# Lab03 – udev Rules and Testing

## ELEC ENG 3EY4

Brent Menheere - menheerb – 400362843

Aum Shah – shaha124 – 400388075

L04

February 8, 2024

***Please Note; Due to an error with NoMachine, we were not able to remotely access the car, and in consequence all screenshots have been replaced with photos of the lab monitor. We were able to resolve this issue at the end of the lab with the help of our TAs by updating NoMachine on our laptop and our car and restarting them both. We apologize if the pictures we took are not the clearest.***


**Question 1:**
The command we used to navigate to the rules, is `cd /etc/udev/rules.d.` We used these commands to navigate to the specific folder required to add the new files.

**Question 2:**
Using our Linux knowledge from Lab 1 we added the rule file for the lidar by using the command `sudo gedit rplidar.rules` . We used this to create the rplidar rule file, to put the rules in.

**Question 3:**

We used the command `sudo gedit vesc.rules` to create the vesc rules file, to put the rules in.

**Question 4:**

We used the command `sudo gedit joypad-f710.rules` to create the controller rule file, to put the rules in.

**Question 5:**

We used the command `sudo gedit 99-realsense-libusb.rules` to create the camera rule file, to put the rules in.

**Question 6:**

The `control --reload-rules` command reloads the udev rules configuration. The `udevadm trigger` command prompts udev to reapply the rules and make the system aware of any changes made to the ruleset. This makes sure that we can update the rules without interrupting current operations that are going on.

**Question 7:**

We need to edit the ROS launch files, "rplidar.launch" file, because we need to configure the lidar sensor's parameters, ensuring proper communication. We add that line so that the serial port knows where to look for when it needs to initialize the group of nodes specified for rplidar.

**Question 8:**

When executing the command `rostopic echo scan/ranges[360],` we observed data points collected from the back of the lidar. When a laptop was put in front of the lidar at around 20 cm, the output in Figure 1 was displayed. The values shown represent the distance between the lidar and the nearest object in meters, thus the output was around 0.2.

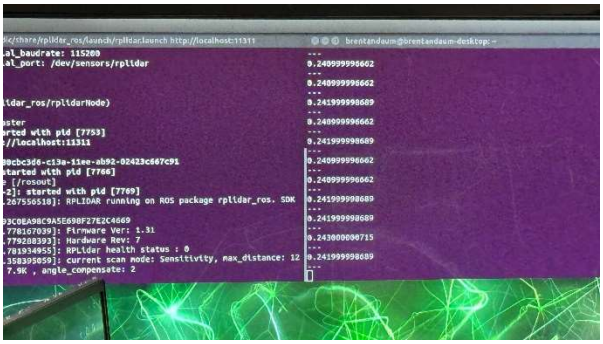



*Figure 1 Output of lidar at 20 cm*

We then moved the laptop back to about a 100 cm from the lidar. The output in Figure 2 shows that the reading changed to around 1.0, representing 1 meter distance between the object and the lidar.

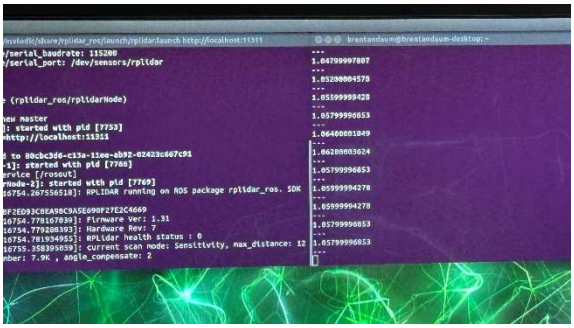



*Figure 2 Output of lidar at 1 m*

**Question 9:**

When viewing the output of the lidar system with `rviz`, we achieved the image shown in Figure 3. The point cloud represents objects seen by the lidar on a 2D plane. We observed that each of the red objects shown in the point cloud represented objects and people within the lab room. Near the top left and top right of the point cloud, straight lines can be seen, these lines show the walls of the lab room. At the centre of the grid, a collection of points can be seen which show our group along with objects close to us

(wall, monitor, laptops, etc.). We notices that when either group member moved our arms or bodies, the action was reflected in the point cloud.

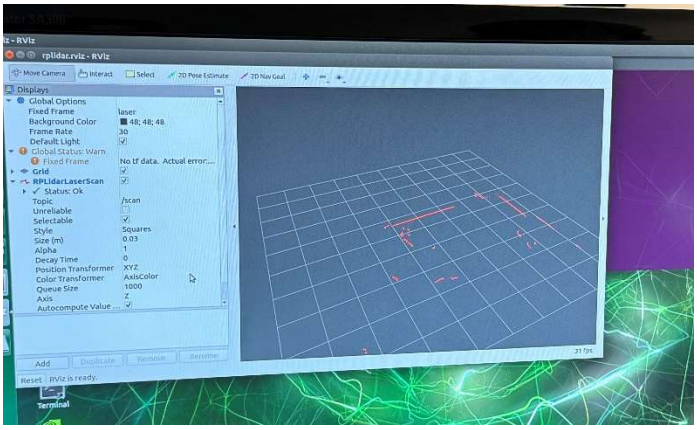



*Figure 3 Point cloud map of lidar*

**Question 10:**

Shown below are the screenshots of both `image_raw` and `image_rect_raw`, respectivly:

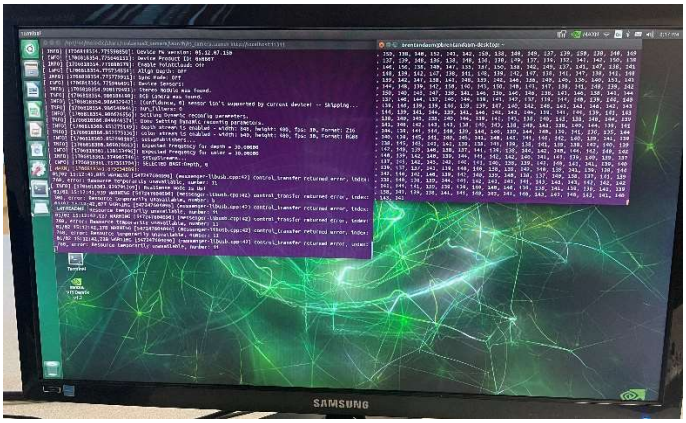



*Figure 4 image_raw*

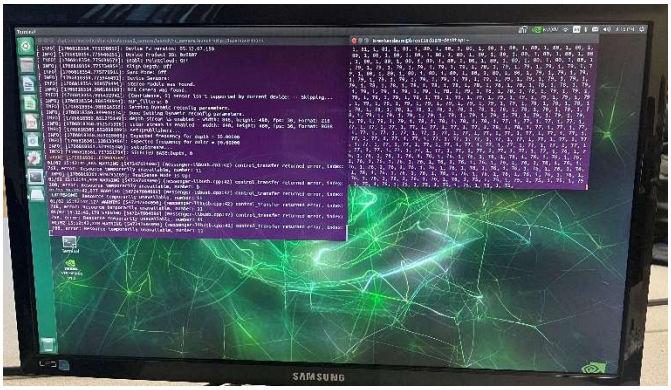



*Figure 5 image_rect_raw*

**Question 11:**

Shown in Figure 6 is a photo of our group in both a colored image and depth, simultaneously. The depth camera image reflects the colored photo image. The depth camera represents the raw image by showing the distance in different shades of darkness. The closer an object is the darker the image will surround that object, and the further it is the lighter it will show that object.

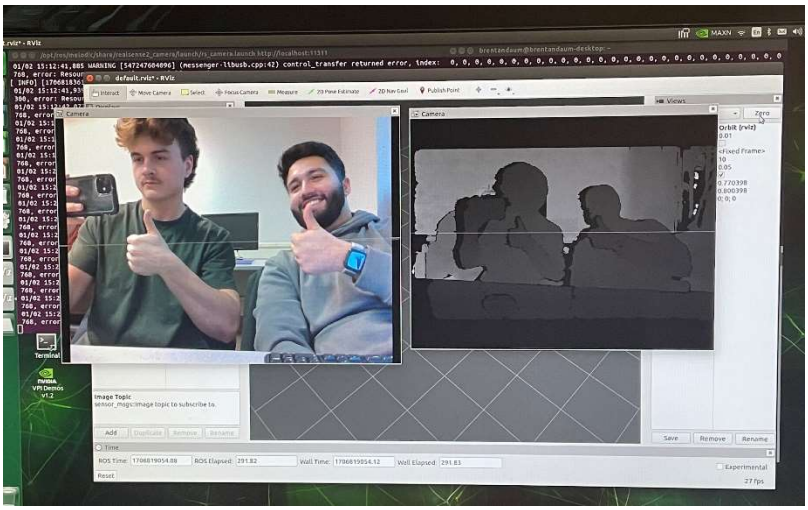


*Figure 6 Camera and depth camera*

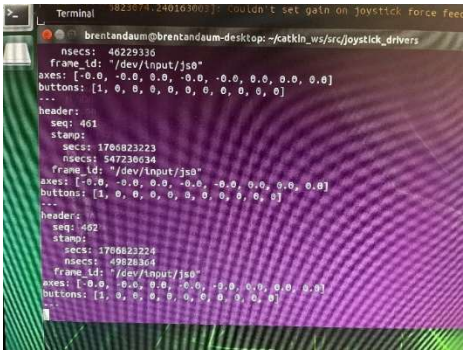
**Question 12:**

We added these files to help configure the controller. The command lines we added in the launch file `<node name = "joy_node" pkg="joy" type="joy_node">` to configure the `joy_node`. After that we used `<param name = "dev" value "= /dev/input/js0"/>` to use the joystick at `/dev/input/js0`. Then we used `<param name = "autorepeat_rate" value = "2" />` to set the autorepeat rate to 2 Hz for transmitting messages when there is no change in the joystick state. Finally, we added `<param name = 'coalesce interval" value = "0.05" />` to adjust the coalesce interval to control the rate at which messages are published.

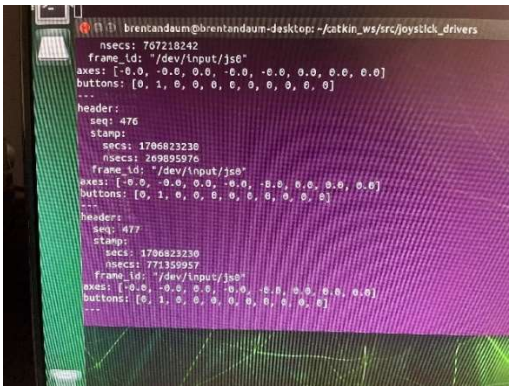
**Question 13:**

The series of figures below show the two arrays, axes and buttons, after each button is pressed.

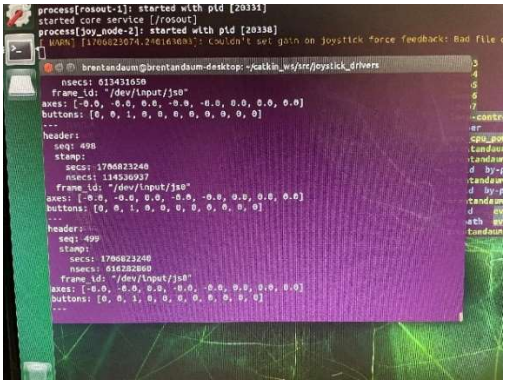
Button A:



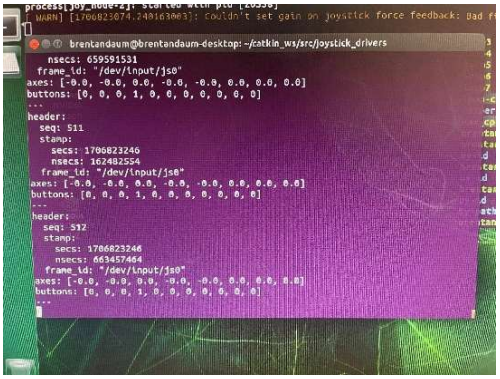
Button B:



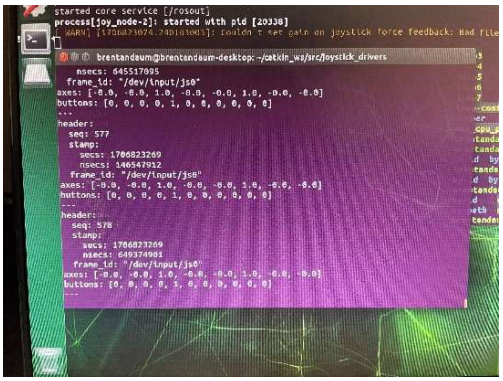

Button X:



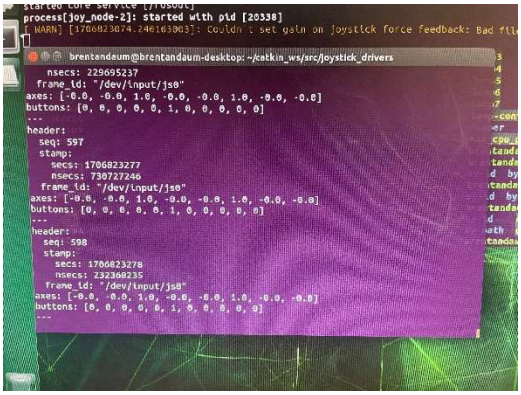

Button Y:



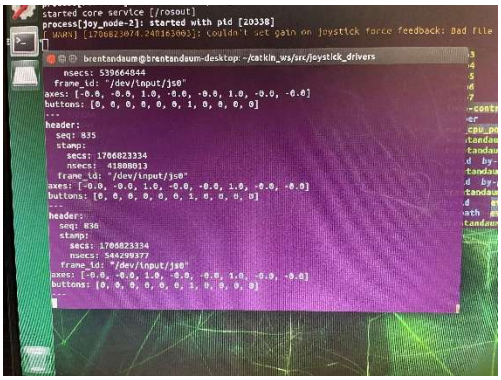

LB:



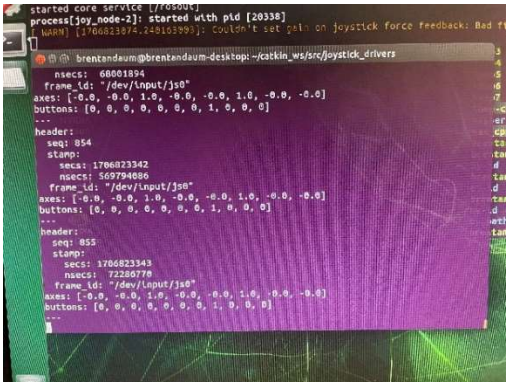

RB:



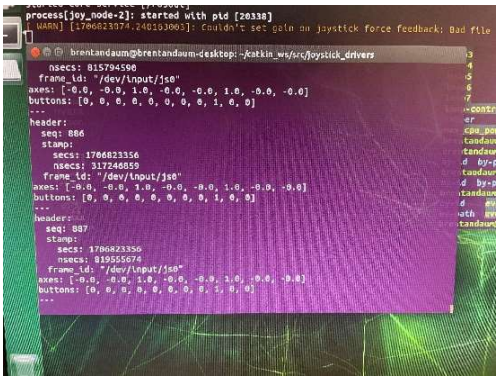

Button Back:



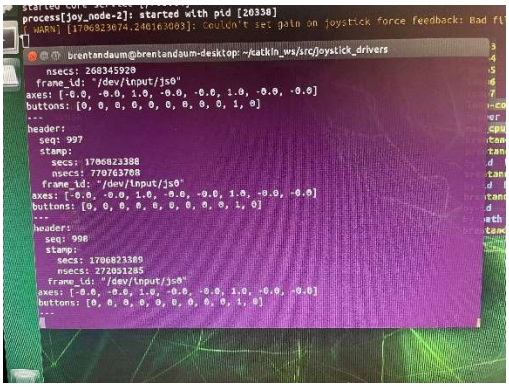

Button Start:



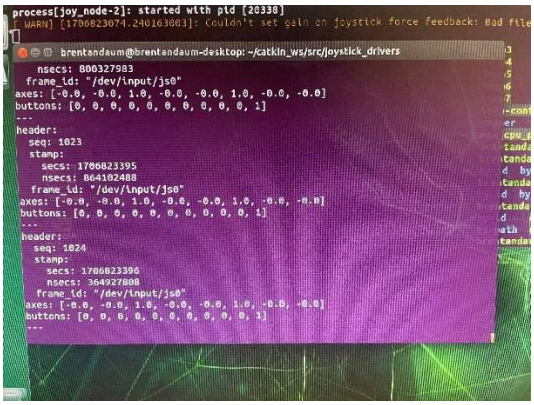

Logitech Button:



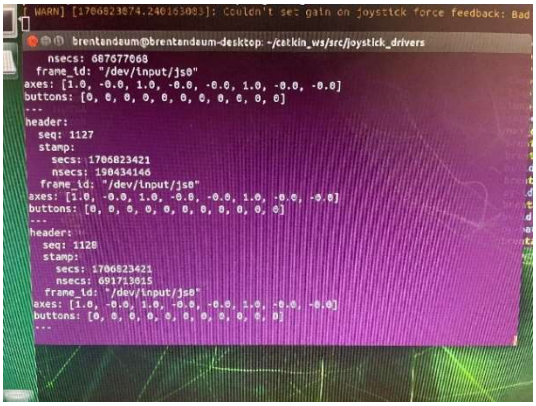

L3:



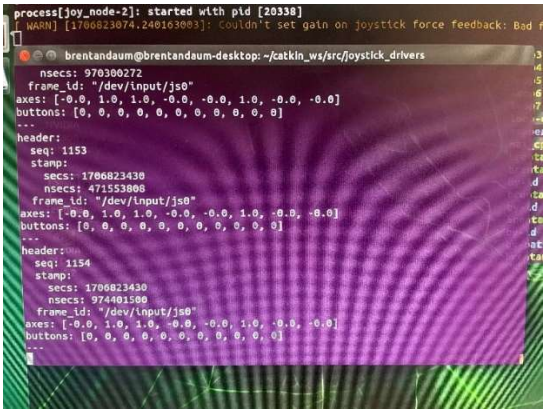

R3:



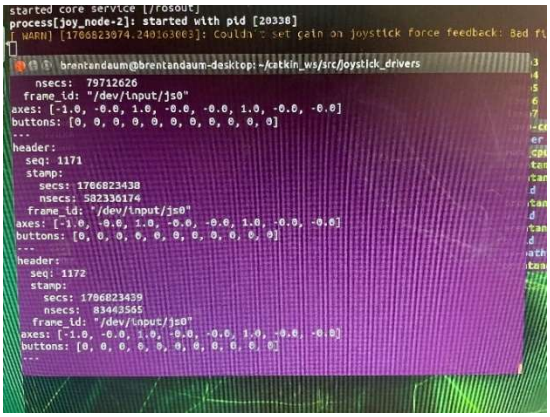

Left Stick Left:



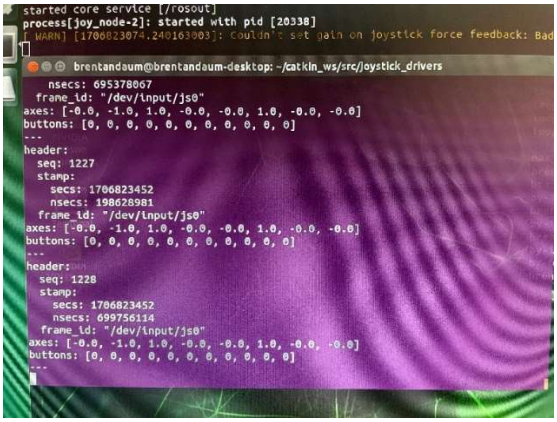

Left Stick Up:



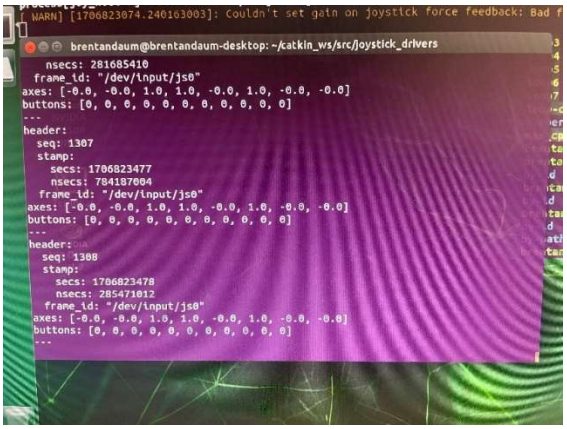

Left Stick Right:



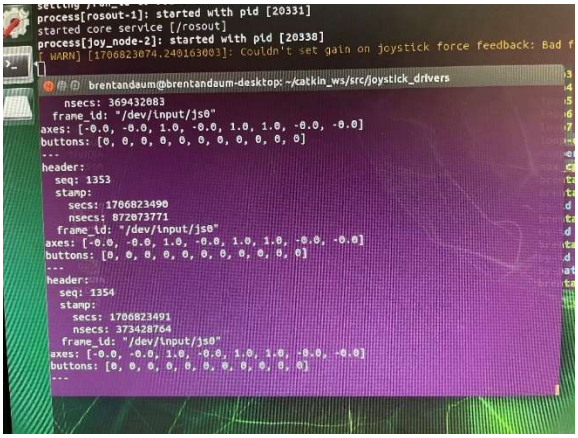

Left Stick Down:



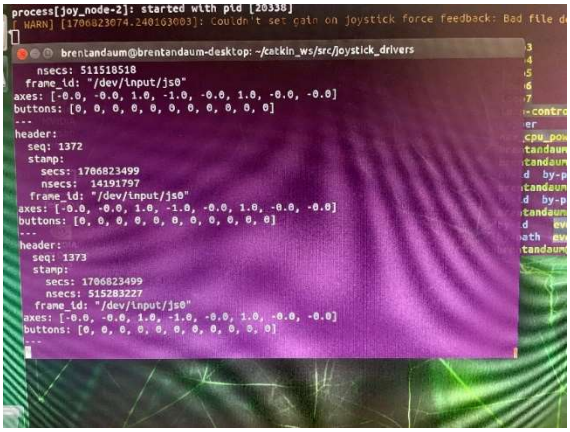

Right Stick Left:



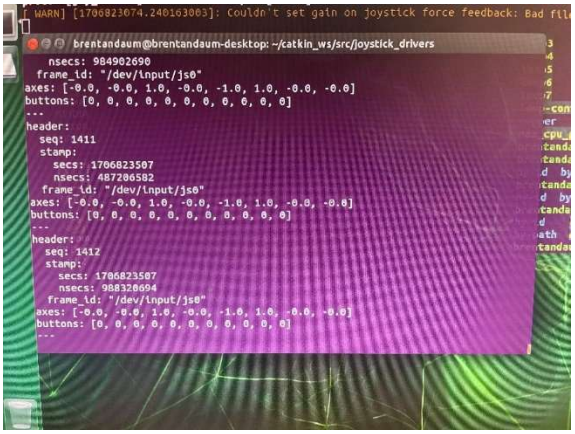

Right Stick Up:



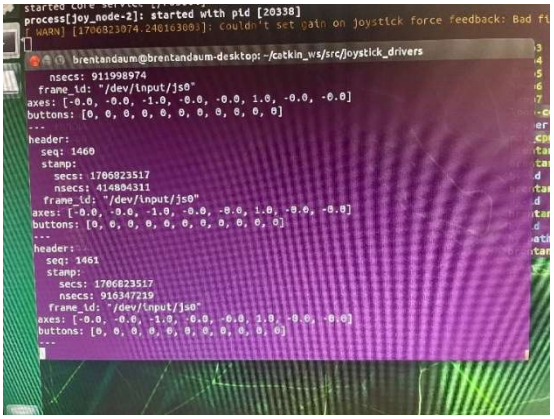

Right Stick Right:



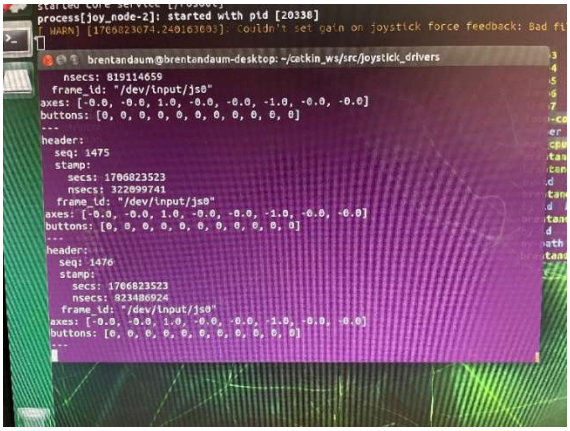

Right Stick Down:



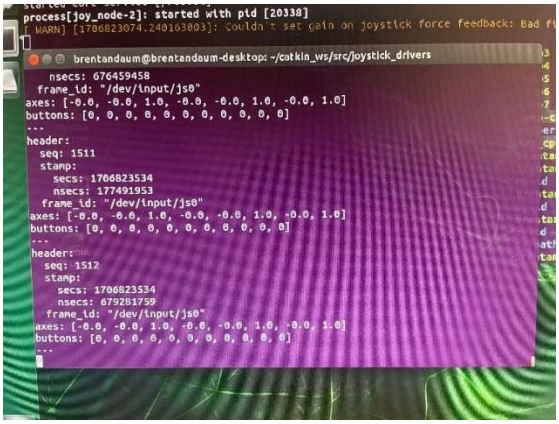

LT:



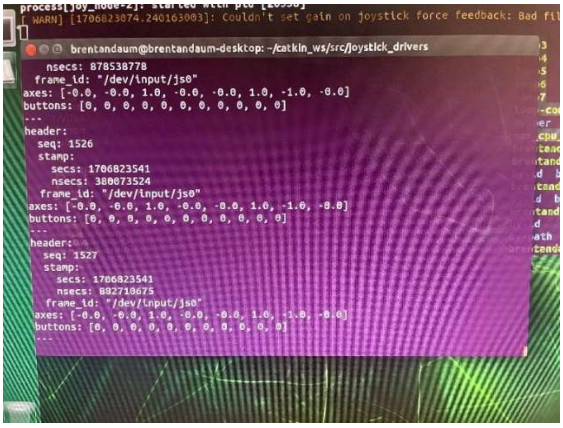

RT:



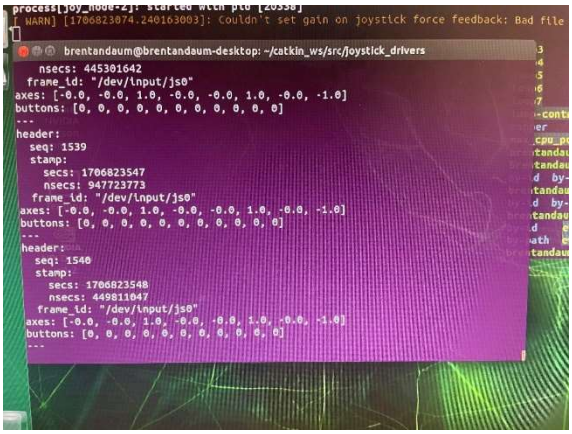

D-Pad Up:



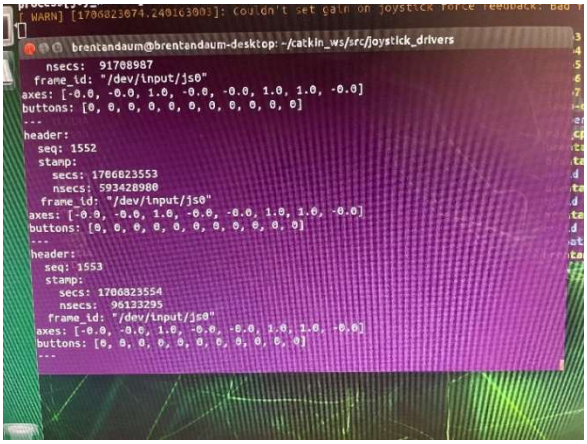

D-Pad Right:




D-Pad Down:




D-Pad Left:




The following table describes the axes array behavior as different controls are manipulated. Most controls are represented as a 0 unless manipulated, however each trigger (RT and LT) is represented as a 1.0 when not pressed.

| Index | Control | Behaviour |
|---|---|---|
| 0 | Left Stick (x-axis) | Left = 1.0, Right = -1.0 |
| 1 | Left Stick (y-axis) | Up = 1.0, Down = -1.0 |
| 2 | LT | Unpressed = 1.0, Pressed = -1.0 |
| 3 | RT | Unpressed = 1.0, Pressed = -1.0 |
| 4 | Right Stick (x-axis) | Left = 1.0, Right = -1.0 |
| 5 | Right Stick (y-axis) | Up = 1.0, Down = -1.0 |
| 6 | D-Pad (x-axis) | Left = 1.0, Right = -1.0 |
| 7 | D-Pad (y-axis) | Up = 1.0, Down = -1.0 |

The table below describes the buttons array behavior as different buttons are pressed. Each button is represented as a 1 when pressed and 0 when unpressed.

| Index | Button |
|---|---|
| 0 | A |
| 1 | B |
| 2 | X |
| 3 | Y |
| 4 | LB |
| 5 | RB |
| 6 | Back |
| 7 | Start |
| 8 | Logitech |
| 9 | L3 (Pressing the Left Stick in) |
| 10 | R3 (Pressing the Right Stick in) |

**Question 14:**

In our controller we mapped both analog and digital buttons. Analog signals are used for tasks that require a smooth and continuous (e.g. 0 through 1) representation of data, such as the joysticks, L3 and R3 Triggers, and the d-pad. We use digital signal for discrete (either 0 or 1), and accurate readings such as the buttons A, B, X, Y.