dr inż. Adam Słota
Instytut Technologii Maszyn i Automatyzacji Produkcji, Politechnika Krakowska

GENEROWANIE SKOORDYNOWANYCH RUCHÓW ROBOTÓW W PRZESTRZENI 3D – WERYFIKACJA W ŚRODOWISKU WIRTUALNYM

W pracy przedstawiono algorytm generowania trajektorii dla zadania transportowego realizowanego przez dwa roboty. Algorytm wyznacza wymagane pozycje i orientacje chwytaków robotów w trakcie skoordynowanych ruchów. W celu weryfikacji uzyskanych danych wykorzystano system Delmia. Zaproponowano procedurę 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. Zbudowany wirtualny model stanowiska wykorzystano do weryfikacji zdefiniowanych zadań dla robotów. Opracowany tok postępowania zilustrowano przykładem.

GENERATION OF COORDINATED ROBOT MOTION IN 3D SPACE – VERIFICATION IN VIRTUAL ENVIRONMENT

In the paper an algorithm of generation trajectories for transport tasks executed by two robots is presented. The algorithm calculates positions and orientations of robots' grippers during coordinated motion. For verification purposes application of Delmia system is suggested. A procedure aiming at: transferring data from LabVIEW to Delmia, building model of 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. Virtual model of robotized cell is used for robot's tasks verification. For illustration purposes an example is presented.

1. WSTĘP

Obserwowany w ostatnim czasie rozwój w dziedzinie robotyki dotyczy zarówno zwiększenia możliwości i zakresu zastosowania pojedynczych robotów (możliwości układów sterowania, dynamika układów napędowych, sterowanie pozycyjno-siłowe), jak również systemów zrobotyzowanych. W przypadku systemów zrobotyzowanych, bezkolizyjne działanie systemu oraz zapewnienie wykonania określonych fragmentów procesu w wymaganej kolejności realizowane jest przez synchronizację punktów programów różnych robotów za pomocą sygnałów wejścia/wyjścia. Obok takiej dyskretnej koordynacji działań robotów rozwijane są rozwiązania dotyczące ciągłej koordynacji trajektorii dwóch lub większej liczby robotów. Rozwiązania takie stosowane są na przykład przy przenoszeniu dużych i ciężkich przedmiotów czy spawaniu, gdy ze względu na kształt i przestrzenne usytuowanie spoiny wymagana jest zmiana orientacji łączonych elementów w czasie realizacji procesu. Dostarczane przez producentów rozwiązania w tym zakresie koncentrują się wokół zastosowania jednego kontrolera do sterowania kilkoma robotami. Rozwiązanie *DUALARM* [3] firmy Fanuc pozwala sterować ruchami dwóch robotów, rozwiązania *Multi arm control* [6] firmy Fanuc i *MultiMove* [5] firmy ABB umożliwiają sterowanie przy pomocy jednego kontrolera ruchami czterech robotów. Rozwiązania te pozwalają na zaprogramowanie i zrealizowanie transportu przedmiotu przez dwa roboty. Jeden kontroler wykonuje programy zawierające procedury skoordynowanych ruchów dla obydwu robotów, a wymagane wzajemne położenie i orientacja punktów *TCP* chwytaków jest zapewniona przez zaprogramowanie ruchów jednego robota w ruchomym układzie współrzędnych, poruszanym przez drugiego robota [8].

Dotychczasowe prace autora dotyczyły opracowania algorytmów koordynacji trajektorii dwóch manipulatorów, sterowanych przez oddzielne kontrolery, realizujących wspólnie zadanie transpor-

towe. Prace te obejmowały generowanie trajektorii dla manipulatorów kartezjańskich, bez uwzględniania wzajemnej orientacji punktów *TCP* manipulatorów. Opracowany algorytm został zaimplementowany w systemie LabVIEW. Działanie algorytmu zostało zweryfikowane dla modelu symulacyjnego uwzględniającego dynamikę manipulatora kartezjańskiego. Opracowany algorytm, scharakteryzowany w punkcie 2.1, ma za zadanie wyznaczanie on-line kolejnych punktów trajektorii w czasie działania manipulatorów. Jednak ze względu na zastosowane założenia upraszczające [7] (np. przegubowe połączenie transportowanego przedmiotu z manipulatorem, nieuwzględnianie konfiguracji stanowiska i możliwych kolizji) uzyskane wyniki wymagają dalszej analizy i weryfikacji. Można to zrealizować na dwa sposoby. Pierwszy to rozbudowa już istniejącego modelu w systemie LabVIEW, tak aby uwzględniał wszystkie istotne dla działania układu zagadnienia. Sposób drugi to przeniesienie danych, uzyskanych w systemie LabVIEW dla modelu uproszczonego, do innego systemu w celu ich weryfikacji.

Specyfika systemu LabVIEW i jego graficzny język programowania sprawiają, że system doskonale sprawdza się w budowie modeli i symulacji systemów dynamicznych, jednak implementacja na przykład wykrywania kolizji podczas realizacji ruchu robota byłaby bardzo trudna. Konfiguracja stanowiska, weryfikacja osiągalności punktów trajektorii czy analiza kolizji to zadania, do wykonania których o wiele lepiej wykorzystać systemy służące do programowania off-line robotów przemysłowych. Oprogramowanie takie dostarczane jest przez wiodących producentów robotów (*Robot Studio* firmy ABB, *RoboGuide* firmy Fanuc, *PC Roset* firmy Kawasaki). Programy tego typu wspomagają programowanie robotów danego producenta z wykorzystaniem dostępnych wirtualnych modeli 3D robotów i urządzeń pomocniczych stanowiska. Rozwiązaniem alternatywnym są systemy uniwersalne umożliwiające konfigurację i weryfikację działania stanowisk zrobotyzowanych z wykorzystaniem robotów różnych producentów – udostępniające bogate biblioteki robotów, dające możliwość definiowania modeli własnych urządzeń oraz udostępniające interfejs wymiany danych z innymi systemami. Systemem takim jest Delmia firmy Dassault Systems.

Celem pracy jest rozbudowa algorytmu generowania trajektorii o możliwość wyznaczania orientacji układów współrzędnych, które definiują orientację chwytaków robotów realizujących wspólnie zadanie transportowe. Ten etap zostanie przedstawiony w punkcie 2.2. Kolejny etap to przeniesienie uzyskanych danych do systemu Delmia, gdzie na ich podstawie zbudowany zostanie model stanowiska umożliwiający wstępną weryfikację rozwiązania w wirtualnym środowisku 3D. Opis rozwiązania tego zadania na przykładzie wybranego stanowiska zamieszczono w sekcji 3.

2. ALGORYTM WYZNACZANIA TRAJEKTORII

2.1. Obliczanie pozycji punktów *TCP* robotów

Algorytm wyznacza współrzędne kartezjańskie punktów trajektorii w układzie współrzędnych zadania, które stanowią wartość zadaną pozycji punktu *TCP* do zrealizowania przez układy napędowe robota. Przyjęto, że układy współrzędnych zadania obydwu robotów mają taką samą orientację. Dane wejściowe algorytmu to:

- położenie układu współrzędnych zadania robota B względem układu zadania robota A – $D_B^A(dx_B^A, dx_B^A, dx_B^A)$,
- współrzędne pozycji punktu *TCP* robota A w chwili początkowej w układzie współrzędnych zadania robota A – $A^P(x_A^P, y_A^P, z_A^P)$,
- współrzędne pozycji punktu *TCP* robota B w chwili początkowej w układzie współrzędnych zadania robota B – $B^P(x_B^P, y_B^P, z_B^P)$,
- współrzędne pozycji docelowej punktu *TCP* robota A w układzie współrzędnych zadania robota A – $A^D(x_A^D, y_A^D, z_A^D)$,
- współrzędne pozycji docelowej punktu *TCP* robota B w układzie współrzędnych zadania robota B – $B^D(x_B^D, y_B^D, z_B^D)$,
- zaprogramowane prędkości ruchu punktu *TCP* robota A – V_A^P i robota B – V_B^P.

Odległość pomiędzy punktami *TCP* w chwili początkowej (A^P oraz B^P) to odległość referencyjna $l_0=|A^P–B^P|$, która ma być zachowana w czasie realizacji ruchu. W aktualnej wersji algorytmu ruch zaprogramowany to ruch wzdłuż linii prostej z pozycji bieżącej do pozycji docelowej z zaprogramowaną prędkością. Ruch taki, realizowany przez dwa roboty, prowadzi do zmiany odległości pomiędzy ich punktami *TCP* ($\Delta l(t) = l_0 – l(t)$). Dlatego ruch zaprogramowany uzupełniany jest o składową korekcyjną. Ruch korekcyjny wykonywany jest wzdłuż kierunku określonego przez bieżącą pozycję punktów *TCP* obydwu robotów, a jego celem jest minimalizacja zmiany odległości pomiędzy punktami *TCP*. Wartość prędkości ruchu korekcyjnego obliczana jest na podstawie zmiany odległości pomiędzy punktami *TCP* $\Delta l(t)$ jako suma dwóch składników: części proporcjonalnej do wyliczonej zmiany odległości $\Delta l(t)$ oraz części proporcjonalnej do całki ze zmiany odległości $\Delta l(t)$.

Dla osiągnięcia zaprogramowanych pozycji docelowych A^D oraz B^D wymagane jest, aby odległość pomiędzy nimi była równa odległości referencyjnej l_0.

2.2. Obliczanie orientacji chwytaków robotów

Sztywne uchwycenie transportowanego przedmiotu przez chwytaki obydwu robotów wymaga wzajemnego dostosowania orientacji chwytaków w czasie realizacji zadania transportu, tak aby zachowana była początkowa orientacja chwytaków względem transportowanego przedmiotu. W celu wyznaczenia wymaganej orientacji chwytaków robotów wzdłuż trajektorii przyjęto następującą orientację układu współrzędnych chwytaka robota *A* (robota *B*) w chwili początkowej:

- kierunek i zwrot osi x_{T0}^A (x_{T0}^B) określony jest przez wektor łączący punkt *TCP* robota *A* z punktem *TCP* robota *B* (punkt *TCP* robota *B* z punktem *TCP* robota *A*),
- oś y_{T0}^A (y_{T0}^B) jest równoległa do płaszczyzny X_ZY_Z układu współrzędnych zadania robota *A* (robota *B*), a jej zwrot jest taki, aby oś z_{T0}^A (z_{T0}^B) tworzyła z osią z układu zadania robota *A* (robota *B*) kąt mniejszy lub równy 90°,
- układ $x_{T0}^A\, y_{T0}^A\, z_{T0}^A$ ($x_{T0}^B\, y_{T0}^B\, z_{T0}^B$) jest układem prawoskrętnym.

Wyznaczone według przedstawionego w punkcie 2.1 algorytmu pozycje punktów *TCP* robotów częściowo określają wymaganą orientację chwytaków. Ruch punktu *TCP* robota realizowany jest w płaszczyźnie prostopadłej do kierunku wyznaczonego przez chwilowe pozycje punktów *TCP* obydwu robotów.

Pozycje te definiują chwilową orientację osi x_T^A (x_T^B) robota *A* (robota *B*). Do wyliczenia pozostaje orientacja osi y_T^A, z_T^A (y_T^B, z_T^B) w płaszczyźnie ruchu. Orientacja tych osi określona będzie przez kąt obrotu wokół osi x_T^A (x_T^B) względem orientacji początkowej. Kąt obrotu δ wokół osi x_T^A (x_T^B) jest czwartą współrzędną, obok współrzędnych pozycji, którą należy wyznaczyć dla pełnego opisu punktów trajektorii chwytaków robotów.

Dane wejściowe do wyznaczenia kąta obrotu δ to:

- wartość początkowa kąta obrotu dla robota $A – \delta_A^P$ oraz robota $B – \delta_B^P$,
- wartość docelowa kąta obrotu dla robota $A – \delta_A^D$ oraz robota $B – \delta_B^D$,
- zaprogramowane prędkości obrotu dla robota A - ω_A^P i robota $B - \omega_B^P$.

Ze względu na wymaganą zgodność kątów obrotu oraz możliwe różnice w rzeczywistej prędkości obrotu, wartość zadana kąta δ (kąta obrotu wokół osi x_T układu współrzędnych chwytaka) jest wyznaczana w sposób analogiczny, jak współrzędne pozycji.

Prędkość obrotu wyliczana jest jako suma dwóch składników: prędkości zaprogramowanej i prędkości korekcyjnej. Prędkość obrotu korekcyjnego chwytaka robota *A* (robota *B*) obliczana jest na podstawie różnicy pomiędzy rzeczywistymi kątami obrotu chwytaków $\Delta\delta_A(t)= \delta_B(t)–\delta_A(t)$ ($\Delta\delta_B(t)= \delta_A(t)–\delta_B(t)$), jako suma dwóch składników: części proporcjonalnej do wyliczonej różnicy $\Delta\delta_A(t)$ ($\Delta\delta_B(t)$) oraz części proporcjonalnej do całki z różnicy $\Delta\delta_A(t)$ ($\Delta\delta_B(t)$).

Orientacja układu współrzędnych chwytaka jest określona przez oś x_T (wyznaczoną przez wektor łączący punkty *TCP* chwytaków obydwu robotów) oraz kąt obrotu δ wokół osi x_T. Orientację

wyrażono przy pomocy kątów obrotu wokół stałych osi układu współrzędnych zadania w kolejności: obrót o kąt α wokół osi x_Z, obrót o kąt β wokół osi y_Z oraz obrót o kąt γ wokół osi z_Z [1].

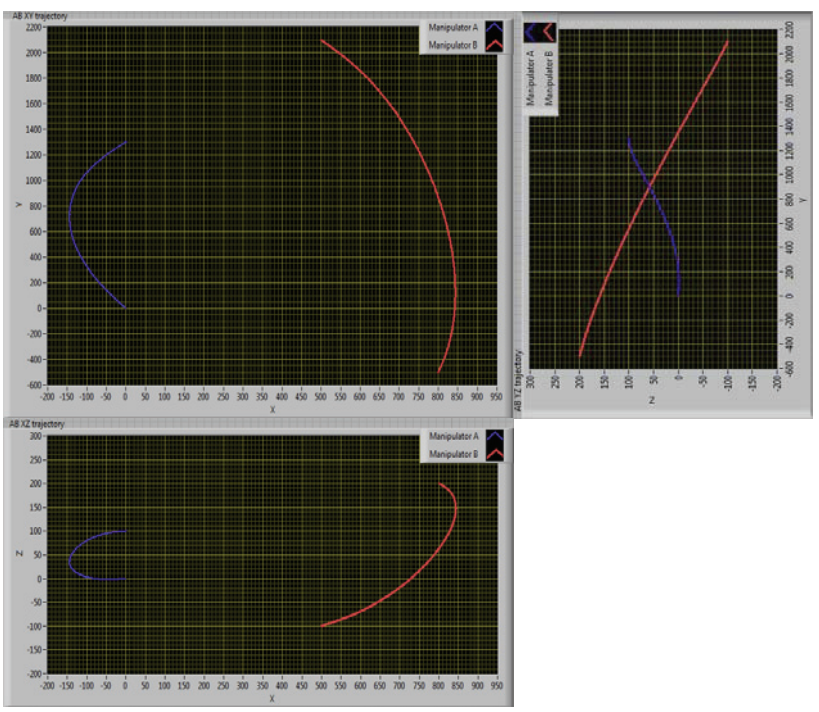
W celu weryfikacji opracowanego algorytmu zaimplementowano go w systemie LabVIEW z wykorzystaniem modułu *Simulation Module*. Wykorzystano model służący do generowania pozycji punktów trajektorii dla przypadku płaskiego, opracowany na potrzeby pracy [7]. Model ten rozbudowano o wyznaczanie pozycji punktów trajektorii w przestrzeni 3D oraz uzupełniono o wyliczanie kątów obrotów określających orientacje chwytaków robotów w trakcie ruchu. W modelu symulacyjnym uwzględniono, poza przedstawieniem wyników symulacji w postaci wykresów, możliwość zapisu do pliku wartości numerycznych. Do plików tekstowych zapisywane są współrzędne pozycji oraz kąty obrotów określających orientację chwytaka wzdłuż wyznaczanej trajektorii. Wyniki te posłużą jako dane wejściowe do weryfikacji algorytmu w środowisku wirtualnym. Dążenie do możliwie małych zmian odległości $\Delta l(t)$ pomiędzy chwytakami w trakcie ruchu wymaga małych kroków czasowych przy obliczaniu kolejnych punktów trajektorii, co pociąga za sobą dużą liczbę wyznaczonych punktów. W celu ograniczenia liczby punktów przy przenoszeniu danych do systemu Delmia, do plików tekstowych oprócz współrzędnych pozycji i orientacji, zapisywany jest również czas osiągania kolejnych punktów, licząc od początku symulacji.

2.3. Przykład wygenerowanych trajektorii

Przyjęto, że do realizacji zadania transportu wykorzystane będą dwa roboty firmy Fanuc *S-420F*. Transportowanym przedmiotem są trzy połączone na stałe profile prostokątne o wymiarach 200×160×1000 mm. Trajektorie przenoszenia przedmiotu wygenerowano w systemie LabVIEW dla danych wejściowych: $D_B^A(800, -500, 200)$, $A^P(0, 0, 0)$, $B^P(0, 0, 0)$, $A^D(0, 1300, 100)$, $B^D(-300, 2600, -300)$, $V_A^P = 50$ mm/s, $V_B^P = 200$ mm/s, $\delta_A^P = \delta_B^P = 0$ rad, $\delta_A^D = \delta_B^D = \pi/4$ rad, $\omega_A^P = 0{,}015$ rad/s, $\omega_B^P = 0{,}016$ rad/s. Wartości parametrów korektorów wynoszą:

- korekcja położenia: $k_A = k_B = 20$ 1/s, $T_A = T_B = 0{,}2$ s,
- korekcja kąta $k_{\delta A} = k_{\delta B} = 10$ 1/s, $T_{\delta A} = T_{\delta B} = 0{,}1$ s.

Współrzędne trajektorii obliczono stosując stały krok czasowy *0,005* s. Rzuty wygenerowanych trajektorii na płaszczyzny układu współrzędnych zadania robota *A* przedstawia rys. 1.



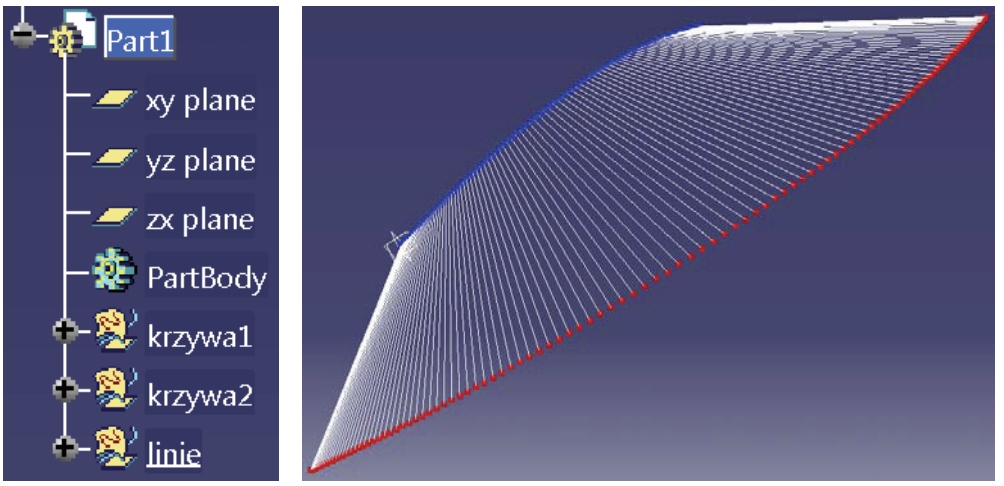
Rys. 1. Widok wygenerowanych trajektorii

Uzyskane wyniki (wartości czasu, współrzędne pozycji i orientacji) zapisano w plikach tekstowych.

3. WERYFIKACJA W SYSTEMIE DELMIA

3.1. Import danych do systemu Delmia

W celu przeniesienia do systemu Delmia wygenerowanych w systemie LabVIEW i zapisanych w plikach tekstowych współrzędnych pozycji i orientacji chwytaków robotów, wykorzystano udostępniany przez system Delmia interfejs API. W pierwszej kolejności przygotowano dokument typu *CATPart*, w którym umieszczono zbiory elementów geometrycznych: *krzywa1*, *krzywa2* oraz *linie*. Przygotowane w języku *VisualBasic* makro otwiera plik (funkcja *OpenTextFile*) i odczytuje współrzędne pozycji punktów trajektorii, w zbiorach *krzywa1* i *krzywa2* tworzy punkty reprezentujące kolejne położenia punktów *TCP* chwytaków odpowiednio robota *A* i robota *B* (funkcja *AddNewPointCoord*), a w zbiorze *linie* tworzy linie łączące odpowiednie pary punktów (funkcja *AddNewLinePtPt*).



Rys. 2. Utworzone punkty trajektorii

Utworzony model (rys. 2) przedstawia kształt trajektorii punktów *TCP* robotów. Ze względu na dużą liczbę punktów trajektorii odczytane dane z plików są filtrowane, a punkty i linie tworzone są dla kolejnych położeń chwytaków w chwilach będących ustaloną wielokrotnością kroku symulacji. Dla przedstawianego przykładu kolejne punkty tworzone są co *0,5* s.

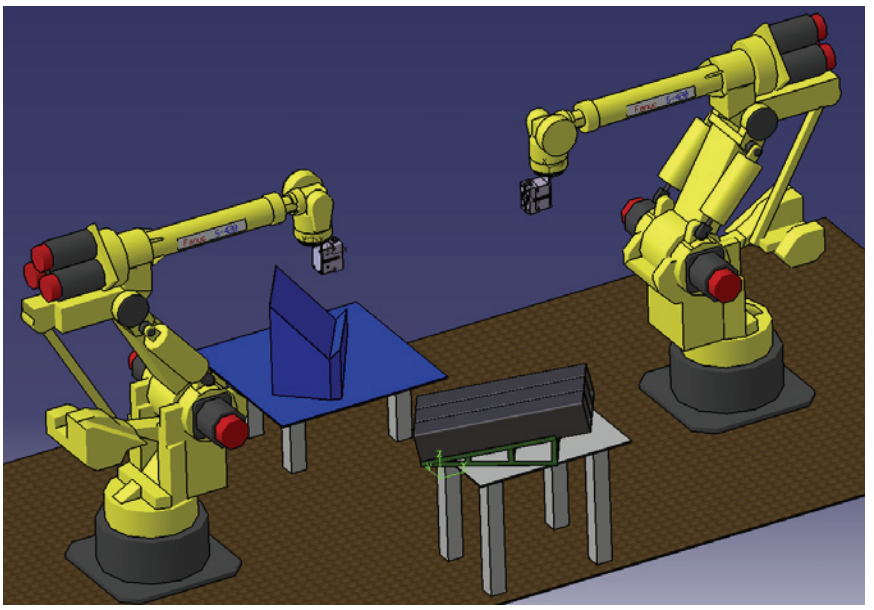
3.2. Model stanowiska w systemie Delmia

Wirtualny model stanowiska tworzony jest w dokumencie typu *CATProcess*. Do dokumentu tego wczytywane są i wstępnie rozmieszczane elementy stanowiska. W gałęzi *ResourceList* umieszczono: modele robotów *Fanuc S-420F* zaczerpnięte z biblioteki modeli robotów systemu Delmia, modele chwytaków firmy Festo *HPGT-80-A-B G2* [4] oraz własne modele elementów dodatkowych (stoły, podłoga). Chwytaki zamontowano do robotów. W gałęzi *ProductList* umieszczono model transportowanego elementu oraz dokument z wygenerowanymi punktami przedstawiającymi wymagane trajektorie. Model transportowanego przedmiotu jest umieszczony tak, aby jego oś była zgodna z linią łączącą pierwsze punkty trajektorii, a następnie przesunięty o 100 mm w ujemnym kierunku osi *Z* globalnego układu współrzędnych. Przesunięcie to wynika z potrzeby uniesienia przedmiotu przed wykonaniem ruchu transportu określonego przez wygenerowane trajektorie. Na rys. 3 przedstawiono wirtualny model stanowiska.

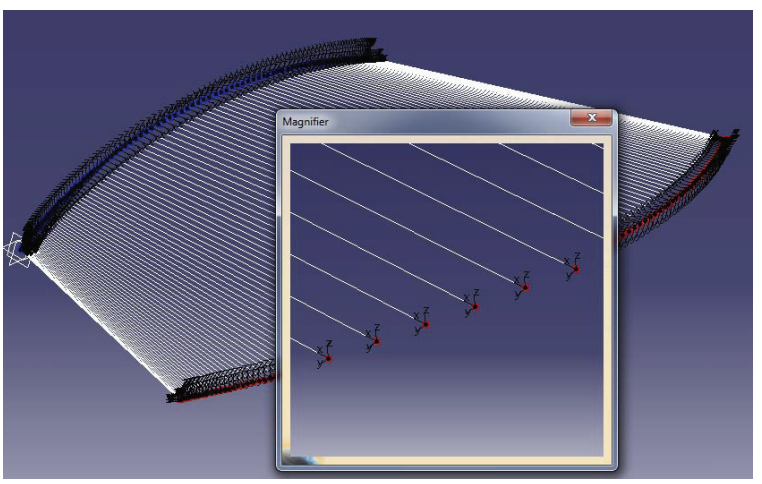
W kolejnym kroku dla każdej trajektorii tworzona jest w dokumencie procesu grupa *tagów*[1]. Następnie przy pomocy opracowanego makra, dla każdego punktu trajektorii tworzony jest *tag* . Funkcja *CreateTag* tworzy *taga*, funkcja *SetXYZ* ustala jego położenie, a funkcja *SetYPR* definiuje orien-

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

tację układu *tag* przy pomocy kątów obrotu wokół osi stałych. Przy tworzeniu *tagów* przyjęto orientację jak w punkcie 2.2. Widok utworzonych *tagów* dla wygenerowanych trajektorii przedstawiono na rys. 4.



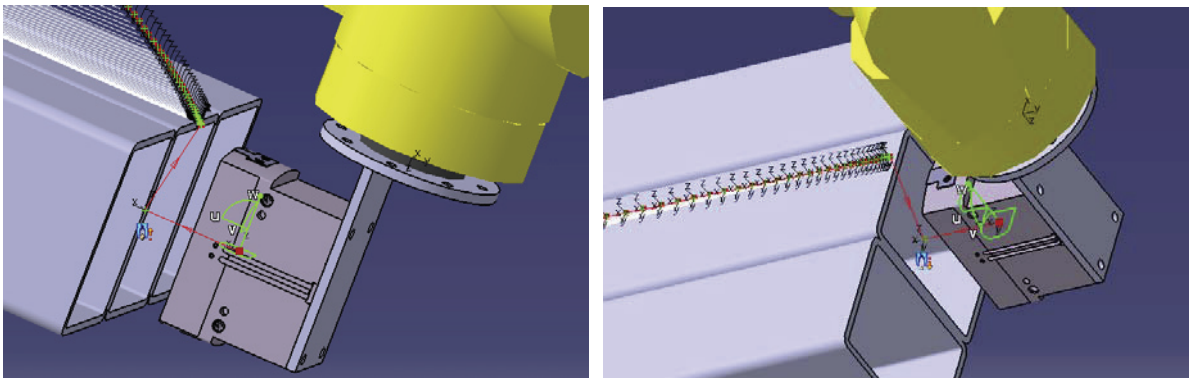
Rys. 3. Widok modelu stanowiska w systemie Delmia



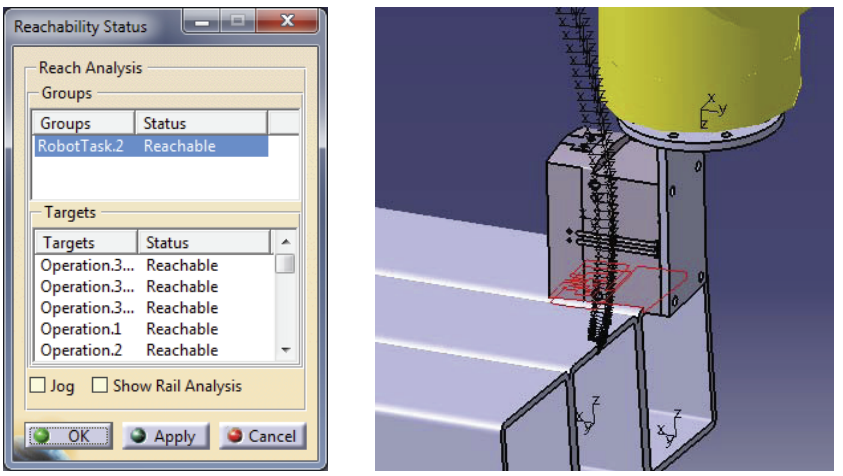
Rys. 4. Widok utworzonych tagów

Przeniesienie przedmiotu realizowane będzie w kolejnych etapach: (i) dojazd do pozycji uchwycenia przedmiotu, (ii) uniesienie przedmiotu, (iii) przeniesienie, (iv) opuszczenie przedmiotu oraz (v) odjazd do pozycji bazowej.

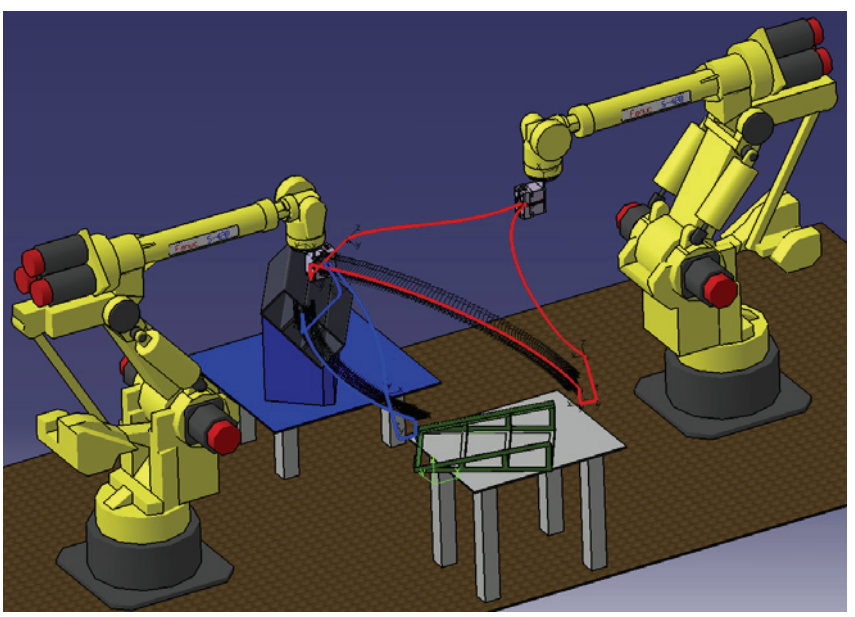
Dla każdego robota tworzone jest zadanie. Do zadania robota dodawane są utworzone wcześniej *tagi* opisujące etap (iii) trajektorii. Dla każdego dodanego *taga* tworzona jest operacja zawierając ruch punktu *TCP* do tego *taga*. Następnie definiowane są parametry ruchu: interpolacja typu *linear* oraz prędkość ruchu określona jako czas pokonania dystansu pomiędzy kolejnymi *tagami*. Jest to wartość czasu przyjęta przy tworzeniu kolejnych punktów w systemie Delmia na podstawie danych uzyskanych w LabVIEW. W przedstawianym przykładzie czas ten wynosi *0,5* s i jest taki sam dla zadań obydwu robotów ze względu na konieczność osiągania przez obydwa roboty kolejnych punktów w tym samym czasie. Kolejny krok to dodanie do zadania operacji dojazdu do pozycji uchwycenia i ruchu uniesienia przedmiotu oraz ruchu opuszczenia i odjazdu do pozycji bazowej, jak również zdefiniowanie akcji uchwycenia i upuszczenia przedmiotu (rys. 5).



Rys. 5. Akcje uchwycenia i upuszczenia przedmiotu



Rys. 6. Weryfikacja osiągalności i kolizji



Rys. 7. Widok modelu stanowiska z trajektoriami robotów

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.

Dla zdefiniowanych zadań ruchu sprawdzono osiągalność punktów trajektorii (pozycji i orientacji). W przypadku wystąpienia punktów nieosiągalnych można wykorzystać dostępne narzędzie, wyznaczające w zadanym obszarze i z zadanym krokiem lokalizacje robota, w których zdefiniowane zadanie jest realizowalne. Następnie przeprowadzono symulację działania stanowiska z włączoną opcją wykrywania kolizji. Po wykryciu kolizji (rys. 6) zmodyfikowano trajektorię dojazdu chwytaka do punktu uchwycenia przedmiotu.

Na rys. 7 przedstawiono widok modelu stanowiska po wykonaniu zadania transportu, z zaznaczeniem zrealizowanych przez punkty *TCP* robotów trajektorii.

4. PODSUMOWANIE

Opracowany algorytm generowania skoordynowanych ruchów robotów kartezjańskich dla realizacji zadania transportu obejmuje wyznaczanie pozycji i orientacji punktów *TCP*. Ze względu na specyfikę systemu LabVIEW, w którym algorytm został zaimplementowany i testowany, przy generowaniu trajektorii nie uwzględniano 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. W celu uwzględnienia wymienionych powyżej zagadnień zaproponowano wykorzystanie środowiska graficznego Delmia, które posiada wymagane funkcjonalności. W systemie tym, korzystając z biblioteki modeli robotów oraz samodzielnie zdefiniowanych modeli chwytaków i dodatkowych elementów stanowiska, zbudowano wirtualny model stanowiska do realizacji zadania transportowego przez dwa roboty. Dane opisujące trajektorie, wygenerowane w systemie LabVIEW i zapisane w plikach tekstowych, przeniesiono do systemu Delmia, gdzie na ich podstawie zdefiniowano zadania ruchu robotów realizujących transport przedmiotu. Zadania te uzupełniono o ruchy dojazdu/odjazdu oraz akcje uchwycenia i upuszczenia przedmiotu. Korzystając z dostępnych funkcjonalności systemu Delmia dla tak zdefiniowanego modelu sprawdzono poprawność działania stanowiska: sprawdzono osiągalność punktów trajektorii robotów oraz możliwość wystąpienia kolizji.

BIBLIOGRAFIA

[1] Craig J.: Wprowadzenie do robotyki, Mechanika i sterowanie, WNT, Warszawa 1993.

[2] http://www.festo.com/pnf/pl_pl/products/catalog?action=search&key=hgple.

[3] http://www.dinsaonline.com/aplicaciones/pdf/dualram_system/dualarmSystem.pdf.

[4] http://xdki.festo.com/xdki/data/doc_ENGB/PDF/EN/HGPT-B_EN.PDF.

[5] MultiMove 3HAC021272-001_RevG_en.pdf, ABB, Application manual – MultiMove.

[6] R-J3iC_MultiArm_[R-J3iC_HandlingTool_Operator_ManualMAROCH70008051E _REV.A].pdf, Fanuc Robotics.

[7] Słota A.: Model koordynacji trajektorii efektorów dwóch manipulatorów kartezjańskich z uwzględnieniem dynamiki układów napędowych, Pomiary Automatyka Robotyka 2/2009, ISSN 1427-9126 Indeks 339512, s. 569−576.

[8] Słota A., Domka M.: Programowanie skoordynowanych ruchów robotów na przykładzie robotów ABB i Fanuc, Pomiary Automatyka Robotyka 2/2010, ISSN 1427-9126 Indeks 339512, s. 635−644.