Universidad Zaragoza
1542

Trabajo Fin de Grado

Título del trabajo: Diseño y Fabricación de un Vehículo Teledirigido Controlado por Raspberry Pi Pico W con Capacidades de Sensado y Visualización Remota

English title: Design and Fabrication of a Raspberry Pi Pico W-Controlled Remote Vehicle with Sensing and Remote Visualization Capabilities

Autor:

Miguel María Sancho-Tello Larraz

Director/es

Carlos Orrite Uruñuela

Ingeniería de Tecnologías Industriales

ESCUELA DE INGENIERÍA

2024

RESUMEN

Este proyecto consiste en la creación de un coche teledirigido con capacidades de sensado y visualización remota, controlado a través de una interfaz web. Los objetivos principales son desarrollar una experiencia de usuario intuitiva para el control del coche, mejorar su capacidad para navegar de forma segura detectando y evitando obstáculos, y proporcionar una cámara para visualización remota que permita ver el entorno desde la perspectiva del coche. Además, el proyecto tiene como objetivo servir de herramienta educativa en la asignatura de Sistemas Electrónicos Digitales, permitiendo a los estudiantes explorar conceptos de electrónica y programación de manera práctica.

Para el diseño del coche, se considera cuidadosamente la selección de cada componente, buscando un equilibrio entre rendimiento y costo. El chasis está diseñado para organizar los componentes de forma eficiente, facilitando tanto la estabilidad del coche como posibles futuras modificaciones. Este enfoque permite construir un coche que no solo funciona bien, sino que también puede ser mejorado y adaptado con facilidad.

El software desarrollado permite una comunicación fluida entre el usuario y el coche, garantizando que los comandos se ejecuten de manera efectiva y que el usuario reciba retroalimentación en tiempo real sobre el estado del coche. La incorporación de una cámara para visualización remota mejora la capacidad de control a distancia y la interacción con el vehículo en diferentes escenarios, enriqueciendo la experiencia de aprendizaje en la asignatura.

A lo largo del desarrollo, se abordan diversos temas que abarcan desde el diseño y la elección de componentes hasta la programación y la integración de sistemas. Esto permite obtener una comprensión integral de cómo se construye un proyecto de este tipo desde cero. Aunque el proyecto concluye en este punto, la estructura flexible y modular del diseño abre la puerta a futuras mejoras y nuevas aplicaciones, demostrando que las posibilidades son tan amplias como la creatividad y la imaginación lo permitan.

ÍNDICE

1. Introducción 6

1.1. Contexto 6

1.2. Estado del arte 7

1.2.1. Principales productos en el mercado 7

1.2.2. Principales problemáticas y áreas de mejora 9

1.3. Objetivos y planificación 10

1.3.1. Objetivos del proyecto 10

1.3.2. Planificación modular 11

2. Diseño del sistema 13

2.1. Especificaciones del coche teledirigido 13

2.2. Arquitectura del sistema 13

2.3. Elección de componenetes 14

3. Desarrollo del hardware 16

3.1. Construcción del chasis 16

3.2. Integración de motores 16

3.3. Integración de sensores 17

3.4. Integración de la cámara esp32 18

4. Desarrollo del software 20

4.1. Programación en micropython 21

4.2. Interfaz web en html 22

4.3. Comunicación entre hardware y software 23

4.4. Configuración de cámara esp23 24

5. Implementación y pruebas 25

5.1. Ensamblaje del coche 25

5.2. Pruebas funcionales 25

5.3. Funcionamiento del coche 26

6. Resultados y discusión 28

6.1. Logros alcanzados 28
6.2. Limitaciones y mejoras futuras 30
7. Conclusiones 30
7.1. Recapitulación 30
7.2. Aprendizajes 31
8. Referencias 33
8.1. Bibliografía 33
8.3. Referencias de código 35
Anexo I: Control PWM en Micropython 36
Anexo II: Sockets en la comunicación entre software y hardware 40
Anexo III: Desglose de componentes 42
Anexo IV: Diseño del chasis en autocad 47
Anexo V: Detalles técnicos de la programación en micropython 48
Anexo VI: Configuración técnica de la cámara web en esp32-cam 50

1. INTRODUCCIÓN

1.1.CONTEXTO

En el mundo actual, la convergencia entre la electrónica digital, la programación y las tecnologías de comunicación ha impulsado la creación de dispositivos inteligentes y autónomos que facilitan una amplia gama de aplicaciones, desde la automatización del hogar hasta la vigilancia remota y la robótica educativa. La evolución hacia la "Internet de las Cosas" (IoT) ha fomentado el desarrollo de sistemas embebidos que pueden interactuar con su entorno a través de sensores y actuar mediante actuadores, todo ello controlado de forma remota o autónoma mediante algoritmos sofisticados.

Los coches teledirigidos con capacidades de sensado y visualización remota son un claro ejemplo de esta tendencia tecnológica. Estos dispositivos no solo permiten un control remoto preciso, sino que también pueden recopilar datos del entorno y transmitir información visual en tiempo real. Este tipo de soluciones se vuelve crucial en escenarios donde la presencia humana directa es limitada o riesgosa, como en la exploración de áreas de difícil acceso, la inspección de infraestructuras críticas o en aplicaciones de rescate. Además, desde el punto de vista educativo y experimental, estos proyectos proporcionan un terreno fértil para la práctica y la innovación en campos como la robótica, la programación de sistemas embebidos, la inteligencia artificial y la visión por computadora.

En el marco de la asignatura de Sistemas Electrónicos Digitales, este proyecto no solo presenta una oportunidad de aplicar conceptos teóricos en un contexto práctico, sino que también permite a los estudiantes enfrentar desafíos reales de diseño e implementación. A través de la construcción de un coche teledirigido con capacidades de sensado y visualización remota, se integran conocimientos adquiridos sobre microcontroladores, sensores, actuadores, protocolos de comunicación y programación en entornos de desarrollo embebido, como MicroPython. Además, el desarrollo de una interfaz de control en HTML y la programación de la comunicación entre hardware y software permite abordar aspectos críticos de la interacción humano-máquina y de la creación de interfaces intuitivas y funcionales.

Este proyecto aporta un valor significativo a la asignatura al ofrecer un puente tangible entre teoría y práctica, motivando a los estudiantes a resolver problemas técnicos, optimizar recursos

y colaborar en un entorno interdisciplinario. La motivación para llevar a cabo este trabajo radica en la necesidad de aplicar los conocimientos de la asignatura en un proyecto integral que abarque desde el diseño del sistema hasta la implementación y prueba del producto final. Asimismo, permite explorar los desafíos de la integración de sistemas y fomentar la creatividad en la resolución de problemas, preparando así a los estudiantes para futuros desafíos en el campo de la electrónica digital y la ingeniería de sistemas embebidos.

Este enfoque práctico y aplicado no solo refuerza los aprendizajes teóricos, sino que también estimula la innovación y la investigación, fomentando un pensamiento crítico y una actitud proactiva hacia la solución de problemas reales, lo cual es esencial en la formación de ingenieros y tecnólogos en el contexto de un mundo cada vez más digital e interconectado.

1.2.ESTADO DEL ARTE

En el mercado actual, los coches teledirigidos con capacidades de sensado y visualización remota se han diversificado considerablemente, abarcando desde juguetes para el público general hasta herramientas avanzadas utilizadas en investigación, vigilancia y operaciones de rescate. Estos dispositivos varían en complejidad, funcionalidad, y precio, dependiendo de sus características técnicas, el tipo de sensores integrados, la calidad de la transmisión de video en tiempo real, y las capacidades de control remoto.

1.2.1. Principales Productos en el Mercado

1. Juguetes y Modelos para Aficionados:
    - Los coches teledirigidos básicos, destinados a niños y aficionados, suelen ofrecer funciones limitadas de control remoto y, en algunos casos, incluyen cámaras básicas que permiten la transmisión de video a un dispositivo móvil. Marcas como Syma o DROCON ofrecen modelos que priorizan la facilidad de uso y la experiencia de juego, con precios accesibles que los hacen populares entre el público general.
    - Problemas y Áreas de Mejora: Estos modelos, aunque accesibles, suelen tener limitaciones significativas en cuanto a la calidad de imagen, la distancia de

transmisión y la capacidad de maniobra. Además, la durabilidad y la vida útil de la batería son cuestiones comunes que requieren mejoras.

2. Modelos Semi-Profesionales para Uso Educativo y de Investigación:
    - Existen modelos más avanzados que son utilizados en entornos educativos y de investigación, como los ofrecidos por DFRobot o SunFounder, los cuales vienen equipados con microcontroladores programables (como Arduino o Raspberry Pi) y diversos sensores (ultrasonido, infrarrojos, cámaras de alta definición, etc.). Estos dispositivos permiten a los usuarios personalizar el coche según sus necesidades y experimentar con distintos algoritmos de control y navegación.
    - Problemas y Áreas de Mejora: Aunque estos modelos son más versátiles y ofrecen un nivel considerable de personalización, requieren conocimientos técnicos para su configuración y programación. Además, la interfaz de usuario para su control remoto no siempre es intuitiva, y la integración de diferentes sensores puede resultar problemática en términos de compatibilidad y gestión de energía.
3. Modelos Profesionales para Aplicaciones Especializadas:
    - Los coches teledirigidos de nivel profesional, utilizados en sectores como la seguridad, la vigilancia y la inspección industrial, están equipados con sensores de alta precisión, cámaras térmicas o de alta definición, y sistemas de comunicación robustos. Marcas como DJI o Parrot han adaptado sus tecnologías de drones para desarrollar vehículos terrestres con capacidades avanzadas de telemetría y control remoto.
    - Problemas y Áreas de Mejora: A pesar de sus avanzadas capacidades, estos modelos son costosos y a menudo requieren configuraciones especializadas. La principal área de mejora radica en la reducción del costo y la mejora de la autonomía de operación. Además, la integración de inteligencia artificial para la toma de decisiones autónomas en entornos complejos es un campo que aún está en desarrollo y que presenta oportunidades significativas de innovación.

1.2.2. Principales Problemáticas y Áreas de Mejora

1. Autonomía y Gestión de Energía:
    - Uno de los desafíos más comunes en todos los niveles de productos es la limitación en la autonomía, tanto en términos de duración de la batería como en la capacidad de transmitir datos de forma continua. Mejorar la eficiencia energética y desarrollar soluciones de gestión de energía más avanzadas, como sistemas de recarga inalámbrica o baterías de mayor capacidad, son áreas de mejora fundamentales.
2. Calidad de la Transmisión de Video:
    - La transmisión de video en tiempo real es una característica esencial para muchos de estos dispositivos, especialmente en aplicaciones de vigilancia o inspección. Sin embargo, la latencia, la calidad de la imagen y la estabilidad de la señal aún son problemáticas, particularmente en entornos con obstáculos o interferencias electromagnéticas. Mejorar los protocolos de transmisión y la calidad de las cámaras integradas es una prioridad.
3. Facilidad de Uso e Interfaz de Usuario:
    - Aunque algunos modelos avanzados ofrecen una amplia gama de funciones, a menudo la curva de aprendizaje es empinada debido a interfaces de usuario poco intuitivas o a la necesidad de conocimientos técnicos previos. El desarrollo de interfaces más amigables, así como la implementación de sistemas de control basados en gestos o comandos de voz, podría hacer estos productos más accesibles.
4. Capacidades de Navegación y Evitación de Obstáculos:
    - La navegación autónoma y la evitación de obstáculos siguen siendo áreas críticas de desarrollo. La mayoría de los modelos todavía dependen en gran medida del control manual o de algoritmos simples de evitación de colisiones. La integración de inteligencia artificial y aprendizaje automático para mejorar estas capacidades es un campo de investigación en crecimiento.
5. Compatibilidad y Modularidad:

- Muchos de los productos existentes enfrentan problemas de compatibilidad cuando se trata de integrar nuevos sensores o módulos. Crear sistemas más modulares que permitan a los usuarios añadir y actualizar componentes sin complicaciones significativas podría hacer que estos productos sean más flexibles y duraderos.

En resumen, aunque el mercado de coches teledirigidos con capacidades de sensado y visualización remota ha avanzado considerablemente, aún existen importantes áreas de mejora que abren la puerta a la innovación. La mejora de la autonomía, la calidad de transmisión, la facilidad de uso, las capacidades de navegación y modularidad son los focos principales en los que se centran los desarrollos actuales y futuros

1.3.OBJETIVOS Y PLANIFICACIÓN

Tras analizar las problemáticas actuales en el mercado de coches teledirigidos con capacidades de sensado y visualización remota, se ha identificado que las principales áreas de mejora están en la interfaz de usuario y en la capacidad de navegación remota y evitación de obstáculos. Debido a las limitaciones presupuestarias del proyecto, se ha decidido enfocar los esfuerzos en estas áreas específicas, con el objetivo de desarrollar un coche teledirigido que ofrezca una experiencia de usuario más intuitiva y eficiente, así como una navegación autónoma más segura y confiable.

1.3.1. Objetivos del Proyecto

El objetivo principal de este trabajo es explorar y evaluar diversas alternativas para construir un coche teledirigido que mejore la experiencia del usuario y las capacidades de navegación autónoma. A través de una planificación modular, se pretende llevar a cabo la implementación que mejor se adapte a estos objetivos, aprovechando los recursos disponibles de manera eficiente y maximizando el aprendizaje en cada fase del proyecto.

1. Mejora de la Interfaz de Usuario:
    - Desarrollar una interfaz de control que sea intuitiva, accesible y que minimice la necesidad de conocimientos técnicos avanzados por parte del usuario.

2. Capacidad de Navegación Remota y Evitación de Obstáculos:
    - Desarrollar y probar algoritmos de navegación que permitan al coche moverse de forma autónoma en entornos complejos, evitando obstáculos de manera eficaz.
    - Integrar sensores de ultrasonido, infrarrojos o cámaras que proporcionen datos en tiempo real para la toma de decisiones autónomas, optimizando la respuesta del coche ante posibles colisiones.
3. Aprendizaje transversal:
    - Abarcar la mayor cantidad posible de campos y estudiar todas las posibilidades para poder adquirir un criterio propio y poder tomar decisiones basadas en el conocimiento durante este proyecto y proyectos futuros.
    - Aprender y estudiar diferentes lenguajes de programación que sirvan como base para futuros estudios y proyectos.

1.3.2. Planificación Modular

La planificación del proyecto seguirá un enfoque modular, dividiendo el desarrollo en diferentes bloques que permitirán una evolución progresiva y ordenada. Este enfoque facilitará la evaluación continua de los resultados y la integración de mejoras en cada etapa. Los módulos principales se centrarán en:

- Diseño del Hardware: Selección y montaje de componentes que soporten las capacidades de navegación y sensado necesarias.
- Desarrollo de Software para la Interfaz de Usuario: Creación de una interfaz web en HTML que permita un control remoto intuitivo y fluido del coche.
- Implementación de Algoritmos de Navegación: Programación de rutinas en MicroPython para el manejo de sensores y la toma de decisiones de navegación en tiempo real.
- Pruebas e Iteración: Fase de pruebas en la que se evaluará el desempeño del coche en distintos escenarios, identificando áreas de mejora y aplicando ajustes según sea necesario.

Este enfoque modular permite centrarse primero en qué se quiere conseguir en cada fase del proyecto, sin detenerse aún en los detalles técnicos del cómo se implementará cada solución. La idea es mantener una flexibilidad que permita explorar diferentes alternativas y adaptar las soluciones a las necesidades que vayan surgiendo.

2. DISEÑO DEL SISTEMA

2.1.ESPECIFICACIONES DEL COCHE TELEDIRIGIDO

El diseño del coche teledirigido se enfoca en obtener un rendimiento equilibrado, priorizando una velocidad moderada y la asequibilidad de los componentes. Se busca una velocidad segura y controlada para adaptarse a diversos entornos, priorizando la precisión de la navegación sobre la velocidad máxima. Este enfoque permite el uso de componentes más accesibles en términos de costos, facilitando así la exploración y experimentación en el ámbito de la robótica. Para el control de velocidad, se ha implementado un control PWM, que permite ajustar la velocidad de los motores de manera eficiente.

El coche incluirá capacidades de visualización remota mediante una cámara integrada, lo que permitirá al usuario recibir un feed de video en tiempo real del entorno del vehículo. Esta funcionalidad es esencial para aplicaciones que requieren supervisión visual, como la exploración de áreas de difícil acceso. Además, contará con un sistema de detección de obstáculos mediante sensores que posibilitarán la navegación autónoma y la evitación de colisiones, mejorando la seguridad y el rendimiento del sistema en entornos complejos.

Finalmente, la interacción en tiempo real con el usuario se garantizará mediante una interfaz de control intuitiva que permitirá ajustar la navegación y monitorear el estado del coche de manera dinámica. Este enfoque proporciona una mayor flexibilidad y facilidad de uso, haciendo que el proyecto sea accesible para un público más amplio y útil en aplicaciones prácticas que requieran precisión en la navegación y control remoto eficiente.

2.2.ARQUITECTURA DEL SISTEMA

La arquitectura del sistema integra tanto el hardware como el software, optimizando la comunicación y la eficiencia operativa del coche teledirigido. En el núcleo del sistema, se implementa un microcontrolador Raspberry Pi Pico W compatible con MicroPython, que actúa como el cerebro central del coche. Este componente central gestiona las operaciones críticas del vehículo, desde el control de los motores y la interpretación de datos de los sensores hasta la administración de la comunicación bidireccional con la interfaz web.

En cuanto a la comunicación, se emplea una arquitectura bidireccional que permite una interacción fluida entre el coche y la interfaz web. La interfaz web, desarrollada en HTML, se convierte en el puente virtual entre el usuario y el coche teledirigido. Se opta por la implementación de la tecnología WebSocket (Anexo II), ya que esta facilita la transmisión instantánea de datos en ambas direcciones, lo que garantiza una experiencia de usuario en tiempo real, permitiendo tanto la interacción del usuario con los sensores como viceversa. Este diseño arquitectónico proporciona una base sólida para el control remoto del coche desde cualquier dispositivo conectado a internet, brindando al usuario un control preciso y la posibilidad de una retroalimentación inmediata sobre el estado y las acciones del vehículo. En las secciones posteriores, se detallará la construcción del hardware, la programación en MicroPython, la interfaz web en HTML y la implementación de la cámara para concretar esta visión arquitectónica.

2.3.ELECCIÓN DE COMPONENETES

La elección de componentes se ha llevado a cabo considerando cuidadosamente la funcionalidad y la accesibilidad económica del proyecto. Se ha adquirido el controlador de motores L298N para gestionar eficientemente los motores de corriente continua, proporcionando un control preciso sobre la tracción y la dirección del coche. Los motores DC, junto con las ruedas seleccionadas, fueron elegidos por su adecuación al propósito del proyecto, buscando un equilibrio entre velocidad y maniobrabilidad.

Para la alimentación del sistema, se ha optado por una batería de 12V, proporcionando la potencia necesaria para el funcionamiento sostenido del coche. La elección de sensores incluye un sensor ultrasónico de distancia HC-SR04 y un sensor de temperatura para monitorear las condiciones ambientales. Además, se incorpora una cámara ESP32 para la visualización remota, lo que permite al usuario recibir imágenes en tiempo real del entorno del vehículo.

La Raspberry Pi Pico W se integra como el microcontrolador principal, aprovechando su capacidad para ejecutar MicroPython y su compatibilidad con una variedad de periféricos. Además, cuenta con la posibilidad de conexión Wifi, esencial para el desarrollo del proyecto.

Esta elección de componentes refleja una estrategia de diseño que busca la eficacia, la economía y la flexibilidad en la implementación del coche teledirigido controlado desde una página web. En el Anexo III se puede ver la justificación de la elección de estos componentes, así como sus características principales.

3. DESARROLLO DEL HARDWARE

3.1.CONSTRUCCIÓN DEL CHASIS

El chasis del coche teledirigido fue diseñado utilizando AutoCAD, un software ampliamente utilizado para diseño asistido por computadora que permite una alta precisión en la creación de planos y estructuras. Durante el diseño, se consideraron cuidadosamente las dimensiones y el espacio necesario para acomodar todos los componentes electrónicos y mecánicos del sistema, garantizando una distribución óptima que facilitara tanto la estabilidad como la accesibilidad.

El diseño del chasis incluyó áreas específicas destinadas a la colocación de piezas clave, como la Raspberry Pi Pico W, el controlador de motores L298N, los sensores y la batería de 12V. Se integraron agujeros, posicionados para fijar las ruedas al chasis de manera segura y asegurar una adecuada alineación y funcionalidad del sistema de tracción. Además, se incorporaron agujeros adicionales para permitir el paso ordenado de los cables, evitando cruces innecesarios que pudieran interferir con el movimiento de las ruedas o con la conectividad de los componentes.

El plano y medidas del chasis se puede encontrar en el Anexo IV

3.2.INTEGRACIÓN DE MOTORES

La integración de los motores en el coche teledirigido se llevó a cabo mediante el uso de cuatro motores de corriente continua (DC) de 3-12V cada uno. Los motores fueron seleccionados con ejes que permitieron una conexión eficiente con las cuatro ruedas del vehículo. La elección de cuatro motores se basó en la búsqueda de un equilibrio óptimo entre tracción y estabilidad durante la conducción. La facilidad de conexión entre los ejes y las ruedas facilitó el proceso de montaje, permitiendo una integración rápida y segura.

Para estimar la velocidad máxima del coche, se realizaron cálculos basados en un radio de rueda de 3 cm. Utilizando la fórmula de velocidad angular $v=\omega \cdot r$, donde v es la velocidad lineal, ω es la velocidad angular y r es el radio de la rueda, se determinaron las velocidades máximas teóricas, con un rango desde 2km/h hasta 12km/h aproximadamente. Este análisis permitió ajustar parámetros como la tensión de alimentación y la relación de reducción para optimizar

el rendimiento del coche dentro de límites seguros y eficientes. Cabe resaltar que la velocidad teórica no tiene en cuenta el peso del coche y su rozamiento por el suelo, por lo que las velocidades reales se verán significativamente reducidas, llegando a no avanzar el coche con baja alimentación.

La conexión de los motores con el módulo de control L298N fue esencial para gestionar la dirección y la velocidad de manera precisa. El módulo L298N, mediante su capacidad de control de motores bidireccional, permitió una integración eficiente y versátil. Las salidas del módulo se conectaron a cada motor, permitiendo el control individual de la velocidad y la dirección. Este enfoque proporcionó una base sólida para el desarrollo de la funcionalidad de conducción remota del coche teledirigido desde la interfaz web.

3.3.INTEGRACIÓN DE SENSORES

La integración de sensores en el coche teledirigido incluye un sensor ultrasónico de distancia HC-SR04 para la detección de obstáculos y un sensor de temperatura DS18B20 para el monitoreo ambiental. El sensor HC-SR04, montado en la parte frontal del chasis, utiliza ondas ultrasónicas para medir la distancia a posibles obstáculos, lo que permite al coche tomar decisiones de navegación, como frenar o cambiar de dirección. Este sensor se conecta a los pines GPIO de la Raspberry Pi Pico W y se controla mediante un algoritmo en MicroPython que calcula la distancia en función del tiempo de retorno de las ondas ultrasónicas.

Por otro lado, el sensor de temperatura DS18B20 se ha integrado mediante el protocolo 1-Wire, permitiendo lecturas precisas de la temperatura ambiental. Este sensor digital se encuentra ubicado en una posición protegida del chasis para evitar daños y asegurar una medición efectiva. La información recopilada de ambos sensores se gestiona y se transmite al usuario a través de la interfaz de control web, mejorando la percepción del entorno y la capacidad de respuesta del coche en diferentes condiciones operativas.

Además, se intentó incorporar dos sensores HW201, específicamente diseñados para detectar el cambio de color blanco-negro mediante tecnología infrarroja, con el objetivo de que el robot pudiera tener un sistema de seguimiento de líneas. Sin embargo, es importante destacar que, a pesar de la compra de estos sensores de infrarrojos, se encontraron dificultades en su

calibración adecuada para detectar cambios de color de manera consistente, por lo que la implementación del sistema no ha sido posible debido a esta limitación.

3.4.INTEGRACIÓN DE LA CÁMARA ESP32

La integración de la cámara ESP32 en el coche teledirigido se ha realizado con el objetivo de proporcionar visualización remota y aumentar la capacidad de control del vehículo en tiempo real. Para maximizar el rango de visión de la cámara y permitir un control más flexible de la perspectiva, se ha montado sobre las hélices de un servomotor. Esta configuración permite que la cámara pueda rotar y ajustarse en diferentes direcciones, mejorando significativamente la capacidad de observación del entorno.

El servomotor, controlado desde el microcontrolador principal, proporciona un movimiento suave y preciso, lo que permite al usuario cambiar el ángulo de visión de la cámara de manera dinámica a través de la interfaz de control. Esta disposición no solo optimiza la visualización en tiempo real para evitar obstáculos y navegar de manera segura, sino que también facilita el monitoreo de áreas circundantes sin mover el chasis del coche, mejorando la funcionalidad general del sistema de control remoto.

La unión entre la cámara y el servomotor se ha realizado mediante celo flexible, proporcionando una unión fija, pero facilitando el movimiento de las hélices sin la presencia de más cables y sistemas de sujeción.

Imagen: Cámara ESP32 y Servomotor

Con todo ello, el resultado final del hardware montado se puede ver en la siguiente imagen, destacar que, pese al desorden de cables que se puede encontrar, en el siguiente apartado se podrá ver un esquema de las conexiones.

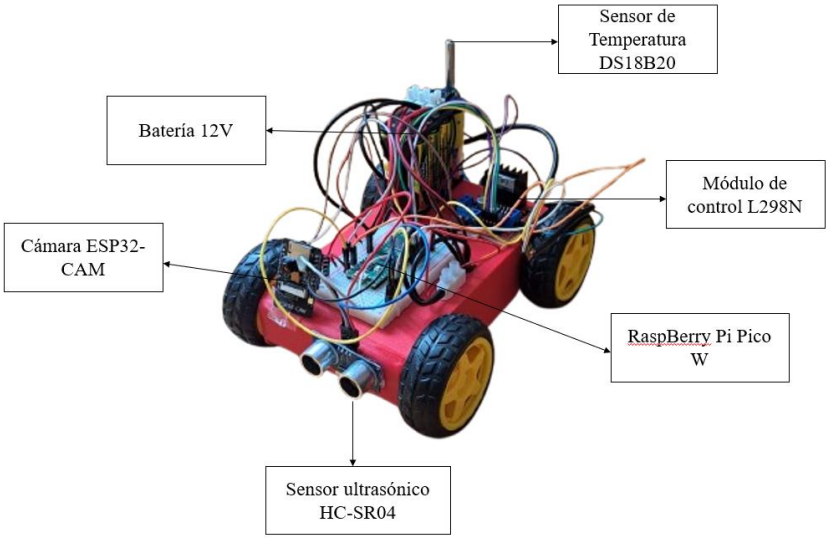
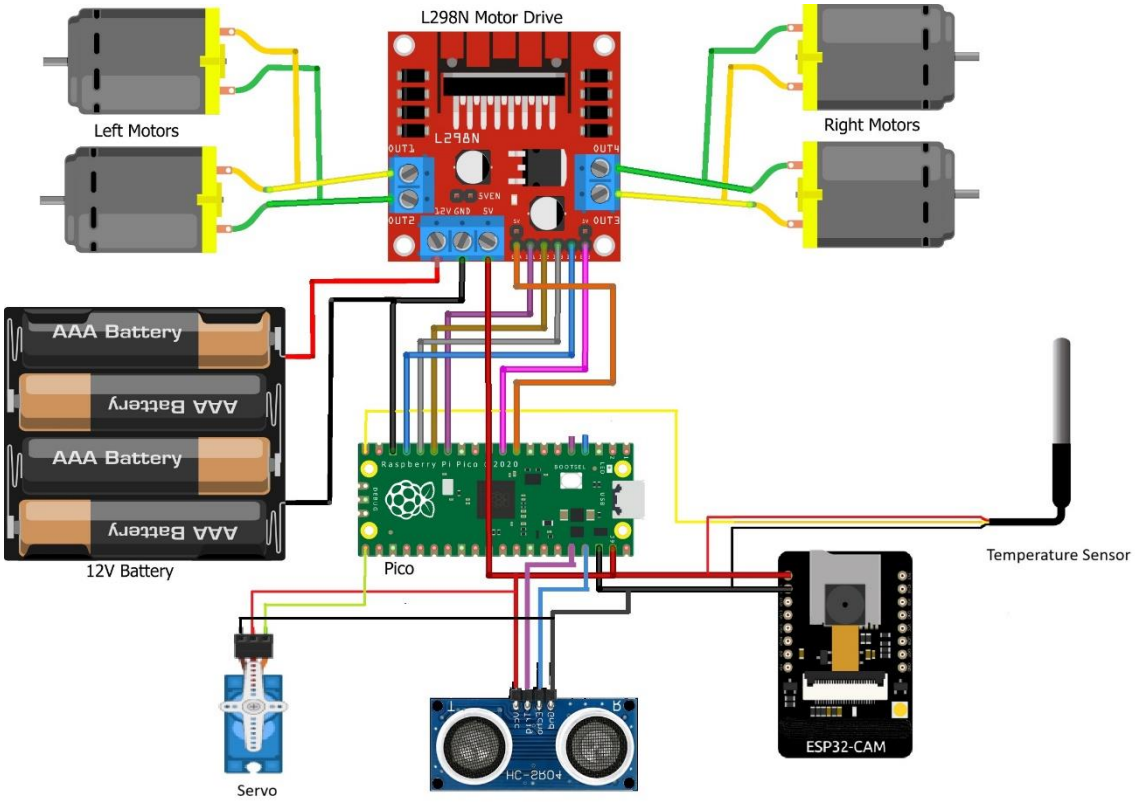


Imagen: Esquema del Hardware

4. DESARROLLO DEL SOFTWARE

En el apartado de desarrollo de software, se ha implementado una combinación de tecnologías y lenguajes de programación para crear una interfaz web interactiva y controlar el sistema del coche de manera eficiente. Se ha utilizado HTML, CSS y JavaScript para diseñar y programar la interfaz de usuario, proporcionando una experiencia visual y funcional. Para la programación de los microcontroladores, se han empleado MicroPython para la Raspberry Pi Pico W y C para la cámara ESP32, permitiendo un control preciso del hardware y una integración fluida entre los componentes del sistema.

A continuación, se puede explorar el esquema general de conexiones del proyecto.



Todos los cables rojos hacen referencia a la toma de corriente (VCC). Por otra parte, los cables negros hacen referencia a la conexión a tierra (GND). Los cables con otros colores están conectados a diferentes pines de control de la Raspberry.

4.1.PROGRAMACIÓN EN MICROPYTHON

El código del coche teledirigido está escrito en MicroPython y se organiza en varias secciones clave que abordan diferentes aspectos del control del vehículo y la interacción con el entorno. La estructura del código comienza con la configuración de los módulos necesarios para la conexión WiFi, el manejo de sockets para comunicación web, y la integración de sensores y actuadores como los motores, el sensor ultrasónico de distancia, el sensor de temperatura DS18B20 y la cámara montada en un servomotor.

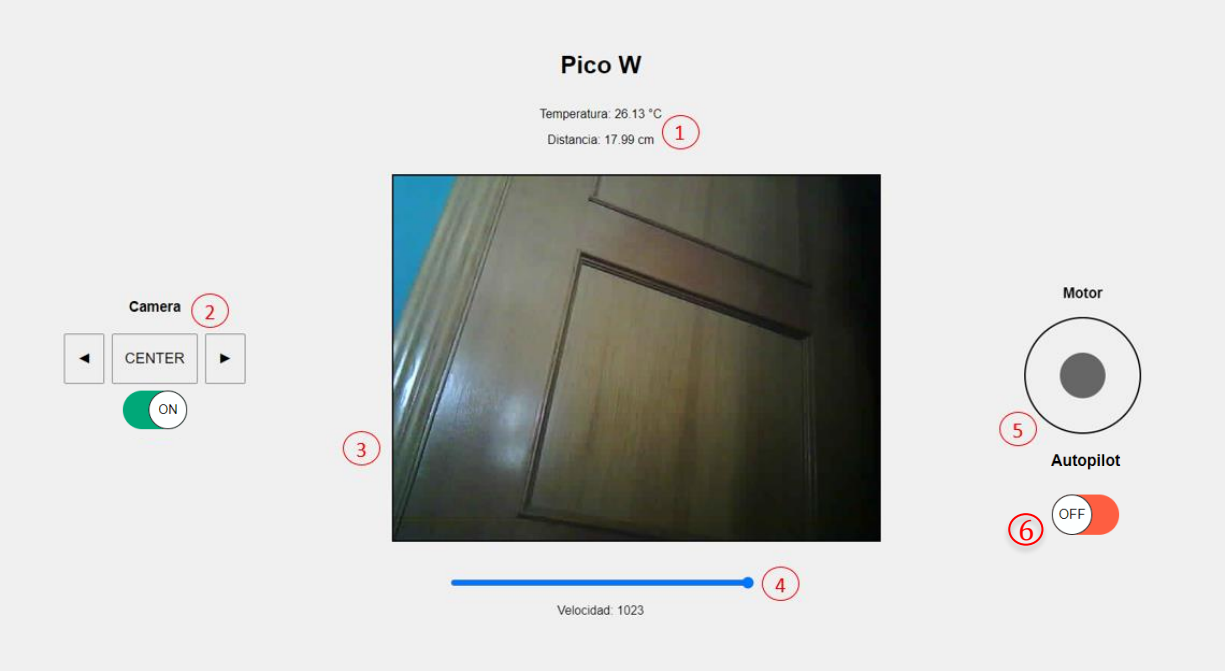
El código incluye funciones para manejar la conexión a una red WiFi, la configuración de un servidor web que permite el control remoto del coche, y la generación de respuestas dinámicas en HTML para visualizar datos en tiempo real, como la temperatura y la distancia a obstáculos. Los módulos PWM son utilizados para el control de velocidad de los motores, mientras que los servomotores controlan la orientación de la cámara. La lógica de control permite movimientos básicos del coche (avanzar, retroceder, girar a la izquierda o derecha) y el ajuste de la velocidad. Asimismo, se incluyen funciones para la lectura de sensores que proporcionan datos esenciales al usuario para el manejo eficiente del vehículo.

Inicialmente, se valoró la posibilidad de utilizar Anvil como plataforma para la comunicación remota, dada su facilidad de uso para crear interfaces web y conectar dispositivos IoT. Sin embargo, esta opción fue descartada porque Anvil no soporta la transmisión de video en tiempo real, una funcionalidad crítica para el proyecto dado que la visualización remota del entorno es esencial para el control efectivo del coche. Optar por un servidor web local permite enviar comandos y recibir tanto datos de sensores como la transmisión de video en tiempo real, satisfaciendo así los requisitos del proyecto.

Para profundizar en los detalles de la programación en MicroPython, consultar el Anexo V o el link a GitHub donde se encuentra el código en su totalidad.

4.2.INTERFAZ WEB EN HTML

La página web se ha diseñado combinando el desarrollo en HTML, JavaScript y CSS. Se han incorporado diversas funcionalidades que se explicarán a continuación.



NÚMERO 1: Información en tiempo real sobre temperatura y distancia del sensor al objeto más cercano. Se trata de un panel meramente informativo y que va actualizándose en tiempo real. Destacar que el sensor de ultrasonidos se encuentra colocado en la parte frontal del vehículo, permitiendo detectar los objetos enfrentados al vehículo, pero siendo complicado detectar los objetos que se encuentran laterales al vehículo.

NÚMERO 2: Control de la cámara. Las flechas y el botón center controlan el servomotor enganchado a la cámara. Esto nos permite tener un rango de visualización mucho más amplio. El interruptor controla el encendido y apagado de la cámara. Si bien por defecto va a estar siempre encendida, existe la posibilidad de apagarla si fuera necesario. El control de estas funcionalidades también puede hacerse mediante el uso del teclado del ordenador, siendo la tecla “C” la que enciende y apaga la cámara y las teclas “A”, “S” y “D” las que permiten controlar el servomotor.

NÚMERO 3: Visualización del vídeo en directo. Este recuadro consta de un “iframe”, herramienta de desarrollo web que permite la visualización de una página web externa, en la cual está ubicada la emisión en directo del video de la cámara ESP32.

NÚMERO 4: Control de velocidad del coche. Mediante esta barra interactiva, se puede controlar la velocidad de los motores DC del coche, la velocidad por defecto es mínima y se encuentra en un rango de 0 a 1023 (16 bits).

NÚMERO 5: El control de la dirección del vehículo se lleva a cabo mediante un joystick interactivo. De esta forma, la experiencia del usuario es mucho mejor y permite un control suave y preciso, incorporando control PWM también para realizar los giros. De igual forma que con el movimiento del servomotor, este control de velocidad se puede realizar con las fechas del teclado si se esta controlando desde un ordenador.

NÚMRO 6: Piloto Automático. Este botón pone el coche en piloto automático, emulando el movimiento de un “Roomba”. En este modo es capaz de detectar obstáculos de forma autónoma y evitarlos convenientemente, reduciendo su velocidad y girando hasta encontrar un camino liberado.

Estas son todas las funcionalidades con las que cuenta el coche teledirigido y cómo se realiza la interacción persona-coche. Para su desarrollo, se ha primado la facilidad de interacción del usuario y la claridad de funcionamiento del sistema. Este es un factor clave en el desarrollo ya que mediante esta web se puede controlar la totalidad del vehículo y ejecutar funciones complejas de una forma fácil y “user friendly”

4.3.COMUNICACIÓN ENTRE HARDWARE Y SOFTWARE

La comunicación entre el software y el hardware del coche teledirigido se gestiona a través de un servidor web local que permite la interacción en tiempo real. El microcontrolador, en este caso la Raspberry Pi Pico W, se conecta a una red WiFi y aloja un servidor que facilita tanto el envío de datos de sensores a la interfaz web como la recepción de comandos del usuario para controlar el coche.

El envío de datos se realiza desde los sensores (como el sensor ultrasónico y el sensor de temperatura) hacia la interfaz web, proporcionando información actualizada del entorno del coche. Por otro lado, la recepción de comandos permite al usuario controlar los motores y la cámara desde la web. Esta comunicación es manejada mediante sockets, que establecen una conexión entre el cliente (navegador web) y el servidor (microcontrolador), permitiendo la transmisión de datos de manera continua.

4.4. CONFIGURACIÓN DE CÁMARA ESP23

La configuración de la cámara web en este proyecto se lleva a cabo utilizando una ESP32-CAM, que permite capturar video y transmitirlo en tiempo real a través de un servidor web alojado en la ESP32. Este servidor web maneja peticiones para mostrar el flujo de video o imágenes capturadas desde la cámara, facilitando el control remoto del coche teledirigido. La configuración implica conectar la ESP32 a una red WiFi, inicializar la cámara y crear un servidor web que sirva el flujo de video a los clientes que se conecten.

El código de configuración de la cámara se ha realizado desde Arduino IDE y se divide en varias partes, que incluyen la inicialización de la cámara, la configuración del servidor web, el manejo de peticiones HTTP, y el procesamiento y transmisión de imágenes en tiempo real. La ESP32-CAM se programa para capturar imágenes en formato JPEG, que luego se transmiten a los clientes que acceden al flujo de video mediante una conexión HTTP.

Para más información sobre la configuración de la cámara se recomienda consultar el Anexo VI.

5. IMPLEMENTACIÓN Y PRUEBAS

5.1.ENSAMBLAJE DEL COCHE

Una vez que se confirmó el correcto funcionamiento del código y se completó la fase de programación, se procedió al ensamblaje físico del coche teledirigido. Dada la naturaleza experimental del proyecto, se optó por una solución práctica y eficiente en términos de coste y disponibilidad de recursos. Todos los componentes fueron asegurados en el chasis utilizando cinta de doble cara para garantizar una fijación firme y ordenada.

Para la fijación de los motores y el servomotor al chasis, se empleó silicona caliente, proporcionando un agarre robusto y al mismo tiempo permitiendo una fácil manipulación en caso de necesidad de ajustes o mejoras. El cableado, un aspecto crítico para mantener la integridad de la conexión eléctrica, fue cuidadosamente ensamblado a mano para asegurar una disposición organizada y evitar posibles interferencias entre los cables. Aunque se reconoce que una placa diseñada específicamente podría haber proporcionado una solución más estructurada y profesional, se optó por esta aproximación debido a las limitaciones presupuestarias. A pesar de estas consideraciones, el ensamblaje cuidadoso y funcional del coche teledirigido ha permitido un desempeño sólido y ha facilitado la implementación práctica de la propuesta.

5.2.PRUEBAS FUNCIONALES

El apartado de pruebas funcionales es una fase crucial para verificar el rendimiento y la coherencia del coche teledirigido. Inicialmente, se lleva a cabo una evaluación exhaustiva de la conexión Wi-Fi para garantizar una comunicación estable entre la interfaz web y el hardware. Una vez confirmada la conectividad, se procede a realizar pruebas específicas de las funciones del coche teledirigido, comenzando con las operaciones básicas de movimiento como "adelante", "atrás", "izquierda" y "derecha", para más adelante comprobar la funcionalidad del control desde la interfaz web, probando el “joystick” para el control de dirección del coche y realizando los ajustes necesarios.

Durante este proceso, se realizaron ajustes y correcciones en el código según fuera necesario, abordando posibles problemas de respuesta, sincronización o cualquier anomalía en el

comportamiento del coche. Las pruebas se llevaron a cabo de manera incremental, asegurándose de que cada función individual operara correctamente antes de avanzar a la siguiente. Este enfoque paso a paso permitió una identificación eficaz de posibles problemas y la aplicación de soluciones específicas.

En cuanto a la cámara, la implementación de su código fue eficaz desde el primer momento, creándose una web local en la que consultar el video en directo y permitiendo una edición de parámetros centrados en la fluidez y calidad de la retransmisión. Mediante ajuste fino, se ha podido llegar a una configuración óptima en la que el video se puede visualizar de forma cómoda y fluida.

Estas pruebas fueron realizadas en todo momento con la Raspberry conectada al ordenador, y a la hora de ejecutar el código desde el propio microcontrolador (main.py), ocurrió un error que se deduce fue causado por un cortocircuito. Se intentó ejecutar el código desde el microcontrolador y desde el ordenador a la vez, causando un fallo que obligó a resetear el microcontrolador varias veces y que más tarde causó la imposibilidad de ejecutar el código desde el mismo, obligando a la adquisición de otro microcontrolador Raspberry Pi Pico W.

5.3.FUNCIONAMIENTO DEL COCHE

El funcionamiento del coche es simple. Primero, en la configuración inicial del código, es necesario ingresar el nombre y la contraseña de la red WiFi en la que se desea ejecutar el proceso. En este caso, la red es proporcionada por un teléfono móvil en modo "Compartir datos". También es importante conectar el dispositivo desde el que se accederá a la interfaz web, ya que, al realizar una implementación local, esta es la única forma de garantizar un funcionamiento adecuado.

Una vez que todos los dispositivos (cámara ESP32, Raspberry Pi Pico W y ordenador) están conectados a la red, cada uno de ellos tendrá una dirección IP asignada. Estas direcciones IP se pueden consultar en la configuración del router o, en este caso, en los ajustes del dispositivo móvil que actúa como router. Luego, es necesario ingresar la dirección IP de la cámara en el apartado correspondiente del código de la página web y, finalmente, introducir la dirección IP

de la Raspberry Pi en cualquier navegador. Esto permitirá acceder a la página web y controlar el coche de manera remota.

6. RESULTADOS Y DISCUSIÓN

6.1.LOGROS ALCANZADOS

El proyecto del coche teledirigido con capacidades de sensado y visualización remota ha logrado cumplir con los principales objetivos propuestos, centrados en mejorar la interfaz de usuario y la capacidad de navegación remota y evitación de obstáculos. A continuación, se detallan los logros alcanzados en cada uno de estos objetivos:

Mejora de la Interfaz de Usuario:

Uno de los objetivos clave era desarrollar una interfaz de control intuitiva y accesible para permitir un manejo remoto eficiente del coche teledirigido. Este objetivo se ha conseguido mediante la creación de una interfaz web basada en HTML, que permite a los usuarios controlar el vehículo de manera sencilla a través de cualquier navegador web. La interfaz facilita el envío de comandos para mover el coche en diferentes direcciones, ajustar la velocidad y controlar la orientación de la cámara, lo cual se realiza de manera fluida gracias a la implementación de un servidor web local utilizando sockets TCP en la Raspberry Pi Pico W. La interfaz también permite visualizar en tiempo real los datos de los sensores, como la temperatura y la distancia a obstáculos, proporcionando una experiencia de usuario completa y robusta.

Capacidad de Navegación Remota y Evitación de Obstáculos:

Se ha logrado integrar y programar de manera efectiva un sensor ultrasónico HC-SR04 para la detección de obstáculos, lo que permite al coche navegar de forma segura en diferentes entornos. El sensor se ha configurado para medir distancias en tiempo real, y su integración con el microcontrolador permite que el coche tome decisiones autónomas, como detenerse o cambiar de dirección cuando detecta un obstáculo cercano. Esta funcionalidad cumple con el objetivo de mejorar la seguridad y eficiencia del vehículo durante la navegación remota.

Además, se ha implementado un sistema de control de motores mediante PWM (Modulación de Ancho de Pulso), que permite un ajuste preciso de la velocidad del coche, garantizando un manejo suave y controlado. Esto, junto con la capacidad de detección de obstáculos, ha resultado en una navegación remota más precisa y eficiente.

Visualización Remota mediante Cámara Web:

Otro objetivo importante era proporcionar una visualización remota en tiempo real del entorno del coche. Este objetivo se ha logrado mediante la integración de una cámara ESP32-CAM, montada sobre un servomotor que permite ajustar el ángulo de visión. La configuración de un servidor web en la ESP32-CAM ha permitido la transmisión de video en tiempo real, accesible a través de la interfaz web. Esta capacidad de visualizar el entorno y ajustar la perspectiva de la cámara según sea necesario ha mejorado significativamente el control y la operabilidad del coche en situaciones de navegación remota.

Optimización del Control y la Eficiencia del Sistema:

Se ha implementado un sistema de comunicación eficiente entre el software y el hardware, utilizando sockets TCP para gestionar tanto el envío de datos desde los sensores como la recepción de comandos del usuario. Esta comunicación bidireccional ha permitido una interacción en tiempo real entre el usuario y el coche, optimizando el control remoto y proporcionando una respuesta rápida y precisa a las entradas del usuario.

En conclusión, los logros alcanzados en este proyecto reflejan el cumplimiento de los objetivos establecidos al inicio. Hemos conseguido mejorar significativamente la interfaz de usuario, optimizar la navegación y la evitación de obstáculos, y proporcionar una visualización remota en tiempo real, todo ello dentro de los límites de presupuesto y componentes accesibles. Esto convierte al coche teledirigido en una plataforma robusta y versátil para la exploración y experimentación en el ámbito de la robótica y la electrónica digital.

7. CONCLUSIONES

7.1.LIMITACIONES Y MEJORAS FUTURAS

A pesar de los logros alcanzados, se reconocen ciertas limitaciones en el diseño actual. La calibración de los sensores infrarrojos ha presentado desafíos, afectando la detección de cambios de color. Además, la fijación del hardware mediante cinta de doble cara y silicona caliente, aunque funcional, podría beneficiarse de una solución más estructurada y modular.

Otra limitación importante es la eficiencia de la batería, que podría mejorarse para proporcionar una mayor autonomía y tiempo de operación del coche teledirigido. Actualmente, el sistema depende de una batería de 12V que, aunque suficiente para los componentes utilizados, podría ser optimizada con un sistema de gestión de energía más avanzado o baterías de mayor capacidad y menor peso.

Para mejoras futuras, se contempla la posibilidad de desarrollar una placa específica que integre los componentes de manera más eficiente. Asimismo, se explorarán opciones para mejorar la precisión de los sensores infrarrojos y optimizar la respuesta del coche a cambios en el entorno. También se considerará el uso de técnicas de inteligencia artificial para mejorar la navegación autónoma y la detección de obstáculos, lo que permitiría al coche adaptarse mejor a condiciones cambiantes o complejas.

Además, la composición del software permite añadir al coche todas las funcionalidades que se quiera (navegación GPS, sistema de sonido, luces led…), el único límite es el presupuesto y la imaginación. Estas mejoras futuras podrían convertir al coche en una plataforma aún más versátil y adaptada a diversas aplicaciones, desde la educación y la investigación hasta usos más avanzados en robótica y sistemas autónomos.

7.2.RECAPITULACIÓN

En este proyecto, se ha desarrollado un coche teledirigido con capacidades de sensado y visualización remota, alcanzando los objetivos de mejorar la interfaz de usuario y optimizar la navegación remota y evitación de obstáculos. Utilizando una combinación de hardware accesible y programación en MicroPython, se han integrado sensores de distancia y temperatura, un sistema de control de motores con PWM, y una cámara ESP32-CAM para transmisión en tiempo real. Aunque el diseño actual presenta limitaciones en la calibración de algunos sensores y la eficiencia de la batería, se han identificado mejoras futuras que permitirán expandir las funcionalidades del coche, adaptándolo a más aplicaciones y optimizando su rendimiento y autonomía.

7.3.APRENDIZAJES

A lo largo de este proyecto, se ha abordado una amplia variedad de temas que abarcan desde el diseño y elección de componentes hasta la programación en distintos lenguajes, como MicroPython y C, para integrar diversos sistemas y funcionalidades. Este enfoque integral ha permitido adquirir un entendimiento profundo sobre cómo desarrollar un proyecto de robótica desde cero, considerando todas las etapas, desde la planificación inicial hasta la implementación y pruebas. La elección cuidadosa de sensores, motores, microcontroladores y la creación de un sistema de comunicación eficiente han sido clave para entender cómo cada componente y decisión de diseño afecta al resultado final y al rendimiento del coche teledirigido.

Gracias a la amplia selección de contenidos que han abarcado el trabajo, he conseguido estudiar, aprender e implementar la programación en diferentes lenguajes, entendiendo las ventajas de cada uno de ellos y pudiendo valorar consecuentemente la mejor herramienta a utilizar en cada momento. Asimismo, al haber hecho una exhaustiva comparación de los productos del mercado para el desarrollo del hardware, me ha servido para profundizar en el mundo de la robótica. Esto me ha permitido hacerme una idea de las infinitas aplicaciones que

este tiene y de los futuros desafíos que afronta, muchos de ellos centrados en la inteligencia artificial.

El proyecto concluye aquí en su forma actual, pero las posibilidades de aplicaciones y añadidos son prácticamente ilimitadas. Gracias a la flexibilidad del diseño y la modularidad del software, el coche puede expandirse para incluir nuevas funcionalidades, como navegación GPS, sistemas de sonido, más sensores ambientales, o incluso capacidades de inteligencia artificial para la navegación autónoma. Este proyecto ha sentado una base sólida que no solo cumple con los objetivos planteados, sino que también abre la puerta a futuras mejoras e innovaciones, demostrando que la robótica y la programación pueden ser tan vastas y creativas como el propio ingenio y la imaginación lo permitan.

8. REFERENCIAS

8.1. BIBLIOGRAFÍA

MicroPython en Raspberry Pi Pico. *(s.f.). MicroPython Documentation*. Obtenido de https://docs.micropython.org: Referencia utilizada para la programación del microcontrolador Raspberry Pi Pico W con MicroPython, cubriendo el uso de GPIO, PWM y bibliotecas de comunicación.

ESP32-CAM Video Streaming Server Tutorial. *(s.f.). Random Nerd Tutorials*. Obtenido de https://randomnerdtutorials.com/esp32-cam-video-streaming-web-server/: Guía práctica utilizada para configurar la cámara ESP32-CAM para transmitir video en tiempo real mediante un servidor web.

Python Socket Programming. *(s.f.). Python.org*. Obtenido de https://docs.python.org/3/library/socket.html: Documentación oficial de Python que detalla la programación de sockets TCP/IP, fundamental para establecer la comunicación entre el coche teledirigido y la interfaz web.

DS18B20 Temperature Sensor Guide. *(s.f.). Maxim Integrated*. Obtenido de https://www.maximintegrated.com/en/products/interface/sensor-ds18b20.html: Información sobre la configuración y uso del sensor de temperatura DS18B20 en proyectos de electrónica digital.

PWM Motor Control with MicroPython. *(s.f.). Electronics Hub*. Obtenido de https://www.electronicshub.org/: Guía utilizada para implementar el control de velocidad de los motores del coche teledirigido mediante PWM.

How to Use Sockets in Python for Networking. *(s.f.). Real Python*. Obtenido de https://realpython.com/python-sockets/: Referencia que detalla el uso de sockets en Python para comunicación en red, aplicada en la comunicación entre el coche y la interfaz web.

8.2. DOCUMENTACIÓN TÉCNICA

ESP32-CAM Datasheet and Technical Manual. *Espressif Systems*. Documento técnico que detalla las especificaciones de hardware, características, y configuraciones posibles de la ESP32-CAM, crucial para la integración de la cámara en el proyecto.

L298N Motor Driver Datasheet. *STMicroelectronics*. Información detallada sobre el controlador de motores L298N, explicando sus capacidades de manejo de corriente, pines de control, y su uso para manejar motores de corriente continua.

HC-SR04 Ultrasonic Sensor Datasheet. *Elecfreaks*. Documento que proporciona detalles técnicos sobre el funcionamiento del sensor ultrasónico HC-SR04, utilizado para la detección de obstáculos.

DS18B20 Temperature Sensor Datasheet. *Maxim Integrated*. Especificaciones del sensor de temperatura DS18B20, incluyendo su rango de temperatura, precisión, y el protocolo 1-Wire para la comunicación.

Raspberry Pi Pico W Datasheet. *Raspberry Pi Foundation*. Información técnica sobre las capacidades del microcontrolador Raspberry Pi Pico W, especialmente en la implementación de WiFi, GPIO, y comunicación en red.

ESP-IDF Programming Guide for ESP32. *Espressif Systems*. Documentación oficial del framework de desarrollo de ESP32 que ofrece guías sobre cómo utilizar la biblioteca ESP-IDF para la configuración avanzada de la cámara.

HTML5 and JavaScript Documentation. *Mozilla Developer Network (MDN).* Referencias y guías técnicas sobre el uso de HTML y JavaScript para desarrollar la interfaz web del coche teledirigido y manejar la transmisión de datos y comandos en tiempo real.

8.3.REFERENCIAS DE CÓDIGO

MicroPython Libraries. *(s.f.). MicroPython Documentation*. Obtenido de https://docs.micropython.org

ESP32 Camera Web Server Code. *(s.f.). GitHub Repository by Random Nerd Tutorials*. Obtenido de https://github.com/RuiSantosdotme/ESP32-CAM

OneWire and DS18X20 Libraries for MicroPython. *(s.f.). MicroPython GitHub Repository*. Obtenido de https://github.com/micropython/micropython-lib

Python Socket Programming Examples. *(s.f.). Real Python*. Obtenido de https://realpython.com/python-sockets/ entre el coche teledirigido y la interfaz web.

PWM Control in MicroPython. *(s.f.). Electronics Hub*. Obtenido de https://www.electronicshub.org/

ESP-IDF for ESP32 Libraries. (*s.f.). Espressif Systems GitHub Repository*. Obtenido de https://github.com/espressif/esp-idf

WebServer Library for ESP32-CAM. *(s.f.). Arduino Library Manager*. Obtenido de https://www.arduino.cc/en/guide/libraries

HTML5, CSS, and JavaScript for Web Interfaces. *(s.f.). Mozilla Developer Network (MDN)*. Obtenido de https://developer.mozilla.org/en/

Machine Library for MicroPython. *(s.f.). MicroPython Documentation*. Obtenido de https://docs.micropython.org/en/latest/library/machine.html

Anexo I: Control PWM en MicroPython

El Control por Modulación de Ancho de Pulso (PWM, por sus siglas en inglés) es una técnica ampliamente utilizada en la electrónica para regular la potencia entregada a un dispositivo, como un motor de corriente continua (DC). En el contexto de un coche teledirigido, el PWM permite controlar la velocidad de los motores ajustando la cantidad de energía que reciben, sin la necesidad de variar el voltaje de alimentación. En este anexo, se explicará cómo se implementa el control PWM en MicroPython, un lenguaje de programación liviano diseñado para microcontroladores como la Raspberry Pi Pico W.

¿Qué es el PWM?

PWM es un método para simular una tensión variable aplicando un voltaje constante de manera intermitente. Este encendido y apagado rápido de la señal se denomina ciclo de trabajo (duty cycle). El ciclo de trabajo se define como el porcentaje de tiempo que la señal está en alto (encendido) durante un período completo de la señal. Por ejemplo, un ciclo de trabajo del 50% significa que la señal está encendida el 50% del tiempo y apagada el otro 50%.

La frecuencia de la señal PWM determina cuántas veces la señal pasa por un ciclo completo de encendido y apagado en un segundo. Una frecuencia alta y un ciclo de trabajo bajo pueden resultar en un control de velocidad fino y eficiente para motores DC.

Implementación de PWM en MicroPython

MicroPython facilita el uso del PWM mediante su biblioteca estándar, que permite a los desarrolladores definir la frecuencia y el ciclo de trabajo para pines GPIO (General Purpose Input/Output) específicos. A continuación, se presenta un ejemplo de código básico para implementar PWM en la Raspberry Pi Pico W para controlar un motor DC.

from machine import Pin, PWM

import time

```
# Configuración del pin GPIO para PWM
motor_pin = Pin(15)  # Pin donde se conecta el motor
pwm = PWM(motor_pin)

# Configuración de la frecuencia de PWM
pwm.freq(1000)  # Frecuencia de 1000 Hz

# Función para ajustar la velocidad del motor
def set_speed(duty_cycle):
    pwm.duty_u16(int(duty_cycle * 65535 / 100))  # Convertir porcentaje a rango de 16 bits

# Ejemplo de uso: Aumentar gradualmente la velocidad del motor
for i in range(0, 101, 10):  # Incrementos del 10%
    set_speed(i)
    time.sleep(1)  # Pausa de 1 segundo

# Detener el motor
set_speed(0)
```

En el código anterior:

- Se configura el PWM en el pin GPIO 15.

- La frecuencia se establece en 1000 Hz.
- La función set_speed(duty_cycle) ajusta la velocidad del motor según un ciclo de trabajo especificado en porcentaje (0-100%).

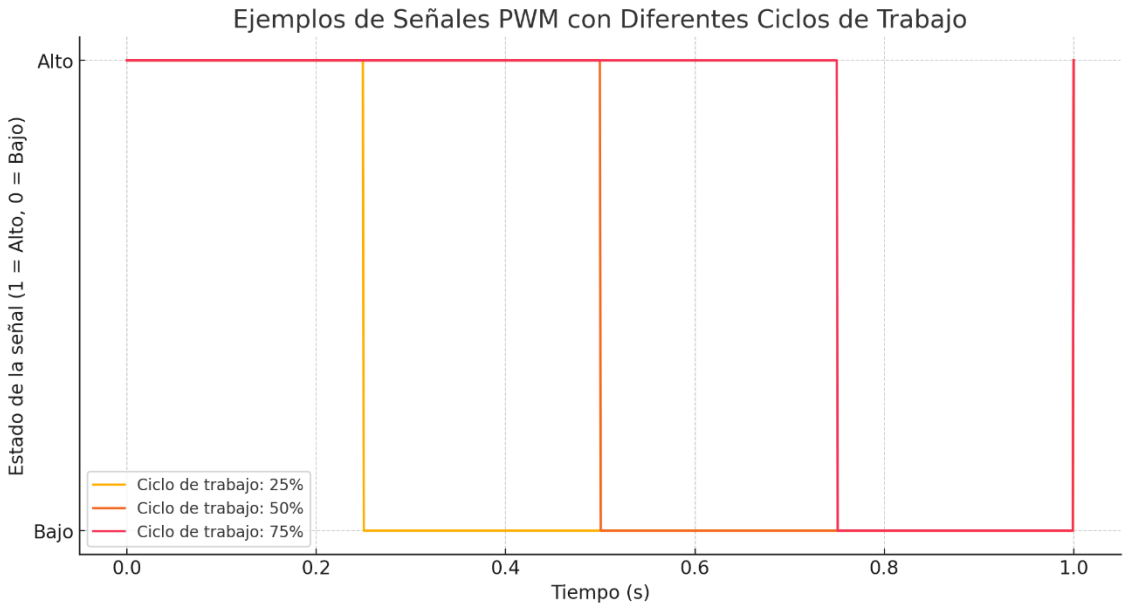
Gráfica de Señal PWM

A continuación, se presenta una gráfica que muestra cómo varía la señal PWM para diferentes ciclos de trabajo:

1. Ciclo de trabajo del 25%: La señal está en alto el 25% del tiempo y en bajo el 75% del tiempo.
2. Ciclo de trabajo del 50%: La señal está en alto el 50% del tiempo y en bajo el 50% del tiempo.
3. Ciclo de trabajo del 75%: La señal está en alto el 75% del tiempo y en bajo el 25% del tiempo.

Voy a generar una gráfica para visualizar estos ejemplos de PWM.

Ejemplos de Señales PWM con Diferentes Ciclos de Trabajo



La gráfica muestra cómo varían las señales PWM para diferentes ciclos de trabajo. A medida que el ciclo de trabajo aumenta, la cantidad de tiempo que la señal está en "alto" (encendido)

durante cada período también aumenta, lo que resulta en una mayor potencia entregada al motor y, por lo tanto, en una mayor velocidad.

Anexo II: Sockets en la Comunicación entre Software y Hardware

Los sockets son utilizados para gestionar la comunicación entre el microcontrolador (Raspberry Pi Pico W) y el navegador web del usuario, permitiendo el control remoto en tiempo real del vehículo. Los sockets TCP (Transmission Control Protocol) son empleados debido a su capacidad para establecer conexiones fiables y garantizar la entrega de datos de manera ordenada y sin pérdidas, lo cual es crucial para el control preciso del coche y la recepción continua de datos de los sensores.

Funcionamiento de los Sockets en este Proyecto

1. Creación del Socket y Configuración del Servidor:
    - El servidor se configura en la Raspberry Pi Pico W utilizando un socket TCP. Esto implica crear un socket que escucha en un puerto específico (el puerto 80 para HTTP), aceptando conexiones entrantes desde clientes (los navegadores web).

```
connection = socket.socket(socket.AF_INET, socket.SOCK_STREAM)  # Creación del socket TCP

connection.bind((ip, 80))  # Asignación de IP y puerto

connection.listen(1)  # Escuchar una conexión entrante
```

2. Recepción de Comandos desde el Navegador:
    - Cuando un usuario interactúa con la interfaz web, se envían solicitudes HTTP al servidor. El socket del servidor acepta estas conexiones y recibe los datos enviados. Estos datos contienen comandos, como avanzar o girar, que el microcontrolador procesa para controlar el coche.

```
cliente, addr = connection.accept()  # Aceptar la conexión entrante

peticion = cliente.recv(1024)  # Recibir datos del cliente
```

3. Envío de Datos de Sensores a la Web:

- Los datos obtenidos de los sensores (como la temperatura y la distancia a obstáculos) se envían de vuelta al navegador mediante el socket. El servidor construye una respuesta HTTP con estos datos, que se muestra en tiempo real en la interfaz web del usuario.

```
cliente.sendall(response)  # Enviar datos y respuesta al cliente
```

4. Cierre de la Conexión:
    - Una vez que la solicitud del usuario ha sido procesada y la respuesta ha sido enviada, la conexión se cierra. Esto libera los recursos del servidor y lo prepara para aceptar nuevas conexiones.

```
cliente.close()  # Cerrar la conexión
```

Conclusión

En este proyecto, los sockets TCP permiten la comunicación bidireccional eficiente entre el servidor en el microcontrolador y el cliente web. Esto facilita tanto el control remoto del coche teledirigido como la visualización de datos en tiempo real, proporcionando una experiencia de usuario robusta y dinámica.

Anexo III: Desglose de Componentes

Para el desarrollo de este proyecto, se han seleccionado componentes específicos que cumplen con los requisitos de funcionalidad, compatibilidad y costo. A continuación, se presentan los componentes elegidos junto con sus características técnicas y precios.

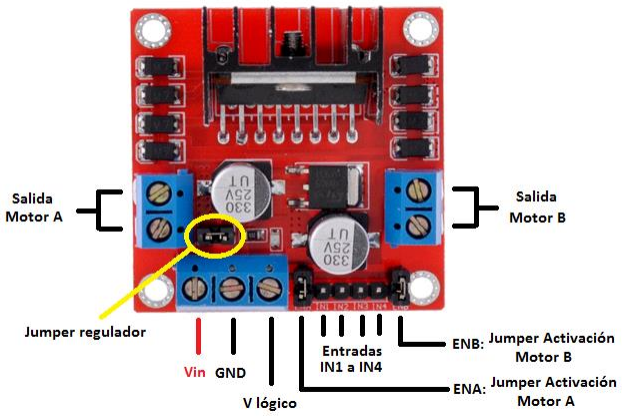
1. Sensor de Temperatura DS18B20

- Descripción: Sensor digital de temperatura.
- Características Técnicas:
  - Rango de medición: -55°C a +125°C
  - Precisión: ±0.5°C en el rango de -10°C a +85°C
  - Alimentación: 3.0V a 5.5V
  - Interfaz de comunicación: 1-Wire
- Precio: 2,50 €

2. Sensor de Ultrasonidos HC-SR04

- Descripción: Sensor de distancia por ultrasonidos.
- Características Técnicas:
  - Rango de medición: 2 cm a 400 cm
  - Precisión: ±3 mm
  - Ángulo de medición: <15°
  - Alimentación: 5V
- Precio: 3,00 €

3. Módulo de Control L298N



- Descripción: Módulo controlador de motores dual H-Bridge.
- Características Técnicas:
  - Voltaje de operación: 5V a 35V
  - Corriente máxima: 2A por canal
  - Dos canales de control para motores
  - Soporta control PWM
- Precio: 4,50 €

4. Motores 12V Arduino con Ruedas (4 unidades)

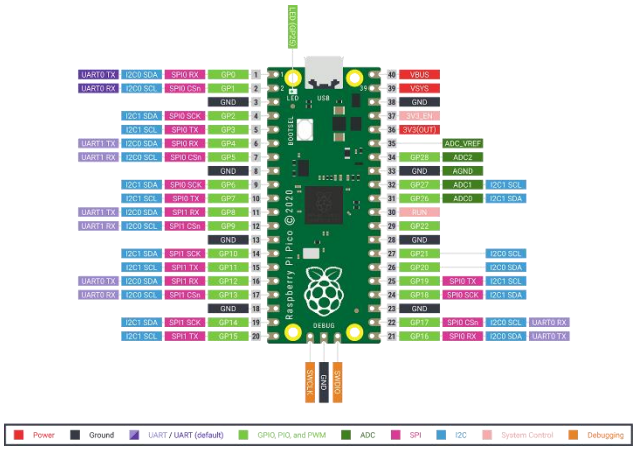
- Descripción: Motores DC con ruedas acopladas, aptos para aplicaciones de robótica.
- Características Técnicas:
  - Voltaje de operación: 12V
  - Velocidad: 200 RPM
  - Par: 1.2 kg·cm
  - Corriente máxima: 300 mA
- Precio por unidad: 6,00 €
- Precio total (4 unidades): 24,00 €

5. Batería de 12V

- Descripción: Batería recargable de 12V para alimentar los motores y otros componentes.
- Características Técnicas:
  - Voltaje nominal: 12V
  - Capacidad: 2000 mAh
  - Tipo: Litio-ion

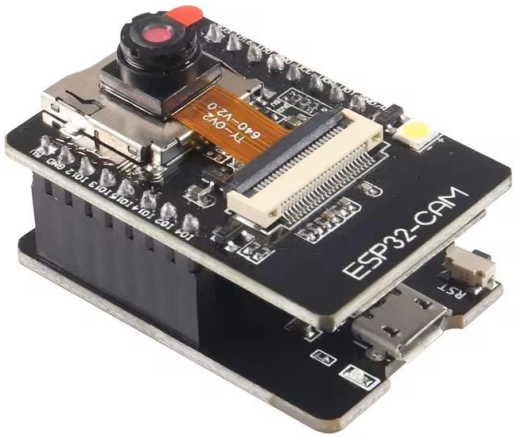
- Precio: 15,00 €

6. Raspberry Pi Pico W



- Descripción: Microcontrolador con conectividad inalámbrica WiFi.
- Características Técnicas:
  - Microprocesador: RP2040 dual-core ARM Cortex-M0+
  - Memoria RAM: 264 KB
  - Conectividad: WiFi
  - Pines GPIO: 26
- Precio: 7,00 €

7. Cámara ESP32



- Descripción: Módulo de cámara con capacidad de conexión WiFi.
- Características Técnicas:
  - Microprocesador: Tensilica Xtensa LX6 dual-core
  - Resolución de la cámara: 2MP
  - Conectividad: WiFi y Bluetooth
  - Pines GPIO: 34
- Precio: 8,50 €

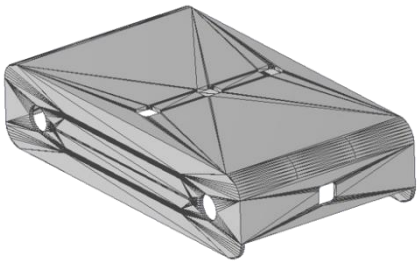
Tabla de Características Técnicas y Precios

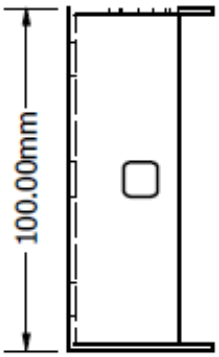
Componente	Características Principales	Precio (€)
Sensor de Temperatura DS18B20	Rango de -55°C a +125°C, Precisión ±0.5°C	2,50
Sensor de Ultrasonidos HC-SR04	Rango de 2 cm a 400 cm, Precisión ±3 mm	3,00
Módulo de Control L298N	Voltaje de 5V a 35V, 2A por canal	4,50
Motores 12V Arduino (4x)	Voltaje 12V, 200 RPM, Par 1.2 kg·cm	24,00
Batería de 12V	12V, 2000 mAh, Litio-ion	15,00
Raspberry Pi Pico W	RP2040, 264 KB RAM, WiFi	7,00
Cámara ESP32	Resolución 2MP, WiFi, Bluetooth	8,50

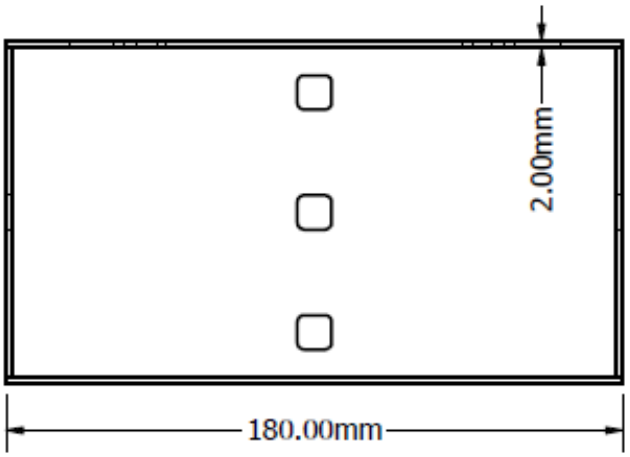
Precio Total: 64,50 €

Anexo IV: Diseño del Chasis en AutoCad

PARTS LIST		
ITEM	QTY	PART NUMBER
1	1	







DRAWN miguel Sancho-Tello	Date Printed 04/09/2024		
CHECKED		TITLE Robot Car Chassis	
QA			
MFG			
APPROVED			
		SIZE A	DWG NO robot_chassis / REV
		SCALE -2147483648 : 1	SHEET 1 OF 1

Anexo V: Detalles Técnicos de la Programación en MicroPython

El código del coche teledirigido en MicroPython se estructura de la siguiente manera:

Importación de Librerías y Configuración Inicial:

Se importan las librerías necesarias para el manejo de la red WiFi (network), la comunicación de sockets (socket), y el control de hardware (machine, PWM, onewire, ds18x20). Además, se configuran los pines para el sensor de temperatura DS18B20 utilizando el protocolo 1-Wire.

Control del Servomotor para la Cámara:

La cámara está montada en un servomotor que se controla mediante PWM para proporcionar un rango de visión ajustable. La función set_servo_angle_slowly() ajusta gradualmente el ángulo del servomotor, permitiendo movimientos suaves para cambiar la dirección de la cámara de manera precisa.

Control de Motores mediante PWM:

Los motores del coche se controlan a través de pines configurados con PWM, lo que permite un ajuste fino de la velocidad. Las funciones set_motor_speed(), forward(), backward(), left(), right(), y stop() permiten manejar la dirección y velocidad del coche, proporcionando un control completo del movimiento.

Conexión a la Red WiFi y Configuración del Servidor Web:

La función conectar() permite conectar el coche a una red WiFi utilizando las credenciales predefinidas. Una vez conectado, se establece un servidor web mediante la función open_socket(), que escucha solicitudes HTTP y permite el control remoto del coche.

Lectura de Sensores:

La función read_temperature() utiliza el sensor DS18B20 para obtener lecturas de temperatura, mientras que read_distance() usa el sensor ultrasónico HC-SR04 para medir distancias a obstáculos cercanos. Estos datos se actualizan dinámicamente en la interfaz web.

Servidor Web y Respuesta a Peticiones:

La función serve() maneja las solicitudes entrantes del servidor web, interpretando comandos para el control de la cámara y los motores, así como ajustes de velocidad. También se incluyen datos de los sensores en la respuesta HTML para proporcionar información en tiempo real al usuario.

Manejo de Errores y Reinicio:

El bloque try...except asegura que el sistema pueda recuperarse de errores inesperados o interrupciones, reiniciando el microcontrolador en caso de un error grave o de una interrupción manual.

Esta estructura modular y bien organizada del código permite un control eficiente del coche teledirigido, integrando múltiples funcionalidades de hardware y software que facilitan la interacción en tiempo real y la operación remota del sistema.

El Código al completo, tanto el utilizado en MicroPython como los utilizados para la interfaz web y la configuración de la cámara puede encontrarse en el siguiente repositorio: https://github.com/msanchotello/Robot_Pico_W.

Anexo VI: Configuración Técnica de la Cámara Web en ESP32-CAM

La configuración de la cámara web con la **ESP32-CAM** se realiza en varios pasos clave. A continuación, se describe en detalle cómo se configura y opera la cámara para transmitir video en tiempo real:

1. **Inicialización de la Cámara:**
    - La configuración de la cámara comienza con la inclusión de las librerías necesarias (esp_camera.h, WiFi.h, WebServer.h). Se definen los pines específicos utilizados por el modelo de cámara, en este caso, el **AI THINKER**. Luego, se inicializan las variables de conexión WiFi con el SSID y la contraseña correspondientes para conectar la ESP32 a la red.

```
#include "esp_camera.h"

#include <WiFi.h>

#include <WebServer.h>



#define CAMERA_MODEL_AI_THINKER

#include "camera_pins.h"
```

    - Después, se configura la cámara con la estructura camera_config_t y se establecen los pines de la cámara y la frecuencia de reloj del sensor. Dependiendo de si la PSRAM está disponible, se ajusta la resolución de la imagen y la calidad del JPEG.

```
camera_config_t config;

config.ledc_channel = LEDC_CHANNEL_0;

config.ledc_timer = LEDC_TIMER_0;
```

```
config.pin_d0 = Y2_GPIO_NUM;

config.pin_d1 = Y3_GPIO_NUM;

// ... otros pines ...

config.xclk_freq_hz = 20000000;

config.pixel_format = PIXFORMAT_JPEG;
```

2. **Configuración del Servidor Web:**
    - Se crea un servidor web utilizando la clase WebServer y se define el puerto de escucha (puerto 80). Los manejadores de rutas se configuran para responder a diferentes solicitudes HTTP, como la raíz del servidor (/), la transmisión de video (/stream), y las páginas de error.

cpp

Copiar código

```
WebServer server(80);

server.on("/", handleRoot);

server.on("/stream", HTTP_GET, handleJpgStream);

server.onNotFound(handleNotFound);
```

3. **Manejo de Peticiones y Transmisión de Imágenes:**
    - El manejador handleJpgStream() es responsable de transmitir el flujo de video en formato **multipart/x-mixed-replace**. Este formato permite enviar una secuencia de imágenes JPEG que el navegador del cliente interpreta como un flujo de video.

```
void handleJpgStream() {
```

```
server.setContentLength(CONTENT_LENGTH_UNKNOWN);
server.send(200, "multipart/x-mixed-replace; boundary=frame");
while (true) {
  camera_fb_t * fb = esp_camera_fb_get();
  if (!fb) {
    Serial.println("Camera capture failed");
    server.send(500, "text/plain", "Camera capture failed");
    return;
  }
  server.sendContent("--frame\r\n");
  server.sendContent("Content-Type: image/jpeg\r\n");
  server.sendContent("Content-Length: " + String(fb->len) + "\r\n\r\n");
  server.sendContent((const char*)fb->buf, fb->len);
  server.sendContent("\r\n");
  esp_camera_fb_return(fb);
  if (server.client().connected()) {
    delay(100);
  } else {
```

```
            break;
        }
    }
}
```

- Este código captura una imagen de la cámara utilizando esp_camera_fb_get(), envía los datos de la imagen como un frame JPEG a través de la conexión HTTP abierta, y luego libera el frame buffer con esp_camera_fb_return(fb).

4. **Inicialización y Conexión WiFi:**
    - En la función setup(), la ESP32 se conecta a la red WiFi utilizando las credenciales proporcionadas. Una vez conectada, se inicializa la cámara y se comienza a escuchar por conexiones entrantes al servidor web.

```
void setup() {
    Serial.begin(115200);
    WiFi.begin(ssid, password);
    while (WiFi.status() != WL_CONNECTED) {
        delay(500);
        Serial.print(".");
    }
    Serial.println("WiFi connected");
    // Inicialización de la cámara
    esp_camera_init(&config);
```

```
  // Configuración del servidor web

  server.begin();

  Serial.printf("Camera Ready! Use 'http://%s/stream' to connect\n", WiFi.localIP().toString().c_str());

}
```

La configuración de la cámara web en la **ESP32-CAM** permite transmitir video en tiempo real mediante un servidor HTTP, lo cual es esencial para la operación remota del coche teledirigido. Este enfoque proporciona una solución eficiente para la visualización en tiempo real, asegurando que los datos de video se transmitan de manera continua y con la latencia mínima posible.