Biuletyn WAT
Vol. LIX, Nr 4, 2010

Projekt aplikacji wspierających konfigurowanie procesu paletyzacji kartonów dla robotów przemysłowych firmy Kawasaki i metody ich testowania

WOJCIECH KACZMAREK, MARCIN MISIEJUK[1]

Wojskowa Akademia Techniczna, Wydział Mechatroniki, Instytut Systemów Mechatronicznych,
00-908 Warszawa, ul. S. Kaliskiego 2
[1]ASTOR, 00-739 Warszawa, ul. Stępińska 22/30

Streszczenie. W artykule przedstawiono projekt aplikacji serwera wspierającego konfigurowanie procesu paletyzacji kartonów dla robotów przemysłowych firmy Kawasaki. Zaprezentowano rozwiązanie w oparciu o protokół UDP (ang. *User Datagram Protocol*) skupiając się przede wszystkim na omówieniu sposobów testowania poprawności aplikacji, tworzonych dla robotów przemysłowych na potrzeby zautomatyzowanych stanowisk produkcyjnych. W artykule poruszono również problem ograniczeń środowiska PC-Roset do programowania robotów w trybie off-line przy tego typu działaniach.
Słowa kluczowe: roboty przemysłowe firmy Kawasaki, proces paletyzacji, protokół komunikacji UDP, programowanie robotów przemysłowych
Symbole UKD: 681.51

1. Wstęp

Podobnie jak mechatronika, robotyka jest dynamicznie rozwijającą się, interdyscyplinarną dziedziną, łączącą wiedzę i umiejętności z różnych dyscyplin naukowych, szczególnie z mechaniki, elektroniki i informatyki. Zagadnienia tej naukowo-technicznej dyscypliny obejmują zarówno rozważania teoretyczne, jak i działania praktyczne związane szczególnie ze sterowaniem obiektami oraz procesami technologicznymi. Dzisiaj trudno wyobrazić sobie świat bez automatów, tak w prywatnym życiu ludzi, jak i w sferze przemysłowej [5].

Korzyści wynikające z automatyzacji i robotyzacji zauważono już w latach 60., kiedy to po raz pierwszy w fabryce Forda, do przenoszenia ciężkich elementów, użyto robota o nazwie UNIMATE. Wprowadzenie robotów do przemysłu rozwiązało wiele problemów, począwszy od ekonomicznych (zastąpienie ludzi pracującymi 24 godziny na dobę robotami), a skończywszy na socjologicznych (żądania związków zawodowych). Rosnąca konkurencja spowodowała konieczność obniżenia kosztów produkcji oraz wymusiła wzrost jakości i niezawodności produktów. Jedynym sposobem osiągnięcia tych celów stała się automatyzacja linii produkcyjnych fabryk. Dziś robotyka rozwija się lawinowo i samodzielnie działające, inteligentne maszyny nie są jedynie elementem ludzkiej wyobraźni, ale działającymi urządzeniami. Wyręczają one człowieka w coraz większym zakresie, powodując znaczny wzrost bezrobocia, szczególnie wśród robotników. Przewiduje się, że w najbliższym okresie zmniejszenie zatrudnienia, będące następstwem automatyzacji i robotyzacji przemysłu, spowoduje około 30% wzrost liczby bezrobotnych w krajach wysoko rozwiniętych [5].

Lawinowo rozwijająca się technika komputerowa, zwłaszcza w dziedzinie automatyki i robotyki, wskazuje na celowość poszukiwania nowych rozwiązań dla środowisk programistycznych w zastosowaniach przemysłowych. Rozwiązania takie powinny zwiększać możliwości przeprogramowywania robotów przemysłowych przy jednoczesnym upraszczaniu ich obsługi.

Idąca w dziesiątki, a nawet setki liczba robotów na poszczególnych etapach produkcji spowodowała, iż przestawienie się fabryki na nowy asortyment jeszcze do niedawna wiązało się z zatrzymaniem całego cyklu produkcyjnego na długie okresy czasowe, co powodowało z kolei znaczne straty. Główni światowi producenci robotów przemysłowych (m.in.: ABB, FANUC, KUKA, Kawasaki, Mitsubishi) wraz z dynamicznym wzrostem liczby robotów na zautomatyzowanych liniach produkcyjnych zauważyli potrzebę opracowania i rozwijania środowisk umożliwiających programowanie robotów w trybie off-line. Wszyscy liczący się producenci proponują systemy, dzięki którym w przestrzeni wirtualnej daje się zaprojektować całe linie technologiczne, co znacznie minimalizuje czas przestoju fabryk, zwłaszcza przy projektowaniu nowych linii produkcyjnych. Przykładami tego typu środowisk mogą być:

— RobotStudio firmy ABB,
— RoboGuide firmy FANUC,
— Cosimir firmy Mitsubishi,
— Kuka SimPro firmy KUKA.

Dostępne na rynku rozbudowane środowiska programistyczne nie zaspokajają jednak wszystkich potrzeb potencjalnych odbiorców. W pierwszej kolejności należy zaznaczyć, iż nie są to aplikacje tanie, co powoduje, że małe i średnie przedsiębiorstwa nie decydują się na ich zakup. Po drugie są to systemy bardzo rozbudowane i tylko specjalistyczne szkolenia, które również nie należą do tanich, pozwalają na pełne ich wykorzystanie. I po trzecie, środowiska te nie są powszechnie używane ze względu na ich „młody wiek" (są to produkty stosunkowo nowe). Należy mieć nadzieję, że w nie-

dalekiej przyszłości absolwenci uczelni technicznych (również polskich) będą potrafili z nich korzystać. Już dzisiaj na Wydziale Mechatroniki Wojskowej Akademii Technicznej studenci zapoznają się m.in. ze środowiskami RobotStudio firmy ABB, RoboGuide firmy FANUC, Cosimir firmy Mitsubishi oraz Pc-Roset firmy Kawasaki.

2. Proces paletyzacji a problem krótkich serii produkowanego asortymentu

Paletyzacja jest procesem polegającym na ustawianiu na palecie w ustalonej konfiguracji przedmiotów, które pobierane są z bufora linii produkcyjnej. Paleta po zapełnieniu jest transportowana do magazynu, a na jej miejscu ustawiana jest kolejna, pusta paleta, po czym cały cykl jest powtarzany. Rolę paletyzatora może pełnić maszyna (najczęściej robot przemysłowy) lub człowiek, a sama paletyzacja może przebiegać na wiele sposobów [5].

Najprostsza jest paletyzacja ręczna, bez użycia urządzeń wspomagających. Jest to jednak sposób mało efektywny, główne ze względu na monotonię procesu oraz niejednokrotnie znaczną masę paletyzowanych detali. W wyniku tych niekorzystnych czynników, taki sposób paletyzacji występuje zazwyczaj w słabo rozwiniętych przedsiębiorstwach cechujących się małą produkcją.

W rozwijających się małych i średnich przedsiębiorstwach do paletyzacji wdrażane są roboty przemysłowe, najczęściej portalowe lub przegubowe. Dzięki tego typu rozwiązaniom, paletyzacja przebiega w sposób ciągły, ekonomiczny i bezpieczny. Wykorzystanie elastycznych robotów przemysłowych daje możliwość zmiany specyfikacji paletyzowanych detali (w pewnym zakresie może zmienić się ich masa oraz wymiary geometryczne). Jest to dużym atutem dla firm o zmieniającej się okresowo produkcji.

Przy doborze urządzeń paletyzujących należy zwrócić uwagę nie tylko na wielkość produkcji w danym przedsiębiorstwie, ale również na ilość dostępnej na hali produkcyjnej powierzchni. Jeżeli dostępna powierzchnia jest duża, wtedy można ten czynnik pominąć, jednak w przypadku, gdy jest stosunkowo mała, należy przeanalizować wszystkie możliwości i wybrać najkorzystniejszą. Największy obszar zajmują roboty portalowe, z uwagi na wymaganą przez ich urządzenia peryferyjne dodatkową przestrzeń [4]. W przypadku znacznych ograniczeń powierzchniowych, rozwiązaniem jest zaprojektowanie urządzeń specjalistycznych lub użycie robota albo zespołu robotów przegubowych. Wybór pomiędzy powyższymi wariantami powinien być dokonany w oparciu o wiele innych czynników (koszty, funkcjonalność, obsługiwalność itp.).

Obecnie, z uwagi na częstą produkcję krótkich serii, bardzo ważną cechą urządzeń paletyzujących stał się czas niezbędny na przystosowanie ich do nowej produkcji. W przypadku robotów przemysłowych czas ten jest minimalizowany przez zastosowanie środowisk do programowania robotów w trybie off-line

(np.: firma Kawasaki udostępnia środowisko PC-Roset). Jednak, jak wcześniej wspomniano, alternatywa ta nie wydaje się być najlepsza, zwłaszcza dla małych i średnich przedsiębiorstw.

Celowe wydaje się prowadzenie badań nad aplikacjami dedykowanymi wybranym, najczęściej robotyzowanym procesom technologicznym (np.: spawanie, paletyzacja, klejenie). Głównym celem takich poszukiwań powinno być zwiększenie wydajności i dochodowości linii produkcyjnych, przy jednoczesnym skróceniu czasu szkoleń operatorów oraz uproszczeniu samych środowisk programistycznych. Celowość takiego rozumowania jest wyraźnie widoczna w dobie rosnącej konkurencji, która niesie za sobą konieczność oferowania krótkich serii produkowanych artykułów (np.: zmiana kształtu butelek czy opakowań, etykiet, drobne modyfikacje wyglądu i funkcjonalności, dodawanie nowych modułów do istniejącego produktu), a tym samym częste modyfikacje uruchomionego cyklu produkcyjnego [2, 4, 5].

Obserwuje się, iż czołowe na rynku światowym firmy produkujące roboty przemysłowe (m.in.: ABB, FANUC, Mitsubishi) zdają się podążać w tym kierunku. Większość z nich proponuje obok bardzo rozbudowanych i posiadających ogromne możliwości środowisk do programowania robotów w trybach off-line, aplikacje wspierające konkretne procesy technologiczne (np.: firma ABB ma w swojej ofercie m.in. aplikacje: Arc Welding, Spot Welding, Die Casting, Assembly). Ciągle jednak można znaleźć procesy technologiczne, dla których takie oprogramowanie nie jest oferowane, a które są bardzo popularne na zautomatyzowanych liniach produkcyjnych. Tworzenie takich aplikacji można podzielić na trzy etapy:

— opracowanie właściwego środowiska z odpowiednim interfejsem użytkownika (może być zrealizowane w dowolnym języku programowania),
— opracowanie nowego lub wykorzystanie istniejącego protokołu transmisji, umożliwiającego przesłanie niezbędnych do poprawnego funkcjonowania robota parametrów do jego kontrolera i przygotowanie aplikacji umożliwiającej przepływ danych w tym protokole,
— opracowanie elastycznej aplikacji w języku programowania danego robota i zaimplementowanie jej w jego kontrolerze.

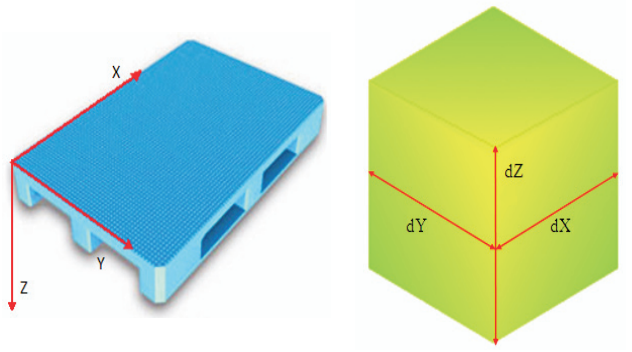
3. Opis położeń oraz ruchów wykonywanych przez robota w procesie paletyzacji

Przestrzeń robocza robota przemysłowego w przypadku zastosowania w paletyzacji jest definiowana przez dwie pozycje w układzie przestrzennym oraz wymiary palety. Każda pozycja musi być zdefiniowana przez sześć współrzędnych P(X, Y, Z, O, A, T), gdzie:

— X — przesunięcie na osi X,
— Y — przesunięcie na osi Y,

— Z — przesunięcie na osi Z,
— O — obrót wokół osi X,
— A — obrót wokół osi Y,
— T — obrót wokół osi Z.

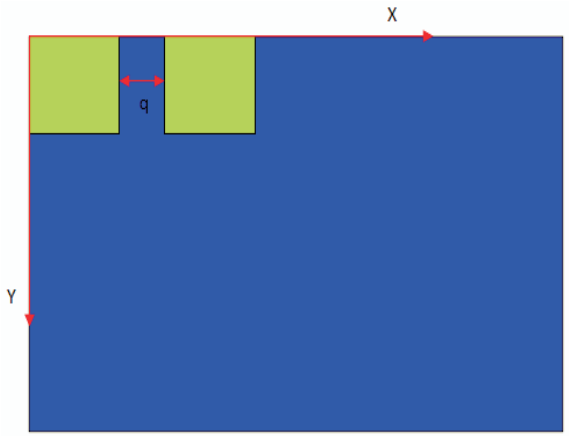
Pierwszą z wyżej wymienionych pozycji jest punkt uchwycenia przedmiotu — obiektu manipulacji (pu). Robot, aby pobrać obiekt, zawsze ustawia się w tej pozycji. Drugą z pozycji jest punkt umiejscowienia początku układu współrzędnych związanego z paletą (po) o współrzędnych (0, 0, 0, 0, 0, 0) w układzie kartezjańskim (rys. 1). Pozostałe punkty, w których obiekty odkładane są na palecie, zapisywane są jako przesunięcie punktu po i generowane są automatycznie przez oprogramowanie. Przesunięcie to wynika z wymiarów obiektu, odstępów pomiędzy obiektami oraz algorytmu, według którego obiekty są paletyzowane.



Rys. 1. Definicje: układu współrzędnych związanego z paletą oraz wymiarów paletyzowanego detalu

Nie ma konieczności deklarowania tego układu w oprogramowaniu robota, aczkolwiek można utworzyć w nim układ o wyżej wymienionych parametrach przy pomocy funkcji FRAME. W przypadku, gdy nie zostanie zdefiniowany układ skojarzony z paletą, wszystkie punkty będą zapisywane w pamięci programu w identyczny sposób, jednak robot będzie się poruszał w układzie związanym z narzędziem (TOOL). Jest to pewne utrudnienie podczas tworzenia programu, ponieważ układ współrzędnych związany z narzędziem nie pokrywa się z układem związanym z paletą i jego ustawienie względem układu bazowego zależne jest od ustawienia narzędzia. Jedyną zaletą tego rozwiązania jest brak konieczności definiowania dodatkowego układu współrzędnych. Dalszy opis, dla uproszczenia, będzie przedstawiony w oparciu o układ współrzędnych związany z paletą.

Jeżeli kolejny obiekt manipulacji będzie układany obok poprzedniego (rys. 2) z odstępem q oraz tak, aby obydwa obiekty leżały wzdłuż osi X układu związanego z paletą (osie X i dX będą się pokrywały), wtedy punkt odłożenia kolejnego obiektu będzie zapisany jako $Po_2 = (Dx + q, 0, 0, 0, 0, 0)$. Niekiedy, jeżeli zachodzi taka potrzeba, jed-

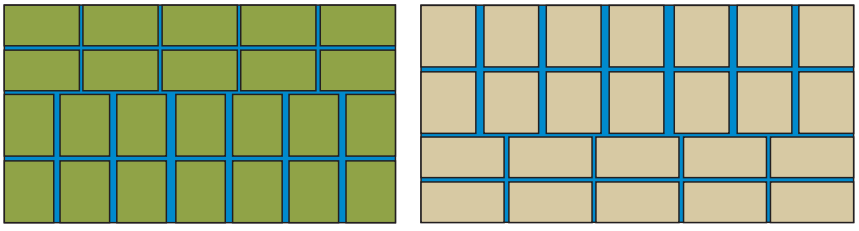


Rys. 2. Konfiguracja obiektów na palecie

norazowo paletyzowanych jest kilka obiektów (wymaga to zastosowania specjalnych chwytaków oraz odpowiedniego pozycjonowania obiektów przed ich uchwyceniem). Grupa paletyzowanych jednorazowo obiektów jest traktowana jako jeden obiekt.

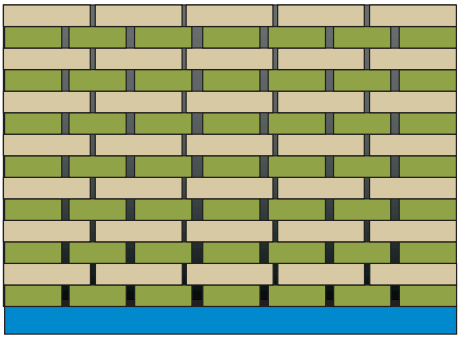
Paletyzowane obiekty powinny być rozmieszczone na palecie optymalnie, czyli tak aby maksymalnie wykorzystać dostępną na niej przestrzeń oraz zminimalizować ryzyko uszkodzenia. W tym celu definiowane są odstępy pomiędzy poszczególnymi obiektami paletyzacji, ich sposób rozmieszczenia na palecie oraz liczba warstw. Najczęściej konfiguruje się położenia obiektów dla dwóch warstw, które są następnie na przemian powielane (warstwy parzyste i nieparzyste — rys. 3). Aby zapełniona paleta była stabilna, należy tak skonfigurować obydwie warstwy, żeby krawędzie przedmiotów leżących na warstwie parzystej w jak najmniejszym stopniu pokrywały się z krawędziami przedmiotów leżących na warstwie nieparzystej.

Po sprawdzeniu poprawności konfiguracji warstw należy ocenić liczbę warstw, które można ułożyć na palecie. Należy wziąć pod uwagę wytrzymałość przedmiotów oraz ich ciężar. Im stosunek ciężaru do wytrzymałości jest mniejszy, tym ilość warstw na palecie może być większa. Przy paletyzacji niektórych środków chemicznych należy dodatkowo wziąć pod uwagę ciśnienie powstałe w pierwszej warstwie pod



Rys. 3. Przykładowy widok warstw: nieparzystych i parzystych (widok z góry)

wpływem nacisku pozostałych warstw, ponieważ może dojść do wzrostu temperatury, a w rezultacie do zainicjowania reakcji chemicznej (np. spalania). Przykładowe ułożenie warstw na palecie (bez przekładek) przedstawiono na rysunku 4.



Rys. 4. Przykładowy widok zapełnionej palety (widok z boku)

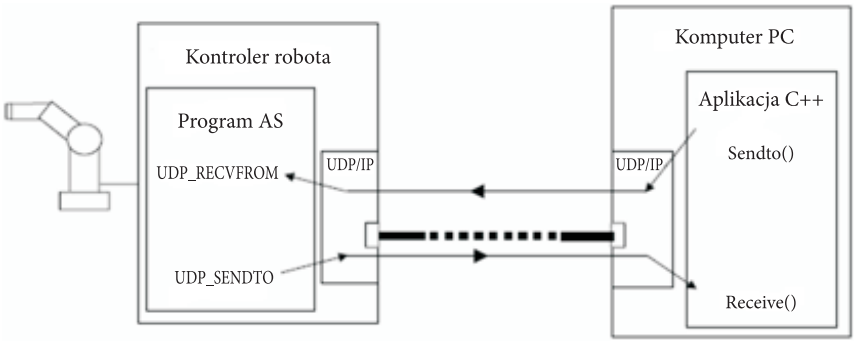
Niekiedy, przy paletyzacji przedmiotów, których nie możemy skonfigurować w warstwie w wyżej opisany sposób lub gdy są to przedmioty o małej wytrzymałości na uszkodzenia mechaniczne, między warstwami wykorzystywane są przekładki (papierowe lub z tworzywa sztucznego). Zwiększają one stabilność palety oraz zmniejszają ryzyko uszkodzenia spaletyzowanych przedmiotów.

4. Projekt aplikacji umożliwiających komunikowanie się komputera PC z robotem Kawasaki

Do skomunikowania komputera PC z robotem przemysłowym posłużono się protokołem UDP (*User Datagram Protocol*). Protokół ten nie ustanawia połączenia logicznego pomiędzy węzłami sieci na potrzeby transmisji. Dane są nadawane do wszystkich węzłów sieci, jednak odbiera je ten węzeł, którego adres jest zgodny z adresem odbiorcy znajdującym się w pakiecie. Na rysunku 5 przedstawiono ogólny schemat komunikacji komputera PC z robotem przemysłowym firmy Kawasaki przy użyciu protokołu UDP [1].

W celu odebrania danych przychodzących z komputera PC do kontrolera robota, opracowano stosowane programy komputerowe i zaimplementowano je na obu urządzeniach. Po stronie robota zaimplementowano aplikację serwera (hosta) i to właśnie przedstawienie jej budowy jest zasadniczym celem niniejszego artykułu, natomiast po stronie komputera PC aplikację klienta.

Wykorzystująca protokół UDP aplikacja serwera została napisana w języku AS (język programowania robotów Kawasaki) i jest oparta o procedurę UDP_RECVFROM [6, 7].



Rys. 5. Ogólny schemat komunikacji komputera PC z robotem Kawasaki przy użyciu protokołu UDP

4.1. Aplikacja klienta

Aby przesłać dane z komputera PC do robota w oparciu o protokół UDP, opracowano aplikację klienta. Aplikacja ta jest odpowiedzialna za wysłanie odpowiednio przygotowanych danych (parametrów robota niezbędnych do realizacji procesu paletyzacji) na określony port robota o zadanym adresie IP. Danymi wejściowymi są zmienne wygenerowane przez aplikację graficzną oraz zmienne wprowadzone przez użytkownika. Użytkownik podaje zmienne dotyczące połączenia, czyli adres IP robota i numer portu, oraz prędkość robota wyrażoną w procentach. Funkcja doboru prędkości robota jest wprowadzona w celu umożliwienia przetestowania wygenerowanej trajektorii ruchu w zwolnionym tempie, ze względu na warunki bezpieczeństwa. Po przeprowadzeniu testu można zwiększyć prędkość i na nowo wysłać dane do robota. Opis danych wejściowych przedstawiono w tabeli 1. W związku z tym, iż parametry robota udostępniane są przez aplikację napisaną w środowisku Borland C++ Builder 6, również w tym przypadku posłużono się właśnie tym środowiskiem [3, 8].

Tabela 1

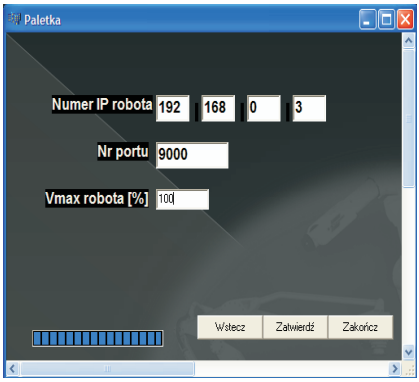
Dane wejściowe aplikacji klienta

Opis zmiennej	Oznaczenie zmiennej	Format zmiennej
Adres IP robota	IPAdres	AnsiString
Numer portu	Port	AnsiString
Prędkość robota	cpredkosc	AnsiString
Ilość kartonów w warstwie nieparzystej	ik	Integer
Ilość kartonów w warstwie parzystej	ik1	Integer

cd. tabeli 1

Ilość warstw	il_w	Integer
Wysokość kartonu	wysokosc_kartonu	Integer
Numer offsetu	Offset	Integer
Zmienna określająca ilość warstw, po których występuje przekładka	przekladki	Integer
Wysokość przekładki	wys_przekladki	Integer
Długość kartonu	dlugosc_kartonu	Integer
Szerokość kartonu	szerokosc_kartonu	Integer
Współrzędne X kartonów warstwy nieparzystej	Iksy[0…ik]	Integer
Współrzędne Y kartonów warstwy nieparzystej	Igreki[0…ik]	Integer
Obroty wokół osi Z kartonów warstwy nieparzystej	Obroty[0…ik]	Integer
Współrzędne X kartonów warstwy parzystej	Iksy1[0…ik1]	integer
Współrzędne Y kartonów warstwy parzystej	Igreki1[0…ik1]	Integer
Obroty wokół osi Z kartonów warstwy parzystej	Obroty1[0…ik1]	Integer

Na rysunku 6 przedstawiono okno dialogowe zawierające funkcje umożliwiające podanie adresu IP, numeru portu i prędkości robota.



Rys. 6. Widok jednego z okien dialogowych aplikacji komunikacyjnej klienta

Efektem zadziałania aplikacji klienta jest wygenerowanie niezbędnych do poprawnej pracy robota przemysłowego danych (tab. 2), zapisanie ich do tablicy „bufor” oraz wysłanie do kontrolera robota.

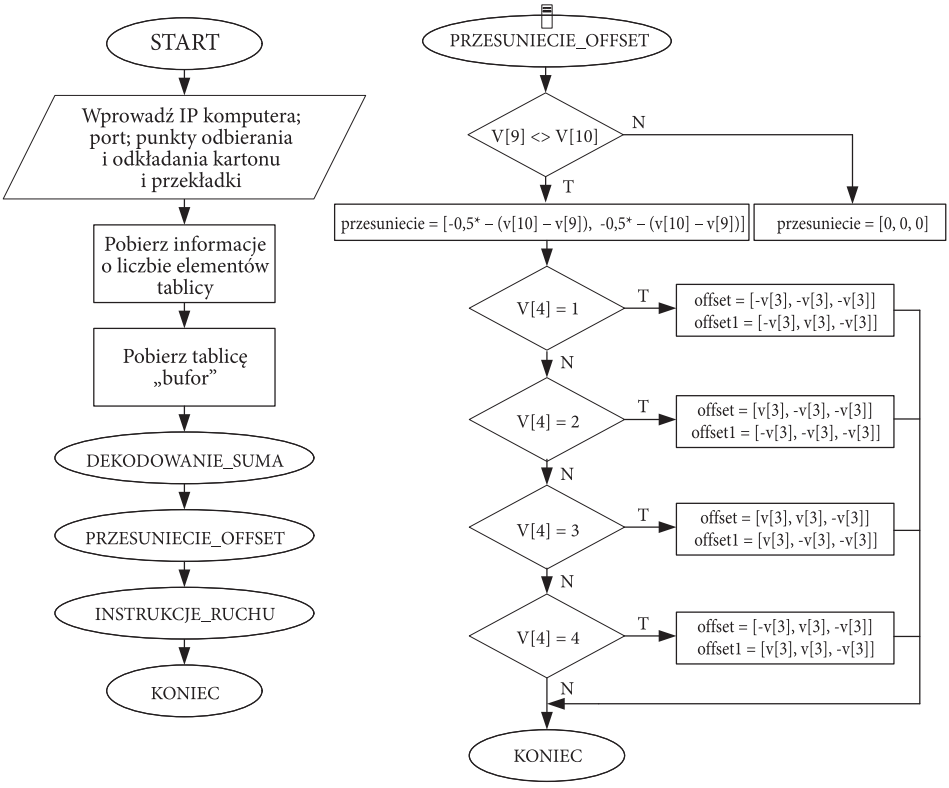
Tabela 2

Zawartość tablicy „bufor”

Opis zmiennej	Oznaczenie zmiennej	Ilość znaków
Liczba kartonów w warstwie nieparzystej	cn	4
Liczba kartonów w warstwie parzystej	cp	4
Liczba warstw	cw	4
Wysokość kartonu	cz	4
Numer offsetu	coffset	4
Prędkość robota	ccpredkosc	4
Zmienna określająca ilość warstw, po których występuje przekładka	cprzekladka	4
Wysokość przekładki	czpr	2
Suma kontrolna	csuma_kontr	6
Długość kartonu	cdx	4
Szerokość kartonu	cdy	4
Puste pole [0000]	--	4
Współrzędne X kartonów warstwy nieparzystej	cxn[0…ik]	4
Współrzędne Y kartonów warstwy nieparzystej	cyn[0… ik]	4
Obroty wokół osi Z kartonów warstwy nieparzystej	cktn[0… ik]	4
Współrzędne X kartonów warstwy parzystej	cxp[0… ik1]	4
Współrzędne Y kartonów warstwy parzystej	cyp[0… ik1]	4
Obroty wokół osi Z kartonów warstwy parzystej	cktp[0… ik1]	4

4.2. Aplikacja serwera

Aplikacja serwera (rys. 7) została opracowana w języku AS (język programowania robotów Kawasaki). Danymi wejściowymi są dane wprowadzane przez użytkownika (IP komputera PC, numer portu i cztery punkty bazowe) oraz dane wysyłane z aplikacji klienta. Punkty, które wprowadza użytkownik, to punkt chwytania kartonu (#pu), punkt początkowy układu związanego z paletą (po) oraz opcjonalnie punkty chwytania i odkładania przekładki (#pu_pr i po_pr). Dane wysłane z komputera PC odbierane są poprzez wywołanie procedury UDP_RECVFROM. Po odebraniu wiadomości o długości komunikatu gniazdo sieciowe otwierane jest ponownie w celu odebrania danych. Po odebraniu komunikat zapisany jest do tablicy $cnt. Długość elementu tablicy to piętnaście znaków, więc do każdego elementu tablicy zapisany



Rys. 7. Uproszczony algorytm aplikacji serwera oraz funkcji przesuniecie_offset

zostanie ciąg znaków odpowiadający wartościom kolejnych trzech zmiennych przedstawionych w tabeli 3. W każdym z elementów tablicy $cnt[0] znajdują się trzy separatory (przecinki), które oddzielają od siebie trzy elementy znajdujące w każdym z ciągów. Elementy tablicy $cnt przedstawione są w tabeli 3.

Tabela 3

Zawartość elementów tablicy $cnt[0]

Element tablicy	Zmienne przechowywane w elemencie
$cnt[0]	cn, cp, cw,
$cnt[1]	cz, offset, cpredkosc,
$cnt[2]	cprzekladka, czpr, csuma_kontr,
$cnt[3]	cdx, cdy, 0000,
$cnt[4…cn]	cxn[4…cn], cyn[4…cn], cktn[4…cn],
$cnt[cn+4…cn+cp+4]	cxp[cn + 4…cn + cp + 4], cyp[cn + 4…cn + cp + 4], cktp[cn + 4… cn + cp + 4],

Po odebraniu komunikatu tablica $cnt, do której zapisane są dane jest dekodowana i dane w niej zawarte są rzutowane ze zmiennych ciągu znaków na zmienne rzeczywiste. Odbywa się to w dwu krokach. W pierwszym kroku dekodowane są elementy tablicy $cnt o numeracji od 0 do 3, i zapisywane są jako tablica v[0…11]. Opis zmiennych zawartych w tablicy „v" przedstawia tabela 4. Następnie dekodowana jest część tablicy $cnt, w której zawarte są współrzędne kartonów (ilość tych elementów jest równa liczbie kartonów). Dane zawarte w tych elementach tablicy zapisywane są do trzech tablic (x[0…v[1] + v[2]], y[0…v[1] + v[2]], kt[0…v[1] + v[2]]). Następnie sprawdzana jest suma kontrolna, wyznaczane są punkty pośrednie i trajektoria robota.

Tabela 4

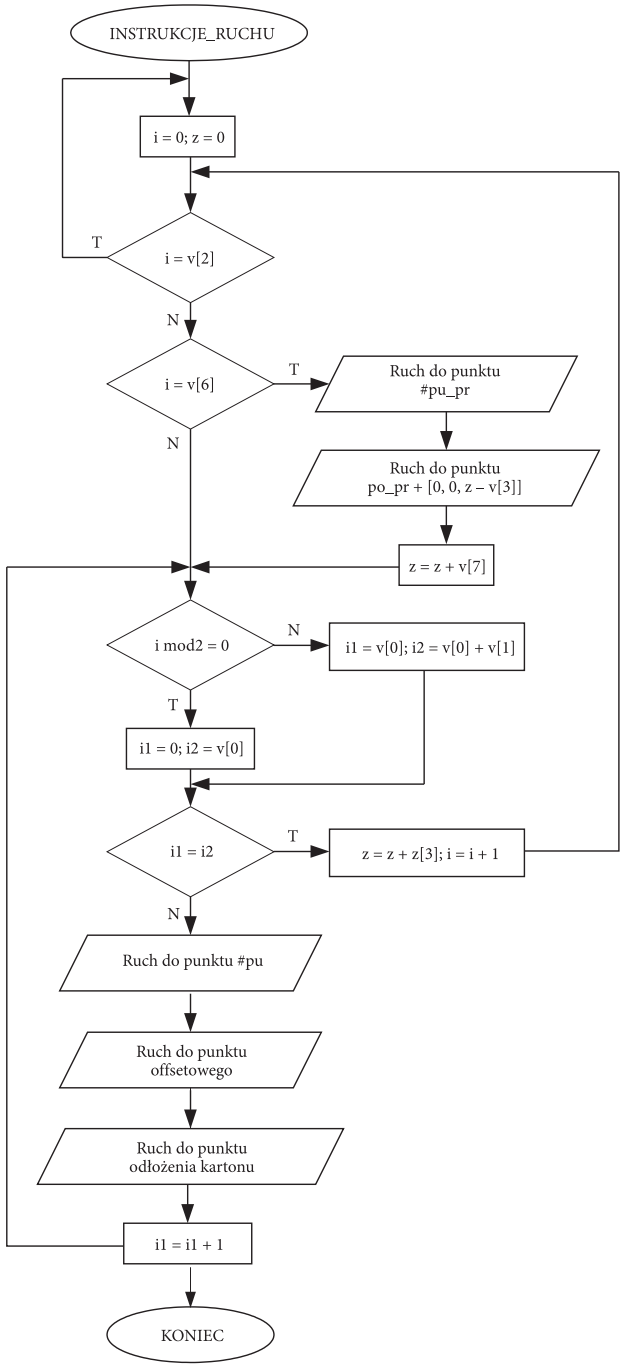
Zmienne tablicy „v"

Opis zmiennej	Nazwa zmiennej w programie robota	Nazwa zmiennej w aplikacji klienta
Liczba kartonów w warstwie nieparzystej	v[0]	cn
Liczba kartonów w warstwie parzystej	v[1]	cp
Liczba warstw	v[2]	cw
Wysokość kartonu	v[3]	cz
Numer offsetu	v[4]	coffset
Prędkość robota	v[5]	ccpredkosc
Ilość warstw, po których występuje przekładka	v[6]	cprzekladka
Wysokość przekładki	v[7]	czpr
Suma kontrolna	v[8]	csuma_kontr
Długość kartonu	v[9]	cdx
Szerokość kartonu	v[10]	cdy
Puste pole [0000]	v[11]	--

W wyniku dekodowania (funkcja DEKODOWANIE_SUMA) wszystkie dane zawarte w tablicy $cnt zostają wyodrębnione i zamienione ze zmiennych tekstowych na wartości rzeczywiste, a następnie zsumowane w celu sprawdzenia sumy kontrolnej.

W wyniku działania funkcji PRZESUNIECIE_OFFSET (rys. 7) zostaje przesunięty punkt chwytania przedmiotu oraz wyznaczone są pozycje pośrednie efektora robota. Przesunięcie punktu chwytania kartonu wymagane jest z uwagi na to, że współrzędne kartonów generowane przez aplikację graficzną są pozycjami względem lewego górnego rogu kartonu. Przesunięcie to dotyczy wyłącznie kartonów obróconych o 90 stopni.

W wyniku działania funkcji INSTRUKCJE_RUCHU (rys. 8) wyznaczane są kolejne pozycje punktu odłożenia kartonu (po) i przekładki (po_pr) oraz generowana

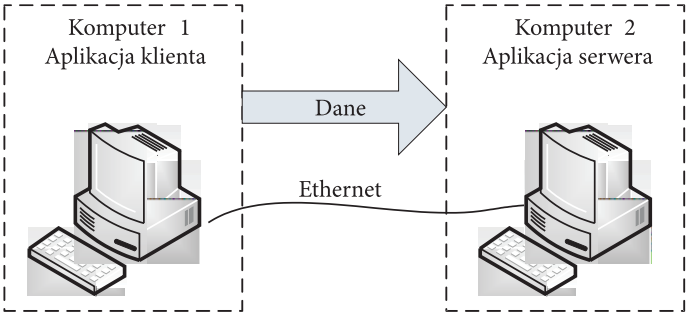


Rys. 8. Algorytm funkcji instrukcje_ruchu

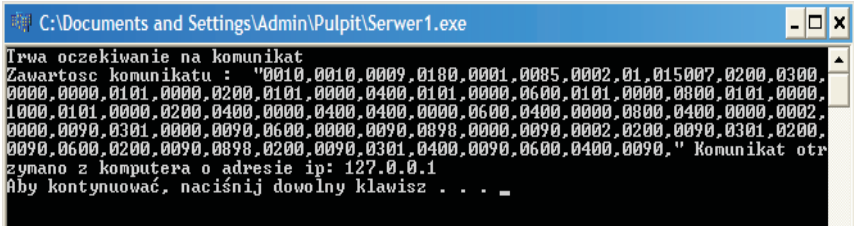
jest trajektoria ruchu robota na podstawie wyznaczonych punktów. Wyznaczane jest również położenie punktu offsetowego. Wszystkie pozycje wyznaczane są w każdej iteracji, a algorytm został opracowany z uwzględnieniem optymalizacji trajektorii oraz bezpiecznego przejścia pomiędzy poszczególnymi punktami węzłowymi.

4.3. Testowanie oprogramowania

W pierwszej fazie testowania aplikacja serwera została opracowana w środowisku Borland C++ Builder 6. Takie rozwiązanie umożliwiło uruchomienie jej na komputerze PC, należącym do tej samej sieci, do której należał komputer z aplikacją klienta (rys. 9 i 10) i przetestowanie całości oprogramowania bez konieczności podłączania rzeczywistego robota. Aby uruchomić aplikację serwera i klienta na jednym komputerze, należy wpisać 127.0.0.1 jako adres IP robota w aplikacji klienta. Jako numer portu należy wpisać 2000. Po tak przeprowadzonych próbach, do testów wykorzystano środowisko do programowania robotów Kawasaki w trybie off-line — PC-Roset.



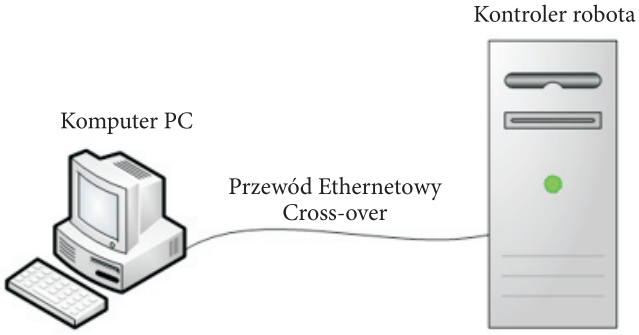
Rys. 9. Komunikacja pomiędzy dwoma komputerami PC



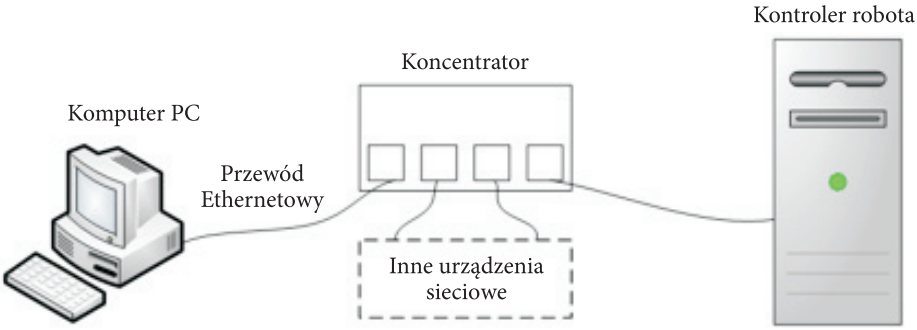
Rys. 10. Wizualizacja przykładowego komunikatu wysłanego z aplikacji klienta

4.4. Testowanie oprogramowania przy wykorzystaniu rzeczywistego robota Kawasaki

Pełne przetestowanie proponowanej aplikacji przeprowadzono przy użyciu robota Kawasaki. Rozważono dwa sposoby połączenia sprzętowego obu urządzeń, połączenie bezpośrednie przy użyciu przewodu typu cross-over (rys. 11) oraz połączenie z wykorzystaniem koncentratora (rys. 12). Do połączenia logicznego natomiast posłużono się terminalem KCwinTCP uruchamianym na komputerze PC (rys. 13).



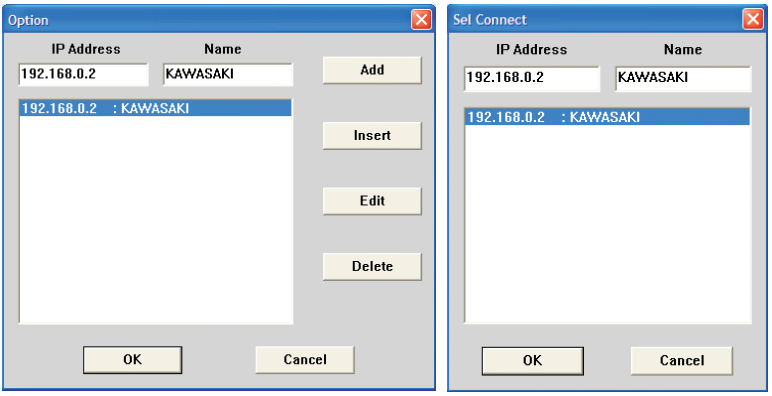
Rys. 11. Połączenie bezpośrednie



Rys. 12. Połączenie przy użyciu koncentratora

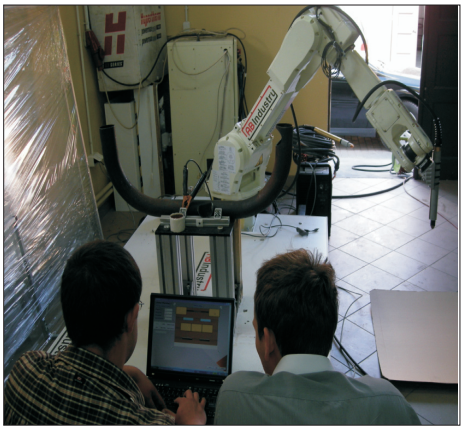
KCwinTCP jest oprogramowaniem dostarczanym wraz z robotem przemysłowym Kawasaki. Pozwala ono na programowanie robota z poziomu komputera PC, sterowanie robotem przy użyciu odpowiednich instrukcji, tworzenie oraz edycję programów, operacje kopiowania programów z komputera PC do kontrolera robota i odwrotnie. Wykorzystanie terminala KCwinTCP jest alternatywną metodą sprawowania kontroli nad robotem, a wszystkie zawarte w nim funkcje są dostępne z poziomu panelu operatorskiego [9].

Komputer PC, na którym uruchomiony zostanie terminal i kontroler robota, musi należeć do tej samej klasy numerów IP.



Rys. 13. Widok okien opcji oraz wyboru robota terminala KCwinTCP

Test polegał na skonfigurowaniu kartonów na palecie za pomocą aplikacji graficznej i wysłaniu danych wygenerowanych na tej podstawie do kontrolera robota (rys. 14). Testy oprogramowania potwierdziły założenie, iż przeprogramowanie robota w przypadku zmiany parametrów paletyzacji jest dużo szybsze i łatwiejsze oraz że nie jest do tego działania konieczna znajomość programowania robota Kawasaki w języku AS.



Rys. 14. Testowanie opracowanych aplikacji

5. Podsumowanie

W artykule przedstawiono wybrane zagadnienia związane z problematyką tworzenia aplikacji mających na celu uproszczenie i skrócenie czasu przeprogramowywania robotów przemysłowych w konkretnych zastosowaniach.

Tworzenie tego typu oprogramowania podzielono na trzy odrębne etapy:

— opracowanie właściwego środowiska z odpowiednim interfejsem użytkownika (może być zrealizowane w dowolnym języku programowania),
— opracowanie lub wykorzystanie istniejącego protokołu transmisji, umożliwiającego przesłanie niezbędnych do poprawnego funkcjonowania robota parametrów do jego kontrolera i przygotowanie aplikacji umożliwiającej przepływ danych w tym protokole,
— opracowanie elastycznej aplikacji w języku programowania danego robota i zaimplementowanie jej w jego kontrolerze;

skupiając się w szczególności na etapie trzecim.

Zaproponowane mechanizmy umożliwiają łatwe oraz intuicyjne przeprogramowanie robotów firmy Kawasaki wykonujących proces paletyzacji bez konieczności znajomości ich języka programowania. Główną zaletą takiego rozwiązania jest udostępnienie operatorowi prostego w obsłudze środowiska, skracającego znacznie czas przekonfigurowania robota w przypadku zmiany parametrów paletyzacji (wielkość palety, rozmiary kartonów, liczba warstw itd.). Należy tutaj zaznaczyć, iż wyposażenie zrobotyzowanej komory produkcyjnej do paletyzacji w dodatkowy panel operatorski pozwoli na programowanie robotów bez konieczności używania do tego celu panelu nauczania, co pozwoli na skrócenie czasu szkolenia załogi oraz zabezpieczenie robota przed niechcianymi zmianami konfiguracyjnymi w przypadku, kiedy stanowisko nie jest obsługiwane przez wykwalifikowanego inżyniera.

Opracowana w środowisku Borland C++ Builder 6 aplikacja komunikacyjna klienta podczas przeprowadzania testów została połączona z opracowaną wcześniej aplikacją graficzną (ze względu na obszerność zagadnienia nie została przedstawiona w niniejszym artykule). Dzięki temu nie ma potrzeby uruchamiania dwóch odrębnych programów na komputerze PC. Aplikacja serwera, czyli oprogramowanie kontrolera robota, została opracowana w języku AS, który jest podstawowym narzędziem programowania robotów Kawasaki.

Dodatkowo dla celów testowych opracowana została aplikacja serwera, umożliwiająca odebranie danych wygenerowanych za pomocą aplikacji graficznej dedykowanej procesowi paletyzacji na komputerze PC. Oprogramowanie to, w połączeniu z środowiskiem PC-Roset, do programowania robotów firmy Kawasaki w trybie off-line, może posłużyć do testowania utworzonego oprogramowania oraz komunikacji w sieci LAN. Należy tutaj wspomnieć o braku możliwości bezpośredniego podłączenia zewnętrznej aplikacji do środowiska PC-Roset, co jest znacznym ograniczeniem w przypadku tworzenia dedykowanych dla robotów Kawasaki aplikacji. Istnienie takiej możliwości (bezpośredniego podłączenia) pozwoliłoby na pełne testowanie „zewnętrznego oprogramowania" w trybie off-line, bez konieczności podłączania rzeczywistego robota. W obecnej sytuacji testowanie można przeprowadzić, wprowadzając uzyskane z aplikacji klienta dane ręcznie.

Oprogramowanie zostało przetestowane na robocie Kawasaki w firmie AB Industry. Test polegał na skonfigurowaniu kartonów w aplikacji graficznej, a następnie przesłaniu wygenerowanych na tej podstawie danych do kontrolera robota. Sprawdzona została zgodność danych po stronie nadawczej i odbiorczej oraz zgodność trajektorii ruchów robota z trajektorią wynikającą z konfiguracji kartonów w aplikacji graficznej. Test potwierdził prawidłowe działanie opracowanego oprogramowania.

Tworzenie nowych i rozwijanie już istniejących środowisk wspierających programowanie robotów w trybie off-line jest jednym z najprężniej rozwijających się kierunków związanych z rozwojem robotyki. Przy czym nie należy tutaj zapominać o tworzeniu i rozwijaniu specjalistycznych, dodawanych do oprogramowania głównego pakietów oprogramowania, których głównym zadaniem będzie udostępnianie prostych w obsłudze interfejsów zrozumiałych dla operatorów konkretnych stanowisk produkcyjnych, a jednocześnie wprowadzających pewien poziom elastyczności, co w dzisiejszych czasach przy często zmieniającym się produkowanym asortymencie jest niezmiernie ważne.

Artykuł wpłynął do redakcji 28.07.2009 r. Zweryfikowaną wersję po recenzji otrzymano w listopadzie 2009 r.

LITERATURA

[1] *AS Language Reference Manual*, Kawasaki Heavy Industries LTD, 2007.
[2] P. Dutkiewicz, W. Wróblewski, *Modelowanie i sterowanie robotów*, PWN, Warszawa, 2003.
[3] M. Flenov, *C++ Elementarz hakera*, Helion, 2005.
[4] J. Honczarenko, *Roboty przemysłowe. Elementy i zastosowanie*, WNT, Warszawa, 1992.
[5] W. Kaczmarek, *Elementy robotyki przemysłowej*, WAT, Warszawa, 2008.
[6] *PC-Roset Light— Pierwsze Kroki*, ASTOR Sp. z. o.o.
[7] *Programowanie w języku AS*, ASTOR Sp. z. o.o.
[8] A. Stasiewicz, *C++ Builder. Całkiem inny świat*, Helion, 2005.
[9] *TCP/IP Communication Manual*, Kawasaki Heavy Industries LTD 2007.
[10] K. Tomaszewski, *Roboty przemysłowe*, WNT, Warszawa, 1993.

W. KACZMAREK, M. MISIEJUK

Project of applications supporting configuration of palletization process of packets for Kawasaki industrial robots and methods of their testing

Abstract. In this paper, the project of application for supporting configuration of palletization process was presented. The authors touched a very important problem of creating applications for Kawasaki industry robots that give the possibility of the hardware configuration of palletization processes. Moreover, in the article, the authors presented how the information from PC is sent to a robot computer.

Keywords: Kawasaki industrial robots, palletization process, UDP communication, robots programming
Universal Decimal Classification: 681.51