Revisiting Visual-Inertial Structure-From-Motion for Odometry and SLAM Initialization

Georgios Evangelidis and Branislav Micusik

***Abstract*—In this letter, an efficient closed-form solution for the state initialization in visual-inertial odometry (VIO) and simultaneous localization and mapping (SLAM) is presented. Unlike the state-of-the-art, we do not derive linear equations from triangulating pairs of point observations. Instead, we build on a direct triangulation of the unknown $3D$ point paired with each of its observations. We show and validate the high impact of such a simple difference. The resulting linear system has a simpler structure and the solution through analytic elimination only requires solving a 6×6 linear system (or 9×9 when accelerometer bias is included). In addition, all the observations of every scene point are jointly related, thereby leading to a less biased and more accurate solution. The proposed formulation attains up to 50 percent decreased velocity and point reconstruction error compared to the standard closed-form solver, while it is $4\times$ faster for a 7-frame set. Apart from the inherent efficiency, fewer iterations are needed by any further non-linear refinement thanks to better parameter initialization. In this context, we also present a non-linear optimizer that optionally refines the initial parameters. The superior performance of the proposed solver is established by quantitative comparisons with the state-of-the-art solver.**

***Index Terms*—Visual-inertial SLAM, computer vision for automation.**

I. Introduction

VISUAL odometry [26] or SLAM [4] solutions, whereby the pose of an agent within an unknown map is tracked, have become a necessity with the advent of autonomous robots and Augmented Reality (AR) wearables that are equipped with cameras. The underlying geometry problem that needs solving is the Structure-from-Motion (SfM) problem that aims at recovering the structure of a scene, as well as the poses of a moving camera, from image correspondences [10].

In principle, visual data would suffice to solve SfM. In practice, however, apart from the scale ambiguity when a monocular sensor is used, the use of scene-dependent visual observation raises accuracy and efficiency issues. This led to the design of mixed sensors that combine visual sensing with other modalities. A successful paradigm is the fusion of visual with inertial data which has been proven to be beneficial for odometry solutions [5]. The integration of inertial data, typically delivered by an Inertial Measurement Unit (IMU), not only provides valuable information for the ego-motion estimation, but it also resolves ambiguities of visual cues (low-texture, fast motion etc).

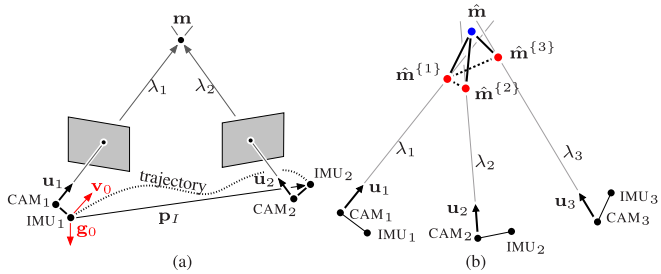

Fig. 1. (a) The *visual-inertial triangulation* principle: the camera baseline is decomposed to the camera-IMU distances and to the IMU displacement $\mathbf{p}_I$ that linearly depends on velocity $\mathbf{v_0}$ and local gravity $\mathbf{g}_0$ through the kinematic equation. (b) Multi-view case: The total distance between the single reconstruction $\hat{\mathbf{m}}$ and all the candidates $\hat{\mathbf{m}}^{\{i\}}$ (solid lines) is minimized by the proposed solver. Instead, [22] minimizes the distance between the candidate pairs (dashed lines), that is, $\tilde{\mathbf{m}}^{\{1\}}$ plays the role of $\hat{\mathbf{m}}$ (*see text for details*).

The resulting VIO problem is usually cast into either a filtering formulation [18], [23] or a chain of optimizations [13], [17]. Therefore, the initialization of the state is required to either start or recover from divergence. The state typically includes the pose and the velocity of the sensor, while the reconstruction of the map points is implicitly required. When the sensor is strictly static, state initialization reduces into a simple orientation problem using only accelerometer data. However, a moving sensor makes the initialization more complicated and visual-inertial SfM (vi-SfM [22]) must be solved. In addition, inexpensive inertial sensors and Rolling-Shutter (RS) cameras make vi-SfM even more challenging due to biased readings and sequential readout, respectively.

Martinelli [22] introduced a linear model for vi-SfM that builds on the triangulation principle, referred here to as *visual-inertial triangulation* (see Fig. 1(a)). The derivation stems from the fact that the camera displacement can be expressed by a kinematic differential equation whereby, under some assumptions, unknown state and auxiliary parameters become linearly dependent. As a result, a closed-form solution becomes feasible.

We build on the same principle, but from the perspective of the multi-view midpoint algorithm [32]. Instead of relating multiple pairs of rays, we jointly relate all the image observations with their generator, that is, the scene $3D$ point (see Fig. 1(b)). This, in general, leads to different constraints on the linear dependence among state and auxiliary parameters with two main advantages. Firstly, it allows the elimination of auxiliary variables at negligible cost, such that the initial velocity and orientation against the gravity axis can be determined by solving a 6×6 linear system. Secondly, the joint dependence of all the point observations

Manuscript received September 23, 2020; accepted January 15, 2021. Date of publication February 5, 2021; date of current version February 23, 2021. This letter was recommended for publication by Associate Editor M. Magnusson and Editor S. Behnke upon evaluation of the reviewers' comments. *(Corresponding author: Georgios Evangelidis.)*

The authors are with the Snap Inc. Fleischmarkt 3-5, 1010 Vienna, Austria (e-mail: georgios@snap.com; bmicusik@gmail.com).

Digital Object Identifier 10.1109/LRA.2021.3057564

from the single yet unknown map point makes the estimator less biased. As a result, an inherently efficient and more accurate closed-form solution becomes available. The differences and advantages against the formulation of [22] are discussed in detail in Sec. IV. We also combine the proposed solver with a non-linear refinement that better models any underlying non-linearity, such as the dependence on the gyroscope bias [14].

To the best of our knowledge, this is the first work that focuses on the special structure of the resulting linear system for vi-SfM. While most prior work solves a large linear system [6], [14], [22], [25], we here show that this is unnecessary. The structure of the derived system matrix allows for a very efficient elimination and solution.

II. Related Work

Although most of the methods assume known initial conditions for VIO [18], [23], there has not been much related work that focuses on the initialization per se. Provided a calibrated device, the linear dependence among state parameters that leads to global linear system was discussed in [15], whereby an observability analysis along with a closed-form initializer were presented. A simpler closed-form solver and its ability to provide finite solutions were then presented in [22]. Our work differs from [22] in the way the linear system is built and solved. The robustness of [22] against biased IMU readings was investigated by [14] and, to account for the gyroscope bias, a non-linear refinement method was proposed. The work of [2] then built on [14], [22] and improved the method via multiple optimization loops.

The above methods adopt a tightly-coupled fusion strategy. Instead, visual SfM problem can be first solved and IMU data can be later integrated in a more loosely-coupled framework [12], [16], [25]. In this context, [16] suggested using visual SfM to obtain camera velocity differences which are then combined with integrated IMU data to recover the scale and gravity direction. The initialization part of [25] used scaleless poses from ORB-SLAM [24] and then solved several sub-problems to initialize the state and the biases along with the absolute scale. This multi-step solution for the parameter initialization was then adapted in [27].

An uncalibrated device makes the initialization harder [6], [12]. Even if the biases are assumed known, the unknown orientation between camera and IMU makes the model non-linear and iterative optimization is necessary. In [6], a linear solver feeds a non-linear estimator. Instead, [12] builds on the multi-step approach of [25] to jointly calibrate the extrinsics and initialize the state parameters. In a real scenario, however, the joint solution of calibration and initialization problem using only the very first few frames might make VIO or SLAM prone to diverge.

All the above methods silently assume a global-shutter sensor. Consumer devices, though, are mostly equipped with RS cameras. This means that RS effects need to be properly handled [1], [11], [19]. In [29], a continuous-time SLAM model that uses temporal basis functions copes with continuous capturing and the high-rate of IMU data. In this line of work, [30] uses a better spline representation that avoids singularities of [29] and minimizes in batch a visual-inertial cost function. A very similar problem is solved by [31] but the cost terms are properly balanced by predicted residual variances. None of these methods, however, focuses on SLAM initialization. They typically solve smaller problems to partially initialize the estimators, or simply start from identity poses and structure at infinity. Since our test platform is a stereo RS rig, we take into account the RS readout time and the different camera poses per image row in Sec. VI (see also the discussion about RS in Sec. IV).

III. Problem Formulation

Assume a $3D$ point $\mathbf{m}$ in a reference coordinate system (RCS) being seen at N different times by a moving camera:

$$\lambda_i \mathtt{R}_{C_i} \mathbf{u}_i + \mathbf{p}_{C_i} = \mathbf{m}, \quad i = 1, \ldots, N, \tag{1}$$

where $\mathbf{u}_i$ is the normalized (calibrated) unit vector of the image observation, λ_i is the distance between the point and the camera, $\mathtt{R}_{C_i}$ is the matrix that characterises the rotation from the camera coordinate system (CCS) to the RCS, $\mathbf{p}_{C_i}$ is the camera position in the RCS, at the time $t_i = n_i T_s$, where $n_i \in \mathbb{N}$ and T_s is a sufficiently low sampling time.

Assume also an intrinsically and extrinsically (against the camera) calibrated IMU that is rigidly mounted to the moving rig. Without loss of generality, the data sampling period of the inertial sensor can be set to T_s. If we now consider the IMU frame at time $t_0 = 0$ as the RCS, the camera position $\mathbf{p}_{C_i}$ can be written as

$$\mathbf{p}_{C_i} = \mathbf{p}_{I_i} + \mathtt{R}_{I_i} \mathbf{p}_C^I, \tag{2}$$

where $\mathtt{R}_{I_i}$, $\mathbf{p}_{I_i}$ represent the orientation and position, respectively, of the IMU in the RCS at time t_i and $\mathbf{p}_C^I$ is the known position of the camera in the IMU frame.

Let us now consider a constant acceleration kinematic model [21] that describes the position of the IMU over time. Provided that $\mathbf{p}_{I_0} = \mathbf{0}$ and $\mathbf{v}_0$ are the position and velocity, respectively, of the IMU in the RCS at t_0, the successive integration of acceleration data writes

$$\mathbf{p}_{I_i} = t_i \mathbf{v}_0 + \frac{t_i^2}{2} \mathtt{R}_W \mathbf{g}_W + \frac{T_s^2}{2} \sum_{k=0}^{n_i - 1} \beta_{ki} R_{I_k} (\boldsymbol{\alpha}_{I_k} + \mathbf{b}_a), \tag{3}$$

where $\mathbf{g}_W$ is the gravity vector in the world coordinate system (WCS), $\mathtt{R}_W$ is the matrix that represents the rotation from the WCS to the RCS, $\boldsymbol{\alpha}_{I_k}$ is the measured acceleration at time t_k, $\mathbf{b}_a$ is the accelerometer bias compensation that is considered constant for short integration times, and $\beta_{ki} = 2(n_i - k) - 1$ is the resulting coefficient after unfolding recursive integrations.

The IMU calibration implies rigid corrections of gyroscope and accelerometer axes, but sensor biases are affected by several sources and their refinement may be beneficial. As seen in (3), $\mathbf{b}_a$ is added in a linear way. However, a gyroscope bias offset would break the linearity and its use through a non-linear refinement, when needed, is preferred [2], [14]. If we now assume corrected gyroscope data, the rotation matrix can be computed by [21] $\mathtt{R}_{I_i} = \prod_{k=0}^{n_i - 1} \exp(\boldsymbol{\omega}_k T_s)$, where $\boldsymbol{\omega}_k$ is the gyroscope measurement. As a result, $\mathtt{R}_{C_i}$ can be as well estimated using the known orientation of the CCS in the IMU frame $\mathtt{R}_C^I$, that is, $\mathtt{R}_{C_i} = \mathtt{R}_{I_i} \mathtt{R}_C^I$.

The equations (1), (2) and (3) can be combined into a single matrix form, that is,

$$\begin{bmatrix} t_i \mathtt{I}_3 & \frac{t_i^2}{2} \mathtt{I}_3 & \mathtt{B}_i & -\mathtt{I}_3 & \mathtt{R}_{C_i} \mathbf{u}_i \end{bmatrix} \begin{bmatrix} \mathbf{z} \\ \mathbf{m} \\ \lambda_i \end{bmatrix} = \mathbf{c}_i, \tag{4}$$

where $\mathbf{z} = [\mathbf{v}_0^\top, \mathbf{g}_0^\top, \mathbf{b}_a^\top]^\top$, $\mathbf{g}_0 = \mathtt{R}_W \mathbf{g}_W$ is the gravity in the RCS, $\mathbf{c}_i = -\mathtt{R}_{I_i}\mathbf{p}_I^C - \frac{T_s^2}{2}\sum_{k=0}^{n_i-1}\beta_{ki}R_{I_k}\boldsymbol{\alpha}_{I_k}$ is a constant vector that includes a weighted sum of rotated accelerations, $\mathtt{B}_i = \frac{T_s^2}{2}\sum_{k=0}^{n_i-1}\beta_{ki}R_{I_k}$ is a weighted sum of rotation matrices, and $\mathtt{I}_3$ is the 3×3 identity matrix. As a result, each visual observation adds three equations and one unknown λ parameter, thus shaping a linear system of $3N \times (N+12)$ from a single point. Multiple points are typically needed and a large linear system is built.

It is customary to align the z-axis of the WCS with the gravity axis and set $\mathbf{g}_W = [0,\ 0,\ \gamma]^\top$, where γ is the gravity magnitude. This makes $\mathbf{g}_0$ a scaled version of the third column of matrix $\mathtt{R}_W$, while $\|\mathbf{g}_0\|_2 = \gamma$. Since $\mathtt{R}_W$ is estimated from $\mathbf{g}_0$ and $\mathbf{g}_W$, any rotation around $\mathbf{g}_W$ cannot be recovered. In addition, $\mathbf{b}_a$ is not always separable from $\mathbf{g}_0$ unless the system rotates, since $\mathtt{R}_{I_k} = \mathtt{I}_3$ implies $\mathtt{B}_i = \frac{t_i^2}{2}\mathtt{I}_3$, which is equal to the coefficient of $\mathbf{g}_0$. As a result $\mathtt{R}_{I_k} \neq \mathtt{I}_3$ makes $\mathbf{b}_a$ observable, while the constraint may be needed depending on the underlying case. E.g., in the particular case of rotation around at least two axes, the gravity constraint is not necessary for the biased case (see Property 15 in [22]).

IV. Closed-Form Solution

Let us consider M map points, stacked into a vector $\boldsymbol{\mu} = [\mathbf{m}_1^\top, \ldots, \mathbf{m}_M^\top]^\top$, and let λ_{ji}, $\mathbf{u}_{ji}$, $\mathtt{B}_{ji}$ and $\mathbf{c}_{ji}$ denote the above defined variables for the j-th point projected into the camera frame at the time t_i. For the sake of simplicity, we assume that each point has the same number of N observations (captured at N different times) while in practice each point can have a different number of observations. Then, the entire linear system can be written as

$$[\mathtt{V} \mid \mathtt{W} \mid \mathtt{Q}] \begin{bmatrix} \mathbf{z} \\ \boldsymbol{\mu} \\ \boldsymbol{\lambda} \end{bmatrix} = \mathbf{c}, \tag{5}$$

where $\boldsymbol{\lambda} = [\lambda_{11} \ldots, \lambda_{MN}]^\top$, $\mathtt{V}$ is a $3MN \times 9$ matrix with $[t_{ji}\mathtt{I}_3 \ \ \frac{t_{ji}^2}{2}\mathtt{I}_3 \ \ \mathtt{B}_{ji}]$ being the rows that correspond to ji-th observation, $\mathtt{W}$ is a $3MN \times 3\ M$ block matrix of diagonal-structure[1] with each $3N \times 3$ block being $\mathtt{Y}_j = -[\mathtt{I}_3, \ldots, \mathtt{I}_3]^\top$ (the block that multiplies $\mathbf{m}_j$), $\mathtt{Q}$ is a $3MN \times MN$ block matrix of diagonal structure with each 3×1 block being $\mathbf{q}_{ji} = \mathtt{R}_{C_{ji}}\mathbf{u}_{ji}$ (the block that multiplies λ_{ji}), and $\mathbf{c}$ is a $3MN$-element vector, shaped after stacking $\mathbf{c}_{ji}$.

The vector $\boldsymbol{\lambda}$ contains auxiliary variables and its elimination is meaningful. Commonly, one would multiply from the left with the projection matrix $\mathtt{P} = \mathtt{I} - \mathtt{Q}(\mathtt{Q}^\top\mathtt{Q})^{-1}\mathtt{Q}^\top$. Recall, however, that each block of $\mathtt{Q}$ is a unit vector, hence $(\mathtt{Q}^\top\mathtt{Q})^{-1} = \mathtt{I}$. As a consequence, the block diagonal matrix $\mathtt{P} = \mathtt{I} - \mathtt{Q}\mathtt{Q}^\top$ can be computed without any inversion and such an elimination comes at negligible cost. The system one needs to initially construct is the following:

$$[\mathtt{PV} \mid \mathtt{PW}] \begin{bmatrix} \mathbf{z} \\ \boldsymbol{\mu} \end{bmatrix} = \mathtt{P}\mathbf{c}. \tag{6}$$

Despite the large initial unknown vector in (5), the resulting linear system in (6) has now less unknowns than the one of [14], [22], since $M \ll MN$. Note that homogeneous equations that relate pairs of λ-based reconstructed points are also pushed to the linear system of [22], thus increasing the number of rows. We do not add such constraints here since all the image observations of a single point are *jointly* related through a single unknown parameter.

We now proceed with a second elimination step that further reduces the above linear system into one that only solves for the IMU state. One can optionally back-substitute to compute the points, when needed. To this end, we apply the projection operator $\mathtt{I} - \mathtt{PWHW}^\top\mathtt{P}^\top$, where $\mathtt{H} = (\mathtt{W}^\top\mathtt{PW})^{-1}$ since $\mathtt{P}$ is symmetric and idempotent. However, it is straightforward to show that $\mathtt{W}^\top\mathtt{PW}$ is a block diagonal matrix of size $3M \times 3\ M$, with each block being defined by $N(\mathtt{I}_3 - \frac{1}{N}\sum_{i=1}^{N}\mathbf{q}_{ji}\mathbf{q}_{ji}^\top)$. Hence, computing $\mathtt{H}$ requires inverting each 3×3 block, which is given by a simple analytical formula.

The elimination of map points finally leads to the following overdetermined system with 9 unknowns,

$$\mathtt{PGV}\mathbf{z} = \mathtt{PG}\mathbf{c}, \tag{7}$$

where $\mathtt{P} = \mathtt{I} - \mathtt{Q}\mathtt{Q}^\top$ and $\mathtt{G} = \mathtt{I} - \mathtt{WHW}^\top\mathtt{P}$. Apart from the fact that $\mathtt{P}$ is a block diagonal matrix, the computation of $\mathtt{WHW}^\top$ from $\mathtt{H}$ involves only repetitions of $\mathtt{H}$'s blocks. As a consequence, a least-squares solution leads to a 9×9, or a 6×6 in the unbiased case, linear system which can be efficiently built. It is worth mentioning that one can iteratively update the resulting symmetric matrix after visiting every point, thus building the linear system very efficiently, even without using large and/or sparse matrices. The constraint $\|\mathbf{g}_0\|_2 = \gamma$ is now added to the reduced system. There are several options to solve the constrained problem, e.g., solving the unconstrained one and using a one-step refiner, adding a quadratic constraint in a convex optimization framework, or applying QR decomposition to name a few.

When the point reconstruction is required, one can use the following equation to compute the coordinates:

$$\boldsymbol{\mu} = \mathtt{HW}^\top\mathtt{P}(\mathbf{c} - \mathtt{V}\mathbf{z}^*), \tag{8}$$

where $\mathbf{z}^*$ is the solution of (7).

Finally, the rotation matrix $\mathtt{R}_W$ is computed from the vectors $\mathbf{g}_0$ and $\mathbf{g}_W$, while $\mathbf{v}_0$ and $\mathbf{m}_j$ are expressed in the WCS by $\mathtt{R}_W^\top\mathbf{v}_0$ and $\mathtt{R}_W^\top\mathbf{m}_j$, respectively. Since the origin of the WCS can be arbitrarily chosen, it can be identified with the origin of the RCS.

The above formulation can be seen as a generalization of the multi-view midpoint triangulation algorithm [32], which builds on known poses to reconstruct the point that is closest (on average) to the observation rays. Here, the poses are unknown. The camera positions depend on the same unknown set, thus making the reconstruction of different points dependent on each other. One might assume the parallelism of $\mathbf{m} - \mathbf{p}_{C_i}$ with $\mathbf{u}_i$, thus setting their cross product equal to zero and ignoring λ_i (DLT, [10]). However, the midpoint algorithm is simpler and more efficient for the multi-view case, in particular here where each point is not reconstructed independently. Moreover, it directly provides the sign of λ's for a cheirality check.

Rolling Shutter: We deliberately refer to the time index in (1) in order to consolidate global and rolling shutter cases. Simply, all the visual observations of a single image have the same timestamp in global shutter mode. Instead, RS cameras capture image rows sequentially and the timestamp of a visual observation can be given by $t_i = \tau_i + \tilde{u}_i^y\Delta\tau$, where τ_i is the timestamp of the first image row, $\tilde{u}_i^y$ is the lens-distorted y-coordinate (image

[1] unlike the block-diagonal matrix, each block has non-square size.

row) of $\mathbf{m}$'s projection onto the image and $\Delta\tau$ is the readout time per image row. As a result, each row has a different pose and (3) differs per image feature. More than one IMU samples correspond to a single image and an interpolation scheme can provide an IMU sample per t_i.

Stereo camera: The proposed modelling is valid with either a monocular or a binocular sensor. Recall that $\mathbf{m}$ is expressed in RCS and $\mathbf{u}_i$ may regard any frame of either sensor. The integration time needed to reliably initialize the state may be different though. The stereo baseline leads to larger camera displacement, which in turn leads to better triangulation. For instance, given two successive stereo frames, the displacement from the *current left* to the *next left* camera is most of the times smaller than the distance between the *current left* and the *next right* camera. As a result, the integration time that is needed to cover a sufficient baseline is smaller, that is, less frames can be considered.

Resolvability: The resolvability of vi-SfM problem is discussed in detail in [22]. Our solver differentiates in the way Eq. (4) is used, that is, the linear dependencies remain the same. Provided a varying acceleration, a minimum number of 5 frames would suffice for a unique solution, even with a single point. When the system also rotates in 3D (around two or more axes), the biased case is uniquely solvable, when at least 6 frames are used. The use of a second or third point relaxes these constraints in some cases [22]. In practice, the use of a bunch of long tracks is recommended. Therefore, the above numbers could regard non-successive frames.

The stereo camera makes the problem solvable with less frames, since points are observable even from single stereo frames. For instance, when acceleration and rotation vary, 3 frames would suffice to estimate the state in the unbiased case for any number of points, while one more frame is required when bias is included. In the particular case of RS cameras, even less frames make the problem solvable because each scanline can be seen as a different "frame". The analysis of several motion and structure cases [22] for stereo and/or rolling shutter cameras is long and we leave it for a feature work. It is worth mentioning that, in practice, more frames are required for a reliable solution.

Outlier handling: So far, we silently assume that the visual correspondences are inliers, up to a reasonable tracking error. In practice, the tracks may include outliers. The above solution can be used as a minimal solver combined with a RANSAC-like scheme to cope with the outliers. But this would make the initializer quite slow. A better approach is to combine RANSAC with the tracker that provides visual correspondences, so that the solver receives outlier-free data. As an example, RANSAC on fundamental matrix removes inconsistent matches in [2]. In Sec.VI, we adopt the same selection scheme on raw matches that come from the ECC tracker [8] on FAST corners [28]. Although the RS effect makes the essential matrix globally invalid, it is sufficient to detect outliers given that any time-varying rotation is compensated via gyroscope data integration. Instead, the generalized essential matrix [3] can be used when the RS effect is quite strong (very fast camera motion).

A. Relation to [22]

Eq. (4) is also used as the starting equation in [22], but pairwise ray differences eliminate the unknown point. However, all the possible pairs should be considered for an equivalent solution, since (4) does not exactly hold. In addition, the mutually dependent reconstruction of multiple points makes the two solutions even more different.

The advantages of the proposed solver compared to [14], [22] can be summarized as follows:

- The linear system has a simple, uniquely defined structure and auxiliary parameters are eliminated at negligible cost. This leads to a solver that only requires inverting, or decomposing, a very small matrix. The structure of the system matrix in [14], [22] depends on how the observations are paired and on how the points appear in frames. The elimination of λ's in [14], [22] would require inverting a large sparse matrix.
- The estimation is better conditioned since all the point observations are jointly related through the single yet unknown point that generates them.
- The scene points in a single RCS are directly reconstructed by the solver. The model naturally extends to a bundle adjustment scheme with the same parameters, e.g. by applying a projection operator. Instead, initial map points in a single RCS or CCS should be pre-computed when the solver of [14], [22] is used.

V. Non-Linear Refinement

The underlying application may require a more accurate initializer while the available hardware may support computationally demanding operations. Therefore, we here discuss a refinement of the initial state and the reconstructed points within an iterative optimization framework. Note that we do not solve a multi-keyframe visual-inertial bundle adjustment problem whereby multiple states are optimized [2], which could be also employed for the temporal state tracking in an incremental way. Instead, we simply optimize the image reprojection error *w.r.t.* the initial single-frame state and structure, and optionally the biases.

Despite the geometric meaning of the closed-form minimization, visual observations are back-projected in $3D$ space through an unknown depth, which may give some unwanted freedom to the solver. The projection of the error distance onto the space where measurements are observed (image or plane $z = 1$ of the CCS) makes more sense, while it makes λ disappear. The total error that needs minimizing becomes

$$f(\mathbf{x}) = \sum_{i,j} d(\pi(\mathtt{R}_{C_i}^\top \mathbf{m}_j - \mathtt{R}_{C_i}^\top \mathbf{p}_{C_i}), \pi(\mathbf{u}_{ji})) \tag{9}$$

where $\mathbf{x} = [\mathbf{z}^\top, \boldsymbol{\mu}^\top]^\top$, $d(\cdot,\cdot)$ is the (squared) Euclidean distance of the arguments, and $\pi(\mathbf{u}) = [u_x/u_z,\ u_y/u_z]^\top$ is the common perspective projection. When the constraint $\|\mathbf{g}_0\|_2 = \gamma$ must be enforced, a constrained optimization can be avoided by a proper parameterization of $\mathbf{g}_0$. Since the rotation around the gravity axis is not observable, the unknown rotation can be paremeterized by the axis-angle vector $\boldsymbol{\phi} = [\phi_x, \phi_y, 0]^\top$. Using the Rodrigues formula [9] (exponential map) that provides $\mathtt{R}_W$ from $\boldsymbol{\phi}$, the equation $\mathbf{g}_0 = \mathtt{R}_W \mathbf{g}_W$ leads to the parameterization

$$\mathbf{g}_0 = \gamma \left[\frac{\sin(\|\boldsymbol{\phi}\|)}{\|\boldsymbol{\phi}\|}\phi_y,\ -\frac{\sin(\|\boldsymbol{\phi}\|)}{\|\boldsymbol{\phi}\|}\phi_x,\ \cos(\|\boldsymbol{\phi}\|) \right]^\top, \tag{10}$$

which makes the constraint valid.

The gyroscope bias $\mathbf{b}_g$ is inserted into the model in a non-linear way [9]. We evaluate the contribution of modelling $\mathbf{b}_g$ in Sec. VI.

VI. Experiments

A. Experimental Setup

We are experimenting with a stereo RS camera rigged with an IMU. To get realistic data with ground truth (GT) structure and poses, we process data from Snap Spectacles. States resulting from a Kalman filter on visual-inertial data play the role of GT states and high-order splines on IMU data provide ideal gyroscope and acceleration readings, such that a continuous integrator perfectly interpolates between the filter states. IMU data are then sampled at 800 Hz and finally, noise and time varying biases are added based on the calibrated variances of the used device. The frame readout time is 10 ms, that is, 8 IMU samples are available per frame.

A virtual stereo RS camera of VGA resolution follows the resulting trajectory within a virtual $3D$ scene and images are rendered at 30Hz [7]. When GT image correspondences are needed, single virtual $3D$ points along with their reprojections are created. Given a reference frame, we back-project 100 evenly spaced image points with random depth in range [1 m, 15 m] and the points are in turn re-projected into adjacent frames. As mentioned, a RANSAC scheme on matches from an ECC-based tracker [8] on FAST corners [28] provides real correspondences on rendered images. The tracked features do not necessarily appear in all the frames of the window. We consider tracks from 5 or 7 stereo frames, but we modify the frame downsampling factor to change the integration time and the rig displacement. A linear interpolation scheme estimates IMU samples at feature times. In all the experiments below, the initialization problem is solved from scratch per tested frame window, without using any prior information from previous solutions.

B. Closed-Form Performance Evaluation

We compare the performance of the proposed solver against the solver of [22]. We do not consider biases here and we deal with them in Sec. VI-C. We refer to the proposed solver as *point-to-observation* (*p2o*) pairing scheme as opposed to *observation-to-observation* (*o2o*) pairing paradigm of [14], [22].

The GT states regard a sequence from a Spectacles wearer who is almost static for 1 s and s/he then walks forward for 12 s while looking around. Such a sequence mixes translational and rotational movements while it includes instant stationary parts. We here use virtual points and the tracks of GT image observations that are affected by additive Gaussian noise of known deviation σ_u.

First, we test the robustness of the solvers in terms of the tracking error, which found to be the dominant parameter that affects the performance. For each value of σ_u in the range $[0, 0.5]$ pixel, a sliding window of 5 temporarily downsampled frames is used. The downsampling factor is $N_f = 3$, thus defining a total integration time of 0.46 s. Any frame window with GT velocity magnitude below 0.01 m/s is discarded. In total, 50 realizations per window are executed. The *relative* magnitude error and the angular error are the evaluation criteria to assess the velocity and gravity direction estimation, respectively. As for the point reconstruction error, the error distance per point is normalized by its depth. The average error as a function of σ_u, with and without the gravity norm constraint, is shown in Fig. 2. As seen, the proposed *p2o* formulation is more robust and provides more accurate estimations, while its superiority against the *o2o* scheme grows with the tracking error. When the gravity norm constraint is enforced, the performance improvement is not noticeable in most of the cases.

Next, we investigate how the integration time affects the performance. We repeat the experiment with $\sigma_u = 0.3$ and test several downsampling factors N_f from 1 to 9, which implies integration times from 0.13 s to 1.46 s. Fig. 3 shows the error as a function of integration time. As expected, the shorter the integration time is, the more sensitive the solvers are. The angle error of gravity estimation, in particular, can reach 3° at very short integration times. However, it seems that the solvers provide acceptable results after 0.5 s. The proposed solver outperforms and achieves more accurate estimation at any integration time. The velocity and the point reconstruction error is decreased by 50% across the whole tested range. Again, the benefit from enforcing the gravity magnitude constraint is minor.

As verified by [14], the accelerometer bias, when separable from the gravity, does not affect the closed-form solution. Rather, a relatively large gyroscope bias may affect the performance. We reached similar conclusions for both solvers. Finally, we did not notice negative λ's in our experiments.

C. Refinement Performance Evaluation

We here evaluate the contribution of the closed-form solvers to the non-linear refinement. Our refence is the refiner discussed in Sec. V. The rotated gravity vector is modelled by (10) and the gyroscope bias is optionally modelled. To initialize the structure when *o2o* solver is used, we average the λ-based reconstructions per point. A modification of the Levenberg-Marquardt framework of [20] is used for minimization. All the thresholds of the stop criteria are set to 10^{-9} and we let the algorithm terminate.

Fig. 4 shows the error per iteration, for the case of $N_f = 3$ and $\sigma_u = 0.3$, averaged over all tested frame windows. Notably, when starting from the *o2o* solution, more than five refinement iterations are needed to reach the accuracy of the *p2o* solver. As a result, similar accuracy can be achieved with much less operations. All of the counterparts reach comparable floor values after 12 iterations. When the gyroscope bias is modelled, the extra non-linearities may lead to slower convergence. We also noticed the superiority of the proposed solver in terms of the frequency of convergence, when a lower number of iterations is allowed (e.g. 5 or 10).

The contribution from bias modelling is not evident here. This is, most likely, because of low noise levels of the used IMU compared to the dominant tracking error. We experimentally confirmed the ability of the refiner to estimate high yet unrealistic biases. When an iterative optimizer is used, another approach is to use the unconstrained linear solver to get the initial velocity and gravity, and then let the optimizer refine the parameters and estimate the biases.

D. Real Correspondences

Different trajectories and motions are here combined with different $3D$ scenes, while the rig is static at the beginning. An extended Kalman filter, well initialized from

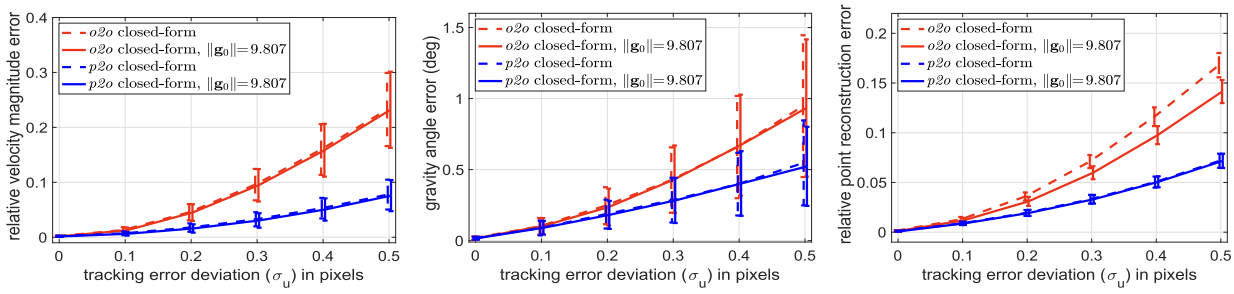

Fig. 2. (a) Velocity estimation, (b) gravity orientation estimation and (c) point reconstruction error as a function of point tracking error; the integration time is 0.46 seconds ($N_f = 3$).

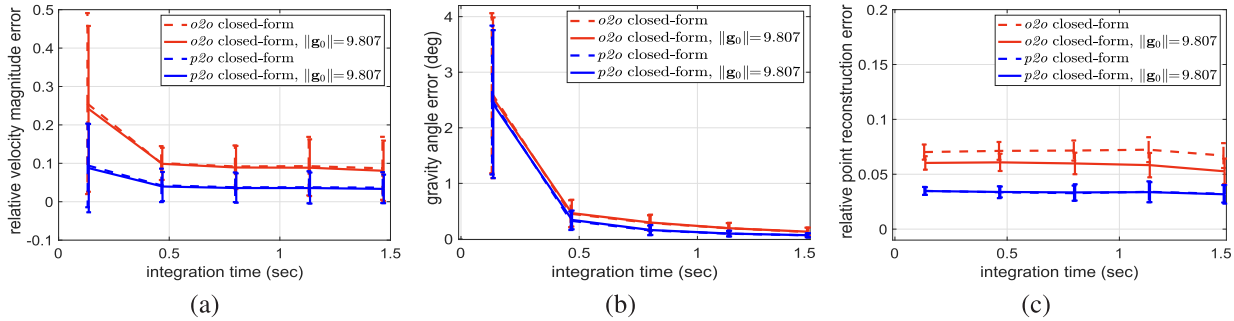

Fig. 3. (a) Velocity estimation, (b) gravity orientation estimation and (c) point reconstruction error as function of integration time; the point tracking error deviation is 0.3 pixels.

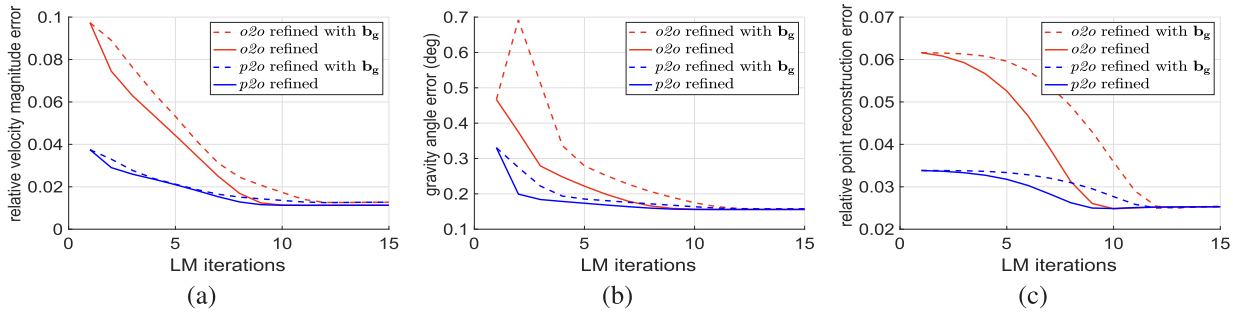

Fig. 4. (a) Velocity estimation, (b) gravity orientation estimation and (c) point reconstruction error attained per iteration with non-linear refinement; the point tracking error deviation is 0.3 pixels and the integration time is 0.46 seconds. Dashed lines show errors when the gyroscope bias is estimated.

accelerometer data due to static part, is employed for reference [23]. A maximum number of 200 points is tracked per frame. Again, 5 downsampled frames with $N_f = 3$ are used per frame window. To compensate for mismatches and non-Gaussian error, Cauchy loss [10] with (9) is also tested. A maximum number of 15 iterations is used for the refinement.

Table I summarizes the performance of the algorithms on three sequences: WALKING, HEADMOVING and RUNNING. The average error over all frame windows is shown. The *p2o* solver obtains better estimates than the *o2o* solver. When a non-linear refiner follows, the error further decreases. Fig. 5 shows the error over time. The *absolute* velocity magnitude error is here shown. Interestingly, for WALKING and HEADMOVING sequences, the velocity estimation from running the closed-form per frame window is comparable with the one from Kalman filter, despite the accurate initialization of the latter. The RUNNING sequence is more challenging because of jumping while jogging. The

TABLE I
COMPARISON OF AVERAGE PERFORMANCE PER SEQUENCE

	velocity error percentage(%) / gravity error in degrees(°)		
	WALKING	RUNNING	HEADMOVING
o2o closed-form	2.98 / 0.147	5.01 / 0.607	9.82 / 0.216
p2o closed-form	**2.76 / 0.143**	**2.42 / 0.339**	**6.52 / 0.205**
o2o refined	2.77 / 0.141	0.45 / 0.145	5.40 / 0.164
p2o refined	2.77 / 0.140	0.45 / 0.145	5.40 / 0.163
o2o refined (Cauchy)	2.72 / 0.124	2.96 / 0.313	4.77 / 0.149
p2o refined (Cauchy)	2.71 / 0.125	0.62 / 0.166	4.76 / 0.148
VIO (Kalman filter)	3.05 / 0.099	3.84 / 0.258	4.85 / 0.151

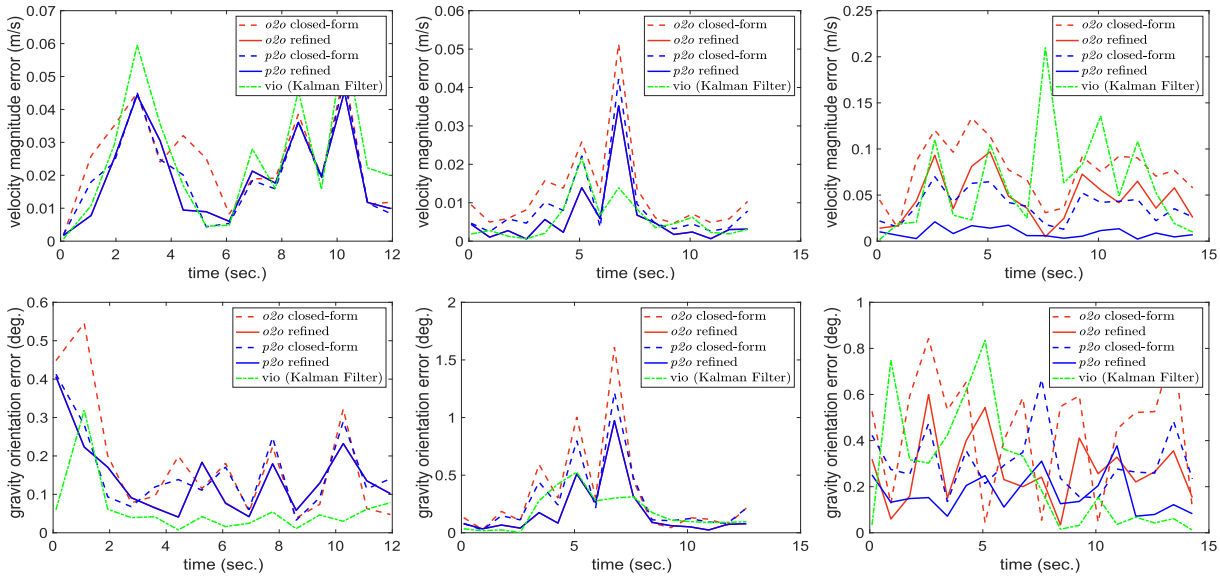

Fig. 5. (*top*) Velocity and (*bottom*) gravity orientation error per frame when tracking features on rendered images; the integration time is 0.46 s.

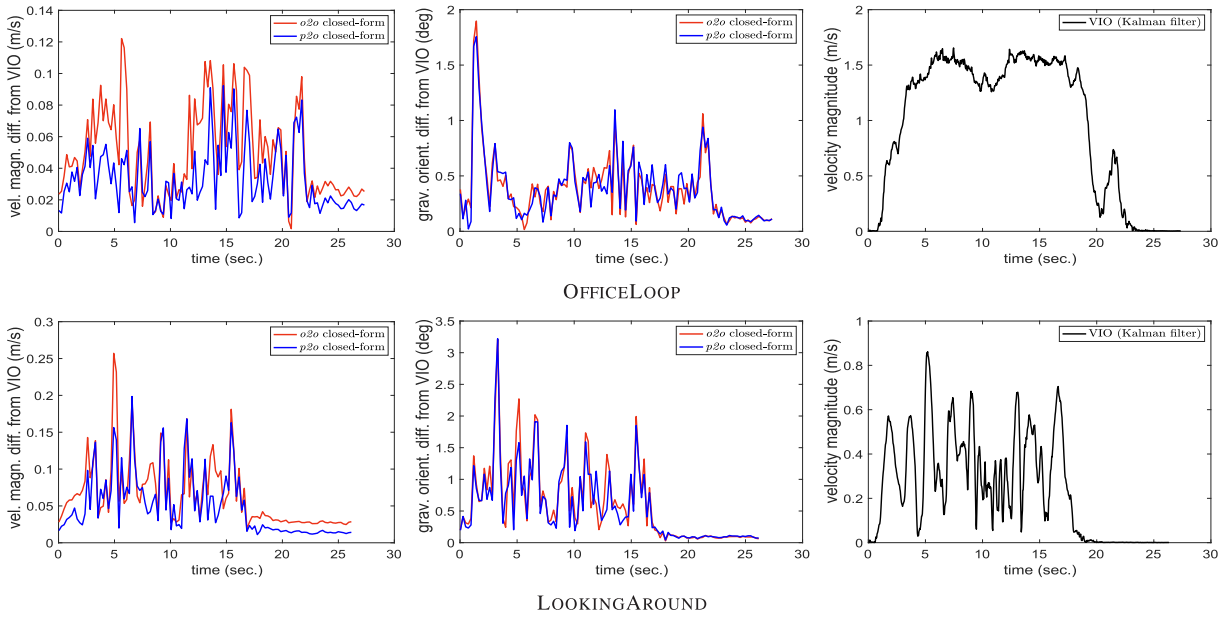

Fig. 6. (*Left*) Velocity and (*middle*) gravity orientation difference from VIO baseline (extended Kalman filter) after testing the closed-form solvers on real data delivered by Snap Spectacles. *Right:* The VIO velocity magnitude per frame.

proposed solver combined with the refiner clearly outperforms in this case.

E. Real Data

We here focus on the closed-form solvers and test them on two real sequences acquired from Snap Spectacles in a typical open office space. The wearer is static in the beginning as well as in the end of the recording. The first sequence, named OFFICELOOP, is a 25 m loop-shaped walking sequence within the office. The second sequence, named LOOKINGAROUND, is more challenging and has rotational motion with strong velocity variation where the wearer looks around without stepping. As with the synthetic data, VGA images at 30 Hz and IMU data at 800 Hz are used.

We *re-initialize* the state of every frame using a moving window of 7 frames with $N_f = 3$. Since GT data are not available, we show in Fig. 6 the deviation from the VIO baseline which uses prior information per state estimation. The two solvers provide similar gravity orientations, while *p2o* provides more accurate velocities. As expected, both solvers obtain less accurate estimations for LOOKINGAROUND sequence due to the rapid velocity and rotation changes.

TABLE II
COMPARISON OF AVERAGE TIMES PER SEQUENCE

	average running times (C++, i7@2.6GHz)	
	OFFICELOOP	LOOKINGAROUND
o2o closed-form	4.27ms	7.50ms
p2o closed-form	**1.18**ms	**1.96**ms

Time comparison: Both closed-form solvers were implemented with Eigen C++ library. A sparse linear system and solver are used for *o2o*. Instead, the block structure of matrices in (7) enables the direct construction of the least-squares solution system. That is, for the unbiased case, we start with a 6×6 zero matrix and 6×1 zero vector and we update them after processing each track (point). Such an implementation is memory efficient too. Given IMU data and the visual tracks, the average build-and-solve times over all tested frame windows are shown in Table II. As a result, we obtain a $4\times$ faster initialization for a 7-frame window.

VII. CONCLUSION

A new closed-form solver for the vi-SfM problem was suggested, in the context of VIO and SLAM initialization. The mathematical derivation along with the experimental validation show that the solver is more accurate and faster than the state-of-the-art. Either as a stand-alone solver or combined with a non-linear optimizer that further refines the initial state, it offers a significant speedup to the initialization phase of VIO and SLAM pipelines.

REFERENCES

[1] A. Bapat, T. Price, and J.-M. Frahm, "Rolling shutter and radial distortion are features for high frame rate multi-camera tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 4824–4833.

[2] C. Campos, J. Montiel, and J. Tardós, "Fast and robust initialization for visual-inertial SLAM," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2019, pp. 1288–1294.

[3] Y. Dai, H. Li, and L. Kneip, "Rolling shutter camera relative pose: Generalized epipolar geometry," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 4132–4140.

[4] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse, "MonoSLAM: Real-time single camera slam," in *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 6, pp. 1052–1067, Jun. 2007.

[5] J. Delmerico and D. Scaramuzza, "A benchmark comparison of monocular visual-inertial odometry," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2018, pp. 2502–2509.

[6] T. Dong-Si and A. I. Mourikis, "Estimator initialization in vision-aided inertial navigation with unknown camera-imu calibration," in *Proc. IEEE Int. Conf. Intell. Robots Syst.*, 2012, pp. 1064–1071.

[7] "Unreal engine," *Epic Games*, 2019. [Online]. Available: http://www.unrealengine.com

[8] G. D. Evangelidis and E. Z. Psarakis, "Parametric image alignment using enhanced correlation coefficient maximization," in *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 30, no. 10, pp. 1858–1865, Oct. 2008.

[9] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza, "On-manifold preintegration for real-time visual-inertial odometry," *IEEE TRO*, vol. 33, no. 1, pp. 1–21, Feb. 2017.

[10] R. I. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. 2nd edition, New York, NY, USA: Cambridge Univ. Press, 2004.

[11] J. Hedborg, P.-E. Forssen, M. Felsberg, and E. Ringaby, "Rolling shutter bundle adjustment," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2012, pp. 1434–1441.

[12] W. Huang, H. Liu, and W. Wan, "An online initialization and self-calibration method for stereo visual-inertial odometry," *IEEE TRO*, vol. 36, no. 4, pp. 1153–1170, Aug. 2020.

[13] V. Indelman, S. Williams, M. Kaess, and F. Dellaert, "Information fusion in navigation systems via factor graph based incremental smoothing," *Elsevier RAS*, vol. 61, no. 8, pp. 721–738, 2013.

[14] J. Kaiser, A. Martinelli, F. Fontana, and D. Scaramuzza, "Simultaneous state initialization and gyroscope bias calibration in visual inertial aided navigation," *IEEE RAL*, vol. 2, no. 1, pp. 18–25, Jan. 2017.

[15] A. Martinelli, "Vision and imu data fusion: Closed-form solutions for attitude, speed, absolute scale and bias determination," *IEEE TRO*, vol. 28, no. 1, pp. 44–60, Feb. 2012.

[16] L. Kneip, S. Weiss, and R. Siegwart, "Deterministic initialization of metric state estimation filters for loosely-coupled monocular vision-inertial systems," in *Proc. IEEE Int. Conf. Intell. Robots Syst.*, 2011, pp. 2235–2241.

[17] S. Leutenegger, S. Lynen, M. Bosse, R. Siegwart, and P. Furgale, "Keyframe-based visual-inertial odometry using nonlinear optimization," *Sage IJRR*, vol. 34, no. 3, pp. 314–334, 2015.

[18] M. Li and A. I. Mourikis, "Improving the accuracy of ekf-based visual-inertial odometry," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2012, pp. 828–835.

[19] Y. Ling *et al.*, "Modeling varying camera-imu time offset in optimization-based visual-inertial odometry," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 484–500.

[20] M. L. A. Lourakis and A. A. Argyros, "Is levenberg-marquardt the most efficient optimization algorithm for implementing bundle adjustment?" in *Proc. IEEE Int. Conf. Comput. Vis.*, vol. 2, 2005, pp. 1526–1531.

[21] T. Lupton and S. Sukkarieh, "Visual-inertial-aided navigation for high-dynamic motion in built environments without initial conditions," *IEEE TRO*, vol. 28, no. 1, Feb. 2012.

[22] A. Martinelli, "Closed-form solution of visual-inertial structure from motion," *Int. J. Comput. Vision*, vol. 106, no. 2, pp. 138–152, Jan. 2014.

[23] A. I. Mourikis and S. I. Roumeliotis, "A multi-state constraint kalman filter for vision-aided inertial navigation," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2007, pp. 3565–3572.

[24] R. Mur-Artal, J. Montiel, and J. D. Tardós, "ORB-SLAM: A versatile and accurate monocular slam system," *IEEE TRO*, vol. 31, no. 5, pp. 1147–1163, Oct. 2015.

[25] R. Mur-Artal and J. D. Tardós, "Visual-inertial monocular slam with map reuse," *IEEE Robot. Automat. Lett.*, vol. 2, no. 2, pp. 796–803, Apr. 2017.

[26] D. Nistér, O. Naroditsky, and J. R. Bergen, "Visual odometry," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2005, doi: 10.1109/CVPR.2004.1315094.

[27] T. Qin and S. Shen, "Robust initialization of monocular visual-inertial estimation on aerial robots," in *Proc. IEEE Int. Conf. Intell. Robots Syst.*, 2017, pp. 4225–4232.

[28] E. Rosten, R. Porter, and T. Drummond, "FASTER and better: A. machine learning approach to corner detection," in *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 1, pp. 105–119, Jan. 2010.

[29] P. Furgale, T. D. Barfoot, and G. Sibley, "Continuous-time batch estimation using temporal basis functions," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2012, pp. 2088–2095.

[30] A. Patron-Perez, S. Lovegrove, and G. Sibley, "A spline-based trajectory representation for sensor fusion and rolling shutter cameras," *Int. J. Comput. Vision*, vol. 113, pp. 208–219, 2015.

[31] H. Ovrén and P.-E. Forssén, "Spline error weighting for robust visual-inertial fusion," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 321–329.

[32] P. Sturm, S. Ramalingam, and S. K. Lodha, "On calibration, structure from motion and multi-view geometry for generic camera models," in *Imaging Beyond the Pinhole Camera (Computational Imaging And Vision Ser. 33)*, vol. 33, K. Daniilidis and R. Klette, Eds. Dordrecht, Netherlands: Springer, 2006, pp. 87–105.