RBE/CS 549 Project 4
Deep and Un-Deep Visual Inertial Odometry

Blake Bruell
Worcester Polytechnic Institute
babruell@wpi.edu

Cole Parks
Worcester Polytechnic Institute
cparks@wpi.edu

***Abstract*—In this project, we implement a visual-inertial odometry system that fuses data from IMUs and cameras to estimate the motion of a robot using. In Phase 1 we implement the stereo Multi-State Constraint Kalman Filter (MSCKF) to estimate the state of the robot, using stereo cameras, evaluated on the EuRoC dataset.**

I. Phase 1

A. Initializing Gravity and Bias

A six-degree-of-freedom IMU, used to measure rotation and acceleration using a gyroscope and accelerometer, necessitates calibration to correct bias in the sensors. This is done by calculating the mean of a set of stationary readings to determine the bias, since the only force acting on the accelerometer is gravity and there should be no torques acting on the gyroscope. The gyro bias is then subtracted from all subsequent gyroscope readings, and the gravity is initialized as [0,0, -gravity_norm]. This estimation of gravity is not perfect, so the gravity vector is updated during the filter process.

B. Batch IMU Processing

Since the features and the IMU data do not come in at the same rate, we want to batch IMU messages. In practice, when feature is received, all IMU messages in the IMU message buffer which have a timestamp prior to the feature's timestamp are processed using the IMU process model. This is ensures that the IMU state is updated to the time of the feature observation.

C. Process Model

Batch processing is responsible for processing IMU messages and updating the IMU state within a specified time frame. It iterates through the IMU messages, discarding those already processed and stopping at the time bound. For each unprocessed message, it applies the process model to update the IMU state based on angular velocity and linear acceleration measurements. The function updates the IMU state's timestamp and ID, and removes processed messages from the buffer. This function is crucial for accurate state propagation and synchronization between the IMU and Visual Odometry components, contributing to reliable sensor fusion and localization.

Mathematically, the process model can be described as follows [1]:

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

where ${}^{I}_{G}\dot{\hat{\mathbf{q}}}(t)$ is the unit quaternion which described the rotation from the global from (G) to the IMU frame (I), $\hat{\boldsymbol{\omega}} \in \mathbb{R}^3$ and $\hat{\mathbf{a}} \in \mathbb{R}^3$ are the IMU measurements of angular velocity and acceleration respectively with the biases removed.

$$\Omega(\hat{\boldsymbol{\omega}}) = \begin{bmatrix} -\lfloor \hat{\boldsymbol{\omega}}_{\times} \rfloor & \boldsymbol{\omega} \\ -\boldsymbol{\omega}^{T} & 0 \end{bmatrix}$$

where $\lfloor \hat{\boldsymbol{\omega}}_{\times} \rfloor$ is the skew symmetric matrix of $\hat{\boldsymbol{\omega}}$.

Based on Equation 1, we get the following linearized continuous dynamics for the error IMU state:

$$\dot{\hat{x}}_I = \mathbf{F}\hat{x}_I + \mathbf{G}\mathbf{n}_I^{\top} \tag{2}$$

where $\mathbf{n}_I^{\top} = \begin{pmatrix} \mathbf{n}_g^{\top} & \mathbf{n}_{wg}^{\top} & \mathbf{n}_a^{\top} & \mathbf{n}_{wa}^{\top} \end{pmatrix}$ is the noise vector and $\mathbf{F}$ and $\mathbf{G}$ are the state transition matrix and the noise matrix respectively.

$\mathbf{F}$ is given by:

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

and $\mathbf{G}$ is given by:

$$\mathbf{G} = \begin{bmatrix} -\mathbf{I}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{I}_3 & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & -C({}^{I}_{G}\hat{\mathbf{q}})^{\top} & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{I}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 \end{bmatrix} \tag{4}$$

D. Predict New State

When new images and IMU readings are received, we predict the new state using an Extended Kalman Filter, which utilizes a 4th order Runge-Kutta integration to update the IMU's state based on new accelerometer and gyroscope data. This function calculates the norm of the gyroscope measurements and sets up the Omega matrix to update the IMU's orientation quaternion. Depending on the gyroscope norm, it adjusts the quaternion calculation for numerical stability. The method computes intermediate values for velocity and position using the Runge-Kutta method, applying transformations based on the IMU's current state and corrected acceleration. The final updated state, including orientation, velocity, and position, is then recalculated and stored back into the state server, readying the system for subsequent updates.

E. State Augmentation

State Augmentation adds a new camera pose and updates the covariance matrix when a new image is received

In visual-inertial odometry systems, the state augmentation function is crucial for integrating the most recent camera state updates based on the latest IMU data. The function specifically adjusts the camera's position (GpC) and orientation (C_Gq) using the previous IMU state information. The pose of the camera is computed using the following equations [2]:

$$GpC = GpI + C(C_Gq)^T IpC$$

$$C_Gq = C_Iq \otimes I_Gq$$

Where GpC represents the global position of the camera, GpI is the global position of the IMU, $C(C_Gq)^T$ is the rotation matrix derived from the quaternion describing the camera's orientation relative to the global frame, and IpC is the relative position vector from the IMU to the camera. The quaternion operation $\otimes$ signifies quaternion multiplication, which is used to combine the orientation of the IMU (I_Gq) and the relative orientation from the IMU to the camera (C_Iq).

Following the calculation of the new camera pose, the state covariance matrix P is augmented to reflect the updated state uncertainty. This is achieved through a Jacobian matrix J, which maps the influence of the new camera state onto the overall system uncertainty:

$$J = \begin{bmatrix} C(I_{Cq}) & 0_{3\times9} & 0_{3\times3} & I_3 & 0_{3\times3} \\ \lfloor C(I_{Gq})^T I_{pC\times} \rfloor & 0_{3\times9} & I_3 & 0_{3\times3} & I_3 \end{bmatrix}$$

The updated state covariance matrix $P_{k|k}$ is computed as:

$$P_{k|k} = JP_{K|K}J^T$$

This matrix J effectively accounts for the effects of camera motion relative to the IMU, ensuring that updates in camera position and orientation are accurately reflected in the state covariance.

F. Adding Feature Observation

After a feature is detected, it is added to the feature map server, which is done by getting the current IMU state ID and number of features in the map server, and iterating over all of the new features, creating new features for unseen features and updating existing features. After, the tracking rate is updated.

G. Measurement Update

For the measurement update, we first decompose the Jacobian to reduce its computational complexity using QR decomposition. We then calculate the residual between the predicted and observed feature positions, and compute the Kalman gain and update the state and covariance. The residual is calculated by subtracting the predicted feature position from the observed feature position. The state is then updated by adding the Kalman gain multiplied by the residual, and the covariance is updated by subtracting the Kalman gain multiplied by the observation model from the identity matrix. Also, the IMU state, extrinsics, and camera states are updated by applying small angle quaternion operations.

H. Results and Discussion

The graphs and images are the results of running our VIO on the MH_01_easy EuRoC dataset are shown in Appendix A. The trajectory and relative errors throughout the run are shown.

References

[1] 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*, 2007, pp. 3565–3572. DOI: 10.1109/ROBOT.2007.364024.

[2] K. Sun, K. Mohta, B. Pfrommer, *et al.*, "Robust stereo visual inertial odometry for fast autonomous flight," *CoRR*, vol. abs/1712.00036, 2017. arXiv: 1712.00036. [Online]. Available: http://arxiv.org/abs/1712.00036.

APPENDIX A
CLASSICAL VIO RESULTS

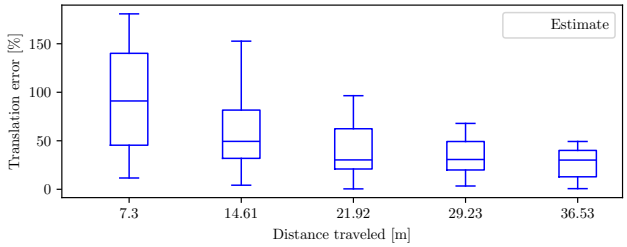

Fig. 1: Relative Translation Error Percentage

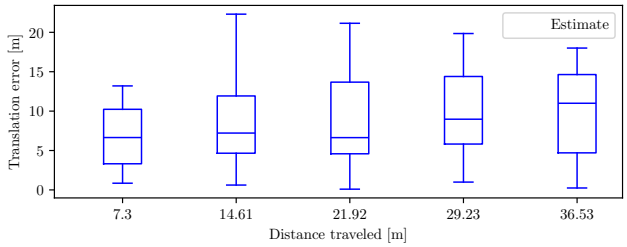

Fig. 2: Relative Translation Error

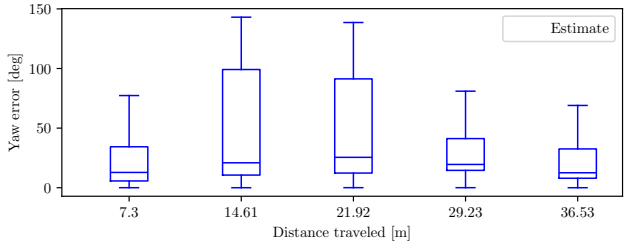

Fig. 3: Relative Yaw Error

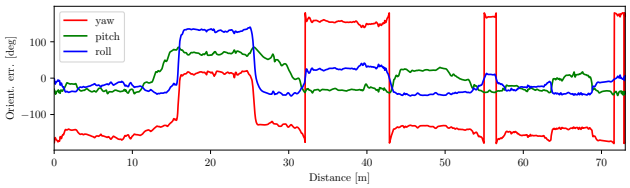

Fig. 4: Rotation Error

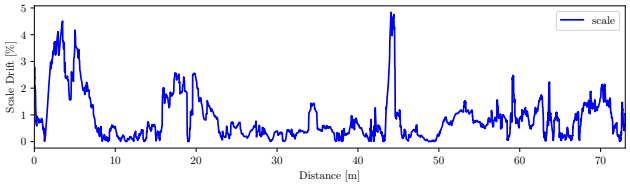

Fig. 5: Scale Error

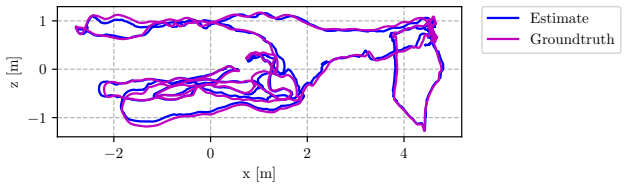

Fig. 6: Trajectory Side

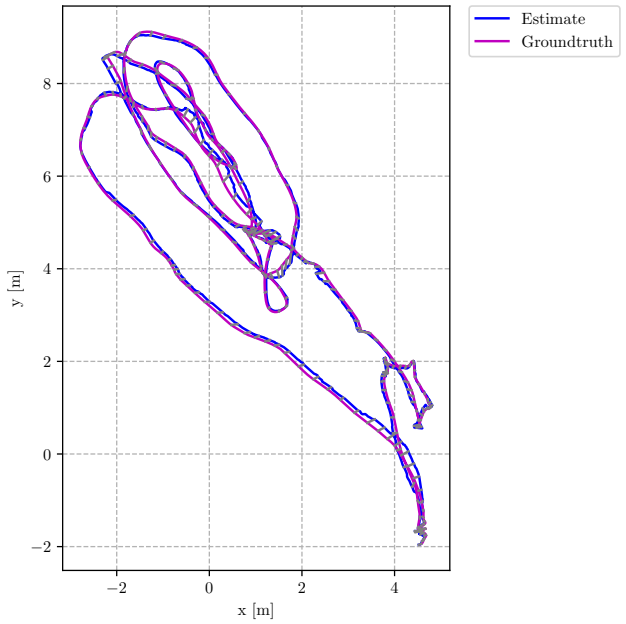

Fig. 7: Trajectory Top

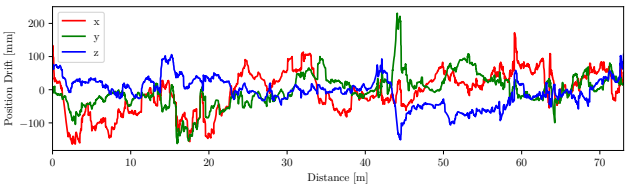

Fig. 8: Translation Trror