Universidad de San Andrés, Depto. de Ingeniería

Trabajo Práctico Final

Parte A

I-402 - Principios de la Robótica Autónoma

Prof. Ignacio Mas, Tadeo Casiraghi y Bautista Chasco

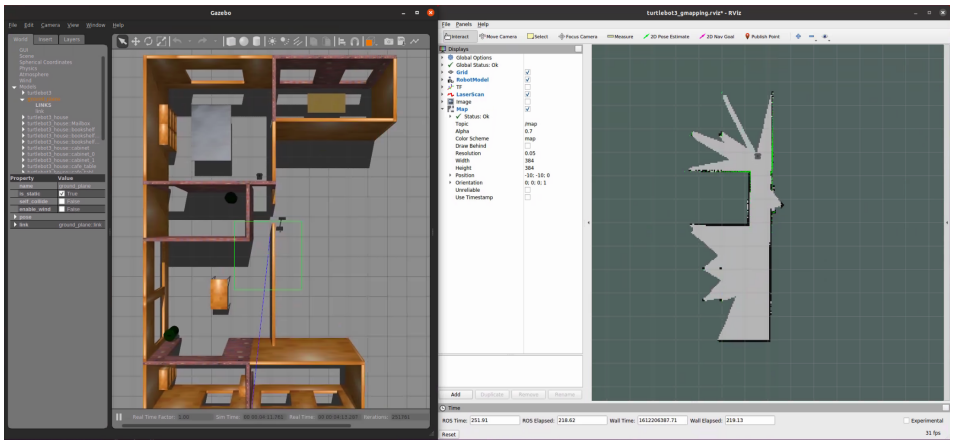
27 de octubre de 2025

Fecha límite de entrega: 05/12/2025, 23.59hs.
Modo de entrega: Enviar por el Aula Virtual del Campus **en un solo archivo comprimido** el paquete de ROS con código comentado y el informe pdf.

En este trabajo práctico final, los alumnos deberán integrar los principales conceptos abordados a lo largo de la materia de Principios de la Robótica Probabilística mediante la implementación de un sistema autónomo de localización y mapeo (SLAM). Utilizando un robot TurtleBot3 simulado en el entorno de Gazebo, el objetivo en esta primera parte (Parte A) será que el robot explore un entorno desconocido tipo laberinto y construya un mapa del mismo mientras estima su propia posición.

Esta etapa representa un caso de aplicación realista donde convergen múltiples herramientas vistas durante la cursada, como la estimación de estado en presencia de ruido, el uso de sensores inexactos (como el LIDAR) y la fusión de información sensorial. La correcta implementación de esta fase es fundamental, ya que el mapa generado y la precisión de localización serán la base sobre la cual se desarrollará la Parte B, centrada en la navegación autónoma.

1. SLAM - Generación del mapa



En esta primera etapa, los alumnos deberán implementar un sistema de SLAM utilizando el robot TurtleBot3 en un entorno simulado en Gazebo. El objetivo principal es que el robot explore de forma manual o autónoma un entorno tipo laberinto y construya un mapa de ocupación del mismo empleando los datos del sensor LIDAR y la odometría.

Esta etapa permite aplicar de forma práctica los algoritmos de estimación de estado y mapeo estudiados en la materia, ya sea mediante técnicas como SLAM con Filtro de Partículas (FastSLAM) o SLAM con Filtro de Kalman Extendido (EKF SLAM), o cualquier otro método pertinente. Al finalizar esta sección, los alumnos deberán exportar el mapa generado y verificar su calidad, ya que servirá como base para la navegación en la siguiente fase del trabajo.

1.1. Preparación del entorno

Antes de comenzar con la implementación de SLAM, es necesario configurar correctamente el entorno de simulación. Para esto se debe tener ya instalado ROS2 Humble, Gazebo y los paquetes de Turtlebot3 ya sea nativamente en una computadora con Ubuntu 22.04 (o cualquier otro OS que lo permita) o mediante Robostack. Recuerden que también es una opción hacer el trabajo en las computadoras del laboratorio de informática del edificio Sullair, pero si se desea hacer eso se deberá tener mucho cuidado de no dejar el código desarrollado en la computadora.

Tendrán que descargar el paquete de simulaciones custom nuevamente ya que se han agregado entornos nuevos. No olviden compilar el código nuevamente.

1.2. Lanzamiento del robot y teleoperación

Una vez que el entorno de simulación está correctamente configurado, el siguiente paso es lanzar el robot TurtleBot3 dentro de un escenario tipo laberinto y permitir que explore el entorno para que más adelante pueda construirse el mapa. Para eso en dos terminales debe correr:

- Terminal 1: `ros2 launch turtlebot3_custom_simulation custom_casa.launch.py`
- Terminal 2: `ros2 run turtlebot3_teleop teleop_keyboard`

1.3. Odometría

La odometría que deben utilizar es la que se publica en el tópico `calc_odom`. Esta es la posición estimada del robot. Tenga en cuenta que puede tener bastante error acumulado. Si buscan actualizar su odometría utilizando el modelo de deltas ($\delta_{\theta_1}, \delta_{\theta_2}, \delta_{trans}$) tendrán que tomar la diferencia con la odometría del tiempo anterior.

1.4. SLAM en ROS 2

En esta etapa, deberán implementar su propio algoritmo de SLAM utilizando los datos publicados por el TurtleBot3 simulado. El objetivo es construir un mapa del entorno mientras se estima simultáneamente la pose del robot. Sin importar el método de mapeo que eligan, deberán poder generar un mapa ya sea de landmarks o de grilla. Al finalizar la ejecución deberán guardar este mapa ya que para la parte B del TP Final la utilizarán para localizarse.

1.5. Visualización con RViz

Durante la ejecución del SLAM, es altamente recomendable usar RViz para visualizar los datos de entrada y la salida de su sistema:
Elementos útiles a visualizar:

- /scan: visualización de los rayos del LIDAR.
- /odom: Posición real del robot.
- /`calc_odom`: Posición del robot estimada por odometría.
- /belief: Posición corregida del robot.
- /map o /landmarks: mapa generado por su algoritmo.

Pueden cargar un archivo .rviz preconfigurado, o configurar la vista manualmente.

2. Evaluación del mapa

Para evaluar la calidad y corrección del mapa generado por su algoritmo de SLAM, se tendrán en cuenta los siguientes aspectos:

2.1. Coherencia con el entorno simulado

El mapa debe reflejar correctamente la estructura del entorno utilizado en Gazebo. Las paredes, obstáculos y pasillos deben estar representados de forma clara y precisa en el mapa. No debe haber grandes espacios libres donde en realidad hay paredes, ni paredes o features que no existen en el entorno. Se debe mapear la totalidad del entorno.

2.2. Resolución y nivel de detalle

La resolución del mapa o la cantidad de features deben ser suficientes para distinguir los elementos importantes del entorno (puertas, esquinas, obstáculos). El mapa no debe contener ruido excesivo, ni saltos abruptos que dificulten su uso para navegación.

2.3. Consistencia temporal

El mapa debe ser estable a lo largo del tiempo: actualizaciones constantes no deben generar cambios erráticos o inconsistencias. El robot debe ser capaz de construir el mapa completo sin perder la ubicación o generar grandes desviaciones.

2.4. Uso para navegación

El mapa generado debe permitir la planificación de rutas entre puntos arbitrarios dentro del laberinto. Se probará la capacidad del robot para localizarse y desplazarse correctamente usando el mapa generado.

3. Entregables

Los paquetes a resolver se entregarán de forma modular. Serán dos, uno para cada parte del final (A y B).