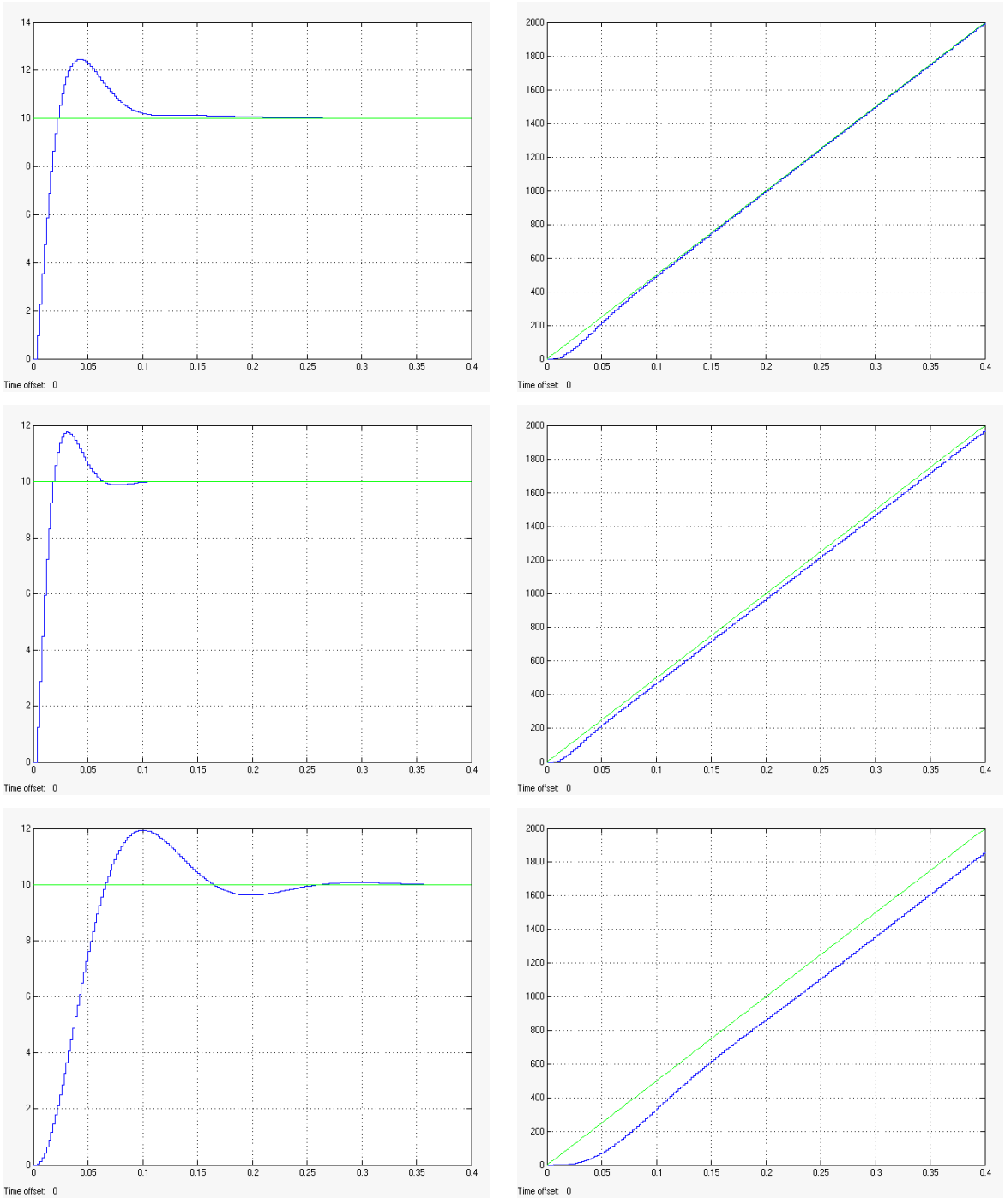
Na obrázku 4.3 jsou přehledně uspořádány charakteristiky s výsledky jednotlivých simulací pro různé typy regulátorů a vstupní signály typu skok a rampa. Aby bylo možné

mezi sebou porovnávat přechodové charakteristiky pro jednotlivé typy regulátorů, je velikost žádané hodnoty zvolena tak, aby u žádného typu regulátoru ještě nedocházelo k omezení akční veličiny. Konstanty P, PD i PSD regulátoru jsou navrženy frekvenční metodou se stejnou fázovou bezpečností $\Delta\varphi = 45°$

Ze simulací je vidět, že dynamika PD a PSD při odezvě na skok žádané polohy je podobná. PD regulátor ale podle předpokladu již při sledování pohybu konstantní rychlostí vykazuje znatelnou trvalou odchylku. Dynamika P regulátoru je oproti PD i PSD podstatně horší. Rovněž trvalá odchylka při sledování rychlosti (na vstupu je lineární rampa) je ještě větší než u PD. Celkově výsledky simulací odpovídají teoretickým předpokladům. Porovnáním regulátorů navržených pro stejnou fázovou bezpečnost ukazuje, že z hlediska doby regulace a trvalé odchylky má použití PSD regulátoru pro dynamiku servomechanismu výrazný přínos. Navíc můžeme pozorovat, jak pomocí parametru $\Delta\omega$ syntézy můžeme ovlivnit kompromis mezi rychlostí přechodového děje, která je maximální u PD regulátoru a ovlivňuje kvalitu výpalku ve vrcholech a přesností sledování rampy, které má zase vliv na přesnost rozměrů výpalku. Přechodová charakteristika PSD regulátoru odpovídá parametru $\Delta\omega = 5$

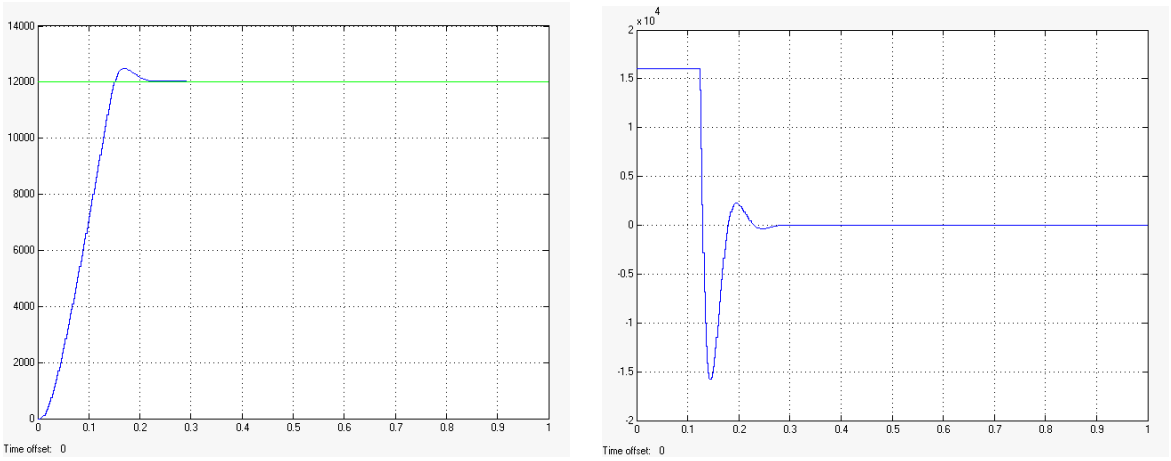
Další simulace, které vidíme na obrázku 4.4, naopak ukazují chování systému za předpokladu omezení akční veličiny, aby bylo možné srovnání s reálnými přechodovými charakteristikami uvedenými v následující kapitole. Přestože je v algoritmu implemetováno omezení integrační složky, při nastavené hodnotě tohoto omezení na maximální rozsah PWM, ukázal se i tak nepříznivý vliv zbytkového windupu. V tomto případě ale postačilo snížení omezení integrační složky na hodnotu -5000, 5000. Nebylo nutné zavádět do algoritmu řízení žádné dodatečné nulování integrační složky. Tato hodnota postačí k překonání počáteční necitlivosti i k dostatečně přesnému sledování rampy a současně se rychle vynuluje, dojde-li k jejímu dosažení z důvodu omezení akční veličiny. Průběhy které jsou na obrázku 4.4 a obrázcích 4.5, 4.6 a 4.7 s charakteristikamy naměřenými s reálným servopohonem jsou pro toto nastavení omezení integrační složky. Výstupní hodnoty se měří v impulsech snímače, vstupní jsou dány PWM a mají rozsah -16000, 16000.

V další podkapitole bude testováno chování reálného servopohonu, pro podobné nastavení regulátoru a se stejnou hodnotou omezení integrační složky.



Odezva na jednotkový skok Odezva na skok rychlosti

Obrázek 4.3: Simulace pro PSD, PD a P regulátor



Odezva systému Průběh akční veličiny

Obrázek 4.4: Simulace PSD regulátoru s omezením akční veličiny,$\Delta\varphi = 45°, \Delta\omega = 5$

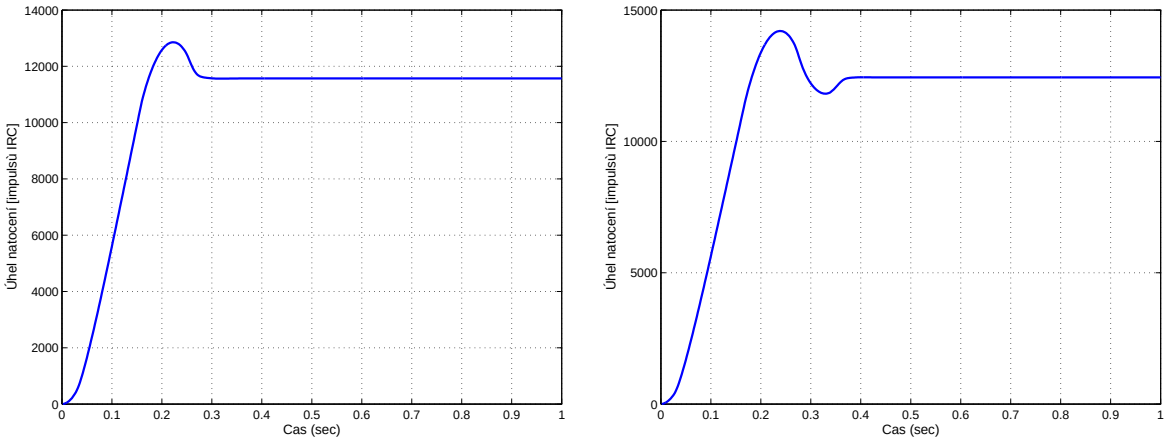
4.3 Experimenty na reálném servopohonu

Zatímco výsledky předchozích simulací byly určeny především pro ověření vhodnosti použitého algoritmu PSD regulátoru, neboť stále vycházely z identifikovaného lineárního modelu a nepředpokládaly vliv nepřesnosti modelu nebo měření, v následujícím případě byl experiment proveden s reálným servomotorem. Snahou bylo připravit experiment tak, aby bylo možné srovnání s předchozími simulacemi na obrázku 4.4. Použili jsme obdobný typ motoru jako je na řezacím stroji. Experimenty byly však vzhledem k obtížné dostupnosti řezacího stroje prováděny na samotném servopohonu s převodovkou. Vzhledem k tomu, že snímání úhlu natočení zůstává po namontování servopohonu na řezací stroj na hřídeli motoru, jsou rozdíly z hlediska návrhu řízení minimální. Z hlediska vlivu vzorkování se jedná o nejhorší případ, protože dominantní časová konstanta samotného motoru je kratší než u servopohonu spojeného s další setrvačnou hmotou.

Následující přechodové charakteristiky byly získány tak, že po nastavení parametrů regulátoru PSD v řídící jednotce servopohonu s podle výsledků algoritmu automatického seřízení regulátoru, byl pomocí vývojového prostředí pro programování DSP procesoru Metrowerks pozastaven program v řídící jednotce a tím přerušena regulační smyčka. Díky tomu bylo možné změnit ručně polohu hřídele servopohonu. Vzhledem k tomu že čítač inkrementálního snímače polohy je v řídící jednotce řešen nezávisle na běhu programu DSP procesoru, byla tato změna v registru DSP procesoru zaznamenána. Po následném spuštění programu byla reakce stejná jako na skok žádané polohy, přičemž se servo vrátilo do původní polohy. Data s přírůstky polohy natočení hřídele byla během regulace po

sériové lince odesílána do nadřízeného počítače. Tam byla uložena do souboru. Charakteristiky jsou vykresleny s použitím zachycených dat v souboru v prostředí Matlab. Tímto způsobem bylo možné řídící systém otestovat, aniž by bylo třeba dalších úprav software řídící jednotky.

Cílem tohoto experimentu bylo srovnáním se simulacemi ukázat vliv nelinearit a nepřesnosti modelu na dynamiku řízení. Chtěli jsme také ukázat, jaký význam má v tomto případě volba parametru fázové bezpečnosti při automatickém nastavování regulátoru. Na obrázcích 4.5, 4.6 a 4.7 proto vidíme záznam přechodové charakteristiky pro PSD regulátor navržený pro tři různé fázové bezpečnosti a pro dvě různé hodnoty parametru na obrázcích a) je to $\Delta\omega = 5$, a na obrázcích b) pak $\Delta\omega = 10$

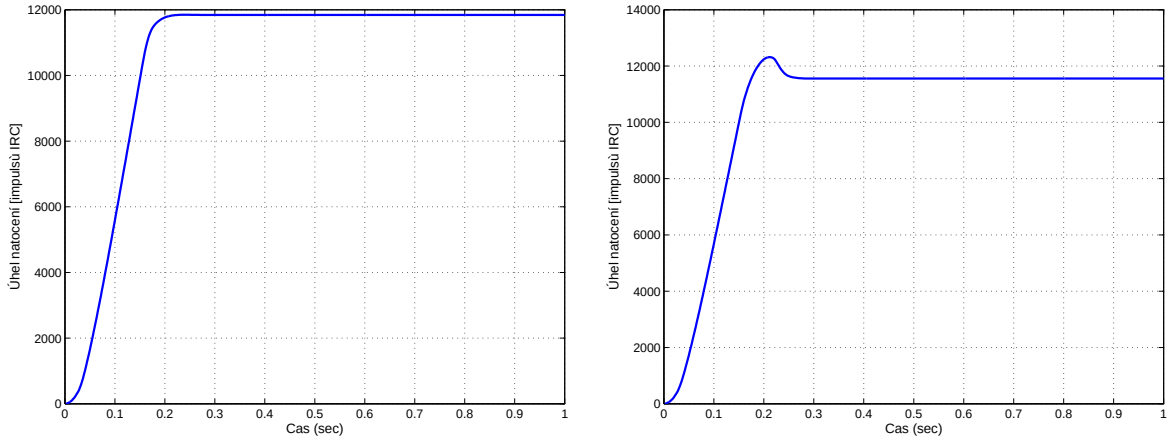


a) b)

Obrázek 4.5: Přechodová charakteristika servopohonu $\Delta\varphi = 45°$

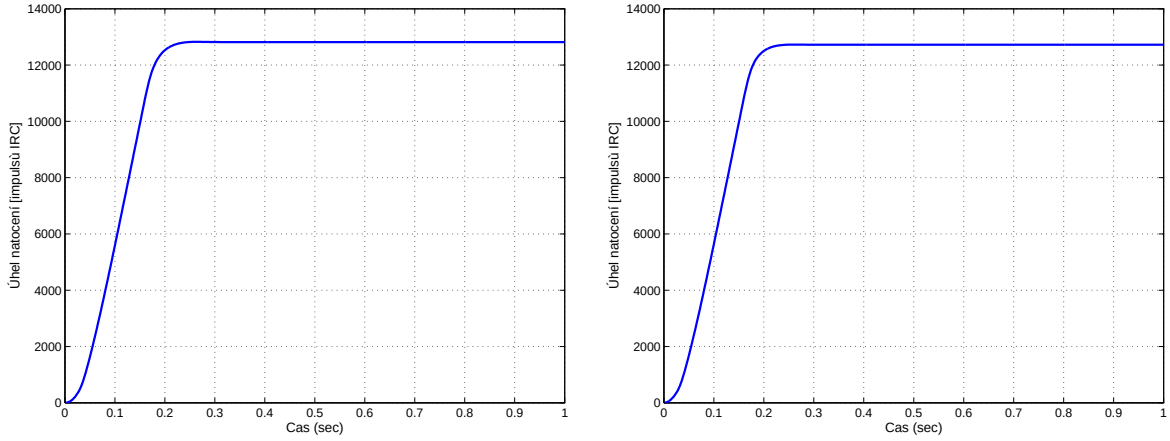
Oproti simulacím s lineárním modelem mají reálné přechodové charakteristiky o něco větší amplitudu překmitu. Kmity se ale z důvodu necitlivosti motoru kolem počátku způsobené třením rychle utlumí, takže ve výsledku jsou charakteristiky prakticky identické. Pokud ale zvýšíme požadovanou fázovou bezpečnost při syntéze dojde k zmenšení odchylek v přechodovém ději od lineárního modelu a regulace probíhá bez překmitu. Můžeme tedy díky použité frekvenční metodě syntézy snadno eliminovat nepřesnost identifikovaného modelu a nelinearity systému vhodnou volbou fázové bezpečnosti.

Z hlediska nasazení systému na řezacím stroji lze z provedených experimentů usuzovat, že výhodnější bude volba vyšší hodnoty fázové bezpečnosti, abychom odstranili překmit a současně vyšší podíl integrační složky pro odstranění necitlivosti pohonu pro malé hodnoty napětí, která je způsobena třením. Vhodným nastavením by mohlo být $\Delta\omega = 5$ a fázová bezpečnost v rozmezí $\Delta\varphi = 52°$ až $\Delta\varphi = 60°$. Je třeba si ale uvědomit,



a) b)

Obrázek 4.6: Přechodová charakteristika servopohonu $\Delta\varphi = 52°$



a) b)

Obrázek 4.7: Přechodová charakteristika servopohonu $\Delta\varphi = 60°$

že v reálném nasazení algoritmu bude servopohon pracovat ve výhodnějších podmínkách. Žádanou polohu bude generovat interpolátor, který generuje přírůstky žádané polohy řádově v jednotkách až desítkách milisekund. Skokové přírůstky polohy nabývají tedy hodnot jednotek pulsů. Při regulaci proto půjde zejména o přesné sledování změn rychlosti, které mohou být na vrcholech obrysu skokové. Skok žádané hodnoty s takovým přírůstkem, jaký byl použit v experimentech, by se v normálním provozu neměl vyskytnout.

Cílem této kapitoly bylo demonstrovat chování algoritmu v reálných podmínkách a ukázat přínos jeho použití. Bylo by tedy zbytečné vypisovat číselně všechny konstanty

nastavení regulátorů použité k testování algoritmu. V následující tabulce proto uvedeme pro příklad jen hodnoty vypočtené ze stejného modelu, jaký byl získán při testech, ale jen pro dvě vybrané hodnoty fázové bezpečnosti a pro případ nastavení parametru $\Delta\omega = 5$

-	K	TI	TD
$\Delta\varphi = 45°$	18.8	0.0886	0.0123
$\Delta\varphi = 60°$	10.01	0.128	0.0178

Tabulka 4.1: Příklad vypočtených konstant regulátoru

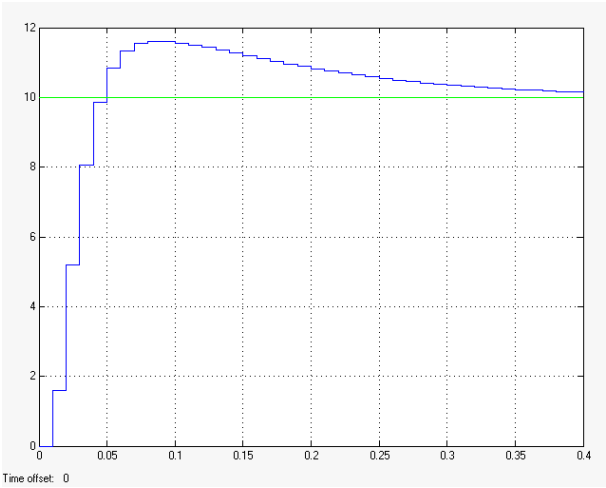
4.4 Perioda vzorkování

V této kapitole ověříme, jestli je zvolená perioda vzorkování $T_s = 0.002s$ pro daný servopohon dostatečně krátká a v jakém rozsahu periody T_s je automatické nastavení regulátoru funkční. Perioda T_s byla zvolena co nejkratší s ohledem na výpočetní výkon DSP procesoru. Pro ověření vyjdeme z hodnot regulátoru uvedených v tabulce 4.1. Pro tyto hodnoty můžeme vyčíslit nerovnost (2.16), odvozenou pro ověření ekvivalentího chování algoritmu PSD s PID regulátorem.

$$\frac{T_s}{TI} = 0.0225 < \frac{TD}{T_s} = 6.15$$

Vidíme, že nerovnost je splněna řádově, neznamená to ovšem, že by použitá vzorkovací perioda byla zbytečně krátká. Naopak experimentálním ověřením se ukazuje, že splnění této podmínky je pouze postačujícím kriteriem, a že pro funkčnost celého systému automatického seřízení regulátoru je třeba, aby hodnota $\frac{TD}{Ts}$ byla nejméně rovna $10 \cdot \frac{Ts}{TI}$, což v tomto případě odpovídá zhruba $Ts = 0.01s$. Pro kvalitní regulaci je ale použitá perioda vzorkování $T_s = 0.002s$ nutná.

Pro demonstraci vlivu pomalejšího vzorkování byla odsimulována přechodová charakteristika na obrázku 4.8. Pro její získání jsme vyšli ze stejného modelu jako v předcházejících případech. Ten byl ale převzorkován pro periodu $Ts = 0.01s$ a pro tento model s přidaným dopravním zpožděním provedena frekvenční syntéza. Simulace byla provedena s použitým PSD algoritmem, kde bylo ale třeba přepočítat konstanty na nové vzorkování a převzorkovaným modelem, bez dopravního zpoždění.



Obrázek 4.8: Přechodová charakeristika $Ts = 0.01s$

4.5 Časová náročnost programu

Pro přibližné ověření časové náročnosti algoritmu na různých platformách jsme provedli měření doby trvání výpočtu programu `ls_model` pro 1000 a 2000 vzorků identifikačního průběhu a programu `fr_PSD`. Program `fr_PSD` počítal frekvenční charakteristiku s dělením 512 bodů. Programy byly spuštěny s potlačeným výstupem na obrazovku přesměrováním standartního výstupu na zařízení `/dev/null`. Načtení vstupních a výstupních dat je v čase ale započítáno. Doba trvání výpočtu byla měřena pomocí programu `time`, který je součástí distribuce Linuxu. Měření jsme provedli pro počítač s prosecorem Pentium II 266 MHz a pamětí 64MB a počítač s procesorem AMD Athlon XP 2200+ 1,8 GHz, L2 Cache 256kB, pamět 512 MB. Výsledky jsou uvedeny v tabulce 4.2.

-	Pentium II	Athlon XP
`ls_model` 2000 vzorků	0.056 s	0.007 s
`ls_model` 1000 vzorků	0.032 s	0.004 s
`fr_PSD`	0.026 s	0.005 s

Tabulka 4.2: Časová náročnost programu

Z uvedených časů je patrné, že doba potřebná k výpočtu pro automatické seřízení regulátoru bude s jakýmkoliv vestavěným počítačem pro řízení systému s operačním systémem Linux zanedbatelná. Výsledný kód by bylo možné navíc ještě zefektivnit použitím knihoven ATLAS optimalizovaných pro cílovou platformu, případně použitím výpočtu frekvenční charakteristiky v menším rozlišení. Na dostatečně výkonném hardware s ope-

račním systémem pracujícím v reálném čase by pak rychlost syntézy, ve spojení s rekurzivní identifikací, mohla umožnit i samočinné nastavení PSD regulátoru.

Kapitola 5

Závěr

V této práci jsme se zabývali návrhem a implementací algoritmu pro automatické seřízení regulátoru servopohonů řezacího stroje. Seznámili jsme se se současnými problémy řízení pohybu hořáku a se strukturou stávajícího řídícího systému řezacího stroje. V úvodní kapitole jsme se také pokusili o shrnutí základních přístupů a hlavních problémů adaptivní regulace. Cílem tohoto shrnutí problematiky bylo navrhnout algoritmus vhodný pro efektivní implementaci v jazyce C, který by umožnil zlepšení kvality regulace servopohonů řezacího stroje a zaručil jeho automatické seřízení s přiměřenými nároky na výpočetní výkon řídícího systému a s přihlédnutím ke struktuře řídícího systému řezacího stroje.

Optimálním řešením se ukázalo použití PSD regulátoru doplněného sytémem automatického seřízení parametrů (ATC) s obdobnou strukturou jakou využívají STC regulátory. Seřízení parametrů je prováděno na základě identifikace parametrů modelu metodou nejmenších čtverců pomocí jednorázového měření dat a následnou syntézou frekvenční metodou. Teoretická část práce je věnována popisu těchto metod zaměřenému zejména na jejich aplikaci v tomto praktickém případě. Nevyhnuli jsme se proto rozvaze o optimálním řádu modelu a vhodném vzorkováním, aby bylo možné řešení přizpůsobit pro konkrétní pohony používané na řezacím stroji.

Teoretickým zvládnutím problematiky a otestováním její funkčnosti v prostředí Matlab ale práce nekončí. Naopak její hlavní částí je implementace celého algoritmu v jazyce C umožnujícím program zkompilovat pro různé modifikace řídícího sytému, ať už pro popsanou variantu s málo výkonnou jednotkou nadřízeného počítače, tak pro novější řešení využívající architektury PC a speciálních základních desek miniATX vhodných pro nasazení v průmyslovém provozu. Společným rysem řídících systémů je využití operačního systému Linux, který se ukázal být zejména díky možnosti využití matematických knihoven GSL výborným řešením. Dosažená rychlost a přesnost výpočtu byla naprosto

dostatečná i na počítači PC s procesorem Pentium II, který byl jako nejpomalejší pro otestování k dispozici. Systém automatického seřízení bylo ale také třeba doplnit vhodným algoritmem PSD regulátoru a metodou snímání dat pro identifikaci. Tyto činnosti zajišťuje řídící jednotka s DSP procesorem, kde je výpočetní a paměťová náročnost ještě větším omezením. Požadavkem je výpočet v reálném čase s krátkou vzorkovací periodou, proto bylo nezbytné použití vhodného algoritmu PSD regulátoru s ošetřením windupu a s výpočtem v přesnosti integer, aby byla doba potřebná pro výpočet akčního zásahu dostatečně krátká. Rovněž snímání dat pro identifikaci bylo třeba zajistit součinností řídící jednotky servopohonů a nadřízeného počítače. Popisu této implementační problematiky byla věnována hlavní část práce.

V poslední kapitole jsou soustředěny poznatky získané simulacemi a měřením na reálném systému. Aby bylo navržené řešení skutečně použitelné v praxi, bylo třeba ukázat vliv dvou základních parametrů syntézy, kterými jsou fázová bezpečnost $\Delta\varphi$ a poměr integrační a derivační složky označovaný $\Delta\omega$, na regulaci reálného systému s nepřesnostmi a nelinearitami. Na měření s pohonem se ukázalo, že tyto dva parametry umožnují snadno nastavit kompromis mezi rychlostí a robustností regulace vůči nelinearitám a chybám modelu a že jejich nastavením nebude v praxi problém dosáhnout optimálního chování servosmyčky. Také shoda přechodových charakteristik systému reálného a teoretického lineárního se ukázala vyhovující už pro hodnotu $\Delta\varphi = 45°$.

Další měření na reálných datech se týkala ověření řádu modelu a vzorkovací periody a je na ně odkazováno už z teoretické části, kde bylo třeba vhodné hodnoty zvolit. Byla zahrnuta pro přehlednost do této, kapitoly protože také ilustrují meze funkčnosti navrženého řešení. Vyhodnocení parametru χ^2, které je využito ke stanovení řádu modelu, by bylo i vhodnou metodou, kterou by bylo možné provádět diagnostiku opotřebení mechaniky stavu stroje. Určuje totiž úspešnost, s jakou se shodují data s teoretickým lineárním modelem. Dlouhodobým sledováním parametru χ^2 při stále stejném identifikačním signálu by mohlo být detekováno zvetšení vůlí nebo tření v mechanismu projevujících se nelineárním chováním. Pro obtížnou dostupnost řezacího stroje nebylo ale možné takové experimenty do této práce zahrnout.

Výhodou implementované metody oproti obvykle používaným jednodušším postupům automatického seřízení PID regulátoru, založených na heuristických metodách (Prokop a Korbel, 2006) nebo řada příkladů v (Leva et al., 2003), často využívajících k identifikaci relé ve zpětné vazbě je, že vychází z přesnějšího modelu, který může být teoreticky libovolného řádu. Vhodnými parametry identifikačního experimentu lze odstranit případný nežádoucí vliv nelinearit, kterými jsou v našem případě tření v okolí počátku rozběhu

pohonu a proudové omezení napájecího zdroje při velkých skocích napětí. Implementovaný systém automatického seřízení tak lze použít nejen pro servopohony ale i pro širokou škálu systémů včetně systémů s dopravním zpožděním.

Vzhledem k jeho efektivní implementaci je možné jej použít na řídících systémech s distribuovaným řízením. Implementovaná metoda neklade velké nároky na nadřízený počítač ani z hlediska výpočetního výkonu, ani nevyžaduje zpracování v reálném čase.

Naopak vzhledem k tomu, že struktura řešení vychází z struktury STC regulátorů, bylo by možné s použitím dostatečně výkonného počítače s operačním systémem pracujícím v reálném čase, ověřit využití implementované frekvenční metody, ve spojení s rekurzivní identifikací metodou nejmenších čtverců, pro implementaci samočinně se nastavujícího PSD regulátoru.

Literatura

Anderson, E. et al. (1999), *LAPACK Users' Guide, Third Edition*, SIAM. <http://www.netlib.org/lapack/lug/index.html>.

Bobál, V. et al. (1999), *Praktické aspekty samočinně se nastavujících regulátorů: algoritmy a implementace*, Brno: VUTIUM.

Fujinaka, T. et al. (2000), Stabilization of double inverted pendulum with self-tuning neuro-pid, *in* 'IEEE International Joint Conference on Neural Networks'.

Galassi, M. et al. (2006), *GNU Scientific Library Reference Manual*, second edn.

Halmo, L. (2004), Numerické algoritmy pro polynomiální matice v jazyce c++, Master's thesis, ČVUT FEL.

Havlena, V. a Štecha, J. (2000), *Moderní teorie řízení*, Praha: Vydavatelství ČVUT.

Horáček, P. (2001), *Systémy a modely*, Praha: Vydavatelství ČVUT.

John, J. (2003), *Systémy a řízení*, Praha: Vydavatelství ČVUT.

John, J. (2006), *Systémy - elektronická učební pomůcka [on-line]*. Poslední revize 5.1.2007 <http://dce.felk.cvut.cz/sri2/ss/>.

Leva, A., Cox, C. a Ruano, A. (2003), 'Hands-on pid autotuning: a guide to better utilisation [on-line]', *IFAC Professional Briefs* . Poslední revize 5.1.2007 <http://www.ifac-control.org/publications/pbriefs>.

Neuhauser, J. (2004), Adaptivní řízení elektrohydraulických servomechanismů, Master's thesis, ČVUT FEL.

Pivoňka, P. (2003), Vyšší formy řízení, Technical report, VUT Brno. < www.fme.vutbr.cz/opory/pdf/uai/vyssi_formy_rizeni/VFR03.pdf>.

Prokop, R. a Korbel, J. (2006), 'Relé ve zpětné vazbě aneb převrat v návrhu regulátorů', *Automatizace* **49**(3), 190–195.

Sundareswaran, K. a Begum, S. R. (2004), 'Genetic tuning of a power system stabilizer', *European Transactions On Electrical Power* **35**(14), 151–160. <http://www.cs.utsa.edu/~whaley/papers/spercw04.ps>.

Trnka, P. (2006), 'Optimální rozhodování a řízení, úloha na cvičení'. <http://dce.felk.cvut.cz/orr>.

Whaley, R. C. a Petitet, A. (2005), 'Minimizing development and maintenance costs in supporting persistently optimized BLAS', *Software: Practice and Experience* **35**(2), 101–121. <http://www.cs.utsa.edu/~whaley/papers/spercw04.ps>.

Štecha, J. (2004), *Optimální rozhodování řízení*, Praha: Vydavatelství ČVUT.

Příloha A

Algebraicko numerický přístup k řešení

V rámci práce byla ještě alternativně vyzkoušena pro syntézu regulátoru algebraická metoda výpočtu stabilní oblasti s numerickou optimalizační metodou k nalezení parametrů regulátoru. Přestože využití parametrizace regulátoru algebraickými metodami je lákavé svojí exaktností a možností využití různých kriterií optimality, ukázal se zvolený postup v praktickém případě příliš komplikovaný. Proto je algebraický návrh regulátoru popsán jen pro případ P regulátoru s využitím matematických nástrojů Matlab a Maple. Výsledky jsou nicméně zajímavé, a jsou proto uvedeny pro srovnání s frekvenční metodou v této příloze.

Algoritmus návrhu regulátoru vychází ze stejného modelu systému jako frekvenční metoda návrhu. Z identifikovaného modelu je vypočten přenos uzavřené smyčky s regulátorem, v tomto případě reprezentovaným zesílením k. Pomocí obecně sestavené Juryho tabulky pro daný řád modelu získáme vztahy pro výpočet stabilní oblasti uzavřené smyčky. Jedná se o soustavu polynomiálních nerovnic. Řešením je v případě P regulátoru interval zesílení. Pro případ PSD regulátoru by se jednalo o oblasti v prostoru $\mathcal{R}^3$. Pro stabilní oblast (v našem případě interval), je nalezeno optimum. V tomto jednoduchém příkladě bylo použito minimalizace lineární regulační plochy Fibonacciho metodou.

A.1 Podrobnější popis metody

A.1.1 Určení stabilní oblasti zesílení P regulátoru

Pro určení oblasti stability lze využít Juryho test, který je popsán podrobně v (Horáček, 2001). Vyjádříme charakteristický polynom

$$A + kB = z^3 + a_2 z^2 + (a_1 + k \cdot b_1)z + (a_0 + k \cdot b_0) \ . \tag{A.1}$$

Pro obecný charakteristický polynom ve tvaru $\tilde{a}_0\, z^n + \tilde{a}_1\, z^{n-1} + \cdots + \tilde{a}_n$ sestavíme Juryho tabulku následujícím způsobem:

$\tilde{a}_0$	$\tilde{a}_1$	$\cdots$	$\tilde{a}_{n-1}$	$\tilde{a}_n$	
$\tilde{a}_n$	$\tilde{a}_{n-1}$	$\cdots$	$\tilde{a}_1$	$\tilde{a}_0$	$\alpha_n = \frac{\tilde{a}_n}{\tilde{a}_0}$
$\tilde{a}_0^{n-1}$	$\tilde{a}_1^{n-1}$	$\cdots$	$\tilde{a}_{n-1}^{n-1}$		
$\tilde{a}_{n-1}^{n-1}$	$\tilde{a}_{n-1}^{n-1}$	$\cdots$	$\tilde{a}_{n-1}^{n-1}$		$\alpha_{n-1} = \frac{\tilde{a}_{n-1}^{n-1}}{\tilde{a}_0^{n-1}}$
$\vdots$					
$\tilde{a}_0^0$					

Tabulka A.1: Juryho tabulka

kde

$$\tilde{a}_i^{k-1} = \tilde{a}_i^k - \alpha_k \tilde{a}_{k-i}^k \ , \tag{A.2}$$

$$\alpha_k = \frac{\tilde{a}_k^k}{\tilde{a}_0^k} \ . \tag{A.3}$$

Juryho tabulka byla pro obecné parametry charakteristického polynomu sestavena v Maple a není zde z důvodu rozsahu celá uvedena – viz program `jury.mw`. Charakteristický polynom je stabilní, tedy má póly uvnitř jednotkové kružnice, pokud je splněna následující podmínka:

Nechť, $a_0 > 0$ potom platí $a_0^k > 0 \ \ \forall k = 0, 1, \cdots, n-1$.
V našem případě budou první tři řádky tabulky

$$\begin{bmatrix} 1 & a2 & a1 + kb1 & a0 + kb0 \\ a0 + kb0 & a1 + kb1 & a2 & 1 \\ 1 - (a0 + kb0)^2 & a2 - (a1 + kb1)\,(a0 + kb0) & a1 + kb1 - a2\ (a0 + kb0) & 0 \\ \vdots & \vdots & \vdots & \vdots \end{bmatrix} .$$

Další řádky jsou samozřejmě stále složitější, takže je nelze dále vypsat, ale výpočetně není problém, do jednou odvozeného obecného zápisu Juryho tabulky opakovaně dosadit parametry získané identifikací.

Po dosazení koeficientů jsme získali v Maple pro určení stability soustavu tří algebraických nerovnic, které tvoří prvky $\tilde{a}_0^0, \tilde{a}_0^1, \tilde{a}_0^2$ Juryho tabulky. Kořeny jejich čitatelů představují hranice intervalů. Po dosazení stejných parametrů, s jakými byly uskutečněny experimenty v kapitole 4 získáme tuto soustavu

$$\begin{aligned} 0 &< 0.9995598690 - 0.000007153521714\,k - 0.0000000290668401\,k^2 \\ 0 &< -1464157712000000.0 + 20714935260000.0\,k + 12988278500.0\,k^2 - \\ &\quad -4158.60544\,k^3 - 8.448811934\,k^4 \\ 0 &< 50167960640.0\,k^6 + 2252664289000000.0\,k^5 + 1.047480514 \cdot 10^{19}\,k^4 - \\ &\quad -7.497398312 \cdot 10^{22}\,k^3 - 4.007741394 \cdot 10^{26}\,k^2 + 2.751123921 \cdot 10^{28}\,k + \\ &\quad +1.519447443 \cdot 10^{26}\ . \end{aligned} \tag{A.4}$$

Řešením jednotlivých nerovnic jsou následující intervaly

$$(-5989, 5742)$$

$$(-38635, -5989) \cup (-1665, 67.8) \cup (5742, 39740)$$

$$(-\infty, -38635) \cup (-6088, -5989) \cup (-5989, -1665) \cup (-0.552 \cdot 10^{-2}, 67.8) \cup \\ \cup (5742, 5742) \cup (39740, \infty)\ . \tag{A.5}$$

Současné splnění všech tří podmínek získáme průnikem intervalů platnosti jednotlivých rovnic. V tomto případě je stabilní oblast v intervalu $(-0.552 \cdot 10^{-2}, 67.7992)$. Za předpokladu ideálního lineárního astatického sytému by spodní hranice měla být 0. Vypočtená hodnota se nule blíží, což svědčí o správnosti modelu.

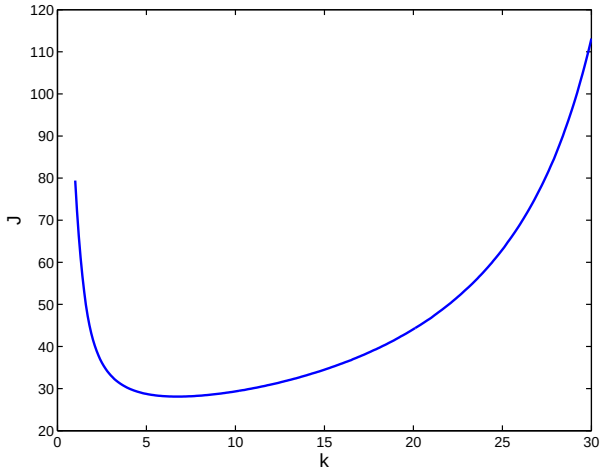
Tím že jsme určili stabilní oblast přenosu uzavřené smyčky, velmi jsme si ulehčili nalezení optimálních parametrů nějakou numerickou metodou. V další kapitole ukážeme velmi jednoduchý příklad algoritmu, kterým by bylo možné vhodný regulátor najít.

A.1.2 Minimalizace lineární regulační plochy Fibonacciho metodou

Pro výběr optimálního zesílení ze stabilního intervalu, lze využít, v případě numerické metody optimalizace, minimalizaci obecného integrálního kriteria ve tvaru

$$J = \int_0^T f(g(t), t)dt \ .$$

Pro návrh zesílení zpětné vazby servomechanismu je například možné použít kriterium lineární regulační plochy $J = \int_0^\infty |e(t) - e(\infty)|dt$, kde $e(t)$ je regulační odchylka při jednotkovém skoku řízení. Průběh kriteria pro dané parametry modelu je vidět na orázku A.1. Minimum kriteria leží v intervalu$(6.8187, 6.8249)$ Pro určení optimálního

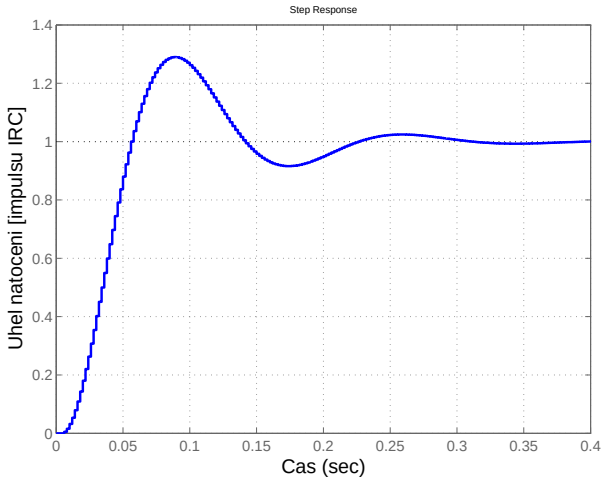


Obrázek A.1: Průběh kriteria J v okolí optimální hodnoty

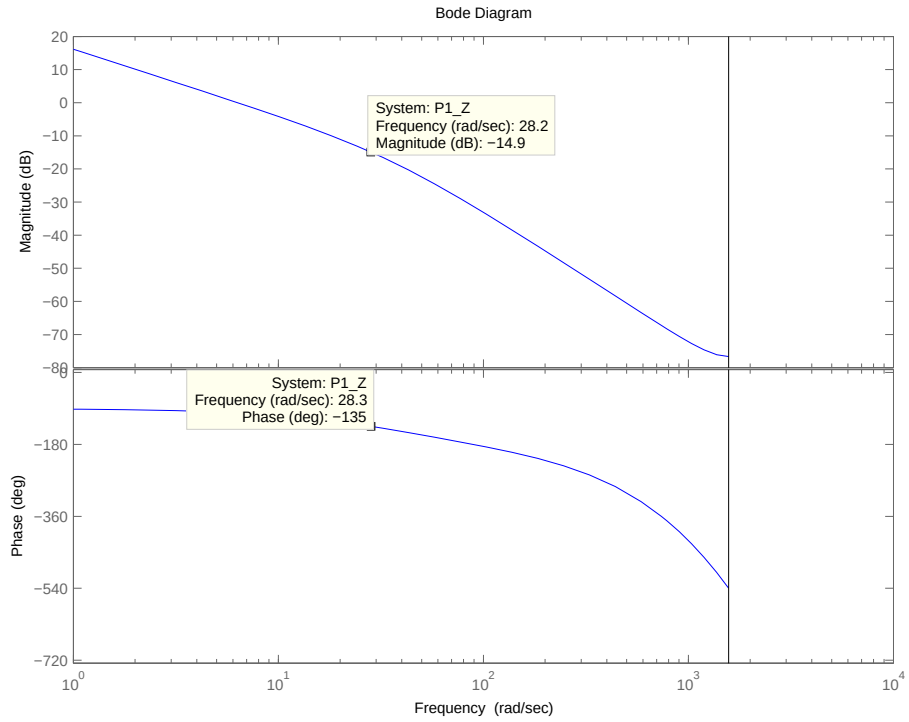
zesílení s touto přesností stačilo 20 iterací Fibonacciho metody (Štecha, 2004). Na obrázku A.2 je vidět přechodová charakteristika uzavřené smyčky pro minimum lineární regulační plochy, při hodnotě $k = 6.822$. Výsledek syntézy podle minima regulační plochy můžeme porovnat s návrhem P regulátoru frekvenční metodou znázorněném na obrázku A.3. Zesílení regulátoru nám v případě požadované bezpečnosti $\Delta\varphi = 45°$ vyjde přibližně $10^{14.9/20} = 5.6$. Přechodovou charakteristiku pro P regulátor s touto fázovou bezpečností můžeme najít na obrázku 4.3. Z frekvenční charakteristiky systému můžeme také obráceně určit že minima kriteria dosahujeme při fázové bezpečnosti 40°.

A.2 Závěr

Na jednoduchém příkladě jsme ukázali možnost alternativního přístupu k syntéze. Její praktické využití by bylo ale problematické. Parametrizace stabilní oblasti pro daný typ regulátoru nám sice poskytuje důležitou informaci o systému, která značně usnadňje



Obrázek A.2: Přechodová charakteristika pro minimum kriteria J



Obrázek A.3: Návrh P regulátoru frekvenční metodou

následnou optimalizaci, nicméně s rostoucí složitostí regulátoru se parametrizace výrazně komplikuje. Otázkou praktické realizace by byla ale také volba vhodného kriteria, neboť jediné kriterium těžko bude vyjadřovat veškeré požadavky na kvalitu regulace. Navíc v závislosti od zvolené numerické metody by bylo třeba zajistit další požadavky na zvolené kriterium jako je neexistence lokálních extrémů monotónost a podobně. Na druhou stranu

použitím vhodné gradientí metody bychom mohli zjednodušit parametrizaci nalezením nějaké menší stabilní podoblasti z které by bylo možné začít první iteraci algoritmu.