Student: Pupaza Danut-Catalin
Grupa: 342B2
Data: 09.05.2022

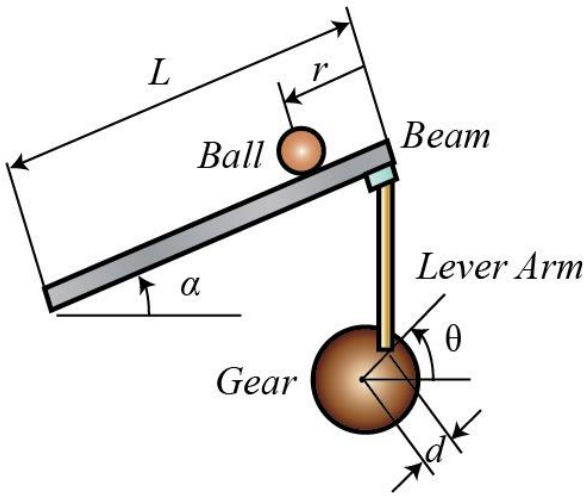
Proiect
~Sisteme Integrate de Conducere~
~Instalatia ball and beam~

- **Cuprins:**

1. Introducere – prezentarea problemei

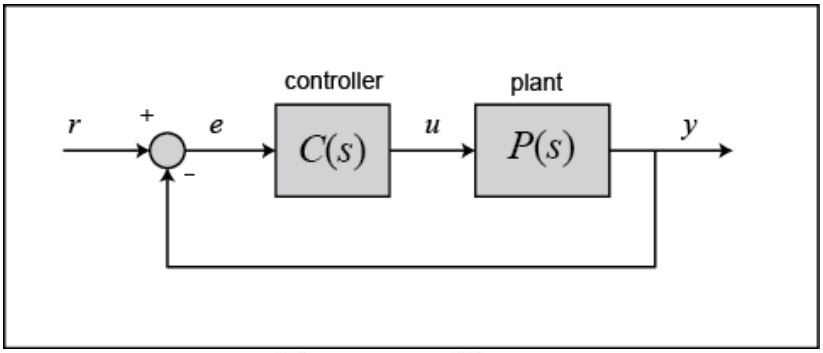
- **Instalatia Ball and Beam**

O minge/roata este plasata pe o bara de suport, vezi figura de mai jos, unde este lasata sa se rostogoleasca cu 1 grad de libertate de-a lungul lungimii barii de suport.
Un brat de parghie este atașat la bara de suport la un capat și la un servomotor la celalalt capat. Pe masura ce angrenajul servo se rotește cu un unghi theta, parghia schimba unghiul dintre orzontala si bara de suport(alpha). Cand unghiul este schimbat din poziția orizontala, gravitația face ca mingea/roata sa se rostogoleasca de-a lungul fasciculului.



2. Definirea solutiei

- **Pentru acest sistem va fi proiectat un controler PID astfel incat poziția mingii sa poata fi manipulata.**



$$u(t) = K_p e(t) + K_i \int e(t)dt + K_p \frac{de}{dt}$$

- **Pasii urmati in alegerea constantelor(experimental)**
  a) Am ales Kp astfel incat roata sa se duca dintr-un capat in altul al barii de support, dar nu agresiv
  b) Am ales Kd astfel incat sistemul sa compenseze viteza rotii si astfel sa se opreasca pe loc, dar fara sa creeze o instabilitate constanta
  c) Am ales Ki astfel incat sistemul sa elimine micile erori finale, dar fara sa creeze o instabilitate constanta
  d) Am reajustat experimental cele trei constante dupa ce am compus controllerul PID cu cele trei valori alese la pasii precedenti

3. Componente utilizate

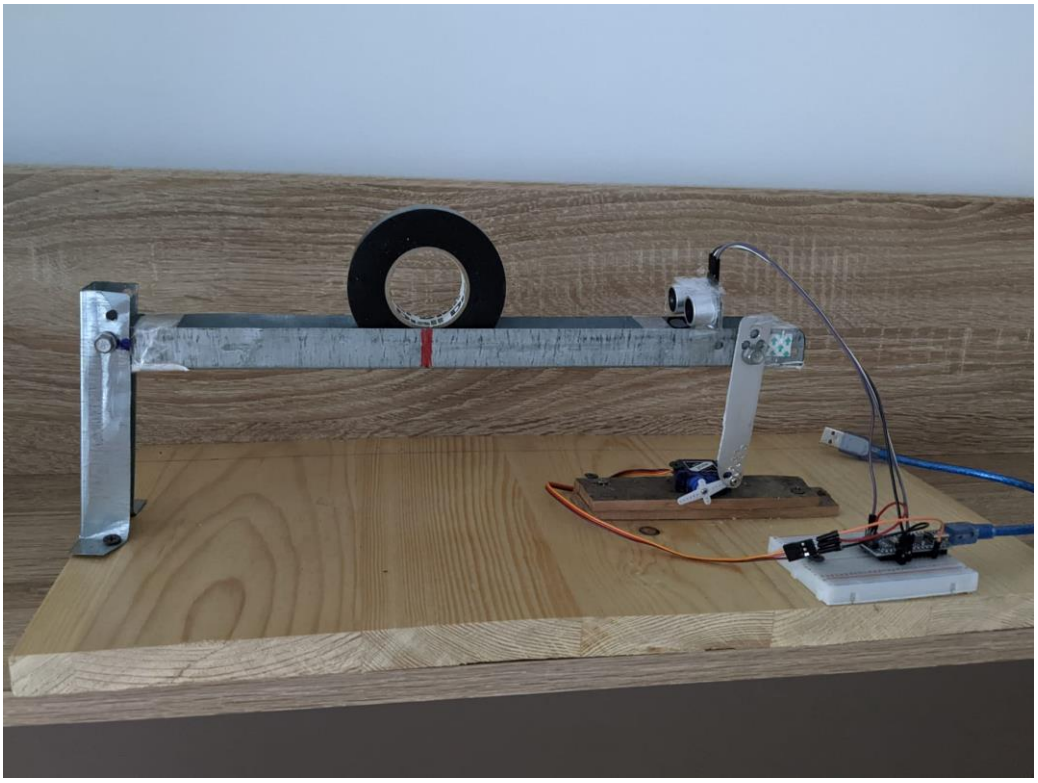
- Arduino Nano

- Servomotor SG90

- Senzor ultrasonic HC-SR04

- Fire de conectare
- Foi si profile de metal
- Rola de banda izolera
- Suport de lemn

4. **Prezentarea implementarii hardware**



5. **Prezentarea implementarii software**

- **Cod Arduino**

```
#include <Wire.h>
#include <Servo.h>

//////////////////////Inputs/outputs//////////////////////
```

```
const int trigPinSensor = 2;
const int echoPinSensor = 3;

Servo myservo;  // create servo object to control a servo, later attatched to D9
//////////////////////////////////////////////////


//////////////////////Variables//////////////////////
int Read = 0;
float distance = 0.0;
float elapsedTime, time, timePrev;       //Variables for time control
float distance_previous_error, distance_error;
int period = 50;  //Refresh rate period of the loop is 50ms
//////////////////////////////////////////////////


//////////////////PID constants//////////////////////
float kp=20; //--------------------------------------------------------------Mine was 8
float ki=0.2; //-------------------------------------------------------------Mine was 0.2
float kd=3000; //------------------------------------------------------------Mine was 3100
float distance_setpoint = 10;          //Should be the distance from sensor to the middle of the
bar in cm
float PID_p, PID_i, PID_d, PID_total;
//////////////////////////////////////////////////



void setup() {
  //analogReference(EXTERNAL);
  Serial.begin(9600);
  myservo.attach(9);  // attaches the servo on pin 9 to the servo object
  myservo.write(125); //Put the servo at angle 125, in balance

  pinMode(trigPinSensor, OUTPUT);
  pinMode(echoPinSensor, INPUT);

  time = millis();
}
//////////////////////////////////////////////////


void loop() {
  if (millis() > time+period)
  {
    time = millis();
    distance = get_dist();
    distance_error = distance_setpoint - distance;
    PID_p = kp * distance_error;
```

```
   float dist_diference = distance_error - distance_previous_error;
   PID_d = kd*((distance_error - distance_previous_error)/period);

   if(-5 < distance_error && distance_error < 5)
   {
    PID_i = PID_i + (ki * distance_error);
   }
   else
   {
    PID_i = 0;
   }

   PID_total = PID_p + PID_i + PID_d;
   PID_total = map(PID_total, -150, 150, 0, 150);

   //if(PID_total < 20){PID_total = 20;}
   //if(PID_total > 150) {PID_total = 150; }
   PID_total = PID_total+30;

   myservo.write(PID_total);
   distance_previous_error = distance_error;
  }
}




float get_dist()
{
  float durationSensor, distanceSensor;

  digitalWrite(trigPinSensor, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPinSensor, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPinSensor, LOW);

  durationSensor = pulseIn(echoPinSensor, HIGH);
  distanceSensor = (durationSensor*.0343)/2;
  Serial.print("DistanceSensor: ");
  if(distanceSensor >34){ distanceSensor = 34;}

  Serial.println(distanceSensor);
  return(distanceSensor);
}
```

- **Cod Processing**

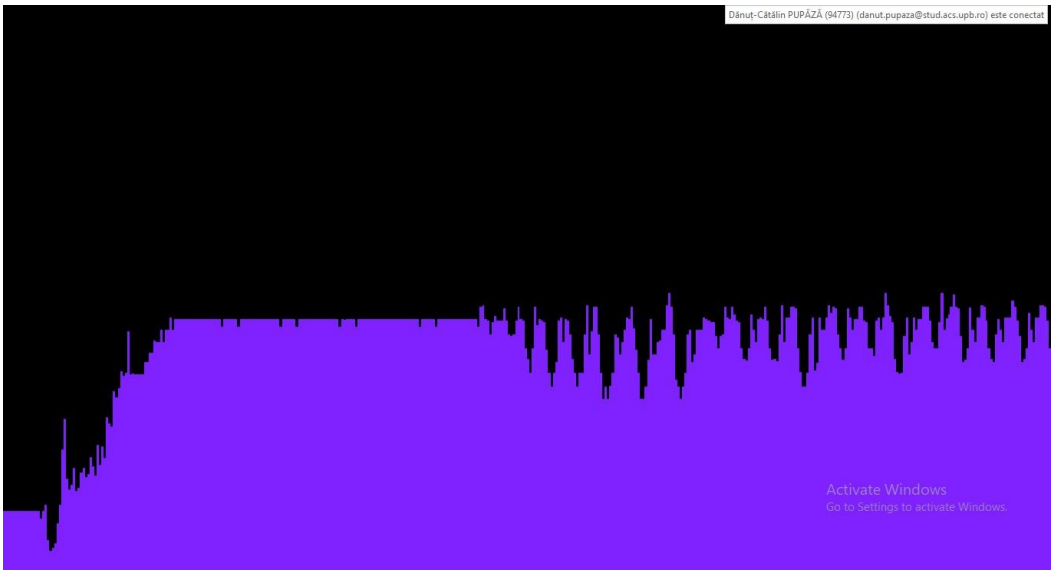
```
import processing.serial.*;

 Serial myPort;       // The serial port
 int xPos = 1;        // horizontal position of the graph
 float inByte = 0;

 void setup () {
  // set the window size:
  size(1400, 800);

  // List all the available serial ports
  // if using Processing 2.1 or later, use Serial.printArray()
  println(Serial.list());

  // I know that the first port in the serial list on my Mac is always my
  // Arduino, so I open Serial.list()[0].
  // Open whatever port is the one you're using.
  myPort = new Serial(this, Serial.list()[0], 9600);

  // don't generate a serialEvent() unless you get a newline character:
  myPort.bufferUntil('\n');

  // set initial background:
  background(0);
 }

 void draw () {
  // draw the line:
  stroke(127, 34, 255);
  line(xPos,1000* height, xPos,   height - inByte);

  // at the edge of the screen, go back to the beginning:
  if (xPos >= width) {
    xPos = 0;
    background(0);
  } else {
    // increment the horizontal position:
    xPos++;
  }
 }

 void serialEvent (Serial myPort) {
  // get the ASCII string:
  String inString = myPort.readStringUntil('\n');

  if (inString != null) {
    // trim off any whitespace:
```

```
    inString = trim(inString);
    // convert to an int and map to the screen height:
    inByte = float(inString);
    println(inByte);
    inByte = map(inByte, 0, 1023, 0, height);
  }
}
```

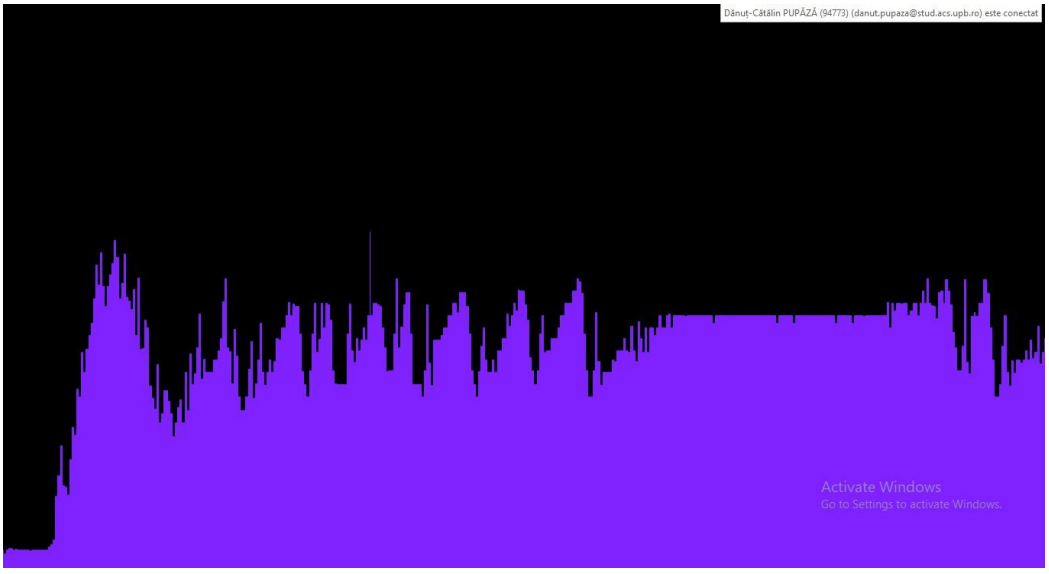
6. **Rezultate la testarea aplicatiei**

- **Link pentru vizuarizarea stabilizarii facute de controllerul implementat - https://www.youtube.com/watch?v=yQe52y5OX00**

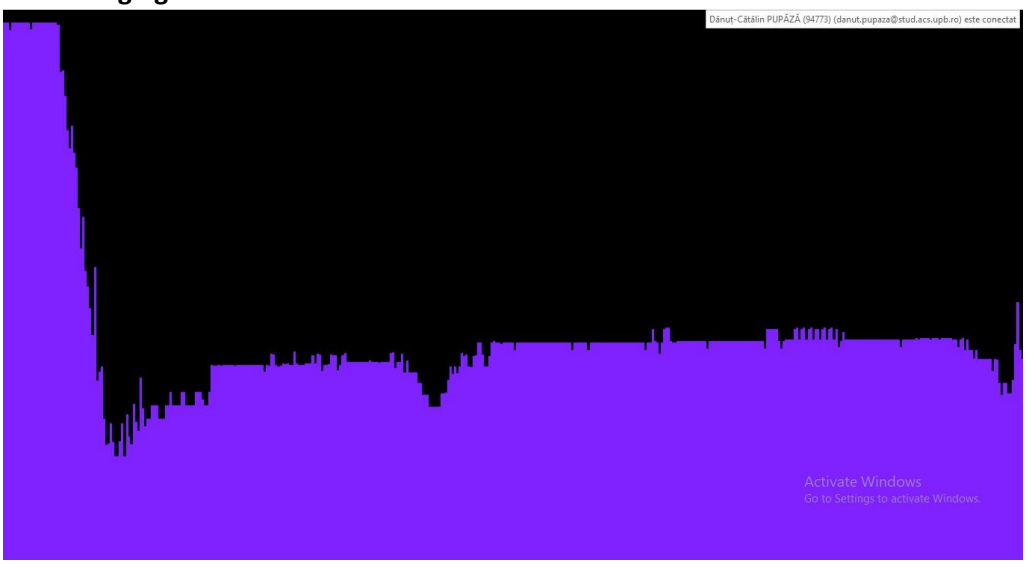
- **Processing - grafic iesire - test 1**



- **Processing - grafic iesire - test 2**



- **Processing - grafic iesire - test 3**



7. **Bibliografie**

1. https://ctms.engin.umich.edu/
2. https://www.youtube.com/watch?v=JFTJ2SS4xyA&t=531s
3. https://www.youtube.com/watch?v=FidxDZ7X6OI&t=84s
4. https://www.youtube.com/watch?v=5fG3ongbo9o&t=10s
5. https://www.youtube.com/watch?v=YOPTksabdbM
6. https://www.youtube.com/watch?v=8Yx-G-wGNHw