Names: Kieran Stone, Andrew Lan, Mateo Mann

Part 1:

1. x: -0.280021, y: -6.41761e-05, z: 0.209972

Part 4:

1. The robot reads the ground sensor values, then if its state is speed_measurement it moves forward until it detects the start line, sets is new state to line_follower and then calculates its speed. Then if its state is line_follower it wil update the wheel speed based on the ground sensor inputs so that it stays on the line. Then it calculates the odometryand updates pose_x, pose_y and pose_theta. Then it checks if it has found the start line by checking if the ground sensorse are under 400 for 0.1 seconds and if it finds the start line it resets the pose_x, pose_y and pose_theta values. Finally, it reports its current pose and sets the wheel velocity before repeating the loop.
2. The robot will move faster or slower depending on how the timestep varies
3. Its speed it 0.166 m/s
4. It's pose should show 0,0,0
5. We implemented the loop closure code by checking if all the ground sensor values are less than 400, if they are we check if the start_line_time is None (it is initialized to None outside of the while loop). If the start_line_time is None then it gets set to the current time. If it is not none then it checks if the difference between the current time and the start_line_time is greater than 0.1. If it is greater than 0.1 then the pose values are reset to 0. If not all of the sensor values are less than 400, then the start_line_time gets set to None.
6. We spent about 6 hours on this lab
7. The implementation works quite well. The robot follows the line very closely and executes precise turns on the corners. It correctly detects the starting line and does not accidentally detect it anywhere else on the board. The odometry works pretty well. It doesn't exactly get to 0,0,0 as it approaches the starting line, but it is very close.