ZESZYTY NAUKOWE POLITECHNIKI ŚLĄSKIEJ 1994
Seria: AUTOMATYKA z.115 Nr kol. 1251

Krzysztof ROMANOWSKI, Waldemar WRÓBLEWSKI
Katedra Automatyki, Robotyki i Informatyki, Politechnika Poznańska

RÓWNOLEGŁY ALGORYTM OBLICZENIOWY ZAGADNIENIA ODWROTNEGO DYNAMIKI MANIPULATORÓW

Streszczenie: W pracy omówiono paralelizację algorytmu obliczeniowego zagadnienia odwrotnego dynamiki manipulatorów, opracowanego z wykorzystaniem formalizmu Lagrange'a i macierzy rotacji (3×3). Omówiono również wersję algorytmu z predykcją pomocniczych danych. Przedstawione zostały rozważania na temat obliczeniochłonności i złożoności obliczeniowej algorytmów w zależności od liczby stopni swobody manipulatora i liczby wykorzystywanych procesorów.

PARALLEL COMPUTATIONAL ALGORITHM OF THE ROBOT INVERSE DYNAMICS PROBLEM

Summary: A parallelization of an algorithm of the solution of the manipulator inverse dynamics problem is presented. The algorithm is based on the Lagrange formalism and makes use of (3×3) rotation matrices. A version of the algorithm with prediction of temporary data is also described. The computational complexity of the parallel algorithm with respect to the number of degrees of freedom of the manipulator and to the number of processors used is considered.

PARALLEL ALGORITHMUS DER BERECHNUNGEN FÜR UMKEHRAUFGABE DER ROBOTERDYNAMIK

Zusammenfassung: In diesem Bericht wurde die Parallelisierung der Berechnungen für die Umkehraufgabe der Roboterdynamik beschrieben. Der hier präsentierte Algorithmus wurde mittels der Lagrangeschen Gleichungen und der Rotationmatrizen (3×3) erarbeitet. Zusätzlich auch eine Version des Algorithmus mit der Prediktion wurde beschrieben. Ebenso wird die Disskusion über die Abhängigkeit der Berechnungsaufwand von der Anzahl der Freiheitsgrade eines Manipulatores und der Anzahl der verwendeten Prozessoren präsentiert.

1. Wstęp

Metody rozwiązywania odwrotnego zagadnienia dynamiki manipulatorów oparte są zazwyczaj na formalizmie Lagrange'a albo Newtona-Eulera. Generalnie formalizm Newtona-Eulera pozwala na otrzymanie najefektywniejszego algorytmu w postaci ogólnej, bez przeprowadzania indywidualizacji (ang. *customization*) z uwzględnieniem parametrów geometrycznych i dynamicznych konkretnej struktury manipulatora. Jednak zastosowanie macierzy rotacji (3×3) i zależności rekurencyjnych w ramach formalizmu Lagrange'a prowadzi do algorytmu o porównywalnej efektywności, zwłaszcza dla praktycznie spotykanej liczby ogniw manipulatora [5]. W niniejszej pracy przedstawiono wersje tego algorytmu wykorzystujące obliczenia równoległe. Są one przeznaczone dla systemów wieloprocesorowych typu MIMD (ang. Multiple Instruction, Multiple Data) o pamięci rozproszonej, takich jak np. systemy transputerowe. Systemy tego rodzaju są dogodną platformą dla algorytmów charakteryzujących się równoległością na poziomie zadań i stosunkowo luźną zależnością między danymi przetwarzanymi przez poszczególne zadania.

Zasadą podziału zadań obliczeniowych prezentowanego algorytmu jest przydzielenie osobnego procesora lub pary procesorów dla obliczeń odpowiadających każdemu z węzłów manipulatora. W typowym przypadku manipulatora o sześciu ogniwach wymagany jest więc system sześcio- lub dwunastoprocesorowy. Zakłada się, że procesory połączone są w konfigurację *pipeline* umożliwiającą dwukierunkową transmisję danych. Kierunki transmisji odpowiadają kierunkom zależności rekurencyjnych: od podstawy do końcówki manipulatora (*forward*) – w ten sposób wyznaczane są wielkości kinematyczne w kolejnych węzłach – oraz od końcówki do podstawy manipulatora (*backward*) – tak obliczane są wielkości dynamiczne w kolejnych węzłach.

Zależności rekurencyjne wprowadzają silny związek między danymi odnoszącymi się do kolejnych węzłów. Aby nie dopuścić do zdominowania pracy systemu przez transmisje danych, zastosowano predykcję wielkości pośrednich potrzebnych w obliczeniach [6], możliwą w sytuacji, gdy całość obliczeń powtarza się dla kolejnych próbek czasowych. Dzięki temu procesory realizujące poszczególne ogniwa rekurencji nie czekają na siebie, lecz korzystają z wartości obliczonych dla poprzedniej próbki. Postępowanie takie prowadzi nieuchronnie do pewnych błędów obliczeń, które maleją przy skracaniu okresu próbkowania.

2. Podstawowy algorytm obliczeniowy

Rozpatrzmy manipulator o N ogniwach, stanowiący otwarty łańcuch kinematyczny, którego kolejne ogniwa są numerowane od nieruchomej podstawy (ogniwo 0) do jego końcówki (ogniwo N). Złącza są numerowane tak, że i-te złącze jest umiejscowione między ogniwem $(i-1)$-ym a i-tym, przy czym każde złącze reprezentuje jeden stopień swobody.

Uogólnione siły τ_i ($i = 1, \cdots, N$), działające w poszczególnych złączach manipulatora, są określone równaniami Lagrange'a o postaci:

$$\tau_i = \frac{d}{dt}\left(\frac{\partial K}{\partial \dot{q}_i}\right) - \frac{\partial K}{\partial q_i} + \frac{\partial V}{\partial q_i}, \quad i = 1, \cdots, N, \tag{1}$$

gdzie q_i jest i-tą współrzędną uogólnioną, tzn. współrzędną naturalną ogniwa będącą jednym z parametrów Denavita-Hartenberga (D-H). K, V są odpowiednio całkowitą energią kinetyczną, obliczoną względem inercjalnego układu współrzędnych, oraz energią potencjalną manipulatora.

Po przyporządkowaniu poszczególnym ogniwom układów współrzędnych według notacji D-H, macierz rotacji ${}^{i-1}_{i}R$ i wektor położenia ${}^{i-1}P_i$, określające orientację i-tego układu współrzędnych względem $i-1$-go układu oraz położenie początku i-tego układu współrzędnych względem $i-1$-tego układu, określone są następująco [2]:

$${}^{i-1}_{i}R = \begin{bmatrix} \cos\theta_i & -\cos\alpha_i \sin\theta_i & \sin\alpha_i \sin\theta_i \\ \sin\theta_i & \cos\alpha_i \cos\theta_i & -\sin\alpha_i \cos\theta_i \\ 0 & \sin\alpha_i & \cos\alpha_i \end{bmatrix}, \qquad {}^{i-1}P_i = \begin{bmatrix} a_i \cos\theta_i \\ a_i \sin\theta_i \\ d_i \end{bmatrix}, \tag{2}$$

gdzie a_i, α_i, d_i oraz θ_i są parametrami D-H i-tego ogniwa manipulatora.

Zastosowanie opisu kinematyki manipulatora za pomocą macierzy rotacji (3×3) i wektora położenia (3×1), zamiast tradycyjnie stosowanych macierzy (4×4) transformacji jednorodnych prowadzi do zmniejszenia liczby wykonywanych działań elementarnych podczas obliczania sił uogólnionych. Przykładowo, przy mnożeniu dwóch macierzy (3×3) uzyskuje się redukcję liczby wykonywanych działań o przeszło 50% w stosunku do wykonywania tej operacji na macierzach (4×4) [5].

Wykonanie formalnych operacji różniczkowań, występujących we wzorze (1), dla manipulatora opisanego według notacji (2), a następnie wykorzystanie rekurencyjnych zależności określających prędkości i przyspieszenia prowadzi do rekurencyjnego algorytmu obliczeniowego [4, 5], który można zaimplementować w systemie jednoprocesorowym. Algorytm ten można przedstawić następująco:

1. Obliczenia rekurencyjne od podstawy do końcówki manipulatora (*forward*).
   Zadanie $F_i, i = 1, \cdots, N$

$$ {}^{0}_{i}R(t) = {}^{0}_{i-1}R(t)\, {}^{i-1}_{i}R(t) \tag{3}$$

$$\frac{\partial\, {}^{0}_{i}R(t)}{\partial q_i} = {}^{0}_{i-1}R(t)\, \frac{\partial\, {}^{i-1}_{i}R(t)}{\partial q_i} \tag{4}$$

$$ {}^{0}_{i}\dot{R}(t) = {}^{0}_{i-1}\dot{R}(t)\, {}^{i-1}_{i}R(t) + \frac{\partial\, {}^{0}_{i}R(t)}{\partial q_i}\, \dot{q}_i(t) \tag{5}$$

$$ {}^{0}_{i}\ddot{R}(t) = {}^{0}_{i-1}\ddot{R}(t)\, {}^{i-1}_{i}R(t) + 2\, {}^{0}_{i-1}\dot{R}(t)\, \frac{\partial\, {}^{i-1}_{i}R(t)}{\partial q_i}\, \dot{q}_i(t) +$$

$$+ \; {}^{0}_{i-1}R(t)\,\frac{\partial^2\,{}^{i-1}_{i}R(t)}{\partial q_i^2}\,\dot{q}_i^2(t) + \frac{\partial\,{}^{0}_{i}R(t)}{\partial q_i}\,\ddot{q}_i(t) \tag{6}$$

$${}^{0}P_i(t) = {}^{0}P_{i-1}(t) + {}^{0}_{i-1}R(t)\,{}^{i-1}P_i(t) \tag{7}$$

$$\frac{\partial\,{}^{0}P_i(t)}{\partial q_i} = {}^{0}_{i-1}R(t)\,\frac{\partial\,{}^{i-1}P_i(t)}{\partial q_i} \tag{8}$$

$${}^{0}\ddot{P}_i(t) = {}^{0}\ddot{P}_{i-1}(t) + {}^{0}_{i-1}\ddot{R}(t)\,{}^{i-1}P_i(t) + 2\,{}^{0}_{i-1}\dot{R}(t)\,\frac{\partial\,{}^{i-1}P_i(t)}{\partial q_i}\,\dot{q}_i(t) +$$

$$+ \; {}^{0}_{i-1}R(t)\,\frac{\partial^2\,{}^{i-1}P_i(t)}{\partial q_i^2}\,\dot{q}_i^2(t) + \frac{\partial\,{}^{0}P_i(t)}{\partial q_i}\,\ddot{q}_i(t) \tag{9}$$

2. Obliczenia rekurencyjne od końcówki do podstawy manipulatora (*backward*). Zadanie $B_i, i = N, \cdots, 1$

$$b_i(t) = m_i + b_{i+1}(t) \tag{10}$$

$$e_i(t) = m_i\,{}^{0}\ddot{P}_i^T(t) + {}^{i}n_i^T\,{}^{0}_{i}\ddot{R}^T(t) + e_{i+1}(t) \tag{11}$$

$$D_i(t) = {}^{i}n_i\,{}^{0}\ddot{P}_i^T(t) + J_i\,{}^{0}_{i}\ddot{R}^T(t) + {}^{i}_{i+1}R(t)\,D_{i+1}(t) + {}^{i}P_{i+1}(t)e_{i+1}(t) \tag{12}$$

$$c_i(t) = {}^{i}n_i + b_{i+1}(t)\,{}^{i}P_{i+1}(t) + {}^{i}_{i+1}R(t)\,c_{i+1}(t) \tag{13}$$

$$\tau_i(t) = \mathrm{tr}\left(\frac{\partial\,{}^{0}_{i}R(t)}{\partial q_i}D_i(t) + \frac{\partial\,{}^{0}P_i(t)}{\partial q_i}e_i(t)\right) -$$

$$- \; g^T\left(\frac{\partial\,{}^{0}_{i}R(t)}{\partial q_i}c_i(t) + b_i(t)\frac{\partial\,{}^{0}P_i(t)}{\partial q_i}\right) \tag{14}$$

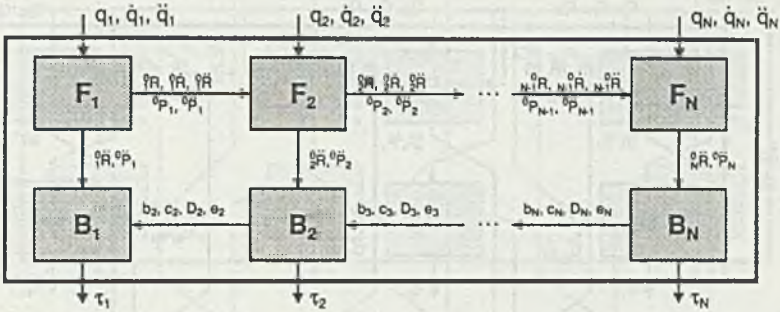
gdzie m_i jest masą i-tego ogniwa, ${}^{i}n_i = m_i\,{}^{i}P_{Ci}$ jest momentem statycznym i-tego ogniwa względem i-tego układu współrzędnych (${}^{i}P_{Ci}$ jest wektorem położenia środka masy), J_i jest macierzą pseudoinercji i-tego ogniwa względem i-tego układu współrzędnych, zaś g jest wektorem przyspieszenia grawitacyjnego. Występujące w powyższych wzorach wielkości ${}^{i-1}_{i}R(t)$ oraz ${}^{i-1}P_i(t)$ są obliczane na podstawie parametrów geometrycznych ogniwa, z których d_i lub θ_i stanowi współrzędną uogólnioną, w zależności od tego,czy i-te ogniwo jest przesuwne,czy obrotowe:

$${}^{i-1}_{i}R(t) = f_1(a_i, \alpha_i, d_i(t), \theta_i(t)) \tag{15}$$

$${}^{i-1}P_i(t) = f_2(a_i, \alpha_i, d_i(t), \theta_i(t)) \tag{16}$$

Wartości początkowe są następujące: $q_0 = 0$, $\dot{q}_0 = 0$, $\ddot{q}_0 = 0$, $e_{N+1} = 0$, $D_{N+1} = 0$, $c_{N+1} = 0$, $b_{N+1} = 0$. Pomocnicze wielkości, wyznaczane z zależności (10)-(13), mają następujące rozmiary: b_i – skalar, c_i – wektor kolumnowy (3×1), D_i – macierz (3×3), e_i – wektor wierszowy (1×3).

Aby obliczyć wartości sił uogólnionych τ_i, $i = 1, \cdots, N$, należy zatem wykonać sekwencyjnie wszystkie obliczenia etapów 1 i 2 powyższego algorytmu. Schemat blokowy przepływu danych dla tego algorytmu przedstawiono na rys. 1. Ze względu na występowanie dwóch etapów rekurencji algorytm ten jest analogiczny do algorytmu opartego na formalizmie Newtona-Eulera [2]. Jego złożoność obliczeniowa jest też równa $O(N)$.



Rys. 1. Przepływ danych dla podstawowego algorytmu obliczeniowego
Fig. 1. Data flow of the basic computational algorithm

3. Algorytm obliczeń równoległych z predykcją

Aby umożliwić podjęcie obliczeń przez zadania F_i, B_i bez oczekiwania na zakończenie zadań F_{i-1}, B_{i+1}, można dokonać predykcji pomocniczych wielkości, obliczanych przez te ostatnie zadania. Taki sposób zwiększenia efektywności obliczeniowej, polegający na wykorzystaniu zamiast bieżących wartości ich wartości z poprzednich próbek czasowych, przedstawiono m.in. w pracach [1, 3]. Predykcja, wprowadzana w omawianym algorytmie, polega na zastąpieniu bieżących wartości wielkości obliczanych przez zadania F_{i-1}, B_{i+1} przez ich wartości pochodzące z poprzedniej próbki czasowej. Algorytm z predykcją ma wówczas postać:

1. Obliczenia rekurencyjne od podstawy do końcówki manipulatora (*forward*). Zadanie $F_i, i = 1, \cdots, N$

$$ {}^0_iR(t) = {}^0_{i-1}R(t-\Delta t)\,{}^{i-1}_iR(t) \tag{17} $$

$$ \frac{\partial\,{}^0_iR(t)}{\partial q_i} = {}^0_{i-1}R(t-\Delta t)\,\frac{\partial\,{}^{i-1}_iR(t)}{\partial q_i} \tag{18} $$

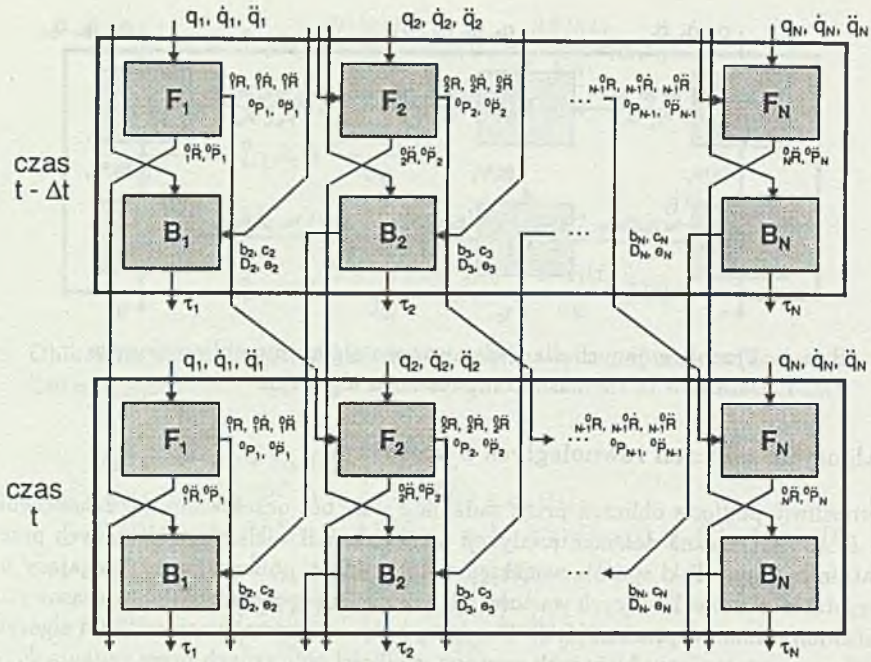
$$ {}^0_i\dot{R}(t) = {}^0_{i-1}\dot{R}(t-\Delta t)\,{}^{i-1}_iR(t) + \frac{\partial\,{}^0_iR(t-\Delta t)}{\partial q_i}\,\dot{q}_i(t) \tag{19} $$

$$ \begin{aligned} {}^0_i\ddot{R}(t) &= {}^0_{i-1}\ddot{R}(t-\Delta t)\,{}^{i-1}_iR(t) + 2\,{}^0_{i-1}\dot{R}(t-\Delta t)\,\frac{\partial\,{}^{i-1}_iR(t)}{\partial q_i}\,\dot{q}_i(t) + \\ &+ {}^0_{i-1}R(t-\Delta t)\,\frac{\partial^2\,{}^{i-1}_iR(t)}{\partial q_i^2}\,\dot{q}_i^2(t) + \frac{\partial\,{}^0_iR(t-\Delta t)}{\partial q_i}\,\ddot{q}_i(t) \end{aligned} \tag{20} $$

$$ {}^0P_i(t) = {}^0P_{i-1}(t-\Delta t) + {}^0_{i-1}R(t-\Delta t)\,{}^{i-1}P_i(t) \tag{21} $$

$$ \frac{\partial\,{}^0P_i(t)}{\partial q_i} = {}^0_{i-1}R(t-\Delta t)\,\frac{\partial\,{}^{i-1}P_i(t)}{\partial q_i} \tag{22} $$

$$ {}^0\ddot{P}_i(t) = {}^0\ddot{P}_{i-1}(t-\Delta t) + {}^0_{i-1}\ddot{R}(t-\Delta t)\,{}^{i-1}P_i(t) + $$

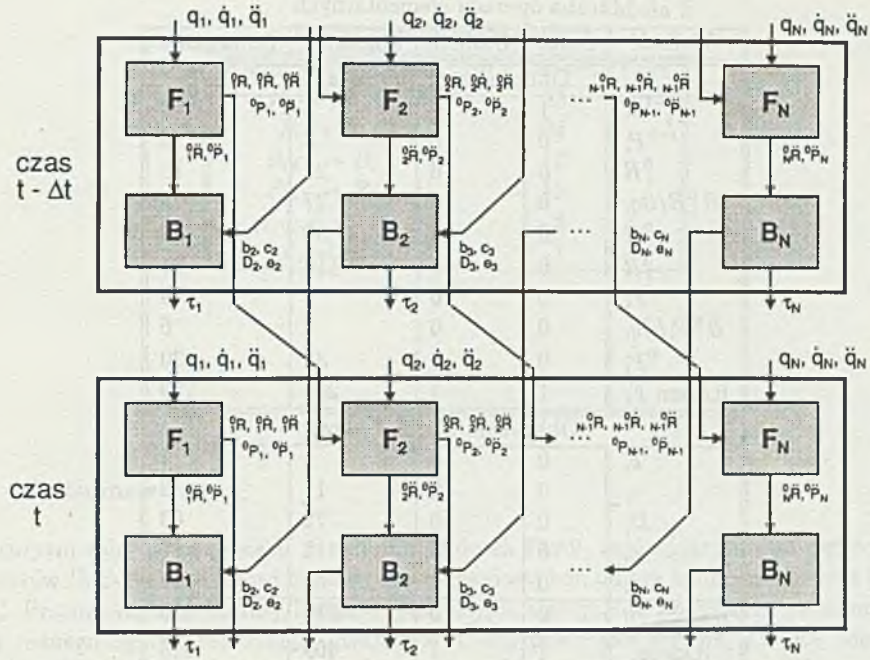


Rys. 2: Przepływ danych dla algorytmu obliczeniowego z predykcją
Fig. 2. Data flow of the computational algorithm with prediction

$$
\begin{aligned}
&+ \; 2\,{}^{0}_{i-1}\dot{R}(t-\Delta t)\,\frac{\partial\,{}^{i-1}P_i(t)}{\partial q_i}\,\dot{q}_i(t) + \\
&+ \; {}^{0}_{i-1}R(t-\Delta t)\,\frac{\partial^2\,{}^{i-1}P_i(t)}{\partial q_i^2}\,\dot{q}_i^2(t) + \frac{\partial\,{}^{0}P_i(t-\Delta t)}{\partial q_i}\,\ddot{q}_i(t) \qquad (23)
\end{aligned}
$$

2. Obliczenia rekurencyjne od końcówki do podstawy manipulatora (*backward*). Zadanie $B_i, i = N, \cdots, 1$

$$
\begin{aligned}
b_i(t) &= m_i + b_{i+1}(t-\Delta t) && (24)\\
e_i(t) &= m_i\,{}^{0}\ddot{P}_i^T(t-\Delta t) + {}^{i}n_i^T\,{}^{0}_{i}\ddot{R}^T(t-\Delta t) + e_{i+1}(t-\Delta t) && (25)\\
D_i(t) &= {}^{i}n_i\,{}^{0}\ddot{P}_i^T(t-\Delta t) + J_i\,{}^{0}_{i}\ddot{R}^T(t-\Delta t) + \\
&+ {}^{i}_{i+1}R(t)\,D_{i+1}(t-\Delta t) + {}^{i}P_{i+1}(t)e_{i+1}(t-\Delta t) && (26)\\
c_i(t) &= {}^{i}n_i + b_{i+1}(t-\Delta t)\,{}^{i}P_{i+1}(t) + {}^{i}_{i+1}R(t)\,c_{i+1}(t-\Delta t) && (27)\\
\tau_i(t) &= \mathrm{tr}\left(\frac{\partial\,{}^{0}_{i}R(t-\Delta t)}{\partial q_i}D_i(t) + \frac{\partial\,{}^{0}P_i(t-\Delta t)}{\partial q_i}e_i(t)\right) -
\end{aligned}
$$



Rys. 3. Przepływ danych dla zmodyfikowanego algorytmu obliczeniowego z predykcją
Fig. 3. Data flow of the modified computational algorithm with prediction

$$- \; g^T \left(\frac{\partial \, {}^0_i R(t-\Delta t)}{\partial q_i} c_i(t) + b_i(t) \frac{\partial \, {}^0 P_i(t-\Delta t)}{\partial q_i} \right) \tag{28}$$

Schemat blokowy przepływu danych dla algorytmu z predykcją przedstawiono na rys. 2. Występujące w powyższych wzorach wielkości ${}^{i-1}_{i}R(t)$ oraz ${}^{i-1}P_i(t)$ są obliczane na podstawie zależności (15), (16).

4. Zmodyfikowany algorytm obliczeń równoległych z predykcją

W sytuacji, gdy zadania F_i, B_i są wykonywane przez ten sam procesor, a w związku z tym transmisja danych między nimi nie wprowadza istotnego opóźnienia, istnieje możliwość wykonania ich kolejno. W takim przypadku zadanie B_i może wykorzystać bieżące wartości obliczane przez zadanie F_i, zamiast pochodzących z predykcji. Schemat blokowy zmodyfikowanego w ten sposób algorytmu przedstawiono na rys. 3.

Tabela 1

Liczba operacji elementarnych

Wielkość	Sinus	Cosinus	Mnożenie	Dodawanie
Obliczenia F_i (*forward*)				
${}^{i-1}_{i}R$	1	1	4	0
${}^{i-1}P_i$	0	0	2	0
${}^{0}_{i}R$	0	0	27	18
$\partial\, {}^{0}_{i}R/\partial q_i$	0	0	27	18
${}^{0}_{i}\dot{R}$	0	0	36	27
${}^{0}_{i}\ddot{R}$	0	0	110	81
${}^{0}P_i$	0	0	9	9
$\partial\, {}^{0}P_i/\partial q_i$	0	0	9	6
${}^{0}\ddot{P}_i$	0	0	38	30
Razem F_i	1	1	262	189
Obliczenia B_i (*backward*)				
b_i	0	0	0	1
e_i	0	0	12	12
D_i	0	0	72	63
c_i	0	0	12	12
τ_i	0	0	51	41
Razem B_i	0	0	147	129
Razem	1	1	409	318

5. Oszacowanie wymogów obliczeniowych

W celu oszacowania zasobów obliczeniowych wymaganych przez powyższe algorytmy w tabeli 1 przedstawiono liczby operacji dodawania, mnożenia i obliczania wartości funkcji sinus i cosinus wykonywanych w trakcie wyznaczania poszczególnych wielkości [5], a w tabeli 2 długości bloków danych przesyłanych między zadaniami.

Oznaczmy przez $t(x)$ czas wykonania obliczeń zadania x lub przesyłu danych w kierunku x. Widać, że dominującymi czasem obliczeń jest $t(F_i)$, a przesyłu $t(F_i \to F_{i+1})$. Czas obliczeń dla jednej chwili próbkowania przy użyciu podstawowego algorytmu w postaci sekwencyjnej wynosi $N(t(F_i) + t(B_i))$. Implementacja zaprezentowanego algorytmu z predykcją w taki sposób, że dla każdego z zadań F_i, B_i przydzielony jest osobny procesor, pozwala na wykonanie obliczeń dla jednej chwili próbkowania (dla wszystkich ogniw manipulatora) w czasie $t(F_i) + t(F_i \to F_{i+1})$ przy $2N$ procesorach. Implementacja zmodyfikowanego algorytmu z predykcją, gdzie osobny procesor przydzielony jest każdej parze zadań F_i, B_i, pozwala osiągnąć czas $t(F_i) + t(B_i)$ przy N procesorach. W praktyce czas obliczeń jest nieco większy, gdyż ponadto wykonywane są operacje inicjujące obliczenia rekurencyjne oraz organizujące przesyłanie danych między procesorami.

Tabela 2

Liczba przesyłanych danych

Kierunek przesyłu	Liczba danych `float`
$F_i \rightarrow F_{i+1}$	33
$F_i \rightarrow B_i$	24
$B_{i+1} \rightarrow B_i$	16
$F_{i+1} \rightarrow B_i$	12
$q_i, \dot{q}_i, \ddot{q}_i \rightarrow F_i$	3
$F_i \rightarrow \tau_i$	1

6. Podsumowanie

Algorytm zaimplementowano w systemie Quintek Fast9, zawierającym dziewięć transputerów IMS T805. Stanowi on kartę rozszerzeniową komputera kompatybilnego z IBM PC. Przeprowadzone zostały badania symulacyjne algorytmów na jednym transputerze dla jednego ogniwa rekurencji. Aktualnie trwają prace nad wykorzystaniem pełnych możliwości systemu.

LITERATURA

[1] Binder E.E., Herzog J.H.: Distributed Computer Architecture and Fast Parallel Algorithms in Real-Time Robot Control. IEEE Transactions on Systems, Man, and Cybernetics, vol.16, nr 4, 1986, s. 543–549.

[2] Craig J.J.: Introduction to Robotics. Mechanics and Control. Addison-Wesley Publishing Company, Reading 1989.

[3] Vuskovic M., Liang T., Anantha K.: Decoupled Parallel Recursive Newton-Euler Algorithm for Inverse Dynamics. Proceedings of the 1990 IEEE International Conference on Robotics and Automation, Cincinnati, OH, USA, 1990, s. 832–838.

[4] Warczyński J., Wróblewski W.: Inverse Dynamics Algorithm for Robot Manipulators. Foundations of Control Engineering, vol.13, nr 2, Poznań 1988, s. 93–100.

[5] Wróblewski W.: Metody modelowania dynamiki manipulatorów dla celów sterowania. Rozprawa doktorska, Poznań 1992.

[6] Yamakita M., Hoshino Y., Morimoto K., Furuta K.: Parallel Implementation of Newton-Euler Algorithm with One Step Ahead Prediction. Proceedings of the 1991

IEEE International Conference on Robotics and Automation, Sacramento, CA, USA, 1991, s. 1842–1849.

Recenzent: Dr inż. Tadeusz Szkodny

Wpłynęło do Redakcji do 30.04.1994r.

Abstract

The computation of a manipulator inverse dynamics is usually accomplished by using a method based on either the Lagrange or the Newton-Euler formulation. This paper presents a recursive algorithm using the Lagrange formulation with 3×3 rotation matrices that has been parallelized for MIMD (Multiple Instruction, Multiple Data) computing architectures with distributed memory, such as transputer systems.

The parallelization makes use of the recursive nature of the algorithm. The calculations pertaining to particular manipulator joints are placed on separate processors so that there is a one-to-one mapping between the joints and the processors. The processors are connected to form a bi-directional pipeline, which transmits the data coupling the joint equations in two directions. One direction corresponds to the forward recursion, in which the generalized positions, velocities, and accelerations are computed, and the other to the backward recursion, in which the generalized forces are found. An alternative mapping is also presented, where a pair of processors instead of one is dedicated to each joint, so that the forward and backward recurrence computations are placed on different processors. Accordingly, the single bi-directional pipeline is replaced by two uni-directional pipelines, each dedicated to one direction of the recurrence data flow.

A distributed-memory architecture presents a problem with recursive relationships, since there is a strong data dependence between the calculations performed by the particular processors, which hampers effective parallelization. The required interprocessor data flow may easily become the dominant system activity. Therefore the recurrence equations have been modified to make use of predicted instead of actual values of internal kinematic quantities. As it is assumed that the calculation of the inverse dynamics is performed repeatedly in time, the prediction is an extrapolation of the values at previous time instants.

The performance characteristics of the presented algorithm, both theoretical and measured, with respect to the number of processors and the number of manipulator joints, are discussed.