Project 4 - Visual Inertial Odometry Phase 1

Karthik Mundanad
Robotics Engineering Department
Worcester Polytechnic Institute
Email: krmundanad@wpi.edu

Kushagra Srivastava
Robotics Engineering Department
Worcester Polytechnic Institute
Email: ksrivastava1@wpi.edu

***Abstract*—Phase 1 of the project focuses on implementing a tightly-coupled sensor fusion approach for filter-based stereo visual inertial odometry using multi-state constraint Kalman filter (S-MSCKF). We evaluate and report the performance of S-MSCKF using our custom-formulated functions on the EuRoC dataset.**

I. Phase 1

Specifically, for phase 1, we implemented seven functions of S-MSCKF [1]. The IMU state is defined as:

$$\mathbf{x}_I = \left({}^I_G\mathbf{q}^\top \quad \mathbf{b}_g^\top \quad {}^G\mathbf{v}_I^\top \quad \mathbf{b}_a^\top \quad {}^G\mathbf{p}_I^\top \quad {}^I_C\mathbf{q}^\top \quad {}^I\mathbf{p}_C^\top \right)^\top$$

where the quaternion ${}^I_G\mathbf{q}$ represents the rotation from the inertial frame to the body frame. ${}^G\mathbf{v}_I \in R^3$ and ${}^G\mathbf{p}_I \in R^3$ represent the velocity and position of the body frame in the inertial frame. $\mathbf{b}_g \in R^3$ and $\mathbf{b}_a \in R^3$ are the biases of the measured angular velocity and linear acceleration from the IMU. ${}^I_C\mathbf{q}$ and ${}^I\mathbf{p}_C \in R^3$ represent the relative transformation between the camera frame and the body frame.

A. Initialization of Gravity and Bias

The first 200 IMU messages were used to initialize the gravity (g) and gyroscope bias (b_g) (the robot is at rest). The gyroscope bias is initialized as the average of the of first 200 gyroscope readings while gravity is initialized as the norm of the first 200 accelerometer readings. The gravity vector is thus defined as $\left[0, 0, -g_{norm}\right]$. The initial orientation is defined as the quaternion from $-g$ to $-g_{norm}$.

B. Batch IMU Processing

The batch IMU processing function processes all the IMU messages received until a new image is received from the camera since the operating frequency of an IMU is much higher than that of a camera. This function propagates the state of the IMU by processing the IMU measurements with timestamps less than that of the current image. For each unprocessed IMU message in the buffer, the state of the IMU is propagated using the process model. This function ensures time synchronization between the IMU and Camera message updates.

C. Process Update

As per [1], the IMU state is modeled in continuous time as per the following equations:

$$\begin{aligned} {}^I_G\dot{\hat{\mathbf{q}}} &= \frac{1}{2}\Omega(\hat{\omega}){}^I_G\hat{\mathbf{q}}, \quad \dot{\hat{\mathbf{b}}}_g = \mathbf{0}_{3\times1}, \\ {}^G\dot{\hat{\mathbf{v}}} &= C\left({}^I_G\hat{\mathbf{q}}\right)^\top \hat{\mathbf{a}} + {}^G\mathbf{g}, \\ \dot{\hat{b}}_a &= \mathbf{0}_{3\times1}, \quad {}^G\dot{\hat{\mathbf{p}}}_I = {}^G\hat{\mathbf{v}}, \\ {}^I_C\dot{\hat{\mathbf{q}}} &= \mathbf{0}_{3\times1}, \quad {}^I\dot{\hat{\mathbf{p}}}_C = \mathbf{0}_{3\times1} \end{aligned} \tag{1}$$

where $\hat{\omega}$ and $\hat{\mathbf{a}}$ are the IMU measurements for angular velocity and acceleration respectively with biases removed.

$$\hat{\omega} = \omega_m - \hat{\mathbf{b}}_g \tag{2}$$

$$\hat{\mathbf{a}} = \mathbf{a}_m - \hat{\mathbf{b}}_a \tag{3}$$

$$\Omega\left(\hat{\omega}\right) = \begin{bmatrix} -\lfloor\hat{\omega}_\times\rfloor & \omega \\ -\omega^\top & 0 \end{bmatrix} \tag{4}$$

The linearized dynamics of the error IMU state is formulated as:

$$\dot{\tilde{\mathbf{x}}}_I = \mathbf{F}\tilde{\mathbf{x}}_I + \mathbf{G}\mathbf{n}_I \tag{5}$$

$$\mathbf{F} = \begin{bmatrix} -\lfloor\hat{\omega}_\times\rfloor & -\mathbf{I}_3 & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \\ -C\left({}^I_G\hat{\mathbf{q}}\right)^\top \lfloor\hat{\mathbf{a}}_\times\rfloor & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & -C\left({}^I_G\hat{\mathbf{q}}\right)^\top & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{I}_3 & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \end{bmatrix} \tag{6}$$

$$\mathbf{G} = \begin{bmatrix} -\mathbf{I}_3 & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{I}_3 & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & -C\left({}^I_G\hat{\mathbf{q}}\right)^\top & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{I}_3 \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \end{bmatrix} \tag{7}$$

where $\mathbf{n}_I^\top = \left(\mathbf{n}_g^\top \; \mathbf{n}_{wg}^\top \; \mathbf{n}_a^\top \; \mathbf{n}_{wa}^\top\right)^\top$. $\mathbf{n}_g$ and $\mathbf{n}_a$ represent the Gaussian noise of the gyroscope and accelerometer measurement. $\mathbf{n}_{wg}$ and $\mathbf{n}_{wa}$ are the random walk rate of the gyroscope and accelerometer measurement biases. The discrete-time state transition matrix of Equation 5 and discrete time noise covariance matrix are defined as,

$$\mathbf{\Phi}_k = \mathbf{\Phi}(t_{k+1}, t_k) = \exp\left(\int_{t_k}^{t_{k+1}} \mathbf{F}(\tau)d\tau\right)$$
$$\mathbf{Q}_k = \int_{t_k}^{t_{k+1}} \mathbf{\Phi}(t_{k+1}, \tau)\mathbf{GQG}\mathbf{\Phi}(t_{k+1}, \tau)^\top d\tau$$

The matrix exponential for the continuous time covariance matrix $\mathbf{\Phi}_k$ is approximated up to the 3^{rd} order as per the power series expansion.

$$\mathbf{\Phi}_k = \mathbf{I}_{21\times 21} + F(\tau)d\tau + \frac{1}{2}(F(\tau)d\tau)^2 + \frac{1}{6}(F(\tau)d\tau)^3 \quad (8)$$

The propagated covariance of the IMU state is,

$$\mathbf{P}_{II_{k+1|k}} = \mathbf{\Phi}_k \mathbf{P}_{II_{k|k}} \mathbf{\Phi}_k^\top + \mathbf{Q}_k$$

Partitioning the covariance of the whole state as,

$$\mathbf{P}_{k|k} = \begin{bmatrix} \mathbf{P}_{II_{k|k}} & \mathbf{P}_{IC_{k|k}} \\ \mathbf{P}_{IC_{k|k}}^\top & \mathbf{P}_{CC_{k|k}} \end{bmatrix}$$

the full uncertainty propagation can be represented as,

$$\mathbf{P}_{k+1|k} = \begin{bmatrix} \mathbf{P}_{II_{k+1|k}} & \mathbf{\Phi}_k \mathbf{P}_{IC_{k|k}} \\ \mathbf{P}_{IC_{k|k}}^\top \mathbf{\Phi}_k^\top & \mathbf{P}_{CC_{k|k}} \end{bmatrix}$$

D. Predict New State

Since we are working in discrete time, the dynamics defined in Equation 5 were propagated using 4^{th} order Runge-Kutta numerical integration which for a given ODE $\dot{x}_t(t) = f(t, x_t(t), u(t))$ with initial condition $x_t(t_0) = x_0$ is given by,

$$\begin{aligned} k_1 &= f(x_n, u_n) \\ k_2 &= f\left(x_n + \frac{\Delta t}{2}, \frac{u_n + u_{n+1}}{2}\right) \\ k_3 &= f\left(x_n + \frac{\Delta t}{2}, \frac{u_n + u_{n+1}}{2} + \frac{k_2}{2}\right) \\ k_4 &= f(x_n + \Delta t k_3, u_{n+1}) \\ x_t(t_{n+1}) &= x_t(t_n) + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4) \end{aligned}$$

where $u(t)$ are the control points at the time intervals given by

$$\begin{aligned} u(t_n) &= u_n \\ u\left(t_n + \frac{\Delta t}{2}\right) &= \frac{u_n + u_{n+1}}{2} \\ u\left(t_n + \Delta t\right) &= u_{n+1} \end{aligned}$$

E. State Augmentation

Upon receiving new images, the state should be augmented with a new camera state. The pose of the new camera state is computed from the latest IMU state.

$${}^C_G\hat{\mathbf{q}} = {}^C_I\hat{\mathbf{q}} \otimes {}^I_G\hat{\mathbf{q}}, \quad {}^G\hat{\mathbf{p}}_C = {}^G\hat{\mathbf{p}}_C + C\left({}^I_G\hat{\mathbf{q}}\right)^\top {}^I\hat{\mathbf{p}}_C$$

The covariance matrix is also augmented.

$$\mathbf{P}_{k|k} = \begin{bmatrix} \mathbf{I}_{21+6N} \\ \mathbf{J} \end{bmatrix} \mathbf{P}_{k|k} \begin{pmatrix} \mathbf{I}_{21+6N} \\ \mathbf{J} \end{pmatrix}^\top \quad (9)$$

$$\mathbf{J} = \begin{bmatrix} C\left({}^I_G\hat{\mathbf{q}}\right) & \mathbf{0}_{3\times 9} & \mathbf{0}_{3\times 3} & \mathbf{I}_3 & \mathbf{0}_{3\times 3} \\ -C\left({}^I_G\hat{\mathbf{q}}\right)^\top \lfloor {}^I\hat{\mathbf{p}}_{C\times} \rfloor & \mathbf{0}_{3\times 9} & \mathbf{I}_3 & \mathbf{0}_{3\times 3} & \mathbf{I}_3 \end{bmatrix} \quad (10)$$

F. Adding New Feature Observations

This function adds feature observations from a new image to the map server by creating new map features. Each feature has its feature ID (i) and current state ID (j) and is represented for the stereo set as

$$Z_j^i = \left[u_{j,1}^i, v_{j,1}^i, u_{j,2}^i, v_{j,2}^i\right]^T \quad (11)$$

G. Measurement Update

This function takes measurement matrix H and residual matrix r (see Section III B of [1]) to calculate the Kalman Gain K. Applying QR decomposition on H, we can obtain Q and T_H.

$$H = \begin{bmatrix} Q & Q_2 \end{bmatrix} \begin{bmatrix} T_H \\ O \end{bmatrix} \quad (12)$$

Residual r_n is computed as:

$$r_n = Q^T r = T_H \tilde{X} + n_n \quad (13)$$

The Kalman gain K is computed as:

$$K = PT_H^T(T_H P T_H^T + R_n)^{-1} \quad (14)$$

Since computing matrix inverse is unstable, K can be computed by solving the following system of equations:

$$(T_H P T_H^T + R_n)K^T = T_H P \quad (15)$$

The correction in state and state covariance matrix P is updated by:

$$\Delta X = K r_n \quad (16)$$
$$P_{k+1|k+1} = (I - KT_H)P_{k+1|k} \quad (17)$$

H. Discrepancies in open-source implementation and S-MSCKF

We found two discrepancies in the source code and [1].

- The source code follows Equation 17, however, in the paper (it adopts the modeling from [2]), the covariance update is given by:

  $$P_{k+1|k+1} = (I - KT_H)P_{k+1|k}(I - KT_H)^T + KR_nK^T \quad (18)$$

  We observed that our code did not work if the above equation was used.
- The source code implements a different equation for J in Equation 10. However, we followed the definition in [1] for our implementation. The difference arises due to different conventions for representing quaternions. The authors use the Hamilton convention to derive J but their implementation follows the JPL convention.

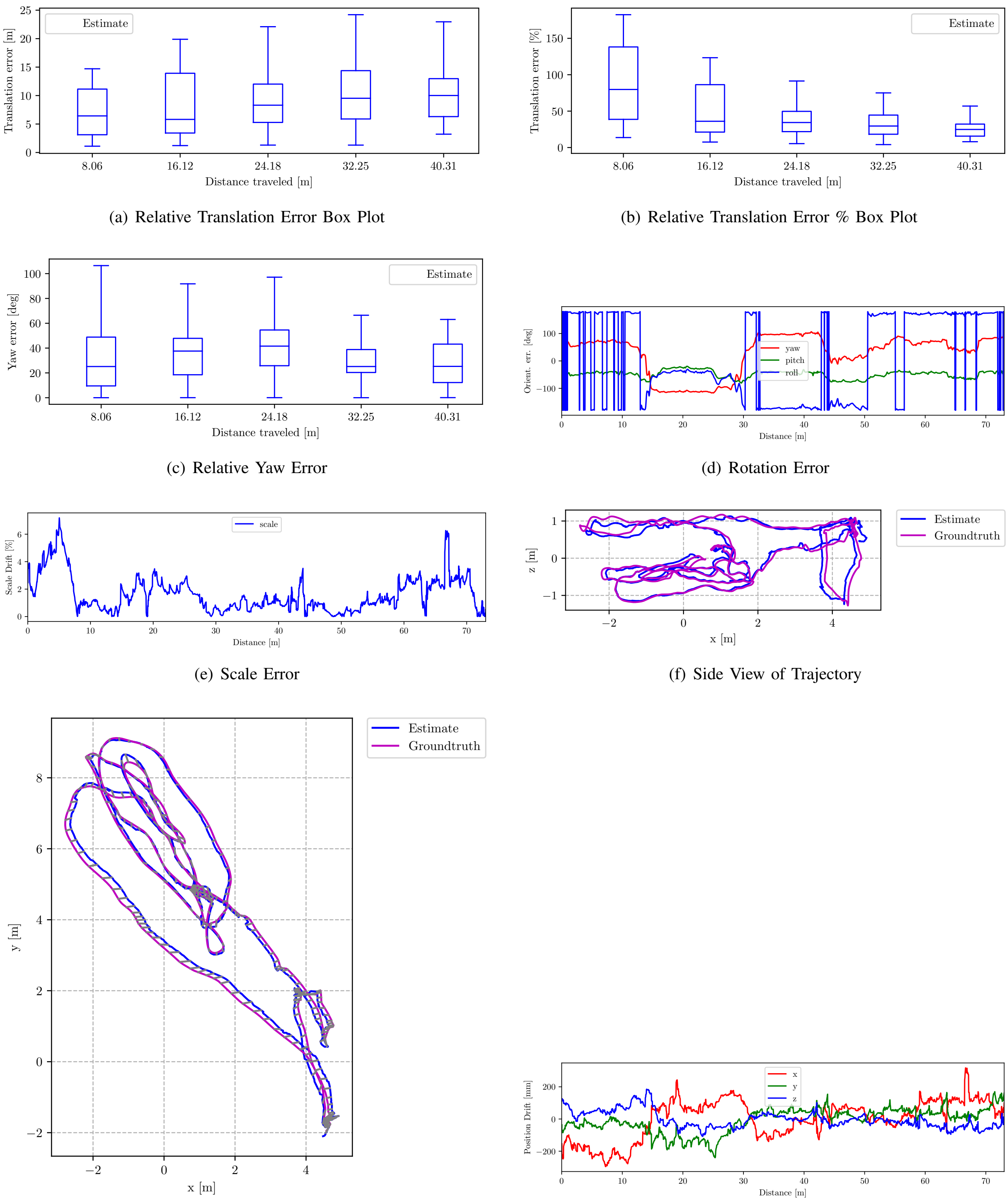
I. Results

We used an open-source repository to compute the error metrics and the plots. The absolute median trajectory error (ATE) is 0.1399m and the root mean square translation error (RMSE) is 0.1543m.

References

[1] K. Sun, K. Mohta, B. Pfrommer, M. Watterson, S. Liu, Y. Mulgaonkar, C. J. Taylor, and V. Kumar, "Robust stereo visual inertial odometry for fast autonomous flight," 2018. 1, 2

[2] A. I. Mourikis and S. I. Roumeliotis, "A multi-state constraint kalman filter for vision-aided inertial navigation," in *Proceedings 2007 IEEE international conference on robotics and automation*. IEEE, 2007, pp. 3565–3572. 2



(a) Relative Translation Error Box Plot

(b) Relative Translation Error % Box Plot

(c) Relative Yaw Error

(d) Rotation Error

(e) Scale Error

(f) Side View of Trajectory

(g) Top View of Trajectory

(h) Translation Error

Fig. 1: Results for our implementation of S-MSCKF