Zbigniew ŁUKASIK, Aldona KUŚMIŃSKA-FIJAŁKOWSKA, Jacek KOZYRA

ROBOT PRZEMYSŁOWY W PROCESIE PRODUKCYJNYM

W artykule opracowano algorytm oraz oprogramowanie umożliwiające nanoszenie nadruku w procesie wytwarzania produktu finalnego w oparciu o robota Kawasaki. W wyniku adaptacji robota proces nanoszenia nadruku ulegnie wyraźnemu przyśpieszeniu, powodując zwiększenie dziennej liczby wykonywanych operacji. Proponowana koncepcja zwiększa efektywność, jakość zachodzących procesów. Podjęty w pracy kierunek badań jest aktualny i może być interesujący dla przemysłu

WSTĘP

Jeszcze do niedawna polscy inwestorzy odnosili się bardzo sceptycznie do wykorzystywania robotów przemysłowych w różnego rodzaju procesach produkcyjnych, traktując je wyłącznie jako bardzo kosztowną inwestycję. Takie postępowanie przedsiębiorców wynika z dążenia do optymalizacji kosztów w krótkim okresie, pomijając jednocześnie wymierne korzyści długookresowe[14],[10].

Inwestycja w zainstalowanie robota do procesu produkcyjnego - dzięki m.in. poprawie wydajności produkcji, wysokim stopniu powtarzalności ruchów, obniżeniu awaryjności oraz redukcji kosztów - bardzo szybko się zwraca, stanowiąc jednocześnie źródło wymiernych korzyści ekonomicznych.[5],[12],[7]

W większości zakładów przemysłowych wykorzystuje się technikę druk tamponowego, która polega na przenoszeniu farby z wzoru znajdującego się na formie drukowej za pomocą tamponu na powierzchnię przedmiotu. Jest to pracochłonna czynność i wymaga dużego doświadczenia od pracownika, aby nadruk był wysokiej jakości.

Adaptacja robota do nanoszenia nadruku w procesie wytwarzania finalnego produktu, pozwoli usprawnić procesy w przedsiębiorstwie, a także zwiększy efektywność, jakość zachodzących procesów przy ciągłości zasilania, gdyż układy sterujące robotami wymagają wysokiej jakości dostawy energii elektrycznej.[11], [15], [9]

1. TWORZENIE PROGRAMÓW REALIZUJĄCYCH ZADANIA W OPARCIU O ROBOTA PRZEMYSŁOWEGO

Tworzenie programów realizujących zadania, dla których budowany jest system to jeden z najważniejszych etapów integracji robota. Każdy z producentów wyposaża swoje roboty w języki programowania wysokiego poziomu różniące się miedzy sobą, co do stosowanego nazewnictwa instrukcji, struktury programu, czy sposobu jego tworzenia. Wszystkie języki oferują jednak podobne mechanizmy do realizacji zadań: procedury, podprogramy, instrukcje wyboru, pętli, ruchu oraz operacji wejścia/wyjścia. Przed przystąpieniem do tworzenia programów należy poznać wszystkie elementy wchodzące w skład stanowiska, osprzęt robota i przestrzeń roboczą ramienia.

Pamiętać należy o przemyślanym wprowadzaniu wszelkich poprawek, tworzeniu i używaniu punktów. Pozwala to uniknąć niebezpiecznego, przypadkowego lub nieoczekiwanego zachowania ramienia robota i innych elementów zintegrowanych na stanowisku zarówno w trakcie tworzenia jak i podczas wykonywania utworzonego programu. [2],[3] Proces tworzenia programu można podzielić na kilka faz (Rys. 1)

Jednym z najważniejszych etapów programowania jest „uczenie" punktów docelowych ruchu przez przemieszczanie ramienia roboczego za pomocą Teach Pendant'a i zapamiętywanie jego wybranych ustawień [6]. Czasami punkty docelowe tworzy się za pomocą oprogramowania zewnętrznego. Ten sposób zwany, jako Offline Programming, używany jest do wstępnego stworzenia siatki punktów w celu sprawdzenia, czy projektowana aplikacja jest w stanie spełnić wszelkie oczekiwania zanim zostanie fizycznie wykonana. Innym jego zastosowaniem jest wprowadzanie poprawek poza stanowiskiem roboczym bez ingerencji w pracę systemu. Punkty docelowe składają się z kilku elementów takich jak:

- położenie przestrzenne X, Y, Z,
- orientacja przestrzenna, czyli skręcenie osi względem danego układu współrzędnych narzędzia, które mówi o przestrzennym ułożeniu narzędzia roboczego.

Tworzenie sitki punktów należy zacząć od określenia punktu wyjściowego (Home Position).[13] Tworzona siatka punktów musi być zawsze zamknięta. Znaczy to tyle, że pierwszym i ostatnim punktem cyklicznego ruchu robota musi być punkt wyjściowy. Używając instrukcji ruchu w czasie programowania ruchów do punktów docelowych trzeba przewidywać, czy dany ruch będzie możliwy do wykonania oraz, czy nie spowoduje kolizji ramienia robota lub jego osprzętu z innymi elementami systemu. Siatkę punktów programu tworzy się jedną lub kilkoma z wymienionych metod:

1. Programowanie online, w trakcie, którego ramię robota jest przemieszczane, a jego położenia zapamiętywanie metodami:
   - Teach, Program and Repeat – naucz, programuj i powtarzaj – przez przemieszczanie ramienia z użyciem Teach Pendant'a.
   - Walk Through and Folow – przeprowadź i podążaj – przez bezpośrednie poruszanie narzędziem roboczym przymocowanym do ramienia robota.
2. Programowanie offline – z użyciem oprogramowania na zewnętrznym komputerze

Rys. 1. *Fazy tworzenia programu*

Roboty Kawasaki mogą być programowane za pomocą jednego z dwóch języków programowania, języka blokowego oraz języka AS. Wybór zależy od rodzaju aplikacji bądź preferencji programisty. Język blokowy jest programowaniem uproszczonym, a realizowany jest za pomocą wirtualnego Teach Pendanta. W każdym kroku zapisywane są dane potrzebne do przemieszczania robota. Jeden krok składa się z dwóch części, pierwsza część to dane o aktualnym położeniu robota w przestrzeni, natomiast druga część to tak zwane dane pomocnicze. Wśród danych pomocniczych możemy wyróżnić prędkość, informację o interpolacji, bądź też sposób poruszania się robota. Tak, więc w danych o położeniu mamy zawarte położenie kątowo kiści robota, zaś w danych pomocniczych informację, w jaki sposób ta pozycja zostanie osiągnięta. Jest to prosta i elastyczna metoda programowania, sprawdzająca się głównie w takich aplikacjach jak zgrzewanie, malowanie, klejenie.[4]

Język AS pozwala użytkownikowi na bardziej zaawansowaną w opcje możliwość programowania. Jest wykorzystywany w przypadku, gdy język blokowy nie jest wystarczający. Składa się z dwóch typów instrukcji, instrukcji sterujących oraz poleceń do monitorowania, które są interpretowane w wierszu poleceń i sterując wykonaniem programu. Język AS pozwala na zmianę procesu podczas wykonywania programu, w momencie wykonywania programu głównego, w tle może pracować kilka innych programów. Programy mogą być tworzone offline przy użyciu dowolnego edytora tekstowego, a ich archiwizacja może być dokonana przez port Ethernet, RS232, lub kartę CF w złączu PCMCIA.

Programowanie oraz planowanie trajektorii ruchu robota przemysłowego jest bardzo ważne z praktycznego punktu widzenia.[8] Robot nie wykona żadnych czynności określonych przez zadanie, bez uprzedniego zaprogramowania go przez użytkownika. Aby np. przenieść przedmiot z punktu A do punktu B trzeba określić trajektorię, po której będzie poruszał się robot. Zadanie jest wykonywane w trójwymiarowej przestrzeni kartezjańskiej, nazywanej często przestrzenią zewnętrzną (lub przestrzenią zadań) w odróżnieniu od przestrzeni wewnętrznej, w której są określone współrzędne poszczególnych ogniw manipulatora. Robot przemysłowy Kawasaki posiada sześć stopni swobody. Ruchy poszczególnych stopni swobody składają się na ruchy końcówki manipulatora (ruch umownego środka chwytaka robota) w przestrzeni kartezjańskiej. [1]

Do programowania robota za pomocą komputera wykorzystano program KCwinTCP. Umożliwia on wykonanie m.in. następujących operacji: wprowadzanie poleceń języka AS, tworzenie programów, zapisywanie i ładowanie programów. KCwinTCP posiada trzy tryby pracy: monitora (*monitor mode*), edytora (*editor mode*) oraz odtwarzania (*playback mode*).

Tryb monitora jest podstawowym trybem systemu AS. Z tego trybu przechodzi się do trybu edycji i odtwarzania. Służy do wykonywania poleceń wprowadzanych za pomocą klawiatury komputera. Mogą to być komendy ruchu, ustawianie parametrów ruchu, sterowanie wyjściami robota, zdalne konfigurowanie parametrów kontrolera, zapisywanie pozycji robota. Tryb ten stanowi alternatywną metodę sterowania robotem.

Tryb edytora służy do tworzenia nowych programów lub modyfikacji już istniejących. W trybie tym system wykonuje wyłącznie polecenia edycji. Wywołanie trybu następuje po wprowadzeniu komendy *EDIT nazwa_programu* natomiast wyjście za pomocą komendy *E*.

Kursor zmieni postać z > na *?*. Pisanie programu polega na wprowadzaniu komend i zatwierdzaniu ich klawiszem ENTER.

Tryb odtwarzania uruchomiony jest podczas odtwarzania programu. W tym trybie zrealizowane mogą być tylko wybrane polecenia, np.:

- ABORT – zatrzymuje wykonywanie programu po zakończeniu bieżącego kroku;
- HOLD – natychmiast zatrzymuje wykonywanie programu;
- CONTINUE – wznawia zatrzymanie programu (instrukcją HOLD i ABORT).

2. PROGRAMOWANIE ZROBOTYZOWANEGO STANOWISKA DO NANOSZENIA NADRUKU NA ELEMENTACH W OPARCIU O ROBOTA KAWASAKI FS 003N

Instrukcje języka AS obejmują polecenia sterujące wykonaniem programu, definiowania pozycji robota, ruchowe, sterujące prędkością i dokładnością osiągania pozycji docelowej oraz narzędziem.

Oprócz instrukcji ruchu wymienionych poniżej w Tabela 1. istnieją jeszcze dwie instrukcje, które służą do ruchu po okręgu Tabela 2

Tab. 1. *Instrukcje ruchu*

JMOVE nazwa_zmiennej_pozycji	przesuwa robota w ruchu z interpolacją osiową.
LMOVE nazwa_zmiennej_pozycji	przesuwa robota w ruchu z interpolacją liniową.
JAPPRO nazwa_zmiennej_pozycji, odległość	przesuwa robota do pozycji znajdującej się o odległość od punktu wzdłuż osi Z układu współrzędnych. Ruch odbywa się w interpolacji osiowej
LAPPRO nazwa_zmiennej_pozycji, odległość	przesuwa robota do pozycji znajdującej się o odległość od punktu wzdłuż osi Z układu współrzędnych. Ruch odbywa się w interpolacji liniowej.
JDEPART odległość	przesuwa robota do pozycji znajdującej się o odległość od bieżącego punktu wzdłuż osi Z układu współrzędnych. Ruch odbywa się w interpolacji osiowej.
LDEPART odległość	przesuwa robota do pozycji znajdującej się o odległość od bieżącego punktu wzdłuż osi Z układu współrzędnych. Ruch odbywa się w interpolacji liniowej

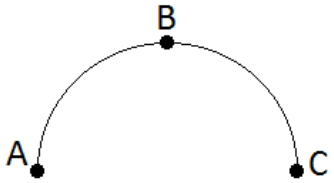
Tab. 2. *Instrukcje ruchu*

C1MOVE nazwa_zmiennej_pozycji1	przesuwa robota do punktu znajdującego się w połowie drogi na trajektorii koła;
C2MOVE nazwa_zmiennej_pozycji2	przesuwa robota do punktu znajdującego się na końcu trajektorii koła;

Po użyciu tych komend robot będzie poruszał się po łuku, w którym *pozycja1* jest pozycją pośrednią, a *pozycja2* końcową łuku. Do pozycji początkowej łuku końcówka robota może dojechać w dowolny sposób. Poniżej pokazano przykłady:

Przykład 1
LMOVE A
C1MOVE B
C2MOVE C

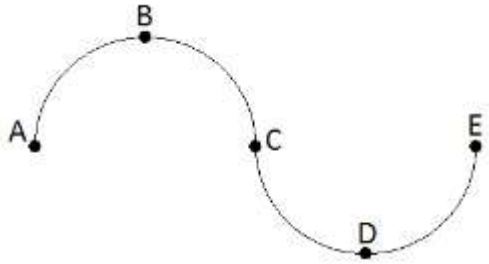
Robot przesuwa się z interpolacją liniową do punktu A, a następnie z interpolacją kołową przesuwa się po łuku ABC (Rys. 2)



Rys. 2. *Ruch robota po łuku ABC*

Przykład 2
LMOVE A
C1MOVE B
C2MOVE C
C1MOVE D
C2MOVE E

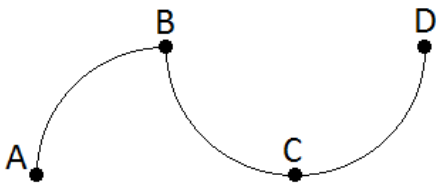
Robot przesuwa się z interpolacją liniową do punktu A, po czym z interpolacją kołową przesuwa się po łuku ABC, a następnie po łuku CDE (Rys. 3)



Rys. 3. *Ruch robota po łukach ABC i CDE*

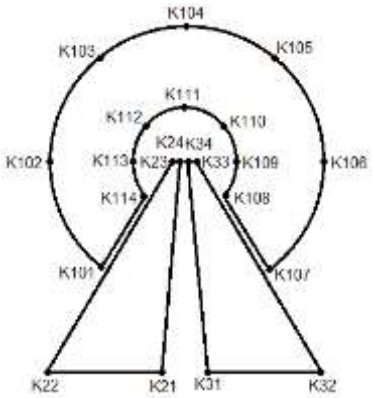
Przykład 3
LMOVE A
C1MOVE B
C1MOVE C
C2MOVE D

Robot przesuwa się z interpolacją liniową do punktu A, po czym z interpolacją kołową przesuwa się do połowy łuku ABC, a następnie po łuku BCD (Rys. 4)



Rys. 4. *Ruch robota po łukach AB i BCD*

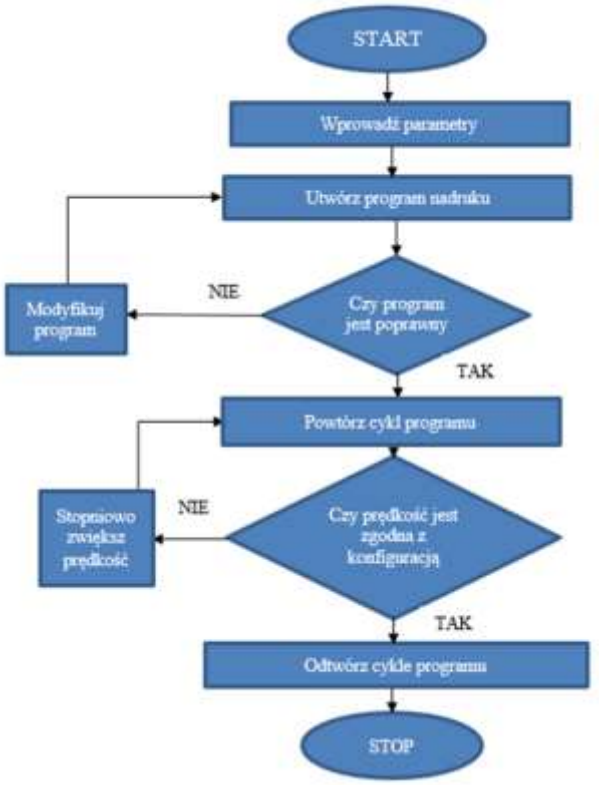
W wyniku opracowania koncepcji adaptacji robota przemysłowego do nanoszenia nadruku (Rys. 5.) na elemencie w procesie wytwarzania finalnego produktu opracowano algorytm (Rys. 6.) oraz specjalizowane oprogramowanie.



Rys. 5. *Nadruk na elemencie (przyładowy)*

Procedura tworzenia programu:

```
.PROGRAM nadruk()
 1      JMOVE #start
 2      JAPPRO k101,5
 3      LMOVE k101
 4      C1MOVE k102
 5      C2MOVE k103
 6      C1MOVE k104
 7      C2MOVE k105
 8      C1MOVE k106
 9      C2MOVE k107
10      LMOVE k108
11      C1MOVE k109
12      C2MOVE k110
13      C1MOVE k111
14      C2MOVE k112
15      C1MOVE k113
16      C2MOVE k114
17      LMOVE k101
18      LDEPART 5
19      LAPPRO k21,5
20      LMOVE k21
21      LMOVE k22
22      LMOVE k23
23      LMOVE k24
24      LMOVE k21
25      LDEPART 5
26      LAPPRO k31,5
27      LMOVE k31
28      LMOVE k32
29      LMOVE k33
30      LMOVE k34
31      LMOVE k31
32      LDEPART 5
.END
```



Rys. 6. *Algorytm programu*

PODSUMOWANIE

Proponowana adaptacja robota przemysłowego do nanoszenia nadruku przyniesie szereg korzyści:

- **optymalne gospodarowanie.** Wyeliminowanie niedogodności w codziennym cyklu pracy, co przekłada się na koszty.
- **zwiększenie wydajności operacyjnej.** Zaimplementowanie zrobotyzowanego stanowiska pozwoli na poprawienie, jakości nadruku. Proces nanoszenia nadruku ulegnie wyraźnemu przyśpieszeniu, powodując zwiększenie dziennej liczby wykonywanych operacji
- **poprawa jakości.** Ponieważ proces będzie zrobotyzowany skróceniu ulegnie czas wykonania operacji nadruku i jakości wykonania.

BIBLIOGRAFIA

1. Opracowanie Zakładu Automatyzacji Procesów Uniwersytetu Technologiczno- Humanistycznego w Radomiu
2. Łukasik Z. Automatyzacja procesów sterowania i zarządzania. Wydawnictwo PR, Radom 2011
3. Stanowisko laboratoryjne „Integracja zrobotyzowanych systemów przemysłowych" Opracowanie Zakładu Automatyzacji Procesów
4. www.astor.com.pl.
5. Łukasik, Z., Kuśmińska-Fijałkowska, A. & Kozyra, J. Technologiczny proces spawania z wykorzystaniem robota przemysłowego. TTS Technika Transportu Szynowego 12/2015, 2707–2710 (2015).
6. Kuśmińska-Fijałkowska, A., Łukasik, Z. & Nowakowski, W. Concept of robotics in manufacturing company. Logistyka 4, 4387–4392 (2015)
7. Kuśmińska-Fijałkowska, A. & Mastepan, A. Security robotic industrial processes. ЗБІРНИК НАУКОВИХ ПРАЦЬ УКРАЇНСЬКОЇ ДЕРЖАВНОЇ АКАДЕМІЇ ЗАЛІЗНИЧНОГО ТРАНСПОРТУ 60–66 (2015).
8. Łukasik, Z., Kuśmińska-Fijałkowska, A. & Nowakowski, W. Europe's energy efficiency requirements for household appliances. Przegląd Elektrotechniczny 91, 194–196 (2015).
9. Łukasik, Z., Kozyra, J. & Kuśmińska-Fijałkowska, A. Efektywne ograniczanie zużycia energii elektrycznej w zakładach przemysłowych. TTS Technika Transportu Szynowego 12/2015, 2702–2706 (2015).
10. Kuśmińska-Fijałkowska, A., and Z. Łukasik. "Efekty wynikające z wdrożenia Systemu Zarządzania J ścią." *Logistyka* (2014).Pniewska, B. & Pniewski, R. Logika rozmyta w sterowaniu robotem LEGO. TTS Technika Transportu Szynowego 2917–2923 (2013).
11. Kozyra J. Produkcja energii elektrycznej z odnawialnych nośników energii, Odnawialne Źródła Energii,,ISBN 978-83-7789-175-9,39-47,2013,Instytut Technologii i Eksploatacji - PIB Radom
12. Zakrzewski, B., and D. Zakrzewska. "Przedsiębiorstwo produkcyjne i procesy realizowane w sferze produkcji." Logistyka (2014).
13. Bozek, Pavol; Ivandic, Zeljko; Lozhkin, Alexander; et al.. Solution to the characteristic equation for industrial robot's elliptic trajestories TEHNICKI VJESNIK-TECHNICAL GAZETTE Volume: 23 Issue: 4 Pages: 1017-1023 Published: JUL-AUG 2016
14. Przerembel S., Krzyszkowski A.: Organizacja transportu w usłudze logistycznej zakładu produkcyjnego Logistyka 3/2012
15. Kozyra, Jacek. "Monitorowanie i diagnozowanie uszkodzeń w procesach wytwarzania i przesyłu energii elektrycznej." *Logistyka* 3, CD 1 (2015): 2415-2425.

Adaptation of robot application for printing in the process of production of final product

In this paper the algorithm and application software for printing in the manufacture of the final product based on a robot Kawasaki. As a result of adaptation of the robot application process printing is noticeably accelerated, resulting in increased daily count-by operations. The proposed concept increases the efficiency, the quality of the processes. Taken in the direction of research work it is current and can be interesting for industry

Autorzy:

Prof. dr hab. inż. **Zbigniew Łukasik** – Uniwersytet Technologiczno-Humanistyczny w Radomiu Wydział Transportu i Elektrotechniki.

dr inż. **Aldona Kuśmińska-Fijałkowska** Uniwersytet Technologiczno-Humanistyczny w Radomiu Wydział Transportu i Elektrotechniki

dr inż. **Jacek Kozyra** Uniwersytet Technologiczno-Humanistyczny w Radomiu Wydział Transportu i Elektrotechniki.