Ćwiczenie C

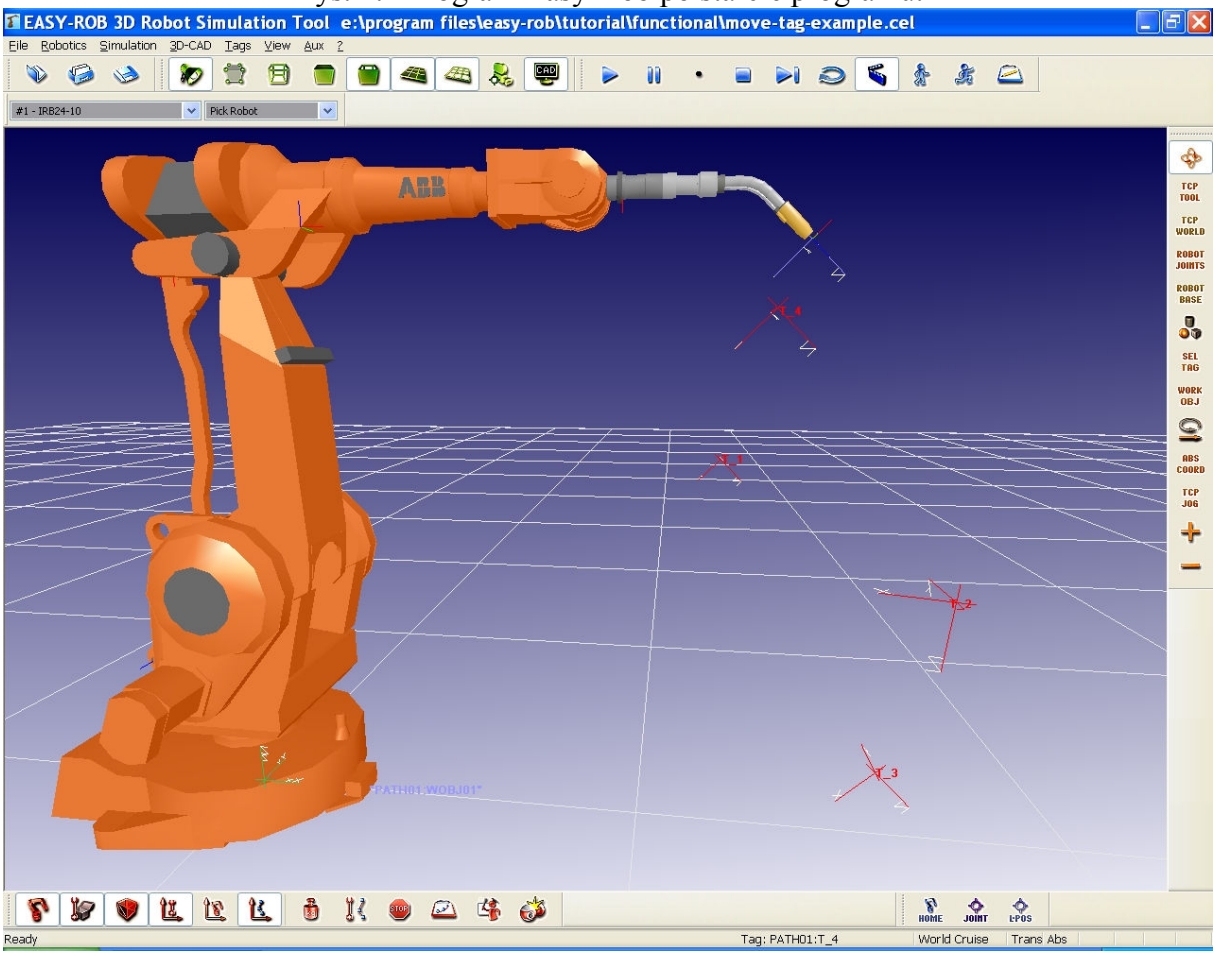
Symulacja pracy robotów w środowisku EASY- ROB.

1. Wprowadzenie

Program Easy-Rob (ver5.3) jest kompleksowym środowiskiem do symulacji pracy robota. Programowanie odbywa się przy pomocy języka zbliżonego do tego jaki występuje na stanowisku rzeczywistym, uzupełnionego o funkcje pozwalające na efektywną wizualizację procesu modelowania.
Programowanie off-line ma szereg zalet w stosunku do programowania on-line:
- chroni zawczasu przed możliwymi błędami
- zmniejsza koszty przeprogramowywania robotów nie wymagając zatrzymania linii produkcyjnych
- pozwala na diagnostykę szerszego zakresu zmiennych niż zazwyczaj jest to możliwe na stanowisku rzeczywistym.

Warunkiem jego wykorzystania jest kalibracja modelu w stosunku do robota rzeczywistego i jego środowiska, co samo w sobie jest zadaniem złożonym i nie zawsze możliwym do przeprowadzenia z wystarczającą dokładnością.

Rys. 1.1 Program Easy-Rob po starcie programu.



Każda sesja pracy w tym środowisku składa się z reguły z następujących kroków:

1. Załadowanie robota.
2. Edycja jego programu.
3. Odtworzenie programu z odpowiednim monitoringiem i ewentualną rejestracją wybranych przebiegów.

2. Menu programu.

Dla elastycznej realizacji funkcji modelowania służy rozbudowane menu.
W tej części instrukcji zostanie opisane menu wizualizowane za pomocą ikon.
(uwagi ogólne :

-waga poszczególnych pozycji zależy oczywiście od wybranego zadania, szczególną uwagę jednak należy zwrócić na pozycje wytłuszczone

- większość funkcji dostępnych jest również w rozwijalnym menu tekstowym)

2.1 Menu górne:

Które składa się z następujących części:

- Menu ładowania (od lewej):

1. ładowanie robota z jego środowiskiem i programem (format pliku *.cel)
2. ładowanie jw. z biblioteki
3. zachowanie stworzonego robota, środowiska i programu

- Menu Widoku - przełączniki:

1. zmiana oświetlenia sceny
2. wizualizacja sceny pełna/punktowa
3. wizualizacja sceny pełna/szkieletowa
4. zmiana sposobu zaokrąglenia
5. podłoga widoczna bądź nie
6. podłoga w schemacie pełnym bądź szkieletowym
7. nagrywanie AVI
8. przełącznik widoku symulacji

- **Menu symulacji pracy robota:**

1. Przycisk startu
2. Przycisk pauzy
3. Kontynuacja

4. Zakończenie programu.
5. Przełącznik pracy krokowej.
6. Ustawienie pracy cyklicznej.
7. Przełącznik animacji 3d.
8. Zmniejsza krok symulacji
9. Wydłuża krok symulacji
10. Otwiera okno wartości, które mogą być monitorowane w czasie pracy robota.

Menu dolne:

składa się z następujących części:

- Przełączniki wizualizacji:

1. robota (widoczny bądź nie).
2. narzędzia
3. obiektu zewnętrznego
4. układu współrzędnych punktu odniesienia narzędzia (***TCP*** -*Tool Center Point*), względem którego odbywa się zwykle programowanie ruchów.
5. pozostałych układów współrzędnych związanych z robotem
6. układów współrzędnych związanych z pozycjami docelowymi tzw. *Tags*.

- **Przełączniki związane z odtwarzaniem trajektorii:**

1. Uwzględnienie modelu dynamiki
2. Aktywacja/ dezaktywacja śladu kreślonego w przestrzeni przez narzędzie (***TCP***)
3. Aktywacja/ dezaktywacja wyłączników krańcowych (które ograniczają ruch robota w złączach)
4. Wizualizacja okna wykonania programu
5. Otwarcie okna nauki dla robota
6. Przełącznik detekcji kolizji robota z obiektami zewnętrznymi.

- funkcje obsługi obiektów CAD:
1. przełącznik globalnego układu współrzędnych dla obiektu CAD
2/3. wybór następnego/poprzedniego obiektu w aktywnej grupie
4/5 wpisanie pozycji absolutnej/względnej dla wybranego obiektu
6. zerowanie pozycji dla obiektu jw.
7. zapamiętanie obiektu jw.

. - menu poruszania robotem
1. ruch do pozycji bazowej
2. ruch w pozycji złączowej
3. ruch do pozycji absolutnej w kartezjańskim (globalnym) układzie współrzędnych

Menu boczne (po prawej stronie):

To menu z reguły daje więcej możliwości niż jedna funkcja bądź dwie. Aktywacja poszczególnych funkcji następuje tu poprzez naciskanie ciągłe lewego<L>/prawego<P> przycisku myszy (rolę przycisku może odgrywać też przycisk przewijania) bądź obu naraz <L+P> dla podświetlonej wybranej ikony.

1. Operowanie widokiem sceny:
<L> obrót sceny
<P> zoom
<L+P> ruch sceny we współrzędnych kartezjańskich

2. Ruch robota we współrzędnych związanych z narzędziem (TCP).
3. Jw. dla ruchu we współrzędnych globalnych.
4. Ruch robotem w poszczególnych osiach.
Podwójne kliknięcie pozwala na przełączenie między grupami osi 1-3 i 4-6 oraz wybór dodatkowych funkcji:
- aktywacja detekcji kolizji, wyłączników krańcowych, ruchu w układzie TCP i globalnym.
Ponadto można wybrać do wykonania komendę z predefiniowanego zbioru.
5. Ruch robota względem podłogi.
6. Przesunięcie(translacja) < L>/ obrót <P>
wybranego obiektu.
7. Poruszanie wybranej pozycji docelowej (tzw. Tag-u)
8. Praca z obiektami.
9. Przełącznik między obrotem a translacją. (ustawienia istotne dla pozostałego menu)
10. Przełącznik współrzędnych.
11. Ruch robotem z zachowaniem pozycji środka narzędzia (TCP).
12/13. Zwiększenie/zmniejszenie kroku w trybie nauki

3. Rozwijalne Menu Tekstowe.

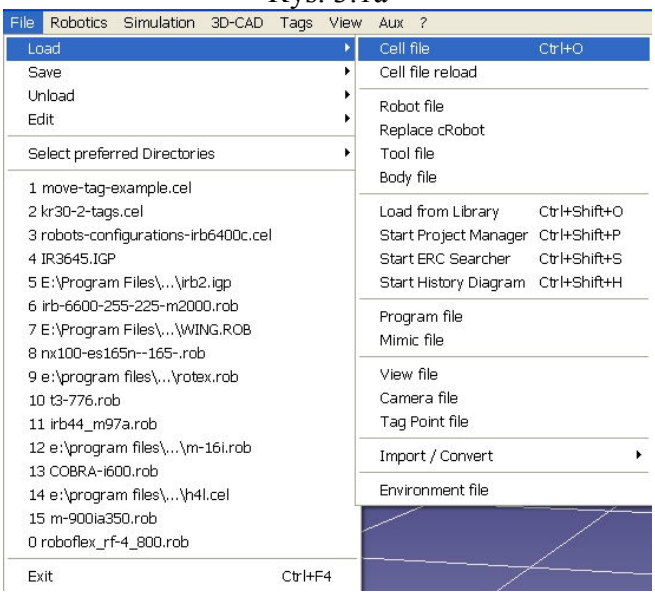
Menu to ma znacznie więcej funkcji niż opisane w punkcie 2. często zresztą je dublując. Z tego względu zostaną tu omówione wybrane pozycje związane bezpośrednio z ciągiem czynności jakie powinien wykonać student w czasie symulacji pracy robota - zatem opis jest tu bardziej chronologiczny niż funkcjonalny.

Ogólnie chronologia czynności - jak wspomniano na wstępie - podczas modelowania pracy robota przedstawia się następująco:

A. *Załadowanie robota.*
B. Napisanie programu.
C. Odtworzenie wraz z monitoringiem jego zachowania oraz wybranych wartości.

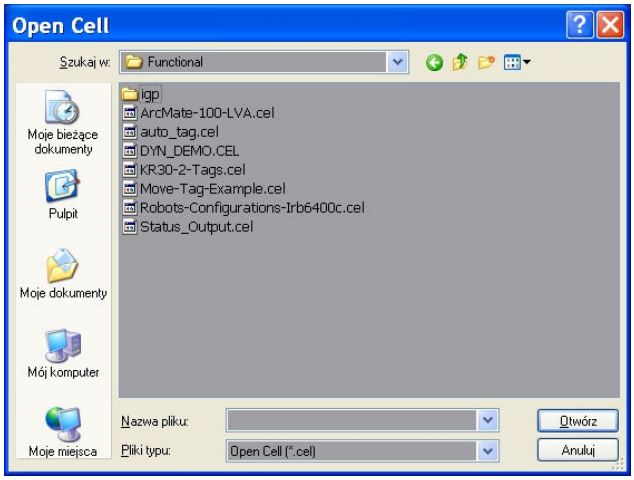
3.1 Pierwszą czynnością jaką należy wykonać po uruchomieniu programu jest załadowanie robota wraz z jego środowiskiem i przypisanym mu programem (Rys 3.1).
Umożliwia to funkcja ładowania (także pierwsza górna ikona) pliku z rozszerzeniem *.cel, która zawiera te elementy. Elementy mogą być też ładowane indywidualnie:
Robot: plik : *.rob, program:plik *.prg itd.

Rys. 3.1a



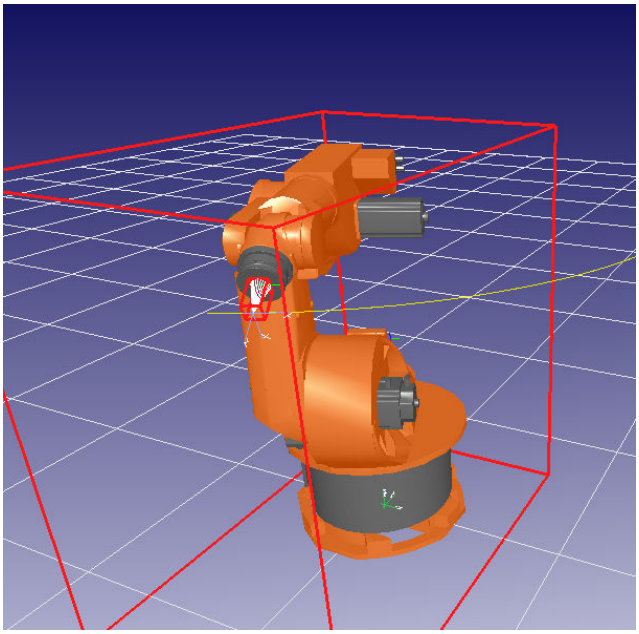
.

Rys 3. 1b

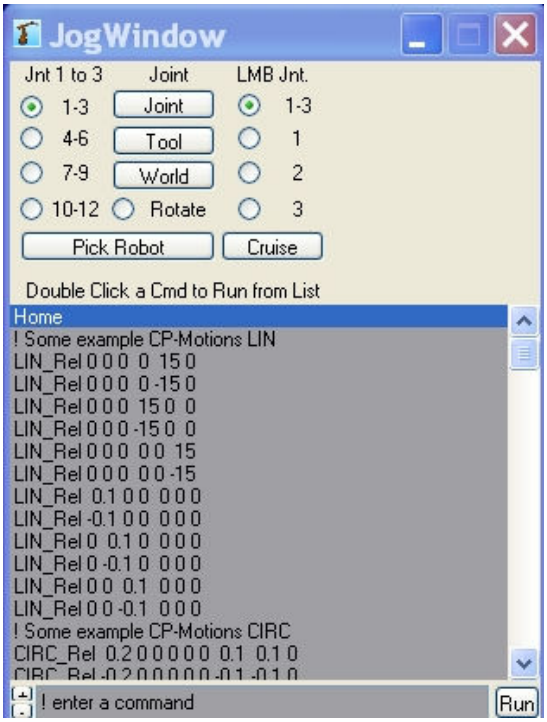


3.2 Po załadowaniu należy określić strukturę robota, zakresy jego ruchów (przestrzeń roboczą) oraz położenie w globalnym układzie współrzędnych. Tym obiektem dla którego ruchy są programowane jest najczęściej narzędzie, dokładniej, jego punkt odniesienia (*TCP*).

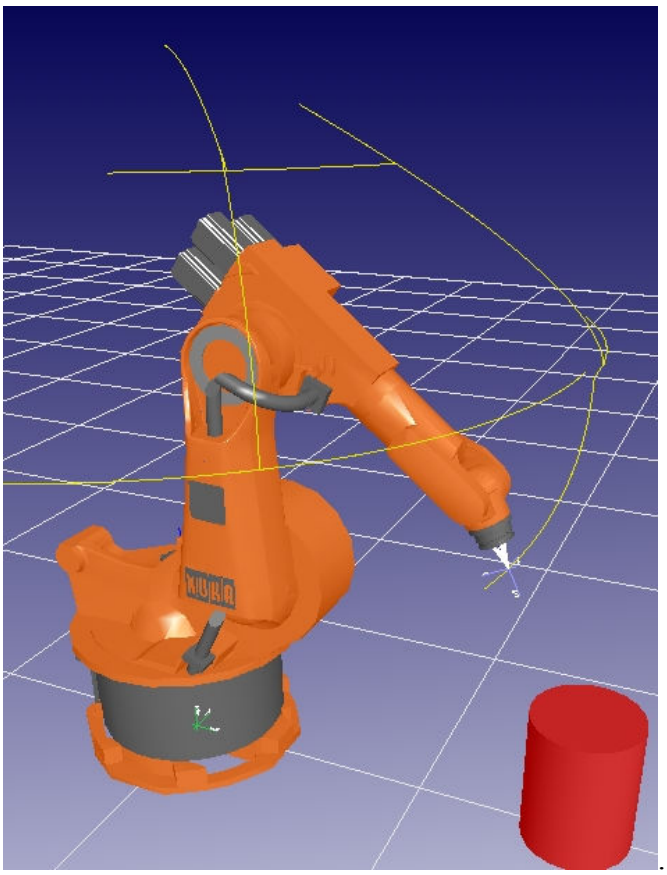
Te zależności można zidentyfikować poruszając robotem w wybranych jego osiach, wybierając z menu bocznego pozycję **Robot Joints** uprzednio opisaną w rozdziale 2.
Na rysunku 3.2.2 widzimy efekt poruszania kolumną robota < L > (słabo widoczny tu ślad kreślony przez koniec narzędzia). Wykorzystując pozostałe kombinacje przycisków myszy < P > < L+P >
możemy otrzymać ruchy w osiach 2 i 3.
Podwójne kliknięcie pozwala uczynić widoczne okno Rys 3.2.4 wykorzystując to menu Jog - możemy przełączyć na ruch w osiach 4-6 (rys. 3.2.3b) oraz ustalić warunki tego ruchu. Użyteczną funkcją jest tu także możliwość wyboru jednej z predefiniowanych komend do jednorazowego wykonania.



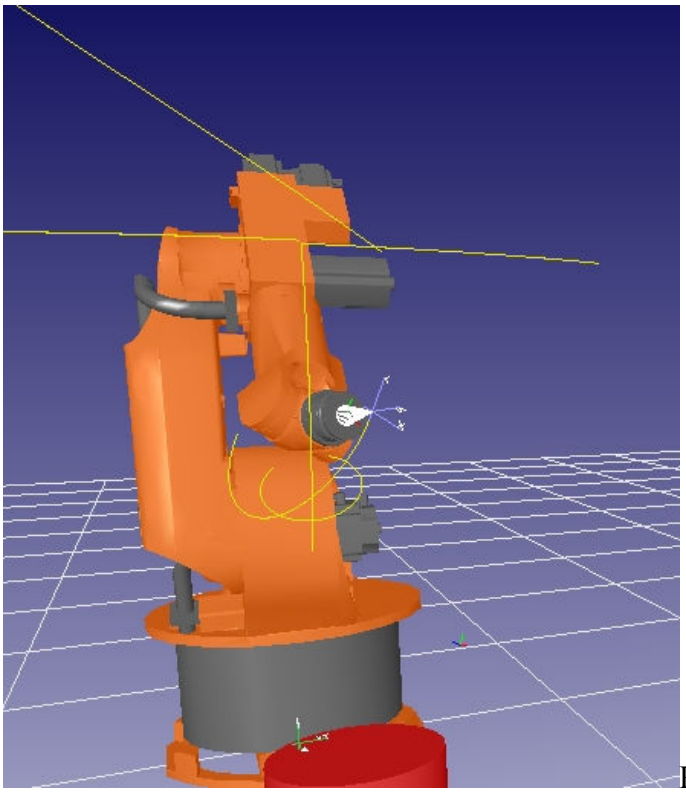
Rys. 3.2.2 Ruch kolumną



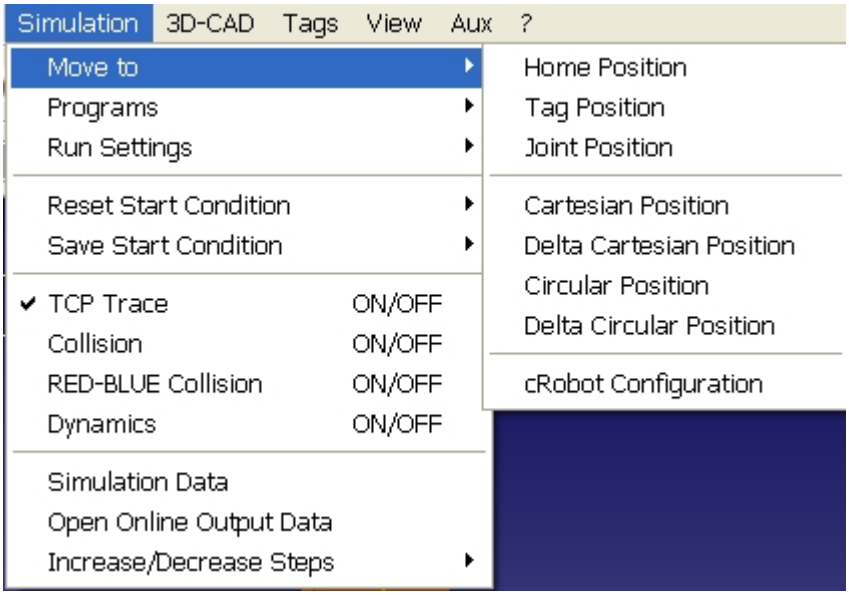
Rys. 3.2.4 Okno Jog



. Rys.3.2.3a Ruch w osiach 1-3



Rys. 3.2.3b Dodatkowo ruch w osiach – 4-6

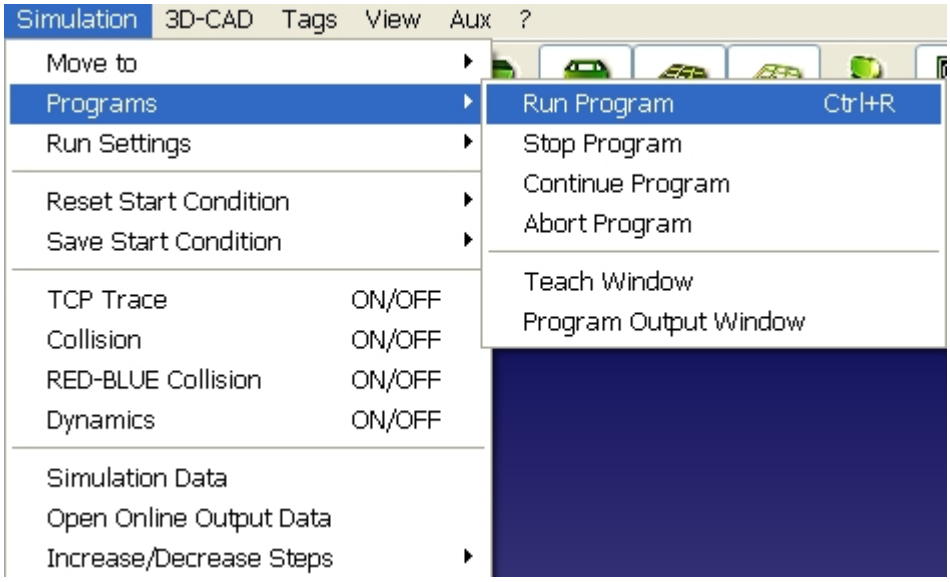


Możliwość generowania ruchu mamy także z dolnego paska menu bądź z menu tekstowego *"*****Simulate** ***Move to**" - rys 3.2.5 .

Ruch może być również wygenerowany poprzez załadowanie i wykonanie wybranego programu, co umożliwia menu **Robotics\Robot Program\LoadProgram------Run Program**

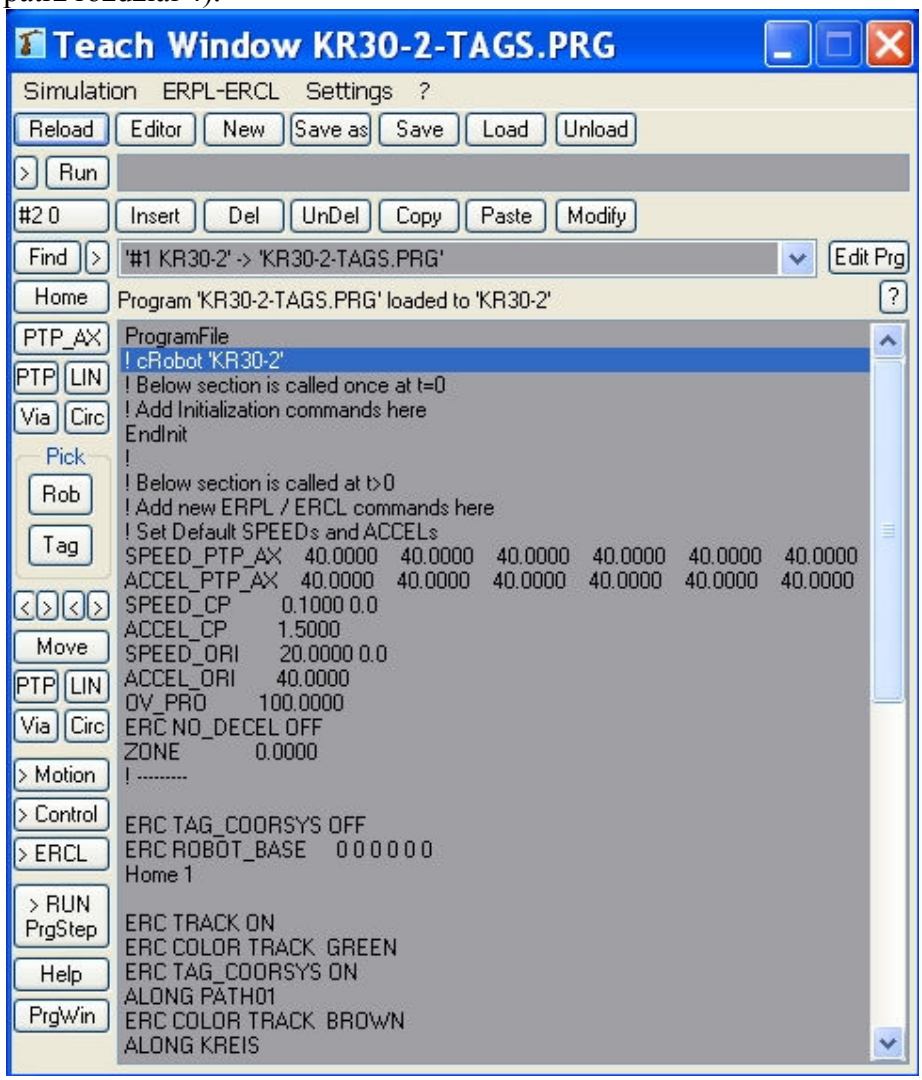
Z tego menu można też wywołać standardowy edytor tekstowy do edycji tego programu: **Edit Program**. Edycja programu w tym przypadku jest zwykłą edycją tekstową dlatego należy zwracać uwagę na dopuszczalną składnię rozkazów w celu uniknięcia błędów syntaktycznych.

Przygotowany program może zostać następnie uruchomiony. Najwygodniej jest tu korzystać z górnego paska menu, ale można też to wykonać z poziomu menu tekstowego: ***"\Simulate\Program's\Run Program***"(rys. 3.2.7↓).

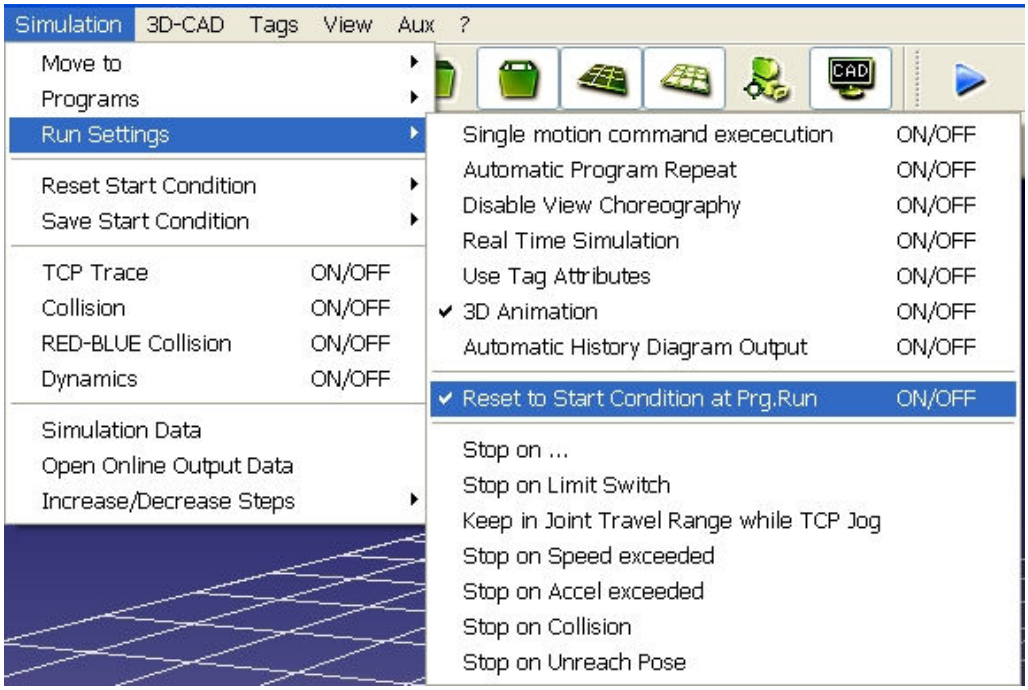


.

Wykorzystując to menu możemy też przeprowadzić edycję programu korzystając z **Teach Window**
(rys. 3.2.8↓). Korzystanie z tej opcji edycji zabezpiecza przed wprowadzeniem błędów syntaktycznych do tekstu programu ; poza tym możemy bezpośrednio wybierać dostępne komendy (patrz rozdział 4).



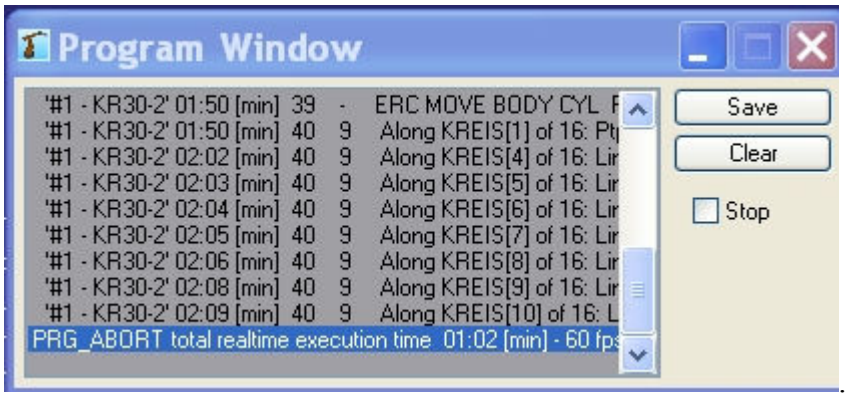
.



Sposób odtworzenia programu zależy od ustawień wybranych w "**Simulate\Run Settings**\" (rys. 3.2.9↑), które możemy też zmienić w górnym pasku menu.

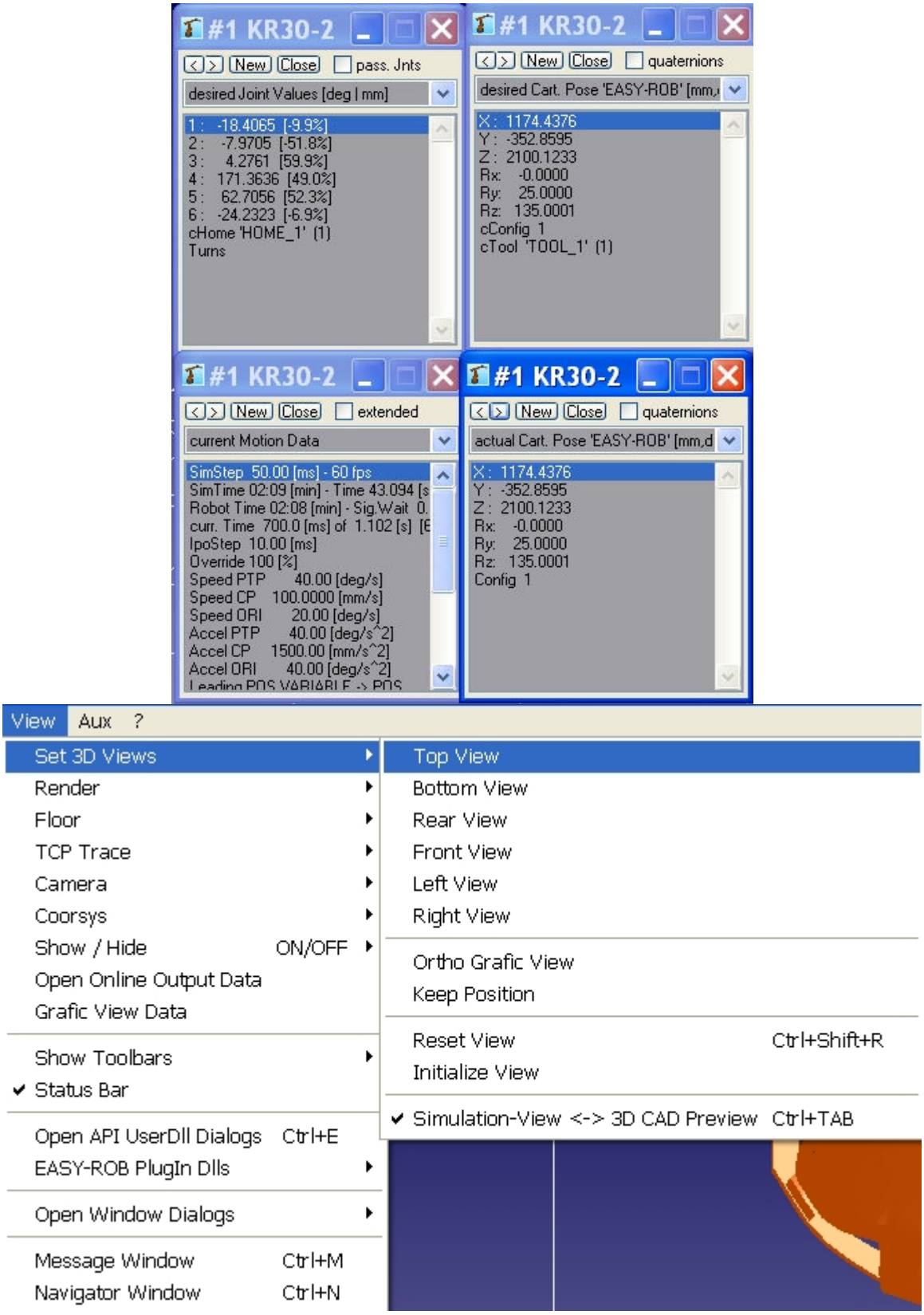
Wykonanie programu możemy śledzić na kilku poziomach:

- w warstwie wykonania poszczególnych komend (rys. 3.2.10↓) - **Simulate\Pogram's\Program Output**"



- w warstwie monitoringu wybranych wielkości fizycznych okno **io_ouput**, które najwygodniej uaktywnić w górnym pasku menu , skrajna pozycja prawa. (bądź **Simulate\Online output Data**)

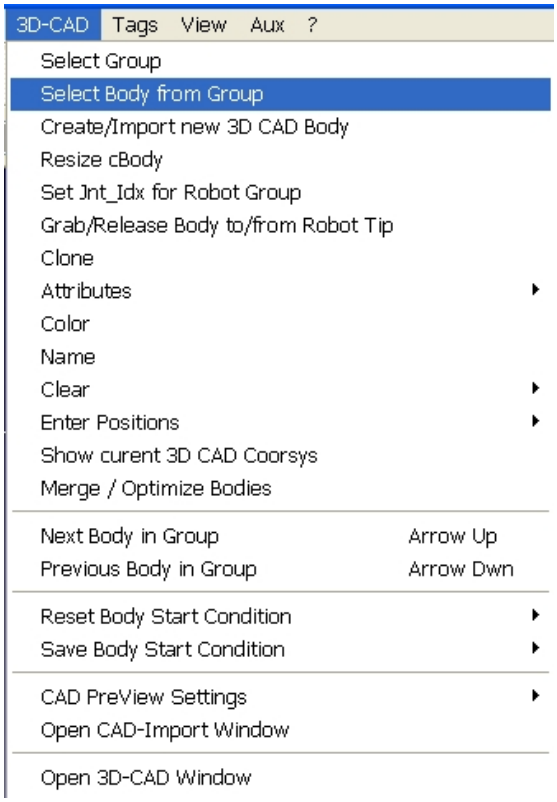
Okno to w zasadzie jest zbiorem 15 różnych okien które mogą być kolejno uaktywniane bądź kasowane w zależności od potrzeb. (Rys. 3.2.11 ↓- 4 wybrane okna)



- w warstwie wizualnej obserwując np. czy nie ma kolizji między robotem a otoczeniem

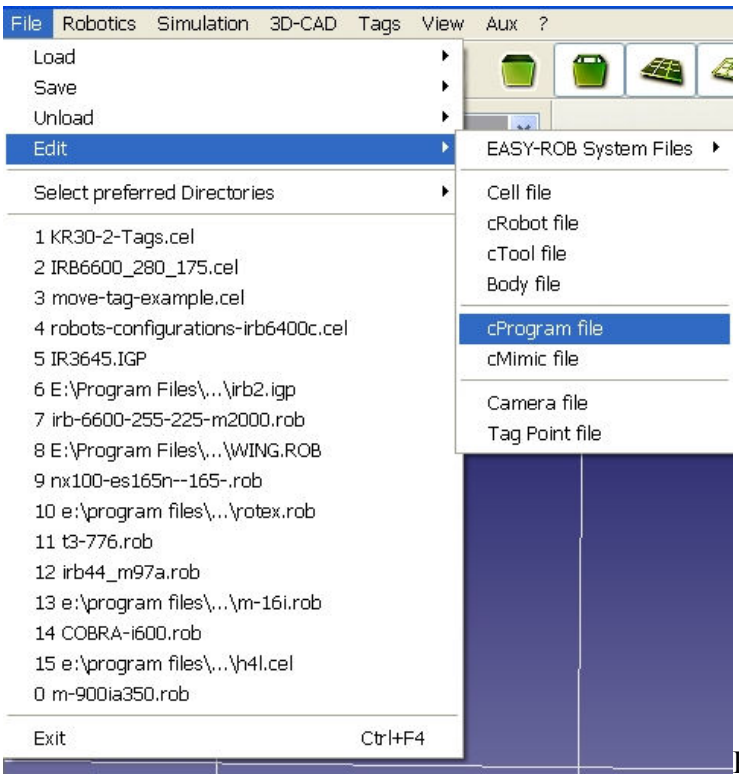
Dostosowanie widoku do wymagań użytkownika można wykonać korzystając z menu: **View\Set 3D view** i wybrać np. widok z góry (**Top View**) (Rys. 3.2.12).

Inną funkcją która może pomóc uczynić np. strukturę robota bardziej jawną jest aktywacja menu
3D-CAD\select group a następnie **3D-CAD\select Body from Group** np. dla zmiany koloru części robota **..\Color** (rys. 3.2.13).



W tej grupie menu znajduje się też ważna pozycja ***\Grab\Release Body to/from Robot Tip*** umożliwiająca chwytanie/ zwalnianie obiektu przez robota.
Należy tu powiedzieć, że sama operacja chwytu jest zaimplementowana w sposób czysto matematyczny i polega na "wklejeniu" obiektu w układ współrzędnych narzędzia robota i trwałe z nim związanie. "Chwycenie" zatem odbywa się zatem w każdych warunkach, nawet przy braku fizycznego kontaktu miedzy robotem a obiektem, a ustalenie warunków poprawnego chwytu spoczywa całkowicie na programiście, i nie jest sprawdzane w żaden sposób automatycznie.

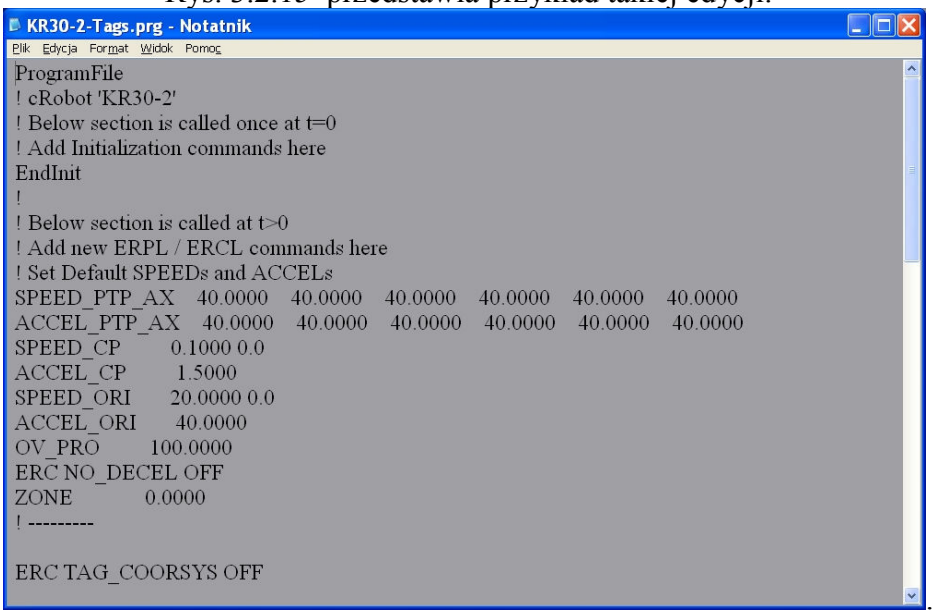
Powróćmy teraz do edycji programu.
Jak wspomniano edycji możemy dokonywać używając standardowego edytora tekstowego, chociaż zalecane jest korzystanie z **Teach Window**.
W pierwszym przypadku można edytor wywołać także z menu **File\Edit\Loaded Program** (rys. 3.2.14)



Rys 3.2.14

.

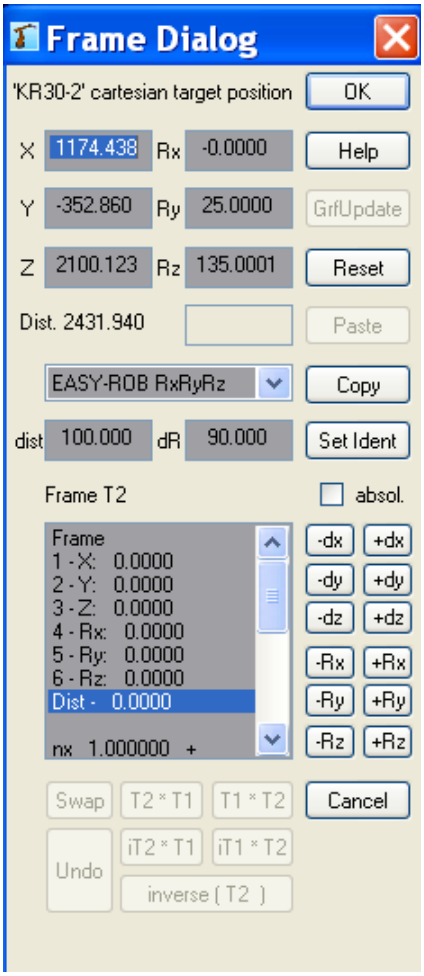
Rys. 3.2.15 przedstawia przykład takiej edycji.

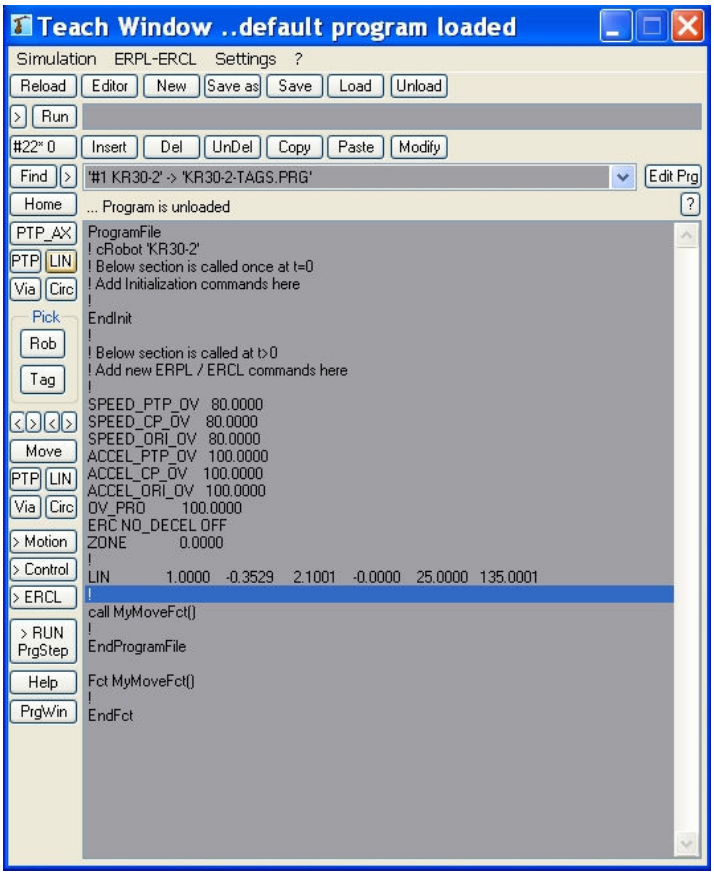


.

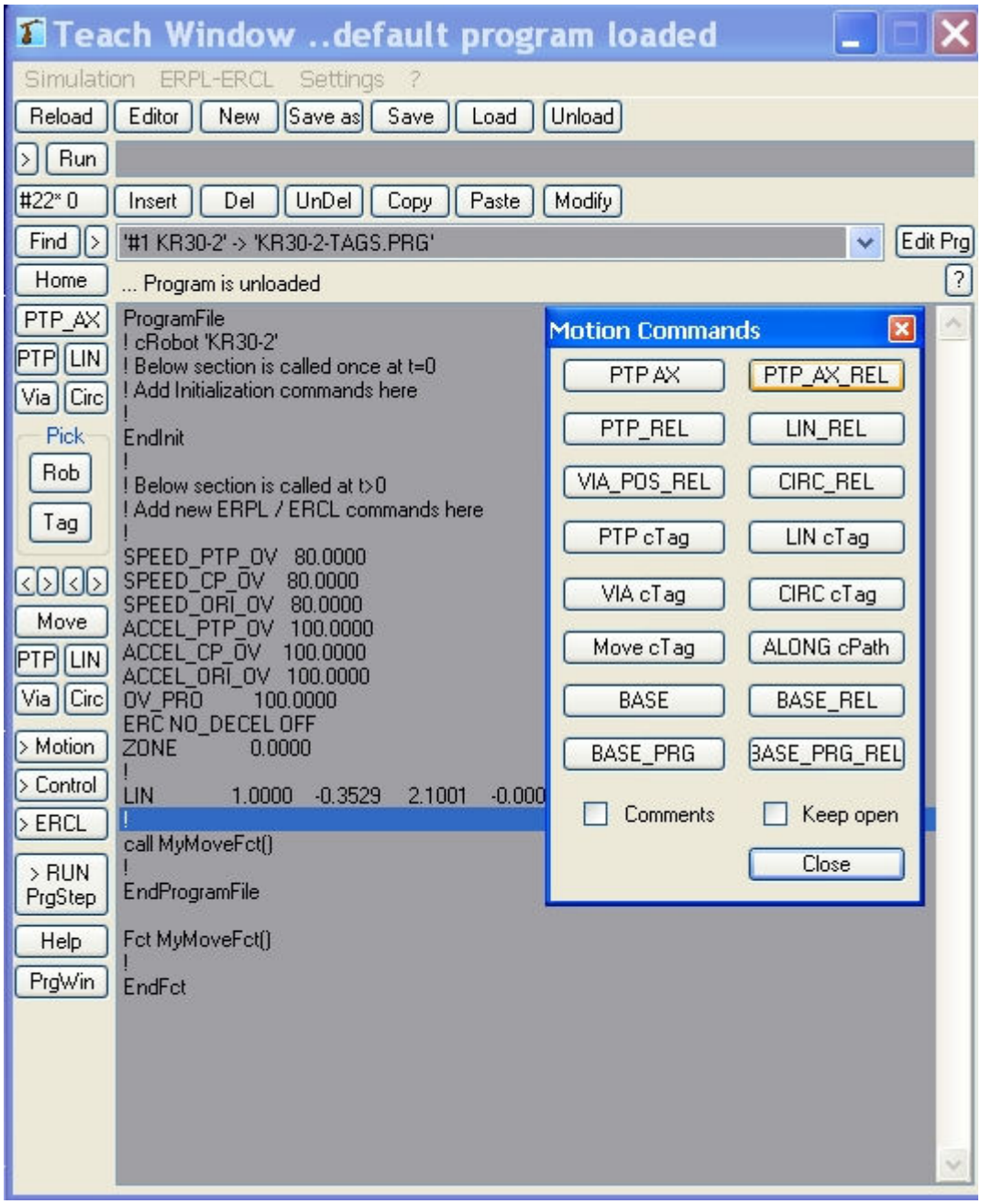
Wygodnym sposobem programowania ruchu robota jest wygenerowanie tego ruchu z menu bocznego w globalnym układzie kartezjańskim poprzez podanie trzech wartości współrzędnych oraz trzech wartości kątów orientacji (są to wielkości zawsze związane z układem współrzędnych TCP).

Po uzyskaniu zadanej pozycji może być ona wpisana bezpośrednio do programu za pomocą kliknięcia na wybrany typ ruchu np. liniowy (LIN) jak na rysunku poniżej, gdzie została zaprogramowana zmiana pozycji w osi X na wartość 1.000.

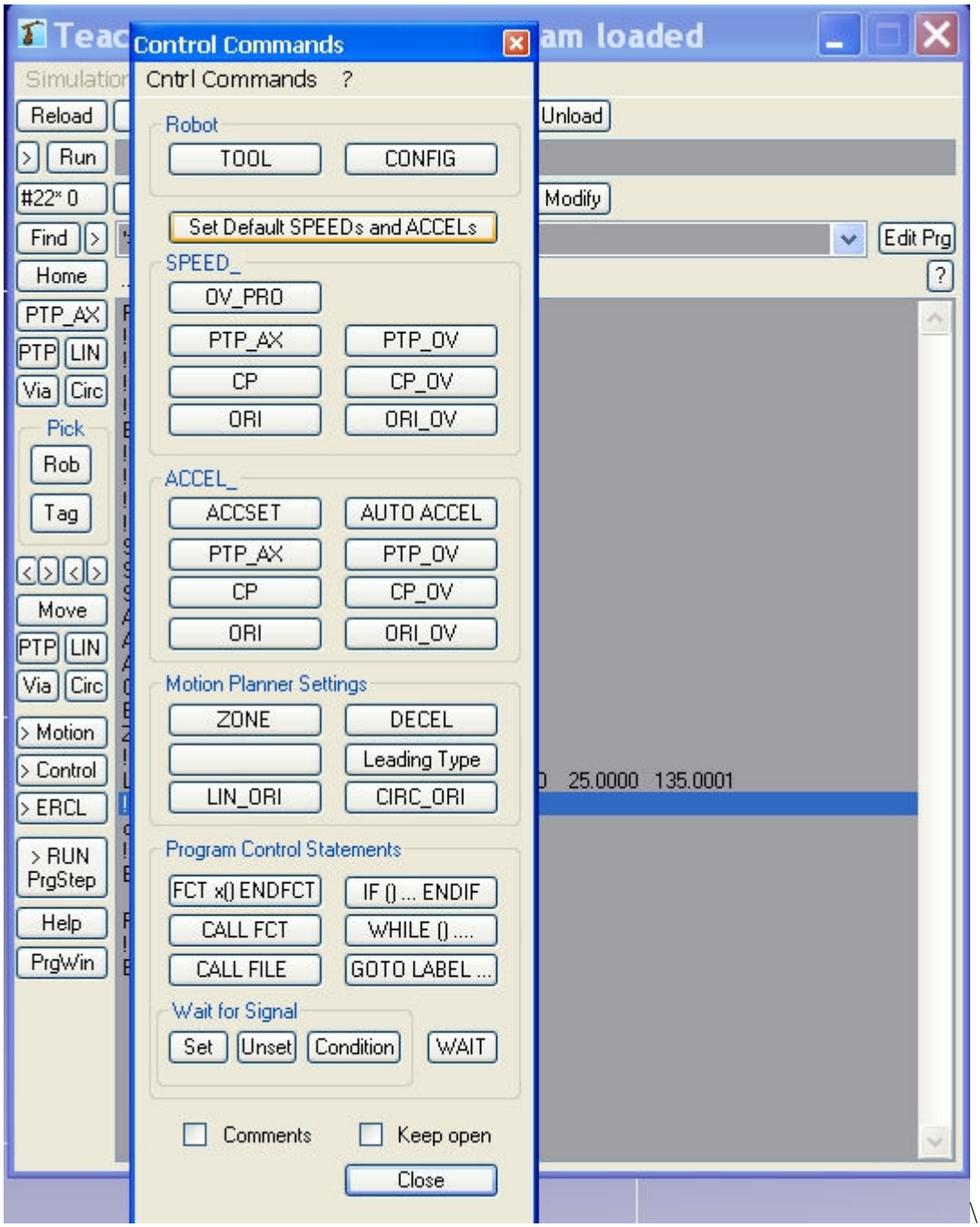

Frame Dialog
'KR30-2' cartesian target position
OK
X 1174.438 Rx -0.0000
Help
Y -352.860 Ry 25.0000
GrfUpdate
Z 2100.123 Rz 135.0001
Reset
Dist. 2431.940
Paste
EASY-ROB RxRyRz
Copy
dist 100.000 dR 90.000
Set Ident
Frame T2
absol.
Frame
1 - X: 0.0000
2 - Y: 0.0000
3 - Z: 0.0000
4 - Rx: 0.0000
5 - Ry: 0.0000
6 - Rz: 0.0000
Dist - 0.0000
nx 1.000000 +
-dx +dx
-dy +dy
-dz +dz
-Rx +Rx
-Ry +Ry
-Rz +Rz
Swap
T2 * T1
T1 * T2
Cancel
Undo
iT2 * T1
iT1 * T2
inverse (T2)


Teach Window ..default program loaded
Simulation ERPL-ERCL Settings ?
Reload Editor New Save as Save Load Unload
Run
#22* 0 Insert Del UnDel Copy Paste Modify
Find
'#1 KR30-2' -> 'KR30-2-TAGS.PRG'
Edit Prg
Home
... Program is unloaded
PTP_AX
PTP LIN
Via Circ
Pick
Rob
Tag
Move
PTP LIN
Via Circ
> Motion
> Control
> ERCL
> RUN PrgStep
Help
PrgWin
ProgramFile
! cRobot 'KR30-2'
! Below section is called once at t=0
! Add Initialization commands here
!
EndInit
!
! Below section is called at t>0
! Add new ERPL / ERCL commands here
!
SPEED_PTP_OV 80.0000
SPEED_CP_OV 80.0000
SPEED_ORI_OV 80.0000
ACCEL_PTP_OV 100.0000
ACCEL_CP_OV 100.0000
ACCEL_ORI_OV 100.0000
OV_PRO 100.0000
ERC NO_DECEL OFF
ZONE 0.0000
!
LIN 1.0000 -0.3529 2.1001 -0.0000 25.0000 135.0001
!
call MyMoveFct()
!
EndProgramFile
Fct MyMoveFct()
!
EndFct

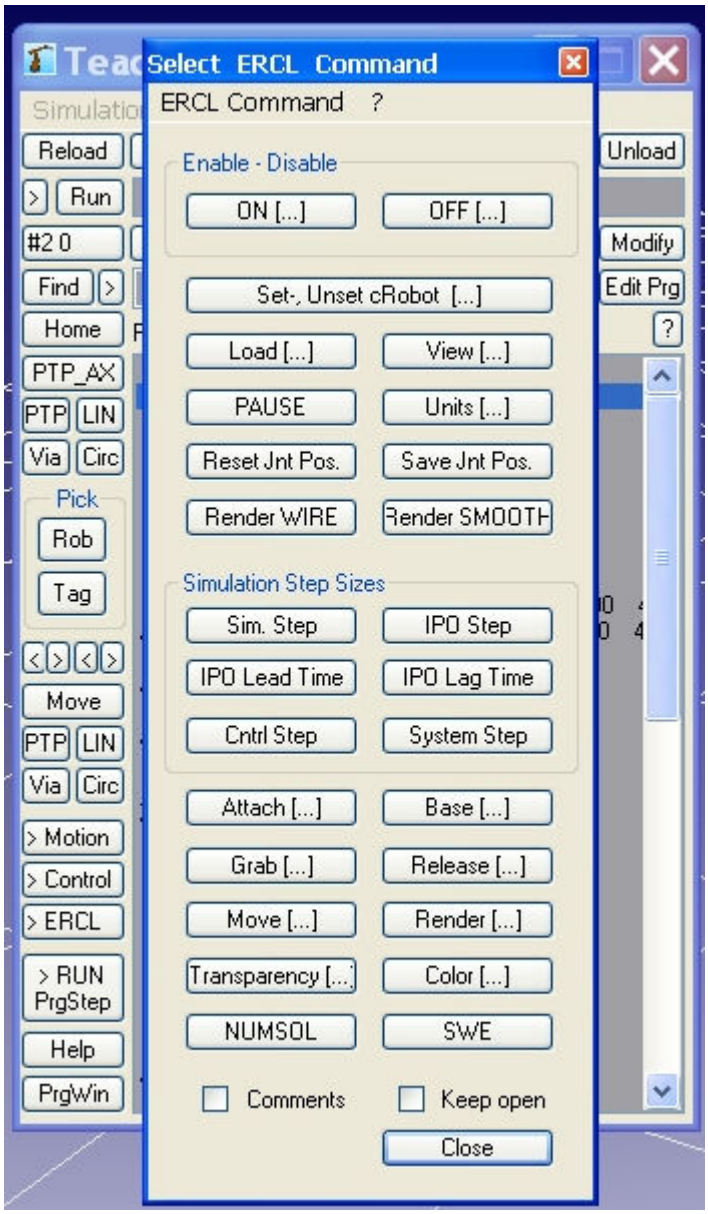
Okno nauki umożliwia dostęp do różnorodnych rozkazów które zostały pogrupowane w 3 główne grupy.
Pierwszą grupę stanowią rozkazy ruchu.



Druga ważna grupa to rozkazy sterowania, gdzie można ustawić wiele z wartości domyślnych odtwarzania trajektorii (prędkość, przyśpieszenie) a nawet wprowadzać zmienne sterujące przebiegiem programu.



Trzecią grupę stanowią rozkazy języka komend, wśród których najistotniejsze są przełączniki On/Off, wśród których między innymi znajduje się przełącznik widoczności śladu kreślonego przez końcówkę robota.
W tej grupie znajdują się też rozkazy chwytania (Grab) i zwalniana obiektów (Release) przez robot.



Podstawą programowania jest zatem zbiór dostępnych składający się na język programowania robotów - ERPL [Easy Robot Programming Language] oraz ERCL [Easy Robot Command Language szczegółowemu opisowi wybranych komend poświęcony jest następny rozdział.

4. Język programowania robotów.

Język programowania robotów *ERPL* jest językiem zbliżonym do języków stosowanych w programowaniu robotów uzupełnionym o specyficzne rozkazy *ERCL* pozwalające efektywnie wizualizować pracę robota wg potrzeb użytkownika.

Składa się z kilku grup rozkazów. Omówione tu będą te najważniejsze:

Komendy ruchu:

SPEED_CP dx dxe[m/s] - ustalenie prędkości dla ruchu typu CP (Continous Path)
SPEED_PTP v ve[m,deg] - ustalenie prędkości dla ruchu typu PTP
(Point to Point)
OV_PRO x [%] - dotyczy podanej w procentach (0 -200%) możliwości przekraczania prędkości

CONFIG n [] - ustawienie aktywnej konfiguracji. Warto zawsze w sposób jawny na początku przypisać np. *config 1*.
Konfiguracja określa sposób przeliczania trajektorii , może przyjmować różne wielkości dla różnych robotów. Dla robota 6-DOF (o sześciu stopniach swobody) jest z reguły 8 różnych konfiguracji i wybór liczby od 1 do 8 zmienia czasem w istotny sposób wykonania komendy ruchu.

TOOL X Y Z A B C *[m,deg]* - ustalenie położenia narzędzia w stosunku do końcowego punktu robota.

EXT_TCP X Y Z A B C [m,deg]

BASE X Y Z A B C *[m,deg]* - układ bazy dla punktu docelowego w układzie globalnym
BASE_REL X Y Z A B C [m,deg]- jw. ale w relacji do TCP
BASE_PRG X Y Z A B C [m,deg]
BASE_PRG_REL X Y Z A B C [m,deg]

HOME n [] - operacja bazowania do wybranej konfiguracji
PTP X Y Z A B C *[m,deg*]- ruch typu PTP do pozycji jak podają parametry w układzie globalnym
PTP_AX q1 .. qn *[m,deg*]- ruch PTP w osiach do zadanych kątów
PTP_AX_REL q1 .. qn *[m,deg]*- jw. z tym że ruch względem aktualnej pozycji o zadane kąty
PTP_REL dX dY dZ dA dB dC *[m,deg]*- ruch względny PTP odniesiony do układu narzędzia TCP
PTP tagname - ruch PTP do wskazanej pozycji docelowej (tzw. tag-u)

LIN_ORI VARIABLE, FIX, TANGENTIAL - ustalenie orientacji względem trajektorii ruchu liniowego.
LIN X Y Z A B C *[m,deg]*- ruch liniowy do punktu w układzie globalnym
LIN_REL dX dY Dz dA dB dC *[m,deg]*- liniowy ruch względny
LIN TagName - ruch liniowy do wskazanej pozycji docelowej

Podobne znaczenie jak powyżej mają instrukcje ruchu kołowego

CIRC_ORI VARIABLE, FIX, TANGENTIAL

Odróżnia je od poprzednich potrzeba wskazania punktu pośredniego, przez który ruch będzie wykonywany.

VIA_POS X Y Z A B C *[m,deg]*- wskazanie punktu pośredniego
VIA_POS_REL dX dY dZ dA dB dC *[m,deg]* - jw. ale relatywnie do aktualnej pozycji
VIA_POS TagName - poprzez punkt pobrany z konstraktu TagName

Sam ruch z klasyfikacją jak w PTP i LIN

CIRC X Y Z A B C *[dX2 dY2 dZ2] [m,deg]*
[X2 Y2 Z2] via point
CIRC_REL dX dY dZ dA dB dC *[dX2 dY2 dZ2] [m,deg]*
[dX2 dY2 dZ2] via point
CIRC TagName

Jest to ruch kołowy do pozycji docelowej wskazanej w parametrach instrukcji , która powinna z reguły być poprzedzona wskazaniem punktu przez który ma przechodzić trajektoria. Patrz ponadto przykład na zakończenie rozdziału.

Komendy pomocnicze:

MSG "" – wysłanie komunikatu tekstowego
WAIT x *[sec]* – oczekiwanie przez zadany czas [s]

Wywołanie i definicje podprocedur:

CALL fct_name()
FCT fct_name()
ENDFCT

Komendy specyficzne dla środowisku programowania EASY-ROB:

ERC 'EASY-ROB Command'

ERC SET_DEFAULTS
ERC SIM_STEP x [sec]
ERC CNTRL_STEP x [sec]
ERC SYSTEM_STEP x [sec]
ERC IPO_STEP x [sec]
ERC IPO_LEAD_TIME x [sec]
ERC IPO_LAG_TIME x [sec]

ERC TRACK ON,OFF - uaktywnienie/ dezaktywacja śladu jaki zostawia narzędzie w środowisku
ERC DYNAMICS ON,OFF
ERC STOP_SWE ON,OFF - uaktywnienie/ dezaktywacja wyłączników krańcowych
ERC COLLISION ON,OFF - uaktywnienie/ dezaktywacja detekcji kolizji
ERC STOP_COLLISION ON,OFF - uaktywnienie/ dezaktywacja stopu po wykrytej kolizji

ERC ROBOTJOINTS ON,OFF - wyświetla(ON) pozycje złączowe
ERC ROBOTPOSITIONS ON,OFF - jw. ale w układzie kartezjańskim

ERC PRG_WIN ON,OFF - wyświetla okno programu

Komendy wizualizacji:

ERC FLOOR ON,OFF
ERC FLOOR_RENDER ON,OFF
ERC EXT_TCP ON,OFF
ERC ORTHOGRAFIC ON,OFF
ERC DISPLAY_ROBOT ON,OFF
ERC DISPLAY_ROBOT_COORSYS ON,OFF
ERC DISPLAY_TOOL ON,OFF
ERC DISPLAY_BODYS ON,OFF
ERC TCP_COORSYS ON,OFF
ERC BASE_COORSYS ON/OFF
ERC CREATE_TARGET_TAGS ON/OFF
ERC RESET_ALL_POSITIONS_JOINTS ON/OFF
ERC NO_DECEL ON/OFF
ERC GRAFIC_UPDATE ON/OFF
ERC DISPLAY_TAGS ON/OFF
ERC STATUS_OUTPUT ON/OFF [1-at simstep,2-at target pose] [flnname] [fct# 0-12]

ERC RENDER FLAT
ERC RENDER WIRE
ERC RENDER BBOX ON,OFF
ERC RENDER BODY [ROBOT,TOOL] bodyname WIRE [FLAT,BBOXWIRE,BBOXFLAT,INVISIBLE]
ERC RENDER BODY_GRP [ROBOT_GRP,TOOL_GRP] WIRE [FLAT,BBOXWIRE,BBOXFLAT,INVISIBLE]

_____
Zmiany koloru robota, narzędzia czy obiektu w odpowiednim miejscu programu mogą pomóc uzyskać bardziej efektywną wizualizację:

ERC COLOR BODY [ROBOT,TOOL] bodyname color
ERC COLOR BODY_GRP [ROBOT_GRP,TOOL_GRP] color
ERC COLOR TRACK [TRACK_DYN] color - możliwość zmiany koloru śladu
ERC COLOR TAG color
color = [BLUE, GREEN, CYAN, RED, MAGENTA, BROWN, LIGHTGRAY, DARKGRAY, LIGHTBLUE,
LIGHTGREEN, LIGHTCYAN, LIGHTRED, LIGHTMAGENTA, YELLOW, WHITE]

Dodatkowe komendy związane ze stanem robota:

ERC STOP - bezwarunkowy STOP
ERC RESET JOINTPOSITION - zerowanie współrzędnych złączowych
ERC SAVE JOINTPOSITION

Komendy ładowania:

ERC LOAD TOOL fln (nazwa pliku)
ERC LOAD VIEW fln
ERC LOAD ROBOT fln
ERC LOAD BODY fln
ERC LOAD RECORDING fln
ERC LOAD ENVIRONMENT fln | DEFAULT

ERC RUN_RECORDING [recordingfile.rec] - uruchamia nagrywanie trajektorii do wskazanego pliku

Komendy ruchu
dotyczące odpowiednio obiektu (body), narzędzia (tool) i całego robota(jego bazy) względem układu globalnego: do pozycji absolutnej lub o ruch względny

ERC MOVE BODY bodyname XYZ ABC [m,deg]
ERC MOVE TOOL bodyname XYZ ABC [m,deg]
ERC MOVE ROBOT bodyname XYZ ABC [m,deg]
ERC MOVE_REL BODY bodyname dXdYdZ dAdBdC [m,deg]
ERC MOVE_REL TOOL bodyname dXdYdZ dAdBdC [m,deg]
ERC MOVE_REL ROBOT bodyname dXdYdZ dAdBdC [m,deg]
ERC MOVE_REL BODY_GRP bodyname dXdYdZ dAdBdC [m,deg]
ERC MOVE_REL TOOL_GRP bodyname dXdYdZ dAdBdC [m,deg]
ERC MOVE_REL ROBOT_GRP bodyname dXdYdZ dAdBdC [m,deg]
ERC MOVE_REL LIST listname dXdYdZ dAdBdC [m,deg]

ERC BASE BODY bodyname
ERC BASE TCP

Komendy widoku

ERC VIEW steps n
ERC VIEW hither x
ERC VIEW yonder x
ERC VIEW screen x
ERC VIEW zweight x
ERC VIEW zoom x
ERC VIEW zoom_in x
ERC VIEW zoom_out x
ERC VIEW tcp_rot_tcp ABC
ERC VIEW tcp_rot_world ABC
ERC VIEW world_rot_base ABC
ERC VIEW world_rot_world ABC

Komendy chwytania (grab) , wypuszczenia(release) obiektu identyfikowanego przez nazwę (identyfikacja nazwy poprzez menu **3D-CAD/Select Body from group**)

ERC GRAB BODY 'bodyname' - chwyta jeden obiekt
ERC GRAB BODY_GRP - chwyta całą grupę obiektów
ERC RELEASE BODY 'bodyname' - zwalnia jeden obiekt
ERC RELEASE BODY_GRP - zwalnia całą grupę obiektów

Metodyka pisania programów.

Poniżej został przedstawiony program przygotowany dla "manipulatora" typu podwójne ramię (1dh). Zadaniem manipulatora było wykreślenie trójkąta oraz półokręgów na nim opisanych przy dwóch różnych konfiguracjach. Pozycje zostały uzyskane z wykorzystaniem ręcznego poruszania manipulatorem we współrzędnych kartezjańskich oraz wpisane do programu za pomocą **Teach Window**.

```
ProgramFile
! prgfln dz_1dh3.prg
call my_fct_name()
EndProgramFile

fct my_fct_name()
WAIT        2.0000
CONFIG      1
PTP         0.0252    1.3621    0.0000
90.0000  -133.8538    0.0000
LIN         0.0252    1.3621    0.0000
90.0000  -133.8538    0.0000
LIN        -0.3148    1.3621    0.0000
90.0000  -121.5063    0.0000
LIN        -0.3148    0.9101    0.0000
90.0000   -96.0963    0.0000
LIN         0.0252    1.3621    0.0000
90.0000  -133.8543    0.0000
CIRC_ORI VARIABLE
VIA_POS    -0.3148    1.3621    0.0000
90.0000  -121.5063    0.0000
CIRC       -0.3148    0.9101    0.0000
90.0000   -96.0963    0.0000
 MSG " -------Zmiana konfiguracji--------"
CONFIG      2
 MSG " ------Wymaga ruchu PTP------"
PTP        -0.3148    0.9101    0.0000
90.0000   -96.0963    0.0000
LIN         0.0252    1.3621    0.0000
90.0000  -133.8538    0.0000
LIN        -0.3148    1.3621    0.0000
90.0000  -121.5063    0.0000
LIN        -0.3148    0.9101    0.0000
90.0000   -96.0963    0.0000
LIN         0.0252    1.3621    0.0000
90.0000  -133.8543    0.0000
CIRC_ORI VARIABLE
VIA_POS    -0.3148    1.3621    0.0000
90.0000  -121.5063    0.0000
CIRC       -0.3148    0.9101    0.0000
90.0000   -96.0963    0.0000
```

Przy pisaniu tego typu programów zaleca się zwracać uwagę na następujące kwestie:

1. Program jest plikiem tekstowym, ale edycja tekstowa nie daje gwarancji, że wpisany punkt został poprawnie , gdyż:
- współrzędne mogą być dla danego modelu robota wzajemnie zależne (tutaj 5-ta współrzędna .(kąt) zależy od 2 pierwszych współrzędnych metrycznych)
- mogą być poza jego zakresem

Z tego względu opcja **teach window** jest najbardziej zalecana dla uzyskania poprawnych współrzędnych.

2. Przy programowaniu niektórych typów ruchów np. LIN i CIRC należy zwrócić uwagę na to czy jest on wykonywalny bez zmiany konfiguracji. Konieczność jej zmiany w trakcie wykonywania programu prowadzi tu do błędu.

3 Z powyższego względu zaleca się w programie jawną deklaracją aktualnej konfiguracji.

4. Te same punkty mogą służyć dla wygenerowania różnych typów ruchów, a więc konstrukcja instrukcji może polegać jedynie na skopiowaniu parametrów i zamianie typu ruchu np. LIN na CIRC jak w przykładzie powyżej.
5. Należy pamiętać, że instrukcja CIRC powinna być poprzedzona wskazaniem punktu pośredniego czyli instrukcją VIA_POS. Kolejność trzech punktów aktualnego, pośredniego i docelowego wyznacza łuk skierowany wzdłuż którego zostanie wykonany ruch.

5. Program ćwiczeń:

Uwagi ogólne:

*Wszelkie tworzone samodzielnie pliki programów powinny być umieszczane wyłącznie w folderze Proj**Proj_Usr** a ich nazwy powinny zaczynać się od stałego identyfikatora grupy np. dla grupy RG3 pierwszy plik RG3_dha.prg, itp. Nie należy modyfikować plików źródłowych z folderu Proj/Proj*

Zapoznać się z możliwościami programu Easy_Rob śledząc poszczególne punkty niniejszej instrukcji, następnie wykonać następujące zadania:

Wczytać któryś z robotów 6-DOF wraz ze środowiskiem pracy np. paleta.cel.

1. Zapoznać się ze sposobem programowania robota – różnymi sposobami osiągania przez niego zadanego położenia i orientacji.
2. Z elementów rozmieszczonych na palecie ułożyć konstrukcję wskazaną przez prowadzącego.
3. Zmieniając narzędzie na narzędzie do spawania zespawać daną konstrukcję.

.