Progress review #3

Gauri Gandhi

Team G Robographers

Teammates:

Rohit Dashrathi

Jimit Gandhi

Tiffany May

Sida Wang

ILR #4

November 12, 2015

a. Individual Progress

For the third progress review, I worked on the Turtlebot odometry and gyro calibration and the detection of April Tags through camera. Meanwhile, I also accomplished the teleoperation of the turtlebots through keyboard and also worked with some some basic application of the MS Kinect. I also worked on the team assignments like the PDR presentation and the PDS PCB design.

I. Turtlebot Caliberation

In the last progress review, I could not accomplish the calibration of the turtlebots but was done with the bringing up of the turtlebots both on the workstation (my laptop) and the netbook of the turtlebot. To avoid network issues like the last time, this time we decided to continue all the work related to the project in the Advanced Agents Lab on the first floor of NSH. So I started working with the turtlebot 3 on the Advanced Agent's lab network itself. On this network, the IP addresses of both the master(netbook) and the slave(workstation) were determined as follows using the ifconfig command:

1. Workstation: 192.168.1.120
2. Netbook: 192.168.1.53

So I made the following changes in the .bashrc file of the workstation:

$ echo export ROS_MASTER_URI=http://192.168.1.53:11311

$ echo export ROS_HOSTNAME=192.168.1.120

After that, we took Jaineel Dalal's help to launch the calibration node. We used the calibrate1.launch file created by their team on each of the netbooks. I wrote the following commands on the terminal of the workstation:

$ ssh turtlebot@turtlebot03

$ roslaunch turtlebot_bringup minimal.launch

Then, I set the gyro_measurement_range to 250 by using rqt_reconfigure (figure 1):

$ rosrun rqt_reconfigure rqt_reconfigure

Then in another terminal window, I launched the calibration node:

$ roslaunch turtlebot_calibration calibrate.launch

Running this node gave the correction factors to be multiplied with the odometry and gyro measurements values in the dynamic reconfigure window. After multiple iterations of running the calibration node and multiplying the correction factors, I got the final calibration factors close to 1.

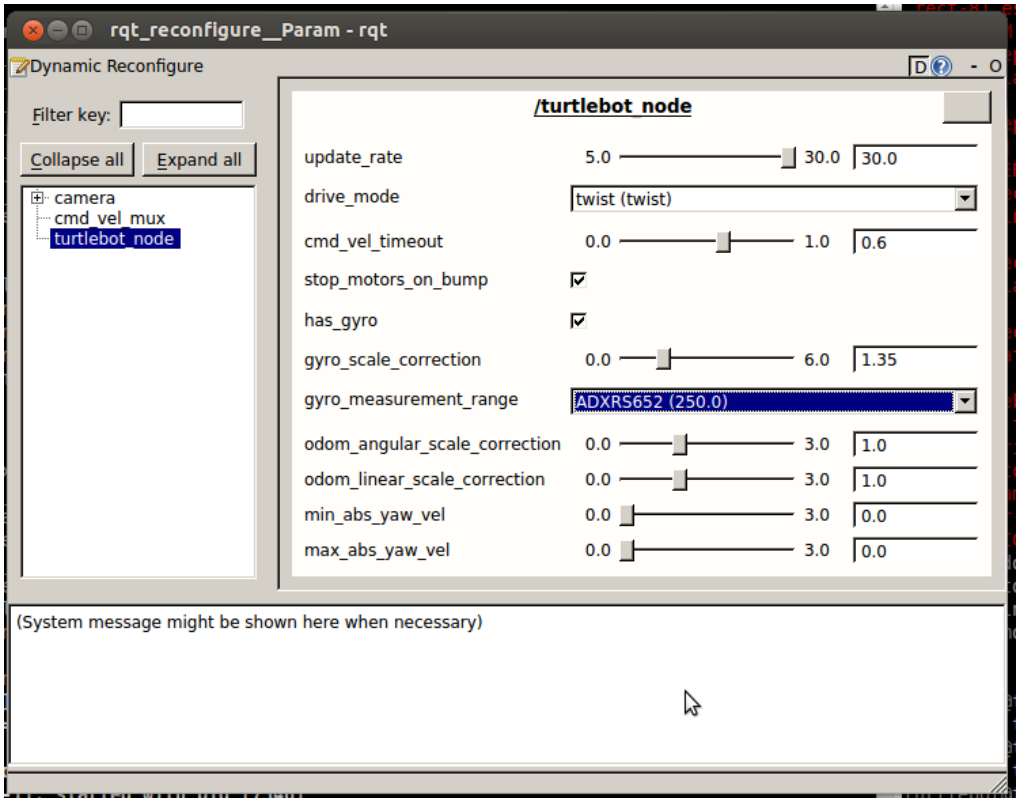

Figure 1 rqt_reconfigure

II. April tag detection

My second goal for this progress review was to detect April Tags using the Kinect. So I downloaded the open source April Tag C++ Library by Michael Kaess using the following command:

$ sudo apt-get install subversion cmake libopencv-dev libeigen3-dev libv4l-dev

Then I compiled it as follows:

$ cd apriltags

$ make

To check the working, I ran the demo code for the detection of the april tags using the webcam of the laptop.

$ ./build/bin/apriltags_demo

This resulted in the april tag being detected by the camera. It correctly identified the tag id of the april tag correctly and also gave the estimate of the six degrees of freedom of the tag, i.e., x, y, z, roll, pitch and yaw (figure 2 and 3). So we finally decided to continue the human detection using the april tags and the webcam for the Fall Validation Experiment.

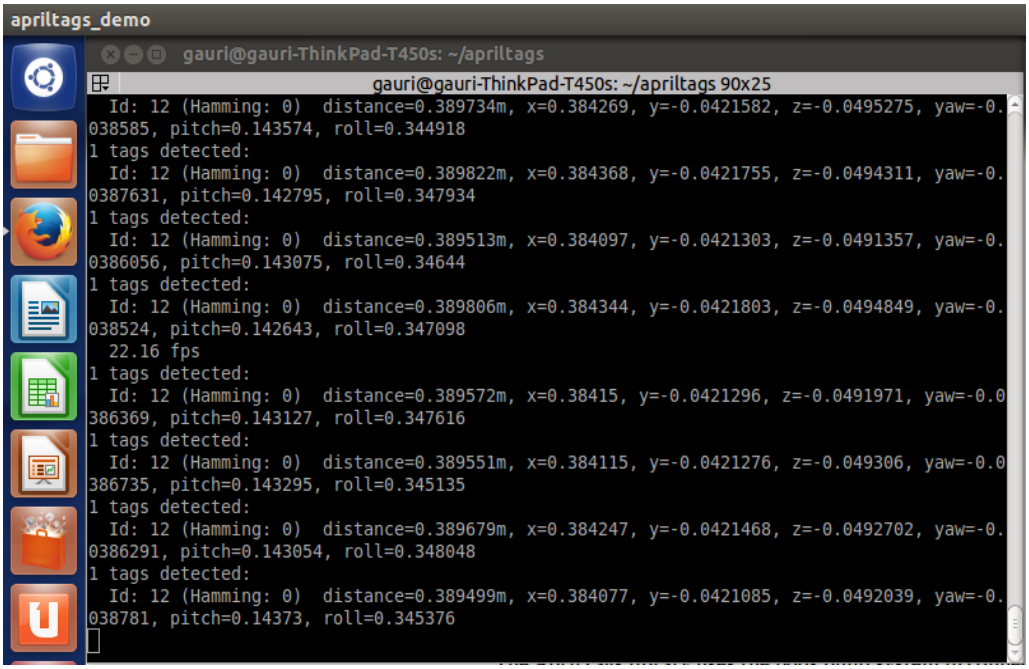

Figure 2 Values obtained from the detection of the april tag

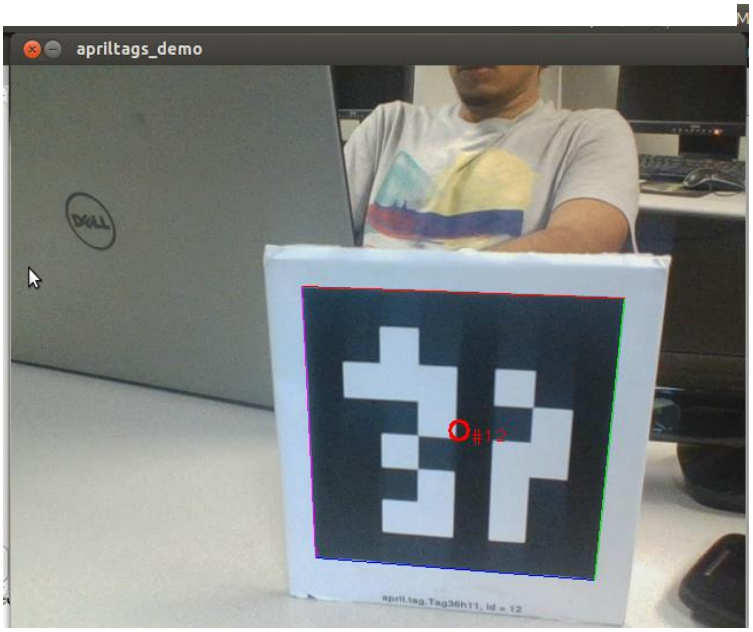

Figure 3 April tag used for detection

III. Turtlebot teleoperation

To get ourselves more equipped with the turlebots and ros, we tried running the teleoperation node using the keyboard of the workstation. To do this, we wrote the following on the terminal:

roslaunch turtlebot_teleop keyboard_teleop.launch

Key presses in this terminal could now be used to control the robot.

IV. Playing with Kinect

To get more equipped with the Kinect operations, I tried to understand its basic features and functions. Initially, I started working with the turtlebot Kinect and was using the ROS libraries for its operation. After doing the minimal launch, I launched the 3dsensor node for the turtlebot.

$ roslaunch turtlebot_bringup 3dsensor.launch

Then I opened rviz to see the 3D visualizations from the kinect:

$ roslaunch turtlebot_rviz_launchers view_robot.launch

Then I chose the topics I wanted to display from the rviz window (figure 4).

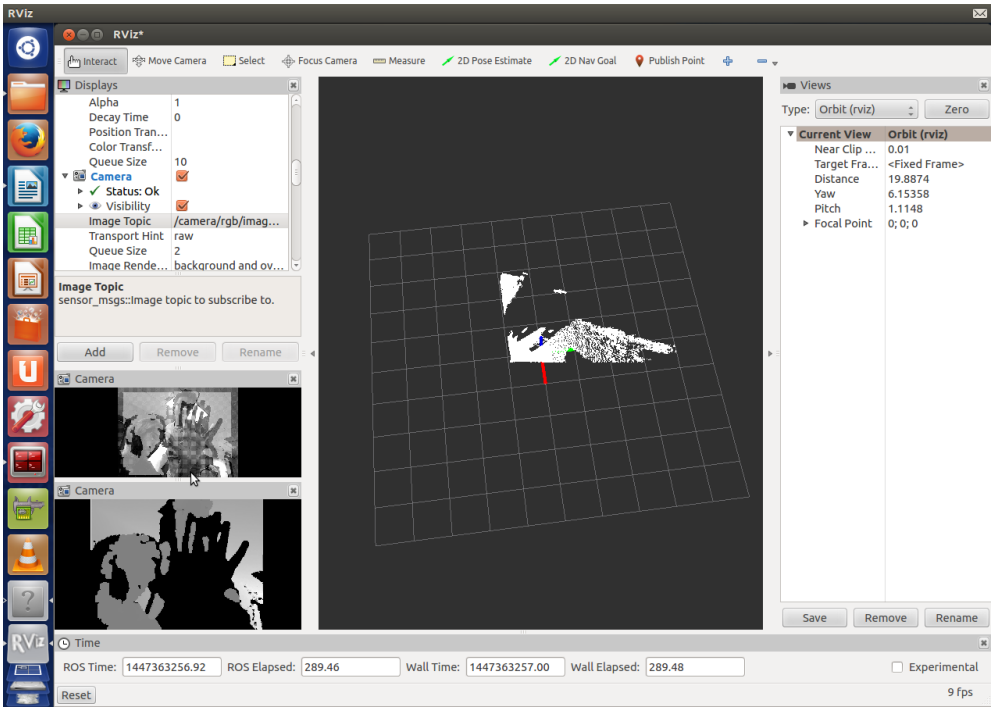

Figure 4 This window shows the actual image obtained through kinect, the depth image and the point cloud on the grid

Then I switched to a new Kinect independent from the turtlebot hardware (obtained from Sasanka Nagavalli) and connected it to my laptop. This time I used the openni library to obtain the same results as above.

$ roslaunch openni_launch openni.launch

I also downloaded the ROS independent library for hooking up the Kinect with my laptop. I wrote the following commands on the terminal:

$ sudo apt-get install cmake freeglut3-dev pkg-config build-essential libxmu-dev libxi-dev libusb-1.0-0-dev python

$ sudo add-apt-repository "deb http://archive.canonical.com/ lucid partner"

$ sudo apt-get update
$ sudo apt-get install sun-java6-jdk
Then I installed the graphviz to view the output.
$ sudo apt-get install doxygen mono-complete graphviz
# in libfreenect directory, in the KinectLibs dir
$ mkdir build
$ cd build
$ c make ..
$ make
$ sudo make install
$ sudo ldconfig /usr/local/lib64/
$ sudo chmod a+rw /dev/bus/usb//
$ sudo chmod a+rw /dev/bus/usb//
After that I ran glview:
$ glview
This gave me the output as shown in figure 5.

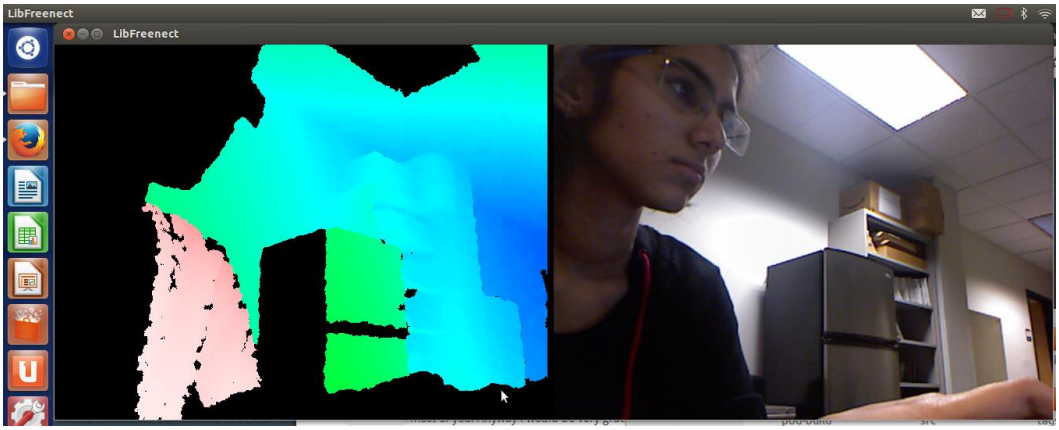

Figure 5 Kinect output on glview

b. Challenges

I. The first challenge we faced was in launching the calibration nodes for the turlebots. So, Jaineel helped us with that. He changed the depth_registration topic to true in the source code of the node and asked us to use the calibrate1.launch file created by his team last year. That worked for us.

II. As the turtlebots are old, the calibration results are not accurate enough to perform precise path planning.

III. I could not figure out how to use the C++ april tag library with the Kinect. Therefore I downloaded the C++ library for the Kinect and it still didn't work. So, if required to detect the april tags using the Kinect in future, I will write a new

node in ROS that will subscribe to the Kinect and detect the april tags using the transfer function given in its library.

IV. Initially I was not getting the depth registration cloud and the point cloud from the Kinect on the turtlebot. But when I used the openni launch file it worked. Jaineel told that this also had the same problem of the depth_registration topic as the calibration.

V. While using the independent Kinect hardware, the rviz was giving the error that the frame doesn't exist. So I typed the following to specify the tf for the gridmap and it worked:

$ rosrun tf static_transform_publisher 0.0 0.0 0.0 0.0 0.0 0.0 map my_frame 100

c. Team Work

For this progress review, I worked with Jimit to calibrate and tele operate the turtlebots. Jimit also wrote the node for clicking the photo autonomously from the incoming video stream. Rohit worked on the CAD model of the elevation rod for the pan-tilt unit. He also revised the project schedule. Tiffany worked on integrating the Arduino code for running the pan-tilt motors with the matlab code for face detection. Sida worked on the 3D reconsruction of images obtained from the multiple cameras. We also finished the PDR presentation by completing the parts being allotted to each one of us. I worked with Jimit and Rohit to compile the complete presentation. For the PDS PCB task also, we divided the tasks. I wrote the conceptual review report. Tiffany designed the circuit for the power board on the paper and also did the heat and power analysis. Rohit created the required libraries. Jimit designed the schematic. I worked on the layout design along with Jimit. Then I generated the Bill of Materials. Rohit finally compiled everything and generated the manufacturing files for the board. Apart from all these, Rohit, Sida, Tiffany and Jimit worked on setting up the environment for the Fall Validation Experiment.

d. Future Plans

Before the next progress review, I will be working on the following tasks:

I. Obtain the parameters from the april tags for navigation.

II. Determine the distance from the human.

III. Navigate autonomously towards the human detected.

IV. Update the team website.

Also, Rohit will be working on the manufacturing and assembly of the elevation axis for the pan-tilt unit. Jimit and I will work together on the navigation part. He will also complete the clicking image code. Sida and Tiffany will delve further into the Intraface features to properly dtect the human face and headpose. Rohit and Tiffany will work together on integrating the Arduino and matlab codes for running the pan tilt unit according to the detected head pose.

e. References

1. Turtlebot ROS tutorials
wiki.ros.org
2. April tag C++ library by Michael Kaess
http://people.csail.mit.edu/kaess/apriltags/
3. Setting up Kinect for running with Ubuntu
http://www.kdab.com/setting-up-kinect-for-programming-in-linux-part-1/
4. 2014 Team C Roborn's ILRs
https://sites.google.com/site/mrsdproject201415teamc/project-documents