AUTOMATYKA • 2009 • Tom 13 • Zeszyt 3

Maciej Garbacz*, Mieczysław Zaczyk*

Algorytmy ruchu w nieznanym otoczeniu dla robota Khepera III

1. Wprowadzenie

W artykule zostanie przedstawiona implementacja algorytmu omijania przeszkód dla robota mobilnego z wykorzystaniem środowiska programowego MATLAB/Simulink. Do badań użyto robota Khepera III poruszającego się po specjalnym blacie ograniczonym bandami z możliwością ustawiania przeszkód. Aby robota można było traktować jako jednostkę autonomiczną, konieczne jest wyposażenie go w czujniki zbierające informacje o otoczeniu [3]. Do prawidłowego działania algorytmu realizującego określony cel konieczna jest znajomość przestrzeni otaczającej robota. Wykorzystano czujniki ultradźwiękowe i podczerwieni. Przedstawione algorytmy mają za zadanie omijanie występujących w obszarze przeszkód. Bardziej zaawansowane algorytmy nawigacji umożliwiają zadawanie robotowi dowolnego dopuszczalnego położenia w przestrzeni bądź realizację poleceń typu „jedź wzdłuż ściany" czy „podążaj środkiem wolnej przestrzeni", „idź do celu" [5]. W zadaniach nawigacji możliwe jest również wykorzystywanie dodatkowych urządzeń zewnętrznych, takich jak np. kamera umieszczona ponad obszarem roboczym. Buduje się wówczas mapę, wykorzystując informacje z obrazu kamery. Zaawansowane zagadnienia nawigacji wymagają zaangażowania sztucznej inteligencji. Nadrzędny układ decyzyjny musi odpowiednio reagować na pojawiające się nieraz sprzeczne ze sobą sygnały z podsystemów pomiarowych. Przedstawione w pracy algorytmy bazują na informacjach pochodzących z dwóch rodzajów czujników – ultradźwiękowych i zbliżeniowych. Robot na bieżąco śledzi sytuację w otaczającej go przestrzeni, dzięki czemu zdolny jest do reakcji w przypadku zagrożenia kolizji np. z drugim poruszającym się robotem.

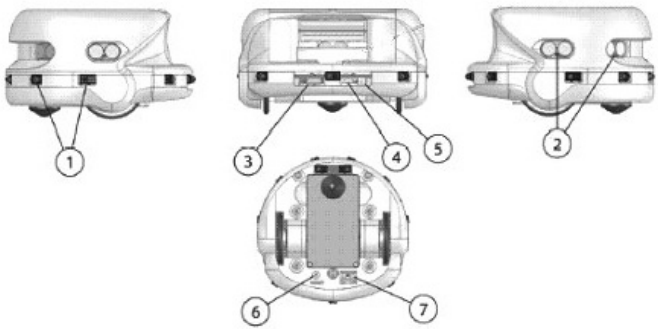
2. Robot Khepera III

2.1. Budowa

Robot porusza się na dwóch kółkach, umieszczonych w jednej osi. Kółka obleczono gumą dla uzyskania lepszej przyczepności. Dodatkowo Khepera III podparty jest w jednym

* Katedra Automatyki, Akademia Górniczo-Hutnicza w Krakowie

punkcie, co zapewnia stabilne poruszanie się w przestrzeni roboczej. Dookoła obudowy robota umieszczono dziewięć czujników podczerwieni oraz dodatkowe dwa czujniki zbliżeniowe pod spodem, umożliwiające wykrywanie na niewielką odległość przeszkód bądź krawędzi stołu, po którym robot się porusza (rys. 1). Robot ma również możliwość zmierzenia odległości od przeszkody, za pomocą pięciu wbudowanych sensorów ultradźwiękowych (sonary). Do napędu wykorzystano dwa wysokiej klasy silniki DC (jeden dla każdego koła) zapewniające sprawne i dokładne sterowanie ruchem robota. Obydwa koła robota napędzane są silnikami DC sprzężonymi z przekładnią o przełożeniu 43,2:1. Silniki maja własne wbudowane enkodery przyrostowe, umiejscowione na osi silnika, dające 16 impulsów na obrót wału silnika. W sumie daje to 691,2 impulsów na obrót koła, co odpowiada 54 impulsom na milimetr przejechanej drogi (średnica koła wynosi 41 mm, co daje 128,8 mm przejechanej drogi na pełny obrót tarczy koła). Maksymalna osiągalna prędkość robota wynosi 298 mm/s [2].

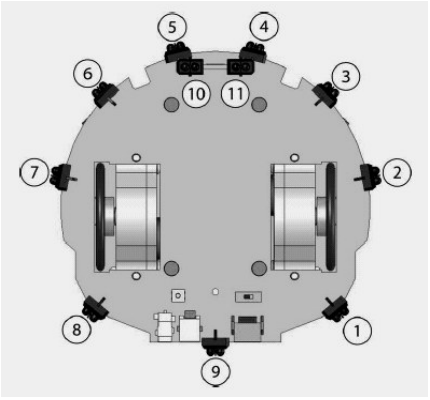


Rys. 1. Widok robota Khepera III: 1 – czujnik podczerwieni, 2 – czujniki ultradźwiękowe, 3 – główny port komunikacyjny, 4 – port miniAB USB, 5 – port ładowania/zasilania, 6 – reset, 7 – wł/wył

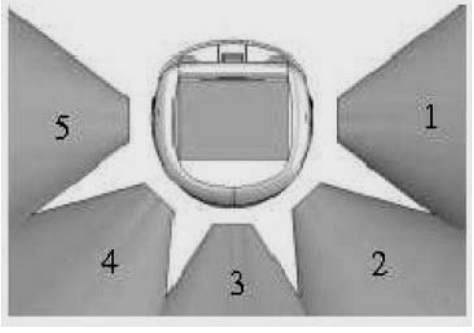
Rysunek 2 przedstawia rozmieszczenie czujników zbliżeniowych, a rysunek 3 – sonarów. Czujniki zbliżeniowe nie dają informacji o odległości od przeszkody, jedynie o jej bliskości (odczyt rośnie w miarę zbliżania się do przeszkody). Czujniki podczerwieni mogą również pracować jako czujniki światła. Zastosowane w robocie sonary dają pomiar odległości w zakresie od 20 cm do 4 m [2].

Podstawowym sposobem komunikacji pomiędzy komputerem a robotem mobilnym Khepera III jest protokół komunikacji szeregowej RS232. Podczas takiej komunikacji komputer pracuje jako „master" a robot Khepera jako „slave". Każde połączenie z robotem jest inicjowane przez komputer, a komunikacja realizowana jest poprzez przesyłanie komunikatów ASCII: każde pojedyncze połączenie składa się z dwóch części:

- rozkazu wysyłanego z komputera: rozpoczynającego się dużą literą, po której następują (jeśli są konieczne) numeryczne lub znakowe parametry oddzielone przecinkiem;
- odpowiedzi, wysyłanej z robota do komputera: rozpoczynającej się małą literą (taką jak w rozkazie), po której następują (jeżeli rozkaz dotyczy odczytu stanu czujników) numeryczne parametry odpowiedzi oddzielone przecinkami.



Rys. 2. Rozmieszczenie czujników zbliżeniowych (widok od spodu)



Rys. 3. Rozmieszczenie sonarów

Dostępne rozkazy można podzielić na dwie grupy:

1) rozkazy dotyczące konfiguracji robota (ustawienie parametrów protokołu szeregowego, ustawianie parametrów regulatorów położenia i prędkości, ustawianie parametrów sonarów);
2) rozkazy związane ze sterowaniem robota (zadawanie pozycji, zadawanie prędkości, odczyt czujników zbliżeniowych, odczyt czujników światła, odczyt odległości z sonarów).

Taki sposób komunikacji z robotem Khepera umożliwia programowanie przy użyciu dowolnego oprogramowania udostępniającego łączność poprzez port szeregowy COM. Na bazie udostępnionych dla robota rozkazów zrealizowany został zestaw funkcji pozwalających na programowanie w środowisku MATLAB-a [2]. Robot wyposażony jest standardowo w urządzenie Bluetooth, zapewniające bezprzewodową komunikację pomiędzy nim a komputerem sterującym.

2.2. Kinematyka

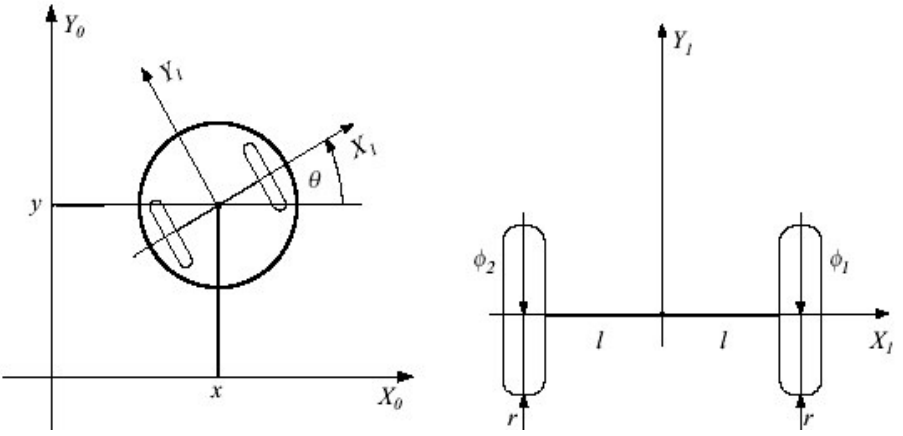
Model kinematyki dla robota mobilnego Khepera (rys. 4) przyjmuje postać:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \\ \dot{\phi}_1 \\ \dot{\phi}_2 \end{bmatrix} = \begin{bmatrix} -\sin\theta & 0 \\ \cos\theta & 0 \\ 0 & 1 \\ \frac{-1}{r} & \frac{-l}{r} \\ \frac{1}{r} & \frac{-l}{r} \end{bmatrix} \cdot \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \tag{1}$$

gdzie:

u_1 – prędkość liniowa robota,

u_2 – prędkość zmian orientacji robota.



Rys. 4. Model kinematyki

Z modelu kinematyki (1) można wyznaczyć równania na u_1 i u_2:

$$u_1 = r \cdot \frac{\dot{\phi}_2 - \dot{\phi}_1}{2}$$

$$u_2 = -r \cdot \frac{\dot{\phi}_1 + \dot{\phi}_2}{2 \cdot l} \tag{2}$$

Przekształcając równanie kinematyki (1), przy uwzględnieniu zależności (2) otrzymujemy związek między składowymi prędkości robota a prędkościami kół

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \frac{r \cdot \sin\theta}{2} & -\frac{r \cdot \sin\theta}{2} \\ -\frac{r \cdot \cos\theta}{2} & \frac{r \cdot \cos\theta}{2} \\ -\frac{r}{2 \cdot l} & -\frac{r}{2 \cdot l} \end{bmatrix} \cdot \begin{bmatrix} \dot{\phi}_1 \\ \dot{\phi}_2 \end{bmatrix} \tag{3}$$

3. Implementacja algorytmów ruchu

Na potrzeby implementacji i testowania algorytmów nawigacji i sterowania robota Khepera III w nieznanym otoczeniu wybrane zostało środowisko MATLAB/Simulink (R2007b). Komunikacja z robotem wykonywana jest poprzez pakiet funkcji zrealizowanych dla tego środowiska [2]. W zaimplementowanych algorytmach dla oceny interakcji z otoczeniem wybrane zostały sonary i czujniki zbliżeniowe. Informacja z sonarów wykorzystywana jest dla wykrycia przeszkód i modyfikacji trasy w dużej odległości od robota. Jeżeli odległość spada poniżej pewnej zadanej wartości (gdy pomiar z sonarów nie jest wiarygodny), algorytmy przełączają się na działanie posiłkujące się informacją z czujników zbliżeniowych. Poniżej przedstawione zostały trzy wybrane algorytmy poruszania się robota Khepera III w nieznanym otoczeniu z omijaniem przeszkód: algorytm „decyzyjny" oraz dwa algorytmy bazujące na idei Braitenberga [1], której istotą jest bezpośrednie połączenie modułów percepcji i modułów wykonywania ruchu.

3.1. Algorytm „decyzyjny"

Schemat blokowy algorytmu w Simulinku przedstawiono na rysunku 5. Odczyt pomiarów z sonarów realizowany jest przez podsystem *'Sonar'*, odczyt czujników zbliżeniowych przez blok *'Read Proximity'* a wysyłanie zadanej prędkości do robota przez blok *'Set Speed'*. W podsystemie *'Kinematics & Visualisation'* rozwiązywane jest zadanie kinematyczne dla uzyskania pozycji i orientacji robota oraz wizualizacji ruchu. Podsystem *'Tim'* zawiera s-funkcję zapewniającą działanie układu z zadanym czasem próbkowania. Działanie algorytmu „decyzyjnego" opiera się na kilku prostych regułach:

A) Reguły dla sonarów:
   – *jeżeli* pomiar z sonarów 2, 3 i 4 *jest większy* od 25 cm, *to* robot porusza się z dużą prędkością do przodu,

– *jeżeli* pomiar któregoś z sonarów 2,3 i 4 *jest mniejszy* od 25 cm, *to* układ sterowania przełącza się na czujniki zbliżeniowe.

B) Reguły dla czujników zbliżeniowych:

– zmniejszenie prędkości ruchu (bo wykryta przez sonary przeszkoda);

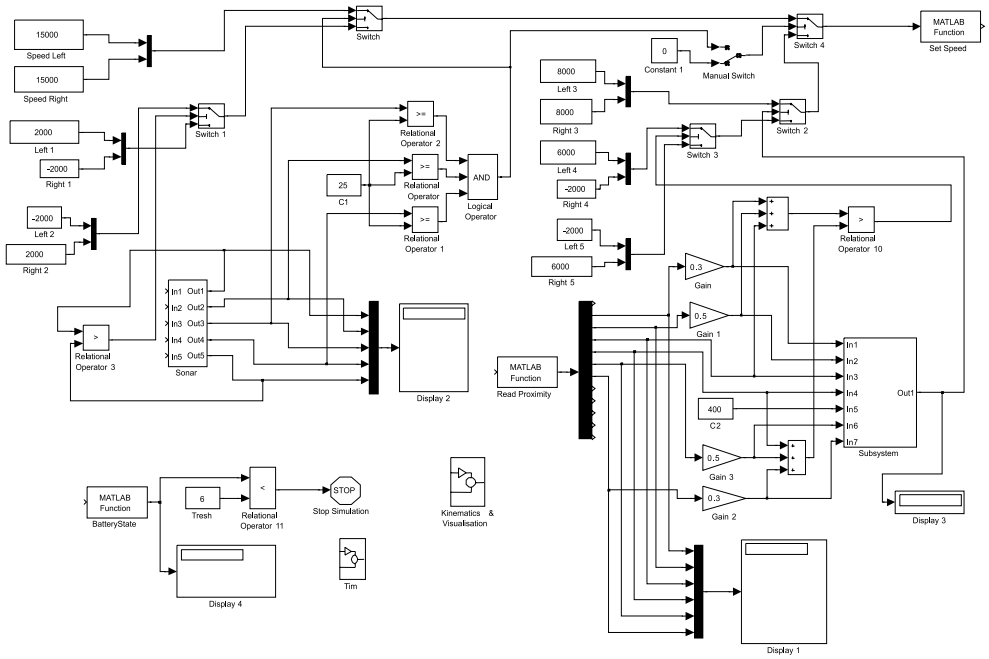
– pomiar realizowany z czujników 2, 3, 4, 5, 6 i 7 (rys. 2);

– dla pomiarów z czujników przyjęto wyznaczone doświadczalnie współczynniki wagowe;

– *jeżeli* pomiary z czujników *są mniejsze* od progu, *to* robot porusza się do przodu z zadaną prędkością;

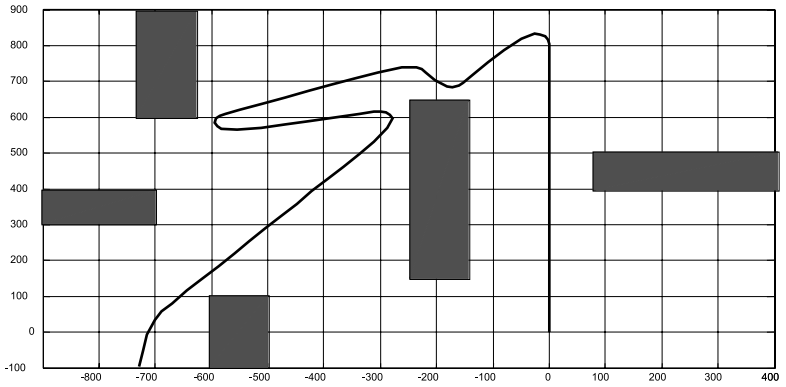
– *jeżeli* pomiar z któregoś czujnika *jest większy* od progu, *to* robot zmienia kierunek ruchu;

– *jeżeli* suma pomiarów czujników z lewej strony (2, 3 i 4) *jest większa* od sumy pomiarów czujników z prawej strony (5, 6 i 7), *to* robot skręca w prawo z zadaną prędkością (przeszkoda po lewej stronie), *jeżeli* suma pomiarów czujników z lewej strony *jest mniejsza* od sumy pomiarów czujników z prawej strony, *to* robot skręca w lewo z zadaną prędkością (przeszkoda po prawej stronie).



Rys. 5. Schemat blokowy układu z algorytmem „decyzyjnym”

Przykładowy przebieg jazdy robota w otoczeniu z przeszkodami przedstawiono na rysunku 6.



Rys. 6. Ilustracja działania algorytmu „decyzyjnego"

3.2. Algorytm „Braitenberg I"

Idea tzw. „pojazdu Braitenberga" [1] polega na bezpośrednim połączeniu modułów percepcji i wykonywania ruchu czyli połączeniu czujników z elementami wykonawczymi (napędami). Każde takie połączenie ma przypisane wagi. W zależności od zastosowanych czujników i wag robot może wykonywać różne zadania. Schemat układu sterowania wykorzystujący tę ideę przedstawiono na rysunku 7. Część wykorzystująca sonary działa identycznie jak w poprzednim algorytmie. Algorytm Braitenberga zastosowano w podukładzie związanym z czujnikami zbliżeniowymi. Wykorzystano pomiary z czujników od 1 do 8 (rys. 2) z eksperymentalnie dobranymi wagami.

Przykładowy przebieg jazdy robota przedstawiono na rysunku 8.

Na podstawie sumy pomiarów czujników z lewej i prawej strony robota wyliczane są prędkości dla koła lewego i prawego:

$$w_l \cdot \sum_{i=1}^{4}(w_i \cdot c_i) - w_p \cdot \sum_{i=5}^{8}(w_i \cdot c_i) = u_l \Rightarrow v_l = v_{lbaz} + u_l \tag{4}$$

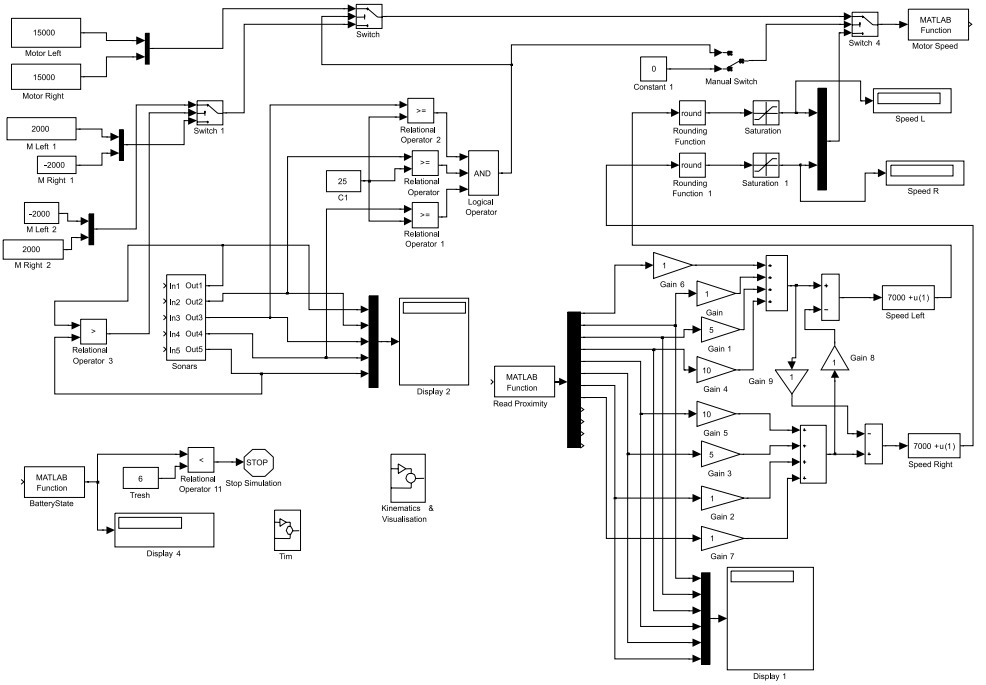
$$-w_l \cdot \sum_{i=1}^{4}(w_i \cdot c_i) + w_p \cdot \sum_{i=5}^{8}(w_i \cdot c_i) = u_p \Rightarrow v_p = v_{pbaz} + u_p \tag{5}$$

gdzie:

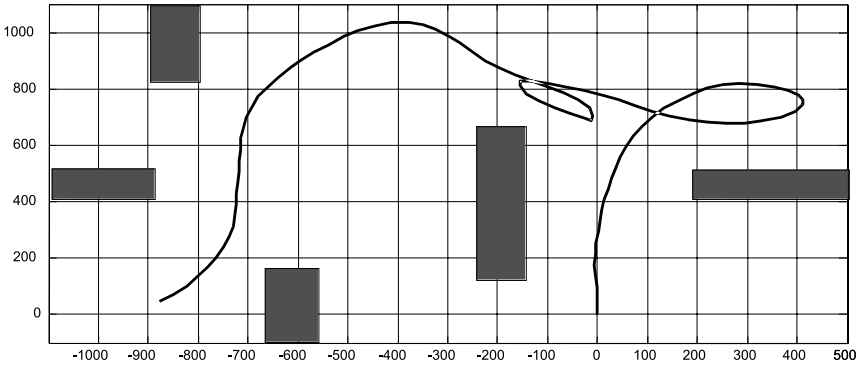
v_l, v_p – prędkość lewego i prawego koła,

v_{lbaz}, v_{pbaz} – zadana prędkość bazowa dla lewego i prawego koła,

c_i, w_i – pomiar i waga dla i-tego czujnika.



Rys. 7. Schemat blokowy układu z algorytmem „Braintenberg I”



Rys. 8. Ilustracja działania układu z algorytmem „Braintenberg I”

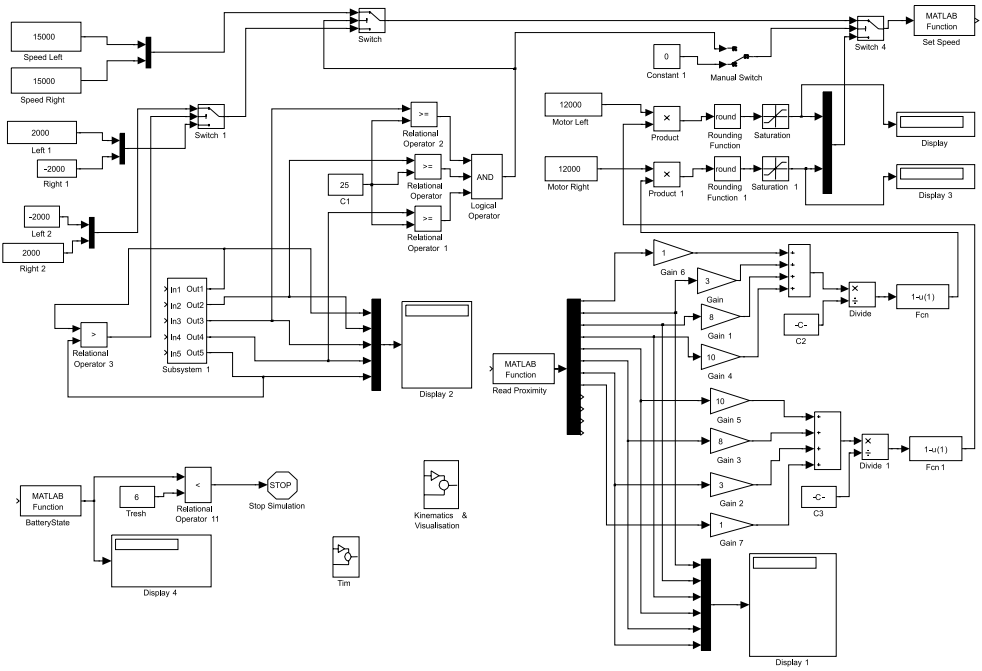
3.3. Algorytm „Braintenberg II”

Przedstawiony poniżej algorytm jest pewną modyfikacją poprzedniego polegającą na tym, że stan czujników z lewej strony wpływa na prędkość koła prawego, a czujniki z prawej strony zmieniają prędkość koła lewego. Odczyty czujników zostały znormalizowane przez maksymalne możliwe do uzyskania wartości. Prędkości dla lewego i prawego koła są wyznaczane następująco:

$$\sum_{i=1}^{4}(w_i \cdot c_{iz}) = u_l \Rightarrow v_p = v_{pbaz} \cdot (1-u_l) \tag{6}$$

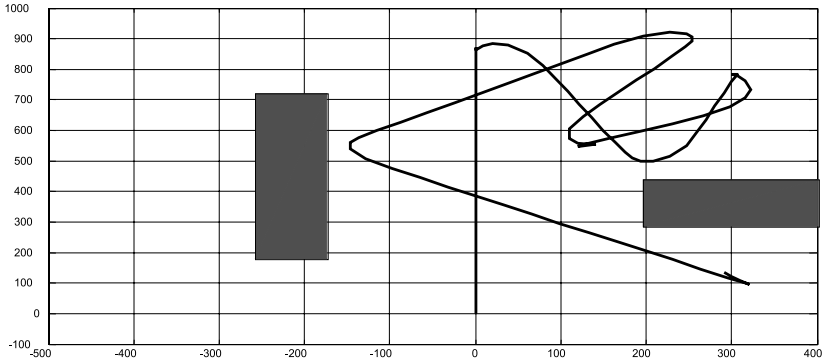
$$\sum_{i=5}^{8}(w_i \cdot c_{iz}) = u_p \Rightarrow v_l = v_{lbaz} \cdot (1-u_p) \tag{7}$$

gdzie c_{iz} – znormalizowane odczyty czujników.



Rys. 9. Schemat blokowy algorytmu „Braintenberg II”

Przykładowy przebieg jazdy robota przedstawiono na rysunku 10.



Rys. 10. Ilustracja działania układu z algorytmem „Braintenberg II"

4. Podsumowanie

Przedstawione w pracy rozważania pokazują możliwości wykorzystania bardzo wygodnego i uniwersalnego środowiska MATLAB/Simulink do implementacji i testowania algorytmów nawigacji i sterowania dla mobilnego robota kołowego Khepera III. Realizacja tego zadania była możliwa poprzez wykorzystanie stworzonego pakietu funkcji sterujących. Wybrane środowisko badawcze pozwala na łatwą zmianę i dobór parametrów algorytmów i testowanie wpływu tych zmian na zachowanie robota. Uzyskane wyniki testów przeprowadzonych dla trzech zaprezentowanych algorytmów ruchu robota w nieznanym otoczeniu z omijaniem przeszkód można uznać za zadowalające.

Literatura

[1] Braitenberg V., *Vehicles: Experiments in synthetic psychology*. MIT Press., Cambridge 1984.
[2] Garbacz M., Zaczyk M., *Robot mobilny Khepera III – oprogramowanie dla środowiska MATLAB.* Automatyka (półrocznik AGH), t. 12, z. 3, 2008, 759–767.
[3] Garbacz M., Laboratoryjny *robot mobilny Khepera II.* Automatyka (półrocznik AGH), t. 9, z. 3, 2005, 393–400.
[4] Giergiel J., Hendzel Z., Żylski W., *Kinematyka, dynamika i sterowanie mobilnych robotów kołowych*. PWN, Warszawa, 2002.
[5] Trojnacki M., Szynkarczyk P., *Autonomia robotów mobilnych – stan obecny i perspektywy rozwoju.* Pomiary Automatyka Robotyka, 9/2008, 5–9.
[6] *Khepera III User Manual*, ver. 2.1, K-Team S.A., Switzerland, 2008.
[7] MATLAB 7.0 *User Guide*. The Mathworks Inc., 2007.