AN INVESTIGATION INTO TRAJECTORY ESTIMATION IN UNDERGROUND MINING ENVIRONMENTS USING A TIME-OF-FLIGHT CAMERA AND AN INERTIAL MEASUREMENT UNIT

T.T. Ratshidaho[1]; J.R. Tapamo[2*], J. Claassens[3] & N. Govender[4]

[1,3,4]Mobile Intelligence Autonomous Systems
Council for Scientific and Industrial Research Pretoria, South Africa
[1]tratshidaho@csir.co.za, [4]ngovender@csir.co.za

[2]School of Engineering, University of Kwazulu-Natal, South Africa
tapamoj@ukzn.ac.za

ABSTRACT

One of the most important and challenging tasks for mobile robots that navigate autonomously is localisation – 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 an unknown environment is done by tracking the trajectory of the robot on the basis of the initial pose. Trajectory estimation becomes a challenge if the robot is operating in an unknown environment that has a scarcity of landmarks, is GPS-denied, has very little or no illumination, and is slippery – such as in underground mines. This paper attempts to solve the problem of estimating a robot's trajectory in underground mining environments using a time-of-flight (ToF) camera and an inertial measurement unit (IMU). In the past, this problem has been addressed by using a 3D laser scanner; but these are expensive and consume a lot of power, even though they have high measurement accuracy and a wide field of view. Here, trajectory estimation is accomplished by the fusion of ego-motion provided by the ToF camera with measurement data provided by a low cost IMU. The fusion is performed using the Kalman filter algorithm on a mobile robot moving on a 2D planar surface. The results show a significant improvement on the trajectory estimation. A Vicon system is used to provide groundtruth for the trajectory estimation. Trajectory estimation only using the ToF camera is prone to errors, especially when the robot is rotating; but the fused trajectory estimation algorithm is able to estimate accurate ego-motion even when the robot is rotating.

OPSOMMING

Een van die belangrikste en uitdagendste take vir mobiele robotte om selfstandig te kan navigeer is lokalisering. Lokalisering van 'n robot in 'n onbekende omgewing word gedoen deur die volg van die trajek van die robot (die aanvanklike posisie moet bekend wees). Trajekskatting raak uitdagend as die robot moet funksioneer in 'n onbekende omgewing met 'n tekort aan landmerke, geen GPS opvangs, baie swak of geen verligting en 'n gladde oppervlak – soos in ondergrondse myne. Hierdie artikel poog om die probleem van die skatting van 'n robot se trajek in ondergrondse mynbou omgewing met 'n tyd-van-vlug kamera en traagheid meet eenheid op te los. In die verlede is hierdie probleem aangespreek deur die gebruik van 'n 3D laserskandeerder. 3D laserskandeerders is duur en gebruik baie krag, al is hulle baie akkuraat met 'n wye veld van sig. In hierdie artikel is trajek skatting gedoen deur die samesmelting van die ego-beweging, gekry van die TVV kamera, en die meting data voorsien deur 'n goedkoop TME. Die samesmelting is uitgevoer deur gebruik te maak van die Kalmanfilter algoritme op 'n mobiele robot wat in 'n 2D plat vlak beweeg. Die resultate toon 'n verbetering op die trajekskatting. 'n Vicon stelsel word gebruik om die begin posisie te verskaf vir die trajekskatting. Trajekskatting slegs met die behulp van die TVV kamera is geneig tot foute, veral wanneer die robot draai. Die trajekskatting algoritme is in staat om akkuraat ego-beweging te skat, selfs wanneer die robot draai.

* Corresponding author

1 INTRODUCTION

There has been a lot of interest recently in autonomous mobile robots, especially those being used in dangerous work environments. While the mobile robot is in operation, it must be able to navigate within the environment. To do so autonomously, one of the most important and challenging tasks is localisation – the process whereby a robot locates itself within a map of a known environment, or in relation to a known starting point within an unknown environment. Also, for a mobile robot to perform any useful task, it is important to know the pose (position and orientation) of the robot at any point during a specific task.

The problem of localisation becomes a major challenge if the robot is operating in unknown environments that have few landmarks, are GPS-denied, have very little or no illumination, and are slippery – such as underground mine stopes. In such an environment, the robot is localised by tracking the trajectory of the robot, given the initial pose.

Trajectory estimation in environments like underground mines has been successfully achieved by using three-dimensional (3D) laser scanners [1] [2] [3]. These provide high accuracy 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) [4]. But these sensors are very expensive and consume a lot of power.

In this paper, the estimation of a robot's trajectory is performed by fusing a time-of-flight (ToF) camera's ego-motion with data from a low-cost inertial measurement unit (IMU). The ToF camera is a compact sensor that provides both range and amplitude images, and is an attractive option because it can operate at video frame rate, and has a working range of 10 metres, compared with the Microsoft® Kinect sensor, which has a working range of 3 metres.

The IMU consists of three accelerometers and three gyroscopes that are orthogonally mounted such that measurements are on the x, y and z axes. By integrating linear acceleration and angular velocity from the accelerometers and gyroscopes respectively, the inertial navigation system (INS) is able to track the position, velocity, and orientation of the robot [5]. Fusion was performed using the Kalman filter algorithm [6], the IMU measurements as inputs, and the ToF camera ego-motion estimates as observations.

The objectives for the paper are the following:

- To investigate different methods for improving the quality of the ToF camera images;
- To investigate the different algorithms for filtering ToF camera images and the implementation of an algorithm;
- To investigate various ways of tracking the camera pose using ToF images, and to select the most accurate and suitable ones for underground mine stopes;
- To develop an algorithm that fuses IMU data and ToF ego-motion to estimate a more accurate trajectory;
- To evaluate the fusion method using the Vicon motion capture system (a system that tracks an object's pose with high accuracy).

The rest of the paper is organised as follows: Section 2 gives the background and related work on ToF camera ego-motion and the fusion of the camera's ego-motion with IMU data; Section 3 describes the design and analysis of the full system; Section 4 gives the experimental results and discussions; and Section 5 presents the conclusion and discusses future work.

2 BACKGROUND AND RELATED WORK

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. The 3D information is obtained by using the phase-shift principle [7]. 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 travelled by the light. Two ToF cameras are generally used in robotics: the camCube from pmdTechnologies®[†], and the Swiss Ranger from Mesa Imaging[‡]. Our experiments used the Swiss Ranger ToF cameras. Even though both cameras use the same principle to measure distance, their sensors tend to have different characteristics, based on the number and arrangement of the modulated NIR light sources [8]. The accuracy of these sensors is limited by errors that are caused by the measurement principle, the architecture of the sensors, and the environment (background light and reflectivity of objects). These errors can be divided into systematic errors and non-systematic errors [9] [10]. Systematic errors are those errors that behave in a systematic way, making them easy to model and correct through calibration. Calibration methods to remove systematic errors have been proposed in [9] [11] [12] [13]. Non-systematic or random errors are highly dependent on the scene configuration, implying that they cannot be corrected during calibration, but have to be detected during the application. Most of the dominant non-systematic errors are jump edge points, phase wrapping, or aliasing and noise.

Ego-motion refers to the process of estimating the trajectory of a camera that is used in localising a mobile robot. The camera's ego-motion is estimated by registering two consecutive range images to find a rigid transformation (translation and rotation). Mobile robot trajectory estimation has successfully been applied on 3D laser scanner data collected in underground environments [1] [2].

To our knowledge, no work has been done on estimating a mobile robot trajectory operating in underground mining environments using a ToF camera. These cameras are suitable because of their low power consumption and high frame rate. The widely-used algorithm to register two 3D range images is iterative closest point (ICP) [4] [14]. There are two forms of ICP: point-to-point ICP [4] and point-to-plane ICP [14]. A point-to-point ICP was applied in [1] and [2] to 3D laser scanner range images in underground and outdoor environments for map-building and localisation respectively. A point-to-point ICP finds the rigid transformation by iteratively minimising the root mean square (RMS) error between corresponding points, while point-to-plane ICP finds the rigid transformation by minimising the RMS error between points in one range image and the tangent plane in the other range image. In [15], a more thorough study is presented of the ICP algorithm with its variants. The ICP algorithm is divided into six stages, and different variants were compared in terms of convergence speed and accuracy.

In general, point-to-point ICP is faster than point-to-plane ICP; but point-to-plane converges in few iterations. ICP tends to be 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 [16]. The latter is possible with ToF cameras, since they have high frame rates.

The first application of a ToF camera in mobile robots trajectory estimation was performed by Ohno et al. [17], using a second generation Swiss Ranger ToF camera (SR2000). A modified point-to-point ICP algorithm with K-Dimensional (KD)-tree for finding corresponding points was used. The algorithm was evaluated using a motion capture system with an accuracy of 10mm. For a simple one degree of freedom motion, the average error was 17% for translation and 15% for rotation. The algorithm produced erroneous results while the robot was moving in a rubble-filled environment. The errors were mostly caused

[†] http://www.pmdtec.com/
[‡] http://www.mesa-imaging.ch/

by rotational motion; but other causes were the uncalibrated distance measurements of the ToF camera, and the fact that edge points were used as source points for estimating the transformation. Edge points are mostly classified as jump edge points, which is erroneous.

A thorough study of the Swiss Ranger ToF camera can be found in [10], where a SR3000 ToF camera was used for map-building and trajectory estimation. To get accurate ToF images, photogrammetric and distance calibration was performed as in [9] [18]. A point-to-point ICP algorithm was modified by introducing frustrum culling, KD-trees, and a jump edge filter. Frustrum culling was used to find mismatches and the non-overlaps between two range images. The frustrum culling method is used to find points that are in the current range image but were not present in the previous image, and vice versa.

The ToF camera also produces amplitude/intensity images that are similar to grey-scale images. Two-point feature extraction algorithms were implemented to track points on the amplitude images, and to use their corresponding 3D points to estimate the transformation using a least squares method proposed by Arun et al. [19]. The point features algorithms implemented by May et al. [10] are scale invariant feature transform (SIFT) [20] and Kanade Lucas Tomasi (KLT) [21]. The limiting factor with point features is that the ToF camera has a low resolution, which means that few features are detected; and the situation is substantially worse if the camera operates in an environment with few features, such as a mine.

May et al. [10] also implemented a direct registration efficient second order method (ESM) visual tracking algorithm [22] [23] for trajectory estimation. Instead of tracking point features, ESM tracks both rigid and non-rigid bodies in the images. A ToF camera was mounted on an industrial robot arm with high accuracy that was used as groundtruth. A point-to-point ICP had a higher accuracy than all the other algorithms that were implemented. Both SIFT and KLT were faster than ICP. This was because no matching of 3D points was necessary, as the exact point of correspondence was known.

Dario [13] proposed a multi-frame registration algorithm that averages 30 images per pose, to obtain more accurate distance measurements. The speeded up robust features (SURF) [24] algorithm was used to track point features in the amplitude images, and the 3D points were used to compute the transformation. To make the algorithm more robust, a least median square estimator [25] was used to estimate the transformation, and jump edges were also rejected by a pixel removal algorithm [13]. This algorithm is not applicable to mobile robots, since it requires the camera to be stationary to capture 30 images.

Alternative methods to ICP for trajectory estimation using 3D range images have been proposed in [26] [27] [28]. Pathak et al. [26] used a method called 'minimally uncertain maximum consensus', which finds planes in the range images, iteratively finds the correspondence of these planes, and computes the transformation using the Davenport q-method [29] for rotation and the least squares method for translation. This algorithm requires the environment to have a large number of planes – and at least three non-parallel planes – to compute the translation.

The algorithm proposed in [28] simultaneously estimates the ego-motion and segments the images by curve fitting. The algorithm minimises the energy function, which consists of ego-motion and curve fitting parameters using the variational approach. The algorithm is best suited to an environment where there are moving objects.

Villaverde and Graña [27] proposed an algorithm for ego-motion estimation using artificial neural networks. Their algorithm has three stages. The first is pre-processing: range images are filtered for distances using phase-wrap. In the second stage, neural gas networks [30] [31] perform vector quantisation on the filtered range images to produce codebooks. In the third stage, these are used to compute the transformation using an evolutional strategy module.

2.2 Fusion of ToF camera ego-motion and IMU data

Trajectory estimation using the ToF camera's ego-motion is prone to errors, especially when the robot is turning. The ToF ego-motion needs to be fused with information from another sensor to increase accuracy and reduce errors. IMU – which measures linear acceleration and angular velocity to track the pose using the INS algorithm – is mostly used to enhance the camera's ego-motion estimation. INS has rapid errors, mainly caused by the noise and biases in the IMU measurements, making the INS error unbounded if it is used alone [5].

Data fusion is the process of combining measurements from different sensors in order to estimate a robust and complete description of the state of interest. It has a wide range of applications in robotics. A good introduction to the fusion of visual and inertial sensors in the application of localising a mobile robot by estimating the trajectory can be found in [32]. The camera and the inertial sensor complement each other in estimating the mobile robot trajectory. The main advantage is that they do not require an external infrastructure, meaning that they can operate in unknown environments.

Most methods used in data fusion make use of stochastic techniques, with the Kalman filter [6] and the particle filter [33] 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 particle filter is a recursive implementation of the Monte Carlo method, which describes state estimates as the probability distribution, using a set of weighed samples of an underlying state space. Particle filters are suitable for problems where dynamic processes and measurement models are non-linear and the noise is not Gaussian [34]. For our application, the Kalman filter is used because of its non-linear system, and because the noise can be modelled as Gaussian.

Even though the ToF camera ego-motion estimation has been studied for at least a decade, there is limited research on the topic of fusing it with an IMU sensor data in order to increase the accuracy of the trajectory estimation. To our knowledge, Droeschel et al. [35] are the only researchers who have fused ToF camera data with IMU data, using an SR3000 ToF camera with Xsens MTi IMU. The Kalman filter was used to fuse the ToF camera ego-motion with the IMU data. Motion estimates of the camera were computed by applying the SIFT algorithm [20] to the amplitude images to find corresponding point features between two consecutive images. The 3D points were then used to compute the motion estimates, using the least squares method [19]. By fusion, the translation error was reduced by 1006mm in translation and by 25.4° in rotation.

3 DESIGN AND ANALYSIS

The design and analysis of the full ToF-IMU system that was capable of trajectory estimation of a mobile robot is divided into three sections. The first is the pre-processing step; the second is the ToF ego-motion estimation; and the third is the fusion of ToF ego-motion and IMU data.

3.1 3D ToF pre-processes

The pre-processing step is applied to all the ToF images to reduce the errors in the ego-motion estimation algorithms. In this work, a SR4000 from Swiss Ranger is used. This is the fourth-generation ToF camera from Swiss Ranger, and most of the systematic errors – such as internal scattering, which was an issue in the third-generation cameras – have been eliminated [13]. The SR4000 does not have temperature compensation, so a 40-minute warm-up time is allowed before any experiments are performed, as suggested in [8]. The non-systematic errors, which are mostly dependent on the scene configuration, are handled by the various filters discussed below.

3.1.1 Jump edge filter

Jump edge points occur when the transition between background and foreground objects is sudden. The ToF camera measurements show a smooth transition, as shown in Figure 1a. If these points are used for ego-motion estimation, they produce erroneous results, since they are random. A jump edge filter is adopted from [10] due to its simplicity and accuracy in finding jump edge points. Assume that $\mathbf{P} = \{\, \boldsymbol{p_i} \in \boldsymbol{R^3} \,|\, \mathrm{i} = 1, \cdots, \mathrm{N}\}$ represents a 3D point cloud from the ToF camera. To check if a point p_i is a jump edge, a set of eight neighbouring points $\boldsymbol{P_m} = \{\boldsymbol{p}_{i,m} \,|m = 1, \cdots, 8\}$ is extracted from the point cloud P. The angle θ_i between p_i and neighbouring points $p_{i,m}$ with the third vertex at the focal point of the camera is defined as

$$\boldsymbol{\theta_i} = \mathbf{max}_{m=1:8}\, \mathbf{sin}^{-1}\left(\frac{\|\boldsymbol{p_{i,m}}\|}{\|\boldsymbol{p_{i,m}} - \boldsymbol{p_i}\|}\, \mathbf{sin}\, \boldsymbol{\varphi}\right) \quad \textbf{(1)}$$

where φ is the apex angle of the two neighbouring pixels. An experimental environment was set-up to test the jump edge filter and the filtered image (see Figure 1b). The angle, defined in (1), is compared with the threshold angle to determine whether a point is a jump edge. The threshold angle is determined experimentally, depending on the camera. For this work, an angle of 175°is used.

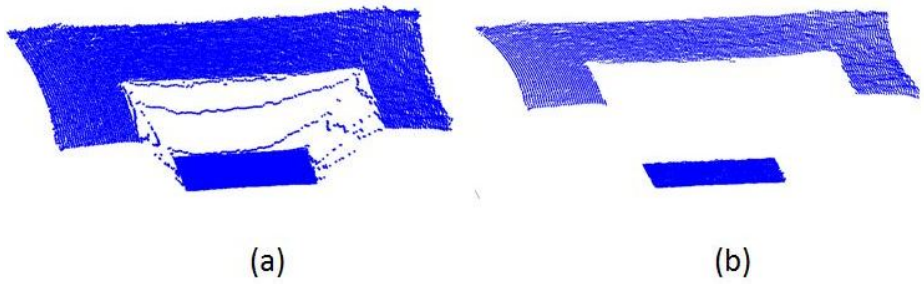


Figure 1: (a) Point cloud with jump edges as they can be seen between the transitions; and (b) point cloud with jump edge points removed using a jump edge filter

3.1.2 Non-homogeneous 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. The ToF camera was placed facing a white wall at different distances. At each position, 100 images were captured and their standard deviation computed. The standard deviation for each pixel at different locations is shown in Figure 2.

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 were removed.

3.1.3 Phase wrap filter

Phase wraps occur when the emitted light travels further than the maximum distance, which is controlled by the modulation frequency, of the ToF camera. Since the main application is for trajectory estimation, the implemented methods detect the phase wraps, but do not correct them. In cases where these points are needed for mapping or object reconstruction, methods such as the ones proposed in [36] [37] can be used. The method implemented here makes use of both range and amplitude images. A confidence map of the pixels is used as the product of the squared range (**d**) image and amplitude (**A**) image, which should give an approximately constant value, since distance and amplitude are inversely proportional. The confidence map **η** is computed as:

$$\boldsymbol{\eta} = \boldsymbol{A}.\boldsymbol{d^2} \quad (2)$$

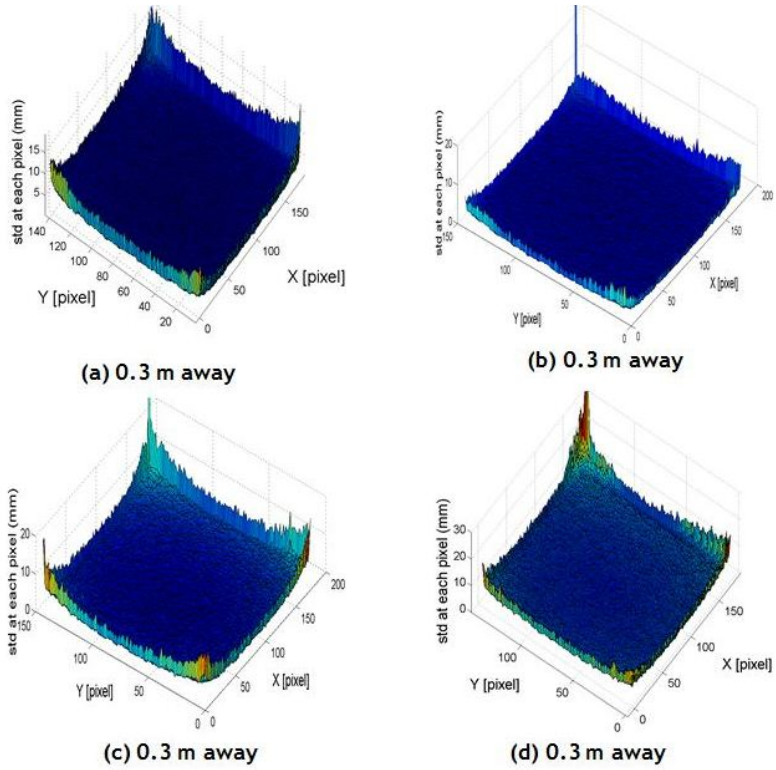


Figure 2: Standard deviation for 100 images while the ToF camera is stationary

The confidence map is compared with the threshold confidence map ($\boldsymbol{\eta_{thres}}$) to determine whether a pixel is wrapped. The threshold is computed to accommodate non-uniform illumination. The threshold is computed as:

$$\boldsymbol{\eta_{thres}} = \left(1 - \frac{(w_i - c_w)^2 + (h_i - c_h)^2}{10342}\right) T \ , \tag{3}$$

where w_i, h_i are the pixels' locations, c_w, c_h is the center pixel location, and T is the maximum value of the confidence map. This is determined experimentally, depending on the reflectivity of the environment. In this experiment, T is set to 5e4.

Another motivation for this simple method is that the camera is used in an environment with a uniform reflectivity. This means that it will not reject unwrapped points, and it will be able to determine wraps even if the whole image is phase wrapped – unlike in [37]. The application of the filter is shown in Figure 3.

3.2 ToF camera ego-motion estimation

A full 6D model is used for estimating ToF camera ego-motion. The camera pose at time t is represented by $\boldsymbol{T_t}$ and the transformation $\Delta\boldsymbol{T_t^{t+1}}$ transforms it to pose, T_{t+1} defined as:

$$\boldsymbol{T_{t+1}} = \boldsymbol{T_t T_t^{t+1}} \tag{4}$$

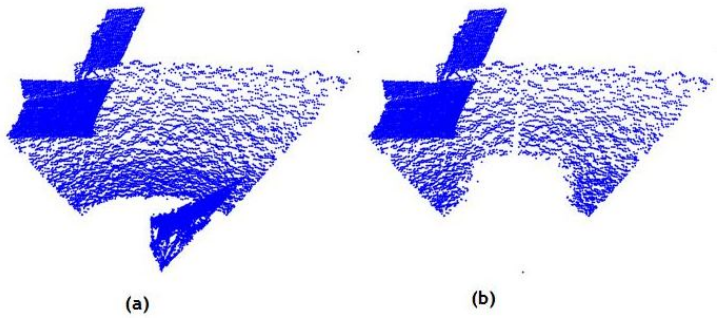


Figure 3 : Point cloud (a) before the phase-wrap filter, and (b) after the phase-wrap filter has been applied

The pose is represented by a 3×3 rotation matrix R and translation $\boldsymbol{t} = [t_x\ t_y\ t_z]^t$ in a matrix form as:

$$\boldsymbol{T} = \begin{bmatrix} \boldsymbol{R} & \boldsymbol{t} \\ \boldsymbol{0} & \boldsymbol{1} \end{bmatrix} \quad \textbf{(5)}$$

Since the mobile robot will be operating in an underground environment without unique landmarks, point feature extraction algorithms such as SIFT, SURF, and KLT do not work. Two variants of ICP are implemented: point-to-point ICP [4], and point-to-plane ICP [14]. The ICP algorithm tries to find the transformation between a base point cloud and scene point cloud by iteratively minimising the root mean square (RMS) error. The algorithm can be explained in five stages [15], summarised below.

1. **Selection of source points**
   The ToF camera provides 25,344 points, and thus using all these points will be time-consuming. By sampling a certain number of points such that the accuracy is not affected, the time taken can be decreased. Source points are randomly selected from the base point cloud.
2. **Matching of points**
   Once the source points have been sampled on the base point cloud, they must be matched to the corresponding points on the scene point cloud. The exact correspondence is not known. It is assumed that the closest point on the scene point cloud is the corresponding point. The closest point is found using a KD-tree.
3. **Weighting**
   This gives the quality of the corresponding points from the previous stage. A uniform weighing of unity is used.
4. **Rejection of outlier**
   Points that are classified as outliers are rejected. An adaptive distance threshold and multi-pairing rejection filters were implemented. An adaptive threshold filter rejects corresponding points with a distance greater than a certain threshold. Sometimes one point corresponds to multiple points. These points are rejected by a multi-pairing filter.
5. **Error metric**
   This refers to either a point-to-point ICP or a point-to-plane ICP. A point-to-point ICP algorithm finds the transformation by minimising the sum of the squared distances between the corresponding pairs. It can be formulated as:

$$\boldsymbol{\varepsilon_{pp}(R,t)} = \sum_j \left\| \boldsymbol{Rs_j + t - d_j} \right\|^2 \quad \textbf{(6)}$$

A point-to-plane ICP minimises the sum of the squared distance between points in the base point cloud and the tangent planes of the corresponding point on the scene point cloud. It is formulated as

$$\varepsilon_{pl}(\boldsymbol{R},\boldsymbol{t}) = \sum_j \left[\left(\boldsymbol{R}\boldsymbol{s}_j + \boldsymbol{t} - \boldsymbol{d}_j\right).\boldsymbol{n}_j\right]^2, \quad (7)$$

where n_j is the normal vector of the surface place of the base point cloud. The surfaces normal are estimated using least squared methods proposed by Mitra et al. [38]. Unlike the point-to-point, the point-to-plane does not have a closed form solution, and it is solved by linearisation of the transformation by using a small angle approximation ($\sin\theta = 0\ and\ \cos\theta = 1$). Algorithm 1 shows an implemented point-to-plane algorithm that is similar to the point-to-point as well – except that there is no surface normal computation.

Algorithm 1 ICP point-to-plane algorithm

Get the base point cloud (**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 (**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
  Equate the base point cloud to scene point cloud for the next iteration
end for

3.3 Fused ToF-IMU ego-motion estimation

The fusion of ToF ego-motion and IMU data is performed using the Kalman filter method. ToF camera ego-motion is erroneous, especially with rotational movement; and IMU data can be useful in these situations to provide a robust estimation. For our implementation, the problem of trajectory estimation is simplified into a 3D trajectory motion that can be illustrated by a mobile robot moving on a 2D planar surface, as shown in Figure 4.

The fusion performed using the Kalman filter with a ten element state vector, **x**, is represented as:

$$\boldsymbol{x} = \left[\boldsymbol{V_I} \quad \boldsymbol{b_a} \quad \omega \quad \boldsymbol{b_g} \quad {}^{G}\boldsymbol{p_I} \quad \theta \quad a_\theta\right], \quad (8)$$

where $\boldsymbol{V_I}$ is the velocity of IMU with respect to IMU frame , $\boldsymbol{b_a}$ is the accelerometer bias, ω is the angular velocity, b_g is the gyroscope bias, ${}^{G}\boldsymbol{p_I}$ is the position of IMU with respect to global frame, θ is the orientation of IMU, and a_θ is the angular acceleration. The Kalman filter algorithm is divided into a time update stage and a measurement update stage. In the time update stage, the IMU measurements that are sampled at 100 Hz are integrated to track orientation, angular velocity, and the position of the IMU. The IMU measurements are modelled as:

$$\omega_m(t) = \omega(t) + b_g(t) + n_{rg}(t), \quad (9)$$

$$\boldsymbol{a_m}(\boldsymbol{t}) = \boldsymbol{R_G^I}\left({}^{G}\boldsymbol{a}(\boldsymbol{t}) - {}^{G}\boldsymbol{g}\right) + \boldsymbol{b_a}(\boldsymbol{t}) + \boldsymbol{n_{ra}}(\boldsymbol{t}), \quad (10)$$

where ω_m and $\boldsymbol{a_m}$ are the measured angular velocity and linear acceleration, while ω and $\boldsymbol{a}$ are the actual angular velocity and linear acceleration respectively. n_{rg} and $\boldsymbol{n_{ra}}$ represent zero-mean Gaussian white noise.

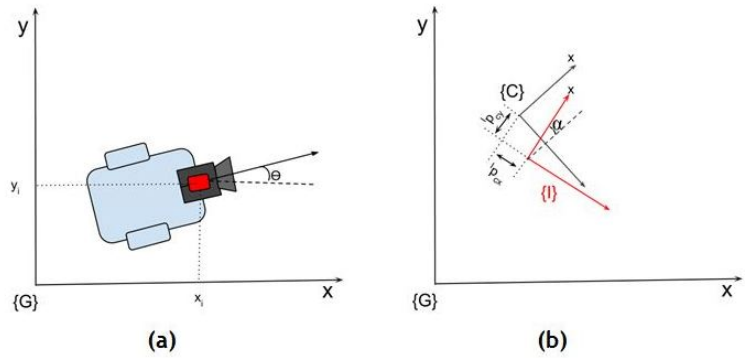


Figure 4: (a) Simplification of the 2D planar problem where (x,y) is the position and θ is the orientation. (b) Co-ordinates frames where {G} is the global frame, {I} is the IMU frame, and {C} is the camera frame. ${}^{i}P_c$ is the transformation.

A state error vector is defined as:

$$\tilde{\boldsymbol{x}} = \boldsymbol{x} - \hat{\boldsymbol{x}}, \tag{11}$$

where $\boldsymbol{x}$ *is* the actual state vector and $\hat{\boldsymbol{x}}$ is the estimated state vector. A continuous time error state equation is defined as

$$\dot{\tilde{\boldsymbol{x}}} = \boldsymbol{F_c}\tilde{\boldsymbol{x}} + \boldsymbol{G_c}\boldsymbol{n}, \tag{12}$$

where

$$\boldsymbol{F_c} = \begin{bmatrix} \boldsymbol{0_{2\times2}} & \boldsymbol{-I_{2\times2}} & \boldsymbol{0_{2\times1}} & \boldsymbol{0_{2\times1}} & \boldsymbol{0_{2\times2}} & \boldsymbol{\hat{R}_\theta \Omega \hat{a}} & \boldsymbol{0_{2\times2}} \\ \boldsymbol{0_{2\times2}} & \boldsymbol{0_{2\times2}} & \boldsymbol{0_{2\times1}} & \boldsymbol{0_{2\times1}} & \boldsymbol{0_{2\times2}} & \boldsymbol{0_{2\times2}} & \boldsymbol{0_{2\times2}} \\ \boldsymbol{0_{1\times2}} & \boldsymbol{0_{1\times2}} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0_{1\times2}} & \boldsymbol{0} & \boldsymbol{1} \\ \boldsymbol{0_{2\times2}} & \boldsymbol{0_{2\times2}} & \boldsymbol{0_{2\times1}} & \boldsymbol{0_{2\times1}} & \boldsymbol{0_{2\times2}} & \boldsymbol{0_{2\times2}} & \boldsymbol{0_{2\times2}} \\ \boldsymbol{\hat{R}_\theta} & \boldsymbol{0_{2\times2}} & \boldsymbol{0_{2\times1}} & \boldsymbol{0_{2\times1}} & \boldsymbol{0_{2\times2}} & \boldsymbol{\hat{R}_\theta \Omega \hat{V}_I} & \boldsymbol{0_{2\times2}} \\ \boldsymbol{0_{2\times2}} & \boldsymbol{0_{1\times2}} & \boldsymbol{1} & \boldsymbol{-1} & \boldsymbol{0_{1\times2}} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0_{2\times2}} & \boldsymbol{0_{2\times2}} & \boldsymbol{0_{2\times1}} & \boldsymbol{0_{2\times1}} & \boldsymbol{0_{2\times2}} & \boldsymbol{0_{2\times2}} & \boldsymbol{0_{2\times2}} \end{bmatrix}$$

$$\boldsymbol{\Omega} = \begin{bmatrix} \boldsymbol{0} & \boldsymbol{-1} \\ \boldsymbol{1} & \boldsymbol{0} \end{bmatrix} \qquad \boldsymbol{R_\theta} = \begin{bmatrix} \boldsymbol{\cos\theta} & \boldsymbol{-\sin\theta} \\ \boldsymbol{\sin\theta} & \boldsymbol{\cos\theta} \end{bmatrix}$$

$$\boldsymbol{G_c} = \begin{bmatrix} \boldsymbol{-I_{2\times2}} & \boldsymbol{0_{2\times2}} & \boldsymbol{0_{2\times1}} & \boldsymbol{0_{2\times1}} \\ \boldsymbol{0_{2\times2}} & \boldsymbol{I_{2\times2}} & \boldsymbol{0_{2\times1}} & \boldsymbol{0_{2\times1}} \\ \boldsymbol{0_{1\times2}} & \boldsymbol{0_{1\times2}} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0_{1\times2}} & \boldsymbol{0_{1\times2}} & \boldsymbol{0} & \boldsymbol{1} \\ \boldsymbol{0_{2\times2}} & \boldsymbol{0_{2\times2}} & \boldsymbol{0_{2\times1}} & \boldsymbol{0_{2\times1}} \\ \boldsymbol{0_{1\times2}} & \boldsymbol{0_{1\times2}} & \boldsymbol{-1} & \boldsymbol{0} \\ \boldsymbol{0_{1\times2}} & \boldsymbol{0_{1\times2}} & \boldsymbol{0} & \boldsymbol{0} \end{bmatrix}$$

$$\boldsymbol{n} = \begin{bmatrix} \boldsymbol{n_{ra}} \\ \boldsymbol{n_{wa}} \\ \boldsymbol{n_{rg}} \\ \boldsymbol{n_{wg}} \end{bmatrix}$$

The covariance matrix is computed as:

$$\boldsymbol{P_{k+1/k}} = \boldsymbol{\Phi_k P_{k/k} \Phi_k^T} + \boldsymbol{Q_d}, \tag{13}$$

where $\boldsymbol{\Phi}$ is the state transition matrix and Q_d is the system noise covariance matrix

$$\boldsymbol{\Phi(t+\Delta t, t) = exp(F_c \Delta t)}, \quad \textbf{(14)}$$

$$\boldsymbol{Q_d = \int_{t_k}^{t_k+T} \Phi(t_{k+1}, \tau) G_c Q_c G_c^T \Phi(t_{k+1}, \tau)}, \quad \textbf{(15)}$$

and $\boldsymbol{Q_c}$ is the covariance matrix of the IMU measurements that depend on the noise characteristics of accelerometer and gyroscope.

The measurement update stage takes the ToF ego-motion estimation as measurements, and updates the state vector. ToF ego-motion estimates a change in position and change in orientation. Since the mobile robot will be moving on a 2D surface, this is simplified into $[t_x \quad t_y \quad \Delta\theta]$. Since the time difference is also known, the measurements are taken as linear velocity and angular velocity. Given ω and V defined as:

$$\boldsymbol{V = \begin{bmatrix} v_x \\ v_y \end{bmatrix} = \begin{bmatrix} t_x/\Delta t \\ t_y/\Delta t \end{bmatrix} \text{ and } \omega = \Delta\theta/\Delta t.}$$

measurements are modelled as:

$$\boldsymbol{\hat{z} = h(x) = \begin{bmatrix} R_\alpha V \\ \omega \end{bmatrix} + \eta}, \quad \textbf{(16)}$$

where R_α is the orientation difference between the IMU frame and the ToF camera frame, estimated during the calibration of the ToF-IMU system; $\boldsymbol{\eta}$ is the noise from the ToF ego-motion. The Jacobian matrix for the measurement error state is:

$$\boldsymbol{H = \begin{bmatrix} R_\alpha & 0_{2\times2} & 0_{2\times2} & 0_{2\times2} & 0_{2\times2} & 0_{2\times2} \\ 0_{1\times2} & 0_{1\times2} & 1 & 0 & 0_{1\times2} & 0_{1\times2} \end{bmatrix}}. \quad \textbf{(17)}$$

Assuming that propagated state estimate $\boldsymbol{X_{k/k}}$, propagated covariance matrix estimate $\boldsymbol{P_{k/k+1}}$, current measurement z, estimated measurement $\hat{z}$, and the error measurement Jacobian matrix H are computed, then updated state estimate $\boldsymbol{X_{k/k+1}}$ is computed using Algorithm 2.

Algorithm 2 Updating the state estimates of the Kalman filter

1: Residual r according to

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

2: Covariance of the residual S as

$$\mathrm{S} = \mathrm{HPH}^T + \mathrm{R}$$

2: Kalman gain

$$\mathrm{K} = \mathrm{PH}^T\mathrm{S}^{-1}$$

3: Error vector

$$\tilde{\mathrm{x}} = \mathrm{Kr}$$

4: Update the state vector as

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

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

$$\mathrm{P}_{k+1/k+1} = (\mathrm{I} - \mathrm{KH})\mathrm{P}_{k+1/k}(\mathrm{I} - \mathrm{KH})^T + \mathrm{KRK}^T$$

4 EXPERIMENTAL RESULTS AND DISCUSSIONS

All the algorithms are implemented in MATLAB® and tested on an offline dataset. The dataset is collected using a PC running the Robot Operating System (ROS)[§] on Ubuntu 12.04. The final system is supposed to operate in an underground mine environment, but because

[§] http://www.ros.org/wiki/

of the inherent difficulties in obtaining the groundtruth in an underground mine, an artificial mine stope was used, as shown in Figure 5. The dataset was collected in the evening to simulate the darkness of underground mines.

The groundtruth is collected using the Vicon motion capture system. The Vicon system is a motion capture system that uses passive infra-red reflective markers to track the pose of an object in space. It has sub-millimetre accuracy and an update rate of more than 100 Hz.

To evaluate the results, let $\boldsymbol{T_t^e}$ be the estimate pose, $\boldsymbol{T_t^G}$ the groundtruth pose provided by the Vicon system at time t. The evaluation is done in terms of absolute error $\boldsymbol{M_t^{abs}}$ and incremental error $\boldsymbol{M_t^{inc}}$, defined in equations (18) and (19).

$$\boldsymbol{M_t^{abs} = \left(T_t^g\right)^{-1} T_t^e = \begin{bmatrix} \Delta R_{abs} & \Delta t_{abs} \\ 0 & 1 \end{bmatrix}.}$$

Figure 5: Artificial mine stope (18)

Translational error $\boldsymbol{e_{abs,\Delta t}}$ and rotational error$\boldsymbol{e_{abs,\Delta R}}$ are computed as:

$$\boldsymbol{e_{abs,\Delta t} = \|\Delta t_{abs}\|,}$$
$$\boldsymbol{e_{abs,\Delta R} = \left\|[\alpha \quad \beta \quad \gamma]^T\right\| = \|f_{\Delta R}(\Delta R_{abs})\|,}$$

where $[\alpha \quad \beta \quad \gamma]^T$ are the Euler angles. Similarly, the incremental errors are measured by the sum of the difference of the change in the real pose ΔT_t^g and change in the estimated pose $\boldsymbol{\Delta T_t^e}$

$$\boldsymbol{M_{inc,t} = \left(\Delta T_t^g\right)^{-1} \Delta T_t^e = \begin{bmatrix} \Delta R_{inc,t} & \Delta t_{inc,t} \\ 0 & 1 \end{bmatrix},} \qquad \textbf{(19)}$$
$$\boldsymbol{e_{inc,\Delta t} = \sum_i \left\|\Delta t_{inc,i}\right\| \ , \quad e_{inc,\Delta R} = \sum_i \|f_{\Delta R}(\Delta R)\|.}$$

For the ToF ego-motion estimation, the RMS errors and the number of iterations are also used as a measure of evaluation. RMS error measures the errors that remain after two point clouds have been registered. Before the ego-motion estimation algorithms are applied, the pre-processing steps, which include 40 minutes warm-up time, the jump edge filter, the phase-wrap filter, and boundary points rejection, are applied.

The ICP algorithms were tested on the dataset collected on an artificial mine stope, where the mobile robot moved at an average linear velocity of 0.2 m/s and angular velocity of 10°/s. The path was approximately 8m long. The trajectory estimation is shown in Figure 6.

Both the point-to-point ICP and point-to-plane ICP trajectory estimates follow the correct path until the mobile robot starts to turn or to rotate. After turning, the error in the rotation causes both the trajectories to diverge from the actual path. The absolute and incremental errors are shown Figure 7. The absolute translation error at the end of the path was 4.33m for point-to-plane ICP and 1.03m 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 ICP produces better results for translation, but suffers a large margin of error in rotation. The reason for good rotational results in point-to-plane is that ICP minimises the error between a point and a plane rather than point-to-point. The incremental errors, which give a better description of the error, show that the point-to-

point error was 7.32m and 119.6° for translation and rotation respectively, while the point-to-plane ICP error was 13.57m and 123.8° for translation and rotation.

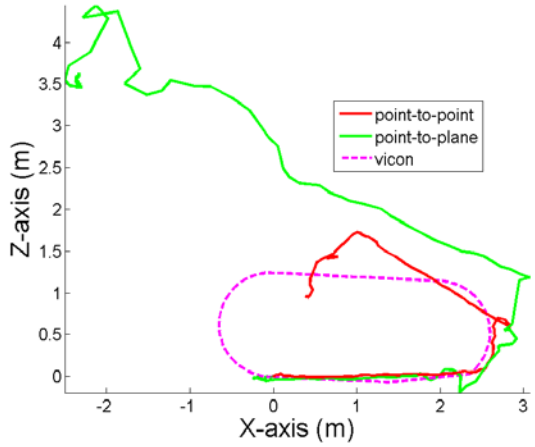


Figure 6: Comparison of point-to-point ICP and point-to-plane ICP with Vicon estimates

Note that the rate of error increase is high while the robot is turning. This is because the ToF camera produces erroneous measurements for rotation due to its measurement principle.

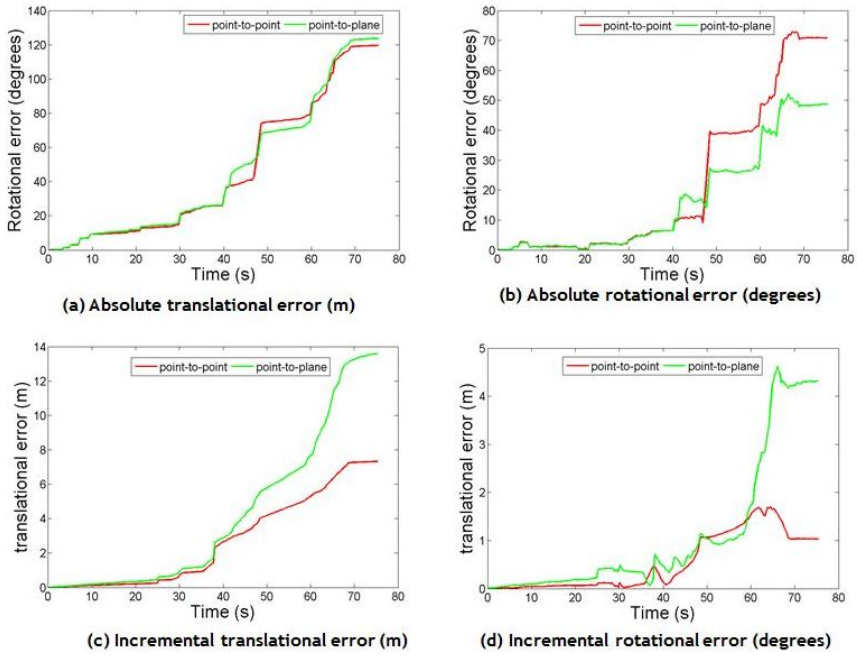
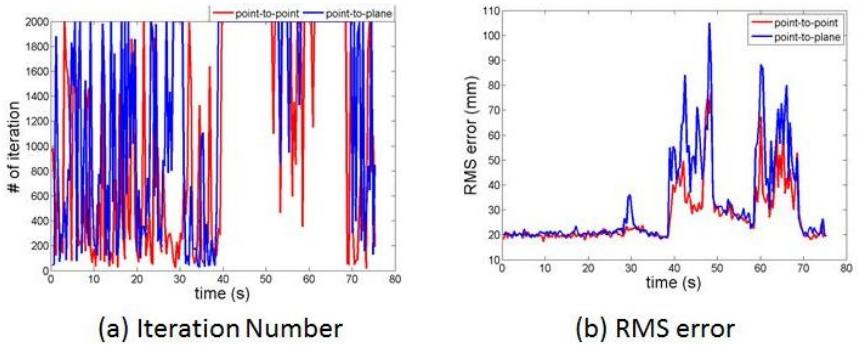


Figure 7: ICP-based algorithm errors where (a) and (b) represent the absolute errors and (c) and (d) show incremental errors

The RMS errors and number of iterations are shown in Figure 8; they show similar results for rotational motion. This can be improved by incorporating a low-cost IMU.

The fusion was implemented using point-to-point ICP, since it proved to be the most accurate one for our application. The transformation between IMU and ToF camera is estimated using the technical drawings of these sensors. The Kalman filter implementation was tested on an 8.7m long path, with an average linear velocity of 0.2 m/s and average

angular velocity of 30°/s. The angular velocity is higher than in the previous experiment in order to show maximum improvement of Kalman filter. The trajectory estimations are shown in Figure 9.



(a) Iteration Number (b) RMS error

Figure 8 (a) Number of iteration over time and (b) RMS error for both point-to-point ICP and point-to-plane ICP

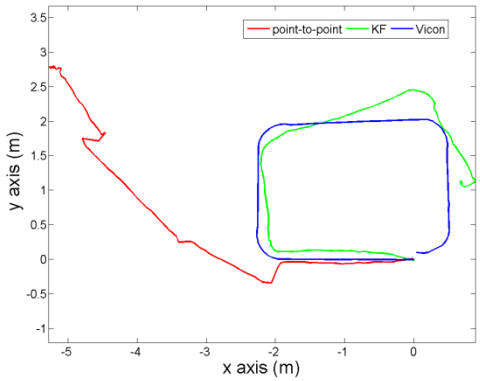


Figure 9: Comparison of point-to-point ICP, Kalman filter-based, and Vicon groundtruth

The Kalman filter trajectory estimation shows a huge improvement compared with the point-to-point ICP. In terms of absolute errors, translational error was reduced by 4.72m and 260.2° for rotational error. The translation incremental error was reduced by 7.17m, 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. See Figure 10 for incremental and absolute errors.

These results compare with the results of fusing ToF ego-motion with IMU data done by Droeschel et al. [35], where the SIFT algorithm was used for ToF ego-motion estimation. The robot similarly moved along a square path with an approximate total length of 4.8m. For the SIFT algorithm to find enough features, posters and calibration checkerboard were placed in the environment. The SIFT algorithm produced better results than the point-to-point ICP used in this paper. This is due to the differences in the environment.

5 CONCLUSION AND FUTURE WORK

This research presented the design and implementation of a system capable of estimating the trajectory of a robot operating in an underground mining stope. The estimation is performed using a ToF camera and an IMU. Even though the main goal was investigate the fusion of ToF camera ego-motion with IMU data, some of the questions that were answered were:

- Which methods were best for filtering out erroneous points for the ToF camera?

- Which algorithm was best for ego-motion of the ToF camera data in an underground mine stope?
- By how much does the fusion improve the trajectory estimation?

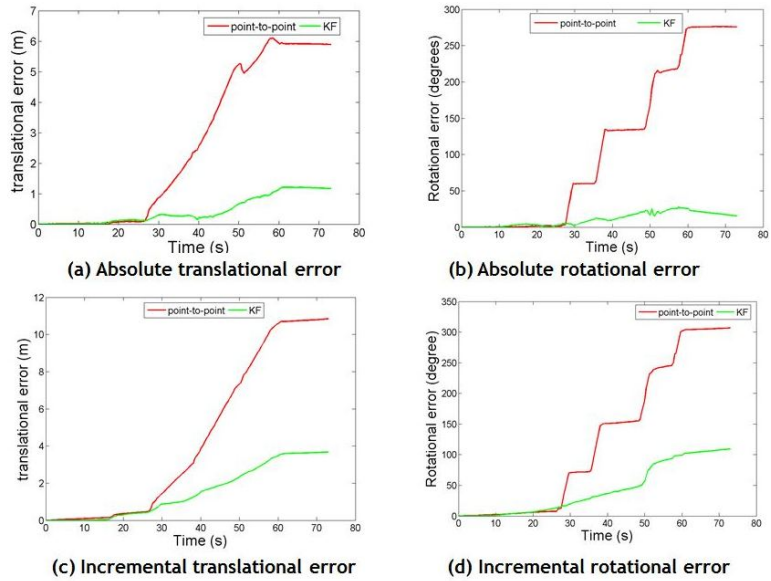


Figure 10: Trajectory estimation errors where (a) and (b) are the absolute errors and (c) and (d) are the incremental errors

The point-to-point ICP produced better results than the point-to-plane ICP. These algorithms suffer high errors when the robot is turning, due to the error in the ToF measurements. The fusion of ToF ego-motion with IMU data increased the accuracy of the trajectory estimation of a mobile robot. The fusion was performed using the Kalman filter algorithm.

The transformation between the ToF camera and the IMU was estimated using the technical drawings of these sensors, but a more accurate transformation can be estimated using the extended Kalman filter. The fusion was only performed for a robot operating on a 2D planar surface; future work will involve extending it to a full 6D motion.

REFERENCES

[1] **Nüchter, A., Lingemann, K., Hertzberg, J. & Surmann, H.** 2007. 6D SLAM—3D mapping outdoor environments: Research Articles, *J. Field Robot., V*ol. 24, August 2007, pp. 699-722.

[2] **Nüchter, A., Surmann, H., Lingemann, K., Hertzberg, J. & Thrun, S.** 2004. 6D SLAM with an application in autonomous mine mapping, in *Proceedings of the IEEE International Conference of 3D Digital Imaging and Modeling (3DIM)*.

[3] **Magnusson, M., Lilienthal, M. & Duckett, T.** 2007. Scan registration for autonomous mining vehicles using 3d-ndt, *Journal of Field of Robotics,* pp. 803-827.

[4] **Besl, P.J. & McKay, N.D.** 1992. A method for registration of 3-d shapes, *IEEE Trans. Pattern Anal. Mach. Intell.,* vol. 14, pp. 239-256.

[5] **Woodman**, **O.J.** 2007. An introduction to inertial navigation, University of Cambridge, Computer Laboratory.

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

[7] **Lange, R.** 2000. 3D Time-of-Flight distance measurement with custom solid-state image sensors in CMOS/CDD-Technology.

[8] **Piatti, D. & Rinaudo, F.** 2012. SR-4000 and camcube 3.0 time-of-flight (tof) cameras: Tests and comparison, in *Remote Sensing*.

[9] **Fuchs, S. & Hirzinger, G.** 2008. Extrinsic and depth calibration of tof-cameras, in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*.

[10] **May, S., Droeshel, D., Holz, D., Fuchs, S., Malis, E., Nüchter, A. & Hertezberg, J.** 2009. Three-dimensional mapping with time-of-flight cameras, *J. Field Robotics,* vol. 26, pp. 934-965.

[11] **Kalmann, T., Remondino, F. & Ingensand, H.** 2006. Calibration for increased accuracy of the range imaging camera SwissRanger, *International Archives of Photogrammetry, Remote Sensing amd Spatial Information Sciences*.

[12] **Lindner, M. & Kolb, A.** 2006. Lateral and depth calibration of PMD-distance sensors, *ISVC,* pp. 524-533.

[13] **Piatti, D.** 2011. *Time-of-Flight camera: Tests, calibration and multi-frame registration for automatic 3D object reconstruction*, PhD thesis, Politecnico Di Torino: Doctoral School of Environment and Territory XXII cycle.

[14] **Chen, Y. & Medioni, G.** 1992. Object modelling by registration of multiple range images, *Image Vision Comput.,* vol. 20, pp. 145-155.

[15] **Rusinkiewicz, S. & Levoy, M.** 2001. Efficient variants of the ICP algorithm, in *Third International Conference on 3D Digital Imaging and Modelling (3DIM)*.

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

[17] **Ohno, K., Nomura, T. & Tadokoro, S.** 2006. Real-time robot trajectory estimation and 3d map construction using 3d camera, in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

[18] **Fuchs, S. & May S.** 2008. Calibration and registration for precise surface reconstruction with time of flight cameras, *Int. J. Intell. Syst. Technol. Appl.,* vol. 5, pp. 274-284.

[19] **Arun, K.S., Huang, S.H. & Blostein, S.D.** 1987. Least-squares fitting of two 3-d point sets, in *IEEE Trans. Pattern Anal. Mach. Intell.*.

[20] **Lowe, D.G.** 2004. Distinctive image features from scale-invariant keypoints, *Int. J. Comput. Vision,* vol. 60, pp. 91-110.

[21] **Kanade, T. & Tomasi, C.** 1991. Detection and tracking of point features, *International Journal of Computer Vision*.

[22] **Ezio, M.** 2011. Esm visual tracking. [Online]. Available: http://esm.gforge.inria.fr/.

[23] **Malis, E.** 2007. An efficient unified approach to direct visual tracking of rigid and deformable surfaces, *IROS,* pp. 2729-2737.

[24] **Bay, H., Tuytelaars, T. & Gool, L.V.** 2006. SURF: Speed up robust features, in *ECCV*.

[25] **Rousseeuw, P. & Levroy, A.** 2003. *Robust regression and outlier detection*. Wiley Series in Probability and Statistics.

[26] **Pathak, K., Birk, A., Vaskevicius N. and Poppinga, J.** 2010. Fast registration based on noisy planes with unknown correspondences for 3-d mapping, in *IEEE Transactions on Robotics,* Vol. 26, pp. 424-441.

[27] **Villaverde, I. & Grana, M.** 2011. Neuro-evolutionary mobile robot egomotion estimation with a 3d tof camera, *Neural Computing and Applications, vol. 20, no. 3, pp. 345-354*.

[28] **Wang, S. & Yu, H.** 2011. A variational approach for ego-motion estimation and segmentation based on 3d tod cameras, in *4th International Congress on Image and Signal Processing (CISP), vol. 3, pp. 1160-1164*.

[29] **Shuster, M.D.** 2006. The generalised wahba problem, *The Journal of the Astronautical Sciences,* vol. 54, no. 2, pp. 245-259.

[30] **Martinetz, T. & Schulten, K.** 1991. A neural-gas network learns topologies, *Artificial Neural Networks,* vol. I, pp. 397-402.

[31] **Martinetz, T., Berkovich, S. & Schulten, K.** 1993. Neural-gas network fo vector quantization and its application to time-series prediction, *IEEE Transactions on Neural Networks,* vol. 4, no. 4, pp. 558-569.

[32] **Corke, P., Lobo, J. & Dias, J.** 2007. An introduction to inertial and visual sensing, *The international Journal of Robotics,* vol. 26, pp. 519-535.

[33] **Durrant-Whyte, H. & Henderson, T.** 2008. Springer handbook of robotics, *Multisensor Data Fusion*, pp. 585-610.

[34] **Arulampalam, M.S., Maskell, S. & Gordon, N.** 2002. A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking, *IEEE Transactions on Signal Processing,* vol. 50, pp. 174-188.

[35] **Droeschel, D., May, S., Holz, D. & Behnke, S.** 2011. Fusion time-of-flight cameras and inertial measurement units for ego-motion estimation, *AUTOMATIKA,* vol. 52, no. 3, pp. 189-198.

[36] Droeschel, D., Holz, D. & Behnke, S. 2010. Multi-frequency phase unwrapping for time-of-flight cameras, *IROS,* pp. 1463-1469.

[37] Droeschel, D., Holz, D. & Behnke, S. 2010. Probabilistic phase unwrapping for time-of-flight cameras, *ISR/ROBOTIK,* pp. 1-7.

[38] Mitra N.J., Nguyen, A. & Guibas, L. 2004. Estimating surface normals in noisy point cloud data, *Special issue of International Journal of Computational Geometry and Applications,* vol. 14, pp. 261-276.