Visual-Inertial Tracking using Pre-integrated Factors

Tarlan Bakirli, Yihao Wang

March 14, 2023

Abstract

There have been taken big leaps in Visual Odometry which estimates position and orientation of an agent using camera frames. Even though camera outputs are very precise in slow motion scenarios, it tends to become not accurate if the agent is moving in high speed. On the other hand, IMU provide precise data in high speed scenarios even though it has large uncertainty in slow motion. IMU data are complementary to camera frames and result in much more robust estimates. In this project, we implemented naive visual inertial odometry pipeline to achieve much more robust estimations by preintegrating IMU measurements alongside camera frames. We utilise functionalities from basalt library to integrate imu data and do bundle adjustment on the last few frames. Our VIO pipeline shows better results for easy sequences of the chosen dataset compared to Visual Odometry results. Codes can be found on GitLab [1].

1 Introduction

In industrial applications, robots and MAVs (Micro Air Vehicles) often require one or more sensors in addition to a camera, such as an IMU, LiDAR, and others. Therefore, data fusion techniques can be used to obtain more precise and accurate estimations. As a representative, IMUs were introduced by Todd Lupton and Salah Sukkarieh in [1], and VIORB-SLAM was proposed by Rául Mur-Artel and Juan D. Tardós in [2].

While traditional visual odometry has been widely used, it has some limitations. First, it relies only on camera frames. In the case of a monocular setup, it may face the challenge of an unknown scale. Even with binocular cameras, the frame rate is limited by current technology, which is around 100Hz. Additionally, the precision of cameras can be a problem when the camera moves at high speed, leading to blurry images or, in the worst case, too few matched inliers between two keyframes, resulting in a sudden drift in the estimated camera trajectory.

IMUs, on the other hand, can complement the camera's capabilities. IMUs provide measurements at a high output rate, around 1000 Hz, and are relatively precise and accurate in high-speed scenarios at a small range. In contrast, when the body is still, the images are clear and accurate, which can help correct the small drift of IMU measurements.

By incorporating an IMU, we can obtain more robust motion estimates, although it introduces additional complexity due to motion computation. Depending on the data type, VIO frameworks can be loosely or tightly coupled. In the former, camera and IMU estimates are computed separately and then fused together, whereas in the latter, camera and IMU data are fused first before performing state estimation.

For the VIO pipeline, there are three different algorithms: full smoothers, fixed-lag smoothers, and filtering approaches. Full smoothers (or batch nonlinear least-squares algorithms) estimate the complete history of poses, whereas fixed-lag smoothers (or sliding window estimators) consider a window of the latest poses, and filtering approaches only estimate the latest state.

[1] Codes: https://visnav-ws22.gitlab.vision.in.tum.de/visnav_ws22/team3/visnav_ws22

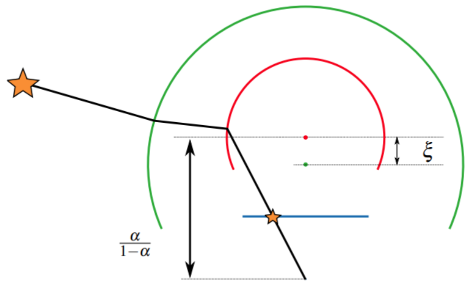

Figure 1: Double Sphere model proposed in [3].

Both fixed-lag smoothers and filters marginalize older states and absorb the corresponding information in a Gaussian prior [2]. In our experiment, considering both the complexity and effectivity, we used the sliding window estimators.

In summary, our contributions are:

- Realized a tightly-coupled visual-inertial odometry using a camera and IMU for stability and scale observability.
- Pre-integrated IMU measurements between frames to estimate IMU pose, biases, and velocity.
- Optimized the position of landmarks and the pose of the camera/IMU through bundle adjustment.
- Visualized the frames, estimated trajectory, and ground truth.
- Evaluated the approach using ATE and RPE, achieving a decrease from 0.149 (without IMU) to 0.115 (with IMU) in ATE.

Overall, our approach improves the accuracy and robustness of motion estimation for MAVs by combining the strengths of a camera and an IMU. The results demonstrate the effectiveness of our method for real-world applications.

2 Visual-Inertial Preliminaries

Our Visual-Inertial Odometry system operates on a continuous stream of binocular camera frames and IMU measurements. To fuse these two sources of data, we utilize several mathematical concepts that require a solid understanding of geometric principles. Therefore, before delving into the specifics of our implementation, we will review some important geometric concepts that will be useful in understanding the fusion process.

We use a double-sphere camera model [3] to project 3D points $\mathbf{X}_\mathrm{C} \in \mathbb{R}^3$ in the camera reference frame C, onto the 2D image plane $\mathbf{x}_\mathrm{C} \in \Omega \subset \mathbb{R}^2$. The structure of the double sphere model is as Figure 1. This projection is done using a projection function $\pi : \mathbb{R}^3 \to \Omega$, given by:

$$\pi(\mathbf{x}, \mathbf{i}) = \begin{bmatrix} f_x \frac{x}{\alpha d_2 + (1-\alpha)(\xi d_1 + z)} \\ f_y \frac{y}{\alpha d_2 + (1-\alpha)(\xi d_1 + z)} \end{bmatrix} + \begin{bmatrix} c_x \\ c_y \end{bmatrix}, \tag{1}$$

$$d_1 = \sqrt{x^2 + y^2 + z^2}, \tag{2}$$

$$d_2 = \sqrt{x^2 + y^2 + (\xi d_1 + z)^2}. \tag{3}$$

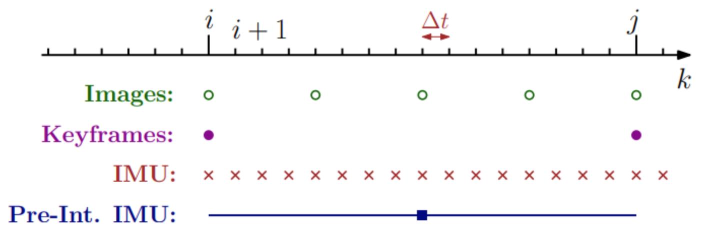

Figure 2: Different rates for IMU and camera [4].

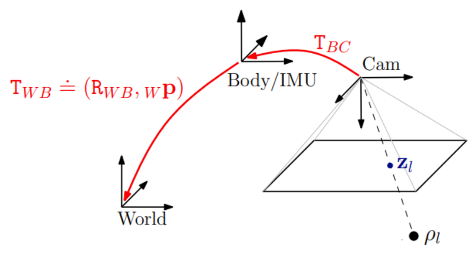

Figure 3: The pose of the body frame B w.r.t. the world frame W. We assume that the body frame coincides with the IMU frame [4].

Here, $[f_x, f_y]^T$ is the focal length, $[c_x, c_y]^T$ is the principal point, ξ is the shifted length of the centers, and α is the parameter for pinhole projection. We undistort the coordinates of the extracted keypoints from the image, so that they can be matched to the projected points using equation 1.

To provide more information for the motion estimation, we use the IMU measurements. Our experiment data is tight coupled, which means for every frame, there is an IMU data can be exactly matched. Figure 2 shows the data between frame i and frame j. Then, we utilize the preintegration. A schematic representation of the preintegration and connectivity of the factor graph underlying the VIO problem is given in Figure 4. The motion can be expressed in terms of the preintegration $\Delta\mathbf{R}$, $\Delta\mathbf{v}$, and $\Delta\mathbf{p}$, which are computed from all the measurements. We use the IMU preintegration method described in [4] to calculate these values:

$$\begin{aligned}
\mathbf{R}_{\mathtt{WB}}^{i+1} =& \mathbf{R}_{\mathtt{WB}}^{i} \Delta\mathbf{R}_{i,i+1} \mathrm{Exp}\left(\left(\mathbf{J}_{\Delta R}^{g} \mathbf{b}_g^i\right)\right) \\
{}_{\mathtt{W}}\mathbf{v}_{\mathtt{B}}^{i+1} =& {}_{\mathtt{W}}\mathbf{v}_{\mathtt{B}}^{i} + \mathbf{g}_{\mathtt{W}} \Delta t_{i,i+1} \\
& + \mathbf{R}_{\mathtt{WB}}^{i} \left(\Delta\mathbf{v}_{i,i+1} + \mathbf{J}_{\Delta v}^{g} \mathbf{b}_g^i + \mathbf{J}_{\Delta v}^{a} \mathbf{b}_a^i\right) \\
{}_{\mathtt{W}}\mathbf{p}_{\mathtt{B}}^{i+1} =& {}_{\mathtt{W}}\mathbf{p}_{\mathtt{B}}^{i} + {}_{\mathtt{W}}\mathbf{v}_{\mathtt{B}}^{i} \Delta t_{i,i+1} + \frac{1}{2} \mathbf{g}_{\mathtt{W}} \Delta t_{i,i+1}^2 \\
& + \mathbf{R}_{\mathtt{WB}}^{i} \left(\Delta\mathbf{p}_{i,i+1} + \mathbf{J}_{\Delta p}^{g} \mathbf{b}_g^i + \mathbf{J}_{\Delta p}^{a} \mathbf{b}_a^i\right)
\end{aligned} \tag{4}$$

where the Jacobians $\mathbf{J}_{(\cdot)}^{a}$ and $\mathbf{J}_{(\cdot)}^{g}$ can approximate the effect of changing the biases without recomputing the preintegrations. Both preintegrations and Jacobians can be calculated efficiently as IMU measurements arrive.

In addition, we assume that the IMU frame "B" coincides with the body frame we want to track, and that the transformation between the camera and the IMU is fixed and known from prior calibration the camera and IMU are assumed to be fixed together, the calibration process determines the transformation $\mathbf{T}_{\mathrm{WB}} = [\mathbf{R}_{\mathrm{WB}} | {}_{\mathrm{W}}\mathbf{p}_{\mathrm{B}}]$ between their coordinate systems.

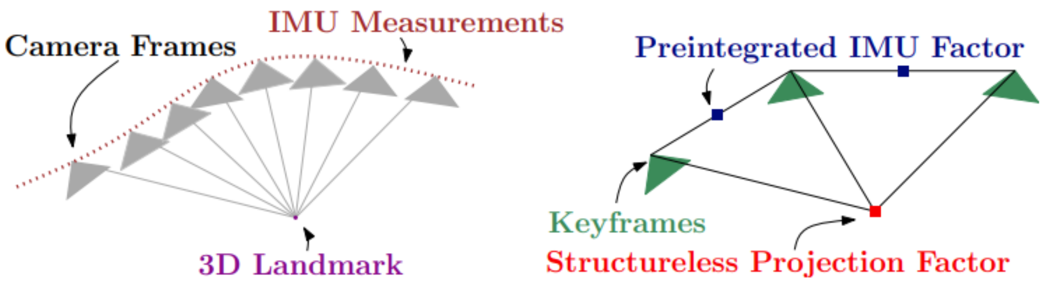

Figure 4: Left: visual and inertial measurements in VIO. Right: factor graph in which several IMU measurements are summarized in a single preintegrated IMU factor and a structureless vision factor constraints keyframes observing the same landmark [4].

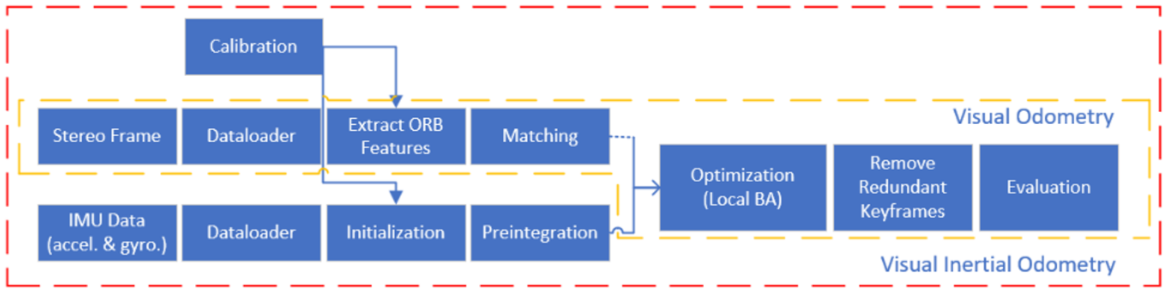

Figure 5: Visual Odometry (yellow) and Visual Inertial Odometry (red) Pipeline.

3 Visual-Inertial Odometry

Our pipeline, as shown in Figure 5, involves both Visual Odometry and Visual Inertial Odometry for comparison and evaluation. Our pipeline starts by loading IMU measurements in addition to camera frames which was already implemented in base project. After we initialize constants as well as variables to be estimated, we preintegrate IMU measurements into camera frames. Then we combine our variables which needs to be optimized with variables from VO and do local bundle adjustment as optimization. We only optimize for 7 keyframes and 3 frames. Finally, we do evaluation based on absolute trajectory error and relative pose error.

3.1 Data Loading

Different from the traditional visual odometry, which needs camera data only, IMU data also need to be loaded here. The original camera data contain the timestamps and the images, while the IMU data contain the timestamps, acceleration and angular velocity.

For evaluation, we also need to load the ground truth. The ground truth data give the information on position, velocity and bias for the accelerator and gyroscope.

After loading the data as strings, we save them on an unordered map, in which the key is the timestamp. Therefore, we could effectively call them in the further steps.

3.2 Calibration

Before initialization, we need to implement a stereo camera calibration using Ceres and the camera models. we can project it using the current camera calibration and estimate of camera position. By minimizing the difference between detected points and projected points

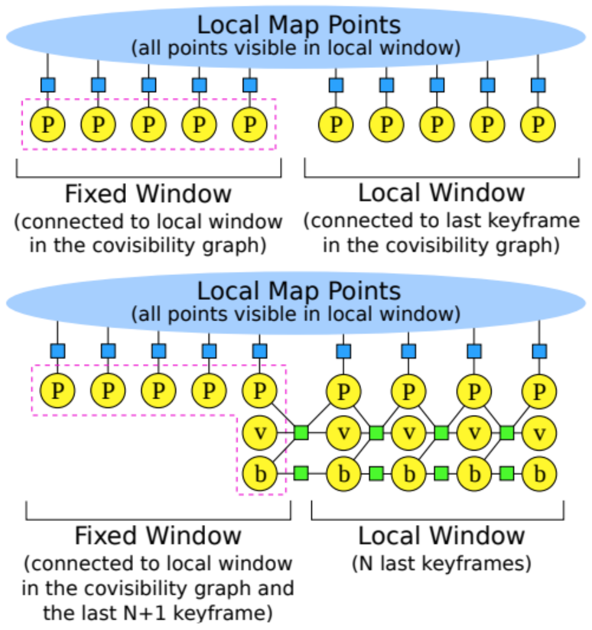
Local Map Points
(all points visible in local window)
P
Fixed Window
(connected to local window in the covisibility graph)
Local Window
(connected to last keyframe in the covisibility graph)
Local Map Points
(all points visible in local window)
P
v
b
Fixed Window
(connected to local window in the covisibility graph and the last N+1 keyframe)
Local Window
(N last keyframes)

Figure 6: Top: Visual Odometry's Local BA. Bottom: Visual Inertial Odometry's Local BA [2].

we can perform the stereo camera calibration. Apart from this, we also need to calibrate for IMU.

That's why, in the real-world implementation, in order not to re-implement IMU calibration, we have leveraged basalt [5] to obtain calibration parameters. Firstly, *basalt_calibrate* then *basalt_calibrate_imu* commands are executed with default values to get camera and IMU calibration parameters respectively. Then we have manually updated our calibration file with obtained parameters.

3.3 IMU Initialization

The biases of the gyroscope and acceleration are tiny and our pipeline is naive implementation of VIO. Therefore, for computation efficiency, we set bias_gyro and bias_accel as {0, 0, 0}. Additionally, because our binocular cameras can achieve the scale by image processing, there is no need for initialization for scale.

For gravity, we assign the gravity direction and world frame as g_I = {0,0,-1} and set g = {0, 0, -9.81}. For pose estimation, we use the initial rotation from acceleration and consider the initial translation as a zero vector. In this way, we assign the coordinate of IMU. Since in some sequences, the initial few frames are still and therefore, we set the initial velocity as {0, 0, 0}.

3.4 Preintegration

Preintegration is the first building block of Visual Inertial Odometry. IMU measurements have to be integrated among frames as shown visually in Figure [2] so that joint optimization results in more accurate estimates. We perform preintegration as in the equation 2. In practice, we directly use the method in Basalt [5], which omitted the bias of the accelerator and gyroscope. Even though [2] integrates IMU measurements per keyframe, we integrate per frame. In our case doing integration per keyframe results in drift of the IMU measurements, because distance between keyframes are large. After preintegration, we obtain the estimated

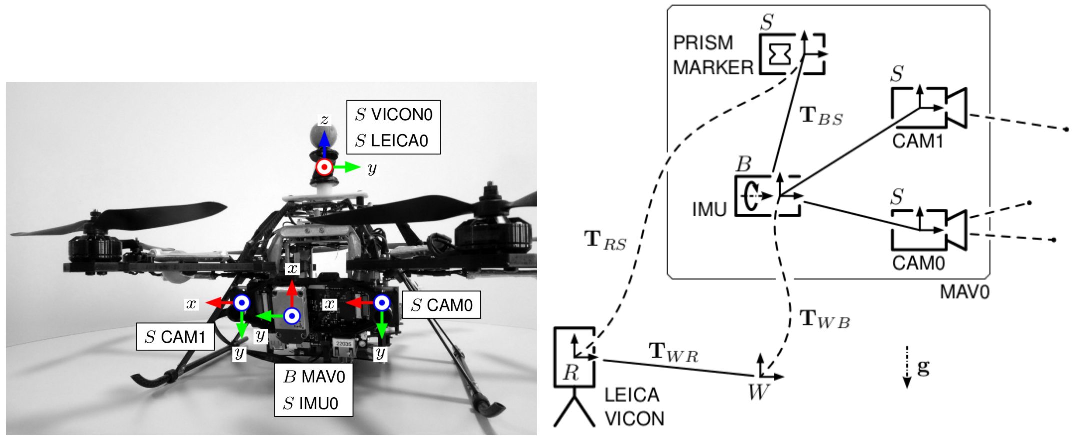

Figure 7: Platform and sensors.

velocity and position for every frame.

3.5 Optimization

After preintegration, we optimize the current frame j by minimizing the feature reprojection error of all matched points and an IMU error term. Different from the normal visual odometry, that chooses a certain number of camera keyframes for optimization, we here choose fewer keyframes and just a few frames for IMU. Initially we choose 7 keyframes and 3 frames, and then we adjust them according to the experiments results.

When tracking is performed just after a map update the IMU error term links current frame j with last keyframe i:

$$\begin{aligned}\theta &= \left\{\mathbf{R}^j_{\mathtt{WB}}, {}_{\mathtt{W}}\mathbf{p}^j_{\mathtt{B}}, {}_{\mathtt{W}}\mathbf{v}^j_{\mathtt{B}}, \mathbf{b}^j_g, \mathbf{b}^j_a\right\} \\ \theta^* &= \underset{\theta}{\operatorname{argmin}} \left(\sum_k \mathbf{E}_{\text{proj}}(k,j) + \mathbf{E}_{\text{IMU}}(i,j)\right) \end{aligned} \tag{5}$$

In which the reprojection error between keyframes k and j is

$$\begin{aligned}\mathbf{E}_{\text{proj}}(k,j) &= \rho\left(\left(\mathbf{x}^k - \pi(\mathbf{X}^k_{\mathtt{C}})\right)^T \mathbf{\Sigma}_{\boldsymbol{k}} \left(\mathbf{x}^k - \pi(\mathbf{X}^k_{\mathtt{C}})\right)\right) \\ \mathbf{X}^k_{\mathtt{C}} &= \mathbf{R}_{\mathtt{CB}}\mathbf{R}^j_{\mathtt{BW}}\left(\mathbf{X}^k_{\mathtt{W}} - {}_{\mathtt{W}}\mathbf{p}^j_{\mathtt{B}}\right) + {}_{\mathtt{C}}\mathbf{p}_{\mathtt{B}} \end{aligned} \tag{6}$$

and the IMU error is

$$\begin{aligned}\mathbf{E}_{\text{IMU}}(i,j) &= \rho\left(\left[\mathbf{e}^T_R\, \mathbf{e}^T_v\, \mathbf{e}^T_p\right] \mathbf{\Sigma}_I \left[\mathbf{e}^T_R\, \mathbf{e}^T_v\, \mathbf{e}^T_p\right]^T\right) + \rho\left(\mathbf{e}^T_b \mathbf{\Sigma}_R \mathbf{e}_b\right) \\ \mathbf{e}_R &= \operatorname{Log}\left(\left(\Delta\mathbf{R}_{ij}\operatorname{Exp}\left(\mathbf{J}^g_{\Delta R}\mathbf{b}^j_g\right)\right)^T \mathbf{R}^i_{\mathtt{BW}}\mathbf{R}^j_{\mathtt{WB}}\right) \\ \mathbf{e}_v &= \mathbf{R}^i_{\mathtt{BW}}\left({}_{\mathtt{W}}\mathbf{v}^j_{\mathtt{B}} - {}_{\mathtt{W}}\mathbf{v}^i_{\mathtt{B}} - \mathbf{g}_{\mathtt{W}}\Delta t_{ij}\right) \\ &\quad - \left(\Delta\mathbf{v}_{ij} + \mathbf{J}^g_{\Delta v}\mathbf{b}^j_g + \mathbf{J}^a_{\Delta v}\mathbf{b}^j_a\right) \\ \mathbf{e}_p &= \mathbf{R}^i_{\mathtt{BW}}\left({}_{\mathtt{W}}\mathbf{p}^j_{\mathtt{B}} - {}_{\mathtt{W}}\mathbf{p}^i_{\mathtt{B}} - {}_{\mathtt{W}}\mathbf{v}^i_{\mathtt{B}}\Delta t_{ij} - \frac{1}{2}\mathbf{g}_{\mathtt{W}}\Delta t^2_{ij}\right) \\ &\quad - \left(\Delta\mathbf{p}_{ij} + \mathbf{J}^g_{\Delta p}\mathbf{b}^j_g + \mathbf{J}^a_{\Delta p}\mathbf{b}^j_a\right) \\ \mathbf{e}_b &= \mathbf{b}^j - \mathbf{b}^i \end{aligned} \tag{7}$$

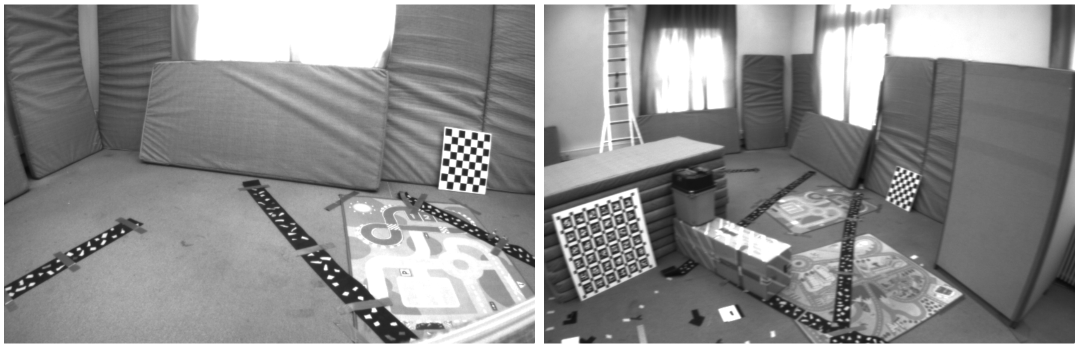
Figure 8: The examples of V1 Dataset.

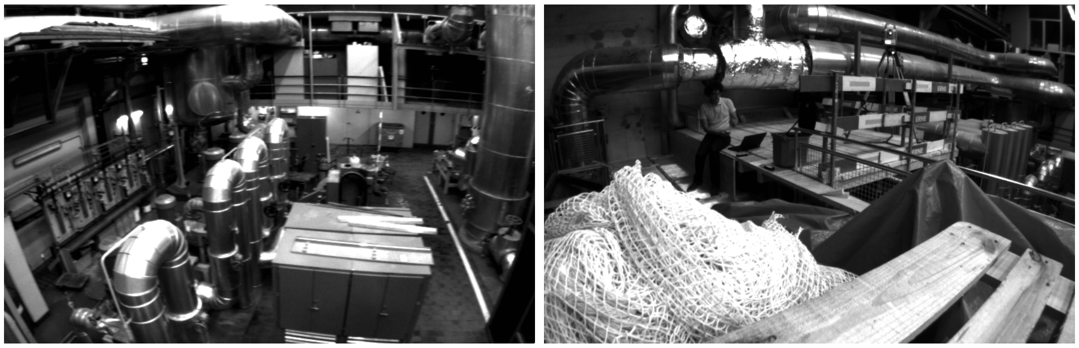
Figure 9: The examples of MH Dataset.

where $\mathbf{\Sigma}_I$ is the information matrix of the preintegration and $\mathbf{\Sigma}_R$ of the bias random walk, and ρ is the Huber robust cost function [2]. We compute the IMU residual function via Basalt [5] and solve this optimization problem with Ceres [6].

4 Experiments

4.1 Evaluation Metrics

After optimization, we evaluate the accuracy of our Visual-Inertial Odometry using Absolute Trajectory Error (ATE) and Relative Projection Error (RPE). Both metrics are shown in translation Root Mean Square Error (RMSE).

The ATE measures the difference between points of the true and the estimated trajectory, which is the performance of SLAM. It first associates the estimated poses with ground truth, then aligns the estimated trajectory with the ground truth using singular value decomposition, and finally computes the difference between each pair of poses and ground truth using translation RMSE.

Comparably, the RPE measures the drift of a visual odometry system, for example, the drift per second. It shows the performance of visual odometry. It computes the error of relative motion between timestamp pairs, and then considers rotation and translation (pose).

The RMSE equation is used to calculate the error in both cases:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2} \tag{8}$$

In this project, we have implemented ATE which includes aligning SE3 transformation

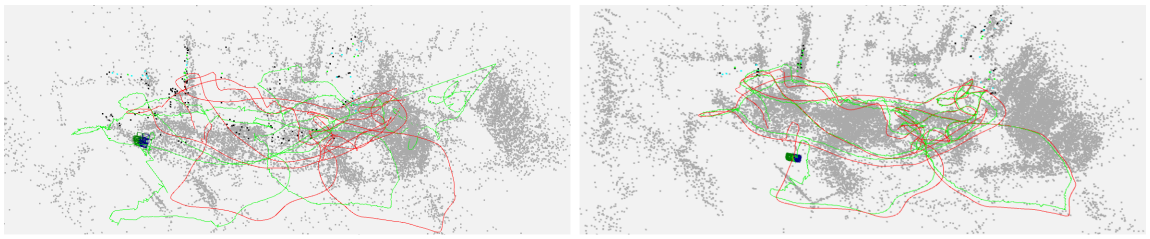

Figure 10: Results on V1_01_easy sequence. Figure on the left shows result without IMU, figure on the right shows result with IMU.

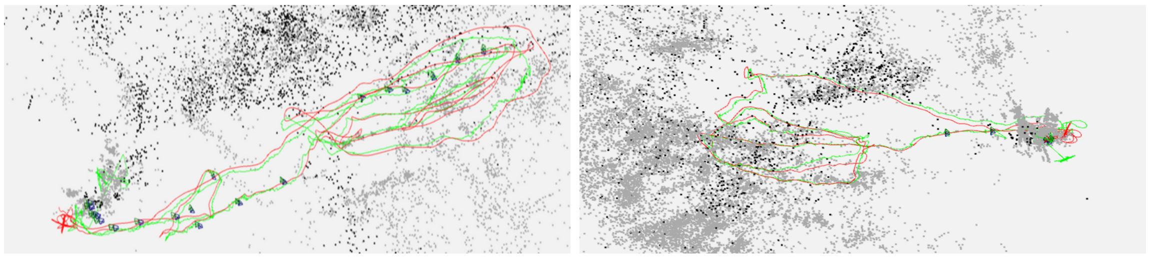

Figure 11: Figure on the left shows results on MH_01_easy sequence with IMU data. Figure on the right shows results on MH_02_easy sequence with IMU data.

between estimated and ground truth associations. On the other hand we haven't implemented RPE measurement tool ourselves, instead leveraged existing implementation [7].

4.2 Dataset

We evaluate our Visual-Inertial Odometry in the EuRoC dataset [8]. It contains 11 sequences recorded from a micro aerial vehicle (MAV), flying around two different rooms and an industrial environment. They are classified as easy, medium and difficult, according to the illumination, motion speed, etc. We only evaluate our odometry on Vicon Room 1 and Machine Hall Easy and Medium Sequences which are shown in Figure 8 and Figure 9 respectively.

Each sequence in EuRoC dataset contains a pair of binocular camera images, IMU data, and ground truth. Figure 7 shows the coordinates between different sensors.

4.3 Results

We have run both visual odometry and visual inertial inertial odometry (ours) in all sequences in Table 1. We have run all combinations multiple times and picked the maximum performing results. Absolute Trajectory Error drops substantially in V1_01_easy sequence. This is achieved by IMU data constraining camera motion which leads to robust and stable motion. We can see clearly from Figure 10 left that estimated trajectory differs from ground truth trajectory a lot while Figure 10 right shows estimated and ground truth trajectories are similar. On the other hand, since MH sequences are more difficult than Vicon Room sequences, ATE of VIO is slightly smaller in MH_01_Easy sequence than ATE of VO. We can also easily see from Figure 12 that as the difficulty is increased VIO starts to fail. This occurs due to the fact that our solution is naive implementation of IMU integration.

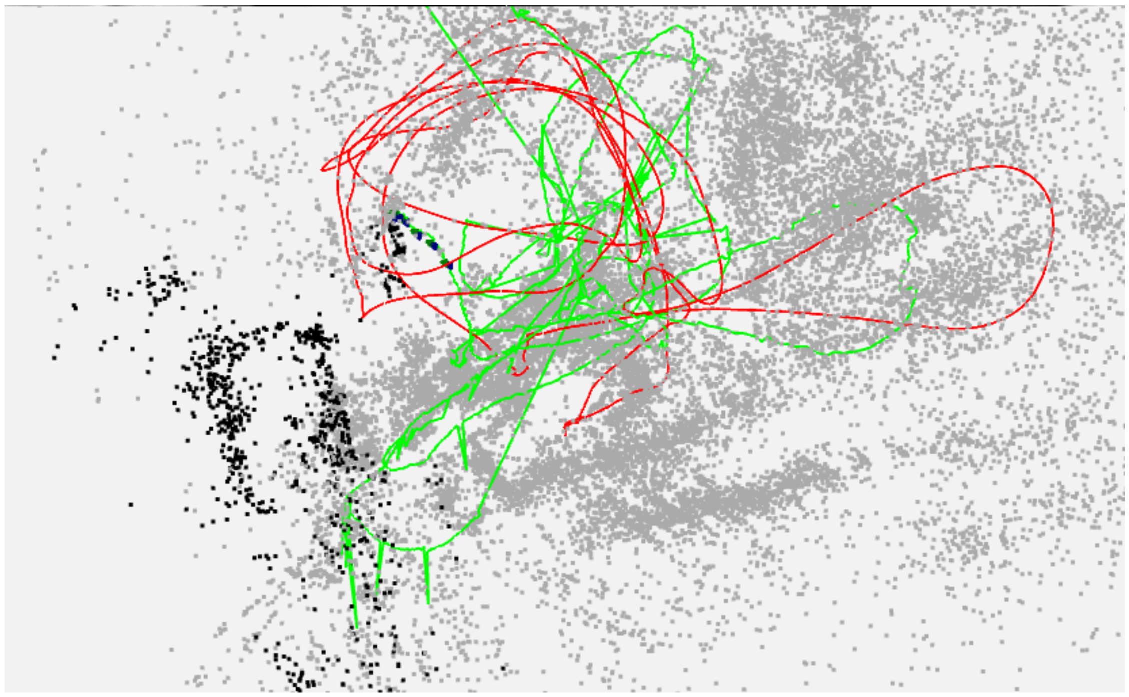

Figure 12: Results on MH_03_medium sequence.

	VO		VIO	
Error	**ATE (RMSE)**	**RPE (Translation Mean)**	**ATE (RMSE)**	**RPE (Translation Mean)**
V1_01_easy	0.715	3.770	**0.186**	**3.301**
V1_02_medium	2.037	**3.356**	**1.943**	3.358
MH_01_easy	0.421	6.088	**0.352**	**4.351**
MH_02_easy	**0.254**	**4.064**	0.326	6.055
MH_03_medium	3.251	**5.350**	**3.240**	5.875

Table 1: Trajectory Accuracy in EuRoC Dataset (Raw Ground-Truth).

5 Conclusions

In conclusion, our experiment involves the implementation of a Visual Inertial Odometry approach that utilizes the complementary data from an IMU. This integration results in a localization method that exhibits significantly less drift compared to traditional VO approaches, where drift can accumulate over time. Our implementation of Visual Inertial Odometry, which fuses the data from both visual and inertial sensors, outperforms traditional VO according to our experiments. This demonstrates the effectiveness of data fusion and highlights the potential benefits of incorporating multiple sensor modalities for improved localization.

5.1 Limitations

Our VIO is more computationally costly because we have more parameters to estimate than VO. So we have to keep window size for bundle adjustment smaller than in VO. Additionally, we have simplified initialization, which we set some variables as constants or as 0. It leads to inaccurate results.

5.2 Future Work

As we have already mentioned our VIO is naive implementation of IMU integration, there is room for improvements. Firstly, more complex scenarios need to be considered while initialization (e.g. MAV is not still at the beginning). In parallel, acceleration and gyroscope biases need to be considered. Secondly, we can challenge our VIO to do benchmarking on different datasets such monocular and RGB-D. In monocular datasets, we need to take into consideration initialization of scale. Lastly, in SLAM systems, one important challenge is loop closure. Our VIO can be improved to deal with loop closure in future.

References

[1] Todd Lupton and Salah Sukkarieh. Visual-inertial-aided navigation for high-dynamic motion in built environments without initial conditions. *IEEE Transactions on Robotics*, 28(1):61–76, 2012.

[2] Raúl Mur-Artal and Juan D Tardós. Visual-inertial monocular slam with map reuse. *IEEE Robotics and Automation Letters*, 2(2):796–803, 2017.

[3] Vladyslav Usenko, Nikolaus Demmel, and Daniel Cremers. The double sphere camera model. In *2018 International Conference on 3D Vision (3DV)*. IEEE, sep 2018.

[4] Christian Forster, Luca Carlone, Frank Dellaert, and Davide Scaramuzza. On-manifold preintegration for real-time visual–inertial odometry. *IEEE Transactions on Robotics*, 33(1):1–21, feb 2017.

[5] TUM Computer Vision Group. Basalt library.

[6] Sameer Agarwal, Keir Mierle, and The Ceres Solver Team. Ceres Solver, 3 2022.

[7] TUM Computer Vision Group. Useful tools for the rgb-d benchmark.

[8] Michael Burri, Janosch Nikolic, Pascal Gohl, Thomas Schneider, Joern Rehder, Sammy Omari, Markus W Achtelik, and Roland Siegwart. The euroc micro aerial vehicle datasets. *The International Journal of Robotics Research*, 2016.