Dr inż. Adam Słota
Politechnika Krakowska, Wydział Mechaniczny
slota@mech.pk.edu.pl

Procedura tworzenia modelu stanowiska w systemie Delmia w celu weryfikacji algorytmu generowania skoordynowanych ruchów robotów

Streszczenie
W pracy przedstawiono algorytm generowania trajektorii dla zadania transportowego realizowanego przez dwa roboty. W celu weryfikacji uzyskanych danych zaproponowano wykorzystanie systemu Delmia. W pracy przedstawiono procedurę dotyczącą przeniesienia danych z systemu LabVIEW do systemu Delmia oraz budowy, na ich podstawie, modelu stanowiska oraz definiowania zadań dla robotów. Etapy pracochłonne procedury zostały zautomatyzowane przy pomocy makr, pozostałe są realizowane manualnie. Opracowany tok postępowania zilustrowano przykładem.

Abstract
In the paper an algorithm of generation trajectories for transport tasks executed by two robots is presented. For verification purposes application of Delmia system is suggested. In the work a procedure aiming at: transferring data from LabVIEW to Delmia, building a model of the robotized cell and robot task definition is described. Time consuming steps of the procedure are automated with the use of macros, others are executed manually. For illustration purposes an example is presented.

1. Wstęp

Programowanie off-line robotów przemysłowych może być realizowane z wykorzystaniem oprogramowania dedykowanego do robotów danego producenta (*RoboGuide*, *RobotStudio*, *PCRoset*). Programy te zapewniają wspomaganie w zakresie możliwości oferowanych przez kontrolery robotów. Umożliwiają programowanie pojedynczych robotów oraz definiowanie punktów synchronizacji programów różnych robotów i/lub urządzeń pomocniczych przez sygnały wejścia/wyjścia. W przypadku kontrolerów z funkcją sterowania dodatkowymi osiami (np. pozycjonera) lub sterowania dwoma robotami, umożliwiają programowanie zadań z ciągłą koordynacją ruchów (systemy *DualArm* firmy Fanuc [1], *MultiMove* firmy ABB [3]).
Innym rozwiązaniem są systemy uniwersalne, takie jak *Delmia*. Dostępność bibliotek modeli robotów różnych producentów, możliwość importowania modeli urządzeń stanowisk zrobotyzowanych, jak również możliwość tworzenia własnych modeli urządzeń pozwala tworzyć wirtualne stanowiska, odzwierciedlające wszystkie istotne geometryczne i kinematyczne cechy stanowisk zrobotyzowanych. Cechy te pozwalają na efektywną pracę w środowisku wirtualnym na etapie opracowywania koncepcji stanowiska, ustalania rozmieszczenia komponentów, definiowania i weryfikacji zadań robotów.
Poszukiwanie nowych rozwiązań, dotyczących ciągłej koordynacji ruchów dwóch robotów sterowanych przez oddzielne kontrolery, wymaga budowy nowych modeli oraz weryfikacji uzyskanych rozwiązań. Wcześniejsze prace autora, krótko scharakteryzowane w rozdziale 2, dotyczyły generowania współrzędnych trajektorii

robotów kartezjańskich, realizujących wspólnie zadanie transportu jednego przedmiotu. Opracowane algorytmy i modele, zaimplementowane w systemie *LabVIEW*, uwzględniały zagadnienia dynamiki układów napędowych manipulatorów [4] oraz weryfikację stabilności [5]. Wykorzystanie uzyskanych wyników do definiowania zadań rzeczywistych robotów wymaga sprawdzenia realizowalności wyznaczonych trajektorii na istniejącym stanowisku lub konfiguracji nowego stanowiska. Do tego celu zaproponowano wykorzystanie systemu *Delmia v5*, dostępnego na Wydziale Mechanicznym Politechniki Krakowskiej. Cechy systemu, takie jak: otwartość, możliwość importu danych zewnętrznych oraz automatyzacji powtarzalnych działań przy pomocy makr, umożliwiają realizację postawionego zadania.
W rozdziale 2 przedstawiono w zarysie algorytm generowania trajektorii manipulatorów kartezjańskich realizujących wspólnie zadanie transportowe. W rozdziale tym zaprezentowano również wyniki działania algorytmu uzyskane w programie *LabVIEW*. W rozdziale 3 zamieszczono tok postępowania przy przenoszeniu danych uzyskanych w programie *LabVIEW* do systemu *Delmia*, budowie na ich podstawie uproszczonego modelu stanowiska, definiowaniu oraz wstępnej weryfikacji zadań robotów.
Podstawowym celem niniejszej pracy jest sprawdzenie możliwości i przedstawienie toku postępowania, mającego na celu wykorzystanie danych zewnętrznych do budowy modelu stanowiska zrobotyzowanego i definiowania zadań robotów w środowisku graficznym *3D*. Przedstawiony w pracy przykład ma charakter ilustracyjny i nie stanowi rozwiązania rzeczywistego zadania.

2. Algorytm generowania trajektorii dla robotów kartezjańskich

Przyjęto, że trajektoria obydwu robotów wyznaczana jest przez ten sam algorytm, na podstawie znajomości aktualnego położenia punktów *TCP* dwóch robotów – jedna instancja algorytmu wyznacza współrzędne punktu *TCP* robota *A*, druga współrzędne punktu *TCP* robota *B*. Dodatkowo założono, że dla algorytmu wyznaczającego trajektorię punktu *TCP* jednego robota nie są dostępne zaprogramowane parametry ruchu (położenie docelowe i prędkość ruchu) punktu *TCP* drugiego robota.
Kolejne punkty trajektorii wyznaczane są jako złożenie dwóch ruchów:

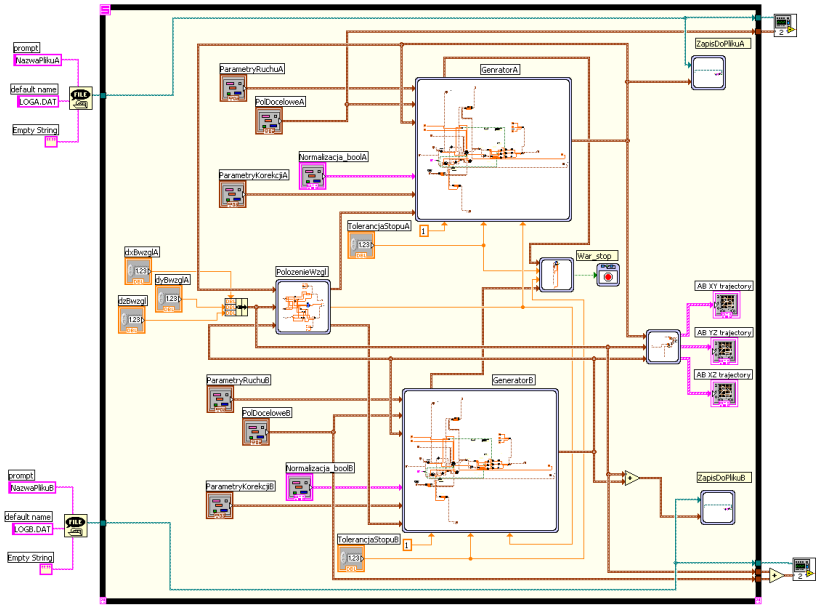
- ruchu z położenia bieżącego do zaprogramowanego położenia docelowego (w aktualnej wersji ruch ten jest realizowany wzdłuż linii prostej), ze stałą zaprogramowaną prędkością,
- ruchu korekcyjnego wzdłuż kierunku łączącego punkty *TCP* obydwu robotów.

Celem ruchu korekcyjnego jest minimalizacja zmiany odległości pomiędzy punktami *TCP* robotów. Ze względu na „nieznajomość" celu drugiego robota, do wyliczenia prędkości ruchu korekcyjnego wykorzystano strukturę układu automatycznej regulacji ze zmianą odległości pomiędzy punktami *TCP* jako uchybem. Do wyliczenia wartości prędkości ruchu korekcyjnego wykorzystano regułę proporcjonalno-całkującą (*PI*).
Dla danych wejściowych:

- współrzędnych położenia początkowego punktów *TCP* robota A – $AP(x^{AP}, y^{AP}, z^{AP})$ oraz robota B – $BP(x^{BP}, y^{BP}, z^{BP})$,
- współrzędnych położenia docelowego punktów *TCP* robota A – $A^D(x_A{}^D, y_A{}^D, z_A{}^D)$ oraz robota B – $B^D(x_B{}^D, y_B{}^D, z_B{}^D)$,
- zaprogramowanych prędkości ruchu punktu *TCP* robota A - $V_A{}^P$ i robota B - $V_B{}^P$,

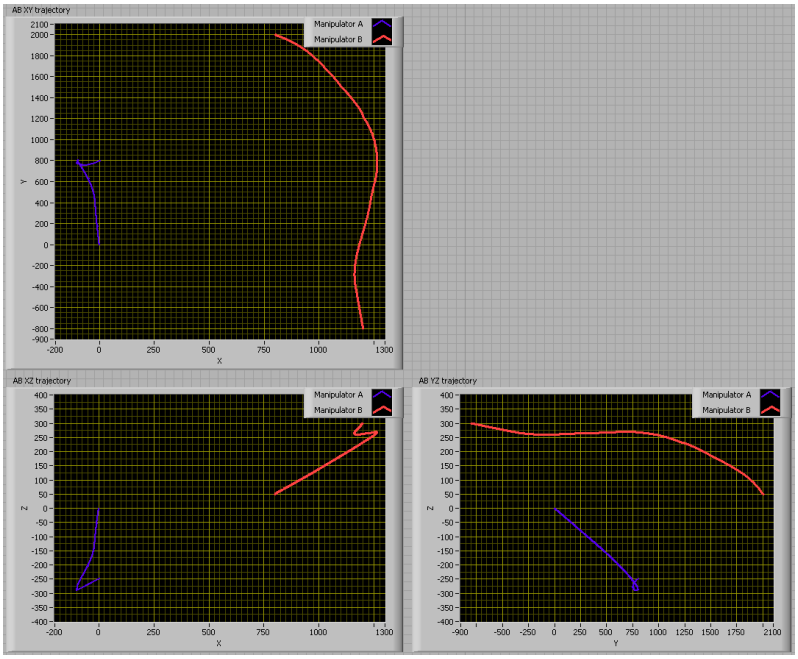
wyliczane są kolejne punkty trajektorii jako złożenie ruchu zaprogramowanego i korekcyjnego. Kształt wyznaczonych trajektorii będzie różny, w zależności od

wartości parametrów bloków korekcyjnych (współczynnik wzmocnienia k_A/k_B i czas zdwojenia T_A/T_B), względnego udziału ruchu korekcyjnego oraz parametrów zaprogramowanego ruchu.
Algorytm generowania trajektorii został zaimplementowany w systemie *LabVIEW*. Ogólny widok modelu pokazano na rysunku 1.



Rys. 1 Diagram blokowy modelu w systemie LabVIEW

Na rysunku 2 zamieszczono rzuty wygenerowanych trajektorii na płaszczyzny układu współrzędnych dla danych wejściowych: $A^P(0, 0, 0)$, $B^P(1200, -800, 300)$, $A^D(0, 800, -250)$, $B^D(800, 2000, 50)$, $V_A^P=50[mm/s]$, $V_B^P=200[mm/s]$ i parametrów korektorów: $k_A=k_B=20[1/s]$, $T_A=T_B=0,2[s]$.



Rys. 2 Przykład wygenerowanych trajektorii

W przedstawionym przykładzie wartości współrzędnych punktów trajektorii obydwu robotów są wyznaczone w tym samym układzie współrzędnych. Wyznaczone współrzędne punktów trajektorii zostały zapisane w dwóch plikach tekstowych.

3. Tworzenie modelu stanowiska w systemie Delmia

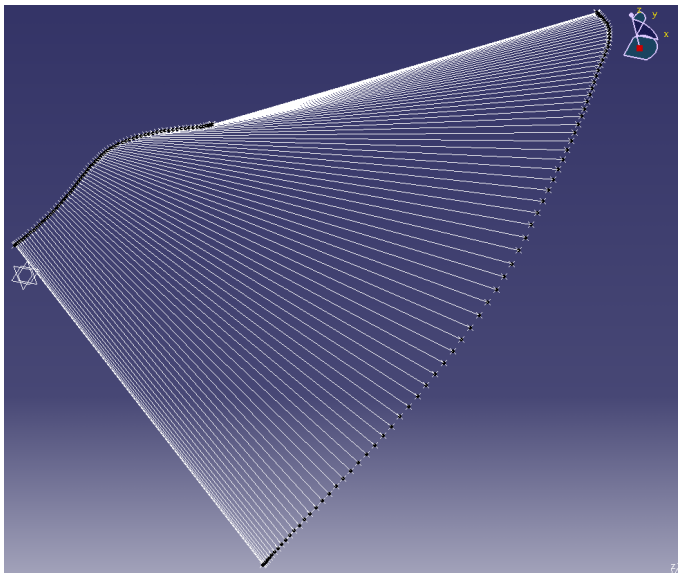
Kolejnym krokiem jest przedstawienie toku postępowania, mającego na celu wykorzystanie danych w przygotowanym formacie do budowy modelu stanowiska zrobotyzowanego. Dla ilustracji problemu przyjęto, że do realizacji zadania transportu wykorzystane będą dwa roboty 4-osiowe firmy Kawasaki *ZD130S*. Transportowanym przedmiotem są trzy połączone na stałe profile prostokątne o wymiarach *200 x 160 x 1000* mm. Ze względu na liczbę stopni swobody robota przygotowano trajektorie ruchu, dla których zmiana orientacji przenoszonego przedmiotu będzie następować tylko poprzez obrót wokół osi równoległej do osi 4 pary kinematycznej robotów. Trajektoria przenoszenia przedmiotu składa się z trzech fragmentów: uniesienie przedmiotu (w płaszczyźnie pionowej), przeniesienie przedmiotu (w płaszczyźnie poziomej) oraz opuszczenie (w płaszczyźnie pionowej). Fragment drugi trajektorii wygenerowano w systemie *LabVIEW* dla danych wejściowych $A^P(0, 0, 0)$, $B^P(800, -600, 0)$, $A^D(0, 800, -250)$, $B^D(1100, 1100, 0)$, $V_A^P=10[mm/s]$, $V_B^P=200[mm/s]$ oraz parametrów korektorów: $k_A=k_B=20[1/s]$, $T_A=T_B=0,2[s]$. Zapisane w programie *LabVIEW* pliki ze współrzędnymi kartezjańskimi fragmentu drugiego trajektorii przeniesiono do systemu *Catia/Delmia*. W tym celu wykorzystano możliwość automatyzacji powtarzalnych działań przy pomocy makr zapisanych w języku *Visual Basic*. Fragmenty pierwszy i trzeci (uniesienie i opuszczenie przedmiotu) to przemieszczenia równoległe punktów *TCP* obydwu robotów, dlatego będą wprost zdefiniowane w systemie *Delmia*. Poniżej przedstawiono etapy tworzenia środowiska w systemie Delmia.

Etap 1: Import danych z plików tekstowych.

Etap ten obejmuje otwarcie pliku ze współrzędnymi punktów trajektorii robota *A*, i dla kolejnych wartości współrzędnych utworzeniu w wybranym dokumencie typu *CATPart* punktów tworzących trajektorię, powtórzenie tego działania dla trajektorii robota *B* oraz utworzenie linii łączących pary punktów obydwu trajektorii. Punkty i linie tworzone są w oddzielnych zbiorach danych geometrycznych – rysunek 3.

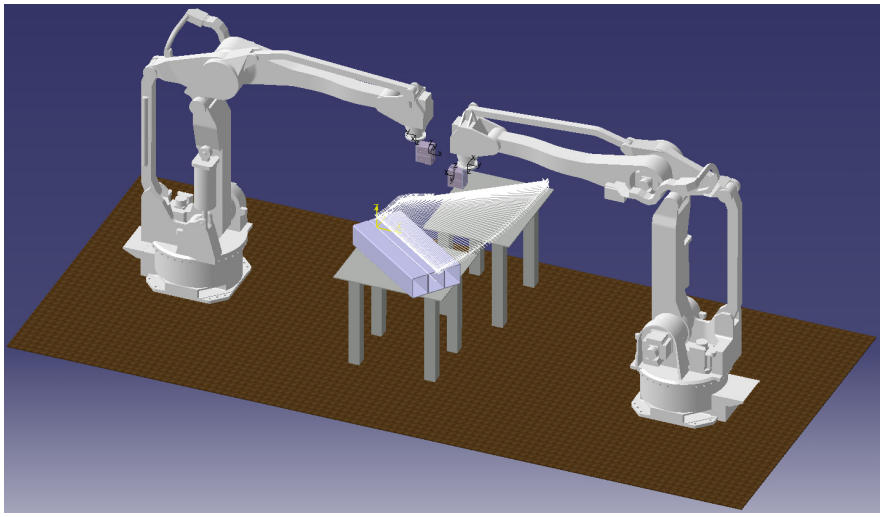
Rys. 3 Widok drzewka dokumentu do wczytania danych

Na rysunku 4 pokazano uzyskane zbiory punktów oraz linie łączące pary punktów obydwu trajektorii dla przenoszenia przedmiotu - fragment drugi trajektorii. Tworzenie punktów i linii jest realizowane automatycznie przez przygotowane makro.



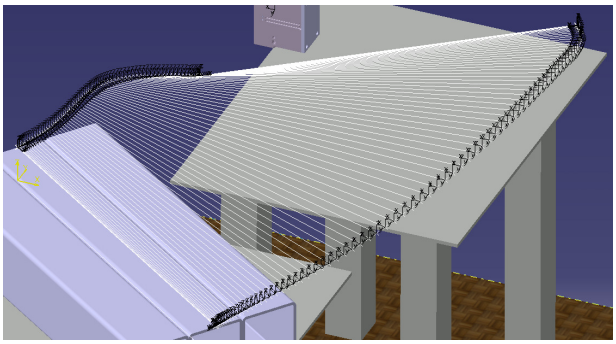
Rys. 4 Punkty i linie łączące pary punktów trajektorii

Etap 2: Zdefiniowanie struktury pliku procesu.
W etapie tym tworzony jest dokument typu *CATProcess*, do którego są wczytywane i wstępnie rozmieszczane elementy środowiska. Jako zasób procesu (gałąź *ResourceList* w drzewku specyfikacji) wczytano dokument *Krzywe.CATProduct*, przygotowany w etapie 1 - zawierający punkty trajektorii. Następnie wczytano modele robotów i chwytaków oraz pozostałe elementy stanowiska. W przedstawianym przykładzie wykorzystano modele 4-osiowych robotów firmy Kawasaki – *ZD130S,* zaczerpnięte z pracy [2], oraz chwytaki firmy Festo *HPGT-B 80*. Chwytaki są montowane do robotów. Jako produkt (gałąź *ProductList*) wczytano model transportowanego elementu. Etap ten jest realizowany ręcznie. Widok wstępnie ustawionego modelu przedstawiono na rysunku 5.



Rys. 5 Widok stanowiska w systemie Delmia

Etap 3: Utworzenie grup *tagów*[1].
Dla każdej trajektorii tworzona jest ręcznie w dokumencie procesu grupa *tagów*, następnie do grupy tej dla każdego punktu trajektorii, utworzonego w etapie 1, wstawiany jest *tag* umieszczony w tym punkcie. Wstępna orientacja tworzonego *taga* jest zgodna z globalnym układem współrzędnych. Orientacja ta jest następnie zmieniana, tak aby: oś *Z* pozostała niezmieniona, oś *X* leżała w płaszczyźnie wyznaczonej przez oś *Z* oraz linię łączącą dwa odpowiadające sobie punkty trajektorii robota *A* i robota *B* oraz była skierowana w kierunku punktu *TCP* drugiego robota, oś *Y* tworzyła z osiami *X* oraz *Z* układ prawoskrętny. Tworzenie *tagów* wykonywane jest automatycznie na podstawie przygotowanego makra. Utworzone dla punktów trajektorii *tagi* przedstawiono na rysunku 6.

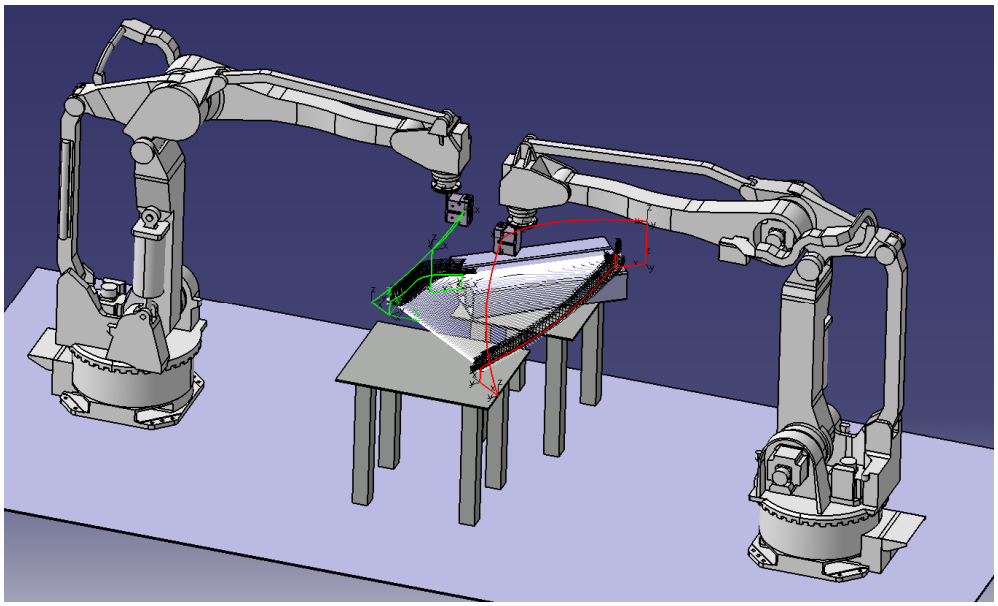


Rys. 6 Widok utworzonych tagów

Etap 4: Zdefiniowanie i konfiguracja zadań ruchu dla robotów.
Dla każdego robota tworzone jest zadanie. Następnie do tego zadania dodawane są *tagi* z odpowiedniej grupy *tagów* – dla każdego dodanego *taga* tworzona jest operacja zawierająca ruch punktu *TCP* do tego *taga*. Następnie zmieniany jest domyślny rodzaj ruchu pomiędzy punktami: *joint* na *linear*. W celu określenia prędkości przemieszczania punktu *TCP* robota definiowany jest profil ruchu, określający czas pokonania dystansu pomiędzy punktami. Ze względu na konieczność osiągania przez obydwa roboty kolejnych punktów w tym samym czasie, wartość czasu ruchu dla obydwu robotów jest jednakowa.
Etap 5: Uzupełnienie zadań ruchu.
Utworzone w etapie 4 zadania zawierają tylko operacje ruchu pomiędzy zaimportowanymi punktami – związane bezpośrednio z realizacją drugiego fragmentu trajektorii. Zadania transportu należy uzupełnić o ruchy unoszenia – przed operacjami ruchu uzyskanymi w etapie 4 i ruchy opuszczania – po operacjach ruchu z etapu 4. Dodatkowo zadania należy uzupełnić o ruchy dojazdu z pozycji początkowej do miejsca uchwycenia przedmiotu (punkt przed ruchem unoszenia) oraz odjazdu od miejsca upuszczenia przedmiotu (punkt po ruchu opuszczania) do pozycji początkowej.
Ponadto zadania robotów należy uzupełnić o akcje uchwycenia i upuszczenia przedmiotu. Równoczesne rozpoczęcie realizacji ruchu transportu zapewni synchronizacja pomiędzy kontrolerami robotów poprzez sygnały we/wy – odpowiednie akcje oczekiwania na sygnał wprowadzono jako warunki rozpoczęcia ruchu po uchwyceniu przedmiotu.
Etap 6: Sprawdzenie realizowalności zadania.

[1] *Tag* to układ współrzędnych określający wymagane położenie i orientację punktu TCP robota w przestrzeni roboczej

Dla wstępnego ustawienia robotów na stanowisku należy sprawdzić, czy wszystkie punkty trajektorii (w zdefiniowanym położeniu i orientacji) są osiągalne. Można wykorzystać również dostępne narzędzie, wyznaczające w zadanym obszarze i z zadanym krokiem lokalizacje robota, w których zdefiniowane zadanie jest realizowalne.
Na rysunku 7 przedstawiono widok modelu stanowiska po wykonaniu zadania transportu, z zaznaczeniem zrealizowanych przez punkty *TCP* robotów trajektorii.



Rys. 7 Widok modelu stanowiska z trajektoriami robotów

4. Podsumowanie

Przedstawiony w rozdziale 2 algorytm generowania skoordynowanych ruchów punktów *TCP* robotów kartezjańskich dla realizacji zadania transportu nie uwzględniał konfiguracji stanowiska: wzajemnego ustawienia robotów i lokalizacji transportowanego przedmiotu. Nie był brany pod uwagę problem osiągalności wyznaczonych punktów, czy możliwość wystąpienia kolizji robotów z elementami stanowiska oraz kolizji pomiędzy robotami i/lub transportowanym przedmiotem. Dodatkowo wyznaczane trajektorie dotyczyły bezpośrednio przeniesienia przedmiotu z położenia początkowego do położenia docelowego, bez ruchów pomocniczych (dojazdu, odjazdu) oraz akcji uchwycenia czy upuszczenia przedmiotu. W celu rozwiązania tych problemów zaproponowano wykorzystanie środowiska graficznego *Delmia*, które posiada wymagane funkcjonalności. Celem pracy było przedstawienie możliwości wykorzystania generowanych w systemie *LabVIEW* trajektorii ruchu robotów do zbudowania modelu stanowiska w systemie *Delmia*, który posłuży za „stanowisko testowe” do weryfikacji uzyskanych wyników. Opracowano interfejs umożliwiający export danych z systemu *LabVIEW*. Następnie opracowano procedurę obejmującą import tych danych do systemu *Delmia* oraz wykorzystanie ich do budowy modelu stanowiska. Pracochłonne, wymagające powtarzalnych działań etapy procedury zostały zautomatyzowane przy pomocy makr, zadania proste do wykonania w interfejsie graficznym zostały zrealizowane manualnie. W celu ilustracji przedstawionego toku postępowania opracowano i zaprezentowano przykład.

5. Literatura

[1] http://www.dinsaonline.com/aplicaciones/pdf/dualram_system/dualarmSystem.pdf
[2] A. Konicki: *Programowanie zrobotyzowanego stanowiska do paletyzacji metodą symulacyjną w Delmii*, Praca Dyplomowa, Politechnika Krakowska, Kraków 2010
[3] MultiMove 3HAC021272-001_RevG_en.pdf, ABB, *Application manual – MultiMove*
[4] A. Słota: *Model koordynacji trajektorii efektorów dwóch manipulatorów kartezjańskich z uwzględnieniem dynamiki układów napędowych,* Pomiary Automatyka Robotyka 2/2009, str. 569-576
[5] A. Słota: *Model koordynacji trajektorii efektorów manipulatorów kartezjańskich – weryfikacja stabilności,* Pomiary Automatyka Robotyka 2/2010, str. 628-634