České vysoké učení technické v Praze

Fakulta elektrotechnická
Katedra řídící techniky

DIPLOMOVÁ PRÁCE

Couvání auta s přívěsem

Praha, 2010 Bc. Bohumír Baier

Prohlášení

Prohlašuji, že jsem svou diplomovou práci vypracoval samostatně a použil jsem pouze podklady (literaturu, projekty, SW atd.) uvedené v přiloženém seznamu.

V Praze, dne 14.5.2010 …….

podpis

Poděkování:

Děkuji vedoucímu diplomové práce Ing. Františku Vaňkovi za pomoc, cenné rady a velkou trpělivost při tvorbě této práce.

Abstrakt

Tato práce se zabývá popisem a řízením auta s přívěsem. V práci se provádí návrh trajektorie mezi překážkami pomocí algoritmu dynamického programování. Pro automatické couvání byly navrhnuty regulátory typu, PID, LQ a MPC. Grafické zobrazení couvání vozíku se vykresluje pomocí virtual reality toolboxu v matlabu. Uživatelské rozhraní umožňuje srovnání ručního a automatického řízení.

Abstract

This work focuses on description and driving of a car with a trailer. The trajectory through the obstacles is evaluated using the dynamic programming algorithm. For the automatic reverse there are PID, LQ and MPC regulators. The graphic description of a trailer reverse is displayed with the matlab virtual reality toolbox. The user interface allows the comparison of manual and automatic driving.

Obsah

1.Úvod 1
2. Matematický model 2
2.1 Auto 2
2.2 Popis přívěsu 4
2.3 Další přívěs 6
2.4 Zjednodušení rovnic 8
2.5 Otáčení čtyř kol 9
3. Návrh referenční trajektorie 10
3.1 Body pro trajektorii 11
3.2 Možnosti propojení 12
3.3 Optimální proložení bodů 13
4. Proložení referenční trajektorie 15
4.1 Popis překážek 15
4.2 Ohraničení referenční trajektorie 17
4.3 Dynamické programování 18
5. Návrh řízení 22
5.1 Lineární systém 22
5.2 Řízení PID regulátorem 24
5.3 Řízení LQ regulátorem 29
5.4 Řízení MPC regulátorem 32
5.5 Porovnání regulátorů 36
6. Vizualizace 3D 40
6.1 Ruční ovládání 42
7. Závěr 43
Literatura 44
Seznam použitého software I
Obsah přiloženého CD II

Seznam obrázků

2.1 Popis auta ... 2
2.2 Posun auta při natočení kol ... 3
2.3 Připojení vozíku ... 4
2.4 Pohyb vozíku za autem ... 4
2.5 Připojení druhého vozíku ... 6
2.6 Pohyb druhého vozíku ... 6
2.7 popis auta s otáčením 4 kol ... 9
3.1 Obecná mapa ... 10
3.2 Úprava překážek ... 11
3.3 Možné body od překážky ... 11
3.4 Body pro optimální proložení ... 12
3.5 Možnosti propojení ... 12
3.6 Úprava pro výpočet ... 13
3.7 Nalezená trajektorie ... 13
4.1 Změna tvaru překážky ... 15
4.2 Změna překážek ... 16
4.3 Omezení prostoru ... 17
4.4 Omezení prostoru, překážkami ... 17
4.5 Optimální cesta ... 20
5.1 Proložení nelineárního systému ... 23
5.2 Schéma zapojení PD, P ... 24
5.3 Návrh PD ... 25
5.4 Frekvenční a fázová charakteristika návrhu PD ... 25
5.5 Sledování reference PD,P ... 26
5.6 Sledování trajektorie PD,P ... 27
5.7 Úhel odchylky od trajektorie ... 27
5.7 Sledování reference LQ ... 30
5.8 Sledování trajektorie LQ ... 31
5.9 Sledování reference MPC ... 34
5.10 Sledování trajektorie MPC ... 35

5.11A SLEDOVÁNÍ ÚHLU 36
5.11B POROVNÁNÍ ŘÍZENÍ 37
5.11C AKČNÍ ZÁSAHY 37
5.12A SLEDOVÁNÍ ÚHLU 38
5.12B POROVNÁNÍ ŘÍZENÍ 38
5.12C AKČNÍ ZÁSAHY 39
6.1 POHLED SHORA 40
6.2 POHLED ZE STRANY 41
6.3 POHLED ZE PŘEDU 42
6.4 RUČNÍ OVLÁDÁNÍ 42

1.Úvod

Tato práce se skládá z několika částí. První část, a vlastně ta nejdůležitější, je popis problému. Matematické vyjádření chování reálného systému. Je samozřejmé, že popsat reálný model přesně není ani možné. Rovnice by byly složité a pro řízení nevhodné. Proto některé složky, které ovlivňují pohyb jednoduše zanedbáme.

Asi největší ze zanedbaných položek, které můžou značně ovlivňovat reálný systém, je váha. Pokud je například vozík velmi těžký a auto má motor vpředu, díky tření vozíku s vozovkou, může docházet k posunutí konce auta. Ke tření také dochází v kloubu spojení auta s přívěsem. Také jakákoli terénní nerovnost může změnit pohyb.

To byly jen ty nejvýraznější položky, jinak je jich mnohem víc, ale jsou už méně patrné. Všechny tyto uvedené položky jednoduše zanedbáme. Rychlost pohybu uvažujeme konstantní. Díky matematickému popisu dostaneme dynamické rovnice.

Představme si skutečné auto s vozíkem, které couvá. Je jasné, že se jedná o nestabilní systém a při špatném řízení může dojít ke zlomení oje vozíku. Proto dalším úkolem bude navrhnout regulátory, které dokáží bezpečně zacouvat. Musíme otestovat zda jsou regulátory funkční. Pro toto je vytvořené velké parkoviště, na které můžeme umístit překážky ve formě úseček. Dále na ně umístíme odkud má couvat a kam. Navrhneme trajektorii, po které by se couvání mělo provádět, aby auto nenarazilo do žádné překážky.

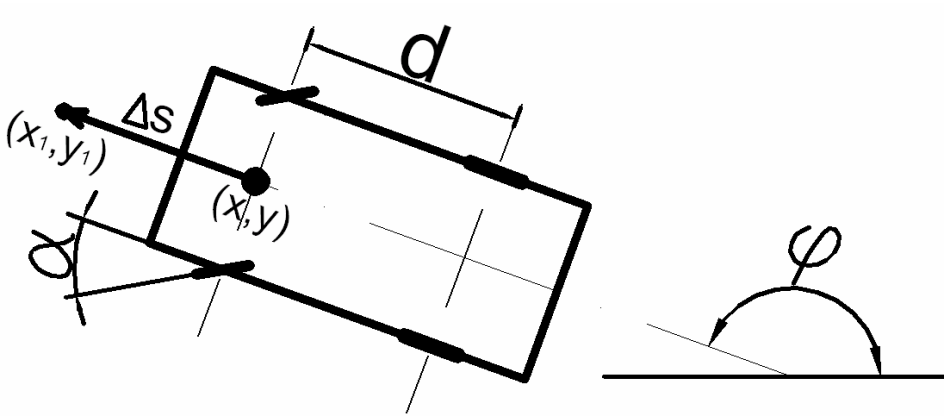
Návrh trajektorie musí být docela přesný, aby nedocházelo ke kontaktu s překážkami. Tato navržená trajektorie bude vstupem pro návrh regulátorů. V této práci se pokusím navrhnout tři - jmenovitě PID, LQ, MPC. Vzhledem k tomu, že se jedná o simulaci, bude záviset na rychlosti výpočtu. Nikdo nebude čekat deset minut než se vypočte návrh řízení. Což může ale znamenat zmenšení přesnosti.

Po návrhu řízení se přechází k vizualizaci. Ta se provede pomocí virtual reality toolboxu. Pro celý výpočet i následné simulace se použije matlab. Díky 3D objektům se vykreslí pohyb mezi překážkami. Dále si uživatel může vyzkoušet, jestli je chopen zacouvat stejně dobře jako regulátor. Ještě doplním, že uživatel zadává překážky i odkud a kam se má couvat.

V této práci se pokusím popsat rozdíl natáčení pouze předních kol a natáčení všech 4 kol auta při couvání s přívěsem. Termín „přívěs“ je dosti obecný a i v simulaci se pro názornost použije klasický vozík. Proto taky nerozlišuji mezi těmito termíny. Zmíním se i o problému dvou vozíků zapojených za sebou.

2. Matematický model

2.1 Auto



Obrázek 2.1 Popis auta

Pro veškerý pohyb modelu auta i vozíku je rychlost konstantní

$$\frac{ds}{dt} = v = konst \qquad (2.1)$$

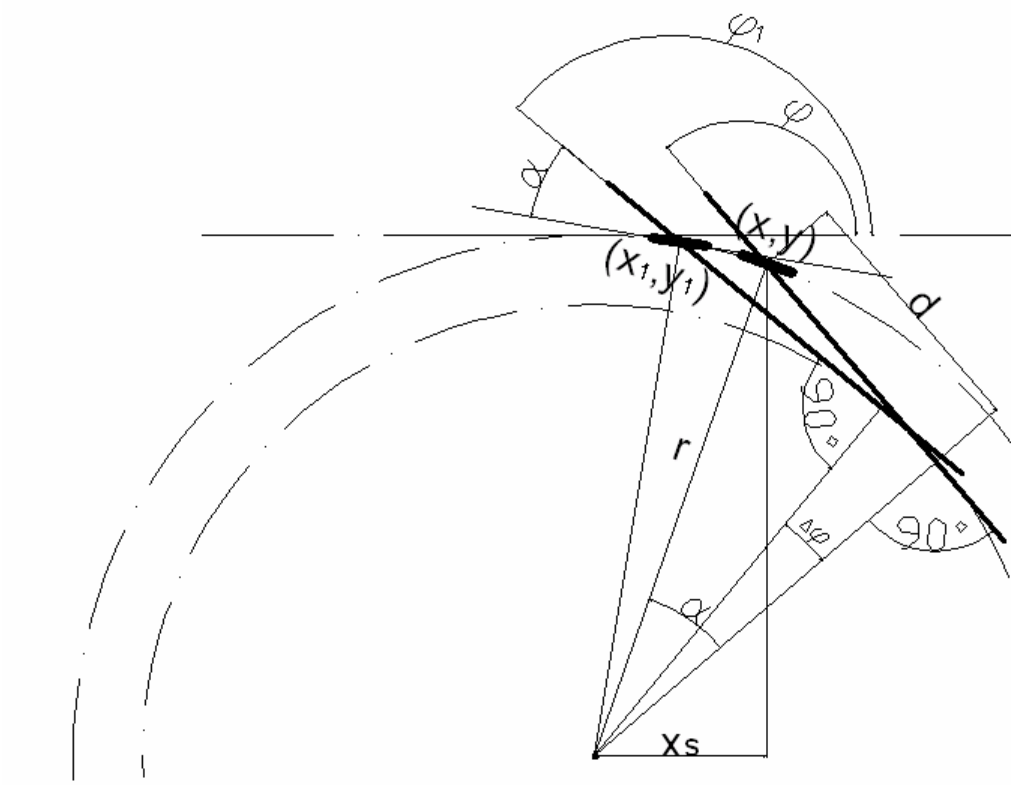
Změna polohy auta o element posunutí *ds*:

$$x_1 = x + \Delta x$$
$$y_1 = y + \Delta y \qquad (2.2)$$

pro auto jedoucí rovně tj. bez natočení kol (α = 0), jsou elementární přírůstky polohy podle rovnice 2.3

$$\Delta x = ds \cdot \cos\varphi$$
$$\Delta y = ds \cdot \sin\varphi \qquad (2.3)$$

Pro případ natočení kol jsou rovnice složitější, pohyb auta je potom vidět na obrázku 2.2. Jedná se o případ, kdy natočení kol není nulové $\alpha \neq 0$.



Obrázek 2.2 Posun auta při natočení kol

Poloměr kružnice jde podle vzorce 2.4 vypočítat pouze pokud $\alpha \neq 0$, jinak je poloměr ∞.

$$r = \frac{d}{\sin\alpha} \qquad (2.4)$$

Přírůstek úhlu pootočení auta o elementární posun *ds* popisuje rovnice 2.5

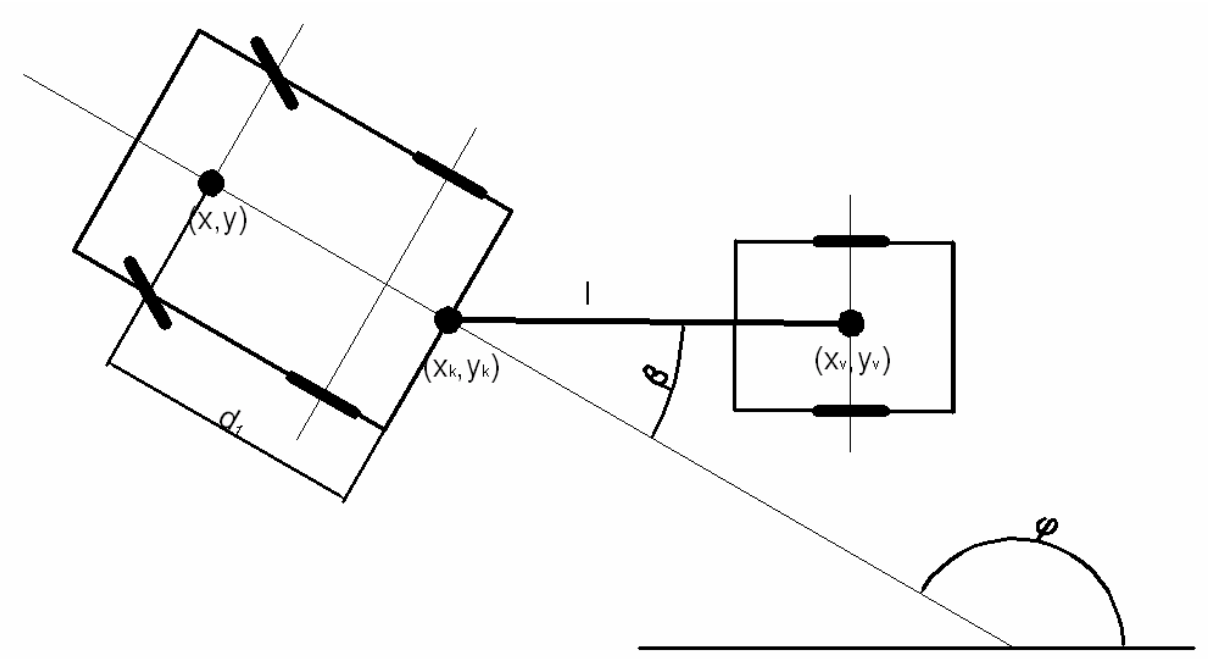
$$\Delta\varphi = \frac{ds}{r} \qquad (2.5)$$

Rovnice 2.6 pro přírůstek pohybu ve směru *x* a *y* při natočení nápravy a pohybu o element *ds*.

$$\begin{aligned}
&\varphi_1 = \varphi + \Delta\varphi \\
&x_s = r \cdot \cos(\varphi - \frac{\pi}{2} + \alpha) \\
&y_s = r \cdot \sin(\varphi - \frac{\pi}{2} + \alpha) \\
&x_{s1} = r \cdot \cos(\varphi_1 - \frac{\pi}{2} + \alpha) \\
&y_{s1} = r \cdot \sin(\varphi_1 - \frac{\pi}{2} + \alpha) \\
&\Delta x = x_{s1} - x_s = r\left(\sin(\varphi_1 + \alpha) - \sin(\varphi + \alpha)\right) \\
&\Delta y = y_{s1} - y_s = r\left(-\cos(\varphi_1 + \alpha) + \cos(\varphi + \alpha)\right) \qquad (2.6)
\end{aligned}$$

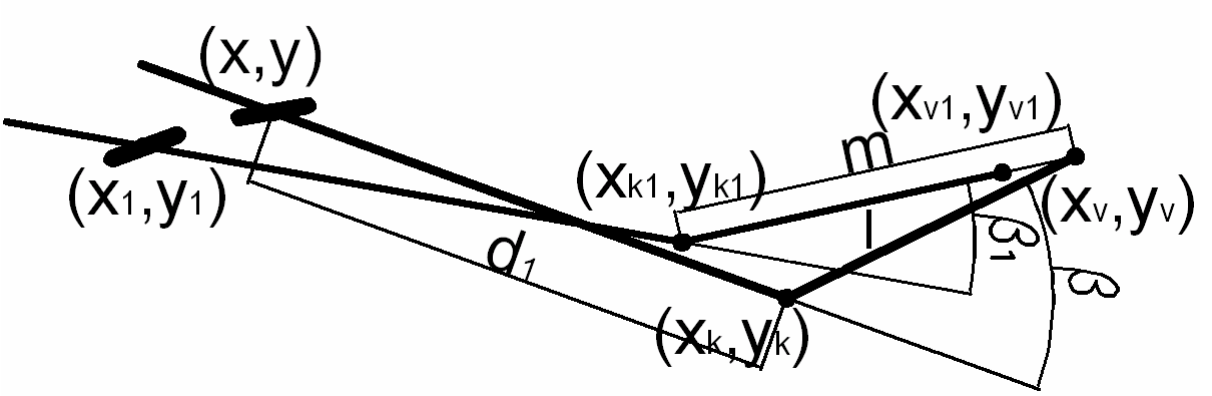
Zapojením vozíku za auto podle obrázku 2.3, bereme v úvahu, že vozík pohyb auta nezmění, auto je v našem případě daleko těžší než vozík.

2.2 Popis přívěsu



Obrázek 2.3 Připojení vozíku

Při pohybu vozíku je nutné přepočítat, kde se nachází koule auta, za kterou je připojen. Obrázek 2.4 ukazuje pohyb vozíku za autem, které se pohnulo o elementární délku *ds*.



Obrázek 2.4 Pohyb vozíku za autem

Umístění koule auta se vypočte podle rovnice 2.7

$$x_k = x - d_1 \cos\varphi$$
$$y_k = y - d_1 \sin\varphi \qquad (2.7)$$

Z polohy tažného zařízení vypočítáme polohu vozíku, za předpokladu znalosti úhlu β rovnice 2.8

$$x_v = x_k - l\cos(\varphi + \beta)$$
$$y_v = y_k - l\sin(\varphi + \beta) \qquad (2.8)$$

Po elementárním posunu o *ds* se poloha závěsu vozíku posune podle vzorce 2.9

$$x_{k1} = x_1 - d_1\cos\varphi_1$$
$$y_{k1} = y_1 - d_1\sin\varphi_1 \qquad (2.9)$$

Dále můžeme popsat původní polohu vozíku, pomoci nových úhlů φ_1 , β_1.

$$\beta_1 = \beta + \Delta\beta$$
$$x_v = x_{k1} - m\cos(\varphi_1 + \beta_1)$$
$$y_v = y_{k1} - m\sin(\varphi_1 + \beta_1) \qquad (2.10)$$

Rovnice 2.8 a 2.10 spojíme a dostaneme dvě rovnice o dvou neznámých m , $\Delta\beta$.

V tomto případě nás zajímá pouze proměnná

$$\Delta\beta = \arctan\left(\frac{\Delta y_k\cos(\varphi_1 + \beta) - \Delta x_k\sin(\varphi_1 + \beta) - l\sin(\Delta\varphi)}{\Delta y_k\sin(\varphi_1 + \beta) + \Delta x_k\cos(\varphi_1 + \beta) + l\cos(\Delta\varphi)}\right) \qquad (2.11)$$

kde

$$\Delta x_k = x_{k1} - x_k$$
$$\Delta y_k = y_{k1} - y_k \qquad (2.12)$$

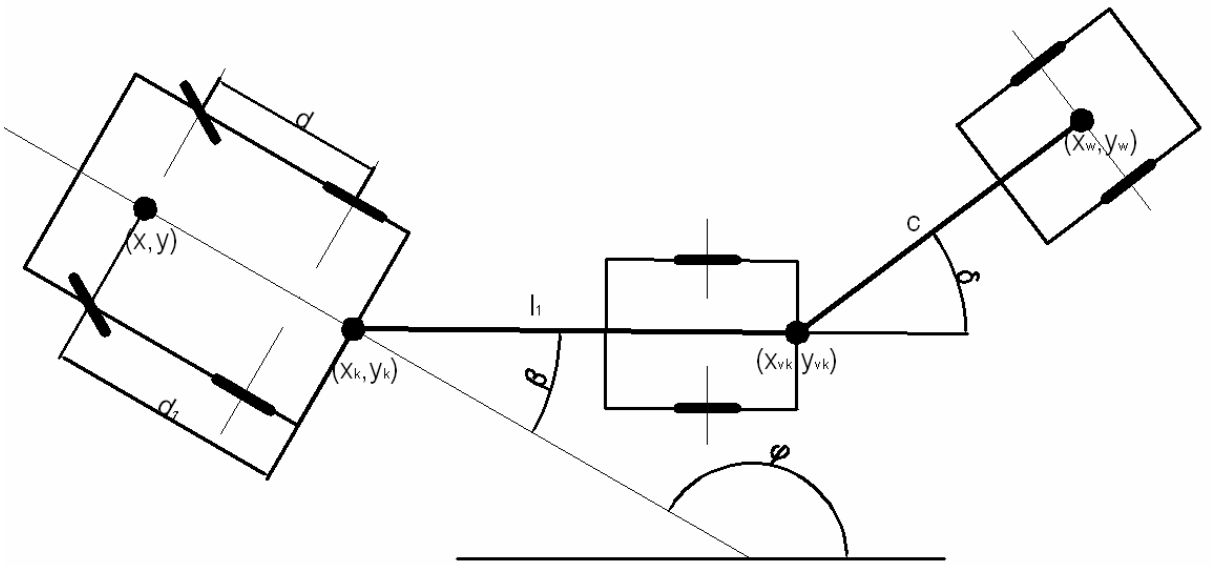
Rovnice 2.11 je velmi obecná, proto do ní dosadíme rovnice 2.6, 2.7, 2.9, 2.12, po zjednodušení dostaneme

$$\Delta\beta = \arctan\left(\frac{l\sin(\Delta\varphi) + r(\cos(\alpha - \beta) - \cos(\alpha - \beta - \Delta\varphi)) + d_1(\sin(\beta + \Delta\varphi) - \sin(\beta))}{-l\cos(\Delta\varphi) - r(\sin(\alpha - \beta) - \sin(\alpha - \beta - \Delta\varphi)) - d_1(\cos(\beta + \Delta\varphi) - \cos(\beta))}\right) \qquad (2.13)$$

V rovnici 2.13 se nevyskytuje úhel φ, protože na něm vůbec nezáleží. Další substituce do rovnice nemají smysl, nic se nevykrátí a zbytečně se zvětší. Pro vypočet je ovšem nutná.

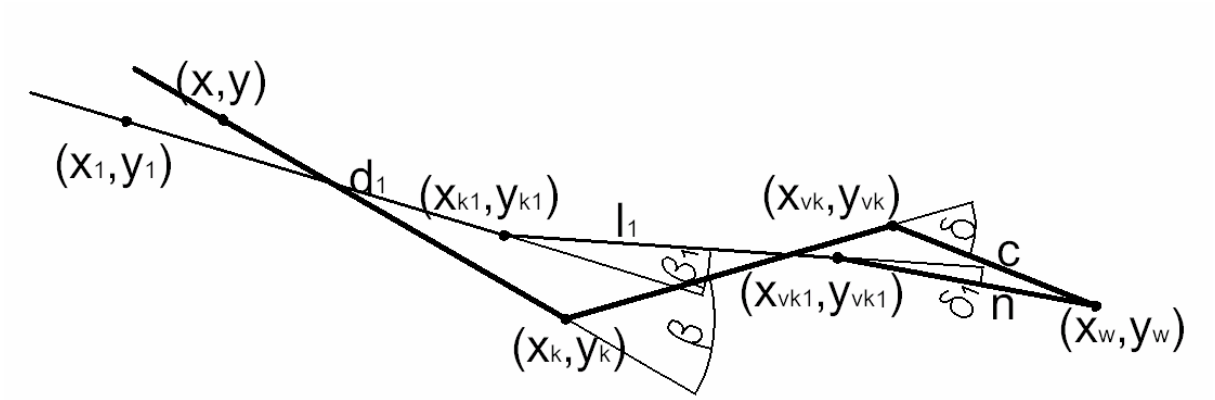
2.3 Další přívěs

Zapojení druhého vozíku k prvnímu i s popisem všech důležitých úhlů a vzdáleností je vidět na obrázku 2.5



Obrázek 2.5 Připojení druhého vozíku

I v tomto případě další připojený vozík nijak neovlivňuje pohyb předchozího vozíku či auta. Pohyb druhého vozíku je vidět na obrázku 2.6 když se auto pohne o elementární délku *ds*.



Obrázek 2.6 Pohyb druhého vozíku

Na konci prvního vozíku je připevněno tažné zařízení, pro upevnění oje druhého vozíku. Rovnice tohoto bodu jsou

$$x_{vk} = x_k - l_1 \cos(\varphi + \beta)$$
$$y_{vk} = y_k - l_1 \sin(\varphi + \beta) \qquad (2.14)$$

Při posunu auta se závěs druhého vozíku posune na pozici

$$x_{vk1} = x_{k1} - l_1 \cos(\varphi_1 + \beta_1)$$
$$y_{vk1} = y_{k1} - l_1 \sin(\varphi_1 + \beta_1) \qquad (2.15)$$

Polohu druhého vozíku, jehož souřadnice jsou uprostřed v ose kol vozíku, jsou

$$x_w = x_{vk} - c\cos(\varphi + \beta + \delta)$$
$$y_w = y_{vk} - c\sin(\varphi + \beta + \delta) \qquad (2.16)$$

Polohu druhého vozíku jde vyjádřit i po posunu

$$x_w = x_{vk1} - n\cos(\varphi_1 + \beta_1 + \delta_1)$$
$$y_w = y_{vk1} - n\sin(\varphi_1 + \beta_1 + \delta_1) \qquad (2.17)$$

Rovnice 2.16 a 2.17 se rovnají. Máme dvě neznámé a nás zajímá pouze

$$\Delta\delta = \arctan\left(\frac{\Delta y_{vk}\cos(\varphi_1 + \beta_1 + \delta) - \Delta x_{vk}\sin(\varphi_1 + \beta_1 + \delta) - c\sin(\Delta\varphi + \Delta\beta)}{\Delta y_{vk}\sin(\varphi_1 + \beta_1 + \delta) + \Delta x_{vk}\cos(\varphi_1 + \beta_1 + \delta) + c\cos(\Delta\varphi + \Delta\beta)}\right) \qquad (2.18)$$

kde

$$\delta_1 = \delta + \Delta\delta$$
$$\Delta x_{vk} = x_{vk1} - x_{vk}$$
$$\Delta y_{vk} = y_{vk1} - y_{vk}$$

Stačí si povšimnout, že obecná rovnice pro první vozík 2.13 a obecná rovnice pro druhý vozík 2.18 jsou skoro stejné. Abychom dostali rovnici 2.18 stačí substituovat do 2.13 následovně

$$\Delta\beta \to \Delta\delta$$
$$\beta \to \delta$$
$$\varphi \to \varphi + \beta$$
$$\Delta\varphi \to \Delta\varphi + \Delta\beta$$
$$\Delta x_k \to \Delta x_{vk}$$
$$\Delta y_k \to \Delta y_{vk}$$

Z toho plyne, že jakýkoliv další přidaný vozík se nemusí odvozovat, ale jen vhodně substituovat proměnné, podle výše zmíněného příkladu. Pro simulaci se používají hodnoty.

$$ds = 0.1m \qquad d = 2.5m \qquad d_1 = 3.5m \qquad l = 2.1m$$

2.4 Zjednodušení rovnic

Odvozené rovnice 2.6 a 2.13 mají nespojitost při nulovém natočení kol $\alpha = 0$. V tomto bodě neexistuje derivace. A to značně komplikuje samotný návrh řízení. Dá se navrhnout přepínatelný systém, když rovnice 2.6 v okolí nulového řízení nahradíme rovnicemi 2.3. Přepínatelné systémy komplikují návrh řízení. Nejlepší bude pokud upravíme rovnice tak, aby byly v celé oblasti řízení spojité. Všechny rovnice které nás zajímají obsahují $\Delta\varphi$.
Z rovnice 2.5 pro výpočet úhlu $\Delta\varphi$ je jasné, že pokud dělíme malé číslo velkým dostaneme minimální číslo. Proto můžeme využít známého jevu goniometrických funkcí:

$$\sin(\Delta\varphi) \approx \Delta\varphi$$
$$\cos(\Delta\varphi) \approx 1$$

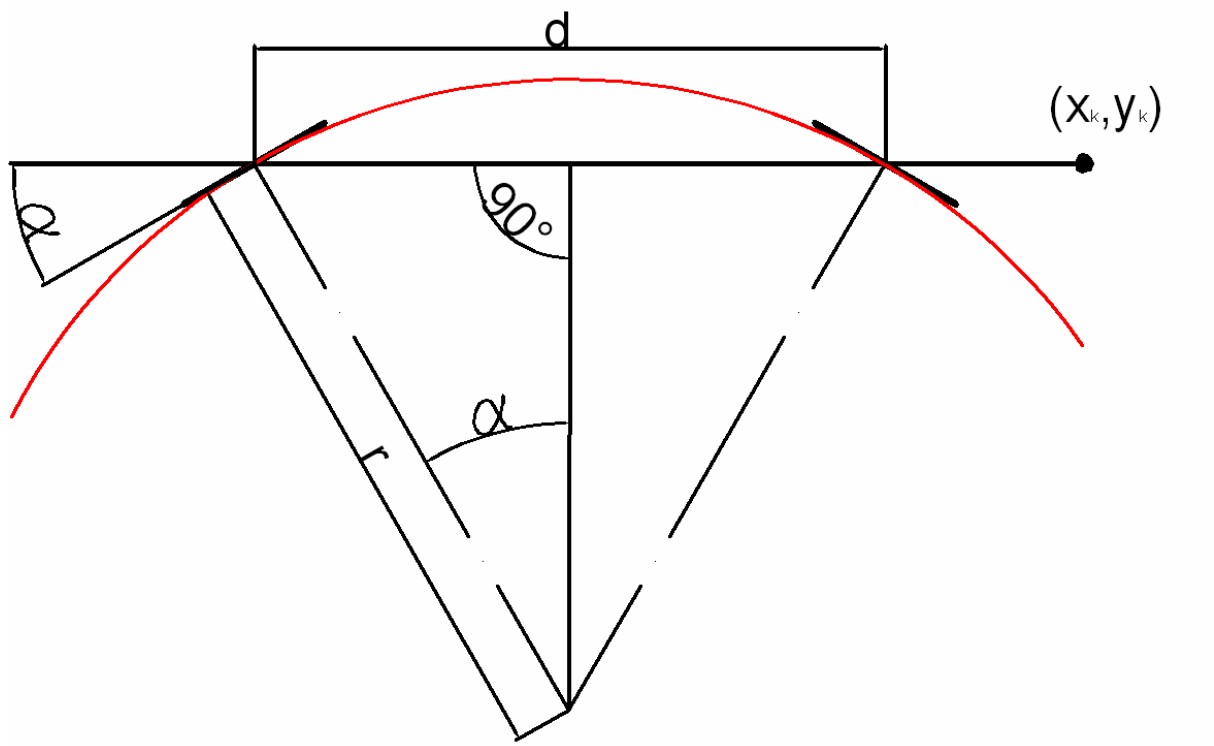
Toto zjednodušení platí pouze do *5°*. Čím více se této hranici blížíme, tím větší chyba nastává. Přiblížení k této hranici závisí na *ds*. Takže stačí udržovat *ds* co nejmenší - v okolí *0*. Problém se sám vyřešil, protože při zvětšování *ds* není zaručena stabilita řízení. Pro řízení i návrhy regulátorů se využívá $ds = -0.1$, kde mínus značí směr pohybu couvání. To znamená, že auto urazí za jeden krok 0.1m. Při aplikování zjednodušení dostaneme po úpravách následující rovnice:

$$\begin{aligned}
\Delta\varphi &= \frac{ds \cdot \sin(\alpha)}{d} \\
\Delta x &= ds \cdot \cos(\alpha + \varphi) \\
\Delta y &= ds \cdot \sin(\alpha + \varphi) \\
\Delta\beta &= -arctg\left(\frac{2 \cdot ds \cdot l \cdot \sin(\alpha) + ds \cdot d1 \cdot \sin(\alpha + \beta) + ds \cdot (d1 - 2 \cdot d) \cdot \sin(\alpha - \beta)}{2 \cdot l \cdot d + ds \cdot d1 \cdot \cos(\alpha + \beta) + ds \cdot (-d1 + 2 \cdot d) \cdot \cos(\alpha - \beta)}\right)
\end{aligned} \tag{2.19}$$

Rovnice 2.19 neobsahují $\Delta\varphi$ ani *r* a jsou spojité v celém stavovém prostoru potřebném pro řízení. Tyto rovnice se dál neupravují. Tento výsledný stav se používá pro řízení.

2.5 Otáčení čtyř kol

Při otáčení všech čtyř kol auta se jeho pohyb i vliv na vozík změní. Obrázek 2.7 popisuje situaci, kdy se natáčejí obě nápravy auta o stejný úhel α. Červený oblouk značí trasu, po které se nápravy budou pohybovat. V tomto případě se obě nápravy pohybují v zatáčce po stejné kružnici.



Obrázek 2.7 popis auta s otáčením 4 kol.

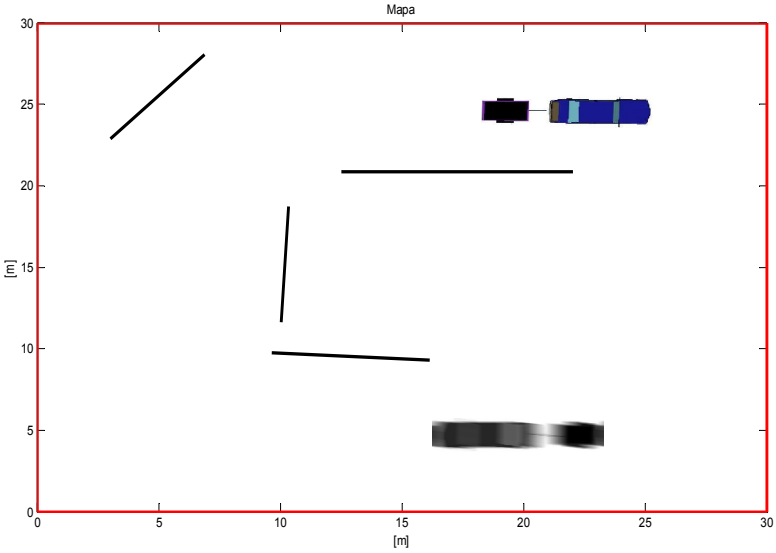
S obrázku 2.7 je vidět, že poloměr otáčení auta *r*, se vypočítá jako

$$r = \frac{\frac{1}{2}d}{\sin\alpha} \qquad (2.20)$$

Rovnici 2.20 porovnáme s rovnicí 2.4. Zjistíme, že pokud má auto obě natáčené nápravy, stačí vzdálenost mezi nimi podělit dvěma. Tuto vzdálenost dosadit do rovnice 2.4 pro otáčení jen jedné nápravy. A další chování se odvíjí právě od vzdálenosti *d,* vzhledem k tomu, že závisí jen na této konstantě. Rovnice pro pohyb auta a vozíku jsou stejné, jako v předešlém případě jen se substitucí $d \rightarrow d/2$.

3. Návrh referenční trajektorie

Pokusím se navrhnout trajektorii, po které je přívěs schopen zacouvat tak, aby se vyhnul překážkám a pod správným úhlem dorazil na požadované souřadnice. Nejlepší možný způsob je využít variačních metod. Výsledek je optimální z hlediska řízení (v našem případě minimální řízení odpovídá co nejmenší změně volantu v průběhu celého couvání). Ale tuto metodu se mi podařilo použít jen v jistých speciálních případech. Obecné řešení jsem nenalezl. Z tohoto důvodu jsem navrhl vlastní algoritmus. Obrázek 3.1. nazveme „mapou“. Při zadávání do mapy je požito myši. Mapa je dostatečně velká, je možné si ji představit jako velké parkoviště, jehož okraje znázorňuje červený rámeček. Rozměry na osách mapy jsou v metrech. Díky velikosti mapy, která zůstává neměnná, může uživatel vykreslit přesně požadovanou situaci. Na mapě se nacházejí dva hlavní objekty. Modré auto s přívěsem znázorňuje startovní polohu, jeho šedý rozmazaný stín představuje cílovou pozici auta s přívěsem. Přívěs je vždy pod stejným úhlem jako auto a to startovní i cílový. Překrývání startovní a cílové pozice je zakázáno, není možné ani umístit auto s přívěsem mimo mapu či přes překážku.

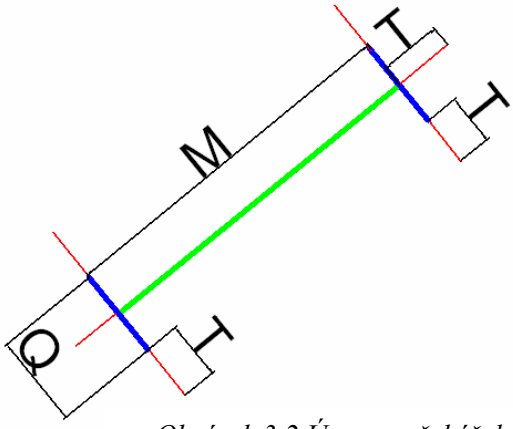


Obrázek 3.1 Obecná mapa

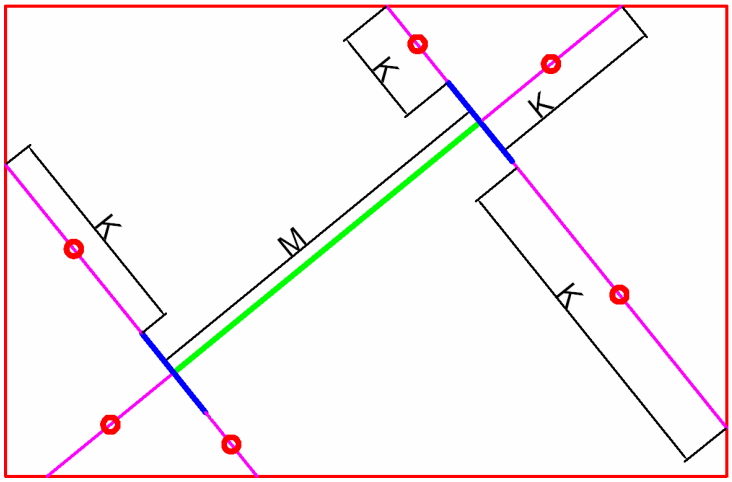
Překážky se zadávají nejdříve. Mají tvar úsečky a jako takové je jejich šířka zanedbatelná. Zadavatel určuje jejich polohu, délku i úhel natočení. Překážky se mohou vzájemně překrývat. Nemohou se však umístit mimo mapu. Jejich počet je omezen. Složitost výpočtu trajektorie roste lineárně v závislosti na počtu překážek. Čtyři překážky stačí pro návrh většiny problémů. Je potřeba si uvědomit, že okraje tvoří další čtyři omezení. Při chybějících překážkách možno využít okraje.

3.1 Body pro trajektorii

Návrh referenční trajektorie, pomocí níž je schopen regulátor zacouvat s přívěsem, musí odpovídat vlastnostem systému. Víme, že velké změny úhlu nebude moci regulátor sledovat. Pro začátek upravíme překážky. Na obrázku 3.2. je znázorněna původní překážka délky *M* zelenou úsečkou. Pro správný výpočet se přidají dvě úsečky. Znázorněny modrou barvou o délce *Q* budeme je nazývat „příčníky“. Tyto příčníky jsou ovšem co nejkratší. Na obrázku 3.2. je jejich délka záměrně zvětšena, aby šlo vidět jejich umístění. Modré příčníky jsou kolmé na překážku a jejich střed leží na koncích překážky. Příčníky jsou přidány i na počáteční a cílovou pozici auta. Díky těmto příčníkům a překážkám se vypočítají body pro případné proložení. Každá překážka , příčníky i auta se prodlouží dokud nenarazí na jinou překážku či okraj. Toto prodloužení je znázorněno na obrázku 3.3. fialovou barvou s délkou *K*. Pro názornost je použita pouze jedna překážka a proto naráží pouze do okrajů.



Obrázek 3.2 Úprava překážek

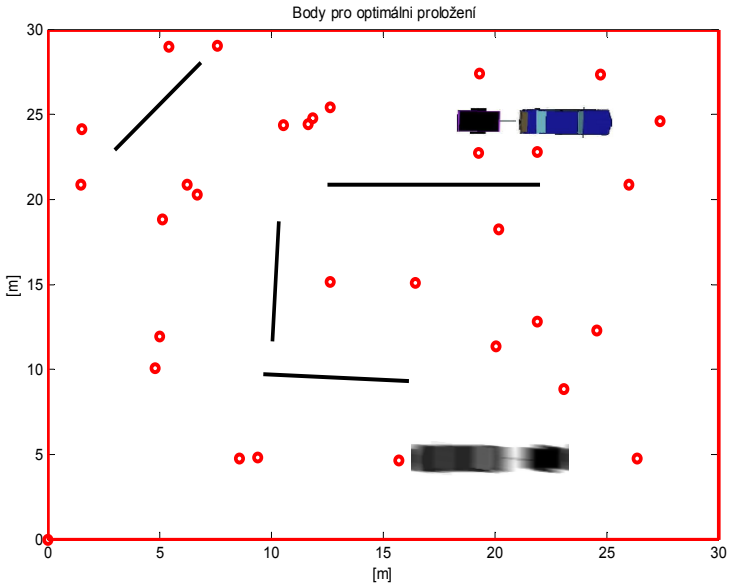


Obrázek 3.3 Možné body od překážky

Body, které nás zajímají, leží přesně v polovině úsečky *K*. Jsou znázorněny červeným kroužkem. Pokud ovšem délka *K* je kratší než šířka auta, není na tuto pozici bod umístěn. To samé platí, pokud je bod blíž k okraji nebo překážce než je polovina šířky auta. Každá překážka tedy za ideálních okolností dává 6 bodů. Pro čtyři překážky a dvě auta to znamená 36 bodů. Například kdyby se použila čtvercová síť s roztečí 0.5m, bylo by bodů 3600. Nalezení optimální cesty je známý problém „obchodního cestujícího“. Proto bylo nutné nalézt minimální počet bodů, které jsou potřeba pro nalezení cesty v jakémkoli zadání mapy.

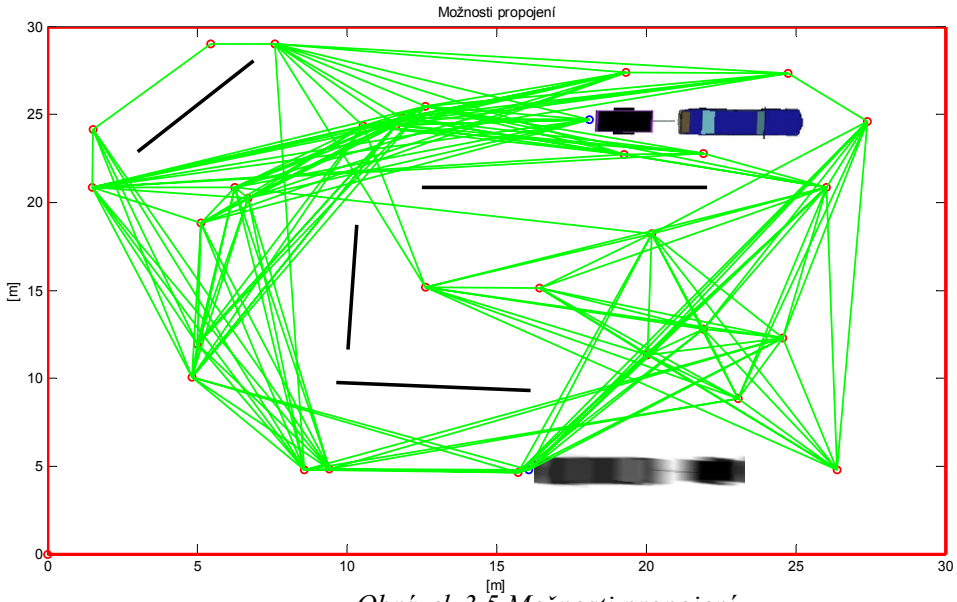
3.2 Možnosti propojení

V minulé podkapitole jsme si vypočítali body, které stačí proložit a dostaneme hrubý rys referenční trajektorie. Pro lepší představu obrázek 3.1 byl doplněn o referenční body, jak můžeme vidět na obrázku 3.4.



Obrázek 3.4 Body pro optimální proložení

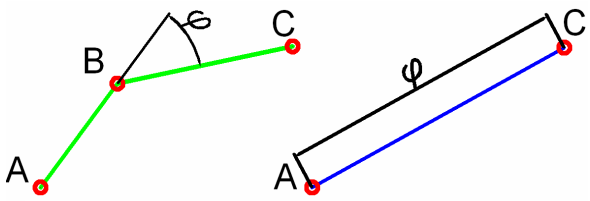
Je vidět, že většina bodů je mimo oblast která nás zajímá. Ty body, které potřebujeme k nalezení cesty mají správnou polohu. Pro jiné rozložení mapy povedou k řešení jiné body. Nejprve proložíme každý bod s každým. Propojení všech bodů se všemi znázorňuje obrázek 3.5. Aby spojení dvou bodů nevedlo příliš blízko vedle překážky, udělá se úprava znázorněná na obrázku 3.2. Tato úprava je prosté prodloužení překážek a příčníků o polovinu šířky auta. Na obrázku 3.2 se jedná o červenou úsečku s kótou *T*. Pokud takto prodloužené překážky a příčníky křižuje nějaká spojnice dvou bodů, potom tyto dva body nejsou propojeny. Na obrázku 3.5 je pěkně vidět, jak se propojení bodů drží v bezpečné vzdálenosti od překážek. Můžeme si představit, že se auto s přívěsem po těchto propojeních pohybuje. V případě příliš blízkého průjezdu kolem překážky by došlo k poškození auta.



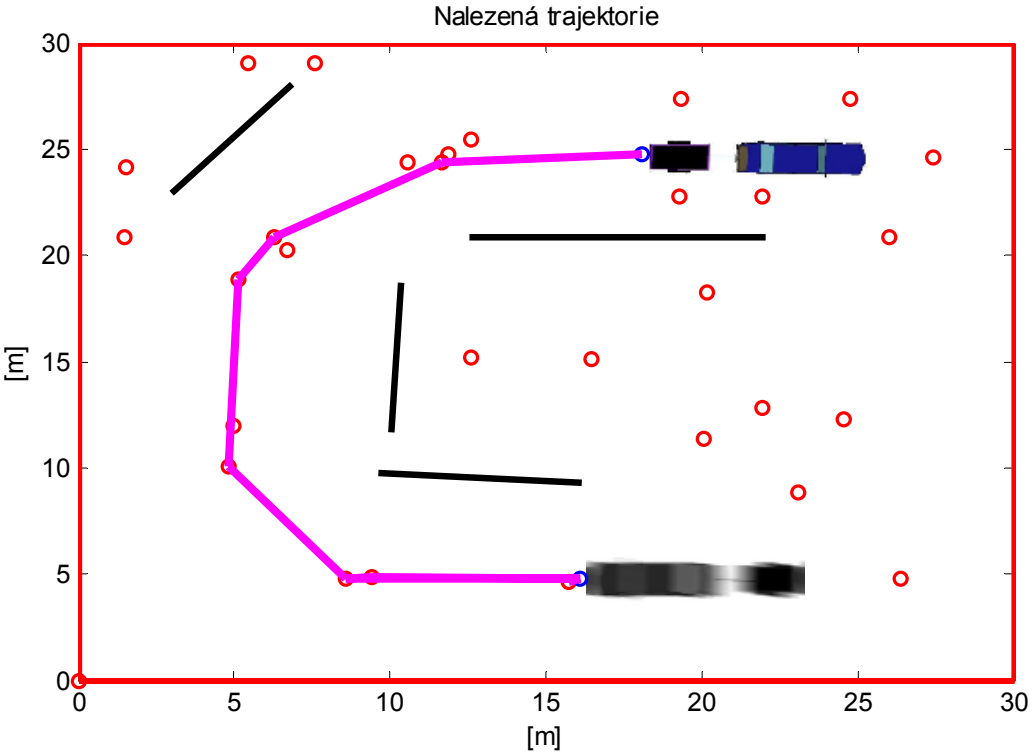
Obrázek 3.5 Možnosti propojení

3.3 Optimální proložení bodů

Propojení všech bodů je hodně i po vyřazení přebytečných. Mezi těmito propojkami hledáme ty, co nás dovedou od startovního auta k cílovému, s co nejmenší změnou úhlu. Úhel, který nás zajímá mezi body A a C, je zobrazen na obrázku 3.6 jako φ. Vlevo na obrázku 3.6 vidíme původní situaci, vpravo vidíme jak se situace upraví pro výpočet. V danou chvíli nás nezajímá bod B. Pro zvýhodnění menších úhlů se používá kvadrát. Tím se vyřeší problém i ze záporností hran. Pro výpočet se používá upravený Dijkstrův algoritmus, který potřebuje mít všechny hrany s kladným ohodnocením. Takže se hledá minimální kostra ve smyslu kvadratických úhlů. Dijkstrův algoritmus pro názornost popisuji níže. Je potřeba si uvědomit, že převedení úhlu v hranu podle obrázku 3.6. se děje každý krok algoritmu. A algoritmus se nepohne do bodu *C,* ale jen do bodu *B*. odkud se znovu pro další krok vypočítají všechny úhly do bodů, se kterými je propojen výchozí bod. Výpočet se zastaví až tehdy, kdy auto dosáhne cílového bodu *B* a vozík bodu *C*. Na obrázku 3.7 je vidět nalezení trajektorie s předem vypočítaných bodů.



Obrázek 3.6 Úprava pro výpočet



Obrázek 3.7 Nalezená trajektorie

Algoritmus výpočtu:

Vstup: ohodnocený graf *G* pro počáteční uzel *s*, který odpovídá startovací poloze vozíku.
Výstup: pole uzlů *D(k)*, udávající nejkratší vzdálenost mezi uzlem *s* a uzlem *k – cílová poloha vozíku*

INICIALIZACE:

Do množiny bodů *X* viditelného z počátečního uzlu *s*.

Vytvoř asociativní pole čísel pro každý viditelný uzel *D(u)*.

Inicializuj hodnoty pole *D*:

Počáteční uzel $s = 0$

Každý uzel *u* z množiny *X* = ohodnocení hrany *(s,u)* po přepočtu φ^2

VÝPOČET:

Opakuj dokud není minimální hodnota *D(w)* $w = k$

Najdi uzel *w* s minimální hodnotou *D(w)*.

Přidej uzel *w* do množiny *X*.

Pro každý uzel *u* sousedící s uzlem *w,* který není v množině *X* proveď:

hodnota *D(u)* je minimum ze stávající hodnoty a *D(w)* plus ohodnocení hrany *(w,u)*-do *D()* se nepřidá *w,* ale podle obrázku 3.6, kde pro tento případ *u* je *A, w* je *C* se do *D()* přidá bod *B* s ohodnocením $\varphi^2 + u$.

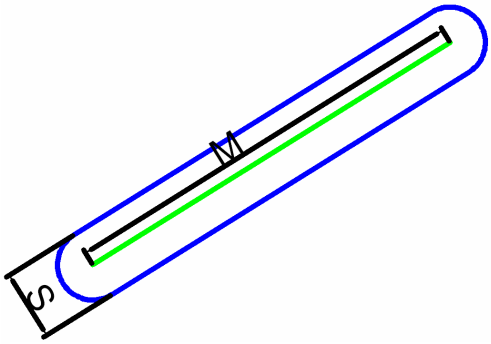
Nalezená trajektorie je jistě optimální z hlediska minimální změny úhlu vzhledem k daným bodům. Ale i taková trajektorie obsahuje velké změny úhlů. Pokud regulátor nemá predikční schopnosti, jinými slovy auto si nenadjede pokud neví že přijde prudká zatáčka, tak pravděpodobně do překážky nabourá. Vozík může také ztratit stabilitu a zlomit oje vozíku. Proto chceme, aby to zvládly uřídit všechny 3 regulátory, které použijeme. Musíme tento prvotní návrh trajektorie proložit takovou, která lépe odpovídá našemu systému.

4. Proložení referenční trajektorie

Před popsáním samotné metody proložení referenční trajektorie, je nutné uvést metody, bez kterých by algoritmus proložení nemohl fungovat. Jedná se o dvě základní - přesnější popis překážek a ohraničení kolem referenční trajektorie.

4.1 Popis překážek

Při zadávání do mapy je překážka pouhá úsečka a jako taková nemá šířku. Auto má přesně danou šířku. Pro snadný výpočet je nutné, aby bylo auto pouhý bod. Takže je logické, že překážky musí změnit svůj tvar. Tímto problémem jsme se zabývali už v předchozí kapitole. Tam nám stačilo pouze její prodloužení, jak je vidět na obrázku 3.2. Fungovalo to spolehlivě, protože se jednalo o vzájemnou polohu úseček. Teď to nestačí. Při prokládání trajektorie nebo i při výpočtu řízení dostaneme bod se souřadnicemi $[x, y]$, u kterého se musí určit zda je mimo překážku. Ideální situace rozšíření překážky je vidět na obrázku 4.1. Zelená úsečka délky M je původní překážka. Modrý obal kolem ní je ideální úprava pro výpočet. Obal má šířku auta. Je to opis kolem původní překážky ve vzdálenosti půlky šířky auta. Problémem je matematický popis. Z hlediska rychlosti výpočtu je potřeba jednoduchý vzorec. Obal velmi připomíná elipsu, ale ta není moc vhodná. U konců se značně zužuje, což by způsobilo velké chyby při výpočtu. Vyšší mocniny elipsy se k ideálnímu tvaru přibližují. Obecné parametrické rovnice 4.1 popisující vyšší mocniny někdy nazývané termínem „Superellipse“.



Obrázek 4.1 Změna tvaru překážky

$$x(\theta) = |\cos(\theta)|^{\frac{2}{n}} \cdot a \cdot sign(\cos(\theta))$$
$$y(\theta) = |\sin(\theta)|^{\frac{2}{n}} \cdot b \cdot sign(\sin(\theta)) \qquad (4.1)$$

Po zkoušení se ukázalo jako nejlepší proložení n=4, vyšší řády už měly málo zaoblené rohy. Parametrické rovnice popisují střed elipsy v počátku kartézských souřadnic.

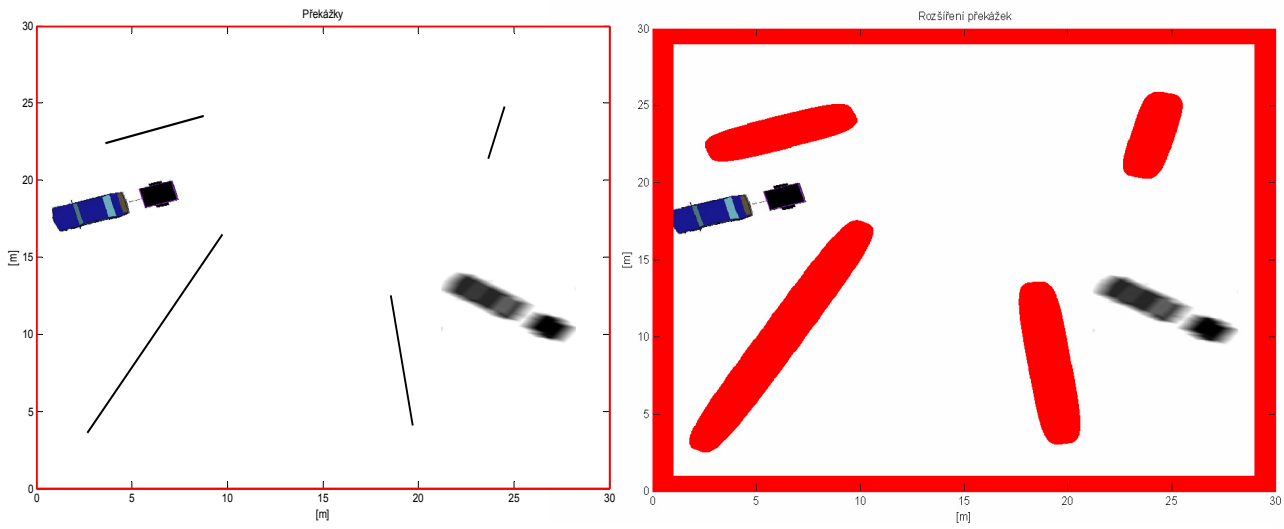
Počátek posuneme pouhým přičtením nových středů. Větší problém je natočení elipsy. Standardně jsou poloosy rovnoběžné s osami. Vzorec 4.2 popisuje obecné natočení souřadnic $[x, y]$ o úhel φ.

$$x1 = x \cdot \cos(\varphi) + y \cdot \sin(\varphi)$$
$$y1 = -x \cdot \sin(\varphi) + y \cdot \cos(\varphi) \qquad (4.2)$$

Nové souřadnice $[x1, y1]$. Dosazením rovnic 4.1 do rovnic 4.2 a připočtením posunutí středu $[xs, ys]$, dostaneme nové parametrické rovnice odpovídající přesně našim požadavkům. Takto nově upravené parametrické rovnice převedeme na obecnou rovnici 4.3.

$$\left(\frac{(y - ys)\cdot\cos(\varphi) - (x - xs)\cdot\sin(\varphi)}{b}\right)^4 + \left(\frac{(x - xs)\cdot\cos(\varphi) + (y - ys)\cdot\sin(\varphi)}{a}\right)^4 = 1 \qquad (4.3)$$

Rovnice znázorňuje okraj překážky. Jakmile se rovnice změní na nerovnici, přímo nám indikuje, zda je bod v překážce či mimo ni. Stačí zadat 5 parametrů pro každou překážku. Kde parametr *b,* malá poloosa, je pro všechny stejná s velikostí poloviny šířky auta. Velká poloosa *a* se rovná polovině délky překážky plus *b*. Úhel natočení překážky je φ. Střed překážky $[xs, ys]$. Jak se změní mapa zvětšením překážek ukazuje obrázek 4.2. Vlevo je původní mapa, vpravo je upravená mapa o rozšíření překážek a okrajů.

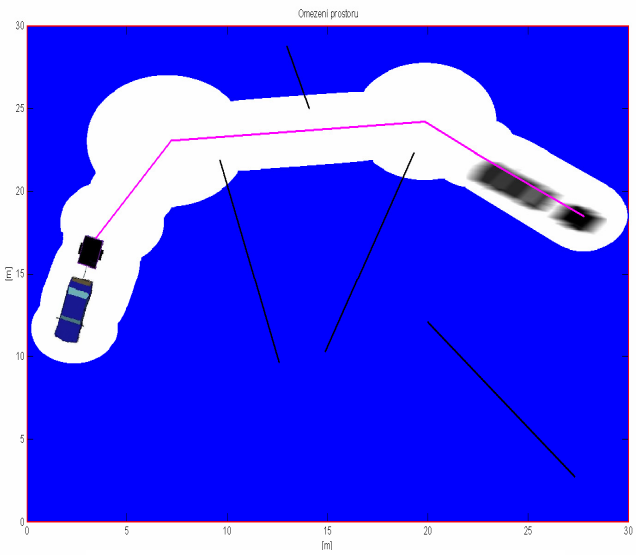


Obrázek 4.2 Změna překážek

4.2 Ohraničení referenční trajektorie

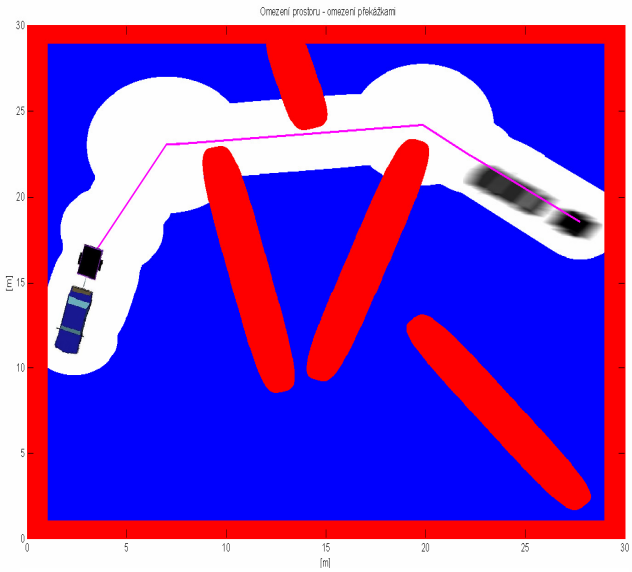
Při prokládání prvního návrhu referenční trajektorie se musí omezit prostor. Pokud bychom to neudělali, řešili bychom stejnou úlohu jako na začátku. Pro ohraničení použijeme to samé jako v předchozím případě, rovnici 4.3. Jediná změna v pěti parametrech je *b=2,2.*
To ovšem nestačí. Ještě musíme umístit do bodů, kde se trajektorie láme, kružnice. Rovnice 4.4 popisuje právě tuto kružnici. Pravá strana rovnice představuje kvadratický poloměr. Mění se v závislosti na úhlu φ. Je to ten samý úhel jako na obrázku 3.6

$$(x - xs)^2 + (y - ys)^2 = 2 + \frac{6 \cdot |\varphi|}{\pi} \qquad (4.4)$$



Obrázek 4.3 Omezení prostoru

Tato změna poloměru zajistí velké změny úhlu. Tak vznikne dostatečný prostor pro najetí do zatáčky. Na následujícím obrázku 4.3 tento efekt kružnice můžeme pozorovat. Bílá místa jsou oblasti, kam se může pohybovat. V kritických oblastech má tak větší prostor pro manévrování. Tento způsob není úplně ideální, ale je velmi rychlý a jednoduchý. Na obrázku nejde vidět rozložení překážek a s jejich omezením prostoru se musí taky počítat. Obrázek 4.4. je doplněn i o omezení překážkami. Tím se ještě víc omezí prostor pro proložení.



Obrázek 4.4 Omezení prostoru, překážkami

4.3 Dynamické programování

Omezení prostoru bylo nezbytné k zrychlení výpočtu. Při dynamickém programování se musí ještě zvolit krok posunu. Popis dynamického programování zde neuvádím, protože je možné ho nalézt buď [1] nebo obecněji v [2]. Když bude tento krok příliš malý, výpočet bude trvat velmi dlouho. Naopak, pokud bude krok příliš dlouhý, nemusí být řešení vůbec nalezeno. Využijeme vlastnosti dynamického programování a budeme se při výpočtu pohybovat od cíle ke startovní poloze. To znamená, že nebudeme couvat, ale pohybovat se dopředu. Tím odpadne problém s nestabilitou přívěsu při couvání. A to také znamená, že můžeme omezit stavy řízení. Ideální je omezení na tři stavy: pohyb rovně, pohyb s maximální výchylkou doleva, pohyb s maximální výchylkou doprava. Popsáno matematicky:

$$a \in \left\{ -\frac{\pi}{7}; 0; \frac{\pi}{7} \right\}$$

Kde *a* je úhel natočení předních kol auta. Značení zůstává stejné jako v druhé kapitole. Pohyb auta s přívěsem udávají rovnice 4.5. Kde *t* je krok, o kolik se auto posune vpřed. Krok *t = 1* odpovídá posunu o 0.1m. Krok *t = 10* je posun o jeden metr.

$$\begin{aligned} f &= \frac{ds \cdot t \cdot \sin(\alpha)}{d} + f_0 \\ x &= \frac{d \cdot \sin\left(\frac{ds \cdot t \cdot \sin(\alpha)}{d}\right) - d \cdot \sin(f_0) + x_0 \cdot \sin(\alpha)}{\sin(a)} + x_0 \\ y &= \frac{-d \cdot \cos\left(\frac{ds \cdot t \cdot \sin(\alpha)}{d}\right) + d \cdot \cos(f_0) + y_0 \cdot \sin(\alpha)}{\sin(\alpha)} + y_0 \end{aligned} \qquad (4.5)$$

Zdánlivě složité rovnice jsou obecné. Pro výpočet, kde *a* nabývá pouze 3 stavů a *t* je konstantní po celou dobu, se mění pouze čtyři parametry $[f_0, x_0, y_0, \beta_0]$. Rovnice pro výpočet úhlu natočení vozíku vzhledem k autu, označován β není uvedena v rovnicích 4.5. Obecný tvar je příliš složitý, a proto ani při vypočtu proložení trajektorie se nepoužívá. Vzorec popisující jeden krok úhlu β je uveden v druhé kapitole 2.13. Díky pohybu vpřed dochází jen k malým výchylkám v rozmezí $\beta \in \langle -1, 1 \rangle$. Proto je možno použít lineární

proložení s minimální chybou. Rovnice 4.6 popisuje posun úhlu β o jeden krok. Obecně pro *t* kroků si jistě každý dopočítá. Obecný vývoj stavu pro lineární stacionární rovnice nalezneme v [4].

$$\beta = 0.965 \cdot \beta_0 - 0.0524 \cdot \alpha \qquad (4.6)$$

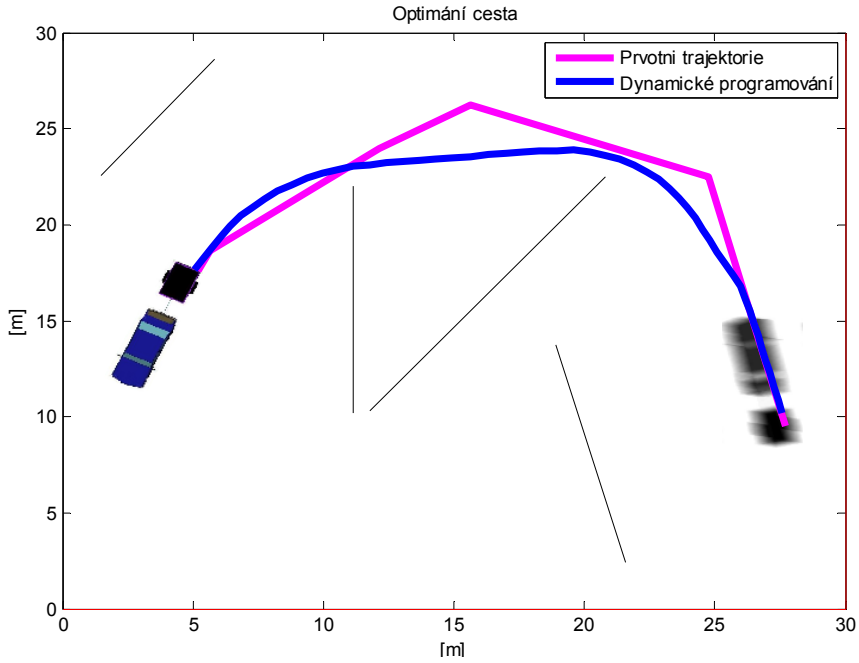
Tím jsme popsali rovnice pohybu. Polohu vozíku určíme z rovnice 2.8. Právě poloha vozíku se musí pohybovat v omezeném prostoru trajektorie. Pohyb dlouhý 10m při kroku $t = 2$ je $\frac{10}{t \cdot 0.1}$ =50, takže potřebuje 50 iterací k cíli, při 3 možnostech vstupu. Každý krok je počet možností 3^{50}, omezení kolem trajektorie zmenší počet řešení na 3^{20}. Jedná se o přibližný výpočet závislý na konkrétní mapě. Ukazuje nám nedostatečné omezení stavového prostoru. Pro zmenšení možností zvětšíme krok. Jako nejvýhodnější z hlediska rychlosti se ukázal *t* = *7*, takže se auto pohne při jedné iteraci o 0.7m.

Další ořezání možností se provádí pomocí mřížky. Používá se 3 rozměrná matice, kde rozměry jsou $[f, x, y]$. Rozměry této matice závisí na mapě. Úhel *f* se mění o konstantní hodnotu proto první rozměr je jasně určený. Další dva rozměry jsou stejné a závisí na tom, jak moc chceme řešení ořezat. Pro $t = 7$ se ukázalo jako nejlepší čtverec o straně 0.2m. Funguje to tak, že první vypočítaný bod, který se v dané oblasti objeví, zapíše do matice logickou 1. Všechny ostatní vypočtené možnosti, které mají stejný úhel a spadají do stejného čtverce o straně 0.2m, nejsou používané dál ve výpočtu a jsou odstraněny. Výpočet skončí buď nalezeným řešením, nebo končí po *n* iteracích.

Nemůžeme počítat, že nalezené řešení bude přesná poloha auta na startu. To by platilo jen v případě velmi jemného kroku a neomezování stavového prostoru. V našem případě se spokojíme, přiblíží-li se poloha auta na vzdálenost 0.15m ke startovní poloze auta. Poloha vozíku se musí přiblížit nejméně na stejnou vzdálenost ke startovní poloze vozíku. Všechny možnosti, kam se auto pohnulo, se každou iteraci ukládají i s odkazem na minulý stav. Na konci se cílové řešení přepočítá zpátky díky těmto odkazům. Tak dostaneme přesnou referenční trajektorii, podle které jsou schopny všechny regulátory dojet k cílové pozici.

Tímto způsobem nalezená trajektorie má ovšem i své nevýhody. Změny úhlů odpovídají možnostem systému. Nalezená cesta bude určitě ta nejkratší možná z hlediska omezení. To ale způsobí, že při sledování této reference se budeme pohybovat velmi blízko kolem překážek. Oblouky mají minimální možné poloměry. To je také důvod, proč se

záměrně používá na hledání trajektorie maximální výchylka kol $a = \pm \pi/7$. Pro řízení je umožněna výchylka kol $a = \pm \pi/6$. Větší výchylka umožňuje lepší sledování reference a také menší pravděpodobnost kolize. Maximální počet *n* iterací závisí na délce prvotní referenční trajektorie. Délka prvotní trajektorie se vydělí 0.7m a připočte se pár kroků navíc, tak se dostane *n*. Jsou i speciální případy, kdy nedojde k nalezení řešení. Pro takové stavy se provede výpočet znovu s krokem *t*=20 pouze s omezením překážek a s mřížkou s roztečí 0.4m. Takto nalezená trajektorie se označí jako prvotní a provede se znovu její proložení jako v předcházejícím případě.



Obrázek 4.5 Optimální cesta

Na obrázku 4.5 je vidět efekt proložení i jeho důsledky. Pohybuje se velmi těsně kolem překážek. Kolize s překážkou při řízení většinou nenastane. Při sledování reference mají regulátory víc odstředivý charakter v zatáčkách, reagují o krok později. Výjimkou je prediktivní regulátor s klouzavým horizontem.

Algoritmus výpočtu:

Vstup: startovní poloha auta s přívěsem , cílová poloha auta s přívěsem, omezení překážkami a omezení pohybu kolem prvotní trajektorie.

Výstup: optimální trajektorie s konstantním krokem

INICIALIZACE:

$[f, x, y, b]$ přiřaď cílovou polohu auta s přívěsem.

Vytvoř tří rozměrnou matici logického typu H(*f, x, y*).

Vytvoř matici G pro historii a na první pozici přiřaď $[f, x, y, b]$

VÝPOČET:

Opakuj i = 1 : *n*

Zjisti velikost G(i) přiřaď do *m*

Opakuj j = 1 : *m*

Pro G(i,j) vypočti 3 nové stavy $[f, x, y, b]_{1-3}$

$[f, x, y, b]_{1-3}$ pokud je v překážce - odeber

$[f, x, y, b]_{1-3}$ pokud je mimo vytyčený prostor popsaný v 4.2 - odeber

$H([f, x, y]_{1-3})$ pokud je logická 1 - odeber

$H([f, x, y]_{1-3})$ zapiš na pozici logickou 1

Pokud je $[f, x, y, b]_{1-3}$ v okruhu startovní pozice přejdi na *

Zbylé $[f, x, y, b]_{1-3}$ přidej do G(i+1) přidej i *j*-tou pozici (odkud se tam dostal)

Opakuj

Opakuj

* přepočítej G pomocí *j*

5. Návrh řízení

Navrhnout algoritmus, který umožní zacouvat s přívěsem na uživatelem libovolně zvolené místo, je hlavní účel této práce. Návrh trajektorie jsme probrali v minulé kapitole. Pokud se takový návrh zdaří, je možno přejít k samotnému řízení. Vhledem k tomu, že se jedná o nelineární nestabilní systém, bude potřeba návrh, který dokáže sledovat referenční trajektorii. Pro řízení se použijí tři druhy regulátorů PID, LQ a MPC. Ani jeden z těchto tří druhů řízení nevyužívá pro návrh nelineární systém.

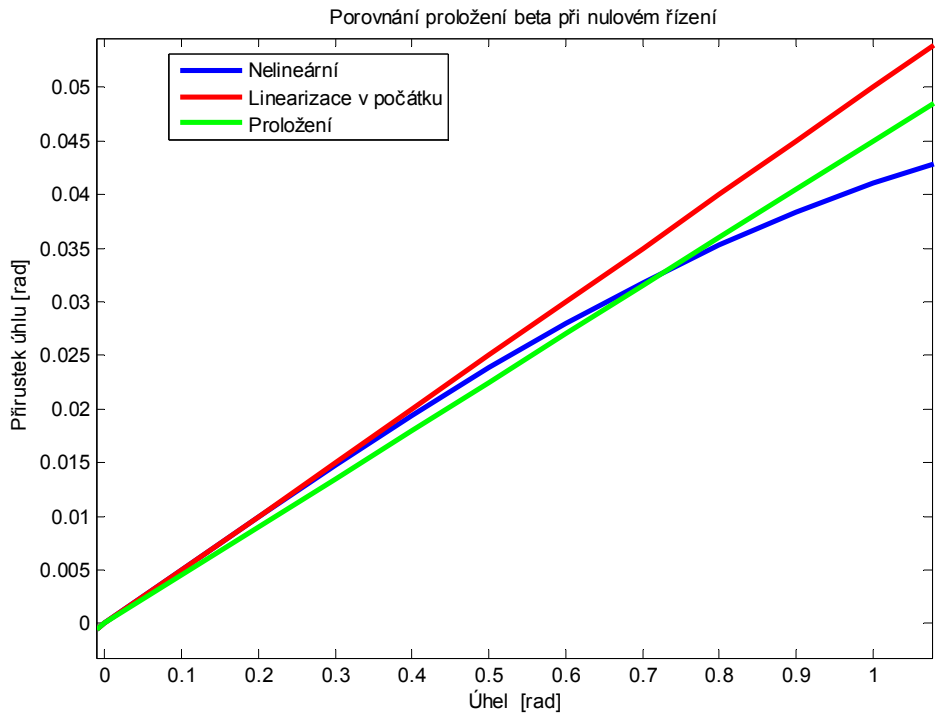
5.1 Lineární systém

Za prvé se musíme rozhodnout, kterou veličinu budeme řídit. Rovnice 2.19. uvádí čtyři rovnice. Pokud se rozhodneme pro řízení polohy vozíku, systém se ještě rozšíří o rovnice 2.8. Poloha vozíku se počítá pro 3D grafiku. Tím dostaneme čtyři stavové a dvě výstupní rovnice. Pokud si prohlédneme rovnice 2.19, konkrétně druhou a třetí pro určení polohy auta, zjistíme, že nemá žádný stabilní stav. φ se neustále mění v závislosti na řízení. To by ale znamenalo, že se musí provádět linearizace v každém kroku výpočtu. To komplikuje návrh regulátoru. Problém by se dal vyřešit použitím exaktní linearizace. Prohlédneme-li si rovnice 2.19, najdeme v každé stavové rovnici obsažené α, tedy řízení. A to je taky důvod, proč ji nemůžeme použít. Řízení koncové polohy je tedy velmi náročné.

Proložení prvotní trajektorie dynamickým programováním nám dává i jinou možnost řízení. Toto proložení nám zaručuje, že trajektorie odpovídá schopnostem řízení. Dává nám to možnost navrhnout řízení na základě úhlů. Tedy první a čtvrtá rovnice 2.19. Místo šesti rovnic dostaneme dvě. A navíc tyto rovnice mají rovnovážný stav. Pracovní bod můžeme přesunout do tohoto rovnovážného stavu a provést linearizaci. Úhel musí mít správnou velikost, protože se opakuje po 2π. O to se stará jednoduchá logika. Úhel systému je vždy v nejmenší vzdálenosti od reference.

Řízení musí být co nejrychlejší. Nedokážeme-li sledovat referenci co nejpřesněji, dopouštíme se velké chyby v poloze. Při návrhu lineárního systému nepoužívám linearizaci. Nevýhoda linearizace je, že v okolí pracovního bodu kopíruje systém. Takže v okolí je minimální chyba. Ale ve větší vzdálenosti se chyba zvětšuje. Názorně to je vidět na obrázku

5.1. Jde o čtvrtou rovnici 2.19. Při řízení se využívá hlavně okolí kolem velkých výchylek a právě okolí kolem nuly nemusí být tak přesné. V zatáčce při couvání nastává úhel nejčastěji kolem 0.7*rad*. Proložení je navržené tak, aby kolem tohoto bodu byla minimální odchylka. Na obrázku je vidět, že derivace v pracovním bodě nula, se při větších výchylkách vzdaluje od nelineárního systému.



Obrázek 5.1 Proložení nelineárního systému

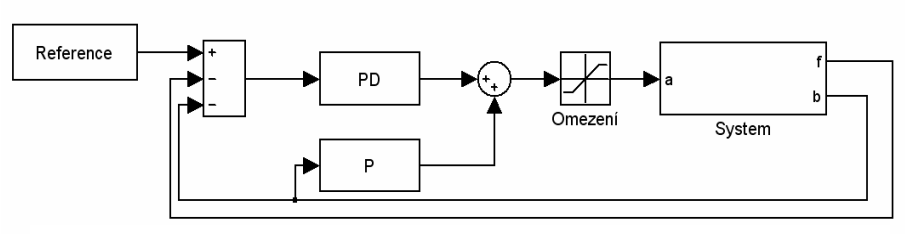
A právě v bodě 0.7*rad* je příliš daleko. Na obrázku je znázorněna modrou barvou původní nelineární křivka, při nulovém řízení $\alpha = 0$. Řízení posouvá pouze křivku vzhůru nebo dolů. Na změnu tvaru má minimální vliv. Právě zelená přímka prokládá kritický bod 0.7*rad*. Podobně se proloží i první rovnice 2.19. A vzniknou následující lineární rovnice:

$$\begin{aligned} \varphi(t+1) &= \varphi(t) - 0.0386 \cdot \alpha \\ \beta(t+1) &= 1.045 \cdot \beta(t) + 0.06 \cdot \alpha \end{aligned} \qquad (5.1)$$

Lineární rovnice, díky kterým můžeme navrhnout řízení, aniž bychom museli v každém bodě počítat linearizaci původního systému.

5.2 Řízení PID regulátorem

Obecný název PID regulátor je zažitý, v našem případě integrační složku nebudeme používat. Důvod je jasný když se podíváme na rovnice 5.1. První rovnice pro φ má ve spojitém případě všechny stavy nulové a závisí pouze na řízení. Takže jeden pól systému je nulový. A pro systém obsahující astatismus se integrační složka zpravidla nepoužívá. Takže pro návrh řízení použijeme jen konstanty PD. Při samotném řízení PD jsme schopni sledovat referenci. To ovšem díky omezení vstupu nemusí udržet stabilitu. Musíme přidat ještě jeden regulátor -v tomto případě stačí P-proporcionální. Schéma zapojení obou regulátorů je vidět na obrázku 5.2. Regulátor PD se stará o sledování reference. P regulátor má konstantní referenci 0. Má zajistit, aby úhel naklonění vozíku, byl co nejmenší. Při špatném nastavení ztratíme stabilitu když je P příliš malé. Pokud je P příliš velké, znemožní sledování reference. Návrh PD regulátoru pro sledování reference se provádí pomocí GMK (Geometrické Místo Kořenů).

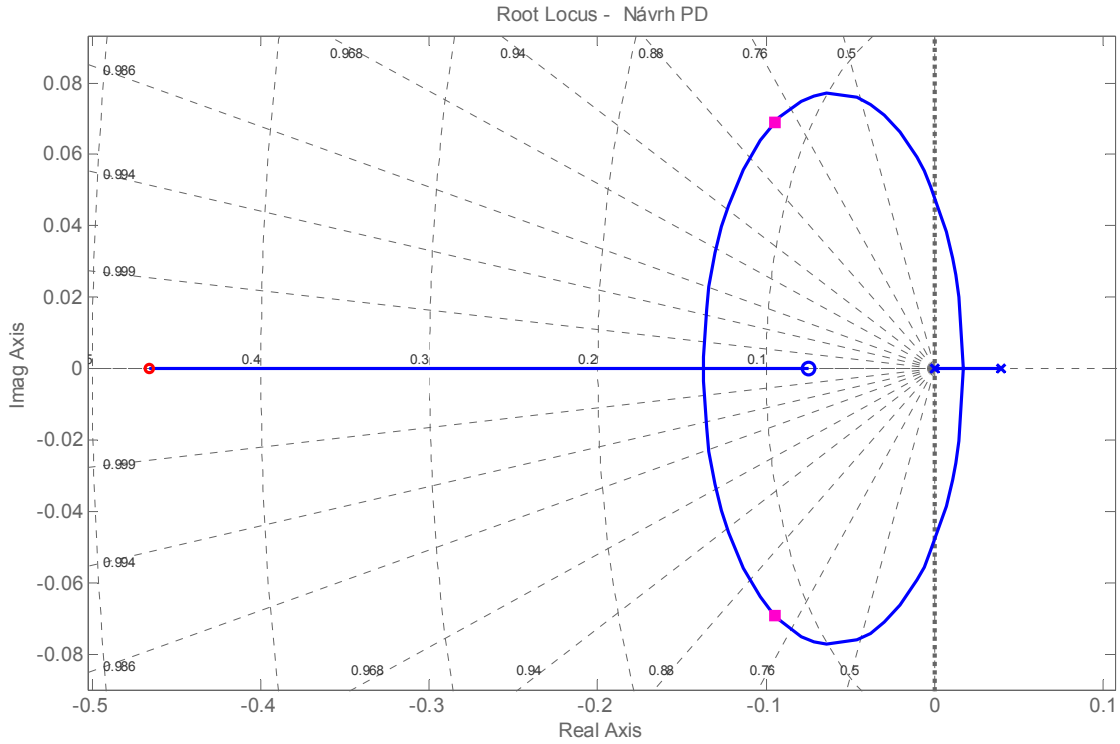


Obrázek 5.2 Schéma zapojení PD, P

$$LS(s) = \frac{0.0202 \cdot s + 0.0015}{s^2 - 0.0392 \cdot s} \qquad (5.2)$$

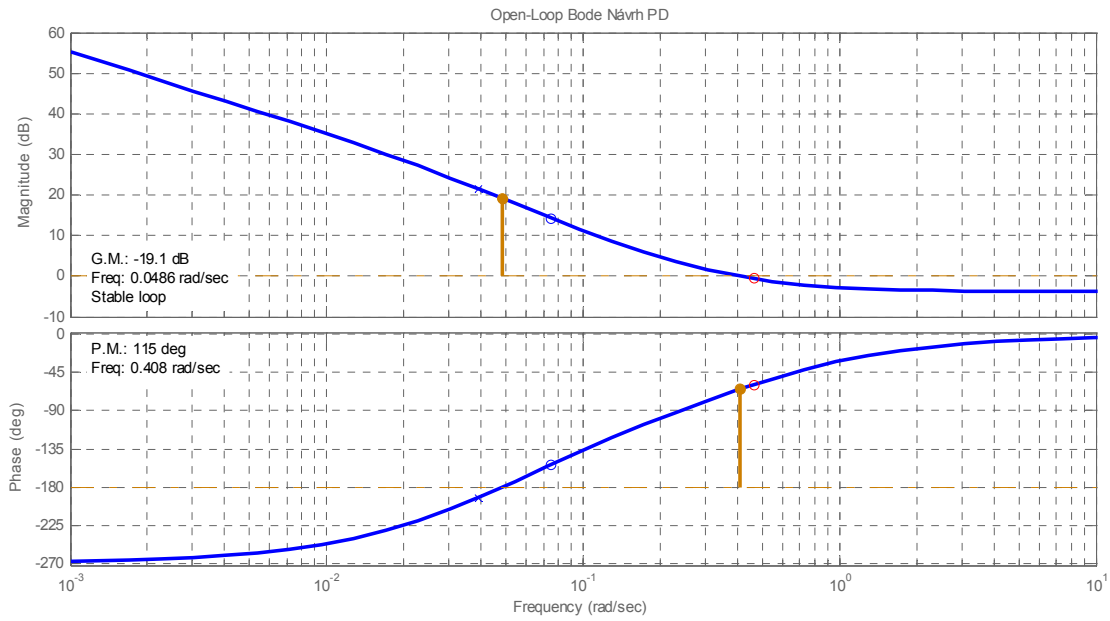
Rovnice 5.2 uvádí přenos systému ve spojitém případě. Pro diskrétní případ je vzorkování T=1. Návrh je prováděn v matlabu pomocí nástroje sisotool. Bližší informace o GMK, nebo taky nazývané Root – Locus nalezneme v [5].

Na obrázku 5.3 je vidět návrh PD. Z přenosu systému vidíme, že máme dva póly a jednu nulu. Póly se značí modrým křížkem, nuly modrým kolečkem. Jeden pól odpovídající úhlu natočení auta φ je v nule, to je na mezi stability. Druhý pól je v pravé polorovině komplexní roviny. Takže je nestabilní. To taky znamená že řízení je nezbytné. Nula systému je stabilní a usnadňuje řízení. Systém je druhého řádu, pro sledování reference by stačila i samotná proporcionální složka. Derivační složku přidáváme kvůli tlumení kmitání. Nula regulátoru je umístěna na obrázku 5.3 červeným kolečkem. Výsledné zesílení znázorňují fialové čtverečky. Protože jsou v levé polorovině je uzavřená smyčka stabilní.



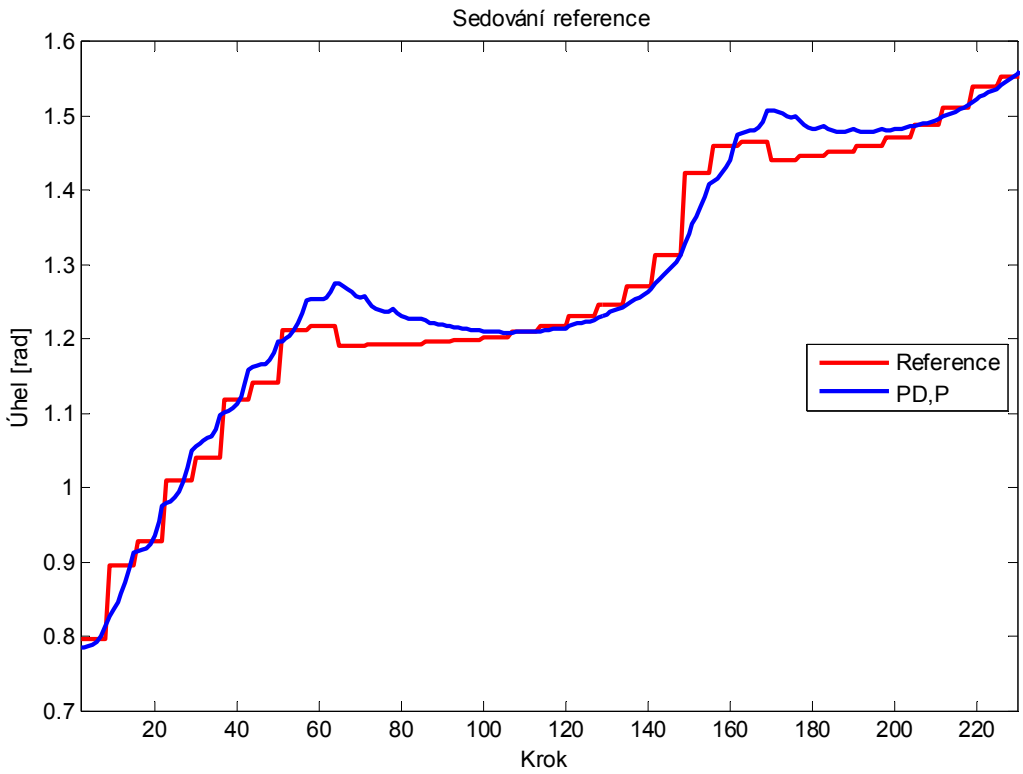
Obrázek 5.3 Návrh PD

Problém s tímto typem regulátoru je, že nepočítá s možnými omezeními. V našem případě s omezením akčního zásahu α a omezením stavu β. Proto je taky na obrázku 5.2 před vstupem do systému omezení, které ořízne akční zásah regulátorů. Dal by se navrhnout jistě lepší „rychlejší“ regulátor, ale právě kvůli omezení je navržený ten nejlepší. Fázová i frekvenční charakteristika návrhu na obrázku 5.4 s fázovou i amplitudovou bezpečností. Obrázek návrhu samotného P regulátoru pro stabilitu úhlu β neuvádím. Je to návrh pro systém prvního řádu s nestabilním pólem. Stačí najít jen vhodné zesílení a uzavřená smyčka



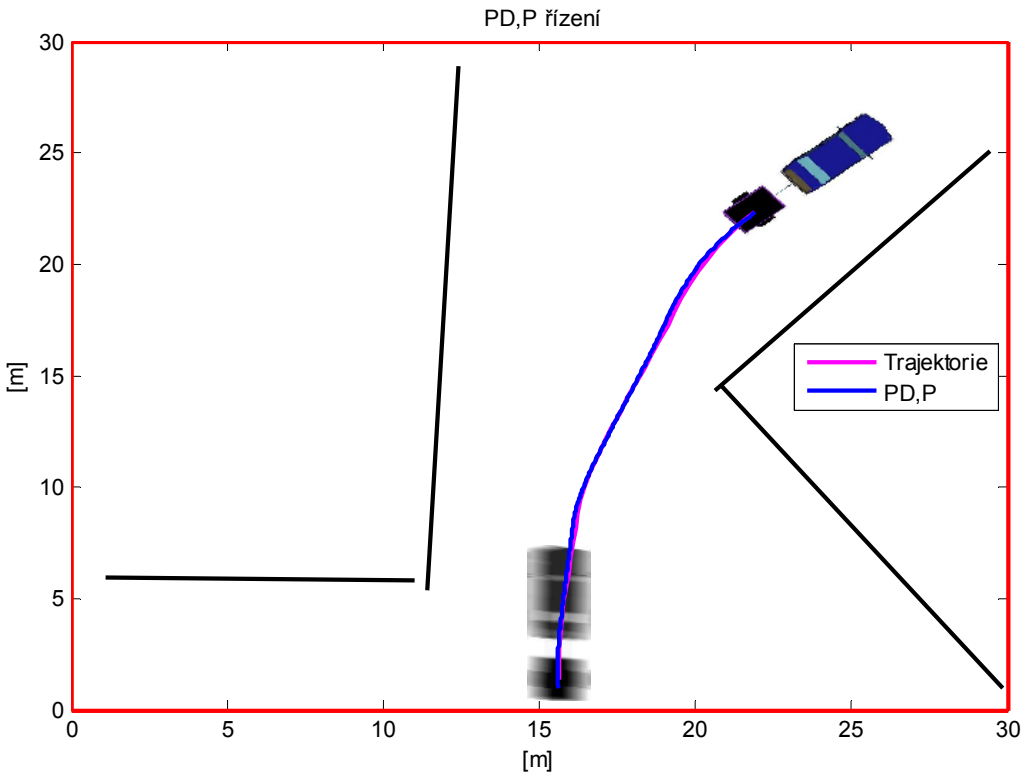
Obrázek 5.4 Frekvenční a fázová charakteristika návrhu PD

bude stabilní. Zesílení musí odpovídat přibližnému návrhu PD. Na obrázku 5.5 a 5.6 je vidět



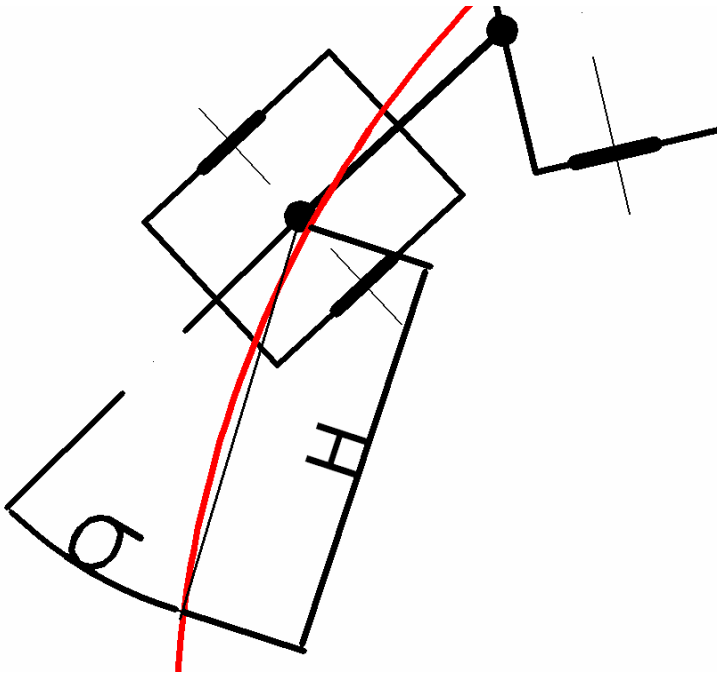
Obrázek 5.5 Sledování reference PD,P

funkčnost řízení PD,P regulátoru. Na obrázku 5.5 to vypadá, že se regulátor přibližuje k referenci jen velmi pomalu. Na obrázku 5.6, který je totožný, tento jev nenastává. Na prvním obrázku je vidět změnu úhlu, na druhém pohyb $[xv, yv]$. Reference pro PD,P regulátor se mění v závislosti na poloze vozíku. Díky dynamickému programování jsme dostali ideální trajektorii. Stačilo by vzít z ideální trajektorie úhel natočení vozíku β, přičíst k němu úhel natočení auta φ a máme předem známou trajektorii. Pro tento případ regulátoru to není možno použít. V zatáčkách dochází k pozdějšímu přiblížení k referenčnímu úhlu, to má za následek že poloha vozíku $[xv, yv]$ je v cíli velmi vzdálená od požadované. Taková reference není vhodná, požadujeme-li správnou cílovou polohu auta a vozíku.



Obrázek 5.6 Sledování trajektorie PD,P

Aby nedocházelo k přílišnému vzdalování od reference, musíme ji přizpůsobovat pohybu auta a poloze přívěsu.



Obrázek 5.7 Úhel odchylky od trajektorie

Obrázek 5.7 ukazuje přibližování k referenční trajektorii. Pro řízení se reference skládá ze tří částí, dvě jsou určené při výpočtu dynamického proložení. Jsou tedy po celou dobu výpočtu neměnné. Při ideálním sledování není potřeba proměnlivá část. Třetí složka reference se mění právě na vzájemné poloze navržené trajektorie a momentální poloze a natočení vozíku. Vzdálenost *H* nezbytná pro výpočet úhlu σ, se mění jen minimálně. K zvětšení vzdálenosti *H* dochází, jen když se vozík příliš vzdálí od navržené trajektorie. Minimální vzdálenost *H* je jeden metr. Výpočet úhlu σ je složitější. V bodě dotyku úsečky H s navrženou trajektorií se vypočte tečna. Úhel který tečna svírá nazveme ξ. Dále zjistíme skutečný úhel natočení přívěsu. Úhel natočení β je poloha vzhledem k autu. Proto k úhlu β připočte natočení auta φ a dostaneme skutečné natočení vozíku v prostoru.

$$\sigma = \xi - (\varphi + \beta)$$

Výpočet úhlu σ máme. Pro nás je důležité hlavně jeho znaménko, které nám určí kam se bude řídit. První dvě složky vzorce 5.3 se počítají daleko snadněji. První je úhel navržené trajektorie a druhý je jeho derivace. Z navržené trajektorie zjistíme úhel ϕ a z derivace zjistíme změnu úhlu $\Delta\phi(t)$.

$$w(t) = \phi(t) - n \cdot \Delta\phi(t) + m \cdot \sigma(t) \qquad (5.3)$$

Změna úhlu $\Delta\phi(t)$ se přidává pro lepší korekci v zatáčkách. Při výpočtu se ještě tato změna úhlu přenásobí váhou *n*. Ta nám vlastně udává jak moc je nutná. A váha *m* nám říká jak moc se můžeme vzdálit od trajektorie

5.3 Řízení LQ regulátorem

Návrh LQ řízení je daleko snazší než tomu bylo u PID. Nemusíme se zabývat stabilitou, určitá robustnost je zaručena. Problém nastává s omezením, se kterým návrh nepočítá. Při návrhu nám stačí odladit váhové matice Q, R.

Postup návrhu stručně popíši, podrobnější popis najdeme v [1]. Pomocí následující rovnice

$$u = -K(t) \cdot x(t) \qquad (5.4)$$

se bude vypočítávat řízení. Stačí přenásobit stavy systému záporným Kalmanovým zesílením a dostaneme řízení. V našem případě známe všechny stavy a nemusíme je odhadovat. Díky tomu se návrh ulehčí. Při návrhu se bude jednat o takzvaný problém „servomechanismu“. Jedná se o minimalizaci kritéria:

$$J = \frac{1}{2} x^T(N) Q' x(N) + \frac{1}{2} \sum_{t=0}^{N-1} \left\{ e^T(t) Q' e(t) + u^T(t) R' u(t) \right\}$$

Kde e je regulační odchylka a vypočteme ji:

$$e = r(t) - y(t) = C_2 x_2(t) - C_1 x_1(t) - D_1 u(t)$$

V našem případě přímá vazba mezi vstupem a výstupem neexistuje, takže D je nulové. Pro výpočet se stavová matice a vstupní musí upravit podle:

$$x(t+1) = \begin{bmatrix} A_1 & 0 \\ 0 & A_2 \end{bmatrix} x(t) + \begin{bmatrix} B_1 \\ 0 \end{bmatrix} u(t)$$

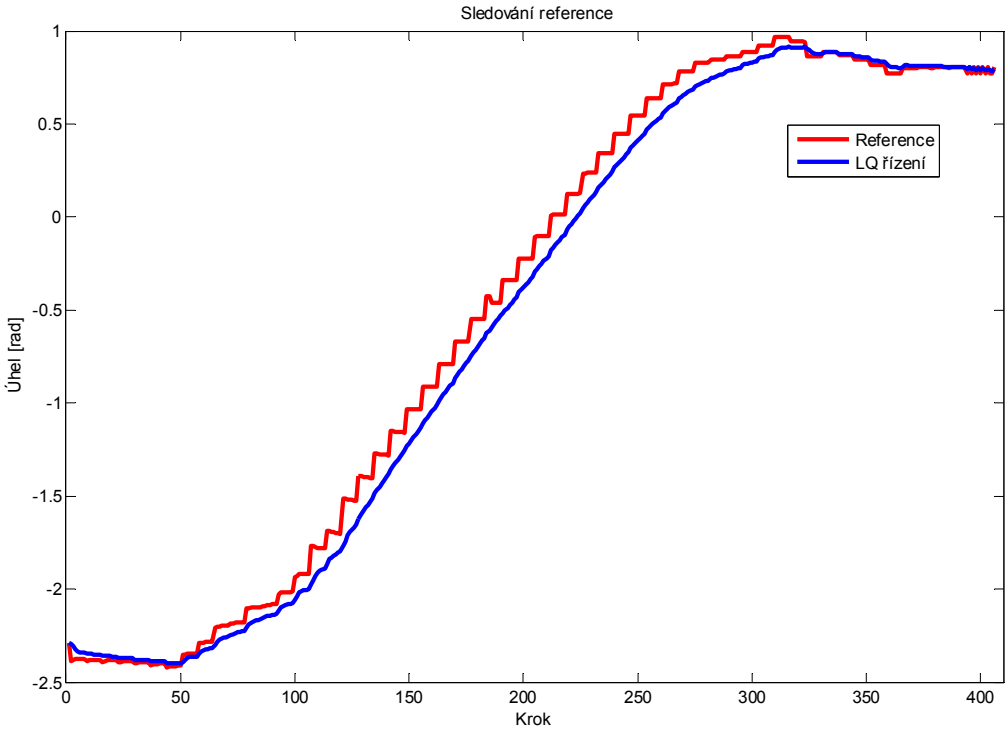
Matice A_2 je v našem případě jednotková. Ještě musíme upravit váhové matice:

$$Q = \begin{bmatrix} C_1^T Q' C_1 & -C_1^T Q' C_2 \\ -C_2^T Q' C_1 & C_2^T Q' C_2 \end{bmatrix}$$

Matice R se upravovat nemusí, D je nulové. Nyní stačí dosadit do Riccatiho rovnice:

$$K(t) = \left(R + B^T P(t+1) B\right)^{-1} \left(S^T + B^T P(t+1) A\right)$$
$$P(t) = \left(A - BK(t)\right)^T P(t+1) \left(A - BK(t)\right) + K^T(t) R K(t) + Q$$

V našem případě S je nulové. Výpočet Riccatiho rovnice opakujeme dokud se neustálí. Výsledné Kalmanovo zesílení použijeme pro řízení, dosadíme do rovnice 5.4. Pokud není

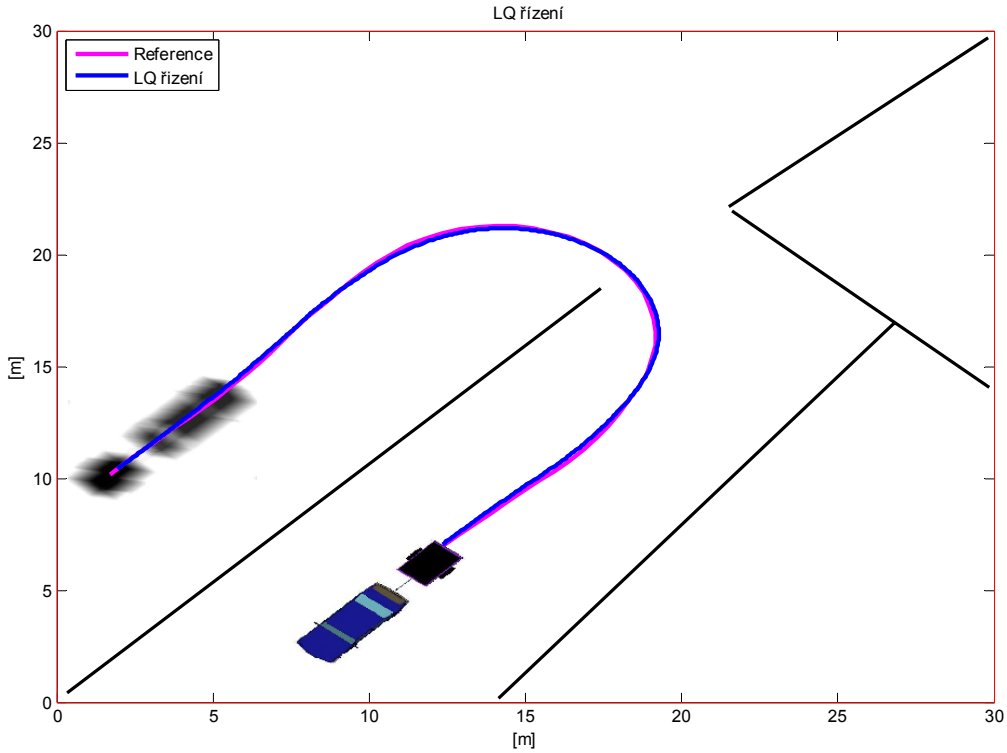


Obrázek 5.7 Sledování reference LQ

akční zásah v požadovaném rozsahu, dojde k jeho omezení na maximální možnou hranici. Řízení musí být v rozsahu $u \in \left\langle \pi/6, -\pi/6 \right\rangle$. Na obrázku 5.7 a 5.8 vidíme chování navrženého regulátoru. Vliv posledního členu rovnice 5.3 je patrný ke konci řízení, kolem kroku 400 na obrázku 5.7. Váhové matice pro návrh LQ regulátoru jsou:

$$Q' = \begin{bmatrix} 1000 & 0 \\ 0 & 1 \end{bmatrix}$$
$$R' = [0.1]$$

První stav matice Q' odpovídá sledování reference podle vzorce 5.3. Druhý stav zaručuje stabilitu vozíku. Neboli řídí výchylku vozíku β do nuly. Na obrázku 5.8 vidíme jak přesné je sledování navržené trajektorie. I přes to, že řídíme úhel natočení, můžeme pomocí přidané logiky dosáhnout velkou přesnost polohy. Požadovaná poloha je přesně splněna. Odchylka se počítá v jednotkách centimetrů. Samozřejmě ne vždy se to povede s tak malou chybou. O přesnosti se zmíním v kapitole o porovnání kvality řízení.



Obrázek 5.8 Sledování trajektorie LQ

Na obrázku 5.8 je fialová reference zcela překryta. Řízení LQ regulátoru ukazuje modrá čára. Dostatečně dlouhá a proměnlivá reference umožňuje zjistit, zda námi navržený regulátor je stabilní. A je schopný sledovat referenci.

5.4 Řízení MPC regulátorem

Třetí návrh řízení, který použijeme pro sledování reference je tak zvaný MPC. Umožňuje zahrnout omezení a počítá s predikcí. Nevýhoda spočívá v jeho složitosti výpočtu, a proto není vhodný pro reálný čas. Pro náš případ simulace ho můžeme použít. Nejdřív trochu teorie, kterou podrobněji naleznete v [6,7] a v [8].

Lineární případ popisuje první a druhá uvedená literatura. Způsob nelineárního řízení popisuje třetí uvedená literatura. Právě tento způsob řízení by byl nevhodnější. Můžeme zahrnout nejen omezení vstupu a úhlu natočení vozíku, ale hlavně omezení překážkami. Při implementaci tohoto řízení, se projevila výpočetní náročnost této metody. Časy vypočtu s použitím klouzavého horizontu a pěti optimalizací popsané v [8], se nepohybovaly pod 10 minut. Při couvání okolo 5metrů. Po odstranění překážek z výpočtu se časy výrazně zkrátily. Pokud nezahrneme do výpočtu překážky, pak je daleko snažší používat řízení úhlu, jako v předešlých dvou případech. Návrh řízení bude z upravených stavových rovnic 5.1. Při návrhu se bude minimalizovat kritérium:

$$J = \sum_{t=0}^{T_p-1} \left\{ q(t)[y(t) - w(t)]^2 + r(t)[u(t)]^2 \right\}$$

Kde T_p je horizont predikce, neboli kolik vzorků dopředu vidí. Váhy $q(t)$ a $r(t)$ udávají kvalitu sledování. Výpočet T_p kroků dopředu:

$$y_t = Vx(t) + Su_t$$

Je potřeba ještě uvést že matice V a S vypadají takto

$$V = \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{T_p-1} \end{bmatrix} \qquad S = \begin{bmatrix} D & & & \\ CB & D & & \\ \vdots & & \ddots & \\ CA^{T_p-2}B & \cdots & CB & D \end{bmatrix}$$

Vstupní a výstupní vektory mají tento tvar

$$u_t = \begin{bmatrix} u(0) \\ u(1) \\ \vdots \\ u(T_p - 1) \end{bmatrix} \qquad y_t = \begin{bmatrix} y(0) \\ y(1) \\ \vdots \\ y(T_p - 1) \end{bmatrix}$$

Vývoj stavu na horizontu T_p popisuje předchozí rovnice. Pokud nepoužijeme omezení, můžeme psát přímo rovnici pro výpočet řízení. Při použití omezení nejdou napsat jednoduché rovnice pro výpočet. Pro tento případ se používá kvadratické programování. Jak funguje kvadratické programování najdeme [2]. Kvadratické programování řeší minimalizační úlohu v tomto tvaru:

$$\min_u \left(\frac{1}{2} u^T H u + f^T u \right)$$

V matlabu je primárně nadefinovaná funkce „quadprog“, která řeší úlohu kvadratického programování. Zbývá dodat jak dostaneme matici H a vektor f

$$H = S^T Q S + R$$
$$f^T = (V x_t - w_t)^T Q S$$

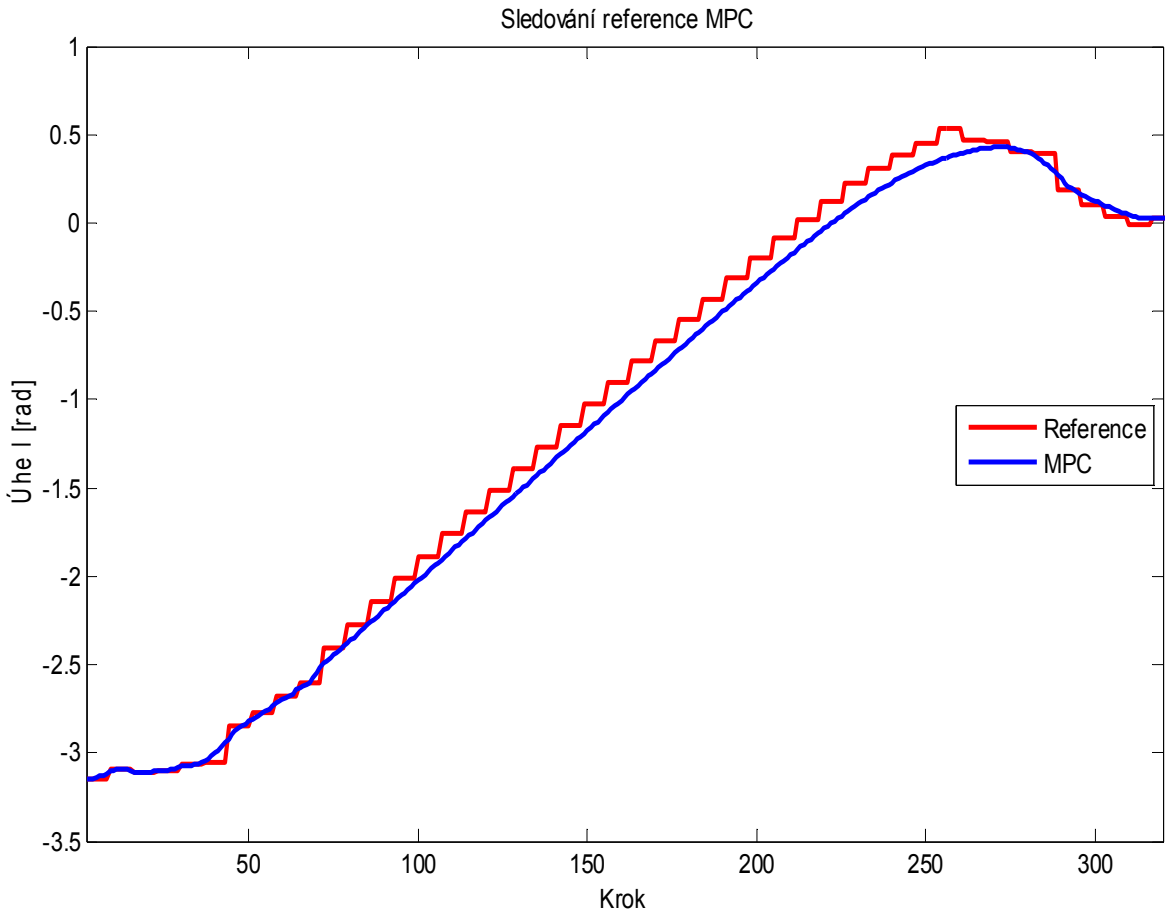
Funkce kvadratické programování má víc vstupů. Ty odpovídají omezujícím kriteriím.
V našem případ se spokojíme s těmito:

$$-\pi/6 \le u(t) \le \pi/6$$
$$-1 \le \beta(t) \le 1$$

Výpočet provádíme s takzvaným klouzavým horizontem. Tzn. že vezmeme pouze první krok z vypočtené řídící posloupnosti a přivedeme ho na vstup systému. To opakujeme dokud nedorazíme k cílovému stavu. Teorii máme za sebou, jak to funguje v našem případě vidíme na obrázku 5.9 a 5.10. Pro výpočet jsou váhy kvality řízení nastaveny na

$$Q = 50$$
$$R = 1$$

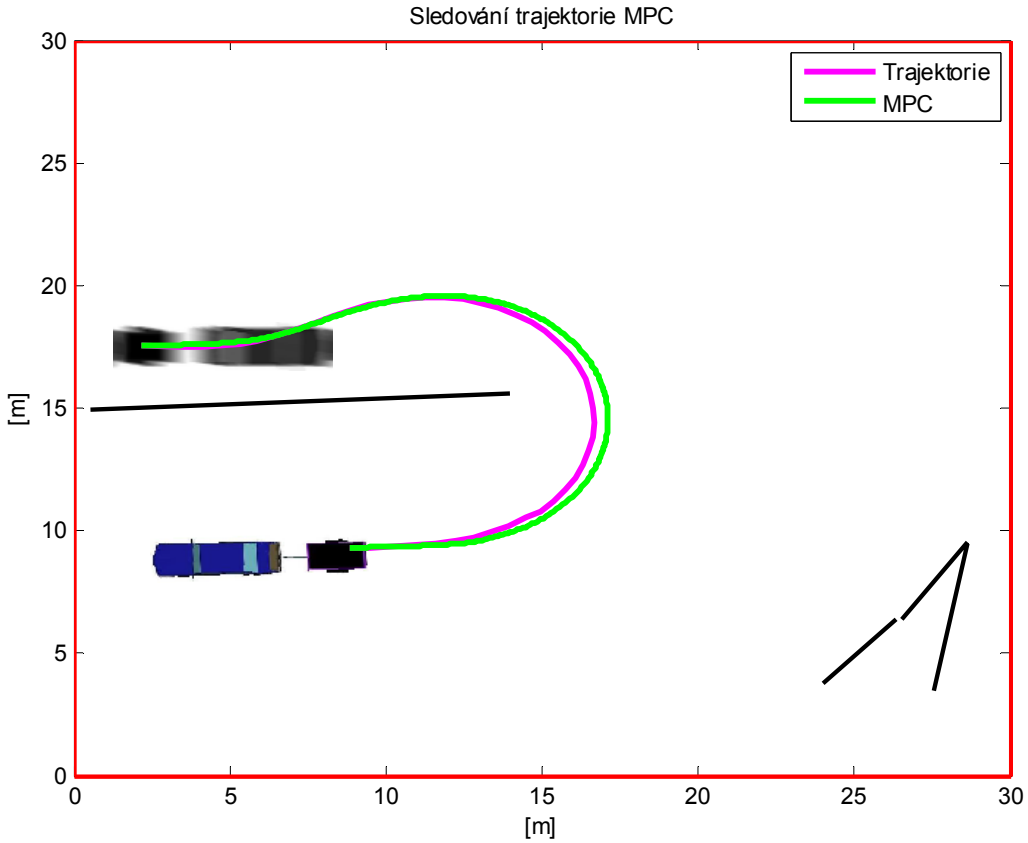
Snažíme se, aby regulační odchylka byla minimální. Jako ostatně u všech navržených regulátorů. Jakou energii vynaložíme na řízení je podřadné, protože máme horní hranici omezenou. Je jedno jestli se kola natočí málo, nebo na maximální výchylku, pokud jsou řízeny automaticky.



Obrázek 5.9 Sledování reference MPC

V tomto případě můžeme vidět na obrázku 5.9 prediktivní schopnost. V oblasti 40-ti kroků nesleduje příliš striktně referenci, protože díky predikci ví, že přijde prudká změna úhlu a připraví se na to. To samé je vidět kolem 280-tého kroku výpočtu.

Každý krok je vlastně pohyb auta o 0.1m směrem vzad. O kolik se posune vozík je proměnlivé a záleží na kvalitě regulace. Reference vypočtená podle rovnice 5.3.



Obrázek 5.10 Sledování trajektorie MPC

Obrázek 5.10 ukazuje, jak je možné i velké úhly uřídit, a přitom neztratit stabilitu. Zdá se, že trajektorii nekopíruje tak dokonale, jako v předešlém případě LQ. Ale poloměr oblouku, který řídí je daleko menší, než v předchozím případě.

V další kapitole porovnávám všechny navržené regulátory.

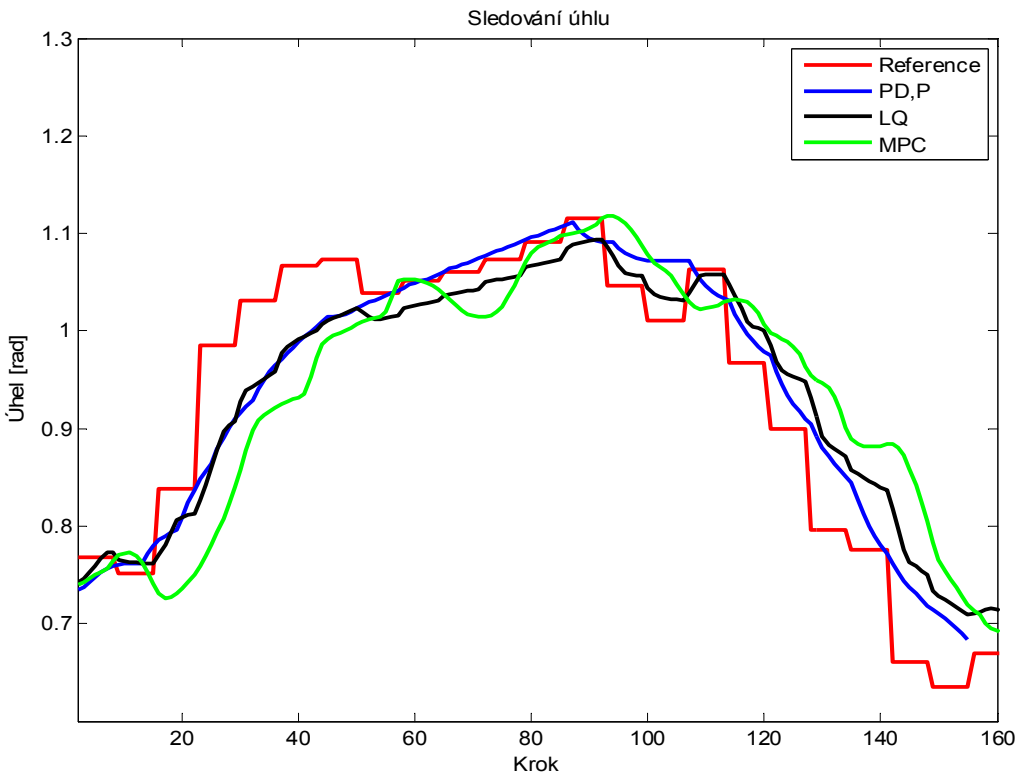
5.5 Porovnání regulátorů

Na porovnání kvality regulace, jsem připravil dva typické problémy.

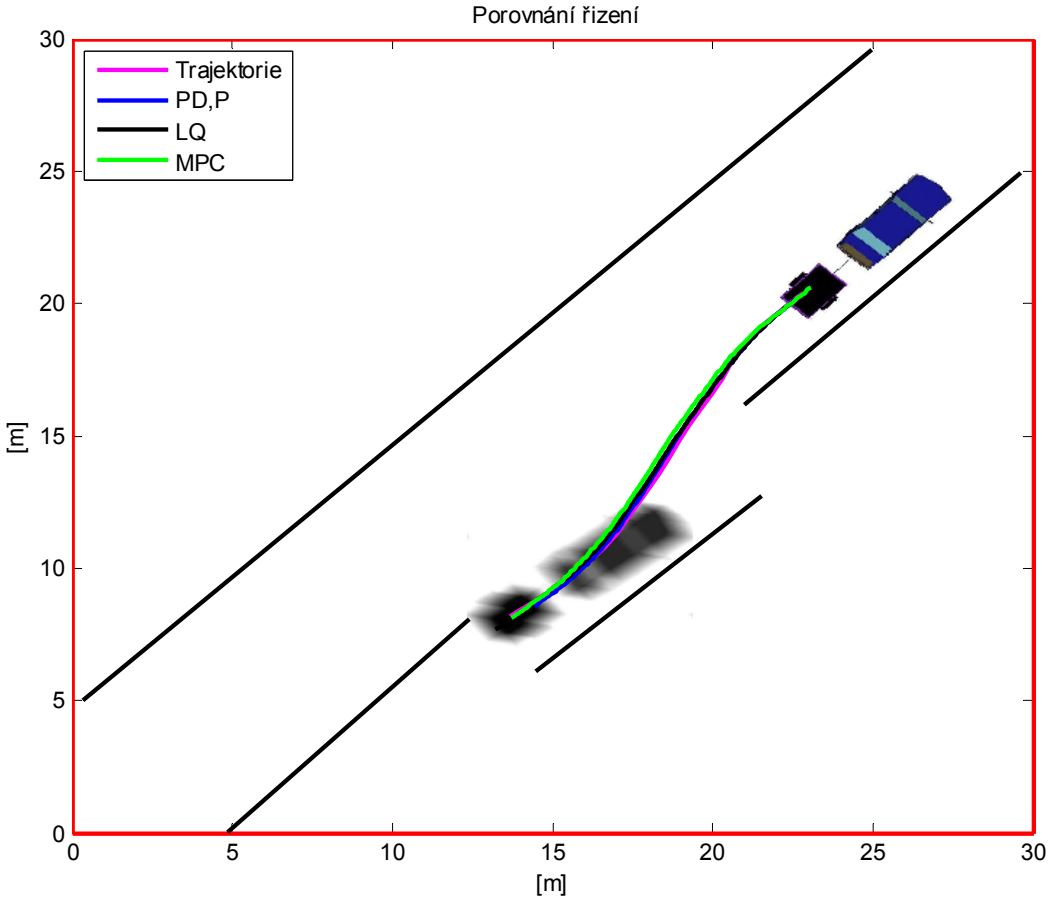
První se týká zasouvání mezi auta. Je to vlastně jen změna pruhu.

Při druhém couvání se změní úhel o $\pi/2$. Jedná se o klasické couvání do garáže či na vymezené parkoviště.

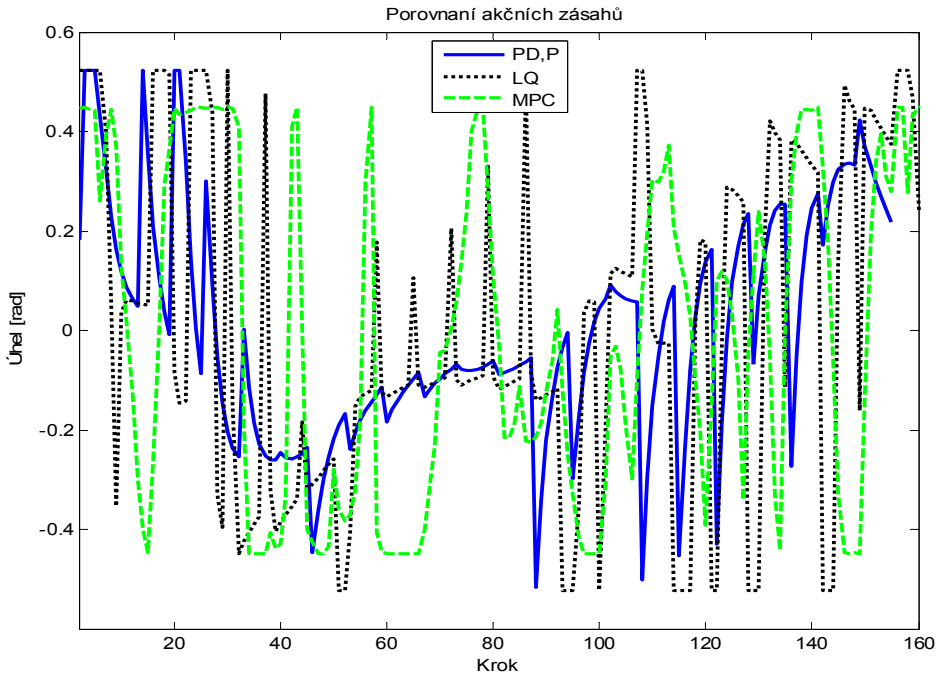
Všechny regulátory počítají řízení podle úhlu. Reference se vypočítá pomocí vzorce 5.3. Na obrázku 5.11a můžeme vidět právě sledování této reference pomocí všech tří navržených regulátorů. Jedná se o pohyb zacouvání mezi auta. Návrh reference se provádí s maximální změnou úhlu $\alpha = \pm\pi/7$. Tzn. že minimální poloměr zatáčky bude 5.7m pro auto. Proto ne všechny návrhy, které uživatel zadá do mapy, naleznou referenční trajektorii. Reference na obrázku 5.11a a 5.12a je vykreslena podle vzorce 5.3 bez třetího členu. Jedná se o zjednodušení, jinak by musely být vykresleny tři reference pro každý regulátor zvlášť, a to by znamenalo velmi nepřehlednou situaci. Takže na obrázku 5.11a a5.12a má reference pouze orientační charakter. Jde spíše o porovnání dalších tří regulátorů.



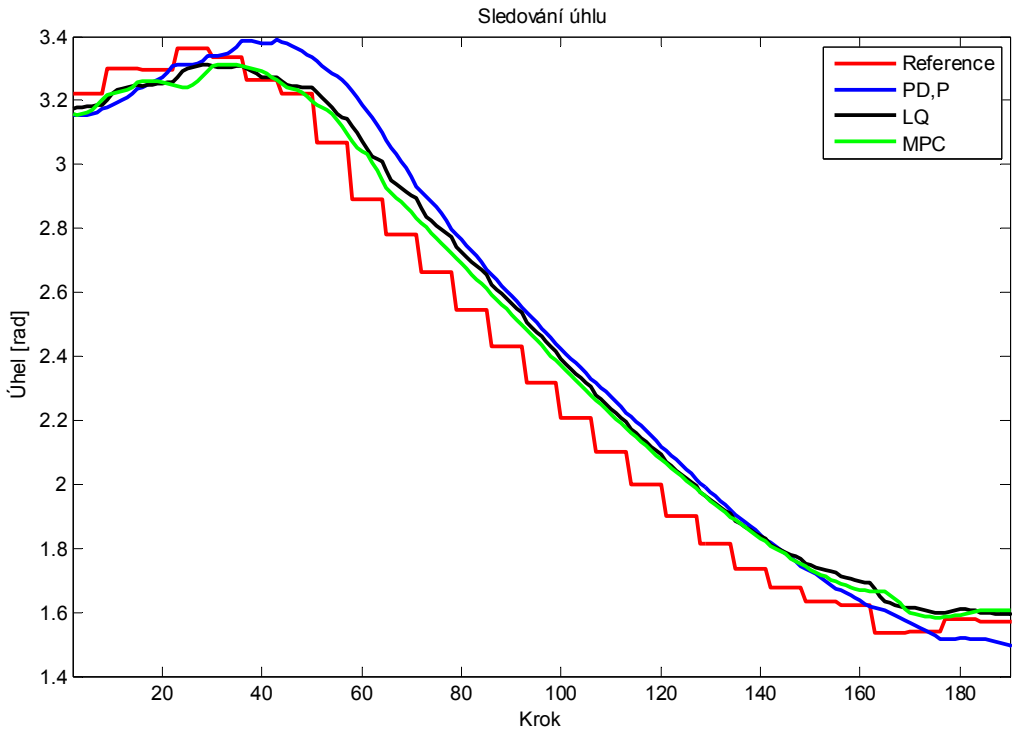
Obrázek 5.11a Sledování úhlu



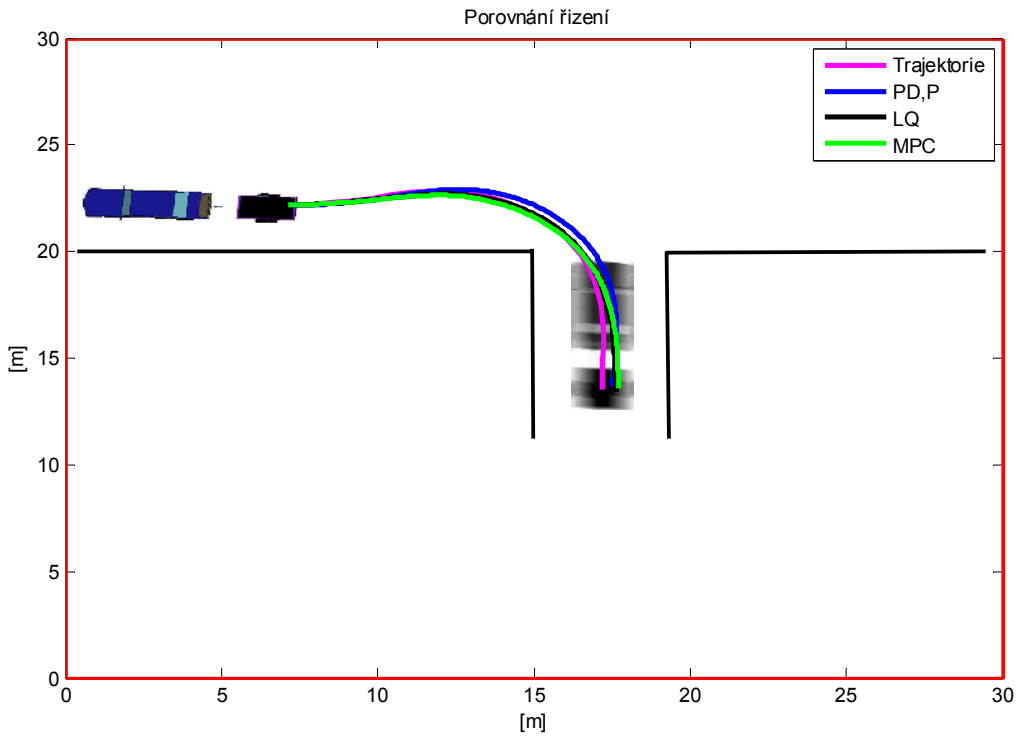
Obrázek 5.11b Porovnání řízení



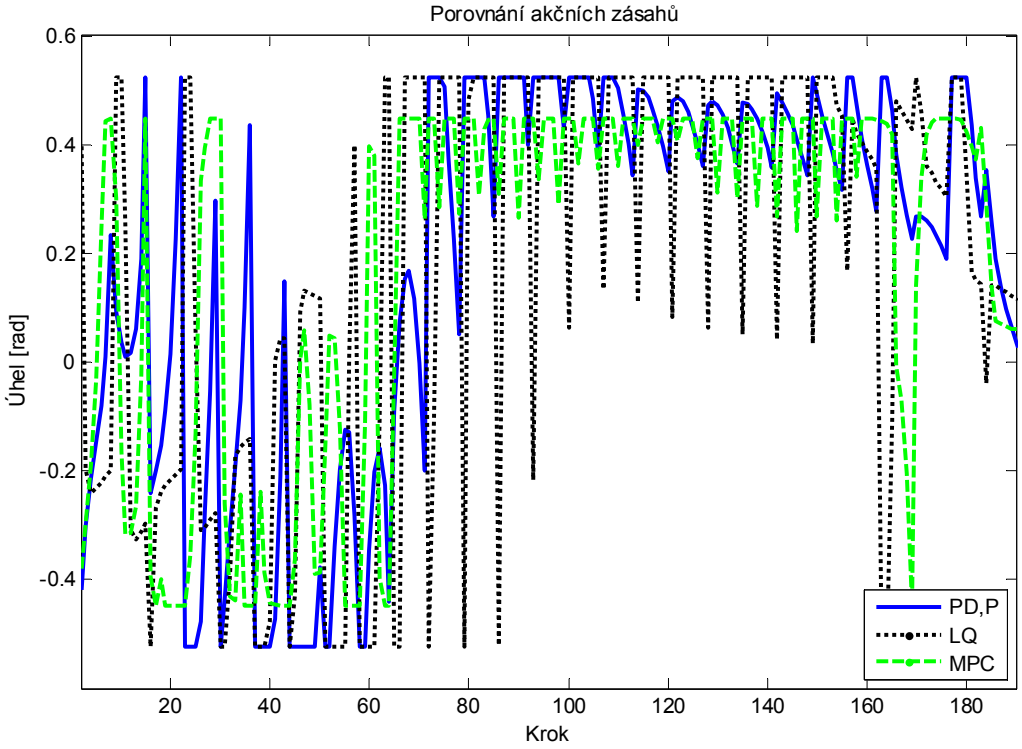
Obrázek 5.11c Akční zásahy



Obrázek 5.12a Sledování úhlu



Obrázek 5.12b Porovnání řízení

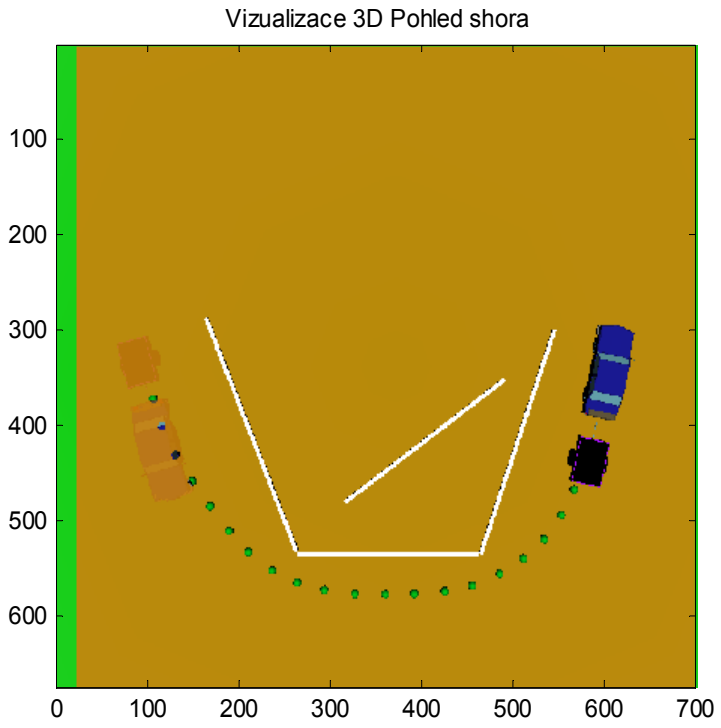


Obrázek 5.12c Akční zásahy

Na obrázku 5.11b a 5.12b vidíme pohyb vozíku. Na prvním obrázku se při regulaci příliš nevzdálil od polohy trajektorie. Na druhém obrázku je situace složitější. Nejmenší možný poloměr zatáčky je pro nestabilní systém náročné sledovat, aby neztratil stabilitu. Proto je odchylka v koncovém bodě kolem 0.6m. V prvním případě se jednalo o odchylku 0.3m. Odchylka úhlu je za to minimální. Velká odchylka polohy se dá minimalizovat pokud se po každých 5m pohybu přepočítá dynamickým programováním nová trajektorie řízení. Potom se pohybujeme v odchylkách do 0.1m. Tato hodnota je už přijatelná. Při simulaci se tato metoda nepoužívá - prodlužuje výpočet. Podíváme-li se na obrázky 5.11c a 5.12c vidíme změny řízení jednotlivých regulátorů. Nejedná se o žádné jednoduché průběhy. Je to způsobeno návrhem trajektorie. Kde dynamické programování používá pouze tři stavů řízení. Vzhledem k tomu, že dva stavy jsou maximální možná výchylka, řízení logicky musí vypadat stejně.

6. Vizualizace 3D

Pro vizualizace používáme matlabovký toolbox virtual reality. Ten umožňuje práci s 3D objekty přímo z matlabu. Nakreslíme jednoduché objekty. Složitější objekty, jako v tomto případě, není vhodné kreslit v matlabu. Lepší je použít nějaký program, který je přímo určen



Obrázek 6.1 Pohled shora

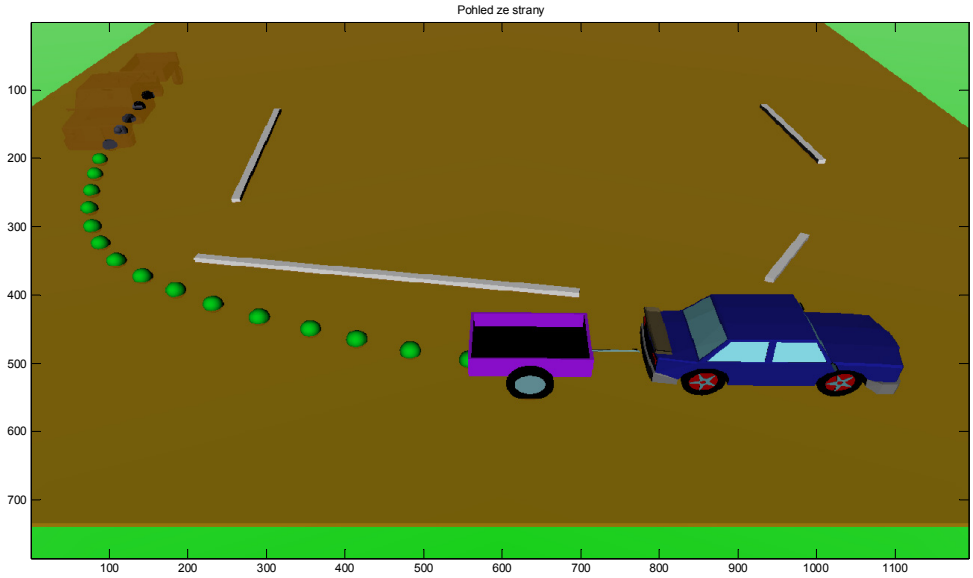
na kreslení 3D objektů. Po té stačí už jen konvertovat do formátu WRL. Když máme nakreslené objekty, vykreslíme je. To se provádí tak, že umísťujeme polohu středu objektu do prostoru. Určíme jeho úhel a případné zvětšení, barvu nebo další možné atributy. Díky neustálému opakování získáme pohyb objektů.

Důležité je správné umístění kamery. To je místo, odkud se na projekci díváme. Je nutné mít i osvětlení kompozice. Na obrázku 6.1 je pohled na mapu shora. Je to stejný pohled jako na obrázcích 5.12b či 5.11b nebo dalších v této práci uvedené. Pohled shora je nejpřehlednější. Barvy aut jsou stejné jako při zadávání. Podložka parkoviště je tmavě zlatou barvou. Zelený pruh který můžeme vidět vlevo na obrázku je pozadí v barvě trávy. Pokud pohled otočíme uvidíme oblohu se sluncem. To je pro nás i zdroj světla na kompozici. Také proto nevidíme žádné stíny. Překážky mají bílou barvu. Oproti 2D kde se používá úsečka, dostaly překážky šířku i tloušťku. Šířka překážky je 0.1m.

Vizuální kolize nenastane, protože do výpočtu je tato šířka překážky započítána. Kolize se počítá v matlabu, takže se vykreslování zastaví před kolizí. Nejsou nastavené senzory na kolizi objektu či případnou deformaci.

Na obrázku 6.1 vidíme zelené tečky. To je ukazatel trasy řízení. Při vykreslování couvání se auto pohybuje a přitom se umísťují právě tyto zelené tečky v dráze, po které se vozík pohybuje. Těchto teček je dvacet a jsou rovnoměrně rozmístěny po celé dráze couvání. Tečky jsou vykresleny také z důvodu srovnání ručního a automatického řízení. Pomocí nich může uživatel zacouvat do cílové pozice, stačí je jen sledovat.

Předdefinované kamery jsou celkem tři. Pohled první z nich je na obrázku 6.1. Na obrázku 6.2 je pohled z boku. A na obrázku 6.3 je pohled třetí kamery ze předu auta.



Obrázek 6.2 Pohled ze strany

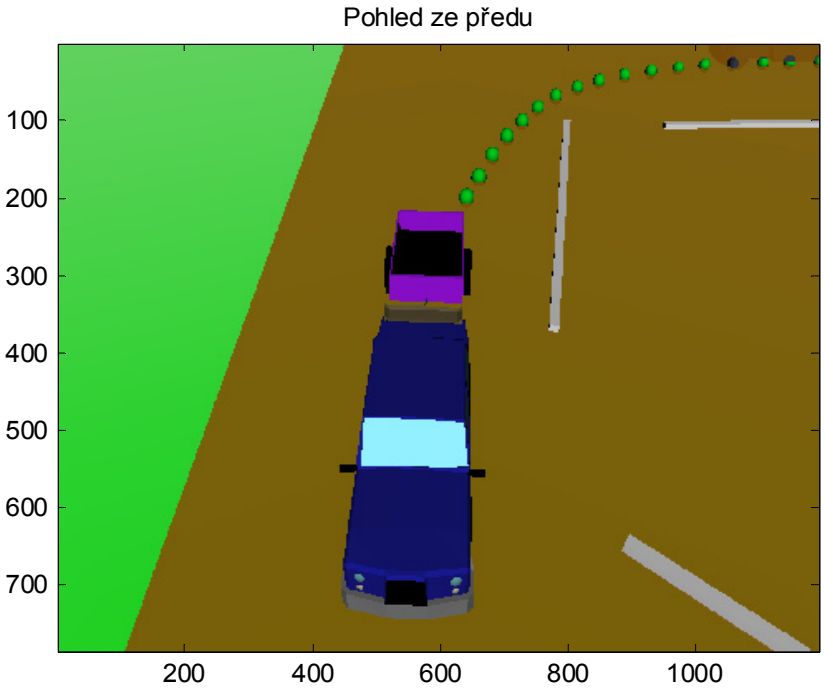
Při pohledu ze strany vidíme lépe 3D objekty. Pohyb dotváří otáčení kol auta při pohybu. Díky diskům ve tvaru hvězdice můžeme vidět i směr pohybu. Přední kola auta se vytáčí na základě akčního zásahu regulátoru. Obrázek 6.2 má i větší rozlišení, které je způsobeno přiblížením objektu.

Pohyb z boku sleduje statická kamera stejně jako pohyb shora. Kamery je možno natočit podle vlastní potřeby. Poloha je pouze předefinovaná pro zjednodušení.

Třetí pohled je dynamický a závisí na poloze a úhlu natočení auta. Tato kamera se nejlépe hodí pro ruční ovládání.

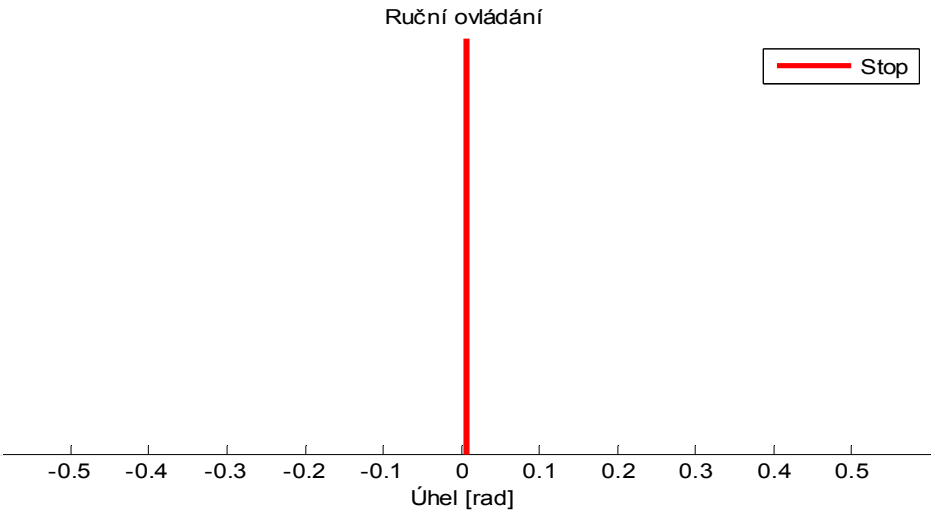
6.1 Ruční ovládání

Uživatel má možnost při ručním ovládání zjistit jestli je schopen zacouvat na danou polohu. Tuto možnost může využít i pokud se nepovede návrh trajektorie. Má možnost si nastavit pohled, který mu nejlépe vyhovuje. Nejlepší je třetí kamera která se natáčí s aktuálním



Obrázek 6.3 Pohled ze předu

natočením auta. Při ručním ovládání se nastavuje pozice pomocí myši. Na obrázku 6.4 vidíme ukazatel natočení předních kol. Červená linka ukazuje momentální nastavení. Rychlost pohybu je konstantní. Po předvedení jak umí zacouvat regulátor se objeví obrázek 6.4. Pro pohyb je nutné držet zmáčknuté tlačítko myši. Stačí kliknout na červenou linku a natáčet kola auta. Popis na ose x je přímo úhel natočení v radiánech.



Obrázek 6.4 Ruční ovládání

7. Závěr

V této práci jsem se zabýval couváním přívěsu. Z odvozených rovnic v druhé kapitole jsem zjistil, že se jedná o nestabilní, nelineární systém. Tedy takový systém který je nutné řídit.
Z uvedených rovnic pro polohu auta je vidět, že neobsahuje žádný rovnovážný stav. Při řízení polohy by se musela používat linearizace v každém kroku výpočtu. V této práci kde se počítá pouze simulace a je potřeba brát ohled na rychlost výpočtu, to není vhodné použít. To je důvodem navržení řízení na základě úhlů.

Uživatel má možnost zadat reálnou situaci pomocí čtyř překážek. Pro zadané překážky, požadovanou a cílovou polohu auta se provede výpočet. Dynamickým programováním se provede návrh trajektorie. Tato trajektorie je ze všech cest nejkratší a navíc odpovídá vlastnostem systému. Díky těmto vlastnostem trajektorie jsem navrhl rychlé řízení, aniž bychom ztratili stabilitu.
V páté kapitole srovnávám všechny navržené regulátory. Uvedeny jsou nejčastější situace, které při couvání nastávají.
Kvalitu řízení zobrazuje 3D grafika s pomocí virtual reatly, kde blízkost průjezdu kolem překážek je patrná. V 3D grafice má uživatel možnost s pomocí myši provést vlastní regulaci. Ruční řízení má stejné omezení jako v případě regulátorů.
Situace, kdy má auto natočené všechny čtyři kola, je také v této práci popsána. Možnost zapojení dvou vozíků do série zvětšuje výpočetní složitost. Dynamické programování použité na hledání trajektorie není možné použít. Další proměnná a přidání omezení komplikují návrh. Díky nezdařenému návrhu jsem nenavrhnul pro možnost dvou vozíku regulaci.
Na přiloženém CD je zdrojový kód programu popsaný v této práci.

Literatura

[1] J. Štecha, V. Havlena: Moderní teorie řízení. Skriptum. ČVUT, Praha 2000

[2] J. Štecha: Optimální rozhodování a řízení. Skriptum ČVUT, Praha 1999

[3] S.Čelikovský: Nelineární systémy. Skriptum ČVUT, Praha 2005

[4] J. Štecha, V. Havlena: Teorie dynamických systémů. Skriptum ČVUT, Praha 1999

[5] J. D. Powell, G. F. Franclin: Feedback control of dynamic system. Prentice Hall, 2006

[6] J. M. Maciejowski: Predictive control with constraints. Prentice Hall 2001

[7] V. Havlena: Moderní teorie řízení. Přednáškové Slides. ČVUT, Praha 2006

[8] M.J. Tenny; S.J. Wright and J.B. Rawlings: Nonlinear model predictive control via feasibility-pertrubed sequential quadratic programming. Computational Optimization and Applications, 2004.

Seznam použitého software

Matlab verze:7.5(R2007b)

Virtual Reality Toolbox verze: 4.6 (R2007b)

Image Processing Toolbox verze: 6.0 (R2007b)

AutoCad 2006

Obsah přiloženého CD

Struktura adresářů na přiloženém cd je následující:

/dokumenty

Text diplomové práce ve formátech (doc, pdf).

/obrázky

Obrázky použité v diplomové práci.

/videa

Ukázkové videa automatického couvání.

/program

Všechny zdrojové kódy