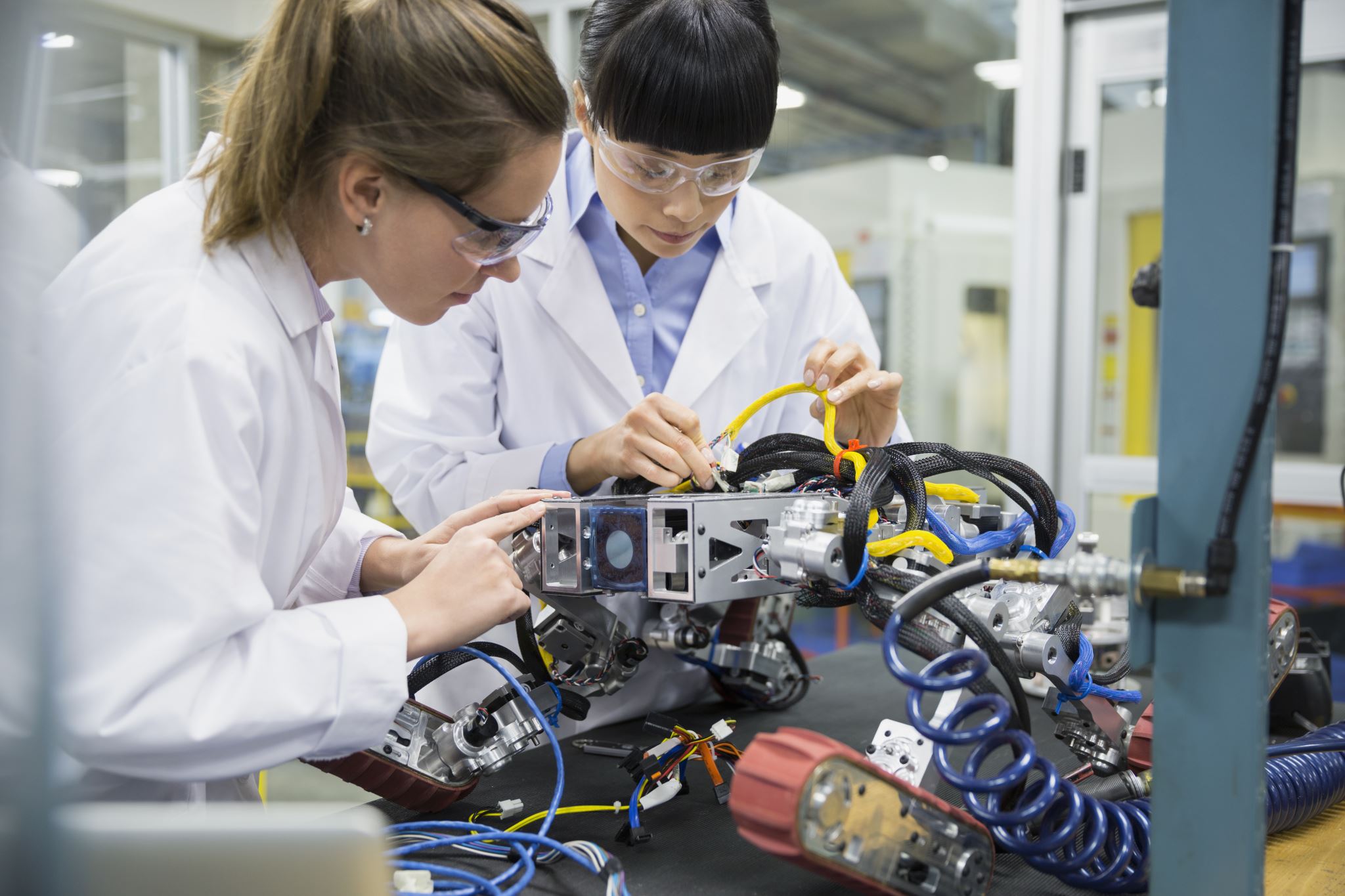
2022

Gabriel Mortensen

Project #1 Report



# Introduction

In this project I was given data collected by a real Novatel DGPS, Mircostrain IMU 3DM-GX2, and Seekur robot encoders. I used this information along with the Kalman Filter provided in class to plot various sets of data. These graphs will be used to answer the questions established by Dr. La in the “Project 1. Mobile Robot Location using Kalman Filter” document.

# Prior Statement | How to Run | Overall Structure of Files (a)

In the document provided by the professor it notes, “Design the Kalman filter to fuse GPS, IMU and Wheel Encoder data to output a smooth and more accurate pose (position and orientation) of the robot and Write Matlab or C/Cpp code (or other languages you prefer) to implement the Kalman filter.”

This project was coded using MatLab and can be executed by typing ‘Project\_1’ into the command prompt. During the run time of the code the user (TA) can select various options that will provide a series of different graphs. The files associated with this code can be in Project\_1\_Code.zip. The following figures further detail how the files are organized and how the code can be executed.

Graphical user interface, application, table

Description automatically generated with medium confidence  
*Fig.a.1: Files within the zip file*

Graphical user interface, application

Description automatically generated   
*Fig.a.2: Command line in promt used to run the code.*   
Graphical user interface, application

Description automatically generated *Fig.a.3: First instance of running the code where user is presented with menu.*

# Why the KF Outperforms the Individual Sensor (GPS, IMU, etc.)

The Kallman Filter outperforms the individual sensor because it utilizes several methods to predict and calibrate the data. Instead of handling raw data, the Kalman Filter employs inputs from state and covariance predictions to form an “educated guess” on what the future output is expected to be. Because the Kalman Filter uses these complex yet efficient calculations, the approximation of the next state is more accurate than the sensors baseline interpretation.

# Results to Prove Kalman Filter Outperforms Individual Sensor (b)

To demonstrate the accuracy of the Kalman Filter compared to the individual sensor this subsection will analyze Fig.b.1.

Chart, line chart

Description automatically generated  
*Fig.b.1: Two graphs are placed next to one another. The first a Fusion of GPS+IMU and Odometry position in terms of x and y, the second is the same but with radian and iteration.*

For the leftmost graph it is evident that the GPS and the Odometry readings have errors. Throughout the graphing of the circular motion the GPS will periodically make sharp divots or have missing gaps and the Odometry readings will sometimes curve too far out. The Kalman filter mediates between these two graphs showing more preference towards GPS data (which has a smaller rate of error on average when compared with the provided excel data sheet) at the same time ignoring the jagged misreading noted by the professor on Lecture 9 Slide 15 (or Fig.b.2).

Diagram

Description automatically generated   
*Fig.b.2.: Although the GPS has an error at the point indicated in the slides the Kalman Filter maintains its approximate position because of its ability to predict and calibrate the given data thus reducing errors.*

For the rightmost graph in Fig.b.1 it may be difficult to tell how the Kalman Filter is more efficient from afar, however if the graph is zoomed in as shown in Fig.b.3, the data presents itself clearly.

Chart, line chart

Description automatically generated  
*Fig.b.3.: Zoomed in version of rightmost graph from Fig.b.1.*

Here it is evident that the IMU data appears jagged while the odometry data indicates turns that are too “sharp”. The Kalman Filter data handles these turns smoothly as demonstrated by its lack of jagged lines. At the bottom left of the graph a turn is made and the GPS/odometry data makes a ridged incline, however the Kalman Filter gradually changes as would be expected of a robot that likely altered its orientation at a standard rate of rotation. In conclusion the Kalman Filter uses previous sets of data and predicts outcomes that can fill holes in potential gaps of data, smooth out jagged lines of data produced by the misreading of the sensor and mediate between two sets of data by applying certain weight to more accurate factors (further established in section d).

# Effects of altering the covariance of the sensor data (c)

For this section I used the variables provided in lecture 9 when initializing the states. I found that when altering covariance for GPS it is evident that the only element that changes is that of the Kalman Filter in the X and Y position graph. There are two reasons for this, the first is that the Kalman Filter is the only element of the three that relies on values specified by the covariance of GPS. More specifically, the R matrix stores covariance of the GPS and is used in the Kalman filter equation as seen in Fig.c.1 and Fig.c.2

Diagram, text

Description automatically generated  
*Fig.c.1:* The Kalman Filter utilizes the R matrix in its calculations.

Text

Description automatically generated with low confidence Text

Description automatically generated with medium confidence  
*Fig.c.2:* The R matrix uses the covariance of GPS in its calculations.

The fact that the R matrix contains data for the covariance of GPS in its calculations explains the behavior of the graph and how the Kalman Filter is the only set of data that changes. The Covariance is used to estimate the error therefore the Kalman filter produces random results when the error is negative because error must be positive or 0 as a set of data can either be perfect (0) or erroneous (greater than 0). The Kalman Filter graph shrinks when the covariance of GPS is positive because the Kalman Filter is compensating for the magnitude in covariance (more detail given in following section). The second reason (the reason the linear position graph changes) is because GPS data only handled linear position, that is, X and Y positions. As such, the radian and iteration graphs are unaltered by changes in GPS covariance.

## Adding noise to GPS covariance to Entire Data Set

When adding noise to covariance I utilized the provided code by Dr. La. In this case, there are two factors to consider in the noise, the standard deviation and the mean. In this situation one can think of the mean as a constant that increases the covariance data set. The higher the magnitude of mean, the higher the magnitude of covariance. A high magnitude of covariance means that the data provided to the Kalman Filter (GPS data) is considered erroneous. The Kalman Filter uses a weighing mechanism when considering covariance thus altering the prediction/result of the output. In this case the data is constructed in a circle. Due to this path an increase in covariance would urge the Kalman Filter to consider the change of the data to be significantly less than the GPS data provided. More specifically, the Kalman Filter relies on the previous data (as it is more skeptical of the current data with the covariance change) thus resulting in a spiral shaped KF since the GPS data is oriented in a circle. Figures Fig.c1.1-1.3 highlight the factor of shift established by the mean.

Chart

Description automatically generated  
*Fig.c.1.1: Output of XY position with std set to 0 and mean set to 1.* Graphical user interface, chart

Description automatically generated  
*Fig.c.1.2: Output of XY position with std set to 0 and mean set to 5.*

Graphical user interface, application

Description automatically generated  
*Fig.c.1.3: Output of XY position with std set to 0 and mean set to 10.*

When modifying the covariance of GPS by any negative value, the Kalman filter would become chaotic as demonstrated in Fig.c.1.3. This is because a value less that is negative defies the logic of erroneous data as a dataset can either be perfect (0) or nonperfect (less than 0) as mentioned in section b.

Graphical user interface, chart

Description automatically generated  
*Fig.c.1.4: Output of XY position with std set to 0 and mean set to -1.*

The next factor is standard deviation. Standard deviation in this case is the random factor added onto the data. In some ways this element could be considered the “actual noise” as it modifies the set with unpredictable values. The higher the magnitude of standard deviation the more unpredictable the Kalman Filter is from the predetermine path shown in Fig.c1.1 – 1.3. Fig.c.1.5-1.7 demonstrate the chaotic factor directly corelated with standard deviation.

Chart

Description automatically generated  
*Fig.c.1.5: Output of XY position with std set to 1 and mean set to 5.*

Graphical user interface, chart

Description automatically generated  
*Fig.c.1.6: Output of XY position with std set to 2 and mean set to 5.*

Graphical user interface, chart

Description automatically generated  
*Fig.c.1.7: Output of XY position with std set to 2.2 and mean set to 5.*

## Adding noise to certain periods of GPS covariance data

When adding GPS covariance manipulation to certain periods of the data the results are very similar to the previous section with two exceptions. The first exception is that when one inserts a negative for mean the chaos happens only within the timestamp specified (Fig.c2.1).

Graphical user interface, chart

Description automatically generated  
*Fig.c.2.1: Output of XY position with std set to 0 and mean set to -1 from time 1 to 1000.*

The second exception is that if the Kalman Filter has a large gap in data after the timestamp specified by the user then a portion of the result will break. If the gap is small enough then the Kalman Filter will not break at that point. Fig.c2.2 demonstrates an example of a broken output due to the large gap while Fig.c2.3 shows a successful time section manipulation of covariance.

Graphical user interface, chart

Description automatically generated  
*Fig.c.2.2: Output of XY position with std set to 1 and mean set to 5 from time 1000 to 2000.*

Graphical user interface, chart

Description automatically generated  
*Fig.c.2.3: Output of XY position with std set to 1 and mean set to 5 from time 1 to 1000.*

# Effects of altering covariance of the sensor data (IMU) (d)

IMU is short for Inertial Measurement Unit and is used to “measure and report specific gravity and angular rate of an object” [1]. Because IMU is focused on angular rates and motion, the linear position graph will be unaffected by the IMU covariance modification as shown in the two following sections. The Kalman Filter Heading appears to be directly correlated to the IMU covariance. When the IMU covariance is modified, the Kalman Filter output changes in the Iteration/Radian graph. The direct correlation of covariance and Kalman Filter makes sense because the Kalman Filter is the only element to use the covariance of IMU in the form of the R matrix for calibrating data.

Text

Description automatically generated

Text, letter

Description automatically generated with medium confidence  
*Fig.d.1:* The R matrix uses the covariance of IMU in its calculations.

Diagram, text

Description automatically generated  
*Fig.d.2:* The Kalman Filter utilizes the R matrix in its calculations.

The Kalman Filter output appears to favor the actual IMU data as shown by the unmodified graph.

Chart, line chart

Description automatically generated

*Fig.d.3:* The unmodified output of the data with focus on the Radiation and Iteration graph.

This is shown in the figure as the Kalman Filter (blue) almost completely overlaps the IMU data (green). The reason for the Kalman Filter’s preference for IMU is because the IMU covariance value is less than the odometry covariance value. The odometry covariance value is a constant of 0.01 as shown in the creation of the R matrix.

Table

Description automatically generated with medium confidence

Text, letter

Description automatically generated with medium confidence  
*Fig.d.4:* The creation of the R matrix uses 0.01 as the constant for odometry covariance.

Meanwhile the IMU covariance is a constant of 0.001225. This may be difficult to determine if one simply observes the code, however examining the data of the given text file in excel it can easily be determined. This excel document is provided with the submission and is titled “DataSheet\_Proj\_1”.

Table

Description automatically generated  
*Fig.d.5:* The data of the given text file makes the covariance of IMU a constant of 0.001225.

Covariance as stated by the professor is the error of the original data. Therefore, because the error of the IMU data is significantly less than that of the odometry data (covariance) the Kalman Filter associates the actual IMU data with being the more accurate value. Results of what happens when one modifies IMU covariance can be observed in the next two sections. It should be noted that for this section I used the variables provided in lecture 9 when initializing the states.

## Adding noise to IMU covariance to Entire Data Set

Similar to the GPS covariance modification there are two values to consider, standard deviation and mean. The mean changes the covariance by a constant while the standard deviation adds randomized data to distort the Kalman Filter from its current path as the weight variables to determine the result becomes unbalanced as shown in Fig.d.1.1. As mentioned in an earlier section the Kalman Filter will break when given a negative value for mean as data can either be perfect (0) or less than perfect (greater than 0).

Chart, histogram

Description automatically generated  
*Fig.d.1.1:* Result of covariance IMU being manipulated by mean -1 and standard deviation 0.

When modifying the IMU by a positive mean the covariance the graph of the Kalman Filter begins to ignore sharp changes in data made by the IMU heading. Fig.d.1.2-4 demonstrate results of steadily increasing the magnitude of the mean during IMU covariance modification.

Chart, line chart

Description automatically generated  
*Fig.d.1.2:* Result of covariance IMU being manipulated by mean 0.5 and standard deviation 0.

Chart, line chart

Description automatically generated

*Fig.d.1.3:* Result of covariance IMU being manipulated by mean 1 and standard deviation 0. Chart, line chart

Description automatically generated  
*Fig.d.1.4:* Result of covariance IMU being manipulated by mean 5 and standard deviation 0.

Depending on the value of the mean the Kalman Filter ignores jagged increases and decreases in data, this is because the Kalman Filter believes these increases and decreases are errors. The more the higher the magnitude the covariance the more skeptical the Kalman Filter becomes concerning changes in data. From this characteristic it can be determined that the higher the magnitude in covariance, the more linear the Kalman Filter output becomes as shown by the previous graphs. The standard deviation is similar in nature to the GPS covariance in that the higher the standard deviation the more chaotic the Kalman Filter becomes on its assigned path. The reason for this chaotic behavior is because by adding random values along the dataset different weight attributes are assigned to the Kalman Filter which distorts the interpretation and prediction of data. Fig.d.1.5-7 demonstrate the effect of increasing standard deviation.

Chart, line chart

Description automatically generated  
*Fig.d.1.5:* Result of covariance IMU being manipulated by mean 5 and standard deviation 1.2.

Chart, line chart

Description automatically generated  
*Fig.d.1.6:* Result of covariance IMU being manipulated by mean 5 and standard deviation 1.65.

Chart, line chart

Description automatically generated  
*Fig.d.1.7:* Result of covariance IMU being manipulated by mean 5 and standard deviation 2.

## Adding noise to certain periods of IMU covariance data

When manipulating sections of the IMU covariance two occurrences happen. If the mean is negative, then the Kalman Filter output will break as per usual in the time section indicated. If the result is positive, then the Kalman Filter will ignore the sharp edges of the IMU header in the time stamp. During my experiments I have found that there is no breakage in the Kalman Filter unlike time sections of GPS covariance manipulation as the IMU covariance is graphed in such a way that it is very difficult to construct large gaps.

Chart

Description automatically generated  
*Fig.d.2.1:* Result of covariance IMU being manipulated by std 0 and mean of -1 from timestamp 1 to 1000.

Chart, line chart

Description automatically generated

*Fig.d.2.2:* Result of covariance IMU being manipulated by mean 5 and std 3 from timestamp 1 to 1000.

# Effects of adding noise to GPS position with changed covariance (e)

For this section I will be utilizing the same variables the professor used in his Coding Instructions document which can be found on canvas in the project 1 files. For this section I altered Covariance of GPS and the data of GPS as requested by the project description. Because of the different initiation variables, it should be noted that changes in covariance concerning standard deviation and mean act differently than the previous sections in that the effect of the two variables are scaled down significantly as demonstrated in Fig.e.1 and Fig.e.2.

Graphical user interface, chart

Description automatically generated  
*Fig.e.1: Results of XY position graph with std 0 and mean* ***100*** *with Coding Instruction variables.*

Graphical user interface, chart

Description automatically generated  
*Fig.e.2: Results of XY position graph with std 0 and mean* ***5*** *with Lecture 9 variables.*

# Adding noise to GPS position to Entire Data Set

Because the noise added to the GPS data is the same as the noise added to the covariance of GPS the Kalman Filter can adjust itself accordingly. This means that even with the changes in noise the Kalman Filter can maintain a somewhat accurate result. An example can be observed in Fig.e.1.1 and Fig.e.1.2.

Graphical user interface, chart

Description automatically generated  
*Fig.e.1.1: Results of XY position graph with std 0.1 and mean* 0.5 *with Coding Instruction variables.*

Chart

Description automatically generated

*Fig.e.1.2: Zoomed in version of Fig.e.1.1 towards a point of GPS error.*

By observing Fig.e.1.2 it is evident that even though an error occurs with GPS data the Kalman filter can mediate between the GPS data difference. Despite this fact there is a factor of unpredictability presented by the data. As standard deviation increases in magnitude the Kalman Filter becomes more uncertain of the data presented to it. This is especially true considering how just a slight change in values can result in significant results when using variables associated with the Instruction Coding document as shown by Fig.e.1.3

Chart

Description automatically generated  
*Fig.e.1.3: Same approximation location as Fig.e.1.2 except std is 0.15.*

Despite this, it can be determined that by having the same noise added to covariance the Kalman Filter outputs the best result as can be possible for various changes in data. Like every prediction there is a factor of uncertainty, however it is significantly less than what it would be without noise considerations.

## Adding noise to certain periods of GPS position data

When doing only a section it is evident that the results are nearly the exact same (as is expected when compared to previous sections). This means that if a large gap of data occurs right after the time section indicated by the user, then the Kalman Filter will temporarily break down. It also means that only the time section specified by the user will be affected. Some examples can be observed in Fig.e.2.1 through Fig.e.2.3.

Graphical user interface, chart

Description automatically generated  
*Fig.e.2.1: Mean 0.5 and Std 0.15 during t = 1 and t = 1000.*

Graphical user interface, chart

Description automatically generated *Fig.e.2.2: Mean 0.5 and Std 0.19 during t = 2000 and t = 3000.*

Graphical user interface

Description automatically generated  
*Fig.e.2.3: Mean 0.5 and Std 0.175 during t = 1500 and t = 2500.*

# Conclusion

This report focused on analyzing data from every segment mentioned in the instructions.pdf. By accessing the resulting graphs at certain sections, one was able to better understand the functionality of the Kalman Filter and determine the meaning of important variables. Overall, I would personally assess this project as a successful analysis of the provided data as it helped me better understand the functionality of the code provided in Lecture 9 as well as the amazing attributes associated with the Kalman Filter.

# Coding Portion

During lecture it was announced that we should attached our project code to the report of the project, as such, the code can be found below:

%Gabriel Mortensen

%Feb. 2022

%CPE 470

%Project 1

function Project\_1()

%Initialization for clearing all input

clc, close all

clear

%Obtain data via reading .txt

[time, data] = rtpload('EKF\_DATA\_circle.txt');

%Calculate total length of data for future reference (A.K.A total time)

total= 1:length(data.O\_x);

%Declare variables

reply = 0; %obtains user input

new\_reply = 0; %also obtains user input

min\_input = -1; %user input should they desire a period of noise or measuing covariation

max\_input = -1; %user input should they desire a period of noise or measuing covariation

extra\_covariance\_query = -1;%In case user wants to do both GPS and GPS Covariance portion manipulation (not full data)

min\_input\_extra\_GPS\_co = -1; %extra range if user wants to do both GPS and GPS Covariance portion manipulation (not full data)

max\_input\_extra\_GPS\_co = -1; %extra range if user wants to do both GPS and GPS Covariance portion manipulation (not full data)

%Obtain User Input

while (reply < 1 || reply > 4 )

disp('!!!Gabriel Mortensen Project 1 CPE 470!!!')

disp('~~~~~~~~~~~~~~~~~~~~~~~~~~~~~')

disp('What would you like to do?')

disp(' 1: Perform Kalman filter and output a smooth and more accurate pose (position and orientation) of the robot from EKF\_DATA\_circle.txt.')

disp(' 2: Change the covariance of the sensor data (GPS) and check the output of the KF, then plot.')

disp(' 3: Change the covariance of the sensor data (IMU) and check the output of the KF, then plot.')

disp(' 4: Add noise to GPS position with changed covariance, then plot.')

reply = input('');

disp('~~~~~~~~~~~~~~~~~~~~~~~~~~~~~')

end

%If user wants to do anything other than normal KF, get more input

if (reply > 1 && reply < 5)

while(new\_reply < 1 || new\_reply > 2)

disp('~~~~~~~~~~~~~~~~~~~~~~~~~~~~~')

disp('Where would you like to add the noise?')

disp(' 1. Add noise to the entire data set')

disp(' 2. Add noise to certain periods of data')

new\_reply = input('');

disp('~~~~~~~~~~~~~~~~~~~~~~~~~~~~~')

end

%Obtain the standard deviation from the user

disp('~~~~~~~~~~~~~~~~~~~~~~~~~~~~~')

disp('What Standard Deviation for the noise?');

standard\_deviation = input('');

%Obtain the mean from the user

disp('What is the Mean Value for the noise?');

mean\_data = input('');

disp('~~~~~~~~~~~~~~~~~~~~~~~~~~~~~')

%Generate two sets of random data (GPS x random and y)

random\_values\_pos = standard\_deviation.\* randn([length(total), 2]) + mean\_data.\* ones([length(total), 2]);

if(new\_reply == 2)

disp('~~~~~~~~~~~~~~~~~~~~~~~~~~~~~')

fprintf('Range of "time" for data goes from 1 to %d.\n', length(total))

while (min\_input < 1 || min\_input > length(total)-1)

fprintf('Input minimum range for data set (must be number from 1 to %d) \n', length(total)-1)

min\_input = input('');

end

while (max\_input > length(total) || max\_input < min\_input || max\_input < 1 )

fprintf('Input maximum range for data set (must be <= %d and >= min value) \n', length(total))

max\_input = input('');

end

disp('~~~~~~~~~~~~~~~~~~~~~~~~~~~~~')

end

end

%Get Odometry IMU and GPS data (x, y, theta)

Odom\_x = data.O\_x;

Odom\_y = data.O\_y;

Odom\_theta = data.O\_t;

%Obtain GPS position

Gps\_x = data.G\_x;

Gps\_y = data.G\_y;

%If user does not choose 4 then keep orginally stored data

if (reply ~= 4)

disp('Normal storing for GPS data.')

end

%If case 4 and full set requested, Add noise to GPS position

if (reply == 4 && new\_reply == 1)

disp('Adding random noise to all x and y GPS data.');

Gps\_x = data.G\_x + random\_values\_pos(:, 1);

Gps\_y = data.G\_y + random\_values\_pos(:, 2);

end

%If case 4 and portion then use for loop to assign random noise to portion

if (reply == 4 && new\_reply == 2)

fprintf('Assigning random noise to range %d to %d for position of GPS. \n', min\_input, max\_input);

for index=min\_input:max\_input

Gps\_x(index) = data.G\_x(index) + random\_values\_pos(index, 1);

Gps\_y(index) = data.G\_y(index) + random\_values\_pos(index, 2);

end

end

%Get covariance data for GPS

Gps\_Co\_x = data.Co\_gps\_x;

Gps\_Co\_y = data.Co\_gps\_y;

%Keep stored data if user choose 1 or 3 case

if (reply == 1 || reply == 3)

disp('Normal storing for GPS Covariance data.')

end

%If case 2 full data set or 4 (either), modify covariance by random noise

if ((reply == 2 && new\_reply == 1) || (reply == 4 && new\_reply == 1))

fprintf('Adding random noise to all Covariance of GPS\n');

Gps\_Co\_x = data.Co\_gps\_x + random\_values\_pos(:, 1);

Gps\_Co\_y = data.Co\_gps\_y + random\_values\_pos(:, 2);

end

%If case 2 add noise to covariance if porition requested

if ((reply == 2 && new\_reply == 2) || (reply == 4 && new\_reply == 2))

fprintf('Assigning random noise to range %d to %d for Covariance of GPS. \n', min\_input, max\_input);

for index=min\_input:max\_input

Gps\_Co\_x(index) = data.Co\_gps\_x(index) + random\_values\_pos(index, 1);

Gps\_Co\_y(index) = data.Co\_gps\_y(index) + random\_values\_pos(index, 2);

end

end

%Store IMU data

IMU\_heading = data.I\_t;

%Store IMU covarience

IMU\_Co\_heading = data.Co\_I\_t;

%If case 3 not chosen, keep stored data

if(reply ~= 3)

disp('Normal storing for IMU Covariance data.')

end

%If case 3 full porition then add noise to all of data

if (reply == 3 && new\_reply == 1)

fprintf('Adding random noise to all Covariance of IMU \n');

IMU\_Co\_heading = data.Co\_I\_t + random\_values\_pos(:, 1);

end

if(reply == 3 && new\_reply == 2)

fprintf('Assigning random noise to range %d to %d for Covariance of IMU. \n', min\_input, max\_input);

for index=min\_input:max\_input

IMU\_Co\_heading(index) = data.Co\_I\_t(index) + random\_values\_pos(index, 1);

end

end

% Calibrate IMU to match with the robot's heading initially

IMU\_heading = IMU\_heading +(0.32981-0.237156)\*ones(length(IMU\_heading),1);

%For EKF\_DATA3

%Obtain user input on particular variables

variable\_input = 0;

while(variable\_input < 1 || variable\_input > 2)

disp('~~~~~~~~~~~~~~~~~~~~~~~~~~~~~')

disp('Which P Q R and V would you like to use?')

disp(' 1. Use the Coding Insturction Docx Variables')

disp(' 2. Use the Lecture 9 Vairbles')

variable\_input = input('');

end

disp('~~~~~~~~~~~~~~~~~~~~~~~~~~~~~')

%\*\*\*\*\*\*\*\*INITIALIZE STATES & OTHER INFORMATION\*\*\*\*\*\*\*\*\*\*\*%

%Depending on user input store different attributes

if(variable\_input == 1)

%Velocity of the robot

V = 0.14;%0.083;

%Distance between 2 wheel

L = 1; %meter

%Angular Velocity

Omega = V\*tan(Odom\_theta(1))/L;

%set time\_step

delta\_t = 0.001; %0.001

s.x = [Odom\_x(1); Odom\_y(1); V; Odom\_theta(1); Omega]; %Enter State (1x5)

%Enter transistion matrix A (5x5)

s.A = [1 0 delta\_t\*cos(Odom\_theta(1)) 0 0;

0 1 delta\_t\*sin(Odom\_theta(1)) 0 0;

0 0 1 0 0;

0 0 0 1 delta\_t;

0 0 0 0 1];

%Define a process noise (stdev) of state: (Student can play with this number)

%Enter covariance matrix Q (5x5) for state x

s.Q = [.0004 0 0 0 0; %For EKF\_DATA\_circle

0 .0004 0 0 0;

0 0 .001 0 0;

0 0 0 .001 0;

0 0 0 0 .001];

% s.Q = [.000000004 0 0 0 0; %For EKF\_DATA\_Rutgers\_ParkingLot

% 0 .000000004 0 0 0;

% 0 0 .001 0 0;

% 0 0 0 .001 0;

% 0 0 0 0 .001];

%Define the measurement matricx H:

%Enter measurement matrix H (5x5) for measurement model z

s.H = [ 1 0 0 0 0;

0 1 0 0 0;

0 0 1 0 0;

0 0 0 1 0;

0 0 0 0 1];

%Define a measurement error (stdev)

%Enter covariance matrix R (5x5) for measurement model z

s.R = [.04 0 0 0 0;

0 .04 0 0 0;

0 0 .01 0 0;

0 0 0 0.01 0;

0 0 0 0 .01];

%B matrix initialization:

s.B = [ 1 0 0 0 0;

0 1 0 0 0;

0 0 1 0 0;

0 0 0 1 0;

0 0 0 0 1];

%Enter initial value of u (5x5)

s.u = [0; 0; 0; 0; 0];

%Enter initial covariance matrix P (5x5)

s.P = [.001 0 0 0 0;

0 .001 0 0 0;

0 0 .001 0 0;

0 0 0 .001 0;

0 0 0 0 .001];

end

if(variable\_input == 2)

V = 0.44;%Velocity of the robot

L = 1; %Distance between 2 wheel

Omega = V\*tan(Odom\_theta(1))/L; %Angular Velocity

delta\_t = 0.001; %set time\_step

s.x = [Odom\_x(1); Odom\_y(1); V; Odom\_theta(1); Omega]; %Enter State (1x5)

s.Q = [.00001 0 0 0 0; %you can play with this matrix to see what happens

0 .00001 0 0 0;

0 0 .001 0 0;

0 0 0 .001 0;

0 0 0 0 .001]; %Enter covariance matrix Q (5x5) for state x

s.H = [ 1 0 0 0 0;

0 1 0 0 0;

0 0 1 0 0;

0 0 0 1 0;

0 0 0 0 1]; %Enter measurement matrix H (5x5) for measurement model z

s.R = [.1 0 0 0 0;

0 .1 0 0 0;

0 0 .01 0 0;

0 0 0 0.01 0;

0 0 0 0 .01]; %Enter covariance matrix R (5x5) for measurement model z

s.B = [ 1 0 0 0 0;

0 1 0 0 0;

0 0 1 0 0;

0 0 0 1 0;

0 0 0 0 1];

s.u = [0; 0; 0; 0; 0];

s.P = [.01 0 0 0 0;

0 .01 0 0 0;

0 0 .01 0 0;

0 0 0 .01 0;

0 0 0 0 .01]; %Enter initial covariance matrix P (5x5)

end

%Begin Analyzing

for t=1:length(total)

%Enter transistion matrix A (5x5)

s(t).A = [1 0 delta\_t\*cos(Odom\_theta(t)) 0 0;

0 1 delta\_t\*sin(Odom\_theta(t)) 0 0;

0 0 1 0 0;

0 0 0 1 delta\_t;

0 0 0 0 1];

%Enter covariance matrix R (5x5) for measurement model z

s(t).R = [Gps\_Co\_x(t) 0 0 0 0;

0 Gps\_Co\_y(t) 0 0 0;

0 0 .01 0 0;

0 0 0 IMU\_Co\_heading(t) 0;

0 0 0 0 .01];

%Enter measurement vector

s(t).z = [Gps\_x(t); Gps\_y(t); V; IMU\_heading(t); Omega];

% perform a Kalman filter iteration (you need to implement it)

s(t+1)=Kalman\_Filter(s(t));

%Store Kalman filter x and y into their variables for plotting

%Note that X matrix has [x, y, velocity, theta (orientation), accelration]

%Therefore x(1, :) is x and x(2, :) is y

x\_position\_kalman(t) = s(t).x(1,:);

y\_position\_kalman(t) = s(t).x(2,:);

t\_theta\_orientation\_kalman(t) = s(t).x(4, :);

end

%Obtain user input concerning graphs

graph\_input = 0;

while (graph\_input < 1 || graph\_input > 4)

disp('~~~~~~~~~~~~~~~~~~~~~~~~~~~~~')

disp('Which graphs would you like to see:')

disp(' 1. Position X Y ')

disp(' 2. Position Radian Iteration ')

disp(' 3. Both ')

graph\_input = input('');

disp('~~~~~~~~~~~~~~~~~~~~~~~~~~~~~')

end

if (graph\_input == 1 || graph\_input == 3)

%Plot the position of the robot using various sets of data

if(graph\_input == 3)

subplot(1,2,1);

end

hold on

plot(Odom\_x, Odom\_y, '.r')

plot(Gps\_x, Gps\_y, '.g')

plot(x\_position\_kalman, y\_position\_kalman, ".b" )

%plot(Odom\_x, Odom\_y, 'r.', Gps\_x, Gps\_y, 'r.', x\_position\_kalman, y\_position\_kalman, 'r.')

grid on

title('Fusion of GPS+IMU and ODOMETRY in Position')

xlabel('X(m)')

ylabel('Y(m)')

xlim([-15 10])

ylim([-5 20])

legend('Odometry','GPS Calibrated', 'Kalman Output')

end

if (graph\_input == 2 || graph\_input == 3)

%Plot the angular position of the robot using GPS+IMU

if(graph\_input == 3)

subplot(1,2,2);

end

plot(time, Odom\_theta, "red", time, IMU\_heading, "green", time, t\_theta\_orientation\_kalman, "blue", LineWidth=1.1)

grid on

title('Fusion of GPS+IMU and ODOMETRY in Position')

xlabel('Iteration')

ylabel('Radian')

xlim([0 4000])

ylim([-4 4])

legend('Odometry Heading','IMU Heading', 'Kalman Filter Heading')

end

end

function [s] = Kalman\_Filter(s)

%Prediction Phase

%prediction of state X

X\_priori = s.A \* s.x + s.B \* s.u;

%prediction of error covariance P

P\_priori = s.A \* s.P \* transpose(s.A) + s.Q;

%Kalman Gain Calculation

Kalman\_Gain = P\_priori \* transpose(s.H) \* (s.H \* P\_priori \* transpose(s.H) + s.R)^(-1);

%Calibration Phase

%Calibration of state X

s.x = X\_priori + Kalman\_Gain \* (s.z - (s.H \* X\_priori));

%Calibration of covariance P

s.P = (eye(5) - Kalman\_Gain \* s.H) \* P\_priori;

%Overall steps based on L8, slide 11, CPE 470

end

# Resources

1. <https://www.vectornav.com/resources/inertial-navigation-articles/what-is-an-inertial-measurement-unit-imu>