National Conference with International Participation
ENGINEERING MECHANICS 2006
Svratka, Czech Republic, May 15 – 18, 2006
paper no. 329

THE AUTOMATIC GENERATION OF WALKING POLICIES FOR A FOUR-LEGGED ROBOT IN A NONDETERMINISTIC SPACE

V. Ondroušek[*], T. Březina[**]

Summary: *Both the automatic policies generation and its analysis for four legged walking robot is given in this contribution. Policies generation is based on state space search which is represented by a tree. The set of elementary production rules is used for tree generation, chosen with respect to obtain suitable tree complexity and algorithm response. Further the selection of tasks operators are discussed. Those operators represent the performance of basic movements of walking robot. Policy generation is made through the A* algorithm. The policies are continuously generated as short-time plans of robot movements.*

1. Úvod

Nedílnou součástí návrhu kráčivých robotů je realizace autonomní soustavy, která bude schopna pohybu v neznámém proměnlivém prostředí. Při pohybu robotu jsou řešeny stěžejní úlohy jako je plánování cesty robotu anebo lokalizace v mapě. Velmi důležité je rovněž nalezení vhodného stylu chůze mobilního robotu.Jedná se o úlohu plánování. Cílem této úlohy je nalezení optimální cesty. Cestu tvoří posloupnost stavů a realizovaných operátorů, které způsobují přechod mezi stavy. Jednotlivé stavy reprezentují konkrétní konfiguraci robotu. Realizace operátoru představuje pravidlo pro změnu konfigurace robotu a tím vytvoření nového stavu. Pro řešení této problematiky lze úspěšně využít informovaných metod prohledávání stavového prostoru. Pro dosažení přijatelné složitosti úlohy je podstatná nejen volba algoritmu, ale také množina pravidel pro generování nových stavů společně s hodnotící funkcí, která výrazným způsobem ovlivňuje velikost generovaného stavového prostoru a tím i složitost celé úlohy. V příspěvku jsou navrženy dvě různé množiny produkčních pravidel a příslušných hodnotících funkcí, které vedou na různě velké prohledávací prostory a různé styly chůze čtyřnohého kráčivého robotu. Pro účely ověření vhodnosti použitého algoritmu a navržených pravidel byla vytvořena softwarová simulace čtyřnohého kráčivého robotu ve 2D s konstantní výškou těla nad povrchem.

2. Použitý přístup

Při výběru vhodného algoritmu pro generování stavového prostoru, bylo hlavním kritériem dosažení použitelné složitosti řešení s ohledem na dobu odezvy algoritmu. Z tohoto důvodu je zcela vyloučeno uvažovat metody slepého prohledávání a je nutné použít některou

[*] Vít Ondroušek, Ing., VUT v Brně, FSI ÚAI, Technická 2, 61669 Brno, ČR
e-mail: yondro00@stud.fme.vutbr.cz
[**] Tomáš Březina, doc., RNDr., Ing., CSc., VUT v Brně, FSI ÚAI, Technická 2, 61669 Brno, ČR
e-mail: brezina@fme.vutbr.cz

s informovaných metod. Za nejjednodušší heuristickou metodu lze považovat gradientní algoritmus (Pearl, 1984). Tento algoritmus expanduje stav, který byl dosud vyhodnocen hodnotící funkcí jako nejlepší a vyhodnocuje jeho následníky. Předchůdce tohoto stavu je zapomenut, stejně tak jako jeho sourozenci. Z tohoto plyne hlavní nevýhoda čistě gradientních strategií; expanze stavového prostoru může skončit v lokálním minimu hodnotící funkce, které však nemusí být globálním extrémem. Ve zvoleném přístupu proto byla použita modifikace gradientního algoritmu, označovaná jako uspořádané prohledávání (best-first search). Modifikace spočívá v zapamatování si předchůdce každého vygenerovaného stavu, které pak ve svém důsledku umožňuje nalezení globálního extrému, jenž představuje nalezení v jistém smyslu optimálního řešení. Algoritmus metody uspořádaného prohledávání (Mařík,1993):

1. Počáteční stav zapiš do seznamu OPEN, seznam CLOSED je prázdný.
2. Pokud je seznam OPEN prázdný, řešení neexistuje, ukonči prohledávání.
3. Ze seznamu OPEN vyber stav *i* s nejmenší hodnotou *f(i)*. V případě většího počtu stavů se stejnou minimální hodnotou *f(i)* prověř, zda některý z těchto stavů není stavem cílovým, v takovém případě jej vyber; jinak vyber mezi stavy se stejnou minimální hodnotou *f(i)* libovolně.
4. Vymaž stav *i* se seznamu OPEN a zařaď jej do seznamu CLOSED.
5. Je-li stav *i* cílovým stavem, řešení je nalezeno, ukonči prohledávání.
6. Expanduj stav *i*; pro každého následovníka *j* stavu *i* vypočítej hodnotící funkci *f(j)*. Pokud stav *j* není ani v seznamu OPEN ani v seznamu CLOSED, zařaď jej do seznamu OPEN. Pokud je stav *j* již v seznamu OPEN nebo CLOSED, avšak s ohodnocením větším než právě vypočtené *f(j)* změň jeho ohodnocení na *f(j)*, změň jméno rodičovského uzlu v zápisu uzlu a zařaď ho do seznamu OPEN.
7. Pokračuj krokem č.2.

Z popisu algoritmu vyplývá, že pořadí v jakém jsou uzly generovány, je zcela závislé na hodnotící funkci *f(i)*. Její nevhodný návrh proto může vést na vytvoření nepřípustně velkého stavového prostoru, což může v důsledku znemožnit nalezení optimálního řešení. Proto je nutné věnovat návrhu hodnotící funkce velkou pozornost. Jestliže formálně vyjádříme hodnotu funkce *f(i)* v *i-tém* stavu jako součet ceny *g(i)* optimální cesty z počátečního stavu s_0 do stavu *i*, a ceny *h(i)* optimální cesty ze stavu *i* do některého z cílových stavů, pak hovoříme o algoritmu A, tj. $f(i) = g(i) + h(i)$. V řešené úloze chůze robotu, však hodnoty funkcí *g(i)* a *h(i)* nejsou známi, proto musíme použít jejich odhady $g(i)^*$ a $h(i)^*$. Funkce $g(i)^*$ pak představuje minimální dosud zjištěnou cenu přechodu z počátečního stavu do stavu *i*. Funkce $h(i)^*$ se označuje jako heuristická a vyjadřuje kvantitativně náš odhad cesty z uzlu do cíle.

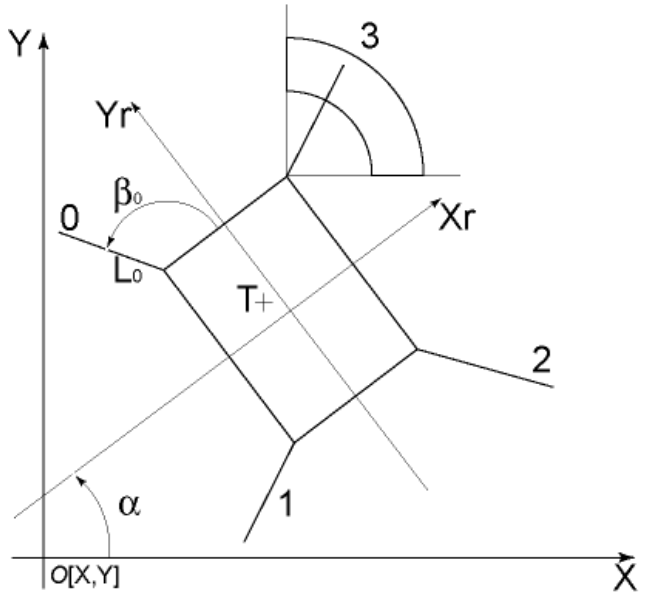
3. Implementace

Pro testování navržených produkčních pravidel a hodnotících funkcí bylo použito softwarové simulace chůze čtyřnohého robotu v rovinném prostředí s konstantní výškou těla nad povrchem. Implementace byla provedena použitím objektově orientovaného programování v prostředí Borland Delphi 6 a zahrnuje jak vlastní algoritmus, tak i zjednodušený kinematický model robotu. Vyvinutý program obsahuje třídy reprezentující vlastnosti a metody potřebné pro reprezentaci a expanzi stavového prostoru, pro reprezentaci stavu robotu a rovněž jednoduchý grafický interface znázorňující chůzi robotu dle nalezeného řešení.

3.1 Reprezentace stavu robotu

Problém chůze robotu je pro zjednodušení řešen v rovině těla robotu. Tělo robotu má obdélníkový tvar o rozměrech $a \cdot b$. Ve vrcholech obdélníků jsou umístěny klouby noh. Každá noha má definován svůj pracovní prostor, jedná se o výseč mezikruží, viz obr.1. Oblast je určena minimální a maximální délkou nohy, $L_i \in \langle L_{i\min}, L_{i\max} \rangle$, a minimálním a maximálním úhlem natočení nohy vůči ose X_r, $\beta_i \in \langle \beta_{i\min}, \beta_{i\max} \rangle$. U každé nohy lze definovat různé rozsahy délek i úhlů. Poloha těžiště robotu je obecně různá od středu těla.

Pro potřeby výpočtů je zavedeno několik souřadných systémů. Absolutní souřadný systém $O[x, y]$ v němž se robot pohybuje a relativní souřadný systém $O'[x_r, y_r]$ jehož počátek je posunut do středu těla robotu a pootočen o úhel φ oproti absolutnímu souřadnému systému. Kromě kartézských souřadnic jsou pro každou nohu zavedeny souřadnice polární $P_i(L_i, \beta_i)$, kde L_i je délka i-té nohy a β_i je úhel, který svírá i-tá noha s osou X_r. Kladná orientace úhlů je zavedena proti směru hodinových ručiček.



Obr.1 Schéma stavu robotu v zavedených souřadných systémech

V každém stavu robotu je k dispozici informace o geometrické vzdálenosti středu těla od cílového bodu určeného v absolutním souřadném systému, a rovněž máme k dispozici výchylku osy Y_r od cílového bodu. Tyto informace představují výrazné zjednodušení oproti reálnému řízení, kdy máme k dispozici tyto informace pouze poté, co dojde k lokalizaci v mapě prostředí. Z časových důvodů však není možné provádět lokalizaci na mapě v každém novém stavu robotu.

3.2 Expanze stavového prostoru

Expanze stavového prostoru je prováděna pomocí elementárních produkčních pravidel. Aplikací produkčního pravidla na stav robotu získáme nový stav robotu, čímž dojde k rozšíření stavového prostoru úlohy o nový stav. Pro potřebu chůze robotu byla zavedena pravidla, které lze formálně rozčlenit do tří skupin:

Rotace těla kolem jeho středu: maximální úhel o který lze robot rotovat kolem středu těla je omezen především pracovním prostorem noh. V této skupině jsou zařazena dvě produkční

pravidla; rotace robotu o stanovený úhel a snížení odchylky od ideálního směru na minimum. Stanoveným úhlem je myšlen maximální možný úhel, který je umožněn stávající konfigurací robotu tak, aby nedošlo k porušení tříbodového staticky stabilního postavení robotu.

Posuv těla je obecně možné provádět v libovolném směru v závislosti na pracovním prostoru noh. Pro simulaci byla použita pouze pravidla pro posuv v kladném i záporném směru osy Y_r o experimentálně stanovené ΔY, nebo o maximální možné ΔY směrem k cíli.

Pohyb nohy: do této skupiny náleží pravidla měnící délku nohy a úhel mezi nohou a osou X_r. Např. pohyb nohy na nominální délku a úhel.

3.3 Hodnotící funkce

Implementovaný algoritmus generování stavového prostou používá hodnotící funkci ve tvaru $f^*(i) = g^*(i) + h^*(i)$. Z předpisu funkce je patrné, že jsou použity odhady funkcí *g(i)* a *h(i)*, jedná se proto o algoritmus A. V některých navrhovaných hodnotících funkcích je za hodnotu $g^*(i)$ dosazena hodnota $f^*(j)$, kde *j* je předchůdcem stavu *i*. Funkce pak má kumulativní charakter, a zohledňuje veškeré stavy na právě plánované cestě k cíli.

4. Dosažené výsledky

S uvážením výše uvedených skutečností, bylo navrženo a testováno několik desítek hodnotících funkcí a několik různě obsáhlých množin produkčních pravidel. Veškerá následná experimentální ověření probíhala za pomoci uvedeného software. Pro potřeby simulace byly zvoleny neměnné parametry robotu, viz. tab.1.

Tab. 1.: použité parametry robotu

Rozměry těla robotu	
šířka *a*	2 [j]
délka *b*	3 [j]
Pracovní prostor noh	
Minimální délka *Lmin*	1 [j]
Maximální délka *Lmax*	2 [j]
$\beta_{0\min} = \beta_{1\min}$	3/4 Pi [rad]
$\beta_{0\max} = \beta_{1\max}$	5/4 Pi [rad]
$\beta_{2\min} = \beta_{3\min}$	1/4 Pi [rad]
$\beta_{2\max} = \beta_{3\max}$	7/4 Pi [rad]

Každý test byl postupně proveden pro dva různé výchozí stavy: a) osa X_r robotu svírá s osou X úhel $\alpha_1 = (1/4)Pi\,[rad]$, b) osa X_r robotu svírá s osou X úhel $\alpha_2 = (3/2)Pi\,[rad]$, nohy jsou nastaveny do nominální polohy. Cílový stav byl vždy umístěn do bodu [-10,10].

Experimentálně byly prověřeny dvě různé množiny produkčních pravidel, ***A*** a ***B***. Množina ***A*** je tvořena pouze 11-ti pravidly:

1. rotace těla robotu kolem středu tak, aby se minimalizovala odchylka robotu od směru k cíly.
2. translace robotu ve směru osy Y_r o $\Delta Y = b/3$, viz. tab.1
3. maximální možná translace robotu ve směru osy Y_r

4. až 7 nastavení i-té nohy na maximální délku a maximální úhel
8. až 11 nastavení i-té nohy na střední délku a nominální úhel, tj. Pi rad pro nohy 0,1 a 0 rad pro nohy 2,3.

Množina ***B*** je tvořena 27 pravidly a výrazným způsobem rozšiřuje množinu ***A*** o pravidla manipulující s nohama:

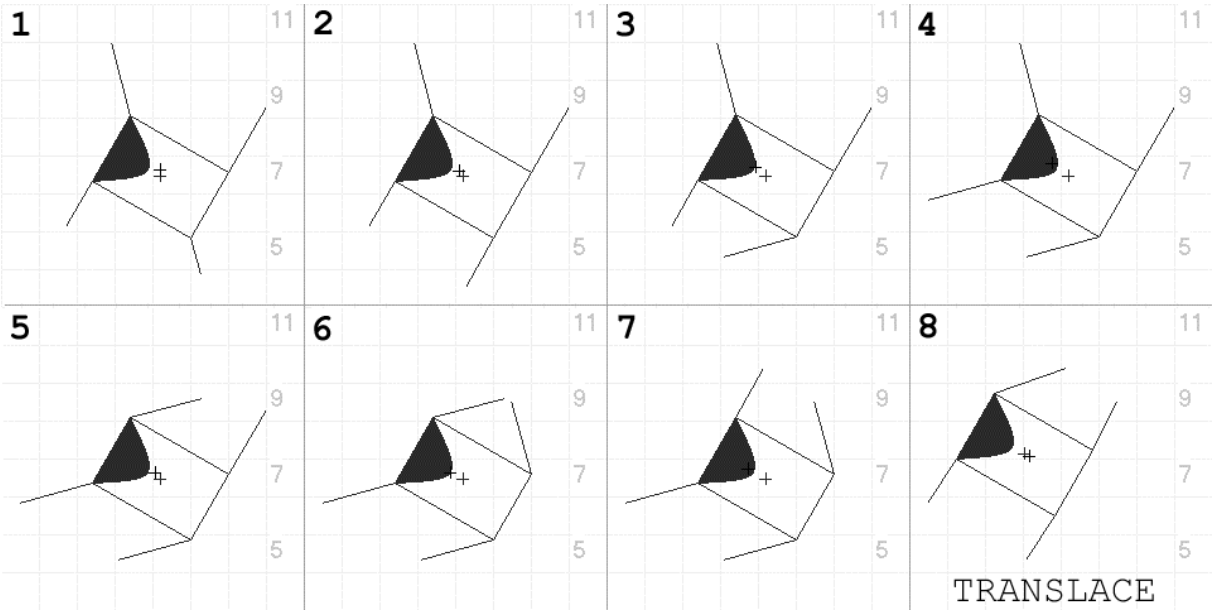
12. až 15 nastavení i-té nohy na maximální délku
16. až.19 nastavení i-té nohy na minimální délku
20. až.23 nastavení i-té nohy na maximální pracovní úhel
24. až.27 nastavení i-té nohy na minimální pracovní úhel

V testovaných hodnotících funkcích jsou použity tyto symboly: *vzd(i)* reprezentuje geometrickou vzdálenost stavu *i* od cílového stavu, *dev(i)* udává výchylku stavu *i* od ideálního směru robotu, *step(i)* nabývá hodnoty 1 pokud stav *i* vznikl z předchůdce aplikací pravidla měnící polohu nohy, jinak hodnoty 0. V předpisu hodnotící funkce jsou dále použity koeficienty $k_1, k_2, k_3;\ k_i \in \langle 0,100 \rangle$.

První navržená hodnotící funkce má tvar:

$$h^*(i) = k_1 vzd(i) + k_2 dev(i)\ , \qquad (1)$$

kde stav *i* je přímým následníkem stavu *j*. Konstanty k_i byly stanoveny experimentálně: $k_1 = 1$, $k_2 = 5$. Všimněme si, že v zápisu rovnice (1) chybí hodnota $H(j)$, tzn. hodnotící funkce *H* neuvažuje při výpočtu ohodnocení stavu *i* ohodnocení přímého předchůdce. Funkce *h** tedy nezohledňuje cenu *g(i)* cesty z počátečního stavu s_0 do stavu *i*. Při použití produkčních pravidel z množiny ***A*** a výchozího stavu, $\alpha_1 = (1/4)Pi\,[rad]$, bylo expandováno celkem 284 stavů a nalezené optimální řešení obsahovalo 45 stavů. Při použití produkčních pravidel z množiny ***B*** a výchozího stavu, $\alpha_2 = (3/2)Pi\,[rad]$, bylo expandováno již 18948 stavů a nalezené řešení obsahovalo 211 stavů. Styl výsledné chůze je však velmi neefektivní. Nalezené řešení obsahuje velké množství redundantních pohybů noh, , viz. posloupnost stavů obr.2.. Translace těla robotu jsou prováděny většinou po menších vzdálenostech, namísto propojení translačních pohybů do větších posuvů.

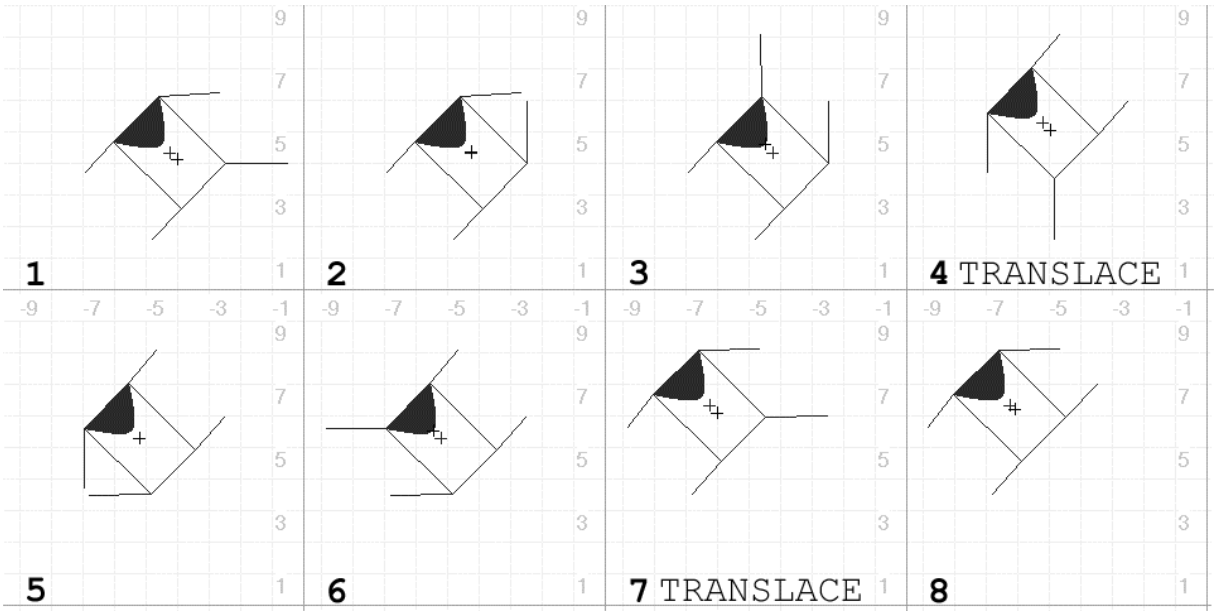


Obr.2: Redundantní pohyby noh mezi translačním pohybem

Další navržená hodnotící funkce má tvar:

$$h^*(i) = k_1\left(vzd(i) - vzd(j)\right) + k_2\left(dev(i) - dev(j)\right) + h^*(j), \qquad (2)$$

kde stav *i* je přímým následníkem stavu *j*. Konstanty k_i byly stanoveny experimentálně: $k_1 = 1$, $k_2 = 5$. Rovnice (2) na rozdíl od (1) již uvažuje při výpočtu ohodnocení dosažené konfigurace robotu (stav *i*) rovněž konfiguraci předcházející (stavu *j,* který je přímý předchůdce stavu *i*). Funkce *h** tedy zohledňuje cenu *g(i)* cesty z počátečního stavu s_0 do stavu *i*. Při použití produkčních pravidel z množiny ***A*** a výchozího stavu, $\alpha_1 = (1/4)Pi\,[rad]$, bylo expandováno celkem 284 stavů a nalezené optimální řešení obsahovalo 45 stavů. Při použití produkčních pravidel z množiny ***B*** a výchozího stavu, $\alpha_2 = (3/2)Pi\,[rad]$, bylo expandováno 9072 stavů a nalezené řešení obsahovalo 168 stavů. Porovnáním výsledků vidíme, že při použití pouze 11 produkčních pravidel nenastala žádná změna a také výsledný styl chůze je totožný jako při použití rov. (1), viz obr.2. Výrazný rozdíl však nastal při použití množiny ***B*** produkčních pravidel. Velikost expandovaného prostoru klesla na polovinu, a rovněž délka nelezené cesty je kratší přibližně o 20%. Tato modifikace tedy přináší výhodu především v menším počtu expandovaných stavů a tím i kratší odezvě algoritmu. Styl chůze však zůstává nadále neefektivní, viz obr.2, a je proto zapotřebí předpis hodnotící funkce dále modifikovat. Poznamenejme, že změnou konstant k_i docílíme pouze nalezení odlišné trajektorie, po níž se robot pohybuje k cílovému stavu, nikoliv však výrazné změny počtu stavů tvořících nalezenou cestu.



Obr.3: Optimální styl chůze robotu

Další varianta navržené hodnotící funkce má tvar:

$$h^*(i) = k_1\left(vzd(i) - vzd(j)\right) + k_2\left(dev(i) - dev(j)\right) + k_3 step + h^*(j), \qquad (3)$$

kde stav *i* je přímým následníkem stavu *j*. Tato modifikace hodnotící funkce uvažuje při výpočtu ohodnocení stavu *i* počet pohybů noh uskutečněných na cestě z počátečního stavu s_0 do stavu *i*. Lze tedy předpokládat, že výsledný styl chůze bude obsahovat minimální počet pohybů noh, a tím bude energeticky méně náročný. Hodnotící funkce byla opět testována pro obě množiny uvedených produkčních pravidel. Při použití množiny ***A*** a výchozího stavu,

$\alpha_1 = (1/4)Pi\,[rad]$, bylo expandováno celkem 39826 stavů a nalezené optimální řešení obsahovalo 40 stavů. Konstanty k_i byly stanoveny experimentálně: $k_1 = 1$, $k_2 = 5$, $k_3 = 0,4$. Při použití produkčních pravidel z množiny ***B*** a výchozího stavu, $\alpha_2 = (3/2)Pi\,[rad]$, bylo expandováno 36672 stavů a nalezené řešení obsahovalo pouze 100 stavů, což představuje méně než polovinu stavů oproti hodnotící funkci (2). Pro tento případ byly konstanty k_i stanoveny takto: $k_1 = 1$, $k_2 = 50$, $k_3 = 0,2$. Použití hodnotící funkce *H*, (3), vede na expanzi výrazně většího stavového prostoru, ovšem nalezená cesta obsahuje poloviční počet stavů oproti předchozím návrhům předpisu hodnotící funkce, navíc vyvinutý styl chůze lze považovat za optimální, viz obr.3. V porovnání s předchozím řešením obsahuje nalezené řešení menší množství pohybu noh a produkční pravidla translace robotu v ose Y_r generují posuvy o větší ΔY.

5. Závěr

V příspěvku je navržen způsob automatického generování strategií chůze čtyřnohého robotu. Pro vytváření strategií je úspěšně použito heuristické metody uspořádaného prohledávání (best-first search) stavového prostoru. Tento algoritmus používá pro určení pořadí, ve kterém jsou expandovány nové stavy, hodnotící funkci. Správnost navržených funkcí je ověřena pomocí vytvořeného software, zahrnujícího zjednodušený kinematický model robotu ve 2D s konstantní výškou těla robotu nad rovinným povrchem.. V článku jsou diskutovány celkem tři navržené hodnotící funkce za použití dvou různých množin produkčních pravidel. Testy ukázaly, že je vhodné do hodnotící funkce zahrnout geometrickou vzdálenost aktuálního stavu od cílového stavu a výchylku od ideálního směru. Pro dosažení energeticky výhodného stylu chůze je vhodné do návrhu hodnotící funkce zahrnout počet uskutečněných pohybů noh na cestě z počátečního do aktuálního stavu. Další práce budou zaměřeny na možnost užití jiného algoritmu prohledávání stavového prostoru a řešení úlohy v prostoru.

6. Poděkování

Publikovaných výsledků bylo dosaženo za podpory ministerstva školství, mládeže a tělovýchovy České republiky, výzkumný záměr MSM 0021630518 "Simulační modelování mechatronických soustav".

7. Literatura

Havel I. M., Robotika, SNTL, Praha, 1980

Mařík V., Štěpánková O., Katanský J., a kol., Umělá inteligence 1, Academia, 1993

Ondroušek V., Návrh a implementace koordinačního mechanismu řídicích členů chůze robotu, Diplomová práce, FSI VUT v Brně, 2004

Pearl J., Heuristics, Intelligent Search Strategies for Computer Problem Solving, Addison-Wesley, Reading, Mass., 1984

Řeřucha V., Inteligentní řízení kráčejícího robota, Vojenská akademie v Brně, 1997

Winston P.H., Artifical Intelligence, Addison-Wesley, Reading, Mass., 1992