UNIVERSITY OF KWAZULU-NATAL
COLLEGE OF AFRICULTURE, ENGINEERING AND SCIENCE

FUSION OF TIME OF FLIGHT (ToF) CAMERA's EGO-MOTION AND INERTIAL NAVIGATION

by

Thikhathali Terence Ratshidaho

Supervisor:

Professor Jules-Raymond Tapamo

In fulfillment of the Masters of Science in Engineering
College of Agriculture, Engineering and Science, University of KwaZulu-Natal

25 July 2013

Abstract

For mobile robots to navigate autonomously, one of the most important and challenging task is localisation. Localisation refers to the process whereby a robot locates itself within a map of a known environment or with respect to a known starting point within an unknown environment. Localisation of a robot in unknown environment is done by tracking the trajectory of a robot whilst knowing the initial pose. Trajectory estimation becomes challenging if the robot is operating in an unknown environment that has scarcity of landmarks, is GPS denied, is slippery and dark such as in underground mines.

This dissertation addresses the problem of estimating a robot's trajectory in underground mining environments. In the past, this problem has been addressed by using a 3D laser scanner. 3D laser scanners are expensive and consume lot of power even though they have high measurements accuracy and wide field of view. For this research work, trajectory estimation is accomplished by the fusion of an ego-motion provided by Time of Flight(ToF) camera and measurement data provided by a low cost Inertial Measurement Unit(IMU).

The fusion is performed using Kalman filter algorithm on a mobile robot moving in a 2D planar surface. The results shows a significant improvement on the trajectory estimation. Trajectory estimation using ToF camera only is erroneous especially when the robot is rotating. The fused trajectory estimation algorithm is able to estimate accurate ego-motion even when the robot is rotating.

Declaration Supervisor

As the candindate's supervisor I agree to the submission of this dissertation

_____

Prof Jules-Raymond Tapamo

Declaration Plagiarism

I, Thikhathali Terence Ratshidaho, declare that

1. The research reported in this dissertation, except where otherwise indicated, is my original research.
2. This dissertation has not been submitted for any degree or examination at any other university.
3. This dissertation does not contain other persons data, pictures, graphs or other information, unless specifically acknowledged as being sourced from other persons.
4. This dissertation does not contain other persons' writing, unless specifically acknowledged as being sourced from other researchers. Where other written sources have been quoted, then:
   (a) Their words have been re-written but the general information attributed to them has been referenced
   (b) Where their exact words have been used, then their writing has been placed in italics and inside quotation marks, and referenced.
5. This dissertation does not contain text, graphics or tables copied and pasted from the Internet, unless specifically acknowledged, and the source being detailed in the dissertation and in the References sections.

Signed

_____

Declaration Publications

1. Conference	Robotics and Mechatronics Conference of South Africa 2012 (RobMech2012)
Title of the paper	ToF Camera Ego-Motion Estimation
Authors	Thikhathali Terence Ratshidaho (Main Author) Jules Raymond Tapamo; Jonathan Claassens Natasha Govender
ISBN number	978-1-4673-5184-3
2. Conference	4th CSIR Biennial Conference: Real problems relevant solutions
Title of the paper	ToF Camera Ego-Motion Estimation (Poster)
Authors	Thikhathali Terence Ratshidaho (Main Author) Jules Raymond Tapamo; Jonathan Claassens Natasha Govender
3. Conference	Fourth CSIR Emerging Researcher Symposium
Title of the paper	Fusion of ToF camera's Ego-motion and Inertial Navigation (Poster)
Authors	Thikhathali Terence Ratshidaho (Main Author) Jules Raymond Tapamo; Natasha Govender
4. Journal Paper	South African Journal of Industrial Engineering (accepted)
Title	An Investigation into Trajectory Estimation in Underground Mining Environments using Time of Flight Camera and Inertial Measurement Unit.
Authors	Thikhathali Terence Ratshidaho (Main Author) Jules Raymond Tapamo Natasha Govender

Signed

_____

Acknowledgements

I would like to acknowledge the following people for the guidance and advice throughout this work.

- My university supervisor, Prof. Jules Raymond Tapamo, for his constant guidance, discussions and advice, sometimes even after hours on Skype.
- My CSIR supervisors, Natasha Govender and Jonathan Claassens for their insightful discussions and help throughout the project. Also thank Natasha for her fast respond with the write-up feedback.
- My friends, Zwivhuya Mulibana, Ndivhuwo Makondo and Nzudzanyo Ranwaha for their help in proof reading the dissertation.
- My parents, for their endless support and always trying to make me laugh to reduce stress.
- My siblings, Murendeni for the support and Wayne for your support and for proof reading.
- My girlfriend Belinda Matebese for her endless support and guidance.
- Finally, but not least, I would like to thank almighty God for everything.

Contents

List of Figures

List of Tables

List of Algorithms

Nomenclature

Mathematical variables

$\mathbf{M}$	A matrix, Capital bold letter
$\mathbf{m}$	A 3×1 vector, small bold letter
m	A variable, small normal letter
$\bar{q}$	Quaternion

Acronyms

3D	Three Dimensional
AD	Allan Deviation
AVAR	Allan Variance
AHRS	Attitude and Heading Reference System
ANN	Approximate Nearest Neighbour
CCD	Charge Coupled Device
CMOS	Complementary Metal Oxide Semiconductor
DoF	Degree of Freedom
EKF	Extended Kalman Filter
ESM	Efficient Second order Method
fps	frame per second
GPS	Global Positioning System

ICP Iterative Closest Point

IMU Inertial Measurement Unit

INS Inertial Navigation System

KD K Dimensional

KLT Kanade Lucas Tomasi

LED Light Emitting Diode

MEMS Micro-Electro-Mechanical Systems

MUMC Minimally Uncertain Maximum Consensus

NDT Normal Distribution Transform

NIR Near InfraRed

PC Personal Computer

pose position and orientation

RANSAC RANdom SAmple Consensus

RMS Root Mean Square

SIFT Scale Invariant Feature Transform

SURF Speeded Up Robust Feature

ToF Time of Flight

Chapter 1

Introduction

1.1 Background

Recently, there has been a lot of interest in autonomous mobile robots, especially in dangerous work environments for human beings. While the mobile robot is in operation, it must be able to navigate within the environment. For mobile robots to navigate autonomously, one of the most important and challenging tasks is localisation. Localisation refers to the process whereby a robot locates itself within a map of a known environment or with respect to a known starting point within an unknown environment. Secondly, for a mobile robot to perform any useful task, it is important to know its pose (position and orientation) at any point in time during the process of that specific task.

Localisation methods can be grouped into relative and absolute ones. Relative localisation methods make use of onboard sensors to estimate the robot's pose using dead reckoning methods. The concept of dead reckoning methods refers to the process of calculating current pose using the previously estimated pose. Localisation systems such as wheel odometry, which use sensors to measure wheel rotation to estimate the odometry and Inertial Navigation System (INS) that integrate measurements from accelerometers and gyroscopes to estimate position, velocity and orientation, all fall under the relative localisation methods. The downfall of these methods is cumulative errors since they make use of previous estimates which might be erroneous to compute the current pose.

Absolute localisation methods make use of landmarks and beacons within the envi-

ronment to estimate the absolute robot pose. The most common absolute localisation method is the Global Positioning System (GPS) which uses signals from satellites to estimate the position of the robot. GPS, however, only works in outdoor environments and the satellites signals can also be blocked by tall structures.

The problem of localisation becomes challenging if the robot is operating in unknown environments, which have scarcity of landmarks, are GPS denied, slippery and dark such as underground mines. In such environments, absolute localisation systems are not applicable without altering the environment and relative localisation systems such as robot onboard odometry experience slippage which introduces drift to the estimated pose. Localisation in unknown environment is achieved by tracking the trajectory of the robot knowing the initial pose.

Trajectory estimation in environments similar to underground mines has successfully been achieved by using Three Dimensional (3D) laser scanners [2]. 3D laser scanners provide high accurate distance measurements with a wide field of view, making it easy to estimate the transformation between two scans using registration algorithms such as Iterative closest Point (ICP) [3]. These sensors, however, are very expensive and consume a lot of power.

1.2 Problem Statement

The problem addressed in this dissertation is the design, development and implementation of a system capable of estimating the trajectory of a robot operating in underground mine stope. The underground mine stopes are generally 0.5 m high, 30 m wide and 30 in length. Underground environments are characterised by scarcity of unique landmarks, absence of GPS signals, slipperiness and a possible absence of light.

The trajectory assists mobile robots in localisation and map building of the environment. Robot trajectory can be accurately estimated using 3D laser scanners in the same environmental conditions. The only concern with 3D laser scanners is that they are expensive and they consume a lot of power due to mechanically moving parts. Autonomous robots have a limited battery power, hence using low power consumption sensors would be to an advantage because it would operate for extended periods.

1.3 Methodology

The estimation of a robot's trajectory is performed by fusing a Time of Flight (ToF) camera's ego-motion with the Inertial Navigation System (INS) provided by a low cost Micro-Electro-Mechanical Systems (MEMS) Inertial Measurement Unit (IMU). The combined price and power consumption of a ToF camera and an IMU are less than that of a 3D laser scanner.

ToF cameras are compact sensors which provides both range and amplitude images. They are attractive because they operate at a video frame rate (max 54 fps) and have a working range of up to 10 m, compared to Microsoft® Kinect sensor that has a working range of 3 m. ToF cameras are characterised by a number of errors that limit the accuracy and precision of the sensor. After a critical literature review of ToF cameras and the types of errors associated with them, filtering and calibration methods are implemented to reduce and reject erroneous points.

Three algorithms for estimating ToF camera ego-motion are investigated and implemented. Two of these algorithms are variants of the ICP algorithm. One uses a modified point-to-point ICP [3] and the other uses point-to-plane ICP [4] algorithm. The third algorithm makes use of Scale Invariant Feature-Transform (SIFT) [5] algorithm to track features, and 3D information of these features are extracted from range images to estimate the trajectory.

ToF camera ego-motion is then fused with the INS provided by the IMU to estimate a more robust robot trajectory. IMU consist of 3 accelerometers and 3 gyroscopes that are orthogonally mounted such that measurements are in x, y and z axes. By integrating linear acceleration and angular velocity from accelerometers and gyroscopes respectively, INS is able to track position, velocity and orientation of the robot [6]. The fusion was done by using the Kalman filter algorithm [7], using IMU measurements as inputs and ToF camera ego-motion estimates as observations.

1.4 Aims and Objectives

The primary objectives of this project are:

- to investigate different methods for the calibration of ToF camera to improve the quality of the ToF 3D and intensity images;

- to investigate the different algorithms for filtering ToF camera images and to implement one of them;
- to investigate various ways of tracking camera pose using ToF images and select the most accurate and suitable for underground mine stopes;
- to develop an algorithm for INS using low-cost IMU;
- to develop an algorithm that fuses INS and ToF ego-motion to estimate a more accurate trajectory;
- to evaluate the fusion method using Vicon motion capture system.

The ultimate goal of the project is to develop a system capable of tracking a robot's trajectory in unknown environments with a scarcity of unique landmarks, no GPS signals and absence of light. This will be done by fusion of ego-motion from ToF camera and INS from IMU. The research question to be answered in this research is:

- Is it possible to estimate a useful trajectory by fusion ToF camera with IMU measurements?

1.5 Contributions

Although the main goal of this work is to design, develop and implement a fused ToF-IMU system that estimates a robot trajectory, novel contributions to the field of robotics are made. The main contributions are that:

- ToF ego-motion estimation algorithms are implemented, tested on data with characteristics similar to underground mining environment and evaluated using high accuracy motion capture system;
- Calibration algorithm of normal camera and IMU is adopted and modified for ToF camera and IMU calibration which was tested on simulated data;
- Fusion of ToF camera and IMU is designed and implemented. The fusion algorithm also incorporate the transformation between the two sensors to increase the accuracy.

The algorithms developed for trajectory estimation can easily be used to assist in map building, augmented reality and object reconstruction with just few modifications of the algorithm.

1.6 Thesis Outline

This dissertation discusses the fusion of ToF cameras' ego-motion with INS provided by the IMU. Chapter 2 gives the literature review which is subdivided into ToF ego-motion estimation, Inertial Navigation and Fusion of ToF camera's ego-motion and IMU data. The design and analysis of the methods used are discussed in chapter 3. Simulations and experimental results to validate the methods used are presented in chapter 4. In chapter 5, conclusions and recommendations for future work, based on the results obtained are made.

Chapter 2

Literature Review

In this chapter, work related to trajectory estimation for mobile autonomous robot using a 3D ToF camera and IMU is discussed. ToF camera is used to estimate the ego-motion using captured images while IMU data is used to estimate its position, velocities and orientation using Inertial Navigation System (INS) algorithm.

The first section of the chapter details ego-motion estimation algorithms of ToF camera. Errors that affect the accuracy of the range data of the ToF cameras are discussed followed by some of the measures that can be used to handle these errors. Different algorithms for ego-motion estimation are reviewed and accessed if they are suitable for South African underground mining environments.

The second section of the chapter reviews methods for improving ToF camera's ego-motion by fusing ego-motion with IMU data. This section starts by reviewing algorithms for finding the transformation between the ToF camera and the IMU and then followed by the actual algorithms for fusing ToF camera's ego-motion with IMU data. The final section gives review of the related work where a CCD camera images and IMU data are fused to estimate the trajectory of a robot or vehicle.

2.1 ToF Camera Ego-Motion Estimation

The ToF camera is a compact sensor that produces 3D range images and amplitude images at a video frame rate [8]. 3D range images are in a point cloud form arranged in x, y and z axes of the ToF camera, while the amplitude images give the amplitude of the reflected light at each pixel of the camera, for more details see appendix A. The 3D

range and amplitude information are obtained by using the phase-shift principle [9]. A scene is illuminated with a modulated Near Infrared (NIR) light, the reflected light is measured and the phase-shift computed is proportional to the distance traveled by the light (see Appendix A for ToF measurement principle). There are two ToF cameras that are generally used in robotics. These are camCube from pmdTechnoloties4®[1] and Swiss Ranger from Mesa Imaging[2]. This discussion focuses more on the Swiss Ranger ToF cameras, even though both cameras use the same principle to measure distance. These sensors tend to have different characterics based on the number and the arrangement of the modulated NIR light sources [10].

The accuracy of these sensors is limited by errors which are caused by the measurement principle, the architecture of the sensors and the environment (background light and reflectivity of objects). The errors affecting the accuracy of the sensor are described below, followed by various methods which can be used to reduce these errors.

2.1.1 Time of Flight (ToF) Camera Errors

The errors of ToF camera can be divided into systematic and non-systematic errors as shown in [11] [12]. Systematic errors refer to those that behave in a systematic way which make them easy to model and correct through calibration.

2.1.1.1 Systematic Errors of ToF Camera

The most common systematic errors are:

- **Distance-related errors**: These errors are caused by the assumption that the emitted NIR light is perfectly sinusoidal, which introduces errors that are related to the distance between the object and the camera. These errors follow a sinusoidal form in which the amplitude can vary between 60 mm and 160 mm [13].
- **Inhomogeneous illumination**: These errors exist due to the LEDs configuration. Pixels at the boundary receive less reflected light compared to the centre pixels and this affects the distance accuracy as it is dependant on the amount of reflected light measured on the sensor [9].
- **Amplitude related errors**: The intensity of the reflected light depends on the reflectivity of the object. This causes objects that are at the same distance with

[1] http://www.pmdtec.com/
[2] http://www.mesa-imaging.ch/

different reflectivity to have different distance measurements.

- **Fixed pattern noise**: This is an offset that is different at each pixel and is caused mostly by the manufacturing process.

Systematic errors can be reduced or corrected by calibrating the sensor. Two forms of calibration are possible for the ToF camera, that is photogrammetric and distance calibration. Photogrammetric calibration is possible using variants methods, one such method is shown in [14] and is possible since amplitude images can be modelled using the pin hole camera model. Photogrammetric calibration estimates intrinsic parameters such as the focal length, optical center and lens distortion of the ToF camera.

Due to certain properties of the ToF camera, there are challenges in performing a normal photogrammetric calibration with ToF camera images. Due to the low resolution, it is difficult to find a definite corner point of square in a checkerboard which is generally used for calibration [14]. In addition, because of the small apex angle, large features occupy too many pixels and cannot be placed too far from the ToF camera because errors will be high [15].

These challenges can be solved by using a board with NIR LED lights as target features instead of a checkerboard like in [15]. In [16], a normal checkerboard was used for photogrammetric calibration. Definite positions of the corners were found by increasing the resolution of the image using bi-linear sampling methods followed by the histogram normalisation of the images. Patti [17] used a white test field with black circular targets at different distances to estimate intrinsic parameters.

Distance calibration has been performed by modelling the error model at the camera measurement range. Kahlmann *et al.* [15] built a look-up table to model the distance error. In [16], the error was modelled by fitting a cubic B-spline while in [12][13] distance error was modelled by fitting a M-penalised spline. These calibration methods were necessary in the early generation on ToF camera model since the error was fluctuating with a maximum accuracy of 160 mm compared to the current generation with fluctuating error of amplitude 10 mm [17]. Piatti [17] performed similar experiments on an SR4000 and modelled the distance error with a sinusoidal model. According to the results, the model was producing more erroneous results for some section and this proved that manufacturers calibration has removed all possible systematic errors.

2.1.1.2 Non-Systematic Errors of ToF Camera

Non-systematic errors are highly dependant on the scene configuration which implies that they cannot be corrected during calibration but have to be detected online. Some of the dominated non-systematic errors are:

- **Multipath reflection**: These errors occur when light travels through multiple paths before returning to the sensor causing superposition errors.
- **Jump edge points**: These points occur when the transition between the foreground object and the background object is sudden as the ToF camera measures a smooth transition as seen in Figure 2.1.
- **Light scattering**: Background lights are measured with the reflected NIR sensor light which causes objects to appear closer than they actually are. This is the reason why ToF cameras cannot operate in direct sunlight.
- **Phase wrapping/Aliasing**: The maximum distance that a ToF camera can measure is limited by the modulation frequency. Objects that are further than the maximum distance are wrapped back into the ToF camera measurement range as shown in Figure 2.2.
- **Noise**: Lange [9] divides the noise that affects the accuracy of ToF camera into; photo charge conversion noise, quantization noise and electronic shot noise. While other forms of noise can be removed, electronic shot noise can not be removed and it is the dominant one.
- **Motion blur**: This occurs when the ToF camera or objects in the field of view are moving. It is highly noticeable when the camera has high angular velocities which show that reflected light is measured back to wrong pixels.

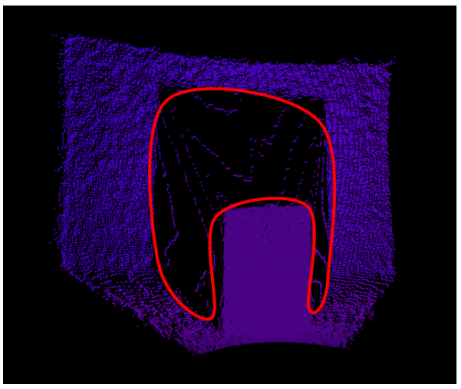
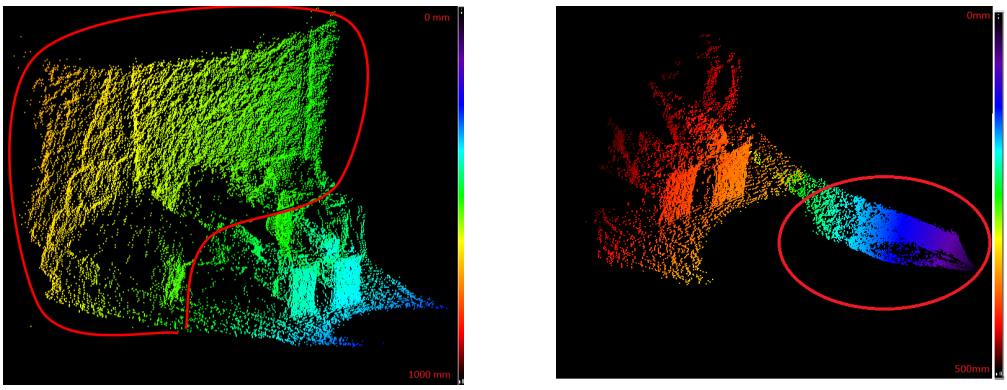


Figure 2.1: A point cloud illustrating jump edges bounded by the red line



(a) (b)

Figure 2.2: (a) shows the point cloud of an environment where the maximum distance was set to 10 m and (b) shows the point of the same environment where the maximum distance was set to 5 m. Wrapped points are enclosed by a red shape.

The errors that remain after calibration are mostly non-systematic and can be detected, rejected or corrected online during the application. May *et al.* [11] and Piatti [17] proposed methods for finding the jump edge points by considering eight neighbouring pixels. In [11], a point is classified as a jump edge if the angles between that point and the neighbouring points spanned at the focal point of the camera are greater than some predefined threshold angle; while in [17], a point is classified as a jump edge if it differs from one or more of its neighbouring points by a certain threshold which is computed using the Ground Sample Distance[1] of the ToF camera. These algorithms perform similarly but the method proposed by [11] is sensitive to noise and therefore the me-

[1]Ground Sample Distance refers to the distance between the pixels centers measured from the ground

dian filter must be applied to prevent useful points from being classified as a jump edge.

The phase wrapping problem can be solved by using the ratio of the amplitude and the distance [18]. The downfall of this method is that if some points that are wrapped, belong to objects with high a reflective surface they will not be picked up. This will then cause error in the ego-motion estimation. Droeschel *et al.* [19] proposed a method that uses probabilistic approaches to find phase jumps in the depth map. The limitations of the algorithm occur when all the measurements are wrapped and since the gradient of the surface is used, it will not be able detect phase wraps. The second limitation is that the algorithm cannot compute the number of phase wraps and it is just assumed to be one. These limitations can be removed by using multi-frequency modulation as shown in [20]. For multi-frequency to be applied in real-time, two ToF cameras are necessary.

Instead of performing calibration, erroneous distances can be removed by rejecting pixels with low amplitude values. This method usually rejects all points in the boundary pixels because of the non uniform illumination [11]. This confidence map is useful for application where there is no motion. The SR4000 provides the confidence map as one of its outputs. The map gives the confidence of each of the pixel measurements using the distance and amplitude temporal variations [8]. Reynolds *et al.* [21] proposed a supervised learning algorithm using random forest regressor that is trained using 2D laser scanner to provide groundtruth data. The algorithm was applied on SR3000 and tested in a different environment from which the training data was collected. The algorithm performs well and it can detect jump edges as well. The limitation to the algorithm is that it is time consuming since the confidence for each pixel has to be computed.

In [17] and [15] it was shown that temperature changes have an effect on the accuracy of distance measurement with distance variance of up to 6 mm. Piatti suggests through experiments that a minimum of 40 minutes is needed for warm up time so that the measurements can be stable.

2.1.2 Ego-Motion Estimation

Ego-motion refers to the process of estimating the trajectory of a camera. This is useful in localising a mobile robot. The estimation of the camera's trajectory is performed by

registering two consecutive range images to find a rigid transformation (rotation and translation). This has been successfully applied to a 3D laser scanner in underground mines to estimate the pose of the camera as well as mapping the environment [22]. The success of 3D laser scanner is due to high measurement precision, accuracy and a wide field of view.

ToF cameras are favourable because of low power consumption and high frame rate. The widely used algorithm to register 3D range images is the Iterative Closest Point (ICP) [3][4]. There are two forms of ICP which are point-to-point [3] and point-to-plane [4]. A point-to-point ICP was applied in [22] on a 3D laser scanner range images in a mine environment for map building and localisation. A point-to-point ICP [3] finds the rigid transformation by iteratively minimising the Root Mean Squared (RMS) error between the corresponding points, while the point-to-plane ICP [4] finds the rigid transformation by minimising the RMS error between points in one range image and the tangent plane of the corresponding points in the other range image. In [23], a more thorough study of the ICP algorithm with its variants are presented; here the ICP algorithm was divided into six stages and different variants were compared in terms of convergence speed and accuracy.

In general each iteration of point-to-point ICP is faster than the iteration point-to-plane ICP but point-to-plane converges with fewer iterations. ICP has the tendency of being trapped in local minima. This can be avoided by providing ICP with an initial guess or by applying ICP to images with small relative transformation [24]. This is possible with ToF camera since they have a high video frame rate (max: 54 frame per second (fps) [8]).

One of the first applications of ToF camera in mobile robots was used in navigation. The Swiss Ranger was used for obstacle avoidance [25]. A SR2000 was used and it proved to be more advantageous compared to a 2D laser scanner. The first application in trajectory estimation on a mobile robot was done by Ohno *et al.* [26]. An SR2000 ToF camera was used and point-to-point ICP was modified in order to handle ToF errors and to be applicable in real-time applications. The correspondence between two range images was assumed to be the closest point in the other range image and it was found by Approximate Nearest Neighbour (ANN) search algorithm. The ANN algorithm was applied on edge points which are extracted from intensity images using the prewitt operator. To make the ICP algorithm more robust, mismatched correspondences were

rejected using adaptive distance threshold and surface normal angle difference threshold. The distribution of 3D points was used to estimate the initial transformation to avoid being trapped in local minima.

The algorithm was applied in a snake-like mobile robot moving in a 2D planar environment. A motion capture system with an accuracy of 10mm was used to provide groundtruth data. For a simple one degree of freedom (DoF) motion the average error was 17% for translation and 15% for rotation. The algorithm produced erroneous results when the robot moved in a rubble environment. The error was mostly caused by rotational motion. The reason for the bad results was due to the fact that ToF camera was not calibrated and edge points were used to estimate transformation. Edge points of the ToF camera are erroneous due to jump edge points.

May *et al.* [11] performed a thorough study on the applications of the SR3000 ToF camera in map building and trajectory estimation of a mobile robot. To get accurate ToF images, photogrammetric and distance calibration was performed as in [12][13]. A point-to-point ICP [3] was modified for the application. KD-trees [27] were used to find the corresponding pairs assuming that the nearest point on the other image is the corresponding pair. Frustum culling was used for rejecting mismatch and non overlap points. Frustum culling is a method for finding points which are in a new range image but are not present in the previous range image and vice versa. Compared to the normal ICP algorithm, frustum culling ICP produces more accurate results especially when the transformation consists of large angles.

In [11], two point feature-based tracking estimation algorithms were implemented. These algorithms operate by extracting point features from consecutive intensity images, and using their corresponding 3D points to estimate the transformation. A least square algorithm by Arun *et al.* [28] is used to compute the transformation. Scale Invariant Feature Transform (SIFT) [5] and Kanade Lucas Tomasi (KLT) [29] algorithms were implemented and the mismatches were found using RANdom SAmple Consensus (RANSAC) [30]. The limiting factor with point features is that ToF images have low resolution meaning that there are few features which are detected.

A direct registration Efficient Second order Method (ESM) visual tracking [31][32] algorithm was also implemented for trajectory estimation. Instead of tracking point features, ESM tracks both rigid bodies and non-rigid bodies in the images. The algo-

rithm performs better than SIFT and KLT algorithms in rotation motion but suffers in translational motion. This is due to distance measurement errors.

In [11], experiments were conducted where the ToF camera was mounted on an industrial robot end effector. This was done to obtain groundtruth but it also limited the motion of the camera to the workspace of the robot. The ToF camera was moving in a circular path with a diameter of 180 mm while the camera was facing outwards. In terms of accuracy, point-to-point ICP was rated the best followed by a SIFT based ego-motion estimation. Both SIFT and KLT were very fast compared to ICP. This is due to the fact that no matching of 3D points is done since its 2D point features are tracked in the intensity images. To increase the accuracy, loop closure was implemented. This requires the robot to return to a previously visited location and it must be able to detect that it has been there before.

Piatti [17] proposed a multi-frame registration algorithm that averages about 30 frames per camera pose. The measurements are captured while the camera is stationary to increase the accuracy. Amplitude images of SR4000 were used for point feature detection and tracking was done using the Speeded Up Robust Feature (SURF) [33] algorithm. SURF algorithm was picked over SIFT because it uses a vector of length 64 instead of 128 to describe a feature which makes it faster. In an effort to make the algorithm more robust, saturated pixels were replaced by a bilinear interpolation using eight neighbouring points. Rejecting these points would limit the SURF algorithm since ToF camera has low resolution. The outliers in the matched features were rejected using least median squares estimator [34] while jump edges were rejected using mixed pixel removal [17]. The transformation was estimated using a least square solution.

To test the algorithm, ToF camera was mounted on a calibration bar and moved in a straight line at an interval of 0.2 m while maintaining the orientation. The objects in the ToF camera's view were within a 1.4 m range such that a sinusoidal error model [17] for systematic errors could be applied to reduce errors. For a single 0.2 m step, translation error was 0.013 m and rotational error was 0.35°. The algorithm was only tested for translation motion which means the results are not quantitative. The limitation to the algorithm is that the ToF camera must be stationary to capture approximately 30 images which is time consuming.

Similar to [17], an SR3000 ToF camera was used for Simultaneous Localisation and

Mapping (SLAM) [1] [35]. A SURF [33] algorithm was implemented to detect and track point features between two consecutive images. A 3D point of the feature was computed by interpolating four neighbouring 3D points around the feature. The distance errors were corrected by a model that relates to the standard deviation of the 3D point of the pixel with the intensity reading. The distance error model corrected scaling and offset errors. The relative transformation was computed using Arun's least square algorithm [36] and RANSAC [30] was used to reject outliers in the mismatch of the features.

The experimental setup consisted of the ToF camera mounted on a push trolley. Data was collected in an environment similar to a search and rescue environment. The trajectory of the camera compared to the groundtruth showed that the algorithm performed well and loop closure increased the accuracy. Only the graphical representation of the trajectory estimation was presented in [35].

In the literature, point-to-plane ICP [4] is rarely implemented with the ToF camera for measurements. The reason for this is that they produce noisy measurements. To implement point-to-plane ICP, surface normals have to be computed, and the algorithms used are the least squared methods [37] which are sensitive to noise and are computationally expensive. A fast surface normal computation algorithm that computes the normals using the derivatives of the surface can be found in [38] but point clouds must be converted to spherical coordinates. A plane in a range image can be computed by the algorithm proposed in [39]. The algorithm randomly selects a point with its neighbouring points and computes a plane using least squares algorithm [37]. Points around are iteratively included to the plane region if their distance to the plane is less than a certain predefined threshold.

Alternative methods to ICP for trajectory estimation using 3D range images have been proposed in [40] [41] [42]. Pathak *et al.* [40] proposed a method for ego-motion estimation suitable for environments with large number of surface planes. The algorithm computes surface planes from ToF images using the algorithm proposed in [39]. They call the algorithm Minimally Uncertain Maximum Consensus (MUMC) because of the approach it uses to find planes correspondences between two consecutive range images which is similar to RANSAC. To find correspondences, initially planes are grouped in two pairs and their corresponding consensus of the rotation and translation are com-

[1] SLAM refers to simultaneously mapping of an unknown environment while localising the robot within the same map at the same time

puted using a least squares solution. The pairs with the largest consensus is selected as the correct planes correspondences. The final transformation is then computed using the correct correspondences and the computation of the rotation and the translation components are separated. Rotation is computed using a Davenport q-method [43] which is based on a quaternion. Translation is computed using least square method and it requires at least three non-parallel planes to compute translation in three axes while rotation computation requires at least two non-parallel plane pairs.

The MUMC [40] algorithm was tested on SR3000 ToF camera in an environment with many planes. The algorithm was compared to point-to-point ICP [3], point-to-plane ICP [4] and 3D Normal Distribution Transform (NDT) [44], which is explained later in this section. The MUMC algorithm performed better than the other three. Point-to-point ICP produced the worst results. MUMC converged faster and required less memory. Point-to-plane ICP seems to work better than point-to-point ICP. This was because the point-to-point ICP algorithm was being trapped in local minima because it does not take into consideration the geometry of the point cloud. Another advantage of the MUMC algorithm is that it also produces the uncertainty in the estimates of the computed pose. The algorithm is limited to an environment that has a large number of surface planes.

In [41], an algorithm that estimates ego-motion simultaneously with segmentation by curve fitting was proposed. The problem was formulated with an energy function that has three terms: *conformity of ego-motion, non-conformity of ego-motion and regulation of the curve*. The energy function was minimised using a variational approach with the assumption that the objects in the field of view are stationary. The variational approach algorithm iteratively solves for the transformation between two images using the steepest descent method. Curve fitting for segmentation is solved using the level set method. This is performed until both parameters converge.

The variational approach algorithm [41] was tested in an indoor environment using the SR3000 ToF camera. The experiments consisted of a person and ToF camera both moving. The algorithm was able to estimate the velocity of the camera with an accuracy of 80% after 100 iterations. The algorithm is promising especially for situations where the robot is operating in an environment where objects are moving. The algorithm was tested on few image frames while the camera was in translation movement. Since no rotational motion was tested and the actual camera trajectory was not represented, it

is not possible to compare the algorithm with the ICP based methods. The velocity groundtruth was obtained from tracking the features in the ToF images which is not accurate since the ToF camera is characterised by noise that limits its accuracy.

Villaverde and Graña [42] proposed an algorithm for ego-motion estimation using artificial neutral networks. The algorithm has three stages where the first one is preprocessing. Range images are filtered for distances with phase-wrap. The product of the amplitude and distance is used to detect pixels with phase wrap since the product of amplitude is approximately constant for pixels with no phase-wrap. In the second stage, neural gas networks [45][46] perform vector quantisation on the filter range images to produce codebooks. The produced codebooks have a specific size and try to maintain the captured objects and spatial shapes in the field of view of the camera. Finally, the last stage uses the codebooks from consecutive images to estimate the transformation using the evolution strategy module. The evolution strategy first finds the corresponding points between the two codebooks using KD-trees [11] and estimates the rotation and the translation by median fitting the correspondence points.

In [42], data for the experiments were recorded in an empty room while the robot moved against the wall. Therefore, the groundtruth could be estimated by the room dimensions. The algorithm used was compared to the variants of ICP [3] algorithms. The Neural Gas algorithm produced the best results. The ICP methods produced worse results due to the fact that they were not modified to handle the ToF data. The total accumulative error for the neural gas algorithm [42] was 794.266 mm while the error for one of the variants of ICP was 3419.386 mm. The algorithm produced more erroneous results when the motion consisted of rotational velocity and it was more computationally expensive compared to the ICP based algorithms. Other drawbacks of the algorithm is that it cannot handle small overlaps. The algorithm assumes that the robot is moving in a 2D planar surface, which limits the application.

Underground mines present lots of serious or even fatal risks to workers [47][48][49]. Autonomous mobile robots are being utilised for safety inspections of the mine before miners can be sent underground. In [22] and [44] methods for 3D scans produced by a laser scanner for scan registration of mine data were presented. Nüchter *et al.* modified the point-to-point ICP [3] algorithm for the implementation. The modified algorithm used KD-tree for fast matching of correspondence points. A combination of a median and reduction filter was used to reduce the number of ICP points but leaving enough

so that ICP would not get trapped in local minima. To ensure a global consistency, odometry estimates were used as initial guess for the ICP algorithm.

In [44], scan registration algorithm for mine data was performed using the Normal Distribution Transform (NDT). NDT is a method for registering two 2D scans that was developed by Bider and Straßer [50]. The NDT algorithm represents the point cloud as normal distribution in order to solve for the transformation unlike ICP which uses points. The first step of the algorithm is to divide the scans into cells. In each cell, where there is a minimum required number of points, the mean vector and its covariance matrix are computed using the surface normals. In each cell, a probability density function is computed using both the covariance and the mean vector. This describes the probability of finding a point in a specific location within the cell. The probability density function is used to compute the transformation between the two scans using Newton's method coupled with the line search [44].

The method proposed by Magnusson *et al.* [44] is an extension of the 2D NDT proposed in [50] to 3D. The algorithm was compared to point-to-point ICP algorithm with KD-trees. The 3D NDT algorithm performed better than ICP in terms of speed and accuracy. The scans were taken at an interval of 4 m using the stop and scan method. The algorithm success is due to the nature of the environment in which the data was collected that is the surfaces were planar.

2.2 Fusion on ToF and IMU

Data fusion is the process of combining measurements from different sensors in order to estimate a robust and complete description of state interest of which in our case it is the pose of a mobile robot. It has a wide range of applications in robotics. This literature review is limited to the fusion of visual and inertial sensors in the application of localisation of a mobile robot by estimating the ego-motion of a vision-inertial system. Corke *et al.* [51] gave an introduction to fusion of visual and inertial sensors. They show that these sensors complement each other in estimating robot motion. The main advantage is that they do not require an external infrastructure, meaning that they can operate in unknown environments. Inertial sensors have higher derivative orders, that is, angular velocity of order one and linear acceleration of order two thus making it suitable for fast motion and a camera of order zero suitable for slow motion. By fusing the two sensors, a more robust sensor capable of operating in fast and slow

motion is developed. Most methods used in data fusion make use of stochastic techniques [52][53], with Kalman filters [54] and Particle filters [53] being the two main ones.

The Kalman filter is an optimal recursive filter that estimates the state of linear systems at a given time from the estimated state of the previous time and the current measurements. It is optimal in the sense that it tries to minimise the estimated error covariance of the estimated state. The Kalman filter assumes that both the dynamic process and measurement models are disturbed by a zero-mean gaussian noise. Most real world systems have some form of non-linearities either in the dynamic process model or measurement model, or both. In this case, the Extended Kalman Filter (EKF) [7][55] is employed which linearises the models around the previous state estimates by using a Taylor series based approximation.

The Particle filter [53] is a recursive implementation of the Monte Carlo method that describes state estimates as the probability distribution using a set of weighed samples of an underlying state space. Particle filters are suited for problems where dynamic process and measurement models are non-linear and the noise is non-gaussian [53]. The accuracy of a particle filter is directly related to the number of samples chosen to represent particles. The sample number does have a negative effect on the computational time. The Particle filter is limited by high dimensional state estimates, since the number of samples required for a certain model increases exponentially with state dimensions, thus increasing the computational time.

Kalman filter based fusion is favoured because of the high dimensionality in the estimates of tracking a pose using vision and a Inertial Measurement Unit (IMU). In addition, noise in camera and IMU measurements can be assumed to be gaussian.

2.2.1 Camera-IMU Calibration

Before the fusion of two sensors can be performed, a 6 DoF transformation between the two sensors must be known. An error in this transformation introduces errors which result in the inaccuracy of state estimates. One of the ways of estimating this transformation is by using dimensions from technical drawings assuming that the mounting of these sensors is accurate because it might introduce errors. An additional sensor can also be used to find the transformation just like in the Mars Exploration Rover mission [56] where a 3D laser scanner was used. The problem is that 3D laser scanners

are expensive.

Lobo *et al.* [57] proposed an algorithm for finding the transformation between the IMU and camera which is a modified hand/eye calibration [58]. This is used to find the transformation between the camera and a robot manipulator end effector. The algorithm separates the transformation into rotation and translation. Rotation is computed by allowing both the camera and IMU to measure the vertical direction. IMU measures vertical direction by measuring linear acceleration which is equal to the gravitational acceleration when the unit is stationary. A checkerboard is placed vertically such that the camera can use the corners of the squares to find the vertical direction. Horns' method [59] is then used to find the rotation between the IMU and camera frames in a quaternion co-ordinate frame. Lobo *et al.* [57] solved the translation part by using a turntable and a position rig that can be adjustable such that the IMU-camera unit rotates about the IMU center. Translation can be computed by using the camera measurements since the unit will be rotating at the IMU center.

The drawbacks of the algorithm are that it does not provide an accurate measurement of the estimated transformation. Secondly, separating the computation of rotation and translation, causes the rotational error to propagate to the translation estimates. Thirdly, this algorithm does not incorporate noise characteristics of the IMU such as acceleration and gyroscope biases. These drawbacks were eliminated by Mirzaei *et al.* [60] and Hol *et al.* [61].

Mirzaei *et al.* [60] solved the problem of finding the transformation by deriving an EKF based algorithm that does not separate the computation of rotation from translation and also estimates IMU biases. The algorithm also gives a level of accuracy of the estimated transformation in a form of a covariance matrix. The IMU measurements are used as inputs to the EKF and the camera measurements are used as observations. Because of the high non-linearity in the measurement model, Mirzaei used Iterative EKF [7] to decrease the inaccuracy yielded by non-linearities of the measurement model. The algorithm does not require any additional equipment except the checkerboard placed vertically. The initial value of the transformation which is used in the EKF is hand-measured. In the initialisation stage of the algorithm, the IMU-camera unit is placed in a static position for approximately 180 seconds to compute the IMU biases and the initial state covariance by forcing zero linear and angular velocities. The algorithm was able to find the transformation with standard deviation of 0.1 mm

and 0.05° for translation and rotation respectively on a real data experiments. The limitation of this algorithm lies in the linearisation of the non-linear system. There is also no accurate way of placing the checkerboard such that it is perfectly parallel to the gravity vector.

Hol *et al.* [61] solved the problem similarly to Mirzaei. The difference was that instead of computing the state error using Kalman gain of EKF, the innovation of EKF was used to form an optimisation problem which falls under system identification. It was then solved by using gray-box system identification [62][63]. Innovation refers to the difference between the EKF estimated features locations, which is a function of the transformation and the feature location as observed by the camera. A checkerboard which is placed horizontally is used to provide features for the camera. The algorithm also estimates the gravitational vector which is something that Mirzaei only computed at the initialisation stage. The algorithm performed in a similar way to the Mirzaei algorithm with a final standard deviation of 0.9 mm and 0.04° in translation and rotation respectively. The same limitation of not being able to place the checkerboard perfectly horizontal, introduces errors in the algorithm.

2.2.2 Camera-IMU Fusion

The ToF cameras are sensors that provide range and amplitude images at the video frame rate. These cameras can facilitate in a 6 DoF trajectory estimation since they can provide 3D information of the environment [26][11]. These sensors suffer from motion blur because of the manner in which they measure distance, which causes errors when estimating the trajectory. The algorithm which is mostly used for estimating the trajectory is Iterative Closest point (ICP) [4][3]. ICP has been shown to get trapped in local minima, especially when the motion consists of high rotational velocities.

Even though the ToF camera ego-motion estimation has been studied for a decade now, there is very limited literature on the topic of fusing it with IMU sensor data in order to increase the accuracy of the trajectory estimation. To our knowledge, Droeschel *et al.* [64] are the only researchers that have fused ToF camera and IMU data. An SR3000 ToF camera from Swiss ranger with Xsens MTi IMU was used. These sensors were mounted rigidly to a mobile platform which was moving on a 2D planar surface which simplified the problem to 3 DoF trajectory (position x, y and orientation θ).

Droeschel *et al.* [64] used a linear Kalman filter to predict the robot velocity estimates using IMU measurements as input and motion estimates from the ToF camera as observations. Motion estimates of the camera were computed by applying the SIFT [5] algorithm on amplitude images to find corresponding features between consecutive images. The 3D points were then used to compute the motion estimates using Arun's method [36]. This method uses least squares fitting to find the transformation in the form of a rotation matrix and translation components $(\mathbf{R}, \mathbf{t})$. The algorithm was tested by moving the mobile robot in a 1.2 m square planar lab with a background such that SIFT could detect features. Their results showed a huge improvement on the trajectory estimation compared to the trajectory estimated using the ToF data only. Fused results showed that the IMU improved the estimation especially when the robot was going around a corner. Errors were reduced by 1006 mm in translation and 25.4° in rotation. Groundtruth was found by applying ICP algorithm in the data provided by the 2D laser scanner since the robot was moving in a 2D planar.

An extension to the work of Droeschel *et al.* could be to develop a full 6D trajectory estimation algorithm that uses the EKF and develop a ToF camera error model to be used for the measurement covariance since the RMS error from registration algorithm was used. The 6D transformation between IMU and ToF camera must be computed as well.

A similar system was designed by Hesch *et al.* [65] where the data from a 2D sick laser scanner was fused with IMU data to estimate the trajectory in 6 DoF. The system operates in an indoor environment in which the map is known prior to the experiment. The system estimates the position and orientation by employing EKF. The IMU measurements are used to estimate the position and orientation while the laser scanner measurements correct the estimates. If the system is stationary, zero velocity update is used instead of using laser scanner measurements. The zero velocity update stage comprises setting the linear and angular velocities to zero. At the same time the IMU drift is corrected without violating the covariance of the estimates. To determine whether the system was stationary, Mahalanobis[1] distance is computed using residual and innovative covariance. If the Mahalanobis distance is less than a certain threshold then it is stationary.

Hesch *et al.* modified the calibration algorithm by Mirzaei and Roumeliotis [60] to

[1]Mahalanobis is a distance based on the correlation between variables or the variance-covariance matrix

find the transformation between the 2D laser scanner and an IMU. The initialisation of the state estimates was performed by placing the system in a static position while performing zero velocity updates. The algorithm was tested on a closed loop corridor with the total length of approximately 120 m for the period of 8.5 minutes. A maximum standard deviation of 9.16 cm and 0.1° in translation and rotation was achieved respectively. Even though this algorithm performed better, the fact that the 3D map of the environment must be known before hand is a limitation. This problem can be eliminated by using a sensor which provides 3D range information such as a 3D laser scanner or ToF cameras.

In this chapter, work related to trajectory estimation of a mobile robot by fusing ToF camera's ego-motion with IMU data was presented. The review was limited to Swiss ranger ToF camera and a low cost IMU. Because the mobile robot will be operating in underground environments that have scarcity of unique landmarks, are GPS denied, are slippery and possibly absent of light, this limits the kind of methods which can be used. Since there are no unique features or landmarks in underground environment, algorithms such as SIFT [5], SURF [33] and KLT [29] are not applicable. Algorithms such as MUMC [40] and NDT [50] require lots of planar surfaces which is not always the case in underground mines. ICP based algorithms are favourable for these environments and they have been applied before using 3D laser scanners [22]. Hence, ICP algorithms will be investigated further and SIFT algorithm was implemented to show that it is no effective underground.

The fusion of the two sensors is performed using Kalman filter algorithm. Before fusion is performed it is necessary to find the transformation between the two sensors. Two algorithms with similar setup were derived by Mirzaei *et al.* [60] and Hol *et al.* [61] which are EKF based algorithms. The algorithm by Mirzaei *et al.* is preferred because it would be easy to modify it to estimate the trajectory of the mobile robot. A fusion algorithm that fuses ToF camera's ego-motion with IMU data was presented by Droeschel *et al.* [64], but since they used SIFT algorithm for ToF camera ego-motion estimation, this algorithm is not suitable for underground mining environments. The algorithm by Droeschel does not consider the transformation between the two sensors, which also introduces errors.

2.2.3 Related Work

Fusion of the normal CCD camera images with IMU data is a well destabilised topic in robotics. It has some similarities with the fusion of a ToF camera and IMU data in terms of the ego-motion estimation. In essence, some of the algorithms used in normal CCD cameras can be adapted to fuse ToF camera data with IMU data.

Hol *et al.* [66] fused camera data with IMU data to track the trajectory of the camera. The EKF algorithm was used for fusion. The IMU measurements were used as inputs to estimate the position and orientation of the camera and the camera measurements are used for error correction. Because the CCD camera does not provide 3D information about the environment, it is a challenge to estimate a 6D trajectory without a known map or scene model. In the experiments, Hol *et al.* used a scene model consisting of two perpendicular planes. The camera and IMU were rigidly mounted on an industrial robot end effector that can move in 6D. The robot has an absolute accuracy of 2 mm and repeatability of 0.2 mm in translation. To test the algorithm, the sensor unit was moved in an eight-shaped motion for 5.4 seconds in a trajectory of approximately 2.6 metres. During the experiment, the scene model was always in the view of the camera and the absolute accuracy of 20 mm and 1° in translation and rotation respectively. The industrial robot trajectory was used as the groundtruth. Hol *et al.* [66] showed that fusion has the benefits of making camera ego-motion estimation robust even in situations were the scene model is out of the camera view. In addition, it reduces the computation required for the ego-motion compared to using the camera only. This algorithm can therefore be applied in real-time.

Similarly, Armesto *et al.* [67] performed the fusion of camera data with IMU data in order to estimate the trajectory of the IMU-camera system using EKF. To obtain the groundtruth trajectory of the system, the IMU-camera system was rigidly mounted on an industrial robot. A planar checkerboard was used to provide features for the camera. The algorithm used different linear and angular velocities and it produced good results. The results showed that at low velocity, the algorithm relies more on the camera and on the IMU for fast motion as discussed in [51].

An application of IMU-camera trajectory estimation where the scene model or map is unknown is presented by Mourikis and Roumeliotis [68]. An EKF based method was used to fuse IMU measurements with the camera observations. IMU measurements are

used to predict the system's position and orientation while the camera observations are used for correction. The differences that set this algorithm apart from the others lie in the measurement model. Instead of stacking camera observation per camera pose, they were grouped according to the features that were being tracked. By using a series of tracked features, the geometric constraint is obtained which does not require any 3D information about the features, which is why no map or scene model is needed. This reduces the computational cost to a point where the algorithm is able to operate in real-time. The algorithm was tested by collecting data while the IMU-camera system was mounted rigidly on a car moving through the streets of a residential area where the map is available for evaluating the results. The trajectory was 3.3 km long and the experiment lasted nine minutes. The final translational error was less than ten metres, which was measured by finding the difference between an estimated end position and the measured end position.

Chapter 3

Design and Analysis

The design and analysis of the subsystems that are used for ToF-IMU trajectory estimation system are discussed in this chapter. Firstly, the preprocess steps of the ToF camera that handles some of the errors are described. Secondly, algorithms for the ToF ego-motion estimation that are considered for this dissertation are detailed followed by an Inertial Navigation System (INS) algorithm for tracking IMU. Finally, the method for fusing ToF ego-motion with IMU data will be discussed. The overall system is illustrated by Figure 3.1.

By critically analysing the literature review, ICP algorithms are more suited for underground environments because of unstructured background and no unique features or landmarks in underground mines. Hence, in our design, more concentration will be paid on the ICP algorithms. Because ToF camera's images are corrupted by motion blur while the ToF camera is rotation, the ToF ego-motion will be fused with the IMU measurements using the Kalman filter algorithm. Motion blur in the ToF camera images is caused by the architecture of the camera and can not be removed using normal image processing techniques. In this dissertation, it is addressed by fusion ToF ego-motion with IMU measurements.

3.1 3D ToF Preprocesses

The preprocessing step is applied on all the ToF images to reduce the error in the ego-motion algorithms. In this dissertation, a SR4000 [8] from Swiss Ranger is used and a summary of the specifications are presented in Table 5.1 in Appendix A. This is the fourth generation of ToF cameras from Swiss Ranger and most of the systematic

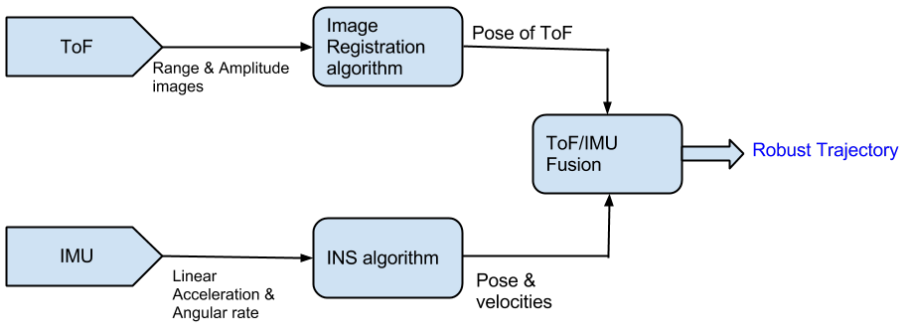


Figure 3.1: Overview of the system for the trajectory estimation

errors, such as the internal scattering which was an issue in third generation [17] have been reduced. Piatti [17] modelled the systematic errors of the SR4000 ToF camera and showed that it does not improve the accuracy of the distance measurements and for distances greater than 5 meters the modelled added errors. In this dissertation, only manufacturer's calibration is used. Since SR4000 does not have temperature compensation, a 40 minute warm-up time is allowed before any experiment is performed, as suggested by Piatti [17]. Methods used for handling non-systematic errors such as jump edge points, phase wraps and non-uniform illumination are described in the following subsections.

3.1.1 Jump Edge Filter

Jump edge points occur when the transition between background and foreground objects is sudden whereas the ToF camera measurements show a smooth transition as shown in Figure 3.3(a). If these points are used for ego-motion estimation, they produce erroneous results since they are random. A jump edge filter was adopted from [11] due to its simplicity and accuracy in finding jump edge points. Assume that $\mathbf{P} = \{\mathbf{p}_i \in \mathbf{R}^3 \mid i = 1, \cdots, N\}$ represents a 3D point cloud from the ToF camera. To check if a point $\mathbf{p}_i$ is a jump edge, a set of eight neighbouring points $\mathbf{P}_m = \{\mathbf{p}_{i,m} \mid m = 1, \cdots, 8\}$ are extracted from the point cloud $\mathbf{P}$. The angle θ_i between $\mathbf{p}_i$ and neighbouring points $\mathbf{p}_{i,m}$ with third vertex at the focal point of the camera (refer to Figure 3.2 for an illustration) is defined mathematically as

$$\theta_i = \max_{m=1:8} \arcsin\left(\frac{\|\mathbf{p}_{i,m}\|}{\|\mathbf{p}_{i,m} - \mathbf{p}_i\|} \sin\varphi\right) \tag{3.1}$$

where φ is the apex angle of the two neighbouring pixels. An experiment was set-up to test the jump edge filter and the filtered image is shown in Figure 3.3(b). The angle is compared to the threshold angle to determine if a point is a jump edge. The threshold angle is determined experimentally depending on the ToF camera. An experiment was setup where different angles were used and 175° gave good results hence for this dissertation a threshold angle of 175° is used.

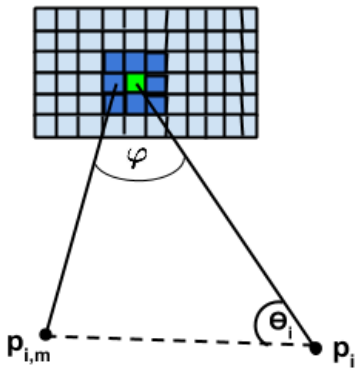
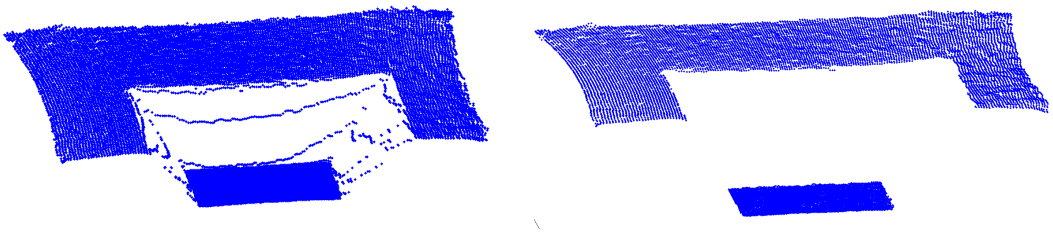


Figure 3.2: Diagram illustrates how a jump edge filter works. The image pixels are shown with eight neighbouring pixels in blue and a point $\mathbf{P}_i$ in green.

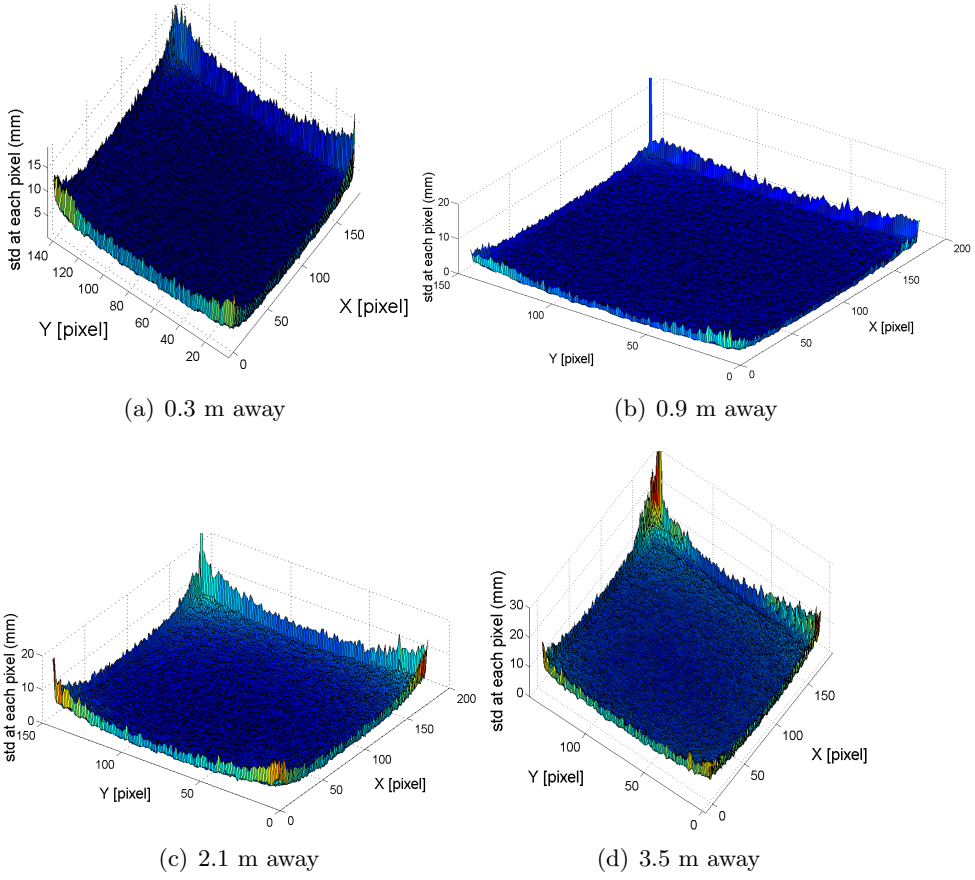


(a) (b)

Figure 3.3: (a) shows a point cloud with jump edges as they can be seen between the transition and (b) shows a point cloud with jump edge points removed using jump edge filter.

3.1.2 Inhomogeneous Illumination

Due to the arrangement of LEDs that emit modulated NIR light, pixels on the boundary of the image receive less reflected light. The amount of reflected light is directly related to the accuracy of the measured distance. ToF camera was placed facing a white flat wall at different distances to the wall. At each position, 100 images were captured and their standard deviation was computed. The standard deviation for each pixel at different location is shown in Figure 3.4.



(a) 0.3 m away

(b) 0.9 m away

(c) 2.1 m away

(d) 3.5 m away

Figure 3.4: Standard deviation for 100 images while the ToF camera is stationary

The boundary pixels show high values of standard deviation especially the corner pixels, meaning that these pixels are very inaccurate. In the experiments performed, boundary pixels are removed.

3.1.3 Phase Wrap Filter

Phase wraps occur when the emitted light travels a distance longer that the maximum distance of the ToF camera which is controlled by the modulation frequency. Since the main application is for trajectory estimation, implemented methods only detect the phase-wraps but do not correct them. In cases where these points are needed for mapping or object reconstruction, methods such as one proposed in [19] can be applied. These methods have some shortcomings since they cannot determine the number of wraps. To be able to determine the number of wraps, the multi-frequency of the emitted light must be employed [20] and this require the use of two cameras to be applicable in real-time applications. The other reason why using unwrapped points for ego-motion estimation is erroneous is because distance accuracy is proportional to distance. These measurements might be useful for mapping but are likely to produce erroneous results for ego-motion estimation. The method we implemented make use of both range and amplitude images. A confidence map of the pixels is used as the product of the squared range ($\mathbf{d}$) image and amplitude ($\mathbf{A}$) image which should give approximately a constant value since distance and amplitude are inversely proportional. The confidence map $\boldsymbol{\eta}$ is computed as

$$\boldsymbol{\eta} = \mathbf{A} \cdot \mathbf{d}^2 \tag{3.2}$$

The confidence map is compared to the threshold confidence map ($\boldsymbol{\eta}_{thres}$) to determine if a pixel is wrapped. The threshold confidence map has values that are decreasing quadratically (expressed in (3.4)) with pixels from the center pixels to the boundary pixels to accommodate non-uniform illumination (see Figure 3.5).

$$\boldsymbol{\eta} \leq \boldsymbol{\eta}_{thres} \tag{3.3}$$

and $\boldsymbol{\eta}_{thres}$ is given by

$$\boldsymbol{\eta}_{thres} = \left(1 - \frac{(w_i - c_w)^2 + (h_i - c_h)^2}{13088}\right) T \tag{3.4}$$

where w_i, h_i are the pixels locations, c_w, c_h are the center pixel locations and T is the maximum value of the confidence map which is determined experimentally depending on the reflectivity of the environment and in this experiment, T is set to 5×10^4. The value 13088 is chosen such that the pixels at the center are compared to the full value of T and as it gets closer to the boundary pixels, a fraction of T is used. The value 13088 depends on the number of pixels of ToF camera images. Another motivation for this simple method is that the camera is used in an environment with a uniform reflectivity meaning it will not reject unwrapped points and it will be able to determine wraps even if the whole image is phase wrapped unlike in [19].

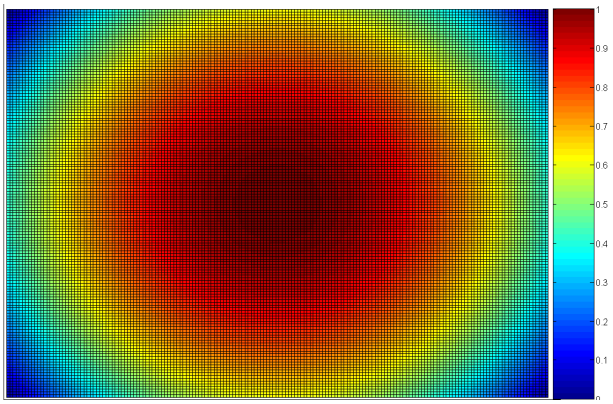
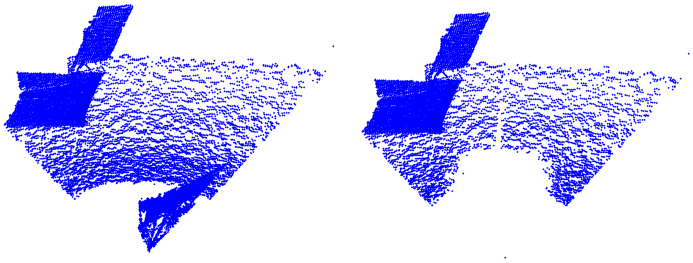
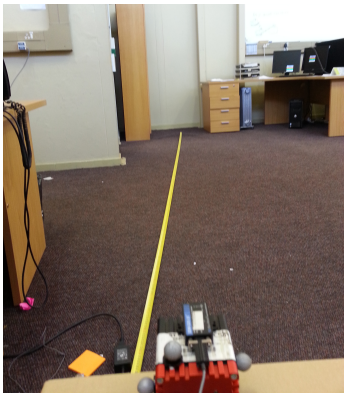


Figure 3.5: How the threshold of the confidence map varies across the pixels to accommodate inhomogeneous illumination

To test the proposed phase filter, data was collected in an environment shown in Figure 3.6(c). The ToF camera was placed such that the background wall is further than 5 m, in order to have point on the wall wrapped as shown in Figure 3.6(a). The point cloud after the filter has been applied as can be seen in Figure 3.6(b). The filter proposed produces good results but it seems to remove some points on the floor as well. This is as a result of the floor been carpeted and also having a different reflectivity compared to the wall which makes it return light with low amplitude.



(a) (b)



(c)

Figure 3.6: (a) shows the point cloud before the phase-wrap detector filter was applied, (b) shows the point cloud after the phase wrap points have been removed and (c) shows the environment where the images were collected to test the phase-wrap detection algorithm.

The camera has a modulation frequency of 30 MHz, which correspond to the maximum measurement distance of 5 m. This is an advantage since the calibration range of the camera is 0.8 m to 8 m.

3.2 ToF Camera Ego-Motion Estimation

In this section, the ToF camera ego-motion estimation algorithms that are implemented are described. A 6D motion model was used and it is presented in its simplified version in Figure 3.7. The camera pose at time t is represented by $\mathbf{T}_t$ and a transformation $\Delta\mathbf{T}_t^{t+1}$ transforms it to pose $\mathbf{T}_{t+1}$. This is defined mathematically as

$$\mathbf{T}_{t+1} = \mathbf{T}_t \Delta\mathbf{T}_t^{t+1} \tag{3.5}$$

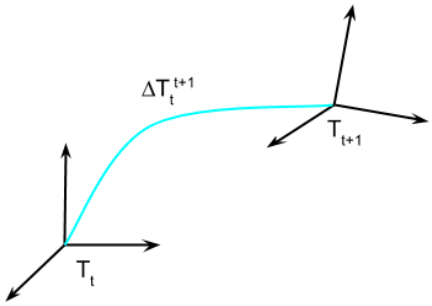


Figure 3.7: 6D motion model used for estimating trajectory

The pose is represented by rotation $\mathbf{R}$ and translation $\mathbf{t} = [t_x,\ t_y,\ t_z]^T$ in a matrix form as

$$\mathbf{T} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ 0 & 1 \end{bmatrix} \tag{3.6}$$

When computing the transformation, incremental angles are assumed to be small ($\sin\theta \approx \theta$ and $\cos\theta \approx 1$) and hence $\mathbf{R}$ can be simplified to

$$\mathbf{R} = \mathbf{R}_\alpha \mathbf{R}_\beta \mathbf{R}_\gamma = \begin{bmatrix} 1 & -\gamma & \beta \\ \gamma & 1 & -\alpha \\ -\beta & \alpha & 1 \end{bmatrix} \tag{3.7}$$

where α, β and γ are the rotations with respect to x, y and z axes respectively.(See Appendix B for derivation of $\mathbf{R}$)

3.2.1 Iterative Closest Point Methods

ICP algorithms are well known methods for registering two point clouds. Both point-to-point ICP [3] and point-to-plane ICP [4] are modified and implemented such that their performance can be compared for underground mining data. If point-to-plane ICP

performs better than point-to-point ICP, this will encourages methods like Minimally Uncertain Maximum Consensus [39] and 3D Normal Distributions Transforms [44] to be implemented in the same data. These ICP algorithms can be explained in five stages [23]. The algorithm tries to find the transformation between a base point cloud $\{\mathbf{B} = \mathbf{b}_i | i = 1, \cdots, n_b\}$ and scene point cloud $\{\mathbf{S} = \mathbf{s}_i | i = 1, \cdots, n_s\}$ by iterating the following stages:

1. **Selection of Point**

   In this stage source points are chosen from the base point cloud which will be used for the estimation of the transformation. ToF camera provides 25 344 points and using all these points will be time consuming and does not improve the accuracy. A subset of the points is chosen such that accurate results are still obtained with an improvement on the speed. Edge points can be used as source points similar to [26] by taking advantage of the fact that in the underground mines there are support pillars as shown in Figure 3.8. The extracted edge points

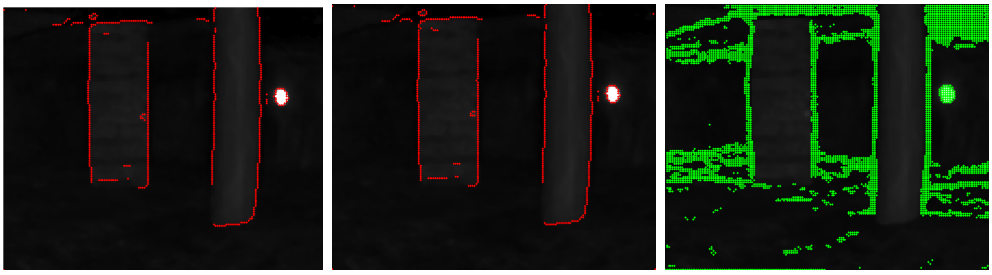
   

   Figure 3.8: Picture of pillars taken from the artificial underground mine

   using prewitt and sobel edge detection operator can be seen in Figure 3.9(a) and Figure 3.9(b) respectively. The results show that most of these edge points will be rejected by the jump edge filter as it can be seen in Figure 3.9(c). Using edge points is erroneous due to the nature of ToF measurement principle, hence it was eliminated. Instead, pseudo random sampling method which samples 500 points is used to select source points. By using sampled points, time consumption is reduced and the algorithm still performs similarly to when all the points are used.

2. **Matching**

   Once the source points have been selected in the base point cloud, the next step is finding the corresponding points in the scene point cloud. Since the exact point correspondence is unknown, it is assumed that the nearest point on the scene point cloud is the corresponding point to that base point. More than 50 % of ICP time is spent on finding corresponding points. To increase the ICP speed,



(a) (b) (c)

Figure 3.9: (a) shows edge detection results from prewitt operator and (b) shows a sobel operator results while (c) shows the results from the jump edge filter.

KD-trees are used for finding the closest point. A KD-tree is an algorithm for partitioning data structure which organises points in k-dimensional space. This has application to finding the nearest neighbours fast and efficiently. A mex function implementation for MATLAB® was used which is available online[1] and in [69].

3. **Weighting**
   This stage gives the quality of the corresponding points from the previous stage which is mostly based on the distance between pairs. A uniform weight of unity was assigned to all the corresponding point.

4. **Rejection**
   Rejection of corresponding points which are classified as outliers. Since the correspondence is found by finding the closest point, it is possible that one point on scene point cloud is assigned to more than one point in the data point cloud in situations similar to what is illustrated in Figure 3.10. In this situation, a multi-pairing rejection filter is applied to reject all the correspondence except the one with the smallest distance even though in Figure 3.10 it does not yield the correct correspondence and the error introduced will be minimal. Additionally, an adaptive distance threshold was used to reject pairs with a distance greater than distance d. Distance d is the sum of the mean distance and standard deviation of all the corresponding pairs.

5. **Error Metric**
   Error metric refers to either a point-to-point or a point-to-plane ICP error mea-

[1] http://www.mathworks.com/matlabcentral/fileexchange/21512-KD-tree-for-matlab

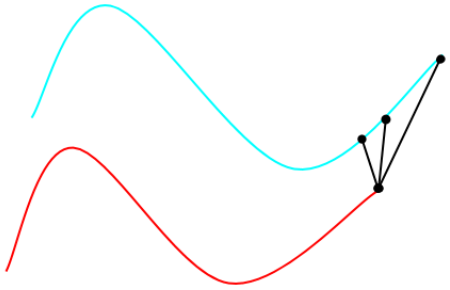


Figure 3.10: An example of multiple pairing that need to be rejected

sure. A point-to-point ICP algorithm finds the transformation by minimising the sum of the squared distances between the corresponding pairs. Figure 3.11(a) illustrates how a point-to-point finds the transformation and is formulated as

$$\xi_{pp}(\mathbf{R}, \mathbf{t}) = \sum_j \|\mathbf{R}\mathbf{s}_j + \mathbf{t} - \mathbf{b}_j\|^2 \tag{3.8}$$

Equation (3.8) has a close form solution which was developed by Arun *et al.* [36] that uses Singular Value Decomposition (SVD) to compute the transformation $(\mathbf{R}, \mathbf{t})$.

A point-to-plane scheme minimises the sum of the squared distance between points in base point cloud and the tangent planes of the corresponding points on the scene point cloud. Figure 3.11(b) is an illustration in this regard and it is mathematically represented as

$$\xi_{pl}(\mathbf{R}, \mathbf{t}) = \sum_j [(\mathbf{R}\mathbf{s}_j + \mathbf{t} - \mathbf{b}_j) \cdot \mathbf{n}_j]^2 \tag{3.9}$$

where $\mathbf{n}_j$ is the normal vector of the surface plane of the base point cloud. The surface normals are estimated using least squared methods as proposed by Mitra and Nguyen [37]. A point-to-plane scheme does not have closed form solution, hence (3.9) was linearised according to Low [70]. Refer to Appendix B for the computation of transformation for both point-to-point and point-to-plane ICP.

These stages are iterated until the transformation converges or maximum number of iterations is reached. Algorithm 1 shows a point-to-plane algorithm implemented. The difference between point-to-point and point-to-plane is that no planes computation is required for point-to-point ICP.

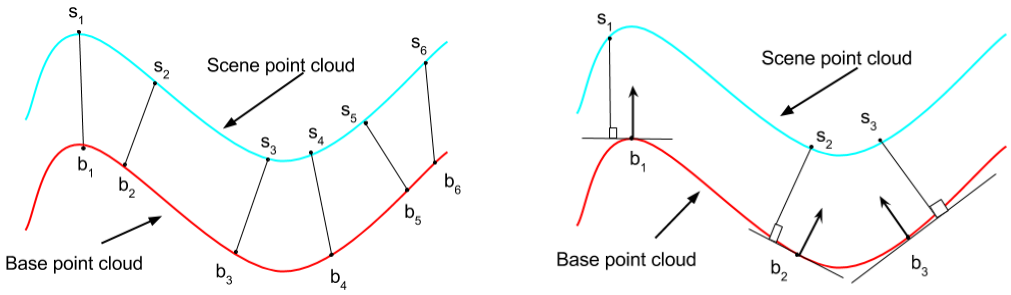


Figure 3.11: (a) Illustration of point-to-point ICP and (b) illustration of point-to-plane ICP.

3.2.2 Feature-Based Method

Since ToF cameras provide range and amplitude images, in this algorithm both images are utilised in order to estimate the camera's trajectory. Amplitude images are similar to gray-scale images, which means feature detection and tracking algorithms applicable to gray scale images are applicable in ToF amplitude images.

A summary of feature detector algorithms can be found in [71]. In this algorithm, the SIFT algorithm [5] was used due to its robustness to rotation, scale and illumination change. SIFT uses 128-dimensional keypoint to describe a feature and its secret lies in the convolution of the images with the difference of gaussian function [5]. There are implementations of these algorithms available on the Internet. David Lowe, who proposed the SIFT [5] algorithm also has a MATLAB® implementation online[1]. In this dissertation, implementation by Vedaldi and Fulkerson was used which is also available online [69] and it is referred to as vlfeatSIFT. Figure 3.12 shows the vlfeatSIFT tracking features in two consecutive images.

The SIFT algorithm is used to track features between two consecutive amplitude images. Then, 3D points of these features are used to estimate the transformation using a least square method developed by Arun *at el.* [36]. The 3D point of a feature is taken as the average of eight neighbouring points. SIFT algorithm contains some mismatches which are called outliers that are also visible in Figure 3.12. The outliers are rejected using the RANSAC algorithm [30]. At least three point correspondences are needed to compute the transformation. RANSAC randomly selects three points which

[1] http://www.cs.ubc.ca/ lowe/keypoints/

Algorithm 1 ICP point-to-plane algorithm

Get the base point cloud ($\mathbf{B}$)
Apply jump edge filter
Apply a phase-wrap filter
Initialise the first pose $\mathbf{T}_0$ to 4×4 identity matrix
for $i \leftarrow$ to $NumberOfImages$ **do**
    Get the scene point cloud ($\mathbf{S}$)
    Apply jump edge filter
    Apply a phase-wrap filter
    Initialise the transformation ($\Delta\mathbf{T}$) to 4×4 identity matrix
    repeat
        Randomly select source points in the scene point cloud
        For every source point, find the corresponding point in the base point cloud (KD-tree)
        Multiple pairing rejection for source points with more than one pair
        Adaptive mean Outlier filter
        Compute the transformation ($\Delta\bar{\mathbf{T}}$)
        Update $\Delta\mathbf{T}$ with the computed transformation ($\Delta\bar{\mathbf{T}}$)
        Apply the transformation ($\Delta\mathbf{T}$) on the scene point cloud
    until It converges or reach maximum number of iterations
    Compute the current pose using equation (3.5)
    Equate the base point cloud to scene point cloud for the next iteration
end for

are used to compute a transformation. If more than 80% of the points are inliers, then all these points are used to compute the final transformation; otherwise, another three pair of correspondence is randomly selected. The SIFT based algorithm is summarised in Algorithm 2.

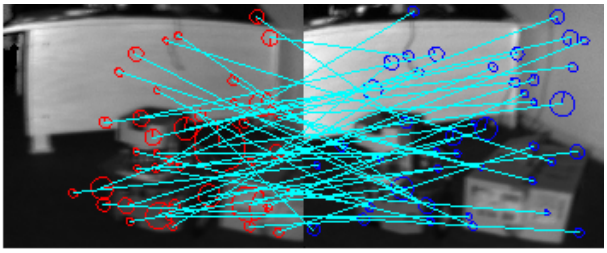


Figure 3.12: SIFT algorithms tracking point features between two consecutive amplitude images

3.3 INS Estimation

Inertial Navigation System (INS) is a system capable of tracking a vehicle's position and orientation using measurements from accelerometers and gyroscopes given the initial position, velocity and orientation relative to the global frame of reference. Figure 3.13 illustrates the navigation frames used in this section where a body frame is attached to the vehicle and global frame is fixed with the earth frame.

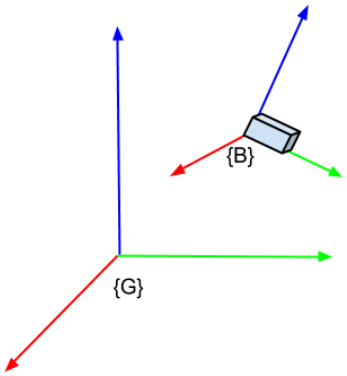


Figure 3.13: Navigation frames where {G} is the global frame and {B} is the body frame

A strap-down system using IMU is used because of low cost, small size and no moving components as opposed to stable platform systems. A strap-down system refers to a systems where both the accelerometer and gyroscope are rigidly mounted to the vehicle. A 3DM-GX3®-25 IMU [72] from MicroStrain® which contains three orthogonal rate-gyroscope and three orthogonal accelerometers that measures angular velocity and linear acceleration, respectively, is used.

Algorithm 2 SIFT-based ego-motion

Get the base point cloud ($\mathbf{B}$) and base amplitude image ($\mathbf{A}_b$)
Apply jump edge filter on the point cloud $\mathbf{B}$
Apply phase-wrap filter on the point cloud $\mathbf{B}$
Find SIFT features in the base amplitude image $\mathbf{A}_b$
Initialise the first pose $\mathbf{T}_0$ to 4×4 identity matrix
for $i \leftarrow$ to $NumberOfImages$ **do**
    Get the scene point cloud ($\mathbf{S}$) and base amplitude image ($\mathbf{A}_s$)
    Apply jump edge filter on the point cloud $\mathbf{S}$
    Apply phase-wrap filter on the point cloud $\mathbf{S}$
    Find SIFT features in the scene amplitude image $\mathbf{A}_s$
    Match the features in $\mathbf{A}_b$ and $\mathbf{A}_s$
    while $Inliers \leq 0.8$ (RANSAC) **do**
        Randomly select three matches
        Estimate the transformation $\Delta\bar{\mathbf{T}}$
        Compute the number of *inliers*
    end while
    Compute the final transformation $\Delta\mathbf{T}$ using all the inliers
    Compute the current pose using equation (3.5)
    Equate the base point cloud and amplitude images to scene point cloud and amplitude images respectively for the next iteration
end for

Orientation represents the transformation from the global frame to the body frame and is tracked by integrating angular velocities ($\boldsymbol{\omega} = [\omega_x,\ \omega_y,\ \omega_z]^T$) from the rate-gyroscope. The orientation in INS derivation is represented by a quaternion to simplify the derivation of the EKF in Section 3.4. A derivation using either Euler angles or directional cosine matrix can be found in [73]. A quaternion represents orientation using a four parameter representation.

$$\bar{q} = \begin{bmatrix} q_1 \\ q_2 \\ q_3 \\ q_4 \end{bmatrix} = \begin{bmatrix} k_x \sin(\theta/2) \\ k_y \sin(\theta/2) \\ k_z \sin(\theta/2) \\ \cos(\theta/2) \end{bmatrix} = \begin{bmatrix} \mathbf{q} \\ q_4 \end{bmatrix} \tag{3.10}$$

which is generally defined as

$$\bar{q} = q_4 + q_3 i + q_2 j + q_3 k \quad \text{where} \quad i^2 = j^2 = k^2 = -1 \tag{3.11}$$

Appendix C gives a complete definition with some useful identities and properties of quaternions. Notably, the definition used here is not Hamiltonian. The propagation of orientation as new angular velocities are received is computed assuming that the angular velocity $\boldsymbol{\omega}$ is constant over the integration period δt

$$\bar{q}_{k+1} = \mathbf{A}(\boldsymbol{\omega}, \delta t) \otimes \bar{q}_k \tag{3.12}$$

where

$$\mathbf{A}(\boldsymbol{\omega}, \delta t) = \begin{bmatrix} \frac{\boldsymbol{\omega}}{|\boldsymbol{\omega}|} \sin(\frac{|\boldsymbol{\omega}|}{2}\delta t) \\ \cos(\frac{|\boldsymbol{\omega}|}{2}\delta t) \end{bmatrix} \tag{3.13}$$

To track the position and velocity, acceleration measurements are firstly projected into the global frame using the orientation such that the gravity can be subtracted. Because accelerometers measure inertial force, it always measure gravitational force which needs to be removed from the measurements so that they can be useful. Secondly, gravity corrected measurements are integrated once to obtain velocity and second time to obtain position. The integration is performed using forth order Runge Kutta method for more accurate results.

$$y_{k+1} = y_k + \frac{1}{6}\left(k_1 + 2k_2 + 2k_3 + _4\right) \tag{3.14}$$

where

$$k_1 = \delta t f(t_k, y_k) \tag{3.15}$$

$$k_2 = \delta t f(t_k + \frac{\delta t}{2}, y_k + \frac{k_1}{2}) \tag{3.16}$$

$$k_3 = \delta t f(t_k + \frac{\delta t}{2}, y_k + \frac{k_2}{2}) \tag{3.17}$$

$$k_4 = \delta t f(t_k + \delta t, y_k + k_3) \tag{3.18}$$

The algorithm is illustrated graphically in Figure 3.14.
INS algorithm suffers from cumulative error mostly caused by biases and noise in both accelerometers and rate-gyroscope. To show how fast the error propagates, a data set with 100 samples was recorded for stationary IMU and the INS algorithm was run for 30 seconds for all the data set (see to Figure 3.15). Since it is stationary, any change in position or orientation is regarded as an error. Before data was collected, IMU was calibrated using a software provided by the manufacturers to estimate gyro bias.

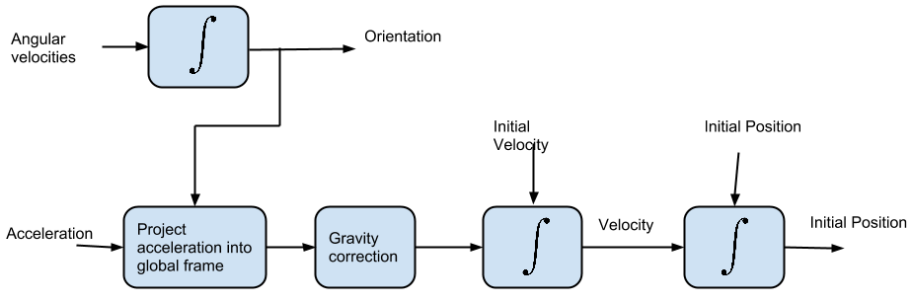


Figure 3.14: INS algorithm

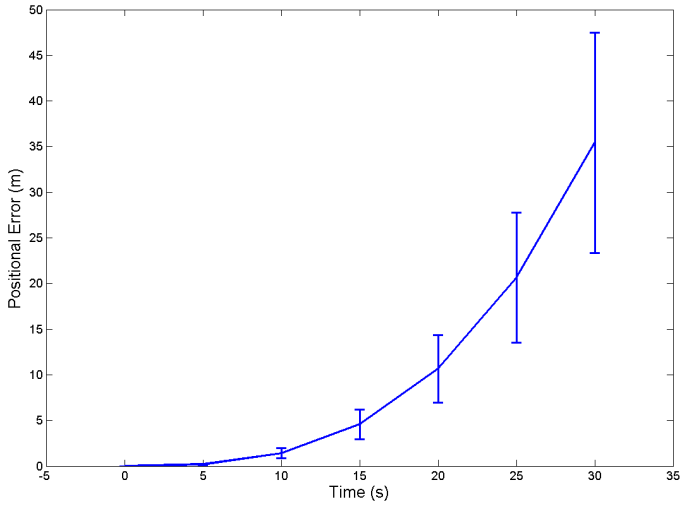


Figure 3.15: Error propagation of INS for 30 seconds while stationary. The bars represents the standard deviation

Another cause of fast error propagation is the error in the estimated gravity vector. Gravity magnitude varies depending on the location. In this experiments it was computed using an equation that depends on latitude L, height above sea level h and earth radius R_0 [73].

$$g(0) = 9.780318(1 + 5.3025 \times 10^{-3} \sin^2 L - 5.9 \times 10^{-6} \sin^2 2L) \tag{3.19}$$

$$g(h) = \frac{g(0)}{1 + {}^{h}/_{R_0}} \tag{3.20}$$

where $g(0)$ is the gravity at latitude L at sea level.

3.4 Fused ToF-IMU Ego-Motion Estimation

The fusion of ToF camera's ego-motion and IMU data is performed using Kalman filter-based methods because it is able to handle K dimensional state elements and the fact that the dynamic and measurement processes can be modelled as gaussian processes. Before fusion is performed, it is necessary to find the transformation between the sensors being fused. An EKF-based algorithm which is adopted from [60] is proposed and tested on simulated dataset. The transformation between sensors being fused is incorporated into the kalman filter algorithm which reduces the drift in the trajectory estimation. It is assumed that the mobile robot is moving on a 2D planar surface, where trajectory consist of three elements(x, y and θ). The calibration algorithm will be explained next followed by 2D planar fusion algorithm.

3.4.1 Calibration of ToF-IMU System

Before fusion can be performed, the transformation between ToF and IMU needs to be estimated. Inaccuracy in the transformation causes errors in the estimated trajectory. The process of finding transformation is termed calibration and is performed using Extended Kalman Filter (EKF). This approach is adopted from [74] and [60] where a normal camera and an IMU were calibrated using a checkerboard.

In [60], checkerboard was used to provide target features that are aligned with gravity and were extracted from the camera images to correct IMU estimates and at the same time estimate the transformation between camera and IMU. Since ToF camera has low resolution, using a checkerboard to provide features is a challenge. Instead, a board with reflectance markers that are 12 mm in diameter was used as shown in Figure 3.16. All the pixels that cover the reflectance markers are saturated. These pixels are clustered into groups and a circle is fitted covering all the pixels and the centre is used as the target point. Since these pixels are saturated, distance is found by averaging eight neighbouring pixels just outside the circle.

Figure 3.17 shows the assembly of the ToF-IMU system used for the experiment. For calibration, refer to Figure 3.18 that illustrates the setup consisting of reflectance board and ToF-IMU system.

Figure 3.16: Calibration board that consist of reflective markers

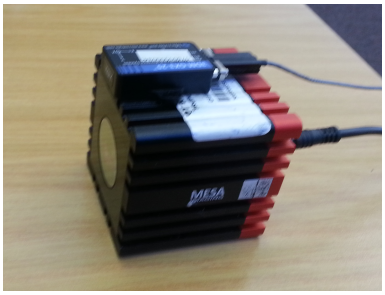


Figure 3.17: ToF-IMU system

EKF consists of two stages which are the time update stage and the measurement update stage. In the time update stage, IMU measurements are integrated to estimate IMU position, velocity and orientation, while in the measurement update stage, reflectance positions are extracted from ToF images and are used as measurements to estimate transformation between ToF and IMU and at the same time correct the IMU estimates.

Time Measure Update

The IMU measures angular velocity and the linear accelerations which are modelled as

$$\boldsymbol{\omega}_m(t) = \boldsymbol{\omega}(t) + \mathbf{b}_g(t) + \mathbf{n}_{rg}(t) \tag{3.21}$$

$$\mathbf{a}_m(t) = \mathbf{R}_G^I({}^G\mathbf{a}(t) - {}^G\mathbf{g}) + \mathbf{b}_a(t) + \mathbf{n}_{ra}(t) \tag{3.22}$$

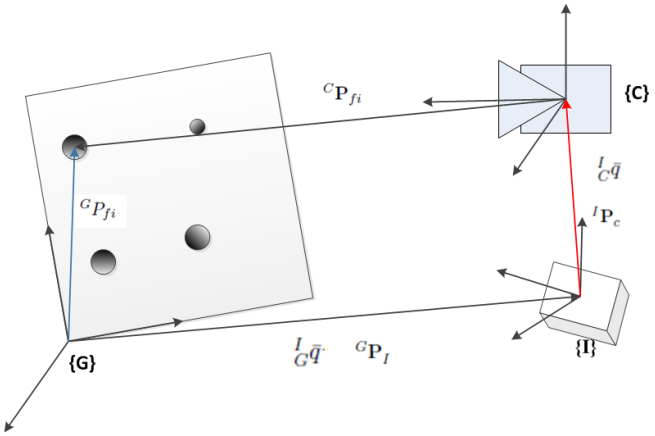


Figure 3.18: Illustration of the relation between known features f_i, ToF camera frame {C}, IMU frame {I} and global frame {G}. The ToF-IMU 6 DoF transformation is denoted by $({}^{I}\mathbf{P}_{C}, {}^{I}_{C}\bar{q})$ and the position of the known features with respect to {C} is ${}^{C}\mathbf{P}_{f_i}$.

where $\boldsymbol{\omega}_m$ and $\mathbf{a}_m$ are the measured angular velocity and linear acceleration, while $\boldsymbol{\omega}$ and $\mathbf{a}$ are the actual angular velocity and linear acceleration respectively. The variables $\mathbf{b}_g$ and $\mathbf{b}_a$ are the gyroscopic bias and accelerometer bias. The noise is represented by $\mathbf{n}_{rg}$ and $\mathbf{n}_{ra}$ which are assumed to be zero-mean gaussian white noise. $\mathbf{R}^{I}_{G}$ is a 3×3 rotational matrix that transforms a vector from a global frame to an IMU frame. The state estimates vector $\mathbf{x}$ used is similar to one used in [60] and is defined as

$$\mathbf{x} = \begin{bmatrix} {}^{I}_{G}\bar{q}^{T} & \mathbf{b}_g^{T} & {}^{G}\mathbf{V}_{I}^{T} & \mathbf{b}_a^{T} & {}^{G}\mathbf{P}_{I}^{T} & {}^{I}_{C}\bar{q}^{T} & {}^{I}\mathbf{P}_{C}^{T} \end{bmatrix}^{T} \tag{3.23}$$

where ${}^{G}\mathbf{V}_I$ and ${}^{G}\mathbf{P}_I$ are the velocity and position of IMU in the global frame reference respectively and $\begin{bmatrix} {}^{I}_{C}\bar{q}^{T}, & {}^{I}\mathbf{P}_{C}^{T} \end{bmatrix}^{T}$ is the transformation between IMU and ToF camera.

The state estimates propagation equations with the expected operator($\hat{}$) are

$$\dot{\hat{{}^I_G\bar{q}}}(t) = \frac{1}{2}\mathbf{\Omega}(\hat{\boldsymbol{\omega}}(t)){}^L_G\hat{\bar{q}}(t) \tag{3.24}$$

$$\dot{\hat{\mathbf{b}}}_g(t) = \mathbf{0}_{3\times 1} \tag{3.25}$$

$$\dot{{}^G\hat{\mathbf{V}}}_I = \mathbf{C}({}^G_I\hat{\bar{q}}(t))\ \hat{\mathbf{a}}(t) + {}^G\mathbf{g} \tag{3.26}$$

$$\dot{\hat{\mathbf{b}}}(t) = \mathbf{0}_{3\times 3} \tag{3.27}$$

$$\dot{{}^G\hat{\mathbf{P}}}_I(t) = {}^G\hat{\mathbf{V}}_I(t) \tag{3.28}$$

$$\dot{{}^I_C\hat{\bar{q}}}(t) = \mathbf{0}_{3\times 3} \tag{3.29}$$

$$\dot{{}^I_C\hat{\mathbf{P}}}(t) = \mathbf{0}_{3\times 3} \tag{3.30}$$

where $\mathbf{C}(\bar{q})$ is a 3×3 rotational matrix representation of a quaternion orientation $\bar{q}$ and $\dot{()}$ represents the derivative. The error state vector $\tilde{\mathbf{x}}$ used to correct the state estimates is defined as

$$\tilde{\mathbf{x}} = \mathbf{x} - \hat{\mathbf{x}} \tag{3.31}$$

where $\mathbf{x}$ is the actual state and $\hat{\mathbf{x}}$ is the estimated state. For quaternion error, a different error model is defined as $\delta\theta$

$$\delta\bar{q} = \bar{q} \otimes \hat{\bar{q}} = \begin{bmatrix} \hat{\mathbf{k}}\sin(\delta\theta/2) \\ \cos(\delta\theta/2) \end{bmatrix} \approx \begin{bmatrix} \frac{1}{2}\delta\boldsymbol{\theta} \\ 1 \end{bmatrix} \tag{3.32}$$

where $\bar{q}$ is the actual quaternion orientation and $\hat{\bar{q}}$ is the estimated quaternion orientation. A general EKF continuous time error state equation is defined as

$$\dot{\tilde{\mathbf{x}}} = \mathbf{F}_c\tilde{\mathbf{x}} + \mathbf{G}_c\mathbf{n} \tag{3.33}$$

where $\mathbf{F}_c$ and $\mathbf{G}_c$ are found by differentiating (3.31) and (3.32) and substituting (3.28) and (3.24) and their real values $({}^I_G\bar{q}, {}^G\mathbf{p}_I)$:

$$\mathbf{F}_c = \begin{bmatrix} -\lfloor\hat{\boldsymbol{\omega}}\times\rfloor & -\mathbf{I}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times9} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times9} \\ -\mathbf{C}^T({}^I_G\hat{\bar{q}})\lfloor\hat{\mathbf{a}}\times\rfloor & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & -\mathbf{C}^T({}^I_G\hat{\bar{q}}) & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times9} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{I}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times9} \\ \mathbf{0}_{6\times3} & \mathbf{0}_{6\times3} & \mathbf{0}_{6\times3} & \mathbf{0}_{6\times3} & \mathbf{0}_{6\times9} \end{bmatrix} \tag{3.34}$$

$$\mathbf{G}_c = \begin{bmatrix} -\mathbf{I}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{I}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & -\mathbf{C}^T({}^I_G\hat{\bar{q}}) & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{I}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{6\times3} & \mathbf{0}_{6\times3} & \mathbf{0}_{6\times3} & \mathbf{0}_{6\times3} \end{bmatrix} \tag{3.35}$$

$$\mathbf{n} = \begin{bmatrix} \mathbf{n}_{rg} \\ \mathbf{n}_{wg} \\ \mathbf{n}_{rq} \\ \mathbf{n}_{wa} \end{bmatrix} \tag{3.36}$$

where $\mathbf{n}_{wg}$ and $\mathbf{n}_{wa}$ are the random walks for gyroscopic and accelerometer biases. $\lfloor\mathbf{q}\times\rfloor$ is a skew matrix and defined as

$$\lfloor\mathbf{q}\times\rfloor = \begin{bmatrix} 0 & -q_3 & q_2 \\ q_3 & 0 & -q_1 \\ -q_2 & q_1 & 0 \end{bmatrix} \tag{3.37}$$

IMU measurements are sampled at 100 Hz and the INS algorithm described in section 3.3 above is applied to estimate the state vector. At the same time, covariance matrix $\mathbf{P}$ of the state estimates is updated as

$$\mathbf{P}_{k+1/k} = \boldsymbol{\Phi}_k \mathbf{P}_{k/k} \boldsymbol{\Phi}_k^T + \mathbf{Q}_d \tag{3.38}$$

where $\boldsymbol{\Phi}$ is the error state transition matrix and $\mathbf{Q}_d$ is the system noise covariance

matrix

$$\mathbf{\Phi}(t+\Delta t, t) = \exp(\mathbf{F}_c \Delta t) \tag{3.39}$$

$$\mathbf{Q}_d = \int_{t_k}^{t_k+T} \mathbf{\Phi}(t_{k+1}, \tau)\mathbf{G}_c\mathbf{Q}_c\mathbf{G}_C^T\mathbf{\Phi}^T(t_{t+1}, \tau)dt \tag{3.40}$$

$\mathbf{Q}_c$ is the covariance matrix of the IMU measurements that depends on the noise characteristics of accelerometer and gyroscope.

Measurement Update State

As the ToF-IMU system continuously moves while it capture ToF camera images and IMU data, 3D positions of the reflectance features are extracted from ToF images to update the state and covariance estimates. The measurements are modelled as

$$\mathbf{z}_i = \begin{bmatrix} x_i \\ y_i \\ z_i \end{bmatrix} + \boldsymbol{\eta}_i \tag{3.41}$$

where $\boldsymbol{\eta}_i$ is the zero-mean gaussian noise describing the accuracy of ToF distance measurements. Measurement error vector $\tilde{\mathbf{z}}$ is defined as

$$\tilde{\mathbf{z}} = \mathbf{z} - \hat{\mathbf{z}} \tag{3.42}$$

where $\mathbf{z}$ is the actual measurement from ToF camera and $\hat{\mathbf{z}}$ is the estimated measurement using state estimates as

$$\mathbf{z} = h(\mathbf{x}) = \mathbf{C}^T({}^I_c\bar{q})\mathbf{C}({}^I_G\bar{q})({}^G\mathbf{P}_{fi} - {}^G\mathbf{P}_I) - \mathbf{C}^T({}^I_c\bar{q}){}^I\mathbf{P}_C \tag{3.43}$$

substituting $\mathbf{z}$ and $\hat{\mathbf{z}}$ described by (3.44) in (3.42), the measurement error $\tilde{\mathbf{z}}$ can be expressed as

$$\begin{aligned}\tilde{\mathbf{z}} = {} & (\mathbf{C}^T({}^I_C\hat{\bar{q}})\lfloor\mathbf{C}({}^I_G\hat{\bar{q}})^G\mathbf{P}_{fi}\times\rfloor - \mathbf{C}^T({}^I_C\hat{\bar{q}})\lfloor\mathbf{C}({}^I_G\hat{\bar{q}})^G\mathbf{P}_I\times\rfloor)\delta^I_G\boldsymbol{\theta} + \\ & (-\mathbf{C}^T({}^I_C\hat{\bar{q}})\lfloor\mathbf{C}({}^I_G\hat{\bar{q}})^G\mathbf{P}_{fi}\times\rfloor + \mathbf{C}^T({}^I_C\hat{\bar{q}})\lfloor\mathbf{C}({}^I_G\hat{\bar{q}})^G\mathbf{P}_I\times\rfloor + \\ & \mathbf{C}^T({}^I_C\hat{\bar{q}})\lfloor{}^I\mathbf{P}_I\times\rfloor)\delta^I_c\boldsymbol{\theta} - (\mathbf{C}^T({}^I_C\hat{\bar{q}})\mathbf{C}({}^I_G\hat{\bar{q}}))^G\tilde{\mathbf{P}}_I - (\mathbf{C}^T({}^I_C\hat{\bar{q}}))^I\tilde{\mathbf{P}}_c\end{aligned} \tag{3.44}$$

The Jacobian matrix for the measurement error state is given by

$$\mathbf{H}_i = \begin{bmatrix} \mathbf{J}^i_{\theta_{IG}} & \mathbf{0}_{3\times 9} & \mathbf{J}^i_{P_{GI}} & \mathbf{J}^i_{\theta_{IC}} & \mathbf{J}^i_{P_{IC}} \end{bmatrix} \tag{3.45}$$

where

$$\mathbf{J}^i_{\theta_{IG}} = \mathbf{C}^T({}^I_C\hat{\bar{q}})\lfloor \mathbf{C}({}^I_G\hat{\bar{q}})({}^G\mathbf{P}_{fi} - {}^G\mathbf{P}_I)\times \rfloor \tag{3.46}$$

$$\mathbf{J}^i_{\mathbf{P}_{GI}} = \mathbf{C}^T({}^I_C\hat{\bar{q}})(\lfloor \mathbf{C}({}^I_G\hat{\bar{q}}))^G\tilde{\mathbf{P}}_I \rfloor - \lfloor \mathbf{C}({}^I_G\hat{\bar{q}})^G\mathbf{P}_{fi}\times \rfloor + \lfloor {}^I\mathbf{P}_C \times \rfloor) \tag{3.47}$$

$$\mathbf{J}^i_{\theta_{IC}} = -\mathbf{C}^T({}^I_C\hat{\bar{q}})\mathbf{C}({}^I_G\hat{\bar{q}}) \tag{3.48}$$

$$\mathbf{J}^i_{\mathbf{P}_{IC}} = -\mathbf{C}^T({}^I_C\hat{\bar{q}}) \tag{3.49}$$

When N features are available, the measurement vector $\mathbf{z}$ is stacked as $\mathbf{z} = \begin{bmatrix} \mathbf{z}_1^T & , \cdots , & \mathbf{z}_N^T \end{bmatrix}^T$ and the error measurement jacobian matrix as $\mathbf{H} = \begin{bmatrix} \mathbf{H}_1^T & , \cdots , & \mathbf{H}_N^T \end{bmatrix}^T$.

Assuming that the propagated state estimate $\mathbf{x}_{k/k+1}$, propagated covariance matrix estimate $\mathbf{P}_{k/k+1}$ and the current measurement $\mathbf{z}$ are available, then the updated state estimates $\mathbf{x}_{k+1/k+1}$ is computed using Algorithm 3.

Algorithm 3 Update of the state estimates

1: Set $\mathbf{x}^0 = \mathbf{0}_{21\times 1}$
2: **repeat**
3: Compute estimated measurement $\hat{\mathbf{z}}$ using (3.43)
4: Compute residual r according to

$$\mathbf{r} = \mathbf{z} - \hat{\mathbf{z}}$$

5: Compute the Jacobian matrix $\mathbf{H}$ using (3.45)
6: Compute the covariance of the residual $\mathbf{S}$ as

$$\mathbf{S} = \mathbf{H}\mathbf{P}\mathbf{H}^T + \mathbf{R}$$

7: Compute the Kalman gain

$$\mathbf{K} = \mathbf{P}\mathbf{H}^T\mathbf{S}^{-1}$$

8: Compute the Error estimate vector

$$\tilde{\mathbf{x}}^j = \mathbf{K}(\mathbf{r} + \mathbf{H}\mathbf{x}^{\mathbf{j-1}})$$

9: Update quaternion orientation in the state vector as

$$\hat{\bar{q}}_{k+1/k+1} = \delta\hat{\bar{q}} \otimes \hat{\bar{q}}_{k+1/k}$$

where

$$\delta\hat{\bar{q}} = \begin{bmatrix} \delta\mathbf{q} \\ \sqrt{1 - \delta\mathbf{q}^T\delta\mathbf{q}} \end{bmatrix}$$

or if $\delta\mathbf{q}^T\delta\mathbf{q} > 1$

$$\delta\hat{\bar{q}} = \frac{1}{\sqrt{1 + \delta\mathbf{q}^T\delta\mathbf{q}}} \begin{bmatrix} \delta\mathbf{q} \\ 1 \end{bmatrix}$$

and

$$\delta\mathbf{q} = \frac{1}{2}\delta\boldsymbol{\theta}$$

10: Update the remaining elements of the state vector as

$$\mathbf{x}_{k+1/k} = \hat{\mathbf{x}}_{k+1/k} + \tilde{\mathbf{x}}$$

11: **until** $\mathbf{x}$ converges
12: Compute the updated covariance matrix

$$\mathbf{P}_{k+1/k+1} = (\mathbf{I} - \mathbf{K}\mathbf{H})\mathbf{P}_{k+1/k}(\mathbf{I} - \mathbf{K}\mathbf{H})^T + \mathbf{K}\mathbf{R}\mathbf{K}^T$$

3.4.2 2D Planar ToF-IMU Fusion

If the mobile robot is operating in a 2D planar surface, the problem is reduced from full 6D trajectory to 3D trajectory motion. See Figure 3.19(a) for an illustration of a 3D motion model and Figure 3.19(b) for the illustration of the transformation between the ToF camera frame and IMU frame in 3D motion. This transformation needs to be known to estimate more accurate results. Fusion is performed using the Kalman filter

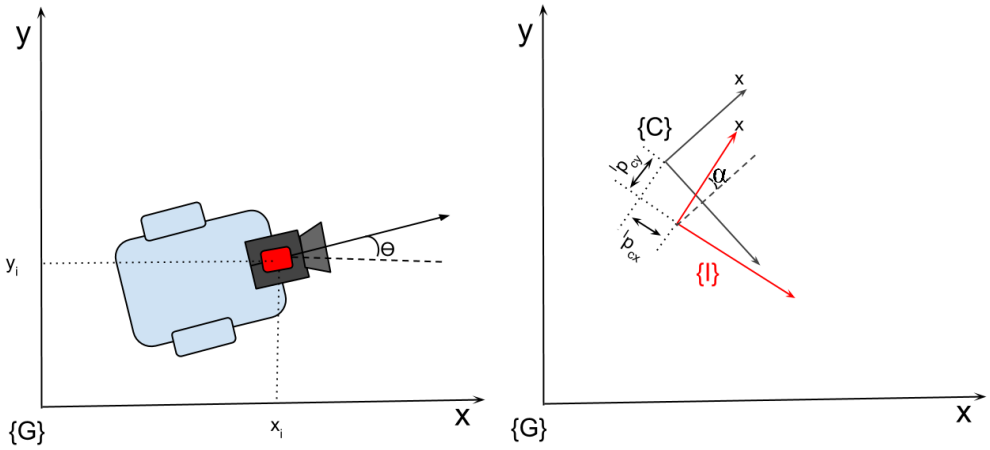


Figure 3.19: (a) shows the simplification on the 2D planar problem where (x_i, y_i) is the position and θ is the orientation. (b) shows the co-ordinates frames where $\{\mathbf{G}\}$ is the global frame, $\{\mathbf{I}\}$ is the IMU frame and $\{\mathbf{C}\}$ is the camera frame. ${}^{I}\mathbf{p}_{C}$ and α represents the transformation between IMU and the ToF camera

algorithm. A ten element state vector $\mathbf{x}$ used is

$$\mathbf{x} = \begin{bmatrix} \mathbf{v}_I & \mathbf{b}_a & \omega & b_g & {}^{G}\mathbf{p}_I & \theta & a_\theta \end{bmatrix}^T \tag{3.50}$$

Note that $\mathbf{v}_I$ is the velocity of the IMU with respect to IMU frame, otherwise if it is with respect to global frame, the measurement model becomes non-linear which requires EKF. EKF is sub-optimal since linearisation errors can not be removed. Similarly to EKF, the Kalman filter is subdivided into two stages, namely the time update stage and the measurement update stage.

The Time Update Stage

The same gyroscopic and accelerometer model in (3.21) and (3.22) respectively are used. IMU measurements are sampled at 100 Hz. To track the orientation θ, angular

velocity is integrated once while acceleration is integrated once to track velocity and the second time to track the position. Since this is a simple system and measurements are updated at a fast rate, a rectangular integration is used as follows

$$\theta_{t+1} = \theta_t + \omega_t \delta t \tag{3.51}$$

$$\mathbf{v}_{t+1} = \mathbf{v}_t + \mathbf{a}_t \delta t \tag{3.52}$$

$$\mathbf{p}_{t+1} = \mathbf{p}_t + \mathbf{v}_t \delta t \tag{3.53}$$

The full state propagation equations with the expected operator $\hat{()}$ are

$$\dot{\hat{v}}_I = \hat{\mathbf{a}} = \mathbf{a}_m - \hat{\mathbf{b}}_a \tag{3.54}$$

$$\dot{\hat{b}}_a = \mathbf{0}_{2\times 1} \tag{3.55}$$

$$\dot{\hat{\omega}} = \hat{a}_\theta = \frac{\hat{\omega}_k - \hat{\omega}_{k-1}}{\delta t} \tag{3.56}$$

$$\dot{\hat{b}}_g = 0 \tag{3.57}$$

$${}^G\hat{\mathbf{p}}_I = \mathbf{R}_{\hat{\theta}} \hat{\mathbf{v}}_\mathbf{I} \tag{3.58}$$

$$\dot{\hat{\theta}} = \hat{\omega} = \omega_m - \hat{b}_g \tag{3.59}$$

$$\dot{\hat{a}}_\theta = \dot{\hat{a}}_\theta \tag{3.60}$$

where $\mathbf{R}_\theta$ is a 2×2 rotational matrix

$$\mathbf{R}_\theta = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \tag{3.61}$$

The error state vector is defined in (3.31) and the general continuous time error state is defined in (3.33) where $\mathbf{F}_c$, $\mathbf{G}_c$ and $\mathbf{n}$ are defined as

$$\mathbf{F}_c = \begin{bmatrix} \mathbf{0}_{2\times 2} & -\mathbf{I}_{2\times 2} & \mathbf{0}_{2\times 1} & \mathbf{0}_{2\times 1} & \mathbf{0}_{2\times 2} & \hat{\mathbf{R}}_\theta \mathbf{\Omega} \hat{\mathbf{a}} & \mathbf{0}_{2\times 1} \\ \mathbf{0}_{2\times 2} & \mathbf{0}_{2\times 2} & \mathbf{0}_{2\times 1} & \mathbf{0}_{2\times 1} & \mathbf{0}_{2\times 2} & \mathbf{0}_{2\times 1} & \mathbf{0}_{2\times 1} \\ \mathbf{0}_{1\times 2} & \mathbf{0}_{1\times 2} & 0 & 0 & \mathbf{0}_{1\times 2} & 0 & 1 \\ \mathbf{0}_{2\times 2} & \mathbf{0}_{2\times 2} & \mathbf{0}_{2\times 1} & \mathbf{0}_{2\times 1} & \mathbf{0}_{2\times 2} & \mathbf{0}_{2\times 1} & \mathbf{0}_{2\times 1} \\ \hat{\mathbf{R}}_\theta & \mathbf{0}_{2\times 2} & \mathbf{0}_{1\times 2} & \mathbf{0}_{1\times 2} & \mathbf{0}_{2\times 2} & \hat{\mathbf{R}}_\theta \mathbf{\Omega} \hat{\mathbf{v}}_I & \mathbf{0}_{2\times 1} \\ \mathbf{0}_{1\times 2} & \mathbf{0}_{1\times 2} & 1 & -1 & \mathbf{0}_{1\times 2} & 0 & 0 \\ \mathbf{0}_{2\times 2} & \mathbf{0}_{2\times 2} & \mathbf{0}_{2\times 1} & \mathbf{0}_{2\times 1} & \mathbf{0}_{2\times 2} & \mathbf{0}_{2\times 1} & \mathbf{0}_{2\times 1} \end{bmatrix} \tag{3.62}$$

where

$$\mathbf{\Omega} = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \tag{3.63}$$

$$\mathbf{G}_c = \begin{bmatrix} -\mathbf{I}_{2\times2} & \mathbf{0}_{2\times2} & \mathbf{0}_{2\times1} & \mathbf{0}_{2\times1} \\ \mathbf{0}_{2\times2} & \mathbf{I}_{2\times2} & \mathbf{0}_{2\times1} & \mathbf{0}_{2\times1} \\ \mathbf{0}_{1\times2} & \mathbf{0}_{1\times2} & 0 & 0 \\ \mathbf{0}_{1\times2} & \mathbf{0}_{1\times2} & 0 & 1 \\ \mathbf{0}_{2\times2} & \mathbf{0}_{2\times2} & \mathbf{0}_{2\times1} & \mathbf{0}_{2\times1} \\ \mathbf{0}_{1\times2} & \mathbf{0}_{1\times2} & -1 & 0 \\ \mathbf{0}_{1\times2} & \mathbf{0}_{1\times2} & 0 & 0 \end{bmatrix} \tag{3.64}$$

$$\mathbf{n} = \begin{bmatrix} \mathbf{n}_{ra} \\ \mathbf{n}_{wa} \\ n_{rg} \\ n_{wg} \end{bmatrix} \tag{3.65}$$

The covariance matrix $\mathbf{P}$ is computed using (3.38) and (3.39).

The Measurement Update Stage

The ToF ego-motion algorithms described in Section 3.2 produce a 3×1 translation vector $\mathbf{t}$ and 3×3 rotational matrix $\mathbf{R}$. In this case since the robot is moving in a plane, $[t_x, t_y]^T$ will be used for translation; $\mathbf{R}$ is converted to Euler angles. The rotation around z axis is used as the change in orientation $\Delta\theta$. The time difference (Δt) between each image is known, assuming a constant velocity, the linear and angular velocity can be computed as

$$\mathbf{v} = \begin{bmatrix} v_x \\ v_y \end{bmatrix} = \begin{bmatrix} t_x/\Delta t \\ t_y/\Delta t \end{bmatrix} \tag{3.66}$$

$$\omega = {}^{\Delta\omega}/_{\Delta t} \tag{3.67}$$

These velocities are used as measurements updates on the Kalman filter algorithm. Measurement model is modelled as

$$\mathbf{z} = \begin{bmatrix} \mathbf{v} \\ \omega \end{bmatrix} + \eta \tag{3.68}$$

The measurement error vector $\tilde{\mathbf{z}}$ is computed as in (3.42) and the estimated measurement $\bar{\mathbf{z}}$ is computed as

$$\hat{\mathbf{z}} = h(\mathbf{x}) = \begin{bmatrix} \mathbf{R}_\alpha \mathbf{v} \\ \omega \end{bmatrix} + \boldsymbol{\eta} \tag{3.69}$$

The Jacobian matrix for the measurement error state is

$$\mathbf{H} = \begin{bmatrix} \mathbf{R}_\alpha & \mathbf{0}_{2\times2} & \mathbf{0}_{2\times1} & \mathbf{0}_{2\times1} & \mathbf{0}_{2\times2} & \mathbf{0}_{2\times2} \\ \mathbf{0}_{1\times2} & \mathbf{0}_{1\times2} & 1 & 0 & \mathbf{0}_{1\times2} & \mathbf{0}_{1\times2} \end{bmatrix} \tag{3.70}$$

Assuming that propagated state estimate $\mathbf{x}_{k/k+1}$, propagated covariance matrix estimate $\mathbf{P}_{k/k+1}$, current measurement $\mathbf{z}$, estimated measurement $\bar{\mathbf{z}}$ and the error measurement jacobian matrix $\mathbf{H}$ are computed, then updated state estimates $\mathbf{x}_{k+1/k+1}$ is computed using Algorithm 4.

Algorithm 4 Updating the state estimates of the Kalman filter

1: Residual r according to

$$\mathbf{r} = \mathbf{z} - \hat{\mathbf{z}}$$

2: Covariance of the residual $\mathbf{S}$ as

$$\mathbf{S} = \mathbf{H}\mathbf{P}\mathbf{H}^T + \mathbf{R}$$

2: Kalman gain

$$\mathbf{K} = \mathbf{P}\mathbf{H}^T\mathbf{S}^{-1}$$

3: Error vector

$$\tilde{\mathbf{x}} = \mathbf{K}\mathbf{r}$$

4: Update the state vector as

$$\hat{\mathbf{x}}_{k+1/k+1} = \hat{\mathbf{x}}_{k/k+1} - \tilde{\mathbf{x}}$$

5: Updated covariance matrix $\mathbf{P}_{k+1/k+1}$ is computed as

$$\mathbf{P}_{k+1/k+1} = (\mathbf{I} - \mathbf{K}\mathbf{H})\mathbf{P}_{k+1/k}(\mathbf{I} - \mathbf{K}\mathbf{H})^T + \mathbf{K}\mathbf{R}\mathbf{K}^T$$

Chapter 4

Experimental Results and Discussions

In this section, simulation and experimental results for ToF ego-motion estimation algorithms, INS algorithm, calibration of ToF-IMU system and fusion of ToF data with IMU data to estimate a more robust trajectory estimation are presented and the results are discussed.

All the algorithms are implemented in MATLAB® and tested on an offline dataset. Since the algorithms are implemented on MATLAB®, the evaluation of the algorithms will be on the accuracy of trajectorys estimation. The computational cost of the algorithm is out of the scope of this dissertation. The dataset used is collected using a PC running Robot Operating System (ROS) on Ubuntu. ROS is an open source distribution that provides libraries and tools to help software developers create robot applications. It provides hardware abstraction, device drivers, libraries, visualizers and etc. This has an advantage of synchronising all the sensors and the groundtruth system to the same time frame which makes it easy to compare and evaluate trajectory estimation results. The collected dataset is saved in rosbag which is a recording tool under ROS. A package written in C++ is used to convert this data to a format that can be read in MATLAB®. The experimental set-up and evaluation measures are presented in next section.

4.1 Experimental Set-Up

The final system is suppose to operate in an underground mining environment. Because of inherent difficulties in obtaining the groundtruth in underground mines, an artificial mine stope shown in Figure 4.1 is used instead to test the algorithms. The artificial mine is designed to be as close as possible to the actual underground mine stope. The real underground mine stope is characterised with the absence of unique landmarks, rough surfaces, roof supported by pillars and darkness. In general, the size is about 1 m high, 30 m wide and 30 m in length. To validate the dataset used in the experiments, the artificial mine used is built to be as close as possible to the real mine, see Figure 4.1 for the artificial mine. The artificial mine has rough surfaces and the roof is supported by wooden pillars and pillars built from bricks. To limit the number of landmarks and features like in the real underground mines, the mine surface in made of material similar to that found in platinum mines. All the data is collected at night after 19:30 South African Standard Time (SAST) with all light off to imitate darkness in the underground mines. In dimensions, the artificial mine is 1 m high, 5 m wide and 5 m in length. The stope is placed in the laboratory and the Vicon system is used to provide accurate trajectory estimations that are then be used as groundtruth. As it

Figure 4.1: Artificial mine stope

can be seen in Figure 4.1, the floor is very rough. A platform capable of operating in this terrain must be used. An iRobot® 510 PackBot®[1] is used, shown in Figure 4.2. In some experiments where the robot is moving in a 2D planar surface, a pioneer P3-Dx mobile robot is used as shown in Figure 4.3

[1]PackBot® is a mobile packform capable of climbing stairs and operate in rubble in environments. It was designed for United States military.

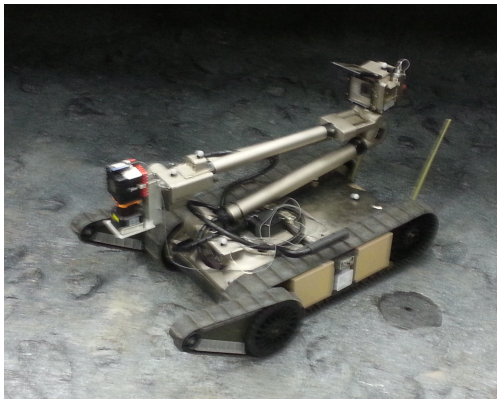


Figure 4.2: iRobot 510 PackBot with the ToF-IMU system mounted rigidly on the manipulator

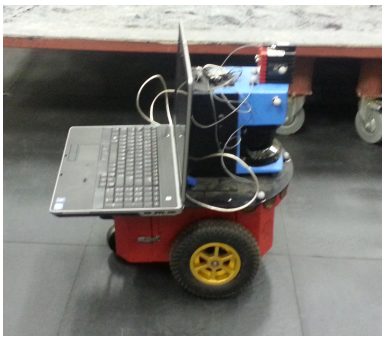


Figure 4.3: Pioneer mobile robot with ToF-IMU system mounted rigidly

4.2 Evaluation Measures

To evaluate the results, the groundtruth of the trajectory of the mobile robot is used. Since the exact motion is known for the simulated data, the groundtruth is found from motion model, while the Vicon system is used in real data experiments. Vicon system is a motion capture system that uses passive infra-red reflective markers to track pose of an object in space. It has a sub-millimeter accuracy and an update rate of more than 100 Hz.

Using the same motion model shown in Figure 3.7, let $\mathbf{T}_t^e$ be the estimated pose at time t and $\mathbf{T}_t^g$ be the groundtruth pose at time t. The transformation $\mathbf{M}_t^{abs}$ between $\mathbf{T}_t^e$ and $\mathbf{T}_t^g$ represents an absolute error between the estimated pose and the groundtruth pose

and is computed as

$$\mathbf{M}_t^{abs} = (\mathbf{T}_t^g)^{-1}\mathbf{T}_t^e = \begin{bmatrix} \Delta\mathbf{R}_{abs} & \Delta\mathbf{t}_{abs} \\ 0 & 1 \end{bmatrix} \tag{4.1}$$

The translation error $e_{abs,\Delta t}$ and rotational error $e_{abs,\Delta\mathbf{R}}$ are computed as

$$e_{abs,\Delta\mathbf{t}} = \|\Delta\mathbf{t}_{abs}\| \tag{4.2}$$

$$e_{abs,\Delta\mathbf{R}} = \|[\alpha\ \beta\ \gamma]^T\| = \|\mathbf{f}_{\Delta\mathbf{R}}(\Delta\mathbf{R}_{abs})\| \tag{4.3}$$

where $[\alpha\ \beta\ \gamma]^T$ are the Euler angles, $\mathbf{f}_{\Delta\mathbf{R}}$ is a function that converts a 3×3 rotational matrix to Euler angles and $(\|\ \|)$ is a 2-norm. See appendix D for different representations of orientation and how to convert from one form to another.

The incremental errors $\mathbf{M}_t^{inc}$ are measured as the sum of the difference of the change in real pose $\Delta\mathbf{T}_t^g$ and change in estimated pose $\Delta\mathbf{T}_t^g$

$$\mathbf{M}_{inc,t} = (\Delta\mathbf{T}_t^g)^{-1}\Delta\mathbf{T}_t^e = \begin{bmatrix} \Delta\mathbf{R}_{inc,t} & \Delta\mathbf{t}_{inc,t} \\ 0 & 1 \end{bmatrix} \tag{4.4}$$

and the incremental translational and rotational errors are then computed as

$$e_{inc,\Delta\mathbf{t}} = \sum_i \|\Delta\mathbf{t}_{inc,i}\| \tag{4.5}$$

$$e_{inc,\Delta\mathbf{R}} = \sum_i \|\mathbf{f}_{\Delta\mathbf{R}}(\Delta\mathbf{R})\| \tag{4.6}$$

For the ToF ego-motion estimation, RMS errors and number of iterations are also used as a measure of evaluation. RMS error measures the error that remains after two point clouds have been registered. This value is affected by the ToF error which ranges between 10 mm to 50 mm for SR4000 ToF camera. The number of iterations gives an indication of how many iterations the ICP algorithm performs until the algorithm converges. A maximum number of iterations is set to 2000 for all the experiments therefore, an iteration of 2000 would mean the ICP algorithm did not converge.

4.3 ToF Ego-Motion Estimation

Before the ego-motion estimation algorithms are applied, the preprocessing steps that are described in Section 3.1 are applied. This includes a 40 minutes warm-up time,

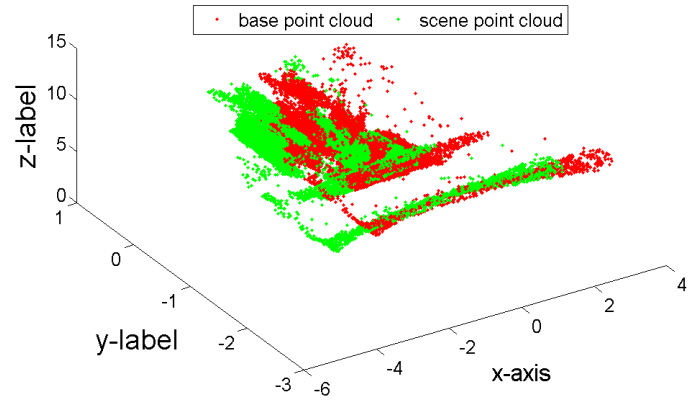
jump edge filter, phase-wrap filter and rejection of boundary points. The algorithms tested are point-to-point ICP, point-to-plane ICP and SIFT-based trajectory estimation. These algorithms are tested on synthetic data and real data.

4.3.1 Simulation Experiments

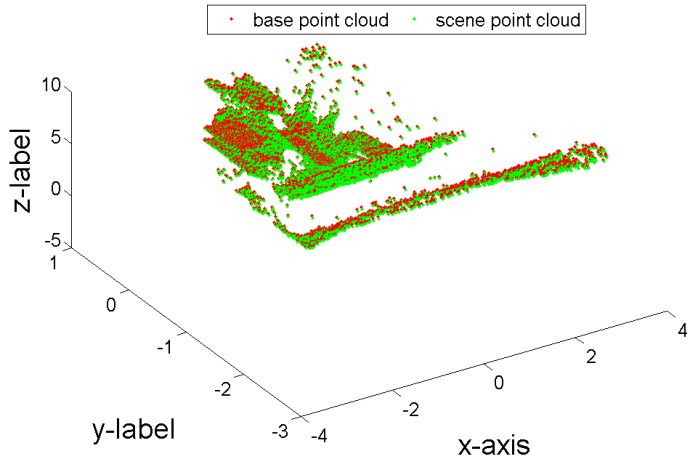
For synthetic data, a point cloud from a ToF camera was transformed using motion model (3.5). A zero mean gaussian noise with a standard deviation of 30 mm was added on the point clouds to model ToF measurement errors. The translation steps ranged from 10 cm to 25 cm in each axis and orientation ranged between 1° and 10° in each axis represented in Euler representation.

Point-to-point ICP and point-to-plane ICP trajectory estimation algorithms are applied on generated synthetic data. Figure 4.4 shows one example of the two point clouds before and after ICP registration. The point cloud was translated by $\mathbf{t} = [0.2,\ 0,\ 0.2]^T$ m and rotated by 10° on the y-axis. The final transformation from the ICP algorithm is $[0.244,\ 0.002,\ 0.173]^T$ for translation and $[0.241°,\ 9.8989\ °,\ 0.109°]$ Euler angles for rotation.

To show the accuracy of the algorithm, the point clouds were transformed one axis at a time and also all three axes at the same time both in translation and rotation. See Table 4.1 and Table 4.2 for the summary of the point-to-point and point-to-plane ICP results.



(a)



(b)

Figure 4.4: (a) shows base point cloud and scene point cloud before registration and (b) shows base and scene point clouds after ICP registration

For the translational motion either in one axis or all three axes, both point-to-point and point-to-plan ICP algorithms can find the correct transformation with an accuracy of less than 10 mm and 0.5° for translation and rotation respectively. For rotational motion in one axis, the algorithms can find the accurate transformation with an accuracy of less than 1° up until the rotation is greater than 8°. When the point cloud is rotated in all axes, both point-to-point and point-to-plane ICP algorithms can not find the correct transformation. The RMS error is still below 20 mm which means that ICP is being trapped in local minima since it is converging. This can be avoided by IMU as an Attitude and Heading Reference System (AHRS) to provide rotational

Table 4.1: Point-to-point ICP simulated results showing the actual transformation and the estimated transformation from the ICP algorithm together with the final RMS error

Actual transformation		Point-to-point		
translation (m)	rotation (°)	translation (m)	rotation (°)	RMS (m)
[0.05; 0.0; 0.0]	[0; 0; 0]	[0.048; 0.002; 0.001]	[-0.01; 0.01; -0.01]	0.0184
[0.10; 0.0; 0.0]	[0; 0; 0]	[0.095; -0.002; 0.000]	[0.07; -0.03; -0.31]	0.0195
[0.15; 0.0; 0.0]	[0; 0; 0]	[0.148; -0.003; 0.002]	[0.00; 0.01; 0.30]	0.0182
[0.20; 0.0; 0.0]	[0; 0; 0]	[0.197; -0.003; 0.003]	[-0.07; -0.01; 0.39]	0.0181
[0.05; 0.05; 0.05]	[0; 0; 0]	[0.051; 0.051; 0.050]	[0.03; -0.01; -0.14]	0.0190
[0.10; 0.10; 0.10]	[0; 0; 0]	[0.094; 0.102; 0.099]	[-0.045; -0.02; -0.08]	0.0177
[0.15; 0.15; 0.15]	[0; 0; 0]	[0.147; 0.145; 0.150]	[0.05; -0.01; 0.05]	0.0191
[0.20; 0.20; 0.20]	[0; 0; 0]	[0.195; 0.202; 0.204]	[-0.04; 0.02; -0.34]	0.0193
[0.0; 0.0; 0.0]	[0; 2; 0]	[0.002; 0.000; 0.001]	[0.03; 2.02; 0.07]	0.0174
[0.0; 0.0; 0.0]	[0; 4; 0]	[0.002; 0.001; 0.003]	[0.05; 3.99; 0.05]	0.0175
[0.0; 0.0; 0.0]	[0; 6; 0]	[0.001; 0.004; 0.013]	[0.32; 5.97; 0.15]	0.0181
[0.0; 0.0; 0.0]	[0; 8; 0]	[0.006; 0.006; 0.028]	[0.92; 7.97; 0.15]	0.0173
[0.0; 0.0; 0.0]	[0; 10; 0]	[0.006; 0.008 0.028]	[1.68; 9.90; 0.10]	0.0178
[0.0; 0.0; 0.0]	[1; 1; 1]	[0.002; -0.001; 0.000]	[3.16; 1.03; 2.91]	0.0186
[0.0; 0.0; 0.0]	[2; 2; 2]	[-0.003; -0.001; 0.001]	[6.15; 2.03; 6.26]	0.0192
[0.0; 0.0; 0.0]	[3; 3; 3]	[-0.005; -0.001; 0.009]	[9.35; 2.91; 9.36]	0.0182
[0.0; 0.0; 0.0]	[4; 4; 4]	[-0.006; 0.002; 0.016]	[12.69; 3.76; 12.08]	0.0192

transformation as initial guesses for ICP.

Table 4.2: Point-to-plane ICP simulated results showing the actual transformation and the estimated transformation from the ICP algorithm together with the final RMS error

Actual transformation		Point-to-plane		
translation (m)	rotation (°)	translation (m)	rotation (°)	RMS (m)
[0.05; 0.0; 0.0]	[0; 0; 0]	[0.048; 0.002; 0.001]	[-0.01; 0.00; -0.01]	0.0184
[0.10; 0.0; 0.0]	[0; 0; 0]	[0.095; -0.002; -0.000]	[0.07; -0.02; -0.31]	0.0195
[0.15; 0.0; 0.0]	[0; 0; 0]	[0.148; -0.003; 0.002]	[0.00; 0.01; 0.30]	0.0182
[0.20; 0.0; 0.0]	[0; 0; 0]	[0.196; -0.002; 0.003]	[-0.07; -0.01; 0.39]	0.0180
[0.05; 0.05; 0.05]	[0; 0; 0]	[0.051; 0.051; 0.049]	[-0.03; -0.01; -0.14]	0.0190
[0.10; 0.10; 0.10]	[0; 0; 0]	[0.094; 0.102; 0.099]	[-0.05; -0.01; -0.08]	0.0177
[0.15; 0.15; 0.15]	[0; 0; 0]	[0.147; 0.145; 0.149]	[0.05; -0.01; 0.05]	0.0191
[0.20; 0.20; 0.20]	[0; 0; 0]	[0.195; 0.201; 0.204]	[-0.04; 0.02; -0.34]	0.0193
[0.0; 0.0; 0.0]	[0; 2; 0]	[0.000; 0.001; 0.000]	[-0.08; 2.00; 0.26]	0.018
[0.0; 0.0; 0.0]	[0; 4; 0]	[0.006; 0.004; -0.004]	[-0.53; 4.06; 0.23]	0.0181
[0.0; 0.0; 0.0]	[0; 6; 0]	[0.000; 0.004; -0.010]	[-1.05; 6.00; 0.46]	0.0177
[0.0; 0.0; 0.0]	[0; 8; 0]	[-0.003; -0.004; -0.005]	[-0.53; 7.97 ; 0.44]	0.0186
[0.0; 0.0; 0.0]	[1; 1; 1]	[0.006; 0.000; -0.001]	[3.07; 1.12; 2.87]	0.0185
[0.0; 0.0; 0.0]	[2; 2; 2]	[0.002; -0.002; -0.004]	[5.98; 2.18; 6.14]	0.0182
[0.0; 0.0; 0.0]	[3; 3; 3]	[0.005; 0.002; -0.009]	[8.36; 3.40; 8.24]	0.0182
[0.0; 0.0; 0.0]	[4; 4; 4]	[0.001; 0.010; -0.011]	[10.45; 4.60; 11.48]	0.0167

4.3.2 Real Data Experiments

ToF trajectory estimation algorithms are applied on real ToF dataset collected from the lab simulating the mine. A mobile robot as shown in Figure 4.3 is moving in 2D planar surface in the lab as shown in Figure 4.5. Note that even though the robot is moving in a 2D planar surface, a full 6D trajectory algorithm is applied. Trajectory estimation results of point-to-point and point-to-plane ICP algorithms are presented in Figure 4.6 below and are compared to the groundtruth provided by the Vicon motion capture system.

The path is approximately 8 m long. The average linear velocity is 0.2 m/s while the angular velocity is $10°/s$. Both point-to-point ICP and point-to-plane ICP trajectories estimates follow the correct path until the mobile robot starts to rotate. After turning, the error in the rotation causes both the trajectory estimation algorithms to deviate from the actual path. The reason for the high error when the robot is turning is due to the ToF errors. When the ToF camera has high angular velocities, emitted light is measured back at wrong pixels which gives wrong measurements. From Figure 4.6, Point-to-plane ICP algorithm seems to be too far from the groundtruth path. The

Figure 4.5: The environment where the robot moves in a 2D planar surface, with pillars to imitate the mining environment

glitch at the corners from the ICP trajectories estimates is caused by the fact that ToF camera is mounted with an offset from the mobile robot centre of rotation. The glitch only appears when the robot is turning to the left.

The absolute and incremental errors for both translation and rotation are presented in Figure 4.7 where Figure 4.7(a) and Figure 4.7(b) show the absolute errors. The absolute translational error at the end of the path was 4.33 m for point-to-plane ICP and 1.03 m for point-to-point ICP. For rotation, it was 48.67° and 70.8° for point-to-plane and point-to-point ICP respectively. According to the absolute errors, point-to-point produces better results for translation but suffers large amount of errors in rotation. The reason why point-to-plane is producing good results for rotation lies in the fact that it minimises error between a point and a plane instead on a closest points as in point-to-point ICP. Moreover, absolute error does not give a proper evaluation of the error since it is dependant on the path. From the incremental errors shown in Figure 4.7(c) and Figure 4.7(d), it can be noted that point-to-point ICP is producing better results. The point-to-point ICP final incremental error was 7.32 m and 119.6° for translation and rotation respectively while for the point-to-plane ICP it is 13.57 m and 123.8° for translation and rotational errors respectively. Note that the rate of error increase is high while the robot is rotating and afterwards the error continues to increase. This is caused by motion blur which is highest for ToF camera during rotational motion. Motion blur produces erroneous point clouds that affect the trajectory estimation algorithms.

The RMS errors and number of iterations from both point-to-point and point-to-plane

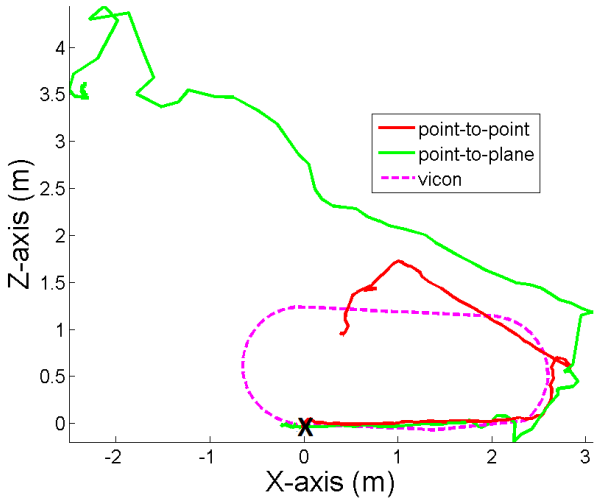
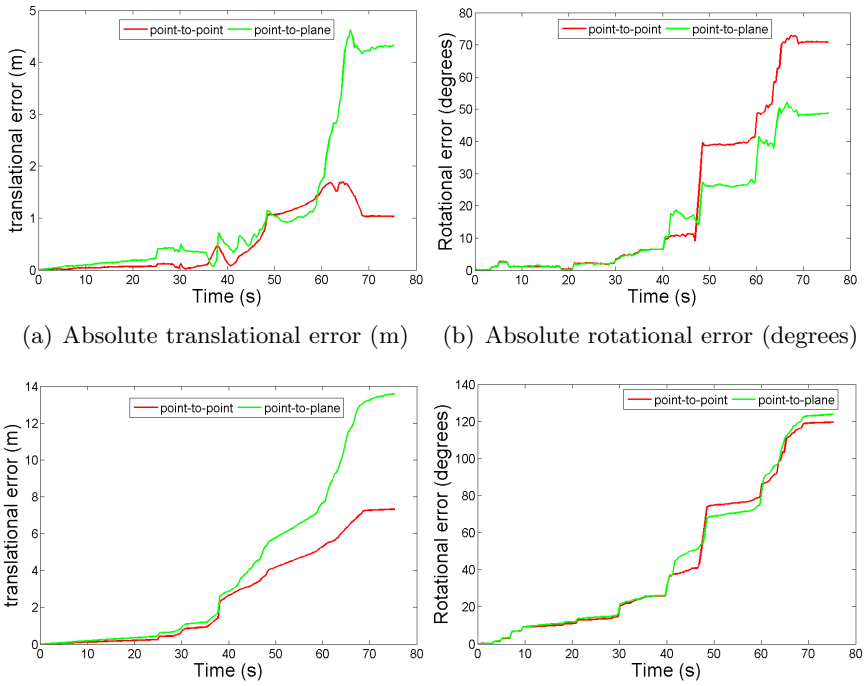


Figure 4.6: Comparison of point-to-point (–) and point-to-plane (–) ICP with the Vicon (–) estimates. The starting point is shown by a cross(**X**)

ICP are presented in Figure 4.8. The average RMS error is 20 mm while the robot was only moving in translational motion. The RMS error increased instantly to an average of 60 mm when the robot was turning as it can be seen in Figure 4.8(b) between time $[40 : 50]$ and time $[60 : 70]$. This supports the fact that the error of ToF ego-motion is high when the robot is turning. This can be verified by the number of iterations (see Figure 4.8(a)) where the maximum number of iteration is reached for time interval $[40 : 50]$ and time $[60 : 70]$, showing that ICP did not converge. The results of RMS errors and number of iterations still support point-to-point ICP algorithm as the most suited for this application since it produces less RMS error and number of iterations even though the difference is very small.

The SIFT algorithm was applied on the same dataset and the trajectory estimation was compared to the Vicon groundtruth and the results is shown in Figure 4.9. The SIFT features were too few and required SIFT algorithm to be applied on image to image which increases the drift error.

To show that the SIFT algorithm is working, data was collected in the environment consisting of enough features. Features ranged from posters, calibration boards and boxes with different colors. The results of SIFT algorithm was compared to the point-to-point ICP and the groundtruth provided by the Vicon system. The trajectories comparison is shown in Figure 4.10. Note that for this experiment, the ToF camera was mounted



(a) Absolute translational error (m)

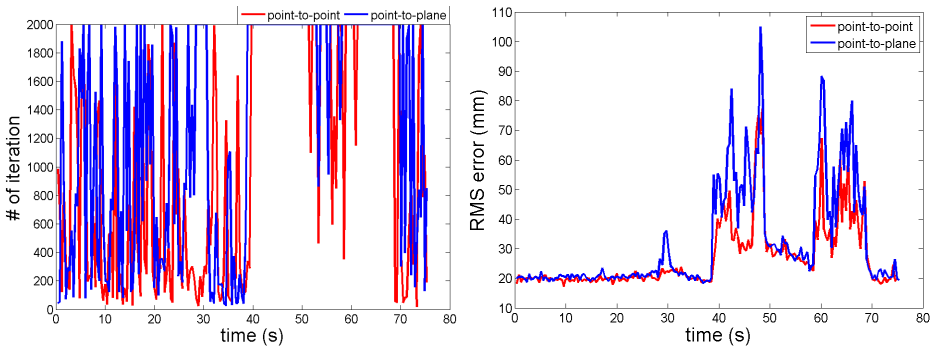
(b) Absolute rotational error (degrees)

(c) Incremental translational error(m)

(d) Incremental rotational error(degrees)

Figure 4.7: ICP based algorithm errors where (a) and (b) represent the absolute translational and rotational errors while (c) and (d) show the incremental translational and rotational errors

on a trolley which was then moved. The motion only consist of translation in 2D plane without rotation. One advantage of SIFT ego-motion estimation algorithm is that it is hundreds of times faster than ICP ego-motion estimation algorithms. This is due to the fact that with SIFT algorithm, the exact point correspondences is known through SIFT features which are very fast.



(a) (b)

Figure 4.8: Iterations and RMS error where (a) shows the number of iteration and (b) shows the RMS error for both point-to-point ICP and point-to-plane ICP

These algorithms were also tested in the real underground gold mine. Data was collected in Kromdraai Gold Mine in Gauteng, South Africa. The mine is no longer operating but it has features that are useful for this application. See Figure 4.11(a) for one of the views in the mine. Because Vicon system can not be taken underground, there is no groundtruth to evaluate these results. The experiment was setup such that a ToF camera was moving in a straight path of length 1.42 m. Refer to Figure 4.11(b) for an experimental setup where two camera tripods with level spirit and a marked string between them were used. The point-to-point ICP, point-to-plane ICP and SIFT based ego-motion estimation algorithms were applied to the dataset. The trajectory estimation compared to the nominal path is shown in Figure 4.12.

In the underground mine dataset, SIFT-based algorithm produced worse results. The SIFT algorithm only detected on average 15 features of which only half were successfully matched to the consecutive image. Normally SIFT algorithm can track 70 features from one image to another. Most of these matched features were not reliable and produced erroneous transformation estimation resulting in an erroneous trajectory estimation. To be able to track features, the SIFT algorithm had to be applied on each image, this increased the drift error as it can seen in Figure 4.12 that SIFT-based trajectory estimation was going back and fourth. Both point-to-point ICP and point-to-plane ICP algorithms produced better results. The path is almost straight as the nominal path. The nominal path is 1.42 m long while the paths from point-to-point and point-to-plane are 1.26 m and 1.18 m respectively. The point-to-point ICP gives better results than the point-to-plane ICP as it can be noted from the total lengths from each algorithm.

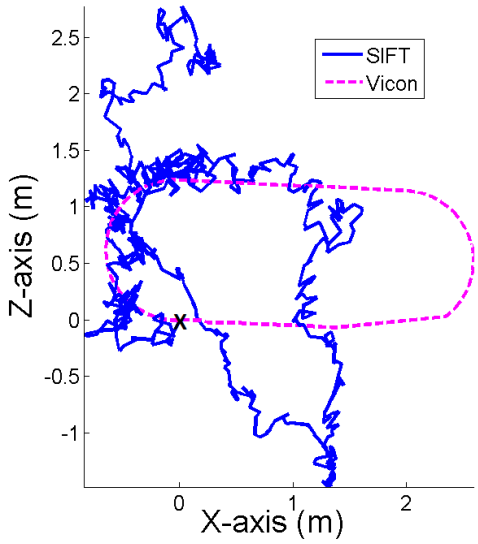


Figure 4.9: Comparison of the SIFT trajectory estimation algorithm and the Vicon groundtruth. The starting point is shown by a cross($\mathbf{x}$)

As it can be seen from Figure 4.12, the ICP path is not perfectly straight, this is due to the errors in the orientation estimation. When the ToF camera was in motion, the orientation was not changing, it was only moving in translation. When evaluating the orientation estimation, it can be seen in Figure 4.13 that the orientation is not constant as it should be.

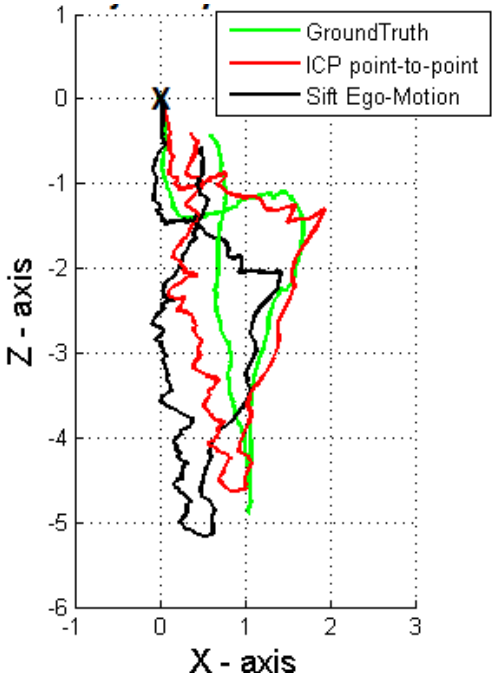


Figure 4.10: Comparison of point-to-point ICP, SIFT algorithm. The starting point is shown by a cross(**x**)

Error in the orientation is high in the y-axis with point-to-point ICP going up to 13° and point-to-plane going up to 17°. Figure 4.14(a) shows the number of iterations and RMS error. RMS error is averaring 25 mm and 20 mm as it can be seen in Figure 4.14(b). The reason for a slight change in the RMS error is due to the change in the distance of the objects on the field of view of ToF camera as it moves. The accuracy of the ICP algorithm, depends on the kind of the environment in which the dataset is collected, the accuracy of 3D point clouds, the field of view of the camera. For every different environment, the ICP algorithm needs to be adapted to that specific environment. Most of the experiments in the literature are performed in a small confined space. This gives an advantage since ToF measurement are of high accuracy for small distances.

The experiments performed in [26] consisted of one motion per axis at a time with a linear velocity of 0.2 m/s and angular velocity of $5°/s$. The incremental error was 17% for translation and 15% for rotation. Note that the error would be high for motion consisting of transformation in more than one axis. The groundtruth is provided by the motion capture system with an accuracy of 2 cm. Dario [17] performed a multi-frame

(a) (b)

Figure 4.11: Real underground experiments, (a) shows how the Kromdraai Gold mine looks while and (b) shows the experimental setup followed for capturing data with some form of groundtruth

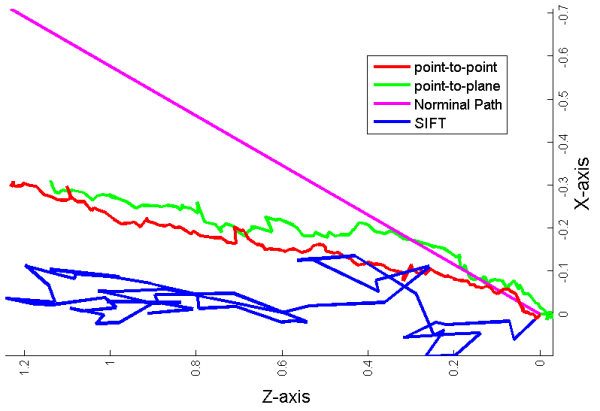
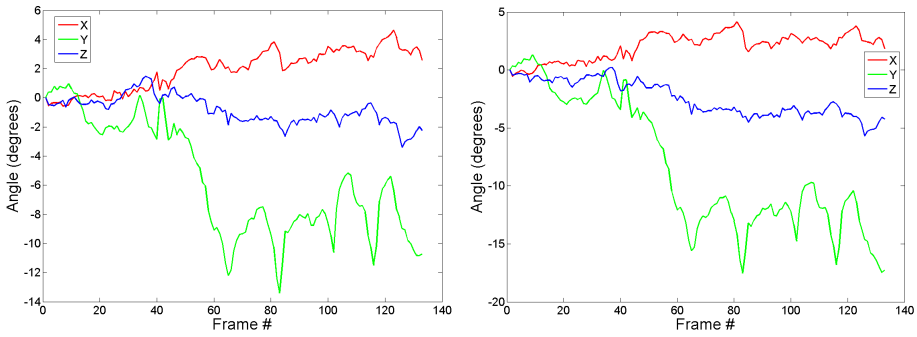


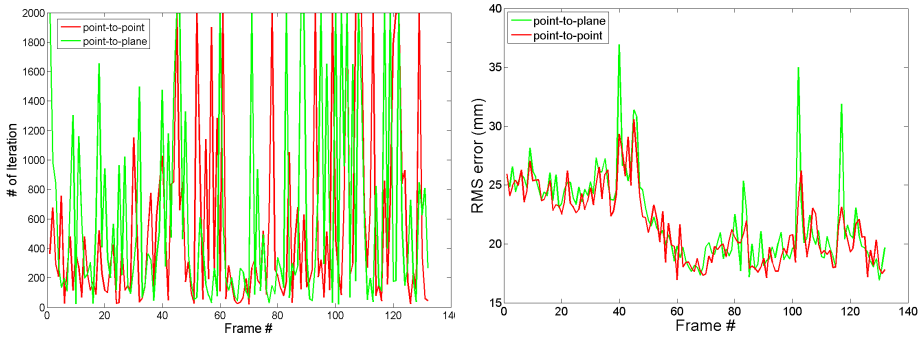
Figure 4.12: Trajectory estimation of real underground mine data. Red line – is the point-to-point ICP, green line – is the point-to-plane ICP, blue line – is the SIFT based and purple line – is the nominal path

registration using SURF algorithm. The ToF camera was mounted on measurement rack and it was moved at a step size of 0.2 m. A translational accuracy of 6.5% was achieved. High accuracy is due to high distance measurements achieved by averaging 30 frames while the camera was stationary. No rotational experiments were performed. May *et al.* [11] collected the dataset by mounting the ToF camera on an industrial robot to be able to obtain the groundtruth of the robot arm trajectory. The ToF camera path was 950 mm and the camera was rotated 360° in the process. The accuracy in rotation for point-to-point ICP was 12% and 20% for SIFT algorithm. The error in translation was 1 m. The simulated data experiments performed in this dissertation is 1.8% for both translational step of 0.2 m and a rotational step of 2°. The real data from the mine show an accuracy of 14% in translation for a 1.42 m straight path. Rotational



(a) (b)

Figure 4.13: Rotation errors where (a) shows the orientation from point-to-point ICP algorithm and (b) shows orientation estimates from point-to-plane ICP, both in Euler angles



(a) (b)

Figure 4.14: Underground experimental results where (a) shows the number of iteration while (b) shows the RMS error for both point-to-point ICP and point-to-plane ICP

error depends on how the camera was rotating and on average the error was 33%. The angular velocity is $10°/s$, which is high compared to $5°/s$ used in [26] and an accuracy of 15% was achieved. The ICP algorithm implemented shows an improvement in translation considering that our robot was moving faster. The rotational error still remains and is handled by the fusion of ToF ego-motion with IMU data.

4.4 INS Estimation

Testing INS algorithm is a challenge in real data experiments because of fast error propagation which is mostly caused by IMU biases and error in the gravity vector estimation (See Figure 3.15 for an example of the INS error propagation). Synthetic data is used to evaluate the INS algorithm. The data is generated using IMU model which is modelled using Simulink® and MATLAB®. The IMU model is built with noise characteristics according to 3DM-GX3®-25 IMU [75]. To compare noise characteristics of the IMU model and 3DM-GX3®-25 IMU, Allan Variance is used. Allan Variance is a technique for analysing a time sequence data to find noise characteristics and stability. For a full description of Allan variance method, see [1]. Ten hours of raw data was collected for a stationary IMU and the computation of Allan Deviation is summarised in Algorithm 5. The averaging time τ is incremented until the number of bins are

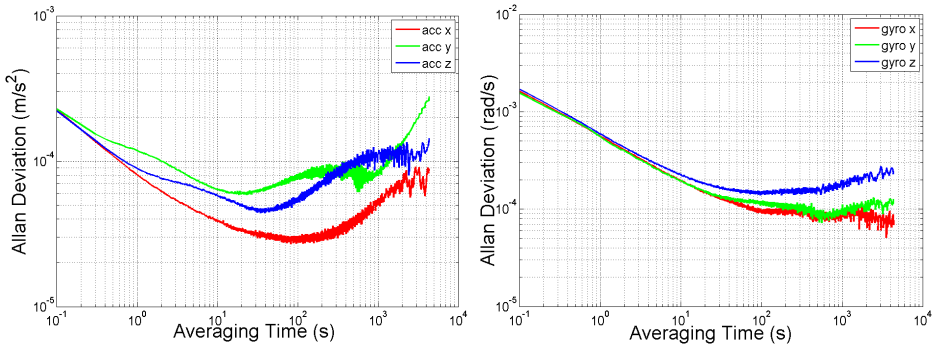
Algorithm 5 Allan deviation computation

1: Set the averaging time (τ) to an initial value
2: **repeat**
3: Divide data into bin based on the averaging time τ
4: Average data in each bin to obtain a list $\{d(\tau)_1, d(\tau)_2, \cdots, d(\tau)_n\}$, where n is the number of bins.
5: Allan Deviation (AD) is then calculated as

$$AD(\tau) = \sqrt{AVAR}\sqrt{\frac{1}{2(n-1)}\sum_i (d(\tau)_{i+1} - d(\tau)_{i+1})^2} \tag{4.7}$$

6: Increment the averaging time (τ)
7: **until** Number of bins are less than 9

reasonably small and AD versus average time τ is plotted on a log-log scale as shown in Figure 4.15. The plots in Figure 4.15 can be used to find quantisation noise, random walk, rate random walk, rate ramp correlated noise, sinusoidal noise and bias stability as described in [1] and a sample of the plot is shown in Figure 4.16. For the purposes of this project, where MEMS IMU is used, the important processes are random walk and bias instability which are summarised in Table 4.3 and Table 4.4. These noise measurements were then used in modelling IMU measurements.



(a) (b)

Figure 4.15: IMU Allan Deviation where (a) is Accelerometer and (b) is Gyroscope

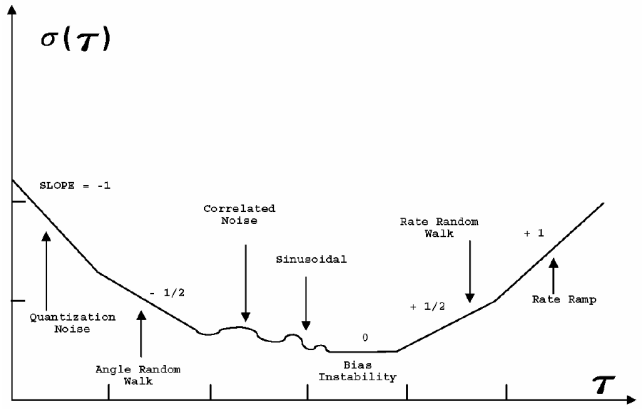


Figure 4.16: A sample of Allan variance analysis plot [1]

	Bias Instability	Velocity Random walk
X axis	2.807e-5 rad/s	8.075e-5 $rad/\sqrt{s}$
Y axis	6.082e-5 rad/s	1.177e-4 $rad/\sqrt{s}$
Z axis	4.651e-5 rad/s	8.898e-5 $rad/\sqrt{s}$

Table 4.3: Accelerometer Noise Measurements

	Bias Instability	Angle Random walk
X axis	9.392e-5 rad/s	5.609e-4 $rad/\sqrt{s}$
Y axis	1.105e-4 rad/s	5.5644-4 $rad/\sqrt{s}$
Z axis	1.47e-4 rad/s	5.9234-4 $rad/\sqrt{s}$

Table 4.4: Gyroscope Noise Measurements

To evaluate the algorithm, both the constant biases of accelerometer and gyroscope biases are set to zero while random walks noise is still included. Constant biases in

angular rates reading causes error in the orientation and since gravity magnitude is of high accelerations, wrong projected accelerations cause fast error propagation. The trajectory estimation of INS algorithm is compared to the actual trajectory as shown in Figure 4.17. The experiment above is 25 seconds long for a path which is 8 m long.

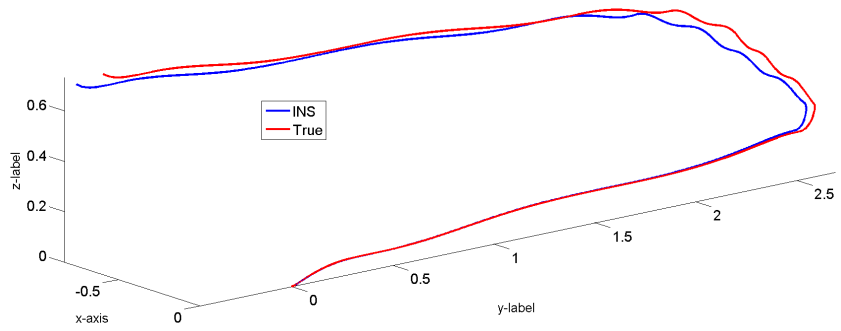


Figure 4.17: Comparison of trajectory estimated using INS algorithm (–) and the true trajectory (–)

Comparing the results to the drift error illustrated in Figure 3.15, it is clear that biases and error in the gravity vector are responsible for fast error growth. This has also been validated by the experiments performed by Woodman [6]. Using the same evaluation measures described in Section 4.2, the absolute error and incremental error of INS algorithm are implemented and the results are shown in Figure 4.18.

The error seems to be increasing at a constant rate as it can be seen in Figure 4.18(c) and 4.18(d). This can also be seen from the trajectory estimation results shown in Figure 4.17 where the INS trajectory is constantly moving away from the actual trajectory. The rate of error increase is greatest when there are rotational velocities. This can be observed from both incremental and absolute errors in Figure 4.18(d) and 4.18(b) respectively.

Note that the INS algorithm is applied on the synthetic data without IMU biases because during the fusion of ToF and IMU, IMU biases are estimated by Kalman filter and are subtracted from IMU measurements. The INS algorithm is applied for time interval of less than a second, hence the error propagation is also small meaning that the estimates are not corrupted by drift. IMU biases are estimated with Kalman filter and removed from the measurements.

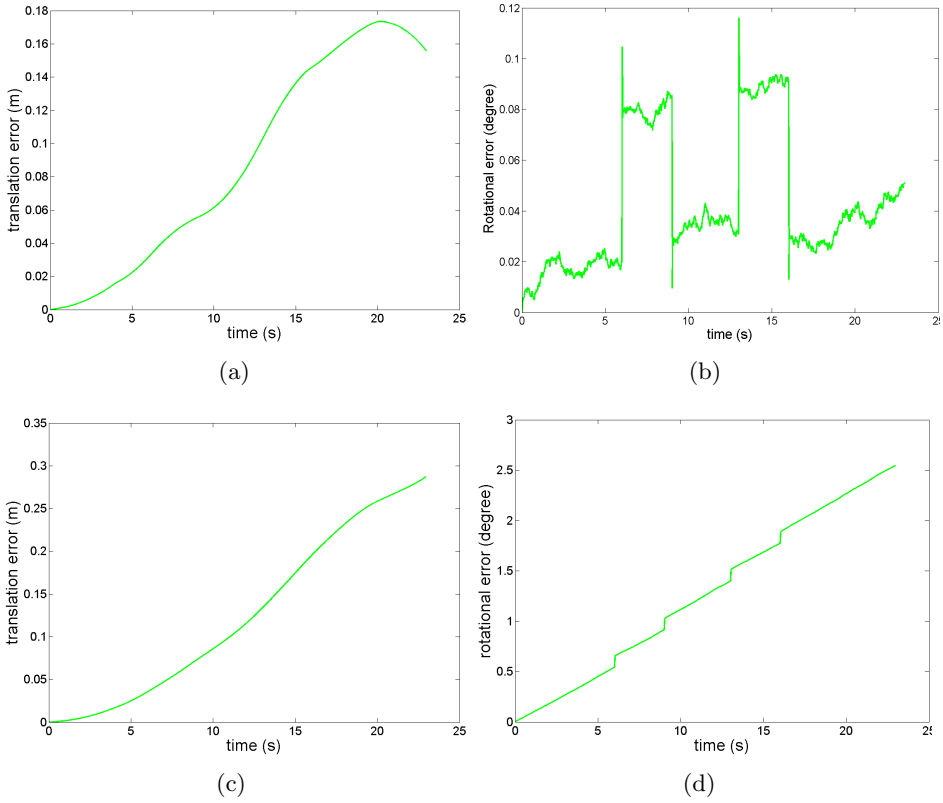


Figure 4.18: Trajectory error where (a) and (b) shows the absolute errors while (c) and (d) shows the incremental errors, expressed in translation and rotation respectively

4.5 ToF-IMU System Calibration

The calibration of ToF-IMU system to estimate the transformation between IMU and ToF camera can be performed using the EKF algorithm as explained in Section 3.4.1 as it has been applied in normal cameras and IMU. To validate the calibration algorithm for ToF camera and IMU, simulated data is generated using Simulink® and MATLAB®. The same IMU model used in Section 4.4 with the noise characteristics similar to 3D-GX3®-25 IMU is used. Since ToF camera is only extracting 3D points of the features, ToF measurements are modelled as

$$\mathbf{z}_{ToF} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} + \eta \tag{4.8}$$

where η is the additive noise with the standard deviation of 10 mm. Note that 10 mm is just a typical accuracy of a ToF camera for a target object of high reflectivity. The calibration board used for simulated data has four features. IMU measurements are sampled at 100 Hz while ToF camera measurements are sampled at 10 Hz. Data was collected for a total time of 20 seconds where the first 5 seconds, the ToF-IMU system was stationary in order to estimate the initial estimates of state vector, covariance matrix and IMU biases.

The initial transformation error was [40 30 40] mm with a standard deviation of 50 mm and [2° 1° 1.5°] with a standard deviation of 2° for translation and rotation respectively. The trajectory estimation from the EKF calibration algorithm compared to the groundtruth is shown in Figure 4.19

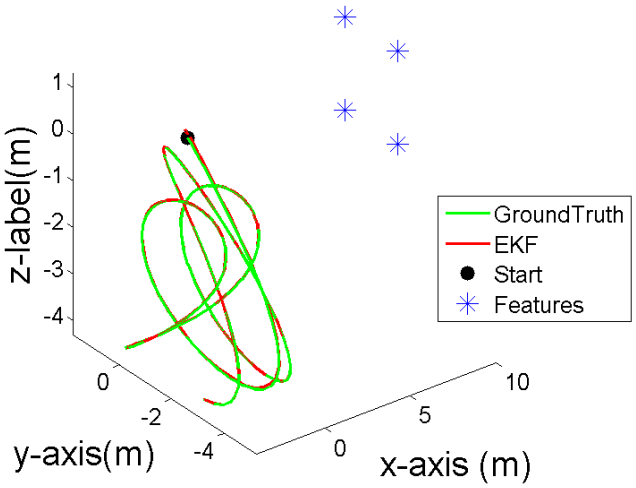
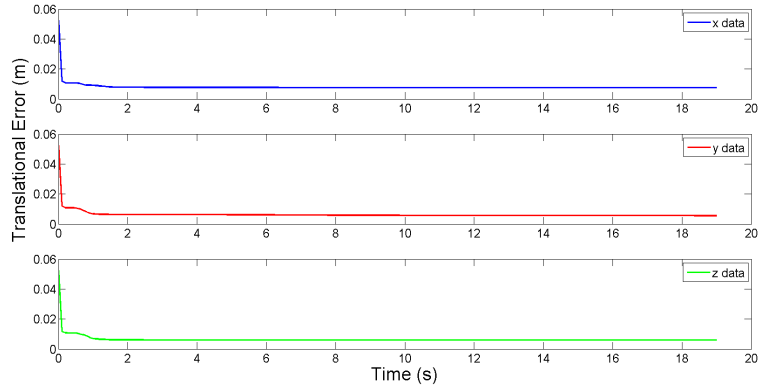
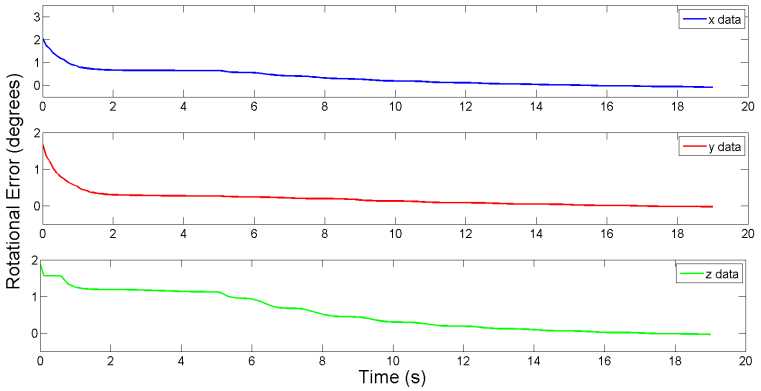


Figure 4.19: Trajectory estimation of ToF-IMU system compared with the groundtruth trajectory. The four features used are also shown

The final translation was found with an accuracy of 5 mm and 0.2° for translation and rotation respectively. See Figure 4.20 for the error in the transformation as computed from the covariance matrix.



(a)

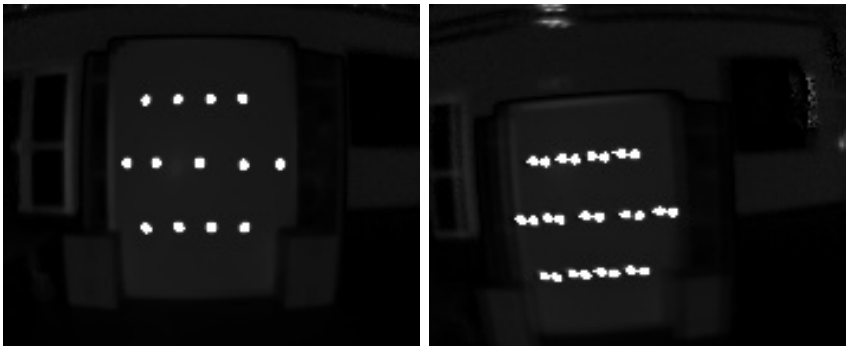


(b)

Figure 4.20: ToF-IMU system transformation error where (a) shows translational error and (b) shows the rotational error

Real experiment data was collected in the lab, see Figure 4.21 for the the experimental setup. For the calibration algorithm to give good results, it requires ToF-IMU system to undergo a full 6 DoF motion. Even though good results can also be achieved by rotational motion only, it requires longer time [60]. Because motion has to consist of rotational motion and it has to be moving fast such that signal-to-noise ratio of IMU measurements is high enough to reduce drift, ToF camera point clouds are corrupted because of motion blur (see Figure 4.22). As it can noted in Figure 4.22, the motion blur is only on the features and the background does not have motion blur. Motion blur in the ToF camera are different from that of normal passive cameras, this means that existing deblurring algorithms in image processing are not applicable.An option for future reference would be to build a calibration board of NIR LED as features since they can be easily detected.

Figure 4.21: ToF-IMU system calibration setup showing ToF-IMU system on the tripod and the calibration board placed vertically



(a) (b)

Figure 4.22: Calibration experiments where (a) shows a ToF image while the Camera is stationary and (b) shows the image with motion blur while the camera is in motion

For this application, the transformation between IMU and ToF camera was estimated using the technical drawing which manufacturers provided with the devices manuals. The focal length of the SR4000 ToF camera [8] is 10 mm. This is used to estimate the transformation which turned out to be accurate for the application.

4.6 Fused ToF-IMU System

The fusion of ToF camera and IMU data when the Pioneer mobile platform is moving in a 2D planar surface as shown in Figure 4.5 is implemented as described in section 3.4.2. ToF ego-motion estimation is performed using point-to-point ICP algorithm as it has been shown that it is the most accurate for this application.

Figure 4.23 shows the Kalman filter fused trajectory of ToF-IMU system compared with the point-to-point ICP trajectory and groundtruth trajectory provided by the Vicon system. The path is 8.7 m long. The linear velocity is 0.2 m/s while the angular

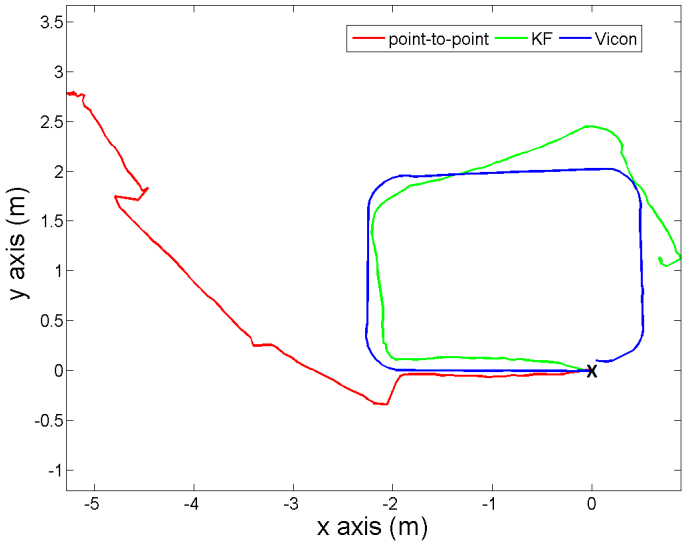


Figure 4.23: Comparison of point-to-point (–), Kalman filter (–) and Vicon groundtruth (–). The starting point is shown by a cross(**X**)

velocity was set to $30°/s$ to show maximum improvement of the fusing IMU data with ToF data. Point-to-point ICP trajectory is accurate until the robot starts having angular velocities as it can be seen in Figure 4.23. By fusing point-to-point ICP results with IMU data using Kalman filter, there is significant improvements especially when the mobile robot is going around the corner or turning. Both the incremental and absolute errors are shown in Figure 4.24.

Because the angular velocities were high, there were high levels of motion blur which is why the ToF ego-motion could not find the correct transformation. In terms of absolute errors, translational error was reduced by 4.72 m and 260.2° for rotational error.

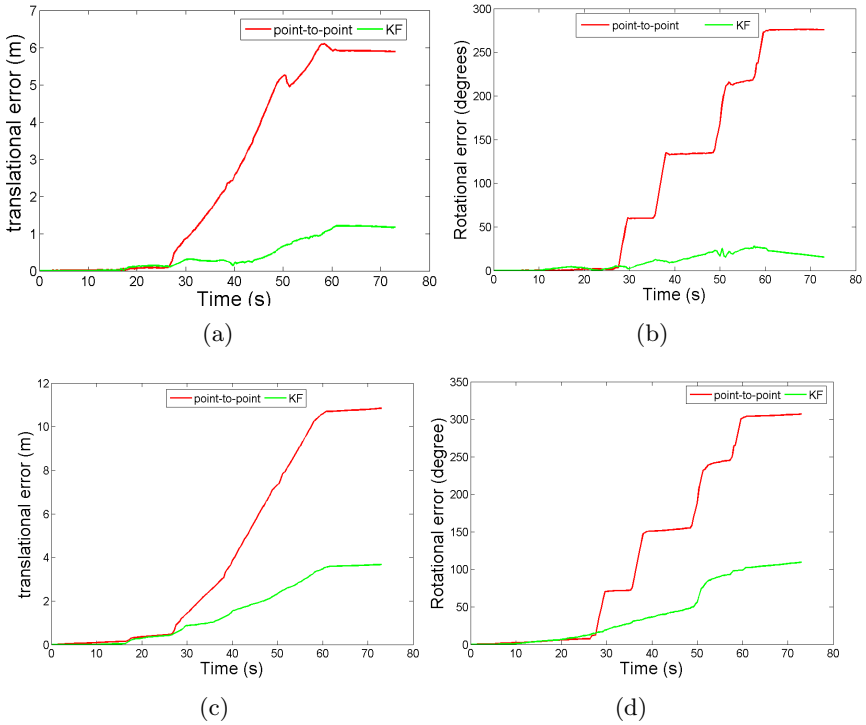


Figure 4.24: Trajectory estimation error where (a) and (b) shows the absolute errors while (c) and (d) shows the incremental errors, expressed in translation and rotation respectively

Incrementally, translational error was reduced by 7.17 m while the rotational error was reduced by 197.5°. This experiment also shows how important orientation is for the estimated trajectory to be as close as possible to the actual trajectory.

This compares with the results of fusing ToF with IMU done by Droeschel *et al.* [64], where SIFT algorithm was used for ToF ego-motion. The robot was similarly moving along a square path with an approximately total length of 4.8 m. For the SIFT algorithm to find enough features, posters and calibration checkerboard were placed in the environment. Incremental translational error was reduced by 1 m and rotational error by 25.4°. One common conclusion from work done by Droeschel *et al.* and the work presented here is that IMU improves the trajectory estimation of ToF camera especially when the robot is turning.

In this chapter, algorithms implemented were tested and evaluated. For the ToF camera's ego-motion, the algorithm most suited for the underground mining environment

is the point-to-point ICP. This algorithm would be applicable to all environment where there are no unique features, otherwise SIFT algorithm would be a more suited algorithm. ToF camera's ego-motion algorithms experience high errors when the robot is undergoing rotational motion. The error was caused by the architecture of the ToF camera and it was handled by fusion the ego-motion with IMU data. By fusing these two sensors a more robust and accurate trajectory of the mobile robot was estimated. The algorithm is applicable to 2D planar surfaces.

The limitation to the algorithm is that it needs the robot to be operating in 2D planar environments, whereas the real underground mining environment consist of rough terrain.

Chapter 5

Conclusions and Future Work

5.1 Conclusions

This study has presented the design and implementation of a system capable of estimating the trajectory of a robot operating in underground mining environments. Robot trajectory is traditionally estimated using GPS for outdoor applications or laser scanners and camera for indoor environments. Because in underground environments there is no GPS signal for GPS to function, few distinct features for a normal camera to be used and Laser scanners consume lots of power, these sensors are avoided. This work investigated the fusion of ToF camera with IMU data and in the process, some of the questions that were answered are

- Which methods are best for filtering out erroneous points of ToF camera.
- Which algorithm is best for ego-motion of ToF camera in underground mining environments.
- By how much does fusion of ToF ego-motion and IMU data improve the trajectory estimation of a robot.

The study conducted for ToF camera errors suggested that most of the ToF errors that fall under systematic errors have been reduced substantially by manufacturers calibration. The remaining non-systematic errors, of which the most dominant ones are jump edge points, phase-wrapped points and uniform illumination of NIR light, are considered . A jump edge filter that was able to detect and reject jump edge points from the point cloud images was implemented. A phase-wrap filter that uses the product of amplitude images and squared distance images was developed. The effect of non-uniform illumination was handled by rejected boundary pixel points. Together with a

40 minutes warm-up time, these filters were applied on the ToF data before ego-motion estimation algorithms were applied to reduce these errors.

ToF Camera Ego-motion

The investigations were conducted to find a more suited algorithm for the ToF camera ego-motion estimation. The algorithms considered were point-to-point ICP, point-to-plane ICP and SIFT-based ego-motion estimation algorithm. ICP methods iteratively finds the transformation by minimising the distance between two point cloud images. The difference between point-to-point ICP and point-to-plane ICP is that point-to-point minimises the sum of the squared distance between points while point-to-plane minimises the sum of the squared distances between a point and a plane on the other point cloud. The SIFT-based algorithm tracks features from one amplitude image of a ToF camera to the next and use the corresponding 3D points to estimate the transformation. The SIFT-based method is very fast compared to the ICP based methods, this makes it more applicable to real-time applications.

Simulated data was used to evaluate point-to-point ICP and point-to-plane ICP. It is concluded that these algorithms can estimate an accurate transformation for translation motion despite getting trapped in local minima for transformation consisting of high rotations. In testing all three algorithms on a real dataset, it is concluded that point-to-point is more suited for the underground mine stope environment. The SIFT algorithm fails to match features in the underground data because of too few unique features. To validate the SIFT algorithm, it was tested on the environment with enough features and it produced a trajectory similar to point-to-point ICP algorithm. Point-to-plane ICP produced similar results to point-to-point but the error produced was higher than the error produced by point-to-point. The extra error is a combination of linearisation error and the fact that the stope consists mostly of circular pillar and point-to-plane ICP is most suited for an environment with planar surfaces.

A ToF camera produces erroneous point clouds when it is rotating. This introduces high errors in the trajectory estimation algorithms. RMS errors show on average low value for translation motion and high value for rotation motion. RMS errors give an indication of ToF errors since it gives the root squared mean of two point clouds after registrations. It was noted that the ICP algorithm does not always converge when there is rotational motion.

INS Algorithm

The investigation was conducted on INS for trajectory estimation using IMU. INS suffers from high error propagation caused by IMU biases and noise in the IMU measurements and the error in the gravity vector estimates. A full 6D INS algorithm is implemented and tested on the simulated dataset generated by an IMU model built using MATLAB® and Simulink®. Allan Variance was used to find noise characteristics of IMU and these noise characteristics were used in modelling IMU model. The INS algorithm implemented uses quaternion. The algorithm is applied on a dataset where the exact gravitational vector and constant biases are known and this reduces the rate of propagation for the algorithm to produce accurate results. For real data experiments, biases are estimated using Kalman filter.

Fusion of ToF Camera and IMU

The study shows that before fusion can be performed between any two sensors, the transformation between these sensors must be estimated. For our application, this transformation was estimated using technical drawings of IMU and ToF camera. The accuracy of this transformation can be increased by refining the estimated transformation using EKF algorithm. The algorithm was adopted from normal camera and IMU calibration. It was validated with simulated data. The EKF algorithm was not tested on real data because of the high motion blur of reflectance objects which were used as point features.

The investigation conducted here for fusion of ToF camera ego-motion estimates and IMU data involved a mobile robot moving in a 2D planar surface. The ToF ego-motion was estimated using point-to-point ICP algorithm and the fusion was performed using Kalman filter algorithm. IMU measurements were used to predict the robot's velocities while ToF camera was providing measurements to update IMU predictions. The experimental results showed an improvement on the ToF camera ego-motion especially when the robot was turning. The fused trajectory was compared to the groundtruth trajectory provided by the Vicon system.

5.2 Future Work

This work has presented a system for trajectory estimation of mobile robot operating in underground environments where there are no GPS signals and no unique features or landmarks. The work was divided into ToF camera ego-motion, INS, calibration of ToF camera and IMU and Fusion of ToF camera ego-motion and IMU data. ToF camera's ego-motion was computed using point-to-point ICP while fusion was performed using Kalman filter for a mobile robot moving on a 2D planar surface. The results show that fused trajectory estimation of ToF camera and IMU is more accurate than ToF camera trajectory estimation alone. To further increase the accuracy of fused trajectory results, the transformation between IMU and ToF camera must be refined using EKF algorithm and to reduce the effect of motion blur, a different calibration board with NIR LEDs as point features must be used. These NIR LEDs can be setup such that they emit different NIR light to that of a ToF camera so that they can be easily detected. NIR LEDs board has been used in [15] for photogrammetric calibration of ToF camera. The algorithm proposed here needs to be modified to handle a full 6D motion. Before a full 6D algorithm can be applied, a more accurate gravity vector must be estimated. A method that can be used to find the groundtruth when the robot is operating in the real mine is proposed in Appendix E, but still needs to be improved.

Appendix A : ToF Camera Measurement Principle

Time of flight (ToF) cameras are sensors that are able to provide range images, amplitude images and sometimes intensity images at a video frame rate. The main components are Light Emitting Diodes (LEDs) and Charge-Coupled Device/ Complementary Metal Oxide Semiconductor (CCD/CMOS). The LEDs illuminates the environment with a modulated Near Infrared (NIR) light, which is reflected by the object back to the camera and is measured by the CCD/CMOS sensor of the camera. The CCD/CMOS demodulate the measured signal and distance which is proportional to the phase shift is computed using phase-shift principle [9]. The illuminated light is assumed to be sinusoidal.

Figure 5.1 shows the intensity of the emitted NIR light and measured NIR light for one pixel. At each pixel, four sampling points at an interval of 90° are used to compute phase shift φ_i, amplitude a_i and offset b_i, where i is pixel number [76] [9].

$$\varphi_i = \arctan\left(\frac{c(\tau_0) - c(\tau_2)}{c(\tau_1) - c(\tau_3)}\right) \tag{5.1}$$

$$a_i = \frac{\sqrt{([c(\tau_0) - c(\tau_2)]^2 + [c(\tau_1) - c(\tau_3)]^2)}}{2} \tag{5.2}$$

$$b_i = \frac{c(\tau_0) + c(\tau_1) + c(\tau_2) + c(\tau_3)}{4} \tag{5.3}$$

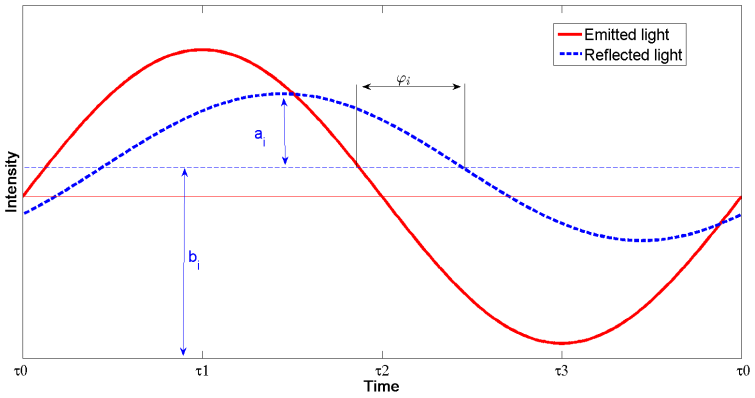


Figure 5.1: Emitted (solid red) and received (dashed blue) light of the ToF camera for one period (one wavelength)

The distance d_i is then computed as

$$d_i = \frac{c}{2f_m}\frac{\varphi_i}{2\pi} \tag{5.4}$$

where c is the speed of light, f_m is the modulation frequency and the non-ambiguity distance l of the camera is computed as

$$l = \frac{c}{2f_m} \tag{5.5}$$

SR4000 Swiss Ranger ToF camera from Mesa Imaging is used throughout the experiments of this dissertation, see the specifications in Table 5.1.

Table 5.1: Specification for SR4000 Swiss Ranger ToF camera

Pixel array size	176(h) × 144 (v)
Field of view	43°(h)×34°(v)
Calibrated distance	0.8m ⟶ 8 m
Absolute accuracy	± 15 mm [1]
Illumination wavelength	850 nm
Maximum frame rate	50 fps
Focus length	10 mm
Dimension	65×65×68 mm
Weight	470 g

Appendix B : Computation of Transformation of ICP Methods

The computation of the transformation $(\mathbf{R}, \mathbf{t})$ of the point-to-point and point-to-plane ICP variants are described. This is a continuation of algorithms described in Section 3.2. In this section we assume two 3D point cloud, base point cloud $\mathbf{B} = \{\mathbf{b}_i | i = 1, \cdots, N\}$ and scene point cloud $\mathbf{S} = \{\mathbf{s}_i | i = 1, \cdots, N\}$. It is assumed that base point $\mathbf{b}_i$ corresponds to $\mathbf{s}_i$.

Point-to-point variant

The point-to-point error metric minimises the sum of the squared distances between base point cloud and the scene point cloud. The error metric is expressed as

$$\xi_{PP}(\mathbf{R}, \mathbf{t}) = \sum_i \|\mathbf{R}\mathbf{s}_i + \mathbf{t} - \mathbf{b}_i\|^2 \tag{5.6}$$

There exist a closed form solution which can either be solved using orthornormal matrix, unit quaternion, dual quaternion or discrete cosine matrix [27]. A discrete cosine matrix approach which was developed by Arun *et al.* [36] is used. The computation of the transformation between base point cloud $\mathbf{B}$ and scene point cloud $\mathbf{S}$ is simplified in Algorithm 6

Algorithm 6 Transformation computation

1: Find the centroid of the point clouds

$$s_{cent} = \frac{1}{N} \sum_i \mathbf{s}_i$$
$$b_{cent} = \frac{1}{N} \sum_i \mathbf{b}_i$$

2: Compute the covariance matrix $\mathbf{H}$

$$\mathbf{H} = \sum_{i=1}^{N} (\mathbf{s}_i - s_{cent}) (\mathbf{b}_i - b_{cent})^T$$

3: Compute the singular value decomposition of $\mathbf{H}$

$$\mathbf{H} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$$

4: Compute the rotation $\mathbf{R}$

$$\mathbf{R} = \mathbf{V}\mathbf{U}^T$$

Note that the determinant of $\mathbf{R}$ must be 1 to be a rotational matrix.

5: Compute the translation $\mathbf{t}$

$$\mathbf{t} = b_{cent} - \mathbf{R}s_{cent}$$

Point-to-Plane

Point-to-plane error metric minimises the sum of the squared distances between each of the point on the base point cloud and the tangent plane on the corresponding point of the scene point cloud. The error metric function is represented as

$$\xi_{PL}(\mathbf{R},\mathbf{t}) = \sum_i [(\mathbf{R}\mathbf{s}_i + \mathbf{t} - \mathbf{b}_i)\cdot\mathbf{n}_i]^2 \tag{5.7}$$

Equation (5.7) does not have a closed form solution and is usually solved using non-linear least square methods like Levenberg-Marquardt method. These methods are slow and can not be applied in the real-time applications. Since the transformation is assumed to be very small, small angle approximation ($\sin\theta = 0$ and $\cos\theta = 1$) is used. The rotational matrix is then simplified as

$$\mathbf{R} = \mathbf{R}_\gamma\mathbf{R}_\beta\mathbf{R}_\alpha \tag{5.8}$$

$$= \begin{bmatrix} \cos\gamma\cos\beta & -\sin\gamma\cos\alpha + \cos\gamma\sin\beta\sin\alpha & \sin\gamma\sin\alpha + \cos\gamma\sin\beta\cos\alpha \\ \sin\gamma\cos\beta & \cos\gamma\cos\alpha + \sin\gamma\sin\beta\sin\alpha & -\cos\gamma\sin\alpha + \sin\gamma\sin\beta\cos\alpha \\ -\sin\beta & \cos\beta\sin\alpha & \cos\beta\cos\alpha \end{bmatrix} \tag{5.9}$$

$$= \begin{bmatrix} 1 & -\gamma & \beta \\ \gamma & 1 & -\alpha \\ -\beta & \alpha & 1 \end{bmatrix} \tag{5.10}$$

By representing the transformation as

$$\mathbf{M} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & -\gamma & \beta & t_x \\ \gamma & 1 & -\alpha & t_y \\ -\beta & \alpha & 1 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5.11}$$

the error metric equation (5.7) is expressed as

$$\xi_{PL}(\mathbf{M}) = \sum_i ((\mathbf{M}\mathbf{s}_i - \mathbf{b}_i)\cdot\mathbf{n}_i) \tag{5.12}$$

Which can be simplified to

$$\xi_{PL} = \mathbf{A}\mathbf{x} - \mathbf{c} \tag{5.13}$$

where

$$\mathbf{c} = \begin{bmatrix} n_{1x}b_{1x} + n_{1y}b_{1y} + n_{1z}b_{1z} - n_{1x}s_{1x} - n_{1y}s_{1y} - n_{1z}s_{1z} \\ n_{2x}b_{2x} + n_{2y}b_{2y} + n_{2z}b_{2z} - n_{2x}s_{2x} - n_{2y}s_{2y} - n_{2z}s_{2z} \\ \vdots \\ n_{Nx}b_{Nx} + n_{Ny}b_{Ny} + n_{Nz}b_{Nz} - n_{Nx}s_{Nx} - n_{Ny}s_{Ny} - n_{Nz}s_{Nz} \end{bmatrix} \tag{5.14}$$

$$\mathbf{x} = \begin{bmatrix} \alpha & \beta & \gamma & t_x & t_y & t_z \end{bmatrix}^T \tag{5.15}$$

and

$$\mathbf{A} = \begin{bmatrix} n_{1z}s1y - n_{1y}s1z & n_{1x}s1z - n_{1z}s1x & n_{1y}s1x - n_{1x}s1y & n_{1x} & n_{1y} & n_{1z} \\ n_{2z}s2y - n_{2y}s2z & n_{2x}s2z - n_{2z}s2x & n_{2y}s2x - n_{2x}s2y & n_{2x} & n_{2y} & n_{2z} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ n_{Nz}sNy - n_{Ny}sNz & n_{Nx}sNz - n_{Nz}sNx & n_{Ny}sNx - n_{Nx}sNy & n_{Nx} & n_{Ny} & n_{Nz} \end{bmatrix} \tag{5.16}$$

The least-squares solution is

$$\mathbf{x} = \mathbf{A}^{\dagger}\mathbf{c} \tag{5.17}$$

where $\mathbf{A}^{\dagger}$ is a moore-penrose pseudoinverse of $\mathbf{A}$. The transformation is computed using (5.11) with solution $\mathbf{x}$.

Appendix C : Orientation Representation

Throughout this work, orientation is represented in different formats ranging from rotation matrices, Euler angles and Quaternion. This section will describe how to convert the transformation from one format to another.

Rotational matrix is a 3×3 matrix that can be defined as

$$\mathbf{R} = \mathbf{R}_\gamma \mathbf{R}_\beta \mathbf{R}_\alpha \tag{5.18}$$

where $\mathbf{R}_\gamma$, $\mathbf{R}_\beta$ and $\mathbf{R}_\alpha$ are the rotations about x-axis, y-axis and z-axis respectively. Note that $\mathbf{R}$ has 12 possibilities depending on the order of $\mathbf{R}_\gamma$, $\mathbf{R}_\beta$ and $\mathbf{R}_\alpha$ since matrix multiplication does not commute. These rotations are defined as

$$\mathbf{R}_\gamma = \begin{bmatrix} \cos\gamma & -\sin\gamma & 0 \\ \sin\gamma & \cos\gamma & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{5.19}$$

$$\mathbf{R}_\beta = \begin{bmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{bmatrix} \tag{5.20}$$

$$\mathbf{R}_\alpha = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha \\ 0 & \sin\alpha & \cos\alpha \end{bmatrix} \tag{5.21}$$

Simplifying (5.18), $\mathbf{R}$ is given as

$$\mathbf{R} = \mathbf{R}_\gamma \mathbf{R}_\beta \mathbf{R}_\alpha \tag{5.22}$$

$$= \begin{bmatrix} \cos\gamma\cos\beta & -\sin\gamma\cos\alpha + \cos\gamma\sin\beta\sin\alpha & \sin\gamma\sin\alpha + \cos\gamma\sin\beta\cos\alpha \\ \sin\gamma\cos\beta & \cos\gamma\cos\alpha + \sin\gamma\sin\beta\sin\alpha & -\cos\gamma\sin\alpha + \sin\gamma\sin\beta\cos\alpha \\ -\sin\beta & \cos\beta\sin\alpha & \cos\beta\cos\alpha \end{bmatrix} \tag{5.23}$$

Angles α, β and γ are the Euler angles. Assuming the same order used in (5.18), Eulers angles can be computed from rotation matrix as

$$\beta = -\arcsin(R_{31}) \tag{5.24}$$

$$\alpha = \arctan 2(R_{32}, R_{33}) \tag{5.25}$$

$$\gamma = \arctan 2(R_{12}, R_{11}) \tag{5.26}$$

The function $\arctan 2$ is similar to $\arctan$ with the difference is that $\arctan 2$ uses the signs of the variables to determine in which quadrant the angle falls. To convert back to rotation matrix from Euler angles, Euler angles α, β and γ are substituted into (5.23) .

With regard to quaternion format, it represented by four element vector as

$$\bar{q} = \begin{bmatrix} q_1 \\ q_2 \\ q_3 \\ q_4 \end{bmatrix} = \begin{bmatrix} \mathbf{q} \\ q_4 \end{bmatrix} \tag{5.27}$$

To determine the transformation in rotation matrix $\mathbf{R}$ from quaternion $\bar{q}$, lets assume that ${}^G\mathbf{P}$ is a point expressed in global {G} coordinate frame. To transform ${}^G\mathbf{P}$ from global frame to local frame {L}, it can be defined as

$${}^L\mathbf{P} = \mathbf{R}\,{}^G\mathbf{P} \tag{5.28}$$

where $\mathbf{R}$ is a 3×3 rotational matrix that transform a vector from {G} to {L} coordinate frame.

This can be expressed in quaternion as follows

$$^{L}\bar{P} = {}^{L}_{G}\bar{q} \otimes {}^{G}\bar{P} \otimes ({}^{L}_{G}\bar{q})^{-1} \tag{5.29}$$

$$^{L}\bar{P} = \begin{bmatrix} \mathbf{P} \\ 0 \end{bmatrix} \tag{5.30}$$

where ${}^{L}_{G}\bar{q}$ is the orientation of the global frame with respect to local frame and $\bar{P}$ is a quaternion representation the 3D point $\mathbf{P}$. Equation (5.29) can be simplified further to

$$^{L}\bar{P} = \begin{bmatrix} (2q_4^2 - 1)I_{3\times 3} - 2q_4\lfloor \mathbf{q}\times \rfloor + 2\mathbf{q}\mathbf{q}^T \\ 0 \end{bmatrix} \begin{bmatrix} {}^{G}\mathbf{P} \\ 0 \end{bmatrix} \tag{5.31}$$

hence the rotational matrix can be expressed as

$$\mathbf{R} = (2q_4^2 - 1)\mathbf{I}_{3\times 3} - 2q_4^2\lfloor \mathbf{q}\times \rfloor + 2\mathbf{q}\mathbf{q}^T \tag{5.32}$$

To compute quaternion from a rotation matrix $\mathbf{R}$

$$T = \text{trace}(\mathbf{R}) = R_{11} + R_{22} + R_{33} \tag{5.33}$$

$$\bar{q} = \begin{bmatrix} \sqrt{1 + 2R_{11} - T}/2 \\ (R_{12} + R_{21})/(4q_1) \\ (R_{13} + R_{31})/(4q_1) \\ (R_{23} - R_{32})/(4q_1) \end{bmatrix} = \begin{bmatrix} (R_{12} + R_{21})/(4q_2) \\ \sqrt{(1 + 2R_{22} - T)}/2 \\ (R_{23} + R_{32})/(4q_2) \\ (R_{31} - c_{13})/(4q_2) \end{bmatrix} \tag{5.34}$$

$$= \begin{bmatrix} (R_{13} + R_{31})/(4q_3) \\ (R_{23} + R_{32})/(4q_3) \\ \sqrt{1 + 2c_{33} - T}/2 \\ (R_{12} + R_{21})/(4q_3) \end{bmatrix} = \begin{bmatrix} (R_{23} - R_{32})/(4q_4) \\ (R_{31} - R_{13})/(4q_4) \\ (R_{12} - R_{21})/(4q_4) \\ \sqrt{1 + T}/2 \end{bmatrix} \tag{5.35}$$

A solution is chosen as one that corresponds to the maximum of $(|q_1|, |q_2|, |q_3|, |q4|)$ for maximum numerical accuracy. Each of the four quaternion solutions will be singular

when the pivotal element is zero.

Appendix D : Quaternion Representation

Quaternion transformation are based on the observation that two coordinate frames are related by a single rotation θ about some axis $\hat{\mathbf{k}} = \begin{bmatrix} k_x & k_y & k_z \end{bmatrix}^T$. Quaternion, $\bar{q}$ is given as

$$\bar{q} = \begin{bmatrix} k_x \sin(\theta/2) \\ k_y \sin(\theta/2) \\ k_z \sin(\theta/2) \\ \cos(\theta/2) \end{bmatrix} \tag{5.36}$$

and it can generally be defined mathematically as

$$\bar{q} = q_4 + q_1 i + q_2 i + q_3 k, \qquad \text{where} \quad i^2 = j^2 = k^2 = -1 \tag{5.37}$$

and q_1, q_2, q_3 are the complex elements of the quaternion and q_4 is real element. The following notation was used for complex number, note that it is different from the Hamilton notation.

$$ij = -k, \quad jk = -i, \quad ki = -j, \quad ji = k, \quad kj = i, \quad ik = j \tag{5.38}$$

Similarly to DCM matrix that has a determinant of 1, quaternion rotation is a unit quaternion and it should satisfy

$$|\bar{q}| = \sqrt{\bar{q}^T \bar{q}} = 1 \tag{5.39}$$

Let the imaginary part of the quaternion be represented by

$$\mathbf{q} = \begin{bmatrix} k_x \sin(\theta/2) \\ k_y \sin(\theta/2) \\ k_z \sin(\theta/2) \end{bmatrix} \tag{5.40}$$

Useful Identities and Properties

In this section some of the properties derived in [74] and [77] where $\mathbf{a}$ and $\mathbf{b}$ are used as 3×1 vectors.

1. Skew symmetric matrix

$$\lfloor \mathbf{q}\times \rfloor = \begin{bmatrix} 0 & -q_3 & q_2 \\ q_3 & 0 & -q_1 \\ -q_2 & q_1 & 0 \end{bmatrix} \tag{5.41}$$

2. Omega matrix

$$\Omega(\mathbf{q}) = \begin{bmatrix} -\lfloor \mathbf{q}\times \rfloor & \mathbf{q} \\ -\mathbf{q}^T & 0 \end{bmatrix} \tag{5.42}$$

3. Identity quaternion

$$\bar{q}_I = \begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix}^T \tag{5.43}$$

4. Inverse of a quaternion

$$\bar{q}^{-1} = \begin{bmatrix} -q_1 \\ -q_2 \\ -q_3 \\ q4 \end{bmatrix} \tag{5.44}$$

5. Anti-Commutativity of skew-symmetric matrix

$$\lfloor \mathbf{a}\times \rfloor \mathbf{b} = -\lfloor \mathbf{b}\times \rfloor \mathbf{a} \tag{5.45}$$

6. Distributivity over addition

$$\lfloor \mathbf{a}\times \rfloor + \lfloor \mathbf{b}\times \rfloor = \lfloor \mathbf{a} + \mathbf{b}\times \rfloor \tag{5.46}$$

Appendix E : Groundtruth Algorithm

This section outlines the method that can be used to obtain the groundtruth of the ego-motion of the ToF-IMU system. The groundtruth is referred to as the estimation of the robot's trajectory in which the accuracy is known and the estimated trajectory is close to the actual trajectory. This work was done during the course of the masters before an artificial mine was acquired.

Setup

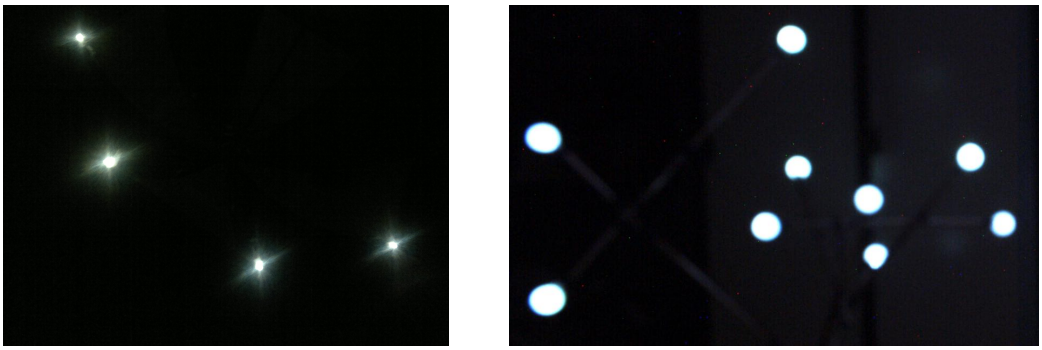
The following equipments were used:

- Prosilica camera mounted rigidly with the ToF as shown in Figure 5.2(a).
- A structure with the Light-Emitting-Diodes (LEDs) that are used as features is shown in Figure 5.2(b).

The testing of the system was done in a dark environment mimicking the underground mine environment. The LED structure will have to be in the field of view of the Prosilica camera at all times during the collection of data. Note that LEDs are inside ping-pong balls so that they can produce a more constant shape for LED extraction. See Figure 5.3 for the comparison of LED with and without a ping-pong ball. The section below describes the algorithm for obtaining the groundtruth.

(a) (b)

Figure 5.2: (a) Shows the ToF camera rigidly mounted with the Prosilica camera and (b) shows the LED structure used to track the camera



(a) (b)

Figure 5.3: (a) shows an image of LEDs before they are inside a ping pong balls and (b) shows the LEDs inside a ping pong ball

Groundtruth Algorithm

This algorithm can be used to provide groundtruth during the collection of data in the mine. Before data can be collected, calibration on the camera must be performed to compute the intrinsic calibration matrix and distortion coefficient. Calibration matrix is used in the estimation of the camera orientation and position from 3D points to 2D image points correspondence. Distortion coefficient is used to undistort the images for higher accuracy. Open source Computer Vision[1] (OpenCV) is used for calibration together with a checker-board.

[1] www.opencv.com

The LED structure has six LEDs attached to the tip as shown in Figure 5.2(b). The distances between the LEDs are measured using Vicon system and are used to build a 3D model of the LED structure. One LED is randomly selected as the origin and the others are described with respect to it.

The following subsection will explain the process of extracting a pixel which represent the centroid of the LED in the image stream. At this point it is assumed that the images have been undistorted and they have been converted to gray-scale images.

LED Extraction

Since the LED structure will always be in the field of view of the camera during data collection and the experiment is performed in a dark environment, the pixel with the highest value are the ones containing LEDs. Non-Maxima suppression (NMS) algorithm [78] is used to find the pixel with highest value. A MATLAB implementation of NMS algorithm by Peter Kovesi can be found in [79]. The algorithm takes threshold intensity value, radius of the region to look for the maximum and the image and output positions of the pixels of maximum intensities.

The next step in extracting LEDs centroid is to cluster all the pixels that belong to the same LED in one cluster. Normalised cut [80] clustering algorithm was found to perform better compared to K-means. A Matlab implementation that uses mex files to speed up the normalised cut[1] algorithm was used. The code takes in pixel locations, which is the output of the NMS algorithm and the number of cluster, of which in this case is the maximum number of LEDs.

In case where less than six LEDs are in view, normalised cut tend to divide one of the LED into two LEDs. This is handled by applying a pixel distance threshold between the LEDs. If the distance between two LEDs is less than some predefined threshold, it is considered as one LED by averaging them. To make sure that the centroid actually represent the LED, the intensity value must be greater or equal to the threshold used in the NMS algorithm. This is also used to verify the clustering algorithm, if pixels from different LEDs are clustered as one cluster, the average position will be in between and the intensity is likely to be less than the threshold.

[1] http://www.cis.upenn.edu/∼jshi/software

The following section will describe how to estimate the camera pose using the 3D coordinates of the LED's and their corresponding 2D points on the image.

Estimation of the Camera Pose

The problem of estimating the camera pose is called perspective n points (PnP) and can be stated as:
Given an intrinsic calibration matrix and a set of corresponding points between 3D points and the 2D projection of those points, find the pose of the camera.
This problem has been solved in [81]. A Matlab implementation is also available [1]. It takes in a list of the 3D coordinates, the corresponding 2D coordinates in the image plane and the intrinsic calibration matrix as an input and output the transformation $(\mathbf{R}, \mathbf{t})$ of the camera with respect to the LED structure reference frame and the coordinates of the LEDs with respect to the camera coordinates system.

The correspondence between the 3D points and the 2D points in the image is unknown. If a lot of points were used, algorithms such as Random Sample and Consensus (RanSAC) algorithm [30] could be used. This problem was resolved by applying PnP algorithm on all possible correspondences and use the computed transformation to calculate the 2D projection of 3D points. The sum of square errors between the computed 2D projection and the 2D position computed by NMS algorithm is calculated. The correspondence that has the lowest error is considered as the correct correspondence. Since this algorithm is not applied in real-time and there are few points, it is feasible. This can be computed by letting $\mathbf{R}$ and $\mathbf{t}$ be the rotation and translation from the LED structure frame of reference to camera frame of reference respectively from the output of the PnP algorithm and $\mathbf{K}$ be the intrinsic calibration matrix. The matrix $\mathbf{P}$ that projects 3D point into 2D image plane can be computed as

$$\mathbf{P} = \mathbf{K} * [\mathbf{R} \mid \mathbf{t}] \tag{5.47}$$

[1] http://cvlab.epfl.ch/software/EPnP/

and the 2D point is computed as

$$\mathbf{X} = \mathbf{P} * \mathbf{X}_{3D} \tag{5.48}$$

$$\mathbf{X}_{2D} = \begin{bmatrix} \mathbf{X}(1)/\mathbf{X}(3) \\ \mathbf{X}(2)/\mathbf{X}(3) \end{bmatrix} \tag{5.49}$$

where $\mathbf{X}_{3D}$ is a 3D point and X2D is the 2D projection of the 3D point. A full derivation can be found in [82]

To compute all the possible correspondence, the combination of LEDs that are in field of view during capture and the total number of LEDs is computed. The permutation of each combination is then computed and PnP algorithm [81] is applied on all the correspondences. The pseudo code of groundtruth estimation algorithm is in Algorithm 7.

Algorithm 7 Groundtruth algorithm

1: Build the LED 3D structure
2: **for** Until images are finished **do**
3: Read image
4: Undistort image using distortion coefficient
5: Change the image to gray-scale
6: Compute centroid of the LED using NMS
7: Compute all possible corresponding between the LEDs centroid 2D points and 3D points from the model
8: Apply PnP algorithm on all possible correspondences and save $\mathbf{R}$ and $\mathbf{t}$, as well as the error between the LED centroid points computed by NMS and the ones computed by projecting 3D points using $\mathbf{R}$ and $\mathbf{t}$ from PnP algorithm.
9: The correspondences that has the smallest error are considered as the correct matches.
10: Transformation $(\mathbf{R}, \mathbf{t})$ corresponding to the smallest error is used to compute ego-motion of the camera.
11: **end for**

Results

The experiment was carried out in MIAS laboratory. Images from the Prosilica camera and Vicon position of the camera were recorded. The time stamp of which each image

and Vicon position were recorded as well such that comparison can be done easily. Figure 5.4 shows the position estimation of the camera using the Vicon system (shown in red) and estimation using PnP algorithm (shown in blue). In this experiment the camera was moving towards the LED structure for approximately 2 meters.

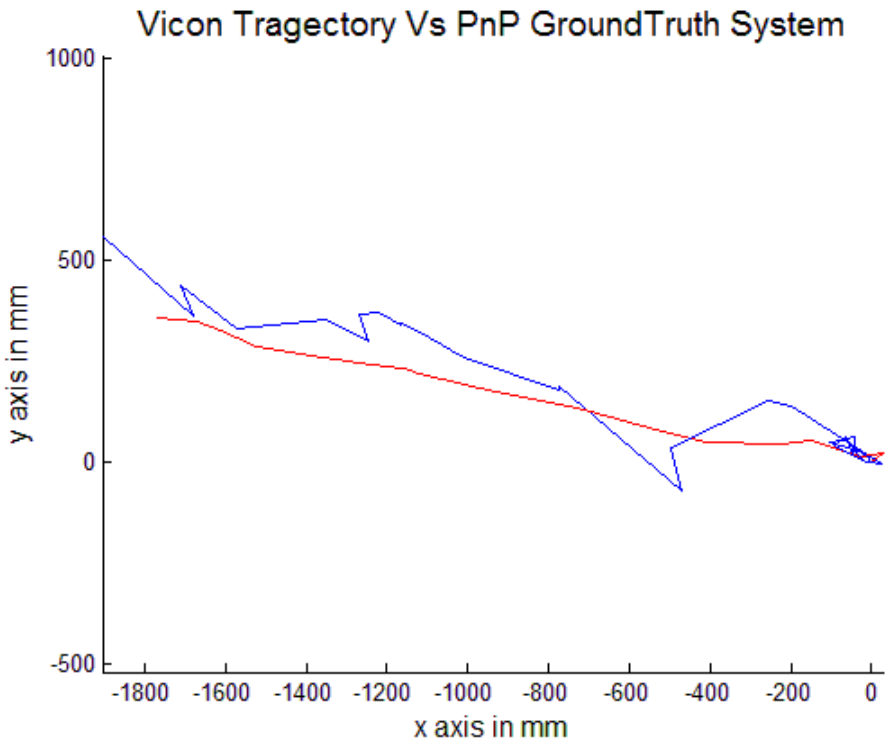


Figure 5.4: The comparison of motion estimation using Vicon system (red) and PnP system (blue)

The pose of the camera provided by the Vicon system is taken as the actual pose of the camera since it has an accuracy of less than a millimeter. The error is calculated by finding the difference between the Vicon pose and the PnP pose at the same time. The average error is less than 15 cm in translation and the rotation error is less than 14 degree.

The system still need improvement in terms of finding LED optimal structure and updating the code to be more robust. After the artificial mine was acquired, this part was abandoned since it does not fall under the scope of the project.

References

[1] I. S. 647-2006, *IEEE Standard Specification Format Guide and Test Procedure for Single-axis Laser Gyros, Annex c.* IEEE Std, 2006. xi, 71, 72

[2] A. Nüchter, K. Lingemann, J. Hertzberg, and H. Surmann, "6d slam-3d mapping outdoor environments: Research articles," *J. Field Robot.*, vol. 24, pp. 699–722, Aug 2007. 2

[3] P. J. Besl and N. D. McKay, "A method for registration of 3-d shapes," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 14, pp. 239–256, Feb. 1992. 2, 3, 12, 13, 16, 17, 21, 33

[4] Y. Chen and G. Medioni, "Object modelling by registration of multiple range images," *Image Vision Comput.*, vol. 10, pp. 145–155, Apr. 1992. 3, 12, 15, 16, 21, 33

[5] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vision*, vol. 60, pp. 91–110, Nov. 2004. 3, 13, 22, 23, 37

[6] O. J. Woodman, "An introduction to inertial navigation," Tech. Rep. UCAM-CL-TR-696, University of Cambridge, Computer Laboratory, Aug. 2007. 3, 73

[7] A. H. Jazwinski, *Stochastic Processes and Filtering Theory Mathematics in Science and engineering*, vol. 64. Academic Press, 1970. 3, 19, 20

[8] Mesa Imaging, *SR4000 User Manual: Mesa Imaging version 2.0 2010.* 6, 11, 12, 26, 77

[9] R. Lange, *3D Time-of-Flight Distance Measurement with Custom solid-State Image Sensors in CMOS/CDD-Technology.* PhD thesis, Department of Electrical Engineering and Computer Science at University of Siegen, 2000. 7, 9, 85

[10] D. Piatti and F. Rinaudo, "Sr-4000 and camcube3.0 time of flight (tof) cameras: Tests and comparison," *Remote Sensing*, vol. 4, no. 4, pp. 1069–1089, 2012. 7

[11] S. May, D. Droeschel, D. Holz, S. Fuchs, E. Malis, A. Nüchter, and J. Hertzberg, "Three-dimensional mapping with time-of-flight cameras.," *J. Field Robotics*, vol. 26, no. 11-12, pp. 934–965, 2009. 7, 10, 11, 13, 14, 17, 21, 27, 69

[12] S. Fuchs and G. Hirzinger, "Extrinsic and depth calibration of tof-cameras," in *2008 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2008), 24-26 June 2008, Anchorage, Alaska, USA*, IEEE Computer Society, 2008. 7, 8, 13

[13] S. Fuchs and S. May, "Calibration and registration for precise surface reconstruction with time of flight cameras," *Int. J. Intell. Syst. Technol. Appl.*, vol. 5, pp. 274–284, Nov. 2008. 7, 8, 13

[14] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, pp. 1330–1334, 1998. 8

[15] T. Kahlmann, F. Remondino, and H. Ingensand, "Calibration for increased accuracy of the range imaging camera SwissRanger," in *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences,* (Isprs, ed.), (Dresden, Germany), 2006. 8, 11, 84

[16] M. Lindner and A. Kolb, "Lateral and depth calibration of PMD-distance sensors.," in *ISVC (2)*, pp. 524–533, 2006. 8

[17] D. Piatti, *Time-of-Flight camera:tests, calibration and multi-frame registration for automatic 3D object reconstruction.* PhD thesis, Politecnico Di Torino : Dpctoral school of Environment and Territory XXII cycle, 2011. 8, 10, 11, 14, 27, 68

[18] S. Oprisescu, D. Falie, M. Ciuc, and V. Buzuloiu, "Measurements with tof cameras and their necessary corrections," in *2007 International Symposium on Signals, Circuits and Systems (ISSCS),*, vol. 1, pp. 1–4, 2007. 11

[19] D. Droeschel, D. Holz, and S. Behnke, "Probabilistic phase unwrapping for time-of-flight cameras," in *ISR/ROBOTIK*, pp. 1–7, 2010. 11, 30, 31

[20] D. Droeschel, D. Holz, and S. Behnke, "Multi-frequency phase unwrapping for time-of-flight cameras," in *IROS*, pp. 1463–1469, 2010. 11, 30

[21] M. Reynolds, J. Dobos, L. Peel, T. Weyrich, and G. J. Brostow, "Capturing time-of-flight data with confidence," in *Proceedings of the 2011 IEEE Conference on Computer Vision and Pattern Recognition*, CVPR '11, (Washington, DC, USA), pp. 945–952, IEEE Computer Society, 2011. 11

[22] A. Nüchter, H. Surmann, K. Lingemann, J. Hertzberg, and S. Thrun, "6d slam with an application in autonomous mine mapping," in *In Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 1998–2003, 2004. 12, 17, 23

[23] S. Rusinkiewicz and M. Levoy, "Efficient variants of the ICP algorithm," in *Third International Conference on 3D Digital Imaging and Modeling (3DIM)*, June 2001. 12, 34

[24] F. Pomerleau, S. Magnenat, F. Colas, M. Liu, and R. Siegwart, "Tracking a depth camera: Parameter exploration for fast icp," in *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2011. 12

[25] J. Weingarten, G. Gruener, and R. Siegwart, "A state-of-the-art 3d sensor for robot navigation," in *Intelligent Robots and Systems, 2004. (IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on*, vol. 3, pp. 2155–2160 vol.3, 2004. 12

[26] K. Ohno, T. Nomura, and S. Tadokoro, "Real-time robot trajectory estimation and 3d map construction using 3d camera," in *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2006, October 9-15, 2006, Beijing, China*, pp. 5279–5285, IEEE, 2006. 12, 21, 34, 68, 70

[27] A. Nüchter, *3D Robotic Mapping: The Simultaneous Localization and Mapping Problem with Six Degrees of Freedom*. Springer Tracts in Advanced Robotics, Springer, 2009. 13, 87

[28] M. S. Arulampalam, S. Maskell, and N. Gordon, "A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking," *IEEE TRANSACTIONS ON SIGNAL PROCESSING*, vol. 50, pp. 174–188, 2002. 13

[29] C. Tomasi and T. Kanade, "Detection and tracking of point features," tech. rep., International Journal of Computer Vision, 1991. 13, 23

[30] M. A. Fischler and R. C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981. 13, 15, 37, 100

[31] E. Malis, "An efficient unified approach to direct visual tracking of rigid and deformable surfaces," in *IROS*, pp. 2729–2734, 2007. 13

[32] M. Ezio, "Esm visual tracking web site," Feb. 2011. 13

[33] H. Bay, T. Tuytelaars, and L. V. Gool, "Surf: Speeded up robust features," in *In ECCV*, pp. 404–417, 2006. 14, 15, 23

[34] P. Rousseeuw and A. Leroy, *Robust Regression and Outlier Detection*. Wiley Series in Probability and Statistics, Wiley, 2003. 14

[35] J. J. Wang, G. Hu, S. Huang, and G. Dissanayake, "3d landmarks extraction from a range iimage data for slam," in *Australasian Conference on Robotics and Automation (ACRA)*, 2009. 15

[36] K. S. Arun, T. S. Huang, and S. D. Blostein, "Least-squares fitting of two 3-d point sets," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 9, pp. 698–700, May 1987. 15, 22, 36, 37, 87

[37] N. J. Mitra, A. Nguyen, and L. Guibas, "Estimating surface normals in noisy point cloud data," in *special issue of International Journal of Computational Geometry and Applications*, vol. 14, pp. 261–276, 2004. 15, 36

[38] H. Badino, D. Huber, Y. Park, and T. Kanade, "Fast and accurate computation of surface normals from range images," in *International Conference on Robotics and Automation (ICRA)*, no. CMU-RI-TR-, May 2011. 15

[39] J. Poppinga, N. Vaskevicius, A. Birk, and K. Pathak, "Fast plane detection and polygonalization in noisy 3d range images," in *IROS*, pp. 3378–3383, 2008. 15, 34

[40] K. Pathak, A. Birk, N. Vaškevičius, and J. Poppinga, "Fast registration based on noisy planes with unknown correspondences for 3-d mapping," *IEEE Transactions on Robotics*, vol. 26, pp. 424–441, June 2010. 15, 16, 23

[41] S. Wang and H. Yu, "A variational approach for ego-motion estimation and segmentation based on 3d tof camera," in *4th International Congress on Image and Signal Processing (CISP)*, vol. 3, pp. 1160–1164, 2011. 15, 16

[42] I. Villaverde and M. Graña, "Neuro-evolutionary mobile robot egomotion estimation with a 3d tof camera," *Neural Computing and Applications*, vol. 20, no. 3, pp. 345–354, 2011. 15, 17

[43] M. D. Shuster, "The Generalized Wahba Problem," *The Jornal of the Astronautical Sciences*, vol. 54, no. 2, pp. 245–259, 2006. 16

[44] M. Magnusson, A. Lilienthal, and T. Duckett, "Scan registration for autonomous mining vehicles using 3d-ndt," *Journal of Field Robotics*, pp. 803–827, 2007. 16, 17, 18, 34

[45] T. Martinetz, S. Berkovich, and K. Schulten, ""Neural-gas" Network for Vector Quantization and its Application to Time-Series Prediction," *IEEE-Transactions on Neural Networks*, vol. 4, no. 4, pp. 558–569, 1993. 17

[46] T. Martinetz and K. Schulten, "A "Neural-Gas" Network Learns Topologies," *Artificial Neural Networks*, vol. I, pp. 397–402, 1991. 17

[47] D. Matomela, "Rising sa mines deaths need urgent attention." http://www.iol.co.za/business/rising-sa-mine-deaths-need-urgent-attention-1.1055349, Date accessed: April 2011. 17

[48] S. Schultz, "Two miles underground." Online - http://www.princeton.edu/pr/pwb/99/1213/microbe.shtml, Date Accessed: December 1999. 17

[49] "Annual report 2010/2011." http://www.info.gov.za/view/DownloadFileAction?id=152675, Date accessed: December 2011. 17

[50] P. Biber and W. Staßer, "The normal distributions transform: A new approach to laser scan matching," in *Intl. Conference on Intelligent Robots and System (IROS)*, 2003. 18, 23

[51] P. Corke, J. Lobo, and J. Dias, "An introduction to inertial and visual sensing," *The International Journal of Robotics*, vol. 26, pp. 519–535, 2007. 18, 24

[52] H. Durrant-Whyte and T. Henderson, *Springer Handbook of Robotics*, ch. Multisensor Data Fusion, pp. 585–610. Springer, 2008. 19

[53] M. S. Arulampalam, S. Maskell, and N. Gordon, "A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking," *IEEE TRANSACTIONS ON SIGNAL PROCESSING*, vol. 50, pp. 174–188, 2002. 19

[54] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Transactions of the ASME–Journal of Basic Engineering*, vol. 82, pp. 35–45, 1960. 19

[55] P. Maybeck, *Stochastic Models, Estimation and Control Vol 2.* No. v. 2 in Mathematics in Science and Engineering, Elsevier Science, 1982. 19

[56] A. Johnson, R. Willson, J. Goguen, J. Alex, and D. Meller, "Field testing of the mars exploration rovers descent image motion estimation system," in *Proc. IEEE Int. Conf. Robot. Autom*, pp. 4463–4469, 2005. 19

[57] J. Lobo and J. Dias, "Relative pose calibration between visual and inertial sensors," *I. J. Robotic Res.*, vol. 26, no. 6, pp. 561–575, 2007. 20

[58] R. Y. Tsai and R. K. Lenz, "A new technique for fully autonomous and efficient 3d robotics hand-eye calibration," in *Proceedings of the 4th international symposium on Robotics Research*, (Cambridge, MA, USA), pp. 287–297, MIT Press, 1988. 20

[59] B. K. P. Horn, "Closed-form solution of absolute orientation using unit quaternions," *Journal of the Optical Society of America A*, vol. 4, no. 4, pp. 629–642, 1987. 20

[60] F. M. Mirzaei and S. I. Roumeliotis, "A kalman filter-based algorithm for imu-camera calibration," in *IROS*, pp. 2427–2434, 2007. 20, 22, 23, 43, 45, 76

[61] J. D. Hol, T. B. Schön, and F. Gustafsson, "Modeling and calibration of inertial and vision sensors," *I. J. Robotic Res.*, vol. 29, no. 2-3, pp. 231–244, 2010. 20, 21, 23

[62] S. Graebe, *Theory and Implementation of Gray Box Identification.* PhD thesis, Royal Institute of Technology, 1990. 21

[63] L. Ljung, *System identification (2nd ed.): theory for the user.* Upper Saddle River, NJ, USA: Prentice Hall PTR, 1999. 21

[64] D. Droeschel, S. May, D. Holz, and S. Behnke, "Fusing time-of-flight cameras and inertial measurement units for ego-motion estimation," *AUTOMATIKA*, vol. 52, no. 3, pp. 189–198, 2011. 21, 22, 23, 79

[65] J. A. Hesch, F. M. Mirzaei, G. L. Mariottini, and S. I. Roumeliotis, "A 3d pose estimator for the visually impaired," in *IROS*, pp. 2716–2723, 2009. 22

[66] J. D. Hol, T. B. Schn, H. Luinge, P. J. Slycke, and F. Gustafsson, "Robust real-time tracking by fusing measurements from inertial and vision sensors.," *J. Real-Time Image Processing*, vol. 2, no. 2-3, pp. 149–160, 2007. 24

[67] L. Armesto, J. Tornero, and M. Vincze, "Fast ego-motion estimation with multi-rate fusion of inertial and vision," *I. J. Robotic Res.*, vol. 26, no. 6, pp. 577–589, 2007. 24

[68] A. I. Mourikis and S. I. Roumeliotis, "A multi-state constraint kalman filter for vision-aided inertial navigation," in *in Proc. IEEE Int. Conf. on Robotics and Automation*, pp. 10–14, 2007. 24

[69] A. Vedaldi and B. Fulkerson, "VLFeat: An open and portable library of computer vision algorithms." `http://www.vlfeat.org/`, Date Accessed: March 2012. 35, 37

[70] K.-L. Low, "Linear least-squares optimization for point-to-plane icp surface registraion," tech. rep., University of North Carolina at Chapel Hill, Feb 2004. 36

[71] R. Siegwart and I. Nourbakhsh, *Introduction to Autonomous Mobile Robots*. MIT Press, bradford book ed., 2004. 37

[72] MicroStrain, *3DM-GX3 Data Communications Protocol*. 39

[73] D. Titterton, J. Weston, and I. of Electrical Engineers, *Strapdown Inertial Navigation Technology, 2nd Edition*. Institution of Engineering and Technology, 2004. 40, 42

[74] N. Tranwny and S. Roumeliotis, "Indirect kalman filter for 3d pose estimation," tech. rep., University fo Minnesota, Dept. of Computer Science & Engineering, 2005. 43, 96

[75] MicroStrain, *3DM-GX3 Data Communication Protocol*. 71

[76] T. Oggier, M. Lehmann, R. Kaufmann, M. Schweizer, M. Richter, P. Metzler, G. Lang, F. Lustenberger, and N. Blanc, "An all-solid-state optical range camera for 3d real-time imaging with sub-centimeter depth resolution (swissranger)," in *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, vol. 5249, pp. 534–545, Feb. 2004. 85

[77] W. G. Breckenridge, "Quaternions - proposed standard conventions," tech. rep., Interoffice Memorandumiom, 1999. 96

[78] A. Neubeck and L. Van Gool, "Efficient non-maximum suppression," in *Proceedings of the 18th International Conference on Pattern Recognition - Volume 03*, ICPR '06, (Washington, DC, USA), pp. 850–855, IEEE Computer Society, 2006. 99

[79] P. D. Kovesi, "MATLAB and Octave functions for computer vision and image processing." Centre for Exploration Targeting, School of Earth and Environment, The University of Western Australia, Date Accessed: July 2012. Available from: http://www.csse.uwa.edu.au/~pk/research/matlabfns/. 99

[80] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, pp. 888–905, 1997. 99

[81] V. Lepetit, F. Moreno-Noguer, and P. Fua, "Epnp: An accurate o(n) solution to the pnp problem," *Int. J. Comput. Vision*, vol. 81, pp. 155–166, Feb. 2009. 100, 101

[82] R. I. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521540518, second ed., 2004. 101