Koncepcja programowania i sterowania robotów w systemie SINUMERIK

An issue for programming and control of robots in the SINUMERIK system

TOMASZ MAKSELON
BOGUSŁAW PYTLAK*

DOI: 10.17814/mechanik.2015.7.325

Przedstawiono koncepcję programowania i sterowania robotów w układzie sterowania numerycznego SINUMERIK 840D sl. Przyjęto założenie programowania ruchu robotów w kartezjańskim układzie współrzędnych *XYZ* za pomocą G-kodów. W tym celu opracowano rozwiązanie problemu transformacji tego ruchu na rzeczywiste osie robota z wykorzystaniem akcji synchronicznych. Rozłożenie programowania na dwie płaszczyzny pozwala na znaczne uproszczenie struktury programów nadrzędnych koordynujących pracę robotów i stanowiska obróbkowego oraz programów sterujących ruchami poszczególnych robotów. Ponadto utworzono dodatkowe okna dialogowe ułatwiające obsługę robotów i ich programowanie metodą uczenia.
SŁOWA KLUCZOWE: sterowanie robotów, układ sterowania CNC, akcje synchroniczne, integracja automatyzacji

The paper presents a concept of practice of programming and control of robots in the SINUMERIK 840D sl control system. An assumption was made that the motions of robots should be programmed in the Cartesian coordinate system XYZ using G-codes. To this effect the problem of transformation of the movements as relevant to the XYZ system to the movements related to the real robot axes has been solved by using synchronized actions. With the program suitably shared in two planes the structure of the main programs coordinating the work of robots and machining station and the programs which control the movements of individual robots could be significantly simplified.
Also, additional dialog procedure with the windows provided for easy control and programming of the robots by the learning method was implemented.
KEYWORDS: control of robots, CNC control system, synchronized actions, integration of automation.

Uniwersalność oraz bardzo szeroki zakres funkcjonalny układu sterowania SINUMERIK, zwłaszcza serii 840D sl, pozwalają na realizację procesu sterowania wielu niekonwencjonalnych obrabiarek i urządzeń CNC. Jednym z nietypowych zastosowań układu sterowania SINUMERIK jest wykorzystanie go do kontrolowania i programowania pracy robota. Do głównych zalet tego rozwiązania należy zaliczyć możliwość programowania ruchów robota w postaci G-kodów oraz możliwość jednoczesnego kontrolowania pracy obrabiarki i obsługującego ją robota (a nawet kilku robotów) za pomocą jednego układu sterowania. Wadą jest natomiast konieczność dostosowania systemu do konkretnej konstrukcji robota.

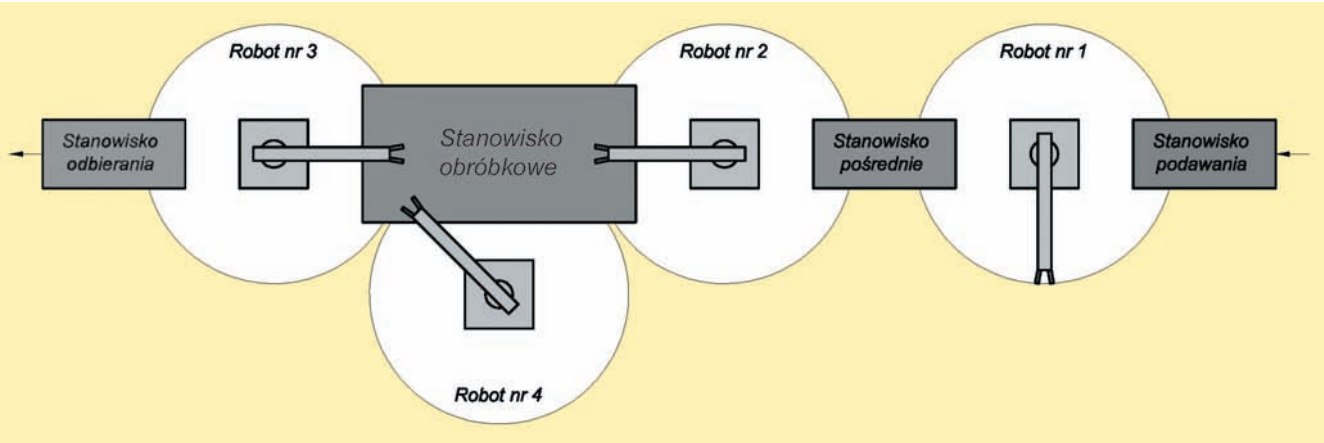
W praktyce klienci zazwyczaj decydują się na zakup gotowych robotów z własnym układem sterowania, co jednak wymaga zaznajomienia się z układem sterowania oraz sposobem programowania zarówno obrabiarki, jak i robota, a następnie – rozwiązania problemu komunikacji pomiędzy nimi. Oczywiście prowadzi się prace nad integracją automatyzacji, np. obrabiarek oraz obsługujących je robotów. Przykładem takiego gotowego rozwiązania jest pakiet technologiczny Run MyRobot opracowany wspólnie przez firmy Siemens i KUKA, który pozwala na programowanie i sterowanie robotów KUKA z poziomu układu sterowania SINUMERIK.

W przypadku robota zaprojektowanego i wykonanego według własnego projektu pojawia się jednak problem: jak nim sterować i jak go zaprogramować? Rozwiązaniem może być wykorzystanie układu sterowania obrabiarki, np. SINUMERIK 840D sl.

W artykule poddano analizie gniazdo obróbkowe obsługiwane przez cztery roboty (rys. 1). Wszystkie roboty, a także stanowisko obróbkowe, są sterowane za pomocą jednego układu sterowania SINUMERIK 840D sl.

Robot nr 1 pobiera elementy ze stanowiska podawania i odkłada na stanowisko pośrednie, skąd robot nr 2 odbiera te elementy i przekazuje na stanowisko obróbkowe. Elementy po obróbce odbiera robot nr 3 i przekłada na stanowisko odbierania. Robot nr 4 pełni rolę pomocniczą – czyści uchwyt mocujący detale na stanowisku obróbkowym.

Opisana w artykule koncepcja sterowania i programowania robotów obsługujących gniazdo obróbkowe została wdrożona i sprawdza się w warunkach przemysłowych.



Rys. 1. Schemat gniazda obróbkowego obsługiwanego przez cztery roboty

* Mgr inż. Tomasz Makselon (tomasz.makselon@siemens.com) – Siemens Sp. z o.o., dr inż. Bogusław Pytlak (bpytlak@ath.bielsko.pl) – Akademia Techniczno-Humanistyczna w Bielsku-Białej

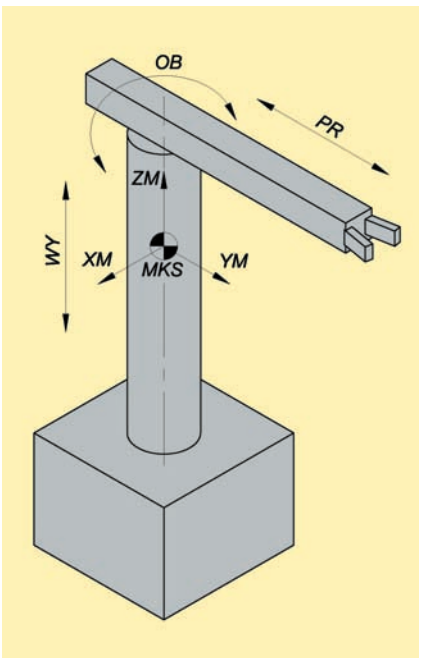
Sterowanie ruchem robotów

Konstrukcja każdego z omawianych robotów bazuje na konfiguracji cylindrycznej [1]. Każdy z nich ma dwie osie liniowe – zapewniające wysuw ramienia (oś *PR*) i jego ruch góra–dół (oś *WY*) – oraz jedną oś obrotową (oś *OB*) służącą do obrotu ramienia (rys. 2). Do poruszania osiami robotów zastosowano napędy typu SINAMICS S120 firmy Siemens, aby zapewnić pełną integrację z układem sterowania SINUMERIK.

Biorąc pod uwagę cylindryczny obszar pracy robota, początkowo rozważano bezpośrednie programowanie jego rzeczywistych osi w układzie biegunowym. Podejście to odrzucono jednak ze względu na charakterystyczne dla tego przypadku ograniczenie konstrukcyjne – polegające na tym, że wraz z obrotem ramienia robota zmienia się jego wysokość i wysięg – oraz wynikającą z niego konieczność wykonywania dodatkowych obliczeń, a także ze względu na mało wygodny sposób programowania biegunowego.

Druga koncepcja zakładała programowanie robota w kartezjańskim układzie współrzędnych *XYZ* (rys. 2), tak jak w przypadku większości obrabiarek CNC. W tym celu zdefiniowano w układzie sterowania dodatkowe symulowane osie maszynowe *XM*, *YM* i *ZM* oraz osie kanałowe *X*, *Y* i *Z*. Osie *XM*, *YM* i *ZM* tworzą kartezjański maszynowy układ współrzędnych *MKS*, który można traktować jako odpowiednik układu współrzędnych podstawy robota [2]. Z kolei osie *X*, *Y* i *Z* tworzą układ współrzędnych przedmiotu obrabianego *WKS*, który może być traktowany jako odpowiednik układu współrzędnych bazy robota. Użyte tu określenia *maszynowego układu współrzędnych* i *układu współrzędnych przedmiotu obrabianego* nie są – niestety – w pełni adekwatne do omawianego zastosowania układu sterowania SINUMERIK, gdyż wywodzą się z terminologii obrabiarek CNC, a nie robotów.

W prezentowanym podejściu głównym problemem jest odpowiednia transformacja ruchu w kartezjańskim układzie współrzędnych *XYZ* na rzeczywiste osie robota *PR*, *OB* i *WY* (rys. 3), czyli rozwiązanie odwrotnego zadania kinematyki. Do tego celu wykorzystano funkcjonalność układu sterowania SINUMERIK oraz narzędzie akcji synchronicznych [3, 4]. Przekształcenia ruchu w układzie *WKS* w osiach *X*, *Y* i *Z* na ruch w układzie *MKS* w osiach *XM*, *YM* i *ZM* dokonuje sam układ sterowania, stosując przekształcenia odwrotne do przesunięć punktu zerowego oraz funkcje FRAME.



Rys. 2. Poglądowa konstrukcja analizowanego robota wraz z zaznaczonymi osiami oraz kartezjańskim maszynowym układem współrzędnych *MKS*

W analizowanym przypadku układ *WKS* pokrywał się z układem *MKS*, gdyż nie było potrzeby wprowadzania pomiędzy nimi dodatkowych przesunięć (rys. 3). Pozostała więc jedynie transformacja ruchu w układzie *MKS* na ruch w rzeczywistych osiach robota. Transformacja ta powinna działać w każdym trybie pracy: JOG, MDA i AUTO. Należało zatem użyć działających modalnie akcji typu IDS wraz ze słowem kluczowym WHENEVER, które zapewnia wykonywanie akcji w każdym takcie interpolatora, gdy tylko warunek akcji jest spełniony. Do poruszania osiami robota przyjęto polecenie pozycjonowania POS[oś]. Dla uproszczenia obliczeń dotyczących pozycji osi robota *PR*, *OB* i *WY* przyjęto początek układu *MKS* w osi obrotu robota (rys. 3).

Współrzędne *XM*, *YM* i *ZM* określają położenie punktu kodowego narzędzia TCP (*tool center point*). W obliczeniach jest brana pod uwagę długość narzędzia (chwytaka) przechowywana w zmiennej D_CHWYT typu GUD, co zapewnia poprawność obliczeń dla narzędzi o innej długości. W przypadku omawianej aplikacji wystarczające jest podanie długości narzędzia tylko w jednej osi, choć istnieje także możliwość dodania wymiarów w drugiej i trzeciej osi oraz uwzględniania ich w obliczeniach.

Do obliczenia kąta obrotu osi *OB* wykorzystano dwuargumentową funkcję ATAN2, która na podstawie dwóch prostopadłych do siebie wektorów (*XM* i *YM*) zwraca kąt wektora sumarycznego, który jest liczony względem drugiego argumentu funkcji:

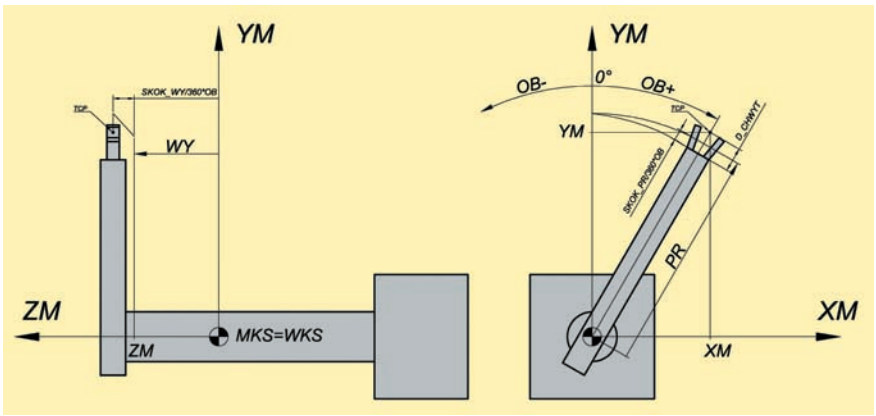
$$OB = \mathrm{ATAN2}(XM, YM) \tag{1}$$

To oznacza, że osią odniesienia dla kąta *OB* jest oś *YM*+, co odpowiada wyjściowemu położeniu ramienia robota „na wprost”, tj. wzdłuż osi *YM*. Z kolei wartość promienia *PR* jest wyliczana za pomocą równania:

$$PR = \sqrt{XM^2 + YM^2} - \mathrm{D\_CHWYT} + \frac{\mathrm{SKOK\_}PR \cdot OB}{360} \tag{2}$$

Od długości wyznaczonej na podstawie współrzędnych *XM* i *YM* należy zatem odjąć długość uchwytu D_CHWYT i dodać do niej składnik SKOK_*PR*·*OB*/360, którego zadaniem jest kompensacja skrócenia ramienia robota w wyniku jego obrotu o kąt *OB*. Wyznaczenie wysokości *WY* jest najprostsze, gdyż ruch w osi *ZM* ma bezpośrednie przełożenie na oś *WY*. W tym przypadku dodatkowo należy uwzględnić składnik zmiany wysokości SKOK_*WY*·*OB*/360, kompensujący zmianę wysokości ramienia robota w wyniku jego obrotu o kąt *OB*:

$$WY = Z + \frac{\mathrm{SKOK\_}WY \cdot OB}{360} \tag{3}$$



Rys. 3. Sposób obliczenia pozycji osi robota *PR*, *OB* i *WY*

Definicje akcji synchronicznych dla tych zależności są następujące:

```
IDS=1 WHENEVER SYG_RM[0]==0 DO
    POS[OB]=AC(ATAN2($$AA_IM[XM],$$AA_IM[YM]))
IDS=2 WHENEVER SYG_RM[0]==0 DO
    POS[PR]=AC(SQRT(POT($$AA_IM[XM])+POT($$AA_
    IM[YM])) – D_CHWYT+SKOK_PR/360*$$AA_IM[OB])
IDS=3 WHENEVER SYG_RM[0]==0 DO POS[WY]=AC($$AA_
    IM[ZM]+SKOK_WY/360*$$AA_IM[OB])
```

Akcje te są wykonywane wtedy, gdy wartość zmiennej SYG_RM[0] typu GUD (są to specjalne GUD-y przeznaczone dla akcji synchronicznych) jest równa 0. Zmienna ta decyduje o wykonywaniu opisywanej transformacji. Należy zwrócić jeszcze uwagę na zmienne $$AA_IM[oś] służące do odczytywania położenia poszczególnych osi *XM*, *YM* i *ZM*. Są to zmienne przebiegu głównego (oznaczenie: $$A...), co gwarantuje odczyt aktualnych współrzędnych w czasie rzeczywistym.

Oprócz tego zdefiniowane zostały dodatkowe akcje:

```
IDS=4 EVERY $A_IN[1]==1 DO RESET(1) RESET(2) RESET(3)
IDS=5 EVERY $A_IN[2]==1 DO SYG_RM[0]=0
IDS=6 EVERY $A_IN[3]==1 DO SYG_RM[0]=1
```

Zadaniem akcji nr 4 jest kasowanie działania akcji nr 1÷3, gdy za każdym razem na wejściu cyfrowym $A_IN[1] pojawi się 1. Natomiast akcje nr 5 i 6 zmieniają wartość zmiennej SYG_RM[0] w zależności od sygnału na wejściach $A_IN[2] i $A_IN[3] – w ten sposób zapewniona jest kontrola spełnienia warunku niezbędnego do wykonywania transformacji kinematycznej.

Podane akcje synchroniczne obowiązują dla pojedynczego robota, którego obsługę i programowanie realizuje się w ramach jednego kanału układu sterowania SINUMERIK. Dla pozostałych trzech robotów należy te akcje powielić trzykrotnie oraz zaprojektować dodatkowe trzy kanały. Rozwiązanie problemu uruchomienia tych akcji synchronicznych wraz ze startem układu sterowania bazuje na cyklu producenta PROG_EVENT.SPF, którego moment wykonania określa dana maszynowa MD20108 – np. po uruchomieniu układu sterowania. Aby rozszerzyć funkcjonalność cyklu PROG_EVENT.SPF, należy utworzyć cykl producenta CYCPE_MA.SPF zawierający omawiane akcje synchroniczne. Po utworzeniu cyklu CYCPE_MA.SPF istnieje możliwość jego wywołania w cyklu PROG_EVENT.SPF.

Ponadto w przypadku bezpośredniego ruchu w osiach robota *PR*, *OB* i *WY* – np. w trybie pracy JOG – wymienione akcje są zastępowane akcjami wyliczającymi pozycję osi *XM*, *YM* i *ZM*, realizującymi proste zadanie kinematyki:

```
IDS=10 WHENEVER (SYG_RM[1])==0 DO
    PRESETON(XM,(($VA_IM[PR]+D_CHWYT) – SKOK_
    PR/360*$VA_IM[OB])*SIN($VA_IM[OB]))
IDS=11 WHENEVER (SYG_RM[1])==0 DO
    PRESETON(YM,(($VA_IM[PR]+D_CHWYT) – SKOK_
    PR/360*$VA_IM[OB])*COS($VA_IM[OB]))
IDS=12 WHENEVER (SYG_RM[1])==0 DO PRESETON(ZM,$VA_
    IM[WY] – SKOK_WY/360*$VA_IM[OB])
```

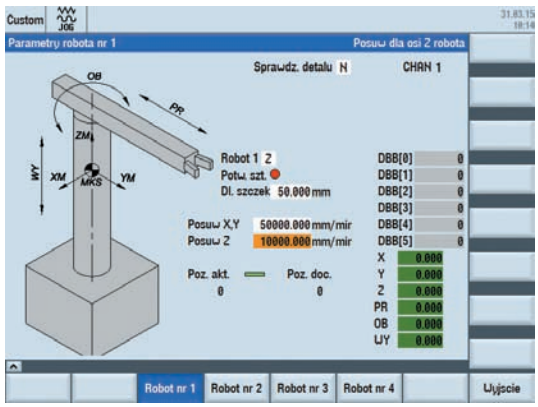
Na podstawie aktualnej pozycji enkoderów, odczytanej ze zmiennych systemowych $VA_IM[oś], określających aktualne położenie osi *PR*, *OB* i *WY*, zostały wyliczone aktualne pozycje osi *XM*, *YM* i *ZM*. Te z kolei zostały przyporządkowane – za pomocą polecenia PRESETON(wartość) – do osi układu *MKS*.

Obsługa robotów

Do obsługi robotów wykorzystano panel sterowania HT 8 (rys. 4), który przypomina panel sterowania robotów. Zapewnia on niezbędną mobilność podczas obsługi i programowania (np. poprzez uczenie pozycji) robotów. Pozycje rzeczywistych osi robota *PR*, *OB* i *WY* są wyświetlane w formie dodatkowych osi. Poruszanie osiami robota *X*, *Y* i *Z* oraz *PR*, *OB* i *WY* – np. w trybie JOG – jest realizowane za pomocą odpowiednio skonfigurowanych przycisków na panelu operatora (rys. 4). Dodatkowo panel umożliwia wybór numeru kanału (robota lub stanowiska obróbkowego) oraz załączenie i wyłączenie posuwu.

Rys. 4. Widok panelu sterowania HT 8 do obsługi i programowania robotów

Do sterowania ruchami szczęk robota, a także do wprowadzania ich długości, służy okno dialogowe (rys. 5) w obszarze Custom (Użytkownik) układu sterowania. W oknie widoczne są: aktualny numer kanału (CHAN 1), aktualne współrzędne *X*, *Y*, *Z*, *PR*, *OB* i *WY* robota oraz zmienne PLC (DBB[0]÷DBB[5]), realizujące komunikację pomiędzy poszczególnymi robotami i stanowiskiem obróbkowym.



Rys. 5. Okno dialogowe do obsługi dodatkowych funkcji robota

W polu „Robot nr 1” można ręcznie sterować otwieraniem i zamykaniem szczęk („Z” oznacza zamknięte szczęki). To, czy dany detal został uchwycony, potwierdza czujnik – jego sygnał jest wyświetlany w polu „Potw. szt.” (● oznacza, że detal nie został uchwycony). Poniżej tego pola znajduje się pole „Dl. Szczek”, w którym podaje się rzeczywistą długość chwytaka robota. Kontrolę uchwycenia detalu można wyłączyć w polu „Sprawdz. detalu” („N” oznacza wyłączenie kontroli). Do definiowania wartości posuwu dla ruchów robota przeznaczone są pola „Posuw X,Y” (zmienna FFF_XY) oraz „Posuw Z” (zmienna FFF_Z). W oknie wyświetlane są także informacje dotyczące aktualnej i docelowej pozycji robota: „Poz. akt.” i „Poz. doc.”.

Programowanie robotów

W prezentowanym podejściu do programowania robotów starano się utrzymać kompatybilność z programowaniem obrabiarek CNC oraz wykorzystać zalety programowania robotów (np. metodę uczenia). Podejście to bazuje na dwóch płaszczyznach programowania. Na pierwszej zostały utworzone programy nadrzędne, koordynujące pracę gniazda

obróbkowego, natomiast na drugiej zaprogramowano same ruchy robotów i stanowiska obróbkowego. Głównym zadaniem nadrzędnych programów sterujących jest takie skoordynowanie pracy stanowiska obróbkowego, poszczególnych robotów oraz stanowisk podawania, odbierania i pośredniego, aby uniknąć ewentualnych kolizji. Oto fragment takiego programu dla robota nr 1:

```
...
;Wjazd do stanowiska podawania
WHILE NOT(($A_DBB[0])B_AND('B00000001'))
  MSG("Brak zezwolenia na wjazd do stanowiska podawania")
  G4 F0.01
  STOPRE
ENDWHILE
MSG()

STOPRE
$A_DBB[2]=($A_DBB[2])B_OR('B00100000') ;Ustawienie
    potwierdzenia pozycji nr 2
$A_DBB[4]=1 ;Pozycja aktualna robota
$A_DBB[5]=2 ;Pozycja docelowa robota
STOPRE

CALL "ROBOT_1_RUCHY" BLOCK POZYCJA_1 TO
    POZYCJA_2

STOPRE
$A_DBB[4]=2 ;Pozycja aktualna robota
$A_DBB[5]=3 ;Pozycja docelowa robota
STOPRE

CALL "ROBOT_1_RUCHY" BLOCK POZYCJA_2 TO
    POZYCJA_3
...
```

W programie tym poleceniem CALL wywoływane są poszczególne fragmenty podprogramu ROBOT_1_RUCHY.SPF, w którym zaprogramowano tylko same ruchy robota na poszczególne pozycje. Przykładowy fragment programu ROBOT_1_RUCHY.SPF dla ruchów robota nr 1:

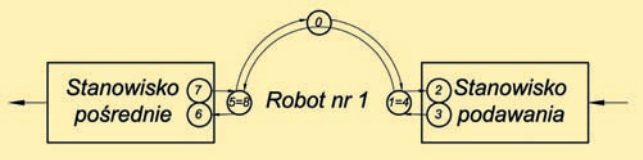
```
...

POZYCJA_1:
G641 ADIS=_ADIS[2]
G1 X=_X[2] Y=_Y[2] Z=_Z[2] F=FFF_XY ;Wjazd do stanowiska
    podawania
POZYCJA_2:
G602
G1 X=_X[3] Y=_Y[3] Z=_Z[3] FB=FFF_Z ;Dojazd do detalu
M60 ;Zamkniecie szczek
G4 F0.2
G1 X=_X[2] Y=_Y[2] Z=_Z[2] FB=FFF_Z ;Podniesienie detalu
POZYCJA_3:
G641 ADIS=_ADIS[4]
...
```

W tworzeniu i modyfikacji tego typu programów bardzo pomocny jest graf przedstawiający kolejność wykonywanych przez robota ruchów na poszczególne pozycje (rys. 6).

Aby móc programować poszczególne pozycje robota metodą uczenia, zdefiniowano trzy kanałowe zmienne tablicowe _X, _Y, _Z[nr_pozycji] typu GUD, których zadaniem jest przechowywanie nauczonych pozycji. Dla uproszczenia procesu uczenia utworzono w obszarze Custom okno dialogowe, w którym operator może zapamiętać daną pozycję w zmiennych _X, _Y, _Z (rys. 7).

W polu „Nr pozycji" można bezpośrednio podać numer uczonej pozycji lub określić ją za pomocą przycisków „NR+" i „NR–" (zmieniających numer pozycji o 1 w górę lub w dół). Współrzędne danej pozycji są wyświetlane w polach „X", „Y" i „Z". W celu zapamiętania danej pozycji należy nacisnąć



Rys. 6. Graf ruchów dla robota nr 1 z zaznaczeniem jego pozycji

Rys. 7. Okno dialogowe do zapamiętywania poszczególnych pozycji

przycisk „Przejmij pozycje” – w tym momencie zostają zapamiętane aktualne współrzędne osi *X*, *Y* i *Z*. Edycja zapamiętanych współrzędnych dla danego punktu następuje po naciśnięciu przycisku „Edytuj pozycje”, a zapisanie zmienionych wartości – po naciśnięciu przycisku „Zapisz zmiany”.

Dodatkowo istnieje możliwość parametryzacji polecenia płynnego przechodzenia pomiędzy blokami G641 ADIS=, dla każdej pozycji robota [5]. Wartość ADIS określa tzw. wielkość ścinania, tj. drogę do końca danego bloku, po której następuje przełączenie na następny blok. Jest ona zapamiętana w zmiennej _ADIS[nr_pozycji]. W efekcie robot nie osiąga zadanej pozycji, lecz mija ją w pewnej odległości. Odpowiada to ruchowi robota z podaną wartością przybliżenia CONT. Gdy dana pozycja ma zostać osiągnięta w sposób dokładny, należy w programie ruchów robota zastosować funkcję zatrzymania dokładnego, np. G602.

Podsumowanie

Zaprezentowana koncepcja sterowania robotami za pomocą układu sterowania SINUMERIK:

- wykorzystuje istniejący układ sterowania obrabiarki do programowania w G-kodach i sterowania ruchami obsługujących ją robotów,
- jest bardzo dobrą alternatywą dla sterowania robotami własnej konstrukcji,
- wymaga rozwiązania problemu transformacji ruchu w kartezjańskim układzie współrzędnych na ruch poszczególnych osi robota (i zależy od konstrukcji danego robota) – np. za pomocą narzędzia akcji synchronicznych,
- dzięki opracowaniu dodatkowych okien dialogowych pozwala operatorowi w prosty i przyjazny sposób kontrolować robota i programować jego pozycję metodą uczenia.

Przedstawione w pracy podejście zostało wdrożone w przemyśle i działa bezproblemowo.

LITERATURA

1. Honczarenko J. „*Roboty przemysłowe. Budowa i zastosowanie*”. Warszawa: WNT, 2010.
2. Materiały szkoleniowe firmy KUKA.
3. Stryczek R., Pytlak B. „*Elastyczne programowanie obrabiarek*”. Warszawa: PWN, 2011.
4. Instrukcja producenta/serwisowa SINUMERIK 840D sl/828D: „*Function Manual Synchronized Actions*”, 03/2013.
5. Instrukcja użytkownika SINUMERIK 840D sl/828D: „*Podręcznik programowania. Podstawy*”, 03/2013. ■