Laberinto De Pelotas

Integrantes: Alma Gutierrez, Santiago Pont, Gustavo
Profesor: Gonzalo Consorti

.

Carpeta De Campo

Proyecto: Laberinto de Pelotas Controlado con Arduino

Integrantes: Alma Gutierrez, Santiago Pont, Gustavo

Profesor: Gonzalo Consorti

Descripción del proyecto: Este proyecto tiene como objetivo desarrollar un sistema interactivo donde un laberinto es controlado mediante dos joysticks analógicos en los ejes X e Y. Usando un Arduino como controlador, los joysticks ajustan la inclinación del laberinto en tiempo real para guiar una pelota hasta una meta.

Introducción

El laberinto de pelotas es un proyecto que combina diseño mecánico, electrónica y programación. Este sistema interactivo busca ser educativo y entretenido, proporcionando un ejemplo práctico del control de sistemas mediante entradas analógicas y servomotores controlados por Arduino.

Investigación Inicial

Antes de iniciar el proyecto, se investigaron sistemas de control de movimiento con Arduino y proyectos similares. La investigación abarcó el uso de servomotores, joysticks analógicos y la integración de estos componentes en un diseño funcional.

Materiales Utilizados

- 1 Arduino Uno R3
- 2 Servomotores MG995: Permiten ajustar las inclinaciones del laberinto.
- 2 Joysticks analógicos: Capturan las entradas del usuario para controlar el sistema.
- Base de laberinto: Diseñada e impresa en 3D.

Diseño del Sistema

El sistema consta de tres módulos principales: 1. La base del laberinto, diseñada para ser inclinada mediante los servomotores.
2. El sistema de entrada, compuesto por joysticks que envían señales analógicas al Arduino.
3. La electrónica de control, donde el Arduino interpreta las señales y ajusta la inclinación del laberinto.

Imagen 1: Codigo

```
#include <Servo.h>

Servo Xservo;
Servo Yservo;

int VRXpin=A0;
int VRYpin=A1;

int XServoPin=9;
int YServoPin=10;

int WVx;
int WVy;

int Xval;
int Yval;

int dt=100;

int Xrange=10;
int Yrange=20;

int adjustX=0;
int adjustY=0;
```

`#include <Servo.h>`: Importa la librería para controlar servos.

`Servo Xservo` y `Servo Yservo`: Crea dos objetos servo para controlar los motores X y Y.

`VRXpin` y `VRYpin`: Definen los pines analógicos A0 y A1 como entradas del joystick.

`XServoPin` y `YServoPin`: Pines digitales 9 y 10 donde se conectan los servos.

`WVx` y `WVy`: Almacenan los valores ajustados que se enviarán a los servos.

`Xval` y `Yval`: Capturan las lecturas del joystick.

`dt`: Retardo en milisegundos entre cada iteración del `loop`.

`Xrange` y `Yrange`: Rango de movimiento de los servos en grados.

`adjustX` y `adjustY`: Compensaciones para ajustar la posición base de los servos.

Imagen 2: codigo

```
void setup() {

pinMode(VRXpin, INPUT);
pinMode(VRYpin, INPUT);

pinMode(XServoPin, OUTPUT);
pinMode(YServoPin, OUTPUT);

Xservo.attach(XServoPin);
Yservo.attach(YServoPin);
}
```

`pinMode(VRXpin, INPUT)` y `pinMode(VRYpin, INPUT)`: Configuran los pines del joystick como entradas.

`pinMode(XServoPin, OUTPUT)` y `pinMode(YServoPin, OUTPUT)`: Configuran los pines de los servos como salidas.

`Xservo.attach(XServoPin)` y `Yservo.attach(YServoPin)`: Asocian los objetos servo a los pines especificados.

Imagen 3:

Descripción:cables

```
void loop() {

Xval=analogRead(VRYpin);
WVx=(Xrange/1023.)*Xval*(-1)+Xrange+adjustX;
Yval=analogRead(VRXpin);
WVy=(Yrange/1023.)*Yval+adjustY;

Xservo.write(WVx);
Yservo.write(WVy);

delay(dt);
}
```

```
Xval=analogRead(VRYpin);
Yval=analogRead(VRXpin);
```

Lee valores analógicos del joystick (0 a 1023) de los pines A0 y A1.

```
WVx=(Xrange/1023.)*Xval*(-1)+Xrange+adjustX;
WVy=(Yrange/1023.)*Yval+adjustY;
```

`WVx` (eje X):

- Convierte la lectura del joystick (0 a 1023) al rango definido por `Xrange`.
- Multiplica por `-1` para invertir la dirección.
- Suma `adjustX` para aplicar un desplazamiento.

`WVy` (eje Y):

- Similar a `WVx`, pero sin invertir la dirección.

```
Xservo.write(WVx);
Yservo.write(WVy);
```

Envía los valores ajustados (`WVx`, `WVy`) a los servos, moviéndolos a las posiciones correspondientes.

Imagen 4:

Descripción:Cables conectados

```
#include <Servo.h>
#define x A2
#define y A3


Servo servoBasex;//Asigno un nombre específico
Servo servoBaseY;
void setup() {
   servoBasex.attach(A0);//Pin a utilizar para servo
   servoBasex.write(0);  //asigno 0 al servo motor
   servoBaseY.attach(A1);//Pin a utilizar para servo
   servoBaseY.write(0);  //asigno 0 al servo motor
   pinMode(y,INPUT);
   pinMode(x,INPUT);
   Serial.begin(9600);
}

void loop() {
    int valor = analogRead( A2 );
  int grados = map(valor,  0, 1023, 0, 255);
  servoBasex.write(grados);

int valory = analogRead( A3 );
  int gradosy = map(valory,  0, 1023, 0, 255);
  servoBaseY.write(gradosy);


  Serial.println( grados);

}
```

Este seria el otro codigo que copiamos para saber con cual ir, procedo a explicarlo salteando algunas cosas que ya explique.

```
void setup() {
   servoBasex.attach(A0);//Pin a utilizar para servo
   servoBasex.write(0);  //asigno 0 al servo motor
   servoBaseY.attach(A1);//Pin a utilizar para servo
   servoBaseY.write(0);  //asigno 0 al servo motor
   pinMode(y,INPUT);
   pinMode(x,INPUT);
   Serial.begin(9600);
}
```

Configuración de servos:

- **`servoBasex.attach(A0)`** y **`servoBaseY.attach(A1)`**: Vincula los servos a los pines analógicos A0 y A1.
- **`servoBasex.write(0)`** y **`servoBaseY.write(0)`**: Posiciona ambos servos en el ángulo inicial de 0 grados.

Configuración de pines:

- Los pines `x` (A2) y `y` (A3) se configuran como entradas para leer los valores del joystick.

Serial:

- **`Serial.begin(9600)`**: Configura la velocidad de comunicación serial para monitorear datos en la consola.

```
    int valor = analogRead( A2 )
int valory = analogRead( A3 );
```

`analogRead(A2)`: Lee el valor analógico del pin `x` (0 a 1023).

`analogRead(A3)`: Lee el valor analógico del pin `y` (0 a 1023).

```
int grados = map(valor,  0, 1023, 0, 255);
int gradosy = map(valory,  0, 1023, 0, 255);
```

La función **`map()`** transforma un valor en un rango dado a otro:

- **Entrada**: `valor` o `valory` (de 0 a 1023, salida del joystick).
- **Salida**: `grados` o `gradosy` (de 0 a 255, rango del servo).

Esto ajusta la posición del servo en proporción a la entrada del joystick.

```
servoBasex.write(grados);
servoBaseY.write(gradosy);
```

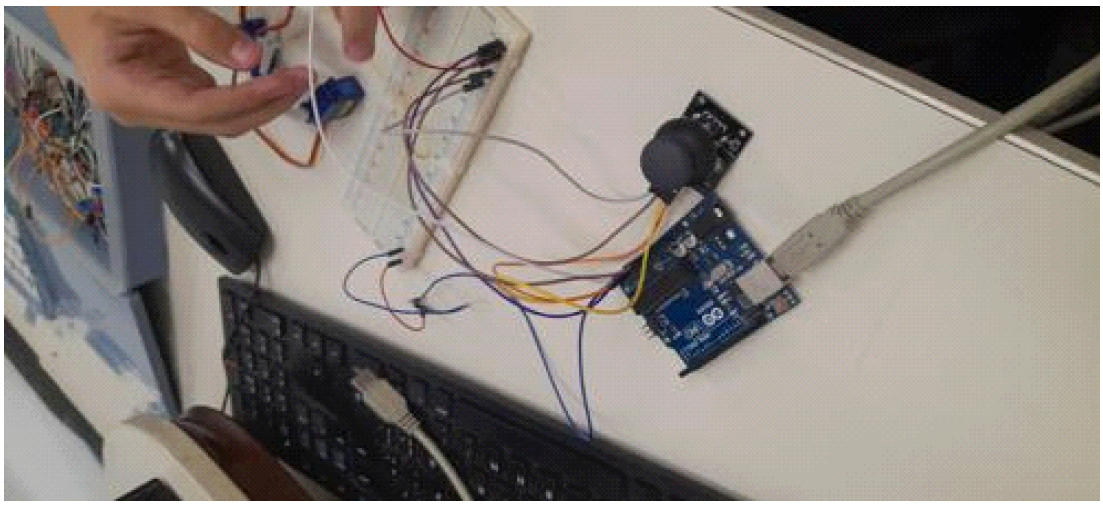
`servoBasex.write(grados)`: Posiciona el servo del eje X según el valor mapeado.

`servoBaseY.write(gradosy)`: Posiciona el servo del eje Y según el valor mapeado.

Al final Decidimos usar el primer codigo que es mas facil de usar y explicar solo tenemos que calibrar el Servomotor antes de usarlo.

Imagen 4: Análisis y Contexto

Descripción:Cables conectados

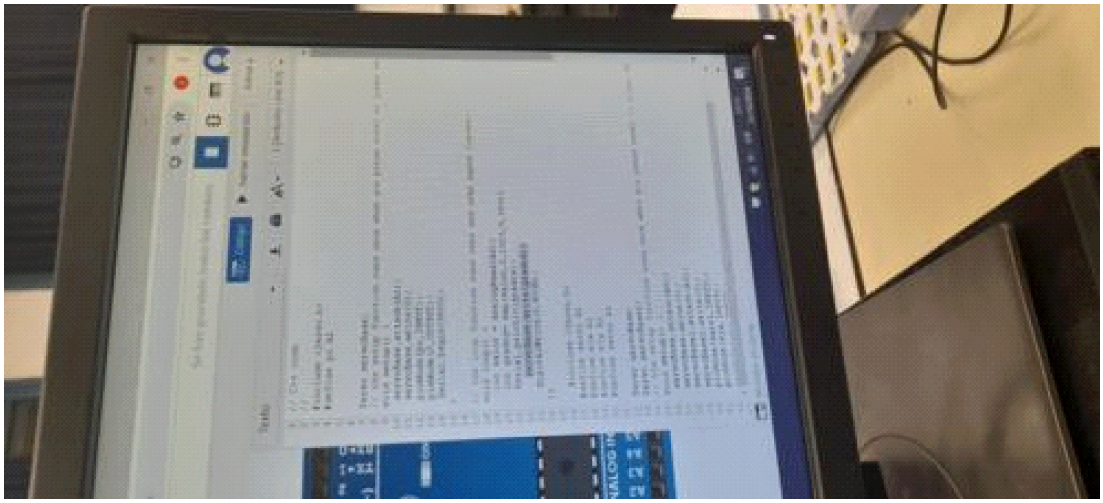


Cuando conectamos los dos servomotores y los ejes X y Y, probamos el código primero en Tinkercad usando potenciómetros, y todo funcionó correctamente. Sin embargo, cuando intentamos hacer la misma prueba con entradas analógicas en lugar de los potenciómetros, el sistema no funcionó de la misma manera.

El posible motivo de esta diferencia puede estar relacionado con cómo Tinkercad maneja las entradas analógicas en comparación con un sistema físico real. En Tinkercad, los potenciómetros pueden simular de manera más directa el comportamiento de señales analógicas, mientras que al usar entradas analógicas reales

Imagen 6: Análisis y Contexto

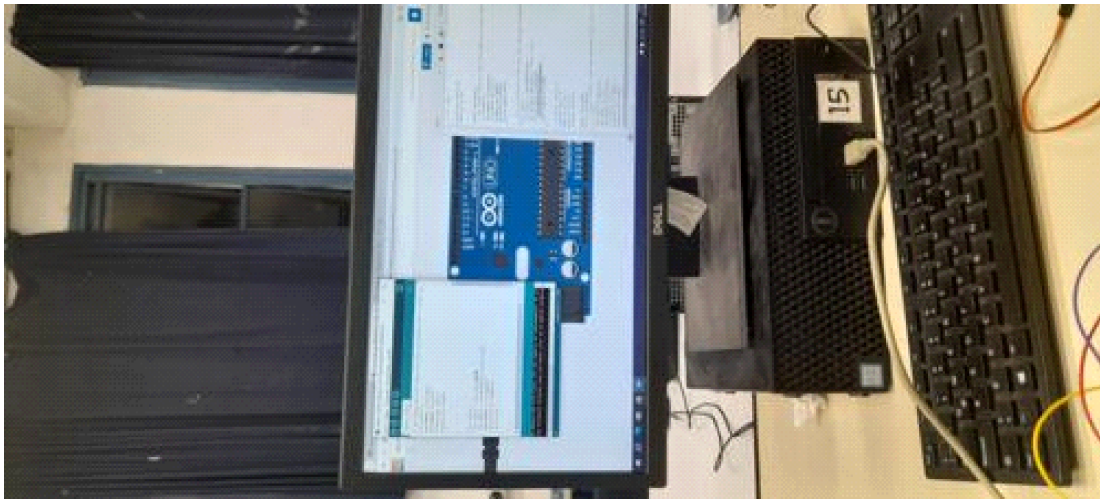
Descripción:



El codigo que probamos con los potenciometros porque no andaba con analogicos.

Imagen 7: Análisis y Contexto

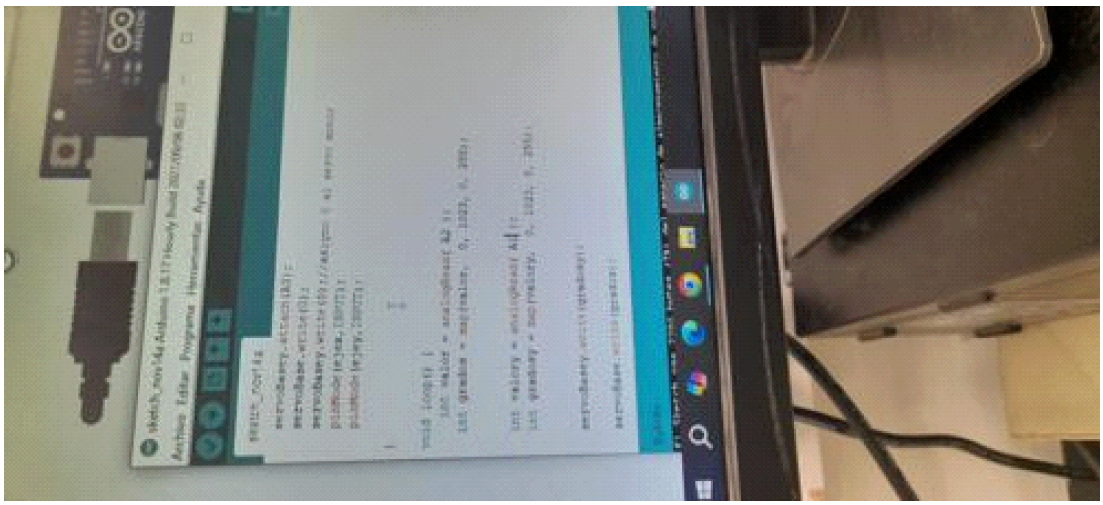
Descripción:



Estabamos probando como funcionaba el arduino fisico.

Imagen 8: Análisis y Contexto

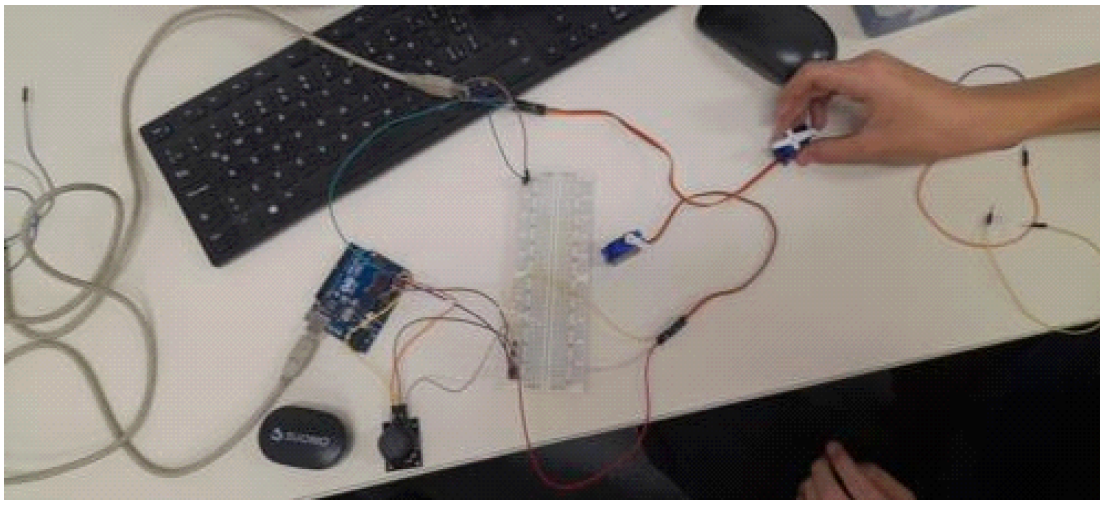
Descripción:



Estabamos probando este codigo para ver si bajaba el delay pero solo giraba un servomotor

Imagen 9:

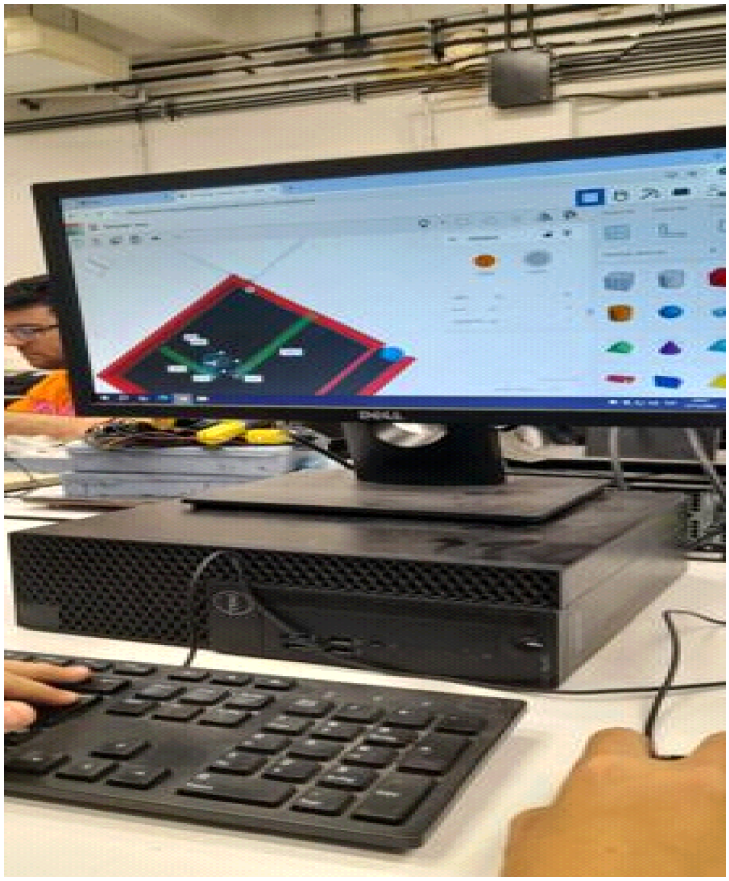
Descripción:



Como se conectan los servos y el analogico

Imagen 10:

Descripción:

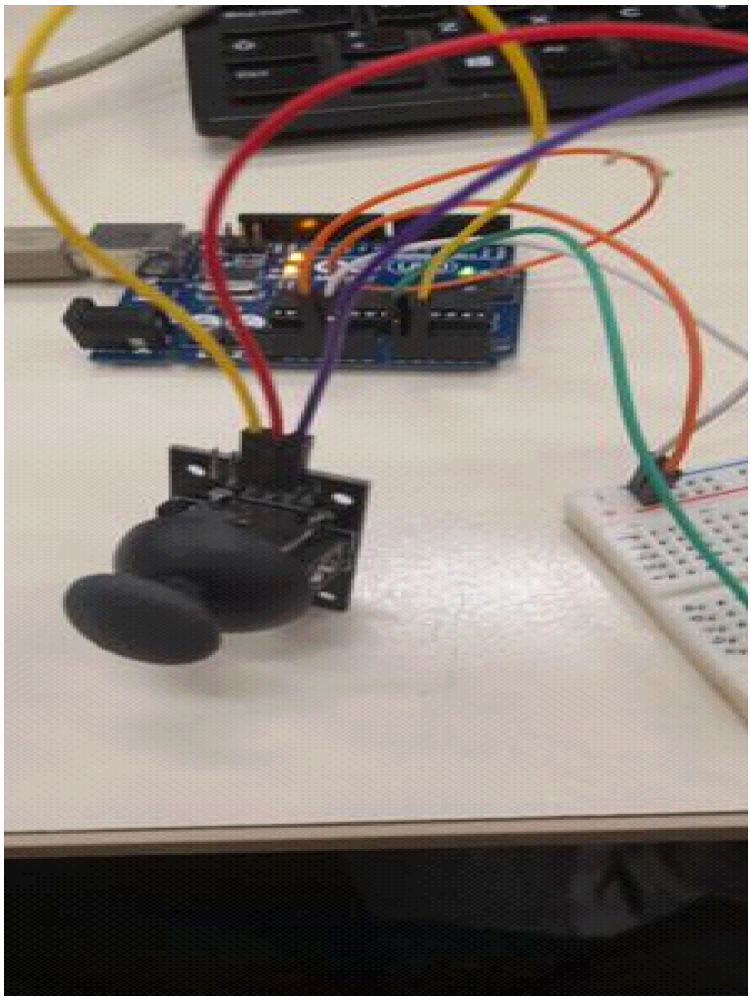


Acá estábamos en una de las etapas más importantes del proyecto, probando cómo quedaba mejor la entrada del laberinto. Ajustamos los ángulos y dimensiones para asegurarnos de que no solo fuera funcional, permitiendo un acceso fluido para la pelota, sino también estéticamente agradable, logrando un diseño que reflejara el cuidado en los detalles.

Además, nos enfocamos en la colocación de los muros interiores. Probamos distintas configuraciones para garantizar que fueran lo suficientemente sólidos como para guiar la pelota en el recorrido, pero sin ser un obstáculo que interfiera con su movimiento natural. El desafío estaba en encontrar un equilibrio entre la estabilidad de los muros y el espacio necesario para que la pelota pudiera desplazarse libremente.

Al mismo tiempo, trabajábamos en la construcción de la base del laberinto. Diseñamos una estructura estable y resistente que pudiera soportar el peso de todo el montaje, sin afectar la precisión de los movimientos controlados por los servomotores. Esto incluyó evaluar materiales, probar uniones y asegurarnos de que los servos tuvieran un rango de movimiento adecuado para inclinar la plataforma con suavidad y rapidez.

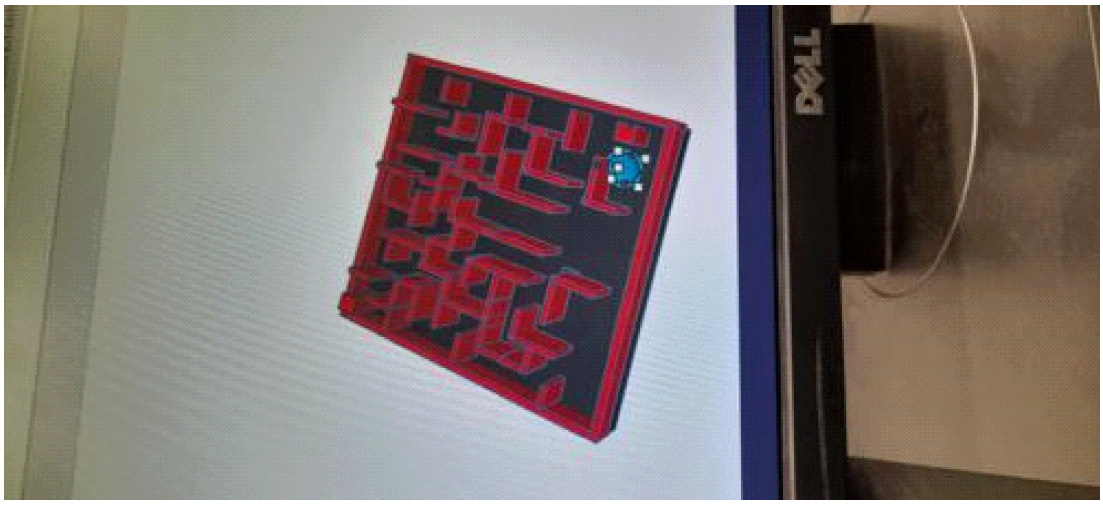
Imagen 11:



Descripción: Acá empezamos a probar con componentes analógicos, después de estar trabajando en Tinkercad con los potenciómetros y simulaciones básicas. La idea ahora es llevar lo aprendido al mundo físico, experimentando con circuitos reales y comprobando cómo interactúan los diferentes elementos en la práctica.

Nos enfocamos en los potenciómetros como un punto de partida porque son herramientas versátiles y esenciales para regular resistencias de forma manual. Además, permiten explorar conceptos como divisores de tensión y el control de señales analógicas. En esta etapa, conectamos los potenciómetros a placas de prueba (protoboards), fuentes de alimentación y multímetros para observar en tiempo real cómo varía la salida al girar la perilla.

Imagen 12:



diseñar el laberinto, ajustando detalles para que el recorrido de la pelota fuera fluido y desafiante. Nos enfocamos en que las paredes fueran lo suficientemente altas para evitar que la pelota se saliera, y en darle una disposición interesante al trazado para probar bien el control con los servomotores. Aunque simple, el diseño debía cumplir con la funcionalidad básica y permitir que el sistema funcionara correctamente en las pruebas.

Resultados y Ajustes

El proyecto demostró cómo utilizar un joystick analógico para controlar dos servomotores, aplicando los conceptos de lectura de entradas analógicas y mapeo de valores en un entorno práctico. La implementación permitió mover una plataforma que sirve como un laberinto para guiar una pelota hacia un objetivo deseado.

Durante las pruebas, se realizaron ajustes clave para mejorar el rendimiento del sistema:

1. **Calibración del joystick**: Esto aseguró que los valores de entrada analógica se correspondieran con los ángulos correctos de los servos, eliminando errores en los límites o movimientos inesperados.
2. **Optimización del código**: Se redujo el tiempo de respuesta del sistema ajustando la lógica de procesamiento y eliminando posibles retardos innecesarios.

Resultados obtenidos:

- El sistema mostró una respuesta adecuada en la mayoría de los escenarios, con movimientos suaves y precisos en cada eje.

- Sin embargo, se detectó un **retardo leve** al operar ambos ejes simultáneamente, posiblemente debido a las limitaciones del tiempo de procesamiento del microcontrolador o de los servos.