UNIVERSITATEA TEHNICĂ
DIN CLUJ-NAPOCA
FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE

Program pentru comanda unui utilaj cu comandă numerică

Damian Mihai Sebastian

Universitatea Tehnica din Cluj-Napoca

15-10-2023

Cuprins

1. Introducere

1.1 Context

Mașina de tăiat cu flacără este un dispozitiv specializat utilizat pentru tăierea metalelor folosind o flacără de oxigen. Această metodă de tăiere este cunoscută și sub denumirea de "tăiere cu flacără oxiacetilenică" și reprezintă una dintre tehnicile tradiționale de prelucrare a metalelor.

Principiul de funcționare al mașinii de tăiat cu flacără implică utilizarea unei flăcări pentru a încălzi rapid metalul la temperaturi ridicate, iar apoi un flux de oxigen este dirijat spre zona încălzită pentru a genera o reacție chimică de oxidare a metalului. Aceasta determină eliminarea materialului topit și crearea unei adâncituri sau decupaje în metal.

Mașinile de tăiat cu flacără pot fi folosite pentru tăierea unei game variate de metale, inclusiv oțel carbon, oțel inoxidabil și aluminiu. Acestea au aplicații extinse în industria metalurgică, construcții, reparații navale, ateliere de prelucrare a metalului, precum și în operațiuni de tăiere și prelucrare a metalului la scară industrială.

Pentru a controla precis operațiunile de tăiere, mașinile de tăiat cu flacără pot fi echipate cu sisteme CNC (Computer Numerical Control), care permit programarea și automatizarea mișcărilor și a parametrilor de tăiere, asigurând astfel o prelucrare mai precisă și eficientă a metalelor.

1.2 Specificații

Cu ajutorul mediului integrat de dezvoltare (IDE) IntelliJ Idea, voi dezvolta o interfață grafică (GUI) și un program capabil să genereze un G-code. Prin intermediul acestui program, utilizatorul va avea posibilitatea de a configura și controla desenarea prin intermediul unei mașini CNC.

Pentru a realiza desenarea cu mașina CNC, voi folosi mediul integrat de dezvoltare dedicat plăcuței Arduino. Această mașină CNC este echipată cu două motoare pas cu pas și un servomotor, care permit controlul precis asupra mișcării pe axele X, Y și Z. Pentru a activa aceste motoare, se va utiliza scutul de antrenare a motorului L293D și placa Arduino Mega.

Pentru a realiza desenul dorit cu această mașină CNC, este necesară obținerea G-code-ului corespunzător caracterului sau figurii pe care dorim să o reproducem.

1.3 Obiective

În limbajul Java, voi dezvolta o interfață grafică (GUI) care va permite utilizatorului să selecteze tipul de desen pe care dorește să-l creeze. Utilizatorul va avea opțiunea de a alege între desenarea unui segment și desenarea unui arc de cerc prin intermediul unor butoane dedicate. Prin apăsarea acestor butoane, se va iniția procesul de desen, unde fiecare click corespunzător butonului va adăuga un segment sau un arc de cerc, în funcție de alegerea făcută.

În plus, va fi implementat un al treilea buton, care va permite utilizatorului să salveze desenul în formatul ".txt" sub forma unui G-code. Acest G-code va putea fi ulterior transmis unei mașini fizice prin intermediul mediului integrat de dezvoltare (IDE) dedicat plăcuței Arduino. Prin intermediul acestei funcționalități, desenul generat în GUI va fi reprodus pe o foaie de hârtie sau pe alt suport similar cu ajutorul mașinii respective.

De asemenea, intenționez să integrez în aplicație un buton dedicat funcționalității de ștergere a desenului curent de pe planșa de desenare, facilitând astfel posibilitatea de a reface desenul fără necesitatea reluării aplicației la fiecare inițiere a unei noi creații.

2. Studiu biografic

Codul G, cunoscut sub denumirea de G-code, constituie un limbaj de programare fundamental utilizat pentru controlul mașinilor CNC (Computer Numerical Control). G-code constă într-o serie de comenzi sau instrucțiuni precise care guvernează mișcarea, viteza și operațiunile mașinilor CNC. Acest limbaj de programare este de o importanță crucială în programarea și controlul precis al mașinilor CNC într-o varietate de aplicații, inclusiv prelucrarea, gravura, tăierea și tehnologia de imprimare 3D.

Linia unui comenzi in G-code are următoarea structura*[2]*: G## X## Y## Z## F##

Un exemplu de linie a unei comenzi: **G01 X247 Y11 Z-1 F400**

- În primul rând, prima comandă folosită este cea de tip G-code, și anume G01, care semnifică „deplasare în linie dreaptă către o poziție specifică”.
- Urmează declararea poziției sau coordonatelor prin specificarea valorilor pentru axele X, Y și Z.
- În concluzie, prin intermediul valorii F, stabilim viteza de avans sau viteza la care mișcarea va fi executată.

Pentru a obține o înțelegere mai detaliată a exemplului, linia G01 X247 Y11 Z-1 F400 furnizează mașinii CNC instrucțiuni precise pentru efectuarea unei deplasări în linie dreaptă de la poziția sa curentă către coordonatele X247, Y11 și Z-1 mm, cu o viteză de 400 mm/min.

3. Analiza

Mai jos voi detalia toate comenzile de tip G-code pe care le voi genera prin intermediul interfeței grafice (GUI) dezvoltate în limbajul Java.

- **G00 - Poziționare rapidă**

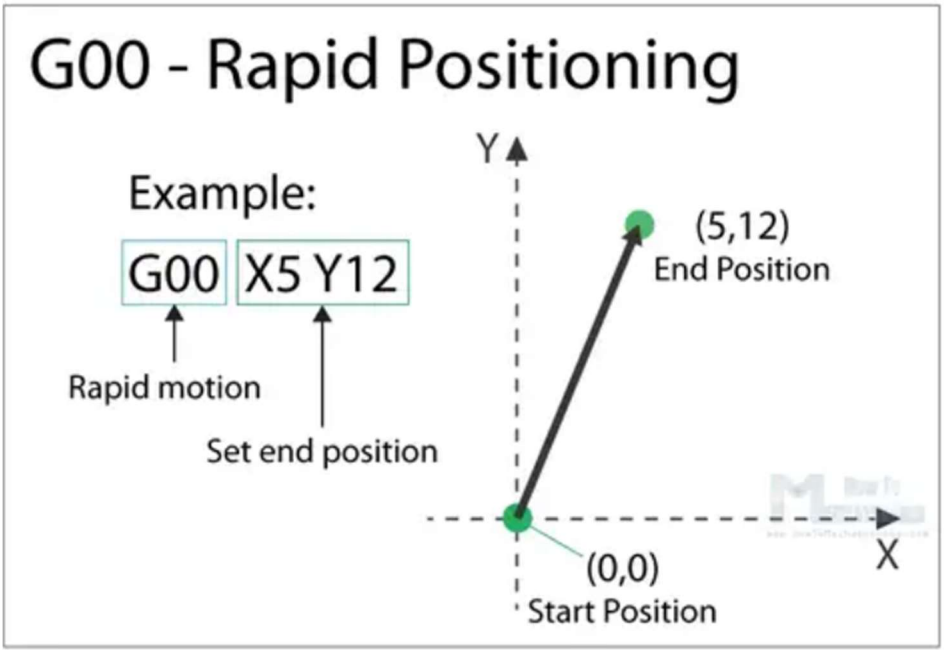

Figura 1 Comanda G00 ([2])

Scopul acestor instrucțiuni constă în a muta cursorul la o poziția specifică, fără a efectua acțiunea de “tăiere” , în contextul proiectului meu de scriere pe hârtie.

- **G01 - Interpolare liniară**

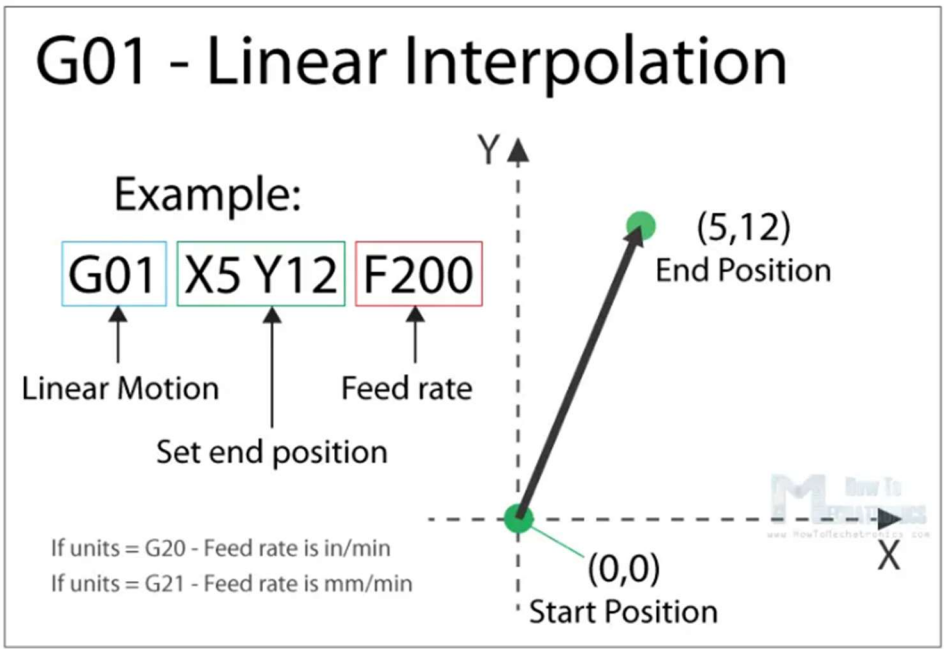

Figura 2 Comanda G01 ([2])

Scopul acestor instrucțiuni constă în a deplasa cu precizie cursorul la o poziție specifică și a efectua operația de "tăiere", în cadrul proiectului meu, care implică scrierea pe hârtie.

- **G21 - Această comandă indică setarea unităților de măsură la milimetri.** Atunci când G21 este activat, toate coordonatele introduse în G-code sunt interpretate ca fiind exprimate în milimetri.
- **G90 - Această comandă indică setarea modului de poziționare absolută.** Atunci când este activat modul G90, toate coordonatele introduse în G-code sunt interpretate ca fiind poziții absolute în raport cu sistemul de coordonate al mașinii CNC.
- **G92 - Această comandă este folosită pentru a seta poziția curentă a mașinii la o valoare specificată.**
  Este adesea folosită pentru a reinițializa sau pentru a ajusta poziția curentă a sistemului de coordonate. De exemplu, G92 X0 Y0 Z0 ar putea seta poziția curentă a mașinii la originea sistemului de coordonate.
- **G4 P150 - Această comandă indică o pauză în execuția programului pentru o perioadă specificată.**
  În acest caz, G4 P150 specifică o pauză de 150 de milisecunde.
- **M300 S30 – Unealta de scris se coboară**
- **M300 S50 - Unealta de scris se ridică**

- **M18 - Această comandă dezactivează toate motoarele pe mașină.**

Este utilizată pentru a opri motoarele și pentru a împiedica deplasarea mașinii.

- **M17 - Această comandă activează toate motoarele pe mașină.**
  Este utilizată pentru a porni motoarele și pentru a permite deplasarea mașinii în funcție de comenzile G-code ulterioare.
- **M01 - Această comandă indică un punct de oprire programat (stop opțional).**
  Atunci când M01 este întâlnit într-un program G-code, execuția programului se va opri, și operatorul va fi întrebat dacă dorește să continue sau să ignore această oprire programată.

Interacțiunea utilizatorului cu aplicația de desenare și generare a G-code-ului se va realiza prin intermediul interfeței grafice realizate prin IDE-ul IntelliJ, procesul desfășurându-se astfel:

1. Utilizatorul va utiliza click-ul stâng al mouse-ului pentru a plasa puncte pe planșa de desen.

2. După ce utilizatorul este mulțumit de desenul realizat prin intermediul punctelor, el apasă butonul corespunzător pentru desenarea segmentelor sau pe butonul dedicat desenării arcelor de cerc.

3. Odată ce segmentele și/sau arcele de cerc sunt afișate pe ecran, utilizatorul poate relua pașii 1 și 2 pentru a modifica desenul.

4. La finalizarea desenului, utilizatorul apasă butonul de generare a G-code, ceea ce generează un fișier G-code bazat pe punctele desenate.

5. După generarea G-code-ului, utilizatorul poate apăsa butonul pentru ștergerea desenului curent de pe planșa de desenare, iar acest buton poate fi accesat în orice moment. Acest aspect este util, deoarece utilizatorul poate dori să refacă desenul.

Interacțiunea utilizatorului cu aplicația de transmitere a G-code-ului către CNC se va realiza prin intermediul interfeței grafice realizate prin IDE-ul Processing, procesul desfășurându-se astfel:

1. Utilizatorul inițiază procesul selectând portul serial corespunzător prin care este conectat la placa Arduino.
2. După aceasta utilizatorul este apoi solicitat să aleagă fișierul G-code pe care dorește să-l transmită către CNC.
3. După ce a selectat fișierul, utilizatorul este solicitat să confirme opțiunea, inițiind astfel procesul de desenare.

4. Design

Un aspect fundamental în cadrul dezvoltării acestui proiect este componenta hardware. Astfel, a fost necesar să efectuez conexiunile între toate componentele hardware conform ilustrației prezentate în *Figura 3*. Cele două motoare pas cu pas au fost conectate la scutul de antrenare a motorului L293D. Fiecare motor are două bobine, fiecare dintre acestea dispunând de o intrare și o ieșire, motiv pentru care fiecare motor a fost conectat prin intermediul a patru fire la scut. De asemenea, servomotorul a fost conectat la același scut. Scutul L293D a fost interconectat cu placa Arduino Mega, iar pentru a asigura alimentarea adecvată a tuturor motoarelor, acesta a fost alimentat de la o sursă de 5V. Placa Arduino a fost conectată direct la un laptop prin intermediul unui cablu, permițând astfel transmiterea scriptului care controlează toate mișcările motoarelor.

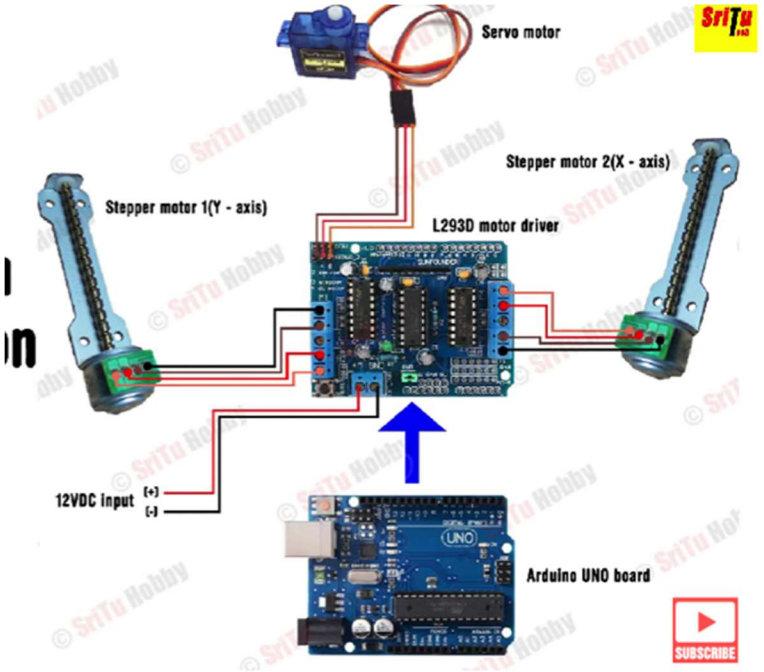

Figura 3 Schema electrica a circuitului ([1])

Conform schemei de montaj, ansamblul fizic ar trebui să se prezinte în modul următor, ca în *Figura 4*:

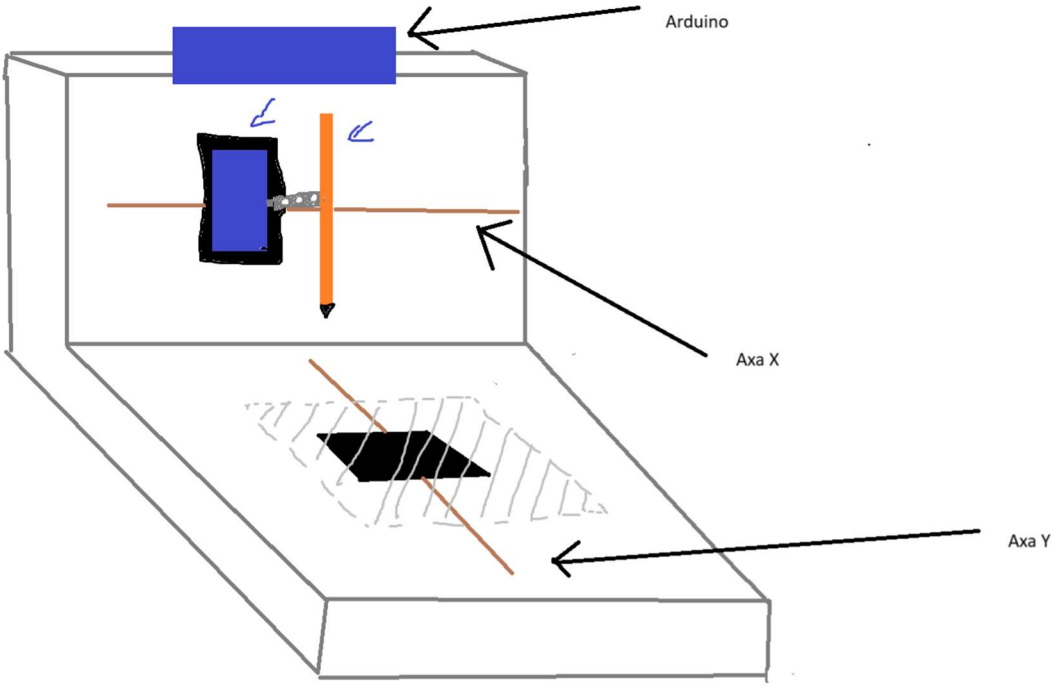

Figura 4 Ansamblul fizic al CNC-ului

Un alt element esențial al acestui proiect este reprezentat de interfața grafică, prin intermediul căreia utilizatorul poate realiza desene și genera G-code-ul corespunzător. Conform schiței prezentate în Figura 5, această interfață ar trebui să se configureze astfel:

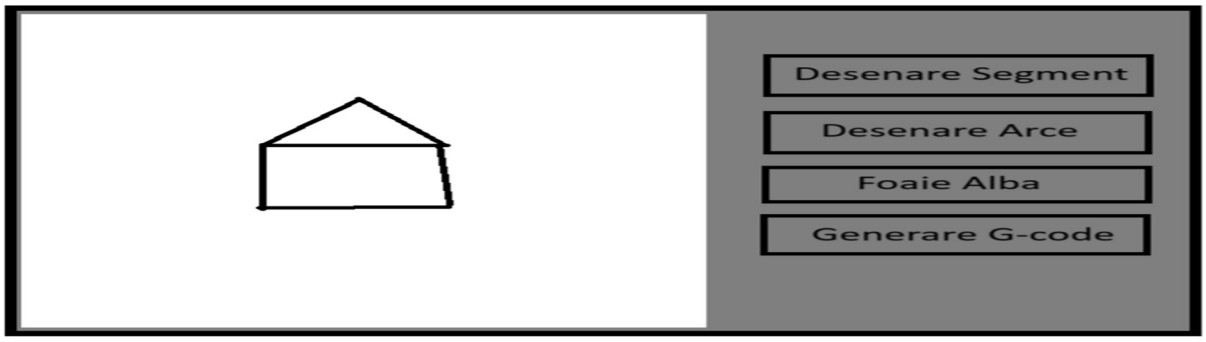

Figura 5 Interfața Grafică

5. Implementare

Conform schiței ilustrate în Figura 4, am realizat ansamblul fizic al mașinii CNC conform prezentării din Figura 6.1 și Figura 6.2. Am dezasamblat două unități CD-ROM pentru a extrage cele două motoare stepper, utilizate pentru deplasarea pe axele X și Y. Capacele metalice ale cutiilor CD-ROM au fost adaptate în calitate de suporturi. Fiecare motor a fost fixat în patru puncte cu ajutorul șuruburilor, motorul inferior responsabil pentru deplasarea pe axa Y, iar motorul superior pentru axa X. Pe suprafața fiecărui motor am aplicat o placă metalică pentru a obține o suprafață plană. Pe placa metalică asociată motorului inferior, am plasat hârtie, aceasta fiind fixată cu ajutorul unor magneți pentru a preveni orice mișcare în timpul desenării. Pe placa metalică asociată motorului superior, am montat un motor servo, care ridică și coboară pixul, având totodată atașat pixul în sine. Cele două motoare stepper și motorul servo sunt conectate la placa Arduino conform schemei electrice din Figura 3.

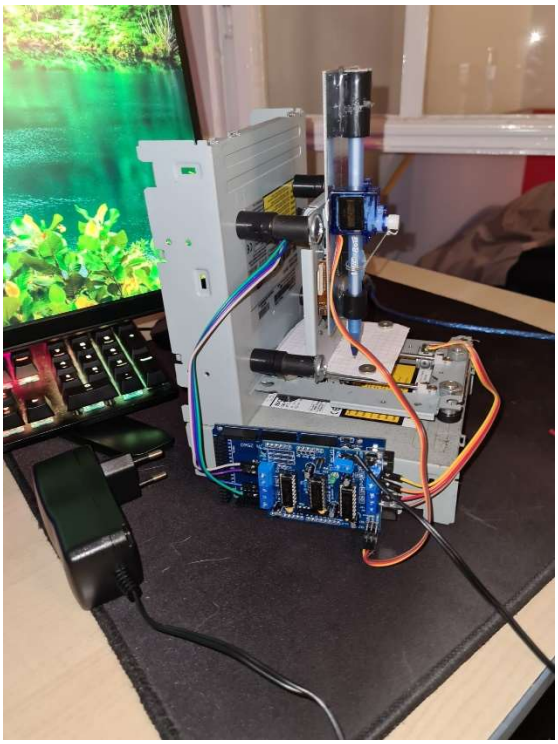

Figura 6.1 Ansamblul CNC-ului

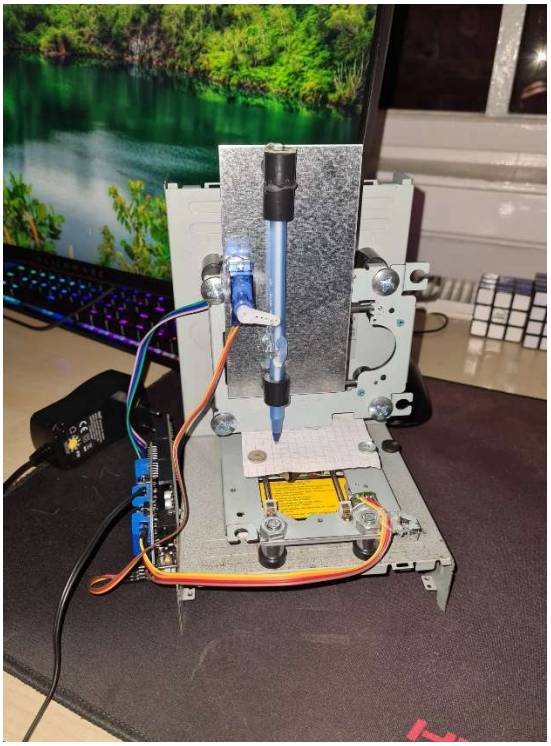

Figura 6.2 Ansamblul CNC-ului

Interfața grafică dezvoltată prin intermediul mediului de dezvoltare integrat IntelliJ se conformează *Figura 7*. Conform ilustrației, se observă o zonă albă de desenare în partea stângă, cu dimensiunile de 40x40 mm. Desenarea se realizează prin intermediul click-lui stâng al mouse-ului, fiecare click plasând un punct pe suprafața de desen. Prin apăsarea butonului "Segment", se vor trasa linii între punctele stabilite. Pentru a activa butonul "Arc", este necesar să se plaseze două puncte pe suprafața de desen, primul reprezentând un punct de pe arcul cercului, iar cel de-al doilea indicând centrul cercului. Minim un arc

de cerc trebuie să fie selectat pentru a fi desenat. Astfel, arcele de cerc vor fi desenate în funcție de opțiunile selectate prin casetele de bifare. Apăsarea butonului "Foaie Albă" va șterge întregul conținut de pe suprafața de desen la momentul respectiv, efectuând practic o resetare completă. Butonul "Generare G-code" creează un fișier G-code care conține desenul aflat pe suprafața de desenare.

Figura 7 Implementarea Interfeței Grafice

Interfața grafică dezvoltată în mediul de dezvoltare integrat Processing este implementată conform *Figura 8*. În conformitate cu prezentarea grafică, apăsarea tastei "p" permite utilizatorului să selecteze portul serial al plăcii Arduino. Manevrarea motoarelor de pe axele X și Y se realizează prin apăsarea săgeților de pe tastatură. Ridicarea sau coborârea instrumentului de scris se poate realiza prin apăsarea tastelor "5" sau "2". Prin apăsarea tastei "s", se afișează setările pentru GRBL. Tasta "h" readuce CNC-ul în poziția inițială, numită "acasă". Tasta "0" configurează poziția curentă a CNC-ului ca poziție de "acasă". Apăsarea tastei "g" deschide o interfață cu fișierele din calculator, de unde utilizatorul poate selecta fișierul dorit pentru a-l transmite CNC-ului. Tasta "x" oprește execuția CNC-ului.

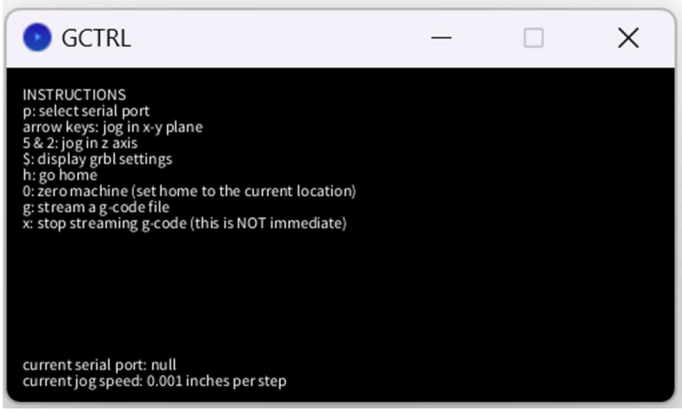

Figura 8 Interfața Grafică

Funcția „segmentButton” din *Figura 9* este utilizată în momentul în care se apasă în interfața de desenare pe butonul „Segment”. Aceasta generează doi vectori cu coordonatele X și Y a punctelor de pe plasa de desen, după care se realizează desenare pe ecran cu ajutorul funcției „drawPolyline” din clasa „Graphics”.

```
segmentButton.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        int[] xArray = new int[alX.size()];
        int[] yArray = new int[alY.size()];

        for(int i = 0; i < alX.size(); i++)
        {
            xArray[i] = alX.get(i);
            yArray[i] = alY.get(i);
        }

        Graphics graphics = drawPanel.getGraphics();
        graphics.drawPolyline(xArray, yArray, alX.size());

        ArrayList<Integer> coordsList = new ArrayList<>();
        for(int i = 0; i < alX.size(); i++)
        {
            coordsList.add(alX.get(i));
            coordsList.add(alY.get(i));
        }
        linesMap.put(line++,coordsList);

        alX.clear();
        alY.clear();
    }
});
```

Figura 9 Funcția de desenare a liniilor

Funcția „arcButton” din *Figura 10.1, Figura 10.2, Figura 10.3* și *Figura 10.4* este utilizată în momentul în care se apasă în interfața de desenare pe butonul „Arc”. Aceasta generează, în funcție de un punct de pe raza cercului si centru cercului, doi vectori cu toate coordonatele X și Y a punctelor de pe circumferința cercului respectiv, după care în funcție de căsuțele selectate, care reprezintă cadranele cercului, se desenează pe ecran arcele respective.

```java
arcButton.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        if (alX.size() >= 2 && (a1CheckBox.isSelected() || a2CheckBox.isSelected() || a3CheckBox.isSelected() || a4CheckBox.isSelected())) {
            int xPoint = alX.get(0);
            int yPoint = alY.get(0);
            int xCenter = alX.get(1);
            int yCenter = alY.get(1);
            int radius = (int) Math.sqrt((xCenter - xPoint) * (xCenter - xPoint) + (yCenter - yPoint) * (yCenter - yPoint));

            Graphics graphics = drawPanel.getGraphics();

            int[] coordsXList = new int[4 * radius];
            int[] coordsYList = new int[4 * radius];
            int lenght = 0;
            int last;

            // 1
            for(int i = 0; i < radius; i++)
            {
                coordsXList[lenght] = xCenter + radius - i;
                coordsYList[lenght++] = yCenter - (int)Math.sqrt(radius * radius - (xCenter - (xCenter + radius - i)) * (xCenter - (xCenter + radius - i)));
            }
            last = lenght - 1;

            //2
            for(int i = 0; i < radius; i++)
            {
                coordsXList[lenght] = xCenter - i;
                coordsYList[lenght++] = coordsYList[last - i];
            }
```

Figura 10.1 Funcția de desenare a arcelor de cerc

```java
            //3
            for(int i = 0; i < radius; i++)
            {
                coordsXList[lenght] = xCenter - radius + i + 1;
                coordsYList[lenght++] = yCenter + (int)Math.sqrt(radius * radius - (xCenter - (xCenter - radius + i)) * (xCenter - (xCenter - radius + i)));
            }
            last = lenght - 1;

            //4
            for(int i = 0; i < radius; i++)
            {
                coordsXList[lenght] = xCenter + i + 1;
                coordsYList[lenght++] = coordsYList[last - i];
            }

            if(a1CheckBox.isSelected()) {
                int[] cadran1X = new int[radius];
                int[] cadran1Y = new int[radius];
                for(int i = 0; i < radius; i++){
                    cadran1X[i] = coordsXList[i];
                    cadran1Y[i] = coordsYList[i];
                }
                graphics.drawPolyline(cadran1X, cadran1Y, cadran1X.length);

                ArrayList<Integer> coordsList = new ArrayList<>();
                for(int i = 0; i < cadran1X.length; i++)
                {
                    coordsList.add(cadran1X[i]);
                    coordsList.add(cadran1Y[i]);
                }
                linesMap.put(line++,coordsList);
            }
```

Figura 10.2 Funcția de desenare a arcelor de cerc

```
if(a2CheckBox.isSelected()) {
    int[] cadran2X = new int[radius];
    int[] cadran2Y = new int[radius];
    for(int i = 0; i < radius; i++){
        cadran2X[i] = coordsXList[radius + i];
        cadran2Y[i] = coordsYList[radius + i];
    }
    graphics.drawPolyline(cadran2X, cadran2Y, cadran2X.length);

    ArrayList<Integer> coordsList = new ArrayList<>();
    for(int i = 0; i < cadran2X.length; i++)
    {
        coordsList.add(cadran2X[i]);
        coordsList.add(cadran2Y[i]);
    }
    linesMap.put(line++,coordsList);
}

if(a3CheckBox.isSelected()) {
    int[] cadran3X = new int[radius];
    int[] cadran3Y = new int[radius];
    for(int i = 0; i < radius; i++){
        cadran3X[i] = coordsXList[2 * radius + i];
        cadran3Y[i] = coordsYList[2 * radius + i];
    }
    graphics.drawPolyline(cadran3X, cadran3Y, cadran3X.length);

    ArrayList<Integer> coordsList = new ArrayList<>();
    for(int i = 0; i < cadran3X.length; i++)
    {
        coordsList.add(cadran3X[i]);
        coordsList.add(cadran3Y[i]);
    }
    linesMap.put(line++,coordsList);
}
```

Figura 10.3 Funcția de desenare a arcelor de cerc

```
        if(a4CheckBox.isSelected()) {
            int[] cadran4X = new int[radius];
            int[] cadran4Y = new int[radius];
            for(int i = 0; i < radius; i++){
                cadran4X[i] = coordsXList[3 * radius + i];
                cadran4Y[i] = coordsYList[3 * radius + i];
            }
            graphics.drawPolyline(cadran4X, cadran4Y, cadran4X.length);

            ArrayList<Integer> coordsList = new ArrayList<>();
            for(int i = 0; i < cadran4X.length; i++)
            {
                coordsList.add(cadran4X[i]);
                coordsList.add(cadran4Y[i]);
            }
            linesMap.put(line++,coordsList);
        }

        alX.clear();
        alY.clear();
    }else{
        JOptionPane.showMessageDialog(frame,  message: "Trebuie marcate doua puncte si sa fie selectat minim un cadran!",  title: "Argumente insuficiente pentru desenare
    }
}
```

Figura 10.4 Funcția de desenare a arcelor de cerc

Funcția „generareGCodeButton” din *Figura 11.1* și *Figura 11.2* este utilizată în momentul în care se apasă în interfața de desenare pe butonul „Generare G-code”. Aceasta creează un fișier „.txt” cu numele format din data si ora curenta. După care se scrie în fișier linii de code de tipul G-code.

```
generareGCodeButton.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        FileWriter f;
        PrintWriter p;
        DateTimeFormatter dtf = DateTimeFormatter.ofPattern("dd-MM-yyyy_HH-mm-ss");
        LocalDateTime now = LocalDateTime.now();

        String nameFile = "G" + dtf.format(now) + ".txt";
        try{
            f = new FileWriter(nameFile, append: true);
            p = new PrintWriter(f);

            p.write( s: "G21 (metric ftw)\n");
            p.write( s: "G90 (absolute mode)\n");
            p.write( s: "G92 X0.00 Y0.00 Z0.00\n\n");

            p.write( s: "M300 S30 (pen down)\n");
            p.write( s: "G4 P150 (wait 150ms)\n");
            p.write( s: "M300 S50 (pen up)\n");
            p.write( s: "G4 P150 (wait 150ms)\n");
            p.write( s: "M18 (disengage drives)\n");
            p.write( s: "M01 (Was registration test successful?)\n");
            p.write( s: "M17 (engage drives if YES, and continue)\n\n");

            for(int i = 0; i < line; i++){
                p.write( s: "(Polyline consisting of 1 segments.)\n");
                p.write( s: "G1 X" + (double)(linesMap.get(i).get(0))/10.0 + " Y" + (double)(400 - linesMap.get(i).get(1))/10.0 + " F3500.00\n");
                p.write( s: "M300 S30.00 (pen down)\n");
                p.write( s: "G4 P150 (wait 150ms)\n");
                for(int j = 2; j < linesMap.get(i).size(); j += 2)
                {
                    p.write( s: "G1 X" + (double)(linesMap.get(i).get(j))/10.0 + " Y" + (double)(400 - linesMap.get(i).get(j+1))/10.0 + " F3500.00\n");
                }
```

Figura 11.1 Funcția de generare a G-code-ului

```
            p.write( s: "M300 S50.00 (pen up)\n");
            p.write( s: "G4 P150 (wait 150ms)\n\n");
        }

        p.write( s: "\n\n(end of print job)\n");
        p.write( s: "M300 S50.00 (pen up)\n");
        p.write( s: "G4 P150 (wait 150ms)\n");
        p.write( s: "M300 S255 (turn off servo)\n");
        p.write( s: "G1 X0 Y0 F3500.00\n");
        p.write( s: "G1 Z0.00 F150.00 (go up to finished level)\n");
        p.write( s: "G1 X0.00 Y0.00 F3500.00 (go home)\n");
        p.write( s: "M18 (drives off)\n");

        p.close();
        f.close();
    }catch (IOException ioException) {
        ioException.printStackTrace();
    }
}
```

Figura 11.2 Funcția de generare a G-code-ului

Funcțiile „penUp” și „penDown” din *Figura 12* este utilizată pentru a ridica pixul de pe foaie în momentul când citește din fișier comanda „M300 S50.00”, respectiv să pună pixul pe foaie în momentul când citește din fișier comanda „M300 S30.00”.

```
void penUp() {
  penServo.write(penZUp);
  delay(penDelay);
  Zpos=Zmax;
  digitalWrite(15, LOW);
    digitalWrite(16, HIGH);
  if (verbose) {
    Serial.println("Pen up!");


  }
}
//  Lowers pen
void penDown() {
  penServo.write(penZDown);
  delay(penDelay);
  Zpos=Zmin;
  digitalWrite(15, HIGH);
    digitalWrite(16, LOW);
  if (verbose) {
    Serial.println("Pen down.");


  }
}
```

Figura 12 Funcțiile de ridicare si coborâre a pixului

Funcțiile „drawLine” din *Figura 13* este utilizată pentru deplasa pixul din poziția curenta (x0, y0) la noua poziție (x1, y1) deplasând motoare stepper pe axele X si Y.

```
void drawLine(float x1, float y1) {

  if (verbose)
  {
    Serial.print("fx1, fy1: ");
    Serial.print(x1);
    Serial.print(",");
    Serial.print(y1);
    Serial.println("");
  }

  //  Bring instructions within limits
  if (x1 >= Xmax) {
    x1 = Xmax;
  }
  if (x1 <= Xmin) {
    x1 = Xmin;
  }
  if (y1 >= Ymax) {
    y1 = Ymax;
  }
  if (y1 <= Ymin) {
    y1 = Ymin;
  }

  if (verbose)
  {
    Serial.print("Xpos, Ypos: ");
    Serial.print(Xpos);
    Serial.print(",");
    Serial.print(Ypos);
    Serial.println("");
  }
```

```
  if (verbose)
  {
    Serial.print("x1, y1: ");
    Serial.print(x1);
    Serial.print(",");
    Serial.print(y1);
    Serial.println("");
  }

  //  Convert coordinates to steps
  x1 = (int)(x1*StepsPerMillimeterX);
  y1 = (int)(y1*StepsPerMillimeterY);
  float x0 = Xpos;
  float y0 = Ypos;

  //  Let's find out the change for the coordinates
  long dx = abs(x1-x0);
  long dy = abs(y1-y0);
  int sx = x0<x1 ? StepInc : -StepInc;
  int sy = y0<y1 ? StepInc : -StepInc;

  long i;
  long over = 0;

  if (dx > dy) {
    for (i=0; i<dx; ++i) {
      myStepperX.onestep(sx,STEP);
      over+=dy;
      if (over>=dx) {
        over-=dx;
        myStepperY.onestep(sy,STEP);
      }
    delay(StepDelay);
    }
  }
```

```
  else {
    for (i=0; i<dy; ++i) {
      myStepperY.onestep(sy,STEP);
      over+=dx;
      if (over>=dy) {
        over-=dy;
        myStepperX.onestep(sx,STEP);
      }
      delay(StepDelay);
    }
  }

  if (verbose)
  {
    Serial.print("dx, dy:");
    Serial.print(dx);
    Serial.print(",");
    Serial.print(dy);
    Serial.println("");
  }

  if (verbose)
  {
    Serial.print("Going to (");
    Serial.print(x0);
    Serial.print(",");
    Serial.print(y0);
    Serial.println(")");
  }

  //  Delay before any next lines are submitted
  delay(LineDelay);
  //  Update the positions
  Xpos = x1;
  Ypos = y1;
}
```

Figura 13 Funcția de mișcare a mortarelor intre doua puncte

6. Testare si validare

Mai departe voi arăta câteva exemple de testare:

In *Figura 14* se poate observa cum s-au desenat punctele pe planșa.

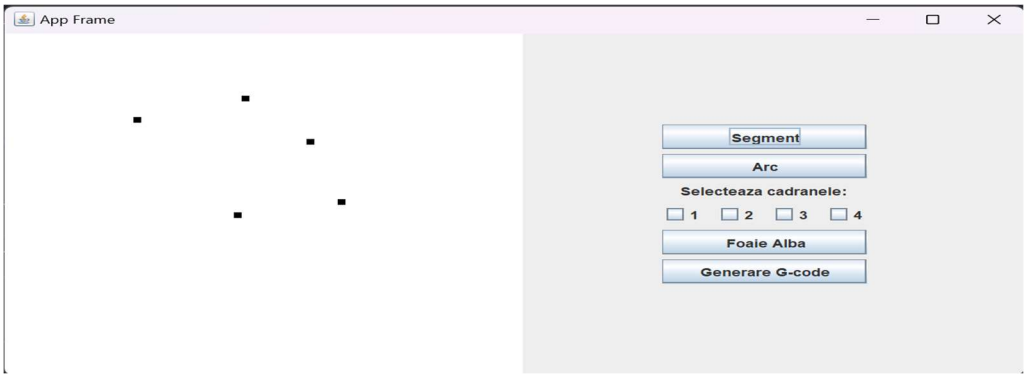

Figura 14 Puncte desenate

In *Figura 15* se poate observa cum s-au desenat segmentele intre acele puncte in momentul apăsării pe butonul „Segment”.

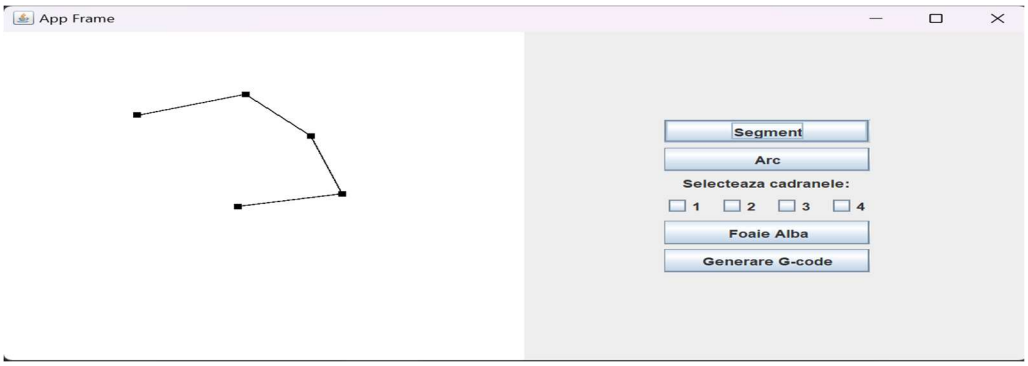

Figura 15 Segmente desenate

In *Figura 16* se poate observa cum s-au desenat arcele selectate in dreapta in urma apăsării butonului „Arc”.

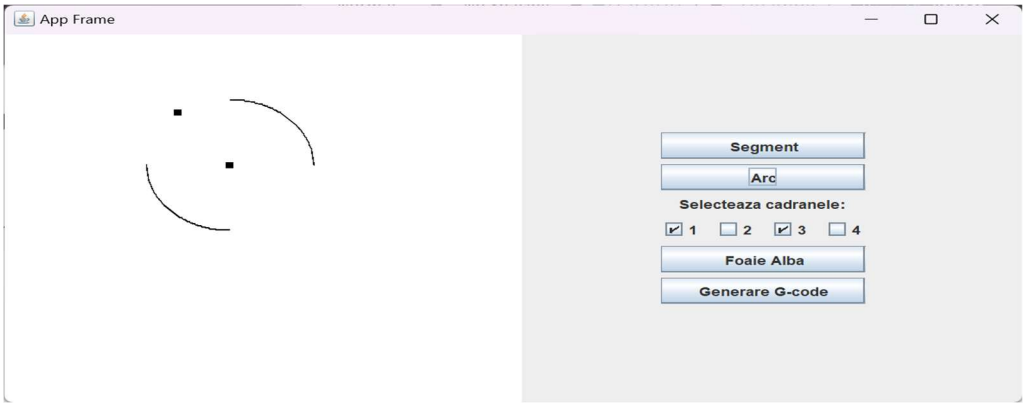

Figura 16 Arce desenate

In *Figura 17* se poate observa mesajul de eroare pentru desenarea arcelor.

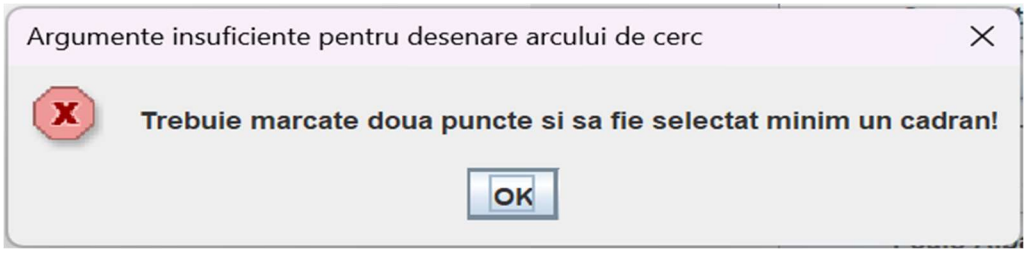

Figura 17 Mesaj de eroare

In *Figura 18* se poate observa cum s-a șters desenul curent de pe planșa de desenare in urma apăsării pe butonul „Foaie Alba”.

Figura 18 Golire planșa

In *Figura 19* se poate observa cum s-a generat un fișierul G-code in urma apăsării pe butonul „Generare G-code”.

Figura 19 Fișier G-code

In Figura 20 este se poate observa cum CNC-ul a reușit sa deseneze un desen dat de mine.

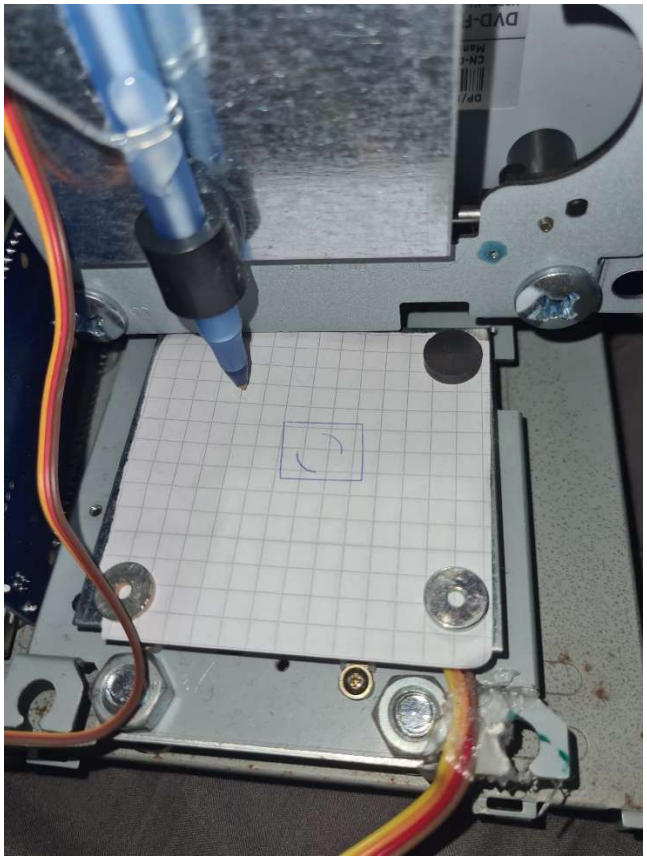

Figura 20 Funcționalitatea CNC-ului

7. Concluzii

Scopul fundamental al acestui proiect constă în dezvoltarea și implementarea unui program dedicat unei mașini de tăiat cu flacără. Acest program are ca obiectiv generarea unui cod G, ce va fi interpretat de o plăcuță Arduino Mega, pentru a direcționa desenarea unor secvențe compuse din segmente și arce de cerc pe o suprafață de hârtie sau material similar. Aceste secvențe sunt configurate prin intermediul unei interfețe cu utilizatorul (UI) care permite utilizatorului să le definească cu ajutorul mouse-ului.

Aspectul ce m-a motivat profund în acest proiect a fost posibilitatea de a concepe și construi o mașină fizică ce simulează funcționalitatea unei mașini de tăiat cu flacără, utilizând în acest proces plăcuța Arduino Mega. Deosebit de captivant a fost faptul că am avut ocazia să lucrez cu această plăcuță Arduino pentru prima dată.

8. Bibliografie

[1] SriTu Hobby, “How to make a DIY Arduino CNC drawing machine” [Online]. Disponibil: https://srituhobby.com/how-to-make-a-diy-arduino-cnc-drawing-machine-at-home/

[2] Dejan, “G-code Explained” [Online]. Disponibil: https://howtomechatronics.com/tutorials/g-code-explained-list-of-most-important-g-code-commands/

[3] WHIMSICAL VIBES, „Java- Draw Polylines with Mouse Clicks” [Online]. Disponibil: https://www.youtube.com/watch?v=OQJpRPMH_44

Tabelul figurilor