

**Tank-esito**

Andrea Sarahi Avalos Salcedo, Verónica Barragán Zepeda, Gisel Maqueda Regalado, Axel López López, Ricardo Flores Sánchez, Cristian García Sánchez, Martin Eduardo Ruvalcaba Mendoza, José Alberto Cárdenas Aguilar, Jorge Humberto Avalos Gálvez, Ignacio Chávez Núñez

Instituto Tecnológico Nacional de México Campus Jiquilpan

Sistemas Programables y Graficación

Mtros. Francisco Armando Payan Guerrero y Erick Dalet Villanueva Mascort

05 de Junio de 2024

**Resumen**

En este proyecto se desarrolló un sistema de control a distancia para un tanque utilizando diversas tecnologías, incluidas Arduino, ESP32, Unity, y múltiples componentes electrónicos. El proyecto aborda aspectos cruciales como la configuración de la red y la conectividad, la integración entre Unity y Arduino, y el diseño de la lógica de control del tanque.

La configuración de la red se realizó utilizando un módem LINKSYS EA4500, asegurando una conexión estable a través de un proceso meticuloso de conexión y configuración del módem, creación de la red WiFi, y aseguramiento mediante un cambio de nombre y contraseña. Se utilizó la aplicación Fing para monitorear los dispositivos conectados a la red, garantizando así la seguridad y la eficiencia de la red.

La integración entre Unity y Arduino se logró mediante la implementación de una clase conectionArduino.cs en Unity, utilizando Visual Studio. Esta clase maneja la conexión TCP con el módulo WiFi ESP32, permitiendo la transmisión de datos entre Unity y el hardware del tanque. La conexión se establece con un método Connect, mientras que los datos se envían a través de SendData, y la conexión se cierra mediante CloseConnection.

En cuanto al diseño del sistema de control, el ESP32 se configuró para conectarse a la red WiFi y recibir comandos desde Unity. El código desarrollado en Unity maneja los controles del tanque, permitiendo el movimiento a través de varios botones. Además, se implementó un sistema de cámara utilizando una RawImage en Unity para mostrar imágenes capturadas por la cámara de la computadora, ya que no se pudo conectar una cámara externa al tanque. Este sistema incluye funcionalidad de zoom y una máscara circular para mejorar la visualización.

Adicionalmente, se implementó una retícula en la interfaz de usuario del juego. La clase Reticula gestiona la visualización de líneas horizontales y verticales en la imagen del juego. La retícula puede ser mostrada o escondida mediante la interacción del usuario, facilitando la alineación y orientación en el entorno visual del juego.

Este proyecto integra de manera efectiva múltiples tecnologías para crear un sistema robusto y funcional, proporcionando un control preciso y una visualización detallada del entorno del tanque a través de una interfaz intuitiva en Unity.

**Tabla de Contenidos**

[Capítulo 1 Introducción 6](#_Toc410628920)

* 1. [Antecedentes 7](#_Toc410628921)
  2. [Objetivos Generales 8](#_Toc410628922)
  3. [Objetivos Especificos 1](#_Toc410628922)

[Capítulo 2 Descripción del Proyecto 9](#_Toc410628925)

2.1. [Alcance del Proyecto 9](#_Toc410628921)

2.2. [Limitaciones 10](#_Toc410628921)

2.3. [Restricciones 11](#_Toc410628921)

[Capítulo 3 Requisitos Funcionales 12](#_Toc410628925)

3.1. [Controles del Tanque 12](#_Toc410628921)

3.2. [Visualización en Unity 12](#_Toc410628921)

3.3. Elevación del Cañón [12](#_Toc410628921)

3.4. [Lanzamiento de Proyectiles 12](#_Toc410628921)

3.5. [Comunicación Inalámbrica 13](#_Toc410628921)

[Capítulo 4 Requisitos No Funcionales 13](#_Toc410628925)

4.1. [Rendimiento 1](#_Toc410628921)3

4.2. [Usabilidad 13](#_Toc410628921)

4.3. [Confiabilidad 1](#_Toc410628921)3

4.4. [Mantenimiento 1](#_Toc410628921)4

4.5. [Seguridad 1](#_Toc410628921)4

4.6. [Compatibilidad 1](#_Toc410628921)4

[Capítulo 5 Tecnologías y Herramientas Utilizadas 14](#_Toc410628925)

5.1. [Unity 15](#_Toc410628921)

5.2. [Arduino Wifi (ESP32) 15](#_Toc410628921)

5.3. [Motores DC 16](#_Toc410628921)

5.4. [Servomotores 16](#_Toc410628921)

5.5. [Impresora 3D 17](#_Toc410628921)

5.6. [Baterias 18](#_Toc410628921)

[Capítulo 6 Arquitectura del Sistema 18](#_Toc410628925)

6.1. [Descripción de la integración entre Unity y Arduino 18](#_Toc410628921)

6.2. Descripción de la Arquitectura de Comunicación  [22](#_Toc410628921)

[Capítulo 7 Planificación y Cronograma 23](#_Toc410628925)

[Capítulo 8 Diseño y Fabricación de Piezas con Impresoras 3D 2](#_Toc410628925)4

8.1. [Selección de Materiales 24](#_Toc410628921)

8.2. [Modelado 3D de las Piezas 25](#_Toc410628921)

8.3. [Proceso de Impresión 3D 25](#_Toc410628921)

8.4. [Pintura y Acabado 26](#_Toc410628921)

[Capítulo 9 Diseño del Tanque con Componentes Electrónicos 27](#_Toc410628925)

9.1. [Estructura del Tanque 27](#_Toc410628921)

9.2. [Integración de Componentes Electrónicos 33](#_Toc410628921)

[9.2.1 Posicionamiento y Fijación de los Componentes. 34](#_Toc410628927)

[9.2.2. Conexión de los Componentes al Arduino Mega y al ESP32. 34](#_Toc410628927)

[9.2.3. Cableado y Enrutamiento de Cables. 34](#_Toc410628927)

9.3. [Ensamblaje de Piezas y Componentes Electrónicos 35](#_Toc410628921)

[Capítulo 10 Configuración de la Red y Conectividad 37](#_Toc410628925)

10.1. [Conexión del modem 38](#_Toc410628921)

10.2. [Creación de la Red 38](#_Toc410628921)

10.3. [Cambio de nombre y Estructura de la Red 39](#_Toc410628921)

10.4. [Conexión del Tanque a la Red WiFi 39](#_Toc410628921)

[Capítulo 11 Diseño del Sistema de Control 40](#_Toc410628925)

11.1. Implementación de la Lógica de Control del Tanque [40](#_Toc410628921)

[Capítulo 12 Archivos Adjuntos 42](#_Toc410628925)

12.1. Cámara [42](#_Toc410628921)

12.2. Archivo Device [47](#_Toc410628921)

12.3. Archivo Reticula [53](#_Toc410628921)

[Capítulo 13 Pruebas y Depuración Final del Sistema 57](#_Toc410628925)

13.1. Pruebas de Operación Completa  [58](#_Toc410628921)

13.2. Verificación de la Conectividad WiFi  [58](#_Toc410628921)

13.3. Ajustes Finales [59](#_Toc410628921)

# Capítulo 1 Introducción

En la intersección entre la electrónica y el desarrollo de videojuegos, surge un proyecto: el desarrollo de un tanque que puede ser controlado y gestionado a través de Unity, una popular plataforma de desarrollo de juegos en 3D. Este proyecto fusiona la ingeniería de hardware con la programación de software para crear una experiencia interactiva única.

El objetivo principal de este proyecto es diseñar un sistema que permita a los usuarios controlar un tanque. Para lograr esto, se implementarán varios componentes electrónicos, como motores, sensores y controladores, que interactuarán con el software de Unity a través de una conexión bidireccional.

En primer lugar, se abordará la parte de la electrónica, que involucra la construcción del propio tanque. Esto implica la selección de las impresiones en 3D de todas las piezas del tanque, los motores adecuados para el movimiento del tanque y la retroalimentación del entorno. Se explorarán diferentes técnicas de control, como PWM (modulación por ancho de pulso) para el control de velocidad de los motores y el uso de sensores ultrasónicos o de infrarrojos para la detección de obstáculos.

Una vez completada la parte de hardware, se procederá a la fase de desarrollo de software utilizando Unity. Aquí, se crearán las configuraciones necesarias para que el tanque se pueda mover y controlar. Se implementarán controles de usuario que permitan al jugador interactuar con el tanque, como la capacidad de moverse, girar y disparar.

La comunicación entre el hardware y el software será crucial para el éxito del proyecto. Se utilizarán protocolos de comunicación estándar, como Wi-Fi, para establecer una conexión confiable entre el tanque y la aplicación de Unity. Esto permitirá que los comandos enviados desde Unity se traduzcan en acciones físicas realizadas por el tanque, y que los datos recopilados por los sensores se utilicen para actualizar el estado del entorno virtual en tiempo real.

## Antecedentes

El uso de plataformas de desarrollo como Unity para el control de dispositivos físicos ha ganado popularidad en los últimos años debido a su flexibilidad y capacidad de integración con diferentes componentes electrónicos. En proyectos anteriores, se han utilizado sistemas similares para controlar robots y vehículos de manera remota, aprovechando las capacidades de programación de Unity y la conectividad de los microcontroladores como el ESP32. Estos proyectos han demostrado la viabilidad de integrar software y hardware para crear sistemas controlados de manera remota con aplicaciones en educación, entretenimiento y prototipado rápido. En este proyecto específico, se ha partido de un prototipo existente, ajustando sus dimensiones mediante el software Cura para adaptarse a los requerimientos actuales.

## Objetivo General

El objetivo principal del proyecto es desarrollar un tanque controlado electrónicamente que pueda ser manejado de forma remota utilizando Unity como plataforma de control. Implicando la integración de componentes electrónicos, con el fin de proporcionar una experiencia interactiva y funcional que permita el control del movimiento y el disparo del tanque.

## Específicos:

* Garantizar que el tanque responda de manera precisa y sin demoras a las órdenes enviadas desde la aplicación en el móvil o a través de la computadora, asegurando un control remoto efectivo.
* Asegurar que el tanque se mueva sin problemas en todas las direcciones.
* Lograr que el cañón del tanque se eleve y baje de forma suave y precisa, permitiendo apuntar con precisión antes de disparar.
* Garantizar que el mecanismo de lanzamiento de proyectiles cargue, apunte y dispare las balas NERF.
* Asegurar que las piezas impresas en 3D sean lo suficientemente resistentes para resistir el uso continuo del tanque.
* Asegurar que el tanque sea controlado de manera intuitiva mediante el teclado y el ratón en Unity, además de la aplicación móvil para control desde el celular.
* Realizar pruebas detalladas del tanque en diferentes situaciones y entornos, identificando cualquier problema y realizando ajustes necesarios para garantizar su funcionamiento óptimo.

# Capítulo 2 Descripción del Proyecto

El proyecto consiste en la creación de un tanque controlado por electrónica y software desarrollado en Unity, basado en la modificación de un prototipo existente. El tanque estará compuesto por una serie de componentes electrónicos, incluyendo Arduino Mega, ESP32 (Arduino WiFi), servomotores, motores DC, un módulo driver de motor, entre otros. Las partes físicas del tanque serán fabricadas mediante impresión 3D, siguiendo las dimensiones especificadas. El control del tanque se realizará a través de teclas y ratón en Unity, con la posibilidad adicional de controlarlo mediante una aplicación móvil, ya que también se desarrollará un APK utilizando el software de Unity. El tanque contará con una movilidad mediante los motores DC y tendrá capacidades de disparo controladas por servomotores y motores de drones, utilizando proyectiles de balas Nerf.

## 2.1. Alcance del Proyecto

El proyecto abarca el desarrollo integral de un tanque controlado remotamente, incluyendo la modificación de piezas de un prototipo existente, la impresión de dichas piezas utilizando impresoras 3D, la integración de componentes electrónicos y el desarrollo de un sistema de control en Unity. Se espera lograr un prototipo funcional que pueda ser controlado de forma remota a través de Unity. El tanque contará con funcionalidades de movimiento y disparo, operadas a través de Unity que se comunicará con un ESP32 para gestionar los motores y servomotores necesarios. La culminación del proyecto implicará un tanque completamente ensamblado y funcional, capaz de ser controlado remotamente mediante comandos en Unity y a través de un dispositivo móvil.

## 2.2. Limitaciones

* Limitación de tiempo: El tiempo disponible para la realización del proyecto puede ser limitado, lo que puede afectar en cuanto a la profundidad de las pruebas y la optimización del sistema.
* Capacidades del Hardware: La precisión y velocidad del control del tanque están limitadas por las capacidades del ESP32 y los motores DC utilizados. Los servomotores y motores de drones también tienen limitaciones en cuanto a su potencia y respuesta.
* Conectividad: La comunicación entre Unity y el ESP32 está sujeta a la estabilidad de la conexión Wi-Fi, lo que puede limitar el rango de operación y la confiabilidad del control remoto.
* Duración de la Batería: El tiempo de operación del tanque está limitado por la capacidad de las baterías utilizadas para alimentar los motores y componentes electrónicos.

## 2.3. Restricciones

* Restricción de materiales: La fabricación de las piezas del tanque se realizará exclusivamente mediante impresión 3D, lo que limita la selección de materiales disponibles.
* Restricción de tamaño: Las dimensiones del tanque estarán definidas por las capacidades de la impresora 3D utilizada, lo que puede influir en el diseño y la funcionalidad del vehículo.
* Restricción de peso: El tanque deberá mantener un peso adecuado para garantizar su movilidad y estabilidad, lo que puede limitar la cantidad y el tipo de componentes electrónicos que se pueden integrar.
* Restricción de energía: La energía disponible para alimentar los componentes electrónicos del tanque estará limitada, lo que puede influir en el diseño y la eficiencia energética del sistema.
* Restricciones de Tiempo y Recursos: El proyecto debe completarse dentro de un tiempo determinado y con los recursos disponibles. Esto incluye limitaciones en el tiempo de desarrollo, la disponibilidad de impresoras 3D, y el presupuesto para componentes electrónicos.

# Capítulo 3 Requisitos Funcionales

## 3.1. Controles del Tanque

* El tanque debe poder moverse hacia adelante, hacia atrás, hacia la izquierda y hacia la derecha, utilizando las teclas del teclado o los controles táctiles en la aplicación móvil.

## 3.2. Visualización en Unity

* Permitir el control del tanque exclusivamente a través del teclado y ratón, sin una interfaz gráfica adicional en Unity.

## 3.3. Elevación del Cañón

* Se debe implementar la capacidad de elevar y bajar el cañón del tanque, permitiendo apuntar antes de disparar, controlado ya sea desde Unity o mediante la aplicación móvil.

## 3.4. Lanzamiento de Proyectiles

* El tanque debe ser capaz de cargar, apuntar y disparar proyectiles (balas NERF), con la opción de realizar disparos precisos utilizando los controles de Unity o de la aplicación móvil.

## 3.5. Comunicación Inalámbrica

* El sistema debe utilizar el ESP32 para establecer una comunicación inalámbrica con Unity, permitiendo el control remoto del tanque.

# Capítulo 4 Requisitos No Funcionales

## 4.1. Rendimiento

* El sistema debe ser capaz de procesar los comandos de control en tiempo real, con una latencia mínima entre el envío de comandos desde Unity y la respuesta del tanque.

## 4.2. Usabilidad

* Debe ser fácil de usar y comprender tanto em Unity como en la aplicación móvil, permitiendo controlar el tanque sin necesidad de conocimientos técnicos avanzados.

## 4.3. Confiabilidad

* El sistema debe ser robusto y funcionar de manera consistente, evitando fallos durante la operación del tanque.

## 4.4. Mantenimiento

* El diseño del tanque debe permitir un fácil acceso a los componentes internos para facilitar su mantenimiento y reparación.

## 4.5. Seguridad

* El sistema debe incluir medidas de seguridad para evitar daños a los usuarios y al entorno, especialmente al considerar el uso de balas NERF y la operación de motores.

## 4.6. Compatibilidad

* Los componentes electrónicos deben ser compatibles entre sí y con el ESP32, asegurando una integración sin problemas.

# Capítulo 5 Tecnologías y Herramientas Utilizadas

Para la implementación y control del tanque, se han utilizado diversas tecnologías y herramientas que facilitan tanto el diseño, la construcción, como el control del tanque, permitiendo una integración eficiente de hardware y software, asegurando que el tanque funcione correctamente y responda a los comandos. A continuación, se detallan las principales tecnologías y herramientas empleadas en este proyecto.

## 5.1. Unity

Unity es un motor de desarrollo de videojuegos que permite crear experiencias interactivas en 2D y 3D, conocido por su flexibilidad y capacidad para integrar diferentes componentes y tecnologías. En este proyecto, Unity sirve como el entorno de desarrollo principal para la creación del software de control del tanque. A través de Unity, los usuarios pueden controlar los movimientos y el sistema de disparo del tanque de manera efectiva. Además, Unity implementa los algoritmos de control necesarios, enviando comandos específicos al ESP32 para gestionar los motores y servomotores del tanque. También establece y mantiene la comunicación inalámbrica entre el entorno de desarrollo y el hardware del tanque, garantizando una respuesta rápida y precisa a los comandos del usuario. Además, se desarrolló una aplicación móvil APK en Unity para permitir el control del tanque desde dispositivos móviles, proporcionando una opción adicional y conveniente para los usuarios.

## 5.2. Arduino Wifi (ESP32)

El ESP32 es un microcontrolador que incorpora conectividad Wi-Fi y Bluetooth, lo que lo convierte en una herramienta versátil para el control remoto. En este proyecto, el ESP32 es el elemento principal del sistema de control del tanque, desempeñando múltiples funciones importantes. Recibe los comandos enviados desde Unity a través de la red Wi-Fi, procesando estos comandos para controlar los motores DC y los servomotores. Esto incluye gestionar la velocidad y dirección de los motores de propulsión, así como el ajuste preciso de la posición del cañón del tanque para las funciones de elevación. La capacidad de procesamiento del ESP32 y su conectividad inalámbrica es esencial para el control remoto del tanque en tiempo real.

## 5.3. Motores DC

Los motores de corriente continua (DC) son componentes esenciales en el proyecto, ya que proporcionan los movimientos necesarios para que el tanque se desplace. En total, se utilizan cuatro motores DC para mover el tanque en diferentes direcciones y controlar su velocidad. Estos motores son conocidos por su simplicidad y eficiencia, ya que su velocidad se puede controlar fácilmente mediante la variación del voltaje que reciben. Esto permite una gran precisión en el control del movimiento del tanque. Estos motores se controlan a través del L298P módulo driver de motor Arduino UNO R3 Shield, que regula la velocidad y dirección de los motores en respuesta a los comandos recibidos del ESP32.

## 5.4. Servomotores

Los servomotores son motores especiales equipados con un sistema de control integrado que permite posicionar el eje en una posición específica con gran precisión. En este proyecto, se utiliza un servomotor grande para controlar el cañón del tanque, permitiendo elevarlo y bajarlo según sea necesario para disparar las balas NERF. La precisión de los servomotores es fundamental para asegurar que el cañón se posicione correctamente, lo que es crucial para la funcionalidad de disparo. Estos servomotores se controlan mediante señales PWM (Modulación por Ancho de Pulso) que la ESP32 genera, lo que permite un control preciso y fiable del movimiento del cañón.

## 5.5. Impresora 3D

La impresión 3D es una tecnología de fabricación que ha revolucionado la producción de piezas personalizadas y prototipos. En este proyecto, se utiliza una impresora 3D para fabricar todas las piezas del tanque, siguiendo las dimensiones especificadas. La impresión 3D ofrece una gran flexibilidad en el diseño, permitiendo así iterar rápidamente y ajustar las piezas según sea necesario durante el proceso de desarrollo. Además, la impresión 3D es una opción rápida para producir piezas en pequeñas cantidades.

Las piezas estructurales del tanque fueron impresas utilizando una impresora Shenzhen Creality. El software Cura se empleó para preparar los archivos de impresión, ajustando parámetros como la densidad de relleno. Las piezas impresas incluyen el cuerpo del tanque, los costados, las tapas, los engranajes y las orugas. La capacidad de fabricar componentes precisos y personalizados a través del uso de impresoras 3D permite un ensamblaje más fácil y eficiente del tanque, asegurando que todas las piezas encajen perfectamente y funcionen como se espera.

## 5.6. Baterías

Se utilizan baterías para alimentar los diferentes componentes del tanque. En total, se emplean cuatro baterías, dos baterías grandes para alimentar los motores DC que proporcionan la energía necesaria para el movimiento del tanque, y dos baterías más pequeñas, una blanca y una azul, para alimentar el Arduino Mega y el ESP32, respectivamente. Estas baterías proporcionan la energía necesaria para el funcionamiento del sistema electrónico del tanque, asegurando su correcto rendimiento durante las operaciones.

Estas tecnologías y herramientas han sido seleccionadas y utilizadas cuidadosamente para asegurar que el tanque funcione de manera óptima, permitiendo así un control preciso.

# Capítulo 6 Arquitectura del Sistema

## 6.1. Descripción de la integración entre Unity y Arduino

Para integrar Unity con Arduino, se desarrolló una clase llamada conectionArduino.cs en Visual Studio. Esta clase facilita la comunicación entre Unity y el módulo WiFi ESP32. Dentro de la clase, se utiliza un cliente TCP (TcpClient) para establecer la conexión con el módulo, y un flujo de red (NetworkStream) para enviar y recibir datos.

El método Connect se encarga de establecer la conexión con el módulo ESP32, recibiendo como parámetros el nombre del host y el número de puerto al que se conectará el cliente TCP.

El método SendData permite enviar datos al módulo ESP32. Toma como parámetro los datos que se enviarán al módulo y los convierte en bytes antes de enviarlos a través del flujo de red.

Finalmente, el método CloseConnection se utiliza para cerrar la conexión con el módulo ESP32, asegurando una gestión adecuada de los recursos y la liberación de la conexión.

A continuación, se muestra el código completo que se utilizó:

using System;

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using System.Net.Sockets;

using System.Text;

/// <summary>

/// Clase conectionArduino para manejar la conexión con el módulo WiFi ESP8266.

/// </summary>

public class conectionArduino

{

// Cliente TCP para la conexión con el módulo WiFi ESP8266

private TcpClient client;

// Flujo de red para enviar y recibir datos

private NetworkStream stream;

/// <summary>

/// Método Connect para establecer la conexión con el módulo WiFi ESP8266.

/// </summary>

/// <param name="host">El nombre del host al que se conectará el cliente TCP.</param>

/// <param name="port">El número de puerto en el host al que se conectará el cliente TCP.</param>

public void Connect(string host, int port)

{

client = new TcpClient(host, port);

stream = client.GetStream();

}

/// <summary>

/// Método SendData para enviar datos al módulo WiFi ESP8266.

/// </summary>

/// <param name="data">Los datos que se enviarán al módulo WiFi ESP8266.</param>

public void SendData(string data)

{

if (client != null)

{

byte[] bytes = Encoding.ASCII.GetBytes(data);

stream.Write(bytes, 0, bytes.Length);

}

}

/// <summary>

/// Método CloseConnection para cerrar la conexión con el módulo WiFi ESP8266.

/// </summary>

public void CloseConnection()

{

stream.Close();

client.Close();

}

}

## 6.2. Descripción de la Arquitectura de Comunicación

La arquitectura de comunicación entre Unity y Arduino se basa en una conexión TCP/IP a través de Wi-Fi. Unity actúa como el cliente TCP, mientras que el módulo WiFi ESP32 funciona como el servidor.

Cuando Unity necesita enviar comandos al tanque, llama al método SendData de la clase conectionArduino, proporcionando los datos necesarios para controlar el movimiento y otras acciones del tanque.

Por otro lado, el módulo ESP32, que está escuchando en un puerto específico, recibe los datos enviados por Unity a través de la conexión TCP. Luego interpreta estos datos y ejecuta las acciones correspondientes, como controlar los motores para el movimiento del tanque o ajustar la posición del cañón.

Esta arquitectura de comunicación permite una interacción fluida entre Unity y Arduino, permitiendo el control remoto del tanque desde una aplicación de escritorio o móvil con Unity.

# Capítulo 7 Planificación y Cronograma

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Actividad** | **Objetivos** | **Asignaciones** | **Fecha de inicio** | **Fecha de termino** |
| A partir del 2 de Mayo del 2024 | | | | |
| Calibración de la SP32 | Lograr la conexión con código y en físico de un arduino uno y SP32, posteriormente con Unity | - Conseguir la ESP32. Realizar el código de la conexión, y hacer pruebas con el puerto serial | 2 de mayo del 2024 | En curso |
| Armado de las orugas y pruebas con motores (solo circuitos con corriente) | Comprender el armado de las piezas en 3d, además de su armado, integración de las orugas y los engranajes, armado de los motores | Impresión de piezas en 3d, armado de las piezas de la oruga, prueba de motores con la pila temporal, armado del mecanismo de las orugas | 2 de mayo del 2024 | En curso |
| Implementación arduino | Creación del código Arduino para las conexiones precisas entre los componentes | Conseguir avances mediante la velocidad de los motores y conexión serial entre los componentes como los leds y unity | 8 de mayo del 2024 | En curso |
| Alambrado de componentes fuera del prototipo | Alambrado de los microcontroladores | Alambrado Arduino ESP32, alambrado arduino motores, integración de la corriente y la tierra de forma general para los diferentes componentes | 9 de Mayo del 2024 |  |
| Pintura del prototipo | Lograr un aspecto más profesional del proyecto | Pintura del prototipo y armado posterior con la nueva imagen | 14 de Mayo del 2024 |  |
| Pruebas y depuración finales | Depurar el proyecto y dejarlo funcionando a cómo se pide | Echar a andar el servidor wifi dónde se almacenarán las instrucciones de acción del prototipo. Hacer pruebas de los indicadores del proyecto, y búsqueda de posibles errores | 15 de mayo del 2024 |  |
| Presupuestos y administración de logística | Conocer los gastos hasta el momento del proyecto | Sacar las sumas pertinentes, el gasto total, | 9 de Mayo del 2024 | En curso |

# Capítulo 8 Diseño y Fabricación de Piezas con Impresoras 3D

El proceso de diseño y fabricación de las piezas del tanque es una parte crucial del proyecto, ya que asegura que todos los componentes encajen y funcionen correctamente. A continuación, se presenta un análisis detallado de cada fase, desde la selección de materiales hasta el proceso de impresión.

## 8.1. Selección de Materiales

La selección adecuada de materiales es esencial para garantizar la calidad y funcionalidad de las piezas del tanque. Para este proyecto, se opta por el uso de filamento PLA (ácido poliláctico) debido a sus propiedades versátiles y su facilidad de impresión. El PLA es un polímero biodegradable y biocompatible derivado de recursos naturales como el almidón de maíz o la caña de azúcar, lo que lo hace respetuoso con el medio ambiente.

Los filamentos PLA en los colores White, Red, Black y Green, con un peso de 1KG cada uno, se seleccionan para diferenciar visualmente las distintas partes del tanque y agregar un toque estético al proyecto. Además, se asegura que los parámetros de impresión del PLA se ajusten a las especificaciones del fabricante, con una temperatura de impresión entre 190°C y 230°C y una temperatura de la cama caliente entre 40°C y 50°C.

Además, el PLA ofrece una combinación de resistencia y facilidad de uso que lo hace ideal para aplicaciones como la fabricación de piezas de tanques RC. Su capacidad para producir detalles finos y su resistencia a la deformación térmica lo convierten en una opción confiable para este proyecto.

## 8.2. Modelado 3D de las Piezas

El modelado 3D de las piezas del tanque es un proceso importante que requiere precisión y atención a los detalles. Aunque se parte de un prototipo existente, se emplea el software Cura para optimizar las dimensiones y la geometría de las piezas. Cura ofrece herramientas avanzadas que permiten ajustar parámetros como la densidad de relleno, la velocidad de impresión y el soporte, lo que garantiza una impresión eficiente y resistente.

Durante el modelado, se consideran aspectos funcionales, estéticos y de integración con otros componentes del tanque. Se busca maximizar la resistencia de las piezas mientras se minimiza el consumo de material y el tiempo de impresión. Además, se realizan pruebas de ajuste para asegurar que las piezas encajen correctamente y cumplan con los requisitos de diseño.

## 8.3. Proceso de Impresión 3D

Una vez completado el modelado 3D de las piezas y optimizados los ajustes en Cura, se procederá al proceso de impresión 3D. Este proceso se llevará a cabo utilizando la impresora Shenzhen Creality y los filamentos PLA seleccionados previamente.

Antes de iniciar la impresión, se verificará que la impresora esté correctamente calibrada y nivelada para garantizar una adhesión adecuada a la plataforma de impresión. Se ajustarán los parámetros de impresión según las recomendaciones del fabricante del filamento y las especificaciones de diseño de las piezas.

El proceso de impresión 3D puede llevar varias horas o incluso días, dependiendo del tamaño y la complejidad de las piezas. Durante el proceso de impresión, se supervisará el funcionamiento de la impresora para detectar cualquier anomalía o problema técnico que pueda afectar la calidad de las piezas, como atascos de filamento o desprendimiento de capas, y se realizarán ajustes si es necesario.

Una vez completada la impresión de todas las piezas, se procederá a retirarlas con cuidado de la plataforma y se realizará un proceso de acabado si es necesario, como eliminación de soportes o lijado de superficies rugosas.

## 8.4. Pintura y Acabado

Después de imprimir y lijar las piezas, se procederá a la fase de pintura. Para este proyecto, las piezas del tanque serán pintadas con pintura en aerosol negra y amarilla para crear un patrón de camuflaje. Las orugas del tanque también serán pintadas en negro con detalles de camuflaje amarillo.

La pintura en aerosol se elige por su capacidad de proporcionar una cobertura uniforme y un acabado duradero. Se aplicarán varias capas finas para evitar acumulaciones y asegurar una adherencia adecuada. Primero, se limpiarán las piezas impresas para eliminar cualquier residuo de polvo o grasa que pueda afectar la adherencia de la pintura.

El patrón de camuflaje se aplicará mediante el uso de plantillas y técnicas de enmascarado para obtener un diseño consistente y realista. Se alternarán capas de negro y amarillo, permitiendo que cada capa se seque completamente antes de aplicar la siguiente. Este proceso no solo mejora la estética del tanque, sino que también proporciona una capa adicional de protección a las piezas impresas en 3D.

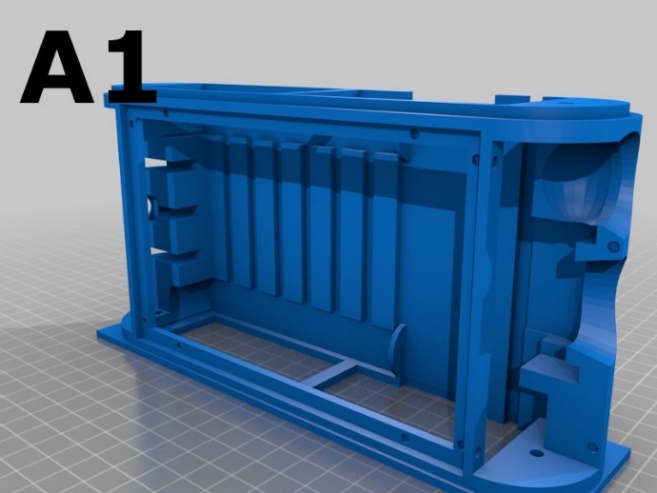
# Capítulo 9 Diseño del Tanque con Componentes Electrónicos

El diseño del tanque no solo se centra en la fabricación precisa de sus piezas físicas mediante impresión 3D, sino que también se integran diversos componentes electrónicos que permiten su funcionamiento remoto. Este proceso incluye la estructuración del tanque, la integración de los componentes electrónicos y el ensamblaje final de todas las piezas.

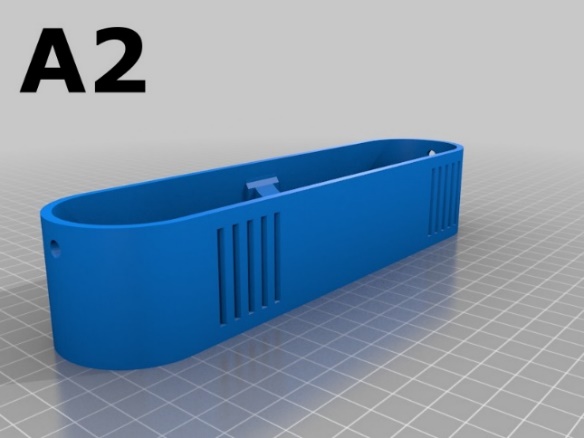
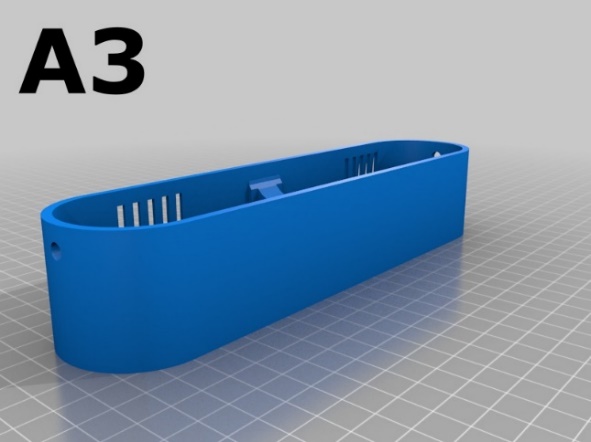
## 9.1. Estructura del Tanque

El diseño del tanque se basa en piezas previamente diseñadas de un prototipo existente. Para la impresión de estas piezas se utilizó el software Cura, optimizando parámetros para asegurar precisión y robustez. A continuación, se describe cada componente estructural en detalle:

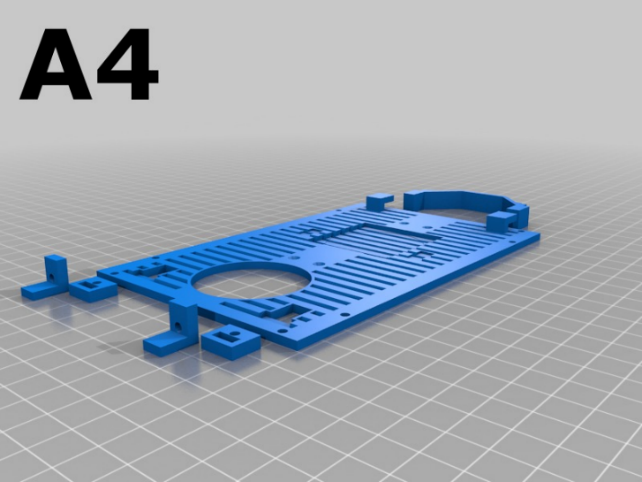
* Cuerpo (A1): La pieza principal del tanque, con dimensiones de 207 mm de largo, 50 mm de alto y 106 mm de ancho, sirve como la base que soporta todos los demás componentes. Esta pieza fue impresa utilizando Cura, que permitió ajustar la densidad de relleno y la orientación de impresión para garantizar robustez y precisión. Durante la validación del prototipo, se comprobó que el cuerpo podía albergar todos los circuitos electrónicos, motores y baterías, con suficiente espacio y accesibilidad para el cableado. Se añadieron refuerzos internos para soportar el peso de los componentes y resistir el desgaste durante el uso prolongado.



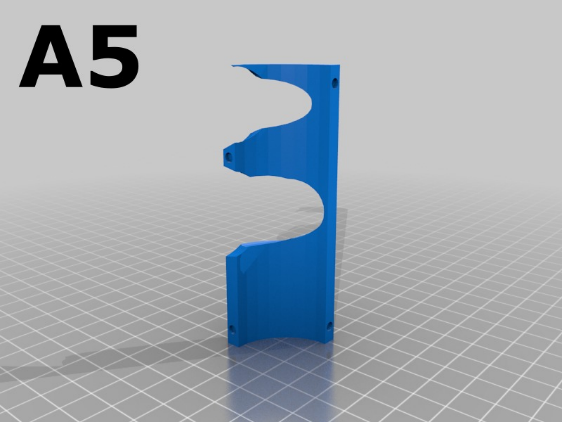
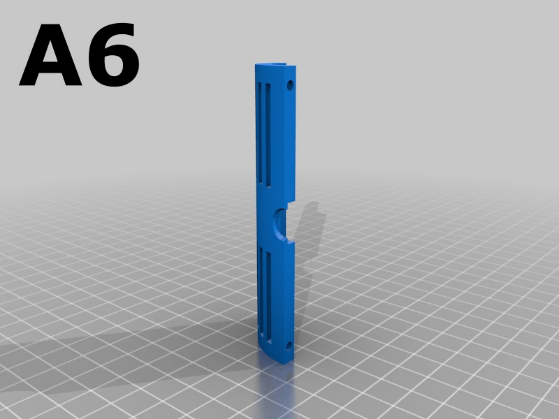
* Costado izquierdo (A2) y Costado Derecho (A3): Ambas piezas laterales, cada una de 207 mm de largo, 50 mm de alto y 40 mm de ancho, se diseñaron para integrarse perfectamente con el cuerpo del tanque. En el prototipo, estas piezas incluían ranuras y orificios específicos para la fijación de los motores DC y sus engranajes. Utilizando Cura, se optimizó la orientación de impresión y la densidad del relleno para mejorar la calidad de las piezas y reducir el uso de material de soporte. Además, se realizaron pruebas de resistencia para asegurar que las piezas laterales pudieran soportar la tensión y el estrés durante el funcionamiento del tanque.

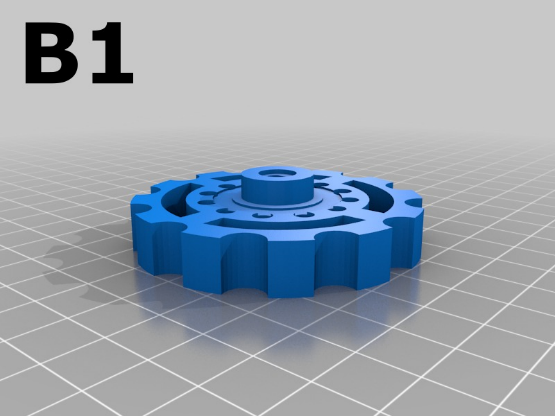
* Tapa superior y soportes (A4): Esta pieza, de 209 mm de largo, 96 mm de alto y 11 mm de ancho, cubre la parte superior del tanque y proporciona una plataforma estable para montar el cañón y los servomotores. Con Cura, se lograron superficies lisas y precisas, esenciales para el montaje de los componentes de los servomotores, lo que permite un control más preciso de estos elementos.



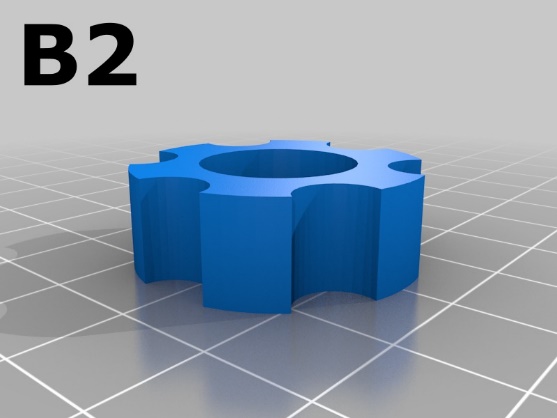
* Tapa Frontal (A5) y Tapa Trasera (A6): Estas tapas, de 89.65 mm de largo, 24.99 mm de alto y 24.99 mm de ancho, protegen los componentes internos y permiten el acceso a los puertos de carga y conexión. El prototipo ayudó a definir las ubicaciones óptimas para las aberturas de ventilación y los puntos de conexión. Cura permitió ajustar la precisión de los detalles y asegurar un ajuste perfecto entre las piezas, lo que garantiza una protección eficiente de los componentes internos y facilita el mantenimiento y las actualizaciones.

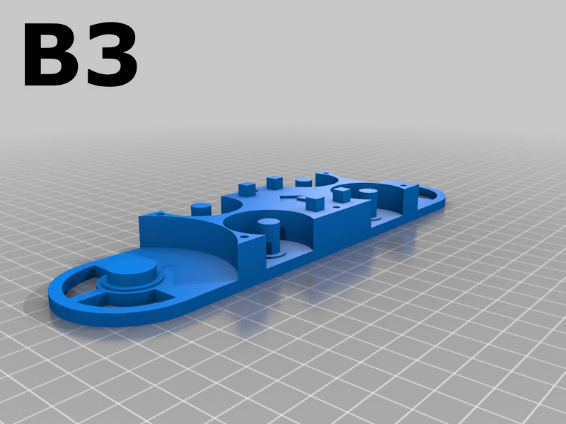
* Engrane de motor (B1): Estos engranajes tienen dimensiones de 59.46 mm de largo, 59.46 mm de alto y 15 mm de ancho. Son componentes esenciales para transmitir la potencia de los motores DC al sistema de tracción del tanque. Su diseño robusto garantiza un acoplamiento eficiente y duradero con los engranajes pequeños y los eslabones de la oruga, contribuyendo así al movimiento fluido y controlado del tanque, para este proyecto se utilizaron 4 piezas de engrane de motor.



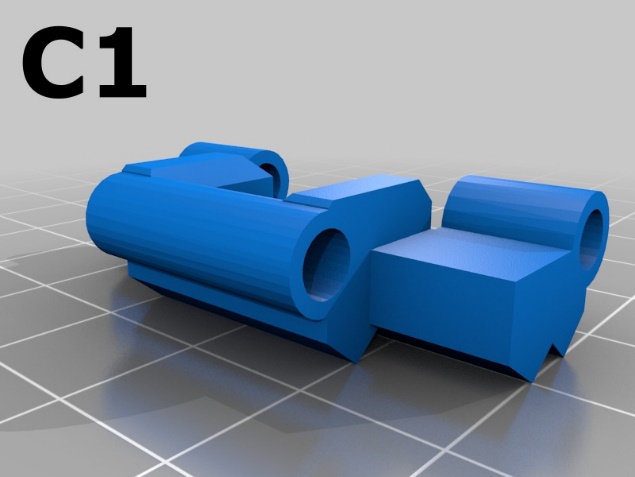
* Engrane pequeño (B2): Estos engranajes miden 28 mm de largo, 30 mm de alto y 10 mm de ancho. Son responsables de transmitir el movimiento desde los engranajes de motor a los eslabones de la oruga, permitiendo el desplazamiento suave y controlado del tanque. Su tamaño compacto y diseño preciso aseguran una operación sin problemas del sistema de tracción. Para este proyecto se utilizaron 8 piezas de engrane pequeño.



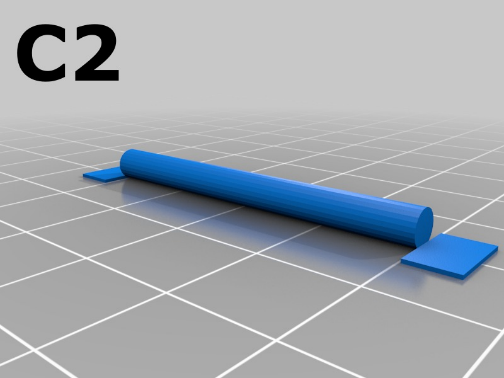
* Soporte de engranes: Con dimensiones de 207 mm de largo, 50 mm de alto y 21 mm de ancho, estos soportes proporcionan una base estable y resistente para los engranajes de motor y los engranajes pequeños. Su diseño ergonómico y robusto garantiza que los engranajes estén correctamente alineados y protegidos dentro del cuerpo del tanque, asegurando un funcionamiento suave y confiable del sistema de tracción. Para este proyecto se utilizaron 2 piezas de soporte de engranes.



* Eslabón de oruga: Estos eslabones tienen medidas de 22.3 mm de largo, 40 mm de alto y 8 mm de ancho. Son los componentes principales del sistema de tracción del tanque, permitiendo que se mueva de manera fluida y estable sobre diferentes superficies. Su diseño resistente y duradero garantiza una tracción óptima y una larga vida útil del tanque, incluso en condiciones adversas. Para este proyecto se utilizaron 64 piezas de eslabón de oruga.



* Pin de unión oruga: Con dimensiones de 50 mm de largo, 3 mm de alto y 8 mm de ancho, estos pines son los encargados de conectar los eslabones de la oruga entre sí, formando una cadena continua que impulsa el movimiento del tanque. Su diseño simple y funcional permite un ensamblaje fácil y seguro de los eslabones, asegurando una tracción uniforme y eficiente en todas las condiciones. Para este proyecto se utilizaron 64 piezas de pin de unión de oruga.



## 9.2. Integración de Componentes Electrónicos

La integración de los componentes electrónicos en el diseño del tanque es un proceso importante que implica la disposición cuidadosa y la interconexión de cada componente para garantizar un funcionamiento eficiente y confiable del sistema. En este proyecto, se emplean varios componentes electrónicos, entre ellos un Arduino Mega, un ESP32, motores DC, servomotores y un módulo L298P driver de motor Arduino UNO R3 Shield, así como el cableado necesario para establecer conexiones seguras y estables entre ellos. A continuación, se detalla el proceso de integración de estos componentes:

## 9.2.1. Posicionamiento y Fijación de los Componentes

El primer paso en la integración de los componentes electrónicos es determinar la ubicación adecuada de cada uno dentro del cuerpo del tanque. Se reservan áreas específicas para montar el Arduino Mega y el ESP32, asegurando que estén accesibles para realizar conexiones y cargar programas. Los motores DC se colocan en los costados del tanque, mientras que los servomotores se montan en el cañón para controlar su elevación. El módulo L298P se sitúa en una posición estratégica para facilitar la conexión con los motores DC.

## 9.2.2. Conexión de los Componentes al Arduino Mega y al ESP32

Una vez posicionados, se procede a conectar los componentes electrónicos al Arduino Mega y al ESP32 según el esquema de conexiones previamente definido. Los motores DC se conectan al módulo L298P, que a su vez se conecta al Arduino Mega para recibir señales de control. Los servomotores se conectan directamente al Arduino Mega para recibir señales de control PWM y así permitir su movimiento preciso. El ESP32 se conecta al Arduino Mega mediante un cableado serial para facilitar la comunicación entre ambos y permitir el control remoto del tanque a través de la aplicación Unity.

## 9.2.3. Cableado y Enrutamiento de Cables

Con todos los componentes conectados, se realiza el cableado necesario para establecer conexiones eléctricas entre ellos. Se emplea cableado adecuado para cada tipo de conexión, asegurando un flujo de corriente seguro y estable. Para las pruebas del tanque, se utilizaron jumpers y después para el armado final se evitó el uso de jumpers y se optó por un enrutamiento cuidadoso de los cables para evitar interferencias y posibles cortocircuitos. Los cables se fijan en su lugar con un amarrado colocando un poco de cinta alrededor y después se suelda para evitar movimientos no deseados que podrían afectar el funcionamiento del tanque.

## 9.3. Ensamblaje de Piezas y Componentes Electrónicos

El ensamblaje de las piezas y componentes electrónicos del tanque es una etapa importante que combina todas las partes físicas y electrónicas en un sistema cohesivo y funcional. Este proceso requiere precisión y atención al detalle para garantizar que todas las piezas encajen correctamente y que los componentes electrónicos estén bien conectados y operativos.

El primer paso en el ensamblaje es la preparación de todas las piezas impresas en 3D. Como se mencionó anteriormente, después de imprimir las piezas en 3D se eliminó cualquier exceso de material de soporte o imperfección resultante del proceso de impresión. También, las piezas fueron lijadas para suavizar las superficies y asegurar que las piezas encajen perfectamente. Las piezas como el cuerpo principal, los costados, las tapas, los eslabones de la oruga y los engranajes deben estar limpias y listas para el ensamblaje.

Con las piezas ya preparadas, se comienza el ensamblaje de la estructura del tanque. El cuerpo principal del tanque, junto con los costados izquierdo y derecho, se ensambla primero, utilizando tornillos para asegurar una unión firme y estable. A continuación, se montan la tapa superior y los soportes, seguidos por las tapas frontal y trasera, cerrando así la estructura principal del tanque. Durante este proceso, se asegura que todos los orificios y espacios para los componentes electrónicos estén alineados correctamente.

Una vez que la estructura está completa, se procede a la instalación de los componentes electrónicos. El Arduino Mega y el ESP32 se montan en el interior del cuerpo del tanque, en posiciones accesibles y seguras. Se utilizan tornillos y soportes específicos para fijar estos componentes en su lugar, asegurando que estén firmemente sujetos.

Los motores DC se instalan en los costados del tanque, conectándolos a los soportes de engranes y asegurándolos con tornillos. El módulo L298P, responsable de controlar estos motores, se fija en una ubicación central que facilite el cableado y las conexiones. Los servomotores se montan en el cañón del tanque, uno para controlar la elevación y otro para el movimiento, asegurando que estén alineados y bien sujetos.

Con los componentes electrónicos en su lugar, se procede a la conexión de todos los cables necesarios. Como se mencionó anteriormente, se utilizan cables adecuados para cada tipo de conexión, asegurando un flujo de corriente seguro y estable. Se evita el uso de jumpers, optando por un cableado directo y seguro. Los cables se enrutan cuidadosamente a través del tanque, utilizando abrazaderas o bridas para mantenerlos ordenados y prevenir enredos o interferencias. Las conexiones entre el Arduino Mega, el ESP32, los motores DC, los servomotores y el módulo L298P se verifican para asegurar que todas sean firmes y correctas.

Los eslabones de la oruga se ensamblan utilizando los pines de unión, formando una cadena continua que rodea las ruedas del tanque. Este ensamblaje debe ser preciso para asegurar que las orugas se muevan suavemente y sin obstáculos. Una vez ensambladas, las orugas se instalan en su lugar, conectando los engranajes de motor y los engranajes pequeños para permitir el movimiento controlado del tanque.

Con todas las piezas y componentes ensamblados, se realizan pruebas exhaustivas para verificar el funcionamiento del tanque. Se prueban los movimientos del tanque, el control del cañón y el mecanismo de disparo para asegurar que respondan adecuadamente a los comandos de la aplicación Unity. Cualquier ajuste necesario se realiza en esta etapa para garantizar el rendimiento óptimo del tanque. Se revisan todas las conexiones eléctricas para asegurar que no haya cortocircuitos o conexiones sueltas.

# Capítulo 10 Configuración de la Red y Conectividad

La configuración de la red y la conectividad son aspectos esenciales para el funcionamiento óptimo del tanque controlado a distancia. Además de establecer la red WiFi para el tanque, se utilizó una aplicación llamada Fing para visualizar los dispositivos conectados a la red, lo que facilita el monitoreo y la gestión de los dispositivos conectados.

## 10.1. Conexión del Módem

El primer paso en la configuración de la red es conectar correctamente el módem. Para este proyecto, se está utilizando un módem estándar como el LINKSYS EA4500, el cual debe estar bien conectado a la fuente de energía y al proveedor de servicios de Internet (ISP). En primer lugar, se asegura de que el módem esté conectado a una toma de corriente adecuada y que esté encendido. Los módems suelen tener indicadores LED que muestran su estado operativo; por lo tanto, es importante verificar que todas las luces necesarias estén encendidas. Además, se utilizó el cable de red proporcionado por su ISP para conectar el módem a la toma de Internet. Este cable generalmente se conecta al puerto WAN (Wide Area Network) del módem. Una vez conectados los cables, se debe verificar nuevamente los indicadores LED del módem para asegurarse de que está conectado a Internet. La luz de "Internet" debe estar fija o parpadeando, lo que indica una conexión activa y estable.

## 10.2. Creación de la Red

Con el módem correctamente conectado y funcionando, el siguiente paso es la creación de la red WiFi que será utilizada por el tanque. Para ello, se utilizó una computadora, para conectarse al módem mediante un cable Ethernet. Después se abrió un navegador web y se ingresó la dirección IP del módem. Una vez ingresada la dirección, se debe ingresar las credenciales de administrador del módem. Estas credenciales suelen estar impresas en una etiqueta en el módem o proporcionadas por el ISP. Después de iniciar sesión, se navega a la sección de configuración de la red inalámbrica del módem. Aquí se podrá establecer el nombre (SSID) de la red WiFi y configurar otros parámetros básicos necesarios para su funcionamiento. Además, es fundamental seleccionar un tipo de seguridad para la red, como WPA2, y después se establece una contraseña segura para evitar accesos no autorizados.

## 10.3. Cambio de Nombre y Contraseña de la Red

Para asegurar la red y hacerla fácilmente identificable, se debe cambiar el nombre (SSID) y la contraseña predeterminados que vienen configurados en el módem. En la sección de configuración inalámbrica del módem, en el campo donde se ingresa el SSID, se cambia el nombre a uno único para identificar la red de manera clara y sencilla. Esto no solo facilita la identificación de la red entre otras disponibles, sino que también puede incrementar la seguridad. A continuación, se debe configurar una contraseña segura para la red. Una vez realizados estos cambios, se debe de guardar todas las configuraciones en el módem. Después se debe reiniciar el módem para aplicar estos cambios.

## 10.4. Conexión del Tanque a la Red WiFi

Con la red configurada y asegurada, el siguiente paso es conectar el tanque a la red WiFi utilizando el ESP32. Primero, se debe asegurar de que el ESP32 esté programado correctamente para conectarse a la red WiFi. En el código del ESP32, se debe ingresar el nuevo SSID y la contraseña de la red que se configuró anteriormente. Este paso es importante para establecer una comunicación efectiva entre el tanque y la red. Una vez programado el ESP32, encienda el tanque y permita que el ESP32 se conecte a la red. Se verifica la conexión imprimiendo en el monitor serial del IDE de Arduino. Este proceso implica que el ESP32 intentará conectarse a la red WiFi utilizando las credenciales proporcionadas y, una vez conectado, confirmará la conexión mostrando la dirección IP asignada en la red local.

# Capítulo 11 Diseño del Sistema de Control

## 11.1. Implementación de la Lógica de Control del Tanque

La lógica de control del tanque se implementó para permitir movimientos precisos y el control del cañón. Aquí se detalla el código para el control del movimiento del tanque en Unity:  
using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.UI;

public class Movimiento : MonoBehaviour

{

public DeviceDiscover deviceDiscover;

private Vector3 lastMousePosition;

public Button ButtonStop, ButtonW, ButtonA, ButtonS, ButtonD, ButtonQ, ButtonE, ButtonU, ButtonJ, Button0;

void Start()

{

ButtonA.onClick.AddListener(() => deviceDiscover.SendData("A"));

ButtonD.onClick.AddListener(() => deviceDiscover.SendData("D"));

ButtonW.onClick.AddListener(() => deviceDiscover.SendData("W"));

ButtonQ.onClick.AddListener(() => deviceDiscover.SendData("Q"));

ButtonE.onClick.AddListener(() => deviceDiscover.SendData("E"));

ButtonS.onClick.AddListener(() => deviceDiscover.SendData("S"));

ButtonStop.onClick.AddListener(() => deviceDiscover.SendData("P"));

ButtonU.onClick.AddListener(() => deviceDiscover.SendData("U"));

ButtonJ.onClick.AddListener(() => deviceDiscover.SendData("J"));

Button0.onClick.AddListener(() => deviceDiscover.SendData("0"));

}

}

Este script se encarga de asignar los botones de la interfaz de usuario (UI) en Unity para enviar comandos al ESP32. Cada botón envía un comando específico que controla el movimiento del tanque y la elevación del cañón.

# Capítulo 12 Archivos Adjuntos

## 12.1. Cámara

El siguiente código se define una clase Camara en Unity que se encarga de mostrar la imagen de la cámara de la computadora en un objeto Canvas, dado que no se pudo conectar una cámara externa al sistema. En el método Start, se inicializan los componentes RawImage y WebCamTexture, se ajusta el tamaño y la escala del RawImage para que coincida con la pantalla, y se inicia la transmisión de la cámara web. También se define el tamaño de zoom y se crea una máscara circular mediante el método CreateCircleMaskTexture. En el método Update, se detecta si se ha hecho clic con el botón derecho del mouse y se llama al método ToggleZoom, que alterna el estado del zoom, ajustando el tamaño de RawImage en consecuencia. La máscara circular se aplica en el método OnGUI cuando el zoom está activado. El método CreateCircleMaskTexture genera una textura circular transparente en el interior y negra en el exterior, que se usa como máscara. A continuación se muestra el código:  
using UnityEngine;

using UnityEngine.UI;

public class Camara : MonoBehaviour

{

RawImage rawImage;

WebCamTexture webCamTexture;

// Variable para controlar si el zoom está activado o desactivado

bool isZoomed = false;

// Tamaño de zoom

Vector2 zoomedSize;

// Textura para la máscara circular

Texture2D circleMaskTexture;

void Start()

{

rawImage = GetComponent<RawImage>();

webCamTexture = new WebCamTexture();

rawImage.rectTransform.sizeDelta = new Vector2(Screen.width, Screen.height);

rawImage.rectTransform.localScale = new Vector3(-1, 1, 1);

rawImage.texture = webCamTexture;

webCamTexture.Play();

// Inicializa el tamaño de zoom

zoomedSize = new Vector2(Screen.width \* 1.5f, Screen.height \* 1.5f);

// Crea una textura circular para recortar la imagen

circleMaskTexture = CreateCircleMaskTexture(Screen.width, Screen.height);

}

void Update()

{

// Verificar si se ha hecho clic con el botón derecho del mouse

if (Input.GetMouseButtonDown(1))

{

ToggleZoom();

}

}

void ToggleZoom()

{

// Cambiar el estado del zoom

isZoomed = !isZoomed;

// Aplicar zoom si está activado

if (isZoomed)

{

// Modificar el tamaño del RawImage para simular el zoom

rawImage.rectTransform.sizeDelta = zoomedSize;

}

else

{

// Restaurar el tamaño original

rawImage.rectTransform.sizeDelta = new Vector2(Screen.width, Screen.height);

}

}

// Función para crear una textura circular para recortar la imagen

Texture2D CreateCircleMaskTexture(int width, int height)

{

Texture2D texture = new Texture2D(width, height);

Color[] colors = new Color[width \* height];

float centerX = width / 2f;

float centerY = height / 2f;

float radius = Mathf.Min(centerX, centerY);

for (int y = 0; y < height; y++)

{

for (int x = 0; x < width; x++)

{

float distance = Mathf.Sqrt((x - centerX) \* (x - centerX) + (y - centerY) \* (y - centerY));

if (distance <= radius)

{

colors[y \* width + x] = Color.clear; // Hace transparente el círculo interior

}

else

{

colors[y \* width + x] = Color.black; // Hace negro el área exterior

}

}

}

texture.SetPixels(colors);

texture.Apply();

return texture;

}

void OnGUI()

{

// Dibuja la máscara circular solo cuando el zoom está activado

if (isZoomed)

{

GUI.DrawTexture(new Rect(0, 0, Screen.width, Screen.height), circleMaskTexture, ScaleMode.StretchToFill);

}

}

}

## 12.2. Archivo Device

El siguiente código es el archivo DeviceDiscover en Unity, el propósito de este código es buscar y manejar la conexión del módulo WiFi ESP32 en la red a la que está conectada la aplicación. Está clase utiliza una instancia de conectionArduino para gestionar la conexión TCP con el ESP32. Al iniciar, el método Start lo que hace es obtener la dirección IP local del dispositivo y verificar que se encuentre el ESP32 en una dirección IP específica mediante una solicitud HTTP a la ruta definida en esp8266Path. El método GetLocalIPAddress lo que hace es obtener la dirección IP local del dispositivo escaneando todas las interfaces de red activas. El método CheckDevice envía una solicitud HTTP a la IP del ESP32 y, si se detecta, establece una conexión TCP. Si no se puede conectar, reinicia el proceso. El método OnApplicationQuit cierra la conexión TCP al salir de la aplicación. Además, el método SendData permite enviar datos al ESP32 si la conexión está establecida. Este código facilita la interacción entre Unity y el módulo ESP32 para controlar dispositivos de forma remota a través de una red WiFi. A continuación, se muestra el código completo:  
using System.Collections;

using UnityEngine;

using System.Net.Sockets;

using System.Net.NetworkInformation;

using UnityEngine.Networking;

using UnityEngine.UI;

/// <summary>

/// Clase DeviceDiscover para descubrir y manejar la conexión con el módulo WiFi ESP8266.

/// </summary>

public class DeviceDiscover : MonoBehaviour

{

// Instancia de la clase conectionArduino para manejar la conexión con el módulo WiFi ESP8266

private conectionArduino conectionArduino;

// Ruta para verificar la presencia del módulo ESP8266

public string esp8266Path = "/check-esp8266";

// Dirección IP del módulo ESP8266

private string ipAdress = "192.168.1.148";

/// <summary>

/// Método Start que se llama antes de la primera actualización del frame.

/// </summary>

void Start()

{

string ipAddress = GetLocalIPAddress();

Debug.Log("Direccion IP local: "+ipAddress);

StartCoroutine(CheckDevice(ipAdress));

//textoIp = GameObject.Find("Text").GetComponent<TextMesh>();

}

/// <summary>

/// Método GetLocalIPAddress para obtener la dirección IP local del dispositivo.

/// </summary>

/// <returns>La dirección IP local del dispositivo.</returns>

string GetLocalIPAddress()

{

string ipAddress = "";

NetworkInterface[] interfaces = NetworkInterface.GetAllNetworkInterfaces();

foreach (NetworkInterface intf in interfaces)

{

if (intf.OperationalStatus == OperationalStatus.Up &&

intf.NetworkInterfaceType == NetworkInterfaceType.Wireless80211)

{

IPInterfaceProperties ipProps = intf.GetIPProperties();

foreach (UnicastIPAddressInformation addr in ipProps.UnicastAddresses)

{

if (addr.Address.AddressFamily == AddressFamily.InterNetwork)

{

ipAddress = addr.Address.ToString();

break;

}

}

}

}

return ipAddress;

}

/// <summary>

/// Método CheckDevice para verificar la presencia del módulo ESP8266 en la dirección IP especificada.

/// </summary>

/// <param name="ipAddress">La dirección IP donde se verificará la presencia del módulo ESP8266.</param>

IEnumerator CheckDevice(string ipAddress)

{

Debug.Log(ipAddress);

string url = "http://" + ipAddress + esp8266Path;

using (UnityWebRequest www = UnityWebRequest.Get(url))

{

www.certificateHandler = new CustomCertificateHandler();

yield return www.SendWebRequest();

if (www.result == UnityWebRequest.Result.Success)

{

Debug.Log("Encontrado ESP8266 en: " + ipAddress);

conectionArduino = new conectionArduino();

conectionArduino.Connect(ipAddress, 80);

Debug.Log("Conectado a " + ipAddress);

// imprimir la Ip en el texto

//textoIp.text = "IP: "+ipAddress;

}

else

{

Debug.Log("No se pudo conectar a " + ipAddress + ": " + www.error);

Start();

}

}

}

/// <summary>

/// Método OnApplicationQuit para manejar el cierre de la aplicación.

/// </summary>

void OnApplicationQuit()

{

conectionArduino.CloseConnection();

}

/// <summary>

/// Método SendData para enviar datos al módulo WiFi ESP8266.

/// </summary>

/// <param name="s">Los datos que se enviarán al módulo WiFi ESP8266.</param>

public void SendData(string s)

{

if (conectionArduino != null)

{

conectionArduino.SendData(s);

Debug.Log(s);

//textoIp.text = s;

}

}

}

## 12.3. El Archivo Reticula

El siguiente archivo contiene la implementación de la clase Reticula en Unity, la cual está diseñada para gestionar la visualización de una retícula en la imagen del juego. Esta retícula está compuesta por líneas horizontales y verticales representadas por imágenes. En el método Start, se inicializa la visualización de la retícula según el estado actual de la bandera ban, que controla su visibilidad. Durante la ejecución del juego, el método Update se encarga de verificar si se ha presionado el botón derecho del ratón para cambiar el estado de la bandera ban y actualizar la visualización de la retícula en consecuencia. Finalmente, el método ShowReticle se utiliza para controlar directamente la visualización de la retícula al modificar la propiedad enabled de cada imagen de la retícula, basándose en el parámetro show, que determina si la retícula debe mostrarse o no. A continuación, se muestra el código completo:

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.UI;

/// <summary>

/// Clase Reticula para manejar la visualización de una retícula en la interfaz de usuario.

/// </summary>

public class Reticula : MonoBehaviour

{

// Imágenes que representan las líneas horizontales de la retícula

public Image h0;

public Image h1;

public Image h2;

// Imágenes que representan las líneas verticales de la retícula

public Image v0;

public Image v1;

public Image v2;

public Image v3;

public Image v4;

public Image v5;

// Bandera para controlar la visualización de la retícula

private bool ban = false;

/// <summary>

/// Método Start que se llama antes de la primera actualización del frame.

/// </summary>

void Start()

{

ShowReticle(ban);

}

/// <summary>

/// Método Update que se llama una vez por frame.

/// </summary>

void Update()

{

// Si se presiona el botón derecho del ratón

if (Input.GetMouseButtonDown(1))

{

// Cambia el estado de la bandera

if (ban == false)

{

ban = true;

}

else

{

ban = false;

}

// Actualiza la visualización de la retícula

ShowReticle(ban);

}

}

/// <summary>

/// Método ShowReticle para controlar la visualización de la retícula.

/// </summary>

/// <param name="show">Si es verdadero, la retícula se muestra. Si es falso, la retícula se oculta.</param>

void ShowReticle(bool show)

{

h0.enabled = show;

h1.enabled = show;

h2.enabled = show;

v0.enabled = show;

v1.enabled = show;

v2.enabled = show;

v3.enabled = show;

v4.enabled = show;

v5.enabled = show;

}

}

# Capítulo 13 Pruebas y Depuración Final del Sistema

La fase de validación y pruebas del sistema es importante para asegurar que todos los componentes del tanque funcionen de manera efectiva y conforme a los requisitos especificados. En esta se abarca desde la comprobación del funcionamiento general del tanque hasta la optimización final de la conectividad y el rendimiento. A continuación, se detalla cada una de estas pruebas y ajustes.

## 13.1. Pruebas de Operación Completa

Las pruebas de operación completa son esenciales para verificar que el tanque funcione correctamente en todos sus aspectos. En esta etapa, se realizaron diversas pruebas para asegurarse de que el tanque se mueva, dispare y se controle según lo previsto. Primero, se verifico la movilidad del tanque, comprobando que los cuatro motores DC funcionen adecuadamente, permitiendo que el tanque avance, retroceda y gire según las instrucciones enviadas desde Unity. Además de la movilidad, se probó el sistema de disparo. Los servomotores encargados de levantar y mover el cañón se verificaron para garantizar que puedan posicionar el cañón correctamente. Los motores de drones que lanzan las balas NERF también se probaron para asegurar que disparen con la fuerza y precisión adecuadas. Se realizaron varios ciclos de disparo para verificar el sistema de lanzamiento.

## 13.2. Verificación de la Conectividad WiFi

La verificación de la conectividad WiFi es fundamental para asegurar una comunicación fluida entre el tanque y el sistema de control en Unity. En esta etapa, se comprobó que el ESP32 se conectó correctamente a la red WiFi y mantuvo una conexión estable. Con la aplicación Fing se revisó que dispositivos estaban conectados a la red WiFi, y se comprobó que el tanque estaba conectado. Se realizarón pruebas de latencia y alcance para asegurarse de que el tanque responda rápidamente a los comandos desde Unity. Esto incluyó mover el tanque a diferentes distancias del módem y en distintos entornos para evaluar la estabilidad de la conexión.

## 13.3. Ajustes Finales

Una vez que el tanque ha pasado todas las pruebas, se verificó que el tanque es capaz de realizar todas las tareas especificadas de manera eficiente, se realizó una revisión completa del hardware y el software, asegurando que todas las partes funcionen juntas de manera cohesiva. Por lo cual, se considera listo para su uso en aplicaciones prácticas y demostraciones.