Silviu RĂILEANU

Utilizarea modelelor digitale pentru proiectarea structurilor și aplicațiilor robotizate

Aplicații în RobotStudio

Silviu RĂILEANU

UTILIZAREA MODELELOR DIGITALE PENTRU PROIECTAREA STRUCTURILOR ȘI APLICAȚIILOR ROBOTIZATE

Aplicații în RobotStudio

EDITURA POLITEHNICA PRESS
București, 2020

Adresa: Calea Griviţei, nr. 132
10737, Sector 1, Bucureşti
Telefon: 021 402.90.74

Referenți științifici:
Prof. univ. dr. ing. **Theodor BORANGIU**
Conf. dr. Ing. **Florin ANTON**

ISBN 978-606-515-943-3

Utilizarea modelelor digitale pentru proiectarea structurilor și aplicațiilor robotizate. Aplicații în RobotStudio

Conf. Dr. Ing. Silviu RAILEANU,

2020

Cuprins

1. Sisteme de fabricație flexibilă ..... **6.**
2. Descrierea functionalităților RobotStudio ..... **25.**
3. Principii generale de lucru (Procedura pick and place, Modul de lucru grafic)..... **46.**
4. Principii generale de lucru (Limbajul de programare Rapid, Programarea în mod text)..... **65.**
5. Stive multidimensionale..... **95.**
6. Componente active..... **100.**
7. Sisteme multirobot ..... **111.**
8. Proiecte propuse ..... **130.**
9. Proiect rezolvat..... **143.**
10. Bibliografie ..... **150.**

Capitol 1: Sisteme de fabricație flexibilă

Cuprins

1. Introducere 6
2. Interconectare posturi de lucru – resursă de tip bandă transportoare controlata de PLC 8
3. Robot industrial de tip SCARA – Adept Cobra s600 11
4. Robot industrial de tip articulat vertical – Adept Viper 650 14
5. Robot industrial de tip cartezian - Adept Python 16
6. Controller robot 17
7. Automatul programabil 18
9. Sistem de inspectie si ghidare vizuala AdeptSight 21
10. Exemplu de producție 22
11. Norme de protecția muncii și securitate 23
12. Intrebări: 24

1. Introducere

Cartea are ca obiectiv prezentarea unor metode de realizarea a unor modele digitale care vor fi folosite pentru dezvoltarea offline a unor solutii robotizate. Vor fi abordate următoarele aspecte: proiectare sistem, realizare model digital (inclusiv programare traiectorii robot) și validare scenariu/scenarii de lucru.

Cea mai recentă tendință în fabricație constă în cererea unui grad ridicat de personalizare a produselor finite. Cu toate acestea, în sistemele de producție actuale, tranziția către personalizarea în masă generează complexitate și costuri ridicate pe care producătorii trebuie să le gestioneze pentru a se menține competitivi și pentru a avea un proces durabil. Capacitatea de a oferi mai multe variante pe model și de a introduce noi modele mai rapid, este constrânsă de tehnologiile actuale și de resursele de producție în serie.

În trecut, reducerea costurilor și îmbunătățirea calității au fost principalele cerințe impuse companiilor și lanțurilor de aprovizionare pentru a supraviețui și a prospera. Prin urmare, companiile au încercat să reducă costurile și să îmbunătățească în mod simultan calitatea, concentrându-și eforturile pe activitatea de producție. În prezent, flexibilitatea și fiabilitatea se manifestă ca dimensiuni critice ale producției.

Vizând piețe noi, companiile producătoare au introdus conceptul de sistem flexibil de fabricație (SFF, en.: Flexible Manufactuinr System - FMS) pentru a trece peste compromisul clasic dintre fiabilitate și calitate. Sistemele de fabricație flexibile reduc forța de muncă și, în consecință,

variabilitatea proceselor, îmbunătățind calitatea produsului. În același timp, sistemele de fabricație flexibile oferă mai multe alternative de producție existând astfel posibilitatea de a trece peste perioadele de mentenanță și de reactualizarea a stocurilor fără întreruperi, sporind astfel fiabilitatea livrărilor. Rezumând, sistemele de fabricație flexibile crește productivitatea generală, îmbunătățește calitatea produsului final și, în același timp, reduce vulnerabilitățile din cauza variațiilor neașteptate ale volumului, ale componenței și a termenelor scadente ale comenzilor.

Sistemele flexibile de fabricație oferă 3 tipuri de flexibilitate: a) Flexibilitatea mașinilor este capacitatea de a produce noi produse sau noi combinații din produse actuale, b) Flexibilitatea de rutare este abilitatea de a utiliza mai multe rute pentru a produce același articol sau mai multe dispozitive pentru a efectua aceeași operație și c) Flexibilitatea la nivel de volum este capacitatea de a varia cantitatea comenzii fără a modifica costul, în principal prin proceduri de configurare automate.

Un sistem flexibil de fabricație este un aranjament de mașini de prelucrare automată interconectate prin sisteme de transport și sisteme de manipulare a materialelor. Procesul de fabricație necesită vizitarea unui set de stații de lucru folosind un set de căi tehnologice, furnizate de sistemul de manipulare a materialelor și validate de o unitate de inspecție automată. Piesele aprobate ies din FMS; piesele respinse por fi reprocesate, dar cel mai des sunt declarate rebuturi. Un sistem de fabricație flexibil poate fi înțeles ca o celulă de fabricație automatizată, compusă dintr-un set de stații de lucru interconectate prin unități de transfer și manipulare automate, precum și sisteme de stocare, conduse de un sistem de calcul industrial și, în general, distribuit. O celulă de fabricație automatizată poate prelucra simultan diferite tipuri de piese în diferite stații de lucru care se adaptează automat la variații neașteptate ale mixului și ale volumului comenzilor.

O implementare a unui sistem de fabricație poate fi realizată printr-una din următoarele tipuri de infrastructură: a) tip linie, în care stațiile de lucru sunt poziționate secvențial și permit doar fluxul direct de materiale de tip încărcare și descărcare de obicei în puncte extreme, separate, b) tip buclă, în care stațiile de lucru sunt poziționate în structură de tip U, și permite doar circulația materialului; încărcarea și descărcarea se fac în același punct; c) tip scară, în care stațiile de lucru sunt poziționate în perechi; materialului îi este permis să circule între și în jurul posturilor; încărcarea și descărcarea se fac ambele în același punct; d) tip deschis, în care sisteme mobile tip AGV (en.: Automated Guided Vehicle) se deplasează liber între stațiile de lucru, transferând materialul în curs de procesare; încărcarea și descărcarea se fac ambele în același punct; și e) infrastructură centrată pe robot, în care stațiile de lucru sunt poziționate în jurul unuia sau mai multor roboți, ceea ce permite orice fel de mișcare a materialului (Fig.1-1).

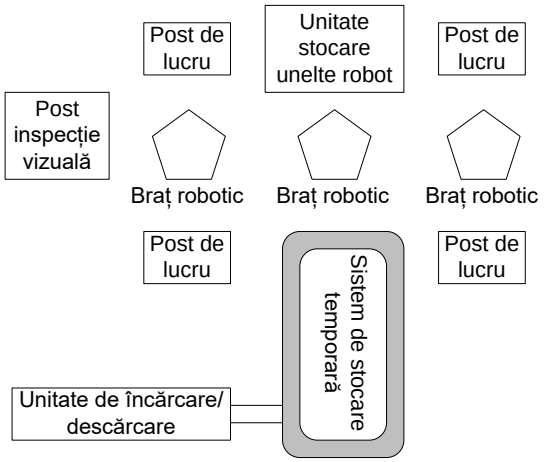


Fig. 1-1 - Infrastructură centrată pe robot

Structura tipică a unui sistem de fabricație flexibil constă în mașini cu comandă numerică (CNC) sau posturi de lucru dedicate (ex.: montarea de subansambluri), interconectate de un sistem automat de transport, sistem de transport, inspecție automată, toate acestea fiind conduse de un sistem integrat. În această structură brațe robotizate (roboți indutriali) pot servi și ca sisteme de manipulare a materialelor. Un robot industrial este o echipament programabil, de uz general, cu caracteristici antropomorfe. Acesta se aseamănă unui braț uman, are capacitatea de a lua decizii, și prin intermediul unor semnale digitale poate răspunde la stimulii senzoriali și comunica cu alte echipamente. Componentele esențiale ale unui robot sunt manipulatorul mecanic, acționările care transformă energia electrică, hidraulică sau pneumatică în putere mecanică pentru mișcare, senzorii care măsoară poziția, viteza, forța sau cuplul, unitatea de control și unitatea de putere. Principalele sarcini efectuate de roboți în fabricație sunt sudarea, manipularea, vopsirea, asamblarea și paletizarea.

În cele ce urmează sunt prezentate elementele principale ale unei celule de fabricație flexibile.

2. Interconectare posturi de lucru – resursă de tip bandă transportoare controlata de PLC

Sistemul de transport și transfer al materialelor este realizat cu elemente de conveior *Bosch-Rexroth* din familia TSplus, având structura din figura de mai jos și caracteristicile:

- *Tipul transmisiei*: banda principală transportoare "twin-track" cu lanţri duble de antrenare a port-paletelor, în circuit închis (lungime 5.2 m), și derivaţii laterale liniare de lungime 1 m pentru dirijare (intrare / ieşire) palete în posturile de lucru de pe bandă ale celor 4 roboți articulaţi vertical și orizontal;
- *Forma*: rectangulară, circuit închis, buclare prin 2 module de transfer;
- *Acţionarea elementelor de banda* principală și a derivaţiilor: independentă, prin motoare electrice;
- *Tipul derivaţiilor* de trasee: prin 8 unități de transfer port-paleta bidirecţionale, acţionate electro-pneumatic;
- *Capacitate de transfer port-palete* inter-derivaţii laterale;
- *Capacitate de conectare la alimentator de piese în circuit închis*, cu banda colectoare înclinată și banda orizontală prevăzută cu detector de prezență piesă;
- *Posturi de oprire a port-paletelor în 5 locaţii fixe*, cu opritoare acţionate electro-pneumatic;
- *1 opritor de port-palete de siguranţa* pe derivaţia de ieşire;
- *Suporţi verticali* h = 1.10 m de susţinere a benzilor transportoare;
- *Port-palete de dimensiune* 150 mm x 200 mm;
- *Viteza de antrenare* a elementelor de banda: min. 6 m / min;
- *Control* prin automat programabil
- *Identificare și trasabilitate palete* prin cititori / inscriptor de date pe pastile magnetice atașate paletelor în funcţie de tipul produsului și secvenţa de operaţii ce se execută asupra sa

Celula de fabricaţie constă în cinci posturi de lucru şi un sistem de transport aşa cum se poate vedea în Fig. 1-2. Cele cinci posturi de lucru sunt constituite în jurul unor roboţi Adept ce pot executa operaţii de asamblare/profilare şi sunt organizate după cum urmează:

- două posturi echipate cu roboţi articulaţi orizontal Cobra 600 care efectuează:
  - montarea de piese tip ax, r,l şi t.
  - inspecţie video în faza intermediară şi finală.
- două posturi echipate cu roboţi articulaţi vertical Viper 650 şi maşini unealtă (CNC) subordonate acestora putându-se efectua:
  - montare piese tip ax, l, i, t, i-modificat, şaibă.
  - manipularea pieselor brute în vederea prelucrării pe maşinile unealtă subordonate.
  - inspecţie video în faza intermediară şi finală.
- un post de alimentare echipat cu un robot articulat orizontal Cobra 800 şi două feed-ere de piese tip AnyFeeder şi care are poate efectua:
  - identificarea pieselor vrac din recipientele anyfeeder-elor.
  - poziţionarea acestora pe palete de transport.

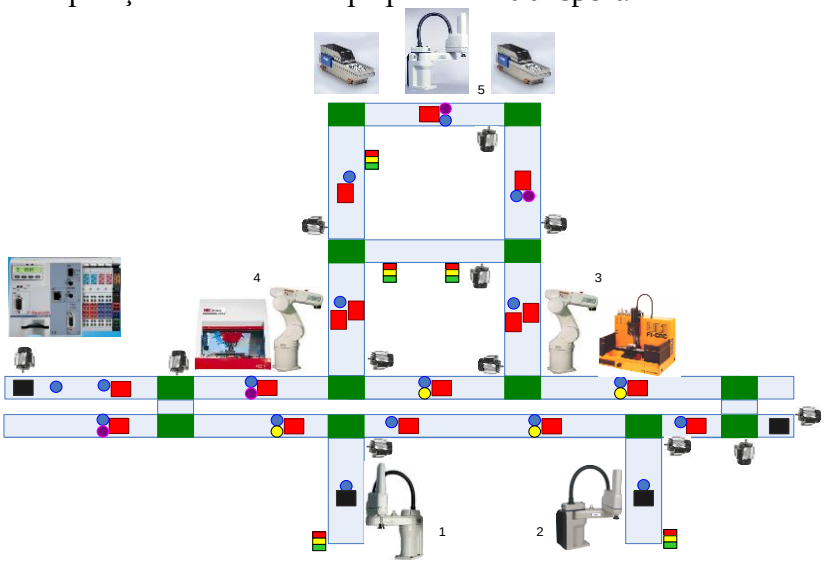


Fig. 1-2 – Structura celulei flexibile de fabricație

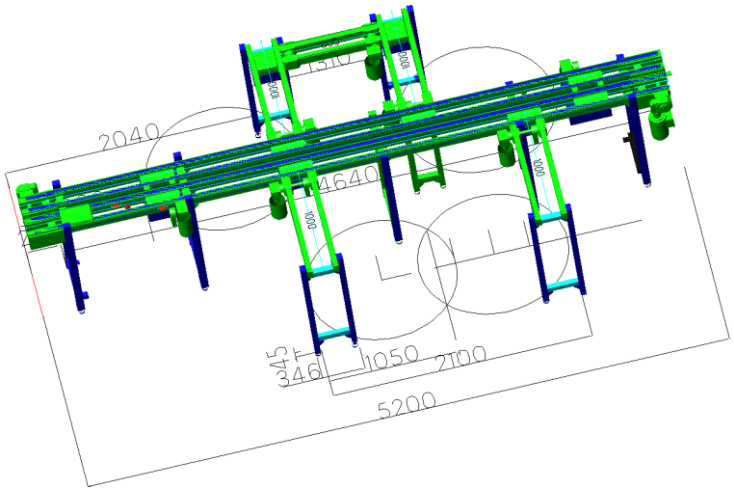


Fig. 1-3 – Linia de transfer a produselor

Configurarea acestui tip de structură de fabricație a avut în vedere asigurarea unor funcţionalități care sa permită dezvoltarea unei game extinse de scenarii de planificare și control al fabricaţiei, cu opţiuni de tipul:

- gestiune de depozite prismatice indexate, pe categorii de materiale: brute, semifabricate, componente de asamblare, montaje;
- fabricație (prelucrare / montaj / sortare / depozitare) de *n* produse pe *m* resurse M_i, $1 \le i \le m$ (maşina / roboți), fiecare produs implicând *k* operaţii din care *j* operaţii posibil de executat de către resursa M_j;
- moduri de execuție a secvenţei de producţie: manual, semiautomat, automat, în ciclu unic / cu repetiţie
- cooperare între roboți într-un subspaţiu comun spațiilor lor de dexteritate;
- schimbări de sens de mişcare în transportul materialelor;
- transfer direct de materiale între posturi robot articulat vertical;
- acces al robotului la fereastra de bandă conveior, vizualizată de camera video staţionară, pentru inspecţie online / trasabilitate și acces la produse în mişcare pe bandă;
- trasabilitate produse la nivel de resursă de procesare (transfer, manipulare) / operaţie executată / lot de produse.

Elementul principal care asigură circulaţia produsului prin sistem este port-paletul (Fig.1-4). Aceasta este folosit pentru transportul paletelor, iar pe palete sunt depozitate piesele asupra cărora se fac prelucrările. Port-paletele (160 x 160 mm) alunecă pe suprafaţa mediului conveior, aceasta metodă având următoarele avantaje:

Poziţionare. Paletele sunt poziţionate în dreptul staţiilor de lucru cu o precizie de ± 0.05mm atunci când sunt folosite unităţi de poziţionare, permiţând efectuarea operaţiilor de asamblare de mare precizie.

Montare. Paletele sunt prevăzute cu dispozitive de fixare pentru a securiza componentele pentru asamblări manuale sau automate. De asemenea, există posibilitatea ca pe paleți să se monteze dispozitive de identificare a lor şi transport al unor date/informaţii folosite pentru coordonarea secvenţelor de asamblare.

Oprire. Port-paletele pot fi oprite pe bandă indiferent de orientarea lor pe conveior. Ele pot fi de asemenea acumulate în cozi şi apoi li se poate da drumul una câte una.

Detecţia. Senzori montaţi pe spatele sau lateralul paletelor funcţionează împreună cu conveiorul pentru detecţia prezenţei paletelor. Orientarea şi rutarea paletelor poate fi controlată folosind dispozitivele de identificare Rexroth ID/10 sau ID/80.

Fig. 1-4 – Port-palet

Întreg sistemul de transport, inclusiv cititoarii/inscriptoarii de pastile magnetice sunt controlați de automatul programabil IndraControl40 fabricat de *BoschRexroth*. In Fig. 1-5 este prezentată banda transportoare cu toate intrările și ieșirile sale. Pentru conectarea acestora la automat, cât și pentru comunicația cu roboții, automatul este prevăzut cu 128 semnale digitale.

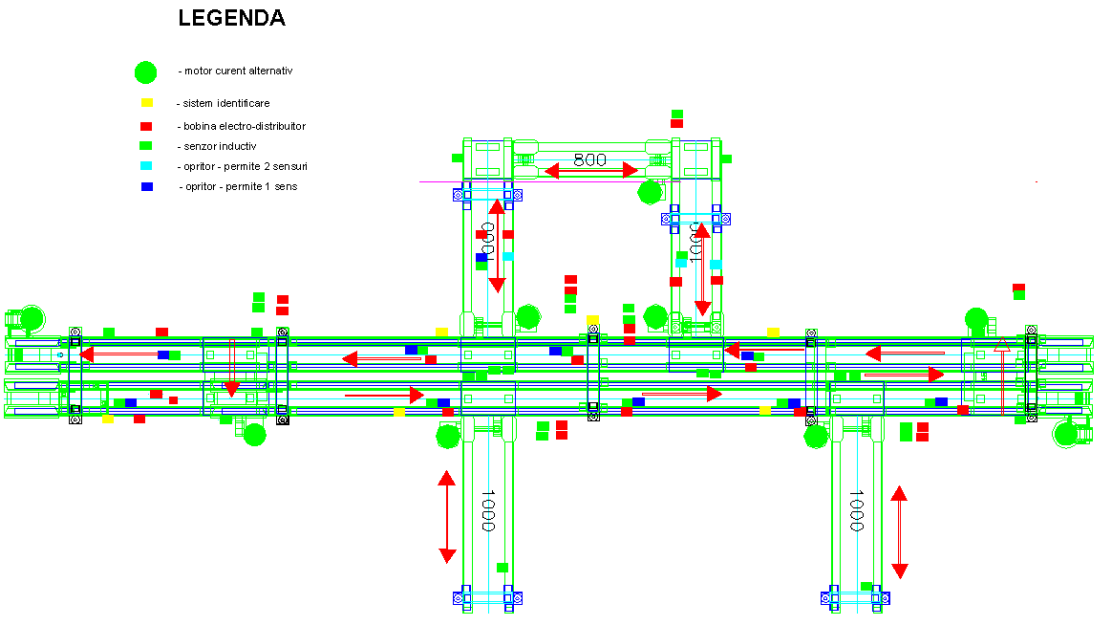


Fig. 1-5 – Semnalele digitale aferente sistemului de transport

Sistemul de identificare și stocare a datelor este folosit pentru a controla transportul și producția în sistemul de asamblare. Pe de o parte, informațiile legate de obiecte sunt folosite la controlul proceselor și al pașilor de procesare, pe de altă parte aceste date sunt folosite la coordonarea paletelor prin ramificatii. În cadrul acestui sistem identificarea port-paletelor se face prin chip-uri purtătoare de date atașate fiecărei port-palete.

Cei șase roboți de lucru sunt din familia Adept Technology:

- 2 roboți articulaţi vertical, tip Adept Viper 650;
- 3 roboți articulaţi orizontal de tip Adept Cobra s600,
- iar robotul de alimentare cu materiale este de tip Adept Pithon 3D Cartezian.

3. Robot industrial de tip SCARA – Adept Cobra s600

Sistemul robot de înaltă performanță Adept Cobra s600 (Fig. 1-6) utilizat în aplicații de asamblari mecanice, manipularea materialelor, împachetare materiale, alimentare/descărcare utilaje, înșurubare și alte operații ce necesită automatizare precisă și sigură.

Fig. 1-6 – Robotul Adept Cobra s600

Performanțele robotului Adept Cobra s600 (Fig. 1-6):

- Encodere absolute pentru calibrare ușoară;
- Encodere de rezoluție înaltă pentru urmărirea traiectoriei de înaltă precizie la viteze mici;
- Motoare de eficiență înaltă oferă o performanță ridicată cu un cuplu mai mare;
- Drivere cu armonice de inerție redusă ce oferă accelerația maximă;
- Senzorii de temperatură ce monitorizează motoarele și amplificatoarele, oferă performanță maximă și fiabilitate;
- Rată de actualizare de 8kHz pentru servo-control și urmărirea traiectoriei.

Fiabilitatea și întreținerea robotului Adept Cobra s600:

- Senzorii de temperatură integrați monitorizează temperatura în servo-motoare pentru prevenirea defecțiunilor;
- Afișarea diagnosticului permite depanarea rapidă;
- Înlocuirea facilă a sub-ansamblelor electronice.

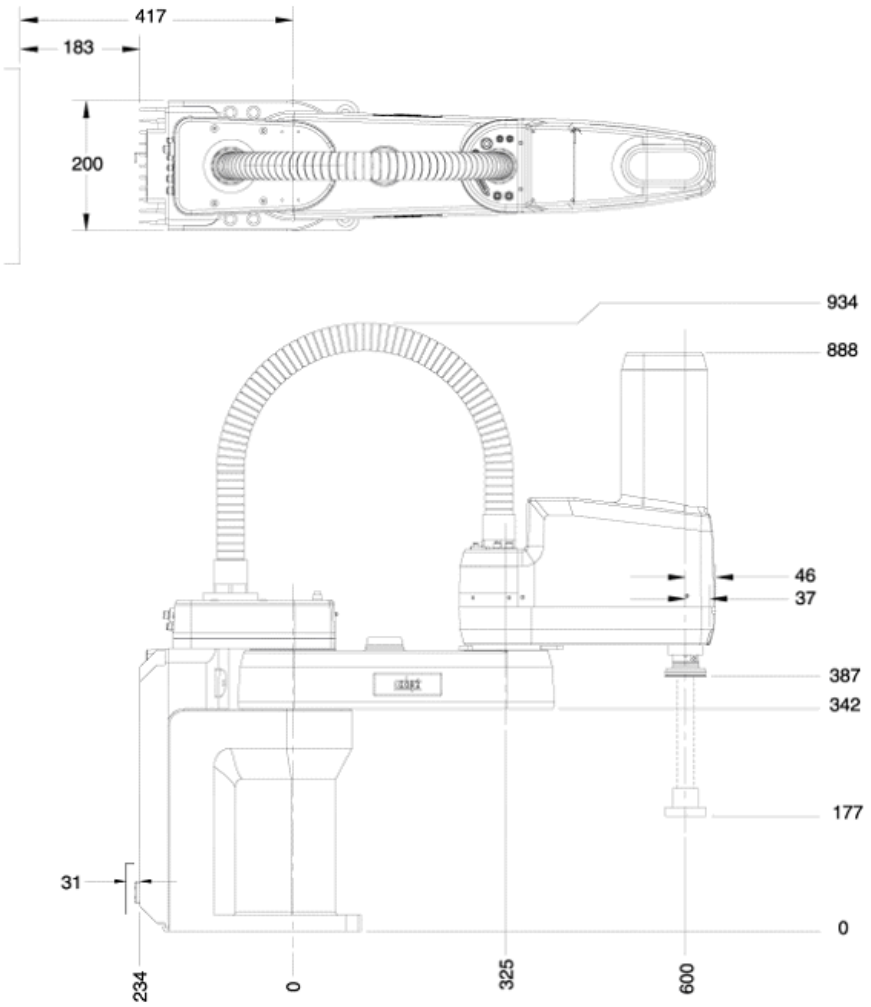


Fig. 1-7– Specificațiile dimensionale ale robotului Adept Cobra s600

Specificațiile tehnice braț mecanic robotic:

- Anvelopa de lucru (cursa maximă cu brațul in extensie): 600mm;
- Sarcina de lucru transportabilă: 2 kg (mediu), 5.5 kg (maxim);
- Moment de inerție articulația 4: 450 kg-cm^2;
- Forța de împingere a articulației 4: 35kgf;
- Ciclu Adept susținut: 0.45 s (2kg)
- Domeniile de deplasare:
    - Articulația 1: ±105°;
    - Articulația 2: ±150°;
    - Articulația 3: 210mm;
    - Articulația 4: ±360°
- Vitezele în articulații:
    - Articulația 1: 386°/s;
    - Articulația 2: 720°/s;

- Articulația 3: 1100mm/s;
- Articulația 4: 1200°/s

- Repetabiliatate:
  - XY: ±0.017mm
  - Z: ±0.003mm
  - Theta: ±0.019°

- Conexiuni utilizator interne: Electrice 24 (12 perechi torsadate), pneumatice 2x 6mm si 3x 3mm, DeviceNet;

- Canale de intrări/ieșiri digitale: 12 intări digitale, 8 ieșiri digitale, 4 ieșiri cu releu;

- Greutate: 41kg.

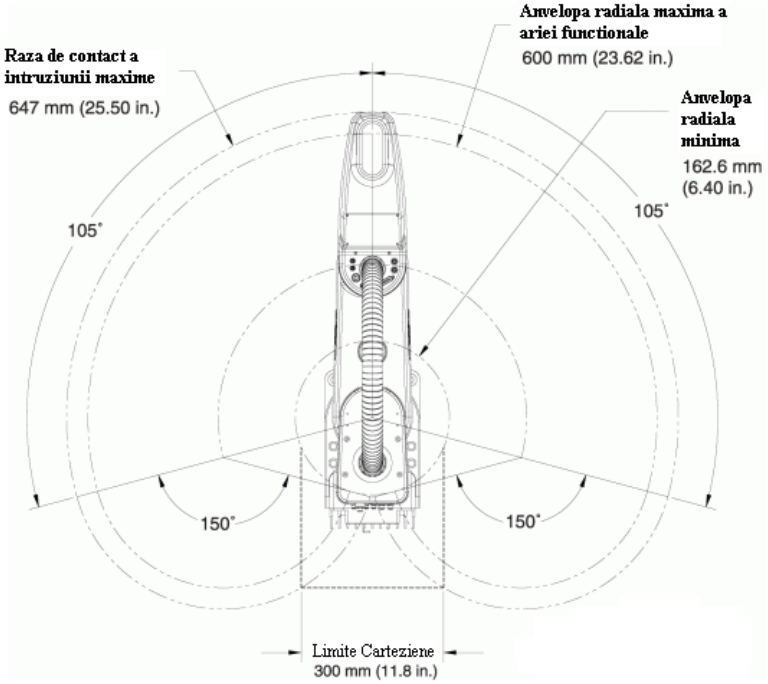


Fig. 1-8 – Anvelopa de lucru a brațului robot Adept Cobra s600

4. Robot industrial de tip articulat vertical – Adept Viper 650

Braţul robot articulat vertical de tip Adept Viper (Fig. 1-9) oferă 6 grade de libertate, 6 axe, fiind recomandat datorită complianţei, repetabilităţii și preciziei superioare de poziţionare / urmărire de traiectorii complexe. Acest braţ este ideal pentru operaţii de manipulare și interacţiune cu materiale – în scop de transfer, montaj, împachetare a materialelor, alimentare / descărcare utilaje și alte aplicații care cer o automatizare precisă și rapidă.

Sistemul Adept Viper s650 include următoarele componente:

- Braţ robot articulat vertical Adept Viper 650;
- Adept SmartController CX (cu software instalat);

- Şasiu de putere PA-4 cu servo controllere și amplificatoare;
- Panou frontal cu Oprire de Urgență (consolă operator de proces);
- Cablu robot lungime 4 m;
- Software AdeptWindows;
- Software NFS (Network File Server);
- Ethernet TCP/IP;
- Gripper electro-pneumatic tip paralel;
- Documentaţie completă utilizator.

Braţul mecanic robot are urmatoarele specificatii tehnice:

- Anvelopa de lucru (cursa maxima cu braţul in extensie: 653 mm);
- Sarcina de lucru transportabila: 5kg (max.);
- Domeniile de deplasare (6 axe, 6 articulaţii de rotaţie):
  - Axa (articulaţia) 1: ± 170°,
  - Axa (articulaţia) 2: -170°,+45°,
  - Axa (articulaţia 3): -29°, +256°
  - Axa (articulaţia) 4: ±190°,
  - Axa (articulaţia) 5: ±120°,
  - Axa (articulaţia) 6: ± 360°;
- Momente de inerţie (max.): Articulaţiile 4,5:0.295 kgm^2; articulaţia 6: 0.045 kgm^2
- Viteza compusa (max.) la vârf: 8200 mm/s;

Fig. 1-9 – Robot Adept Viper 650.

- Viteze in articulaţii:
  - Articulaţia 1: 328°/s,
  - Articulaţia 2: 300°/s,
  - Articulaţia 3: 375°/s,
  - Articulaţia 4: 375°/s,
  - Articulaţia 5: 375°/s,
  - Articulaţia 6: 600°/s;
- Repetabilitate (XYZ): ± 0.020 mm;

- Conexiuni utilizator interne: 10 electrice, 1 pneumatic;
- Frâne: Articulaţiile 2-6;
- Grad de protecţie: IP40;
- Posibilitati de montare: sol / masa sau tavan;
- Greutate: 28kg.

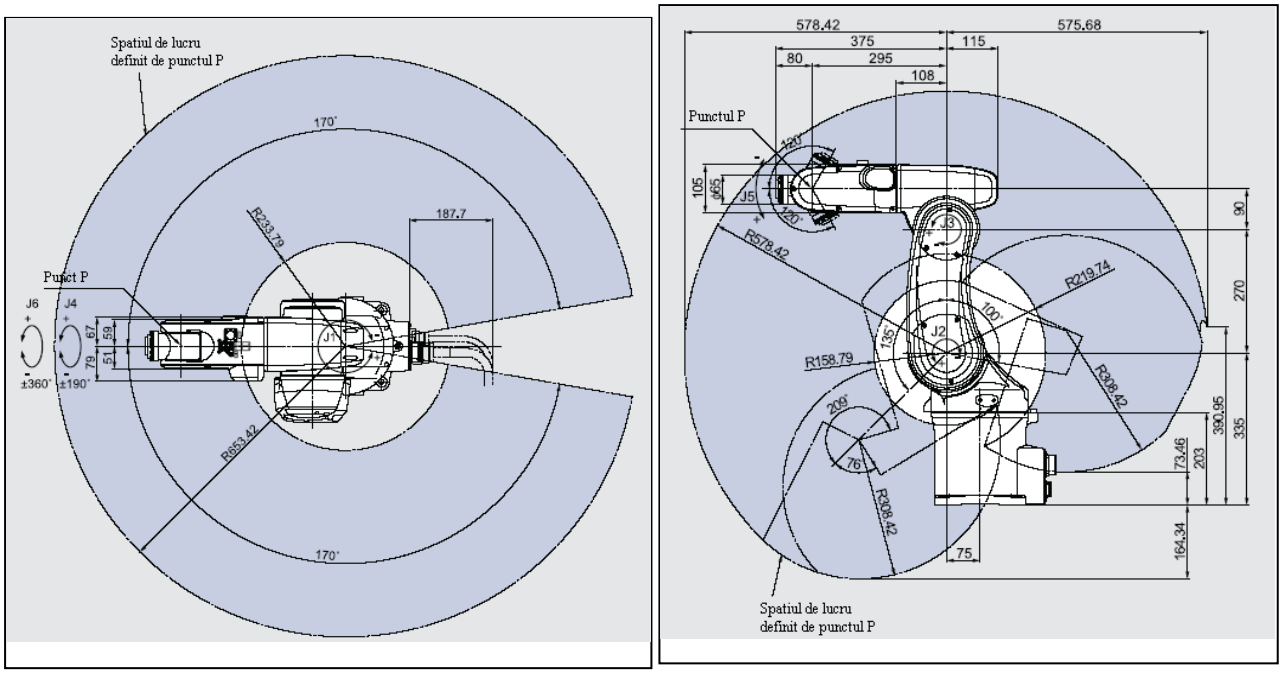


Fig. 1-10 – Dimensiuni braţ, anvelopa de lucru (vedere de sus și laterală)

5. Robot industrial de tip cartezian - Adept Python

Modulele liniare de înaltă calitate Adept Python (Fig. 1-11) se folosesc pentru aplicaţii de asamblare şi aplicaţii care cer o automatizare precisă şi rapidă.

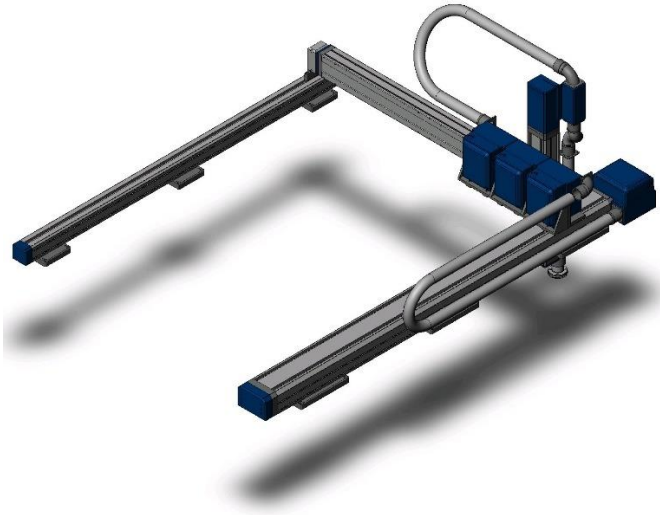


Fig. 1-11 – Robotul cartezian Adept Python

Această configuraţie foloseşte un modul tip L18 de lungime 1630mm, un modul L12 de 1055mm, un modul L08 de lungime 850mm şi un modul de rotaţie Theta tip LT1.

Componentele robotului sunt:

- Modulele Adept Python L18, L12, L08 şi LT1;
- Adept SmartController CX (cu software instalat);
- Kit de expansiune PDU3;
- Panou frontal cu Oprire de Urgenţă (consolă operator de proces);
- Cablu robot lungime 4 m;
- Software AdeptWindows;
- Ethernet TCP/IP;
- Gripper electro-pneumatic tip paralel;
- Documentaţie completă utilizator.

Spaţiul de lucru al acestui robot este un paralelipiped cu dimensiunile 1200x600x400 mm.

6. Controller robot

Toate resursele robot pot fi programate şi controlate prin intermediul unui controller extern „Adept SmartController” (Fig. 1-12), şi rulează o platformă de control distribuit al mişcării „Adept SmartServo”.

Controllerul integrează opţiuni de vedere artificială precum şi posibilitatea urmăririi evoluţiei unei eventuale benzi transportoare, aflate în raza de acţiune a robotului. Este prevăzut cu port IEEE 1394 (FireWire) prin care pot fi conectate module de intrări/ ieşiri digitale sau module suplimentare de generare a mişcării. Controllerul mai este prevăzut şi cu Fast Ethernet respectiv Device Net, precum şi cu un modul sDIO care cuprinde 32 de intrări digitale şi 32 ieşiri digitale, toate izolate galvanic, şi o interfaţă IEEE 1394.

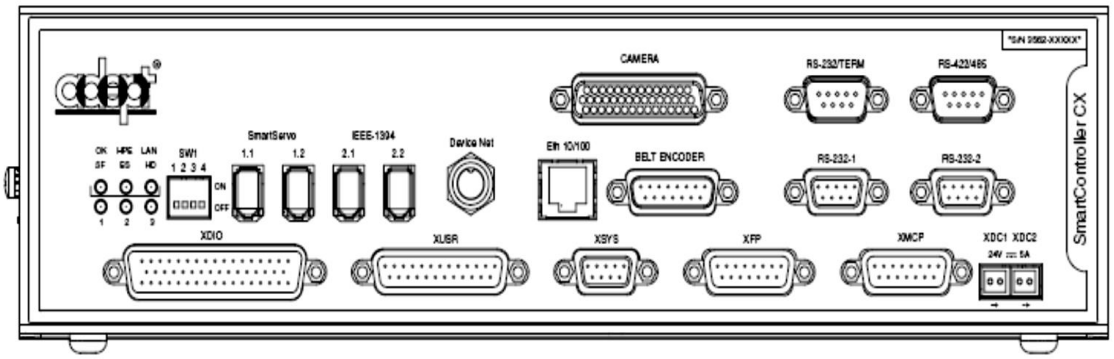


Fig. 1-12 – Adept SmartController CX

În Fig. 1-13 sunt prezentate conexiunile care trebuie realizate între calculator, controller şi robot.

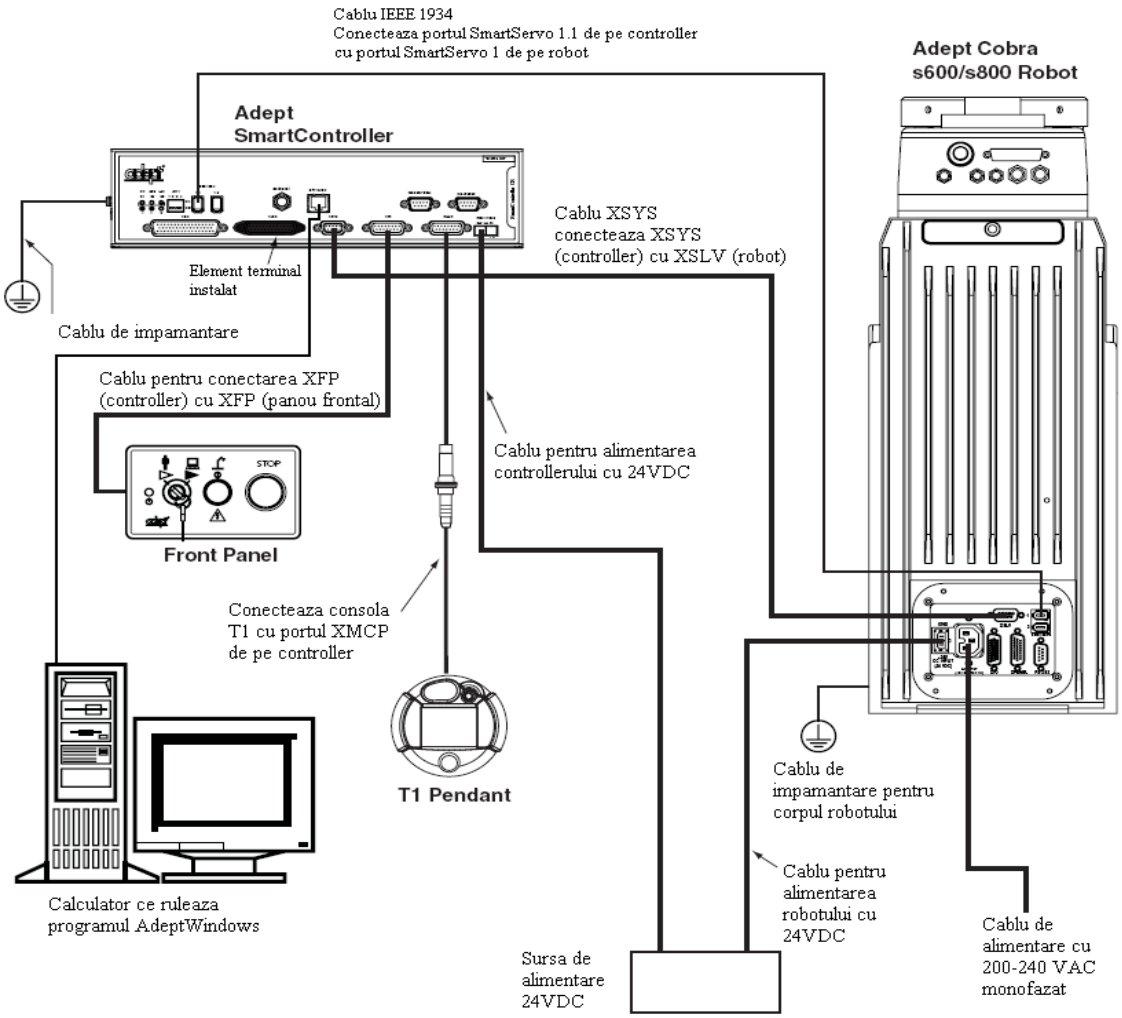


Fig. 1-13 – Conexiunile existente între calculator, controller şi robot

7. Automatul programabil

Comanda elementelor mecanice constituente ale sistemului de transport este efectuată de un automat programabil Bosch-Rexroth seria IndraControl L40. Automatul are următoarea configuraţie de module (Fig. 1-14):

- 5 module de câte 32 intrări digitale
- 1 modul de 8 intrări digitale
- 3 module de câte 32 ieşiri digitale

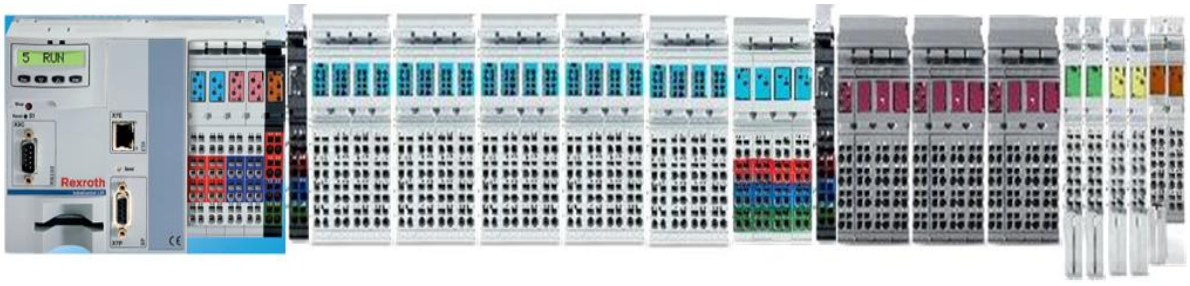


Fig. 1-14 – Configuraţia automatului IndraControl L40

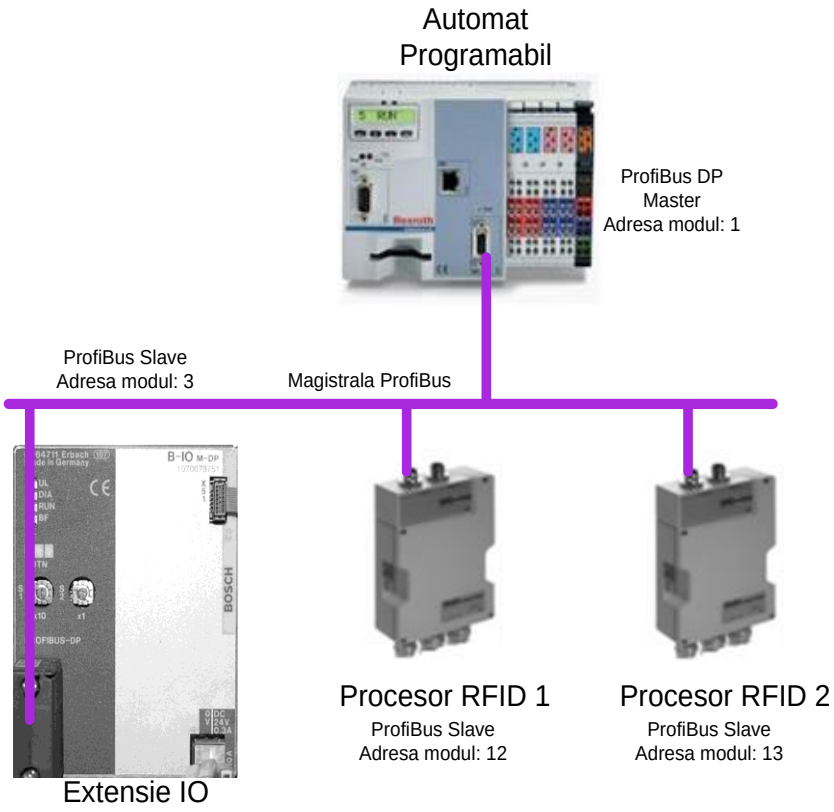


Fig. 1-15 – Extensie I/O si cititoare pe Profibus

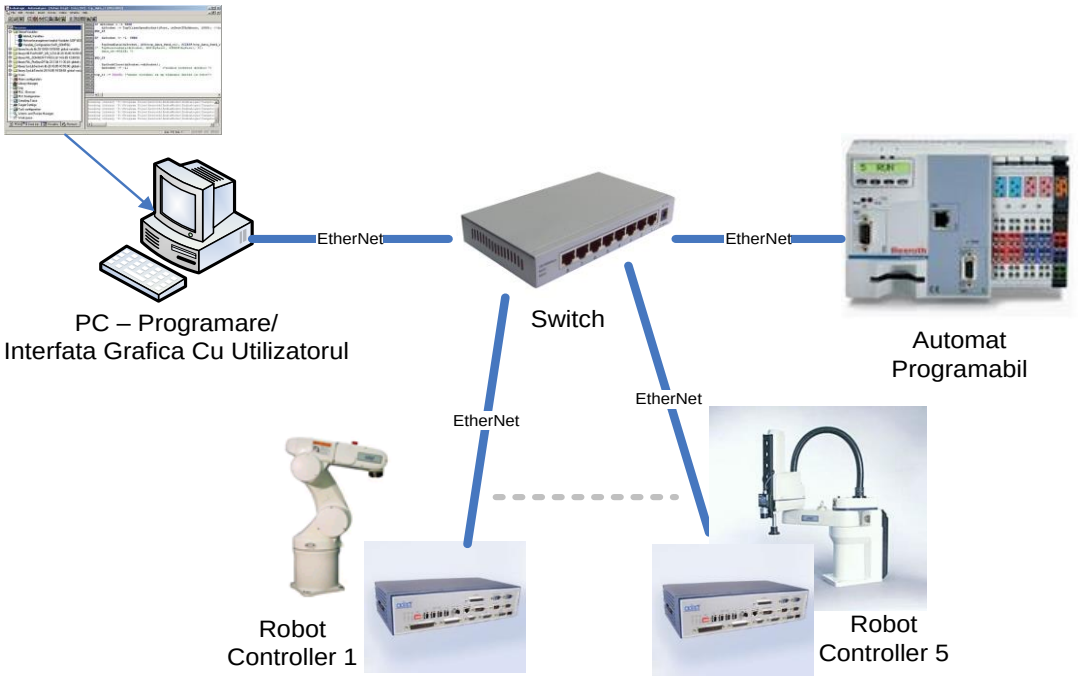


Fig. 1-16 – Conexiunea EtherNet PC-Automat Programabil

8. Resursă de prelucare mecanică de tip EMCO Concept Mill 105

Fig. 1-17 – Maşina de frezat cu comandă numerică EMCO Concept Mill 105

Maşina CNC EMCO concept Mill 105 din figura 1-17 are datele tehnice prezentate în tabelul 1-1.

Tabel 1-1. – Date tehnice maşină CNC EMCO concept Mill 105

Tip	Maşină de frezat cu 4 axe: 3 carteziene şi una rotativă, cu magazie de scule cu schimbător automat şi elemente de automatizare şi interfaţare cu robot industrial
Materiale prelucrate	Metale, materiale plastice
Cursă efectivă axele X/Y/Z	200/150/250mm
Motor acţionare freză	Asincron de curent alternativ, 1.1 kW, 150-5000 rpm
Motoare acţionare axe carteziene	Pas cu pas
Viteza de traversare rapidă axele X/Y/Z	5000 mm/min
Acurateţe de poziţionare axele X/Y/Z	3/3/4 mm
Forţa de înaintare maximă axele X/Y/Z	2000/2000/2400N
Magazie de scule	10 scule, prindere SK30
Controller	Siemens SINUMERIK 840D (emulat)
Software	Pachet CAD/CAM: Emco CamConcept Mill
Elemente de automatizare	Uşă automată, cu acţionare pneumatică
	Menghină pneumatică
	Interfaţă cu robot industrial (semnale digitale I/O şi comunicaţie DeviceNet)
Sistem de ungere	Central, funcţionare automată

Alimentare	220V, 50/60Hz, 1.4kVA, siguranţă 16A
Elemente de siguranţă	Limitatoare de cursă pe cele 3 axe
	Buton de avarie
	Conformitate cu standardele: CE EN292, EN60204, EEC machine guiding rules
Dimensiuni (LxDxH)	1135 x 1100 x 1100 mm
Greutate:	400 kg

9. Sistem de inspectie si ghidare vizuala AdeptSight

In vederea asigurării a) functionalității de ghidare vizuală a braţului manipulator robot și b) integrării sistemului robot in producţie/servicii, există soluţia AdeptSight, bazată pe hardware PC și software video în scopul ghidării robotului Adept Viper s650 (respectiv Adept Cobra s600) și inspecţiei vizuale. Aceasta soluţie asigură compatibilitate cu controllerul SmartController CX și limbajul V+. Licenţele furnizate permit instalarea programelor de gestiune vizuala de scene fixe, recunoastere după model, măsurători, instrumente VA, localizare și utilitarelor de calibrare electromecanică și cameră-robot.

Soluția de vedere artificiala AdeptSight include:

- Software AdeptSight (extensie V+);
- Camere industriale FireWire și USB3.0, compatibile cu sistemul de ghidare vizuală, scanare progresivă, obiective cu lentile C-mount;

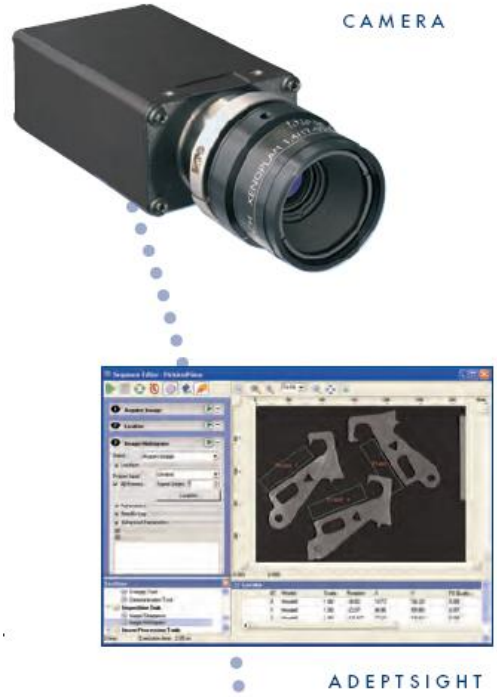


Fig. 1-18 – Sistem de ghidare și inspecție vizuală AdeptSight

Conectivitatea pachetului AdeptSight este realizată prin:

- SmartController;

- PC cu cerințe minime: procesor Intel Pentium 4 la minim 2 GHz ; spaţiu pe disk/memorie 500 MB/256 MB; monitor XGA cu rezoluţie 1024x728; I/O: port IEEE 1394 (FireWire) sau placa de reţea Ethernet (10/100/1000), port USB;

- Cerinţe software ale AdeptSight: .NET Framework 2.0; Windows® XP (SP2) sau Windows 2000 (SP4); AdeptDesktop 2.0.

10. Exemplu de producție

In funcție de operațiile care se doresc să se efectueze în posturile de lucru de către resursele robot, acestea trebuie prevăzute cu depozite care să conțină piesele necesare. În Fig. 1-19 este prezentată distribuția pieselor în posturile de lucru cu observația că în fiecare lăcaș se află piese pe 4 niveluri.

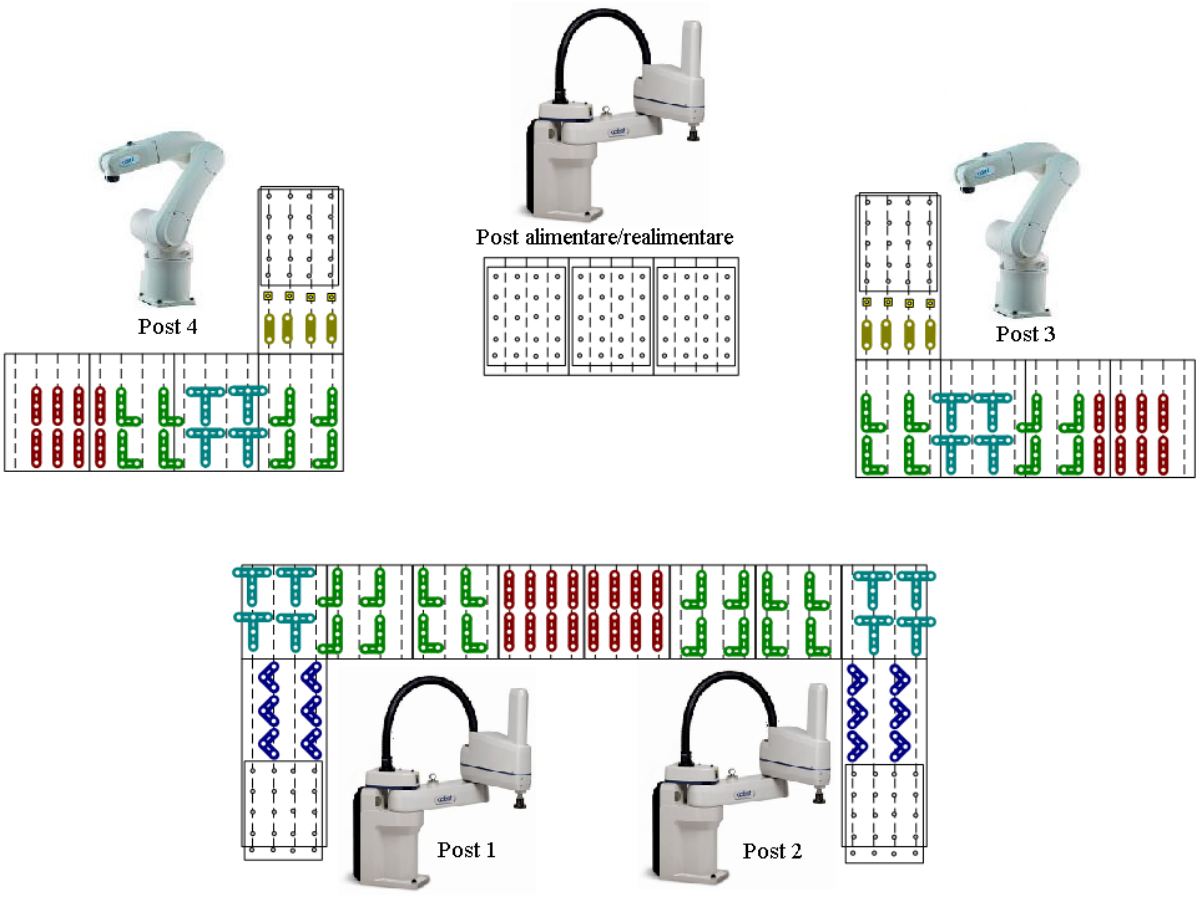


Fig. 1-19 – Distribuția pieselor în posturile de lucru

Pentru validarea sistemului avem urmatoarele tipuri de produse (Fig. 1-20):

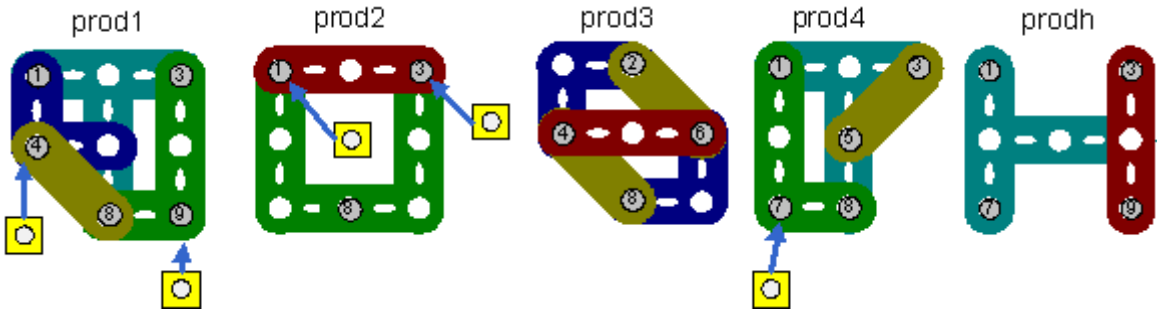


Fig. 1-20 – Tipurile de produse propuse pentru testarea sistemului de fabricație

11. Norme de protecția muncii și securitate

În elaborarea și controlul sistemelor robotizate trebuie avut grijă la următoarele componente: anvelopa robotului cu tot cu efector terminal și piesa manipulată, cabluri electrice și pneumatice (Fig. 1-21).

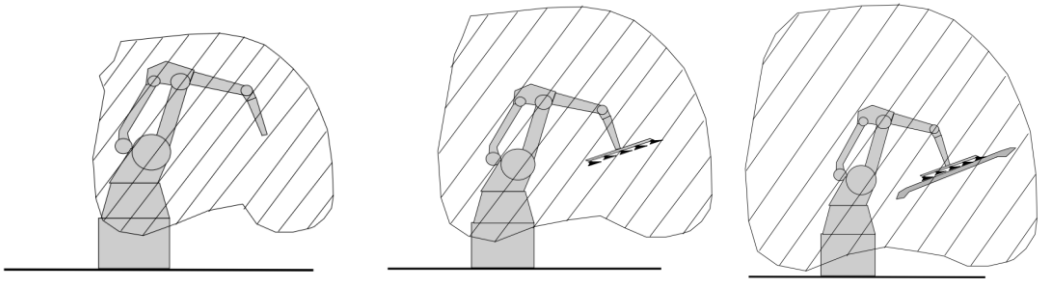


Fig. 1-21 – Spatiul de lucru a) fara efector terminal, b) cu efector terminal, c) cu obiect in

In figura 1-21 sunt evidențiate spațiile de lucru ale robotului, acestea trebuind restricționate. În cazul în care constrângerile fizice (dimensiunea celulei) nu permit atunci vor fi impuse restricții software asupra spațiului de lucru, rezultând astfel un spațiu de lucru restricționat (Fig. 1-22). Aceste spații sunt separate cu bariere fizice și optice de operatorul uman (Fig. 1-23). În eventualitatea în care operatorul intra în spațiul de lucru acesta trebuie detectat și securitatea activată rezultând oprirea de urgență a robotului.

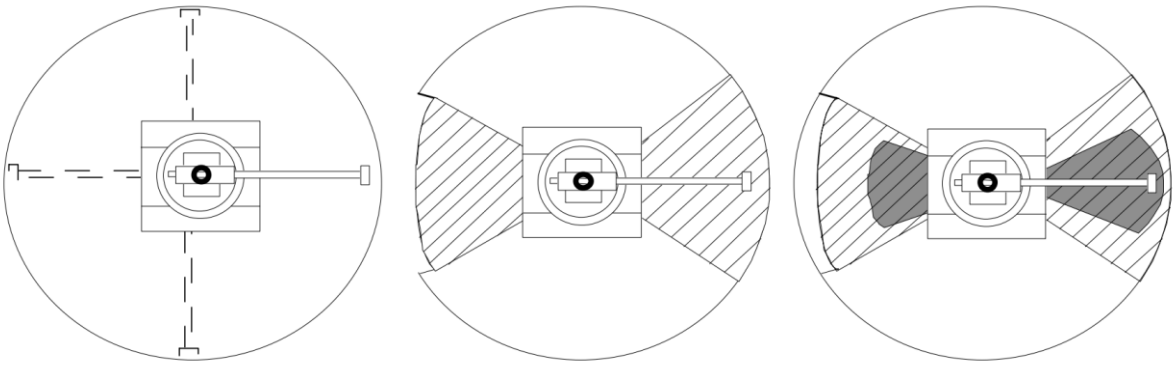


Fig. 1-22 – Spațiul teoretic de lucru/ Spațiul restricționat/ Spațiul de lucru (în mod automat)

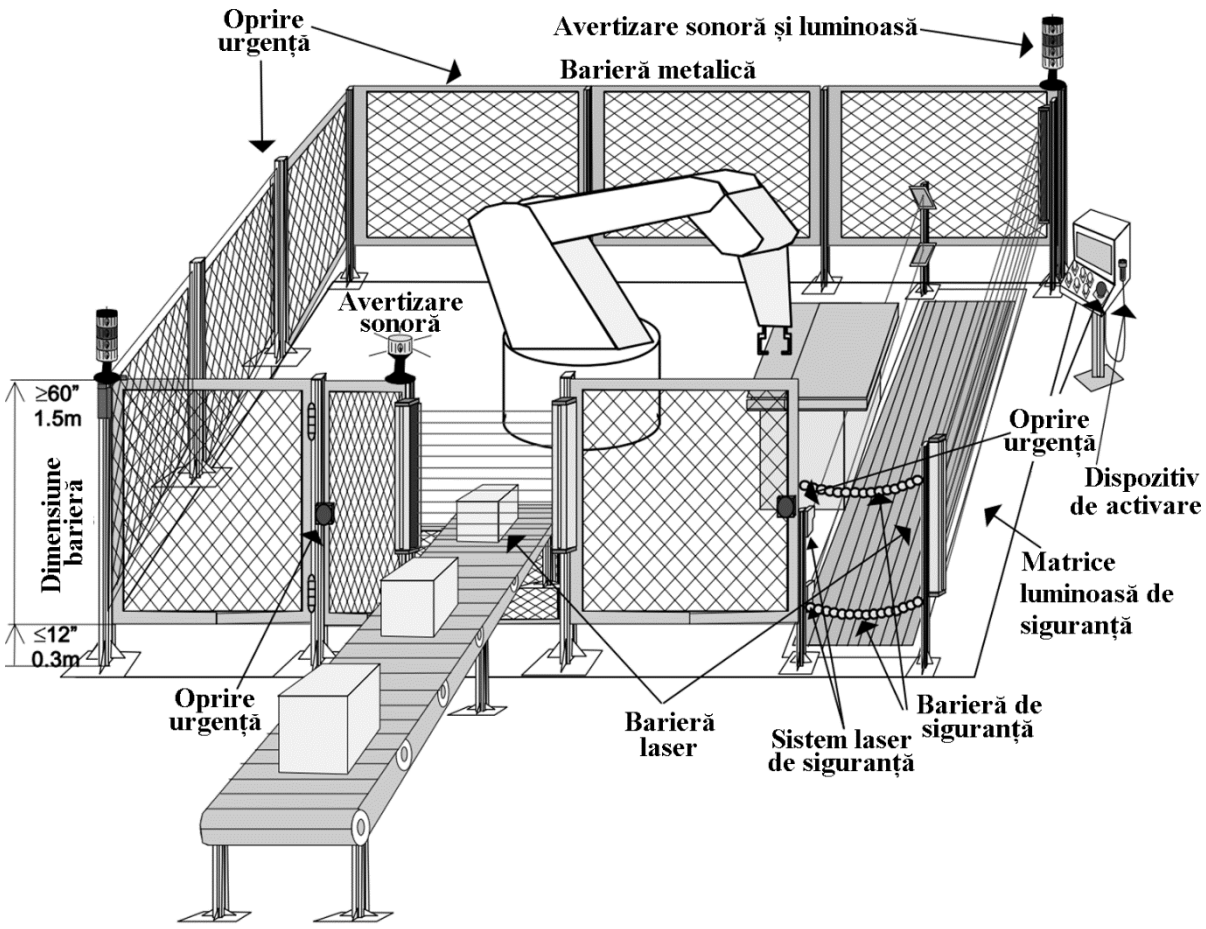


Fig. 1-23 – Structura tipică de post de lucru cu robot industrial

efectorul terminal

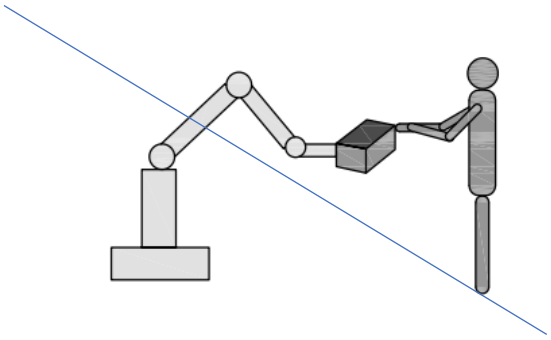


Fig. 1-24 – Aceasta modalitate de lucru este interzisă în modul de lucru automat

12. Intrebări:

1. Ce îi lipsește robotului pentru a putea face operații?
2. Ce fel de acționări sunt în sistemul de fabricație?
3. Care este diferența Profibus-Profinet?

Capitol 2: Descrierea functionalităților RobotStudio

Cuprins

1. Descriere aplicație RobotStudio 25
2. Pornirea aplicației RobotStudio 27
3. Perspective de lucru 28
   a. Meniul Home 29
   b. Adăugare obiecte în spațiul de lucru (en.: Modeling) 32
   c. Meniul de simulare (en.: Simulation) 35
   d. Meniul Controller 39
   e. Meniul RAPID – Editorul de programe 42
4. Procedura adăugare unealtă 43
5. Export proiect 44
6. Sistem de coordonate 44
7. Exerciții 45
8. Intrebari 45

1. Descriere aplicație RobotStudio

Aplicația RobotStudio (https://new.abb.com/products/robotics/robotstudio) este folosită pentru proiectarea, realizarea digitală, programarea și validarea sistemelor mono și multi-robot cu și fără accesul direct la echipamentele fizice. RobotStudio oferă instrumentele pentru a crește profitabilitatea unui sistem robot, permițând efectuarea de sarcini precum pregătirea inginerilor, programare și optimizare fără a perturba producția. Câteva caracteristici ale programării offline a echipamentelor de tip robot industrial sunt enumerate mai jos:

- reducerea riscurilor aferente operării pe instalația fizică
- demararea rapidă a proiectelor, întâi in mediul digital
- timpi de schimbare a traiectoriei micșorați
- productivitate crescută

În cadrul unei celule robotizate există un set de componente hardware care sunt utilizate pentru a lucra simultan în realizarea de sarcini diferite. Mai jos sunt prezentate pe scurt aceste elemente, ele fiind incluse în aplicația RobotStudio atât la nivel de componentă grafică, cât și (în anumite cazuri) la nivel de componentă inteligentă având posibilitatea de a condiționa comportamentul prin instrucțiuni specifice:

- braț manipulator – robotul fizic;
- controller – roboții ABB sunt conduși de un controller IRC5 compus dintr-un modul de control și un modul de acționare pentru fiecare manipulator robot din sistem;
- dispozitiv de mișcare și programare robot (FlexPendant), conectat la modulul de comandă. Programarea pe FlexPendant este denumită „programare online”;
- unealtă – dispozitiv montat de obicei pe robotul manipulator pentru a-i permite să îndeplinească sarcini specifice, cum ar fi prinderea, tăierea sau sudarea. Instrumentul poate fi, de asemenea, staționar (nu este montat pe robot; de asemenea, numit „instrument extern”).

RobotStudio este o aplicație software creată de firma ABB, utilizată în modelarea, programarea offline și simularea roboților industriali. Permite utilizatorului să exerseze, să programeze și să optimizeze fără a perturba producția în incinta în care se află celula robot. Mai mult, se bazează pe ABB Virtual Controller, o copie exactă a sistemului de operare ce rulează pe roboții fizici, din producție. Acest lucru permite simulări realiste utilizând programe robot reale și fișiere de configurare identice cu cele folosite pe linia de producție.

Programul RobotStudio (Fig. 2-1) conține editorul RAPID, limbaj în care sunt sunt programați roboții de la ABB. Editorul RAPID furnizează un mediu de dezvoltare similar cu Microsoft Visual Studio și oferă sugestii și suport în timp ce utilizatorul scrie programul robot. Argumentele implicite pot fi completate în mod automat. Aplicația mai conține și unealta Virtual FlexPendant, cu ajutorul căreia utilizatorul poate controla și monitoriza controlorul virtual în același mod precum un controller real.

Componenta principală a acestei interfețe este reprezentarea grafică a robotului programat și a mediului de lucru. Folosind mouse-ul utilizatorul poate indica locații sau obiecte pe ecranul grafic. Proiectarea interfeței utilizator stabilește exact cum utilizatorul interacționează cu ecranul pentru a specifica un program robot. Același dispozitiv de indicare poate specifica într-un meniu modurile sau funcții de interes.

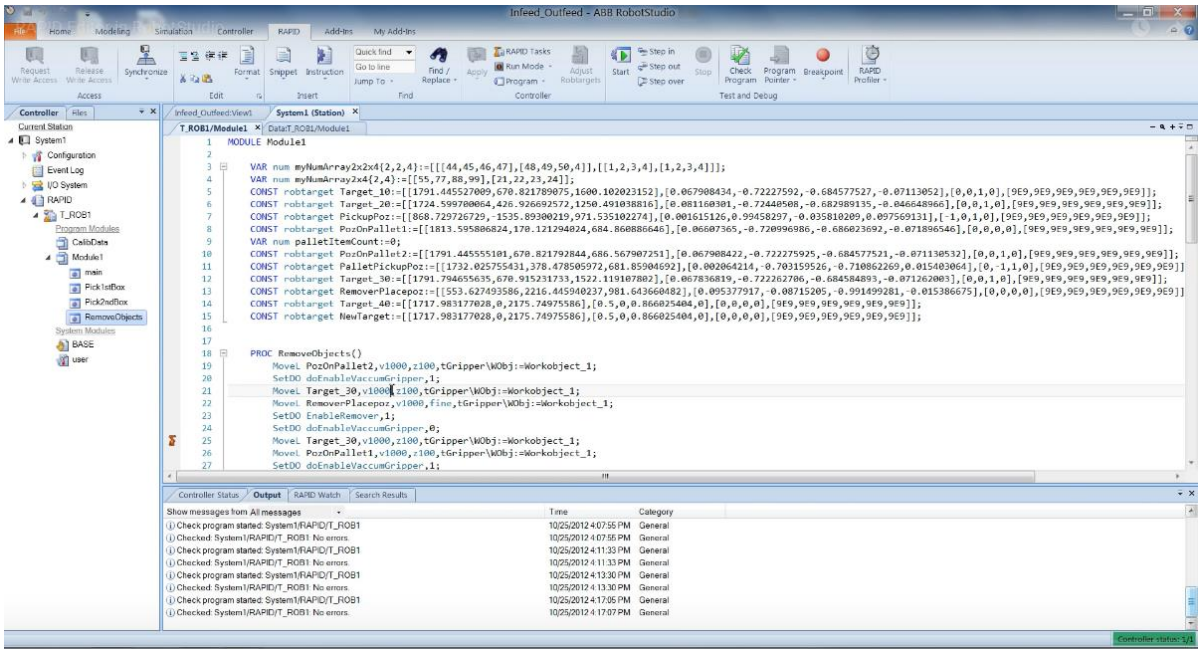


Fig. 2-1. Programul RobotStudio

Un program robot se mai numește traiectorie pentru ca în urma instrucțiunilor program punctul condus descrie o traiectorie. Programarea unui echipament de tip robot industrial

presupune două etape, nu neapărat succesive: 1) învățarea punctelor suport și 2) secvențierea instrucțiunilor de mișcare și sincronizare. Pentru realizarea traiectoriei robot poate fi folosit robotul fizic sau o un simulator. Dacă se utilizează robotul în timpul programării, atunci se consideră programare online. Dacă nu se utilizează, atunci se programează offline. La rândul lor, aceste criterii se împart în mai multe metode (Fig. 2-2):

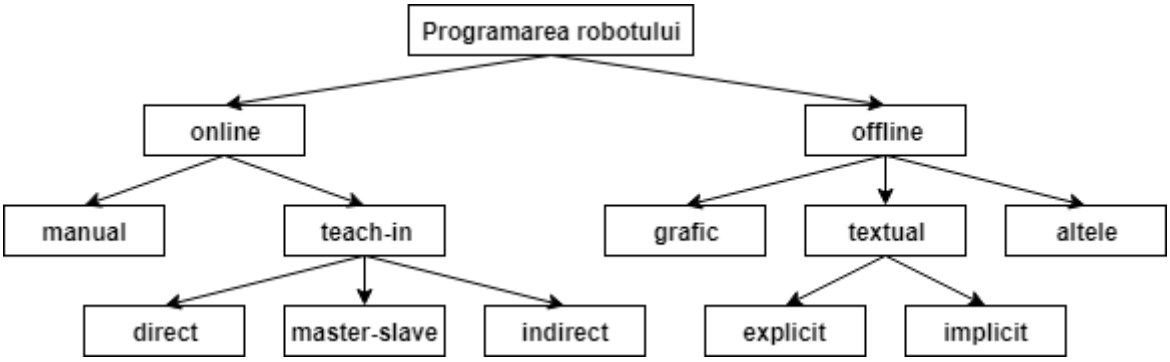


Fig. 2-2. Metode de programare a roboților

Pentru programarea unei aplicații se pot folosi și combinații de mai multe metode de programare. Se obișnuiește să se folosească programarea teach-in pentru corecția pozițiilor planificate într-un program creat prin metoda offline.

Materialul de față are ca obiectiv deprinderea cu metoda de programare offline și configurarea mediului de lucru virtual în vederea simulării diferitelor tipuri de aplicații industriale.

2. Pornirea aplicației RobotStudio

În urma lansării aplicației utilizatorul este pus să aleagă unul din două tipuri de proiecte (Fig. 2-3): 1) proiect gol fără controller și 2) proiect cu controller și braț robotic. Pentru realizarea unei simulări este necesar un controller virtual atașat roboților existenți, de aceea și în cazul 1 controllerul și roboții vor fi atașați ulterior, dar existând posibilitatea de adăugare a mai multor astfel de echipamente cu configurații particulare.

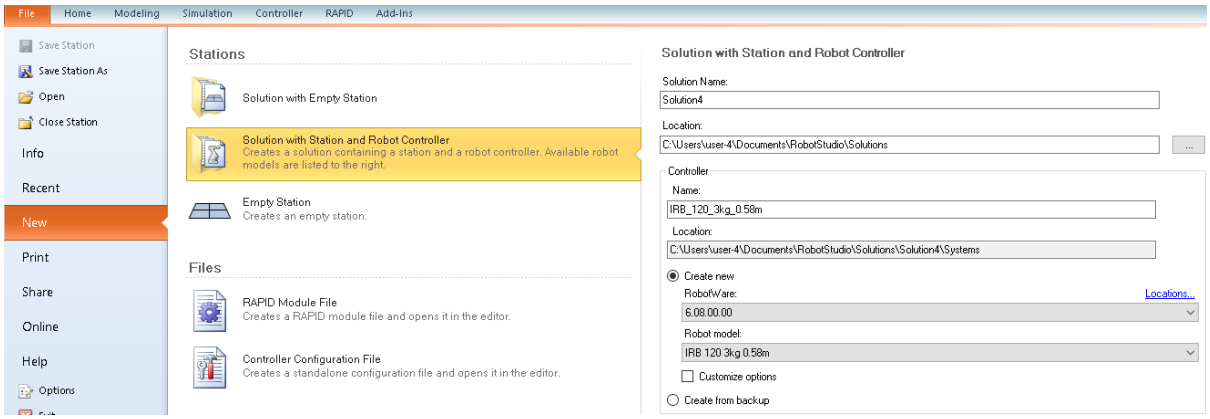


Fig. 2-3 – Tipurile de proiecte RobotStudio

Pentru început vom considera un proiect de tip robot unic (en.: single robot). În acest caz va trebui selectată versiunea sistemului de operare robot (se recomandă folosirea sistemului RobotWare 6.08) și tipul de robot. Este recomandat ca proiectul să primească un nume sugestiv

(ex.: Aplicatie_sudura_V2) întrucât salvarea proiectului nu se poate face în versiuni, iar opțiunile de configurare a controllerului să fie lăsate pentru după crearea proiectului. Din meniul de brațe robotice se observă tipul brațului (ex.: SCARA, articulat vertical, ș.a.) precum și alonja, respectiv masa maximă ce poate fi manipulată. Pentru ca aplicația simulată să fie cât mai aproape de realitate se recomandă folosirea unui braț robotic cât mai aproape de cotele fizice ale celulei.

În urma apăsării butonului Create proiectul este generat și echipamentul ales împreună cu mediul de lucru pot fi inspectate în mod 3D folosind mouse-ul împreună cu combinația de taste CTRL și SHIFT astfel:

- Buton stânga mouse – selecție obiect prim plan;
- Buton dreapta – selecție proprietăți obiect, dacă este cazul sau configurare proprietăți perspectivă;
- Buton stânga cu CTRL apăsat – translație perspectivă vedere
- Buton stânga cu CTRL și SHIFT apăsate – rotație perspectivă vedere

3. Perspective de lucru

Fereastra principală din aplicație este fereastra spațiului de lucru în care se află o prezentare vizuală a proiectului. Această fereastră dispune de o serie de butoane care sunt folosite de utilizator pentru a selecta mai ușor anumite elemente din spațiul de lucru (Fig. 2-4).

Fig. 2-4 – Butoanele disponibile în spațiul de lucru

Cele mai folosite butoane de selecție (Fig. 2-4) sunt:

- Butonul *Part Selection*  se folosește pentru a selecta un singur element din spațiul de lucru;
- Butonul *Snap Object*  se folosește pentru a selecta punctele de margine ale unei componente sau obiect;
- Butonul *Snap Surface*  se folosește pentru a selecta o suprafață;
- Butonul *Snap Center*  este folosit pentru a selecta centrul unui plan din spațiu;
- Butonul *Snap Mid*  se folosește pentru a selecta mijlocul unei linii.

Exemplu de utilizare: în multe aplicații este nevoie să se selecteze mijlocul unei suprafețe (ex.: mijlocul capacului unui cilindru); în acest caz se vor selecta succesiv Snap Surface și Snap Center și apoi la glisarea mouseului peste suprafața respectivă se va observa marcarea cu o sferă a punctului respectiv (Fig. 2-5).

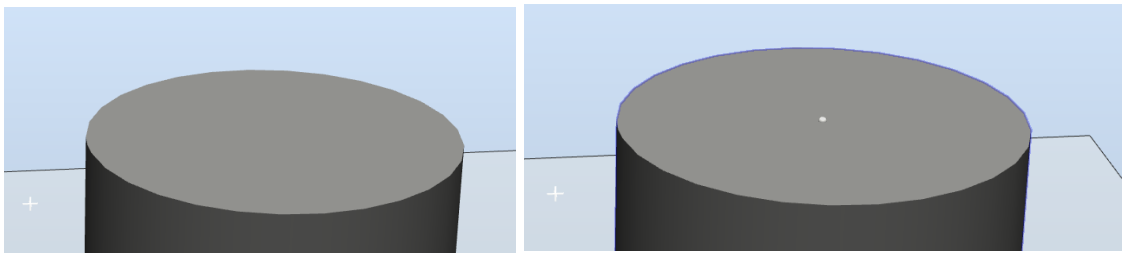


Fig. 2-5 – Selecție centru cilindru

Pentru realizarea proiectului și pentru simulare sunt disponibile cinci meniuri:

- Home;
- Modeling;
- Simulation;
- Controller;
- RAPID.

a. Meniul Home

Home (Fig. 2-6) este meniul de bază care este folosit pentru a crea un sistem, pentru programarea traiectoriilor și plasarea obiectelor. Acesta este folosit pentru a adaugă unul sau mai mulți roboți, pentru a învăța puncte și pentru a configura instrucțiuni, pentru a mișca obiectele din spațiul de lucru, pentru a crea sisteme multi-robot în cazul selecției opțiunii (Solution with empty controller) și pentru a dimensiona elementele grafice (puncte, sisteme de coordinate, etc).

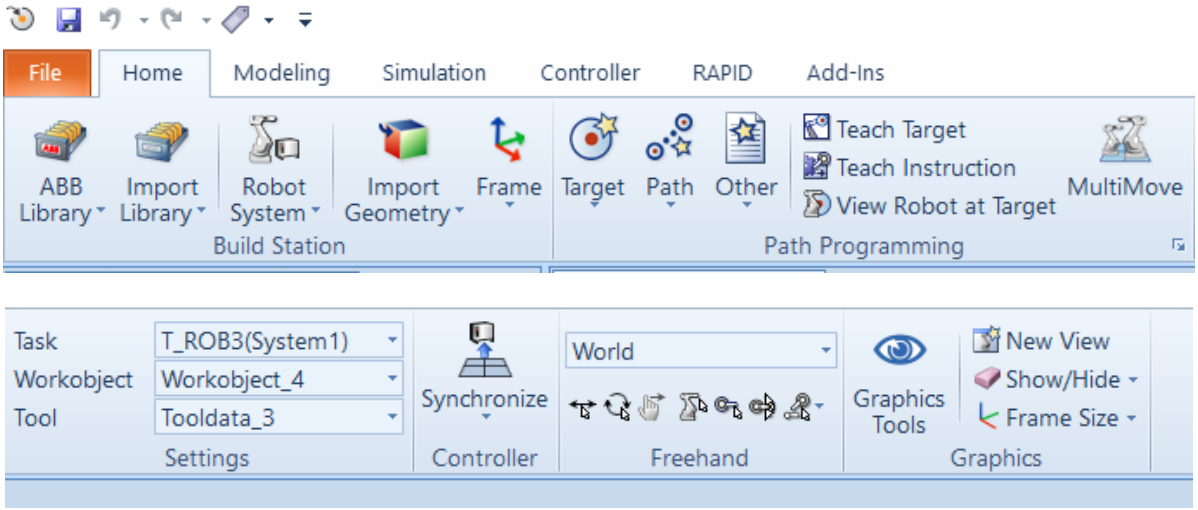


Fig. 2-6 – Componență meniu Home

Tot din această perspectivă (meniul Layout/Paths&Targets/Tags – Fig. 2-8, Fig. 2-9) se pot adăuga diferite tipuri de comportamente pentru obiectele selectate, respectiv se poate face o programare grafică a robotului, aici incluzând definirea punctelor, a sistemelor de coordonate și a traiectoriilor posibile.

Pentru configurarea individuală a elementelor grafice se selectează perspectiva (fereastra) Layout și apoi se face click dreapta pe componenta dorită, aceasta putând fi de 3 tipuri: robot, component pasivă sau componentă inteligentă. Astfel, se pot configura atribute generice precum poziția, locația, culoarea sau vizibilitatea (în cazul componentelor pasive) sau atribute particulare precum mișcarea unor axe individuale, mișcarea liniară sau anvelopa de lucru în cazul unui obiect de tip robot industrial. La poziționare precum și la mișcarea echipamentelor trebuie avut grijă la sistemul de coordonate în care se face operația (Fig. 2-7).

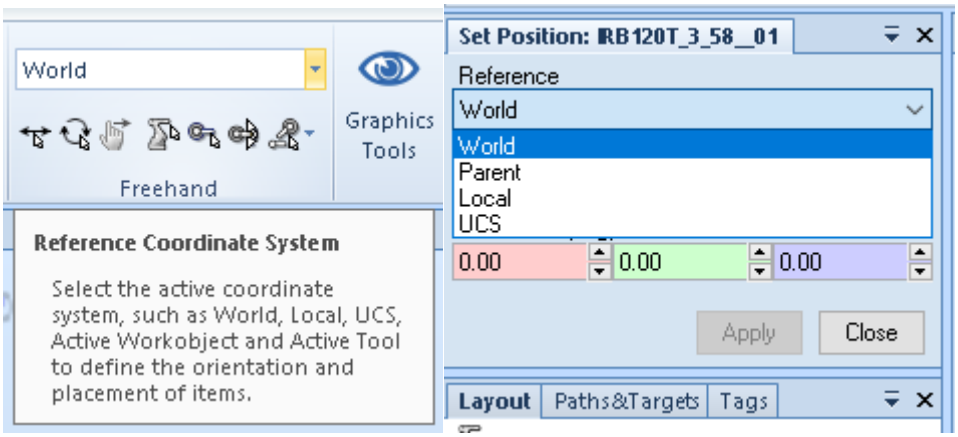


Fig. 2-7 – Tipuri de sisteme de coordonate

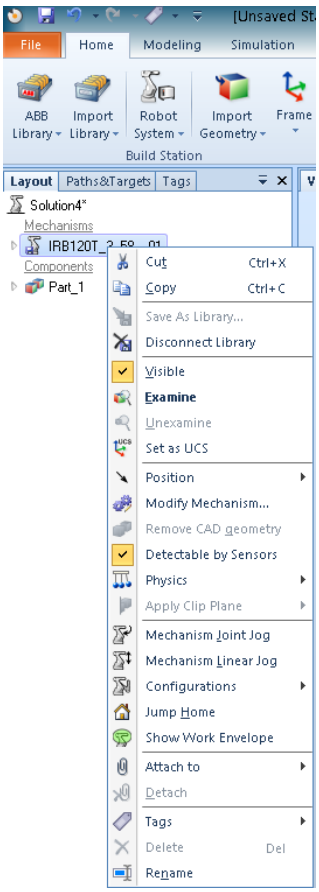


Fig. 2-8 – Fereastră Layout

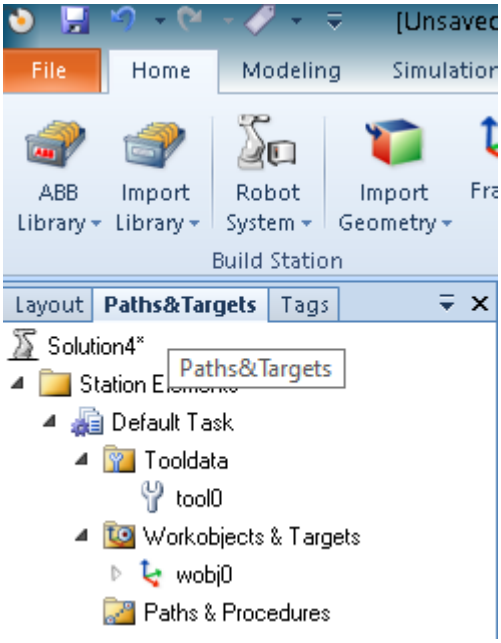


Fig. 2-9 – Fereastră Paths&Targets

Pentru adăugarea unui punct există două posibilități: 1) definirea punctului folosind butonul Target (Fig. 2-10), respectiv 2) mișcarea robotului în punctul dorit și apoi învățarea folosind comanda Teach Target (Fig. 2-11). Punctele învățate sau definite se regăsesc în secțiunea Paths&Targets -> robotul cu sistemul de coordonate selectat -> Workobjects & Targets -> Sistem de coordonate (en.: Workobject) -> punct de interes denumit și țintă (Target) (Fig. 2-12, chenarul albastru).

Realizarea celei de-a doua părți a programului robot, secvența de instrucțiuni de mișcare, se poate face în mod grafic prin crearea unei traiectorii în secțiunea Paths și aducerea punctelor definite anterior în această secțiune (Drag&Drop) (Fig. 2-12, chenarul roșu). La realizarea traiectoriei trebuie selectate tipurile corecte de instrucțiuni de mișcare (Fig 2-13).

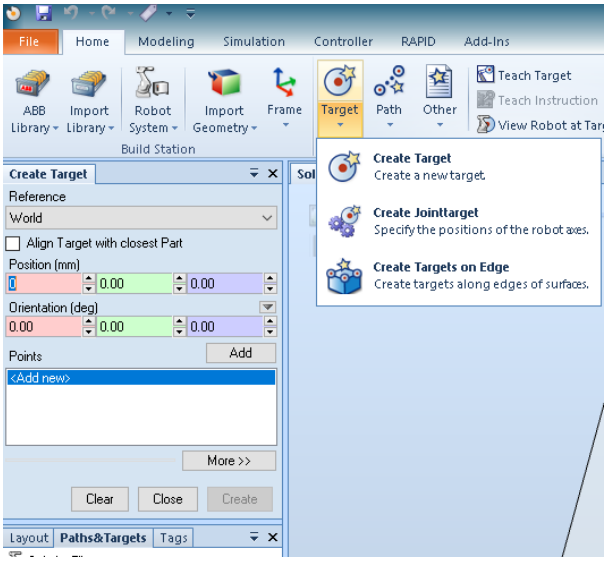


Fig. 2-10 – Definirea unui punct în spațiul de lucru (Atenție!!! – punctul se poate să nu fie atins de robot dacă, de exemplu, este în afară anvelopei de lucru)

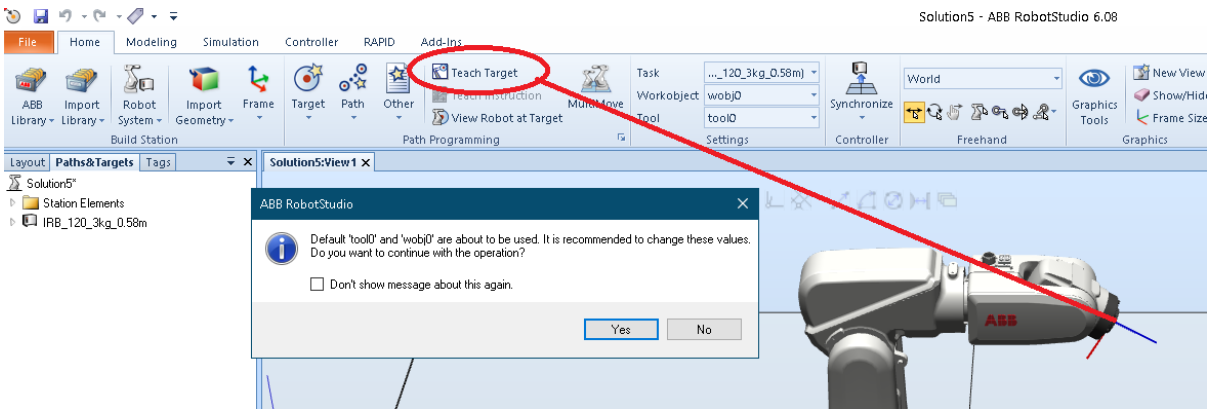


Fig. 2-11 – Învățarea punctului curent în care se afla robotul (Atenție!!! la unealta folosită pentru a atinge punctul)

b. Adăugare obiecte în spațiul de lucru (en.: Modeling)

Tabul Modeling conține controalele necesare pentru a crea componente sau obiecte noi, pentru a le grupa și a realiza măsurători (Fig. 2-14).

- Butonul *Component Group* realizează o grupare de obiecte, inițial nulă; obiectele se adaugă cu click stânga;
- Butonul *Empty Part* adaugă un nou obiect nul în spațiul de lucru;
- Butonul *Smart component* deschide o fereastră cu un editor numit *Smart Component Editor*. Acesta permite crearea, modificarea și legarea unei componente active/ inteligente folosind o interfață grafică;
- Butonul *Import Geometry* permite încărcarea de modele ralizate cu aplicații externe (este folosit în general la adaugarea în proiect a obiectelor fizice manipulate, respectiv a uneltelor/efectoarelor terminale pentru a crea o asemănare cât mai bună cu mediul fizic.

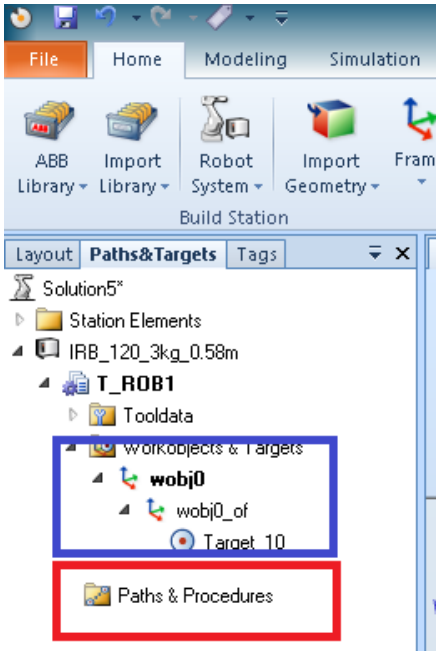


Fig. 2-12 – Definirea și verificarea punctelor învățate și a traiectoriilor

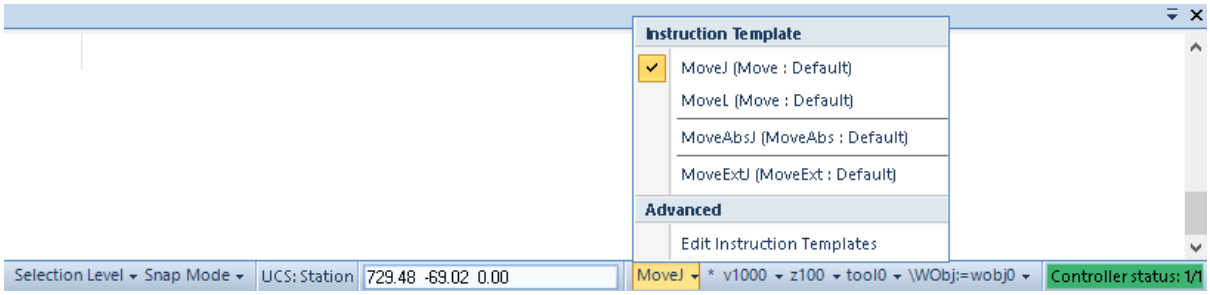


Fig. 2-13 – Selecția tipului instrucțiunii de mișcare

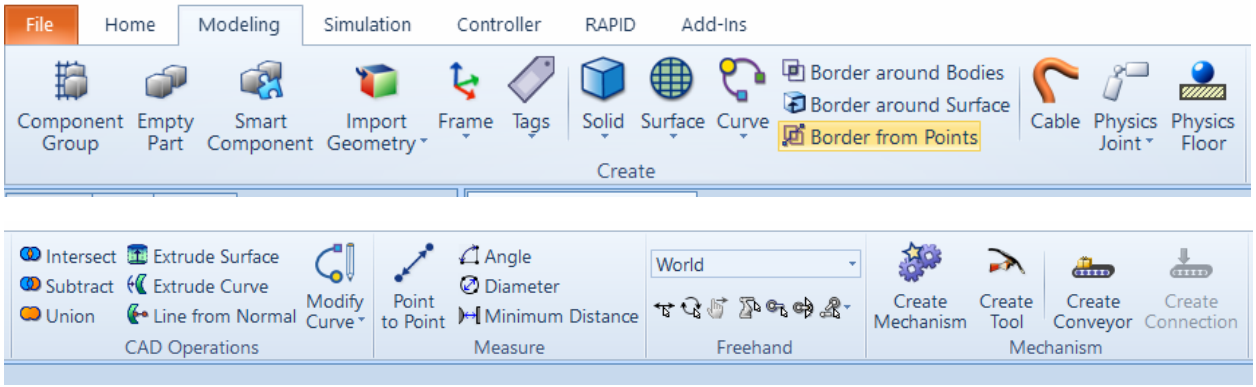


Fig. 2-14. Prezentare meniu Modeling

O componentă inteligentă este un obiect din RobotStudio care poate avea sau nu o reprezentare grafică și care are un comportament complex (se mișcă după anumite reguli, are o anumită logică implementată etc.). Acest tip de componentă este folosit pentru modelarea unor elemente precum: semnale și proprietăți matematice, primitive, senzori folosiți la detecția de prezență, acțiuni (clonarea și eliminarea unor de piese clonate, atașarea segmentelor între ele), manipulatoare (axe de mișcare liniare și de rotație), ș.a.

Editorul de componente programabile (Smart Component) (Fig. 2-15) următoarele componente:

- tabul *Compose* care este folosit pentru a adăuga obiectele ce vor fi folosite pentru componenta respectivă;
- tabul *Design* conține o afișare vizuală pentru structura componentei; include conexiunile interne, legăturile între obiecte și structura proprietăților;
- tabul *Properties and Bindings* conține proprietățile dinamice ale componentei și proprietățile legăturilor pentru componenta respectivă;
- tabul *Signals and Connections* se folosește pentru a defini semnalele de intrare /ieșire pentru componenta respectivă și conexiunile între semnale.

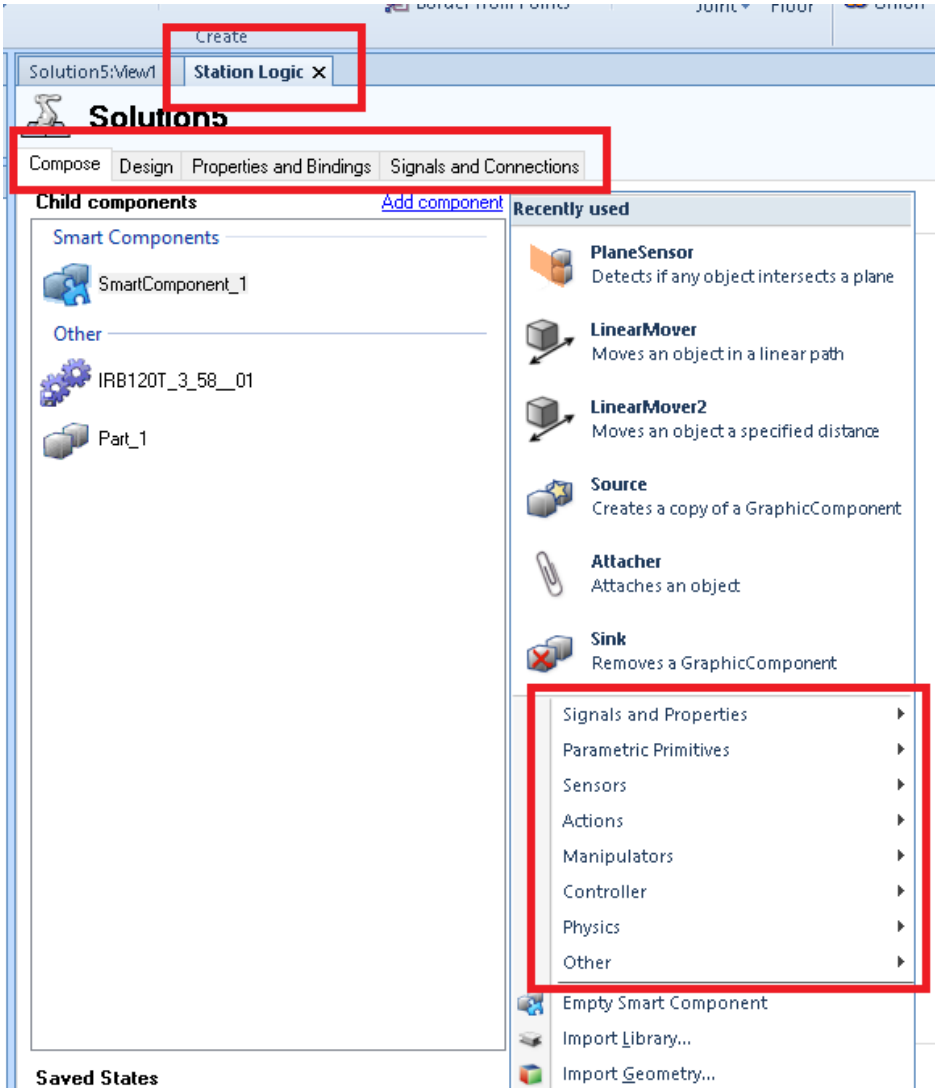


Fig. 2-15 – Crearea de componente active (en.: SmartComponent)

Pentru a crea obiecte în spațiul de lucru aplicația oferă următoarele facilități (Fig.2-14):

- Butonul Solid este folosit pentru a crea figuri geometrice spațiale de diferite forme și dimensiuni precum: pătrate, dreptunghiuri, conuri, cilindri, piramide și cercuri. Acestea se pot crea cu o anumită orientare în spațiul de lucru;
- Butonul Surface este folosit pentru a crea suprafețe de diferite forme și dimensiuni;
- Butonul Curve este folosit pentru a crea linii definite de două sau mai multe puncte. Astfel putem crea linii drepte, arce de cerc, curbe, elipse, etc.
  Pentru a realiza măsurători avem la dispoziție o grupare de butoane (Fig.2-14, jos):

  - butonul *Point to Point* pentru a măsura distanța între două puncte în spațiul de lucru;
  - butonul *Angle* pentru a măsura unghiul dintre două linii;
  - butonul *Diameter* pentru a măsura diametrul unui cerc;
  - butonul *Minimum Distance* măsoară distanța minimă dintre două obiecte din spațiul de lucru.

Toate elementele create vor fi adăugate în fereastra Layout (Fig. 2-16) sub următoarea structură:

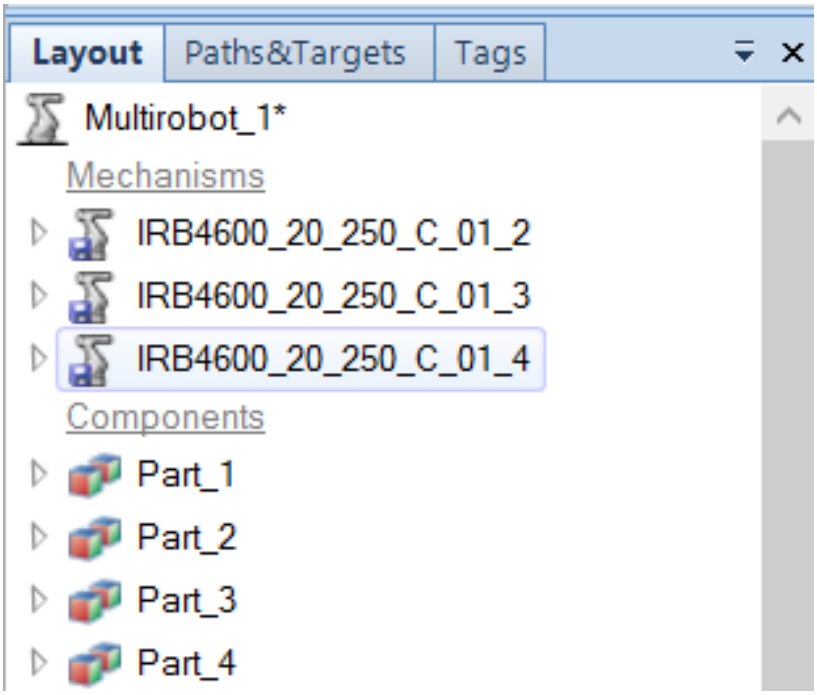


Fig. 2-16 – Fereastra Layout

Pentru a crea mecanisme avansate aplicația oferă următoarele facilități (Fig.14, dreapta jos):

- Butonul *Create Mechanism* este folosit pentru a crea un mecanism de tipul: robot, unealtă, axă externă sau un dispozitiv;
- Butonul *Create Tool* este folosit pentru a crea o unealtă nouă plecând de la un obiect deja existent sau nu. Avem opțiunea de a adăuga o masă pentru unealtă exprimată în kg, opțiunea de a defini un centru de greutate precum și un moment de inerție pentru unealta respectivă;
- Butonul *Create Conveyor* se folosește pentru a crea o banda transportoare iar butonul *Create Connection* pentru a realiza conexiunile între benzile transportoare din ansamblu.

c. Meniul de simulare (en.: Simulation)

Tabul Simulation (Fig. 2-17) conține controalele pentru setările inițiale, configurațiile, controlul, monitorizarea și înregistrarea simulărilor.

Toate simulările pornesc dintr-o stare inițială și în funcție de modul de rulare, ciclu individual respectiv rulare continuă, poate exista sau nu o stare finală, aceasta din urmă fiind diferită de starea inițială (ex.: o piesă este preluată și mutată în altă parte). Este recomandat să se acorde o atenție deosebită stărilor sistemului întrucât aici sunt salvate valorile I/O, pozițiile pieselor, obiectele prezente și alte elemente care influențează modul de rulare al traiectoriei. Astfel, este recomandat ca dupa fiecare simulare să se apeleze funcția de reset la starea inițială și în cazul unor modificări să se creeze o stare inițială nouă.

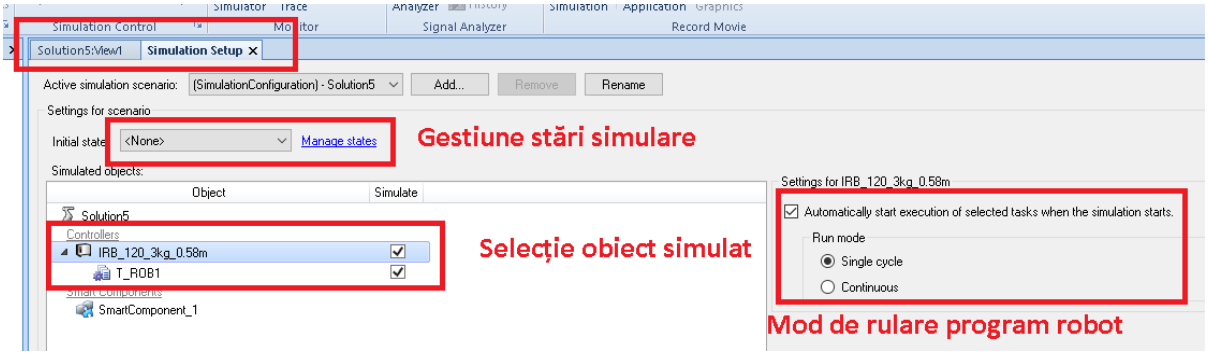


Fig.2-17 – Prezentare fereastră Simulation Setup

Pentru a realiza setările inițiale ale simulării se utiliză butoanele următoare (Fig. 2-17, sus stânga):

- Butonul *Simulation Setup* este folosit pentru a seta secvența în care se vor executa programele împreună cu punctul de start și pentru a crea diferite scenarii de funcționare pentru obiectele simulate în program. Avem de asemenea opțiunea de a alege o stare inițială care conține valorile variabilelor, valorile semnalelor de intrare/ieșire precum și structura grafică a spațiului de lucru;
- Butonul *Simulation Logic* are două opțiuni:
  - *Station Logic* care are caracteristici asemănătoare cu caracteristicile componentei inteligente și anume compunerea, introducerea proprietăților și a legăturilor, adăugarea semnalelor și conexiunile acestora și aranjarea sistemului;
  - *Event Manager* care este folosit pentru a adăuga evenimente și comportamentul pe care trebuie să îl realizeze sistemul la întâlnirea acelui eveniment.

Pentru a crea o noua stare a sistemului avem la dispoziție doua variante:

- Folosind butonul *RESET -> Save Current State* care va deschide o fereastră în care se poate denumi starea, se poate adăuga o descriere și se poate selecta ce valori sa fie salvate;
- Folosind butonul *Simulation Logic -> Station Logic*.

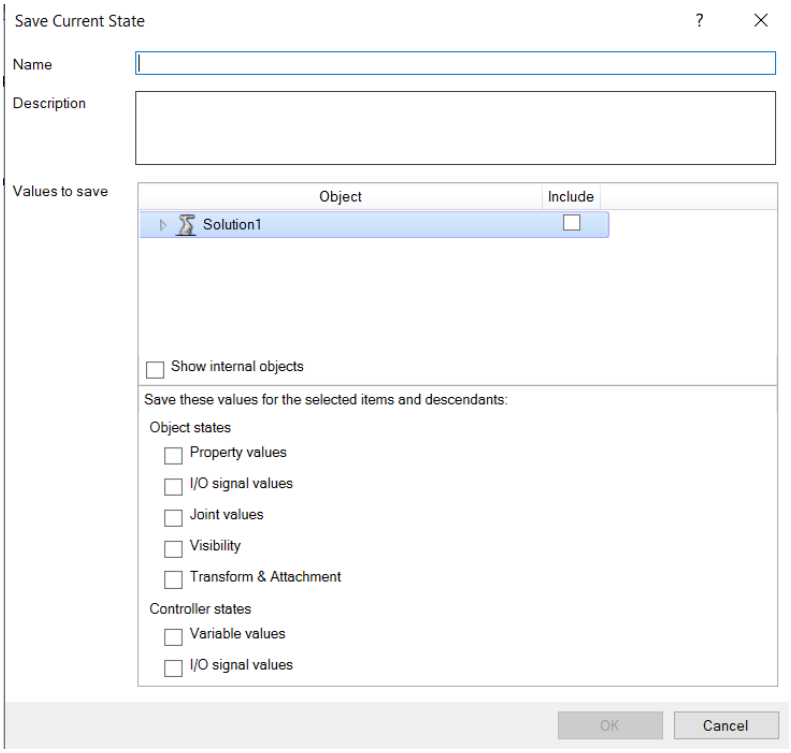


Fig.2-18 – Fereastra Save Current State

Pentru configurarea simulării se va folosi fereastra Simulation Setup (Fig. 2-19) în care se va selecta starea inițială (învățată anterior) și se vor bifa procesele ce se dorește a fi simulate (fiecare proces îi aparține unui robot). Tot din această fereastra mai putem alege dacă dorim ca programul să se execute o singură dată sau să fie executat ciclic (în acest caz este recomandat ca programul să conțină și o secvență de oprire).

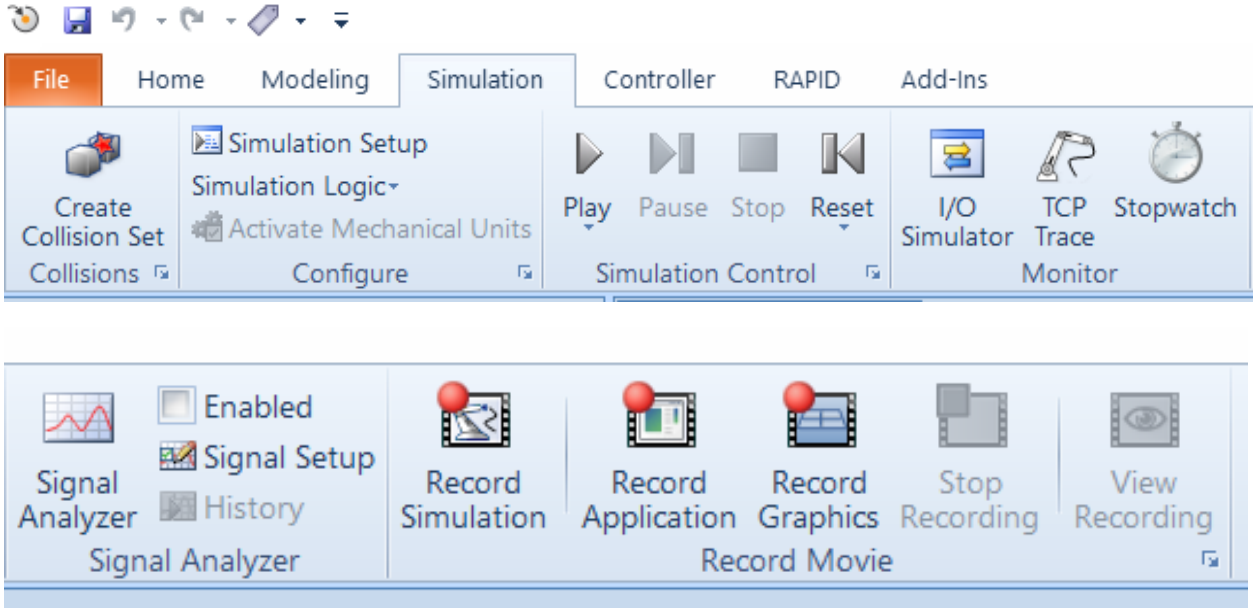


Fig. 2-19 – Prezentare meniu Simulation

Pentru prezentarea aplicației sub formă de film avem un grup de butoane la dispoziție pentru a controla simularea și anume butonul Play, Pause, Stop, Reset și Resume:

- Butonul *Play* pornește simularea cu condițiile inițiale și execută programul conform scenariului de funcționare ales de utilizator;
- Buton *Pause* întrerupe execuția simulării cu posibilitatea de a fi reluată din punctul de unde a rămas prin apăsarea butonului *Resume*;
- Butonul *Stop* întrerupe execuția programului fără posibilitatea dea mai putea fi reluată din punctul în care a rămas;
- Butonul *Reset* este folosit pentru a reseta sistemul și a-l aduce în starea inițiala. Tot prin intermediul acestui buton se poate salva o nouă stare respectiv se poate deschide o interfață pentru a putea organiza stările deja salvate sau pentru a înregistra o nouă stare.

Pentru monitorizarea aplicației avem la dispoziție următoarele butoane:

- Butonul *I/O Simulator* se folosește pentru a simula schimbarea unei ieșiri/intrări digitale pentru a verifica comportamentul sistemului la apariția acelei schimbări. Aici trebuie avut atenție la selecția modulului de care aparține intrarea/ieșirea dorită (Fig. 2-20).

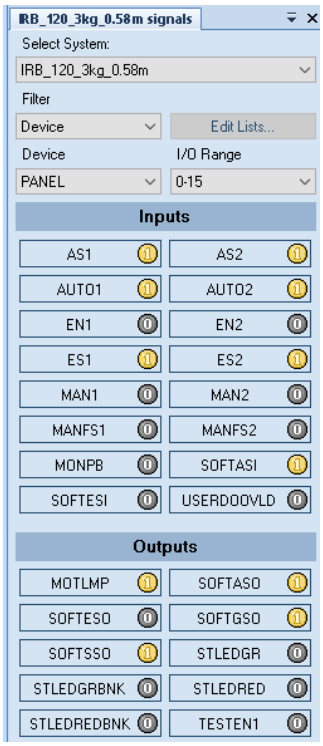


Fig. 2-20 – Meniul simulare semnale

- Butonul *Stopwatch* este folosit pentru a măsura durata dintre două puncte de pe traiectorie sau pentru întreg procesul;
- Butonul *Signal Analyzer* este util pentru a afișa și analiza semnalele ce provin de la controllerul robotului; având posibilitatea de a optimiza programul. Această opțiune este valabilă atât pentru controllerul virtual cat și pentru cel real;
- Butonul *Signal Setup* se folosește pentru a salva semnalele din program pentru a putea fi folosite la următoarea simulare;

- În acest tab mai există posibilitatea de a înregistra simularea folosind butonul *Record Simulation*; simularea este înregistrată pe disc sub forma unui clip video.

d. Meniul Controller

Tabul Controller (Fig. 2-21) conține controalele pentru a gestiona controllerul real și pentru a sincroniza, configura și pentru a organiza procesele pentru controllerul virtual.

Cele mai importante butoane din acest meniu sunt următoarele:

- Butonul *Add Controller* este folosit pentru a adăuga un nou controller dacă acesta nu există deja în aplicație;
- Butonul *Restart* este folosit pentru a reporni controllerul real sau virtual când acesta nu mai funcționează corect sau când se dorește ca anumite operații sa aibă efect, operații precum: schimbarea sistemului de coordonate a bazei oricărui robot ce aparține de controller, schimbarea configurației unui robot, adăugarea altor echipamente în sistem;
- Butonul *Events* se folosește pentru a vedea toate evenimentele ce au avut loc; acestea sunt colorate astfel: albastru pentru evenimente de informare, galben pentru atenționări și roșu pentru erori care necesită a fi eliminate pentru a putea continua.

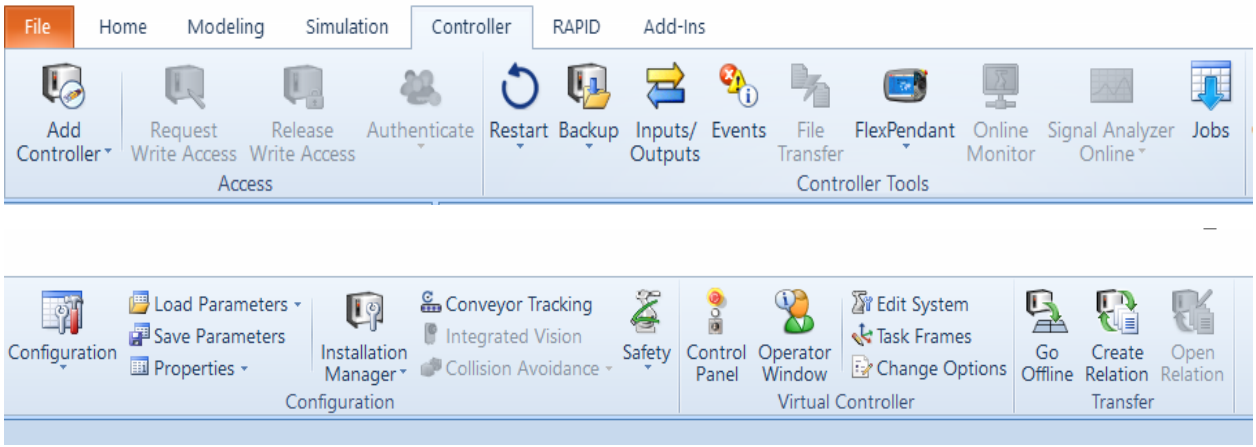


Fig. 2-21 – Prezentare meniu Controller

Pentru a gestiona controllerul virtual sau cel real avem la dispoziție o grupare de butoane denumită *unelte controller* (en: Controller Tools) ce cuprinde următoarele:

- Butonul *Inputs/Outputs* este folosit pentru a vizualiza și modifica semnalele de intrare și de ieșire pentru controller. Acestea pot fi de diferite tipuri: semnale digitale de intrare, semnale digitale de ieșire, semnale analogice de intrare, semnale analogice de ieșire, grupuri de semnale de intrare și grupuri de semnale de ieșire;
- Realizarea unei copii de rezervă se realizează folosind butonul *Backup*; cu acest buton se realizează și aducerea sistemului într-o stare anterioară folosind o copie de rezervă anterior creată;
- Butonul *Configuration* este foarte important deoarece cu ajutorul acestuia utilizatorul poate să vizualizeze și să editeze parametrii unei categorii a sistemului localizat în controller. Acesta este folosit pentru a vizualiza instanțe, parametri dar și pentru editarea acestora; pentru copierea și mutarea instanțelor între diferite categorii și pentru

adăugarea unor noi instanțe sau pentru ștergerea unora vechi. Pentru simulare de aici se pot adăuga semnale digitale respectiv fire de execuție suplimentare pentru controllerul robot. Este recomandat ca toate aceste configurații să fie făcute o dată la începutul proiectului și apoi menținute. Modificarea configurațiilor poate duce la pierderea traiectoriilor.

Acest meniu dispune de câteva opțiuni cum ar fi:

- *Controller* care prezintă setările generale ale controllerului (firele de execuție, opțiuni, setări Rapid, evenimente, condiții de resetare etc.);
- *Communication* care prezintă parametrii legați de comunicația controllerului cu alte dispozitive (setări IP, portul serial, protocoale de comunicație, client DNS, rutare IP etc.);
- *Motion* cu care se pot vizualiza și modifica toți parametrii referitori la mișcarea echipamentelor legate la controller;
- *Man-machine communication* care se referă la parametri pentru interacțiunea directă a utilizatorului cu echipamentele din sistem;
- *Add Signals* pentru a adăuga semnal noi;
- *I/O* pentru a vizualiza și modifica toți parametrii referitori la sistemul de intrare/ieșire. Din această opțiune se pot analiza toate semnalele din controller, elementele de rețea și permisiunile aferente.

Acest tab conține și un set de butoane disponibile doar atunci când folosim un controller virtual și un alt set de butoane atunci când avem conectat un controller real (fizic).

Pentru controllerul virtual avem la dispoziție butoanele:

- butonul *Virtual FlexPendant* care deschide o fereastră ce înfățișează un Flex Pendant virtual (Fig. 2-22) cu aceleași funcții ca sistemul decontrol real. Acesta este folosit pentru controlul manual al robotului/roboților din sistem;
- butonul *Control Panel* este folosit pentru a modifica modul de operare al sistemului. Acesta deschide o fereastră în care se poate comuta între modurile de funcționare:
  - auto – automat;
  - manual – cu viteză de mișcare redusă;
  - manual full speed – mod automat cu viteză maximă.
- Butonul *Edit System* care deschide o fereastră ce conține funcțiile pentru a crea și vizualiza configurațiile avansate de sistem cum ar fi: schimbarea controllerului, alterarea originii sistemului de axe general, calibrarea sistemului și setarea axelor externe. Fereastra prezintă următoarele butoane:
  a. butonul *Motors On* pentru a porni motoarele din sistem;
  b. *Emergency Stop* care este un buton pentru oprirea de urgență;
  c. *Enable Device* disponibil în modul manual pentru a porni motoarele;
  d. *Release Device* pentru a opri motoarele;
  e. *Reset Emergency Stop* care este disponibil după efectuarea opririi de urgență pentru a reseta starea.

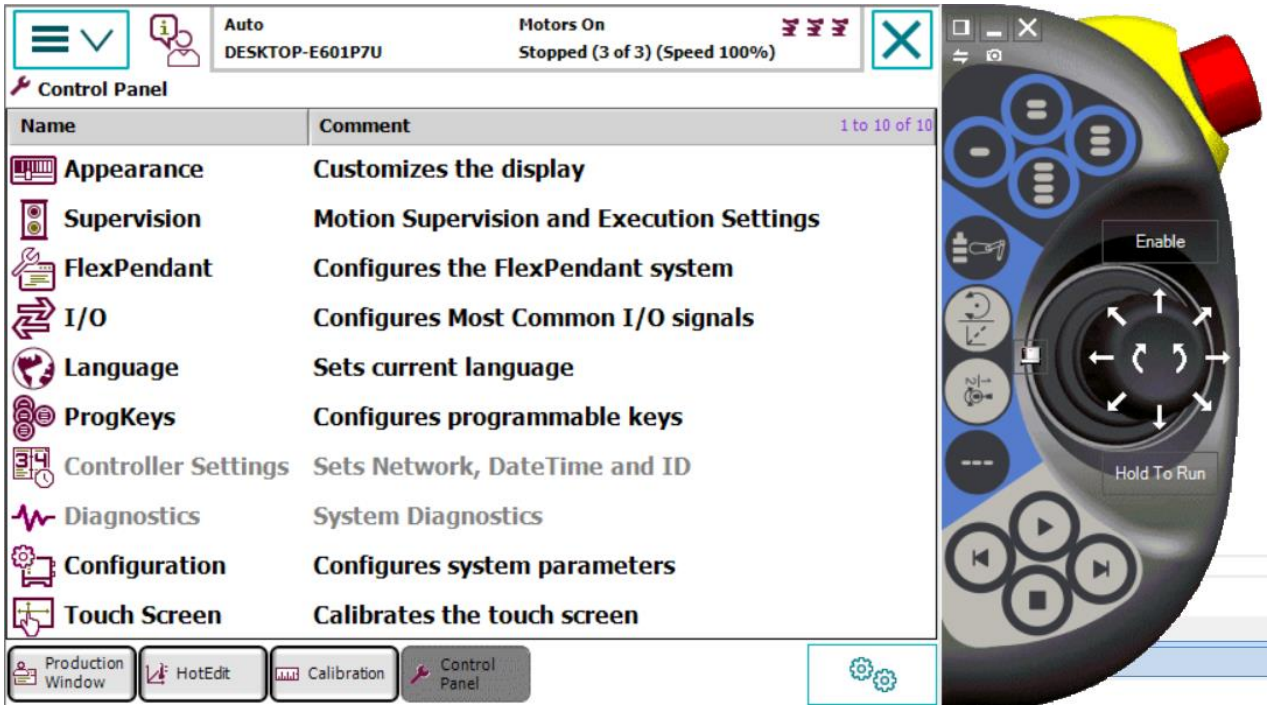


Fig. 2-22 – FlexPendant virtual

Pentru controllerul real avem disponibile butoanele:

- Butonul *Request Write* Access pentru a permite scrierea pe controller;
- Butonul *Release Write* Access pentru a nu mai permite utilizatorilor să poată scrie pe controller;
- Folosind *Authenticate* putem sa ne autentificăm cu un alt utilizator sau să ne deconectăm; de asemenea putem să editām conturile utilizatorilor;
- Fereastra *File Transfer* se folosește pentru a transfera fișiere între un dispozitiv de tip PC și controller;
- Butonul *FlexPendant Viewer* pentru a genera automat o captură de ecran a FlexPendant-ului ;
- Butonul *Properties* este folosit pentru a redenumi controllerul, a seta data și ora, a modifica ID-ul controllerului, a vizualiza proprietățile controllerului și a sistemului;
- Butonul *Online Monitor* permite utilizatorului să monitorizeze robotul conectat la controller de la distanța.

Elementele menționate mai sus pot fi accesate și făcând click dreapta pe robotul/roboții din soluția curentă (Fig. 2-23). Din acest meniu (click dreapta controller robot -> *Change options*) se pot face configurațiile controllerului implicit (*Change Options*), astfel putându-se adăuga opțiuni precum multitasking (*Engineering tools -> 623-1 Multitasking*), comunicație retea (*Communication -> 616-1 PC Interface*), urmărire conveior/lucrul cu axe externe (*Motion coordination -> 1552-1 Tracking Conveyor Unit*), ș.a.

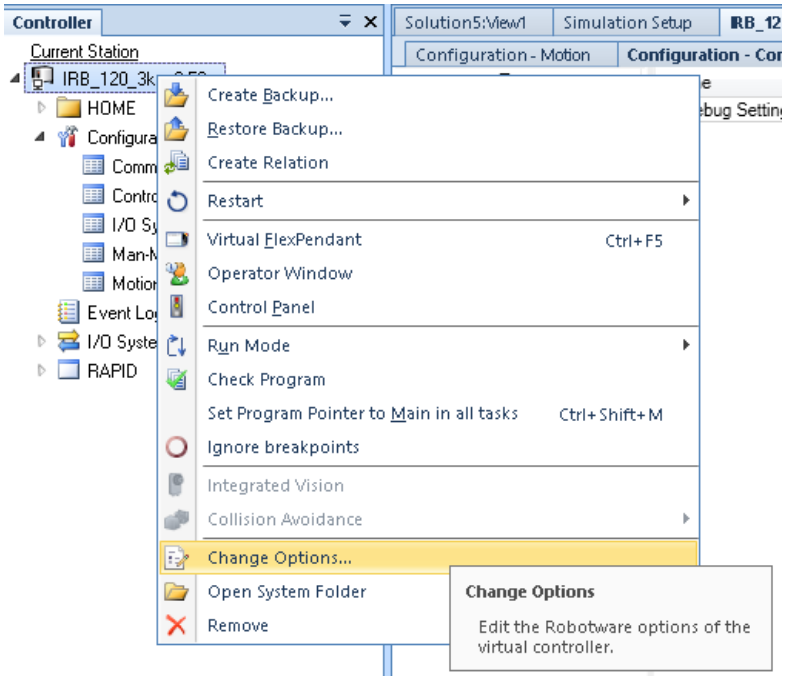


Fig. 2-23 – Accesarea opțiunilor controllerului stației curente

e. Meniul RAPID – Editorul de programe

Tabul RAPID (Fig.2-24) conține uneltele și funcționalitățile necesare pentru a crea, edita, și pentru a gestiona programele în limbajul RAPID. Acest tab mai prezintă și butoane pentru editarea textului (copiere text, tăiere text, adăugare text, spațiere text etc.) și opțiuni de căutare a unui text.

- Butonul *Synchronize* se folosește pentru a sincroniza sistemul cu codul RAPID alegând opțiunea *Syncronize to RAPID* și respectiv pentru a sincroniza codul RAPID cu sistemul alegând opțiunea *Syncronize to Station*;
- Butonul *Apply* care se folosește pentru a aplica modificările aduse codului;
- Butonul *Format* pentru a pune codul în formatul standard Rapid.

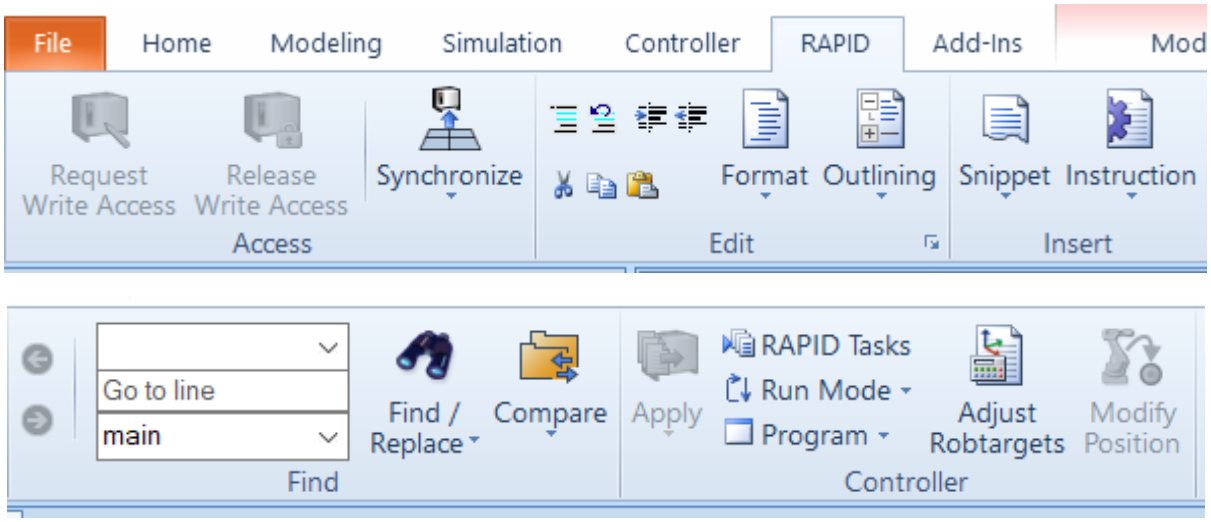


Fig.2-24 (a) – Prezentare meniu RAPID

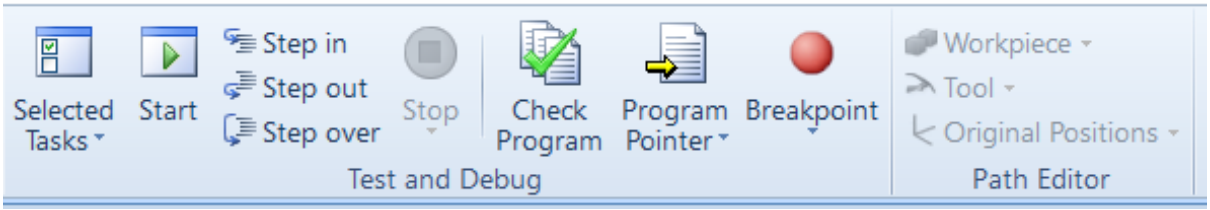


Fig.2-24 (b) – Prezentare meniu RAPID

Programul în limbajul Rapid este accesibil din fereastra *Controller* situată în stânga ferestrei programului RobotStudio (Fig. 2-25); care este vizibilă atunci când este selectat tabul *RAPID* sau tabul *Controller*.

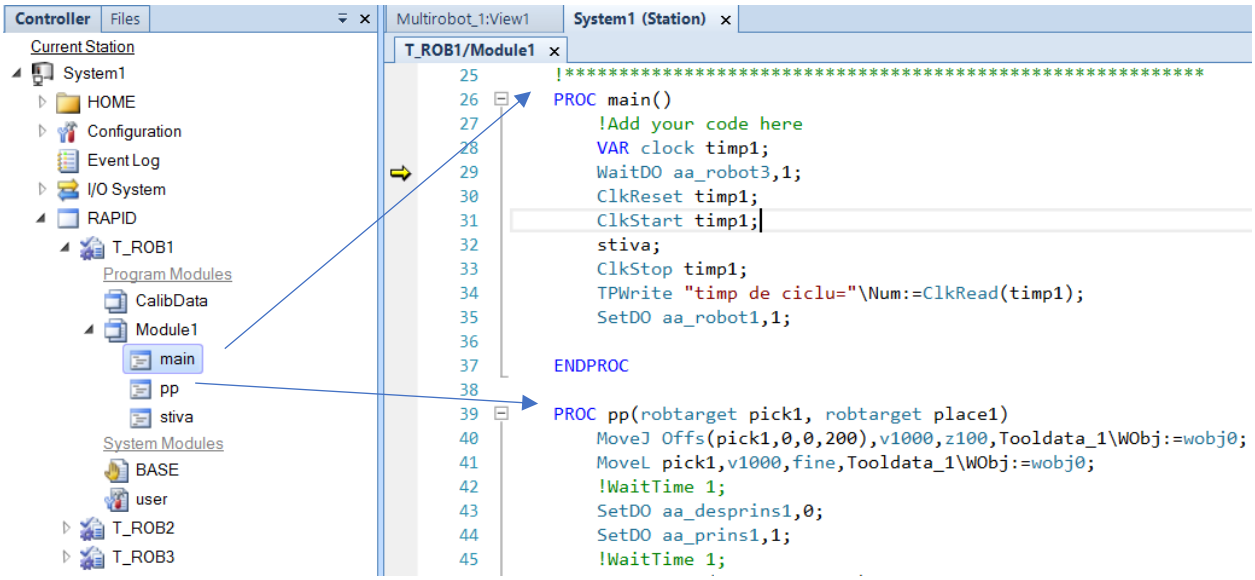


Fig. 2-25 – Fereastra Controller

În fișierul RAPID avem o structură modulară corespunzătoare firelor de execuție repartizate pentru fiecare robot în parte. În modulul creat de utilizator se editează codul Rapid prin crearea de proceduri. Limbajul Rapid este un limbaj de programare de nivel înalt, intuitiv asemănător cu limbajele C, C++, Java etc.

4. Procedura adăugare unealtă

Pas 1: creare unealta grafică;

Pas 2: atașare unealtă la robot prin drag&drop pe robot în meniul Layout;

Pas 3: specificare dimensiune unealta -> *Paths&Targets -> Tooldata -> Create Tooldata*;

Pas 4: selecție unealtă -> *Paths&Targets -> Tooldata -> Set as active*;

Pas 5: verificare unealtă -> reorientare robot și verificare că se rotește în jurul punctului dorit (vizual cel specificat la pasul 1);

Pas 6: specificare acționări unealtă.

5. Export proiect

În RobotStudio versionarea se face prin exportul proiectului și importarea lui sub forma de fișier **.rspag* (Fig. 2-26). Importarea se face prin lansarea fișierului exportat mai sus. Trebuie să se dea un nume diferit proiectului în caz ca acesta este pe același PC.

Fig. 2-26 – Exportul unui proiect RobotStudio

6. Sistem de coordonate

Toate operațiile de definire de obiecte, mișcare și rotație se fac relativ la un sistem de coordonate. Cele mai des folosite sunt sistemele de coordonate sunt sistemul de coordonate local (relativ la piesa la care se face operația) și global (relativ la celulă) (Fig. 2-27).

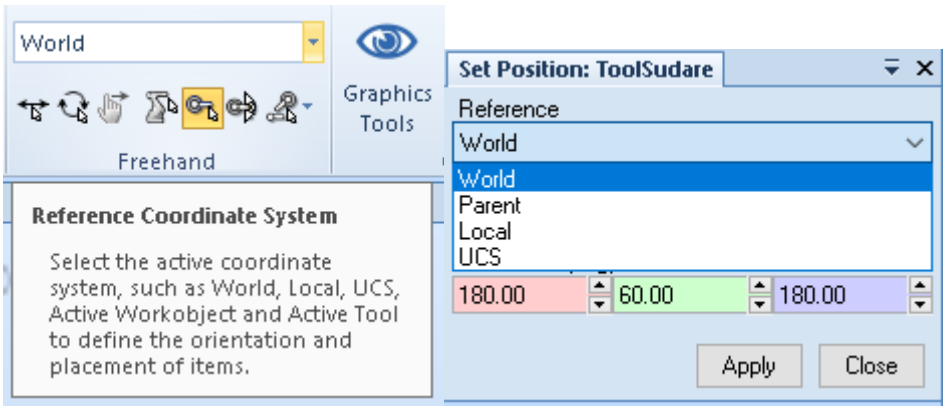


Fig. 2-27(a) – Selecția unui sistem de coordonate pentru realizarea unor operațiuni de poziționare în mediul de lucru

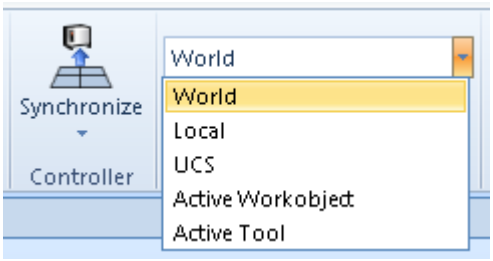


Fig. 2-27(b) – Selecția unui sistem de coordonate pentru realizarea unor operațiuni de poziționare în mediul de lucru

7. Exerciții

Realizare program care atinge 3 laturi ale unui cub.

8. Intrebari

a) De ce în cazul unui punct definit offline există posibilitatea ca robotul să nu îl poată atinge?
b) Dați exemple de tipuri de unelte robot și specificați clasele de aplicații în care se folosesc.

Capitol 3: Principii generale de lucru
Procedura *pick and place*|
Modul de lucru grafic

Cuprins

1 Tipuri de efectoare terminale și modelarea acestora 46
2 Conceptul de punct condus și sistem de coordonate atașat efectorului terminal 47
3 Adaugare element vizual sub formă de obiect manipulat 48
4 Adaugare transformare de tip tool 49
5 Adaugare semnale ieșire pentru controlul gripperului 52
6 Realizarea unui obiect de tip gripper static 56
7 Realizarea unui obiect de tip gripper cu grade de libertate 58
8 Procedura de paletizare a unui obiect 59
9 Verificare traiectorie, conceptul de configurație robot 61
10 Crearea unui efector terminal cu mai multe capete 62
11 Crearea unei traiectorii curbilinii 63
12 Realizarea de măsurători pe un mediu existent 63
13 Exerciții 64
14 Intrebări 64

1 Tipuri de efectoare terminale și modelarea acestora

Sarcina unui robot industrial este de a manipula un efector terminal. Acesta din urmă este montat la capătul unui braț robotizat (pe segmentul terminal denumit flanșă) și este conceput pentru a interacționa cu mediul. Natura exactă a acestui dispozitiv depinde de sarcina realizată de robot, efectorul terminal fiind de tip gripper (manipulare obiecte) sau unealtă (realizarea unei operațiuni asupra unui obiect, ex.: sudură, vopsire, ș.a.) (Fig. 3-1). În cadrul acestui material vom opera cu efectoare terminale care manipulează obiecte, acționarea lor realizându-se printr-un număr restrâns de instrucțiuni (ex.: prinde/eliberează) implementate folosind ieșiri digitale (Fig. 3-2). Pentru o simulare cât mai reală, RobotStudio permite realizarea grafică a funcționării a gripperelor cu degete (prindere prin impact), cu ace (prindere prin inserție) și a gripperelor atractive (gripper magnetic). În cele ce urmează vor fi prezentate modalitățile în care un efector terminal poate fi creat și atașat unui robot industrial și ulterior acționat.

Fig. 3-1 – Tipuri de efectoare terminale (sudură, manipulare, inspecție)

Fig. 3-2 – Tipuri de grippere (cu vid, cu degete, magnetic)

2 Conceptul de punct condus și sistem de coordonate atașat efectorului terminal

În mod implicit, punctul condus al unui robot industrial este capătul ultimului segment intitulat flanșă (en.: tool mounting flange – TMF). Pentru a se trece de la sistemul flanșei la sistemul uneltei se definește o transformare de tip unealtă (Fig. 3-3). În cazul unui efector terminal de tip gripper punctul condus este localizat în mijlocul degetelor care realizează prinderea.

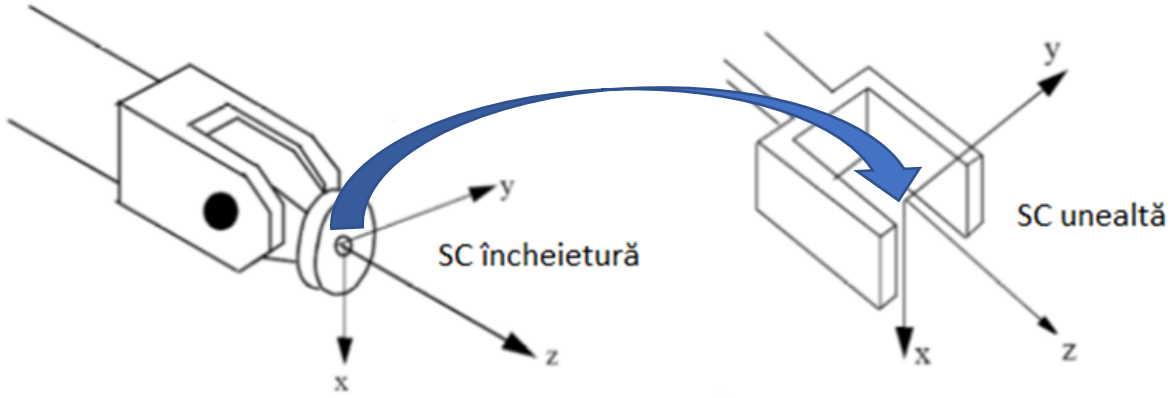


Fig. 3-3 – Transformarea TOOL

3 Adaugare element vizual sub formă de obiect manipulat

Pentru a simula vizual un efector terminal trebuie adăugat un obiect grafic și atașat robotului. În acest sens se va crea un obiect solid din meniul Modelling care să imite cât mai bine unealta fizică prin una din urmatoarele proceduri: a) adaugarea unui obiect de tip cub, cilindru, con sau piramida (Modelling > Solid), b) importarea unui obiect realizat într-o aplicație externă (Modelling>Import Geometry>Browse for geometry), c) realizarea unui obiect prin compunerea, substragerea sau intersecția unor obiecte existente (Modelling > Intersect/Substract/Union). Obiectul rezultat care se găsește în secțiunea Components a ferestrei Layout va fi ulterior atașat brațului robotic (Fig. 3-4). Atașarea componentei la piesă se va face suprapunând cele două sisteme de coordonate (sistemul de coordonate atașat punctului condus al robotului, respectiv sistemul de coordonate al componentei). În acest sens se va avea grijă ca sistemul de coordonate al piesei să fie poziționat în locația corectă. Dacă nu este cazul atunci se va modifica originea sistemului de coordonate al componentei din fereastra Layout > click dreapta componentă > Modify > Set local origin (Fig. 3-5).

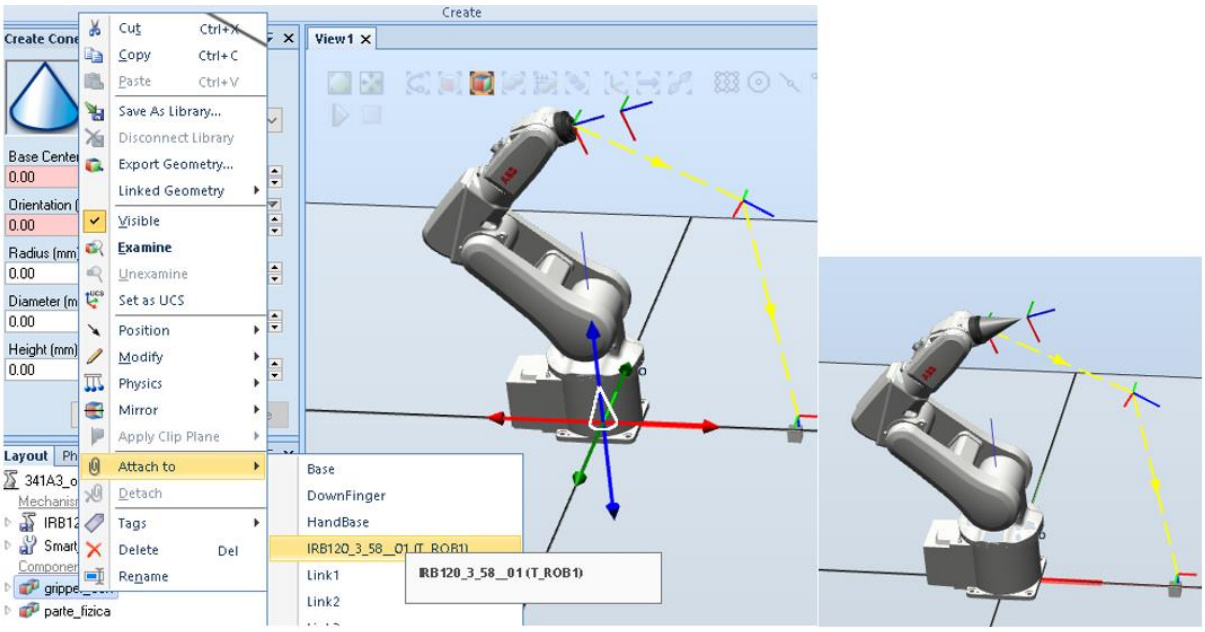


Fig. 3-4 – Procedura de atașare a unei componente la brațul robotic

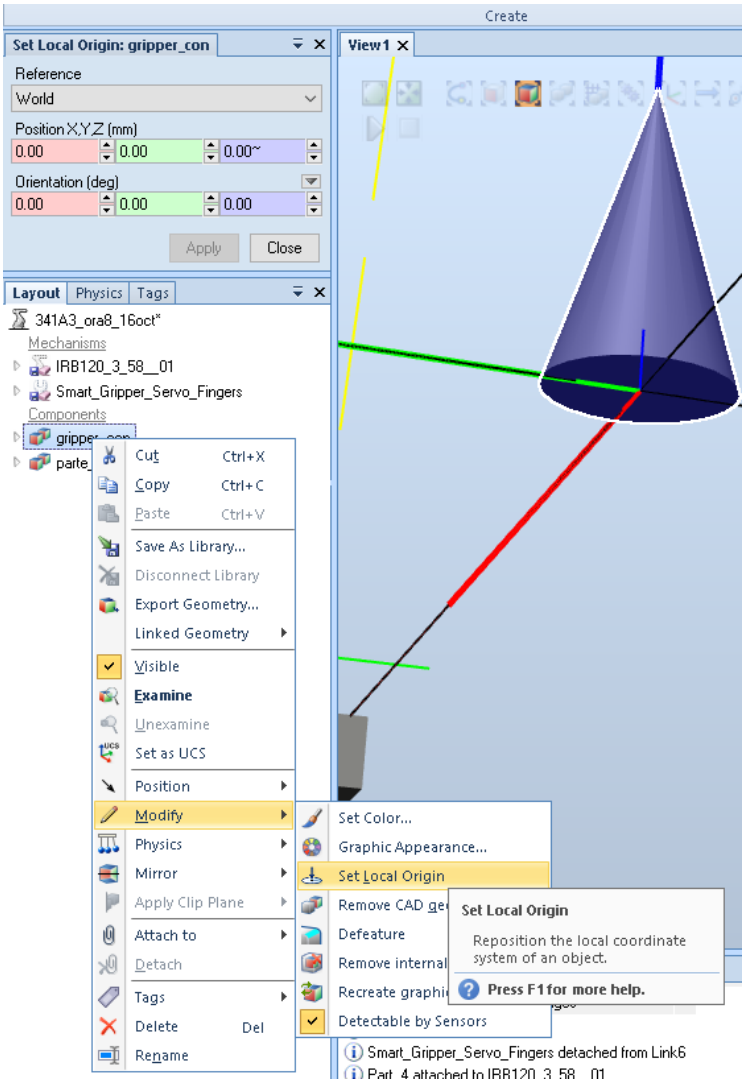


Fig. 3-5 – Procedura de schimbare a sistemului de coordonate al unei componente

4 Adaugare transformare de tip tool

După realizarea procedurii de atașare a unei componente la brațul robotic, punctul condus al robotului rămâne tot centrul flanșei. Schimbarea acestui punct se face prin definirea unei transformări de tip TOOL (perspectivă *Home* > fereastră *Paths&Targets* > selecție ramură *Tooldata* > click dreapta și *Create Tooldata*) (Fig. 3-6). În cazul simulării este sufficient să se definească cele 6 componente (3 translații și 3 rotații relative la sistemul de coordonate al punctului condus al brațului robotic). Pentru aplicații pe echipamente fizice modalitatea de operare cu efector terminal este îmbunătățită prin specificarea centrului de masă și a greutății pentru a putea lua în considerare aceste elemente în controlul mișcării.

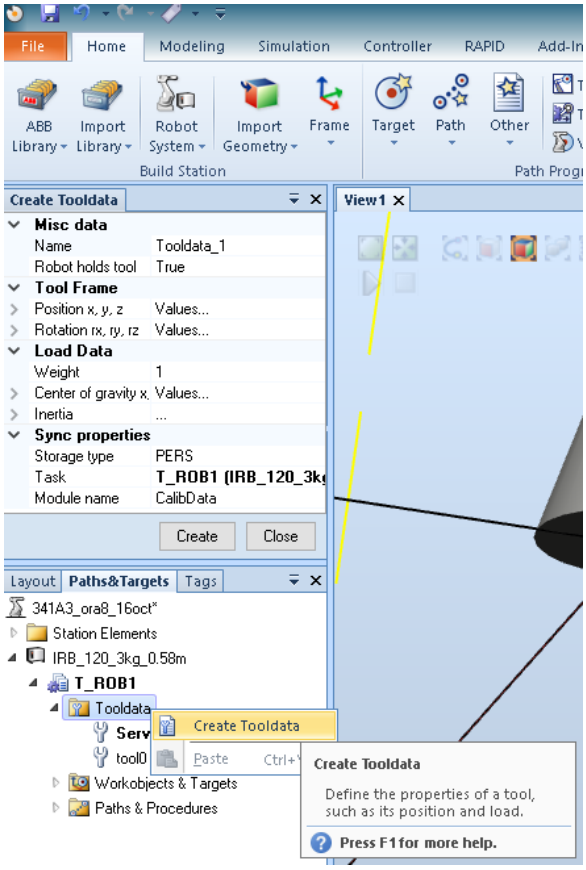


Fig. 3-6 – Definirea unei transformări de tip TOOL

Procedura de mai sus poate fi reluată pentru mai multe unelte, acestea fiind utilizate ulterior selectiv în funcție de ceea ce este montat/atașat de flanșa robotului. La nivel de instrucțiune de mișcare se specifică printre alți parametrii și unealta cu care operează robotul (Fig. 3-7).

```
MoveJ [\Conc] ToPoint [\ID] Speed [\V] | [\T] Zone [\Z] [\Inpos]
    Tool [\WObj]

MoveJ p5, v2000, fine \Inpos := inpos50, grip3;
```

Fig. 3-7 – Structură instrucțiune MOVEJ (sus) și exemplu utilizare cu transformarea TOOL *grip3* (jos)

Instrucțiunea MOVEJ este folosită pentru a deplasa robotul dintr-un punct în altul liniar în articulații traiectoria punctului condus fiind non-liniară în cele mai multe cazuri.

Structura:

MoveJ [\Conc] , ToPoint , [\ID] , Speed , [\V] | [\T] , Zone , [\Z] ,[\Inpos] , Tool , [\WObj];

Unde :

- [\Conc] Concurrent este folosit pentru a executa instrucțiunile următoare în timp ce robotul se află în mișcare. În general nu se folosește;
- ToPoint de tip robtarget punctul de destinație al robotului;
- [\ID] Synchronization id este folosit la sincronizarea mișcării prin grupare;
- Speed viteza cu care se va executa mișcarea în procente;
- [\V] velocity este folosit pentru a seta viteza în mm/s;
- [\T] time -timpul de mișcare al robotului;
- Zone - reprezintă zona unde se poate mișca robotul;
- [\Z] zone – acuratețea cu care respecta atingerea punctului ToPoint; se specifică în mm;
- [\Inpos] In position – poziția de oprire;
- Tool specifică unealta folosită pentru mișcare;
- [\WObj] sistemul de coordonate care este folosit ca referință în instrucțiune.

În modul grafic o nouă unealtă este specificată ca unealtă implicită astfel (navigare până la unealta dorita > click dreapta pe ea și *Set as active*. Corectitudinea instalării uneltei se poate face în modul mișcare de rotație (reorientare) prin verificarea faptului că robotul se rotește în jurul punctului nou definit, respectiv se deplasează de-a lungul axelor noi definite (Fig. 3-8).

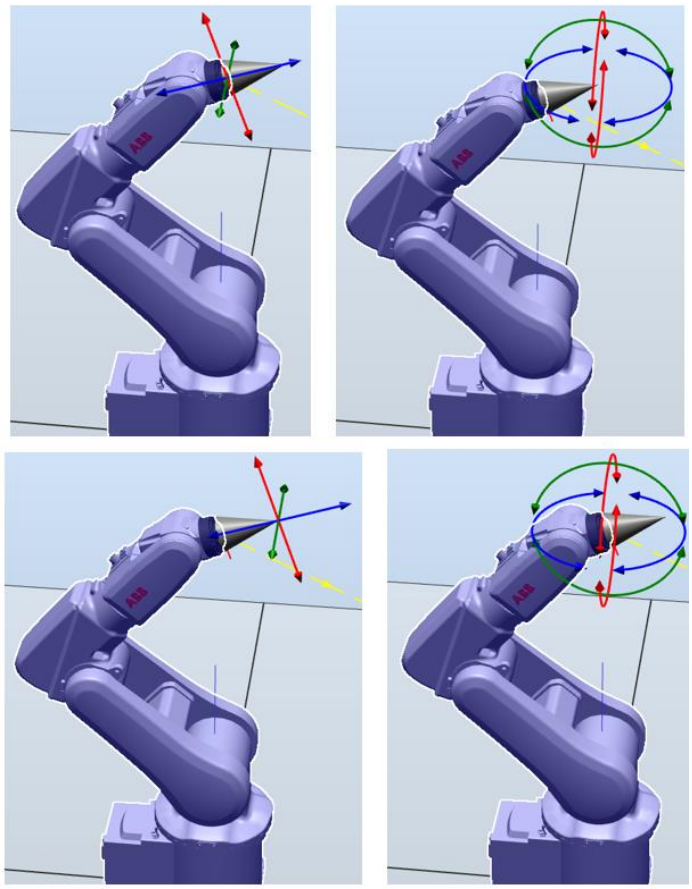


Fig. 3-8 – Diferența dintre locația sistemului de coordonate unealtă și a sistemului de coordonate flanșă

5 Adaugare semnale ieșire pentru controlul gripperului

Pentru a crea o unealtă funcțională trebuie sa creăm cate două semnale digitale de ieșire pentru fiecare unealtă în parte; din tabul Controller alegem butonul Configuration->I/O System ->Click dreapta pe Signal ->new signal. Un semnal pentru activarea prinderii și celălalt pentru desprindere(Controllerul trebuie să fie repornit pentru ca semnalele sa fie aplicate).

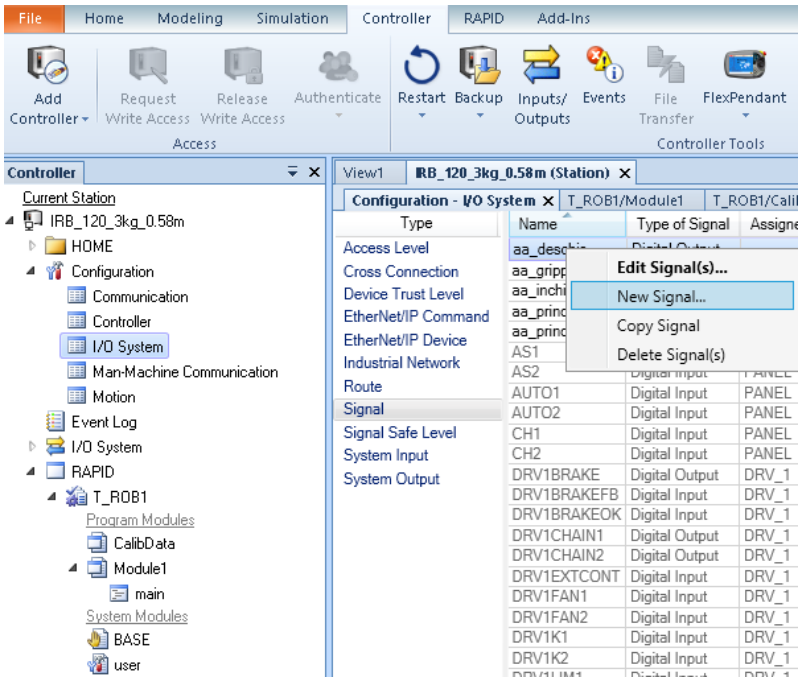


Fig. 3-9 – Crearea unui nou semnal robot

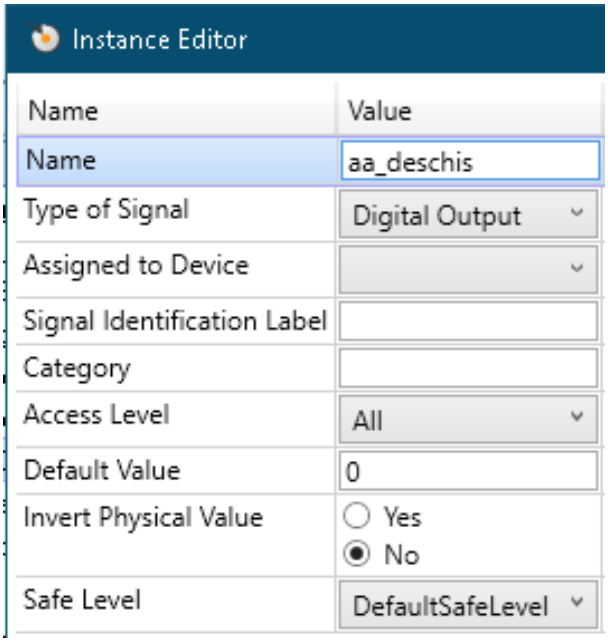


Fig. 3-10 – Configurarea unui semnal digital de ieșire (necesită restartarea controllerului *Controller -> Restart -> Warm restart*)

Vizualizarea și modificarea semnalelor în mod vizual se face din *Simulation > I/O Simulator* prin selecția dispozitivului corespunzător la care a fost atașat semanulul la punctul anterior prezentat în Fig. 3-11.

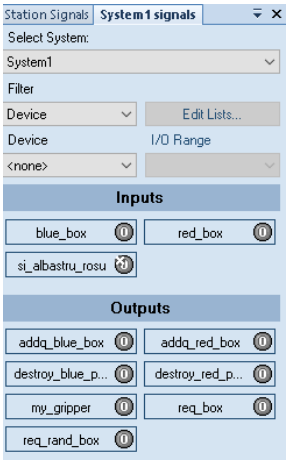


Fig. 3-11 – Modalitate acces semnale digitale în meniul de simulare

Odată create semnalele acestea trebuie să producă anumite evenimente la activarea lor (trecerea din 0 în 1 logic). Din tabul Simulation se alege butonul Simulation Logic și opțiunea Event manager care va deschide o fereastră (Fig. 3-12) care este folosită pentru a adăuga pentru fiecare unealtă un eveniment care să prindă piesa de unealta respectivă și un eveniment care să desprindă piesa de unealtă.

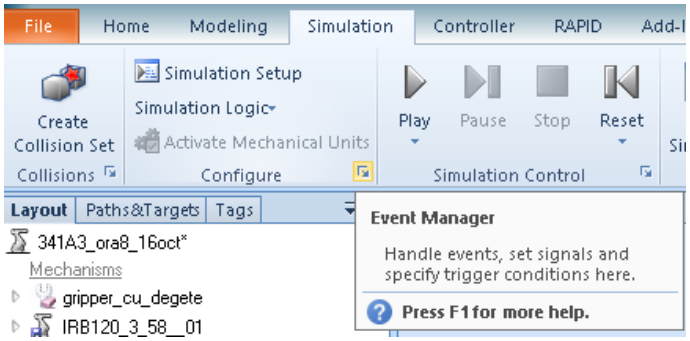


Fig. 3-12 – Accesarea ferestrei de gestiune a avenimentelor (Event Manager)

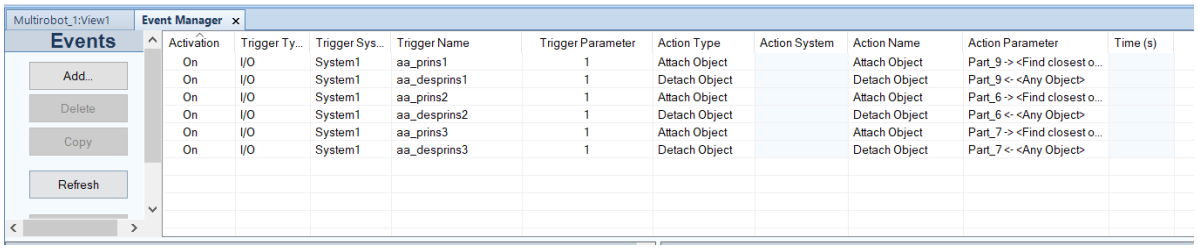


Fig. 3-13 – Fereastra Event Manager

Pentru a se realiza prinderea respectiv desprinderea corectă a pieselor trebuie ca orientarea sistemului de coordonare local al fiecărei piese să corespundă cu orientarea sistemului de coordonate al uneltei (se recomandă a avea originea în vârful uneltei).

Există o serie de motive pentru care este necesară simularea grafică a manipulării obiectului de interes. Dintre aceste motive amintim detecția de coliziuni obiect mediu de lucru, vizualizare anvelopă completă robot, verificare accesibilitate puncte interes sau demonstrație vizuală a modului de lucru. Prinderea unui obiect cu gripperul, respectiv eliberarea unui obiect din gripper se poate realiza fie prin adăugarea de evenimente de tip atașare/dezatașare sau prin

simularea gravitațională/forța de impact dintre degetele gripperului și obiectul manipulat. În cele ce urmează va fi prezentată modalitatea de atașare a obiectului/obiectelor de gripperul robotului în vederea manipulării.

Pașii necesari prinderii/dezprinderii obiectului de interes de gripper odată ce semnalele de control gripper și transformarea TOOL au fost definite sunt: i) adăugare eveniment din Event Manager (Add) > Selecția tipului de activare (pe baza schimbării valorii semnalului) > Selecția semnalului și a valorii la care se se execută evenimentul > Selecția acțiunii de realizat (Fig. 3-13); în cazul de față vom opera cu prindere/desprindere (en.: attach/detach object) > Definirea (Fig. 3-14) obiectului manipulat, a acțiunii (prindere/desprindere), a obiectului de care se prinde și eventual a distanței (offset) de prindere. Pentru aplicații generice în care nu se știe a priori cu ce piesă se operează (pentru a fi selectată din lista de piese prinse respectiv desprinse) există posibilitatea de a prinde cea mai apropiată piesă de obiectul selectat respectiv, să se desprindă toate piesele de obiectul selectat. Pentru cazul în care se operează cu un singur obiect (gripper cu prindere simplă) acest mod de operare funcționează corect. În cazul gripperelor cu prindere multiplă va trebui să se definească câte un gripper simplu pentru fiecare prindere. În momentul atașării obiectului de gripper sistemul de coordonate al său va fi suprapus peste sistemul de coordonate gripper. Astfel, se recomandă ca punctul de prindere si sistemul de coordonate al obiectului să coincidă, altfel piesa va fi atașată într-un mod nefiresc grafic.

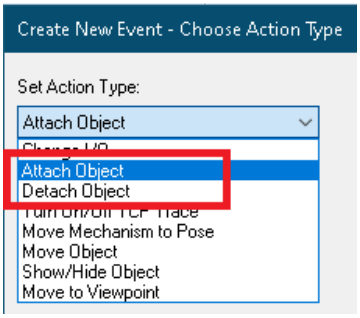


Fig. 3-14 – Tipuri de evenimente ce se pot realiza din *Event Manager*

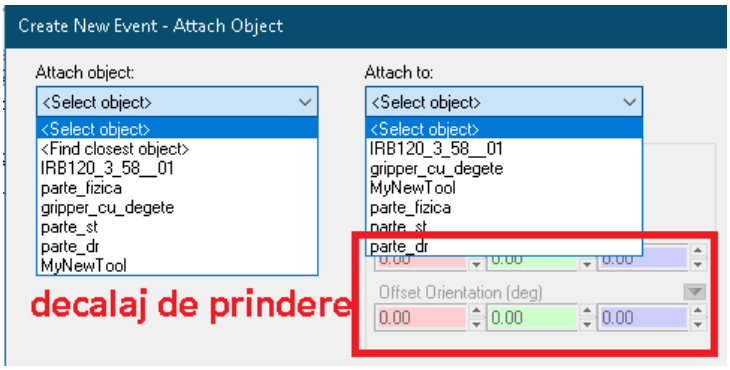


Fig. 3-15 – Definirea evenimentului de prindere/desprindere, respectiv specificarea decalajului de prindere

Pentru prinderea unei piese cu un gripper acționat de o ieșire digitală, succesiunea de mișcări/comenzi este următoarea: a) mișcare punct condus (ex.: centru magnet, mijlocul distanței dintre degete) peste punctul de prindere cu selecția opțiunii fine (punctul este punct final și nu punct de trecere) din instrucțiunea de mișcare MOVEJ, b) acționarea ieșirii aferente prinderii piesei SetDO, c) realizarea unei temporizări pentru prinderea corectă a piesei (Obs.:

această temporizare este absolut necesară în mediul fizic, valoarea ei fiind dată de natura acționării – pneumatic, electric, etc) WaitTime, d) mișcare pe traiectoria de depunere.

Instrucțiunea MOVEJ este folosită pentru a deplasa robotul dintr-un punct în altul liniar în articulații traiectoria punctului condus fiind non-liniară în cele mai multe cazuri.

Structura:

MoveJ [\Conc] , ToPoint , [\ID] , Speed , [\V] | [\T] , Zone , [\Z] ,[\Inpos] , Tool , [\WObj];

Unde :

- [\Conc] Concurrent -este folosit pentru a executa instrucțiunile următoare în timp ce robotul se află în mișcare. În general nu se folosește;
- ToPoint de tip robtarget -punctul de destinație al robotului;
- [\ID] Synchronization id -este folosit la sincronizarea mișcării prin grupare;
- Speed -viteza cu care se va executa mișcarea în procente;
- [\V]- velocity este folosit pentru a seta viteza în mm/s;
- [\T]- time -timpul de mișcare al robotului;
- Zone - reprezintă zona unde se poate mișca robotul;
- [\Z] zone – acuratețea cu care respecta atingerea punctului ToPoint; se specifică în mm;
- [\Inpos] In position – poziția de oprire;
- Tool -se specifică unealta folosită pentru mișcare;
- [\WObj] -sistemul de coordonate care este folosit ca referință în instrucțiune.

Instrucțiunea MOVEL este folosită pentru a deplasa robotul dintr-un punct în altul urmărind o mișcare liniară.

Structura:

MoveL [\Conc] , ToPoint , [\ID] , Speed , [\V] | [\T] , Zone , [\Z] , [\Inpos] , Tool , [\WObj] , [\Corr];

Unde:

- [\Conc] Concurrent -este folosit pentru a executa instrucțiunile următoare în timp ce robotul se află în mișcare. În general nu se folosește;
- ToPoint de tip robtarget -punctul de destinație al robotului;
- [\ID] Synchronization id -este folosit la sincronizarea mișcării prin grupare;
- Speed -viteza cu care se va executa mișcarea în procente;
- [\V]- velocity este folosit pentru a seta viteza în mm/s;
- [\T]- time -timpul de mișcare al robotului;
- Zone reprezintă zona unde se poate mișca robotul;
- [\Z] zone – acuratețea cu care respecta atingerea punctului ToPoint; se specifică în mm;
- [\Inpos] In position – poziția de oprire;
- Tool -se specifică unealta folosită pentru mișcare;
- [\WObj] -sistemul de coordonate care este folosit ca referință în instrucțiune;
- [\Corr] -corecția.

Instrucțiunea SetDO este folosită pentru a schimba valoarea unui semnal de iesire digital cu sau fără întârziere sau sincronizare. Pentru acționare gripper.

Structură:

SetDO [\SDelay]|[\Sync] , Signal , Value;

Unde:

- [\SDelay] întârzierea care se dorește în s;
- [\Sync] sincronizare;
- Signal numele semnalului care se dorește a fi schimbat;
- Value valoare dorită pentru semnal; 0 sau 1.

Instrucțiunea WaitTimeeste folosită pentru a aștepta o anumită perioadă de timp. Această instrucțiune poate fi, de asemenea, utilizată pentru a aștepta până când robotul și axele externe s-au oprit..

Structură:

WaitTime [\InPos] Time

Unde:

- [\InPos] Dacă se folosește acest argument, atunci robotul și axele externe trebuie să se oprească înainte ca timpul de așteptare să înceapă să fie numărat.;
- Time - Timpul, exprimat în secunde, pentru care programul așteaptă. Min. valoare 0 s. Max. valoare fără limită. Rezoluție 0,001 s

Pentru verificare se poate realizsa un program din interfața grafică și apoi sincroniza cu programul Rapid. Altfel, verificarea se poate face din meniul Simulation > I/O Simulator.

6 Realizarea unui obiect de tip gripper static

Pentru a simplifica modul de operare și pentru putea a reutiliza elementul de tip efector terminal proiectat cu tot cu transformarea/transformările TOOL atașate există posibilitatea de a crea o componentă de tip unealtă (Fig.3-16).

Fig. 3-16 – Definirea unui gripper ca și mecanism sau unealtă de tip TOOL

Primul pas în crearea componentei îl reprezintă proiectarea grafică. Acest model grafic trebuie proiectat din originea sistemului de coordonate global, acesta fiind punctul de aplicare ulterioară a uneltei pe flanșa robotului. Procesul de creare are 2 pași: 1) specificare nume unealtă și selecție componentă grafică utilizată (se recomandă folosirea unei componente grafice create anterior care să imite cât mai bine gripperul fizic) (Fig. 3-17) și 2) specificarea transformării/transformărilor până în punctul condus al uneltei (Fig. 3-18). Dacă nu se stiu dimensiuniel gripperului, se pot folosi uneltele grafice (ex.: Snap Object, Snap Center/End, ș.a.) pentru a selecta punctele de interes.

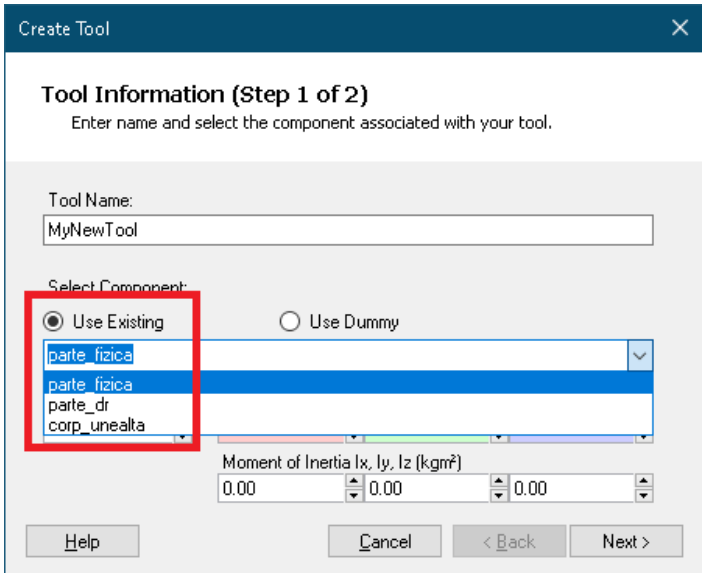


Fig. 3-17 – Definirea suportului grafic al uneltei

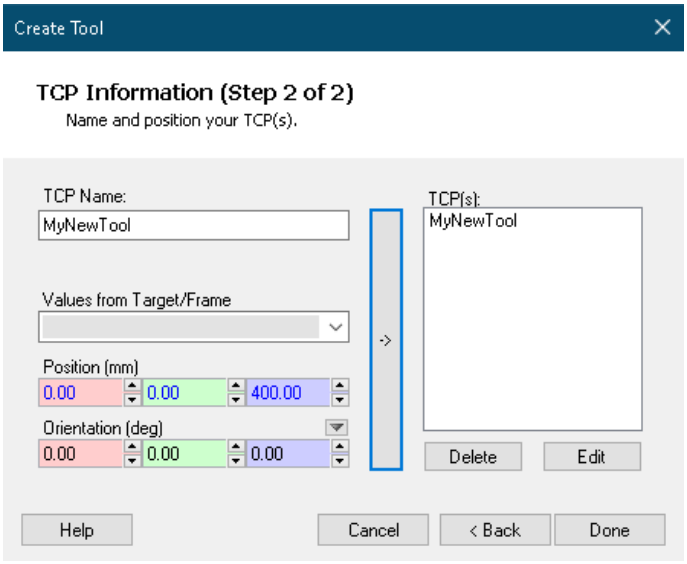


Fig. 3-18 – Definirea transformării/transformărilor TOOL pentru unealta definită anterior

Avantajul lucrului cu componente de tip TOOL este că mecanismul nou creat poate opera în mod unitar cu elementul grafic și cu transformările necesare. Adițional componenta poate fi exportată prin si importată într-un alt proiect conform (Fig. 3-19). În cazul în care nu este permis exportul trebuie deconectată conponenta apoi exportată și importată. Procedura de utilizare a componentei este similară cu procedura de utilizare prezentată mai sus.

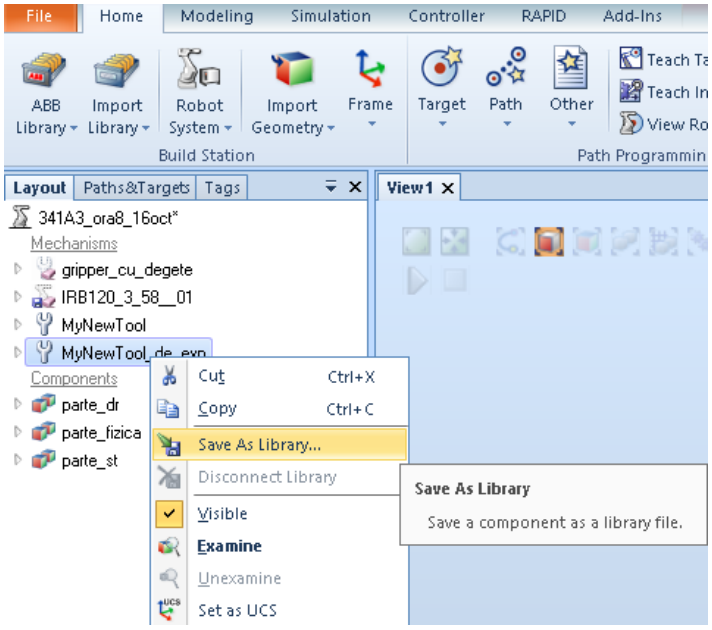


Fig. 3-19 – Modalitatea de exportare, respectiv importare (Import Library/ Home) a unei componente de tip *TOOL*

7 Realizarea unui obiect de tip gripper cu grade de libertate

Pentru realizarea unui gripper care are posibilitatea de mișcare se folosește meniul Create Mechanism > Tool. Crearea mecanismului se face din originea sistemului de coordonate global al proiectului, acesta fiind punctul de montare pe flanșa robotului. Trebuie urmăriți următorii pași de lucru:

Pas 1: creare componente/segmente gripper (în cazul unui gripper cu 2 degete compoentele vor fi baza gripperului și cele 2 degete) și dispunerea lor în postura cere formează mecanismul;

Pas 2: Create Mechanism > Tool și completare câmpuri segmente (links), articulații (Joints) și transformari (Tooldata);

Pas 3 (Segmente componente): se selectează segmentele componente începând cu segmentul 1 reprezentând baza mecanismului (este obligatoriu ca orice mecanism să aibă o bază);

Pas 4 (articulații): în cazul gripperului cu 2 degete acesta are 2 articulații de tip prismatic care mișcă fiecare deget în mod individual;

Pas 5 (transformări *TOOL*): se configurează transformările relativ la segmentul dorit.

Pas 6: compilare mecanism

Pas 7: definire poziții de lucru

Pas 8: din Event Manager se definesc evenimente care mișcă mecanismul dorit la o poziție învățată anterior. Procedura de utilizare a unui gripper inteligent in conjunctie cu componente dinamice (manipulare prin impact) – se adaugă în event manager un eveniment de atașare pe lângă evenimentul de închidere gripper.

8 Procedura de paletizare a unui obiect

Sarcina principală a unui robot industrial este aceea de a automatiza manipularea de obiecte în spațiul de lucru. Procesul de mutare a unui obiect din punctul A în punctul B, prin prinderea sa cu un efector terminal de tip gripper se numește în mod uzual procedura pick-and-place. Dacă este realizată de o manieră generică, procedura poate fi refolosită atunci când trebuie manipulate obiecte între diferite puncte (punctele de prindere – pick – respectiv de depunere – place –, precum și parametrii procedurii fiind ulterior stabiliți. În realizarea unei proceduri de manipulare sunt de interes următoarele elemente:

- Punctele finale ce trebuie deservite
- Punctul de siguranță, respectiv punctul de sincronizare (safe) – toate traiectoriile încep și se termină în punctul safe
- Punctele de trecere care definesc suportul traiectoriei
- Limitarea de viteza (eventual și de accelerație/decelerație) pe traiectorie între punctele suport
- Distanțele de apropiere/depărtare de punctele de interes, respectiv de punctele de trecere
- Tipul mișcării (liniar/neliniar) pe diferitele porțiuni ale traiectoriei
- Acționarea gripperului (putem introduce întârzieri)

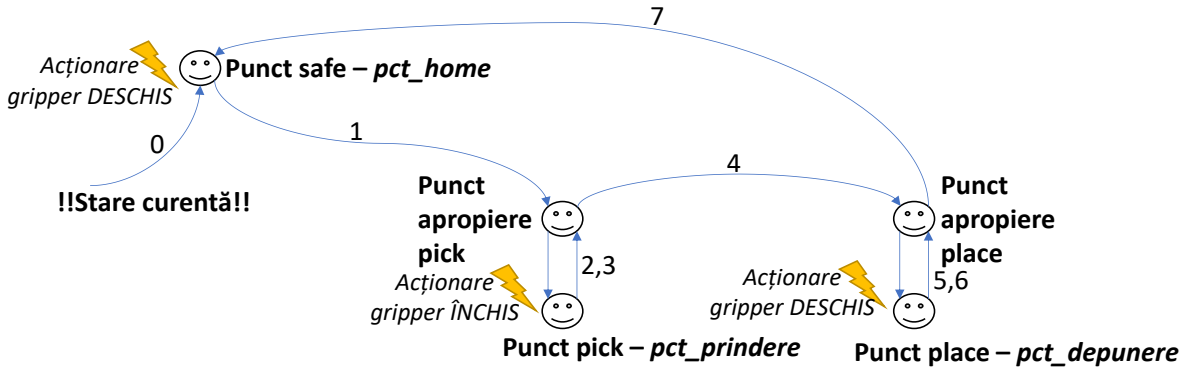


Fig. 3-20 – Traiectoria pick-and-place

Din figura de mai sus rezultă că robotul poate porni dintr-o stare care nu este poziția implicită de siguranță. Această stare este de obicei starea inițială a sistemului. Mișcările 0, 1, 4, ,7 sunt realizate fără nici o constrângere fizică de aceea nu este necesar ca ele să respecte o traiectorie impusă (MOVEJ). Mișcările 2, 3, 5, 6 pot avea constrângeri de traiectorie de aceea este indicat ca ele să fie executate liniar (MOVEL). Se va avea atenție la învățarea respectiv definirea punctelor, acest proces realizându-se cu instrucțiunile *Teach target* și *Target.* Traiectoria teoretică de mai sus este implementată în RobotStudio conform (Fig. 3-21).

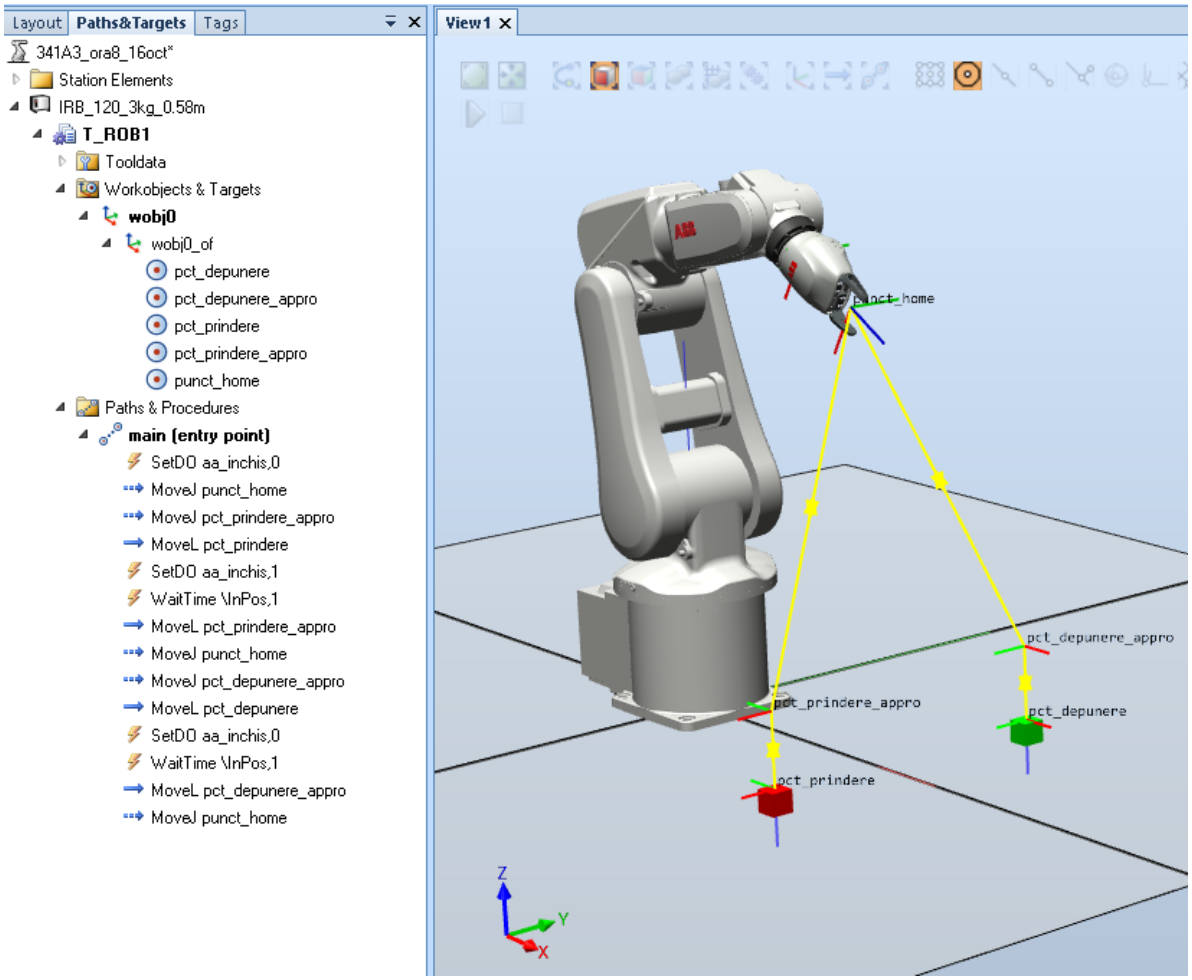


Fig. 3-21 – Implementarea traiectoriei pick_and_place în RobotStudio

Toată traiectoria poate fi implementată direct din interfața grafică prin adaugarea punctelor în traiectoria Main (Fig. 3-22), cu particularitățile fiecărei instrucțiuni de mișcare (liniar in articulații, respectiv liniar în cartezian), respectiv de acționare și temporizare. La sfârșitul realizării traiectoriei trebuie sincronizată simularea cu programul Rapid.

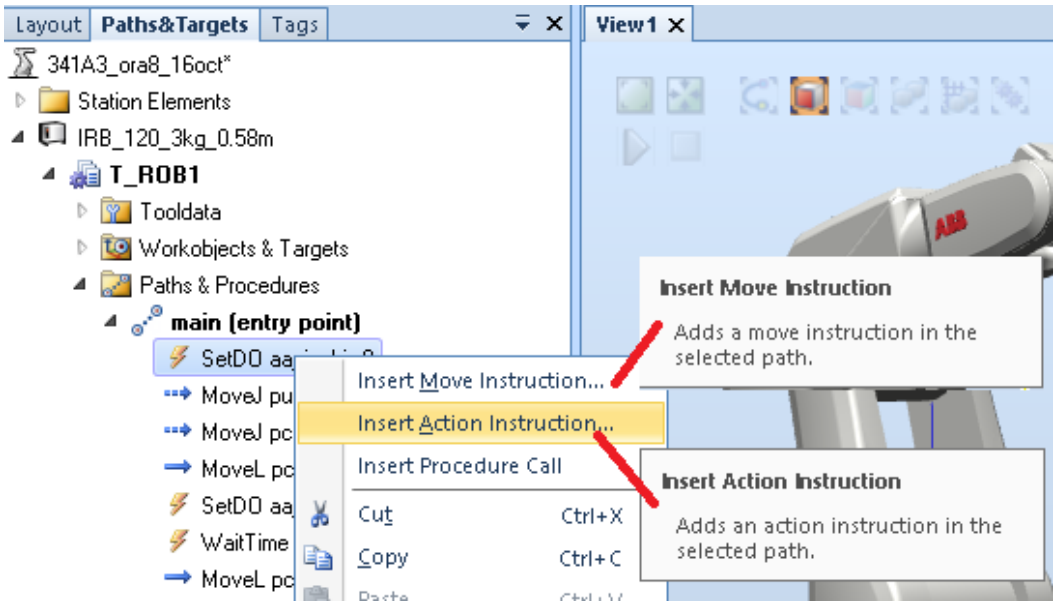


Fig. 3-22 – Realizarea unei traiectorii din interfața grafică

9 Verificare traiectorie, conceptul de configurație robot

Aplicația RobotStudio face în mod automat verificarea traiectoriei. În primă instanță pot apărea erori din cauza faptului că punctele dorite sunt în afara anvelopei de lucru a robotului, au orientări neconforme cu spațiul de lucru cu dexteritate, respectiv configurația robotului în punctul dorit nu este posibilă (Fig. 3-23) (Obs.: dacă nu se face o specificare a configurațiilor în toate punctele de lucru atunci aceasta este pusă în mod automat și există posibilitatea ca ea să nu fie validă). În primele două cazuri corectarea erorii se face aducând punctul în spațiul de lucru respectiv notând configurația robotului și specificând-o la fiecare instrucțiune de mișcare. Într-un caz real (se lucrează direct cu brațul fizic) nu au cum să apară erori de anvelopă din cauza faptului că toate punctele vor fi învățate mișcând brațul, acest lucru asigurând existența soluției de cinematică inversă. Legat de a doua problemă, schimbarea configurației robot, aici trebuie avut grija deoarece această procedură are ca rezultat o mișcare amplă a brațului, mișcare ce poate crea coliziuni. Simularea permite astfel observarea (în mod automat) a mișcărilor neconforme care în anumite cazuri pot genera coliziuni.

Ca și recomandare generală toate instrucțiunile de mișcare ar trebui sa fie liniare în articulații mai puțin acolo unde traiectoria nu impune un alt tip de mișcare.

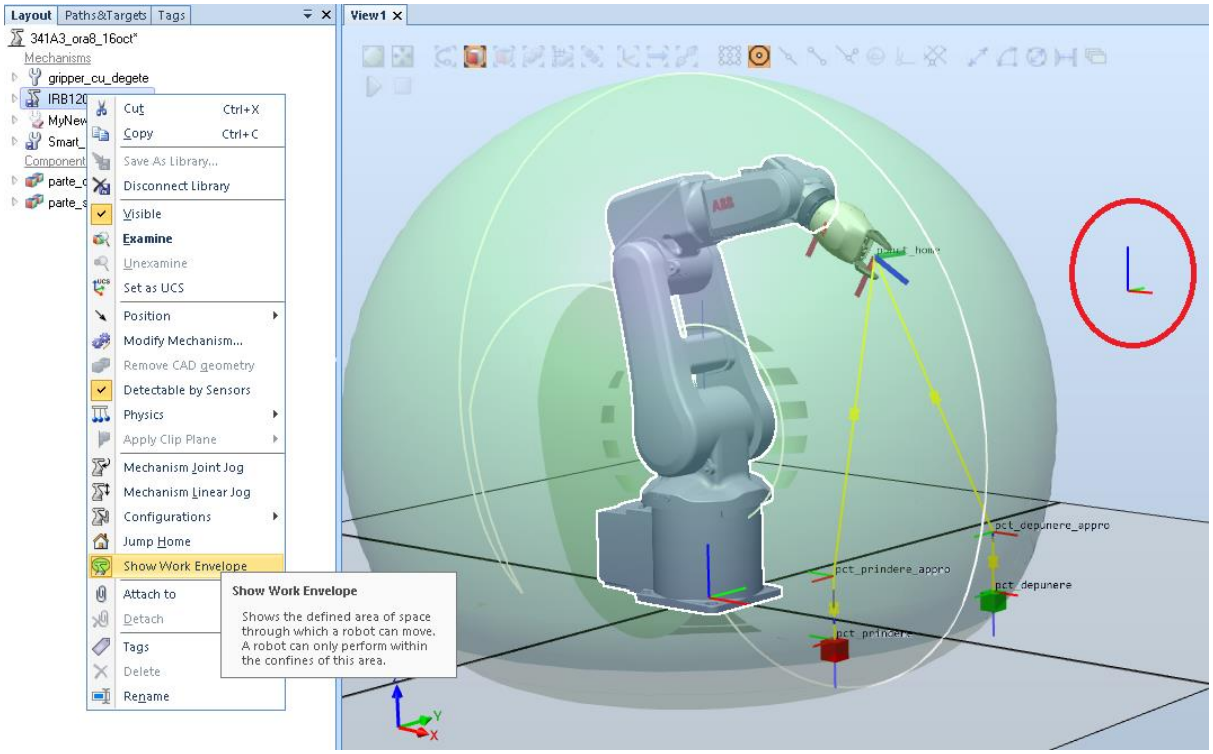


Fig. 3-23 – Detecția unui punct în afara anvelopei de lucru a robotului

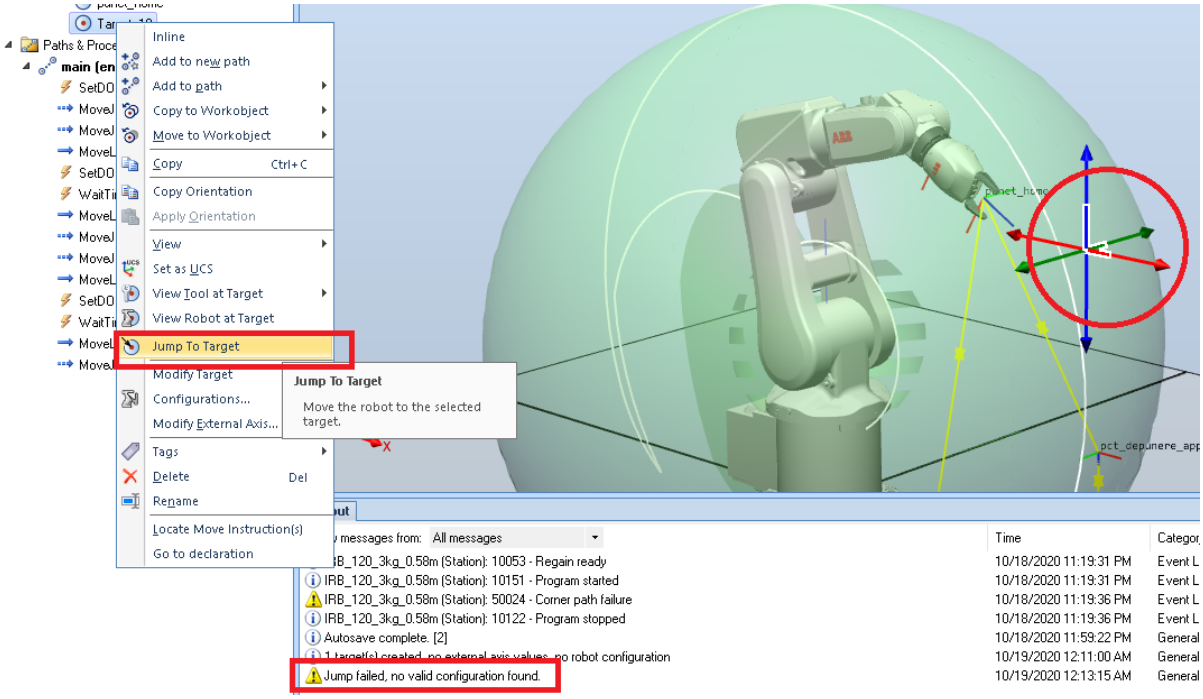


Fig. 3-24 – Detecția unui punct a cărui orientare nu poate fi satisfăcută chiar dacă este în anvelopa de lucru

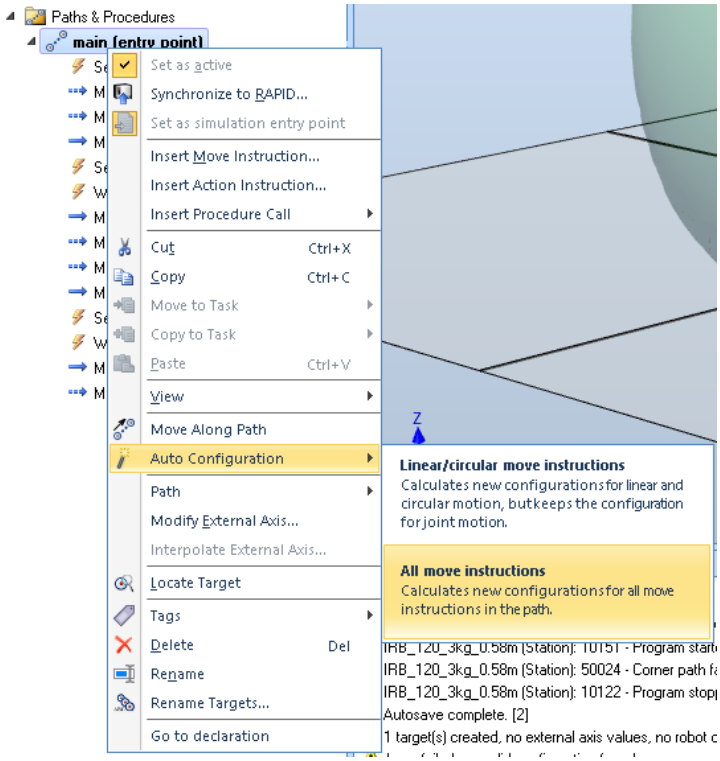


Fig. 3-25 – Realizarea automată a configurațiilor instrucțiunilor de mișcare

10 Crearea unui efector terminal cu mai multe capete

Un efector terminal cu mai multe capete se poate realiza prin adăugarea simultană a 2 efectoare terminale simple. Trebuie avut grijă la montarea acestor efectoare terminale simple

pe flanșa robotului astfel încât să nu se suprapună între ele. Operarea cu fiecare efector în parte (mișcare și atașare obiecte) se face prin specificarea cărei unelte să se folosească în cadrul instrucțiunii de mișcare și atașarea corespunzătoare a obiectelor la efectorul terminal dorit.

11 Crearea unei traiectorii curbilinii

Pentru realizarea unei traiectorii curbilinii se folosește opțiunea AutoPath urmată de selecția muchiilor necesare traiectoriei (Fig. 3-26). Astfel se vor genera un set de puncte suport urmate de instrucțiunile de mișcare aferente (liniar în articulații, în cartezian sau mișcare circulară). În mod implicit există posibilitatea ca punctele suport ale traiectoriei rezultate să nu poată fi atinse de robot, de aceea este necesară ajustarea lor ulterioară.

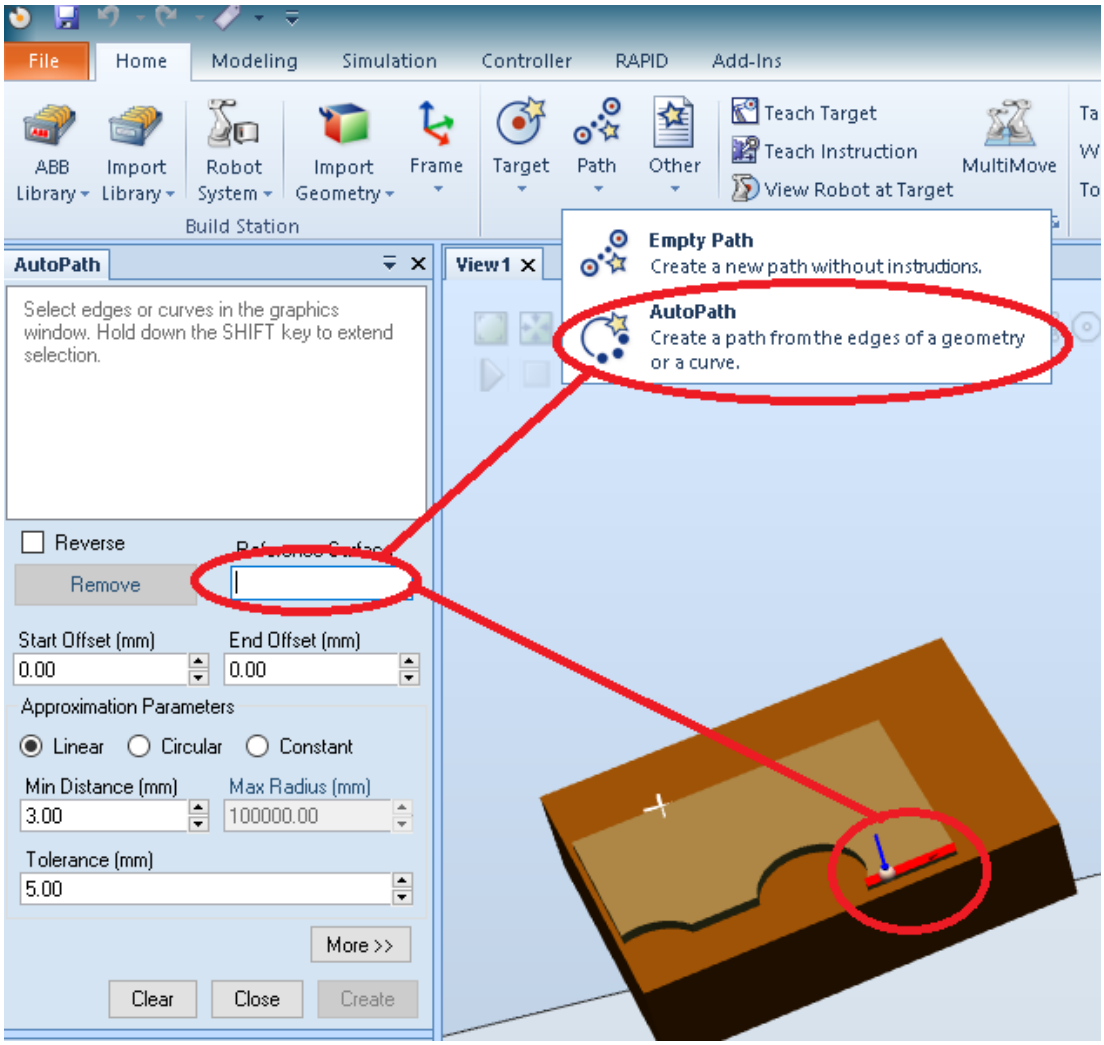


Fig. 3-26 – Realizarea unei traiectorii curbilinii

12 Realizarea de măsurători pe un mediu existent

Pentru a evalua în mod precis distanțele dintre obiectele mediului digital creat există unealta de măsurare punct-la-punct (Fig. 3-27). Astfel, se pot măsura distanțe (direct sau pe axe), precum și înclinări. Pentru o selecție optimă a punctelor între care se realizează măsurarea se recomandă folosirea unor elemente de selecție adiționale precum: selecție suprafață și selecție muchie/centru/margine (Fig. 3-28).

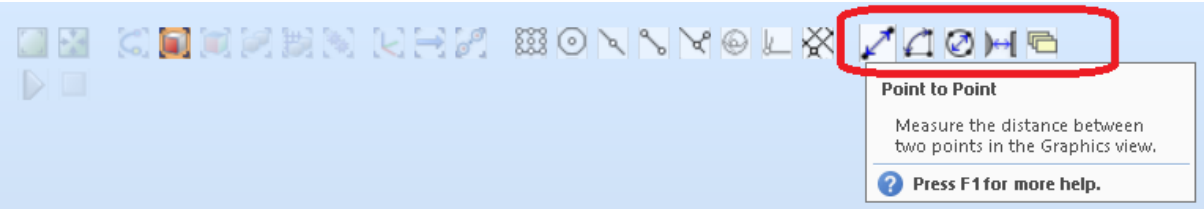


Fig. 3-27 – Accesarea uneltei de măsurat

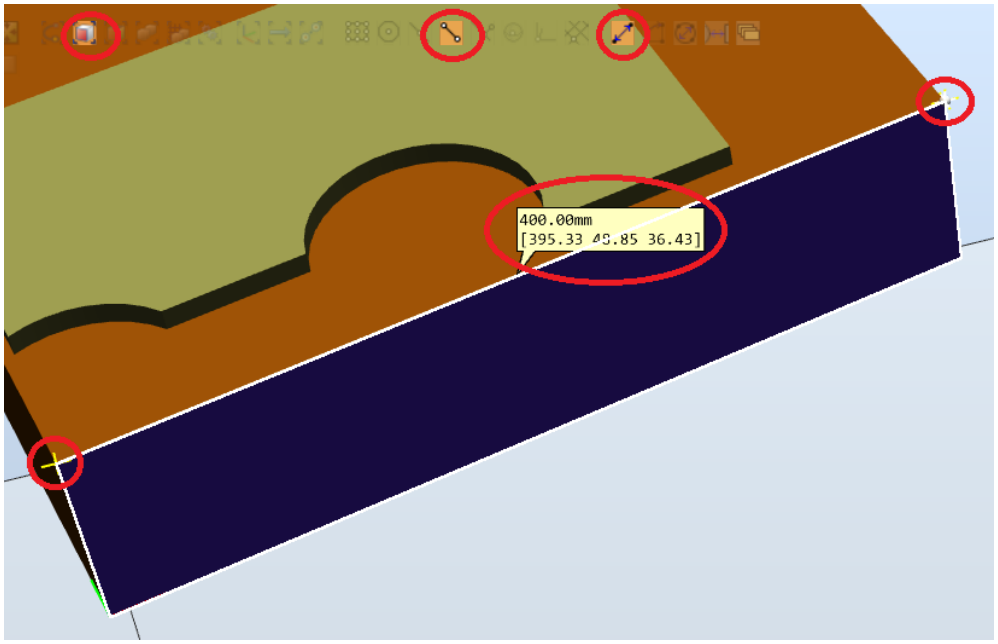


Fig. 3-28 – Realizarea unei măsurători

13 Exerciții

Realizare traiectorie pick-and-place cu manipularea fizică a obiectului.

14 Intrebări

a) Dați un exemplu de proces în care se folosește un efector terminal cu mai multe capete de lucru. Cum se poate realiza un gripper cu mai multe capete de lucru? Pont: se realizează prin definirea mai multor transformări TOOL.

b) Dați un exemplu de punct care este în anvelopa de lucru dar nu poate fi atins. Ce este de făcut în acest caz?

Capitol 4: Principii generale de lucru Limbajul de programare Rapid Programarea în mod text

Cuprins

1 Programarea unui robot industrial 66

1.1 Mediu de dezvoltare pentru robotică - robotică virtuală 66

1.2 Limbaje de programare a roboților 67

1.3 Exemplu de program robot 69

2 Tipuri de date 70

2.1 pos – translație 71

2.2 orient – orientare 71

2.3 confdata – configrație braț robotic 72

2.4 robtarget – poziția în sistemul de coordonate cartezian a punctului condus 72

2.5 robjoint – valoarea articulațiilor pentru care robotul ajunge în punctul condus 74

2.6 wobjdata – poziția și orientarea unui sistem de coordonate relativ la sistemul de coordonate global 74

2.7 ToolData 75

2.8 Zonedata 76

2.9 Speeddata 77

2.10 Alte instrucțiuni 78

3 Instrucțiuni și funcții Rapid 78

3.1 MOVEJ 79

3.2 MOVEL 80

3.3 SEARCHL 82

3.4 Generare de noi puncte in mod automat, funcțiile OffsS, RelTool 83

3.5 WaitDI / WaitDO 84

3.6 SETDO 84

3.7 WaitTime 85

3.8 EulerXYZ & OrientZYX 85

3.9 CPos, CRobT 86

3.10 Distance 86

3.11 CalcJointT & CalcRobT 86
3.12 Instrucțiuni matematice 87
3.13 Instrucțiunea poseMult 87
4 Comparație între limbajele V+ și RAPID în termeni de operații mișcare 88
5 Exemple de programare Rapid (+rulare cu tot cu pointer si rulare inainte si inapoi) 89
5.1 Prezentare module sistem și module program 89
5.2 Moduri de operare 89
5.2.1 Modul automat 90
5.2.2 Modul manual 91
5.3 Crearea unui nou modul program cu tot cu subrutină 91
5.3.1 RobotStudio 91
5.3.2 FlexPendant 92
5.4 Apel subprogram / Lansare in execuție 93
6 Exercitii 94
7 Intrebari 94

1 Programarea unui robot industrial

1.1 Mediu de dezvoltare pentru robotică - robotică virtuală

Viteza mare de procesare și capacitățile mari de stocare ale computerelor moderne permit generarea și utilizarea mediilor virtuale pentru modelarea și simularea proceselor de fabricație. Aceste medii virtuale includ programarea virtuală și operarea mașinilor-unelte și a roboților. Funcționarea roboților virtuali într-un mediu virtual se bazează pe același software și pe aceleași programe ca funcționarea roboților reali într-un mediu real. Programarea diferitelor operațiuni de producție precum sudară cu arc, sudară in puncte, asamblarea, curățarea, pulverizarea, tăierea, turnarea, șlefuirea, lustruirea, deservirea mașinilor, manipularea, vopsirea, ambalarea sau paletizarea pot fi realizate într-un sistem virtual. După transferul programelor din mediul virtual în controllerul real, roboții reali vor opera conform simulării.

Acest mediu virtual de robotică permite dezvoltarea de noi sisteme robotice, selectarea și programarea roboților cu mult înainte ca sistemul real robot să fie dezvoltat, transportat și instalat. Structura programelor și depanarea lor într-un mediu virtual reduce timpul pentru pregătirea sistemului robot și garantează o fiabilitate mai mare a sistemului.

1.2 Limbaje de programare a roboților

Metodele de programare a roboților industriali se împart în două categorii: manuale și automate. Programarea manuală a roboților (Fig. 4-1) poate fi împărțită în programare bazată pe text și metode de programare grafică. Programarea bazată pe text este programarea în limbaje de programare robot care pot fi împărțite în funcție de structura limbajului și comenzile aferente. Programul robot constă în realizarea succesiunii de comenzi de mișcare și sincronizare cu elementele exterioare. Programul robot poate include subrutine, rutine ciclice și ramuri ale secvențelor de comandă. Pentru a crea modele mai bune ale operațiilor tehnologice exeuctate dar și pentru a dispune de metode de programare mai convenabile, în ultimii ani s-a acordat o atenție sporită metodelor de programare grafică. În acest caz, programul este realizat prin diagrame de flux algoritmice, scheme grafice sau scheme funcționale. În multe cazuri, programarea grafică, în comparație cu programarea text, are o curbă mai rapidă de învățare necesitând un timp mai scurt pentru învățare și generarea de programe.

Programarea automată a roboților se bazează pe sisteme de învățare, sisteme demonstrative sau sisteme instructive. În cazul roboților industriali, programarea prin demonstrație, cu ajutorul dispozitivului de comandă (en.: teach pendant - TP, manual control pendant - MCP), este mai frecventă. Traiectoria de referință învățată de operator pentru manipulator, va fi stocată automat în memoria robotului și va fi executată ulterior de controllerul robot.

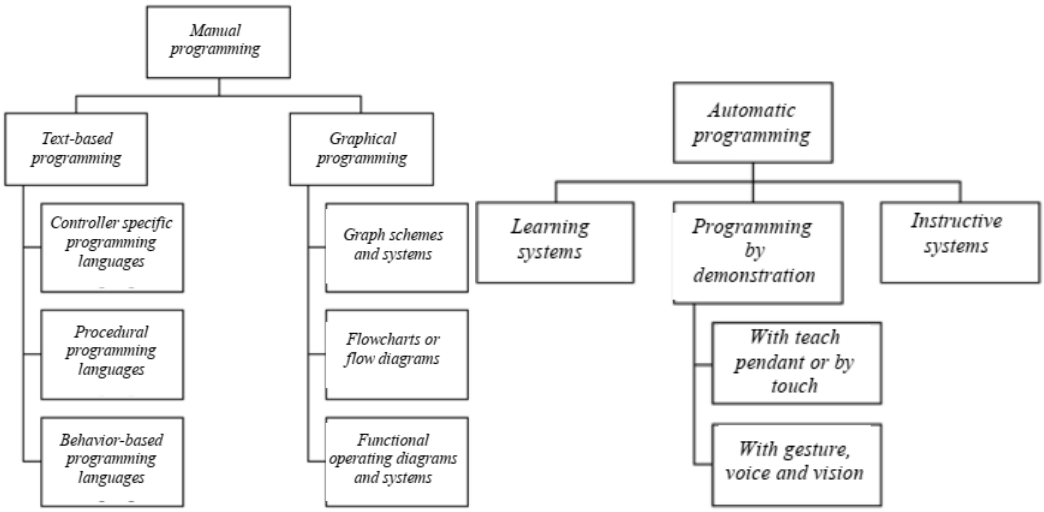


Fig. 4-1 – Metodele manual și automat de programare a roboților industriali

Metoda de lucru în realizarea unui program robot

Planificarea operațiilor este realizată în prima etapă de programare după cum urmează:

- planificarea sarcinii de realizat și determinarea rolului robotului
- descrierea procesului de producție cu diagrame de flux pentru a specifica sarcinile de lucru concrete pentru robot și alte mașini din sistemul robotului, inclusiv:
    - descompunerea procesului de lucru și definirea operațiunilor de lucru detaliate
    - definirea pozițiilor
    - definirea semnalelor de intrare și ieșire

- structura diagramelor de lucru, subrutine, rutine și module de programe pentru un robot

Pentru programarea robotului sunt definite următoarele elemente software:

Modul program

Fiecare modul de program conține date și rutine pentru a realiza o anumită sarcină. Programul este împărțit în module pentru a ușura înțelegera globală a funcționalității și pentru a facilita gestionarea programului. Fiecare modul reprezintă de obicei o acțiune specială a robotului sau una similară. Toate modulele de program vor fi eliminate atunci când se șterge un program din memoria controllerului. Modulele de program sunt scrise de utilizator.

Date program

Datele sunt valori și declarații stabilite în modulele program sau sistem. Datele sunt rafinate prin instrucțiunile din același modul sau în conexe în funcție de vizibilitatea lor.

Rutină

Parte a unui modul, rutina conține un set de instrucțiuni care definesc ceea ce face sistemul robot. O rutină poate conține date locale necesare instrucțiunilor.

Rutină de start

Un tip special de rutină (întâlnită și sub numele de Main) ce reprezintă punctul de start al programului. Numele ei poate fi schimbat, dar ca și concept, fără rutina de start programul robot nu poate porni.

Instrucțiune

Fiecare instrucțiune este o cerere pentru ca un anumit eveniment să aibă loc, de ex. „mișcă punctul condus într-o anumită poziție" sau „Setați o ieșire digitală specifică". Instrucțiunile, sintaxa și funcția acestora sunt descrise în detaliu în manualul tehnic al limbajului de programare (ex.: Technical reference manual, RAPID Instructions, Functions and Data types).

Modul sistem

Fiecare modul de sistem conține date și rutine pentru a îndeplini o anumită funcție. Programul este împărțit în module, în principal pentru a îmbunătăți înțelegerea per ansamblu și pentru a di gestionat mai simplu. Fiecare modul reprezintă o acțiune robot. Toate modulele

sistem vor rămâne după ce este executată o comandă de tipul *Delete program*. Modulele sistem sunt în mod uzual scrise de fabricant sau, în anumite cazuri, de integratorul sistemului.

Fiecare robot este programat într-un limbaj de robot proprietar. Cu toate că sunt diferite ca și sintaxă limbajele de programare robot oferă comenzi sistem similar precum și o sintaxă nu foarte diferită (ex.: pentru mișcarea punctului condus în limbajul V+ avem instrucțiunea MOVE iar în Rapid avem MOVEJ). Astfel, abilitățile obținute într-un limbaj pot fi refolosite pentru a dezvolta traiectorii în alte limbaje de programare a roboților. În mod uzual instrucțiunile/comenzile robot se impart în următoarele categorii:

- Instrucținui pentru controlul mișcării și poziționare – acestea conțin informații despre locația finală în sistem cartezian sau de variabile interne (valori ale articulațiilor), metoda de interpolare, viteză unealtă și intervale de timp
- Instrucțiuni pentru controlul programului – conțin informații despre pornirea și oprirea programului, ramificații, cicluri repetitive, contorizare evenimente și condiții de întrerupere. Aplicațiile robot pot fi rulate în două modalități: un singur ciclu (en.: single cycle) și în buclă (en.: continuous), în producție fiind aleasă a doua variantă.
- Instrucțiuni pentru controlul intrărilor și ieșirilor, aici intrând acționarea efectorului terminal și sincronizarea cu celelalte elemente ale sistemului.
- Instrucțiuni de comunicație pentru transferul de date între robot și celelalte elemente de comandă și control (ex.: automate programabile).
- Alte instrucțiuni folosite pentru gestiunea sistemului (creare fișier tip *log*), setare de parametrii, selecție program, resetare informații eronate.

Diferența între o instrucțiune și comandă: instrucțiunea se dă din program (script) în timp ce comanda se dă din consolă (rând pe rând, secvențial).

1.3 Exemplu de program robot

În figura de mai jos este ilustrată o traiectorie robot (Fig.4-2). Pentru realizarea acestui proces trebuie ținut cont de următoarele: unde să ducem punctul condus, pe ce traiectorie, ce semnale să acționăm/cum ne sincronizăm cu dispozitivul master.

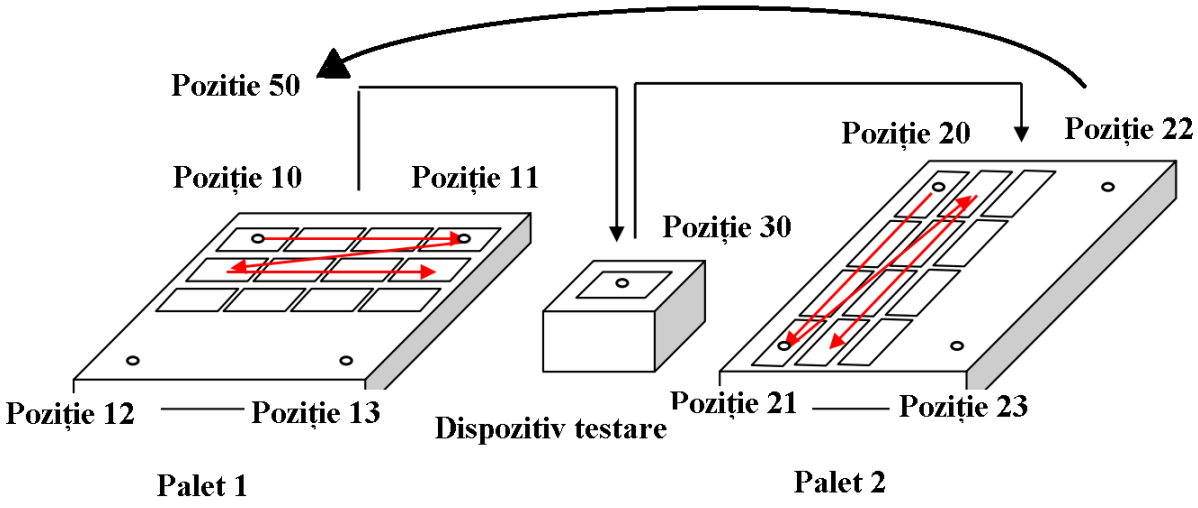


Fig. 4-2 – Exemplu de traiectorie robot

2 Tipuri de date

Limbajul de programare al roboților de la producătorul ABB este Rapid. Acesta este utilizat prin intermediul aplicației grafice RobotStudio care permite atât programarea grafică, ideală în modul de lucru vizual, dar și programarea în mod text recomandată în cazul aplicațiilor reale datorită flexibilității sporite. Atât din mod grafic cât și din mod text rezultă codul Rapid care se execută pe controllerul atașat iar rezultatul poate fi observat in fereastra View a aplicației. Legătura între modulele de mai sus și modul de realizare a sincronizării este reprezentată în fogura de mai jos (Fig. 4-3).

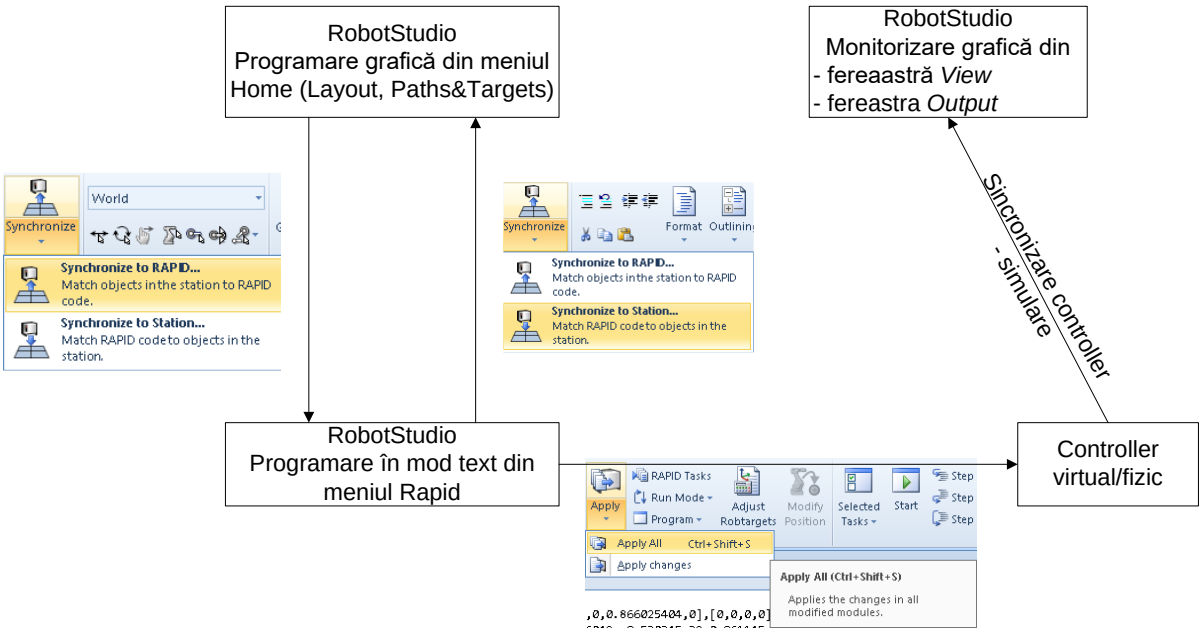


Fig. 4-3 – Sincronizarea schimbărilor între modulele aplicației RobotStudio și controllerul atașat

Procedura de sincronizare interfață grafică – Rapid este următoarea:

a. După realizarea modificărilor grafice se face sincronizarea cu Rapid (Fig. 4-3), urmând apoi modificarea implicită pe controller;
b. Dupa realizarea modificărilor în mod text acestea trebuie verificate (*Check Program*), apoi aplicate pe controller și apoi se face sincronizarea cu interfața grafică (Fig. 4-3). Corectitudinea programului, din punct de vedere al sintaxei, se poate vedea în fereastra *Output*.

ATENȚIE: nu toate instrucțiunile definite în meniul Rapid se vor regăsi în Paths & Targets Paths Procedures procedură specifică. Un exemplu sunt instrucțiunile de ciclare (FOR) .În acest sens se recomandă evitarea utilizării combinate a programării grafice cu cea de tip text.

c. Actualizarea datelor adăugate în mod grafic, respectiv text se face doar dacă acestea intră în cadrul unei traiectorii, acestea din urmă sincronizându-se în mod automat.

În cadrul acestui material se va pune accent pe elementele necesare realizării unei traiectorii robot în modul text folosind dispozitivul de mișcare manuală (FlexPendant) pentru învățarea punctelor și lansarea în execuție a programelor. O descriere completă a funcționalităților controllerului în termeni de date, instrucțiuni și procese de lucru se regasesc

în manualele tehnice de referință ale Rapid (Technical reference manual, RAPID Instructions, Functions and Data types; Technical reference manual, RAPID overview;). Pentru simplitate acest material va puncta doar acele tipuri de date și instrucțiuni care sunt necesare programelor prezentate.

Un program Rapid este compus din instrucțiuni și date. Ca orice limbaj de programare, Rapid conține un set de instrucțiuni standard de condiționare a execuției (ex.: FOR, IF, CASE) precum și un set de instrucțiuni speciale, dedicate mișcării articulațiilor (ex.: MOVEJ) și lucrului cu intrările și ieșirile digitale (ex.: SetDO). În continuare, pe baza particularităților prezentate anterior vom trece în revistă principalele instrucțiuni de mișcare și lucrul cu intrările și ieșirile, respectiv principalele tipuri de date cu care se va opera în realizarea unor traiectorii de complexitate scăzută.

Există 3 tipuri de date:

- Date de tip variabila (VAR) – pot primi o nouă valoare în timpul executării programului;
- Date de tip persistent (PERS) – care la resetarea programului reflectă ultima valoare primită;
- Date de tip constant (CONST) – au o valoare statică și nu pot fi schimbate (punctele amplasament sunt declarate ca și constante pentru că nu are sens să fie schimbate în program).

Structura de date particulară unui sistem de coordonate robot este sistemul de coordonate. Acesta poate fi fix sau mobil și este definit prin 6 componente: 3 poziții și 3 rotații. Structurile de date care se folosesc în operarea cu sisteme de coordonate sunt:

2.1 pos – translație

Descrie poziția în termeni de coordonate pe X, Y și Z. Observație: a nu se confunda cu *pose*, structură de date ce conține și orientarea punctului.

Exemplu de utilizare

```
VAR pos pos1;
...
pos1 := [500, 0, 940];
```

coordonată Z
coordonată Y
coordonată X

Fig. 4-4 – Structura de date pentru memorarea poziției unui sistem de coordonate

2.2 orient – orientare

Este folosit pentru a exprima orienări ale uneltei și rotații ale sistemelor de coordonate. Are 4 componente denumite cuaternioni care descriu rotația a două sisteme de coordonate rotite în spațiu. Pentru conversia grade (ZYX intrinsec) cuaternioni Rapid are la dispoziție 2 funcții (OrientZYX și EulerZYX) care vor fi descrise în secțiunea următoare.

Exemplu de utilizare

```
VAR orient orient1;
.
orient1 := [1, 0, 0, 0];
           q1  q2 q3  q4
```

Fig. 4-5 – Structura de date care memorează orientarea unui sistem de coordonate

2.3 confdata – configrație braț robotic

Structura este folosită pentru definirea configurației axelor robotului. De obicei este posibil să se obțină aceeași poziție și orientare a efectorului terminal al robotului în mai multe modalități, folosind diferite seturi de unghiuri ale axelor de mișcare. Numim aceste elemente configurații robot diferite. Modul de specificare a configurației robotului diferă pentru diferite tipuri de roboți. Cu toate acestea, pentru majoritatea tipurilor de roboți, aceasta include definirea locației de rotație aferente axelor 1, 4 și 6. De exemplu, dacă axa 1 este între 0 și 90 de grade, atunci cf1 = 0, vezi figura de mai jos

Exemplu de utilizare

```
confdata cf1, cf4, cf6, cfx
                        număr care specifică una din cele 8 configurații posibile 0...7
                  cuadrantul axei 6
            cuadrantul axei 4
      cuadrantul axei 1
```

Fig. 4-6 – Structura de date care memorează configurația axelor unui robot

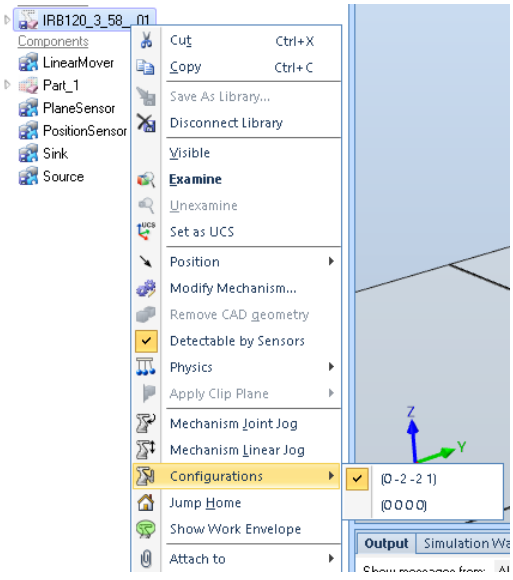


Fig. 4-7 – Modalitatea de alegere a configurației robot

2.4 robtarget – poziția în sistemul de coordonate cartezian a punctului condus

Variabilele de tipul robtarget sunt folosite pentru a defini poziția robotului cu orientarea dorită. Acest tip de date vor fi folosite în instrucțiunile de mișcare a robotului. Robotul poate

atinge poziția dorită cu efectorul terminal aplicat în mai multe moduri, de aceea configurația axelor este specificată.

Componente robtarget:

- trans- translation: poziția punctului memorat pe axa x,y și z exprimată în mm;
- rot- rotation: exprima orientarea uneltei sub forma unei structuri de quaternioni (q1, q2, q3, q4);
- robconf- configurația axelor robotului (cf1, cf4, cf6, cfx). Dacă punctul este definit din interfață grafică atunci configurația implicită atașată este (0,0,0,0). Această configurație trebuie testată pentru a fi validată. În cazul învățării punctului cu comanda *Teach Target* va fi memorată configurația robotului din punctul respectiv;
- extax - external axes: poziția axelor externe (ex.: dacă robotul lucrează pe un pod rulant spațiul de lucru este extins, memorându-se și valoarea exei de mișcare aferente axei liniare/de rotație)

Pe lânga modalitatea grafică de definire a unui punct robot există și o modalitate manuală de definire. Această modalitate este utilizată atunci când se operează cu un echipament real.

Pas 1: Mișcare robtot din FlexPendant, meniu Jogging (Fig. 4-7)

Pas 2: Program data -> robjoint -> Show Data -> New (echipamentul trebuie pus în modul manual). Tot de aici, selectând punctul dorit se poate altera valoarea lui.

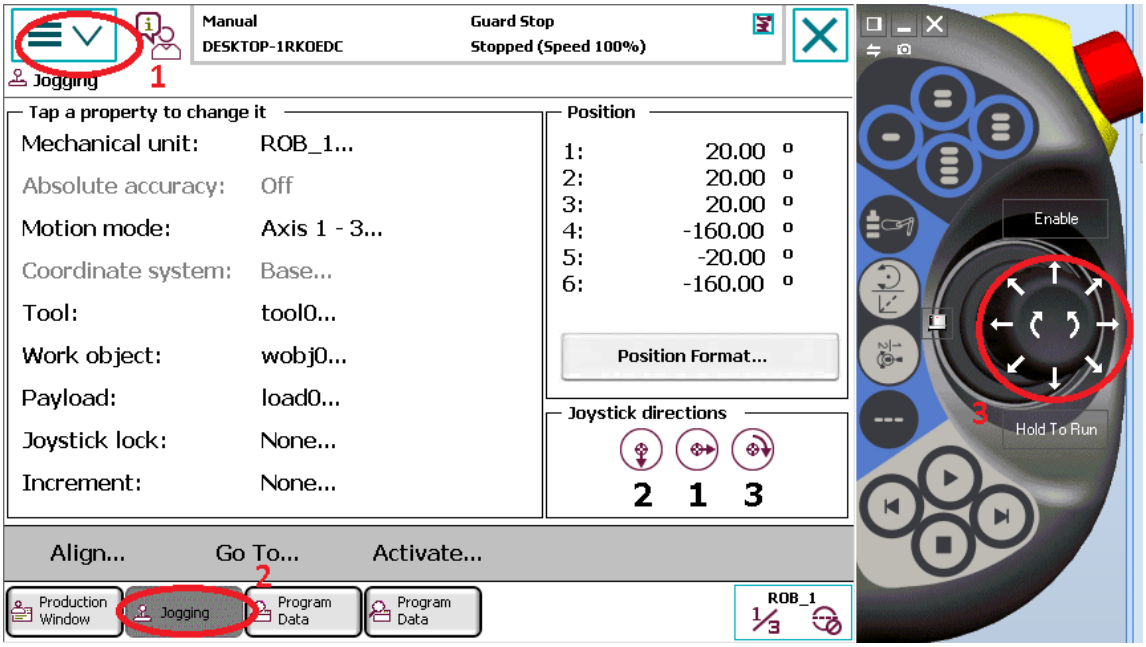


Fig. 4-7 – Utilizarea meniului Jogging (FlexPendant) pentru mișcarea robotului

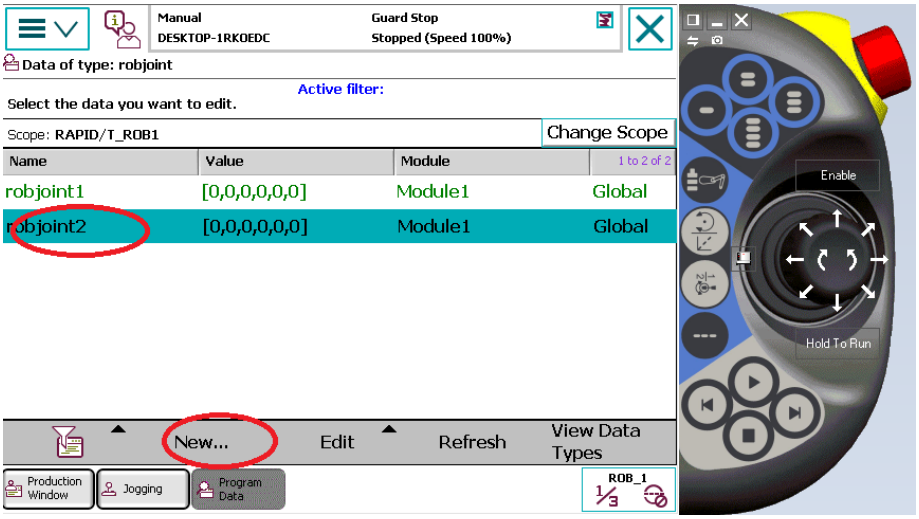


Fig. 4-8 – Utilizarea meniului Program Data (FlexPendant) pentru memorarea unui nou punct

Exemplu de utilizare

poziție orientare

```
CONST robtarget p15 := [ [600, 500, 225.3], [1, 0, 0, 0], [1, 1,
    0, 0], [ 11, 12.3, 9E9, 9E9, 9E9, 9E9] ];
```

axe externe

configurație braț

Fig. 4-9 – Structura de date folosită pentru memorarea unui punct robot

2.5 robjoint – valoarea articulațiilor pentru care robotul ajunge în punctul condus

Variabila robjoint definește poziția robotului folosind valorile axelor de mișcare. La modul general un braț robotic are 6 componente de tip axă de mișcare. Pentru un robot articulat vertical (6 axe de mișcare de tip rotativ) aceast tip de variabilă folosește de toate cele 6 componente. Variabilele robotjoint sunt folosite pentru a stoca poziția axelor(de la 1 la 6) în grade. Poziția axelor este definită ca rotația în grade(pozitivă sau negativă) pentru axele respective raportată la poziția de calibrare a axelor. Modificarea și alterarea unui punct de tip *robjoint* (punct de precizie) se face similar cu punctele de tip *robtarget*. Legătura între punctele robjoint și robtarget se face folosind cinematica robotului (vezi funcțiile CalcRobT și CalcJointT).

2.6 wobjdata – poziția și orientarea unui sistem de coordonate relativ la sistemul de coordonate global

Variabilele de tipul wobjdata descriu sistemele de coordonate cu care robotul operează. Aceste sisteme de coordonate pot fi fixe sau mobile. În cadrul unui astfel de sistem se face definirea de puncte carteziene. Toate datele de tip *wobjdata* se memorează în modulul *CalibData*.

Componente:

- robhold (*robot hold*): definește dacă robotul manipulează sau nu respectivul sistem de coordonate, este de tip boolean (true – este atașat);

- ufprag (*user frame programmed*): definește dacă sistemul de coordonate este fix sau mobil, este de tip boolean (true – este fix);
- ufmec (*user frame mechanical unit)*: numele unității mecanice cu care se face coordonarea robotului, este de tip string;
- uframe (*user frame*): sistemul de coordonate definit de utilizator, este de tip pose;
- oframe(*object frame*): sistemul de coordonate în care este definit, este de tip pose.

Un proiect Rapid are întotdeauna minim un sistem de coordonate (wobj0, acesta coincide cu sistemul de coordonate din baza robotului). Se pot defini sisteme de coordonate adiționale pentru o mai bună referențiere a punctelor în spațiul de lucru (Fig. 4-10). Dacă avem un obiect situat pe o masă și stim pozițiile relative ale obiectului respectiv ale mesei, atunci exprimarea poziției lor se recomandă să se facă folosind compuneri de sisteme de coordonate (transformări spațiale). Avantajul definirii de sisteme de coorodonate este că acesta poate fi atașat unui obiect (ex.: masa pe care se află obiectul) și în momentul mișcării mesei se mișcă și punctul atașat obiectului, nemaifiind nevoie să învățăm un nou punct).

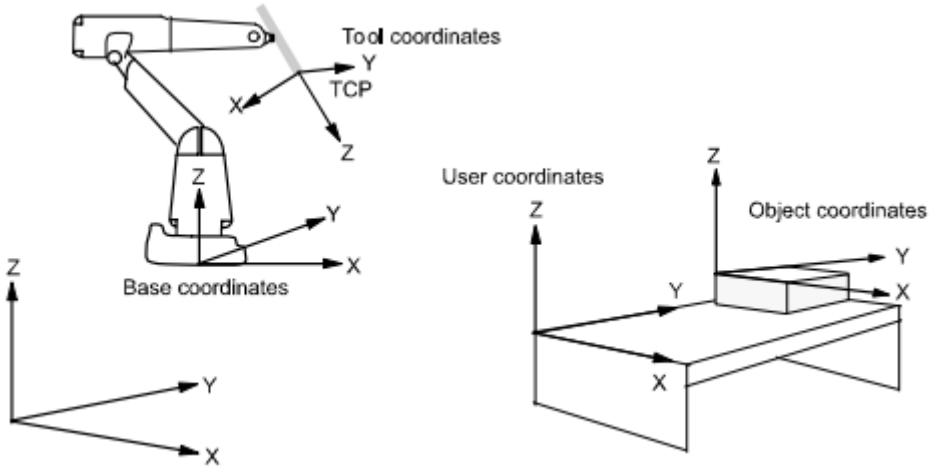


Fig. 4-10 – Definirea coordonatelor unui punct în sistemul de coordonate definit de utilizator

Exemplu de utilizare

Fig.4-11 – Structura de date folosită pentru memorarea unui punct sistem de coordonate

2.7 ToolData

Variabilele ToolData sunt folosite pentru a descrie transformarea care duce din centrul flanșei robotului în punctul condus dorit (gripper, echipament de sudură, etc) (Fig. 4-12). Unealta este descrisă atât în termeni geometrici cât și în termeni fizici (greutate, distribuție a masei) pentru a putea optimiza mișcarea robotului.

Componente:

- robhold: definește dacă robotul folosește sau nu unealta respectivă, este de tip boolean. În majoritatea cazurilor robotul manipulează unealta (robhold =true);
- tframe: sistemul de coordonate al uneltei și anume poziția punctului condus - TCP (distanța pe axele x, y și z în mm raportat la sistemul de coordonate al încheieturii) și orientarea sistemului de coordonate al uneltei (quarternioni q1, q2, q3, q4 raportat la sistemul de coordonate al încheieturii);
- tload :date fizice ale uneltei (greutatea, centrul de masă, momentul de inerție).

Exemplu de utilizare

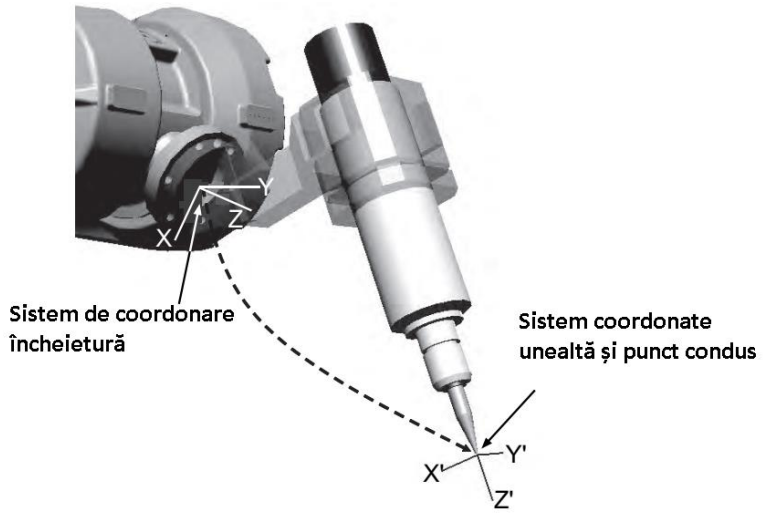


Fig. 4-12 – Transformarea sistemului de coordonate condus (TOOL)

Exemplu de utilizare

transformarea din punctul de aplicare

robotul ține unealta

```
PERS tooldata gripper := [ TRUE, [[97.4, 0, 223.1], [0.924, 0,
     0.383 ,0]], [5, [23, 0, 75], [1, 0, 0, 0], 0, 0, 0]];
```

masa în Kg

centrul de masă

Fig. 4-13 – Structura de date folosită pentru definirea unei transformări TOOL

2.8 Zonedata

Este folosit pentru a specifica poziția finală, cât de aproape de poziția specificată se consideră operațiunea încheiată pentru a trece la următoarea instrucțiune. Avem două tipuri de puncte: Stop point (punct de stop) și fly-by point(punct intermediar) (Fig. 4-14, Fig. 4-15). Punctul de stop este predefinit ca „fine”. Un punct de tip fly-by point este un punct care nu va fi niciodată atins deoarece direcția mișcării se va modifica înainte de a ajunge în acel punct.

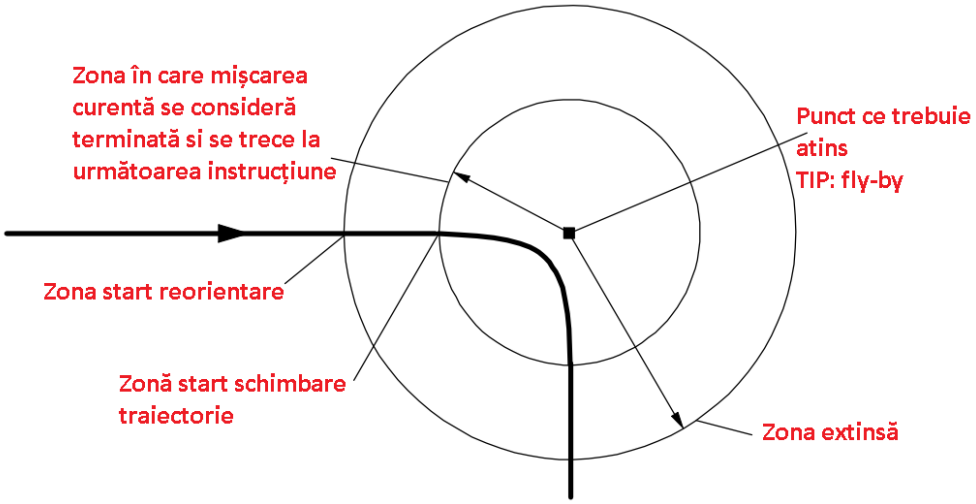


Fig. 4-14 – Modul de definire a unui punct de trecere (*en.: fly-by point*)

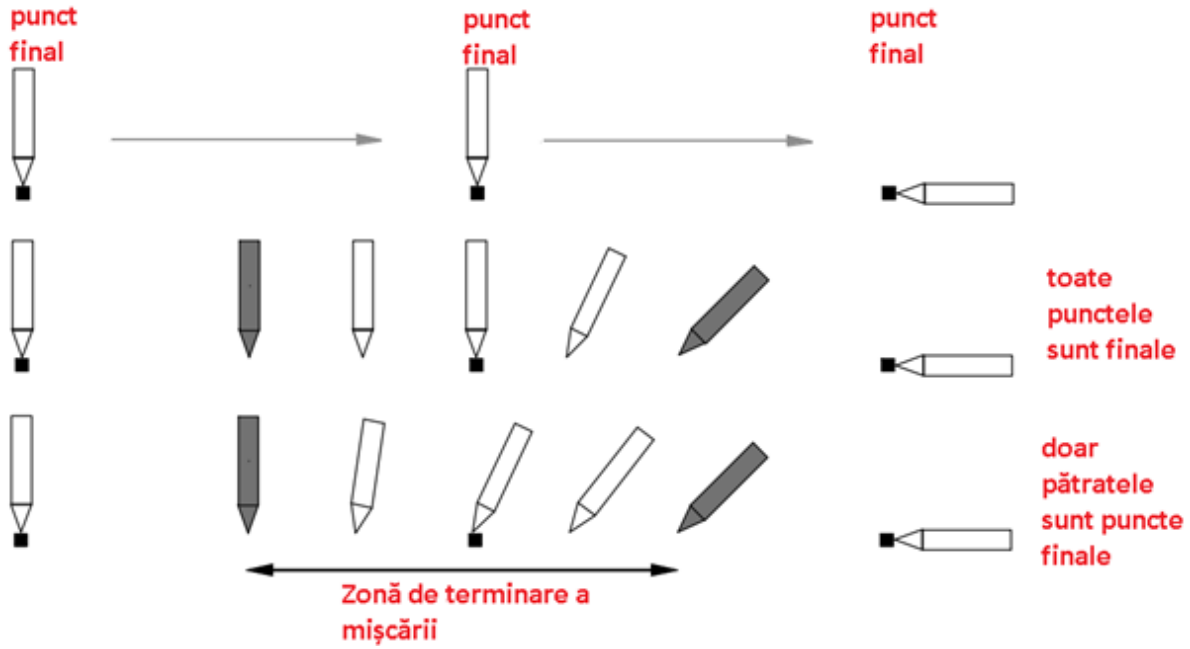


Fig. 4-15 – Modul în care sunt alterate pozițiile și orientările atunci când se operează cu puncte finale, respective cu puncte de trecere

2.9 Speeddata

Specifică viteza limită (în mm/sec) cu care se va mișca punctul condus al robotului.

Exemplu de utilizare

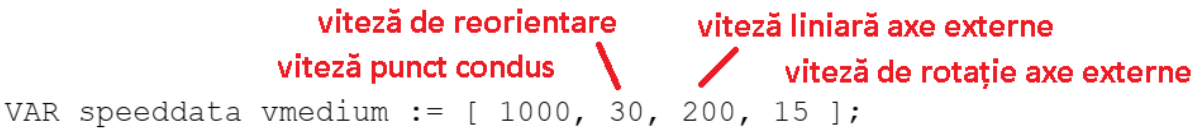


Fig. 4-16 – Structura de date pentru memorarea vitezei diferitelor axe

Un număr de date despre viteză sunt deja definite în sistem. Acestea sunt stocate în constante cu numele v*viteză* unde *viteză* are valori întregi de la 5 la 7000. Această viteză se aplică vitezei liniare a punctului condus, celelate viteze fiind constante: reorientarea se face cu 500°/sec, mișcarea axelor externe se face cu 5000mm/sec și reorientarea axelor externe se face cu 1000°/sec.

2.10 Alte instrucțiuni

Atribuirea se face cu := și comentariile se fac cu ! .

3 Instrucțiuni și funcții Rapid

Principala funcție a controllerului robot este aceea de a coordona modul de mișcare a brațului manipulator în diferite modalități. Mișcările robotului sunt definite ca mișcări de poziționare după principiul „mergi de la poziția actuală la o poziție nouă". Calea dintre aceste două poziții este apoi calculată automat de controllerul robot. Caracteristicile de bază ale mișcării, cum ar fi tipul mișcării punctului condus, sunt specificate prin alegerea instrucțiuniunilor de poziționare adecvate. Caracteristicile de mișcare rămase sunt specificate prin definirea datelor care sunt argumente ale instrucțiunii:

- Date de poziție (poziția finală pentru robot și axe suplimentare)
- Date despre viteză (viteza limită dorită)
- Date despre zona de apropiere (precizia poziției)
- Date unealtă manipulată (poziția punctului condus)
- Date sisteme de coordonate folosite (poziția sistemului de coordonate curent relativ la sistemul de coordonate al bazei)

Atât robotul, cât și axele externe de mișcare sunt poziționate folosind aceleași instrucțiuni. Axele externe sunt deplasate cu o viteză constantă, ajungând în poziția finală în același timp cu robotul. În timpul executării programului, instrucțiunile de poziționare din programul robotului controlează toate mișcările. Sarcina principală a instrucțiunilor de poziționare este de a furniza următoarele informații despre cum să se efectueze mișcarea:

- Punctul final al mișcării (definit ca poziția punctului condus constând în poziția și orientarea efectorului terminal, configurația axelor de mișcare ale robotului și poziția axelor de mișcare externe);
- Metoda de interpolare utilizată pentru a ajunge la punctul de destinație, de exemplu interpolare în spațiul variabilelor interne, interpolare liniară sau interpolare de cerc;
- Viteza limită a robotului și a axelor externe;
- Datele zonei de terminare a mișcării (definesc modul în care robotul și axele suplimentare trebuie să treacă de punctul de destinație, de exemplu punctul final (en.: stop point) sau de trecere/intermediar (en.: fly-by point)).
- Sistemul de coordonate (unealtă, utilizator și obiect) utilizat pentru mișcare.

În cele ce urmează sunt prezentate principalele instrucțiuni și funcții Rapid necesare realizării unor traiectorii de complexitate scazută.

3.1 MOVEJ

Instrucțiunea este folosită pentru a muta robotul dintr-un punct în altul atunci când mișcarea nu trebuie să fie în linie dreaptă. Robotul și axele externe se deplasează în poziția de destinație de-a lungul unei căi neliniare. Toate axele ating poziția de destinație în același timp. Când acuratețea traseului nu este importantă, acest tip de mișcare este folosit pentru a muta efectorul terminal dintr-o poziție în alta. Interpolarea în articulații permite, de asemenea, comanda de axe externe într-o singură mișcare. Toate axele se deplasează de la punctul de pornire la punctul de destinație la viteza constantă a axei. Mișcarea liniară în articulații este adesea cel mai rapid mod de deplasare între două puncte, deoarece axele robotului urmează cea mai apropiată cale între punctul de pornire și punctul de destinație (din perspectiva unghiurilor axei) (Fig. 4-17).

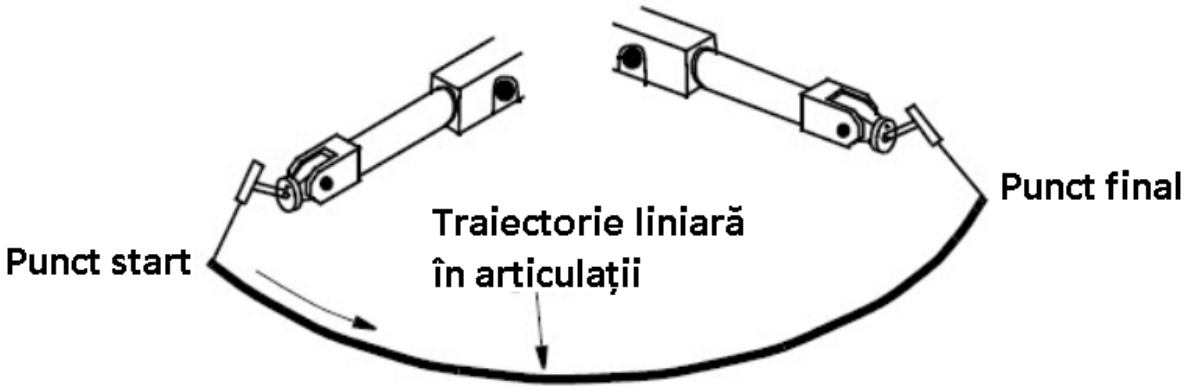


Fig. 4-17 – Mișcare liniară în articulații

Structura instrucțiunii de mișcare MOVEJ

MoveJ [\Conc] , ToPoint , [\ID] , Speed , [\V] | [\T] , Zone , [\Z] ,[\Inpos] , Tool , [\WObj];

Unde :

- [\Conc] Concurrent -este folosit pentru a executa instrucțiunile următoare în timp ce robotul se află în mișcare. În general nu se folosește;
- ToPoint de tip robtarget -punctul de destinație al robotului;
- [\ID] Synchronization id -este folosit la sincronizarea mișcării prin grupare;
- Speed -viteza cu care se va executa mișcarea în procente;
- [\V]- velocity este folosit pentru a seta viteza în mm/s;
- [\T]- time -timpul de mișcare al robotului;
- Zone - reprezintă zona unde se poate mișca robotul;
- [\Z] zone – acuratețea cu care respectă atingerea punctului ToPoint; se specifică în mm;
- [\Inpos] In position – poziția de oprire;
- Tool -se specifică unealta folosită pentru mișcare;
- [\WObj] -sistemul de coordonate care este folosit ca referință în instrucțiune.

3.2 MOVEL

Instrucțiunea este folosită pentru a mișca punctul condus liniar către punctul destinație (Fig. 4-18). Daca punctul condus rămâne constant atunci instrucțiunea este folosită pentru a reorienta robotul. Pentru a obține o traiectorie liniară în sistemul de coordonate al operatorului , axele robotului trebuie să urmeze o cale neliniară în spațiul articulațiilor. Orientarea efectorului terminal rămâne constantă pe parcursul întregii mișcări, cu excepția cazului în care a fost programată o reorientare. Dacă efectorul terminal este reorientat, acesta este rotit cu viteză constantă.

Fig. 4-18 – Mișcare liniară în spațiul cartezian

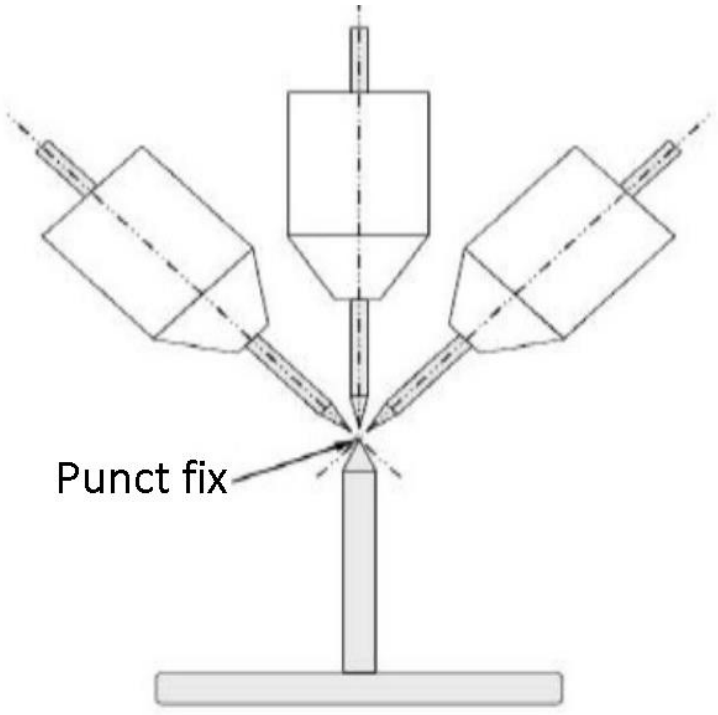


Fig. 4-19 – Reorientare efector terminal

Structură: MoveL [\Conc] , ToPoint , [\ID] , Speed , [\V] | [\T] , Zone , [\Z] , [\Inpos] , Tool , [\WObj] , [\Corr];

Unde:

- [\Conc] Concurrent -este folosit pentru a executa instrucțiunile următoare în timp ce robotul se află în mișcare. În general nu se folosește;
- ToPoint de tip robtarget -punctul de destinație al robotului;

- [\ID] Synchronization id -este folosit la sincronizarea mișcării prin grupare;
- Speed -viteza cu care se va executa mișcarea în procente;
- [\V]- velocity este folosit pentru a seta viteza în mm/s;
- [\T]- time -timpul de mișcare al robotului;
- Zone reprezintă zona unde se poate mișca robotul;
- [\Z] zone – acuratețea cu care respecta atingerea punctului ToPoint; se specifică în mm;
- [\Inpos] In position – poziția de oprire;
- Tool -se specifică unealta folosită pentru mișcare;
- [\WObj] -sistemul de coordonate care este folosit ca referință în instrucțiune;
- [\Corr] -corecția.

ToPoint

Tip de date robtarget. Punctul destinație al robotului și axele externe. Este definit ca o poziție anume sau stocată direct in instrucțiune (marcat cu un * în instrucțiune).

Speed

Tip de date speeddataDatele de viteză care se aplică mișcărilor. Datele de viteză definesc viteza punctului central al instrumentului, reorientarea sculei și axele externe.

[\V] (Velocity)

Tip de date numeric. Acest argument este utilizat pentru a specifica viteza unui TCP în mm/s direct în instrucțiune. Înlocuiește apoi viteza corespunzătoare specificată în datele de viteză.

[\T] (Timp)

Tip de date numeric. Este folosit pentru a specifica timpul total în secunde în care robotul se mișcă. Înlocuiește apoi datele de viteză corespunzătoare.

Zone

Tip de date numeric. Este folosit pentru a specifica acuratețea poziției pentru punctul central al uneltei robotului direct în instrucțiune. Lungimea traseului de colț este dată în mm, care înlocuiește zona corespunzătoare specificată în datele zonei.

[\Inpos] (In position)

Tip de date stoppointdata. Acest argument se folosește în a specifica criteriile de convergență pentru poziția TCP a robotului în punctul de oprire. Datele de punct de oprire înlocuiesc zona specificată în parametrul Zone.

Tool

Tip de date tooldata. Scula folosită când robotul se mișcă. Punctul central al uneltei (TCP) este punctul mișcat la punctul destinație specificat.

[\WObj] (Work Object)

Tip de date wobjdata. Obiectul de lucru (sistemul de coordonate) de care este legată poziția robotului în instrucțiune. Acest argument poate fi omis și, dacă da, atunci poziția este legată de sistemul de coordonate mondial. Dacă, pe de altă parte, se utilizează un TCP staționar sau axe externe coordonate, atunci acest argument trebuie specificat.

[\TLoad] (Total load)

Tip de date tooldata. Argumentul \TLoad descrie încărcătura totală utilizată în mișcare. Încărcătura totală reprezintă încărcătura sculei împreună cu încărcătura pe care o cară. Dacă se folosește argumentul \TLoad, atunci loaddata în tooldata curentă nu este luată în considerare. Dacă argumentul \TLoad este setat la load0, atunci argumentul \TLoad nu este luat în considerare și în schimb se folosește loaddata în tooldata curentă. Pentru a putea utiliza argumentul \TLoad este necesar să se seteze valoarea parametrului de sistem ModalPayLoadMode la 0. Dacă ModalPayLoadMode este setat la 0, nu mai este posibilă utilizarea instrucțiunii GripLoad. Sarcina totală poate fi identificată cu rutina de serviciu LoadIdentify. Dacă parametrul de sistem ModalPayLoadMode este setat la 0, operatorul are posibilitatea de a copia loaddata de la instrument la o variabilă persistentă loaddata existente sau noi atunci când execută rutina de serviciu.

3.3 SEARCHL

Această instrucțiune este folosită pentru a căuta o poziție atunci când TCP-ul robotului (Tool Center Point) se miscă linear. In timpul miâcării, robotul citește continuu un semnal de intrare digital și la modificarea acestuia din 0 in 1 efectueaază o citire a poziției curente.

Aceasta instrucțiune poate fi utilizată în main task -T_ROB1 sau în task-urile de mișcare specifice sistemului MultiMove.

Exemple:

SearchL di1, sp, p10, v100, *probe*;

TCP-ul corespunzator *probe* se mișcă liniar către punctul aflat în poziția p10 cu viteza v100. Când semnalul digital di1 se activează, poziția curentă va fi stocată în variabila sp.

SearchL \Stop, di2, sp, p10, v100, probe;

TCP-ul corespunzator probe se misca liniar catre pozitia p10. Cand valoarea semnalului di2 trece din 0 in 1, pozitia curenta este stocata in sp, iar robotul se opreste imediat. Dacă semnalul nu se activează pe parcursul mișcării atunci apare o eroare care trebuie analizată.

3.4 Generare de noi puncte in mod automat, funcțiile OffsS, RelTool

Pentru alterarea unor puncte în sistemul de coordonate global respectiv în sistemul de coordonate al efectorului terminal montat pe brațul robotic limbajul Rapid pune la dispozitie 2 funcții OFFS

Instrucțiunea OFFS este folosită pentru a deplasa un punct cu o anumită distanța în sistemul de coordonate curent.

Exemplu utilizare

p5:= Offs(p5, 0, 2, 20);

Punctul p5 este deplasat 2mm pe axa Y și 20mm pe axa Z.

Structură: Offs (Point , XOffset , YOffset , ZOffset);

Unde:

- Point punctul de tipul robtarget ce se dorește a fi deplasat.
- XOffset – deplasarea pe axa X în mm.
- YOffset – deplasarea pe axa Y în mm.
- ZOffset – deplasarea pe axa Z în mm.

Instrucțiunea RelTool este folosită pentru a deplasa și/sau a roti un punct cu o anumită distanța/unghi în raport cu unealta curentă.

Structură: RelTool (Point , Dx , Dy , Dz , [\Rx] , [\Ry] , [\Rz]);

Unde:

- Point punctul de tipul robtarget ce se dorește a fi deplasat/rotit;
- Dx deplasarea pe axa X în mm;
- Dy deplasarea pe axa Y în mm;
- Dz deplasarea pe axa Z în mm;
- [\Rx] rotatia pe axa X în grade;
- [\Ry] rotatia pe axa Y în grade;
- [\RZ] rotația pe axa Z în grade.

3.5 WaitDI / WaitDO

Robotul poate fi programat să aștepte o anumită perioadă de timp sau să aștepte până la îndeplinirea unei condiții arbitrare (ex.: activarea/dezactivara unei intrări/ieșiri digitale)

Instrucțiunea WaitDO se folosește pentru a aștepta ca un semnal digital de ieșire să fie setat(0 sau 1).

Structură: WaitDO Signal, Value [\MaxTime] [\TimeFlag];

Unde:

- Signal: numele semnalului digital;
- Value: valoarea dorită pentru semnal 0 sau 1;
- [\MaxTime]: durata de timp maximă permisă, dacă este atinsă această perioadă de timp se returnează eroare;
- [\TimeFlag]: parametrul de ieșire care conține valoarea TRUE când se atinge perioada de timp maximă permisă, daca este inclus acest parametru nu se mai returnează eroare la depășirea perioadei de timp maxim.

3.6 SETDO

Instrucțiunea SetDO este folosită pentru a schimba valoarea unui semnal de iesire digital cu sau fără întârziere sau sincronizare. Pentru acționare gripper.

Structură: SetDO [\SDelay]|[\Sync] , Signal , Value;

Unde:

- [\SDelay] întârzierea care se dorește în s;
- [\Sync] sincronizare;
- Signal numele semnalului care se dorește a fi schimbat;

Value valoare dorită pentru semnal; 0 sau 1.

3.7 WaitTime

WaitTime este folosit pentru a aștepta o anumită perioadă de timp. Această instrucțiune poate fi, de asemenea, utilizată pentru a aștepta până când robotul și axele externe s-au oprit.

Structură: WaitTime [\InPos]Time;

Unde:

[\InPos] - Dacă se folosește acest argument, atunci robotul și axele externe trebuie să se oprească înainte ca timpul de așteptare să înceapă să fie numărat. Acest argument poate fi utilizat numai dacă task-ul controlează unități mecanice.

Time - Timpul, exprimat în secunde, în care execuția programului este supendată. Valoare minimă, valoare maximă fără limită. Rezoluție 0,001 s.

Exemplu de utilizare: la acționarea efectorului terminal, cand se operează cu grippere acționate pneumatic sau magnetice este recomandat sa se aștepte un interval de timp pentru a se face bine prinderea piesei înainte de reluarea traiectoriei.

3.8 EulerXYZ & OrientZYX

Sistemul de operare RobotWare lucrează cu orientări ale sistemelor de coordonate exprimate în cuaternioni. Pantru a face conversia unghiuri Euler <-> cuaternioni sistemul oferă 2 funcții după cum urmează:

EulerZYX calculează cele 3 unghiuri Euler în convenția ZYX intrinsec: întâi rotația peste Z, apoi rotația peste noul Y și apoi rotația peste noul X.

OrientZYX formează o orientare din unghiurile Euler specificate.

Exemplu de utilizare EulerZYX (cei 3 parametri \X, \Y și \Z se exclud mutual, prezența a mai mult de 1 parametru generând o eroare):

```
VAR num anglex;
VAR num angley;
VAR num anglez;
VAR pose object;
...
anglex := EulerZYX(\X, object.rot);
angley := EulerZYX(\Y, object.rot);
anglez := EulerZYX(\Z, object.rot);
```

Exemplu de utilizare OrientZYX:

```
VAR num anglex;
VAR num angley;
VAR num anglez;
VAR pose object;
...
object.rot := OrientZYX(anglez, angley, anglex)
```

3.9 CPos, CRobT

Pentru citirea poziției, respectiv a poziției și a orientării curente a robotului folosind o unealtă și un sistem de coordonate specificate limbajul Rapid are la dispoziție funcțiile Cpos și CrobT.

3.10 Distance

Funcția calculează distanța (Fig. 4-20) între 2 puncte din spațiu folosind formula de mai jos:

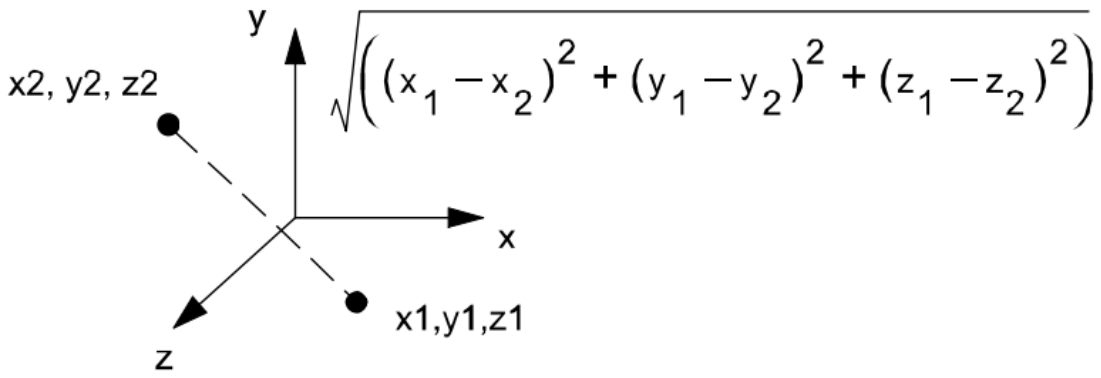


Fig. 4-20 – distanța între 2 puncte P1(x1,y1,z1) și P2(x2,y2,z2) în spațiu

3.11 CalcJointT & CalcRobT

Transformarea bidirecțională între coordonatele carteziene și valorile articulațiilor pentru care se atinge punctul respectiv se calculează folosind funcțiile CalcJointT & CalcRobT după cum este descries în figura de mai jos:

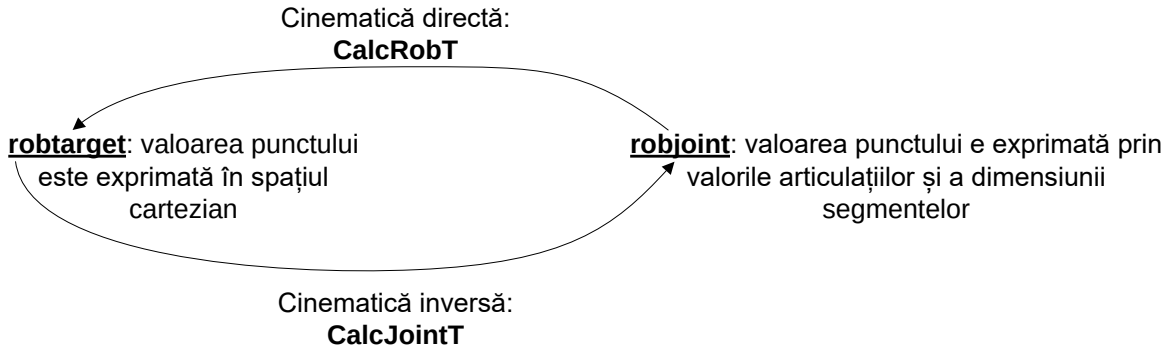


Fig. 4-21 – Modelele cinematice directe și inverse (legătura dintre spațiul cartezian și cel al articulațiilor)

3.12 Instrucțiuni matematice

Pentru realizarea de stive ordonate de piese este nevoie de instrucțiunile matematice care calculează câtul și restul unei operații de împărțire. Aceste funcții sunt DIV și MOD.

Exemplul următor ilustrează modalitatea de folosire a funcțiilor DIV și MOD:

```
reg1 := 14 DIV 4
reg2 := 14 MOD 4
```

Valoarea returnată de DIV este 3, acesta reprezentând câtul împărțirii. Valoarea returnată de MOD este 2 aceasta reprezentând restul împărțirii.

3.13 Instrucțiunea poseMult

PoseMult este o operație folosită pentru a calcula produsul a două transformări de poziție. O utilizare tipică este de a calcula o nouă poziție ca rezultat al unei deplasări care acționează asupra unei poziții originale. În figura de mai jos (Fig. 4-22) este ilustrată modalitatea de multiplicare a 2 poziții.

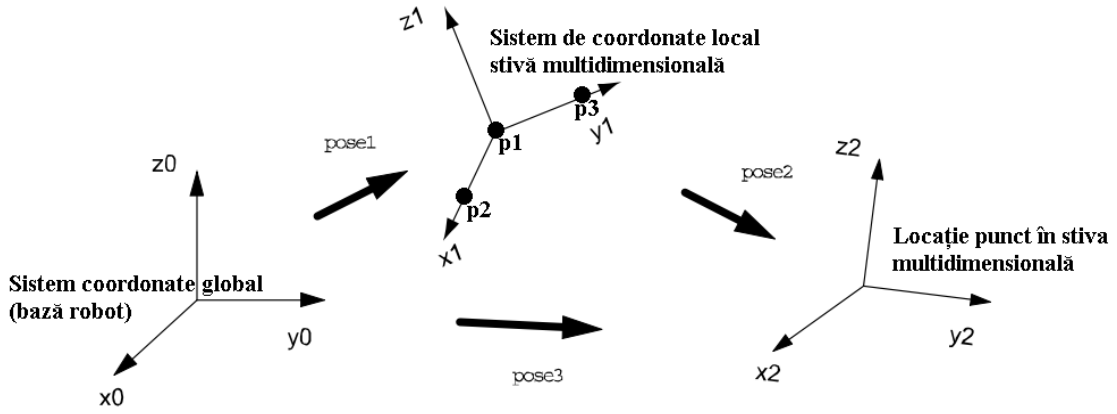


Fig. 4 -22 – Principiul de calcul al punctelor de deservit pentru stive multidimensionale rotite peste toate axele

pose1 reprezintă locația sistemului de coordonate 1 exprimată în sistemul de coordonate 0. *pose2* reprezintă locația sistemului de coordonate 2 exprimată în sistemul de coordonate 1. Transformarea care dă *pose3*, locația sistemului de coordonate 2 exprimată în sistemul de coordonate 0, este obținută de produsul celor două transformări:

```
VAR pose pose1;
VAR pose pose2;
VAR pose pose3;
...
pose3 := PoseMult(pose1, pose2);
```

4 Comparație între limbajele V+ și RAPID în termeni de operații mișcare

Tabel 4-1. Comparație între limbajul V+ și RAPID din punct de vedere al instrucțiunilor

V+	Descriere	RAPID	Descriere
MOVE	Mișcare unghiulară	MoveJ	Mișcare liniară în articulații
MOVES	Mișcare dreaptă	MoveL	Mișcare liniară în cartezian
APPRO	Apropiere către o locație	Se generează un punct intermediar cu Offs sau RelTool și apoi este atins cu mișcare liniară	Apropiere/depărtare de o locație
DEPARTS	Depărtare de la o locație		
BREAK	Suspendă execuția programului până ce mișcarea curentă este încheiată.	Zonă de toleranță *fine*.	În Rapid atingerea punctului final se face declarând mișcare spre acesta cu zonă *fine* – punct final.
SPEED	Viteza mișcărilor	SpeedRefresh	Viteza mișcărilor
OPENI/ SIGNAL	Deschide imediat gripper-ul robotului.	SetDo ….. 1;	Setează o ieșire digitală pe semnal 1.
CLOSEI/ SIGNAL	Închide imediat gripper-ul robotului.	SetDo ….. 0;	Setează o ieșire digitală pe semnal 0.
SIG	Așteaptă o intrare	WaitDO	Așteaptă o intrare
CALL	Suspendă executarea programului curent și continuă execuția cu un program nou (adică o subrutină).	ProcCall	Suspendă executarea programului curent și continuă execuția cu un program nou (adică o subrutină).
=SHIFT	Returnează o valoare de transformare ce rezultă din deplasarea poziției parametrului de transformare cu cantitățile de deplasare.	pose	Schimbă o coordonată a sistemului în altă coordonată. Componente: trans, rot

5 Exemple de programare Rapid (+rulare cu tot cu pointer si rulare inainte si inapoi)

5.1 Prezentare module sistem și module program

Pentru programarea brațului manipulator se poate utiliza atât RobotStudio cât și modulul de comandă manuală FlexPendant. FlexPendant este cel mai potrivit pentru modificarea programelor, cum ar fi punctele, traiectoriile și valoarea semnalelor de intrare/ieșire, în timp ce RobotStudio este preferat pentru programarea mai complexă. După cum a fost specificat mai sus traiectoriile sunt organizate în module de tip sistem și program, acestea din urmă conținând funcții și proceduri care implementează traiectoriile dorite.

5.2 Moduri de operare

Echipamentul are două moduri de lucru automat (X) și manual (D) (și 100% manual), comutarea între cele două relizându-se prin acționarea unui comutator mecanic situat pe controller (Fig. 4-23).

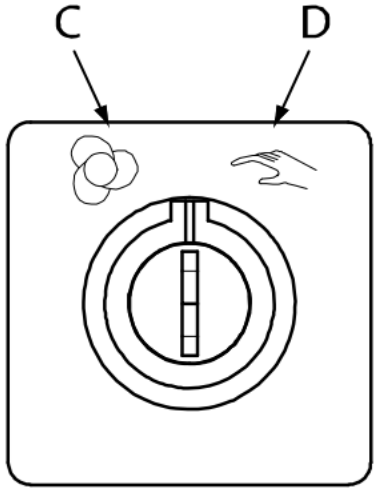


Fig. 4-23 – Comutarea între modurile de lucru manual și automat

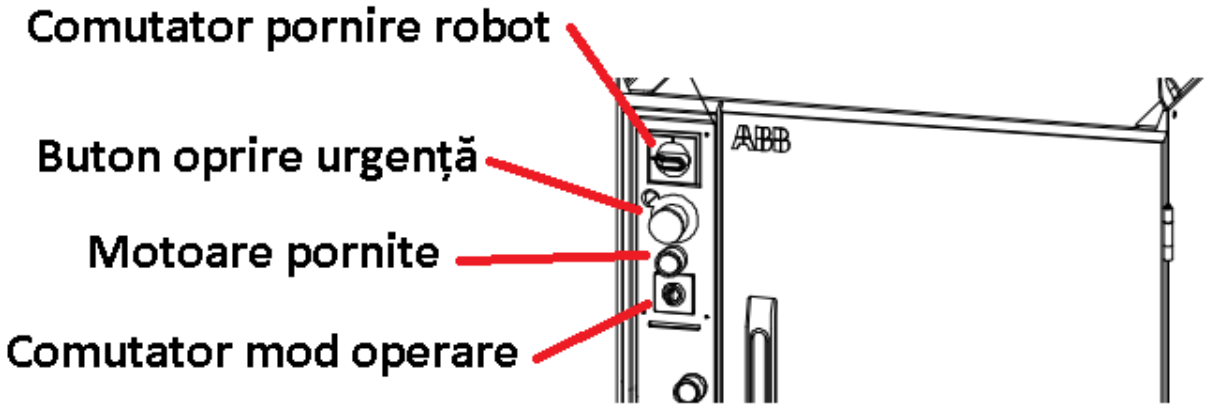


Fig. 4.24 – Comutatoarele și butoanele de siguranță de pe controllerul robot

Cele două moduri de lucru sunt implementate pentru a-l face pe utilizator conștient de riscurile aferente rulării aplicației cu viteză ridicată în mediul real – coliziuni. Astfel, în modul de lucru manual viteză este limitată la 200mm/sec iar pentru execuția programului este necesar ca operatorul să stea cu mâna pe butonul de siguranță cu trei poziții de pe FlexPendant. În modul de lucru automat se trece punând comutatorul din figura de mai sus (Fig 4-24) pe poziția C – automat – urmat de confirmarea Motoare active (en.: enable motors). O altă direfență intre cele 2 moduri de lucru este că în modul manual pentru a se edita programul este nevoie de o cerere specială care este validate de pe FlexPendant în timp ce în modul automat (auto) nu este nevoie de nici o cerere fiind necesar doar aplicarea modificărilor la sfârșitul editării. Modul manual este ideal pentru programarea pe un echipament real ea asigurând roboticianul că echipamentul nu va porni imediat ce pornește programul (este nevoie de o confirmare fizică). Într-o situație de lucru reală este descurajată utilizarea modului automat (sau de producție) pentru progrmarea echipamentului. Având în vedere că acest manual descrie modul de lucru cu simulatorul (acesta neprezentând riscuri de rănire prin coliziune) se poate opera în siguranță, din rațiuni de simplitate, în modul automat. Comutarea între cele 2 moduri în simulator se face din meniul *Controller* > *Virtual FlexPendant*. În urma acționării acestei comenzi apare un echipament virtual de control (Fig. 4-25).

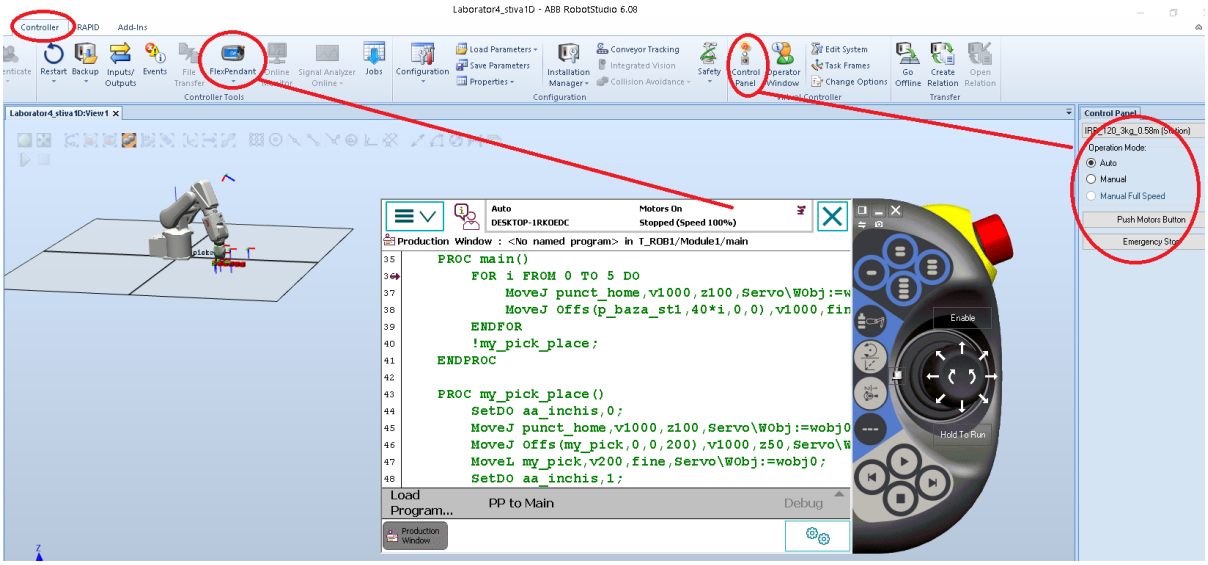


Fig. 4-25 – Modul de accesare a echipamentului virtual de control, respectiv a panoului de selecție a modului de lucru

5.2.1 Modul automat

În modul automat, funcția de siguranță a dispozitivului de activare este ocolită astfel încât manipulatorul să poată să se deplaseze fără intervenția omului. Modul automat este modul de operare în care sistemul de control al robotului funcționează în conformitate cu programul de sarcini, cu măsuri de protecție funcționale. Acest mod permite controlul manipulatorului, de exemplu, prin utilizarea semnalelor I/O de pe controller. Un semnal de intrare poate fi utilizat pentru a porni și a opri un program RAPID, altul pentru a activa motoarele de pe manipulator.

Sarcini efectuate în mod normal în modul automat: procese de pornire și oprire, încărcarea, pornirea și oprirea programelor RAPID, întoarcerea manipulatorului pe traseul acestuia la revenirea în funcțiune după o oprire de urgență, copierea de rezervă a sistemului,

restaurarea copiilor de rezervă, instrumente de curățare, pregătirea sau înlocuirea obiectelor de lucru, efectuarea altor sarcini orientate spre proces.

Joggingul nu este posibil în modul automat.

5.2.2 Modul manual

În modul manual, mișcarea manipulatorului este controlată manual. Dispozitivul de activare trebuie apăsat pentru a activa motoarele manipulatorului, adică pentru a permite mișcarea. Modul manual este folosit la programare și la verificarea programului. La unii roboți există două moduri manuale, modul manual de viteză redusă și modul manual de viteză maximă.

În modul manual, manipulatorul este operat cu personal în imediata vecinătate. Manevrarea unui manipulator industrial este potențial periculoasă și, prin urmare, manevrele ar trebui efectuate în mod controlat.

Sarcini efectuate în mod normal în modul automat: mișcarea manipulatorului înapoi pe traseu atunci când se întoarce la operațiune după o oprire de urgență, corecția valorii semnalelor I/O după condiții de eroare, crearea și editarea programelor RAPID, pornirea, trecerea și oprirea executării programului, de exemplu în timpul testării unui program, reglarea pozițiilor programate.

5.3 Crearea unui nou modul program cu tot cu subrutină

5.3.1 RobotStudio

Crearea de module, proceduri și funcții noi se face din meniul Rapid (Fig. 4-26). Acest modul regrupează toate datele, procedurile și funcțiile necesare realizării unei traiectorii. Pentru rularea în producție sistemul accepta minim un modul și doar unul.

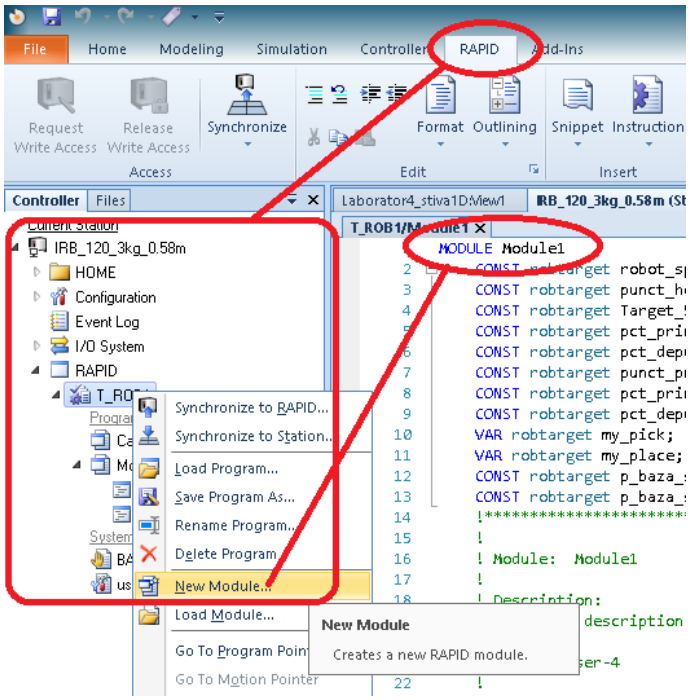


Fig. 4-26 – Crearea unui modul program nou

5.3.2 FlexPendant

Crearea unui modul

a. Din meniul ABB se apasă pe Program Editor.
b. Se apasă pe Modules.
c. Se apasă pe File –> New Module.
d. Se scrie numele modulului, apoi se apasă OK.
e. Se selectează tipul modului: program sau sistem. Se apasă OK.

Crearea unei rutine

a. Din meniul ABB se apasă pe Program Editor.
b. Se apasă pe Routines.
c. Se apasă pe File –> New Routine.
d. Se scrie numele rutinei, apoi se apasă OK.
e. Se selectează tipul de rutină: procedură – folosită pentru o rutină normală fără să întoarcă valoare, funcție – utilizată pentru o rutină normală cu returnare de valoare, capcană (en.: trap) – pentru o rutină de întrerupere.
f. Se alege modulul în care se adaugă rutina.

Adăugarea instrucțiunilor

a. Din meniul ABB se apasă pe Program Editor.
b. Se atinge instrucțiunea sub care se dorește adăugarea unei noi instrucțiuni.
c. Se apasă pe Add instrunction, apoi se alege din listă.

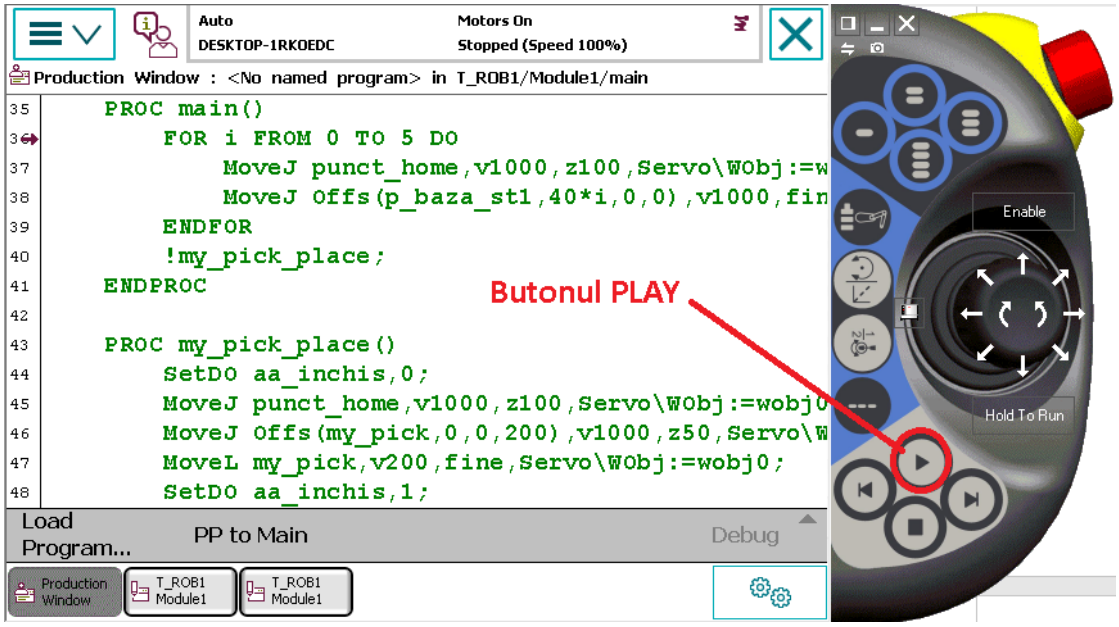


Fig. 4-27 – Programare pe FlexPendant

5.4 Apel subprogram / Lansare in execuție

Lansarea în execuție a unei traiectorii se poate face în două modalități: butonul Play din meniul simulare (Simulation) (Fig. 4-28) și butonul Run de pe FlexPendant-ul virtual (Fig. 4-27).

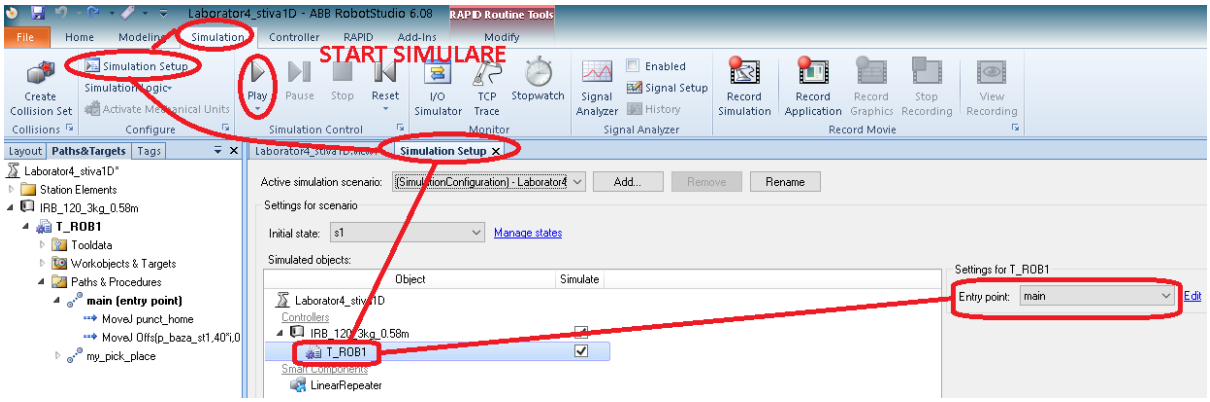


Fig. 4-28 – Lansarea unei simulări și configurarea traiectoriei și stării de start

În ambele cazuri trebuie specificat punctul de intrare. În modul simulare trebuie specificată și starea mediului (localizarea componentelor și starea intrărilor/ieșirilor). Se poate face simularea și fără specificarea stării curente, dar în urma unor rulări repetate pot apărea comportamente nedorite. Specificarea stării curente se face din meniul aferent butonului Reset (tab Simulation) (Fig. 4-29).

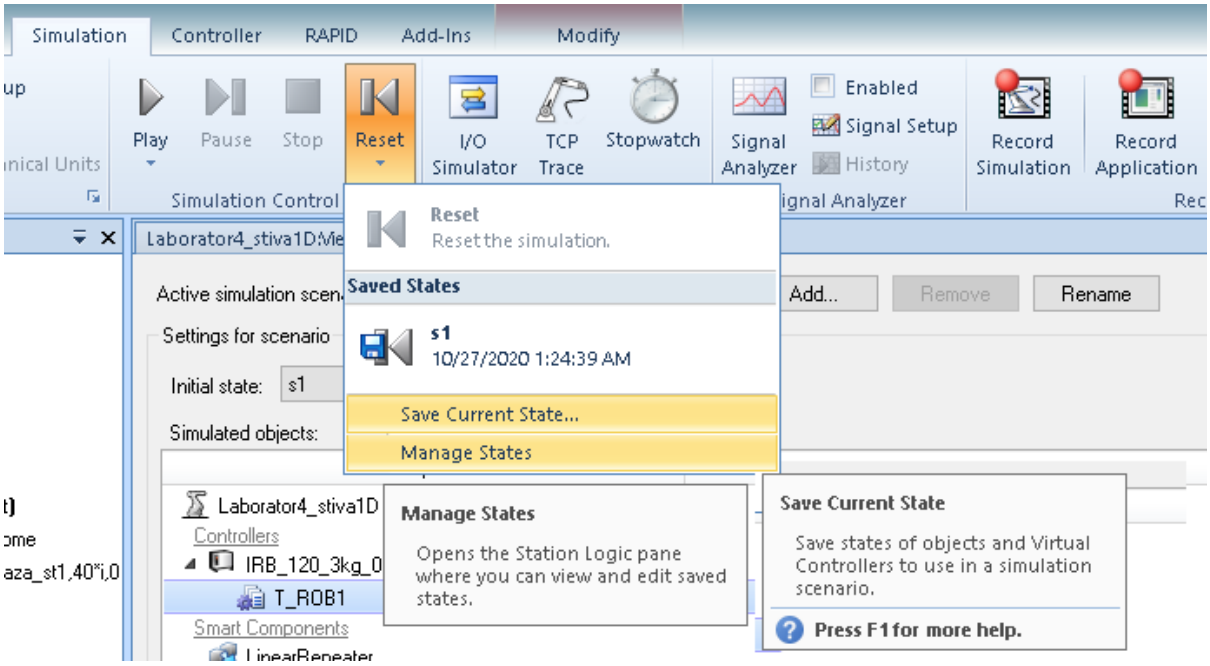


Fig. 4-29 – Salvarea stării curente și specificarea stării inițiale

Lansarea traiectoriei din FlexPendant se face din modul producție sau din Program Editor după cum urmează:

Din fereastra de producție:

1) În meniul ABB se apasă pe Production Window –> PP to Main.

   (**PP to Main** din fereastra de producție va reseta pointer-ul programului la intrarea de producție în toate sarcinile normale, inclusiv în sarcinile dezactivate în panoul de selectare a sarcinii.)
2) Se apasă pe butonul Start de pe FlexPendant pentru începerea programului.

Din Program Editor:

1) În meniul ABB se apasă pe Program Editor –> Debug –> PP to Main

   (**PP to Main** din Program Editor va reseta pointer-ul programului la intrarea de producție numai în sarcina specificată, chiar dacă sarcina este dezactivată în panoul de selectare a sarcinii.)
2) Se apasă pe butonul Start de pe FlexPendant pentru începerea programului.

6 Exercitii

Realizare aplicatie paletizare 1D de pe axa Y pe axa Z. Simularea se va face pentru un set de 10 obiecte. Modelul de gripper, structura sa (simetric/asimetric), modelul de piesă manipulată precum și modelul de prindere vor fi alese cursanți.

7 Intrebari

a) Cum diferă traiectoria de mai sus dacă pozițiile succesive rotite sunt rotite cu 90º?
b) Realizați aceeași aplicație, dar cu un gripper multi-cap (2 capete – numarul de piese să se dividă cu numărul de capete).

Capitol 5: Stive multidimensionale

Cuprins

1 Stive multidimensionale 95

2 Stive multidimensionale cu poziții rotite 99

3 Exerciții 99

4 Întrebări 99

1 Stive multidimensionale

În mod uzual în sistemele de fabricație produsele trebuie asamblate sau vin ordonate urmând a fi procesate ulterior. Deservirea individuală a fiecărei locații este ineficientă și în anumite cazuri imposibilă de aceea este de preferat ca în aceste cazuri piesele să fie procesate folosind conceptul de stivă multidimensională. O stivă multidimensională reprezintă un set de piese identice ordonate după una, două sau trei axe. Aceasta este caracterizată de o dispunere uniformă a pieselor – distanțe egale între piese de-a lungul axelor. În figura de mai jos sunt ilustrate diferite ordonări de piese pe 1, 2 respectiv 3 axe (Fig. 5-1). Aceste axe nu trebuie să fie paralele și de același sens cu axele sistemului de coordonate robot pentru ca sistemul de operare să poată procesa în mod automat punctele. Metoda de lucru cu stive multidimensionale este următoarea: se învață un punct și restul să fie generate matematic. Pentru simplitate punctul învățat se recomandă să fie la o extremitate, acesta purtând denumirea de bază a stivei.

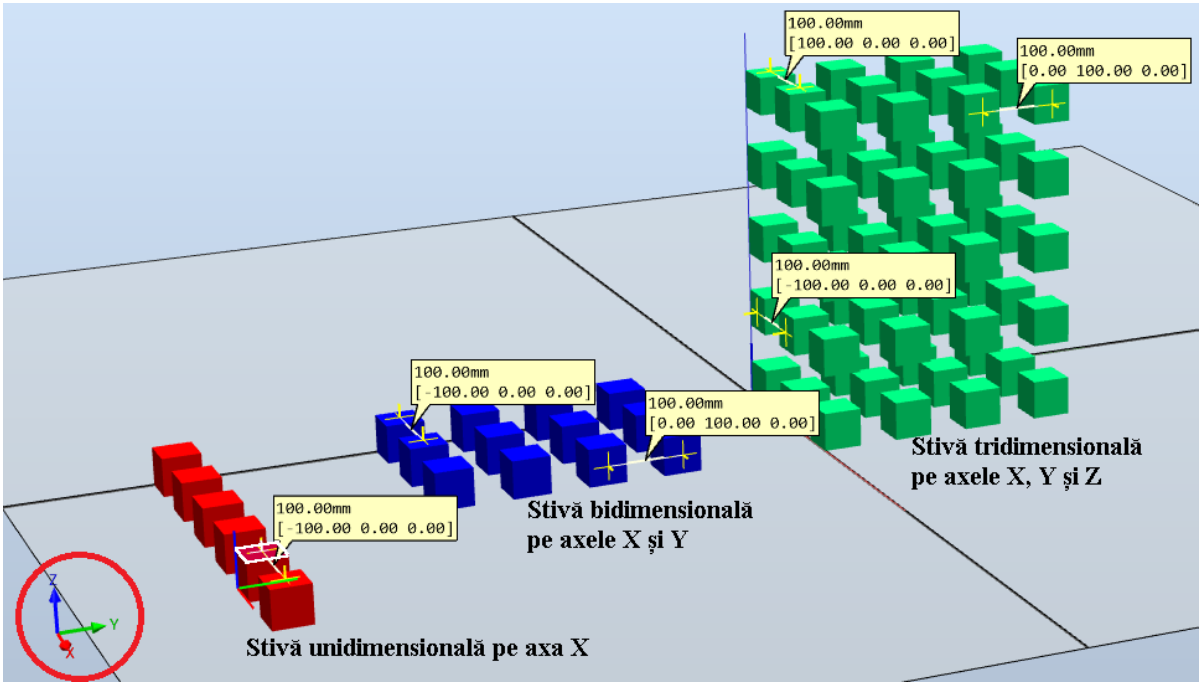


Fig. 5-1 – Diferite tipuri de stive multidimensionale

Cu toate că stivele se aseamănă ca și principiu cu vectorii (1D, 2D și 3D) nu este recomandată procesarea folosind câte un index pentru fiecare dimensiune ci folosirea unui singur index ce reprezintă contorul pieselor din stivă și plasarea ei în funcție de acest contor.

```
Pentru i=1...nr_piese_stiva
     pozitie(i) (x,y,z, rx,ry,rz)=f1(i)*dx, f2(i)*dy, f3(i)*dz
     apel traiectorie manipulare
```

În continuare vom trata stivele 3D, acestea fiind cele mai generice. Prin reducerea unei axe sau a 2 axe la 1 se obțin stive 2D respectiv 1D. Vor fi tratate două cazuri: a) stive care au axele de ordonare paralele cu axele robotului, caz în care se poate folosi instrucțiunea de translatare *Offs* și b) stive care au axele rotite față de axele robotului caz în care calculul poziției de deservire va fi calculat folosind o compunere de 2 transformări. Cele 2 transformări sunt: o transformare din baza robotului până în baza stivei rotite urmată de o transformare locală conform punctului a).

In momentul deservirii unei stive contează modalitatea de umplere/golire, aici fiind nevoie ca robotul să respecte un set de constrângeri precum: să pornească operațiile de la un capăt (nu din centrul stivei) și să deservească puncte accesibile (ex.: în cazul unei stive 3D nu se pot lua piese de jos, acestea nefiind accesibile din cauza pieselor dispuse deasupra). În anumite situații cea de-a doua constrângere nu există. De exemplu, pentru o stivă bidimensională (pe axele X și Y / planul orizontal XOY) pot exista două modalități (Fig. 5-2): i) deservire piese pe axa X apoi pe axa Y, respectiv b) deservire piese pe axa Y apoi pe axa X. În cazul 3D sunt în total 6 modalități: XYZ, XZY, YXZ, YZX, ZXY, ZYX.

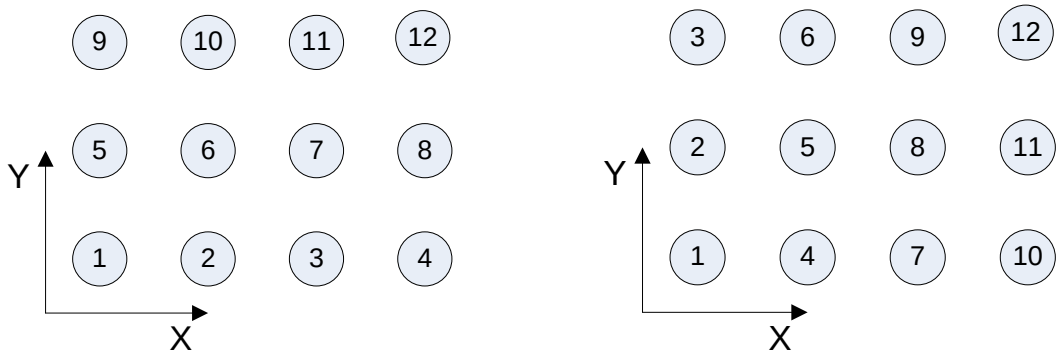


Fig. 5-2 – Diferite modalități de deservire a punctelor unei stive bidimensionale

Principala caracteristică a deservirii stivelor este modalitatea de generare a punctelor de interes în funcție de indexul curent (i). Astfel sunt necesare 2 funcții: *restul împărțirii* pentru resetarea indexului pe axa curentă a stivei și *câtul împărțirii* pentru incrementarea indexului după completarea axei precedente.

```
Pentru i=1…12
     Poziție pe X = rest (i-1)/4 * dx   !pentru     completarea
întâi pe X apoi pe Y
     Poziție pe Y = cât (i-1)/4 * dy    !  pentru  completarea
întâi pe X apoi pe Y
```

```
Poziție pe X = cât (i-1)/3 * dx    !pentru    completarea
întâi pe Y apoi pe X

Poziție pe Y = cât (i-1)/3 * dy    !  pentru  completarea
întâi pe Y apoi pe X
```

Se observă că funcția "rest" este folosită pentru resetare index **și** funcția „cât" este folosită pentru incrementare index (nu are limită). La adăugarea celei de-a treia axe va trebui ca după completarea unei "felii" să se reseteze indexul pe ambele axe precedente.

În Rapid se vor folosi funcțiile `mod` si `div` pentru calculul restului și al câtului.

```
VAR num j;
VAR pose placePosition;
FOR i FROM 0 TO 11 DO

    my_pick:=Offs(p_baza_st1,40*(i mod 6),70*(i div 6),0);
    my_place:=Offs(p_baza_st2,40*(i MOD 3),-40*(j mod 3),40*(i DIV 9));
    my_pick_place;
ENDFOR
```

Fig. 5.3 – Metoda de calcul a deplasamentelor pe X, Y și Z în sistemul de coordonate global

Pentru stive rotite sistemul de coordonate atașat (axele locale de ordonare a pieselor) nu mai este cunoscut (în situația precedentă acest sistem era sistemul de coordonate din baza robotului). Acesta este cazul general deoarece în realitate este dificil de asigurat paralelismul celor 2 sisteme de coordonate, precedându-se în următorul mod: se poziționează robotul pentru a putea deservi postul/posturile de lucru, se poziționează stiva astfel încât să intre în spațiul de lucru al robotului și apoi se calculează rotirea celor 2 sisteme folosind instrucțiunea *DefFrame(p1,p2,p3)* unde punctele p1, p2 reprezintă axa OX, perpendiculara p3, (p1p2) reprezintă axa OY, iar axa OZ rezultă din regula mâinii drepte.

Mergând pe principiul de mai sus se pot învăța (cu robotul sau din simulator) un set de 3 puncte pe baza cărora se calculează transformarea până în baza stivei, apoi punctul decalat din stivă este alterat la nivel local. Structura de date de tip *pose* este modificată conform formulelor stivelor multidimensionale (cu orientare 0) rezultând o transformare locală – *local_trans*. Din transformarea care duce în baza stivei se obține o structură de tip pose – global_trans, iar din compunerea celor 2 transformări se obține transformarea care duce în punctul de interes. Dacă se adaugă la această transformare configurația robotului se obține un punct de tip țintă carteziană care teoretic trebuie atinsă (dacă piesele au fost dispuse în spațiul de lucru al robotului). Pentru a verifica corectitudinea soluției se pot face 2 verificări: o verificare offline care constă în mișcarea robotului pe și peste stiva deservită, respectiv o verificare online care constă în calculul soluției de cinematică inversă. Dacă cinematica inversă are soluție atunci robotul poate ajunge în punctul calculat (se apelează *CalcJointT* și se verifică variabila *ERRNO* dacă apare o eroare). Adițional se poate specifica faptul că în atingerea punctului se poate schimba configurația de lucru (ConfJ/ConfL).

Dacă adițional se dorește rotirea punctelor deservite atunci trebuie alterat în mod individual structura *rot* aferentă acestor puncte. Lucrul acesta se poate face obținând unghiurile Euler (EulerZYX) la care se adaugă deplasamentul dorit apoi se reface poziția (OrientZYX).

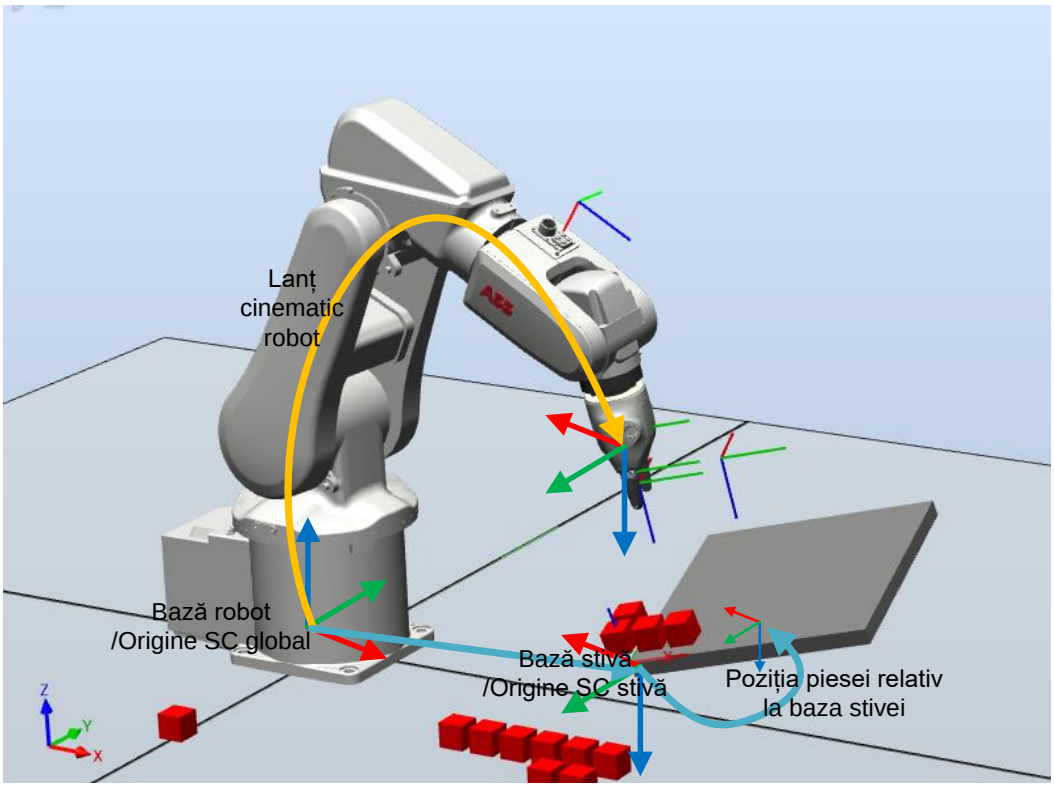


Fig. 5-4 – Principiul de calcul al punctelor de deservit pentru stive multidimensionale rotite peste toate axele

```
PROC main()
    VAR num j;
    VAR pose placePosition;
    FOR i FROM 0 TO 11 DO

        my_pick:=Offs(p_baza_st1,40*(i mod 6),70*(i div 6),0);

        j:=i DIV 3;

        !calcularea pe stiva
        !ConfL and ConfJ
        global_trans.trans:=p_plan_inclinat.trans;
        global_trans.rot:=p_plan_inclinat.rot;

        local_trans.trans.x:=40*(i MOD 3);
        local_trans.trans.y:=-40*(j mod 3);
        local_trans.trans.z:=-50-40*(i DIV 9);
        local_trans.rot:=OrientZYX(0,0,0);

        placePosition:=PoseMult(global_trans,local_trans);

        my_place.trans:=placePosition.trans;
        my_place.rot:=p_plan_inclinat.rot;
        my_pick_place;
    ENDFOR
    MoveJ punct_home,vmax,z100,Servo\WObj:=wobj0;

    !my_pick_place;
ENDPROC
```

Fig. 5-5 – Metoda de calcul a deplasamentelor pe X, Y și Z pentru o stivă multidimensională rotită peste toate cele 3 axe

2 Stive multidimensionale cu poziții rotite

Poziția relativă a unei piese într-o stivă multidimensională poate fi alterată individual folosind setul de funcții OrientZYX, respectiv EulerZYX și OrientZYX dacă nu se știe apriori valoarea orientării punctului alterat. Trebuie avut în vedere că sistemul RobotWare operează intern cu cuaternioni în reprezentarea matematică a punctelor, iar reprezentarea operator este de tip Euler ZYX intrinsec (întâi rotația peste Z, apoi peste Y', apoi peste X").

Mai jos este prezentat codul pentru dispunerea unui set de piese într-o stivă unidimensională pe axa Z cu rotații de 0°, repectiv 90° pentru piese succesive:

```
local_trans.trans.x:=40*(i MOD 3);
local_trans.trans.y:=-60*(j mod 3);
local_trans.trans.z:=-50-40*(i DIV 9);
local_trans.rot:=OrientZYX(90*(i mod 2),0,0);

placePosition:=PoseMult(global_trans,local_trans);
```

Fig. 5-6 – Modalitatea de alterare a rotației unei transformări

OBS: trebuie avută mare atenție la configurația finală a punctelor de depunere (ConfJ, ConfL))

3 Exerciții

Se cere:

- să se realizeze o piesă de 20/20/20mm care se va clona de 12 ori într-o stivă bidimensională pe X și Y global
- să se realizeze traiectoria robot care paletizează piesele din stiva de mai sus într-o stiva tridimensionala (3 pe X, 3 pe Y, 2 pe Z).
- OBS: se vor paletiza atâtea piese cât se poate
- Piesele adiacente pe axa de completare vor fi rotite la 90°

4 Întrebări

Cum se definește un sistem de coordonate folosind doar partea de poziție a coordonatelor carteziene?

Capitol 6: Componente active

Cuprins

1 Componente Smart (en.: SmartComponent) 100

1.1 Creare/clonare și distrugere 102

1.2 Obiecte gravitaționale 106

1.3 Mișcarea automată 107

1.4 Crearea, salvarea, exportarea și importarea de componente active 107

1.5 Realizarea de entități de tip senzor prezență și măsurare distanță 109

1.6 Conexiuni SmartComponent – controller robot, accesul variabilelor controller în componentele active 109

2 Exerciții 110

3 Întrebări 110

1 Componente Smart (en.: SmartComponent)

Pentru vizualizarea în simulator a proceselor care au loc în sistemul fizic (ex.: prezentare piese, evacuare piese, transport, ș.a.) aplicația pune la dispoziție entitatea de tip componentă activă – en.: SmartComponent. Aceste componente sunt accesibile spre a fi create din meniul Simulation -> (*Simulation logic*) -> Station logic (Fig. 6-1).

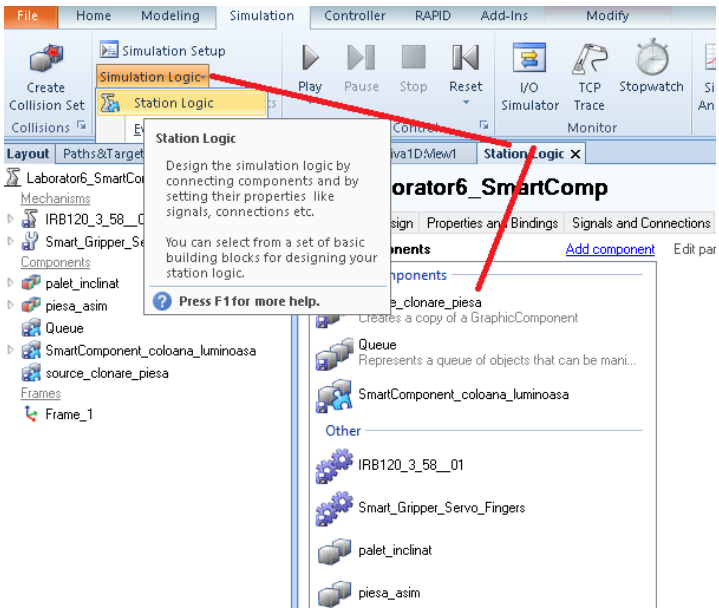


Fig. 6-1 – Metoda de acces la entitatea activă de tip SmartComponent

Din acest meniu se pot adăuga diferite tipuri de componente active pentru: procesare de semnale/date (e.g. logică booleană, conversie variabile, ș.a.), creare de obiecte, detecție/

senzoristică, realizare acțiuni (clonare, distrugere, ș.a.), manipulare obiecte, acces la variabilele controller-ului, specificare proprietăți fizice/cinematice și procesare generică (ex.: coadă de manipulat obiecte, generare numere aleatoare, ș.a.) (Fig. 6-2). După selectarea opțiunii *Add component* și alegerea componentei dorite aceasta este adăugată în secțiunea *Smart Components* din meniul *Station Logic* (Fig. 6-2). În acest meniu componentele active pot fi adăugate direct în rădăcină, sau grupate într-o componentă inteligentă pentru a fi gestionate mai ușor (ex.: crearea unei coloane luminoase constând în mai multe componente active care își schimbă culoarea în funcție de un set de ieșiri digitale ale controllerului robot).

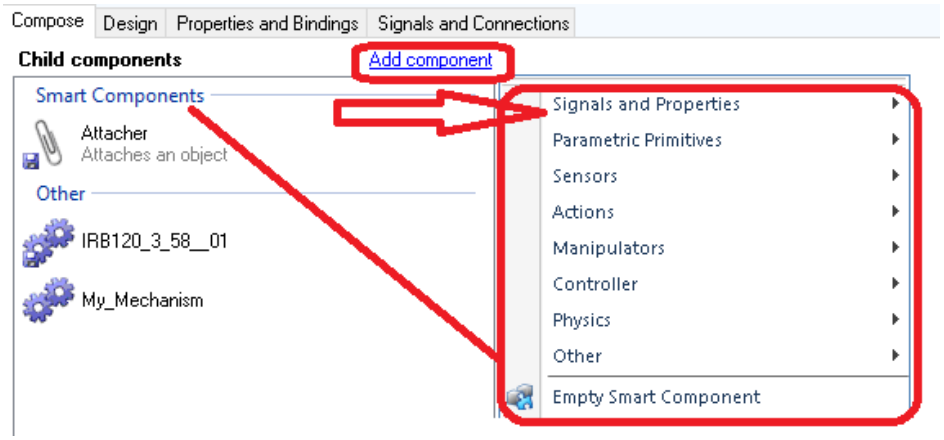


Fig. 6-2 – Modalitatea de acces la entitatea activă de tip SmartComponent

Verificarea proprietăților componentelor active se poate face din meniul *Compose* sau din Layout -> Components, urmat de click dreapta pe componentă dorită și apoi *Propreties*. Din această fereastră (Properties) se pot configura un set de proprietăți de bază ale componentelor active precum obiectele asupra cărora acționează. În urma configurării din meniul Properties trebuie confirmată modificarea prin apăsarea butonului *Apply*.

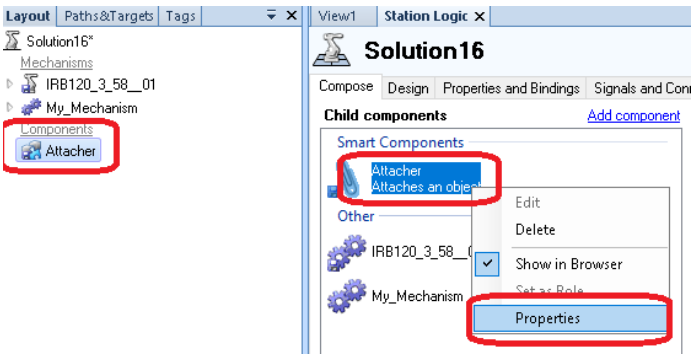


Fig. 6-3 – Configurarea proprietăților componentelor active

Componentele active se pot grupa în fluxuri pentru simularea de procese complexe precum crearea unui obiect și apoi distrugerea sa după un anumit interval de timp (Fig. 6-4). Această operațiune se realizează din secțiunea *Design* a meniului *Station Logic*. Aici se regăsesc un număr suplimentar de proprietăți ale componentelor active care pot fi configurate, de data aceasta prin realizarea unor conexiuni de tip *DragAndDrop* (săgeți) între proprietăți similare ale componentelor diferite. Planșa de modelare suportă un număr ridicat de componente, existând posibilitatea de a realiza operații de mărire/micșorare pentru vizualizarea tuturor. În situația în care se operează cu procese foarte complexe se recomandă gruparea

componentelor de bază în componente active generice pentru a putea gestiona mai simplu conexiunile, dar și pentru a avea posibilitatea de a reutiliza aceste componente. Conexiunea la controller a componentelor active se realizează prin expandarea intrărilor/ieșirilor digitale din componenta *System1*. Aceasta din urmă nu apare ca și componentă grafică în meniul *Compose*, respectiv în *Components*.

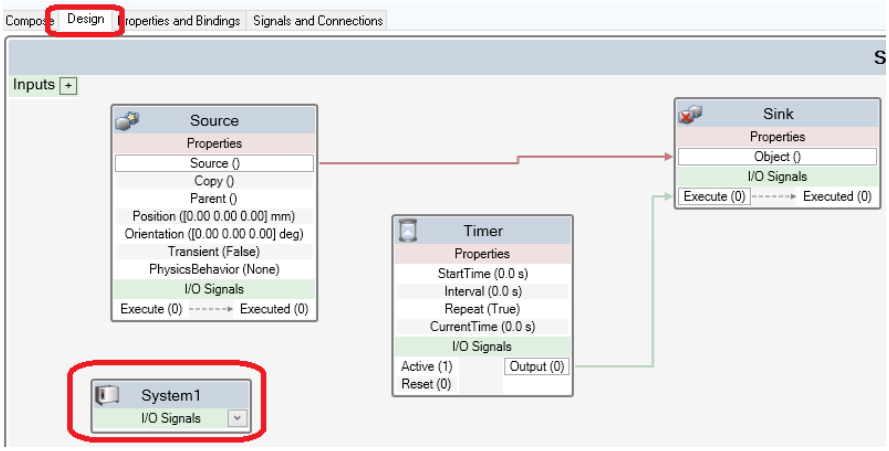


Fig. 6-4 – Realizarea unui flux logic între componente active

Componentele logice pot rula atât în modul static cât și în modul Play simulation (mod automat (Fig. 6-5)). Trebuie avut atenție la salvarea stării curente atunci când se operează în modul automat deoarece structura sistemului și poziționarea elementelor poate fi alterată în urma rulării componentelor active (ex.: o componentă activă de tip LinearMove – mișcare – va altera poziția elementelor componente, iar readucerea lor la loc se poate dovedi dificilă dacă nu s-au memorat datele necesare). Componentele active rulează în mod automat doar cu simularea pornită pe modul continuu (Fig. 6-5). În cazul în care simularea nu este în modul continuu componentele active vor funcționa doar pe parcursul mișcării robotului (ex.: dacă se dorește mișcarea unui obiect creat pe un interval mai mare decât ciclul robot, atunci în cazul Single cycle componenta se va mișca doar cât simularea este activă – cât se mișcă robotul).

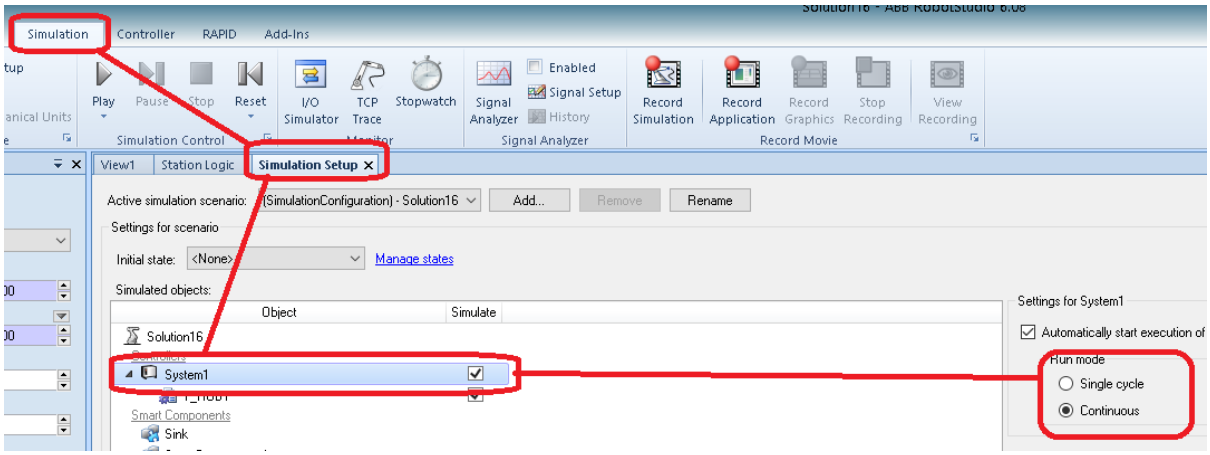


Fig. 6-5 – Trecerea simulării în modul continuu

1.1 Creare/clonare și distrugere

În cazul în care nu este posibilă tratarea pieselor sub formă de stive din cauza numărului mare, atunci se limitează numărul de piese prin creare și distrugere dinamică în punctele de

preluare respectiv de depunere. În sistemele fizice se folosesc dispozitive externe de prezentare/evacuare precum benzi conveioare sau containere care se schimbă ca urmare a golirii/umplerii. În simulator se folosesc componente active de tip *Source* (pentru clonare) respectiv *Sink* (pentru distrugere).

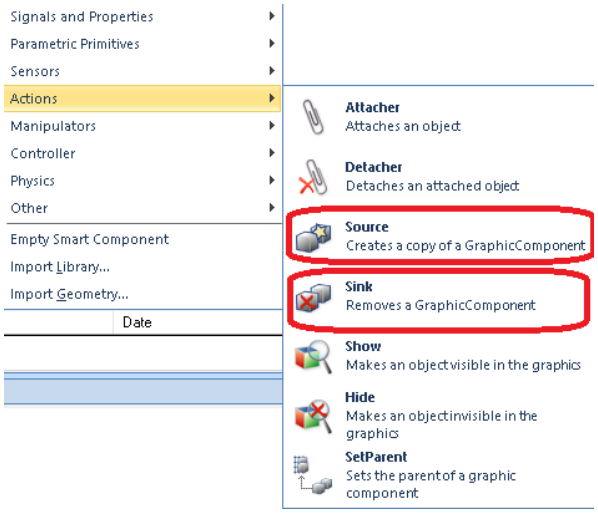


Fig. 6-6 – Trecerea simulării în modul continuu

Pentru clonarea unei componente se folosește componenta *Source* la care se configurează ce se clonează, poziția, comportamentul și semnalele necesare interacțiunii cu elementele din mediu (I/O digitale). În captura de ecran de mai jos (din meniul Design) (Fig. 6-6) se pot observa toate elementele configurabile ale componentei de clonare. De remarcat că această componentă nu are element grafic, ea existând doar în meniul *Station Logic*. Ea poate fi configurată atât din meniul de proprietăți cât și din meniul *Design*. Diferența dintre cele două constă în faptul că din *Properties* se specifică acele proprietăți constante pe când în *Design* se specifică proprietățile variabile (configurabile la rulare ca urmare a unei combinații de I/O digitale). Exemplu: obiectul clonat este de obicei un obiect existent și se specifică din *Properties* pe când clonele/copiile sale sunt elemente variabile și sunt specificate ca un flux (copia intră într-o altă componentă cum ar fi o componentă de distrugere; între clonare și distrugere obiectul este manipuslat de robot).

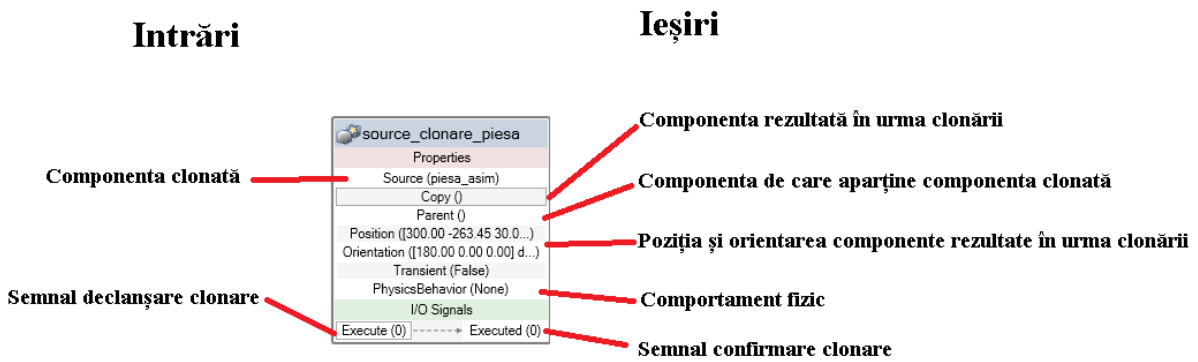


Fig. 6-7 – Configurarea unei componente de clonare în modul *Design*

Dacă o componentă de clonare este apelată de mai multe obiectele/componentele rezultante nu păstrează conexiunile pentru obiectele mai vechi astfel încât trebuie păstrate toate aceste obiecte pentru situația în care se dorește a fi șterse. Acesta este cazul uzual în care se

creează un set de obiecte care sunt paletizate și apoi distruse pentru a se începe un nou ciclu robot. În funcție de complexitatea procesului de manipulare precum și în funcție de cum se dorește a poziționa, respectiv atașa, obiectele ce rezultă în urma clonării există două posibilități de a stoca elementele precedente: a) folosirea unei componente active de tip coadă (Queue) și b) dispunerea componentelor într-o relație de tip copil părinte. Primul caz se pretează pentru situații în care în urma creării se vor realiza operații mai complexe precum manipulări folosind componente active. Cel de-al doilea caz se pretează pentru situații în care se dorește ca ansamblul rezultat în urma creării relațiilor părinte copil se manipulează ca un întreg (ex.: se dispun piese pe un platou apoi platoul este manipulat ca o entitate de sine stătătoare).

În cazul lucrului cu cozi de obiecte structura de manipulare este următoarea: o componentă de tip *Source* (pentru clonare) înseriată cu o componentă de tip *Queue* (pentru memorare). Legăturile din meniul *Design* sunt prezentate în figura de mai jos (Fig. 6-8) cu observația că trebuie definite 2 ieșiri digitale responsabile cu clonarea pieselor plus adăugarea lor în coadă și o ieșire responsabilă cu golirea cozii.

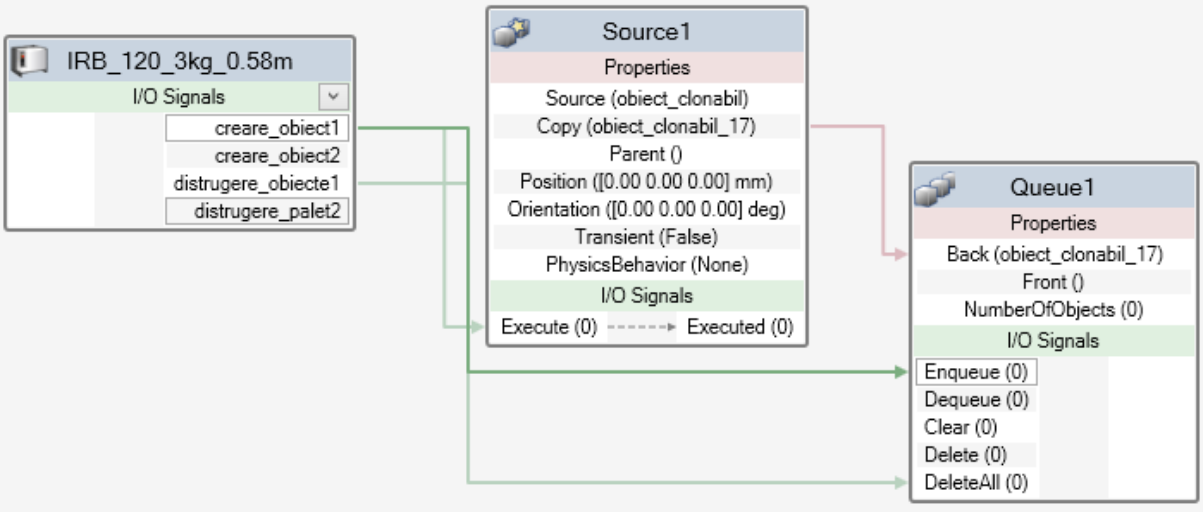


Fig. 6-8 – Secvențierea comenzilor pentru crearea unui set de piese ce ulterior va fi distrus folosind un obiect de tip coadă (*Queue*)

Lucrul cu componente care sunt în relația copil părinte presupune utilizarea unei componente active de tip Smart Component (Empty Smart Component)(Fig. 6-9), aceasta fiind singurul tip de componentă activă care poate lua rolul de părinte. Pașii următori presupun crearea paletului și a pieselor și atașarea lor. Trebuie avut mare grijă ca atașarea pieselor de palet să nu se facă pe același semnal de creare a lor întrucât în acest caz robotul va manipula întreg ansamblul. Astfel, atașarea (activarea semnalului de atașare) va fi realizată după procesul de depunere. Odată creat ansamblul, el poate fi accesat ca un singur obiect activ. Se va avea grijă la poziționarea pieselor relativ la palet și la componenta activă de tip părinte, aceasta din urmă fiind cea care va fi atașată efectorului terminal pentru manipulare. Structura și succesiunea de comenzi/fluxul logic sunt prezentate în figura de mai jos (Fig. 6-10). Spre deosebire de prima soluție în acest caz avem 5 componente active în plus față de componenta sistem responsabilă cu I/P controller: 2 componente de clonare (*Source*) piesă respectiv componentă Smart Component (*Empty Smart Component*), o componentă de setare a părintelui pieselor clonate, și o componentă de distrugere a clonei (*Sink*). Clonarea paletului se poate face

automat în componenta *Empty Smart Component* pe semnalul de creare din componenta *Source* aferentă.

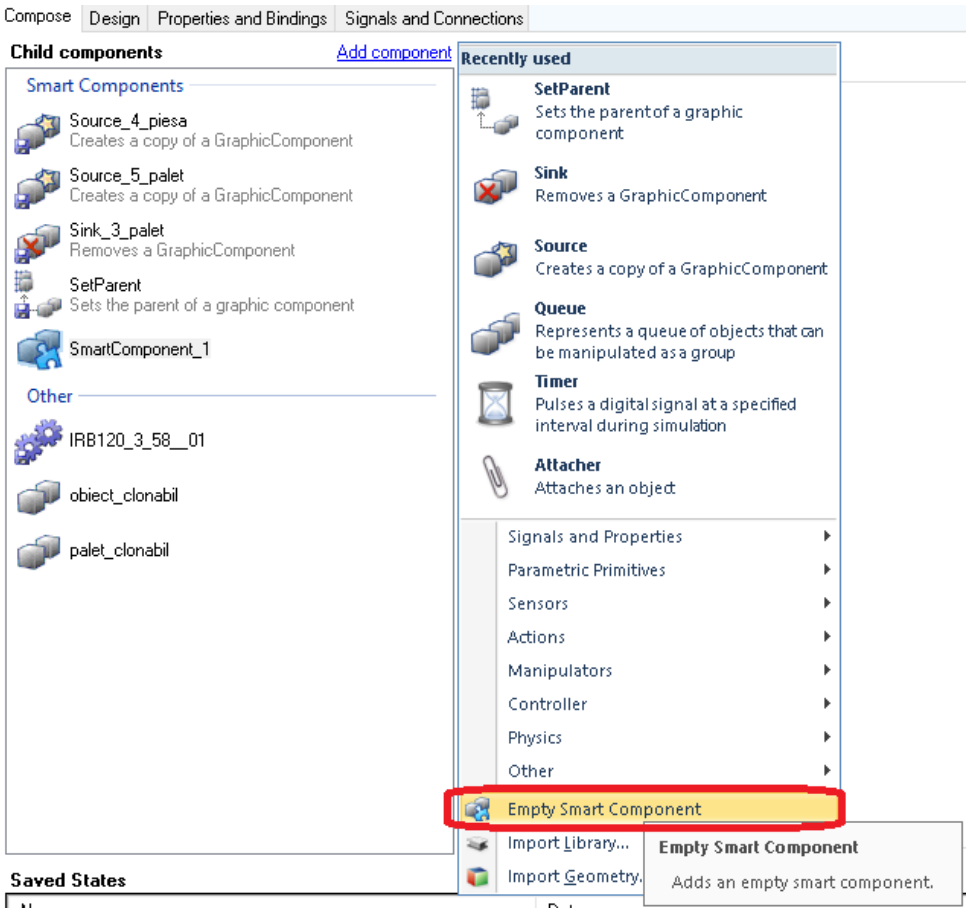


Fig. 6-9 – Adăugarea unei componente de tip Empty Smart Component

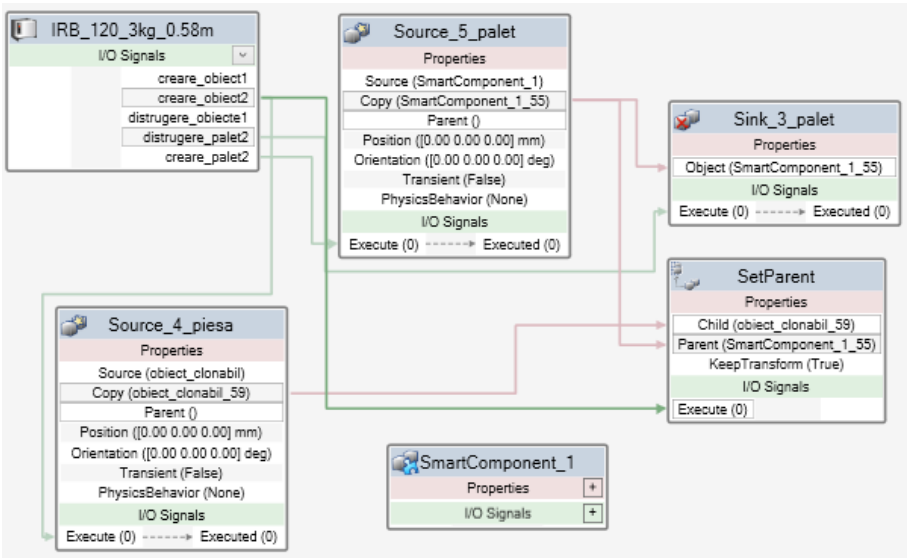


Fig. 6-10– Secvențierea comenzilor pentru crearea unui set de piese ce ulterior va fi distrus folosind un obiect de tip *Empty Smart Component*

1.2 Obiecte gravitaționale

Aplicația permite realizarea de simulări gravitaționale în modul de lucru continuu (en. Continous), existând 4 tipuri de comportamente pe care obiectele grafice din simulare le pot primi: inactiv, fix, cinematic și dinamic (Fig.6-11). În mod implicit robotul (segmentele brațului robotic) și efectoarele terminale au comportament cinematic (care nu poate fi schimbat), iar obiectele de tip componente geometrice sunt inactive. Tipul comportamentului obiectelor poate fi specificat fie ulterior creării (Fig. 6-11) fie la creare (Fig. 6-12). Cea de-a doua modalitate este utilă în cazul clonării de piese atunci când se dorește ca obiectele rezultate să aibă un comportament specific (ex.: se clonează niște piese care cad într-o cutie aranjându-se după gravitație, simulându-se astfel o scenă nestructurată).

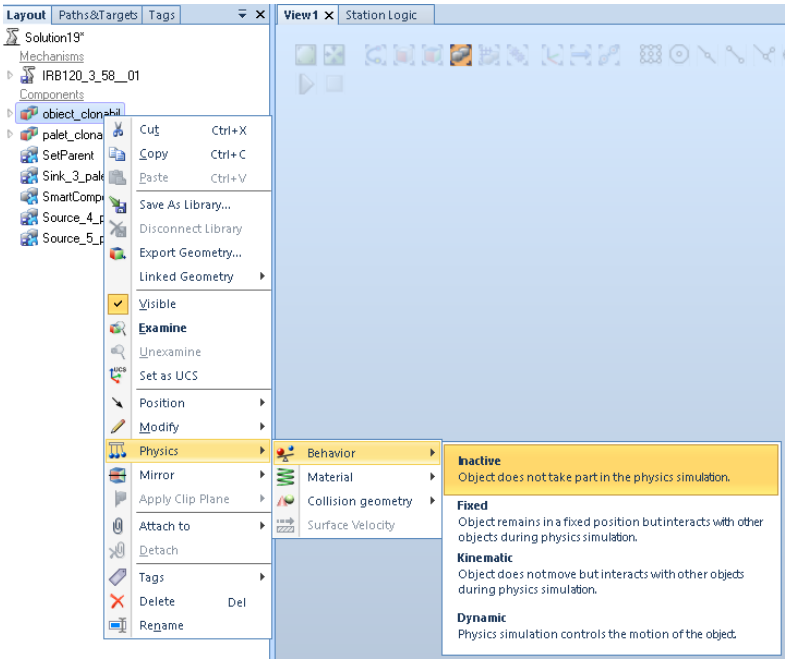


Fig. 6-11– Specificarea comportamentului fizic al unei componente ulterior creării

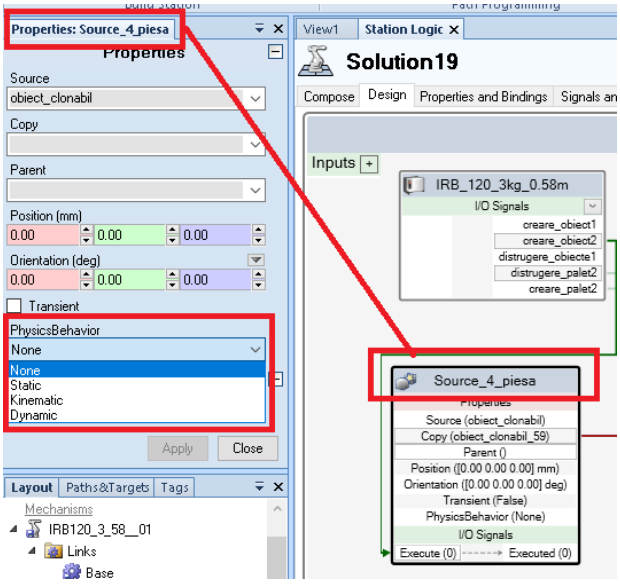


Fig. 6-12– Specificarea comportamentului fizic al unei componente la creare (în componenta activă de tip *Source*)

1.3 Mișcarea automată

Componentele active folosite pentru mișcarea de obiecte se regăsesc în secțiunea Manipulators (poza). Astfel, o componentă poate fi mutată liniar respectiv circular, cu o viteză fixă sau pe o distanță fixă. Pentru simularea mișcării mai multor piese pe conveior se poate folosi o combinație de *Empty Smart Component*, *Source*, *LinearMover* și *Sink*.

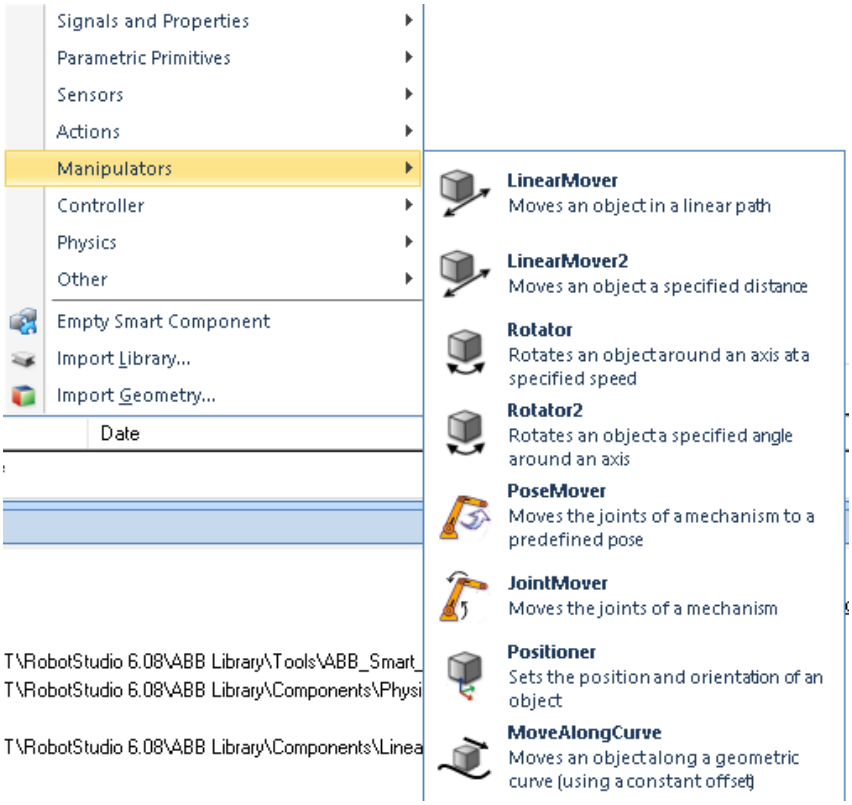


Fig. 6-13– Tipurile de componente active folosite la mișcarea unui obiect

1.4 Crearea, salvarea, exportarea și importarea de componente active

Crearea unei coloane luminoase:

Pas 1: se crează geometria aferentă componentei cu tot cu culori de bază; elementele componente se copiază ca și copii în componenta active de tip *Empty Smart Component* (Fig. 6-14)

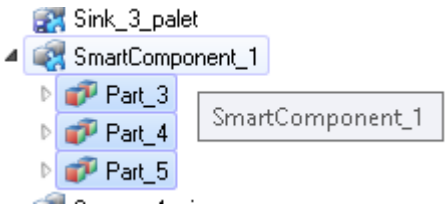


Fig. 6-14 – Popularea unei componente *Empty Smart Component* cu componente pasive

Pas 2: se folosește componenta active *Highlighter* pentru alterarea culorii elementelor individuale (Fig. 6-15)

Pas 3: se definesc semnale interne ale componentei *Empty Smart Component* (Fig. 6-13)

Pas 4: se atașează semnale digitale ale controllerului robot la semnalele interne ale componentei *Empty Smart Component* (Fig. 6-16)

Pas 4: componenta poate fi activată/testată din codul Rapid în modul de lucru automat prin setarea/resetarea valorii semnalelor aferente

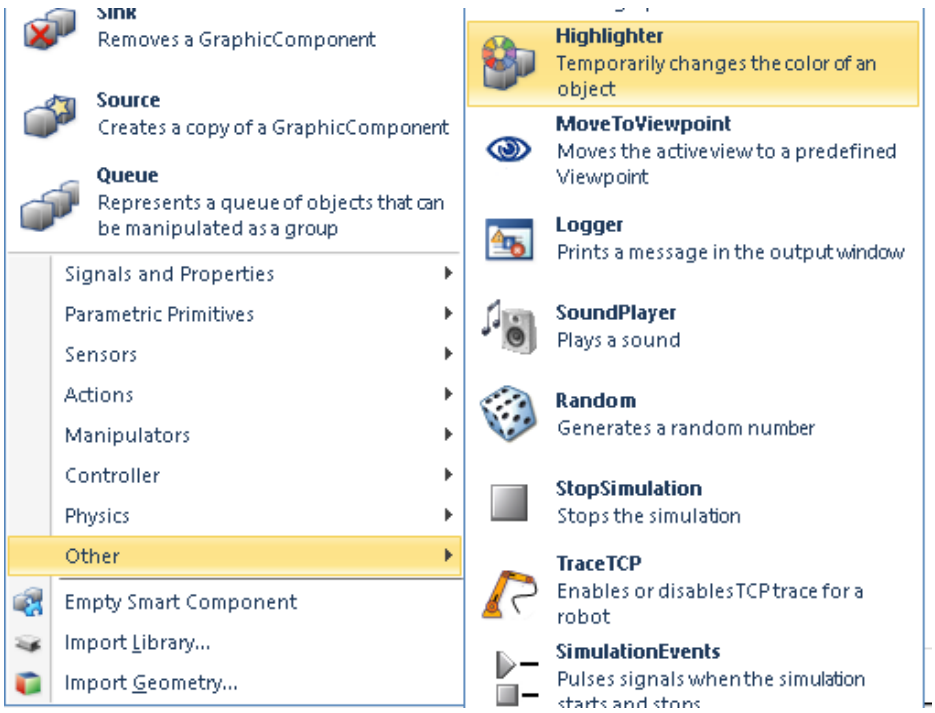


Fig. 6-15– Localizarea componentei *Highlighter*

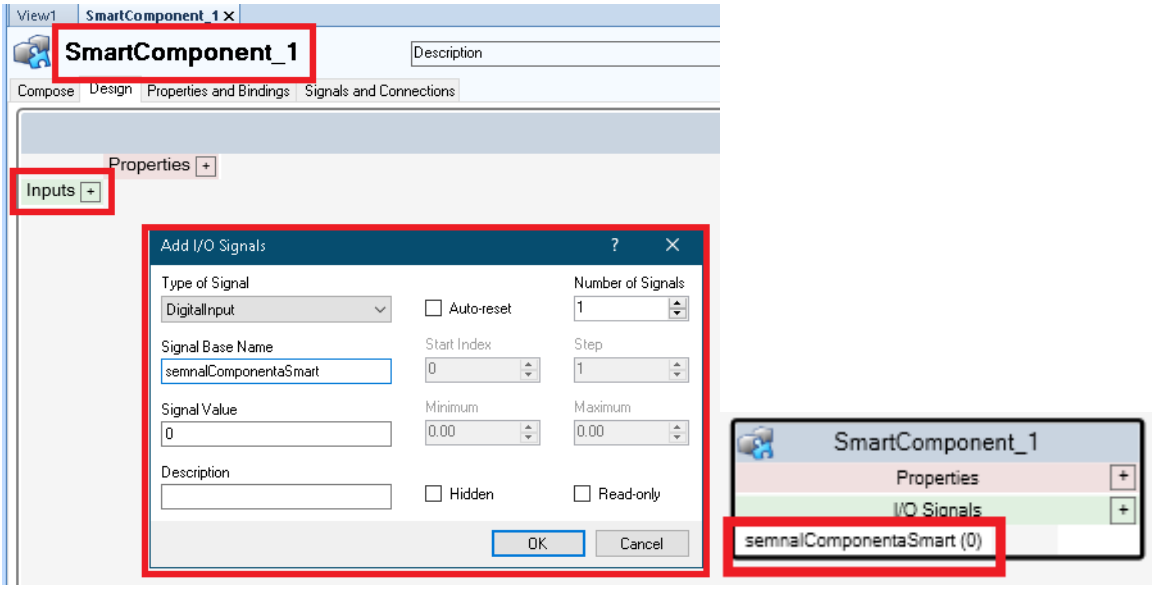


Fig. 6-16– Crearea semnalelor interne componentei EmptySmart Component (stânga) și utilizarea lor în rădăcina de componente active (dreapta)

1.5 Realizarea de entități de tip senzor prezență și măsurare distanță

În cazul în care vrem să avem o logică în spatele modului de lucru cu piesele (ex.: utilizarea vederii artificiale în ghidarea robotului, mișcare pe bandă, ș.a.) se poate folosi o componentă de tip senzor (Fig. 6-17). Atenție la I/O digitale aferente senzorului: *Active* – este o intrare care activează senzorul, *Sensor Out* este ieșirea senzorului (True – ceva a intrat în senzor).

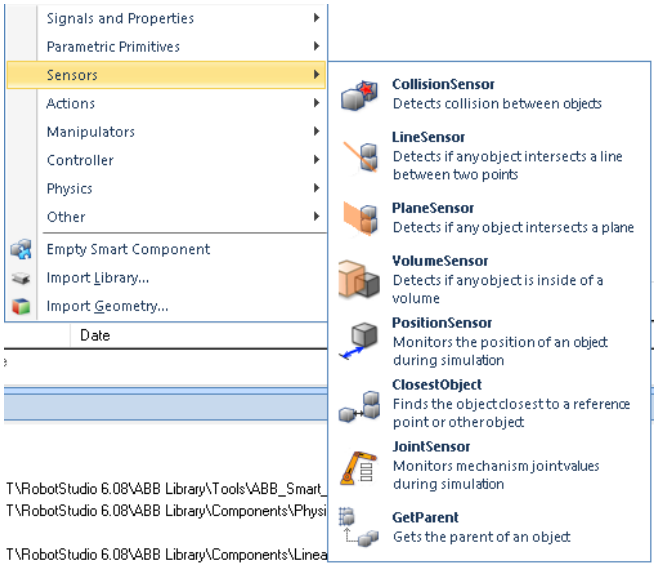


Fig. 6-17 – Tipuri de senzori disponibili în RobotStudio

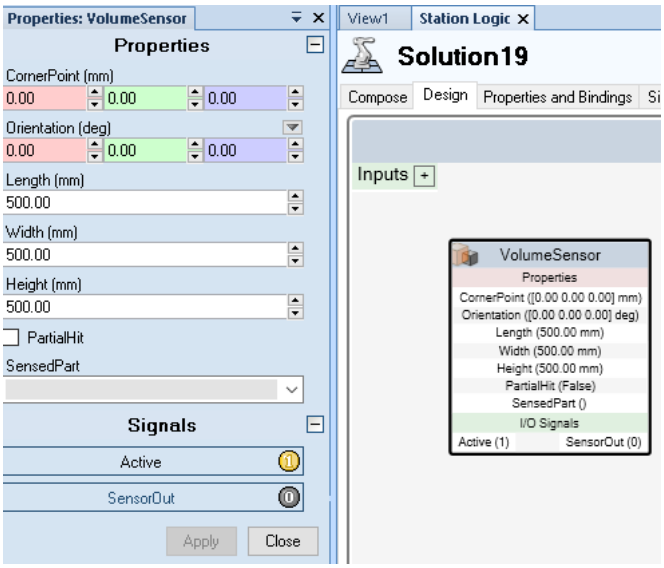


Fig. 6-18 – Elemente configurabile la senzorul de volum

1.6 Conexiuni SmartComponent – controller robot, accesul variabilelor controller în componentele active

Se folosește pentru citirea, respectiv modificarea unei variabile Rapid dintr-o componentă activă și viceversa (Fig. 6-19). Exemple de situații în care este necesar accesul la variabilele Rapid: a) pozitionarea și citirea poziției obiectelor în simulator, b) generarea de

semnale aleatoare care au nevoie de instrumente matematice neexistente în componentele active disponibile, În mod uzual, la fel ca și celelalte componente active și în această situație configurațiile din modul rulare se execută din meniul *Design* prin realizarea fluxului informațional aferent.

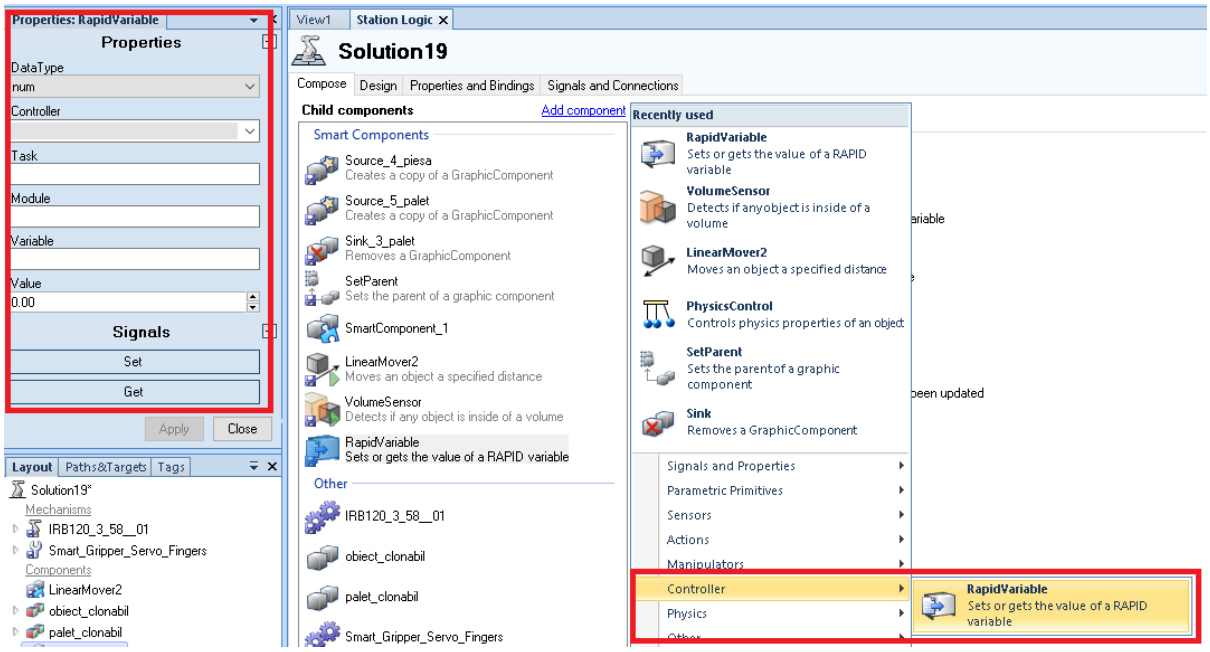


Fig. 6-19 – Adăugarea și configurarea unei variabile Rapid ca și componentă activă

2 Exerciții

Se cere:

- să se realizeze o piesă de 20/20/20mm care se va clona de 12 ori într-o stivă bidimensională pe X și Y global
- să se realizeze traiectoria robot care paletizează piesele din stiva de mai sus într-o stiva tridimensionala (3 pe X, 3 pe Y, 2 pe Z).
- OBS: se vor paletiza atâtea piese cât se poate

- Piesele adiacente pe axa de completare vor fi rotite la 90°

Elemente suplimentare:

- Programul va rula în mod continuu, obiectele fiind generate și distruse în mod automat
- Starea postului de lucru va fi indicată vizual printr-o componentă activă de tip coloană luminoasă (roșu – defect, verde – așteaptă semnal de start, galben – robot în lucru)

3 Întrebări

Cum se definește un sistem de coordonate folosind doar partea de poziție a coordonatelor carteziene?

Capitol 7: Sisteme multirobot

Cuprins

1 Schimbarea opțiunilor implicite ale controllerului robot 111
1.1 Multitasking – adăugare de task separat 112
1.2 Urmărire conveior 114
1.3 Lucrul cu echipamente de rețea 116
2 Sisteme multirobot 118
2.1 Creare sistem multirobot 119
2.2 Cadre de referință 121
3 Axe externe de mișcare 122
3.1 Implementare mecanism poziționare 122
3.2 Extensia unui mecanism de poziționare la axa de mișcare și utilizarea acesteia ca suport robot 123
3.3 Utilizarea de axe externe de mișcare ca suport robot/piesă (caz 1: mecanism robot – axă externă) 125
4 Sincronizare și mișcare coordonată în sistemele multirobot 126
4.1 Utilizarea semnalelor digitale pentru sincronizarea mișcării sistemelor multirobot 126
4.2 Mișcare coordonată 126
4.3 Utilizarea de axe externe de mișcare ca suport piesă (caz 2) 128
5 Exerciții 128
6 Întrebări 129

1 Schimbarea opțiunilor implicite ale controllerului robot

Aplicația RobotStudio permite configurarea opțiunilor aferente sistemului de operare robot (RobotWare) astfel încât să se extindă funcționalitatea sa (Fig. 7-1). În mediul real extinderea funcționalității controllerului presupune achiziția de licențe, respectiv dispozitive electronice dedicate (ex.: controller de mișcare pentru axe externe). În simulator extensia opțiunilor implicite presupune doar adăugarea unor module la controllerul virtual, module care permit realizarea de noi funcționalități precum: operarea cu taskuri multiple, sincronizare, lucrul cu modulul de comunicație pe rețea, ș.a. Se recomandă ca modificarea opțiunilor controllerului să se realizeze la crearea proiectului conform cerințelor din proces (ex.: sistem multitasking, comunicație pe rețea, etc.) deoarece ajustarea ulterioară putând aduce modificări proiectului în curs de realizare.

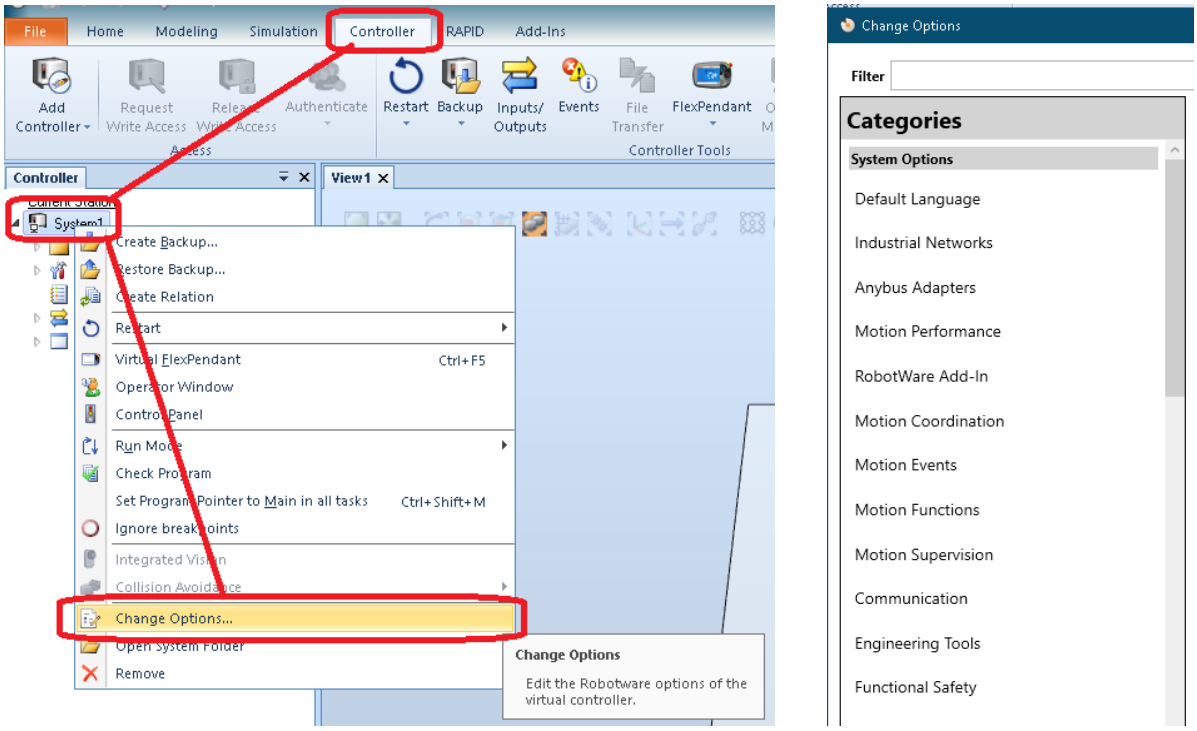


Fig. 7-1 – Accesul la modulul de configurare a opțiunilor controllerului robot

1.1 Multitasking – adăugare de task separat

Scopul opțiunii Multitasking este de a putea executa mai multe programe în paralel. Printre exemplele reale care necesită rularea în paralel a mai multor task-uri se numără: a) supervizarea continuă a unor semnale (controllerul robot poate lua sarcina unui PLC, dar nu se compară dpdv al timpilor de răspuns), b) lucrul cu modulul de comandă manuală unde este nevoie de task-uri paralele pentru ca interacțiunea cu operatorul să nu blocheze sistemul, sau c) controlul și activarea/dezactivarea echipamentelor externe. Pentru adăugarea separată a opțiunii Multitasking se procedează ca în figura de mai jos (Fig. 7-2).

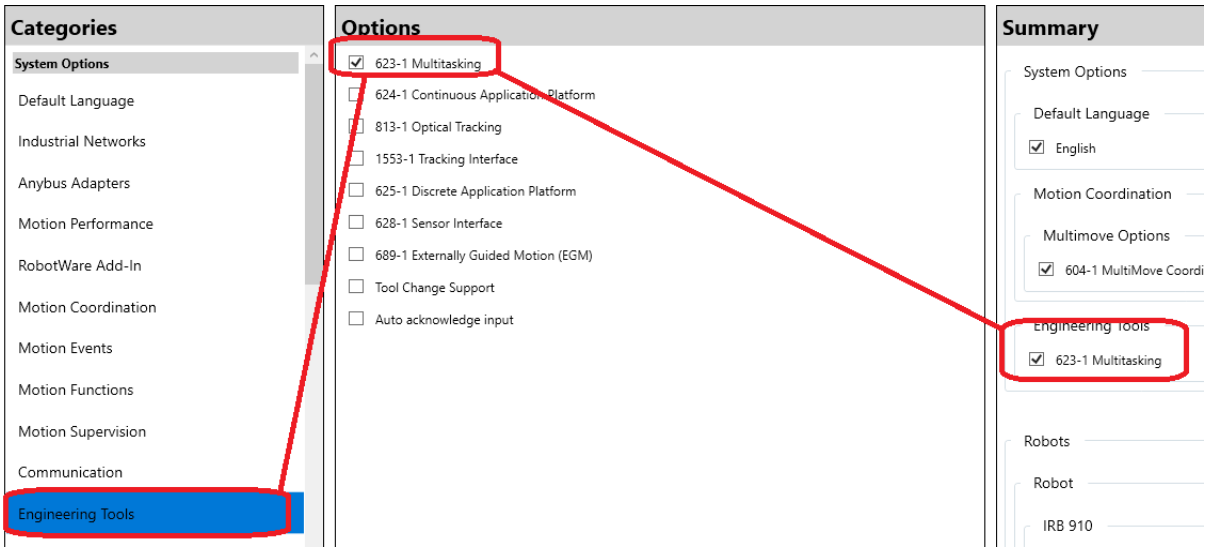


Fig. 7-2 – Adăugarea opțiunii de multitasking pentru controllerul virtual

Conform procedurii de mai sus pot fi definite un număr de 20 de task-uri care să ruleze în paralel. Fiecare task este compus dintr-un program și un set de module sistem locale task-ului (Fig. 7-3).Variabilele și constantele sunt locale task-ului, dar datele de tip persistent nu sunt locale. Fiecare task are propriile proceduri de tratare a erorilor, iar rutinele de procesare sunt declanșate doar de stările proprii ale task-ului.

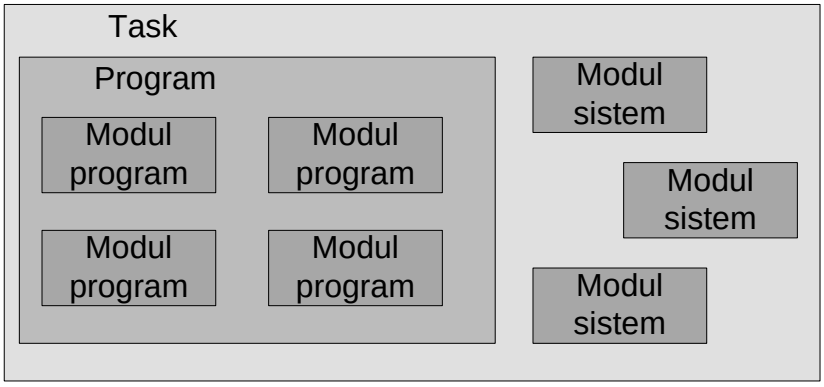


Fig. 7-3 – Structura unui task Rapid

Adăugarea unui task nou se face conform figurii de mai jos, realizându-se ulterior și configurațiile din partea dreaptă (Fig. 7-4).

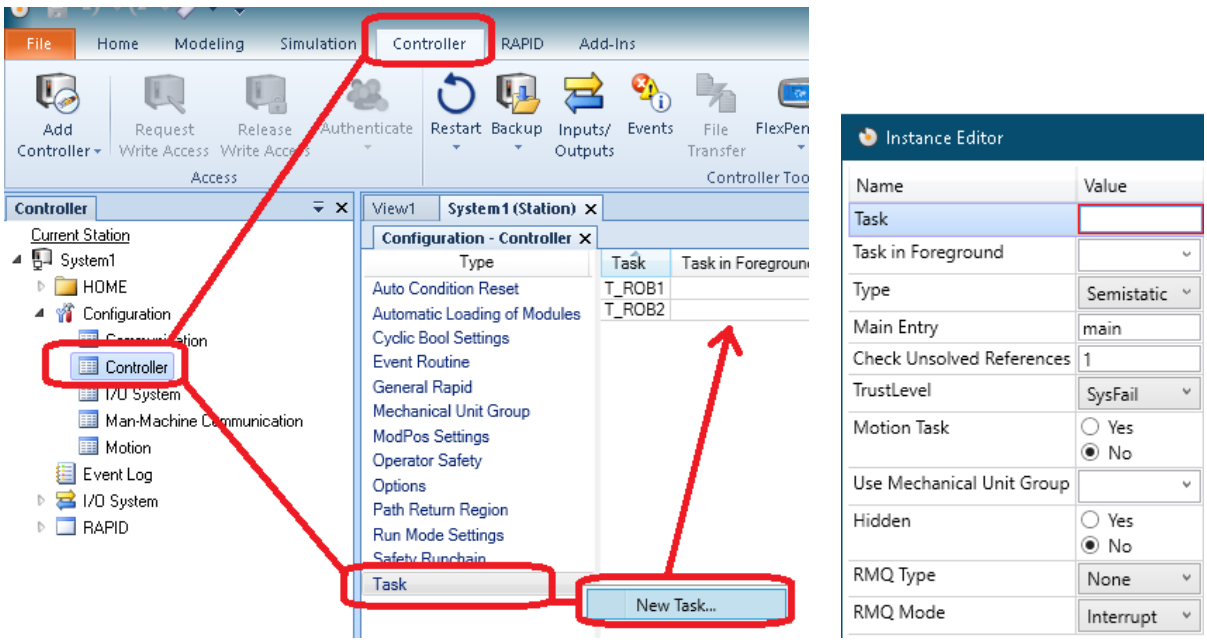


Fig. 7-4 – Adăugarea unui task nou pe controllerul virtual și configurarea opțiunilor de rulare

La modul minimal trebuie configurate următoarele elemente:

a) Numele task-ului – acesta trebuie să fie unic cu respectarea convențiilor pentru identificare sistem (ex.: să nu aibă același nume cu o unitate mecanică). Editarea ulterioară a numelui este echivalentă cu ștergerea task-ului și adăugarea unului nou, rezultatul fiind pierderea a tot ce s-a scris în acel task, precum linii de cod.
b) Tipul task-ului – există trei tipuri de task-uri (NORMAL, STATIC și SEMISTATIC) acestea controlând comportarea sistemului la start/restart în felul următor: NORMAL – pornire manuală la start și stop; reacționează la butonul de

urgență; STATIC – la restartare sistem pornește de unde a rămas și în mod implicit nu reacționează la oprirea de urgență; SEMISTATIC – la restartare pornește de la început și nu reacționează la apăsarea butonului de urgență. Pentru simulări uzuale (ex.: așteptare) se recomandă folosirea tipului NORMAL.

După creare se pot adăuga și modulele program cu tot cu programele necesare care ulterior vor fi configurate ca puncte de intrare (Fig. 7-5).

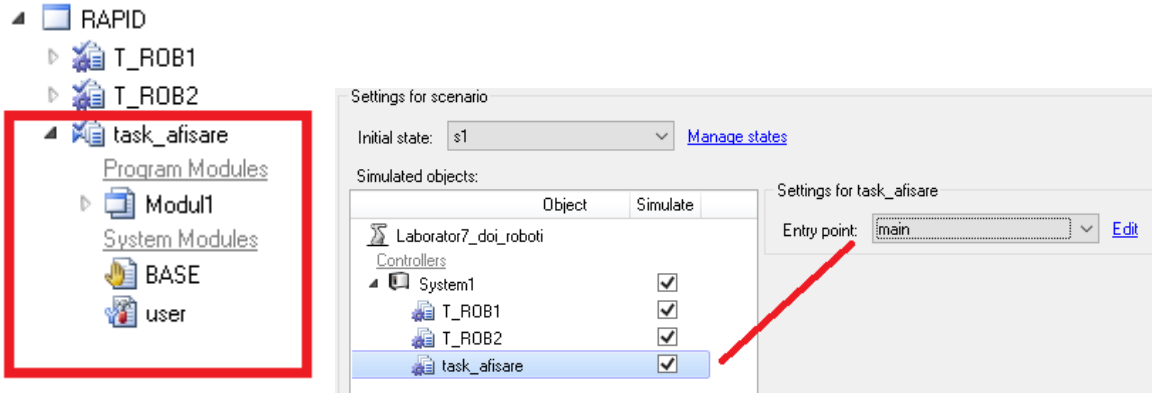


Fig. 7-5 – Adăugarea codului aferent unui task și configurarea punctului de intrare

1.2 Urmărire conveior

Pentru realizarea unui proiect în care se urmăresc piese dispuse pe conveior trebuie adăugate următoarele opțiuni: urmărire conveior(Fig. 7-6).

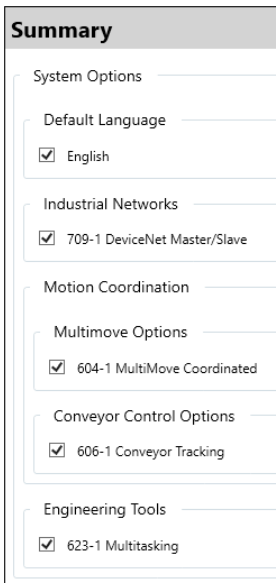


Fig. 7-6 – Opțiuni controller pentru urmărirea de piese în mișcare (pe conveior)

Realizarea unei instalații pentru urmărirea conveiorului se face utilizând următoarele componente (Fig. 7-7): Switch de sincronizare (A), Coveior (B), Encoder (C), Interfața encoder-ului (D), Controller IRC5 (robot industrial) (E).

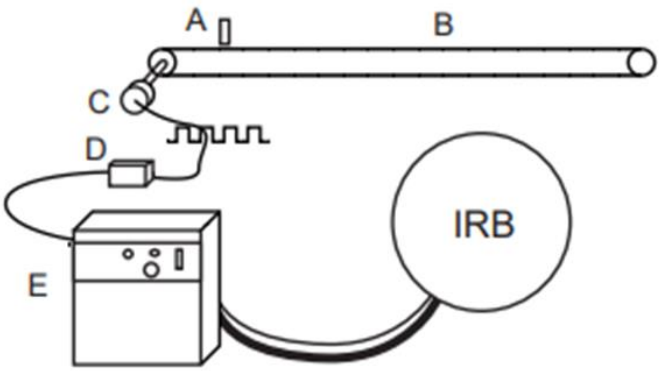


Fig. 7-7 – Structură de control pentru urmărire coveior folosind un controller robot

Pentru crearea unui obiect de tip conveior cu tot cu funcționalitățile aferente (clonare piesă, reglare viteza, urmărire piese, sincronizare robot-conveior) se folosește opțiunea *Create Conveyor* din meniul *Modelling* (Fig. 7-8).

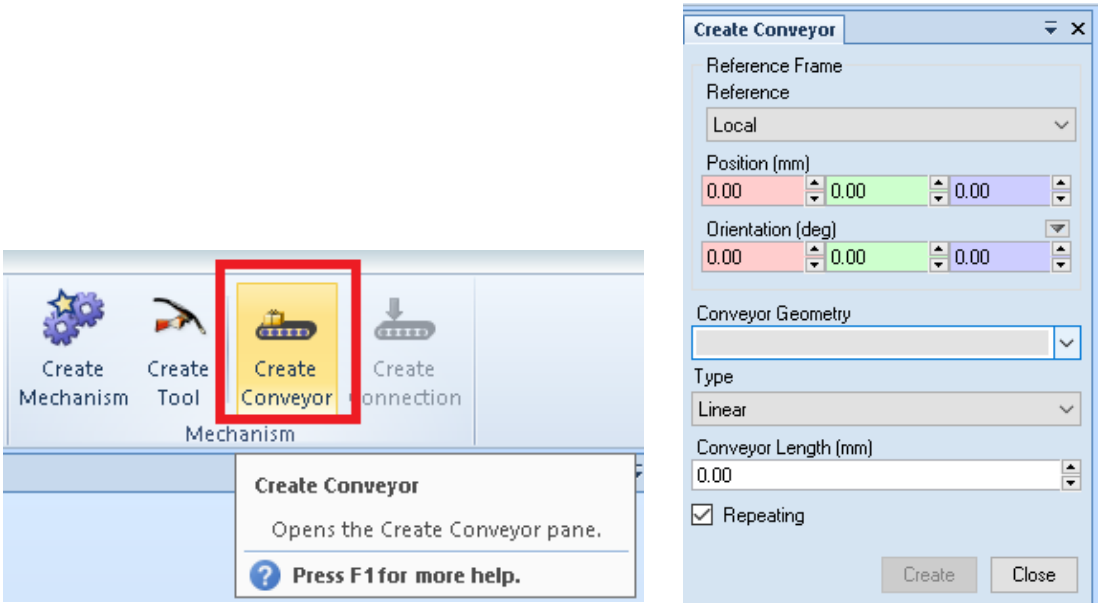


Fig. 7-8 – Crearea unei structuri de tip conveior și configurarea parametrilor aferenți

Mai jos sunt prezentate o serie de caracteristici ale sistemului robot-conveior în configurația de urmărire:

a) *Precizia* – în modul automat, la viteza conveiorului constantă de 150 mm/s, punctul condus al robotului (TCP) va rămâne la +/- 2 mm de traseu. Această distanță este menținută indiferent dacă conveiorul este în mișcare sau nu. Acest lucru este valabil atâta timp cât robotul se află înăuntrul limitelor sale dinamice cu mișcarea conveiorului. Această cifră depinde de calibrarea robotului și a conveiorului și este aplicabilă pentru urmărirea liniară (Fig. 7-9)
b) *Obiecte în așteptare* – pentru opțiunea ”urmărirea conveiorului”, se menține o coadă de până la 254 de obiecte care au trecut de comutatorul de sincronizare. Pentru opțiunea de indexare a conveiorului de control, coada poate conține până la 100 de obiecte.

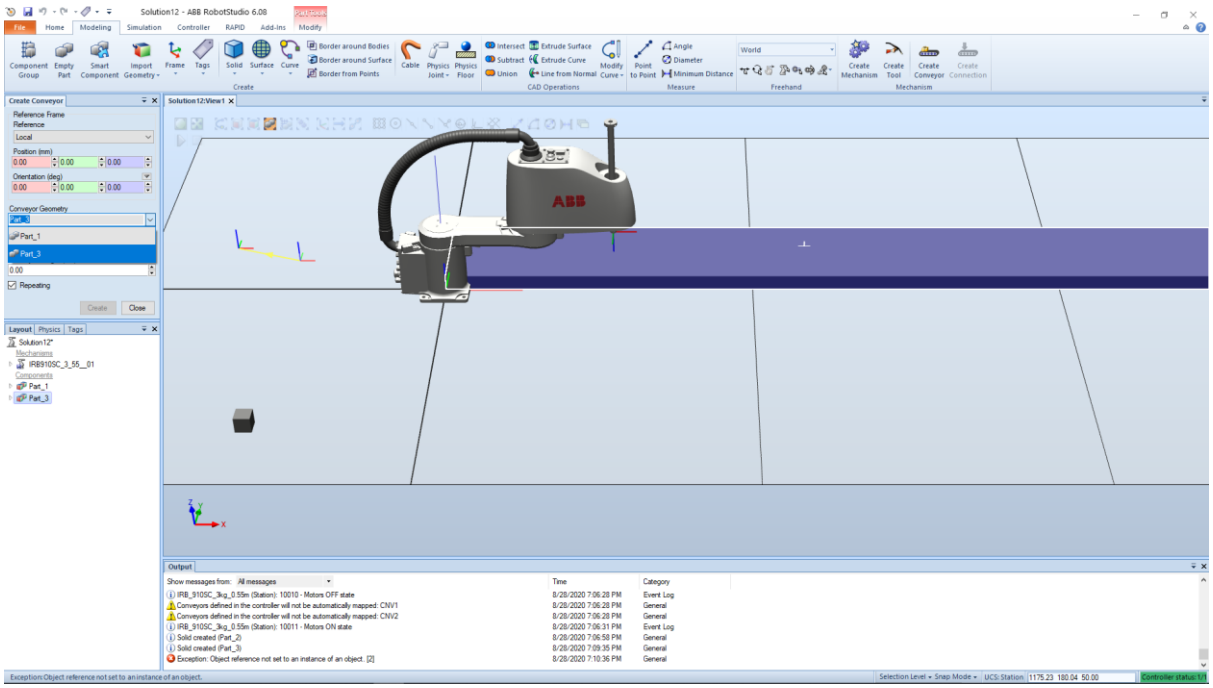


Fig. 7-9 – Transformarea unui obiect solid în conveior

a) *Fereastra de start* – un program poate alege să aștepte un obiect care se află într-o fereastră trecută de punctul de plecare sau se așteaptă ca un obiect să treacă o anumită distanță sau să ia imediat primul obiect din coada de urmărire a obiectelor. Obiectele care depășesc fereastra de pornire sunt omise automat.
b) *Accesarea datelor conveiorului printr-un program RAPID* – un program RAPID are acces la numărul de obiecte în așteptare, poziția curentă și viteza conveiorului. Un program RAPID poate elimina primul obiect din coada de așteptare sau toate obiectele din coadă.
c) *Distanța maximă* – se poate specifica o distanță maximă de urmărire pentru a opri robotul de la urmărire, în afara zonei de lucru.
d) Urmărirea piesei pe conveior – dacă robotul este montat pe o linie liniară, atunci sistemul poate fi configurat astfel încât pista va urma transportorul și va menține poziția relativă la conveior. Viteza TCP în raport cu obiectul de lucru de pe conveior va fi în continuare programată.

1.3 Lucrul cu echipamente de rețea

Necesitate: realizarea de jurnale (en.: log) avansate despre modalitatea de lucru a echipamentelor se poate utiliza o conexiune la rețea. Prin această conexiune se poate monitoriza sistemul multirobot dintr-o aplicație externă care înregistrează evenimentele generate de controllerul virtual. Pentru lucrul cu echipamentele de rețea trebuie adăugată opțiunea interfață calculator (Fig. 7-10). În mod implicit echipamentele fizice dispun de o astfel de conexiune, fiind folosită în mod uzual pentru legături la sisteme de vedere artificială.

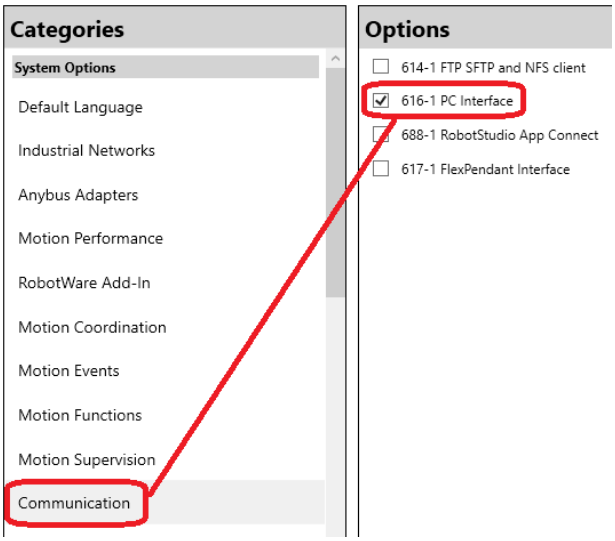
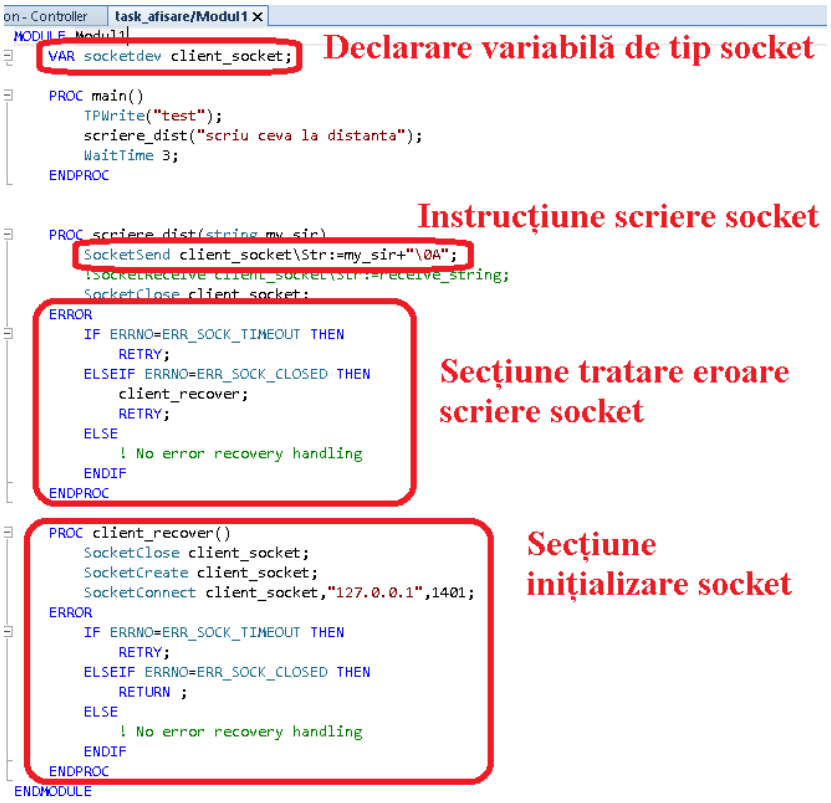


Fig. 7-10 – Opțiuni controller pentru adăugarea unei interfețe de rețea

Mai jos este prezentată o figură cu o captură de ecran care detaliază modul de realizare al unui client ETHERNET (Fig. 7-11). În mod implicit robotul este client, responsabil de inițierea comunicației fiind fie echipamentul de monitorizare (PC) fie dispozitivul master peste postul robotizat (PLC). Cu toate acestea echipamentul este capabil și de realizarea unui server TCP. Pentru primul caz este prezentat în figura de mai jos un exemplu de realizare a unui client TCP (poza). Acest cod poate fi testat folosind aplicația Hercules (Fig. 7-12) (https://www.hw-group.com/software/hercules-setup-utility) capabilă de realizarea de conexiuni seriale și TCP/UDP atât în mod client cât și în mod server acolo unde este cazul.




```
MODULE Modul1
    VAR socketdev client_socket;

    PROC main()
        TPWrite("test");
        scriere_dist("scriu ceva la distanta");
        WaitTime 3;
    ENDPROC

    PROC scriere_dist(string my_sir)
        SocketSend client_socket\Str:=my_sir+"\0A";
        !SocketReceive client_socket\Str:=receive_string;
        SocketClose client_socket;
    ERROR
        IF ERRNO=ERR_SOCK_TIMEOUT THEN
            RETRY;
        ELSEIF ERRNO=ERR_SOCK_CLOSED THEN
            client_recover;
            RETRY;
        ELSE
            ! No error recovery handling
        ENDIF
    ENDPROC

    PROC client_recover()
        SocketClose client_socket;
        SocketCreate client_socket;
        SocketConnect client_socket,"127.0.0.1",1401;
    ERROR
        IF ERRNO=ERR_SOCK_TIMEOUT THEN
            RETRY;
        ELSEIF ERRNO=ERR_SOCK_CLOSED THEN
            RETURN ;
        ELSE
            ! No error recovery handling
        ENDIF
    ENDPROC
ENDMODULE
```

Fig. 7-11 – Instrucțiuni Rapid pentru realizarea unui program de tip client

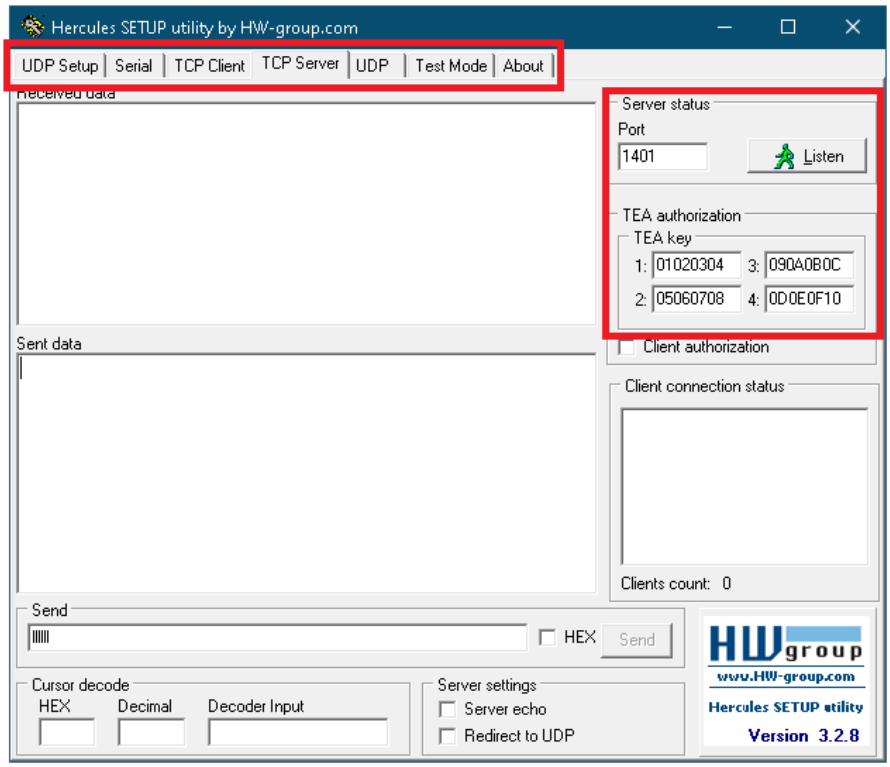


Fig. 7-12 – Captură de ecran aplicație Hercules

2 Sisteme multirobot

Opțiunea MultiMove permite unui controller să gestioneze mai mulți roboți. Acest lucru permite generarea de economii cu achiziția echipamentelor hardware (2 brațe robotice – 1 controller în loc de 2) dar și o coordonare avansată între diferiți roboți și alte mecanisme externe de tip mecanism de poziționare sau axă externă. Exemple de astfel de situații includ: a) mai mulți roboți pot opera asupra unui obiect în mișcare, b) un robot poate mișca sistemul de coordonate în care cel de-al doilea lucrează, c) mai mulți roboți pot coopera pentru a manipula (ridica) obiecte mai grele decât sarcina individuală de lucru a fiecăruia. Adăugarea opțiunii MultiMove presupune adăugarea implicită a opțiunii Multitasking.

Legat de sistemele multirobot avem următoarea terminologie:

- Coordonare – Un robot care este coordonat cu un sistem de coordonate va urmări mișcările acelui sistem de coordonate
- Sincronizare – Mișcări (sau așteptări) simultane. Sincronizarea se referă la o verificare în timp nu în spațiu (sisteme de coordonate). Tot sincronizare se va numi și folosirea I/O digitale pentru a sincroniza accesul la spații de lucru comune.

Controllerul robot permite adăugarea de 2 opțiuni MultiMove care permit implementarea de mișcări independente și coordonate (sau semi-coordonate).

Mișcarea multirobot independentă se referă la programe robot care rulează pe taskuri diferite fără coordonare sau sincronizare. Uneori, chiar dacă mișcările nu trebuie coordonate, traiectoriile robot pot avea dependențe. De exemplu, dacă un robot depune un obiect pe care îl va ridica un al doilea robot, primul robot trebuie să termine acțiunea de depunere a obiectului înainte ca al doilea robot să-l poată apuca. Acest tip de interacțiune se poate face prin instrucțiuni de așteptare, I/O digitale respectiv variabile persistente pentru semnalizare.

Mișcarea multirobot semi-coordonată se referă la procese cu mai mulți roboți care lucrează în același sistem de coordonate (en.: workobject) fără mișcări sincronizate, atâta timp cât obiectul de lucru nu se mișcă. Un element de poziționare poate muta obiectul de lucru atunci când roboții nu sunt coordonați cu acesta, iar roboții pot fi coordonați cu obiectul de lucru atunci când acesta nu se mișcă. Comutarea între deplasarea obiectului și coordonarea roboților se numește mișcare semi-coordonată. Mișcările semi-coordonate necesită o anumită sincronizare între traiectoriile robot (ex.: WaitSyncTask). Elementul de poziționare trebuie să știe când poate fi repoziționat obiectul de lucru, iar roboții trebuie să știe când pot lucra asupra obiectului de lucru. Cu toate acestea, nu este necesar ca fiecare instrucțiune de repoziționare să fie sincronizată. Avantajul este că fiecare robot poate lucra independent pe sistemul de coordonate comun. Dacă roboți diferiți efectuează sarcini foarte diferite, acest lucru poate economisi timp de ciclu în comparație cu situația în care se sincronizează toate mișcările robotului.

Mișcarea multirobot coordonată se referă la procese în care mai mulți roboți operează pe același sistem de coordonate în mișcare. Echipamentul de poziționare sau robotul care deține sistemul de coordonate de lucru și roboții care lucrează cu obiectul de lucru trebuie să aibă mișcări sincronizate. Aceasta înseamnă că traiectoriile roboților care gestionează fiecare câte o unitate mecanică, execută instrucțiuni de mișcare simultane. Modul de mișcare sincronizat este pornit prin executarea unei instrucțiuni *SyncMoveOn* în fiecare traiectorie robot. Modul de mișcare sincronizată este încheiat prin executarea unei instrucțiuni *SyncMoveOff* în fiecare traiectorie robot. Numărul instrucțiunilor de mișcare executate între *SyncMoveOn* și *SyncMoveOff* trebuie să fie același pentru toate traiectoriile implicate. Mișcările sincronizate coordonate economisesc de obicei timp de ciclu deoarece roboții nu trebuie să aștepte în timp ce obiectul de lucru este mutat. De asemenea, permite roboților să coopereze în moduri care altfel ar fi dificil sau imposibil de realizat.

2.1 Creare sistem multirobot

Pentru a se crea un sistem multirobot se va crea un proiect de tipul Solution with Empty Station și se vor adăuga elementele sistemului(roboți industriali, mecanisme, unelte etc).

Crearea sistemului se va realiza folosind meniul Home->Robot System->From Layout (Fig. 7-13).

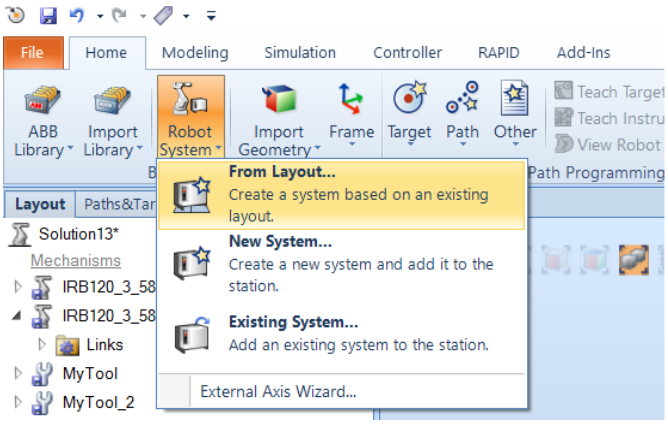


Fig. 7-13 – Metoda de acces pentru crearea sistemului

Din fereastra de creare se realizează setări ale proiectului (Fig. 7-14):

a) specifică numele, locația de salvare și versiunea de RobotWare dorită;
b) se selectează mecanismele care vor fi folosite(roboții și alte mecanisme);
c) configurarea task-urilor;
d) alegerea opțiunilor și cadrului de referință.

Un element important în crearea sistemului este Configurarea pe task-uri și anume gruparea roboților cu task-urile corespunzătoare. În Fig.14 se prezintă o configurare posibilă.

Alegerea opțiunilor sistemului este realizată automat cu elementele necesare pentru sisteme multimove și anume selectarea opțiunilor:604-1 MuliMove Coordinated și 623-1 Multitasking. Se pot adăuga și elemente adiționale precum Collision Detection,Conveyor Tracking, etc. care permit realizarea unor proiecte cu cerințe complexe.

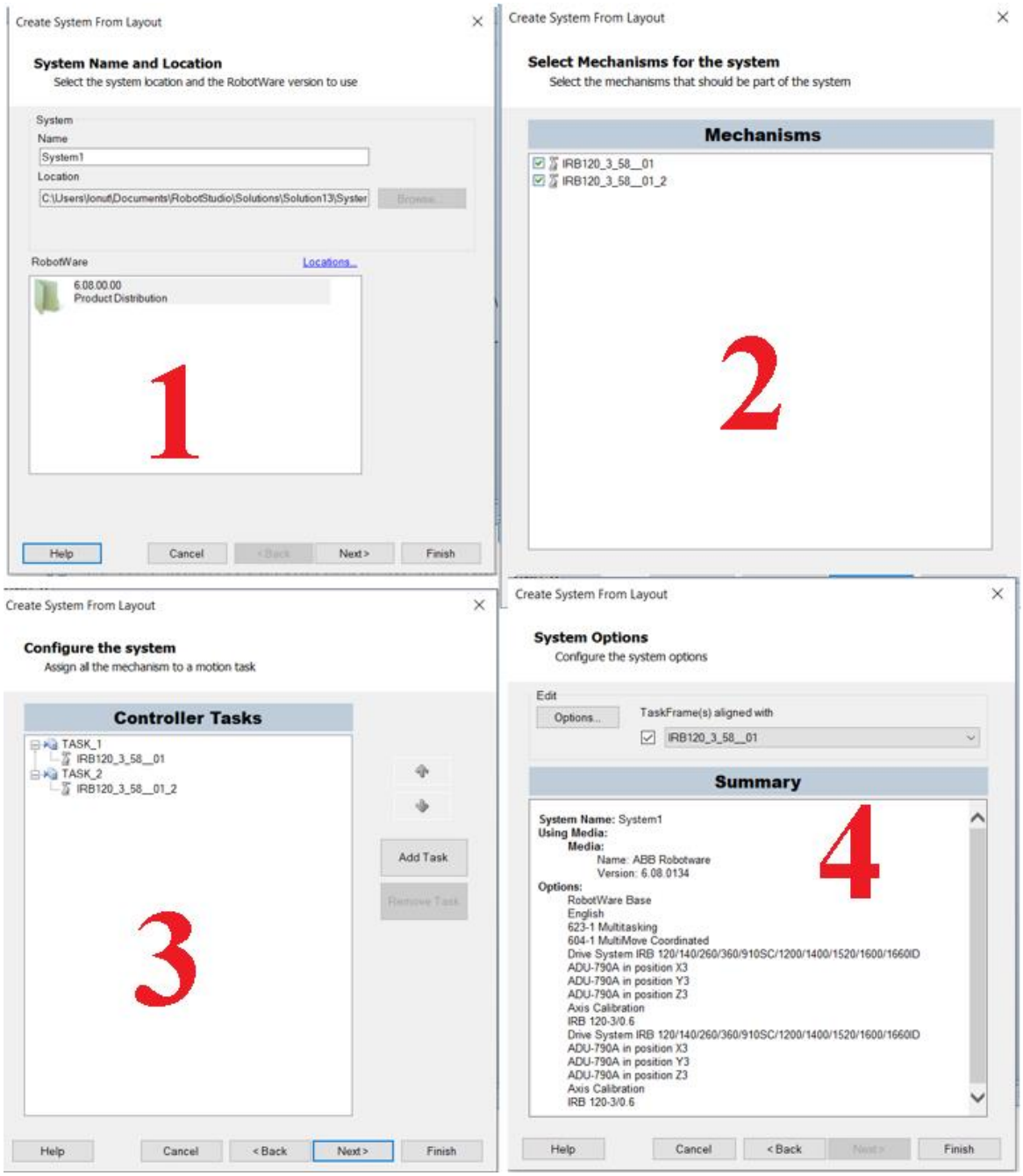


Fig. 7-14 – Configurarea proiectului

Sistemul creat este compus din mai multe task-uri(fiecare task are un robot industrial asociat); un astfel de sistem se poate observa în Fig. 7-14.

Elementele constructive ale proiectului precum învățarea punctelor, a traiectoriilor robot, crearea workobject-urilor, definirea transformărilor uneltelor se realizează la nivel de task.

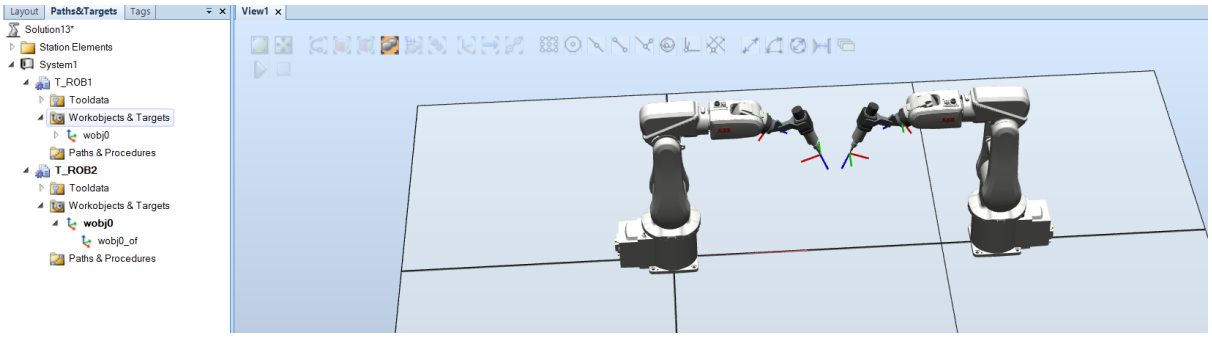


Fig. 7-15 – Organizarea sistemului

În etapa de configurare a simulării se selectează task-urile ce se doresc a fi simulate; o astfel de configurare se poate observa în Fig. 7-16.

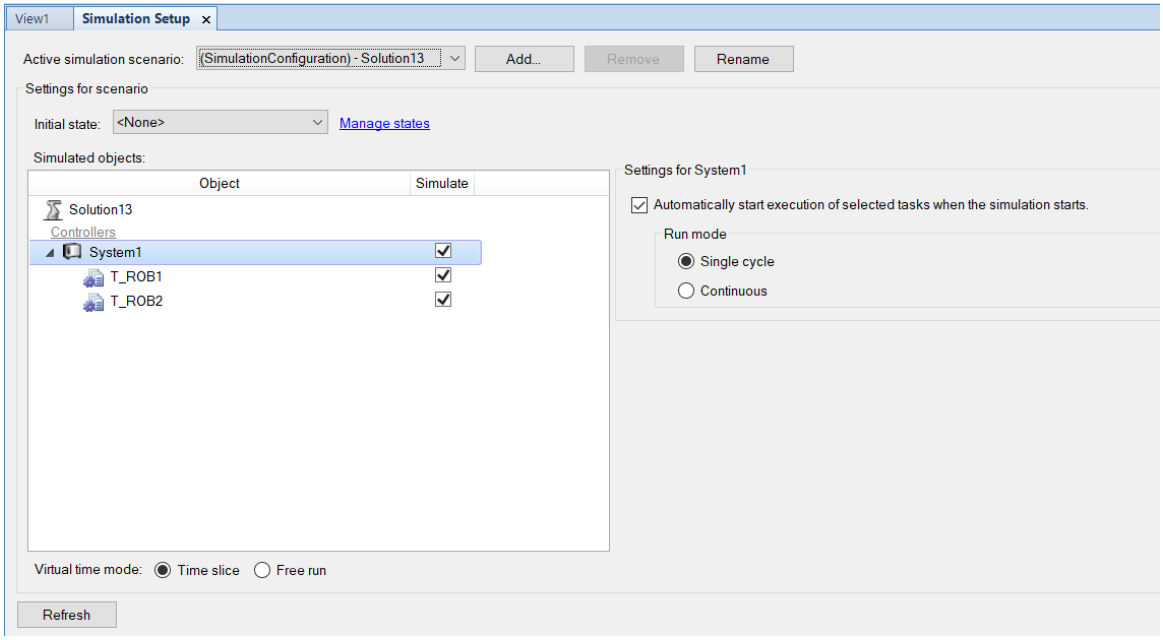


Fig. 7-16 – Configurarea simulării pentru un sistem multirobot

Realizarea elementelor de siguranță(evitarea coliziunilor) precum și elementele pentru sincronizare trebuie să se realizeze la nivel de proiectare(programare) de către utilizator având la dispoziție instrucțiuni de tip wait și posibilitatea folosirii intrărilor/ieșirilor digitale pentru a realiza sincronizarea dorită a traiectoriilor robot.

2.2 Cadre de referință

În sistemele multirobot controlul se face relativ la un sistem de referință unic, sistemul sarcinilor (en.: *Task Frame*). În mod suplimentar fiecare braț robotic are un sistem de coordonate propriu relativ la care se învață punctele din spațiul de lucru. Accesarea acestor

sisteme se face din meniul *Controller -> Edit System* (pentru vizualizarea și editarea sistemelor de coordonate individuale ale roboților), respectiv *Controller -> Task Frames* pentru editarea sistemului de coordonate al sarcinilor (Fig. 7-17).

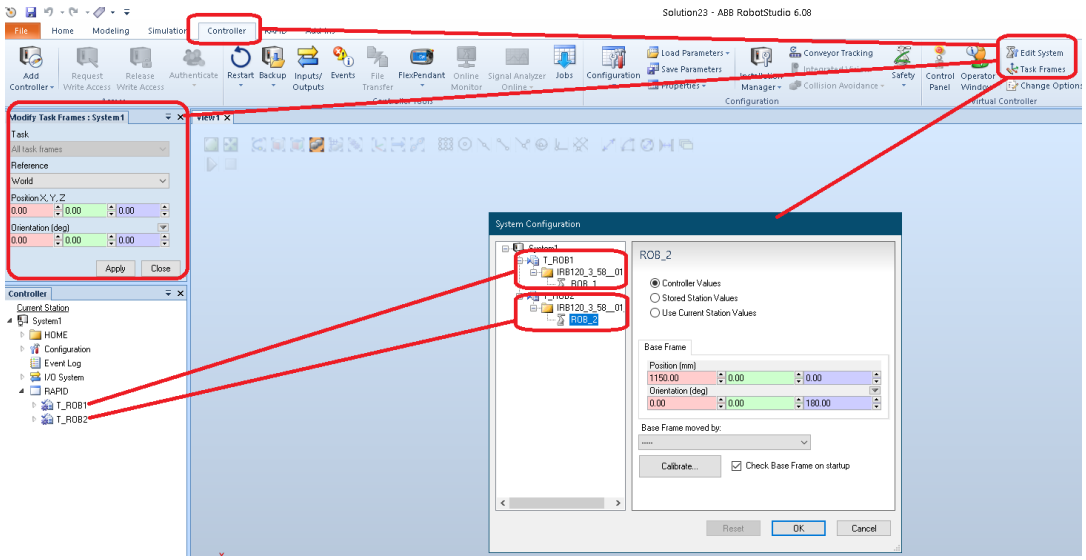


Fig. 7-17 – Modalitatea de acces la sistemele de coordonate global și robot

3 Axe externe de mișcare

Pentru extensia spațiului de lucru și a dexterității unui robot industrial în raport cu spațiul de lucru (piesa deservită) există două posibilități: a) adăugarea de axe de mișcare suplimentare robotului respectiv b) realizarea de mecanisme cu flexibilitate sporită de prezentare a pieselor. Ambele cazuri sunt realizate prin folosirea unor axe externe de mișcare, axe care pot avea un număr finit de poziții sau o mișcare continuă. În mod uzual axele de mișcare cu poziții limitate se numesc mecanisme de prezentare sau poziționare în funcție de unde sunt poziționate (la capătul sau la baza robotului/atașate robotului sau robotul atașat de ele). Avantajul mecanismelor de prezentare și (sau) poziționare este simplitudinea lor, acestea având în general acționări pneumatice și nenecesitând un echipament de conducere în afară de un set de acționări digitale care pot fi fie din controllerul robotului fie dintr-un automat programabil. Avantajul axelor externe îl reprezintă faptul că acestea se pot mișca concomitent cu robotul permițând o dexteritate sporită respectiv scurtarea timpilor de ciclu (operațiile se execută "*din mers*").

3.1 Implementare mecanism poziționare

Pentru realizarea unui mecanism extern (robot, unealta, axă externă, dispozitiv) RobotStudio dispune de opțiunea *Create Mechanism* accesibilă din meniul *Modelling* (poza). Mai jos sunt prezentați pașii ce trebuie urmați pentru a crea un mecanism de tip axă externă (acest meccanism poate avea mai multe axe, ex: un mecanism de poziționare pe X și Z):

Pas 1: Creare structură grafică suport mecanism (se recomandă ca elementele de tip segment să aibă culori diferite pentru a se vizualiza mai bine.

Pas 2: Utilizare opțiune *Create Mechanism* (Fig. 7-18) pentru alegerea mecanismului de realizat și apoi configurarea după caz a segmentelor, a axelor de mișcare, a sistemului de coordonate condus și în final realizarea unei calibrări. OBS: sistemul de coordonate condus trebuie lăsat la valoare implicită propusă de sistem.

Pas 3: Adițional se pot defini anumite poziții (Fig. 7-19) pentru mecanismul de tip axă externă, poziții la care se poate duce în mod automat la activarea unei ieșiri digitale ca urmare a configurării din *Event Manager* (Event Manager -> Add -> I/O signals changed -> selecție semnal de acționare a mecanismului -> Move Mechanism to Pose -> selecție mecanism și poziție predefinită).

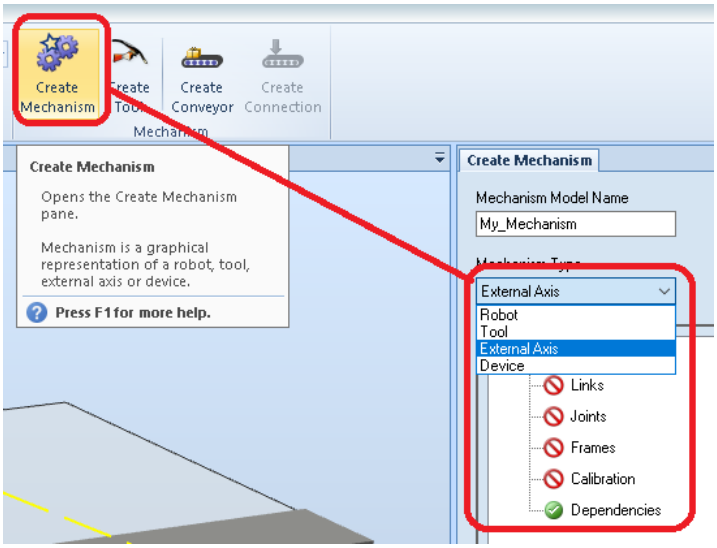


Fig. 7-18 – Opțiunea de creare a unui mecanism de tip axă externă

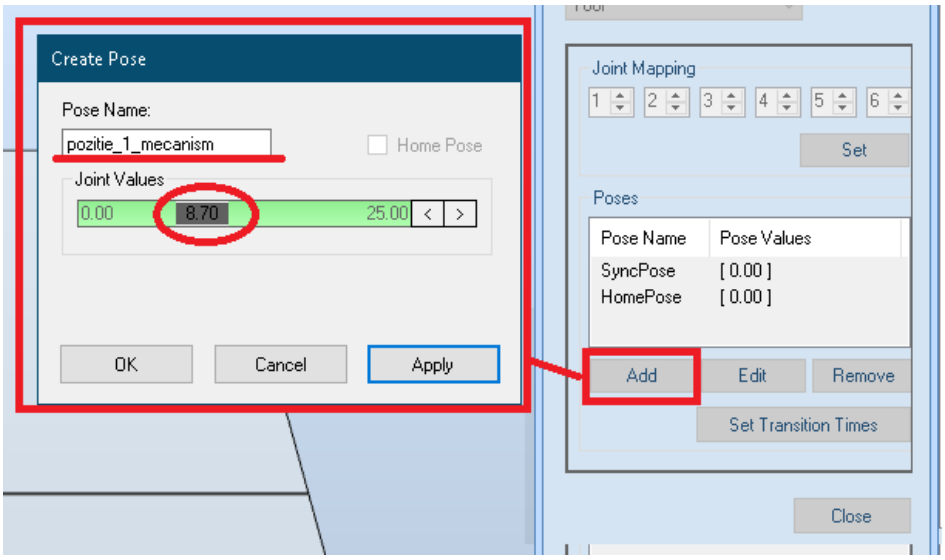


Fig. 7-19 – Definirea de poziții pentru mecanismul de tip axă externă

3.2 Extensia unui mecanism de poziționare la axa de mișcare și utilizarea acesteia ca suport robot

Din punct de vedere al simulării un mecanism de poziționare este un dispozitiv cu poziții predefinite care se mișcă la acțiunea unor semnale, neintrând în cinematica robotului. Dacă adăugăm un controller de mișcare mecanismului de poziționare (cu condiția ca acesta să aibă și un traductor de poziție pentru cazul real) se poate obține o axă de mișcare. Valoarea acesteia din urmă poate fi stocată într-un punct robot în secțiunea axe externe (pot fi memorate, și folosite, maxim 6 axe externe). Pentru crearea unei axe externe trebuie întâi creat mecanismul

suport (3.1) și apoi mecanismului i se adaugă controllerul de mișcare din meniul *Home* -> *Robot System* -> *External Axis Wizard*. Această funcționalitate nu este adăugată în mod implicit la RobotStudio, adaugarea ei putându-se face ulterior din meniul Add-Ins (Fig. 7-20).

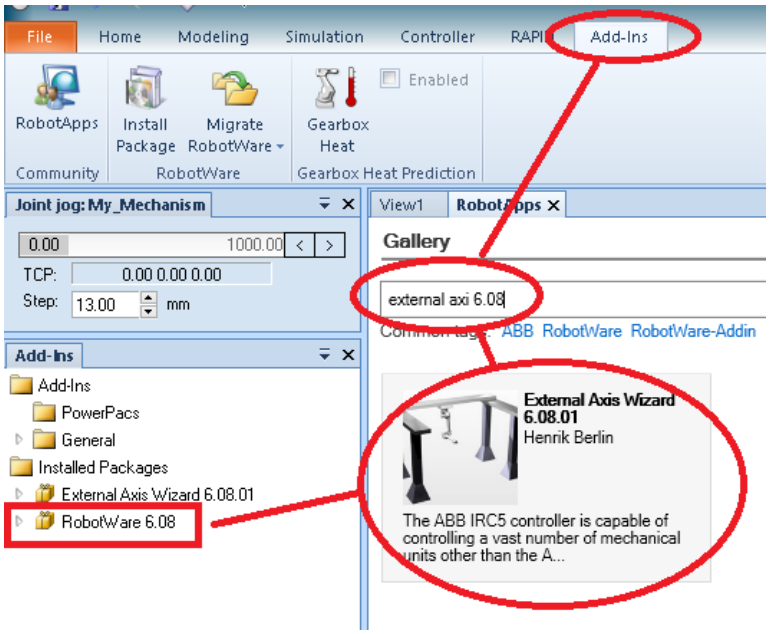


Fig. 7-20 – Adăugare funcționalitate lucru cu axe externe

Pentru adăugarea unui controller de mișcare la axa externă (axele externe) se va folosi meniul *External Axis Wizard* importat anterior din *Home* (Fig. 7-21). Ulterior se selectează articulațiile cărora le va fi adăugat controller de mișcare , respectiv controllerul de mișcare (Fig. 7-22). După crearea configurației robotul este adăugat prin *Drag&Drop* la axa de mișcare cu bifarea pozitivă a tuturor opțiunilor (coordonare cu axele externe). Ulterior dacă se va încerca mișcarea robotului în articulații se va vedea că apar și axele suplimentare, care la memorarea punctelor vor fi memorate în secțiunea aferentă de axe externe (Fig. 7-23).

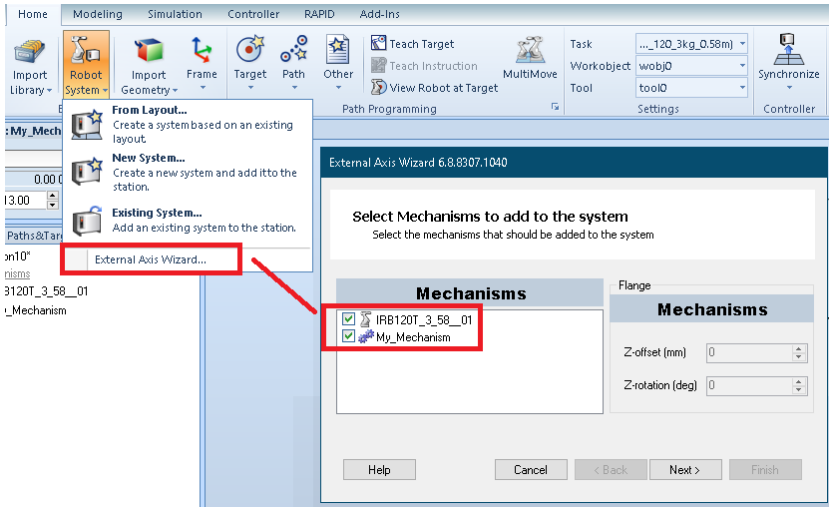


Fig. 7-21– Configurarea mecanismului de control pentru axa externă

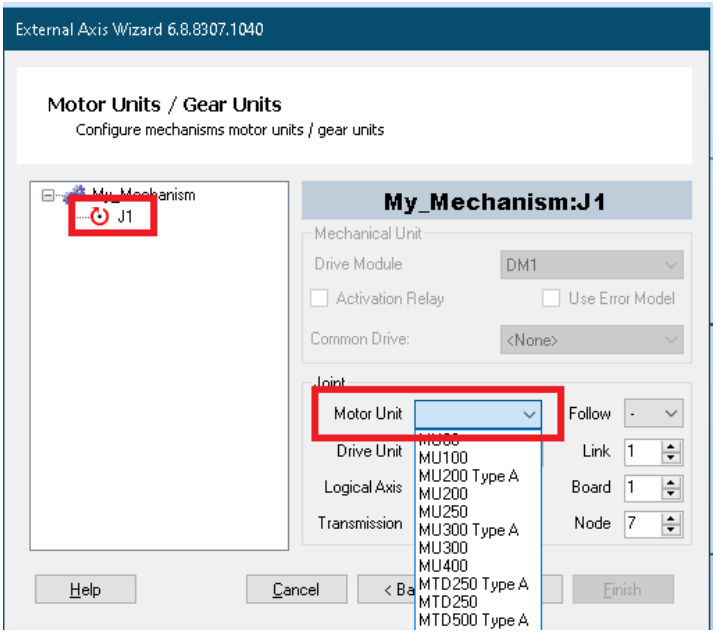


Fig. 7-22– Selectarea controllerului de mișcare pentru fiecare axă de mișcare externă

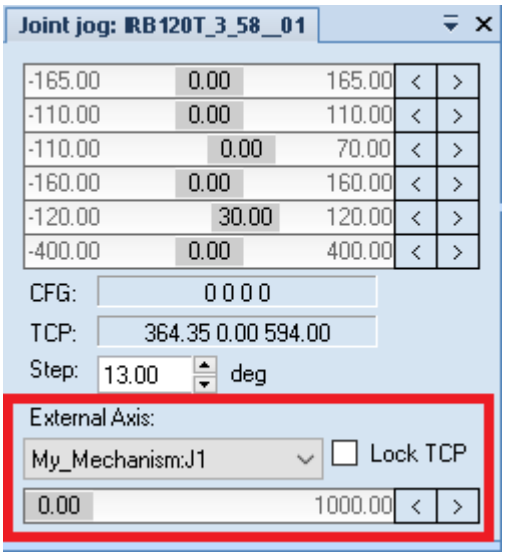


Fig. 7-23– Mișcarea la comun a tuturor axelor sistemului robot-mecanism de tip axă externă

```
CONST robtarget Target_10:=[[364.353826402,0,593.999995375],[0.499999968,0,0.866025422,0],[0,0,0,0],[0,9E+09,9E+09,9E+09,9E+09,9E+09]];
CONST robtarget Target_20:=[[657.445881714,-363.180198442,593.999995375],[0.339079813,0.636454723,0.587303515,-0.367457274],[-2,-1,0,0],[686.666666667,9E+09,9E+09,9E+09,9E+09,9E+09]];
```

Fig. 7-24– Structura unui punct cartezian cu tot cu axele externe de mișcare (OBS: valoarea 9E+09 specifică faptul că axa de mișcare nu a fost activate/nu a fost conectată)

3.3 Utilizarea de axe externe de mișcare ca suport robot/piesă (caz 1: mecanism robot – axă externă)

În cazul în care brațul robotic nu face față din punct de vedere al anvelopei de lucru există posibilitatea montării acestuia pe un pod rulant (de obicei cu un singur grad de libertate, dar pot fi și situații cu mai multe grade de libertate). În această situație gradul de libertate adițional este implementat printr-o axă de mișcare externă, condusă de un modul adițional al

controllerului robot. Această axă poate fi folosită și ca suport pentru piesa de lucru (de obicei în aplicații de sudare unde este nevoie de o dexteritate sporită pentru modalități de acces diferite).

În ambele cazuri se recomandă ca procedură de lucru mișcarea individuală a sistemului robot-axă externă în poziția dorită (conform traiectoriei) urmată de învățarea instrucțiunii de mișcare aferente (liniar, axe, absolut axe, axe externe) (Fig. 7-25).

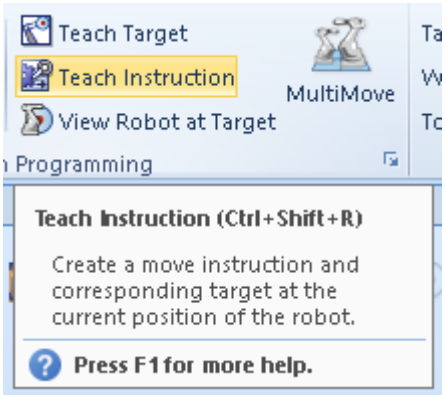


Fig. 7-25 – Utilizarea opțiunii de învățare a instrucțiunii de ajuns în punctul curent

Pentru situația în care se dorește ca robotul să funcționeze pe un pod rulant (sau alt tip de axă/axe externă(e)) se vor urma pașii din 3.2.

4 Sincronizare și mișcare coordonată în sistemele multirobot

4.1 Utilizarea semnalelor digitale pentru sincronizarea mișcării sistemelor multirobot

Un sistem robot poate dispune de mai multe module de intrare/ieșire. Aceste module au mai multe canale digitale sau analogice care trebuie conectate la semnale logice pentru a putea fi utilizate. În cadrul unei traiectorii accesul la un canal se face prin semnalul logic atașat. Un canal fizic poate fi conectat la mai multe semnale logice sau la nici unul. În cazul simulării se poate opera direct cu semnalele logice, cu observația că acestea trebuie conectate prin modulul *Event Manager* la diferite evenimente de interes (ex.: acționare gripper, element de poziționare, alt I/O robot). Instrucțiunile de bază în operarea cu semnale (WaitDO/WaitDi, SetDO, PulseDO) și modalitatea de adăugare sunt descrise în capitolul Principii generale de lucru. De asemenea, se poate realiza și testarea la rulare a unui semnal (IF do1=1 THEN ...), dar atunci când sunt utilizate pentru semnalizare nu este recomandat acest mod de lucru.

4.2 Mișcare coordonată

MultiMove – realizarea unui proiect de mișcare multirobot (exemplu)

Exemplu în care 2 roboți realizează urmărirea unei traiectorii comune în mod coordonat (se mișcă sincron pe aceleași traiectorii)

Un controller care conduce mai multe brațe are nevoie de module de comandă suplimentare, dar poate realiza mișcări coordonate cu setul de brațe fizice conduse (Fig. 7-26). Până la 4 brațe pot fi conduse de un controller.

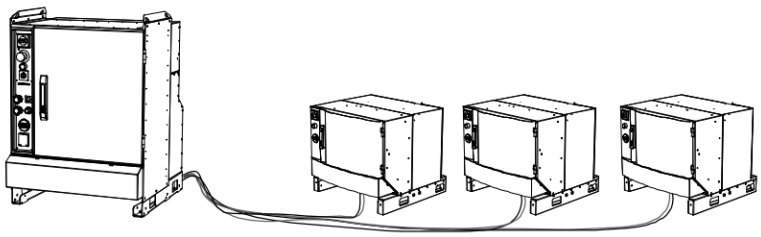


Fig. 7-26 – Sistem MultiMove echipat cu comunicație ETHERNET și semnale de siguranță

Tipurile de aplicații care implică utilizarea mai multor brațe pentru același proces includ, dar nu se restricționează la: sudare, vopsire și manipulare paralelă. Principalele avantaje ale utilizării mișcării coordonate țin de: a) posibilitatea de a sincroniza elemente individuale ale traiectoriilor pentru a evita coliziuni sau pentru a respecta procese fizice și, în anumite situații, de b) minimizarea costului investiției prin achiziția unui singur controller acolo unde este cazul – procese alăturate. În figura de mai jos sunt prezentați 2 roboți care operează

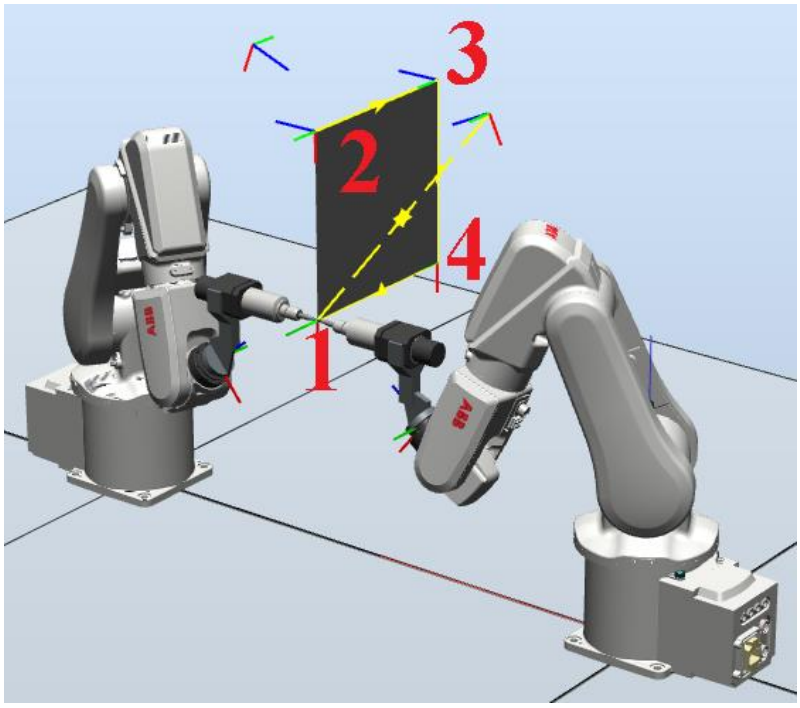


Fig. 7-27 – Mișcări coordonate a 2 roboți pe traiectoria 1-2-3-4-1

Instrucțiunile și structurile de date minimale pentru realizarea de mișcări coordonate sunt:

a) tasks – structură de date de tip vector în care sunt stocate denumirile task-urilor din sistemul curent. Este folosită în diferite instrucțiuni de coordonare pentru a avea o imagine asupra tuturor task-urilor ce trebuie coordonate. Variabilele de acest tip sunt persistente, ele fiind vizibile și având aceeași valoare pe toate task-urile controllerului.
b) syncident – este folosit pentru a identifica un punct din program unde se va aștepta ca instrucțiunile programelor care cooperează să atingă același punct de sincronizare. Numele acestor variabile (identitate) trebuie să fie același în toate task-urile cooperative, în punctele în care se realizează sincronizarea. Acest tip de

date este folosit în instrucțiuni precum *WaitSyncTask*, *SyncMoveOn*, și *SyncMoveOff*.

c) *SyncMoveOn / SyncMoveOff* – instrucțiuni folosite pentru a porni/opri o secvență de mișcări sincronizate și, în majoritatea cazurilor, mișcări coordonate. În primul rând, toate sarcinile programului implicat vor aștepta să se sincronizeze într-un punct de oprire/ și apoi planificatorul de mișcare pentru sarcinile programului implicat este setat în modul sincronizat sau independent (nesincronizat) după caz.

d) Identno – o valoare numerică sau o variabilă de tip identno este utilizată în ID-ul argumentului oricărei instrucțiuni de acționare a efectorului terminal executată între instrucțiunile *SyncMoveOn* și *SyncMoveOff*. Toate instrucțiunile de mișcare executate între instrucțiunile *SyncMoveOn* și *SyncMoveOff* trebuie să aibă ID-ul argumentului specificat. Argumentul ID trebuie să fie același pentru toate instrucțiunile de mișcare (în fiecare traiectorie) care ar trebui să se execute simultan.

În figura de mai jos sunt ilustrate instrucțiunile aparținând traiectoriei de mai sus pentru robotul R1. Pentru robotul R2 traiectoria va fi în oglindă, respectând identificatorii punctelor suport.

```
PERS tasks task_list{2}:=[["T_ROB1"],["T_ROB2"]];
VAR syncident sync1;
VAR syncident sync2;
VAR syncident sync3;
```

Variabile globale pentru fiecare traiectorie

```
PROC Path_10()
    SyncMoveOn sync1,task_list;
    MoveL p_R1_start_2\ID:=10,v1000,fine,MyTool\WObj:=wobj0;
    MoveL p_R1_start_3\ID:=20,v1000,fine,MyTool\WObj:=wobj0;
    MoveL p_R1_start_4\ID:=30,v1000,fine,MyTool\WObj:=wobj0;
    MoveL p_R1_start\ID:=40,v1000,fine,MyTool\WObj:=wobj0;
    SyncMoveOff sync3;
ENDPROC
```

Fig. 7-28 – Captură ecran traiectorie coordonată

4.3 Utilizarea de axe externe de mișcare ca suport piesă (caz 2)

Exemplu poziționare robot în sistem de coordonate axă externă – proiect MultiMove în care axa externă este controlată de un task separat de cel al robotului. Se va face coordonare între taskuri.

5 Exerciții

- Realizarea unei stive cu 2 roboți care pun piese alternativ – sincronizare I/O digitale

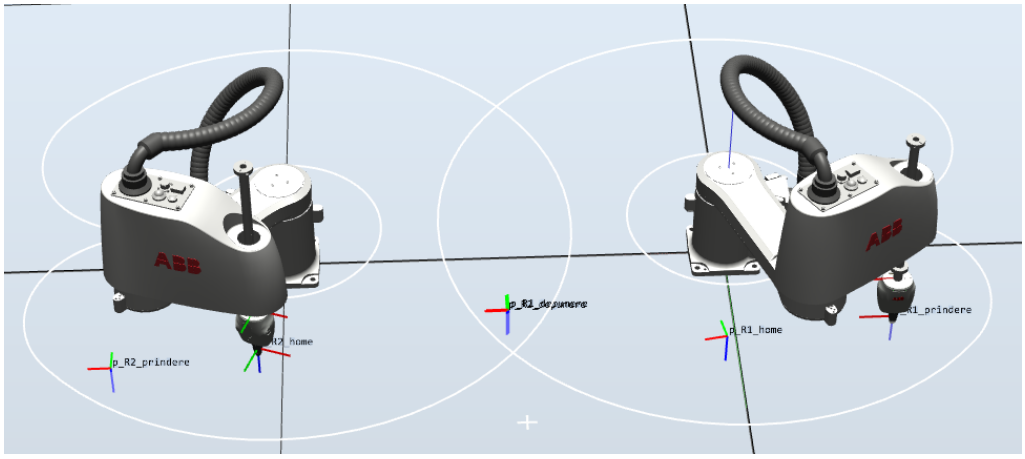


Fig. 7-29– Sistem multirobot care interacționează prin semnale digitale pentru partajarea unui spațiu de lucru comun

Se va realiza un proiect pentru a demonstra interacțiunea între doi roboți care partajează un spațiu de lucru comun folosind semnale digitale. Roboții vor depune alternativ piese în spațiul de lucru. Atenție la situațiile în care vitezele sunt disproporționate (ex.: un robot poate realiza toată traiectoria de depunere în intervalul în care celălalt realizează doar o parte dintr-o mișcare elementară – MoveJ/MoveL).

- Realizarea operațiunii de manipulare pentru un robot situat pe o axă liniară de mișcare

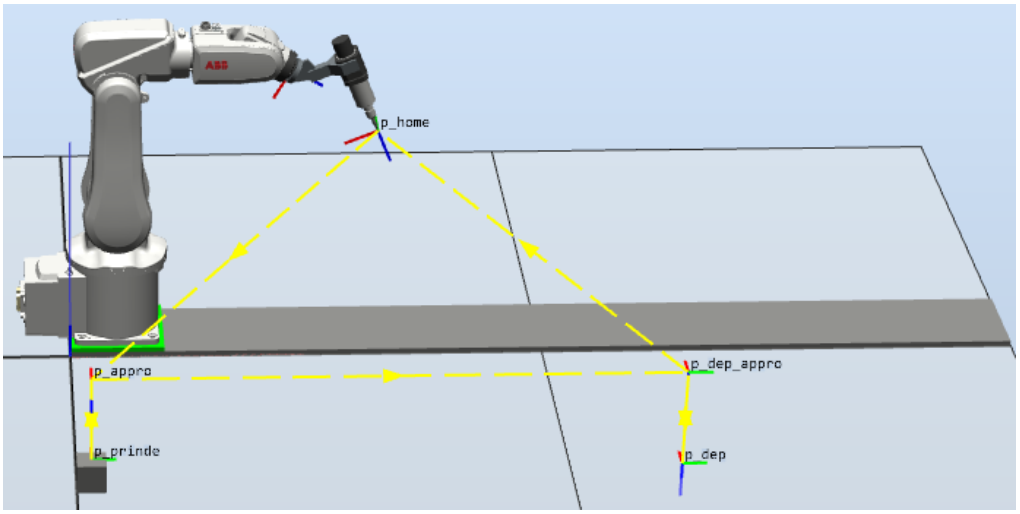


Fig. 7-30 – Robot montat pe pod rulant

6 Întrebări

Care este principiul de mișcare pentru sisteme multiaxă/multirobot?

Care este diferența între un mecanism de poziționare și o axă externă?

Capitol 8: Proiecte propuse

Cuprins

1 Introducere 130

2 Realizare operațiuni multiple cu roboți care ucrează în paralel / secvențial 131

3 Sistem de 2 roboți pentru realizarea și paletizarea unui ansamblu de piese 133

4 Gestiune scenă nestructurată folosind vederea artificială 134

5 Gestiune scenă nestructurată folosind gripper inteligent 135

6 Deservire utilaje si evacuare (gripper multiplu) 136

7 Repliere inteligentă ca urmare a unei opriri de urgență 138

8 Realizarea de stive cu piese care vin aleator pe bandă/Realizare ansambluri pe paleți (logistică) 138

9 Gestionare scene mobile (piese în mișcare pe conveior) 139

10 Testare componente (1) 140

11 Testare componente (2) 141

12 Deservire utilaje folosind un robot montat pe o axă externă de mișcare 142

1 Introducere

Roboții industriali au fost și sunt intensiv folosiți în domeniul fabricației, în special în ramura auto (en.: *automotive*; producția de mașini și de piese folosite la mașini) în aplicații precum: sudura cu arc electric (en: Arc Welding), sudura în puncte (en.: Spot welding), manipulare/paletizare/asamblare (en: materials handling, picking, packing and paletizing), deservire mașini cu comandă numerică /CNC (en.: machine tending), vopsit, procesare (en.: mechanical cutting, grinding, deburring and polishing), depunere de materiale adezive (en.: gluing, adhesive sealing and sparying materials), inspecție video sau decupare laser. Treptat, acest tip de echipamente devin o soluție viabilă și pentru alte tipuri de aplicații în cadrul procesului extins de fabricație (lanț de aprovizionare) precum în logistică, unde sunt folosite pentru a manipula produse în vederea stocării, preluării și ambalării automate. În general traiectoriile deservite pot fi descrise prin puncte, dar sunt și aplicații în care traiectoria este descrisă matematic (ex.: sudare sau vopsit). Indiferent de aplicație procesul de proiectare al unei celule robotizate ar trebui să țină cont de a) efectorul terminal manipulat, b) traiectoria pe care trebuie mișcat, c) constrângerile de masă manipulată, dexteritate spațiu de lucru, distanță de operare și timp de ciclu și d) modalitatea de interconectare atât cu dipozitivele de comandă (sisteme de gestiune a producției, automate progrmabile, ș.a.), cât și cu dispozitivele controlate (efector terminal, mecanisme de poziționare sau axe externe).

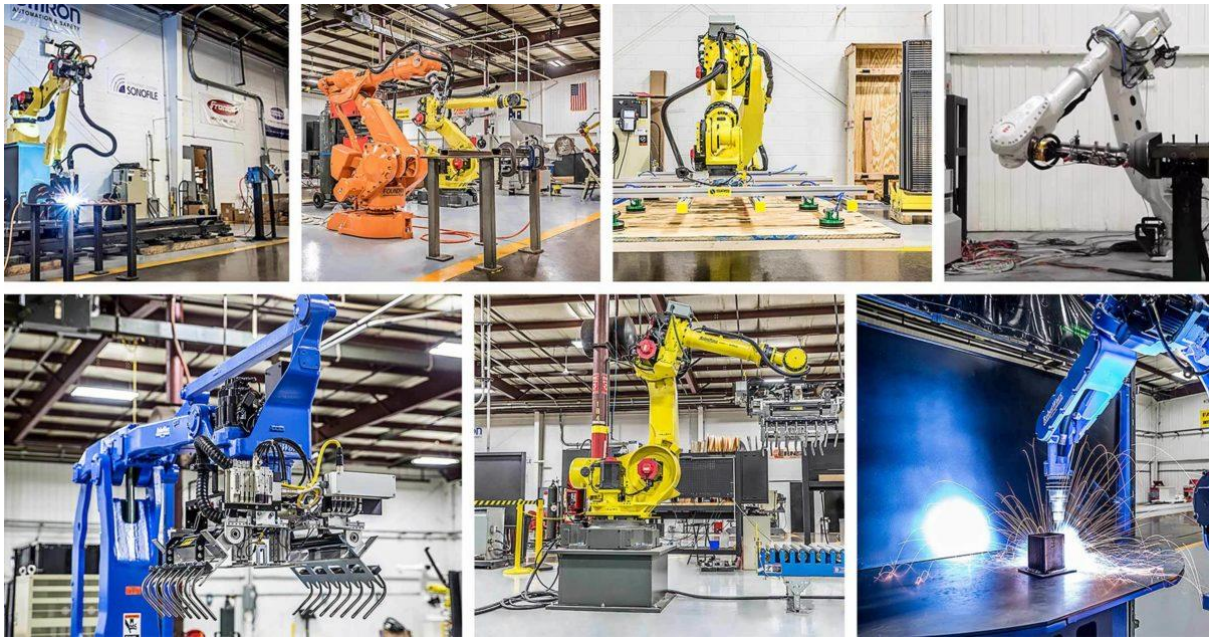


Fig. 8-1 – Exemple de utilizare a echipamentelor de tip robot industrial în producție

Având în vedere multitudinea de variabile care afectează realizarea unei celule robotizate este recomandată simularea modului de lucru de o manieră cât mai apropiată de realitate. Scopul final al unei astfel de simulări este alegerea unui anumit tip de robot în funcție de distanța de lucru, sarcina manipulată și dexteritate, precum și găsirea unei soluții optime de implantare (poziționare) care să satisfacă constrângerile de mediu. Astfel, în acest capitol sunt prezentate un set de aplicații în care roboți industriali sunt folosiți la partea de manipulare, insistându-se asupra acelor aspecte particulare ce trebuie configurate pentru a simula comportarea reală a mediului (ex.: modelarea unui efector terminal inteligent, a unei piese în mișcare pe un conveior, a achiziției infoprmației legate de poziția unui obiect în spațiu folosind un sistem de vedre artificială, ș.a.), cu exemple reale acolo unde este cazul.

2 Realizare operațiuni multiple cu roboți care ucrează în paralel / secvențial

Pe liniile de asamblare a automobilelor, o mare parte a muncii este realizată acum de roboți, mai degrabă decât de oameni. În primele etape ale fabricării automobilelor, roboții sudează piesele caroseriei și ajută operatorii umani să plaseze componente (Fig. 8-2). În acest context se propune realizarea unui proiect în care un set de 3 roboți industriali să execute operații de montare cu depunere într-o zonă comună. Proiectul constă într-o aplicație multirobot în care robotul R1 realizează o stivă tridimensională (4 pe X, 4 pe Y, 2 pe Z), apoi roboții R2 și R3 realizează fiecare în parte o stivă bidimensională (2 pe X, 8 pe Y; 8 pe X, 2 pe Z). Robotul R1 lucrează la început pentru realizarea stivei, apoi lucrează în paralel roboții R2 și R3 pentru dezasamblarea stivei. Procesele de realizare a stivei 3D, respectiv realizarea stivelor 2D au loc pe aceeași platformă, roboții coordonându-se între ei prin semnale digitale.

Caracteristici proces:

- Se vor folosi piese de tip cub (50/50/50) și un palet de 300/300;
- Se vor sincroniza roboții folosind intrări/ieșiri digitale;

- Se vor distruge piesele de pe palet la finalul unui ciclu de execuție folosind ieșiri digitale;
- Se vor folosi efectoare terminale cu degete.

Fig. 8-2 – Utilizarea de roboți industriali pentru fixarea prin sudare a unei caroserii

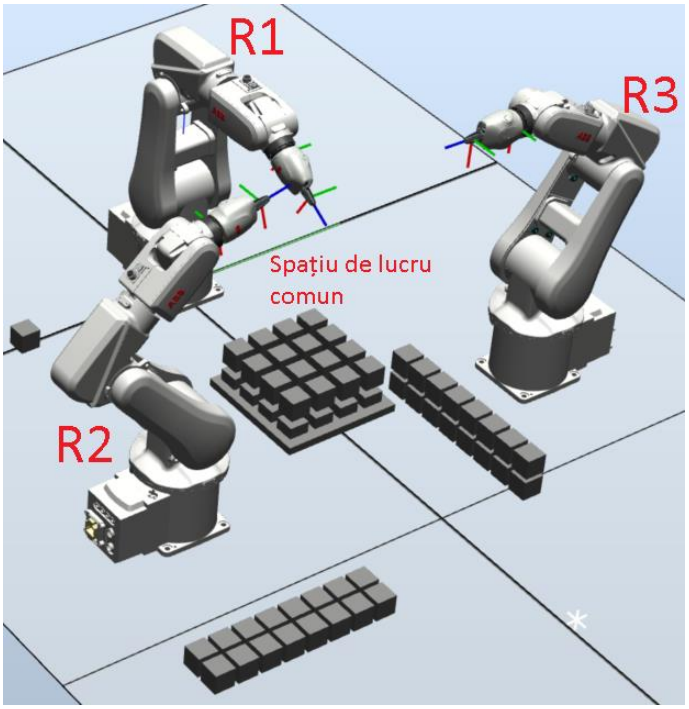


Fig. 8-3 – Exemplu de dispoziție a roboților pentru proiectul curent

Aspectele particulare ale acestui proiect (față de un proiect standard care implică mișcarea unui singur robot și acționarea ieșirilor aferente efectorului terminal) constau în: sistem multirobot în care se folosește lucrul cu semnale digitale pentru semnalizarea între roboți a terminării operațiilor. Aceste aspecte particulare sunt tratate în capitolele 7 (sisteme multirobot), respectiv 3 (adăugare de semnale digitale de intrare/ieșire).

3 Sistem de 2 roboți pentru realizarea și paletizarea unui ansamblu de piese

Se cere realizarea unui model digital și a programelor de control a roboților pentru un sistem multirobot (2 roboți) care realizează umplerea unei cutii cu piese și apoi manipularea ei. Primul robot realizează o stivă 2D pe paletul din imagine (poza) folosind piese generate la punct fix. După completarea paletului, R1 cere lui R2 printr-un semnal digital (urmat de o confirmare) evacuarea ansamblului *palet cu piese*. După ce R2 a evacuat ansamblul R1 așteaptă un nou palet și repornește operația de umplere. Se va realiza crearea și distrugerea dinamică a obiectelor manipulate din proiect.

Caracteristici proces:

- Se vor folosi piese de tip cub (50/50/50) și un palet de 300/300 (XOY);
- Primul robot pune piese pe palet;
- Al doilea robot ia paletul când este gata și îl evacuează;
- Un palet nou se crează la acționarea unei ieșiri digitale și se distruge cu tot cu piesele de pe el după ce este manipulat și depozitat de robotul 2.

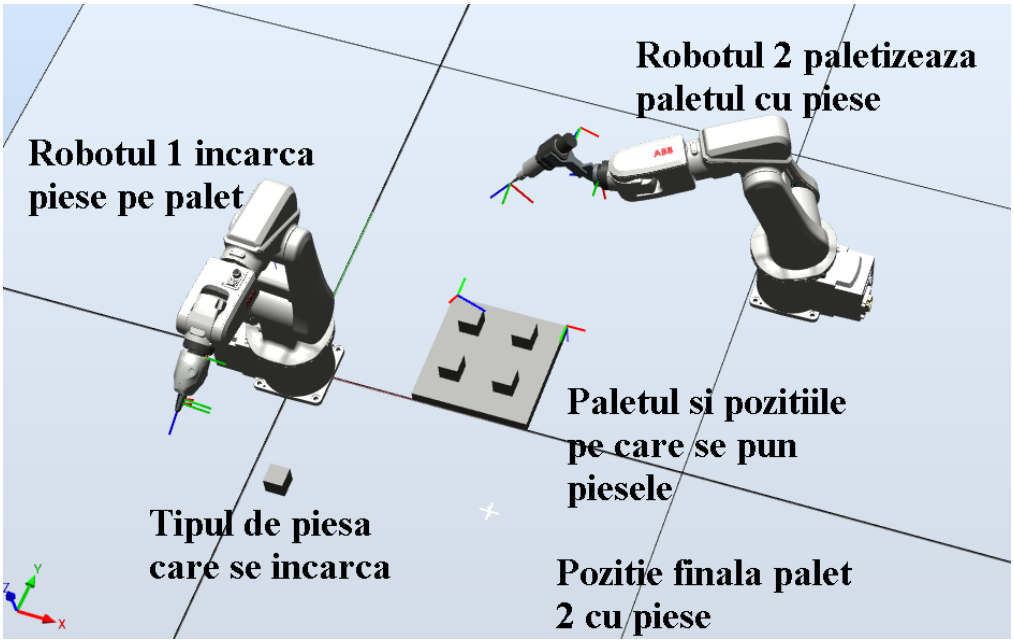


Fig. 8-4 – Exemplu de layout pentru proiectul 1

Elementul caracteristic acestui proiect constă în manipularea la comun a pieselor situate pe palet. O soluție pentru gestiunea la comun a acestor obiecte o reprezintă componenta activă de tip SetParent (StationLogic -> Add Component -> Actions -> SetParent). Astfel, după crearea, manipularea și depunerea unui obiect pe palet se execută această componentă care atașeaza obiectul de palet. Ulterior, la mișcarea paletului, acesta se va mișca cu tot cu obiectele de pe el. Descrierea modului de operare cu o componentă activă este realizată în capitolul 6.

4 Gestiune scenă nestructurată folosind vederea artificială

Posturile robotizate care operează în medii nestructurate (piese dispuse aleator, ex.: pe o bandă transportoare, într-un container, etc) devin din ce în ce mai căutate în aplicațiile industriale datorită simplitudinii realizării metodei de prezentare a pieselor. Cazul cel mai generic este acela al paletizării de obiecte dispuse aleator într-un container (en.: bin picking). Rezolvarea acestei probleme se poate face în multiple moduri care variază de la utilizarea unor senzori 2D sau 3D pentru localizarea pieselor până la utilizarea de grippere inteligente care detectează coliziunea cu obiectele necesare și apoi le manipulează. În cel de-al doilea caz este necesar un tip special de grippere, de obicei grippere magnetice.

Sistemul Bin Picking este capabil să proceseze obiecte dintr-o varietate de materiale care sunt distribuite aleator. Este un sistem capabil să prelucreze materialul distribuit aleator în containere nestructurate și să îl manipuleze pe linia de fabricație. Totul realizat cu un sistem precis de vedere 3D capabil să detecteze diferite tipuri de obiecte și sisteme robotizate, capabile să ghideze roboții pentru extragerea sigură a obiectelor (Fig. 8-5).

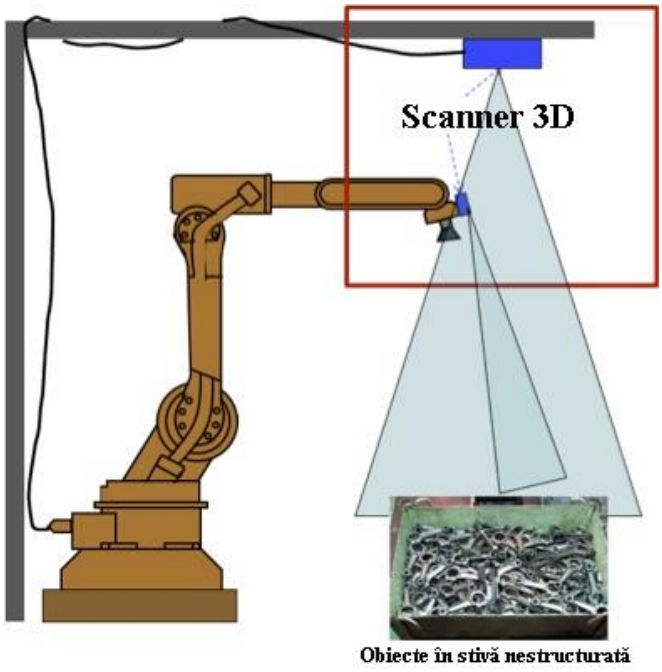


Fig. 8-5 – Stivă de piese dispuse în mod aleator. Utilizarea vederii 3D pentru detecția și manipularea selectivă de piese

În conextul prezentat mai sus se dorește golirea unui container plin cu piese puse în mod nestructurat (aruncate). Robotul este prevăzut cu un efector terminal asimetric de tip magnet. Poziția pieselor este cunoscută apriori prin intermediul unui sistem de vedere artificială simulat printr-o componenta activă (senzor de poziție). Trebuie avut atenție la posibilele coliziuni gripper/robot-container. Prinderea este rotativă și trebuie facută cu evitarea coliziunii.

Caracteristici proces:

- Dimensiune piese 50/50/50 mm
- Dimensiune cutie 500/300/100 mm

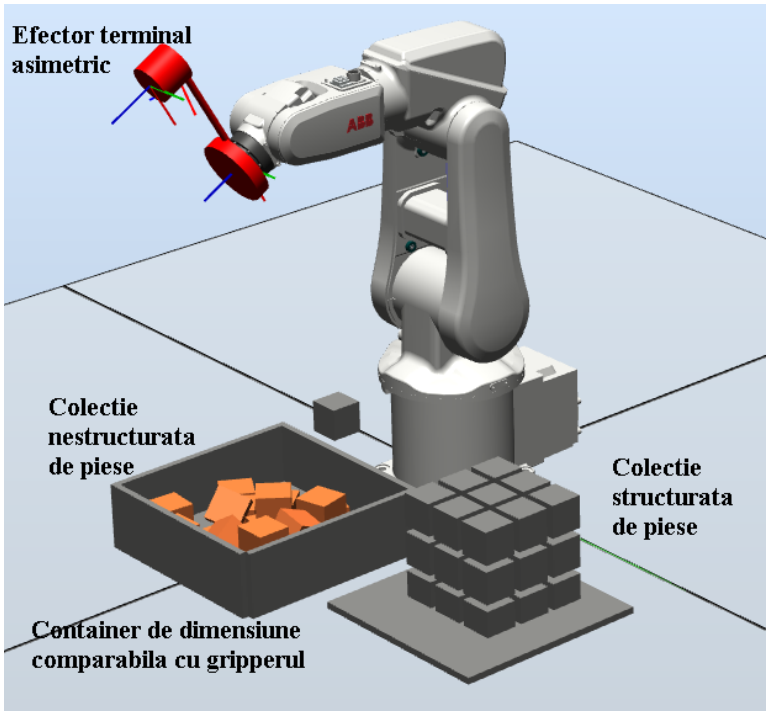


Fig. 8-6 – Model digital al structurii de paletizare care folosește informații de la un sistem de vedere 3D

Elementele caracteristice acestui proiect constau în: a) generarea aleatoare de piese în container și b) calculul poziției pieselor. Prima problemă se va rezolva prin generarea succesivă de obiecte cu comportament de tip dinamic (supuse legilor gravitației). Aceste obiecte vor cădea de la distanță fixă în container generând astfel o stivă nestructurată de obiecte. Cea de-a două problemă se va rezolva prin folosirea unei suite de componente active de felul următor: ClosestObject (pentru preluare obiect), PositionSensor (pentru preluare obiect detectat anterior) și RapidVariable (pentru comunicarea acestei variabile către programul Rapid la acționarea unui semnal digital).

5 Gestiune scenă nestructurată folosind gripper inteligent

În contextul proiectului precedent se dorește golirea unui container plin cu piese puse in mod nestructurat (aruncate). Robotul este prevazut cu un efector terminal de tip magnet cu senzor de detecție a impactului cu piesele manipulate (Fig. 8-7). Robotul efectuează sondări pe verticală și în momentul în care a dat de o piesă se oprește, activează gripperul și apoi paletizează piesa în stiva structurată din partea dreaptă.

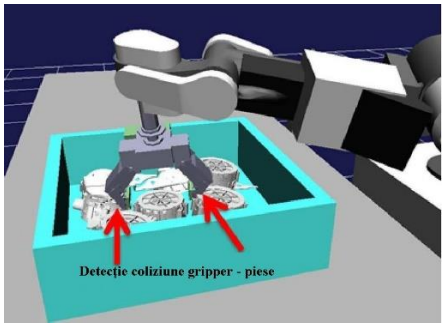


Fig. 8-7 – Exemplu de gripper cu capacitatea de detecție a coliziunii

Caracteristici proces (Fig. 8-8):

- Dimensiune piese 50/50/50 mm
- Dimensiune cutie 500/300/100 mm

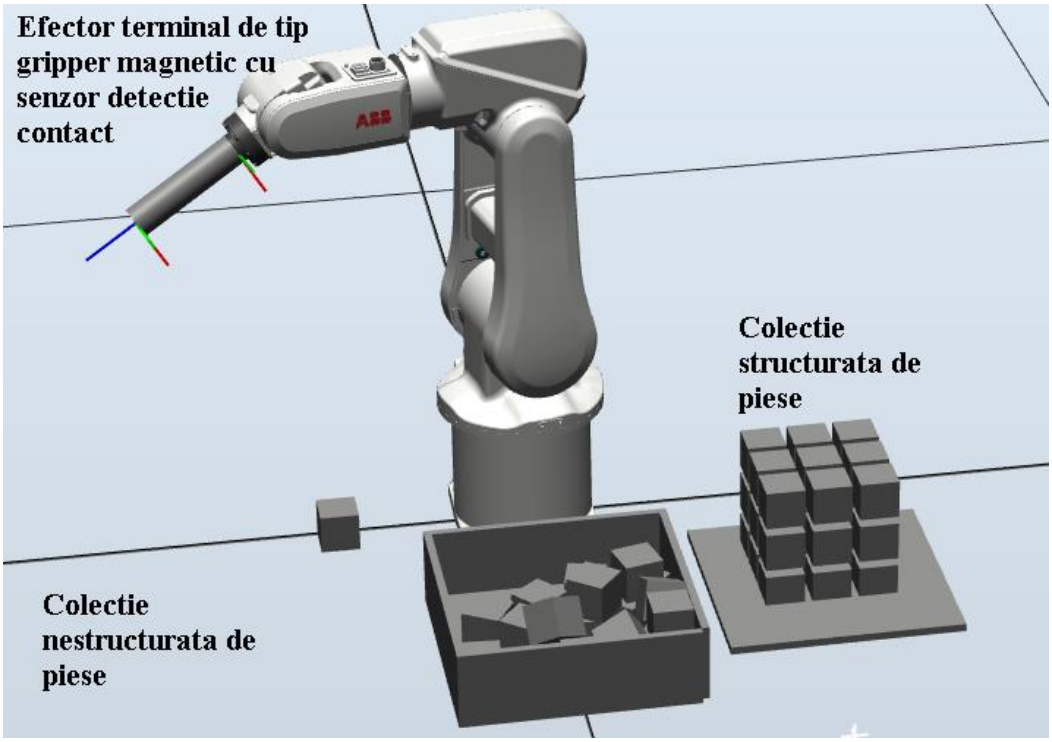


Fig. 8-8 – Model digital al structurii de paletizare care folosește gripper cu detecție a impactului

Elementele caracteristice acestui proiect constau în: a) utilizarea componentei active de tip senzor de volum pentru detecția coliziunilor și b) utilizarea instrucțiunii SearchL pentru sondarea pe verticală condiționată de senzorul de volum. La detecția unei piese se oprește robotul, se atașează piesa la gripper și se paletizează în stivă. Când s-a ajuns la fundul containerului se marchează zona respectivă ca goală și se continuă până când toate zonele au fost golite.

6 Deservire utilaje si evacuare (gripper multiplu)

Una din aplicațiile roboților industriali des întâlnite constă în deservirea unei mașini cu comandă numerică (Fig. 8-9). Caracteristicile acestei aplicații sunt constrângerile multiple ale brațului robotic, care nu trebuie să intre în coliziune cu echipamentul deservit, precum și viteza ridicată de deservire a acestuia astfel încât CNC-ul să nu aștepte fără să proceseze piese. Pentru a scădea timpul de ciclu, în industrie o soluție clasică este utilizarea de grippere cu mai multe capete care să opereze cu mai multe piese în același timp.

În acest context se propune un proiect care să simuleze procesul de deservire (alimentare cu piese și evacuare piese) a două mașini cu comandă numerică. Grafic o mașină cu comandă numerică este o cutie cu o deschidere limitată, deschidere prin care robotul trebuie

să scoată piesa procesată și apoi să introducă o nouă piesă pentru a fi procesată. Mașina CNC are un timp de procesare, ea operând în paralel cu mișcarea robotului.

Fig. 8-9 – Soluție automată de alimentare a unei mașini cu comandă numerică (CNC) folosind un brațrobotic articulat vertical

O descriere grafică a proiectului este dată în figura de mai jos (Fig. 8-10): un robot industrial cu efector terminal de tip gripper asimetric dual alimentează 2 mașini CNC piese cu timpi variabili de execuție. Cele 2 mașini vor fi deservite pe măsură ce devin libere după principiul prima mașină liberă va fi deservită prima. Un proces de deservire constă în pregătirea unei piese de depus și în momentul eliberării CNC-ului aferent este preluată piesa din CNC și depusă piesa pregătită. Piesa preluată din CNC va fi depusă într-un punct de evacuare diferit de punctul de preluare a pieselor neprocesate.

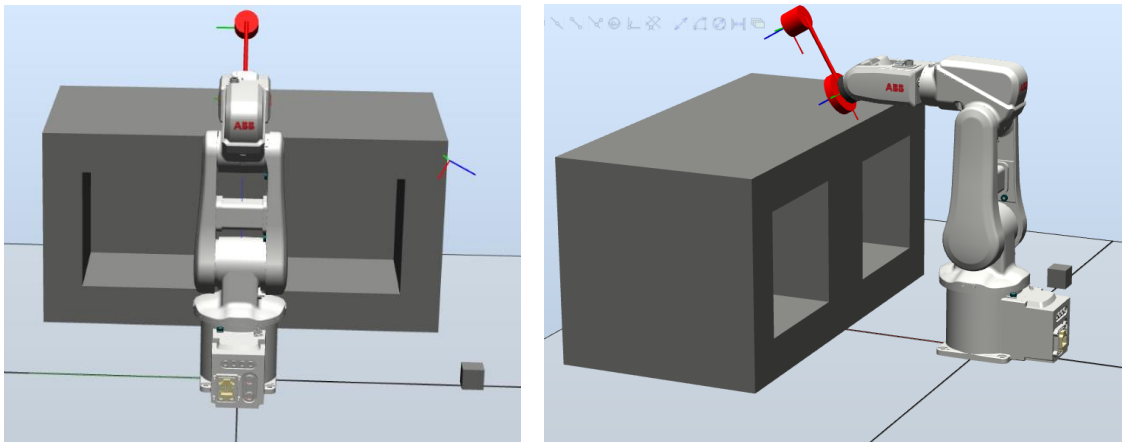


Fig. 8-10 – Model digital al sistemului dual robot-CNC

Elementele caracteristice acestui proiect constau în: a) realizarea unui gripper dual (acesta constă în realizarea a 2 grippere simple care vor fi montate simultan pe robot, atașarea / dezlipirea de obiecte realizându-se selectiv, în funcție de traiectorie), b) simularea procesării paralele pe mașinile CNC (acest lucru se realizează folosind task-uri adiționale în care se realizează o temporizare simplă urmată de o semnalizare pe semnale digitale) și c) realizarea unei verificări simultane a 2 semnale digitale (acest lucru se realizează folosind o conexiune de tip Cross connection).

7 Repliere inteligentă ca urmare a unei opriri de urgență

Sistemul aferent proiectului precedent este prevăzut cu un semnal de siguranță la apariția căruia procesul se oprește instantaneu. Pentru a relua procesul robotul trebuie să se reîntoarcă în punctul de start. Acest proces este unul complicat în special dacă pozițiile de deservire a mașinilor CNC sunt foarte „strânse" (necesită un set de mișcări combinat pentru a scoate robotul în siguranță din poziția respectivă – ex.: mișcare liniară locală în cartezian, mișcare în articulații, ș.a.). În acest caz este de preferat ca replierea să se facă în mod automat, în funcție de localizarea și configurația robotului.

Se cere, ca urmare a unei opriri de urgență, să se realizeze o procedură automată de mișcare a robotului în punctul de siguranță oricare ar fi punctul de oprire.

Pentru realizarea acestei proceduri se vor defini un set de zone de lucru ale robotului din care trebuie executate un set standard de operații pentru replierea în siguranță. Apartenența robotului la o zonă, precum și configurația sa vor fi testate înainte de începerea procesului de repliere.

8 Realizarea de stive cu piese care vin aleator pe bandă/Realizare ansambluri pe paleți (logistică)

Robotizarea posturilor de lucru este una dintre cele mai flexibile și mai rentabile soluții de ambalare pentru piața competitivă de astăzi. Precizia ridicată, reducerea accidentelor de muncă și rentabilitatea ridicată a investiției fac din sistemele de ambalare robotizate o alegere din ce în ce mai mare pentru rezolvarea provocării procesării eficiente a produselor finite. Robotul va prelua cutii de pe un conveior de acumulare și le va așeza pe palet în funcție de tipul lor. Robotul stochează cutiile de pe palet conform unui șablon programat. Robotul așează, de asemenea, foi de separare pe fiecare palet și între straturile de cutii pentru stabilitate. Când paletul este complet, acesta va fi evacuat și se va începe completarea următorului palet. Paletul se va deplasa printr-o perdea de senzori laser pentru a permite operatorilor să o scoată cu un stivuitor. Paleți adiționali sunt preluați dintr-un distribuitor automat de paleți (Fig. 8-11).

Fig. 8-11 – Post de lucru robotizat pentru realizarea de paleți

În spiritul procesului descris mai sus se propune un proiect în care un robot industrial realizează completarea unuor paleți cu cutii conform figurii de mai jos. Cutiile sunt de 2 tipuri

(roșu și albastru) și vin aleator pe banda transportoare din figura de mai jos (Fig. 8-12). Prezența cutiei precum și tipul ei sunt semnalizate robotului prin intrări digitale. În funcție de tipul cutiei aceasta va fi paletizată pe paletul aferent. În momentul în care pe un palet s-au depus toate cutiile necesare robotul se va întoarce în poziția de siguranță și va activa o ieșire pentru schimbarea paletului aferent. Prezența unui palet gol va fi semnalizată printr-o intrare digitală. Sistemul trebuie să funcționeze continuu până în momentul acționării unui semnal de siguranță.

Caracteristici:

a) dimensiune cutie – 200X400X200 mm
b) dimensiune palet stocare – Europalet1200X800 mm

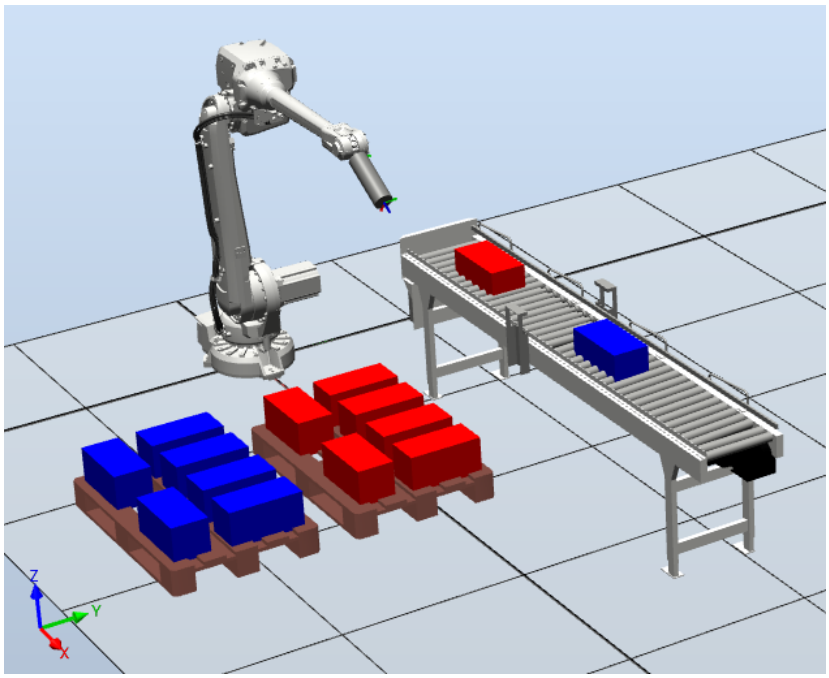


Fig. 8-12 – Structură proiect paletizare cutii

Aspecte particulare constau în a) generarea de obiecte de paletizat în mod aleator și b) realizarea unei stive conform unui șablon specificat. Aceste elemente vor fi detaliate în capitolul următor (exemplu de realizare a unui proiect în RobotStudio).

9 Gestionare scene mobile (piese în mișcare pe conveior)

În aplicațiile industriale cu cadență ridicată, procesarea pieselor care sunt prezentate pe un conveior se face din mișcare, fără oprirea acestuia. Astfel, robotul trebuie să se sincronizeze cu conveiorul (viteză relativă 0) pentru o perioadă de timp necesară acționării efectorului terminal, după care piesa este manipulată către punctul de destinație. O ilustrare grafică a acestui proces este reprezentată în figura (Fig. 8-13).

În acest context se cere paletizarea unor piese în mișcare folosind un echipament de tip robot industrial. Se vor paletiza toate piesele care vin pe un conveior în stive 3D. În urma completării unei stive se va opri banda transportoare se eliberează stiva și apoi se repornește procesul.

Fig. 8-13 – Proces industrial de procesare a obiectelor în mișcare

Etape prindere din mișcare:

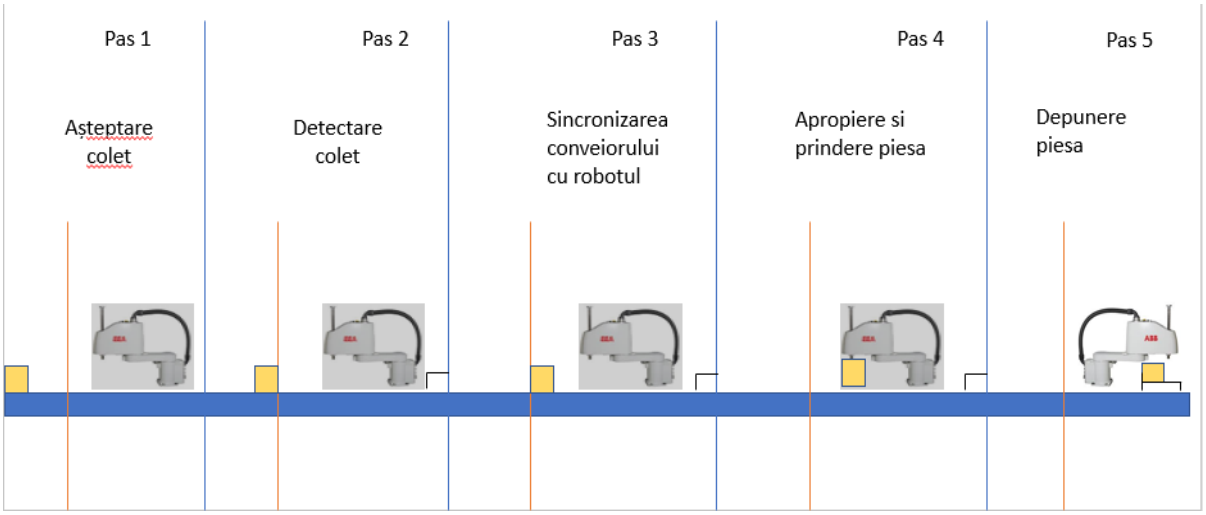


Fig. 8-14 – Etapele preluării unui obiect de pe un conveior în mișcare

Aspectul particular al proiectului constă în crearea și utilizarea unei axe externe de mișcare cu care să se sincronizeze robotul.

10 Testare componente (1)

Se cere să se realizeze traiectoria robot care preia piese de pe palet 1 și le depune pe palet 2 cu verificarea conformității piesei. Piesele manipulate vor fi trecute printr-o locație intermediară în care vor fi testate. O testare constă în punerea unui semnal de ieșire pe adevărat și verificare ulterioară a unui semnal de intrare care indică faptul că piesa este defectă sau nu. Doar piesele bune vor fi depuse pe paletul 2, cele defecte fiind depuse într-o poziție de unde vor fi eliminate ulterior. În momentul terminării unui palet cu piese de testat acesta va fi înlocuit de un palet nou. În momentul în care se termină de umplut paletul cu piese bune se va semnaliza pentru a se înlocui paletul cu unul gol. Proiectul va fi realizat de așa natură astfel încât să opereze în continuu.

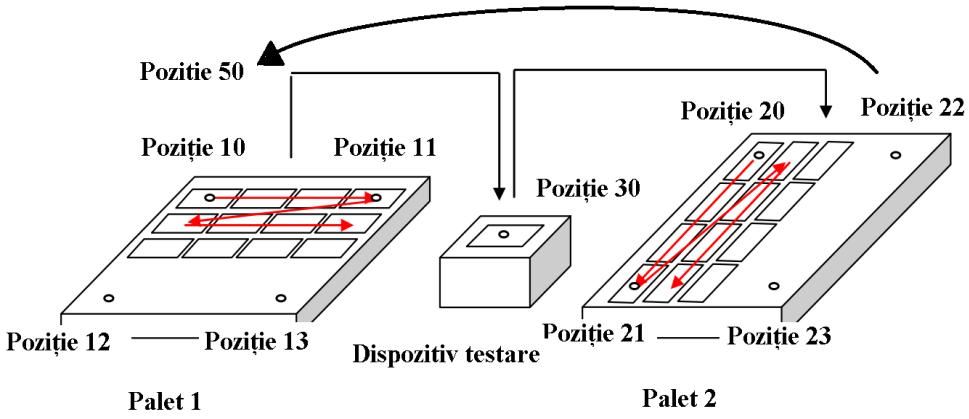


Fig. 8-15 – Pozițiile ce trebuie deservite și traiectoria aferentă manipulării unor obiecte ce trebuie testate

11 Testare componente (2)

Proiectul presupune mișcare coordonată (Fig. 8-16) a 2 roboți pentru verificarea planeității unui obiect de tip cub prin scanarea acestuia cu un palpator. Cubul va fi manipulat de un robot prevăzut cu efector terminal de tip ventuză/magnet. Acest robot va prezenta cubul pentru inspecție robotului 1 care va realiza un set de traiectorii de scanare pe fiecare față a cubului. Pentru prezentarea tuturor fețelor robotul 2 va trebui să preia cubul din diferite poziții. Palpatorul constă într-un senzor de tip volum care va fi mișcat pe suprafața verificată. Dacă senzorul întâlnește o denivelare atunci produsul va fi dat ca defect.

Se vor realiza 2 tipuri de produse: produs bun și produs defect (cu denivelare pe o față). Procesul se rulează o singură dată prin specificarea tipului de piesă verificat printr-o intrare digitală.

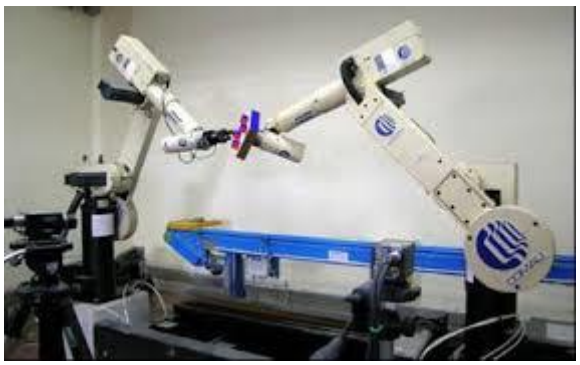


Fig. 8-16 – Roboți cu mișcări coordonate (un robot se mișcă într-un sistem de coordonate manipulat de un al doilea robot)

Observații

- se consideră că produs bun înseamnă produs fără denivelări externe
- pentru robotul de prezentare se va defini un punct virtual de lucru în centrul cubului scanat. Acest punct va fi atins cu diferite orientări
- se va folosi instrucțiunea SearchL să se testeze planeitatea unor piese procesate mecanic. Piesele vor fi aduse în postul de procesare de același braț robotic.

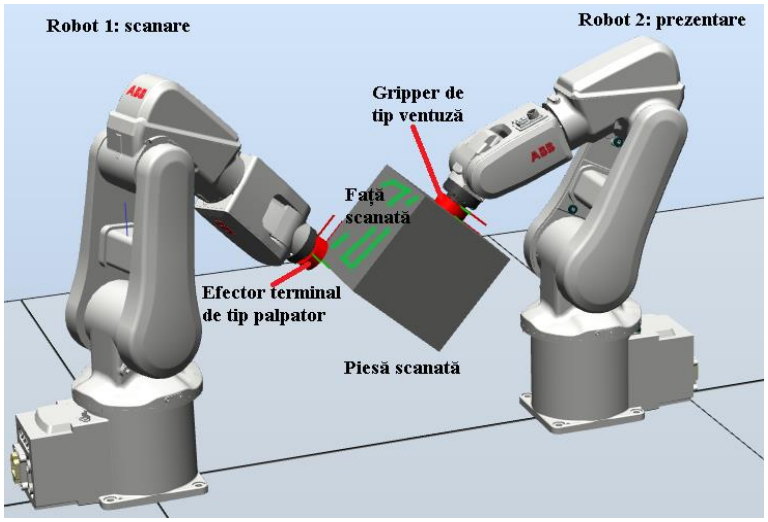


Fig. 8-17 – Model digital al sistemului dual robot (un robot manipulează piesa de testat și un al doilea verifică existența unor denivelări pe obiectul testat)

Aspectele particulare ale proiectului constau în a) mișcarea coordonată a celor două echipamente pentru a putea acoperi suprafața scanată și b) realizarea de prinderi/fixări multiple a piesei pe același efector terminal (acest lucru se realizează prin specificarea unui decalaj de atașare la robot, în cazul de față decalajul constând într-o rotație).

12 Deservire utilaje folosind un robot montat pe o axă externă de mișcare

Pentru extinderea spațiului de lucru al unui robot în aplicații industriale se montează acest echipament pe un pod rulant (Fig. 8-18 stânga), existând posibilitatea ca prin adăugarea unui modul de comandă a unei axe liniare să se controleze punctele folosind 7 grade de libertate. Proiectul presupune alimentarea a 3 mașini CNC cu piese care vin pe un conveior. Piesele vor fi preluare de un braț robotic montat pe un pod rulant pentru a se extinde spațiul de lucru conform diagramei din (Fig. 8-18 dreapta).

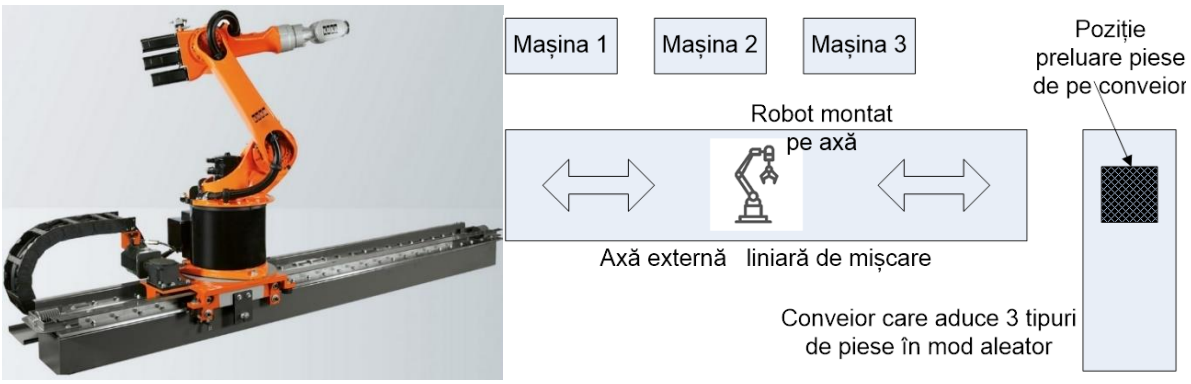


Fig. 8-18 – Robot industrial montat pe un pod rulant (axă externă liniară) / Infrastructură robot-axă liniară-masini CNC

Capitol 9: Proiect rezolvat

Cuprins

1 Descriere context 143

2 Cerințe 144

3 Configurare mediu de lucru (amplasament obiecte, efector terminal) 144

4 Definire semnale digitale și evenimente 145

5 Puncte suport traiectorie 145

6 Traiectorii robot 146

7 Configurare componente active pentru generare obiecte în mod aleator 147

8 Calcul timp de ciclu 148

1 Descriere context

Un robot industrial realizează completarea unuor paleți (tip Euro-palet – 1200X800 mm) folosind cutii care vin pe un conveior de acumulare conform figurii de mai jos. Cutiile de dimensiunea 200X400X200 mm, sunt de 2 tipuri (roșu și albastru) și vin aleator pe banda transportoare (Fig. 9-1). Prezența cutiei precum și tipul ei sunt semnalizate robotului prin intrări digitale. În funcție de tipul cutiei aceasta va fi paletizată pe paletul aferent. În momentul în care pe un palet s-au depus toate cutiile necesare robotul se va întoarce în poziția de siguranță și va activa o ieșire pentru schimbarea paletului aferent. Prezența unui palet gol va fi semnalizată printr-o intrare digitală. Sistemul trebuie să funcționeze continuu până în momentul acționării unui semnal de siguranță.

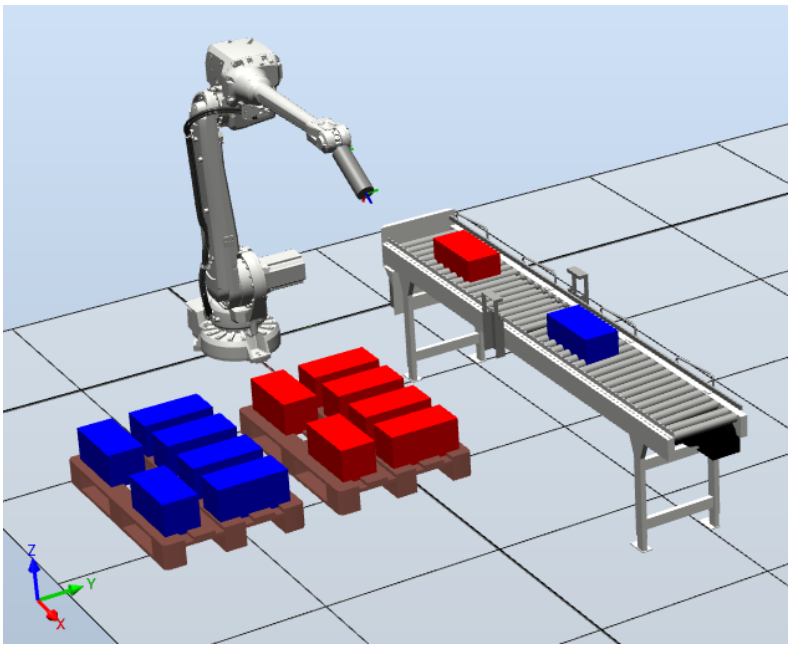


Fig. 9-1 – Exemplu de dispoziție a roboților pentru proiectul curent

2 Cerințe

Realizarea în format digital a unui proiect de automatizare care implică un robot industrial presupune următoareale etape: a) realizarea modelului virtual, b) alegerea tipului de robot (sau de roboți dacă este un proiect multirobot) și a efectorului terminal astfel încât să poată fi deservite pozițiile necesare, în acest pas se stabilește și poziția robotului/roboților, c) descrierea intrărilor și ieșirilor necesare pentru interconectarea cu mediul (automat programabil) și cu dipozitivul manipulat (e.g.: gripper), d) descrierea punctelor suport pentru traiectoriile robotului, e) descrierea traiectoriilor, f) implementarea traiectoriei și a componentelor active care simulează mediul real, g) calculul timpului de ciclu (sau a timpului mediu de ciclu în cazul în care preluarea se realizează din poziții variabile – ex.: o cutie care se golește).

Dintre avantajele dezvoltării acestui model digital enumerăm următoarele: a) validarea traiectoriei, a tipului de gripper și a amplasamentului robotului fără a fi nevoie să se opereze pe sistemul fizic, lucru care ar necesita scoaterea din producție a acestuia din urmă, b) detectarea posibilelor coliziuni și corecția traiectoriilor, c) calculul timpului de ciclu care intervine direct în calculul de rentabilitate al soluției.

3 Configurare mediu de lucru (amplasament obiecte, efector terminal)

În proiectarea soluției pentru aplicația descrisă mai sus se pornește de la amplasamentul fizic al structurii de tip conveior (cu tot cu poziția de acumulare) și a locațiilor în care sunt localizați Euro-paleții (Fig. 9-2). Pentru a putea poziționa robotul fără probleme pe această schemă se recomandă crearea unei soluții RobotStudio goale (fără controller). Deoarece în astfel de situații efectorul terminal este de tip gripper cu ventuză se va adăuga la proiect un astfel de echipament care să extindă pe axa Z finală a robotului punctul condus. După realizarea geometriei dorite și verificarea faptului că punctele ce se doresc a fi deservite se află în anvelopa de lucru a robotului, cu tot cu efectorul terminal montat, se crează soluția cu controller (*Robot System -> From Layout*).

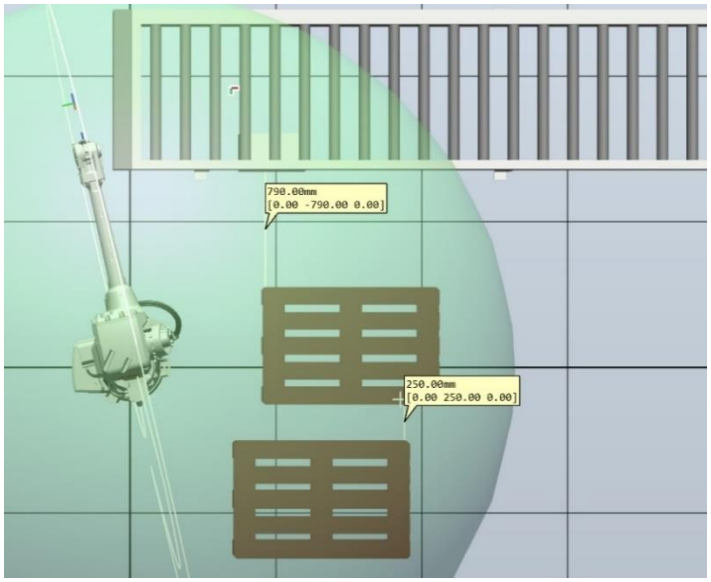


Fig. 9-2 – Vedere de ansamblu despre localizarea robotului relativ la conveiorul de prezentare a cutiilor și a locațiilor punctelor de depunere (Euro paleți)

4 Definire semnale digitale și evenimente

Pentru operarea efectorului terminal și a mediului se vor folosi următoarele semnale:

Tabel 9-1 – Descrierea semnalelor digitale folosite la soluția de automatizare

Nume semnal	Tip semnal	Descriere
cerere_cutie	Ieșire digitală	Semnalul este activat pentru a se crea o cutie în punctul de preluare. Tipul acestei cutii va fi generat aleator și specificat robotului prin semnalul *tip_cutie*. Cererea se va face pe front crescător.
cerere_pal1	Ieșire digitală	Semnalul este activat pentru a se goli paletul 1 (cutii roșii). Cererea se va face pe front crescător.
cerere_pal2	Ieșire digitală	Semnalul este activat pentru a se goli paletul 2 (cutii albastre). Cererea se va face pe front crescător.
gripper	Ieșire digitală	Semnalul realizează atașarea, respectiv dezlipirea celui mai apropiat obiect de efectorul terminal. Atașarea se va face pe valoarea adevărat și dezlipirea se va face pe valoarea fals. Configurarea acestei funcționări se va face din *Simulation -> Event Manager*.
conf_cutie	Intrare digitală	Semnal digital prin care se confirmă faptul că există o cutie creată în punctul de preluare. O valoare pozitivă indică prezența cutiei, respectiv o valoare falsă indică faptul că nu a fost creată încă cutia.
tip_cutie	Intrare digitală	Semnal digital prin care se specifică tipul de cutie generată (adevărat – cutie roșie, fals – cutie albastră). Semnalele confirmare și tip vor fi folosite la comun pentru validarea prinderii și a locației de depunere.
conf_pal1	Intrare digitală	Semnal digital legat la postul de lucru (va fi accesat prin modulul *I/O Simulator*) prin care se confirmă faptul că pot fi depuse cutii pe paletul 1. Adevărat – cutia poate fi depusă, fals – cutia nu poate fi depusă.
conf_pal2	Intrare digitală	Semnal digital legat la postul de lucru (va fi accesat prin modulul *I/O Simulator*) prin care se confirmă faptul că pot fi depuse cutii pe paletul 1. Adevărat – cutia poate fi depusă, fals – cutia nu poate fi depusă.

Semnalele vor fi adăugate din meniul *Controller -> Configuration -> I/O System - > Signal*. Semnalele vor avea tipul stabilit în tabelul de mai sus și tipul de acces *All*.

Evenimentele aferente semnalelor vor fi adăugate din secțiunea *Event Manager* (pentru evenimentele asociate efectorului terminal: atașare/dezlipire piesă de gripper), respectiv din secțiunea *Design* (pentru evenimentele asociate mediului de lucru: generare/ștergere cutie, generare/ștergere palet).

5 Puncte suport traiectorie

Având în vedere poziționarea nerepetitivă a cutiilor pe palet, precum și numărul mic de poziții ce trebuie deservite se preferă să se învețe fiecare punct în parte, depunerea realizându-se conform unui index aferent fiecărui palet (Fig. 9-3). Acest index va fi incrementat de fiecare dată când se realizează o depunere, iar la atingenrea valorii maxime se resetează indexul iar paletul se golește. Alte puncte necesare în realizarea proiectului sunt punctul de siguranță (safe), cel din care pornește și se termină traiectoria robot, punctul de prindere a cutiilor de pe conveior și punctele intermediare (generate în mod automat cu instrucțiunile Offs/RelTool)

necesare operației de paletizare (en.: pick-and-place). Acestea din urmă se vor alege astfel încât depunerea să se realizeze în siguranță (în cazul de față, pentru cutii cu înălțimea de 200mm, se va alege o apropiere/depărtare ușor superioară acestei valori).

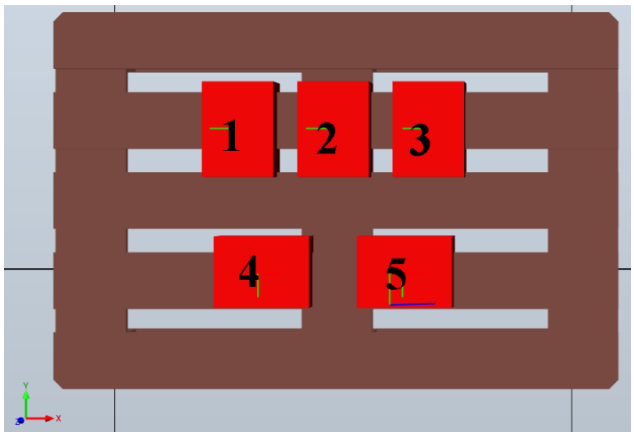


Fig. 9-3 – Indexarea pozițiilor de depunere pe Euro palet

6 Traiectorii robot

Pe controllerul robot se definesc următoarele traiectorii/proceduri: *Main* și *pick_and_place*. Punctul de intrare în program este procedura *Main*. Aceasta este traiectoria asociată mișcării în punctul de siguranță (*safe*) și calculelor aferente alegerii depozitului utilizat și a punctelor intermediare. Din această traiectorie se apelează traiectoria pick_and_place ce realizează mișcarea cutiei între punctele calculate în *Main* și transmise ca și variabile globale (prindere – pick, și depunere – place). Mai jos este dat pseudocodul traiectoriei *Main* și o ilustrare grafică a celor două traiectorii la comun (Fig. 9-4).

Pseudocod traiectorie Main

- `Pas 1: mișcare în punctul de siguranță`
- `Pas 2: cerere cutie`
- `Pas 3: decide traiectoria (roșu sau albastru) în funcție de cutia generată la pasul 2`
- `Pas 4: calcul puncte de preluare și depunere cutie, apel traiectorie manipulare, incrementare index aferent tipului de cutie generat la pasul 2`
- `Pas 5: verifică dacă s-a umplut un palet. În caz afirmativ semnalizează golirea paletului`
- `Pas 6: dacă s-a cerut golirea unui palet se așteaptă confirmarea aferentă`

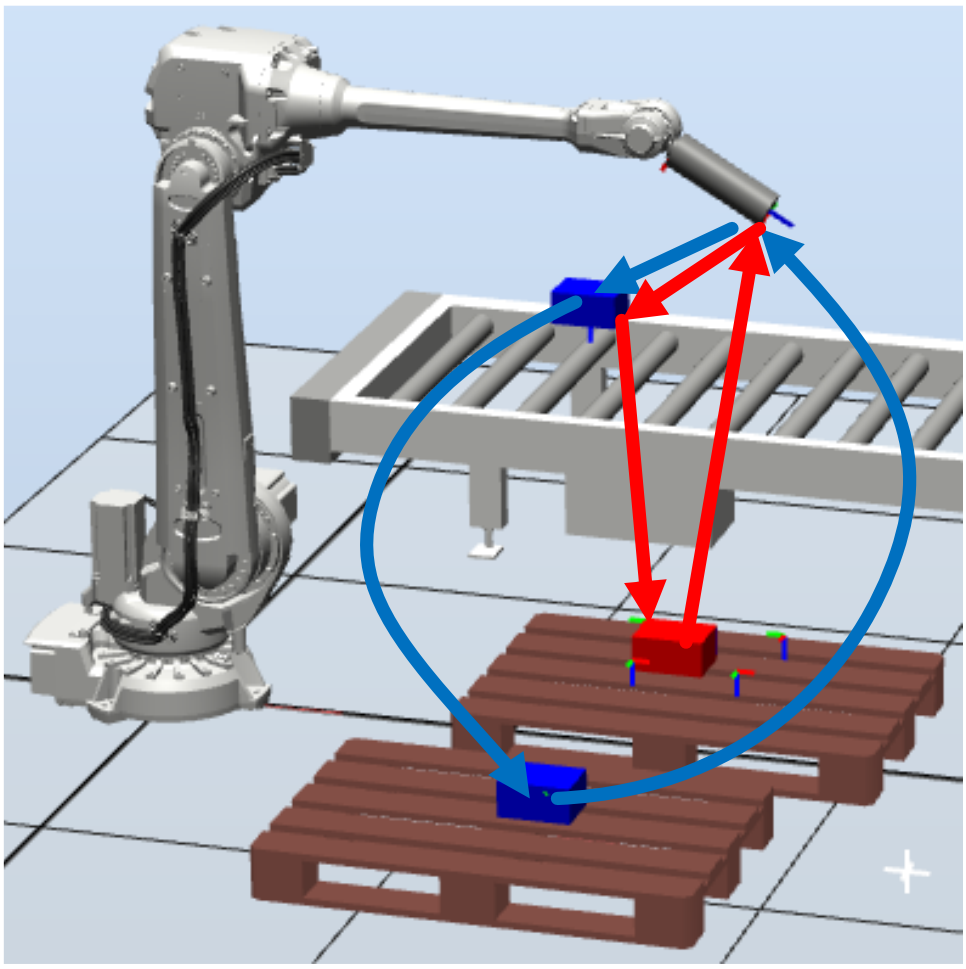


Fig. 9-4 – Traiectoriile aferente celor 2 tipuri de cutii

7 Configurare componente active pentru generare obiecte în mod aleator

Pentru generarea în mod aleator de cutii se va folosi componenta activă *Random* care generează un număr real într-un interval stabilit. Ulterior acest număr este testat relativ la o constantă și rezultatul va genera o cutie de un anumit tip. Pentru o generare echilibrată de cutii roșii și albastre se va folosi un prag de 50% astfel: generare număr aleator în intervalul [0, 1], dacă numărul este mai mare ca 0.5 atunci cutia generată este roșie, altfel este albastră. O reprezentare grafică a componentelor active care generează aleator cutii roșii și albastre este ilustrată în figura de mai jos (Fig. 9-5).

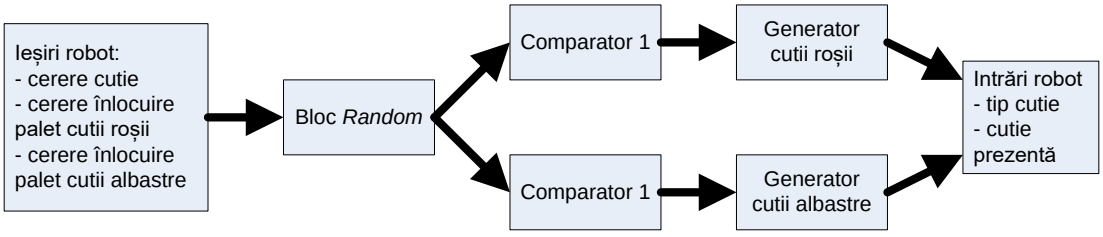


Fig. 9-5 – Diagramă logică pentru generarea de cutii în mod aleator

Pentru blocurile generatoare de cutii (Source) se va avea grilă ca după execuția blocului să se returneze valoarea semnalului la 0. Având în vedere că generarea se face la acționarea

semnalului *cerere cutie*, care coboară apoi la 0, între comparator 1&2 și blocul generator se poate folosi un bloc de tip ȘI logic, bloc ce coboară semnalul la 0 și apoi îl ridică la 1.

8 Calcul timp de ciclu

Timpul de ciclu este un parametru important în aplicațiile robotizate, el fiind unul din informațiile necesare atunci când se decide automatizarea unui post de lucru. În general acesta trebuie să fie mai mic decât timpul de ciclu al utilajului deservit. Pentru calculul precis al timpului de ciclu se folosește o variabilă de tip ceas (Clock). Această variabilă este pornită pentru contorizare înainte de apelul procedurii pick_and_place și este oprită imediat după terminarea execuției acesteia. Valoarea înregistrată va fi afișată la consola robotului. Mai jos este un exemplu de utilizare a variabilei de tip *clock*:

```
var clock my_clock;
...
PROC main()
…
ClkReset my_clock;
ClkStart my_clock;
pick_and_place
ClkStop my_clock;
TPWrite "Procesul de paletizare a durat "\Num :=
ClkRead(my_clock);

ENDPROC
```

Rezultatul acestei secțiuni de cod poate fi vizualizat pe modulul de comandă manuală accesibil din meniul *Controller -> FlexPendant -> Virtual FlexPendant*. Utilizarea acestui modul de comandă presupune rularea traiectoriei în mod producție direct din *FlexPendant* conform ilustrațiilor de mai jos (Fig. 9-6).

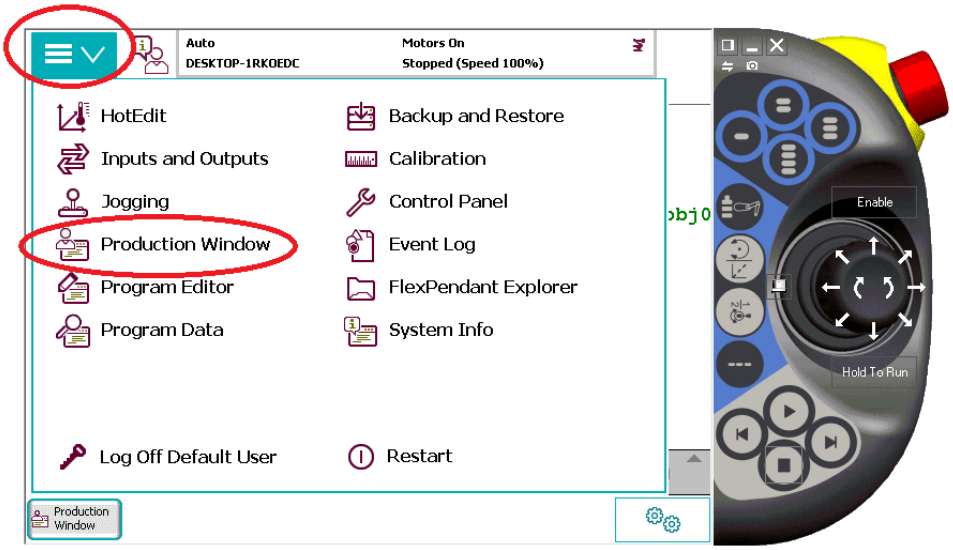


Fig. 9-6 – Acces meniu producție

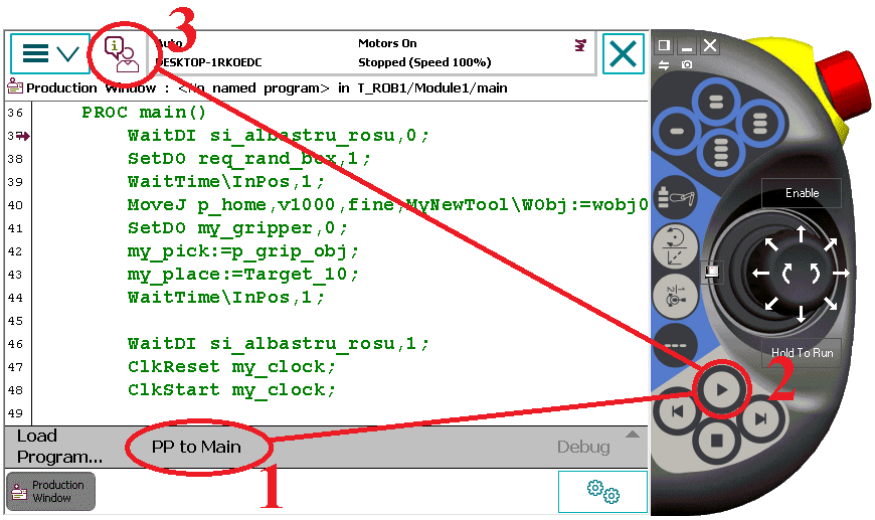


Fig. 9-7 – Setare pointer program la traiectoria Main (1), lansare în execuție (2) și acces la interfața de afisare a informațiilor robot (3)

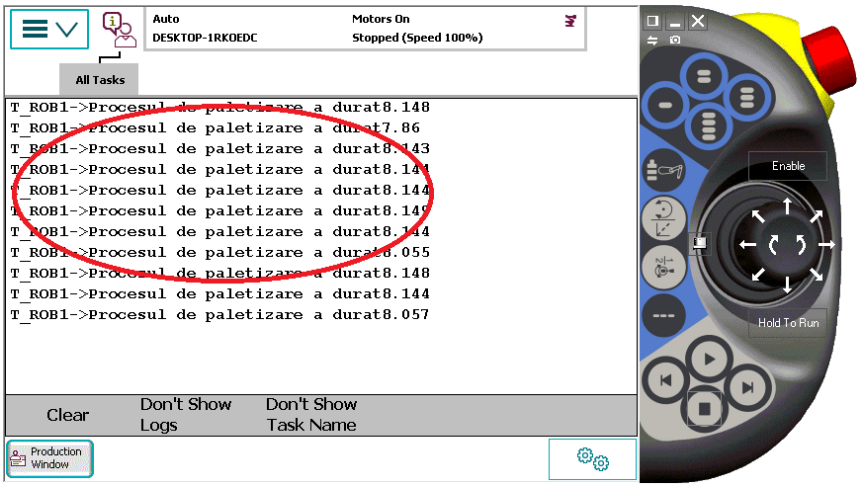


Fig. 9-8 – Vizualizare date scrise cu TPWrite

Bibliografie

- Programarea roboților, Borangiu Theodor, Dumitrache Alexandru, Anton Florin, Editura AGIR, seria Robotica si fabricatie integrata, Bucuresti, 2011, ISBN 978-973-720-337-3, 2011
- Robot modelling and simulation, Theodor Borangiu, Florin Ionescu, Editura Academiei Romana & Editura AgirISBN: ISNN Academy: 973-27-0927-8 & ISBN Agir: 973-8130-87-5, 2002
- John J. Craig; Introduction to Robotics, Mechanics and Control; Third Edition; Prentice Hall, ISBN 0-13-123629-6, 2005
- T. Lehtla. Introduction to robotics. TTU, Dept. of Electrical Drives and Power Electronics, Tallinn, 2008
- Technical reference manual. RAPID Instructions, Functions and Data types. RobotWare 6.06, Document ID: 3HAC050917-001, Revision: F, 2017
- Application manual MultiMove, RobotWare 6.08, Document ID: 3HAC050961-001, Revision: E, 2018
- Technical reference manual. System parameters, RobotWare 6.08, Document ID: 3HAC050948-001, Revision: J, 2018
- Technical reference manual. RAPID overview, RobotWare 6.08, Document ID: 3HAC050947-001, Revision: H, 2018 Application manual
- Controller software IRC5. RobotWare 6.08, Document ID: 3HAC050798-001, Revision: H, 2018
- Operating manual. RobotStudio, 2019.5, Document ID: 3HAC032104-001, Revision: AB, 2019