

# CERCETĂRI PRIVIND PROIECTAREA ȘI REALIZAREA UNUI ROBOT DE PALETIZARE

## RESEARCH ON THE DESIGN AND CONSTRUCTION OF A PALLETIZING ROBOT

STOICA Raluca Georgiana, STOICA Radu Ionuț, Oțelea Răzvan Florin  
Facultatea: I.I.R., Specializarea: Robotică, Anul de studii: II, (e-mail: [ralucastoica123@gmail.com](mailto:ralucastoica123@gmail.com))

Conducători științifici: Ș.I. dr.ing. **Marinela MARINESCU**, Ș.I. dr.ing. **Larisa BUȚU**

*REZUMAT: În această lucrare sunt prezentate etapele proiectării, realizării și programării unui robot industrial de gabarit mic, folosit pentru paletizarea produselor. Robotul poate fi folosit în scop didactic, în lucrările de laborator, pentru demonstrații practice privind mișcările efectuate de un astfel de robot. Pentru programarea robotului s-a folosit codul ARDUINO.*

*ABSTRACT: In this paper are presented the stages of design, realization and programming of a small industrial robot, used for palletizing products. The robot can be used for didactic purposes, in laboratory works, for practical demonstrations on the movements performed by such a robot. The ARDUINO code was used to program the robot.*

*CUVINTE CHEIE: robot paletizare, cod ARDUINO*

### 1. Introducere

Scopul lucrării este de a proiecta un robot de paletizare cu sarcina portantă mică pentru a putea fi utilizat în aplicații de laborator sau în alte scopuri didactice (pregătirea studenților/elevilor pentru viitoare concursuri de robotică).

Din punctul nostru de vedere, acest proiect poate fi un bun punct de reper în înțelegerea funcționării aplicației de paletizare, de asemenea fiind posibilă și înțelegerea avansată a proiectării unui astfel de robot.

### 2. Stadiul actual

Problema analizată de către noi este în stadiul de implementare și testare, în acest moment încercăm să testăm mai multe programe pe care le-am încărcat pe placa de bază ARDUINO UNO, unele fiind mai complexe. Programul conceput de noi, este funcțional, robotul putând face operații de paletizare și manipulare simple.

### 3. Proiectarea, realizarea și programarea robotului de paletizare

#### 3.1. Scurtă prezentare a operației de paletizare

Paletizarea reprezintă operația de dispunere volumică ordonată, în plan orizontal (sub formă de straturi cu înălțime omogenă) și pe verticală (sub formă de straturi multiple) pe dispozitive de transport standardizate denumite paleți, a diferitelor categorii de obiecte (produse ambalate în cutii de carton cu formă paralelipipedică, saci cu materiale vrac de tip granule sau pulberi, seturi de obiecte multiple preinfoliate – sticle cu apă / ulei etc.) manipulate individual sau în grup de către roboți industriali sau mașini automate de paletizare. Paletizarea se realizează pe paleți cu dimensiuni reglementate prin

standarde internaționale pentru a se facilita unificarea condițiilor de stocare, transport și manipulare a acestora. Efectorii utilizați în aplicațiile de paletizare au o construcție specială, aceasta fiind aleasă în funcție de forma, dimensiunile și caracteristicile de rigiditate a obiectelor de manipulat. Pentru aplicația noastră, efectorul este de tip “GRIPPER” și poate efectua operații de paletizare obiecte de dimensiuni mici și de masa mică sau medie.

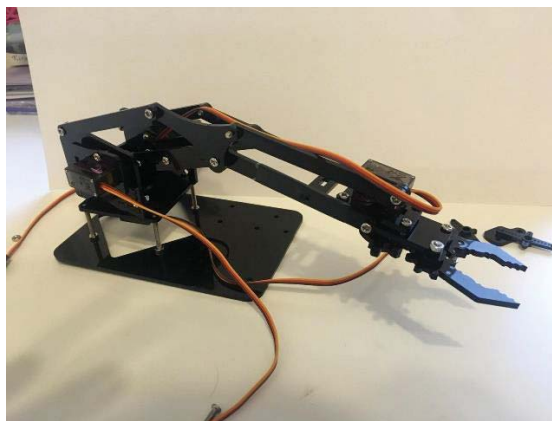


Fig. 1. Robotul asamblat.

Pentru calculul șuruburilor s-au folosit formulele:

$$\sigma_{ac} = \frac{R_{p02}}{c} \quad , \quad A_{nec} = \frac{F_c}{\sigma_a} = \frac{\pi \cdot d_3^2}{4} = \frac{\gamma \cdot F_c}{\sigma_a}; (1)$$

$$d = \sqrt{\frac{4 \cdot \gamma \cdot F}{\pi \cdot \sigma_a}} \quad , \quad tg \Psi = \frac{p}{\pi \cdot d_2} \quad , \quad tg \phi' = \frac{\mu}{\cos \frac{\alpha}{2}}; (2)$$

$$l_f = L, \quad \lambda = \frac{l_f}{i_{min}}, \quad i_{min} = \sqrt{\frac{I_{min}}{A}}, \quad A = \frac{\pi \cdot d_3^2}{4}, (3)$$

$$I_{min} = \frac{\pi \cdot d_3^4}{64}, \quad c_f = \frac{F_{cr}}{F} \leq c_{fa}, \quad Z = \frac{F}{\frac{\pi}{4} \cdot (d^2 - D^2) \cdot \sigma_{as}} (4)$$

$$\sigma_i = \frac{M_i}{W_y}, \quad \tau_f = \frac{F_1}{A}, \quad \sigma_{echv} = \sqrt{\sigma_i^2 + 4 \cdot \tau_f^2} \leq \sigma_a, (5)$$

$$M_{1,2} = F_1 \cdot tg(\Psi + \phi') = M_{2,1}, \quad \sigma = \frac{F}{\frac{\pi \cdot d_3^2}{4}}, \quad \tau = \frac{M_{1,2}}{\frac{\pi \cdot d_3^3}{16}} (6)$$

$$\sigma_{echv} = \sqrt{\sigma_i^2 + 4 \cdot \tau_f^2} \leq \sigma_a (7)$$

### 3.2. Programarea robotului

Robotul nostru este un robot de tip braț articulat, de gabarit mic, având 4 cuple de rotație. Fiecare din acesta este acționat de un servomotor. Servomotoarele sunt acționate prin intermediul unei plăci de bază ARDUINO, pe care am achiziționat-o de pe site-ul oficial al firmei. O dată cu achiziționarea plăcuței de bază, am ales tot de la firma ARDUINO servomotoarele compatibile cu această placă de bază și cu aplicația noastră. Programul a fost scris tot în acest limbaj de programare, după cum urmează:

```

#include <Servo.h>
#include <OLEDD_I2C.h>

//define the servos
Servo servo1;
Servo servo2;
Servo servo3;
Servo servo4;

OLEDD myOLEDD(SDA, SCL);
extern uint8_t SmallFont[];

//define the buttons
const int button1 = 12;
const int button2 = 13;

//define variable for values of the button
int button1Pressed = 0;
boolean button2Pressed = false;

//define potentiometers
const int pot1 = A1;
const int pot2 = A2;
const int pot3 = A3;
const int pot4 = A4;

//define variable for values of the potentiometers
int pot1Val;
int pot2Val;
int pot3Val;
int pot4Val;

//define variable for angles of the potentiometer
int pot1Angle;
int pot2Angle;
int pot3Angle;
int pot4Angle;

//define variable for saved position of the servos
int servo1PosSave[] = {1,1,1,1,1,1,1,1};
int servo2PosSave[] = {1,1,1,1,1,1,1,1};
int servo3PosSave[] = {1,1,1,1,1,1,1,1};
int servo4PosSave[] = {1,1,1,1,1,1,1,1};

void setup() {
  myOLEDD.begin();
  myOLEDD.setFont(SmallFont);
  Serial.begin(9600);

  //define attached pins of the servos
  servo1.attach(3);
  servo2.attach(4);

  myOLEDD.begin();
  myOLEDD.setFont(SmallFont);
  Serial.begin(9600);

  //define attached pins of the servos
  servo1.attach(3);
  servo2.attach(4);

  //define buttons as input units
  pinMode(button1, INPUT);
  pinMode(button2, INPUT);

  servo1.write(90);
  servo2.write(90);
  servo3.write(90);
  servo4.write(90);

  myOLEDD.clearScreen();
  myOLEDD.print("MERT ARDUINO AND TECH", CENTER, 8);
  myOLEDD.print("PRESS TO SAVE BUTTON", CENTER, 42);
  myOLEDD.update();
}

void loop() {
  //read the potentiometer values and define the servo angle to
  //the potentiometer value with the map function
  pot1Val = analogRead(pot1);
  pot1Angle = map(pot1Val, 0, 1023, 10, 179);
  pot2Val = analogRead(pot2);
  pot2Angle = map(pot2Val, 0, 1023, 10, 150);
  pot3Val = analogRead(pot3);
  pot3Angle = map(pot3Val, 0, 1023, 10, 170);
  pot4Val = analogRead(pot4);
  pot4Angle = map(pot4Val, 0, 1023, 10, 170);

  //servos move to mapped angles
  servo1.write(pot1Angle);
  servo2.write(pot2Angle);
  servo3.write(pot3Angle);
  servo4.write(pot4Angle);

  //if button1 is pressed (HIGH), save the potentiometers position
  //as long as button1 is pressed
  if(digitalRead(button1) == HIGH){
    button1Pressed++;
    switch(button1Pressed){
      case 1:
        servo1PosSave[0] = pot1Angle;
        servo2PosSave[0] = pot2Angle;
        break;
      case 2:
        servo1PosSave[1] = pot1Angle;
        servo2PosSave[1] = pot2Angle;
        servo3PosSave[1] = pot3Angle;
        servo4PosSave[1] = pot4Angle;
        Serial.println("Position #2 Saved");
        myOLEDD.clearScreen();
        myOLEDD.print("MERT ARDUINO AND TECH", CENTER, 8);
        myOLEDD.print("POSITION #2 SAVED", CENTER, 42);
        myOLEDD.update();
        delay(1500);
        break;
      case 3:
        servo1PosSave[2] = pot1Angle;
        servo2PosSave[2] = pot2Angle;
        servo3PosSave[2] = pot3Angle;
        servo4PosSave[2] = pot4Angle;
        Serial.println("Position #3 Saved");
        myOLEDD.clearScreen();
        myOLEDD.print("MERT ARDUINO AND TECH", CENTER, 8);
        myOLEDD.print("POSITION #3 SAVED", CENTER, 42);
        myOLEDD.update();
        delay(1500);
        break;
      case 4:
        servo1PosSave[3] = pot1Angle;
        servo2PosSave[3] = pot2Angle;
        servo3PosSave[3] = pot3Angle;
        servo4PosSave[3] = pot4Angle;
        Serial.println("Position #4 Saved");
        myOLEDD.clearScreen();
        myOLEDD.print("MERT ARDUINO AND TECH", CENTER, 8);
        myOLEDD.print("POSITION #4 SAVED", CENTER, 42);
        myOLEDD.update();
        delay(1500);
        break;
      case 5:
        servo1PosSave[4] = pot1Angle;
        servo2PosSave[4] = pot2Angle;
        servo3PosSave[4] = pot3Angle;
        servo4PosSave[4] = pot4Angle;
        Serial.println("Position #5 Saved");
        myOLEDD.clearScreen();
        myOLEDD.print("MERT ARDUINO AND TECH", CENTER, 8);
        myOLEDD.print("POSITION #5 SAVED", CENTER, 42);
        myOLEDD.update();
        delay(1500);
        break;
      case 6:
        servo1PosSave[5] = pot1Angle;
        servo2PosSave[5] = pot2Angle;
        servo3PosSave[5] = pot3Angle;
        servo4PosSave[5] = pot4Angle;
        Serial.println("Position #6 Saved");
        myOLEDD.clearScreen();
        myOLEDD.print("MERT ARDUINO AND TECH", CENTER, 8);
        myOLEDD.print("POSITION #6 SAVED", CENTER, 42);
        myOLEDD.update();
        delay(1500);
        break;
      case 7:
        servo1PosSave[6] = pot1Angle;
        servo2PosSave[6] = pot2Angle;
        servo3PosSave[6] = pot3Angle;
        servo4PosSave[6] = pot4Angle;
        Serial.println("Position #7 Saved");
        myOLEDD.clearScreen();
        myOLEDD.print("MERT ARDUINO AND TECH", CENTER, 8);
        myOLEDD.print("POSITION #7 SAVED", CENTER, 42);
        myOLEDD.update();
        delay(1500);
        break;
      case 8:
        servo1PosSave[7] = pot1Angle;
        servo2PosSave[7] = pot2Angle;
        servo3PosSave[7] = pot3Angle;
        servo4PosSave[7] = pot4Angle;
        Serial.println("Position #8 Saved");
        myOLEDD.clearScreen();
        myOLEDD.print("MERT ARDUINO AND TECH", CENTER, 8);
        myOLEDD.print("POSITION #8 SAVED", CENTER, 28);
        myOLEDD.print("PRESS TO MOVE BUTTON", CENTER, 42);
        myOLEDD.update();
        delay(1500);
        break;
    }
  }

  //if button2 pressed (HIGH), the servos move saved position
  if(digitalRead(button2) == HIGH){
    button2Pressed = true;
  }

  if(button2Pressed){
    for(int i=0; i<8; i++){
      servo1.write(servo1PosSave[i]);
      servo2.write(servo2PosSave[i]);
      servo3.write(servo3PosSave[i]);
      servo4.write(servo4PosSave[i]);
      myOLEDD.clearScreen();
      myOLEDD.print("MOVING...", CENTER, 42);
      myOLEDD.update();
      delay(2000);
    }
  }
  delay(100);
}

```

Fig.2. Codul ARDUINO folosit pentru programarea Robotului de paletizare.

### 3.3. Proiectarea și realizarea pieselor componente ale robotului de paletizare

În ceea ce privește proiectarea robotului, am folosit soft-ul de desen AutoCAD, în care am realizat desenele de execuție ale pieselor componente ale robotului (fig. 3 – 13).

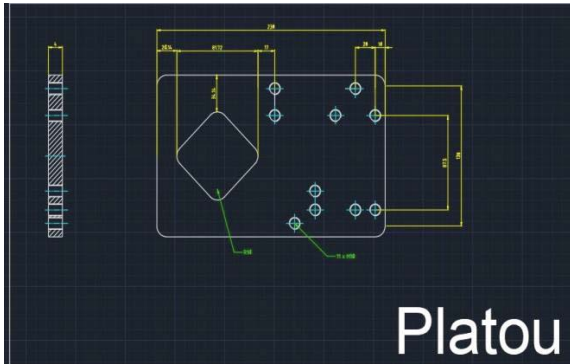


Fig.3. Placa de bază a robotului;

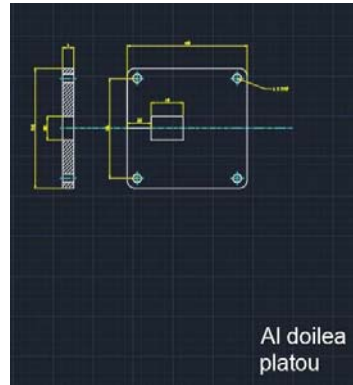


Fig.4. Al doilea platou.

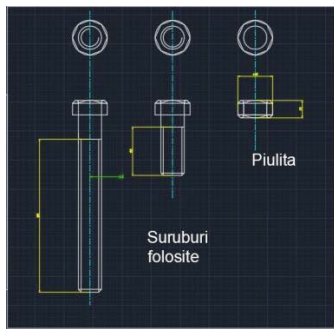


Fig.5. Șuruburile și piulițele folosite;

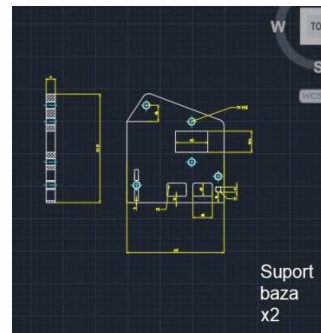


Fig.6. Suport bază;

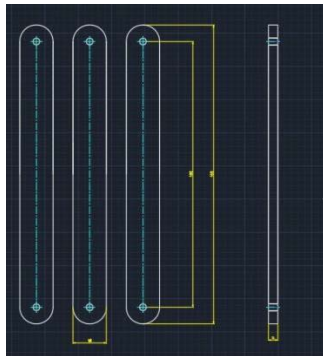


Fig. 7. Componentele lanțului cinematic.

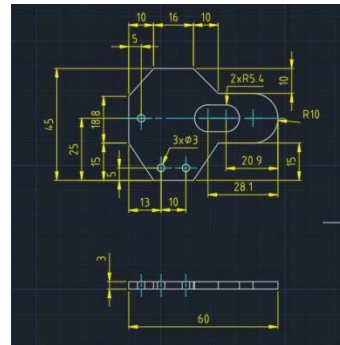


Fig. 8. Support gripper.

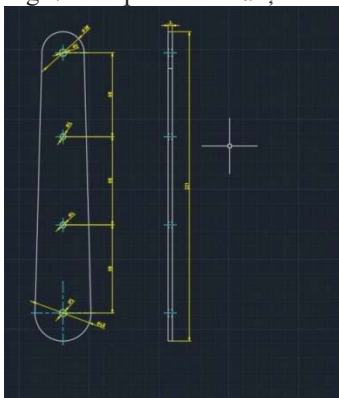


Fig. 9. Componentele lanțului cinematic;



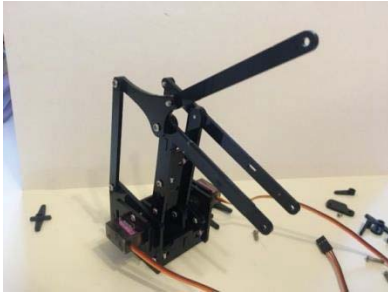


Fig. 18. Adaugarea celui de-al doilea segment.



Fig. 19. Rezultatul asamblării fără end-effector;



Fig. 20. Rezultatul final al asamblării.



Fig. 21. Testarea programului ARDUINO.

#### 4. Concluzii

În concluzie, robotul proiectat de noi poate îndeplini un scop didactic, ajutând astfel aplicațiile de laborator și viitorii specialiști în domeniul roboticii, acesta fiind unul dintre cele mai simple exemple de roboți industriali.

#### 5. Bibliografie

- [1]. Paduraru, G., “Sisteme cu șuruburi de mișcare”;
- [2]. Nicolescu, A., „Curs-Robotică 2”;
- [3]. Site-ul oficial ARDUINO: <https://www.arduino.cc/>;
- [4]. Gafitianu, M., „Organe de mașini”;
- [5]. Buzdugan, G., „Rezistența materialelor”;

#### 6. Notății

Următoarele simboluri sunt utilizate în cadrul lucrării:

$\sigma_{ac}$ =tensiunea normală admisibilă la curgere;

$l_f$ =lungimea de flambaj;

$\sigma_i$ =tensiunea normală de încovoiere;

$\tau_f$ =tensiunea tangențială de forfecare;

$I_{min}$ =momentul de inerție minim;

$i_{min}$ =raza de inerție minimă;

$M_{1,2}$ =momentul de înșurubare;

$M_{2,1}$ =momentul de deșurubare;

$d$ =diametrul șurubului;

$F_c$ =forța critică;

$A_{nec}$ =aria de calculat.