Open Source Rover: Software Instructions

Authors: Michael Cox, Eric Junkins, Olivia Lofaro

Jet Propulsion Laboratory
California Institute of Technology

Contents

1 Flashing the Arduino Code

In this section we will be flashing the code that runs on the arduino to control the LED matrix in the head. The following steps should be performed on your laptop or development machine (not the raspberry pi)

1. Install the Arduino IDE used for loading code onto the arduino:

   https://www.arduino.cc/en/Main/Software

2. Clone the code repo:

   (a) git clone https://github.com/nasa-jpl/osr-rover-code.git

3. Build our custom library:

   (a) Select the downloaded Arduino folder from within the osr-rover-code repo and create a .ZIP file from it

   (b) Rename the Zip file to OsrScreen.zip

4. Load the sketch onto the Arduino

   (a) Unplug the Arduino shield JST cable so the Arduino isn't powered by the control board

   (b) Connect the Arduino to your development machine with USB cable

   (c) Open Arduino IDE

   (d) Select Sketch - Include Library - Add .Zip Library

   (e) Select the OsrScreen.zip folder created previously

   (f) Click the Upload button in the Sketch Window

5. To load the example in the Arduino IDE:

   (a) File - Examples - OsrScreen - OsrScreen

2 Setting up the Raspberry Pi 3

In this section, we'll go over setting up the Raspberry Pi and setting up all the code that will run the rover. Our rover uses ROS (Robotic Operating System), which will be installed on the Raspbian operating system; we will set up both of these below. We will also set up the bluetooth pairing from the android device.

Note that you are free to use other versions of Raspberry Pi, ROS, or OS, but setting these up is not covered here and it is not guaranteed that those will work.

2.1 Getting the Raspberry Pi running

The first step is to install Raspbian on your Raspberry Pi. To install Raspbian, we recommend following the "Getting started with Raspberry Pi" tutorial below. ****NOTE: These instructions were tested and verified using Raspbian Stretch. Newer versions of Raspbian (such as Buster) may cause complications with these instructions. We therefore recommend installing Raspbian Stretch instead of the latest/default version of Raspbian, which the tutorial below will recommend. Do not flash NOOBS onto your card; instead download the Raspbian Streth image below and skip the NOOBS step in the tutorial.** In addition, we have created a pre-built image with both Raspbian Stretch and ROS already installed. You can download this image as a last resort if you can't get Raspbian and ROS installed and working on your own, but please note that you will miss out on some the learning opportunities that come with debugging a software installation by doing so.

Tutorial for installing the latest version of Raspbian on a new Raspberry Pi:

https://projects.raspberrypi.org/en/projects/raspberry-pi-getting-started

Direct downloadable Raspbian Stretch image:

https://downloads.raspberrypi.org/raspbian/images/raspbian-2019-04-09/2019-04-08-raspbian-stretch.zip

Custom JPL image with both Raspbian Stretch and ROS pre-built:

https://drive.google.com/drive/folders/1RbYWDRthpcktqkPEHb7qVqDiy27EiAc1

****NOTE: If you install the Custom JPL image with Raspbian Stretch and ROS pre-built, skip to step 2.4.**

Once you finish the above tutorial and install Raspbian Stretch, you should be able to boot your Raspberry Pi and see a desktop!

2.2 Installing ROS

If you do not download the pre-built Raspbian Stretch / ROS image above, you will need to build ROS yourself:

First, make sure you update all the packages on your pi:

```
sudo apt-get update
```

```
sudo apt-get upgrade
```

(The above commands may take a few minutes to run.)

Next, we will install ROS (and specifically, the version called 'Kinetic').

NOTE: Follow the ROS installation directions in the link below CAREFULLY! Do not ignore warnings, as they will lead to issues later in the installation. Some known "gotchas" to note before you get started are:

- Make sure you install the Kinetic "desktop_full" version of ROS, not the desktop ros-comm, or other versions.

- ROS changed their GPG keys as of June 7 2019. The ROS install instructions do not mention this as of July 2019 and will not work unless you update your GPG keys. See https://discourse.ros.org/t/new-gpg-keys-deployed-for-packages-ros-org/9454 for instructions on how to update your GPG keys.

- Make sure you follow directions for KINETIC, not Indigo, Melodic, or any other ROS version.

Once you familiarize yourself with the above issues, go ahead and complete your ROS installation by following the instructions at:

http://wiki.ros.org/ROSberryPi/Installing%20ROS%20Kinetic%20on%20the%20Raspberry%20Pi

2.3 Setting up serial communication

In this project we will be using serial communication to talk to the motor controllers. Serial communication is a communication protocol that describes how bits of information are transferred between one device and another. You can find more information on serial communication at:

- https://learn.sparkfun.com/tutorials/serial-communication

Run the following commands on the Pi to setup/enable the serial communication for the RoboClaws:

```
sudo raspi-config
```

In the raspi-config menu, set the following options:

- Interface Options $->$ Serial

- Would you like a login shell to be accessible over serial? $->$ No

- Would you like the serial port hardware to be enabled? $->$ Yes

- Would you like to reboot now? $->$ Yes

Once the Pi reboots, open up a terminal again and look at the serial devices:

```
ls -l /dev/serial*
```

Make sure that this shows serial0 -> ttyS0 . If it does not, ensure that you have followed every step in this tutorial in order. Next, edit the /boot/cmdline.txt file:

```
sudo nano /boot/cmdline.txt
```

Change **ONLY** the part with "console = ..." to read "console=tty1" and remove any other instance where it references console. The first bit of that line should look similar to the Figure 1: [1]

```
dwc_otg.lpm_enable=0 console=tty1 root=/dev/mmcblk0p7
```

Figure 1: boot/cmdline.txt File

Below is a link to help with the above steps if you need additional help:

- https://spellfoundry.com/2016/05/29/configuring-gpio-serial-port-raspbian-jessie-including-pi-3/.

Once you've completed the above steps, reboot the Pi.

```
sudo reboot
```

2.4 Building the Rover Code to work with your ROS installation

On the Raspberry Pi, open up a terminal **(ctl + alt + t)** and then type the following commands[2]:

Create a catkin workspace which will contain all ROS compilation and source code files:

Create a catkin workspace directory and move into it:

```
mkdir -p ~/osr_ws/src && cd ~/osr_ws
```

Build a basic, empty catkin project: `catkin_make`

If this doesn't report any errors, check if there are two new directories `~/osr_ws/build` and `~/osr_ws/devel`

Source your newly created ROS environment:

```
source /opt/ros/kinetic/setup.bash
```

[1] It is okay if it does not exactly match what we show here; the important part is that the "console=tty1" flag matches.
[2] In this document, terminal commands will be `highlighted`.

```
source ~/osr_ws/devel/setup.bash
```

Use nano (or your editor of choice) to add the above two 'source' commands to the end of your ∼/.bashrc file, each on its own line. This will run those lines for you every time you open a new terminal window so you don't have to manually run those each time.

```
sudo nano ~/.bashrc
```

Now continue the installation steps at https://github.com/nasa-jpl/osr-rover-code.

2.5 Init Script

Starting scripts on boot using ROS can be a little more difficult than starting scripts on boot normally from the Raspberry Pi because of the default permission settings on the RPi and the fact that that ROS cannot be ran as the root user. The way that we will starting our rover code automatically on boot is to create a service that starts our roslaunch script, and then automatically run that service on boot of the robot. Further information on system service scripts running at boot can be found at:

- https://www.linode.com/docs/quick-answers/linux/start-service-at-boot/.

There are two scripts in the "Software/Init Scripts" folder. The first is the bash file that runs the roslaunch file, and the other creates a system service to start that bash script. Open up a terminal on the raspberry Pi and execute the following commands.

```
cd /home/pi/osr/Init\ Scripts
```

```
sudo cp LaunchOSR.sh /usr/bin/LaunchOSR.sh
```

```
sudo chmod +x /usr/bin/LaunchOSR.sh
```

```
sudo cp osr_startup.service /etc/systemd/system/osr_startup.service
```

```
sudo chmod 644 /etc/systemd/system/osr_startup.service
```

Your osr_startup service is now installed on the Pi and ready to be used. The following are some commands related to managing this service which you might find useful:

Desciption	Command
Start service	sudo systemctl start osr_startup.service
Stop service	sudo systemctl stop osr_startup.service
Enable service (runs on boot of RPi)	sudo systemctl enable osr_startup.service
Disable service (doesn't run on boot of RPi)	sudo systemctl disable osr_startup.service
Check status of service	sudo systemctl status osr_startup.service
View live service list	sudo journalctl -f

Note: We do not recommend enabling the service until you have verified that everything on your robot runs successfully manually. Once you enable the service, as soon as you power on the RPi it will try and run everything. This could cause issues if everything has not yet been fully tested and verified. Additionally, if you are doing development of your own software for the robot we suggest disabling the service and doing manual launch of the scripts during testing phases. This will help you more easily debug any issues with your code. Once you have fully tested the robot and made sure that everything is running correctly by starting the rover code manually via `roslaunch osr_bringup osr.launch`, enable the startup service on the robot with the command below:

```
sudo systemctl enable osr_startup.service
```

At this point, your rover should be fully functional and automatically run whenever you boot it up! Congratulations and happy roving!!