

GRUPO 1

Proyecto AUTO TANGO 6

ELECTRÓNICA
MICROCONTROLADA

PROFESORES

MORALES, JORGE
VERA, GONZALO

ALUMNOS

- BIRGE, ADOLFO FEDERICO.
- CARUNCHIO, CARLOS JAVIER.
- CARRIZO, ESTEBAN DARÍO.
- FERREYRA, MARÍA LUCIANA.
- GUTIÉRREZ, EMMA VILMA.
- ROMERO, GISELA DE LOURDES.

ÍNDICE DE CONTENIDOS

1	INTRODUCCIÓN	3
1.1	FASES DEL PROYECTO	3
1.2	PLANTEAMIENTO DEL PROBLEMA	4
1.3.	PROPÓSITO DEL PROYECTO	5
1.5	JUSTIFICACIÓN	6
2	OBJETIVOS.....	6
2.1	OBJETIVO GENERAL:	6
2.2	OBJETIVOS ESPECÍFICOS:.....	7
3	METODOLOGÍA.....	7
3.1	PRESENTACIÓN DEL DIAGRAMA DE BLOQUES.....	7
3.2	ESTIMACIÓN DE LOS TIEMPOS.....	8
3.3	CRONOGRAMA:.....	8
3.4	DESCRIPCIÓN DE MATERIALES NECESARIOS	9
3.5	Material Usado en el armado del TANGO 6	9
3.6	TECNOLOGÍAS/HERRAMIENTAS/ SOFTWARE:.....	15
4	IMPLEMENTACIÓN DE LA ESTRUCTURA DEL AUTO 4WD	16
4.1	FUNCIONAMIENTO DE AUTO.....	17
4.2	DESARROLLO DE CÓDIGO PARA EL FUNCIONAMIENTO DEL AUTO	20
4.3	CÓDIGOS DE MOVIMIENTO.....	21
4.4	CÓDIGO DE DHT11	24
4.5	CÓDIGO PRUEBA DE DISTANCIA	25
4.6	CÓDIGO PRUEBAS- MOVIMIENTO Y CONTROL DE SERVO	26
4.7	CÓDIGO DE FUNCIONAMIENTO	27
4.8	ESQUEMA ELECTRÓNICO V1.0.....	34
5	MEJORAS O ACTUALIZACIONES Y CAMPO DE APLICACIÓN	34
6	RESULTADOS.....	43
7	CONCLUSIONES	45
8	BIBLIOGRAFÍA	46

TABLA DE ILUSTRACIONES

Ilustración 1- Diagrama de bloques	7
Ilustración 2- Diagrama de Gantt.....	8
Ilustración 3- ESP32	10
Ilustración 4 - HC-SR04.....	11
Ilustración 5 - Características HC-SR04.....	11
Ilustración 6 - DHT11	12
Ilustración 7 - Motor DC de 6V	13
Ilustración 8 - Driver puente H L298N	14
Ilustración 9 - Características driver puente H L298N	14
Ilustración 10 - SERVO SG90	15
Ilustración 11 - Kit Arduino + ESP32.....	16
Ilustración 12 - kit chasis auto.....	16
Ilustración 13 - Auto ARMADO.....	17
Ilustración 14 - App Blynk.....	18
Ilustración 15 - Pantalla de comandos App Blynk.....	18
Ilustración 16 - Página webs Blynk.....	19
Ilustración 17 - Interfaz App Blynk.....	19
Ilustración 18 - Interfaz Blynk App.....	20
Ilustración 19 - Diagrama Electrónico V2.0.....	34
Ilustración 20 - Módulo Sensor óptico TCRT5000.....	34
Ilustración 21 - Esquema de detección de línea	35
Ilustración 22 - ESP32 CAM	35
Ilustración 23 - Placa Raspberry pi.....	36
Ilustración 24 - Ejemplo de Conectividad – ESP32	37
Ilustración 25 - Nuestro Proyecto.....	37
Ilustración 26 - Ejemplo de ruedas	38
Ilustración 27 - Ejemplo de suspensión.....	38
Ilustración 28 - Ejemplo de motores	38
Ilustración 29 - Batería recargable AA de 1.2V.....	39
Ilustración 30 - Powerbanks 5V.....	39
Ilustración 31 - Baterías de Níquel – MetalHidruro	40
Ilustración 32 - Baterías de polímero de Litio (LiPo)	40
Ilustración 33 - Paneles Solares	41
Ilustración 34 - kit solar para construir.....	41
Ilustración 35 - Módulo GPS GY-NEO6MV2	41
Ilustración 36 - ESP32 WEB SERVER.....	42
Ilustración 37 - Sensor Profesional.....	42
Ilustración 38 – Auto Prototipo Tango 6	43
Ilustración 39- Auto Tango 6 Final	43
Ilustración 40 - Ficha de seguimiento 1.....	44
Ilustración 41 - Ficha de seguimiento 2.....	44
Ilustración 42 - Ficha de seguimiento 3.....	45

PROYECTO AUTO TANGO 6

AUTORES: Birge, Adolfo Federico. Carunchio, Carlos Javier. Carrizo, Esteban Darío. Ferreyra, María Luciana. Gutiérrez, Emma Vilma. Romero, Gisela de Lourdes.

Resumen

La finalidad del proyecto es la realización de un prototipo vehículo auto a escala de cuatro ruedas, implementando con un módulo de ESP32, controlado por mando a distancia (WIFI o tecnología Bluetooth), controlado con el sensor de obstáculos y un propósito específico, en este caso con un sensor de temperatura y humedad, que permita la obtención de parámetros del medio ambiente en un invernadero. Se realizará una introducción al proyecto, para luego proceder a la realización del mismo. Se trabajará, en principio, sobre la selección de los componentes necesarios para la realización del proyecto y sus respectivos costos, y luego se abordarán las formas de implementación, conexiones electrónicas, programación, interfaz de usuario y diseño mecánico de las distintas partes que lo componen. Al final, se mostrará el prototipo realizado, posibles mejoras o actualizaciones del prototipo y las conclusiones del proyecto.

Palabras clave: Circuito, comunicación serial, robot, sensor, tecnología.

Abstract

The purpose of the project is the realization of a four-wheeled prototype car, implemented with an ESP32 module, controlled by remote control (WIFI or Bluetooth technology), controlled with an obstacle sensor and a specific purpose, in this case with a temperature and humidity sensor, to obtain environmental parameters in a greenhouse. An introduction to the project will be given, and then we will proceed to the realization of the project. First, the selection of the necessary components for the realization of the project and their respective costs will be discussed, followed by the implementation, electronic connections, programming, user interface and mechanical design of the different parts that compose it. At the end, the realized prototype, possible improvements or updates of the prototype and the conclusions of the project will be shown.

Keywords: Circuit, serial communication, robot, sensor, technology.

1 INTRODUCCIÓN

El proyecto consiste en crear un robot automóvil económico de cuatro ruedas, que implementando un módulo de ESP32, controlado por mando a distancia (WIFI o tecnología Bluetooth), mediante la utilización de sensores: de obstáculos, sensor de temperatura y de humedad; y un mando por medio de una aplicación celular para movilizar, pueda dar información acerca de las propiedades del medio ambiente que lo rodea.

Este robot ahorrará la utilización de muchos artefactos de medición diferentes, teniendo varios de ellos en un solo producto y como futura actualización, con la información disponible de una manera cómoda y eficiente.

Elegimos empezar por la parte electrónica; todo lo relacionado a las conexiones eléctricas, alimentación, selección del controlador para el proyecto, conexión y elección de los sensores más eficientes y útiles a la hora de recolectar datos ambientales. Luego, seguiremos por el desarrollo de las partes del armazón del auto. El controlador será programado para una placa ESP32; se armará el código con las funciones de movimiento del auto.

Por último, se elegirá el protocolo de comunicación con el usuario y el modo de mando.

1.1 FASES DEL PROYECTO

El proyecto debe desarrollarse siguiendo una serie de pasos que sirven de sustento para agilidad y comodidad del mismo:

- Reunión del equipo de trabajo.
- Planteamiento de la problemática a solucionar.
- Presentación de un diagrama de bloques de la solución propuesta.
- Estimación de los tiempos en los que se llevará a cabo cada una de las fases del proyecto.
- Toma de mediciones.
- Graficar los datos obtenidos a partir de las mediciones.
- Análisis de los datos recolectados.
- Considerar que el proyecto tenga un componente de RSU.
- Informe del proyecto, conversión a artículo científico y presentación final de informe en formato de artículo científico (paper).

1.2 PLANTEAMIENTO DEL PROBLEMA

Control climático en invernaderos

Desde siempre, la agricultura se ha visto afectada por factores externos como lluvia, viento o granizo, que hasta hace poco no se podían controlar sino empleando técnicas de riego, fertilización o cultivo bajo abrigo. Para manejar el clima dentro de un invernadero debemos tener en cuenta parámetros tan importantes como la temperatura, la humedad, la radiación y la concentración de CO₂.

Es por ello por lo que hoy día se han desarrollado modernas técnicas para el control de clima que nos permiten depender cada vez menos de los factores climáticos y que favorecen que la rentabilidad de una explotación esté directamente relacionada con nuestro esfuerzo. (InterEmpresas, s.f.)

Factores a controlar y su importancia en la producción

Los factores climáticos que afectan a la producción agrícola son la luz, la temperatura, la humedad, la concentración de CO₂, el viento y la lluvia. Cuanto más control tengamos sobre ellos, el éxito y la seguridad del agricultor en su actividad productiva se verá incrementada.

En el interior de un invernadero, los factores climáticos afectan de diferente forma sobre el cultivo, de modo que los podemos controlar incidiendo en distintos puntos; La variación de parámetros tales como la luz, la temperatura o la concentración de CO₂ afecta de forma directa sobre la fotosíntesis de la planta, de modo que los procesos de respiración y división celular se ven alterados de algún modo. Por otro lado, con la aplicación de agua y de nutrientes podemos influir sobre la temperatura de las raíces y la humedad del aire, lo que implica una variación en la división y en el crecimiento celular.

Con el control de los factores climáticos se obtiene un incremento de la calidad y la producción, aumentando la rentabilidad del cultivo. Además, es posible adelantar la siembra y con ello la recolección, lo que significa que los productores pueden colocar sus productos en el mercado antes que otros, es decir, cuando los precios sean más favorables; también pueden producir en épocas extremas de frío y de calor, en las que mermaba la calidad y producción.

En definitiva, podrían ser más competitivos en el mercado actual con productos de alta calidad demandados en los mercados de destino, pues los productos que no cumplen con estas características verán dificultada su comercialización lo que dará lugar, finalmente, a la desaparición de muchos agricultores.

1.3. PROPÓSITO DEL PROYECTO

El Propósito del Proyecto es desarrollar un auto IoT que monitorea y controla el clima de manera inteligente, las temperaturas de un invernadero, manteniendo el cultivo en condiciones óptimas sin precisar una supervisión manual permanente. Por ejemplo, ser capaz de detectar en qué momento es necesario ventilar o activar el sistema de riego.

1.4. ANTECEDENTES

El término “Internet de las Cosas” (IoT) fue empleado por primera vez en 1999 por el pionero británico Kevin Ashton para describir un sistema en el cual los objetos del mundo físico se podían conectar a Internet por medio de sensores.¹

Ashton acuñó este término para ilustrar el poder de conectar a Internet las etiquetas de identificación por radiofrecuencia (RFID)² que se utilizaban en las cadenas de suministro corporativas para contar y realizar un seguimiento de las mercancías sin necesidad de intervención humana.

Hoy en día, el término Internet de las Cosas se ha popularizado para describir escenarios en los que la conectividad a Internet y la capacidad de cómputo se extienden a una variedad de objetos, dispositivos, sensores y artículos de uso diario.

Aunque el término “Internet de las Cosas” es relativamente nuevo, el concepto de combinar computadoras y redes para monitorear y controlar diferentes dispositivos ha existido durante décadas. Por ejemplo, a fines de la década de 1970 ya había en el mercado sistemas disponibles para monitorear los medidores conectados a la red eléctrica de forma remota a través de las líneas telefónicas.

En la década de 1990, los avances en la tecnología inalámbrica permitieron la difusión de soluciones corporativas e industriales “máquina a máquina” (M2M) para monitorear y operar diferentes equipos. Sin embargo, muchas de estas primeras soluciones M2M se basaban en redes dedicadas especialmente construidas para este propósito y en estándares propietarios o específicos de la industria, no en redes basadas en el Protocolo de Internet (IP) y los estándares de Internet. El uso del protocolo IP para conectar a Internet dispositivos que no son computadoras no es una idea nueva. El primer “dispositivo” para Internet —una tostadora conectada vía IP que se podía encender y apagar a través de Internet— se presentó en una conferencia sobre Internet realizada en 1990. Durante los años siguientes se fueron conectando otras “cosas” vía IP, entre ellas una máquina de refrescos en la Universidad Carnegie Mellon en Estados Unidos y una cafetera en el Trojan Room de la Universidad de Cambridge en el Reino Unido (que permaneció conectada a Internet hasta 2001). Luego de estos coloridos inicios, una

¹ (Society, s.f.)

² (Wikipedia “Radio-Frequency Identification.”, 2015)

robusta área de investigación y desarrollo dedicada a las “redes de objetos inteligentes” ayudó a sentar las bases de la Internet de las Cosas como la conocemos hoy. Si la idea de conectar objetos entre sí y a Internet no es nueva, es razonable preguntar por qué la Internet de las Cosas es un tema que hoy en día está ganando popularidad. (Report-InternetOfThings, s.f.)

1.5 JUSTIFICACIÓN

Los invernaderos llevan décadas siendo una pieza clave en la productividad del campo. Gracias a la conectividad IoT, los sensores y las plataformas de gestión Big Data, dicho rendimiento se ha disparado en los últimos años.

El sector agroalimentario ha pasado a ser objeto de especial atención en cuanto a la incorporación de tecnologías avanzadas, por lo tanto, se pretende que este proyecto aborde la realización de un prototipo de auto robot que sea capaz de tomar esas mediciones, por lo pronto de temperatura y humedad, evitando obstáculos. (Agro, 2021)

La agricultura en conexión, podría ayudar a resolver dos de los grandes desafíos del planeta: aumentar la producción de alimentos, al tiempo que se minimiza el impacto sobre el entorno.

La digitalización y la conectividad de los invernaderos, tiene dos grandes consecuencias: cultivar más y mejor, y minimizar el impacto de la agricultura en el medio ambiente. Así, evitan las condiciones que favorecen las potenciales afecciones de plagas y enfermedades fúngicas, evitando o minimizando la necesidad de utilizar productos fitosanitarios.

Por otro lado, el menor consumo de agua y energía, la aplicación precisa de fertilizantes (evitando la contaminación del suelo y los acuíferos), la disminución del uso de productos químicos, son también consecuencia directa de la agricultura Smart, permitiendo una producción más ecológica y sostenible desde el punto de vista medioambiental.

2 OBJETIVOS

2.1 OBJETIVO GENERAL:

El objetivo principal del proyecto es la implementación y control de manera directa de un automóvil a escala desde un dispositivo móvil. El microcontrolador será el encargado de recibir y enviar las órdenes de control a los componentes, actuando como centro de control del automóvil. Para implementar el envío y recepción de datos se hace uso un módulo bluetooth como intermediario entre el usuario y el controlador.

El proyecto se puede reducir a la siguiente lista de objetivos:

- Conocimiento y control de los componentes hardware implementados.
- Entender y conocer a fondo la funcionalidad del microcontrolador, estudiando las ventajas y desventajas que ofrece con respecto a otros posibles controladores.
- Control directo del vehículo desde nuestro Smartphone mediante tecnología bluetooth.

Funciones extra del auto robot:

El Modo automático esquiva obstáculos, mediante sensores controlando por mando a distancia (WIFI o tecnología Bluetooth), controlado con el sensor de obstáculos y un propósito específico, en este caso con un sensor de temperatura y humedad.

2.2 OBJETIVOS ESPECÍFICOS:

- Investigar los principios tecnológicos que conforman un automóvil autónomo y los tipos de algoritmos de una red neuronal.
- Diseñar los diferentes sistemas mecánicos y electrónicos que componen el AGV (del inglés Automated Guided Vehicle: Vehículo Autónomo Guiado).
- Construir el chasis que permita realizar las tareas solicitadas al AGV.
- Ajustar y configurar los diferentes elementos que conforman el AGV.
- Desarrollar algoritmos para la construcción de mapas de localización de ambientes agrícolas.
- Realizar pruebas de funcionamiento mecánico
- Realizar pruebas del sistema de la red, aprendizaje profundo.

3 METODOLOGÍA

3.1 PRESENTACIÓN DEL DIAGRAMA DE BLOQUES

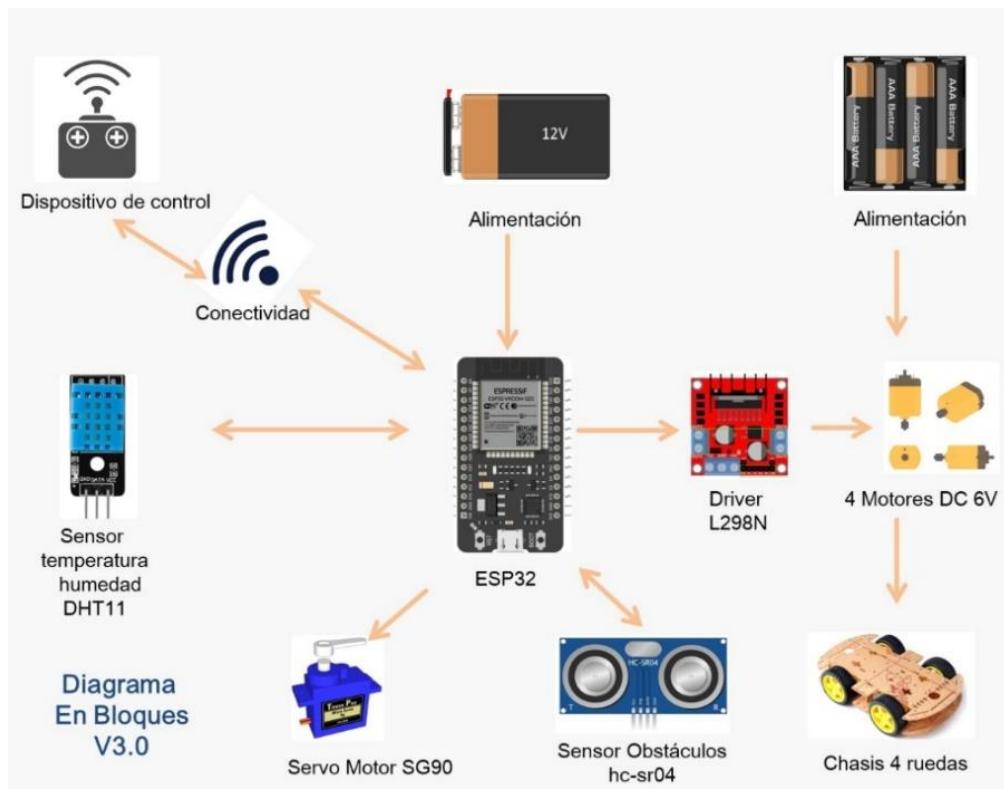


Ilustración 1- Diagrama de bloques

3.2 ESTIMACIÓN DE LOS TIEMPOS

DIAGRAMA DE GANTT

El **diagrama de Gantt** es una herramienta gráfica cuyo objetivo es exponer el tiempo de dedicación previsto para diferentes tareas o actividades a lo largo de un tiempo total, en pos de realizar el proyecto en el tiempo solicitado por los profesores de la cátedra hemos dispuesto el siguiente desarrollo:

En algunos casos hubo unas pequeñas variaciones de tiempo, pero se logró cumplir con lo planteado grupalmente.

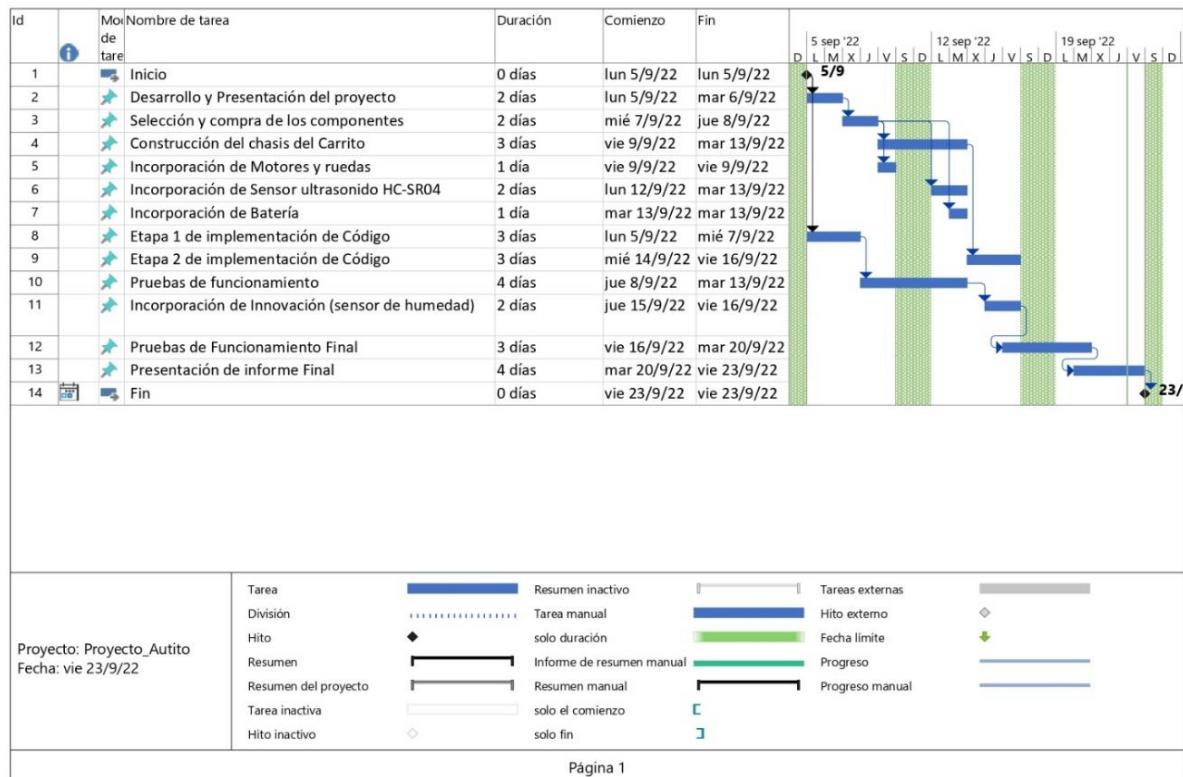


Ilustración 2- Diagrama de Gantt

3.3 CRONOGRAMA:

Desarrollo y Presentación del proyecto	2 días
Selección y compra de los componentes	2 días
Construcción del chasis del Carrito	3 días
Incorporación de Motores y ruedas	1 día

Incorporación de Sensor ultrasonido HC-SR04	2 días
Incorporación de Batería	1 día
Etapa 1 de implementación de Código	3 días
Etapa 2 de implementación de Código	3 días
Pruebas de funcionamiento	4 días
Incorporación de Innovación (sensor de humedad)	2 días
Pruebas de Funcionamiento Final	2 días
Presentación de informe Final	4 días

DURACIÓN APROXIMADA DEL PROYECTO 15 DÍAS.

3.4 DESCRIPCIÓN DE MATERIALES NECESARIOS

Componentes necesarios:

- Módulo ESP32.
- Alimentación ESP32 (no definida).
- Chasis 4WD con 4 motores DC de 6V, con sus cajas reductoras. Rack de 4 pilas.
- Módulo L298N controlador de motores.
- Motor servo SG90. - Sensor ultrasonido HC-SR04.
- Sensor Temperatura y Humedad DHT11.
- Dispositivo de control (Teléfono Celular).

3.5 Material Usado en el armado del TANGO 6

ESP32 Wifi & Bluetooth SoC Module Creado por Espressif Systems, ESP32 es un sistema de bajo consumo y bajo costo en un chip SoC (System On Chip) con Wi-Fi y modo dual con Bluetooth. En el fondo, hay un microprocesador TensilicaXtensaLX6 de doble núcleo o de un solo núcleo con una frecuencia de reloj de hasta 240MHz. ESP32 está altamente integrado con switch de antena, balun para RF, amplificador de potencia, amplificador de recepción con bajo nivel de ruido, filtros y módulos de administración de energía, totalmente integrados dentro del mismo chip. Diseñado para dispositivos móviles; tanto en las aplicaciones de electrónica, y las de IoT (Internet de las cosas), ESP32 logran un consumo de energía ultra bajo a través de funciones de ahorro de energía. Incluye la sintonización de reloj con una resolución fina, modos de potencia múltiple y escalado de potencia dinámica.

Características principales:

- Procesador principal: Tensilica Xtensa LX6 de 32 bits.
- Wi-Fi: 802.11 b / g / n / e / i (802.11n @ 2.4 GHz hasta 150 Mbit / s).
- Bluetooth: v4.2 BR / EDR y Bluetooth Low Energy (BLE).
- Frecuencia de Clock: Programable, hasta 240MHz. Rendimiento: hasta 600DMIPS.
- ROM: 448KB, para arranque y funciones básicas.
- SRAM: 520KiB, para datos e instrucciones.

FUNCIÓN EN TANGO 6: El modulo será el que programemos para poder dar órdenes de movimiento, para que los sensores reconozcan obstáculos, y para poder hacer las conexiones, mediante el bluetooth, a la app del celular para poder manejar al auto.



Ilustración 3- ESP32

HC-SR04

Sensor de distancia

HC-SR04 es un sensor de distancia de baja precisión basado en ultrasonidos. Con él permite medir distancias de una forma sencilla y rápida, aunque en principio no se suele usar para eso. Lo más frecuente es que se utilice como un transductor para detectar obstáculos y poderlos evitar mediante otros mecanismos asociados a la respuesta del sensor. El aspecto del HC-SR04 es muy característico y se reconoce con facilidad porque tiene dos «ojos» que realmente son los dispositivos de ultrasonidos que integra este módulo. Uno de ellos es un emisor de ultrasonidos y el otro un receptor. Trabaja a una frecuencia de 40 KHz, por tanto, es inaudible para los seres humanos.

FUNCIÓN EN TANGO 6: el HC-SR04, el emisor emitirá ultrasonidos y cuando reboten en un objeto u obstáculo que se encuentre en el camino serán captados por el receptor. El circuito se encargará de hacer los cálculos necesarios de ese eco para determinar la distancia.



Ilustración 4 - HC-SR04

Características	
Alimentación	+5v DC
Frecuencia de trabajo	40 KHz
Consumo (suspendido)	< 2mA
Consumo (trabajando)	15mA
Ángulo efectivo	< 15°
Distancia	2cm a 400cm *
Resolución	0.3 cm

*A partir de 250cm la resolución no es buena

Ilustración 5 - Características HC-SR04

DHT11

Sensor digital de temperatura y humedad

El DHT11 es un sensor digital de temperatura y humedad relativa de bajo costo y fácil uso. Utiliza un sensor capacitivo de humedad y un termistor para medir el aire circundante, y muestra los datos mediante una señal digital en el pin de datos (no posee salida analógica). Es bastante simple de usar tanto en hardware como software. El único inconveniente de este sensor es que sólo se puede obtener nuevos datos una vez cada 2 segundos.

El sensor DHT11 se caracteriza por tener la señal digital calibrada, asegurando alta estabilidad y fiabilidad a lo largo del tiempo. El sensor integra sensores resistivos para temperatura (termistor) y otro para humedad. Puede medir la humedad en un rango desde 20% hasta 90% y temperatura en el rango de 0°C a 50°C. Cada sensor DHT11 está estrictamente calibrado en laboratorio, presentando una extrema precisión en la calibración. Los coeficientes de calibración se almacenan como programas en la memoria OTP, que son empleados por el proceso de detección de señal interna del sensor.

El protocolo de comunicación emplea un único hilo o cable, por lo tanto, hace que la integración de este sensor en nuestros proyectos sea rápida y sencilla. En comparación con el DHT22, este sensor es menos preciso, menos exacto y funciona en un rango más pequeño de temperatura / humedad, pero su empaque es más pequeño y de menor costo.

ESPECIFICACIONES TÉCNICAS

- Voltaje de Operación: 3V - 5V DC
- Rango de medición de temperatura: 0 a 50 °C
- Precisión de medición de temperatura: ± 2.0 °C
- Resolución Temperatura: 0.1°C
- Rango de medición de humedad: 20% a 90% RH.
- Precisión de medición de humedad: 4% RH.
- Resolución Humedad: 1% RH
- Tiempo de sensado: 2 seg.

Interface: Digital Serial PINES

1. Alimentación: +5V (VCC)
2. Datos
3. No Usado (NC)
4. Tierra (GND)

FUNCIÓN EN EL TANGO 6: Hará las mediciones de temperatura y humedad, lo que hace que no provoque recalentamientos en el dispositivo.

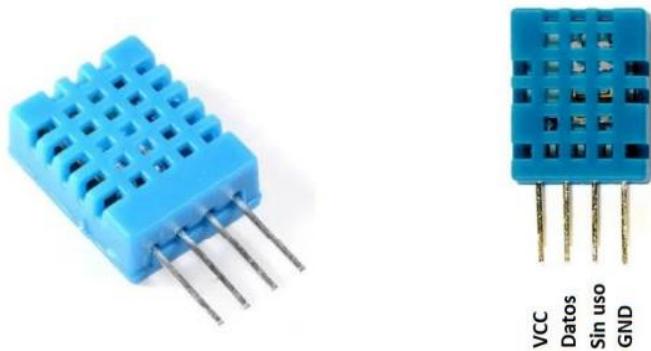


Ilustración 6 - DHT11

Motor DC de 6V

Este motor DC posee una caja reductora integrada que le permite entregar un buen torque en un pequeño tamaño y bajo voltaje. La carcasa del motor es de plástico resistente, no toxico y de color amarillo.

ESPECIFICACIONES TÉCNICAS

- Voltaje de Operación: 3V – 6V
- Velocidad Angular nominal: 200 RPM
- Reducción: 48:1
- Consumo de corriente sin carga: 100mA (a 5V)
- Consumo de corriente nominal: 140mA (a 5V)
- Consumo de corriente eje detenido: 500mA (a 5V)
- Diámetro eje para rueda: 55 mm Peso: 40g

FUNCIÓN EN TANGO 6: Este dispositivo electromecánico nos permitirá reducir la velocidad del motor, traduciéndolo a una velocidad menor, pero con una fuerza mecánica mayor.



Ilustración 7 - Motor DC de 6V

El driver puente H L298N

El driver puente H L298N es el modulo más utilizado para manejar motores DC de hasta 2 amperios. El chip L298N internamente posee dos puentes H completos que permiten controlar 2 motores DC o un motor paso a paso bipolar/unipolar. El módulo permite controlar el sentido y velocidad de giro de motores mediante señales TTL que se pueden obtener de microcontroladores y tarjetas de desarrollo como Arduino, Raspberry Pi o Launchpadsde Texas Instruments. El control del sentido de giro se realiza mediante dos pines para cada motor, la velocidad de giro se puede regular haciendo uso de modulación por ancho de pulso (PWM por sus siglas en inglés). Tiene integrado un regulador de voltaje LM7805 de 5V encargado de alimentar la parte lógica del L298N, el uso de este regulador se hace a través de un Jumper y se puede usar para alimentar la etapa de control.

ESPECIFICACIONES TÉCNICAS

- Chip: L298N
- Canales: 2 (soporta 2 motores DC o 1 motor PAP)

- Voltaje lógico: 5V
- Voltaje de potencia (V motor): 5V - 35V DC
- Consumo de corriente (lógico): 0 a 36mA
- Capacidad de corriente: 2A (picos de hasta 3A)
- Potencia máxima: 25W
- Dimensiones: 43 * 43 * 27 mm
- Peso: 30g

FUNCIÓN EN TANGO 6: Con el voltaje que posee nos permite alimentar la parte del control de modulo (ESP32).

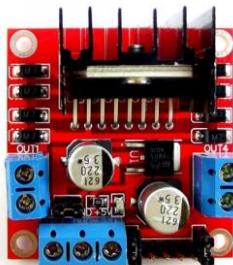


Ilustración 8 - Driver puente H L298N

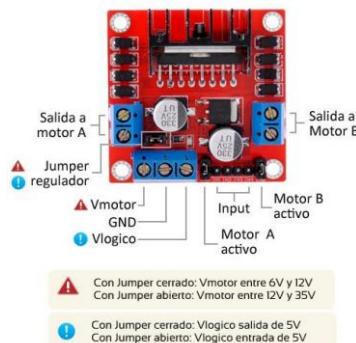


Ilustración 9 - Características driver puente H L298N

SERVO SG90

Servomotor de tamaño pequeño ideal para proyectos de bajo torque y donde se requiera poco peso. Muy usado en aeromodelismo, pequeños brazos robóticos y mini artrópodos. Un servo ideal para aprender a programar en Arduino. Puede rotar aproximadamente 180 grados (90° en cada dirección). Tiene la facilidad de poder trabajar con diversidad de plataformas de desarrollo como Arduino, PICs, Raspberry Pi, o en general a cualquier microcontrolador.

Para su uso con Arduino, recomendamos conectar el cable naranja al pin 9 o 10 y usar la Librería "Servo" incluida en el IDE de Arduino. Para la posición 0° el pulso es de 0.6ms, para 90° es de 1.5ms y para 180° 2.4ms. Posee un conector universal tipo "S" que encaja perfectamente en la

mayoría de los receptores de radio control incluyendo los Futaba, JR, GWS, Cirrus, Hitecy otros. Los cables en el conector están distribuidos de la siguiente forma: Café = Tierra (GND), Rojo = VCC (5V), Naranja = Señal de control (PWM).

ESPECIFICACIONES TÉCNICAS

- Voltaje de alimentación: 3.0 - 7.2V DC
- Velocidad: 0.1seg / 60 grados
- Torque reposo: 1.3Kg.cm (4.8V), 1.6Kg.cm (6.0V)
- Ancho de pulso: 4useg (Deadband)
- Engranajes: Nylon
- Longitud del conductor: 150mm
- Dimensiones: 22*11.5*27 mm
- Peso: 9g

FUNCIÓN EN TANGO 6: será la conexión con el sensor de movimiento permitiendo a este detectar dentro de un radio de 180º cualquier estructura y evitar que el auto choque.



Ilustración 10 - SERVO SG90

3.6 TECNOLOGÍAS/HERRAMIENTAS/ SOFTWARE:

- Microsoft Office
- VSCode
- PlatformIO
- Soldador
- App Móvil, para control del Auto(Blynk)
- Adobe Suite
- Corel Draw
- Canva online

4 IMPLEMENTACIÓN DE LA ESTRUCTURA DEL AUTO 4WD

El kit resulta en una mayor comodidad a la hora de colocar todos los componentes mediante una distribución que aproveche el espacio útil y además permita la manipulación de dichos componentes. A continuación, se expone una breve lista con los componentes del kit.



Ilustración 11 - Kit Arduino + ESP32



Ilustración 12 - kit chasis auto

Componentes del kit:

- 2 x chasis del automóvil inteligente (metacrilato)
- 4 x Ruedas de dirección
- 1 x compartimento de las pilas 2 (18650 Batería de litio recargable 3,7 V)
- 4 x encoders de velocidad
- 4 x Motorreductores o motores DC 3 ~ 12V (120RPM con tensión de 3V)
- Tornillería
- Kit Arduino uno + Módulo ESP32

4.1 FUNCIONAMIENTO DE AUTO

Nuestro **TANGO 6**, auto robot de 4 ruedas y 4 motores con tracción en todos, alimentados estos últimos con batería recargable de 12v (están colocadas entre chasis), y controlados por un drive L298N implementado el proyecto con una placa de desarrollo ESP32, en este caso NodeMCU de 30 pines, que controla propiamente los 4 motores a través del puente L298N, un servo motor SG90 con posición fija en centro(90°) haciendo ángulos de + - 45° según hacia donde gire el robot(izquierda o derecha) acompañando el movimiento, el mismo tiene montada una estructura para sostener un sensor de ultrasonido, en este caso el modelo HC-SR04, que cumple la función de detectar y hacer evitar obstáculos al robot.

Para ésta demostración, como el auto es controlado manualmente, la actitud evasiva es frenarse al encontrar un objeto a menos de 30 cm, detenerse, retroceder medio segundo de tiempo, y esperar detenido el próximo comando de quien lo opera. Para distinguir éste accionar (el sensor detecta un objeto) se hace encender el led azul integrado en la ESP32.

Además, el auto robot para cumplir el propósito específico, cuenta con un sensor de temperatura y humedad, modelo DHT11, que tampoco se ve, ya que se encuentra entre chasis.

Este último, al igual que la placa ESP32, el motor servo y el sensor de ultrasonido están alimentados con un rack de 4 pilas AA de 1.5v.

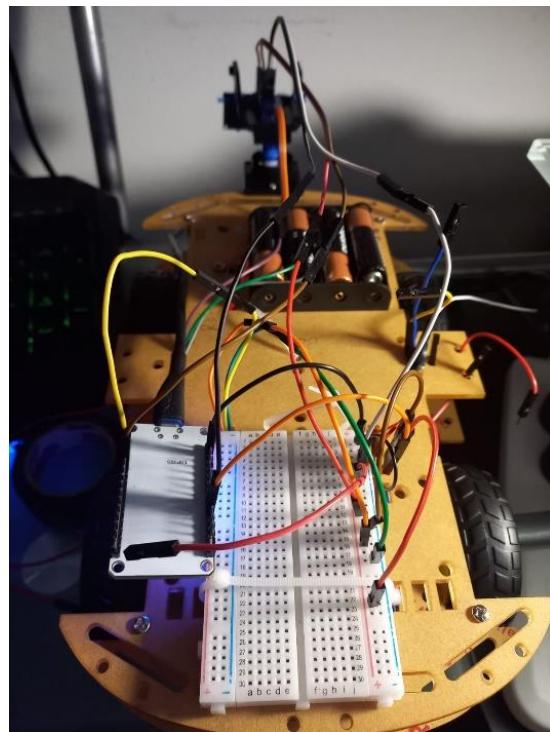


Ilustración 13 - Auto ARMADO

Para comandar el auto robot, se eligió Blynk, una plataforma de internet de las cosas (IoT) de marca blanca que ofrece software, firmware, soluciones web y aplicaciones móviles a miles de pequeñas, medianas y grandes empresas de todo el mundo y es muy popular. Sirve para conectar dispositivos a la nube, diseñar aplicaciones para controlarlos y supervisarlos de forma remota, y administrar miles de productos implementados.

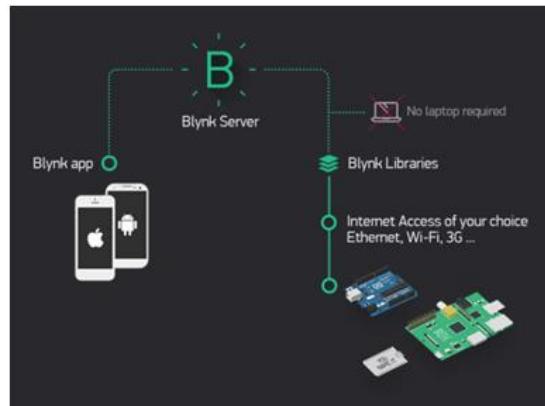


Ilustración 14 - App Blynk

En nuestro caso, utilizamos la App Móvil para simular un joystick de 4 botones (Adelante, Atrás, Derecha, Izquierda) que se encarga de dirigir los movimientos del auto, a través de una conexión WIFI inicial, que como se ve en lo últimos segundos, se desconecta la señal WIFI de Móvil, y el auto sigue conectado al mismo quedando establecida una conexión temporal entre ambos dispositivos.



Ilustración 15 - Pantalla de comandos App Blynk

La utilización de la plataforma es muy sencilla, se descarga la App móvil al teléfono, y también se puede utilizar la consola a través de su página web: <https://blynk.io/>.

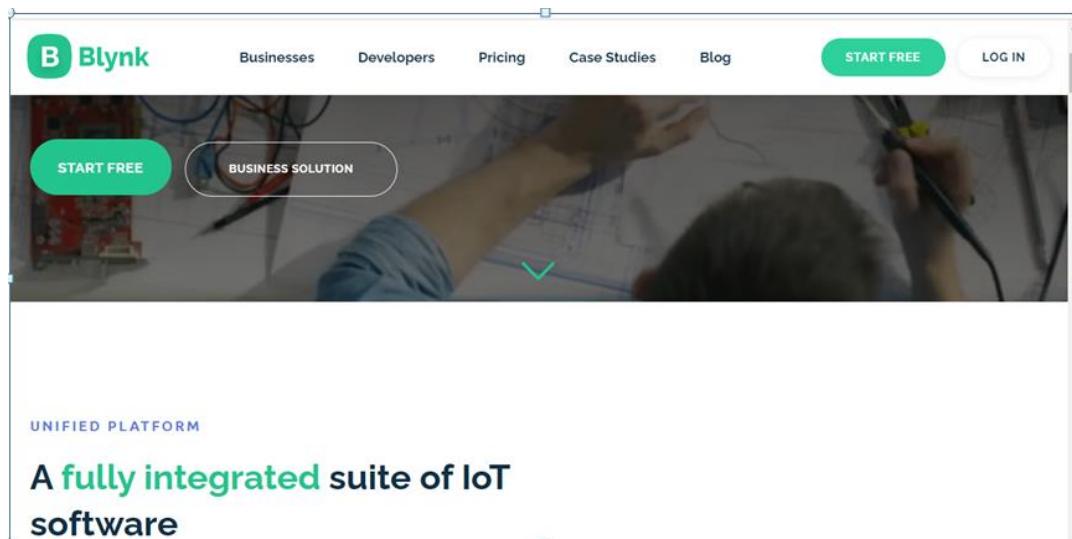


Ilustración 16 - Página webs Blynk

Desde la aplicación móvil se puede crear un usuario e ingresar luego de un mail de confirmación, y en la versión gratuita se puede utilizar hasta dos placas de desarrollo, se crea un proyecto eligiendo la placa y tipo de conexión.

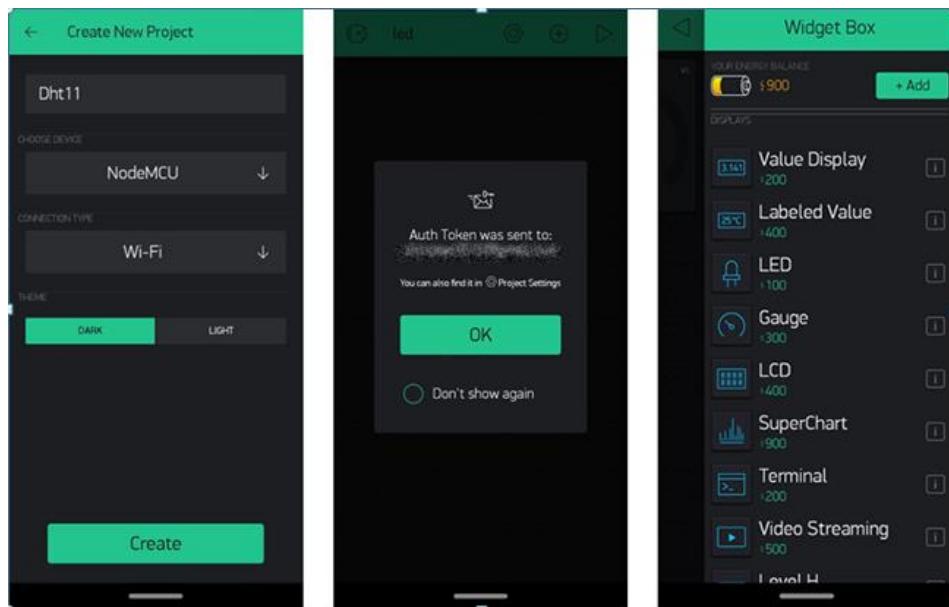


Ilustración 17 - Interfaz App Blynk

Se genera un Token, que actualmente se envía por mail, que junto con el nombre del proyecto y un ID del mismo, que servirá para la implementación del código donde se podrá establecer la conexión entre la placa y la plataforma.

Para ello existe una librería llamada Blynk, en nuestro caso incluiremos en nuestro código la sentencia: #Include <BlynkSimpleEsp32.h>. Tanto disponible en Arduino IDE como en PlatformIO.

En el modo de desarrollador podemos realizar nuestra App de propósito.

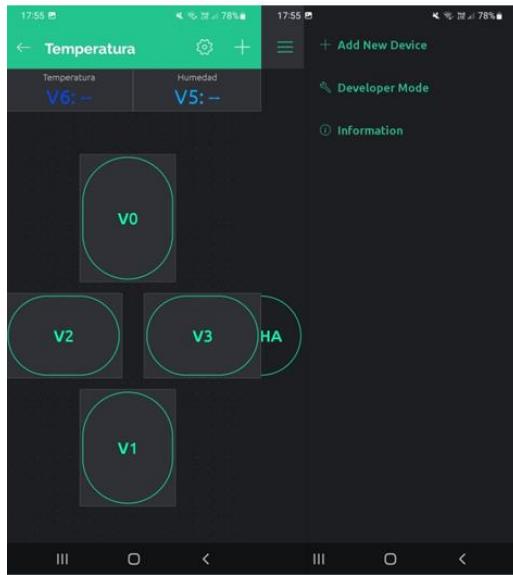


Ilustración 18 - Interfaz Blynk App

Como se observa en la imagen de arriba las variables V1, V2, V3 Y V4 representan pines “virtuales” los cuales pueden ser asociados correctamente dentro de la implementación de código a pines propios de la placa, o a cualquier dato, valor o registro que la misma pueda parametrizar, y así ser enviado a la nube propia de Blynk y ser visualizada en su app.

También en la imagen se ven definidos los pines virtuales V5 y V6 que representan la humedad y temperatura respectivamente, valores que son tomados por el sensor DHT11, que a través de la placa, el uso de la librería de Blynk, y el desarrollo correcto para obtener esos valores, se los puede observar en la App móvil instantáneamente.

Se puede ver en el video demostrativo (compartido en plataforma GitHub) que la humedad comienza el 20%, baja a 18% y termina en 32%. La temperatura no varió por estar dentro del mismo espacio reducido.

Cumpliendo así con el proyecto del auto robot, transformándose en un termómetro móvil.

4.2 DESARROLLO DE CÓDIGO PARA EL FUNCIONAMIENTO DEL AUTO

Desarrollo de código para el funcionamiento del auto robot controlar sus motores mediante una app móvil que se conecta por WIFI y un servo motor que acompaña los movimientos de giro, configurar el sensor de obstáculos, y el sensor de temperatura y humedad.

Librerías utilizadas (con sintaxis del código) y breve explicación:

```
#include <WiFi.h>
```

Librería para la conectividad Wifi entre la placa ESP32 y el teléfono móvil.

```
#include <WiFiClient.h>
```

Librería para establecer la conexión Wifi.

```
#include <DHT.h>
```

Librería del sensor de temperatura y humedad, modelo DHT11.

```
#include <DHT_U.h>
```

Librería para funcionalidades del sensor.

```
#include <NewPing.h>
```

Librería que facilita la codificación para el uso del sensor de ultrasonido, en este caso el modelo HC SR-04.

```
#include <ESP32Servo.h>
```

Librería para controlar funcionamiento del servo motor modelo SG90

```
#include <BlynkSimpleEsp32.h>
```

Librería para gestionar la conexión entre la placa ESP32 y el teléfono móvil, a través de la app que ofrece la plataforma Blynk, que nos permite controlar los movimientos del robot y además recibir los parámetros de lectura del sensor DHT11 y verlos de manera casi instantánea.

4.3 CÓDIGOS DE MOVIMIENTO

```
#include  
<Arduino.h>
```

```
#define LED 2
```

```
// Motor A  
int motor1Pin1 = 27;  
int motor1Pin2 = 26;  
int enable1Pin = 14;  
// Motor B  
int motor2Pin1 = 4;  
int motor2Pin2 = 2;  
int enable2Pin = 15;
```

```

// Configurar las propiedades PWM (Muy importante)
const int freq = 30000;
const int pwmChannel = 0;
const int resolution = 8;
int dutyCycle = 255;
void setup()
{
    // sets the pins as outputs:
    pinMode(motor1Pin1, OUTPUT);
    pinMode(motor1Pin2, OUTPUT);
    pinMode(enable1Pin, OUTPUT);
    pinMode(motor2Pin1, OUTPUT);
    pinMode(motor2Pin2, OUTPUT);
    pinMode(enable2Pin, OUTPUT);
    pinMode(LED, OUTPUT);

    // configurar funcionalidades LED PWM
    ledcSetup(pwmChannel, freq, resolution);

    // adjunte el canal al GPIO para ser controlado
    ledcAttachPin(enable1Pin, pwmChannel);
    ledcAttachPin(enable2Pin, pwmChannel);

    Serial.begin(9600);

    // testing
    Serial.print("Testeando motores...");
    ledcWrite(pwmChannel, dutyCycle);
}

void loop()

// Mueva el motor de CC hacia adelante a la velocidad máxima
Serial.println("Moviendo hacia adelante");
digitalWrite(motor1Pin1, HIGH);
digitalWrite(motor1Pin2, LOW);
digitalWrite(motor2Pin1, HIGH);

```

```

digitalWrite(motor2Pin2, LOW);
digitalWrite(LED, HIGH);
delay(2000);

// Detengar los motores 1 segundo
Serial.println("Motores detenidos");
digitalWrite(motor1Pin1, LOW);
digitalWrite(motor1Pin2, LOW);
digitalWrite(motor2Pin1, LOW);
digitalWrite(motor2Pin2, LOW);
digitalWrite(LED, LOW);
delay(1000);

// Mueva el motor de CC hacia atrás a la velocidad máxima
Serial.println("Moviendo hacia atras");
digitalWrite(motor1Pin1, LOW);
digitalWrite(motor1Pin2, HIGH);
digitalWrite(motor2Pin1, LOW);
digitalWrite(motor2Pin2, HIGH);
digitalWrite(LED, HIGH);
delay(2000);

// Detengar los motores 1 segundo
Serial.println("Motores detenidos");
digitalWrite(motor1Pin1, LOW);
digitalWrite(motor1Pin2, LOW);
digitalWrite(motor2Pin1, LOW);
digitalWrite(motor2Pin2, LOW);
digitalWrite(LED, LOW);
delay(1000);

// Mover motores hacia adelante con velocidad en aumento (anda)
digitalWrite(motor1Pin1, HIGH);
digitalWrite(motor1Pin2, LOW);
digitalWrite(motor2Pin1, HIGH);
digitalWrite(motor2Pin2, LOW);
dutyCycle = 200;
while (dutyCycle <= 255)
{
    ledcWrite(pwmChannel, dutyCycle);
    Serial.print("Avanzar con ciclo de trabajo");
    Serial.println(dutyCycle);
}

```

```

    dutyCycle = dutyCycle + 5;
    delay(500);
}
dutyCycle = 255;
}

```

4.4 CÓDIGO DE DHT11

```

// Incluimos
librería
#include <Adafruit_Sensor.h>
#include <DHT.h>
#include <DHT_U.h>

// Definimos el pin digital donde se conecta el sensor
#define DHTPIN 13
// Dependiendo del tipo de sensor
#define DHTTYPE DHT11

// Inicializamos el sensor DHT11
DHT dht(DHTPIN, DHTTYPE);

void setup()
{
    // Inicializamos comunicación serie
    Serial.begin(9600);

    // Comenzamos el sensor DHT
    dht.begin();
}

void loop()
{
    // Esperamos 5 segundos entre medidas
    delay(5000);

    // Leemos la humedad relativa
    float h = dht.readHumidity();
    // Leemos la temperatura en grados centígrados (por defecto)
    float t = dht.readTemperature();
}

```

```

// Leemos la temperatura en grados Fahrenheit
float f = dht.readTemperature(true);

// Comprobamos si ha habido algún error en la lectura
if (isnan(h) || isnan(t) || isnan(f))
{
    Serial.println("Error obteniendo los datos del sensor DHT11");
    return;
}

// Calcular el índice de calor en Fahrenheit
float hif = dht.computeHeatIndex(f, h);
// Calcular el índice de calor en grados centígrados
float hic = dht.computeHeatIndex(t, h, false);

Serial.print("Humedad: ");
Serial.print(h);
Serial.print(" %\t");
Serial.print("Temperatura: ");
Serial.print(t);
Serial.println(" °C ");
Serial.print(f);
Serial.print(" *F\t");
Serial.print(" Índice de calor: ");
Serial.print(hic);
Serial.print(" *C ");
Serial.print(hif);
Serial.println(" *F");
}

```

4.5 CÓDIGO PRUEBA DE DISTANCIA

```

#include
<NewPing.h>
//aqui hay que definir Trig y Echo
#define trigPin 18
#define echoPin 19

//Definir la distancia máxima en centímetros en 400-500
#define MAX_DISTANCE 400

```

```

//Crear objeto llamado Sonar.
NewPing Sonar(trigPin, echoPin, MAX_DISTANCE);

void setup() {
  Serial.begin(9600);
}
void loop() {
  // Un delay de 50 ms entre pings (aprox 20 pings / seg). 30ms es el retardo minimo entre pings:
  delay(500);
  // crear variable y pasarle en valor de Sonar.ping_cm
  int distancia = Sonar.ping_cm();
  // Enviar ping, obtener la distancia en cm e imprimir el resultado
  Serial.println("Distancia = " + String(distancia)+ " CM");
}

```

4.6 CÓDIGO PRUEBAS- MOVIMIENTO Y CONTROL DE SERVO

```

// Incluimos
la librería
para poder
controlar el
servo

```

```

#include <ESP32Servo.h>
// Declaramos la variable para controlar el servo
Servo servoMotor;
void setup() {
  // Iniciamos el monitor serie para mostrar el resultado
  Serial.begin(9600);
  // Iniciamos el servo para que empiece a trabajar con el pin 9
  servoMotor.attach(13);
  // Inicializamos al ángulo 0 el servomotor
  servoMotor.write(0);
}
void loop() {
  // Vamos a tener dos bucles uno para mover en sentido positivo y otro en sentido negativo

  // Para el sentido positivo
  for (int i = 0; i <= 180; i++)
  {
    // Desplazamos al ángulo correspondiente
    servoMotor.write(i);
  }
}

```

```

    // Hacemos una pausa de 25ms
    delay(25);
}

// Para el sentido negativo
for (int i = 179; i > 0; i--)

```

4.7 CÓDIGO DE FUNCIONAMIENTO

```

#define BLYNK_TEMPLATE_ID "TU ID DEL DISPOSITIVO"           // Id del dispositivo en app móvil
Blynk

#define BLYNK_DEVICE_NAME "NOMBRE DE TU DISPOSITIVO EN BLYNK"      // Nombre
del dispositivo en app móvil Blynk

#define BLYNK_AUTH_TOKEN "TU TOKEN DE DE BLYNK" // Token otorgado por app móvil Blynk

#define BLYNK_PRINT Serial


#include <WiFi.h>      // Librería para conexión WIFI

#include <WiFiClient.h> // Libre de uso para WIFI

#include <BlynkSimpleEsp32.h> // Librería de interfaz entre modulo y app Blynk

#include <DHT.h>        // Librería de sensor DHT11

#include <DHT_U.h>       // Librería de control DTH11

#include <NewPing.h>     // Librería para facilitar el uso del sensor de ultrasonido

#include <ESP32Servo.h>   // librería para controlar el servomotor SG90


char auth[] = BLYNK_AUTH_TOKEN; // Variable para Token Blynk
                                // para definir con qué frecuencia se envían datos a la app Blynk

char ssid[] = "NOMBRE DE TU RED WIFI"; // Variable con Nombre de red WIFI

char pass[] = "TU CONTRASEÑA WIFI"; // Variable con Contraseña WIFI


Servo servoMotor; // variable para controlar el servoSg90


#define DHTPIN 13 // PIN digital elegido para obtener datos del Sensor DHT11(temperatura y humedad)

#define DHTTYPE DHT11 // defino cual es el sensor que ese usa, en este caso DHT 11

```

```

DHT dht(DHTPIN, DHTTYPE); // Inicializo el sensor DHT11

// aquí hay que definir Trig y Echo
#define trigPin 12
#define echoPin 18
#define DistanciaTope 250 // Defino la distancia máxima en centímetros

NewPing Sonar(trigPin, echoPin, DistanciaTope); // Creo objeto llamado Sonar para tomar valor del
ultrasonido

BlynkTimer timer; // para definir con qué frecuencia se envían datos a la app Blynk

// Motores A

int IN1 = 5; // Asigno pin 4 al pin 1 del lado A del driver L298N
int IN2 = 4; // Asigno pin 2 al pin 1 del lado A del driver L298N
int ENA = 15; // Asigno pin 15 para la habilitación de motores del lado A

// Motores B

int IN3 = 27; // Asigno pin 27 al pin 1 del lado B del driver L298N
int IN4 = 26; // Asigno pin 26 al pin 1 del lado B del driver L298N
int ENB = 25; // Asigno pin 14 para la habilitación de motores del lado B

bool adelante = 0; // variable lógica para controlar auto hacia adelante
bool atras = 0; // variable lógica para controlar auto hacia atrás
bool izq = 0; // variable lógica para controlar auto hacia izquierda
bool der = 0; // variable lógica para controlar auto hacia derecha

void RegistroTempHum() //----Función para obtener temperatura y humedad-----////
{
    float h = dht.readHumidity(); // Tomo en la variable el dato de la humedad
    float t = dht.readTemperature(); // Tomo en la variable el dato de la humedad

    if (isnan(h) || isnan(t)) // Chequeo que las lecturas sean correctas

```

```

{

Serial.println("Error al obtener lecturas del sensor!"); // Pruebas por pantalla serie

return;

}

Blynk.virtualWrite(V5, h); // Entrego la lectura de humedad al pin virtual en la app Blynk

Blynk.virtualWrite(V6, t); // Entrego la lectura de humedad al pin virtual en la app Blynk

} //----- Fin función RegistroTempHum()-----//
```

```

void setup() //----- SETUP Configuración Inicial -----///

{

Serial.begin(9600); // Inicializo puerto Serie

servoMotor.attach(19); // El servo va a trabajar con el pin 13

servoMotor.write(90); // Inicializo en ángulo 90 del servo motor

Blynk.begin(auth, ssid, pass); // Inicializo app Blynk con datos de WIFI

dht.begin(); // Inicializo el sensor DTH

timer.setInterval(5000L, RegistroTempHum); // envío cada 5 segundos información del sensor DHT11

//----- Todos los pines involucrados con los motores como salida-----//
```

```

pinMode(ENA, OUTPUT);

pinMode(ENB, OUTPUT);

pinMode(IN1, OUTPUT);

pinMode(IN2, OUTPUT);

pinMode(IN3, OUTPUT);

pinMode(IN4, OUTPUT);

pinMode(2, OUTPUT);
```

```
} //----- Fin SETUP -----//
```

```
void Detenido() //----- Función que detiene todos los motores-----//
```

```
{
servoMotor.write(90);

digitalWrite(IN1, LOW);

digitalWrite(IN2, LOW);
```

```

analogWrite(ENA, 0); // Velocidad motor A

// Direccion motor B

digitalWrite(IN3, LOW);

digitalWrite(IN4, LOW);

analogWrite(ENB, 0); // Velocidad motor B

}

void Adelantar() ///////////Función para mover hacia adelante el auto///////////

{

servoMotor.write(90);

digitalWrite(IN1, HIGH);

digitalWrite(IN2, LOW);

analogWrite(ENA, 255); // Velocidad motor A

// Direccion motor B

digitalWrite(IN3, HIGH);

digitalWrite(IN4, LOW);

analogWrite(ENB, 255); // Velocidad motor B

}

void Retroceso() /////// Función para mover hacia atrás el auto /////////////

{

servoMotor.write(90);

// Direccion motor A

digitalWrite(IN1, LOW);

digitalWrite(IN2, HIGH);

analogWrite(ENA, 200); // Velocidad motor A

// Direccion motor B

digitalWrite(IN3, LOW);

digitalWrite(IN4, HIGH);

analogWrite(ENB, 200); // Velocidad motor B

}

void HaciaIzquierda() ///// Función para girar hacia la izquierda el auto////////

{

```

```

servoMotor.write(135);

// Dirección motor A

digitalWrite(IN1, HIGH);

digitalWrite(IN2, LOW);

analogWrite(ENA, 200); // Velocidad motor A

// Dirección motor B

digitalWrite(IN3, HIGH);

digitalWrite(IN4, LOW);

analogWrite(ENB, 0); // Velocidad motor B

}

void HaciaDerecha() ////////// Función para girar hacia la derecha el auto //////////

{

// Dirección motor A

servoMotor.write(45);

digitalWrite(IN1, HIGH);

digitalWrite(IN2, LOW);

analogWrite(ENA, 0); // Velocidad motor A

// Dirección motor B

digitalWrite(IN3, HIGH);

digitalWrite(IN4, LOW);

analogWrite(ENB, 200); // Velocidad motor B

}

BLYNK_WRITE(V0) // función para registro de dato en pin virtual de app Blynk

{

adelante = param.asInt(); // Si es valor es 1 el auto va hacia adelante

}

BLYNK_WRITE(V1) // función para registro de dato en pin virtual de app Blynk

{

atras = param.asInt(); // Si es valor es 1 el auto va hacia atrás

}

BLYNK_WRITE(V2) // función para registro de dato en pin virtual de app Blynk

```

```

{
  izq = param.asInt(); // si es valor es 1 el auto gira a la izquierda
}

BLYNK_WRITE(V3) // función para registro de dato en pin virtual de app Blynk

{
  der = param.asInt(); // si es valor es 1 el auto gira a la derecha
}

void MoverAuto() ///// Función para combinar joystick digital con movimientos/////
{
  delay(200);

  int distancia = Sonar.ping_cm(); // variable que obtiene en valor del Sonar(ultrasonido)

  Serial.println("Distancia = " + String(distancia) + " CM"); // Pruebas por pantalla serie

  if (distancia > 30)

  {
    if (adelante == 1)

    {
      Adelantar();
    }

    else if (atras == 1)

    {
      Retroceso();
    }

    else if (izq == 1)

    {
      HaciaIzquierda();
    }

    else if (der == 1)

    {
      HaciaDerecha();
    }

    else if (adelante == 0 && atras == 0 && izq == 0 && der == 0)

```

```

{
  Detenido();
}

}

else

{
  Detenido();
  delay(200);
  Retroceso();
  delay(400);
  Detenido();
}

}

if ((distancia < 30))

{
  digitalWrite(2, HIGH);
}

else

{
  digitalWrite(2, LOW);
}

}

} /////////// fin función mover auto///////////

```

```

void loop() //-----Ejecución del Loop----- //////
{
  Blynk.run(); // Ejecuto plataforma Blynk para conexión y transmisión de datos
  timer.run(); // Ejecuto temporizado para el envío de datos
  MoverAuto(); // Movilidad del auto robot
}

```

```
//////////
```

4.8 ESQUEMA ELECTRÓNICO V1.0.

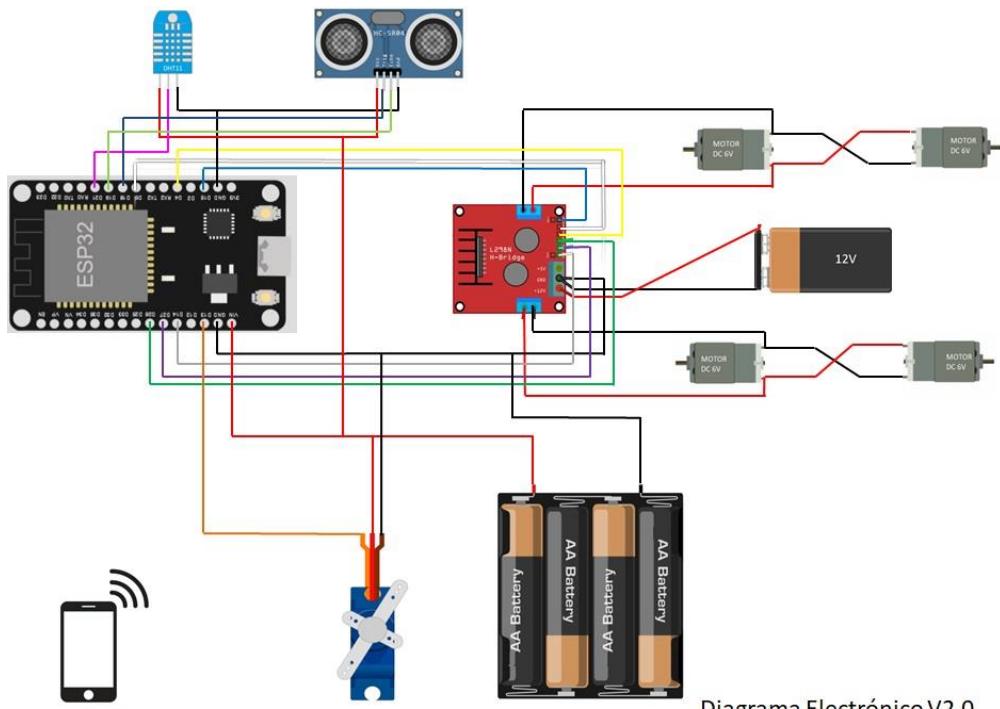


Diagrama Electrónico V2.0

Ilustración 19 - Diagrama Electrónico V2.0

5 MEJORAS O ACTUALIZACIONES Y CAMPO DE APLICACIÓN

- **Seguidor de línea y cámara**

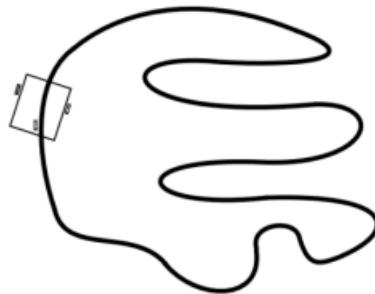
Utilizando el desarrollo del termómetro móvil, nuestro robot TANGO06, al mismo se le puede implementar un sistema seguidor de línea para automatizar su recorrido. La manera más sencilla es utilizar por lo menos dos sensores, proponemos como ejemplo el Módulo sensor óptico TCRT5000.



Ilustración 20 - Módulo Sensor óptico TCRT5000

Estos sensores, correctamente implementados, le dan al robot la capacidad de seguir un camino trazado por una línea, la cual es de un color que contrasta con el color del resto del suelo, es decir, si la línea es negra, el suelo será blanco, y viceversa. Estos sensores se ajustan a la anchura de la pista, y cuando uno de los dos se active, ayudados por el registro de una cámara, significará que

el robot está saliendo de la línea o que se aproxima un giro, y por lo tanto se tiene que ajustar el movimiento.



Esquema de un seguidor de línea recorriendo un circuito.

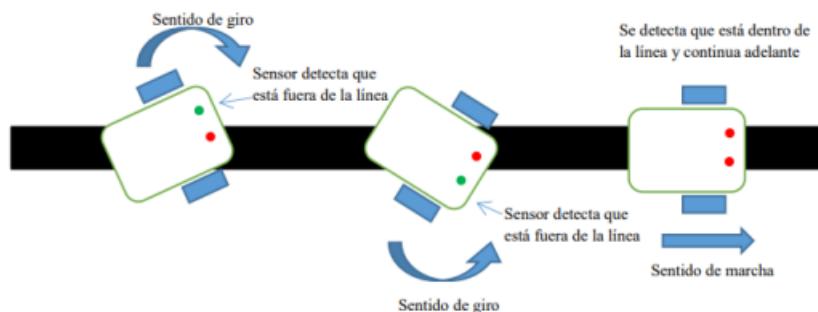


Ilustración 21 - Esquema de detección de línea

Para la implementación de la cámara se podría utilizar la versión ESP32 CAM, la cual contiene las mismas prestaciones que la placa de desarrollo utilizada en nuestro proyecto, sumando propiamente la cámara y además conexión para tarjeta MicroSD.



Ilustración 22 - ESP32 CAM

También es posible el uso de la placa Raspberry pi y algún módulo de cámara oficial de la misma marca.



Ilustración 23 - Placa Raspberry pi

En cuanto al campo de aplicación, nuestro Termómetro móvil (también mide humedad) junto a estas aplicaciones de ejemplo podrá ser implementado en lugares con un suelo liso de pequeñas a medianas dimensiones, donde sea fácil implementar las líneas de seguimiento como puede ser viveros, invernaderos convencionales o hidropónicos, depósitos de alimentos o productos sensibles a la temperatura u humedad. Además, siendo un proyecto que no deja de ser escalable se pueden agregar más sensores para mejorar el seguimiento lineal sorteando saltos de líneas e irregularidades, y adaptarlo a terrenos más grandes, como campos de siembra, plantaciones, granjas avícolas, etc.

- **Almacenamiento de datos y registros, local y en la nube**

En nuestro proyecto, a través de un sensor medimos temperatura y humedad y estos datos se visualizan en una app móvil instantáneamente, pero por una cuestión de funcionalidad y practicidad lo ideal es que estos registros puedan ser almacenados de alguna manera. La placa de desarrollo ESP32 cuenta con conectividad WIFI y Bluetooth, y además como vimos en el ítem anterior la ESP32 CAM tiene ranura para MicroSD, por lo que eligiendo un protocolo adecuado (Wifi, MQTT; Web Server, LoRa, etc.), una base de datos local o en nube, con el desarrollo de código abierto correspondiente y sus bibliotecas, cualquier aparato de uso cotidiano puede ser desarrollado y convertido en un dispositivo IoT.

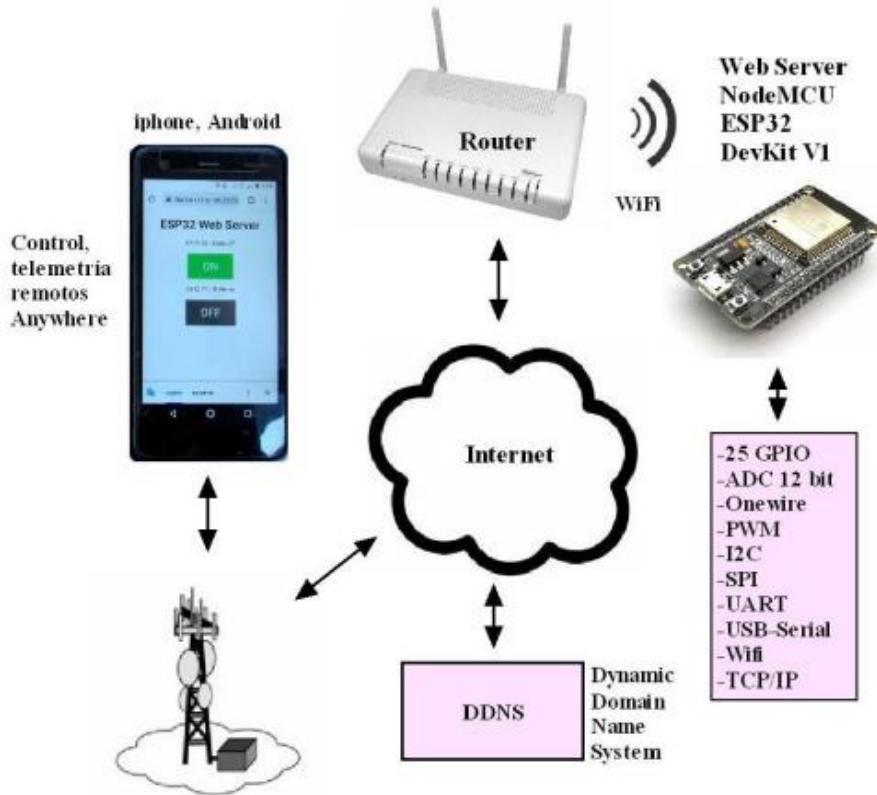


Ilustración 24 - Ejemplo de Conectividad – ESP32

- Mejoras en las ruedas, motores, suspensión

De acuerdo al campo de aplicación se pueden encontrar infinidad de opciones para mejorar la tracción del auto robot. Ruedas de distinto tamaño y propósito de uso, junto con un agregado de suspensión, combinando con motores de distintas potencias.

Nuestro proyecto



Ilustración 25 - Nuestro Proyecto

Ejemplos de ruedas



Ilustración 26 - Ejemplo de ruedas

Ejemplos de Suspensión:



Ilustración 27 - Ejemplo de suspensión

Ejemplos de motores:



Ilustración 28 - Ejemplo de motores

- Autonomía energética, pantalla solar y baterías recargables.

Con el fin de optimizar la funcionalidad del auto robot, y dotarlo de autonomía en su funcionamiento, además de contribuir al cuidado del medioambiente, se proponen alternativas de alimentación energética con pilas y baterías recargables, con opciones de recarga a través de un panel solar.

Batería recargable AA de 1.2V



Ilustración 29 - Batería recargable AA de 1.2V

Esta pila tiene un voltaje inferior (1.2V), es una opción económica ya que necesitaríamos 5 de estas y un cargador, después de esto no tendremos que estar renovando y comprándolas. Características: Carga de 500 a 1000 mAh en pilas de NiCd y una carga de entre 600 y 2500 en pilas de NiMh. La intensidad máxima es de aproximadamente 1A de forma sostenible, lo suficiente para proyectos pequeños.

Powerbanks 5V



Ilustración 30 - Powerbanks 5V

Esta opción puede tener un costo elevado, pero puede también ser una gran opción de alimentación. Características: Traen un regulador de voltaje, por lo que los 5V que entrega siempre van a ser regulados. Son recargables y pueden entregar altas capacidades de energía, donde encontraremos Powerbanks de hasta 17000 mAh. La intensidad máxima es baja, logrando obtener por lo general 2A como mucho, lo cual hace que sea muy reducida para proyectos más grandes.

Baterías de Níquel – MetalHidruro



Ilustración 31 - Baterías de Níquel – MetalHidruro

Esta es una opción que abarca una gama alta entre las alternativas recargables, también son baterías NiMh a diferencia que estas vienen integradas, pueden ser de menor tamaño y traen un conector. Las más comunes tienen en general 5 celdas con 6V o 8 celdas con 9.6V. Las primeras son una opción excelente para proyectos que incluyen motores DC o servomotores. Características: Tienen una densidad energética media-alta, podemos encontrar de estas con capacidades entre los 300 hasta los 5000 mAh. Dependiendo de su modelo, pueden entregar una capacidad energética de hasta 4C, pudiendo ser hasta 15A en caso de grandes baterías. Son algo costosas, y se necesita un cargador, que también tiene su costo, y los cables deben ser de corriente alta que también tienen su valor costoso

Baterías de polímero de Litio (LiPo)

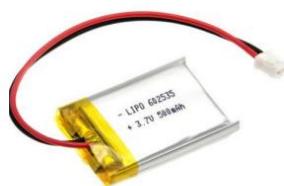


Ilustración 32 - Baterías de polímero de Litio (LiPo)

Esta es la opción más pro para alimentar nuestro proyecto, se dan en varios voltajes, en relación con el número de celdas. Las LiPo de 2 celdas proporcionan entre 7.4 y 8.4V, y las de 3 celdas proporcionan entre 11.1 y 12.6V. Ambas opciones son ideales para la alimentación de placas de desarrollo. Propiedades de las baterías LiPo: Las baterías de 2 celdas (2S) pueden ser usadas para motores DC y servomotores siempre y cuando su tensión sea reducida a un voltaje de entre 6 y 7V. Las baterías de 3 celdas (3S) son apropiadas para usarse en motores pasos a paso o motores busheles. Esta opción tiene la densidad energética más alta entre todas las opciones mencionadas, pudiendo encontrar baterías con capacidades de hasta 5000 mAh. Son la opción más cara en primera instancia, pero a largo plazo resultan siendo la opción más rentable si se le sabe dar un buen uso, el cual requiere de mucho cuidado, incluso hasta la carga, ya que pueden ser peligrosas porque contienen mucha energía almacenada.

- **Paneles Solares**



Ilustración 33 - Paneles Solares

Kit solar PARA CONSTRUIR una unidad de alimentación para palcas de desarrollo de 5V. Se compone de un panel solar, una batería lipo de 600mAh, un cargador de batería y un elevador de tensión con salida de 5V. Incluye un conector microUSB que puedes usar para cargar la batería en vez de usar el panel solar. Características: Panel solar 5V 1,25W (110 x 69mm) Batería Lipo de 3,7V 600mAh Alimentación del cargador: 5V Carga máxima: 480mA. A todas las alternativas de alimentación recargable se le puede implementar un panel solar para la recarga de la batería, y se pueden realizar adaptaciones como los siguientes ejemplos:



Ilustración 34 - kit solar para construir

- **GPS**

Al ser un auto robot con cierta autonomía puede ser necesario rastrear la ubicación del mismo monitoreándolo. GPS, Servicio de Posicionamiento Global es la opción para el seguimiento en terreno. Se puede usar la placa de desarrollo ESP32 junto a una shield GPS, como por ejemplo el Módulo GPS GY-NEO6MV2, y tener trazada la ubicación en Google Maps, por ejemplo.



Ilustración 35 - Módulo GPS GY-NEO6MV2

Funcionalidad íntegra en página web

Podemos integrar todo lo descripto anteriormente en una página web, es decir, ver los registros y parámetros actuales de temperatura, chequear con el GPS la posición actual de nuestro auto robot y los recorridos que ha realizado mediante una interfaz gráfica como puede ser Google Maps. Visualizar el nivel de batería sería también posible, como acceder a la cámara si elegimos implementar el proyecto con el módulo ESP32 CAM, y a cualquier parámetro y/o funcionalidad que quisiéramos agregarle a nuestro proyecto convirtiendo la placa ESP32 en un verdadero servidor Web.



Ilustración 36 - ESP32 WEB SERVER

- **IA, Estación meteorología, Servicio meteorológico Nacional**

A las lecturas de temperatura y humedad de nuestro proyecto se le podría agregar más parámetros como presión atmosférica con un barómetro, dirección y velocidad del viento con un anemómetro, precipitación con un pluviómetro, y hasta radiación solar. Transformándolo así una verdadera estación meteorológica móvil, con la posibilidad de brindar datos a organismos oficiales como al Servicio Meteorológico Nacional, INTI, INTA, etc.; y de uso general doméstico, industrial y agropecuario. Logrando así prever tormentas, granizo, heladas y por qué no tornados, ciclones o tifones. La imagen abajo muestra una estación meteorológica de diseño de unos 100 Euros, podemos imaginar la segunda imagen sobre nuestro TANGO06 la cual no supera los 40 cm de alto y pesa unos 400 gramos.



Ilustración 37 - Sensor Profesional

Por último, para destacar que la escalabilidad no tiene tope en este tipo de implementación con todos los datos registros, las mejoras, actualizaciones, campos de implementación y disponibilidad de conectividad podemos crear y entrenar una aplicación de inteligencia artificial que optimice la funcionalidad y previsión climática a tras de estos datos meteorológicos obtenidos.

6 RESULTADOS

A continuación, presentamos los resultados obtenidos a partir de la aplicación de los instrumentos:

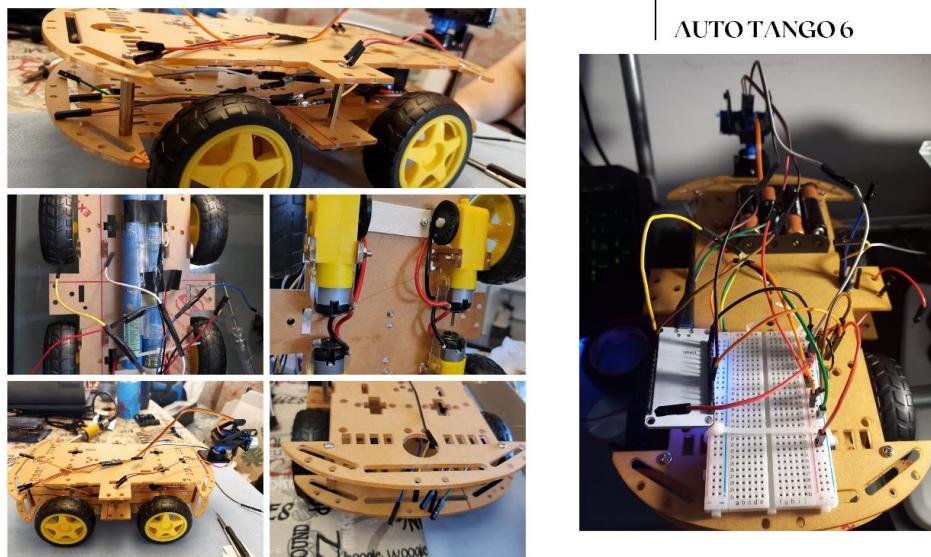


Ilustración 38 – Auto Prototipo Tango 6

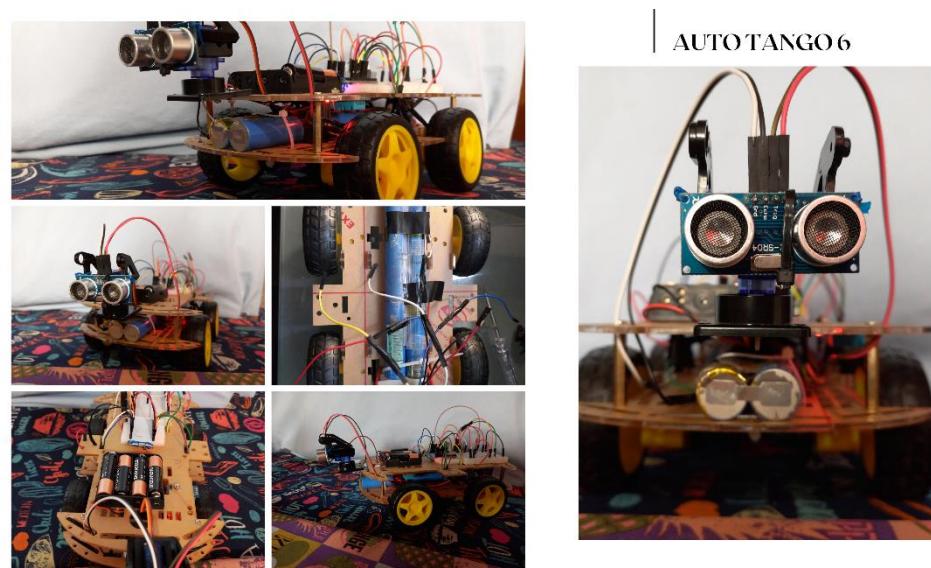


Ilustración 39- Auto Tango 6 Final

FICHA DE SEGUIMIENTO SEMANA 1 (del 5 al 9 de septiembre) DE PROYECTOS:

DIA DE LA SEMANA	TAREA HABLADA/REALIZADA	RESOLUCIÓN
LUNES	Se habló y se debatió que Proyecto íbamos a llevar a cabo.	No se llegó a un acuerdo debido a que había varias ideas en mesa sin 1 más sólida.
MARTES	Se habló y se debatió que Proyecto íbamos a llevar a cabo.	Se logró llegar a una preselección de 2 ideas.
MIERCOLES	Se habló y se debatió que Proyecto íbamos a llevar a cabo.	Se llegó a un acuerdo de hacer el autito pero con una innovación a futuro para no dejarlo tan simple. -
JUEVES	Se habló de los precios extraídos de los proveedores y se empezó a diagramar quien se iba a encargar de cada paso a seguir.	No hubo gran avance en cuanto al precio total ni las tareas. -
VIERNES	Se habló de los precios extraídos de los proveedores y se empezó a diagramar quien se iba a encargar de cada paso a seguir.	Algunas tareas ya quedaron establecidas y el precio final del Proyecto aproximadamente ya está definido. - Se grabó el video semanal

Ilustración 40 - Ficha de seguimiento 1

FICHA DE SEGUIMIENTO SEMANA 2 (del 12 al 16 de septiembre) DE PROYECTOS:

DIA DE LA SEMANA	TAREA HABLADA/REALIZADA	RESOLUCIÓN
LUNES	Se sigue con el proceso de armado del autito	Armamos el chasis
MARTES	Se sigue con el proceso de armado del autito	Incorporamos batería y sensor HC-SR04
MIERCOLES	Se sigue con el proceso de armado del autito	Se comienza a implementar el código. -
JUEVES	Se sigue con el proceso de armado del autito	Se sigue implementando el código. -
VIERNES	Se sigue con el proceso de armado del autito	Se termina de implementar el código y se realizar algunas pruebas. -
SABADO Y DOMINGO	Tarea Extra por demoras.	Se terminó de armar y codificar el carrito, restan pruebas.-

Ilustración 41 - Ficha de seguimiento 2

FICHA DE SEGUIMIENTO SEMANA 3 (del 19 al 23 de septiembre) DE PROYECTOS:

DIA DE LA SEMANA	TAREA HABLADE/REALIZADA	RESOLUCIÓN
LUNES	Se implementaron pruebas al prototipo final.	Hubo algunos problemas con las baterías por lo que tuvimos que implementar otras nuevas.
MARTES	Se implementaron pruebas al prototipo final.	Hubo algunos problemas con los códigos de programación.
MIERCOLES	Se implementaron pruebas al prototipo final.	Se quemó el servo y tuvimos que cambiarlo.
JUEVES	Charla final ultimando detalles.	Se habló de lo que íbamos a presentar en video y se terminaron de destinar las últimas tareas de edición de contenido en github.
VIERNES	Se presentó el prototipo final funcionando.	Hicimos una prueba final del autito funcionando a la perfección. Se grabó el video final.

Ilustración 42 - Ficha de seguimiento 3

7 CONCLUSIONES

Para llevar a cabo la realización de este proyecto han sido necesarios los conocimientos adquiridos a lo largo de toda la formación del módulo de Electrónica Microcontrolada en el ISPC. Sin embargo, las nociones sobre electrónica digital y analógica son claramente las más empleadas a nivel práctico, todo esto unido a una base previa de conocimientos sobre programación. Mediante la suma de estos conocimientos fue posible la realización del proyecto. Una buena forma de valorar los resultados de un proyecto es remitirse a los objetivos marcados en un principio e intentar realizar una valoración objetiva.

En cuanto al primero de ellos se puede afirmar que se ha conseguido un buen entendimiento del funcionamiento de todos los componentes hardware utilizados, además de un buen control y manejo del microcontrolador empleado, descubriendo y aplicando las distintas herramientas de las que dispone. Otro de los objetivos propuestos era obtener un control total del vehículo pudiendo manejarlo de forma fluida, objetivo que se puede dar por alcanzado, puesto que mediante el control de la velocidad de los distintos modos de operación se ajusta la conducción a las necesidades del entorno (dependiendo del terreno se precisará de más potencia de los motores para manejarse correctamente).

En referencia a los objetivos secundarios, se ha conseguido implementar un algoritmo capaz de esquivar todo tipo de obstáculos en conducción automática, controladas desde el Smartphone. En cuanto a posibles aspectos a mejorar, la presentación, por ejemplo, se podría mejorar la imagen añadiendo una carrocería o usando otro tipo de materiales comerciales para conseguir una distribución de los componentes más compacta. Con respecto al software, se podría mejorar el algoritmo de la función esquiva obstáculos para ser más eficiente, tomando más medidas de distancia, y conseguir de esta forma una conducción más fluida

8 Bibliografía

Agro, T. (7 de Julio de 2021). <https://tecnoagro.com.mx/no.-149/tecnologia-iot-para-invernaderos-mejorando-la-agricultura>.

InterEmpresas. (s.f.). <https://www.interempresas.net/Horticola/Articulos/77307-Control-climatico-en-invernaderos.html>.

Report-InternetOfThings. (s.f.). <https://www.internetsociety.org/wp-content/uploads/2017/09/report-InternetOfThings-20160817-es-1.pdf>.

Society, I. (s.f.). <https://www.internetsociety.org/wp-content/uploads/2017/09/report-InternetOfThings-20160817-es-1.pdf>.

Wikipedia “Radio-Frequency Identification.”, T. F. (6 de Septiembre de 2015). https://en.wikipedia.org/wiki/Radiofrequency_identification.

Wikipedia. (s.f.). https://es.wikipedia.org/wiki/Diagrama_de_Gantt.

OTRAS FUENTES CONSULTADAS

NaylampMechatronics

https://naylampmechatronics.com/blog/53_robot-movil-controlado-por-bluetooth.html

Dynamoelectronics

<https://dynamoelectronics.com/partes-para-armar-un-robot-seguidor-de-linea/>

Youtube

https://www.youtube.com/watch?v=DeE5_g14CdE

https://www.youtube.com/watch?v=_1gSxXOV4cs

Blog Uelectronics

<https://blog.uelectronics.com/robotica/crea-tu-propio-robot-educacional-a-control-remotobluetooth/>

Mercado Libre

[https://listado.mercadolibre.com.ar/cordoba-robotica#D\[A:cordoba%20robotica\]](https://listado.mercadolibre.com.ar/cordoba-robotica#D[A:cordoba%20robotica])

TalosElectronics

<https://www.taloselectronics.com/blogs/tutoriales/programar-esp32-con-ide-arduino>