1. **Introduction and Lab Description :**

For our first lab we will be using one set of encoders and our own code to control the robot and make it follow a predetermined path with a predetermined speed. We will use and draw upon multiple pieces of hardware in the lab in order to build our software for the tasks. Given a set radius and speed we will be instructed to make our robot follow a semi-circle within the given time.

1. **Materials:**
2. VNC software Download
3. Arduino kit
4. Raspberry Pie
5. **File List:**
6. Encoders.py
7. Driver.py
8. **Task 1:**

To complete task 1 we started by using our CalibrateSpeed() function in order to grab the instantaneous speed values at different intervals. Based on different incremented PWM motor values that we have recorded with the CalibrateSpeed() function, we were then able to set our speed based on user input. To change the speed all we have to do for RPS is input our desired speed and the bot will look at its data and select the appropriate speed based on the mapped speed for that value. Our next step was to calculate IPS so we could change our speed value based on our measurements in inches traveled per second. We were able to do this by calculating the circumference and dividing it by 32. Every tick that goes by would indicate a section and every 32 is a revolution. So using this data we are able to calculate IPS.

1. **Task 2:**

Our second task was to move the robot in a straight line in a specified time. We used the input time seconds to get our needed speed. From here we were able to use our setSpeedRPS() function in order to keep the wheels going the same speed but in opposite direction. The inversing of the wheels allows for movement in a single direction, since the wheels are in fact mirrored. Once the speed needed is calculated the robot will move in a straight line at the given speed. If the speed given is to fast it will just use the max speed the servos can currently go and vice versa for values to low.

1. **Encoders.py/Driver.py:**

In order to make our code more compact we first combined our servo code with our encoder code into one file. This file would hold all of our functions and allow our driver file to call upon them in an efficient and reusable way as seen in the code snippets below:

(Add code snippets here)

1. **Data Analysis/Results/Youtube Video:**

(When functions are finished and we are able to code for circle we can fill this in.)

1. **Conclusion:**

(Finish this the 4th or 5th upon finishing.)

**References**

**(Possibly reference TA or professor maybe even lecture slides)**