DESIGN ASSIGNMENTS - DUE DATE 12/17/2018

DESIGN ASSIGNMENT 1 [15]

**Arduino based Control:**

1A. Complete **Smart robot testing program** –

**Youtube Link:** [**https://youtu.be/dQ7wS0OxplE**](https://youtu.be/dQ7wS0OxplE)

This part was a simple demo of the bot executing some predefined controls.

Forward, Back, Left, Right, Circle, repeat.

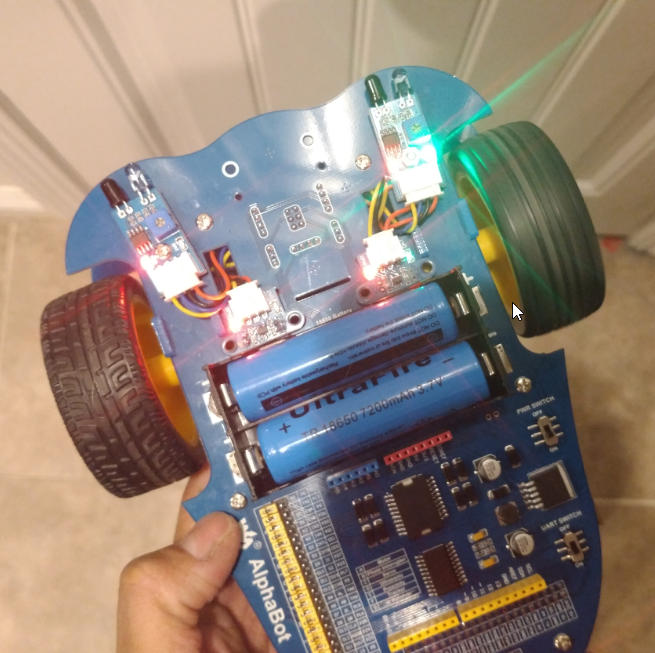
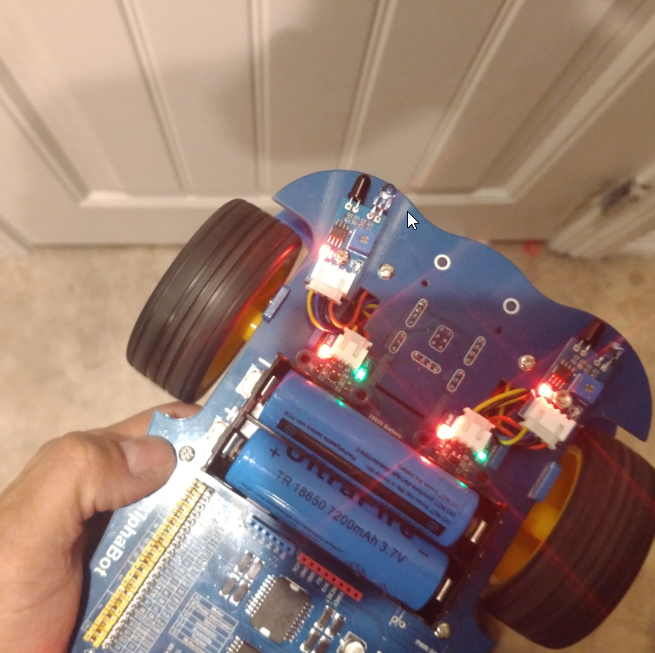
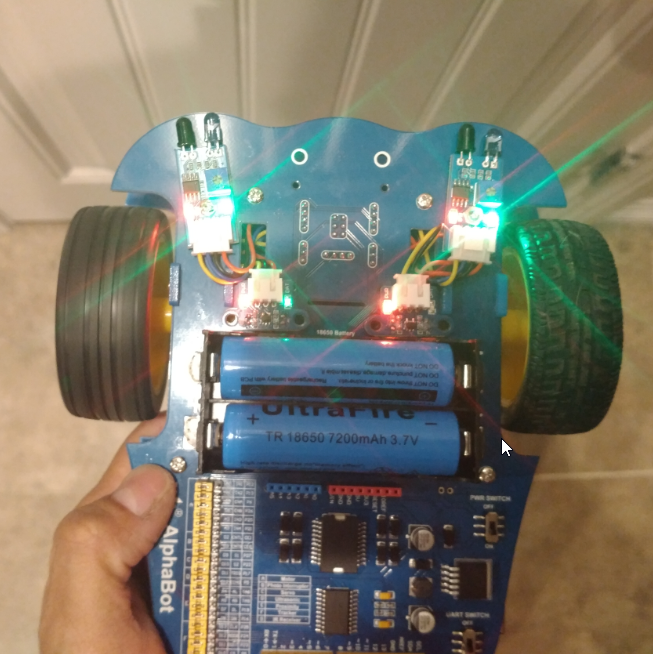
The motion is not exact this is to be expected because of the dynamic elements at play: friction, differences in motor torque etc. The motion is as stated above but begins to drift with time as explained in class.

1B. Complete **Infrared obstacle avoidance program** –

**Youtube Link:** [**https://youtu.be/AqsV\_Ya5vSo**](https://youtu.be/AqsV_Ya5vSo)

**Youtube Link:** [**https://youtu.be/LYSUiWQToDg**](https://youtu.be/LYSUiWQToDg)

This program defines pins 7 & 8 as the left and right sensors. The idea is that when calibrated to have an approximate equal reaction distance between the sensors as the bot approaches an obstacle which ever sensor reacts first will in turn cause the bot to turn in the opposite direction to avoid the obstacle. Once again dynamic elements are at play in the bots’ reactions and directionality elements such as friction, sensor calibration, sensor angles etc.



At same distance both sensors detect an obstacle as shown by the green DATA LED’s. In the center you see that when the right sensor has no obstacle, but the left does, the left wheel remains on while the right stops causing the bot to turn right and vice versa.

DESIGN ASSIGNMENT 2 [15]

**Raspberry Pi based Control:**

2A. **Infrared obstacle program** –

**Youtube Link:** [**https://youtu.be/Kbs7gq2DJV0**](https://youtu.be/Kbs7gq2DJV0)

This time we used the Rpi3 to execute the obstacle avoidance program. For the Rpi3 to use the AlphaBot the headers on the bot must be switched to have the bots’ modules interact with the Rpi pins.



DESIGN ASSIGNMENT 3 [15]

**Arduino and raspberry Pi combined control**

3A. **Webiopi**

**Youtube Link:** [**https://youtu.be/2aLHv9mTa9A**](https://youtu.be/2aLHv9mTa9A)

This again showcases the obstacle avoidance program however, in this case we used the WebIOPi to login into the bot and observed the commands being executed by the bot on a terminal.

DESIGN ASSIGNMENT 4 [20]

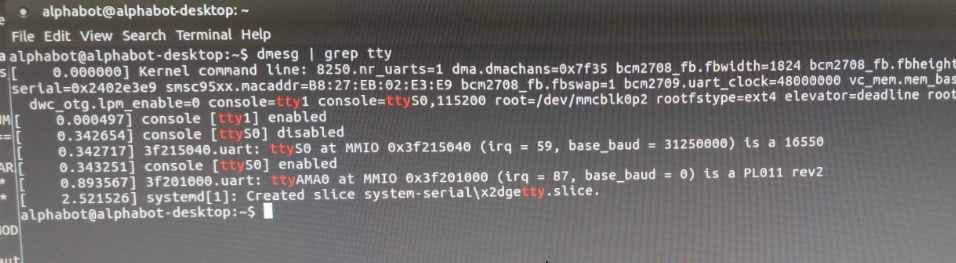
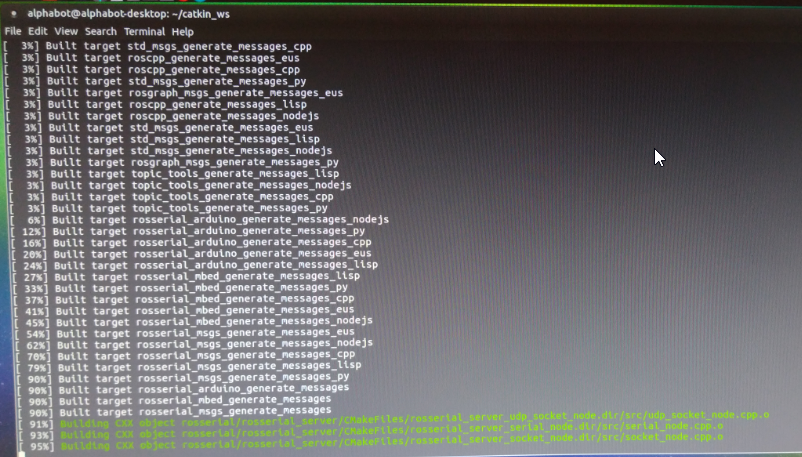
4A. Install ROS (any compatible version) on the Rpi3 of the Alphabot. Create a ros package called

“alphabot\_manual” based on roscpp

**Youtube Link:** [**https://youtu.be/ztJyftFCXco**](https://youtu.be/ztJyftFCXco)

Unfortunately, I was unable able to interface the Joycon with the Arduino/Pi setup. I was unable to install the rosserial libraries. The Pi would keep freezing at around 95% of building an object. I let it sit for approximately 40 minutes, but no progress was made three times. I even attempted to use a make shift heat sink in case that was the issue but no luck. I was able to successfully launch the .launch file with ros but got errors, the videos show this. I modified some of the tty settings and some serial commands seemed to work. Further work is needed to get the Joycon working.

4A.2 Perform manual controlled motion of alphabot using wired or wireless joystick.



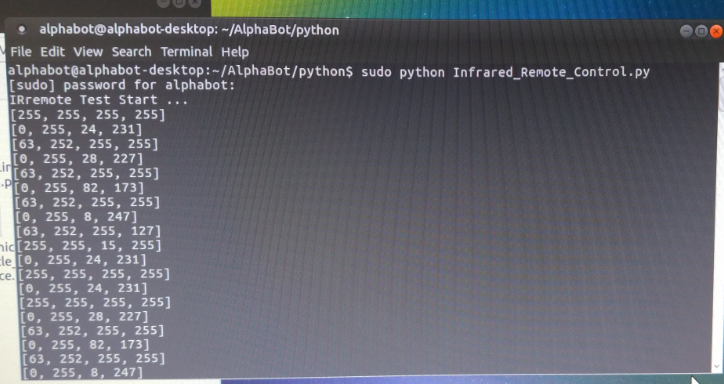


4A.3 Perform manual controlled motion of alphabot using IR RX-TX Controller.

**Youtube Link:** [**https://youtu.be/\_6y20aZlsGk**](https://youtu.be/_6y20aZlsGk)

**Youtube Link:** [**https://youtu.be/XZ6lSuPTOTY**](https://youtu.be/XZ6lSuPTOTY)

The easiest of the three requests it worked easily with just the Pi. The Arduino code for some reason didn’t function for the IR controller.



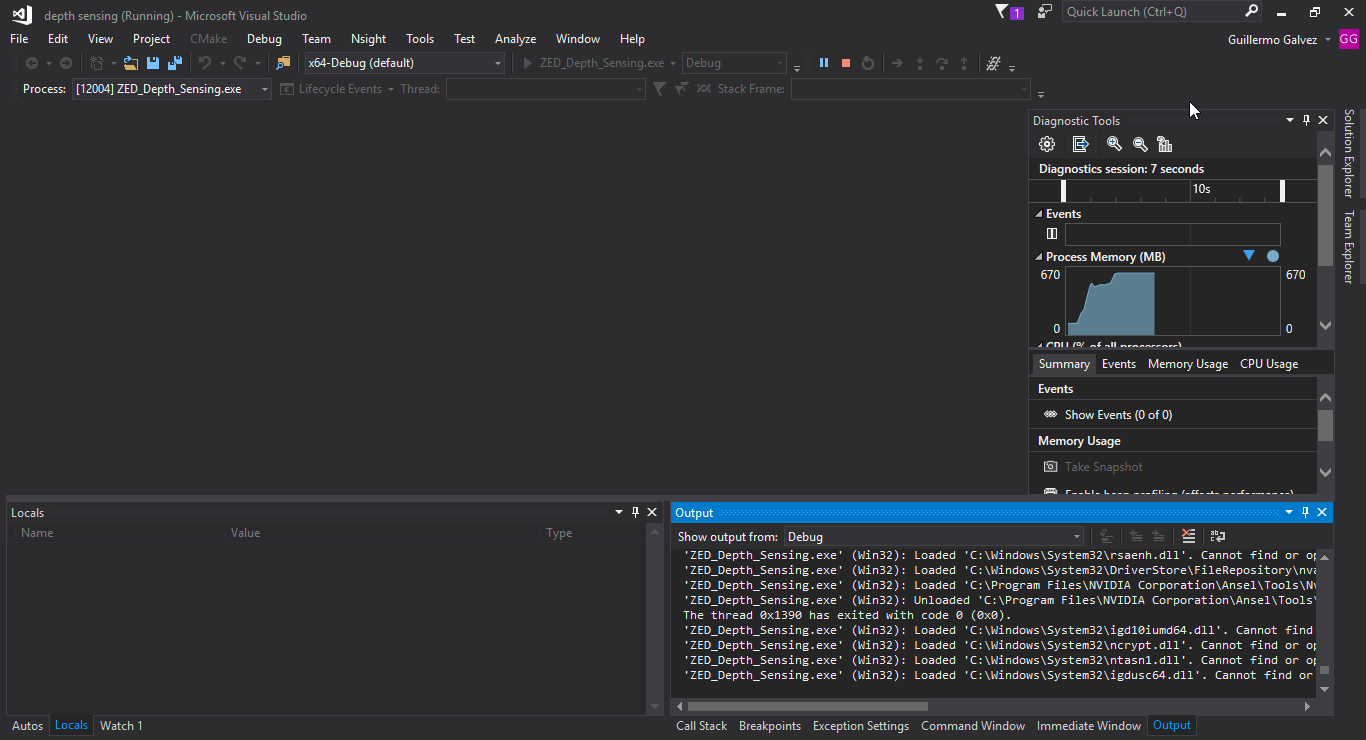
DESIGN ASSIGNMENT 5 [35]

5A. Interface the provided RGBD sensor to ROS on the host machine to collect 2D or 3D PC data

This task was the most time consuming in the end because of all the time that was spent trying to get the ZED stereo camera to work with a VM. In the end the most I was able to do is build and runs some of the examples in Visual Studio Community.

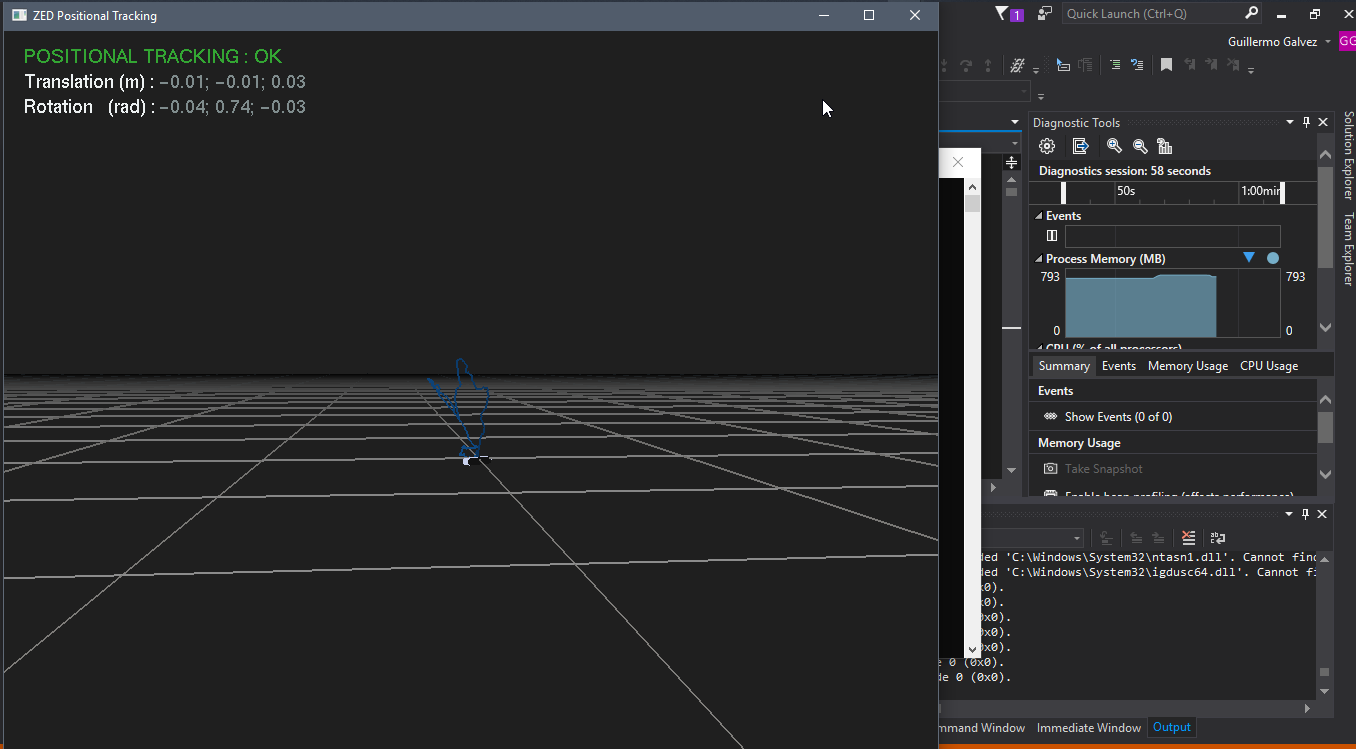
Depth Sensing:

**Youtube Link:** <https://youtu.be/2ffU1IR6N-o>



Positional Tracking:

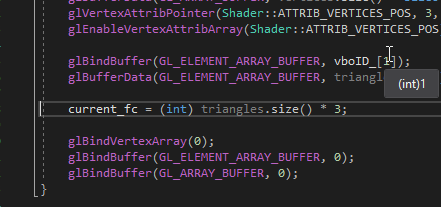
**Youtube Link:** <https://youtu.be/QJvEzi2Ns80>



Spatial Mapping:

**Youtube Link:** <https://youtu.be/lliTvJu6Uns>

Spatial mapping could not run continuously. An exception is triggered stating there is an access violation likely somewhere in this area of the coding. Of course, just because this area of the coding is where the exception happens it doesn’t mean that the issue is necessarily here.



5B. Using the above generated data or any map data available demonstrate any of the localization,

I know this is what you wanted to most but without getting the zed to working correctly it is hard to implement a SLAM being such a novice. Will continue to try over break.