Kidnapped Vehicle Project

Introduction

A robot has been kidnapped and transported to a new location! Luckily it has a map of this location, a (noisy) GPS estimate of its initial location, and lots of (noisy) sensor and control data.

In this project, I implemented a 2-dimensional particle filter in C++. My particle filter was given a map and some initial localization information (analogous to what a GPS would provide). At each time step my filter also get observation and control data.

Running the Code

The programs that have been written to accomplish the project are: src/particle_filter.cpp, and src/particle_filter.h. The program src/main.cpp has already been filled out, no need to modify it.

This project made a 'build' directory in the project folder, and start compiling by doing the following:

```
cmake ..
make
```

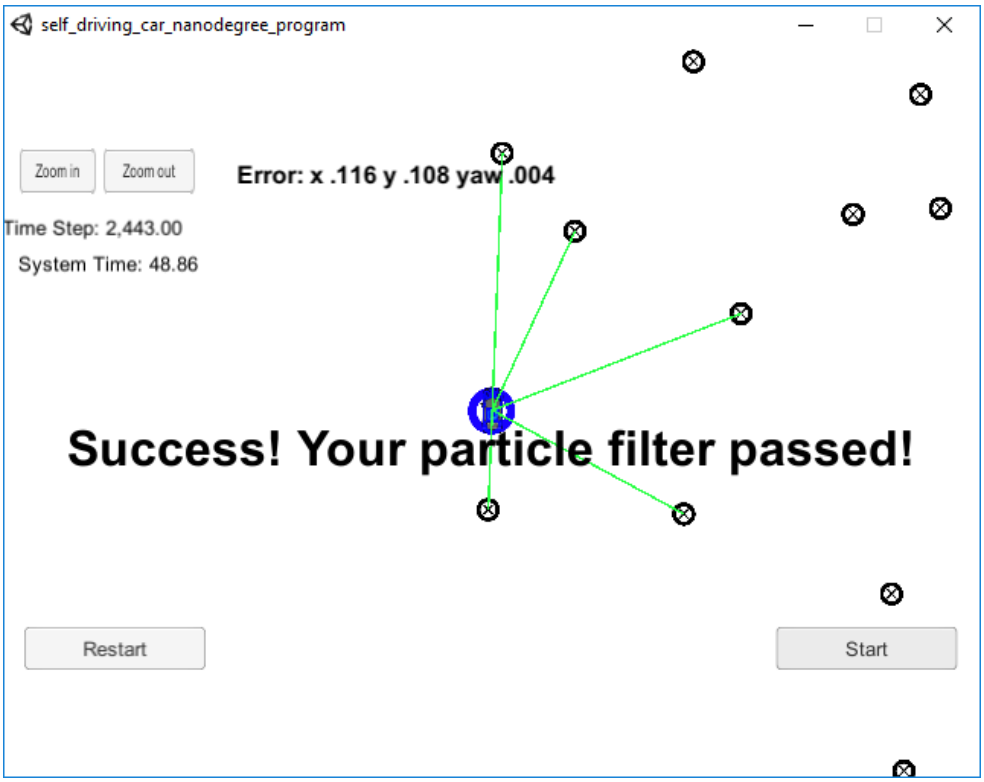
The C++ code will be compiled without errors.

Accuracy

To start the simulator, launched it and select the EKF/UKF project, running command:

```
./particle_filter
```

Then, click start button in the simulator. The running results is as following picture:



From the picture above, you can see that the output says "Success! Your particle filter passed!". It means our project has met the criteria on accuracy.

Performance

The particle filter should complete execution within the time of 100 seconds. Overtime will result in a message as 'You ran out of time' on the screen of the simulator.

From the picture above, we can see the execution time is 48.86 second – It's only less than a half of requested time to be used. So, our project has met the criteria on performance.

General

The project meets this criteria on general because the methods I write in particle_filter.cpp behave as expected.

Conclusion

This project implements the particle filter on a bicycle motion model to perform its prediction step when translating the initial particle positions followed by a nearest neighbor data association to associate each measurement with a landmark. Concretely, each measurement vector received by the particle filter is first transformed into map coordinates and then associated with a landmark ID provided in the given map using the minimum Euclidean distance between the predicted measurement and actual landmark coordinates.