# Rolling-Shutter Modelling for Direct Visual-Inertial Odometry

David Schubert[1,2], Nikolaus Demmel[1], Lukas von Stumberg[1,2], Vladyslav Usenko[1] and Daniel Cremers[1,2]

***Abstract*— We present a direct visual-inertial odometry (VIO) method which estimates the motion of the sensor setup and sparse 3D geometry of the environment based on measurements from a rolling-shutter camera and an inertial measurement unit (IMU).**

**The visual part of the system performs a photometric bundle adjustment on a sparse set of points. This direct approach does not extract feature points and is able to track not only corners, but any pixels with sufficient gradient magnitude. Neglecting rolling-shutter effects in the visual part severely degrades accuracy and robustness of the system. In this paper, we incorporate a rolling-shutter model into the photometric bundle adjustment that estimates a set of recent keyframe poses and the inverse depth of a sparse set of points.**

**IMU information is accumulated between several frames using measurement preintegration, and is inserted into the optimization as an additional constraint between selected keyframes. For every keyframe we estimate not only the pose but also velocity and biases to correct the IMU measurements. Unlike systems with global-shutter cameras, we use both IMU measurements and rolling-shutter effects of the camera to estimate velocity and biases for every state.**

**Last, we evaluate our system on a new dataset that contains global-shutter and rolling-shutter images, IMU data and ground-truth poses for ten different sequences, which we make publicly available. Evaluation shows that the proposed method outperforms a system where rolling shutter is not modelled and achieves similar accuracy to the global-shutter method on global-shutter data.**

## I. INTRODUCTION

Many robotics applications rely on motion estimation and 3D reconstruction. Laser rangefinders, RGB-D cameras [1], GPS and many other sensors can be used to solve these tasks, but cameras are the most popular choice for such applications, because they are cheap, lightweight and small. They are passive sensors, so they do not interfere with each other when placed in the same environment. Several works have shown the application of cameras for robot navigation [2], [3] and autonomous driving [4].

Most visual odometry methods assume that cameras have a global shutter, and with this assumption show impressive results in 3D reconstruction and motion estimation (e.g. [5], [6]). A global-shutter camera exposes all pixels in the image simultaneously. However, rolling-shutter CMOS sensors are widespread in consumer devices (e.g. tablets, smartphones), but also in the automotive sector and in the motion picture industry. A rolling-shutter camera exposes rows sequentially with some delay and reads them one by one. This leads to large image distortions in the presence of fast motion, as can be seen in Fig. 2. Neglecting rolling-shutter effects leads to significant drift in the estimated trajectory and inaccurate 3D reconstruction [7], but when these effects are modelled correctly the system can achieve accuracy similar to global-shutter systems (Fig. 1).

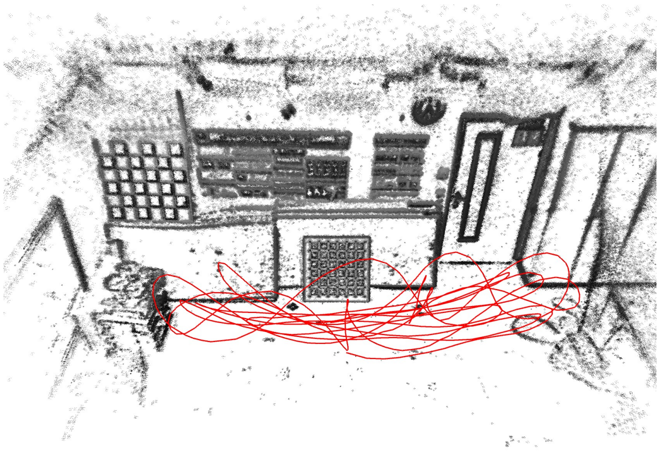

Fig. 1. Reconstructed camera trajectory (red) and sparse 3D reconstruction of our method on sequence 6 of our new dataset. Despite diverse motion patterns that revisit different parts of the scene multiple times, edges in the sparse point cloud stay very consistent and show little drift.

There exist two major types of approaches for visual odometry. Indirect methods (e.g. [6]) align pixel coordinates of the matched keypoints, whereas direct methods (e.g. [5]) align image intensities based on the photoconsistency assumption. Direct methods outperform indirect methods in weakly textured environments, but they are more sensitive to geometric noise [5]. Rolling-shutter effects introduce strong geometric changes and thus, for direct methods, it is much more important to model rolling shutter to achieve good results than for indirect methods. Unlike indirect methods, with direct methods the capture time of a point in its target frame is not directly known after selecting the point in its host frame, so the *rolling-shutter constraint* [8] has to be imposed in order to retrieve the capture time.

Another challenge for visual odometry methods is the lack of robustness in areas with low number of features, or when performing fast maneuvers. In the case of monocular cameras they are also not able to reconstruct the scale of the environment. By combining a camera with an inertial measurement unit (IMU) we can make the system more robust and, given sufficient excitation, estimate the metric scale of the environment. IMU measurements do not suffer from outliers and with corrected bias provide accurate short-term motion prediction.

[1] The authors are with the Computer Vision Group, Technical University of Munich, Germany, {schubdav, demmeln, stumberg, usenko, cremers}@in.tum.de

[2] The authors are with Artisense Corporation

In this paper, we propose a novel direct visual-inertial odometry method for rolling-shutter cameras. Our approach estimates pose, linear velocity and biases for each keyframe and the inverse depth of the points that are tracked by the system. To model the continuous motion of the camera we also optimize a twist in the camera frame that is coupled to the IMU velocity and biases, and use a constant-twist motion assumption to represent the continuous trajectory. This way we can incorporate rolling-shutter effects into the optimization.

We evaluate our method on ten challenging sequences from a newly recorded dataset that we make publicly available. The dataset features not only IMU data and rolling-shutter images, but also simultaneously recorded global-shutter images for comparison. To our knowledge, such a dataset currently does not exist. We compare our method to a state-of-the-art approach for global-shutter cameras, demonstrating that systems that model rolling shutter can achieve similar performance to global-shutter VIO methods running on global-shutter data.

In summary, our contributions are:

- a tight integration of rolling-shutter visual and inertial information in a direct odometry system,
- velocity and bias estimation not only from the IMU measurements, as in other methods for global-shutter visual-intertial odometry, but also from rolling-shutter effects of the images,
- a dataset that contains sequences simultaneously captured with global-shutter and rolling-shutter cameras that are time-aligned with IMU and motion capture data,
- an extensive evaluation of the proposed system on the collected dataset and comparison to the baseline global-shutter method.

The dataset and additional information about the system are available on:

https://vision.in.tum.de/data/datasets/rolling-shutter-dataset

## II. RELATED WORK

Visual-inertial odometry (VIO) can be grouped into two major approaches. Filtering-based methods keep a probabilistic representation of the state of the system, that includes a mean and a covariance matrix to capture correlations between variables. One example is ROVIO [9], [10], that uses an Extended Kalman Filter (EKF) and photometric residuals by comparing image patches, which are tightly coupled. An important extension of the EKF is the Multi-State Constraint Kalman Filter (MSCKF) [11], [12], that includes constraints from observing features in multiple images and does not require feature positions in the state vector, which yields a computational complexity linear in the number of features. This has already led to a variant for the rolling-shutter case [13].

On the other hand, optimization-based approaches have gained popularity. They set up an energy function that incorporates models of the sensors and perform a non-linear optimization to estimate parameters. Information from older frames can be kept in the system using marginalization. This approach has been successfully demonstrated with OKVIS [14], [15]. Direct examples of optimization-based VIO are given in [16], [17]. The latter is based on DSO [5], a state-of-the-art monocular visual odometry system. In its optimization backend, a global bundle adjustment is performed on a set of recent keyframes and a sparse set of points. In [17], the method is extended with a tightly coupled IMU integration and a method to tackle delayed scale observability in the presence of marginalization priors, which is a problem when the scale moves too far from its linearization point. Their strategy is to keep two marginalization priors with different linearization points and switch to the newer one when needed. A full visual-inertial SLAM system is given by VINS-Mono [18], which is also based on non-linear optimization. A visual-inertial extension of ORB-SLAM [19], [6], a state-of-the-art keyframe-based SLAM approach, is given in [20]. Another method [21] proposes a B-spline representation of the trajectory to incorporate rolling shutter and measurements of different sensors.

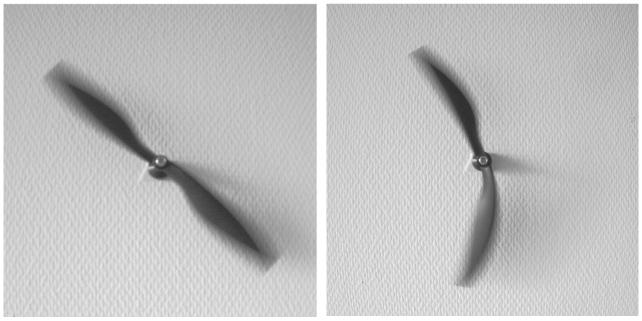

Fig. 2. Difference between global-shutter (left) and rolling-shutter (right) images when exposed to fast motion. Both images were triggered at the same time. The rotating propeller appears distorted when a rolling-shutter sensor is used.

Modelling rolling shutter in the domain of direct odometry methods has been attempted with different sensor modalities. The RGB-D method in [1] uses not only a photometric error term, but also a geometric error term due to the availability of depth measurements. Direct monocular approaches have been presented in [22], [7]. While the first decouples motion and structure estimation and enforces the rolling-shutter constraint softly by introducing additional time variables, the latter performs a global bundle adjustment and explicitly solves the rolling-shutter constraint.

Contrary to the other methods, we present a direct visual-inertial rolling-shutter odometry method, which combines existing rolling-shutter [7] and visual-inertial [17] approaches and couples the underlying variables with a new energy term.

## III. NOTATION

In this paper, poses from the special Euclidean group SE(3) are represented as $4 \times 4$ matrices

$$\mathbf{T} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0}_{1\times 3} & 1 \end{bmatrix}, \tag{1}$$

where $\mathbf{R} \in \mathrm{SO}(3)$ is a rotation matrix from the special orthogonal group, and $\mathbf{t} \in \mathbb{R}^3$ is a three-dimensional translation component.

In order to optimize scale and gravity direction, we will also use transformations from $\mathbb{R}^+ \times \mathrm{SO}(3)$, which include a positive scale $s \in \mathbb{R}^+$ and a rotation matrix $\mathbf{R} \in \mathrm{SO}(3)$ and act as the matrix

$$\mathbf{T} = \begin{bmatrix} s\mathbf{R} & \mathbf{0}_{3\times1} \\ \mathbf{0}_{1\times3} & 1 \end{bmatrix}, \tag{2}$$

so they can also be seen as a similarity transform with zero translation component.

We will also make use of the exponential map,

$$\exp\colon \mathfrak{se}(3) \to \mathrm{SE}(3)\,, \tag{3}$$

to map elements from the Lie algebra $\mathfrak{se}(3)$ to the Lie group $\mathrm{SE}(3)$, which, in matrix representation, is just the matrix exponential (but has a closed form in this particular case). Lie algebra elements $\hat{\boldsymbol{\xi}}$ are generated from vectors $\boldsymbol{\xi} \in \mathbb{R}^6$ using the hat operator. We use the convention that the first three components of $\boldsymbol{\xi}$ correspond to translation and the last three components correspond to rotation. Using the exponential map, it is possible to parametrize poses as a function of time as

$$\mathbf{T}(t) = \exp(\hat{\boldsymbol{\xi}}t)\mathbf{T}_0\,, \tag{4}$$

starting from pose $\mathbf{T}_0 \in \mathrm{SE}(3)$ at $t = 0$. This is called a constant-twist interpolation. For brevity, we will not use the hat operator inside the exponential function. Whenever the exponential function acts on a vector, it denotes a composition of the hat operator and the exponential. Similarly, we will call 6D vectors twists.

When representing 3D points in different coordinate systems $A$ and $B$, the pose that converts a point's representation $\mathbf{p}_A$ in system $A$ to its representation $\mathbf{p}_B$ in system $B$ is written $\mathbf{T}_{BA}$, and the transformed point is calculated as

$$\mathbf{p}_B = \mathbf{T}_{BA}\mathbf{p}_A\,, \tag{5}$$

where 3D points $(x, y, z)^\top$ are represented as $\mathbf{p} = (x, y, z, 1)^\top$.

The coordinate systems we use in this paper are

| | |
|---|---|
| $\mathrm{W_m}$ | metric world, |
| $\mathrm{W_f}$ | world with freely chosen scale, |
| $\mathrm{C_m}$ | metric camera, |
| $\mathrm{C_f}$ | camera with freely chosen scale, |
| $\mathrm{I}$ | IMU (metric). |

An overview of these systems including the transformations between them is also given in Fig. 3. The reason why world and camera systems exist twice is that scale and gravity direction might not be observable from the beginning, hence the visual system starts estimating camera poses $\mathbf{T}_{\mathrm{C_f W_f}}$ with a freely chosen scale, which can later be converted to a metric scale using the to-be-optimized variable $\mathbf{T}_{\mathrm{W_m W_f}}$, which includes scale and rotation for gravity alignment. The transformation from non-metric to metric camera $\mathbf{T}_{\mathrm{C_m C_f}}$ does not contain any additional variables, as it uses the scale of $\mathbf{T}_{\mathrm{W_m W_f}}$, but identity rotation. Note also that IMU-to-world poses do not act as additional optimization variables, but are calculated from world-to-camera poses, which are optimized.

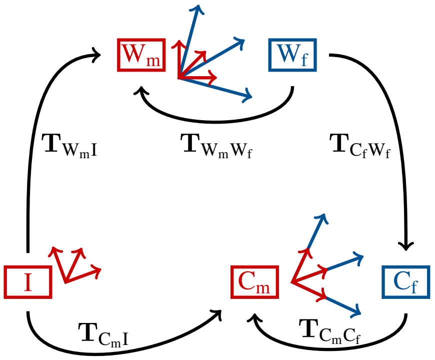
$\mathrm{W_m}$
$\mathrm{W_f}$
$\mathbf{T}_{\mathrm{W_m I}}$
$\mathbf{T}_{\mathrm{W_m W_f}}$
$\mathbf{T}_{\mathrm{C_f W_f}}$
$\mathrm{I}$
$\mathrm{C_m}$
$\mathrm{C_f}$
$\mathbf{T}_{\mathrm{C_m I}}$
$\mathbf{T}_{\mathrm{C_m C_f}}$

Fig. 3. Coordinate systems and transformations used in this paper. The coordinate system abbreviations are: I: IMU, W: world, C: camera. The subscript m denotes that a coordinate system has a metric scale, while the subscript f denotes that the coordinate system has a freely chosen scale. In this illustration also the colours indicate whether units are metric (red) or not (blue). The algorithm optimizes world-to-camera poses $\mathbf{T}_{\mathrm{C_f W_f}}$ which are directly used for the photometric energy. IMU factors use IMU-to-world poses $\mathbf{T}_{\mathrm{W_m I}}$. The transformation $\mathbf{T}_{\mathrm{W_m W_f}}$ between the metric and the non-metric world is another optimization variable, which does not only include scale, but also a rotation for gravity alignment. $\mathbf{T}_{\mathrm{C_m C_f}}$ includes only scale, the same one as in $\mathbf{T}_{\mathrm{W_m I}}$. The IMU-to-camera transformation $\mathbf{T}_{\mathrm{C_m I}}$ is known from calibration.

## IV. MODEL

In this section, we detail our formulation of a visual-inertial, rolling-shutter-aware energy, including brief reviews of previous methods that our method builds upon. The model described here only applies to the optimization backend, which jointly optimizes depth and keyframe variables, while the visual-inertial frontend that provides initializations operates as in [17] and assumes global-shutter data. Hence, the words frame and keyframe are used interchangeably. As shown in Fig. 3, there is a metric and a non-metric world. For the photometric energies, non-metric poses are used throughout. This has the advantage that tracking can be started right at the beginning of a sequence, while scale and gravity direction are optimized using the IMU information, instead of having to wait until scale is observable. The IMU factors, on the other hand, are calculated using metric poses.

### *A. Photometric Energy*

To model the world-to-camera pose of frame $i$, i.e. the pose that converts a point in the world frame to the corresponding point in the camera frame, a constant twist model as in [7] is used to parametrize the pose as a function of time,

$$\mathbf{T}_i(t) = \exp(\boldsymbol{\xi}_i t)\mathbf{T}_i^0\,, \tag{6}$$

where $\boldsymbol{\xi}_i \in \mathbb{R}^6$ is the twist for frame $i$ and $\mathbf{T}_i^0$ the central pose, corresponding to $t = 0$. The algorithm is operating on pre-undistorted images, hence the capture time of a pixel at coordinates $(x, y)$ in the undistorted image is given by

$$t(x, y) = f_{\mathrm{d}}(x, y) - y_0\,. \tag{7}$$

The distortion function $f_{\text{d}}$ is known from camera calibration. It maps the point into the original, distorted image, where only the $y$-coordinate is relevant for the capture time, as this is the readout direction of the rolling-shutter sensor. The offset $y_0$ is the vertical middle of the distorted image. We are free to measure time in pixel units, only later when the twist is compared to the IMU variables and measurements, the correct time conversion factor has to be chosen.

As in [5], the photometric energy is a triple sum over the current set of keyframes $\mathcal{F}$, the points $\mathcal{P}_i$ hosted in a specific keyframe $i$ and the set of keyframes $\text{obs}(\mathbf{p})$ in which a specific point $\mathbf{p}$ is observed,

$$E_{\text{ph}} = \sum_{i\in\mathcal{F}} \sum_{\mathbf{p}\in\mathcal{P}_i} \sum_{j\in\text{obs}(\mathbf{p})} E_{\mathbf{p}j}\,. \tag{8}$$

The energy contribution $E_{\mathbf{p}j}$ by the observation of point $\mathbf{p}$ in frame $j$ is obtained by comparing intensities at the point's location in the host frame and its location when projected into the target frame. The projection into the target frame is not straightforward, as the pose of the target frame is needed, but the pose of the target frame depends on time, i.e. the pixel coordinate in the target frame, which in turn depends on the pose. To solve this mutual dependency, the rolling-shutter constraint [8] is solved iteratively as in [7].

### *B. Visual-Inertial Factors*

We use IMU factors with preintegrated measurements as implemented in GTSAM[1] (based on [23], [24], [25]), as they have been used in [17], [16]. Note that poses in this subsection are in the metric world, so in practice they have to be calculated from the non-metric camera poses using the current estimate for $\mathbf{T}_{\text{W}_\text{m}\text{W}_\text{f}}$. The state of frame $i$ consists of the pose (rotation $\mathbf{R}_i$ and translation $\mathbf{p}_i$), a translational velocity $\mathbf{v}_i$ of the IMU in the metric world frame and a bias vector $\mathbf{b}_i \in \mathbb{R}^6$,

$$\mathbf{s}_i = [\mathbf{R}_i, \mathbf{p}_i, \mathbf{v}_i, \mathbf{b}_i]\;. \tag{9}$$

From keyframe $i$ to the next keyframe $j$, measurements are integrated to obtain a prediction for the state of keyframe $j$. Starting with $\Delta\mathbf{p} = \mathbf{0}$, $\Delta\mathbf{v} = \mathbf{0}$ and identity rotation $\Delta\mathbf{R} = \mathbf{I}$, those quantities are iteratively updated as

$$\Delta\mathbf{p} \leftarrow \Delta\mathbf{p} + \Delta\mathbf{v}\Delta t\,, \tag{10}$$

$$\Delta\mathbf{v} \leftarrow \Delta\mathbf{v} + \Delta\mathbf{R}(\mathbf{a} - \mathbf{b}_i^{\text{a}})\Delta t\,, \tag{11}$$

$$\Delta\mathbf{R} \leftarrow \exp((\boldsymbol{\omega} - \boldsymbol{b}_i^{\text{g}})\Delta t)\,. \tag{12}$$

Here, $\Delta t$ is the time difference between two IMU measurements, $\mathbf{a}$ the current accelerometer measurement, $\boldsymbol{\omega}$ the current gyroscope measurement, $\mathbf{b}_i^{\text{a}}$ the three accelerometer components of the bias $\mathbf{b}_i$ corresponding to frame $i$ and $\mathbf{b}_i^{\text{g}}$ the three gyroscope components. In this case, the exponential maps 3D rotational velocities to rotation matrices in $\mathrm{SO}(3)$.

Integrating all measurements between keyframe $i$ and keyframe $j$ as in Eqs. 10-12 yields the preintegrated measurements $\Delta\mathbf{R}_{ij}$, $\Delta\mathbf{v}_{ij}$ and $\Delta\mathbf{p}_{ij}$. To save computation time, the preintegration is not redone once the bias changes. Instead, the preintegrated measurements are linearized as functions of the bias. These linearized functions will be denoted $\Delta\mathbf{R}_{ij}(\mathbf{b}_i^{\text{g}})$, $\Delta\mathbf{v}_{ij}(\mathbf{b}_i^{\text{g}}, \mathbf{b}_i^{\text{a}})$ and $\Delta\mathbf{p}_{ij}(\mathbf{b}_i^{\text{g}}, \mathbf{b}_i^{\text{a}})$ and are calculated as detailed in [23], Eq. 44.

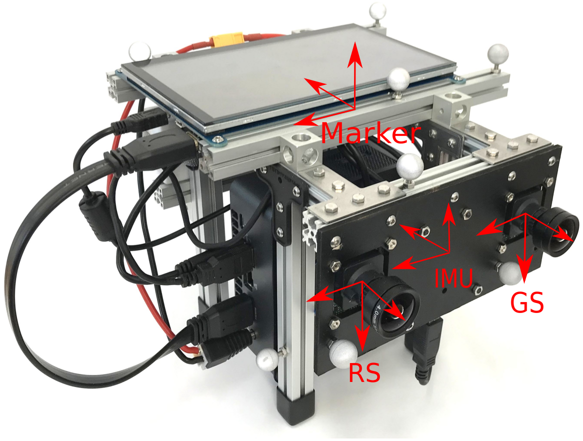

Fig. 4. Camera setup that was used to acquire our new dataset. One camera is set to global-shutter mode and the other camera is set to rolling-shutter mode. Both cameras are hardware-synchronized with the IMU, and the transformations between all frames are pre-calibrated. Ground-truth data is recorded using a motion capture system. Time alignment for all sequences is done by aligning rotational velocities computed from the motion capture system and the gyroscope data.

The state predictions for frame $j$ are then calculated as

$$\hat{\mathbf{R}}_j = \mathbf{R}_i \Delta\mathbf{R}_{ij}(\mathbf{b}_i^{\text{g}})\,, \tag{13}$$

$$\hat{\mathbf{p}}_j = \mathbf{p}_i + (t_j - t_i)\mathbf{v}_i + \frac{1}{2}(t_j - t_i)^2\mathbf{g} + \mathbf{R}_i\Delta\mathbf{p}_{ij}(\mathbf{b}_i^{\text{g}}, \mathbf{b}_i^{\text{a}})\,, \tag{14}$$

$$\hat{\mathbf{v}}_j = \mathbf{v}_i + (t_j - t_i)\mathbf{g} + \mathbf{R}_i\Delta\mathbf{v}_{ij}(\mathbf{b}_i^{\text{g}}, \mathbf{b}_i^{\text{a}})\,, \tag{15}$$

where $\mathbf{g}$ is the gravity vector and $t_i$ and $t_j$ are the timestamps of frames $i$ and $j$.

The residuals are then calculated as

$$\mathbf{r}_{\Delta\mathbf{R}_{ij}} = \log\left(\hat{\mathbf{R}}_j^\top \mathbf{R}_j\right)\,, \tag{16}$$

$$\mathbf{r}_{\Delta\mathbf{v}_{ij}} = \mathbf{R}_i^\top(\mathbf{v}_j - \hat{\mathbf{v}}_j)\,, \tag{17}$$

$$\mathbf{r}_{\Delta\mathbf{p}_{ij}} = \mathbf{R}_i^\top(\mathbf{p}_j - \hat{\mathbf{p}}_j)\,, \tag{18}$$

$$\mathbf{r}_{\mathbf{b}_{ij}} = \mathbf{b}_j - \mathbf{b}_i\,. \tag{19}$$

which are stacked in a residual vector $\mathbf{r}_{ij}$. These residuals then lead to the energy contribution

$$E_{ij} = \mathbf{r}_{ij}^\top \boldsymbol{\Sigma} \mathbf{r}_{ij}\,, \tag{20}$$

with appropriate covariances $\boldsymbol{\Sigma}$ as derived in [23]. Summing these energies over the set of pairs of consecutive frames $\mathcal{C}$ yields the energy

$$E_{\text{IMU}} = \sum_{(i,j)\in\mathcal{C}} E_{ij}\,. \tag{21}$$

[1] https://bitbucket.org/gtborg/gtsam

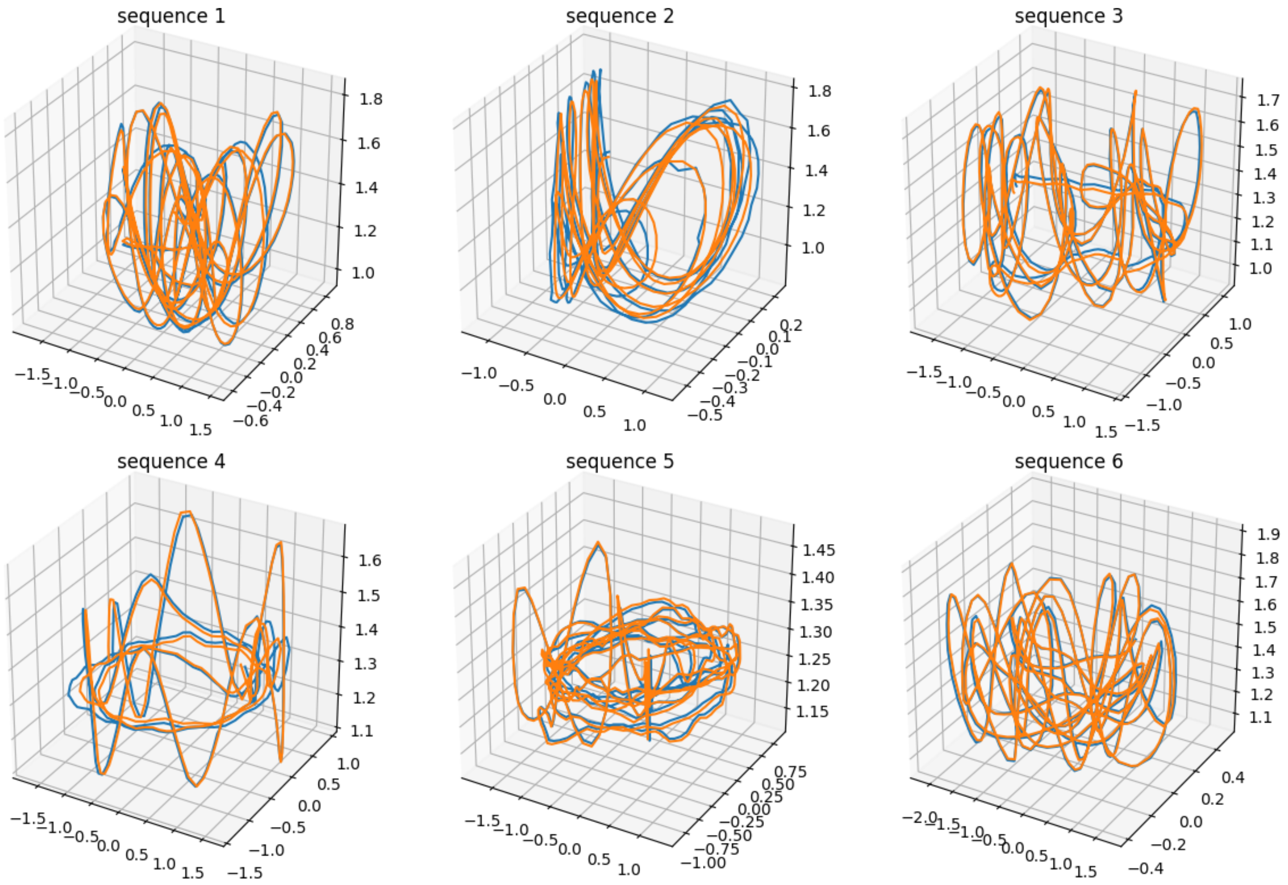

Fig. 5. Trajectory plots for the first 6 sequences after SE(3) alignment of reconstructed trajectories (blue) and ground truth (orange), axes in meters. To give representative examples, a run with error $e_{\text{ate}}$ close to the median error is shown for each sequence.

### C. *Twist Energy*

In [7], the system is stabilized by a prior for the twist $\boldsymbol{\xi}_i$ introduced in Eq. 6, that assumes a smooth motion between keyframes. Using an IMU, we are in a much more comfortable situation, as it provides high-frequency measurements and hence a much more up-to-date prior for the twist, leading to the novel formulation for the twist energy that we propose. From the IMU, we directly obtain a gyroscope measurement $\boldsymbol{\omega}$, which is biased by $\mathbf{b}^{\mathrm{g}}$, and the state includes the translational velocity $\mathbf{v}$ as an optimization variable. For the ease of notation, we drop the keyframe index $i$, but still all variables belong to a certain keyframe, in particular to the timestamp of its mid-pose $\mathbf{T}_i^0$.

The velocity $\mathbf{v}$ is the velocity of the IMU in the metric world frame $\mathrm{W_m}$. We first rotate this velocity into the IMU frame,

$$\begin{aligned} \mathbf{v}^{\mathrm{IMU}} &= \mathbf{R}_{\mathrm{IW_m}}\mathbf{v} && (22)\\ &= \mathbf{R}_{\mathrm{C_mI}}^{-1}\mathbf{R}_{\mathrm{C_mC_f}}\mathbf{R}_{\mathrm{C_fW_f}}\mathbf{R}_{\mathrm{W_mW_f}}^{-1}\mathbf{v}\,, && (23) \end{aligned}$$

where $\mathbf{R}_{BA}$ is the rotation of $\mathbf{T}_{BA}$, thus trivially $\mathbf{R}_{\mathrm{C_mC_f}} = \mathbf{I}$. We cannot use $\mathbf{T}_{\mathrm{W_mI}}^{-1}$ directly to obtain $\mathbf{R}_{\mathrm{IW_m}}$, as $\mathbf{T}_{\mathrm{W_mI}}$ is not an optimization variable. It needs to be expressed as a function of the non-metric camera poses $\mathbf{T}_{\mathrm{C_fW_f}}$.

Now we have the required quantities for the IMU twist,

$$\boldsymbol{\xi}^{\mathrm{IMU}} = \begin{bmatrix} \mathbf{v}^{\mathrm{IMU}} \\ \boldsymbol{\omega} - \mathbf{b}^{\mathrm{g}} \end{bmatrix}, \tag{24}$$

which could be used to approximate the motion of the IMU. We are, however, interested in the motion of the camera as given in Eq. 6, so the twist has to be converted using the adjoint $\operatorname{Adj}(\mathbf{T}_{\mathrm{C_mI}})$ of the relative pose between camera und IMU,

$$\boldsymbol{\xi}^{\mathrm{cam}} = -\operatorname{Adj}(\mathbf{T}_{\mathrm{C_mI}})\boldsymbol{\xi}^{\mathrm{IMU}}\,. \tag{25}$$

The adjoint is a $6 \times 6$ matrix and has the property

$$\mathbf{T}\exp(\boldsymbol{\delta}) = \exp\left(\operatorname{Adj}(\mathbf{T})\boldsymbol{\delta}\right)\mathbf{T}\,. \tag{26}$$

Thus, we can show that acting with $\boldsymbol{\xi}^{\mathrm{IMU}}$ on the IMU pose has the same effect as acting with $\boldsymbol{\xi}^{\mathrm{cam}}$ on the camera pose:

$$\begin{aligned} &\mathbf{T}_{\mathrm{W_mI}}\exp\left(\boldsymbol{\xi}^{\mathrm{IMU}}t\right) && (27)\\ &= \mathbf{T}_{\mathrm{C_mW_m}}^{-1}\mathbf{T}_{\mathrm{C_mI}}\exp(\boldsymbol{\xi}^{\mathrm{IMU}}t) && (28)\\ &= \mathbf{T}_{\mathrm{C_mW_m}}^{-1}\exp\left(\operatorname{Adj}(\mathbf{T}_{\mathrm{C_mI}})\boldsymbol{\xi}^{\mathrm{IMU}}t\right)\mathbf{T}_{\mathrm{C_mI}} && (29)\\ &= \left(\exp\left(-\operatorname{Adj}(\mathbf{T}_{\mathrm{C_mI}})\boldsymbol{\xi}^{\mathrm{IMU}}t\right)\mathbf{T}_{\mathrm{C_mW_m}}\right)^{-1}\mathbf{T}_{\mathrm{C_mI}} && (30)\\ &= \left(\exp\left(\boldsymbol{\xi}^{\mathrm{cam}}t\right)\mathbf{T}_{\mathrm{C_mW_m}}\right)^{-1}\mathbf{T}_{\mathrm{C_mI}}\,. && (31) \end{aligned}$$

So far the translational components are in metric units. To obtain an appropriate prior for the twist in Eq. 6, the translational components $\boldsymbol{\xi}_{\mathrm{t}}^{\mathrm{cam}}$ have to be divided by the scale $s$ of the transformation $\mathbf{T}_{\mathrm{W_m W_f}}$, while the rotational components $\boldsymbol{\xi}_{\mathrm{r}}^{\mathrm{cam}}$ stay unaffected. Hence, the prior for the twist is

$$\tilde{\boldsymbol{\xi}} = t_{\mathrm{d}} \begin{bmatrix} \boldsymbol{\xi}_{\mathrm{t}}^{\mathrm{cam}}/s \\ \boldsymbol{\xi}_{\mathrm{r}}^{\mathrm{cam}} \end{bmatrix} , \tag{32}$$

which is also scaled by the time difference $t_{\mathrm{d}}$ between two consecutive image rows, as this is the unit of time chosen in Eq. 7. Finally, the energy contribution is a weighted squared deviation, summed over all frames:

$$E_{\text{twist}} = \sum_{\mathcal{F}} (\tilde{\boldsymbol{\xi}} - \boldsymbol{\xi})^{\top} \boldsymbol{\Sigma} (\tilde{\boldsymbol{\xi}} - \boldsymbol{\xi}) , \tag{33}$$

with a manually chosen diagonal matrix $\boldsymbol{\Sigma}$.

### D. Optimization

We optimize the total energy

$$E = E_{\text{ph}} + \alpha E_{\text{IMU}} + \beta E_{\text{twist}} \tag{34}$$

using Gauss-Newton optimization. The parameters $\alpha$ and $\beta$ are balancing weights. As in [5], keyframes are marginalized once the set of keyframes grows too large. Once a variable is part of the marginalization term, its linearization point is not changed any more to keep the system consistent. This is usually a good approximation, as the state estimates do not fluctuate heavily, but for scale, this is not the case. We therefore use the approach of [17] and keep a second version of the marginalization Hessian that only contains newer IMU factors, which can be used once the scale estimate moves too far from the linearization point of the current marginalization Hessian. As the newly introduced twist energy depends on scale, we also include it with two different linearization points in the two versions of the marginalization Hessian.

## V. NEW ROLLING-SHUTTER DATASET

Since we want to compare our novel rolling-shutter VIO method to the baseline global-shutter method not only on rolling-shutter images, but also on global-shutter images, we need a dataset that provides both simultaneously. To the best of our knowledge, there is no such dataset suitable for VIO evaluation. Therefore, we recorded our own dataset and make it publicly available. This dataset comprises 10 challenging indoor sequences spanning a total of around 7 min and 475 m traversed distance. Tab. I shows some statistics of the individual sequences, where the mean velocities and accelerations are computed as numerical derivatives of the ground-truth IMU poses.

The sensor setup as depicted in Fig. 4 includes two uEye UI-3241LE-M-GL cameras by IDS with Lensagon BM4018S118 lenses by Lensation. The cameras record time-synchronized images at 20 Hz with the left camera running in global-shutter mode and the right camera in rolling-shutter mode and a row time difference of approximately 29.47 µs. The 1280x1024 grayscale images are captured with a linear response function at 16 bit to retain the full dynamic range of the imager, and in our direct VIO approach we additionally use pre-calibrated vignette compensation as in [5]. The Bosch BMI160 accelerometer and gyroscope readings at 200 Hz are time-synchronized with the cameras in hardware. Ground-truth motion is recorded with an OptiTrack Flex13 motion capture system that uses ceiling-mounted cameras to track IR-reflective markers mounted on the sensor setup at 120 Hz. A simple median filter discards outlier motion capture poses and linear interpolation is used to compute reference poses at keyframe times.

TABLE I

SEQUENCES IN OUR ROLLING-SHUTTER DATASET

| Seq. | Duration | Length | Mean Velocity | | Mean Acceleration | |
|---|---|---|---|---|---|---|
| | [s] | [m] | [m/s] | [°/s] | [m/s$^2$] | [°/s$^2$] |
| 1 | 40 | 46 | 1.1 | 62 | 3.2 | 233 |
| 2 | 27 | 37 | 1.4 | 73 | 4.1 | 271 |
| 3 | 50 | 44 | 0.9 | 56 | 2.3 | 220 |
| 4 | 38 | 30 | 0.8 | 39 | 1.1 | 148 |
| 5 | 85 | 57 | 0.7 | 43 | 0.8 | 167 |
| 6 | 43 | 51 | 1.2 | 50 | 2.6 | 252 |
| 7 | 39 | 45 | 1.1 | 92 | 2.7 | 148 |
| 8 | 53 | 46 | 0.9 | 91 | 1.8 | 103 |
| 9 | 45 | 46 | 1.0 | 137 | 3.8 | 539 |
| 10 | 54 | 41 | 0.7 | 116 | 2.1 | 605 |
| Total | 475 | 442 | 0.9 | 75 | 2.2 | 267 |

Similar to [26] we calibrate camera and IMU intrinsics, as well as extrinsics between both cameras, the IMU and the motion capture markers. In all calibration sequences both cameras are in global-shutter mode to ensure accurate results. Since the motion capture poses are not time-synchronized during recording, we perform temporal alignment for each evaluation sequence by aligning gyroscope measurements to angular velocities estimated from motion capture.

With the dataset we provide our calibration results, preprocessed sequences with IMU intrinsics compensated (scale, axis-alignment, constant bias) and ground-truth poses geometrically and temporally aligned to the IMU frame. On top of that, raw data and calibration sequences are also available to facilitate custom calibration schemes.

## VI. QUANTITATIVE EVALUATION

We run extensive evaluations on our newly taken dataset. The baseline method is a global-shutter method as in [17]. It also integrates visual and inertial measurements, but does not feature a rolling-shutter model. Both [17] and our new method are operated with 2000 points and a maximum of 7 keyframes. As the new dataset contains pairs of similar rolling-shutter and global-shutter sequences, not only can we compare the performance of the new method with the baseline method operating on rolling-shutter data, but also with the baseline method operating on global-shutter data.

Inertial methods can observe scale even with a monocular camera, provided a non-degenerate motion. In our case, the final scale estimate is used to convert the optimized non-metric poses to metric poses. Thus, to compare the estimated trajectories with the corresponding ground truth, no scale alignment has to be performed. The evaluation metric in this section is the absolute trajectory error after SE(3) alignment,

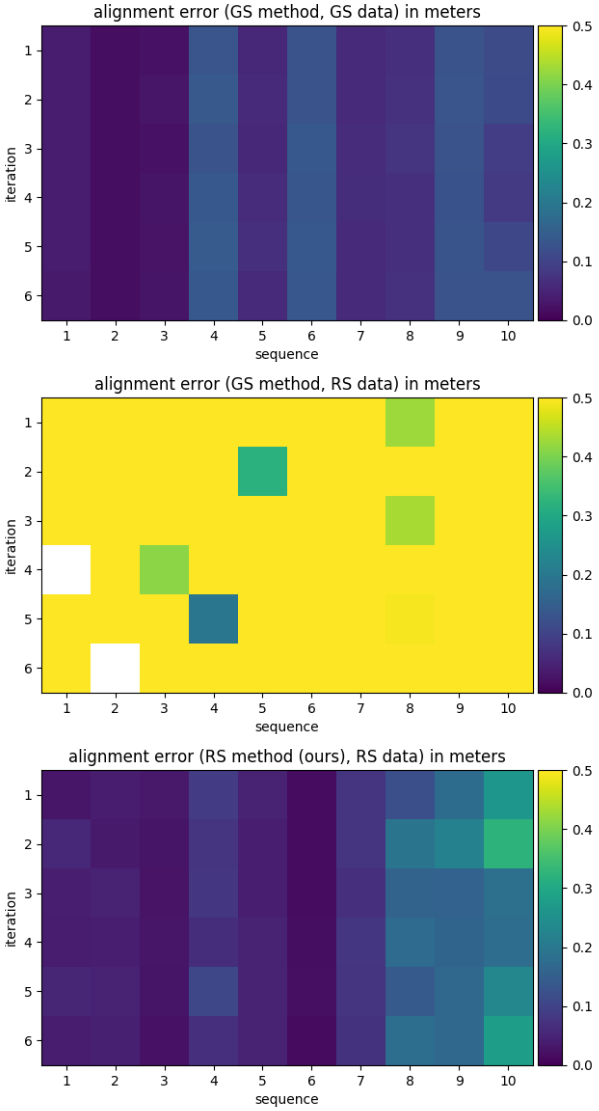

Fig. 6. Colour-coded absolute trajectory error $e_{\text{ate}}$ after $\mathrm{SE}(3)$ alignment. Each of the 10 sequences from our new dataset has been run 6 times in three different modes: the global-shutter baseline method operating on global-shutter images (top); the global-shutter baseline method operating on rolling-shutter images (middle); our new rolling-shutter method operating on rolling-shutter images. White squares correspond to runs that failed entirely due to numerical instability. The baseline method produces stable results on global-shutter data, but mostly fails on rolling-shutter data. With our new method, similar results as for the baseline method on global-shutter data can be achieved.

defined as

$$e_{\text{ate}} = \min_{\mathbf{T}\in\mathrm{SE}(3)} \sqrt{\frac{1}{n}\sum_i \|\mathbf{T}\mathbf{p}_i - \hat{\mathbf{p}}_i\|^2}\,, \tag{35}$$

where $i$ is iterated over all keyframes in the whole sequence, and $n$ is the number of such keyframes. The keyframe positions estimated by an algorithm are denoted $\mathbf{p}_i$ and the corresponding ground-truth positions are denoted $\hat{\mathbf{p}}_i$.

To gather statistics, each sequence of the dataset has been run 6 times for each mode, with different random seeds which influences point selection. The three modes are

- running the global-shutter baseline method on global-shutter images;
- running the global-shutter baseline method on rolling-shutter images;
- running our new rolling-shutter method on rolling-shutter images.

We do not compare with the similar purely visual method in [7], as it cannot observe scale, which makes a fair comparison difficult. In Fig. 6 the results of these experiments are shown in a colour plot. Each coloured square corresponds to one run and 6 squares in the same column correspond to the 6 runs of the respective sequence. The colour of the square encodes the absolute trajectory error $e_{\text{ate}}$. The results of the global-shutter method on global-shutter data show a stable and accurate performance. Running the same algorithm on rolling-shutter data drastically changes the results. Apart from five runs, all runs are beyond the colour scale. Individual inspection showed that this is not inaccuracy, but instability, as the tracking results diverged far from the ground truth. Two runs failed entirely, which means the system became numerically unstable. These results are interesting when compared to the results in [7] (a similar, purely visual method), where the global-shutter method was not entirely unstable, but often estimated trajectories with drift. Possibly, the rolling-shutter images can be approximately explained with a slightly altered trajectory, but if there is inertial data that contradicts this slightly altered trajectory, the system breaks.

Using the new rolling-shutter method redeems most of the problems with rolling-shutter data. Apart from sequence 10, it shows very stable performance with accuracies similar to the global-shutter method on global-shutter data.

A more quantitative comparison is given in Tab. II. For each of the three modes and for each of the 10 sequences, the median of the absolute trajectory error $e_{\text{ate}}$ over all 6 runs is given. The errors of the global-shutter method on rolling-shutter data are larger than for the other two modes by orders of magnitude in most cases. A comparison of the global-shutter method on global-shutter data with the rolling-shutter method on rolling-shutter data does not yield a clear preference for all sequences. There are more sequences with more accurate results for the global-shutter method on global-shutter data, but also some sequences where the rolling-shutter method on rolling-shutter data is more accurate. One possible explanation for the latter case is that due to the time shift between rows, rolling-shutter images provide additional information about velocities. On the other hand, one reason for less accurate results by the rolling-shutter method might be the constant-twist assumption, which is violated in the presence of large accelerations. This reasoning is supported by the fact that sequence 10, the sequence with the largest mean rotational acceleration, is the sequence with the least accurate results. Remedy for the constant-twist model would be brought by a shorter row time difference, as then the twist only has to be valid over a shorter time interval. The rolling shutter in our dataset is rather at the slower end of the scale, so the shutter may well be faster in other use cases.

A visual impression of the tracking result of our algorithm

TABLE II

ABSOLUTE TRAJECTORY ERROR AFTER $\mathrm{SE}(3)$ ALIGNMENT, MEDIAN OVER 6 RUNS, IN METERS. MINIMUM FOR EACH SEQUENCE IN BOLD.

| Seq. | GS method, GS data | GS method, RS data | Ours, RS data |
|---|---|---|---|
| 1 | **0.038** | 79.591 | 0.040 |
| 2 | **0.018** | 40.725 | 0.044 |
| 3 | **0.027** | 1.803 | 0.028 |
| 4 | 0.137 | 0.970 | **0.079** |
| 5 | 0.060 | 0.683 | **0.049** |
| 6 | 0.135 | 2.352 | **0.017** |
| 7 | **0.061** | 28.336 | 0.075 |
| 8 | **0.070** | 0.501 | 0.168 |
| 9 | **0.128** | 218.152 | 0.168 |
| 10 | **0.111** | 482.021 | 0.246 |

on the first 6 sequences is given in Fig. 5. It shows the estimated camera trajectory together with ground truth, after $\mathrm{SE}(3)$ alignment. For each sequence, a run with error $e_{\mathrm{ate}}$ close to the median error was selected. Qualitatively, the estimated trajectories have very similar shapes compared to the ground truth, with small deviations visible.

One significant drawback of our method is runtime. As our approach combines energy terms and variables of [7] and [17], it is slower than the sub-realtime performance reported in [7], so possible speedups remain an open challenge.

## VII. Conclusion

In this paper, we present a direct sparse visual-inertial odometry approach for rolling-shutter cameras. We estimate poses, linear velocities and biases for a set of keyframes. These variables are coupled to the twist used to represent the continuous motion of the camera that is needed to model rolling-shutter projection. This way, unlike global-shutter methods, we estimate velocities and biases from both IMU measurements and rolling-shutter effects.

Due to the lack of suitable datasets, we recorded a new dataset with global-shutter and rolling-shutter images, IMU data and ground-truth poses from motion capture and make it publicly available. Our evaluation on this dataset shows that we can achieve similar accuracy to conventional methods on global-shutter data by explicitly modelling and exploiting rolling-shutter effects in the visual-inertial odometry.

## References

[1] C. Kerl, J. Sturm, and D. Cremers, "Robust odometry estimation for RGB-D cameras," in *Proc. of the IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2013.

[2] S. Weiss, M. Achtelik, S. Lynen, M. Chli, and R. Siegwart, "Real-time onboard visual-inertial state estimation and self-calibration of MAVs in unknown environments," in *Proc. of the IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2012.

[3] A. Stelzer, H. Hirschmüller, and M. Görner, "Stereo-vision-based navigation of a six-legged walking robot in unknown rough terrain," *The International Journal of Robotics Research / Int. Journal of Robotics Research (IJRR)*, 2012.

[4] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? The KITTI vision benchmark suite," in *Proc. of the IEEE Int. Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[5] J. Engel, V. Koltun, and D. Cremers, "Direct sparse odometry," *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 3, pp. 611–625, 2018.

[6] R. Mur-Artal and J. D. Tardós, "ORB-SLAM2: An open-source SLAM system for monocular, stereo, and RGB-D cameras," *IEEE Transactions on Robotics*, vol. 33, no. 5, pp. 1255–1262, 2017.

[7] D. Schubert, N. Demmel, V. Usenko, J. Stückler, and D. Cremers, "Direct sparse odometry with rolling shutter," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 682–697.

[8] M. Meingast, C. Geyer, and S. Sastry, "Geometric models of rolling-shutter cameras," *arXiv preprint cs/0503076*, 2005.

[9] M. Bloesch, S. Omari, M. Hutter, and R. Siegwart, "Robust visual inertial odometry using a direct EKF-based approach," in *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robot Systems (IROS)*, 2015.

[10] M. Bloesch, M. Burri, S. Omari, M. Hutter, and R. Siegwart, "Iterated extended Kalman filter based visual-inertial odometry using direct photometric feedback," *The International Journal of Robotics Research*, vol. 36, no. 10, pp. 1053–1072, 2017.

[11] A. I. Mourikis and S. I. Roumeliotis, "A multi-state constraint Kalman filter for vision-aided inertial navigation," in *Proceedings 2007 IEEE International Conference on Robotics and Automation*. IEEE, 2007, pp. 3565–3572.

[12] M. Li and A. Mourikis, "High-precision, consistent EKF-based visual-inertial odometry," *The International Journal of Robotics Research / Int. Journal of Robotics Research (IJRR)*, vol. 32, 2013.

[13] M. Li, B. H. Kim, and A. I. Mourikis, "Real-time motion tracking on a cellphone using inertial sensing and a rolling-shutter camera," in *2013 IEEE International Conference on Robotics and Automation*. IEEE, 2013, pp. 4712–4719.

[14] S. Leutenegger, S. Lynen, M. Bosse, R. Siegwart, and P. Furgale, "Keyframe-based visual-inertial odometry using nonlinear optimization," *The International Journal of Robotics Research / Int. Journal of Robotics Research (IJRR)*, 2014.

[15] S. Leutenegger, P. Furgale, V. Rabaud, M. Chli, K. Konolige, and R. Siegwart, "Keyframe-based visual-inertial SLAM using nonlinear optimization," *Proceedings of Robotis Science and Systems (RSS) 2013*, 2013.

[16] V. Usenko, J. Engel, J. Stückler, and D. Cremers, "Direct visual-inertial odometry with stereo cameras," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2016, pp. 1885–1892.

[17] L. Von Stumberg, V. Usenko, and D. Cremers, "Direct sparse visual-inertial odometry using dynamic marginalization," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 2510–2517.

[18] T. Qin, P. Li, and S. Shen, "Vins-mono: A robust and versatile monocular visual-inertial state estimator," *IEEE Transactions on Robotics*, vol. 34, no. 4, pp. 1004–1020, 2018.

[19] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardós, "ORB-SLAM: a versatile and accurate monocular SLAM system," *IEEE transactions on robotics*, vol. 31, no. 5, pp. 1147–1163, 2015.

[20] R. Mur-Artal and J. D. Tardós, "Visual-inertial monocular SLAM with map reuse," *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 796–803, 2017.

[21] S. Lovegrove, A. Patron-Perez, and G. Sibley, "Spline fusion: A continuous-time representation for visual-inertial fusion with application to rolling shutter cameras." in *BMVC*, vol. 2, no. 5, 2013, p. 8.

[22] J.-H. Kim, C. Cadena, and I. Reid, "Direct semi-dense SLAM for rolling shutter cameras," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2016, pp. 1308–1315.

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

[24] L. Carlone, Z. Kira, C. Beall, V. Indelman, and F. Dellaert, "Eliminating conditionally independent sets in factor graphs: A unifying perspective based on smart factors," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2014, pp. 4290–4297.

[25] T. Lupton and S. Sukkarieh, "Visual-inertial-aided navigation for high-dynamic motion in built environments without initial conditions," *IEEE Transactions on Robotics*, vol. 28, no. 1, pp. 61–76, 2012.

[26] 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.