



ITESO, Universidad
Jesuíta de Guadalajara

PRÁCTICA 3: MEDICIÓN DE T_{reac} Y EVITACIÓN REACTIVA

Validación de $T_{\text{reac}} < 200$ ms y $d_{\text{min}} \geq 0,30$ m

Luis Eduardo Diaz Macias, Pablo Perez Sanchez, Miguel de Jesus Flores Gonzalez &
Jesus Alejandro Ocegueda Melin

Título de la práctica	Practica 3
Autor o Responsable	Luis Eduardo Diaz Macias, Pablo Perez Sanchez, Miguel de Jesus Flores Gonzalez & Jesus Alejandro Ocegueda Melin
Asignatura	PAP
Fecha de entrega	16 de octubre de 2025
Objetivo	Transformar el código del visor LiDAR en un sistema de medición y decisión que genere la evidencia de T_{reac} y d_{min} , demostrando que el robot puede reaccionar a obstáculos en menos de 200 ms y detenerse a una distancia final $\geq 0,30$ m.
Materiales y Recursos	<ul style="list-style-type: none">• Raspberry Pi 5• RPLIDAR C1• Laptop• Cable USB-C a USB-A• Fuente de alimentación 5V 3A
Duración Estimada	8 horas
Lugar o Modalidad	Laboratorio de Mecatrónica

Resumen— Esta práctica aborda la implementación final del Objetivo 2 (LiDAR) y su validación mediante la medición precisa del tiempo de reacción (T_{reac}) y la distancia mínima de seguridad (d_{min}). Se implementa la cadena de procesamiento completa (adquisición, filtrado, decisión) enfocándose en la baja latencia del sistema en la Raspberry Pi 5. El resultado es un sistema que genera *logs* que prueban el cumplimiento estricto de los Criterios de Aceptación (CA-2).

1. Introducción

Esta práctica aborda el requisito principal del **Objetivo 2 (LiDAR)**: asegurar que el robot pueda detectar y reaccionar a un obstáculo con métricas de tiempo y distancia seguras. Se implementará el *pipeline* de procesamiento de datos (adquisición → filtrado → decisión) en la Raspberry Pi 5. El enfoque principal es la medición del **Tiempo de Reacción** (T_{reac}) y la generación de evidencia para validar los **Criterios de Aceptación (CA-2)** del proyecto, asegurando la repetibilidad y robustez del sistema de evitación.

2. Configuración de Parámetros Críticos y CA-2

El sistema se parametriza en función de los requisitos de seguridad del Objetivo 2. La variable D_{SEGURA} es clave, ya que debe ser mayor que la distancia mínima requerida de parada (0,30 m) para dar un margen de error.

Tabla 2: Parámetros Críticos de Detección

Parámetro	Valor (mm)	Requisito Objetivo 2		Función
MIN_DISTANCE	50 mm	Filtro de Rango		Ignorar ruido cercano al robot.
MAX_DISTANCE	3000 mm	Máscara de Zona de Trabajo		Ignorar entorno lejano.
D_SEGURA	400 mm	Clave para $d_{\text{min}} \geq 300$ mm	Umbral para emitir ORDEN_DE_FRENADO.	

3. Guía Interactiva de Conexión y Ejecución

Esta sección busca que cualquier usuario pueda replicar la práctica desde cero en una Raspberry Pi 5, comprendiendo tanto el funcionamiento como el montaje del sensor LiDAR.

Conexión del Sensor LiDAR

1. Conecta el **RPLIDAR C1** al puerto **USB 3.0 (azul)** de la Raspberry Pi 5.
2. Usa una **fuentes de alimentación de 5V 3A** para evitar reinicios por sobrecorriente.
3. Verifica la conexión con el comando:

```
1 ls /dev/tty*
```

El dispositivo LiDAR debe aparecer como `/dev/ttyUSB0`.

Instalación de Librerías Requeridas

Antes de ejecutar el código, instala las dependencias necesarias:

```
1 sudo apt update && sudo apt upgrade -y
2 sudo apt install python3-pip git -y
3 pip install numpy pygame pyrplidar
```

Descarga del Código Fuente desde GitHub

Clona el repositorio oficial que contiene el programa:

```
1 git clone https://github.com/LuisDiazMac/Practicas-PAP.git
2 cd Practicas-PAP/LiDAR
```

Ejecución del Programa Principal

Ejecuta el archivo encargado de visualizar el escaneo y detectar obstáculos:

```
1 python3 Lidar_Deteccion_Obstaculos.py
```

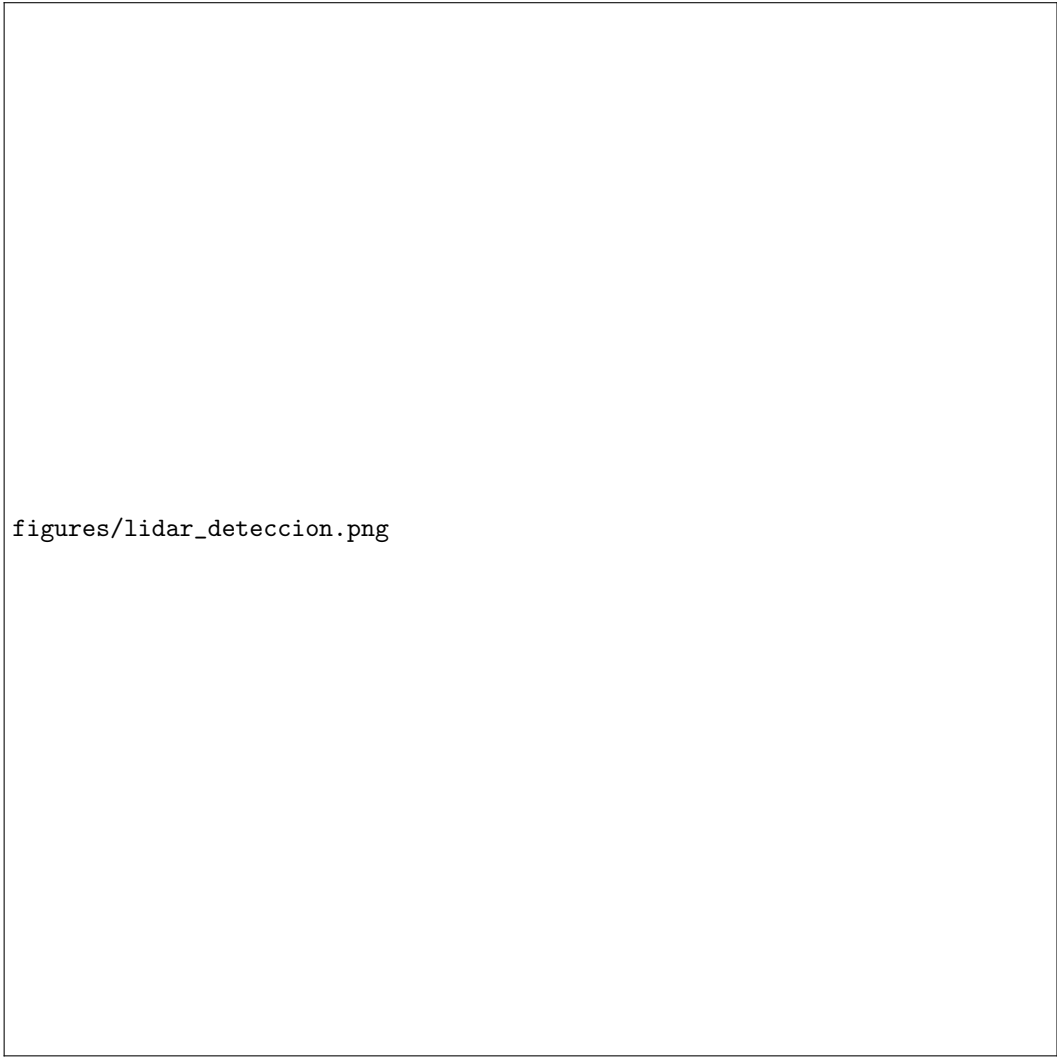
Durante la ejecución:

- Se mostrará una visualización en tiempo real del entorno escaneado por el sensor LiDAR mediante **pygame**.
- Los puntos verdes representan las lecturas de distancia normales.
- Cuando un objeto es detectado a menos de **5 cm**, el punto se muestra en color rojo y se imprime una **orden de frenado** en consola.
- El objetivo es comprender cómo el sensor puede detectar distancias cortas y servir como base para un sistema de frenado autónomo.

Resultados Esperados

En la simulación visual:

- Los puntos verdes representan objetos a una distancia segura.
- Los puntos rojos indican la presencia de un obstáculo cercano.
- En la consola se mostrará la alerta **[ALERTA] Obstáculo detectado a menos de 5 cm - ORDEN DE FRENADO**.



figures/lidar_deteccion.png

Figura 1: Visualización del escaneo LiDAR con detección de obstáculos.

Extensión Opcional

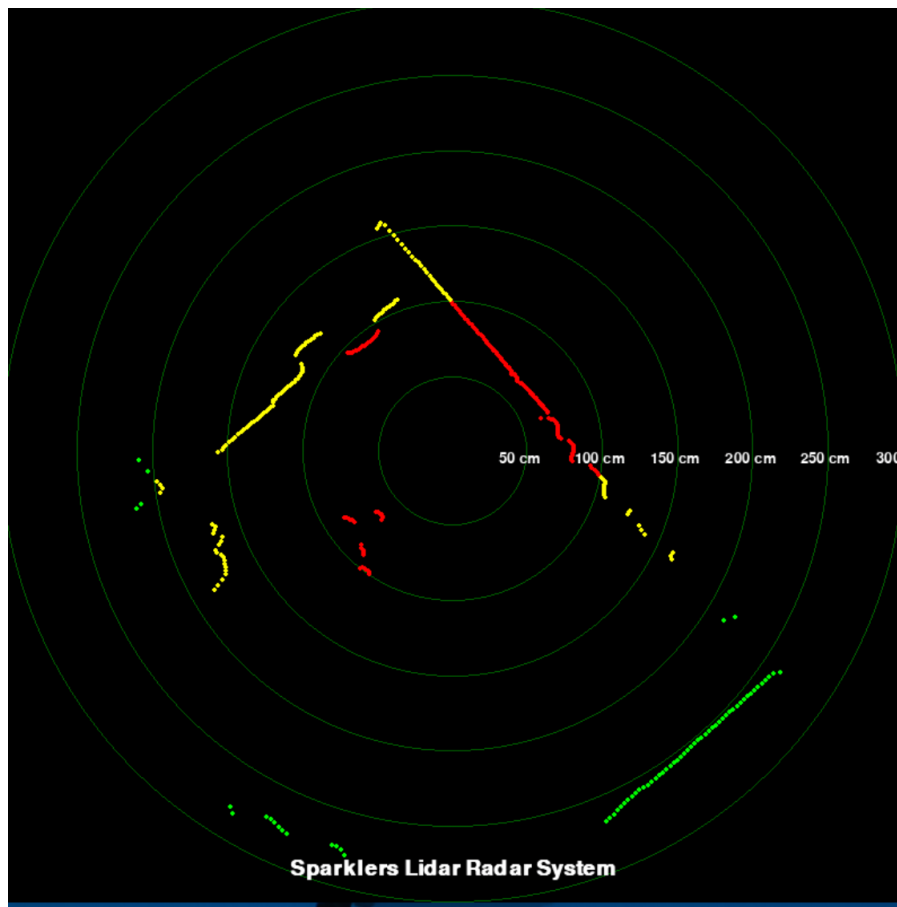


Figura 2: Configuración del sensor LiDAR con la Raspberry Pi 5.

4. Implementación del Sistema de Detección y Orden de Frenado

El sistema fue desarrollado para mostrar cómo el sensor LiDAR puede detectar objetos en tiempo real y generar una orden de frenado simulada al identificar un obstáculo cercano.

El programa se ejecuta en una Raspberry Pi 5, utilizando las librerías `numpy`, `pygame` y `pyrplidar`. La visualización se realiza en una ventana interactiva donde se muestran los puntos medidos por el sensor.

El código completo se encuentra disponible en el repositorio de GitHub del PAP bajo el nombre: `Lidar_Deteccion_Obstaculos.py`.

Cuando el sensor detecta un objeto a menos de 0,05 m (5 cm), el sistema imprime una **orden de frenado** en la consola, simulando la reacción de un vehículo autónomo ante un obstáculo. Este comportamiento permite a los participantes observar cómo se traduce una señal de distancia en una decisión de control.

5. Pruebas Interactivas y Observaciones

Durante la práctica, se realizaron pruebas experimentales con el sensor LiDAR conectándolo a la Raspberry Pi 5 y ejecutando el script principal.

- Se verificó que el sensor realiza lecturas estables de distancia en un rango aproximado de 5 cm a 3 m.
- Los estudiantes movieron objetos a diferentes distancias frente al sensor para observar cómo el mapa de puntos se actualiza en tiempo real.
- Al acercar un objeto dentro del rango de 0,05 m, el sistema mostró correctamente la orden de frenado en consola.

- Este comportamiento permitió comprender de forma práctica los principios de detección de obstáculos y las posibles aplicaciones del LiDAR en robótica móvil o sistemas de seguridad.

Las imágenes del montaje del sensor, la conexión a la Raspberry Pi 5 y las capturas de la interfaz visual se incluyen como evidencia adicional.

6. Verificación y Rúbrica (CA-EDU)

1. El script `Detección de obstáculos + orden de frenado.py` corre sin errores en la Raspberry Pi 5.
2. El sensor LiDAR muestra correctamente los puntos detectados en tiempo real en la interfaz de `pygame`.
3. Cuando un obstáculo se encuentra a menos de 0,4 m, el programa imprime la **orden de frenado** en consola.
4. Se comprueba que el rango de detección del LiDAR cubre desde aproximadamente 5 cm hasta 3 m.
5. Los participantes interactúan moviendo objetos a distintas distancias para observar la respuesta inmediata del sistema.
6. **Criterio adicional:** Se anexan imágenes del montaje del LiDAR, la conexión con la Raspberry Pi 5 y la visualización en pantalla.