

Individual Lab Report 4

Progress Review 3

Pulkit Goyal

November 10, 2017

Team F - Falcon Eye

Pulkit Goyal

Pratibha Tripathi

Yuchi Wang

Rahul Ramkrishnan

Danendra singh

Individual Progress

I primarily contributed in running the variation localization sensors(IMU and Encoders) on Husky, configuring the WiFi network for the system, teleoperation setup of Husky and attempted to connect the Bebop2 as a client to the network instead of Host.

1. IMU

I, along with Pratibha worked on reading the data from um7 IMU. First step was to finalize the rx-tx connection for the IMU to read the data in the Husky's mini-pc. We finalized on a ftdi cable to connect the IMU to USB port on Mini-pc. Initially we tried to read the raw data from the serial port in ubuntu using cat command. We faced some permission issues, which is explained in detail in challenges section. After that we installed the ROS driver for the IMU, directly reading the rostopic echo /imu/data.

```
x: -0.9623646924
y: 0.0467620349
z: 0.2546567098
w: 0.0824462008
orientation_covariance: [0.002741552146694444, 0.0, 0.0, 0.0, 0.002741552146694444, 0.0, 0.0, 0.0, 0.007615422629706791]
angular_velocity:
x: -0.000433264905381
y: -8.01257299392e-05
z: 0.00164294646858
angular_velocity_covariance: [1.0966208586777776e-06, 0.0, 0.0, 0.0, 1.0966208586777776e-06, 0.0, 0.0, 0.0, 1.0966208586777776e-06]
linear_acceleration:
x: -0.00508858841835
y: -0.0448820292424
z: -10.3205838572
linear_acceleration_covariance: [0.0015387262937311438, 0.0, 0.0, 0.0, 0.0015387262937311438, 0.0, 0.0, 0.0, 0.0015387262937311438]
---
header:
seq: 2134
stamp:
secs: 1510367890
nsecs: 380743275
frame_id: imu_link
orientation:
x: -0.9630696477
y: 0.0457885252
z: 0.2534817843
w: 0.0779814839
orientation_covariance: [0.002741552146694444, 0.0, 0.0, 0.0, 0.002741552146694444, 0.0, 0.0, 0.0, 0.007615422629706791]
angular_velocity:
x: 9.57219808555e-05
y: 0.000906044008306
z: 0.00053180024025
angular_velocity_covariance: [1.0966208586777776e-06, 0.0, 0.0, 0.0, 1.0966208586777776e-06, 0.0, 0.0, 0.0, 1.0966208586777776e-06]
linear_acceleration:
x: -0.00966762102101
y: -0.045040169813
z: -10.3219048767
linear_acceleration_covariance: [0.0015387262937311438, 0.0, 0.0, 0.0, 0.0015387262937311438, 0.0, 0.0, 0.0, 0.0015387262937311438]
---
Administrator@teamf:/dev$ rostopic echo /imu/data
```

Fig1 - IMU reading on Terminal

As of now we are just reading the raw values of IMU, acceleration and orientation values. We need to apply Dead Reckoning to get the location of Husky.

2. Encoders

I worked on this with Pratibha. Husky's encoders are in-built and the data is received from the communication cable which is connected via Serial port between Husky and mini-pc. The Husky's ROS driver has odometry node which publishes encoders data on rostopic echo /odometry/filtered. We carried out some experiments about the drift of location data coming from odometry. The location values from Odometry were drifting significantly after 1-2 m of distance.

```

712538791432e-28, 0.0, 0.0, 0.0, -6.503955193979679e-24, -3.0307712538790953e-28, 1.0939422958858538e-06]
---
header:
  seq: 118450
  stamp:
    secs: 1510367972
    nsecs: 161909342
  frame_id: odom
child_frame_id: base_link
pose:
  pose:
    position:
      x: 0.0
      y: 0.0
      z: 0.0
    orientation:
      x: -6.37604315236e-29
      y: 3.21864788558e-28
      z: -0.442509996812
      w: 0.89676357125
  covariance: [10363820592368.445, 0.03635969590098398, 8.520283743896955e-20, 0.0, 0.0, 0.0, -0.0363586267426399, 10363820592368.445, 3.30683
42672432725e-20, 0.0, 0.0, 0.0, 8.520283743897075e-20, 3.306834267243364e-20, 4.999195507994937e-07, 0.0, 0.0, 0.0, 0.0, 0.0, 4.998391568
328793e-07, 1.6678553318393462e-31, 7.95177515430472e-21, 0.0, 0.0, 0.0, -1.6678553318393462e-31, 4.998391568328793e-07, 6.096987264989151e-21
, 0.0, 0.0, 0.0, 7.951775154304536e-21, 6.096987264989395e-21, 7465401.816061982]
twist:
  linear:
    x: 0.0
    y: 0.0
    z: 0.0
  angular:
    x: -5.04222377974e-22
    y: -2.34976604392e-26
    z: 0.00110441355615
  covariance: [22909286.992094137, -2.5624345518324913e-29, -1.0250814179884158e-24, 0.0, 0.0, 0.0, -2.562434551832472e-29, 22909286.992094137
, -7.859699579977119e-25, 0.0, 0.0, 0.0, -1.0250814179884158e-24, -7.859699579977118e-25, 4.998793380123498e-07, 0.0, 0.0, 0.0, 0.0, 0.0, 4.995180581261081e-07, 4.0383040187555145e-28, -6.5035526716876895e-24, 0.0, 0.0, 0.0, 4.0240741562499773e-28, 4.995180581261081e-07, -3.0307
712538791432e-28, 0.0, 0.0, 0.0, -6.503955193979679e-24, -3.0307712538790953e-28, 1.0939422958858538e-06]
---

```

Fig2 Encoders reading on terminal

Rahul has worked on reading the data from GPS.

We need to use EKF to fuse the location value from GPS, Encoders, IMU and finally location of april tag placed on husky to get better localization accuracy of Husky.

But as the autonomy of Husky is planned for Spring semester, we plan to take this up forward in next semester.

3. Tele-operation of Husky

I, along with Yuchi, worked on teleoperation of Husky. In this task we were required to operate the husky using the gamepad controller and gui both present on the remote pc connected to the Husky's mini-pc with local WiFi network. We executed all the ROS nodes running on husky's mini-pc normally and used the commands on remote PC to connect it to the mini-pc's ROS master on network. We executed following commands on remote-pc after connecting the gamepad controller to it:

```

Export ROS_MASTER_URI=http://192.168.1.127:11311
192.168.1.127 - IP of the Husky's pc.
11311 default port of ROS master node.

```

```
Rosrun joy joy_node _autorepeat_rate:=60
```

To read the data of the joystick connected on the remote pc at rate repeat rate of 60.

```
Rosrun topic_tools relay /joy /joy_teleop /joy
```

Relaying the joy topic as joy_teleop which will be relayed to ROS master on network.

We also had to add the hostname of the remote pc to husky's mini-pc's host file and vice-versa.

```
frame_id: ''
axes: [-0.0, 0.0, 0.0, 0.0, -0.0, 0.0]
buttons: [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
---
header:
  seq: 5915
  stamp:
    secs: 1510368762
    nsecs: 78846225
  frame_id: ''
axes: [-0.0, 0.0, 0.0, 0.0, -0.0, 0.0]
buttons: [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
---
```

Fig3 Transmitting gamepad values on network

4. Rviz

Pratibha and me, subscribed to all the topics created above in Rviz. We analyzed data from IMU and Odometry in Rviz. We also added a separate node named Interactive Markers. It helped us control the Husky with the GUI.

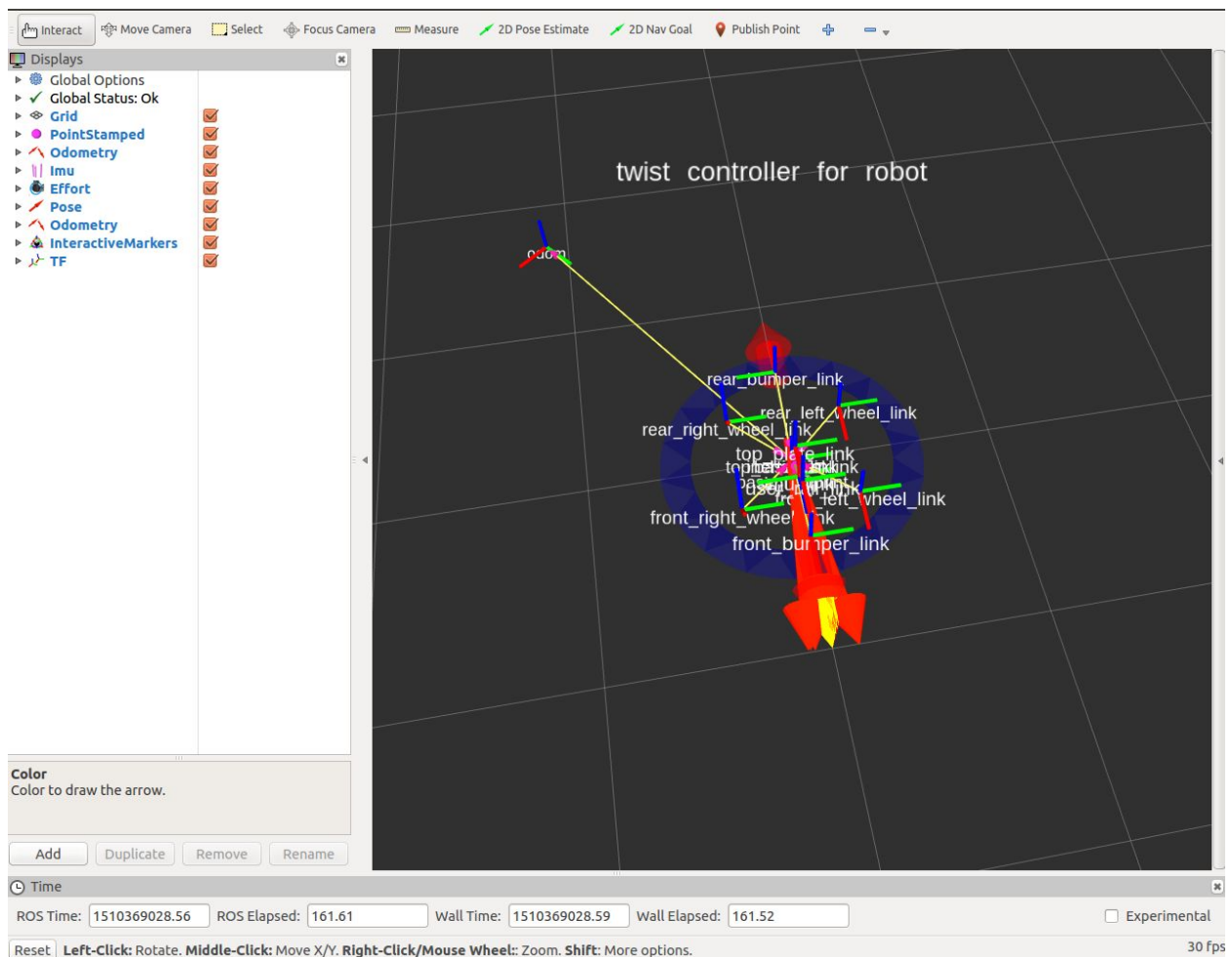


Fig4 - Combined data on Rviz

5. WiFi router's SSID configuration

We(me and Yuchi) created a SSID on the router and enabled dhcp services on the router. We also provided static IP to Husky's mini-pc and remote pc's WiFi ports.

6. WiFi connection's range test

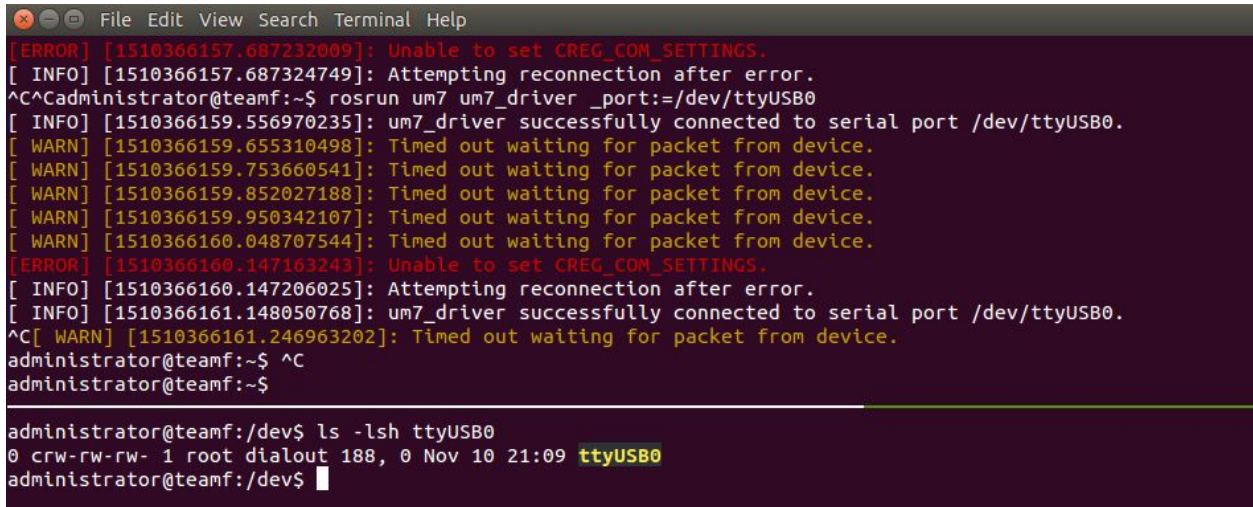
We(Me and Pratibha) tested the range of connection with remote-pc and husky's pc with the setup router. Initial range of the network was calculated to be 27 meters. As the size of the testing arena is specified as 50m x 50m in requirements, we extended the range to 58m by replacing the router's default antennas with the bigger ones we got from MRSD lab. As we plan to place the base station with remote-pc in the middle of the arena, we will have range of 50m around the router.

7. Bebop connection to the setup Wifi network as a client

I worked on this with support from Yuchi. Bebop is configured by default to host it's own network. We accessed the husky operating system using telnet to change the settings so that Bebop can connect as a client to the network hosted by the WiFi Router. After many attempts we couldn't make it work, we are working on the plans to make the Bebop connect to the network.

Challenges

1. The IMU which we connected using ftdi on USB was getting read as serial device in /dev/ttyUSB0. But we were not able to read the data even with sudo permissions. After thorough analysis we realised the permissions of ttyUSB0 was set to some random user and group.



```
File Edit View Search Terminal Help
[ERROR] [1510366157.687232009]: Unable to set CREG_COM_SETTINGS.
[ INFO] [1510366157.687324749]: Attempting reconnection after error.
^C^Cadministrator@teamf:~$ roslaunch um7 um7_driver _port:=/dev/ttyUSB0
[ INFO] [1510366159.556970235]: um7_driver successfully connected to serial port /dev/ttyUSB0.
[ WARN] [1510366159.655310498]: Timed out waiting for packet from device.
[ WARN] [1510366159.753660541]: Timed out waiting for packet from device.
[ WARN] [1510366159.852027188]: Timed out waiting for packet from device.
[ WARN] [1510366159.950342107]: Timed out waiting for packet from device.
[ WARN] [1510366160.048707544]: Timed out waiting for packet from device.
[ERROR] [1510366160.147163243]: Unable to set CREG_COM_SETTINGS.
[ INFO] [1510366160.147206025]: Attempting reconnection after error.
[ INFO] [1510366161.148050768]: um7_driver successfully connected to serial port /dev/ttyUSB0.
^C[ WARN] [1510366161.246963202]: Timed out waiting for packet from device.
administrator@teamf:~$ ^C
administrator@teamf:~$

administrator@teamf:/dev$ ls -lsh ttyUSB0
0 crw-rw-rw- 1 root dialout 188, 0 Nov 10 21:09 ttyUSB0
administrator@teamf:/dev$
```

Fig5 - Not able to read IMU data

So we used chown to change the ownership of the port to the user we were logged in with.
chown administrator:administrator ttyUSB1

```
administrator@teamf:~$  
administrator@teamf:~$ rosrun um7 um7_driver _port:=/dev/ttyUSB1  
[ INFO ] [1510367776.995424666]: um7_driver successfully connected to serial port /dev/ttyUSB1.  
[ INFO ] [1510367777.010771161]: Received packet 02 without data.  
[ INFO ] [1510367777.011686435]: Setting update rate to 20Hz  
[ INFO ] [1510367777.020691518]: Received packet 02 without data.  
[ INFO ] [1510367777.026746400]: Received packet 04 without data.  
[ INFO ] [1510367777.032839363]: Received packet 05 without data.  
[ INFO ] [1510367777.038854517]: Received packet 06 without data.  
[ INFO ] [1510367777.044724087]: Received packet 08 without data.  
[ INFO ] [1510367777.045356575]: Sending command: zero gyroscopes  
[ INFO ] [1510367777.050930065]: Received packet ad without data.
```

Fig6 - Readable IMU data

2. For teleoperation of husky on the network, after doing all the setup we were facing problem in sending the gamepad controller data across the ROS master running on network. After intense debugging we realised that we are required to add the hostname of one system to another to make the system work.
3. Even after trying the methods given on internet we tried to change the network settings of Bebop to make it a client instead of a host. We are discussing the issue with the developer community on the forums. Still waiting for the reply.
4. We tried detection of April tag placed on ground using Bebop's camera feed. We are able to localize the april tags but the location is very unstable. We need to try different filters as suggested by our sponsors.
5. We found it really difficult to test the system outside in the cold weather. Testing outside becomes necessary as we are dependent on the GPS values.

Teamwork

We realised working in pair of two makes both the team members efficient as you always have second opinion about any issue, so you tend to solve it faster. Also, it helps to manage the unavailability of team members as two people have the knowledge about running a particular component or module.

Yuchi worked with me together on WiFi SSID setup, Bebop as a client setup, teleoperation of Husky as explained above. Pratibha worked with me to integrate the IMU and encoders with ROS installed on Husky. Yuchi also worked on April tag implementation using camera feed from Bebop. Danendra majorly worked with Yuchi on testing of Bebop's GPS and it's autonomous navigation. He also worked with pratibha to design the PCB for the system. Rahul worked on integration of Velodyne puck with ROS drivers for obstacle detection. He also worked on setting up the GPS initially with arduino and then with the ROS driver. He also wrote a publisher to convert the GNSS data to GPS coordinates. Pratibha and Rahul worked on fabricating and assembling the mechanical setup as designed in the CAD model previous week.

Future Plans

This week we had separate meetings with all the stakeholders including progress review with the TAs , John and also with our sponsor Katia. We received a lot of feedback to make the FVE a success. We plan to heavily focus on the FVE and incorporate those feedbacks to solve the current issues in this week.

- Connect Bebop 2 with the router network: Pulkit and Pratibha
- April tag location with Bebop: Yuchi and Rahul
- Bebop's GPS based navigation: Danendra and Yuchi