Visual-Inertial Tracking using Preintegrated Factors

Yun-Jin Li & Nils Keunecke

Visual Navigation Practical - Technische Universität München

Abstract

We present our project on visual inertial odometry. In addition, to the visual odometry system which was built in the first half of the practical, our approach integrates IMU data into the existing system. IMUs can provide metric scale to the map of a VO system by computing relative pose updates between successive camera frames. We use IMU preintegratation as proposed by Forster et al [1] to deal with the challenges arising from fusing camera and IMU data. Our results show that the constructed VIO system is more accurate and robust on the majority of the sequences of the EuRoC dataset compared to the VO system implemented in class, without sacrificing the important real-time capability. All code is available on GitHub: `https://github.com/yunjinli/tum_visnav22_project`.

1 Introduction

Simultaneous localization and mapping (SLAM) is the underlying technology of state-of-the-art vision-based navigation algorithms. At the moment, it is usually realized with visual odometry algorithms. Over the years, significant improvements have been made in regard to the accuracy, robustness, and real-time capability. All of these metrics are essential to allow SLAM usage in platforms from mobile robots, to self-driving vehicles, and autonomous UAVs.

Visual Odometry solely relies on cameras to facilitate 6 degrees of freedom (DoF) pose estimation for the camera itself, as well as the estimation of the position of 3D landmarks in space. Cameras have many advantages over other sensor platforms such as LiDAR or radar sensors as they are cheap, widely available and can capture a 3D scene's geometry as well as its texture. However, with a monocular camera we are limited to reconstructing a 3D scene up to an unknown scale. Multiple solutions exist to overcome this issue. Stereo camera setups can be used as features, observed by both cameras, can be triangulated in metric space. However, stereo setups still have multiple drawbacks. They get increasingly inaccurate in their depth estimations if the baseline (ie. the distance between the two cameras) is multiple orders of magnitude smaller than the distance from the camera to the observed landmarks. Additionally, VO systems, even stereo ones, have no way to bridge small segments in a sequence of images in which the visual odometry fails because of a lack of features, motion blur, or quick changes of illumination. These gaps lead to a complete separation from the reconstructed map and the re-initialization with a new map is often the only viable solution. Inertial measurement units (IMU) are a type of sensor that can overcome these issues. Like cameras, they are cheap and ubiquitous - every modern smartphone has them. IMUs provide sensor information that is complementary to cameras. They deliver high frequency acceleration measurements which can be integrated into relative pose updates over time.

This project therefore aims at expanding the visual odometry system developed in the first half of the Vision-based Navigation practical course with IMU data into a functioning visual inertial odometry system. Our implementation should follow the approach proposed by Forster et al. [1] and will make use of reference implementations found in the extensive libraries of the Basalt project [2]. Both papers will be discussed in depth in the next section. The remaining paper is structured as follows: In section 2 we will discuss related literature before we explain our method in depth. In section 4, we present our experimental findings. In the last section, we discuss the impact of our findings and the shortcomings and potential future work for our approach.

2 Related work

SLAM has been a very fast moving subdomain of computer vision research ever since the first successful SLAM papers like MonoSLAM [3]. Nowadays, there are two main branches in SLAM research: On the one hand, we have photometric approaches [4] optimizing pixel intensities, and on the other we have geometric approaches [5] which compute some form of intermediate features. However, the main idea has remained the same: tracking features along a sequence of images and triangulate them in 3D world coordinates, while also computing the pose of the camera. Bundle Adjustment is used for pose refinement. Similar to many other fields of computer vision research, SLAM has seen a shift to deep learning-based methods in the previous years like [6]. This is however not the only direction in which SLAM research is conducted. Other sensor types can be used to improve the quality of the reconstruction. In the early years of real-time SLAM, RGB-D cameras were particularly popular[7]. Other approaches make use of LiDAR sensors [8].

Nonetheless, camera setups are usually preferred for the many reasons as outlined in the previous section. IMUs can support camera setups and many approaches investigate the combination of a visual odometry system (VO) with an IMU into a Visual Inertial Odometry system (VIO) [9] [10]. The addition of the IMU is non-trivial. Forster et al. propose a preintegration theory that performs preintegration on the manifold of the rotation group [1]. They can overcome the trade-offs which previously had to be made when using IMU data in VIO between filtering approaches, which suffer from linearization errors and smoothing approaches which are very slow. The authors show improved performance over previous approaches and are still considered to be among the SOTA.

The data association of the camera with IMU data, the calibration of IMU data, and IMU preintegration are non-trivial. Additionally, the optimization of VIO systems is more complex than for bare visual odometry systems. Basalt [2] is a very-well maintained repository which implements a lot of the SOTA visual SLAM algorithms and comes with a lot of functionalities surrounding IMU preintegration as well. When developing new methods it is appealing as a reference implementation.

3 Method description

3.1 Status Quo

During the first phase of the practical course, we were expected to construct a visual odometry (VO) method which uses image pairs from a stereo camera setup to perform SLAM. Towards this goal, based on a single initial image pair, the VO extracts features from both images using ORB keypoints [11] and matches them between the two images using BRIEF descriptors [12]. Matches are further distilled using the epipolar constraint and RANSAC [13]. A Bag-of-words approach is used to track the features over multiple images.

Now, a map is initialized based on an initial camera pair, and the observed features are triangulated between the two cameras using the matched descriptors. They are added to the map as 3D landmarks (map points). After this initialization, we iteratively add new camera pairs and corresponding landmarks using PnP in a RANSAC scheme. As minor errors occur in all of the previous steps we have to optimize the 6DoF pose of all cameras as well as the 3DoF position of all landmarks using Bundle Adjustment (See the grey section in figure 3).

To make this optimization real-time capable, we do not add all frames to the optimization problem, but only keyframes that are created based on a set of criterions. A sliding optimization window is additionally used, to further decrease the number of parameters in the optimization problem and in order to keep the size relatively constant over time. Old keyframes and their respective observed 3D landmarks are marginalized and will no longer be optimized in the Bundle Adjustment problem.

Even though the visual odometry can already accomplish quite nice results in the simple dataset, used during the exercises, it's worth noting that the capability of the VO is still quite limited. Therefore, the goal of this project is to extend the current VO baseline to a Visual-Inertial Odometry (VIO) by integrating IMU measurements.

3.2 IMU model and motion integration

The IMU gives us acceleration and angular velocity. Note that these terms are affected by white noises and slowly varying biases. Because of the time constraint of the project, we would like to make things as simple as possible. Therefore, we neglect the noises and biases in our VIO system. Based on the previous work [1], we can derive the state at $t + \Delta t$ as follow in equation 1. Note that $\mathbf{R}(t)$, $\mathbf{v}(t)$, and $\mathbf{p}(t)$ are the rotation, velocity, and position estimated at time t and they are all observed by the world coordinate frame. The assumption we make here is that between the interval $[t, t + \Delta t]$, $\mathbf{R}(t)$ is constant. This is the reason why the integration in equation 1 would only be reasonable when Δt is small.

$$\begin{aligned}
\mathbf{R}(t+\Delta t) &= \mathbf{R}(t)\mathrm{Exp}(\boldsymbol{\omega}(t)\Delta t)\\
\mathbf{v}(t+\Delta t) &= \mathbf{v}(t) + \mathbf{g}\Delta t + \mathbf{R}(t)\mathbf{a}(t)\Delta t\\
\mathbf{p}(t+\Delta t) &= \mathbf{p}(t) + \mathbf{v}(t)\Delta t + \frac{1}{2}\mathbf{g}\Delta t^2 + \frac{1}{2}\mathbf{R}(t)\mathbf{a}(t)^2
\end{aligned} \tag{1}$$

3.3 IMU preintegration

As we already mentioned, IMU measurements have a high frequency, while the frames usually don't. Therefore, in order to handle the high-frequency IMU data, we preintegrate several consecutive IMU measurements between frames and compute the overall delta state, this delta state will later be used to compute the residual, that is the so-called preintegrated factor as described in equation 2. We basically use the framework developed in basalt [2] but neglect all the bias terms to compute the delta state and calculate the preintegrated factor which is expected to be minimized. The illustration of IMU preintegration between frames can be seen in figure 2, where we compute the preintegrated factor with all the IMU measurements between consecutive frames. This preintegrated factor would later serve as a constraint when we perform the optimization.

$$\begin{aligned}
\mathbf{r}_{\Delta\mathbf{R}} &= \mathrm{Log}(\Delta\mathbf{R}\mathbf{R}_j^T\mathbf{R}_i)\\
\mathbf{r}_{\Delta\mathbf{v}} &= \mathbf{R}_i^T(\mathbf{v}_j - \mathbf{v}_i - \mathbf{g}\Delta t) - \Delta\mathbf{v}\\
\mathbf{r}_{\Delta\mathbf{p}} &= \mathbf{R}_i^T(\mathbf{p}_j - \mathbf{p}_i - \frac{1}{2}\mathbf{g}\Delta t^2) - \Delta\mathbf{p}
\end{aligned} \tag{2}$$

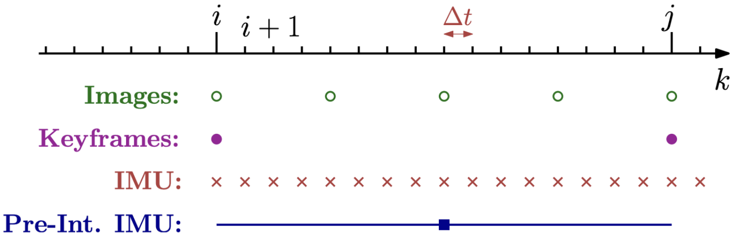

Figure 1: Different rates for IMU and camera. (Refer to [1])

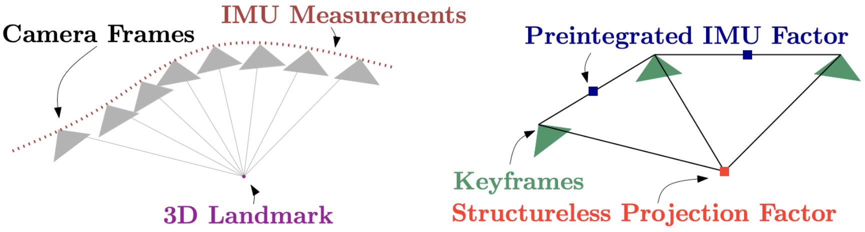

Figure 2: (Left) Visual and inertial measurements in VIO. (Right) Factor graph in which several IMU measurements are summarized in a single preintegrated IMU factor and a structureless vision factor constraints keyframes observing the same landmark. (Refer to [1])

3.4 Optimization

Based on the residual derived from the previous section, we can perform the VIO bundle adjustment with IMU preintegration. Here, we propose our own factor graph which is used during optimization and is shown in figure 3. Apart from simply performing normal BA on 10 keyframes to minimize the reprojection errors, we also actively optimize the residuals of the IMU preintegration and also the reprojection errors over the latest 3 frames. The preintegration of the IMU could serve as a constraint during the optimization which makes it more robust.

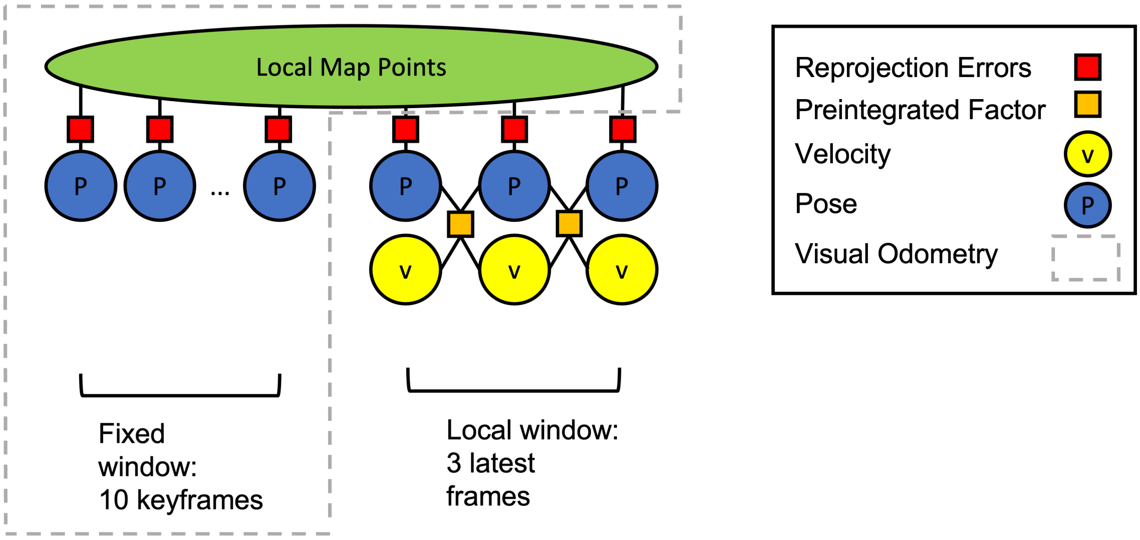

Figure 3: The factor graph of our VIO system. The area in the grey box is the original VO framework. Note that the red and orange boxes are the terms that we try to minimize while optimizing all the green, blue, and yellow bubbles.

4 Experiments and results

In the following section, the previously presented approach is evaluated on the EuRoC MAV dataset [14], which is an indoor dataset collected on a small drone. We evaluate our visual inertial odometry approach and use the visual odometry implementation from the tutorials as the baseline. While the EuRoC MAV dataset contains 11 sequences due to time constraints we only evaluated our approach on the following sequences: *Machine Hall 01*, *Machine Hall 02*, *Machine Hall 03*, *Machine Hall 04*, and *Machine Hall 05*. As our VIO pipeline current cannot recover from a complete loss of 3D landmarks and the non-deterministic behavior of our 2nd order optimization scheme as well as the map initialization, we repeat each sequence for 50 runs. In Table 2, 3, 4, 5, and 6, we display the quantitative results for our approach for each of the five sequences. The authors of the dataset rate them as easy, easy, medium, difficult, and difficult respectively.

4.1 Error Threshold

Based on our observation, the reasonable range of Absolute Trajectory Error (ATE) is below 3 meters, where we can still observe some similar parts of the estimated trajectory that seem to match the trajectory of the ground true. ATE that's above this value would look something similar to both of the estimated trajectories in figure 6, where we can hardly see any alignment between the estimation and the ground truth. Therefore, in order to compute reasonable statistics to compare the performance of both methods, we have to set an ATE threshold to filter out some outlier cases and mark them as failures that are either caused by poor optimization or bad localization of the camera. However, there exist some hard sequences as we mentioned before, where both methods appear to perform poorly. If we simply set an error threshold to 3 meters for all the datasets, we wouldn't be able to get enough successful cases in the 50 runs. Hence, for harder datasets such as *Machine Hall 03*, *Machine Hall 04*, and *Machine Hall 05*, the error thresholds are set, such that the VIO can achieve half of the successful cases over the 50 runs. The resulting *Error Threshold Table* can be seen in table 1.

	ATE Threshold (m)
MH01 (easy)	3.0
MH02 (easy)	3.0
MH03 (medium)	6.9
MH04 (difficult)	4.3
MH05 (difficult)	6.8

Table 1: Absolute trajectory error threshold for 5 datasets, if the absolute trajectory error computed is above the threshold, we consider this run as a failure case.

4.2 Quantitative Evaluation

We use a common metric for the evaluation of the visual SLAM system and compare the root-mean-squared error (RMSE) of the absolute trajectory error (ATE) [15] in meters between the trajectory computed by the VO/VIO and the ground truth trajectory provided by the authors. Afterward computing the trajectory but before computing the ATE, we perform singular value decomposition (SVD) alignment of the two trajectories.

For easy sequences, we set the ATE thresholds to 3 meters. According to table 2, we can see that both methods perform quite nicely and VIO achieves a lower average ATE and failure rate. For *Machine Hall 02*, it's worth noting that the failure rate of VIO is 4 times lower than that of VO. From this sequence, we can already observe the performance boost with the help of IMU preintegration.

For hard sequences, we set the ATE threshold according to the performance of the VIO (See table 1). From table 4, 5, 6, we can see different extents of improvements from the VIO.

Overall, based on the statistics, we can conclude that the VIO is capable of increasing the robustness and reducing the ATE in most of the sequences.

Machine Hall 01	avg. ATE	min	max	#failures	failure rate
VO (baseline)	1.069	0.236	2.822	5	10%
VIO (ours)	**0.831**	**0.094**	**2.793**	**3**	**6%**

Table 2: Quantitative Evaluation for the Machine Hall 01 sequence. We calculate the average, minimum, and maximum absolute trajectory error over the 50 runs. Note that the failure is determined by the error threshold defined in 1 according to different difficulty levels for each dataset.

Machine Hall 02	avg. ATE	min	max	#failures	failure rate
VO (baseline)	1.438	**0.272**	**2.922**	12	24%
VIO (ours)	**1.144**	0.326	2.996	**3**	**6%**

Table 3: Quantitative evaluation for the Machine Hall 02 sequence. We use the same metrics as stated in Table 2.

Machine Hall 03	avg. ATE	min	max	#failures	failure rate
VO (baseline)	**5.398**	**3.220**	**6.732**	31	62%
VIO (ours)	5.491	3.979	6.842	**25**	**50%**

Table 4: Quantitative evaluation for the Machine Hall 03 sequence. We use the same metrics as stated in Table 2.

Machine Hall 04	avg. ATE	min	max	#failures	failure rate
VO (baseline)	2.928	2.034	**4.180**	39	78%
VIO (ours)	**2.763**	**1.321**	4.183	**25**	**50%**

Table 5: Quantitative evaluation for the Machine Hall 04 sequence. We use the same metrics as stated in Table 2.

Machine Hall 05	avg. ATE	min	max	#failures	failure rate
VO (baseline)	5.202	3.082	**6.499**	37	74%
VIO (ours)	**4.465**	**1.450**	6.773	**25**	**50%**

Table 6: Quantitative evaluation for the Machine Hall 05 sequence. We use the same metrics as stated in Table 2.

4.3 Qualitative Evaluation

In addition to the quantitative evaluation we used the GUI provided during the tutorial to perform a qualitative evaluation as well. As expected, it can be observed that the computed trajectory and the ground truth trajectory align better when the VIO is used. Additionally, during the previous section, we define the cases that have errors greater than the threshold as failure cases. We investigate those cases and try to find the possible reasons that lead to failure. It turns out that there are some frames that are extremely badly localized and consequentially have a big difference in position and orientation from the previous frames. This usually happens when the camera moves at a higher velocity or the scene is completely dark. Although VIO also fails in such scenarios sometimes, the experiments show that it's more robust compared to VO in such scenarios overall. This could be observed especially in *Machine Hall 04* and *Machine Hall 05* sequence (see figure 7), where the drone flies in a dark environment for a period of time at high speed. This kind of scenario seems to be quite tricky for VO as it usually localizes the camera badly once entering the dark scene. However, unlike VO, VIO seems to be able to handle this kind of environment well, as it usually wouldn't badly localize the camera with the help of IMU. In order to have a clear picture of the scenarios described above, we record and compare the performance for both VO and VIO in MH04 `https://youtu.be/aNgcuXywrX4` and MH05 `https://youtu.be/fA9gDHygKfg`

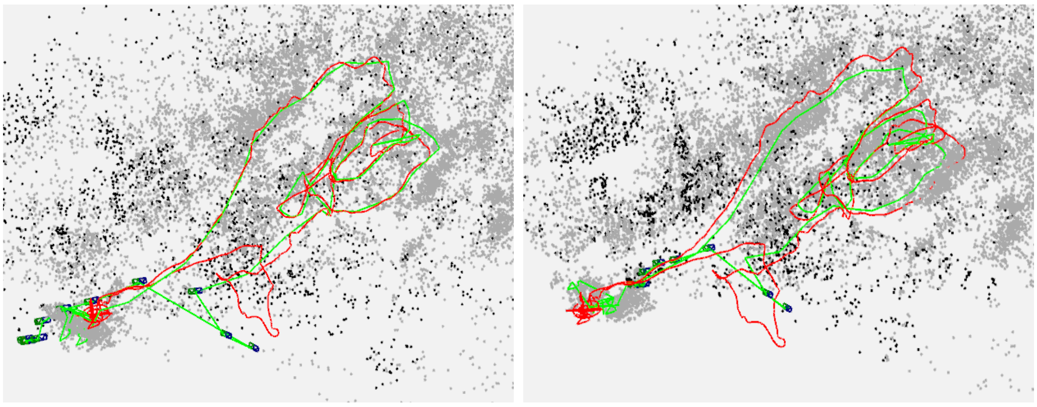

Figure 4: Qualitative results on Machine Hall 01 sequence. The green trajectories are the estimated trajectories from our VIO (left) and baseline VO (right), the red trajectories are ground true, and the grey and black points are landmarks.

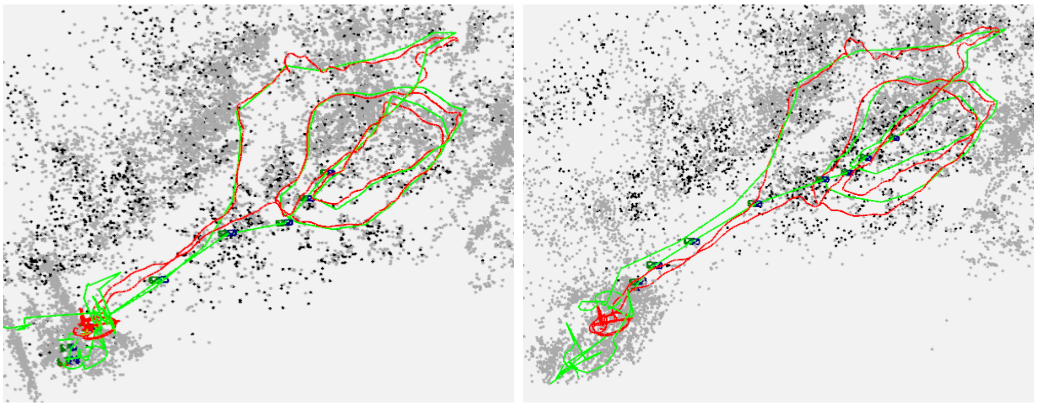

Figure 5: Qualitative results on Machine Hall 02 sequence. Same configurations as stated in figure 4

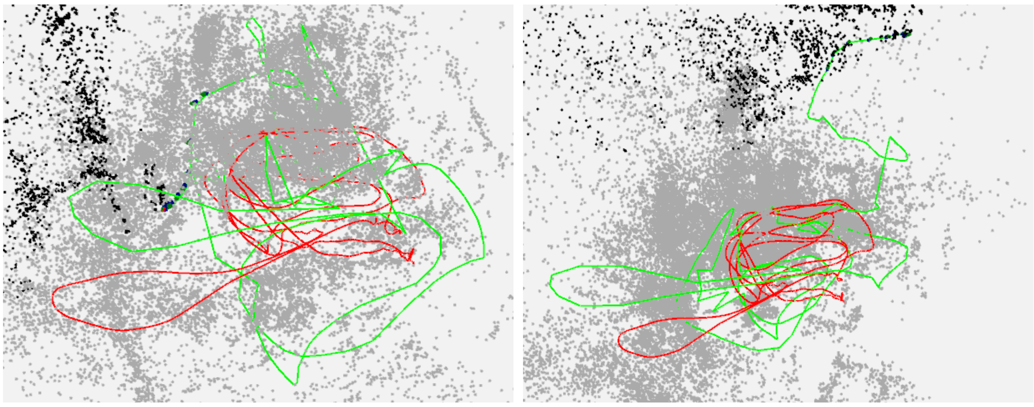

Figure 6: Qualitative results on Machine Hall 03 sequence. Same configurations as stated in figure 4

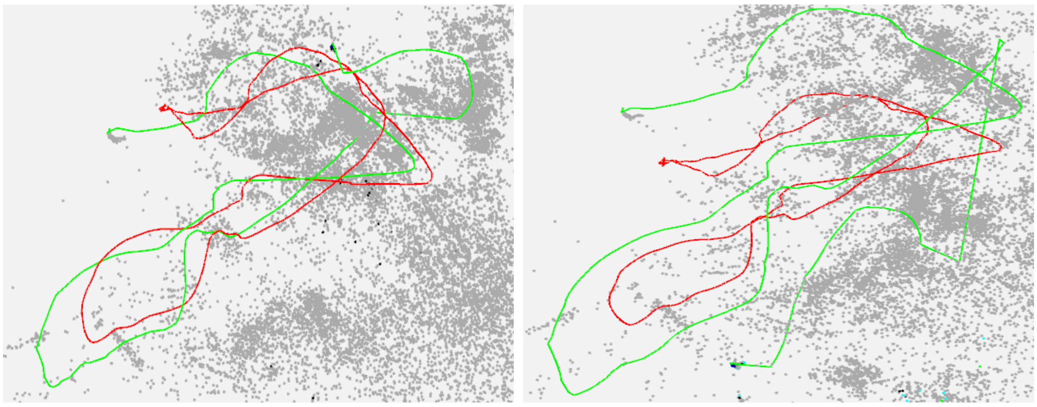

Figure 7: Qualitative results on Machine Hall 04 sequence. Same configurations as stated in figure 4

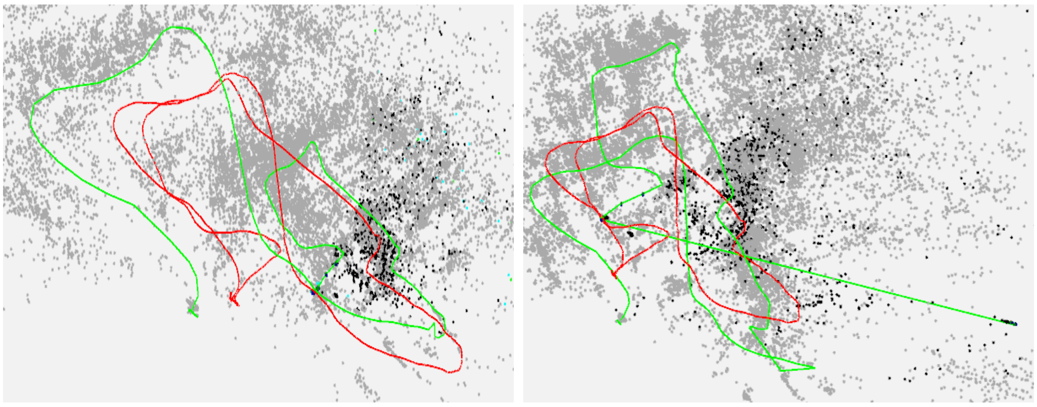

Figure 8: Qualitative results on Machine Hall 05 sequence. Same configurations as stated in figure 4

As can be observed, the general geometry of the calculated trajectory is correct for each of the five sequences, except for sequence 3 (see figure 6), where both methods perform badly overall. To sum up, the quality of the alignment varies a lot between the 5 sequences due to the different difficulty of each sequence, but one thing we can be sure of is that VIO does have an overall better performance than VO.

5 Discussion & Conclusion

In this project, we have convincingly shown the advantages of extending a visual odometry system into a visual inertial odometry system. The implemented approach does outperform the baseline implementation during the first phase of the practical course in most sequences. Additionally, we have shown that IMU preintegration can handle the challenges that arise when fusing camera with IMU data in a meaningful way. Adding the pose as well as the velocity of recent frames to the optimization problem which previously only contained keyframes, does provide us with the expected reduction in absolute trajectory error (ATE).

Overall, we can consider this project to be successful as it delivers all the expected results and is inline with a general trend in the community towards visual inertial SLAM systems.

6 Future Work

While the results presented in this project are impressive and compare well against the baseline implementation, it should be noted that the results are far away from the performance of state-of-the-art visual inertial SLAM papers. The BASALT repository [2] already implemented a lot of additional features. A good overview of the current SOTA can be found in Beghdadi et al. [16].

Nonetheless, we want to briefly outline potential extensions of our own implementation: As outlined in the introduction an IMU can help the VO to reinitialize itself in the same map after a complete loss of all active 3D landmarks. In the current state of our implementation, such a loss will lead to a failure of the pipeline. Additionally, the current optimization scheme only considers the 6DoF pose and velocity of the camera but ignores the bias of the IMU. It is expected that the VIO system can be significantly improved by considering the bias of the gyroscope and the accelerometer. The implementation also lacks some typical elements of SLAM systems such as the ability to detect and close loops in the map.

References

[1] Christian Forster, Luca Carlone, Frank Dellaert, and Davide Scaramuzza. On-manifold preintegration theory for fast and accurate visual-inertial navigation. *CoRR*, abs/1512.02363, 2015.

[2] Vladyslav Usenko, Nikolaus Demmel, David Schubert, Jörg Stückler, and Daniel Cremers. Visual-inertial mapping with non-linear factor recovery. *IEEE Robotics and Automation Letters*, 5(2):422–429, 2019.

[3] Andrew J Davison, Ian D Reid, Nicholas D Molton, and Olivier Stasse. Monoslam: Real-time single camera slam. *IEEE transactions on pattern analysis and machine intelligence*, 29(6):1052–1067, 2007.

[4] Jakob Engel, Vladlen Koltun, and Daniel Cremers. Direct sparse odometry. *IEEE transactions on pattern analysis and machine intelligence*, 40(3):611–625, 2017.

[5] Raul Mur-Artal, Jose Maria Martinez Montiel, and Juan D Tardos. Orb-slam: a versatile and accurate monocular slam system. *IEEE transactions on robotics*, 31(5):1147–1163, 2015.

[6] Zachary Teed and Jia Deng. Droid-slam: Deep visual slam for monocular, stereo, and rgb-d cameras. *Advances in neural information processing systems*, 34:16558–16569, 2021.

[7] Gibson Hu, Shoudong Huang, Liang Zhao, Alen Alempijevic, and Gamini Dissanayake. A robust rgb-d slam algorithm. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1714–1719. IEEE, 2012.

[8] Wolfgang Hess, Damon Kohler, Holger Rapp, and Daniel Andor. Real-time loop closure in 2d lidar slam. In *2016 IEEE international conference on robotics and automation (ICRA)*, pages 1271–1278. IEEE, 2016.

[9] Gabriel Nützi, Stephan Weiss, Davide Scaramuzza, and Roland Siegwart. Fusion of imu and vision for absolute scale estimation in monocular slam. *Journal of intelligent & robotic systems*, 61(1-4):287–299, 2011.

[10] Stefan Leutenegger, Simon Lynen, Michael Bosse, Roland Siegwart, and Paul Furgale. Keyframe-based visual–inertial odometry using nonlinear optimization. *The International Journal of Robotics Research*, 34(3):314–334, 2015.

[11] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. Orb: An efficient alternative to sift or surf. In *2011 International conference on computer vision*, pages 2564–2571. Ieee, 2011.

[12] Michael Calonder, Vincent Lepetit, Mustafa Ozuysal, Tomasz Trzcinski, Christoph Strecha, and Pascal Fua. Brief: Computing a local binary descriptor very fast. *IEEE transactions on pattern analysis and machine intelligence*, 34(7):1281–1298, 2011.

[13] Martin A Fischler and Robert C Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[14] 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.

[15] David Prokhorov, Dmitry Zhukov, Olga Barinova, Konushin Anton, and Anna Vorontsova. Measuring robustness of visual slam. In *2019 16th International Conference on Machine Vision Applications (MVA)*, pages 1–6. IEEE, 2019.

[16] Ayman Beghdadi and Malik Mallem. A comprehensive overview of dynamic visual slam and deep learning: concepts, methods and challenges. *Machine Vision and Applications*, 33(4):54, 2022.