Monocular Visual-Inertial Odometry for 3D Positioning

Choong Yi Fung
School of Electrical & Electronic
Universiti Sains Malaysia
Penang, Malaysia
choongcyf@student.usm.my

Professor Dr. Mohd Zaid Abdullah
School of Electric & Electronic
Universiti Sains Malaysia
Penang, Malaysia
mza@usm.my

***Abstract*—Odometry is a localization technique that has been widely employed in an autonomous mobile platform. In general, odometry estimates pose of a mobile platform over time using the onboard sensor measurements and categorized according to the respective sensors. It is convinced that the fusing of data from a proprioceptive sensor and an exteroceptive sensor will yield a better accuracy in comparison to that of the standalone odometry. Hence, the objective of the research is to study the application of a monocular visual-inertial odometry and compare the respective performance to that of the inertial odometry and the visual odometry. The sensor fusion method is constructed based on the multi-state constraint Kalman filter (MSCKF) fusing framework, which include the initialization of the state vector, state propagation, state augmentation, measurement model and the state update. We present experimental comparisons of the visual-inertial odometry (VIO), inertial odometry (IO) and the visual odometry (VO) on KITTI dataset with real and synthetic features. A root mean square error test is conducted to further validate the result where the proposed method recorded the lowest root mean square error (RMSE) and effectively reduced the RMSE of IO and VO. In a nutshell, the filter based monocular VIO able to perform localization and proven to be more accurate than that of the IO and a VO.**

Keywords*— *odometry, localization, sensor fusion, proprioceptive sensor, exteroceptive sensor, MSCKF

I. Introduction

Navigation, defined from Oxford Languages, as the process or activity of accurately ascertaining one’s position and planning-and-following a route. In the recent years, the development of a navigation system has been highlighted by the autonomous industry as one of the major challenges in the construction of an autonomous mobile platform. Few important criteria of an autonomous mobile platform are mentioned by Mohamed et al. (2019), which include a dependable navigation capability and the ability to self-localize over time.

The statement motivates the study on odometry which is a localization technique that has been widely employed in an autonomous mobile platform. In general, odometry estimates pose of a mobile platform over time using onboard sensor measurements [2]. As suggested by Mingyang Li in his work, the key decision in designing a system for odometry purposes is the selection of appropriate sensors [3]. The sensors are further categorized into proprioceptive sensors and exteroceptive sensors with the former provide measurements of quantities related to the motion of the mobile platform, whereas the latter capture key information from the surroundings.

An example of a proprioceptive sensor is an inertial measurement unit (IMU) which provides the position and orientation of a mobile platform relative to a known starting point. It is identified by Aqel et al. (2016) that the IMU has the advantage of not subjecting to external references as it is self-contained. Despite that, IMU suffers from position drift in long term application due to accumulative drift errors. On the contrary, the optical camera is an example of an exteroceptive sensor which has the advantage in improving the accuracy of localization due to the rich geometric information obtained from images, yet, the application is computational-costly, and the information extraction techniques are complicated. However, it is convinced from Dunn (2013) that the combination of both improves the navigation capability as the exteroceptive sensors are a natural supplement to proprioceptive sensors such as IMU. This is supported by the fact that the geometric information from an exteroceptive sensor can be used to correct the errors accumulated during the integration of the IMU measurements [3].

Generally, it is understood that the increase in the number of sensors implemented will lead to an increase in the system accuracy due to the rich information obtained from multiple sensors. However, to install multiple sensors, a trade-off must be done between the cost of setup and the computation efficiency of the system [6]. Hence, it is indeed a challenge to implement a monocular setup of an exteroceptive sensor in the localization of an odometry.

Therefore, the research centers on the application of a monocular visual-inertial odometry (VIO) in performing pose estimation in 3D space, using information from both inertial measurement units (IMU) and a monocular camera. The Multi-state constraint Kalman Filter (MSCKF) is to be implemented as the fusion algorithm for both sensors in the tightly-couple monocular VIO framework. This is supported by the fact that MSCKF could be performed on all the suggested hardware platforms by Delmerico & Scaramuzza (2018) such as laptop and single-board embedded computer with a consistent accuracy.

II. Related Works

The visual-inertial odometry (VIO) is a localization method based on both vision and IMU. It integrates both methods and overcomes the limitations from both sides. There are number of ways in categorizing the method in conducting VIO, however, in this section, the focus is mainly on the fusion frameworks for visual and inertial data. The VIO can be categorized based on how the visual and inertial data are fused: filter based and optimization-based, based on when the measurements are fused: loosely coupled and tightly coupled [8].

The optimization-based approaches, also known as smoothing-based approaches by Mohamed et al. (2019), estimate the pose by jointly optimizing key information extracted from images and inertial measurements from IMU sensor [9]. Therefore, they outperform filter-based approaches in terms of accuracy. However, carrying out iterative minimization of a least square error function requires more computational resources. The research further categorized the method into three main branches: fixed-lag smoothing, full smoothing and incremental smoothing. A work developed by Hong (2018) proposed a novel monocular VIO algorithm in estimating the motion state of unmanned aerial vehicles (UAV) with high accuracy [10]. The method proposed the fusion of monocular visual information and pre-integrated inertial measurements in a joint optimization framework which was similar to the work developed by Tong (2018) in [11]. The latter focused in reducing the computational burden in relocalization by adding a loop detection module in the tightly coupled formulation whereas the former focused in improving the overall accuracy by introducing a robust online initialization algorithm which enforce the relative pose constraints between keyframes acquired, increasing the reliability of the pose estimated .

On the other hand, filter-based approaches are among the earliest approaches used to solve VIO and Simultaneous Localization and Mapping (SLAM) problems. The approach is simplified into two main components, the prediction step and the update step [1]. In filter based VIO, the dynamic model of a platform is computed by using linear and angular velocities measured from the IMU whereas the dynamic model is used in the prediction step to predict motion of the platform. In addition, key information, such as features, or pixel intensities extracted from captured images are utilized in the measurement model to update the predictions in the update step. Moreover, it is understood that the priority of a filter-based approach is to include the IMU pose and map point positions in the filter state, then recursively propagate and update the state as the measurements from IMU and camera arrive respectively [8]. There are several examples on the application of filter-based approaches such as extended Kalman Filter (EKF), MSCKF, and unscented Kalman Filter (UKF). Some example on the filter-based approach VIO, a work by Rodrigo (2016) in used the EKF in fusing the information obtained from IMU measurements and the visual features acquired. However, the depth estimation of the system is not well conditioned [12]. Besides, Ma (2019) proposed a related work based on MSCKF and improvised the original algorithm with the addition of Ackermann error state measurements in a tightly coupled manner. The additional constraints from the Ackermann measurements are used to improve the pose estimation accuracy [13].

The MSCKF is a structureless method for VIO in which, it can attain better precision and consistency, due to its less strict probabilistic assumption and delayed linearization [1]. The authors further explained that the MSCKF ensures a good compromisation between computational cost and precision as the framework has complexity linear to the number of landmarks. However, one of the main drawbacks of the conventional MSCKF algorithm is that it has incorrect observability properties, which leads to the slight inconsistency in performance. The MSCKF is classified as "sliding window'' by Mourikis & Roumeliotis (2007) because it maintains a sliding window of camera poses in the filter state and uses the feature measurements to impose probabilistic constraints on the camera poses. . The MSCKF makes optimal use of the geometric constraints that arise from observing the same feature in different images without adding them to the state [15]. In comparison with the EKF algorithms, Li and Mourikis proved that both EKF and MSCKF algorithms would yield the same, optimal estimates for IMU pose if the systems were linear-Gaussian [14]. However, the actual measurement models are nonlinear, which allows the MSCKF to outperform EKF algorithms in terms of accuracy.

III. METHODOLOGY

The proposed method was implemented using MATLAB based on the algorithm described and began with the initialization of the MSCKF state vector. The algorithm processed the available IMU measurements in state propagation. Each time an image is recorded, the image processing frontend detect and track the feature points in the image before augmenting the state by pushing the current camera pose to the sliding window and updating the covariance accordingly. For the updates, the feature positions are triangulated using Gauss-Newton optimization. The reprojection errors of the estimated feature positions with respect to the observed values will be utilized as filter residuals. The residuals will then be used in performing filter correction, which yield the observed error state. The error state will then be injected to the nominal state and reset to zero.

A. State Vector

The IMU state is described by the vector:

$$X_{IMU,k} = \left[q_{IG,k}^{T} \quad b_{g,k}^{T} \quad {v_{G,k}^{IG}}^{T} \quad b_{a,k}^{T} \quad {p_{G,k}^{IG}}^{T}\right]^{T} \tag{1}$$

where $q_{IG,k}$ is the unit quaternion describing the rotation from global frame (G) to IMU frame (I). $p_{G,k}^{IG}$ and $v_{G,k}^{IG}$ are the iMU position and velocity with respect to the global frame (G). b_g and b_a are the biases affecting the gyroscope and accelerometer measurements respectively. The IMU error-state is defined as:

$$\tilde{X}_{IMU,k} = \left[\delta\theta_{I}^{T} \quad \tilde{b}_{g,k}^{T} \quad {\tilde{p}_{G,k}^{IG}}^{T} \quad \tilde{b}_{a,k}^{T} \quad {\tilde{p}_{G,k}^{IG}}^{T}\right]^{T} \tag{2}$$

From (2), the quaternion from IMU error-state is expressed differently. The rotational error $\delta\theta$ is used instead of the error quaternion δq. Since attitude corresponds to three degree of freedom, using $\delta\theta$ to describe the attitude errors is a minimal representation. $\delta\theta$ is defined according to:

$$\delta q \cong \left[\frac{1}{2}\delta\theta^{T} \quad 1\right]^{T} \tag{3}$$

At time k, the nominal state of the MSCKF consists of the current IMU state estimate and the estimates of N past camera poses in which active feature tracks were visible. The MSCKF state vector is defined as:

$$\hat{X}_{k} = \left[{\hat{X}_{IMU,k}}^{T} \quad {\hat{q}_{C_1G}}^{T} \quad {\hat{p}_{G}^{C_1G}}^{T} \cdots {\hat{q}_{C_NG}}^{T} \quad {\hat{p}_{G}^{C_NG}}^{T}\right]^{T} \tag{4}$$

Where $\hat{q}_{C_iG}$ and $\hat{p}_G^{C_iG}$, at $i = 1 \dots N$ are the estimates of the camera attitude and position, respectively. Similarly, the MSCKF error state vector at time k is defined as:

$$\tilde{X}_k = \begin{bmatrix} \tilde{X}_{IMU,k}{}^T & \delta\theta_{C_1}^T & \tilde{p}_G^{C_1G^T} \dots & \delta\theta_{C_N}^T & \tilde{p}_G^{C_NG^T} \end{bmatrix} \quad (5)$$

B. State Propagation

The state propagation equations are first derived in continuous-time IMU system model before discretized for implementation. Based on the continuous-time system modelling, the time evolution of the mean estimated IMU state is defined as:

$$\begin{gathered} \dot{\hat{q}}_{IG} = \frac{1}{2}\Omega(\hat{\omega})\hat{q}_{IG}, \\ \dot{\hat{b}}_g = 0_{3\times1}, \\ \dot{\hat{v}}_I^G = \hat{C}_{IG}^T(a_m - \hat{b}_a) - 2(\omega_G^\times)\hat{v}_I^G - (\omega_G^\times)^2\hat{p}_I^G + g^G, \\ \dot{\hat{b}}_a = 0_{3\times1}, \\ \dot{\hat{p}}_I^G = \hat{v}_I^G \end{gathered} \quad (6)$$

where $\hat{C}_{IG}$ is the rotation matrix corresponding to the quaternion $\hat{q}_{IG}$, a_m is the accelerometer measurement, g^G is the gravitational acceleration and ω_m is the gyroscope measurement. The IMU measurement incorporate the effects of the planet's rotation, $\omega_G^\times$ with magnitude equal to $7.292 \times 10^{-5}\, {}^{rad}/_{s}$. Next, Ω and $\hat{\omega}$ are defined where ω is the rotational velocity expressed in the IMU frame.

$$\Omega(\hat{\omega}) = \begin{bmatrix} -\hat{\omega}^\times & \hat{\omega} \\ -\hat{\omega}^T & 0 \end{bmatrix} \quad (7)$$

$$\hat{\omega}^\times = \begin{bmatrix} 0 & -\hat{\omega}_z & \hat{\omega}_y \\ \hat{\omega}_z & 0 & -\hat{\omega}_x \\ -\hat{\omega}_y & \hat{\omega}_x & 0 \end{bmatrix}$$

$$\hat{\omega} = \omega_m - \hat{C}_{IG}\omega_G - \hat{b}_g - n_g$$

The linearized continuous time-model for the IMU error state is defined as:

$$\dot{\tilde{X}}_{IMU} = F\tilde{X}_{IMU} + Gn_{IMU} \quad (8)$$

where n_{IMU} , the system noise is defined as:

$$n_{IMU} = \begin{bmatrix} n_g^T & n_{wg}^T & n_a^T & n_{wa}^T \end{bmatrix}^T \quad (9)$$

The Jacobians F and G which can be found in Equation 3.48 are defined in (10) and (11) respectively.

$$F = \begin{bmatrix} -\hat{\omega}^\times & -I_3 & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} \\ -\hat{C}_{IG}^T\hat{a}^\times & 0_{3\times3} & -2\omega_G^\times & -\hat{C}_{IG}^T & -(\omega_G^\times)^2 \\ 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & 0_{3\times3} & I_3 & 0_{3\times3} & 0_{3\times3} \end{bmatrix} \quad (10)$$

$$G = \begin{bmatrix} -I_3 & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & I_3 & 0_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & 0_{3\times3} & -\hat{C}_{IG}^T & 0_{3\times3} \\ 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & I_3 \\ 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} \end{bmatrix} \quad (11)$$

For the discrete-time implementation, the measurements are sampled with period T and used for state propagation in the MSCKF. The IMU state estimate is then propagated using fifth order Runge-Kutta integration. The MSCKF covariance matrix will be propagated and hence, the propagated output is given by a $(15 + 6N) \times (15 + 6N)$ matrix:

$$\hat{P}_{\bar{k}} = \begin{bmatrix} \hat{P}_{\overline{II},k} & \varphi(t_{k-1}+T, t_{k-1})\hat{P}_{IC,k-1} \\ \hat{P}_{IC,k-1}^T\varphi(t_{k-1}+T, t_{k-1})^T & \hat{P}_{CC,k-1} \end{bmatrix} \quad (12)$$

where $\hat{P}_{CC,k-1}$ is the $6N \times 6N$ covariance matrix of the camera pose estimates and $\hat{P}_{IC,k-1}$ is the correlation between the errors in the IMU state and the camera pose estimates.

Next, $\hat{P}_{\overline{II},k}$ is the 15×15 covariance matrix of the evolving IMU state, computed by the integration of (13) for time interval $(t_{k-1} + T)$.

$$\dot{\hat{P}}_{\overline{II}} = F\hat{P}_{\overline{II}} + \hat{P}_{\overline{II}}F^T + GQ_{IMU}G^T \quad (13)$$

where Q_{IMU} is the covariance matrix of system noise n_{IMU} and computed offline during calibration. Similarly, the state transition matrix $\varphi(t_{k-1} + T, t_{k-1})$ is computed with the integration of the differential equation, (14), with initial condition of $\varphi(t_{k-1}, t_{k-1}) = I_{15}$.

$$\dot{\varphi}(t_{k-1} + \tau, t_{k-1}) = F\varphi(t_{k-1} + \tau, t_{k-1}), \quad (14)$$
$$\tau \in [0, T]$$

C. State Augmentation

The state augmentation is initiated each time an image is available, where the MSCKF state will be augmented with the current camera pose estimate. The camera pose estimate is computed from the IMU pose estimate as:

$$\hat{q}_{CG} = q_{CI} \otimes \hat{q}_{IG} \quad (15)$$

$$\hat{p}_C^G = \hat{p}_I^G + \hat{C}_{IG}^T p_C^I \quad (16)$$

where q_{CI} is the quaternion expressing the rotation between the IMU and camera frames whereas p_C^I is the position of the origin of the camera frame with respect to the IMU frame. $\hat{C}_{IG}^T$ is the rotation matrix corresponding to $\hat{q}_{IG}$. The MSCKF covariance matrix is augmented according to:

$$P_{k-1} \leftarrow \begin{bmatrix} I_{6N+15} \\ J \end{bmatrix} \hat{P}_{k-1} \begin{bmatrix} I_{6N+15} \\ J \end{bmatrix}^T \quad (17)$$

where the Jacobian is given by:

$$J = \begin{bmatrix} \hat{C}_{CI} & 0_{3\times9} & 0_{3\times3} & 0_{3\times6N} \\ \left(\hat{C}_{IG}^T p_C^I\right)^\times & 0_{3\times9} & I_3 & 0_{3\times6N} \end{bmatrix} \quad (18)$$

D. Measurement Model

The state correction equations are employed for updating the state estimates. In the construction of the equations, the MSCKF defines a residual, r, an exteroceptive measurement error corresponding to an observation $z_i^{(j)}$ of feature f_i from the respective camera pose, C_i:

$$r_i^{(j)} = z_i^{(j)} - \hat{z}_i^{(j)} \quad (19)$$

where

$$\hat{z}_i^{(j)} = \frac{1}{\hat{Z}_{C_i}^{(j)}} \begin{bmatrix} \hat{X}_{C_i}^{(j)} \\ \hat{Y}_{C_i}^{(j)} \end{bmatrix} \quad (20)$$

with

$$\hat{p}_{C_i}^{f_jC_i} = \begin{bmatrix} \hat{X}_{C_i}^{(j)} \\ \hat{Y}_{C_i}^{(j)} \\ \hat{Z}_{C_i}^{(j)} \end{bmatrix} = \hat{C}_{C_iG}\left(\hat{p}_G^{f_jG} - \hat{p}_G^{C_iG}\right) \quad (21)$$

The linearization of (21) about the estimates the estimates for the camera pose and feature position produces an estimate of the residual $r_i^{(j)}$:

$$r_i^{(j)} \cong H_{x,i}^{(j)}\tilde{x}_i + H_{f,i}^{(j)}\tilde{p}_G^{f_iG} + n_i^{(j)} \quad (22)$$

where $H_{x,i}^{(j)}$ and $H_{f,i}^{(j)}$ are the Jacobians of the measurement $r_i^{(j)}$ with respect to the filter state and the position of the feature, respectively. $\tilde{p}_G^{f_iG}$ is the error in the position estimate of the feature. $n_i^{(j)}$ is a zero-mean Gaussian noise. The $H_{x,i}^{(j)}$ and $H_{f,i}^{(j)}$ are given by:

$$H_{x,i}^{(j)} = \left[0 \quad J_i^{(j)}\left(\hat{p}_{C_i}^{f_jC_i}\right)^{\times} \quad -J_i^{(j)}\hat{C}_{C_iG} \quad 0\right] \quad (23)$$

$$H_{f,i}^{(j)} = J_i^{(j)}\hat{C}_{C_iG} \quad (24)$$

with $J_i^{(j)}$ given by:

$$J_i^{(j)} = \frac{1}{\left(\hat{Z}_{C_i}^{(j)}\right)^2}\begin{bmatrix} \hat{Z}_{C_i}^{(j)} & 0 & -\hat{X}_{C_i}^{(j)} \\ 0 & \hat{Z}_{C_i}^{(j)} & -\hat{Y}_{C_i}^{(j)} \end{bmatrix} \quad (25)$$

Next, the algorithm stacks the residuals of all the measurements of the feature, thus obtained the equation:

$$r^{(j)} \cong H_X^{(j)}\tilde{x} + H_f^{(j)}\tilde{p}_G^{f_jG} + n^{(j)} \quad (26)$$

where $r^{(j)}$, $H_X^{(j)}$, $H_f^{(j)}$ and $n^{(j)}$ are the block vectors with the respective elements. Since the term $H_f^{(j)}\tilde{p}_G^{f_jG}$ is correlated to the state, $r^{(j)}$ is not in the correct form for the MSCKF and must be modified. The correlation causes the filter estimates to drift further form the true values. Hence, a unitary matrix A is defined, with its columns form the basis of the left nullspace of $H_f^{(j)}$. $r^{(j)}$ is then modified by projecting it into the nullspace to obtain a residual equation in corrected form:

$$r_0^{(j)} = A^T\left(r^{(j)}\right) \cong A^T H_X^{(j)}\tilde{x} + A^T n^{(j)} \quad (27)$$

$$\therefore r_0^{(j)} = H_0^{(j)}\tilde{x}^{(j)} + n_0^{(j)} \quad (28)$$

The residual equation defined is independent of the errors in the feature coordinates, thus, usable for the MSCKF state update. This section is concluded with the geometric constraints imposed by the observation of a static feature from multiple cameras defined.

E. State Update

The state update is the final step performed by the algorithm. It applies the geometric constraints defined to observe and correct the errors. Each time a new image is recorded by the system, the sliding window is expanded with the augmentation of the new camera pose and the number of camera poses, N increases. The update step is triggered once the maximum allowable number of camera poses, N_{max}, is achieved and the algorithm proceeds with the removal of old window poses to make space for the new ones. The algorithm stacks residuals $r_0^{(j)}$ in a single vector:

$$r_0 = H_0\tilde{x} + n_0 \quad (29)$$

Due to the large dimension of the vector, QR-decomposition is used on H_0 to reduce the computational complexity of the update step:

$$H_0 = [Q_1 \quad Q_2]\begin{bmatrix} T_H \\ 0 \end{bmatrix} \quad (30)$$

where Q_1 and Q_2 are unitary matrices, while T_H is an upper-triangular matrix. The substitution of (30) into (29) produced:

$$r_0 = [Q_1 \quad Q_2]\begin{bmatrix} T_H \\ 0 \end{bmatrix}\tilde{x} + n_0 \quad (31)$$

Equation (31) is then pre-multiplied by $[Q_1 \quad Q_2]^T$:

$$\begin{bmatrix} Q_1^T r_0 \\ Q_2^T r_0 \end{bmatrix} = \begin{bmatrix} T_H \\ 0 \end{bmatrix}\tilde{x} + \begin{bmatrix} Q_1^T n_0 \\ Q_2^T n_0 \end{bmatrix} \quad (32)$$

Since the quantity of $Q_2^T n_0$ is only noise, it is discarded and thus, a new residual term is defined.

$$r_n = Q_1^T r_0 = T_H\tilde{x} + Q_1^T n_0 \quad (33)$$

$$\therefore r_n = T_H\tilde{x} + n_n$$

Where n_n is noise vector with covariance matrix and equivalent to:

$$R_n = Q_1^T R_0^{(j)} Q_1 \quad (34)$$

Finally, the Kalman gain and the state correction is formulated and computed:

$$K_k = \hat{P}_{\bar{k}} T_{H,k}^T\left(T_{H,k}\hat{P}_{\bar{k}}T_{H,k}^T + R_{n,k}\right)^{-1} \quad (35)$$

$$\Delta X_k = K_k r_{n,k} \quad (36)$$

The state covariance matrix is updated accordingly:

$$\hat{P}_k = \left(I_{6N+15} - K_k T_{H,k}\right)\hat{P}_{\bar{k}}\left(I_{6N+15} - K_k T_{H,k}\right)^T + K_k R_{n,k} K_k^T \quad (37)$$

F. MATLAB Implementation of VIO

This section of the methodology explains the implementation details of the MSCKF algorithm in the monocular VIO framework. The development of the project is mainly done in MATLAB on KITTI dataset with the utilization of vision toolbox, some notable functions such as triangulation and feature detection.

The program began by loading the desired dataset, comprised of the image data, IMU data, calibration data and the ground truth data. The algorithm parameters were tuned before the initiation of the program. With the data available, the program initialized the MSCKF state according to the state vector defined in the MSCKF algorithm which include the nominal state and the error state. The program proceeded with the function state propagation. State propagation is conducted for every IMU measurement recorded by the system. The propagated state is stored as the result of a dead-reckoning system and further used to compare with the result of the proposed VIO framework. Fig. 1 illustrated the overall VIO implementation in the project.

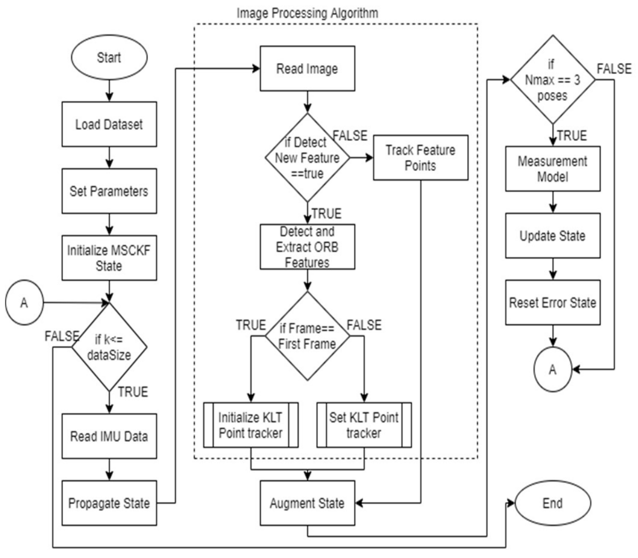

Fig. 1 Flowchart of the overall implementation

Next, the image processing frontend recorded image and processed it differently with respect to the condition of whether there was a need to detect new feature points from an image. If indeed there was, the new feature points will be detected using Oriented FAST and Rotated BRIEF (ORB) feature detector and extracted, otherwise, the feature points tracks will be tracked using Kanade-Lukas-Tomasi (KLT) point tracker based on the tracked feature points identified in the previous image. The former only happened in two type of cases where the first one was the new feature detection on the image of the first frame whereas the second one was the new feature detection on the image after the completion of the state update. For the KLT point tracker, it was implemented using the predefined function from vision toolbox, *PointTracker*.

For the update step, it will not be triggered unless N_{max} achieved three poses. The function further updated the state using the Kalman gain computed from the information acquired from the feature tracks. Lastly, the trajectory of the platform based on the MSCKF state is plotted using the predefined function in MATLAB, *3D plot*, in comparison with the trajectory plotted based on the ground truth information.

G. *MATLAB Implementation of VO*

The VO implementation was utilized based on the study conducted by Alatur & Schaefer (2018), which applied the optical flow method in tracking the feature points from a set of consecutive images. The algorithm began with the bootstrap initialization which ensure the algorithm had enough information to begin with. The aim of bootstrapping was to identify the landmarks which was a key component in initiating the 3D-2D correspondences technique. The landmark was known from the triangulation of points originated from two consecutive images which required the essential matrix to be computed beforehand. Fig. 2 illustrated the flowchart of the monocular VO algorithm implemented in the research.

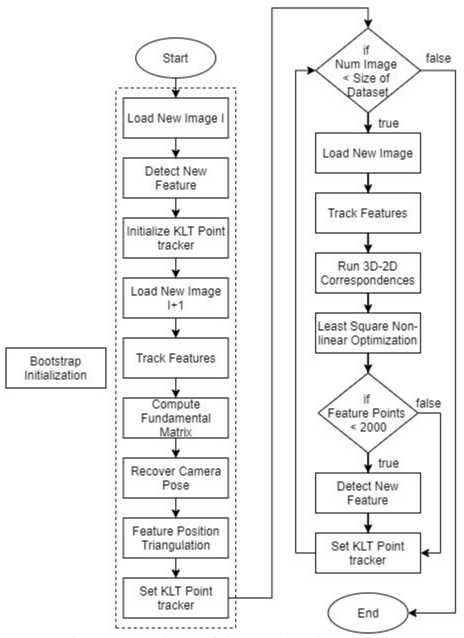

Fig. 2 Flowchart of the VO implementation

Once the bootstrap was initialized, the system proceeded to the main loop of the program. For each time an image was recorded, the key feature points were tracked from the respective image based on the tracked features obtained from the previous images before computed using the predefined 3D-2D correspondences function in MATLAB vision toolbox, *estimateworldcamerapose*. The expected output would be the 3×4 matrix consist of the location and orientation of the mobile platform. The nonlinear optimization was utilized to minimize the reprojection error. The feature points were triangulated to produce landmarks, which were vital for the 3D-2D correspondences of the next iteration. In the case where the tracked feature points went below 2000, most probably due to the major changes in the structure of the environment, a new set of feature points will be detected. Lastly, the feature points in this frame will be used to set the KLT point tracker before acquiring a new image

IV. RESULT AND DISCUSSION

A. *Traversed path comparison between the VIO, IO and VO*

To test the performance of each method, the traversed path of each odometry was constructed as shown in Fig. 2, Fig. 3 and Fig. 4 respectively. The paths were constructed using a total of 90 frames from the synchronized KITTI dataset, in which all the IMU data and the camera data were rectified and arranged in sync. The green path represented the traversed path constructed using the ground truth data or global positioning system (GPS), the blue path represented the traversed path constructed using the data from VIO, the red path represented the traversed path constructed using the data from IO and the cyan path represented the traversed path constructed using the data from VO. Fig. 3 illustrated the traverse paths of IO and ground truth in 3D.

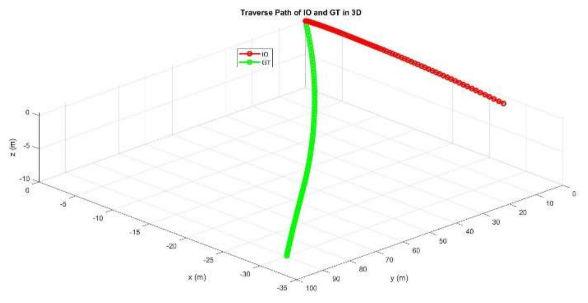

Fig. 3 Traversed path of the IO and ground truth

Fig. 4 illustrated the traverse paths of VO and ground truth in 3D.

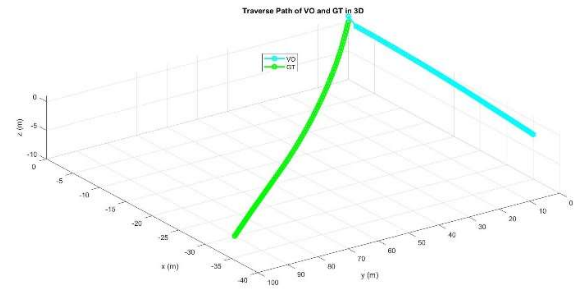

Fig. 4 Traversed path of the VO and ground truth

Fig. 5 illustrated the traverse paths of VIO and ground truth in 3D.

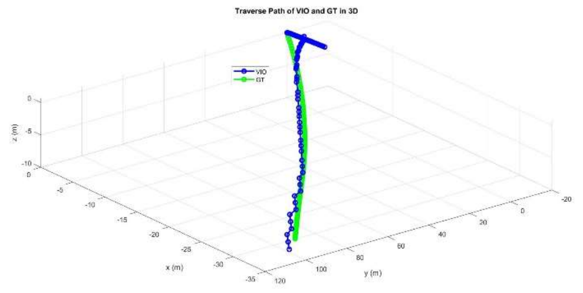

Fig. 5 Traversed path of the VIO and ground truth

The desired result from the test would be the case where the blue path positioned as close as possible to the green path. This would allow the fulfilment of the project's first objective where the application of VIO has successfully estimated the pose of a platform over time. Initially as observed from the result displayed in Fig. 3, Fig. 4 and Fig. 5, taking the green path as reference, it was identified that all the paths move in an undesired direction and deviated away from the reference path. However, with the red path and the cyan path remained in the same direction and travelled along the x-axis, the green path made a huge change in the direction travelled, thus reduced the deviation from the blue path.

From a rough observation, it was thus observed that the VIO produced a more accurate result in comparison with IO and VO. Hence, it was concluded from the test that the VIO method managed to perform pose estimation over time by travelling along the ground truth. The accuracy of each method was further validated in the next test.

B. *Performance analysis of the VIO, IO and VO*

This section validated the result obtained from the first test, it was identified from a rough observation that the VIO yielded a more accurate result than that of the IO and the VO. Hence, the accuracy of each method was further quantified by the computation of the deviation in the path travelled and the root mean square error (RMSE) respectively. Fig. 6 illustrated the deviation in the path travelled in term of x-axis, y-axis and z-axis.

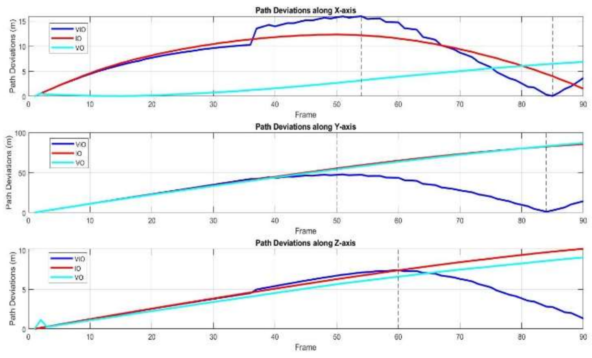

Fig. 6 Deviation of the traversed paths in term of x-axis, y-axis and z-axis

Based on Fig. 6, the maximum deviation, d_{max} and average deviation, $\bar{d}$ in each axis were identified and grouped according to the respective path. The estimated path deviations recorded by the proposed VIO (blue path) along the x-axis reached the peak at frame 54 before decreased gradually to 0. Whereas for the estimated paths along the y-axis and z-axis, the deviations recorded by the proposed VIO (blue path) reduced over time after frame 50 and frame 60 respectively. TABLE I displayed the deviations recorded in each path.

TABLE I The deviations recorded by the tested odometry

Axis	VIO (green)		IO (red)		VO (cyan)	
	d_{max} (m)	$\bar{d}$ (m)	d_{max} (m)	$\bar{d}$ (m)	d_{max} (m)	$\bar{d}$ (m)
X	15.96	7.78	12.32	8.64	7.49	3.14
Y	47.61	27.15	89.34	52.00	92.74	51.82
Z	7.43	4.25	10.79	6.03	9.71	5.38

Based on TABLE I, VIO recorded the lowest average deviation and maximum deviation in Y-axis, at 27.15m and 47.61m respectively. The major different among the three method was identified in Y-axis, where both IO and VO recorded a much higher deviation in average as compare to that of the VIO, recorded at 52.00m and 51.82m respectively. Moreover, VIO recorded the lowest average deviation and maximum deviation in Z-axis as well, at 4.25m and 7.43m respectively, which contributed to the low RMSE presented in the next section. The RMSE was used frequently in validating the performance of the VIO, IO and the VO, and to examine the quality of the estimated trajectory. RMSE was derived as:

$$RMSE_f = \sqrt{\frac{x_{d,f}^2 + y_{d,f}^2 + z_{d,f}^2}{3}} \qquad (38)$$

$$v_{d,f} = v_{a,f} - v_{m,f}$$

where $v_{d,f}$ is the deviate position value at frame f, computed by the subtraction of the measured position value at frame f, $v_{m,f}$ and the actual position value at frame f, $v_{a,f}$. The position values are interpreted in term of x-axis, y-axis and z-axis.

It was advisable to analyse the accuracy in a variation of methods especially the RMSE test as it was a standard way to measure error of a model in predicting quantitative data. Fig. 7 illustrated the result obtained from the RMSE test.

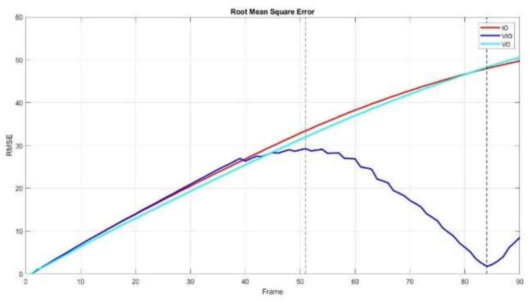

Fig. 7 RMSE of the VIO, IO and VO respectively.

It was observed that the RMSE of VIO, IO and VO increased gradually in the initial stage, however, VIO recorded a significant change after the 52nd frame. The RMSE of VIO decreased gradually whereas the RMSE of IO and VO increased in a similar manner. TABLE II extracted the significant information from Fig. 3.

TABLE II The important information extracted from Figure 4.3.

Odometry	RMSE (m)		Duration (s)
	Peak	Average	
VIO	29.25	16.69	69.13
IO	52.02	30.72	17.40
VO	54.01	30.14	78.32

Based on the extracted result from RMSE test, it was identified that VIO recorded the lowest average error at 16.69m. The effort in combining the key information from both IMU and monocular camera was proven to be effective as it reduced the RMSE of IO by 45.7% and the RMSE of VO by 44.6%. This showed that the VIO was more accurate compare to that of the IO and the VO.

In this work, even though the objectives of the research had been achieved as proven in the results, the proposed method have several limitation in term of system accuracy and performance. Based on Fig. 4, it was observed that there were deviations between the VIO path and the reference path with the greatest average deviation recorded at 27.15m along y-axis, which proved that the system has some space for improvement. Despite that, the fusion of the IMU measurements with the visual information was able to reduce the RMSE of the standalone odometry by more than 40%. At average RMSE of 16.69m, it was certainly hope that the system reliability could be further improved as an ideal case was to travel as close as possible to the ground truth paved by the GPS and score as low as possible in the RMSE test.

V. CONCLUSION

The project is concluded with both the objectives achieved. The proposed method utilized the MSCKF as the filter-based fusing framework in combining data from both IMU and the monocular camera. The implementation was done using MATLAB in estimating the pose of a mobile platform. The respective implementation was conducted by simulating the proposed method on a KITTI dataset which was briefly introduced in the methodology. As discussed in the results and discussions section, the traversed path of the proposed method travelled along the path paved by the ground truth, despite some deviations between the paths, the direction travelled by the path was on track to that of the reference path. Hence, it was deduced that the first objective of the research was fulfilled.

The second objective was to investigate the use of a monocular VIO in improving the accuracy of the respective standalone odometry. Hence, the accuracy of each odometry was examined roughly by comparing the traversed path paved by each odometry. It was identified the VIO proposed the trajectory with the least deviations from the reference path. This was further validated when the RMSE of each odometry was computed with the VIO scored the least in average at 16.69m, which could be visualized from Fig. 5.

In the same section, the effort in combining both standalone odometry, inertial and visual, was proven to be effective as it reduced the RMSE of IO by 45.7% and the RMSE of VO by 44.6%. Despite that, there was a space for improvement of the method proposed in the research, it was certainly hoped that the average RMSE could be reduced to somewhere lower than 10 such as 8 or 9, further increase the reliability of the odometry.

In conclusion, the project was a success with the accomplishment of research objectives and certainly, provided greater insight in the development of odometry.

VI. ACKNOWLEDGEMENT

Foremost, I would like to express my sincere gratitude to my project supervisor Professor Dr. Mohd Zaid Abdullah for the continuous support of my research, for his knowledge and enthusiasm. His guidance assisted me in all the time of research and writing of thesis. Besides, I would like to thank the staff committees of School of Electrical & Electronic, Universiti Sains Malaysia for the support and guidelines provided in the completion of the research. I would like to take this opportunity to appreciate Dr. Wan Mohd Yusof Rahiman Wan Abdul Aziz, for giving me helpful comments and advices on the research.

VII. REFERENCES

[1] S. A. S. Mohamed, M. H. Haghbayan, T. Westerlund, J. Heikkonen, H. Tenhunen, and J. Plosila, “A Survey on Odometry for Autonomous Navigation Systems,” *IEEE Access*, vol. 7, pp. 97466–97486, 2019, doi: 10.1109/ACCESS.2019.2929133.

[2] F. T. Gonz´, “Visual Inertial Odometry for Mobile Robotics,” *Tutorial*, 2015.

[3] M. Li, A. Mourikis, J. Farrell, and W. Ren, “UNIVERSITY OF CALIFORNIA RIVERSIDE Visual-Inertial Odometry on Resource-Constrained Systems,” 2014.

[4] M. O. A. Aqel, M. H. Marhaban, M. I. Saripan, and N. B. Ismail, “Review of visual odometry: types, approaches, challenges, and applications,” *Springerplus*, vol. 5, no. 1, 2016, doi: 10.1186/s40064-016-3573-7.

[5] W. C. Dunn, “Visual-Inertial Odometry with Depth Sensing Using a Multi-State Constraint Kalman Filter,” pp. 1–50, 2013.

[6] R. Giubilato, M. Pertile, and S. Debei, “A comparison of monocular and stereo visual FastSLAM implementations,” *3rd IEEE Int. Work. Metrol. Aerospace, Metroaerosp. 2016 - Proc.*, no. June, pp. 227–232, 2016, doi: 10.1109/MetroAeroSpace.2016.7573217.

[7] J. Delmerico and D. Scaramuzza, “A Benchmark Comparison of Monocular Visual-Inertial Odometry Algorithms for Flying Robots,” *Proc. - IEEE Int. Conf. Robot. Autom.*, pp. 2502–2509, 2018, doi: 10.1109/ICRA.2018.8460664.

[8] X. Qiu, H. Zhang, W. Fu, C. Zhao, and Y. Jin, “Monocular visual-inertial odometry with an unbiased linear system model and robust feature tracking front-end,” *Sensors (Switzerland)*, vol. 19, no. 8, pp. 5–9, 2019, doi: 10.3390/s19081941.

[9] V. Usenko, J. Engel, J. Stuckler, and D. Cremers, “Direct visual-inertial odometry with stereo cameras,” *Proc. - IEEE Int. Conf. Robot. Autom.*, vol. 2016-June, pp. 1885–1892, 2016, doi: 10.1109/ICRA.2016.7487335.

[10] E. Hong and J. Lim, “Visual-inertial odometry with robust initialization and online scale estimation,” *Sensors (Switzerland)*, vol. 18, no. 12, pp. 1–16, 2018, doi: 10.3390/s18124287.

[11] T. Qin, P. Li, and S. Shen, “VINS-Mono: A Robust and Versatile Monocular Visual-Inertial State Estimator,” *IEEE Trans. Robot.*, vol. 34, no. 4, pp. 1004–1020, 2018, doi: 10.1109/TRO.2018.2853729.

[12] R. Munguía, E. Nuño, C. I. Aldana, and S. Urzua, “A Visual-Aided Inertial Navigation and Mapping System,” *Int. J. Adv. Robot. Syst.*, vol. 13, no. 3, pp. 1–13, 2016, doi: 10.5772/64011.

[13] F. Ma, J. Shi, Y. Yang, J. Li, and K. Dai, “ACK-MSCKF: Tightly-coupled ackermann multi-state constraint kalman filter for autonomous vehicle localization,” *Sensors (Switzerland)*, vol. 19, no. 21, 2019, doi: 10.3390/s19214816.

[14] A. I. Mourikis and S. I. Roumeliotis, “A multi-state constraint Kalman filter for vision-aided inertial navigation,” *Proc. - IEEE Int. Conf. Robot. Autom.*, pp. 3565–3572, 2007, doi: 10.1109/ROBOT.2007.364024.

[15] L. E. Clement, V. Peretroukhin, J. Lambert, and J. Kelly, “The Battle for Filter Supremacy: A Comparative Study of the Multi-State Constraint Kalman Filter and the Sliding Window Filter,” *Proc. -2015 12th Conf. Comput. Robot Vision, CRV 2015*, pp. 23–30, 2015, doi: 10.1109/CRV.2015.11.