SLOVENSKÁ TECHNICKÁ UNIVERZITA V BRATISLAVE
FAKULTA CHEMICKEJ A POTRAVINÁRSKEJ TECHNOLÓGIE

EVIDENČNÉ ČÍSLO: FCHPT-5414-43114

RIADENIE SÚSTAVY VOZIDIEL

DIPLOMOVÁ PRÁCA

2018 **Bc. Milan Horňák**

2018 **Bc. Milan Horňák**

SLOVENSKÁ TECHNICKÁ UNIVERZITA V BRATISLAVE
FAKULTA CHEMICKEJ A POTRAVINÁRSKEJ TECHNOLÓGIE

Evidenčné číslo: FCHPT-5414-43114

RIADENIE SÚSTAVY VOZIDIEL

DIPLOMOVÁ PRÁCA

Študijný program: automatizácia a informatizácia v chémii a potravinárstve
Študijný odbor: 5.2.14. automatizácia
Školiace pracovisko: Ústav informatizácie, automatizácie a matematiky
Vedúci záverečnej práce: Ing. Juraj Holaza, PhD.

2018 **Bc. Milan Horňák**

Slovenská technická univerzita v Bratislave
Ústav informatizácie, automatizácie a matematiky

Fakulta chemickej a potravinárskej technológie
Akademický rok: 2017/2018
Evidenčné číslo: FCHPT-5414-43114

ZADANIE DIPLOMOVEJ PRÁCE

Študent: **Bc. Milan Horňák**

ID študenta: 43114

Študijný program: automatizácia a informatizácia v chémii a potravinárstve

Študijný odbor: 5.2.14. automatizácia

Vedúci práce: Ing. Juraj Holaza, PhD.

Miesto vypracovania: Bratislava

Názov práce: **Riadenie sústavy vozidiel**

Jazyk, v ktorom sa práca vypracuje: slovenský jazyk

Špecifikácia zadania:

Cieľom práce je navrhnúť autonómne riadiace algoritmy pre robotické systémy, za účelom zabezpečenia bezpečnej a efektívnej premávky. Uvažovaný systém pozostáva z viacerých robotických vozidiel, ktorých úlohou je sledovanie cesty a dodržiavanie požadovanej vzdialenosti medzi autom a prekážkou pred ním. Práca je zameraná na implementáciu decentralizovaného a centralizovaného riadenia.

Úlohy:

1. Zostrojenie matematického modelu n-vozidiel zoradených za sebou.
2. Návrh centralizovaného a decentralizovaného riadenia.
3. Simulačné overenie navrhnutých riadiacich prístupov.
4. Verifikácia riadiacich algoritmov na reálnom zariadení.

Rozsah práce: 50

Zoznam odbornej literatúry:

1. Kalúz, M. – Holaza, J. – Janeček, F. – Blažek, S. – Kvasnica, M. A Robotic Traffic Simulator for Teaching of Advanced Control Methods. In *IFAC-PapersOnLine* :. s. 338–343.
2. Kvasnica, M. *Real-Time Model Predictive Control via Multi-Parametric Programming:Theory and Tools: Theory and Tools.* Saarbrücken : VDM Verlag Dr. Müller, 2009. 231 s. ISBN 978-3-639-20644-9.

Riešenie zadania práce od: 27. 03. 2018

Dátum odovzdania práce: 13. 05. 2018

Bc. Milan Horňák

študent

prof. Ing. Miroslav Fikar, DrSc

vedúci pracoviska

prof. Ing. Miroslav Fikar, DrSc

garant študijného programu

Poďakovanie

Ďakujem vedúcemu diplomovej práce Ing. Jurajovi Holazovi, PhD. za čas, ktorý mi venoval pri zodpovedaní obrovského množstva otázok spojených s diplomovou prácou, za jeho odborné vedenie, návrhy a pripomienky, bez ktorých by táto práca nevznikla. V neposlednom rade ďakujem mojej rodine a priateľom za podporu počas celého štúdia.

Abstrakt

Táto práca sa zaoberá problematikou riadenia sústavy modelových robotických vozidiel prediktívnym regulátorom s modelom. V práci je uvedená charakterizácia systému vozidiel. Práca popisuje základné aspekty prediktívneho riadenia s modelom a možnosti riadenia systému použitím decentralizovaného a centralizovaného spôsobu riadenia. Sú v nej porovnané priebehy riadenia procesu bežne používanými regulátormi s prediktívnym regulátorom. Súčasťou práce sú grafické závislosti popisujúce priebehy riadenia systému.

Kľúčové slová: robotické vozidlá; MPC; prediktívny regulátor; centralizovaný systém; decentralizovaný systém

Abstract

This paper deals with the problem of control of the robotic vehicles system with the model predictive control. It describes the characteristics of the vehicle system. The thesis describes the basic aspects of model predictive control and differecnes between centralised and distributed way of control. There is a comparison between behaviour of models controlled by commonlly used controllers and model predictive controller. Process control charts representing the robotic vehicles systems control behaviour are also included in the thesis.

Key words: robotic vehicles; MPC; centralised system; distributed system

Obsah

Zoznam ilustrácií 16

Úvod 17

1 Optimalizácia 18

1.1 Konvexná optimalizácia 18

1.1.1 Ohraničenia 19

1.1.2 Účelová funkcia 20

1.1.2.1 Lineárne programovanie 20

1.1.2.2 Kvadratické programovanie 21

1.2 Riešenie optimalizačných problémov 23

2 Modelovanie 25

2.1 Prístupy k modelovaniu 26

2.2 Typy modelov 27

2.2.1 Vstupno-výstupný opis 27

2.2.1.1 Diferenčná rovnica 27

2.2.1.2 Prenos 28

2.2.2 Stavový opis 29

2.3 Stabilita 29

2.4 Pozorovače 31

2.4.1 Pozorovač stavov 32

2.4.1.1 Modelovanie poruchy systému 33

3 Riadiace algoritmy 34

3.1 PID 34

3.2 Fuzzy 37

3.3 MPC 40

3.3.1 MPC pre problém regulácie 40

3.3.2 MPC pre problém sledovania 41
3.3.3 Štandardná formulácia 42
3.3.4 Formulácia s pozorovačom stavov a poruchy 43
3.3.5 Tvrdé a mäkké ohraničenia 43
3.3.6 MPC v regulačnom obvode 44
3.4 Porovnanie regulátorov 45
4 Riadenie sústavy vozidiel 46
4.1 Opis systému 47
4.2 Riadenie v MATLAB-e 49
4.3 Prístupy k riadeniu 50
4.3.1 Decentralizované riadenie 50
4.3.2 Centralizované riadenie 51
4.3.3 Modelovanie riadeného systému 52
4.4 Simulačná časť 54
4.5 Riadenie reálneho systému 56
4.5.1 PI riadenie 56
4.5.2 Centralizované riadenie MPC 57
4.5.3 Decentralizované riadenie MPC 58
4.5.4 Zhodnotenie 59
5 Záver 61
Zoznam použitej literatúry 62

Zoznam ilustrácií

Obrázok 1 Konvexná a nekonvexná množina 19
Obrázok 2 Konvexné mnohouholníky 20
Obrázok 3 Problém lineárneho programovania 21
Obrázok 4 Problém kvadratického programovania 22
Obrázok 5 Spojitý a diskrétny signál 26
Obrázok 6 Stabilita systémov 30
Obrázok 7 Hranica stability 31
Obrázok 8 Pozorovač v regulačnom obvode 32
Obrázok 9 PID regulátor 35
Obrázok 10 Fuzzy funkcie príslušnosti 38
Obrázok 11 Prediktívny regulátor princíp činnosti 42
Obrázok 12 Prediktívny regulátor v obvode 44
Obrázok 13 Počet obetí dopravných nehôd v EU [35] 46
Obrázok 14 Robotické vozidlo 48
Obrázok 15 Decentralizovaný prístup k riadeniu 51
Obrázok 16 Centralizovaný prístup k riadeniu 52
Obrázok 17 Simulačné porovnanie regulátorov 55
Obrázok 18 Riadenie použitím PI regulátora 57
Obrázok 19 Riadenie použitím centralizovaného prediktívneho regulátora 58
Obrázok 20 Riadenie použitím decentralizovaného prediktívneho regulátora 59
Obrázok 21 Narazenie na ohraničenia 60

Úvod

Automatizácia sa zaoberá teoretickými princípmi a praktickými aspektmi návrhu riadiacich algoritmov. Prístroje ovládané takýmito algoritmami sa nachádzajú prakticky všade okolo nás. Ide o automatickú práčku, klimatizáciu, čerpadlo, elektrickú rúru. Každé z týchto zariadení má v sebe zakomponovaný určitý návod ako postupovať, správať sa, v určitej situácii.

V počítačovej simulácii je možné nastaviť čerpadlu záporný prietok, otvoriť ventil na viac ako 100%. Ale pre zvýšenie použiteľnosť riadiacich algoritmov je nutné vziať do úvahy ohraničenia reálneho sveta.

Matematická optimalizácia sa zaoberá hľadaním riešení, ktoré sú podľa stanovených kritérií lepšie ako aktuálny stav. Je to proces zlepšovania, ktorého výsledkom je optimálne riešenie, to najlepšie možné [1].

MPC, prediktívne riadenie s modelom, rieši riadenie ako optimalizačný problém. Čím nie len hľadá najlepšie možné riešenie ale dokáže pracovať aj s ohraničeniami. V našej práci využívame princípy a metódy teórie riadenia s cieľom navrhnúť prediktívny regulátor s pre decentralizovaný a centralizovaný systém sústavy robotických vozidiel. Uvedieme metódu návrhu prediktívnych regulátorov, ktorých funkčnosť a kvalitu riadenia porovnáme s bežne používanými regulátormi.

V závere práce zhrnieme výsledky a porovnáme navrhnuté prediktívne regulátory. Táto práca by mala obsahovať základné poznatky z oblasti syntézy prediktívnych regulátorov. Aktuálnosť témy spočíva v tom, že prediktívne riadenie je dôležitou súčasťou riadiacich systémov. Ďalším dôvodom aktuálnosti je súvislosť témy so zameraním nášho ústavu.

1 Optimalizácia

S rozvojom techniky a výpočtových technológií prichádzajú stále novšie a efektívnejšie možnosti ako zostrojovať alebo vyhodnocovať riadiace algoritmy. Čo umožňuje, aby rozhodovanie, spôsob riadenia bol optimálny, aby poskytoval najlepšie možné výsledky podľa zvoleného kritéria. Z toho vyplýva, že na nájdenie optimálneho riešenia je potrebné tieto podmienky poznať. Medzi takéto podmienky najčastejšie patria spotreba energie, materiálu, alebo najrýchlejšie dosiahnutie žiadaného stavu. Riešením optimalizačnej úlohy je optimálne riešenie, ktoré je zlučiteľné so všetkými fyzikálnymi obmedzeniami [2].
Matematicky zapisujeme optimalizačný problém ako:

$$f^* = \min_x f(x) \tag{1.1}$$

$$\text{s.t. } x \in X, \tag{1.2}$$

kde je $x \in R^n$ – vektor optimalizovaných premenných,
$f(x) : R^n \to R$ – účelová funkcia,
f^* – hodnota účelovej funkcie v optime,
$X \in R^n$ – množina všetkých prípustných hodnôt x.

Optimalizačný problém predstavuje problém výberu tej najlepšej možnosti zo súboru všetkých dostupných možností. Premenná x predstavuje uskutočnenú voľbu; ohraničenia X sú pevné reštrikcie alebo špecifikácie, ktoré obmedzujú možné výbery a hodnota účelovej funkcie hovorí o cene uskutočneného výberu. Riešením optimalizačného problému je taký výber z vektora optimalizovaných premenných, ktorý zodpovedá najlepšiemu výberu, medzi všetkými možnosťami ktoré spĺňajú požiadavky [3].

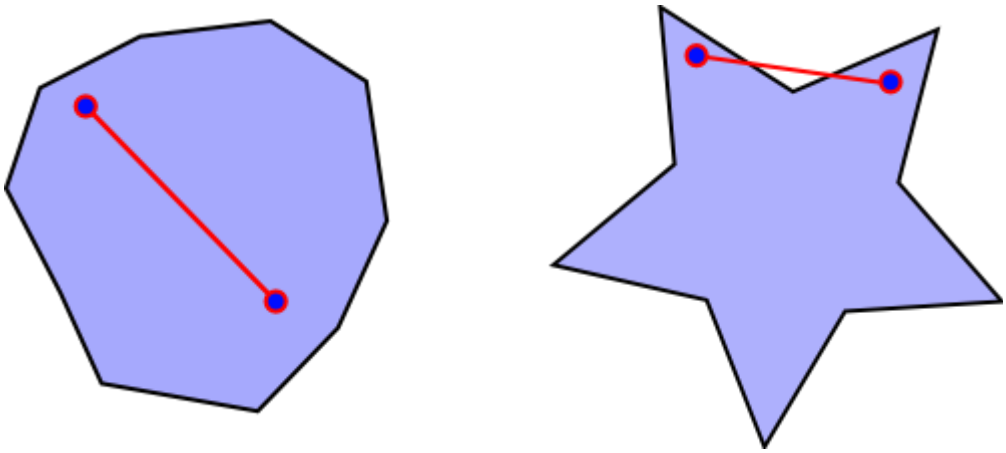
1.1 Konvexná optimalizácia

Optimalizačný problém je konvexný, ak účelová funkcia (1.1) a funkcie tvoriace ohraničenia (1.2) sú konvexné, čo znamená, že vyhovujú nerovnosti:

$$f_i(\alpha x + \beta y) \leq \alpha f_i(x) + \beta f_i(y) \tag{1.3}$$

$\forall x, y \in R^n$, $\forall\, \alpha, \beta \in R$ s $\alpha + \beta = 1, \alpha \geq 0, \beta \geq 0$. Optimalizačný problém je nekonvexný, ak účelová funkcia nemá konvexný tvar. V tejto práci sa zaoberáme riešením optimalizačných problémov v konvexnom tvare. Na základe geometrickej definície hovoríme o množine, že je

konvexná, ak úsečka, ktorá spája ľubovoľné dva body z množiny, leží celá v tejto množine (Obr. 1) [3] .



Obrázok 1 Konvexná a nekonvexná množina

1.1.1 Ohraničenia

Riešiteľnosť každého optimalizačného problému určujú práve jeho ohraničenia. Používajú sa na lepšiu znázornenie fyzikálnych vlastností systémov alebo na sa zabezpečenie stability. Delíme ich do dvoch kategórií:

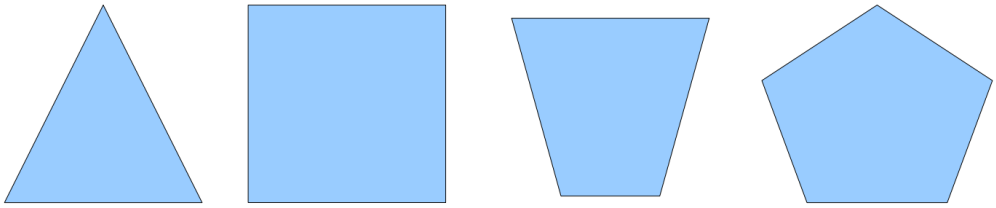
- Tvrdé ohraničenia musia byť vždy splnené, inak by bol optimalizačný problém neriešiteľný. Ide o fyzické obmedzenia, ako percento otvorenia ventila alebo produkciu nezáporného množstva výrobkov.
- Mäkké ohraničenia môžu byť porušené, ale úroveň ich porušenia sa odráža v účelovej funkcii, a preto sa minimalizuje. Sú to rôzne bezpečnostné obmedzenia alebo všetky druhy komfortných obmedzení.

Ohraničenia nastavujú presnú hodnotu alebo vymedzujú vzťah medzi optimalizovanými premennými. Vyčleňujú určité časti priestoru, v ktorých sa nachádza prípustné riešenie optimalizačného problému.

Polyhedron P je konvexná a uzavretá množina určená ako priesečník konečného množstva n_H uzavretých afinných polpriestorov $a_i^T x \leq b_i, a_i \in R^n, b \in R, \forall i = 1, \ldots, n_H$, značíme:

$$P = \{x \in R^n \mid Ax \leq b\}, \tag{1.4}$$

kde $A \in R^{n_H \times n}, b \in R^{n_H}$. Množinu P nazývame polytop ak ide o ohraničený polyhedron. Polytopy tvoriace ohraničenia v dvojrozmernom priestore sú konvexné mnohouholníky (Obr. 2) [3].



Obrázok 2 Konvexné mnohouholníky

Na základe funkcií tvoriacich ohraničenia, delíme optimalizačné problémy na:

i. neohraničená optimalizácia
ii. ohraničená optimalizácia
    a. ohraničenia v tvare nerovnosti
    b. ohraničenia v tvare rovnosti
    c. kombinácia dvoch predošlých

1.1.2 Účelová funkcia

Vo všeobecnosti rozlišujeme druhy alebo typy optimalizačných problémov podľa tvaru účelovej funkcie. Najčastejšie sa stretávame s účelovou funkciou v lineárnom alebo kvadratickom tvare, hovoríme o riešení optimalizačných problémov lineárneho resp. kvadratického programovania [3].

1.1.2.1 Lineárne programovanie

Dôležitou skupinou optimalizačných problémov je lineárne programovanie, v ktorom účelová funkcia a všetky funkcie určujúce ohraničenia sú lineárne:

$$\min_{x} \; c^T x + r \tag{1.5a}$$

$$\text{s. t. } g_i^T x \leq b_i \tag{1.5b}$$

$$h_i^T x = d_i, \tag{1.5c}$$

kde je $x \in R^n$ – vektor optimalizovaných premenných,

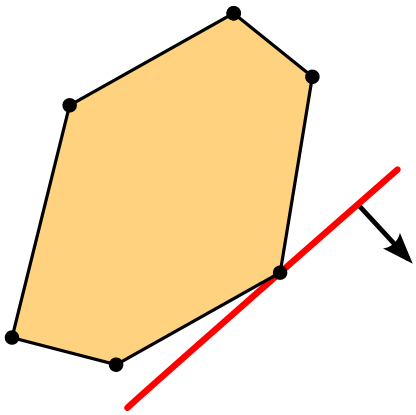
$c^T x + r: R^n \to R$ – účelová funkcia,

$g_i^T x \leq b_i: R^n \to R, i = 1, \ldots, m$ – ohraničenie účelovej funkcie v tvare nerovnosti,

$h_i^T x = d_i: R^n \to R, i = 1, \ldots, m_e$ – ohraničenie účelovej funkcie v tvare rovnosti.

Vektory $c, g_1, \ldots, g_m, h_1, \ldots, h_{m_e} \in R^n$ a skaláre $r, b_1, \ldots, b_m, d_1, \ldots, d_{m_e} \in R$ sú známe parametre, ktoré špecifikujú účelovú funkciu a funkcie ohraničení.

Zložitosť problému v závislosti od počtu optimalizovaných premenných n a počtu ohraničení m je $n^2 m$ za predpokladu, že $m \geq n$. Použitím výpočtového softwéru možno ľahko vyriešiť problémy so stovkami optimalizovaných premenných a tisíckami ohraničení na bežnom počítači za pár sekúnd. Ak je optimalizačný problém riedky alebo má ohraničenia, ktoré možno priamo dosadiť do účelovej funkcie, možno za prakticky zanedbateľný čas vyriešiť problémy so stovkami až tisíckami optimalizovaných premenných a ohraničení. Lineárne programovanie je v súčasnosti neoddeliteľnou súčasťou mnohých výpočtových nástrojov ako aj praktických aplikácií [3].



Obrázok 3 Problém lineárneho programovania

Obrázok (Obr. 3) znázorňuje jednoduchý problém lineárneho programovania s dvoma premennými a šiestimi ohraničeniami. Množina realizovateľných riešení tvorená ohraničeniami je zobrazená žltou farbou. Účelová funkcia je reprezentovaná červenou čiarou a šípkou, ktorá ukazuje smer poklesu hodnoty účelovej funkcie, v ktorom optimalizujeme.

1.1.2.2 Kvadratické programovanie

Ďalšou významnou skupinou optimalizačných problémov je kvadratické programovanie, v ktorom účelová funkcia obsahuje kvadratický člen a všetky funkcie určujúce ohraničenia sú lineárne:

$$\min_x \frac{1}{2} x^T P x + q^T x + r \tag{1.6a}$$

$$\text{s.t. } g_i^T x \leq b_i \tag{1.6b}$$

$$h_i^T x = d_i, \tag{1.6c}$$

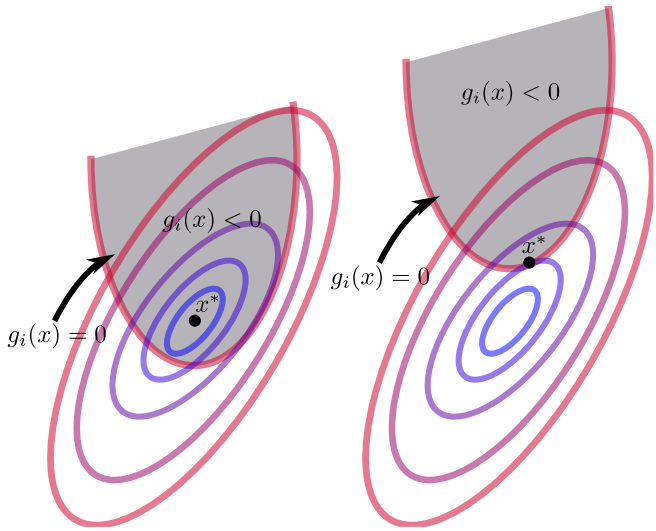
kde je $x \in R^n$ – vektor optimalizovaných premenných,

$\frac{1}{2} x^T P x + q^T x : R^n \to R$ – účelová funkcia,

$g_i^T x \leq b_i : R^n \to R, i = 1, \ldots, m$ – ohraničenie účelovej funkcie v tvare nerovnosti ,

$h_i^T x = d_i : R^n \to R, i = 1, \ldots, m_e$ – ohraničenie účelovej funkcie v tvare rovnosti.

Matica $P \in n \times n$, vektory $q, g_1, \ldots, g_m, h_1, \ldots, h_{m_e} \in R^n$ a skaláre $r, b_1, \ldots, b_m, d_1, \ldots, d_{m_e} \in R$ sú známe parametre, ktoré špecifikujú účelovú funkciu a funkcie ohraničení. V prípade, ak je matica P pozitívne definitná, má kladné vlastné čísla, optimalizačný problém má jedno riešenie [3].



Obrázok 4 Problém kvadratického programovania

Ilustrácia (Obr. 4) znázorňuje závislosť polohy optimálneho riešenia x^* problému kvadratického programovania s ohraničením od konkrétneho umiestnenia ohraničení. Účelová funkcia je reprezentovaná vrstevnicami, ohraničenie parabolou $g_i(x)$, pričom uvažujme $b_i = 0$.

1.2 Riešenie optimalizačných problémov

V súčasnosti sa veľké úsilie kladie na vývoj algoritmov na riešenie rôznych druhov optimalizačných problémov. Účinnosť algoritmov, schopnosť riešiť optimalizačný problém sa značne líši a závisí od konkrétnej formulácie účelovej funkcie a funkcii tvoriacich ohraničenia, počtu optimalizovaných premenných a obmedzení [3].

Na riešenie problému lineárneho programovania nie je daný analytický vzorec priamo, ale existujú rôzne účinné metódy ich riešenia. Medzi najpoužívanejšie patrí simplexova metóda, alebo aj novšie metóda vnútorného bodu. Rovnako ako na riešenie problému lineárneho programovania, ani na kvadratické programovanie nie je daný analytický vzorec priamo, ale existujú rôzne účinné metódy ich riešenia. Spôsob riešenia je odlišný pre problémy s ohraničeniami v tvare nerovnosti, ohraničeniami v tvare rovnosti alebo neohraničený problém. Medzi najpoužívanejšie patrí metóda aktívnych množín, metóda vnútorného bodu, rôzne gradientové a poklesové metódy. Kvadratické programovanie je jedným z najpoužívanejších typov optimalizačného problému, nakoľko funkcia spotreby, či už materiálu alebo energie je v kvadratickom tvare [4].

Populárnejšími sa stávajú globálne metódy riešenia optimalizačných problémov, heuristické algoritmy. V priebehu riešenia algoritmus postupne prechádza možné riešenia a hľadá to, ktoré mu viac vyhovuje. Motiváciu k použitiu heuristických metód je, že si dokážu poriadiť s lokálnymi extrémami. Takýmito metódami sú Nelderov-Meadov algoritmus alebo simulované žíhanie (*simulated annealing*) [4].

V súčasnosti je optimalizácia dôležitým nástrojom v mnohých oblastiach. Používa sa v projektovom inžinierstve, v potravinárskych a priemyselných výrobách, ako aj v ostatných oblastiach priemyslu. Pri väčšine týchto aplikácií matematickú optimalizáciu využíva osoba zodpovedná za rozhodovanie, návrh riadiaceho systému alebo operátor, ktorý dohliada na proces, kontroluje výsledky a reaguje na vzniknuté situácie. S rozvojom výpočtovej techniky sa postupne znižuje výpočtový čas optimalizácie [3].

Teóriu optimalizácie podrobne spracovali Boyd a Vandenberghe [3], riešenie konkrétnych druhov optimalizačných problémov nájde čitateľ v prácach autorov Dupačová

a Lachout [4], Hamala a Trnovská [5]. V našej práci používame na riešenie optimalizačných problémov doplnok MATLAB-u *yalmip* s použitím riešiteľa *Gurobi*.

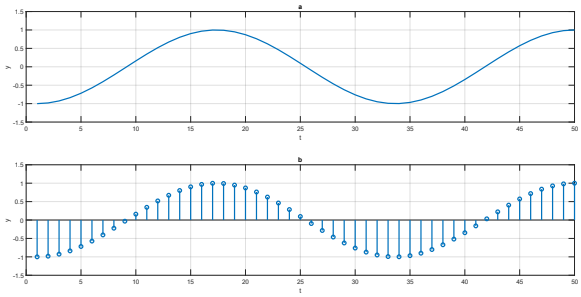
2 Modelovanie

Ak chceme navrhnúť riadiaci algoritmus pre konkrétny proces alebo zariadenie, potrebujeme poznať jeho vlastnosti, jeho správanie sa v určitých situáciách, pracovných bodoch. Zaujíma nás, ako proces reaguje na podnety. Naším objektom záujmu je proces, alebo zariadenie s ktorým chceme pracovať. Objekt je časť sveta okolo nás, ktorú skúmame. V našom prípade spomínané zariadenie. Matematický model je abstrakcia reálneho objektu, ktorá s určitou presnosťou popisuje jeho správanie sa, najmä reakcie na vonkajšie zásahy. Systém je objekt, ktorého prvky sú známe a vieme popísať ich vzájomné vzťahy. Najčastejšie nás kvôli jednoduchosti zaujíma závislosť jednej veličiny v systéme od veličiny ktorú máme možnosť meniť. Veličiny, ktoré meníme nazývame riadiace veličiny, vstupy do systému. Tie veličiny, ktoré sa menia od zmeny vstupov nazývame výstupy. Na systém však majú vplyv aj veličiny, na ktorých zmenu nemáme žiadny dosah, tieto veličiny nazývame poruchy[6].

Medzi hlavné dôvody na vytváranie matematických model procesov patrí: umožňujú sledovanie dynamiky procesu bez nutnosti experimentov, hľadanie vhodných pracovných podmienok procesov, simulácia možných nebezpečných situácií, analýza možnosti vzniku porúch, návrh riadiacich algoritmov systémov a sú vhodné tiež na trénovanie obsluhy zariadení, operátorov [6].

Systémy delíme na základe matematických modelov:

- dynamický, statický: zmeny veličiny v čase,
- spojitý, diskrétny: spôsob zmeny v čase,
- so sústredenými parametrami, s rozloženými parametrami : zmena veličín v priestore,
- stacionárny, nestacionárny: zmena parametrov menia v čase,
- lineárny, nelineárny: opis vzťahov medzi veličinami,
- deterministický, stochastický: výskyt náhodných veličín.



Obrázok 5 Spojitý a diskrétny signál

Obrázok znázorňuje rozdiel medzi spojitým a diskrétnym signálom. Spojitý signál, na (Obr. 5a), nadobúda konkrétnu hodnotu v každom časovom okamihu, nepretržite. Diskrétny signál je postupnosť, ktorá pozostáva zo série hodnôt veličiny v čase. Na rozdiel od signálu spojitého nie je funkciou nepretržitého argumentu, možno ho však získať odberom vzoriek z kontinuálneho signálu v určitom vzorkovacom čase. Tento časový úsek nazývame perióda vzorkovania, v tejto práci značíme T_s. Perióda vzorkovania je pevne stanovená a jej veľkosť ovplyvňuje podobu diskrétneho signálu (Obr. 5b).

V tejto práci sa využívame dynamické modely, ktoré sa menia v diskrétnom čase, sú deterministické a lineárne. Práve takéto modely sa najčastejšie používajú na návrh pokročilejších riadiacich algoritmov.

Systémy delíme na základe počtu vstupných a výstupných veličín na:

- SISO (*Single Input Single Output*), systém s jedným vstupom a jedným výstupom
- SIMO (*Single Input Multi Output*), systém s jedným vstupom a viacerými výstupmi
- MISO (*Multi Input Single Output*), systém s viacerými vstupmi a jedným výstupom
- MIMO (*Multi Input Multi Output*), systém s viacerými vstupmi a viacerými výstupmi

2.1 Prístupy k modelovaniu

Existuje viacero spôsobov ako sa získať model procesu, v tejto časti uvedieme v súčasnosti najpoužívanejšie prístupy, opíšeme ich a porovnáme výhody a nevýhody ich použitia.

1. Deterministický, analytický model vzniká na základe analýzy znalostí matematických, fyzikálnych a chemických vlastností modelovaného systému. Zakladá sa na odvodzovaní vlastností systému zo všeobecných poznatkov, postupujeme od všeobecného ku konkrétnemu. Keďže bol vytvorený dedukciou na základe vzťahov a vlastností, pri procese tvorby nie je potrebná prítomnosť reálneho zariadenia. Medzi nevýhody patrí nevyhnutnosť znalostí fyzikálneho správania sa systému ako aj hodnôt parametrov. Je pomerne komplexný a zložitý, častokrát použitie zjednodušujúcich predpokladov môže viesť k nepresnostiam v jeho správaní sa modelu.
2. Experimentálny, štatistický model sa snaží len reprodukovať správanie sa výstupu systému ako reakciu na zmenu vstupov. Postup zostrojenia je indukčný, ide o zovšeobecnenie jedinečného správania sa. Zvolená matematická formulácia sa odlišuje od rovníc, ktoré skutočne opisujú systém. Výsledkom je model, ktorý bude fungovať presne v podmienkach, pri ktorých boli stanovené pôvodné parametre. Výhodou prístupu je to, že tvorba vyžaduje malú alebo dokonca žiadnu znalosť systému a preto môžu byť pomerne rýchlo skonštruovaný aj pre matematicky komplikované systémy. Medzi nevýhody prístupu patrí potreba práce s reálnym zariadením na získanie experimentálnych výsledkov. V prípade použitia v inej oblasti ako pracovná oblasť experimentu nemusí byť použiteľný.

V súčasnosti sa využíva aj kombinovaný prístup k modelovaniu, ktorý kombinuje výhody analytického a experimentálneho prístupu [6].

2.2 Typy modelov

Na základe vzťahov medzi veličinami modelu rozlišujeme dva základné typy modelov, model na základe vzťahov medzi vstupmi a výstupmi a stavový opis systému.

2.2.1 Vstupno-výstupný opis

V prípade použitia modelov vychádzajúcich zo vzťahov medzi vstupnými výstupmi veličinami používame model v tvare diferenčnej rovnice a model v tvare prenosovej funkcie.

2.2.1.1 Diferenčná rovnica

Ak chceme sledovať reakcie výstupu dynamického systému na zmenu vstupných veličín v diskrétnom čase potrebujeme vyjadriť vzťah medzi veličinami. V tomto prípade máme

k dispozícii postupnosti hodnôt výstupnej a vstupnej veličiny v určitých sekvenčných časoch, periódach vzorkovania. Vyjadrenie vzťahu medzi vstupnou a výstupnou veličinou vo forme diferenčnej rovnice má tvar:

$$a_N y(k+N) + a_{N-1} y(k+N-1) + \cdots a_1 y(k+1) + a_0 y(k) = \tag{2.1a}$$

$$= b_M u(k+M) + b_{M-1} u(k+M-1) + \cdots b_1 u(k+1) + b_0 u(k), \tag{2.1b}$$

kde je N – rád systému,

a_i, b_i – koeficienty rovnice,

$u(k), u(k+1), \dots, u(k+M)$ – vstupná veličina a jej hodnoty v sekvenčnom čase, meniacom sa o periódu vzorkovania T_s,

$y(k), y(k+1), \dots, y(k+N)$ – výstupná veličina a jej hodnoty v sekvenčnom čase, meniacom sa o periódu vzorkovania T_s,

$N \geq M$ – podmienka fyzikálnej realizovateľnosti systému.

2.2.1.2 Prenos

Z–transformácia konvertuje diskrétny signál, ktorý je sekvenciou reálnych alebo imaginárnych čísel, do komplexnej reprezentácie frekvenčnej oblasti. Ide o diskrétny ekvivalent Laplaceovej transformácie [7]. Z pôvodnej diferenčnej rovnice dostávame:

$$G(z) = \frac{P(z)}{Q(z)} = \frac{b_M z^M + b_{M-1} z^{M-1} + \cdots + b_1 z + b_0}{a_N z^N + a_{N-1} z^{N-1} + \cdots + a_1 z + a_0} \tag{2.2a}$$

$$P(z) = \sum_{\mu=0}^{M} b_\mu z^\mu \tag{2.2b}$$

$$Q(z) = \sum_{\eta=0}^{N} a_\eta z^\eta \tag{2.2c}$$

kde je z – operátor komplexnej premennej,

N – stupeň polynómu menovateľa systému,

M – stupeň polynómu čitateľa systému,

$G(z)$ – prenosová funkcia v diskrétnom čase,

$z = p_\mu$, pre ktoré platí $P(p_\mu) = 0$ je koreň čitateľa, nula systému,

$z = q_\eta$, pre ktoré platí $Q(q_\eta) = 0$ je koreň menovateľa, pól systému.

2.2.2 Stavový opis

V oblasti riadenia procesov je reprezentácia stavového priestoru matematickým modelom fyzického systému ako množiny vstupných, výstupných a stavových premenných vzťahujúcich sa na diferenciálne rovnice alebo diferenčné rovnice prvého rádu [8].

Stavový opis poskytuje prehľadný a kompaktný spôsob modelovania a analýzy systémov s viacerými vstupmi a výstupmi. V prípade použitia prenosovej funkcie s M vstupmi a J výstupmi by sme museli vytvoriť $M \times J$ prenosových funkcií na získanie všetkých informácií o systéme. Použitie stavového opisu nie je viazané na lineárne systémy ani na nulové počiatočné podmienky [9].

V prípade diskrétneho času pre N stavov, M vstupov a J výstupov dostávame:

$$x(k+1) = Ax(k) + Bu(k) \tag{2.3a}$$

$$y(k) = Cx(k) + Du(k), \tag{2.3b}$$

kde je $x(k) \in R^N$ – stavový vektor,

$y(k) \in R^J$ – vektor výstupov,

$u(k) \in R^M$ – vstupný vektor, vektor riadenia,

$A : (N \times N)$ – stavová matica, matica dynamiky,

$B : (N \times M)$ – váhová matica vstupu,

$C : (J \times N)$ – váhová matica stavového vektora,

$D : (J \times M)$ – váhová matica výstupu.

Λ, pre ktoré platí:

$$|A - \Lambda I| = 0, \tag{2.3c}$$

kde I je jednotková matica nazývame vlastný vektor matice A

$\lambda_i \; \forall \; i \in 1, \cdots, N$ pre ktoré platí:

$$|A - \Lambda I| = (\lambda_1 - \Lambda)(\lambda_2 - \Lambda) \cdots (\lambda_N - \Lambda) \tag{2.3d}$$

nazývame i-te vlastné číslo matice A.

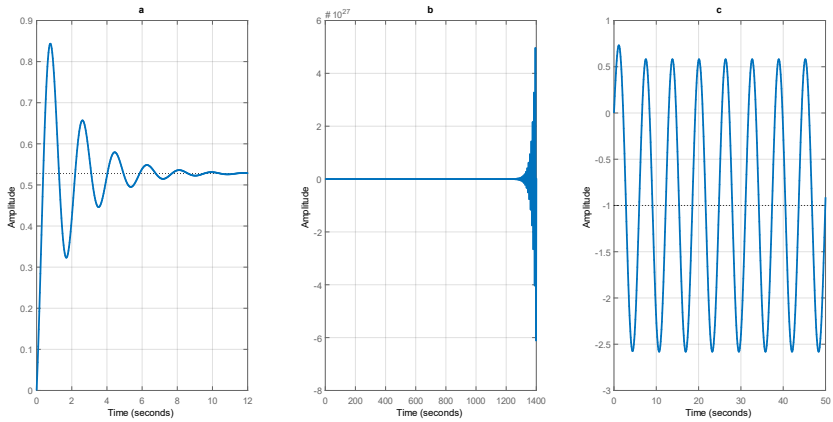
2.3 Stabilita

Jednou zo základných vlastnosti systémov je stabilita. Keďže je naším cieľom dynamický systém riadiť, udržať jeho výstup na žiadanej hodnote, potrebujeme vedieť ako sa tento výstup správa. Na základe správania sa, delíme systémy na:

a. stabilné systémy, ich výstup postupne klesá, alebo rastie, prípadne tlmene kmitá, ale ustáli sa v okolí konkrétnej hodnoty (Obr. 6a)

b. nestabilné systémy, ich výstup klesá do $-\infty$, rastie do ∞ alebo postupne zväčšuje veľkosť svojich kmitov (Obr. 6b)

c. systémy na hranici stability, ich výstup harmonicky kmitá okolo konkrétnej hodnoty (Obr. 6c)

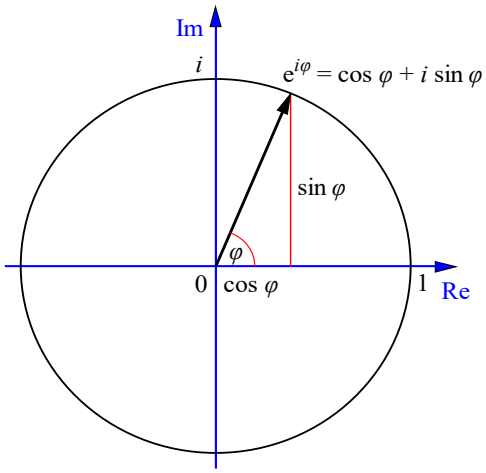


Obrázok 6 Stabilita systémov

(Obr. 6) znázorňuje porovnanie stability rôznych systémov, vlastné čísla znázornených systémov sú nasledovné:

$$\lambda_a = \begin{pmatrix} -0.58 + 0.2i \\ -0.58 - 0.2i \\ 0.05 + 0.0i \end{pmatrix} \lambda_b = \begin{pmatrix} -1.00 + 0.4i \\ -1.00 - 0.4i \\ 0.05 + 0.0i \end{pmatrix} \lambda_c = \begin{pmatrix} 0.54 + 0.8i \\ 0.54 - 0.8i \\ 0.37 + 0.0i \end{pmatrix} \tag{2.4}$$

Analýza stability riadiacich systémov súvisela s riešením diferenciálnych rovníc. Maxwell analyzoval stabilitu Wattovho odstredivého regulátora. Linearizoval diferenciálne rovnice, aby sa získal charakteristickú rovnica systému [10]. Routh poskytol numerickú techniku na určenie stability koreňov charakteristickej rovnice [11]. Vishnegradsky analyzoval stabilitu regulátorov pomocou diferenciálnych rovníc nezávisle od Maxwella [12]. V roku 1893 Stodola študoval reguláciu vodnej turbíny pomocou techník Vishnegradského. Modeloval dynamiku pohonu a zahrnul do jeho analýzy dopravné oneskorenie ovládacieho mechanizmu. Bol prvý, kto spomenul pojem systémovej časovej konštanty. Nehľadiac na prácu Maxwella a Routha, položil otázku, ako určiť stabilitu charakteristickej rovnice Hurwitzovi, ktorý ho riešil nezávisle[13]. Lyapunov od roku 1892 študoval stabilitu nelineárnych diferenciálnych rovníc pomocou všeobecného definície pojmu energie[14].



Obrázok 7 Hranica stability

Hranicou stability je pre diskrétne systémy jednotková kružnica (Obr. 7). Spojitosť medzi prenosovou funkciou a stavovým opisom spočíva v tom, že korene menovateľa prenosu sú vlastné čísla matice dynamiky stavového opisu. Keďže opisuje vlastnosti systému, menovateľa prenosovej funkcie nazývame aj charakteristickou rovnicou [6].

Pre stabilitu teda platí:

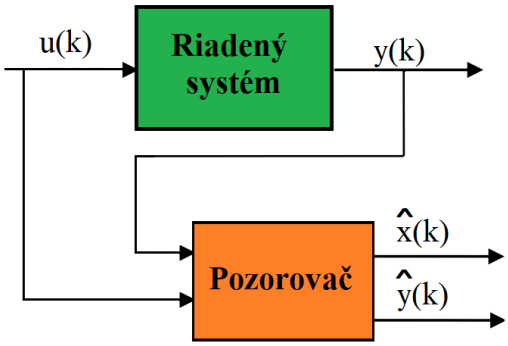
- diskrétny systém je stabilný, ak všetky korene charakteristickej rovnice, všetky vlastné čísla matice dynamiky ležia vo vnútri jednotkového kruhu,
- diskrétny systém je na hranici stability, ak aspoň jeden koreň charakteristickej rovnice, jedno vlastné číslo matice dynamiky leží na jednotkovej kružnici,
- spojitý systém je nestabilný, ak aspoň jeden koreň charakteristickej rovnice, jedno vlastné číslo matice dynamiky leží mimo jednotkového kruhu [6].

2.4 Pozorovače

Poruchy sú veličiny, vplývajúce na dynamický systém, na ktorých zmenu nemáme žiadny dosah. Niektoré môžeme merať priamo, u iných môžeme sledovať ich vplyv na inú meranú veličinu. Jednou z možností ako znížiť vplyv pôsobenia porúch je zistiť ich veľkosť použitím rozšíreného pozorovača stavov.

2.4.1 Pozorovač stavov

Nie vždy je možné všetky stavy procesu merať. Ak chceme poznať správanie sa stavov, ktoré nemeriame je možné určiť ich správanie použitím pozorovača. Pozorovač je dynamický systém, ktorého stavy a výstupy konvergujú k stavom a výstupom systému, ktorý pozoruje (Obr. 8).



Obrázok 8 Pozorovač v regulačnom obvode

Znalosť stavu systému je potrebná na riešenie mnohých problémov spojených s teóriou riadenia. Vo väčšine praktických prípadov nemožno fyzický stav systému priamo merať. Namiesto toho sa pozorujú účinky pôsobenia vstupov na výstupy systému. Ak je systém pozorovateľný, je možné odhadnúť stav systému z meraní jeho výstupov pomocou pozorovača stavu [16].

Uvažujme stavový opis v diskrétnom tvare (2.6). Premenné pozorovača stavu sú označené so strieškou $\hat{x}(k)$ na odlíšenie od premenných rovníc pôvodného pozorovaného dynamického systému $x(k)$.

$$\hat{x}(k+1) = Ad\hat{x}(k) + L\big(y(k) - \hat{y}(k)\big) + Bu(k) \tag{2.5a}$$

$$\hat{y}(k) = C\hat{x}(k) \tag{2.5b}$$

Pozorovač je stabilný, ak chyba pozorovača $e\,(k) = \hat{x}(k) - x\,(k)$ konverguje k nule $\lim_{k \to \infty} e\,(k) = 0$. Chyba pozorovača je $e\,(k+1) = (A - LC)e\,(k)$.

Pozorovač diskrétneho systém je preto stabilný, keď matica $A - LC$ má všetky vlastné hodnoty vo vnútri jednotkového kruhu. Výstup pozorovaného systému vráti späť ako vstup násobený matricu zosilnení K:

$$u(k) = -K\hat{x}(k) \tag{2.6}$$

K a L volíme nezávisle na zabezpečenie celkovej stability systému. Zvyčajne sa vlastné čísla výrazu $A - LC$ volia 10-krát rýchlejšie ako vlastné čísla $A - BK$. Tento typ pozorovača sa označuje Luenbergerov pozorovač podľa autora [17].

2.4.1.1 Modelovanie poruchy systému

Pozorovač môžeme použiť nie len na odhad stavov a výstupov, ale aj ako kvantifikátor poruchy pôsobiacej na proces. Uvažujme stavový opis v diskrétnom tvare (2.6) a nemerateľnú poruchu pôsobiacu na výstup riadeného dynamického systému d, pričom C_{dis} je matica poruchy:

$$y(k) = Cx(k) + C_{dis}d(k) \tag{2.7a}$$

$$d(k+1) = d(k) \tag{2.7b}$$

Rozšírený stavový opis pozorovača, ktorý obsahuje stavy pôvodného systému ale aj poruchu v tvare:

$$\begin{bmatrix} \hat{x}(k+1) \\ \hat{d}(k+1) \end{bmatrix} = \begin{bmatrix} A & 0 \\ 0 & I \end{bmatrix} \begin{bmatrix} x(k) \\ d(k) \end{bmatrix} + \begin{bmatrix} B \\ 0 \end{bmatrix} u(k) + \begin{bmatrix} L_x \\ L_d \end{bmatrix} \big(y(k) - \hat{y}(k)\big) \tag{2.8a}$$

$$\hat{y}(k) = \begin{bmatrix} C & C_{dis} \end{bmatrix} \begin{bmatrix} x(k) \\ d(k) \end{bmatrix}. \tag{2.8b}$$

V prípade záujmu môže čitateľ viac informácií o problematike šumu nájsť v práci autorov Motchenbacher a Connelly [15], aspektmi pozorovačov sa zaoberali Sontag [16], Mikleš a Fikar [17].

3 Riadiace algoritmy

Úlohou riadenia procesov je dosiahnutie a udržanie želaného správania sa systémov. To sa dosahuje návrhom programov na to určených, riadiacich algoritmov. V tejto časti rozoberieme základné metódy riadiacich algoritmov, popíšeme princíp činnosti a zhodnotíme ich výhody a nevýhody.

Uvažujme riadený systém, v ktorom chceme zabezpečiť dosiahnutie žiadaného stavu v tvare (2.3). Riadiacim algoritmom rozumieme matematický predpis ktorý určí a nastaví taký vstup do riadiaceho systému, akčný zásah, aby sa jeho aplikovaním postupne odstránil rozdiel:

$$e(k) = r(k) - y(k) \tag{3.1}$$

kde $r(k) \in R^J$ značí referenciu, žiadanú hodnotou výstupu. Veličinu $e(k)$ nazývame regulačná odchýlka. Cieľom riadenia je stabilizovať riadený systém a dosiahnuť, aby regulačná odchýlka v ustálenom stave, čiže trvalá regulačná odchýlka, bola odstránená:

$$\lim_{k \to \infty} e\,(k) = 0 \tag{3.2}$$

3.1 PID

V praxi sa v súčasnosti najčastejšie používajú regulátory, ktorých algoritmus sa skladá z troch zložiek, proporcionálnej, integračnej a derivačnej, skrátene PID regulátor. Každá zo zložiek na základe hodnoty regulačnej odchýlky, jej derivácie resp. integrálu v čase vytvára svoj príspevok k akčnému zásahu. V roku 1939 predstavila spoločnosti *Taylor Instrument Companies* novú verziu svojho pneumatického regulátora *Fulscope*, kde okrem proporcionálneho a integrálneho zásahu, tento nový nástroj poskytoval zásah, *pre-act*. V tom istom roku pridala spoločnosť *Foxboro Instrument Company* funkciu *hyper-reset* na ich pneumatický regulátor *Stabilog*. *Pre-act* a *Hyper-Reset* boli prvé regulátory umožňujúce deriváciu signálu regulačnej odchýlky, ako prvé poskytovali možnosť riadenia PID algoritmom [19], [20].

V praxi je možné použiť rôzne spôsoby prepojenia vzájomných väzieb medzi zložkami PID regulátora, najčastejšie používaná je štandardná forma:

$$C(s) = K_p\left(1 + \frac{1}{T_i}\frac{1}{s} + T_d s\right), \tag{3.3}$$

kde je K_p – proporcionálne zosilnenie [-]

T_i – integračná časová konštanta [s]

T_d – derivačná časová konštanta [s]

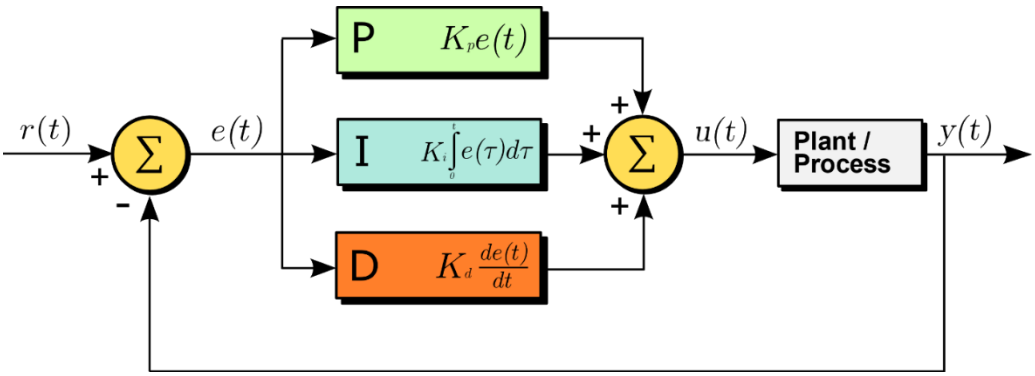
štandardný tvar je možné prepísať do paralelnej formy:

$$C(s) = K_p + \frac{K_p}{T_i}\frac{1}{s} + K_p T_d s \ = K_p + K_i \frac{1}{s} + K_d s. \tag{3.4}$$

Môžeme sa stretnúť aj so sofistikovanejšími verziami základného spôsobu riadenia, spoločnosť *Honeywell* používa tvar:

$$C(s) = K_p \left(1 + \frac{1}{T_i}\frac{1}{s}\right)\frac{T_d s + 1}{\varphi T_d s + 1}, \tag{3.5)k}$$

táto forma umožňuje vhodným nastavením parametra $\varphi \in (0,1)$ zníženie preregulovania pri rýchlych zmenách regulačnej odchýlky alebo vplyve porúch. Rýchle zmeny žiadanej hodnoty derivačná zložka zosilňuje, čo môže spôsobiť kmitanie akčného zásahu. [2], [21].



Obrázok 9 PID regulátor

(Obr. 9) znázorňuje zapojenie PID regulátora v regulačnom obvode. Každá zo zložiek regulátora určitým spôsobom ovplyvňuje jeho správanie. Hlavnou zložkou regulačného obvodu je proporcionálna konštanta, ktorá určuje vplyv regulačnej odchýlky v aktuálnom čase na akčný zásah. Integračná zložka ovplyvňuje hodnotu integrálu regulačnej odchýlky, určuje vplyv minulého správania sa systému na súčasný akčný zásah. D zložka násobí deriváciu regulačnej odchýlky, pôsobí proti jej zmenám [2], [21].

Pre správne fungovanie PID regulátora je vhodné uvedomiť si vplyv konštánt na správanie sa regulačného obvodu. Najdôležitejšou časťou je pôsobenie proporcionálnej zložky, zosilnenie. Veľkosť parametra umožní priblíženie sa žiadanej veličine, sama však nedokáže odstrániť trvalú regulačnú odchýlku. Integračná zložka sa používa práve na odstránenie regulačnej odchýlky. Jej vplyv sa ale prejavuje pomalšie ako vplyv proporcionálnej zložky, čím sa spomaľuje regulačný pochod, zväčšuje sa perióda kmitov, a zhoršuje sa stabilita obvodu. Derivačná zložka pôsobí

rýchlejšie, tlmí kmity, zrýchľuje regulačný pochod a zlepšuje stabilitu. Ale v prípade šumu alebo rýchlych zmien žiadanej hodnoty spôsobuje nežiadúce kmitanie.

Najpoužívanejšou formou je plná forma PID regulátora. V prípade integračných vlastností riadeného procesu alebo ak trvalá regulačná odchýlka nespôsobuje problém sa používajú P a PD forma. V prípade procesu, ktorého šum nie je možné odfiltrovať sa používajú P a PI forma. [2], [21], [22].

Jednou z možností ako jednoducho implementovať PID algoritmus je:

```
r, Kp, Ki, Kd, Ts, Nsim
e_min = 0
suma = 0
for i = 1:Nsim
  tic
  e = r - y
  sum = sum + e * Ts
  der = (e - e_min) / Ts
  u = Kp * e + Ki * sum + Kd * der
  e_min = e
  toc
  pause(Ts-toc)
end
```

V tomto prípade sú dve premenné, suma do integračnej zložky a minulá hodnota regulačnej odchýlky, inicializované na nulu, potom začne slučka. N_{sim} je celkový počet krokov simulácie. Regulačná odchýlka sa vypočíta odpočítaním nameranej hodnoty výstupu od aktuálnej žiadanej hodnoty. Potom sa aktualizujú hodnoty reprezentujúce integrál a deriváciu. Súčtom proporcionálnej, integračnej a derivačnej zložky vzniká akčný zásah. Hodnota regulačnej odchýlky sa uloží ako minulá, aby sa použila pri ďalšej diferenciácii, program potom čaká a slučka prebehne so zmenenými hodnotami. V prípade implementácie v prostredí MATLAB-u slúžia príkazy *tic*, *toc*, na začatie, resp. zastavenie počítania prejdeného času, príkaz *pause* pozastaví algoritmus, aby bol cyklus vykonaný za periódu vzorkovania.

Medzi výhody PID regulátorov patrí jednoduchosť syntézy, ladenia a implementácie. Nie je potrebné poznať model procesu, PID dokáže riadiť systém ako čiernu skrinku. Ale keďže

nepoznáme model procesu, nie sme schopní pracovať s ohraničeniami. PID je možné použiť iba pre SISO systémy[2], [21], [22].

Metódami syntézy PID regulátorov sa zaoberali Ziegler a Nichols [23], Åström a Hägglund [24], ktorí navrhli vzorce pri výpočet konštánt regulátora pre konkrétne podmienky správania sa regulačného pochodu. Používané formy PID regulátora a spôsoby implementácie diskrétnom čase prehľadne spracoval Kalúz [25]. Odporúčame čitateľovi aj práce autorov King [18], Beneš [21], Maixner [2], Bakošová a Fikar [22], kde nájde množstvo riešených príkladov a detailných vysvetlení problematiky riadenia.

3.2 Fuzzy

Fuzzy riadiaci algoritmus je založený na fuzzy logike - matematickom aparáte, ktorý analyzuje vstupné hodnoty ako logické premenné, ktoré nadobúdajú spojite hodnoty medzi 0 a 1 na rozdiel od klasickej logiky, ktorá pracuje s diskrétnymi hodnotami, buď 1 alebo 0 (pravda alebo nepravda) [2].

V protiklade s klasickou logikou, kde prvok ς buď patrí alebo nepatrí do množiny χ, Zadehova funkcia príslušnosti, určuje nakoľko prvok do množiny s neostrými hranicami patrí:

$$\mu_{\chi}(\varsigma) = \mathfrak{f} \tag{3.10a}$$

$$\mathfrak{f} \in \langle 0,1 \rangle \tag{3.10b}$$

čiže stupeň príslušnosti prvku ς k množine χ je $\mathfrak{f}$ [2], [26].

V prípade použitia fuzzy riadenia je najskôr potrebné:

1. Výber premenných pre vstupy a výstupy
2. Stanovenie hodnôt premenných (Obr. 10).
3. Definovanie funkcii príslušnosti premenných (Obr. 10).
4. Určenie pravidiel vo formáte:

$$Þ_i: \left(ς_1 \in \chi_{1,i}, ς_2 \in \chi_{2,i}, \cdots, ς_3 \in \chi_{3,i}\right) \rightarrow ч \in \Psi_i, \tag{3.11}$$

   kde $Þ_i$ je i-te pravidlo,

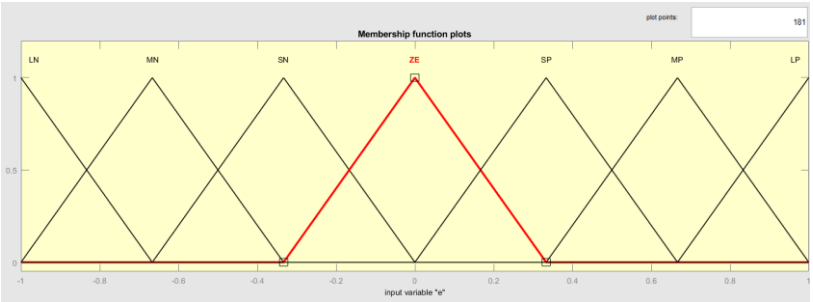
   $ς_1 - ς_3$ sú vstupné veličiny,

   $ч$ je výstupná veličina,

   $\chi_{1,i} - \chi_{3,i}$ sú hodnoty vstupných veličín v $Þ_i$,

   Ψ_i je hodnota výstupnej veličiny v $Þ_i$.

5. Určenie spôsob prevodu reálnych vstupov na fuzzy vstupy (fuzzifikácia), fuzzy výstupov na reálne výstupy (defuzzifikácia).



Obrázok 10 Fuzzy funkcie príslušnosti

V prípade Fuzzy regulátora je ladenie možné zmenou konštánt ovplyvňujúcich vstupy do vyhodnocovacieho procesu a konštanty limitujúcej maximálny akčný zásah. Uvažujúc bežne používaný Fuzzy PI regulátor sú vstupmi regulačná odchýlku e a jej zmenu Δe. Ladiacu konštantu prvého vstupu do fuzzy regulátora, regulačnej odchýlky, značíme K_e, ladiacu konštantu druhého vstupu, zmeny regulačnej odchýlky, K_{de} a ladiacu konštantu ovplyvňujúcu výstup z regulátora, maximálny akčný zásah, značíme K_u.

Jednoduchá implementácia Fuzzy regulátora:

```
r, Ku, Ke, Kde, Ts, F, Nsim
e_min = 0
u_min = 0
for i = 1:Nsim
  tic
  e = r - y
  u = Ku*(u_min + evalfis([Ke*e; Kde*(e-e_min)],F))
  e_min = e
  u_min = u
  toc
  pause(Ts-toc)
end
```

Fuzzy pravidlá vytvorené vo *fuzzy toolboxe* MATLAB-u je možné pomocou príkazu *readfis* uložiť do premennej, v tomto príklade F. Dve premenné, minulý akčný zásah a minulá hodnota regulačnej odchýlky, sú inicializované na nulu, potom začne slučka. N_{sim} je celkový počet krokov simulácie. Regulačná odchýlka sa vypočíta odpočítaním nameranej hodnoty výstupu od aktuálnej žiadanej hodnoty. Vyhodnotením pravidiel príkazom *evalfis* použitím váhovaných vstupov regulátora získame zmenu akčného zásahu. Túto hodnotu pripočítame k poslednej použitej hodnote akčného zásahu čím získame aktuálny akčný zásah. Hodnota regulačnej odchýlky a akčného zásahu sa uložia ako minulé, program potom čaká a slučka prebehne so zmenenými hodnotami. Príkazy *tic, toc,* na slúžia začatie, resp. zastavenie počítania prejdeného času, príkaz *pause* pozastaví algoritmus, aby bol cyklus vykonaný za periódu vzorkovania.

Fuzzy logika je ľahko použiteľná a poskytuje intuitívnejší prístup bez ďalekosiahlej zložitosti. Fuzzy logika toleruje nepresné údaje. Je postavená na štruktúrach kvalitatívneho popisu používaného v každodennom jazyku [2], [26].

Medzi výhody Fuzzy regulátorov patrí jednoduchosť syntézy, ladenia a implementácie. Čím väčší počet funkcií príslušnosti priradíme jednotlivým premenným, tým viac pravidiel je potrebné vytvoriť. Čím viac pravidiel tým väčšia citlivosť regulátora, ale aj zložitosť ich vyhodnocovania. Fuzzy regulátor ale nie je schopný pracovať s ohraničeniami a je možné ho použiť iba pre SISO systémy.

Viac o fuzzy množinách a fuzzy riadení môže čitateľ nájsť v prácach Maixnera [2], Zadeha [26], prípadne v [27].

3.3 MPC

Prediktívne riadenie s modelom, *model predictive control* (MPC), je v súčasnosti v priemysle najviac využívaná metóda riadenia vyššieho stupňa. Jednou z príčin vzniku tejto formy riadenie bola ropná a plynová kríza v roku 1973. V dôsledku klesajúcich zdrojov boli rafinérie nútené optimalizovať prevádzku a množstvá výroby sa snažili tlačiť čo najbližšie k ich limitným parametrom. Motiváciou pre použitie prediktívneho riadenia je možnosť zohľadnenia limitných stavov, vstupov a výstupov pri návrhu regulátora [28].

V 80-tych rokoch optimalizačné snahy vyvrcholili. Richalet, Rault, Testud a Papon v práci *Model Predictive Heuristic Control* vytvorili softvér na identifikáciu a riadenie. V práci nazvanej *Dynamic Matrix Control* Cutler a Ramaker pre spoločnosť *Shell Oil* popísali nový spôsob riadenia systémov s viacerými vstupmi a viacerými výstupmi pre proces destilácie. Obe práce pochádzali priamo z priemyselnej praxe a priniesli novú metódu riadenia dnes známeho ako MPC. Avšak zložité výpočty predstavovali pre pamäť používaných počítačov často neriešiteľný problém [28], [29].

3.3.1 MPC pre problém regulácie

V oblasti riadenia procesov, rozlišujeme 2 základné úlohy. Cieľom regulácie je kompenzovať vplyv porúch pôsobiacich na výstupnú veličinu a dosiahnuť jej konvergenciu k nemeniacej sa žiadanej hodnote. Tento problém je možné jednoducho vyriešiť použitím MPC. Uvažujme riadený systém opísaný modelom v tvare (2.3), optimalizačný problém so vstupmi $\theta(k)$ pre problém regulácie je potom možné zostaviť v tvare:

$$\min_{U(k)} \sum_{k=0}^{N-1} \left(x(k)^T Q x(k) + u(k)^T R u(k) \right) \tag{3.12a}$$

$$\text{s.t. } u_{\max} \geq u(k) \geq u_{\min}, \forall k \in \{1, \cdots, N-1\} \tag{3.12b}$$

$$y_{\max} \geq y(k) \geq y_{\min}, \forall k \in \{1, \cdots, N-1\} \tag{3.12c}$$

$$x(k+1) = Ax(k) + Bu(k), \forall k \in \{1, \cdots, N\} \tag{3.12d}$$

$$y(k) = Cx(k), \forall k \in \{1, \cdots, N-1\}, \tag{3.12e}$$

kde Q je pozitívne semidefinitný váhový koeficient stavov,

R je pozitívne definitný váhový koeficient vstupov,

$u_{\max}, u_{\min}$ je maximálna alebo minimálna povolená hodnota riadiacej veličiny,

$y_{\max}, y_{\min}$ je maximálna alebo minimálna povolená hodnota riadenej veličiny,

N je predikčný horizont, časový úsek, na ktorom sa vykonáva predikcia správania sa procesu na základe modelu v tvare (3.12d), (3.12e).

Prediktívny regulátor pracuje nasledovne:

1. Použitím metódy na riešenie optimalizačného problému (1.2) získame na základe vektora vstupov:

$$\theta(k) = [x(k)^T, u(k-1)^T] \tag{3.13}$$

trajektóriu optimálnych akčných zásahov:

$$U(k)^* = [u(1)^*, \cdots, u(N-1)^*] \tag{3.14}$$

2. Z vypočítanej sekvencie predikovaných akčných zásahov $U(k)^*$ sa použije do procesu iba prvý z vypočítaných $u(1)^*$. Túto metódu nazývame princíp pohyblivého predikčného horizontu.

Na riadenie použitím prediktívneho regulátora potrebujeme poznať:

- dynamický model procesu v tvare (2.1), (2.2), (2.3), alebo (2.5),
- v prípade riadenia na referenciu potrebujeme poznať jej trajektóriu v čase,
- optimalizačnú účelovú funkciu (3.12a) a ohraničenia optimalizovaných premenných (3.12b), (3.12c), (3.12d) a (3.12e) na predikčnom horizonte.

3.3.2 MPC pre problém sledovania

Druhou úlohou riadenia je sledovanie žiadanej veličiny. Podstatou sledovania je zaistiť výstupu k meniacej sa referencii. Jedna z formulácií MPC na riešenie problému sledovanie je v tvare:

$$\min_{U(k)} \sum_{k=0}^{N-1} (y(k) - y^S)^T Q (y(k) - y^S) + (u(k) - u^S)^T R (u(k) - u^S) \tag{3.15}$$

kde premenné s označením $\boldsymbol{s}$ v hornom indexe sú veličiny ustáleného stavu systému. do ktorého sa chceme dostať. Vstupný vektor do optimalizácie (3.13) bude v tomto prípade v tvare : $\theta(k) =$

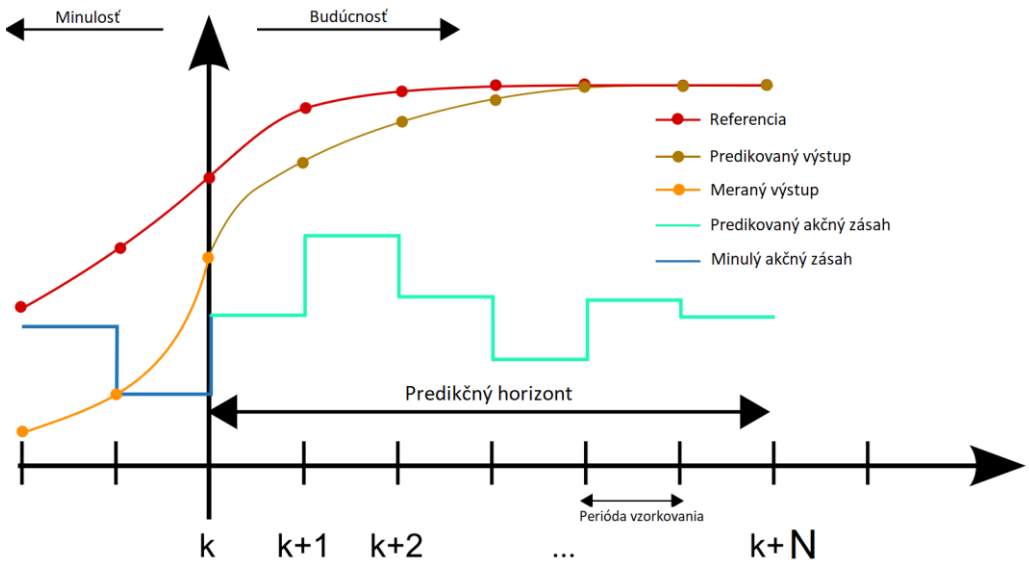
$[x(k)^T, u(k-1)^T, (y^S)^T]$Čím viac sa hodnota riadiacej a riadnej veličiny líši od ich hodnôt v ustálenom stave, tým väčšia je hodnota účelovej funkcie. V ustálenom stave je hodnota takto naformulovanej účelovej funkcie nulová.

3.3.3 Štandardná formulácia

Podobne ako v prípade PID aj MPC umožňuje zavedenie integrátora do regulačného obvodu. Realizuje sa formuláciu optimalizačného problému s prírastkami riadenia, v tvare:

$$\min_{U(k)} \sum_{k=0}^{N-1} \big(y(k) - r(k)\big)^T Q\big(y(k) - r(k)\big) + \big(u(k) - u(k-1)\big)^T R\big(u(k) - u(k-1)\big) \tag{3.16}$$

V tomto prípade sú predchádzajúca hodnota riadiacej veličiny $u(k-1)$, ako aj hodnota referencie $r(k)$ známymi premennými v účelovej funkcii. Teda vektor vstupov do optimalizácie (3.13) je v tomto prípade v tvare: $\theta(k) = [x(k)^T, u(k-1)^T, r(k)^T]$



Obrázok 11 Prediktívny regulátor princíp činnosti

Na základe znalostí modelu procesu máme dostupné informácie do času k. Prediktívny regulátor získa sekvenciu optimálnych akčných zásahov (3.14), použije prvý, ostatné zahodí. Opakovaním optimalizácie a aplikovaním prvého z trajektórie akčných zásahov sa výstup systému postupne dostane na referenciu.(Obr. 11)

3.3.4 Formulácia s pozorovačom stavov a poruchy

V prípade ak na riadený proces majú výrazný vplyv veličiny, na ktorých zmenu nemáme dosah je možné do ohraničení zahrnúť Luenbergerov pozorovač stavu rozšírený o modelovanie poruchy (2.7) :

$$\min_{U(k)} \sum_{k=0}^{N-1} \big(y(k) - r(k)\big)^T Q\big(y(k) - r(k)\big) + \big(u(k) - u(k-1)\big)^T R\big(u(k) - u(k-1)\big) \tag{3.17a}$$

$$\text{s. t. } u_{\max} \geq u(k) \geq u_{\min} \tag{3.17b}$$

$$y_{\max} \geq y(k) \geq y_{\min} \tag{3.17c}$$

$$\hat{x}(k+1) = A\hat{x}(k) + Bu(k) + L_x\big(y(k) - \hat{y}(k)\big) \tag{3.17d}$$

$$\hat{d}(k+1) = I\hat{d}(k) + L_d\big(y(k) - \hat{y}(k)\big) \tag{3.17e}$$

$$y(k) = C\hat{x}(k) + Du(k) + C_{dis}\hat{d}(k) \tag{3.17f}$$

V tomto prípade je však potrebné zaradiť odhadované stavy ako aj poruchu ako vstupy do optimalizácie, čím získavame vektor vstupných parametrov (3.13) v tvare $\theta(k) = \big[\hat{x}(k)^T, \hat{d}(k)^T u(k-1)^T, r(k)^T\big]$.

3.3.5 Tvrdé a mäkké ohraničenia

Ako bolo spomenuté v (1.1.1) v optimalizačnom probléme rozlišujeme tvrdé a mäkké ohraničenia. Mäkké obmedzenia sa dodržiavajú, ak je to možné, ale budú porušená, ak je to jediný spôsob, ako zabrániť porušovaniu tvrdých obmedzení. Do ohraničenia, ktoré chceme zmäkčiť, zavedieme pomocnú premennú $s \in R$:

$$y_{\max} + s \geq y(k) \geq y_{\min} - s \tag{3.18a}$$

$$s \geq 0 \tag{3.18b}$$

Túto premennú zároveň pridáme do účelovej funkcie ako novú optimalizovanú premennú s pozitívne semidefinitnou váhou Q_s:

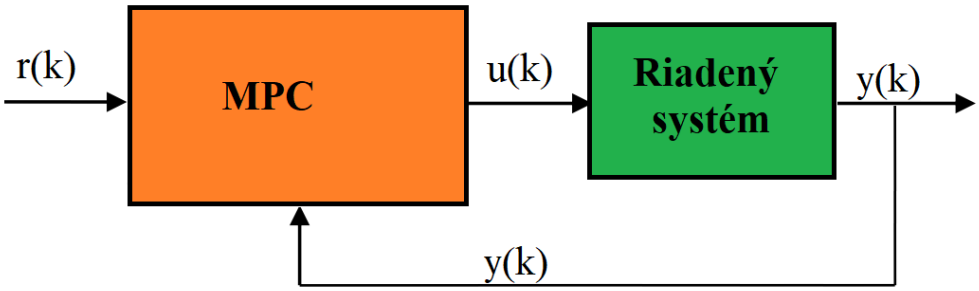
$$\min_{U(k),s} \sum_{k=0}^{N-1} \Big(x(k)^T Q x(k) + \big(u(k) - u(k-1)\big)^T R\big(u(k) - u(k-1)\big) + sQ_s s\Big) \tag{3.19}$$

V prípade ak je možné nájsť optimálne riešenie vzhľadom na tvrdé aj mäkké ohraničenia hodnota zmäkčovacej premennej $\boldsymbol{s}$ bude nulová. Ak to možné nie je, $\boldsymbol{s}$ bude mať nenulovú

hodnotu, nájde sa optimálne riešenie vzhľadom na tvrdé a zmäkčené obmedzenia. Avšak každé navýšenie premennej $\boldsymbol{s}$, každé zmäkčenie ohraničenia spôsobí zvýšenie hodnoty účelovej funkcie.

3.3.6 MPC v regulačnom obvode

V tejto časti si ukážeme jeden so spôsobov použitia prediktívneho regulátora v regulačnom obvode. Uvažujme dynamický model procesu v tvare (2.1), (2.2), (2.3), alebo (2.5).



Obrázok 12 Prediktívny regulátor v obvode

Jednoduchá implementácia prediktívneho regulátora (Obr. 12) použitím *yalmip*-u [40]:

```
r, u0, x0, Nsim
ctrl = optimizer(F,G,opt,[x',u',w'],u(:,1))
for i = 1:Nsim
  if i == 1
        theta = [x0',u0',r']
    else
        theta = [x',u',r']
    end
    u = ctrl(theta)
    [y,x] = system(u)
end
```

Použitím doplnku MATLAB-u *yalmip* je možné pomocou príkazu *optimizer* uložiť optimalizačný problém (3.12a) ako F s ohraničeniami (3.12b), (3.12c), (3.12d) a (3.12e) ako G, nastaveniami preferovanej metódy riešenia opt do špeciálneho objektu, čím vzniká predpripravený nízkoúrovňový číselný formát, ktorý môže byť použitý na efektívne riešenie série

podobných optimalizačných problémov, v tomto príklade ho nazývame $ctrl$. Do premennej $theta$ ukladáme premenné potrebné na riešenie optimalizačného problému (3.13). Výstupom z $ctrl$ už nie je celá sekvencia optimálnych akčných zásahov, ale využitím posuvného predikčného horizontu, len jej prvý člen. Na základe použitého akčného zásahu získame nové stavy, nové výstupy systému a algoritmus pokračuje iteračne.

3.4 Porovnanie regulátorov

Riadenie použitím MPC sa líši od PID, Fuzzy v mnohých aspektoch. MPC ako jediný algoritmus z uvažovaných zvláda pracovať s obmedzeniami. Umožňuje nastavenie hraničných limitov pre každú riadenú veličinu. Potom využije rozsah medzi týmito limitami čím zabráni porušeniu obmedzení a zníži celkovú hodnotu účelovej funkcie. MPC ďalej umožňuje, aby bol akčný zásah ohraničený. Pri PID, Fuzzy by limitovanie riadenej veličiny mohlo spôsobiť odchýlku od referencie. MPC poskytuje optimálny akčný zásah, pričom optimalizácia prebieha v každej perióde vzorkovania na posuvnom horizonte, čo umožňuje optimalizáciu v reálnom čase aj oproti obmedzeniam [18].

Na základe teoretických poznatkov z tejto kapitoly, môžeme porovnať prístupy k návrhu regulátorov podľa nasledovných kritérií (Tab. 1).

Tabuľka 1 Porovnanie vlastností regulátorov

	PID	**Fuzzy**	**MPC**
Jednoduché ladenie	áno	áno	áno
Optimálny akčný zásah	nie	nie	áno
SISO aj MIMO systémy	nie	nie	áno
Práca s ohraničeniami	nie	nie	áno

4 Riadenie sústavy vozidiel

Ako dôsledok rastu počtu vozidiel a nedostatočných zmien v cestnej infraštruktúre sa nežiaduce dopravné situácie stávajú každodenným problémom vo všetkých veľkých mestách. Často dochádza ku kolíziám alebo vzniku cestných kolón, čo vedie k celkovému spomaleniu dopravy ako aj k zvýšenej spotrebe paliva [32]. Pri nehodách dochádza nie len k ekologickým a ekonomickým škodám, ale aj k stratám na životoch. Podľa správy Európskej rady pre bezpečnosť dopravy za rok 2016 zomrelo v Európskej únii na následky dopravných nehôd viac ako 25-tisíc ľudí (Obr. 14), z čoho asi pätina životov vyhasla ako následok požitia alkoholu za volantom [33].

Obrázok 13 Počet obetí dopravných nehôd v EU [35]

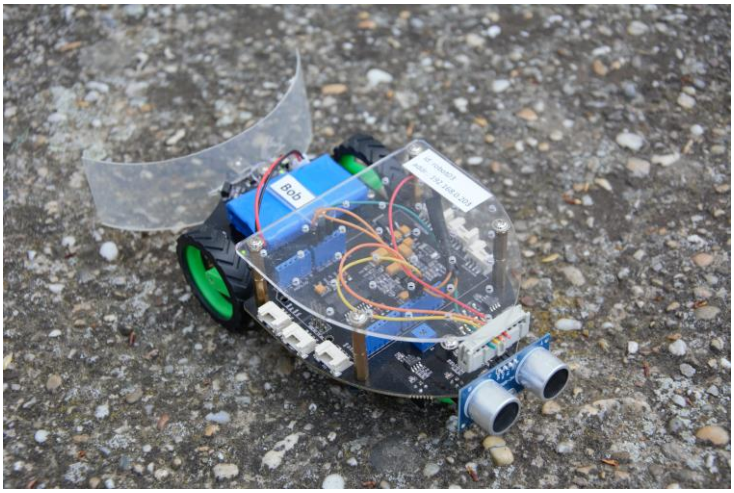
Na zníženie rizika vzniku takýchto kritických situácií využívajú vodiči vo svojich vozidlách informačné a riadiace systémy. Úlohou týchto systémov je pomôcť vodičom efektívne vyriešiť dopravné situácie. Na riadenie sústavy vozidiel počas cestnej premávky je možné použiť viacero prístupov, známe sú výsledky prác použitím adaptívneho riadenia [34], použitím genetického algoritmu [35], i použitím prediktívneho riadenia s modelom [36]. MPC je možné použiť na riadenie dopravnej zápchy [37], na autonómne riadenie premávky [38], ale žiadna z týchto prác neponúka spôsob riadenia vzhľadom na obmedzenia výstupov aj akčných zásahov.

Väčšina dostupných prác prináša počítačové simulácie riadenia premávky. Je to z dôvodu, že vykonanie experimentov v skutočnej premávke by bolo finančne náročné. Jedným z najpoužívanejších alternatívnych prístupov je riadenie malých laboratórnych modelov vozidiel [39].

V tejto práci podobne tvorí riadený systém sústava robotických vozidiel. Cieľom riadenia je dosiahnuť želanú vzdialenosť medzi vozidlami. Riadiaca veličina, ktorou chceme vzdialenosť dosiahnuť a udržať, je rýchlosť motorov. Vozidlá sa pritom budú pohybovať po čiernej čiaru reprezentujúcu vozovku.

4.1 Opis systému

Našim cieľom je simulovať riadenie cestnej premávky. Riadený systém tvorí skupina robotických vozidiel. Každé vozidlo (Obr. 15) je vybavené riadiacou jednotkou Arduino Yún, ktorá obsahuje mikropočítač s procesorom Atheros a distribúciu Linuxu OpenW doplnenú o Ethernet, WiFi, MicroSD a USB rozhranie. Druhá počítačová jednotka je 8-bitový mikroprocesor Atmel ATMega32u4, ktorý spravuje signálové rozhranie. Komunikáciu medzi spracovateľskými jednotkami zabezpečuje sériový modul UART TTL. Automobilová elektronická doska obsahuje dva jednosmerné motory, riadiacu elektroniku motora, batériu, päť svetelných senzorov odrazu povrchu na sledovanie čiary a ultrazvukový snímač vzdialenosti. Ultrazvukový senzor na prednej časti robotického vozidla meria vzdialenosť od prekážky. V našom prípade je prekážkou vozidlo, ktoré sa pohybuje pred naším uvažovaným vozidlom, prípadne iný objekt nachádzajúci sa na vozovke [32].



Obrázok 14 Robotické vozidlo

Na ovládanie vozidla, možno použiť dva spôsoby komunikácie. Prvý, založený na štandardných žiadostiach HTTP POST, otvára spojenie iba pre jeden pár požiadaviek. Druhý používa WebSocket, ktorý je plne duplexný. Protokol je založený na protokole TCP. WebSocket otvorí komunikačný kanál a udržiava ho otvorený, až kým sa nedostaví pokyn na zastavenie. To umožňuje, aby komunikácia bola rýchlejšia, pretože nie je potrebné vytvárať vždy nové spojenie pri prenose údajov. Táto metóda je tiež menej náročná na množstvo doplňujúcich údajov, nachádzajúcich sa v každej správe. Komunikácia vozidiel je navrhnutá tak, aby poskytovali odpoveď na požiadavku danú riadiacim softvérom. Každé vozidlo poskytuje tri spôsoby pripojenia riadiaci softvér. Sú to:

1. WebSocket server – Riadiaci softvér plní úlohu klienta a môže vytvoriť jedno pripojenie na vozidlo využitím služieb WebSocket v sieti. V tomto prípade sa vozidlá správajú ako servery, a komunikácia s každým vozidlom sa spracováva oddelene.

2. WebSocket klient - V tomto prípade samotné vozidlá plnia úlohu klienta komunikácie. Každý klient vytvára spojenie a sprostredkovateľ správy poskytuje WebSocket pre riadiaci softvér. Toto umožňuje klientovi súčasne komunikovať s viacerými vozidlami naraz.

3. HTTP server - poskytuje podobný princíp komunikácie ako prvá metóda. Hlavným rozdielom je, že nepoužíva spínací protokol, ale je oddelené žiadosti [32].

4.2 Riadenie v MATLAB-e

Aplikačné programové rozhranie založené na MATLAB-e poskytuje trieda *RoboBug*. Táto trieda zabezpečuje pripojenie a konfigurácia robotov a umožňuje zadávanie príkazov. Použitie triedy je veľmi jednoduché a intuitívne. Vytvorenie objektu, ktorý reprezentuje robotické vozidlo, umožňuje nasledujúci príkaz:

```
vozidlo01 = RoboBug('192.168.0.201')
```

Triedny konštruktér akceptuje iba jeden argument, IP adresu robotického vozidla v rámci lokálnej siete. Po vytvorení triedy je potrebné vozidlo pripojiť použitím príkazu:

```
vozidlo01.connect()
```

Tento príkaz vyvolá spojenie *WebSocket* medzi MATLAB-om a robotickým vozidlo. Vzhľadom k tomu, že *WebSocket* nie je bežnou súčasťou programového vybavenia MATLAB-u používanie triedy *RoboBug* vyžaduje dva doplnky, ktoré je potrebné nainštalovať. Ide o triedy *matwebsocks* a *wsclient* založené na jazyku Java, ktoré je možné získať a nainštalovať pomocou nástroja na správu nástrojov *tbxmanager* [40].

Po úspešnom vytvorení spojenia s robotickým vozidlo je možné priamo ovládať a monitorovať pochody vozidla pomocou nasledujúcich príkazov:

```
% automatické sledovanie čiary au, hodnota 0 alebo 1:
vozidlo01.setAuto(au)
% určenie rýchlosti vozidla w, hodnota 0 až 127:
vozidlo01.setSpeed(w)
% pomer rýchlosti na jednotlivé kolesá df, hodnota 0 až 1:
vozidlo01.setDrvFact(df)
% určenenie bezpečnej vzdialenosti sd[cm]:
vozidlo01.setStopDist(sd)
% získanie informácie z IČ senzorov [cm]:
vzdialenost = vozidlo01.getDistance()
% získanie informácie zo svetelných senzorov:
Senzory = vozidlo01.getSensors()
% okamžité zastavenie vozidla:
vozidlo01.stop()
% odpojenie vozidla:
vozidlo01.close()
```

V našom prípade je riadenou veličinou vzdialenosť medzi vozidlami, resp. medzi vozidlom a prekážkou. Informáciu o vzdialenosti získavame z ultrazvukových senzorov. Riadiacu veličinou je hodnota rýchlosti vozidla, ktorú regulátor po vyriešení optimalizačného problému priamo zadá konkrétnemu vozidlu. Na sledovanie čiary je už k dispozícií vytvorený algoritmus, ktorý len inicializujeme príslušnou funkciou.

4.3 Prístupy k riadeniu

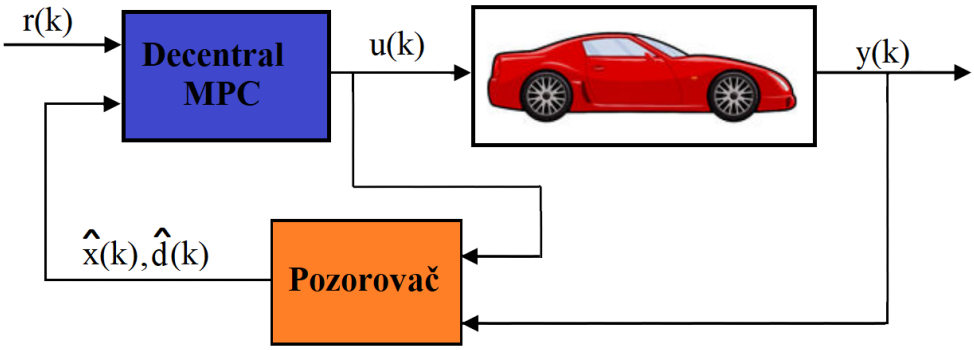
V oblasti riadenia skupiny vozidiel možno uvažovať dva prístupy. Často používaným prístupom je riadenie každého vozidla ako samostatného, autonómneho systému, ktorý berie ohľad na ostatných účastníkov našej improvizovanej premávky. V tejto práci hovoríme o tomto prístupe ako o decentralizovanej forme riadenia. Ďalší prístup považuje celú sústavu vozidiel za jednotný systém. V tomto prípade je úroveň riadenia vyššia. Nakoľko systém má informácie o všetkých prítomných vozidlách. Tento prístup v práci označujeme ako centralizované riadenie.

4.3.1 Decentralizované riadenie

Decentralizovaný systém je ten, v ktorom zložky nižšej úrovne pracujú s menšími informáciami, nie sú závislé od pokynov vyššieho vplyvu. V tomto systéme každá jeho zložka je rovnako zodpovedná za prispievanie k globálnemu celkovému výkonu, že pôsobí na lokálnej úrovni vhodným spôsobom.

V prírode pozorujeme decentralizované správanie v prípade kolónií hmyzu ako mraveniská mravcov, úle včiel, kde sú zložky systému usporiadané do celkov. Tieto celky sú následne pracujú na svojich čiastkových úlohách. Príkladom decentralizovaného riadenia je aj spoločnosť, kde rôzne oddelenia riadia rôzni manažéri.

V prípade decentralizovaného riadenia procesov je systém riadenia rozdelený na menšie samostatné celky. Každý z celkov je samostatný, riadený vlastným regulátorom. Výhodou tohto prístupu je jednoduchosť použitia. Medzi nevýhody patrí znížená schopnosť reagovať na zmeny, zásahy do systému, lebo žiaden z menších celkov nemá informácie o činnosti a vlastnostiach ostatných celkov. V tejto práci v prípade použitia decentralizovaného prístupu budeme modelovať neznámy stav systému, rýchlost predchádzajúceho vozidla ako aj poruchu na systém pôsobiacu (Obr.16) požitím rozšíreného Luenbergerovho pozorovača (2.7).



Obrázok 15 Decentralizovaný prístup k riadeniu

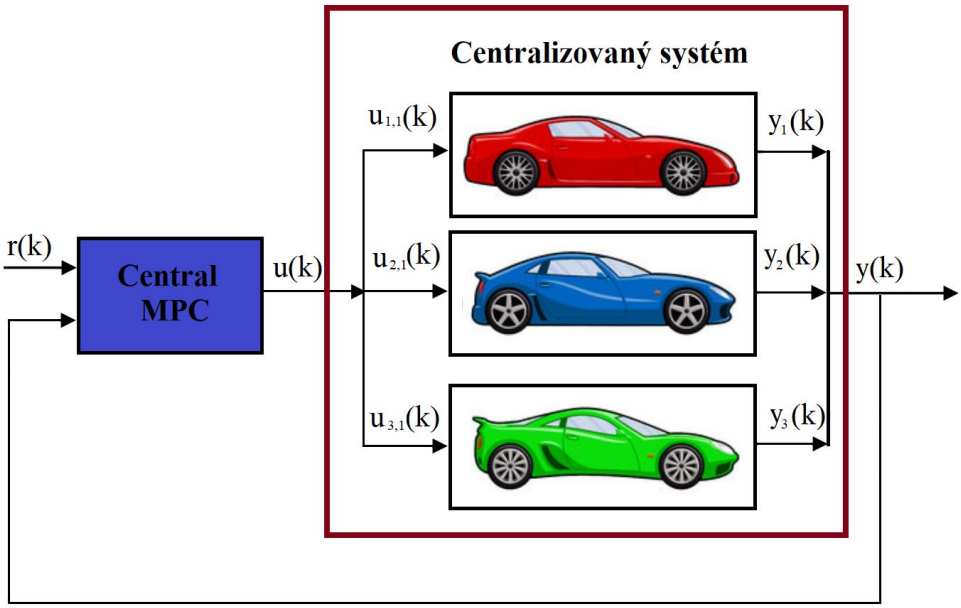
V tejto práci v prípade použitia decentralizovaného prístupu budeme modelovať neznámy stav systému, rýchlost predchádzajúceho vozidla ako aj poruchu na systém pôsobiacu (Obr.16) požitím rozšíreného Luenbergerovho pozorovača (2.7).

4.3.2 Centralizované riadenie

Centralizovaný systém je ten, v ktorom sú všetky zložky závislé od pokynov vyššieho, nadriadeného vplyvu. V tomto systéme má riadiaci mechanizmus všetky dostupné informácie a na ich základe rozhoduje. Príkladom centralizovaného riadenia je aj spoločnosť, ktorú celú, všetky oddelenia riadi jeden líder.

V prípade centralizovaného prístupu k riadeniu procesov riadi celý systém jeden centrálny regulátor. Tento regulátor disponuje všetkými informáciami o systéme a na ich základe riadi menšie celky. Výhodou tohto prístupu je schopnosť reagovať na zásahy do systému, na základe množstva informácii a schopnosti riadiť všetky časti súčasne. Nevýhodou je náročnosť na komunikáciu medzi regulátorom a časťami ktoré ovláda. V tejto práci uvažujeme nasledovný prístup (Obr. 17). Centralizovaný regulátor ako použitý riadiaci systém ovláda všetky vozidlá

súčasne, pritom má k dispozícii všetky dostupné informácie o systéme. Preto nie je potrebné pristúpiť k modelovaniu stavov ako v prechádzajúcom prípade.



Obrázok 16 Centralizovaný prístup k riadeniu

4.3.3 Modelovanie riadeného systému

V prípade použitia centralizovaného prístupu je celá sústava vozidiel riadená jedným, centrálnym regulátorom. Predpokladom je konštantná rýchlosť v_1 prvého auta, ako vozidla na nasledovanie. Rýchlosť prvého vozidla riadením neovplyvňujeme V našom prípade dolný index pri veličine značí vozidlo, ktorého sa veličina týka. Ale uvažovanú riadenú vzdialenosť x počítame od prvého vozidla ako lídra. Čiže vzdialenosť medzi prvým a druhým vozidlom x_1 riadime rýchlosťou druhého vozidla v_2. Pre systém zložený z n vozidiel, kde zápis veličiny s bodkou $\dot{x}$ značí časovú deriváciu, dostávame diferenciálne rovnice:

$$\dot{v}_1 = 0 \tag{4.1a}$$

$$\dot{x}_i = v_i - v_{i+1} \tag{4.1b}$$

$$y = \begin{bmatrix} x_1 \\ \vdots \\ x_{n-1} \end{bmatrix} \tag{4.1c}$$

$$\forall i \in \{1, \dots, n-1\} \tag{4.1d}$$

Z diferenciálnych rovníc získame matice spojitého stavového opisu:

$$\dot{x} = \mathrm{A}\begin{bmatrix} \mathrm{v}_1 \\ \mathrm{x}_\mathrm{i} \\ \vdots \\ \mathrm{x}_{\mathrm{n}-1} \end{bmatrix} + \mathrm{B}\begin{bmatrix} \mathrm{v}_2 \\ \vdots \\ \mathrm{v}_\mathrm{n} \end{bmatrix} \tag{4.2a}$$

$$y = C\begin{bmatrix} v_1 \\ x_i \\ \vdots \\ x_{n-1} \end{bmatrix} + D\begin{bmatrix} v_2 \\ \vdots \\ v_n \end{bmatrix} \tag{4.2b}$$

Získaný spojitý stavový opis diskretizujeme:

$$\frac{v_1(k+1) - v_1(k)}{Ts} = 0 \tag{4.3a}$$

$$\frac{x_i(k+1) - x_i(k)}{Ts} = v_i(k) - v_{i+1}(k), \tag{4.3b}$$

kde čas $k+1$ sa líši od času k o hodnotu času periódy vzorkovania T_s. Z diferenčných rovníc získame matice diskrétneho stavového opisu:

$$x(k+1) = A\begin{bmatrix} v_1(k) \\ x_i(k) \\ \vdots \\ x_{n-1}(k) \end{bmatrix} + B\begin{bmatrix} v_{i+1}(k) \\ \vdots \\ v_n(k) \end{bmatrix} \tag{4.4a}$$

$$y(k) = C\begin{bmatrix} v_1(k) \\ x_i(k) \\ \vdots \\ x_{n-1}(k) \end{bmatrix} + D\begin{bmatrix} v_{i+1}(k) \\ \vdots \\ v_n(k) \end{bmatrix} \tag{4.4b}$$

$$A = \begin{bmatrix} 0 & 0 & 0 & \dots & 0 & 0 \\ Ts & 0 & 0 & \dots & 0 & 0 \\ 0 & 0 & 0 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & 0 & 0 \end{bmatrix} \tag{4.4c}$$

$$B = \begin{bmatrix} 0 & 0 & 0 & \dots & 0 & 0 \\ -Ts & 0 & 0 & \dots & 0 & 0 \\ Ts & -Ts & 0 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & Ts & -Ts \end{bmatrix} \tag{4.4d}$$

$$C = \begin{bmatrix} 0 & 1 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix} \tag{4.4e}$$

$$D = \begin{bmatrix} 0 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{bmatrix} \tag{4.4f}$$

V prípade decentralizovaného prístupu je každé nasledujúce vozidlo riadené vlastným regulátorom. Decentralizovaný systém získame ak $n = 2$.

4.4 Simulačná časť

V tejto časti budeme simulovať riadenie modelu sústavy robotických vozidiel vybranými regulátormi a porovnáme kvalitu riadenia. V simuláciách vzchádzame z modelu v tvare (4.2) pre dve vozidlá. Na účely porovnania sme zvolili dve zmeny referencie, vzdialenosti medzi vozidlami, z hodnoty 10 cm na hodnotu 15 cm a naspäť. Uvažujme, že na sa vedúce vozidlo pohybuje rýchlosťou 10 $\frac{cm}{s}$ a počiatočnú vzdialenosť medzi vozidlami je 0 cm, čiže regulátor musí najprv počkať kým sa vzdiali vozidlo lídra.

Použitím príkazu programu MATLAB *pidTunner* sme na základe modelu získali konštanty PI regulátora. Na základe teoretických poznatkov v časti (3.1) sme sa rozhodli vynechať D zložku kvôli výskytu šumu na vstupe reálneho riadeného procesu. Použili sme formu regulátora uvedenú v (3.4). Konštanty získané príkazom sme ladili na získanie čo najlepšieho regulačného pochodu. Na riadenie systému bol použitý PI regulátor s parametrami:

$$K_p = -2.53; K_i = -0.28 \tag{4.5}$$

V prípade použitia Fuzzy regulátora sme použili Fuzzy Toolbox programu MATLAB. Uvažovali dva vstupy do regulátora regulačnú odchýlku e a jej zmenu Δe. Výstupom z regulátora bola zmena rýchlosti vozidla Δv, čím sme vytvorili Fuzzy PI regulátor. Pre každý vstup sme vytvorili sedem jazykových premenných, ktoré môžu nadobúdať: veľmi kladná, stredne kladná, málo kladná, nulová, málo záporná, strede záporná a veľmi záporná. Kombináciou všetkých možných vstupov sme získali 49 pravidiel pre Fuzzy regulátor s konštantami:

$$K_e = 0.01; K_{de} = 0.02; K_u = 10 \tag{4.6}$$

V prípade použitia centralizovaného prístupu prediktívneho regulátora uvažujme formuláciu účelovej funkcie s integrátorom a zmäkčovacími premennými (3.19) s maximálnymi a minimálnymi ohraničeniami na akčný zásah v tvare (3.13b), mäkkými ohraničeniami na výstupy v tvare (3.18) a modelom v tvare (3.13d), (3.13e). Pričom zvolené hodnoty váh sú:

$$R = 0.01; Q = 100;\ Q_s = 10\,000 \tag{4.7}$$

Na rozdiel od centralizovaného prístupu, decentralizovaný systém ma k dispozícii menšie množstvo informácií, nedisponuje poznatkom o rýchlosti vozidla pred ním. Stav, ktorým je rýchlosť predošlého vozidla budeme preto modelovať Luenbergerovým pozorovačom (2.5).

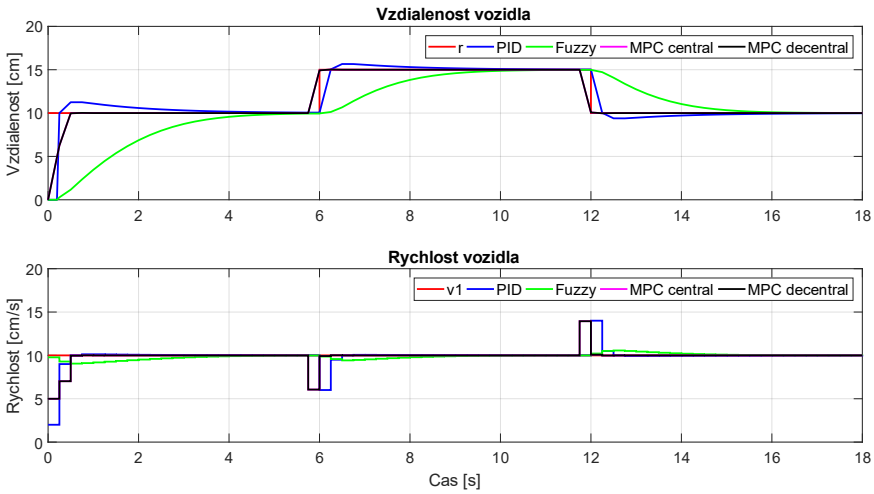
Využijeme tvar účelovej funkcie a ohraničení (3.16) doplnené o zmäkčovacie premenné (3.18). Pričom zvolené hodnoty váh sú:

$$R = 0.1; Q = 100;\ Q_s = 10\,000 \tag{4.8}$$

a hodnoty zosilnení Luenbergerovho pozorovača stavov a poruchy, získané príkazom programu MATLAB *dlqr*:

$$L_x = [0.2784 \quad 0.9917]; L_d = 0.0016 \tag{4.9}$$

Použitím vytvorených regulátorov sme získali priebehy (Obr. 17). Na obrázku je žiadaná veličina vzdialenosti medzi vozidlami ako aj rýchlosť vedúceho vozidla znázornená červenou farbou, riadiaca a riadená veličina PI regulátora modrou, Fuzzy regulátora zelenou, centralizovaného MPC purpurovou a decentralizovaného MPC čiernou farbou.



Obrázok 17 Simulačné porovnanie regulátorov

Dokázali sme, že modelovaný systém je možné riadiť všetkými uvažovanými regulátormi. Na základe poznatkov tejto časti možno zhodnotiť (Tab. 2).

Tabuľka 2 Simulačné porovnanie regulátorov

Porovnávacie kritérium	PI	Fuzzy	MPC central	MPC decentral
Čas výpočtu [ms]	0.26	3.61	6.96	6.74
SSO-E	211.34	251.01	164.04	170.57
SSO-U	7238	8314	7105	7136

V prípade výpočtovej náročnosti najlepšie obstál PI regulátor, čím len potvrdil implementačnú jednoduchosť. Správanie sa regulátora sme porovnávali na základe dvoch kritérií. SSO-E je suma štvorcov odchýlok vzhľadom na dodržanie regulačnej odchýlky, na základe tohto kritéria najlepšie obstál centralizovaný prediktívny regulátor. SSO-U je suma štvorcov odchýlok vzhľadom na veľkosť akčného zásahu, v prípade vozidiel predstavuje kritérium spotreby paliva. Aj toto kritérium mal najnižšie centralizovaný prediktívny regulátor, ale rozdiel v hodnotách všetkých porovnávaných kritérií bol medzi centralizovaným decentralizovaným prediktívnym regulátorom minimálny. V prípade posúdenia rýchlosti regulácie, najrýchlejšie dosiahne žiadanú hodnotu PI regulátor, ale to za cenu, že ako jediný dosahuje preregulovanie 1.25%.

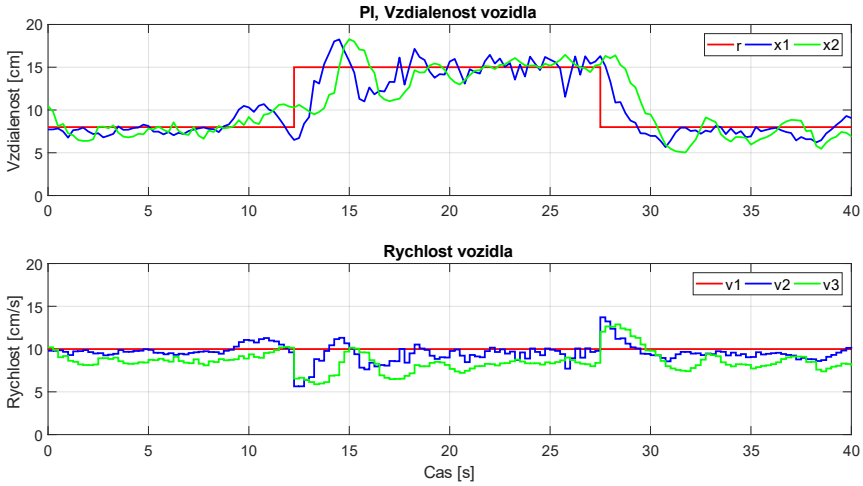
4.5 Riadenie reálneho systému

V tejto časti budeme realizovať riadenie reálneho systému robotických vozidiel PI regulátormi, centralizovanými a decentralizovaným MPC. V simuláciách vzchádzame z modelu v tvare (4.2) pre tri vozidlá. Na účely porovnania MPC regulátorov sme zvolili dve zmeny referencie, vzdialenosti medzi vozidlami, z hodnoty 10 cm na hodnotu 15 cm a naspäť. Na účely porovnania PI regulátorov sme zvolili dve zmeny referencie, vzdialenosti medzi vozidlami, z hodnoty 8cm na hodnotu 15 cm a naspäť. Uvažujme, že počiatočnú vzdialenosť od prekážky si vozidlá zistia použitím ultrazvukového snímača. Pre porovnanie berieme do úvahy porovnávacie kritéria SSO_E a SSO_U pre dodržanú vzdialenosť a akčný zásah tretieho vozidla.

4.5.1 PI riadenie

Vytvorili sme viacero PI regulátorov, ktoré sme testovali na riadenie sústavy robotických vozidiel Našli sme viaceré PI regulátory, ktoré riadia systém sústavy robotických vozidiel. Naše pokusy

ukázali, že zvýšenie K_p zlepšilo kvalitu regulačného pochodu, ale dodatočné navýšenie K_i už nemá efekt n zlepšenie regulačného pochodu. Z nášho porovnania najlepšie vyšiel PI regulátor, s použitím ktorého sme následne realizovali riadenie (Obr. 18). Na grafe je znázornená žiadaná hodnota vzdialenosti vozidiel a rýchlosť vedúceho vozidla červenou farbou, rýchlosť druhého

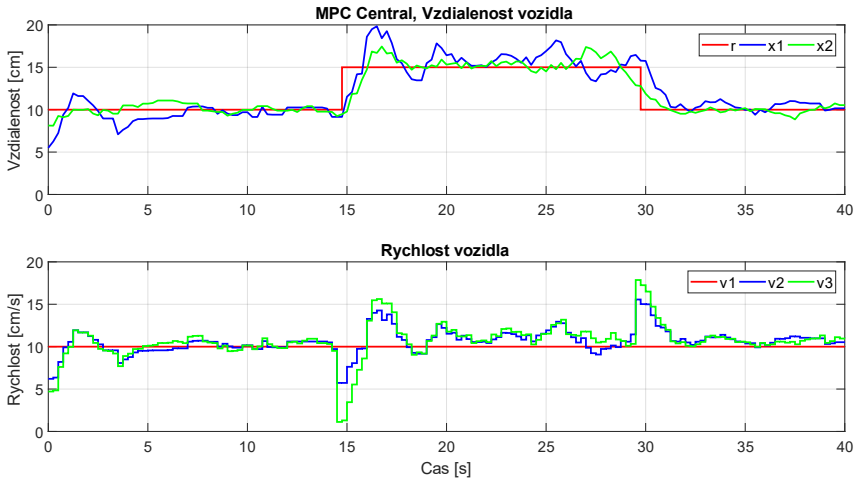


Obrázok 18 Riadenie použitím PI regulátora

vozidla a vzdialenosť, ktorú ňou dosahuje modrou, rýchlosť tretieho vozidla a vzdialenosť, ktorú ňou dosahuje zelenou farbou Z priebehu vidieť oneskorenie tretieho auta v porovnaní s druhým na referenciu. Tretie vozidlo dosiahlo hodnotu kritéria dodržania požadovanej vzdialenosti $SSO_E = 468$ a hodnotu kritéria spotreby paliva $SSO_U = 316\,340$.

4.5.2 Centralizované riadenie MPC

Uvažujme centralizovaný prístup k riadeniu použitím MPC podobne ako bol použitý v simulačnej časti. Použitím rôznych kombinácií váhových konštánt sme vytvorili viacero prediktívnych regulátorov, ktoré sme použili na riadenie sústavy robotických vozidiel. Pokusy tejto časti potvrdili teoretické poznatky, že vhodné zvýšenie Q zlepšuje kvalitu regulačného pochodu ale navýšenie R, ktoré penalizuje veľkosť zmeny akčného zásahu, už neprináša výrazné zmeny kvality riadenia. Z nášho porovnania najlepšie vyšiel regulátor, použitím ktorého sme získali riadenie (Obr. 19)



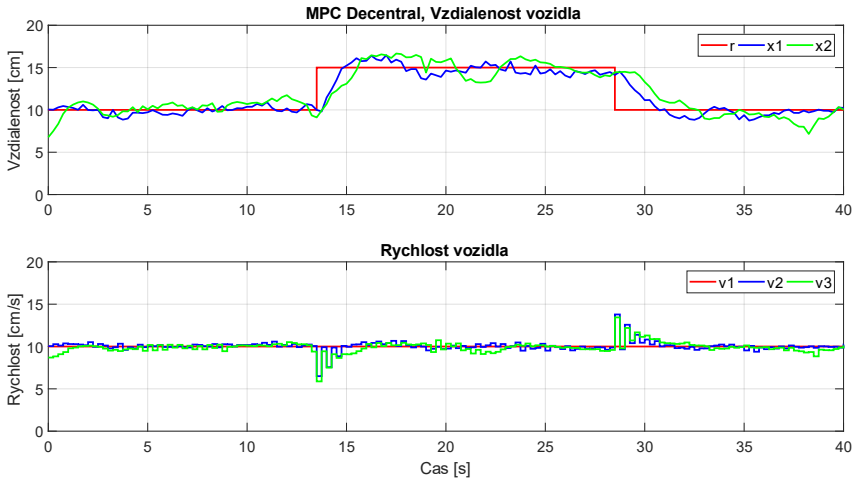
Obrázok 19 Riadenie použitím centralizovaného prediktívneho regulátora

Na tomto grafe je žiadaná hodnota vzdialenosti robotických vozidiel ako aj rýchlosť vozidla lídra znázornená červenou farbou, rýchlosť druhého vozidla a vzdialenosť, ktorú ňou dosahuje modrou, rýchlosť tretieho vozidla a vzdialenosť, ktorú ňou dosahuje zelenou farbou. V tomto prípade sa v porovnaní s PI riadením znížilo oneskorenie reakcie na zvýšenie žiadanej hodnoty. . Tretie vozidlo dosiahlo hodnotu kritéria dodržania požadovanej vzdialenosti $SSO_E = 432$ a hodnotu kritéria spotreby paliva $SSO_U = 246\,560$.

4.5.3 Decentralizované riadenie MPC

V prípade decentralizovaného prístupu k riadeniu použitím prediktívneho regulátora využijeme formuláciu, ktorá obsahuje modelovanie neznámych stavov a poruchy (3.17), použitím Luenbergerovho pozorovača. Kombináciou najlepších váhových konštánt z riadenia centralizovaným spôsobom a konštánt pozorovača sme vytvorili decentralizované prediktívne regulátory, ktoré sme použili na riadenie sústavy robotických vozidiel. Počas riadenia sme ladili hodnoty Luenbergerovho pozorovača s cieľom dosiahnuť najlepšiu možnú kvalitu riadenia. V tejto časti sme našli decentralizované prediktívne regulátory, ktoré vďaka modelovaniu stavov a poruchy dosahujú lepšiu kvalitu riadenia ako použité centralizované prediktívne regulátory a PI regulátory. Najlepším z navrhovaných bol decentralizovaný prediktívny regulátor, ktorého použitím sme získali riadenie (Obr. 24), kde je žiadaná hodnota vzdialenosti robotických vozidiel a rýchlosť vozidla určeného na nasledovanie znázornená červenou farbou, rýchlosť druhého

vozidla a vzdialenosť, ktorú ňou dosahuje modrou, rýchlosť tretieho vozidla a vzdialenosť, ktorú ňou dosahuje zelenou farbou.



Obrázok 20 Riadenie použitím decentralizovaného prediktívneho regulátora

V tomto prípade sa v porovnaní s predošlými riadeniami podstatnejšie znížilo oneskorenie reakcie na zvýšenie žiadanej hodnoty. Tretie vozidlo dosiahlo hodnotu kritéria dodržania požadovanej vzdialenosti $SSO_E = 391$ a hodnotu kritéria spotreby paliva $SSO_U = 214\,375$.

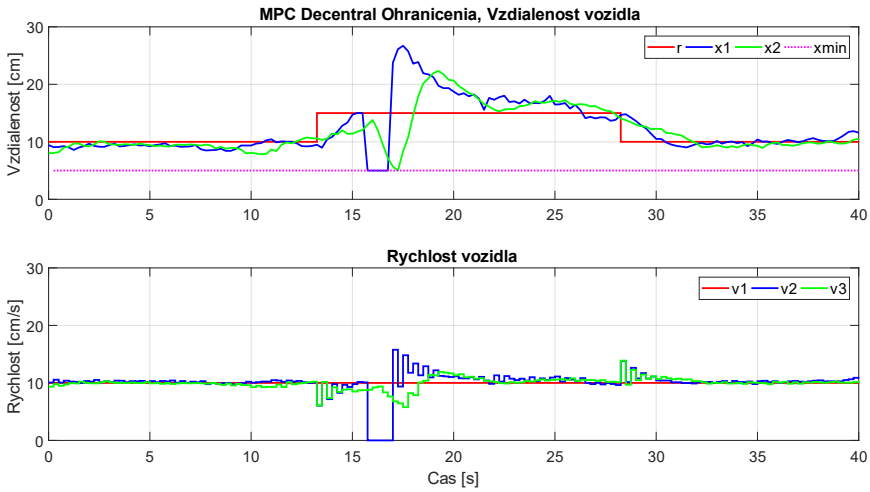
4.5.4 Zhodnotenie

Použitie MPC pre sústavu robotických vozidiel zabezpečuje lepšiu kvalitu riadenia ako klasický PI regulátor. V porovnaní centralizovaného a decentralizovaného prístupu má teoretickú výhodu centralizovaný kvôli väčšiemu množstvu informácií, s ktorými disponuje. V tejto práci sme však vhodným modelovaním neznámych stavov a porúch decentralizovaného systému tento problém eliminovali. Navrhnutý decentralizovaný prediktívny regulátor vybavený pozorovačom stavov a poruchy dosiahol najnižšie hodnoty porovnávacích kritérií. Výhodou MPC oproti všetkým ostatným uvažovaným regulátorom v tejto práci je možnosť zohľadnenia či už systémových, alebo bezpečnostných ohraničení (Tab. 3).

Tabuľka 3 Porovnanie regulátorov

Porovnávacie kritérium	**PI**	**MPC central**	**MPC decentral**
SSO-E	468	432	391
SSO-U	316340	245560	214375

Teraz si demonštrujeme schopnosť práce s ohraničeniami, použijeme náš najlepší regulátor Decentralizované MPC 3. Súčasťou ohraničení je minimálna bezpečná vzdialenosť 5 cm na ktorú sa pokúsime naraziť dočasným zastavením druhého vozidla. Získavame priebeh, kde žiadaná hodnota vzdialenosti robotických vozidiel znázornená červenou farbou, rýchlosť druhého vozidla a vzdialenosť, ktorú ňou dosahuje modrou, rýchlosť tretieho vozidla a vzdialenosť, ktorú ňou dosahuje zelenou farbou, hodnota ohraničenia purpurovou farbou (Obr. 21).



Obrázok 21 Narazenie na ohraničenia

V čase vzniku nebezpečnej situácie vzdialenosť vozidiel druhého vozidla od prekážky a tretieho vozidla od druhého postupne klesá, na čo obe zareagujú znížením rýchlosti. Dokonca v prípade úplného zastavenia druhého vozidla a spomalenia tretieho nedôjde ku priblíženiu sa prekážke na menšiu vzdialenosť ako je povolená v obmedzeniach regulátora. Po následnom odstránení prekážky obe vozidlá zrýchlia a opäť dosiahnu a udržia referenciu.

5 Záver

Odbor automatizácie sa zaoberá metódami návrhu riadiacich algoritmov. V oblasti riadenia vozidiel v cestnej premávke je v súčasnosti vážnym problémom nehodovosť. Pri nehodách dochádza ku škodám ekonomickým a ekologickým, no častokrát aj k stratám na životoch. Táto práca ponúka riešenie problému nehodovosti v podobe nahradenia ľudského vodiča za riadiaci algoritmus. Riadeným systémom, pre ktorý sme navrhli riadiaci algoritmus je sústava robotických vozidiel.

V úvode práce opisujeme teoretické aspekty matematickej optimalizácie, modelovania systémov a porovnávame algoritmy používané na riadenie procesov od implementačne najjednoduchšieho PID až po prediktívne riadenie s modelom, MPC. V porovnaní s ostatnými uvažovanými regulátormi prediktívny regulátor je ako jediný schopný zahrnúť do výpočtov ohraničenia, limitovať stavy, vstupy alebo výstupy procesu. MPC rieši problematiku riadenia ako optimalizačný problém. Optimalizácia je proces zlepšovania súčasného stavu, ktorého výsledkom je optimálny zásah do systému, ten najlepší možný vzhľadom na dodržanie všetkých obmedzení.

V tejto práci sme predstavili kroky potrebné na implementáciu prediktívnych regulátorov pre centralizovaný a decentralizovaný prístup k riadeniu. V prípade decentralizovaného spôsobu riadenia je každé vozidlo ovládané svojim regulátorom, ktorý má informácie len o vozidle, ktoré ovláda. V prípade centralizovaného prístupu riadi jeden centrálny regulátor sústavu vozidiel ako celok, má k dispozícií všetky dostupné informácie o všetkých vozidlách. Simulačná časť obsahuje modelové závislosti priebehov riadenia sústavy robotických vozidiel vybranými regulátormi. Správnosť simulácií sme následne demonštrovali riadením reálneho systému sústavy robotických vozidiel PI a prediktívnymi regulátormi. Správnosť fungovania prediktívnych regulátorov sme dokázali nielen na základe kritéria kvality riadenia, ale aj simulovaním situácie pri ktorej riadený systém narazil na svoje obmedzenia. Práca obsahuje grafické závislosti priebehov riadenia použitými regulátormi.

Jedným z hlavných cieľov práce bolo nielen vytvoriť riadiaci systém s použitím prediktívneho riadenia, demonštrovať jeho funkčnosť, ale aj oboznámiť čitateľa s procesom jeho tvorby. Kódy v jazyku MATLAB, vytvorené a použité v tejto práci je možné nájsť na:

https://github.com/aim522/Control-of-Robotic-Vehicles

Zoznam použitej literatúry

1. Jerz, V.: Simulácia a optimalizácia v procese prijímania rozhodnutí. Habilitačná práca. Bratislava: SjF STU 2009
2. Maixner, L. et alteri: Automatizace a automatizační technika 2: Systémové pojetí automatizace., Computer Press, Brno, 2014.
3. Boyd S. & Vandenberghe L.: Convex Optimization, New York: Cambridge Univ. Press, 2004
4. Dupačová, J. & Lachout, P.: Úvod do optimalizace, MatfyzPress, Praha, 2011.
5. Hamala, M. & Trnovská, M.: Nelineárne programovanie, Epos, Bratislava, 2012.
6. Mészáros, A.: Prednášky k predmetu Modelovanie, 2014.
7. Kanasewich, E. R.: Time sequence analysis in geophysics (3rd ed.). University of Alberta, 1981.
8. Vasilyev, A.S. & Ushakov, A.V.: Modeling of dynamic systems with modulation by means of Kronecker vector-matrix representation, Scientific and Technical Journal of Information Technologies, Mechanics and Optics, 2015.
9. Yi, R. et alteri: Econometric Analyses of International Housing Markets, Routledge, 2016.
10. Maxwell, J.C.: On Governors, Proc. Royal Soc. London, vol. 16, pp. 270-283, 1868.
11. Routh, E.J.: A Treatise on the Stability of a Given State of Motion, London: Macmillan & Co., 1877.
12. Vyshnegradsky, I.A.: On Controllers of Direct Action, Izv. SPB Tekhnolog. Inst., 1877.
13. Bissell, C. C: Stodola, Hurwitz and the genesis of the stability criterion. Int. J. Control. 50, 1989.
14. Pontryagin, L.S. et alteri: The Mathematical Theory of Optimal Processes, New York: Wiley, 1962.
15. Motchenbacher, C.D. & Connelly, J. A.: Low-noise electronic system design. [s.l.] : Wiley Interscience, 1993.
16. Sontag, E.: Mathematical Control Theory: Deterministic Finite Dimensional Systems. Second Edition. Springer. ISBN 0-387-98489-5, 1998.
17. Mikleš, J. & Fikar, M.: Modelovanie, identifikácia a riadenie procesov 2, Vydavateľstvo STU, Bratislava, 2004.

18. King, M.: Process Control: A Practical Approach. Second edition.John Wiley & Sons Inc., Chichester, West Sussex, United Kingdom, 2016.
19. Bennett, S.: A history of control engineering, 1930-1955. IET. p. 48, 1992.
20. Bennett, S.: Development of the PID Controller. Control Systems, IEEE. 13. 58 - 62, 1994.
21. Beneš, P. et alteri: Automatizace a automatizační technika 3: Přostredky automatizační techniky., Computer Press, Brno, 2014.
22. Bakošová, M. & Fikar, M.: Radenie procesov, Nakladateľstvo STU, Bratislava, 2012.
23. Ziegler, J.G & Nichols, N. B.: Optimum settings for automatic controllers. Transactions of the ASME. 64: 759–768, 1942.
24. Åström, K.J.: Introduction to Stochastic Control Theory, New York: Academic Press, 1970.
25. Kalúz, M: Multipurpose and Low-Cost Architectures for Automatic Control Remote Laboratories. ÚIAM FCHPT STU v Bratislave, 2014.
26. Zadeh, L. A.: Fuzzy sets and systems, In: International Journal of General Systems, 17:2, 129 — 138, 1990.
27. Horňák, M: Fuzzy riadenie destilačnej kolóny. Bakalárska práca, ÚIAM FCHPT STU, Bratislava, 2016.
28. Nunes, G. C.: Design and Aalysis of Multivariable Predictive Control Applied to an Oil-water-gas Separator: A Polynomial Approach, University of Florida, 2001.
29. Holaza, J: Fast and Memory-Efficient implementation of Model Predictive Control. ÚIAM FCHPT STU v Bratislave, 2016.
30. Wang, L: Model Predictive Control System Design and Implementation Using MATLAB®. Springer Science & Business Media. pp. xii, 2009.
31. Nikolaou, M: Model predictive controllers: A critical synthesis of theory and industrial needs, Advances in Chemical Engineering, Academic Press, Volume 26, Pages 131-204, 2001.
32. Kalúz, M. et alteri: A Robotic Traffic Simulator for Teaching of Advanced Control Methods. In IFAC-PapersOnLine ∴. s. 338–343.
33. Eurostat, the statistical office of the European Union, Over 25 000 victims of road accidents in the EU in 2016, Eurostat. 19.11.2017, Dostupné na: ec.europa.eu/eurostat/web/products-eurostat-news/-/EDN-20171119-1
34. Kesting, A. et alteri: Adaptive cruise control design for active congestion avoidance. Transportation Research Part C: Emerging Technologies, 2008.

35. Dezani, H. et alteri:Controlling traffic jams on urban roads modeled in coloured petri net using genetic algorithm. 38th Annual Conference on IEEE Industrial Electronics Society, 2012.
36. Zegeye, S. et alteri:Reduction of travel times and traffic emissions using model predictive control. American Control Conference, 2009.
37. Deo, P. et alteri: Model Predictive Control for Multi-Class Traffic Flows.12th IFAC Symposium on Control in Transportation Systems, 2009.
38. Martinec, D., Sebek, M., and Hurák, Z.: Vehicular platooning experiments with racing slot cars. CCA, IEEE International Conference, 2012.
39. Marcolino, L. and Chaimowicz, L.: Traffic control for a swarm of robots: Avoiding target congestion. Intelligent Robots and Systems, 2009.
40. tbxManager, doplnok programu MATLAB, Dostupné na: http://www.tbxmanager.com/