



**Universidade Federal de Uberlândia-UFU**  
**Faculdade de Engenharia Mecânica**  
**Graduação em Engenharia Mecatrônica**  
**Sistemas Digitais para Mecatrônica**



## **Trabalho Prático 1**

Carlos Eduardo Cardia Fernandez  
Davi da Silva Estrela  
Laura Bueno Ferreira  
Rafael de Lima Costa  
Talles Martins de Carvalho

11911EMT016  
11911EMT005  
11911EMT017  
11611EMT011  
11911EMT014

Uberlândia-MG  
Junho de 2023

## Introdução

O trabalho em questão consiste na construção de um sistema de controle embarcado, de forma mais precisa um pêndulo invertido por roda de reação. O que une os conhecimentos anteriores obtidos num curso de graduação de engenharia com os conhecimentos da disciplina em questão. Para a realização deste foi necessário a construção física do projeto e para seu controle foi desenvolvido um código em MatLab para seu funcionamento.

## Códigos computacionais

A seguir estão presentes os códigos escritos para o controle do projeto:

```
#include <sTune.h> // faz a sintonia
#include <PID_v1.h> //executa o PID

uint32_t settleTimeSec = 10;
uint32_t testTimeSec = 500;
const uint16_t samples = 500;
const float inputSpan = 200;
const float outputSpan = 1000;
float outputStart = 0;
float outputStep = 50;
float tempLimit = 150;
uint8_t debounce = 1;

// variables
double input, output, setpoint = 0, setpoint1, kp, ki, kd; // PID_v1
float Input, Output, Setpoint = 0, Kp, Ki, Kd; // sTune

sTune tuner = sTune(&Input, &Output, tuner.ZN_PID, tuner.directIP, tuner.printOFF);
PID myPID(&input, &output, &setpoint, kp, ki, kd, P_ON_M, DIRECT);
```

```

//motor

#define pinPwmMotor 9 // pwm

//float output = 0.0;

#define m1 4

#define m2 5


int anguloPOT() {
    int pot = 0;
    int angle = 0;
    pot = analogRead(A0);
    angle = int(map(pot, 0, 1023, -130, 130)-21);
    //Serial.println(angle);
    return angle;
}


//angulo do eixo do motor
float angulo = 0.0, erro = 0.0;


//setup
void setup() {
    //definindo o modo de funcionamento dos pinos
    pinMode(pinPwmMotor, OUTPUT);
    pinMode(m1, OUTPUT);
    pinMode(m2, OUTPUT);
    //comunicação serial
    Serial.begin(57600);

    tuner.Configure(inputSpan, outputSpan, outputStart, outputStep, testTimeSec,
        settleTimeSec, samples);

    tuner.SetEmergencyStop(tempLimit);
}

```

```

void loop() {
    //leitura do angulo pela serial

    angulo=anguloPOT();

    switch (tuner.Run()) {
        case tuner.sample:
            Input = angulo;
            break;

        case tuner.tunings:
            tuner.GetAutoTunings(&Kp, &Ki, &Kd);
            myPID.SetOutputLimits(0, outputSpan * 0.1);
            myPID.SetSampleTime(outputSpan - 1);
            debounce = 0;
            setpoint = Setpoint, output = outputStep, kp = Kp, ki = Ki, kd = Kd;
            Output = outputStep;
            myPID.SetMode(AUTOMATIC);
            myPID.SetTunings(kp, ki, kd);
            break;

        case tuner.runPid:
            Input = angulo;
            input = Input;
            myPID.Compute();
            Output = output;

            break;
    }
}

```

```
//faixa adequada de funcionamento para o PWM
```

```
Serial.print(" U: ");
```

```
Serial.print(Output);
```

```
if(Output>255){
```

```
    Output = 255;
```

```
} else if (Output<70){
```

```
    Output = 70;
```

```
}
```

```
Serial.print(" Setpoint: ");
```

```
Serial.print(Setpoint);
```

```
Serial.print(" Angulo: ");
```

```
Serial.println(angulo);
```

```
erro = Setpoint - angulo; // calcula o erro
```

```
if(abs(erro)<=2){ // estabelece que o algoritmo funcionará dentro de uma margem de erro  
igual duas vezes a resolução e define o sentido de giro
```

```
    analogWrite(pinPwmMotor,255);
```

```
    digitalWrite(m1, HIGH);
```

```
    digitalWrite(m2, HIGH);
```

```
}else if(angulo > Setpoint){
```

```
    analogWrite(pinPwmMotor,Output);
```

```
    digitalWrite(m1, LOW);
```

```
    digitalWrite(m2, HIGH);
```

```
}else if ( angulo < Setpoint){
```

```
    analogWrite(pinPwmMotor,Output);
```

```
digitalWrite(m1, HIGH);  
digitalWrite(m2, LOW);  
}  
}
```

## Construção física

A seguir estão presentes imagens da construção física do projeto. Ademais, no GitHub está disponível a modelagem em CAD das peças impressas de forma 3D.

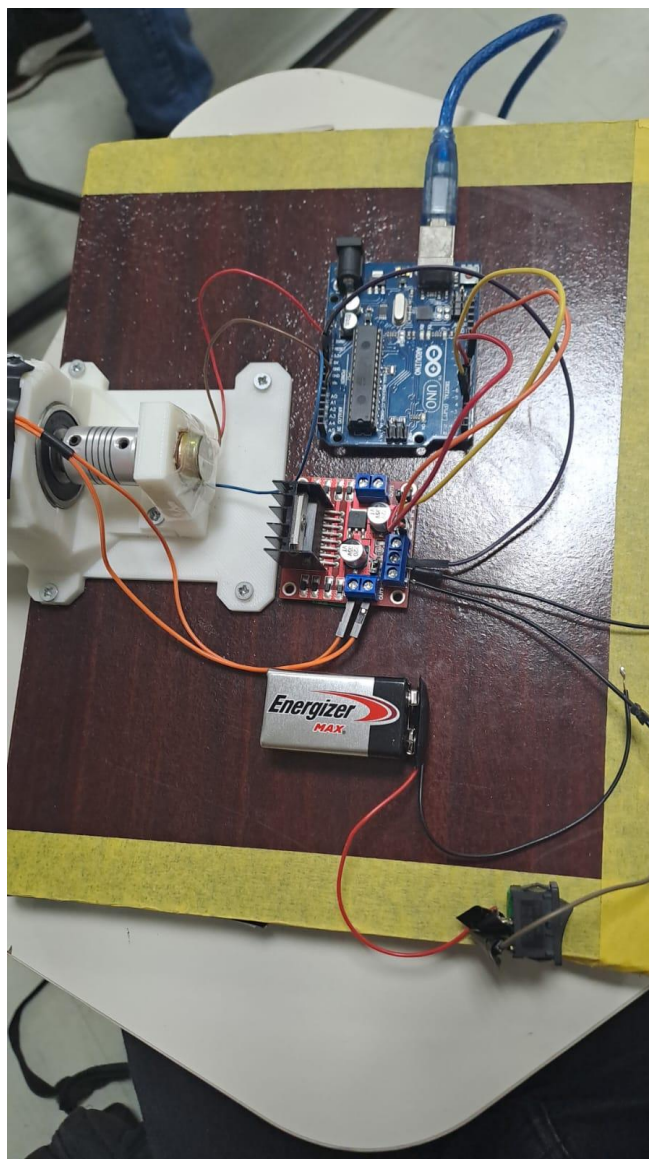


Figura 1: Componentes do pêndulo invertido

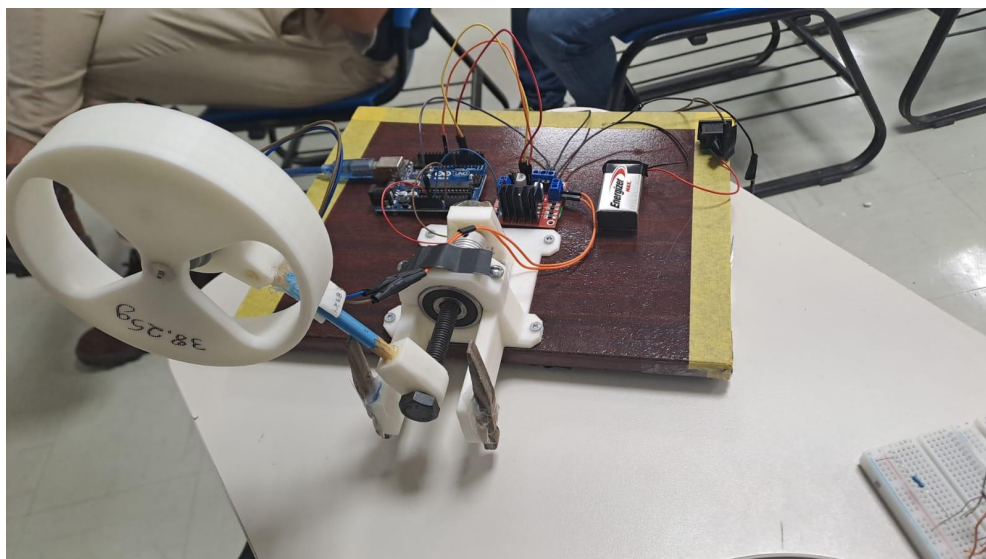


Figura 2: Montagem completa do pêndulo invertido