PLANOWANIE TRASY - Metody Mapy Dróg

[A.R. Wolczowski, Self-correcting trajectory planning using modified visibility graph, Proc. of 6th IFAC Symp. SYROCO' 00, Vienna 2000, Vol. 2, Pergamon, 2001, pp. 611-616]

- Metoda Grafu Widzialności (Visibility Graph)
- Planowanie trasy auto-korekcyjnej w oparciu o VG

Metoda Grafu Widzialności (Visibility Graph)

Metoda Grafu Widzialności opiera się na odwzorowaniu spójności części swobodnej (C_{free}) przestrzeni konfiguracyjnej (C) robota, na siatkę 1-wymiarowych krzywych zwanych mapą dróg R. Graf Widzialności (VG) jest grafem nieskierowanym, którego wierzchołkami są konfiguracja początkowa q_{init} i docelowa q_{goal} robota oraz wszystkie wierzchołki C-przeszkód, a krawędziami odcinki proste łączące wszystkie „widzące się" wierzchołki.
Efektywne algorytmy konstruują graf VG oraz znajdują w nim najkrótszą ścieżkę w czasie $O(n^2)$, gdzie n liczba wszystkich wierzchołków C-przeszkód. Ścieżka taka stanowi linię łamaną przechodzącą przez wierzchołki przeszkód.

Model przestrzeni roboczej

Metody mapy dróg operują na dwóch modelach przestrzeni roboczej:
- pierwotnym – geometrycznym, opisującym dopuszczalne pozycje robota, i
- wtórnym – sieciowym, opisującym dopuszczalne sekwencje pozycji robota podczas jego ruchu w otoczeniu.

Obydwa modele bazują na przestrzeni konfiguracyjnej $C = \mathbf{R}^2$. *Model pierwotny* opiera się na opisie kształtu i położenia przeszkód w przestrzeni C. W rozpatrywanej metodzie przeszkody są ograniczone wielobokami. Suma wnętrz wieloboków reprezentuje nieprzejezdny obszar przestrzeni C, a pozostała przestrzeń swobodna C_{free} (wraz z domknięciem $cl(C_{free})$), stanowi obszar ruchu robota.

W *modelu wtórnym* odcinki proste łączące wyróżnione punkty przestrzeni C, przechodzące w C_{free} lub jej domknięciu $cl(C_{free})$, reprezentują sieć dróg R.

Model pierwotny

Odwzorowane do przestrzeni C przeszkody można przedstawić jako elementy zbioru:

$$O = \{o_1, o_2, ..., o_n\} \tag{1}$$

Wierzchołki przeszkód są wyróżnionymi punktami przestrzeni C. Każdą przeszkodę wieloboczną można opisać zbiorem jej wierzchołków:

$$o_i \leftrightarrow Q_i = \{v_i, v_{i+1}, ..., v_{i+r}\}, \quad v_s = q_j(x,y) \in C \tag{2}$$

Liczba ***r*** reprezentuje liczność ***card*** (cardinality) zbioru Q_i.
Kolejne indeksy $i+k$, $i+k+1$ (dla $0 \leq k < r$) oraz indeksy $i+k$, i (dla $k=r$) opisują sąsiadujące ze sobą wierzchołki przeszkody o_i (vide rys. 2.).

Wierzchołki wszystkich przeszkód odwzorowanych do przestrzeni C tworzą zbiór:

$$Q = \bigcup_i Q_i, \quad Q_i \leftrightarrow o_i \in O \tag{3}$$

Krawędzie przeszkód są odcinkami prostymi łączącymi sąsiadujące wierzchołki w każdej z przeszkód i są reprezentowane przez pary połączonych wierzchołków:

$$s_{i,i+1} = (v_{i+j}, v_{i+k}) \in S_i \subset Q_i \times Q_i \tag{4}$$

Indeksy wierzchołków opisujących krawędzie spełniają zależność:

$$i + k = i + (j + 1) \bmod (r + 1) \tag{5}$$

gdzie: $(r+1) = card(Q_i)$.

Model wtórny

Krawędzie przeszkód stanowią w modelu wtórnym odcinki dróg położone w $cl(C_{free})$. Odcinki takie tworzą zbiór S_O:

$$S_O \subseteq \bigcup_i S_i, \quad S_i \leftrightarrow o_i \in O \tag{6}$$

Odcinki proste łączące dopuszczalne pary wierzchołków różnych przeszkód (tj. nieprzecinające krawędzi ze zbioru S_O) reprezentują odcinki dróg położone w C_{free}:

$$s_{i+n,\, j+m} = (v_{i+n}, v_{j+m}) \in S_{F1} \subseteq Q \times Q \tag{7}$$

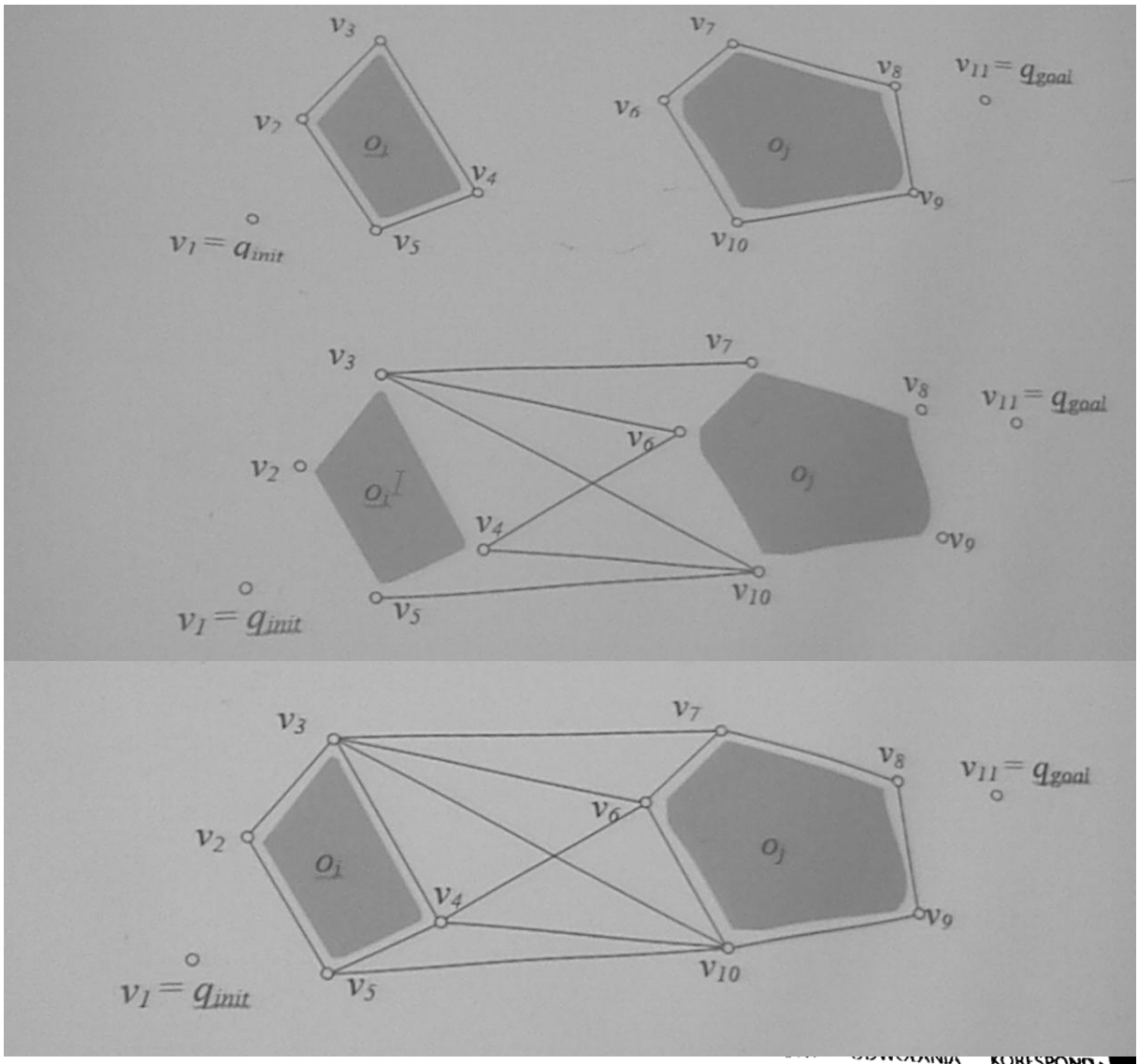
Odcinki proste łączące dopuszczalne pary wierzchołków różnych przeszkód (tj. nieprzecinające krawędzi ze zbioru S_O) reprezentują odcinki dróg położone w C_{free}:

$$s_{i+n,\, j+m} = (v_{i+n}, v_{j+m}) \in S_{F1} \subseteq Q \times Q \tag{7}$$

Odcinki takie tworzą zbiór S_{F1}. Odcinki dróg reprezentowane elementami zbiorów S_O i S_{F1} tworzą sieć dróg R:

$$R \leftrightarrow S_O \cup S_{F1} \tag{8}$$

Budowę przeszkód oraz opartych na nich elementów sieci dróg ilustruje rys. 2.



Rys. 2. Ilustracja elementów pierwotnego i wtórnego modelu przestrzeni roboczej.

Sieć R stanowi punkt wyjścia dla określenia Grafu Widzialności VG. W ujęciu formalnym graf taki jest trójką:

$$\langle Q^*, S^*, F^* \rangle, \tag{9}$$

gdzie: Q^* - zbiór wierzchołków;
S^* - zbiór krawędzi grafu;
F^* - funkcja przyporządkowująca krawędziom grafu koszt związany z ich długością.

Wierzchołki grafu odpowiadają wierzchołkom przeszkód oraz konfiguracji początkowej i końcowej robota:

$$Q^* = Q \cup \{q_{init}, q_{goal}\} \tag{10}$$

Odcinki proste łączące punkty startu i celu z wierzchołkami przeszkód tworzą zbiór dodatkowych odcinków dróg położonych w C_{free}:

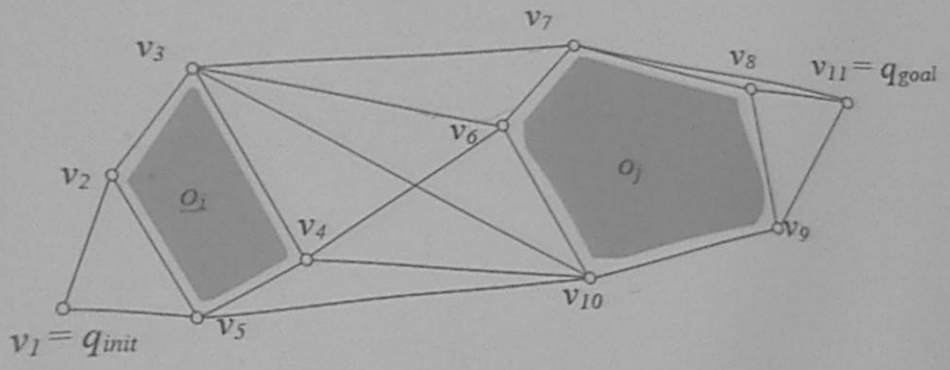
$$S_{F0} \subset \{q_{init}, q_{goal}\} \times (Q \cup \{q_{init}, q_{goal}\}) \quad (11)$$

Zbiór wszystkich odcinków położonych w C_{free} jest sumą zbiorów:

$$S_F = S_{F1} \cup S_{F0} \quad (12)$$

Krawędzie grafu odpowiadają elementom siatki dróg R oraz dodatkowym odcinkom

Graf *VG* zbudowany na przeszkodach z rys. 2, ilustruje rys. 3.



Rys. 3. Graf Widzialności.

Problemy z realizacją praktyczną

Sterowanie ruchem robota mobilnego w ***układzie otwartym*** jest zawsze obarczone błędem pozycji. Błąd ten wynika:

(a) z pominięcia w modelach otoczenia i robota trudnych do uwzględnienia czynników takich jak nierówności terenu, poślizgi kół, oraz

(b) z błędu sterowania ruchem kół.

Błąd pozycji ma charakter kumulacyjny (wzrasta wraz z przebytą odległością) i może uniemożliwić osiągnięcie celu działania. Dlatego sterowanie lokomocją na podstawie wcześniej opracowanego planu wymaga wyznaczania na bieżąco pozycji robota.

Pozycję robota można wyznaczyć na podstawie pomiaru ruchu jego kół. Jednak nawigacja zaliczeniowa jest obarczona podobnym błędem jak sterowanie ruchem w układzie otwartym i pozwala jedynie zmniejszyć niepewność pozycji powodowaną przez błąd sterowania napędem kół.

Z kolei samolokalizacja robota na podstawie oglądu otoczenia, wymaga

a także ruchu robota w trakcie pomiarów, oszacowana pozycja jest nadal obarczona istotnym błędem. Może to prowadzić do kolizji.

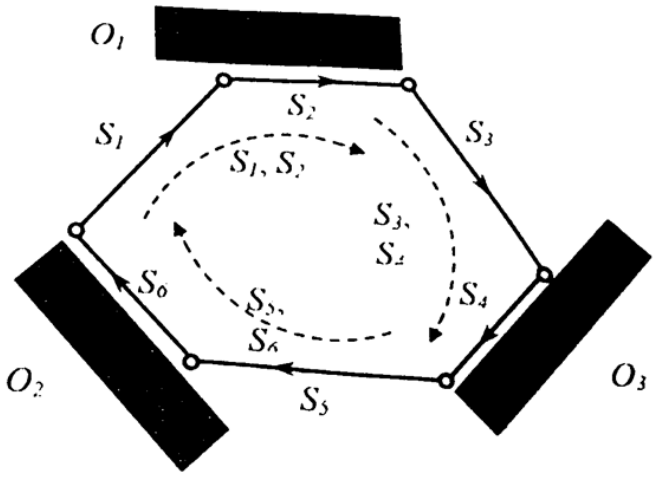
Nawigacja auto-korekcyjna: Koncepcja metody

Podczas działania w otoczeniu wewnętrznym robot mobilny z konieczności często porusza się wzdłuż korytarzy lub ścian obiektów. Na takich odcinkach trasy, podczas „ruchu wzdłuż ściany", można zastosować ***sterowanie lokalne*** zapewniające utrzymanie stałego dystansu do ściany. Taki ruch ma charakter ruchu podatnego – dopasowującego się elastycznie do cech geometrycznych otoczenia.

Sterowanie lokalne nie wymaga dużej mocy obliczeniowej i może korzystać z tanich sensorów niskiego poziomu (takich jak: optyczne sensory zbliżeniowe czy sonary).

Proponowane rozwiązanie polega na celowym planowaniu ścieżki robota w znanym otoczeniu w taki sposób, aby następnie można było ją realizować wyłącznie w oparciu o sterowanie lokalne, bez śledzenia globalnej pozycji robota. Taka ścieżka została określona mianem ***trasy autokorekcyjnej***

sterowanie otwarte. Podczas realizacji każdej kolejnej pary odcinków, błąd pozycji najpierw wzrasta podczas sterowania otwartego, a następnie ulega skorygowaniu podczas ruchu wzdłuż ściany. Zasadę korekcji pozycji ilustruje rys. 1.



Realizacja

Do planowania trasy autokorekcyjnej można wykorzystać *Zmodyfikowany Graf Widzialności*.

Proponowana modyfikacja polega na transformacji grafu VG na taki graf skierowany MVG, którego łuki leżące w C_{free} nie prowadzą do wierzchołków przeszkód lecz do nowych wierzchołków usytuowanych na ich krawędziach.

Graf MVG ma taką właściwość, że co najmniej co drugi odcinek każdej poprowadzonej w nim ścieżki leży w domknięciu przestrzeni swobodnej $cl(C_{free})$, czyli przebiega wzdłuż krawędzi przeszkody. Podczas realizacji działania robota, na tych odcinkach można zastosować sterowanie lokalne, oparte na bezpośrednim sprzężeniu zwrotnym, zapewniające prowadzenie robota wzdłuż ściany.

Niepewność pozycji

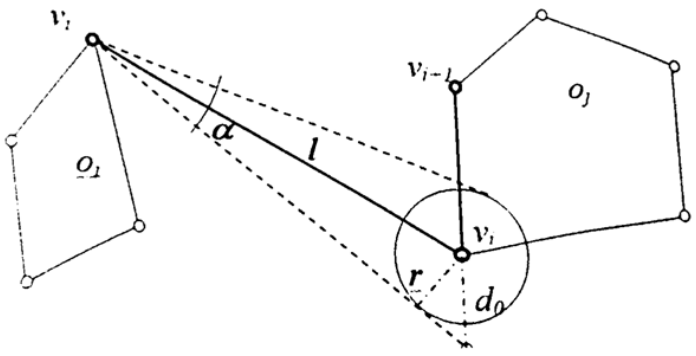
Niepewność położenia robota podczas ruchu po odcinkach drogi typu $s \in S_F$, położonych w przestrzeni swobodnej C_{free} można w uproszczeniu wyrazić jako

okrąg o promieniu r proporcjonalnym do przebytego dystansu l:

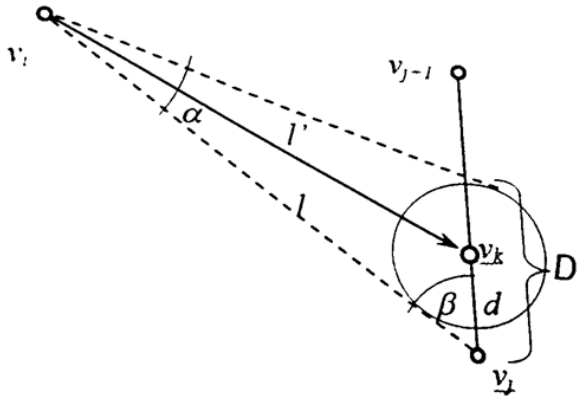
$$r = l \sin \alpha \tag{14}$$

gdzie: α - współczynnik wyznaczony eksperymentalnie.

Przy ruchu z wierzchołka v_i w kierunku wierzchołka v_j, położonego na końcu krawędzi (v_j, v_{j-1}), w najgorszym przypadku robot może minąć punkt v_j w odległości euklidesowej r, lub odległości d_0 liczonej wzdłuż mijanej krawędzi (rys. 4.).



$$d = l \frac{\sin\alpha}{\sin\alpha + \beta} \qquad (16)$$



Rys. 5. Sposób obliczania kierunku ruchu uwzględniający niepewność działania.

- ruchem wzdłuż krawędzi (v_j, v_{j-1}) do wierzchołka v_j.

Przyjmijmy, że krawędź (v_j, v_{j-1}) jest nachylona do kierunku ruchu (czyli do krawędzi (v_i, v_j)) pod kątem $\beta = \angle(v_i, v_j)(v_j, v_{j-1})$. Aby zagwarantować „trafienie" w krawędź, jej długość powinna przekraczać „szerokość" strefy błędu D liczoną wzdłuż krawędzi:

$$\left\| v_j, v_{j+1} \right\| > D = l \frac{\sin 2\alpha}{\sin(2\alpha + \beta)} \qquad (15)$$

Ruch w kierunku wierzchołka v_j należy zastąpić ruchem w kierunku nowego wierzchołka v_k, położonego na krawędzi w odległości d od wierzchołka v_j (rys. 5.):

$$d = l \frac{\sin\alpha}{\sin\alpha + \beta} \qquad (16)$$

Algorytm budowy Zmodyfikowanego Grafu Widzialności

Punktem wyjścia dla określenia grafu MVG jest graf VG oraz (pochodzący z pierwotnego modelu przestrzeni roboczej) zbiór S_O krawędzi przeszkód. W przypadku braku jawnej postaci zbioru S_O, można go odtworzyć na podstawie zbiorów wierzchołków przeszkód:

$$S_O = \{(v_i, v_j) \in S^* / (\exists o_k \in O)(v_i, v_j \in Q_k \leftrightarrow o_i)\} \qquad (17)$$

Kolejne kroki algorytmu transformacji VG w MVG przedstawiają się następująco.

Krok 1. Na zbiorze S^* krawędzi grafu VG określić podzbiór S'_F krawędzi położonych w przestrzeni swobodnej C_{free}, z pominięciem krawędzi prowadzących do q_{goal}:

$$S'_F = S^* \setminus \left(S_O \cup (Q \times \{q_{goal}\})\right) \tag{18}$$

Krok 2. Określić zbiór P par takich incydentnych krawędzi, że pierwsza krawędź leży w C_{free}, a druga w $cl(C_{free})$:

$$P = \{(v_i, v_k)(v_j, v_{j+1}) / (v_i, v_k) \in S'_F \wedge (v_j, v_{j+1}) \in S_O \wedge v_k = v_j\}. \tag{19}$$

Krok 3. Dla każdej pary $(v_i, v_j)(v_j, v_{j+1}) \in P$ należy:

WSTAWIANIE PROJEKTOWANIE UKŁAD STRONY ODWOŁANIA KORESPOND

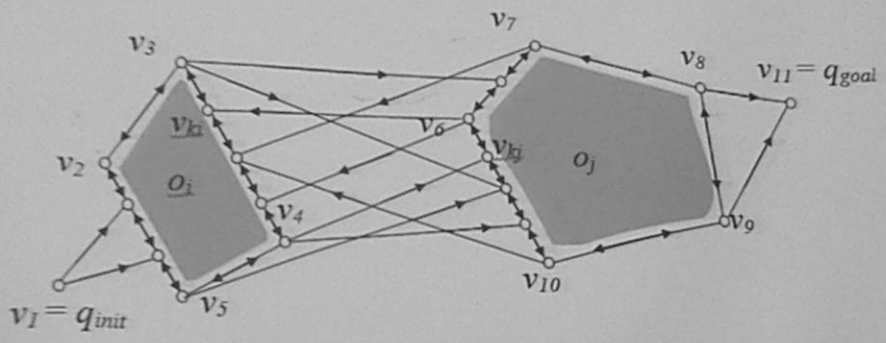
a) obliczyć kąt $\beta = \measuredangle(v_i, v_j)(v_j, v_{j-1})$, długość $l = \| v_i, v_j \|$ oraz odległości d i D;
b) dodać do zbioru RS (removed segments) pierwszą krawędź rozważanej pary;
c) sprawdzić, czy druga krawędź pary spełnia warunek: $\| v_j, v_{j-1} \| > D$, jeżeli ***nie*** to przejść do badania następnej pary, jeżeli ***tak*** to przejść do punktu d);
d) wyliczyć współrzędne (x, y) nowego wierzchołka v_k położonego na krawędzi (v_j, v_{j-1}) w od-ległości d od końca v_j, sprawdzić, czy nowa krawędź (v_i, v_k) przecina się z jakąkolwiek krawędzią $s \in S_O$. Jeżeli ***tak*** to przejść do badania następnej pary, jeżeli ***nie*** to przejść do punktu e);
e) dodać v_k do zbioru V' (new wertex) oraz do zbioru $Q_{(j,j-1)}$ (new wertex of (v_j, v_{j-1}) edge), utworzyć odcinek skierowany $\langle v_j, v_k \rangle$ i dodać go do zbioru A' (new arcs), krawędź (v_j, v_{j-1}) dodać do zbioru PS (segments for partition) oraz do zbioru RS;
f) po zbadaniu wszystkich par ze zbioru P przejść do następnego kroku.

Krok 4. Dla każdej krawędzi (v_j, v_{j-1}) ze zbioru PS należy określić zbiór nowych krawędzi, na jakie dzielą tą krawędź wierzchołki ze zbioru $Q_{(j,j-1)}$:

$$(v_j, v_{j-1}) \rightarrow S_{(j,j-1)} = \{(v_j, v_{k1}), (v_{k1}, v_{k2}), \ldots (v_{kv}, v_{j-1}) / v_{ki} \in Q_{(j,j-1)}\}. \tag{20}$$

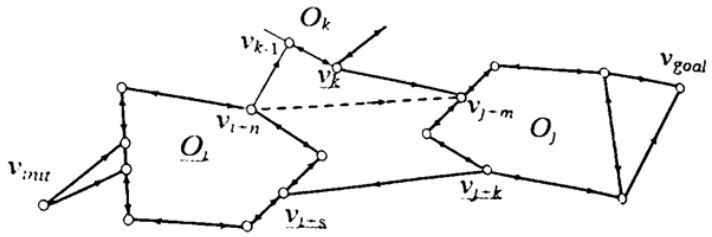
która jak wiadomo dla pierwszej jego części dotyczącej syntezy grafu *VG* wynosi $O(n^2)$.

Przykładowy graf *MVG* otrzymany w wyniku działania opisanego algorytmu ilustruje rys. 6.



Rys. 6. Zmodyfikowany graf widzialności z rys. 3 (dla przejrzystości, pary przeciwnie skierowanych łuków są pokazane jako „podwójnie" skierowane krawędzie).

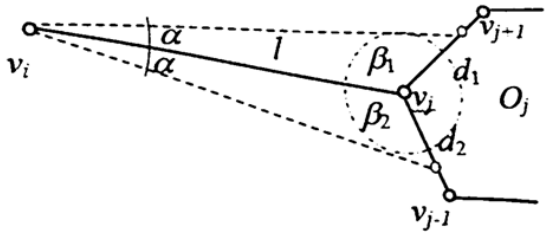
może powodować, że planowane trasy będą nie efektywne (zbyt długie). Sytuację taką ilustruje rys. 7.



Rys. 7. Ilustracja problemu niespójności grafu *MVG*.

Gdy krawędź (v_{i-n} , v_{j-m}) nie spełnia warunku (15), to trasa musi przebiegać dłuższą drogą, przez łuk $<v_{k-1}$, $\underline{v_k}>$ obiektu O_k. Gdy ten łuk również nie istnieje, to w grafie *MVG* nie będzie ścieżki prowadzącej od v_{init} do v_{goal} (graf zachowuje się jak niespójny dla kierunku od v_{init} do v_{goal}). To oznacza, że dla takiej konfiguracji otoczenia trajektoria auto-korekcyjna nie istnieje i trzeba zastosować dodatkowe środki (np. ustawić dodatkową ścianę prowadzącą) lub sięgnąć do innych metod

środki (np. ustawić dodatkową ścianę prowadzącą) lub sięgnąć do innych metod sterowania ruchem, np. metody opisanej w pracy [4].
Można zauważyć, że podczas gdy pojedyncza krawędź jest „za krótka" aby spełnić warunek (15), to dwie sąsiednie krawędzie obrysu przeszkody traktowane łącznie taki warunek mogą spełniać - a dokładniej, każda z nich może spełnić warunek (16) - co pokazuje rys. 8.



Rys. 8. Idea rozszerzenia algorytmu o ruchy swobodne „na narożnik".

...rozszerzenia algorytmu o ruchy swobodne „na narożnik".

Opisany algorytm można rozszerzyć, badając dodatkowo w **Kroku 3** czy obie pary typu $(v_i, \underline{v_j})(\underline{v_j}, v_{j-1})$ i $(v_i, \underline{v_j})(\underline{v_j}, v_{j-1})$ spełniają jednocześnie warunek (16). Gdy ***tak*** to ze zbioru *RS* należy usunąć krawędź $(v_i, \underline{v_j})$ i w **Kroku 5** zastąpić ją dwoma przeciwnie skierowanymi łukami: $<v_i, \underline{v_j}>$, $<\underline{v_j}, v_i>$. Takie rozszerzenie algorytmu zachowuje część oryginalnych krawędzi grafu *VG*, zmniejszając możliwość wystąpienia niespójności. Następuje to jednak kosztem skomplikowania realizacji zaplanowanej trasy auto-korekcyjnej dla ruchów swobodnych, gdyż przy ich zakończeniu trzeba teraz dodatkowo określić, w którą z dwu możliwych krawędzi trafił robot.

enkodery ruchu kół, optyczne sensory zbliżeniowe, oraz sonary. Jego dwu poziomowy system sterowania wykorzystuje 8-bitowy komputer pokładowy oraz komputer zewnętrzny (typu PC), połączone łączem radiowym.
Poziom wykonawczy przyjmuje, kolejkuje i realizuje komendy ruchowe oraz przesyła do poziomu wyższego informacje o sytuacji robota (zakończenie realizacji komendy, obecność obiektów w zasięgu sensorów oraz położenie globalne). Informacja o położeniu jest dostarczana przez system nawigacji przyrostowej.
Robot realizuje dwa rodzaje ruchów:

- ruchy swobodne (ślepe), sterowane w układzie otwartym - na drodze precyzyjnego kształtowania profili prędkości ruchu kół: oraz
- ruchy oparte na sensorach zewnętrznych, sterowane w układzie lokalnego sprzężenia zwrotnego.

Przesyłanie komend z wyprzedzeniem i ich kolejkowanie umożliwia płynny ruch (zakończenie jednego ruchu powoduje automatyczne rozpoczęcie następnego).
Realizacja trajektorii złożonej z ruchów swobodnych jest okresowo korygowana przez poziom nadrzędny, na podstawie informacji dostarczanej z systemu nawigacji przyrostowej.

Poziom planowania (komputer PC) umożliwia automatyczne planowanie off line