Zezwala się na korzystanie z artykułu na warunkach licencji Creative Commons Uznanie autorstwa 3.0

Optymalna strategia przejazdu monocykla przez zbiór punktów referencyjnych w środowisku kolizyjnym

Tomasz Gawron, Maciej Marcin Michałek

Katedra Sterowania i Inżynierii Systemów, Politechnika Poznańska

Streszczenie: W artykule przedstawiono strategię optymalnej realizacji ruchu robota mobilnego o kinematyce monocykla przez zbiór punktów referencyjnych w środowisku kolizyjnym, które ograniczone jest przez znaną *a priori* mapę jego granic. Proponowane rozwiązanie wynika z połączenia algorytmu sterowania VFO zmodyfikowanego dla zadania unikania kolizji oraz algorytmu optymalizacji, polegającego na symulacji robotów wirtualnych. W artykule przedstawiono analizę przestrzeni decyzyjnej tworzonej przez parametry strategii unikania kolizji oraz zaproponowano jej reprezentację w postaci grafu skierowanego. Zaproponowano dwie metody eliminacji potencjalnych cyklicznych ścieżek robota w przypadku skomplikowanych układów przeszkód. Przedstawiono również oszacowanie złożoności pamięciowej algorytmu optymalizacji ruchu. Efektywność proponowanej strategii zilustrowano wybranymi wynikami badań symulacyjnych.

Słowa kluczowe: kinematyka monocykla, VFO, unikanie kolizji, predykcja, optymalizacja

DOI: 10.14313/PAR_212/88

1. Wprowadzenie

Sterowanie robotem mobilnym w środowisku kolizyjnym jest trudnym, lecz bardzo praktycznym zagadnieniem robotyki mobilnej. Ze względu na stopień złożoności problemu i zmienność środowiska ruchu, dokładne odtwarzanie zaplanowanych wcześniej trajektorii lub ścieżek jest w większości przypadków znacznie utrudnione. Trudność ta wynika z konieczności omijania przeszkód, których lokalizacje i kształty nie były dokładnie znane na etapie planowania ruchu. Wraz ze wzrostem liczby przeszkód na drodze robota, jego wynikowa ścieżka coraz bardziej odbiega od tej zaplanowanej, co w wielu przypadkach skutkuje gorszą niż planowana jakością ruchu. Istnienie dużej liczby niewykrytych *a priori* przeszkód, znajdujących się na drodze robota, jest cechą charakterystyczną środowisk, w których pracują roboty ratowniczo-eksploracyjne. Prezentowany algorytm pozwala na uniknięcie opisanego problemu dezaktualizacji zaplanowanych ścieżek robota poprzez inną definicję zadania sterowania. Realizuje on zadanie przejazdu przez zbiór punktów referencyjnych, którego planowanie jest prostsze niż planowanie zadania odtwarzania ścieżki lub trajektorii. Podczas realizacji ruchu prowadzona jest optymalizacja lokalna. Pozwala to na wykorzystanie najbardziej aktualnych danych o środowisku ruchu. Proponowany algorytm realizacji ruchu jest połączeniem algorytmu VFO dla zadania przejazdu przez zbiór punktów referencyjnych [1] rozszerzonego o strategię unikania kolizji [2] z algorytmem optymalizacji polegającym na symulacji robotów wirtualnych. Zagadnienie planowania wspomnianego zbioru punktów referencyjnych nie jest poruszane w tej pracy. Przykładowy algorytm planujący ten zbiór zaprezentowano w [3].

Niektóre z dotychczas podjętych prób rozwiązania problemu optymalizacji ruchu podczas unikania kolizji polegały na przeszukiwaniu podzbioru trajektorii dopuszczalnych (np. [4]), lokalnej modyfikacji pól potencjałowych (np. [5]) lub zastosowaniu mieszanego programowania liniowego (m.in. [6]). Zadanie przejazdu/przelotu przez zbiór punktów (ang. *waypoint following*) rozpatrywano najczęściej w obszarze robotów latających (np. w [7, 8]). Opracowano również metody, w których rezygnuje się z optymalizacji ruchu na rzecz innych aspektów problemu, takich jak niepewności pomiarowe sensorów robota [9].

2. Postawienie problemu

2.1. Kinematyka monocykla

Rozważania zawarte w pracy ograniczone są do robotów o kinematyce monocykla, która opisana jest następująco:

$$\dot{\boldsymbol{q}} = \begin{bmatrix} 1 & 0 \\ 0 & \cos\theta \\ 0 & \sin\theta \end{bmatrix} \boldsymbol{u}, \qquad \boldsymbol{u} \triangleq \begin{bmatrix} \omega \\ v \end{bmatrix}, \tag{1}$$

gdzie $\dot{\boldsymbol{q}}$ jest wektorem prędkości konfiguracyjnych robota, $\boldsymbol{q} \triangleq \begin{bmatrix} \theta & x & y \end{bmatrix}^T = \begin{bmatrix} \theta & \boldsymbol{q}^{*\top} \end{bmatrix}^\top$ stanowi współrzędne konfiguracyjne robota, θ jest orientacją robota, x i y są współrzędnymi punktu prowadzenia platformy robota, $\boldsymbol{u}$ jest wektorem sygnałów sterujących, przy czym odpowiednio, ω i v są prędkościami kątową i postępową platformy robota. Graficzna interpretacja współrzędnych konfiguracyjnych robota widoczna jest na rys. 2.

2.2. Definicja zadania

Przyjmuje się następujące założenia:

Z1. Dany jest opis środowiska ruchu w postaci zbinaryzowanej siatki zajętości.

Z2. Dany jest uporządkowany zbiór W składający się z N punktów referencyjnych dla robota znajdującego się w punkcie $\boldsymbol{q}_0 = \begin{bmatrix} \theta_0 & x_0 & y_0 \end{bmatrix}^\top$. Zbiór ten zdefiniowano następująco:

$$W \triangleq \{w_1; w_2; \dots w_N\}, \qquad w_i \triangleq \{\boldsymbol{q}_{ri}; \mu_i\}, \tag{2}$$

gdzie w_i jest i-tym punktem referencyjnym, $\boldsymbol{q}_{ri}$ jest wektorem zadanych współrzędnych konfiguracyjnych robota dla i-tego punktu referencyjnego, a μ_i względnym współczynnikiem naprowadzania dla sterownika VFO opisanego w sekcji 4.

Z3. Zbiór W zaplanowany jest tak, że możliwa jest kolejna realizacja każdego z punktów referencyjnych gwarantująca sprowadzenie robota do konfiguracji końcowej $\boldsymbol{q}_N$ z założoną dokładnością ϵ w skończonym czasie t_N. Realizacja konfiguracji $\boldsymbol{q}_N$ z dokładnością ϵ oznacza, że musi zachodzić:

$$\forall t \geq t_N \quad ||\boldsymbol{q}_N - \boldsymbol{q}(t)|| < \epsilon. \tag{3}$$

Z4. Każdy z N segmentów ruchu ma skończoną długość. Przez i-ty segment ruchu rozumiana jest ścieżka robota podczas realizacji punktu referencyjnego $\boldsymbol{w}_i$.

Z5. Tylko ostatni punkt referencyjny (punkt końcowy) musi być zrealizowany z dokładnością ϵ. Dla pozostałych punktów referencyjnych stosowany jest warunek przełączenia przedstawiony w formie Algorytmu 1, w którym $\boldsymbol{e}_i^* \triangleq \boldsymbol{q}_{ri}^* - \boldsymbol{q}^*$ jest uchybem pozycji robota w i-tym segmencie ruchu, a ϵ_i jest dokładnością realizacji i-tego punktu referencyjnego. Wartość ϵ_i jest obliczana na bieżąco przez algorytm realizacji ruchu (sekcja 5.5).

Algorytm 1 Warunek przełączenia realizowanego punktu referencyjnego

if $||\boldsymbol{e}_i^*|| < \epsilon_i$ **and** $i < N$ **then**
$i \leftarrow i + 1$

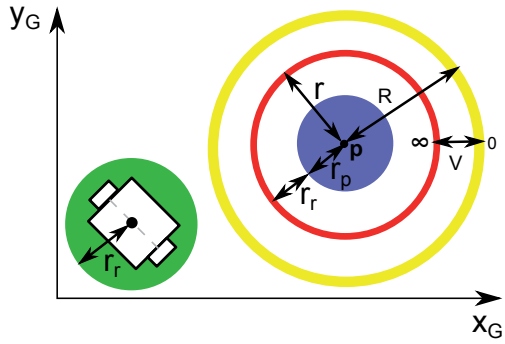
Z6. W środowisku ruchu znajdują się przeszkody statyczne, których lokalizacja nie jest znana *a priori*. Przeszkody te są wykrywane na bieżąco podczas realizacji ruchu robota. Wiedza na temat przeszkód reprezentowana jest przez aktualizowany na bieżąco zbiór P_w zdefiniowany następująco:

$$P_w \triangleq \{\lambda_1; \lambda_2; \ldots \lambda_{N_p}\}, \qquad \lambda \triangleq \{\boldsymbol{p}, r, R, \sigma\}, \tag{4}$$

gdzie λ jest modelem przeszkody, $\boldsymbol{p} \triangleq \begin{bmatrix} x_p & y_p \end{bmatrix}^\top$ pozycją przeszkody w przestrzeni konfiguracyjnej robota, r promieniem koła strefy niedozwolonej dla robota, R promieniem koła strefy oddziaływania przeszkody na robota, a σ zmienną decyzyjną oznaczającą kierunek omijania przeszkody przez robota (lewo, prawo). Przestrzeń zajmowana przez robota modelowana jest jako koło o środku w punkcie $\boldsymbol{q}^*$ i promieniu r_r. Przyjmuje się, że $r = r_p + r_r$, gdzie r_p jest wykrytym promieniem przeszkody. Sposób modelowania przeszkód oraz przestrzeni zajmowanej przez robota widać na rys. 1.

Z7. Wszystkie obszary z przeszkodami mogą nakładać się na siebie w sposób dowolny. Punkty referencyjne mogą leżeć w obszarach oddziaływania lub obszarach niedozwolonych przeszkód.

Zadanie rozpatrywane w pracy polega na wyznaczeniu ograniczonych sterowań $\boldsymbol{u}$, sprowadzających robota do konfiguracji $\boldsymbol{q}_N$, dla której spełniona będzie nierówność 3, po ścieżce bezkolizyjnej i optymalnej w sensie maksymalizacji wskaźnika Ψ_z podanego w sekcji 5.2.



Rys. 1. Model przeszkody i przestrzeni zajmowanej przez robota
Fig. 1. Model of an obstacle and space occupied by the robot

3. Koncepcja robotów wirtualnych

Rozpatrzmy scenariusz przedstawiony na rys. 2 pod kątem realizacji zadania z rozdziału 2.2. Robot realizując punkt $\boldsymbol{q}_{ri+1}$ wjedzie w obszar oddziaływania przeszkody $\boldsymbol{p}_1$. Przeszkoda ta może zostać ominięta lewą lub prawą stroną. Jeśli przeszkoda $\boldsymbol{p}_1$ zostanie ominięta lewą stroną, robot wjedzie także w obszar oddziaływania przeszkody $\boldsymbol{p}_2$. Przeszkodę $\boldsymbol{p}_2$ można ominąć bezpiecznie jedynie prawą stroną ze względu na jej lokalizację. W przypadku ominięcia przeszkody $\boldsymbol{p}_1$ prawą stroną, robot zrealizuje punkt $\boldsymbol{q}_{ri+1}$ bez wjazdu w strefy oddziaływania kolejnych przeszkód. Wynika z tego, że wybór kierunków omijania przeszkód ma wpływ na długość wynikowej ścieżki robota, jej bezkolizyjność oraz szereg jej innych parametrów (patrz rozdział 4.2). Można więc wybierać kierunki omijania przeszkód w każdym segmencie ruchu tak, aby uzyskać optymalną ze względu na przyjęte wskaźniki ścieżkę robota.

Do wyznaczenia ścieżek robota, które mogłyby zostać uzyskane w danym segmencie ruchu proponuje się wykorzystanie koncepcji wirtualnych robotów. Polega ona na wielokrotnym symulowaniu robota wirtualnego, czyli układu zamkniętego składającego się ze sterownika oraz modelu kinematyki robota, z przyjętymi różnymi kierunkami omijania przeszkód (lewo, prawo). W momencie osiągnięcia przez robota punktu $\boldsymbol{q}_{ri}$, wysyłany jest z tego punktu robot wirtualny W1. Symulacja tego robota kończy się w punkcie wjazdu w obszar oddziaływania przeszkody $\boldsymbol{p}_1$. Z tego punktu wysyłane są dwa roboty wirtualne W2 i W3 z początkowymi wartościami przyjętych wskaźników jakości ścieżki równymi wartościom końcowym robota W1. Dla robota W2 przyjmuje się omijanie przeszkody $\boldsymbol{p}_1$ prawą stroną, natomiast dla robota W3 przyjmuje się omijanie przeszkody $\boldsymbol{p}_1$ lewą stroną. Proces ten powtarza się w sposób rekursywny aż do osiągnięcia punktu $\boldsymbol{q}_{ri+1}$ przez roboty wirtualne. Robot W3 jest zatrzymywany po wjeździe w obszar oddziaływania przeszkody $\boldsymbol{p}_2$. Od tego miejsca symulowane są roboty W4 i W5. Ze względu na lokalizację przeszkody $\boldsymbol{p}_2$, symulacja robota W5 kończy się jego kolizją z przeszkodą. Roboty W2 i W4 osiągają bezpiecznie punkt $\boldsymbol{q}_{ri+1}$. Spośród tych dwóch robotów wybierany jest ten z najlepszymi wskaźnikami jakości ścieżki. Wskaźniki te zależą od wyników symulacji robotów, które osiągnęły punkt referencyjny oraz ich poprzedników. Przykładowo, dla robota W4, wskaźniki te zależą także od wyników symulacji robotów W1 i W3. Opi-

sana koncepcja jest kluczowa dla proponowanego algorytmu optymalizacji opisanego szczegółowo w rozdziale 5.

4. Sterownik VFO

Proponowany w pracy algorytm jest ściśle związany ze sterownikiem VFO (ang. *vector field orientation*) dla zadania przejazdu przez zbiór punktów zaproponowanym w [2]. W tej pracy prezentowany jest on w wersji rozszerzonej o strategię unikania kolizji [1]. Zadanie ruchu zdefiniowane w rozdziale 2.2 jest traktowane przez sterownik VFO jako N podzadań sterowania do punktu. Sterownik VFO jest stabilizatorem nieciągłym. Istotą jego działania jest wykorzystanie wektorowego pola zbieżności $\boldsymbol{H}$. Pole to ma charakter prędkości i jest projektowane w taki sposób, że śledzenie tego pola przez prędkość konfiguracyjną robota gwarantuje zbieżność do i-tej konfiguracji referencyjnej $\boldsymbol{q}_i$. Wektorowe pole zbieżności $\boldsymbol{H}$ dla i-tego punktu referencyjnego ma postać (patrz [1, 2]):

$$\boldsymbol{H} = \begin{bmatrix} H_a \\ \boldsymbol{H}^* \end{bmatrix} \triangleq \begin{bmatrix} k_a e_a + \dot{\theta}_a \\ \boldsymbol{h}^* + \boldsymbol{h}_o^* \end{bmatrix}, \tag{5}$$

$$\boldsymbol{H}^* = \begin{bmatrix} H_x & H_y \end{bmatrix}^T, \tag{6}$$

$$\boldsymbol{h}_o^* \triangleq \left(\sum_{k=1}^{N_p} \sigma_k V(d_k) \right) \boldsymbol{R}_o \boldsymbol{h}^*, \tag{7}$$

$$\dot{\theta}_a = \frac{\dot{H}_y H_x - \dot{H}_x H_y}{H_x^2 + H_y^2}, \tag{8}$$

$$e_a \triangleq \mathrm{Atan2c}(\zeta_i H_y, \zeta_i H_x) - \theta, \tag{9}$$

$$\boldsymbol{h}^* \triangleq k_p \boldsymbol{e}_i^* + \boldsymbol{v}_i^*, \tag{10}$$

$$\boldsymbol{v}_i^* \triangleq -k_p \mu_i \zeta_i ||\boldsymbol{e}_i^*||, \tag{11}$$

$$\boldsymbol{R}_o = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}, \tag{12}$$

gdzie k_a jest strojonym przez użytkownika wzmocnieniem sterowania orientującego, e_a jest pomocniczym uchybem orientacji, θ_a jest orientacją pomocniczą, $\boldsymbol{H}^*$ jest polem zbieżności pozycji, $\boldsymbol{h}^*$ jest podstawowym składnikiem pola zbieżności pozycji, $\boldsymbol{h}_o^*$ jest składnikiem pola zbieżności pozycji odpowiadającym za unikanie kolizji z przeszkodami, $\boldsymbol{v}_i^*$ jest prędkością wirtualną, μ_i jest względnym współczynnikiem naprowadzania, $V(d_k)$ jest wartością funkcji potencjałowej z równania (16) dla k-tej przeszkody ze zbioru P_w, $\sigma_k \in \{-1; 1\}$ jest zmienną decyzyjną dla k-tej przeszkody ze zbioru P_w, a $\boldsymbol{R}_o$ jest macierzą rotacji. Na podstawie powyższej postaci wektorowego pola zbieżności sformułowane jest sterowanie:

$$\boldsymbol{u} = \begin{bmatrix} \omega \\ v \end{bmatrix} \triangleq \begin{bmatrix} k_a e_a + \dot{\theta}_a \\ U_2 \cos e_a \end{bmatrix}, \tag{13}$$

gdzie ω jest sterowaniem orientującym, v sterowaniem popychającym, k_a jest wzmocnieniem sterowania orientującego, a U_2 parametrem profilu prędkości. Przyjęto stały profil prędkości, tj. przy zerowym uchybie orientacji (wtedy $\cos e_a = 1$) robot porusza się wzdłuż ścieżki z prędkością postępową $v = U_2$. Parametr U_2 ma wymiar prędkości i jest dobierany przez użytkownika sterownika.

Prędkość wirtualna $\boldsymbol{v}_i^*$ nadaje opisywanemu sterownikowi właściwość nazywaną efektem naprowadzania. Efekt ten powoduje, że ścieżka sterowanego robota w pewnym otoczeniu punktu $\boldsymbol{q}_{ri}$ przebiega w otoczeniu prostej o kierunku zgodnym z orientacją zadaną θ_i. Wielkość tego otoczenia punktu $\boldsymbol{q}_{ri}$ zależy od modułu prędkości wirtualnej $\boldsymbol{v}^*$, który skalowany jest przez względny współczynnik naprowadzania $0 < \mu < 1$. Intensywność efektu naprowadzania rośnie nieliniowo wraz ze wzrostem μ. Wpływ efektu naprowadzania na ścieżkę robota można zaobserwować na rys. 2 (szerzej na ten temat w [10]). Szara, przerywana linia (robot W4') odpowiada ścieżce, która mogłaby zostać uzyskana podczas realizacji punktu $\boldsymbol{q}_{ri+1}$ z względnym współczynnikiem naprowadzania mniejszym niż w przypadku ścieżki oznaczonej na czarno (robot W4).

Składnik $\boldsymbol{h}_o^*$ występujący w równaniu (6) wynika z przeniesienia strategii unikania kolizji zaprezentowanej w [1] do zadania przejazdu przez zbiór punktów referencyjnych. Strategia ta polega na zmianie kierunku wektora pola zbieżności pozycji poprzez dodanie do niego wektora odpychającego o kierunku prostopadłym do kierunku tego pola oraz długości i zwrocie odpowiadającym sumie wartości funkcji potencjałowej V dla wszystkich N_p przeszkód wykrytych z uwzględnieniem znaków określonych przez poszczególne zmienne decyzyjne σ_k. Na rys. 2 pokazano zależność ścieżki robota od zmiennej decyzyjnej σ_1. Ścieżka niebieska jest wynikiem wyboru $\sigma_1 = 1$, który skutkuje omijaniem przeszkody $\boldsymbol{p}_1$ prawą stroną, patrząc z lokalnego układu współrzędnych robota. Ścieżka czarna odpowiada wyborowi $\sigma_1 = -1$, który powoduje omijanie przeszkody $\boldsymbol{p}_1$ lewą stroną.

5. Optymalizacja ruchu

5.1. Problem optymalizacyjny

Globalnie optymalna, w sensie wskaźnika (22), ścieżka robota gwarantująca realizację zadania postawionego w rozdziale 2.2 jest elementem zbioru wszystkich ścieżek dopuszczalnych. Zbiór ten jest gęsty, więc rozpatrywany problem optymalizacyjny nie może być rozwiązany wprost technikami optymalizacji kombinatorycznej, które zakładają skończoność zbioru reprezentującego przestrzeń decyzyjną problemu. Problem ten można zapisać w postaci problemu mieszanego, całkowitoliczbowego programowania liniowego, o którym wiemy, że jest NP-trudny. Złożoność obliczeniowa tak postawionego problemu sprawia, że rozwiązywanie go na bieżąco podczas detekcji przeszkód w środowisku ruchu wydaje się wyjątkowo trudne w praktyce.

Z wyżej opisanych powodów, proponowana jest strategia lokalnej optymalizacji ruchu prowadząca do przybliżonego rozwiązania omawianego problemu optymalizacyjnego. Strategia ta prowadzi do rozwiązania, które można zrealizować sterownikiem zaprezentowanym w rozdziale 4. W strategii tej zastępuje się geometryczny sposób doboru zmiennych

decyzyjnych [1] algorytmem rozwiązującym problem optymalizacyjny sformułowany następująco:

Problem 1 *Dla każdego z N segmentów ruchu wybrać zbiór zmiennych decyzyjnych* $\Sigma_o = \{\sigma_1, \sigma_2, \ldots, \sigma_{N_p}\}$*, który maksymalizuje wskaźnik (22) dla ścieżki robota uzyskanej w danym segmencie poprzez sterowanie VFO z rozdziału 4 z zastosowaniem* Σ_o*.*

Sformułowanie problemu optymalizacyjnego z wykorzystaniem dwuwartościowych zmiennych σ_k decydujących o kierunku omijania przeszkody jest korzystne z wielu względów. Wykorzystanie tych zmiennych nie wymaga ingerencji w sterownik VFO, co daje gwarancję zbieżności konfiguracji robota do konfiguracji zadanej w przypadku braku przeszkód na ścieżce robota. Wykorzystanie algorytmu optymalizacji z rozdziału 5 pozwala na odnalezienie zbioru Σ_o minimalizującego prawdopodobieństwo kolizji robota z przeszkodami lub określenie braku takiego zbioru.

5.2. Wskaźniki jakości ścieżki

Zadanie optymalizacji ruchu rozpatrywane w pracy polega na maksymalizacji zbiorczego wskaźnika jakości ścieżki, który jest kombinacją wskaźników cząstkowych. Na podstawie badań symulacyjnych przyjęto 4 wskaźniki cząstkowe jakości ścieżki robota. Każdy z nich modeluje jeden z ważnych w praktyce aspektów ruchu robota. Wskaźniki jakości ścieżki rozpatrywane są w dziedzinie czasu dyskretnego τ. Przyjmujemy, że okres próbkowania wskaźników jakości wynosi T_p. Wspomniane wskaźniki cząstkowe definiowane są następująco:

Długość ścieżki:

$$\chi_D(\tau) \triangleq \begin{cases} 0 & \text{dla} \quad \tau = 0, \\ \chi_D(\tau-1) + ||\dot{\boldsymbol{q}}^*(\tau)||T_p & \text{dla} \quad \tau > 0. \end{cases} \tag{14}$$

Bezpieczeństwo ścieżki jest nieliniową funkcją odległości robota od wszystkich aktualnie znanych przeszkód:

$$\chi_B(\tau) \triangleq \begin{cases} 0 & \text{dla} \quad \tau = 0, \\ \max\left\{\chi_B(\tau-1); \sum_k^{N_p} V(d_k)\right\} & \text{dla} \quad \tau > 0, \end{cases} \tag{15}$$

gdzie

$$V(d_k) \triangleq \begin{cases} 0 & \text{dla} \quad d_k > R_k, \\ \left(\dfrac{d_k^2 - R_k^2}{d_k^2 - r_k^2}\right)^2 & \text{dla} \quad r_k < d_k \le R_k, \\ \infty & \text{dla} \quad d \le r_k, \end{cases} \tag{16}$$

$$d_k \triangleq ||\boldsymbol{p_k} - \boldsymbol{q}^*||, \tag{17}$$

przy czym $V(d_k)$ jest wartością funkcji potencjałowej dla k-tej przeszkody, a d_k jest odległością robota od środka k-tej przeszkody. Wskaźnik ten rośnie bardzo szybko, jeśli robot znajduje się w bliskiej odległości od chociaż jednej przeszkody.

Koszt sterowania:

$$\chi_E(\tau) \triangleq \begin{cases} 0 & \text{dla} \quad \tau = 0, \\ \chi_E(\tau-1) + ||\boldsymbol{u}(\tau)||T_p & \text{dla} \quad \tau > 0. \end{cases} \tag{18}$$

Gładkość ścieżki $\chi_S(\tau)$ rozumiana jest jako maksimum modułu prędkości kątowej $|\dot{\theta}|$ robota w danym segmencie ruchu. Przy założeniu stałego profilu prędkości postępowej robota, wartość wskaźnika χ_S odpowiada maksymalnej krzywiźnie ruchu robota:

$$\chi_S(\tau) \triangleq \sup_{\tau \ge 0} |\dot{\theta}(\tau)|. \tag{19}$$

Wprowadzono procedurę normalizacji wskaźników cząstkowych, w celu sprowadzenia ich do zakresu $\langle 0; 1\rangle$. Normalizacja wskaźników zdefiniowana jest następującą zależnością:

$$\Psi \triangleq \begin{cases} 0 & \text{dla} \quad \chi_{max} = 0, \\ 1 - \dfrac{\chi}{\chi_{max}} & \text{dla} \quad \chi_{max} > 0, \end{cases} \tag{20}$$

$$\chi_{max} \triangleq \max\{\chi_1; \chi_2; \ldots \chi_j\}, \tag{21}$$

gdzie Ψ jest normalizacją dowolnego wskaźnika cząstkowego $\chi \in X \triangleq \{\chi_D; \chi_B; \chi_E; \chi_S\}$, a χ_{max} jest maksymalną wartością ze zbioru, względem którego przeprowadzana jest normalizacja (np. zbioru wskaźników dla wszystkich robotów wirtualnych z danego segmentu ruchu). Ze względu na normalizację, zbiorczy wskaźnik jakości ścieżki Ψ_z jest zdefiniowany tylko dla ostatniej chwili czasowej horyzontu optymalizacji ruchu τ_n. Zbiorczy wskaźnik jakości ścieżki jest kombinacją liniową poszczególnych wskaźników znormalizowanych:

$$\Psi_z \triangleq K_D\Psi_D + K_B\Psi_B + K_E\Psi_E + K_S\Psi_S, \tag{22}$$

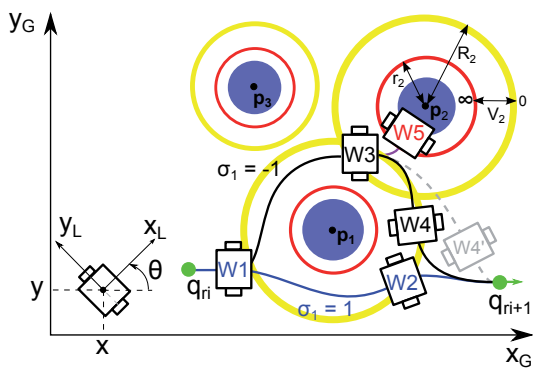
w której wagi K_D, K_B, K_E, K_S są parametrami wybieranymi przez użytkownika z przedziału $\langle 0; \infty)$. Wagi te pozwalają na dostrojenie ścieżek uzyskiwanych poprzez działanie prezentowanego algorytmu do poszczególnych zastosowań. Użytkownik może zwiększyć wpływ istotnych dla niego wskaźników cząstkowych na ocenę ruchu przez algorytm. Zbiorczy wskaźnik jakości ścieżki robota stanowi kryterium, według którego prowadzona jest optymalizacja ruchu (maksymalizacja wskaźnika).

5.3. Przestrzeń decyzyjna

Oznaczmy zbiór przeszkód, których strefy oddziaływania są przecinane przez ścieżkę robota podczas realizacji i-tego segmentu ruchu przez $P_i \subseteq P_w$. Liczba zmiennych decyzyjnych ze zbioru Σ_o wpływających na ścieżkę robota mieści się w przedziale $\langle 0; |P_i|\rangle$, gdzie $|P_i|$ oznacza liczebność zbioru P_i. Liczba ta nie jest znana przed rozpoczęciem procesu optymalizacji ruchu, gdyż zależy ona od wartości poszczególnych elementów ze zbioru Σ_o (zmiennych decyzyjnych), które wpływają na ścieżkę robota podczas realizacji segmentu ruchu. Wynika z tego, że liczebność przestrzeni decyzyjnej (zbioru możliwych decyzji skutkujących różnymi ścieżkami robota) zależy od układu przeszkód i wyboru wartości poszczególnych zmiennych decyzyjnych. Przestrzeń decyzyjna D jest dyskretnym zbiorem o liczebności z przedziału

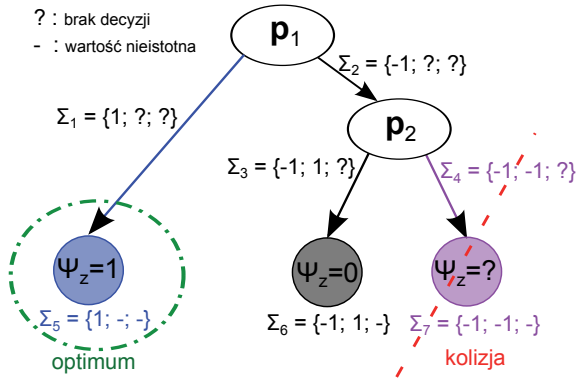
$< 1; 2^{N_p} >$. Ta zależność widoczna jest na rys. 2 i rys. 3. W tym przypadku zachodzi $|D| = 3$ pomimo $N_p = 3$. Przeszkoda $\boldsymbol{p}_3$ jest położona tak, że robot nigdy nie znajdzie się w obszarze jej oddziaływania. Z tego powodu wartość zmiennej σ_3 odpowiadającej kierunkowi omijania przeszkody $\boldsymbol{p}_3$ nie wpływa na ścieżkę robota. Podobnie wybór $\sigma_1 = 1$ (trajektoria niebieska) powoduje, że wartości σ_2 i σ_3 nie wpływają na ścieżkę robota. Można wyobrazić sobie sytuację, w której punkt $\boldsymbol{q}_i$ znajduje się w obszarach oddziaływania przeszkód $\boldsymbol{p}_1$, $\boldsymbol{p}_2$ i $\boldsymbol{p}_3$. Wtedy przestrzeń decyzyjna miałaby maksymalną możliwą liczebność $|D| = 2^{N_p} = 8$. Jeśli żaden z obszarów oddziaływania przeszkód nie leży na ścieżce robota, to $|D| = 1$, $\Sigma_o = \emptyset$ i tym samym $\boldsymbol{h}_o^* = \boldsymbol{0} \Rightarrow \boldsymbol{H}^* = \boldsymbol{h}^*$. W takim przypadku prawo sterowania zostaje zdegenerowane do postaci podanej w [2], a ścieżka robota otrzymana w wyniku tego prawa sterowania przyjmowana jest jako optymalna w sensie przyjętego zbiorczego wskaźnika jakości.

W przypadku rozpatrywanym na rys. 2, spośród trzech możliwych ścieżek robota (czarna, niebieska i fioletowa) tylko dwie są bezkolizyjne (czarna i niebieska). Elementy zbioru D odpowiadające kolizyjnym ścieżkom robota są konsekwencją rozpatrywania również takich wartości zmiennych σ, które powodują przyciąganie robota do przeszkód. Ścieżka fioletowa jest konsekwencją wyboru zmiennej $\sigma_2 = -1$. Wartość tej zmiennej sprawia, że wektor $\boldsymbol{h}_o^*$ kieruje wektor pola zbieżności $\boldsymbol{H}^*$ w stronę przeszkody $\boldsymbol{p}_2$. W tym przypadku jest to zjawisko niekorzystne, lecz dla niektórych układów przeszkód z nakładającymi się obszarami oddziaływania, taki wybór zmiennych decyzyjnych skutkuje ścieżką robota o lepszym wskaźniku jakości. Ścieżki kolizyjne są eliminowane podczas procesu optymalizacji (podrozdział 5.4).



Rys. 2. Możliwe ścieżki robota oraz odpowiadające im roboty wirtualne dla prostego układu przeszkód
Fig. 2. Possible robot paths and corresponding virtual robots for simple layout of obstacles

Dla każdego segmentu ruchu można zapisać graf opisujący przestrzeń decyzyjną odpowiadającą temu segmentowi. Graf reprezentujący przestrzeń decyzyjną dla segmentu $(i, i+1)$ z rys. 2 jest widoczny na rys. 3. Jest to graf skierowany z dwoma typami wierzchołków. Wierzchołki typu I odpowiadają poszczególnym przeszkodom $\boldsymbol{p}_1$, $\boldsymbol{p}_2$ i $\boldsymbol{p}_3$. Wierzchołki typu II odpowiadają zbiorczym wskaźnikom jakości dla poszczególnych możliwych ścieżek robota i zbiorom Σ_i prowadzącym do uzyskania ścieżek o tych wskaźnikach. Krawędzie podgrafu przestrzeni decyzyjnej składającego się z wierzchołków typu I są etykietowane zbiorami Σ_i powodującymi przejazd robota w obszar oddziaływania przeszkody odpowiadającej wierzchołkowi, do którego skierowana jest krawędź. Przyjęto, że $\forall i |\Sigma_i| = N_p$. Elementy zbioru Σ_i przyjmują wartości ze zbioru $\{1; -1; -; ?\}$, gdzie $-$ oznacza brak wpływu wartości danej zmiennej decyzyjnej na ścieżkę robota, a ? oznacza nieznaną wartość zmiennej decyzyjnej. Przyjęty sposób etykietowania krawędzi pozwala na detekcję zbiorów Σ mogących prowadzić do cyklicznych wynikowych ścieżek robota. Detekcja tych zbiorów odbywa się poprzez specyficzną detekcję cykli w grafie przestrzeni decyzyjnej zawartą w Algorytmie 2.



Rys. 3. Przestrzeń decyzyjna dla problemu z rys. 2
Fig. 3. Decision space for problem shown in Fig. 2

Graf przedstawiony na rys. 3 należy czytać w następujący sposób. Przyjęcie zmiennej $\sigma_1 = 1$ po wjechaniu przez robota w obszar oddziaływania przeszkody $\boldsymbol{p}_1$ skutkuje wykonaniem przez robota segmentu ruchu ze wskaźnikiem $\Psi_z = 1$. Przyjęcie odwrotnej wartości tej zmiennej skutkuje wjechaniem przez robota w obszar oddziaływania przeszkody $\boldsymbol{p}_2$, itd. Na zielono zaznaczono wierzchołek typu II reprezentujący optymalny zbiór zmiennych decyzyjnych Σ_o. Na czerwono oznaczono wierzchołek reprezentujący ścieżkę kolizyjną. Kolizyjność tej ścieżki nie wynika z postaci grafu.

Graf przestrzeni decyzyjnej D ma $|D|$ wierzchołków typu II i co najwyżej N_p wierzchołków typu I. Ponadto jeśli przez E_i oznaczymy liczbę krawędzi wchodzących do danego wierzchołka typu I, a przez F_i liczbę krawędzi wychodzących z tego samego wierzchołka, to dla dowolnej przestrzeni D zachodzi propagacja decyzji: $\forall i |F_i| = 2|E_i|$. Wynika to z tego, że każda przeszkoda odpowiadająca wierzchołkowi typu I może być ominięta lewą lub prawą stroną. W grafie przestrzeni decyzyjnej mogą występować cykle. Graf przestrzeni decyzyjnej jest niekompletny (tj. nie każdy wierzchołek jest bezpośrednio połączony ze wszystkimi pozostałymi) i zazwyczaj rzadki w sensie teorii grafów. Z powyższych rozważań wynika, że rozmiar tego grafu jest ograniczony zgodnie z następującymi zależnościami:

$$|D| \le |E| \le |D| + \frac{N_p^2 - N_p}{2}, \quad |D| \le |V| \le |D| + N_p, \tag{23}$$

gdzie $|E|$ jest liczebnością zbioru krawędzi, a $|V|$ jest liczebnością zbioru wierzchołków. Biorąc pod uwagę maksymalną liczebność zbioru D widać, że dla dużych N_p składnik $|D|$ jest dominujący. Wynika z tego, że reprezentacja grafu

przestrzeni decyzyjnej ma w najgorszym przypadku wykładniczą złożoność pamięciową $O(N_p) = 2^{N_p}$. Wydaje się, że przypadek ten występuje niezwykle rzadko. Średnio skomplikowane segmenty ruchu zwykle mają taką topologię, że $N_p < 30$ i $|D| << 2^{N_p}$.

5.4. Algorytm optymalizacji

Graf przestrzeni decyzyjnej jest użytecznym narzędziem koncepcyjnym, lecz utrzymywanie jego pełnej reprezentacji w pamięci wydaje się niekiedy kosztowne (np. dla jednostek wbudowanych pracujących w czasie rzeczywistym lub skomplikowanych układów przeszkód). Taka reprezentacja nie jest potrzebna do rozwiązania Problemu 1. Aby rozwiązać Problem 1 należy wyznaczyć zbiór V_{II} wierzchołków drugiego rodzaju i wybrać jego element o maksymalnym wskaźniku Ψ_z. Wyznaczenie tego zbioru nie wymaga jawnej konstrukcji grafu przestrzeni decyzyjnej.

Wyznaczenie zbioru V_{II} odbywa się poprzez rekursywny proces symulacji robotów wirtualnych (patrz rozdział 3). Oznaczmy przez $Fs(\boldsymbol{q}_{v0}, \boldsymbol{q}_i, \Sigma_v)$ funkcję symulującą ruch robota z warunku początkowego $\boldsymbol{q}_{v0}$ do punktu $\boldsymbol{q}_i$. Podczas symulacji przyjmujemy, że zmienne decyzyjne sterownika VFO są dane zbiorem Σ_v. W momencie wjazdu symulowanego robota w obszar oddziaływania dowolnej przeszkody powinno nastąpić przerwanie symulacji i zwrócenie informacji nt. stanu robota wirtualnego.

Algorytm 2 wykorzystuje funkcję Fs do wyznaczenia zbioru Σ_o będącego rozwiązaniem Problemu 1. Przyjmujemy, że Q jest kolejką FIFO, E_v zbiorem krawędzi grafu przestrzeni decyzyjnej odwiedzonych podczas budowania pojedynczej ścieżki robota, X zbiorem cząstkowych wskaźników jakości ruchu, X_n tym samym zbiorem dla kolejnego odcinka ścieżki, $merge(X_1, X_2)$ operacją wyznaczania zbioru X z dwóch odcinków ścieżki robota, a j jest indeksem rozpatrywanej przeszkody. Operacja $merge(X_1, X_2)$ polega na wyliczeniu poszczególnych wskaźników traktując elementy X_1 jako wartości z chwili $\tau - 1$, a elementy X_2 jako wartości z chwili τ. Operacja ta daje prawidłowe wyniki dzięki monotoniczności zdefiniowanych wskaźników cząstkowych. Operacja *normalize* oznacza normalizację wszystkich wskaźników cząstkowych wg. zależności (20).

Algorytm 2 wykorzystuje stos, aby wykonać przejście w głąb (ang. *depth first traversal*) przez graf przestrzeni decyzyjnej. Graf nie jest budowany uprzednio. Potrzebne w danej chwili części grafu są pośrednio konstruowane na bieżąco. Zbiór E_v opisuje podgraf grafu przestrzeni decyzyjnej, który odpowiada pojedynczej możliwej ścieżce robota. Wykrycie rodzaju cyklu zbudowanego na krawędziach tego grafu jest równoznaczne z wykryciem zbioru Σ, który może prowadzić do cyklicznej ścieżki robota. W takim wypadku badanie danej ścieżki jest przerywane. Prezentowany algorytm ma w najgorszym wypadku złożoność pamięciową $O_m(N_p) = 2^{N_p}$. Należy wspomnieć, że mimo takiego samego wykładniczego ograniczenia zajmowanej ilości pamięci w przypadku jawnej reprezentacji grafu i działania prezentowanego algorytmu, faktyczne zużycie pamięci będzie w większości przypadków niższe dla prezentowanego algorytmu, gdyż będzie zachodzić $|V_I| > |V_{II}|$. Złożoność pamięciowa prezentowanego algorytmu zależy silnie od procedury normalizacji. Przyjęcie procedury normalizacji niekorzystającej z maksimów w zbiorze D (np. maksima przyjęte doświadczalnie), skutkowałoby znacznie niższą złożonością pamięciową $O_m(N_p) = CN_p$, gdzie C jest najdłuższym cyklem w przestrzeni decyzyjnej.

Algorytm 2 Funkcja $\Sigma_o = findSigmas(\boldsymbol{q}_{v0}, \boldsymbol{q}_i)$

$D \leftarrow \emptyset$
$\Sigma_v \leftarrow \{?, ? \ldots ?\}$
$Q.push(\boldsymbol{q}_{v0}, \Sigma_v, \emptyset, \emptyset)$
while $Q.notEmpty$ **do**
  $(\boldsymbol{q}_v, \Sigma_v, E_v) \leftarrow Q.pop$
  $X \leftarrow \emptyset$
  while not robot zatrzymany **do**
    $\boldsymbol{p}_{last} \leftarrow \boldsymbol{p}_j$
    $Fs(\boldsymbol{q}_v, \boldsymbol{q}_i, \Sigma_v)$
    $e_v \leftarrow (\boldsymbol{p}_{last}, \boldsymbol{p}_j, \Sigma_v)$
    if $e_v \in E_v$ **then**
      break
    $E_v \leftarrow E_v \cup \{e_v\}$
    $X \leftarrow merge(X, X_n)$
    if $\sigma_{vj} =?$ **then**
      $\Sigma_v \leftarrow \Sigma_v$ ze zmienną $\sigma_{vj} = 1$
      $\Sigma_{v2} \leftarrow (\Sigma_v$ ze zmienną $\sigma_{vj} = -1$
      $Q.push(\boldsymbol{q}_v, \Sigma_{v2}, X)$
  if robot zatrzymany bez kolizji **then**
    $D \leftarrow D \cup \{(\Sigma_v, X)\}$
$normalize(D)$
RETURN $\Sigma_o \leftarrow max(D)$

5.5. Rozszerzenie algorytmu optymalizacji i sterownika

Dla niektórych układów przeszkód istnieją obszary pola $\boldsymbol{H}^*$, w których składnik odpowiadający za śledzenie jest zdominowany przez wektor odpychający od przeszkód, tj. $\boldsymbol{h}^* << \boldsymbol{h}_o^*$. Powoduje to wytworzenie wiru w polu $\boldsymbol{H}^*$. Wjazd robota w ten wir skutkuje cykliczną ścieżką wynikową robota i brakiem zbieżności do punktu referencyjnego. W celu przeciwdziałania temu efektowi, do prawa sterowania prezentowanego wcześniej dodano Algorytm 3, który łagodzi wymagania co do dokładności realizacji punktu referencyjnego w przypadku wykrycia wspomnianego zjawiska. Skutkuje to wcześniejszym przełączeniem robota na realizację kolejnego punktu i umożliwia ominięcie osobliwego obszaru pola $\boldsymbol{H}^*$.

Algorytm 3 Zapobieganie dominacji wektora odpychającego

if $||\boldsymbol{h}^*||^2 < 0.01||\boldsymbol{H}^*||^2$ **and** $\exists\,(\boldsymbol{p}, r, R) \in P_w : ||\boldsymbol{q}_{ri}^* - \boldsymbol{p}|| < R$ **then**
  $\epsilon_i \leftarrow 2||\boldsymbol{e}_i^*||$

6. Badania symulacyjne

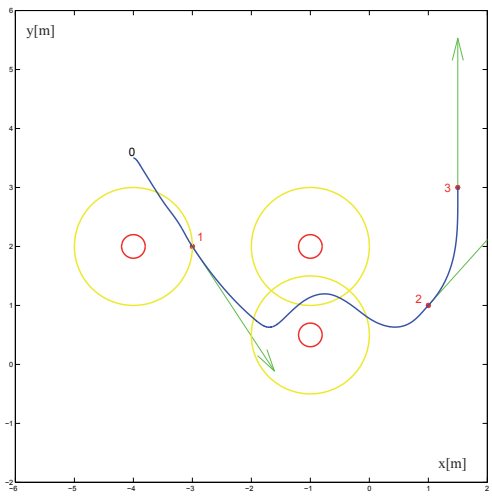
Przedstawione dalej wyniki zostały uzyskane za pomocą symulacji w środowisku MATLAB/Simulink. Czarną cyfrą 0 oznaczano punkt początkowy robota. Czerwone liczby z zie-

lonymi strzałkami lub czerwone punkty z zielonymi strzałkami, oznaczają poszczególne punkty przejazdowe. Kierunek i zwrot strzałek reprezentuje zadaną orientację robota w tych punktach. Punkty numerowane są kolejno, tak że punkt o najwyższym numerze jest punktem końcowym. Niebieskie okręgi wokół punktów przejazdowych mają promień ϵ_i w przypadku, gdy system automatycznie zwiększył wartość ϵ_i realizując Algorytm 3. Symulacje przeprowadzono z domyślną wartością $\epsilon_i = 0{,}05$ m.

Na rys. 4 i 5 przedstawiono scenariusz obrazujący przydatność prezentowanego algorytmu. Zależnie od priorytetów nadanych poszczególnym wskaźnikom, robot jedzie krótką ścieżką między dwoma przeszkodami lub dłuższą, bezpieczniejszą ścieżką. Jak widać, prezentowany system sterowania radzi sobie z nakładającymi się obszarami oddziaływania przeszkód.

Na rys. 6 i 7 zobrazowano działanie systemu sterowania dla skomplikowanych układów przeszkód. Na przykładzie tych układów widać różnorodność ścieżek w zależności od nawet drobnych zmian wag wskaźników jakości ruchu. Scenariusze te demonstrują również skuteczność Algorytmu 3.

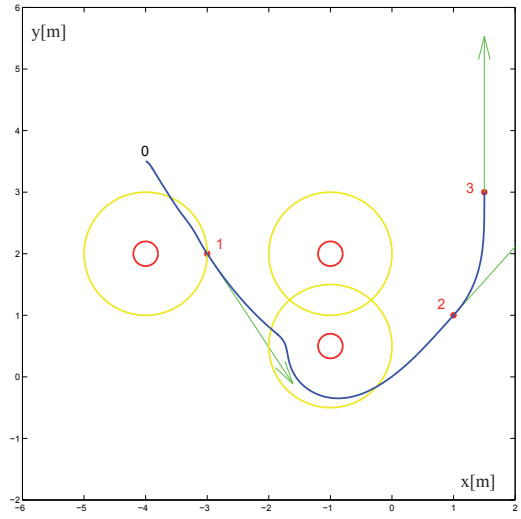
Wyniki szeregu badań symulacyjnych pokazują, że dla zachowania wysokiego bezpieczeństwa ścieżki powinno się wybierać $K_B \geq 2 \max\{K_D; K_E; K_S\}$. Zaobserwowano również podobny wpływ wskaźników χ_E i χ_S na ścieżki robota. Koszt sterowań przy realizacji ścieżki zwykle spada wraz ze wzrostem gładkości tej ścieżki. Podobnie ścieżki bezpieczniejsze są zazwyczaj dłuższe i bardziej gładkie, co czyni χ_D wskaźnikiem przeciwstawnym do pozostałych. Zależności te można wykorzystywać do uzyskiwania bardziej naturalnych wynikowych ścieżek robota. Dobrym punktem bazowym do strojenia wag wskaźników jest przyjęcie $\{K_B = 2; K_E = 1; K_D = 1; K_S = 1\}$.



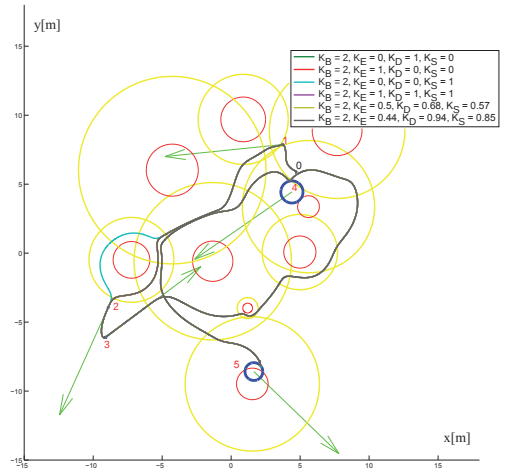
Rys. 4. Wynikowa ścieżka robota dla prostego układu przeszkód z wagami: $\{K_B = 2; K_D = 1; K_S = 1\}$
Fig. 4. Robot path obtained after simulation with simple layout of obstacles and weights: $\{K_B = 2; K_D = 1; K_S = 1\}$

7. Podsumowanie

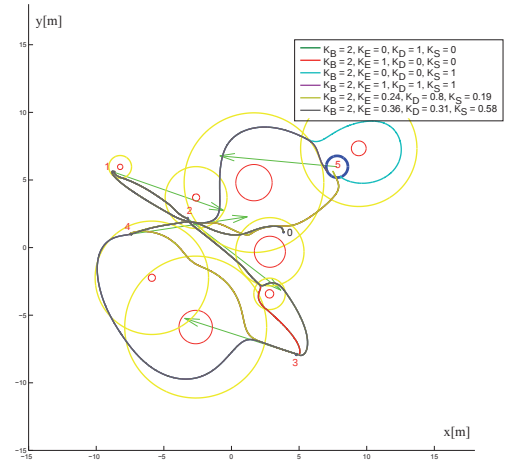
W pracy przedstawiono system sterowania i optymalizacji ruchu wykorzystujący sterownik VFO dla zadania przejazdu



Rys. 5. Wynikowa ścieżka robota dla prostego układu przeszkód z wagami: $\{K_B = 2; K_D = 0; K_S = 1.5\}$
Fig. 5. Robot path obtained after simulation with simple layout of obstacles and weights: $\{K_B = 2; K_D = 0; K_S = 1.5\}$



Rys. 6. Rezultaty działania prezentowanego algorytmu w skomplikowanym układzie przeszkód
Fig. 6. Presented algorithm's performance in presence of complex layout of obstacles



Rys. 7. Rezultaty działania prezentowanego algorytmu w skomplikowanym układzie przeszkód
Fig. 7. Presented algorithm's performance in presence of extremely complex layout of obstacles

przez zbiór punktów referencyjnych. Przedstawiona analiza proponowanego algorytmu optymalizacji wskazuje na jego przydatność w założonych zastosowaniach. Przedstawione wyniki badań symulacyjnych pokazują prawidłowe działanie proponowanego algorytmu. Prezentowany system radzi sobie z przeszkodami nakładającymi się na siebie i przysłaniającymi punkty referencyjne. Wydaje się, że proponowany algorytm optymalizacji ruchu może działać w czasie rzeczywistym. Podczas projektowania algorytmu optymalizacji ruchu brano pod uwagę strukturę sterownika VFO. Pozwoliło to na uzyskanie systemu z wysoce zintegrowanymi komponentami i przyczyniło się do zmniejszenia stopnia skomplikowania końcowego rozwiązania poprzez sformułowanie prostszego problemu optymalizacyjnego.

Podziękowania

Praca finansowana przez NCBiR w ramach Programu Badań Stosowanych z grantu PBS1/A3/8/2012, nr projektu 2224K.

Bibliografia

1. M. Michałek, W. Kowalczyk, and K. Kozłowski, "Strategia śledzenia trajektorii z unikaniem kolizji dla robota mobilnego klasy (2,0)," *Prace Naukowe Politechniki Warszawskiej. Elektronika*, z. 175, t. 2, 381–390, 2010.
2. M. Michałek and K. Kozłowski, "Motion planning and feedback control for a unicycle in a way point following task: The VFO approach," *Int. J. Appl. Math. Comput. Sci.*, vol. 19, no. 4, 533–545, 2009.
3. T. Gawron, "Planowanie ruchu i sterowanie robotem mobilnym dla zadania przejazdu przez zbiór punktów w środowisku kolizyjnym," Master's thesis, Politechnika Poznańska, Katedra Sterowania i Inżynierii Systemów, 2013.
4. D. Fox, W. Burgard, and S. Thrun, "The dynamic window approach to collision avoidance," *Robotics Automation Magazine, IEEE*, vol. 4, no. 1, 23–33, 1997.
5. Y. Cen, L. Wang, and H. Zhang, "Real-time obstacle avoidance strategy for mobile robot based on improved coordinating potential field with genetic algorithm," in *IEEE International Conference on Control Applications*, 2007, 415–419.
6. T. Schouwenaars, J. How, and E. Feron, "Receding horizon path planning with implicit safety guarantees," in *American Control Conference, 2004. Proceedings of the 2004*, vol. 6, 2004, 5576–5581.
7. M. Kim and Y. Kim, "Multiple UAVs nonlinear guidance laws for stationary target observation with waypoint incidence angle constraint," *International Journal of Aeronautical and Space Sciences*, vol. 14, no. 1, 67–74, 2013.
8. D. Nelson, D. Barber, T. McLain, and R. Beard, "Vector field path following for miniature air vehicles," *Robotics, IEEE Transactions on*, vol. 23, no. 3, 519–529, 2007.
9. C. Kyung Hyun, N. Minh Ngoc, and R. M. Asif Ali, "A real time collision avoidance algorithm for mobile robot based on elastic force," *International Journal of Aerospace and Mechanical Engineering*, vol. 2, no. 4, 230–234, 2008.
10. M. Michalek and K. Kozłowski, "Vector-field-orientation feedback control method for a differentially driven vehicle," *Control Systems Technology, IEEE Transactions on*, vol. 18, no. 1, 45–65, 2010.

Optimal waypoint following strategy for unicycle in cluttered environment

Abstract: A strategy for realization and optimization of unicycle's motion during waypoint following task in a highly cluttered environment is presented in this paper. Environment boundaries are assumed to be known beforehand and represented as a map. Proposed system is a marriage of VFO controller, collision avoidance strategy proposed before and novel optimization algorithm based on simulation of multiple virtual robots. Properties of decision space formed by parameters of assumed collision avoidance strategy are analyzed. Directed graph based representation of decision space is proposed and analyzed. Design of underlying optimization algorithm stems from this analysis. Design of the algorithm is followed by memory complexity estimation. Additionally two methods for elimination of cyclic robot paths in extremely complex obstacle configurations are proposed. Effectiveness of presented approach is illustrated by selected results of extensive simulation experiments.

Keywords: unicycle, VFO, collision avoidance, prediction, optimization

Artykuł recenzowany, nadesłany 30.01.2014, przyjęty do druku 28.07.2014.

mgr inż. Tomasz Gawron

Absolwent Wydziału Informatyki Politechniki Poznańskiej. W 2013 r. uzyskał tytuł magistra inżyniera na kierunku Automatyka i Robotyka. Aktualnie jest doktorantem w Katedrze Sterowania i Inżynierii Systemów Politechniki Poznańskiej. Jego prace badawcze dotyczą sterowania systemami nieholonomicznymi z ograniczeniami na stan oraz planowania ruchu dla robotów mobilnych. Interesuje się również problemami w dziedzinie inżynierii oprogramowania systemów robotycznych.

e-mail: tomasz.gawron@doctorate.put.poznan.pl

dr inż. Maciej Marcin Michałek

Maciej M. Michałek uzyskał stopień doktora nauk technicznych w dyscyplinie automatyka i robotyka na Wydziale Informatyki i Zarządzania Politechniki Poznańskiej w 2006 r. Aktualnie pracuje na stanowisku adiunkta w Katedrze Sterowania i Inżynierii Systemów Politechniki Poznańskiej. Jego bieżące zainteresowania i prace badawcze dotyczą algorytmiki sterowania systemami dynamicznymi, szczególnie w zakresie robotyki mobilnej.

e-mail: maciej.michalek@put.poznan.pl