Stabilization Of 3D Pavement Images for Potholes Metrology Using Kalman Filter

Author

ABDULLAH RASHEED

NUST201362512MCEME35513F

Supervisor

Dr. Khurram Kamal

DEPARTMENT OF MECHATRONICS ENGINEERING

COLLEGE OF ELECTRICAL & MECHANICAL ENGINEERING

NATIONAL UNIVERSITY OF SCIENCES AND TECHNOLOGY

ISLAMABAD

JULY, 2017

Stabilization of 3D Pavement Images for Potholes Metrology Using Kalman Filter

Author

ABDULLAH RASHEED

NUST201362512MCEME35513F

A thesis submitted in partial fulfillment of the requirements for the degree of

MS MECHATRONICS Engineering

Thesis Supervisor:

Dr. Khurram Kamal

Thesis Supervisor's Signature: _____

DEPARTMENT OF MECHATRONICS ENGINEERING

COLLEGE OF ELECTRICAL & MECHANICAL ENGINEERING

NATIONAL UNIVERSITY OF SCIENCES AND TECHNOLOGY,

ISLAMABAD

JULY, 2017

Declaration

I certify that this research work titled "*Stabilization of 3D pavement images for potholes metrology using kalman filter"* is my own work. The work has not been presented elsewhere for assessment. The material that has been used from other sources it has been properly acknowledged / referred.

Signature of Student

Abdullah Rasheed

NUST201362512MCEME35513F

Language Correctness Certificate

This thesis has been read by an English expert and is free of typing, syntax, semantic, grammatical and spelling mistakes. Thesis is also according to the format given by the university.

Signature of Student

Abdullah Rasheed

NUST201362512MCEME35513F

Signature of Supervisor

Dr. Khurram Kamal

Copyright Statement

Acknowledgements

Above all else I might want to express gratitude toward God. During the time spent assembling this thesis I understood how genuine this endowment of composing is for me. You give me the ability to have faith in my passion and track my thoughts. I would never have done this without the confidence I have in you, the Almighty.

To my Supervisor Dr. Khurram Kamal: I am stunned! I can scarcely discover the words to express all the astuteness, love and support you've given me. I would express special thanks to him for teaching me Artificial Intelligence and Paradigm of A.I. He is one of the best and will be the best teacher in my life. He is the man of honor who support me in my tough time.

To my Father, Professor Abdur Rasheed Khan: What would I be able to state? You are one of the fundamental reasons that it was GREAT at EME, NUST. I am thankful to the point that I have you in my corner pushing me when I am prepared to surrender. All the great that originates from this thesis I anticipate imparting to you! You are my Buddy and my Hero! Much obliged for not simply accepting, but rather realizing that I could do this! I Love You Always and Forever!

I am in special thanks to Mr. Tayyab Zafar and Mr. Muhammad Usman for their marvelous support and assistance. Each time I stalled out in something, they thought of the arrangement. Without their assistance, I wouldn't have possessed the capacity to finish my first research paper. I value their understanding and direction all through the entire thesis.

In the end, I might want to offer my thanks to everyone and the people who have rendered profitable help to my study.

Devoted to my exceptional guardians and venerated spouse whose enormous help and collaboration drove me to this superb achievement

Abstract

Survey of roads for cracks, holes and any other distress with computerized or robotic imagers are completed on even basis. It is more obvious that moving plate form that capture images produces fluctuations in the image stabilization. Thus, measurements will in fact leads to inaccuracy. This work introduces a complete new approach towards the stabilization of 3D asphalt images. In Kalman filter affine transformation is used in the state space model. Here shaking in the image platform is demonstrated as virtual simulator to estimate the significant translation in the image and correct it via Kalman filter. Later a more robust and dynamic approach with practical implementation has been used with the Extended Kalman filter. A Simulink model is designed for Kalman filter and MATLAB code is written for Extended Kalman filter to demonstrate and to actually implement the technique for the stabilization of images. This work backs to previous effort on the metrology of asphalt images by means of Kinect. Vibration effects and displacement in different potholes with variable ranges has been studied. The Kalman filter is use for pure translation while EKF is use for translation as well as rotation purpose. A substantial difference between Kalman filter and EKF estimation can be seen through error.

Key Words: *UGV (Unman Ground Vehicle), Kinect, Depth Images, Pothole; Color(RGB) images; Extended Kalman and Kalman filter, MPU 6050, Affine Matrices for Transformations;*

Table of Contents

List of Figures

List of Tables

CHAPTER 1: INTRODUCTION

This chapter provides introduction about stability of 3D pavement images when roads are scanned with automatic imagers for distresses. Movement of imaging platform presents flimsiness in pavement images amid picture obtaining process, consequently, gives less precise estimations.

1.1 Problem Statement

Pothole detection is one of the vital errands for the proper planning of repairs and rehabilitation of the asphalt-surfaced pavements. Potholes are critical pieces of information demonstrating structural defects, and precisely recognizing these potholes is one of imperative task for determining an appropriate systems of pavement support and restoration. Despite that, physically identifying and assessing techniques are costly and tedious. In this manner, a few endeavors have been made for building up an innovation which can consequently identify and perceive potholes, which may add to enhance efficiency of survey. But data coming from automatic imagers have instability. This study investigates and analyses pothole detection through automatic scanner which have developed and propose a potential direction of developing an adjustment of 3D asphalt pictures using the Kalman and Extended Kalman filter.

1.2 Background, Scope and Motivation

Generally video recordings are the proper sequence of fames called images by using vibrating or mobile platform. These platforms may be vehicle or any other instrument. After taking video sequence images are then analyzed. However, the trembling of platform is ignored when image analysis is performed. This vibration effects on images introduce numerous instabilities in an asphalt image. This brings in erroneousness and thus, brings forth more treacherous results. Therefore, acquiring video sequence using Microsoft Kinect a 3D stabilization technique is proposed by using the Kalman filter. Here, only pure 3D translation effects are considered. Rotation has non-linear property therefore it is considered in only Extended Kalman filter. State space equation of Kalman filter uses translation and rotation properties of affine transformation and measurement values are input with virtual accelerometer. State space and measurement values

are then used to estimate the rotation and translation of each pixel. Images translated or rotated due to vibration of camera or platform are then stabilized with inverse translation and rotation.

In this regard, Numerous procedures to stabilize video sequence, different systems, techniques and methods have been commenced. triangle based inverse methodology has been proposed to avoid and remove the vibrations effects in consecutive incoming frames, still, the framework cannot influence the movement of moving articles and purposeful movement of panning state of camera. Similarly, to cancel out the rolling-shutter disorientations with Extended Kalman filter and inertial measurement unit (IMU) is designed for mobile phone image stabilization. Camera orientation was only estimated with EKF.

Moreover, image stabilization technique with Kalman filter has been introduced for off road images. This technique maintained the even movements in the images but remove the temporary vacillations in motion. In addition to this, depth camera of Kinect is used to detect an unexpected change in depth and tracking failures. Furthermore, kernel-based technique has been introduced to track pothole. Frames of a video were taken and afterwards calculation is applied. At the point when a pothole is distinguished in an image, the calculation stops to identify the pothole and attempted to track the pothole in the following casings until the point that the pothole leaves the perspective.

1.3 Research Objectives

An automatic road survey and detection of cracks and potholes is very scorching topic nowadays. The expenses of the road upkeep have risen impressively. The damaged asphalt is expanding in a few territories because of the environmental change – substantial snows and rains, wasteful strategies to deplete the water for the roads, and growing traffic also. Stabilization of video for 2D and 3D approaches use a sequence of 2D and 3D transformations. In the order to present the imaging platform movement or vibrations, and remove variations to even these transformations to stabilize the video. Digital picture stabilization method is to steadily evacuate the undesirable rattling phenomena in the picture arrangements with affine transformation and Kalman estimation. Minimal and cost-effective Microsoft Kinect device commonly available now a day. Kinect has two cameras; depth camera provides depth information and second RGB camera captures color pictures.

1.4 Importance

Excess traffic and heavier loads, along with worse weather conditions cause significant corrosion in road materials. These conditions significantly deteriorate a road by forming cracking, localized depressions or potholes. These localized failed areas or cracks not only reduce ride quality, but also provide worse environments for digital imaging. Shaky images provide less accuracy to calculate filling material for a pothole. As result, a complete wastage of filling material. Therefore, digital imaging requires stabilization method to accurately calculate filling material and that's why, the subject has produced substantial interest.

1.5 Potholes Revamping with Image Stabilization

Pothole detection is one of the important task for the proper planning of repairs and rehabilitation of the asphalt surfaced pavements. Existing methods for detection and estimation of potholes usually used sophisticated equipment's and impose computationally intensive tasks. In current practice, sophisticated digital inspection vehicles are used to collect pavement images and video data. Professionals physically check the damage and estimate accordingly. Estimation accurately requires that the incoming video which is the collection of successive images must be stabilize. During image analysis shaking and vibrations of the camera is some time avoided but it has greater impact on the correct calculation of metrology of pothole. Shaky camera or platform may translate, rotate or combine effect of translation and rotation may introduce imprecision and a reduced amount of consistent results. Therefore, the image stability is more important when camera or platform is mounted on a vehicle.

1.6 Research Methodology

For examination and assessment, potholes were imaged using the Kinect sensor utilizing the Open Kinect programming in the Ubuntu Linux condition the metric (mm) depth pictures are utilized for further processing. Matlab is utilized as the processing purpose. Pictures are photographed with Kinect sensor held roughly at 0.8 meters over the ground and imported into the Matlab for further processing. A broad information preparing calculation is composed keeping in mind the end goal to process and concentrate the metrological and in addition portrayal elements of the pothole.

For picture adjustment process the projected procedure utilizes affine matrix with translation and rotation properties in the state space model of Kalman filter for adjustment of 3D depth pictures. The suggested procedure utilizes the calculating centroid of 3D depth picture as a kind of perspective indicate estimate the unadulterated translation and rotation in three axis that are x, y and z.

CHAPTER 2: LITERATURE REVIEW

This section shows the past work done in the field of asphalt pothole image stabilization; moving imaging platforms requires stabilization in any way. Which are detailed in this chapter.

This chapter will give you a review of the research that has been carried out till yet on pothole image stabilization with metrology extraction.

2.1 ADALINE Base Approach

Joon Ki proposes a neural network approach that is use to stabilize a video cameras imaging. It is proposed with edge detection based on ADALINES (adaptive linear neurons). Compression in size and rich zooming makes it essential for cameras to have a video stabilizing system. Digital image stabilization system includes first edge detection second MV motion vector and third zooming with compensation unit of motion. Accuracy can be achieved in detected MV's by obliterating undesirable features like noise in an image, it also reduces calculations by mapping. Where image is presented with 8 bits/pixel. [1]

2.2 DSP (Digital Signal Processing) and Optical Image Stabilization

Uomori write about to developed a full digital image stabilizing system based on motion vectors. This scheme was appropriate for video cameras and VCR's. in this approach, the incoming signal is attached with motion sensor and memory. In motion detector picture is divided into four areas. Motion vector detector facilitates with four areas and correlations matches calculations between continuous fields. the most appropriate vector is chosen by computer for image stabilization. Band extract filter has been developed to reduce enormous calculations. Correlation must be calculated accurately without having higher and lower frequencies because lower frequencies contain flickers and higher frequencies contain noise. These actually interfere with vector calculation. Similarly, if contrast of an image is decreasing it would be difficult to get accurate motion vector. To reduce bad conditions 3 parameters were proposed which are average, minimum correlation and gradient about the lowest point. [2]

Optical image technique was introduced in 1993 by Sato Kochi for a consumer video camera to implement a new approach to stabilize an image. This technique uses feedback servo motors, to control variable, non-linear and uneven fluid prism. This provides a better frequency response. In this paper, authors introduced a new approach using a fluid prism. The system includes the properties, viz:

- Resolution of Picture is intact
- Fluctuation control in optically zoomed pictures

To detect pitch and yaw of video camera two different sensors of angular velocity has been used. These velocities are then supplied to 10 bits ADC. CCD pixel size must be larger than the maximal quantization error. It must be ensured by ADC. The difference of angular velocity signals that is amplitude and frequency, enable microprocessor to distinguishes unexpected fluctuation and use of non-linear integration method single motion vector is calculated. This is called PWM. Passed from lowpass filter that generate signal for the servo motors to control fluid prism movement. [3]

2.3 Global Motion Estimation

In 2002 Vella introduced the adaptive block motion vectors filtering. A decent algorithm for video stabilization. By using block motion vector, motion estimation is achieved. Thus, the similar motion estimator can be used. Detection of undesirable movements, ME motion estimation and compensation is crucial for digital image stabilization. The algorithm has two zones, foreground and background. This all has been completed with a pragmatic technique assuming that frame central zone must have a subject where the background must be close to the border of image. For every part of frame first BM block motion and then motion vectors are estimated. BMV and MVE is use to estimate the global motion for background and foreground. This give an advantage to calculate only one motion vector from different sets of block motion estimation in the order to compute single GMV Global Motion Vector. Instead of calculating two linear histograms only one square histogram is calculated. one for each displacement, vertical and horizontal. It is also considered as a frequent motion vector in the frame. [4]

Nikolaos proposes a different digital image stabilization technique. It estimates global motion in log- polar plane using fuzzy Kalman. For each sub-image single, local motion vector is calculated to extract global motion vector. Motion estimation provides compensation filtered by Kalman. Log-polar transformation is the mathematical representation of human's eyes projections to visual cortex by the retina plane. Its origin is related to vision mechanism. The acceptance of this technique into artificial vision frameworks displays a few favorable circumstances as in visual consideration, rate of throughput and continuous processing. Two-dimensional image displacements are extracted with differential optical flow method. Block matching algorithm is not that accurate, notwithstanding the wrong values are introduced in polar distortion, because of the false grey-value curvature in polar image. The Kalman filter is fed with global motion estimation vector. Kalman system estimate the compensation vector then. Stabilized video is then produced by the information provided by compensation unit. [5]

S. Ertürk in 2004 explains the global motion estimation of phase correlation on sub-image with Kalman estimation to correct motion in images. GM (Global motion) is estimated by detecting local motion in four sub-images using phase correlation. Every single image in the frame from sequence. For phase correlation, fast furrier transform is use for computation, shape of sub-images must be in square shape. Pixel dimensions must be of the power of two. To keep the calculations of motion estimation less, sub image size is restricted to 64x64 and spatial image contents are included for correct estimation. Previous frame sub images based on phase correlation is use to estimate local motion vectors. Amplitude of the most prominent peak against the phase correlation is given as local motion vector in each image. [6]

2.4 AFFINE

Hansen describes in his paper about the pyramid-based technique. A front-end vision system to implement this process. This paper portrays both the calculation and equipment based implementation of the image stabilization system. This approach uses real-time stabilization to eliminate distortions among sequence images. After alignment, image mosaic is produced with a single reference coordinate scheme. A random video frame for stabilization is selected then frames are aligned with one another. Affine transformations are descended to each frame then every frame is aligned with the reference for display. Laplacian is used to estimate affine parameters. Optical

flow is calculated between images via cross correlation. Affine is then use to fit the stream field utilizing weighted minimum squares relapse. Estimated is used to change the prior image to the present image. [7]

Chang in his paper, proposed a system that stabilize video by removing unwanted motions. In this algorithm, optical flow amid sequential frames is calculated the after-camera movement estimation by inserting the calculated visual flow field and trimmed least square TLQ technique to affine. At every single pixel in the frame at instance is the instantaneous motion vector and it is called optical flow. This is important to numerous video capturing study systems; like image mosaic, segmentation of motion and sequence image stabilization. Image in the first place partitioned into reduced P-by-N coincided blocks. camera and object motion is commonly resulted from optical flow vectors. To remove less prevailing object from trimmed least squares method is used to attain strong camera motion. With the T.L.Q algorithm, accumulating the error for all the data points standard derivation is use to find data points having with large errors. These data points are removed for model fitting. [8]

2.5 Bit Plane

Sung introduces a technique base on Grey coded bit plane matching. This technique is more suitable to uneven conditions like moving objects and jitters. Motion estimation is done via coded bit plane. To maintain the accuracy of motion estimation binary Boolean function is use to reduce complex computations. Local motion vector is use to find out global motion vector. To estimate local motion, grey scale image decomposition is introduced via bit plane and grey coded bit plane. Four sub images are placed in grey coded bit plane in a way to estimate local motion. Each vector of sub image in bit plane is calculated by matching of grey coded bit plane with the previous bit plane sub images and decide the optimal match. [9]

2.6 Depth Kinect Sensor

Lui depth information from Kinect sensor is use to stabilize the image. The propose technique solve two hitches; unexpected depth change and tracking failures. Previous techniques are limited with two main reasons. First, these techniques based on homography based frame registration.

Substantial depth changes in image sternly affect homography based frame registration. Second, a homography is restricted to, no camera translation and flat scene to register only two frames. But in real scenario these conditions may not be true for most real videos, and can produce cause sombre distortions results. Additional depth sensor in these two cases is beneficial. With the depth information camera pose, can be estimated accurately by motion estimation between two frames. By using combine colour and depth images to robustly compute 3D camera motion. 2D features points are then matched between two neighbour frames and their depth information is use to estimate relative motion. Cinematography principles are then used to recovered 3D camera trajectories by removing high pitch camera disturbances and low pitch shakes. [10]

Moazzam and K. Kamal develop a technique to study the pavement images using Kinect depth sensor. The system was introduced to study the defects and cracks forms on the road. This was though lacking the image stabilization system but it is use estimate the filler material for the pothole so that there may be a less amount of wastage of material. In the same way, they also study the metrology of different potholes. Trapezoidal rule helped to approximate the volume of pothole. In addition, area, length, and width are also estimated. [11]

Below table will give a batter idea about the literature review with its pros and cons

S. No	Name	Technique	Advantages	Limitations
1	Joon Ki et al [1]	• Edge detection • Motion vector detection • 3x3 window use for mapping • Bistate Adaline use to detect edge • Correlations between two edges with logical operation • Compare first frame with second	• 1 bit requires for pixel intensity • Noise suppression included	• No blur controls • Lack of Image correction • Digital zooming requires extra bits to fill out boundaries
2	Uomori et al [2]	• Signals sent to memory with motion vector detector • Picture divided into 4 areas • Motion vector + correlations matches between the fields • Computer chooses best one	• Best facilitates VCR's+ Camcorders • Avoid high and low frequencies • Avoid bad conditions with average, Minimal and gradient correlation	• Explicit filter Band extract filter was developed to reduce calculations • Very prone to high low frequencies
3	Sato et al [3]	• Servo motors are used for better frequency response • Angular velocity sensors for yaw pitch • Angular velocity signal frequency and amplitude enable MP to respond • PWM generate values for prism	• Keeping picture resolution intact • Fluctuation control in optically zoomed pictures	• Implemented on H/W • Quantization error must be less then pixel size, if not no information will be available
4	Vella et al [4]	• Motion estimation achieved by block matching • It has two main zones foreground and background • For every frame BMV+MV are estimated	• Due to Single MV from BMV only one histogram is to be calculated instead multiple	• Object under consideration must be close to border so that FG and BG are clearly visible

		• Single MV is then calculated from BMV	• This ease the processing and calculation	
5	Nikolaos et al [5]	• Based on Global Motion estimation • 4 local MV computer on sub images to get GMV • Two-dimensional image displacements are extracted with optical method • Block Matching introduces distortion in polar values • Kalman is fed for compensation	• Log polar has an advantage in throughput rate + real time processing	• Limited to only 2D images • BMA reduce the functionality of log polar plane. • Stabilization highly dependent on compensation unit
6	Erturk et al [6]	• Local motions are calculated by using phase correlations to estimate GMV • Images is divided into 4 sub images • For phase correlation FFT used • Kalman is used to correct the amplitude of phase correlations	• FFT reduce the complex calculations • Image size is restricted to 64x64	• Sub images must be in square shape
7	Hansen et al [7]	• Pyramid based image processing • S/W + H/W approach align images into ref coordinate system for image mosaic • Applied affine to each frame • Laplacian pyramid to estimate affine parameters	• Random video sequence can be selected	• Image sequence is very important • Frames must be aligned
8	Chang et al [8]	• Optical flow between frames are calculated • Calculate affine motion with TLS method	•	•

9	Sung et al [9]	• Motion estimation is done via coded bit plane • Grey scale image is decomposed via bit plane and GCBP • 4 sub images are placed in GCBPM to estimate LM	• More suitable to uneven conditions and jitters • Boolean used to reduce calculation	• Grey scale image required • If previous BPM is affected by noise it will automaticall y affect the next one
10	Lui et al [10]	• Kinect depth sensor used • Combine color + Depth images to estimate robust 3d motions • 2d features are matched between connective frames	• Address depth changes when occur • Tracking failure	• 2d features + depth information must be present • If one missing cinematogra phy principals will be failed to implement
11	Mozzam et al [11]	• Used Kinect sensor • depth images were used to analyse the area of pothole • volume is calculated via trapezoidal rule • pothole length, area and width are also calculated	•	• Lack of image stabilization

Table 2-1: Summary Table for Literature Review

CHAPTER 3: TOOLS AND TECHNIQUE

This chapter describes the propose techniques used to achieve the scopes of thesis. The Chapter first section gives brief idea about the tools used in the experimentation and second part explains the technique which stabilizes the video sequence and provide statistical data.

3.1 KINECT

An inexpensive and reduce price 100 dollars Kinect contain Red, Green and Blue (RGB) and infrared (IR cameras). IR camera provides depth while RGB camera assist to distinguish the facial appearance and many more features. This sensor has widespread use in object tracking. Its infrared projector in collaboration with CMOS monochromic sensor gives the room in 3-D style. Kinect sensor camera has some operating features viz.

- IR sensor operates at different frame rates, if frames per seconds' rate changes image resolution changes accordingly.
- Like, at 30 frames per second image resolution would be the IR camera 640 × 480 pixels. similarly, if sensor operate at 10 frames per second rate then image resolution will be higher 1280 × 1024 pixels.
- Practically it is suggested that the image can be taken from 1.8 meters (6 feet). However, testified range vary from 0.8 to 3.5 meters.
- 58° horizontal angle view, 0.8–3.5 m is operating range with 70° diagonal and 45° vertical.
- A tilt motor, an accelerometer, and microphones are built-in.

Strengths of Kinect sensor as compare to other sensors are that it requires less computational efforts, low storage space, less power and cost. In contrast, the sensor measurements may be affected by sunlight, high luminance, shiny metal surfaces, water and dirt.

3.2MPU-6050

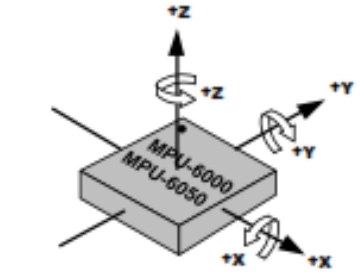


Figure 3.1: Polarity and Orientation of Axes of MPU 6050 [12]

The MPU-60X0 6-axis Motion Tracking device. Where Digital Motion Processor (DMP), 3-axis accelerometer and gyroscope are combined in one package. I2C sensor bus, it accepts direct inputs from 3-axis external compass to provide MotionFusion™ output. The MPU-60X0 is designed for multiple digital sensors, for example pressure sensors. The MPU-60X0 3 16-bit (ADCs) for gyroscope and 16-bits three ADCs for the accelerometer. For precision gyroscope full-scale range vary from ±250 to ±2000°/sec and accelerometer full-scale range is in between ±2g to ±16g.

3.3 Gyroscope Features

The gyroscope in the MPU-60X0 incorporates an extensive variety of components:

- A user-programmable full-scale gyroscope output digitally-X-, Y-, and Z-Axis
- video, image and GPS synchronization is supported by FSYNC pin
- ADCs allow real-time sampling of gyros
- Improved bias and temperature stability improve calibration
- Performance has been improved to reduce noise on low-frequency noise
- Low pass digital programable filter
- 3.6mA low current rating
- 5μA Stand-in current
- Factory adjusted sensitivity factor

3.4 Accelerometer Features

The accelerometer in MPU-60X0 incorporates an extensive variety of components:

- programmable full-scale digital-output with ±2g, ±4g, ±8g and ±16g ratting
- simultaneous sampling requiring no external multiplexer
- 500μA normal operating current
- 10μA current ratting with low power
- Signaling and detection of orientation
- User programed interrupts pins
- Level of G interrupt is High

3.5 Matlab

MATLAB means Matrix Laboratory. It is textural programming environment. It can execute commands directly as typed in the command.

MATLAB Simulink

- Simulink is MATLAB's graphical programming interface.
- Programming is accomplished by connecting various graphical icons in a specific order.
- Simulink graphical icons are collected in what are known as Libraries.
- Simulink programs are known as Models

Model is implemented in SIMULINK. that provides a full block of Kalman filter.

3.6 Kalman Filter

It is an optimum estimator that supposes parameters from inaccurate, incomplete, noisy and inexact observations. It is a recursive model which means that when new input or measurement is available it processed. Optimum means Gaussian noise is only consider in Kalman to minimizes the mean square error. Kalman is use for estimating the linear system. Given the noise mean and standard deviation only for estimation. Dynamic or Non-linear estimators provide improved results.

3.6.1 Kalman Filter Commonness?

- Provide improved results in practice.
- Suitable for direct or online actual or real-world processing.
- Mathematical implementation and formulation is very easy and understandable.
- No inverted equations are requiring for modeling.

3.6.2 Problem Expressing with Kalman Filter

It needs a linear discrete time system describe with vector and additive white noise is use to model undesirable noise or fluctuations.

3.6.3 State Space Definition

The state is the deterministic system which may be a matrix or vector that define the past or previous value of the system. Knowing the state values allows to estimate and predict the system response and outputs the system in the absence of noise.

3.6.4 Representation of State Space

State equation:

$$\acute{m}(\mathcal{I}+1) = F(\mathcal{I})\acute{m}(\mathcal{I}) + B(\mathcal{I})u(\mathcal{I}) + \psi(\mathcal{I}) \qquad \mathcal{I} = 0,1,2,\ldots \tag{3.1}$$

where $\acute{m}(\mathcal{I})$ is the r_x state vector, $u(\mathcal{I})$ is the r_u known input vector, $\psi(\mathcal{I})$white noise but (unknown) with covariance?

$$\xi(\psi(\mathcal{I})\psi(\mathcal{I})' = Q(\mathcal{I}) \tag{3.2}$$

Measurement/ Sensor equation:

$$\breve{y}(\mathcal{I}) = \Pi(\mathcal{I})\acute{m}(\mathcal{I}) + \mathfrak{z}(\mathcal{I}) \quad \mathcal{I} = 1,2,\ldots \tag{3.3}$$

Here $\breve{y}$ is the sensor or measurement matrix, $\Pi(\mathcal{I})$ relates state matrix parameters to measurement, and $\mathfrak{z}(\mathcal{I})$ white measurement noise but having unknown value with known covariance?

$$\xi(\mathfrak{z}(\mathcal{I})\mathfrak{z}(\mathcal{I})' = R(\mathcal{I}) \tag{3.4}$$

Kalman run in two stages one is prediction and second is update. These are given below.

Prediction Cycle:

$$\hat{\acute{m}}(\mathcal{I}+1|\mathcal{I}) = F(\mathcal{I}+1)\acute{m}(\mathcal{I}+1) + B(\mathcal{I}+1)u(\mathcal{I}+1) + \psi(\mathcal{I}+1)$$

Measurement /Sensor Prediction: Time Update

$$\breve{y}(\mathcal{I}+1|\mathcal{I}) = \Pi(\mathcal{I})\hat{\acute{m}}(\mathcal{I}+1|\mathcal{I}) \tag{3.5}$$

Measurement Remaining:

$$\mathfrak{z}(\mathcal{I}+1) = \breve{y}(\mathcal{I}+1) - \hat{\breve{y}}(\mathcal{I}+1|\mathcal{I})$$

State Updated Estimation: Measurement Update

$$\hat{\acute{m}}(\mathcal{I}+1|\mathcal{I}) = \acute{m}(\mathcal{I}+1|\mathcal{I}) + \aleph(\mathcal{I}+1)\mathfrak{z}(\mathcal{I}+1) \tag{3.6}$$

Here $\aleph(\mathcal{I})$ is Kalman Gain.

State covariance:

$$P(\mathcal{I}+1|\mathcal{I}) = P(\mathcal{I}|\tau)F(\mathcal{I})F(\mathcal{I})^T + Q(\mathcal{I}) \tag{3.7}$$

Measurement covariance:

$$S(\mathcal{I}+1) = P(\mathcal{I}+1|\mathcal{I})\Pi(\mathcal{I}+1)^T\Pi(\mathcal{I}) + R(\mathcal{I}+1) \tag{3.8}$$

Filter Gain:

$$\aleph(\mathcal{I}+1) = \Pi(\mathcal{I}+1)^T S(\mathcal{I}+1)^{-1} P(\mathcal{I}+1|\mathcal{I}) \tag{3.9}$$

Updated covariance:

$$P(\mathcal{I}+1|\mathcal{I}+1) = P(\mathcal{I}+1|\mathcal{I}) - \aleph(\mathcal{I}+1)\aleph(\mathcal{I}+1)^T S(\mathcal{I}+1) \quad (3.10)$$

General Table for the Kalman:

Time Update *(prediction)*	Measurement Update *(correction)*
$\hat{x}_k^- = A\hat{x}_{k-1} + Bu_k$	$K_k = P_k^- H^T (HP_k^- H^T + R)^{-1}$
$P_k^- = AP_{k-1}A^T + Q$	$\hat{x}_k = \hat{x}_k^- + K_k(z_k - H\hat{x}_k^-)$
	$P_k = (I - K_k H)P_k^-$

Figure 3.2: Time Update and Measurement Update

3.7 Extended Kalman Filter

This unit, addresses dynamic system which is nonlinear. This system has no extra external input below exactly define a dynamic nonlinear system

$$m_{(d+1)} = a_d(m_d) + \psi_d \quad (3.11)$$

$$z_d = h_d(m_d) + \mathrm{v}_d \quad (3.12)$$

Where,

$$m_d \in \Re^n \; a_d(m_d): \Re^n \to \Re^n \quad (3.12)$$

$$z_d \in \Re^r \; h_d(m_d): \Re^r \to \Re^r \quad (3.13)$$

$$\mathrm{v}_d \in \Re^r$$

$$\psi_d \in \Re^n$$

Where $\{\mathrm{v}_k\}$ and $\{\psi_k\}$ are white Gaussian noises

$$\xi[\mathrm{v}_d \mathrm{v}_d^T] = \Re_d \quad (3.14)$$

$$\xi[\psi_d \zeta_d^T] = Q_d \quad (3.15)$$

$Z_1^d = \{z_{1,} z_2, \ldots, z_{d,}\}$ measurement matrix set. Filter basically use measurement with state space to obtain an estimate

The Computational Origins of the Filter

$$m_{d+i} = \mathrm{m}_{d+i} + A(m_d - \widehat{m}_d) + \psi_d \quad (3.16)$$

Jacobian "A" matrix, where $\partial a_{[l]}$ is the partial derivative with respect to m

$$A_{[l,f]} = \frac{\partial a_{[l]}}{\partial m_{[f]}}(\widehat{m}_d, u_d, \psi_d) \quad (3.17)$$

Jacobian "W" matrix, where partial derivative $\partial a_{[l]}$ with respect to ψ,

$$w_{[l,f]} = \frac{\partial a_{[l]}}{\partial \psi_{[f]}}(\widehat{m}_d, u_d, \psi_d) \quad (3.18)$$

Jacobian "H" matrix, where partial derivative $\partial h_{[l]}$ with respect to m,

$$H_{[l,f]} = \frac{\partial h_{[l]}}{\partial m_{[f]}}(\widehat{m}_d) \quad (3.19)$$

Jacobian "V" matrix, where partial derivative $\partial h_{[l]}$ with respect to v,

$$\mathrm{v}_{[l,f]} = \frac{\partial h_{[l]}}{\partial \mathrm{v}_{[f]}}(\widehat{m}_d) \qquad (3.20)$$

Avoid to mention time step with Jacobean because it has variable time step

A vital element of the EKF is that the Jacobian $H_{[l,f]}$ in the condition for the Kalman gain K_k serves to effectively engender or "amplify" just the significant part of the estimation data. For instance, if no one to one mapping between sensor and state space, the Jacobean will affect the gain of the filter. It will only induce the residual portion and will not affect the state. Then definitely if no one to one mapping is present between state and measurement then the filter will diverge very soon.

3.8 Computing the Derivative

There are two more things to consider:

Computation of the first derivative of an actual signal, without knowing the function of the system.

Generalization of single-valued nonlinear model to multi-value system.

In the order to answer the first question, the first derivative of a function is well-defined as the limit of the difference between consecutive values of that function, divided by the timestep:

$$\xi'(x) = \lim_{\Delta\zeta \to 0} \frac{\xi(x+\Delta x) - \xi(x)}{\Delta\zeta} \qquad (3.21)$$

deducting successive differences of measurement y to approximate its first derivative:

$$\frac{(\ddot{y}_{(n+1)} - y_n)}{timestep\ (\Delta\mathcal{J})} \qquad (3.22)$$

3.9 The Jacobian

A non-linear model is often designed with the jacobian, a matrix whose row size equals to number of measurement inputs or sensors and columns size must be equal to states; yet, the jacobian matrix must have sensor and state current values as well as its partial derivatives This partial derivative is call the Jacobian.

Linear model is as under

$$x_{(\jmath)} = Ax_{(\jmath-1)} + w_\jmath \qquad (3.22)$$

Will becomes

$$x_{(\jmath+1)} = f_\jmath(x_\jmath) + w_\jmath \qquad (3.23)$$

where A matrix is Changed with the Jacobian of the state-transition function f_k. Similarly, H_k is replaced with the h.

In our system

$$f = \begin{bmatrix} 1 & 0 & 0 & \frac{\partial h}{\partial x} \\ 0 & 1 & 0 & \frac{\partial k}{\partial x} \\ 0 & 0 & 1 & \frac{\partial j}{\partial x} \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (3.24)$$

MODEL

$$m_{(d+1)} = a_d(m_d, u_d\,) + \psi_d \qquad (3.25)$$

$$z_d = h_d(m_d) + \mathrm{v}_d \qquad (3.26)$$

PREDICT

$$x_{(d+1)} = a_d(m_d, u_d\,) \qquad (3.27)$$

$$P_d = a_{d-1}P_{d-1}a_{d-1}^T + Q_{d-1} \qquad (3.28)$$

UPDATE

$$B_d = P_d h_d^T(h_d P_d h_d^T + \Re)^{-1} \qquad (3.292)$$

$$m_{(d+1)} = m_{(d)} + B_d(y_d - h_d(m_d)) \qquad (3.30)$$

$$P_{d+1} = (I - B_d h_d)P_d \qquad (3.31)$$

3.10 Transformations

Geometric transformations are used to transform object from one coordinate system to itself. The geometric model experiences change in respect to its MCS (Model Coordinate System). Object represented by point sets are prone to transformation

3.10.1 Fixed Body Motion:

The relative separations between object particles stay steady Affine and Non-Affine maps. Transformed point set ʁ* = л (ϐ, transform object).

When discussing geometric changes, we must be extremely watchful about the object being changed or moved. Here two options can take into account, either the geometric items are changed or the organize framework is changed. These two are firmly related; at the same time, the formulae that do the employment are distinctive.

3.11 Euclidean Transformations

The Euclidean transformation are the most regularly utilized transformation. A Euclidean change is either an interpretation or translation, a turn or rotation, or a reflection. This proposition might talk about interpretations and pivots as it were.

3.11.1 Translations

For 2D translation the only x axis and y axis points are requires

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \qquad (3.32)$$

For translation

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & h \\ 0 & 1 & k \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} x+h \\ y+k \\ 1 \end{bmatrix} \qquad (3.33)$$

Similarly, for inverse translation

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & h \\ 0 & 1 & k \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} x'-h \\ y'-k \\ 1 \end{bmatrix} \qquad (3.34)$$

Similarly, in this fashion it is easy to translate a point in a 3D space. Saying that the point is in the xyz-plane move to new place with the addition of vector <h, k, j>, addition of this vector places the point to new position that is (x', y', z').

Suppose we have equations $q' = q + h, w' = w + k$ and $r' = r + j$. Let consider these points in homogeneous coordinates. That is to say, a column vectors whose fourth component is 1. Thus, the below matrix of point (q, w, r) converts to the following matrix:

$$\begin{bmatrix} q \\ w \\ r \\ 1 \end{bmatrix} \tag{3.35}$$

Then, (x, y, z) and (x', y', z') points are related in the matrix shown below:

$$\begin{bmatrix} q' \\ w' \\ r' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & h \\ 0 & 1 & 0 & k \\ 0 & 0 & 1 & j \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} q \\ w \\ r \\ 1 \end{bmatrix} = \begin{bmatrix} q + h \\ w + k \\ r + j \\ 1 \end{bmatrix} \tag{3.36}$$

In the order to get original values simply put a negative sign in case of translation before the transformation value viz.

$$\begin{bmatrix} q \\ w \\ r \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & -h \\ 0 & 1 & 0 & -k \\ 0 & 0 & 1 & -j \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} q' \\ w' \\ r' \\ 1 \end{bmatrix} = \begin{bmatrix} q' - h \\ w' - k \\ r' - j \\ 1 \end{bmatrix} \tag{3.37}$$

3.11.2 Rotations

2D point for example points in the x-y plane or coordinate is rotated about the origin with some amount of an angle, the relationships can be depicted as:

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\varphi & -\sin\varphi & 0 \\ \sin\varphi & \cos\varphi & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{3.38}$$

Inverse rotation is given as

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\varphi & \sin\varphi & 0 \\ -\sin\varphi & \cos\varphi & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} \tag{3.39}$$

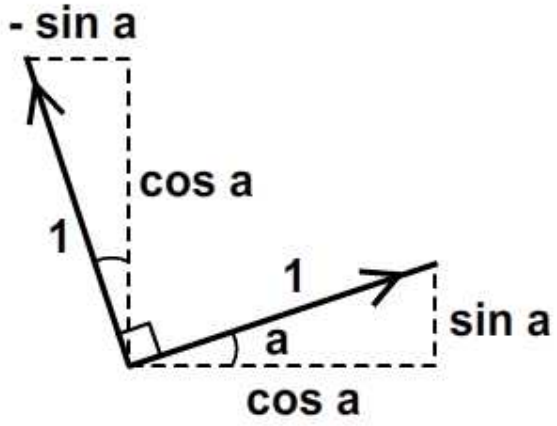


Figure 3.3: Rotation vectors about Origin [13]

It is worth mentioning that in rotation, the magnitude of a unit of both x and y-axis is maintained as shown in the above figure.

In same fashion when a 3D point says (x, y, z) in x-y coordinate is rotated about origin with some amount of an angle the point will become (x', y', z'), 3D rotation has three portions yaw, pitch roll rotations. x-axis rotation is Roll where, y-axis rotation is Pitch about and z-axis rotation is Yaw about. viz:

Roll or x-axis Rotation

$$R_x^{\varphi} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\varphi & \sin\varphi & 0 \\ 0 & -\sin\varphi & \cos\varphi & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.39)$$

Pitch or y-axis Rotation

$$R_y^{\omega} = \begin{bmatrix} \cos\omega & 0 & -\sin\omega & 0 \\ 0 & 1 & 0 & 0 \\ \sin\omega & 0 & \cos\omega & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.39)$$

Yaw or z-axis Rotation

$$R_z^{\delta} = \begin{bmatrix} \cos\delta & \sin\delta & 0 & 0 \\ -\sin\delta & \cos\delta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.39)$$

Where $\omega, \varphi \; and \; \delta$ are the angles of rotations. The complete rotation matrix for moving from the single axis to combine one

$$R_I^B(\varphi,\omega,\delta) = R_x^{\varphi}(\varphi)R_y^{\omega}(\omega)R_z^{\delta}(\delta) \qquad (3.39)$$

Three rotation vectors produces 6 types of matrices having different orders, generally they all are of equal status. It is enlightening to figure the estimations of the six conceivable composite matrices of rotations R and to decide their impact on the world's gravitational field of 1g at first adjusted downwards along the z-axis.

$$R(\alpha,\omega,\delta)\begin{pmatrix}0\\0\\1\end{pmatrix} = R_x^{\varphi}(\varphi)R_y^{\omega}(\omega)R_z^{\delta}(\delta).\begin{bmatrix}0\\0\\1\end{bmatrix} = \begin{pmatrix}-\text{sine}(\omega)\\ \text{sine}(\varphi)\,\text{cose}(\omega)\\ \text{cose}(\varphi)\,\text{cose}(\,\omega)\end{pmatrix} \qquad (3.40)$$

$$R(\omega,\alpha,\delta)\begin{pmatrix}0\\0\\1\end{pmatrix} = R_y^{\omega}(\omega)R_x^{\varphi}(\varphi)R_z^{\delta}(\delta).\begin{bmatrix}0\\0\\1\end{bmatrix} = \begin{pmatrix}-\text{sine}(\omega)\text{cose}(\alpha)\\ \sin(\alpha)\\ \text{cose}(\varphi)\text{cose}(\omega)\end{pmatrix} \qquad (3.41)$$

$$R(\varphi,\delta,\omega)\begin{pmatrix}0\\0\\1\end{pmatrix} = R_x^{\varphi}(\varphi)R_z^{\delta}(\delta)R_y^{\omega}(\omega).\begin{bmatrix}0\\0\\1\end{bmatrix} = \begin{pmatrix}-\text{cose}(\delta)\text{sine}(\omega)\\ \text{sine}(\varphi)\,\text{cose}(\omega)+\sin e(\delta)\,\text{cose}(\varphi)\,\text{sine}(\omega)\\ \cos e(\omega)\,\text{cose}(\varphi)-\sin e(\omega)\,\text{sine}(\delta)\,\text{sine}(\varphi)\end{pmatrix} \qquad (3.42)$$

$$R(\omega,\delta,\alpha)\begin{pmatrix}0\\0\\1\end{pmatrix} = R_y^{\omega}(\omega)R_z^{\delta}(\delta)R_x^{\varphi}(\varphi).\begin{bmatrix}0\\0\\1\end{bmatrix} = \begin{pmatrix}\text{sine}(\varphi)\,\text{cose}(\omega)\,\text{sine}(\delta)-\text{sine}(\omega)\,\text{cose}(\varphi)\\ \text{sine}(\varphi)\,\text{sine}(\delta)\\ \text{cose}(\varphi)\,\text{cose}(\omega)+\text{sine}(\varphi)\,\text{sine}(\delta)\,\text{sine}(\omega)\end{pmatrix} \qquad (3.43)$$

$$R(\delta,\alpha,\omega)\begin{pmatrix}0\\0\\1\end{pmatrix} = R_z^{\delta}(\delta)R_x^{\varphi}(\varphi)R_y^{\omega}(\omega).\begin{bmatrix}0\\0\\1\end{bmatrix} = \begin{pmatrix}\text{sine}(\varphi)\,\text{sine}(\delta)\,\text{cose}(\omega)-\text{cose}(\delta)\,\text{sine}(\omega)\\ \text{cose}(\delta)\,\text{sine}(\varphi)\,\text{cose}(\omega)+\text{sine}(\delta)\text{sine}(\omega)\\ \text{cose}(\varphi)\,\text{cose}(\omega)\end{pmatrix} \qquad (3.45)$$

$$R(\delta,\omega,\varphi)\begin{pmatrix}0\\0\\1\end{pmatrix} = R_z^{\delta}(\delta)R_y^{\omega}(\omega)R_x^{\varphi}(\varphi).\begin{bmatrix}0\\0\\1\end{bmatrix} = \begin{pmatrix}\text{sine}(\delta)\,\text{sine}(\varphi)-\text{cose}(\delta)\,\text{cose}(\varphi)\,\text{sine}(\omega)\\ \text{cose}(\delta)\,\text{sine}(\varphi)+\text{cose}(\varphi)\text{sine}(\omega)\text{sine}(\delta)\\ \text{cose}(\omega)\,\text{cose}(\varphi)\end{pmatrix} \qquad (3.46)$$

It obvious from the Equations 25 to 30 six compound rotations and the six estimations of the deliberate gravitational vector are on the whole extraordinary. A result is that pitch, roll and yaw pivot points are negligible deprived of first characterizing the order wherein these turns are to be connected.

Those rotations which are unsuitable to determin an object orientation, will be rejected instantly. In this regard four of the equations will be removed. The output of accelerometer has three segments yet, since the vector size should constantly measure up to 1g without straight increasing speed, has only flexibility of two degrees. The vector of accelerometer lies on the surface of a circle with covering radius of 1g. It is not subsequently conceivable to unravel for three exceptional estimations that are roll ω, pitch φ and yaw δ. The 4th rotation Equations 27 to 30 bring about the output of accelerometer being an element of every one of the three turn angles and can't in this way be resolved.

Interestingly, the two rotation successions in Equations 25 to 26 depend just on the roll ω and pitch φ and can be explained. The absence of any reliance on the angle of yaw δ is straightforward physically since the primary rotation is in yaw δ around the z-axis of the object. which is at first lined up pointing in downwards in the direction of gravitational field. All accelerometers are totally impervious to rotations in the direction of gravitational field vector and can't be utilized to decide such a rotation.

It is traditional consequently to choose either $R_{\omega\varphi\delta}$ of Equations 25 or the succession $R_{\varphi\omega\delta}$ of Equations 26 to wipe out the yaw δ and permit answer for the roll ω and pitch φ points. The obscure yaw edge δ speaks to the rotation of object from north however its assurance requires the expansion of a magnetometer to make an electronic compass which is outside the extent of this postulation.

Equation 25 can be reshaped as equation 31 to normalize the accelerometer reading G_p

$$\frac{|G_p|}{\|G_p\|} = \begin{pmatrix} -\text{sine}(\varphi) \\ \text{cose}(\varphi)\,\text{sine}(\omega) \\ \text{cose}(\varphi)\,\text{cose}(\omega) \end{pmatrix} \Rightarrow \frac{1}{\sqrt{G_{px}^2+G_{py}^2+G_{pz}^2}} \begin{pmatrix} G_{px} \\ G_{py} \\ G_{pz} \end{pmatrix} = \begin{pmatrix} -\text{sine}(\varphi) \\ \text{cose}(\varphi)\,\text{sine}(\omega) \\ \text{cose}(\varphi)\text{cose}(\omega) \end{pmatrix} \quad (3.47)$$

Understanding the Equation 24 for pitch and roll, utilizing the subscript xyz to indicate that the pitch and roll are registered by the rotation succession R_{xyz} , gives:

$$\tan \omega_{xyz} = \left(\frac{G_{py}}{G_{pz}}\right) \tag{3.48}$$

$$\tan \varphi_{xyz} = \left(\frac{-G_{px}}{G_{py}\sin(\alpha)+G_{pz}\cos(\alpha)}\right) = \frac{-G_{px}}{\sqrt{G_{py}^2+G_{pz}^2}} \tag{3.49}$$

Order $R_{\omega\varphi\delta}$ used in the aerospace industry therefore, it is also called aerospace rotations. Similarly, the calculation of finding tilt angle is given as:

$$G_p.\begin{pmatrix}0\\0\\1\end{pmatrix} = G_{pz} = \left|G_p\right|\cos(\delta) = \frac{G_{pz}}{\sqrt{G_{px}^2+G_{py}^2+G_{pz}^2}} \tag{3.50}$$

3.12 Affine Transformations

In numerous imaging system, identified pictures are liable to geometric distortion presented by point of view abnormalities wherein the position of the camera(s) as for the scene changes the clear measurements of the scene geometry. Applying a relative change to a consistently distorted picture can amend for a scope of point of view bends by changing the estimations from the perfect directions to those really utilized.

An affine tranform is a vital class of straight 2-D geometric changes which maps factors (e.g. intensity of a pixel is situated at position (x_1, y_1) in an information picture) into new factors (e.g. (x_2, y_2) in an output picture) by applying a direct rotation and translation operations.

Similarly, instead 2D; transformation can also be applied in the same way to 3D space vectors . This is accomplished by combining of transformations to a single combined matrix that shows all the transformation. Here sequence among transformation must be maintained. A combined matrix is

$$[\mathbb{T}][\mathfrak{M}] = [X][\mathbb{T}_1][\mathbb{T}_2][T_3] \dots [\mathbb{T}_{n-1}][\mathbb{T}_n] \qquad (3.51)$$

Where $[\mathbb{T}_n]$ are any combination of

- Translation
- Rotation
- Scaling
- Shearing
- Reflection

For combining translation and rotation into a single matrix, where 3-by-3 matrix in the upper-left the $\mathfrak{R}$'s is rotation and h, k and j are the translation vector values. This matrix constitutes a composite matrix for rotations and translation.

$$[\mathfrak{M}'] = \begin{bmatrix} \mathfrak{R}_{(r1,c1)} & \mathfrak{R}_{(r1,c2)} & \mathfrak{R}_{(r1,c3)} & h \\ \mathfrak{R}_{(r2,c1)} & \mathfrak{R}_{(r2,c2)} & \mathfrak{R}_{(r2,c3)} & k \\ \mathfrak{R}_{(r3,c1)} & \mathfrak{R}_{(r3,c2)} & \mathfrak{R}_{(r3,c3)} & j \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot [\mathfrak{M}] \qquad (3.52)$$

Length and angle of measurement is preserved in Euclidean transformations. Additionally, it has no effect on the shape of the geometric object. It means that if there is a line it will transfom to line, in similary way planes, circles and ellipsoid will transform to planes, circles and ellipsoid with no distortion in the shape of an object. The only thing that will change is the position and the orientation. Euclidean transforms are generalized in affine with slight change that is line tranformation to line but circle changed to ellipse with no preservation of angle and length.

3.13 AFFINE in General

The following matrix below is the general affine transformation:

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} \Phi_{(r1,c1)} & \Phi_{(r1,c2)} & \Phi_{(r1,c3)} & \Phi_{(r1,c4)} \\ \Phi_{(r2,c1)} & \Phi_{(r2,c2)} & \Phi_{(r2,c3)} & \Phi_{(r2,c4)} \\ \Phi_{(r3,c1)} & \Phi_{(r3,c2)} & \Phi_{(r3,c3)} & \Phi_{(r3,c4)} \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{3.53}$$

Earlier discussed matrices including translations and rotations one can perceive that all the these matrices can easly fit into (3.53) and thus it is called an affine transform matrix.

As discussed transformations perfomed with affine not really modify the polynomial degree such as parallel, intersecting planes as well as lines are transformed to parallel, intersecting lines/planes. In contrast, transformation perfomed through affine failed to keep the angles and lengths and this lead to change in geometric object shape.

To find the inverse Affine transformation matrix only take the inverse of transformation matrix or transpose in MATLAB. The below equation gives the idea.

$$\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} \Phi_{(r1,c1)} & \Phi_{(r1,c2)} & \Phi_{(r1,c3)} & \Phi_{(r1,c4)} \\ \Phi_{(r2,c1)} & \Phi_{(r2,c2)} & \Phi_{(r2,c3)} & \Phi_{(r2,c4)} \\ \Phi_{(r3,c1)} & \Phi_{(r3,c2)} & \Phi_{(r3,c3)} & \Phi_{(r3,c4)} \\ 0 & 0 & 0 & 1 \end{bmatrix}^{-1} \cdot \begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} \tag{3.54}$$

3.14 Test Rig

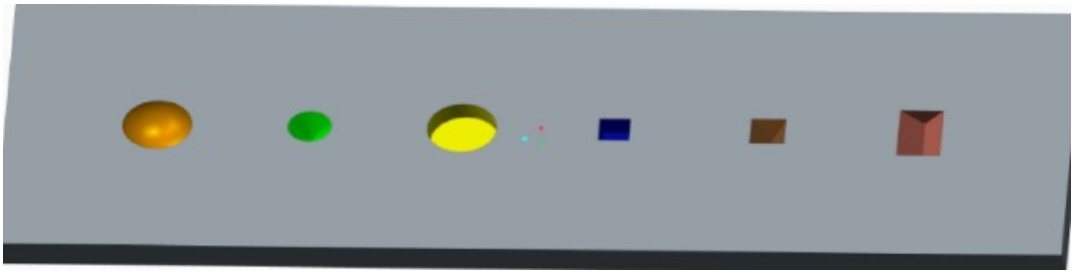


Figure 3.4: Test Rig for Experimentation

This test rig is made to actually implement the model after successful implementation on simulation. Simulink MATLAB application is use to implement the simulation model on virtual accelerometer. The same model of Kalman will then implement on the test rig which is a benchmark for testing the pothole metrology.

This test rig includes Hemisphere, Prism, Pyramid, Cone, Cylinder and Cube. The above figure demonstrates the model of test-rig is designed with CAD. Each item on the rig have its own specifications.

Rig total length: 6.5 feet

Rig total width: 2.5 feet

S.No	Name	Max Depth in mm	Max Volume in cm^3
1	Hemisphere	40	150
2	Prism	78	545
3	Pyramid	100	340
4	Cone	81	403
5	Cylinder	80	950
6	Cube	78	780

Table 2-1: Original depth and Volume of the Rig potholes

Test rig provides a balance condition for the calibration of accelerometer and gyro sensor as the rig is place on a smooth and clean surface. It also provides a template to compare the sensor values with the actual depth and volume values of each object on the rig.

CHAPTER 4: EXPERIMENTATION AND RESULTS

At the current position now we have enough knowledge to formulate the problem into a model. Images of the pothole has been captured at college of EME, NUST Rawalpindi, Pakistan through Microsoft Kinect, for the examination and analysis of pothole in the evening. Excess light passes through Kinect sensor infrared camera cause less accurate image depth that why to avoiding excess exposure of light evening has been chosen. Sensor has taken images from 0.8 m of height from the ground level. As Kinect sensor may be affected by direct sunlight, therefore, sensor is sheltered with a covering to reduce the direct sunlight interference. Depth resolution of the images was set to 640 by 480 pixels at the time of shooting. Setup of experimentation is shown in figure below. OpenKinect library for acquiring images with Kinect under Ubuntu operating is used. MATLAB was used to further analyze the images taken.

Figure 4.1: Test Rig with Kinect and accelerometer are mounted on UGV for Experimentation

The image sequence taken with the Kinect is given below with image on the left is depth image and on the right side RGB image.

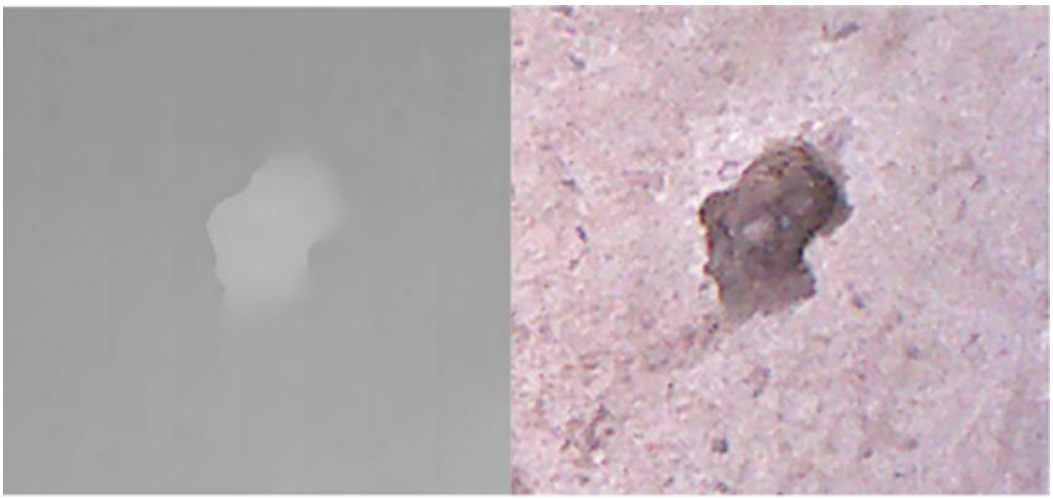


Figure 4.2 : Pothole on the ground for study

Stabilization of 3D asphalt/ pavement images Kalman filter with affine transform approach is suggested. At this stage, pure translation is considered in 3D. Below image gives the idea of the proposed technique.

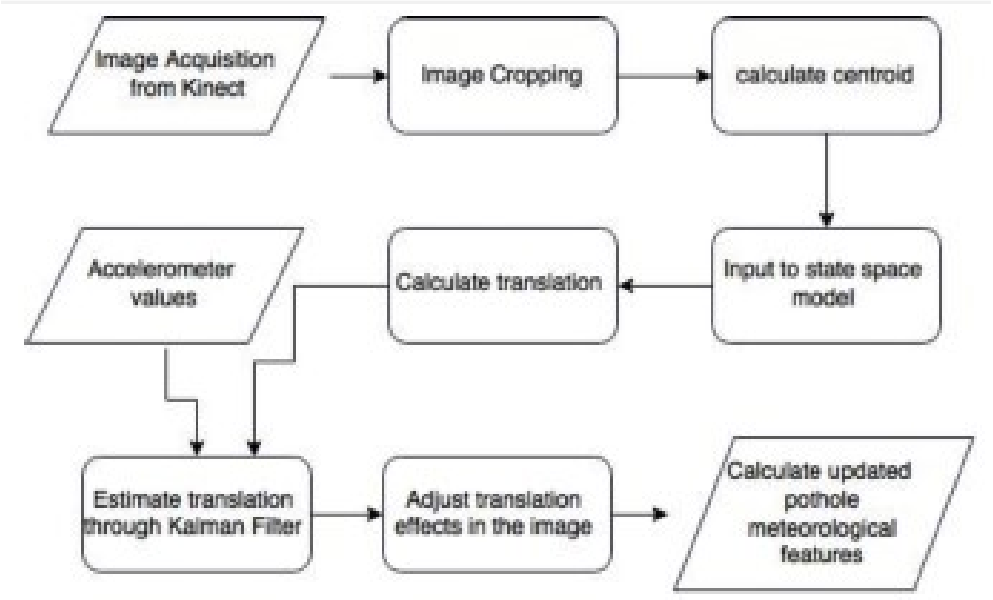


Figure 4.3: Flow algorithm for calculating KF estimation

Acquisition of image frame through Kinect camera has resolution 640 x 480, therefore, it is cropped to 200×200. By introducing the virtual accelerometer random values to translate the images. As shown in the figure below

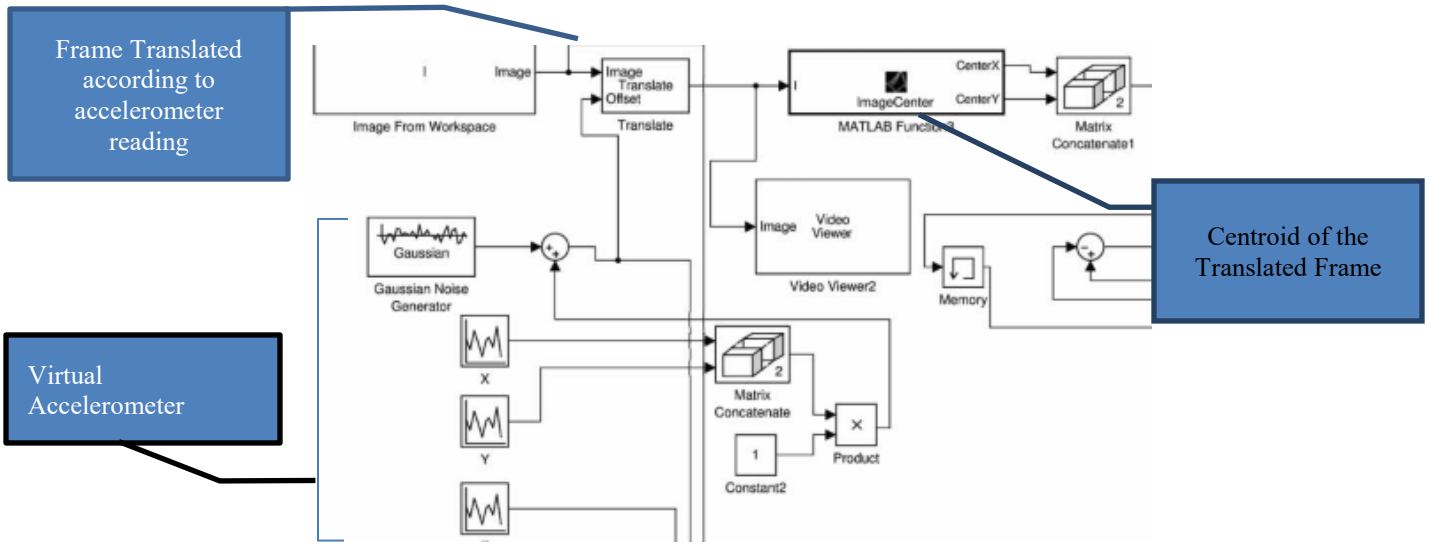


Figure 4.4: Virtual Accelerometer and Image Centroid System in Simulink

Here previous frame is translated to next location in the x and y coordinate. Now the reading coming from the accelerometer is fed into Kalman state space model, after estimation these values will be use to translate the previous frame and find the centroid of the image.

By using the above Equation 37 of Affine translation for 3D to form a state space model for Kalman. State space is the mathematical representation of the problem.

$$\mathbb{F} = \begin{bmatrix} 1 & 0 & 0 & T_x(p) \\ 0 & 1 & 0 & T_y(p) \\ 0 & 0 & 1 & T_z(p) \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.1}$$

Where, translation property of affine is shown in matrix $\mathbb{F}$ as a state space model, $T_i(p)$ is the translation in between the consecutive frames and p is the incremental index of an array. Here left 3 x 3 matrix of Affine transformation matrix is identity because of the fact that the pure translation is considered. Rotation is ignored with one other fact that the rotation model is dynamic.

State space of Kalman block in Simulink is acquainted with the difference in the translation of two frames. The translation in the direction of x, y and z are estimated via Kalman by introducing measurement and the state space values. As shown the figure

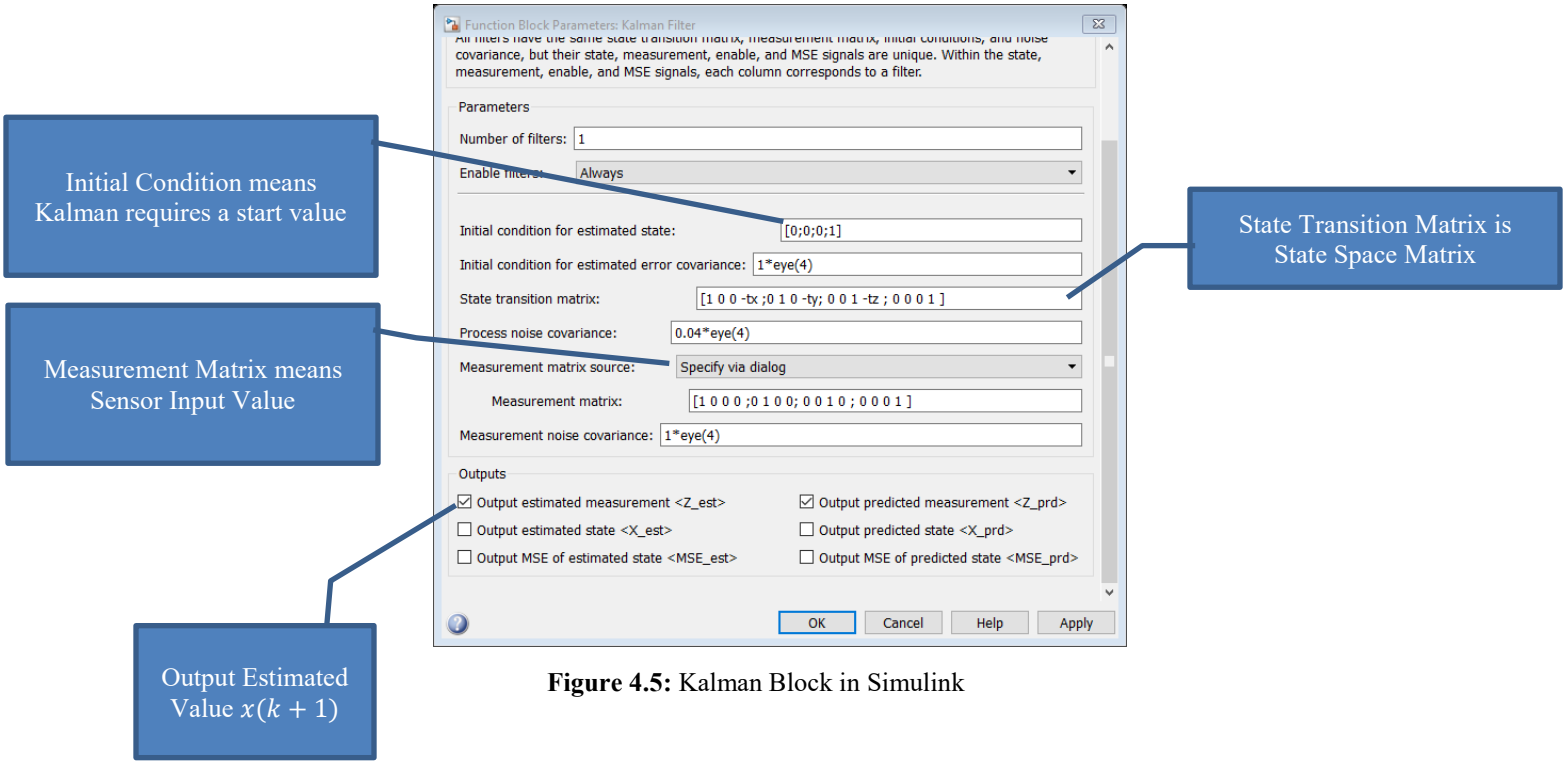


Figure 4.5: Kalman Block in Simulink

Then the centroid of the two images or frames are calculated so to find the translation between the two. The difference between the two will gives pure translation.

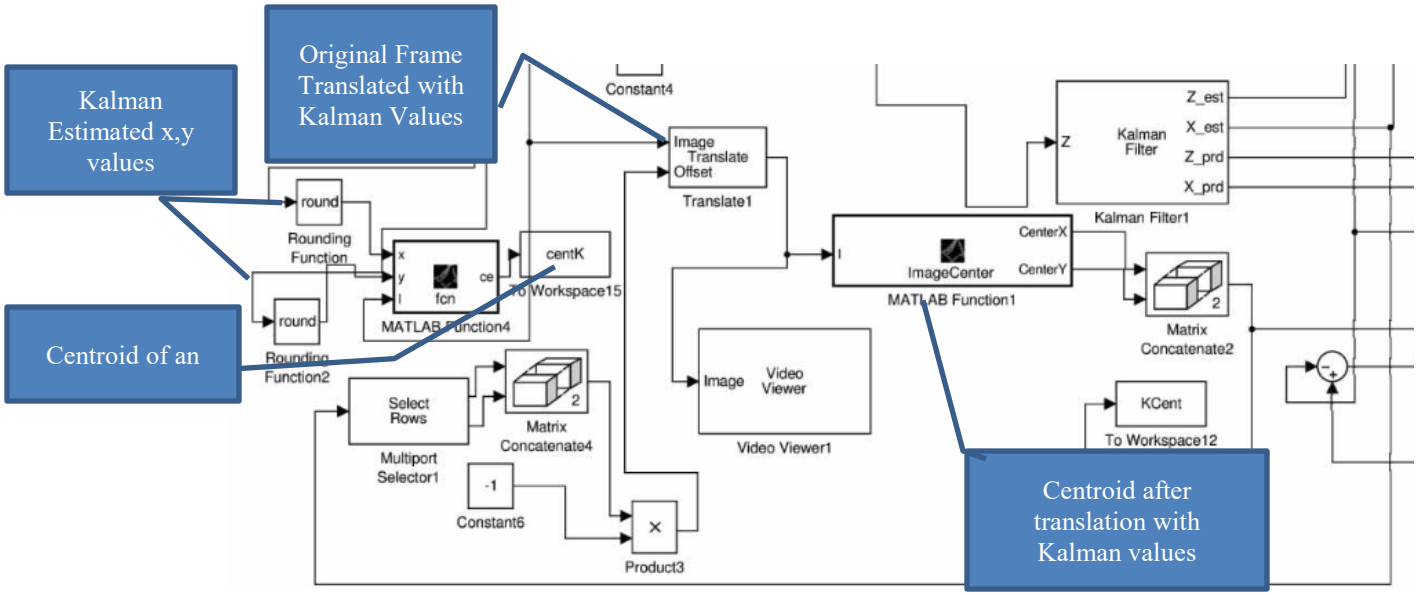


Figure 4.6: Image Translation and Correction System in Simulink

The stabilization of image is achieved from the estimated values of Kalman filter. These values translated the frame close to original one as shown in above figure.

Depth images constitute to determine the metrology (perimeter and volume) of pothole. The algorithm to achieve this is given below.

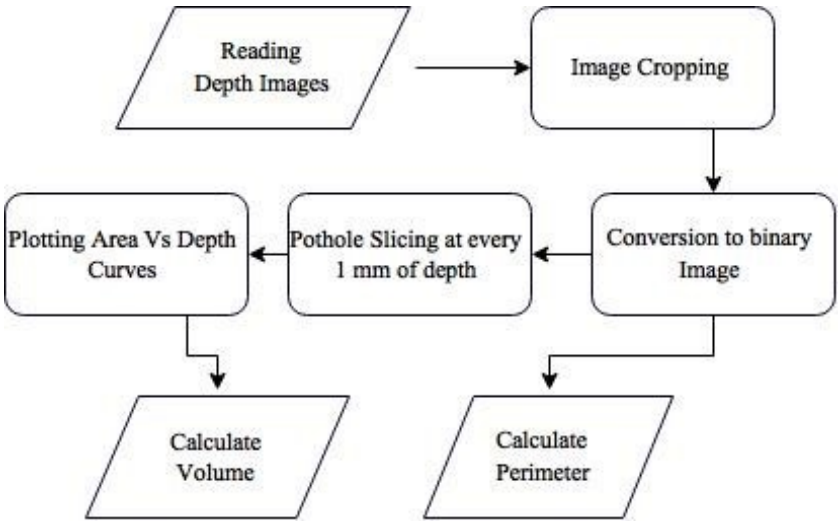


Figure 4.7: Algorithm for calculating the Volume and Perimeter

Depth is calculated after binarization of each frame at every single millimeter. the number of white pixels are calculated for it. Using these area values, volume is calculated by using the area depth graph by accumulating the curve area. These binary pictures with are used to measure the peak depth and perimeter. Full simulation model is given below.

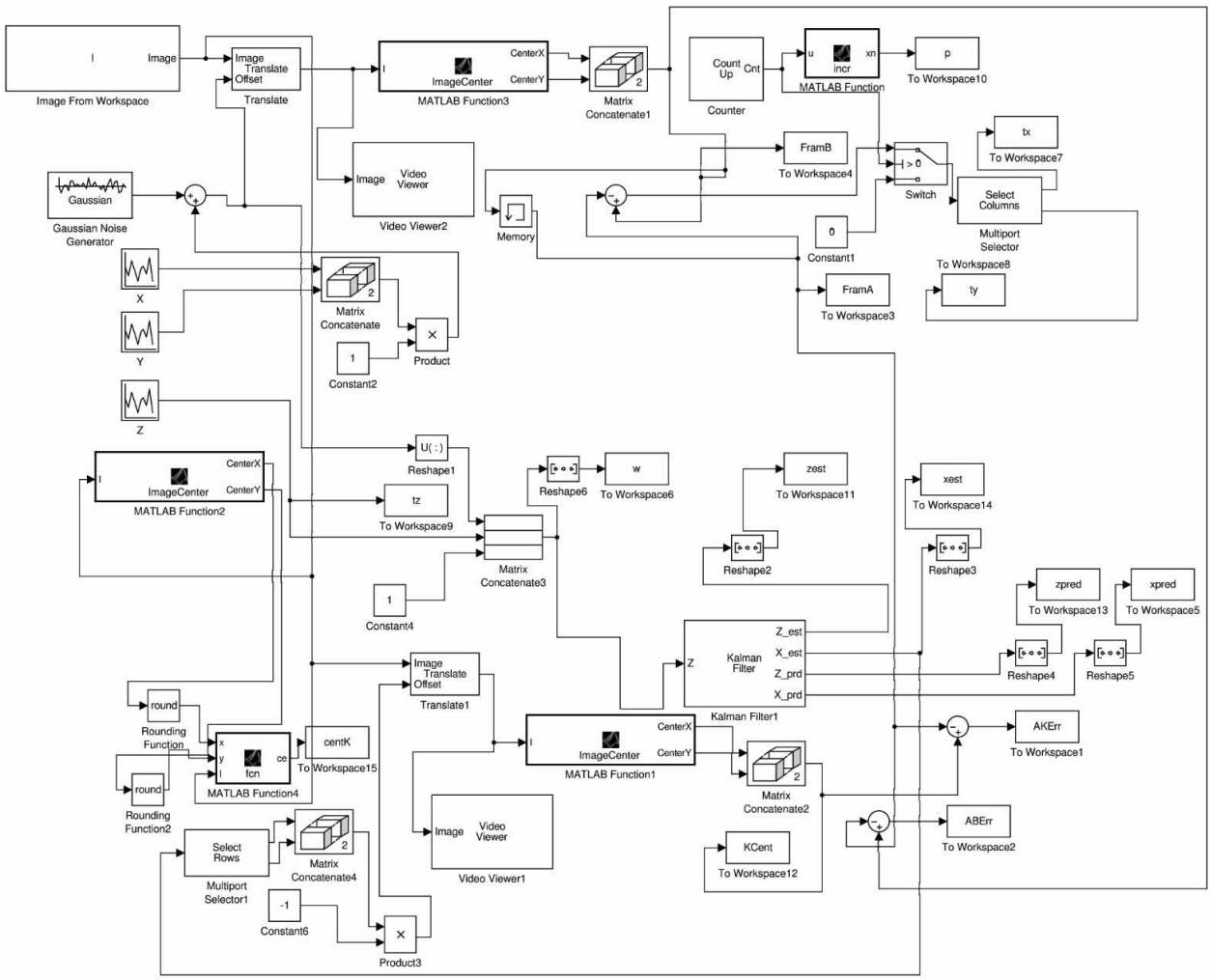


Figure 4.8: Full Estimation and Prediction Model in Simulink

4.1 Modeling via Simulation

Depth images are recorded in a video sequence as shown in the below. Depth image on the right and color RGB image on the left.

Figure 4.9: RGB image of Pothole

Figure 4.10: Depth Image of Pothole

Kalman filter uses value coming from sensor and update the estimate via previous value from the model. Therefore, it is implemented through MATLAB software by using an arbitrary linear movement in each frame in the direction of x, y and z. This simulation has intended to keep the noise zero. white Gaussian noise and variance values are set to 0 and 1 for measurement in the model. Process noise is assumed to be 0 because system for estimation is consider to be noise free, and after simulation process noise is calculated by the model itself. The estimated value coming from Kalman by using the state space model is use to translate the image and find its centroid. Conversely, the sensor/ measured signals/ values are simulated random values that are coming from virtual accelerometer.

4.2 Results

All the values coming from sensor, prediction or estimation using Kalman filter are in millimetre. These values are considered for translation in each frame in x, y, and z plane. All the figures below showing the red line which is measured/ sensor mean values. Green line shows the predicted values. Whereas the estimated values are shown via blue line.

It is obvious from the graphs that the difference in sensor/measured values and the estimated values are because of sensor noise. As the Kalman state space model has no noise that's why it estimates and predict the values near original values if noise is subtracted from the sensor value.

In contrast, estimated values using Kalman lies amidst the red and green values. In other figures estimated and predicted values are close to each other or overlaps. The fact is that the Kalman filter is converging. If the gap become larger and larger with the increment in time then the filter is not designed accurately.

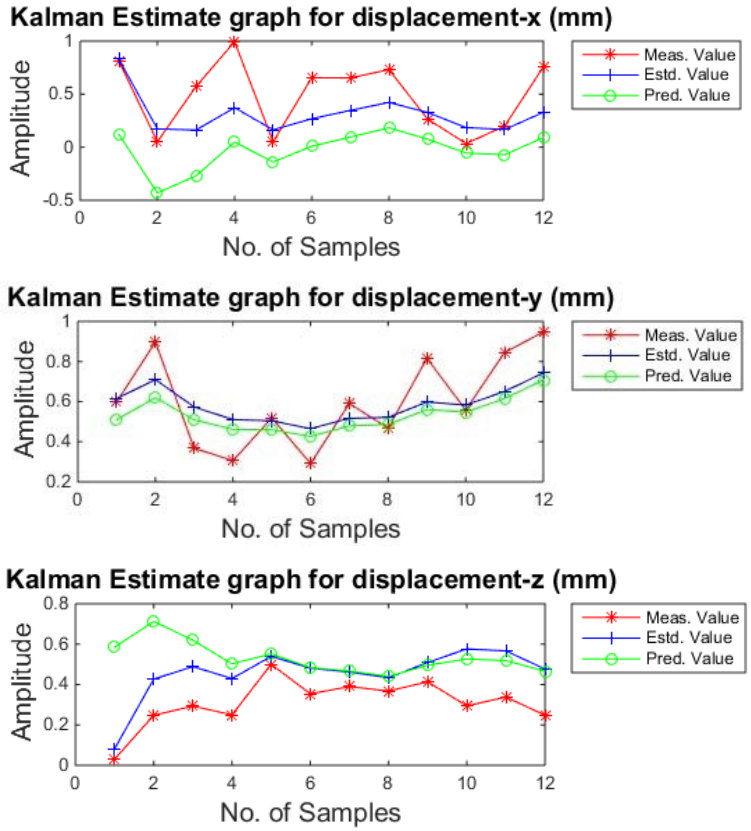


Figure 4.11: Sensor, Estimated and Predicted Kalman Graphs

As compare with x, y and z graphs has faster convergence. The possible reason is that the estimated values in y and z direction are closer to sensor values.

Image on the left side is the difference between original/reference frame and the next or translated frame. The image on the right is the difference between original/ reference frame and the translated image through Kalman estimation.

Figure 4.12: Original and Measured difference

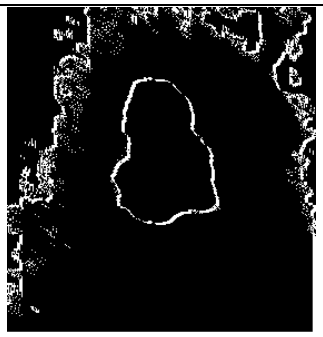


Figure 4.13: Difference of Original and Estimated

It is purely visible from the above picture that thick white line in left image has higher difference where picture on the right has thin line, means that the difference between the reference and deformed frame is less. Perimeter, volume and maximum depth calculated after stabilization of the frames. The table below show the values

	Displacement(mm)	original image	Translated Image	% Error (Translated Image) E_{TI}	Stabilized Image	% Error (Stabilized Image) E_{SI}	$\Delta E=E_{TI}-E_{SI}$
Volume	0.2	38.26	39.86	4.02	39.02	1.96	2.06
Perimeter		110.11	105.32	4.55	107.94	2.01	2.54
Volume	0.4	38.26	40.06	4.49	39.60	3.39	1.10
Perimeter		110.11	114.52	3.85	106.72	3.18	0.67
Volume	0.6	38.26	40.50	5.52	40.06	4.49	1.04
Perimeter		110.11	114.27	3.64	114.52	3.85	0.21
Volume	0.8	38.26	40.50	5.53	40.06	4.49	1.04
Perimeter		110.11	119.70	8.01	114.52	3.85	4.16
Volume	1	38.26	41.23	7.21	40.06	4.50	2.71
Perimeter		110.11	118.19	6.83	113.00	2.56	4.27
Average Volume Error		-	-	**5.38**		**3.09**	**2.29**
Average Perimeter Error		-	-	**2.37**		**1.59**	**0.78**

Figure 4.14: Volume, Perimeter displacement with original and translated values. Average Errors are also visible

4.3 Summary

The technique proposed via Kalman demonstrated improved results for image 3D pothole stabilization. The error in volume calculations are significantly reduced from 5.38% to 3.09% for volume. Similarly, error in perimeter calculations are reduce from 3.27% to 1.59% respectively.

CHAPTER 5. PROBLEM FORMULATION WITH EKF

After successful implementation of simulation in SIMULINK now real reading has been taken by using the Kinect sensor. In addition to this, an accelerometer with gyroscope is used to take the reading from 0.8 mm of distance from the ground. Full setup is implemented on the UGV which is controlled with remote control.

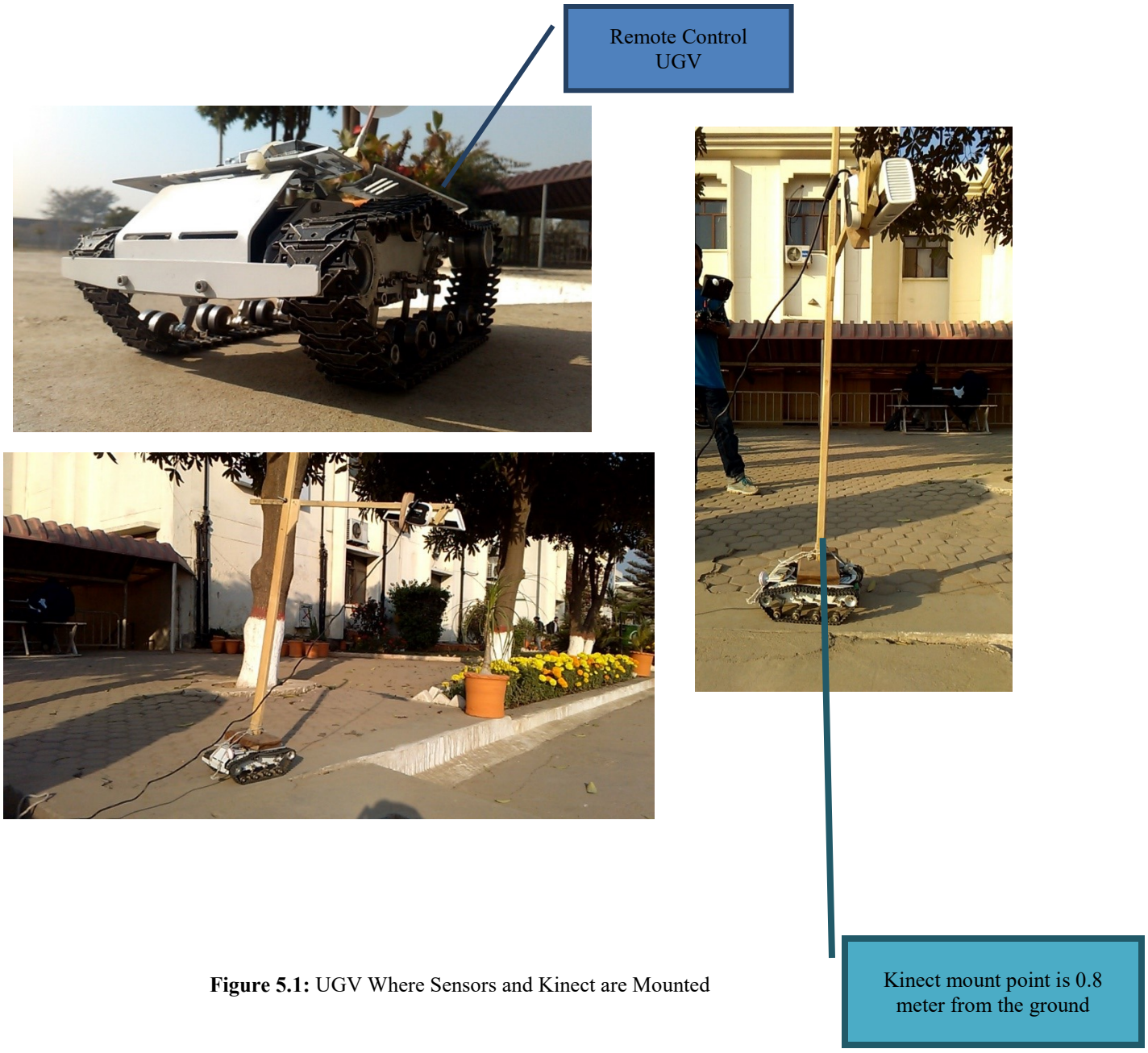


Figure 5.1: UGV Where Sensors and Kinect are Mounted

Complete Flow Diagram of Stabilization System

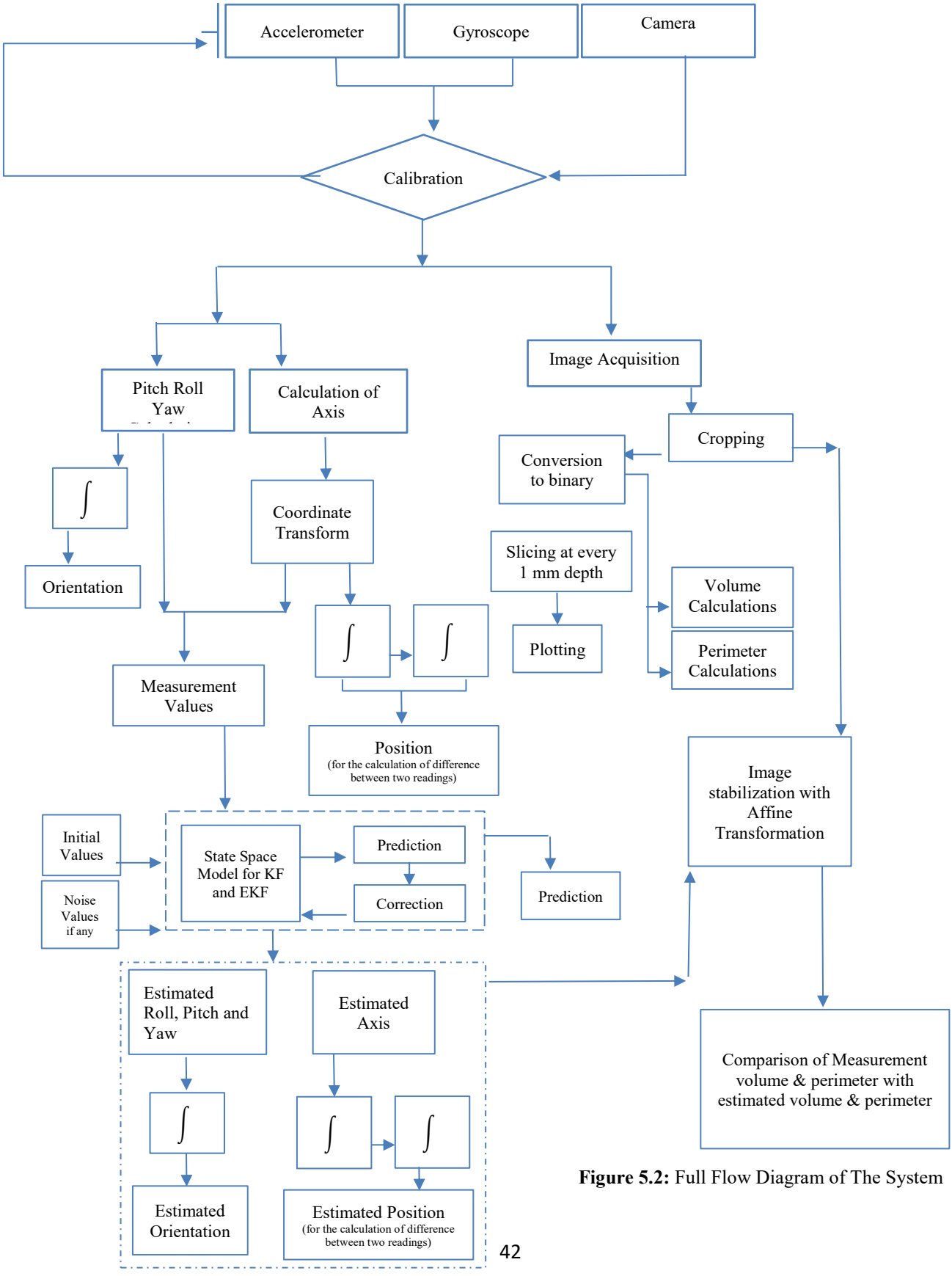


Figure 5.2: Full Flow Diagram of The System

Accelerometer and gyro reading is taken for every 1mm. UAV speed is kept constant at the rate of 1meter per minute. Microcontroller code is use to convert the coming voltages from the sensor to meaningful data. Kinect camera is taking images with rate of 25 frames per seconds. Sampling rate for the controller is

$$Sample\ Rate = \frac{1}{(1+Sampler\ Divider)}\ KHz \qquad (5.1)$$

Sampler Divider is frame rate at which frames will be captured per seconds.

The above equation 64 will give us sampling rate for microcontroller 0.03846 KHz. Baud rate for the serial communication is set to 115200.

5.1 Pseudo Code for Microcontroller

```
{
        Call wake ( );
        Begin transmission ( MPU 6050);
        Delay (100);
        Write  register ≈ 0b000000;
        Close( );
}
Call MPU Calibration( );
MPU_ReadData( );
        Set xangle ⟶ angle_roll_gyro_x;
        Set yanlge ⟶ angle_pitch_gyro_y;
        Set zangle ⟶ angle_yaw_gyro_z;
Loop
{
        ReadData( );
        Set xangle  ⟶ (angle_roll_x+current_gyro_x)/2
        Set yangle  ⟶ (angle_pitch_y+current_gyro_y)/2
        Set zangle  ⟶ (angle_yaw_z+current_gyro_z)/2
    }
```

Similar for accel

Repeat the loop;

Offset calibration of the controller is very important. Reading coming from accelerometer and gyro must be set to final value by removing the offset. Calibration has one more importance when there is rough terrain.

Call wire.h;

Define time ⟶20;

Define radian to degree ⟶ 57.3

Setup ( )

Begin( );

Call wake( );

Setgains( );

Set offset( );

Loop

Gyro angle at x ⟶ x_value_ scaled(time/1000)+anglex);*

Gyro angle at y ⟶ (y_value_ scaled(time/1000)+angley);*

Gyro angle at z ⟶ (z_value_ scaled(time/1000)+anglez);*

calculate angle with tan( );

theta ⟶a2tan( );

set filter gain( )

end

serial out all variable⟶?

5.2 Matlab Code for Accessing Microcontroller

Call Controller Function

Define COM3 port

Define baud rate 115200

Open port for Arduino( );

Readsync( );

Loop

    Check for data( );

End

Loop

    Concatenate different inputs( );

    Check Arduino buffer status( );

5.3 Implementation With K.F

In the order to implement the stabilization on images Kalman model is once again implemented in Simulink. Related images with reading is also implemented with the Extended Kalman filter. This for the purpose to find the difference between the two approaches.

In first place, Kalman filter is implemented in SIMULINK with the reading coming from the microcontroller. First reading of microcontroller has been taken via .m file KINECT.m.

Procedure

- Reading of x, y and z for accelerometer and gyroscope is fed into SIMULINK via variable `sensor_value`
- Initial values for Kalman in x,y and z direction are also put into SIMULINK `InitFcn`
- SIMULINK model parameter configuration is set to Fixed step start from 0 and end time 15. (here 15 means maximum samples or readings). Solver is set to `ode1 (Euler)`
- Kalman block is set according to state space model matrix with giving initial value. Initial value is the average value of first few readings.
- State space modelling and transformation matrices are discussed on pages 14 and 19
- System upon running for fifteen values will take sensor reading from `InitFcn`
- The estimated and predicted values are then store is respective variables

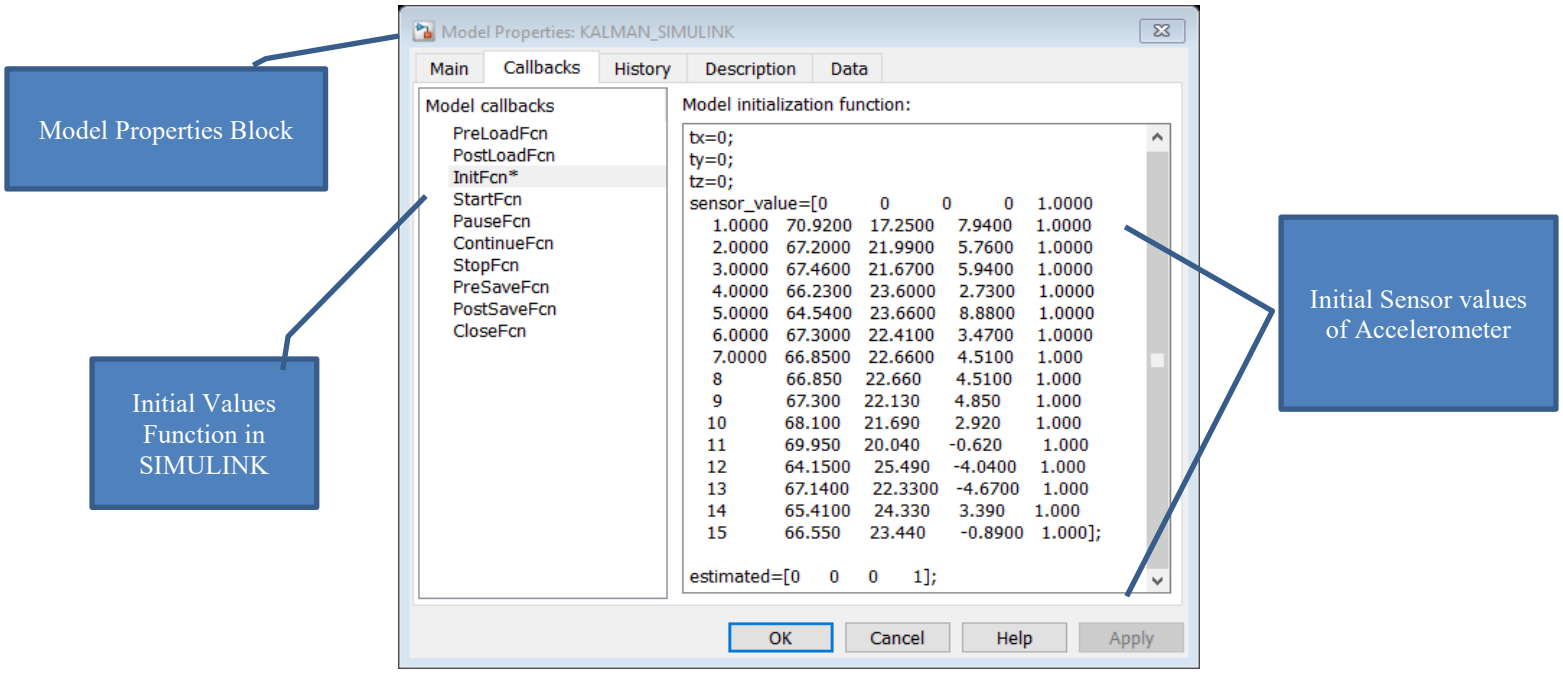


Figure 5.3: Initial Values of Accelerometer for Estimation and Prediction Model in Simulink

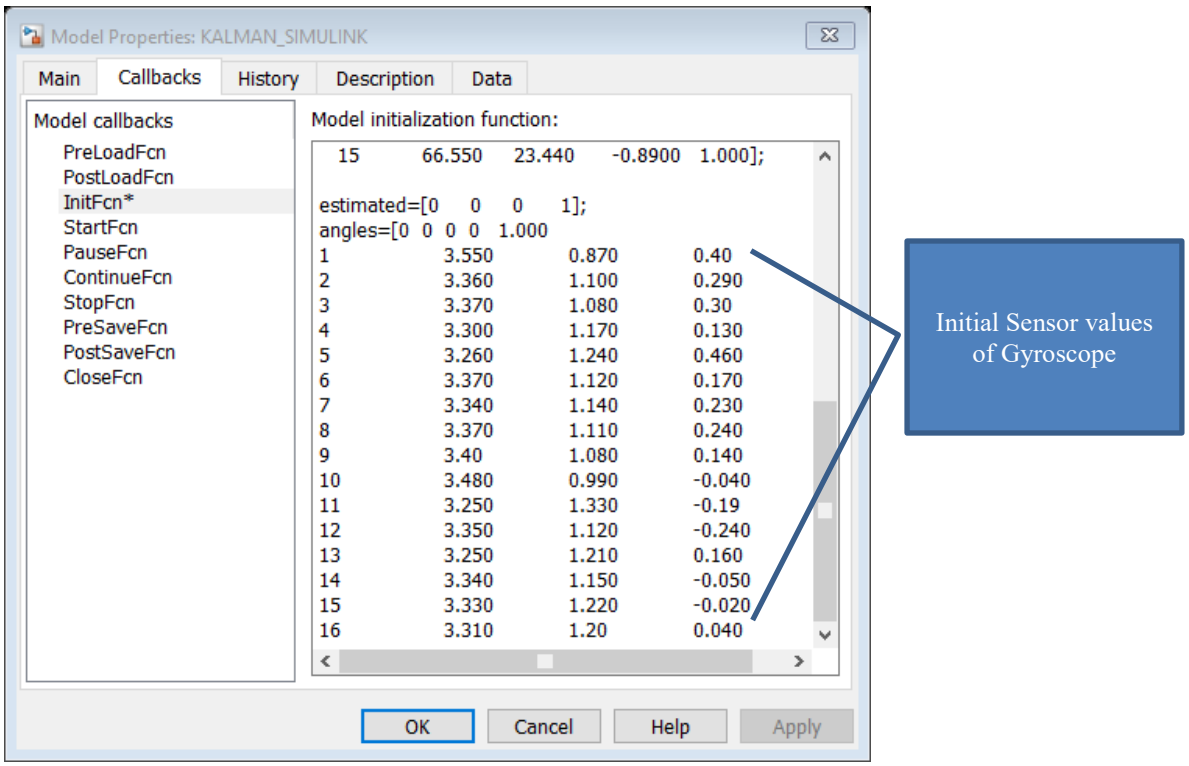


Figure 5.4: Initial Gyro values for Estimation and Prediction Model in Simulink

- When sensor send values accordingly they are compared with previous reading. Difference is required to put into Kalman state space model.
- The difference between two consecutive readings shows that how much translation or rotation has been occurred
- In real values coming from sensors must have a noise and have large difference.
- In addition to this, estimated values coming from Kalman will reduce the error significantly

After initial values, model is set to estimate and predict the sensor values. As shown in the figure below

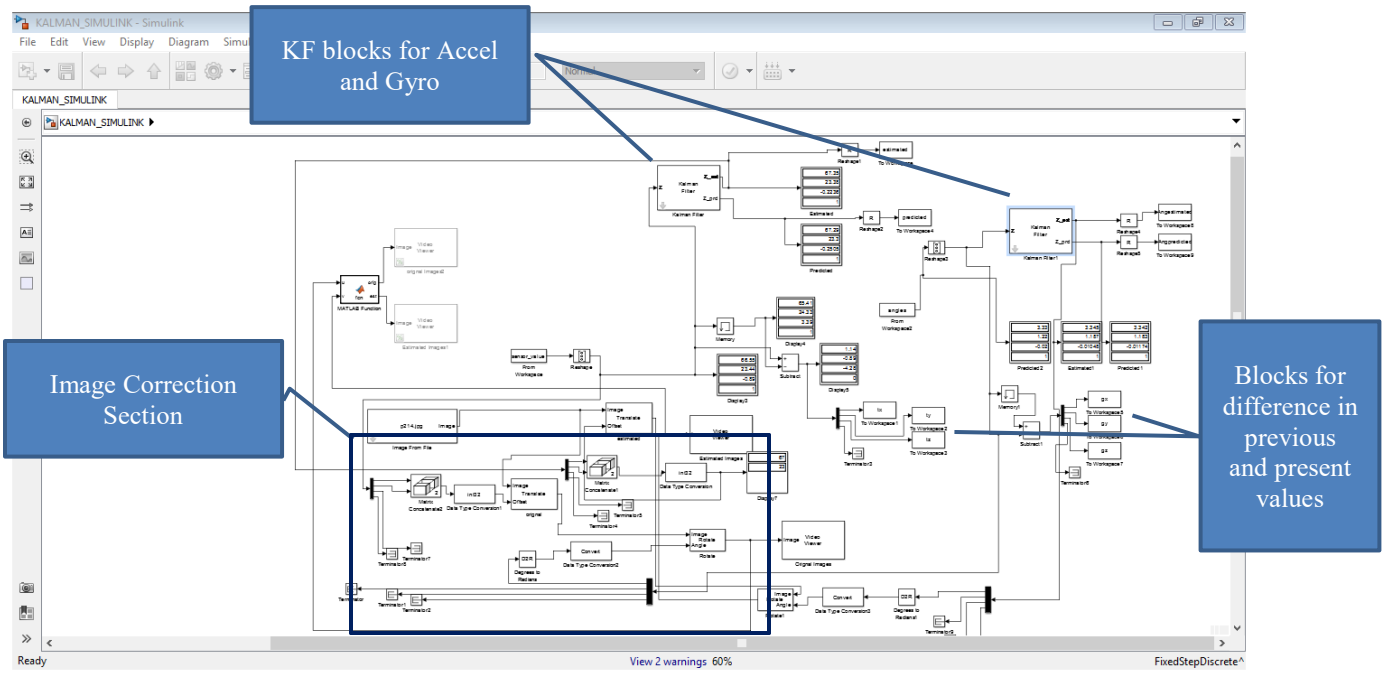


Figure 5.5: Figure shows the Image Correction section with KF block

The two Kalman blocks in the picture shows the complete model for the estimation of accelerometer values and on the right side Kalman model for estimating gyroscope values.

The images coming from the Kinect in the model is stored in the .tiff file format. It worth mentioning that the images are corrected with estimation done by using the rotation and translation block in the model. The difference among original picture and the estimated image is very slight because of the fact 25 frames per seconds one meter per minute.

Until now, we were implementing a linear model in the Kalman. Now the same approach with non-linear properties are implemented with the Extended Kalman Filter. The difference between KF and EKF is that to avoid the uncertainties in KF. Jaccobian matrix with parital derivatives are used in EKF to address the non-linearity.

Complete model is given on the next page

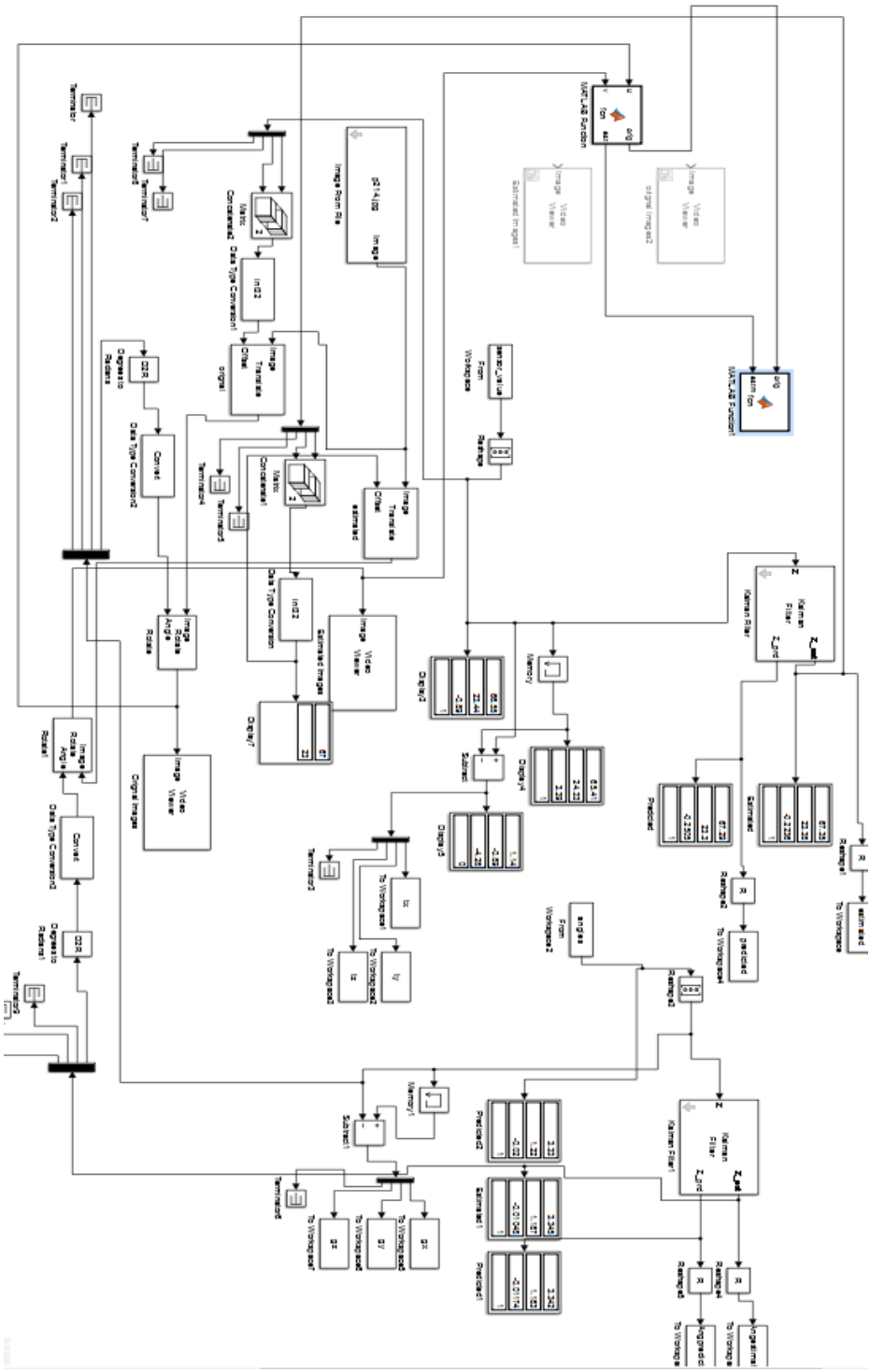


Figure 5.6: Full Estimation and Prediction Model in Simulink

5.4 Implementation With Ekf

This similar approach is used with EKF. The only difference is the jaccobian matrix with partial derivative. As in our case, the reading coming from accelerometer and gyroscope are numerical so here only difference in time will work better. Please refer to equation no 24-25.

Procedure

- Initialize all values that are require for the EKF in the beginning (it use use only once at the start of the code).
- Initially system must have two average values previous value and previous value 1. It is because of the fact that system requires derivative values in state space model.
- State space modeling discussed in detail on page no 14 and 19
- Partial derivative is taken by using the equation 24-25
- Each time when the reading come from sensor it is compare with the previous one. The previous values is then subtracted from the current value and put it to state space model.
- EKF explicitly run and estimate for accelerometer and gyroscope.
- Every reading is put into the EKF funciton `ekf_fun()` for estimation and prediction.
- When filter is runing it currently updating the images with the estimated values. `Imshowpair` command is use to show the difference.
- Most important in filter is the stabilizaiton of an image. For this, a transformation matrix is defined to inverse translate and rotate the image for the purpose of stabilization.
- Translation and rotation properties are used in this filter are AFFINE transform.
- Stabilization of an image requires AFFINE transformation object here `tformr`
- AFFINE also requires a matrix that contains transformation matrix `Tr`
- Applying transformation on the image `affine2d()` function is used
- Difference between the image coming from kinect and the estimated image is shown with the `imshowpair()`

Psudeo Code for the EKF with description is given on the next page

EKF Pseudo Code

Initialize all variables

Previous value_accel⟶initialize;

Previous value_gyro⟶initialize;

P⟶identity value matrix;

H⟶identity value matrix;

V⟶ identity value matrix;

R⟶ identity value matrix;

Initialize sensor values⟶accel;

Initialize sensor values⟶gyro;

%gyro data%

Loop

Call ekf( );

Set Estimated Value;

Set predicted value;

Assign Theta ⟶ Estimated value x

Assign Roh⟶ Estimated value y

Assign PHI⟶ Estimated value z

End

%Accelerometer Data%

Loop

Call ekf( );

Set Estimated Value;

Set predicted value;

Assign Ax⟶ Estimated value;

Assign Ay⟶Estimated value;

Assign Az⟶Estimated value;

End

Define Yaw⟶rotation matrix;

Define Pitch⟶ rotation matrix;

Define Roll⟶rotation matrix;

Tr⟶ the composite Affine matrix with yaw, pitch and roll including Ax and Ay;

Call affine2d(Tr); // correction and translation back section

Call showpair( ); // for showing difference

The following piece of code will basically initialize the values. calling of EKF function continuously and in addition to this, stabilize image is also displayed.

EKF FUNCTION

Function- (arguments)

Define time ⟶1/30 // for derivative

Loop

Derivative⟶ (abs(previous value)-abs(current value));

End

Define A- state space model matrix with derivative values

Perform prediction

*M⟶A*M; // M has to have previous estimate*

*P⟶A*P*A' + W + Q + W'; // perform for covariance*

Calculate Measurement values

MU⟶H+M; // H may have identity matrix

Perform update

Define S ⟶set its equation;

Define K⟶ set gain equation;

Define Estimate- set estimate equation with measured value;

Define Covariance- predicted covariance will be updated;

5.5 Summary

This chapter defines the whole idea of image stabilization via model and codes. Understanding on code reveals that how this whole project is implemented. Calibration of sensors mounted on Kinect is very important. Reading values in MATLAB along with Kinect images requires a lot of processing. The designing of state space model is trickier when there are only values. All the issues are discussed in detail along with the procedure

CHAPTER 6. RESULTS AND DISCUSSION

This part discusses the results that are generated by the Kalman and Extended Kalman. Tables displacement of different axis shows the difference between KF and EKF. These results clearly depict the nature of the KF estimation and EKF estimation.

EKF results better as here we remove non-linearity with the partial derivatives. Tables given below with graphs shows the actual displacement is the x, y and z direction. Similarly, angles are also calculated and displayed on the graph.

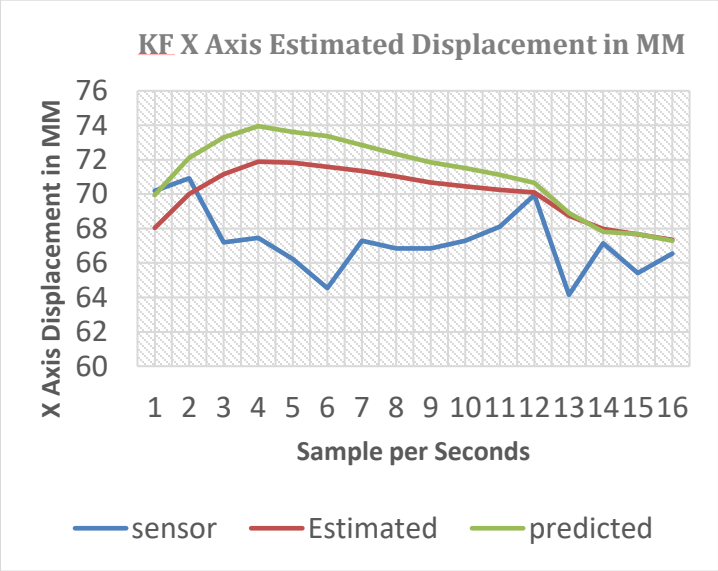
6.1 X Axis Displacement In mm

X AXIS DISPLACEMENT IN MM	KF ESTIMATED X AXIS DISPLACEMENT IN MM	EKF ESTIMATED DISPLACEMENT X AXIS IN MM	DELTA-DISPLACEMENT IN MM (KF-EKF)
70.92	68.03045171	65.43176045	2.598691257
67.2	70.00943556	66.91608472	3.09335084
67.46	71.16005448	66.50782159	4.65223289
66.23	71.88593916	66.37881112	5.507128042
64.54	71.82491073	66.06590713	5.759003601
67.3	71.57692113	65.69148244	5.885438693
66.85	71.34875987	65.69044207	5.658317796
67.3	71.02320682	65.64481633	5.378390492
68.1	70.68288274	65.61158259	5.071300148
69.95	70.45200242	65.62386791	4.828134514
64.15	70.25004964	65.68342449	4.566625152
67.14	70.10568369	65.84365182	4.26203187
65.41	68.73319187	65.60045509	3.132736774
66.55	67.97582813	65.57644112	2.399387008
65.86	67.66589852	65.4843248	2.181573723
66.11	67.34564039	65.46234672	1.88329367

Table 6-1: Displacement with difference in x direction for KF and EKF (Estimation)

X AXIS DISPLACEMENT IN MM	KF PREDICTED X AXIS DISPLACEMENT IN MM	EKF PREDICTED DISPLACEMENT X AXIS IN MM	DELTA-DISPLACEMENT IN MM (KF-EKF)
70.92	69.94461353	65.04897662	4.895636907
67.2	72.11137063	66.70767196	5.403698669
67.46	73.29455884	66.33591988	6.958638951
66.23	73.94895414	65.9934825	7.955471639
64.54	73.61360459	65.85626137	7.757343213
67.3	73.3644329	65.55192189	7.812511009
66.85	72.85231414	65.18250324	7.669810901
67.3	72.33852724	65.16912376	7.169403476
68.1	71.84499865	65.12533451	6.719664141
69.95	71.50564067	65.09195475	6.41368593
64.15	71.12716417	65.1026373	6.024526872
67.14	70.66106589	65.16015801	5.500907882
65.41	68.90983165	65.31578463	3.594047024
66.55	67.80784178	65.08785878	2.719983001
65.86	67.68253925	65.05678729	2.625751965
66.11	67.28973338	64.96765411	2.322079276

Table 6.2: Displacement with difference in x direction for KF and EKF (Prediction)



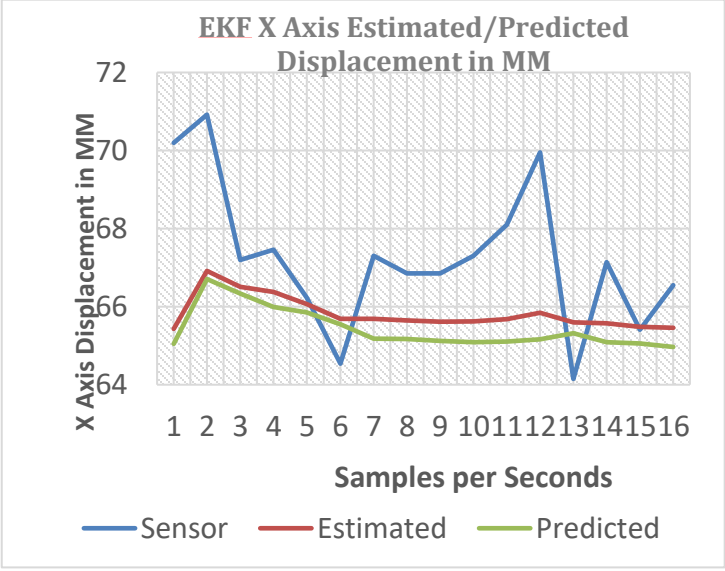


Figure 6.1: Estimated/ Predicted Values graph for KF and EKF in X axis displacement (mm)

These values are taken for the hemisphere from the rig. The reason behind this is that in the real world most of the potholes are likely to be in a sphere shape. These 15 values actually shows the sensors values and on that volume and perimeter will be calculated.

The upper two graphs explain the KF and EKF estimation. If we compare the two graphs we can batter see that estimation of EKF is more batter than KF. It the fact that the convergence of the EKF is slower than the KF but it estimates more sharply than KF. If we analyze the graph on the left it may one can consider that the values on the blue line has more pecks than that of the predicted and estimated one.

To answer this question, it is simple that we have more noise in the sensor values while the estimated one and the predicted one may be noise free. It is because of the fact that the state space model for both KF and EKF are noise free.

KF values when time increases the estimated, predicted and sensor values are approaching to each other. This is because that the convergence of the linear system is more fast as compare to dynamic.

In this regard, if we closely observe the EKF graph it seemed to be diverge but it is not the case. Non-linear system often converges very slowly.

We can see in the tables above 6.1 and 6.2 the last right column shows the difference between the EKF and the KF.

similarly results and the graphs for the Y and Z axis are given below.

6.2 Y Axis Displacement In mm

Y AXIS DISPLACEMENT IN MM	KF ESTIMATED DISPLACEMENT Y AXIS IN MM	EKF ESTIMATED DISPLACEMENT Y AXIS IN MM	DELTA-DISPLACEMENT IN MM (KF-EKF)
18	17.06594751	18.10247113	1.03652362
17.25	19.39456264	18.06662284	1.3279398
21.99	21.87891905	18.52350628	3.355412768
21.67	23.63119371	18.85958449	4.771609219
23.6	24.79422686	19.32973683	5.464490027
23.66	25.94705293	19.57857598	6.36847695
22.41	26.04534904	19.87027111	6.17507793
22.66	26.09602574	20.11576162	5.98026412
22.66	26.04355358	20.3377896	5.705763982
22.13	25.86652256	20.51574806	5.350774492
21.69	25.42576408	20.70368205	4.722082032
20.04	24.3321471	20.8960874	3.436059703
25.49	24.1200354	21.28529546	2.834739939
22.33	23.13915264	21.56366273	1.575489908
24.33	23.5688437	21.73857636	1.830267346
23.44	23.37950045	21.88941558	1.490084865

Table 6-3: Displacement with difference in y direction for KF and EKF (Estimation)

Y AXIS DISPLACEMENT IN MM	KF PREDICTED DISPLACEMENT Y AXIS IN MM	EKF PREDICTED DISPLACEMENT Y AXIS IN MM	DELTA-DISPLACEMENT IN MM (KF-EKF)
18	19.74577406	16.91438202	2.831392043
17.25	22.33727174	16.24361371	6.093658032
21.99	24.86722515	16.88581102	7.98141413
21.67	26.51941468	17.29745498	9.221959696
23.6	27.29839826	17.61531017	9.683088086
23.66	28.4495694	18.05052812	10.39904128
22.41	28.15032501	18.28998256	9.860342453
22.66	27.93747433	18.56125975	9.376214572
22.66	27.67051586	18.79212954	8.878386323
22.13	27.34161611	19.00036324	8.341252866
21.69	26.65372443	19.16805244	7.485671981
20.04	25.10968218	19.34338092	5.766301262
25.49	24.3673311	19.52305971	4.844271396
22.33	22.90397175	19.88067463	3.023297112
24.33	23.59214073	20.14428394	3.447856788
23.44	23.30123064	20.31097748	2.990253164

Table6-4: Displacement with difference in y direction for KF and EKF (Predicted)

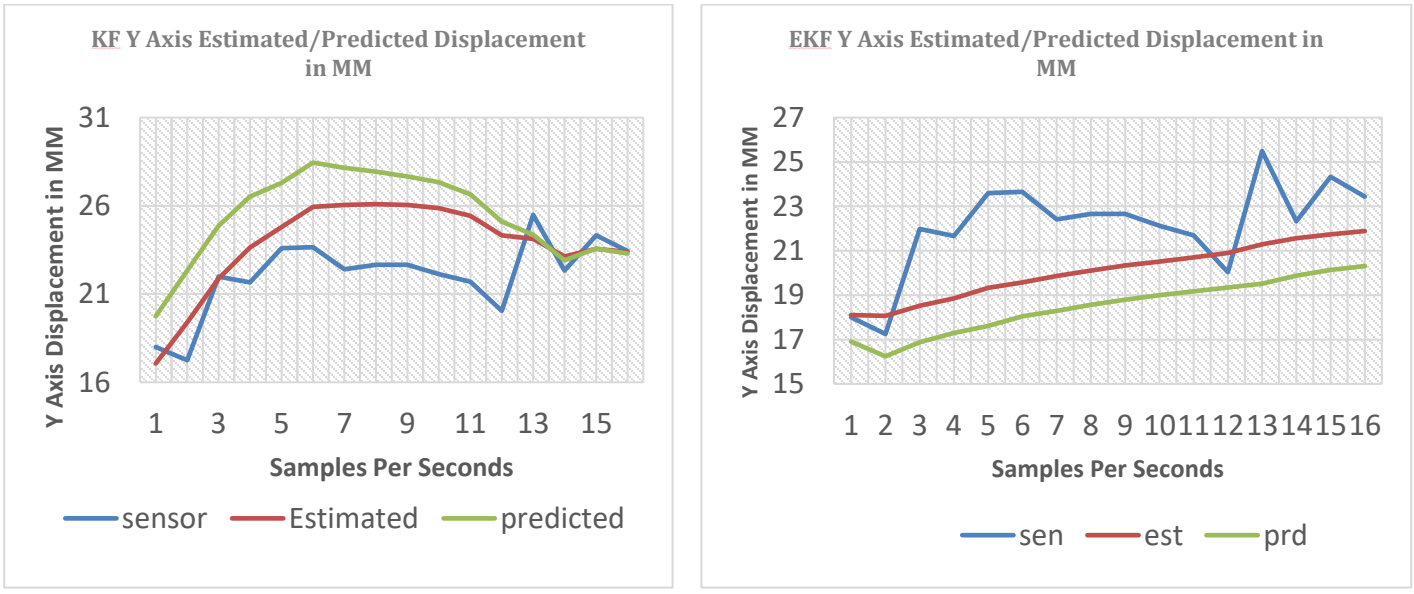


Figure 6.2: Estimated/ Predicted Values graph for KF and EKF displacement (mm) in Y axis

6.3 Z Axis Displacement In mm

Similarly, the table for the displacement in z direction is given below

Z AXIS DISPLACEMENT IN MM	KF ESTIMATED DISPLACEMENT Z AXIS IN MM	EKF ESTIMATED DISPLACEMENT Z AXIS IN MM	DELTA-DISPLACEMENT IN MM (KF-EKF)
6.64	7.656647286	6.662372391	0.994274895
7.94	8.407740269	7.253413704	1.154326565
5.76	8.53801743	6.783024108	1.754993322
5.94	8.252059918	6.506688433	1.745371485
2.73	7.154775423	6.059426798	1.095348625
8.88	7.150047062	5.779709201	1.370337861
3.47	6.014217068	5.57916604	0.435051028
4.51	5.261281677	5.402446426	0.141164749
4.51	4.648463641	5.247264538	0.598800897
4.85	4.214553007	5.13987644	0.925323433
2.92	3.508458122	5.032188203	1.523730082
-0.62	2.2215288	4.934795527	2.713266728
-4.04	0.706559151	4.55702484	3.850465689
-4.67	-0.671945395	4.322644783	4.994590178
3.39	0.066562935	4.184621816	4.11805888
2.89	-0.223628008	4.08224941	4.305877418

Table 6-5: Displacement with difference in z direction for KF and EKF (Estimated)

Z AXIS DISPLACEMENT IN MM	KF PREDICTED DISPLACEMENT Z AXIS IN MM	EKF PREDICTED DISPLACEMENT Z AXIS IN MM	DELTA-DISPLACEMENT IN MM (KF-EKF)
6.64	8.57544496	5.325629612	3.249815348
7.94	9.416669101	6.214285714	3.202383387
5.76	9.562579521	5.741041529	3.821537992
5.94	9.242307108	5.395796636	3.846510472
2.73	8.013348474	5.171348906	2.841999567
8.88	8.008052709	4.820901206	3.187151503
3.47	6.735923117	4.594470731	2.141452386
4.51	5.892635478	4.43321493	1.459420548
4.51	5.206279278	4.292331016	0.913948262
4.85	4.720299368	4.168601006	0.551698362
2.92	3.929473096	4.082107971	0.152634875
-0.62	2.488112256	3.996649057	1.508536801
-4.04	0.791346249	3.919080913	3.127734663
-4.67	-0.752578843	3.626680315	4.379259157
3.39	0.074550487	3.436870994	3.362320506
2.89	-0.250463369	3.324779826	3.575243196

Table 6-6: Displacement with difference in z direction for KF and EKF (Predicted)

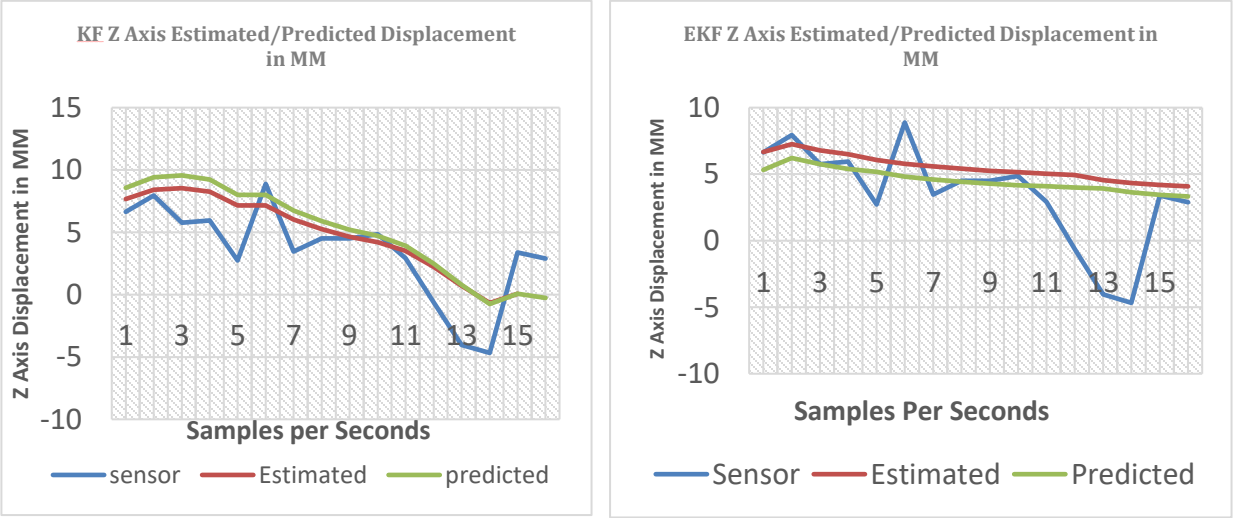


Figure 6.3: Estimated/Predicted Values graph for KF and EKF in Z axis displacement (mm)

6.4 X Axis Rotation In degrees

Here now we will insert the values coming from the gyroscope and these values will be estimated via EKF. It is because of the fact that the angles calculation involves the non-linearity that's why we go for the EKF estimation. The tables will give us a better idea about estimation.

Sensor X AXIS ROTATION IN DEG	EKF ESTIMATED X AXIS ROTATION IN DEG	EKF PREDICTED X AXIS ROTATION IN DEG	DELTA-DISPLACEMENT IN DEG (EKF EST-Sen)
3.55	3.314545253	3.290734535	0.235454747
3.36	3.303346525	3.250992063	0.056653475
3.37	3.296852044	3.277499818	0.073147956
3.3	3.28183923	3.270901316	0.01816077
3.26	3.273559192	3.256289342	0.013559192
3.37	3.270707967	3.247852375	0.099292033
3.34	3.267310044	3.244844254	0.072689956
3.37	3.266492783	3.241491377	0.103507217
3.4	3.266208091	3.240595263	0.133791909
3.48	3.26801486	3.240295219	0.21198514
3.25	3.257100079	3.242018487	0.007100079
3.35	3.252295348	3.231611017	0.097704652
3.25	3.247498076	3.226642366	0.002501924
3.34	3.244811431	3.221882921	0.095188569
3.33	3.24206037	3.219147804	0.08793963
3.31	3.239524951	3.216420706	0.070475049

Table 6-7: Angular Displacement with difference in x direction for EKF (Estimated/Predicted)

6.5 Y Axis Rotation In degrees

Y AXIS ROTATION IN DEG	EKF ESTIMATED Y AXIS ROTATION IN DEG	EKF PREDICTED Y AXIS ROTATION IN DEG	DELTA-DISPLACEMENT IN DEG
0.87	0.967845373	0.907243483	0.097845373
1.1	0.994129543	0.953115265	0.105870457
1.08	1.006368245	0.92827421	0.073631755
1.17	1.031815265	0.940149802	0.138184735
1.24	1.048138648	0.96352059	0.191861352
1.12	1.055182041	0.979060313	0.064817959
1.14	1.063093756	0.98593202	0.076906244
1.11	1.066918763	0.993299095	0.043081237
1.08	1.069603549	0.99700118	0.010396451
0.99	1.068351886	0.999545847	0.078351886
1.33	1.088395082	0.9984987	0.241604918
1.12	1.097208417	1.016567267	0.022791583
1.21	1.106901986	1.025153868	0.103098014
1.15	1.112967478	1.034185857	0.037032522
1.22	1.119858737	1.039967566	0.100141263
1.2	1.126043913	1.046382269	0.073956087

Table 6-8: Angular Displacement with difference in y direction for EKF (Estimated/Predicted)

6.6 Z Axis Rotation In degrees

Z AXIS ROTATION IN DEG	EKF ESTIMATED Z AXIS ROTATION IN DEG	EKF PREDICTED Z AXIS ROTATION IN DEG	DELTA-DISPLACEMENT IN DEG
0.4	0.330347364	0.260848507	0.069652636
0.29	0.306354122	0.214814815	0.016354122
0.3	0.295849801	0.243772934	0.004150199
0.13	0.271187312	0.235079321	0.141187312
0.46	0.257587477	0.215880472	0.202412523
0.17	0.252759727	0.20479429	0.082759727
0.23	0.24706407	0.200730676	0.01706407
0.24	0.245619366	0.196233274	0.005619366
0.14	0.24503261	0.194974224	0.10503261
-0.04	0.247866998	0.194485848	0.287866998
-0.19	0.229612208	0.196644856	0.419612208
-0.24	0.221485397	0.182714842	0.461485397
0.16	0.21346064	0.175997056	0.05346064
-0.05	0.208888097	0.169625502	0.258888097
-0.02	0.204209788	0.16590517	0.224209788
0.04	0.199910563	0.162195025	0.159910563

Table 6-9: Angular Displacement with difference in z direction for EKF (Estimated/Predicted)

In these tables DELTA on the right most shows the error between EKF estimation and the original values coming from the sensors. The detail view of the following can be presented in the graph for better visualization.

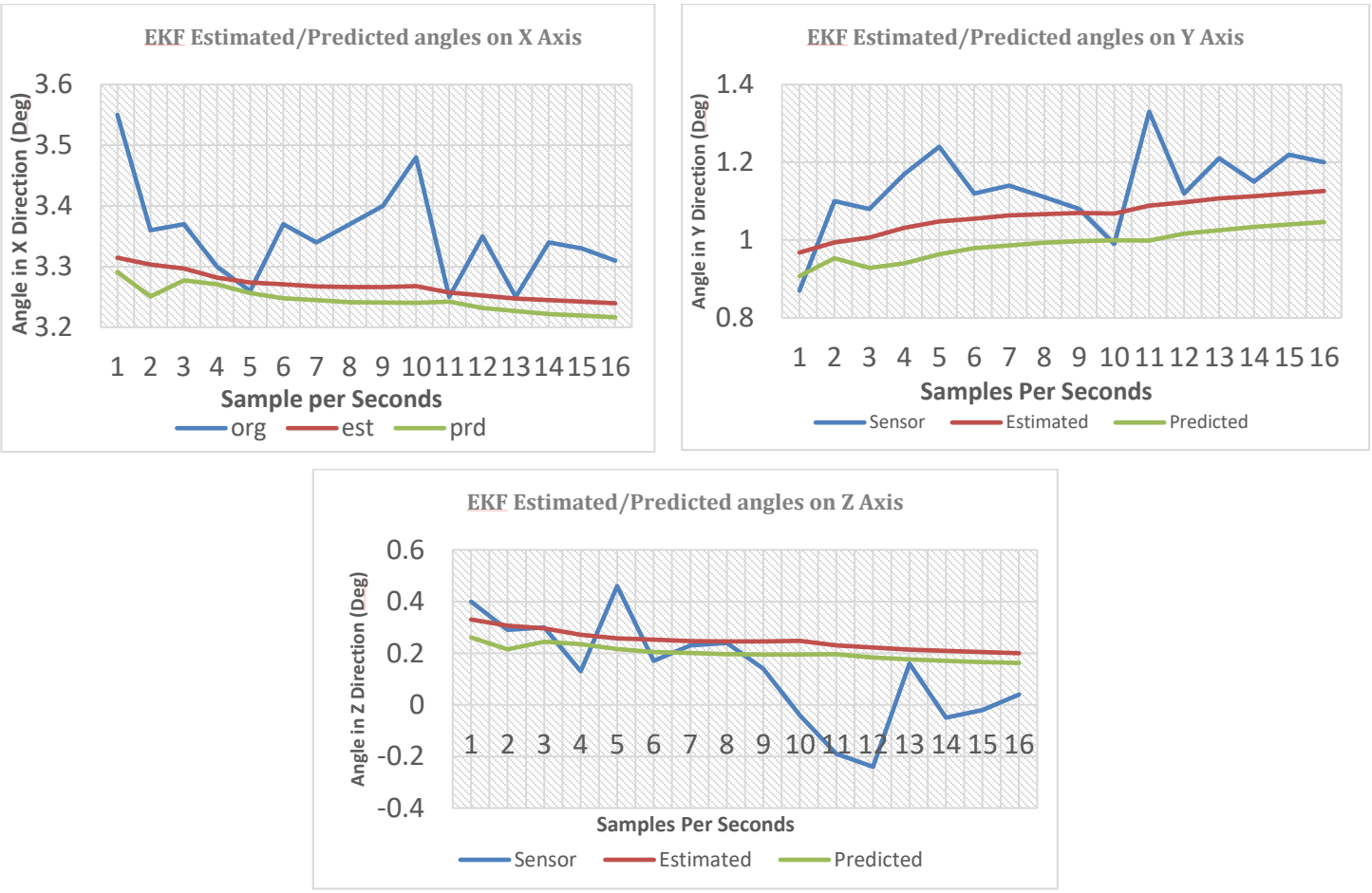


Figure 6.4: Estimated/Predicted Values graph for EKF in X,Y and Z angles

6.7 Comparison Table

The comparison table is an important table where it shows the pothole metrology like volume and perimeter. It also compares the volume and perimeter calculated after the KF and EKF estimation. When image was corrected the volume and perimeter was calculated. Original values in the table shows that it has been calculated with human hands and foot rule.

Similarly, table shows that which techniques is more accurate in terms of error. The more the error less it will be the better technique.

Org Vol	Org Peri	Mrd Vol	Mrd Peri	Vol Cal After KF Est	Vol Cal After EKF Est	Peri Cal After KF Est	Peri Cal After EKF Est	Error in Vol Mrd—KF est	Error in Peri Mrd—KF est	Error in Vol Mrd—EKF est	Error in Peri Mrd—EKF est
150	64.58	158.15	66.3935	144.8	148.35	64.7003	64.836113	13.35	1.6932	9.8	1.5573165
150	64.58	158.35	68.3921	143.55	148.5	64	66.372457	14.8	4.3921	9.85	2.0196429
150	64.58	157.55	66.9544	141.3	147.65	64.2309	65.625964	16.25	2.7235	9.9	1.3284635
150	64.58	161.2	67.9964	144.85	148.05	64.4327	66.097364	16.35	3.5637	13.15	1.8996364
150	64.58	166.75	67.9881	142.8	148.25	62.7374	62.654029	23.95	5.2507	18.5	5.3338706
150	64.58	157.95	66.6382	142.5	148	64.2028	63.139722	15.45	2.4354	9.95	3.4984784
150	64.58	162.25	67.3154	141.95	148	62.3881	63.359144	20.3	4.9273	14.25	3.9563856
150	64.58	158	67.6768	142.45	148.1	64.0313	65.165559	15.55	3.6455	9.9	2.5114406
150	64.58	166.65	69.2282	143.2	147.35	63.8797	66.117383	23.45	5.3485	19.3	3.1104165
150	64.58	171.95	69.7169	143.35	147.6	62.4178	62.09877	28.6	7.2991	24.35	7.6188302
150	64.58	162.5	67.537	142.15	147.5	63.9012	64.63974	20.35	3.6358	15	2.8979726
150	64.58	161.15	67.5247	142.85	147.9	58.8056	65.61432	18.3	8.7191	13.25	1.9097725
150	64.58	165.35	67.2206	141.35	147.8	59.6887	65.674	24	7.5319	17.55	1.5466
150	64.58	168.6	65.8068	141.85	148.15	64.0899	65.1319	26.75	1.7169	20.45	0.6749
150	64.58	159.8	68.4894	141.95	148.05	63.7197	65.009	17.85	4.7697	11.75	3.4804
150	64.58	164.74	68.410	142.05	148.25	63.3495	65.1334	22.6944	5.09156	16.494	3.30766

Table 6-10: Volume & Perimeter calculations after KF and EKF Estimation with errors

In the table above the values in the right most 4 columns show that the KF and EKF significantly reduce the error. Here to notice that the EKF is more robust because as compare with the KF error reduction it significantly reduces the error. Which means that it estimates the batter.

Comparison table for 6 Rig potholes shapes

Name	Org Vol	Org Peri	Avg Mrd Vol	Avg Mrd Peri	Avg Vol Cal After KF Est	Avg Peri Cal After KF Est	Avg Vol Cal After EKF Est	Avg Peri Cal After EKF Est	Avg Error in Vol Mrd—KF est	Avg Error in Peri Mrd—KF est	Avg Error in Vol Mrd—EKF est	Avg Error in Peri Mrd—EKF est
Hemisphere	150	62	162.559	67.70	144.8	63.16	147.968	64.791	17.759	4.54	14.591	2.909
Cone	403	81	408.24	86.566	395.272	83.64	399.9783	80.885	12.967	2.926	8.261	5.681
Prism	545	72	536.44	76.85	538.88	77.556	541.2	70.851	2.44	0.706	4.76	5.99
Pyramid	340	63	332.54	69.44	335.231	58.66	338.85	61.442	2.691	10.78	6.31	7.998
Cylinder	950	95	956.751	101.22	942.985	98.808	947.840	96.445	13.766	2.412	8.911	4.775
Cube	780	83	787.22	87.56	783.660	85.56	779.24	81.465	3.56	2	7.98	6.095

Table 6-11: Volume & Perimeter calculations after KF and EKF Estimation with errors for Hemisphere, Cone and Prism

Name	Avg Error in Vol Original—KF est	Avg Error in Peri Original—KF est	Avg Error in Vol Original—EKF est	Avg Error in Peri Original—EKF est
Hemisphere	5.2	1.16	2.014	2.791
Cone	7.728	2.64	3.0217	0.115
Prism	6.12	5.556	3.8	1.149
Pyramid	4.769	4.34	1.15	1.558
Cylinder	7.015	3.808	2.16	1.44
Cube	3.66	2.56	0.76	1.535

Table 6-12: Average Error table between original and KF, EKF estimated values

Now here are the pictures that has been taken and stabilized

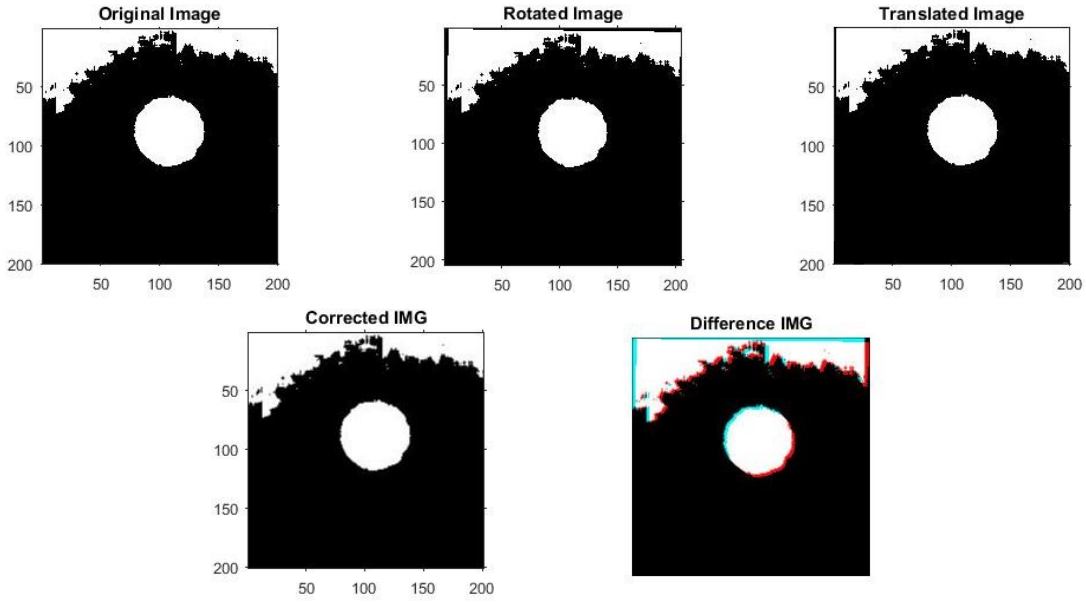


Figure 6.5: Original, Rotated, Translated, Correction and Difference image

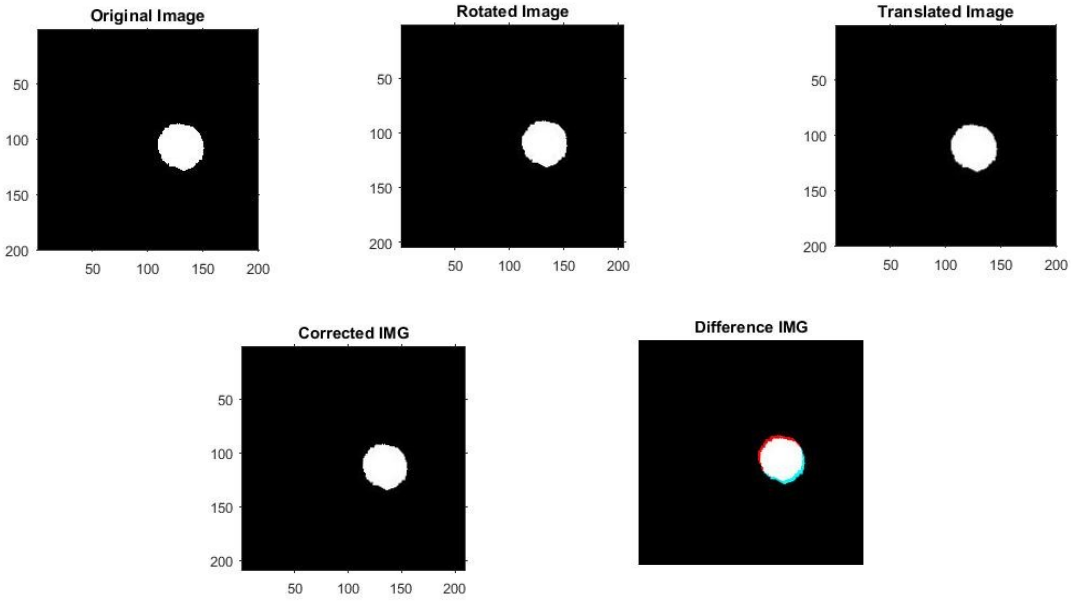


Figure 6.6: Original, Rotated, Translated, Correction and Difference image of Prism

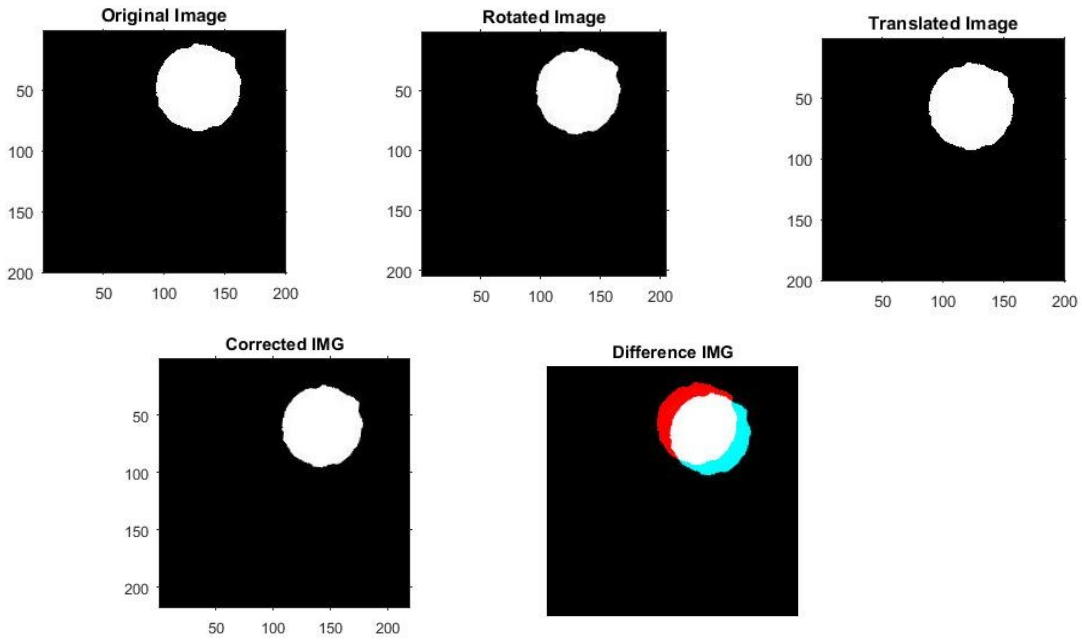


Figure 6.7: Original, Rotated, Translated, Correction and Difference image of Cone

Here the red and cyan shows the difference between the original and corrected image. That how these images were corrected according to the use of KF and EKF

CHAPTER 7: CONCLUSION AND FUTURE WORK

7.1 Conclusion

Extended Kalman filter with Affine transformation is used in the proposed technique to stabilize the images. This is because of to find the exact metrology of the pothole. Which include volume and perimeter. These two will able us to calculate an accurate amount of filling material. The proposed technique indicates promising outcomes for adjustment of 3D pothole pictures the fault in displacement and angle estimations because of precariousness of pictures is impressively diminish.

The stability in images gives the power to accurately calculate the material required for the filling of potholes. This enormously reduce the cost, material and time for surveyors and engineers.

7.2 Future Work

The real-time implementation of different affine effects which involves shearing, scaling. The proposed technique is also implemented on Kinect 1 and Kinect 2, conference and Journal paper writing will be covered up. The proposed technique can also be studied under Unscented Kalman in future.

In addition to this, potholes will be filled with different liquid and then try to find its metrology. It will give us improved idea about the calculation of material for potholes.

REFERENCES

[1] Y. C. Joon ki, **"**An Edge Detection Approach to Digital Image Stabilization Based on Tri-State Adaptive," *CH2978/91/00000-0164 1991* IEEE

[2] A. M. H. I. T. S. Y. K. Uomori, "Automatic Image Stabilization System by Full Digital Signal Processing," *IEEE Transaction on Consumer Electronics*, vol. 36, No. 3 August 1990.

[3] S. I. A. N. M. S. Sato Koichi, "Control Techniques for Optical Image Stabilizing System," *IEEE Control Techniques for optical image stabilization system*, 1993.

[4] A. C. M. M. G. M. Vella, "Digital Image Stabilization by Adaptive Block Motion Vectors Filtering," *IEEE Transactions on Consumer Electronics*, Vol. 48, No. 3, August 2002.

[5] A. G. A. A. Nikolaos, "Image Sequence Stabilization Using Fuzzy Kalman Filtering and Log-polar Transformation," *IEEE* 2005.

[6] S. Erturk, "Digital Image Stabilization with Sub-image phase correlation based global motion estimation," *IEEE Transactions on consumer electronics*, vol.49, No. 4, November 2003.

[7] P. A. K. D. G. W. B. Hansen, "Real-time Scene Stabilization and Mosaic Construction," *IEEE,* Vol. 25, No.4 June 1994.

[8] S. H. L. K. R. L. Chang, "A Robust and Efficient Video Stabilization Algorithm," *IEEE*, Vol. 443, No.342 July 2004.

[9] S. H. L. S. W. J. E. S. K. Sung, "Fast Digital Image Stabilizer Based on Gray-coded Bit-plane Matching," *IEEE Transaction on Consumer Electronics*, Vol. 45, No.3 August 1999

[10] Y. W. L. Y. J. B. P. T. J. S. Liu, "Video Stabilization with a Depth Camera," IEEE Vol. 51, No.13, 2012.

[11] Moazzam, K.Kamal, "Metrology and Visualization of Potholes using the Microsoft Kinect Sensor," *IEEE 16th International annual conference on ITSC* 2013, 6 October 2013.