



# **Conducerea Roboților Industriali**

## **Proiect**

### **Paletizarea cuburilor în forme de litere folosind un braț robotic**

Student: Proaspătu Nicolae-Bogdan

Specializare: Automatică și Informatică Aplicată

An: III

# Cuprins

1. Introducere .....	3
1.1. Descrierea temei alese .....	3
1.2. Obiective propuse .....	3
1.3. Motivația alegerii temei.....	4
2. Tehnologii utilizate .....	5
2.1. URSim (Universal Robots Simulator).....	5
2.2. Brațul robotic Universal Robots (model UR5).....	6
2.3. Limbajul Polyscope / Script UR.....	6
2.4. Oracle VirtualBox.....	7
3. Realizarea proiectului.....	8
3.1. Structura generală a programului .....	8
3.2. Descrierea secvenței de trasare.....	8
3.3. Descrierea secvenței de paletizare.....	10
3.4. Mecanismul de selecție a literelor si declanșarea paletizării .....	11
3.5. Exemplu concret: scrierea si umplerea literei T .....	12
4. Concluzie .....	14
5. Bibliografie .....	15

# 1.Introducere

## 1.1. Descrierea temei alese

Tema acestui proiect se încadrează în domeniul roboticii industriale și urmărește dezvoltarea unei aplicații care simulează un proces de scriere și paletizare automată, cu posibilă aplicabilitate în industrii precum cea a ambalajelor, marcare logistică sau educație tehnologică.

Pornind de la ideea de automatizare a unor activități simple dar repetitive — cum ar fi inscripționarea de simboluri pe suprafețe și umplerea unor forme cu obiecte — proiectul propune programarea unui braț robotic capabil să scrie litere pe o suprafață plană și să plaseze obiecte (cuburi) în interiorul acestora. Un astfel de sistem ar putea fi extins în aplicații reale pentru personalizarea ambalajelor, sortarea obiectelor pe forme predefinite sau instruirea operatorilor în medii de formare profesională.

Implementarea este realizată inițial în simulatorul URSim, oferind un cadru sigur pentru testare, iar mai apoi transferată pe un robot fizic, unde se validează performanța mișcărilor și acuratețea plasării obiectelor. Literele sunt selectate în mod condiționat, iar robotul execută o secvență completă de acțiuni: preluarea unui marker, scrierea literei, plasarea markerului înapoi și umplerea formei cu cuburi.

Prin această temă, se urmărește modelarea unui proces automatizat care combină trasarea și manipularea, punând accent pe precizie, sincronizare și modularitate în controlul robotic.

## 1.2. Obiective propuse

Proiectul își propune realizarea unui sistem robotic funcțional care să combine două acțiuni esențiale: trasarea literelor pe o suprafață și umplerea acestora cu cuburi, simulând astfel un proces de manipulare automatizată cu aplicații posibile în medii educaționale, creative sau industriale.

Un prim obiectiv este dezvoltarea unui program modular și flexibil, care permite alegerea unei litere predefinite (L, I, T sau H) în funcție de condiții logice stabilite prin intrări digitale. În funcție de litera aleasă, robotul trebuie să urmeze o traiectorie precisă pentru a o scrie pe foaie, utilizând un marker atașat la gripper.

Un al doilea obiectiv major este implementarea unui mecanism de paletizare, prin care robotul preia cuburi dintr-o locație prestabilită și le plasează cu precizie în interiorul literei scrise, respectând poziții planificate anterior. Acest proces se realizează cu ajutorul comenzilor `MoveJ` și `MoveL`, pentru a optimiza atât siguranța, cât și eficiența mișcărilor robotului.

Pe parcursul execuției, robotul va utiliza comenzi specifice gripperului pentru a prinde (G40) și a elibera (G80) obiectele, iar toate acțiunile vor fi coordonate într-o structură logică bine definită, cu subprograme dedicate fiecărei litere și fiecărei etape de paletizare.

Proiectul are și un rol formativ, contribuind la înțelegerea și aplicarea practică a conceptelor de bază din programarea robotică: mișcări planificate, control condiționat, structuri modulare și manipulare de obiecte în spațiu tridimensional. Validarea se va face atât în simulatorul URSim, cât și pe robotul fizic, pentru a verifica acuratețea implementării în mediu real.

### 1.3. Motivația alegerii temei

Alegerea acestei teme a fost motivată de interesul personal pentru domeniul roboticii industriale și dorința de a înțelege concret modul în care un braț robotic poate fi programat să execute sarcini complexe. Proiectul oferă o oportunitate practică de aplicare a noțiunilor învățate în cadrul laboratoarelor și cursurilor de **Conducerea roboților industriali**. De asemenea, utilizarea atât a simulatorului URSim, cât și a robotului real permite o tranziție directă de la teorie la practică, evidențiind importanța testării și adaptării în condiții reale.

## 2. Tehnologii utilizate

În realizarea acestui proiect au fost utilizate o serie de tehnologii software și hardware, esențiale pentru programarea, simularea și testarea funcționalității brațului robotic. Acestea sunt prezentate mai jos:

### 2.1. URSim (Universal Robots Simulator)

URSim este un mediu de simulare oferit de Universal Robots, care permite testarea și dezvoltarea programelor robotice fără a utiliza robotul fizic. Simulatorul reproduce fidel interfața Polyscope a robotului, permițând încărcarea programelor .urp, definirea punctelor în spațiu, utilizarea instrucțiunilor de mișcare (MoveL, MoveJ) și simularea comenzilor pentru gripper. Este folosit pentru dezvoltare și testare inițială.

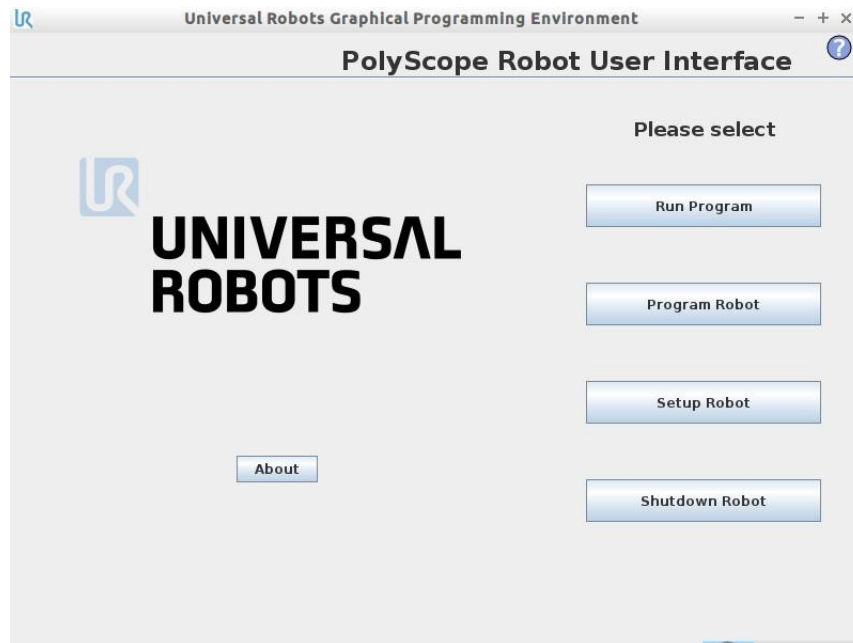


Figura 1. Universal Robots

## 2.2. Brațul robotic Universal Robots (model UR5)

În etapa practică a proiectului, programul este testat pe un robot real aflat în laborator (model UR), cu scopul de a valida comportamentul dezvoltat în simulator în condiții fizice reale. Programul realizat în URSim poate fi transferat și rulat direct pe robot, necesitând doar ajustări minore pentru optimizarea preciziei și adaptarea la particularitățile mediului fizic.



Figura 2. Brațul robotic UR5

## 2.3. Limbajul Polyscope / Script UR

Programarea se face direct în interfața Polyscope, utilizând limbajul specific UR, cu structuri precum IF, Call, MoveL, MoveJ, comenzi pentru gripper și funcții de așteptare (Wait). Acesta permite o structurare clară și modulară a programului, ușor de urmărit și adaptat.

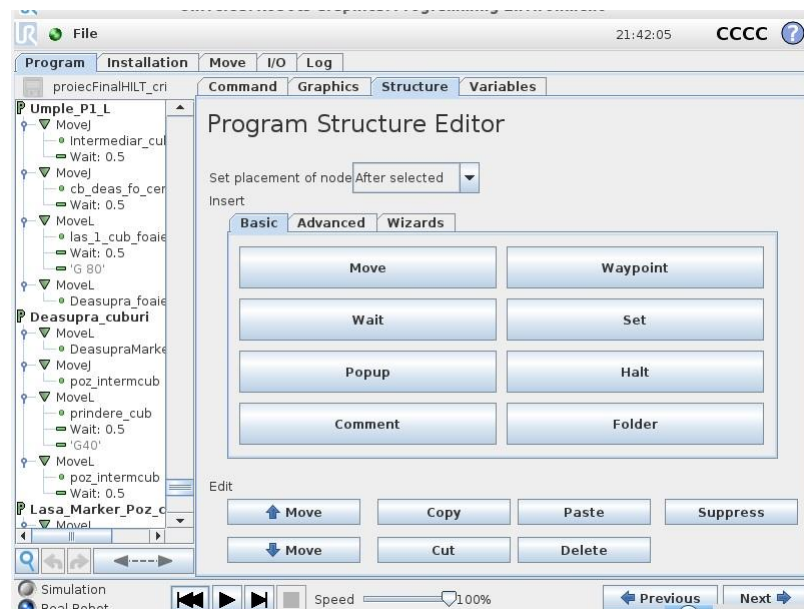


Figura 3. Secvența comenzi și diverse structuri

## 2.4. Oracle VirtualBox

Pentru rularea simulatorului URSim pe un sistem personal, a fost utilizată platforma Oracle VirtualBox. Aceasta permite rularea unei mașini virtuale cu Linux, în care este instalat URSim, asigurând un mediu stabil și portabil de testare.

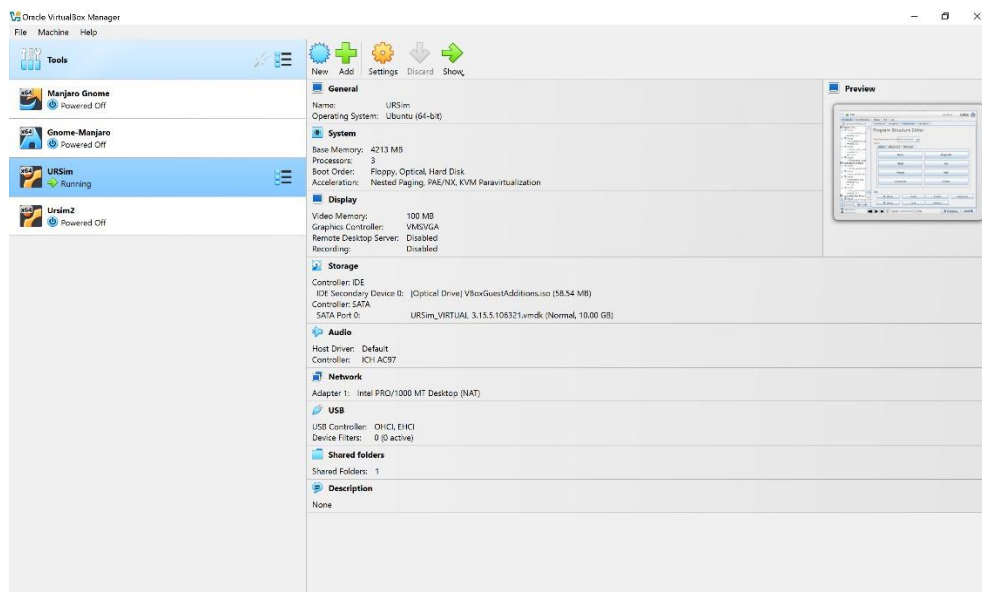


Figura 4. Oracle VirtualBox

### 3. Realizarea proiectului

#### 3.1. Structura generală a programului

Programul realizat este structurat modular, fiind împărțit în secvențe logice clare care reflectă etapele principale ale procesului: trasarea unei litere și umplerea acesteia cu cuburi. Structura generală are la bază un program principal în care sunt apelate, în funcție de condiții logice (IF), subprograme dedicate fiecărei litere.

Execuția începe cu deplasarea brațului robotic în poziția de preluare a markerului, urmată de ridicarea acestuia și deplasarea spre colțul foi de scris. În funcție de semnalul de intrare (digital\_in[x]) activat, este apelat subprogramul corespunzător literei (ex. Desen\_L, Desen\_T, Desen\_H, Desen\_i). După finalizarea scrierii, robotul lasă markerul într-o poziție sigură și trece în mod automat la etapa de paletizare.

Paletizarea constă în apeluri succesive către subprograme de tip Umple\_Px\_Y, fiecare corespunzând unei poziții unde trebuie plasat un cub. Preluarea cuburilor se face dintr-o zonă prestabilită, iar plasarea se realizează pe baza unor puncte definite anterior pe foaie.

Pentru controlul mișcărilor sunt folosite comenzile MoveL și MoveJ. Manipularea markerului și a cuburilor este realizată cu ajutorul comenzilor gripperului: G40, G10 pentru prindere și G80 pentru eliberare.

Această structurare modulară permite testarea individuală a fiecărei litere și a secvențelor de paletizare, precum și o extindere ușoară a proiectului cu litere sau forme noi.

#### 3.2. Descrierea secvenței de trasare

Secvența de trasare reprezintă prima etapă a execuției programului, în care robotul scrie automat o literă pe o foaie folosind un marker fixat în gripper. Procesul este inițiat cu deplasarea robotului în poziția Intermediar\_cub, urmată de mișcarea către preluare\_marker, unde gripperul se închide cu comanda G40 pentru a prinde markerul. După o scurtă pauză (Wait: 0.5), robotul ridică markerul prin mișcarea către poziția ridicare\_marker.

Din această poziție, robotul se deplasează către zona de lucru. Mai întâi, ajunge în deasupra\_colt\_f, apoi coboară în marker\_pe\_colt, punctul de start pentru scrierea literei.

Litera care urmează să fie trasată este determinată pe baza intrărilor digitale (digital\_in[1] până la digital\_in[5]).

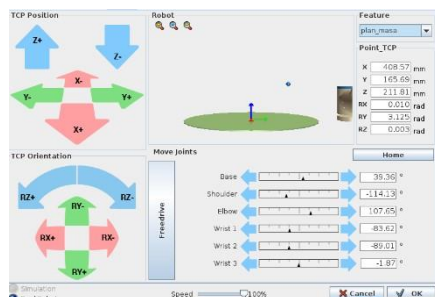


Figura 5. Coordonate intermediar\_cub

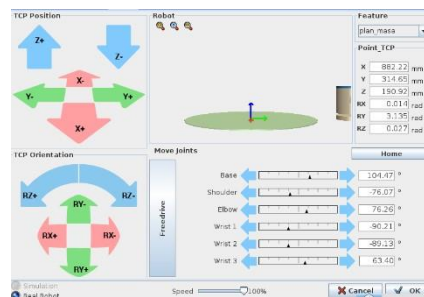


Figura 6. Coordonate deasupra\_colt\_f



Fiecare subprogram conține o secvență de comenzi MoveL, care controlează mișcarea markerului prin puncte intermediare. Toate aceste mișcări sunt de tip MoveL, pentru a asigura o trasare continuă și precisă. Alegerea mișcării liniare este importantă deoarece markerul trebuie să mențină contactul constant cu foaia în timpul desenului.

La finalul desenării, robotul execută subprogramul Lasa\_Marker\_Poz\_cub, unde se deplasează la DeasupraFoaie, DeasupraMarker și coboară în LasaMarker pentru a lăsa markerul cu G40. Apoi, gripperul se deschide (G80) și robotul se retrage în DeasupraMarker2.

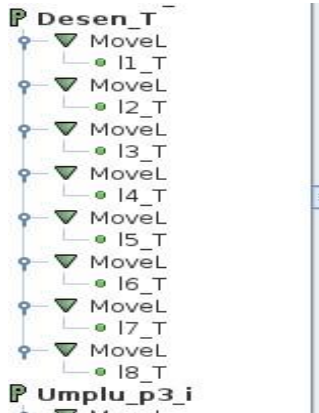


Figura 7. Subprogram\_Desen\_T

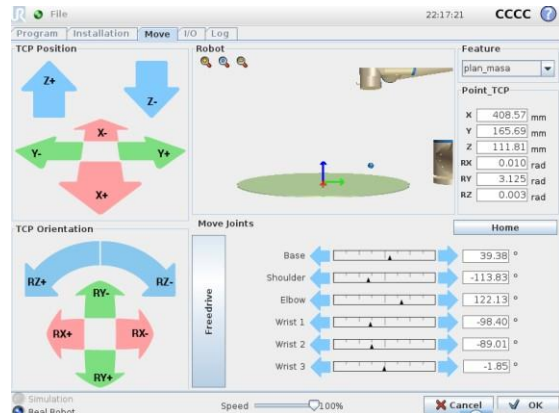


Figura 8. Coordonate Ia\_marker si Lasa\_marker

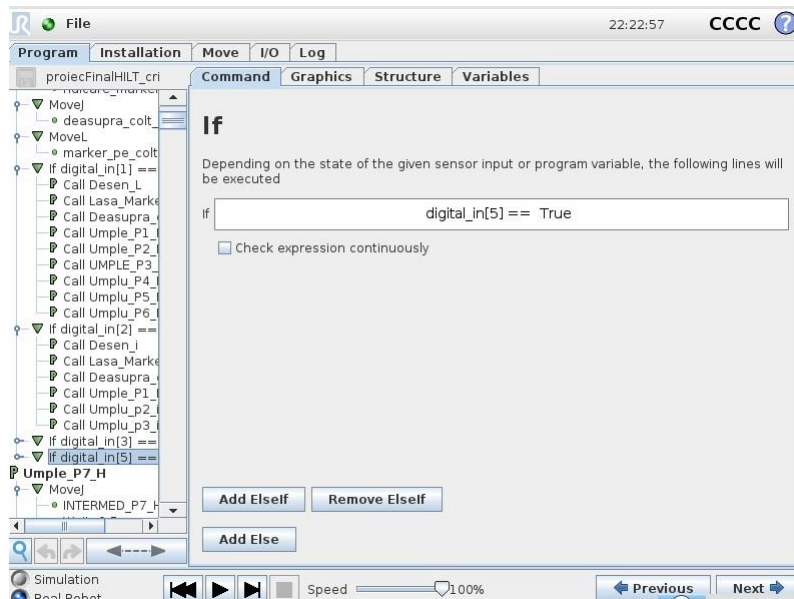


Figura 9. Conditionări\_if

### 3.3. Descrierea secvenței de paletizare

După finalizarea scrierii literei și plasarea markerului într-o poziție sigură, robotul trece automat în etapa de paletizare, care constă în preluarea și plasarea succesivă a unor cuburi în interiorul literei scrise. Această etapă are scopul de a umple forma trasată, simulând o acțiune de manipulare de precizie.

Procesul de paletizare începe cu subprogramul Deasupra\_cuburi, în care robotul se deplasează întâi în poziția DeasupraMarker2, apoi în poz\_intermcub, și în cele din urmă la prindere\_cub, unde gripperul prinde un cub cu comanda G40. După preluare, robotul se ridică înapoi la poz\_intermcub, evitând astfel coliziunile în timpul deplasării către zona de lucru.

Fiecare cub este plasat într-un punct bine definit în interiorul literei, în funcție de litera aleasă. De exemplu, pentru litera L, plasările se fac prin apeluri succesive către subprogramele Umple\_P1\_L, Umple\_P2\_L ... Umple\_P6\_L.

Fiecare dintre aceste subprograme urmează aceeași structură: mișcare de poziționare deasupra cubului (ex. INTERM\_P5\_L), coborâre la punctul de prindere (ex. PREL\_C5\_L), prindere cub (G40) și ridicare, deplasare la punctul deasupra foii (ex. DEAS\_P5\_L\_FOAIE), coborâre și eliberare (LAS\_P5\_L\_FOAIE, G80), retragere înapoi într-un punct intermediar. Pentru fiecare literă, punctele sunt diferite, dar logica este identică.

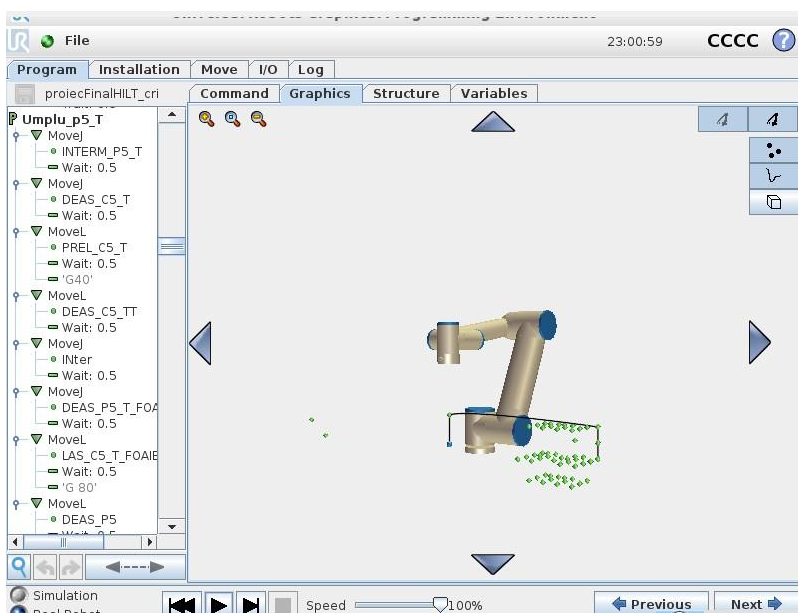


Figura 10. Subprogram cu succesiunea de mișcări necesare pentru umplerea poziției 5 din interiorul literei T

### 3.4. Mecanismul de selecție a literelor și declanșarea paletizării

După ce robotul ajunge cu markerul pe foaie urmează alegerea literei care trebuie desenată. Acest lucru se face în funcție de intrările digitale setate în program (digital\_in[1] pentru litera L, digital\_in[2] pentru litera i, digital\_in[3] pentru H și digital\_in[5] pentru T.

Selectarea literei se face prin activarea uneia dintre intrările digitale, fie direct din simulatorul URSim, fie prin apăsarea butoanelor de pe panoul fizic din laborator. Mai jos sunt prezentate cele două metode de selecție:

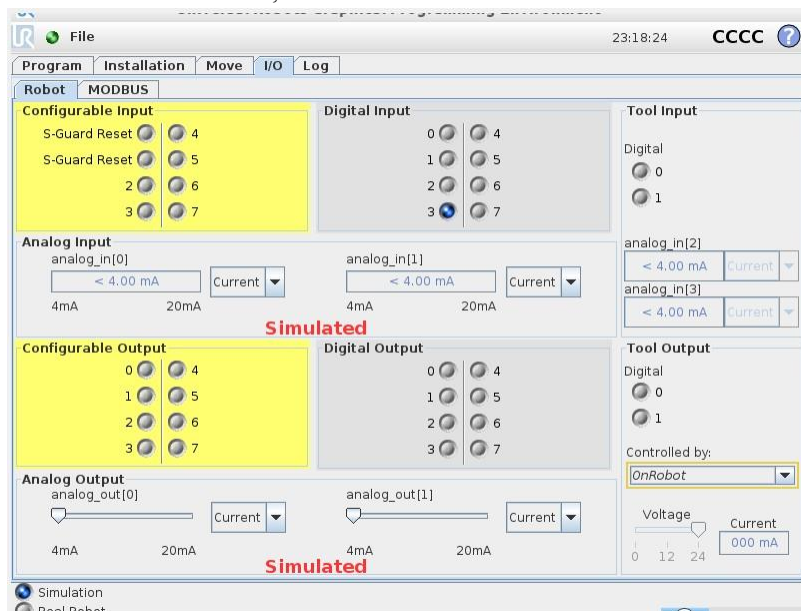


Figura 11. Panoul de intrări/ieșiri (I/O Tab) digitale din URSim.



Figura 12. Panou fizic de comutare a intrărilor și ieșirilor digitale utilizat în laborator

### 3.5. Exemplu concret: scrierea și umplerea literei T

Pas 1: Robotul merge la preluare\_marker, prinde markerul



Figura 13. Inceputul, asemănător pentru toate literele

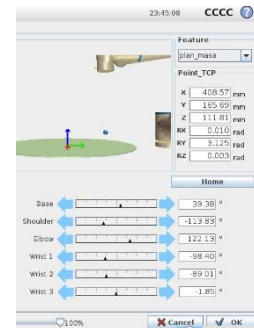


Figura 14. Coordonate preluare\_marker

Pas 2: Se deplasează spre marker\_pe\_colt, incepe trasarea

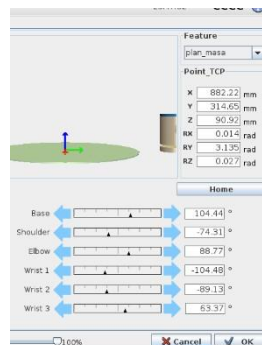


Figura 15. Coordonate marker\_pe\_colt

Pas 3: Este activ digital\_in[5], așadar apelează Desen\_T (punctul de pornire al desenului este același cu cel final)

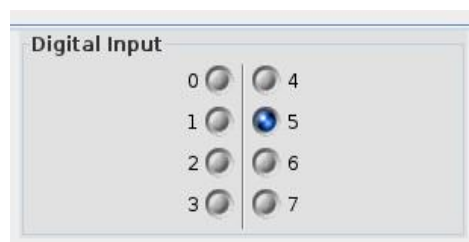


Figura 16. digital\_in[5] == TRUE

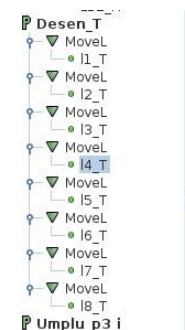


Figura 17. Subprogram Desen\_L

Pas 4: Markerul este lasat in LasaMarker, apelandu-se Lasa\_Marker\_Poz\_cub.

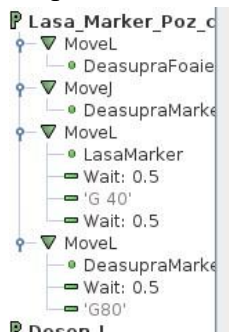


Figura 18. Subprogram Lasa\_Marker\_Poz\_cub

Pas 5: Apelează Deasupra\_cuburi, așadar preia cuburi

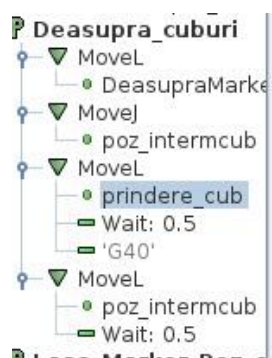


Figura 19. Subprogram Deasupra\_cuburi



Figura 20. Coordonate preluare cuburi

Pas 6: Ruleaza Umple\_P1\_T, Umple\_P2\_T, Umple\_P3\_T, Umple\_P4\_T, Umple\_P5\_T, Umple\_P6\_T.

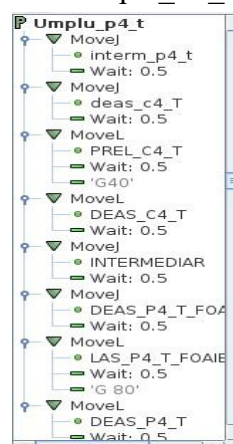


Figura 20. Subprogram Umple\_p4\_L

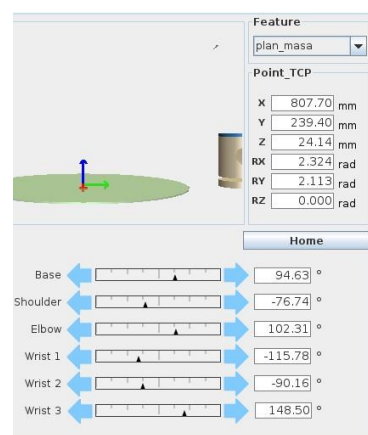


Figura 21. Coordonate lăsare cub în poziția P4 pentru T

## 4. Concluzie

Proiectul realizat a demonstrat modul în care un braț robotic poate fi programat să execute sarcini secvențiale și condiționate, într-un mod automatizat și controlat. Prin combinarea scrierii unei litere cu plasarea precisă a unor cuburi în interiorul acesteia, s-au aplicat concepte esențiale din programarea robotică: mișcări planificate (MoveL, MoveJ), controlul gripperului, lucrul cu planuri de coordonate și utilizarea structurii condiționale IF.

Un alt aspect important a fost testarea în simulatorul URSim, care a permis ajustarea punctelor, verificarea logicii și evitarea coliziunilor. Ulterior, programul a fost pregătit pentru rularea pe robotul fizic, ceea ce a confirmat funcționalitatea și robustețea soluției.

Dificultățile principale au apărut în stabilirea coordonatelor exacte și în sincronizarea mișcărilor cu acțiunile gripperului, însă acestea au fost rezolvate prin testare repetată și ajustări în simulator. De asemenea, organizarea modulară a programului, prin subprograme separate pentru fiecare literă și fiecare punct de paletizare, a ușurat mult procesul de dezvoltare și depanare.

Proiectul rămâne deschis pentru extindere, prin adăugarea altor litere sau forme geometrice, optimizarea traiectoriilor sau integrarea unor senzori care să adapteze comportamentul robotului în timp real.

## 5. Bibliografie

1. Universal Robots. *URScript Programming Language – User Manual*, Universal Robots A/S.
2. Universal Robots. *Polyscope GUI User Manual*, Universal Robots A/S.
3. Pagina oficială: <https://www.universal-robots.com>.