Universidad Autónoma de Nuevo León

Facultad de Ingeniería Mecánica y Eléctrica

Lab. Controladores y Microcontroladores Prog.

Documentacion

**Maestro:** Héctor Hugo Flores Moreno

**Hora:** N5 – N6 Jueves

**Equipo:** 2

|  |  |
| --- | --- |
| Nombre | Matricula |
| Villanueva Moreno Rene Alejandro | 1795612 |
| Beltran Melendez Aldair Alejandro | 1802548 |
| Cavazos Almaguer Rodolfo | 1830322 |

Indice

[¿Qué hace el proyecto? 2](#_Toc212681474)

[¿Cómo funciona internamente? 3](#_Toc212681475)

[1. Arquitectura general del sistema 3](#_Toc212681476)

[2. Interconexión entre bloques 4](#_Toc212681477)

[¿Cómo puedo usarlo o contribuyo al proyecto? 4](#_Toc212681478)

[En software: 4](#_Toc212681479)

[En hardware: 5](#_Toc212681480)

[Para contribuir/ Descargar los archivos 5](#_Toc212681481)

[Montaje y Carga Inicial 6](#_Toc212681482)

[a) Montaje del Circuito 6](#_Toc212681483)

[b) Carga del Código al Arduino 6](#_Toc212681484)

[Ejecución del Entorno Local 6](#_Toc212681485)

[¿Cómo correr tests? 7](#_Toc212681486)

[Preguntas Frecuentes (FAQ) 7](#_Toc212681487)

## ¿Qué hace el proyecto?

Este Proyecto funciona como sensor ultrasónico para detectar objetos cercanos mediante un módulo de sensor ultrasónico simulando el sistema de control aéreo útil para implementar en drones para detectar objetos a su alrededor. El alcance de este trabajo no se limita simplemente a dispositivos aéreos, también puede ser utilizado en terrestres como un carrito inteligente.

## ¿Cómo funciona internamente?

### 1. Arquitectura general del sistema

El proyecto se compone de tres bloques principales:

#### Bloque Sensor (HC-SR04)

Este módulo es el encargado de medir distancias usando ultrasonido.

* **Trig (pin 10):** Arduino envía un pulso de 10 microsegundos.
* **Echo (pin 11):** El sensor responde con una señal cuya duración depende del tiempo que tarda el sonido en ir y volver.

Arduino mide este tiempo y calcula la **distancia** usando la fórmula:

(0.034 cm/μs = velocidad del sonido).

**En otras palabras:**  
El sensor emite un “ping” y escucha su “eco”. Cuanto más rápido regresa, más cerca está el objeto.

#### B. Bloque de Control (Arduino UNO)

El Arduino actúa como el cerebro del sistema.  
Sus funciones principales son:

**Generar el barrido:** Usa el servomotor para mover el sensor paso a paso entre 15° y 165°.

**Controlar el sensor:** Activa el pin Trig y mide la señal Echo.

**Procesar los datos:** Calcula la distancia con base en el tiempo medido.

**Enviar resultados:** Manda el ángulo actual y la distancia por el puerto serial.

#### C. Bloque de Actuador (Servomotor)

El servomotor está conectado al pin 12 y su tarea es rotar el sensor ultrasónico. Usa una señal PWM (modulación por ancho de pulso) para posicionarse en un ángulo específico. Arduino genera pulsos con la librería <Servo.h>. Cada vez que cambia el ángulo, el sensor apunta en una dirección diferente y mide la distancia ahí.

Esto permite al sistema escanear el entorno en un rango de 150°.

## 2. Interconexión entre bloques

Diagrama

El contenido generado por IA puede ser incorrecto.

* El servo depende de Arduino para moverse.
* El sensor depende de Arduino para emitir y medir.
* Arduino coordina ambos y envía los resultados al monitor serial o a un programa visualizador.

\*SI SE DESEA SABER MAS SOBRE EL FUNCIONAMIENTO PUEDE REVISAR EL CODIGO Y LOS DOCUMENTOS DE PRACTICAS QUE ESTAN EN EL REPOSITORIO (<https://github.com/RexAlejandro/Controlador-de-Vuelo>)

## ¿Cómo puedo usarlo o contribuyo al proyecto?

Para poder realizar este proyecto requiere

### En software:

* Arduino IDE (o el que usted guste) <https://www.arduino.cc/en/software>
* Processing - <https://processing.org/download>
* Importar la libreria Servo
* Git (Si desea clonar el Proyecto o solicitar algún cambio) - <https://git-scm.com/>

### En hardware:

* Arduino (Puede usar un microcontrolador similar)
* Módulo de Sensor de ultrasonido
* Servomotor
* Batería de 9V (O puede conectarlo directamente a la computadora si desea)
* Cables

### Para contribuir/ Descargar los archivos

El código se encuentra en la nube en un repositorio de GitHub al cual puede acceder desde el enlace: <https://github.com/RexAlejandro/Controlador-de-Vuelo>

Si desea contribuir siempre puede realizar un fork de nuestro proyecto original y realizar algún cambio y solicitar una Pull Request para que lo implementemos.

#### Ejecutar el código en local: Clonar el Repositorio

Abre una terminal o línea de comandos y clona el repositorio que contiene el código de visualización:

Desde el Bash

git clone <https://github.com/RexAlejandro/Controlador-de-Vuelo>

Revisa la documentación (README) dentro del repositorio clonado para confirmar si el proyecto usa alguna librería especial o algún otro lenguaje.

#### Subir cambios a tu fork

git add .

git commit -m "feat: [Descripción clara de la contribución]"

git push origin feature/nombre-de-tu-mejora

Para Abrir un Pull Request (PR): Vuelve a la página de GitHub de tu *fork* y abre un Pull Request hacia el repositorio original (RexAlejandro/Controlador-de-Vuelo). Describe detalladamente qué hace tu contribución y por qué es necesaria.

## Montaje y Carga Inicial

### a) Montaje del Circuito

Sigue las correcciones del mapeo de pines para asegurar la operatividad con el código proporcionado:

| **Componente** | **Pin en el Código** | **Pin de Conexión** |
| --- | --- | --- |
| **HC-SR04** (Trig) | trigPin = 10 | Digital **10** |
| **HC-SR04** (Echo) | echoPin = 11 | Digital **11** |
| **Servo Motor** (Señal) | myServo.attach(12) | Digital **12** |

### b) Carga del Código al Arduino

1. Abre el *sketch* del radar en el IDE de Arduino.
2. Conecta tu placa **Arduino UNO**.
3. Verifica la selección de la placa y el puerto COM/USB correcto (*Herramientas*).
4. Haz clic en **Subir** para cargar el firmware. El servo comenzará a moverse.
5. Habra Processing y ingrese el código del Arduino IDE a un nuevo sketch y ejecútelo para poder ver el radar.

## Ejecución del Entorno Local

Realice los mismos pasos del punto anterior, pero conectando la placa a la computadora y cargando el código, asegúrese de seleccionar el puerto adecuado, el radar aparecerá desde processing pero siempre puede agregar un módulo de radar si quiere que todo sea físico.

## ¿Cómo correr tests?

Siempre puede ejecutar la aplicación desde el IDE, este ya esta configurado para que le muestre valores útiles desde la consola tales como la distancia, también en el repositorio se agregó un archivo inicial en proteus que puede utilizar como base si desea realizar test.

también puede usar el simulador de Tinkercad para verificar que tanto el montaje como el código funcionan correctamente antes de armarlo físicamente.

<https://www.tinkercad.com/things/2F2I2L3YdKA/editel?returnTo=%2Fdashboard%2Fdesigns%2Fcircuits&sharecode=eQQQH-Y1JWLQfHgqyFGXcCsHEWa90hwFSyQ8MCwwfc0>

Interfaz de usuario gráfica

El contenido generado por IA puede ser incorrecto.

## Preguntas Frecuentes (FAQ)

1. **¿Qué componentes necesito para construir el radar simulado?**

Necesitas un Arduino UNO, un sensor ultrasónico HC-SR04, un servo motor, una protoboard, cables de conexión y un software externo para visualizar los datos.

1. **¿Cuál es el rango máximo del sensor ultrasónico HC-SR04?**

El sensor puede medir distancias desde aproximadamente 2 cm hasta 400 cm (4 metros), dependiendo de las condiciones del entorno.

1. **¿Cómo conecto el servo motor al Arduino?**

Conecta el cable de señal del servo al pin digital 11 del Arduino, y los cables de alimentación (VCC y GND) a la protoboard, que debe estar conectada a 5V y GND del Arduino.

1. **¿Qué software externo se utiliza para visualizar los datos del radar?**

Puedes utilizar software como Processing, Python con bibliotecas gráficas (por ejemplo, matplotlib o Pygame), o incluso plataformas como Tinkercad para simulaciones básicas.

1. **¿Puedo integrar este radar en un simulador de vuelos?**

Sí, este radar simulado puede formar parte de un simulador de vuelos educativo, representando el sistema de detección de obstáculos y monitoreo del entorno de una aeronave.

1. **¿Cuánto tiempo tarda en realizar un barrido completo de 0° a 180°?**

El tiempo de barrido depende del retardo programado entre cada movimiento del servo. En este proyecto, se utiliza un retardo de 30 milisegundos por grado, lo que resulta en aproximadamente 5.4 segundos por barrido completo.

1. **¿Se pueden utilizar otros sensores en lugar del HC-SR04?**

Sí, es posible utilizar otros sensores de distancia como el VL53L0X (sensor láser) o el GP2Y0A21YK0F (sensor infrarrojo), aunque requerirán ajustes en el código y en la forma de interpretar los datos.

1. **¿Cuál es el consumo de energía del sistema?**

El consumo depende de los componentes utilizados. El Arduino UNO consume alrededor de 50 mA, el sensor ultrasónico unos 15 mA, y el servo motor puede consumir entre 100 mA y 250 mA dependiendo de la carga. Se recomienda usar una fuente de alimentación adecuada si se desea operar el sistema de forma continua.

1. **¿Es este proyecto adecuado para principiantes?**

Sí, este proyecto es ideal para principiantes que desean aprender sobre sensores, servos y programación en Arduino. Requiere conocimientos básicos de electrónica y del entorno de desarrollo de Arduino IDE.