Engineering Assignment Coversheet

Please note that you:

- Must keep a full copy of your submission for this assignment
- Must staple this assignment
- Must NOT use binders or plastic folders except for large assignments

Student Number(s)
671116
Group Code (if applicable):

Assignment Title:	Project 2: Height estimation using numerical methods & Kalman filter
Subject Number:	MCEN 90032
Subject Name:	Sensor Systems
Student Name:	Akshay Kumar Bharat Kuma
Lecturer/Tutor:	Mitchell Khoo
Due Date:	23/9/2018

For Late Assignments Only

Has an extension been granted? Yes / No (circle)

A per-day late penalty may apply if you submit this assignment after the due date/extension. Please check with your Department/coordinator for further information.

Plagiarism

Plagiarism is the act of representing as one's own original work the creative works of another, without appropriate acknowledgment of the author or source.

Collusion

Collusion is the presentation by a student of an assignment as his or her own which is in fact the result in whole or in part of unauthorised collaboration with another person or persons. Collusion involves the cooperation of two or more students in plagiarism or other forms of academic misconduct.

Both collusion and plagiarism can occur in group work. For examples of plagiarism, collusion and academic misconduct in group work please see the University's policy on Academic Honesty and Plagiarism: http://academichonesty.unimelb.edu.au/

Plagiarism and collusion constitute cheating. Disciplinary action will be taken against students who engage in plagiarism and collusion as outlined in University policy. Proven involvement in plagiarism or collusion may be recorded on my academic file in accordance with Statute 13.1.18.

STUDENT DECLARATION

Please sign below to indicate that you understand the following statements:

I declare that:

- This assignment is my own original work, except where I have appropriately cited the original source.
- This assignment has not previously been submitted for assessment in this or any other subject.

For the purposes of assessment, I give the assessor of this assignment the permission to:

- Reproduce this assignment and provide a copy to another member of staff; and
- Take steps to authenticate the assignment, including communicating a copy of this assignment to a checking service (which may retain a copy of the assignment on its database for future plagiarism checking).

Student signature ..... Date 23/9/2018.

MCEN90032 Sensor Systems 2018:

Project 2: Height estimation using numerical methods and Kalman filter

Project Completed by:

Akshay Kumar Bharat Kumar, 671116

Due by:

23rd September 2018

A.) INTRODUCTION:

Sensors are used in wide number of different applications, such as in rocket control systems, aeroplanes and more. In the real world, these sensors are not free of noises or disturbances, and if uncontrolled, can result in significant errors in the feedback received by the end user. This is especially important if the sensor fluctuates continuously without reaching some sort of stable system. The purpose of this project was for students to under how the BMP280 pressure sensor and the ADXL335 accelerometer can measure their respective heights and accelerations but fail to accurately capture the height as it is varied. Moreover, the project aimed to teach students how to implement a Kalman filter, which would allow for a relatively accurate means of integrating both sensors mentioned above, in addition, to improving the height tracking ability of this filter.

As such, the project was divided into 3 major parts:

1.) Estimate velocity and distance from acceleration and pressure measurements directly
    a. This involved collecting the data from lifting the pressure sensor and accelerometer from and to a specific height, while collecting the readings at prespecified time steps.
    b. The pressure sensor readings were differentiated twice to obtain both the velocity and the acceleration. Similarly, the accelerometer readings were integrated twice to obtain both the velocity and the height.
2.) Vertical velocity estimation and distance estimation using the Kalman filter
    a. The first part of achieving this was to determine the discrete time state space model of the combined system for both inputs of the accelerometer and pressure sensor heights. This model should include noise and disturbances to account for real world scenarios.
    b. Following this, the Kalman filter should be applied to the system, using the readings collected from the previous part. In doing this, the Q and R matrices should be specified and determined by the student.
3.) Vertical distance estimation using Kalman Filter in real time
    a. Following the Kalman filter design and test, the Arduino was to be programmed to perform a live and real-time Kalman filter implementation, as the system was raised and lowered at varying speeds.

The image adjacent is the circuit board diagram to demonstrate the setup of the pressure sensor and accelerometer for Project 2. With these steps, the following report the procedures, calculations and results of the Kalman filter implementation.

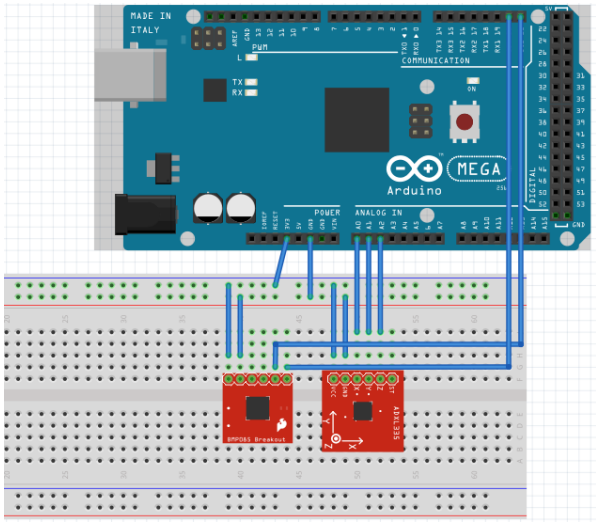


Figure 1: The breadboard diagram of the Arduino, accelerometer and pressure sensor system and how the system has been connected.

B.) PART 1: ESTIMATE VELOCITY AND DISTANCE FROM ACCELERATION AND PRESSURE MEASUREMENTS DIRECTLY

With Part 1 of the project, the aim was to collect raw data and measurements from the accelerometer and pressure sensor at the same prespecified time step as the system is raised and lowered.

The first step was to interface with the pressure sensor to begin collection of pressure data. This pressure data would subsequently be used to calculate the altitude relative to the local sea level pressure which would could identified via the BOM. The interfacing process was outlined via the link provided. (Satujamsaja, 2016)

The heights are very dependent on the surrounding air pressure. As such, to determine the actual height from the sea level, the actual sea level pressure needs to be entered into the Arduino on a semi-regular basis. This air pressure can be obtained from the Bureau of Meteorology. (Government, Australian, 2018)

This height measurement which is calculated via the BMP280 pressure sensor uses the following model: $h = 44300(1 - \left(\frac{p}{p_0}\right)^{0.19})$ (Genovese, 2014)

Where p is the current pressure being measured, and the p_0 is the sea level pressure. Using the information is not very useful as these heights would change significantly at different locations where the pressure sensor is placed. As such, an initial baseline would be necessary so that a relative height could be established. This was done by taking the initial measurement from pressure sensor as a baseline and subsequently subtracting all future readings from this baseline. This would give a relative height change allowing for more accurate altitude readings.

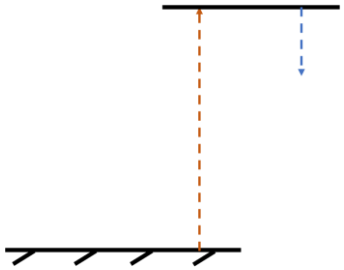


Figure 2: Visual representation of the pathway taken for the data set used to implement the Kalman filter in Part 1 and 2

The system was raised and lowered as per the image adjacent at a fairly consistent speed.

Subsequently, the results were collected into an Excel spreadsheet, “sensors_2_v3”, attached with the submission. To compare the accuracy of the sensors in obtaining relevant representations of positions, velocity and accelerations, the readings of the accelerometer were directly integrated (area under an acceleration vs time graph gives velocity, etc.), and the readings of the pressure sensor were differentiated (height/ time gives velocity, etc.)

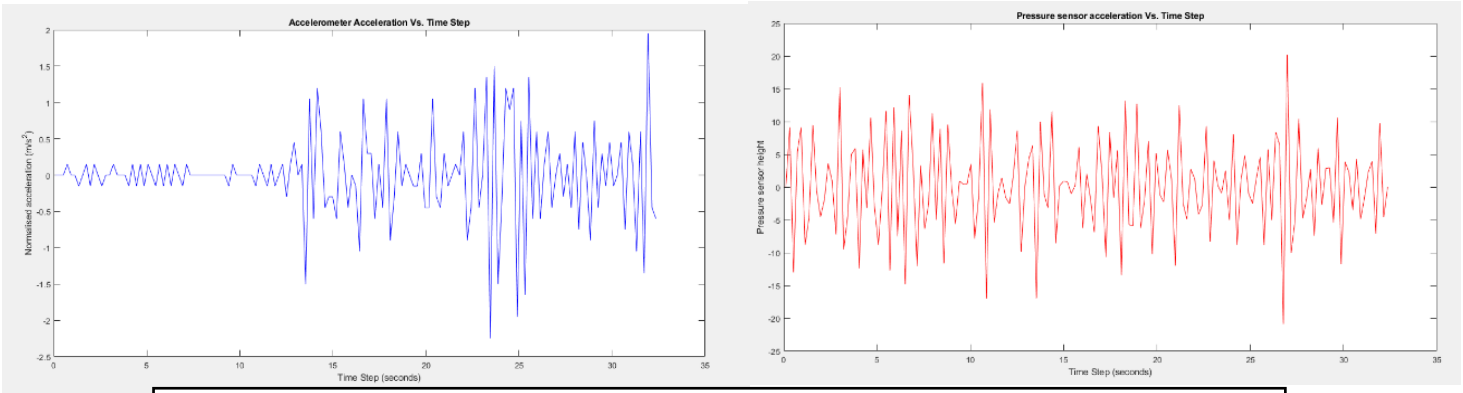


Figure 3: Accelerometer Raw Acceleration and Pressure sensor derivated acceleration

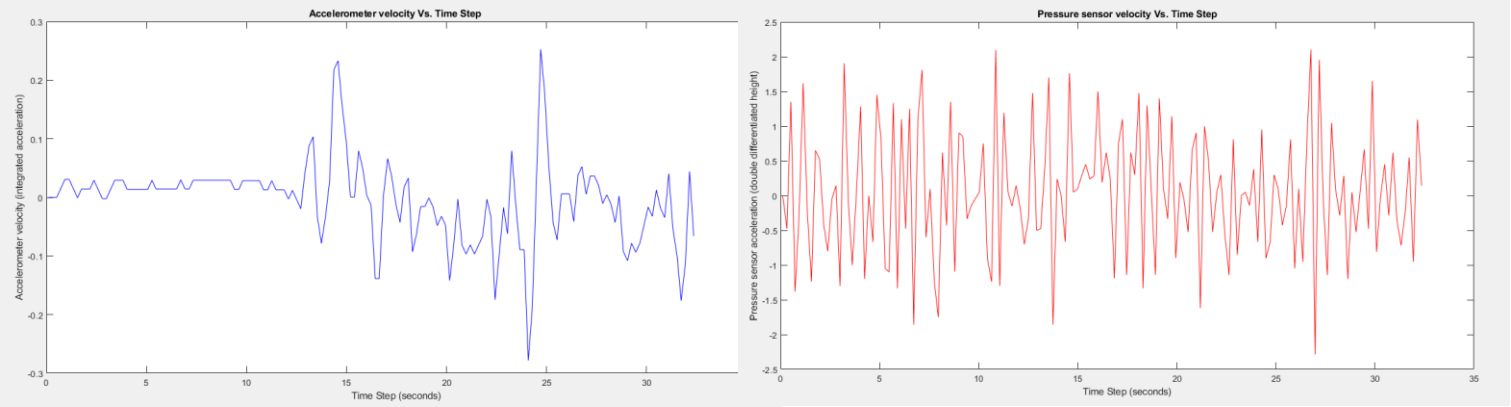


Figure 4: Accelerometer integrated velocity and Pressure sensor derivated velocity

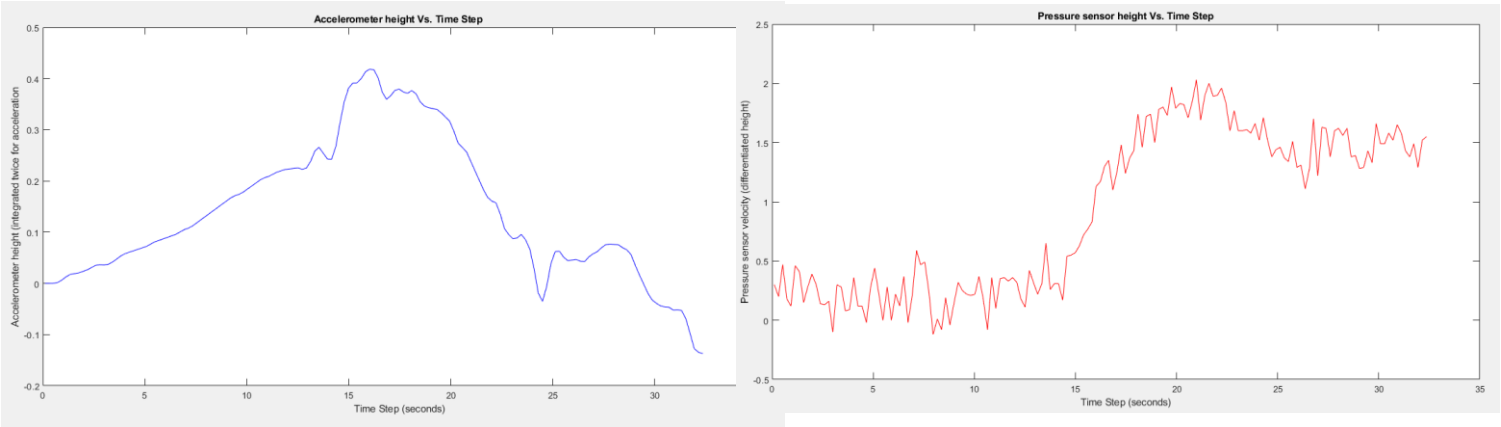


Figure 5: Accelerometer integrated height and Pressure sensor Raw height

From the images above, it can be seen that as the accelerometer data is integrated to reach position/ displacement, the curves begin to smooth. Conversely, as the pressure sensor heights are differentiated, the curves become noisier. However, the ability to accurately track the actual distance moved is lost in the integration process for the accelerometer. Differentiating the pressure sensor data produces no sensible data and very noisy data and so is quite useless. The pressure sensor however is quite able to reasonably track the distances measured and taken, albeit, very noisy and hairy data being produced due to the subtle fluctuations in pressure which directly affect the altitude measurements. As such, the data is not quite ideal and not robust if quick and sudden movements were made. A better system would need to be introduced, which would be the Kalman filter.

C.) PART 2: VERTICAL VELOCITY AND DISTANCE ESTIMATION USING KALMAN FILTER

From the raw data above, the velocity and distance collected and subsequently calculated are quite variable and noisy. Using the raw data from the previous section, a post processing of this data to implement the Kalman filter would be necessary. This was completed using the MATLAB script

attached with this submission: “sensors_2_graphing.m”. However, before this could be reached, a state space model for the accelerometer, the pressure sensor and subsequently, the discretised time state space model was needed. This is shown in the hand calculations below.

i) The continuous-time state space for the accelerometer.

Let x_1 = a (acceleration) $\Rightarrow \dot{x}_2 = x_1$, which is input to system.
x_2 = v (velocity) $\Rightarrow x_2 = \dot{x}_3$
x_3 = p (position). $\Rightarrow$ -

$$\therefore \begin{bmatrix} \dot{x}_3 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x_3 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} a. \quad -(1)$$

where a is the acceleration input into the system.

The system not completely state observable as the system is not full ranked. This is because only velocity & acceleration are captured.

ii) Including the pressure sensor, the continuous time-state space is.

$$X_4 = [1 \quad 0]\, p \quad -(2)$$

This extra term allows for the system to be fully observable as it is now a full rank.

By having both equations (1) & (2), the entire output of the system can be captured. i.e. Acceleration, velocity & position can all be captured by the model.

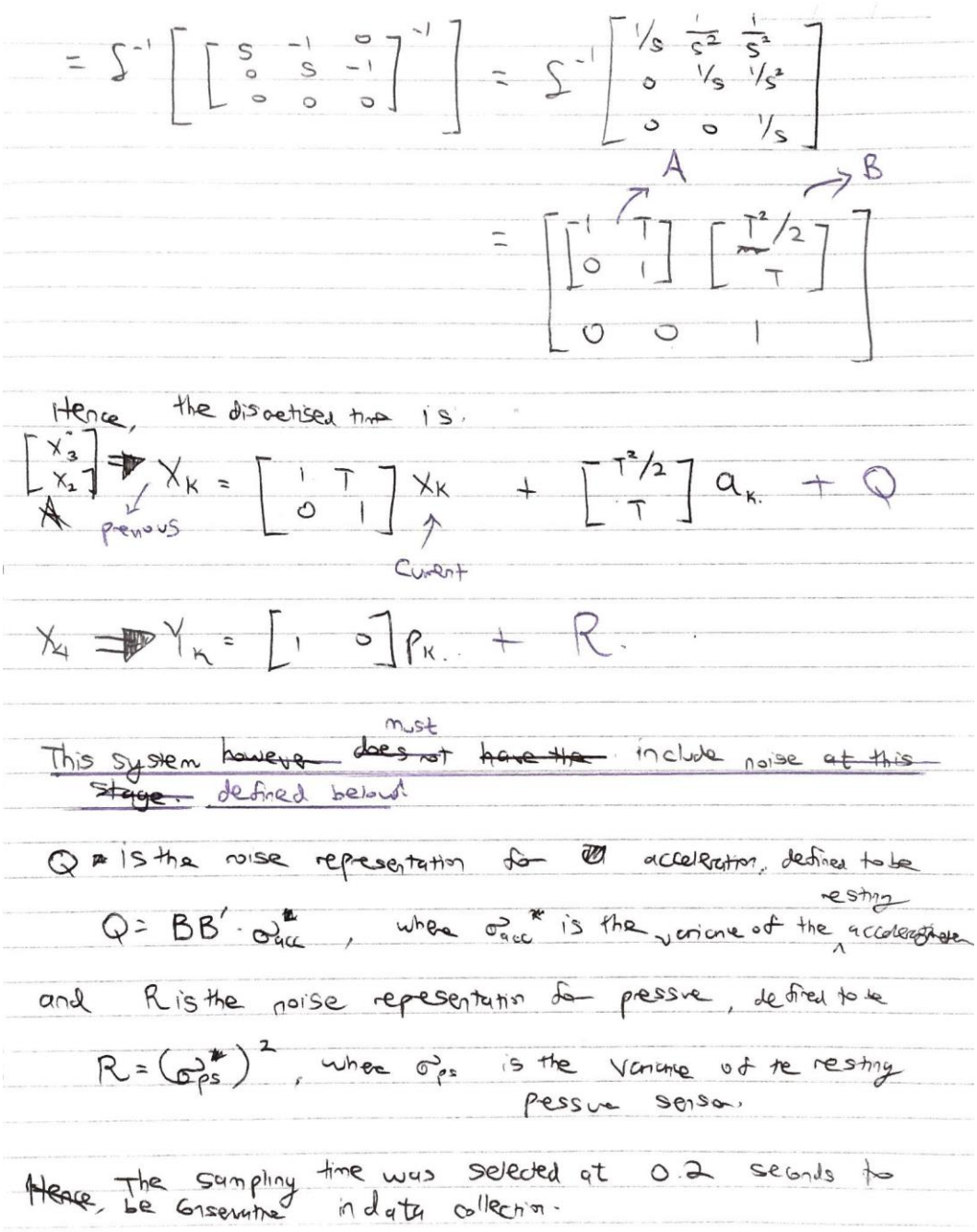
iii) To obtain the discrete-time state space model, a sample time of T, is used.

From (1), we are able to form an exponential representation as:

$$X_k = e^{\begin{bmatrix} A & B \\ 0 & 0 \end{bmatrix} T}.$$

To discretise, we must inverse laplace the system

$$\mathcal{L}^{-1}\left[(sI - \begin{bmatrix} A & B \\ 0 & 0 \end{bmatrix})^{-1}\right] = \begin{bmatrix} A_k & B_k \\ 0 & I \end{bmatrix}$$



iv.) One assumption of the standard Kalman filter is that the initial condition is of zero-mean. This is only satisfied if the starting state of both the position and velocity of the state space model as outlined above are held at rest and are set at an initial baseline of 0 (i.e. stationary and placed at a predefine starting position). The decision of the P matrix (error covariance) would only be affected if the position and velocity of the matrix was known to be exact and precise and devoid of noise.

The error covariance is set to be 1 to account for noise, which is very prevalent in sensors and no sensor is devoid of noise. As such, even if the initial position and velocity were not zero, the P matrix should still be set at one to account for the noise in systems.

v.) The other assumptions of the Kalman filter other than the 2 above are that the update stages should be Gaussian and have white noise (SALZMANN, 1988). The one issue with this is that if the errors in the readings are not Gaussian, the Kalman filter may fail identify it. In the case of white noise, most of the noise being faced by the sensors are of white noise (i.e. the signal has the power in throughout the bandwidth of the signal). If those were not met, i.e. the noise was blue or pink, the Kalman filter may fail to succeed.

vi.) The Kalman Filter was designed based off what was taught in lectures (Palaniswami, 2018)The entire process is summarised in this snip taken from the lectures below.

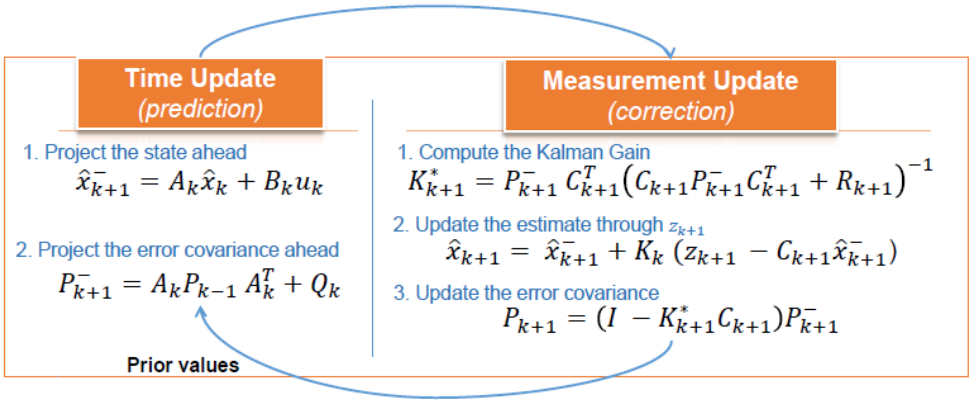


Figure 6: Kalman filter algorithm as per the lectures

Matrix A and B were established earlier. The initial P matrix was set to be a 2 by 2 matrix of 1s to account for noise. U_k is the input from the accelerometer, the z_{k+1} is the input from the pressure sensor. X is the state space positions and velocity at each time step. C is the covariance matrix which was selected to be [1, 0]. K is the Kalman Gain which will help with the correction process to ensure to correct for the noise being produced.

The Q and R matrices, which represent the covariance of the process noise and observation noise (acceleration and pressure sensor noise respectively), were selected based off research completed by the student (Shimkin, 2009). The article cited shows that the Q and R matrix is best defined according to the following description:

$$Q = BB^T\sigma_{acc}^2$$

$$R = \sigma_{press}^2$$

Where B is the B matrix outlined above, B^T is the transpose of the B matrix, σ_{acc} is the variance of idling accelerometer, σ_{press} is the variance of the idling pressure sensor. The variances were calculated by just taking the measurements that are collected from the 2 sensors as they are resting on flat surface and taking the variance of the values. This was subsequently used in the Q and R matrix calculations. However, upon implementing the Kalman filter, it was found to follow the pressure sensor very closely. As a result, a modification to the Q matrix made, with the final Q matrix being in the form of:

$$Q = 0.05BB^T\sigma_{acc}^2$$

vii.) Implementing the Kalman filter produced the following plots, one for velocity comparisons, one for height comparisons and one for a goodness of fit of the Kalman filter to the acceleration and height changes from the accelerometer and pressure sensor respectively.

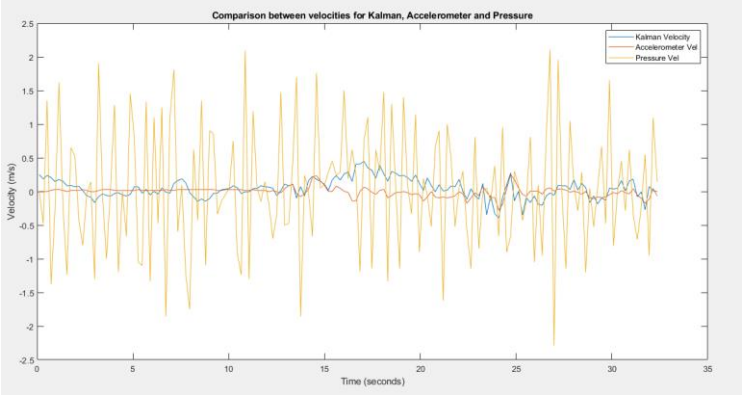


Figure 7: Comparisons for velocities for Kalman, Accelerometer and Pressure sensor

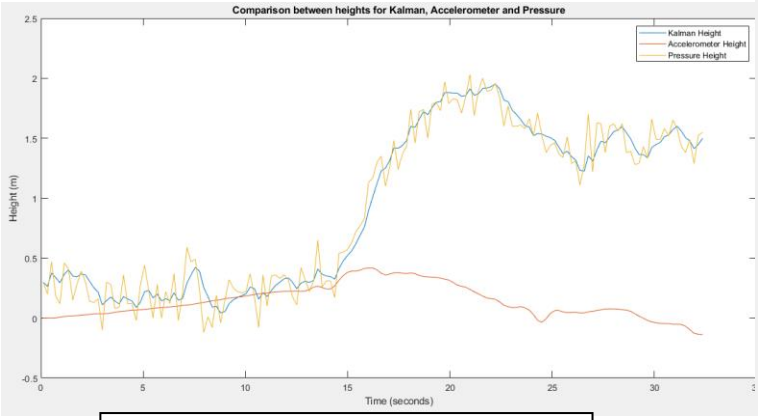


Figure 8: Comparisons for height for Kalman, Accelerometer and Pressure sensor

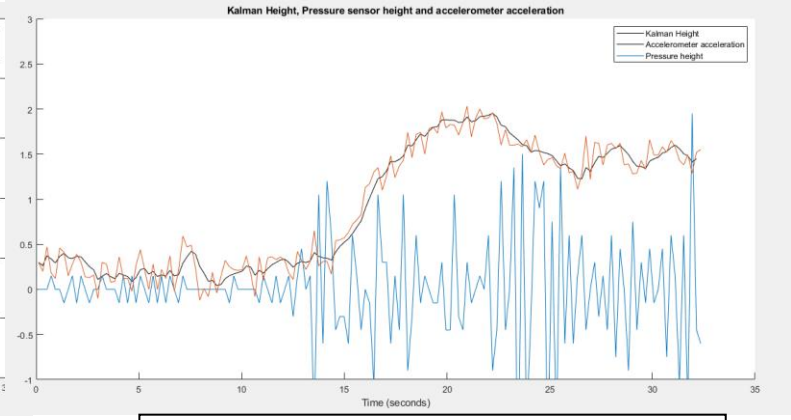


Figure 9: Goodness of fit for Kalman filter to changes in acceleration and height

Figure 7 and 8 capture the velocity and height distribution to a more accurate state. In Figure 7, the Kalman filter shows minimal velocity which is quite similar to what was truly happening whilst lifting and lowering the system. Similarly, Figure 8 is able to track the height fairly smoothly without the noisy fluctuations in the data. Figure 9 shows the Kalman filter and the 2 inputs which influence the state space model of the filter, that is the acceleration from the accelerometer and the height measured by the pressure sensor. As can be seen, it is quite able to track height changes despite some noisy accelerations which were picked up.

D.) PART 3: VERTICAL VELOCITY AND DISTANCE ESTIMATION USING KALMAN FILTER

Following from Part 2, the final objective of the project was to implement the Kalman filter into Arduino IDE. In doing so, there would be live updating and correction as the Arduino system was lifted up and down at varying speeds. The code used to implement this has been attached in this submission under the file name “Kalman_Flter_671116.ino”.

As per the images below, the Kalman filter is well balanced as it is able to accurately keep track of the changes in both acceleration and height of the system and is able to maintain a relatively stable system despite the numerous fluctuations.

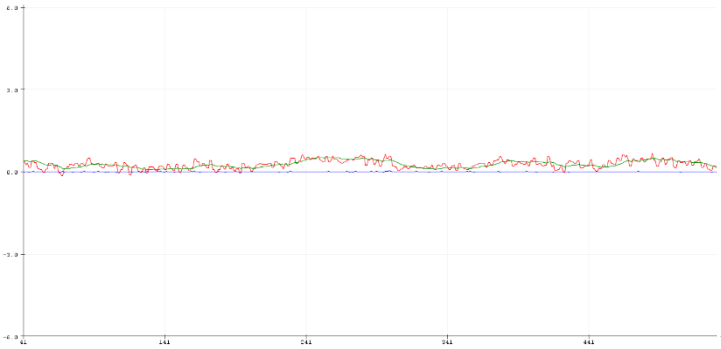


Figure 10: Serial plotter of accelerometer (blue), pressure sensor(red) and the Kalman filter (green) whilst at rest

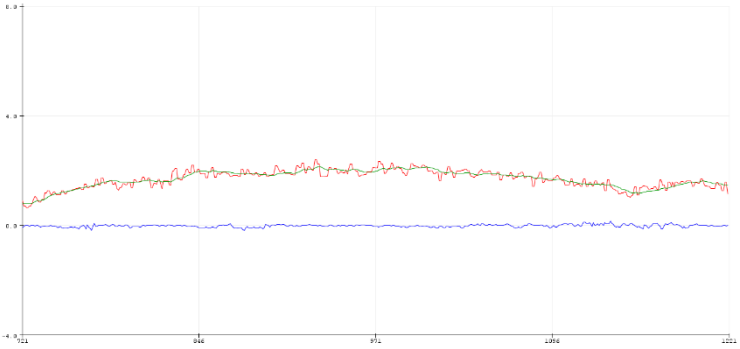


Figure 11: Serial plotter of Arduino system when lifted to a height of 2 metres.

Figure 10 showed the system as it was just placed on the ground. In Figure 11, the Arduino was raised up to a height of 2 metres, which was checked using a tape measure. The Arduino is quite capable of tracking this height as indicated in Figure 11. Figure 12 was the major test of the Kalman filters robust ability to changes in both height and rapid accelerations/ decelerations. As observed, the Kalman filter is able to track these changes quite precisely and accurately.

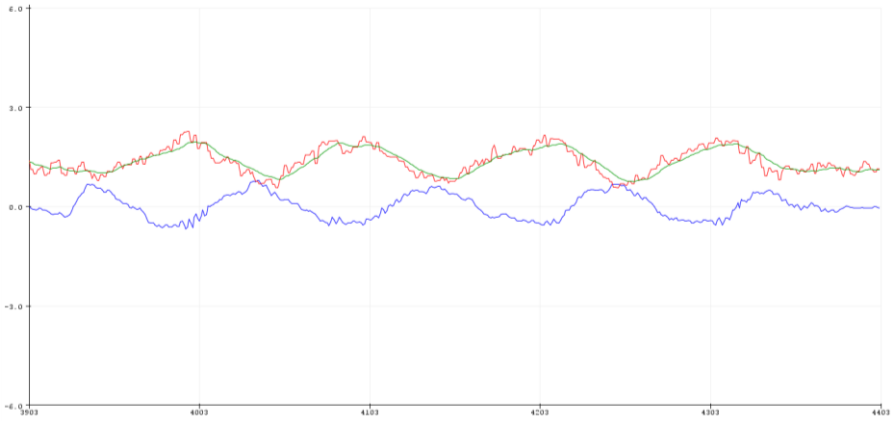


Figure 12: Serial plotter of Arduino system as the system is rapidly raised and lowered in quick succession

E.) CONCLUSION:

From the project and tests conducted, an in-depth understanding as to how the Kalman filter functions was gained, its ability to accurately track changes set to it according to the state space model accelerations. As observed, the Kalman filter is able to track these changes quite precisely and accurately. Moreover, it is able to use both the altitude inputs and the accelerometer inputs to accurately track the distances travelled, whilst removing the noisy input data that is being fed into the system. As such, the project has given the student a greater appreciation for the Kalman filter and how it can be used effectively to account for noisy data and using it to integrate the information from more than 1 sensor.

F.) REFERENCES:

Bibliography

Genovese, A. M. (2014). A Sensor Fusion Method for Tracking Vertical Velocity and Height Based on Inertial and Barometric Altimeter Measurements. *sensors*, 24.

Government, Australian. (2018, September 22). *Bureau of Meteorology*. Retrieved from http://www.bom.gov.au/products/IDV60801/IDV60801.95936.shtml

Palaniswami, P. M. (2018). *Kalman Filter - Lecture 16.* Retrieved from Learning Management System Unimelb: https://app.lms.unimelb.edu.au/bbcswebdav/pid-6577734-dt-content-rid-46834849_2/xid-46834849_2

SALZMANN, M. A. (1988). SOME ASPECTS OF KALMAN FILTERING. *GEODESY AND GEOMATICS ENGINEERING*, 65.

Satujamsaja. (2016, September 18). *Satujamsaja*. Retrieved from http://satujamsaja.blogspot.com.au/2016/09/read-temperature-and-barometric.html

Shimkin, P. N. (2009). Estimation and Identi¯cation in Dynamical Systems (048825). *Technion { Israel Institute of Technology, Department of Electrical Engineering*, 3.