SLOVENSKÁ TECHNICKÁ UNIVERZITA V BRATISLAVE
FAKULTA CHEMICKEJ A POTRAVINÁRSKEJ TECHNOLÓGIE

Evidenčné číslo: FCHPT-5415-61829

HIERARCHICKÉ RIADENIE SKUPINY VOZIDIEL

BAKALÁRSKA PRÁCA

Bratislava 2014 **Filip Janeček**

SLOVENSKÁ TECHNICKÁ UNIVERZITA V BRATISLAVE
FAKULTA CHEMICKEJ A POTRAVINÁRSKEJ TECHNOLÓGIE

Evidenčné číslo: FCHPT-5415-61829

HIERARCHICKÉ RIADENIE SKUPINY VOZIDIEL

BAKALÁRSKA PRÁCA

Študijný program: automatizácia, informatizácia a manažment v chémii a potravinárstve
Číslo študijného odboru: 2621
Názov študijného odboru: 5.2.14 automatizácia, 5.2.52 priemyselné inžinierstvo
Školiace pracovisko: Ústav informatizácie, automatizácie a matematiky
Vedúci záverečnej práce: Doc. Ing. Michal Kvasnica, PhD.
Konzultant: Ing. Martin Kalúz

Bratislava 2014 **Filip Janeček**

Slovenská technická univerzita v Bratislave
Ústav informatizácie, automatizácie a matematiky

Fakulta chemickej a potravinárskej technológie
Akademický rok: 2013/2014
Evidenčné číslo: FCHPT-5415-61829

STU FCHPT

ZADANIE BAKALÁRSKEJ PRÁCE

Študent:	**Filip Janeček**
ID študenta:	61829
Študijný program:	automatizácia, informatizácia a manažment v chémii a potravinárstve
Kombinácia študijných odborov:	5.2.14 automatizácia, 5.2.52 priemyselné inžinierstvo
Vedúci práce:	doc. Ing. Michal Kvasnica, PhD.
Konzultant:	Ing. Martin Kalúz

Názov práce: **Hierarchické riadenie skupiny vozidiel**

Špecifikácia zadania:

Cieľom práce je navrhnúť hierarchický riadiaci systém určený na ovládanie skupiny vozidiel, ktoré sa pohybujú po zakryvenej trajektórii za sebou. Úlohou riadenia je ovládať rýchlosť a smer vozidiel tak, aby nedochádzalo ku kolíziám, aby si vozidlá udržiavali konštantný rozostup.

Úlohy:
* Vytvoriť komunikačné rozhranie medzi vozidlami a počítačom.
* Navrhnúť a implementovať riadiaci systém, ktorý určí vhodnú rýchlosť každého vozidla tak, aby bol udržaný konštantný rozostup a vozidlá sa pohybovali požadovanou rýchlosťou.
* Experimentálne verifikovať získané výsledky.

Rozsah práce: 30

Zoznam odbornej literatúry:

1. 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: 17. 02. 2014

Dátum odovzdania práce: 24. 05. 2014

L. S.

Filip Janeček
študent

prof. Ing. Miroslav Fikar, DrSc.
vedúci pracoviska

prof. Ing. Miroslav Fikar, DrSc.
garant študijného programu

Poďakovanie

Chcel by som poďakovať môjmu školiteľovi Doc. Ing. Michalovi Kvasnicovi, PhD. za pomoc pri pochopení problematiky prediktívneho riadenia, pomoc pri návrhu riadenia pre skupinu vozidiel a dobre mienené rady pri realizovaní mojej práce. Ďalej by som chcel poďakovať môjmu konzultantovi Ing. Martinovi Kalúzovi za pomoc pri riešení technických problémov s robotmi.

Abstrakt

V bakalárskej práci sme sa bližšie pozreli na prediktívne riadenie a jeho využiteľnosť pri riadení skupiny vozidiel. V prvej časti sme sa zaoberali teóriou prediktívneho riadenia, aké sú jeho výhody a čo je potrebné na jeho implementáciu. V druhej časti sme tieto teoretické princípy využili na vytvorenie matematického modelu skupiny vozidiel pre decentralizovaný aj centralizovaný systém, ktorý sme použili na simuláciu riadenia skupiny vozidiel ako aj na riadenie reálnych robotov a reprezentovali sme výsledky graficky aj matematicky v podobe hodnôt účelovej funkcie. Na základe týchto výsledkov sme porovnali decentralizovaný systém s centralizovaným systémom. Na komunikáciu medzi robotmi a PC boli použité triedy vytvorené v Matlabe.

Abstract

In bachelor thesis we took a closer look at model predictive control and its usability in the control of group of vehicles. In the first part we discussed the theory of model predictive control, what are its benefits and what is necessary for its implementation. In the second part we used these theoretical principles to create a mathematical model of group of vehicles for decentralized and centralized system, which we used to simulate the control of group of vehicles as also to control the real robots and we represented the results graphically and mathematically in the form of objective function values. Based on these results, we compared the decentralized system with the centralized system. Classes created in Matlab were used for communication between robots and PC.

Obsah

Úvod

V dnešnej dobe všetko smeruje k automatizácii množstva procesov, čím sa uplatňuje snaha o zníženie nákladov na pracovníkov, zvýšenie bezpečnosti a obmedzenie vplyvu ľudského faktora na tieto činnosti. Prejavuje sa to okrem iných aj v automobilovom priemysle.

Nikto z nás nemá rád státie v ranných kolónach áut, ktoré spôsobujú zdržania. Sú spôsobené tým, že človek nedokáže dostatočne rýchlo reagovať na pohýnajúce sa auto pred sebou, a preto vznikajú časové oneskorenia - tvoria sa kolóny. Keby mal na starosti pohyb auta v týchto situáciách počítač, ktorý by na základe vzdialenosti auta pred nami a jeho predchádzajúceho pohybu dokázal povedať aká má byť naša rýchlosť, aby sme si udržali konštatný rozostup, dokázali by sme tieto oneskorenia minimalizovať v závislosti od periódy vzorkovania, ktorá je u počítača neporovnateľne kratšia, ako reflex človeka.

Cieľom tejto práce je navrhnúť riadiaci systém na základe prediktívneho riadenia (*MPC - Model Predictive Control*), ktorý by sa mohol inštalovať do vozidiel, a bol by aktivovaný vždy, keď sa auto dostane do kolóny. Ďalej porovnáme tento systém, ktorý je decentralizovaný, s centralizovaným systémom. Tento by sa nemohol používať tak jednoducho v praxi, ale bol by využiteľný napr. vo firmách, ktoré prevážajú tovar viacerými nákladnými autami. Teda je známy počet vozidiel, ktoré navzájom dokážu komunikovať - odovzdávať si informácie o svojom stave - rýchlosti a vzdialenosti od auta pred sebou. Tým sa zvýši výkon riadiaceho systému, lebo bude môcť oveľa lepšie určiť, aká má byť rýchlosť na udržanie konštantného rozostupu. Obidva systémy sú zamerané hlavne na bezpečnosť, teda dôležité je udržanie rozostupu.

V prvej časti práce sme sa snažili priblížiť teóriu MPC a v druhej časti najprv simulovať situácie pre decentralizovaný a centralizovaný model, ako aj ich vzájomné porovnanie. Tieto algoritmy sme neskôr aplikovali na reálnych robotov - autá, ktoré chodia po čiare za sebou.

1 Čo je to MPC?

Prediktívne riadenie (*MPC - Model Predictive Control*) je pokročilá metóda procesného riadenia, ktorá sa používala hlavne v chemických továrňach a ropných rafinériách od osemdesiatych rokov 20. storočia. Toto riadenie využíva dynamický model procesu, veľmi často lineárny model získaný identifikáciou systému na záklkade nameraných údajov.
Dnes sa MPC čoraz viac začína využívať aj pri rýchlych procesoch, čo je umožnené veľmi rýchlym rozvojom hardwarovej techniky - teda počítače sú schopné neporovnateľne rýchlejšie počítať optimálny akčný zásah, ktorého výpočet je pri MPC oveľa zložitejší ako napr. pri PID regulátore.
Hlavná výhoda MPC je to, že optimalizuje terajší stav a zároveň sa "pozerá"do budúcnosti, ráta s budúcim stavom systému. To znamená, že MPC optimalizuje určitý časový horizont, ale uplatňuje iba súčasný stav a má schopnosť predvídať budúce udalosti a tým poskytnúť lepšie riadenie. PID a LQR regulátory nemajú túto schopnosť predvídať. MPC je digitálny regulátor.
Ďalšou výhodou MPC je možnosť implementovať a dodržiavať ohraničenia všetkých veličín, teda vstupov, výstupov, aj stavov. Keďže v praxi sa ohraničenia vyskytujú skoro vždy, je pochopiteľné, že MPC je veľmi rozšírené. Sú to napr. obmedzenia akčných zásahov, kedy nesmie byť prekročený prietok kvapaliny, alebo zmena rýchlosti sa musí udržať v nejakom intervale, obmedzenia na stavy, kedy hladina v zásobníku nesmie prekročiť určitú hranicu, lebo by mohlo dôjsť ku havárii, a pod. [1]

1.1 Základné aspekty MPC

Prediktívne riadenie predstavuje súbor riadiacich metód, kde sa na základe predikcíi účinkov riadiaceho algoritmu na riadený výstup používa model riadeného systému na určenie riadiacich vstupov. Tieto metódy obsahujú nasledujúce základné aspekty:

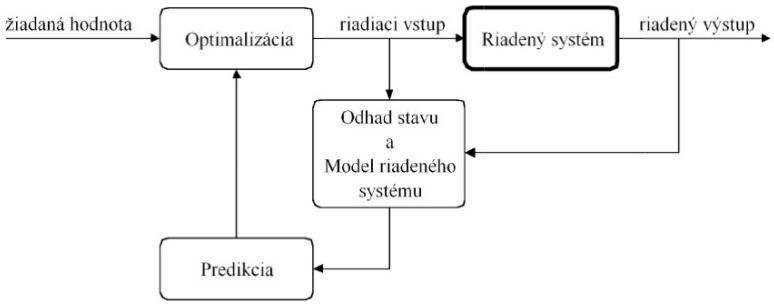
- Pomocou informácií o systéme, ktoré sú aktuálne dostupné a podľa zadanej cesty riadenia sa na predpoveď budúceho výstupu riadenia používa matematický model systému.
- Cieľom je minimalizovať určité *kritérium riadenia* na určitom *predikčnom horizonte* (časovom intervale). Tento sa limitne môže blížiť k nekonečnu kvôli lepšej stabilizácii riadenia.
- Pre daný okamih je použitý iba prvý člen určenej postupnosti riadenia a z nových informácií o stave systému sa určí vždy nová postupnosť krokov. Tento postup sa

nazýva *stratégia pohyblivého horizontu.*

- MPC umožňuje jednoduchú manipuláciu s *obmedzeniami* všetkých veličín (vstupných, stavových aj výstupných) [2].

1.2 Základné problémy MPC

Väčšina prístupov k prediktívnemu riadeniu je formulovaná stavovo. Na obrázku 1.1 vidíme všeobecnú štruktúru stavovej reprezentácie. Funkcie jednotlivých blokov:



Obr. 1.1: Všeobecná štruktúra prediktívneho riadenia. [2]

- *Optimalizácia* - tu sú zahrnuté všetky obmedzenia a hlavnou črtou je, že pracuje kontinuálne. Na základe žiadanej hodnoty určuje riadiacu postupnosť vstupu tak, aby žiadaná hodnota bola čo najlepšie sledovaná riadeným výstupom.
- *Odhad stavu* - tu sa odhaduje súčasný stav, ktorý je potrebný pre blok predikcie.
- *Model riadeného systému* - v tomto bloku sa nachádza matematický model riadeného systému.
- *Predikcia* - veľmi podstatná časť prediktívneho riadenia, tieto algoritmy dávajú budúce vstupné a stavové hodnoty.

Toto je základná štruktúra regulátora pre prediktívne riadenie. Bloky môžu byť pre rôzne prípady modifikované a tiež doplnené o ďalšie, ako napr. identifikácia systému, zisťovanie poruchy, atď. [2]

1.3 Matematické vyjadrenie MPC

Základ MPC tvorí účelová funkcia a obmedzenia, ktoré sú tvorené predikčnou rovnicou a obmedzeniami vstupných, stavových a výstupných veličín. Takto vyzerá matematický zápis:

$$\min \left[\sum_{k=0}^{N-1} (u_{t+k}^T Q_u u_{t+k} + x_{t+k}^T Q_x x_{t+k}) \right] \tag{1.1}$$

$$s.t. \quad x_{t+1} = Ax_t + Bu_t \tag{1.2}$$

$$x_t = x(t) \tag{1.3}$$

$$x_t \in \mathbb{X} \qquad \text{a} \qquad u_t \in \mathbb{U} \tag{1.4}$$

kde kritérium (1.1) je účelová funkcia, N je horizont predikcie, u_{t+k} je vektor vstupov a x_{t+k} je vektor stavov v čase $t+k$, matice Q_u a Q_x sú váhové matice, $\mathbb{U}$ a $\mathbb{X}$ sú množiny, v ktorých sa môžu nachádzať hodnoty vstupov a stavov. Stavový opis modelu je vyjadrený rovnicou (1.2). [3]

Úloha sa nazýva problém riadenia na konečnom horizonte ak N je konečná hodnota, alebo problém riadenia na nekonečnom horiznte ak $N \to \infty$. Počiatok stavového priestoru $x = 0$ a $u = 0$ sa musí nachádzať vo vnútri zlúčiteľnej oblasti, ktorá vyhovuje (1.4). [2]

1.4 Obmedzenia

Ako sme už spomínali, obmedzenia sa v reálnych procesoch vyskytujú takmer vždy. Preto je veľmi dôležité, aby systém riadenia dokázal tieto obmedzenia dodržať. Ak by sme riadili proces, ktorý má obmedzenia, a nebrali by sme ich pri riadení do úvahy, mohlo by sa stať (a je to veľmi pravdepodobné), že riadiaci systém by generoval akčné zásahy, ktoré by tieto obmedzenia prekračovali, a tým by nemuseli byť dodržané ani obmedzenia stavov. Je síce pravda, že by sme mohli tieto akčné zásahy orezať, ale to by potom mohlo viesť buď k nestabilite riadenia systému, alebo by nebola dosiahnutá hodnota žiadanej veličiny.
Nakoniec obmedzenia môžu byť využívané na dosiahnutie lepšej kvality riadenia ako ladiace parametre regulátora. [2]

Ohraničenia môžme z hľadiska charakteru rozdeliť do nasledujúcich skupín: [4]

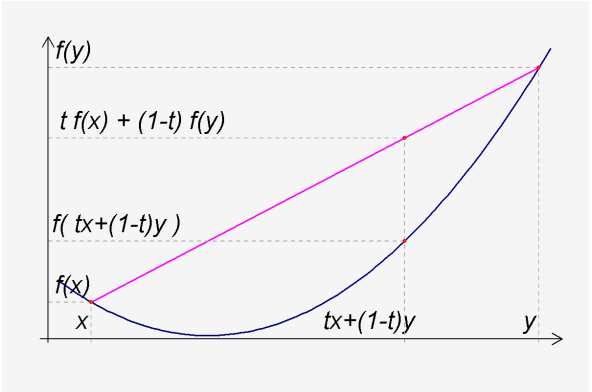
- *Fyzikálne ohraničenia* - hodnoty vstupov sa môžu pohybovať len vo vopred určenom rozsahu a s obmedzenou rýchlosťou.
- *Technolgické ohraničenia* - hodnoty riadených výstupov a stavov musia byť udržané vo vopred určenom intervale, aby bola dodržaná kvalita produkcie.
- *Bezpečnostné ohraničenia* - nesmú byť prekročené hodnoty pomocných veličin, inak by mohlo dôjsť k havárii zariadenia.
- *Stabilizujúce ohraničenia* - týkajú sa stavov a zaručujú, že riadenie procesu bude stabilné.

1.5 Konvexná optimalizácia v MPC

Keďže MPC je oveľa náročnejšie na výpočtovú náročnosť oproti iným spôsobom riadenia, je potrebné to brať do úvahy pri jeho návrhu a zavádzaní. Výpočtová technika musí byť schopná vypočítať akčný zásah za čas menší alebo nanajvýš rovný perióde vzorkovania, t.j. vyriešiť optimalizačný problém so všetkými obmedzeniami v každej perióde vzorkovania.
Na tento účel je veľmi vhodná konvexná optimalizácia, teda účelová funkcia aj podmienky obmedzení sú konvexné. To znamená, že musia spĺňať nasledujúcu požiadavku:

$$f(tx + (1-t)y \leq tf(x) + (1-t)f(y) \tag{1.5}$$

a to musí platiť pre všetky $x, y \in \mathbb{R}^n$ a $t \in [0, 1]$.
Na obrázku 1.2 vidíme graf funkcie, ktorá je kvadratická a zároveň spĺňa podmienku (1.5), takže je konvexná.



Obr. 1.2: Graf konvexnej funkcie. [6]

2 Riadenie skupiny vozidiel

V dnešnej dobe sú čoraz častejšie snahy o zavedenie prediktívneho riadenia do automobilového priemyslu. Dôvodom je prudké zvyšovanie počtu vozidiel, čo má za následok spomaľovanie dopravy, viac nehôd a zlý dopad na životné prostredie. Výsledkom implementácie MPC by malo byť zlepšenie situácie na cestách, zníženie počtu nehôd, zvýšenie plynulosti premávky a v neposlednom rade aj šetrenie životného prostredia.

Pôvodná myšlienka inštalovania riadiaceho systému priamo do palubných počítačov áut dnes už nie je nereálna, ale ani nevyhnutná. Otvára sa myšlienka, že by takýto riadiaci systém mohol byť implementovaný samotným užívateľom, napr. vo forme poťahu na volant, ako to zrealizoval Arjun Iyer z East Falmouth v Massachussets [7]. Tento systém je založený na komunikácii s rovnakými systémami v okolitých autách cez bluetooth. Pomocou vibrácii upozorňuje vodiča buď na prílišné zrýchľovanie alebo brzdenie.

V našom prípade sme sa snažili pomocou riadenia rýchlosti auta v závislosti od jeho aktuálneho rozostupu a rýchlosti auta pred ním udržať žiadaný rozostup medzi autami. Tento rozostup sme merali v $[cm]$ ultrazvukovým snímačom, ktorý sa nachádzal vpredu na robotovi. Takýchto áut sme mali k dispozícii viac, pri našej práci sme používali štyri autá. Prvému autu sme vždy nastavili rýchlosť, a jeho rozostup nás nezaujímal (iba ak by malo do niečoho naraziť, vtedy zastalo). Rýchlosť ďalších áut bola nastavovaná podľa výpočtu MPC, vždy podľa konkrétneho systému riadenia, buď decentralizovaného, kedy riadené auto registruje iba auto pred sebou, alebo centralizovaného, kedy každé auto registruje všetky ostatné.

V ďalšej časti si bližšie ukážeme, ako konkrétne systémy riadenia fungujú, ako sme vytvárali ich matematické modely a ako sme implementovali systém riadenia priamo na roboty.

2.1 Decentralizovaný systém

Napriek tomu, že decentralizovaný systém by teoreticky mal zabezpečiť horšie riadenie ako centralizovaný, z praktického hľadiska je využiteľnejší. Je to preto, lebo nepotrebuje informácie o počte a stave okolitých áut, stačia mu informácie o aute pred ním. Takýto model by bol využiteľný aj v reálnom živote, napr. v kolónach, na semaforoch a križovatkách. Keď by sa auto dostalo do kolóny, jednoducho by sa aktivoval riadiaci systém, a ten by sa postaral o pridávanie, resp. uberanie rýchlosti v závislosti od vzdialenosti od auta pred nami. Keďže v doprave sú prítomné mnohé obmedzenia a MPC je schopné ich brať do úvahy, je na riešenie takéhoto problému veľmi vhodné.

Síce je tento systém z hľadiska implementácie jednoduchší, pre potreby riadenia v našich

podmienkach sme museli poznať počet vozidiel, ktoré budeme riadiť, lebo algoritmus riadenia nebol nahratý priamo v každom robotovi, ale riadenie prebiehalo na základe komunikácie s Matlabom, ktorý počítal optimálne akčné zásahy.

Riadenie prebiehalo nasledovne:

Keď sme spustili riadenie, prvé auto (líder) sa začalo pohybovať. Zmerali sa rozostupy pre všetky autá a následne sa postupne vypočítali optimálne akčné zásahy pre každé auto, v závislosti od aktuálnej rýchlosti predchádzajúceho auta. V praxi, ak by malo každé vozidlo nahratý vlastný algoritmus riadenia, tak by tieto riadiace systémy bežali nezávisle od seba. No v našich podmienkach museli nasledovať za sebou, čo ale nepredstavovalo veľké oneskorenia, lebo pri decentralizovanom systéme je tento výpočet oveľa jednoduchší. Hlavný princíp spočíva v tom, že predchádzajúce auto je vlastne líder pre riadené auto.

2.1.1 Matematický model

Na riešenie nášho problému si musíme vytvoriť matematický model, ktorý je nutné diskretizovať. Pre decentralizovaný systém riadenia berieme do úvahy len auto pred sebou, nestaráme sa o ďalšie autá. V našom prípade auto ide len dopredu po vopred určenej trajektórii, a teda jediná veličina, ktorú budeme meniť, je rýchlosť auta.
Za stavové veličiny si zvolíme vzdialenosť od auta pred nami a jeho rýchlosť. Riadiacou veličinou bude rýchlosť nášho auta.
Na zostavenie matematického modelu musíme využiť poznatky z fyziky. Derivácia dráhy podľa času je rýchlosť a derivácia rýchlosti podľa času je zrýchlenie. Aby sme mohli vytvoriť diskretizovaný model, derivácie nahradíme diferenciami. Označme vzdialenosť od auta pred nami d, jeho rýchlosť v_0 a rýchlosť riadeného auta v. Dostávame spojitý model:

$$\dot{d} = v_0 - v \tag{2.1}$$

$$\dot{v_0} = 0 \tag{2.2}$$

Predpokladáme, že v čase akčného zásahu je rýchlosť auta pred nami konštantná, preto derivácia (2.2) je nulová. Maticový tvar spojitého modelu má tvar:

$$\begin{bmatrix} \dot{d} \\ \dot{v_0} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} d \\ v_0 \end{bmatrix} + \begin{bmatrix} -1 \\ 0 \end{bmatrix} v \tag{2.3}$$

Spojitý model má tvar:

$$\dot{x}(t) = Ax(t) + Bu(t) \tag{2.4}$$

a matice spojitého modelu pred diskretizáciou:

$$A = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \quad B = \begin{bmatrix} -1 \\ 0 \end{bmatrix} \quad C = \begin{bmatrix} 1 & 0 \end{bmatrix} \quad D = 0 \tag{2.5}$$

Spojitý model musíme diskretizovať:

$$\frac{d(t + \Delta t) - d(t)}{\Delta t} \approx \dot{d} = v_0 - v \tag{2.6}$$

Rovnicu (2.6) upravíme:

$$d(t + \Delta t) = d(t) + \Delta t(v_0 - v) \tag{2.7}$$

Za predpokladu konštantnej rýchlosti auta pred nami v čase akčného zásahu:

$$v_0(t + \Delta t) = v_0(t) \tag{2.8}$$

Maticový tvar diskretizovaného modelu získame prepísaním rovníc (2.7) a (2.8):

$$\begin{bmatrix} d \\ v_0 \end{bmatrix} = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} d \\ v_0 \end{bmatrix} + \begin{bmatrix} -\Delta t \\ 0 \end{bmatrix} v \tag{2.9}$$

kde ľavá strana rovnice (2.9) je v čase $t + \Delta t$ a pravá v čase t.

Týmto postupom sme sa dopracovali k stavovému opisu v tvare:

$$x(t + \Delta t) = Ax(t) + Bu(t) \tag{2.10}$$

Obmedzenia vstupov sú $u_{min} \leq u \leq u_{max}$, čo sú obmedzenia pre rýchlosť riadeného auta v. Vektor stavov x obsahuje dve veličiny: d - rozostup medzi autami a v_0 - rýchlosť auta pred nami. Matice pre náš stavový opis modelu sú:

$$A = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix} \quad B = \begin{bmatrix} -\Delta t \\ 0 \end{bmatrix} \quad C = \begin{bmatrix} 1 & 0 \end{bmatrix} \quad D = 0 \tag{2.11}$$

2.2 Centralizovaný systém

Keďže centralizovaný systém potrebuje na svoju realizáciu informácie o celej skupine vozidiel, teda počet vozidiel, rýchlosť každého vozidla a všetky rozostupy, nie je prakticky využiteľný v bežnom živote, lebo nikdy nevieme, koľko áut sa v kolóne nachádza, a ani nedokážeme komuinkovať so všetkými autami. No ak by teoreticky niečo také bolo možné, takýto systém riadenia by mal poskytnúť lepšie výsledky riadenia, lebo by zohľadnil stav kolóny ako celku, a na základe toho by poslal optimálny akčný zásah každému vozidlu.

Avšak na druhej strane centralizovaný systém nie je úplne nevyužiteľný. Mohol by sa aplikovať napr. vo firmách, ktoré rozvážajú tovar viacerými nákladnými autami, ktoré chodia v skupinách. Tu je známy počet aút a tieto môžu navzájom komunikovať.

Riadenie prebiehalo nasledovne:

Po spustení riadenia sa začal pohybovať líder. Zmerali sa rozostupy medzi všetkými autami, a na základe rýchlostí všetkých áut, sa vypočítali optimálne rýchlosti pre každé riadené auto. Vďaka tomu, že náš matematický model zahŕňa kolónu ako celok, výpočet akčných zásahov prebiehal naraz v jednom, síce zložitejšom výpočte.

2.2.1 Matematický model

Na vytvorenie matematického modelu pre centralizovaný systém si musíme zvoliť počet vozidiel. My sme si zvolili štyri vozidlá, teda jedného lídra a tri autá, ktoré ho budú nasledovať. Tak ako pre decentralizovaný systém, aj tu budú stavovými veličinami rýchlosť lídra a všetky rozostupy, a riadiacimi veličinami budú rýchlosti všetkých sledujúcich áut. Označme rýchlosť lídra v_0, rozostupy medzi autami d_1, d_2 a d_3 a rýchlosti sledujúcich áut v_1, v_2 a v_3. Spojitý model má tvar:

$$\dot{v_0} = 0 \tag{2.12}$$

$$\dot{d_1} = v_0 - v_1 \tag{2.13}$$

$$\dot{d_2} = v_1 - v_2 \tag{2.14}$$

$$\dot{d_3} = v_2 - v_3 \tag{2.15}$$

Rýchlosť lídra je v čase akčného zásahu konštantná, preto je derivácia (2.12) nulová. Maticový tvar spojitého modelu má tvar:

$$\begin{bmatrix} \dot{v_0} \\ \dot{d_1} \\ \dot{d_2} \\ \dot{d_3} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} v_0 \\ d_1 \\ d_2 \\ d_3 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 \\ -1 & 0 & 0 \\ 1 & -1 & 0 \\ 0 & 1 & -1 \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix} \tag{2.16}$$

Spojitý model má tvar:

$$\dot{x}(t) = Ax(t) + Bu(t) \tag{2.17}$$

a matice spojitého modelu pred diskretizáciou:

$$A = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad B = \begin{bmatrix} 0 & 0 & 0 \\ -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \end{bmatrix} \quad C = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad D = 0 \tag{2.18}$$

Spojitý model musíme diskretizovať:

$$v_0(t + \Delta t) = v_0(t) \tag{2.19}$$

lebo rýchlosť lídra je v čase akčného zásahu konštantná.

Ďalej pre prvé sledujúce vozidlo:

$$\frac{d_1(t+\Delta t)-d_1(t)}{\Delta t} \approx \dot{d_1} = v_0 - v_1 \tag{2.20}$$

Rovnicu (2.20) upravíme:

$$d_1(t+\Delta t) = d_1(t) + \Delta t(v_0 - v_1) \tag{2.21}$$

druhé sledujúce vozidlo:

$$\frac{d_2(t+\Delta t)-d_2(t)}{\Delta t} \approx \dot{d_2} = v_1 - v_2 \tag{2.22}$$

Rovnicu (2.22) upravíme:

$$d_2(t+\Delta t) = d_2(t) + \Delta t(v_1 - v_2) \tag{2.23}$$

a tretie sledujúce vozidlo:

$$\frac{d_3(t+\Delta t)-d_3(t)}{\Delta t} \approx \dot{d_3} = v_2 - v_3 \tag{2.24}$$

Rovnicu (2.24) upravíme:

$$d_3(t+\Delta t) = d_3(t) + \Delta t(v_2 - v_3) \tag{2.25}$$

Prepísaním rovníc (2.19), (2.21), (2.23) a (2.25) dostaneme maticový tvar diskretizovaného modelu:

$$\begin{bmatrix} v_0 \\ d_1 \\ d_2 \\ d_3 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ \Delta t & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} v_0 \\ d_1 \\ d_2 \\ d_3 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 \\ -\Delta t & 0 & 0 \\ \Delta t & -\Delta t & 0 \\ 0 & \Delta t & -\Delta t \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix} \tag{2.26}$$

kde ľavá strana rovnice (2.26) je v čase $t + \Delta t$ a pravá strana v čase t.

Náš stavový opis má tvar:

$$x(t+\Delta t) = Ax(t) + Bu(t) \tag{2.27}$$

Vektor u predstavuje vstupy, teda rýchlosti sledujúcich áut v_1, v_2 a v_3, a musí spĺňať podmienku $u_{min} \leq u \leq u_{max}$. Vektor stavov obsahuje štyri veličiny: v_0 - rýchlosť lídra a d_1, d_2 a d_3 - rozostupy medzi autami. Matice nášho stavového opisu modelu sú:

$$A = \begin{bmatrix} 1 & 0 & 0 & 0 \\ \Delta t & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad B = \begin{bmatrix} 0 & 0 & 0 \\ -\Delta t & 0 & 0 \\ \Delta t & -\Delta t & 0 \\ 0 & \Delta t & -\Delta t \end{bmatrix} \quad C = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad D = 0 \tag{2.28}$$

2.3 Účelová funkcia

Pre decentralizovaný aj centralizovaný systém je účelová funkcia rovnaká. Použili sme kritérium minimalizujúce kvadrát rozdielu medzi skutočným a žiadaným rozostupom medzi autami a kvadrát akčných zásahov:

$$\min \left[\sum_{k=0}^{N-1} \left(u_{t+k}^T Q_u u_{t+k} + (w_k - y_{t+k})^T Q_y (w_k - y_{t+k}) \right) \right] \tag{2.29}$$

kde w_k je referenčná vzdialenosť od auta pred nami, a y_{t+k} je skutočná vzdialenosť od auta pred nami, pričom platí:

$$y_{t+k} = C x_{t+k} \tag{2.30}$$

2.4 Obmedzenia

Pre naše potreby budú obmedzeniami predikčná rovnica (2.10), resp. (2.27), počiatočná podmienka (1.3) a obmedzenia stavov a vstupov (1.4).

2.5 Simulácie v Matlabe

Vyššie definovaný problém sme riešili v Matlabe pomocou toolboxu Yalmip, ktorý umožňuje veľmi jednoduchý a intuitívny zápis optimalizačných úloh. Na riešenie týchto úloh sme použili spolu s Matlabom toolbox Gurobi.
Povieme si, ako sme vytvárali súbory, ktoré sme použili na riešenie nášho optimalizačného problému. Všetky tieto súbory sa nachádzajú na priloženom CD. Vytvorené súbory:

- *kolona_sim_zmena_rych.m* - Tu je definovaná perióda vzorkovania, počet áut, váhy pre vstupy a stavy a počiatočná podmienka. Podľa toho, ako sa nastaví premenná *simtype* (buď 'central' alebo 'necentral'), je spustený skript *central_mod.m* alebo *necentral_mod.m*. Nastaví sa počet simulačných krokov $Tsim$, zadá sa referenčná vzdialenosť a nastaví sa rýchlosť lídra, ktorá sa počas cyklu dvakrát zmení. Podľa premennej *simtype* sa v cykle, ktorý sa opakuje $Tsim$-krát, spúšťa buď funkcia *central_mpc.m* alebo *necentral_mpc.m*. Do matíc U, X a W sa po každom vykonaní cyklu zapíšu aktuálne hodnoty vstupov, stavov a referencií kvôli budúcemu vykresleniu. Keď skončí cyklus, vypočíta sa konečná hodnota účelovej funkcie a vykreslia sa grafy akčných zásahov a stavov funkciou *vykreslenie_c.m* alebo *vykreslenie_n.m*, v závislosti od premennej *simtype*.

- *kolona_sim_zmena_ref.m* - Jediná zmena oproti skriptu *kolona_sim_zmena_rych.m* je tá, že rýchlosť lídra je konštantná, a počas cyklu sa dvakrát zmení referencia pre rozostup.

- *necentral_mod.m* - Tu sa vytvoria matice spojitého modelu A, B, C a D (2.5) pre decentralizovaný systém. Tie sa diskretizujú príkazom $c2d()$. Nastaví sa tu predikčný horizont, definujú sa ohraničenia stavov, vstupov aj výstupov, nastaví sa maximálna zmena akčného zásahu oproti predchádzajúcemu. Nakoniec sa zavolá funkcia *opt_mpc_soft_delta.m*, ktorá vytvorí optimizer potrebný pre funkciu *necentral_mpc.m*.

- *central_mod.m* - Tu sa pomocou premennej p (počet áut) vytvoria matice spojitého modelu A, B, C a D (2.18) pre centralizovaný systém. Tie sa diskretizujú príkazom $c2d()$. Nastaví sa tu predikčný horizont, definujú sa ohraničenia stavov, vstupov aj výstupov, nastaví sa maximálna zmena akčného zásahu oproti predchádzajúcemu. Nakoniec sa zavolá funkcia *opt_mpc_soft_delta.m*, ktorá vytvorí optimizer potrebný pre funkciu *central_mpc.m*.

- *opt_mpc_soft_delta.m* - Tu sa definuje optimalizačný problém, teda účelová funkcia. Najprv sa zapíšu optimalizované premenné vstupov, stavov, výstupov a referencie, a tiež premennej sx, ktorá je použitá pri mäkkých ohraničeniach stavov. Všetky optimalizované premenné sú v Yalmipe typu *sdpvar*, a musí sa nastaviť ich rozmer (podľa modelu a predikčného horizontu). Nasledujúcim cyklom (iba časť funkcie) sa potom podľa predikčného horizontu vypočíta postupnosť akčných zásahov:

```
for k = (0:N-1)+1
    obj = obj + u(:, k)'*Qu*u(:, k) + ...
        (w - y(:, k))'*Qy*(w - y(:, k));
    obj = obj + ...
        sx(:, k)'*Qs*sx(:, k);
    con = con + [ x(:, k+1) == A*x(:, k) + B*u(:, k) ];
    con = [ con; ...
        y(:, k) == C*x(:, k) + D*u(:, k) ];
    con = con + [-sx(:, k) + xmin <= x(:, k) <= xmax + sx(:, k); ...
        umin <= u(:, k) <= umax; ...
        ymin <= y(:, k) <= ymax];
    con = con + [ sx(:, k) >= 0 ];
    if k >= 2
```

```
            deltau = u(:, k) - u(:, k-1);
        else
            deltau = u(:, 1) - uprev;
        end
        obj = obj + deltau'*Qdu*deltau;
        con = con + [ dumin <= deltau <= dumax ];
    end
```

Premenná *obj* predsatvuje účelovú funkciu a premenná *con* predstavuje ohraničenia. Na konci funkcia vráti *optimizer*, ktorý je potrebný pre funkcie *necentral_mpc.m* a *central_mpc.m*. Je to vlastne preložený zápis z Yalmipu tak, aby s ním bol Matlab schopný pracovať.

- *necentral_mpc.m* - Táto funkcia berie ako parameter *optimizer*, stav všetkých áut, referenciu, predchádzajúci akčný zásah, počet áut a matice diskretizovaného modelu A a B. Vráti nám optimálny akčný zásah pre aktuálny stav a budúci stav po vykonaní akčného zásahu. Keďže je to decentralizovaný systém, tak najprv sa vypočíta akčný zásah pre prvé sledujúce auto, vykoná sa a až potom sa vypočíta akčný zásah pre ďalšie, atď.

- *central_mpc.m* - Táto funkcia berie ako parameter *optimizer*, stav všetkých áut, referenciu, predchádzajúci akčný zásah a matice diskretizovaného modelu A a B. Vráti nám optimálny akčný zásah pre aktuálny stav a budúci stav po vykonaní akčného zásahu.

- *hodnota_obj.m* - Tu sa po skončení cyklu vypočíta konečná hodnota účelovej funkcie pre účely porovnania.

- *vykreslenie_n.m* - Funkcia na vykreslenie grafov vstupov a stavov pre decentralizovaný systém.

- *vykreslenie_c.m* - Funkcia na vykreslenie grafov vstupov a výstupov pre centralizovaný systém.

2.6 Výsledky simulácií

Simulovali sme riadenie pre štyri vozidlá, teda jedného lídra a tri sledovacie autá. Ukážeme si viaceré prípady simulácií s rôzne nastavenými parametrami. Ohraničenia boli rovnaké, maximálna rýchlosť bola $20ms^{-1}$, minimálna rýchlosť bola $0ms^{-1}$, maximálny rozostup bol

$100m$ a minimálny rozostup bol $1m$. Rýchlosť sa mohla zmeniť maximálne o $5ms^{-1}$ oproti predchádzajúcej rýchlosti.

2.6.1 Prípad meniacej sa rýchlosti

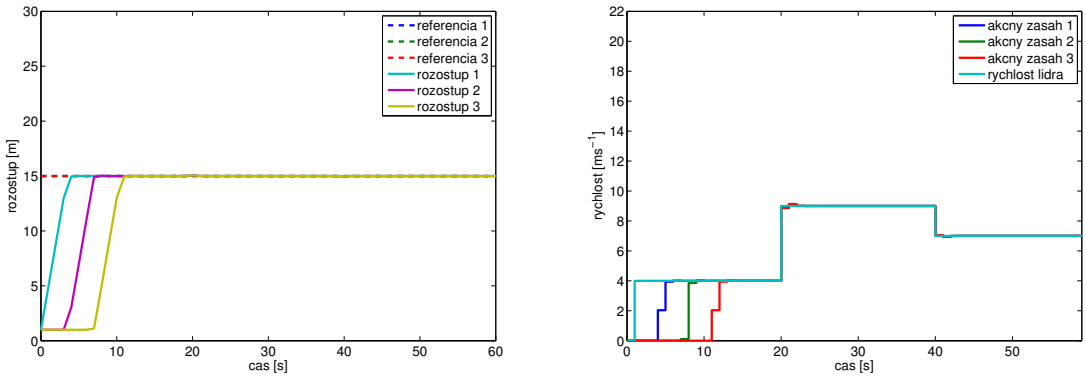
Prvý prípad bude reprezentovať dva scenáre, kedy sa rýchlosť lídra mení nasledovne:

- v prvej tretine simulácie bola $4ms^{-1}$
- v druhej tretine simulácie bola $9ms^{-1}$
- v tretej tretine simulácie bola $7ms^{-1}$

Referencia bola konštantná pre všetky autá $15m$. Váhy pre stavy mali nastavenú hodnotu 100 a váhy pre vstupy hodnotu 1. Na obrázkoch 2.1 - 2.4 sú graficky znázornené priebehy riadenia skupiny vozidiel pre rôzne situácie, ktoré môžu nastať.

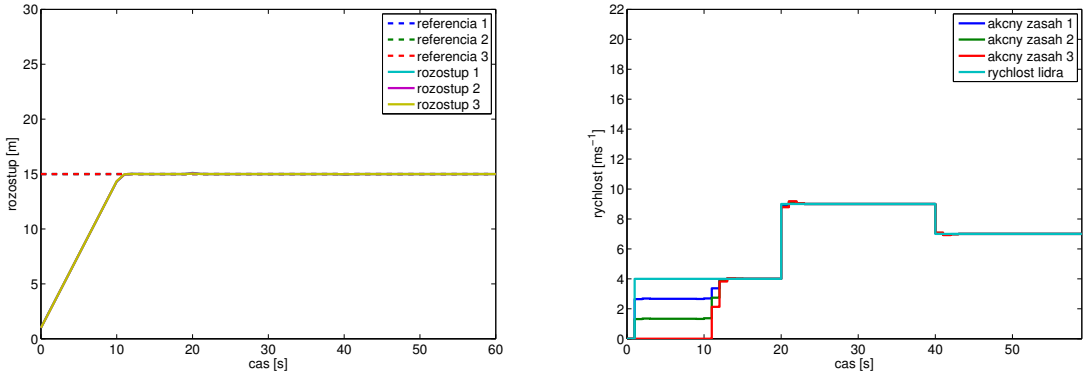
Pohýňanie sa na križovatke

V prvom scenári sme uvažovali počiatočný stav taký, že rýchlosť lídra bola nulová, rýchlosti sledovacích áut boli tiež nulové a rozostupy medzi všetkými autami boli $1m$. Toto by mohlo zodpovedať reálnej situácii, kedy autá stoja na križovatke, a začnú sa pohybovať. Cieľom je, aby sa všetky autá pohli v čo možnom najkratšom čase tak, aby medzi nimi nastal žiadaný rozostup. Sledovacie autá musia v tomto prípade meniť rýchlosť postupne, každému ďalšiemu autu je nastavená nižšia rýchlosť ako autu pred ním, aby sa vytvoril žiadaný rozostup, čo môžme pozorovať na grafoch akčných zásahov (grafy vpravo na obrázkoch 2.1 a 2.2).
Je zrejmé, že pre túto úlohu riadenia by bol vhodnejší centralizovaný systém, pre ktorý je priebeh riadenia znázornený na obrázku 2.2. Vidíme, že oproti decentralizovanému riadeniu - obrázok 2.1, tu nevznikajú tak výrazné oneskorenia, autá sú riadené súčasne a rozostupy medzi nimi sú rovnaké pre konkrétnu periódu vzorkovania, čo sa nedá povedať o decentralizovanom systéme. Lepší priebeh riadenia pre centralizovaný systém potvrdzuje aj hodnota účelovej funkcie, ktorá je pre centralizovaný systém nižšia.



Obr. 2.1: Riadenie a akčné zásahy pre decentralizovaný systém

Hodnota účelovej funkcie: $obj = 313270$



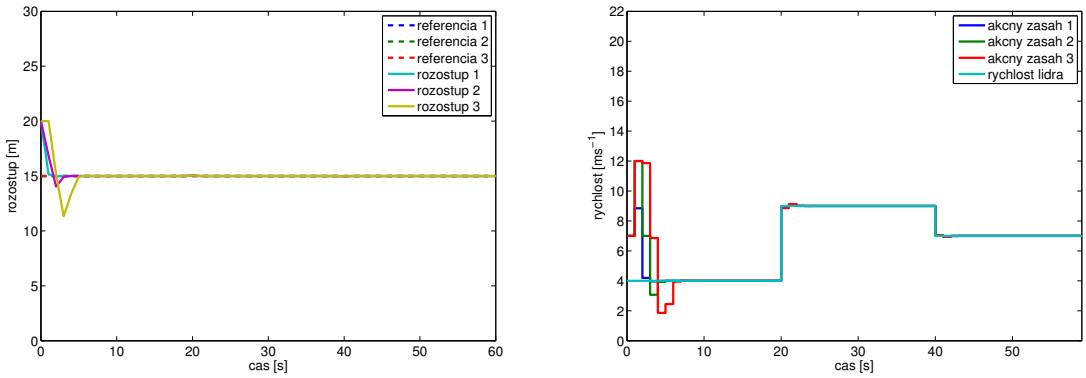
Obr. 2.2: Riadenie a akčné zásahy pre centralizovaný systém

Hodnota účelovej funkcie: $obj = 244430$

Kolóna

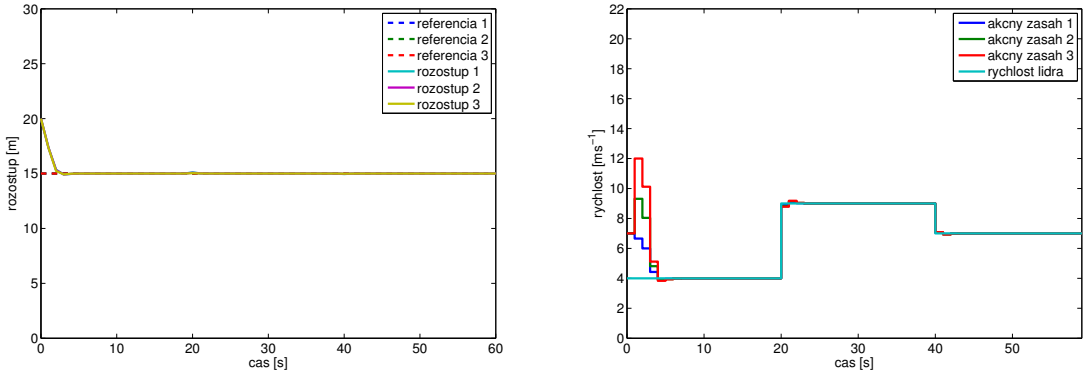
V druhom scenári sme sa snažili simulovať situáciu, kedy sú autá v pohybe, no nemajú medzi sebou žiadaný rozostup. Toto môže reprezentovať autá v kolóne, pričom líder sa na začiatku pohybuje rýchlosťou $4ms^{-1}$ a sledovacie autá majú počiatočnú rýchlosť $7ms^{-1}$. Rozostupy medzi autami na začiatku boli $20m$. To znamená, že sledujúce autá musia ešte zrýchliť, aby rýchlejšie zmenšili rozostupy medzi sebou. V tomto prípade je pre autá na začiatku vypočítaná vyššia rýchlosť, aby sa rozostupy dostali na žiadanú hodnotu. Toto je vidieť aj na grafoch akčných zásahov (grafy vpravo na obrázkoch 2.3 a 2.4)

Pri centralizovanom systéme (obrázok 2.4) nevznikajú prekmity, ktoré sú zrejmé pri decentralizovanom systéme (obrázok 2.3). Lepšie výsledky riadenia pre centralizovaný systém potvrdzuje aj hodnota účelovej funkcie, ktorá je nižšia ako pre decentralizovaný systém.



Obr. 2.3: Riadenie a akčné zásahy pre decentralizovaný systém

Hodnota účelovej funkcie: $obj = 21369$



Obr. 2.4: Riadenie a akčné zásahy pre centralizovaný systém

Hodnota účelovej funkcie: $obj = 18423$

2.6.2 Prípad meniacej sa referencie vzdialenosti

Druhý prípad bude reprezentovať dva scenáre, kedy sa referencia vzdialenosti pre všetky autá mení nasledovne:

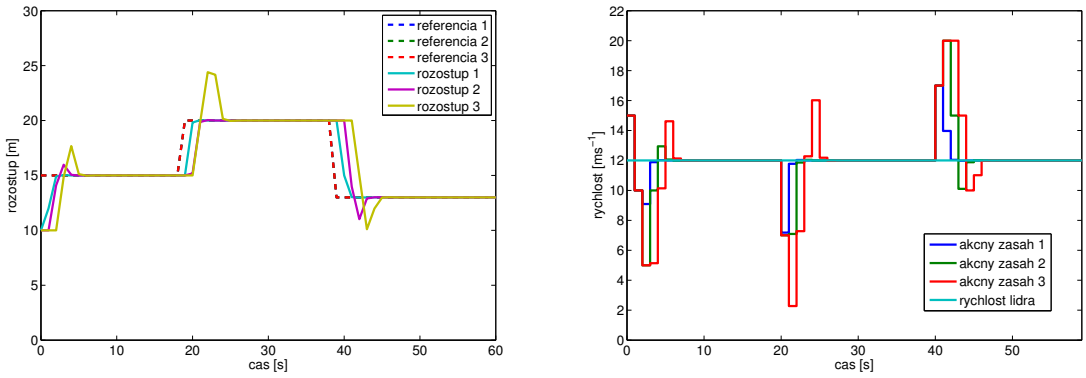
- v prvej tretine simulácie bola $15m$
- v druhej tretine simulácie bola $20m$
- v tretej tretine simulácie bola $13m$

Rýchlosť lídra bola konštantná $12ms^{-1}$. Váhy pre stavy mali nastavenú hodnotu 100 a váhy pre vstupy hodnotu 1. Na obrázkoch 2.5 - 2.8 sú graficky znázornené priebehy riadenia pre rôzne sitúacie, ktoré môžu nastať.

Kolóna - počiatočné znižovanie rýchlosti

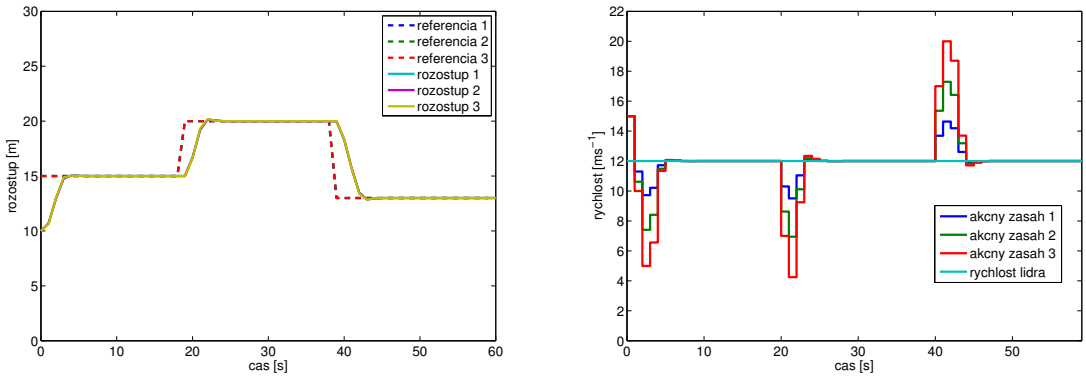
V prvom scenári sme ako počiatočný stav uvažovali rýhchlosť lídra $12ms^{-1}$, rozostupy medzi autami boli $10m$ a rýchlosti sledujúcich aút boli $15ms^{-1}$. Táto situácia by mohla zodpovedať kolóne, kedy chceme, aby autá išli konštantnou rýchlosťou, no rozostupy medzi nimi by sa menili podľa zadanej referencie. Sledovacie autá by museli v tomto scenári najprv znížiť svoju rýchlosť, aby dosiahli požadovaný rozostup. Následne by upravovali rýchlosť podľa toho, či sa žiadaný rozostup od predchádzajúceho auta zväčší alebo zmenší. Je to vidieť aj na grafoch akčných zásahov (pravé grafy na obrázkoch 2.5 a 2.6)

Môžme povedať, že pri centralizovanom systéme nevznikajú takmer žiadne prekmity, čo sa nedá povedať o decentralizovanom systéme, kedy sú prekmity viditeľné. Graf centralizovaného systému takisto potvrdzuje, že autá sú riadené súčasne, pri decentralizovanom je každé auto riadené zvlášť. Lepšie výsledky riadenia pre centralizovaný systém potvrdzuje aj hodnota účelovej funkcie, ktorá je nižšia ako pre decentralizovaný sytém.



Obr. 2.5: Riadenie a akčné zásahy pre decentralizovaný systém

Hodnota účelovej funkcie: $obj = 90871$

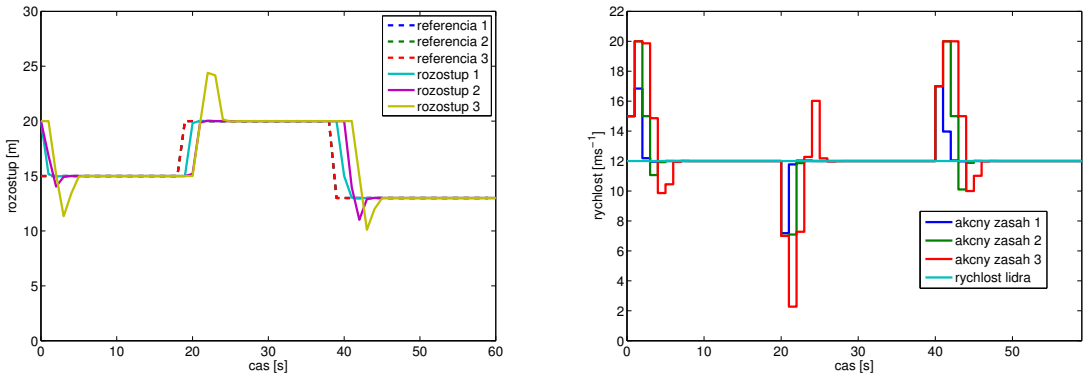


Obr. 2.6: Riadenie a akčné zásahy pre centralizovaný systém

Hodnota účelovej funkcie: $obj = 76994$

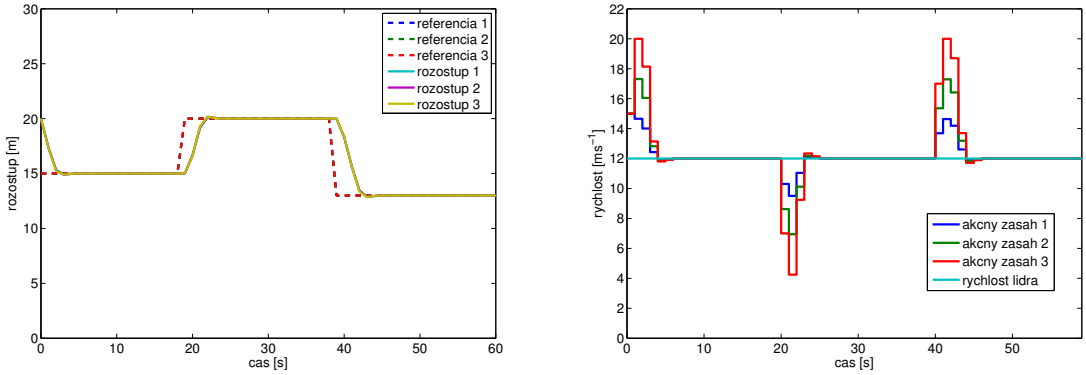
Kolóna - počiatočné zvyšovanie rýchlosti

V druhom scenári sme uvažovali situáciu v kolóne, kedy chceme meniť rozostupy medzi autami. Počiatočná rýchlosť lídra bola $12ms^{-1}$, rozostupy medzi autami boli na začiatku $20m$ a rýchlosti sledujúcich áut boli $15ms^{-1}$. Z toho vyplýva, že sledujúce autá museli na začiatku zvýšiť svoju rýchlosť, aby dosiahli požadovaný rozostup od auta pred sebou. Následne bola rýchlosť upravovaná na základe meniacej sa hodnoty referencie na rozostup nahor alebo nadol, čo je viditeľné aj na grafoch akčných zásahov (pravé grafy na obrázkoch 2.7 a 2.8) Ďalej tu pozorujeme menej prudké zmeny rýchlosti pri centralizovanom systéme. Čo sa týka riadenia, pri centralizovanom systéme (obrázok 2.8) nie sú pozorované skoro žiadne prekmity a riadenie áut prebieha súčasne, pri decentralizovanom systéme (obrázok 2.7) sú prekmity dosť výrazné, a riadenie áut je individuálne, čo vidíme na oneskoreniach pri dosahovaní požadovaného rozostupu. Hodnota účelovej funkcie potvrdzuje lepší priebeh riadenia pre centralizovaný systém, keďže jej hodnota je nižšia ako pri decentralizovanom systéme.



Obr. 2.7: Riadenie a akčné zásahy pre decentralizovaný systém

Hodnota účelovej funkcie: $obj = 87605$



Obr. 2.8: Riadenie a akčné zásahy pre centralizovaný systém

Hodnota účelovej funkcie: $obj = 73281$

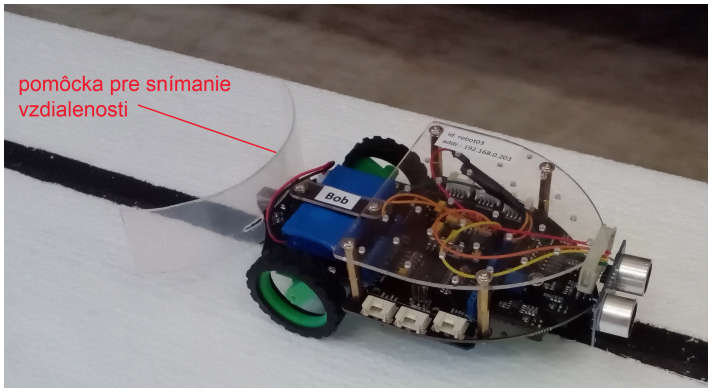
Zhrnutie pre výsledky simulácii

Na základe vyššie uvedených grafov a hodnôt účelovej funkcie môžme povedať, že centralizovaný systém poskytuje lepšie riadenie, čo bolo očakávané.

2.7 Reálne roboty

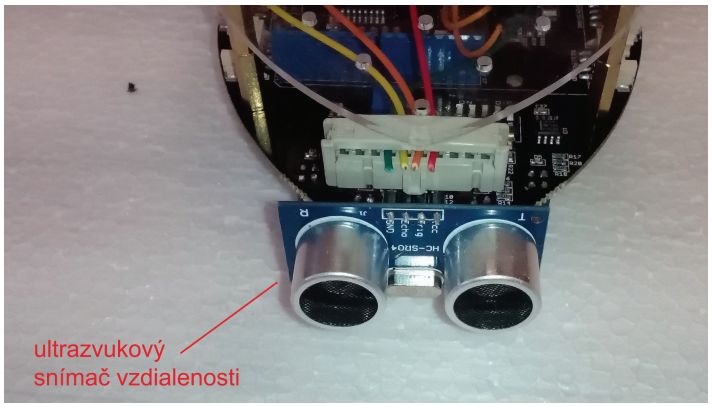
Na riadenie reálnych robotov sme používali takmer rovnaké skripty ako pri simuláciách, no museli sme vykonať isté zmeny, aby MPC používalo na výpočet akčných zásahov stavy robotov, a aby sme mohli posielať vypočítané akčné zásahy robotom. Skrátene povedané, museli sme vyriešiť komunikáciu medzi robotmi a Matlabom.

2.7.1 Konštrukcia robota



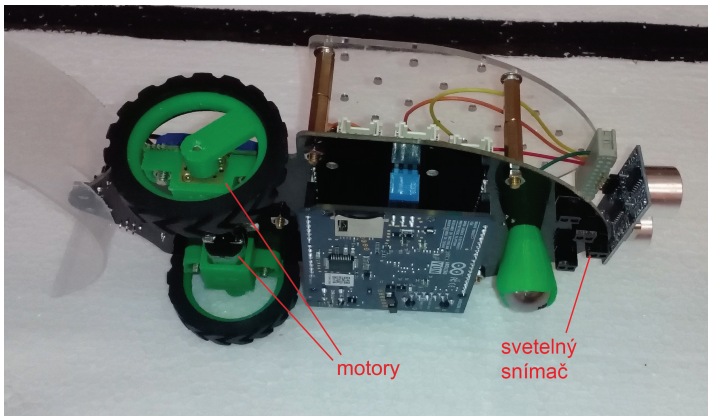
Obr. 2.9: Robot

Na obrázku 2.9 vidíme robota, ktorého sme používali pri našej práci. Pomôcka na snímanie vzdialenosti bola potrebná z toho dôvodu, že roboty nemali v zadnej časti súvislú plochu, preto sme vyrobili plastový 'štít', ktorý umožňoval dobré snímanie vzdialenosti sledujúcim robotom.



Obr. 2.10: Snímač vzdialenosti

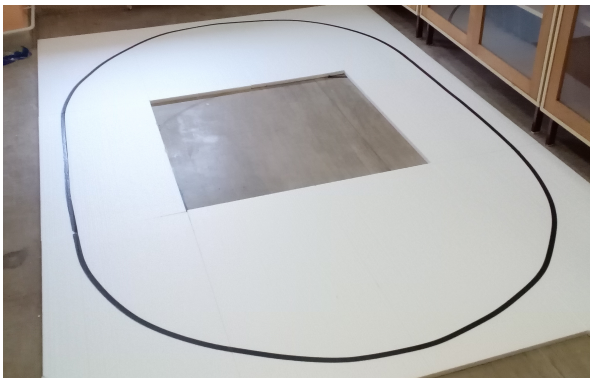
Na obrázku 2.10 vidíme ultrazvukový snímač vzdialenosti, ktorý mal vpredu pripojený každý robot. Tento snímač bol schopný registrovať vzdialenosť od $0cm$ do $100cm$.



Obr. 2.11: Motory a snímače svetelnosti

Na obrázku 2.11 je vidieť, že každé koleso má svoj vlastný motor. Robot zatáčal tým spôsobom, že algoritmus, ktorý riadil smer robota, posielal rôznu rýchlosť otáčania do každého kolesa tak, aby výsledná rýchlosť robota zodpovedala nastavenej rýchlosti podľa toho, akým smerom sa mal robot pohybovať (podľa čiernej čiary). Tieto motory brali ako akčný zásah bezrozmerné číslo od 0 do 127 (nie v $[ms^{-1}]$), čo predstavovalo maximálnu rýchlosť v bezrozmerových jednotkách.

Ďalej tu môžme vidieť svetelné senzory, ktoré reagovali na čiernu čiaru, nakreslenú na podlahe. Týchto senzorov bolo päť vedľa seba, a podľa toho, ktoré registrovali čiaru, robot vedel, ktorým smerom má ísť. Ak čiaru registroval jeden alebo tri stredné senzory, robot šiel rovno. Ak registrovali čiaru senzory na stranách, robot sa pohyboval na tú stranu, na ktorej boli registrujúce senzory.



Obr. 2.12: Dráha

Obrázok 2.12 reprezentuje dráhu, ktorú sme vyrobili z polystyrénu. Čierna čiara bola nakreslená tušom. Plocha, na ktorej bola dráha, mala rozmery $2m$ x $3m$.

2.7.2 Riadenie reálnych robotov

Pre potreby komunikácie medzi robotmi a PC boli vytvorené triedy *BugArray*, *WSClient* a *RoboBug*. Komunikácia s robotmi prebiehala na základe udalostí, teda ak robot dostal nejakú správu, hneď odoslal informáciu o svojom stave naspäť.
Trieda *BugArray* obsahuje nasledovné metódy:

- *BugArray* - Vytvorí pole pre každého robota podľa zadaného 'ID'.
- *connect* - Pripojí všetkých robotov na základe vytvorených polí alebo konkrétneho zadaného robota.
- *close* - Odpojí všetkých robotov alebo zadaného robota.
- *setSpeed* - Nastaví rýchlosť všetkým robotom alebo konkrétnemu zadanému robotovi.
- *getState* - Zistí stav všetkých robotov alebo konkrétneho zadaného robota v tvare [*vzdialenosť od prekážky, rýchlosť robota predo mnou*].
- *stop* - Zastaví všetkých robotov alebo konkrétneho zadaného robota.
- *getSpeed* - Zistí rýchlosti robotov alebo jedného robota.
- *getDistance* - Zistí vzdialenosť od prekážky pre všetkých robotov alebo jedného robota.

Vytvorené boli súbory:

- *kolona_sim.m* - funguje podobne ako pri simulácii, s niektorými zmenami:

  ```
  bugs = BugArray(roboty);
  ```

  Takto sme vytvorili prístup ku každému robotovi. Vektor *roboty* obsahuje číslo každého robota, ktorého chceme pripojiť.

  ```
  bugs.connect();
  ```

  Takto sme robotov pripojili.

  ```
  pociatok = zeros(length(bugs.Bugs), 1);
  bugs.setSpeed(pociatok);
  pause(1);
  bugs.Bugs(1).setSpeed(50);
  ```

Aby sme získali počiatočný stav robotov, museli sme im poslať nejakú správu, v našom prípade to bola nulová rýchlosť. Následne sme prvému robotovi nastavili hodnotu rýchlosti na 50, aby sa začal pohybovať. Podľa premennej *simtype* ('necentral' alebo 'central') sa v cykle spúšťa funkcia *riadenie_nec_roz.m* alebo *riadenie_cen_roz.m*.

- *riadenie_nec_roz.m* - Tu sa pomocou metódy *getState* načíta a uloží stav celej kolóny. Z toho sa potom pre každé sledujúce auto v kolóne vytvorý stavový vektor pre decentralizovaný systém a postupne sa pre každé auto výpočíta optimálny akčný zásah a odošle sa konkrétnemu robotovi.
- *riadenie_cen_roz.m* - Tu sa pomocou metódy *getState* načíta a uloží stav celej kolóny. Z toho sa potom vytvorí stavový vektor pre celú kolónu pre centralizovaný systém. Vypočítajú sa optimálne akčné zásahy pre celú kolónu a odošlú sa naraz všetkým autám.
- *necentral_mod.m* - Bez zmeny oproti simulácii.
- *central_mod.m* - Bez zmeny oproti simulácii.
- *opt_mpc_soft_delta.m* - Bez zmeny oproti simulácii.
- *hodnota_obj.m* - Bez zmeny oproti simulácii.
- *vykreslenie_n.m* - Bez zmeny oproti simulácii.
- *vykreslenie_c.m* - Bez zmeny oproti simulácii.

2.7.3 Výsledky pre riadenie reálnych robotov

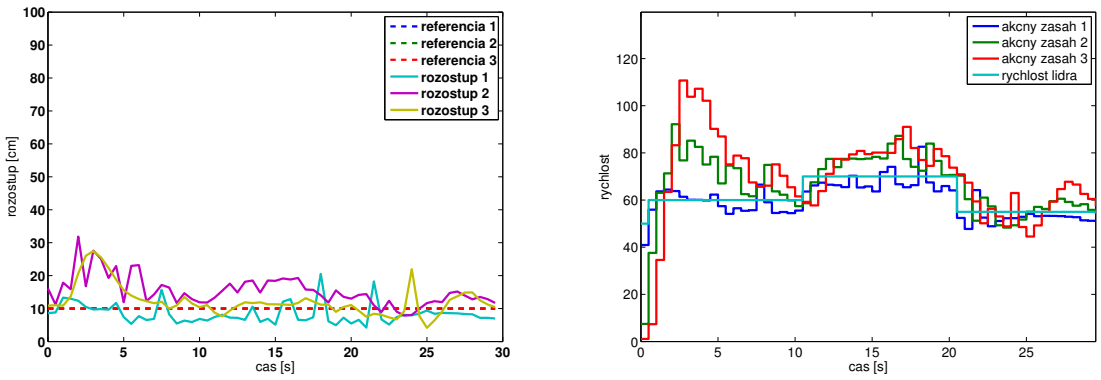
Pri riadení reálnych robotov sme používali štyri vozidlá tak, ako to bolo v simuláciách. Demonštrovali sme štyri scenáre, pričom tieto sa líšili zmenenou referenciou, pri každom scenári sa žiadaný rozostup medzi robotmi zvyšil o $5cm$. Týmto sme chceli zistiť, ako sa bude meniť kvalita riadenia pri zväčšovaní referencie na rozostup. Rýchlosť lídra sa v každom scenári menila nasledovne:

- v prvej tretine simulácie bola 60
- v druhej tretine simulácie bola 70
- v tretej tretine simulácie bola 55

Ohraničenia boli rovnaké, teda maximálna rýchlosť bola 127, minimálna rýchlosť bola 0, maximálny rozostup bol $100cm$, minimálny rozostup bol $3cm$. Perióda vzorkovania bola $0,5s$. Počiatočná rýchlosť lídra bola vždy 50 a počiatočná rýchlosť sledovacích áut bola vždy 0. Váhy pre stavy mali nastavenú hodnotu 100 a váhy pre vstupy hodnotu 1.

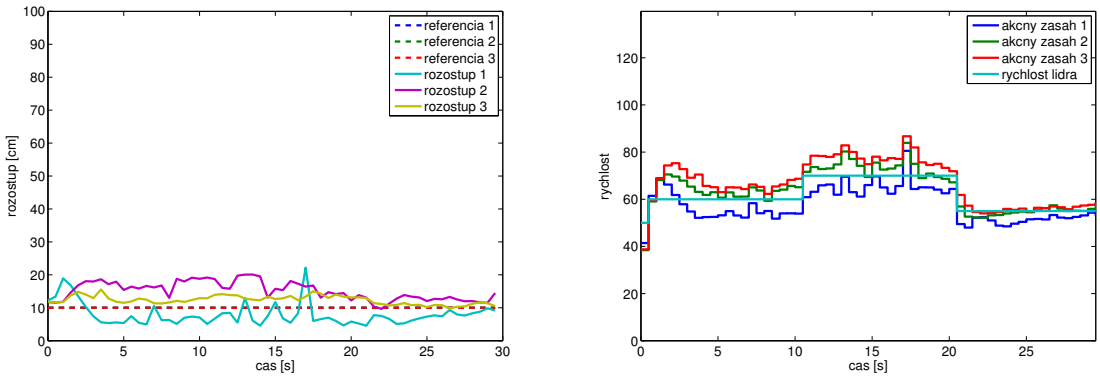
Najmenšia referenčná vzdialenosť - 10cm

Referencia $10cm$ zodpovedala približne polovici dĺžky vozidla. Na obrázkoch 2.13 a 2.14 vidíme grafické znázornenie riadenia pre decentralizovaný, resp. centralizovaný systém, a tiež zodpovedajúce zmeny rýchlosti pre každé auto. Počiatočné rozostupy medzi autami boli pre decentralizovaný systém $d_1 = 8,63cm$, $d_2 = 16,08cm$, $d_3 = 10,89cm$ a pre centralizovaný systém $d_1 = 12,23cm$, $d_2 = 11,72cm$, $d_3 = 11,86cm$. Toto bolo dané tým, ako ďaleko od seba sme rozmiestnili robotov. Žiadané rozostupy boli nastavené na $10cm$. Je zrejmé, že lepšie riadenie bolo dosiahnuté pri centralizovanom systéme, kedy krivky pre všetky autá sú menej kmitavé a bližšie k referencii. Čo sa týka akčných zásahov, vidíme, že pri centralizovanom systéme boli zmeny rýchlosti viditeľne plynulejšie a rýchlosti jednotlivých riadených áut sa viac približovali k rýchlosti lídra. V prospech centralizovaného systému hovorí aj hodnota účelovej funkcie, ktorá je menšia.



Obr. 2.13: Riadenie a akčné zásahy pre decentralizovaný systém

Hodnota účelovej funkcie: $obj = 1320300$

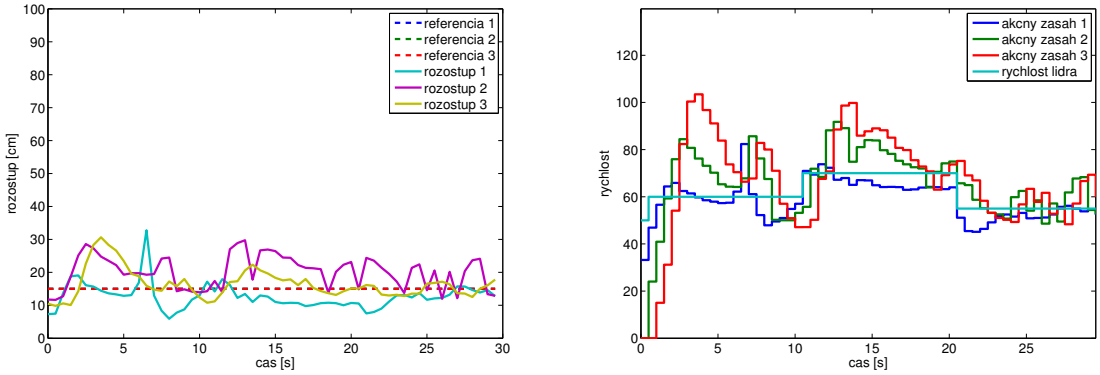


Obr. 2.14: Riadenie a akčné zásahy pre centralizovaný systém

Hodnota účelovej funkcie: $obj = 1066600$

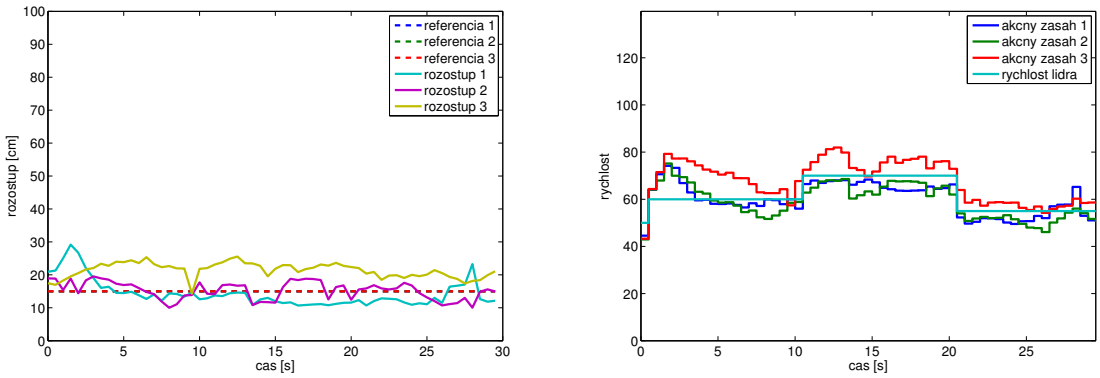
Referenčná vzdialenost - 15cm

Referencia $15cm$ predstavovala asi tri štvrtiny dĺžky auta. Počiatočné rozostupy pre decentralizovaný systém boli $d_1 = 7,32cm$, $d_2 = 11,75cm$, $d_3 = 10,48cm$ a pre centralizovaný systém $d_1 = 20,93cm$, $d_2 = 18,93cm$, $d_3 = 17,46cm$. Na obrázkoch 2.15 a 2.16 vidíme grafické znázornenie riadenia pre decentralizovaný a centralizovaný systém, a tiež zodpovedajúce akčné zásahy pre každé auto. Žiadané rozostupy boli nastavené na $15cm$. Tu sa nám znova potvrdzuje lepšie riadenie pre centralizovaný systém, no vidíme, že všeobecne sa zväčšením žiadaného rozostupu riadenie zhoršilo. Akčné zásahy sú opäť plynulejšie pri centralizovanom systéme. Hodnoty účelovej funkcie sa všeobecne zvýšili, no stále platí, že pre centralizovaný systém je táto hodnota menšia.



Obr. 2.15: Riadenie a akčné zásahy pre decentralizovaný systém

Hodnota účelovej funkcie: $obj = 1325300$

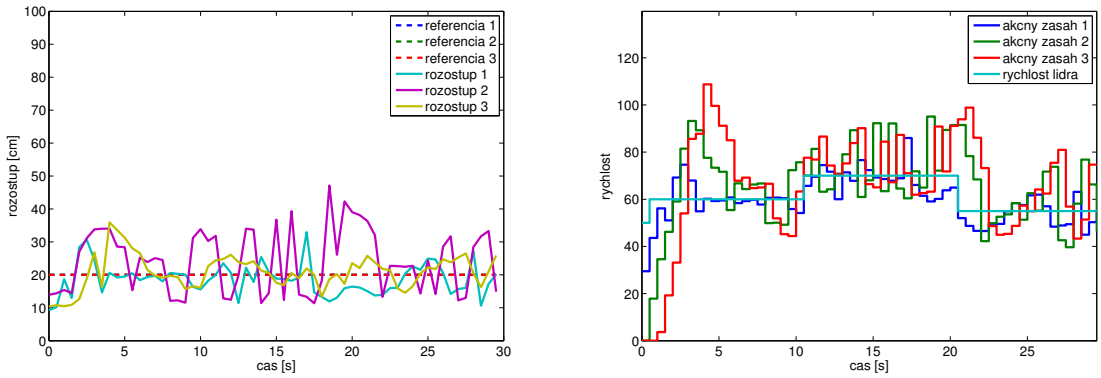


Obr. 2.16: Riadenie a akčné zásahy pre centralizovaný systém

Hodnota účelovej funkcie: $obj = 1129800$

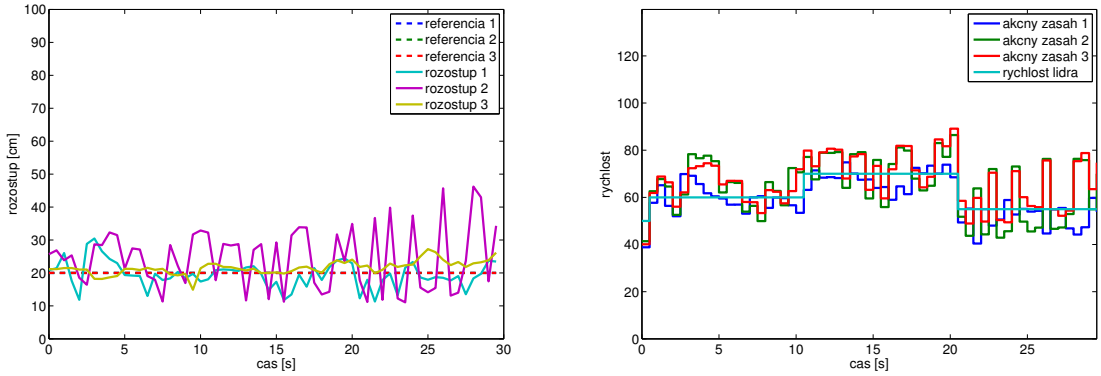
Referenčná vzdialenosť - 20cm

Referencia $20cm$ predstavovala približne dĺžku auta. Počiatočné rozostupy pre decentralizovaný systém boli $d_1 = 9,28cm$, $d_2 = 13,95cm$, $d_3 = 10,38cm$ a pre centralizovaný systém $d_1 = 20,89cm$, $d_2 = 25,70cm$, $d_3 = 21,17cm$. Na obrázkoch 2.17 a 2.18 vidíme grafické znázornenie riadenia pre decentralizovaný a centralizovaný systém, a tiež zodpovedajúce akčné zásahy pre každé auto, kedy žiadané rozostupy boli nastavené na $20cm$. Môžme si všimnúť, že zvyšovaním žiadaného rozostupu medzi autami sa riadenie zhoršuje. Tu sa viditeľne zhoršilo riadenie pre druhé riadené auto, a to pri oboch systémoch. To bolo ale asi spôsobené nie iba zväčšeným rozostupom, ale skôr poruchou snímača vzdialenosti, ktorý často generoval príliš vysoké skokové hodnoty na to, aby boli reálne v danom prípade. Kvôli tomu boli generované veľké akčné zásahy a riadenie pre toto auto bolo menej kvalitné. No stále platí, že centralizovaný systém na tom bol lepšie, ako z hľadiska plynulosti zmien rýchlosti, tak z hľadiska hodnoty účelovej funkcie.



Obr. 2.17: Riadenie a akčné zásahy pre decentralizovaný systém

Hodnota účelovej funkcie: $obj = 1763400$



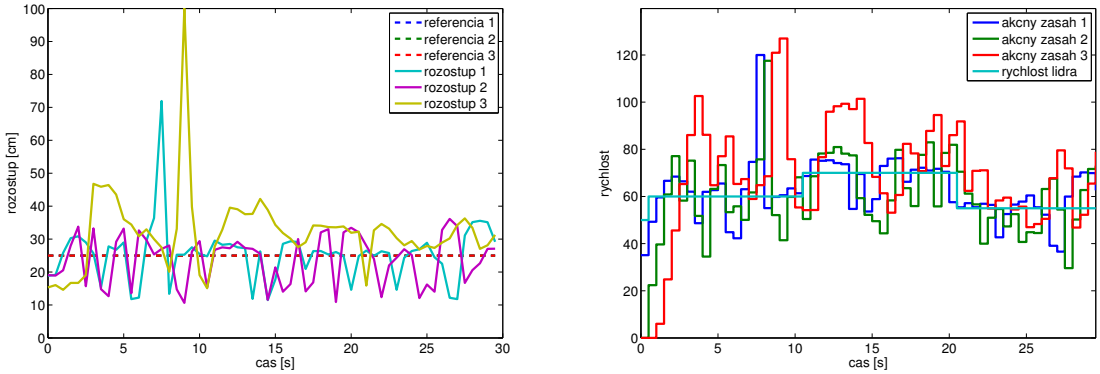
Obr. 2.18: Riadenie a akčné zásahy pre centralizovaný systém

Hodnota účelovej funkcie: $obj = 1531100$

Najväčšia referenčná vzdialenosť - 25cm

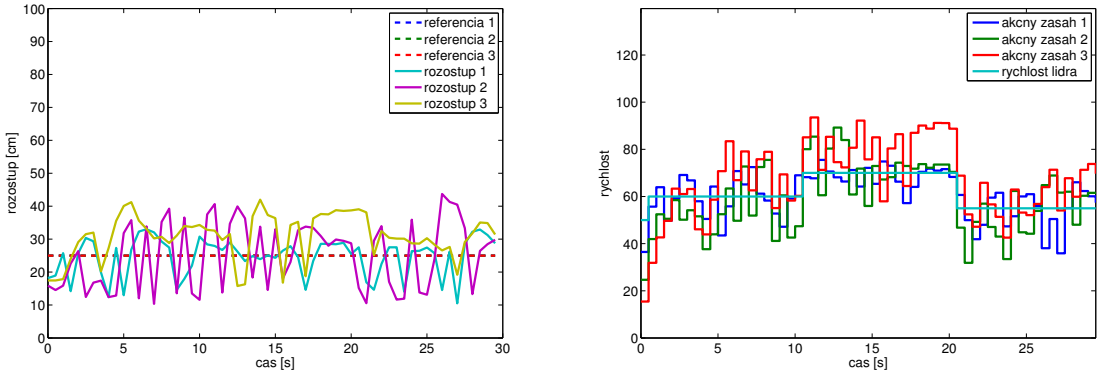
Referencia $25cm$ predstavovala o niečo vačšiu hodnotu ako bola dĺžka auta. Počiatočné rozostupy boli pre decentralizovaný systém $d_1 = 18,87cm$, $d_2 = 19,07cm$, $d_3 = 15,33cm$ a pre centralizovaný systém $d_1 = 18,18cm$, $d_2 = 15,88cm$, $d_3 = 17,46cm$. Na obrázkoch 2.19 a 2.20 vidíme grafické znázornenie riadenia pre decentralizovaný a centralizovaný systém, a tiež zodpovedajúce akčné zásahy pre každé auto, kedy žiadané rozostupy boli nastavené na $25cm$. Pri takto veľkom rozostupe sme na našej dráhe mali niekedy problém dosiahnuť uspokojivé riadenie, lebo sa stávalo, že dve za sebou idúce autá sa vychýlili zo spoločnej orientácie na toľko, že snímač vzdialenosti sledovacieho auta nezachytil auto pred sebou a nameral maximálnu, resp. oveľa väčšiu vzdialenosť, ako bola reálna. To môžeme vidieť pri prvom a treťom aute decentralizovaného systému. V takejto situácii riadiaci systém vygeneroval veľmi

vysoký akčný zásah, čo nepriaznivo pôsobilo na celé riadenie, lebo autá sa mohli takýmto štýlom dostať k sebe veľmi blízko, kým sa stihla zaregistrovať naozajstná vzdialenosť. Toto spôsobilo oveľa väčší rozdiel v hodnotách účelovej funkcie, ako v predošlých pripadoch. Ale aj pri tomto poslednom prípade platí, že centralizovaný systém poskytuje kvalitnejšie riadenie ako decentralizovaný, a to podľa plynulosti akčných zásahov aj podľa hodnôt účelovej funkcie.



Obr. 2.19: Riadenie a akčné zásahy pre decentralizovaný systém

Hodnota účelovej funkcie: $obj = 2717200$



Obr. 2.20: Riadenie a akčné zásahy pre centralizovaný systém

Hodnota účelovej funkcie: $obj = 2039200$

Zhrnutie pre výsledky reálneho riadenia

Na základe vyššie uvedených grafov a hodnôt účelovej funkcie môžeme povedať, že riadením reálnych robotov sme potvrdili hypotézu z úvodu a aj výsledky simulácií, teda že centralizovaný systém poskytuje lepšie riadenie ako decentralizovaný.

Záver

Navrhli sme decentralizováný aj centralizovaný riadiaci systém na základe MPC, ktoré sme overili na simuláciách a aplikovali sme ich aj na reálnych robotov. Na to, aby sa nám to podarilo, sme museli najprv vytvoriť matematický model tak pre decentralizovaný, ako aj pre centralizovaný systém.

Porovnali sme tieto dva systémy na základe vypočítaných hodnôt účelovej funkcie. Môžme povedať, že tak, ako sme predpokladali v úvode, centralizovaný systém poskytol lepšie riadenie ako decentralizovaný.

Pri riadení reálnych robotov sa vyskytovali rôzne poruchy. Najväčším problémom bolo to, že každý robot bol jedinečný, ich reálna rýchlosť závisela od kapacity batérie, ktorú sme ale nevedeli priamo merať. To znamená, že ak mali nastavenú rýchlosť všetky roboty rovnakú, tak reálne nešli rovnako rýchlo. Tým bolo ovplyvnené aj riadenie, keďže výpočet optimálnej rýchlosti závisel na rýchlosti predchádzajúceho auta. Tak sa napr. mohlo stať, že auto, ktoré malo vzdialenosť od auta pred sebou nižšiu, ako bola referencia, sa ešte stále približovalo napriek tomu, že mu bola posielaná nižšia rýchlosť ako autu pred ním.

V budúcnosti by bolo zaujímavé riešiť problém riadenia skupiny vozidiel tak, že riadiacou veličinou by bolo zrýchlenie, a tiež že by sme riadili na základe nameranej reálnej rýchlosti. To by malo odstrániť vyššie opisovanú poruchu jedinečnosti robotov.

Odhliadnuc od týchto porúch môžeme zhodnotiť, že sme úspešne riadili reálnych robotov nami navrhnutým systémom riadenia na základe MPC.

Literatúra

[1] Wikipedia - The Free Encyklopedia: *Model predictive control*[online]. Posledná úprva 2014-05-14. [citované 2014-05-16]. Dostupné na internete: `http://en.wikipedia.org/wiki/Model_predictive_control`

[2] KARAS, Adrián, ROHAĽ-ILKIV, Boris, BELAVÝ, Cyril: *Praktické aspekty prediktívneho riadenia.* Bratislava: Vyd. STU. 2007. ISBN 978-80-89316-06-9

[3] KVASNICA, Michal: *Model Predictive Control (MPC).* Lecture Part 1: Introduction

[4] KUZNETSOV, A. G.: *Constrained predictive control: a brief survey.* Journal A 37(2), 3-8

[5] SCOKAERT, P. O. M.: *Constrained predictive control.* Technical Report OUEL 2023/94. Department of Engineering Science, Oxford University. Parks Road, Oxford, Great Britain.

[6] `http://commons.wikimedia.org/wiki/File:Convex_Function.png`

[7] `https://www.kickstarter.com/projects/rjuniyer/jeane-the-end-of-traffic-jams`