AUTOMATYKA • 2008 • Tom 12 • Zeszyt 3

Konrad Kułakowski*, Jarosław Wąs*, Marcin Szpyrka*

Architektura autonomicznego robota mobilnego z dynamicznym modelem świata**

1. Wprowadzenie

1.1. Motywacja

W ostatnim czasie można zaobserwować rosnącą popularność autonomicznych systemów mobilnych. Niektóre z nich są proste w budowie i obsłudze. Inne służą do wykonywania nieco bardziej skomplikowanych prac, jak choćby autonomicznie działający odkurzacz [32]. Czasem są wykorzystywane do wypełniania ważnych misji wojskowych [20]. Istotną rolę w działaniu autonomicznego systemu mobilnego pełni model otaczającego go świata. Z jednej strony pozwala on na przechowywanie wiedzy o środowisku pracy robota, z drugiej umożliwia przewidywanie zarówno zmian w otaczającym go świecie, jak i efektywne planowanie sekwencji ruchów dopuszczalnych. Systemy inteligentnego sterowania działające w praktyce często reprezentują posiadaną wiedzę o świecie w różny sposób. W zależności od rodzaju robota, stopnia złożoności otoczenia, w którym robot się znajduje, oraz zadań, do których został przeznaczony, może to być reprezentacja oparta o różne rodzaje przestrzeni konfiguracji takie, jak: mapy łąk (*meadow maps*), uogólnione diagramy Woronoja, siatki regularne [22], automaty komórkowe [5], różnego rodzaju mapy wektorowe GIS/GPS, a także wiedzę zapisaną w postaci deklaratywnej. Efektywne wykorzystanie wiedzy zapisanej w różnych formatach wymaga dobrej i przemyślanej architektury całego systemu. W prezentowanej pracy autorzy chcą zaproponować trzywarstwową, komponentową architekturę systemu inteligentnego sterowania integrującą wykorzystanie różnych rodzajów wiedzy. Proponowany podział na trzy warstwy ma za zadanie z jednej strony ułatwić projektowanie szczegółowych rozwiązań, z drugiej pomóc w implementacji, poprzez określenie ogólnych klas ograniczeń czasowych dla poszczególnych komponentów systemu.

* Katedra Automatyki, Akademia Górniczo-Hutnicza w Krakowie
** Praca częściowo finansowana przez MNiSW w ramach grantu: N516 024 32/2878

1.2. Struktura pracy

Budowanie systemów inteligentnego sterowania jest procesem złożonym i wieloetapowym. Wymaga zarówno poszukiwań literaturowych, pracy koncepcyjnej, jak i praktycznej weryfikacji postawionych hipotez w warunkach laboratoryjnych.

Pierwsza część prezentowanej pracy w sekcjach od 1.3 do 1.5 omawia podstawy teoretyczne rozwiązań proponowanych w dalszych częściach pracy, a także zawiera skrótowy przegląd dostępnej literatury. W części drugiej jest przedstawiona propozycja ogólnej architektury systemu inteligentnego sterowania, wraz z opisem poszczególnych komponentów systemu.

1.3. Klasyfikacja architektur systemów inteligentnego sterowania

Pojęcie inteligentnego systemu sterowania nie jest nowe. Za pionierskie prace w tej dziedzinie można uważać HILARE – robota mobilnego skonstruowanego w LAAS (Laboratoire d'Architecture et d'Analyse des Systèmes) [13] lub SHAKEY – robota skonstruowanego na Stanford University [23]. Z biegiem czasu w obrębie architektury systemów inteligentnego sterowania wykształciły się trzy następujące podejścia architektoniczne [2]:

1) system z hierarchicznym planowaniem i sterowaniem,
2) system ze sterowaniem reaktywnym,
3) system ze sterowaniem hybrydowym.

Jednym z najbardziej znanych przykładów architektur wpisujących się w pierwsze z wymienionych podejść jest tzw. *„Referencyjny model architektury inteligentnego sterowania"* zaproponowany przez Albusa [1]. Model ten jest oparty na wcześniejszych pracach [4] opisujących system RCS (*Real-Time Intelligent Control System*). Architektura RCS wyodrębnia cztery współpracujące ze sobą moduły odpowiedzialne za: generację zachowania, modelowanie otoczenia, przetwarzanie sygnałów pochodzących z otoczenia oraz ocenę sytuacji. Każdy z modułów działa na podstawie własnych węzłów obliczeniowych zorganizowanych w hierarchiczne warstwy. Każda z warstw ma dobrze zdefiniowany zakres odpowiedzialności oraz właściwe sobie pojęcie czasu. W podejściu tym bardzo istotną rolę odgrywa model otoczenia robota. Wszystkie decyzje systemu są uzgadniane na bazie tego modelu, podobnie wszystkie wykryte zdarzenia w otoczeniu mogą go zmienić.

Przeciwne podejście stanowi druga z prezentowanych architektur. Zgodnie z zasadą „świat jest sam swoim najlepszym modelem" systemy ze sterowaniem reaktywnym unikają korzystania z wewnętrznego modelu świata, a ocenę sytuacji opierają przede wszystkim na analizie postrzeganych w danym momencie zdarzeń. Systemy tej klasy są szczególnie użyteczne w przypadku działania robota w szybkozmiennym i słabo ustrukturyzowanym środowisku, gdzie konstrukcja precyzyjnego modelu świata mogłaby być trudna [2]. Przykładem tego podejścia może być zaproponowana przez Brooksa *Subsumption Architecture* [7].

Rozwiązaniem, które stara się pogodzić oba poprzednie podejścia, jest sterowanie hybrydowe. U jego podstaw leży przekonanie, że rzeczywiste systemy inteligentnego stero-

wania czasami muszą zachowywać się jak systemy o sterowaniu reaktywnym, tj. szybko reagować na zachodzące zmiany w otoczeniu, a czasem zaplanować swoje dalsze posunięcia w sposób przemyślany, oparty o głęboką analizę posiadanej wiedzy o świecie. W rzeczywistości oba podejścia: reaktywne generowanie zachowania i planowanie hierarchiczne, zwykle nawzajem się uzupełniają. Przykładowo wyznaczenie bezkolizyjnej, optymalnej ścieżki ruchu robota zwykle związane jest z przemyślanym, hierarchicznym planowaniem, natomiast szybka reakcja na niespodziewaną przeszkodę na tej ścieżce będzie typowo reaktywnym zachowaniem systemu. Przykładem tej klasy architektur systemów inteligentnego sterowania są AuRA [3] i PRS [12].

1.4. Reprezentacja wiedzy

Wiedza jest kluczowym elementem systemu, pozwalającym mu na wypełnienie jego misji oraz na prawidłową reakcję w odpowiedzi na zmiany w otaczającym go świecie. Podsystem wiedzy powinien wspierać proces akwizycji danych ze źródeł zewnętrznych, przechowywać wiedzę, dokonywać wnioskowania. Powinien także odkrywać nowe fakty z wykorzystaniem posiadanej wiedzy oraz we właściwy sposób ją udostępniać podsystemowi planowania oraz generacji zachowania. W systemach inteligentnego sterowania wiedza może być obecna w różnoraki sposób. Od prostej postaci bodziec-reakcja w systemach czysto reaktywnych do skomplikowanej, hybrydowej postaci w systemach posiadających rozbudowaną hierarchię sterowania, na przykład system *4D/RCS* [20]. Najbardziej znane sposoby przedstawiania wiedzy stosowane w tej klasie systemów, to: reprezentacja przestrzenna, reprezentacja symboliczna i będąca uogólnieniem dwóch poprzednich reprezentacja hybrydowa [11].

Reprezentacja przestrzenna (*spatial representation*) dotyczy przede wszystkim sposobu przechowywania i przetwarzania wiedzy o otoczeniu robota. Jednym ze sposobów przedstawienia takiej wiedzy jest geometryczne odzwierciedlenie rozmieszczenia elementów w przestrzeni. Najczęściej jest ono reprezentowane jako model rzeczywistej przestrzeni wokół systemu (*world space*) lub też jako *CSpace* (*configuration space*) – zbiór wszystkich możliwych pozycji (konfiguracji), które dany system może osiągnąć. Przykładem podejścia wykorzystującego *CSpace* są siatki regularne wykorzystujące pojęcia zaczerpnięte z teorii automatów komórkowych. Znane są algorytmy planowania trasy ruch robota w oparciu o siatkę automatu komórkowego, np. algorytm Trulla [21] i inne [5, 21, 31]. Inną reprezentacją jest przedstawienie przestrzeni w postaci grafu reprezentującego sieć o zadanej topologii. Węzły tego grafu reprezentują komórki odpowiadające pewnym pozycjom w rzeczywistym otoczeniu systemu. Bardziej efektywne podejścia do tworzenia takiej reprezentacji zakładają przechowywanie węzłów tylko dla takich lokacji, do których bezpiecznie może przemieścić się system.

Reprezentacja symboliczna wiedzy dostarcza sposobów przedstawiania pojęć, faktów oraz relacji pomiędzy nimi. Wiedza w tej postaci jest wygodna w przetwarzaniu i przechowywaniu. Z biegiem czasu powstało wiele różnych metod symbolicznych reprezentacji

wiedzy. Do najbardziej znanych trzeba zaliczyć różne rodzaje i postacie reguł [18], zbiory przybliżone (*rough sets*) [8, 26], ontologie formalne [14], sieci semantyczne, sieci neuronowe [30], a także języki przetwarzania symbolicznego, takie jak Prolog czy Lisp. Każda z tych reprezentacji, w zależności od kontekstu użycia może wprowadzać dodatkowe, szczegółowe rozwiązania. Przykładem reprezentacji symbolicznej wiedzy są także komórkowe sieci neuronowe oraz automaty komórkowe. Z uwagi na ich popularność w modelowaniu i symulacjach różnorakich zjawisk, znalazły one również zastosowanie w modelowaniu zachowania robotów [27].

Próbą połączenia zalet poprzednich rozwiązań jest reprezentacja hybrydowa. W tym podejściu wykorzystuje się równocześnie różne reprezentacje wiedzy. Mogą to być, opisane powyżej, reprezentacje symboliczna i przestrzenna, ale także wiedza proceduralna i parametryczna.

Wiedza proceduralna to informacja o tym, w jaki sposób realizować pewne, dobrze zdefiniowane zadania. Wiedza ta może być wyrażona w postaci diagramu stanów [20], diagramu aktywności, procedury zapisanej w danym języku programowania itp. Wiedza parametryczna jest zwykle związana z najniższym poziomem sterowania systemem. Wartości parametrów opisują działanie poszczególnych siłowników i serwomechanizmów. Procesy uczenia i adaptacji często prowadzą do zmiany wartości parametrów początkowych [16].

1.5. Istniejące przykłady architektur systemów inteligentnego sterowania

Większość projektów architektur systemów inteligentnego sterowania, tworzonych na przestrzeni ostatnich kilkunastu lat, jest oparta o paradygmat sterowania hierarchicznego [1]. Przykładem takich rozwiązań może być praca [33], w której oprócz rozwiązań architektonicznych autorzy proponują systematyczne podejście do projektowania, generacji oraz testowania tej klasy systemów. Bardziej formalnym podejściem do problemów konstrukcji systemów inteligentnego sterowania wyróżnia się praca [25]. W pracy tej autor przedstawia formalnie zdefiniowaną strukturalną hierarchię węzłów obliczeniowych, która przekłada się na strukturę sprzętową systemu. Ciekawym przykładem obiektowo-zorientowanych systemów inteligentnego sterowania są OSCAR [6] i BERRA [19]. Pierwszy z tych systemów jest implementacją architektury hierarchicznej, gdzie komunikacja pomiędzy poszczególnymi warstwami wykorzystuje architekturę CORBA (*Common Object Request Broker Architecture*) [24]. Na uwagę zasługuje tu ciekawa koncepcja hierarchicznego przetwarzania sygnałów. Drugi z systemów (BERRA) wprowadza trzy konceptualne warstwy systemu: warstwę świadomą (*Deliberative Layer*), warstwę reaktywną (*Reactive Layer*) i warstwę wykonywania zadań (*Task Execution Layer*). Przykładem hierarchicznego systemu inteligentnego sterowania, w którym istotną rolę odgrywa model świata, jest Saphira [15]. W generacji zachowania systemu bierze udział logika rozmyta, która stanowi ostatnią warstwę decydującą o takim, a nie innym jego zachowaniu. Innym klasycznym przykładem architektury systemu inteligentnego sterowania skupionym wokół modelu świata jest TCA (*Task Control Architecture*) [29].

2. Ogólna architektura systemu inteligentnego sterowania

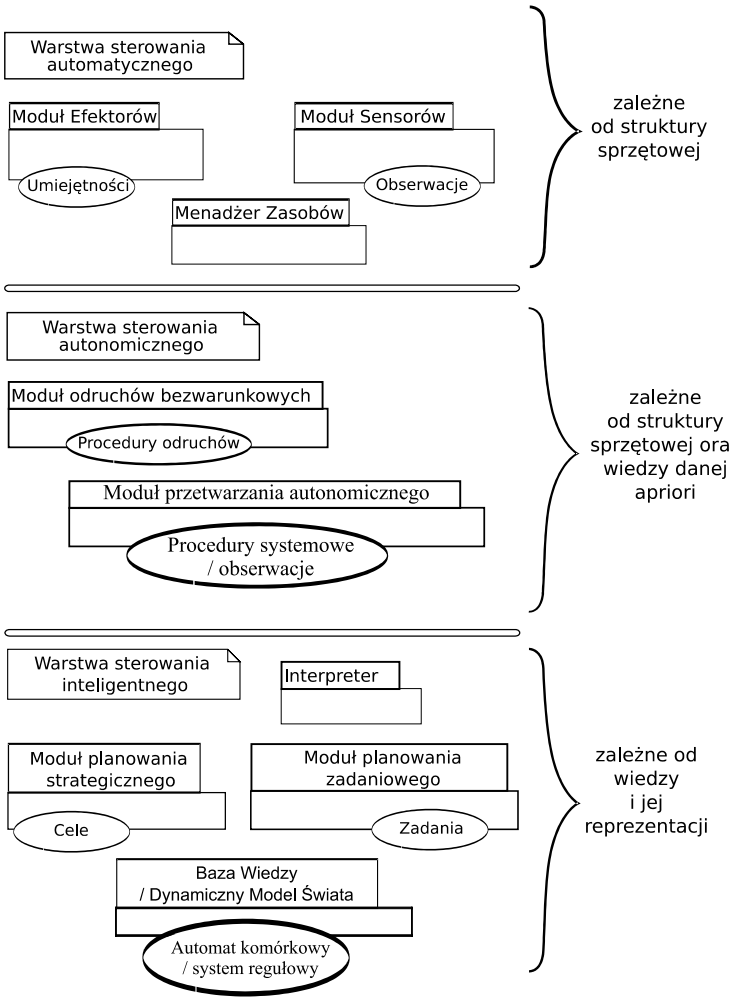
2.1. Hierarchiczna struktura systemu

Proponowana architektura systemu inteligentnego sterowania jako punkt wyjścia przyjmuje referencyjny model architektury inteligentnego sterowania zaproponowany przez Albusa [1]. W odróżnieniu jednak od klasycznego podejścia przyjętego w systemach z hierarchicznym planowaniem i sterowaniem, tu zakłada się, że system powinien w niektórych sytuacjach potrafić zachowywać się reaktywnie, szybko odpowiadając na zmiany zachodzące w jego otoczeniu. To założenie, zgodnie z klasyfikacją proponowaną przez Arkina [2], lokuje prezentowane rozwiązanie w grupie systemów hybrydowych. Proponowana architektura zakłada istnienie trzech warstw systemu:

1) warstwy automatycznej,
2) warstwy autonomicznej,
3) warstwy świadomej.

Pewnego rodzaju nowością jest rozdzielenie warstwy autonomicznej od warstwy automatycznej. W klasycznych konstrukcjach trzywarstwowych funkcjonujących w literaturze jako architektury 3T [21], za zachowania reaktywne jest odpowiedzialna tylko jedna warstwa łącząca w sobie przetwarzanie sygnałów oraz generowanie zachowania. W prezentowanym systemie następuje rozdzielenie tych funkcji, co zdaniem autorów zwiększy możliwość optymalnej implementacji całego systemu w przyszłości.

Najniższa „automatyczna" warstwa systemu składa się z trzech modułów (rys. 1). Pierwszy z modułów gromadzi efektory systemu. Tu też zdefiniowane są grupy efektorów tworzące efektory logiczne (na przykład efektorem logicznym może być ramię robota składające się z kilku przegubów, tym samym wykorzystujące kilka serwomechanizmów). Efektory logiczne wraz z prostymi algorytmami sterującymi stanowią *zbiór podstawowych umiejętności* (*simple skills set*) robota. Późniejsza realizacja wszelkich zadań robota będzie polegać na właściwej parametryzacji i szeregowaniu *umiejętności*. Drugi moduł, *moduł sensorów*, jest odpowiedzialny za logiczne grupowanie czujników oraz wstępne przetwarzanie informacji przez nie przekazywanych. Podobnie jak poprzednio, wygodnie jest wyróżnić tu pojęcie sensora logicznego przetwarzającego i grupującego informację pochodzącą z różnych źródeł. Na przykład informacja o orientacji robota może pochodzić równocześnie z żyroskopu, kompasu magnetycznego oraz z analizy przebytej trasy. Efektem działania sensorów logicznych są *obserwacje*. Na podstawie dostarczonych pozostałym warstwom *obserwacji*, system podejmuje decyzje sterujące zachowaniem systemu. Trzeci moduł, *menadżer zasobów*, jest odpowiedzialny za kontrolę dostępu do wymienionych wcześniej dwóch modułów. W szczególności do zadań tego modułu należy egzekwowanie priorytetów przychodzących poleceń i niedopuszczenie, by równocześnie były wykonywane zadania sprzeczne.



Rys. 1. Warstwy systemu inteligentnego sterowania

Relacje pomiędzy klasami w obiektowej implementacji warstwy automatycznej powinny oddawać fizyczne zależności pomiędzy efektorami i sensorami w realnie istniejącej konstrukcji sprzętowej. Zakres odpowiedzialności poszczególnych klas realizujących oprogramowanie tej warstwy powinien być precyzyjnie zdefiniowany za pomocą parametryzowalnych metod. Wiedza o świecie obecna w obrębie tej warstwy ma charakter parametryczny. Działające oprogramowanie w obrębie tej warstwy ma charakter *systemu czasu rzeczywistego* o ostrych lub mocnych wymaganiach czasowych (*hard real time, firm real time*) i może być, w części lub w całości implementowane jako *system wbudowany*.

Druga warstwa prezentowanego modelu, odpowiedzialna za sterowanie autonomiczne, grupuje występującą w systemie proceduralną wiedzę aprioryczną. Wiedza ta dotyczy

dobrze zdefiniowanych zachowań dających się zapisać w postaci prostych procedur. W obrębie tej warstwy zostały wydzielone dwa moduły: *moduł odruchów bezwarunkowych, moduł przetwarzania autonomicznego*. Zadaniem pierwszego z nich, *modułu odruchów bezwarunkowych*, jest szybkie generowanie prostego zachowania w sytuacjach wymagających natychmiastowej reakcji, w których przetwarzanie wiedzy zapisanej w postaci deklaratywnej mogłoby trwać za długo. Przykładem sytuacji, w której procedury tego modułu mogą być użyte, jest pojawienie się w systemie *obserwacji* sugerującej bliską obecność przeszkody na trasie poruszania się robota. W takim przypadku system powinien zareagować tak szybko jak to tylko możliwe, na przykład zatrzymać się. Moduł odruchów bezwarunkowych powinien być zaimplementowany jako komponent czasu rzeczywistego z silnymi (*firm*) ograniczeniami czasowymi.

Kolejny moduł tej warstwy jest odpowiedzialny za wspomaganie sterowania różnymi komponentami systemu. Zapewnia on niezbędne procedury testujące system, kontrolujące stan zasobów, oraz koordynujące współdziałanie poszczególnych modułów. Może on również generować *obserwacje* informujące inne moduły o stanie systemu. Typowym przykładem takiej notyfikacji może być informacja o stanie baterii robota. W przypadku niskiego poziomu naładowania baterii, po otrzymaniu odpowiedniej informacji od warstwy autonomicznej, system powinien wygenerować zachowanie zmierzające do powtórnego jej naładowania. Ograniczenia czasowe nakładane na rozważany moduł zależą od rodzaju przetwarzania i mogą mieć charakter silnych lub miękkich ograniczeń czasowych.

Trzecia warstwa proponowanego systemu zawiera moduły odpowiedzialne za przetwarzanie i przechowywanie wiedzy zapisanej w postaci deklaratywnej. Moduł planowania strategicznego jest odpowiedzialny za realizację *misji* robota, pojętej jako hierarchiczna struktura *celów* przeznaczonych do realizacji. Każdy z obiektów reprezentujących wybrany *cel* może składać się z wielu podobiektów reprezentujących *cele częściowe*. Najwyższy w hierarchii planowania *cel* będzie nazywany *misją* robota. *Cele częściowe* na tyle szczegółowe, by nie zawierać innych *celów częściowych*, będą nazywane zadaniami (*tasks*). Rolą *modułu planowania strategicznego* jest takie dobranie celów do realizacji, by przy danym stanie wiedzy systemu oraz przy uwzględnieniu napływających do systemu obserwacje z modułu sensorycznego zapewnić optymalną realizację *misji* robota.

Kolejnym elementem *warstwy sterowania inteligentnego* jest *moduł planowania zadaniowego*. Zadaniem tego modułu jest przechowywanie i przetwarzanie wiedzy o *zadaniach*. *Zadanie* może w sposób hierarchiczny zawierać inne podzadania. Zadanie szczegółowe, niezawierające innych podzadań, powinno prowadzić do ustalenia ciągu czynności (*umiejętności*) niezbędnych do realizacji zadania. Modułem odpowiedzialnym za przekształcenie zadań w sekwencje czynności zrozumiałych przez warstwę automatyczną jest *interpreter*.

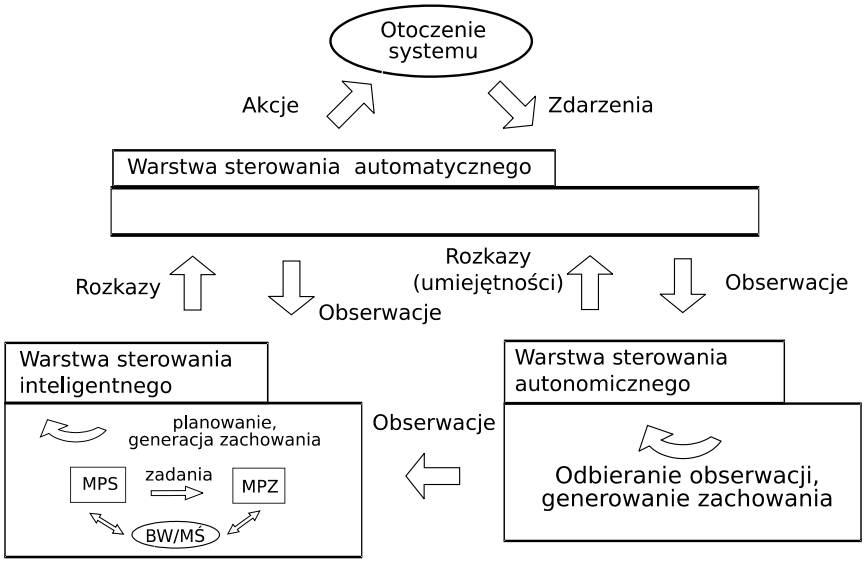
Istotną częścią warstwy inteligentnego sterowania jest moduł zawierający bazę wiedzy oraz model świata. Moduły planujące, podejmując decyzję oraz generując zachowanie, przetwarzają informację dostępną w bazie wiedzy oraz modelu świata. Baza wiedzy może przechowywać wiedzę w różnej postaci. Model świata w proponowanym podejściu jest dynamiczny. Może zmieniać się samoistnie wraz z upływem czasu, może też być zasilany

informacjami pochodzącymi z obserwacji. Moduły planujące mogą również dokonywać predykcji zachowania otoczenia na podstawie tak skonstruowanego modelu świata.

Komponenty warstwy sterowania inteligentnego mogą mieć charakter podsystemów czasu rzeczywistego, przy czym zwykle ograniczenia czasowe nie są ostre. Dodatkowym problemem jest możliwy niedeterminizm obliczeń prowadzonych przez moduły tej warstwy, wynikający z dynamicznie zmieniającego się otoczenia systemu.

2.2. Przepływ sterowania

Podstawowym mechanizmem wspierającym przepływ sterowania pomiędzy poszczególnymi modułami jest przesyłanie wiadomości. Informacja zawarta w wiadomości zależy od modułu wysyłającego oraz odbierającego. Większość komunikacji w systemie będzie odbywać się pomiędzy poszczególnymi warstwami. Wyjątek stanowią moduły warstwy inteligentnego sterowania, które komunikują się zarówno z modułami innych warstw, jak i z sobą nawzajem (rys. 2).



Rys. 2. Komunikacja – przepływ sterowania

Warstwa sterowania automatycznego, poprzez moduł sensorów postrzega zdarzenia zachodzące w otoczeniu systemu. Na tej podstawie przekazuje innym modułom poczynione obserwacje. Moduł efektorów przyjmuje rozkazy zawierające ciągi sparametryzowanych czynności odpowiadających dostępnym umiejętnościom robota. Warstwa sterowania autonomicznego współpracuje z warstwą sterowania automatycznego, przekazując jej rozkazy do wykonania oraz odbiera od niej wiadomości z obserwacjami. Sama również generuje obserwacje informujące o stanie systemu i wysyła je do warstwy sterowania inteligentnego.

W obrębie warstwy sterowania inteligentnego moduł planowania strategicznego (MPS) przekazuje modułowi planowania zadaniowego (MPZ) zadania do wykonania. Obydwa te moduły komunikują się z bazą wiedzy i modelem świata, odczytując i zapisując odpowiednie informację. Interakcja z warstwą automatyczną podobnie jak w przypadku warstwy poprzedniej polega na wysyłaniu rozkazów do wykonania oraz odbieraniu generowanych obserwacji.

2.3. Model świata

Model świata w prezentowanej architekturze może składać się z różnych reprezentacji wiedzy. W proponowanej architekturze informacja o otoczeniu robota jest zapisywana w postaci regularnej siatki automatu komórkowego klasy ECAL [9, 10]. W obrębie tej reprezentacji jest także zawarta wiedza o dynamice i zachowaniu innych obiektów w otoczeniu systemu. Zastosowanie automatu komórkowego pozwoli na przechowywanie wiedzy zarówno o aktualnym stanie otoczenia systemu, jaki i wiedzy o możliwych jego zmianach w przyszłości. Taka postać wiedzy pozwoli również na symulacyjne przewidywanie zmian w otoczeniu.

Dodatkowa informacja o własnościach obiektów znajdujących się w otoczeniu systemu i ich znaczeniu dla robota będzie zapisywana w postaci wiedzy deklaratywnej. (Na przykład z wykorzystaniem systemu regułowego).

Planowanie trasy ruchu robota odbywać się będzie przy użyciu wszystkich dostępnych reprezentacji wiedzy składających się na *model świata* i *bazę wiedzy* z wykorzystaniem właściwych im algorytmów.

3. Podsumowanie

Budowanie systemów inteligentnego sterowania pomimo swojej długiej i bogatej historii wciąż cieszy się niesłabnącą popularnością. Zmieniające się wymagania ludzi wobec maszyn oraz nowe możliwości związane z postępem technicznym stwarzają nieustanne zapotrzebowanie na nowe, lepsze konstrukcje. Ważnym elementem procesu budowy takiego systemu jest stworzenie projektu architektury. Projekt ten stanowi wyjście do dalszych prac, kończących się jego programowo-sprzętową realizacją.

W prezentowanej pracy została przedstawiona nowa, oryginalna architektura systemu inteligentnego sterowania autonomicznym robotem mobilnym. W szczególności, pomimo istnienia podobnych modeli [2, 21], wydaje się, że w opisywanych w literaturze konstrukcjach nie używano dynamicznego modelu świata w sposób, który jest tu proponowany.

W chwili obecnej trwają prace nad budową docelowego systemu, zgodnego z zaproponowanym modelem architektonicznym, na platformie Mindstorms NXT [17]. Na dalszym etapie prac przewidywane jest wykorzystanie jako platformy sprzętowej dla konstruowanego systemu robota kroczącego Hexor II [28].

Literatura

[1] Albus J.S., *RCS: A Reference Model Architecture for Intelligent Control*. IEEE Computer, nr 5, vol. 25, 1992, 56–59.

[2] Arkin R.C., *Intelligent Control of Robot Mobility*. Wiley Press, 2007.

[3] Arkin R.C., Balch T., *Cooperative Multiagent Robotic Systems*. 1998.

[4] Barbera A.J., Fitzgerald M.L., Albus J.S., Haynes L.S., *RCS: The NBS Real-Time Control System*. 1984.

[5] Behring C., Bracho M., Castro M., Moreno J.A., *An Algorithm for Robot Path Planning with Cellular Automata*. Theoretical and Practical Issues on Cellular Automata, Proceedings of the Fourth International Conference on Cellular Automata for Research and Industry, Karlsruhe, 4–6 October 2000, Springer Verlag.

[6] Blum S.A., *From a CORBA-Based Software Framework to a Component-Based System Architecture for Controlling a Mobile Robot*. Berlin, Heidelberg, Springer 2003, 333–344.

[7] Brooks R.A., *A Robust Layered Control System for a Mobile Robot*. IEEE J. Robot. and Auto., 1986, 14–23.

[8] Doherty P., Lukaszewicz W., Skowron A., Szalas A., *Knowledge Representation Techniques – A Rough Set Approach*. Springer Verlag, 2006.

[9] Dudek-Dyduch, E., Wąs, J., *Formalizacja automatów komórkowych w zagadnieniach dynamiki pieszych*. Automatyka (półrocznik AGH), t. 9, z. 3, 2005.

[10] Dudek-Dyduch, E., Wąs, J., *Knowledge representation of Pedestrian Dynamics in Crowd. Formalism of Cellular Automata*. Lecture Notes in Artificial Inteligence, vol. 4029, 2006.

[11] Ge S.S., Lewis F.L., *Autonomous Mobile Robots*. CRC Press, 2006.

[12] Georgeff M.P., Lansky A.L., *Reactive Reasoning and Planning*. Proceedings of the 6th National Conference on Artificial Intelligence, 1987.

[13] Giralt G., Sobek R., Chatila R., *A Multi-Level Planning and Navigation System for a Mobile Robot; A First Approach to HILARE*. International Joint Conference on Artificial Intelligence, 1979.

[14] Guarino N.: *Formal ontology, conceptual analysis and knowledge representation*. International Journal of Human-Computer Studies, 1995, 625–640.

[15] Konolige K., Myers K., Ruspini E., Saffiotti A., *The Saphira Architecture: A Design for Autonomy*. 1997.

[16] Lee J.B., Likhachev M., Arkin R.C., *Selection of Behavioral Parameters: Integration of Discontinuous Switching via Case-Based Reasoning with Continuous Adaptation via Learning Momentum*. IEEE, ICRA, 2002.

[17] Lego: *Lego Mindstorms*. http://mindstorms.lego.com/, 2008.

[18] Ligęza A.: *Logical Foundations for Rule-Based Systems, 2nd Ed*. Springer Verlag, 2006.

[19] Lindström M., Orebäck A., Christensen H.I., *BERRA: A Research Architecture for Service Robots*. IEEE, ICRA, 2000.

[20] Messina E., Murphy K., Lacaze A., Shneier M., Scott H., Shackleford P.W., Michaloski J., Wavering A., Nicholas T.K., Legowik S., Bostelman R., Jacoff A., Falco J., Gilsinn J., Barbera C.A., Fitzgerald M.L., Giorno D.M.D., *4D/RCS: A Reference Model Architecture For Unmanned Vehicle Systems Version 2.0*. 2002.

[21] Marchese F.M., *A directional diffusion algorithm on cellular automata for robot path-planning*. Future Generation Computer Systems, vol. 18, no 7, August, 2002, 983–994, ISSN:0167-739X.

[22] Murphy R.R., *Introduction to AI Robotics*. MIT Press, 2000.

[23] Nilsson N., *Shakey the Robot*. AI Center, SRI International, 1984.

[24] OMG (Object Management Group): *The Comon Object Request Broker: Architecture and Specification*, 1996.

[25] Özgüner Ü., Acar L., *Design of Structure-Based Hierarchies for Distributed Intelligent Control*. Kluwer Academic Publishers, P.J. Antsaklis and K.M. Passino (Eds.), 1993.
[26] Pawlak Z., *Rough Sets. Theoretical Aspects of Reasoning about Data*. Kluwer Academic Publishers, Dordrecht, The Netherlands, 1991.
[27] Siemiątkowska B., *Coordinating the motion of mobile robots using Cellular Neural Network*. Journal of Automation, Mobile Robotics and Intelligent Systems, 2008.
[28] Stenzel: *Projekt Hexor*. http://www.stenzel.com.pl, 2008.
[29] Simmons R., Goodwin R., Haigh K., Koenig S., O'Sullivan J., *A Layered Architecture for Office Delivery Robots*. Autonomous Agents, 1997.
[30] Tadeusiewicz R., *Sieci neuronowe*. Wydawnictwo RM, 1993.
[31] Tzionas P., Tsalides Ph., Thanailakis A., *Cellular Automata-Based Collision-Free Robot Path Planning*. Proceedings of the ASP-DAC '95/CHDL '95/VLSI '95, Chiba, Japan, ISBN: 4-930813-67-0.
[32] Ulrich I., Mondada F., Nicoud J.. *Autonomous vacuum cleaner*. Robotics and Autonomous Systems, t. 19, n. 3–4, 1997.
[33] Zeigler B.P., Chi S., *Model-Based Architecture Concepts for Autonomous Systems Design and Simulation*. Kluwer Academic Publishers, P.J. Antsaklis and K.M. Passino (Eds.), 1993.