UDACITY

‹ Return to Classroom

Kidnapped Vehicle

REVIEW

CODE REVIEW

HISTORY

Meets Specifications

Dear student,

This is a very great attempt as I can see you put a lot of effort to accomplish this project. I enjoyed reviewing your project as your code is very well structured and you have further divided your code into additional functions which makes it easy to follow. I am sure that you also enjoyed working on the project.

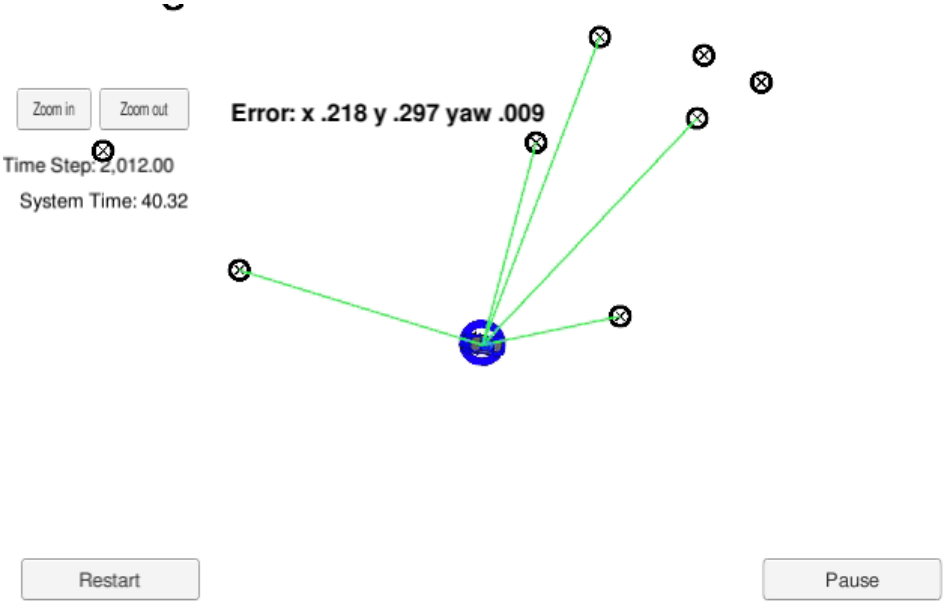
- If you are interested to learn more about the topic, you might want to check *Sebastian's article* about the particle filter in robotics.

Congratulations 🎉 and keep up the good work for the next project!

Accuracy

This criteria is checked automatically when you do ./run.sh in the terminal. If the output says "Success! Your particle filter passed!" then it means you've met this criteria.

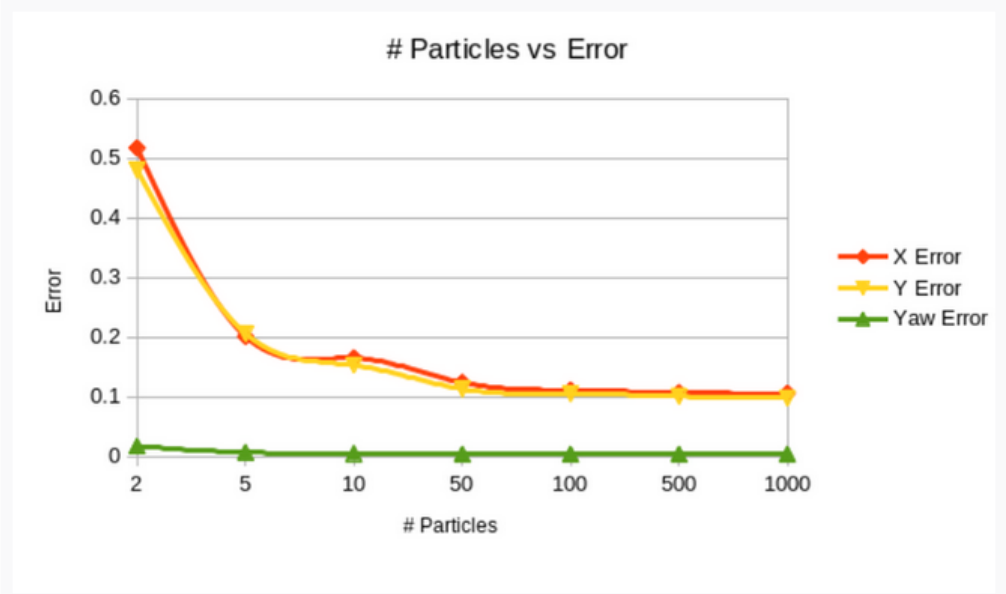
The particle filter implementation passes the grading code in the simulator. Awesome.



Performance

This criteria is checked automatically when you do ./run.sh in the terminal. If the output says "Success! Your particle filter passed!" then it means you've met this criteria.

```
num_particles = 5;
```



Great choice. It is neither too small which would prone to error nor too high as that would be computationally expensive.

```
        temp_land_mark.x = particles[il.x + (cos(particles[il.theta) * observations[il.x) - (sin(particles[il.theta) *
```

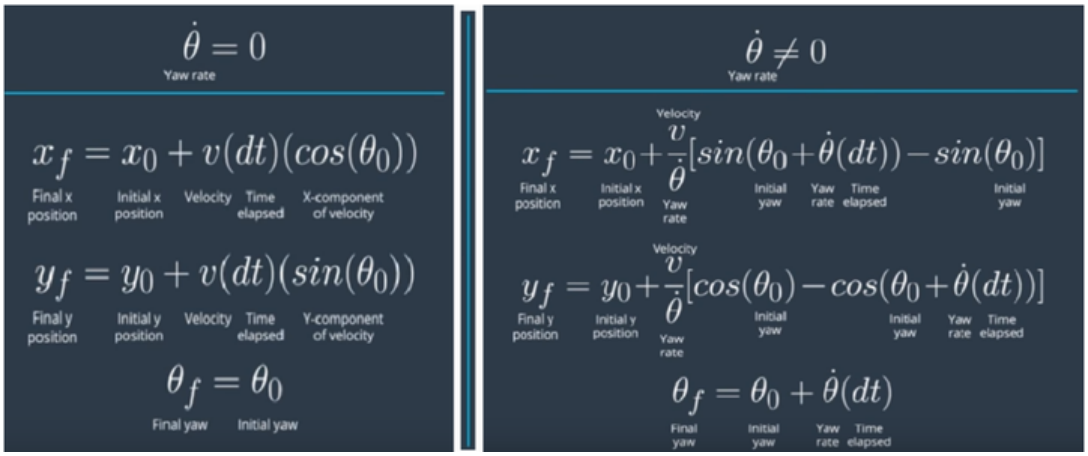
```
observations[j].y);
        temp_land_mark.y = particles[i].y + (sin(particles[i].theta) * observations[j].x) + (cos(particles[i].theta) *
observations[j].y);
```

Nice work transforming the observations from car coordinates to map coordinates.

`if(yaw_rate > 0.000001)`

Good job using the general model else linear model when the yaw rate is close to zero to avoid division by zero.

```
if (yaw_rate is close to zero)
    Use linear Model
else
    use the General Model
```



I went through the rest of the code, and all the functions are properly implemented. Kudos!! 👏🏻

General

There may be ways to "beat" the automatic grader without actually implementing the full particle filter. You will meet this criteria if the methods you write in particle_filter.cpp behave as expected.

The methods in `particle_filter.cpp` behave as expected. All the steps in particle filter(Initialization, Prediction, Update and Resample) is correctly implemented. The code is clear and logical. Nice work.

Further Reading

- You may also check this *paper* which discusses the resampling methods that allow for increased speed of particle filter and require less memory.
- This *research paper* discusses object tracking in a video using particle filter which is an interesting read to get an idea of real world particle filter applications.

DOWNLOAD PROJECT

RETURN TO PATH

Rate this review

START