Installing ROS on the Raspberry Pi 3B+

The following [guide](https://wiki.ros.org/ROSberryPi/Installing%20ROS%20Melodic%20on%20the%20Raspberry%20Pi) was used, assuming the OS on the Raspberry Pi is Raspbian Buster. All of the following commands are executed in the terminal.

Setup ROS Repositories

1. Install the repository keys for ROS.

sudo sh -c 'echo "deb http://packages.ros.org/ros/ubuntu $(lsb\_release -sc) main" > /etc/apt/sources.list.d/ros-latest.list'

sudo apt-key adv --keyserver hkp://ha.pool.sks-keyservers.net:80 --recv-key C1CF6E31E6BADE8868B172B4F42ED6FBAB17C654

1. Ensure application packages on the Pi are up-to-date

sudo apt-get update

sudo apt-get upgrade

1. Install Bootstrap Dependencies

sudo apt install -y python-rosdep python-rosinstall-generator python-wstool python-rosinstall build-essential cmake

1. Initialize rosdep (to install dependency packages later on)

sudo rosdep init

rosdep update

Installing ROS Melodic

1. Create a catkin workspace and move to it.

mkdir -p ~/ros\_catkin\_ws

cd ~/ros\_catkin\_ws

1. Install ROS-Comm version

rosinstall\_generator ros\_comm --rosdistro melodic --deps --wet-only --tar > melodic-ros\_comm-wet.rosinstall

wstool init src melodic-ros\_comm-wet.rosinstall

1. Resolve dependencies (using rosdep)

cd ~/ros\_catkin\_ws

rosdep install -y --from-paths src --ignore-src --rosdistro melodic -r --os=debian:buster

1. Build and source ROS workspace

sudo ./src/catkin/bin/catkin\_make\_isolated --install -DCMAKE\_BUILD\_TYPE=Release --install-space /opt/ros/melodic -j1

sudo ./src/catkin/bin/catkin\_make\_isolated --install -DCMAKE\_BUILD\_TYPE=Release --install-space /opt/ros/melodic -j1

source /opt/ros/melodic/setup.bash

echo "source /opt/ros/melodic/setup.bash" >> ~/.bashrc

Notes:   
If you wish to install additional packages in ROS, you can follow the final steps “4. Maintaining a Source Checkout” in the provided guide. However, I recommend creating a new catkin workspace to install them in as rebuilding everything just to add a few more packages takes time. Creating a new catkin workspace will be covered in “Installing YDLidar-SDK on the Raspberry Pi 3B+”.

The following command should be kept in mind for subsequent package installations. It reduces the hassle for installing ROS dependencies on the Raspberry Pi.

rosdep install --from-paths src --ignore-src --rosdistro melodic -y -r --os=debian:buster

Installing YDLidar-SDK on the Raspberry Pi 3B+

Downloading and building the YDLidar-SDK

1. Install dependencies

sudo apt install cmake pkg-config

1. Download and build

cd ~/catkin\_ws/src //

git clone <https://github.com/YDLIDAR/YDLidar-SDK.git>

cd YDLidar-SDK/build

cmake ..

make

sudo make install

1. Testing

cd ~/YDLidar-SDK/samples

./ydlidar\_test

Notes:   
During testing, ensure that the connections are secured.

If the connections are secured but the health status is not successful in the testing phase, make sure sufficient power is provided to the YDLidar. This can be done by connecting a power source to the micro-sub power port of the adapter.

Installing ROS YDLidar Driver on the Raspberry Pi 3B+

All of the following commands are executed in the terminal.

Creating a catkin workspace

The following [guide](https://wiki.ros.org/ROS/Tutorials/InstallingandConfiguringROSEnvironment) was used.

1. Create catkin workspace

mkdir -p ~/catkin\_ws/src

cd ~/catkin\_ws/

catkin\_make

1. Source catkin workspace

source devel/setup.bash

echo "devel/setup.bash" >> ~/.bashrc

Downloading and building ROS YDLidar Driver

The following [guide](https://github.com/YDLIDAR/ydlidar_ros_driver) was used.

1. Clone and build the package

cd ~/catkin\_ws/src   
git clone <https://github.com/YDLIDAR/ydlidar_ros_driver.git>cd ..   
catkin\_make

1. Testing

roslaunch ydlidar\_ros\_driver X4.launch

Notes:   
View the previous section’s, “Installing YDLidar SDK on the Raspberry Pi 3B+” notes.

Installing Arduino IDE & rosserial on the Raspberry Pi 3B+

All of the following commands are executed in the terminal.

Installing Arduino IDE

The following [guide](https://www.raspberrypi-spy.co.uk/2020/12/install-arduino-ide-on-raspberry-pi/) was used.

1. Download the Linux ARM 32 bits version from the [Arduino website](https://www.arduino.cc/en/software).
2. Unzip the file (replace the hashtags with the version number)

cd ~/Downloads

tar -xf arduino-####-linuxarm.tar.xz

1. Move and install the IDE

sudo mv arduino-#### /opt

sudo /opt/arduino-####/install.sh

Installing rosserial in ROS

The following [guide](https://wiki.ros.org/rosserial_arduino/Tutorials/Arduino%20IDE%20Setup) was used.

1. Download and build rosserial

cd ~/catkin\_ws/src

git clone https://github.com/ros-drivers/rosserial.git

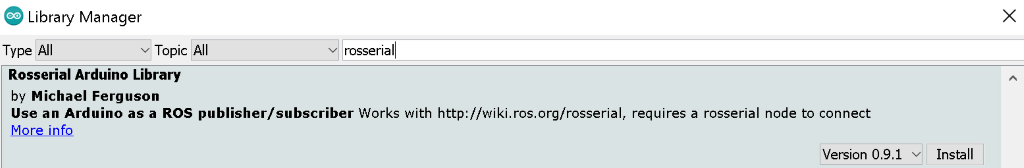
cd ..

catkin\_make

catkin\_make install

Installing rosserial in Arduino

1. Open up the Arduino IDE and go to Tools/Manage Libraries
2. Search for “rosserial” and install the library (shown below)

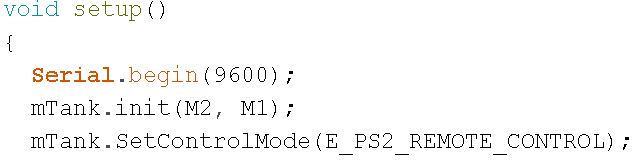


Creating a map using Hector-SLAM

All of the following commands are executed in the terminal.

1. Upload the [ino source code](https://github.com/keywish/keywish-panther-tank) to the Arduino, making sure that mTank.setControlMode is set as shown below

(keywish-panther-tank/Lesson/PantherTank\_AllFunction/PantherTank\_AllFunction.ino) or from our github source code:



1. Start the Lidar

roslaunch ydlidar\_ros\_driver X4.launch

1. Start Hector-SLAM

roslaunch hector\_slam\_launch tutorial.launch

1. Navigate around the room using the PS2 controller
2. Once the map is created, save it

rostopic pub syscommand std\_msgs/String "savegeotiff"

Running the Navigation Stack (simulation)

The following [guide](http://moorerobots.com/blog/post/3) was used. All of the following commands are executed in the terminal.

Create a mapping of the Gazebo world

1. Launch Gazebo world (terminal 1)

roslaunch mybot\_gazebo mybot\_world.launch

1. Start map building (terminal 2)

roslaunch mybot\_navigation gmapping\_demo.launch

1. Launch rviz (terminal 3)

roslaunch mybot\_description mybot\_rviz\_gmapping.launch

1. Start teleop to move the simulated robot (terminal 4)

roslaunch mybot\_navigation mybot\_teleop.launch

1. Save the map

rosrun map\_server map\_saver -f ~/mybot\_ws/src/mybot\_navigation/maps/test\_map

Load the map and Navigation Stack

1. Launch Gazebo world (terminal 1)

roslaunch mybot\_gazebo mybot\_world.launch

1. Start map (terminal 2)

roslaunch mybot\_navigation amcl\_demo.launch

1. Launch rviz (terminal 3)

roslaunch mybot\_description mybot\_rviz\_amcl.launch

1. Using the 2D Nav Goal in the rviz interface, you can specify the location for the robot to attempt to reach, and it will autonomously navigate there



Receiving commands from the website

The following [guide](https://wiki.ros.org/navigation/Tutorials/SendingSimpleGoals) was used. All of the following commands are executed in the terminal.

1. Download simple\_navigation\_goals package from the github source code to ~/catkin\_ws/src
2. Build the package

cd ~/catkin\_ws   
catkin\_make

1. Launch Gazebo world (terminal 1)

roslaunch mybot\_gazebo mybot\_world.launch

1. Start map (terminal 2)

roslaunch mybot\_navigation amcl\_demo.launch

1. Launch rviz (terminal 3)

roslaunch mybot\_description mybot\_rviz\_amcl.launch

1. Launch rosbridge (terminal 4)

roslaunch rosbridge\_server rosbridge\_websocket.launch

1. Run code

rosrun simple\_navigation\_goals simple\_navigation\_goals

1. On the website, click on the “return home” button to send a goal for the simulated robot to navigate to. The robot will then autonomously navigate there.