Štandardná smerová reaktívna navigácia mobilných robotov

Cieľom článku je opísať také základné metódy reaktívnej navigácie mobilných robotov, ktoré predpisujú robotu smer v ďalšom kroku. Medzi tieto základné metódy patria metódy inšpiro-vané prírodou – bug algoritmy, wandering standpoint alebo potenciálové metódy. Článok sa zaoberá stručným opisom týchto metód, pričom sa uvádzajú aj výhody a nevýhody použitia jednotlivých metód.

Úvod

Cieľom reaktívnej navigácie je zabezpečiť správanie robota na základe teórie SMPA (sense – snímaj, map – mapuj, plan – plánuj, act – konaj) [1]. Táto teória pochádza z roku 1984, keď sa skupina vedcov rozhodla zaviesť organizovanú inteligenciu. Tieto inteligentné systémy, teda roboty, majú za úlohu inteligentne reagovať na meniace sa prostredie. To viedlo k vytvoreniu teórie reaktívnej navigácie s využitím umelej inteligencie. Reaktívna navigácia sa odlišuje od plánovanej (globálnej) navigácie v tom, že robot si neplánuje cestu, len reaguje na okamžité zmeny v prostredí v reálnom čase. Požiadavka na výpočet v reálnom čase vedie k minimalizácii výpočtovej náročnosti, preto sú metódy reaktívnej navigácie väčšinou matematicky jednoducho zapísateľné. Voľba metódy reaktívnej navigácie závisí aj od použitých snímačov. Z typu použitých snímačov môže byť odvodená interpretácia informácie zo snímačov (map – mapuj), navigácia (plan – plánuj) a reakcia (act – konaj). Pri aplikáciách reaktívnej navigácie sa predpokladá definovaný štartovací a cieľový bod robota v prostredí alebo sa robot môže v prostredí len potulovať. Ak sú cieľ a štart definované, v riadiacom systéme robota musí existovať nadradený globálny plánovací systém. Metódy reaktívnej navigácie predpisujú správanie robota na základe aktuálnych informácií zo snímačov, ale niektoré metódy vychádzajú aj z lokálnej metrickej mapy, ktorá uchováva informácie o okolí robota v meracom rozsahu snímačov.

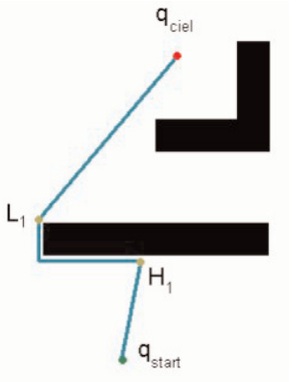
Reaktívna navigácia teda predpisuje správanie robota na základe okamžitej znalosti okolitého prostredia. Predpis správania môže byť vytvorený požiadavkou na smer (smerové reaktívne navigácie) alebo požiadavkou na smer aj rýchlosť (rýchlostné reaktívne navigácie).

1. Bug algoritmy [2] [3]

Sú to jedny z najjednoduchších algoritmov obchádzania prekážok. Bug algoritmy potrebujú poznať polohu robota a cieľa a mať k dispozícii merania zo snímačov vzdialeností. Princíp bug algoritmov spočíva v priamom smerovaní do cieľa, ak robot v tejto ceste nemá prekážku. Verzie jednotlivých algoritmov sa líšia práve v spôsobe obchádzania prekážky. Tieto algoritmy potrebujú presnú informáciu o polohe robota a presne zmerané vzdialenosti k prekážkam. V praxi je toto problém, pretože informácie o polohe robota aj prekážok sú často zašumené.

1.1 Bug 0 [3]

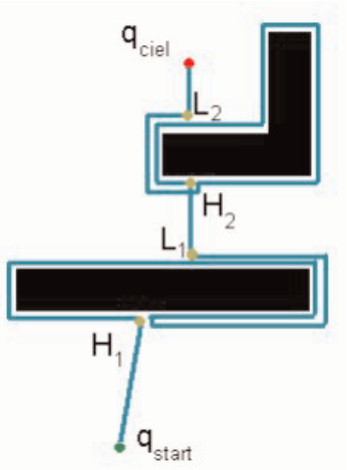
Robot sa pri aplikácii tohto algoritmu vydá zo štartovacej pozície priamo k cieľu. Ak deteguje prekážku, začne ju obchádzať vo vopred zvolenom smere kopírovaním jej obrysov. Ak práve obchádzaná prekážka nepretína spojnicu štartu a cieľa, robot pokračuje priamo v smere tejto spojnice.



Obr.1 Bug 0 algoritmus

1.2 Bug 1 [2] [3] [4]

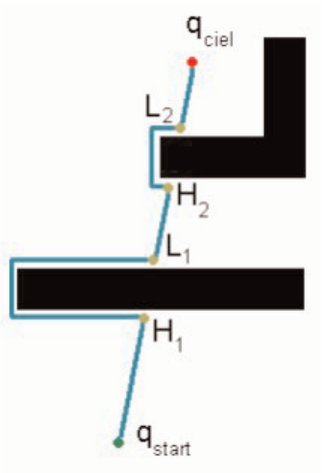
Rovnako ako pri Bug 0 algoritme, aj pri aplikácii tohto algoritmu sa robot vydá k cieľu priamo, kým nenarazí na prekážku. Bod stretu s prekážkou si zapamätá. V tomto bode začne prekážku obchádzať vo zvolenom smere, podobne ako pri Bug 0. Oproti Bug 0 však robot obíde celú prekážku, kým nenarazí opäť na bod, v ktorom začal obchádzať prekážku. Počas obchádzania prekážky robot, resp. riadiaci systém, počíta priamu vzdialenosť k cieľu. Po celom obídení prekážky vyhodnotí bod s najmenšou vzdialenosťou k cieľu. Ďalej vyhodnotí, ktorým smerom sa dostane z bodu stretu s prekážkou do bodu s najkratšou vzdialenosťou k cieľu a presunie sa do tohto bodu. Z neho opäť smeruje priamo k cieľu.



Obr.2 Bug 1 algoritmus

1.3 Bug 2 [2] [3] [4]

Tento algoritmus sa líši od predchádzajúcich v tom, že si počas celého obchádzania prekážok pamätá spojnicu štartu a cieľa. Zo štartovacieho bodu sa začne presúvať smerom k cieľu, kým nenarazí na prekážku. Prekážku začne obchádzať podobne ako v predchádzajúcich variantoch. Obchádza ju však len dovtedy, kým nenarazí na uloženú spojnicu štartu a cieľa. Ak je tento bod bližšie k cieľu ako bod stretu s prekážkou, robot pokračuje po tejto spojnici.



Obr.3 Bug 2 algoritmus

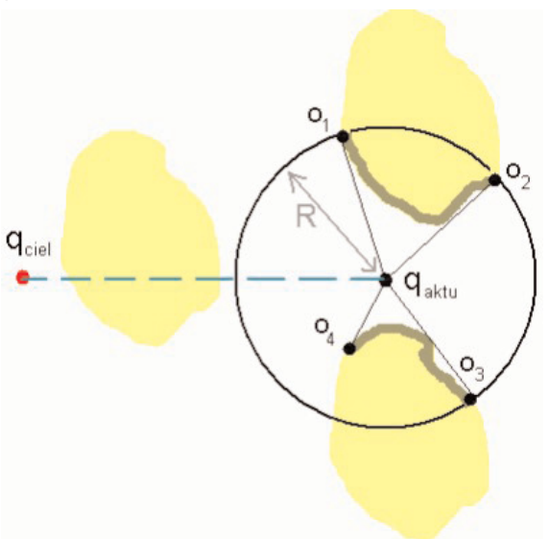
1.4 Bug 1+2 [3]

Táto verzia bug algoritmov sa snaží riešiť nedostatky predchádzajúcich verzií. Algoritmus funguje ako Bug 2, ale len dovtedy, kým robot nenarazí na spojnicu štartu a cieľa v bode, ktorý je ďalej k cieľu ako bod stretu s prekážkou. V tomto prípade dochádza k aplikácii algoritmu Bug 1, ktorý sa snaží nájsť bod opustenia obchádzania prekážky bližší k cieľu ako bod stretu s prekážkou. Tento bod sa potom stáva novým štartovacím bodom a pokračuje sa aplikáciou Bug 2 algoritmu.

1.5 Tangent bug [2] [3] [5]

Tento algoritmus predpokladá existenciu informácie o prekážkach na meracom rozsahu 360°. Ďalej sa uvažuje s obmedzením meracieho rozsahu na dĺžku l. V priestore vymedzenom meracím rozsahom l, teda v lokálnej metrickej mape, je robot schopný určiť časti prekážky o_i, ktoré do tohto priestoru zasahujú.

Pri predpísanom cieli robot najskôr overí, či je na spojnici robota s cieľom vo vymedzenom meracom rozsahu nejaká prekážka. Ak nie, smeruje priamo do cieľa.

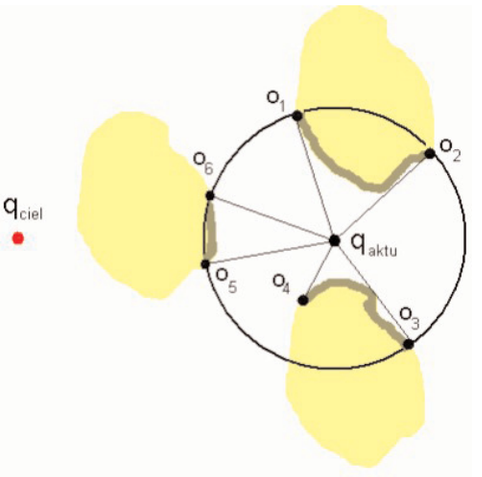


Obr.4 Tangent bug, priame smerovanie do cieľa

Ak sa na spomínanej spojnici objaví prekážka, robot si zo všetkých bodov o_i vyberie taký bod (o_k), pre ktorý platí:

$$h(q_{aktu}, o_k) = \min_i h(q_{aktu}, o_i)$$
$$h(q_{aktu}, o_i) = d(q_{aktu}, o_i) + d(o_i, q_{ciel}) \quad (1)$$

Robot sa v ďalšom kroku začne pohybovať smerom k bodu o_k.



Obr.5 Tangent bug, obchádzanie prekážky

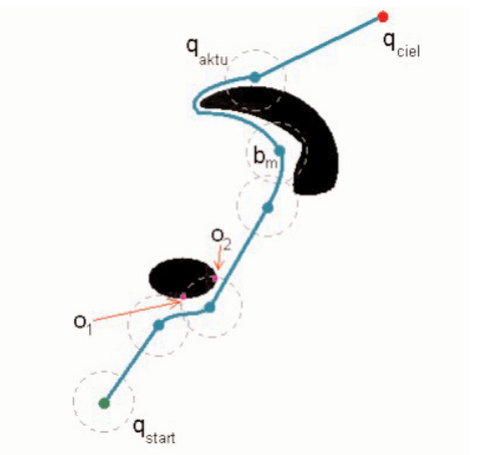
Ak hodnota $h(q_{aktu}, o_k)$ začne narastať, algoritmus sa prepne na metódu Bug 2 a začne prekážku obchádzať sledovaním jej steny. Počas celého pohybu robota v prostredí si algoritmus pamätá najmenšiu vzdialenosť k cieľu od jednotlivých prejdených bodov b_m, ktoré zodpovedajú bodom o_k:

$$d(b_m, q_{ciel}) = \min_j d(b_j, q_{ciel}) \quad (2)$$

Sledovanie okraja prekážky je potom ukončené po splnení podmienky:

$$d(q_{aktu}, q_{ciel}) < d(b_m, q_{ciel}) \quad (3)$$

Ak je táto podmienka splnená, robot môže do cieľa smerovať opäť priamo.

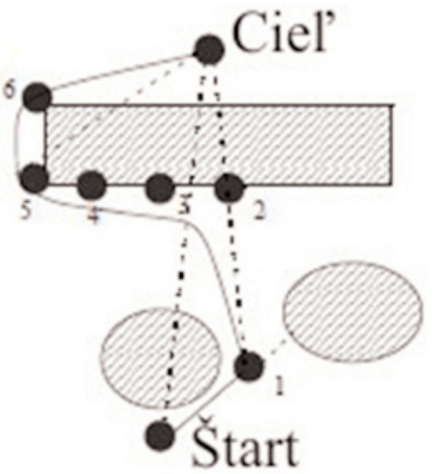


Obr.6 Tangent bug, ukážka prechodu prostredím

2. Wandering Standpoint [6]

Robot sa v prvom kroku pri tomto algoritme snaží dosiahnuť cieľ priamym smerovaním do cieľa. Ak narazí na prekážku v zmysle dosahu svojich snímačov, zmeria veľkosť uhla pre vyhnutie sa vpravo a vľavo od prekážky a zatočí smerom menšieho uhla. Robot potom kopíruje obrysy prekážky dovtedy, kým nie je cesta do cieľa opäť voľná. Príkla-

dom je obrázok. Cieľ nie je dosiahnuteľný z polohy Štart, preto sa robot dostane do bodu 1 kopírovaním obrysu prekážky. V bode 2 je však opäť detegovaná prekážka a robot prepína na režim obchádzania prekážky kopírovaním jej obrysov. Nakoniec je v bode 6 voľná cesta do cieľa. Robot s aplikáciou takéhoto algoritmu môže uviaznuť v lokálnych minimách (napr. prekážka v tvare U).



Obr.7 Wandering standpoint [6]

3. Potenciálové metódy

Tieto metódy vytvárajú potenciálové pole alebo gradient potenciálu na mape prostredia. Pôvodne boli tieto metódy vyvinuté pre robotické manipulátory. Metódy potenciálového poľa považujú robot za bod, ktorý je ovplyvnený umelým potenciálovým polom $U(p)$. Robot sa v prostredí pohybuje pomocou sledovania tohto poľa. Cieľ je definovaný ako minimum v priestore a správa sa ako príťažlivá sila, zatiaľ čo prekážky majú predpísané správanie v podobe odpudivých síl. Superpozíciou všetkých síl aplikovaných na robot vznikne umelé potenciálové pole, ktoré plynule dovedie robot do cieľovej pozície bez kontaktu s prekážkami. Treba si uvedomiť, že tieto metódy aplikované pre reaktívnu navigáciu zlyhávajú v bodoch lokálnych miním. Často nie sú ani optimálne z dôvodu neúplnej znalosti prostredia. Preto sa častejšie využívajú pri globálnom plánovaní dráhy robota.

3.1 Umelé potenciálové pole [4] [7]

Predpokladajme, že robot je v dvojrozmernom priestore definovaný ako bod s orientáciou, teda pomocou (x, y, θ). Ak zavedieme diferenciálovú potenciálovú funkciu $U(p)$, môžeme zadefinovať aj umelú silu $F(p)$ pôsobiacu na pozícii $p=(x, y)$:

$$F(p) = -\nabla U(p) \tag{4}$$

kde $\nabla U(p)$ označuje gradient vektora na pozícii :

$$\nabla U(p) = \nabla U = \begin{bmatrix} \frac{\partial U}{\partial x} \\ \frac{\partial U}{\partial y} \end{bmatrix} \tag{5}$$

Potenciálové pole pôsobiace na robot sa potom vypočíta ako suma príťažlivého potenciálu od cieľa a odpudzujúcich potenciálov od prekážok:

$$U(p) = U_{att}(p) + U_{rep}(p) \tag{6}$$

Podobne môžeme definovať aj sily pôsobiace na robot:

$$F(p) = F_{att}(p) + F_{rep}(p) = -\nabla U_{att}(p) - \nabla U_{rep}(p) \tag{7}$$

Ďalším krokom v definovaní umelého potenciálového poľa je definícia príťažlivého a odpudivého potenciálu. Príťažlivý potenciál môže byť definovaný napríklad takto:

$$U_{att}(p) = \frac{1}{2} k_{att} \rho_{goal}^2(p) \tag{8}$$

kde k_{att} je kladný súčiniteľ a súčiniteľ $\rho_{goal}(p)$ je definovaný ako Euklidovská vzdialenosť $\|p - p_{goal}\|$. Podobne možno definovať aj príťažlivú silu $U(p)$:

$$F_{att}(p) = -\nabla U_{att}(p) = -k_{att} \rho_{goal}(p) \nabla \rho_{goal}(p) = \\ = -k_{att}(p - p_{goal}) \tag{9}$$

Táto sila konverguje k 0, ak sa robot blíži k cieľu.

Hlavnou myšlienkou definície odpudivého potenciálu je vygenerovať odpudivé sily od všetkých známych prekážok. Tento potenciál, resp. sila, by mal byť veľmi veľký v blízkosti prekážok, nemal by však ovplyvňovať pohyb robota vo veľkej vzdialenosti od týchto prekážok. Príkladom takéhoto odpudivého potenciálu môže byť:

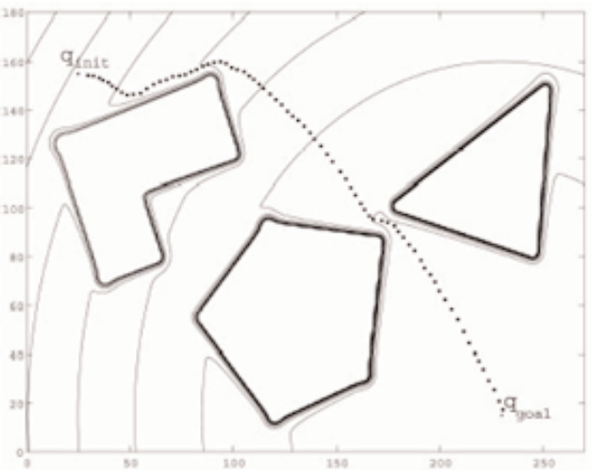
$$U_{req}(p) = \begin{cases} \frac{1}{2} k_{rep} \left(\frac{1}{ń(p)} - \frac{1}{ń_0} \right)^2 & \rho(p) \le \rho(0) \\ 0 & \rho(p) > \rho(0) \end{cases} \tag{10}$$

kde k_{rep} je opäť súčiniteľ, $\rho(p)$ je minimálna kolmá vzdialenosť od pozície robota k objektu a ρ_0 je vzdialenosť vplyvu objektu na dráhu robota. Podobne možno definovať aj odpudivú silu F_{rep}:

$$F_{req}(p) = -\nabla U_{rep}(p) = \\ = \begin{cases} k_{rep} \left(\frac{1}{ń(p)} - \frac{1}{ń_0} \right) \frac{1}{\rho^2(p)} \frac{p - p_{prek}}{\rho(p)} & \rho(p) \le \rho(0) \\ 0 & \rho(p) > \rho(0) \end{cases} \tag{11}$$

kde p_{prek} je pozícia prekážky, ktorá vplýva na dráhu robota.

Výslednú silu môžeme definovať ako súčet príťažlivej a odpudivých síl. Pri ideálnych podmienkach možno nastaviť vektor rýchlosti robota proporčne vzhľadom na vektor výslednej sily a robot by mal smerovať do cieľa bez oscilácií. Avšak, ako už bolo spomenuté, tento prístup je obmedzený vzhľadom na existenciu lokálnych miním vyplývajúcich z tvaru a veľkosti prekážok. Ďalší problém sa objavuje pri výskyte konkávnych prekážok. Pri takejto prekážke sa objaví niekoľko minimálnych vzdialeností $\rho(p)$, čo vedie k oscilácii robota medzi dvoma najbližšími bodmi k objektu.



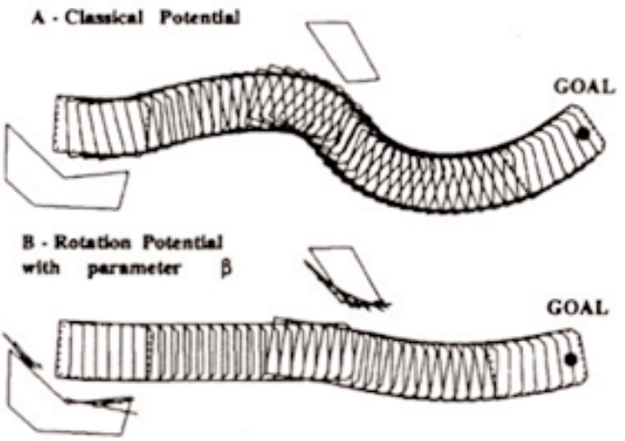
Obr.8 Prechod robota prostredím pomocou metódy umelého potenciálového poľa s vykreslením ekvipotenciálových čiar [4]

3.2 Rozšírené umelé potenciálové pole [4]

Ako všetky potenciálové metódy, aj táto využíva príťažlivé a odpudivé sily. V tejto metóde sú však pridané ďalšie dve potenciálové polia: rotačné a úlohové.

Rotačné potenciálové pole predpokladá, že odpudivá sila je funkciou vzdialenosti od prekážky a orientácie robota vzhľadom na prekážku. To je vyjadrené pomocou súčiniteľa, ktorý zmenšuje odpudivú silu, ak je prekážka paralelne s dráhou robota, čiže táto prekážka neohrozuje robot v zmysle kolízie. Výsledkom je zlepšené sledovanie steny, ktoré nebolo možné dobre vyriešiť pri klasickej metóde umelého potenciálového poľa.

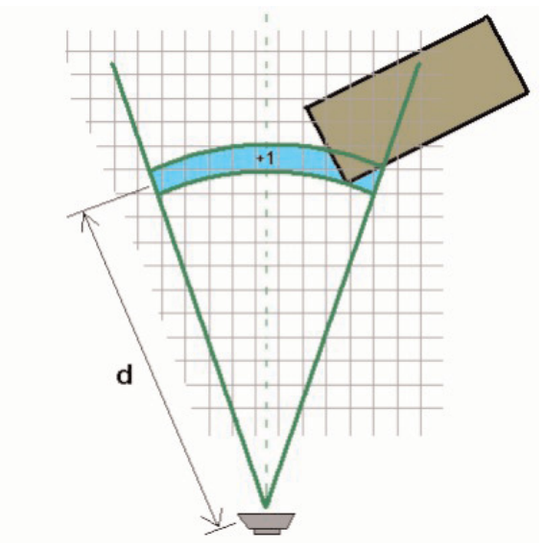
Vstupom do úlohového potenciálového poľa je rýchlosť robota. Toto pole vyfiltruje tie prekážky, ktoré neohrozujú robot v zmysle jeho rýchlosti. Pole definuje sektor pred robotom, na ktorý vplývajú potenciály všetkých prekážok. Ak na tento sektor nevplýva žiadna prekážka, robot nemení smer ani rýchlosť. Výsledkom sú hladšie trajektórie robota v priestore.



Obr.9 Porovnanie metódy klasického a rozšíreného potenciálového poľa [4]

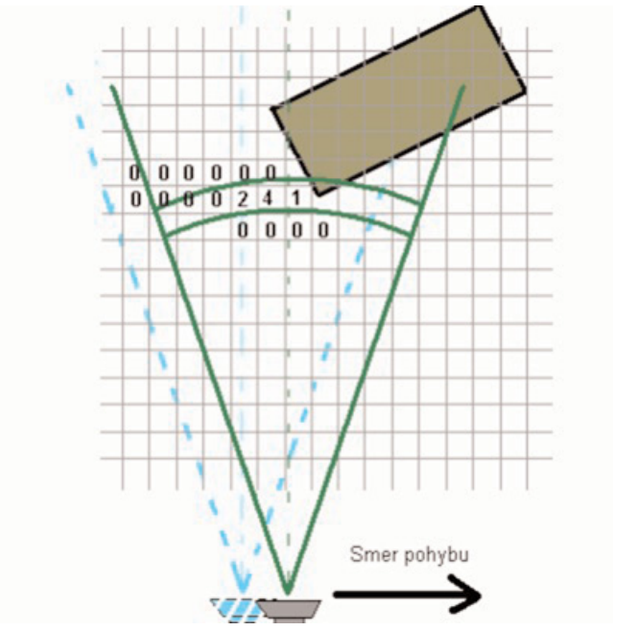
3.3 Pole virtuálnych síl (VFF) [8]

Táto metóda nepoužíva odpudivé sily v zmysle metódy umelého potenciálového poľa. Odpudivé sily sú definované pomocou karteziánskej dvojrozmernej mriežky (t. j. lokálnej metrickej mapy), reprezentujúcej najbližšie okolie robota. Podstatou tvorby tejto mriežky je, že jedno meranie senzora spôsobí zmenu jednej bunky, ktorej hodnota sa inkrementuje o hodnotu jeden. V prípade ultrazvukových snímačov je to bunka na osi snímača v nameranej vzdialenosti.



Obr.10 Interpretácia odmeranej vzdialenosti na bunky

Pri tejto metóde sa vyžaduje vysoká frekvencia vzorkovania hodnôt zo senzorov. Hodnoty buniek musia byť dostatočne „nainkrementované“ na miestach, kde sa v skutočnosti prekážky nachádzajú. V prípade veľkej rýchlosti robota a vysokej vzorkovacej periódy snímania údajov nemožno verne reprezentovať prostredie. Pre danú metódu z toho vyplýva, že vo vzájomnej závislosti je rýchlosť pohybu robota, vzorkovacia frekvencia snímačov a veľkosť jednej bunky použitej mriežky.

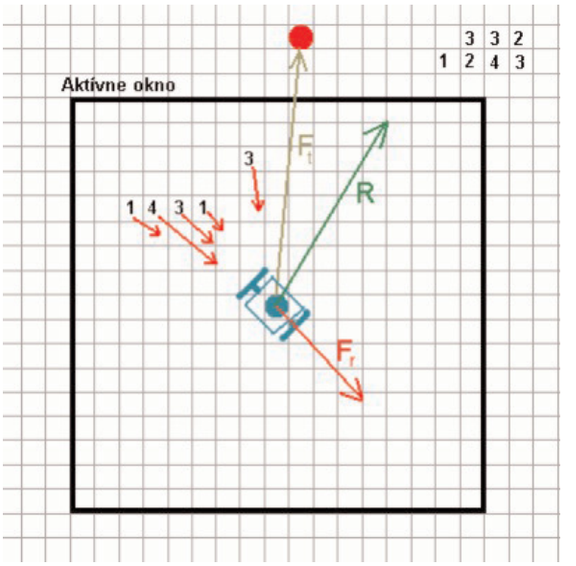


Obr.11 Aktualizácia buniek počas pohybu robota

Robot je počas pohybu sprevádzaný mriežkou s veľkosťou robot sa nachádza uprostred tejto mriežky. Každá bunka, ktorá má nenulovú hodnotu, sa nazýva aktívna bunka a pôsobí na robot odpudivou silou s veľkosťou:

$$F_{i,j} \approx \frac{1}{d_{i,j}^2} \qquad (12)$$

kde $d_{i,j}$ je vzdialenosť aktívnej bunky od robota. Všetky odpudivé sily pôsobiace na robot sa sčítajú do jednej výslednej sily $F_{i,j}$. Výsledná sila udávajúca nový smer robota je potom daná súčtom F_{rep} a F_{att} (sila priťahujúca robot k cieľu). Tu sa teda prejavuje podobnosť s umelým potenciálovým poľom.



Obr.12 Vplyv virtuálnych síl aktívnych buniek na robot

Záver

Štandardné smerové reaktívne navigácie mobilných robotov predstavujú základné metódy pre bezkolízny pohyb robota v prostredí. Tieto metódy sú základným balíkom inteligentnej navigácie mobilného robota v už spoznanom prostredí, pričom práve reaktívna navigácia zabezpečuje bezkolíznosť pohybu či už v spoznanom alebo v neznámom prostredí.

Poďakovanie

Tento článok vznikol pri riešení projektu VEGA 1/0690/09 a KEGA 3/7307/09.

Literatúra

[1] MURPHY, R. R.: Introduction to AI Robotics. Massachusetts Institute of Technology, 2000. ISBN 0-262-13383-0.

[2] CHOSET, H. – LYNCH, K. M. – HUTCHINSON, S. – KANTOR, G. – BURGARD, W. – KAVRAKI, L. E. – THRUN, S.: Principles of Robot Motion (Theory, Algorithms and Implementations). Massachusetts Institute of Technology, 2005. ISN 0-262-03327-5.

[3] PLAKU, E.: CS 336/436 Algorithms for Sensorbased Robotics, Lecture II, III: Bug Path-Planning Algorithms. Department of Computer Science, Laboratory for Computational Sensing and Robotics, Johns Hopkins University, 2010.

[4] SIEGWART, R. – NOURBAKHSH, I. R.: Introduction to Autonomous Mobile Robots. Massachusetts Institute of Technology, 2004. ISBN-13 978-0-262-19502-7.

[5] KAMON, I. – RIMON, E. – RIVLIN, E.: TangentBug: A Range-Sensor-Based Navigation Algorithm. In: The International Journal of Robotics Research, Vol. 17, No. 9, 934-953, 1998. DOI: 10.1177/027836499801700903.

[6] BRÄUNL, T.: Embedded Robotics (Mobile Robot Design and Applications with Embedded Systems). Springer-Verlag Berlin Heidelberg, 2006. ISBN-10 3-540-34318-0.

[7] VIKENMARK, D.: The Obstacle-Restriction Method (ORM) for Reactive Obstacle Avoidance in Difficult Scenarios in Three-Dimensional Workspaces. Master of Science Thesis, Stockholm, Sweden 2006. ISSN-1653-5715.

[8] BORENSTEIN, J. – KOREN, Y.: The Vector Field Histogram – Fast Obstacle Avoidance For Mobile Robots. IEEE Journal of Robotics and Automation Vol 7, No 3, June 1991, pp. 278-288.

prof. Ing. Ladislav Jurišica, PhD.

Ing. František Duchoň PhD.
Ing. Andrej Babinec

Fakulta elektrotechniky a informatiky
Ústav riadenia a priemyselnej informatiky
Ilkovičova 3, 812 19 Bratislava
e-mail: ladislav.jurisica@stuba.sk
frantisek.duchon@stuba.sk
andrej.babinec@stuba.sk