

# Portero Robótico

Aleany Lizeth Pérez Chavez  
A00836152

Gustavo Ángel Hidalgo Romero  
A00835599

Ximena Juen Chow Mendoza  
A01412194

Valeria Sofía Dávila de Ochoa  
A00836048

Fátima Ortiz Sánchez  
A01736896

**Resumen**— Este proyecto se enfoca en el diseño y construcción de un portero robótico capaz de moverse en dos ejes; realizando dos acciones principales interceptando tiros de balón: desplazarse a los lados y golpear la pelota hacia enfrente. Para el funcionamiento del proyecto fue necesario implementar conocimientos clave de la mecatrónica, como el diseño mecánico, selección de componentes de electrónica, técnicas de prototipado, así como la programación, utilizando un microcontrolador (Arduino) en adición con softwares de diseño CAD (SolidWorks) y de programación para el microcontrolador utilizado (Arduino IDE).

**Palabras clave** — portero robótico, mecatrónica, diseño mecánico, microcontrolador, electrónica.

## INTRODUCCIÓN

El diseño y la creación de un portero para un juego de mesa similar al futbolito. El objetivo es desarrollar un sistema que emplee componentes mecatrónicos implementando sistemas de prototipado, mecánicos, electrónicos, y programables para mover el portero a lo largo de la portería, reaccionando de manera efectiva a los movimientos del balón durante el juego. La integración de componentes mecánicos, electrónicos y principios de programación es esencial para el éxito de este proyecto. Se debe de considerar no sólo la eficacia del portero automatizado en términos de velocidad y precisión, sino también su robustez y fiabilidad durante el juego.

## MÉTODOS

Pregunta detonadora: ¿Cómo puede optimizarse el diseño de un portero automatizado para maximizar su eficiencia y precisión en un juego de mesa tipo futbolito?

### I. DISEÑO MECÁNICO

#### A. Prototipado

##### 1) Soportes

Los soportes utilizados, fueron pensados de acuerdo a la mecánica y funcionalidad del proyecto, los cuales fueron diseñados en Solid Works, para posteriormente por medio de la manufactura aditiva, hacer su impresión en 3D. (Véase Anexo para consultar los dibujos de los soportes elaborados)

El soporte del lado izquierdo se diseñó considerando un espacio con las dimensiones de el motor a paso, y con las perforaciones adecuadas para sujetar con tornillería y colocar la polea, en el del lado derecho se permitió un espacio para colocar la polea por donde pasa la banda e igual con las perforación indicadas.

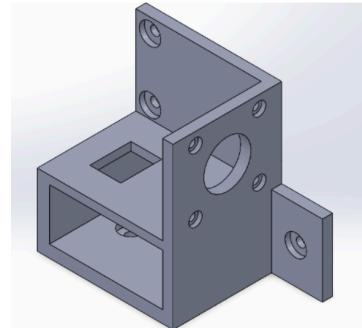


Ilustración 1 Soporte Izquierdo parte uno

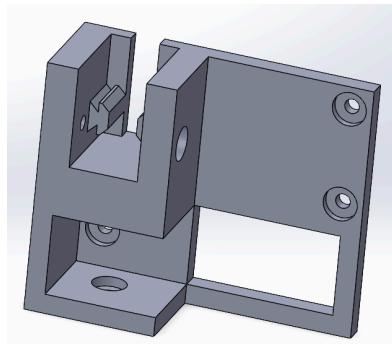


Ilustración 2 Soporte Izquierdo parte dos

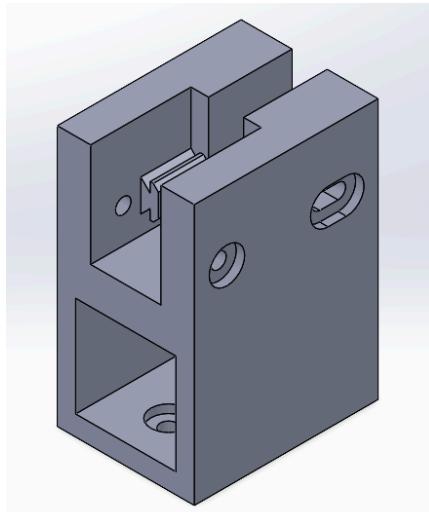


Ilustración 3 Soporte derecho

## 2) Portero

El portero está diseñado de forma rectangular la cual se sujetó al segundo soporte en donde se coloca el solenoide, unidas por el mismo solenoide, se colocaron dos perfiles para alcanzar la altura correspondiente, los que al mismo tiempo cuentan con una base unida al carrito y una placa de 90 grados.. Vease Ilustración 5 Portero.

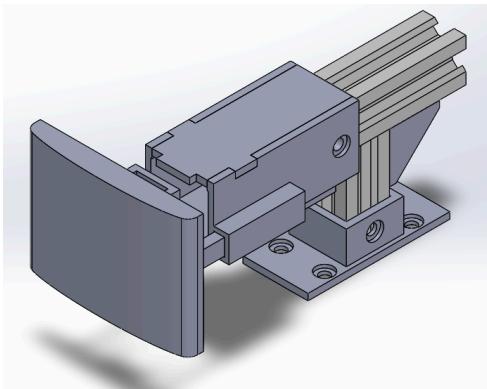


Ilustración 4 Portero

Material	Especificaciones	
	Medidas	Cantidad
Tornillo segmentado	M4 x 6mm	4
Tuerca T perfil 2020, rosca 3 mm	M4 x 6mm	4

De la misma manera se utilizó perfil v2020 de aluminio. Sus especificaciones técnicas incluyen un material de aluminio 6063 T-5, tamaño de 20x20mm, peso de 4.64g por cm, acabado superficial natural (plateado) y momentos de inercia de áreas de  $I_x = 6.988 \times 10^{-9} \text{ m}^4$  e  $I_y = 6.988 \times 10^{-9} \text{ m}^4$ . [1]

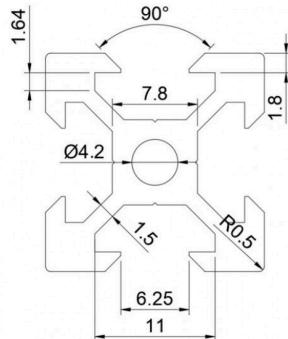


Ilustración 6 Dibujo técnico perfil V2020

## 3) Carcasa

El diseño de la carcasa no es completamente cerrado, se decidió dejar un espacio sin tapa para facilitar el acomodo a la mesa, en la parte superior se dejó el orificio por donde se desliza el portero. Considerando las uniones con los soportes y las salidas de cables de los componentes eléctricos, se dejaron los orificios correspondientes.

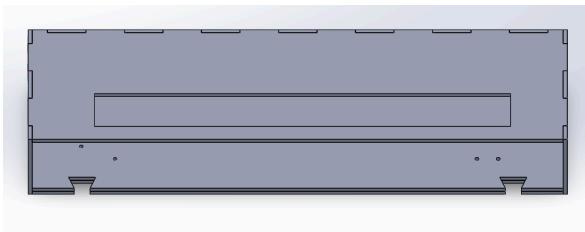


Ilustración 5 Carcasa

### B. Elementos mecánicos:

Para realizar las sujetaciones, se utilizó la tornillería adecuada, la cual se muestra en la siguiente tabla:

TABLE I. TORNILLERÍA

Material	Especificaciones	
	Medidas	Cantidad
Tornillo avellanado, con tuerca hexagonal	M6 x 3 / 4"	5
Tornillo segmentado, con tuerca hexagonal	M6 x 1 / 2"	3
Tornillo segmentado, con tuerca hexagonal	M6 x 3 / 8"	4
Tornillo segmentado, rosca 3 mm	M5 x 10mm	6
Tornillo segmentado, rosca 3 mm	M5 x 6mm	2

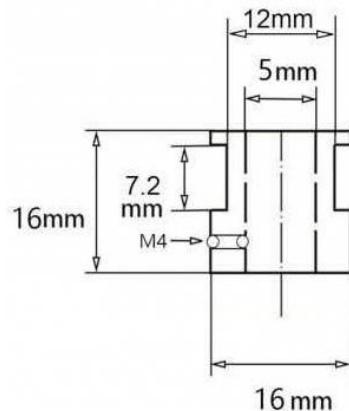


Ilustración 7 Dibujo técnico perfil V2020

TABLE 11. ESPECIFICACIONES DEL ARDUINO UNO

## II. CONTROL

### A. Descripción y características de microcontrolador y tarjeta de desarrollo

Se usó un Arduino Uno el cual cuenta con un microcontrolador ATmega 328P el cual es un microcontrolador de 8 bits, cuenta con 32KB de memoria flash, 2 KB de SRAM y 1KB de EEPROM, así como 23 pines de E/S digitales, 6 entradas analógicas y 6 salidas PWM. Tiene una velocidad que puede alcanzar hasta los 20 MHz, así como el microcontrolador principal en Arduino Uno, que ofrece facilidad de uso y una amplia comunidad de soporte. [2]

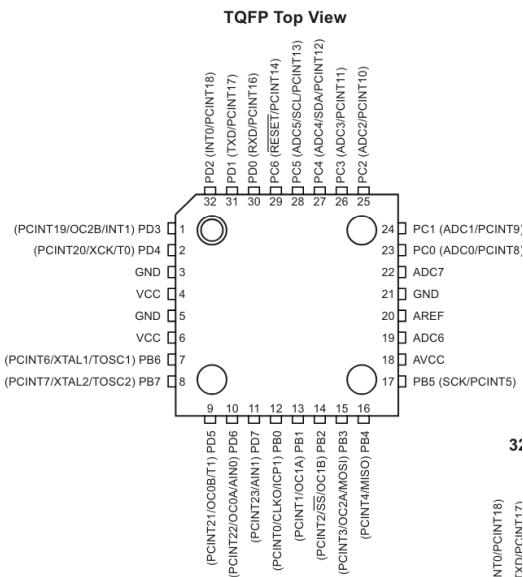


Ilustración 8 Configuraciones de Pin vista Frontal

### 1) Características Arduino Uno

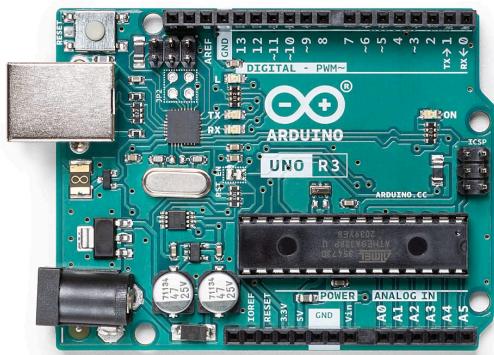


Ilustración 9 Arduino UNO

Características Arduino Uno	
Microcontrolador	ATmega328P
Voltaje de Operación	5V
Voltaje de Entrada (recomendado)	7 -12 V
Voltaje de Entrada (Límites)	6 - 20 V
Pines Digitales de Entrada/Salida	14 (de los cuales 6 proporcionan salida PWM)
Pines de Entrada Analógica	6
Corriente DC por Pin de E/S	40 mA
Corriente DC para Pin de 3.3V:	50 mA
Memoria Flash	32 KB
SRAM	2KB
EEPROM	1KB
Velocidad de Reloj	16 MHz

### 2) Características del Microcontrolador ATMEGA328P



Ilustración 10 Microcontrolador ATMEGA 328P

TABLE 11. ESPECIFICACIONES DEL ATMEGA 328P

Características ATMEGA 328P	
Arquitectura	8-bit AVR
Número de Pines	28
Programación	C/C++
Periféricos Integrados	Timers Comparadores analógicos PWM UART SPI I2C ADC
Pines Digitales de Entrada/Salida	14 (de los cuales 6 proporcionan salida PWM)
Interruptores	Soporte para interrupciones externas e internas.

## III. ELECTRONICA

### A. Componentes electrónicos

Para la realización del portero, se tuvieron que utilizar varios componentes electrónicos, entre ellos se encuentran:

- *Arduino UNO*: Es un microcontrolador con el ATMEGA328P que trabaja a 5V. Cuenta con 14 pines digitales de entrada/salida, 6 entradas analógicas, una conexión USB para la programación, un conector de alimentación y un botón de reinicio.



- *Motor Nema 17*: Es un motor paso a paso el cual tiene una alta precisión de posicionamiento, ya que rota en pequeños pasos discretos, generalmente 1.8 grados por paso y trabaja a 12 V.



- *Solenoid*: Dispositivo electromecánico que convierte energía eléctrica en movimiento lineal trabajando a 12V. a 2A.



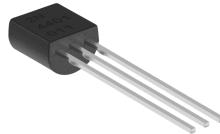
- *Driver A4988*: Controlador de motor paso a paso. Es un módulo que incluye un chip controlador capaz de manejar motores paso a paso con facilidad utilizando 5V a 12V.



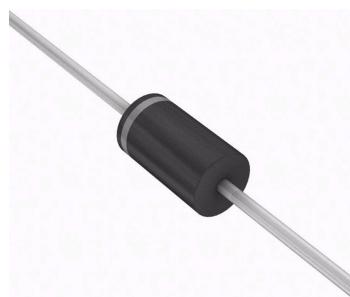
- *Joystick*: Dispositivo de entrada que permite controlar el movimiento en dos ejes (X e Y) el cual trabaja a 5V.



- *Push Button*: Interruptor momentáneo que cambia de estado cuando se presiona.



- *Transistor*: Componente semiconductor que se utiliza para amplificar o conmutar señales electrónicas que pasa 5V a 12V.



- *Diodo*: Componente electrónico que permite el paso de corriente en una sola dirección.



- *Resistencia 1k ohm*: Componente pasivo que se utiliza para limitar la corriente en un circuito.

#### IV. PROGRAMACIÓN

##### 1) Explicación del código

La primera parte del código consta en la configuración inicial, se definen constantes para los pines conectados a los distintos componentes. Por ejemplo, `dir`, `stp`, `MS1`, `MS2`, y `EN` están asignados al motor paso a paso, mientras que `SOLENOID` está conectado a un solenoide y `switchPin` a un botón.

También se configuran los pines como entradas o salidas según sea necesario y se inicia la comunicación Serial.

En cuanto al control de joystick, el código lee la posición del joystick analógico conectado al pin `A5`. Dependiendo de esta posición, controla la dirección del motor paso a paso

para mover el portero hacia adelante o hacia atrás. La dirección se cambia escribiendo un valor alto o bajo en el pin `dir`.

Para la operación del motor, se utilizan pulsos digitales enviados al pin `stp` para mover el motor paso a paso. Los pulsos se controlan mediante `delayMicroseconds(400)`, lo que ajusta la velocidad del motor. Además, el motor se puede habilitar o deshabilitar con el pin `EN`, permitiendo un control preciso de su funcionamiento.

El solenoide se controla leyendo el estado del botón a través de `switchPin`. Si el botón está presionado (estado bajo), el solenoide se activa enviando un alto al pin `SOLENOID`. Si no está presionado, el solenoide se desactiva.

El código asegura que el solenoide y el motor solo funcionen cuando se cumplan condiciones específicas, evitando operaciones accidentales o innecesarias.

## 2) Funcionamiento Integrado

El flujo del programa permite una interacción dinámica en tiempo real con el juego de futbolito. Al manipular el joystick, el usuario puede mover el portero a través del campo, mientras que el solenoide podría actuar como un mecanismo de acción rápida para momentos críticos, como bloquear un tiro. El sistema no solo tiene que responder rápidamente a las entradas del usuario sino también hacerlo de manera fiable y precisa para mantener la jugabilidad y la competencia en el juego. El uso de un Arduino para este tipo de aplicación demuestra cómo la microcontroladora puede ser efectiva en la creación de sistemas de control interactivos y automatizados en contextos recreativos.

Además, la implementación de la comunicación serial es crucial para la depuración y el desarrollo del software, permitiendo al desarrollador hacer ajustes en tiempo real y monitorear el comportamiento del sistema durante las pruebas.

## CONLCUSIÓN

El desarrollo del sistema del portero automático para el juego de futbolito de mesa ha sido un proyecto desafiante y gratificante que ha demostrado el potencial de la integración de la mecánica, electrónica y programación en la creación de soluciones innovadoras e interactivas. Este proyecto no solo ha cumplido con el objetivo de simular los movimientos defensivos de un portero en un juego de futbolito, sino que también ha mejorado la experiencia del juego haciéndola más dinámica y atractiva.

A través del uso de componentes impresos en 3D y la implementación de controladores electrónicos como Arduino, se ha creado un sistema robusto y confiable que responde eficazmente a los comandos del usuario. La utilización de motores paso a paso y solenoides, controlados por señales precisas de un microcontrolador, ha permitido

movimientos suaves y precisos que simulan la agilidad de un portero real.

Este reto también ha servido como una excelente plataforma de aprendizaje, proporcionando una comprensión profunda de cómo diversos componentes y sistemas pueden trabajar juntos para lograr un objetivo complejo. La capacidad para ajustar la respuesta del sistema en tiempo real a través de la programación ha demostrado ser particularmente valiosa, ofreciendo lecciones aplicables en muchos otros campos de diseño y desarrollo de productos.

En conclusión, el proyecto del portero automático ha sido un éxito significativo, reflejando no solo la viabilidad técnica de la idea, sino también el potencial para futuras innovaciones y aplicaciones en juegos interactivos y otros sistemas automatizados. Este proyecto ha establecido un precedente para futuras investigaciones y desarrollos en la intersección de la tecnología y el entretenimiento, abriendo la puerta a nuevas posibilidades en el diseño de sistemas mecatrónicos interactivos.

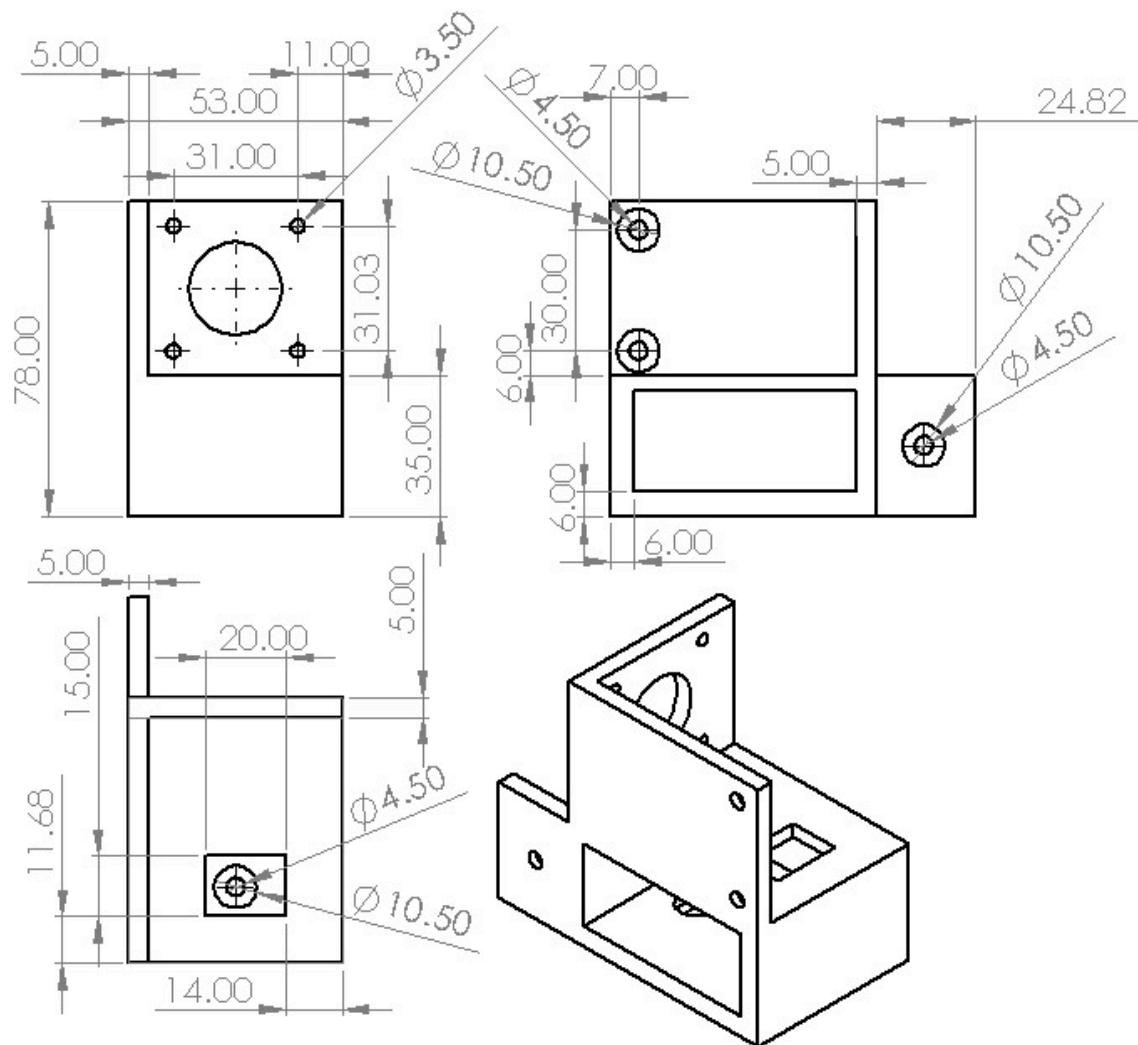
## REFERENCIAS

- [1] *Perfil V-Slot 2020 x centímetro*. (n.d.). Naylamp Mechatronics - Perú. <https://naylampmechatronics.com/perfiles-de-aluminio/693-perfil-v-slot-2020-x-centimetro.html>
- [2] Microchip Technology Inc. (2015). *ATmega328P [Automotive microcontroller] datasheet* (Atmel 7810). [https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P\\_Datasheet.pdf](https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf)
- [3] *Arduino® UNO R3*. (n.d.). <https://docs.arduino.cc/resources/datasheets/A000066-datasheet.pdf>
- [4] *Joystick Module*. (2018). Components101. <https://components101.com/modules/joystick-module>
- [5] *Features and Benefits* • Low R DS(ON) outputs • Automatic current decay mode detection/selection • Mixed and Slow current decay modes • Synchronous rectification for low power dissipation • Internal UVLO • Crossover-current protection • 3.3 and 5 V compatible logic supply • Thermal shutdown circuitry • Short-to-ground protection • Shorted load protection • Five selectable step modes: full. (n.d.). [https://www.pololu.com/file/0J450/a4988\\_DMOS\\_micostepping\\_driver\\_with\\_translator.pdf](https://www.pololu.com/file/0J450/a4988_DMOS_micostepping_driver_with_translator.pdf)
- [6] www.alldatasheet.com.mx. (2022). *LB22F2BQ6506 datasheet(2/6 Pages) E-SWITCH*. Alldatasheet.com.mx. <https://www.alldatasheet.com.mx/html-pdf/1509831/E-SWITCH/LB22F2BQ6506/911/2/LB22F2BQ6506.html>

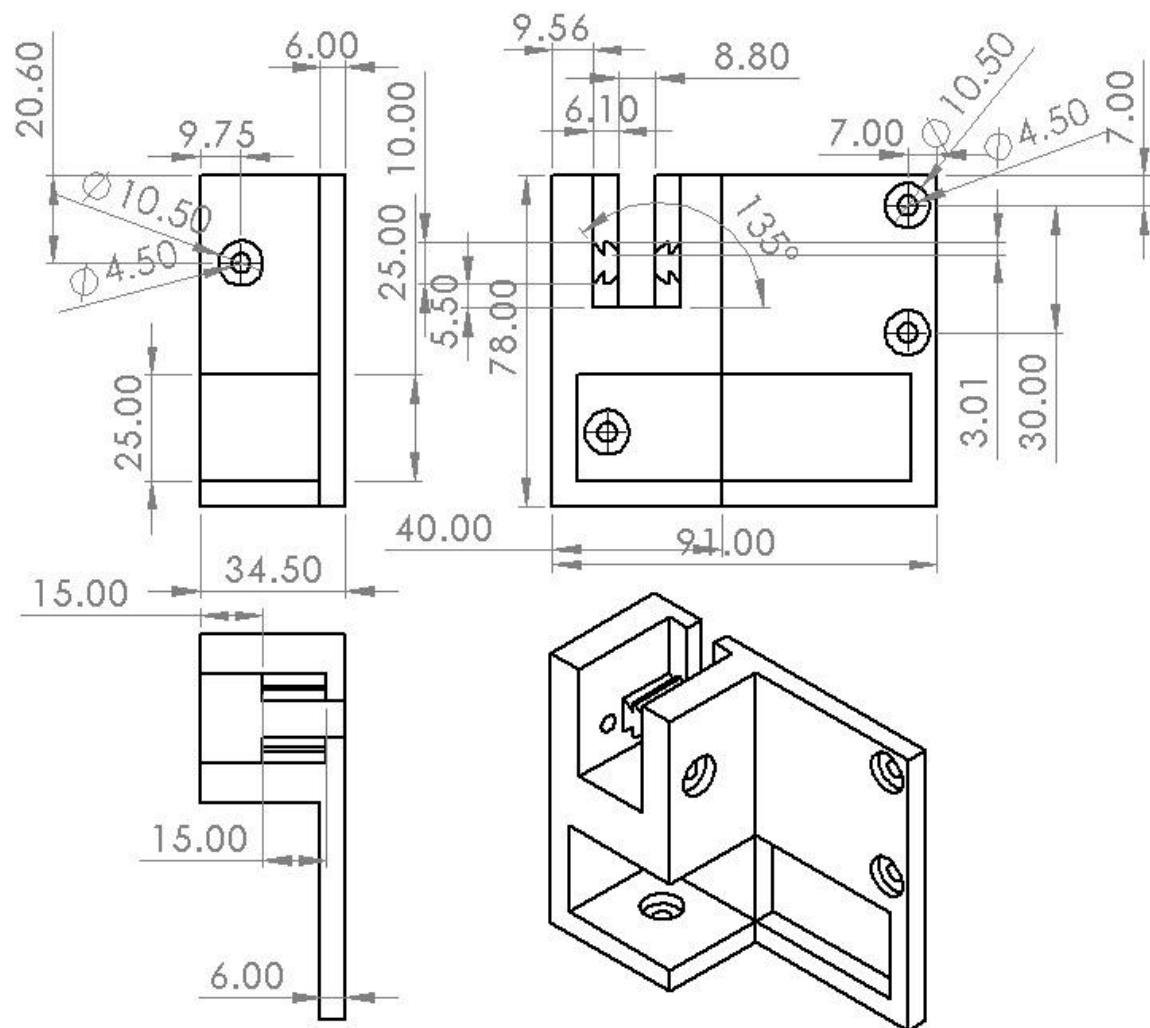
ANEXO

1. Dibujo de los componentes

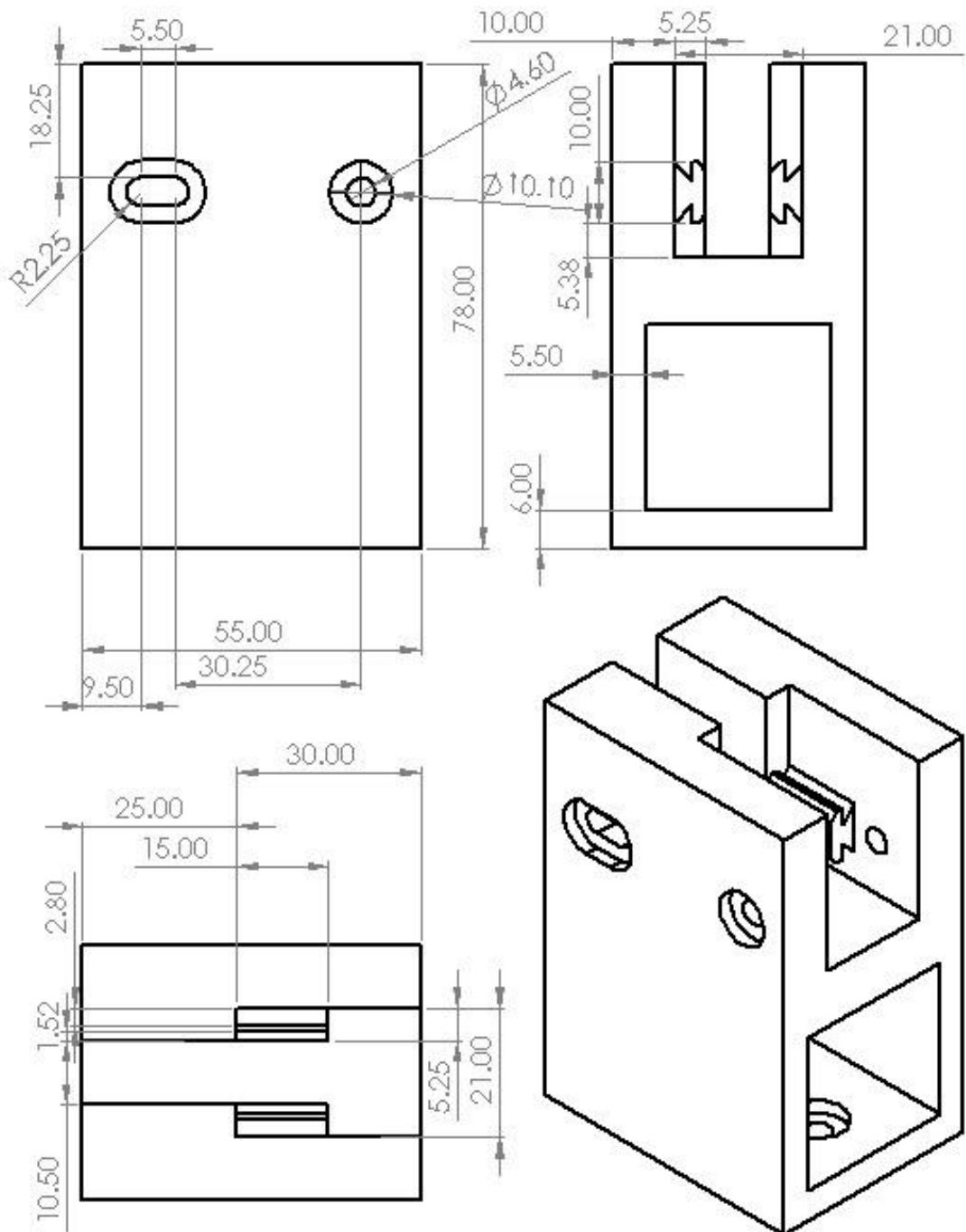
1.1 Soporte Izquierdo parte uno



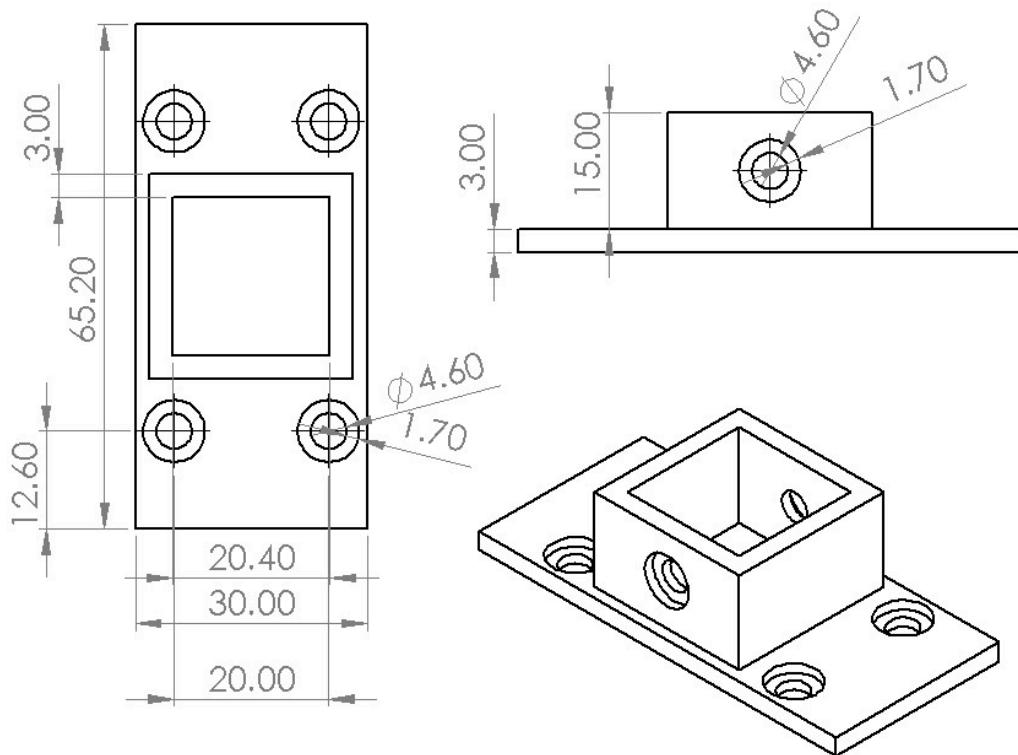
1.2 Soporte Izquierdo parte dos



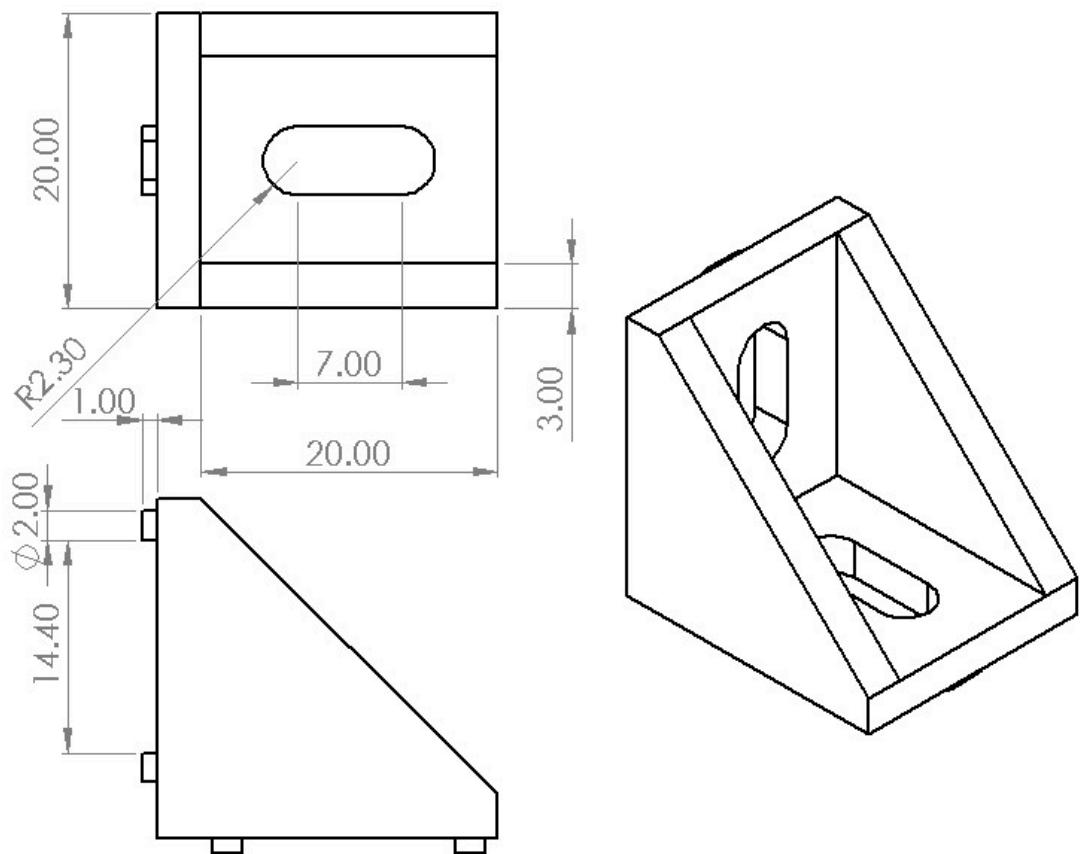
### 1.3 Soporte Derecho



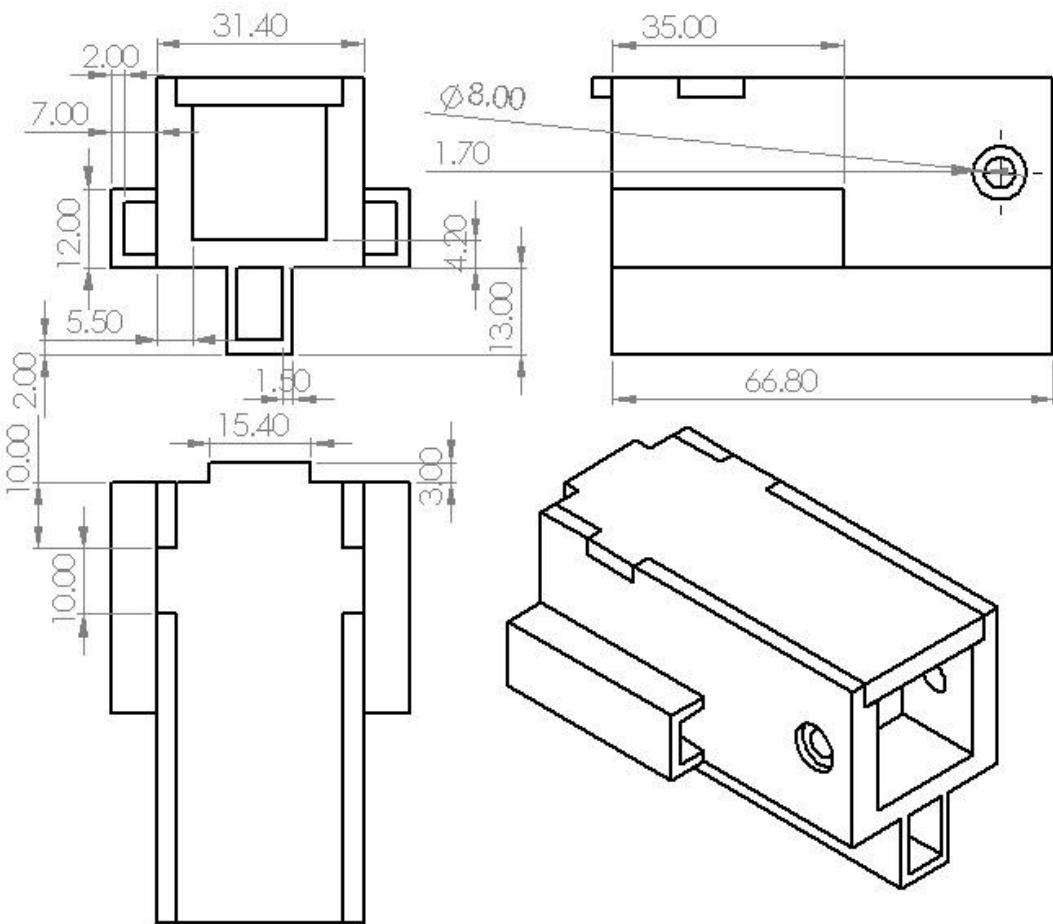
1.4 Base para el carrito



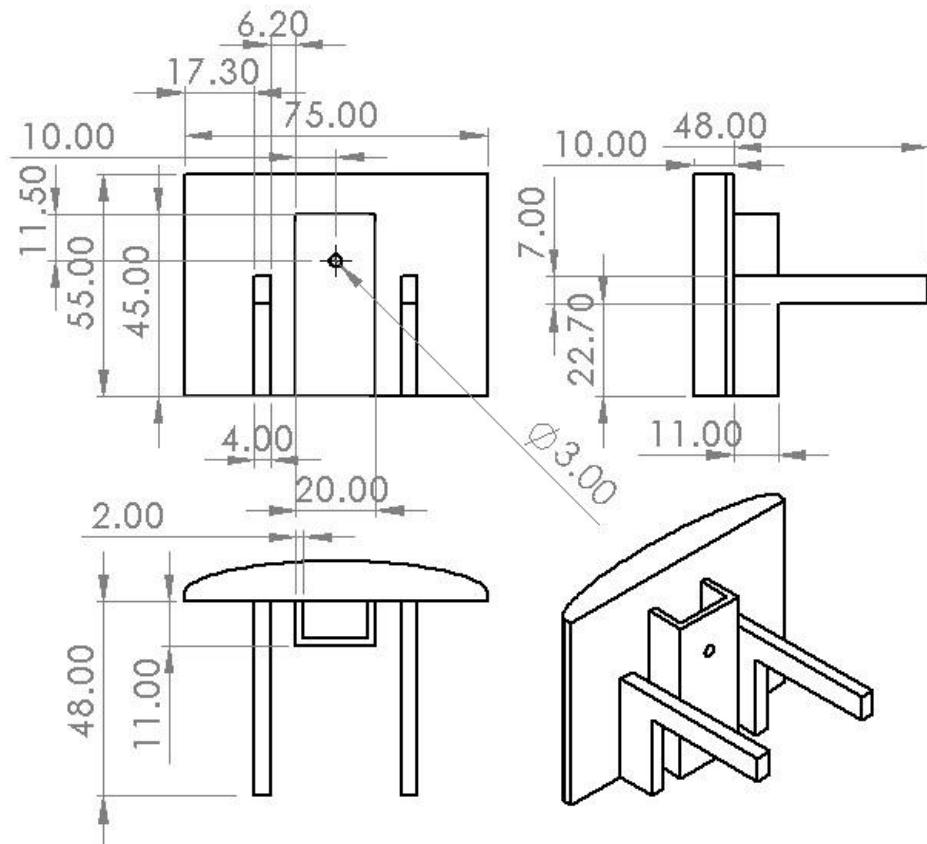
1.5 Placa de unión 90 grados



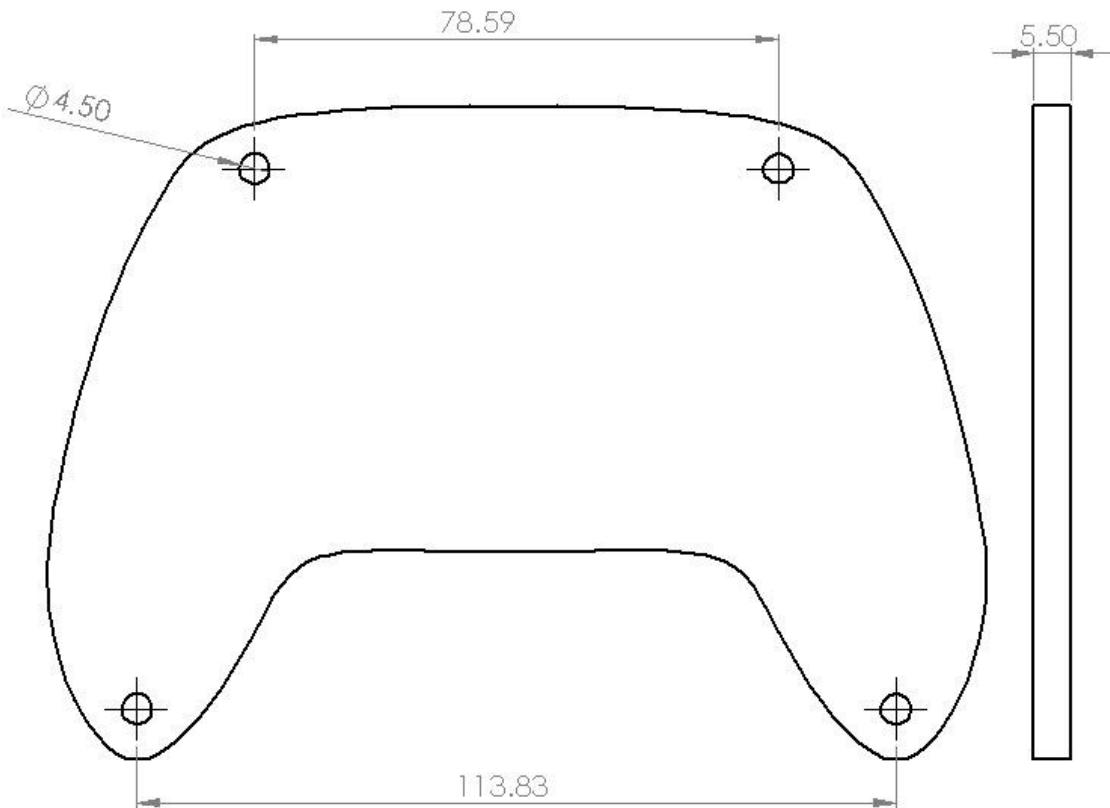
### 1.6 Soporte del solenoide



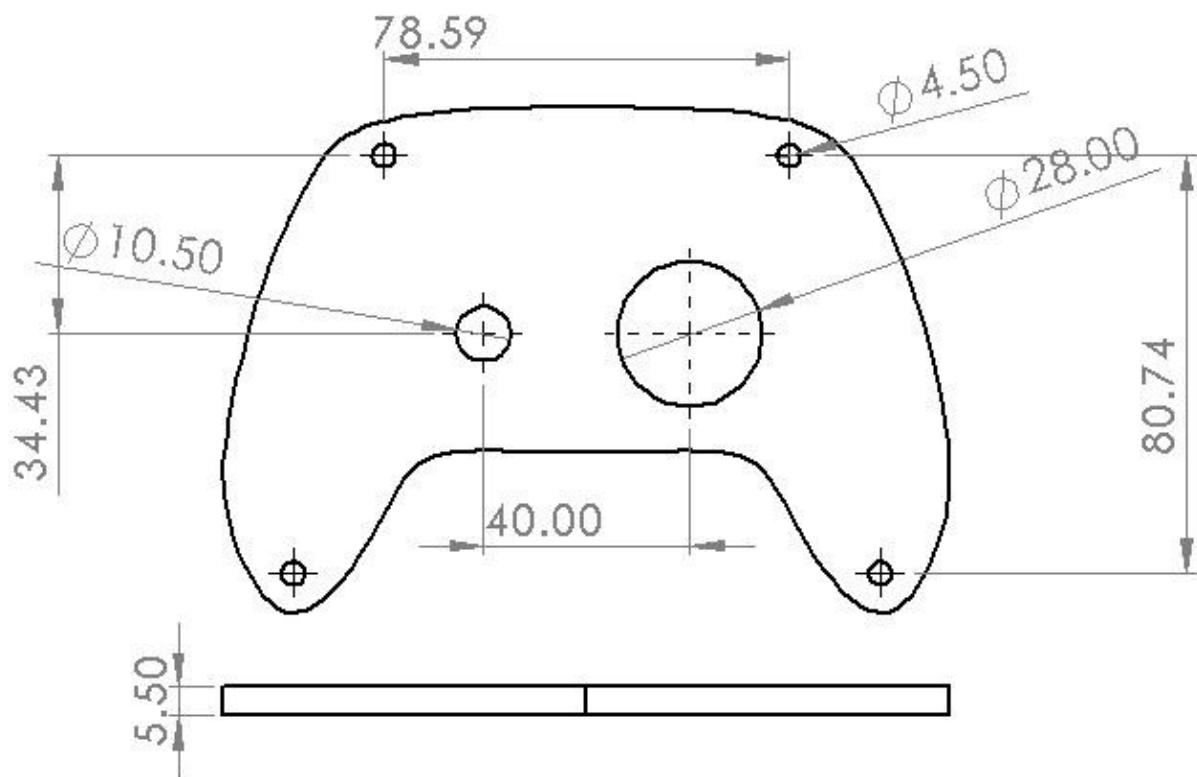
### 1.7 Portero



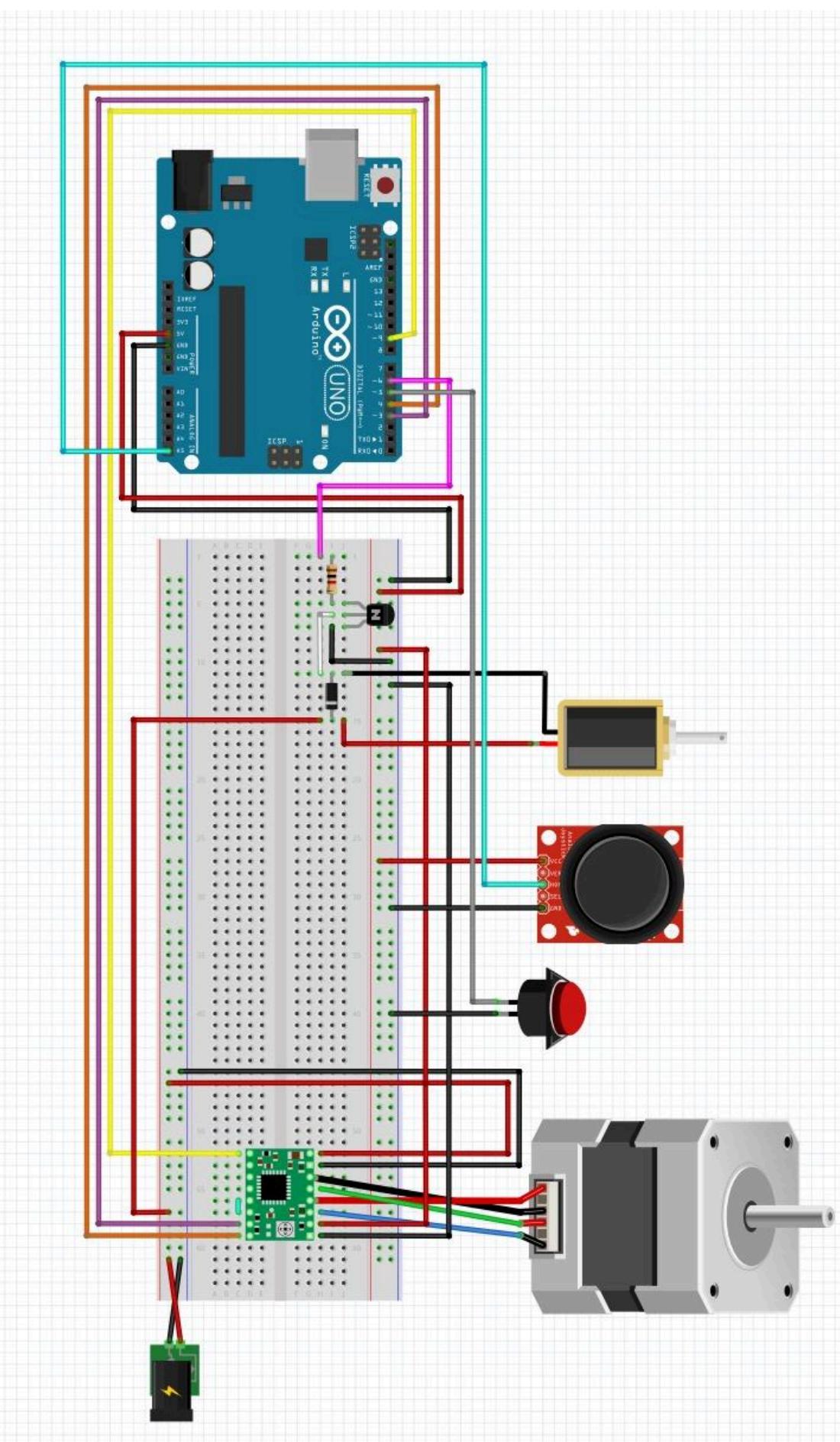
1.8 Control (Base)



1.9 Control (Tapa)



## 2. Circuito Implementado



### 3. Código de Programación

C/C++

```
// Definiciones para el motor
#define stp 3
#define dir 4
#define MS1 8
#define MS2 7
#define EN 9
// Definiciones para el solenoide
#define SOLENOID 6
#define switchPin 5
// Variables
char user_input;
int x;
int y;
int state;
int joyX = A5;
int joyVal;

void setup() {
    // Motor setup
    pinMode(stp, OUTPUT);
    pinMode(dir, OUTPUT);
    pinMode(MS1, OUTPUT);
    pinMode(MS2, OUTPUT);
    pinMode(EN, OUTPUT);

    digitalWrite(stp, LOW);
    digitalWrite(dir, LOW);
    digitalWrite(MS1, LOW);
    digitalWrite(MS2, LOW);
    digitalWrite(EN, HIGH);

    // Solenoide setup
    pinMode(SOLENOID, OUTPUT);
    pinMode(switchPin, INPUT_PULLUP);
```

```

    Serial.begin(9600);
}

void loop() {
    // Control del motor
    while((analogRead(joyX) == 0) || (analogRead(joyX) > 1000)) {
        joyVal = analogRead(joyX);
        digitalWrite(EN, LOW);

        if(joyVal == 0){
            digitalWrite(dir, LOW);
        }
        else if (joyVal > 1000){
            digitalWrite(dir, HIGH);
        }

        digitalWrite(stp,HIGH);
        delayMicroseconds(400);
        digitalWrite(stp,LOW);
        delayMicroseconds(400);
    }
    digitalWrite(EN, HIGH);

    // Control del solenoide
    int btn_Status = digitalRead(switchPin);
    if (btn_Status == LOW) {
        digitalWrite(SOLENOID, HIGH);
    } else {
        digitalWrite(SOLENOID, LOW);
    }
}

```

4. Carpeta hacia prototipo final

<https://drive.google.com/drive/folders/1aCFNIsByXJA9E3nXQwTFHHL84OZFN9RJ?usp=sharing>