Extended Kalman Filter Project

Self-Driving Car Engineer Nanodegree Program

In this project I utilize a kalman filter to estimate the state of a moving object of interest with noisy lidar and radar measurements. Passing the project requires obtaining RMSE values that are lower than the tolerance outlined in the project rubric.

This project involves the Term 2 Simulator which can be downloaded here

This repository includes two files that can be used to set up and install uWebSocketIO for either Linux or Mac systems. For windows you can use either Docker, VMware, or even Windows 10 Bash on Ubuntu to install uWebSocketIO. Please see this concept in the classroom for the required version and installation scripts.

Once the install for uWebSocketIO is complete, the main program can be built and run by doing the following from the project top directory.

1. mkdir build
2. cd build
3. cmake ..
4. make
5. ./ExtendedKF

Tips for setting up your environment can be found here

Note that the programs that need to be written to accomplish the project are src/FusionEKF.cpp, src/FusionEKF.h, kalman_filter.cpp, kalman_filter.h, tools.cpp, and tools.h

The program main.cpp has already been filled out, but feel free to modify it.

Here is the main protcol that main.cpp uses for uWebSocketIO in communicating with the simulator.

INPUT: values provided by the simulator to the c++ program

["sensor_measurement"] => the measurement that the simulator observed (either lidar or radar)

OUTPUT: values provided by the c++ program to the simulator

["estimate_x"] <= kalman filter estimated position x
["estimate_y"] <= kalman filter estimated position y
["rmse_x"]
["rmse_y"]
["rmse_vx"]
["rmse_vy"]

Other Important Dependencies

- cmake >= 3.5
- All OSes: click here for installation instructions
- make >= 4.1 (Linux, Mac), 3.81 (Windows)
- Linux: make is installed by default on most Linux distros
- Mac: install Xcode command line tools to get make
- Windows: Click here for installation instructions
- gcc/g++ >= 5.4
- Linux: gcc / g++ is installed by default on most Linux distros
- Mac: same deal as make - install Xcode command line tools
- Windows: recommend using MinGW

Basic Build Instructions

1. Clone this repo.
2. Make a build directory: `mkdir build && cd build`
3. Compile: `cmake .. && make`
4. On windows, you may need to run: `cmake .. -G "Unix Makefiles" && make`
5. Run it: `./ExtendedKF`

Editor Settings

We've purposefully kept editor configuration files out of this repo in order to keep it as simple and environment agnostic as possible. However, we recommend using the following settings:

- indent using spaces
- set tab width to 2 spaces (keeps the matrices in source code aligned)

Code Style

Try my best to stick to Google's C++ style guide.

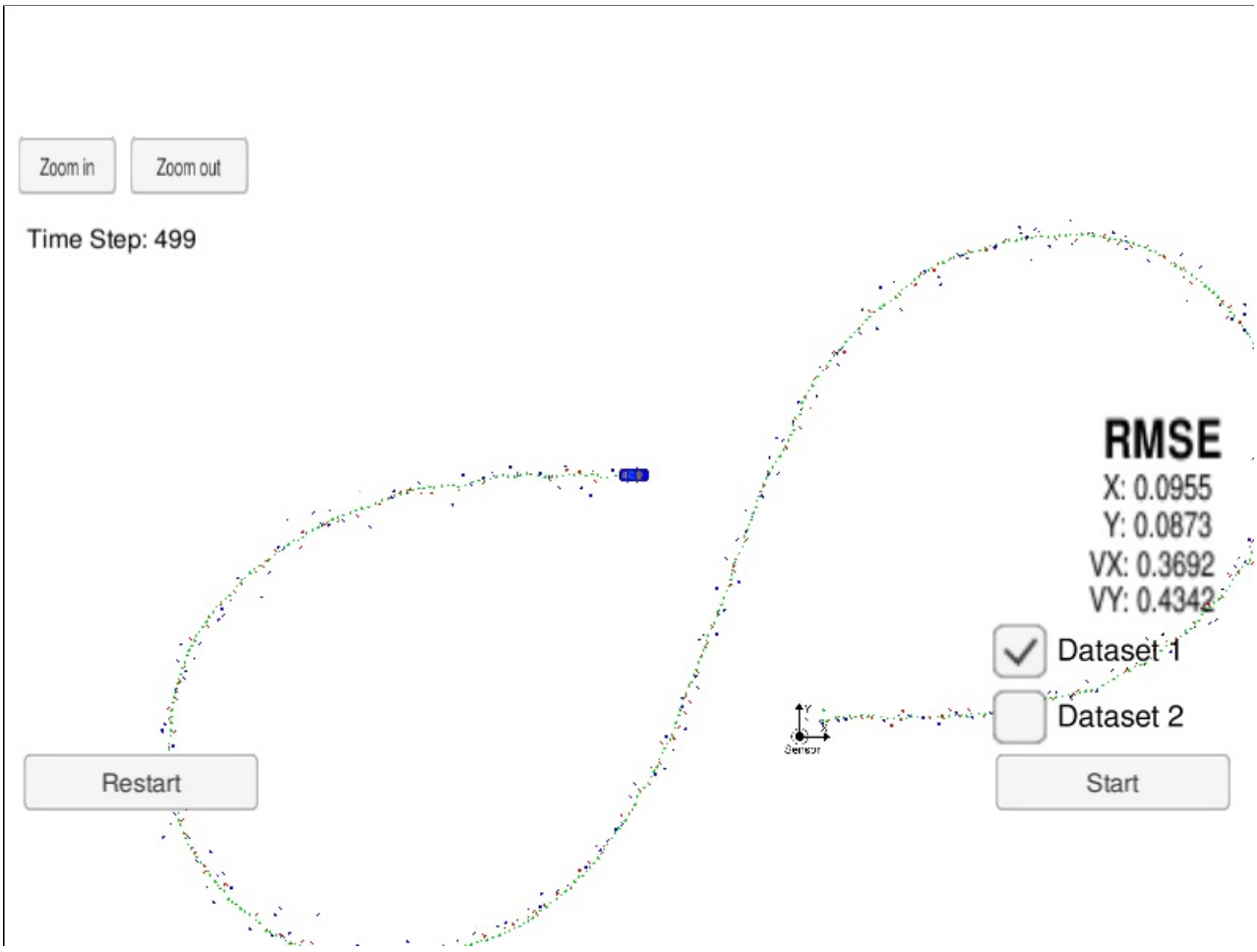
Project Instructions and Rubric

Note: regardless of the changes you make, your project must be buildable using cmake and make!

More information is only accessible by people who are already enrolled in Term 2 of CarND. If you are enrolled, see the project resources page for instructions and the project rubric.

Result

I successfully obtained an RMSE [0.0955, 0.873, 0.3692, 0.4342], which satisfies the requirement RMSE <= [.11, .11, 0.52, 0.52].
And the screenshot is shown below:


Zoom in
Zoom out
Time Step: 499
RMSE
X: 0.0955
Y: 0.0873
VX: 0.3692
VY: 0.4342
Dataset 1
Dataset 2
Restart
Start