ZESZYTY NAUKOWE POLITECHNIKI ŚLĄSKIEJ 1986
Seria: AUTOMATYKA z. 86 Nr kol. 896

Zbigniew Rogowski, Krzysztof Królik
Instytut Mechaniki Precyzyjnej, Warszawa

JĘZYK PROGRAMOWANIA TEKSTOWEGO ROBOTÓW "ROBIMP" JAKO NARZĘDZIE AUTOMATYZACJI PRODUKCJI

Streszczenie. W artykule przedstawiono zwięzły opis języka programowania tekstowego robotów złożonych "ROBIMP". Omówiono podstawowe konstrukcje języka- sposób reprezentacji obiektów geometrycznych, instrukcje ruchowe oraz sterujące w nawiązaniu do obecnych i przewidywanych zastosowań robotów złożonych.

1. Wprowadzenie

Funkcjonowanie robota przemysłowego przejawia się w przemieszczaniu narzędzia /chwytaka, pistoletu malarskiego, zgrzewadła lub innego specjalizowanego urządzenia technologicznego/ po odpowiedniej trajektorii w przestrzeni roboczej. Istotnym zagadnieniem projektowym podczas konstruowania układu sterowania dla robota jest wiec wybór i realizacja możliwie efektywnego sposobu uczenia maszyny wymaganego przez operatora działania, czyli sposobu opisu trajektorii narzędzia podczas wykonywania zadania technologicznego.

Metodą szeroko stosowaną w tym celu, w wielu dawniejszych konstrukcjach robotów złożonych było prowadzenie końca łańcucha kinematycznego manipulatora przez wszystkie wymagane położenia za pomocą sterowników ręcznych lub bezpośrednio ręką operatora.

Nowym, gwałtownie się obecnie rozpowszechniającym sposobem informowania układu sterowania, o jego zadaniu, jest tekstowy opis czynności manipulatora sporządzony w specjalnym języku programowania tekstowego robotów.

Współczesne języki tej grupy, zarówno "handlowe" jak i doświadczalne dopuszczają różny stopień ogólności opisu zadania dla robota, od poleceń działań na własnym łańcuchu kinematycznym manipulatora, poprzez uproszczone modelowanie środowiska i działania na tym modelu aż do automatycznej generacji planu rozwiązania ogólnie opisanego problemu w oparciu o rozbudowany model środowiska stworzony dzięki informacjom zebranym za pomocą "zmysłów" /czujniki wizyjne, dotykowe, termiczne itp./.

Języki najwyższego z wymienionych poziomów /np. PLANNER, SAIL/ służą do badań nad sztuczną inteligencją, a w układach sterowania stosowanych w przemyśle dominuje obecnie pierwszy stopień ogólności opisu zadania

lecz już ich zastosowanie pozwala dostrzec szereg zalet programowania tekstowego w porównaniu z metodami tradycyjnymi.

Najważniejsze z tych zalet to:

- uniezależnienie sposobu uczenia i opisu działania od struktury kinematycznej manipulatora,
- zwięzły i czytelny opis wielowariantowego działania robota przy użyciu typowych konstrukcji językowych: podprogramów, rozgałęzień warunkowych, iteracji,
- skrócenie czasu przerw w pracy zrobotyzowanego gniazda technologicznego przy zmianie programu dzięki możliwości wykonania niektórych etapów programowania /edycja, testowanie/na innym komputerze, bez udziału układu sterowania robota,
- możliwość korzystania ze sprawdzonych wcześniej podprogramów rozwiązujących typowe problemy /np. synchronizacji robota z ruchem przenośnika detali/,
- zwiększenie komfortu i bezpieczeństwa pracy operatora.

2. Język programowania robotów "ROBIMP"

Definicję języka opracowano w Zakładzie Sterowania Robotów Instytutu Mechaniki Precyzyjnej w Warszawie. Język ten ma być środkiem programowania robotów złożonych wyposażonych w obecnie konstruowany, w oparciu o rodzinę mikroprocesorów Intel 8086/87, układ sterowania E-800.

2.1. Struktura programu

Tekst programu w języku ROBIMP ma strukturę wierszową /podobnie jak FORTRAN i BASIC/. Zamieszczony w dalszej części artykułu wyciąg z definicji składni prezentuje główne elementy języka, z których najbardziej charakterystyczne opisano poniżej.

2.2 Standardowe typy obiektów i operacje na obiektach

Obiektami języka są stałe, zmienne /proste oraz indeksowane/, wyrażenia i funkcje. Typ obiektu w przypadku stałej jest określony formą jej zapisu, dla funkcji i zmiennej standardowej jest identyfikowany nazwą a dla zmiennych niestandardowych jest obowiązkowo deklarowany. Z uwagi na przeznaczenie języka, oprócz typu numerycznego występują w nim obiekty innych typów umożliwiające dogodny zapis instrukcji ruchowych oraz instrukcji przesyłających i przetwarzających informację o środowisku robota.

Typy obiektów mających interpretację geometryczną:/rys. 1/

```
punkt = ARRAY [1..3] OF real
wektor = ARRAY [1..3] OF real
układ = ARRAY [1..12] OF real
trajektoria = FILE OF RECORD położenie:układ, czas:real END
```

Dla punktu i wektora wartości liczbowe są interpretowane odpowiednio, jako współrzędne punktu i składowe wektora w kartezjańskim układzie odniesienia. Dla układu wartości te są odczytywanymi kolumnowo wartościami elementów macierzy jednorodnej opisującej położenie obiektu względem układu odniesienia, zgodnie z konwencją przedstawioną w pracy [1]. W przypadku trajektorii "czas" jest okresem upływającym pomiędzy osiągnięciem przez narzędzie kolejnych sąsiednich położeń w przestrzeni.

Typy obiektów reprezentujących przemieszczenia obiektów geometrycznych:

```
obrót = ARRAY [1..9] OF real
transformacja = ARRAY [1..12] OF real
```

Wartości liczbowe są wartościami elementów jednorodnych macierzy obrotu i transformacji według konwencji D-H [1].

Typy obiektów reprezentujących dane binarne:

```
bit = (FALSE , TRUE)
bajt = ARRAY [1..8] OF bit
słowo = ARRAY [1..16] OF bit
```

Pomimo obowiązkowej deklaracji typów dla zmiennych niestandardowych zaleca się nadawanie zmiennym nienumerycznym nazw umożliwiających łatwe rozpoznanie typu /np. za pomocą dwóch pierwszych liter/, co zwiększa czytelność programu. Zasadę tą zastosowano do nazw obiektów na rys. 1.

Operacje przetwarzania danych binarnych i działania na obiektach geometrycznych są zapisywane z wykorzystaniem odpowiednich funkcji standardowych lub operatorów dwuargumentowych, o znaczeniu zależnym od typu użytych argumentów. Operacje logiczne służą do podejmowania decyzji o przyszłym działaniu, zależnie od stanu środowiska robota. Operacje na punktach, wektorach i układach umożliwiają badanie zależności geometrycznych między elementami "sceny" /np. obliczanie odległości/, modyfikację parametrów instrukcji ruchowych oraz rozbudowę modelu środowiska z wykorzystaniem informacji danej pierwotnie /wyznaczanie kierunków prostopadłych do płaszczyzn, wyznaczanie zbioru położeń detali na palecie itp./. Operacje z użyciem obrotów i transformacji służą do wyrażania przemieszczeń składników modelu środowiska.

W programie technologicznym oprócz obiektów o wyżej wymienionych typach może istnieć jeden obiekt o typie "przeszkoda" /rys. 2/. Przeszkoda,

której wartość ustalana jest w wyniku odpowiednich operacji na punktach, określa w przestrzeni roboczej obszar zabroniony dla ruchu narzędzia /ogólniej: dowolnego fragmentu manipulatora/ i zależnie od stopnia "inteligencji" translatora może służyć do badania bezkolizyjności zaprogramowanej trajektorii /w procesie symulacji i testowania/ lub do automatycznej generacji trajektorii bezkolizyjnej,jeśli ruch zadany nie jest w pełni określony /gdy kształt toru między startem i celem nie jest ważny ze względów technologicznych/.

2.3. Instrukcje ruchowe

Instrukcje ruchowe są charakterystycznym elementem języków opisujących czynności manipulatorów. W języku ROBIMP instrukcje te polecają przemieszczać układ odniesienia narzędzia, związany z końcem łańcucha kinematycznego. Przewidziano użycie następujących instrukcji ruchowych:

A/ Ruchy w pełni określone /znane położenie i orientacja narzędzia oraz prędkość ruchu w każdym punkcie trajektorii/:

- przesunięcie prostoliniowe: SHIFT (⟨cel⟩)
- przesunięcie po łuku okręgu: CSHIFT (⟨etap⟩, ⟨cel⟩)
  /ze stałą orientacją względem układu odniesienia/
- obrót układu narzędzia: TURN (⟨etap⟩, ⟨cel⟩)
  /ze stałą orientacją względem toru ruchu/
- odtworzenie trajektorii: FOLLOW (⟨nazwa trajektorii⟩)
- określona zmiana orientacji narzędzia:
  ORIENT (⟨wektor obrotu⟩, ⟨prędkość obrotu⟩)

Ruchy w pełni określone są stosowane do opisywania takich operacji technologicznych jak: spawanie, montaż, obróbka skrawaniem /szlifowanie/, malowanie, czyszczenie i podobne.

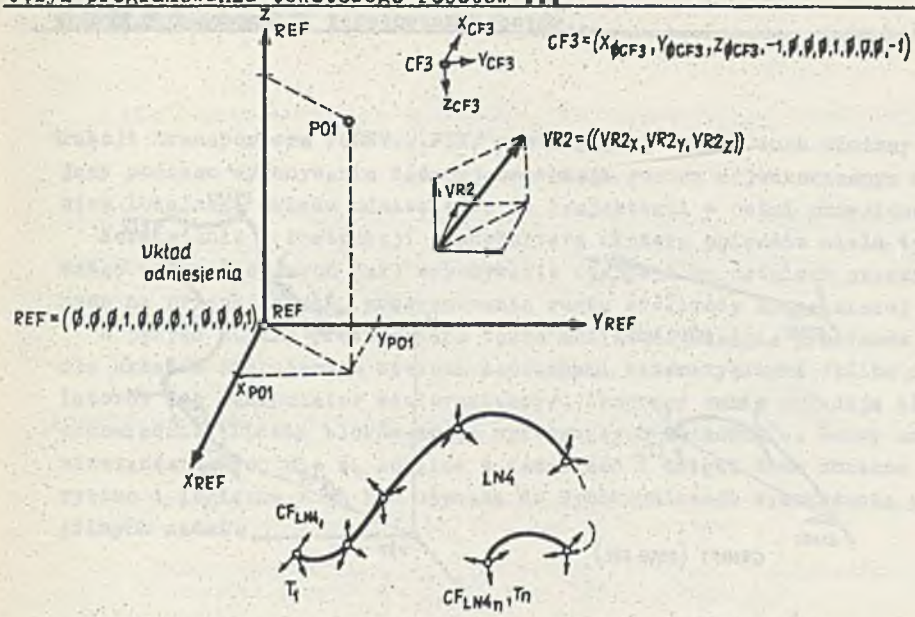
B/ Ruchy nie w pełni określone:

- przemieszczenie narzędzia do zadanego położenia:
  MOVE (⟨lista parametrów ruchu⟩)
- kopiowanie trajektorii: FOLLCOP (⟨nazwa trajektorii⟩)
- swobodna zmiana orientacji narzędzia: ORIENT(⟨nowa orientacja⟩)

Ruchy nie w pełni określone służą głównie do wykonywania operacji transportowych.

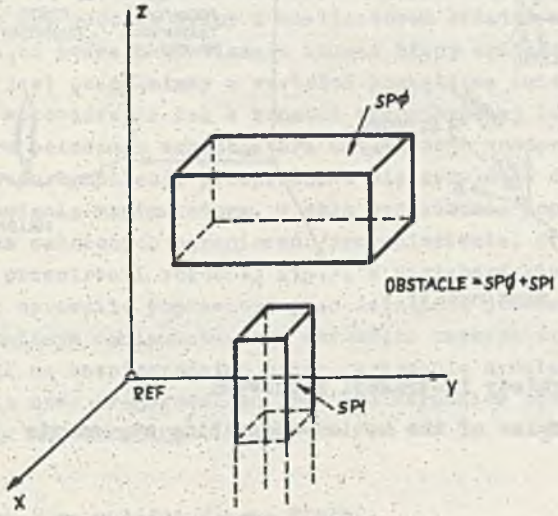
2.4 Instrukcje organizacyjne

Instrukcjami organizacyjnymi są strukturalne instrukcje rozgałęzienia warunkowego oraz iteracji. Ponadto w tekście programu można użyć ins-



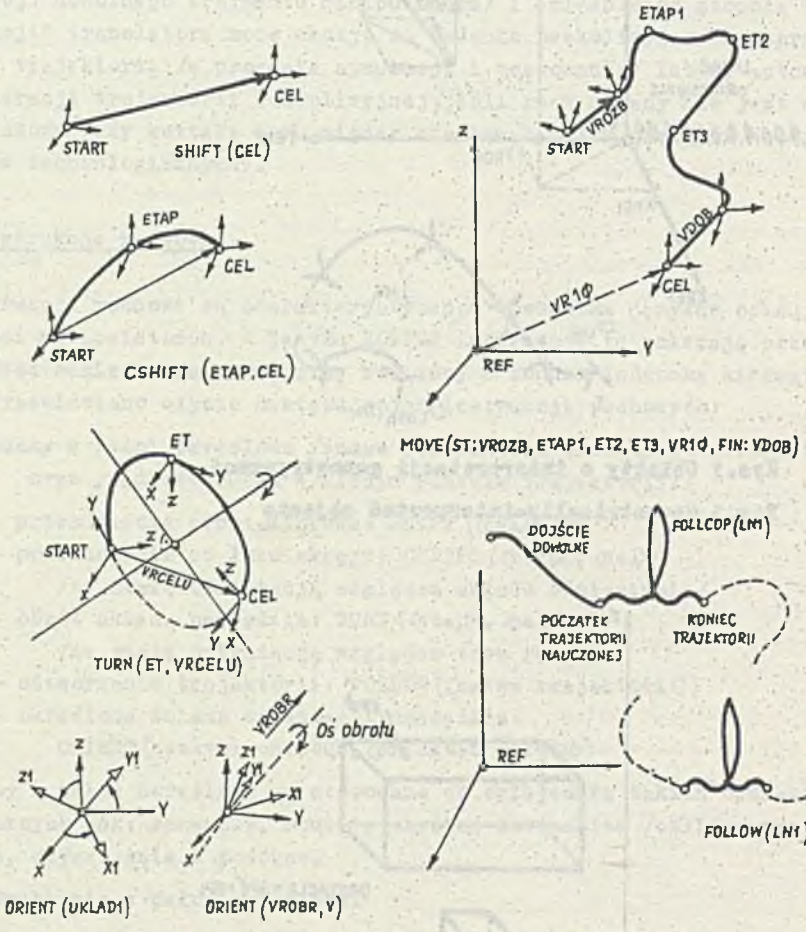
Rys.1 Obiekty o interpretacji geometrycznej

Fig.1 Geometrically-interpreted objects



Rys.2 Przykładowa przeszkoda

Fig.2 Example of obstacle



Rys.3 Przykłady instrukcji ruchowych

Fig.3 Examples of the motion-describing statements

trukcji transportera /CONV...FIN/ pozwalającej opisać ruch złożony powstający podczas wykonywania zadanej sekwencji ruchów z jednoczesnym unoszeniem lokalnego układu odniesienia po trajektorii w pełni określonej.

Korzystanie z instrukcji transportera ułatwia opisanie wielu typowych zadań manipulacyjnych jak: wykonywanie operacji na detalach przemieszczanych na przenośnikach, programowanie ruchu elektrody spawalniczej itp.

W języku ROBIMP przewidziano także możliwość pisania programów pracy dla układów sterujących wieloma łańcuchami kinematycznymi /kilka manipulatorów lub manipulator wieloramienny/. Programy takie składają się z odpowiedniej liczby bloków-zadań wykonywanych równolegle. Nazwy zmiennych niestandardowych nie są lokalne w zadaniach i dzięki temu zmienne numeryczne i logiczne mogą być używane do synchronizacji wykonywania poszczególnych zadań.

3. Etapy programowania i realizacji zadania technologicznego

Na rys. 4 przedstawiono uproszczony schemat blokowy pprogramowania układu sterowania. Przewidziano następujące fazy pracy przy uruchamianiu nowego programu: Tekst źródłowy tworzony jest za pomocą edytora komunikującego się podczas pracy z analizatorem składni wchodzącym w skład kompilatora, co pozwala na bieżąco usuwać błędy syntaktyczne. Gotowy tekst programu jest uzupełniany o wartości początkowe zmiennych geometrycznych poprzez wprowadzenie ich z konsoli operatorskiej lub odczytanie z układu pomiaru położenia manipulatora ustawionego w odpowiedniej pozycji. Następnie po kompilacji przeprowadza się symulację działania programu bez uruchamiania manipulatora, w celu sprawdzenia poprawności pracy z uwzględnieniem nałożonych ograniczeń /przyspieszenia, prędkości, przeszkód, zakresu przestrzeni roboczej itp./. W następnym etapie testowania programu należy sprawdzić poprawność jego działania podczas pracy krokowej przy uruchomionym manipulatorze w warunkach rzeczywistych obciążeń dynamicznych. Z uwagi na bezpieczeństwo pracy, wykonanie symulacji i testowania powinno być dla nowo zredagowanego programu warunkiem umożliwiającym jego pracę w cyklu automatycznym

4. Wyciąg ze składni języka ROBIMP

Program-ROBIMP = nagłówek-programu {deklaracje-zmiennych} segment-programu {segment-programu} "END" "<crl>".

nagłówek-programu = "PROGRAM" nazwa-programu.

segment-programu = nagłówek-segmentu {deklaracje-zmiennych} ciąg-linii
"ENDS" nazwa-manipulatora "<crlf>".

nagłówek-segmentu = "ROBOT" nazwa-manipulatora.

deklaracje-zmiennych = {nazwa-typu lista-nazw-zmiennych}.

nazwa-typu = "REAL"|"POINT"|"VECTOR"|"FRAME"|"ROTATION"|"TRANSFORMATION"|
"LINE"|"HALFSPACE"|"SPACE"|"BIT"|"BYTE"|"WORD".

nazwa-zmiennej = litera litera cyfra

linia-programu = [etykieta] instrukcja "crlf".

instrukcja = instrukcja-ruchowa|instr-przekazu-informacji|instr-organizac|
instr-podstawienia|instr-bierna|instr-pusta|
procedura-standardowa.

instrukcja-ruchowa = instr-SHIFT|instr-CSHIFT|instr-TURN|instr-ORIENT|
instr-FOLLOW|instr-FOLLCOP|instr-MOVE|instr-WAIT

instr-organizacyjna = instr-IF|instr-REPEAT|instr-WHILE|instr-FOR|
instr-GOSUB|instr-RET|instr-transportera.

instr-transportera = "CONV" prędkość-unoszenia "<crlf>" tor-unoszenia "<crlf>"
ciąg-instrukcji "PIN" "<crlf>".

instr-podstawienia = podstawienie-arytmetyczne|podstawienie-geometryczne|
podstawienie-logiczne.

funkcja-standardowa = SIN|COS|TAN|ATN|ABS|SQR|INT|SGN|ROP|obliczanie-
długości-wektora|tworzenie-obrotu|wyznaczanie-
obrotu|wyznaczanie-osi-obrotu|oblic zanie-kąta-
obrotu|tworzenie-transformacji|wyznaczanie-transform
obliczanie-transformacji-odwrotnej.

zmienna-standardowa = "SPEED"|"SPLIM"|" ACLIM"|"ACCUR"|" TIME"|"TPOS"|
"SENS" | "SPCONV".

LITERATURA

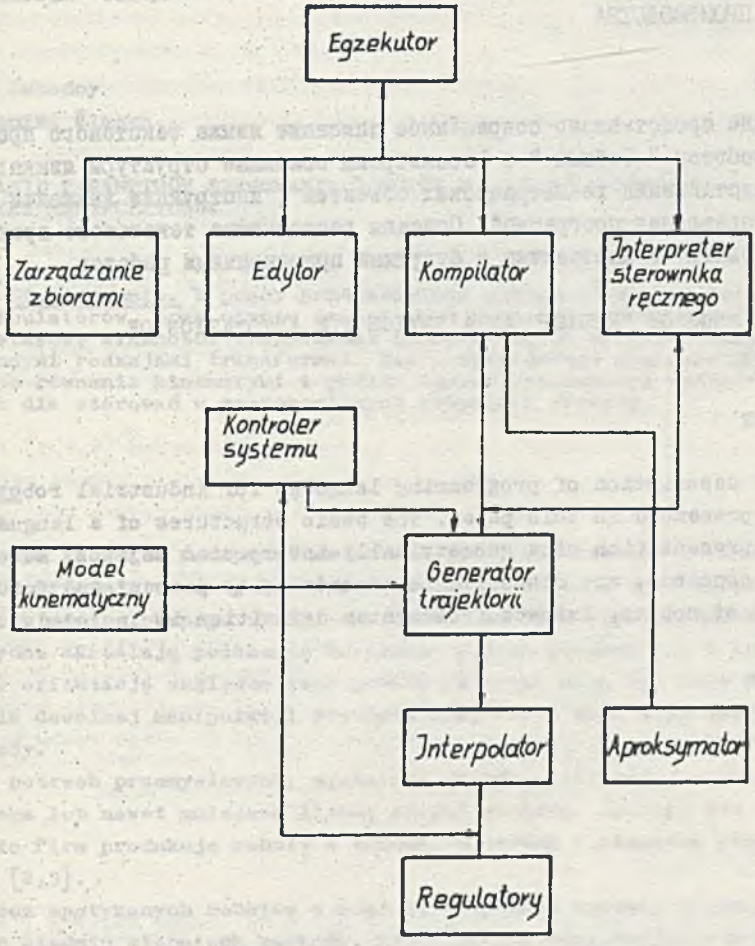
[1] Denavit J. Hartenberg R.S. : A Kinematic Notation for Lower-Pair Mechanisms Based on Matrices , ASME Journal of Applied Mechanics , June 1955.

[2] Rogowski Z. Królik K. : Założenia na język programowania tekstowego robotów , IMP 1983 /z późniejszymi zmianami i uzupełnieniami/.

[3] Blume C. Jakob W. : Programmiersprachen für Industrieroboter , Vogel-Buchverlag , Würzburg 1983.

Recenzent: Prof.dr h.inż.Antoni Woźniak

Wpłynęło do Redakcji do 1986.04.30



Rys.4 Uproszczony schemat blokowy oprogramowania układu sterowania

Fig.4 Simplified block diagram of the control system software

ЯЗЫК ПРОГРАММИРОВАНИЯ РОБОТОВ " РОБИМП " КАК СРЕДСТВО АВТОМАТИЗИ РОВАНИЯ ПРОИЗВОДСТВА

Р е з ю м е

В докладе предотавлено сокращённое описанке языка текстового программирования роботов " Робимп " . Рассмотрены основные структуры языка: род и способ представления геометрических объектов , инструкции движения манипулятора и управления программой. Описаны достоинства текстового программирования в связи с настоящими и будущими применениями роботов.

PROGRAMMING LANGUAGE "ROBIMP" AS A DEVICE FOR AUTOMATION OF MANUFACTURING

S u m m a r y

A consise description of programming language for industrial robots "ROBIMP" is presented in this paper. The basic structures of a language: types and representation of a geometrically-interpreted objects, motoric and control commands, are discussed in accordance to present and future applications of robots, Extract from syntax definition is included.