International Journal of Progressive Sciences and Technologies (IJPSAT)
ISSN: 2509-0119.

http://ijpsat.ijsht-journals.org
Vol. 29 No. 1 October 2021, pp.138-166

A Reliable Linearly Separable Optimization Method for SLAM Estimation based on Stereo Features

Mohamed Hamdy Merzban[*1],Mohamed Hassan M. Mahmoud[*2],Mohamed Abdellatif[*3]

[*1]*Communictions Engineering Dept. , Faculty of Engineering, Fayoum University ,Egypt*
[*2]*Communictions Engineering Dept. , Giza Higher Institute Of Engineering and Technology, Egypt*
[*3] *Japan University of Science and Technology, Egypt*

Abstract – SLAM (Simultaneous Localization and Mapping) based on graphical representation is used to solve optimization non-linear problems. The solutions of graph-based SLAM have n noisy measurements and re-projection errors lead to uncertainty in robot pose and landmark position. The proposed system provides novel two-stage architecture for visual SLAM that was designed and implemented. The new two-stage architecture resulted in an exactly linear graph optimization, by using a linear optimization method for graph optimization, so the problem now becomes linear. We take benefit of all camera measurements and needn't ignore any frames. We enhance accuracy of large scale VSLM system, minimize processing time and decrease computational cost. We show that our system can be used with a higher accuracy and processing time than well-known systems based on bundle adjustment.

Keywords – Graph based SLAM (Simultaneous Localization and Mapping); Stereo Features; Linearization; Consistent Map.

I. INTRODUCTION

SLAM wasSLAM is the basis for any navigation system [1] whose aim is to constructa map for an environment and estimate for robot pose. SLAM is modelled as an optimization hard problem due to there are two unknowns (robot pose and landmark position) and the only given is stream of images. SLAM work can be described by three models first model is measurement model which is used to get location of landmarks in the environment, second model is motion model which is used to find relative pose of robot using e-pipolar geometry (two view geometry), and third model is re-projection (sensor) error model to correct and update of robot pose and landmark position.

A real-time implementation for SFM was proposed by [6]. To enable real-time performance, the author keeps track of a finite window of image frames and applies a RANSAC scheme with less computational cost [7]. Landmarks that disappear from the tracked window frames are removed from the system. This real-time version of SFM were termed visual odometry. Visual odometry suffers from continuous error drift in its pose estimation. In [8], a local BA refinement stage was added to the system to enhance accuracy, the authors keep track of a finite window of key-frames that are selected from the captured camera frames. This window is adjusted continuously by adding new key-frames and removing old ones from the system. In [9] a Visual odometry system was built using a stereo-camera, the features within each image pair from a stereo camera were matched to yield a disparity image.

Another implementation of Visual-SLAM with mono-camera based on an optimization approach was achieved by [10]. In this system, the process of tracking and the process of map update are performed in two parallel threads. The tracking is performed at

Corresponding Author: Mohamed Hamdy Merzban

frame rate, while the map update/optimization is performed at more distant times. This approach assigns more time budget to the map update process, hence enabling larger maps.

As an attempt to simplify map representation, a sparse information matrix that holds correlations between landmarks was used in [11], while in [12] a relative map representation was used, where the relative displacement vector between each two landmarks was maintained.

The inclusion of an IMU sensor to enhance accuracy was achieved in many systems. There are two methods to merge visual and inertial data:

Light Coupling, in which we separate the estimation process into two processes: 1) Visual odometry. 2)Integration of gyroscope's measured angular velocity and accelerometer's measured acceleration to yield orientation, velocity and position. The two concurrent estimators are fused together using EKF. Light coupling is relatively stable and simple but its accuracy is inferior compared to tight-coupling.

(2) Tight coupling, where the IMU measurements are integrated from frame to frame, thus producing inter-frame prediction for the change in orientation and position. These predictions are fused with visual data in the bundle adjustment stage. Tight coupling is more difficult in implementation compared to light coupling, it may suffer from instability if the visual and inertial data are inconsistent, otherwise it is expected to yield more accurate results.

SLAM systems based on mono-camera experience scale ambiguity. The camera path and landmark positions are known up to unknown scale factor.The combination of a mono-camera and IMU can recover the unknown scale besides enhancing estimation accuracy. In [13], an observability analysis for a hybrid visual-inertial SLAM system is conducted. The author states that the scale, IMU sensor biases, and gravity are observable given that motion dynamics are rich enough (i.e. angular velocity and time derivative of acceleration are not zero). The scale is unobservable in the case of constant velocity motion. The developed system uses Extended Kalman Filter (EKF) to estimate the motion path of a vehicle on a road with reported error on the order of 0.5\%. The hybrid visual-inertial approach with mono-camera was also applied in [14, 15, 16] using EKF.

Hybrid visual-inertial navigation systems requires elaborate calibration procedures which includes IMU sensor calibration, and camera to IMU transformation. Precise calibration of an IMU requires special equipment [17]. In [18] a method for in-field calibration of an IMU is developed. The calibration procedure is based upon the fact that an accelerometer at rest can measure the gravity vector which is constant in magnitude and direction. The calibration of a gyroscope is also possible by using gravity vector as a direction reference.

In [19], a vision-aided inertial navigation visual odometry was developed using a stereo-camera and an IMU. The system consists from two components: 1) Visual odometry (VO) module, 2) Fusion module based on EKF that fuses the measurements and inertial measurements. The results show that the combination of VO+IMU provides better accuracy than only.

In [20], a hybrid visual-inertial navigation system was developed using a stereo-camera and an IMU. The inertial measurements are integrated from frame to frame and the integrands are added to the BA optimization together with visual measurements.

The inter-sensor (camera to IMU) calibration can be performed off-line. A more flexible approach is to estimate the inter-sensor calibration on-line. In [16], a system is developed that estimates the inter-sensor calibration on-line, this kind of systems is sometimes called power-and-go. Low-cost commercial IMUs have considerable bias components that are time-varying, therefore an online-bias estimation component is necessary for a long-running system.

II. Related Work

There is a range of related work regarding the SLAM problem that was traditionally solved using two approaches:

1) *EKF or unscented Kalman filte*r are used to estimate the robot pose. The robot pose and landmark positions are considered random variables with Gaussian distributions and represented with their means and covariance matrix. Each measurement update requires the propagation of the mean and covariance matrix.
2) *Optimization approach* is an optimization criteria that is defined to minimize the captured images reprojection-errors. Each new image measurement adds new terms to the optimization function. The optimization function can be minimized

using the Levenberg-Marquardt algorithm. The Jacobians involved in the optimization process are sparse. This sparsity allows very efficient solution to the optimization problem [21].

Nowadays, it is thought that optimization approach is better than filtering ones, the reason is that EKF was shown to be inconsistent in the long-term [22], and it was also shown that optimization approach is less computationally expensive than EKF at the same accuracy. Therefore, most state of the art SLAM algorithms rely on a graph-based representation of the map and optimizing it.

In [23], the system is comprised of several modules including a visual odometry module with frame-rate pose updates and a mapping and registration modules for building a map and registering new frames with respect to it. To increase scalability, the map is represented as a skeleton of key-frames (pose graph) *with non-linear constraints between them*. The non-linear constraints are derived from matched features between frames. This approach allows larger maps, since there is no need to keep track of the image features.

In, Graph Based Methods, as the map-size increases, the computational cost of map update/optimization increases. An attempt to resolve this issue was achieved by [24], where the map is encoded in a relative way and performs bundle-adjustment on a chosen subset of poses and landmarks that are relevant to the pose estimate. In [25], the optimization is done at two levels, at the local window level the point-pose graph is optimized while at the global window level the pose-pose graph is optimized. The two optimization levels are coupled in a single optimization criteria. In [26], a stereo-vision system is designed and employs relative pose graph representation for the map. The system recognizes a sub-part of the relative map as an active region, then optimizes this active region only. The author reports that a localization error of a few meters is possible for several kilo-meters of traveled distance.

Another difficulty arises as the map size increases, the number of features stored in the map increases very rapidly and matching new frame features to the map stored features becomes computationally expensive. Appearance based techniques were utilized to resolve this issue. In [] an algorithm for image search in a database of images was introduced, the algorithm was inspired by the text search techniques used in database systems. The algorithm starts with a training phase, in which feature descriptors are extracted from images and divided into clusters using K-means algorithm. The mean of each cluster is then used to represent the descriptors within it where it is called visual words. In the second phase, each image is characterized by a set of visual words and an inverse index is built which includes each visual word and the images where it exists in. In the third phase, the query image is processed to extract its features, these features are mapped to visual words, and a vector of frequency of occurrence of each visual word is formed. This query vector is then used to search for the matching image.

Based on the success of image query techniques discussed earlier, an appearance-only SLAM technique was devised. Appearance only SLAM is concerned by solving the issue of place-recognition and building a topological map of places. Unlike metric map SLAM, there is no need for map-optimization, therefore the system can be very efficient for large maps.

In [27, 28] an appearance only SLAM system was developed. The system builds a topological map of the places and can achieve loop-closure very efficiently. The system in [29] combined conventional metric SLAM with appearance based methods can achieve a loop-closure very efficiently. The system contains the subsystems shown in figure 3.

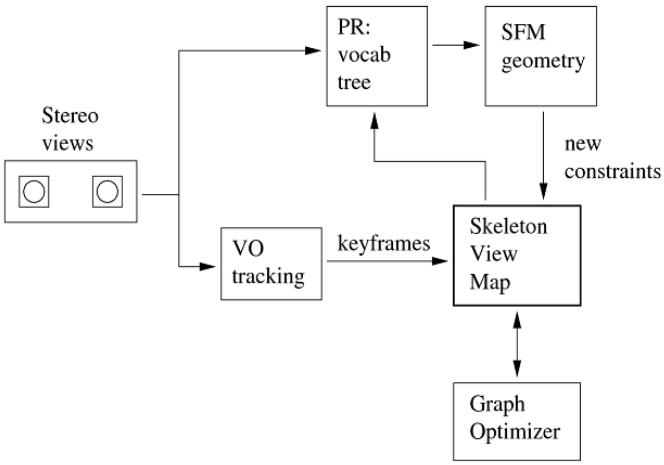


Figure 3: Block Diagram of Konolige and Kurt system, [29].

Although partial map optimization techniques mentioned earlier provides good results in practice, they are not optimal. On the other side, performing global optimization for the whole map repeatedly is computationally expensive for large maps, therefore it was reasoned that optimal estimation of the map necessitates an incremental optimization approach. In [30], the authors devise an exact incremental solution for map optimization. The optimization is reduced to the iterative solution of a linear system in the form **Ax = b.** The solution is based on the fact that matrix **A** is inherently sparse. Matrix **A** can be factorized using Orthogonal Right triangle matrix decomposition (QR factorization) to the product of an orthogonal matrix **Q** and a right triangular matrix **R [**31], with the matrix **R** sparse. As new measurements are added to the system, the linear system is manipulated by adding new variables and using Givens-rotations [31] to recover matrix ….to its triangular form. Using this technique, complete re-linearization of the optimization problem can be avoided for every new measurement although it still should be performed infrequently. In [32], a fluid-relinearization technique was proposed to avoid the costly relinearization of the whole system by linearizing a selected set of variables. This trades small accuracy degradation for much lower computational cost.

III. CONTRIBUTION & OUTLINE

In this paper we design algorithms that yield accurate results within in-door environments were introduced in the literature with errors as small as 10 cm. We develop Visual SLAM algorithms that yield accurate non-drifting pose estimates, because of building and maintaining large maps in out-door environments is still a challenge because of the high computational cost associated with large maps. A new visual-inertial SLAM system will be developed utilizing a mono camera with an IMU. The camera provides a consecutive sequence of images. Each image is searched for distinct image features. These features are tracked throughout the consecutive image frames to estimate the incremental camera pose changes.

The IMU provides measurements for the angular velocity, and acceleration. These measurements are integrated to estimate the relative orientation and displacement between consecutive image frames. The two sensor results are fused together to yield a more accurate result. The proposed system is situated for large out-door environments.

We integrate and evaluate the resulting algorithm on real-world and simulated data. More precisely, the main contribution of this paper is two-fold: (1) we devise a new visual-inertial fusion technique for the sensor combination of mono-camera with an IMU that is based on an optimization approach rather than Extended Kalman Filter (EKF). (2) Another novelty is that we divide the processing into two stages as shown in Figure 4 as follow:

a) Relative Motion Estimation

In this stage, we estimate the motion variables that can be measured directly by observation of the robot relative motion with reference to its surrounding environment; this includes velocity, orientation, and incremental position changes. A novel hybrid visual-inertial odometry is developed to accomplish the task.

b) Absolute motion estimation.

In this stage, the absolute position is estimated. we develop a novel map representation that linearizes the map-optimization process.

Our goal was to design a system in which map-update process (the most computationally expensive process) is an exactly linear operation, therefore, the problem was divided into two stages where the nonlinear nature of SLAM problem was deal with in the first stage enabling a simple linear formulation in the second stage as described in [40].

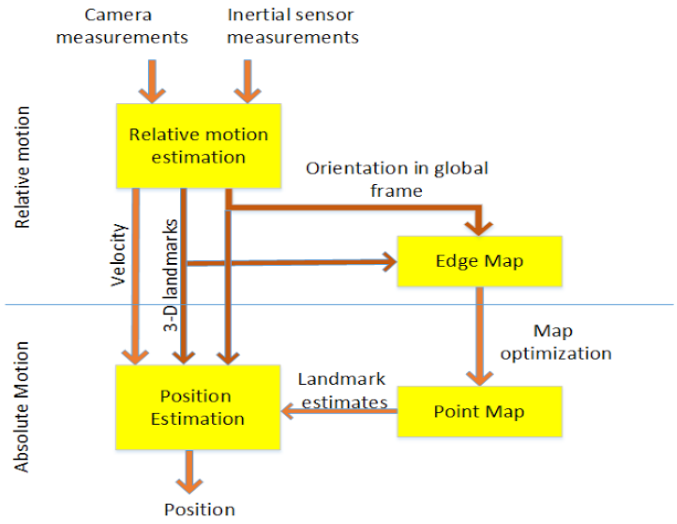


Figure 4: Computational architecture of the proposed SLAM System.

The paper is organized as follows: In chapter II, For Applying SLAM techniques to estimate the pose of a mobile robot involves acquiring measurements from sensors andprocessing them using established techniques to extract useful information, an introductory material for sensor and camera models are provided in this chapter. In chapter III, we introduce a detailed description of our system with its two stages is presented in two sections relative motion estimation section and absolute motion estimation section. Next, we present simulations and real experiment results in chapter IV, and then we evaluate the accuracy, and the robustness for simulated and real-world data. Finally in chapter V, conclusions and Future work are given.

IV. Sensor Modeling and Processing

1) Camera Model

A camera measures the illumination of the surrounding environment. Ideally, each point in the surrounding environment has a corresponding point in the image. The result of a camera measurement is a 2-D matrix of pixels, where each pixel is considered a single point with unique illumination value. We may consider the camera as a projection device that maps points in the world to points in the image. The projection function is not a one-to-one function, alternatively infinitely many points in the world may be projected at the same image point. The projection function of the camera will be developed later in this section. In SFM, we aim to achieve the inverse task i.e. modeling the world using camera measurements which requires performing the inverse mapping, unfortunately, the inverse mapping is not easy because the projection function is not a one-to-one.

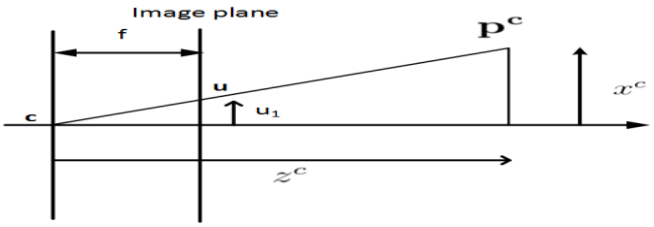
A) Camera Projection Model

a) Pin-hole camera model

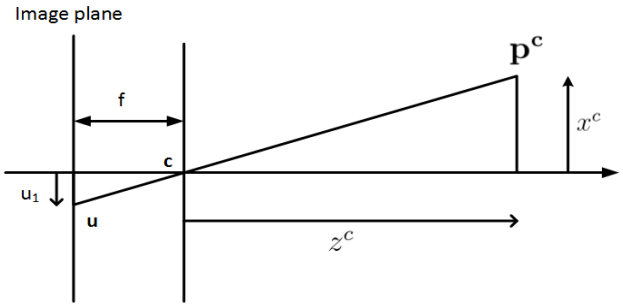
In this model, The camera is modeled as a dark box with infinitesimally small hole that allows light rays to fall on the back plane of the box.

The camera mapping function is termed perspective projection. The relation between the world point $p^w = [x^w \quad y^w \quad z^w]^T$ and its image point $u = [u \quad v]^T$ could be deduced from Figure 5 (a) .Usually the camera pin-hole model depicted in Figure 5

(a) is replaced by the equivalent simpler model depicted in Figure 5 (b) in which the image plane is assumed to lie in front of the camera center.



(a)Real model.



(b) Simplified Model.

Figure 5: Image perspective projection in a pin-hole camera.

Given that the camera center is at c and the focal length (distance between the camera center and image plane) is f, a point $p^c = [x^c \quad y^c \quad z^c]^T$ in camera local frame is projected at a point $u = [u_1 \quad u_2]^T$ in the image plane. We assume that the origin of the camera local frame lies at the camera center, and Z-axis of the camera local frame is the line that passes through the camera center normal to the image plane. the projection equations can be deduced as follows:

$$u_1 = \frac{\mathrm{fx}^c}{z^c} \tag{1}$$

$$u_2 = \frac{\mathrm{fy}^c}{z^c} \tag{2}$$

The homogeneous image plane point is defined by $u^h = [u_1^h \quad u_2^h \quad u_3^h]^T$ where $u_1 = \frac{u_1^h}{u_3^h}$ and $u_2 = \frac{u_2^h}{u_3^h}$, the projection equations (1) and (2) can be written in matrix form as follows:

$$u^h = \begin{bmatrix} \mathrm{fx}^c \\ \mathrm{fy}^c \\ z^c \end{bmatrix} \tag{3}$$

$$u^h = \mathrm{K}p^c$$

$$, K = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{4}$$

where u^h is the homogeneous image plane point and $p^c$is the 3-D local camera frame point. Equation (4) defines the projection function that maps a point in local camera frame p^cinto homogeneous image point u^h. The image plane point $u$image is a dimensionless vector, it represents the tangent of an angle. It is convenient to convert it to pixel units, this means multiplication by some constant to convert angle to pixels. The center of the image may not be at the center of projection, hence

some translation is necessary. So we adjust the form of matrix K that maps local camera frame point $p^c$to homogeneous image point u^h is adjusted as follows:

$$K = \begin{bmatrix} \alpha & \beta & u_0 \\ 0 & \gamma & v_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{5}$$

Matrix K in Equation (5) defines intrinsic camera parameters. These parameters are invariant to the pose of the camera, and can be calibrated easily. A camera is called calibrated if its intrinsic parameters are known. For a point p^w in the world frame, the complete camera projection function could be stated as follows:

$$u^h = K[R_C^{\mathrm{WT}} - R_C^{\mathrm{WT}} t_C^W] \tag{6}$$

where the camera local frame is characterized by rotation matrix R_C^W and t_C^W with respect to the world frame. The pose parameters R_C^W and t_C^W are called extrinsic camera parameters.

Sometimes, we need to determine the locus ray of the local camera frame points projected onto a specific image point, this could be determined using Equation (4) as:

$$p^c = K^{-1} u^h \tag{7}$$

b) Realistic camera models

Pin-hole model is simple, but real cameras utilize lenses which introduces distortion that can be divided into radial and tangential distortion. The effect of radial distortion is shown in Figure 6 , in which straight lines in the image are transformed to curved lines. There are several ways to characterize radial distortion [32], we use the model utilized by OpenCV. Assume there is a world point p^w that has coordinates x^w, y^w, z^w. The projected image point is u. the mapping from p^w to u may be stated as follows:

$$u^h = K[R_C^{\mathrm{WT}} - R_C^{\mathrm{WT}} t_C^W] P^W \tag{8}$$

$$u^d = \begin{bmatrix} \frac{u_1^h}{u_3^h} \\ \frac{u_2^h}{u_3^h} \end{bmatrix} \tag{9}$$

$$u = f_r(u^d) \tag{10}$$

Where, f_r is the radial distortion function, $u^d = [u_1^d \quad u_2^d]$ is the undistorted image plane point and u is the measured image plane point. The radial distortion function f_r is defined as follows:

$$u_1 = u_1^d(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) + 2p_1 u_1^d u_2^d + p_2(r^2 + 2u_2^{d^2}) \tag{11}$$

$$u_2 = u_2^d(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) + 2p_1(r^2 + 2u_2^{d^2}) + 2p_2 u_1^d u_2^d \tag{12}$$

$$r = \sqrt{u_1^{d^2} + u_2^{d^2}} \tag{13}$$

The modified camera model adds new five parameters to the camera parameters namely k_1, k_2, k_3, p_1 and p_2. hence, we need 10 intrinsic parameters plus 6 extrinsic parameters to obtain the camera projection function.

Figure 6: Image distorted by radial distortion (left) versus a perfectly perspective projected image(right).

2) IMU sensor

Inertial measurement unit (IMU) includes:

(1) Three gyroscopes for measuring the angular velocity in three directions. (2) Three accelerometers for measuring the acceleration in three directions. (3) Magnetometer for measuring the direction of the magnetic field.

The usage of magnetometer is sometimes avoided because it may suffer from magnetic interference. The gyroscope and accelerometer measure the angular velocity and acceleration expressed in their body frame. An integrated imu unit usually have all the sensor frames aligned. The coordinate frames are shown in Figure 7 .

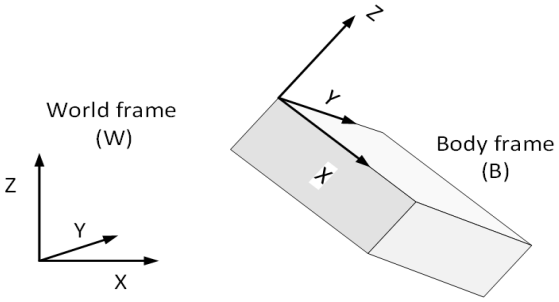


Figure 7: Coordinate frames for the IMU.

Measurements of IMU sensors suffers from:

(1) Axis misalignment: The three orthogonal sensor axes may not be exactly at 90 degree from each other.

(2). Cross talk: It is usually caused by electrical interference between the sensed signals of the three orthogonal axis.

(3) Scaling errors: The measured quantity may be multiplied by some arbitrary scale factor.

(4) Temperature dependence: The scale of the measured quantities vary with temperature.

(5) Noise: The measurements of the sensors are subjected to noise (usually electrical noise).

(6) Interference: Interference with surrounding magnetic fields affects magnetometer.

(7) Bias: An unknown bias is added to the measured signals. This bias changes slowly with time.

Because of the above defects of IMU, it is necessary to calibrate it before usage. IMU units are typically equipped with a processing platform to compensate for temperature variance and help in the calibration process. The IMU calibration process involves finding a 3x3 scale matrix S and a bias vector b for each sensor such that:

$$x_c = \mathrm{S}x_r + b \tag{14}$$

where x_r is the raw measurement, x_c is the calibrated measurement and b is the bias vector. sym:omega_m name=

Assume that the IMU device experiences an arbitrary motion with angular velocity ω and acceleration a. The measured signals are w_m and a_m. The measurement equations for IMU may be stated as follows:

$$w_m = R_w^b \omega + w_b \tag{15}$$

$$a_m = R_w^b a + a_b + R_w^b g \tag{16}$$

Where, $R_w^b$is the rotation matrix that maps world frame (w) to device frame (b),w_m and a_m are gyroscope and accelerometer measurements respectively, w_b and a_b.are gyroscope and accelerometer bias respectively, g is the earth gravity.

3) Visual processing operations

A typical visual odometry system involves: Capture an image from the camera, Extract features from each image , Match new extracted features to the previously extracted features, Estimate relative pose of the new frame with respect to the previous frame, Tune the results using Bundle Adjustment (BA).

3.1) Feature extraction and tracking

Extracting image features is a key-operation for any visual-odometry system. sift is one of the most powerful feature descriptors and works as follows:

a) The image is filtered using the Difference of Gaussian filter M times, where each filtered image is called octave. b) The image is down-sampled by a factor of 2.

c) The image is filtered again and steps 1, 2 are repeated for N times (the number of scales).

d) We obtain a pyramid of Gaussian filtered images.

e) At each scale (each set of consecutive filtered images with the same size) we search for local maxima of the image in spatial (image) space and scale space. The maximum points are considered candidates for a feature location.

f) After all features are detected, a descriptor is built. A SIFT feature descriptor is formed by the calculation of the gradient of the image at feature location, then forming histogram of gradient image. Finally, the descriptor of each feature is a vector of 128 floating point numbers that represent the normalized histogram of image gradient at the feature location.

The next step is to track the image features into the consecutive image frames, this necessitates matching the features of every new frame to the old ones. Two features are considered matched, if their descriptors are similar to a sufficient extent. A matching function is used to compare the two feature descriptors yielding a matching value that will be compared to a threshold to decide if there is a match. The matching function for SIFT is the minimum distance between the two descriptor vectors, and can be described by:

$$f_M(d_1, d_2) = \Sigma_{i=1}^{128}(d_{1_i} - d_{2_i})^2 \tag{17}$$

where d_1: descriptor of the first feature, and d_2 : descriptor of the second feature.

The matching process is subjected to stochastic failures. Two non-matched features may be declared as matched if their descriptors are accidentally similar. This type of error is termed (outlier), since it doesn't follow a noise distribution. It results in a measurement that is not related to the true model.

3.2) Estimation of the relative Pose

RANSAC is usually used for the robust estimation of the relative pose between two camera frames. ransac terminology can be defined as follows: Inlier: A measurement that is consistent with the model (to be estimated) is called an inlier. Inlier measurements may have noise with known distribution, Outlier: A measurement that has no relation with the model is called an outlier, Model: The set of parameters that defines a mathematical model, Model Hypothesis: A set of parameters that are sufficient to describe an assumed model, Scoring function: A function that assigns a score to each pair of a hypothesis and a sample, the score is zero when the sample conforms exactly to the hypothesis, otherwise it provides a value that increases as the deviation of the sample from the hypothesis increases. The scoring value is always clamped to some ceiling threshold.

In RANSAC, we assume many hypothetical models, each of them is called a hypothesis. Each hypothesis is determined using a (minimal set) of samples. If the hypothesis is formed using measurements that are all inliers it is called an inlier hypothesis, otherwise it is called an outlier hypothesis. Each hypothesis is scored against the samples, and a score is assigned to it which is the sum of scores calculated by the scoring function. The hypothesis with the smallest score is the best hypothesis and is chosen to represent the model. In SFM [33], the relative pose between two consecutive frames is the model to be estimated and the matched feature pairs are the samples. The hypotheses are generated by solving a minimal relative pose problem, therefore, we present the minimal relative pose problem solution, then we present the details of RANSAC.

The landmark 3-D location is computed by triangulating its projections in two camera frames with known transformation between them, the problem is challenged by the following issues: 1) the image feature locations are corrupted by noise (usually we assume Gaussian noise of $\pm$ 1 pixel. That means that the rays are not known strictly. 2) The two rays may not intersect in space. 3) When the triangulated point is far from the camera, the two rays are almost parallel, hence the intersection problem is ill-posed. Therefore, to achieve good performance, the triangulation is performed on two steps:

1) An optimal problem is formulated as finding the triangulated points that minimizes the reprojection errors, then the two feature locations that produce this solution are calculated. The reprojection error is defined as the difference between measured feature locations and their reprojected locations as shown in Figure 7 . We use the algorithm of [34].

2) Triangulation of the two feature locations to provide the required space point. A simple linear method as illustrated in [3] can be used for this task.

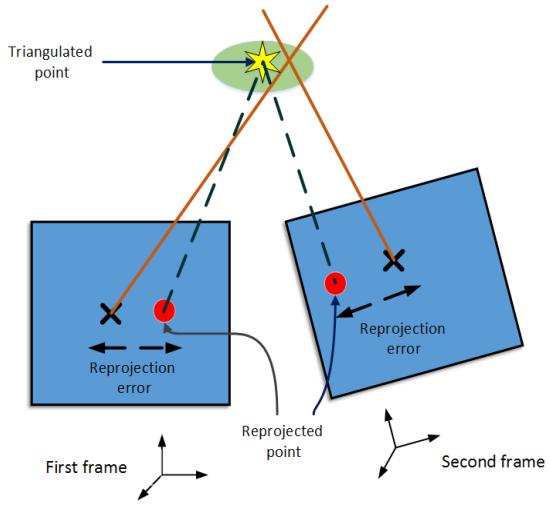


Figure 7: Concept of the reprojection error.

3.3) Bundle adjustment

Visual odometry is pose estimation technique using a camera. It suffers from continuous drift in its estimated pose. The accuracy can be improved by applying a fine tuning stage called bundle-adjustment. Bundle-adjustment is an optimization process that results in optimal estimations of the robot pose and the landmark positions. The bundle adjustment minimizes the objective function defined as the mean square error of reprojection errors as in Equation (18).

$$E = \frac{1}{2}\sum_{i=1}^{2}\sum_{j=1}^{N} \mathrm{P}u_{ij} - \mathrm{G}(\mathrm{P_i}, \mathrm{X_j})\mathrm{P}_{\Sigma_{uij}^{-1}}$$

$$= \frac{1}{2}\sum_{i=1}^{2}\sum_{j=1}^{N} (u_{ij} - \mathrm{G}(\mathrm{P_i}, \mathrm{X_j}))^T \Sigma_{uij}^{-1} (u_{ij} - \mathrm{G}(\mathrm{P_i}, \mathrm{X_j})) \quad (18)$$

In Equation (19), we have two frames and N landmarks that have projections in both frames. u_{ij} is the projection of landmark j onto frame i, while G is the projection function that maps a landmark X_j into frame i. The pose P_i is the pose of frame i, it includes both orientation and translation of frame i. In this optimization problem, u_{ij} are considered to be the measurements that have uncertainty given by covariance matrix Σ_{uij}. u_{ij} is a 2-D vector that represents the location of a detected image feature inside the image plane, it has the units of pixels, its 2×2 uncertainty matrix Σ_{uij} is taken as unity matrix (i.e. the uncertainty is taken as 1 pixel). we assume u_{ij} is the mean of a Gaussian random variable with covariance matrix Σ_{uij}, even so this is not exactly true because u_{ij} is discrete rather than continuous random variable. With the assumption of u_{ij} is a Gaussian random variable, it follows that equation (19) is the optimal unbiased maximum likelihood estimator for P_i and X_j according to the estimation theory. The variables to be optimized are P_i and X_j, it should be mentioned that the projection function G is strongly nonlinear, hence the optimization problem has many minima, hence we should have good initial solutions for X_i and P_i, the previous stage of RANSAC and triangulation provide such good initial solutions.

Equation (19) involves the pose of two frames, but we can estimate only the relative pose between the two frames. This means the estimated pose of the two frames will be floating with the relative pose fixed. To alleviate this problem, we should fix the pose of one of them. We choose to fix the first frame's pose. We can simply assume any pose for the starting frame such as zero position and identity orientation. The problem still has floating parameter, the scale of translation between the two frames is floating as it was mentioned previously that the scale can't be recovered from images captured by a single camera. To improve the accuracy of the estimation, we can optimize over many frames not just one. Equation (19) can be modified as follows:

$$E = \frac{1}{2}\sum_{i=1}^{L}\sum_{j=1}^{N} \mathrm{P}u_{ij} - \mathrm{G}(\mathrm{P}_i, \mathrm{X}_j)\mathrm{P}^2_{\Sigma_{uij}^{-1}}$$

$$= \frac{1}{2}\sum_{i=1}^{L}\sum_{j=1}^{N} (u_{ij} - \mathrm{G}(\mathrm{P}_i, \mathrm{X}_j))^T \Sigma_{uij}^{-1} (u_{ij} - \mathrm{G}(\mathrm{P}_i, \mathrm{X}_j)) \quad (19)$$

Now, we define the optimization function to include the last L frames, we add each new captured image to the set of optimized frames.

The bundle adjustment could be solved using standard non-linear optimization techniques such as Levenberg-Marquardt algorithm. The solution is obtained by the iterative solution of a linear system of the form:

$$H\delta_x = B \quad (20)$$

Where:

$$H = J^T \Sigma^{-1} J + \lambda I$$

$$B = J^T \Sigma^{-1} (x_m - G(x))$$

$$J = \left[\frac{\partial G}{\partial \delta_P} \quad \frac{\partial G}{\partial \delta_X}\right]$$

where δ_P is the incremental pose adjustment and δ_X is the incremental landmark position adjustment. The jacobian J is sparse and this enables very efficient solution for the optimization problem.

V. REALATIVE AND ABSUOLUTE MOTION SUBSYSTEMS

a) RELATIVE MOTION ESTIMATION SUBSYSTEM

In the relative motion estimation stage, we built a visual-inertial odometry system that fuses the camera image measurements and inertial sensor measurements to provide accurate incremental orientation/position estimates. Relative motion estimation stage is implemented in two ways. Filtering approach is presented first where the inertial and visual measurements are fused in the framework of EKF estimation. Then, the more elaborate implementation using an optimization approach on a window of frame poses is presented described in [41].

b) Relative Motion Using Optimization technique

The processing of measurements flows in two parallel pipelines: 1) The visual processing pipeline: Typical visual odometry image processing techniques is employed to extract features from image frames, then calculate initial estimates of incremental pose changes. 2) The inertial data processing pipeline: The IMU measurements are integrated to provide incremental orientation, velocity, and displacement measurements. Finally, the two pipelines are merged in a fusion stage. The detailed system pipeline is shown in Figure 8 to comprise of the following steps:

(a) Capture an image from the camera (each image should be captured with a time stamp).

(b) Receive IMU measurements (each measurement has a timestamp).

(b) Extract the features from each image.

(c) Match new extracted features to previously extracted features (Feature tracking).

(d) Utilize RANSAC to eliminate outliers and establish initial motion estimation.

(e) Calculate scale of the new frame translation by utilizing known points in the scene.

(f) Integrate IMU measurements to determine orientation, incremental velocity, and incremental position.

(g) Fuse camera measurements and IMU measurements to obtain enhanced orientation and position.

The system keeps track of a window of L key-frames as shown in Figure 9 . Only the last M frame poses are subject to optimization, while the first $L-M$ frames are fixed. We keep track of the measurements of the L key-frames that includes measured image feature locations, and the integrated IMU measurements in the period between each frame and the previous one. These measurements are fused together through an optimization module to yield the pose and velocity estimates of the robot at the time instant of each frame. In our work, we use tight coupling approach. The IMU predictions are fused with visual data in an optimization stage. A new optimization function is defined that includes visual and inertial measurements. The inter-frame IMU integrated pose predictions are reset at the beginning of each new camera frame.

We can calculate an initial estimation for the relative pose between two consecutive frames using matched features in both images up to an unknown scale of the translation t. It is well known that the scale cannot be recovered from consecutive images from a single camera. It follows that the triangulated points are also known within unknown scale factor.

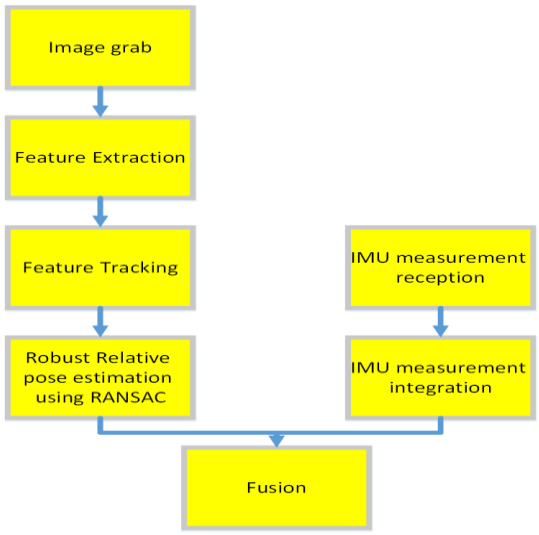


Figure 8: Processing pipeline in the relative motion estimation module.

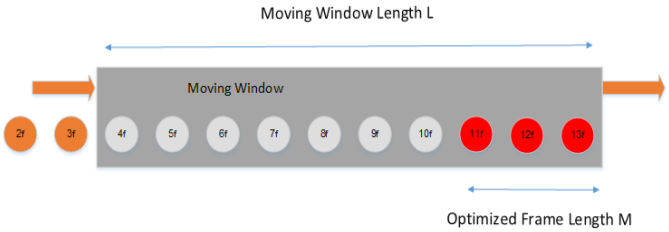


Figure 9: Processing window in relative motion estimation.

The system has two modes, initialization mode and hybrid mode. In the initialization mode, the visual measurements only are used for the pose estimation, and the inertialmeasurements are compared to the visually estimated state to estimate the motion scaleas shown in Figure 10. Once the scale is known with good certainty, the system switchesto the hybrid mode where both the visual and inertial measurements are used in the poseestimation as shown in Figure 11. In both modes, an online gyroscope bias estimationmodule is used to estimate gyroscopic bias.An initial estimation for the relative pose between two consecutive frames can be estimated using matched features in both images up to an unknown scale of the translation t. It is well known that the scale cannot be recovered from consecutive images from a single camera. It follows that the triangulated points are also known within unknown scalefactor. To cope up with this limitation, we set the scale of translation between the first two frames arbitrarily to unity. This fixes the scale of the translation and triangulated points or in other words fixes the scale of our estimated world. Regarding the next frames, we can't put an arbitrary scale as we did in the first two frames, so we have to estimate the true scale based on our previous knowledge of world points and the preceding frames, therefore we propose a new algorithm for the estimation of the relative scale of the new frames.

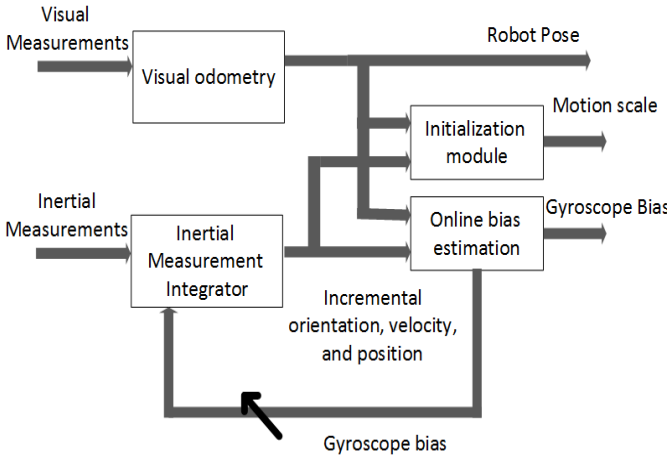


Figure 10: structure of relative motion module in the initialization mode.

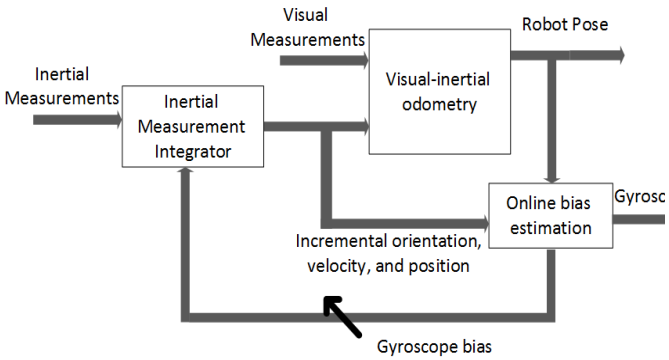


Figure 11: Structure of relative motion module in the hybrid mode.

We assume that we have two measurements ω_m and a_m which are the measured angular velocity and acceleration respectively. The model of the IMU was introduced in chapter II. Σ_ω and Σ_a are the uncertainty of the two measurements, they are usually stated in sensor's datasheet. The result of integration process is to yield the incremental orientation q, incremental position x and incremental velocity v. Besides these three variables, we include bias variables to accommodate the bias uncertainty effect, hence we add ω_b and a_b variables, we assume that we know initial uncertainty Σ_{ω_b} and Σ_{a_b} of ω_b and a_b respectively. The integration interval is from t_{k-1} to t_k. The values of incremental velocity v and incremental position x are reset to zero and the orientation q are reset to identity at the time instant of each new frame. The IMU integration formula may be stated as follows:

$$t_i R(t)(a_m - a_b)\mathrm{dt} \tag{21}$$

$$t_i v\mathrm{dt} \tag{22}$$

$$\frac{\partial q}{\partial t} = q * \begin{bmatrix} 0 \\ \frac{1}{2}[\omega_m - \omega_b]_X \end{bmatrix} \tag{23}$$

$$\omega_b = \text{constant}$$

$$a_b = \text{constant}$$

where $R = \text{Quaternion}[q]$ is defined in [37], and $[\omega]_X$ is defined as:

$$[\omega]_X = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix}) \tag{24}$$

Unfortunately, there is no simple direct formula for integration of angular velocity, alternatively we have a differential Equation (23) that describes the relation between q and ω_m. The angular velocity bias and acceleration bias are unchanged during the integration process.

In reality, the IMU measurements are discrete and hence we should discretize the integration process. There are many approximations to the integration, we use a second order formula based on interpolation of integrand. Assume that $t_k g(t)$dt, the discrete function $f(k)$ is given by [31]:

$$f(k) = f(k-1) + \text{df}$$

$$\text{df} = \text{INT}(g(k), g(k-1), g(k-2); t_k, t_{k-1})(24)$$

$$\text{df} = t_k(g(k) + g(k-1))/2 - t_k^2(g(k) - g(k-2))/(6(t_k + t_{k-1})) + t_k^2(g(k-1) - g(k-2))/(6t_{k-1}) \tag{25}$$

Where t_i and t_{i-1} are the time difference between samples $g(k)$ and $g(k-1)$ and between $g(k-1)$ and $g(k-2)$ respectively. We use Equation (46) for approximating the integrals in Equations (21) and (22). Integration of the orientation is done as follows:

$$q(k) = q(k-1) * \text{dq} \tag{26}$$

$$\text{dq} = \text{Quaternion}[d\Theta] \tag{27}$$

$$d\Theta = \text{INT}(\omega_m(k), \omega_m(k-1), \omega_m(k-2); t_i, t_{i-1}) \tag{28}$$

Equation (26) multiplies old quaternion $q(k-1)$ by the difference quaternion dq, Equation (28) gives the difference $d\Theta$, and we can convert the difference orientation from angle-axis representation to quaternion representation. The previous equations assume that $d\Theta$ is small such that the approximate formulas become reasonably accurate. We can calcute the covariance matrix of q , x , v ,ω_m and a_m using state space representation fot our system.

Using tight coupling approach, we define a hybrid optimization function that contains both inertial and visual measurements as follows:

$$E = \frac{1}{2}\varepsilon_V^T \Sigma_U^{-1} \varepsilon_V + \frac{1}{2}\varepsilon_I^T \Sigma_I^{-1} \varepsilon_I + \frac{1}{2}\varepsilon_N \Sigma_N^{-1} \varepsilon_N \tag{29}$$

Following the windowed visual odometry paradigm, we keep track of the last L frames, and optimize the pose of the most recent M frames. We add the velocity v_i as a new variable to be estimated at every frame i. The optimization function consists of three parts, the first part is related to visual measurements. The second part is the inertial error weighted by Σ_I^{-1}.

In hybrid inertial-mono camera visual odometry, the translation scale, velocity and gravity are weakly observed variables. Scale and velocity are not observable at all during constant-velocity motion scenarios. This is because measured acceleration is zero and doesn't provide any scale information. Therefore, knowledge gained during rich dynamic motion should be accumulated and saved. A prior knowledge of velocity and gravity is saved as a prior vector $N_{\text{nom}} = [v_{\text{nom}}^T \quad g_{\text{nom}}^T]^T$ with its uncertainty matrix Σ_N. v_{nom} represents prior knowledge of v_{L-M-1} and g_{nom} represents prior knowledge of the gravity vector g. The prior information is updated after the addition of every new key-frame and represented in the optimization function by the third term. We focus on the second and third part of the optimization function, let's define the inertial error and prior error:

$$\varepsilon_I = I_m - I = \begin{bmatrix} \varepsilon_{I_1}^T & \varepsilon_{I_2}^T & \dots & \varepsilon_{I_M}^T \end{bmatrix}^T \tag{30}$$

$$\varepsilon_{I_i} = I_{\text{mi}} - I_i = d_{m_i} - R_{i-1}^{w^T}(-v_{i-1}\text{dt}_i + s(t_i - t_{i-1}) + \frac{1}{2}g\text{dt}_i^2) \\ v_{m_i} - R_{i-1}^{w^T}(v_i - v_{i-1} + g\text{dt}_i) \tag{31}$$

$$\psi_{e_i} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} q_{e_i} \tag{32}$$

$$q_{e_i} = q_{m_i} * q_i \ * q_{i-1} \tag{33}$$

$$\varepsilon_N = \begin{bmatrix} v_{\text{nom}} - v_{L-M-1} \\ g_{\text{nom}} - g \end{bmatrix} \tag{34}$$

Equation (31) defines the error term ε_{Ii} of the inertial part that belongs to frame i. q_{m_i}, v_{m_i} and d_{m_i} are the incremental orientation, velocity and position , dt_i is the time difference between frame $i+1$ and frame i and g is the gravity vector. q_i, t_i and v_i are the orientation, position and velocity at each frame referenced to world frame. Although the variables q_i, t_i, and v_i are sufficient to represent all motion that could be estimated, we have to estimate also the gravity vector g since its direction is not known at the beginning.

We use Levenberg Marquardt optimization algorithm (LM) for the parametrization *for pose P_i to minimize error E*. The velocity and gravity are trivially represented as 3-D vector. LM algorithm is used for optimizing Equation (29), we calculate the jacobian and then form the normal equation and solve it iteratively in the same manner we did during bundle adjustment. We define the jacobian$J$ and augmented state Γ as follows:

$$\Gamma = [P_0^T .....P_{M-1}^T X_0^T .....X_{N-1}^T V_0^T .....V_{M-1}^T g^T]^T \tag{35}$$

$$\varepsilon_T = [\varepsilon_V^T \varepsilon_1^T \varepsilon_N^T]^T \quad , \tag{36}$$

$$J = \begin{bmatrix} \frac{\partial G}{\partial P} & \frac{\partial G}{\partial X} & 0 & 0 \\ \frac{\partial I}{\partial P} & 0 & \frac{\partial I}{\partial V} & \frac{\partial I}{\partial g} \\ 0 & 0 & \frac{\partial N}{\partial V} & \frac{\partial N}{\partial g} \end{bmatrix} = \frac{\partial \varepsilon_T}{\partial \Gamma} \tag{37}$$

Once, we obtain jacobian, we can derive the normal equation.

$$H\delta = B \tag{38}$$

$$H = \begin{bmatrix} U & W^T & K^T & R^T \\ W & V & 0 & 0 \\ K & 0 & L & T^T \\ R & 0 & T & Z \end{bmatrix} \tag{39}$$

$$B = \begin{bmatrix} \text{B1} \\ \text{B2} \\ \text{B3} \\ \text{B4} \end{bmatrix} \tag{40}$$

The matrices W and V are defined in the same fashion as their counterparts . We can also express the rest of jacobians.

1. Absolute Motion Estimation Subsystem

1.1. Map Representation

The absolute motion estimation subsystem is the 2nd stage responsible for maintaining a map for the surrounding environment and localizing a robot with respect to it. The estimated relative motion variables resulting from the first stage (Realative Motion Subsystem) are used as input to the second stage. Those variables are incremental pose changes δt and landmark positions y_i referenced to local robot frame.

The landmark measurements y_i are 3-D locations referenced to the local frame with their uncertainty expressed as covariance matrices Σ_{yi}, we convert the point cloud of landmarks to a group of edge measurements y_{ij}^e given by equation (41) by subtracting the locations of any two landmarks. The uncertainty of each edge measurement Σ_{yij} is obtained easily from the uncertainties of the two landmark locations as given by equation (42) as follows.

$$y_{ij}^e = y_j - y_i \tag{41}$$

$$\Sigma_{yij} = \Sigma_{yi} + \Sigma_{yj} \tag{42}$$

It is assumed that the orientation is estimated in the first stage with sufficient accuracy. Excluding orientation from our estimation model turns the estimation problem into a simple linear one. This is beneficial since most of the the processing time is spent in estimation of the map structure. Linearization of the absolute motion problem will help in building an efficient algorithm for solving it. We use a sparse map representation which is useful for the purposes of localization. The map consists of point features. We attach a descriptor to each point feature so that it can be identified later.

The two problems of localization and map building should be solved simultaneously because there is a dependency between absolute landmark positions and the robot pose. This dependency stems from the fact that estimated landmark positions depends on the robot pose. This situation is illustrated in Figure 12 . The absolute position of the shown landmark in Figure 12 is L_w, this position is not directly observable from the camera. The camera measures only the landmark position relative to its pose P_1 or P_2, that is L_{l_1} or L_{l_2}. It is obvious that the measurement is different from different camera poses; hence camera measurement is not sufficient for the determination of the location of the landmark, and we should also know the pose of the robot. The map is parametrized in a novel way. Instead of representing the map as a (point cloud) of landmarks, we represent the map as graph of edges and vertices where the vertices are the landmark positions and the edges are the displacement between them as shown in

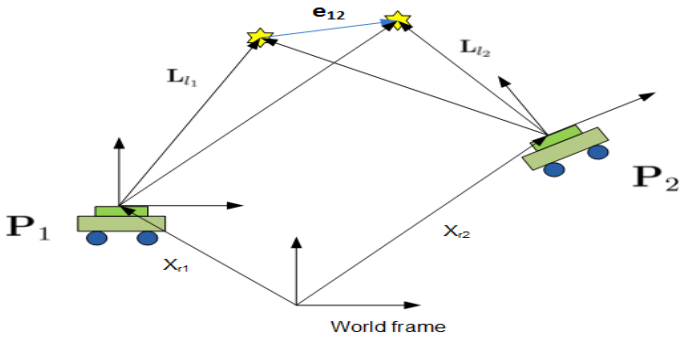


Figure 12: Robot poses relation graph.

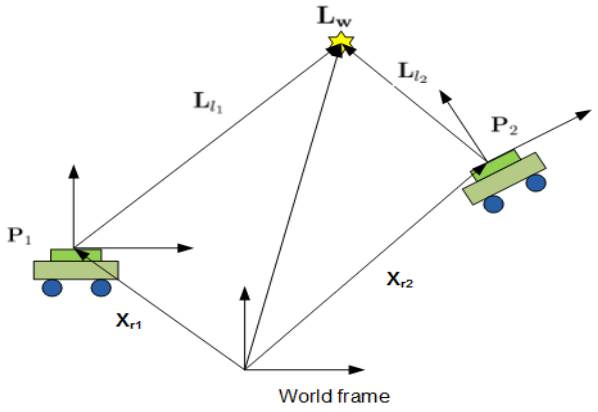


Figure 13: Dependency of absolute landmark position on robot pose

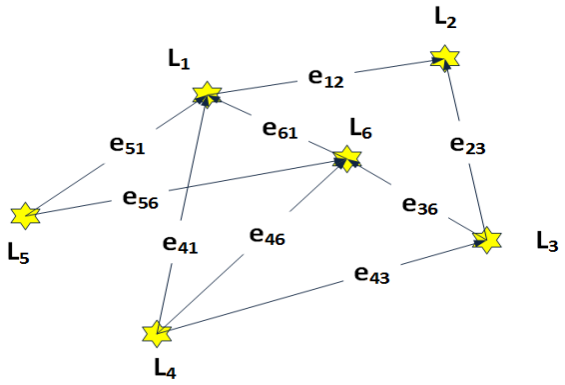


Figure 14: Map represented as a graph.

This representation is illustrated in Figure 13. We can imagine that there is an observer that watches edge e_{12} from pose P_1, and another observer that watches the same edge from pose P_2. The two observers will agree on the edge length. but they will disagree on the edge orientation because each of them has different pose. Also note that the edge vector doesn't depend on its absolute position. Hence the dependency of the edge is on robot orientation only and it doesn't depend on the robot location. Since we estimated robot orientation in the relative motion estimation stage, we are able to know the edge vector in the absolute frame. This representation effectively decouples the problem of robot pose estimation from the map structure estimation.

New map landmarks $L_i$are added to the map as new landmark measurements are submitted to the absolute motion module. New edge variables symb:eij are created as necessary. The edge variables are considered 3-D random vectors with Gaussian distribution with mean $\overline{e_{ij}}$ and covariance matrix Σ_{ij}. For each newly created edge, the edge is initialized to have a mean of $\overline{e_{ij}} = \overline{y_{ij}}$ and a covariance matrix $\Sigma_{ij} = \Sigma_{yij}$. In later sections, we will use the notation e_{ij} to describe edge mean for simplicity.

As new edge measurementsy_{ij} are submitted to the absolute motion module, the edge variable e_{ij} should be updated. Given that the edge variable e_{ij} and the measurement y_{ij} are Gaussian random variables, and using maximum likelihood criteria for the probability density function $P(e_{ij}|y_{ij})$ of the edge e_{ij} given the measurement y_{ij} , we can state the update equation for e_{ij} as follows:

$$e_{ij}(k+1) = (\Sigma_{ij}(k)^{-1} + \Sigma_{yij}(k)^{-1})^{-1}(\Sigma_{ij}(k)^{-1}e_{ij}(k) + \Sigma_{yij}(k)^{-1}y_{ij}(k))$$
$$\Sigma_{ij}(k+1) = (\Sigma_{ij}(k)^{-1} + \Sigma_{yij}(k)^{-1})^{-1} \quad (43)$$

Each edge measurement will be utilized to update the edge map, this indicates that all measurements will be employed without need to discard any of them. The edges are considered statistically independent, therefore each edge measurement affects only one edge variable thus simplifying the map update process considerably. The edge map conveys all the information about the map, but it is still inconsistent and missing the locations of the landmarks. To obtain the landmark locations, an optimization criteria is defined that minimizes the inconsistency error of the edge map. The optimization process needn't be performed at every new captured frame, it can be performed at distant infrequent times. On the other hand, the robot position is estimated with the aid of the map at every new captured frame.

1.2. Optimization Process

. The process of optimization is highlighted in Figure 15 described in [40]. There is an offset ambiguity in the edge map. To resolve this ambiguity we should choose a reference point, a point that its location is certainly known. We define the inconsistency error err_{ij} that results from erroneous edge value e_{ij} that connects L_i to L_j as follows:

$$\text{err}_{ij} = L_j - L_i - e_{ij} \tag{44}$$

It is sufficient to consider the absolute error $|\text{err}_{ij}|$. But since uncertainty in edge vector represented by its covariance matrix Σ_{ij} varies from edge to another we define edge vector normalized error nErr_{ij} as follows:

$$\text{nErr}_{ij} = \text{err}_{ij}^T \Sigma_{ij^{-1}} \text{err}_{ij} = (L_j - L_i - e_{ij})^T \Sigma_{ij^{-1}} (L_j - L_i - e_{ij}) \tag{45}$$

For simplicity let's set $S_{ij} = \Sigma_{ij^{-1}}$. The matrix S_{ij} represents the amount of information or certainty in the edge vector. Now, we can write nErr_{ij} as

$$\text{nErr}_{ij} = (L_j - L_i - e_{ij})^T S_{ij} (L_j - L_i - e_{ij}) \tag{46}$$

Finally, we are now able to define the total map error E as the sum of all edges inconsistency errors as follows:

$$E = \frac{1}{2} \Sigma_{i=1}^{N_l} \Sigma_{j=1, j\neq i}^{N_l} (L_j - L_i - e_{ij})^T S_{ij} (L_j - L_i - e_{ij}) \tag{47}$$

Equation (70) is the objective function to minimize. Edge vectors e_{ij} are known. So it is required to optimize against landmark variables L_i, i=1, 2, ..., N_l. We differentiate the total error E with respect to $L_k$as follows:

$$\frac{\partial E}{\partial L_k} = -\Sigma_{j=1, j\neq k}^{j=N_l} (L_j - L_k - e_{kj})^T S_{kj} + \Sigma_{i=1, i\neq k}^{i=N_l} (L_k - L_i - e_{ik})^T S_{ik} \tag{48}$$

i, j are just dummy variables so that we can write the previous equation as follows:

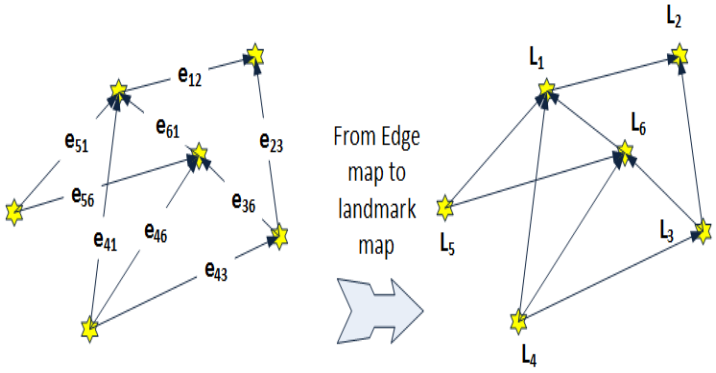


Figure 15: Schematic of map optimization process.

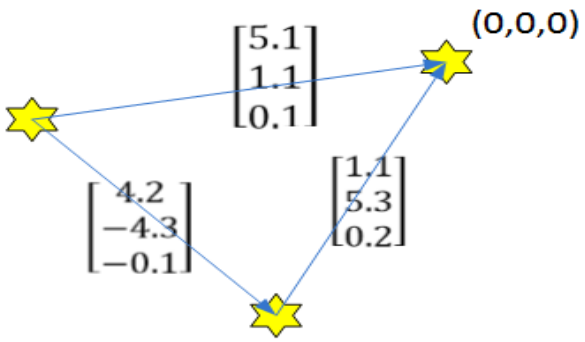


Figure 16 : Inconsistency of the edge map.

$$\frac{\partial E}{\partial L_k} = \Sigma_{i=1,i\neq k}^{i=N_l}(-(L_i - L_k - e_{\mathrm{ki}})^T S_{\mathrm{ki}} + (L_k - L_i - e_{\mathrm{ik}})^T S_{\mathrm{ik}}) = \Sigma_{i=1,i\neq k}^{i=N_l}((L_k - L_i + e_{\mathrm{ki}})^T S_{\mathrm{ki}} + (L_k - L_i - e_{\mathrm{ik}})^T S_{\mathrm{ik}}) \quad (49)$$

But we have $S_{\mathrm{ki}} = S_{\mathrm{ik}}$ since it represents the same edge. Also we have $e_{\mathrm{ki}} = -e_{\mathrm{ik}}$, because reversal of edge indices represents reversal of edge direction. So we could write the previous equation as foolows:

$$\frac{\partial E}{\partial L_k} = 2\Sigma_{i=1,i\neq k}^{i=N_l}(L_k - L_i - e_{\mathrm{ik}})^T S_{\mathrm{ki}} \quad (50)$$

Equalizing the differential equations to zero, we get:

$$\Sigma_{i=1,i\neq k}^{i=N_l}(L_k - L_i - e_{\mathrm{ik}})^T S_{\mathrm{ki}} = 0$$

$$L_k^T \Sigma_{i=1,i\neq k}^{i=N_l} S_{\mathrm{ki}} - \Sigma_{i=1,i\neq k}^{i=N_l} L_i^T S_{\mathrm{ki}} = \Sigma_{i=1,i\neq k}^{i=N_l} e_{\mathrm{ik}}^T S_{\mathrm{ki}} \quad (51)$$

Taking the transposition of both equations:

$$\left(\Sigma_{i=1,i\neq k}^{i=N_l} S_{\mathrm{ki}}\right) L_k - \Sigma_{i=1,i\neq k}^{i=N_l} S_{\mathrm{ki}} L_i = \Sigma_{i=1,i\neq k}^{i=N_l} S_{\mathrm{ki}} e_{\mathrm{ik}} \quad (52)$$

The last equation could be written in matrix form as:

$$\begin{bmatrix} -S_{k1} & -S_{k2} & \dots & (\Sigma_{i=1,i\neq k}^{i=N_l} S_{\mathrm{ki}}) & \dots & -S_{\mathrm{kN}_l} \end{bmatrix} \begin{bmatrix} L_1 \\ L_2 \\ \dots \\ L_k \\ \dots \\ L_{N_l} \end{bmatrix} = \Sigma_{i=1,i\neq k}^{i=N_l} S_{\mathrm{ki}} e_{\mathrm{ik}} \quad (53)$$

Then we differentiate error E with respect to k=1, 2, ..., N_l, we will formulate in matrix form as follows:

$$\mathrm{AL} = B \quad (54)$$

Where$A$, L, and B are defined by:

$$L = \begin{bmatrix} L_1 \\ L_2 \\ \dots \\ L_{N_l} \end{bmatrix} \quad (55)$$

$$A = \begin{bmatrix} (\Sigma_{i=1,i\neq 1}^{i=N_l} S_{1i}) & -S_{12} & \dots & -S_{1N_l} \\ -S_{21} & (\Sigma_{i=1,i\neq 2}^{i=N_l} S_{2i}) & \dots & -S_{2N_l} \\ \dots & & & \\ -S_{N_l 1} & -S_{N_l 2} & \dots & (\Sigma_{i=1,i\neq N_l}^{i=N_l} S_{N_l i}) \end{bmatrix} \tag{56}$$

$$B = \begin{matrix} \Sigma_{i=1,i\neq 1}^{i=N_l} S_{1i} e_{i1} \\ \Sigma_{i=1,i\neq 2}^{i=N_l} S_{2i} e_{i2} \\ \dots \\ \Sigma_{i=1,i\neq N_l}^{i=N_l} S_{N_l i} e_{iN_l} \end{matrix} \tag{57}$$

Equations (55) to (57) are the solution to the optimization problem as expressed in Equation (47). *These results show that map optimization is equivalent to solving a linear system of equations*, So the proposed system is truly linear without approximation and has the following properties : (1) Matrix $A$is symmetric: This is due to the fact that S_{ij} is a symmetric matrix, hence the off diagonal elements of A will be the same. (2) Matrix A is singular with nullity = 3 : This is true because any sub matrix row in matrix A is the negative sum of all other sub matrix rows. (3) Matrix A is positive semidefinite. Because matrix A is symmetric and positive definite, it satisfies the conditions of covariance matrix. We will highlight later the fact that A matrix is the information matrix of the map.

Equation (54) will yield the mean estimation of landmark positions. But, we need also to determine the covariance matrix Σ_L of the landmarks. This is needed for estimation of the robot position.

The covarianceΣ_L is defined as:

$$\begin{aligned} \Sigma_L &= E[(L-\overline{L})(L-\overline{L})^T] \\ &= E[\mathrm{LL}^T] - \overline{L}^2 \end{aligned} \tag{58}$$

An expression is derived for Σ_L where it is shown to be:

$$\Sigma_L^{-1} = \begin{bmatrix} A_{11}^{-1} & 0_{(3N_l-3)x3} \\ 0_{3x(3N_l-3)} & 0_{3x3} \end{bmatrix} \tag{59}$$

Equation (59) gives the expression of Σ_L, where A_{11} is the submatrix of matrix A including all rows and columns and excluding the row and column of the reference landmark. Notice that Σ_L is the inverse of a submatrix of A matrix. So we may consider matrix A as the information matrix.

1.3. Estimation of The Robot Position

The robot position X_r can be estimated by the accumulation of incremental position changes from the relative motion module, then utilizing the observed relative landmarks to correct the position estimate as shown below:

$$X_r(k+1) = X_r(k) + \delta t + n(k) \tag{60}$$

$$y_i(k+1) = L_i(k+1) - X_r(k+1) + v_i(k) \tag{61}$$

Where:

$X_r(k)$: Robot position.

$e_{ij}(k)$: Edge vector between two landmarks i,j.

$L_i(k)$: The position of landmark.

δt: Incremental Position change from the relative motion module.

$y_i(k)$: Landmark measurement from the relative motion module.

n: Robot position prediction noise.

v_i: Landmark i measurement noise.

Kalman filter is used for state estimation. Kalman filter represents the estimated state vector as gaussian random vector with mean X_r and covariance matrix Σ^x. The mean and covariance matrix are propagated from one time step to the next using two steps, prediction then correction. We assume in this section that X_r and Σ^x are known. Initially, we can assume X_r to equal the origin with complete confidence.

(1) Robot position

Let R be the covariance matrix of the noise input n. We can write the Kalman prediction equations as follows:

$$\overline{X_r}(k+1) = X_r(k) + \delta t \tag{62}$$

$$\overline{\Sigma^r}(k+1) = \overline{\Sigma^r}(k) + R \tag{63}$$

(2) Robot position correction

In the correction step we will consider the robot position X_r as the state vector with covariance matrix $\overline{\Sigma^r}_{k+1}$. The robot observes Mlandmarks only from N_l total landmarks. The measurement equation could be stated as follows:

$$y_i(k) = L_j - X_r(k) \; i=1,2,...,M \text{ and } j=1,2,...,N_l \tag{64}$$

$$L_i - y_i(k) = X_r(k) \; i=1,2,...,M \text{ and } j=1,2,...,N_l \tag{65}$$

Equation (88) is the form of measurement equation suitable for Kalman filter, we have M measurement equations. It is convenient to write Equation (65) in matrix form. Let's define the landmark vector $L = \begin{bmatrix} L_1^T & L_2^T & \dots & L_{N_l}^T \end{bmatrix}^T$ with covariance matrix Σ_L and the measurement vector $y(k) = [y_1(k)^T \quad y_2(k)^T \quad \dots \quad y_M(k)^T]^T$ with covariance matrix $Q(k)$ where:

$$Q(k) = \begin{bmatrix} Q_1(k) & 0 & \dots & 0 \\ 0 & Q_2(k) & \dots & 0 \\ . & . & . & . \\ 0 & 0 & \dots & Q_M(k) \end{bmatrix} \tag{66}$$

Let's define the vector L_v to contain the visible subset of landmarks L_i. To simplify notation, let's define matrix M such that $L_v = \text{ML}$. Now, it is possible to define the total measurement $y^t = L_v - y = \text{ML} - y$ with covariance matrix $Q^t = Q + M\Sigma_L M^T$. Now, the Kalman filter correction equations are:

$$X_r(k+1) = \overline{X_r(k+1)} + K(k)\left(y^t(k+1) - y_o^t(k)\right) \tag{67}$$

$$\Sigma^r(k+1) = (I - K(k)H)\overline{\Sigma^r}(k+1) \tag{68}$$

$$K(k) = \overline{\Sigma^r}(k+1)H^T(H\overline{\Sigma^r}(k+1)H^T + Q^t(k))^{-1} \tag{69}$$

$$H = \frac{\partial y^t(k)}{\partial X_r(k)}$$

$$= [I \quad I \quad \dots \quad I]^T \tag{70}$$

$$y_o^t = H\overline{X_r(k+1)} \tag{71}$$

Where:

y_o: Expected measurement vector based on state X_r.

H: Measurement Jacobian matrix.

K: Kalman gain.

The covariance term $M\Sigma_L M^T$ is the trickiest to calculate because it requires the knowledge of the map landmarks covariance matrix Σ_L. Naturally, we have the information matrix Σ_L^{-1} which can be calculated from the edge information as shown in Equation (56). Determination of Σ_L requires an inverse operation of the whole map information matrix which is computationally expensive for large maps. An approximation can be performed by noticing that the matrix $M\Sigma_L M$ is simply a sub-matrix of Σ_L which contains the rows and columns pertaining to visible landmarks only. Therefore, a reasonable approximation is to let $M\Sigma_L M^T = (M\Sigma_L^{-1}M^T)^{-1}$. A sub-matrix of Σ_L^{-1} is obtained first, then the inversion is done on this sub-matrix. This approximation is exact when the visible and invisible landmarks are uncorrelated, otherwise it will omit the effect of invisible landmarks on the robot position.

VI. Experiments and Simulations

The relative motion module of the proposed system was implemented using C++ on a PC with 3 GHz processor, 4 GB memory, and Nividia Quadro 400 video card to evaluate its accuracy and performance. The system can be run in real-time. It was evaluated with custom-collected data sets and data sets downloaded from the internet. The system was implemented from scratch, our software implementation contains the following main components are :Feature extractor , Feature tracker, Five point problem solver, RANSAC robust estimator, and Bundle-adjustment and sensor fusion module.

Third-party libraries were used to perform feature extraction, we evaluated many feature descriptors like ORB, SURF, and SIFT. The ORB descriptor is the fastest while SIFT is the most reliable but computationally expensive. SIFT feature extractor implemented using OpenCV processes a 1024x768 image in more than 1 s. The performance can be improved by utilizing the GPU available in video-cards, therefore we used the library SiftGPU which was able to process an image in 120 ms on average.The feature tracker, five point solver, and RANSAC robust estimator modules were implemented from scratch to have full-control on their parameters and performance optimization. The bundle-adjustment module was implemented from scratch using a simple custom-built linear math library to enhance performance.The feature extraction is the most computationally expensive process in our system. The system is able to run at a frame rate of 10 frame/s when using ORB, but it can only run at : 4 frame/s when using SIFT on our computer. The maximum frame rate can be increased by utilizing a better GPU. The timing cost for processing of one frame of each process in our implementation is shown in Table 1.

Table 1: Processing times of a single image frame.

Module	Processing time
Feature extraction	120 ms
Feature tracking	16 ms
RANSAC	21 ms
Bundle adjustment	146 ms

In Benchmark Datasets Experiments, Visual SLAM community has developed many data sets for SLAM system researchers and developers. There are various data sets comprised of varying sensor combinations. We utilize some of these data sets for the evaluation of our algorithm and for the comparison with the work of others. Data sets containing camera and IMU measurements were utilized. Experiments are performed in two configurations:

Configuration 1: Camera is used alone and inertial sensors are disabled. In this case there is a motion scale ambiguity. For evaluation purposes, we calculate the scale by equating the total real path length to estimated path length.

Configuration 2: Camera is used with inertial sensors enabled. In this configuration there is no scale ambiguity except in the case of the constant velocity motion scenario.

The error criteria used to evaluate the system is defined as: the dvision of distance between the robot real final position and estimated final position over total distance travelled.

$$\text{Error} = \frac{\text{Distance betwen the robot position and final pos.}}{\text{Total distance travelled}} \tag{72}$$

In case of configuration 1, we use the same error criteria but after applying the appropriate scale to estimated results as discussed before.

In Malaga 2009 data set, It was recorded in malaga, spain [39] using a car that travels a distance more than 6 km. The data set contains stero camera, IMU, and RTK GPS for recording of ground truth. The camera calibration and camera to IMU calibration parameters was provided with the data set. Our system is run on a sub-path of this data set in configuration 1 and 2. The results are shown in Figures 18,19 . The errors are displayed in Table 2. The scale drift problem of visual odometry is inherent in Figure 18.

Table 2: Errors for malaga 2009 data set %.

Configuration	**Error**
1	33.9%
2	9.7%

Experiments were performed to assess the accuracy of our algorithm, the camera images was collected at frame rate of 7.5 frame/s and the IMU measurements was collected at a rate of 1150 sample/s, these measurements were stored for later processing and evaluation. Our hardware platform is shown in Figure 23 , an IMU is firmly attached to a camera. We used a Bumblebee 2 camera from Point-Grey which can deliver 1024x768 gray-scale images at a frame rate 30 frame/s. The IMU is a 3-Space USB/RS232 IMU from YEI-technology.

Figure 17: Picture of the hardware platform used in the experiments (Note: only one camera of the stereo is used).

The IMU is calibrated for scale, cross-talk, and bias before usage. we perform the calibration in two steps: *Accelerometer calibration*: The accelerometer is calibrated using the algorithm of [18], we implemented this algorithm using C++. The accelerometer reads only gravity vector in static condition, the gravity has constant magnitude and can be used as a reference for the calibration process. The calibration procedure involves fixing the IMU in various orientations, then an optimization process using down-hill optimizer is performed to estimate the scale and bias of the accelerometer.

Gyroscope calibration: Gyroscope bias can be estimated trivially by measuring its reading at rest. Scale estimation is more difficult. Special hardware - usually a precise turn table - is used for precise calibration of the gyroscope scale. In our case such hardware was unavailable, therefore we relied on the method of [18] followed by a manual try and error adjustment of the parameters.

Afterwards, we evaluated the noise parameters of the IMU. The IMU is held at rest and the measurements are recorded, then the standard deviation of the measurements was determined and is displayed in Table 4. These parameters are required for the IMU measurements integration process.

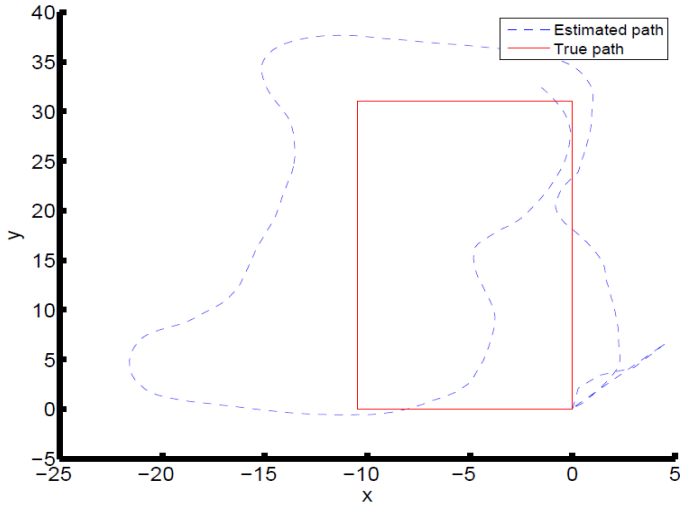


Figure 18: The robot motion path for configuration 1 for malaga dataset.

Table 3: Noise parameters of the IMU sensors.

Sensor	**Noisestandard deviation**
Accelerometer	0.08 ms^{-2}
Gyroscope	0.8 deg/s

The camera to IMU calibration is the determination of the transformation between the camera and the IMU. Precise calibration is necessary to assure stability and accuracy of our algorithm. In our work, the camera to IMU calibration is performed in two steps: (a) A rough calibration is determined using a ruler and protractor. (b) A visual calibration procedure is performed to estimate the calibration with more precision. We used the software toolbox provided by [38] to determine a precise camera to IMU calibration. A rich-motion data set of camera images and IMU measurements was collected and delivered to the toolbox. The temporal timing error between the two sensors was determined to be : $\pm$3ms which is reasonably less than the inter-frame time of 33ms. Once the calibration procedures are finished, we collect a long motion scenario data set for the evaluation of our algorithm.A custom data set is collected using our hardware platform, then processed to provide estimated path. In Figure 20 , the results are shown for configuration 1, while in Figure 21 , the results are shown for configuration 2. The error was determined for each case and displayed in Table 7.

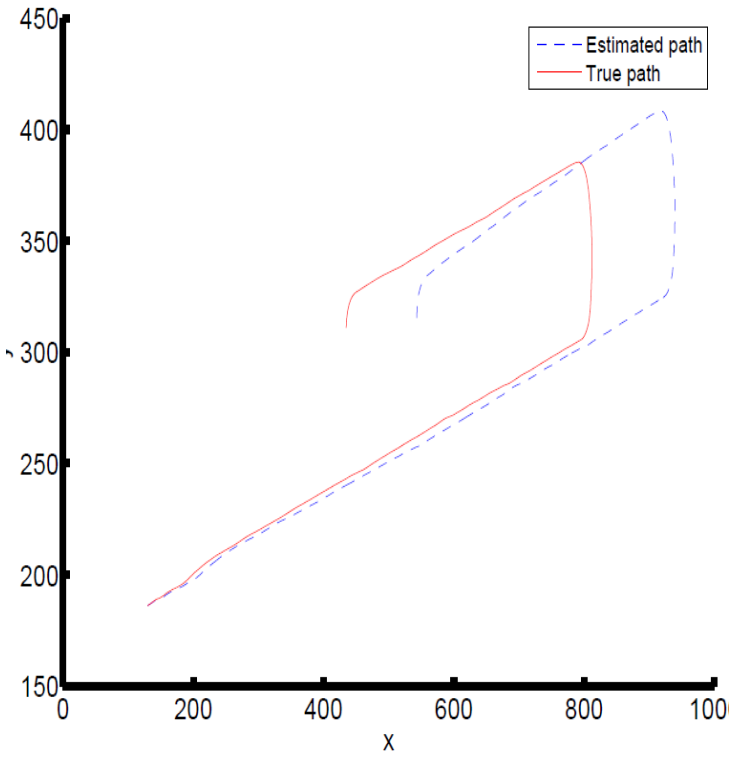


Figure 19: The robot motion path for configuration 2 for malaga dataset.

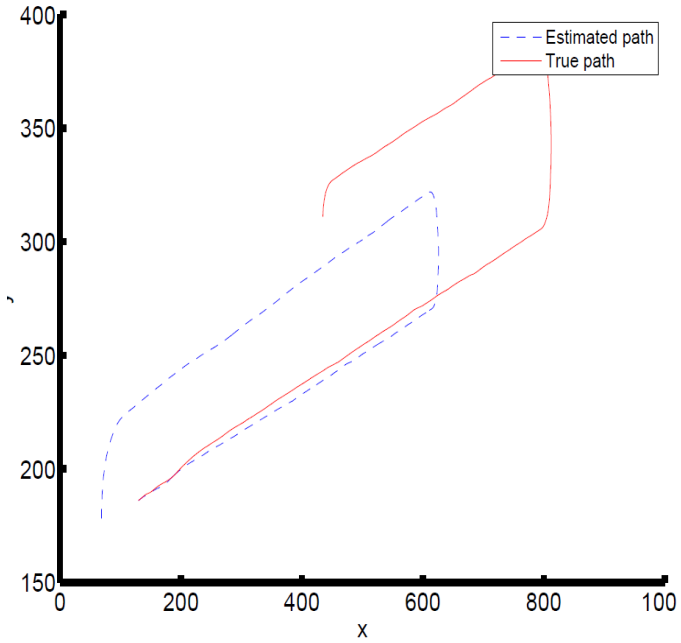


Figure 20: The x-y plane robot motion path for configuration 1.

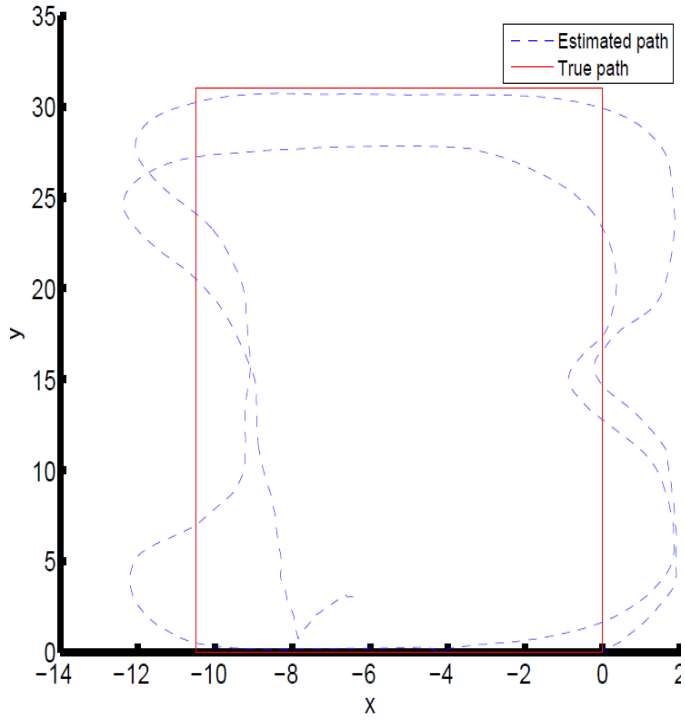


Figure 21: The x-y plane robot motion path for configuration 2.

Differential GPS is usually used to register ground truth for outdoor SLAM systems. The GPS system was not available for our experiments, therefore a simple technique was utilized for the registration of the ground truth. A rectangular path was specified on the ground and the robot was controlled to follow the path closely as possible. Our technique results in a ground truth path with roughly $\pm$ 1 meter error.

Table 4: Errors for the custom collected data set %.

Configuration	Error
1	1.5%
2	6.7%

VII. Conclusions and Future work

We conclude that the new two-stage formulation resulted in an exactly linear map optimization. The advantages of linearity is that linearization is not needed and also that there is no need for initial conditions. The simulations and experiments showed significant improvement of visual SLAM accuracy. Full system simulation reveals the feasibility of our two stage proposed architecture, a position error of less than 10 cm can be achieved with map correction performed each five frames. The scale is well estimated provided that the motion dynamics are rich enough.

It is evident from results that the error is bounded throughout the motion path (i.e. not monotonically increasing), this is due to the fact that the robot observes most of the landmarks during its first loop, afterwards the map is nearly constant. Once the map correction is performed, the map will undergo little change and thus the robot localization error will be bounded. This useful property is achievable by SLAM systems that operate in bounded environments.

REFERENCES

[1] Jia-Qi Fang and Thomas S Huang. "Some experiments on estimating the 3-D motion parameters of a rigid body from two consecutive image frames". In: Pattern Analysis and Machine Intelligence, IEEE Transactions on 5 (1984), pp. 545–554.

[2]Andrew W Fitzgibbon and Andrew Zisserman. "Automatic camera recovery for closed or open image sequences". In: Computer Vision—ECCV'98. Springer, 1998, pp. 311–326.

[3]David Nist´er. "An efficient solution to the five-point relative pose problem." In: IEEE transactions on pattern analysis and machine intelligence 26.6 (June 2004), pp. 756–77.

[4]Bert M. Haralick, Chung-Nan Lee, Karsten Ottenberg, and Michael Nolle. "Review and analysis of solutions of the three point perspective pose estimation problem". In: International Journal of Computer Vision 13.3 (Dec. 1994), pp. 331–356.

[5]Martin A. Fischler and Robert C. Bolles. "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography". In: Communications of the ACM 24.6 (June 1981), pp. 381–395.

[6]David Nist´er, Oleg Naroditsky, and James Bergen. "Visual odometry for ground vehicle applications". In: Journal of Field Robotics 23.1 (2006), pp. 3–20.

[7]David Nist´er. "Preemptive RANSAC for live structure and motion estimation". In:

Proceedings Ninth IEEE International Conference on Computer Vision. IEEE, 2003,199–206 vol.1.

[8]Etienne Mouragnon, Maxime Lhuillier, Michel Dhome, Fabien Dekeyser, and Patrick Sayd. "Generic and real-time structure from motion using local bundle adjustment". In: Image and Vision Computing 27.8 (2009), pp. 1178–1193.

[9]Andrew Howard. "Real-time stereo visual odometry for autonomous ground vehi- cles". In: Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ Interna- tional Conference on. IEEE. 2008, pp. 3946–3952.

[10]Georg Klein and David Murray. "Parallel Tracking and Mapping for Small AR Workspaces". In: 2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality 07.3 (2007), pp. 1–10.

[11]Zhan Wang, Shoudong Huang, and Gamini Dissanayake. "D-SLAM: Decoupled lo- calization and mapping for autonomous robots". In: Robotics Research. Springer,2007, pp. 203–213.

[12]Paul Newman. "On the structure and solution of the simultaneous localisation and map building problem". In: Doctoral diss., University of Sydney (1999).

[13]Eagle S Jones and Stefano Soatto. "Visual-inertial navigation, mapping and local- ization: A scalable real-time causal approach". In: The International Journal of Robotics Research 30.4 (2011), pp. 407–430.

[14]Pedro Pini´es, Todd Lupton, Salah Sukkarieh, and Juan D Tardos. "Inertial aiding of inverse depth SLAM using a monocular camera". In: Robotics and Automation,

2007 IEEE International Conference on. IEEE. 2007, pp. 2797–2802.

[15]Atanas Georgiev and Peter K Allen. "Localization methods for a mobile robot in urban environments". In: Robotics, IEEE Transactions on 20.5 (2004), pp. 851–864.

[16]Stephan Weiss, Markus W Achtelik, Simon Lynen, Margarita Chli, and Roland Siegwart. "Real-time onboard visual-inertial state estimation and self-calibration of mavs in unknown environments". In: Robotics and Automation (ICRA), 2012 IEEE International Conference on. IEEE. 2012, pp. 957–964.

[17] Anthony Kim and MF Golnaraghi. "Initial calibration of an inertial measurement unit using an optical position tracking system". In: Position Location and Navigation Symposium, 2004. IEEE. 2004, pp. 96–101.

[18]WT Fong, SK Ong, and AYC Nee. "Methods for in-field user calibration of an inertial measurement unit without external equipment". In: Measurement Science and Technology 19.8 (2008).

[19] Michael Jean Philippe Tardif, Michel Laverne, Alonzo Kelly, and Anthony Stentz. "A new approach to vision-aided inertial navigation". In: Intelligent Robots and Sys- tems (IROS), 2010 IEEE/RSJ International Conference on. IEEE. 2010, pp. 4161–4168.

[20]Stefan Leutenegger, Paul Timothy Furgale, Vincent Rabaud, Margarita Chli, Kurt Konolige, and Roland Siegwart. "Keyframe-Based Visual-Inertial SLAM using Non- linear Optimization." In: Robotics: Science and Systems. 2013.

[21]Bill Triggs, Philip F McLauchlan, Richard I Hartley, and Andrew W Fitzgibbon. "Bundle adjustment—a modern synthesis". In: Vision algorithms: theory and prac- tice. Springer, 2000, pp. 298–372.

[22]Tim Bailey, Juan Nieto, Jose Guivant, Michael Stevens, and Eduardo Nebot. "Con- sistency of the EKF-SLAM algorithm". In: Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on. IEEE. 2006, pp. 3562–3568.

[23]Kurt Konolige and Motilal Agrawal . " Frame SLAM: From bundle adjustment to realtime visual mapping". In: Robotics, IEEE Transactions on 24.5 (2008), pp. 1066{

1077.

[24]Gabe Sibley, Christopher Mei, Ian Reid, and Paul Newman. "Adaptive RelativeBundle Adjustment". In: IEEE Transactions on Robotics 220.3 (2009), pp. 1–8.

[25] Hauke Strasdat, JMM Montiel, and Andrew J Davison. "Real-time monocular slam: Why filter?" In: Robotics and Automation (ICRA), 2010 IEEE International Con- ference on. IEEE. 2010, pp. 2657–2664.

[26] Christopher Mei, Gabe Sibley, Mark Cummins, Paul M Newman, and Ian D Reid. "A Constant-Time Efficient Stereo SLAM System." In: BMVC. 2009, pp. 1–11.

[27] Mark Cummins and Paul Newman. "FAB-MAP: Probabilistic localization and map- ping in the space of appearance". In: The International Journal of Robotics Research

27.6 (2008), pp. 647–665.

[28]Mark Cummins and Paul Newman. "Probabilistic appearance based navigation and loop closing". In: Robotics and automation, 2007 IEEE international conference on. IEEE. 2007, pp. 2042–2048.

[29]Kurt Konolige, James Bowman, JD Chen, Patrick Mihelich, Michael Calonder, Vin- cent Lepetit, and Pascal Fua. "View-based maps". In: The International Journal of Robotics Research (2010).

[30]Frank Dellaert and Michael Kaess. "Square Root SAM: Simultaneous localization and mapping via square root information smoothing". In: The International Journal of Robotics Research 25.12 (2006), pp. 1181–1203.

[31]William H Press. Numerical recipes 3rd edition: The art of scientific computing.Cambridge university press, 2007.

[32]Zhengyou Zhang. "A flexible new technique for camera calibration". In: PatternAnalysis and Machine Intelligence, IEEE Transactions on 22.11 (2000), pp. 1330–1334.

[33]Richard Hartley and Andrew Zisserman. Multiple View Geometry in Computer Vi-sion. Cambridge University Press, 2003.

[34]Richard I. Hartley and Peter Sturm. \Triangulation". In: Computer Vision andImage Understanding 68.2 (Nov. 1997), pp. 146{157.

[35]Tue-Cuong Dong-Si and A.I. Mourikis. "Estimator initialization in vision-aided iner- tial navigation with unknown camera-IMU calibration". In: Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on. 2012, pp. 1064–1071.

[36]Javier Civera, Andrew J Davison, and J Montiel. "Inverse depth parametrization for monocular SLAM". In: Robotics, IEEE Transactions on 24.5 (2008), pp. 932–945.

[37]Cyril Roussillon, Aurelien Gonzalez, and Joan Sol`a. "RT-SLAM: a generic and real- time visual SLAM implementation". In: Computer Vision . . . (Jan. 2011), p. 10.

[38]Paul Furgale, Joern Rehder, and Roland Siegwart , "Unied temporal and spatial calibration for multi-sensor systems". In: Intelligent Robots and Systems (IROS),2013 IEEE/RSJ International Conference on. IEEE. 2013, pp. 1280{1286.

[39]Jos´e-Luis Blanco, Francisco-Angel Moreno, and Javier Gonz´alez. "A Collection of Outdoor Robotic Datasets with centimeter-accuracy Ground Truth". In: Au- tonomous Robots 27.4 (Nov. 2009), pp. 327–351.

[40]Mohamed H. Mahmoud, N. M. Hussein, Elsayed Hemayed, F. Endres, W. Burgard, and D. Cremers, "A FAST LINEARLY BACK-END SLAM FOR NAVIGATION BASED ON MONOCULAR CAMERA," in International Journal of Civil Engineering and Technology (IJCIET) Volume 9, Issue 12, December 2018, pp. 627-645.

[41] Mohamed H. Merzban, etc.. "Toward multi-stage decoupled visual SLAM system" , 2013 IEEE International Symposium on Robotic and Sensors Environments (ROSE), 2013.