**Title: Standard Operating Procedure for Starting ROS 2 Navigation System via Docker on Windows with ESP32 Integration**

**Objective:** To establish a repeatable method for starting ROS 2 inside a Docker container on a Windows machine (via VS Code) to enable communication with an ESP32-based robot over a local network.

## Prerequisites:

* **Windows machine** with:
  + Docker Desktop installed and running
  + Visual Studio Code with Docker and Remote - Containers extensions
  + A local network with known IP addresses (e.g., laptop IP: 192.168.0.101, ESP32 IP: 192.168.0.109)
* **ESP32 robot** configured with:
  + Motor control
  + OLED display
  + Sensor suite (LIDAR, encoder, IMU, ultrasonic)
  + UDP communication receiver listening on port 4211

## Step-by-Step Startup Procedure:

### 1. Start Docker Container with ROS 2 (Persistent Setup)

Open PowerShell or terminal and run:

# Run the ROS container with persistent storage  
docker run -it --name ros2\_nav\_container ^  
 -p 11311:11311 ^  
 -p 9090:9090 ^  
 -v ros2\_nav\_ws:/root/etech\_ws ^  
 sha256:37a640a84e2bb0f255936c5fb5343a213fae16ea86f11db402a2cc750f357c89

**Key Notes:** - --name ros2\_nav\_container gives the container a name for easy reuse - -v ros2\_nav\_ws:/root/etech\_ws mounts a persistent Docker **volume** so files persist across reboots - The --rm flag is **removed** so the container is not deleted on exit

To restart the container later:

docker start -ai ros2\_nav\_container

docker start -ai ros2\_nav\_container

### 2. Set ROS Networking Environment

Inside the container terminal, set your ROS networking variables:

export ROS\_MASTER\_URI=http://192.168.0.101:11311  
export ROS\_IP=192.168.0.101

Replace 192.168.0.101 with your **laptop’s IP address on the local network**.

### 3. Restore udp\_bridge Package if Missing

If udp\_bridge is not found, recreate it:

mkdir -p ~/etech\_ws/src/udp\_bridge/udp\_bridge  
cd ~/etech\_ws/src/udp\_bridge

Add the following files:

**package.xml**

<package format="3">  
 <name>udp\_bridge</name>  
 <version>0.0.1</version>  
 <description>UDP bridge for cmd\_vel</description>  
 <maintainer email="you@example.com">Your Name</maintainer>  
 <license>MIT</license>  
 <buildtool\_depend>ament\_cmake</buildtool\_depend>  
 <exec\_depend>rclpy</exec\_depend>  
 <exec\_depend>geometry\_msgs</exec\_depend>  
</package>

**setup.py**

from setuptools import setup  
  
package\_name = 'udp\_bridge'  
  
setup(  
 name=package\_name,  
 version='0.0.1',  
 packages=[package\_name],  
 install\_requires=['setuptools'],  
 zip\_safe=True,  
 maintainer='you',  
 maintainer\_email='you@example.com',  
 description='UDP bridge for cmd\_vel',  
 license='MIT',  
 entry\_points={  
 'console\_scripts': [  
 'cmdvel\_udp\_bridge = udp\_bridge.cmdvel\_udp\_bridge:main'  
 ],  
 },  
)

**udp\_bridge/\_\_init\_\_.py**

# Empty

**udp\_bridge/cmdvel\_udp\_bridge.py** Paste your working UDP bridge script here.

Then build:

cd ~/etech\_ws  
colcon build  
source install/setup.bash

### 4. Run the Bridge Node

ros2 run udp\_bridge cmdvel\_udp\_bridge

On another terminal start another ros container or node

docker run -it --rm -e DISPLAY=host.docker.internal:0.0 ros2\_humble

### 5. Control the Robot

Use the ROS 2 keyboard teleop tool:

ros2 run teleop\_twist\_keyboard teleop\_twist\_keyboard

This will send /cmd\_vel messages over the network, received by the ESP32.

### 6. Visualize or Interact (Optional)

You can use:

* **RViz2** (if GUI supported)
* **rosbridge + web tools** (via port 9090)

## ESP32 Responsibilities (Summary)

* Listens for UDP messages from ROS 2 node
* Parses Twist messages into motor commands
* Displays movement direction on OLED
* Interfaces with sensors (e.g., MPU6050, LIDAR, encoders, ultrasonic)

## Conclusion

This procedure enables your Windows laptop running ROS in Docker to fully control and receive data from an ESP32 robot without requiring a Raspberry Pi.

*Ensure all devices are on the same network and IPs remain consistent or are statically assigned.*

**Prepared for:** Autonomous ESP32 Navigation Deployment

**Date:** June 13, 2025

**Author:** [Your Name]