# Comunicación ROS1 Noetic a ROS2 Humble

## Requisitos Previos

* PC con [ROS2 Humble](https://docs.ros.org/en/humble/Installation/Ubuntu-Install-Debians.html) y [docker engine](https://docs.docker.com/engine/install/ubuntu/)
* Robot móvil con ROS1 Noetic ([MyAGVPi2023](https://docs.elephantrobotics.com/docs/myagv-2023-en/2-serialproduct/2.13-myAGV2023/PI/1-HardwareStructure.html))

## Install ROS1 Noetic Docker

Correr los comandos en un terminal

docker pull osrf/ros:noetic-desktop-full

docker run -it --name container\_name --net=host -e DISPLAY=$DISPLAY --volume /home/$USER/Docker\_folder:/home/Docker\_folder -v /tmp/.X11-unix/:/tmp/.X11-unix –devicesudo /dev/ttyUSB0 osrf/ros:noetic-desktop-full

Remplazar el “--name” por el nombre para el contenedor deseado. Y reemplazar la ruta “/home/$USER/Docker\_folder” por la carpeta local que se ligara al Docker.

Luego de realizado este proceso, debemos asegurar que el Docker tenga permisos de administrador para así poder añadir archivos a la carpeta del Docker en uso, siguiendo estas instrucciones:

sudo chown -R $USER:$USER /dirección/a la/carpeta definida anteriormente/

docker start container\_name

docker attach container\_name

docker exec -it container\_name bash

roscore

El comando de “attach” es para ligar un primer terminal del Docker al terminal donde se corre el comando. El “exec” es para ligar un nuevo terminal del Docker en una ventana de terminal aparte. “roscore” inicializa los componentes y servicios principales en los que se basan los nodos de ROS para comunicarse y funcionar correctamente.

## Clone the ros-humble-ros1-bridge-builder repository:

En otro terminal correr:

git clone https://github.com/TommyChangUMD/ros-humble-ros1-bridge-builder.git

cd ros-humble-ros1-bridge-builder

docker build . -t ros-humble-ros1-bridge-builder

docker run --rm ros-humble-ros1-bridge-builder | tar xvzf -

source ~/ros-humble-ros1-bridge/install/local\_setup.bash

Para evitar la necesidad de escribir la línea del source cada vez que se requiera correr el rosbridge, hay que anexarlo al documento de .bashrc mediante el siguiente comando:

echo 'source ~/ros-humble-ros1-bridge/install/local\_setup.bash' >> ~/.bashrc

Los anteriores pasos llevan bastante tiempo, por lo que hay que ser pacientes y esperar a que se realice el proceso de instalación completamente. Es importante recordar estar al pendiente pues pueden ocurrir errores durante este proceso.

El último comando es el que se debe correr para habilitar el rosbridge entre el Docker en ROS1 Noetic y el ROS2 Humble nativo.

ros2 run ros1\_bridge dynamic\_bridge

# Ejemplo en el PC.

* La carpeta de ROS1Master contiene los nodos subscriptores. Se crea con:

docker run -it --name migration --net=host -e DISPLAY=$DISPLAY --volume /home/$USER/...Dirección.../ROS1Master:/home/Docker\_folder -v /tmp/.X11-unix/:/tmp/.X11-unix osrf/ros:noetic-desktop

En otro terminal (fuera del docker):

sudo chown -R $USER:$juan /home/juan/Robotics/Proyecto/Mapeo3D\_LiDAR2D/ROS1Master

* Docker Noetic

docker start migration  
docker attach migration ##(ambos en el mismo terminal)

*docker exec -it migration bash ##(en un terminal diferente)*

roscore

* Dentro del docker, correr:

sudo apt update

apt install nano

apt install gedit

Para identificar la dirección IP:

hostname -I

# En el Robot

Para conectar 2 máquinas corriendo ROS1 con el

export ROS\_MASTER\_URI=[http://192.168.8.109:11311](http://192.168.8.109:11311/)

La IP es la del máster, el comando se debe correr en ambas, así como roscore. Esta línea se debe correr en cada terminal nuevo para que los topics que se corran se vean reflejados en el Master. Anexar la línea al final del bashrc y reemplazar por la dirección IP correcta.

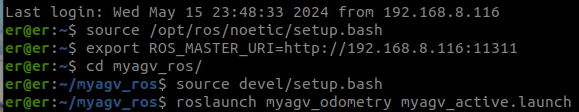
*nano ~/.bashrc*

* Commands to run in ROS1

*source /opt/ros/noetic/setup.bash*

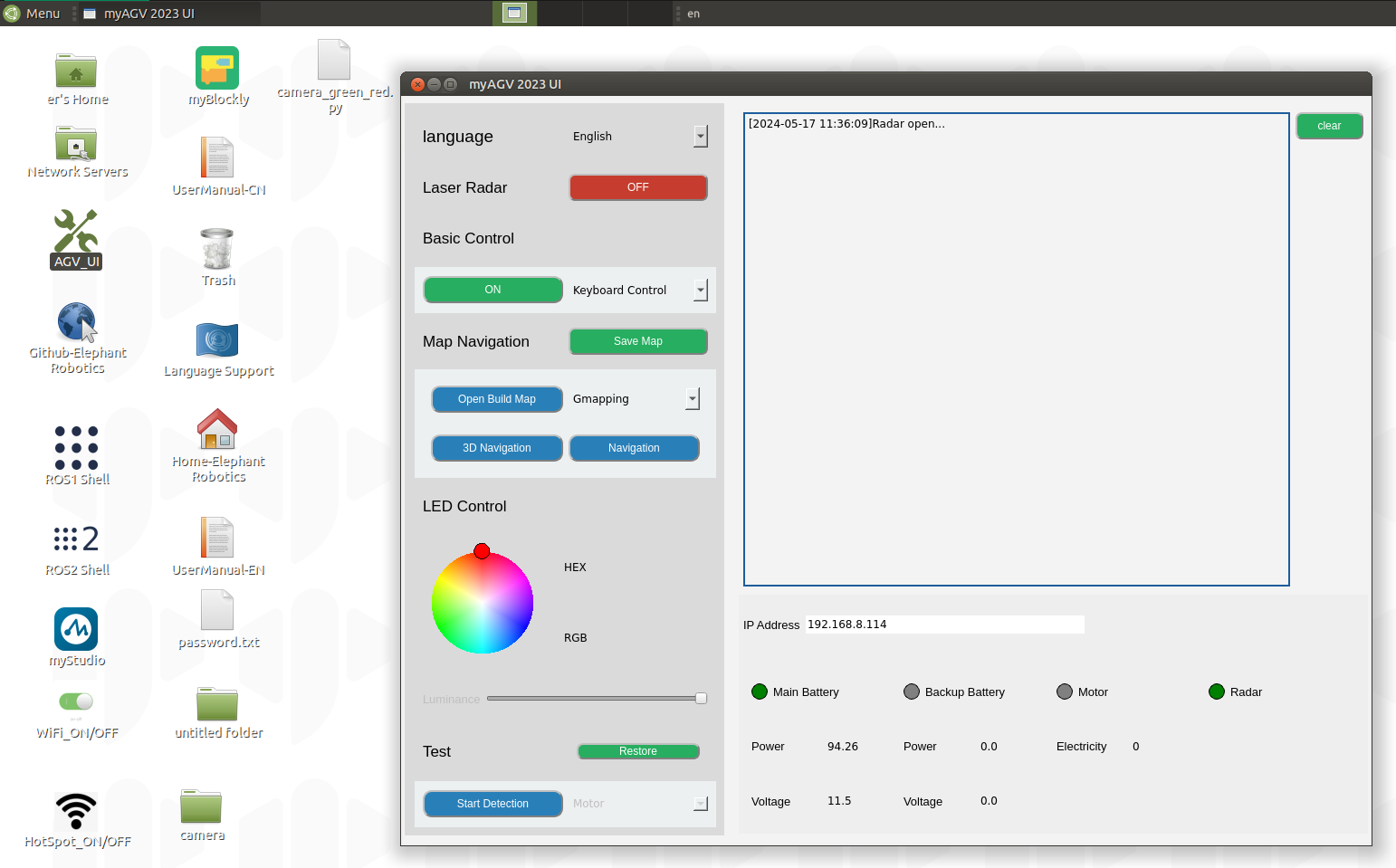
*catkin\_make*

*source devel/setup.bash*

**

Chistoso, por el momento, se debe abrir la GUI del robot, activar el láser (LIDAR) y cerrar el terminal emergente, sin cerrar la GUI. Esto con el fin de que el LIDAR se empiece a mover y posteriormente correr el myagv\_odometry para que el /scan se pueda acceder.

Dentro de la GUI del robot esto es lo único que debe estar corriendo:

**

* Nodo básico que se debe correr para el robot:

*cd myagv\_ros*

*roslaunch myagv\_odometry myagv\_active.launch*

## Resultado de buena comunicación (la anterior imagen).

# Comandos para correr el [Proyecto](https://github.com/JuanR5/Mapeo3D_LiDAR2D)

## Teniendo todo lo anterior claro y correctamente instalado. Para correr el proyecto se debe:

* Tener un terminal con ROS1 Docker, para activar el roscore (ya debe estar configurado el RosMaster)
  + docker start migration
  + docker attach migration
  + roscore
* (Es importante resaltar que se debe hacer una conexión por VNC al robot, para que desde la interfaz se active el lidar RPLidar propio del robot). En un terminal con ssh (ssh er@*dirección IP del robot*):
  + [er@er](mailto:er@er):~/MAP3D$ source devel/setup.bash
  + er@er:~/MAP3D$ roslaunch my\_robot\_pkg robot.launc
* En un otro terminal, correr el RosBridge:
  + ~$ ros2 run ros1\_bridge dynamic\_bridge
* En otro terminal nativo en el workspace de ROS2:
  + ~/Mapeo3D\_LiDAR2D$ ros2 run my\_subscriber\_pkg subscriber\_node
  + ~/Mapeo3D\_LiDAR2D$ ros2 run navigation\_pkg obstacle\_node