Studying the Impact of Inertial Constraints in Two-View Reconstruction for Visual and Visual-Inertial Odometry

Andre Schreiber[1], Hameed Abdul-Rashid[2], and Jongwon Lee[3]

***Abstract*— In this project, we aim to integrate data from an inertial measurement unit (IMU) into a two-view reconstruction pipeline using Python. We utilize libraries like SymForce and OpenCV for functionality like optimization and feature point detection. However, we do not rely on existing libraries for two-view reconstruction, inertial preintegration, or visual-inertial optimization, opting instead to develop these components ourselves to further our understanding of two-view reconstruction and the integration of inertial data into a vision system. We analyze the effect of the inertial data on various quantities of interest–such as reprojection error–for visual odometry (VO) and visual-inertial odometry (VIO) over frame sequences from two publicly available VO/VIO datasets. We find that incorporation of the inertial data can lead to worse reprojection error as compared to visual-only optimization, as the addition of inertial factors forces the optimizer to consider both reprojection error and consistency with IMU measurements in producing its solution. However, incorporating inertial data in VIO resolves the scale ambiguity inherent in VO, enabling estimates of positional changes between frames at scale — a capability not achievable with visual-only solutions. Our code (and links to the data) can be found at: https://github.com/andreschreiber/AE598Project.**

I. Introduction

Odometry is the process of estimating the pose of an object over time using data from sensors like IMUs. However, purely inertial-based approaches to odometry show limitations, as challenges like drift and sensor noise can lead to the build-up of error over time [8]. While the effects of these issues can be remedied (although not eliminated) through the use of high-end sensors, such equipment can be prohibitively expensive in many applications. Visual odometry (VO) provides an alternative approach to inertial odometry and involves estimating the pose of a camera over time by processing a sequence of images captured from the camera. However, VO is not without its own challenges, as noisy, high-dimensional visual data needs to be processed in order to compute relative pose, which can involve significant computation as compared with purely inertial odometry. In addition, VO using a monocular camera suffers from a scale ambiguity, as pose changes can only be recovered up to scale (unlike inertial odometry, which can resolve the metric scale).

[1]Andre Schreiber is with the Department of Electrical Engineering, University of Illinois, Urbana-Champaign, IL 61801, USA (andrems2@illinois.edu)

[2]Hameed Abdul-Rashid is with the Department of Computer Science, University of Illinois, Urbana-Champaign, IL 61801, USA (hameeda2@illinois.edu)

[3]Jongwon Lee is with the Department of Aerospace Engineering, University of Illinois, Urbana-Champaign, IL 61801, USA (jongwon5@illinois.edu)

Visual-inertial odometry (VIO) involves fusing measurements from an inertial device (e.g., an IMU) and a camera to provide enhanced performance. By integrating both visual and inertial data, VIO removes the scale ambiguity seen in visual odometry, while also granting enhanced global information (captured by cameras) to produce reduced drift compared to purely inertial odometry [8]. Thus, VIO could enable enhancements over both inertial and visual odometry systems.

We seek to integrate inertial data into a two-view reconstruction pipeline (which is an important step in the visual odometry process) to gain an understanding of how inertial data is integrated into vision systems. To ensure a sufficient understanding of the challenges involved, we plan to implement each component from scratch in Python, only using libraries like OpenCV for feature point detection and SymForce for non-linear optimization. We utilize a two-view reconstruction implementation that we implement from scratch to perform visual odometry between two camera images (estimating the relative pose and 3D locations of key points up to scale). Then, we implement IMU preintegration from scratch to integrate the IMU measurements and provide the information needed for IMU optimization factors. Finally, we incorporate these integrated IMU measurements into our optimization formulation using a factor-graph approach.

We analyze the effects of integrating the IMU measurements into our two-view reconstruction in terms of reprojection error, computation time, and relative change in pose on frame sequences from the KITTI [7] dataset and EuRoC MAV dataset [1]. We find that the addition of the IMU data into the non-linear optimization can lead to worse reprojection error and fewer inliers as compared to visual-only optimization, as visual-inertial optimization involves additional inertial factors (which use potentially noisy data from an IMU) that the optimizer seeks to optimize in addition to reprojection error. However, we see that the incorporation of inertial data resolves the scale ambiguity seen in the solution produced by only using visual information. Finally, we find that visual-inertial optimization takes significantly more computation time than visual-only optimization by involving more optimization terms and requiring more optimization steps for convergence.

II. Methods

This section expands on each aspect of our proposed project, including the implementation of VO/VIO, the dataset that we use, and the method by which we evaluate our approach.

A. Two-View Reconstruction

Two-view reconstruction is a fundamental component of tasks like structure from motion (SfM) and visual odometry (VO). The two-view reconstruction process involves detecting feature points shared between two images, and using these feature points to estimate the relative poses between the two cameras which captured the images. In addition to estimating camera poses, the 3D coordinate of each feature point is calculated during the two-view reconstruction process. In other words, the problem aims to estimate the poses of two frames i and j :

$$\mathbf{R}_i^W, \mathbf{p}_i^W, \mathbf{R}_j^W, \mathbf{p}_j^W,$$

where the positions $\mathbf{p}_i^W$ and $\mathbf{p}_j^W$ are determined up to scale, and the 3D locations of feature points observed in both views:

$$\mathbf{p}_l^W, \text{ for } l \in \{1, \ldots, L\},$$

which are also up to scale. For this project, we refactor and adapt our two-view reconstruction and SfM code from the lecture to work with images from visual odometry datasets [1], [7].

The implementation of two-view reconstruction (shown in Fig. 1) can be summarized into three key steps: (1) feature detection and matching, (2) computing an analytical solution, and (3) refinement of the analytical solution using non-linear optimization. The first step involves identifying feature points in the two images, which is performed using the SIFT feature point detector [9]. SIFT detects feature points and provides the locations of these points as well as a descriptor that describes a local region around each feature point. Matches between the feature points are found by comparing the descriptors of each feature point in the images and applying a ratio test (to exclude ambiguous matches). Once feature points have been detected and matched, they can be used to obtain an analytical solution for the relative poses of the cameras (using epipolar geometry) and for the 3D locations of the feature points (using triangulation). This analytical solution is sensitive to outliers, so to improve robustness to outliers we utilize RANSAC [4] for our two-view reconstruction. Once the analytical solution is found, it can be used as an initial guess for a non-linear optimization based on reprojection error, where the cost function is defined as follows:

$$\sum_{l=1}^{L} \left(\|\mathbf{r}_{\mathcal{C}_{il}}\|_{\mathbf{\Sigma}_\mathcal{C}}^2 + \|\mathbf{r}_{\mathcal{C}_{jl}}\|_{\mathbf{\Sigma}_\mathcal{C}}^2 \right) + \lambda(\|\Delta\mathbf{p}_{ij}\| - 1)^2,$$

where $\mathbf{r}_{\mathcal{C}_{il}}$ and $\mathbf{r}_{\mathcal{C}_{jl}}$ represent the reprojection errors of the 3D feature point $\mathbf{p}_l$ where $l \in \{1, \ldots, L\}$ for each view i, j and $\mathbf{\Sigma}_\mathcal{C}$ is the modeled covariance associated with these errors, which we set as a 2×2 identity matrix. Additionally, during the optimization, the norm of the change in position between frames, $\Delta\mathbf{p}_{ij}$, is constrained to be one using a regularization parameter λ, which results in a solution where the distance between camera poses is approximately 1.

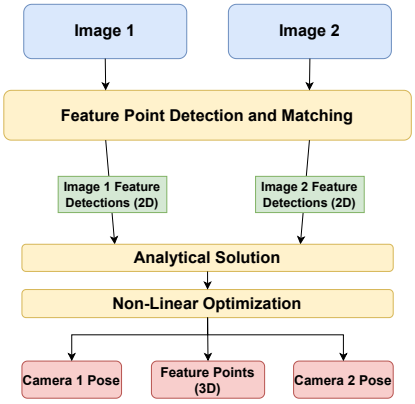

Fig. 1. Overview of the two-view reconstruction process.

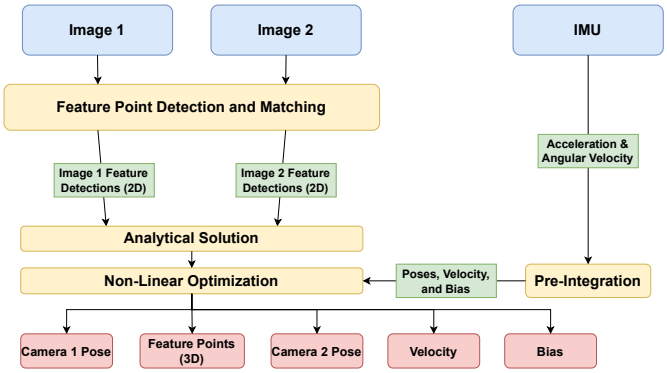

Fig. 2. Modified two-view reconstruction with inclusion of IMU measurements.

B. Two-View Reconstruction with Inertial Data

VIO combines camera and IMU data for state estimation, offering several advantages. One advantage is its usefulness in establishing feature point correspondences over short time spans, particularly under challenging conditions when vision-only correspondence matching may struggle (e.g., excessive motion blur, dim lighting, or areas that lack distinctive visual features). Another advantage is its ability to determine scales for camera and map poses by adding constraints from inertial data between frames. Our project focuses on this second advantage.

Most modern VIO algorithms incorporate two main modules: "camera-IMU alignment" and "visual-inertial optimization." Camera-IMU alignment involves establishing the IMU's orientation relative to gravity and its biases — which are crucial for successful IMU preintegration — as well as scaling the 3D feature point map and trajectory from visual-only multi-view reconstruction [2], [5]. In our work, we assume that the initial orientation of the IMU in the world frame and its biases are already established and provided using open-source data. Furthermore, we set up our visual-inertial optimization to meet the secondary objective by utilizing a regularization term, the details of which will be discussed in the following sections.

1) Visual-inertial optimization: Visual-inertial optimization is built upon the previously mentioned two-view reconstruction problem. The problem now aims to optimize the poses of two frames i and j :

$$\mathbf{R}_i^W, \mathbf{p}_i^W, \mathbf{R}_j^W, \mathbf{p}_j^W$$

where the positions $\mathbf{p}_i^W$ and $\mathbf{p}_j^W$ are now going to be determined at scale, and the 3D locations of feature points observed in both views:

$$\mathbf{p}_l^W, \text{ for } l \in \{1, \ldots, L\}.$$

Additionally, IMU data introduces new variables to the optimization framework of the two-view reconstruction: the velocity $\mathbf{v}_i^W$, and biases for the accelerometer and gyroscope of the IMU ($\mathbf{b}_i^a$ and $\mathbf{b}_i^g$, respectively). For simplicity, we will omit the superscript W denoting the "world frame" in subsequent discussions.

This integration notably includes the "IMU preintegration" process [5], which computes changes in pose between frames by aggregating measurements from the IMU's accelerometer and gyroscope (ranging from tens to hundreds of measurements) into a single factor.

The change in pose between frames i and j, including orientation, position, and velocity, *without compensating gravitational acceleration*, is derived by preintegrating the accelerometer and gyroscope readings $\tilde{\mathbf{a}}_k$ and $\tilde{\boldsymbol{\omega}}_k$ for $k \in i, \ldots, j-1$. This process integrates the IMU measurements to estimate the cumulative effect on pose from one frame to the next:

$$\begin{aligned}
\Delta\tilde{\mathbf{R}}_{ij} &= \prod_{k=i}^{j-1} \operatorname{Exp}\left(\left(\tilde{\boldsymbol{\omega}}_k - \mathbf{b}_i^g\right) \Delta t\right) \\
\Delta\tilde{\mathbf{v}}_{ij} &= \sum_{k=i}^{j-1} \Delta\tilde{\mathbf{R}}_{ik} \left(\tilde{\mathbf{a}}_k - \mathbf{b}_i^a\right) \Delta t \\
\Delta\tilde{\mathbf{p}}_{ij} &= \sum_{k=i}^{j-1} \frac{3}{2} \Delta\tilde{\mathbf{R}}_{ik} \left(\tilde{\mathbf{a}}_k - \mathbf{b}_i^a\right) \Delta t^2,
\end{aligned}$$

where Δt represents the time intervals between consecutive inertial measurements. The residual for the preintegrated IMU measurements is described as: $\mathbf{r}_{\mathcal{I}_{ij}} \doteq \left[\mathbf{r}_{\Delta\mathbf{R}_{ij}}^\top, \mathbf{r}_{\Delta\mathbf{v}_{ij}}^\top, \mathbf{r}_{\Delta\mathbf{p}_{ij}}^\top\right]^\top \in \mathbb{R}^9$, where

$$\begin{aligned}
\mathbf{r}_{\Delta\mathbf{R}_{ij}} &\doteq \log\left(\left(\Delta\tilde{\mathbf{R}}_{ij}\right)^\top \mathbf{R}_i^\top \mathbf{R}_j\right) \\
\mathbf{r}_{\Delta\mathbf{v}_{ij}} &\doteq \mathbf{R}_i^\top \left(\mathbf{v}_j - \mathbf{v}_i - \mathbf{g}\Delta t_{ij}\right) - \Delta\tilde{\mathbf{v}}_{ij} \\
\mathbf{r}_{\Delta\mathbf{p}_{ij}} &\doteq \mathbf{R}_i^\top \left(\mathbf{p}_j - \mathbf{p}_i - \mathbf{v}_i\Delta t_{ij} - \frac{1}{2}\mathbf{g}\Delta t_{ij}^2\right) - \Delta\tilde{\mathbf{p}}_{ij},
\end{aligned}$$

with Δt_{ij} for time interval between frames and $\mathbf{g}$ for gravitational acceleration. The associated covariance $\boldsymbol{\Sigma}_{ij}$ characterizes the *preintegration noise vector* $\boldsymbol{\eta}_{ij}^\Delta$, where

$$\boldsymbol{\eta}_{ij}^\Delta \doteq \left[\delta\boldsymbol{\phi}_{ij}^\top, \delta\mathbf{v}_{ij}^\top, \delta\mathbf{p}_{ij}^\top\right]^\top \sim \mathcal{N}\left(\mathbf{0}_{9\times 1}, \boldsymbol{\Sigma}_{ij}\right).$$

For the detailed derivations of $\boldsymbol{\eta}_{ij}^\Delta$ and $\boldsymbol{\Sigma}_{ij}$, the readers are advised to refer to the supplementary material [6]. Hence, the residual contributes to the cost function as the term $\left\|\mathbf{r}_{\mathcal{I}_{ij}}\right\|_{\boldsymbol{\Sigma}_{ij}}^2$.

Biases, typically modeled as slowly varying over time, require accounting for their change between frames i and j. This is achieved using the following terms:

$$\left\|\mathbf{b}_j^g - \mathbf{b}_i^g\right\|_{\boldsymbol{\Sigma}^{bgd}}^2 + \left\|\mathbf{b}_j^a - \mathbf{b}_i^a\right\|_{\boldsymbol{\Sigma}^{bad}}^2.$$

The covariances are defined as $\boldsymbol{\Sigma}^{bgd} \doteq \Delta t_{ij} \operatorname{Cov}(\boldsymbol{\eta}^{bg})$ and $\boldsymbol{\Sigma}^{bad} \doteq \Delta t_{ij} \operatorname{Cov}(\boldsymbol{\eta}^{ba})$, respectively, where $\operatorname{Cov}(\boldsymbol{\eta}^{bg})$ and $\operatorname{Cov}(\boldsymbol{\eta}^{ba})$ describe the bias evolution over time, modeled as Brownian motion and are assumed to be predetermined.

As a result, the cost function for the two-view reconstruction framework with inertial data integration is defined as follows:

$$\begin{aligned}
&\sum_{l=1}^{L} \left(\left\|\mathbf{r}_{\mathcal{C}_{il}}\right\|_{\boldsymbol{\Sigma}_\mathcal{C}}^2 + \left\|\mathbf{r}_{\mathcal{C}_{jl}}\right\|_{\boldsymbol{\Sigma}_\mathcal{C}}^2\right) \\
&\quad + \left\|\mathbf{r}_{\mathcal{I}_{ij}}\right\|_{\boldsymbol{\Sigma}_{ij}}^2 + \left\|\mathbf{b}_j^g - \mathbf{b}_i^g\right\|_{\boldsymbol{\Sigma}^{bgd}}^2 + \left\|\mathbf{b}_j^a - \mathbf{b}_i^a\right\|_{\boldsymbol{\Sigma}^{bad}}^2 \\
&\quad + \lambda\left(\|\mathbf{p}_j - \mathbf{p}_i\| - \left\|\mathbf{R}_i\Delta\tilde{\mathbf{p}}_{ij} + \mathbf{v}_i\Delta t_{ij} + \frac{1}{2}\mathbf{g}\Delta t_{ij}^2\right\|\right)^2,
\end{aligned}$$

where $\mathbf{r}_{\mathcal{C}_{il}}$ and $\mathbf{r}_{\mathcal{C}_{jl}}$ represent the projection errors of the 3D feature point $\mathbf{p}^l$ where $l \in \{1, \ldots, L\}$ for each view i, j and $\boldsymbol{\Sigma}_\mathcal{C}$ is the modeled covariance associated with these errors, which is set as a 2×2 identity matrix. Additionally, during the optimization, the norm of the change in position between frames expressed in the world frame, $\mathbf{p}_j - \mathbf{p}_i$, is constrained to match the displacement derived from preintegrated data — $\mathbf{R}_i\Delta\tilde{\mathbf{p}}_{ij} + \mathbf{v}_i\Delta t_{ij} + \frac{1}{2}\mathbf{g}\Delta t_{ij}^2$ — using a regularization parameter λ. This approach, not typical in standard visual-inertial optimization formulations, ensures that the scale aligns with the inertial data. We introduce this term to enforce scale alignment directly within the optimization process, eliminating the need for pre-scaling as a preprocessing step.

C. Dataset and Evaluation

Several potential datasets exist for evaluating a VO/VIO system. For example, the KITTI [7] dataset is commonly used for benchmarking VO/VIO approaches and features image sequences collected from a car navigating throughout a city in Germany. Each sequence in KITTI features IMU measurements and high-accuracy GPS readings from a combined GPS/IMU system, as well as camera images captured by several cameras mounted on the vehicle. Ground-truth poses are provided in the dataset with errors of less than 10 cm [11]. The EuRoC MAV dataset [1] features several sequences captured from indoor scenes by a drone, where each sequence has high-accuracy (1 mm) ground-truth pose measurements, as well as IMU readings and camera images from a stereo camera. The TUM VI dataset [11] features several sequences from a hand-held rig with high-accuracy ground-truth poses (1 mm) across both indoor and outdoor settings.

For our experiments, we use the KITTI dataset [7] and EuRoC MAV dataset [1] as they represent interesting scenarios

(outdoor driving and drone flight, respectively) and contain the necessary data for evaluating a visual-inertial odometry system (namely, camera images, IMU readings, and high-accuracy ground-truth poses).

The data collection platform for KITTI [7] is shown in Fig 3. The authors of the dataset provide raw data for each run, as well as post-processed data that synchronizes that data from each sensor and performs additional post-processing (e.g., undistorting the images from the cameras). As we are focused on monocular visual odometry and KITTI features data from several cameras, we opt to use the camera denoted cam0 as the camera for our experiments. The camera provides images with a resolution of 1242×375 and has been calibrated by the authors of the dataset. We use the provided calibration settings for our two-view reconstruction pipeline. The calibration settings specify f_x, f_y, c_x, and c_y, and we use the post-processed data which feature undistorted images (so we do not need to explicitly model camera distortion when using KITTI). In addition, the camera-IMU extrinsic parameters (specifying the pose of the camera relative to the IMU) are crucial measurements for our approach, and we use the values for such parameters that are provided by the authors of KITTI.

We also provide experiments on the EuRoC MAV dataset [1], which uses the data collection setup shown in Fig. 4. Although images from two cameras are provided in the EuRoC MAV dataset, we only use the images from cam0 (which have a resolution of 752×480 pixels). In addition, like with KITTI, calibration settings for intrinsic and extrinsic parameters are provided by the authors of the dataset, and we use the provided values. However–unlike in KITTI–the camera images in EuRoC MAV are not processed to remove distortion and the images show significant distortion effects. To account for this, we undistort feature point locations using a standard distortion model [3] before passing them through our two-view reconstruction pipeline (we use the distortion coefficients provided by the authors of EuRoC MAV for undistorting the points).

We analyze our method using reprojection error on the analytical solution, the visual-only optimization, and the visual-inertial optimization. The reprojection error for a point can be defined as follows:

$$\|\mathbf{r}\| = \|\mathbf{q} - \text{proj}(\mathbf{p})\|,$$

where $\mathbf{q} \in \mathbb{R}^2$ is the (undistorted) location of the 2D detected feature point and $\mathbf{p} \in \mathbb{R}^3$ is the 3D location of the point to be projected (represented in the frame of the camera used for the projection). The function $\text{proj}(\mathbf{p})$ performs projection and depends on the intrinsic parameters of the camera (f_x, f_y, c_x, and c_y):

$$\text{proj}(\mathbf{p}) = \begin{bmatrix} f_x(\frac{\mathbf{p}_x}{\mathbf{p}_z}) + c_x \\ f_y(\frac{\mathbf{p}_y}{\mathbf{p}_z}) + c_y \end{bmatrix},$$

where $\mathbf{p}_x$, $\mathbf{p}_y$, and $\mathbf{p}_z$ are x, y, and z components (respectively) of $\mathbf{p}$.

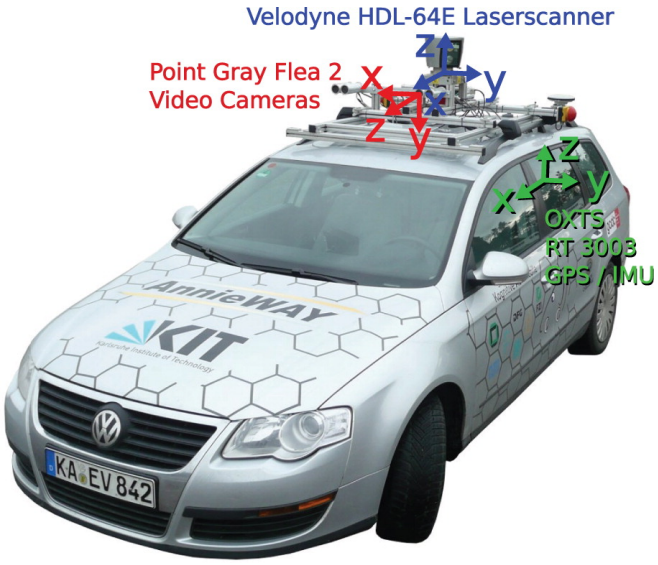

Fig. 3. Data collection platform used in the KITTI dataset [7]. The image is from the paper introducing the dataset.

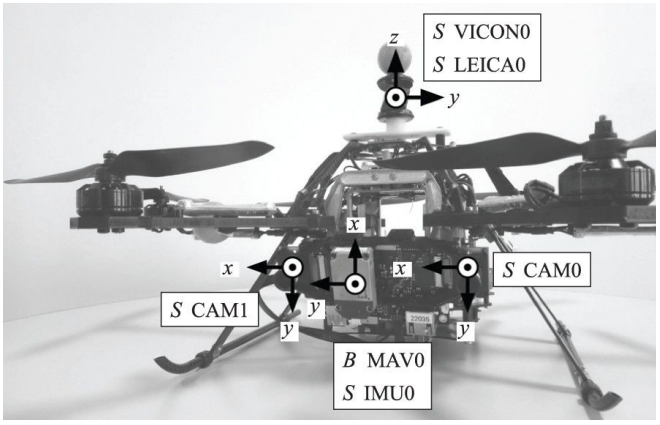

Fig. 4. Overview of the sensor setup on the Asctec Firefly drone used to collect the EuRoC MAV dataset [1]. The drone is equipped with two cameras and one IMU sensor. The image is from the paper which introduced the dataset.

III. Experimental Results

A. Reprojection Error

We present results for reprojection error of each method (analytical visual-only solution, non-linear visual-only optimization, and non-linear visual-inertial optimization) in Table I for both KITTI [7] and EuRoC MAV [1] datasets. We chose a specific pair of frames from the middle of the Machine Hall 01 Sequence (a sequence labeled "easy") on the EuRoC MAV dataset. For the KITTI dataset, we selected a specific pair of frames from the middle of Sequence 22, captured on September 26, 2011.

The mean and standard deviation for the reprojection error of inlier points are reported, as are the number of inliers for each solution. The results in Table I demonstrate that the reprojection error for the analytical solution shows a relatively low error on KITTI (mean inlier error of less than 0.1 pixels), but the error is further reduced after visual-only

non-linear optimization (with both the analytical solution and non-linear optimization showing a similar number of inliers). On the other hand, the visual-inertial optimization shows significantly larger error than both the analytical solution and the visual-only optimization of the analytical solution. In addition, the number of inliers for visual-inertial optimization is significantly smaller than with the two alternative methods. Therefore, visual-inertial optimization leads to significantly worse reprojection error on KITTI. While this might initially seem surprising, both the analytical method and visual-only optimization only use visual data and thus it can be expected that these methods could lead to lower reprojection error than the visual-inertial method (which includes IMU data at a low rate at 10Hz that may conflict slightly with the data captured by the cameras, thus increasing error). However, with all approaches (analytical solution, visual-only optimization, and visual-inertial optimization), there are a non-negligible number of inliers, and the inlier reprojection error remains relatively small (less than 0.5 pixels on average).

The results in Table I on EuRoC MAV show different conclusions. In particular, the analytical solution for EuRoC MAV shows significantly higher error than both visual-only optimization and visual-inertial optimization, with visual-inertial optimization showing the lowest reprojection error on inliers. For these results, we also see a similar number of inliers between the analytical solution and non-linear visual-only optimization, whereas the visual-inertial optimization shows a lower number of inliers, likely due to the inertial data leading to points with higher error as the optimization tries to incorporate inertial constraints that could conflict with purely visual constraints. The seemingly better results of visual-inertial optimization on EuRoC MAV compared to KITTI might stem from the denser inertial data on EuRoC (sampled at 200Hz) compared to KITTI (sampled at 10Hz). This higher sampling rate likely results in a smaller error in preintegrated motion, bringing it closer to the ground truth and, in turn, benefiting the optimization process. The smaller number of inliers could also explain the slightly smaller inlier reprojection error of the visual-inertial approach (as larger errors are excluded from the mean error calculation by being classified as outliers). However, as before with KITTI, the inlier errors for each method are quite small, with the average inlier error for each method being less than 0.5 pixels.

Again, the differing performances of visual-inertial optimization in terms of reprojection error on EuRoC MAV (where visual-inertial optimization outperformed the analytical solution) and KITTI (where visual-inertial optimization performed significantly worse than the analytical solution in terms of reprojection error) could be due to a variety of reasons. For example, KITTI features lower-frequency IMU measurements and larger metric displacements (as the car moves quickly), which could lead the IMU factors to have greater disagreement with the visual information due to drift (thereby negatively affecting the optimization outcome). Meanwhile, the EuRoC MAV dataset features significantly higher-frequency IMU measurements (potentially from a more high-end IMU) and smaller displacements, which (over short time horizons) could lead to more reliable IMU factors that agree more with visual data.

B. Estimated Change in Pose Between Frames

Table II shows the magnitude of the estimated changes in position — $\Delta\mathbf{p}_{ij} \doteq \mathbf{p}_{ij}^{W}$ (in meters) — and orientation — $\Delta\mathbf{R}_{ij} \doteq \mathbf{R}_{j}^{i}$ (in degrees) — between two frames across different methods (analytical solution, visual-only optimization, visual-inertial optimization) and compares them to the ground truth for both the EuRoC MAV and KITTI datasets. The analytical solution and visual-only optimization show identical estimates for the change in position $\Delta\mathbf{p}_{ij}$ on both datasets — equal to one, consistent with their formulations that fix $\Delta\mathbf{p}_{ij}$ to one. Conversely, the visual-inertial optimization presents estimates closer to the ground truth on both datasets — 2.63 m and 2.78 m on KITTI, and 0.32 m on EuRoC MAV for both methods respectively. This improved accuracy is likely due to the incorporation of preintegrated inertial data into the optimization framework, aligning the estimates more closely with ground truth, particularly on EuRoC MAV. Regarding the changes in orientation $\Delta\mathbf{R}_{ij}$, on the other hand, none of the methods consistently provided estimates notably closest to the ground truth.

C. Computation Time

Results for the computation time for each method (in seconds) are shown in Table III. For these results, we note that the analytical solution forms the basis of both non-linear iterative optimization methods; thus, the computation times for both optimization methods reported in Table III should be regarded as *in addition to* the time taken for deriving the analytical solution. Thus, the analytical solution is computed faster than the full optimization solutions, as it forms a prerequisite step for optimization. We also see that the visual-only optimization takes significantly less time to compute than the visual-inertial optimization. Such a result is due to the added computation required to incorporate the inertial terms (including pre-integration and computations for the IMU factors), as well as the greater number of iterative optimization steps needed for convergence of the visual-inertial optimization as compared with the visual-only solution.

We do note that the reported computation times will depend on the parameters of the two-view reconstruction program. For example, using a larger threshold for the ratio test (which is used to discard potentially ambiguous matches) will lead to a larger number of points to use for two-view reconstruction, leading to a longer computation time. In addition, we note that even with the settings used, the run-time performance is still not near real-time (taking seconds to compute a solution in each method). This slow run-time performance can be attributed to un-optimized code with an implementation in Python (rather than C++). Beyond re-writing the code in C++, several optimizations could also be made. For example, the RANSAC implementation of the analytical solution could be parallelized; however, we did not

TABLE I. Reprojection error of analytical solution, visual-only optimization, and visual-inertial optimization methods compared on EuRoC MAV's Machine Hall 01 sequence [1] and KITTI's drive sequence 22 (09/26/2011) [7]. Across both datasets, visual-only optimization produces the lowest reprojection error between the two frames. The number of inliers for the visual-inertial optimization is notably lower than the analytical solution and visual-only optimization.

Dataset	Method	Num of Inliers ⇑	Image 0 Reprojection Error (Pixels) ⇓	Image 1 Reprojection Error (Pixels) ⇓
KITTI [7]	Analytical Solution	**145**	0.087 ± 0.080	0.093 ± 0.088
	Optimization (Visual-Only)	144	**0.075 ± 0.080**	**0.070 ± 0.072**
	Optimization (Visual-Inertial)	40	0.338 ± 0.152	0.321 ± 0.145
EuRoC MAV [1]	Analytical Solution	**118**	0.462 ± 0.724	0.441 ± 0.703
	Optimization (Visual-Only)	112	0.131 ± 0.118	0.124 ± 0.111
	Optimization (Visual-Inertial)	78	**0.112 ± 0.087**	**0.105 ± 0.081**

TABLE II. Magnitude of change in pose between frames for analytical solution, visual-only optimization, and visual-inertial optimization methods compared on EuRoC MAV's Machine Hall 01 sequence [1] and KITTI's drive sequence 22 (09/26/2011) [7]. The visual-inertial optimization's incorporation of inertial data resolves scale ambiguity issues as seen with the relative change in $\Delta\mathbf{p}_{ij}$ (position) between frames when compared to the ground truth.

Dataset	Method	$\Delta\mathbf{p}_{ij}$ [m]	$\Delta\mathbf{R}_{ij}$ [deg]
KITTI [7]	Analytical Solution	1.00	7.11
	Optimization (Visual-Only)	1.00	7.10
	Optimization (Visual-Inertial)	2.63	7.95
	Ground Truth	2.78	7.21
EuRoC MAV [1]	Analytical Solution	1.00	6.88
	Optimization (Visual-Only)	1.00	8.08
	Optimization (Visual-Inertial)	0.32	8.02
	Ground Truth	0.32	7.82

TABLE III. Computation time measurements of single analytical solution, visual-only optimization, and visual-inertial optimization on EuRoC MAV's Machine Hall 01 sequence [1] and KITTI's drive sequence 22 [7]. Although the computation time for the analytical solution is isolated in this table, its output is an input for both optimization-based methods. On both datasets, visual-inertial optimization is an order of magnitude slower than visual-only optimization.

Dataset	Method	Num. of Inliers ⇑	Comp. Time (seconds) ⇓
KITTI [7]	Analytical Solution	**145**	2.91
	Optimization (Visual-Only)	144	**0.83**
	Optimization (Visual-Inertial)	40	11.90
EuRoC MAV [1]	Analytical Solution	**118**	2.41
	Optimization (Visual-Only)	112	**1.03**
	Optimization (Visual-Inertial)	78	10.19

rigorously optimize our code as it was out of scope for this project.

D. Qualitative Examples

We show qualitative results of reprojected points for visual-only and visual-inertial two-view reconstruction on KITTI and EuRoC MAV in Fig. 5 and Fig. 6, respectively. These results reinforce the conclusions in Table I, as both figures show fewer inliers with visual-inertial optimization as compared with visual-only optimization (with the reduction in inliers being particularly prominent on KITTI). In addition, the reprojected 3D points (red) and detected 2D points (blue) for inliers overlap closely, which is consistent with the low inlier reprojection error seen in Table I.

IV. Conclusions

In this work, we show that integrating inertial data into two-view reconstruction optimization resolves scale ambiguity in pose estimates while producing significantly fewer inliers and potentially larger reprojection error. In addition, the visual-inertial strategy requires greater computation time when compared to visual-only methods.

Several possible avenues of future work exist. One interesting direction would be to expand our analysis to incorporate not just two-view reconstruction but also analyze the effect of adding inertial data for sequences of more than two images. Another avenue of work could be to analyze the performance of alternative feature point detectors, such as ORB [10]. The run-time performance of our method leaves much to be desired (with the two-view reconstruction and optimization taking seconds to complete), and a useful real-time visual-inertial odometry system would run at a much higher rate (on the order of milliseconds, to enable pose estimates at camera frame rate). Thus, optimizing our code (for example, by converting it to C++) would be an additional avenue for future work.

ACKNOWLEDGMENT

We would like to thank the authors of the KITTI dataset and EuRoC MAV dataset for making their data openly available. We would also to thank Professor Bretl for the

Fig. 5. Qualitative examples from two frames of the KITTI dataset (on sequence 0022 from 09/26/2011) [7]. In the images, the 2D detected points for inliers are shown in blue, whereas the corresponding reprojected 3D points are shown in red. The two frames on the top used the visual-only optimization method, while the two frames on the bottom used visual-inertial optimization. The visual-inertial optimization results show significantly fewer inliers.

Fig. 6. Qualitative examples from two frames of EuRoC MAV (on sequence Machine Hall 01) [1]. In the images, the 2D detected points for inliers are shown in blue, whereas the corresponding reprojected 3D points are shown in red. The two frames on the top used the visual-only optimization method, while the two frames on the bottom used visual-inertial optimization. The visual-inertial optimization results show fewer inliers.

course, and for the description of two-view reconstruction and structure from motion that was adapted for this report.

REFERENCES

[1] M. Burri, J. Nikolic, P. Gohl, T. Schneider, J. Rehder, S. Omari, M. W. Achtelik, and R. Siegwart, "The EuRoC micro aerial vehicle datasets," *The International Journal of Robotics Research*, vol. 35, no. 10, pp. 1157–1163, 2016.

[2] C. Campos, R. Elvira, J. J. G. Rodríguez, J. M. Montiel, and J. D. Tardós, "Orb-slam3: An accurate open-source library for visual, visual–inertial, and multimap slam," *IEEE Transactions on Robotics*, vol. 37, no. 6, pp. 1874–1890, 2021.

[3] C. B. Duane, "Close-range camera calibration," *Photogramm. Eng*, vol. 37, no. 8, pp. 855–866, 1971.

[4] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[5] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza, "IMU preintegration on manifold for efficient visual-inertial maximum-a-posteriori estimation," in *Robotics: Science and Systems XI*, 2015.

[6] ——, "Supplementary material to: Imu preintegration on manifold for efficient visual-inertial maximum-a-posteriori estimation," Georgia Institute of Technology, Technical Report GT-IRIM-CP&R-2015-001, 2015.

[7] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the KITTI vision benchmark suite," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[8] G. Huang, "Visual-inertial navigation: A concise review," in *2019 International Conference on Robotics and Automation (ICRA)*, 2019, pp. 9572–9582.

[9] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, pp. 91–110, 2004.

[10] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "ORB: An efficient alternative to SIFT or SURF," in *2011 International Conference on Computer Vision*, 2011, pp. 2564–2571.

[11] D. Schubert, T. Goll, N. Demmel, V. Usenko, J. Stückler, and D. Cremers, "The TUM VI benchmark for evaluating visual-inertial odometry," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 1680–1687.