NUS

National University
of Singapore

NSWS SpiderPi Documentation
Setting Up SpiderPi, MQTT Connection, Cartographer SLAM

1. Customize WiFi Connection Upon Rebooting
   Upon rebooting, the Hiwonder RPi has already set its hotspot without an internet connection. If you wish to connect the SpiderPi to your hotspot automatically, do the following:
   a. Enter this command on the RPi terminal:
      ```
      sudo nano /etc/wpa_supplicant/wpa_supplicant.conf
      ```
      You shall see the following:
      ```
      ctrl_interface=DIR=/var/run/wpa_supplicant GROUP=netdev
      update_config=1
      country=SG
      ```
   b. Add the following to the file:
      ```
      network={
            ssid="YOUR_SSID"
            psk="YOUR_PASSWORD"
            key_mgmt=WPA-PSK
      }
      ```
      Customize to your hotspot name and the password and save the file.
   c. Enter these commands on the RPi terminal:
      ```
      sudo ifconfig wlan0 up
      sudo systemctl restart dhcpcd
      sudo reboot
      ```
   d. Check that the RPi indeed connects to your hotspot upon rebooting.

2. Setting Up MQTT Broker and Client Connection
   You must ensure that the RPi is connected to the internet before proceeding to set up communication media to your remote PC. Do the following:
   a. Enter the following commands on the RPi terminal:
      ```
      sudo apt install ufw
      sudo ufw enable
      sudo apt install mosquitto mosquitto-clients
      sudo systemctl enable mosquitto
      sudo ufw allow 1883
      sudo ufw allow 8883
      pip3 install paho-mqtt typing-extensions
      ```
   b. On the remote PC terminal, enter the following commands:
      ```
      sudo apt install mosquitto mosquitto-clients
      sudo systemctl enable mosquitto
      pip3 install paho-mqtt
      ```
   c. To test the connectivity of the Broker and the Client, enter this on the remote PC terminal:
      ```
      mosquitto_sub -h [IP-ADDRESS OF RPI] -t test/topic
      ```
      and on the RPi terminal:
      ```
      mosquitto_pub -h [IP-ADDRESS OF RPI] -t test/topic -m
      "Hello, MQTT"
      ```
      Ensure that you are connected to the same network.

WARNING: Disable the firewall after doing this unless you want to restart everything from writing an image again:

```
sudo ufw disable
```

3. Setting Up Docker, ROS Melodic, RPLidar A1, and IMU on RPi
   As we are using lidar and the driver is available and maintained on ROS, we must set up a docker. Do the following:
   a. Enter the following commands on the RPi terminal:
   ```
   curl -fsSL https://get.docker.com -o get-docker.sh
   sudo sh get-docker.sh
   sudo usermod -aG docker $USER
   newgrp docker

   sudo systemctl enable docker
   sudo systemctl start docker
   sudo chmod a+rw /dev/ttyUSB0
   ```
   b. Create a directory docker_ros and navigate to the directory. You may do that by following:
   ```
   mkdir docker_ros
   cd docker_ros
   ```
   c. We will create a Dockerfile. Follow this:
   ```
   nano Dockerfile
   ```
   Copy and paste the code from this link and save the file.
   d. In the same directory, follow this:
   ```
   nano mqtt_publisher.py
   ```
   Copy and paste the code from this link and save the file. Remember to change the BROKER_ADDRESS to your RPi address. Also, do the following:
   ```
   nano start.sh
   ```
   Copy and paste the code from this link and save the file. Then, make the file executable:
   ```
   chmod +x ./start.sh
   ```
   e. We will then build the docker. Follow this:
   ```
   sudo docker build -t rplidar_mqtt .
   sudo docker run -it --name rplidar_container --device=/dev/ttyUSB0 -p 8883:1883 rplidar_mqtt
   ```
   If the container has already existed, do this:
   ```
   sudo docker stop rplidar_container
   sudo docker rm rplidar_container
   ```
   and you can rerun the container.
   f. Then, change the Mpu6050.py file on the directory /home/pi/SpiderPi/HiwonderSDK by copying and pasting the code from this link.
   g. (Optional) If you wish to access the container, execute this:
   ```
   sudo docker exec -it rplidar_container /bin/bash
   ```
   Once you access the container terminal, you can run these lines:
   ```
   roslaunch rplidar_ros rplidar_a1.launch
   python3 /root/ws/src/mqtt_publisher.py
   ```

4. Create a Start-up Script to Bring Up SpiderPi
   We will create a script to bring up the SpiderPi servers, Docker files, and the MQTT publisher nodes. Do the following:
   a. On the RPi terminal, do the following:
      ```
      cd /home/pi
      nano startup.sh
      ```
      Copy and paste the code from this link and save the file. Make the file executable:
      ```
      chmod +x ./startup.sh
      ```
   b. Then, do the following:
      ```
      nano secondary.sh
      ```
      Copy and paste the code from this link and save the file. Make the file executable:
      ```
      chmod +x ./secondary.sh
      ```
   c. Add to the startup service:
      ```
      crontab -e
      ```
      and add the following line to the file:
      ```
      @reboot /home/pi/startup.sh
      ```
   d. Reboot the RPi to see whether the startup script works or not:
      ```
      sudo reboot
      ```

5. Setting Up MQTT Receiver for ROS2 Humble on Remote PC
   We are interested in receiving the data from the SpiderPi through MQTT to be processed on ROS2 Humble in our remote PC. Do the following:
   a. Create a workspace and navigate to it on the terminal:
      ```
      mkdir spiderpi_ws
      cd spiderpi_ws
      mkdir src
      cd src
      ```
   b. Clone the following repository:
      ```
      git clone git@github.com:MikejR2904/SpiderPi.git
      ```
      and ensure that there are 4 folders of files and 3 startup scripts in your directory. You shall customize the RPi address on the files in the spiderpi_mqtt/spiderpi_mqtt directory.
   c. Build the ROS2 packages:
      ```
      cd ..
      colcon build --symlink-install
      ```
   d. Execute the following code on the spiderpi_ws directory:
      ```
      source install/setup.bash
      ros2 run spiderpi_mqtt imu_node
      ```
      Open another terminal and execute the following:
      ```
      source install/setup.bash
      ros2 run spiderpi_mqtt lidar_node
      ```
      Ensure you are connected to the same network as your hotspot. You should receive the data readily. If you do not receive the data, ensure that Step 2.c. works properly. Check that there are the required ROS2 topics (`/scan` and `/imu`) on the terminal:
      ```
      ros2 topic list
      ```

6. Start IMU Filtering, Odometry Node, and Cartographer Mapping

   We are interested in filtering the IMU data and using them to estimate the displacement and velocities of the SpiderPi. Do the following after ensuring you have built the required packages from the above repository:

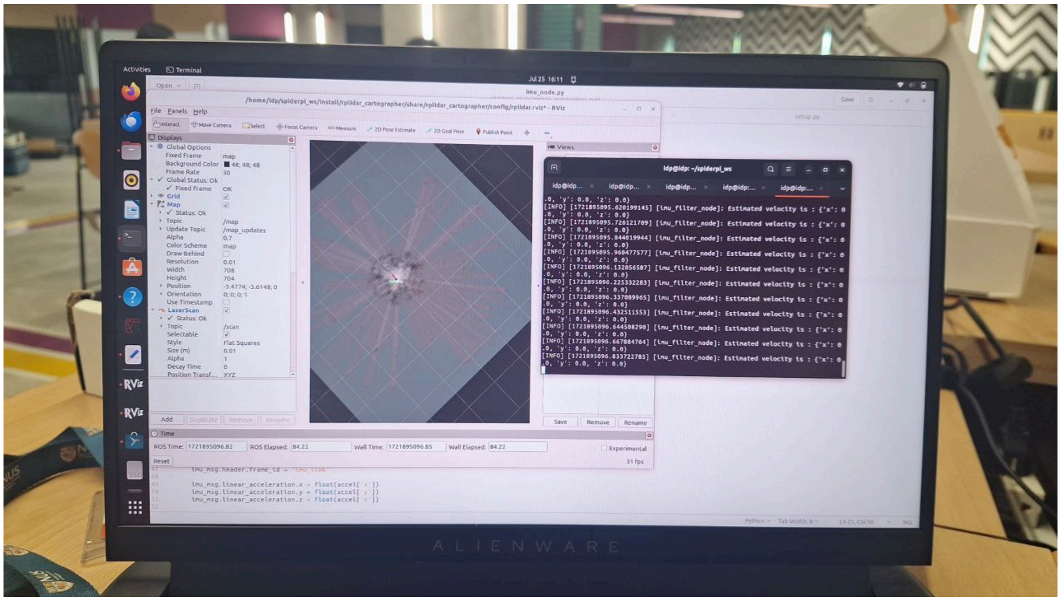
   a. Execute the following code on the spiderpi_ws directory:

   ```
   source install/setup.bash
   ros2 run spiderpi_sensors odometry_filtered_node
   ```

   b. Execute the following on another terminal:

   ```
   source install/setup.bash
   ros2 launch rplidar_cartographer cartographer.launch.py
   ```

   You shall see that RViz2 opens up automatically. You can the example below:



```
Global Options
  Fixed Frame          map
  Background Color     48; 48; 48
  Frame Rate           30
✓ Global Status: Ok
  ✓ Fixed Frame        OK
Grid                   ✓
Map                    ✓
  ✓ Status: Ok
  Topic                /map
  Update Topic         /map_updates
  Alpha                0.7
  Color Scheme         map
  Draw Behind          □
  Resolution           0.01
  Width                708
  Height               704
  Position             -3.4774; -3.6148; 0
  Orientation          0; 0; 0; 1
  Use Timestamp        □
LaserScan              ✓
  ✓ Status: Ok
  Topic                /scan
  Selectable           ✓
  Style                Flat Squares
  Size (m)             0.01
  Alpha                1
  Decay Time           0
  Position Transf...   XYZ
```

```
ROS Time: 1721895096.82   ROS Elapsed: 84.22   Wall Time: 1721895096.85   Wall Elapsed: 84.22
```

```
.0, 'y': 0.0, 'z': 0.0}
[INFO] [1721895095.620199145] [imu_filter_node]: Estimated velocity is : {'x': 0
.0, 'y': 0.0, 'z': 0.0}
[INFO] [1721895095.726121709] [imu_filter_node]: Estimated velocity is : {'x': 0
.0, 'y': 0.0, 'z': 0.0}
[INFO] [1721895095.844019944] [imu_filter_node]: Estimated velocity is : {'x': 0
.0, 'y': 0.0, 'z': 0.0}
[INFO] [1721895095.960477577] [imu_filter_node]: Estimated velocity is : {'x': 0
.0, 'y': 0.0, 'z': 0.0}
[INFO] [1721895096.132056587] [imu_filter_node]: Estimated velocity is : {'x': 0
.0, 'y': 0.0, 'z': 0.0}
[INFO] [1721895096.225332283] [imu_filter_node]: Estimated velocity is : {'x': 0
.0, 'y': 0.0, 'z': 0.0}
[INFO] [1721895096.337089965] [imu_filter_node]: Estimated velocity is : {'x': 0
.0, 'y': 0.0, 'z': 0.0}
[INFO] [1721895096.432511553] [imu_filter_node]: Estimated velocity is : {'x': 0
.0, 'y': 0.0, 'z': 0.0}
[INFO] [1721895096.644508290] [imu_filter_node]: Estimated velocity is : {'x': 0
.0, 'y': 0.0, 'z': 0.0}
[INFO] [1721895096.667884764] [imu_filter_node]: Estimated velocity is : {'x': 0
.0, 'y': 0.0, 'z': 0.0}
[INFO] [1721895096.833722785] [imu_filter_node]: Estimated velocity is : {'x': 0
.0, 'y': 0.0, 'z': 0.0}
```

```
        imu_msg.header.frame_id = 'imu_link'

        imu_msg.linear_acceleration.x = float(accel['x'])
        imu_msg.linear_acceleration.y = float(accel['y'])
        imu_msg.linear_acceleration.z = float(accel['z'])
```