CE 4DS4 - Project 2 - Autonomous Vehicle: Putting it All Together

Professor: Dr. Mohamed Hassan

Section L04

Khaled Hassan, hassak9, 400203796
Nikola Petrevski, petrevsn, 400198379
Zekai Yang, yangz203, 400308646

Due: Thursday, April 4, 2024

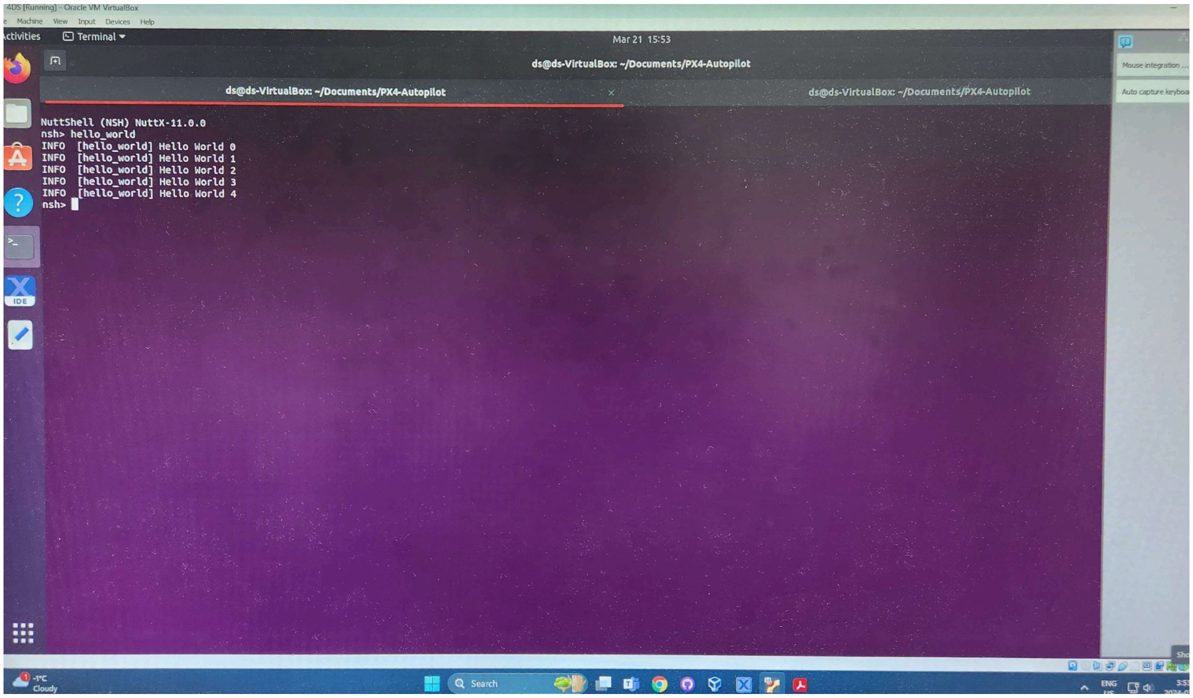
As a future member of the engineering profession, the student is responsible for performing the required work in an honest manner, without plagiarism and cheating. Submitting this work with my name and student number is a statement and understanding that this work is my own and adheres to the Academic Integrity Policy of McMaster University and the Code of Conduct of the Professional Engineers of Ontario.

Submitted by: Nikola Petrevski (petrevsn, 400198379), Khaled Hassan (hassak9, 400203796) & Zekai Yang (yangz203, 400308646)

Declaration of Contributions

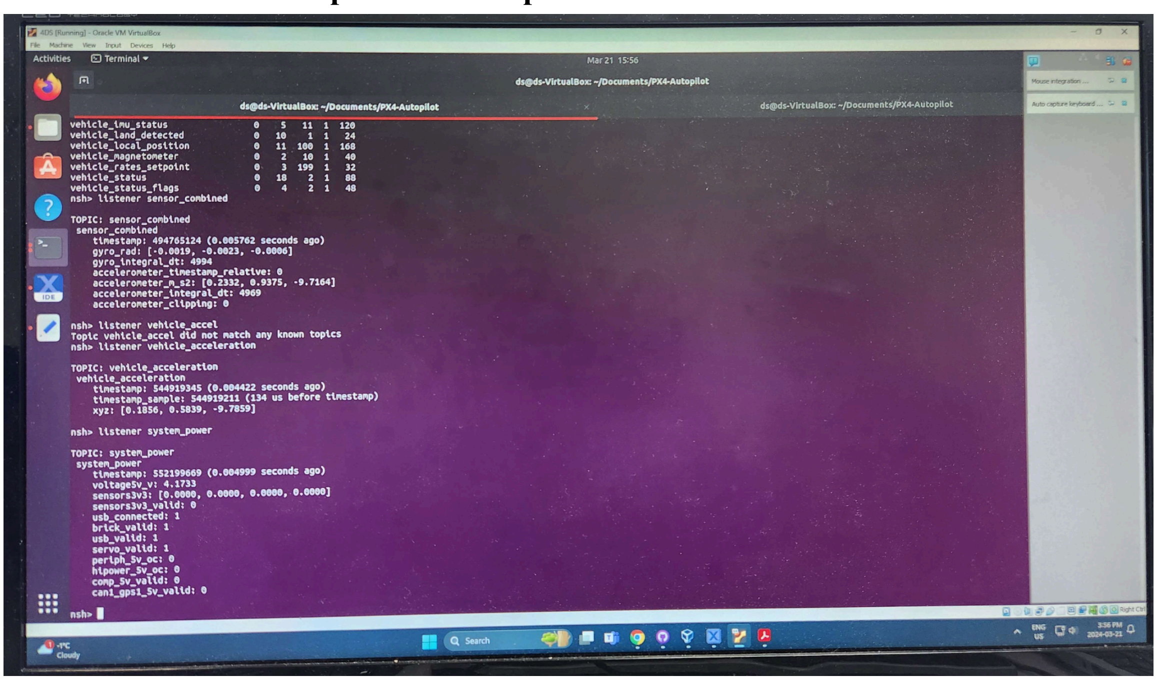
As a group, we tried our best to split all of the work equally. Even if one person physically typed code out for a part, every group member was always present and contributing to the completion of each part of this project.

Experiment 5.2B: Hello World Application



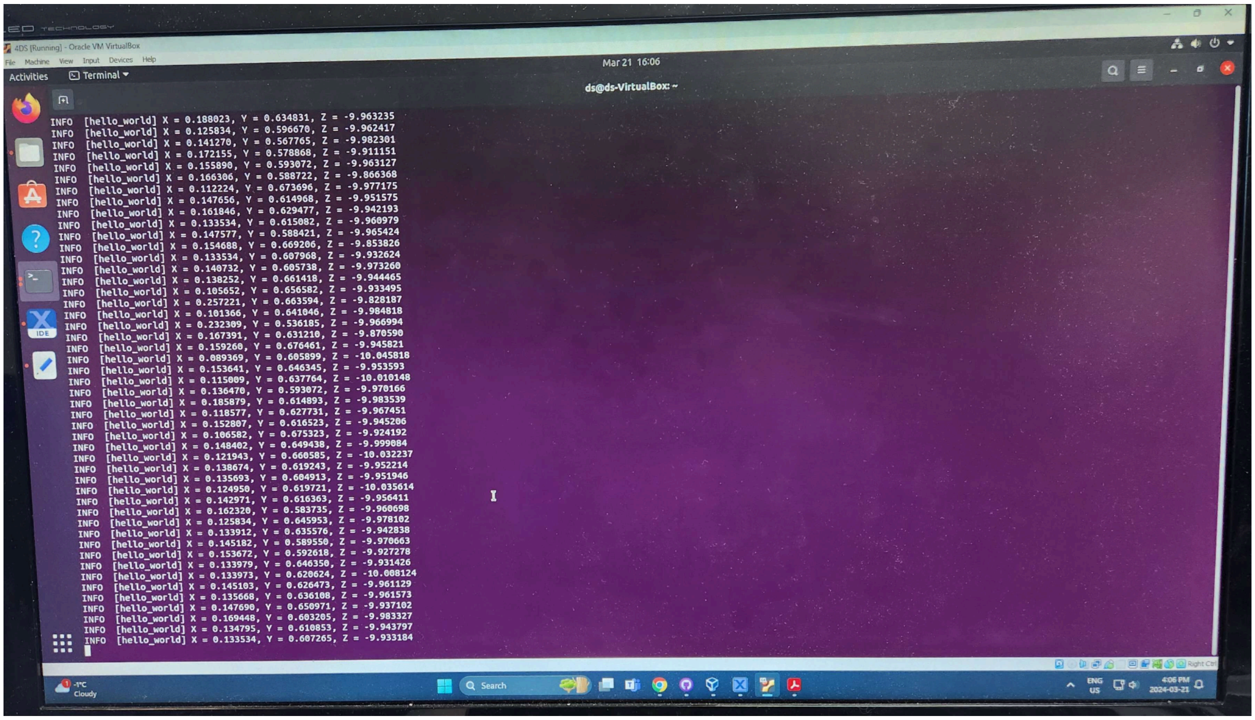
Experiment 5.3: uORB Messaging

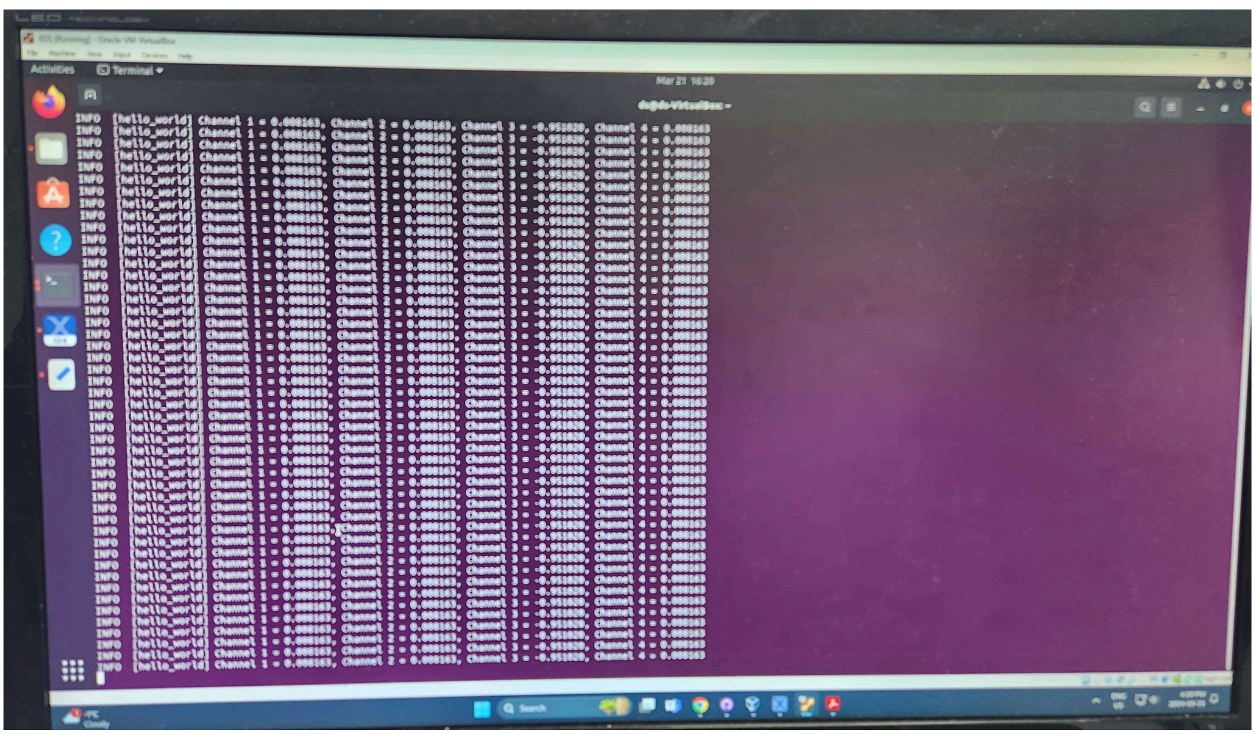
Experiment Setup A: Interact with uORB



Project 2 - Step 0

For this step, we subscribed to the rc_channels data through uORB, and passed their value into a variable. After this, it was simply a matter of creating an infinite while loop and printing each value to the console, as seen below.





Project 2 - Part 1

Part A - Controlling the DC and servo as in Experiment 3C

For this part, the main purpose was to take in user input for the servo angle and the DC motor speed, and send them to their respective motors. To do this, we prompt the user to enter a value between zero and one. This value is then simply directly uploaded to each motor by publishing the inputted values to the test motor uORB topic. A value of 0.5 stops the DC motor and centers the servo motor. If a value outside the range of 0 to 1 is entered, an error message is printed.

Part B - Using the RC to control the motors

The function of this part is similar to part A, only now the speed and angle of the motor are controlled by the RC. To do this through PX4, we subscribe to the rc_channels uORB topic to continually receive updated information on the value of the rc channels. For our implementation, Channel 3 controlled the speed (left thumbstick), and channel 1 controlled the angle (right thumbstick).

The values that were received from the RC channels had a range of -1 to 1, but the motors only accept values between 0 and 1. Therefore, we added 1 and divided each number by 2 in order to normalize them from a range of 0 to 1. After these values are calculated, the DC motor speed and servo angle are updated similar to in part A.

Project 2 - Part 2

For this part, there are two separate codes which run in order to achieve the desired behavior. First, there is the python script that runs on the raspberry pi. This script is responsible for setting up the mavlink connection, reading distance data from the ultrasonic sensor, reading direction data from the camera, and then sending the appropriate commands to the FMU. In order to make the data transmission as easy as possible, it was determined that only a single integer value should be sent over the mavlink connection. To achieve this, we used the fact that there are only 9 unique commands that we would send to the FMU. For each distance threshold (<15cm, 15cm < distance < 50cm, >50cm), there is a fixed speed, and the servo will only be turned all the way left, straight, or right. Therefore, based on the data seen from the sensors, we can encode the servo and DC motor information that the FMU needs into a single integer value, which is sent over mavlink. Distance is determined through the ultrasonic sensor which sends out a signal that is then received back by the sensor itself. The time this process takes is used to calculate distance, similar to the working principle of a lidar system.

Next, there is the C code which runs on the FMU. Once the integer code is received from the raspberry pi, the FMU has to decode this value and translate it into meaningful values that can be uploaded to the servo and DC motors. The motor speed and angle default to a value of 0.5

in order to keep them at rest to begin. Then, there are simply 9 if statements in the code which set the appropriate speed and angle before the values are updated to the respective motors.