1. Justin Vuong and Alex Book
2. It advances the controller time step of the specified duration.
3. If you don't measure the time elapsed since the last odometry update you will get the wrong values because you'd be passing in the wrong value to use to calculate. This would affect the position estimation since the time elapsed value is used to multiply by the x_dot_i and y_dot_i values to get the pose_x and pose_y. If the time elapsed is less than reality, the position estimation would fall short of what it should be, and if the time elapsed is greater than reality, the position estimation would be greater than what it should be.
4. The average speed of the ePuck when covering the 10 cm distance in part 1 was roughly .13 m/s.
5. In an ideal world, the ePuck's pose should show [0, 0, 0] each time when crossing the starting line.
6. If the robot senses the start line (all 3 sensors sense a line) for greater than .4 seconds, the pose is set to [0, 0, 0]. In order to combat the slow corner turns where all three sensors sense a line, we checked that the position of the robot is near the start line while seemingly sensing the start line.
7. We spent roughly 4 hours programming this lab.
8. Yes our implementation works as we expected.