Individual Lab Report #8

Pratik Chatrath

Team A / Team Avengers

Teammates: Tushar Agrawal, Sean Bryan

February 25, 2016

I. Individual Progress

Last PR, I demonstrated successful functioning of the Navigation Stack in simulation. I connected LIDAR to the laptop and used its input in the navigation stack to provide information about obstacles. In simulation, I showed a robot following a search path while avoiding obstacles.

In the past couple of weeks, I took the first steps towards integration of the Navigation Stack into the UAV system. Earlier, we had used visual odometry data and output velocities to show the functioning of navigation stack in simulation. Now, I have been able to establish link between the pixhawk and the navigation stack using the mavros as the medium. I am now able to feed the odometry data of UAV to the Navigation Stack and pass velocity command generated by navigation stack to actual UAV .

I have divided my work from the past few weeks into the below mentioned sections:

1. Linking the Navigation Stack with the Pixhawk

2. Testing the code functionality on Software In The Loop Simulation of the Pixhawk

3. Testing the code functionality on X8+

1. Linking the Navigation Stack with the Pixhawk

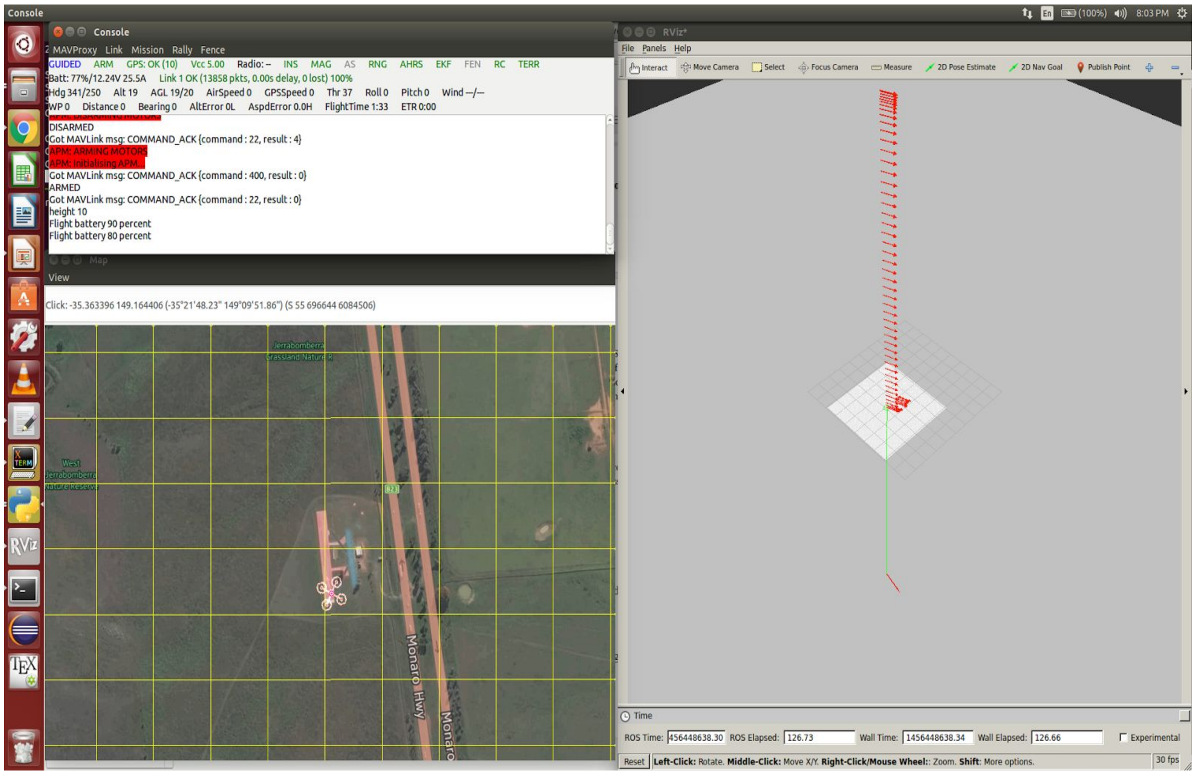
To link the Navigation Stack with the Pixhawk, I dug deep into topics subscribed and published by mavros and navigation stack. After making the ros topic formats between mavros and navigation stack compatible, I went on to implement the following steps:

1. Pass the odometry data received from mavros to navigation stack
2. Changed transform tree in mavros code to make it compatible with my previously written transform tree configuration
3. Relayed the navigation stack /cmd_vel output to mavros topic : /mavros/setpoint_velocity/cmd_vel

In order to get these steps right, I had to go through different versions of mavros to figure out which one provided the odometry data in the format navigation stack accepts it. Finally, I found the version which provided the odometry data in the msg format that is compatible with what navigation stack can accept.

2. Testing code functionality on Software In The Loop Simulation (SITL) of the Pixhawk

To test the functioning of my code I used the SITL. We have found that SITL provides a very good approximation of the actual UAV. We reached this conclusion because the behaviour of the UAV in simulation and on actual X8+ UAV platform is very close. Hence, SITL was a perfect place to test my code. This allowed me to test my code sitting inside a room without having to go to a place where GPS strength is strong enough to fly the actual UAV.


Console
MAVProxy Link Mission Rally Fence
Map
View
RViz*
File Panels Help
Interact Move Camera Select Focus Camera Measure 2D Pose Estimate 2D Nav Goal Publish Point
Jerrabomberra Grassland Nature R
West Jerrabomberra Nature Reserve
Monaro Hwy
Monaro
Time
Reset
30 fps
8:03 PM

```
GUIDED   ARM   GPS: OK (10)   Vcc 5.00   Radio: --   INS   MAG   AS   RNG   AHRS   EKF   FEN   RC   TERR
Batt: 77%/12.24V 25.5A   Link 1 OK (13858 pkts, 0.00s delay, 0 lost) 100%
Hdg 341/250   Alt 19   AGL 19/20   AirSpeed 0   GPSSpeed 0   Thr 37   Roll 0   Pitch 0   Wind ---/---
WP 0   Distance 0   Bearing 0   AltError 0L   AspdError 0.0H   FlightTime 1:33   ETR 0:00
DISARMED
Got MAVLink msg: COMMAND_ACK {command : 22, result : 4}
APM: ARMING MOTORS
APM: Initialising APM...
Got MAVLink msg: COMMAND_ACK {command : 400, result : 0}
ARMED
Got MAVLink msg: COMMAND_ACK {command : 22, result : 0}
height 10
Flight battery 90 percent
Flight battery 80 percent
```

```
Click: -35.363396 149.164406 (-35°21'48.23" 149°09'51.86") (S 55 696644 6084506)
```

ROS Time: 456448638.30 ROS Elapsed: 126.73 Wall Time: 1456448638.34 Wall Elapsed: 126.66 ☐ Experimental

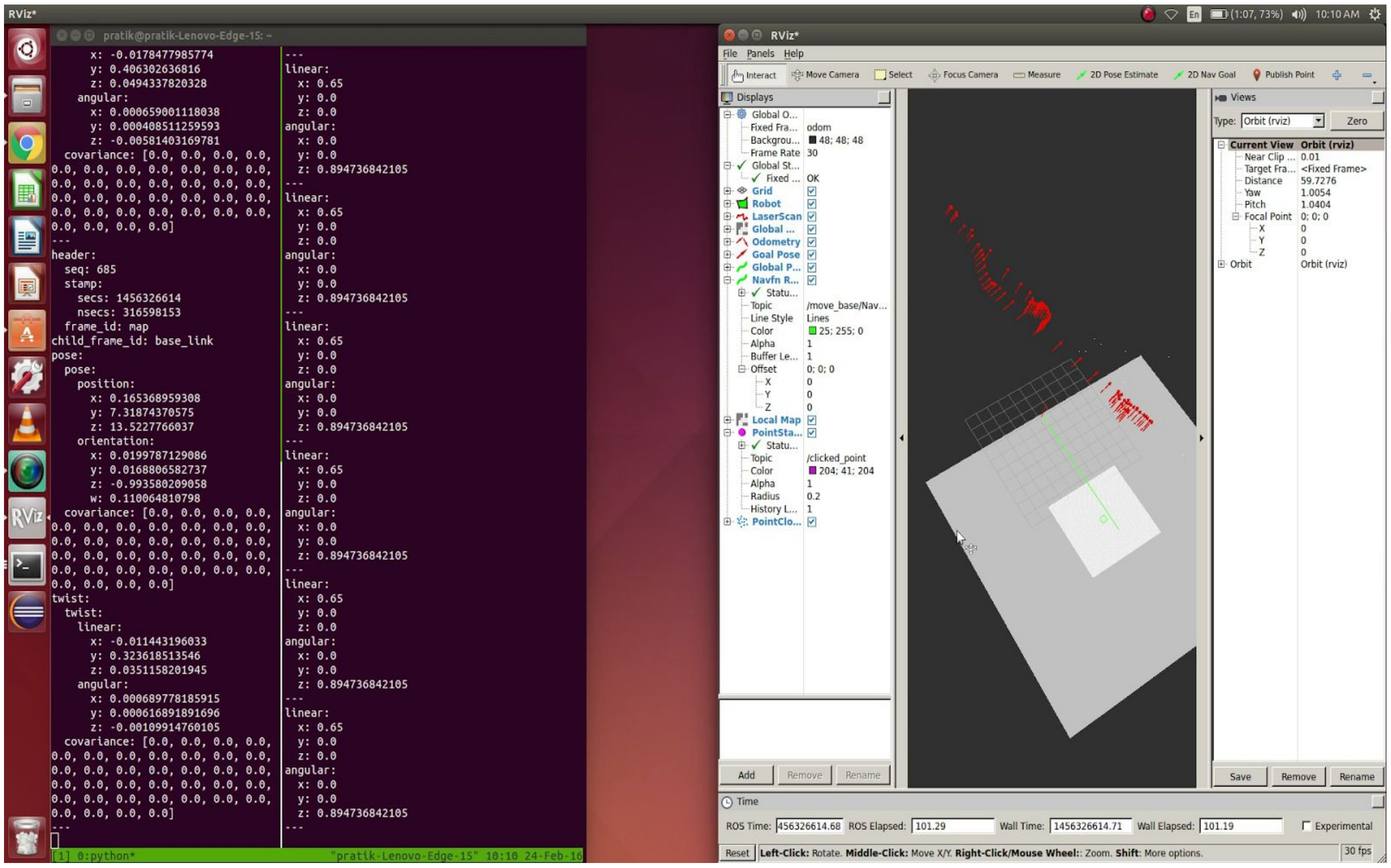
Left-Click: Rotate. **Middle-Click:** Move X/Y. **Right-Click/Mouse Wheel:**: Zoom. **Shift**: More options.

In the above figure, the tab on the upper left is the console which shows the data about the status of the UAV (its altitude etc.). The bottom left figure is the SITL map showing the virtual UAV in a virtual map. On the right is the rviz visualization of the UAV. The red arrow are the location of UAV. As seen in the figure the continuous vertical red line shows the UAVs takeoff. The green line is the path generated by navigation stack to reach the goal position (shown by a single sed line on the surface).

Once I got rid of all the compatibility errors, I could see the odometry data being received by the navigations stack and velocity commands being passed to mavros. This simulation allowed me to see the UAV actually trying to follow the generated velocities.

3. Testing code functionality on X8+

After simulating the process successfully, I tested the code on the X8+ platform. I passed a goal position to the UAV in the rviz and moved the UAV manually towards the goal position. I observed the changes in odometry data received from the UAV and the velocity output of the navigation stack.



```
      x: -0.0178477985774
      y: 0.406302636816
      z: 0.0494337820328
    angular:
      x: 0.000659001118038
      y: 0.000408511259593
      z: -0.00581403169781
  covariance: [0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0]
---
header:
  seq: 685
  stamp:
    secs: 1456326614
    nsecs: 316598153
  frame_id: map
child_frame_id: base_link
pose:
  pose:
    position:
      x: 0.165368959308
      y: 7.31874370575
      z: 13.5227766037
    orientation:
      x: 0.0199787129086
      y: 0.0168806582737
      z: -0.993580209058
      w: 0.110064810798
  covariance: [0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0]
twist:
  twist:
    linear:
      x: -0.011443196033
      y: 0.323618513546
      z: 0.0351158201945
    angular:
      x: 0.000689778185915
      y: 0.000616891891696
      z: -0.00109914760105
  covariance: [0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0]
---
```

```
---
linear:
  x: 0.65
  y: 0.0
  z: 0.0
angular:
  x: 0.0
  y: 0.0
  z: 0.894736842105
---
linear:
  x: 0.65
  y: 0.0
  z: 0.0
angular:
  x: 0.0
  y: 0.0
  z: 0.894736842105
---
linear:
  x: 0.65
  y: 0.0
  z: 0.0
angular:
  x: 0.0
  y: 0.0
  z: 0.894736842105
---
linear:
  x: 0.65
  y: 0.0
  z: 0.0
angular:
  x: 0.0
  y: 0.0
  z: 0.894736842105
---
linear:
  x: 0.65
  y: 0.0
  z: 0.0
angular:
  x: 0.0
  y: 0.0
  z: 0.894736842105
---
linear:
  x: 0.65
  y: 0.0
  z: 0.0
angular:
  x: 0.0
  y: 0.0
  z: 0.894736842105
---
```

```
[1] 0:python*                    "pratik-Lenovo-Edge-15" 10:10 24-Feb-16
```

Global O...	
Fixed Fra...	odom
Backgrou...	■ 48; 48; 48
Frame Rate	30
✓ Global St...	
✓ Fixed ...	OK
Grid	✓
Robot	✓
LaserScan	✓
Global ...	✓
Odometry	✓
Goal Pose	✓
Global P...	✓
Navfn R...	✓
✓ Statu...	
Topic	/move_base/Nav...
Line Style	Lines
Color	🟩 25; 255; 0
Alpha	1
Buffer Le...	1
Offset	0; 0; 0
X	0
Y	0
Z	0
Local Map	✓
PointSta...	✓
✓ Statu...	
Topic	/clicked_point
Color	🟪 204; 41; 204
Alpha	1
Radius	0.2
History L...	1
PointClo...	✓

Type: Orbit (rviz) Zero

Current View	Orbit (rviz)
Near Clip ...	0.01
Target Fra...	<Fixed Frame>
Distance	59.7276
Yaw	1.0054
Pitch	1.0404
Focal Point	0; 0; 0
X	0
Y	0
Z	0
Orbit	Orbit (rviz)

ROS Time: 1456326614.68 ROS Elapsed: 101.29 Wall Time: 1456326614.71 Wall Elapsed: 101.19 Experimental

In the above figure to the extreme left is the odometry data received from the Pixhawk. We can see the position, orientation & linear and angular velocities being published. The second data column from the left is the navigation stack velocity output in form of linear and angular velocity.

The figure on the right is the Rviz visualization of the UAV. The red lines are the positions of the UAV. The green line on the surface is the path planned by the navigation stack in 2D which the UAV tries to follow.

The output of odometry and navigation stack changed as I moved the UAV around manually. This verifies that the Navigation Stack and the Pixhawk are communicating as needed.

Challenges

I faced the following challenges in achieving the above mentioned progress :

1. As I am currently working on the integration of navigation stack with the Pixhawk, the primary challenge is getting both the entities to talk in the same language, essentially the ROS topic format compatibility. As discussed above I solved this issue by updating the mavros version. The new version of mavros has published the odom topic which navigation stack can use. In addition to this, making few other changes in mavros and tf tree also helped.
2. As of now, I have the communication between the Navigation Stack and the Pixhawk setup. However, I am facing difficulty in actually getting the UAV to follow the path autonomously. Diving deeper into the problem, I figured out that the UAV is not able to follow the angular z rotation suggested by the Navigation Stack. This is an issue with the mavros. I am in touch with the person who has written the open source mavros library and the ros community on the ros forum. I have tried a few things that they suggested, however, the issue remains unsolved. Supposedly, the angular z movement bug is resolved in the px4 version of the firmware. Hence, we might have to change our firmware from apm to px4. Our team will be working on this in the coming week.

II. Teamwork

Tushar got the UAV to follow the marker autonomously. As a result of this, the UAV can track the marker even if we manually move the marker around. This completes the integration of marker detection code with flight controller. In addition to this, he also helped me with few of the bugs that I faced during the initial phase of establishing connectivity.

Sean got the Nica drone (the electro-permanent-magnet) control through the Mission Planner working. Now we can click buttons on the GUI interface of Mission Planner to enable and disable the Nica Drone.

III. Future Work

In next PR, I will demonstrate autonomous point A to B navigation of the UAV with obstacle avoidance on the physical X8+ platform. For this first, I will have to fix the angular z movement issue. After resolving that, I will test and tune the parameters of navigation stack so that the UAV can see the obstacles and try to avoid it.