roboty przemysłowe, algorytm programu sterującego, sztuczna inteligencja
industrial robots, algorithm of the control program, artificial intelligence

Piotr ZYZAK[1]

WYKORZYSTANIE ROBOTA PRZEMYSŁOWEGO DO ROZWIĄZYWANIA KLASYCZNEGO ZADANIA SZTUCZNEJ INTELIGENCJI – WIEŻ HANOI

Programowanie robota przemysłowego jest jednym z najtrudniejszych w realizacji zadań w zautomatyzowanych systemach wytwarzania. Jest ono szczególnie trudne z logicznego punktu widzenia, gdyż jest zdeterminowane wymaganiami procesu technologicznego. Składa się z dwóch zadań: podstawowego jakim jest programowanie ruchu i pomocniczego obejmującego realizację warunków synchronizacji robota z procesem technologicznym. W artykule przedstawiono problematykę programowania robota przemysłowego do rozwiązywania klasycznego zadania sztucznej inteligencji – Wież Hanoi. Zamieszczono algorytmy oraz kody programów wykonawczych, do realizacji określonych zadań przez robota. Algorytmy opisują trajektorię ruchu robota, która jest „sumą" następujących po sobie, uporządkowanych kroków elementarnych, po zrealizowaniu których robot osiąga zaplanowany stan. Stany osiągane są drogą realizacji przyjętego sposobu działania.

USE OF AN INDUSTRIAL ROBOT TO SOLVING THE CLASSICAL PROBLEM OF ARTIFICIAL INTELLIGENCE – TOWER OF HANOI

Programming of an industrial robot belongs to one from the most difficult tasks in automated systems of manufacturing. This task is especially difficult from a logical point of view, because is determined by requirements of the technological process. It consists of two tasks: the main one as programming of the motion is, and the auxiliary one - comprising accomplishment of synchronization conditions of the robot with the technological process. In the paper is presented the issue of programming of the industrial robot to solving classical problem of artificial intelligence – Tower of Hanoi. It have been presented algorithms and machine codes of the task programs to accomplishment of determined operations of the robot. The algorithms describe trajectory of motion of the robot, which is a „sum" of successive, ordered elementary steps, after completion of which the robot reaches its planned state. The states are attained through accomplishment of adopted method of the proceeding.

1. WSTĘP

Początków sztucznej inteligencji można się doszukiwać w odległych wiekach, nawet u starożytnych filozofów, w szczególności jeśli rozważamy właśnie filozoficzne aspekty tej dziedziny nauki. Lata 60. i 70. ubiegłego wieku charakteryzują się całkowitą dominacją tzw. podejścia symbolicznego do rozwiązywania różnych zagadnień sztucznej inteligencji. Tak więc stosowano metody indukcji drzew decyzyjnych, logiki predykatów i w pewnym stopniu klasyczne metody probabilistyczne, które jednak nabrały większego znaczenia w późniejszym okresie, z chwilą rozwoju sieci bayesowskich. Rozważając zagadnienia sztucznej inteligencji, powinniśmy mieć pewien punkt odniesienia. Takim punktem odniesienia może być definicja ludzkiej inteligencji, bardzo często określana jako umiejętność przystosowywania się do nowych zadań i warunków życia albo sposobem, w jaki człowiek przetwarza informacje i rozwiązuje problemy. Inteligencja to również umiejętność kojarzenia oraz rozumienia. Stworzone przez człowieka tzw. inteligentne maszyny można zaprogramować tak, aby jedynie w wąskim zakresie imitowały elementy składające się na ludzką inteligencję, takie jak: zapamiętywanie, stawianie i realizacja celów, umiejętność współpracy, formułowanie wniosków, zdolność analizy, tworzenie oraz myślenie koncepcyjne i abstrakcyjne. Przewidywanie i prognozowanie wyników oraz planowanie to także domeny sztucznej inteligencji.

W literaturze przedstawiono rozmaite definicje sztucznej inteligencji:

- Sztuczna inteligencja jest nauką o maszynach realizujących zadania, które wymagają inteligencji, gdy są wykonywane przez człowieka.
- Sztuczna inteligencja stanowi dziedzinę informatyki dotyczącą metod i technik wnioskowania symbolicznego przez komputer oraz symbolicznej reprezentacji wiedzy stosowanej podczas takiego wnioskowania.
- Sztuczna inteligencja obejmuje rozwiązywanie problemów sposobami wzorowanymi na naturalnych działaniach i procesach poznawczych człowieka za pomocą symulujących je programów komputerowych [5].

[1] Akademia Techniczno-Humanistyczna w Bielsku-Białej, 43-309 Bielsko-Biała, ul. Willowa 2

Roboty znajdują obecnie coraz szersze zastosowanie. Wykorzystywane są w najróżniejszych dziedzinach przemysłu, między innymi: w metalowym, samochodowym, ciężkim, stoczniowym, budowy maszyn, a także w spożywczym i chemicznym. Na tak dużą różnorodność zastosowań pozwala duża elastyczność robota oraz jego oprogramowanie. Działanie robota może być kontrolowane przez człowieka, przez wprowadzony wcześniej program bądź przez zbiór ogólnych reguł, które zostają przełożone na działanie robota za pomocą technik sztucznej inteligencji. Roboty często zastępują człowieka przy monotonnych, złożonych z powtarzających się kroków czynnościach, które mogą wykonywać znacznie szybciej od ludzi [6].

Celem poniższej pracy jest przedstawienie metodologii programowania robotów przemysłowych do rozwiązywania klasycznego zadania sztucznej inteligencji – Wież Hanoi. W zadaniu tym istotnym parametrem implikującym sposób jego rozwiązania jest liczba krążków znajdujących się na stosie. Liczba krążków decyduje o sekwencji kroków elementarnych jakie musi wykonać robot przemysłowy oraz konieczności przeprowadzenia niezbędnych obliczeń.

Podstawowy algorytm programowania robota realizującego rozwiązanie zadania sztucznej inteligencji w ściśle zdefiniowanej przestrzeni (układzie współrzędnych) można przedstawić za pomocą kolejnych kroków:

- zdefiniuj zadanie podając położenie docelowe robota Pk (bieżącym położeniem początkowym jest P1),
- określ zbiór możliwych do zrealizowania stanów robota w układzie współrzędnych,
- ustal sposób działania, czyli poszukiwania kolejnych miejsc w przestrzeni pomiarowej,
- parametryzuj określone wielkości potrzebne do realizacji zadania przypisując je do zmiennych,
- zapamiętuj wartości wyznaczonych wielkości,
- wyznaczaj kolejno docelowe pozycje, bazuj na ich współrzędnych.

Należy przyjąć, że zadanie planowania trajektorii robota będzie realizowane w układzie współrzędnych, w którym poszczególne obiekty tworzące Wieżę Hanoi nie zmieniają swojego położenia. Zatem, potencjalna liczba wszystkich stanów jakie robot może osiągnąć i kroków jakie może zrealizować można uznać za skończoną.

Krok elementarny trajektorii robota będzie stanowił jego przemieszczenie wynikające ze zmiany współrzędnych ruchu wyznaczonych w przestrzeni zewnętrznej robota (przestrzeń zadania robota). Krok elementarny wyrażony jest w układzie współrzędnych zewnętrznych robota miejscem docelowym wynikającym, które określić można jako pozycję wspomagająca.

Akcją robota czyli jego elementarne działanie, określające sposób zmiany parametrów geometrycznych ustalonych dla jednego kroku elementarnego jest zdefiniowana pewną funkcją opisującą sposób przejścia tak, by osiągnąć kolejny stan konfiguracji przestrzennej na drodze ruchu członu robota.

Robot przemysłowy MOTOMAN UP6 jest przegubowym robotem sześcioosiowym o typowej kinematyce typu PUMA. Charakteryzuje się udźwigiem 6 kg, mocą napędów 2 kVA, stosunkowo dużą dokładnością (powtarzalność pozycjonowania ok. ±0,08 mm) i szerokim zakresem prędkości ruchów członów (do 9000 cm/min). Napęd robota przemysłowego MOTOMAN UP6 stanowi zespół sześciu silników prądu zmiennego 3-fazowego, wyposażonych w absolutne przetworniki obrotowo-impulsowe (układ pomiarowy), sterowanych przez serwonapędy (przemienniki częstotliwości). Stąd nie jest wymagana każdorazowo po włączeniu zasilania procedura kalibracji (bazowania) robota i od razu jest on gotowy do pracy [4].

2. METODOLOGIA

Wieża Hanoi jest grą logiczną znaną od XIX wieku. Do dyspozycji mamy trzy stosy A, B, C i pewną ilość krążków o różnej średnicy zewnętrznej. Krążki poukładane są na stosie A w kolejności od największej do najmniejszej średnicy przy czym krążek o największej średnicy znajduje się na dole. Zadaniem gracza jest przeniesienie wszystkich krążków ze stosu A na stos C, z wykorzystaniem stosu B. W pojedynczym ruchu można przenosić tylko jeden krążek między dwoma stosami i nigdy krążek o większej średnicy nie może znaleźć się na krążku o mniejszej średnicy.

Algorytm rozwiązywania Wież Hanoi jest klasycznym przykładem algorytmu rekurencyjnego. Charakterystyczną cechą funkcji (procedury) rekurencyjnej jest to, że wywołuje ona samą siebie. Drugą cechą rekursji jest jej dziedzina, którą mogą być tylko liczby naturalne [8]. W algorytmie rekurencyjnym złożony problem jest rozkładany na mniejszy, który potrafimy rozwiązać, a zakończenie algorytmu jest jasno określone. Wieże Hanoi są zadaniem, w którym złożoność obliczeniowa wzrasta bardzo szybko w miarę zwiększania liczby krążków. Najmniejszą ilość ruchów potrzebnych do wykonania zadania oblicza się korzystając ze wzoru [8]:

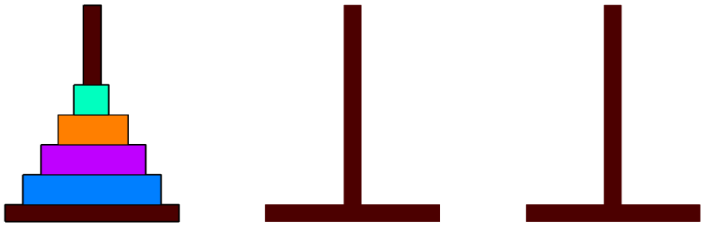
$$h_n = 2^n - 1 \tag{1}$$

gdzie:

h_n – najmniejsza liczba ruchów potrzebna do ułożenia Wież Hanoi,

n – liczba krążków.

W układaniu Wież Hanoi ważne jest czy mamy do czynienia z parzystą bądź nieparzystą liczbą krążków. Jeśli liczba krążków jest parzysta pierwszym stosem na który odkładamy najmniejszy krążek jest stos 2 (rys. 1) jeśli nieparzystą najpierw odkładamy krążek na stos 3.



Rys. 1. *Wieże Hanoi – numeracja stosów [1]*

Do wygenerowania sekwencji kroków - z którego stosu robot ma zabrać krążek i na który ma go odstawić wykorzystano program HANOI.PRO napisany w języku PDC PROLOG. Program „Hanoi.pro" składa się z trzech sekcji: deklaracji predykatów (Predicates), deklaracji klauzul (Clauses) oraz określeniu celu (Goal), gdzie następuje otwarcie i zamknięcie pliku wyjściowego zrzut i deklaracja liczby krążków na stosie początkowym. Program sformułowany w języku Prolog jest prosty, dzięki wykorzystaniu pętli rekurencyjnej. W programie zadeklarowano dwa predykaty move(i,i,i,i), realizujący rekurencyjny algorytm generowania kolejnych ruchów, oraz write(i) zapisujących kolejne ruchy na pliku „zrzut". Plik wyjściowy został tak zredagowany, aby mógł być bezpośrednio wczytany do systemu sterowania robota MOTOMAN UP6. Poniżej zamieszczono listing programu „Hanoi.pro" [1].

```
Predicates

        move(Integer,String,String,String)
        writeN(Integer)

Clauses

        move(1,X,Y,_):-

                n(L), retractall(n(_)),
                L1=L+1,assert(n(L1)),
                writeN(L1),
                write(X),
                write(Y),nl,
                !.

        move(N,X,Y,Z):-

                M=N-1,
                move(M,X,Z,Y),
                move(1,X,Y,Z),
                move(M,Z,Y,X).

        writeN(N):-N>99, writef("0%3.0 B%3.0=",N,N),!.
        writeN(N):-N>9,  writef("00%2.0 B0%2.0=",N,N),!.
        writeN(N):-      writef("000%1.0 B00%1.0=",N,N),!.

Goal

        openwrite(plik,"zrzut"),
        writedevice(plik),
        assert(n(0)),
        X="1",
        Y="2",
        Z="3",
        write("0000 NOP"),nl,
        move(6,X,Y,Z),
        n(L), L1=L+1, writef("00%2.0",L1), write(" END"),
        closefile(plik).
```

Wygenerowane sekwencje ruchów przedstawiono w tabeli 1.

Tab. 1. Wygenerowane sekwencje ruchów przy układaniu Wież Hanoi przez robota przemysłowego

L.p.	2 krążki	3 krążki	4 krążki	5 krążków	6 krążków
1.				z 1 na 2	z 1 na 3
2.				z 1 na 3	z 1 na 2
3.				z 2 na 3	z 3 na 2
4.				z 1 na 2	z 1 na 3
5.				z 3 na 1	z 2 na 1
6.				z 3 na 2	z 2 na 3
7.				z 1 na 2	z 1 na 3
8.				z 1 na 3	z 1 na 2
9.				z 2 na 3	z 3 na 2
10.				z 2 na 1	z 3 na 1
11.				z 3 na 1	z 2 na 1
12.				z 2 na 3	z 3 na 2
13.				z 1 na 2	z 1 na 3
14.				z 1 na 3	z 1 na 2
15.				z 2 na 3	z 3 na 2
16.				z 1 na 2	z 1 na 3
17.				z 3 na 1	z 2 na 1
18.				z 3 na 2	z 2 na 3
19.				z 1 na 2	z 1 na 3
20.				z 3 na 1	z 2 na 1
21.				z 2 na 3	z 3 na 2
22.				z 2 na 1	z 3 na 1
23.				z 3 na 1	z 2 na 1
24.				z 3 na 2	z 2 na 3
25.		z 1 na 2		z 1 na 2	z 1 na 3
26.		z 1 na 3		z 1 na 3	z 1 na 2
27.		z 2 na 3		z 2 na 3	z 3 na 2
28.		z 1 na 2		z 1 na 2	z 1 na 3
29.		z 3 na 1		z 3 na 1	z 2 na 1
30.		z 3 na 2		z 3 na 2	z 2 na 3
31.		z 1 na 2		z 1 na 2	z 1 na 3
32.					z 1 na 2
33.					z 3 na 2
34.					z 3 na 1
35.					z 2 na 1
36.					z 3 na 2
37.					z 1 na 3
38.					z 1 na 2
39.					z 3 na 2
40.					z 3 na 1
41.					z 2 na 1
42.					z 2 na 3
43.					z 1 na 3
44.					z 2 na 1
45.					z 3 na 2
46.					z 3 na 1
47.					z 2 na 1

48.					z 3 na 2
49.			z 1 na 3		z 1 na 3
50.			z 1 na 2		z 1 na 2
51.			z 3 na 2		z 3 na 2
52.			z 1 na 3		z 1 na 3
53.			z 2 na 1		z 2 na 1
54.			z 2 na 3		z 2 na 3
55.			z 1 na 3		z 1 na 3
56.			z 1 na 2		z 1 na 2
57.			z 3 na 2		z 3 na 2
58.			z 3 na 1		z 3 na 1
59.			z 2 na 1		z 2 na 1
60.			z 3 na 2		z 3 na 2
61.	z 1 na 3		z 1 na 3		z 1 na 3
62.	z 1 na 2		z 1 na 2		z 1 na 2
63.	z 3 na 2		z 3 na 2		z 3 na 2

Po przeanalizowaniu danych zawartych w tabeli 1 stwierdzono, że sekwencje ruchów dla dwóch krążków są takie same jak trzy ostatnie sekwencje dla czterech krążków, a te z kolei są takie same jak 15 ostatnich sekwencji dla sześciu krążków. Podobnie wygląda sytuacja dla trzech i pięciu krążków. Podczas pisania programu „Hanoi" wykorzystano tę zależność i zamiast pisania osobnych sekwencji dla dwóch, trzech, czterech, pięciu czy sześciu krążków, napisano tylko dla pięciu i sześciu, a następnie w programie za pomocą zmiennej I010 oraz B000 rozpoczynano wykonywanie sekwencji od określonego miejsca na liście sekwencji dla pięciu lub sześciu krążków.

3. CZĘŚĆ DOŚWIADCZALNA

Do rozwiązywania klasycznego zadania sztucznej inteligencji – Wież Hanoi przez robota przemysłowego Yaskawa MOTOMAN UP6 opracowano program w języku programowania INFORM II [7]. Tak jak w każdym innym języku programowania również INFORM II pozwala używać w programie sterującym zmiennych, traktowanych jako parametry wykonywanych funkcji. Mogą one być modyfikowane w programie poprzez wykonywanie wszelkiego rodzaju instrukcji arytmetyczno – logicznych. W opracowanych programach: głównym, wykonawczym oraz kalibrującym korzystając z dopuszczalnych zakresów adresów zastosowano kilka typów zmiennych (bajt – 8 bitowe, całkowite – integer, podwójny bajt oraz pozycyjne), których oznaczenia kodowe zawierają listingi programów zestawione w tabelach 2÷6. Do opracowania programu „Hanoi" wykorzystano zaimplementowane w układzie sterowania XRC robota przemysłowego MOTOMAN UP6 instrukcje zawarte w następujących grupach funkcji:

- Motion – grupa funkcji związana z obsługą różnych rodzajów interpolacji,
- In/Out – grupa funkcji związana z przetwarzaniem sygnałów dyskretnych (binarnych) oraz analogowych,
- Control – grupa funkcji obejmująca instrukcje skoków bezwarunkowych, pętli, wywołania podprogramów,
- Arith – arytmetyczne związane z wykonywaniem operacji na zmiennych liczbowych i bajtowych.

W tabelach 2÷6 zamieszczono listing (wraz z komentarzami) programu do układania Wież Hanoi przez robota przemysłowego MOTOMAN UP6.

Program ten składa się z:

– Programu głównego – „Hanoi" (tabela 2),
– Programu wykonawczego – „Ruch1" – zawiera instrukcje ruchu robota podczas poszczególnych operacji przenoszenia krążków (tabela 3),
– Sekwencji ruchów dla 5 krążków – „5DISK" (tabela 4),
– Sekwencji ruchów dla 6 krążków – „6DISK" (tabela 5),
– Programu kalibrującego – „USTAW_P123" (tabela 6) – używany przed pierwszym uruchomieniem programu „Hanoi". Podczas jego wykonywania pod odpowiednimi zmiennymi sterownik zapisuje współrzędne punktów na których będą składane stosy z krążkami. Współrzędne punktów zapisywane są w zmiennych pozycyjnych.

W opracowanym programie dla robota przemysłowego Yaskawa MOTOMAN UP6 przewidziano układanie Wież Hanoi dla dwóch, trzech, czterech, pięciu i sześciu krążków. Zrealizowane zadanie dotyczące rozwiązywania zagadnienia Wież Hanoi przez robota przemysłowego ma przedstawić możliwości pisania złożonych programów wykorzystujących odwołania i podprogramy realizowanych przez sterownik XRC, oraz w efekcie zmian parametrów zapisanych w deklarowanych zmiennych ocenić efektywność rozwiązywania klasycznego zadania sztucznej inteligencji Wieże Hanoi rozumianą jako czas rozwiązywania zadania.

Przed wykonywaniem programu „Hanoi" do zmiennej całkowitej I000 należy przypisać liczbę krążków na stosie 1, natomiast do zmiennych całkowitych I002 oraz I003 liczbę krążków na stosach 2 i 3. Jako wartość zmiennej całkowitej I050 należy zadeklarować prędkość, z jaką mają być wykonywane kroki elementarne robota.

Tab. 2. Listing głównego programu "Hanoi"

Nr linii	Instrukcja	Komentarz
0000	NOP	Początek programu
0001	MOVL P002 V=150	Przesunięcie końcówki wykonawczej na pozycje drugiego stosu
0002	SET I001 I000	I000 – liczba krążków jaka będzie użyta w programie. Wartość zmiennej I000 wpisujemy w tablicy zmiennych. I001 – liczba krążków na stosie 1 Ustaw liczbę krążków na stosie 1 na taką samą jak liczba krążków z której będziemy układać wieżę.
0003	SET I002 0	I002 – liczba krążków na stosie 2
0004	SET I003 0	I003 – liczba krążków na stosie 3
0005	JUMP *2 IF I000=2	Przejdź do etykiety 2
0006	JUMP *3 IF I000=3	Przejdź do etykiety 3
0007	JUMP *4 IF I000=4	Przejdź do etykiety 4
0008	JUMP *5 IF I000=5	Przejdź do etykiety 5
0009	JUMP *6 IF I000=6	Przejdź do etykiety 6
0010	JUMP *KONIEC	Przejdź do etykiety KONIEC
0011	*2	Etykieta 2
0012	CALL JOB:6DISK	Wywołanie podprogramu 6DISK. Wczytanie do pamięci sterownika wartości zmiennych bajtowych B (odpowiedniej sekwencji układania wieży Hanoi)
0013	SET I010 60	I010 – numer aktualnej sekwencji. Ustawia aktualną sekwencję na 60 (por. Tabela 1).
0014	SET I004 63	I004 – numer sekwencji po wykonaniu, której robot ma zakończyć działanie.
0015	JUMP *LOOP	Przejdź do etykiety LOOP (zapętlenie programu)
0016	*3	Etykieta 3
0017	CALL JOB:5DISK	Wywołanie podprogramu 5DISK
0018	SET I010 24	Ustawia aktualną sekwencję na 24 (por. tab.1).
0019	SET I004 31	
0020	JUMP *LOOP	
0021	*4	Etykieta 4
0022	CALL JOB:6DISK	
0023	SET I010 48	Ustawia aktualną sekwencję na 4 (por. Tabela 1).
0024	SET I004 63	
0025	JUMP *LOOP	
0026	*5	Etykieta 5
0027	CALL JOB:5DISK	
0028	SET I010 0	Ustawia aktualną sekwencję na 31 (por. Tabela 1).
0029	SET I004 31	

0030	JUMP *LOOP	
0031	*6	Etykieta 6
0032	CALL JOB: 6DISK	
0033	SET I010 0	Ustawia aktualną sekwencję na 0 (por. Tabela 1).
0034	SET I004 63	
0035	*LOOP	
0036	JUMP *KONIEC IF I010=I004	Jeśli numer aktualnej sekwencji równy jest numerowi sekwencji ostatniej program kończy działanie w przeciwnym razie numer aktualnej sekwencji zwiększany jest o 1.
0037	SET I010 I010+1	
0038	SET B000 B[I010]	
0039	CALL JOB:RUCH1	Wywołanie podprogramu RUCH1
0040	JUMP *LOOP	Przejdź do etykiety LOOP
0041	*KONIEC	Etykieta KONIEC
0042	END	Koniec programu

Tab. 3. Listing programu wykonawczego "Ruch1"

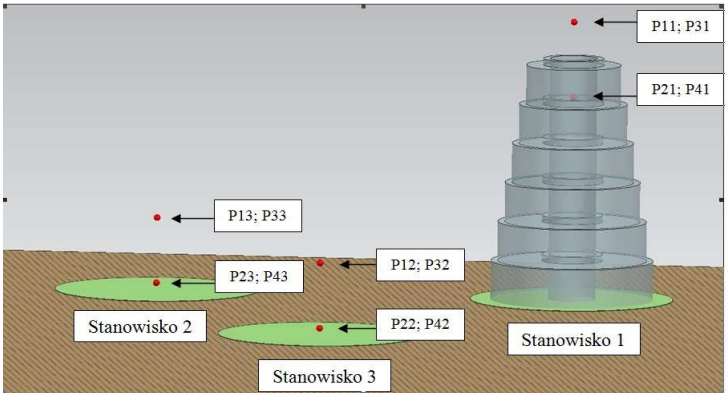
Nr linii	Instrukcja	Komentarz
0000	NOP	Początek programu
0001	DOUT OT#(4) ON	Zamknięcie szczęk chwytaka
0002	JUMP *R_3 IF B000>30	Przejdź do etykiety R_3 jeśli zmienna B000 jest większa od 30
0003	JUMP *R_2 IF B000>20	Przejdź do etykiety R_2 jeśli zmienna B000 jest większa od 20
0004	'R_1	Fragment programu odpowiedzialny za odbieranie detali ze stosu 1
0005	SET P011 P001	Ustaw współrzędne punktu P011 na takie same jak współrzędne punktu P001
0006	SET P021 P001	Ustaw współrzędne punktu P021 na takie same jak współrzędne punktu P001
0007	GETE D001 P001 (3)	Ustaw zmienną D001 na taką samą jaka jest trzecia współrzędna (Z – towa) punktu P001
0008	SET D011 D001+(I001-1)*30000+50000	Obliczanie wysokości stosu 1 i dodanie do niej 50000µm (5cm). Wynik zapisujemy do zmiennej D011.
0009	SET D021 D001+(I001-1)*30000	Obliczanie wysokości stosu 1 i zapisanie jej do zmiennej D021
0010	SETE P011 (3) D011	Ustawienie trzeciej współrzędnej (Z-towej) punktu P011 na taką samą jak wartość zmiennej D011
0011	SETE P021 (3) D021	Ustawienie trzeciej współrzędnej (Z-towej) punktu P021 na taką samą jak wartość zmiennej D021
0012	MOVL P011 V=I050 PL=0 DEC=50	I050 – zmienna odpowiedzialna za prędkość wykonywanych ruchów. Przejazd z interpolacją liniową z opóźnieniem DEC=50 i parametrem PositionLevel=0 do punktu P011
0013	MOVL P021 V=I050 PL=0 DEC=50	Przejazd z interpolacją liniową z opóźnieniem DEC=50 i parametrem PositionLevel=0 do punktu P021
0014	DOUT OT#(4) OFF	Otwarcie szczęk chwytaka
0015	MOVL P011 V=I050 PL=0	Odjazd z interpolacją liniową i parametrem PositionLevel = 0 do punktu P011. W tym momencie w szczękach chwytaka znajduje się przenoszony krążek
0016	SET I001 I001-1	Zmniejszenie o1 liczby krążków na stosie 1
0017	SET B000 B000-10	Zmniejszenie wartości zmiennej bajtowej B000 o 10
0018	JUMP *CEL	Przejście do etykiety CEL
0019	*R_2	Etykieta R_2. Fragment programu odpowiedzialny za odbieranie detali ze stosu 2

0020	SET P012 P002	Ustaw współrzędne punktu P012 na takie same jak współrzędne punktu P002
0021	SET P022 P002	Ustaw współrzędne punktu P022 na takie same jak współrzędne punktu P002
0022	GETE D002 P002 (3)	Ustaw zmienną D002 na taką samą jaka jest trzecia współrzędna (Z – towa) punktu P002
0023	SET D012 D002+(I002-1)*30000+50000	Obliczanie wysokości stosu 2 i dodanie do niej 50000µm (5cm). Wynik zapisujemy do zmiennej D012.
0024	SET D022 D002+(I002-1)*30000	Obliczanie wysokości stosu 1 i zapisanie jej do zmiennej D022
0025	SETE P012 (3) D012	Ustawienie trzeciej współrzędnej (Z-towej) punktu P012 na taką samą jak wartość zmiennej D012
0026	SETE P022 (3) D022	Ustawienie trzeciej współrzędnej (Z-towej) punktu P022 na taką samą jak wartość zmiennej D022
0027	MOVL P012 V=I050 PL=0	Przejazd z interpolacją liniową i parametrem PositionLevel=0 do punktu P012
0028	MOVL P022 V=I050 PL=0	Przejazd z interpolacją liniową i parametrem PositionLevel=0 do punktu P022
0029	DOUT OT#(4) OFF	Otwarcie szczęk chwytaka
0030	MOVL P012 V=I050 PL=0	Odjazd z interpolacją liniową i parametrem PositionLevel = 0 do punktu P012
0031	SET I002 I002-1	Zmniejszenie o 1 liczby krążków na stosie 2
0032	SET B000 B000-20	Zmniejszenie wartości zmiennej bajtowej B000 o 20
0033	JUMP *CEL	Przejście do etykiety CEL
0034	*R_3	Etykieta R_3. Fragment programu odpowiedzialny za odbieranie krążków ze stosu 3
0035	SET P013 P003	Ustaw współrzędne punktu P013 na takie same jak współrzędne punktu P003
0036	SET P032 P003	Ustaw współrzędne punktu P032 na takie same jak współrzędne punktu P003
0037	GETE D003 P003 (3)	Ustaw zmienną D003 na taką samą jaka jest trzecia współrzędna (Z – towa) punktu P003
0038	SET D013 D003+(I003-1)*30000+50000	Obliczanie wysokości stosu 3 i dodanie do niej 50000µm (5cm). Wynik zapisujemy do zmiennej D012.
0039	SET D032 D003+(I003-1)*30000	Obliczanie wysokości stosu 3 i zapisanie jej do zmiennej D032
0040	SETE P013 (3) D013	Ustawienie trzeciej współrzędnej (Z-towej) punktu P013 na taką samą jak wartość zmiennej D013
0041	SETE P032 (3) D032	Ustawienie trzeciej współrzędnej (Z-towej) punktu P032 na taką samą jak wartość zmiennej D032
0042	MOVL P013 V=I050 PL=0	Przejazd z interpolacją liniową i parametrem PositionLevel=0 do punktu P013
0043	MOVL P032 V=I050 PL=0	Przejazd z interpolacją liniową i parametrem PositionLevel=0 do punktu P032
0044	DOUT OT#(4) OFF	Otwarcie szczęk chwytaka
0045	MOVL P013 V=I050 PL=0	Odjazd z interpolacją liniową i parametrem PositionLevel = 0 do punktu P013
0046	SET I003 I003-1	Zmniejszenie o 1 liczby krążków na stosie 3
0047	SET B000 B000-30	Zmniejszenie wartości zmiennej bajtowej B000 o 30
0048	*CEL	Etykieta CEL. Fragment programu odpowiedzialny za odkładanie detali na stos 1
0049	JUMP *CEL2 IF B000=2	Przejście do etykiety CEL2 jeśli zmienna B000 = 2
0050	JUMP *CEL3 IF B000=3	Przejście do etykiety CEL3 jeśli zmienna B000 = 3
0051	SET P031 P001	Ustaw współrzędne punktu P031 na takie same jak współrzędne punktu P001
0052	SET P041 P001	Ustaw współrzędne punktu P041 na takie same jak współrzędne punktu P001
0053	GETE D001 P001 (3)	Ustaw zmienną D001 na taką samą jaka jest trzecia współrzędna (Z – towa) punktu P001

0054	SET D031 D001+I001*30000+50000	Obliczanie wysokości stosu 1 i dodanie do niej 50000µm (5cm). Wynik zapisujemy do zmiennej D031.
0055	SET D041 D001+I001*30000	Obliczanie wysokości stosu 1 i zapisanie jej do zmiennej D041
0056	SETE P031 (3) D031	Ustawienie trzeciej współrzędnej (Z-towej) punktu P031 na taką samą jak wartość zmiennej D031
0057	SETE P041 (3) D041	Ustawienie trzeciej współrzędnej (Z-towej) punktu P041 na taką samą jak wartość zmiennej D041
0058	MOVL P031 V=I050 PL=0	Przejazd z interpolacją liniową i parametrem PositionLevel=0 do punktu P031
0059	MOVL P041 V=I050 PL=0	Przejazd z interpolacją liniową i parametrem PositionLevel=0 do punktu P041
0060	DOUT OT#(4) ON	Zamknięcie szczęk chwytaka
0061	MOVL P031 V=I050 PL=0	Odjazd z interpolacją liniową i parametrem PositionLevel = 0 do punktu P013
0062	SET I001 I001+1	Zwiększenie o 1 liczby krążków na stosie 1
0063	JUMP *KONIEC	Przejście do etykiety KONIEC
0064	*CEL2	Etykieta CEL2. Fragment programu odpowiedzialny za odkładanie detali na stos 2
0065	SET P032 P002	Ustaw współrzędne punktu P032 na takie same jak współrzędne punktu P002
0066	SET P042 P002	Ustaw współrzędne punktu P042 na takie same jak współrzędne punktu P002
0067	GETE D002 P002 (3)	Ustaw zmienną D002 na taką samą jaka jest trzecia współrzędna (Z – towa) punktu P002
0068	SET D032 D002+I002*30000+50000	Obliczanie wysokości stosu 2 i dodanie do niej 50000µm (5cm). Wynik zapisujemy do zmiennej D032.
0069	SET D042 D002+I002*30000	Obliczanie wysokości stosu 2 i zapisanie jej do zmiennej D042
0070	SETE P032 (3) D032	Ustawienie trzeciej współrzędnej (Z-towej) punktu P032 na taką samą jak wartość zmiennej D032
0071	SETE P042 (3) D042	Ustawienie trzeciej współrzędnej (Z-towej) punktu P042 na taką samą jak wartość zmiennej D042
0072	MOVL P032 V=I050 PL=0	Przejazd z interpolacją liniową i parametrem PositionLevel=0 do punktu P032
0073	MOVL P042 V=I050 PL=0	Przejazd z interpolacją liniową i parametrem PositionLevel=0 do punktu P042
0074	DOUT OT#(4) ON	Zamknięcie szczęk chwytaka
0075	MOVL P032 V=I050 PL=0	Odjazd z interpolacją liniową i parametrem PositionLevel = 0 do punktu P032
0076	SET I002 I002+1	Zwiększenie o 1 liczby krążków na stosie 2
0077	JUMP *KONIEC	Przejście do etykiety KONIEC
0078	*CEL3	Etykieta CEL3. Fragment programu odpowiedzialny za odkładanie detali na stosie 3
0079	SET P033 P003	Ustaw współrzędne punktu P033 na takie same jak współrzędne punktu P003
0080	SET P043 P003	Ustaw współrzędne punktu P043 na takie same jak współrzędne punktu P003
0081	GETE D003 P003 (3)	Ustaw zmienną D003 na taką samą jaka jest trzecia współrzędna (Z – towa) punktu P003
0082	SET D033 D003+I003*30000+50000	Obliczanie wysokości stosu 2 i dodanie do niej 50000µm (5cm). Wynik zapisujemy do zmiennej D033.
0083	SET D043 D003+I003*30000	Obliczanie wysokości stosu 2 i zapisanie jej do zmiennej D043
0084	SETE P033 (3) D033	Ustawienie trzeciej współrzędnej (Z-towej) punktu P033 na taką samą jak wartość zmiennej D033
0085	SETE P043 (3) D043	Ustawienie trzeciej współrzędnej (Z-towej) punktu P043 na taką samą jak wartość zmiennej D043

0086	MOVL P033 V=I050 PL=0	Przejazd z interpolacją liniową i parametrem PositionLevel=0 do punktu P033
0087	MOVL P043 V=I050 PL=0	Przejazd z interpolacją liniową i parametrem PositionLevel=0 do punktu P043
0088	DOUT OT#(4) ON	Zamknięcie szczęk chwytaka
0089	MOVL P043 V=I050 PL=0	Odjazd z interpolacją liniową i parametrem PositionLevel = 0 do punktu P043
0090	SET I003 I003+1	Zwiększenie o 1 liczby krążków na stosie 3
0091	*KONIEC	Etykieta koniec
0092	END	Koniec programu

Na rysunku 2 przedstawiono położenie punktów wykorzystywanych w programie „Ruch1”. Położenie tych punktów zmienia się wraz ze zmianą liczby krążków na poszczególnych stanowiskach (stosach), na które są odkładane. Sytuacja przedstawiona na rysunku 2 dotyczy sześciu krążków znajdujących się na stanowisku 1 i braku krążków na pozostałych stanowiskach.



Rys. 2. *Oznaczenia punktów używanych w programie „Ruch1” [1]*

Tab. 4. Listing programu odpowiedzialnego za sekwencje ruchów dla pięciu krążków – „5DISK”

Nr linii	**Instrukcja**	**Komentarz**
0000	NOP	Początek programu
0001	SET B001 12	Zapisanie wartości 12 do zmiennej bajtowej B001. Pierwsze przełożenie krążka odbędzie ze stosu 1 na stos 2.
0002	SET B002 13	Zapisanie wartości 13 do zmiennej bajtowej B002. Przełożenie krążka odbędzie ze stosu 1 na stos 3 .
0003	SET B003 23	Zapisanie wartości 23 do zmiennej bajtowej B003. Przełożenie krążka odbędzie ze stosu 2 na stos 3.
0004	SET B004 12	Zapisanie wartości 12 do zmiennej bajtowej B004. Przełożenie krążka odbędzie ze stosu 1 na stos 2 .
0005	SET B005 31	
0006	SET B006 32	
0007	SET B007 12	
0008	SET B008 13	
0009	SET B009 23	
0010	SET B010 21	
0011	SET B011 31	
0012	SET B012 23	
0013	SET B013 12	
0014	SET B014 13	
0015	SET B015 23	

0016	SET B016 12	
0017	SET B017 31	
0018	SET B018 32	
0019	SET B019 12	
0020	SET B020 31	
0021	SET B021 23	
0022	SET B022 21	
0023	SET B023 31	
0024	SET B024 32	
0025	SET B025 12	
0026	SET B026 13	
0027	SET B027 23	
0028	SET B028 12	
0029	SET B029 31	
0030	SET B030 32	
0031	SET B031 12	
0032	END	Koniec programu

Tab. 5. Listing programu odpowiedzialnego za sekwencje ruchów dla sześciu krążków – „6DISK”

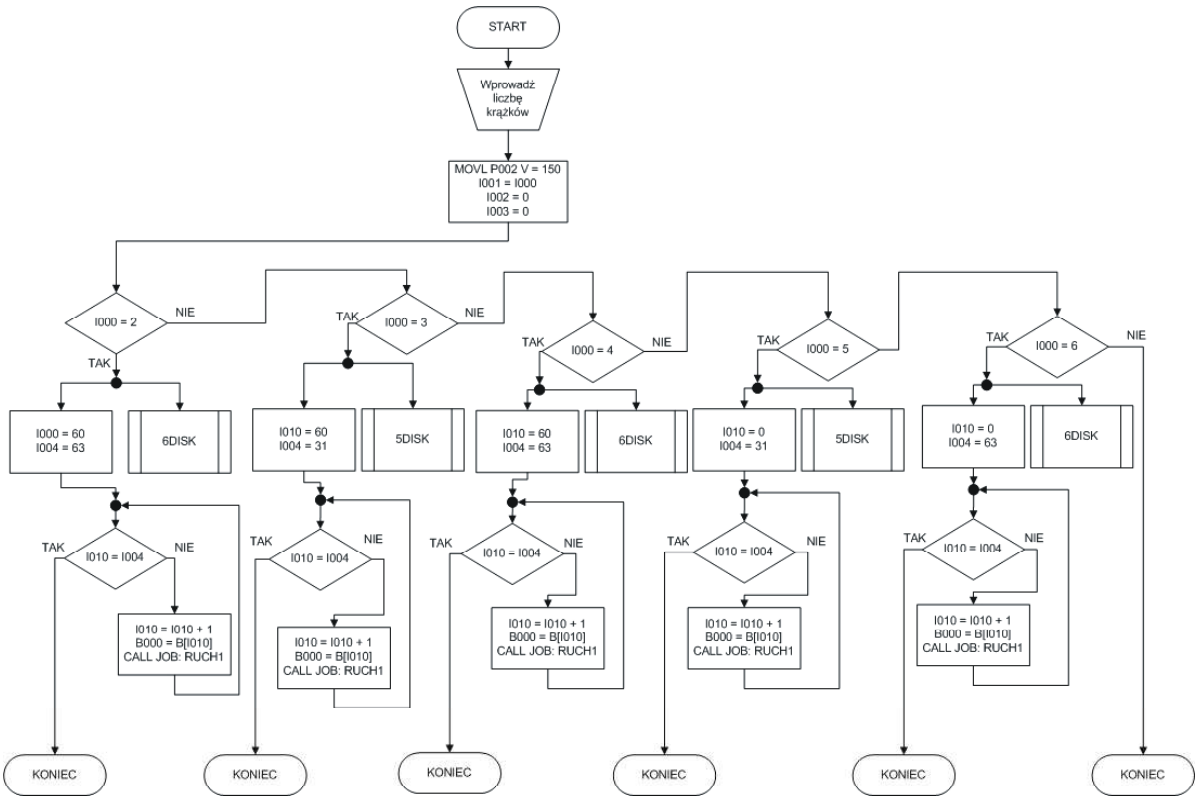
Nr linii	Instrukcja	Komentarz
0000	NOP	Początek programu
0001	SET B001 13	Zapisanie wartości 13 do zmiennej bajtowej B001. Pierwsze przełożenie krążka odbędzie ze stosu 1 na stos 3.
0002	SET B002 12	Zapisanie wartości 12 do zmiennej bajtowej B002. Przełożenie krążka odbędzie ze stosu 1 na stos 2.
0003	SET B003 32	Zapisanie wartości 32 do zmiennej bajtowej B003. Przełożenie krążka odbędzie ze stosu 3 na stos 2.
0004	SET B004 13	
0005	SET B005 21	
0006	SET B006 23	
0007	SET B007 13	
0008	SET B008 12	
0009	SET B009 32	
0010	SET B010 31	
0011	SET B011 21	
0012	SET B012 32	
0013	SET B013 13	
0014	SET B014 12	
0015	SET B015 32	
0016	SET B016 13	
0017	SET B017 21	
0018	SET B018 23	
0019	SET B019 13	
0020	SET B020 21	
0021	SET B021 32	
0022	SET B022 31	
0023	SET B023 21	

0024	SET B024 23	
0025	SET B025 13	
0026	SET B026 12	
0027	SET B027 32	
0028	SET B028 13	
0029	SET B029 21	
0030	SET B030 23	
0031	SET B031 13	
0032	SET B032 12	
0033	SET B033 32	
0034	SET B034 31	
0035	SET B035 21	
0036	SET B036 32	
0037	SET B037 13	
0038	SET B038 12	
0039	SET B039 32	
0040	SET B040 31	
0041	SET B041 21	
0042	SET B042 23	
0043	SET B043 13	
0044	SET B044 21	
0045	SET B045 32	
0046	SET B046 31	
0047	SET B047 21	
0048	SET B048 32	
0049	SET B049 13	
0050	SET B050 12	
0051	SET B051 32	
0052	SET B052 13	
0053	SET B053 21	
0054	SET B054 23	
0055	SET B055 13	
0056	SET B056 12	
0057	SET B057 32	
0058	SET B058 31	
0059	SET B059 21	
0060	SET B060 32	
0061	SET B061 13	
0062	SET B062 12	
0063	SET B063 32	
0064	END	Koniec programu

Tab. 6. Listing programu kalibrującego „USTAW_P123”

Nr linijki	Instrukcja	Komentarz
0000	NOP	Początek programu
0001	DOUT OT#(4) ON	Zamknięcie szczęk chwytaka
0002	MOVJ	Dojazd do pierwszego punktu
0003	GETS PX001 $PX001	Zapamiętanie współrzędnych pierwszego punktu
0004	MOVL	Dojazd z interpolacją liniową do drugiego punktu
0005	GETS PX002 $PX001	Zapamiętanie współrzędnych drugiego punktu
0006	MOVL	Dojazd z interpolacją liniową do trzeciego punktu
0007	GETS PX003 $PX001	Zapamiętanie współrzędnych trzeciego punktu
0008	MOVL P001 V=150	Przejazd z interpolacja liniową przez punkty 1, 2, 3
0009	MOVL P002 V=150	
0010	MOVL P003 V=150	
0011	END	Koniec programu

Dla czytelnego zobrazowania działania opracowanego programu realizującego rozwiązanie klasycznego zadania sztucznej inteligencji – Wież Hanoi przez robota przemysłowego MOTOMAN UP6 na rysunku 3 zamieszczono jego schemat blokowy.



Rys. 3. *Schemat blokowy algorytmu na układanie Wież Hanoi przez robota MOTOMAN UP6 [1]*

4. PODSUMOWANIE

Wykorzystując zaimplementowane instrukcje i funkcje języka programowania INFORM II opracowano program sterujący dla robota przemysłowego Yaskawa MOTOMAN UP6 do rozwiązywania klasycznego zadania sztucznej inteligencji – zagadnienia Wież Hanoi. Bloki opracowanego programu zawierające instrukcję

(funkcję) są wykonywane sekwencyjnie w kolejności ich zapisu w programie, chyba że została użyta funkcja strukturalna powodująca wykonanie skoku w programie czy do podprogramu.

Opracowany program umożliwia układanie Wież Hanoi dla dwóch, trzech, czterech, pięciu i sześciu krążków. Niewątpliwą zaletą tego programu jest jego uniwersalność, wyrażana tym, że użytkownik ma możliwość deklaracji przed uruchomieniem programu ilości krążków znajdujących się na poszczególnych stosach, jak również prędkości ruchu członów robota przy wykonywaniu kolejnych sekwencji kroków elementarnych. Programując robota przemysłowego do rozwiązywania opisanego zadania konieczne jest wykorzystanie programowania w zdeklarowanym przez użytkownika kartezjańskim prawoskrętnym układzie współrzędnych. Oceniając efektywność rozwiązywania opisanego zadania przez robota przemysłowego czasem układania krążków na stosie trzecim stwierdzono, że w zakresie prędkości przemieszczania się członów robota (realizacji kroków elementarnych) od 5000÷9000 cm/min oraz sześciu krążków do ułożenia (63 kroki elementarne do wykonania), czasy rozwiązania są krótsze niż 1min. Otrzymane wyniki zależą również od wartości przyspieszenia przy rozpoczynaniu ruchu (przed osiągnięciem zadanej prędkości) oraz opóźnienia przed zakończeniem ruchu w danym bloku. W opracowanym programie obie wartości przyspieszenia i opóźnienia dobierane są automatycznie przez sterownik. Celem określenia wpływu wartości przyspieszenia i opóźnienia ruchu w danym bloku na czas rozwiązania klasycznego zadania sztucznej inteligencji-Wieże Hanoi, konieczne jest zmodyfikowanie obu tych wartości poprzez określenie dwóch współczynników korygujących: ACC – dla wartości przyspieszenia ruchu i DEC – dla wartości opóźnienia ruchu.

Programowanie robota przemysłowego jest jednym z najtrudniejszych w realizacji zadań w zautomatyzowanych systemach wytwarzania. Jest ono szczególnie trudne z logicznego punktu widzenia, gdyż jest zdeterminowane wymaganiami procesu technologicznego. Składa się z dwóch zadań: podstawowego jakim jest programowanie ruchu i pomocniczego obejmującego realizację warunków synchronizacji robota z procesem technologicznym.

Istotnym zadaniem w przypadku dokonywania pomiarów i obliczeń przez robota jest określenie trajektorii ruchu robota. Jest ona zdefiniowana miejscem docelowym jakie robot powinien osiągnąć i sposobem jakim powinien ją zrealizować. Konieczność unikania kolizji wymusza realizację tego zadania metodą kolejnych kroków, tzw. kroków elementarnych. Oznacza to, że każda trajektoria robota jest „sumą" następujących po sobie, uporządkowanych kroków elementarnych, po zrealizowaniu których robot osiąga zaplanowany stan. Stany osiągane są drogą realizacji przyjętego sposobu działania.

Szybkość rozwiązywania klasycznego zadania sztucznej inteligencji, jakim są Wieże Hanoi przez robota przemysłowego bezpośrednio wynika z ograniczeń sprzętowych.

5. BIBLIOGRAFIA

[1] Giżycki P., Stryczek R.: *Wykonanie pakietu oprogramowania dydaktycznego dla robota Motoman*. Praca niepublikowana. Akademia Techniczno-Humanistyczna w Bielsku-Białej, 2013.
[2] Honczarenko J.: *Roboty przemysłowe. Budowa i zastosowanie*, Wydawnictwo Naukowo-Techniczne, Warszawa 2004.
[3] Knosala R. i zespół.: *Zastosowania metod sztucznej inteligencji w inżynierii produkcji*, Wydawnictwa Naukowo – Techniczne, Warszawa 2002.
[4] Nikiel G.: *Robot przemysłowy Motoman UP6. Budowa, obsługa, programowanie*, Bielsko-Biała, 2011.
[5] Rutkowski L.: *Metody i techniki sztucznej inteligencji*, Wydawnictwo Naukowe PWN, Warszawa 2006.
[6] Zdanowicz R.: *Robotyzacja procesów wytwarzania*, Wydawnictwo Politechniki Śląskiej, Gliwice 2007.
[7] Yaskawa Electric Corporation, *YASNAC XRC. INFORM MANUAL*, 2000.
[8] http://www.algorytm.org.