Dynamický model a exaktní linearizace dvoukolového diferenciálně řízeného robotu
(pracovní text, prosím dále nešířit)

Robert Grepl

2007-06-12

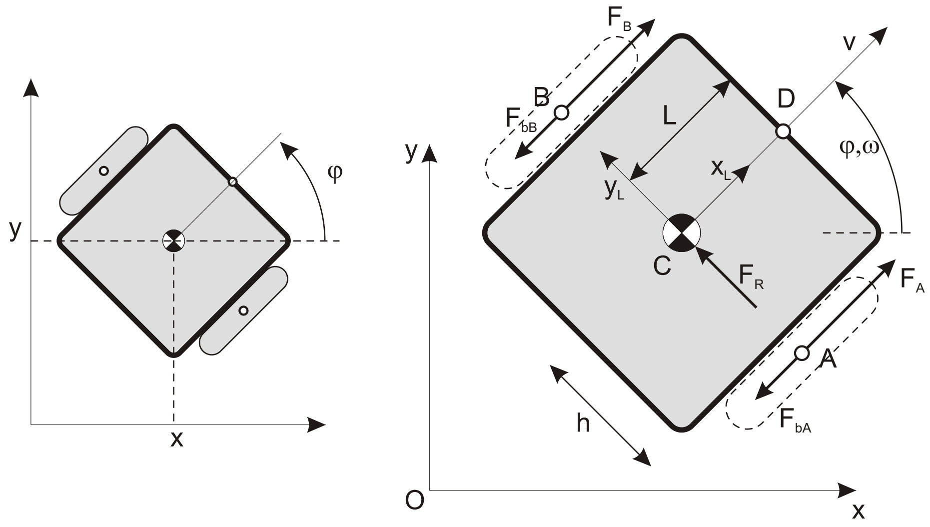
Obsah

1 Úvod

Cílem tototo textu je *popsat tvorbu redukovaného analytického dynamického modelu pro dvoukolového diferenciálně řízeného robota* podle obr. 1 *a následně provést exaktní linearizaci systému.*

Uvažujeme následující předpoklady:

- uvažujeme pohyb jednoho tuhého tělesa v rovině (zanedbáme tedy dynamiku kol)
- uvažujeme nulový boční prokluz
- uvažujeme viskózní odpory při valení kol robotu



Obr. 1:

Dynamický model vytvoříme klasickým přístupem uvolnění, následně transformujeme souřadnice a vyloučíme vektor vazbových sil λ (v našem případě sílu F_R). Tím je redukovaný model vytvořen.

Vytvořený nelineární systém linearizujeme pomocí metody exaktní linearizace. Klíčovým bodem je zde volba řídícího bodu D, která umožní elegantní provedení linearizace.

Poznamenejme ještě, že zvláštnost neholonomních soustav spočívá – intuitivně řečeno – v rozporu mezi tím, že globálně je možné polohovat robota ve všech 3 stupních volnosti (x, y, φ), avšak lokálně pouze ve dvou (x_L, φ). Z matematického hlediska je tento stav vyjádřen počtem ODE prvního řádu, který je pro volné těleso roven šesti (holonomní) a pro náš případ robotu roven pěti.

1.1 Rešerše

Literaturu použitou k sepsání tohoto textu lze v zásadě rozdělit do dvou skupin: modelování dynamiky neholonomních soustav a feedback linearization.

Metoda sestavení redukovaného dynamického modelu je patrně v neholonomní komunitě dosti známa, objevuje se v [2, 1, 3]. Z poměrně obtížně pochopitelných důvodů však autoři neuvěřitelně mlží až … podávají nepřesné informace. Např. některé předpoklady a logické vývody v [2] jsou dosud obestřeny tajemstvím (opravdu natolik nerozumím kinematice???).

Feedback linearization je opět obecně známý koncept [4, 5], překvapivě se ale neobjevuje často jako referenční (např. pro porovnání s NN). V [2] je řešen problém řízení čtyřkolového vozidla s diferenciálním řízením. Zdá se ale, že formálně vede problém na stejné rovnice. Ze zatím nejasných důvodů vede postup na lineární systém třetího řádu.

Při psaní tohoto textu jsme vycházeli především z článků [2, 1].

Vesměs všichni pánové zahrnují do svých úvah, modelů a řízení nejistoty parametrů.

2 Dynamický model s respektováním neholonomních vazeb

2.1 Sestavení modelu metodou uvolňování (Newton)

Těleso má v rovině tři stupně volnosti, k popisu tedy použijeme tří zobecněných souřadnic

$$\mathbf{q} = \begin{bmatrix} x \\ y \\ \varphi \end{bmatrix} \tag{1}$$

Metodou uvolňování sestavíme tři pohybové rovnice soustavy. Na robot působí dvě akční síly od pohonů F_A a F_B podle obrázku 1.

Viskózní tlumící síla působí proti směru pohybu kolečka, tedy ve směru záporné osy x_L. Velikost tlumící síly v kolečku A vypočteme pomocí rychlosti těžiště v_C a úhlové rychlosti okolo těžiště $\dot{\varphi}$ takto:

$$F_{bA} = bv_A = b(v_C + h\dot{\varphi}) \tag{2}$$

Pohyb v příčném (bočním) směru je zamezen silou F_R.

Nyní již můžeme sestavit soustavu tří ODE:

$$
\begin{aligned}
F_A \cos\varphi + F_B \cos\varphi - F_R \sin\varphi - \underbrace{b(\dot{x} + h\dot{\varphi}\cos\varphi)}_{F_{bAx}} - \underbrace{b(\dot{x} - h\dot{\varphi}\cos\varphi)}_{F_{bBx}} &= m\ddot{x} \quad (3) \\
F_A \sin\varphi + F_B \sin\varphi + F_R \cos\varphi - b(\dot{x} + h\dot{\varphi}\sin\varphi) - b(\dot{x} - h\dot{\varphi}\sin\varphi) &= m\ddot{y} \quad (4) \\
F_A h - F_B h - 2bh^2\dot{\varphi} &= I\ddot{\varphi} \quad (5)
\end{aligned}
$$

Tyto rovnice snadno přepíšeme do maticového tvaru

$$
\begin{bmatrix} m & & \\ & m & \\ & & I \end{bmatrix}
\begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{\varphi} \end{bmatrix}
+
\begin{bmatrix} -2b & & \\ & -2b & \\ & & -2bh^2 \end{bmatrix}
\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\varphi} \end{bmatrix}
=
\begin{bmatrix} \cos\varphi & \cos\varphi \\ \sin\varphi & \sin\varphi \\ h & -h \end{bmatrix}
\begin{bmatrix} F_A \\ F_B \end{bmatrix}
-
\begin{bmatrix} -\sin\varphi \\ \cos\varphi \\ 0 \end{bmatrix} F_R \tag{6}
$$

$$
\mathbf{M}\ddot{\mathbf{q}} + \mathbf{V}\dot{\mathbf{q}} = \mathbf{B}\mathbf{f} - \mathbf{A}^T\lambda \tag{7}
$$

V těchto třech rovnicích ($n = 3$) se vyskytují neznámé $\{\ddot{x}, \ddot{y}, \ddot{\varphi}, F_R\}$ a soustavu tedy v této podobě nemůžeme řešit. V případě holonomních soustav (běžných) doplníme rovnice vazbovými podmínkami (pro polohy), tyto dvakrát zderivujeme a dostaneme soustavu DAE řešitelnou např. v Matlabu. *Pro neholonomní soustavy se tento postup použít nedá*, protože mají podmínky vyjádřeny pro rychlosti.

2.2 Klasický způsob simulace neholonomní vazby

Běžný způsob modelování kolových vozidel např. v prostředí Matlab/Simulink nebo SimMechanics pracuje takto:

- těleso má v rovině 3 stupně volnosti
- holonomní vazbovou sílu počítáme jako funkci rychlosti

V našem případě můžeme simulační model snadno sestavit položíme-li

$$
F_R = -k\dot{y}_L \tag{8}
$$

kde k je velké číslo (dobře funguje např. `k = 1e-6;`).

Ve výsledku takto dojde k zamezení pohybu v lokální ose y_L a tedy správnému pohybu vozidla (přesvědčit se můžeme vykreslením integrované hodnoty posuvu během pohybu).

Nevýhodnou tohoto přístupu je zakomponování vysoké „tuhosti“ a tím vysokých vlastních čísel do celého systému. *Důsledkem je (razantní) zpomalení simulace* (`ode45` apod. – automatická volba délky kroku).

2.3 Redukce modelu zahrnutím neholonomních podmínek

Zanedbáme-li u robotu možnost bočního prokluzu, pak nutno definovat vazebnou podmínku

$$\dot{y}_L = 0 \tag{9}$$

kterou můžeme pro pevný souřadný systém O přepsat takto:

$$\frac{\dot{y}}{\dot{x}} = \tan\varphi \tag{10}$$

Tuto jednu neholonomní podmínku ($k = 1$) upravíme do maticového tvaru

$$\dot{x}\tan\varphi - \dot{y} = 0 \tag{11}$$

$$\begin{bmatrix}\tan\varphi & -1 & 0\end{bmatrix}\begin{bmatrix}\dot{x}\\ \dot{y}\\ \dot{\varphi}\end{bmatrix} = 0 \tag{12}$$

$$\mathbf{A}(\mathbf{q})\dot{\mathbf{q}} = \mathbf{0} \tag{13}$$

Poznamenejme, že naše soustava nemá žádnou holonomní podmínku, proto celkový počet omezujících holonomních i neholonomních podmínek je $m = k + 0 = 1$.

Definujeme matici $\mathbf{S}(\mathbf{q})$ o rozměru $n \times n - m$ takovou, že

$$\mathbf{A}(\mathbf{q})\mathbf{S}(\mathbf{q}) = \mathbf{0} \tag{14}$$

Pro náš případ je velmi pěknou definicí následující matice (lze najít např. v [3]):

$$\mathbf{S} = \begin{bmatrix}\cos\varphi & 0\\ \sin\varphi & 0\\ 0 & 1\end{bmatrix} \tag{15}$$

Máme-li vhodnou matici (null space), definujeme vektor $\mathbf{v}$ tak aby platilo:

$$\dot{\mathbf{q}}(\mathbf{q}) = \mathbf{S}(\mathbf{q})\mathbf{v}(t) \tag{16}$$

což je v našem případě (přeznačeno pro další snadnější zápis)

$$\mathbf{v} = \begin{bmatrix}\dot{x}_L\\ \dot{\varphi}\end{bmatrix} = \begin{bmatrix}v\\ \omega\end{bmatrix} \tag{17}$$

Derivujeme rov. 16 podle času

$$\ddot{\mathbf{q}} = \dot{\mathbf{S}}\mathbf{v} + \mathbf{S}\dot{\mathbf{v}} \tag{18}$$

$$\dot{\mathbf{S}} = \begin{bmatrix}-\sin\varphi & 0\\ \cos\varphi & 0\\ 0 & 0\end{bmatrix}\dot{\varphi} \tag{19}$$

a odvozené vztahy dosadíme do 7. Celou rovnici pak zleva vynásobíme $\mathbf{S}^T$.

$$
\begin{aligned}
\mathbf{M\dot{S}v} + \mathbf{MS\dot{v}} + \mathbf{VSv} &= \mathbf{Bf} - \mathbf{A^T}\lambda & (20)\\
\mathbf{S^TMS\dot{v}} + \mathbf{S^T}(\mathbf{VSM\dot{S}})\mathbf{v} &= \mathbf{S^TBf} - \underbrace{S^TA^T}_{=(\mathbf{AS})^\mathbf{T}=\mathbf{0}}\lambda & (21)
\end{aligned}
$$

Vidíme, že se nám podařilo odstranit neznámý vektor λ, což lze považovat za klíčový úspěch. Výsledná rovnice společně s 16 tvoří nový redukovaný model systému

$$
\begin{aligned}
\mathbf{S^TMS\dot{v}} + \mathbf{S^T}(\mathbf{VS} + \mathbf{M\dot{S}})\mathbf{v} &= \mathbf{S^TBf} & (22)\\
\mathbf{\dot{q}} &= \mathbf{Sv} & (23)
\end{aligned}
$$

Výsledný model je soustavou pěti ODE prvního řádu se stavovým vektorem $[v, \omega, x, y, \varphi]$. Těleso v rovině je přitom popsáno 3 ODE druhého řádu (což odpovídá 6-ti ODE prvního řádu). Došlo tedy k redukci dynamického systému vlivem neholonomní vazby. Těleso se stále může pohybovat ve všech třech stupních volnosti x, y, φ, v pravém slova smyslu však nemá tři mechanické stupně volnosti (má 2,5 stupně volnosti – „intuitivní popis neholonomního systému“).

3 Exaktní linearizace (feedback linearization)

Rovnici 22 upravíme do formálně jednoduššího tvaru, význam nových matic je zřejmý porovnáním.

$$
\begin{aligned}
\overline{\mathbf{M}}\mathbf{\dot{v}} + \overline{\mathbf{K}}\mathbf{v} &= \overline{\mathbf{B}}\mathbf{f} & (24)\\
\mathbf{\dot{q}} &= \mathbf{Sv} & (25)
\end{aligned}
$$

Nyní můžeme snadno provést částečnou linearizaci (partially linearizing static feedback [2]).

Vstupní vektor akčních sil $\mathbf{f}$ definujeme nově takto:

$$
\mathbf{f} = \overline{\mathbf{B}}^{-1}(\overline{\mathbf{M}}\mathbf{u} + \overline{\mathbf{K}}\mathbf{v}) \tag{26}
$$

kde $\mathbf{u}$ je nový vstup. Vidíme, že se nám situace zjednoduší takto:

$$
\begin{aligned}
\mathbf{\dot{v}} &= \mathbf{u} & (27)\\
\mathbf{\dot{q}} &= \mathbf{Sv} & (28)
\end{aligned}
$$

Výsledek je to zajímavý, nicméně řídit robota pomocí souřadnice x_L je poněkud nepohodlné. Pokračujme proto dále vstříc lepšímu řešení.

Velice zajímavého výsledku dosáhneme prostým trikem, který lze najít v mnoha příspěvcích [2].

Připomeňme, že naším cílem je kvalitně řídit polohu robotu v rovině danou souřadnicemi jeho těžiště C. Pokud tento požadavek změníme tak, že budeme místo toho řídit polohu bodu D, dosáhneme velice zajímavého výsledku.

Definujme tedy nový vektor řízené veličiny

$$\mathbf{z} = \begin{bmatrix} x + L\cos\varphi \\ y + L\sin\varphi \end{bmatrix} \tag{29}$$

Derivujeme podle času a dostaneme

$$\dot{\mathbf{z}} = \begin{bmatrix} \dot{x} - L\dot{\varphi}\sin\varphi \\ \dot{y} + L\dot{\varphi}\cos\varphi \end{bmatrix} = \begin{bmatrix} \cos\varphi & -L\sin\varphi \\ \sin\varphi & L\cos\varphi \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix} \tag{30}$$

$$\dot{\mathbf{z}} = \mathbf{P}\mathbf{v} \tag{31}$$

Zderivujeme ještě jednou

$$\ddot{\mathbf{z}} = \dot{\mathbf{P}}\mathbf{v} + \mathbf{P}\dot{\mathbf{v}} \tag{32}$$

$$\dot{\mathbf{P}} = \begin{bmatrix} -\sin\varphi & -L\cos\varphi \\ \cos\varphi & -L\sin\varphi \end{bmatrix}\omega \tag{33}$$

Rovnici 27 vynásobíme zleva maticí $\mathbf{P}$ a provedeme další úpravy

$$\mathbf{P}\dot{\mathbf{v}} = \mathbf{P}\mathbf{u} \tag{34}$$

$$\ddot{\mathbf{z}} - \dot{\mathbf{P}}\mathbf{v} = \mathbf{P}\mathbf{u} \tag{35}$$

$$\ddot{\mathbf{z}} = \mathbf{P}\mathbf{u} + \dot{\mathbf{P}}\mathbf{v} \tag{36}$$

a definujeme vstup $\mathbf{u}$ takto:

$$\mathbf{u} = \mathbf{P}^{-1}(\mathbf{r} - \dot{\mathbf{P}}\mathbf{v}) \tag{37}$$

kde $\mathbf{r}$ je znovu (a naposledy) novým předefinovaným vstupem.

Touto kouzelnou úpravou dosáhneme kýženého výsledku – lineárního systému s rozumnou řízenou veličinou. Původní systém je nahrazen dvěmi nezávislými (decoupled) řetězci dvou integrátorů.

$$\ddot{\mathbf{z}} = \mathbf{r} \tag{38}$$

Pozn.: takto jednoduché a přímočaré řešení jsem zatím v žádné publikaci [?, 1] nenašel...

4 Návrh regulátoru

4.1 Soustava druhého řádu

Po exaktní linearizaci řídíme separovanou soustavu druhého řádu 38. Zvolme

$$\mathbf{r} = \ddot{\mathbf{z}}^d - \mathbf{K_0}\tilde{\mathbf{z}} - \mathbf{K_1}\dot{\tilde{\mathbf{z}}} \tag{39}$$

$$\tilde{\mathbf{z}} = \mathbf{z} - \mathbf{z}^d \tag{40}$$

kde $\mathbf{z}^d$ je požadovaná a $\mathbf{z}$ skutečná poloha bodu D robotu. Po úpravě dostaneme

$$\ddot{\tilde{\mathbf{z}}} + \mathbf{K_1}\dot{\tilde{\mathbf{z}}} + \mathbf{K_0}\tilde{\mathbf{z}} = 0 \tag{41}$$

Připomeňme, že matice $\mathbf{K_1}$ a $\mathbf{K_0}$ jsou diagonálními maticemi a rovnice 41 je tedy soustavou $n - m = 2$ *nezávislých harmonických oscilátorů, jejichž vlastnosti můžeme libovolně volit.*

To je klíčová vlastnost separovaných rovnic, které vzniknou metodou exaktní linearizace.

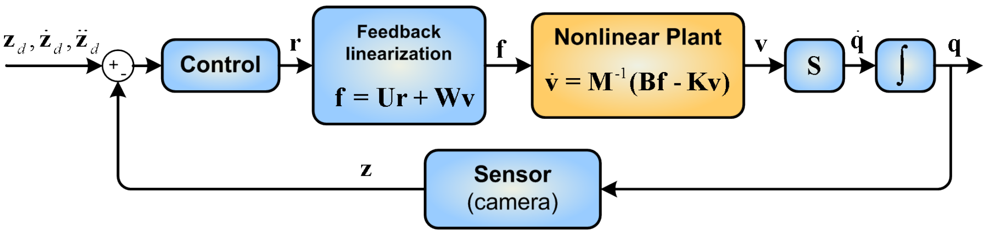
Nastavíme-li $\mathbf{K_0}$ tak, aby rychlost regulace odpovídala našim požadavkům (odpovídá hodnotě Ω^2 mechanického oscilátoru) a dále při

$$\mathbf{K_1} = \begin{bmatrix} 2\sqrt{k_{01}} & 0 \\ 0 & 2\sqrt{k_{02}} \end{bmatrix} \tag{42}$$

dosáhneme kritického tlumení (odpovídá hodnotě $\delta = \Omega$ u mechanického oscilátoru).

4.2 Schéma řízení a výpočet vektoru řízení

Schéma celé řízené soustavy pak můžeme znázornit na obr. 2.



Obr. 2:

Doplňme ještě vyjádření velikosti řídícího vektoru $\mathbf{f}$, kterým řídíme robota. Vyjádříme ho z rovnic 26 a 37 takto:

$$\mathbf{f} = \overline{\mathbf{B}}^{-1}\overline{\mathbf{M}}\mathbf{P}^{-1}\mathbf{r} - \overline{\mathbf{B}}^{-1}(\overline{\mathbf{M}}\mathbf{P}^{-1}\dot{\mathbf{P}} + \overline{\mathbf{K}})\mathbf{v} \tag{43}$$

což zapíšeme jako

$$\mathbf{f} = \mathbf{U}(\varphi)\mathbf{r} + \mathbf{W}(\omega)\mathbf{v} \tag{44}$$

$$\mathbf{U}(\varphi) = \frac{1}{2hL}\begin{bmatrix} mhL\cos\varphi - I\sin\varphi & mhL\sin\varphi + I\cos\varphi \\ mhL\cos\varphi + I\sin\varphi & mhL\sin\varphi - I\cos\varphi \end{bmatrix} \tag{45}$$

Výsledné matice vypadají po úpravě (Maple) následovně:

$$\mathbf{W}(\omega) \;=\; \begin{bmatrix} \frac{2bhL + I\omega}{2hL} & \frac{\omega mL}{2} + hb \\ \frac{2bhL + I\omega}{2hL} & \frac{\omega mL}{2} - hb \end{bmatrix} \tag{46}$$

Je vidět, že pro $h, L \neq 0$ lze řízení vypočítat vždy. Tím je problém vyřešen.

5 Simulační experimenty

5.1 Implementace, porucha, plánovač

Výše popsaný algoritmus byl implementován v prostředí Matlab (řešič `ode45`) a odladěn. Simulační výpočet probíhá v porovnání s klasickým řešením popsaným v části 2.2 velice rychle.

Pro ověření kvality řízení zavedeme do soustavy 7 vektor poruchové síly

$$\mathbf{f_P} = [F_{Px}, F_{Py}, M_{Pz}] \tag{47}$$

$$\mathbf{M}\ddot{\mathbf{q}} + \mathbf{V}\dot{\mathbf{q}} \;=\; \mathbf{Bf} - \mathbf{A}^T\lambda + \mathbf{f_P} \tag{48}$$

který se v rovnici 24 projeví takto:

$$\overline{\mathbf{M}}\dot{\mathbf{v}} + \overline{\mathbf{K}}\mathbf{v} \;=\; \overline{\mathbf{B}}\mathbf{f} + \mathbf{S}^T\mathbf{f_P} \tag{49}$$

Pro naplánování trajektorie (tedy volbu $\mathbf{z}^d$, $\dot{\mathbf{z}}^d$ a $\ddot{\mathbf{z}}^d$ použijeme funkci `jtraj` z Robotic Toolbox for Matlab. Funkce definuje polohu, rychlost a zrychlení pomocí polynomu sedméno stupně.

5.2 Výsledky

Na následujících dvou obrázcích je uvedena požadovaná a skutečná trajektorie pro dva případy nastavení parametrů matice $\mathbf{K_0}$. Matice $\mathbf{K_1}$ byla vždy dopočítána podle 42 (kritické tlumení).

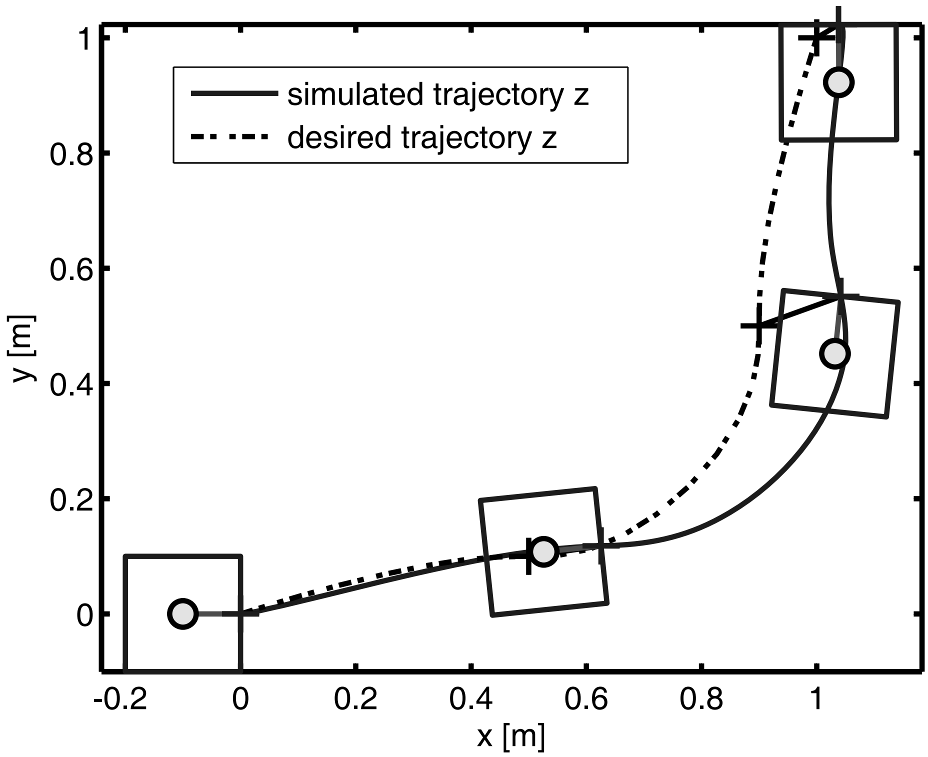
Parametry byly nastaveny následovně:

```
%% Parametry robotu
m = 1    % [kg]
J = 0.10 % [kgm2]
b = 0.1
h = 0.1  % [m]
L = 0.1  % [m]
```

```
y0 = [-L 0 0 0 0]'; % poc. podm

%% Definice uzlu trajektorie
trasa.t = [0 1 2 3];
trasa.x = [0 0.5  .9  1];
trasa.y = [0 0.1   0.5 1];
trasa.dx = [0 0.5  0  0];
trasa.dy = [0 0  0.5  0];
```

Porucha byla u obou variant shodná $\mathbf{f_P} = [1, 0, 0]$ N.

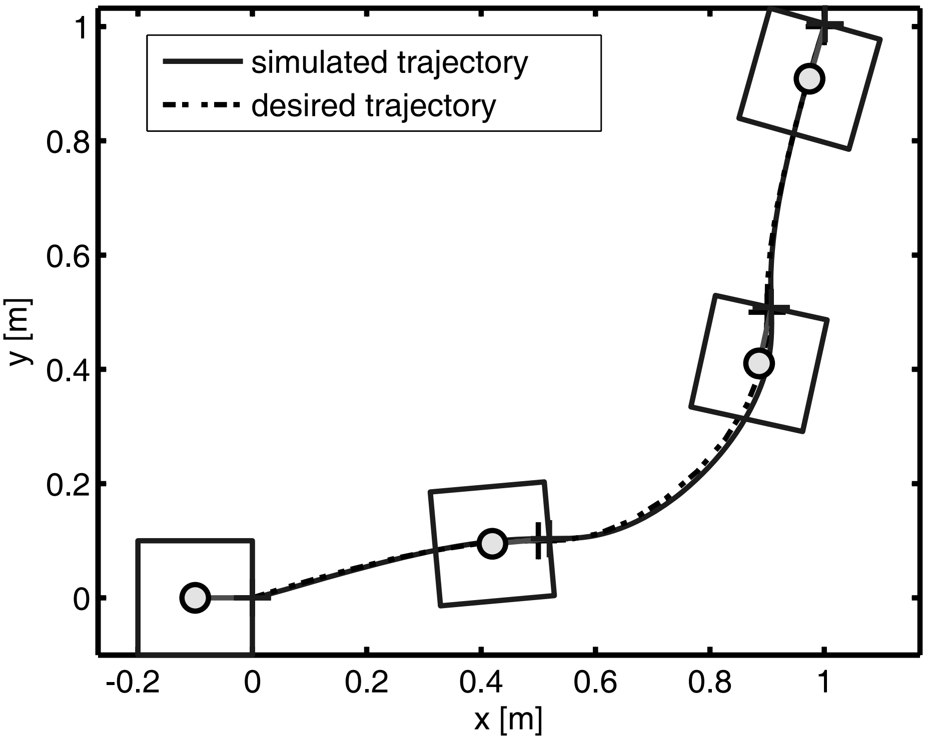


Obr. 3: $k_{01} = k_{02} = 5$

6 Závěr

[1] Oh, C., Kim, M.-S., Lee, J.J.: *Control of nonholonomic mobile robot using an RBF network*, Artificial Life Robotics 8, pp.14-19, 2004

[2] Caracciolo, L., De Luca, A., Iannitti, S.: *Trajectory Tracking Four-Wheel Differentially Driven Mobile Robot*, International Conference on Robotics & Automation, Detroit, Michigan, 1999



Obr. 4: $k_{01} = k_{02} = 50$

[3] Gholipour, A., Yazdanpanah, M. J.: *Dynamic tracking control of Nonholonomic Mobile Robot with Model Reference Adaptation For Uncertainparameters*, ,

[4] Spong, M.W., Hutchinson, S., Vidyasagar, M.: *Robot modeling and control*, , 2006

[5] Sciavicco, L.: *Modelling and Control of Robot Manipulators*, , 2004

6.1 Todo

- klíčové: srovnat kinematicky řízený robot s výše popsaným řízením (je vůbec feedback lin. varianta praktickým přínosem???)
- zahrnout do modelu i dynamiku kol (ovlivní to pouze hodnoty v maticích?)
- přidat zoběcnění a teoretický podklad volby matice S
- ujasnit si pojmy "feedback linearization" a "exact linearization" – je mezi nimi rozdíl? Má "feed. lin." nějaký český překlad?
- citlivost na parametry (především tlumení)
- vyzkoušet jak bude fungovat "feedback linear control" výše popsaného neholonomního modelu na modelu v SimMechanics, který zahrnuje i všemožné prokluzy
- ? do jednoduchého analytického modelu zahrnout dynamiku kol?
- lze někde najít teoretický výklad vlastností neholonomních systémů ?
- nahradit `jtraj` v simulacích nějakým realističtějším plánovačem
- odhad doby výpočtu inv. dyn. modelu, zavedení příslušného zpoždění do simulací
- promyslet, proč používá de Luca "dynamic extension" - tj. přidání dalšího integrátoru – výsledný systém má $\ddot{z} = r$