

Web-aplikace pro analýzu LTI systémů

Bc. Jiří Facuna

Diplomová práce
2014



Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky
akademický rok: 2013/2014

ZADÁNÍ DIPLOMOVÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení: **Bc. Jiří Facuna**
Osobní číslo: **A12399**
Studijní program: **N3902 Inženýrská informatika**
Studijní obor: **Automatické řízení a informatika**
Forma studia: **prezenční**

Téma práce: **Web-aplikace pro analýzu LTI systémů**
Téma anglicky: **A Web-application for LTI Systems Analysis**

Zásady pro vypracování:

1. Vypracujte literární rešerši na dané téma.
2. Navrhněte a realizujte webovou aplikaci, která umožní provést analýzu uživatelem zadaného lineárního časově invariantního systému.
3. Aplikaci koncipujte jako dvojjazyčnou (CZ/ENG) a pro její vytvoření využijte platformu Microsoft .NET spolu s programovým systémem MATLAB.
4. Umožněte zadání jak spojitého, tak i diskrétního jednorozměrového modelu v přenosovém nebo stavovém vyjádření.
5. Vytvořenou aplikaci důkladně otestujte a porovnejte získané výsledky s jiným přístupem.
6. Věnujte dostatečnou pozornost zabezpečení aplikace.

Rozsah diplomové práce:

Rozsah příloh:

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

Seznam odborné literatury:

1. **DOSTÁL, Petr a František GAZDOŠ. Řízení technologických procesů. Zlín: Univerzita Tomáše Bati ve Zlíně. Fakulta aplikované informatiky. 2006. ISBN 80-7318-465-6.**
2. **BALÁTĚ, Jaroslav. Automatické řízení. Praha: BEN – technická literatura, 2004. ISBN 80-7300-148-9.**
3. **RAKUS, David. Web-aplikace pro přímý návrh a ladění regulátorů z experimentálních dat. Zlín, 2009. Diplomová práce. Univerzita Tomáše Bati ve Zlíně, Fakulta aplikované informatiky.**
4. **KARBAN, Pavel. Výpočty a simulace v programech Matlab a Simulink. Praha: BEN-technická literatura, 2007. ISBN 978-80-251-1448-3.**
5. **THE MATHWORKS, Inc. Control System Toolbox: User's Guide. Natick, MA, USA, 2013.**
6. **MICROSOFT. Vytváříme zabezpečené aplikace v Microsoft ASP.NET. Praha: Computer Press, 2004. ISBN 80-251-0466-4.**
7. **ZEMEK, Lukáš. Bezpečnost webových aplikací. Praha, 2012. Bakalářská práce. Univerzita Tomáše Bati ve Zlíně, Fakulta aplikované informatiky.**

Vedoucí diplomové práce:

doc. Ing. František Gazdoš, Ph.D.

Ústav řízení procesů

Datum zadání diplomové práce:

7. března 2014

Termín odevzdání diplomové práce:

11. června 2014

Ve Zlíně dne 7. března 2014

prof. Ing. Vladimír Vašek, CSc.
děkan



prof. Ing. Vladimír Vašek, CSc.
ředitel ústavu

Prohlašuji, že

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

Prohlašuji,

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

Ve Zlíně

.....
podpis diplomanta

ABSTRAKT

Diplomová práce se zabývá využitím webové aplikace spolu s programovým prostředím MATLAB pro studentské účely.

Teoretická část této práce popisuje použité programovací jazyky pro vytvořenou webovou aplikaci, prostředí MATLAB a jejich propojení mezi nimi. Dále je zde zmíněna kapitola, která nás zavádí do oblasti lineárních časově invariantních systémů. Tato kapitola obsahuje různé druhy systémů, vlastností a charakteristik, které se používají ve výsledné aplikaci.

V praktické části práce je popsána struktura vytvořené webové aplikace. Dále i obsahuje popis jednotlivých funkcí aplikace, ze kterých je tvořena webová aplikace pro analýzu lineárních časově invariantních systémů.

Klíčová slova:

MATLAB, Microsoft .NET Framework, ASP.NET, C#, Lineární časově invariantní systém, web-aplikace, analýza

ABSTRACT

This graduation thesis deals with the usage of web application together with the programming / simulation environment MATLAB for student.

The theoretical part of this thesis describes the programming languages used to create the Web application, the MATLAB environment and the connections between them. There is also a chapter, which introduces us to the field of linear time – invariant systems. This chapter contains a variety of systems, features and characteristics that are used in the final application.

The practical part describes the structure of the created web application. Furthermore, contains a description of the individual application functions used for the development of the web application for analysis of linear time – invariant systems.

Keywords:

MATLAB, Microsoft .NET Framework, ASP.NET, C# Linear time - invariant system, web-application, analysis

Děkuji vedoucímu diplomové práce panu doc. Ing. Františku Gazdošovi, Ph.D. za čas, který mi věnoval, odborné vedení a pomoc při zpracování této diplomové práce.

OBSAH

ÚVOD	9
I TEORETICKÁ ČÁST	11
1 LITERÁRNÍ REŠERŠE	12
1.1 ZÁKLADNÍ LITERATURA	12
1.2 APLIKACE PRO ANALÝZU SYSTÉMŮ	12
2 WEBOVÉ APLIKACE	14
2.1 HTML.....	14
2.2 CSS.....	14
2.3 AJAX.....	15
2.4 ASP.NET.....	15
2.5 PROGRAMOVACÍ JAZYK C#	15
2.5.1 Visual Web Developer Express	16
3 PROSTŘEDÍ MATLAB	17
3.1 MATLAB DEPLOYMENT TOOL	17
4 LINEÁRNÍ ČASOVĚ INVARIANTNÍ SYSTÉMY	19
4.1 VNITŘNÍ A VNĚJŠÍ POPIS	19
4.2 DRUHY SYSTÉMŮ Z HLEDISKA SPOJITOSTI V ČASE	21
4.3 ZÁKLADNÍ VLASTNOSTI SYSTÉMU	22
4.3.1 Póly a nuly.....	22
4.3.2 Řád	23
4.3.3 Zesílení	23
4.3.4 Stabilita	23
4.3.5 Řiditelnost a dosažitelnost stavu	24
4.3.6 Řiditelnost (dosažitelnost) výstupu	25
4.3.7 Pozorovatelnost a rekonstruovatelnou stavu	26
4.4 DOPRAVNÍ ZPOŽDĚNÍ.....	28
4.5 MNOHOROZMĚRNÉ SYSTÉMY	29
4.6 ZÁKLADNÍ GRAFICKÉ CHARAKTERISTIKY	29
4.6.1 Přechodová charakteristika	29
4.6.2 Impulsní charakteristika	31
4.6.3 Frekvenční charakteristika v komplexní rovině.....	33
4.6.4 Amplitudově - fázová frekvenční charakteristika v log. souřadnicích	35
4.6.5 Rozložení nul a pólů v komplexní rovině	36
II PRAKTICKÁ ČÁST	37
5 STRUKTURA APLIKACE	38
5.1 MATLAB SOUBORY	38
5.2 SOUBORY .NET.....	40
6 ZABEZPEČENÍ APLIKACE	42
7 MODULY WEBOVÉ APLIKACE	44

7.1	ÚVODNÍ STRANA	44
7.2	POPIS SYSTÉMU	45
7.3	ZADÁVÁNÍ SYSTÉMU	46
7.4	ZKOUŠKA	47
7.5	PŘEVZORKOVÁNÍ SYSTÉMU	48
7.6	PŘEVODY SYSTÉMU	49
7.7	ZADÁVÁNÍ VLASTNOSTÍ	50
7.8	VOLBA GRAFICKÝCH CHARAKTERISTIK	51
7.9	VÝPIS	52
	ZÁVĚR	55
	SEZNAM POUŽITÉ LITERATURY.....	57
	SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK.....	59
	SEZNAM OBRÁZKŮ	60
	SEZNAM PŘÍLOH.....	62

ÚVOD

Postupným vývojem informačních systémů se neustále rozvíjí web a webové aplikace. Každým rokem jsou tvořeny nové systémy, nové vylepšení programovacích jazyků a zvyšuje se rychlost a dostupnost Internetu. Internet se začíná zaplavovat nesčítelným množstvím webových aplikací, které jsou dostupné pro všechny a zadarmo. Nejhlavnějšími důvody je bezplatná dostupnost aplikace, rychlost a jejich šíření. Přístup k Internetu nebyl dříve tak jednoduchý a všudypřítomný jako v dnešní době.

Aktuální vědecko – technická programová prostředí např. MATLAB, jsou vybavena velkým počtem knihoven. I zde se každým rokem vytváří nové funkce a toto programové prostředí se posouvá dál a dál do dalších vědních oborů. Takové systémy jsou určené nejen pro matematiky, fyziky a inženýry, ale při rozšíření knihoven lze využít MATLAB i pro programování aplikací, finanční analýzu, zpracování obrazu a mnohem více. Kvůli možné simulaci různých systémů a rychlých výpočtů se díky spolupráci s univerzitami využívá MATLAB i pro studentské účely.

Hlavním nevýhodou programu MATLAB je jeho rozšíření a to pouze pomocí placené licence. Pro studenty a ty, kteří nevyužijí plný potenciál tohoto prostředí, není už tak atraktivní jako jiné programy určené k jejich specifické potřebě. Tato nevýhoda dala podnět ke vzniku této diplomové práce. Díky přístupu k Internetu mohou studenti využít funkce z programu MATLAB k analýze lineárních časově invariantních systémů.

V první části této diplomové práce jsou popsány programové prostředky, které jsou využity k naprogramování aplikace. Dále je zde krátký popis prostředí MATLABu, kde se využívají výpočetní operace pro aplikaci. Poslední kapitola je věnovaná teoretickému popisu lineárních časově invariantních systémů. Kapitola se zabývá popisem systémů, jejich druhy a základními grafickými charakteristikami systémů. Tato část je nutná pro práci s webovou aplikací.

Cílem diplomové práce bylo vytvořit webovou aplikaci pro analýzu lineárních časově invariantních systémů a to zejména pro pedagogické účely. Praktická část popisuje vytvořenou webovou aplikaci pomocí zvolených programových prostředků společně s MATLABem. Popisuje se zde vytvoření potřebných funkcí v MATLABu a jeho následné převedení pomocí toolboxu Builder NE do souboru, který lze využít k naprogramování webové aplikace. Dále je v práci obsažen krátký popis zabezpečení aplikace. Poslední kapitolkou této diplomové práce je seznámení s aplikací a jeho možné využití pro uživatele

a to zejména pro studentské účely. Uživatelé se mohou vzdáleně připojit na tuto webovou aplikaci a využít ji pro analýzu lineárních časově invariantních systémů.

Tato diplomová práce je jedna z mnoha nástaveb diplomové práce Ing. Davida Rakuse [15].

I. TEORETICKÁ ČÁST

1 LITERÁRNÍ REŠERŠE

Literární rešerše obsahuje knihy, skripta, bakalářské nebo diplomové práce, které popisují základy analýzy pro lineární časově invariantní systémy. Je zde i zmínka o matematických programech a webové aplikaci, které dokážou pracovat s těmito systémy.

1.1 Základní literatura

Jedna z nejznámějších českých knih o automatizaci je [2]. Kniha popisuje celou škálu teorie o automatizaci, popisuje spojité, diskrétní, lineární, nelineární systémy. V knize lze také najít systémy s vnějším nebo vnitřním popisem a mnoho další. Tato kniha je základním zdrojem této diplomové práce a mnoha dalších.

Skripta psaná pro studenty od prof. Ing. Petra Dostála, Csc. a doc. Ing. Františka Gazdoše, Ph.D. [3] jsou základním popisem jednorozměrných dynamických systémů, kde se zabývají jak vnitřním, tak i vnějším popisem systémů. Velká část je zde věnována řízením v uzavřeném obvodu.

Diplomová práce na téma Stavová teorie lineárního řízení od kolegyně Matelové [9] je zaměřena na stavovou algebru. Popisuje zde více druhů stavových popisů, jednorozměrné a mnohorozměrné systémy a jejich vlastnosti. Obsahuje i popis stavového řízení s diskrétním regulátorem typu Dead – beat.

1.2 Aplikace pro analýzu systémů

Webová aplikace Analýza systémů od kolegy Crlíka [1] je naprogramovaná aplikace pro podporu předmětu Základy automatizace. Pro zadávání parametrů systému je více možností a vykresluje 4 charakteristiky zadaného systému. Při spuštění aplikace z různých zařízení je zde možnost přizpůsobit rozlišení.

Matematické programové prostředí MATLAB je ideální program pro různé analýzy, identifikaci, návrhy, řízení, zjišťování vlastností, grafických charakteristik systémů a dalších užitečných věcí. Tento program obsahuje i různá rozšíření jako je Simulink pro grafické návrhy simulací a experimentování s různými systémy. Další rozšíření je např. pro propojení webových aplikací jak pro programovací jazyk C#, tak i pro Javu. Lze si vytvořit program čistě v textové formě anebo si naprogramovat uživatelsky přívětivější prostředí s použitím GUI. Za program si ale je třeba zaplatit licenci a tudíž je třeba si rozmyslet, zda tento program bude dostatečně využit v potřebné míře.

Jako alternativu za placené programy se dá využít program Scilab, který je volně šiřitelný. Je to matematické programové prostředí, které taky obsahuje rozšíření jako je práce s vyššími programovacími jazyky C nebo C++. Dokáže pracovat jak s maticemi a různými funkcemi, tak obsahuje i grafický editor pro tvorbu 2D a 3D grafů. Pro více informací o programu Scilab doporučuji bakalářskou práci na téma Elektronický manuál pro program Scilab od kolegyně Vaculíkové [17].

Další možnou alternativou podobně zaměřeného, volně šiřitelného programu je Octave. Je to obdoba programu MATLAB, který taky pracuje s maticemi, funkcemi, grafickým prostředím nebo cykly a podmínkami. Nemá tolik rozšiřujících modulů jako MATLAB, ale dokáže si poradit s návrhem regulátorů a i s grafickým vykreslováním požadovaných simulací. Pro více informací a zjištění různých funkcí Octave doporučuji diplomovou práci Realizace jednorozměrných regulátorů v Octave od kolegy Halbsguta [5].

2 WEBOVÉ APLIKACE

S nástupem počítačů začal být kladen nárok na software a jeho využití. Naprogramované aplikace se rozdělují do několika skupin, jako např. freeware (bezplatná aplikace), shareware (aplikace chráněná autorským právem), komerční (aplikace za poplatek), atd.

Komerční programy jsou placené aplikace (MATLAB), kde se většinou zaplatí za licenci, někdy i časově omezenou. S možností internetu vznikly tzv. webové aplikace, které lze využít pomocí webového serveru a sítě Internet. Většina naprogramovaných webových aplikací je bezplatná pro všechny uživatele, bez nutnosti instalace. Mohou obsahovat výpočty ze speciálních placených programů (MATLAB). Používají se např. pro internetové obchody, online aukce, diskusní fóra, speciální výpočty a jiné. Webové aplikace se dají vytvořit přímo v programovacím jazyku jako je PHP, ale existují i řada systémů tzv. frameworky. Tyto systémy nabízejí programování na vyšších úrovních a tím i snížení počtu chyb. [15]

Existuje mnoho programovacích jazyků pro psaní webových aplikací, v dalších bodech budou popsány pouze ty, které jsou použity ve výsledné webové aplikaci.

2.1 HTML

Pro popis webových stránek se využívá jazyk HTML (HyperText Markup Language). Byl vytvořen pro publikaci dokumentů na Internetu. Správnou velikostí a barvu písma dosáhneme použitím určitých příkazů do textu. Internetové prohlížeče převedou příkazy jazyka HTML do grafické podoby. Jelikož existuje více prohlížečů a každý z nich musí umět rozlišit formátování textu, je kladen důraz na logickou strukturu. Pokud chceme vytvořit stránku s dynamickou strukturou, musíme HTML spojit s dalšími webovými technologiemi. [14]

2.2 CSS

Nadstavba jazyka HTML je jazyk kaskádové styly (CSS – Cascading Style Sheets). Je určen k definování vzhledu webových stránek. Důvodem vytvoření tohoto jazyka bylo oddělit vzhled stránky od jeho obsahu. Lze tak rychle a pohodlně měnit vzhled bez většího zásahu do kódu. Vytvořená webová aplikace je jedna z aplikací serveru MATSERVER a kvůli zachování designu jsem použil CSS styl od pana Ing. Davida Rakuse. [15]

2.3 AJAX

Abychom nemuseli při každé operaci (aktualizaci) načítat stránku ze serveru, lze využít technologii AJAX (Asynchronous JavaScript and XML). Největší výhodou je, že urychluje uživatelskou práci. Při nějaké změně se nemusí pokaždé obnovovat stránka. Kvůli tomu se aplikace chovají jako desktopové aplikace (aktualizují části stránky, nahrávají nebo ukládají data a to nezávisle na ostatních). Díky tomu se nemusí posílat celý kód na server, ale jen změněná část a tím šetří datové přenosy. Jedna z největších nevýhod u aplikací s použitím AJAX je taková, že znemožňuje použití tlačítka Zpět v prohlížeči (používá se jen pro statické stránky). Další nevýhoda nastává při vypnutí JavaScriptu prohlížeče na straně klienta, v tomto případě je ajaxová aplikace nefunkční. [13][15]

2.4 ASP.NET

Softwarová struktura .NET framework obsahuje webovou platformu ASP.NET, která je použita ve vytvořené aplikaci. V ASP.NET lze použít jakýkoliv programovací jazyk, který je kompatibilní s modulem CLR (Common Language Runtime), včetně Visual Basic a C#. [11]

Předchůdce ASP.NET je verze ASP (Active Server Pages). Zásadní nevýhodou ASP je ta, že programovací kód a HTML kód se míchají do sebe a tím se snižuje rozšiřitelnost a hlavně přehlednost. Stránky se nekompilují a tím dochází k úbytku výkonu. [7]

U ASP.NET se používá odlišný přístup, kde do HTML kódu vložíme tzv. serverové komponenty jako např. Textbox (textové pole), Button (tlačítko). Webová stránka vytvořená pomocí ASP.NET obsahuje jak HTML kód, tak i programovací kód, který zajišťuje funkčnost stránky. Programovací kód není vložen v HTML kódu, ale je ve zvláštní sekci souboru nebo v samostatném souboru. [7]

2.5 Programovací jazyk C#

Jako programovací kód ASP.NET je použit přísně typový objektově orientovaný programovací jazyk C#. Je založen na jazycích Java a C++ (podobá se v syntaxi). Jazyk běží na rozhraní .NET framework, je jednoduchý, výkonný, robustní atd. [12][11]

2.5.1 Visual Web Developer Express

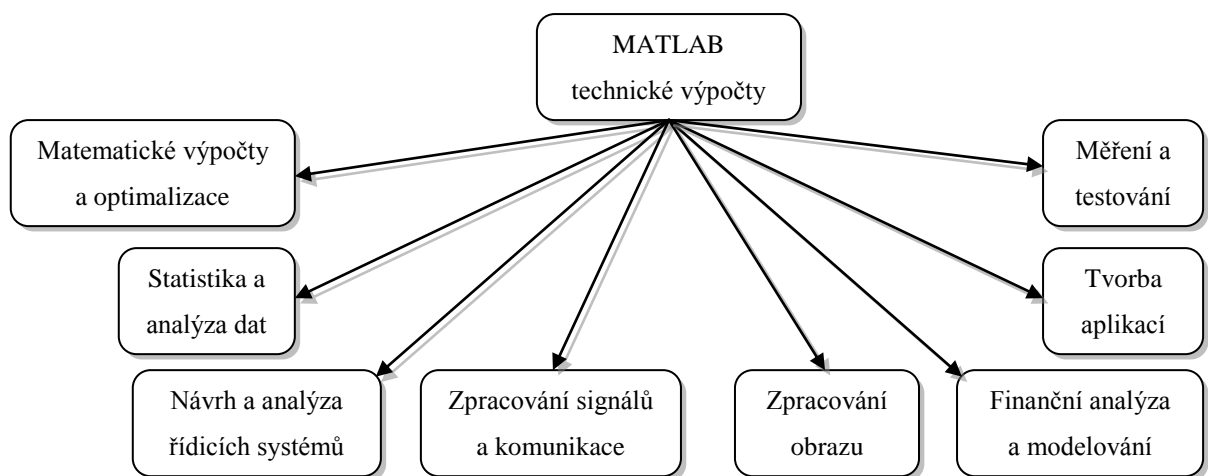
Aplikace Visual Web Developer Express poskytuje rozhraní, které využívá jednoduché nástroje pro vytváření webových aplikací. Software vydaný roku 2010 je použit pro vytvořenou webovou aplikaci. S funkcemi technologie ASP.NET lze využít autorizaci na web, přihlášení, ověřování přihlášení a různých nastavovacích prvků pro zabezpečení aplikace. V aplikaci lze vidět náhled webu, bez nutnosti jeho spuštění přes virtuální server, a poskytuje také nástroje k práci s CSS styly a jiné. [11]

3 PROSTŘEDÍ MATLAB

MATLAB je interaktivní prostředí pro vizualizaci, programování, vědecké a technické výpočty, analýza dat, vytváření algoritmů a jiné. Prostředí MATLAB vyvíjí firma MathWorks, které lze využít v mnoha oborech, jako např. pro inženýry, vědce, matematiky a učitelé pro výuku studentů nebo různých komplexních řešení. Toto prostředí se používá např. pro zpracování signálů, obrazu a videa, pro návrhy řídicích systémů, pro řešení soustav lineárních, diferenciálních rovnic a mnoho jiné. [6]

Na stránkách firmy MathWorks (<http://www.mathworks.com>) se dozvíte vše potřebné k této placené aplikaci. Web obsahuje různé semináře jak pro základní práci, tak i speciální ovládání s konkrétními moduly. Je zde i velká škála dokumentací, příkladů a fórum pro komunikaci mezi uživateli. [6]

Aplikace obsahuje mnoho rozšíření a toolboxů (Obr. 1), kde nejznámější z nich je Simulink. Tento rozšiřující modul se používá pro modelování a simulaci dynamických systémů. [6]



Obr. 1. Rozšíření MATLABu [6]

K této práci bylo potřeba rozšiřující modul pro tvorbu aplikací MATLAB Deployment Tool, který obsahuje toolbox MATLAB Builder NE.

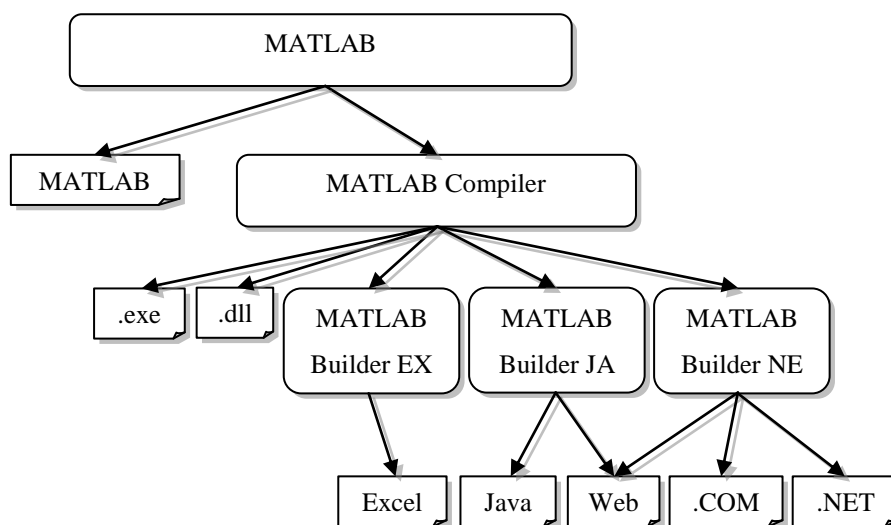
3.1 MATLAB Deployment Tool

V dřívějších verzích zajišťoval komunikaci mezi MATLABem a webovým serverem MATLAB Web Server a zpřístupnil aplikace uživatelům přes webové stránky. Tento modul ale není v novějších verzích obsažen a ani podporován. [6]

MATLAB Deployment Tool obsahuje MATLAB Compiler a ten umožňuje sdílet programy MATLAB jako knihovny pro integraci se společnými programovacími jazyky (C#, JAVA, Excel) nebo jako samostatné aplikace (C a C++).

Rozdělení MATLAB Compiler (Obr. 2):

- MATLAB Builder NE (C#)
- MATLAB Builder JA (JAVA)
- MATLAB Builder EX (Excel)



Obr. 2. Producty MATLABu [8]

MATLAB Builder EX propojuje Excel s MATLABem, kde vytváří různé funkce pro výpočty v Excelu spojené s MATLABem.

MATLAB Builder JA převádí vytvořené aplikace do jedné nebo více tříd JAVA. Lze je využít při programování aplikací pro osobní počítače nebo i web servery.

MATLAB Builder NE převádí aplikace vytvořené v MATLABu do komponent .COM nebo .NET. Komponenty lze využít jak v osobních počítačích, tak i na web serverech.

Tyto Buildery se používají spolu s knihovnou MCR (MATLAB Compiler Runtime), která zajišťuje chod dané komponenty. [8]

V aplikaci je použit toolbox MATLAB Builder NE z verze MATLABu R2008b.

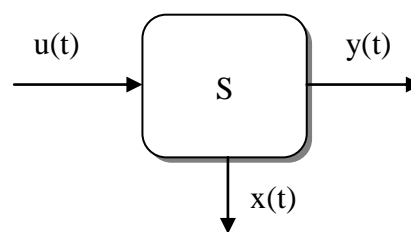
4 LINEÁRNÍ ČASOVĚ INVARIANTNÍ SYSTÉMY

Systémy se třídí dle různých kritérií a hledisek, ale zde bude popsána pouze jejich vybraná část. Pojem lineární systémy znamená, že vazby mezi jednotlivými veličinami jsou lineární. U nelineárních systémů je tento princip opačný, a to tak, že některé vazby těchto systémů jsou nelineární. Systémy můžeme rozlišovat podle změn chování v čase. Invariantní systém neboli stacionární je takový systém, jehož parametry se nemění v čase. U variantního systému se některé jeho parametry v čase mění. Další rozdělení systému je podle změn vstupních a výstupních veličin systému a to na spojité a nespojitě. Spojité systémy mají v každém časovém úseku spojitý signál. Pokud systémy nejsou v každém časovém úseku spojité, systémy se nazývají diskrétní neboli nespojitý. Jeden z dalších kritérií rozdělení je počet vstupů a výstupů systému. Systémy s 1 vstupem a 1 výstupem se nazývají jednorozměrné (SISO) a mnohorozměrné (MIMO) jsou systémy s více vstupy a více výstupy. [3][2]

4.1 Vnitřní a vnější popis

Systémy se dají matematicky popsat několika způsoby. Nejzákladnější rozdělení je vnitřní a vnější popis.

Vnitřní popis nebo také nazvaný stavový popis popisuje reakci mezi vstupem, vnitřním stavem a výstupem systému (Obr. 3).



Obr. 3 Jednorozměrný lineární dynamický systém [2]

Vektorově maticový zápis:

$$\mathbf{x}'(t) = \mathbf{A}(t)\mathbf{x}(t) + \mathbf{B}(t)\mathbf{u}(t) \quad (1)$$

$$\mathbf{y}(t) = \mathbf{C}(t)\mathbf{x}(t) + \mathbf{D}(t)\mathbf{u}(t) \quad (2)$$

,kde $\mathbf{x}(t)$ – stavová proměnná,

$\mathbf{u}(t)$ – vstupní veličina,

$y(t)$ – výstupní veličina.

A – matice systému, B – váhová matice vstupu, C – váhová matice stavu, D – váhová matice vstupu.

Rozměry matic:

- $A: n \times n$
- $B: n \times m$
- $C: r \times n$
- $D: r \times m$

,kde n – počet stavů,

m – počet vstupů,

r – počet výstupů.

Pokud alespoň některá z matic nebo jejich prvky A , B , C , D jsou proměnné v čase, systém je variantní (nestacionární), jinak je systém invariantní (stacionární). [3][2]

Vnější popis znamená popis mezi vstupy a výstupy systému, kde neznáme vnitřní stavy systému. Systém je pro nás jako blok se vstupy a výstupy a popisujeme reakci systému na vstupní signály. Vnější popis se zapisuje několika způsoby, např. přenosovou funkcí, lineární diferenciální rovnicí a jinak. [2]

Lineární diferenciální rovnice:

$$a_n y^{(n)}(t) + a_{n-1} y^{(n-1)}(t) + \dots + a_1 y'(t) + a_0 y(t) = b_m u^{(m)}(t) + \dots + b_0 u(t) \quad (3)$$

,kde a_i , b_j jsou konstantní koeficienty,

$u(t)$ – vstupní veličina,

$y(t)$ – výstupní veličina systému.

Aby byl systém fyzikálně realizovatelný (ryzí), musí platit: $m \leq n$

Přenosová funkce:

$$G(s) = \frac{Y(s)}{U(s)} = \frac{b_m s^m + \dots + b_1 s + b_0}{a_n s^n + \dots + a_1 s + a_0} \quad (4)$$

I zde musí platit podmínka fyzikální realizovatelnosti.[3][2] Jmenovatel přenosové funkce se nazývá charakteristický polynom.

Převod vnitřního popisu na vnější popis lze pomocí vzorce (5):

$$G(s) = \frac{Y(s)}{U(s)} = C(sI - A)^{-1}B + D \quad (5)$$

Více o tomto tématu najdete v [2] nebo [3].

4.2 Druhy systémů z hlediska spojitosti v čase

Jeden z nejdůležitějších rozdělení systémů je rozdělení na systémy spojité a diskrétní.

Jako popis lineárních dynamických **spojitých systémů** se používá Laplaceova transformace. Přímá L – transformace je popsána vztahem (6):

$$F(s) = L\{f(t)\} = \int_0^{\infty} f(t) e^{-st} dt \quad (6)$$

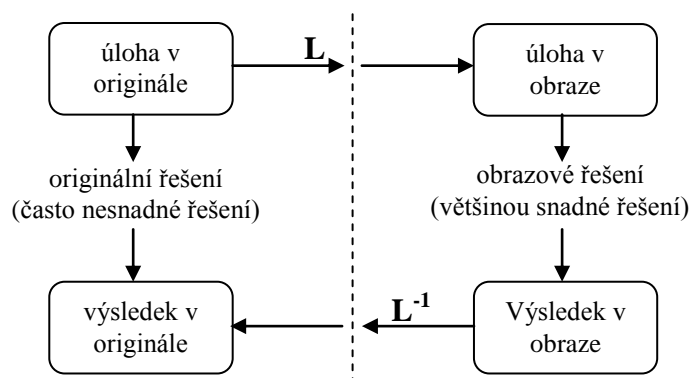
Zpětná L – transformace:

$$f(t) = L^{-1}\{F(s)\} = \frac{1}{2\pi j} \int_{c-j\infty}^{c+j\infty} F(s) e^{st} ds \quad (7)$$

Laplaceova transformace transformuje funkce z časové oblasti do oblasti komplexní. Tato operace umožňuje nahradit složité matematické diferenciální rovnice jednoduchými algebraickými rovnicemi. [2]

Postup výpočtu je znázorněn na obr. 4., kde rovnici (úloha v originále) převedeme za pomoci přímé L – transformace. Tato transformovaná rovnice (úloha v obraze) je lineární algebraická a vypočítáme z ní celkem jednoduše řešení v obraze. Poté za pomoci inverzní L – transformace a tzv. slovníku L – transformace se určí výsledek v originále. [2][3]

Při zpětné L – transformaci je postup opačný, ale při výpočtu mezi obrazy se využívá slovníku (a výsledek je přímo už v originále) nebo jiných operací. [2][3]



Obr. 4. Postup výpočtu při L – transformaci [2]

S rozvojem informačních systémů se pro rychlejší a přesnější zpracování pomocí počítačů využívá diskretní signál. Tento signál je tvořen z posloupnosti diskretních hodnot, které se opakují v určitých časových okamžicích T .

K popsání lineárních dynamických **diskretních systémů** se používá Z – transformace. Přímá Z – transformace je popsána vztahem:

$$F(z) = Z\{f(kT)\} = \sum_{k=0}^{\infty} f(kT) z^{-k} = f(0) + f(T)z^{-1} + f(2T)z^{-2} + \dots \quad (8)$$

Zpětná Z – transformace:

$$f(kT) = Z^{-1}\{F(z)\} = \frac{1}{2\pi j} \int_{c-j\infty}^{c+j\infty} F(z) z^{k-1} dz \quad (9)$$

Z – transformace je obdobná k L – transformaci. Používá se k řešení složitých matematických diferenčních rovnic. Při použití přímé Z – transformace se využívá komplexní mocninné řady. U zpětné Z – transformaci tedy při výpočtu originálu z obrazů se zpravidla používá slovník. Lze ale využít i jiné metody [2]:

- rozvoj obrazu v mocninnou řadu
- rozklad obrazu na částečné (parciální) zlomky
- zpětná Z – transformace pomocí derivace
- definiční vzorec pro zpětnou Z - transformaci

Pro podrobnější popis L a Z – transformace a jejich slovníky viz. [2][3].

4.3 Základní vlastnosti systému

Chování systému se popisuje pomocí základních vlastností systému. V dalších bodech budou popsány základní vlastnosti pomocí spojitého systému.

4.3.1 Póly a nuly

Přenosovou funkci (4) můžeme vyjádřit i jiným způsobem a to pomocí pólů a nul:

$$G(s) = \frac{b_m(s - n_1) \cdot \dots \cdot (s - n_m)}{(s - p_1) \cdot \dots \cdot (s - p_n)} \quad (10)$$

,kde nuly: $n_1 - n_m$,

póly: $p_1 - p_n$.

Pokud ale póly a nuly zobrazíme v komplexní rovině, zjistíme předběžné chování systému. Např. když budou póly v záporné reálné polorovině komplexní roviny, systém bude stabilní (viz 4.3.4). Při reálných pólech bude mít systém aperiodický přechodový děj. Derivační charakter bude převládat v systému, kde jsou nuly v počátku, atd. [2][3]

4.3.2 Řád

Řád systému je u přenosové funkce dán stupněm největší mocniny u polynomu ve jmenovateli. U stavového popisu je dán rozměrem stavového prostoru.

4.3.3 Zesílení

Zesílení systému $Z(s)$ se dá vyjádřit tak, že přivedeme omezený vstupní signál $u(t)$ při nulových počátečních podmínkách a hodnota zesílení je potom podíl mezi ustálenou výstupní hodnotou $y(t)$ a vstupní hodnotou $u(t)$:

$$Z(s) = \frac{\lim_{t \rightarrow \infty} y(t)}{\lim_{t \rightarrow \infty} u(t)} \quad (11)$$

Jestliže máme systém popsáný pomocí (4), tak dostaneme [3]:

$$Z(s) = \frac{b_0}{a_0} \quad (12)$$

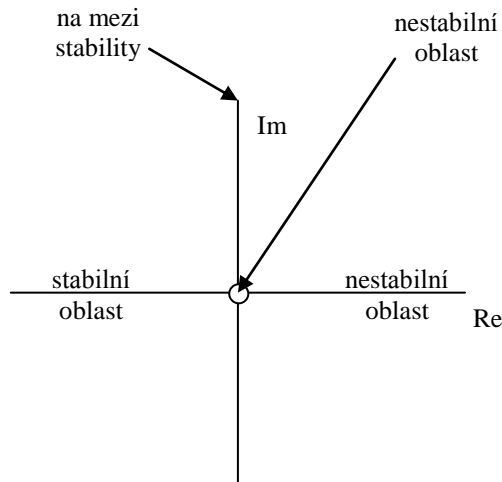
4.3.4 Stabilita

Důležitá vlastnost systému je jeho stabilita. Nejpoužívanější definice stability pro dynamické systémy je Ljapunova stabilita, která se používá pro lineární i nelineární systémy. Dále je BIBO stabilita, kde omezený vstup generuje omezený výstup. Pro spojitě lineární systémy je stabilita určena pomocí pólů, kde stabilní systém je tehdy, když póly systému leží v levé polorovině Gaussovy komplexní roviny (Obr. 5.). U diskretních systémů je systém stabilní, když póly systému leží uvnitř jednotkové komplexní kružnice. Když je charakteristická rovnice vyšších řádů a výpočet kořenů (pólů) je zdlouhavý a náročný, lze použít tzv. kritérií stability. [2][3]

Nejznámější kritéria jsou:

- Algebraická kritéria stability
 - Routh – Shurovo
 - Hurwitzovo

- Geometrická kritéria stability
 - Nyquistovo
 - Michajlovo



Obr. 5. Póly v Gaussově rovině [3]

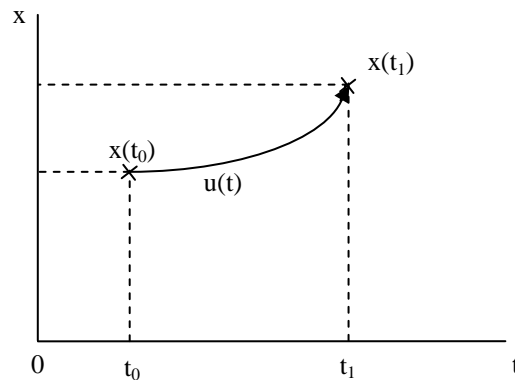
Pro popis dalších vlastností budeme vycházet z popisu systému pomocí rovnice (1) a (2).

4.3.5 Řiditelnost a dosažitelnost stavu

Dosažitelný systém je takový, u kterého existuje časový okamžik $t_0 < t_1$ ($t_1 - t_0$ je konečný interval) a vstup $u(t)$, pomocí kterého lze systém ze stavu $x(t_0)$ převést do stavu $x(t_1)$ (Obr. 6.). Jestliže je rozměr n (počet stavů) stavového prostoru rovno hodnotě (14) matice (13), systém je dosažitelný. [4][2]

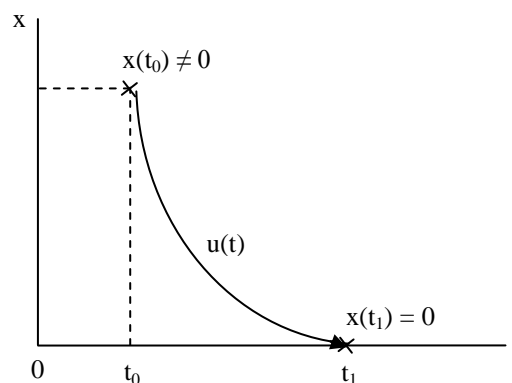
$$Q_R = [B, AB, \dots, A^{n-1}B] \quad (13)$$

$$h(Q_R) = n \quad (14)$$



Obr. 6. Dosazitelnost stavu

Řiditelný systém je takový, u kterého existuje časový okamžik $t_0 < t_1$ ($t_1 - t_0$ je konečný interval) a vstup $u(t)$, pomocí kterého lze systém ze stavu $x(t_0) \neq 0$ převést do stavu $x(t_1) = 0$ (Obr. 7.). Jestliže je rozměr n (počet stavů) stavového prostoru rovno hodnotě (14) matice (13), systém je říditelný. Jestliže u systémů, které jsou spojité a časově invariantní, existuje dosažitelný stav, existuje i říditelný stav. U takových systémů stačí řešit jejich říditelnost. [4][2]



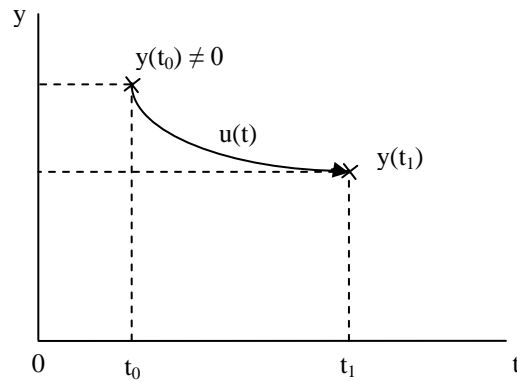
Obr. 7. Řiditelnost stavu

4.3.6 Řiditelnost (dosažitelnost) výstupu

Řiditelný výstup systému existuje, jestliže časový okamžik $t_0 < t_1$ ($t_1 - t_0$ je konečný interval) a vstup $u(t)$, pomocí kterého lze systém z výstupu $y(t_0) \neq 0$ převést na koncový výstup $y(t_1)$ (Obr. 8.). Jestliže je rozměr r (počet výstupů) stavového prostoru rovno hodnotě (16) matice (15), systém má říditelný (dosažitelný) výstup. [4][2]

$$Q_V = [CB, CAB, \dots, CA^{n-1}B] \quad (15)$$

$$h(Q_V) = r \quad (16)$$



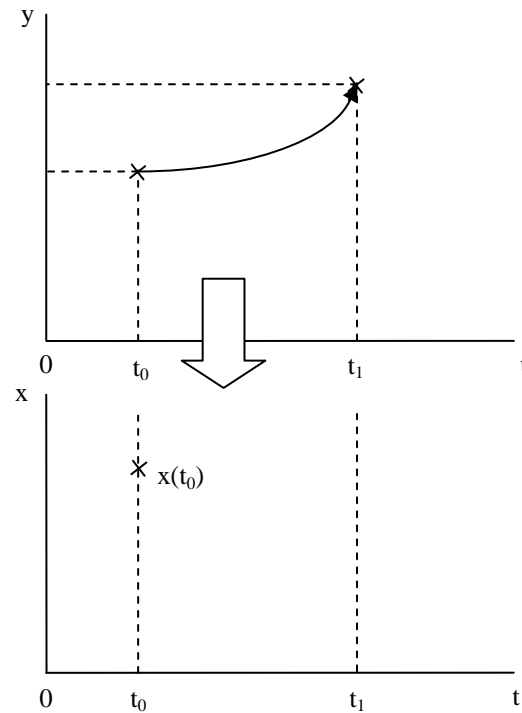
Obr. 8. Řiditelnost výstupu

4.3.7 Pozorovatelnost a rekonstruovatelnou stavu

Pozorovatelný stav systému $x(t_0)$ znamená, že ho lze určit s pomocí budoucích hodnot výstupního vektoru $y(t)$, kde $t_0 < t_1$ ($t_1 - t_0$ je konečný interval) (Obr. 9.). Jestliže je rozměr n (počet stavů) stavového prostoru rovno hodnotě (18) matice (17), systém je pozorovatelný. [4][2]

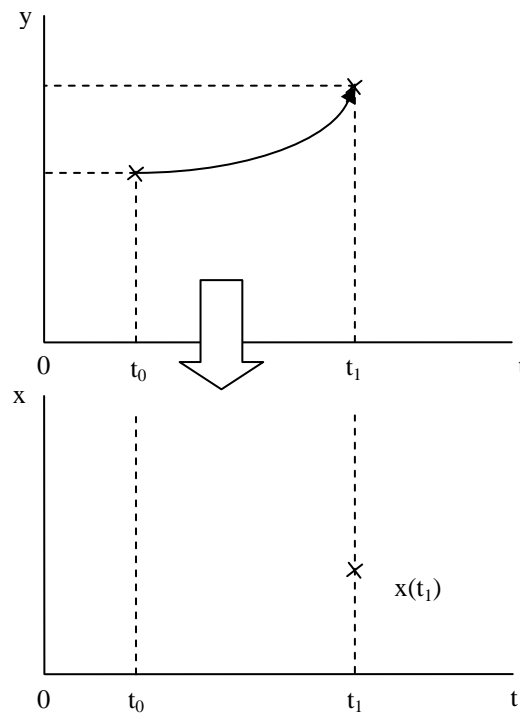
$$Q_P = \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{n-1} \end{bmatrix} \quad (17)$$

$$h(Q_P) = n \quad (18)$$



Obr. 9. Pozorovatelnost stavu

Rekonstruovatelný systém $x(t_0)$ je takový, jestliže na základě předchozích hodnot výstupu, kde časový okamžik $t_0 < t_1$ ($t_1 - t_0$ je konečný interval), lze určit jeho stav $x(t_1)$ (Obr. 10.). Jestliže je rozměr n (počet stavů) stavového prostoru rovno hodnotě (18) matice (17), systém je rekonstruovatelný. Pokud jsou lineární časově invariantní systémy pozorovatelné, tak jsou i rekonstruovatelné. [4][2]



Obr. 10. Rekonstruovatelnost stavu

4.4 Dopravní zpoždění

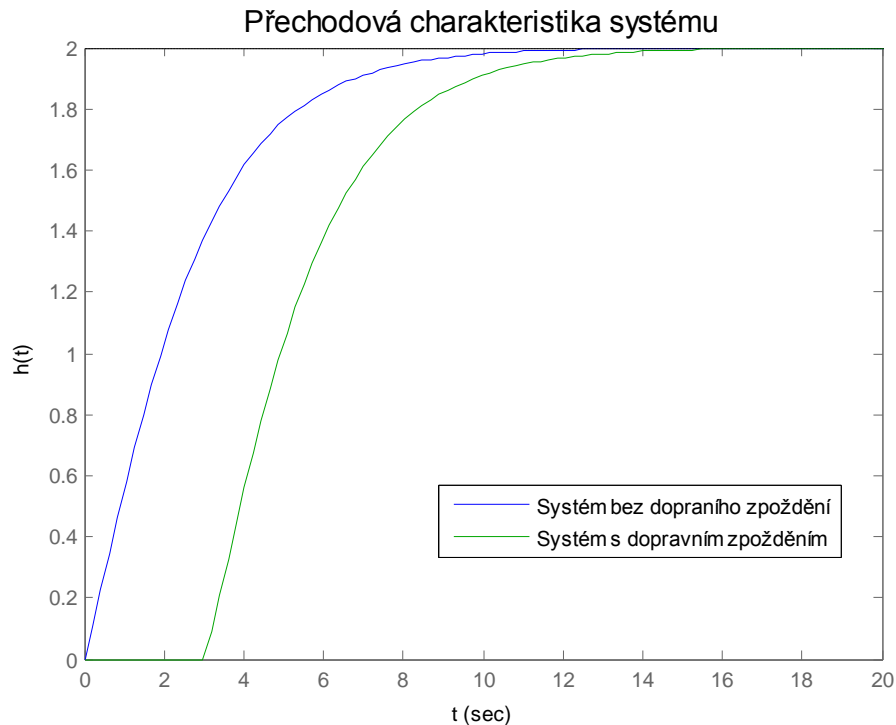
V praxi se u systému často objevuje dopravní zpoždění. Toto zpoždění vzniká v řízených systémech při transportních jevech. Např. při proudění tekutin v potrubí v určitém čase pošleme signál do ventilu k omezení toku tekutin. Čas poslaného signálu k ventilu spolu s časem vykonané práce ventilu je označeno jako τ_d a nazýváme ho dopravním zpožděním. Rovnice s dopravním zpožděním pro lineární časově invariantní systémy je:

$$y^{(n)}(t) + a_{n-1}y^{(n-1)}(t) + \dots + a_1y'(t) + a_0y(t) = b_0u(t - \tau_d) \quad (19)$$

Pro přenosový tvar se dopravní zpoždění zapisuje jako:

$$G(s) = \frac{Y(s)}{U(s)} = \frac{b_0}{s^n + a_{n-1}s^{n-1} + \dots + a_1s + a_0} e^{-\tau_d s} \quad (20)$$

Dynamická charakteristika systému s dopravním zpožděním se projeví jako posunutí na časové ose (Obr. 11.). [3]



Obr. 11. Dopravní zpoždění systému

4.5 Mnohorozměrné systémy

Dosud popsané systémy byly jednorozměrné neboli SISO (single input – single output) systémy. Mají tedy jenom jednu vstupní veličinu, kterou nastavujeme. U MIMO (multiple input – multiple output) systémů je možné nastavovat více veličin (např. kotle, turbíny apod.). Web – aplikace je ale vytvořena pouze pro SISO systémy. [2]

4.6 Základní grafické charakteristiky

4.6.1 Přechodová charakteristika

Přechodová charakteristika graficky znázorňuje přechodovou funkci, kde tato funkce znamená odezvu systému na jednotkový skok při nulových počátečních podmínkách. Tento jednotkový skok (Heavisideův skok)(Obr. 12.) je označen jako $\eta(t)$ a definuje se pomocí vztahu:

$$u(t) = \eta(t) = \begin{cases} 1 & \text{pro } t \geq 0 \\ 0 & \text{pro } t < 0 \end{cases} \quad (21)$$

Laplaceův obraz:

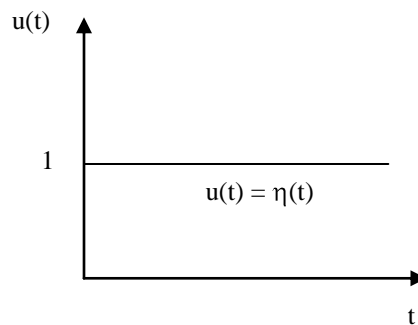
$$L\{\eta(t)\} = L\{1(t)\} = \frac{1}{s} \quad (22)$$

a obraz přechodové funkce:

$$L\{h(t)\} = H(s) = G(s)U(s) = \frac{G(s)}{s} \quad (23)$$

Potom přechodovou funkci vypočteme za pomoci zpětné L – transformace jako:

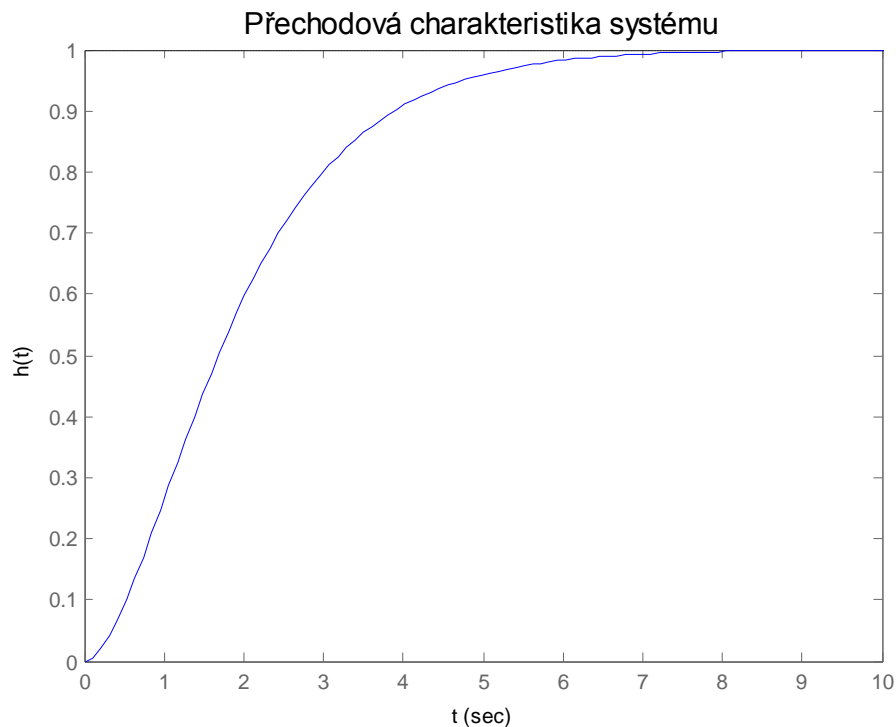
$$h(t) = L^{-1}\left\{\frac{G(s)}{s}\right\} \quad (24)$$



Obr. 12. Jednotkový skok [2]

Příklad přechodové charakteristiky rovnice (25) je vidět na Obr. 13. Více najdete např. v [2] a [3].

$$G(s) = \frac{1}{s^2 + 2s + 1} \quad (25)$$



Obr. 13. Přechodová charakteristika

4.6.2 Impulsní charakteristika

Odezva systému na jednotkový impuls při nulových počátečních podmínkách se označuje jako impulsní funkce a grafickým vyjádřením impulsní funkce je impulsní charakteristika. Jednotkový impuls (Diracův impuls) má nekonečně velkou výšku a nekonečně malou šířku (Obr. 14.). Je to ideální impuls, který se označuje jako $\delta(t)$, není fyzikálně realizovatelný a definuje se pomocí vztahu:

$$\int_{-\infty}^{\infty} \delta(t) dt = 1 \quad (26)$$

,kde $\delta(t) = 0$, pro $t \neq 0$.

Laplaceův obraz:

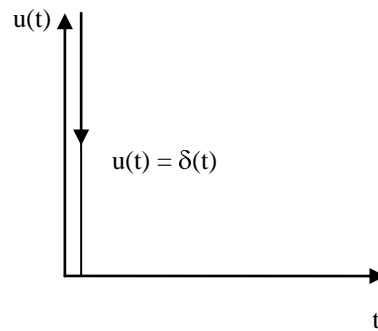
$$L\{\delta(t)\} = 1 \quad (27)$$

a obraz impulsní funkce:

$$L\{g(t)\} = G(s)L\{\delta(t)\} = G(s) \quad (28)$$

Potom impulsní funkci vypočteme za pomoci zpětné L – transformace jako:

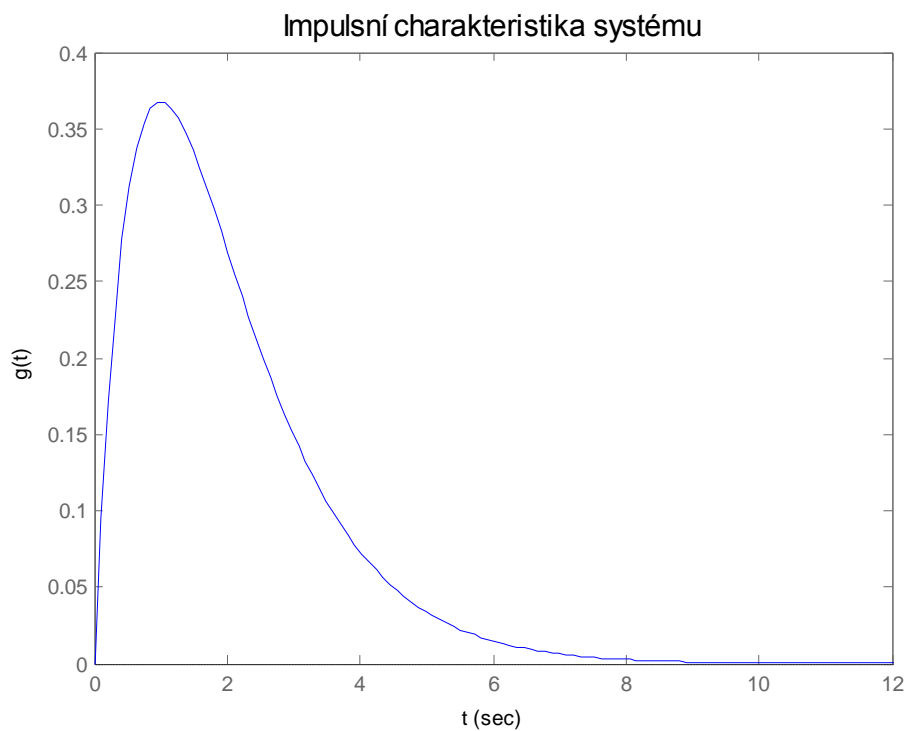
$$g(t) = L^{-1}\{G(s)\} \quad (29)$$



Obr. 14. Jednotkový impuls [2]

Příklad impulsní charakteristiky rovnice (30) je na Obr. 15. Více lze najít např. v literatuře [2] a nebo [3].

$$G(s) = \frac{1}{s^2 + 2s + 1} \quad (30)$$



Obr. 15. Impulsní charakteristika

4.6.3 Frekvenční charakteristika v komplexní rovině

Frekvenční přenos je definován pomocí Fourierových obrazů, kde z jeho spojitého Laplaceova protějšku dáme komplexní proměnnou s rovno $j\omega$ (31) a dostaneme frekvenční přenos ve tvaru (32).

$$G(j\omega) = [G(s)]_{s=j\omega} \quad (31)$$

$$G(j\omega) = \operatorname{Re}[G(j\omega)] + j \operatorname{Im}[G(j\omega)] = P(\omega) + j Q(\omega) \quad (32)$$

,kde $P(\omega)$ – reálná část přenosu,

$Q(\omega)$ – imaginární část přenosu.

Frekvenční přenos lze dále upravit na exponenciální tvar komplexního čísla:

$$G(j\omega) = A(\omega)e^{j\varphi(\omega)} = |G(j\omega)|e^{j \operatorname{arg}G(j\omega)} \quad (33)$$

,kde ω - úhlová frekvence

$A(\omega)$ - amplituda,

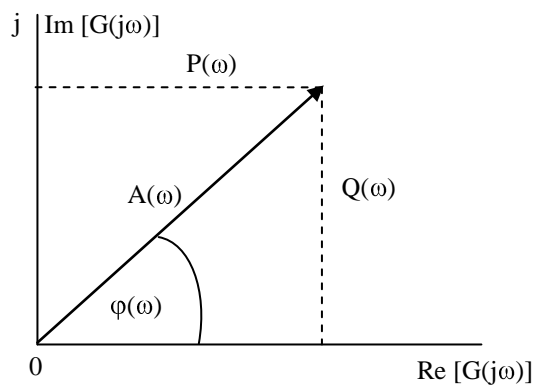
$\varphi(\omega)$ - fáze.

Pro zobrazení bodu frekvenční charakteristiky (Obr. 16.) v komplexní rovině je potřeba získat amplitudu a jeho fázi. Amplituda systému $A(\omega)$ se získá absolutní hodnotou komplexního čísla, kde dle Pythagorovy věty platí:

$$A(\omega) = \sqrt{P^2(\omega) + Q^2(\omega)} \quad (34)$$

Fázi (jeho úhel) $\varphi(\omega)$ vypočítáme:

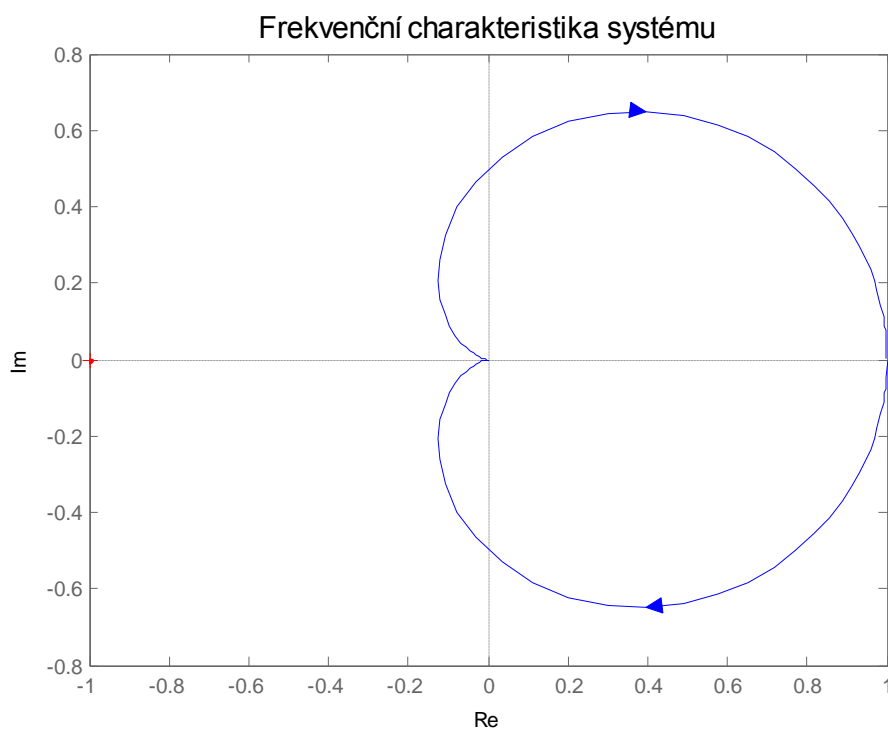
$$\operatorname{tg} \varphi(\omega) = \frac{Q(\omega)}{P(\omega)} \Rightarrow \varphi(\omega) = \operatorname{arctg} \frac{Q(\omega)}{P(\omega)} \quad (35)$$



Obr. 16. Zobrazení bodu frekvenčního přenosu v komplexní rovině [3]

Příklad frekvenční charakteristiky systému v komplexní rovině rovnice (36) je na Obr. 17. Více informací najdeme např. v použité literatuře [2] a [3].

$$G(s) = \frac{1}{s^2 + 2s + 1} \quad (36)$$



Obr. 17. Frekvenční charakteristika v komplexní rovině

4.6.4 Amplitudově - fázová frekvenční charakteristika v log. souřadnicích

Toto grafické zobrazení vykresluje amplitudu a fázi frekvenčního přenosu v logaritmických souřadnicích. Zlogaritmování frekvenčního přenosu (33) dostaneme tvar:

$$\ln G(j\omega) = \ln A(\omega) + j\varphi(\omega) = \ln|G(j\omega)| + j \arg G(j\omega) \quad (37)$$

Pro grafické zobrazení vyjádříme logaritmickou amplitudovou charakteristiku:

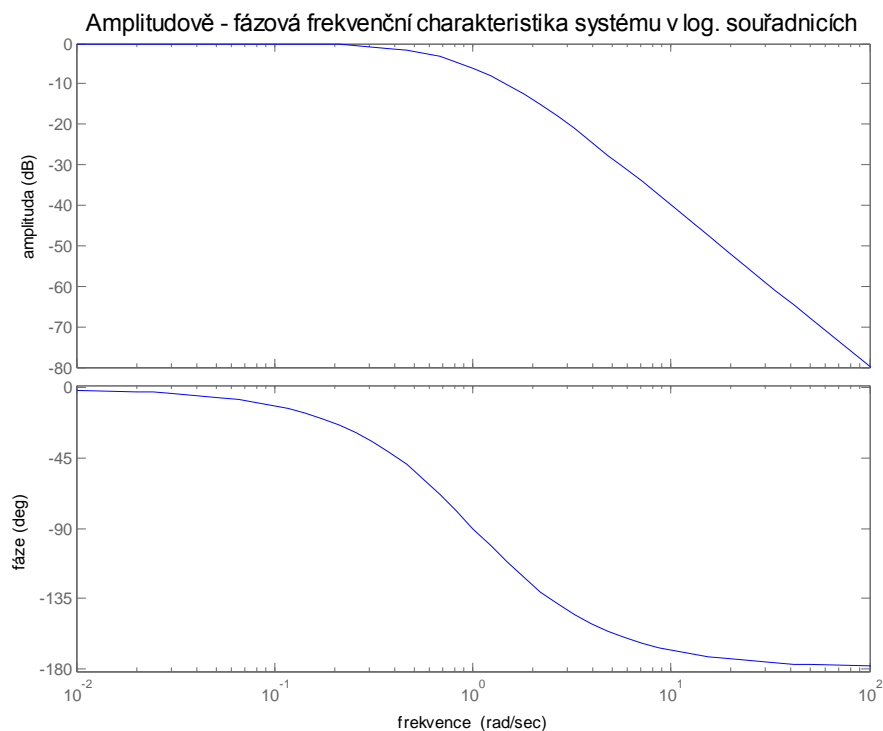
$$\ln|G(j\omega)| = f(\omega) \quad (38)$$

a logaritmickou fázovou charakteristiku:

$$\varphi(\omega) = f(\omega) \quad (39)$$

Příklad amplitudově – fázové frekvenční charakteristiky systému rovnice (40) je na Obr. 18. Další informace nalezneme např. v literatuře [2] a nebo [3].

$$G(s) = \frac{1}{s^2 + 2s + 1} \quad (40)$$



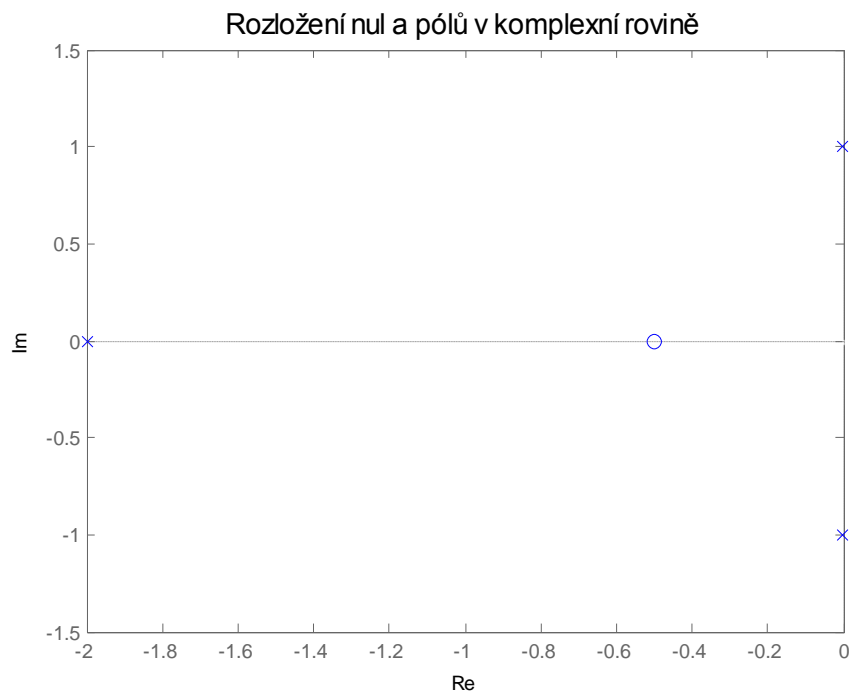
Obr. 18. Amplitudově - fázová charakteristika v log. souřadnicích

4.6.5 Rozložení nul a pólů v komplexní rovině

Rozložení nul a pólů v komplexní rovině se vykresluje z tvaru přenosové funkce, kde jsou rozložené kořeny čitatele (nuly) a jmenovatele (póly) podle rovnice (10) (viz. 4.3.1).

Příklad rozložení nul a pólů v komplexní rovině rovnice (41) je na Obr. 19. [2]

$$G(s) = \frac{s + 0,5}{s^3 + 2s^2 + s + 2} \quad (41)$$



Obr. 19. Rozložení nul a pólů v komplexní rovině (x – póly, o - nuly)

II. PRAKTICKÁ ČÁST

5 STRUKTURA APLIKACE

Aplikace byla naprogramována v programu Microsoft Visual Web Developer 2010 Express. Do tohoto programu byla implementována knihovna z MATLABu R2008b, která byla vytvořena za pomoci toolboxu MATLAB Builder NE.

5.1 MATLAB soubory

V programu MATLAB bylo vytvořeno celkem 41 tříd. Každá třída je rozdělena do 4 kategorií podle typu funkce a to na transfer, test, properties a graphs. Zkompilované třídy jsou vidět v příloze P II.

Transfer je kategorie pro vykonávání změny zadaných dat. Patří sem transformace mezi vnitřním a vnějším popisem, spojitým a diskrétním systémem a jejich různé varianty mezi sebou. Dále zde patří změna diskrétního systému převzorkováním pro změnu vzorkovací periody. Pro zachování formy čísla na 4 platná desetinná čísla, byla vytvořena i funkce zaokrouhlování.

Kategorie **test** obsahuje třídy pro testování zadávaných systémů pro jejich fyzikální realizovatelnost, a jestli zadávaný systém je jednorozměrný nebo mnohorozměrný (kapitola 4.5).

V kategorii **properties** jsou obsaženy minimalizace systémů a jejich vlastnosti. Nachází se zde funkce pro výpočet pólů a nul, řádu systému, zesílení a stability. Dále je zde výpočet pro říditelnost / dosažitelnost stavu, říditelnost výstupu a pozorovatelnost / rekonstruovatelnost stavu. Tyto vlastnosti systémů jsou více probrány v kapitole 4.3.

Poslední kategorie je **graphs**. Tato kategorie obsahuje třídy pro vykreslování grafů jako je přechodová charakteristika, impulsní charakteristika a jiné (viz kapitola 4.6).

Vytvořená knihovna nazvaná LTI.dll obsahuje tyto výše uvedené 4 kategorie pro analýzu lineárních časově invariantních systémů.

Systém lze zadat v přenosovém tvaru nebo ve stavovém popisu (Obr. 28 a 29). Přenosový tvar obsahuje 2 části a to čítec a jmenovatel. Stavový popis obsahuje 4 části a těmi jsou 4 matice, kde je rozdílný počet sloupců a řádků, ale současně se musí dodržet správný rozměr matic (viz. kapitola 4.1).

Při vytváření knihovny byly možné 2 varianty. Vytvořit menší počet tříd, kde bude jedna třída předělávat tvar zadaného systému na druhý a zpátky. S převedeným tvarem se bude nadále počítat pro další třídy nebo se vytvoří třídy pro každý tvar zvlášť.

V **první** variantě by se zmenšil počet vytvořených tříd. Funkce v programu MATLAB může obecně mít více než jednu výstupní hodnotu, ale práce s C# dovoluje pouze jednu výstupní hodnotu. Výstupní hodnota by tedy musela být buď ve tvaru pole, které by obsahovalo všechny hodnoty a při transformaci systému, by musela obsahovat i jestli a jak se tvar změnil, nebo by se funkce volala častěji a výstupní hodnota by vypisovala jinou hodnotu, dle vstupní hodnoty.

Zvolená **druhá** varianta vytvoří více tříd než v prvním případě. Pro každý tvar je vytvořená třída, která se volá taky více než jednou kvůli výpisu jedné výstupní hodnoty, ale ne tak často jako v první variantě. Výstupní hodnota se nemusí nijak složitě zpracovávat, jak by tomu bylo v první variantě při výpisu všech hodnot v poli.

Na obr. 20. je třída pro transformaci spojitého systému na diskretní systém v přenosovém tvaru. Jako vstupní parametry jsou číselný a jmenovatel zadaného systému. Dále je zde vzorkovací perioda a dopravní zpoždění systému. A jako poslední parametr je číslo, které označuje jakou hodnotu má třída vrátit, jestli transformovaný číselný nebo jmenovatel.

```
function[vysl] = spoj_disk_tf(cit1,jm1,vzork,delay,cislo)
% funkce převádí spojitý přenosový tvar na diskretní

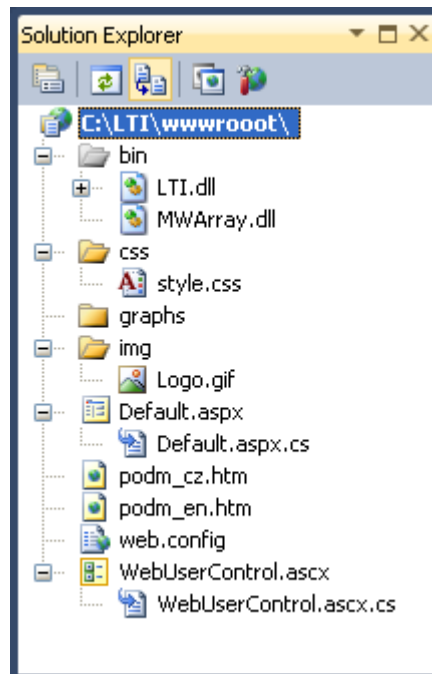
sys = c2d(tf(cit1,jm1,'Inputdelay',delay),vzork,'tustin');
Td = get(sys,'Inputdelay');
[fit,jm]=tfdata(sys,'v');

if cislo == 1
    vysl = fit;
elseif cislo == 2
    vysl = jm;
else
    vysl = Td;
end
end
```

Obr. 20. Převod spojitého systému na diskretní pro přenosový tvar

5.2 Soubory .NET

Webová aplikace je vytvořena několika komponentami a různými prvky pro zobrazení aplikace (viz Obr. 21).



Obr. 21. Souborová struktura

Složka **bin** obsahuje již vytvořenou knihovnu LTI.dll a knihovnu MWArray.dll, která je zkopírovaná z programu MATLAB. Knihovna MWArray.dll obsahuje např. proměnné typu MWArray, MWNumericArray, potřebné metody a jiné. Využívá se ke komunikaci mezi vytvořenou knihovnou v Builder NE a programovacím jazykem C#.

Ve složce **css** je vytvořený kaskádní styl style.css, který se využívá pro zobrazení vytvořené aplikace. Jak je napsáno v kapitole 2.2, je tento styl zkopírovaný pro zachování designu s několika doplňujícími značkami pro LTI - aplikaci.

Další složka **graphs** slouží k ukládání grafů, které si uživatelé nechali vykreslit. V aplikaci byla vytvořena podmínka pro mazání grafů, ale vzhledem k nastavení serveru, kde uživatelé nemohou nic na serveru měnit, byla odstraněna. Grafy se tedy musí mazat ručně.

V poslední složce **img** je uložen titulní obrázek MAT Server.

Soubor **Default.aspx** obsahuje zobrazení úvodní stránky aplikace. Je zde obsažen titulní obrázek MAT Server, modul (popsaný v kapitole 7) a style.css.

Soubory **podm_cz.htm** a **podm_en.htm** obsahují podmínky využití webové aplikace v českém a anglickém jazyce.

Další soubor **web.config** je konfigurační soubor pro webový server a ASP.NET web aplikace.

Posledním souborem je **WebUserControl.ascx**, který popisuje jednotlivé moduly webové aplikace. V části `WebUserControl.ascx` je popsáno zobrazení stránek a využívá k tomu ovládací prvek `MultiView`. Tento prvek obsahuje podprvky `View`, které jsou číslovány od 0 po 8. Existuje tedy 8 listů (modulů), kterými si uživatelé musí projít, aby se dostali k poslednímu (k výpisu, viz kapitola 7.9). Mezi moduly se prochází pomocí prvku `ActiveViewIndex`, které nabývá výše zmíněných hodnot 0 – 8, ale používá se až v programové části `WebUserControl.ascx.cs`. V části `WebUserControl.ascx.cs` je jeho programovací kód, kde je vytvořena výpočetní část aplikace. Byly zde vytvořeny třídy např. pro vynulování všech proměnných, převod mezi typy proměnných (`MWArray` na `double`) a jiné. Program funguje tak, že při prvním načtení, se provede třída `Page_Load`, kde nastaví proměnné a danému uživateli přiřadí unikátní ID. Před vykreslením stránky se provede třída `Page_PreRender`, která zviditelňuje potřebná tlačítka zobrazené stránky a podle stisknutého tlačítka CZ / EN se nastaví i potřebný jazyk. Tlačítka Zpět a Pokračovat provádí funkci podle aktuální hodnoty `ActiveViewIndex`. Tlačítko Domů vynuluje proměnné a vrací se na úvodní stránku (modul 0). Tlačítkem Nápověda se zobrazí nápověda webové aplikace. Kvůli délce souborů jsou programy uloženy v přílohách.

6 ZABEZPEČENÍ APLIKACE

Zabezpečení webové aplikace zajišťuje ověřování a autorizace spolu se zabezpečenou komunikací. Kapitola zabezpečení je popsána pomocí zdrojů [10] a [18].

Obecně se webové aplikace zabezpečují třemi následujícími způsoby:

Pojem **ověřování** znamená identifikace klientů (uživatelů) aplikace. Na webové aplikaci jsou uživatelé ověřováni podle jména a hesla. Tento požadavek ověřování je zpracován aplikačním serverem a databázovým serverem.

Autorizace uživatele určuje práva, které opravňují uživatele provádět změny a i prostředky, které jsou zpřístupněné k použití těchto změn. Tyto prostředky mohou být např. soubory, databáze, klíče registru a jiné.

Zabezpečená **komunikace** mezi serverem a webovým prohlížečem má dvě vlastnosti. Utajení – je použito šifrování, kdyby data byla odcizena. Integrita – zajišťují kódy MAC (Message Authentication Codes), které chrání data při úmyslnými nebo náhodnými úpravami dat při komunikaci.

Zabezpečení ověřováním a autorizací ve vytvořené aplikaci není potřeba, protože aplikace nepracuje s žádnou databází, ke které by se aplikace připojovala. Bezpečnou komunikaci zajišťuje již vytvořený server - MAT Server, který už nastavil kolega Ing. David Rakus [15], na kterém běží operační systém Microsoft Windows Web Server 2008.

Ale pro bezchybnou funkčnost aplikace byly použity pro ověřování platnosti vstupu validační ovládací prvky.

Tyto prvky se používají spolu s AJAXem (kapitola 2.3) a jsou implementovány na prvky TextBox. Existuje několik typů validátorů, ale podrobně budou popsány pouze ty, které se využívají v aplikaci.

RequiredFieldValidator kontroluje, zda ovládací prvek má nějakou hodnotu nebo je prázdný (Obr. 22.).



Obr. 22. RequiredFieldValidator

RangeValidator má za úkol kontrolovat, jestli je zadaná hodnota v nastaveném rozsahu (Obr. 23.). Nastavení proměnných je např. typu string, integer, double a nastavuje se dolní a horní mez.



103 Zadejte kladnou hodnotu od 0 do 100.

Obr. 23. RangeValidator

RegularExpressionValidatoru nastaví určitý specifický výraz a porovnává tuto hodnotu se vstupní hodnotou (Obr. 24.).



[@!] Špatně jste zadali čísel. Zadejte čísel ve tvaru např.: [1 2]

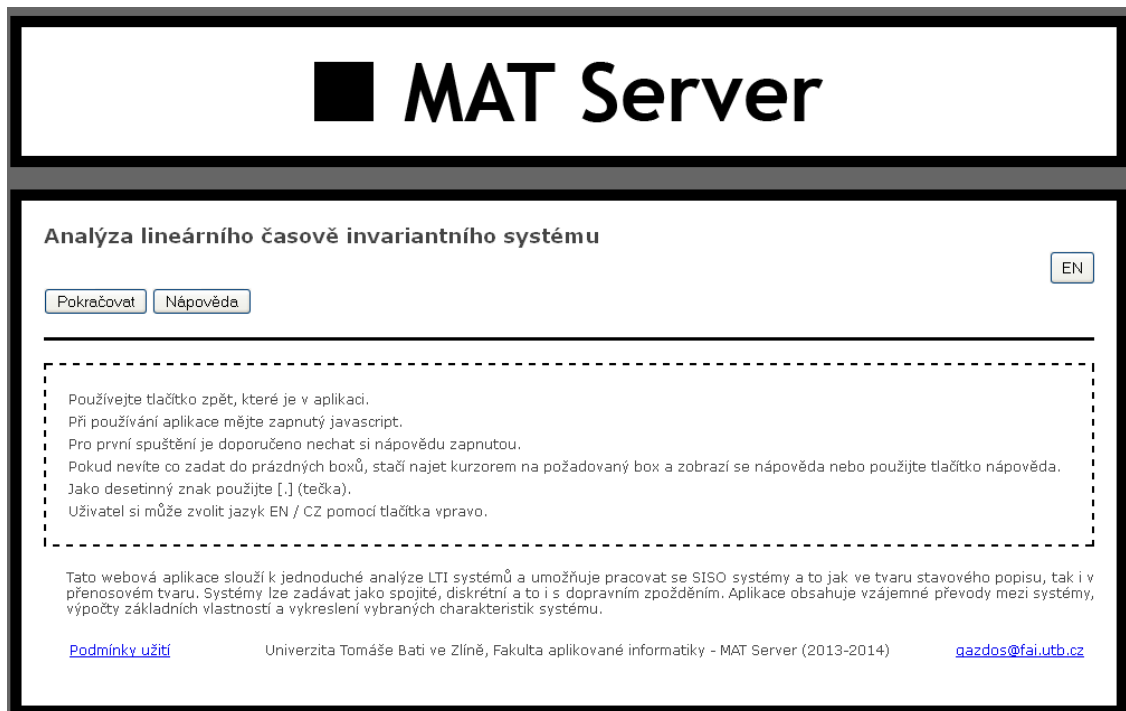
Obr. 24. RegularExpressionValidator

Další prvky jsou popsány v [7].

Kdyby se vyskytla neošetřená chyba, např. některá funkce z MATLABu neumí pracovat se zadaným systémem pomocí použitých metod (musela by se změnit metoda pro výpočet v MATLABu), tak za pomoci příkazu `try – catch` v programovacím jazyku C# se vrátí uživatel zpět na úvodní stránku a uživatel je o tom informován. Vše zadané uživatelem se smaže a musí tedy začít od začátku.

7 MODULY WEBOVÉ APLIKACE

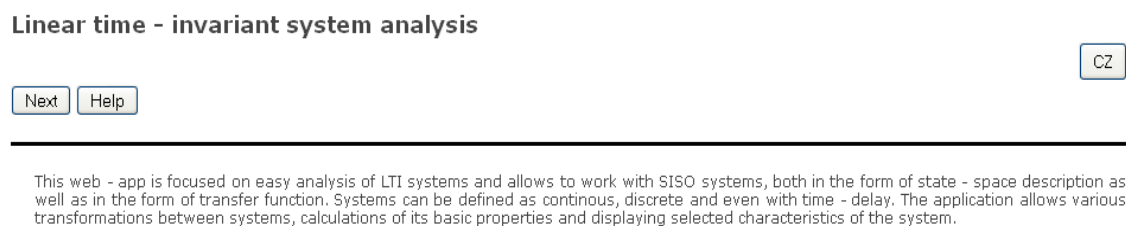
Pomocí ovládacího prvku MultiView byly vytvořeny moduly s označením View0 až View8. Každý modul bude popsán v následujících kapitolách. Na obr. 25. je úvodní stránka s otevřenou nápovědou. Nápověda se nachází v každém modulu a obsahuje informace k používání aktuálně spuštěného modulu. Kvůli omezené velikosti stránek, budou na dalších obrázcích pouze vyřiznuté části.



Obr. 25. Modul - úvodní strana s nápovědou

7.1 Úvodní strana

Obr. 26. znázorňuje úvodní stránku v anglickém jazyce. Je zde stručný popis webové aplikace, k čemu slouží a s čím umí pracovat.



Obr. 26. Modul - úvodní strana v anglickém jazyce

7.2 Popis systému

Další modul obsahuje popis systému. Systém lze zadat buď pomocí přenosového tvaru, nebo stavového popisu (viz. Kapitola 4.1). Pro diskrétní systémy zde lze zadat periodu (0; 100>.

V této aplikaci lze počítat i s dopravním zpožděním a to z intervalu (0; 100>. Při diskrétních systémech program MATLAB pracuje s dopravním zpožděním jen jako s celým číslem - násobkem vzorkovací periody. Ale jelikož MATLAB stejně zaokrouhlí dopravní zpoždění na celé číslo, je zde vytvořená podmínka pro zadávání dopravního zpoždění u diskrétních systémů jen pro celá čísla.

Pokud je požadován spojitý tvar nebo systém bez dopravního zpoždění, nemusí se zadávat žádná čísla nebo uživatel zadá 0.

Pokud uživatel neví co zadat nebo v jakém tvaru zadat např. periodu vzorkování do příslušného textového pole, stačí najet kurzorem myši na toto pole a zobrazí se tzv. nápověda pomocí funkce *tooltip* (Obr. 27.) nebo může využít tlačítko Nápověda. Funkce *tooltip* je zavedena u každého textového pole ve webové aplikaci.

Analýza lineárního časově invariantního systému

EN

[Domů](#) [Zpět](#) [Pokračovat](#) [Nápověda](#)

Zvolte v jakém tvaru budete zadávat systém:

Zadejte periodu vzorkování:

Tv

Pokud chcete diskrétní systém, zadejte vzorkovací periodu např.: 2,5

Zadejte dopravní zpoždění:

Td

Obr. 27. Modul - popis systému

V programové části se v tomto modulu nastaví proměnná, ve které je zapsáno v jakém tvaru se bude systém zapisovat a do dalších proměnných se uloží vzorkovací perioda a dopravní zpoždění.

Všechny příkazy daných modulů se provádí vždy po stisknutí tlačítka Pokračovat.

7.3 Zadávání systému

Jestliže uživatel zvolí tvar, ve kterém chce zadat systém, otevře se jeden z následujících modulů. Pro stavový popis (Obr. 28) se zadává systém ve tvaru 4 matic A , B , C , D . Formát zadávání je obdobný jako v programu MATLAB. Pokud uživatel neví, může opět využít tlačítko Nápověda nebo najet kurzorem myši na dané textové pole. Aby program mohl jít dále, musí uživatel zadat správné rozměry matic pro vytvoření daného systému.

Analýza lineárního časově invariantního systému

EN

Domů Zpět Pokračovat Nápověda

Zadejte matice pro stavový popis:

Matice A: [1]

Matice B: [2 1]

Matice C: [1]

Matice D: [0]

Matice mají špatný rozměr. Rozměry matic musí být: $A = n \times n$; $B = n \times m$; $C = r \times n$; $D = r \times m$.

Obr. 28. Modul - zadávání systému pomocí stavového popisu

Při zvolení formátu zadávání přenosového tvaru (Obr. 29.), uživatel opět zadává vstupní parametry jako v programu MATLAB. V hranatých závorkách zapíše čísla, kde první číslo znamená koeficient u nejvyšší mocniny. Pokud uživatel zadá např. [1 2], výsledný polynom bude pak ve tvaru: $s + 2$.

Analýza lineárního časově invariantního systému

EN

Domů Zpět Pokračovat Nápověda

Zadejte číselník a jmenovatel pro přenosový tvar:

Číselník přenosu:

Jmenovatel přenosu:

Obr. 29. Modul - zadávání systému pomocí přenosového tvaru

Tento modul je v programové části rozdělen podle typu zadávání tvaru. Zkontrolují se zadané proměnné, jestli mají správný rozměr, mají správný tvar atd. Pokud jsou vstupní data správná, zkontroluje se, zda zadaný systém není MIMO nebo fyzikálně

nerealizovatelný. Poté se podle vzorkovací periody (spojité / diskrétní) nechají vypsat data na další modul a systém se převede do tzv. minimální realizace, ale pouze u spojitého systému. Jestli je systém MIMO nebo fyzikálně nerealizovatelný, program přejde na modul zkouška. Pokud bude systém diskrétní, skočí na modul převzorkování systému a jestliže bude spojitý, skočí až na modul převody systému.

Pro výpis systému není potřeba tříd z MATLABu, využívá se jen potřebných vytvořených tříd pro zobrazení přenosového tvaru v polynomu. První využitá třída je pro minimální realizaci spojitého systému. Využívá se zde funkce *minreal*. Uživatel může zadat do aplikace desetinná čísla libovolných rozměrů. Ale aby byl zachovaný formát číslic pro 4 platná desetinná místa, využívá se zde i vytvořená třída pro zaokrouhlování.

7.4 Zkouška

U zadávání přenosového tvaru nemůže dojít k zadání MIMO systémů, protože lze zadat pouze jeden přenos. Pokud ale uživatel zadá ve stavovém popisu MIMO systém, zobrazí se modul zkouška (Obr. 30). Webová aplikace je ale navrhnutá pouze pro SISO systémy a uživatel má pouze možnost vrátit se na začátek pomocí tlačítka Domů a zadat systém znovu.

Analýza lineárního časově invariantního systému

[EN](#)[Domů](#) [Nápověda](#)

Systém je MIMO, tento program je výhradně pro SISO systémy.

Obr. 30. Modul - systém je MIMO

Kromě MIMO systémů, může nastat i případ fyzikálně nerealizovatelného systému. Tedy další možné použití funkcí, jako např. vykreslení charakteristik potom nelze využít. Tak jako v předchozím případě je uživatel upozorněn (viz Obr. 31) a může se vrátit na začátek tlačítkem Domů a zadat systém znovu.

Analýza lineárního časově invariantního systému

EN

Domů Nápověda

System je fyzikálně nerealizovatelný (čitatel má více členů než jmenovatel).

Obr. 31. Modul - systém je fyzikálně nerealizovatelný

7.5 Převzorkování systému

Modul pro převzorkování systému (Obr. 32) se zobrazí pouze, jestli je systém zadán jako diskrétní. Spojitý systém nemůže být převzorkován, pouze změněn na diskrétní tvar pomocí vzorkování systému a to lze až v dalším modulu (kapitola 7.6). Na začátek modulu se vždy zobrazí, co uživatel zadal. Jaký přenos nebo stavový popis, zdali je diskrétní, tak i vzorkovací periodu a pokud má systém i dopravní zpoždění. Pod tento výpis se zobrazí textové pole pro novou vzorkovací periodu. Pokud tedy uživatel chce převzorkovat, může zadat novou periodu opět v rozsahu (0, 100>. Jestliže si uživatel nepřeje převzorkovávat, nemusí zadávat nic a pokračuje tlačítkem Pokračovat na další modul.

Analýza lineárního časově invariantního systému

EN

Domů Zpět Pokračovat Nápověda

Zadali jste systém v přenosovém tvaru:

$$G = \frac{z + 2}{z^2 + 0.5z + 0.025}$$

Td = 1

Tv = 3

Pokud chcete převzorkovat, zadejte novou periodu vzorkování:

Tv

Obr. 32. Modul – převzorkování systému

Modul v programové části zjišťuje, jestli byla zadaná nová vzorkovací perioda. Pokud ano, využívá se třída pro převzorkování s použitím funkce MATLABu *d2d* a převzorkuje daný systém. Kvůli dalšímu zobrazení v modulu převody systému se systém zkouší převést do

tzv. minimální realizace. Po stisknutí tlačítka Pokračovat přejde program do modulu převody systému.

U převádění systémů ze spojitých na diskrétních, z diskrétních na spojitých nebo převzorkování diskrétních systémů se využívá funkcí MATLABu, jako jsou $c2d$, $d2c$ a $d2d$. Tyto funkce pracují s metodou foh a zoh . Tyto metody neumí pracovat se systémy, které mají póly a nuly blízko 0. Proto byla u těchto funkcí zvolena metoda *tustin* (viz. [8] [16]).

7.6 Převody systému

V tomto modulu se opět jako první vypíše zadání uživatele. V tomto modulu je přidána funkce minimální realizace systému. O tom jestli zadaný systém je nebo není v minimální realizaci, se může přesvědčit zaškrtnutím boxu. Pokud je, vypíše se pouze hláška, že systém už nelze více minimalizovat. Ale pokud není, vypíše minimalizovaný systém. S minimálním realizovatelným systémem se už nadále nepracuje. Pokud uživatel chce s tímto systémem pracovat, musí jít na začátek a zadat ho znovu.

Poslední a hlavní funkcí tohoto modulu jsou převody mezi systémy. Počáteční nastavení převodu je zaškrtnutí políčka pro žádný převod. Na obr. 33. je zadán spojitý systém v převodovém tvaru. To znamená, že systém lze převést na 3 různé druhy a to: diskrétní přenosový tvar, spojitý stavový popis a diskrétní stavový popis. Pokud uživatel zvolí převod na diskrétní stavový popis, musí zadat i vzorkovací periodu, jinak nelze pokračovat. Jestliže máme zadaný spojitý systém s dopravním zpožděním, např. 3,1. A chceme tento systém převést na nějaký diskrétní systém, zobrazí se textové pole nejen pro vzorkovací periodu, ale i pro změnu dopravního zpoždění. Jak bylo psáno výše, funkce v MATLABu pracuje u diskrétních systémů s dopravním zpožděním pouze s celými čísly - násobkem vzorkovací periody. Zde je taktéž daná podmínka pouze pro celá čísla.

Tento modul nabízí až 12 možných převodů a to 3 pro každý zadaný systém. Zadané systémy mohou být: spojitý přenosový tvar a stavový popis nebo diskrétní přenosový tvar a stavový popis.

Pokud zaškrtneme políčko pro Minimalizaci systému, vypíše se již minimalizovaný systém diskrétního tvaru nebo převzorkovaného diskrétního tvaru z předchozího modulu. Jestliže systém byl zadán jako spojitý, vypíše se minimalizovaný systém, který byl proveden už v modulu zadávání systému (kapitola 7.3). Hlavním úkolem tohoto modulu ale je, že

obstarává až 12 převodů mezi systémy. O zadávání na příslušný systém se stará RadioButtonList, kde si uživatel vybere mezi možnými převody systému. Jestli je potřeba zadat vzorkovací periodu nebo i nové dopravné zpoždění, technologie AJAX vykreslí příslušná textová pole.

U modulu převody systémů se využívá nejvíce tříd vytvořených v MATLABu. Nejhlavnější použité funkce jsou *c2d* a *d2c* s metodou *tustin*. Dále se zde používá i vytvořená třída pro vykreslování přenosového tvaru v polynomu nebo převedení proměnné z MATLABu typu MWArray na typ v programovacím jazyku C# double.

Zadali jste systém v přenosovém tvaru:

$$G = \frac{2*s + 4}{2*s^2 + s + 0.05}$$

Td = 1

Minimalizace systému:

$$G = \frac{s + 2}{s^2 + 0.5*s + 0.025}$$

Td = 1

- Převod ze spojitého přenosového tvaru na diskrétní přenosový tvar:
- Převod z spojitého přenosového tvaru na spojitý stavový popis:
- Převod ze spojitého přenosového tvaru na diskrétní stavový popis:
- Žádný převod:

Tv

Obr. 33. Modul – převody systému

7.7 Zadávání vlastností

Modul pro zadávání vlastností (Obr. 34) umožňuje zjistit ze systému až 8 vlastností. Na začátku se vypíše, jestli uživatel v předchozím modulu zadal nějaký převod. Pod tímto převodem už lze zadat pomocí zaškrtování polí vlastnosti, které lze získat. Tyto vlastnosti jsou popsány v kapitole 4.3.

U toho modulu se v programovací části starají CheckBoxy zda byl některý z nich zaškrtnut. Pokud nebyl v předchozím modulu zaškrtnut nějaký převod uživatelem, tak se v tomto modulu nemá co vypisovat a stará se pouze o výpočet vlastností do dalšího modulu.

Vytvořené třídy používají různé funkce z MATLABu, jako např. *zero*, *pole*, *order*, *isstable*, *gain* a jiné.

Spojité přenosový tvar na diskrétní stavový popis:

Matice A: [1.5821 -0.602; 1 0]

Matice B: [2; 0]

Matice C: [1.0277 -0.2396]

Matice D: [0.796]

Td = 1

Tv = 1

Výpočet vlastností:

Nuly:

Póly:

Řád:

Stabilita:

Zesílení:

Řiditelnost / dosažitelnost stavu:

Řiditelnost výstupu:

Rekonstruovatelnost / pozorovatelnost stavu:

Obr. 34. Modul – zadávání vlastností

7.8 Volba grafických charakteristik

Předposlední modul nabízí grafické zobrazení vlastností systému. Na začátku se vypíše výpočet zadaných vlastností z minulého modulu, pokud uživatel nějaké zadal. A dále jde vidět na obr. 35., že lze zvolit vykreslení až 5 grafů. Tyto grafy jsou popsány v kapitole 4.6.

Část programu u tohoto modulu je obdobná jako v předchozí části. Modul taktéž obsahuje CheckBoxy.

Pro vykreslování grafů se využívají třídy s funkcemi, jako jsou např.: *step*, *impulse*, *bode* a *nyquist*. Pro vykreslení pozic pólů a nul existuje funkce *pzmap* nebo *iopzmap*. Tyto funkce ale při každém výpisu ve webové aplikaci vypsaly chybu. Proto byly použity výpočty pólů a nul a dále vykresleny s použitím funkce *plot*. Všechny grafy jsou uloženy ve formátu .png pro jejich malou velikost souboru a dále jsou použity příkazy *ImageUrl* pro vykreslení uloženého obrázku.

Analýza lineárního časově invariantního systému

EN

Domů

Zpět

Pokračovat

Nápověda

Nuly: -1.0001 ; 1.0050e-05

Póly: 0.9452 ; 0.6369

Řád: 2

Systém je stabilní.

Zesílení: 80.0020

Systém je úplně říditelný / dosažitelný.

Výstup je říditelný.

Systém je úplně rekonstruovatelný / pozorovatelný.

Vykreslení grafů:Přechodová charakteristika: Impulsní charakteristika: Amplitudově - fázová frekvenční charakteristika v log. souřadnicích: Frekvenční charakteristika v komplexní rovině: Zobrazení nul a pólů v komplexní rovině:

Obr. 35. Modul – zadávání grafických charakteristik

7.9 Výpis

Poslední modul vypíše a vykreslí vše, co uživatel zadal, nechal vypočítat nebo vykreslit (viz Obr. 36). Na začátku se zobrazí zadávaný systém. Pokud by uživatel zadal diskrétní systém a nechal ho znovu převzorkovat, vypsalo by se nový převzorkovaný systém. Další vypsaný systém je pro převod mezi tvary, např. spojitý přenosový tvar na diskrétní stavový popis. Aplikace dále vypíše zvolené vypočítané vlastnosti a vykreslí grafy.

K výpočtu vlastností a vykreslení charakteristik se používá vždy poslední systém, který se vypočítal, kromě minimalizovaného systému. Jestliže chce uživatel např. nechat vykreslit zadaný systém, nesmí ho převádět na jiný. Webová aplikace tedy dovoluje projít všemi

moduly pouze se zadaným systémem na začátku bez jeho změny k výpočtu vlastností, nebo k vykreslování grafů. Ale také dovoluje výpočet vlastností nebo vykreslování grafů změněného systému v jakémkoliv modulu. Záleží tedy na uživateli, jaký systém potřebuje k výpočtu vlastností nebo k vykreslování grafů.

Pokud uživatel v jakémkoliv modulu napíše nebo zaškrtně něco, co nechtěl, může se pomocí tlačítka Zpět vrátit a změnit svoje původní zadání.

Zadali jste:

$$G = \frac{2*s + 4}{2*s^2 + s + 0.05}$$

Td = 1

Spojité přenosové tvar na diskretní stavový popis:

Matice A: [1.5821 -0.602;1 0]

Matice B: [2;0]

Matice C: [1.0277 -0.2396]

Matice D: [0.796]

Td = 1

Tv = 1

Nuly: -1.0001 ; 1.0050e-05

Póly: 0.9452 ; 0.6369

Řád: 2

Systém je stabilní.

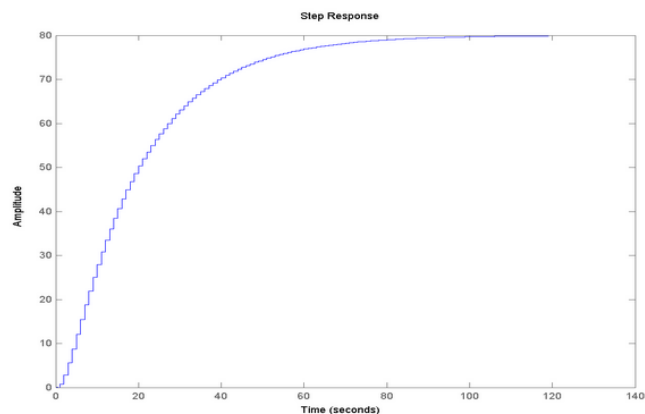
Zesílení: 80.0020

Systém je úplně říditelný / dosažitelný.

Výstup je říditelný.

Systém je úplně rekonstruovatelný / pozorovatelný.

Přechodová charakteristika:

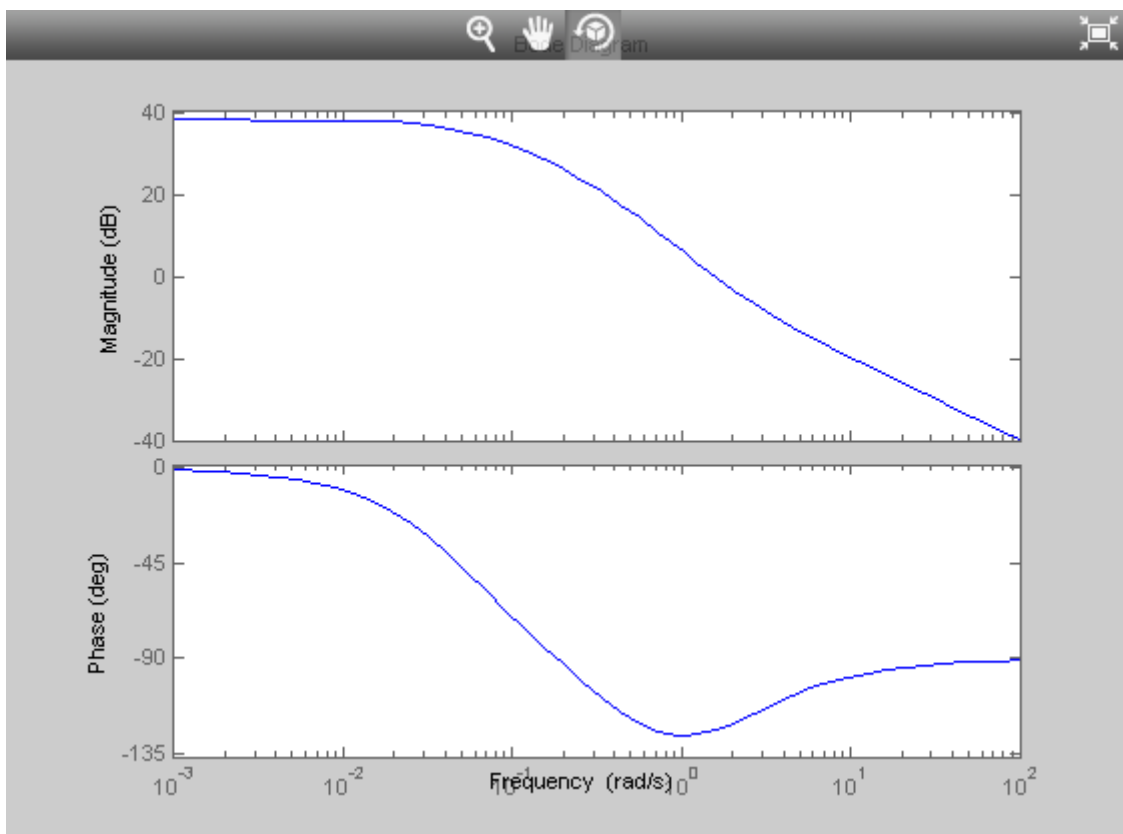


Obr. 36. Modul – výpis

Tento modul je vytvořený pouze pro jakýsi souhrn všech zadaných systémů, převedených systémů, výpočet vlastností a taky pro vykreslení grafů z minulého modulu. Pro uložení grafu nebo jeho vykreslení zvlášť na webový prohlížeč stačí kliknout na požadovaný graf.

Existuje také funkce z MATLABu, která dovoluje vykreslit graf vytvořený v prostředí MATLAB tak, aby se vykreslil ve webovém prohlížeči v interaktivní podobě (Obr. 37.). Vzhledem k možnostem otáčení a přibližování grafů přímo ve webovém prohlížeči je vhodná hlavně pro 3D grafy. Ale možnost zbytečně nevytvářet obrázky typu .png na server a následně je mazat by se dala využít i u 2D grafů. Grafy vypadají lépe a díky možnosti několika operací s nimi dávají lepší dojem z aplikace. Bohužel se nepodařilo nastavit server na tuto funkci a nebyla tedy použita v aplikaci.

Pro další informace o těchto funkcích v prostředí MATLAB najdete v [8] nebo [16] pod názvem WebFigure.



Obr. 37. Funkce WebFigure

ZÁVĚR

Hlavním cílem této diplomové práce bylo navrhnout a realizovat webovou aplikaci pro analýzu LTI systémů za použití frameworku Microsoft .NET a matematického prostředí MATLAB. Aplikace je dostupná na webové adrese <http://matserver.utb.cz/LTI/>.

Teoretická část obsahuje rešerši dříve zpracovaných prací k tomuto nebo podobnému tématu. Nejvíce obsáhlou knihou v českém jazyce k tomuto tématu je [2]. Obsahuje velkou část informací pro skupinu lidí, zabývajících se automatickým řízením jako vhodný teoretický základ do této oblasti.

V teoretické části je dále obsažen popis použitého frameworku a programovacích jazyků, které se využívají ve výsledné aplikaci. Malá kapitola je zde věnována i vědecko – technickému prostředí MATLAB, jeho širokým možnostem a použitému toolboxu k vytvoření aplikace. Poslední kapitola je věnována lineárním časově invariantním systémům. Kapitola obsahuje hlavní druhy popisy systémů, se kterými se můžeme setkat. Okrajově se zabývá použitím L a Z – transformací a různými vlastnostmi systémů. Malá část je věnována i grafickým charakteristikám.

V praktické části se seznamujeme s vytvořenou webovou aplikací. Na začátku je kapitola věnována struktuře a tedy vývoji webové aplikace. Tato struktura aplikace je rozdělena na třídy vytvořené v programu MATLAB a samotnou aplikaci, která tyto třídy využívá k analýze LTI systémů. Aby webová aplikace nemusela dlouho vypočítávat speciální výpočty pro LTI systémy, jsou použity speciální funkce z programu MATLAB. Tyto funkce jsou poté rozděleny do tříd a následně překompilovány pomocí toolboxu Builder NE pro další použití jinými programovacími jazyky. V samotné aplikaci se používá několik jazyků pro výpočetní část a grafické zobrazení výsledků. Pro zobrazení aplikace se využívají tzv. moduly, které pracují jako několik listů a zobrazí se vždy jen potřebný modul.

Další kapitola se věnuje zabezpečení aplikace. Je to krátký popis zabezpečení, které by tyto aplikace měli mít, obzvlášť, jestli využívají nějaký databázový systém, jako např. SQL. Vytvořená webová aplikace nevyužívá žádný takový systém, protože je tvořena speciálními matematickými funkcemi, které potřebují pouze vstupní hodnoty. Hlavní zabezpečení je tedy věnováno vstupům, které je realizováno pomocí validátorů. Ty zabezpečují striktně omezené zadávání, aby vstupní proměnné nenarušily chod aplikace.

Ostatní zabezpečovací prvky, jako např. bezpečný komunikační kanál má na starosti server.

Poslední část praktické práce se věnuje podrobnému popisu webové aplikace. Každý modul je popsán, co uživatel může zadat a jaké všechny možnosti umožňuje tato webová aplikace.

Během programování webové aplikace pro analýzu lineárních časově invariantních systémů se vyskytlo mnoho problémů, ale taky různé možnosti pro vytvoření webové aplikace. Aplikace pro návrh LTI systémů je tvořena především pro pedagogické účely a to zejména pro studenty, kteří nemají možnost pracovat doma se systémem MATLAB nebo podobnými matematickými programy. Aplikace byla testovaná a porovnávaná s programem MATLAB. Pro zadaná vstupní data, webová aplikace i MATLAB vypisují stejná výstupní data a vykreslují stejné charakteristiky systémů.

Tato diplomová práce může tak sloužit jako zdroj informací pro další podobně zaměřené webové aplikace.

SEZNAM POUŽITÉ LITERATURY

- [1] Analýza systému. *Vysoká škola báňská* [online]. 2012 [cit. 2014-05-15]. Dostupné z: <http://winf230e-1.vsb.cz/crl002/Main.aspx>
- [2] BALÁTĚ, Jaroslav. *Automatické řízení*. Praha: BEN - technická literatura, 2004. ISBN 80-7300-148-9.
- [3] DOSTÁL, Petr a František GAZDOŠ. *Řízení technologických procesů*. Zlín: Univerzita Tomáše Bati ve Zlíně, Fakulta aplikované informatiky, 2006. ISBN 80-7318-465-6.
- [4] DOSTÁL, Petr a RADEK MATUŠŮ. *Stavová a algebraická teorie řízení*. Zlín: Univerzita Tomáše Bati ve Zlíně, Fakulta aplikované informatiky, 2010. ISBN 978-80-7318-991-4.
- [5] HALBSGUT. *Realizace jednorozměrových regulátorů v Octave*. Zlín, 2007. Diplomová práce. Univerzita Tomáše Bati ve Zlíně, Fakulta aplikované informatiky.
- [6] KARBAN, Pavel. *Výpočty a simulace v programech Matlab a Simulink*. Brno: Computer Press, 2006. ISBN 80-251-1301-9.
- [7] MACDONALD, Matthew, Mario SZPUSZTA. *ASP.NET 2.0 a C#: tvorba dynamických stránek PROFESIONÁLNĚ*. Brno: Zoner Press, 2006. ISBN 80-86815-38-2.
- [8] MATLAB – Product Help. Návod systému MATLAB R2008b.
- [9] MATELOVÁ. *Stavová teorie lineárního řízení*. Zlín, 2013. Diplomová práce. Univerzita Tomáše Bati ve Zlíně, Fakulta aplikované informatiky.
- [10] MICROSOFT. *Vytváříme zabezpečené aplikace v Microsoft ASP.NET*. Brno: Computer Press, 2004. ISBN 80-251-0466-4.
- [11] MSDN: Microsoft Developer Network [online]. 2014 [cit. 2014-05-15]. Dostupné z: <http://msdn.microsoft.com>
- [12] NASH, Trey. *C# 2010: Rychlý průvodce novinkami a nejlepšími postupy*. Brno: Computer Press, 2010. ISBN 978-80-251-3034-6.
- [13] ODELL, Den. *JavaScript: Průvodce programováním ajaxových aplikací*. Brno: Computer Press, 2010. ISBN 978-80-251-2733-9.

- [14] PÍSEK, Slavoj. *HTML: začínáme programovat*. Praha: Grada Publishing, 2014. ISBN 978-80-247-5059-0.
- [15] RAKUS, David. *Web-aplikace pro přímý návrh a ladění regulátorů z experimentálních dat*. Zlín, 2009. Diplomová práce. Univerzita Tomáše Bati ve Zlíně, Fakulta aplikované informatiky.
- [16] THE MATHWORKS, Inc. *Control System Toolbox: User's Guide*. Natick, MA, USA, 2013.
- [17] VACULÍKOVÁ, Martina. *Elektronický manuál pro program SCILAB*. Zlín, 2006. Bakalářská práce. Univerzita Tomáše Bati ve Zlíně, Fakulta aplikované informatiky.
- [18] ZEMEK, Lukáš. *Bezpečnost webových aplikací*. Praha, 2012. Bakalářská práce. Univerzita Tomáše Bati ve Zlíně, Fakulta aplikované informatiky.

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

AJAX	Asynchronous JavaScript and XML (asynchronní JavaScript a XML)
ASP	Active Server Pages
BIBO	Bounded – Input Bounded – Output (omezený vstup, omezený výstup)
CLR	Common Language Runtime (obecný jazykový převaděč)
CSS	Cascading Style Sheets (kaskádové styly)
GUI	Graphical User Interface (grafické uživatelské rozhraní)
HTML	HyperText Markup Language (značkový jazyk pro hypertext)
LTI	Linear time – invariant system (lineární časově invariantní systém)
MAC	Message Authentication Code
MCR	MATLAB Compiler Runtime
MIMO	Multiple input - multiple output (více vstupů – více výstupů)
PHP	Hypertext Preprocessor (hypertextový preprocesor)
SISO	Single input - single output (jeden vstup – jeden výstup)
SQL	Structured Query Language (strukturovaný dotazovací jazyk)
XML	eXtensible Markup Language (rozšiřitelný značkovací jazyk)

SEZNAM OBRÁZKŮ

<i>Obr. 1. Rozšíření MATLABu</i>	17
<i>Obr. 2. Producty MATLABu</i>	18
<i>Obr. 3 Jednorozměrný lineární dynamický systém</i>	19
<i>Obr. 4. Postup výpočtu při L – transformaci</i>	21
<i>Obr. 5. Póly v Gaussově rovině</i>	24
<i>Obr. 6. Dosažitelnost stavu</i>	25
<i>Obr. 7. Řiditelnost stavu</i>	25
<i>Obr. 8. Řiditelnost výstupu</i>	26
<i>Obr. 9. Pozorovatelnost stavu</i>	27
<i>Obr. 10. Rekonstruovatelnost stavu</i>	28
<i>Obr. 11. Dopravní zpoždění systému</i>	29
<i>Obr. 12. Jednotkový skok</i>	30
<i>Obr. 13. Přejchodová charakteristika</i>	31
<i>Obr. 14. Jednotkový impuls</i>	32
<i>Obr. 15. Impulsní charakteristika</i>	32
<i>Obr. 16. Zobrazení bodu frekvenčního přenosu v komplexní rovině</i>	34
<i>Obr. 17. Frekvenční charakteristika v komplexní rovině</i>	34
<i>Obr. 18. Amplitudově - fázová charakteristika v log. souřadnicích</i>	35
<i>Obr. 19. Rozložení nul a pólů v komplexní rovině (x – póly, o - nuly)</i>	36
<i>Obr. 20. Převod spojitého systému na diskrétní pro přenosový tvar</i>	39
<i>Obr. 21. Souborová struktura</i>	40
<i>Obr. 22. RequiredFieldValidator</i>	42
<i>Obr. 23. RangeValidator</i>	43
<i>Obr. 24. RegularExpressionValidator</i>	43
<i>Obr. 25. Modul - úvodní strana s nápovědou</i>	44
<i>Obr. 26. Modul - úvodní strana v anglickém jazyce</i>	44
<i>Obr. 27. Modul - popis systému</i>	45
<i>Obr. 28. Modul - zadávání systému pomocí stavového popisu</i>	46
<i>Obr. 29. Modul - zadávání systému pomocí přenosového tvaru</i>	46
<i>Obr. 30. Modul - systém je MIMO</i>	47
<i>Obr. 31. Modul - systém je fyzikálně nerealizovatelný</i>	48
<i>Obr. 32. Modul – převzorkování systému</i>	48

<i>Obr. 33. Modul – převody systému</i>	<i>50</i>
<i>Obr. 34. Modul – zadávání vlastností.....</i>	<i>51</i>
<i>Obr. 35. Modul – zadávání grafických charakteristik</i>	<i>52</i>
<i>Obr. 36. Modul – výpis</i>	<i>53</i>
<i>Obr. 37. Funkce WebFigure</i>	<i>54</i>

SEZNAM PŘÍLOH

P I Obsah CD

Další přílohy jsou kvůli velikosti na přiloženém CD:

P II Obsah knihovny LTI















P III M – soubory

P IV Soubory .NET

P V Ostatní soubory

PŘÍLOHA P I: OBSAH CD

V příloženém CD jsou následující složky:

- [-]  Diplomová práce
 -  doc
 -  docx
 -  pdf
- [+]  Knihovna LTI
- [-]  M - soubory
 -  graphs
 -  properties
 -  test
 -  transfer
- [-]  Přílohy
 -  doc
 -  docx
 -  pdf