Individual Lab Report 2

Progress Review 1

Pulkit Goyal

October 20, 2017

Team F - Falcon Eye
Pulkit Goyal
Pratibha Tripathi
Yuchi Wang
Rahul Ramkrishnan
Danendra singh

1 Individual Progress

Last week, I primarily contributed in setting up the Zotak mini-PC for Husky and make the Husky run with Game Controller. I was also in contact with Clearpath Robotics representative for debugging of the issue.

1.1 Husky Robot

Husky is a medium sized robotic development platform. Its large payload capacity and power systems accommodate an extensive variety of payloads, customized to meet research needs. Husky is fully supported in ROS with community driven Open Source code and examples. We borrowed a Husky from George Kantor for MRSD project. We connected the Husky system with the Mini-PC using RS232 cable for the communication.

1.2 Zotak-Mini PC Setup

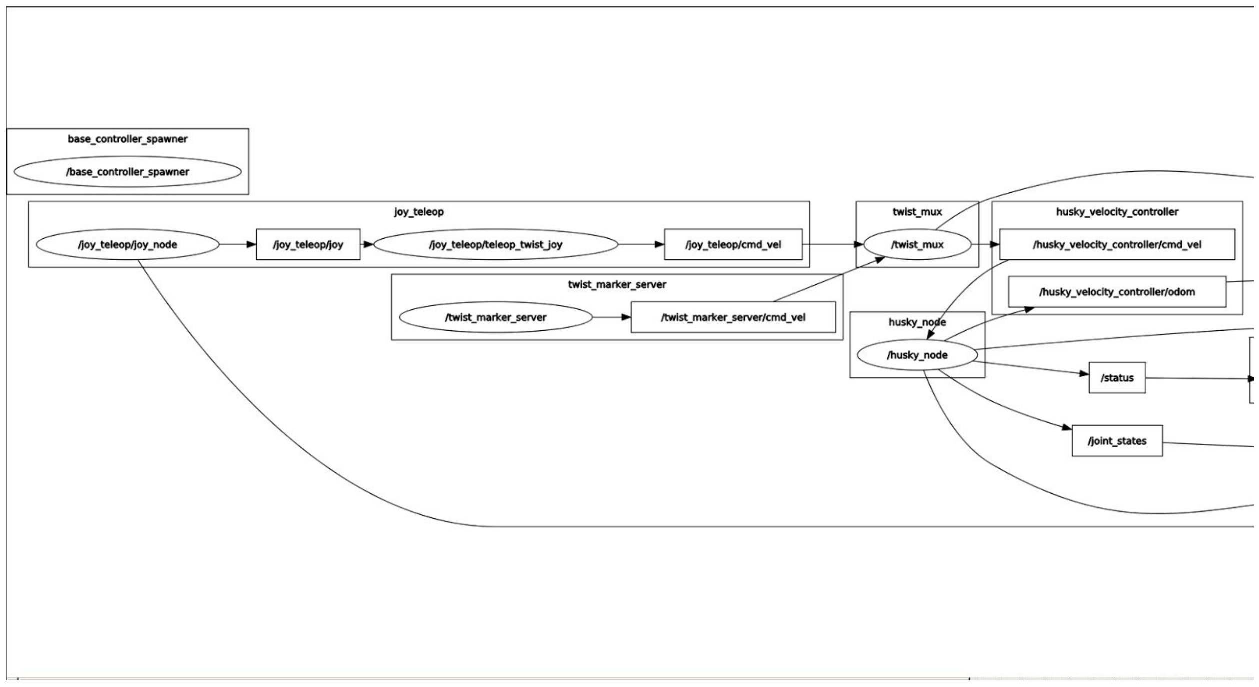
Clearpath has a clone of basic ubuntu operating system which comes with all the husky packages installed on it. I installed the clone on Mini-PC using Clone Zilla. After cloning the basic image onto the Husky robot, Clone Zilla is supposed to download the full image from the internet during installation. The mini-pc couldn't take IP from the DHCP, so I had to configure static IP for the mini-PC to be able to connect to the network for the setup. The standard image provided by clearpath does not have a GUI, as team was having difficulty in accessing the mini-pc using terminal, I installed a lighter GUI, lxde, on the mini-pc.

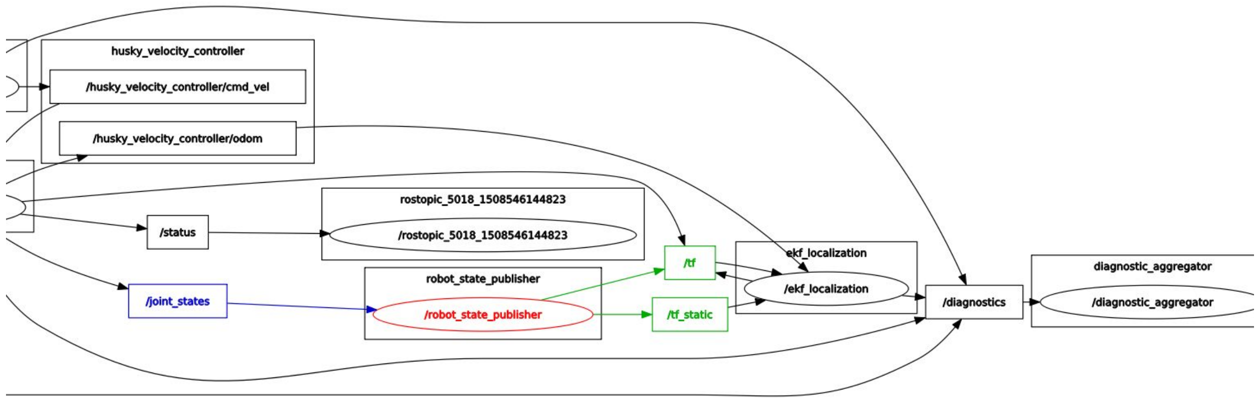
1.3 Communication between Mini-pc and Husky Robot

When the mini-pc was fully setup we were having difficulty in communicating with the husky using RS232 cable. I found that there was a fault with the RS232 to USB cable. I verified this by checking the data exchange and reading the data received from husky on terminal.

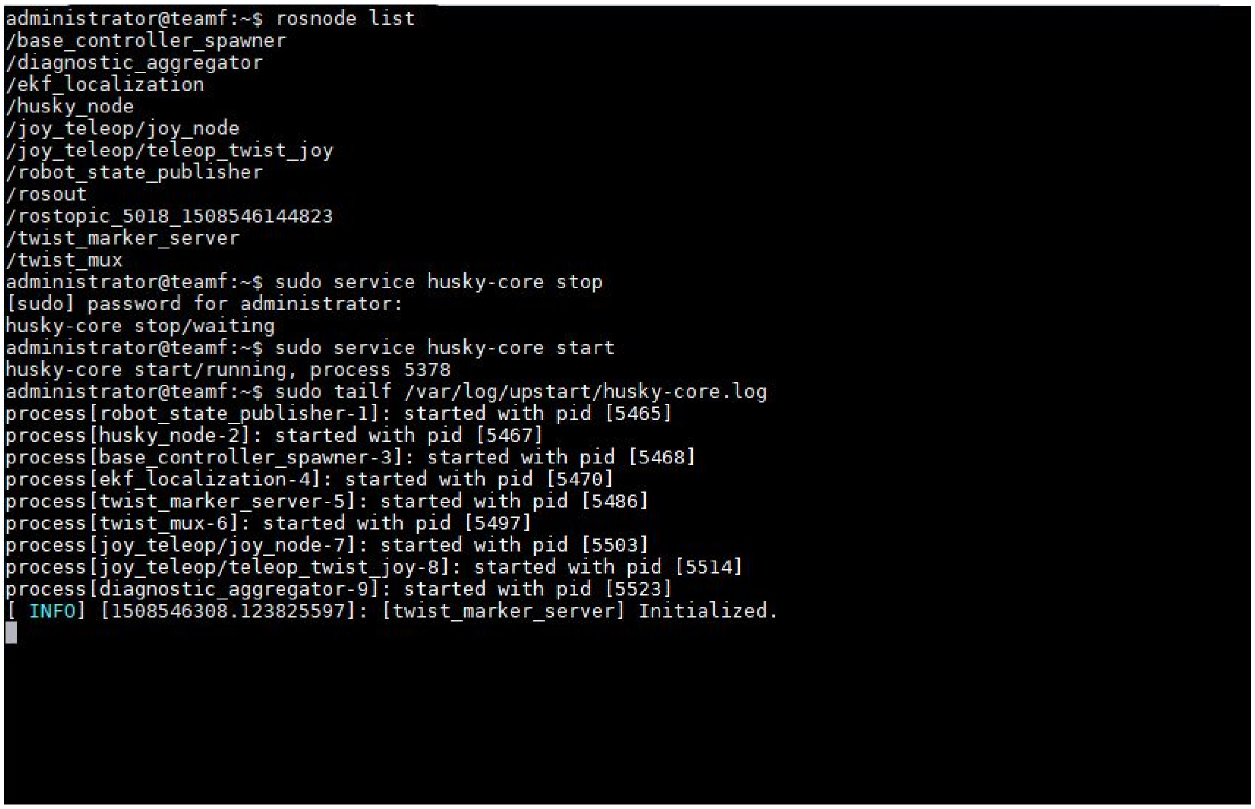
1.4 ROS for Husky

ROS has several nodes such as diagnostics, husky_node, teleop and many more which are essential for robot operation. I went into detail about all the available support from ROS.





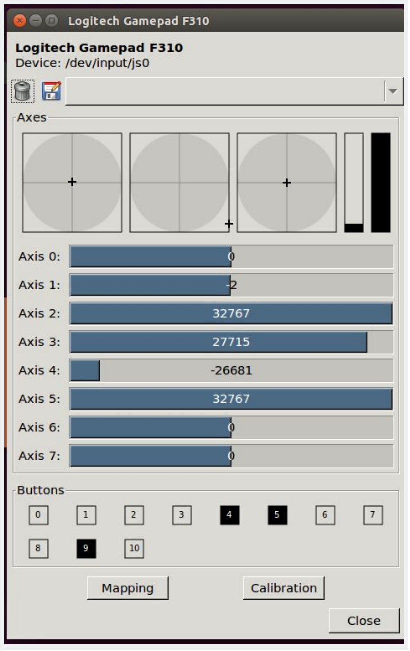
I read about the upstart services, as the image obtained from clearpath has an upstart service which fires up all the required ROS launch file when the mini-pc is starts. I also located the file where the logs are generated for this service.



```
administrator@teamf:~$ rosnode list
/base_controller_spawner
/diagnostic_aggregator
/ekf_localization
/husky_node
/joy_teleop/joy_node
/joy_teleop/teleop_twist_joy
/robot_state_publisher
/rosout
/rostopic_5018_1508546144823
/twist_marker_server
/twist_mux
administrator@teamf:~$ sudo service husky-core stop
[sudo] password for administrator:
husky-core stop/waiting
administrator@teamf:~$ sudo service husky-core start
husky-core start/running, process 5378
administrator@teamf:~$ sudo tailf /var/log/upstart/husky-core.log
process[robot_state_publisher-1]: started with pid [5465]
process[husky_node-2]: started with pid [5467]
process[base_controller_spawner-3]: started with pid [5468]
process[ekf_localization-4]: started with pid [5470]
process[twist_marker_server-5]: started with pid [5486]
process[twist_mux-6]: started with pid [5497]
process[joy_teleop/joy_node-7]: started with pid [5503]
process[joy_teleop/teleop_twist_joy-8]: started with pid [5514]
process[diagnostic_aggregator-9]: started with pid [5523]
[ INFO] [1508546308.123825597]: [twist_marker_server] Initialized.
```

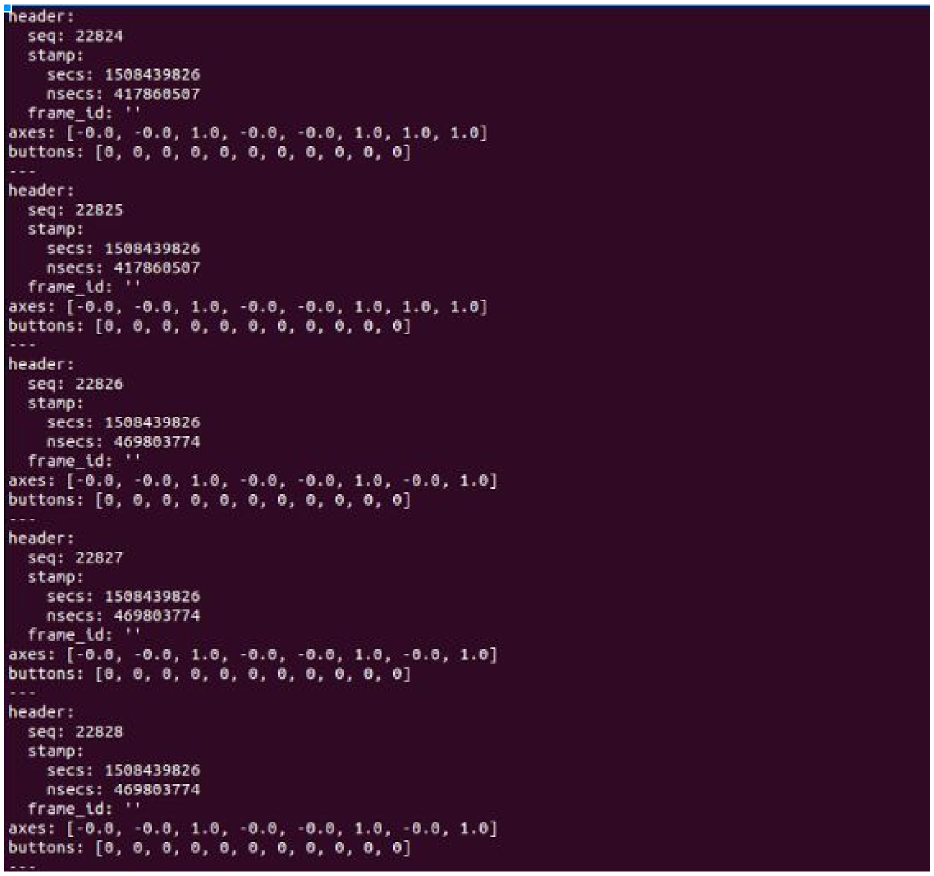
1.5 Debugging the Husky stop state

After the basic ROS setup. I tried reading the Gamepad controller in the Mini-PC. I tried testing the gamepad input with `jstest-gui.`



```
Logitech Gamepad F310
Device: /dev/input/js0
Axes
Axis 0: 0
Axis 1: -2
Axis 2: 32767
Axis 3: 27715
Axis 4: -26681
Axis 5: 32767
Axis 6: 0
Axis 7: 0
```

After that I echoed the /joy_teleop/joy topic which is supposed to publish the values of gamepad to the husky_node for movement of Husky. The values were being read by the controller node, but even then there was no movement in the husky. After echoing /husky_velocity_control/cmd_vel, I concluded that gamepad inputs are not getting converted to husky velocity.



```
header:
  seq: 22824
  stamp:
    secs: 1508439826
    nsecs: 417860507
  frame_id: ''
axes: [-0.0, -0.0, 1.0, -0.0, -0.0, 1.0, 1.0, 1.0]
buttons: [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
---
header:
  seq: 22825
  stamp:
    secs: 1508439826
    nsecs: 417860507
  frame_id: ''
axes: [-0.0, -0.0, 1.0, -0.0, -0.0, 1.0, 1.0, 1.0]
buttons: [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
---
header:
  seq: 22826
  stamp:
    secs: 1508439826
    nsecs: 469803774
  frame_id: ''
axes: [-0.0, -0.0, 1.0, -0.0, -0.0, 1.0, -0.0, 1.0]
buttons: [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
---
header:
  seq: 22827
  stamp:
    secs: 1508439826
    nsecs: 469803774
  frame_id: ''
axes: [-0.0, -0.0, 1.0, -0.0, -0.0, 1.0, -0.0, 1.0]
buttons: [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
---
header:
  seq: 22828
  stamp:
    secs: 1508439826
    nsecs: 469803774
  frame_id: ''
axes: [-0.0, -0.0, 1.0, -0.0, -0.0, 1.0, -0.0, 1.0]
buttons: [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
---
```

I echoed the diagnostics and status topic to check the status of the robot. The status topic gives the voltage and current values for battery, motor drivers. After careful review, and comparison of the values with the ones specified by clearpath for Husky. We concluded that motor drivers aren't getting voltage from the battery, because the status topic showed 24v for the battery but 0v for left and right motor driver.

Hence, we opened the husky and verified that the fuses were intact. We tried checking the motor drivers but the wires and casing was very neatly routed and we decided to not open that up.

We borrowed another working Husky from NREC with help of Prof. Dimi. It didn’t take us much time to run the Husky from NREC with help of setup we did for George’s Husky. We used the same mini-PC and other accessories with husky from NREC. Using NREC’s Husky as reference we debugged issues with Kantor’s Husky and got both the Husky’s working.

2 Challenges

We faced many challenges this week both technical and communication with our sponsor. To start with we faced many issues in getting the Husky to run. We were also struggling from very long about fitting our own interest with interest of our sponsor. We went through many iterations of the use case with various stakeholders. We talked to many experienced people about the scope of project that can be handled within an year and also about the kind of experience that will be required in the team for the execution of the project. After carefully considering all the factors and incorporating the suggestions from our sponsor we rescoped our project. Our new case is as follow: We developed a system with UAV and UGV where UAV will survey the area, and send the traversable waypoints to UGV. UAV will help UGV to make informed path planning decisions for unknown environment.

3 Teamwork

Me, Pratibha and Rahul were mainly focusing on making the Husky work.
The work for UAV, Parrot Beebop 2, was handled by Yuchi and Danendra. Danendra interfaced the drone using ROS and Yuchi interfaced the drone using the SDK provided by parrot. Both were successful in controlling the drone. We finalized on ROS as it has quite a lot of support.

4 Future Plans

For next week I will focus on developing a high level control stack for the Husky with encoders. Likewise, Yuchi and Danendra will on high level control of Beebop 2 drone. We are currently able to give low level commands like move forward, backwards, right or left to both UAV and AGV.
Rahul and Pratibha will focus on finalizing the GPS hardware to be used for Husky and integrate that with ROS. They will also be evaluating the accuracy of GPS sensors available in inventory.