Progress review #12

Gauri Gandhi

Team G Robographers

Teammates:

Rohit Dashrathi

Jimit Gandhi

Tiffany May

Sida Wang

ILR #11

April 13, 2016

a. <u>Individual Progress</u>

For the twelfth progress review, I first worked on upgrading the Ubuntu kernel on each of the Chromebook. Then I worked on removing the bugs from the detection sub-system. I worked with Jimit to implement overhead localization and flocking. Other than these, I also did the IMU and Gyro calibration of the turtlebots with Jimit.

I. Upgrading Ubuntu Kernel

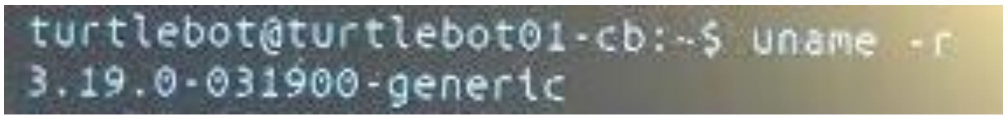
In the last progress review, we faced problems due to the recurrent crashing of the Chromebooks. The reason behind that was the Ubuntu kernel running on the three machines was not fully compatible with the ROCON framework. This problem turned out to be a major risk for us as the whole system relied on the proper functioning of the Chromebooks. So the first thing I implemented for this progress review was to upgrade the Ubuntu kernel on each of the three laptops. The kernel already running on the Chromebook was v3.18.0. I upgraded it to 3.19.0-generic. I followed the following steps for doing this:

(i) I first downloaded the linux-header-generic, linux-header-all and linux-image-generic packages for the 64 bit processor (amd64 versions).

(ii) Then I built the packages using the dpkg command. Dpkg is a package manager for Debian-based systems.

(iii) After this, I updated the Grub Bootloader and rebooted the machine.

After performing all these steps and rebooting the computer, the kernel version got upgraded to the desired version. Now the functioning of the Chromebooks is smoother than before. The system now completely supports the ROCON



```
turtlebot@turtlebot01-cb:~$ uname -r
3.19.0-031900-generic
```

network framework.

Figure 1 The upgraded kernel

II. Debugging the Detection sub-system

There were a few bugs in the detection sub-system that we integrated in the last progress review. Some of them are listed below:

(i) The photo-clicking node when integrated with the sound playing feature was not performing optimally. The sound feature was also running on a loop which was irritating for the users.

(ii) There were multiple image windows opening up on launching the complete system. This was affecting the performance. This was also creating problems for the server computer to handle the inputs from the three Chromebooks.

(iii) The small window that was being created to include only a single person in the frame was not giving the desired results for all the use cases.

So, to solve these issues, we divided the tasks amongst us. I worked on the first issue of photo-clicking and sound-play. Sida worked on the second and the third issue.
As I mentioned above, both the ‘Say Cheese’ and ‘Nice Smile’ sound commands were running on a loop in the best photo clicking node in the master computer. So to solve this problem, I included various check flags in my code. This resulted in the ‘Say Cheese’ voice command running only after the subscribed topic for smile confidence score was receiving some message from any of the three turtlebots. Also, to stop the ‘Nice Smile’ voice command from running on a loop, I used the stopsaying() method from the Sound Client class in the sound play package. To make it more efficient, Jimit and I added some extra flags in the code so that the system handles most of the use cases.

III. Overhead localization and Flocking

After I was done with the above mentioned tasks, I worked with Jimit in implementing the localization and the flocking algorithm. We tried multiple scenarios in the following order for implementing flocking:

(i) Flocked two turtlebots without any master turtlebot guiding the flock.
(ii) Flocked three turtlebots with one of them being tele-operated using keyboard to guide the flock.

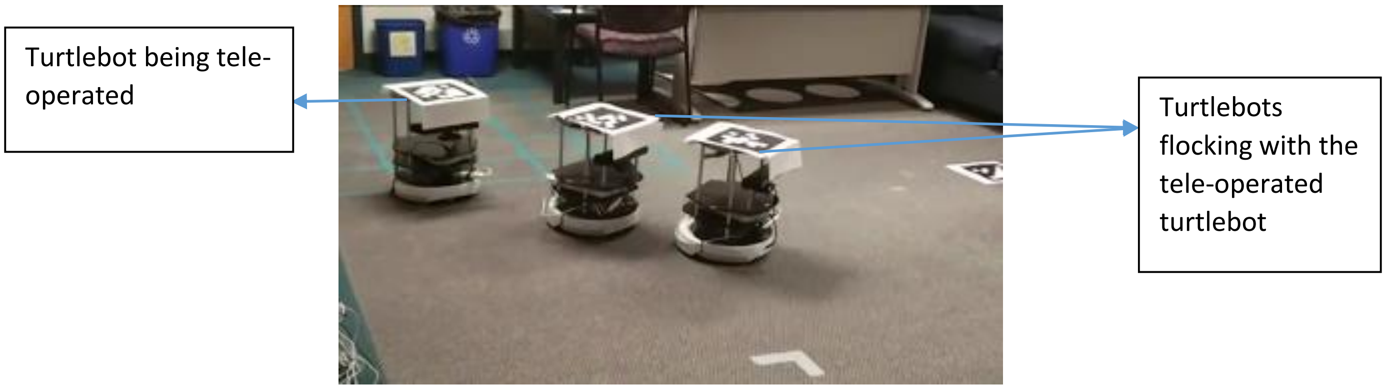

Turtlebot being tele-operated
Turtlebots flocking with the tele-operated turtlebot

Figure 2 *Flocking three turtlebots*

(iii) Flocked three turtlebots with one of them being one of the three turtlebots with the pan-tilt-elevation units installed. The master guiding the flock this time was the one with the PTZ unit. The master rotated and navigated to the person of interest by getting the pose coordinates from the April Tag on the person.

b. Challenges

- **I.** The first challenge I face this time was during the upgradation of the Ubuntu kernels. During my first few tried, I found out that Ubuntu was giving an error message of 'kernel panic' on each startup. I had to switch back every time to the previously installed kernel to boot the machine. After some time, I found that I had mistakenly downloaded the i836 version of the kernel that was meant for the 32 bit machines.
- **II.** Another issue I faced was when Jimit and I were implementing flocking. We noticed that the robots were not flocking unless there were five robots in the environment. To make the three robots flock independently, we had to keep the other two robots outside the flocking scope of the robots. It means, that the rest two turtlebots should not show up as neighbors for the main three robots. To overcome this problem, we changed the bash settings in the server such that the catkin workspace of the other two robots was not getting synced with the running system.

c. Team Work

For this progress review, I collaborated with Jimit to work on flocking. Rohit worked on reprogramming and calibrating all the three pan-tilt units. Now all three pan-tilt units are functional. Sida worked on dealing with the window size issues in the detection sub-system. Tiffany worked on programming the pan-tilt according the position of the April Tag on the person of interest. She also worked with Sida to merge the Arduino part with ROS using rosserial.

d. Future Plans

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

- **I.** Implement aligning of the three turtlebots around the person of interest. I will assist Jimit in this task.
- **II.** Integrating the navigation sub-system with the detection sub-system.

As a team, we will finish the following goals:

- **I.** Calibrating all the new cameras.
- **II.** Integrating and testing the pan-tilt functionality with ROS.
- **III.** Improving the aesthetics.
- **IV.** Testing the complete system for all the specified use cases.

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/
5.Ubuntu Kernel support
http://kernel.ubuntu.com/