Specyfikacja struktur serwomechanizmów wizyjnych

Tomasz Kornuta, Cezary Zieliński
Instytut Automatyki i Informatyki Stosowanej, Politechnika Warszawska

Streszczenie: W artykule przedstawiono formalną metodę opisu złożonych systemów robotycznych, za pomocą której wyspecyfikowano układy realizujące trzy diametralnie różne zachowania robota: ruch pozycyjny w przestrzeni kartezjańskiej, sterowanie oparte o informację wizyjną pochodzącą z ruchomej kamery zintegrowanej z jego chwytakiem oraz sterowanie wykorzystujące informację odebraną z nieruchomej kamery. Przedstawione wyniki eksperymentów potwierdzają poprawność stworzonych układów.

Słowa kluczowe: systemy robotyczne, systemy agentowe, serwomechanizmy wizyjne, specyfikacja

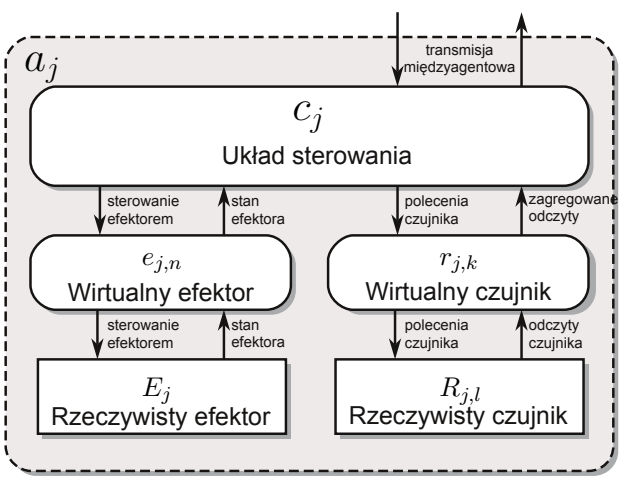
Projektowanie złożonych systemów, jakimi są wielorobotowe systemy sterowania, wymaga odpowiedniej metody opisu, umożliwiającej jej proste przełożenie na właściwą implementację. W poniższym artykule wykorzystano metodę opisu systemów robotycznych opartą na wcześniejszych pracach, prezentowanych np. w [10]. Bazuje ona na podejściu agentowym [2, 7] oraz funkcjach przejścia. W tym artykule zaprezentowano zarys ogólnej metody opisu systemów robotycznych oraz przedstawiono przykład jej wykorzystania do określenia struktury i działania układów sterowania realizujących jedną z najistotniejszych funkcji robotów usługowych, jak i terenowych – koordynację ręka-oko, osiąganą dzięki serwomechanizmom wizyjnym.

1. Serwomechanizmy wizyjne

Serwomechanizmem wizyjnym [3, 4] nazywany jest układ, który na podstawie znajomości aktualnego położenia efektora oraz cel ruchu określonego na podstawie obrazu uzyskanego z kamery wyznacza uchyb, który dalej wykorzystuje do wygenerowania sterowania zmniejszającego ten uchyb. Istnieje szereg kryteriów klasyfikacji serwomechanizmów wizyjnych [5, 8], poniżej omówiono trzy najważniejsze z nich. Pierwszym jest miejsce zamocowania kamery – wyróżnić można serwomechanizmy wykorzystujące kamerę nieruchomą, umieszczoną poza robotem (SAC – *Stand Alone Camera*), oraz kamerę ruchomą, zamocowaną na robocie, np. zintegrowaną z chwytakiem manipulatora (EIH – *Eye In Hand*). Drugie kryterium dotyczy widoczności robota (w szczególności końcówki manipulatora) w obrazie uzyskanym z kamery. Mamy tu do czynienia z systemami obserwującymi końcówkę (ECL – *Endpoint Closed Loop*) oraz jej nieobserwującymi (EOL – *Endpoint Open Loop*). Trzecie kryterium dotyczy przestrzeni, w jakiej obliczany jest uchyb. Uchyb może być wyznaczany w przestrzeni operacyjnej, gdzie uchyb określa się na podstawie pozycji końcówki i celu widzianego przez kamerę w przestrzeni kartezjańskiej (PB – *Position Based*), lub bezpośrednio w przestrzeni cech obrazu (IB – *Image Based*). Każde z wymienionych kryteriów jest niezależne od pozostałych, więc istnieje wiele rodzajów serwomechanizmów. W tym artykule uwagę skupiono na pierwszym kryterium, związanym z lokalizacją kamery, przy czym założono, iż kamera nie będzie obserwowała końcówki manipulatora, a uchyb będzie obliczany w przestrzeni operacyjnej – rozpatrzono więc serwomechanizmy typu PB-SAC-EOL i PB-EIH-EOL.

2. Notacja dotycząca specyfikacji systemów robotycznych

Prezentowana poniżej notacja może być wykorzystania do opisu systemów składających się z wielu agentów, natomiast w tej pracy uwaga została skupiona na pojedynczym agencie. W najogólniejszej postaci agent a_j (rys. 1) posiada efektory E_j, którymi oddziałuje na otoczenie, receptory, R_j, dzięki którym zbiera informacje o stanie środowiska (eksteroreceptory) oraz swego efektora (proprioreceptory), a ponadto dysponuje układem sterowania. Zarówno odczyty z receptorów, jak i polecenia wydawane efektorom podlegają transformacjom. Transformacje te przedstawiają układowi sterowania zarówno stan efektorów jak i odczyty z receptorów w dogodnej do sterowania postaci, dlatego też wyróżniono wirtualny efektor e_j oraz wirtualne czujniki r_j, odpowiedzialne za te transformacje. Układ sterowania komunikuje się z tymi wirtualnymi tworami poprzez bufory komunikacyjne, czasami zwane widokami lub obrazami



Rys. 1. Ogólna struktura agenta a_j
Fig. 1. General structure of an agent a_j

efektorów oraz receptorów, bo przez ich pryzmat układ sterowania, a wiec i jego projektant, postrzegają rzeczywiste efektory i receptory.

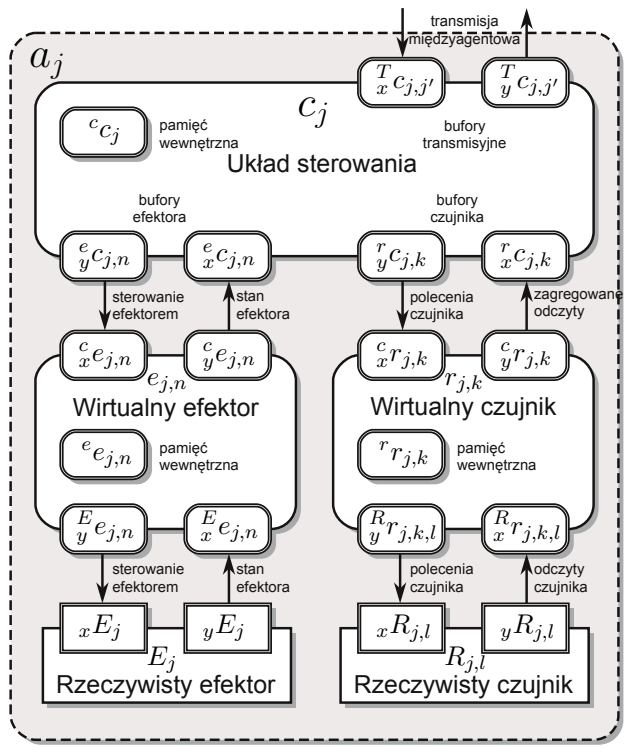
Już z tego wstępnego opisu widać, że liczba różnych elementów agenta jest znaczna i co więcej każdy z wymienionych tworów ma swoje części składowe (wyjawione w dalszej części artykułu). Aby opis działania tak złożonego systemu uczynić czytelnym, należy wprowadzić spójny system oznaczeń. Dla uproszczenia nie wprowadzono oddzielnych symboli dla określenia danego tworu i jego stanu – kontekst odróżnia je dostatecznie. Przyjęto następujący system oznaczeń. Jednoliterowy symbol główny, określający element, do którego się odwołujemy (np. E, R, e, r, c), umieszczany jest w centrum. Aby odwołać się do części składowych tego elementu lub określić chwilę, w której opisujemy jego stan, umieszczamy dodatkowe indeksy wokół symbolu centralnego. Lewy górny indeks jest zarezerwowany, aby określić bufor, do którego się odwołujemy w tym elemencie, lub, w przypadku funkcji, jej rodzaj. Górny prawy indeks określa czas, którego dotyczy rozpatrywany stan. Dolny lewy indeks określa, czy mamy do czynienia z buforem wejściowym (x) czy wyjściowym (y). Indeksy oddzielone przecinkami i umieszczone u dołu po prawej stronie symbolu centralnego określają kolejno: numer agenta, numer elementu oraz jego składowe, lub, w przypadku funkcji, jej numer. Przykładowo ${}_yR^{\iota}_{j,l}$ oznacza odczyt l-tego receptora j-tego agenta w chwili ι – odczyty pochodzą z jego wyjścia, stąd indeks y. Wyjawiona powyżej konwencja powinna ułatwić odczytywanie poszczególnych symboli w miarę ich wprowadzania.

W artykule przyjęto również jednolitą konwencję opisu transformacji układów współrzędnych ${}^{X}_{Y}\mathcal{T}_{W}$. $X \in \{0, E, C\}$ opisuje układ współrzędnych: 0 – związany z podstawą robota (globalny), E – związany z końcówką roboczą, C – związany z kamerą (np. ${}^{0}_{O}\mathcal{T}$ oznacza układ obiektu względem układu globalnego). Symbol $W \in \{c, d, p\}$ jest związany z wartością zmiennej: c to wartość aktualna, p to wartość poprzednia, natomiast d jest wartością pożądaną.

3. Ogólna struktura agenta

Dzięki swym eksteroreceptorom, czyli rzeczywistym czujnikom R_j, agent a_j pobiera z otoczenia informacje o jego stanie. Takich czujników może być wiele, stąd są one indeksowane: $R_{j,l}$. Czujniki wirtualne r_j dokonują agregacji odczytów otrzymanych z przypisanych im eksteroreceptorów. Ponieważ czujników wirtualnych również może być wiele, zostały one poindeksowane: $r_{j,k}$. Agregacja informacji może polegać na kompozycji odczytów uzyskanych z kilku czujników rzeczywistych albo na ekstrakcji użytecznej informacji z pojedynczego złożonego czujnika. Tak więc odczyty z tych samych czujników rzeczywistych mogą być agregowane na różne sposoby, co może być powodem stworzenia różnych czujników wirtualnych. Odczyty czujników wirtualnych przekazywane są do układu sterowania agenta c_j. Układ ten generuje rozkazy sterujące wirtualnym efektorami e_j. Wprawdzie zazwyczaj pojedynczy agent steruje pojedynczym efektorem, ale może on być przedstawiany układowi sterowania na różne sposoby – stąd wielość efektorów wirtualnych $e_{j,n}$. Każdy efektor wirtualny oddziaływuje na układ elektro-mechaniczny efektora E_j.

Przykładowo, zakładając, że efektorem jest manipulator, układ sterowania c_j może wytwarzać rozkazy, których argumenty wyrażane są jako pozycje w przestrzeni operacyjnej (położenie kartezjańskie połączone z którąś reprezentacją orientacji). Mogą wtedy istnieć dwa efektory wirtualne, jeden korzystający z odwrotnego zagadnienia kinematyki, aby uzyskać położenia w przestrzeni konfiguracyjnej (położenia złącz), a drugi z odwrotności jakobianu, aby uzyskać prędkości uogólnione lub przyrosty położenia w złączach. Układ sterowania musi mieć możliwość określenia aktualnego stanu efektora E_j. Dlatego istnieje połączenie w przeciwnym kierunku, dzięki któremu odczyty proprioreceptorów (np. enkoderów) przetwarzane są przez moduł tworzący efektor wirtualny e_j do postaci akceptowanej przez układ sterowania c_j. Ponadto układ sterownia musi mieć wpływ zarówno na sposób agregacji danych przez czujniki wirtualne r_j, jak i ewentualną rekonfigurację receptorów R_j. Stąd drugie połączenie układu sterowania z tymi elementami. Wreszcie agent ma możliwość nawiązywania dwustronnej łączności z innymi agentami $a_{j'}$, $j \neq j'$.



Rys. 2. Ogólna struktura agenta a_j z buforami komunikacyjnymi
Fig. 2. General structure of an agent a_j with communication buffers

Każdy z wymienionych elementów dysponuje swą pamięcią wewnętrzną (choć oczywiście nie musi z niej korzystać). W przypadku czujników wirtualnych jest to pamięć sensoryczna ${}^{r}r_j$, natomiast w przypadku efektora wirtualnego jest to pamięć sterowania ${}^{e}e_j$. Ta pierwsza umożliwia przykładowo uśrednianie odczytów czujnika, natomiast ta druga wybór właściwego rozwiązania odwrotnego zagadnienia kinematyki w przypadku wielości tych rozwiązań. Obie te pamięci pełnią jedynie rolę pomocniczą. Natomiast pamięć znajdująca się w układzie sterowania c_j, a więc ${}^{c}c_j$, ma znaczenie fundamentalne dla funkcjonowania agenta. Służy ona do modyfikowania zachowań robota w zależności od zgromadzonych przez niego doświadczeń.

Opisane powyżej elementy struktury agenta komunikują się ze sobą za pośrednictwem buforów, które zobrazowano na rys. 2. Każdy z elementów ma bufory wejściowe (oznaczone indeksem x u dołu z lewej strony symbolu głównego określającego element, którego dotyczą) oraz bufory wyjściowe (indeks y), a ponadto pamięć wewnętrzną (brak lewego dolnego indeksu). Lewy górny indeks określa albo źródło informacji (dla buforów wejściowych) albo jej adresata (dla buforów wyjściowych). Dolne prawe indeksy określają numer agenta oraz numer tworu, do którego się symbol odnosi. Przykładowo, ${}^{R}_{x}r_{j,k,l}$ jest buforem wejściowym $k$tego czujnika wirtualnego $j$tego agenta, otrzymującego informacje od $l$tego czujnika rzeczywistego tego agenta. Natomiast ${}^{T}_{y}c_{j,j'}$ stanowi bufor transmisyjny układu sterowania $j$tego agenta, czyli a_j, przekazujący informacje agentowi $a_{j'}$. Ogólną zasadą jest to, że duże litery stosowane jako górny lewy indeks, pojawiają się, gdy komunikacja dotyczy sprzętu lub innych agentów, natomiast małe litery związane są z komunikacją wewnątrz agenta.

Na rys. 2 pokazano jedynie strukturę agenta, natomiast nie wyjaśniono samego sposobu jego działania. Każdy z elementów tej struktury posiada co najmniej jedną funkcję, która przekształca dane znajdujące się w buforach wejściowych oraz pamięci wewnętrznej na dane umieszczane w swoich buforach wyjściowych oraz ponownie w swej pamięci. Dla układu sterowania c_j funkcje tego typu nazywane są funkcjami przejścia ${}^{c}f_{,}$ dla czujników wirtualnych r_j są to funkcje czujników wirtualnych ${}^{r}f_j$, natomiast dla efektora wirtualnego e_j funkcje efektora wirtualnego ${}^{e}f_j$. Każda z tych funkcji może być zdekomponowana. Podstawową formą dekompozycji jest rozdzielenie tych funkcji według adresatów generowanych wartości. Funkcja przejścia ${}^{c}f$ tworzy wartości dla: pamięci własnej ${}^{c}c$ układu sterowania za pomocą ${}^{c,c}f_j$, bufora wyjściowego efektora wirtualnego ${}^{e}_{y}c_{,}$ korzystając z ${}^{c,e}f_j$, bufora wyjściowego czujników wirtualnych ${}^{r}_{y}c_{,}$ obliczając ${}^{c,r}f_j$, oraz bufora wyjściowego transmitera ${}^{T}_{y}c_{,}$ wykorzystując ${}^{c,T}f_j$. Wartości wymienionych funkcji obliczane są na podstawie danych dostępnych w pamięci własnej ${}^{c}c$ oraz buforach wejściowych ${}^{e}_{x}c_{,}$ ${}^{r}_{x}c_{i}$ ${}^{T}_{x}c_{.}$ Obliczenia zajmują czas, więc argumenty pochodzą z dyskretnej chwili i, natomiast ich wyniki dostępne będą w chwili $i+1$:

$$\begin{cases} {}^{c}c_j^{i+1} &= {}^{c,c}f_j({}^{c}c_j^{i},\ {}^{e}_{x}c_j^{i},\ {}^{r}_{x}c_j^{i},\ {}^{T}_{x}c_j^{i}) \\ {}^{e}_{y}c_j^{i+1} &= {}^{c,e}f_j({}^{c}c_j^{i},\ {}^{e}_{x}c_j^{i},\ {}^{r}_{x}c_j^{i},\ {}^{T}_{x}c_j^{i}) \\ {}^{r}_{y}c_j^{i+1} &= {}^{c,r}f_j({}^{c}c_j^{i},\ {}^{e}_{x}c_j^{i},\ {}^{r}_{x}c_j^{i},\ {}^{T}_{x}c_j^{i}) \\ {}^{T}_{y}c_j^{i+1} &= {}^{c,T}f_j({}^{c}c_j^{i},\ {}^{e}_{x}c_j^{i},\ {}^{r}_{x}c_j^{i},\ {}^{T}_{x}c_j^{i}) \end{cases} \tag{1}$$

Działanie efektora wirtualnego można zapisać w podobny sposób stosując funkcje: pamięci efektora wirtualnego ${}^{e,e}f_j$, sterującą efektorem rzeczywistym ${}^{e,E}f_j$ i proprioreceptywną ${}^{e,c}f_j$:

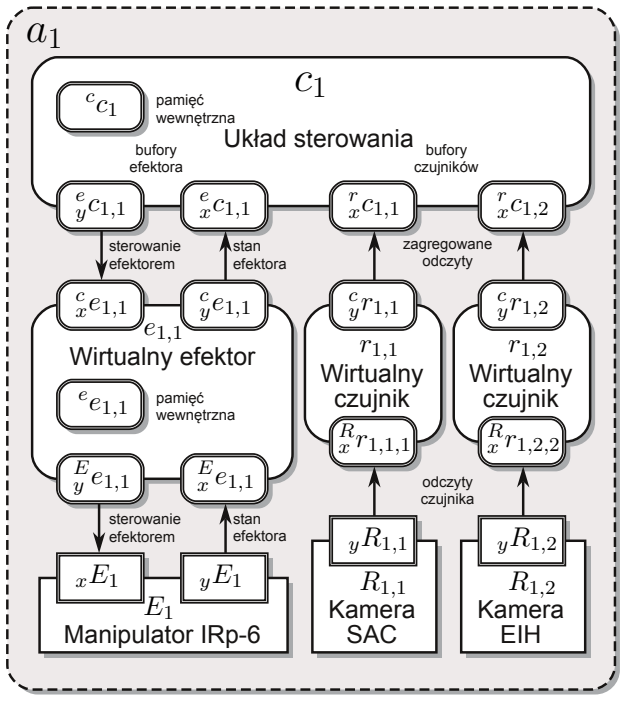
$$\begin{cases} {}^{e}e_j^{\iota+1} &= {}^{e,e}f_j({}^{e}e_j^{\iota},\ {}^{E}_{x}e_j^{\iota},\ {}^{c}_{x}e_j^{\iota}) \\ {}^{E}_{y}e_j^{\iota+1} &= {}^{e,E}f_j({}^{e}e_j^{\iota},\ {}^{E}_{x}e_j^{\iota},\ {}^{c}_{x}e_j^{\iota}) \\ {}^{c}_{y}e_j^{\iota+1} &= {}^{e,c}f_j({}^{e}e_j^{\iota},\ {}^{E}_{x}e_j^{\iota},\ {}^{c}_{x}e_j^{\iota}) \end{cases} \tag{2}$$

Sposób działania wirtualnych czujników nie odbiega od powyższego schematu, ale należy tu skorzystać z funkcji agregujących. Przyjęto tu ten sam symbol określający upływ czasu, co dla efektora, ale oczywiście w ogólności te dwie wielkości są różne. Co więcej, każdy z czujników wirtualnych może funkcjonować z innym cyklem, a więc takich zegarów w systemie może być wiele i nie dość, że zazwyczaj działają one z różnymi częstotliwościami, to jeszcze nie są one wstępnie synchronizowane. To istotne zagadnienie nie jest przedmiotem tego artykułu – jego rozwiązaniu poświęcono pracę [9]. Działanie czujnika wirtualnego można zapisać, stosując funkcje: pamięci sensorycznej ${}^{r,r}f_j$, konfiguracji receptora ${}^{r,R}f_j$ oraz agregacji odczytu czujnika wirtualnego ${}^{r,c}f_j$.

$$\begin{cases} {}^{r}r_j^{\iota+1} &= {}^{r,r}f_j({}^{r}r_j^{\iota},\ {}^{R}_{x}r_j^{\iota},\ {}^{c}_{x}r_j^{\iota}) \\ {}^{R}_{y}r_j^{\iota+1} &= {}^{r,R}f_j({}^{r}r_j^{\iota},\ {}^{R}_{x}r_j^{\iota},\ {}^{c}_{x}r_j^{\iota}) \\ {}^{c}_{y}r_j^{\iota+1} &= {}^{r,c}f_j({}^{r}r_j^{\iota},\ {}^{R}_{x}r_j^{\iota},\ {}^{c}_{x}r_j^{\iota}) \end{cases} \tag{3}$$

Funkcje określone przez (1), (2) oraz (3) tworzą specyfikację działania systemu oraz określają ewolucję jego stanu. Ponieważ w systemie będzie z reguły istniało wiele różnych funkcji, stąd prawy dolny indeks dla tych funkcji przybierze postać złożoną. Przykładowo, ${}^{c}f_{j,m}$ stanowi m-tą funkcję przejścia układu sterowania j-tego agenta.

Ponieważ funkcje te mogą być bardzo złożone, ich zapis w postaci analitycznej może być trudny w odbiorze. Dlatego stworzono pomocniczy zapis rysunkowy, ułatwiający opisanie ich działania.



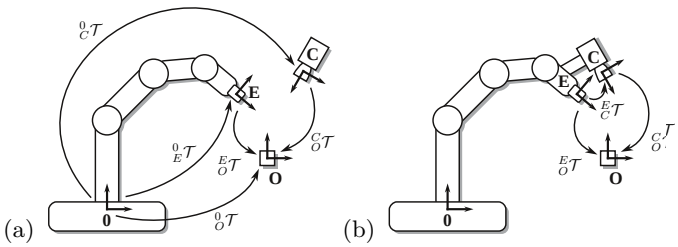
Rys. 3. Struktura agenta a_1
Fig. 3. Structure of the agent a_1

Specyfikacja, na którą składają się struktury danych (pamięć wewnętrzna i bufory komunikacyjne zawarte w c_j, e_j i r_j) oraz wspomniane powyżej funkcje, tworzy opis działania systemu, który jest podstawą dla programistów do napisania oprogramowania sterującego. W dalszej części artykułu przedstawiono specyfikację agenta mogącego wykorzystać zarówno ruch pozycyjny, jak i dwa diametralnie różne (biorąc pod uwagę wzajemne położenie kamery oraz efektora) serwomechanizmy wizyjne do realizacji zadań.

4. Struktura agenta posiadającego wzrok

W rozpatrywanym przypadku (rys. 3) robot posiadający wzrok może sterować jedynie pojedynczym manipulatorem (efektor E_1), tak więc do realizacji koordynacji ruchowo-wzrokowej wystarczy system złożony z jednego agenta a_1. Wynika z tego, iż środki komunikacji z innymi agentami są zbędne. Ponieważ rozważany system będzie wykorzystywał dwa różne czujniki rzeczywiste (rys. 4): kamerę zamocowaną na stałe na scenie (eksteroreceptor $R_{1,1}$) oraz drugą, zintegrowaną z końcówką efektora (eksteroreceptor $R_{1,2}$), dlatego też niezbędne są dwa czujniki wirtualne, $r_{1,1}$ oraz $r_{1,2}$, przetwarzające obrazy uzyskane z kamer. Wirtualny efektor e_1 jest odpowiedzialny za bezpośrednie sterowanie manipulatorem, natomiast za realizację zadania odpowiedzialny jest układ sterowania c_1.

Poniżej przedstawiono specyfikację działania systemu dla obu przypadków. Dodatkowo, dla pełności opisano również działanie robota w ruchu niewykorzystującym informację wizyjną.



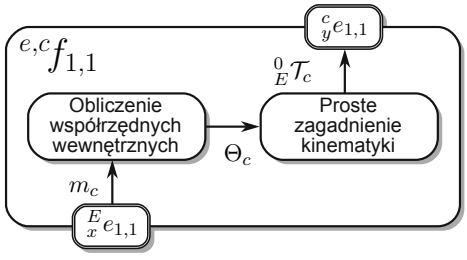
Rys. 4. Wyróżnione układy współrzędnych oraz transformacje pomiędzy nimi dla serwomechanizmów: (a) PB-SAC-EOL oraz (b) PB-EIH-EOL

Fig. 4. Major coordinate frames and transformations between them for: (a) PB-SAC-EOL and (b) PB-EIH-EOL visual servos

4.1. Wirtualny efektor $e_{1,1}$

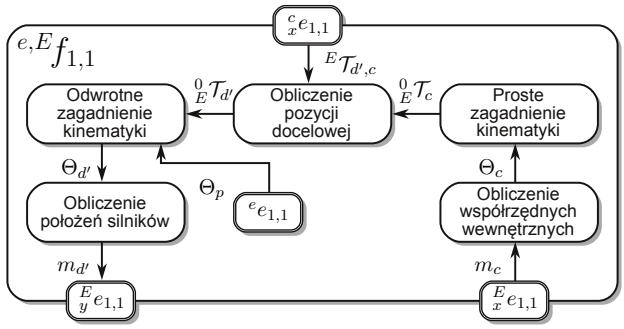
Niezależnie od tego, czy robot porusza się bez użycia wzroku, czy korzysta z kamery, sposób sterowania efektorem jest taki sam. Wystarczy więc jednokrotnie zdefiniować efektor wirtualny $e_{1,1}$, który następnie można wielokrotnie wykorzystywać. Zgodnie z przedstawioną wcześniej dekompozycją należy wyspecyfikować trzy funkcje: proprioreceptywną funkcję ${}^{e,c}f_{1,1}$ (rys. 5), dostarczającą układowi sterowania c_1 aktualną pozycję efektora rzeczywistego, funkcję sterującą efektorem ${}^{e,E}f_{1,1}$ (rys. 6) oraz funkcję pamięci ${}^{e,e}f_{1,1}$ (rys. 7), umożliwiającą zapamiętanie aktualnej pozycji tego efektora.

Funkcja proprioreceptywna (rys. 5) pobiera poprzez bufor ${}^{E}_{x}e_{1,1}$ wektor aktualnych położeń wałów silników m_c odczytany dzięki enkoderom, by wpierw je przekształcić w położenia złącz Θ_c (współrzędne wewnętrzne), a następnie w macierz jednorodną ${}^{0}_{E}\mathcal{T}_c$ określającą pozycję układu odniesienia związanego z efektorem E względem globalnego układu odniesienia 0 (współrzędne w przestrzeni operacyjnej), aby w końcu wynik tych obliczeń wyekspediować do układu sterowania poprzez ${}^{c}_{y}e_{1,1}$.



Rys. 5. Proprioreceptywna funkcja ${}^{e,c}f_{1,1}$ wirtualnego efektora $e_{1,1}$

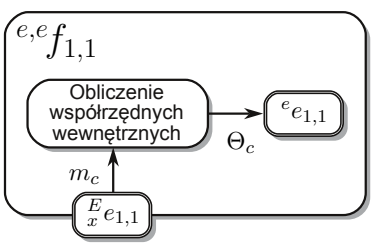
Fig. 5. The virtual effector $e_{1,1}$ proprioceptive function ${}^{e,c}f_{1,1}$



Rys. 6. Funkcja ${}^{e,E}f_{1,1}$ wirtualnego efektora $e_{1,1}$ sterująca efektorem rzeczywistym $E_{1,1}$

Fig. 6. Real effector control function ${}^{e,E}f_{1,1}$ of the virtual effector $e_{1,1}$

Funkcja sterująca efektorem ${}^{e,E}f_{1,1}$ (rys. 6) pobiera poprzez ${}^{E}_{x}e_{1,1}$ aktualne pozycje silników m_c i przekształca je w położenia złącz Θ_c, a następnie oblicza ${}^{0}_{E}\mathcal{T}_c$ rozwiązując proste zagadnienie kinematyki. Układ sterowania poprzez ${}^{c}_{x}e_{1,1}$ przekazuje efektorowi wirtualnemu pożądany przyrost pozycji efektora ${}^{E}\mathcal{T}_{d',c}$. Pożądana pozycja zadana efektora względem globalnego układu odniesienia obliczana jest jako ${}^{0}_{E}\mathcal{T}_{d'} = {}^{0}_{E}\mathcal{T}_c\ {}^{E}\mathcal{T}_{d',c}$. Wyznaczona macierz wykorzystywana jest, wraz z zapamiętanym Θ_p, do rozwiązania odwrotnego zagadnienia kinematyki, a w konsekwencji otrzymywany jest wektor pożądanych pozycji złącz $\Theta_{d'}$. Na tej podstawie wyznaczane są zadawane położenia wałów silników $m_{d'}$.

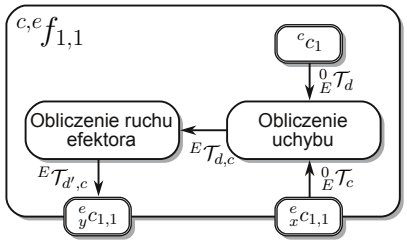


Rys. 7. Funkcja pamięci ${}^{e,e}f_{1,1}$ wirtualnego efektora $e_{1,1}$

Fig. 7. Virtual effector $e_{1,1}$ memory function ${}^{e,e}f_{1,1}$

Funkcja pamięci ${}^{e,e}f_{1,1}$ (rys. 7) pobiera z ${}^{E}_{x}e_{1,1}$ aktualne pozycje silników m_c i przekształca je w położenia złącz Θ_c. Obliczone położenia złącz umieszcza w pamięci wewnętrznej efektora ${}^{e}e_{1,1}$, aby funkcja sterująca mogła je wykorzystać do selekcji właściwego rozwiązania odwrotnego zagadnienia kinematyki w następnym kroku sterowania (jako Θ_p).

Nietrudno zauważyć, że wymienione funkcje efektora wirtualnego wzajemnie powtarzają niektóre obliczenia. Należy jednak pamiętać, że naszym celem jest uzyskanie przejrzystej specyfikacji, a optymalizację kodu wynikowego należy przeprowadzić w trakcie implementacji.



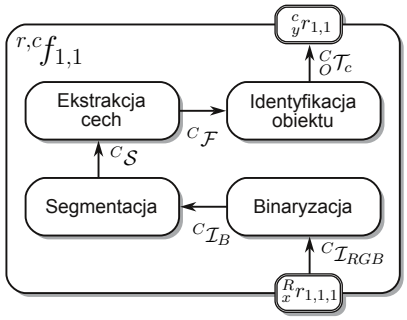
Rys. 8. Funkcja przejścia ${}^{c,e}f_{1,1}$ układu sterowania c_1
Fig. 8. The ${}^{c,e}f_{1,1}$ transition function of the c_1 control subsystem

4.2. Ruch pozycyjny – bez użycia wzroku

Jeżeli robot ma wykonać ruch „na ślepo", nie jest mu potrzebna kamera ani czujnik wirtualny z nią związany. Agent musi przechowywać w pamięci wewnętrznej ${}^{c}c_1$ układu sterowania c_1 gotową trajektorię zadaną albo parametry potrzebne do jej wygenerowania. W tych przypadkach funkcja przejścia ${}^{c,e}f_{1,1}$ wykonuje następujące obliczenia. Z pamięci ${}^{c}c_1$ pobierane są kolejne pozycje zadane efektora ${}^{0}_{E}\mathcal{T}_{d}$. Aktualna pozycja efektora ${}^{0}_{E}\mathcal{T}_{c}$ uzyskiwana jest z efektora wirtualnego za pośrednictwem ${}^{e}_{x}c_{1,1}$. Na ich podstawie obliczany jest pożądany przyrost pozycji, wyznaczany jako ${}^{E}\mathcal{T}_{d,c} = {}^{0}_{E}\mathcal{T}_{d}^{-1}\ {}^{0}_{E}\mathcal{T}_{c}$. Ponieważ przyrost ten może być zbyt duży w stosunku do kwantu czasu, w którym ma go zrealizować efektor wirtualny, dlatego jest on odpowiednio redukowany do przyrostu realizowalnego ${}^{E}\mathcal{T}_{d',c}$, który następnie przekazywany jest efektorowi wirtualnemu poprzez bufor ${}^{e}_{y}c_{1,1}$.

4.3. Wirtualne czujniki $r_{1,1}$ oraz $r_{1,2}$

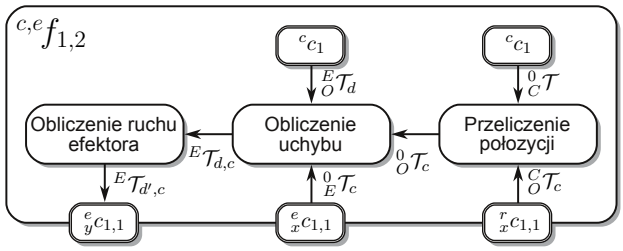
Serwomechanizmy wizyjne wymagają użycia kamer, a w konsekwencji stworzenia czujnika wirtualnego. W prezentowanym przykładowym zadaniu celem obu układów wizyjnych jest lokalizacja niebieskiej piłeczki, za położeniem której będzie podążała końcówka manipulatora. W związku z faktem, iż zadania te są identyczne (z dokładnością do indeksów buforów komunikacyjnych), poniżej wyspecyfikowano jedynie czujnik $r_{1,1}$.



Rys. 9. Funkcja agregująca odczyty ${}^{r,c}f_{1,1}$ wirtualnego czujnika $r_{1,1}$
Fig. 9. The reading aggregation function ${}^{r,c}f_{1,1}$ of the $r_{1,1}$ virtual receptor

Ponieważ działanie tego układu nie wymaga wykorzystania pamięci sensorycznej i konfiguracji czujnika, dlatego wymagane jest jednynie zdefiniowanie funkcji agregującej odczyty ${}^{r,c}f_{1,1}$ (rys. 9). Odebrany z kamery obraz ${}^{C}\mathcal{I}_{RGB}$ poddawany jest procesowi klasyfikacji koloru, w wyniku którego w binarnym obrazie ${}^{C}\mathcal{I}_{B}$ zaznaczone zostają jedynie piksele posiadające odpowiedni kolor. W wyniku działania segmentacji wszystkie sąsiadujące ze sobą piksele o tym samym kolorze będą połączone w segmenty ${}^{C}\mathcal{S} = \langle {}^{C}\mathcal{S}_1,\ \ldots,\ {}^{C}\mathcal{S}_{s_n} \rangle$, z których każdy $s$ty segment zostanie następnie opisany wektorem cech ${}^{C}\mathcal{F}_{s}$ (położenie środka segmentu w obrazie, jego wysokość, szerokość oraz niezmienniki momentowe związane z jego kształtem). Na podstawie wartości cech poszczególnych segmentów ${}^{C}\mathcal{F} = \langle {}^{C}\mathcal{F}_1,\ \ldots,\ {}^{C}\mathcal{F}_{s_n} \rangle$ wybierany jest segment, którego cechy najlepiej pasują do parametrów niebieskiej piłeczki. Następnie pozycja obiektu w układzie kamery ${}^{C}_{O}\mathcal{T}_{c}$, obliczona na podstawie wartości jego cech, zostaje przesyłana przez bufor ${}^{c}_{y}r_{1,1}$ do układu sterowania. Ze względu na kulisty kształt obiektu macierz ${}^{C}_{O}\mathcal{T}_{c}$ ma określone jedynie położenie – orientacja jest w przypadku kul nieistotna.

4.4. Serwomechanizm PB-EOL-SAC



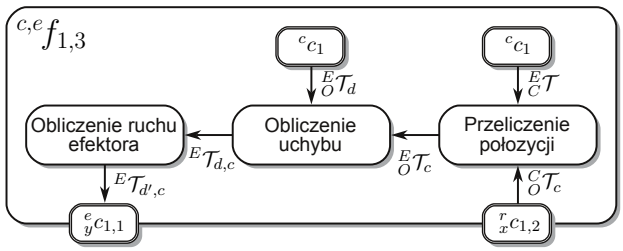
Rys. 10. Funkcja przejścia ${}^{c,e}f_{1,2}$ układu sterowania c_1 odpowiedzialna za realizację serwomechanizmu PB-EOL-SAC
Fig. 10. The ${}^{c,e}f_{1,2}$ transition function of the c_1 control subsystem executing the PB-EOL-SAC visual servo

Ponieważ podczas działania serwomechanizmu PB-EOL-SAC układ sterowania nie zapamiętuje żadnej informacji, nie kontaktuje się z innymi agentami oraz nie konfiguruje swego czujnika wirtualnego, a jego cel ogranicza się do sterowania efektorem wirtualnym, niezbędne jest zdefiniowanie jedynie funkcji przejścia ${}^{c,e}f_{1,2}$ (rys. 10), a pozostałe funkcje (${}^{c,c}f_{1,2}$, ${}^{c,T}f_{1,2}$ oraz ${}^{c,r}f_{1,2}$) są zbędne.

Funkcja ${}^{c,e}f_{1,2}$ wykonuje następujące obliczenia. Układ sterowania c_1 przechowuje w pamięci ${}^{c}c_1$ stałą pozycję nieruchomej (SAC) kamery względem globalnego układu odniesienia ${}^{0}_{C}\mathcal{T}$. Z czujnika wirtualnego $r_{1,1}$ układ sterowania poprzez ${}^{r}_{x}c_{1,1}$ uzyskuje aktualną pozycję śledzonego obiektu ${}^{C}_{O}\mathcal{T}_{c}$. Na tej podstawie wyznaczana jest pozycja obiektu w globalnym układzie odniesienia ${}^{0}_{O}\mathcal{T}_{c} = {}^{0}_{C}\mathcal{T}\ {}^{C}_{O}\mathcal{T}_{c}$. Aby uchwycić dany obiekt efektor musi być odpowiednio w stosunku do niego ustawiony. W związku z tym niezbędna jest znajomość wartości pożądanego odsunięcia ${}^{E}_{O}\mathcal{T}_{d}$, przechowywanego w pamięci ${}^{c}c_1$. Na podstawie obliczonej pozycji obiektu, pożądanego odsunięcia oraz aktualnej pozycji efektora ${}^{0}_{E}\mathcal{T}_{c}$ (dostarczonej przez wirtualny efektor poprzez ${}^{e}_{x}c_{1,1}$) obliczany jest uchyb ${}^{E}\mathcal{T}_{d,c} = {}^{E}_{O}\mathcal{T}_{d}\ {}^{0}_{O}\mathcal{T}_{c}^{-1}\ {}^{0}_{E}\mathcal{T}_{c}$. Analogicznie jak w ruchu pozycyjnym, przyrost ten może być nadmierny w stosunku do kwantu czasu, w którym ma go zrealizować efektor wirtualny, dlatego wyznaczany jest realizowalny przyrost ${}^{E}\mathcal{T}_{d',c}$, przekazywany efektorowi wirtualnemu poprzez ${}^{e}_{y}c_{1,1}$.

4.5. Serwomechanizm PB-EOL-EIH

Funkcja przejścia ${}^{c,e}f_{1,3}$ odpowiedzialna za pracę serwomechanizmu PB-EOL-EIH pokazana została na rys. 11. Ak-



Rys. 11. Funkcja przejścia $^{c,e}f_{1,3}$ układu sterowania c_1 odpowiedzialna za realizację serwomechanizmu PB-EOL-EIH

Fig. 11. The $^{c,e}f_{1,3}$ transition function of the c_1 control subsystem executing the PB-EOL-EIH visual servo

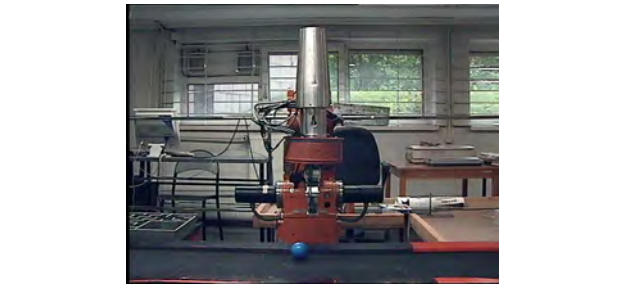
tualna odległość między efektorem a obiektem wyznaczana jest na podstawie aktualnego położenia obiektu w układzie kamery oraz stałej (przechowywanej w pamięci $^{c}c_1$) transformacji pomiędzy układem kamery a końcówką manipulatora $^{E}_{O}\mathcal{T}_c = {}^{E}_{C}\mathcal{T}\ {}^{C}_{O}\mathcal{T}_c$. Następnie uchyb wyznaczany jest jako $^{E}\mathcal{T}_{d,c} = {}^{E}_{O}\mathcal{T}_d\ {}^{E}_{O}\mathcal{T}_c^{-1}$. Dalsze obliczenia, związane z realizowalnością ruchu, są tożsame z tymi wykonywanymi przez $^{c,e}f_{1,1}$ i $^{c,e}f_{1,2}$.

4.6. Eksperymenty

Wyspecyfikowane zachowania zostały zaimplementowane jako system zbudowany na bazie dwóch programowych struktur ramowych (tzw. zręby). Pierwsza z nich, zrąb MRROC++ [13] (ang. *Multi-Robot Research Oriented Controler*), służy do tworzenia sterowników systemów wielorobotowych. MRROC++ został zweryfikowany w wielu różnorodnych robotycznych zadaniach, np. polerowanie i frezowanie obiektów [11] czy układanie kostki Rubika [12]. Drugim zrębem jest FraDIA (ang. *Framework for Digital Image Analysis*) [6], wizyjna struktura ramowa, która umożliwia proste tworzenie różnorodnych zadań wizyjnych.

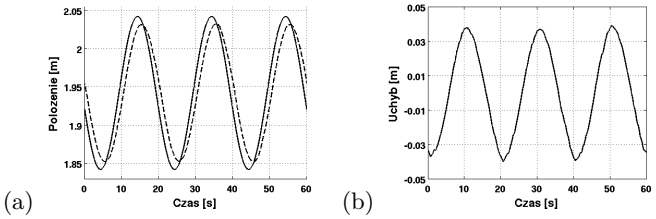
Sposób integracji tych struktur został opisany w pracy [1]. W przypadku serwomechanizmów wizyjnych w ramach MRROC++ zrealizowane zostały funkcje przejścia układu sterowania oraz funkcje wirtualnego efektora, natomiat FraDIA pełniła rolę receptora wirtualnego. Warte podkreślenia jest, iż to samo zadanie wizyjne wykorzystano w obu serwomechanizmach w roli czujników wirtualnych $r_{1,1}$ oraz $r_{1,2}$ – oczywiście z różnymi kamerami.

W celu weryfikacji działania obu serwomechanizmów stworzono zadania dwurobotowe (z dwoma różnymi konfiguracjami kamery). Pierwszy robot (taśmociąg – jego sterownik nie został tu wyspecyfikowany) poruszał obiektem ruchem sinusoidalnym, podczas gdy drugi robot (manipulator IRp-6 – jego sterownik został tu wyspecyfikowany) śledził przedmiot leżący na taśmociągu. Podczas pracy zbierane były pomiary pozycji końcówki roboczej manipulatora oraz pozycji taśmociągu. Na podstawie tych pomiarów obliczono rzeczywisty uchyb serwomechanizmu wizyjnego. Wyniki (rys. 13 oraz rys. 14) potwierdziły stabilne działanie zaimplementowanego układu sterowania, a przez to również poprawność samej specyfikacji.



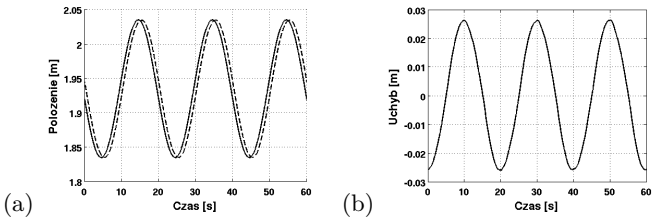
Rys. 12. Zmodyfikowany manipulator IRp-6 z kamerą zintegrowaną z chwytakiem w trakcie eksperymentu śledzenia piłeczki PB-EOL-EIH)

Fig. 12. The modified IRP-6 manipulator with a camera integrated with its gripper during ball tracking experiment



Rys. 13. Śledzenie obiektu z wykorzystaniem serwomechanizmu PB-EOL-SAC: (a) położenie przedmiotu (linia ciągła) i położenie końcówki roboczej (linia przerywana) oraz (b) uchyb

Fig. 13. Object tracking based on the PB-EOL-SAC visual servoing: (a) the location of the object (solid line) and the location of the end-effector (dashed line) and (b) the measured error



Rys. 14. Śledzenie obiektu z wykorzystaniem serwomechanizmu PB-EOL-EIH: (a) położenie przedmiotu (linia ciągła) i położenie końcówki roboczej (linia przerywana) oraz (b) uchyb

Fig. 14. Object tracking based on the PB-EOL-EIH visual servoing: (a) the location of the object (solid line) and the location of the end-effector (dashed line) and (b) the measured error

5. Podsumowanie

W artykule zaprezentowano metodę specyfikacji złożonych systemów robotycznych wykorzystującą podejście agentowe oraz funkcje przejścia. Zaprezentowana metoda, polegająca na wielokrotnej dekompozycji systemu na coraz prostsze elementy oraz operacje, umożliwia opis systemu w detalach umożliwiających ich dalsze przełożenie na implementację. Ważną zaletą zaproponowanej dekompozycji jest fakt, iż poszczególne elementy systemu (np. wirtualne efektory czy czujniki) mogą być implementowane oraz testowane niezależnie, a następnie stosowane w różnych kombinacjach w celu osiągnięcia różnorodnych zachowań.

Jako przykład obrazujący wykorzystanie metody projektowania wybrano złożone zachowania robota realizowane przez serwomechanizmy wizyjne. Uwagę skupiono na dwóch, z punktu widzenia konfiguracji sprzętowej, zupełnie różnych serwomechanizmach, przy czym w obu przypadkach wykorzystano identyczny wirtualny czujnik (podsystem wizyjny) oraz ten sam wirtualny efektor (sterownik efektora). Dodatkowo warto podkreślić, iż poza serwowizją

wspomniany wirtualny efektor został wykorzystany również w ruchu pozycyjnym, w którym nie korzystano z informacji wizyjnej – a więc w diametralnie różnym zachowaniu. Przeprowadzone eksperymenty potwierdziły poprawność działania wyspecyfikowanych zachowań oraz przydatność samego podejścia.

Podziękowania

Autorzy pragną podziękować inż. Mateuszowi Boryniowi za przeprowadzenie eksperymentów oraz zebranie danych pomiarowych wykorzystanych w tym artykule.

Praca finansowana z funduszy statutowych Instytutu Automatyki i Informatyki Stosowanej Politechniki Warszawskiej.

Bibliografia

1. Boryń M., Kornuta T., Zieliński C. (2011): *Struktura ramowa do implementacji i testowania serwomechanizmów wizyjnych*, „Pomiary – Automatyka – Robotyka" 2/2011, 677–686.
2. Cetnarowicz K. (2010): *Inteligencja wokół nas. Współdziałanie agentów softwareowych, robotów, inteligentnych urządzeń*, chapter *M–agent*, EXIT, 137–167.
3. Chaumette F., Hutchinson S. (2006): *Visual Servo Control, Part I: Basic Approaches*, „IEEE Robotics and Automation Magazine" 13(4), 82–90.
4. Chaumette F., Hutchinson S. (2007): *Visual Servo Control, Part II: Advanced Approaches*, „IEEE Robotics and Automation Magazine" 14(1), 109–118.
5. Chaumette F., Hutchinson S. (2008): *The Handbook of Robotics*, chapter *Visual Servoing and Visual Tracking*, Springer, 563–583.
6. Kornuta T. (2010): *Application of the FraDIA vision framework for robotic purposes*, [w:] Bolc L., Tadeusiewicz R., Chmielewski L., Wojciechowski K. (red.): *Proceedings of the International Conference on Computer Vision and Graphics*, Part II, „Lecture Notes in Computer Science", volume 6375, Springer Berlin/Heidelberg, 65–72.
7. Shoham Y. (1993): *Agent-Oriented Programming*, „Artif. Intell." 60(1), 51–92.
8. Staniak M., Zieliński C. (2010): *Structures of visual servos*, „Robotics and Autonomous Systems" 58(8), 940–954.
9. Trojanek P., Zieliński C. (2011): *Dekompozycja i specyfikacja systemów robotowych*, [w:] Trybus L., Samolej S. (red.): *Projektowanie, analiza i implementacja systemów czasu rzeczywistego*, Wydawnictwa Komunikacji i Łączności, 53–64.
10. Zieliński C. (2010): *Formalne podejście do programowania robotów – struktura układu sterującego*, [w:] Ambroszkiewicz S., Borkowski A., Centarowicz K., Zieliński C. (red.): *Inteligencja wokół nas. Współdziałanie agentów softwareowych, robotów, inteligentnych urządzeń*, Seria Monografie Komitetu Automatyki i Robotyki PAN, EXIT, 267–300.
11. Zieliński C., Mianowski K., Nazarczuk K., Szynkiewicz W. (2003): *A Prototype Robot for Polishing and Milling Large Objects*, „Industrial Robot: An International Journal" 30(1), 67–76.
12. Zieliński C., Szynkiewicz W., Winiarski T., Staniak M., Czajewski W., Kornuta T. (2007): *Rubik's cube as a benchmark validating MRROC++ as an implementation tool for service robot control systems*, „Industrial Robot: An International Journal" 34(5), 368–375.
13. Zieliński C., Winiarski T. (2010): *Motion Generation in the MRROC++ Robot Programming Framework*, „International Journal of Robotics Research" 29(4), 386–413.

Specification of visual servo structures

Abstract: The paper presents a formal method of specifying complex robotic systems, applied to the description of three diverse robot behaviors: motion in Cartesian space to a given pose and two types of motions in which the goal was computed on the base of information retrieved from cameras (a camera integrated with the robot gripper and a camera statically mounted above the scene). The presented experimental results confirm the correctness of the developed systems.

Keywords: robot systems, agent systems, visual servoing, formal specification

mgr inż. Tomasz Kornuta

Absolwent Wydziału Elektroniki i Technik Informacyjnych Politechniki Warszawskiej. W 2003 roku uzyskał tytuł inżyniera, w 2005 tytuł magistra inżyniera. Od 2008 roku pracuje na etacie asystenta w Instytucie Automatyki i Informatyki Stosowanej (IAiIS), w ramach którego prowadzi zajęcia dydaktyczne, a od 2009 roku pełni funkcję Kierownika Laboratorium Podstaw Robotyki. Od 2005 roku w ramach doktoratu prowadzi badania związane z projektowaniem systemów robotycznych wykorzystujących paradygmat aktywnego czucia do analizy otoczenia. Jego główne zainteresowania naukowe obejmują wykorzystanie informacji wizyjnej w robotyce.

e-mail: tkornuta@ia.pw.edu.pl

prof. nzw. dr hab. inż. Cezary Zieliński

Jest profesorem nadzwyczajnym Politechniki Warszawskiej na Wydziale Elektroniki i Technik Informacyjnych. W latach 2002-2005 sprawował na tym wydziale funkcję prodziekana ds. nauki i współpracy międzynarodowej, 2005–2008 zastępcy dyrektora Instytutu Automatyki i Informatyki Stosowanej (IAiIS) ds. naukowych, a od 2008 pełni funkcję dyrektora tego instytutu. Od uzyskania habilitacji w roku 1996 pełni rolę kierownika Zespołu Robotyki w IAiIS. Od 2007 roku jest członkiem i sekretarzem Komitetu Automatyki i Robotyki Polskiej Akademii Nauk. Od 2008 roku współpracuje z Przemysłowym Instytutem Automatyki i Pomiarów PIAP. Jego zainteresowania badawcze koncentrują się na zagadnieniach związanych z programowaniem i sterowaniem robotów.

e-mail: c.zielinski@ia.pw.edu.pl