prof. dr hab. inż. Krzysztof Kozłowski
dr inż. Jarosław Majchrzak
dr inż. Maciej Michałek
dr inż. Dariusz Pazderski
Politechnika Poznańska,
Katedra Sterowania i Inżynierii Systemów

SYSTEM PROGRAMOWANIA I STEROWANIA MODUŁOWEGO ROBOTA MOBILNEGO DO ZASTOSOWAŃ TRANSPORTOWYCH W POMIESZCZENIACH

Praca naukowa finansowana ze środków na naukę w latach 2007–2011 jako projekt badawczy rozwojowy R02 009 02.

Praca prezentuje koncepcję oraz realizację wybranych elementów modułowego robota mobilnego do zastosowań transportowych w pomieszczeniach. Zaprezentowano konstrukcję, elementy systemu sterowania oraz sposób programowania zadań dla robota. W projekcie uwzględniono wymogi bezpieczeństwa ludzi oraz założenie przyjaznego systemu programowania.

PROGRAMMING AND CONTROL SYSTEM OF MODULAR MOBILE ROBOT TO TRANSPORTATION TASKS IN CLOSED ENVIRONMENTS

This paper presents an idea and application of selected elements of modular mobile robot used for transportation tasks in closed environments. Some details with respect to robot's mechanical structure, its control system and programming language are described. Additionally, requirements for safety system are formulated and assumptions of user-friendly system for robot programming are discussed.

1. ROBOT MOBILNY DO ZADAŃ USŁUGOWYCH

Przedmiotem tej pracy jest opis konstrukcji robota mobilnego, który został zaprojektowany i wykonany w Katedrze Sterowania i Inżynierii Systemów na Wydziale Informatyki Politechniki Poznańskiej. Na etapie założeń projektowych przyjęto, że celem robota będzie realizacja usług, głównie o charakterze transportowym. Robot jest przeznaczony do transportu ładunków o ciężarze do 120kg, co determinuje jego gabaryty, moc zainstalowanych napędów, pojemność pokładowego źródła energii. Zdefiniowany zakres wykorzystania robota determinuje architekturę systemu sterowania, której postawą jest komputer pokładowy klasy PC z otoczeniem sieciowym i sterownikami motoryki i wieloprocesowym oprogramowaniem. Takie rozwiązanie poprawia efektywność tworzenia oprogramowania. Założenia obejmują tak że wprowadzenie języka programowania robota oraz interfejsu przyjaznego użytkownikowi.

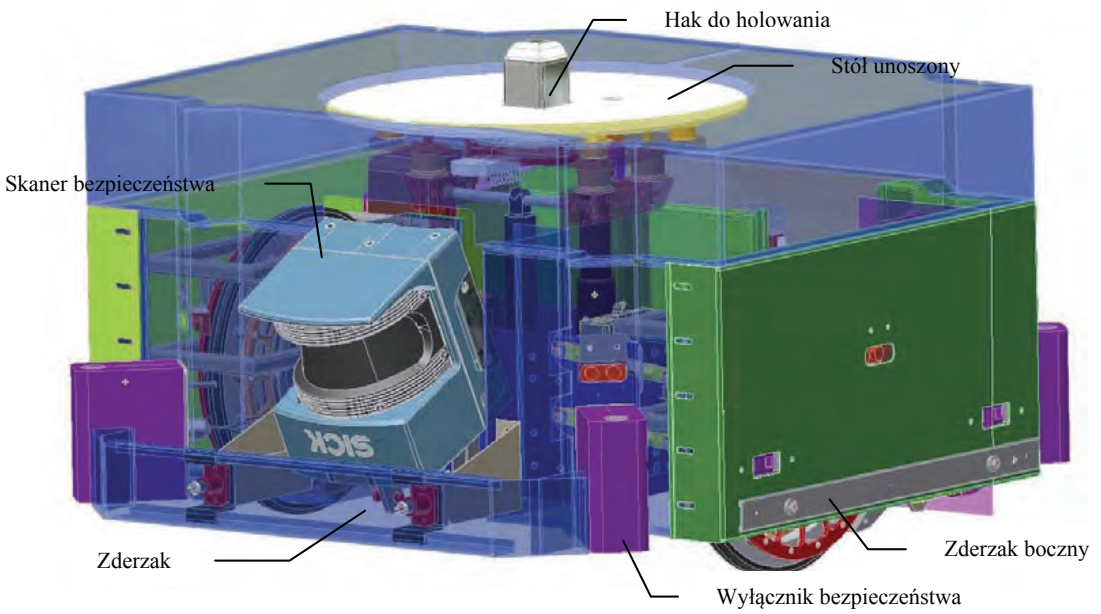
Rozpatrując różne struktury kinematyczne robotów kołowych i ich szczegółowe warianty ostatecznie wybrano do wykonania klasyczny pojazd dwukołowy z napędem bezpośrednim w układzie różnicowym oraz pasywnym samonastawnym kołem podporowym. Struktura ta pozwala uzyskać dużą mobilność (np. możliwość zmiany orientacji robota bez zmiany pozycji) i zadowalającą stateczność mechaniczną konstrukcji. System sterowania i sensoryki został zaprojektowany w sposób modułowy i wraz z mechanizmami i elementami wykonawczymi robota umożliwia holowanie innego pojazdu pasywnego, np. przyczepy. Umożliwia to odpowiednio przygotowane sterowane urządzenie zaczepowe. W wyniku wykorzystania obrotowego połączenia zaczepowego powstaje pojazd dwusegmentowy, którego

sterowanie oparte jest na kontroli pojazdu dwuko łowego. W wyniku wykorzystania po łączenia zaczepowego ustalanego pojazd podstawowy stanowi bazę zestawu zespolonego rotującego razem z nim.

Na konstrukcję modułowego robota składa się (rys. 1):

A. korpus wraz z wewnętrznymi elementami nośnymi,
B. mechanizmy zawieszeń robota z wahaczam i i uk ładami spr ężynującymi, oraz m ocowaniami hamulców,
C. system m echanicznych zderzaków, um ieszczonych wokó ł robota z prze łącznikami stanu.
D. układ podnoszenia i opuszczania zaczepu z m ożliwością blokowania sprz ęgu z przyczepą,
E. układ pozycjonowania (kąta nachylenia) skanera pomiaru odległości.

Na etapie projektowania konstrukcja by ła analizowana pod k ątem optym alizacji wagi i wymiarów zewn ętrznych. Dzi ęki zastosowaniu konstrukcji pow łokowej, która zast ąpiła konstrukcję ram ową wcze śniejszych wersji projektu robota uzyskano redukcj ę ci ężaru o ok. 50 %.

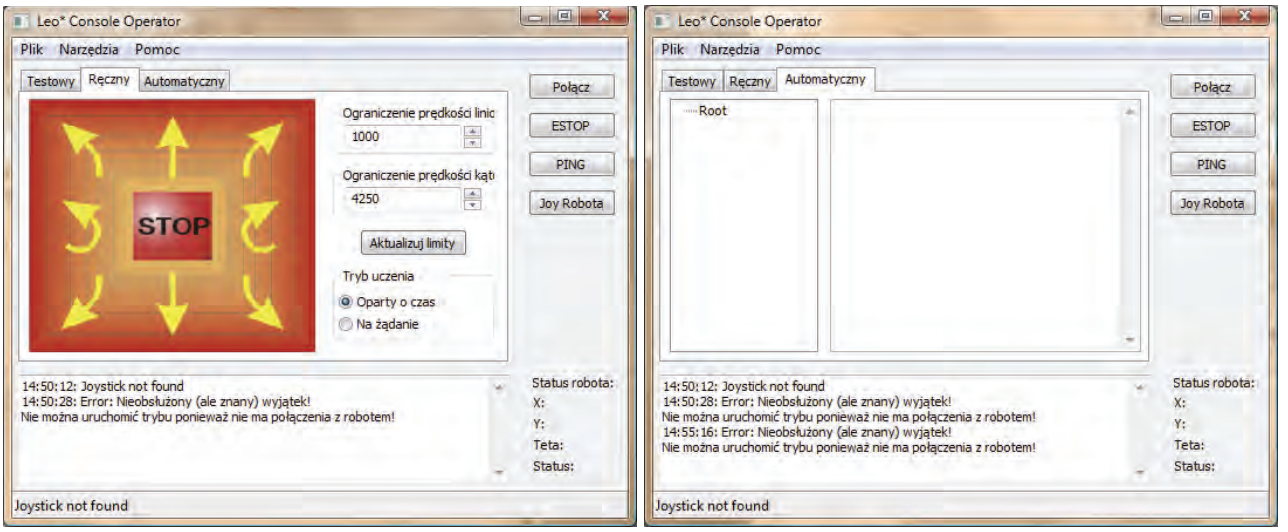


Rys. 1. Robot mobilny zatwierdzony do realizacji o roboczej nawie *LEOnardi*

2. SYSTEM PROGRAMOWANIA ROBOTA

Warto zwrócić uwagę, że odróżnieniu od klasycznych m anipulatorów przem ysłowych do tej pory nie powsta ły standardy program owania robotów m obilnych. W projekcie za łożono, że system programowania robota oparty jest o klasyczną metodę uczenia ścieżki przejazdu (analogicznie jak realizuje się to dla użytkowych robotów manipulacyjnych). W wyniku tego procesu powstaje uporządkowany zestaw punktów reprezentujących ścieżkę opisaną przez pozycję i orientację robota w ka żdym z nich. Podczas procesu program owania robota nast ępuje zapis punktów charakterystycznych przez program ator, którym jest aplikacja konsoli operatorskiej (rys. 2), funkcjonująca w systemie operacyjnym Windows. Jest to możliwe dzięki połączeniom komunikacyjnym (rys. 3) pom iędzy kom puterem programatora i kom puterem pokładowym robota, sterownikiem motoryki robota i sterownikam i osi oraz odpowiednio przygotowanym oprogram owaniem pracuj ącym na ka żdym z tych urz ądzeń. W ynikiem progra-

mowania robota jest zestaw instrukcji zapisany w postaci pliku na dysku konsoli operatorskiej.



Rys. 2. Widok okna konsoli operatorskiej a) w trybie pracy ręcznej, b) w trybie pracy automatycznej

Konsola ta umożliwia sterowanie robotem w trybie ręcznym i zapisywanie punktów ścieżki robota w procesie uczenia oraz odtwarzanie jej w trybie automatycznym jako zlecenie do wykonania dla środowiska uruchomieniowego i generatora ścieżki pracującego na komputerze pokładowym robota.

2.1. Język programowania robota

Zapis programu do wykonania przez generator ścieżki opiera się na języku programowania robota. Tryb pracy automatycznej ma służyć samoczynnemu sterowaniu robotem, w sposób powtarzalny, tak aby móc weryfikować różne algorytmy zaimplementowane w *sterowniku pokładowym*. Sterowanie to ma być realizowane przez *sterownik pokładowy*, a *komputer pokładowy* ma nadzorować ten proces, wysyłając kolejne polecenia. Wprowadzony język prostych poleceń o nazwie *LeoOS Programming Language* (w skrócie LPL) [Fel2008] został uproszczony do tego stopnia, iż jedynymi dopuszczalnymi konstrukcjami są wywołania funkcji. Składnia w tym aspekcie wzorowana jest na składni języka C. Zdefiniowano dwa terminale: terminal *identyfikator* reprezentuje prawidłową instrukcję występującą w języku i może służyć jako nazwa funkcji lub też nazwa zmiennej oraz terminal *liczba*, który reprezentuje liczbę całkowitą z opcjonalnym znakiem. Dopuszczalne znaki zdefiniowane są następująco:

- *identyfikator* = [A-Za-z_][A-Za-z_0-9]*,
- *liczba* = "-"{0,1}[0-9]+.

Bezkontekstowa gramatyka języka składa się z następujących produkcji:

- *args* → (liczba ("," liczba)*)*
- *funkcja* → identyfikator "(" args ")" ";"
- *linia* → funkcja*

Produkcja *args* definiuje dopuszczalne argumenty, jakie może przyjmować funkcja. Produkcja *funkcja* definiuje wywołanie funkcji dostępne w języku. Ostatnia produkcja, *linia* jest produkcją starto-

Tabela 1. Przykład instrukcji w języku LPL

Nr	Instrukcja
1	maxWheelVelocity(80)
2	maxRobotVelocity(500)
3	motorOn()
4	wheelVelocity(50, -50)
5	sleep(5, 500000000)
6	wheelVelocity(0, 0)
7	sleep(0, 100000000)
8	wheelVelocityPercent(100, 100)
9	sleep(5, 0)
10	robotVelocity(200)
11	ptp(1000, 1000, 90, 50)
12	ptpRelative(-1000, 1000, 0, 50)
13	motorOff()

wą definiująca jedną linię kodu. Gramatyka umożliwia zdefiniowanie poleceń dostępnych w języku LPL. Prawidłowo skonstruowany program zgodny z gramatyką, składa się z wielu linii kodu, zgodnych z produkcją startową. Program wykorzystujący polecenia prezentuje tabela 1. W tabeli 2 przedstawiono wybrane polecenia konfiguracyjne i ruchowe oraz funkcje złożone o charakterze makroinstrukcji.

Tabela 2. Zestawienie wybranych poleceń

Lp.	Nazwa polecenia	Opis	Parametry
		Polecenia konfiguracyjne	
1	motorOn();	Programowe włączenie silników robota	-
2	motorOff();	Programowe wyłączenie silników robota	-
3	maxWheelVelocity	Ustawienie maksymalnej prędkości dla kół robota	1: prędkość wyrażona w [obr/min]
4	maxRobotVelocity	Ustawienie maksymalnej prędkości robota	1: prędkość wyrażona w [mm/s]
		Polecenia wykonywania ruchu	
5	wheelVelocity	Zadawanie bezpośrednich prędkości kół robota	1: prędkość lewego koła w [obr/min], 2: prędkość prawego koła w [obr/min]
6	wheelVelocityPercent	Zadawanie bezpośrednich prędkości kół robota w procentach w stosunku do prędkości maksymalnej robota	1: prędkość lewego koła, 2: prędkość prawego koła; prędkości wyrażone są w procentach prędkości maksymalnej
7	ptp	Wykonanie zadania ruchu do punktu	1: współrzędna X punktu w [mm], 2: współrzędna Y punktu w [mm], 3: orientacja robota w stopniach, 4: promień sąsiedztwa wokół punktu w [mm]
8	ptpRelative	Wykonanie zadania ruchu do punktu względem aktualnego położenia	parametry: 1 — przesuniecie robota w osi X w [mm], 2 — przesuniecie robota w osi Y w [mm], 3 — zmiana orientacji robota w stopniach, 4 — promień sąsiedztwa punktu w [mm]
9	sleep	Usypianie procesu sterowania, służy do przedłużania poprzedniego polecenia zadania prędkości na koła	1: czas w [sek], 2: czas w [nsek]
		Polecenia funkcji złożonych	
10	onAskPlace	Włącz poszukiwanie oznaczenia miejsca	
11	offAskPlace	Wyłącz poszukiwanie oznaczenia miejsca	
12	goGetPower	Dokowanie do źródła zasilania	
13	goLockTrailer	Dojazd i pojecie przyczepy	
14	unLockTrailer	Odczepienie przyczepy	

2.2. Architektura systemu programowania

System programowania robota składa się z oprogramowania konsoli operatorskiej, oprogramowania komputera pokładowego oraz sterownika motoryki i sterowników osi. Architektura pokładowego systemu operacyjnego wynika z opisu środowiska otaczającego komputer pokładowy czyli sterownik nadrzędny, który funkcjonować musi jako pośrednik pomiędzy operatorem i sterownikiem motoryki. Realizacja tego zadania narzuca zastosowanie rozwiązania kaskadowej hierarhii typu klient-serwer. Operator korzystający z konsoli operatorskiej jest klientem, który zleca zadania dla robota pełniącego rolę usługodawcy. Sterownik nadrzędny (komputer pokładowy) pełni role pośrednika i tłumacza, i dla operatora jest on serwerem ,

z kolei dla sterownika pokładowego (motoryki) jest on klientem. Architekturę komunikacyjną systemu programowania robota przedstawia rys. 3.

Rys. 3. Architektura klient-serwer — schemat blokowy

Komputer pokładowy w przypadku programowania robota ma wykonywać kilka podstawowych zadań. Pierwsze zadanie to zapewnienie komunikacji pomiędzy oprogramowaniem komputera pokładowego – sterownika nadrzędnego a operatorem robota do sterowania nim z poziomu aplikacji pracującej jako konsola operatorska (KO). Z tego powodu oprogramowanie wchodzące w skład systemu sterowania udostępnia bezprzewodowy dostęp dla konsoli w oparciu o protokół TCP/IP. Dzięki temu operator korzystając z komputera lub PDA, wyposażonych w karty w standardzie IEEE802.11b/g, będzie mógł sterować robotem bez wykorzystywania okablowania.

Drugim podstawowym zadaniem jest zapewnienie komunikacji pomiędzy sterownikiem motoryki (MC) a komputerem pokładowym (PC). Komunikacja ta przebiega w oparciu o przewodową przemysłową magistralę komunikacyjną CAN, pracującą w czasie rzeczywistym. Kolejne zadania pełniące ważne role w procesie programowania robota, to:

- zdalne sterowanie ręczne, realizowane przez operatora,
- programowanie robota w oparciu o uczenie wykorzystujące sterowanie ręczne z obecnością operatora przy robocie,
- realizacja zaprogramowanych zadań sterowania (np. zadanie generowania i wykonywania ścieżki),
- nadzorowanie pracy robota i systemu sterowania w celu reagowania na stany wyjątkowe.

W wyniku zaprogramowania funkcji ruchowych robota w każdym trybie realizuje je sterownik motoryki (MC), który za pomocą magistrali CAN komunikuje się ze sterownikami osi. Sterownik motoryki realizuje właściwy algorytm sterowania dla wyróżnionego typu pojazdu nieholonomicznego. Oprogramowanie tego sterownika zawiera wbudowane praktyczne rozwiązania sterowania znane z literatury przedmiotu (omówione w rozdz. 4.2), takie jak: stabilny i bezpieczny dojazd do zadanego punktu, odtwarzanie ścieżki złożonej z uporządkowanej liczby punktów, realizacja omijania przeszkód zidentyfikowanych przez oprogramowanie komputera pokładowego i zintegrowane z nim sensory, itp.

3. KONCEPCJA SYSTEMU STEROWANIA I JEJ REALIZACJA

3.1. Założenia

Zadania, do których robot został przeznaczony są zadaniami programowego odtwarzania zaplanowanej ścieżki ruchu w otoczeniu przeszkód. Aby było to możliwe, konieczna jest integracja wielu systemów sensorycznych z systemem sterowania na pokładzie robota. Sprzętowym integratorem wszystkich funkcji robota jest jego komputer pokładowy, natomiast programowym jest system zadaniowo - operacyjny oraz zaprojektowane i uruchomione w nim aplikacje. Na komputerze pokładowym pracuje system zadaniowo – operacyjny, który służy jako platforma do realizacji złożonych modułów zadaniowych, takich jak: programowanie i odtwarzanie funkcji ruchowych robota, sterowanie automatyczne, przetwarzanie i integracja

danych sensorycznych z różnych sensorów i systemów sensorycznych, lokalizacja względna i bezwzględna, sterowanie dokowaniem w oznaczonych miejscach przeznaczonych do samoczynnego poboru energii oraz zaczepiania i wyczepienia przyczep, kontrolę przeszkód w obszarze najbliższego otoczenia robota i wykrywanie kierunków i obszarów kolizyjnych z przeszkodami podczas wykonywania zadań ruchowych. Aplikacje w postaci procesów podzielonych na wątki są wykonawcami zaprojektowanych modułów zadaniowych.

Zdolność do realizacji zadań przez robota wynika z odpowiednio przygotowanej koncepcji zadaniowo – operacyjnego systemu robota oraz jej realizacji. System ten jest uruchomiony na komputerze pokładowym i korzysta z zasobów klasycznego systemu operacyjnego Linux. Każde zadanie realizowane w tym systemie operacyjnym jest podporządkowane regułom budowy systemu zadaniowego jako całości.

Podstawą systemu są wzorce procesów i kolejek komunikatów, które są atrybutami wykonywalnych procesów. Powstaje w ten sposób system wielozadaniowy, wieloprocesowy i wielowątkowy. Uruchomienie kolejnego modułu zadaniowego w systemie opiera się na wykorzystaniu wzorca procesu oraz jednego lub kilku wzorców kolejek komunikacyjnych, służących do wymiany danych pomiędzy procesami, i dalej, zapisanego w języku programowania właściwego kodu zadania, które może być podzielone z kolei na podzadania - wątki. Najważniejszą cechą systemu jest wprowadzenie braku współdzielenia przez procesy jakiejkolwiek informacji lub zasobów, a synchronizacja procesów odbywa się za pomocą komunikatów. Moduły zadaniowe nazywane są *agentami*. Atrybutami agentów (pośredników) są ściśle zdefiniowane pomiędzy nimi kanały komunikacyjne jako kolejki FIFO oraz algorytmy funkcjonalne każdego z agentów.

Tak zdefiniowany system zdaniowo-operacyjny ułatwia zespołowe tworzenie oprogramowania obejmującego wiele zadań dla robota oraz dostarcza narzędzia do realizacji dowolnie zaprojektowanej komunikacji pomiędzy agentami.

3.2. Postać systemu sterowania

Modułowy system sterowania oparty jest o składniki sprzętowe oraz aplikacje współpracujące w ramach zadań na poszczególnych elementów sprzętowych wymieniających dane się poprzez dedykowane sieci komunikacyjne (rozwiązanie sprzętowo-programowe) lub poprzez system dynamicznych kolejek komunikacyjnych (rozwiązanie programowe). W pierwszym przypadku wykorzystywane są sieci: ethernet z dostępnym kablowo-radiowym osprzętem i niezbędnym oprogramowaniem konfiguracyjnym oraz sieci CAN w dwóch postaciach: magistrala sterowań STER-CAN oraz magistrala sensoryczna SENSO-CAN. W drugim przypadku są to kolejki komunikatów jako produkty programowania dynamicznego określone wzorcem komunikatów pomiędzy procesami oraz ich wątkami w wieloprocesowym systemie sterowania. Podstawowymi elementami systemu sterowania są: sterownik motoryki wraz z sterownikami napędów robota, komputer pokładowy, którego oprogramowanie systemowe oparte jest o klasyczny system operacyjny oraz wielowątkowe aplikacje realizujące zadania programowania, odtwarzania zadania, generowania ścieżki lub trajektorii. Głównym elementem systemu sterowania jest pokładowy komputer nadrzędny, wypełniający role serwera następujących zadań:

- Sterowanie ręczne w oparciu o joystick programowy w konsoli operatorskiej (rys. 2a), umożliwiając kontrolowane prowadzenie robota w przestrzeni prędkości liniowej i kątowej lub w przestrzeni prędkości kół robota

- Sterowanie ręczne z zapamiętywaniem punktów przejechanej ścieżki, z możliwością korekty pozycji każdego zapamiętanego punktu
- Sterowanie automatyczne w zamkniętej pętli sprzężenia zwrotnego z zadanymi punktami ścieżki z prostą korekcją sensoryczną
- Sterowanie automatyczne w zamkniętej pętli sprzężenia zwrotnego z uwzględnieniem zaawansowanych systemów sensoryki wizyjnej oraz pomiarów odległości i lokalnej chwilowej mapy otoczenia.

4. STEROWANIE NISKOPOZIOMOWE

4.1. Architektura sterownika pokładowego

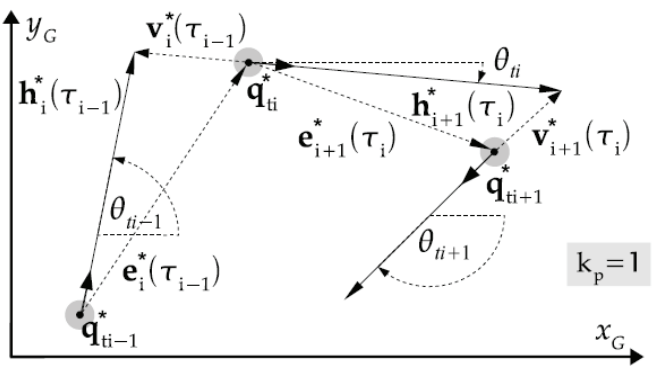
Sterownik pokładowy robota to urządzenie odpowiedzialne za wykonywanie poleceń przekazywanych z komputera pokładowego i realizacji za pomocą elementów wykonawczych, sterowników osi połączonych z silnikami wykonawczymi, realizacji pomiarów i tworzenia systemu pomiarowego opartego o odometryczne przetworniki pomiarowe. Zasadniczą funkcją tego sterownika jest realizacja elementarnego zadania sterowania za pomocą niskopoziomowego algorytmu sterowania z zaprogramowaną i parametryzowaną strategią.

4.2. Algorytm sterowania niskopoziomowego

Dla projektowanego robota można wyróżnić następujące zadania ruchu:

- autonomiczny przejazd przez zbiór punktów referencyjnych/przejazdowych w środowisku z przeszkodami,
- parkowanie w stacji ładowania akumulatorów,
- podjazd i podjęcie pasywnej przyczepy,
- ruch z przyczepą zespoloną z robotem.

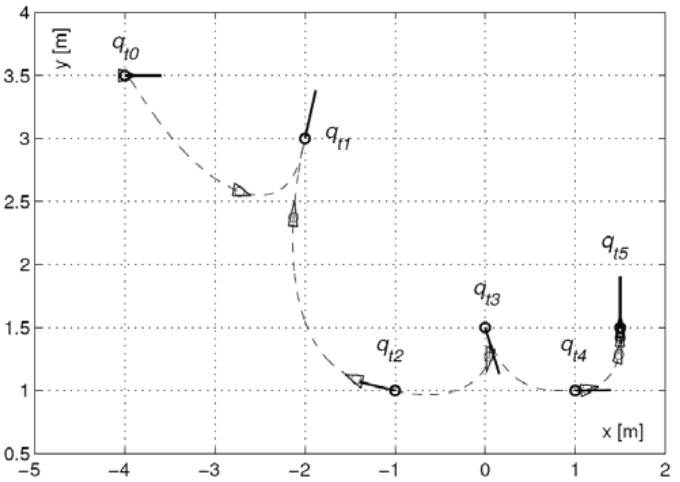
Pierwsze z nich związane jest z praktycznym uproszczeniem zagadnienia odtwarzania w pełni zdefiniowanej ścieżki/trajektorii. Polega ono na zaplanowaniu przejazdu przez zbiór punktów $\boldsymbol{S}_t = \{\boldsymbol{q}_0; \boldsymbol{q}_{1t}; : : : ; \boldsymbol{q}_{Nt}\}$, taki, że pierwszy punkt pokrywa się ze stanem początkowym robota, natomiast punkt $\boldsymbol{q}_{Nt}$ definiuje docelową pozycję i orientację pojazdu. W pracy [MiKo2009_1] zaproponowano prosty algorytm planowania i następującego po nim etapu realizacji przejazdu przez punkty ze zbioru $\boldsymbol{S}_t$ z zadaną i odcinkami stałą prędkością postępową oraz z etapem płynnego zatrzymania robota w punkcie docelowym . Etap planowania polega na określeniu orientacji referencyjnych w poszczególnych punktach pośrednich $\boldsymbol{q}_{it}$ dla $i = N–1...,1$ rozpoczynając obliczenia od punktu docelowego, jak zaznaczono na schemacie z rys. 4.



Rys. 4. Strategia planowania ruchu przez zbiór punktów przejazdowych

Przedstawiona koncepcja planera ruchu oraz metoda sterowania ze sprzężeniem zwrotnym wynika z zastosowania i rozszerzenia autorskiej metody VFO (metody orientowania pól wek-

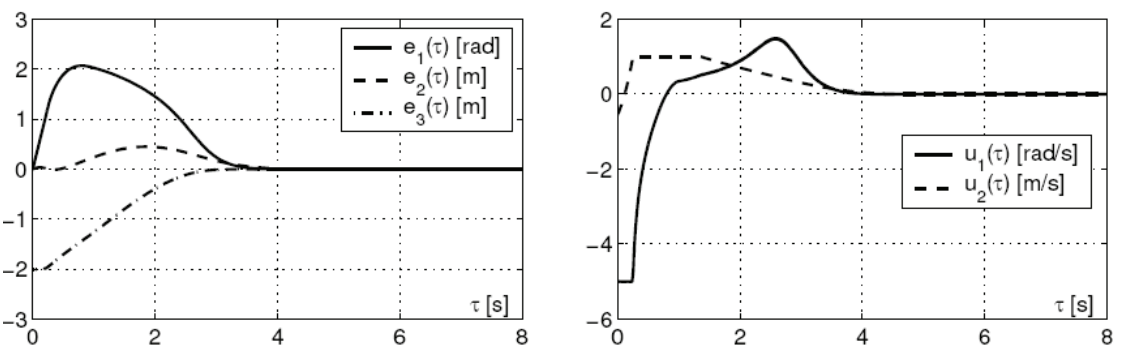
torowych). Elastyczność proponowanego podejścia wynika z możliwości swobodnego określania strategii przejazdu robota (przodem /tyłem) przez bieżący punkt ze zbioru $\boldsymbol{S}_t$ oraz z możliwości prostego i praktycznie użytecznego kształtowania stanów przejściowych poprzez wykorzystanie tzw. *efektu naprowadzania* pojazdu i związanego bezpośrednio z wartością parametrów sterownika VFO. Przykładowe wyniki działania systemu planowania i realizacji ruchu z różnymi strategiami przejazdu przez punkty pośrednie przedstawia rys. 5.



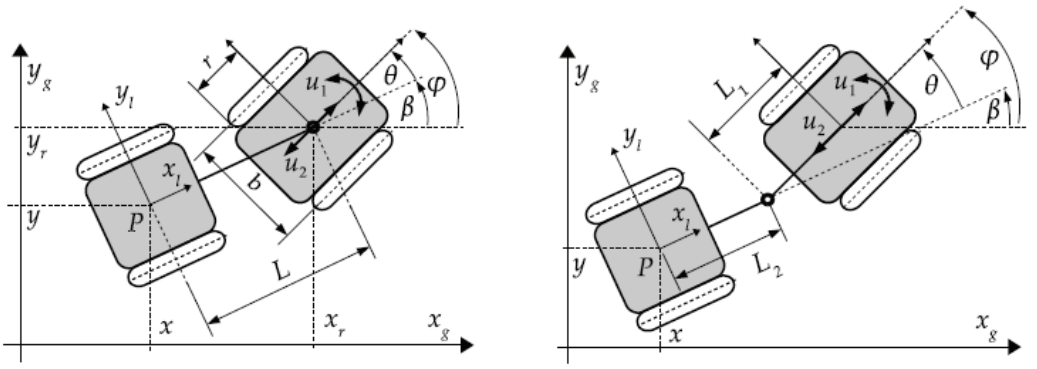
Rys. 5. Przykładowy wynik działania algorytmu planowania i realizacji ruchu przez zbiór punktów przejazdowych z wykorzystaniem algorytmu VFO

Zadanie parkowania robota związane jest z problemem sterowania do punktu, które dla systemów z nałożonymi więzami nieholonomicznymi jest problemem nietrywialnym. W pracy [MiKo2009_1] zaproponowano metodę sterowania do punktu dla robota o kinematyce (2,0) z napędem różnicowym przy czym uwzględniono ograniczenia na dopuszczalne prędkości napędów pojazdu. Prędkości dopuszczalne mogą tu wynikać z fizycznych ograniczeń związanych z zastosowanymi napędami lub też mogą zostać nałożone sztucznie (programowo) tak, aby zachować warunki bezpiecznego ruchu robota w rzeczywistym środowisku z przeszkodami i w obecności człowieka.

Podjęto także problem sterowania z czasie skończonym, które gwarantuje zbieżność uchybów w skończonym interwale czasowym $\tau \in [0, T]$, przy czym horyzont sterowania $T < \infty$ można oszacować *a priori*. Ponadto zastosowany algorytm sterowania zapewnia większą odporność układu zamkniętego (w porównaniu ze stabilizatorami i asymptotycznymi) na pewnego rodzaju długotrwałe zakłócenia działające na pojazd podczas ruchu. Rys. 6 przedstawia wyniki symulacyjne zaproponowanego układu sterowania FT-VFO dla zadania parkowania równoległego.

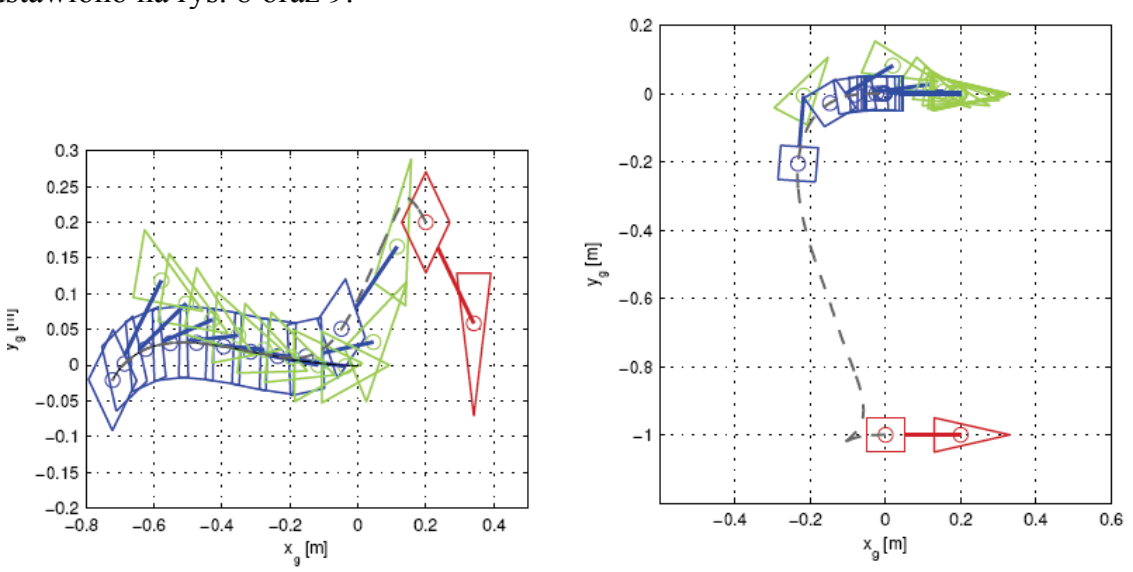


Rys. 6. Przebiegi uchybów oraz ograniczonych sygnałów sterujących uzyskane w układzie sterowania FT-VFO (*Finite-Time-VFO*) dla robota z napędem różnicowym

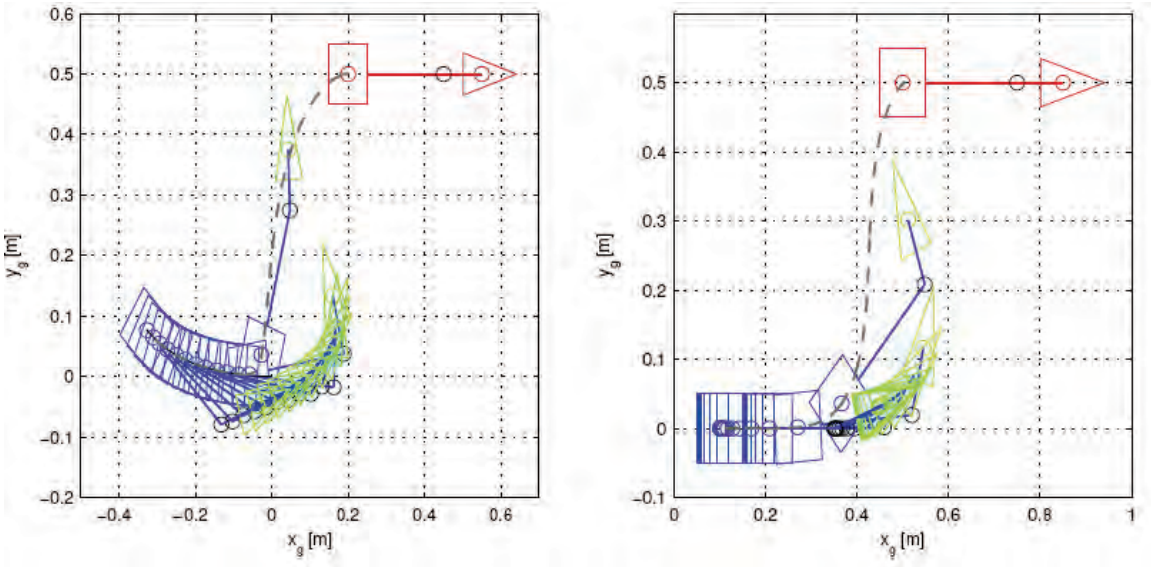


Rys. 7. Definicja zmiennych konfiguracyjnych robota przegubowego oraz dwa sposoby mocowania przyczepy do dwukołowego ciągnika z napędem różnicowym – mocowanie osiowe (lewy) i pozaosiowe (prawy)

Ruch robota mobilnego z dołączoną przyczepą należy do zadań, których trudność realizacji wynika z obecności więzów nieholonomicznych oraz ze strukturalnej niestabilności podsystemu reprezentowanego przez kąt skręcenia przyczepy przy manewrach cofania (składanie się przyczepy – *efekt scyzoryka*). Możliwe są dwa sposoby mocowania przyczepy do ciągnika: osiowe (z punktem mocowania położonym na środku osi kół ciągnika) i pozaosiowe (z punktem mocowania położonym poza osią kół ciągnika), jak przedstawiono na rys. 7. Zadania śledzenia trajektorii dopuszczalnych oraz sterowania do punktu dla pojazdów przegubowych obu typów z rys. 8 zostały przedstawione odpowiednio w pracach [MiKo2009_2] oraz [MiKo2009_3]. Zastosowano tam koncepcję VFO, podając sposób rozszerzenia jej stosowalności na roboty mobilne definiowane modelem o wymiarze wektora konfiguracji większym od trzech. Oba rodzaje zadań potraktowano w podobny zunifikowany sposób poprzez analogię do pierwotnego wykorzystania metody VFO dla ciągnika dwukołowego. Zastosowanie koncepcji VFO było możliwe dzięki wprowadzeniu odwracalnej transformacji sygnałów sterujących i przeformułowaniu równań modeli robotów przegubowych w taki sposób, aby uzyskać podsystem z modelem monocykla. Uzyskane wyniki pokazują ponownie elastyczność w wyborze strategii ruchu pojazdów: ruch przodem/ruch tyłem z zachowaniem stabilności dla podsystemu skręcenia przyczepy podczas manewrów cofania. Przykładowe wyniki symulacyjne przedstawiono na rys. 8 oraz 9.

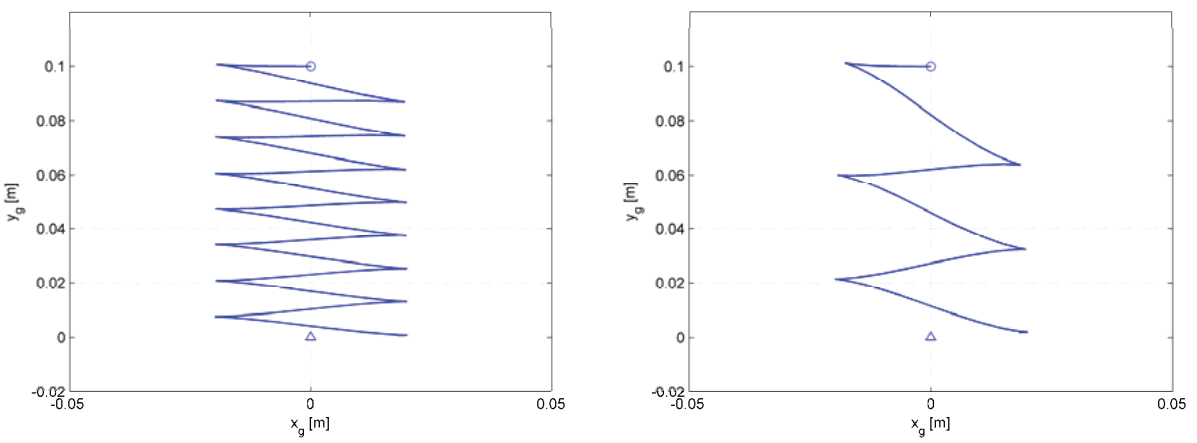


Rys. 8. Wyniki realizacji zadania śledzenia trajektorii (lewy: ruch tyłem) oraz sterowania do punktu (prawy: ruch przodem) w układzie sterowania VFO dla robota z przyczepą mocowaną osiowo



Rys. 9. Wyniki realizacji zadania śledzenia trajektorii (lewy: ruch tyłem) oraz sterowania do punktu (prawy: ruch tyłem) w układzie sterowania VFO dla robota z przyczepą mocowaną pozaosiowo

Algorytm VFO wykorzystywany do realizacji podstawowych zada ń ruchu m aksymalizuje podstawowe kierunki ruchu wyznaczane przez pola wektorowe-generatory zwi ązane z kinematyką robota. W efekcie zadanie sterowania orientacj ą robota staje si ę wtórne wobec uzyskania za łożonej pozycji. Dlatego w stanach przej ściowych zm iany orientacji robota m ogą być stosunkowo du że – w pewnych przypadkach taka realizacja ruchu nie jest akceptowalna. Biorąc pod uwag ę podane ograniczenie w opisywanym robocie opcjonalnie m ożna wykorzystać uniwersalne techniki częstotliwościowe, które pozwalają realizować ruch wysokoenergetyczny w tzw. kierunkach trudnych (wskazywa nych przez nawiasy Liego). W tym celu proponuje si ę im plementacje algorytm ów wykorzystuj ących funkcje transwersalne opracowanych przez Morina i Sam sona [MS2009]. Ilustracj ę działania tych algorytm ów przedstawiają wyniki symulacji numerycznych przedstawionych na rys. 10.



Rys. 10. Ścieżka pozycji dla robota dwukołowego w zadaniu parkowania z wykorzystaniem funkcji transwersalnych. Tunel błędu pozycji założono równy 0,02 m, dopuszczalny błąd orientacji 0,2 rad (po lewej) lub 0,5 rad (po prawej).

Kształtując odpowiednio współczynniki projektowe można uzyskać założony przebieg trajektorii pozycji (leżącej w tunelu o projektowanej szerokości) jak również ograniczenie (do zało-

żonego poziomu) zmian orientacji. Dodatkowo, uwzględniając autorskie metody strojenia tych algorytmów [PiK2008] możliwe jest ograniczanie oscylacyjności ruchu, gdy nie jest ona wymagana do realizacji zadania. Opisywana strategia może być efektywnie wykorzystana w przypadku dokowania robota i realizacji ruchu w bezpośrednim sąsiedztwie punktu zadanego.

5. SYSTEM SENSORYCZNY

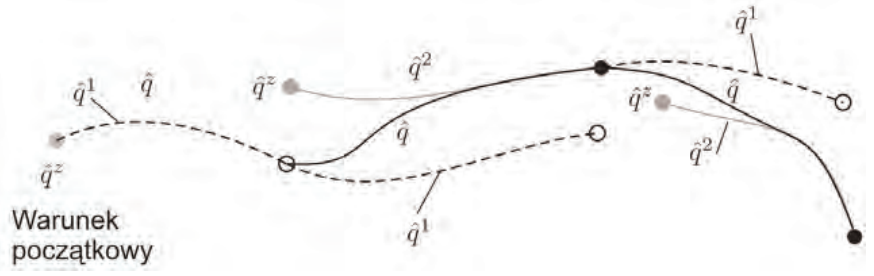
W skład układu sensoryczny robota wchodzą dwa podsystemy. Pierwszy odpowiada za kontrolę stanu wewnętrznego robota i jest bezpośrednio związanym z układem napędowym, pomocniczymi elementami wykonawczymi (obejmuje odczyt stanu liczników opisujących drogę kątową, stanu wyłączników krańcowych, wartości prądu silników, itd.) oraz układem bezpieczeństwa (m. in. pozwala na odczyt stanu zderzaków opisujący miejsce fizycznego kontaktu robota z przeszkodą).

Drugi podsystem pomiarowy jest bardziej złożony i odpowiada za analizę stanu środowiska, którego celem jest lokalizacja robota w środowisku, wykrywanie przeszkód statycznych i dynamicznych oraz sytuacji potencjalnie niebezpiecznych. Układ ten zaprojektowany jako system rozproszony wykorzystuje dalmierze ultradźwiękowe, dalmierze triangulacyjne w zakresie podczerwieni, zintegrowane żyroskopy i akcelerometry, jak również skaner laserowy i wielokamerowy system wizyjny.

5.1. Określanie pozycji i orientacji robota

Z punktu widzenia możliwości realizacji podstawowych zadań ruchu kluczowe znaczenie ma system lokalizacji. Układ ten powinien umożliwić określanie konfiguracji robota w wybranym układzie odniesienia z możliwie dużą precyzją przy zachowaniu krótkich opóźnień. W roli podstawowego systemu lokalizacji zastosowano lokalizację przyrostową realizowaną przez algorytm odometrii, co pozwala uzyskać dobre właściwości dynamiczne lokalnej pętli sprzężenia zwrotnego i dużą częstotliwość pracy układu sterowania ruchem robota. Metoda ta, będąca w istocie techniką predykcyjną bez sprzężenia zwrotnego, nie może być jednak stosowana samodzielnie. Dlatego w celu bezwzględnego określenia konfiguracji robota w wybranym układzie współrzędnych oraz w celu korekcji estymacji konfiguracji zwracanej przez odometrię wykorzystano system wizyjny. Projektując całościowy algorytm lokalizacji założono, że algorytm odometrii jest dobrze nastrojony (tj. współczynniki geometryczne takie jak rozstaw kół oraz ich promienie zostały poprawnie zidentyfikowane np. za pomocą testu UMB [Skrz2007]) i błąd systematyczny jest pomijalnie mały. Z kolei dane zwracane przez system lokalizacji zewnętrznej (w tym przypadku jest nim system wizyjny obserwujący sztuczne znaczniki o znanych współrzędnych wprowadzone do środowiska) zapewniają stabilną estymację, choć obarczoną stosunkowo dużym rozrzutem. Uwzględniając niską częstotliwość aktualizacji danych przez system wizyjny do realizacji fuzji danych zwracanych przez oba algorytmy lokalizacji klasyczne podejście w oparciu o filtr Kalmana nie wydaje się być w rozważanym przypadku właściwe. Z kolei założenie korekcji odometrii wyłącznie w oparciu o informacje podawane przez zewnętrzny algorytm lokalizacji skutkowałoby skokową zmianą wartości konfiguracji, która jest wykorzystywana przez algorytm sterowania. Zaprojektowany algorytm wykorzystuje stosunkowo proste podejście do fuzji danych w oparciu o średnią ważoną. Idea algorytmu polega na jednoczesnym obliczaniu odometrii dla różnych warunków początkowych, które wynikają ze stanu poprzedniego zwracanego przez odometrię oraz no-

wej wartości będącej liniową kombinacją poprzedniej estymacji konfiguracji i danych podawanych przez system zewnętrzny. Na rys. 11 przedstawiono ogólną zasadę algorytmu.



Rys. 11. Ilustracja algorytmu fuzji danych w układzie lokalizacji robota

Szarym kółkiem oznaczono informację o konfiguracji zwracaną przez zewnętrzny system lokalizacji, kółkiem pustym warunek końcowy estymacji konfiguracji za pomocą algorytmu odometrii w chwili pojawienia się informacji zewnętrznej, natomiast kółkiem czarnym końcową wartość estymaty kombinowanej. Ścieżka oznaczona ciągłą czarną linią jest wynikiem kombinacji liniowej danych wyznaczanych przez dwa algorytmy odometrii. Warto zwrócić uwagę na celowość wyboru funkcji wagowej do obliczania kombinacji liniowej, tak aby zapewnić ciągłość ścieżki wypadkowej pomimo wprowadzania dodatkowych danych przez zewnętrzny system lokalizacji.

5.2. Wykorzystanie skanera laserowego

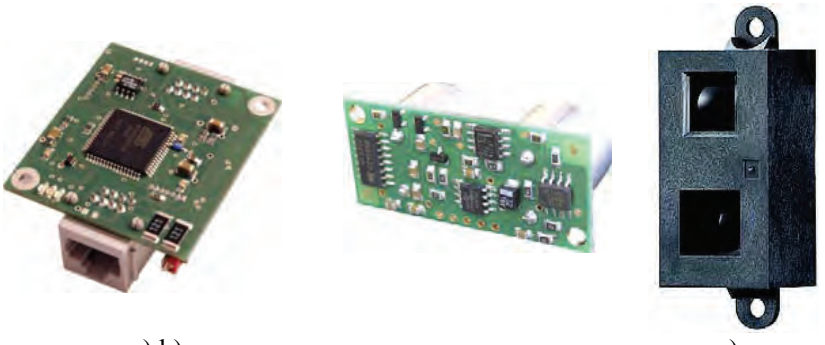
Ważnym elementem systemu lokalizacji jest skaner laserowy, który dostarcza dane do podsystemu mapowania (w postaci elementarnej mapy wektorowej) i sterowania. W bieżącej wersji układu sterowania i sensoryki robota założono, że skaner w pierwszej kolejności ma dostarczać lokalnych informacji o przeszkodach w środowisku poprzez określenie najbliższej odległości od przeszkody i kąta, pod którym jest ona widziana. Warto jednak wskazać na możliwość dalszego rozszerzenia znaczenia skanera w celu wspomagania systemu lokalizacji i korekcji odometrii.

5.3. Elementy sieciowego systemu sensorów pomiaru odległości

System składa się z ustalonej liczby modułów sensorycznych. Ustalenia projektowe pozwalają na obsługę przez jeden moduł pomiarowy pojedynczego sensora podczerwieni oraz pojedynczego sensora ultradźwiękowego [MaPaKo10]. Czujnikami współpracującymi z opisywanym w pracy modułem pomiarowym w jego podstawowej wersji są:

- czujnik ultradźwiękowy (US) SFR-04 firmy Devantech, wyposażony w przetworniki piezoelektryczne i o zasięgu od 0,03 - 3 m,
- optyczny czujnik triangulacyjny (IR) GP2Y3A002K0F firmy Sharp, pracujący w zakresie mierzonej odległości od 0,2–1,5 m i umożliwiający pomiar w pięciu kierunkach o stałej różnicy kątowej pomiędzy nimi.

Każdy z modułów pracuje w cyklu działania aktywnych sensorów do niego podłączonych, wykorzystując przerwania czasowe. Każdy sensor posiada specyficzny ze względu na zasadę działania cykl pomiarowy, co związane jest z asynchronizmem pracy sensorów. Procedura obsługi sensorów została zamknięta w formę maszyny stanu, co umożliwia niezależną transmisję wyników pomiarów natychmiast po ich uzyskaniu.



a) b) c)

Rys. 12. Elementy sieciowego systemu sensorycznego, a) moduł sensoryczny, b) sensor US, c) sensor IR

Jako elementarną jednostkę pomiarową zaprojektowano od podstaw moduł (rys. 12a) obsługujący czujnik podczerwieni (rys. 12c) oraz czujnik ultradźwiękowy (rys 12b). W zrealizowanej wersji pracuje osiem takich zestawów podłączonych do magistrali CAN. Sensory umieszczono w przewidzianych konstrukcją robota miejscach jego korpusu.

Interfejsy modułu pomiarowego zostały przygotowane do współpracy z wybranymi czujnikami odległości. Dwukierunkowa komunikacja z czujnikiem ultradźwiękowym realizowana jest metodą cyfrową i polega na cyklicznej inicjalizacji wysłania wiązki impulsów fali ultradźwiękowej oraz odczycie wyniku na podstawie pomiaru czasu trwania od momentu wysłania do chwili odbioru impulsu wyjściowego (odpowiadającego czasowi przelotu fali akustycznej od sensora do przeszkody i z powrotem). Podobnie sterowanie pracą triangulacyjnego, 5-kanałowego sensora podczerwieni realizowane jest na drodze cyfrowej (wybór kanału, tj. kąta emisji wiązki światła podczerwonego i rozpoczęcie przetwarzania), natomiast odczyt wyniku dokonywany jest poprzez pomiar wartości analogowej z wbudowanego w sensor czujnika PSD.

Z uwagi na pracę w systemie wielu sensorów danego rodzaju, kluczowym zagadnieniem jest rozwiązanie problemu przesłuchów i właściwej separacji. Ponieważ zastosowane czujniki IR lub US są tego samego typu, istnieje możliwość wzajemnego nakładania się wiązek pomiarowych wykorzystujących te same zjawiska fizyczne, skutkiem czego powstaje możliwość otrzymania błędnych wyników pomiarów. Dlatego w projektowanym układzie pomiarowym zaproponowano możliwość określania sekwencji wyzwalania czujników, opartej na jednostkowym czasie pomiaru każdego czujnika, tak aby eliminować ryzyko nakładania się wiązek pomiarowych działających identycznie. W tym dokonano synchronizacji działania modułów. Pomiarami zarządzają dwie niezależne sekwencje, a numer pomiaru w sekwencji ustalany jednoznacznie odpowiednią komendą. Dzięki temu sekwencje mogą być przeprogramowane, umożliwiając wykluczanie nakładania się sygnałów pomiarowych w racjonalnie rozmieszczonych sensorach oraz np. dopasowane sekwencji działania sensorów do kierunku ruchu robota.

5.4. Bezwładnościowy system pomiaru pozycji

Modułem uzupełniającym jest moduł IMU zbudowany w oparciu o układ ADIS16355, który odpowiada za ocenę orientacji robota w przestrzeni 3D wyrażonej względem wektora grawitacji (inklinometr), co pozwala wykrywać stopień nachylenia płaszczyzny, po której porusza się robot. Dodatkowo możliwe jest wykorzystanie tego systemu do wspomagania lokalizacji – zagadnienie to ma jednak charakter rozwojowy.

6. SYSTEM BEZPIECZEŃSTWA

System bezpieczeństwa jest złożony z funkcji poszczególnych sterowników robota. Podstawowymi elementami bezpieczeństwa są programowe ograniczenia prędkości ruchu robota, elementy kontaktowe i zderzaki na obwodzie robota, skaner laserowy (na rys. 1) z wyróżnionymi strefami bezpieczeństwa, których wielkość jest stała, ale docelowo może być modyfikowana w funkcji prędkości postępowej, oraz pomiarowy system sieciowy, który pełni podobną rolę co strefy bezpieczeństwa w skanerze laserowym, ale w przeciwieństwie do niego elementy tego systemu są skierowane we wszystkich kierunkach przemieszczenia się robota włączając ruch obrotowy.

7. PODSUMOWANIE

W pracy przedstawiono koncepcję budowy modułowego robota mobilnego z przyczepą, jego konstrukcję, system sterowania i programowania oraz zestaw sensorów, które umożliwiają autonomiczny ruch robota w środowisku częściowo uporządkowanym. Istotną rolę pełni oryginalny system sterowania zbudowany w oparciu o metodę VFO oraz znaną z literatury metodę z zastosowaniem funkcji transwersalnych. Bardzo istotny jest też system komunikacji wewnętrznej i zewnętrznej oraz nowy język programowania robota. W obecnej realizacji projektu prowadzone są testy systemu sterowania i nawigacji, których wyniki będą dostępne w czasie trwania konferencji Automation.

BIBLIOGRAFIA

[MiKo2009_1] M. Michałek, 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.*, 19(4), s. 533–545, 2009.

[MiKo2009_2] M. Michałek, K. Kozłowski. Tracking and set-point VFO control for an articulated mobile robot with on-axle hitched trailer. *2009 American Control Conference*, s. 919–924, St. Louis, USA, 2009.

[MiKo2009_3] M. Michałek, K. Kozłowski. VFO tracking and set-point control for an articulated vehicle with off-axle hitched trailer. *Proceedings of the European Control Conference 2009*, s. 266–271, Budapeszt, Węgry, 2009.

[PiK2008] D. Pazderski, K. Kozłowski. Trajectory Tracking Control of Skid-Steering Robot – Experimental Validation, Proc. of the 17th IFAC World Congress, s. 5377–5382, 2008.

[MS2009] P. Morin, C. Samson. Control of nonholonomic mobile robots based on the transverse function approach, *IEEE Transactions on Robotics*, Vol. 25 (5), s. 1058–1073, 2009.

[Skrz2007] P. Skrzypczyński, Metody analizy i redukcji niepewności percepcji w systemie nawigacji robota mobilnego, Rozprawy, nr 407, Poznań, Wyd. Politechniki Poznańskiej 2007.

[Fel08] A. Fellenberg, Pokładowy system operacyjny modułowego robota mobilnego, Praca dyplomowa magisterska, Politechnika Poznańska, Katedra Sterowania i Inżynierii Systemów, 2008.

[MaPaKo10] J. Majchrzak, D. Pazderski, K. Kozłowski, Sieć sensorów do pomiarów odległości dla robota mobilnego, Prace naukowe – Elektronika, z.175, problemy Robotyki T. I, red K. Tchoń i C. Zieliński, Wydawnictwo PW, Warszawa 2010, s. 209–220.