Pasos Lidar

- En terminal poner el comando: sudo apt-get install ros-noetic-rplidar-ros

### **Paso 5: Mapeo y Navegación del Hoverboard**

#### **5.1. Preparación de la navegación en ROS**

Antes de integrar SLAM o navegación, asegúrate de que tu robot (hoverboard) esté configurado correctamente para manejar los comandos de movimiento y la integración con sensores.

1. **Estructura de los frame\_ids (Transformaciones)**
   * Los mensajes del LiDAR o Radar necesitan estar relacionados con el sistema de coordenadas del hoverboard (generalmente con el base\_link).
   * Para esto, debes publicar transformaciones estáticas (usando static\_transform\_publisher) o dinámicas (si es que el modelo de tu hoverboard está en movimiento constante).

Ejemplo de static\_transform\_publisher para el LiDAR:  
  
rosrun tf2\_ros static\_transform\_publisher 0.2 0 0.1 0 0 0 base\_link laser\_link

1. **Parámetros del robot**
   * Si el hoverboard ya tiene un modelo URDF (o SDF en Gazebo), asegúrate de que tenga correctamente configurados los parámetros del sensor, como la distancia entre el LiDAR y el base\_link.
   * Si no tienes un URDF aún, lo puedes crear utilizando herramientas como xacro para simplificar la descripción del robot.

#### **5.2. SLAM (Simultaneous Localization and Mapping)**

SLAM es el proceso de crear un mapa del entorno mientras el robot se localiza dentro de él, todo en tiempo real. Usar un LiDAR o radar 2D es ideal para esta tarea. Aquí te explico cómo configurarlo:

1. **Instala y configura un paquete SLAM**: Hay varias opciones de SLAM para ROS, y dos de las más comunes son gmapping y hector\_slam. Te daré los detalles de ambos:  
   **GMapping**:
   * GMapping es un algoritmo de SLAM basado en partículas que funciona bien con sensores 2D (como LiDAR).

Instálalo si no lo tienes:  
  
sudo apt-get install ros-noetic-gmapping

Luego, lanza el nodo de SLAM con el siguiente comando:  
  
roslaunch gmapping slam\_gmapping.launch

1. Esto inicia el proceso de SLAM. Si tienes un LiDAR funcionando en el tema /scan, el algoritmo comenzará a construir un mapa 2D a medida que el robot se mueve.  
   **Hector SLAM**:
   * Hector SLAM es una opción más rápida y precisa para mapeo 2D en entornos donde el robot se mueve rápido o en entornos muy dinámicos.

Instálalo con:  
sudo apt-get install ros-noetic-hector-slam

Para usarlo, lanza:  
  
roslaunch hector\_slam\_launch tutorial.launch

1. Ambos métodos te permiten obtener un mapa de tu entorno que se actualiza en tiempo real.

#### **5.3. Navegación (move\_base)**

Después de que hayas creado el mapa del entorno, el siguiente paso es hacer que el hoverboard navegue de manera autónoma. Para esto, usamos el nodo move\_base, que se comunica con el mapa y el planificador de rutas para enviar comandos de movimiento.

1. **Configura los archivos de parámetros de move\_base**:
   * Los archivos de parámetros de move\_base (generalmente en YAML) configuran el comportamiento de navegación, como las fronteras del mapa, el tipo de planificación y el control del robot.

Ejemplo de un archivo de configuración de move\_base (en YAML):  
  
local\_costmap:

global\_frame: map

robot\_base\_frame: base\_link

update\_frequency: 5.0

rolling\_window: true

width: 6.0

height: 6.0

resolution: 0.05

global\_costmap:

global\_frame: map

robot\_base\_frame: base\_link

update\_frequency: 1.0

static\_map: true

resolution: 0.05

width: 10.0

height: 10.0

**Lanza move\_base**: Inicia move\_base para que el hoverboard pueda realizar navegación autónoma:  
roslaunch move\_base move\_base.launch

1. **Planificación de rutas**:
   * **Global Planner**: Usa un planificador global para encontrar una ruta desde un punto de inicio hasta un destino. Un ejemplo de planificador global es el Navfn.
   * **Local Planner**: El planificador local ajusta los movimientos del robot en tiempo real para evitar obstáculos. Un ejemplo es el DWAPlanner (Dynamic Window Approach).
2. En el archivo de parámetros de move\_base, puedes configurar el tipo de planificador que usarás.

#### **5.4. Publicación de comandos de movimiento**

Para que el hoverboard se mueva, debes publicar comandos en el tema /cmd\_vel. Estos comandos pueden ser generados por el planificador o manualmente.

**Publicación manual de comandos de movimiento**: Para mover el hoverboard manualmente, usa:  
  
rostopic pub /cmd\_vel geometry\_msgs/Twist --set "linear.x:=0.1" "angular.z:=0.0"

1. Esto moverá el hoverboard hacia adelante. Puedes ajustar los valores de linear.x para controlar la velocidad lineal y angular.z para controlar la rotación.
2. **Comandos de navegación autónoma**: Cuando move\_base esté corriendo, los comandos de movimiento se generarán automáticamente basados en el mapa y la ruta planificada.