Globálna navigácia mobilných robotov na báze metrickej mapy

František Duchoň, Ladislav Jurišica

Abstrakt
Článok analyzuje súčasný stav poznatkov v oblasti navigácie mobilných robotov na základe známej globálnej metrickej mape. Sú v ňom predstavené teoretické poznatky zo záplavových algoritmov ako sú Brushfire alebo Wavefront, ako aj modifikácia A* algoritmu pre meniace sa zmapované prostredie - D* algoritmus.

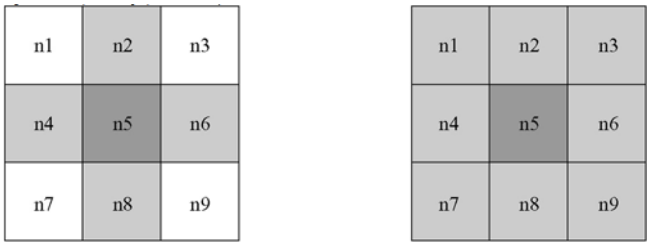
Kľúčové slová: metrická mapa, Brushfire algoritmus, Wavefront algoritmus, D* algoritmus

Úvod

Metrická mapa je jednou z možných foriem reprezentácie prostredia pre mobilné roboty. Metrická mapa vyjadruje metrické vlastnosti objektov v prostredí vzhľadom na vzťah k robotu, teda vzhľadom na jeho polohu. Takéto mapy sú tvorené mriežkovou štruktúrou. Voľba rozlíšenia mriežky, teda veľkosti jednej bunky, závisí od požadovanej presnosti v opozícii s výpočtovou náročnosťou použitej metrickej mapy prostredia. Výhodou použitia metrickej mapy je jej schopnosť opísať prostredie metrickými vlastnosťami, za najväčšiu nevýhodu sa považuje veľká výpočtová náročnosť pri plánovaní cesty. Výskyt objektov v metrickej mape býva na jednotlivých bunkách metrickej mapy vyjadrený pomocou pravdepodobnosti. Predstavené metódy však počítajú s istotou informácie o výskyte objektov v prostredí. Tú je možné dosiahnuť aplikovaním jednoduchého prahovania.

1. Brushfire algoritmus [1]

Existujú dve verzie tohto algoritmu. Prvá vychádza zo 4-susednosti buniek a druhá z 8-susednosti buniek (Obr. 1).

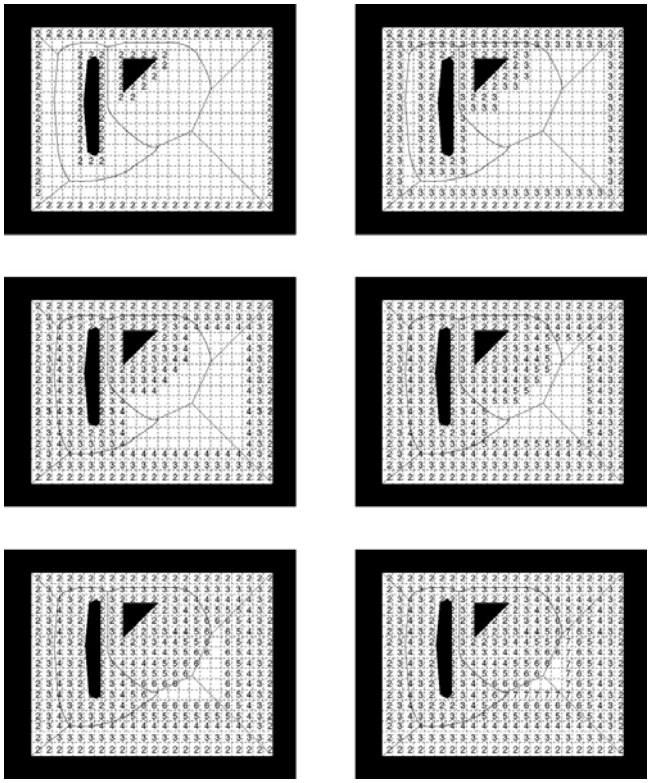


Obr.1 4-susednosť a 8-susednosť [1]

Fig.1 4-connectivity and 8-connectivity

Vstupom do tohto algoritmu je globálna metrická mapa, ktorej bunky majú hodnoty 0 (voľné) alebo 1 (obsadené prekážkou). Výstupom je potom mriežka, kde hodnoty buniek reprezentujú vzdialenosť k najbližšej prekážke. Tieto hodnoty potom môžu byť použité na výpočet potenciálovej funkcie alebo gradientu na určenie optimálnej cesty v prostredí.

V prvom kroku brushfire algoritmu (Obr. 2) sú bunky s nulovou hodnotou susediace s bunkami s hodnotou jedna označené hodnotou dva. Práve tu sa odlišuje rozdiel medzi 4-susednosťou a 8-susednosťou. V ďalšom kroku bunky s nulovou hodnotou susediace s bunkami s hodnotou dva sú označené hodnotou tri. Táto procedúra pokračuje dovtedy, kým nemajú všetky bunky priradenú hodnotu odlišnú od nuly. Princíp pripomína šírenie sa ohňa, teda sú "spaľované" vždy susedné nuly. Vo všeobecnosti môžeme povedať, že bunky s hodnotou nula susedné k bunkám s hodnotou i sú označené hodnotou $i+1$. Gradient vzdialenosti potom možno zostrojiť hľadaním suseda s najnižšou hodnotou. Ak má bunka dve susedné bunky s rovnakou najnižšou hodnotou, jednoducho treba vybrať jednu z nich. Z gradientu potom možno jednoducho určiť odpudivú silu. Aplikáciou príťažlivej sily od cieľa a odpudivej sily od prekážok dostaneme algoritmus podobný metóde umelého potenciálového poľa.

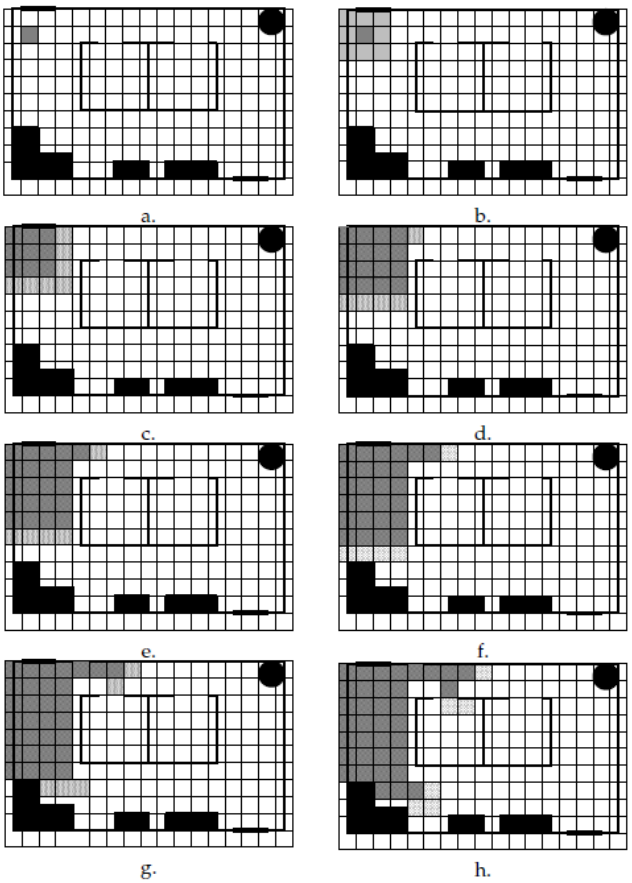


Obr. 2 Algoritmus Brushfire s 8-susednosťou [1]

Fig. 2 Brushfire algorithm with 8-connectivity

2. Wavefront algoritmus [1] [2]

Základný princíp tohto algoritmu spočíva v predstave vodivého priestoru so šírením tepla od štartovacieho bodu po cieľový. Ak existuje cesta medzi týmito dvoma bodmi, "teplo" sa bude tak šíriť, že nakoniec tieto dva body spojí. Inou analógiou je farbenie buniek, kde sa farba rozširuje vždy na susedné bunky. Zaujímavosťou tejto propagácie je, že ako vedľajší efekt je možné vypočítať optimálnu dráhu zo štartovacieho do cieľového bodu. Výsledkom je totižto mapa, ktorá vyzerá ako potenciálové pole.



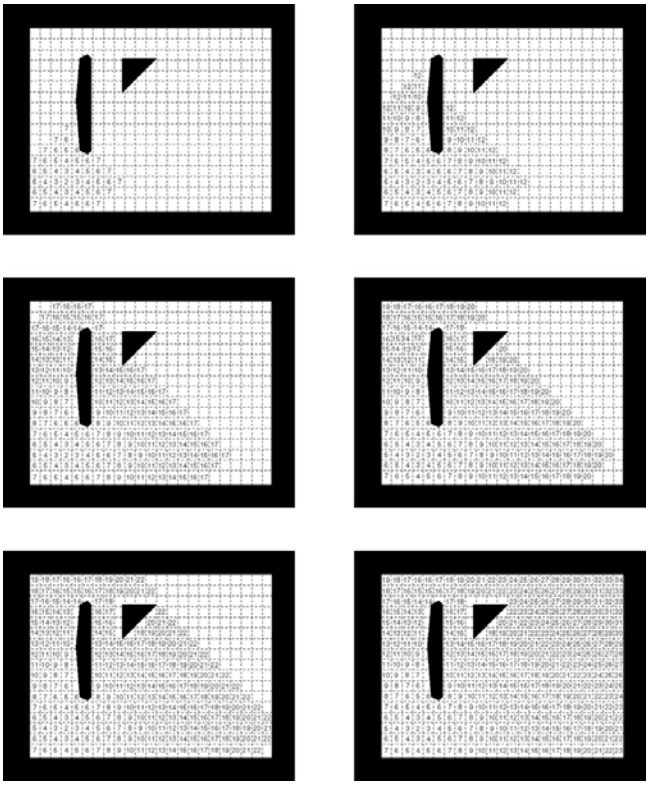
Obr. 3 Wavefront algoritmus. Aktuálne spracovávané elementy sú šedou farbou, staršie "zaplavené" elementy tmavo šedou farbou, prekážky čiernou farbou. [2]

Fig. 3 Wavefront algorithm. Actually processed elemenets have grey colour, older "flooded" elements have darkgrey colour, obstacles have black colour.

Výhodou skupiny algoritmov Wavefront je možnosť zohľadniť do plánovania aj typ terénu. Kým pri prekážke sa predpokladá nulová vodivosť, pre otvorený priestor je táto vodivosť definovaná do nekonečna. Pre rôzne povrchy však možno dodefinovať rôznu vodivosť, a tak je možné robotu predpísať prechod takým prostredím, ktorým aj naozaj prejde. Povrchy s nižšou priechodnosťou pre robot majú teda nižšiu vodivosť. Tento algoritmus tak dokáže prirodzene porovnať dlhšie dráhy s vyššou priechodnosťou a skratky s nižšou priechodnosťou. Je teda využiteľný najmä pri mobilných robotoch do vonkajšieho prostredia.

Princíp výpočtu Wavefront algoritmu je podobný ako pri Brushfire algoritme. Tak isto ako pri Brushfire algoritme, Wavefront algoritmus pracuje nad priestorom globálnej metrickej mapy, kde bunky majú len hodnoty 0 (prázdna) a 1 (obsadená). Rozdiel oproti Brushfire spočíva v tom, že nie je účelom nájsť gradient celého priestoru, ale len cestu medzi cieľom a štartom (Obr. 4). Pozícia štartu a cieľa je teda známa. Bunka obsahujúca cieľ je označená hodnotou 2. V prvom kroku sú všetky bunky s hodnotou 0 susediace s cieľovou bunkou ohodnotené hodnotou 3. V ďalšom kroku sú bunky s hodnotou 0 susediace s bunkami s hodnotou 3 ohodnotené hodnotou 4. Algoritmus sa opakuje dovtedy, kým nie je spojený štart a cieľ. Princíp pripomína vlnu, pričom platí, že všetky bunky na okraji vlny majú rovnakú dĺžku dráhy. Vybraná dráha spočíva vo vybraní dráhy s neustálym poklesom gradientu. Aj keď z jednej bunky môže existovať viacero ciest s poklesom gradientu, z princípu vytvárania Wavefront algoritmu (ohraničenosť voľného priestoru a kontinuita hodnotiacej funkcie) možno povedať, že ku každej bunke bude existovať minimálne jedna susedná bunka s poklesom gradientu.

Pri plánovaní dráhy týmto algoritmom existuje len jedno lokálne minimum, podobne ako pri algoritme Brushfire. Nevýhodou je prílišné približovanie sa k prekážkam.



Obr. 4 Algoritmus Wavefront so 4-susednosťou [1]

Fig. 4 Wavefront algorithm with 4-connectivity.

3. D* algoritmus [1] [2]

Tento algoritmus vychádza z princípu A* algoritmu. Pre každú bunku metrickej mapy je vypočítaná optimálna dráha do cieľa. Tento výpočet je časovo a výpočtovo náročný, avšak predpokladá sa, že prebieha offline ešte pred pohybom robota v priestore. Výpočet všetkých možných ciest do cieľa má výhodu pre reaktívnu navigáciu cez toto prostredie. Ak sa má robot vyhnúť novej prekážke, ktorá sa nevyskytuje v globálnej mape, robot dokáže okamžite pozmeniť dráhu tak, že bude stále optimálna. Pritom výpočet A* algoritmu netreba opäť vykonať pre každú bunku. V podstate možno povedať, že robot si dodefinuje čiastkový cieľový bod. Tento prístup má však aj veľkú nevýhodu. V prípade výskytu veľkého počtu nových prekážok nevyskytujúcich sa v globálnej mape môže dôjsť k zacykleniu sa robota alebo uviaznutiu medzi prekážkami. Riešením tohto problému je neustála aktualizácia globálnej mapy a nový výpočet A* algoritmu. To si však vyžaduje vysoké výpočtové nároky a táto aktualizácia je závislá na presnosti snímačov.

Princíp D* algoritmu možno vysvetliť na obrázku (Obr. 5 až Obr. 16). Robot ma v svojom prostredí pohyblivú prekážku na bunke $(4,3)$. Predpokladajme, že sú to dvere, ktoré sa môžu otvoriť alebo zavrieť. Ak robot považuje tieto dvere za otvorené a počas svojho pohybu zistí, že sú zavreté, pri aplikácií A* algoritmu by musel vypočítať nové potenciálne cesty v prostredí nad celým priestorom medzi štartom a cieľom. Aplikáciou lokálneho prepočítania (teda D* algoritmu) je možné výpočtový čas celej operácie skrátiť.

6	h=6 k=6 b=	h=5 k=5 b=	h=4 k=4 b=	h=3 k=3 b=	h=2 k=2 b=	h=1 k=1 b=	h=0 k=0 Goal
5	h=6.4 k=6.4 b=	h=5.4 k=5.4 b=	h=4.4 k=4.4 b=	h=3.4 k=3.4 b=	h=2.4 k=2.4 b=	h=1.4 k=1.4 b=	h=1 k=1 b=
4	h=6.8 k=6.8 b=	h=5.8 k=5.8 b=	h=4.8 k=4.8 b=	h=3.8 k=3.8 b=	h=2.8 k=2.8 b=	h=2.4 k=2.4 b=	h=2 k=2 b=
3	h=7.2 k=7.2 b=	h=6.2 k=6.2 b=	h=5.2 k=5.2 b=	h=4.2 k=4.2 Gate	h=3.8 k=3.8 b=	h=3.4 k=3.4 b=	h=3 k=3 b=
2	h=7.6 k=7.6 b=	h=6.6 k=6.6 b=	h=5.6 k=5.6 b=	h=5.2 k=5.2 b=	h=4.8 k=4.8 b=	h=4.4 k=4.4 b=	h=4 k=4 b=
1	h=8.0 k=8.0 b=	h=7.0 k=7.0 Start	h=6.6 k=6.6 b=	h=6.2 k=6.2 b=	h=5.8 k=5.8 b=	h=5.4 k=5.4 b=	h=5 k=5 B=
r/c	1	2	3	4	5	6	7

Obr. 5 Inicializácia D* algoritmu [1]

Fig. 5 Inicialization of D* algorithm

Každá bunka je ohodnotená heuristickou hodnotou h, ktorá predstavuje odhad vzdialenosti bunky od cieľa. Táto hodnota pri inicializácii (Obr. 5) nerešpektuje existenciu prekážok v prostredí. Bunky sú ohodnotené aj hodnotou k, ktorá predstavuje odhad najkratšej dráhy k cieľu. Na začiatku výpočtu sú hodnoty h a k pre bunky zhodné. Pre každú bunku sa ukladá aj susedná, z ktorej bola expandovaná - hodnota b. Pri výpočte sa používa rad expandovaných buniek, ktoré budú expandované na základe priority - hodnota k.

h=6 k=6 b=	h=5 k=5 b=	h=4 k=4 b=	h=3 k=3 b=	h=2 k=2 b=	h=1 k=1 b=	h=0 k=0 b=	(7,6) 0 State k
h=6.4 k=6.4 b=	h=5.4 k=5.4 b=	h=4.4 k=4.4 b=	h=3.4 k=3.4 b=	h=2.4 k=2.4 b=	h=1.4 k=1.4 b=	h=1 k=1 b=	
h=6.8 k=6.8 b=	h=5.8 k=5.8 b=	h=4.8 k=4.8 b=	h=3.8 k=3.8 b=	h=2.8 k=2.8 b=	h=2.4 k=2.4 b=	h=2 k=2 b=	
h=7.2 k=7.2 b=	h=6.2 k=6.2 b=	h=5.2 k=5.2 b=	h=4.2 k=4.2 b=	h=3.8 k=3.8 b=	h=3.4 k=3.4 b=	h=3 k=3 b=	
h=7.6 k=7.6 b=	h=6.6 k=6.6 b=	h=5.6 k=5.6 b=	h=5.2 k=5.2 b=	h=4.8 k=4.8 b=	h=4.4 k=4.4 b=	h=4 k=4 b=	
h=8.0 k=8.0 b=	h=7.0 k=7.0 b=	h=6.6 k=6.6 b=	h=6.2 k=6.2 b=	h=5.8 k=5.8 b=	h=5.4 k=5.4 b=	h=5 k=5 b=	

Obr. 6 Expandovanie cieľovej bunky [1]

Fig. 6 Expanded goal cell

h=6 k=6 b(1,6)=	h=5 k=5 b=	h=4 k=4 b=	h=3 k=3 b=	h=2 k=2 b=(6,6)	h=1 k=1 b=(7,6)	h=0 k=0 b=	(6,5) 1.4 (5,7) 2 (5,6) 2 (7,4) 2 (6,4) 2.4 (5,5) 2.4 State k
h=6.4 k=6.4 b(1,5)=	h=5.4 k=5.4 b=	h=4.4 k=4.4 b=	h=3.4 k=3.4 b=	h=2.4 k=2.4 b=(6,6)	h=1.4 k=1.4 b=(7,6)	h=1 k=1 b=(7,6)	
h=6.8 k=6.8 b(1,4)=	h=5.8 k=5.8 b=	h=4.8 k=4.8 b=	h=3.8 k=3.8 b=	h=2.8 k=2.8 b=	h=2.4 k=2.4 b=(7,5)	h=2 k=2 b=(7,5)	
h=7.2 k=7.2 b(1,3)=	h=6.2 k=6.2 b=	h=5.2 k=5.2 b=	h=4.2 k=4.2 b=	h=3.8 k=3.8 b=	h=3.4 k=3.4 b=	h=3 k=3 b=	
h=7.6 k=7.6 b(1,2)=	h=6.6 k=6.6 b=	h=5.6 k=5.6 b=	h=5.2 k= b=	h=4.8 k=4.8 b=	h=4.4 k=4.4 b=	h=4 k=4 b=	
h=8.0 k=8.0 b(1,1)=	h=7.0 k=7.0 b=	h=6.6 k=6.6 b=	h=6.2 k=6.2 b=	h=5.8 k=5.8 b=	h=5.4 k=5.4 b=	h=5 k=5 b=	

Obr. 7 Expandovanie buniek $(6,6)$ a $(7,5)$ [1]

Fig. 7 Cells $(6,6)$ and $(7,5)$ are expanded.

Bunky sú expandované, až kým algoritmus nenarazí na bunku $(4,6)$ (Obr. 8). Jej susedmi sú bunky $(3,6)$ a $(3,5)$, ktoré predstavujú prekážky. Hodnoty h sú pre tieto bunky zvýšené o nejakú konštantne vysokú hodnotu tak, aby bolo zrejmé, že sú to prekážky.

h=6 k=6 b(1,6)=	h=5 k= b=	h=10003 k=4 b=(4,6)	h=3 k=3 b=(5,6)	h=2 k=2 b=(6,6)	h=1 k=1 b=(7,6)	h=0 k=0 b=	(7,3) 3 (6,3) 3.4 (4,5) 3.4 (5,3) 3.8 (4,4) 3.8 (3,6) 4 (4,3) 4.2 (3,5) 4.4 State k
h=6.4 k=6.4 b(1,5)=	h=5.4 k= b=	h=10003.4 k=4.4 b=(4,6)	h=3.4 k=3.4 b=(5,6)	h=2.4 k=2.4 b=(6,6)	h=1.4 k=1.4 b=(7,6)	h=1 k=1 b=(7,6)	
h=6.8 k=6.8 b(1,4)=	h=5.8 k= b=	h=4.8 k= b=	h=3.8 k=3.8 b=(5,5)	h=2.8 k=2.8 b=(6,5)	h=2.4 k=2.4 b=(7,5)	h=2 k=2 b=(7,5)	
h=7.2 k=7.2 b(1,3)=	h=6.2 k= b=	h=5.2 k= b=	h=4.2 k=4.2 b=(5,4)	h=3.8 k=3.8 b=(6,4)	h=3.4 k=3.4 b=(7,4)	h=3 k=3 b=(7,4)	
h=7.6 k=7.6 b(1,2)=	h=6.6 k=6.6 b=	h=5.6 k=5.6 b=	h=5.2 k=5.2 b=	h=4.8 k=4.8 b=	h=4.4 k=4.4 b=	h=4 k=4 b=	
h=8.0 k=8.0 b(1,1)=	h=7.0 k=7.0 b=	h=6.6 k=6.6 b=	h=6.2 k=6.2 b=	h=5.8 k=5.8 b=	h=5.4 k=5.4 b=	h=5 k=5 b=	

Obr. 8 Expandovanie bunky $(4,6)$ [1]

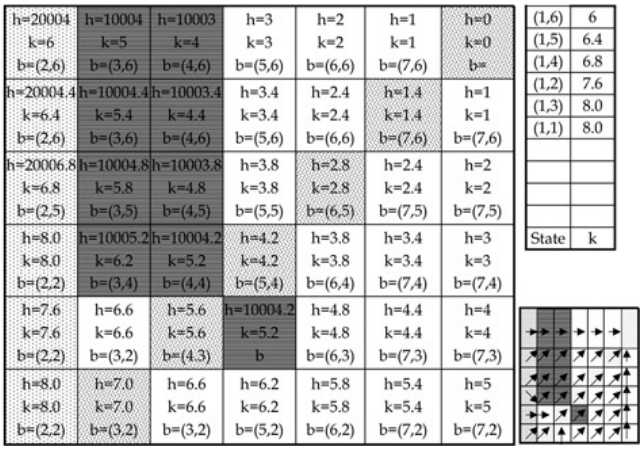
Fig. 8 Cell $(4,6)$ is expanded

Prehľadávanie mapy končí vtedy, keď je expandovaná štartovacia bunka $(2,1)$ (Obr. 9). Na konci výpočtov je rad expandovaných buniek naplnený bunkami $(1,1)$, $(1,2)$, $(1,3)$, $(1,4)$ a $(1,6)$. Optimálnu dráhu potom možno nájsť pomocou spätného prehľadávania ukazovateľov na susedné bunky (Obr. 10).

h=20004 k=6 b=(2,6)	h=10004 k=5 b=(3,6)	h=10003 k=4 b=(4,6)	h=3 k=3 b=(5,6)	h=2 k=2 b=(6,6)	h=1 k=1 b=(7,6)	h=0 k=0 b=	(1,6) 6 (1,5) 6.4 (1,4) 6.8 (1,2) 7.6 (1,3) 8.0 (1,1) 8.0 State k
h=20004.4 k=6.4 b=(2,6)	h=10004.4 k=5.4 b=(3,6)	h=10003.4 k=4.4 b=(4,6)	h=3.4 k=3.4 b=(5,6)	h=2.4 k=2.4 b=(6,6)	h=1.4 k=1.4 b=(7,6)	h=1 k=1 b=(7,6)	
h=20006.8 k=6.8 b=(2,5)	h=10004.8 k=5.8 b=(3,5)	h=10003.8 k=4.8 b=(4,5)	h=3.8 k=3.8 b=(5,5)	h=2.8 k=2.8 b=(6,5)	h=2.4 k=2.4 b=(7,5)	h=2 k=2 b=(7,5)	
h=8.0 k=8.0 b=(2,2)	h=10005.2 k=6.2 b=(3,4)	h=10004.2 k=5.2 b=(4,4)	h=4.2 k=4.2 b=(5,4)	h=3.8 k=3.8 b=(6,4)	h=3.4 k=3.4 b=(7,4)	h=3 k=3 b=(7,4)	
h=7.6 k=7.6 b=(2,2)	h=6.6 k=6.6 b=(3,2)	h=5.6 k=5.6 b=(4,3)	h=10004.2 k=5.2 b=(5,3)	h=4.8 k=4.8 b=(6,3)	h=4.4 k=4.4 b=(7,3)	h=4 k=4 b=(7,3)	
h=8.0 k=8.0 b=(2,2)	h=7.0 k=7.0 b=(3,2)	h=6.6 k=6.6 b=(3,2)	h=6.2 k=6.2 b=(5,2)	h=5.8 k=5.8 b=(6,2)	h=5.4 k=5.4 b=(7,2)	h=5 k=5 b=(7,2)	

Obr. 9Ukončenie D* algoritmu [1]

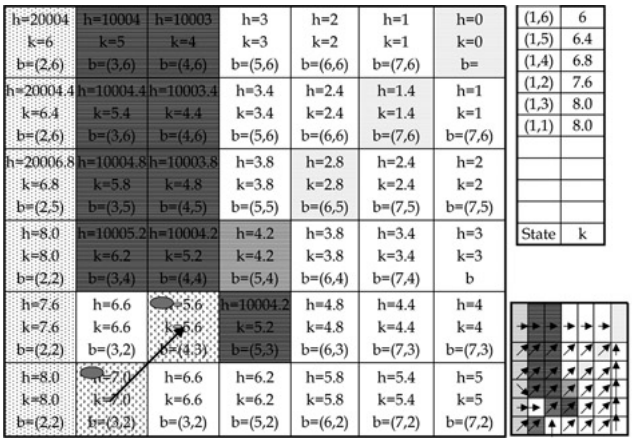
Fig. 9 Termination of D* algorithm



Obr. 10 Nájdenie optimálnej cesty pomocou D* algoritmu [1]

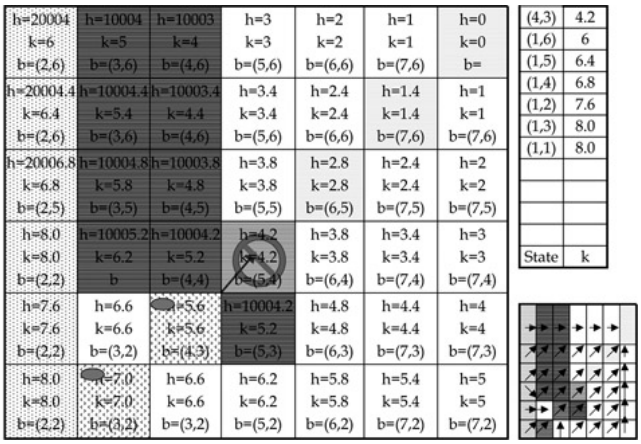
Fig. 10 Optimal path found by D* algorithm

Ak robot prechádza po predpísanej dráhe, pohybuje sa z bunky $(2,1)$ na bunku $(3,2)$ (Obr. 11). Ak by boli otvorené dvere, snažil by sa z bunky $(3,2)$ dostať na bunku $(4,3)$, predpokladajme však, že ich niekto zavrel (Obr. 12). V tomto prípade nie je nutné preplánovať všetky cesty pomocou A* algoritmu, postačí vloženie bunky $(4,3)$ do radu expandovaných buniek (Obr. 13).



Obr. 11 Robot začne sledovať vygenerovanú optimálnu dráhu [1]

Fig. 11 Robot is followign generated path

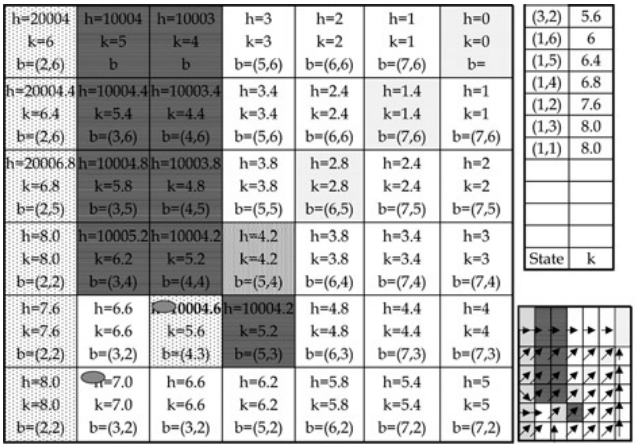


Obr. 12 Robot pomocou snímačov zistil, že bunka $(4,3)$ je nepriechodná [1]

Fig. 12 Cell $(4,3)$ is blocked

Treba si všimnúť, že táto bunka musí mať oproti ostatným v rade najmenšiu hodnotu k (teda najvyššiu prioritu), keďže prehľadávanie sa uskutočňuje len medzi štartom a cieľom.

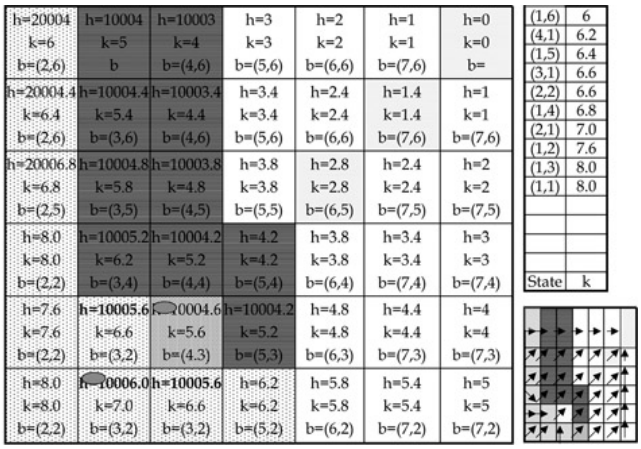
Zákonite tak bunka $(4,3)$ musí mať menšiu hodnotu k ako štart a teda aj ostatné bunky v rade expandovaných buniek. Základnou myšlienkou je potom pokračovanie v expandovaní buniek, ktoré zmenili svoj stav. V tomto prípade je od bunky $(4,3)$ expandovaná bunka $(3,2)$ a jej hodnota h je odvodená od hodnoty bunky $(4,3)$ a konštantnej pokutovej hodnoty, v tomto prípade 10000,4.



Obr. 13 Expandovanie bunky $(4,3)$ [1]

Fig. 13 Cell $(4,3)$ is expanded

Ďalšou expandovanou bunkou je bunka $(3,2)$ (Obr. 14), pretože jej hodnota k je najmenšia v rade expandovaných buniek. Hodnota k tejto bunky je menšia ako hodnota h. Takéto bunky sa nazývajú bunky s vybudeným stavom. Ak sa nejaká bunka nachádza v takomto stave, potom hodnota b, teda ukazovateľ na bunku, z ktorej bola expandovaná, musí byť zmenená. Expanziou bunky $(3,2)$ sa v rade expandovaných buniek objavia bunky $(2,2)$, $(2,1)$, $(3,1)$ a $(4,1)$. Hodnoty pre prvé tri z nich odrážajú skutočnosť, že cesta z nich do cieľa prechádzala dverami. Táto skutočnosť sa neprejavuje pri bunke $(4,1)$, a preto jej hodnota h zostáva nemenná.

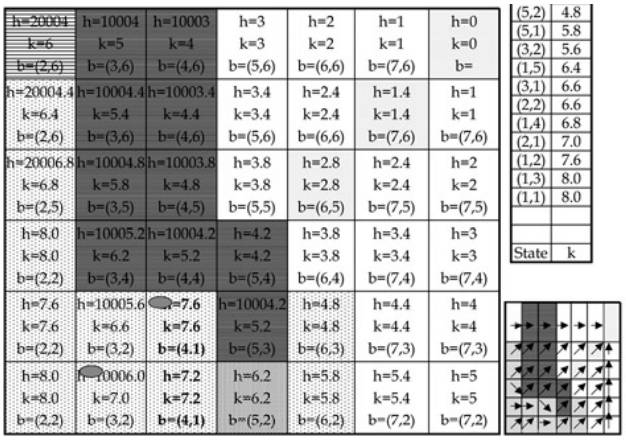


Obr. 14 Expandovanie bunky $(3,2)$ [1]

Fig. 14 Cell $(3,2)$ is expanded

V ďalšom kroku je expandovaná bunka $(1,6)$, ale tá neovplyvňuje hodnotu h svojich susedov. Ďalšou expandovanou bunkou je $(4,1)$ (Obr. 15). Do radu expandovaných buniek pribudnú bunky $(3,2)$, $(3,1)$, $(5,1)$ a $(5,2)$. Hodnoty h pre bunky $(5,1)$ a $(5,2)$ zostanú nemenné. Hodnoty pre bunky

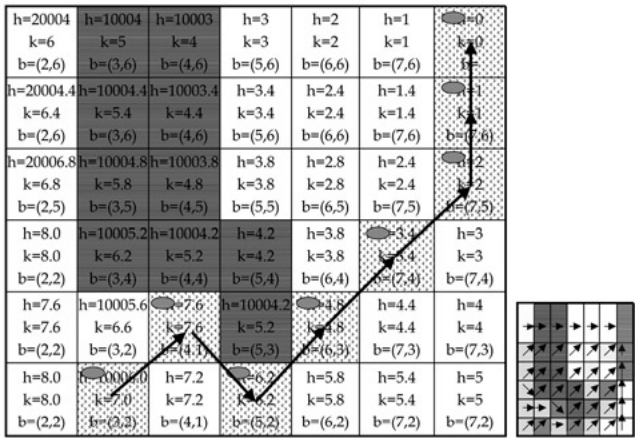
$(3,2)$ a $(3,1)$ sa zmenia, z dôvodu, že sú odvodené od bunky $(4,1)$.



Obr. 15 Expandovanie bunky $(4,1)$ [1]

Fig. 15 Cell $(4,1)$ is expanded

Aktuálna štartovacia bunka $(3,2)$ už teda neobsahuje ukazovateľ na zavreté dvere, ale na bunku $(4,1)$ a je teda možné preplánovať cestu do cieľa (Obr. 16).



Obr. 16 Nájdenie lokálne modifikovanej optimálnej cesty pomocou D* algoritmu [1]

Fig. 16 Modificated optimal path found by D* algorithm

Záver

Napriek nesporným výhodám použitia metrických máp v globálnej navigácii mobilných robotov sa ukazuje ako ich veľká nevýhoda vysoká výpočtová a pamäťová náročnosť. Možno to pozorovať aj pri prezentovaných metódach, kde je nutné pre každú bunku uchovávať informácie nielen o stave bunky (prekážka, voľný priestor), ale aj pracovať s niekoľkými hodnotami nad bunkami. Výsledkom však je metricky ohodnotená optimálna cesta. Použitie týchto metód však zvyšuje nároky na výpočtový systém mobilného robota.

Poďakovanie

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

Literatúra

[1] 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.

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

Abstract

This paper analysis current state in navigation of mobile robots on basis of known global metric map. There is also presented theoretical knowledge from flooding algorithms, such as Brushfire and Wavefront, as well as modification of A* algorithm for changing and mapped environment - D* algorithm.

prof. Ing. Ladislav Jurišica, PhD.,
Ing. František Duchoň, PhD.

Fakulta elektrotechniky a informatiky
Ústav riadenia a priemyselnej informatiky
Ilkovičova 3
812 19 Bratislava
ladislav.jurisica@stuba.sk, frantisek.duchon@stuba.sk