Progress review #8

Gauri Gandhi

Team G Robographers

Teammates:

Rohit Dashrathi

Jimit Gandhi

Tiffany May

Sida Wang

ILR #7

February 11, 2016

a. <u>**Individual Progress**</u>

For the eighth progress review, I first did the literature study of the ROCON framework and then implemented the basic talker and listener communication for the single master and the multi-master.

I. **Literature Study of the ROCON framework**

To establish the ROCON (Robotics in Concert) network among the turtlebots for swarm implementation, I first studied the basics of a multi-master system. Multi-master is primarily a means for connecting autonomously capable ROS subsystems. It could of course all be done underneath a single master - but the key point is for subsystems to be autonomously independent. This is important for wirelessly connected robots that can't always guarantee their connection on the network. Hence, this finds a clear application in our project.

In the literature study of ROCON, I first understood the gateway model for ROS multi-master on local networks. This involved learning of the concepts like flipping, advertising and pulling of the ROS API using the gateways between the different systems on the network. I also studied about the hub and Redis server that is responsible for gateway discovery, unique identification and connection. Once I was clear with basic functioning of ROCON, I tried to make some concert files, publish master information, etc. by going through the ROCON tutorials.

II. **Basic talker and listener launch**

1. **Single master on single machine**

For launching the single-master ROCON on a single machine, I made a concert file that would launch the talker and listener nodes from the rocon_tutorials package. To make it single master, I set the same port parameter, i.e., 11311 for each of the nodes but made different terminals for each of the node launch. This resulted in two spawned terminals with the talker publishing "Hello World" at 15 Hz frequency and the listener receiving the "Hello World" by printing "I heard Hello World".

2. **Multi-master on single machine**

For enabling multi-master ROCON, I set the port parameters for each launch file in the concert file to a different value. I launched the chatter.concert file using the rocon_launch that would launch one concert file acting as the hub, one talker and two listeners on different ports. This resulted in one talker, i.e., the dudette on port number 11312 publishing some information and two listeners, i.e., the dudes on port numbers 11314 and 11315 respectively receiving and printing that information on their terminal as shown in the figure 1.
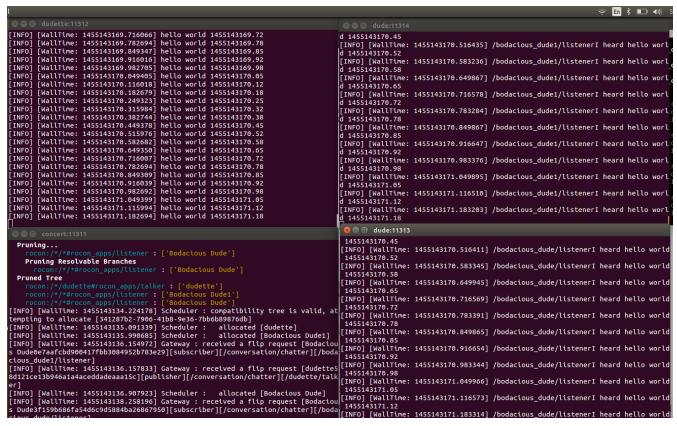
Figure 1 Multi-master ROCON on a single machine

3. Multi-master on multiple machines

For creating a concert of multiple masters on multiple machines, I followed exactly the same steps as I did for the single machine. The only difference this time was to set the local_machine_only parameter in the concert file to false at the time of rocon_launch.

rocon_launch chatter_concert chatter.concert --screen local_machine_only:=false

This again launched one dudette and two dudes on different ports of the same machine. Next, I launched the dude launch file in the second machine. This created one more listener in the concert for the same wireless network as can be seen in the figure 2.
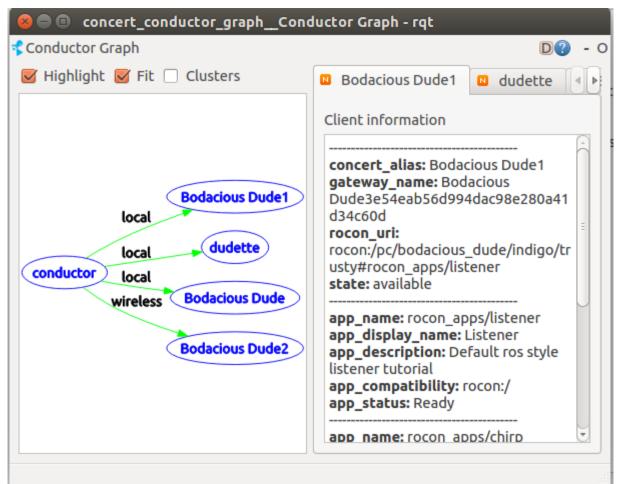
Figure 2 Concert conductor graph showing 2 local listeners and 1 listener in the wireless network

b. Challenges

I. The major challenge I faced this time was during the enabling of the zeroconf for the multi-master setup. By default, the zeroconf parameter was disabled in the chatter concert tutorial. Due to this, the listener and talker nodes were not able to communicate amongst themselves even though they were getting launched in the different ports. To solve this, I tried the following approaches:

1. First, I tried to set the zeroconf parameter to true in all the launch and concert files manually. This approach worked for the multi-master communication on a single machine but did not help for the case of multiple machines.

2. Next, I set the zeroconf parameter again to false in all the files. Then, I wrote the following to disable the avahi-daemon:

   sudo sed -i
   's/^AVAHI_DAEMON_DETECT_LOCAL=1/AVAHI_DAEMON_DETECT_LO
   CAL=0/g' /etc/default/avahi-daemon

This approach worked for both the single machine and multiple machines case. I took help from Sasanka Nagavalli's repositories to setup the ROCON for turtlebots.

II. The next major challenge was to setup the multi-master ROCON for multiple machines. There were a lot of issues in setting and configuring the ROS_MASTER_URI and ROS_HOSTNAME on both of the machines. After trying a lot of permutations and combinations of the IP addresses, I set the ROS_MASTER_URI to the localhost and the ROS_HOSTNAME to the IP address of the machines on the local wireless network. This worked for me.

c. Team Work

For this progress review, I collaborated with Sida to set up the multi-master ROCON. Rohit and Tiffany worked together to learn the concepts of Gazebo for the simulation of our project system. Tiffany created a model of the environment where we will be testing our system and also did the navigation of the turtlebots in the set environment. Rohit and Jimit worked together to set up the ROCON concert in the Gazebo simulated environment. Jimit and Sida worked on upgrading the new Chromebook to Ubuntu 14.04 and installing ROS Indigo on it.

d. Future Plans

For the next Progress Review, I will be working on the following tasks:

I. Setting up the multiple robot face detection and photo-clicking system using Intraface along with Sida

II. Setting up a new router for the turtlebot network along with Tiffany (this is an optional goal).

III. Updating the team website.

As a team, we will work to accomplish the following goals in addition to the above mentioned goals.

I. Test and run the single robot system using the Chromebook.

II. Demonstrate the multiple robot navigation on Gazebo.

e. References

1. Turtlebot ROS tutorials
wiki.ros.org

2. ROCON tutorials for ROS Indigo
wiki.ros.org/rocon/indigo/Guide

3. Gateway model tutorials
http://redmine.robotconcert.org/projects/multimaster/wiki/Gateway_Model

4. Sasanka's bitbucket repository
https://bitbucket.org/snagavallis/turtlebot-swarm-scripts/