ROS + Raspberry Pi

Aaron Bestick

October 24, 2014

A bunch of you are interested in projects which involve vehicles of one type or another. The Raspberry Pi is an ideal platform to use for controlling these, since it's small, reasonably powerful, and runs Linux. This means you can program it in Python and even run ROS.

The Pi isn't powerful enough to do much in the way of image processing or other complicated control. Fortunately though, ROS plays nice with networking, allowing nodes running on multiple networked computers to function as if they were all on one machine. This means you can do all the heavy computational lifting on a laptop or desktop PC, but put a Pi on your car/quadrotor/whatever to interface with all its sensors and actuators. Connect the two together with wifi and ROS nodes on PC and the Pi can talk via normal topics and services, just as if they were all running on the same computer.

1 ROS Network Setup

Getting ROS to work over a network requires a litle bit of setup. Let's assume you have the following hardware:

- PC (your own laptop with ROS or the ROS VM installed)
- Raspberry Pi
- Wireless access point (already set up in Cory 119)
- Ethernet cable

First, connect the Raspberry Pi to the "119Robots" network advertised by the access point. Next, connect your laptop to one of the four blue "Ethernet" ports on the back of the access point. You should now have internet access on both the Pi and the PC.

Note: You can also connect your laptop to the "119Robots" wireless network with key "EE125RobotNet", but Ethernet will produce a more reliable connection if you're doing high-bandwidth stuff like video streaming.

Next, you'll need to set two more environment variables in the `.bashrc` files of every computer in your network:

- `ROS_MASTER_URI`: the address of the ROS master computer (the one running `roscore`)
- `ROS_IP`: the IP address of this specific computer

These variables are discussed more in the following sections.

1.1 PC Configuration

The PC will be the ROS *master*, meaning it will be the computer that runs `roscore` and serves as the central source of information about all the other ROS nodes running on computers connected to the network. Complete the following steps to configure it:

1.1.1 VM Specific Configuration

If you're using the ROS VM you'll need to perform an extra configuration step before you do anything else.

Start the VM, then open the `Devices>Network>Network Settings...` menu (in the host OS, not the VM). Now, change the `Attached to:` setting from `NAT` to `Bridged Adapter`. Click OK and wait while the VM's network reconnects. You can now continue with the steps below.

1.1.2 Configuration

Complete the steps below for either the VM or a native install of ROS:

1. Find the IP address of the computer by typing `/sbin/ifconfig` at the terminal. The computer's IP address will be shown in the `inet addr:` field under either the `eth0` or `wlan0` section (if the PC is connected via Ethernet or WiFi, respectively).
2. Choose a *port* number for ROS to communicate over. This can be any random integer between 1025 and 65535 (ROS uses a default of 11311, but you should change this to avoid conflicts with other students who are using the WiFi access point at the same time as you).
3. Add the following lines to your `.bashrc` file:

   ```
   export ROS_MASTER_URI=http://<ip_address>:<port>
   export ROS_IP=<ip_address>
   ```

   where `<ip_address>` and `<port>` are the values from above.

The PC is now ready to communicate with other computers running ROS over the network. When you're ready to start your ROS nodes, just start `roscore` with

```
roscore -p <port>
```

which tells it to listen for incoming connections on the specified network port. Then run your nodes as you usually would using `rosrun` or `roslaunch`.

1.2 Raspberry Pi Configuration

The Raspberry Pi must be configured similarly. Complete the following steps:

1. Find the IP address of the Pi using `ifconfig` as described above.
2. Add the following lines to your `.bashrc` file:

   ```
   export ROS_MASTER_URI=http://<ip_address>:<port>
   export ROS_IP=<pi_ip_address>
   ```

   where `<ip_address>` and `<port>` are the values from the PC Configuration section above, and `<pi_ip_address>` is the IP address you just found in step 1.

Your Raspberry Pi is now set up to run ROS over the network.

2 Using the Raspberry Pi camera

Once you have the Pi and the PC set up to talk over the network, it's easy to stream images from the Pi's camera to the PC.

1. Start `roscore` on the PC with

   ```
   roscore -p <port>
   ```

   where `<port>` is the value from the previous section.

2. Start streaming the camera data by running

   ```
   rosrun rpi_camera camera_streamer.py
   ```

   on the Raspberry Pi. This will publish data on the `/rpi_camera/image/compressed topic`.

3. Test that the data is being streamed by using `image_view`, as in earlier labs. On the PC, run

```
rosrun image_view image_view image:=/rpi_camera/image/ compressed
```

   If everything is working, you should see the images from the Pi's camera appear in near-real-time on the PC.