

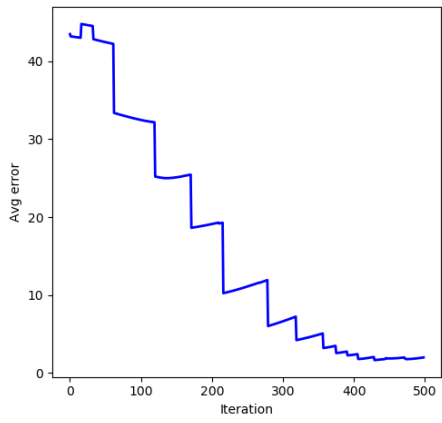
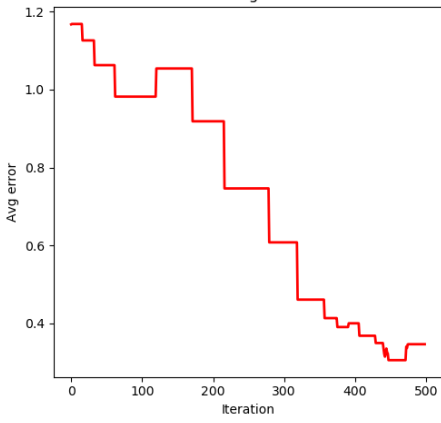
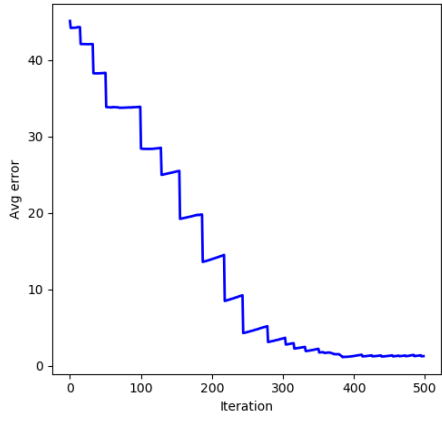
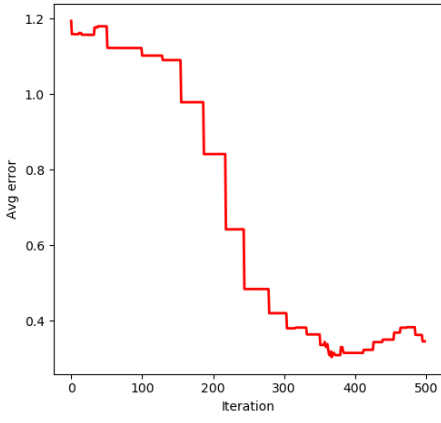
Name: Chun-Yen Yeh (ID: cyeh4), Huey-Chii Liang (ID: hcliang2)

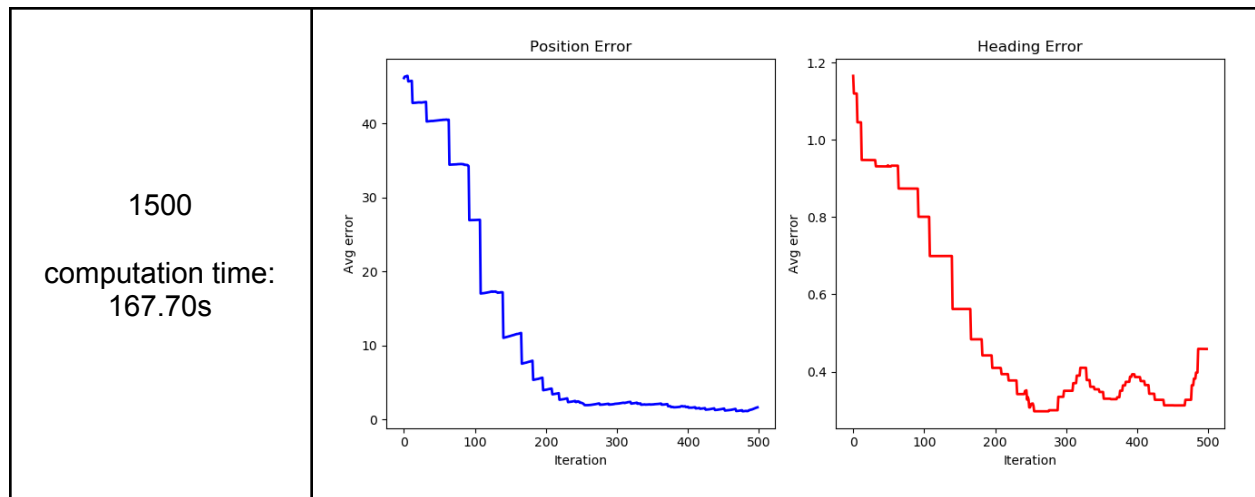
Course: ECE484 - Principles of Safe Autonomy

Date: October 26, 2023

ECE484 - MP3 Group Assignment

Problem 4 (30 points). Using 4 measurement directions, keep the sensor limit constant at 15, run your algorithm with the number of particles 500, 1000, 1500. How does changing the number of particles influence the estimation accuracy, converging speed and computational cost of the algorithm? Record a video of one of the three runs. The video should include the turtle map window. Provide a link to the video and include it in the report.

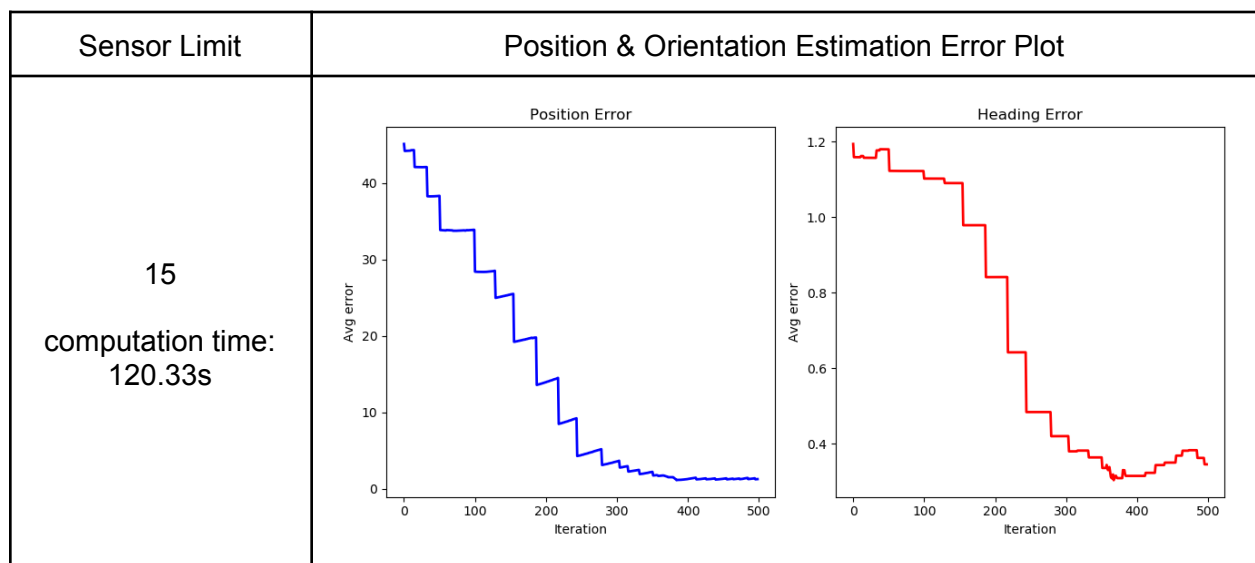
| Number of Particles | Position & Orientation Estimation Error Plot |
|--------------------------------------|---|
| 500 computation time: 99.11s | <div><div><p>Position Error</p><p>Avg error</p><p>Iteration</p></div><div><p>Heading Error</p><p>Avg error</p><p>Iteration</p></div></div> |
| 1000 computation time: 120.33s | <div><div><p>Position Error</p><p>Avg error</p><p>Iteration</p></div><div><p>Heading Error</p><p>Avg error</p><p>Iteration</p></div></div> |

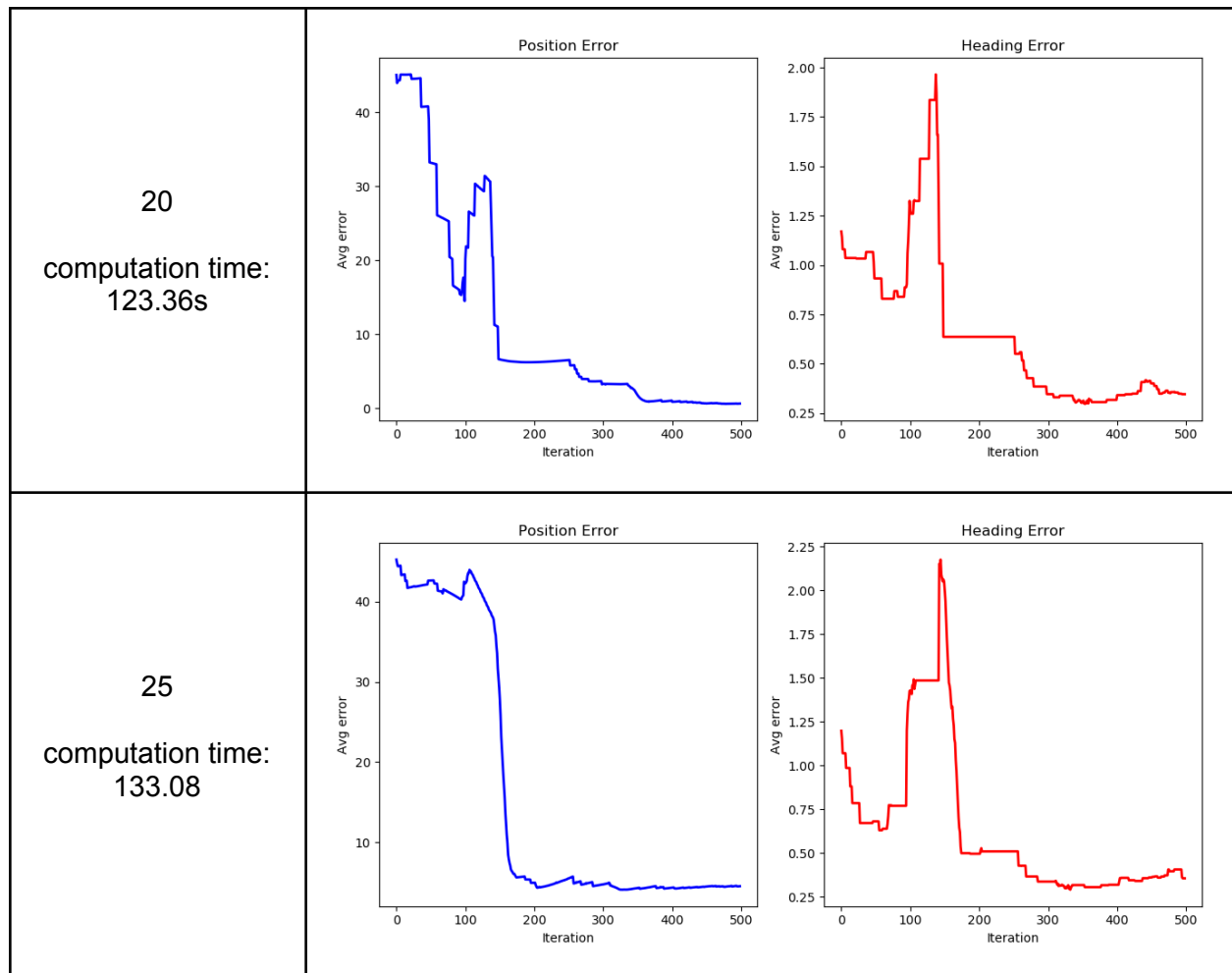


All settings successfully locate the correct target position based on videos. As pictures of position error show, the more particle number initiated, the faster the particles converge and locate the correct target position.

Videos: [particles-500](#) / [particles-1000](#) / [particles-1500](#)

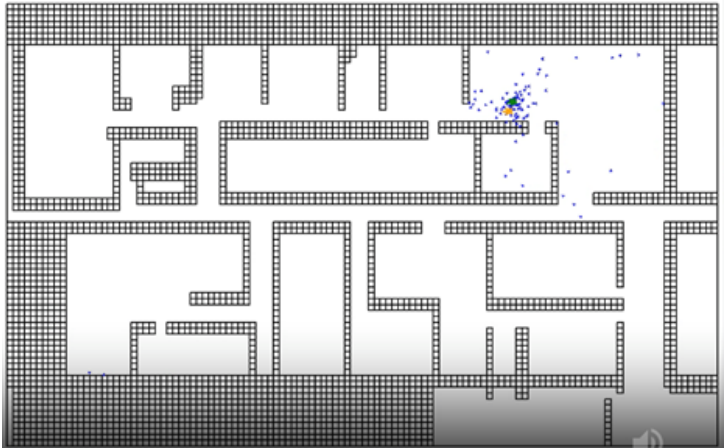
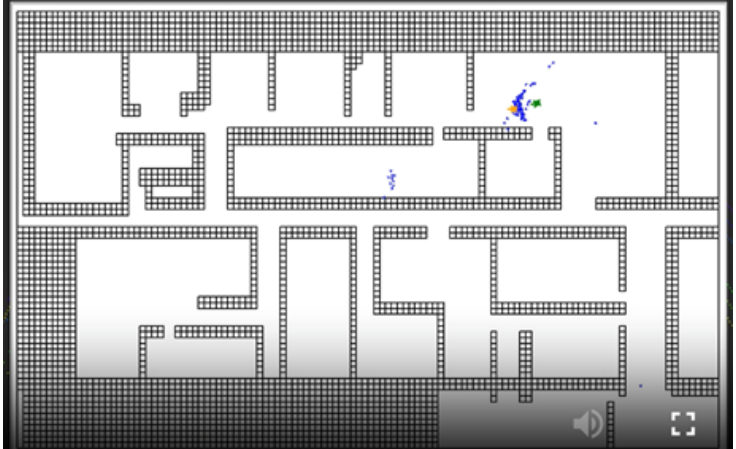
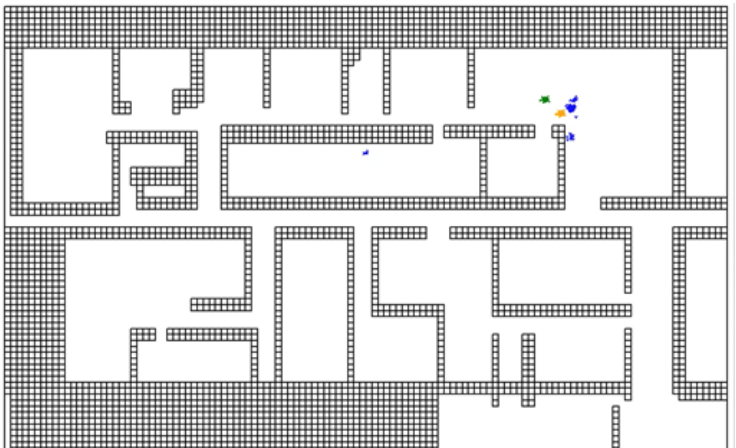
Problem 5 (30 points). Using 4 measurement directions, keep the number of particles constant at 1000, run your algorithm with sensor limit 15, 20, 25. How does changing the sensor limit influence the estimation accuracy, converging speed and computation cost of the algorithm? Record a video of one of the three runs. The video should include the turtle map window. Provide a link to the video and include it in the report.





All settings successfully locate the correct target position based on videos. As pictures of position error show, sensor limit ≥ 20 can lead to sharp error drop around iteration 500. In contrast, sensor limit = 15 presents a gradual error drop process. We believe it is because larger sensor limits provide more accurate information for particles to locate. The convergence is also demonstrated in the following video frames.

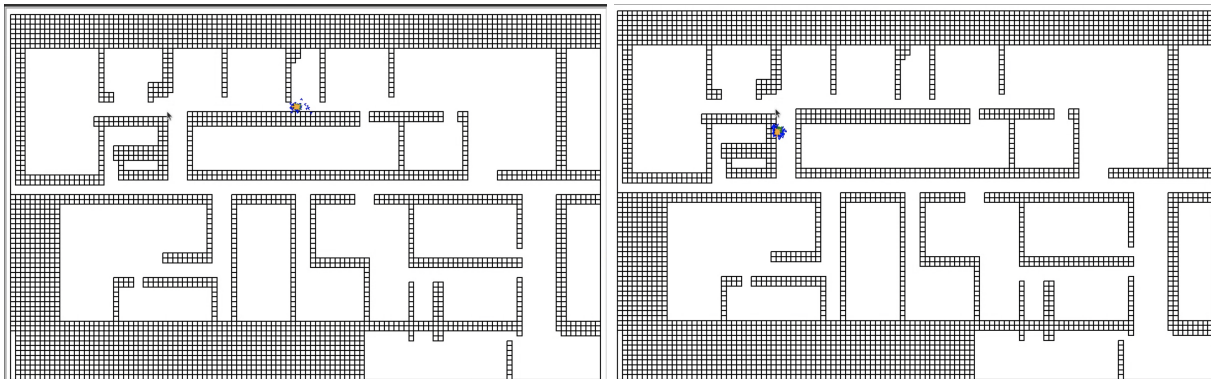
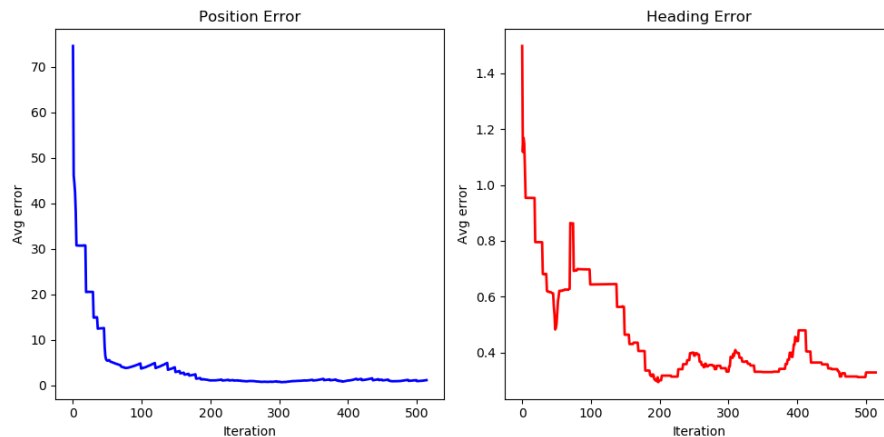
| Sensor Limit | Convergence |
|--------------|-------------|
|--------------|-------------|

| | |
|----|---|
| 15 |  <p>A top-down view of a maze environment. The maze is composed of black walls on a white grid. A small cluster of blue dots is located in the upper right quadrant, near a junction. A single green dot is also visible within this cluster. The bottom right corner of the maze shows a dark grey area, possibly representing a goal or a different terrain type.</p> |
| 20 |  <p>The same maze environment as in the previous frame. The cluster of blue dots has shifted slightly towards the top right. A single green dot is still present. The bottom right corner remains dark grey.</p> |
| 25 |  <p>The same maze environment. The cluster of blue dots is now more dispersed and has moved further towards the top right. A single green dot is still visible. The bottom right corner is dark grey.</p> |

Videos: [sensor-15](#) / [sensor-20](#) / [sensor-25](#)

Problem 6 (10 points). Does the particle filter you implemented performs evenly well through the whole environment after converging? More specifically, does your particle filter have larger prediction errors in some regions of the environment than other regions? If yes, can you explain why this is happening?

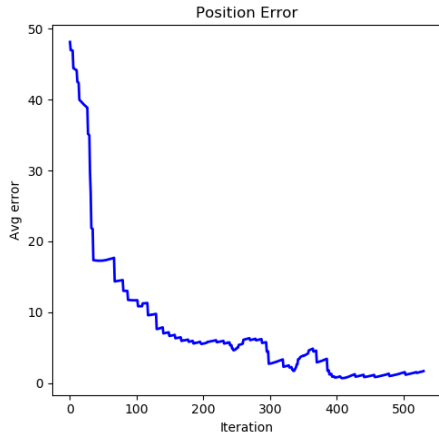
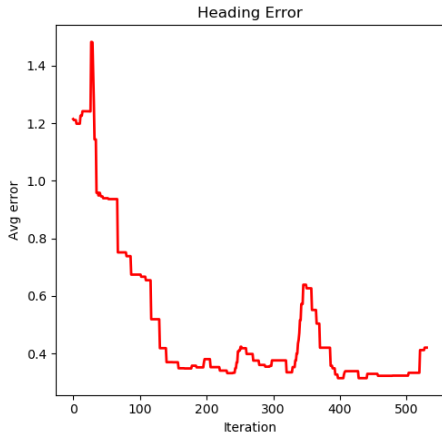
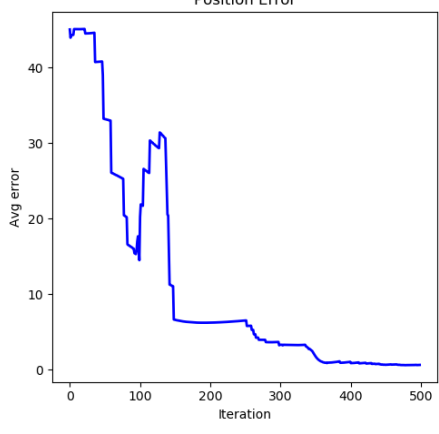
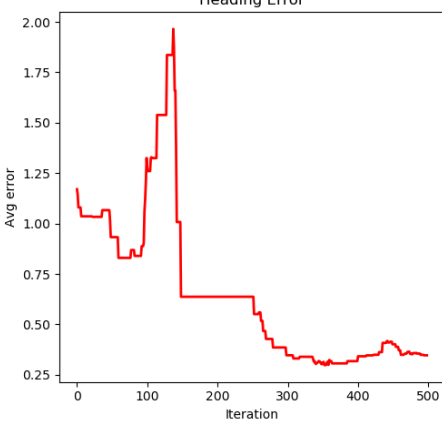
No, our particle filter algorithm performs evenly well across the whole environment after converging. Based on the position error below, it indicates that error remains at a low level once it has converged. We believe the reason for this stability is that the motion model of particles is shared with the target. Therefore, once particles converge at a correct target position, they follow the target along the way through the whole environment.



Videos: [whole-env](#)

Problem 7 (20 points). Modify the LidarProcessing module and the sensor model so that they can make measurements in 8 directions. Run your algorithm with the number of particles 1000 and sensor limit 20. How does having more sensor data influence the estimation accuracy and converging speed of the algorithm? Record a video of the run. Besides the turtle map window, the video should include the RViz window of the sensor measurements. Provide a link to the video and include it in the report.

The table below shows the estimation error plot of the algorithm with 8 lidar sensors and its comparison with the algorithm with 4 lidar sensors. Both have 1000 particles, and a sensor limit of 20m. We can see that the more sensor data, the more accurate the estimation and the faster the convergence. This phenomenon is intuitive: the more lidar sensors, the clearer the car's perception of the environment, so the more accurate the estimation of the heading angle.

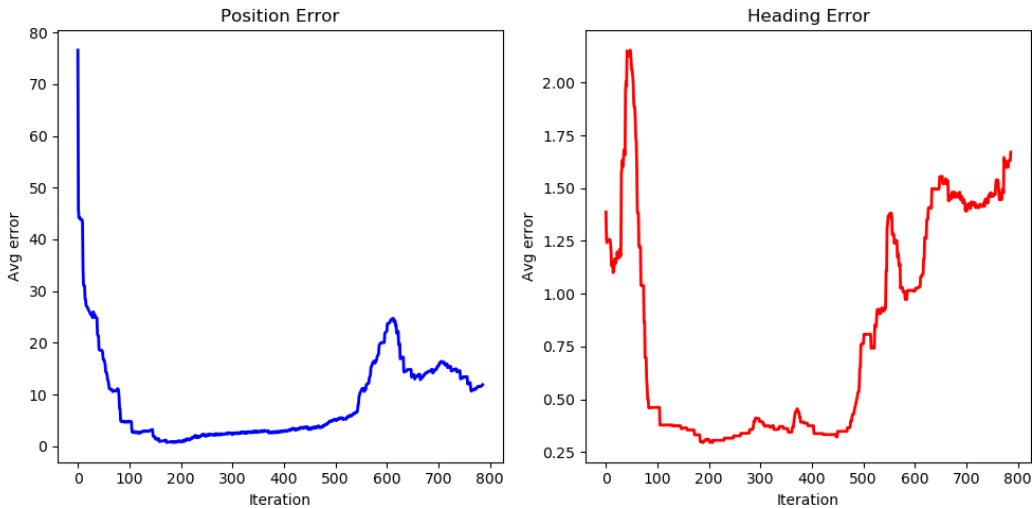
| Number of Sensors | Position & Orientation Estimation Error Plot |
|---|---|
| <p>8</p> <p>Computation time: 202.75s</p> | <div>   </div> |
| <p>4</p> <p>computation time: 123.36s</p> | <div>   </div> |

Videos: [lidar-8](#)

Problem 8 (5 bonus points). Activate the measurement noise as outlined in Section 3.4. This will result in occasional absence of the Lidar measurements. Run your algorithm with particles 1500 and sensor limit 25. How does the missing lidar measurement influence the estimation accuracy and converging speed? Provide a link to the video and include it in the report.

The lack of lidar measurements makes the estimation unstable, as shown in the figure below. The reason is that if the lidar measurement returns “None”, we cannot calculate the likelihood

between the sensor readings of the vehicle and the particles, so we cannot do the weight update and resampling at this timestamp. However, the vehicle is still moving. As a result, the particles tend to lose some important information about the car's motion, leading to errors in estimation of position and orientation that grow larger over time.



Computation time: 226.20s

Videos: [measure_noise-1](#) / [measure_noise-2](#)

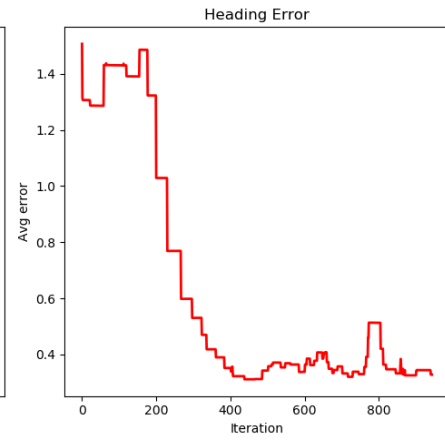
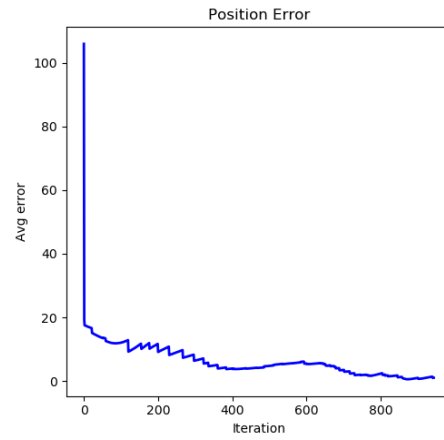
Problem 9 (5 bonus points). Modify the distribution of your initial particles to be within the top-right quadrant. Run your algorithm with particles 500 and sensor limit 15. How does the initial distribution of the particles influence the estimation accuracy and converging speed? Provide a link to the video and include it in the report.

The following table shows the estimation error plot of the algorithm for different initial particle distributions. Both algorithms have 500 particles and a sensor limit of 15m. We can find that if the range of the initial particle distribution is smaller, the algorithm converges faster. This is because there are more particles closer to the actual vehicle position at the beginning, and more particles will be near the actual vehicle position after resampling.

| | |
|-----------------------------------|--|
| Distribution of initial particles | Position & Orientation Estimation Error Plot |
|-----------------------------------|--|

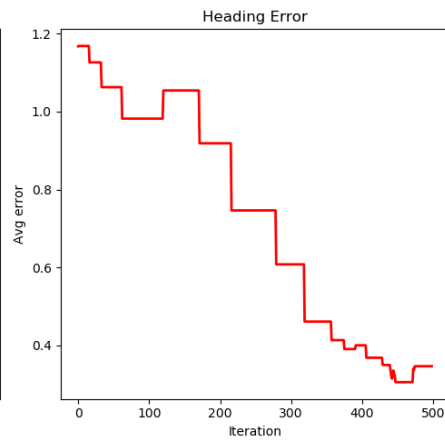
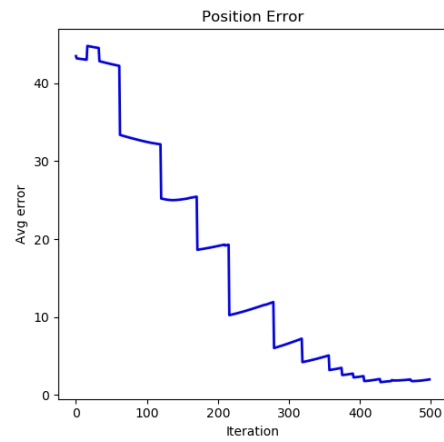
Top-right quadrant

Computation time:
198.92s



The whole map

computation time:
99.11s



Videos: [top-right](#)