Robot Navigation Initial Research

"I pledge my honor that I have abided by the Stevens Honor System".

-Vijayrahul Raja -Ariel Feliz -Xibeizi Ma -Xiang Cao

The objective of this project is to help the robot navigate to a particular location and also identify/ know its own position as time changes. This project is based on analyzing random errors and implementing concepts of Kalman filtering and particle filtering for best approximations.

The Processing software is very suitable for implementing several sections (according to several different operations of the robot) of the code and checking the simulated output in current time. The code will be written completely in Java language.

Particle Filtering (Ariel Feliz and Xibeizi Ma)

This type of filter is an essential tool for tracking and modeling a dynamic system like in this robot navigation project. For example, it could be used to compare and analyze our expected outcome of how the robot navigation changes in time from the given inputs and the expected state the robot at various time intervals. Particle Filtering is also closely related to Kalman filtering but very efficient in solving and bigger and challenging problems. It helps us to our complex model, but an approximate solution could be found unlike Kalman Filtering. The problem of robot localization (location) in known environments/maps could be solved effectively applying this filter as well.

Kalman Filtering (Vijayrahul Raja and Xiang Cao)

Kalman filtering has the main application in the fields of robotic motion planning and control. It can calculate the estimate position and vehicle of the robot. Because of this recursive algorithm, we can run it in real time. All we need are present input measurements, previously calculated states and uncertainty matrix.

Approach:

1: We plan to initially learn about the application of Kalman Filter. Then we will study the five most important formulas, which can be used to get the estimated value, such as the displacement, velocity or the acceleration of the target.

2: Study the two most classic models: constant velocity model(CV) and constant acceleration model(CA).We will simulate the following scenario: the robot moves in a straight line with constant velocity, it can have some noise, using the sensors in in the car, we can get the displacement of the robot, which is of course not very accurate, based on the above conditions, we can use Kalman Filtering to get the accurate values.

3: We have already simulated the above given scenario with Matlab, the further step is to use Processing software and implement the Kalman Filter.

Link to repository on GitHub.com

https://github.com/aafeliz/RobotLocalization

Code part:
Kalman Filter Core Functions:

```
x1=1*x2;
px1=1*px2*1+1.0;
kx=px1*1/(2.0+1*px1*1);
x2=1*x1+kx*(p[0].x-1*x1);
px2=(1-kx*1)*px1;

y1=1*y2;
py1=1*py2*1+1.0;
ky=py1*1/(2.0+1*py1*1);
y2=1*y1+ky*(p[0].y-1*y1);
py2=(1-ky*1)*py1;
```