FILTERING OF IMU DATA USING KALMAN FILTER

A Project

Presented to the faculty of the Department of Electrical and Electronic Engineering

California State University, Sacramento

Submitted in partial satisfaction of
the requirements for the degree of

MASTER OF SCIENCE

in

Electrical and Electronic Engineering

by

Naveen Prabu Palanisamy

FALL
2016

FILTERING OF IMU DATA USING KALMAN FILTER

A Project

by

Naveen Prabu Palanisamy

Approved by:

____________________________________, Committee Chair
Dr. Fethi Belkhouche

____________________________________, Second Reader
Dr. Preetham Kumar

____
Date

Student: Naveen Prabu Palanisamy

I certify that this student has met the requirements for format contained in the University format manual, and that this project is suitable for shelving in the Library and credit is to be awarded for the project.

___________________________, Graduate Coordinator ____________________
Dr. Preetham Kumar Date

Department of Electrical Engineering

Abstract

of

FILTERING OF IMU DATA USING KALMAN FILTER

by

Naveen Prabu Palanisamy

Inertial Measurement Unit (IMU) is a component of the Inertial Navigation System (INS), a navigation device used to calculate the position, velocity and orientation of a moving object without external references. This project develops a method for removing the bias from the accelerometer measurement and estimate the distance travelled and the velocity of a moving object. Estimation is done using the predict and update stages of the Kalman filter, a recursive filter that uses state space techniques. Simulation of the distance and velocity estimation is performed. Multiple accelerometer data with different sampling frequencies are used for analyzing the bias factors.

________________________, Committee Chair
Dr. Fethi Belkhouche

____
Date

ACKNOWLEDGEMENTS

I would like to thank my professor and my guide for this project Dr. Fethi Belkhouche for his support and guidance towards the project. I would like to thank him for always being there to help me understanding the topic and developing the project and also encouraging me to give my best.

Also I would like to thank my Graduate Coordinator, Dr. Preetham Kumar for his consistent support and guidance in my academics, from enrolling for classes until submission of my project. I would also like to thank all my professors throughout my Master's and Bachelor's, for giving me an opportunity to learn and grow up to this level.

Lastly, I would like to express my heartfelt gratitude to my family for supporting me during this project. Also to my friends and roommates, thank you for listening, offering me advice, and supporting me through this entire process. This journey would not have been possible without the support of my family, professors and mentors, and my friends.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

INTRODUCTION

A modern navigation system is an integrated collection of position and orientation sensors, computing and communication hardware and software used to facilitate the movement of people, vehicles, and other moving objects [1]. There are different types of navigation systems like global positioning system (GPS) and inertial navigation system (INS). This project is concerned with the inertial navigation system (INS).

Inertial navigation is an independent navigation technique in which measurements provided by accelerometers and gyroscopes are used to track the position and orientation of an object relative to a known starting point, orientation and velocity [2]. Inertial measurement unit (IMU) is the main component of INS. IMUs consists of miniature Microelectromechanical systems (MEMS) with accelerometers and gyroscopes. They offer a self-contained and inexpensive means for position and orientation estimation, from which a navigation solution is delivered with every new measurement.

Unfortunately, due to the nature and fabrication of the IMU units, their measurements are noisy and have varying DC offsets. As a result of the noise, the unstable bias and inaccurate compensation for the gravity component, the position estimates derived from IMUs become inaccurate and unusable after a short period of time.

The primary sources of errors in inertial sensors are bias, scale factor, and random noise [3].

Bias errors – The simplest type of error. A bias is a constant signal added to the output of a sensor, it is independent of the input. Gyroscopes and accelerometers inside an IMU exhibit some finite measured output, even under the absence of rotation or acceleration and those measurements are called “bias error”.

Scale factor errors – Scale factor or sensitivity is the ratio between the measured output and the change in the input. When compared with bias errors, scale factor errors are not a large contributor to the total error but still need to be corrected. Scale factor effects are most apparent at high acceleration and rotation.

Random noise – Whenever a sensor is used to measure a signal, a random noise (error) is always present in the measurement. Noise is the unwanted signal generated from the internal electronics that interferes with the desired signal measurement. Many error sources are random and can only be described in terms of stochastic processes. A stochastic process is a random time-process.

Some errors in inertial sensors measurement are deterministic and therefore can be determined through specific lab – calibration procedures. The remaining errors are not systematic and they need to be modeled stochastically [3].

1.1 INTRODUCTION TO KALMAN FILTER

In 1960, R.E. Kalman published his famous paper describing a recursive solution to the discrete data linear filtering problem [4]. Since that time, due to advances in digital computing, the Kalman filter has been the subject of extensive research and application, particularly in the area of autonomous or assisted navigation.

A Kalman filter is simply an optimal recursive data processing algorithm. It processes all available measurements, regardless of their precision, to estimate the current value of the variables of interest with use of

- Knowledge of the system and measurement device dynamics.
- The statistical description of the system noise, measurement errors, and uncertainty in the dynamic models.
- Any available information about the initial conditions of the variables of interest.

A system is driven by some known controls, the measuring devices provide the value of certain pertinent quantities. These quantities represent the data available from the physical system for estimation purposes.

Any measurement will be corrupted to some degree by noise, biases, and device inaccuracies. There may also be a number of different measuring devices, each with its own particular dynamics and error characteristics that provide some information about a particular variable, and it would be desirable to combine their outputs is a systematic and optimal manner. A Kalman filter combines all available measurement data, plus prior knowledge about the system and measuring devices, to produce an estimate of the desired variables in such a manner that the error is minimized statistically.

The goal of the project is to filter the noisy acceleration data from the Inertial Measurement Unit using a set of mathematical equations and estimate the state of process in a way that minimizes the errors. This could predict the exact position, velocity and distance of the desired object using the information from the sensors. In this project the Kalman filter is designed in MATLAB which is an interactive software product in which a large number of technical and mathematical procedures are available in a functional format.

This project is organized as follows. In Chapter 2, the basic fundamentals of IMU, its components and working principles as well as the error characteristics are discussed. Chapter 3, describes the Kalman filter, its operations and significance in estimating the position and velocity using the accelerometer data. The Extended Kalman filter is also explained.

In Chapter 4, the implementation of Kalman filter, bias estimation and calibration, estimation of gravity from IMU, estimation of distance and velocity of an object and error analysis are discussed from simulation results and plots. Finally, in Chapter 5, the project is summarized and potential improvements and suggestions are included. The appendix lists the code written in this project to implement the Kalman filter and achieve successful estimation of the distance and velocity of an object.

Chapter 2

INERTIAL MEASURMENT UNIT

An Inertial measurement unit (IMU) is a sensory system used to determine the kinematic variables of the motion of a rigid body based on the inertial effects due to the motion. Inertial sensors such as accelerometers (ACCs) and gyroscopes (gyros) are the core of Inertial Measurement Units utilized in navigation systems. Such sensors are widely used for estimating the position, speed, and altitude of aerial, marine and terrestrial vehicles. The grade of the IMU can vary based on the implemented sensors. Some technologies are expensive, and thus in cost-effective applications, MEMS (Micro-Electro-Mechanical System) technology is preferred. Advances in MEMS have widened the range of possible applications to include areas such as human and animal motion capture [2].

2.1 INERTIAL SYSTEM CONFIGURATIONS

Nearly all IMUs fall into one of the two categories outlined below. The difference between the two categories is the frame of reference in which the rate – gyroscopes and accelerometers operate.

2.1.1 STABLE PLATFORM SYSTEMS

In stable platform systems, the inertial sensors are mounted on a platform which is separated from any external rotational motion [2]. This is achieved by mounting the platform using gimbals (frames), which allow the platform to rotate in all three axes. The platform mounted gyroscopes detect any platform rotations. These signals are fed back to torque motors which rotate the gimbals in order to cancel out such rotations, the platform is aligned with the global frame. To track the orientation of the device, the angles between adjacent gimbals can be read using angle pick-offs. To calculate the position of the device, signals from the platform mounted accelerometers are double integrated. Note that it is necessary to subtract the acceleration due to gravity from the vertical channel before performing the integration.

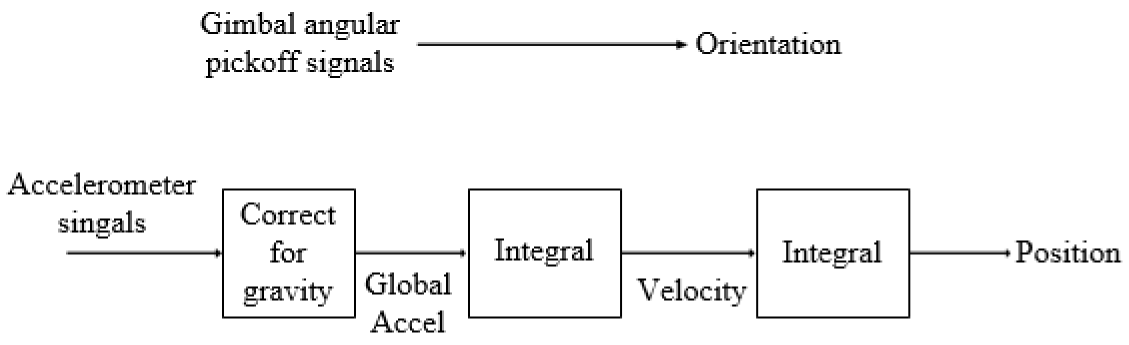

Figure 1 – Stable platform inertial navigation algorithm [2]

2.1.2 STRAPDOWN SYSTEMS

In strapdown systems, the inertial sensors are mounted rigidly onto the device, and therefore the output quantities are measured in the body frame rather than the global frame [5]. To keep track of the orientation, the signals from the rate gyroscopes are ‘integrated’. To track the position, three accelerometer signals are resolved into global coordinates using the known orientation, as determined by the integration of the gyro signals. The global acceleration signals are then integrated as in the stable platform algorithm. This procedure is shown in Figure 2. Stable platform and strapdown systems are both based on the same underlying principles. Strapdown systems have a reduced mechanical complexity and tend to be physically smaller than stable platform systems. These benefits are achieved at the cost of an increased computational complexity. As the cost of computation decreases, strapdown systems are becoming the dominant type of IMU systems.

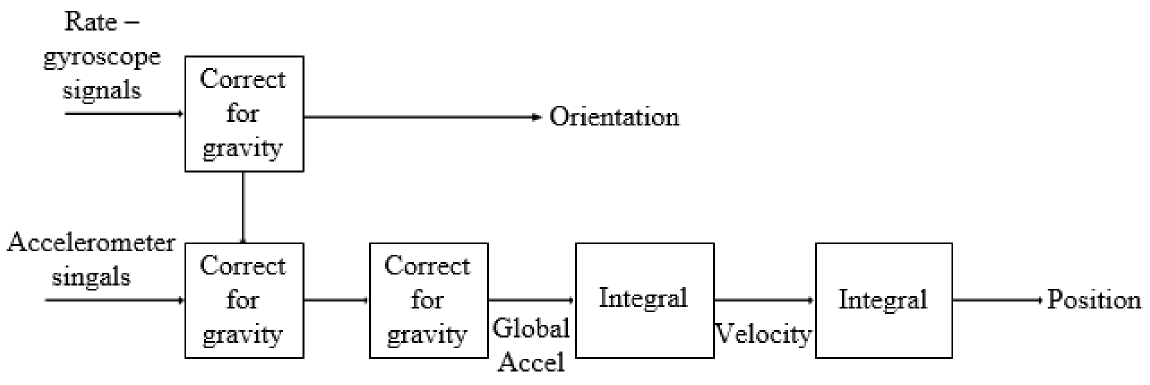

Figure 2– Strapdown inertial navigation algorithm [2]

2.2 GYROSCOPES

A gyroscope is a device that uses earth’s gravity to determine the orientation [5]. In this section different types of gyroscopes are presented.

2.2.1 MECHANICAL GYROSCOPES

A conventional gyroscope consists of a spinning wheel on two gimbals. When a mechanical gyroscope is rotated the angles between adjacent gimbals will change but the wheel will remain at a constant global orientation and the. Angles between the adjacent gimbals can be read using angle pick-offs, which measures the orientation. A conventional gyroscope measures orientation. In contrast, nearly all modern gyroscopes (including the optical and MEMS types) are rate-gyros, which measure the angular velocity.

The main disadvantage of mechanical gyroscopes is that they contain moving parts (like gimbal, axle and wheel). These moving parts led to friction, which causes the output to drift over time. To minimize friction, high-precision bearings and special lubricants are used. Mechanical gyroscopes also require a few minutes to warm up.

2.2.2 OPTICAL GYROSCOPES

A fiber optic gyroscope (FOG) uses the interference of light to measure the angular velocity. A FOG consists of a large coil of optical fiber. To measure rotation, two light beams are sent into the coil in opposite directions. If the sensor is undergoing a form of rotation then the beam travelling in the direction of rotation will experience a longer path to the other end of the fiber than the beam travelling against the rotation. This is known as the Sagnac effect [2]. When the beams exit the fiber they are combined. The phase shift introduced due to the Sagnac effect causes the beams to interfere, resulting in a combined beam whose intensity depends on the angular velocity. It is therefore possible to measure the angular velocity by measuring the intensity of the combined beam.

Ring laser gyroscopes (RLGs) are also based on the Sagnac effect [5]. The difference between fiber optic and ring laser gyroscopes is that in ring laser gyroscopes, laser beams are directed around a closed path using mirrors rather than optical fiber.

Optical gyroscopes contain no moving parts and require only a few seconds to start-up. The accuracy of an optical gyro is largely dependent on the length of the light transmission path (larger is better), which is constrained by the size of the device.

2.2.3 MEMS GYROSCOPES

After years of development, mechanical and optical gyroscopes still have high part counts and require high-precision tolerance and complex assembly techniques and also they remain expensive. In contrast MEMS sensors built using silicon micro-machining techniques have low part counts (a MEMS gyroscope can consist of as few as three parts) and are relatively cheap to manufacture. MEMS gyroscopes make use of the Coriolis Effect, which states that in a frame of reference rotating at angular velocity ω, a mass m moving with velocity v experiences a force [5]:

$$F_c = -2m(\omega v)$$

Some of the advantages of MEMS sensors when compared with mechanical and optical sensors include small size, light weight, low power consumption, short start-up time, low cost, high reliability and low maintenance.

2.3 MEMS GYRO ERROR CHARACTERISTICS

2.3.1 CONSTANT BIAS

In this section we examine the errors which arise in MEMS gyroscopes, and their effect on the integrated (orientation) signal. A constant bias error $\in$, when integrated, causes an angular error which grows linearly with time $\theta\ (t) = \in t$.

The constant bias error of a rate gyroscope can be estimated by taking a long term average of the gyroscope's output while it is not undergoing any rotation. Once the bias is known it is important to subtract it from the output.

2.3.2 BIAS STABILITY / FLICKER NOISE

The bias of a MEMS gyroscope changes over time due to flicker noise in the electronics and other components susceptible to random flickering. Flicker noise is noise with a 1/f spectrum, the effects of which are usually observed at low frequencies in electronic components. At high frequencies, flicker noise tends to be overshadowed by white noise [2]. Bias fluctuations which arise due to flicker noise are usually modeled as a random walk. A bias stability measurement describes how the bias of a device may change over a specified period of time.

2.3.3 TEMPERATURE EFFECTS

Temperature fluctuations due to changes in the environment and sensor self-heating induce movement in the bias. Note that such movements are not included in bias stability measurements which are taken under fixed conditions.

Any residual bias introduced due to a change in temperature will cause an error in the orientation which grows linearly with time. The relationship between bias and temperature is often highly nonlinear for MEMS sensors. Most inertial measurement units (IMUs) contain internal temperature sensors which make it possible to correct for temperature induced bias effects [6].

2.3.4 CALIBRATION ERRORS

The term ‘calibration errors’ refers collectively to errors in the scale factors, alignments, and linearity of the gyroscopes. Such errors tend to produce bias errors that are only observed while the device is turning [3]. They lead to the accumulation of additional drift in the integrated signal, the magnitude of which is proportional to the rate and duration of motion. It is usually possible to measure and correct calibration errors.

2.4 ACCELEROMETER

An accelerometer is a compact device designed to measure acceleration. When the object goes from a standstill to any velocity, the accelerometer responds to the vibrations associated with such movement. It uses microscopic crystals that go under tension when vibrations occur, and from that tension a voltage is generated to create a reading of the acceleration. Accelerometers is a device used to track paths and other measurements in the quantified self-movement. Accelerometers can be broadly classified as mechanical, solid state, and MEMS accelerometers.

2.4.1 MECHANICAL ACCELEROMETER

A mechanical accelerometer consists of a mass suspended by springs. The movement of the mass is measured using a displacement pick-off, giving a signal that is proportional to the force F acting on the mass in the direction of the input axis.

Newton's second law $F = ma$ is then used to calculate the acceleration acting on the device.

2.4.2 SOLID STATE ACCELEROMETER

An example of a solid-state accelerometer is the Surface Acoustic Wave (SAW) accelerometer. A SAW accelerometer consists of a cantilever beam which is vibrated at a particular frequency. A mass is attached to one end of the beam which is free to move. The other end is rigidly attached to the case. When an acceleration is applied along the input axis, the beam bends. This causes the frequency of the surface acoustic wave to change proportionally to the applied strain. By measuring the change in frequency, the acceleration can be determined.

2.4.3 MEMS ACCELEROMETER

Micro-machined silicon accelerometers use the same principles as mechanical and solid state sensors. There are two main types of MEMS accelerometers:

- The first type consists of mechanical accelerometers (devices which measure the displacement of a supported mass) manufactured using MEMS techniques.
- The second type consists of devices which measure the change in frequency of a vibrating element caused by a change of tension, as in SAW accelerometers.

The advantages of MEMS devices apply equally to accelerometers as they do to gyroscopes. They are small, light and have low power consumption and short start-up times [7]. Their main disadvantage is that they are not currently as accurate as accelerometers manufactured using traditional techniques, although the performance of MEMS devices is improving rapidly.

2.5 MEMS ACCELEROMETER ERROR CHARACTERISTICS

In this section we examine the errors which arise in MEMS accelerometers. The important difference between errors arising from accelerometers is that they are integrated twice in order to track position, whereas rate-gyro signals are only integrated once to track orientation.

2.5.1 CONSTANT BIAS

The bias of an accelerometer is the offset of its output signal from the true value, in m/s^2. A constant bias error of $\in$, when double integrated, causes an error in position which grows quadratically with time. The accumulated error in position is

$$S(t) = \in \frac{t^2}{2}$$

where t is the time of integration.

It is possible to estimate the bias by measuring the long term average of the accelerometer's output when it is not undergoing any acceleration. Unfortunately this is complicated by gravity, since a component of gravity acting on the accelerometer will appear as a bias. It is therefore necessary to know the accurate orientation of the device with respect to the gravitational field in order to measure the bias. In practice this can be achieved by calibration routines in which the device is mounted on a turntable whose orientation can be controlled extremely.

2.5.2 FLICKER NOISE / BIAS STABILITY

MEMS accelerometers are subject to flicker noise, which causes the bias to increase over time. Such fluctuations are usually modeled as a bias random walk similar to bias stability in gyroscopes.

2.5.3 TEMPERATURE EFFECTS

As with gyroscopes, temperature changes cause fluctuations in the bias of the output signal. The relationship between bias and temperature depends on the specific device, however it is often highly nonlinear. Any residual bias introduced causes an error in position that grows quadratically with time. If the IMU has a temperature sensor then it is possible to apply corrections to the output signals in order to reduce the temperature dependent effects.

2.5.4 CALIBRATION ERRORS

Calibration errors (errors in scale factors, alignments and output linearity) appear as bias errors visible only when the device is undergoing acceleration. Note that these ‘temporary’ bias errors may be observed even when the device is stationary due to gravitational acceleration.

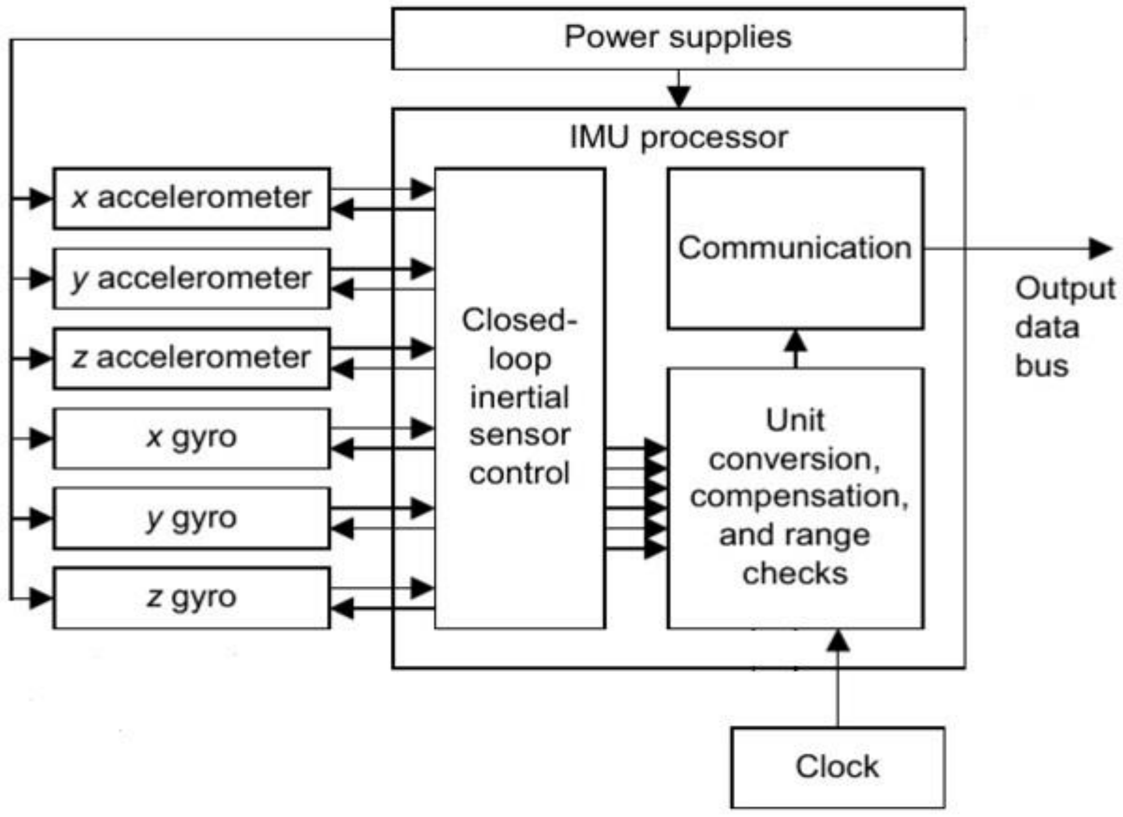

Figure 3 – Schematic of an Inertial Measurement Unit

There will be always errors on the accelerometers and gyroscopes measurements that affect the values of angular velocity, linear acceleration and orientation. To overcome these errors, a special recursive filter called KALMAN filter is considered and designed, which could predict the next position of the object from the initial start position and velocity measurements obtained from the IMU. The Kalman filter algorithm is discussed in detail in next chapter.

Chapter 3

KALMAN FILTER

The development of estimation techniques from noisy measurement goes back to 18th century when it was used by Gauss [8], who introduced least squares methods for estimation. Least squares estimation is one of the most widely used data analysis techniques. To find or estimate the numerical values of the parameters to fit a function to a set of data and to characterize the statistical properties of estimates. Later Fisher, Wiener and Kalman [4] worked on advanced optimal recursive filter techniques in order to achieve the same goal. Gauss addressed the issue for determining how many measurements are needed to estimate unknown quantities.

An optimal estimator, with mathematical proof, minimizes the estimation error in a statistical sense. To achieve this goal, it utilizes observation data and the a priori knowledge of the system. However, it is sensitive to the modeling errors and the statistics of the system, the computational intensity involved with the system estimator is also very hard to understand.

Estimation can be divided in to three forms. When the measurement is available at the same time as the estimate, the problem is stated as filtering. When the measurement is available before the estimate is made, the problem is known as prediction, and when the time of interest is available before the measurement, the problem is known as smoothing. The discussion below presents two of the estimation models introduced by Kalman [9] for estimation of both linear and nonlinear models.

3.1 KALMAN FILTER

The Kalman filter is a recursive predictive filter that uses state space techniques and recursive algorithms. It estimates the state of a dynamic system, even if the precise form of the system is unknown. This dynamic system can be disturbed by some noise, mostly assumed as white noise. To improve the estimated state, the Kalman filter uses measurements that are related to the state. The filter is very powerful in the sense that it supports estimations of the past, present, and even future states.

Thus the Kalman filter consists of two steps:

1. Prediction
2. Correction

In the first step, the state is predicted with the dynamic model. In the second step, it is corrected with the observation model, so that the error covariance of the estimator is minimized. In this sense, it is an optimal estimator [4].

This procedure is repeated at each time step with the state of the previous time step as the initial value. Therefore the Kalman filter is a "recursive filter". Kalman filter uses a feedback control mechanism in order to estimate a process. Noisy measurements are taken as feedback and used to estimate the process. The Kalman filter equations are categorized into two groups: time update and measurement update equations. The time update equations are used to project forward in time the current state and error covariance estimates for estimating the a priori state for the next step. The measurement update equations are used for getting the feedback in form of a new measurement and then using the a priori state estimate to obtain an improved a posteriori state estimate. Thus, the time update equations can be termed as prediction equations, while the measurement update equations can be termed as correction equations [4].

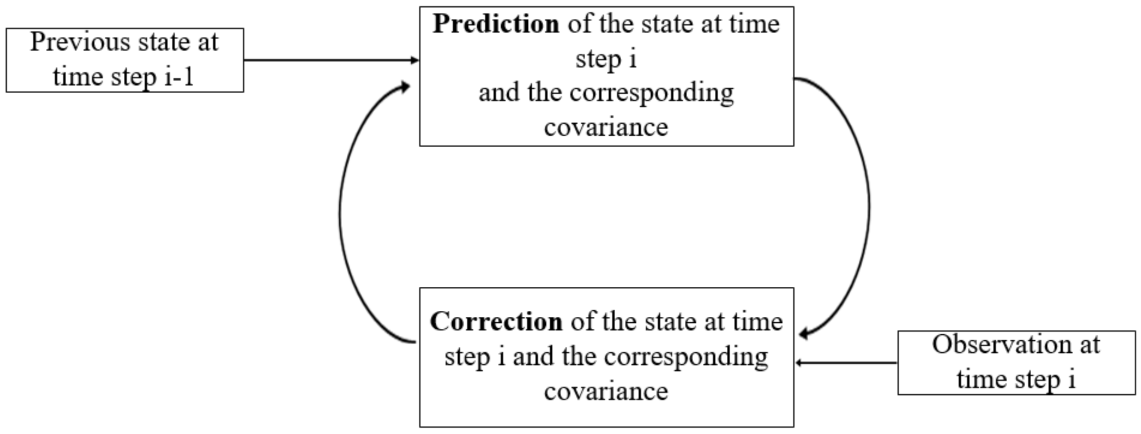
Previous state at time step i-1
Prediction of the state at time step i and the corresponding covariance
Correction of the state at time step i and the corresponding covariance
Observation at time step i

Figure 4 – Kalman filter

The basic components of the Kalman filter are the state vector, the dynamic model and observation model, which are described below.

3.1.1 STATE VECTOR

The state vector contains the variables of interest. It describes the state of the dynamic system and represents its degrees of freedom. The variables in the state vector cannot be measured directly but they can be inferred from measurable values.

Elements of the state vector can be for example the position, velocity, acceleration, etc. In our case the object may move with a constant or varying velocity in a straight or curved path. So the object has three degrees of freedom, the distance d , the velocity v, and acceleration a.

Thus the state vector is

$$x = \begin{bmatrix} d \\ v \\ a \end{bmatrix}$$

The state vector has two values at the same time: that is the *a priori* value, which represents the predicted value before the update, and the *a posteriori* value, which represents the corrected value after the update. The a priori value is typically denoted as x^{-} and the posteriori value is denoted as x.

3.1.2 DYNAMIC MODEL

The dynamic model describes the transformation of the state vector over time. It is represented by a system of differential equations.

$$\frac{d}{dt}x(t) = f(x(t))$$

Assuming we have a linear system, the discrete model can be written as

$$x(k+1) = Ax(k) + Bu(k)$$

where A and B represents the dynamic matrix, $x(k)$ represents the state vector, and $u(k)$ is the input vector.

3.1.3 OBSERVATION MODEL

The observation model represents the relationship between the state and the measurement. In the linear case, the measurement can be described by a system of linear equations, which depend on the state variables. Usually, observations are made at discrete time steps. The discrete measurement model can be written as

$$z(k) = H(x(k))$$

where $z(k)$ the vector of the observations at the time k. H is an observation matrix that describes how the state variables are mapped into outputs.

When the object moves, the accelerometer senses the acceleration values, which can be used to obtain the estimation of position. However, perfect performance and knowledge are impossible to achieve in the real world. Errors between the actual values and the estimated position will accumulate over time.

3.2 KALMAN FILTER ALGORITHM

The Kalman filter is a recursive estimator. This means that the estimated state from the previous time step and the current measurement are needed to compute the estimate for the current state.

The state of the filter is represented by two variables:

- x_{k+1}, the a posteriori state estimate at time k given observations up to and including time k.
- P_{k+1}, the a posteriori error covariance matrix (a measure of the estimated accuracy of the state estimate).

The Kalman filter has two distinct phases: predict and update. The predicted state estimate is also known as the a priori state estimate because, although it is an estimate of the state at the current time step, it does not include observation information from the current time step. In the update phase, a priori prediction is combined with current observation information to refine the state estimate. This improved estimate is termed the a posteriori state estimate.

Kalman filter tries to estimate the state x_k which is governed by a linear stochastic equation as:

$$x_{k+1} = Ax_k + Bu_k + w_k$$

The *state vector* x_k contains the variables to be estimated. The elements in the state vector can be position, velocity and acceleration etc. The state vector has two values at the same time, a priori state x_k and a posteriori state x_{k+1}, A and B are matrices which define the variables to be estimated, u_k is the input vector of the system from which the state x_k is derived. w_k is the process noise and is assumed to be White Gaussian noise with zero mean and small covariance matrix $Q(k)$. The process noise is used to account for unmodeled disturbances that affect the system dynamics.

The measurement or observation of the state is described by a system of linear equations which depends on the state variables. The measurement z_k is defined as:

$$z_k = Hx_k + v_k$$

Here, z_k is the observation made at instant k, while H is observation matrix.

Matrix Q_k represents the *process noise covariance matrix,* which describes the statistics of the noise related to sensor, and R_k represents the *measurement noise covariance matrix*. They are assumed to be independent, with normal probability distribution,

$$p(w) \sim N(0, Q)$$

$$p(v) \sim N(0, R)$$

Matrix A, representing *"the state transition matrix"*, relates the state at previous time step $k - 1$ to the current state. Matrix B, the "control matrix", relates to the control vector u_k to the state x_k. Similarly H, the *"measurement matrix"* relates the current state to the measurement.

3.2.1 PREDICTION

As shown in Figure 4, prediction is performed before measuring the signal. For predicting the current state, the dynamic noise is neglected in the measurement. Hence the equation representing the dynamic model will be:

$$\hat{x}_{k+1} = Ax_k + Bu_k$$

Similarly, the predicted measurement is also considered noise-free and the measurement equation is written as:

$$\hat{z}_k = H\hat{x}_{k+1}$$

The a priori and a posteriori errors are calculated using the a priori state estimate x_{k-1}, given the knowledge of the state at that time instant, and the a posteriori state estimate $\widehat{x}_k$, given the measurement at time k. The a priori estimation error and a posteriori estimation error [4] can then be written as:

$$e_k^- \equiv x_k - x_{k-1}$$

$$e_k \equiv x_k - \widehat{x}_k$$

The a priori estimation error covariance is then written as:

$$P_k^- = E[e_k^- e_k^{-T}]$$

The a posteriori estimation error covariance is written as:

$$P_k = E[e_k e_k^T]$$

The *error covariance matrix* P defines the expectation of the square of the deviation of the state vector estimate from the true value [10]. So in order to predict the state, the errors in the measured state and the previous estimated state $k-1$ are used to learn the behavior of the filter. The error covariance can then be written as:

$$P_k = AP_{k-1}A^T + Q$$

The *system noise covariance matrix* Q defines how the uncertainties of the state estimates increase with time due to noise sources in the system modeled by the Kalman filter, such as unmeasured dynamics and instrument noise. It is always a function of the time interval between iterations.

3.2.2 ESTIMATION

In order to correctly estimate the state, the predicted state is improved with the error in the observation.

$$x_k = x_k + \varDelta x$$

The error is calculated by using the Kalman gain and the error in the predicted and measured states. The Kalman gain $'K'$ is a n x m matrix that serves as a minimizing factor for error covariance. The Kalman gain can be written as:

$$K = P_k H^T (H P_k^- H^T + R)^{-1}$$

The variance of the state estimates is given by the diagonal elements of the error covariance matrix. It is necessary to minimize the trace P_k. From the equation above, it can be seen that as the measurement error covariance is minimized, the Kalman gain is increased and the filter starts trusting its prediction. Conversely, as the error covariance estimate starts approaching zero, the gain also starts decreasing, implying that the error is minimized, hence the predicted estimate can be trusted.

The corrected estimate of the state can then be written as:

$$\hat{x}_k = \hat{x}_k^- + K(z_k - H\hat{x}_k)$$

The difference $(z_k - H\hat{x}_k)$ is called the "residual", and reflects the discrepancy between the predicted and the actual measurement [10]. Thus if the measurement covariance is smaller than that of the predicted state, the measurement weight becomes high and the uncertainty is reduced.

The error covariance matrix of the a posteriori state is also updated to account for the Kalman gain at each predicted measurement as follows

$$P_k = (I - HK)P_k^-$$

Figure 5, below shows the block diagram of Kalman filter algorithm.

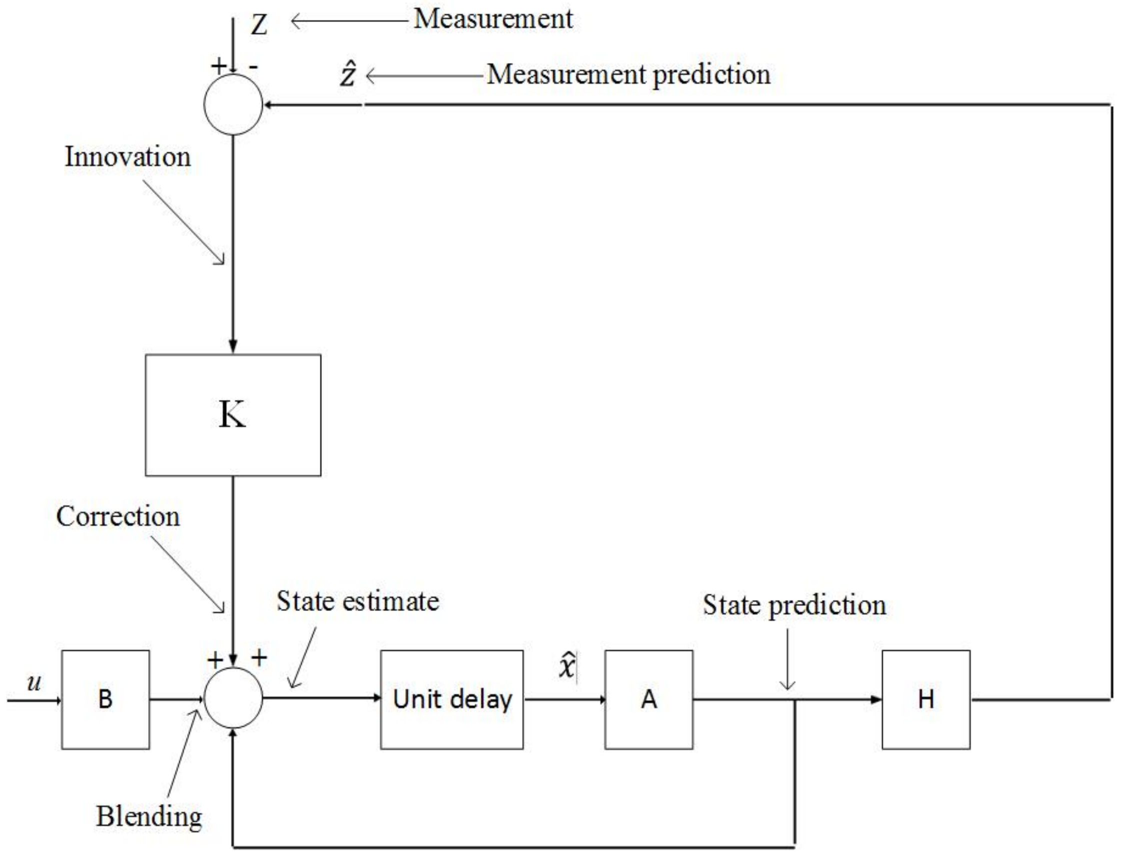

Figure 5 – Discrete time Kalman filter block diagram

A good way for designing a Kalman filter is to first select the states based on the properties of the system that are observable, can be modeled, and contribute to the output of the overall system. Based on the state selection, the system and measurement models can then be derived. Different types of measurement inputs to the same Kalman filter, such as position and velocity or velocity and acceleration, may be accumulated and updated at different rates.

3.3 EXTENDED KALMAN FILTER ALGORITHM

In order to remove noise using a Kalman filter, the process must be described by a linear system. Processes such a vehicle on road, satellite orbiting earth, wave propagation etc. can be approximated by linear systems. But in practice the dynamic or the observation models can be nonlinear.

One approach to deal with nonlinear problems consists of the Extended Kalman filter (EKF). This filter linearizes the system about the current estimated state. Thus the system must be represented by continuously differentiable functions.

An improved state estimate can be obtained with no prior knowledge of a nominal trajectory. The EKF retains the linear calculation of the covariance and filter gain matrices, and updates the state estimate using a linear function of the filter residual.

However it uses the original non-linear equations for the state propagation and output vector.

The filter equations are given as follows [11]:

Time update equations:

$$\hat{x}_{k+1} = f(\hat{x}_k, u_k, 0)$$

$$P_{k+1}^{-} = A_k P_k A_k^T + W_k Q_k W_k^T$$

where A_k is the state transition matrix, Q_k is the process noise covariance W_k is the Jacobians at time step k.

Measurement update equations:

$$\hat{x}_k = f(\hat{x}_k^{-}, u) + K[z_k - h(\hat{x}_k^{-}, 0)]$$

$$K_k = P_k^{-} H_k^T (H_k P_k^{-} H_k^T + R_k)^{-1}$$

$$P_k = (I - H_k K_k) P_k^{-}$$

An important aspect in EKF is that the partial derivatives are evaluated at the current values of the state estimates and control inputs rather than their nominal values. However, these time varying matrices cannot be computed before the state, since they are functions of the state estimates. This makes the computational effort required for the update equations more time consuming.

For EKF, improved state estimates could be obtained from a second order or higher order filter, but the computational effort also increases considerably. In general, in most applications, EKF yields satisfactory results.

Chapter 4

IMPLEMENTAION AND ANALYSIS

4.1 BIAS ESTIMATION AND CALIBRATION

Bias estimation error is the difference between the estimated value and the true value of the parameter being estimated. In IMUs, the output performance suffers from the accelerometer and gyroscope biases. A bias error, if not removed from the measurement, is integrated twice in the distance and once in the velocity estimation, which will cause inaccurate result. So a constant bias (error) in acceleration becomes a linear error in the velocity and a quadratic error in the distance during the estimation.

4.1.1 INPUT RANGE

The IMU can measure only up to a certain input range of the maximum angular rate or acceleration. Acceleration or rotation outside of the range will result in bad or no measurements. A strong vibration due to the movement of the object can lead to poor tracking, as the sensors are already saturated with the signal. The bandwidth of the sensor plays an important part in the ability of the sensors to measure the actual motion. A low bandwidth means high frequency vibration, therefore the resulting measurement suffers from aliasing. To overcome this problem a method called vibration isolation is recommended in situations where significant vibrations are present or when low bandwidth sensors are used. Vibration isolation is the process of isolating an object, such as a piece of equipment, from the source of vibrations.

4.1.2 BIAS

When there is motion from a given physical input, the sensor outputs a measurement offset by the bias. For example, if the IMU is stationary and level, the vertical Z - axis measures the effect of gravitational acceleration. Gravity has a nominal acceleration of $9.81\, m/s^2$, but if the measurement is biased, the IMU may report $9.75\, m/s^2$ or even $9.95\, m/s^2$. The difference between the real value and the output is the bias.

4.1.3 CALIBRATION

The accelerometer bias has a major impact on the overall performance of an IMU. Any bias in the accelerometer output will cause a shift in the measured acceleration vector from its true direction, which will directly lead to errors in the calculated distance. When determining distance, accelerometer bias error will have an influence on the pitch and roll accuracy.

There are different calibration methods used in industry. In this project the measured values are calibrated by taking the mean of the measured value. Neglecting the mean in the measured value will give the bias in the measurement. This bias value is subtracted from the accelerometer value and the estimation is done using the Kalman filter.

In this project the movement of a robot is measured in X, Y and Z directions with respect to time for 60 seconds. However this measurement contains various bias errors. The movement in Z direction is considered in the project and the estimation is done for the acceleration values of Z axis alone. Figure 6, shows the measured acceleration values with respect to time in Z axis. Estimating the same acceleration using the Kalman filter can be seen in Figure 7, which has peaks up and down due to the bias. Therefore estimating the distance and velocity may not be as accurate as we expect.

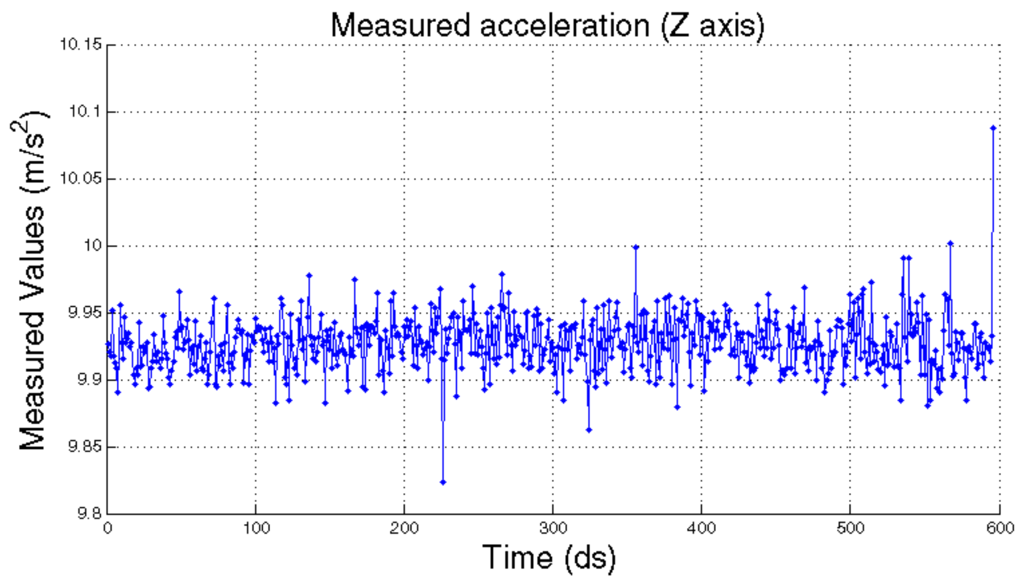

Figure 6 – Measured acceleration in Z – axis (time step = 0.1s)

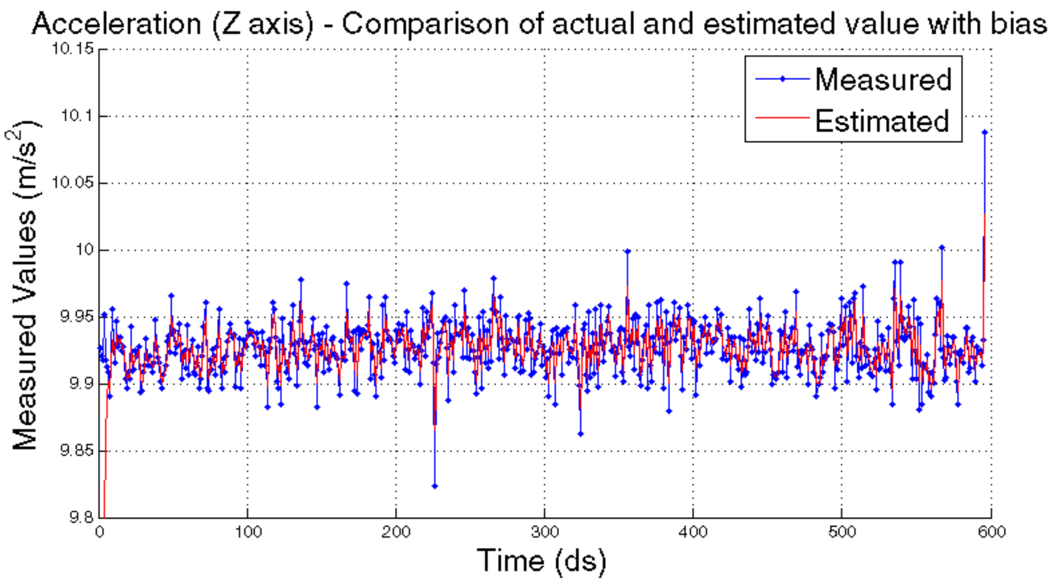

Figure 7 – Acceleration estimation compared with actual value with bias

(time step = 0.1s)

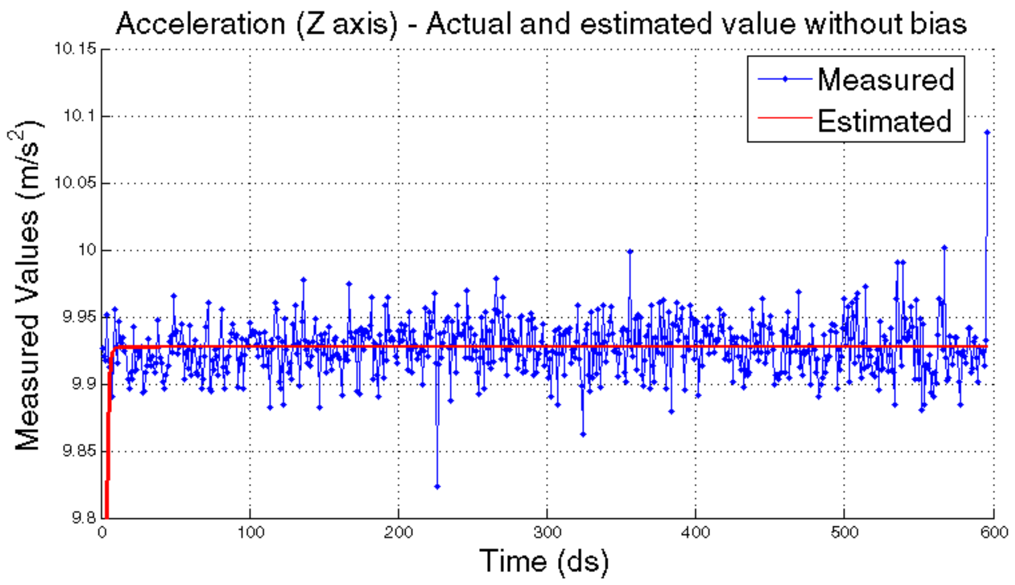

Figure 8 – Acceleration estimation in Z – axis without bias (time step = 0.1s)

In Figures 7 and 8, the blue line indicates the acceleration measurement from the device with the bias and the red line indicates the estimation. Therefore, before filtering the acceleration data using the Kalman filter, the bias in the data is removed from the measurement for estimation. Figure 8, shows the estimation of the acceleration after removing the bias. Figures 9 and 10 show an estimation of the distance travelled and the velocity using Kalman filter.

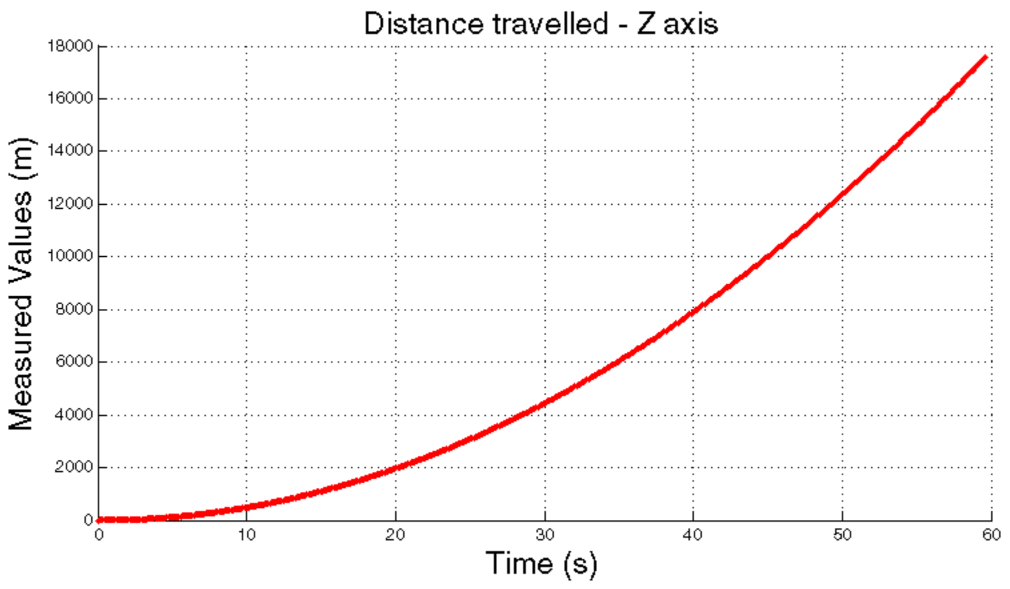

Figure 9 – Distance estimation for Z – axis (time step = 0.1s)

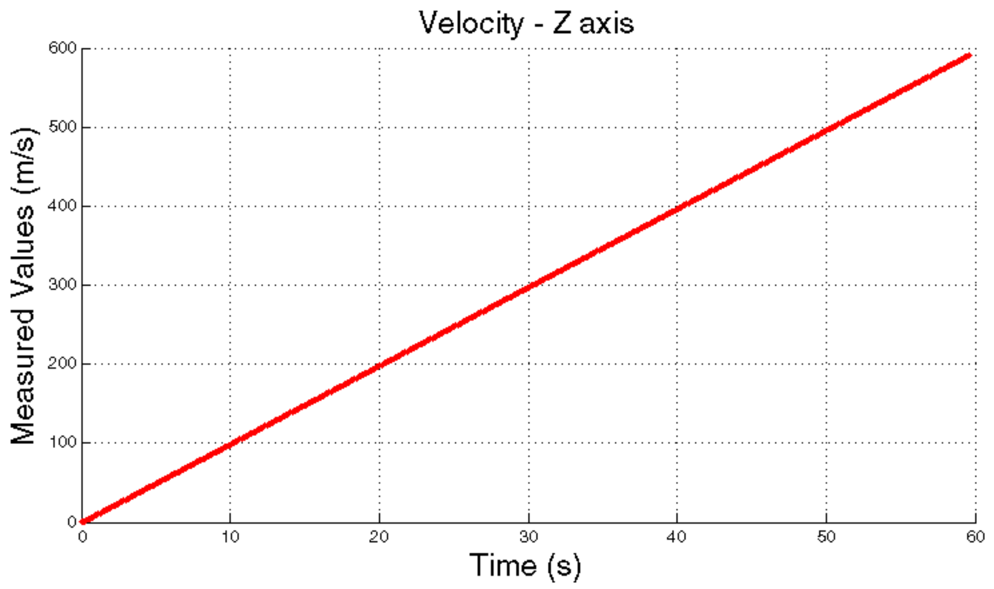

Figure 10 – Velocity estimation for Z – axis (time step = 0.1s)

4.2 IMPLEMENTATION OF KALMAN FILTER

When using Kalman filtering, it is quite simple to choose between a linear, extended, or unscented Kalman filter. If the system has high dynamics and one is trying to estimate the states, then the Kalman filter cycle has to be very fast. It might not be possible to do all the calculations within each cycle. This might pose a problem when a system is on a microprocessor. Secondly, if the system is non-linear then a non-linear Kalman filter should be chosen. This leads to much more complicated calculations which require faster processing units. For these reasons, a normal linear Kalman filter can be used with a much lower cycle resulting in lower requirements for the speed of the processor.

Since we are considering a moving object, the state consists of the object's distance d_k and velocity v_k. The input a_k is the command acceleration and the output y_k is the measured distance and velocity. The acceleration changes instantly and the distance can be measured every Δt second. According to the laws of physics, the velocity v_k will be governed by the following equation:

$$v_k = v_{k-1} + \Delta t(a_{k-1})$$

The present velocity v_k will be equal to the velocity one time step before the current one plus the acceleration command a_k multiplied by Δt. Similarly, the present distance will be equal to the distance one time step before the current one plus the velocity command v_k multiplied by Δt. The distance is given by

$$d_k = d_{k-1} + \Delta t v_k + \frac{1}{2}\Delta t^2 a_{k-1}$$

The state vector x_k can be defined as:

$$x_k = \begin{bmatrix} d_k \\ v_k \\ a_k \end{bmatrix}$$

The state equations can be written as:

$$x_k = \begin{bmatrix} 1 & \Delta t & \frac{\Delta t^2}{2} \\ 0 & 1 & \Delta t \\ 0 & 0 & 1 \end{bmatrix} x_k + \begin{bmatrix} \frac{\Delta t^2}{2} \\ \Delta t \\ 1 \end{bmatrix} a_k + w_k$$

The measurement equations can be written as:

$$y_k = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix} x_k + v_k$$

The estimation of the noise parameters is very important. The process noise corresponds to the noise related to the sensor. The process noise here includes the sensor output noise and its offset value. Similarly, the measurement noise corresponds to the noise in the signal when it is measured.

We initialize $\hat{x}_0$ to zero or to our best initial estimate of the distance, velocity and acceleration. Then we execute the Kalman filter equations once per time step.

From the equations above, the state transition matrix, measurement matrix and other variables are shown in table 1, where Δt in the state transition matrix A denotes the time interval between two consecutive measurements.

State transition matrix A	$\begin{bmatrix} 1 & \Delta t & \frac{\Delta t^2}{2} \\ 0 & 1 & \Delta t \\ 0 & 0 & 1 \end{bmatrix}$
State vector x_k	$\begin{bmatrix} d_k \\ v_k \\ a_k \end{bmatrix}$
Initialization of state vector x_k	$\begin{bmatrix} d_0 \\ v_0 \\ a_0 \end{bmatrix}$
Measurement matrix H	$[0 \quad 0 \quad 1]$
Initialization of covariance matrix P_0 , Measurement matrix R, Process Noise Q	3 x 3 Matrix (given)

Table 1 – Kalman filter matrices

The acceleration value a_k measured for each axis from the accelerometer is used in the state vector x_k respective to the axis which has to be estimated.

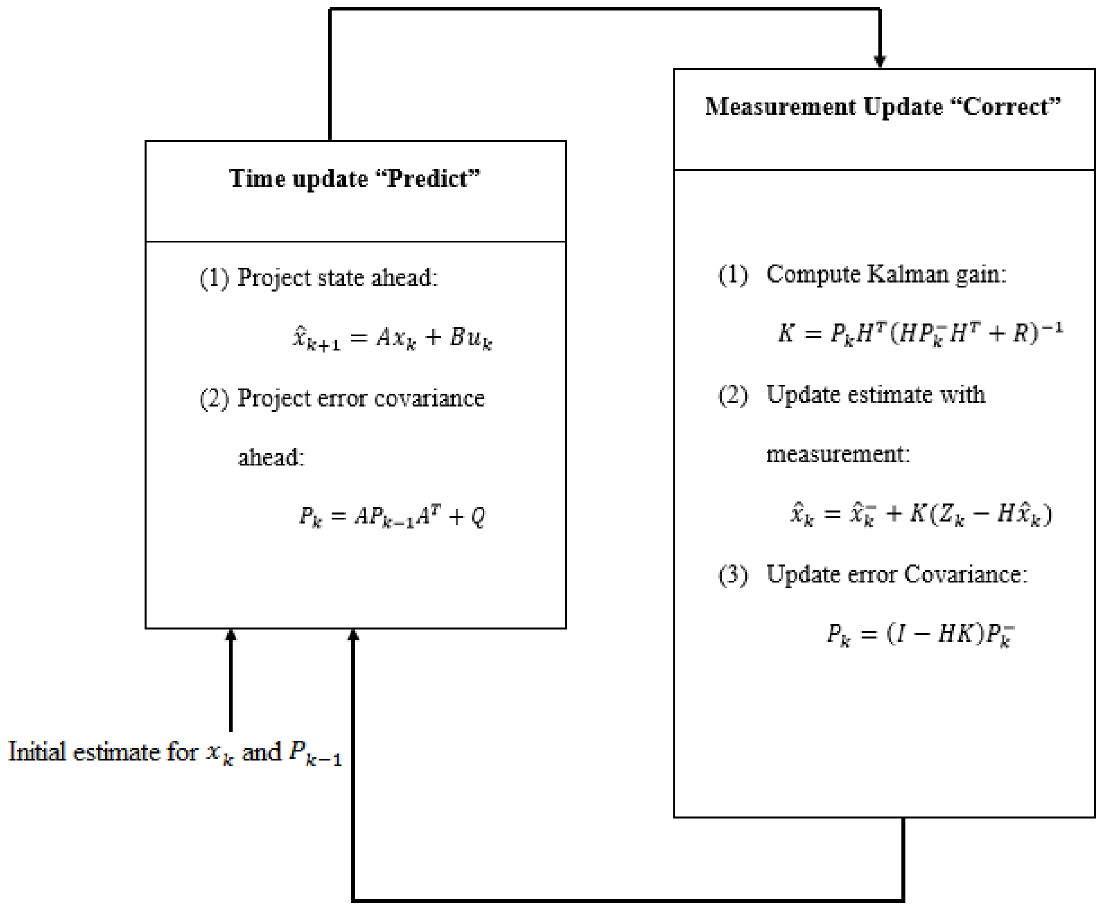
Time update "Predict"
(1) Project state ahead:
$\hat{x}_{k+1} = Ax_k + Bu_k$
(2) Project error covariance ahead:
$P_k = AP_{k-1}A^T + Q$
Measurement Update "Correct"
(1) Compute Kalman gain:
$K = P_k H^T (HP_k^- H^T + R)^{-1}$
(2) Update estimate with measurement:
$\hat{x}_k = \hat{x}_k^- + K(Z_k - H\hat{x}_k)$
(3) Update error Covariance:
$P_k = (I - HK)P_k^-$
Initial estimate for x_k and P_{k-1}

Figure 11 – Complete operation of Kalman filter with equations

Kalman filter settings involve the covariance matrix Q representing model noise, and the covariance matrix R of observation noise. The initial parameter for the error covariance P_k is set at the beginning. Since the Kalman algorithm employs the measurement and time update stages, the parameter covariance matrix P_k is adjusted to reflect the quality of the estimated parameter. The covariance Q affects P_k in the time update stage, whereas in the measurement update P_k, R and H influence the Kalman gain K which in turn affects the covariance P. Therefore, the Kalman filter performance depends upon setting these quantities to the appropriate values that reflect the quality of the measurements.

4.3 ESTIMATION OF GRAVITY FROM IMU

In order to get the values in all three axes, the sensor is held stationary such that the x-axis measures acceleration in the left and right direction, y-axis measures acceleration in the forward and backward directions while z-axis measures gravity. Gravity is always pointing down towards the center of the earth. It is fairly constant, assuming the objects distance from the center of the earth is constant, where $1g$ is equal to $9.81m/s^2$. Since we are considering all three axes, the gravity vector is defined as [0 0 -1] in units of g. The unit g is often used to describe accelerations in relation to the gravity experienced on earth.

In the three axis accelerometer, the measurement is modeled as

$$z_k = a_k - g_k - b_k + v_{A,k}$$

where z_k is the sensor output at time k, a_k corresponds to the accelerations due to linear and rotational movement, b_k is the offset of the sensor, and $v_{A,k}$ is the observed noise. The second term g_k, is the gravity component. Note that these variables are vectors, pointing to a specific direction in a three dimensional Cartesian coordinate system,

$$z_k = \begin{bmatrix} z_k^x \\ z_k^y \\ z_k^z \end{bmatrix}$$

After measuring the acceleration for the device, the data is processed through the stages defined in the Kalman implementation such as calibration, orientation calculation and filtering of the data and then the distance and velocity are calculated. In this project the analysis is done for the accelerometer with the sampling frequency of $16hz$.

Figure 12 shows gravity estimation from accelerometer measurement along the z – axis after removing the bias.

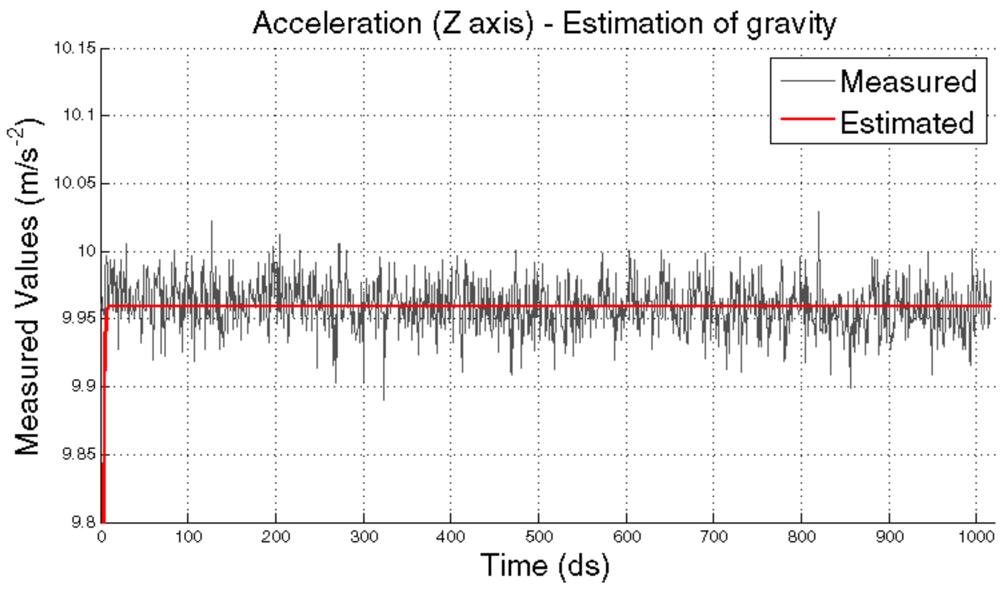

Figure 12 – Gravity estimation from the IMU

The black line is the acceleration measurement and the red line is the accurate acceleration estimation of the object moving which is determined from the Kalman filter. From the observed data, the acceleration is approximately 9.96 m/s^2. This is because the acceleration value includes the earth gravity which is 9.81 m/s^2. So by eliminating the earth's gravity, the actual acceleration is 0.51 m/s^2 approximately.

4.4 MOTION ESTIMATION

Accelerometers measure acceleration but motion estimation requires displacement or distance. Distance estimation can be achieved using the accelerometer value and the Kalman filter to estimate the state vector.

Integrating the acceleration value received from the accelerometer will give the velocity of the object:

$$\int a = v$$

Then integrating the velocity will lead to the distance travelled:

$$\int v = d$$

Since we are using the Kalman filter for estimation, the integration is replaced by the state transition matrix A. Multiplying the acceleration values $[a_x \; a_y \; a_z]$ with the state transition matrix A will provide $[d \; v \; a]$ of the axis at time Δt, where d represents distance travelled, v is the velocity and a indicates the acceleration. The simulation results below show the distance, velocity and acceleration estimation from the measured acceleration values with frequency of $16 \; hz$.

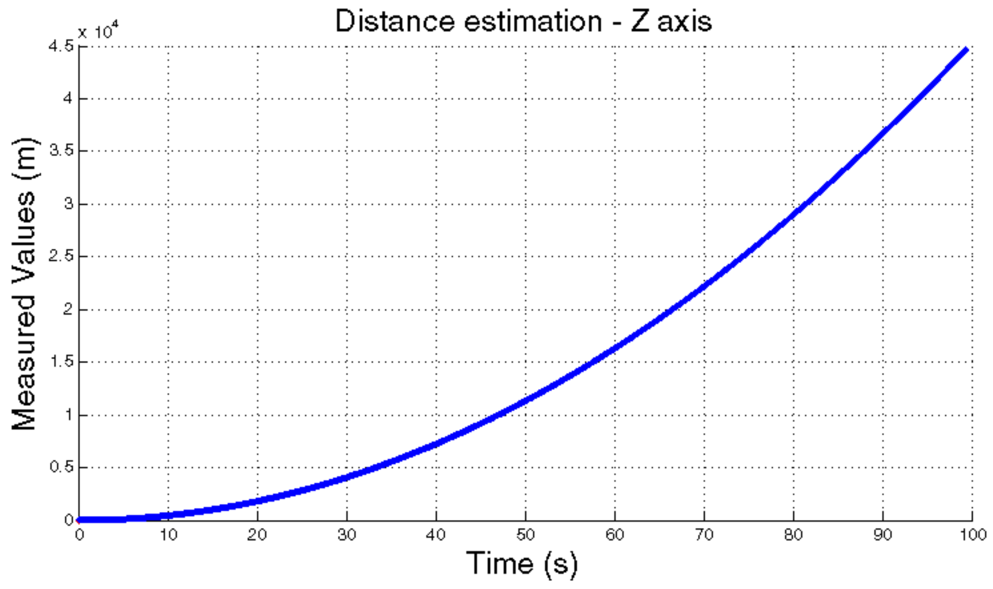

Figure 13 – Distance Estimation (time step = 0.06s)

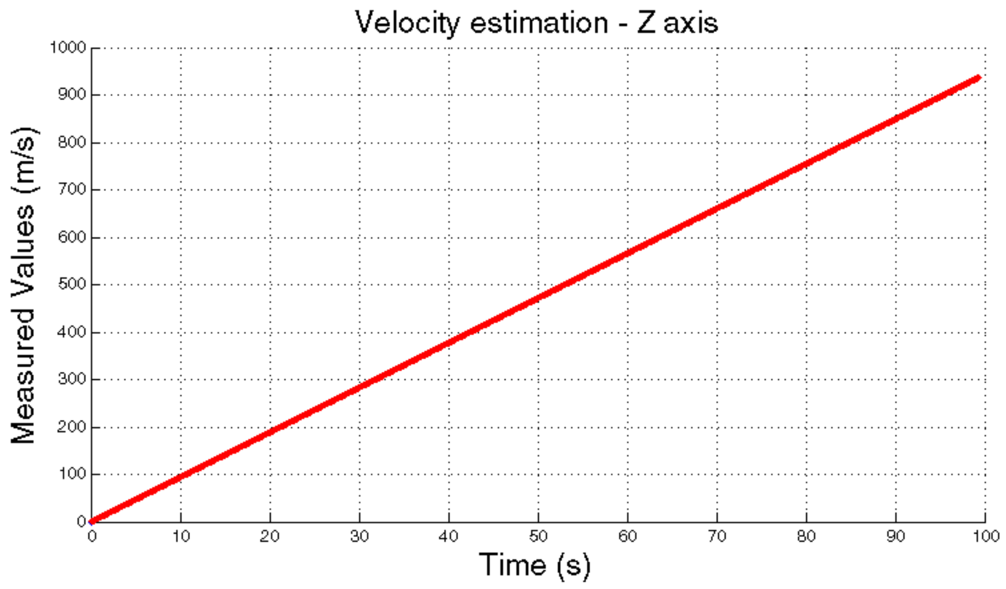

Figure 14 – Velocity Estimation (time step = 0.06s)

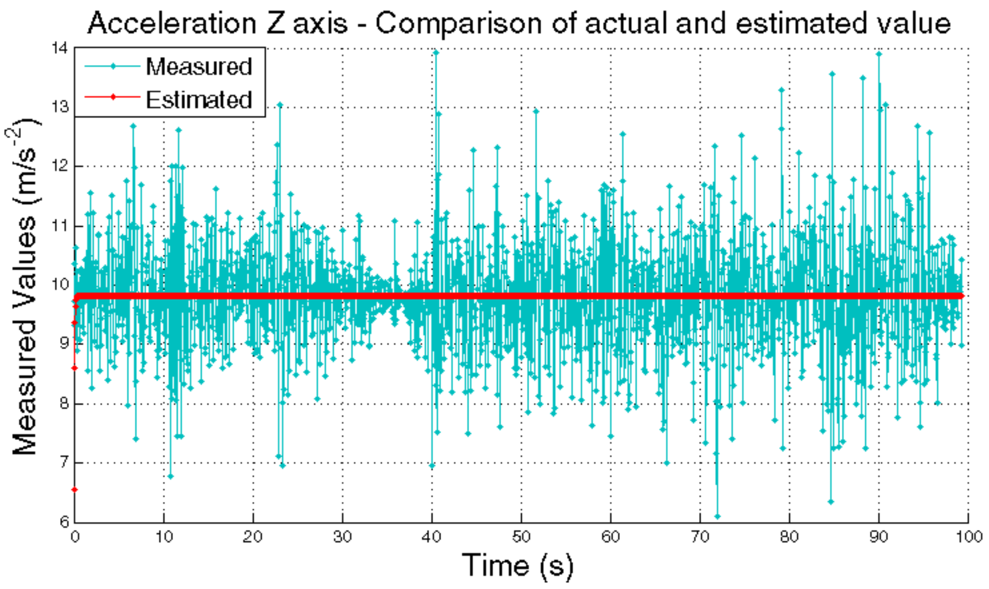

Figure 15 – Acceleration Estimation (time step = 0.06s)

4.5 COMPARISION OF KALMAN FILTER ESTIMATION WITH GPS ESTIMATION

The Global Positioning System (GPS) is a satellite based system for high – accuracy position, velocity and time estimation [12]. GPS uses a coordinate system called World Geodetic System (WGS) to express a point on earth. The global coordinates consist of three components: latitude, longitude and altitude. Currently, many land vehicles are equipped with global positioning systems (GPS), which can give an acceptable positioning. However, GPS is affected by several errors like signal unavailability, voluntary or involuntary signal interference, like jamming, spoofing and ionosphere and troposphere delays [12]. All these errors affect the integrity and reliability of the navigation solution, and only some of them can be reduced or mitigated. Others are intrinsic in GPS functioning so they cannot be removed. On the other hand IMU provides information about position, velocity and attitude with a higher rate than the GPS.

The time and speed of a moving object are collected from a GPS and used for distance and velocity estimation using Kalman filter. Figure 16, shows the distance travelled estimated using Kalman filter.

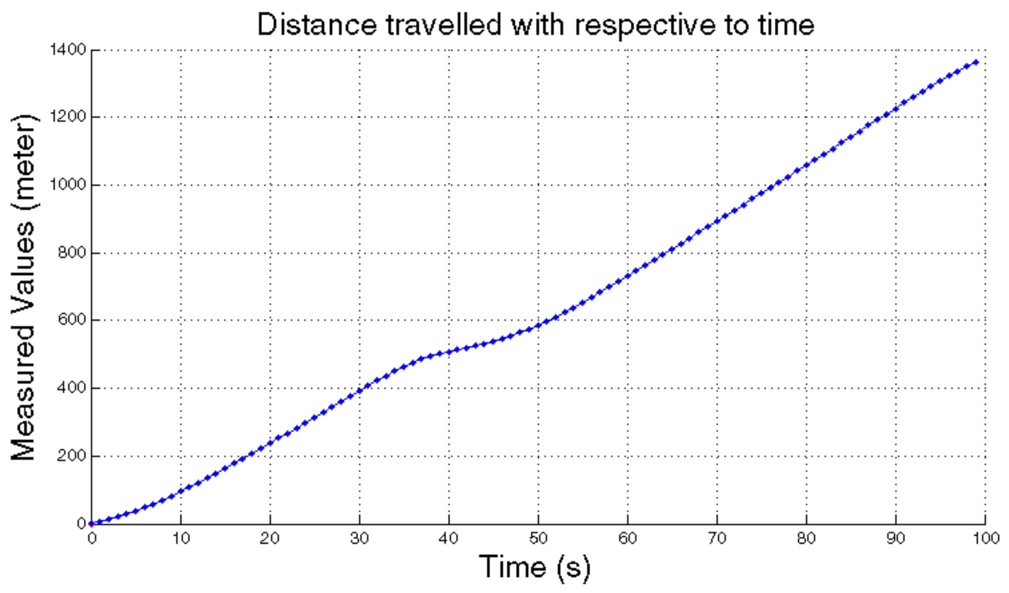

Figure 16 – Distance travelled estimated by Kalman filter using GPS data

Figure 17 shows a comparison of the speed measured from the GPS, and the estimated speed. The blue color indicates the measured velocity from the GPS and the red color indicates the estimated speed using the Kalman filter. In figure 16, the graph is almost constant between 35 seconds and 45 seconds, which is due to the decrease in velocity. This can be also seen in figure 17.

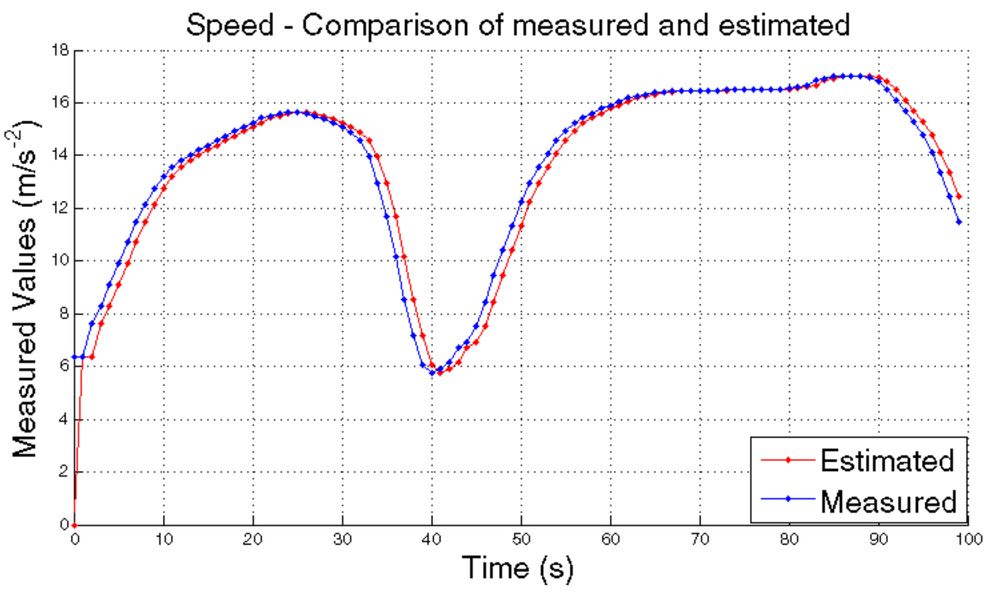

Figure 17 –Comparison of measured and estimated speed using GPS data

4.6 ERROR ANALYSIS

There are different types of errors that might affect the accuracy of the acceleration measurement in the device. In this section the errors are analyzed and discussed.

Generally, errors are divided into two basic categories.

1) Deterministic errors

2) Stochastic errors

4.6.1 DETERMINISTIC ERRORS

The first category of IMU errors is deterministic errors [13]. Deterministic errors can be precisely determined and represented using either scalar numbers or matrices. Their values either remain constant over time or vary in a way that can be mathematically modeled. Deterministic errors can be subdivided as static biases and misalignment errors which remain constant during the measurement process.

4.6.2 STOCHASTIC ERRORS

The second category of IMU errors is the stochastic errors. Unlike deterministic errors, which are inherently stable and repeatable, the behavior of stochastic errors result from random processes. The random nature of stochastic errors means that their values change constantly over time and there is no chance of predicting the exact value of the error at a given time. Stochastic errors are subdivided into the following types as mentioned below.

4.6.2.1 MEASUREMENT NOISE

Measurement noise is a zero-mean random process that appears in the sensor data and obscures its true nature. A simulation of IMU acceleration data with frequency of $16hz$ is shown in Figure 18, it consists of the measured data with bias and the original measurement after removing the bias. The blue color indicates the measurement that was observed from the device which has measurement noise and the red color indicates the actual measurement after removing the bias. For removing the bias, a simple calibration technique is used as explained in section 4.1.

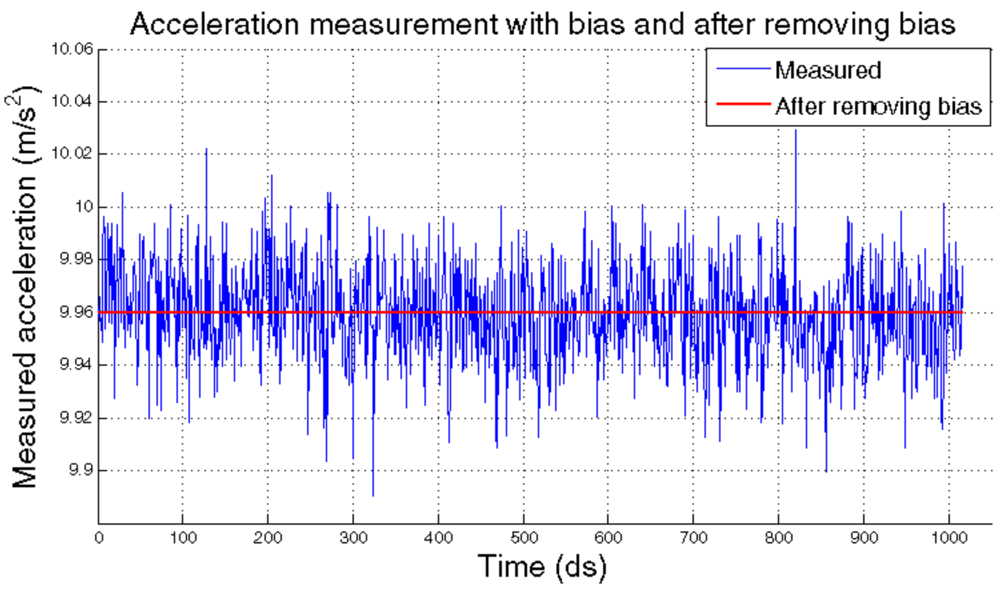

Figure 18 - Measurement noise

4.6.2.2 DRIFTING BIAS

Drifting bias is another type of stochastic error. A drifting bias is an error that starts off at a specific value and then randomly drifts away from that value slowly over time (this type of behavior is commonly referred to as a “random walk” behavior). Figure 19 shows a steady state accelerometer data measured from the IMU at frequency of 11 hz.

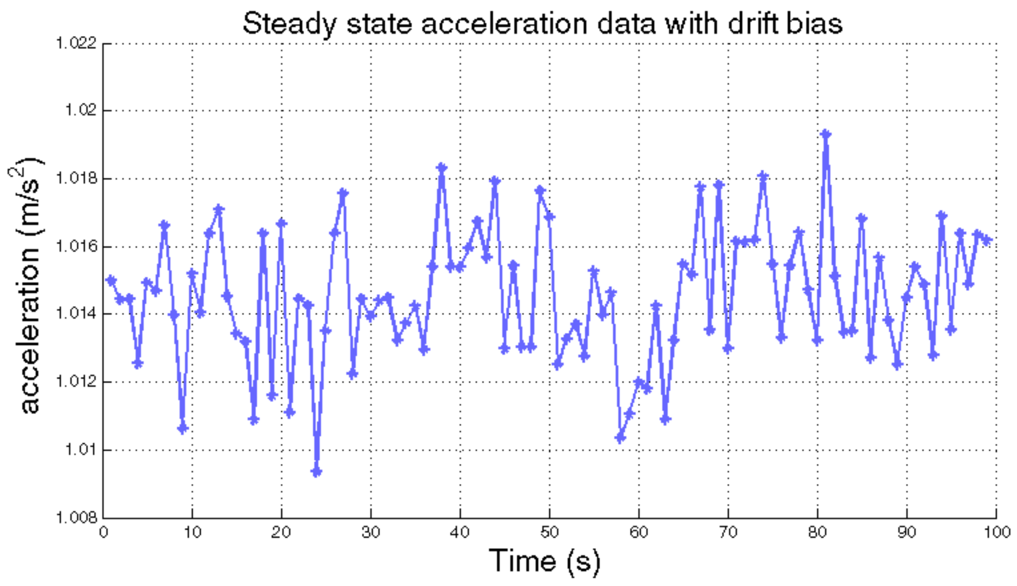

Figure 19 – Drift bias

Because of bias in the measurement, the estimation of the position and velocity of the object might not be accurate and may lead to confusion. The bias is calculated using the mean method and can be seen in the first plot of Figure 20. This bias is removed from the measurement, which allows to get an accurate measurement of the acceleration and therefore the distance and velocity can be estimated using the Kalman filter.

The second and third plots of Figure 20 show the measurement after removing the bias and the estimation from the Kalman filter. The distance and the velocity are estimated and shown in Figure 22.

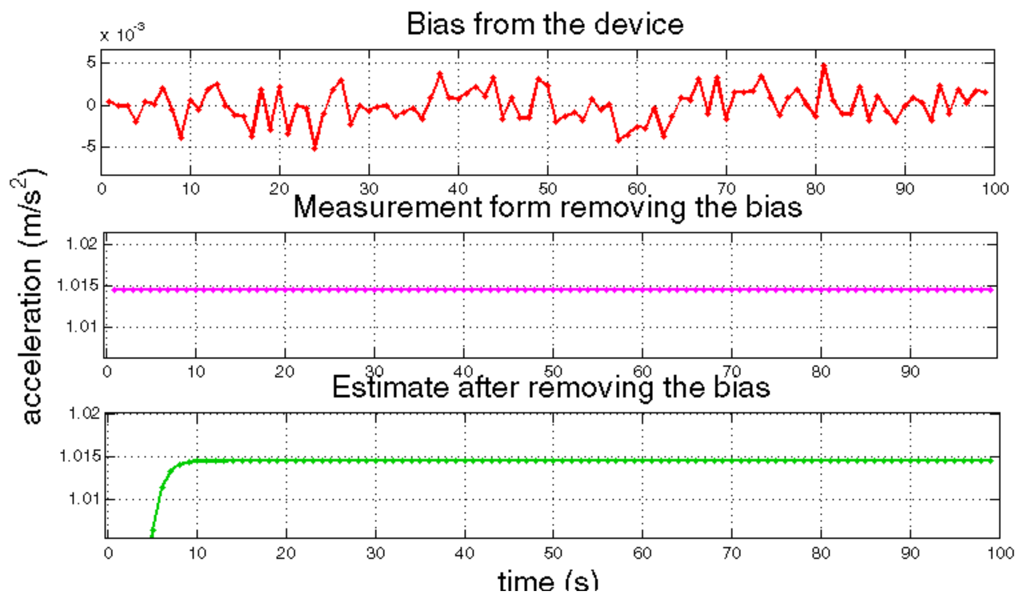

Figure 20 – Bias removed from the device

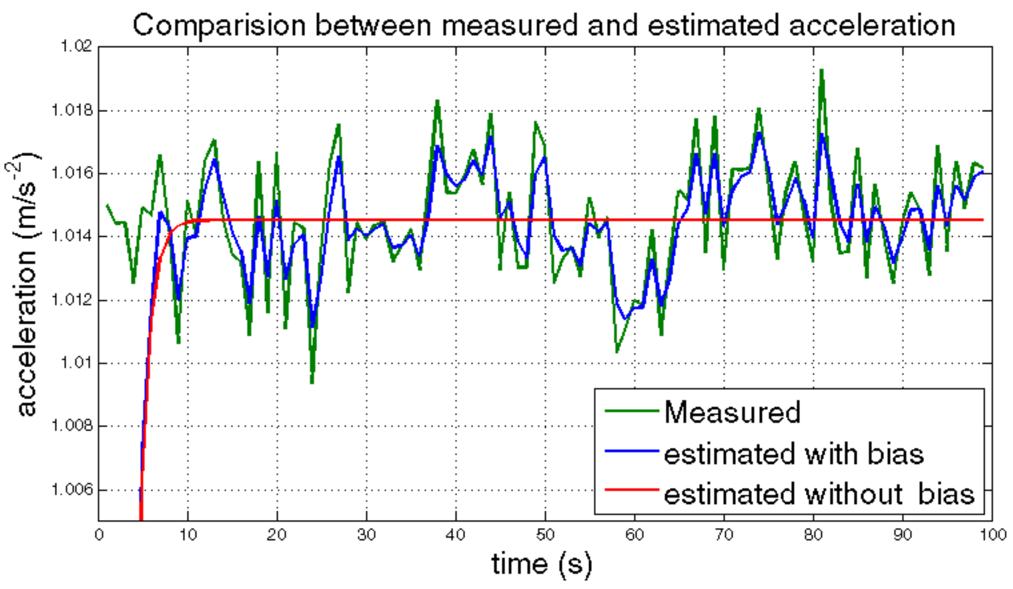

Figure 21 - Comparison of measured and estimated acceleration data

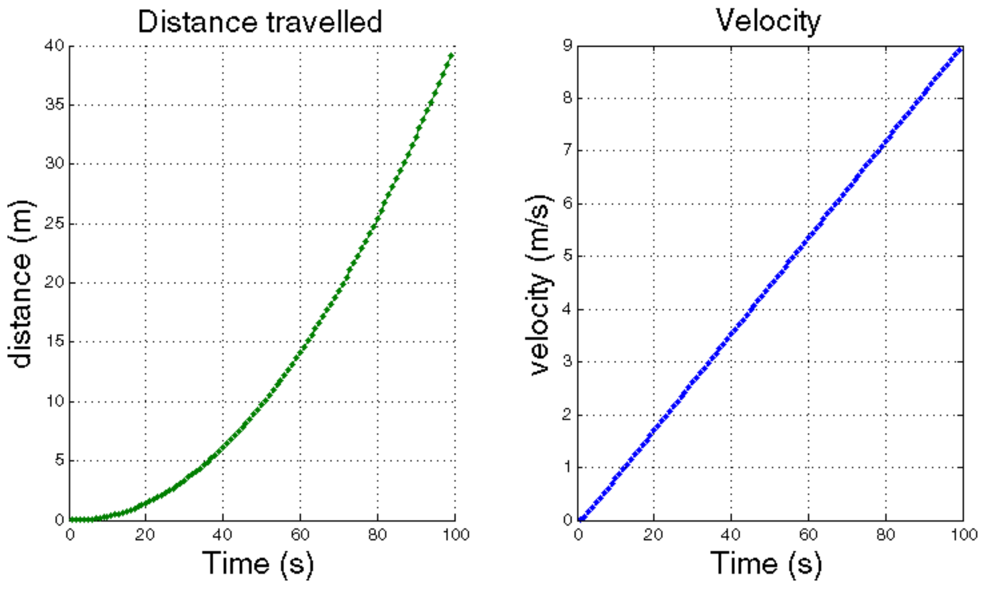

Figure 22 – Distance and velocity estimated using Kalman filter

Chapter 5

CONCLUSION

This project presents a successful implementation of Kalman filter to estimate the motion of an object. The Kalman filter is programmed and simulation results are shown to illustrate the approach. This project uses the Kalman filter for estimating the distance travelled and the speed of an object based on acceleration values obtained from the IMU. The effect of uncertainties and error factors in the measurement are also considered and analyzed, and the results are plotted. The Kalman filter estimation algorithm designed in this project was also compared with the real time GPS estimation. Deterministic and stochastic errors in the measured values are removed by taking the mean value for an accurate estimation of the velocity and the distance travelled by the object.

There are certain aspects of this project that may be explored in future works. For example, in this project only Z - axis is considered. In future, the Kalman filter can be modeled to take all three axes into account for the distance and velocity estimation of the object, which will be more effective and can be implemented in a wide variety of applications.

Appendix A: MATLAB CODE IMPLEMENTING KALMAN FILTER FOR DISTANCE TRAVELLED AND VELOCITY ESTIMATION

```
clc;
clear all;

%%File reading
filename = 'seady_2at_10hz.csv';
num = csvread(filename);

time=[];     u=[];        k=[];
u1=[];       u2=[];       u3=[];
n = 596;         % no of readings
for i = 1:1:n
      time(i) = num(i);
      u1(i) = num(i+n);
      u2(i) = num(i+(2*n));
      u3(i) = num(i+(3*n));
end
time = time';     u1 = u1';   u2 = u2';   u3 = u3';

for t=1:1:n
    if (u1(t)<0)
        u1(t) = u1(t)*(-1);
    else
        u1(t) = u1(t);
    end
end

for t=1:1:n
    if (u2(t)<0)
        u2(t) = u2(t)*(-1);
    else
        u2(t) = u2(t);
    end
end

for t=1:1:n
    if (u3(t)<0)
        u3(t) = u3(t)*(-1);
    else
        u3(t) = u3(t);
    end
end

%% define our meta - variables
duration = size(time); %how long the simulation is
```

```
dt = 0.1;          %10 Hz continuously looking for measurement

%% define update equations
A = [1 dt dt^2/2;0 1 dt;0   0 1];   %transistion matrix
B = 0;
H = [1 0 0;0 1 0;0 0 1];     %genral form matrices ; measuremnet matrix

%% define main variables
u=0;
k1 = [u1 u2 u3];              %control vector ; acceleration matrix
Q= [0; 0; 0];                  %initized state ; [position velocity]
Q_estimate = Q;
P = eye(3);          % q --> estimated process error covariance.
Ex = eye(3);                  %estimate of initial object position
R = [1 0 0 ;0 1 0;0 0 1];

%% Initialize result variables
Z_p = [];    Z_v = [];    Z_a = [];

x_estimate_az = [];     y_estimate_az = [];     z_estimate_az =[];
 %estimate of the object path using Kalman filter

u3_bias=[];       u3_perfect =[];
ll = mean(u3);
 %taking the mean of u3 will give the value and it is subtracted from
%each u3 to find bias and u3_perfect is calculated.

for t = 1:1:duration
    u3_bias(t) = u3(t)-ll;
    u3_perfect(t) = u3(t)-u3_bias(t);
end

%% Kalman Filter for Z

Q_estimate= [0; 0; 0];

for t = 1:1:duration
% Predict
    % Predicted State
    Q_estimate_curr = A * Q_estimate + B * u;    %-----------1
    Z_p = [Z_p; Q_estimate_curr(1)];
    Z_v = [Z_v; Q_estimate_curr(2)];
    Z_a = [Z_a; Q_estimate_curr(3)];

    %predicted next covariance
    P = A * P * A' + Ex;                         %-----------2

% Update
    %Kalman Gain
    K = P*H'*inv(H*P*H'+R);                      %-----------3
    % Update the state estimate.
    y = [Q_estimate_curr(1);Q_estimate_curr(2);u3(t)];
    Q_estimate = Q_estimate_curr + K * (y - H * Q_estimate_curr); %--4

    % update covariance estimation.
    P =  (eye(3)-K*H)*P;                            %-----------5

% Store for Plotting

    x_estimate_az = [x_estimate_az; Q_estimate(1)];
    y_estimate_az = [y_estimate_az; Q_estimate(2)];
    z_estimate_az = [z_estimate_az; Q_estimate(3)];
end

%% Plotting
tt = 1:1:duration;
%x
figure;
hold on
grid on
plot(time(tt),Z_v,'-b.');
plot(time(tt),y_estimate_az,'-r.');
title('Velocity - Z axis');
xlabel('Time (s)');
ylabel ('Measured Values (m/s)');
hold off

figure;
hold on
grid on
plot(time(tt),Z_p,'-r.');
plot(time(tt),x_estimate_az,'-b.');
title('Distance - Z axis');
xlabel('Time (s)');
ylabel ('Measured Values (m)');
hold off

figure;
hold on
grid on
plot(time(tt),u3(tt),'-b.');
plot(time(tt),z_estimate_az,'-r.');
title('acceleration Z axis - Comparison of actual and estimated
value');
xlabel('Time (s)');
ylabel ('Measured Values (m/s^-^2)');
hold off
```

Appendix B: MATLAB CODE FOR COMPARISON OF KALMAN FILTER ESTIMATION WITH GPS ESTIMATION

```
clc;
clear all;

%%File reading
full =[ 0        6.3480
    1.0000    6.3480
    2.0000    7.6444
    3.0000    8.3149
    4.0000    9.1196
    5.0000    9.9243
    6.0000   10.7289
    7.0000   11.4889
    8.0000   12.1595
    9.0000   12.7406
   10.0000   13.1876
   11.0000   13.5453
   12.0000   13.8135
   13.0000   14.0370
   14.0000   14.2158
   15.0000   14.3947
   16.0000   14.5735
   17.0000   14.7076
   18.0000   14.9311
   19.0000   15.0652
   20.0000   15.2440
   21.0000   15.4228
   22.0000   15.5122
   23.0000   15.6017
   24.0000   15.6464
   25.0000   15.6464
   26.0000   15.6017
   27.0000   15.5122
   28.0000   15.3781
   29.0000   15.2440
   30.0000   15.0652
   31.0000   14.8864
   32.0000   14.5735
   33.0000   13.9476
   34.0000   12.9641
   35.0000   11.6677
   36.0000   10.1478
   37.0000    8.5384
   38.0000    7.1526
   39.0000    6.0797
   40.0000    5.7668
   41.0000    5.9009
```

```
42.0000    6.1691
43.0000    6.7056
44.0000    6.9291
45.0000    7.5550
46.0000    8.4490
47.0000    9.4325
48.0000   10.4160
49.0000   11.3548
50.0000   12.2489
51.0000   12.9641
52.0000   13.5453
53.0000   14.0817
54.0000   14.5735
55.0000   14.9311
56.0000   15.2440
57.0000   15.4228
58.0000   15.6017
59.0000   15.7805
60.0000   15.9146
61.0000   16.0487
62.0000   16.1828
63.0000   16.2722
64.0000   16.3169
65.0000   16.4063
66.0000   16.4063
67.0000   16.4510
68.0000   16.4510
69.0000   16.4510
70.0000   16.4510
71.0000   16.4510
72.0000   16.4510
73.0000   16.4957
74.0000   16.4957
75.0000   16.4957
76.0000   16.4957
77.0000   16.4957
78.0000   16.4957
79.0000   16.4957
80.0000   16.5404
81.0000   16.5851
82.0000   16.6746
83.0000   16.8534
84.0000   16.8981
85.0000   16.9875
86.0000   17.0322
87.0000   17.0322
88.0000   16.9875
89.0000   16.9428
90.0000   16.8087
91.0000   16.4957
92.0000   16.0934
```

```
   93.0000   15.6911
   94.0000   15.2887
   95.0000   14.7970
   96.0000   14.1264
   97.0000   13.3665
   98.0000   12.4277
   99.0000   11.4889];

time=[];
u1=[];
n = 100;        % no of readings

for i = 1:1:n
    time(i)= full(i);
    u1(i) = full(i+n);
end
time = time';
u1 = u1';

%% define our meta - variables
duration = size(time); %how long the simulation is
dt = 1;                %continuously looking for measurement

%% define update equations
A = [1 dt;0 1];        %state transition matrix;General form matrices
B = 0;                 %input control matrix;Genral form matrices
u=0;
P = [1 0;0 1];
H = [1 0;0 1];         %genral form matrices ; measuremnet matrix
R = [1 0;0 1]*10^(-3);

k1 = [u1];             %control vector ; acceleration matrix
Q= [0;0];              %initized state ; [position velocity]
Q_estimate = Q;

%% Initialize result variables
Y_p = [];
Y_v = [];

x_estimate_ay = [];    %estimate of the object path using Kalman filter
y_estimate_ay = [];

kal_x_gain_ay =[];
Q_meas_x=[];

%% Kalman Filter for Y
for t = 1:1:duration
% Predict
    % Predicted State
```

```
Ex = [1/3*dt^3 1/2*dt^2;1/2*dt^2 dt]*u1(t);

   Q_estimate_curr = A * Q_estimate + B * u;       %--------------1
    Y_p = [Y_p; Q_estimate_curr(1)];
    Y_v = [Y_v; Q_estimate_curr(2)];

    P = A * P * A' + Ex;                           %--------------2
    Q_meas_x =[Q_meas_x;P];

    K = P*H'*inv(H*P*H'+R);                        %--------------3
    kal_x_gain_ay =[kal_x_gain_ay;K];

    y = H*[Q_estimate_curr(1);u1(t)];
    Q_estimate = Q_estimate_curr + K * (y - H * Q_estimate_curr); %--4

    P =  (eye(2)-K*H)*P;                           %--------------5

% Store for Plotting

    x_estimate_ay = [x_estimate_ay; Q_estimate(1)];
    y_estimate_ay = [y_estimate_ay; Q_estimate(2)];

end

%% Plotting
tt = 1:1:duration;
%x
figure;
grid on
hold on
plot(time(tt),Y_p,'-r.');
plot(time(tt),x_estimate_ay,'-b.');
title('Position - Comparison of Y_p and x_estimate_ay');
xlabel('Time');
ylabel ('Measured Values');
hold off

figure;
grid on
hold on

plot(time(tt),Y_v,'-r.');
plot(time(tt),y_estimate_ay,'-b.');
title('Velocity - Comparison of Y_v and y_estimate_ay');
xlabel('Time');
ylabel ('Measured Values');
hold off
```

REFERENCES

[1] J. Schiller and A. Voisard, Location-Based Services, San Francisco: Elsevier Inc., 2004.

[2] O. J. Woodman, "An Introduction to Inertial navigation," University of Cambridge, London, 2007.

[3] R. Chow, "EDN Network," Epson Electronics America, 01 November 2011. [Online]. Available: http://www.edn.com/design/test-and-measurement/4389260/Evaluating-inertial-measurement-units. [Accessed 03 March 2016].

[4] G. Welch and G. Bishop, An Intoduction to the Kalman Filter, North Carolina: AMC,Inc, 2001.

[5] D. Titterton and J. Weston, Strapdown Inertial Navigation Technology, The American Institute of Aeronautics and Astronautics, 2004.

[6] L. M. Ha, T. D. Tan, N. T. Long and N. D. Duc, "Error Determination of the MEMS IMU," *Journal of Science,* p. 6, 2006.

[7] M. Andrejasic, "MEMS Accelerometers," 2008.

[8] H. Sorenson, "Least-squares estimation: from Gauss to Kalman," *IEEE Journals & Magazines,* vol. 7, pp. 63-68, 1970.

[9] P. S. Maybeck, Stochastic models, estimation and control, Ohio: Academic Press,

1979.

[10] G. Welch and G. Bishop, "The Kalman Filter," [Online]. Available: http://www.cs.unc.edu/~welch/kalman/. [Accessed 20 Februrary 2016].

[11] A. G. Quinchia and G. Falco, "A Comparison of Different Error Modeling of MEMS applied to GPS/INS Integrated Systems: MDPI/journals/sensors," 2013. [Online]. Available: http://www.mdpi.com/1424-8220/13/8/9549. [Accessed 15 June 2016].

[12] Novatel, "IMU Errors and their Effects: Novatel," 21 Februrary 2014. [Online]. Available: http://www.novatel.com/assets/Documents/Bulletins/APN064.pdf. [Accessed 25 July 2016].

[13] P. D. Groves, Principles of GNSS, Inertial and Multisensor Integrated Navigation Systems, London: Artech House, 2013.

[14] M. S. Grewal and A. P. Andrews, Kalman Filtering: Theory and Practice Using MATLAB, Wiely - IEEE Press, 2008.

[15] K. Berntrop, K.-E. Arzen and A. Robertsson, "Sensor Fusion for Motion Estimation of Mobile Robots with Compensation for Out-of-Sequence Measurements," in *11th International Conference on Control, Automation and Systems*, Korea, 2011.

[16] Pires, P. Neto and J. Norberto, "3D position estimation from inertial sensing: Minimizing the error from the process of double integration of accelerations," in *IEEE Conference Publications*, 2013.

[17] F. Orderud, "Comparison of Kalman Filter Estimation Approaches for State Space Models with Nonlinear Measurements," in *Scandinavian Conference on Simulation and Modeling*, SIMS, 2005.

[18] T. Beravs, J. Podobnik and M. Munih, "Three-Axial Accelerometer Calibration Using Kalman Filter Covariance Matrix for Online Estimation of Optimal Sensor Orientation," *IEEE Transactions on Instrumentation and Measurement,* vol. 61, pp. 2501-2511, 2012.

[19] "Extended Kalman Filter," [Online]. Available: http://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL_COPIES/WELCH/kalman.2.html. [Accessed 08 November 2016].