High-fidelity Sensor Modeling and Self-Calibration in Vision-aided Inertial Navigation

Mingyang Li, Hongsheng Yu, Xing Zheng, and Anastasios I. Mourikis
Dept. of Electrical Engineering, University of California, Riverside
E-mail: {mli|hyu|xzheng|mourikis}@ee.ucr.edu

***Abstract*— In this paper, we propose a high-precision pose estimation algorithm for systems equipped with low-cost inertial sensors and rolling-shutter cameras. The key characteristic of the proposed method is that it performs *online self-calibration* of the camera and the IMU, using detailed models for both sensors and for their relative configuration. Specifically, the estimated parameters include the camera intrinsics (focal length, principal point, and lens distortion), the readout time of the rolling-shutter sensor, the IMU's biases, scale factors, axis misalignment, and g-sensitivity, the spatial configuration between the camera and IMU, as well as the time offset between the timestamps of the camera and IMU. An additional contribution of this work is a novel method for processing the measurements of the rolling-shutter camera, which employs an approximate representation of the estimation *errors*, instead of the state itself. We demonstrate, in both simulation tests and real-world experiments, that the proposed approach is able to accurately calibrate all the considered parameters in real time, and leads to significantly improved estimation precision compared to existing approaches.**

I. Introduction

In this work we address the problem of pose estimation using a low-cost inertial measurement unit (IMU) and a rolling-shutter camera. In recent years, the combination of cameras and inertial sensors has emerged as an attractive solution for 3D pose estimation. This is in large part due to the development of low-cost MEMS inertial sensors, which are suitable for use in systems where cost, weight, and power consumption are constrained. The combination of these sensors with low-cost cameras can lead to versatile, inexpensive localization systems, with applications in both robotics (e.g., micro aerial vehicles) and other areas (e.g., navigation aids for the visually impaired). Our interest in the use of a rolling-shutter camera is motivated by the fact that, in most cases, low-cost cameras employ rolling-shutter imagers.

Our main focus in this paper is on the detailed modeling and calibration of the sensors used for pose estimation, a prerequisite for high-quality state estimates. In prior work, the sensors' intrinsic characteristics, relative spatial configuration, and timing, are often calibrated by a combination of offline and online methods. On the one hand, the camera's intrinsic parameters are typically estimated offline via a specialized calibration process. On the other hand, the IMU's biases are typically estimated online, since their values usually drift over time. High-precision visual-inertial estimators sometimes calibrate additional parameters online, such as the camera-to-IMU transformation [1]–[4] or the time-offset that exists between the timestamps of the IMU and the camera [5].

The existing approaches have certain shortcomings. First, performing offline camera calibration is an often tedious process, which should be repeated periodically, since mechanical shocks and other factors can lead to changes in the camera parameters over time. Moreover, we note that, even if high-quality offline calibration is performed, the existence of some uncertainty in the camera parameters is inevitable. Treating the computed values as known constants (the standard approach) leads to unmodeled errors, which degrade the accuracy and reliability of any estimator. Finally, in existing approaches some of the non-ideal characteristics of IMUs (e.g., misalignment between the sensor axes, non-unit scale factors) are typically ignored. However, in low-cost MEMS IMUs these effects are likely to be significant, and ignoring them will reduce accuracy.

To address these limitations, in this paper we present a method for pose estimation and concurrent *self-calibration* that uses high-fidelity models for both the camera and the IMU. Specifically, we are interested in pose estimation in an unknown environment, using inertial measurements and observations of naturally-occurring features tracked by the rolling-shutter camera. We do not use any fiducial points or other knowledge about the structure of the scene. We demonstrate that, in this setting, it is possible to concurrently localize and perform *online* calibration of all of the following quantities:

- the IMU biases
- the misalignment and scale factors of the IMU sensors
- the acceleration dependence (typically called g-sensitivity) of the gyroscope measurements
- the camera-to-IMU spatial configuration
- the camera intrinsic parameters, including lens distortion
- the image readout time of the rolling-shutter camera
- the time offset between the timestamps of the camera and the IMU

We note that the use of a rolling-shutter camera, as opposed to a global-shutter one, requires special treatment: with a rolling-shutter camera each image row is captured at a slightly different time instant, and therefore, from a different camera pose. Since including in the estimator one state for each image row (the "exact" approach) is computationally intractable, all existing methods employ some assumption about the nature of the camera trajectory (see, e.g., [6]–[8]). By contrast, our approach employs *no* assumptions on the form of the camera trajectory itself. Instead, it uses an approximate representation for the time-evolution of the

estimation *errors* during the image readout time. Since these errors are typically small, this leads to only small modeling inaccuracies. Moreover, since the statistical properties of the errors are known in advance, we can compute upper bounds on the worst-case magnitude of these modeling errors.

This novel treatment of the rolling-shutter measurements is employed in conjunction with an extended Kalman filter (EKF)-based estimator. This EKF includes in its state vector all the calibration parameters described, in addition to the "standard" states needed for pose estimation (e.g., position, velocity, orientation, and feature positions). Through both Monte-Carlo simulations and real-world experiments we demonstrate that all the calibration parameters can be estimated with high precision. More importantly, however, our results show that jointly estimating the camera motion and *all* the calibration parameters leads to significant improvement in localization accuracy, compared to previous approaches that perform online estimation of only a subset of parameters.

II. Related Work

To the best of our knowledge, the topic of joint self-calibration of visual and inertial sensors has not been addressed in the past. In what follows, we discuss relevant approaches, divided into three categories:

a) Camera calibration: In the vast majority of cases, real-time vision-based localization algorithms assume that the camera's intrinsic parameters are known in advance, e.g., via an offline calibration (see [9] for a comprehensive review of calibration methods). Camera self-calibration is also possible. In [10] the authors present an online approach to self-calibration, which employs a Sum-of-Gaussians filter for visual SLAM. However, this is a vision-only approach, and thus IMU calibration is not addressed.

For a rolling-shutter camera, in addition to the geometry of the camera's projection model (e.g., focal length, principal point, lens distortion) one must also know the image readout time, i.e., the time needed to capture all the rows of the image. Calibrating this readout time is a much less studied issue. The approaches presented to date are offline ones, and require knowledge about the properties of the scene (e.g., a known calibration pattern in [8], or a LED flashing at a known frequency in [11]), a special motion of the camera [12], or a combination thereof. Moreover, these approaches require the camera's projection geometry to be perfectly known. By contrast, in our work both the camera geometry and the readout time are jointly estimated online.

b) IMU calibration: Gyroscope and accelerometer measurements are typically affected by biases, which are slowly changing over time. The majority of high-precision vision-aided inertial navigation algorithms model these biases, and estimate them online along with the system motion (see., e.g., [1], [2], [13] and references therein). However, the misalignment of the IMU axes, IMU scale factors, and the g-sensitivity of the gyroscope measurements are typically not estimated online. Most often these effects are assumed to be negligible (realistic for high-end, expensive, sensors), while offline calibration methods have also been employed [14], [15]. Our approach removes the need for offline calibration, by estimating all the above systematic errors online.

We note that methods for high-precision IMU calibration using a camera are presented in [16], [17]. In these works, a camera with known intrinsic parameters is used, in conjunction with a known calibration pattern. The visual measurements of known landmarks are then used in bundle adjustment [16] or in an EKF [17], to estimate the IMU characteristics as well as the transformation between the camera and IMU frames. Compared to these methods, the self-calibration approach we describe jointly estimates the IMU *and* camera parameters, without known scene structure.

c) Camera-IMU calibration: The estimation of the *spatial* and the *temporal* relationship between the IMU and camera data streams has been the subject of a number of recent papers. Most work has focused on estimating the transformation (i.e., rotation and translation) between the camera and IMU frames (see, e.g., [1]–[4] and references therein), while the temporal calibration between the camera and IMU is a less-explored topic [5], [18]. In all cases, however, the intrinsic parameters of the camera are perfectly known, and simplified (bias-only) models for the IMU are used. By contrast, in this work we employ detailed models both the camera and the IMU, whose parameters are estimated online, along with the spatial and temporal relationship between two sensors.

III. Estimator formulation

Consider a system equipped with a rolling-shutter camera and an IMU consisting of a 3-axis accelerometer and 3-axis gyroscope. Our goal is to concurrently estimate the state of the system with respect to a global coordinate frame $\{G\}$ and *all* the parameters of the two sensors' models. To track the motion of the system we affix to it a "body" coordinate frame, $\{B\}$, and track the motion of this frame with respect to $\{G\}$. The origin of the body frame is chosen as the point where the three accelerometer axes intersect, while its orientation is determined based on the camera frame, $\{C\}$. Specifically, we *define* the rotation of $\{B\}$ with respect to $\{C\}$ to be a known, constant matrix[1] ${}^C_B\mathbf{R}$. The specific value of ${}^C_B\mathbf{R}$ can be selected at will. For instance, it can be chosen as the identity matrix for simplicity, or set equal to the prior estimate of the camera-to-IMU rotation, so that the axes of $\{B\}$ are "close" to axes of the IMU sensors. In our implementation we use the latter.

A. Hybrid EKF

The state of the body frame is described by the orientation ${}^B_G\bar{\mathbf{q}}$, position ${}^G\mathbf{p}_B$, and velocity ${}^G\mathbf{v}_B$. The EKF-based estimator that we employ to track this state is a modification of the "hybrid" filter originally proposed in [19]. It combines a sliding-window formulation with a feature-based

[1]If we chose the orientation of $\{B\}$ independently of both the IMU and the camera, we would need to estimate both the orientation of $\{C\}$, and the IMU sensors' directions (which are uncertain due to potential misalignment) with respect to $\{B\}$. This can be shown to introduce unidentifiable parameters in the system model. Alternative choices exist to avoid this problem, such as aligning one axis of $\{B\}$ with one axis of the IMU [16]. However, that leads to more complex measurement models.

Algorithm 1 Hybrid EKF algorithm

Propagation: Propagate the state vector using the IMU readings, and the covariance matrix using (4).

Update: When camera measurements become available:

- Augment the sliding window with a new state, and begin image processing.
- For each feature track that is complete after m or fewer images, do the following
  - Triangulate the feature using all its observations.
  - Compute the feature-reprojection residuals (11) and their Jacobians (13), and apply the method of [20] to marginalize out the feature error.
  - Perform a Mahalanobis-distance gating test.
- For the features included in the state vector, compute the residuals (11) and measurement Jacobians (13).
- Perform an EKF update using all the feature residuals.
- Initialize into the state vector features that are still actively tracked after m images.

State Management:

- Remove SLAM features that are no longer tracked.
- Remove all sliding-window states that have no active feature tracks associated with them.

one, and is able to exploit the computational advantages of both formulations. The resulting hybrid algorithm has a low computational cost, and is capable of real-time operation in devices with limited CPU capabilities, as demonstrated in our experiments. We here briefly describe the structure of the estimator (see Algorithm 1), and refer the reader to [19] for further details. Subsequently, in Sections IV and V we present the IMU and camera measurement models used in the estimator to enable the online sensor calibration.

The state vector of the EKF at time-step k is given by[2]:

$$\mathbf{x}_k = \begin{bmatrix} \mathbf{x}_{E_k}^T & \mathbf{x}_{\mathbf{c}}^T & \mathbf{x}_{B_1}^T & \cdots & \mathbf{x}_{B_m}^T & \mathbf{f}_1^T & \cdots & \mathbf{f}_s^T \end{bmatrix}^T \quad (1)$$

where $\mathbf{x}_{E_k}$ is the "evolving state", comprising the current body-frame state as well as the time-varying IMU biases, defined in (3); $\mathbf{x_c}$ is the vector of parameters we seek to calibrate, defined in (10); the states $\mathbf{x}_{B_j}$, $j = 1 \ldots m$, are the body states corresponding to the time instants the past m images were recorded; and $\mathbf{f}_i$, $i = 1..s$, are the currently visible features, in inverse-depth parameterization.

When an IMU measurement is received, it is used to propagate the evolving state and covariance. On the other hand, when a new image is received, the sliding window of states is augmented with a new state. Note that each image is sampled over a time interval of non-zero duration (the rolling shutter readout time). By convention, we here consider that the timestamp associated with each image corresponds to the time instant the *middle* row of the image is captured. Therefore the state corresponding to each image in the filter represents the body-frame state at that time instant.

Point features are extracted and matched in the images, and the resulting feature tracks are processed in one of two ways: if a feature's track is lost after m or fewer images, it is used to provide constraints involving the poses of the sliding window and the calibration parameters. For this purpose, the multi-state-constraint method of [3], [20] is employed, which makes it possible to use the feature measurements without including the feature in the EKF state vector. On the other hand, if a feature is still being tracked after m frames, it is initialized in the state vector and any subsequent observations of it are used for updates as in the EKF-SLAM paradigm.

At each time step, the hybrid filter processes a number of features with each of the two approaches. For each feature the appropriate residuals and Jacobian matrices are computed, and a Mahalanobis-distance gating test is performed. All the features that pass the gating test are then employed for an EKF update. At the end of the update, features that are no longer visible and old sliding-window states with no active feature tracks associated with them are removed. Note that, to ensure the correct observability properties of the linearized system model, and thus improve the estimation accuracy and consistency, the hybrid filter employs fixed linearization points for computing Jacobian matrices [3].

IV. IMU MODEL AND STATE PROPAGATION

Due to the physical characteristics of the sensors, as well as imperfections of the manufacturing process, the IMU measurements are affected by systematic errors (in addition to random noise). For a full description of the sources of errors and their modeling, the reader is referred to [22]. In our work, we employ sensor models that include all the systematic errors that can be modeled linearly and have the most impact on precision.

Let us first consider the accelerometer measurements. Each of the three accelerometer sensors in an IMU provides scalar measurements of specific force, modeled as:

$$a_{m_i} = s_i {}^B\mathbf{u}_i^T {}^B\mathbf{a}_s + b_{a_i} + n_{a_i} \qquad i = 1, 2, 3 \quad (2)$$

where ${}^B\mathbf{u}_i$ is a unit vector along the sensing direction, s_i is a scale factor close to unity, b_{a_i} is a bias, n_{a_i} is random measurement noise, and ${}^B\mathbf{a}_s$ is the specific-force vector:

$${}^B\mathbf{a}_s = {}^B_G\mathbf{R}({}^G\mathbf{a}_B - {}^G\mathbf{g})$$

with ${}^G\mathbf{a}_B$ being the acceleration of the body frame and ${}^G\mathbf{g}$ being the gravity vector. Stacking the measurements of the three accelerometer sensors, we obtain the 3×1 vector:

$$\mathbf{a}_m = \mathbf{T}_a {}^B\mathbf{a}_s + \mathbf{b}_a + \mathbf{n}_a$$

where $\mathbf{T}_a$ is a 3×3 matrix whose i-th row is $s_i {}^B\mathbf{u}_i^T$, and $\mathbf{b}_a$ and $\mathbf{n}_a$ are vectors with elements b_{a_i} and n_{a_i}, respectively.

The gyroscope measurements are modeled as:

$$\boldsymbol{\omega}_m = \mathbf{T}_g {}^B\boldsymbol{\omega} + \mathbf{T}_s {}^B\mathbf{a}_s + \mathbf{b}_g + \mathbf{n}_w$$

where $\mathbf{T}_g$ and $\mathbf{T}_s$ are 3×3 matrices, $\mathbf{b}_g$ is the measurement bias, and $\mathbf{n}_w$ the measurement noise. Similarly to the case of

[2]Notation: The preceding superscript for vectors (e.g., G in ${}^G\mathbf{a}$) denotes the frame of reference with respect to which quantities are expressed. ${}^X_Y\mathbf{R}$ is the rotation matrix rotating vectors from frame $\{Y\}$ to $\{X\}$, and ${}^X_Y\bar{\mathbf{q}}$ is the corresponding unit quaternion [21]. ${}^X\mathbf{p}_Y$ represents the origin of frame Y with respect to frame X. $\mathbf{I}$ represents the identity matrix, and $\mathbf{0}$ the zero matrix. Finally, $\hat{a}$ is the estimate of a variable a, and $\tilde{a} \doteq a - \hat{a}$ is the error of the estimate.

the accelerometer measurements, $\mathbf{T}_g$ arises due to the scale factors in the gyroscope measurements and the misalignment of the gyroscope sensors to the principal axes of $\{B\}$. On the other hand, the matrix $\mathbf{T}_s$ represents the acceleration dependence (g-sensitivity) of the measurements, which can be significant for low-cost MEMS sensors [22].

Our goal is to estimate the values of the accelerometer and gyroscope bias, as well as the matrices $\mathbf{T}_a$, $\mathbf{T}_g$, and $\mathbf{T}_s$. We note that by estimating the matrices $\mathbf{T}_a$ and $\mathbf{T}_g$, we are effectively estimating the IMU sensors' scale factors, the sensors' misalignment, *and* their direction with respect to the camera frame. Specifically, the scale factors of the accelerometer and gyroscope sensors are given by the norm of the rows of $\mathbf{T}_a$ and $\mathbf{T}_g$, respectively, while the sensors' direction in $\{B\}$ is defined by the unit vector corresponding to each row (see (2)). Knowing these unit vectors makes it possible to estimate the misalignment of the sensors. Moreover, since the rotation between the camera frame and $\{B\}$ is by definition a known constant, these unit vectors also provide us with the direction of the IMU sensors with respect to $\{C\}$.

Following standard practice, the bias vectors $\mathbf{b}_a$ and $\mathbf{b}_g$ are modeled as Gaussian random-walk processes, while the matrices $\mathbf{T}_a$, $\mathbf{T}_g$, and $\mathbf{T}_s$ are assumed to be uncertain but constant parameters[3]. Therefore, the evolving state of the EKF, $\mathbf{x}_E$, is given by:

$$\mathbf{x}_E = \begin{bmatrix} {}^B_G\bar{\mathbf{q}}^T & {}^G\mathbf{p}_B^T & {}^G\mathbf{v}_B^T & \mathbf{b}_g^T & \mathbf{b}_a^T \end{bmatrix} \tag{3}$$

On the other hand, the matrices $\mathbf{T}_a$, $\mathbf{T}_g$, and $\mathbf{T}_s$ are part of the calibration-parameter vector $\mathbf{x_c}$ (see (1)). Specifically, we define a 27×1 vector $\mathbf{x}_{\mathbf{c}_{IMU}}$, comprising all the elements of these three matrices, and include it in $\mathbf{x_c}$, as shown in (10).

1) IMU-based propagation: The IMU measurements are used for propagating the state estimates between timesteps. Specifically, given the IMU measurements in a certain time interval, as well as estimates for the IMU parameters, we compute the estimated acceleration and rotational velocity of the body frame as:

$$\begin{aligned} {}^B\hat{\mathbf{a}}_s &= \hat{\mathbf{T}}_a^{-1}\left(\mathbf{a}_m - \hat{\mathbf{b}}_a\right) \\ {}^B\hat{\boldsymbol{\omega}} &= \hat{\mathbf{T}}_g^{-1}\left(\boldsymbol{\omega}_m - \hat{\mathbf{T}}_s\,{}^B\hat{\mathbf{a}}_s - \hat{\mathbf{b}}_g\right) \end{aligned}$$

These are subsequently be used to propagate the estimate of the evolving state via numerical integration, as described in [3]. Moreover, the covariance matrix of the EKF is propagated. For this step, we first compute the Jacobian matrices $\boldsymbol{\Phi}_k$ and $\boldsymbol{\Gamma}_k$, which describe the relationship between the evolving-state errors in propagation:

$$\tilde{\mathbf{x}}_{E_{k+1}} = \boldsymbol{\Phi}_k\tilde{\mathbf{x}}_{E_k} + \boldsymbol{\Gamma}_k\tilde{\mathbf{x}}_{\mathbf{c}_{IMU}} + \mathbf{w}_k$$

where $\mathbf{w}_k$ is the process-noise error, modeled as zero-mean Gaussian with covariance matrix $\mathbf{Q}_k$. The matrices $\boldsymbol{\Phi}_k$ and $\boldsymbol{\Gamma}_k$ are computed analytically, similarly to [3]. Based on the above expression, the covariance matrix of the EKF, $\mathbf{P}_k$, is propagated as

$$\mathbf{P}_{k+1} = \begin{bmatrix} \boldsymbol{\Phi}_k & \boldsymbol{\Gamma}_k & \mathbf{0} \\ \mathbf{0} & \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{I} \end{bmatrix} \mathbf{P}_k \begin{bmatrix} \boldsymbol{\Phi}_k & \boldsymbol{\Gamma}_k & \mathbf{0} \\ \mathbf{0} & \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{I} \end{bmatrix}^T + \mathbf{Diag}(\mathbf{Q}_k, \mathbf{0}) \tag{4}$$

V. Camera model

Let us consider that an image with timestamp t is received. Due to the time delays inevitably affecting each sensor, this timestamp is affected by a time offset, t_d, relative to the IMU timestamps [5]. This means that the middle row of the image was actually captured at time $t+t_d$. Moreover, since we are using a rolling-shutter camera, each of the N rows in the image is captured at a slightly different time instant. If the camera readout time is t_r, these time instants are given by:

$$t_n = t + t_d + \frac{nt_r}{N}, \qquad n \in \left[-\frac{N}{2}, \frac{N}{2}\right]$$

If a feature with position ${}^G\mathbf{p}_f$ is observed at the row sampled at t_n, its image coordinates are described by:

$$\mathbf{z} = \mathbf{h}({}^C\mathbf{p}_f(t_n)) + \mathbf{n} \tag{5}$$

where ${}^C\mathbf{p}_f(t_n)$ is the position of the feature with respect to the camera frame at time t_n, $\mathbf{n}$ is the measurement noise vector, and $\mathbf{h}(\cdot)$ is the camera's projection function. The vector ${}^C\mathbf{p}_f(t_n)$ can be written as:

$${}^C\mathbf{p}_f(t_n) = {}^C_B\mathbf{R}\,{}^B_G\mathbf{R}(t_n)\left({}^G\mathbf{p}_f - {}^G\mathbf{p}_B(t_n)\right) + {}^C\mathbf{p}_B \tag{6}$$

where ${}^C\mathbf{p}_B$ is the position of the origin of $\{B\}$ in the camera frame, which must be estimated. Moreover, the camera projection is modeled by the standard perspective model with radial and tangential distortion [23]. Denoting ${}^C\mathbf{p}_f(t_n) = \begin{bmatrix} {}^Cx_f & {}^Cy_f & {}^Cz_f \end{bmatrix}^T$, we have:

$$\mathbf{h}\left({}^C\mathbf{p}_f\right) = \mathbf{p}_c + \begin{bmatrix} a_u & 0 \\ 0 & a_v \end{bmatrix}\begin{bmatrix} u_d \\ v_d \end{bmatrix} \tag{7}$$

where $\mathbf{p}_c$ is the pixel location of the principal point, a_u and a_v represent the camera focal length measured in horizontal and vertical pixel units, respectively, and

$$\begin{bmatrix} u_d \\ v_d \end{bmatrix} = d\begin{bmatrix} u \\ v \end{bmatrix} + \begin{bmatrix} 2t_1uv + t_2(u^2+v^2+2uv) \\ t_1(u^2+v^2+2uv) + 2t_2uv \end{bmatrix} \tag{8}$$

$$d = 1 + k_1(u^2+v^2) + k_2(u^2+v^2)^2 + k_3(u^2+v^2)^3$$

$$\begin{bmatrix} u \\ v \end{bmatrix} = \frac{1}{{}^Cz_f}\begin{bmatrix} {}^Cx_f \\ {}^Cy_f \end{bmatrix} \tag{9}$$

In the above, k_i, $i=1,2,3$ are the radial distortion coefficients, while t_1, t_2 are the tangential distortion coefficients.

We can now define the 41×1 vector containing all the calibration parameters estimated in the EKF, which includes *all* the parameters appearing in the IMU and camera models:

$$\mathbf{x_c} = \begin{bmatrix} \mathbf{x}_{\mathbf{c}_{IMU}}^T & {}^C\mathbf{p}_B^T & \mathbf{p}_c^T & a_u & a_v & k_1 & k_2 & k_3 & t_1 & t_2 & t_r & t_d \end{bmatrix}^T \tag{10}$$

The estimation of these parameters proceeds as normal in an EKF, by computing the Jacobians of the measurement models with respect to them, and updating their estimates during EKF updates. In this process, the uncertainty of the

[3]Note that, if desired, it is straightforward to also model $\mathbf{T}_a$, $\mathbf{T}_g$, and $\mathbf{T}_s$ by random-walk models. This however was not deemed necessary for the sensors used in our testing.

calibration parameters as well as the effect of this uncertainty on the state estimates, is modeled in a seamless way via the EKF equations.

A. Rolling-shutter modeling

We now show how the measurements of the rolling-shutter camera can be processed. It is important to note that, as seen in (6), the feature measurements at different rows (i.e., different n) in one image depend on body-frame states at *different* time instants. Since it is impractical to include in the state vector all these states, previous approaches typically employ assumptions about the nature of the motion during the image readout time (e.g., constant-velocity motion [6], [7], or higher-order parametric forms [8], [24]). However, low-dimensional models may lead to loss of modeling fidelity, while high-dimensional models incur high computational costs.

We here follow a different approach. Specifically, we begin by observing that in any EKF-based estimator the processing of the feature measurements requires (i) the residual of each measurement, and (ii) a linear (linearized) expression describing the dependence of this residual on the errors of the EKF state vector. The key idea in our approach is to employ *no* assumptions on the form of the trajectory when computing the residual, by exploiting the IMU measurements. Instead, we employ an approximate representation for the trajectory *errors* when linearizing. We express the errors during the entire readout interval as a function of the error in the state at middle of the interval. This, in turn, allows us to only include this one state in the EKF sliding window, obtaining a computationally efficient algorithm. These steps are described in more detail in what follows.

Computing the residual: The residual corresponding to the feature measurement (5) is defined as:

$$\mathbf{r} = \mathbf{z} - \mathbf{h}\left({}^{C}_{B}\mathbf{R}\,{}^{B}_{G}\hat{\mathbf{R}}(\hat{t}_n)\left({}^{G}\hat{\mathbf{p}}_f - {}^{G}\hat{\mathbf{p}}_B(\hat{t}_n)\right) + {}^{C}\hat{\mathbf{p}}_B\right) \quad (11)$$

where $\hat{t}_n = t + \hat{t}_d + \frac{n\hat{t}_r}{N}$. Note that, as described in Section III, the EKF state vector only contains an estimate for the body state at the time instant the middle row of the image is captured, $\hat{\mathbf{x}}_B(t + t_d)$. Therefore, the quantities ${}^{B}_{G}\hat{\mathbf{R}}(\hat{t}_n)$ and ${}^{G}\hat{\mathbf{p}}_B(\hat{t}_n)$ appearing in (11) are not readily available. To obtain them, we integrate the IMU measurements in the time interval $[t + \hat{t}_d, \hat{t}_n]$, starting from $\hat{\mathbf{x}}_B(t + \hat{t}_d)$ (note that backward integration may be necessary). Since we are employing direct integration of the IMU measurements to compute $\hat{\mathbf{x}}_B(\hat{t}_n)$, arbitrary motions can be described as long as they are within the bandwidth of the IMU, and no low-dimensional motion model is needed.

Residual linearization: To derive an expression relating the residual in (11) to the errors in $\hat{\mathbf{x}}_B(t + \hat{t}_d)$, we begin by linearizing the camera observation model in (11), to obtain:

$$\mathbf{r} \simeq \mathbf{H}_{\boldsymbol{\theta}}\tilde{\boldsymbol{\theta}}_B(\hat{t}_n) + \mathbf{H}_{\mathbf{p}}\,{}^{G}\tilde{\mathbf{p}}_B(\hat{t}_n) + \mathbf{H}_{\mathbf{c}}\tilde{\mathbf{x}}_{\mathbf{c}} + \mathbf{H}_{\mathbf{f}}\tilde{\mathbf{f}} + \mathbf{n} \quad (12)$$

where $\mathbf{H}_{\boldsymbol{\theta}}$ and $\mathbf{H}_{\mathbf{p}}$ are the Jacobians of the measurement function with respect to the orientation and position at time $\hat{t}_n$, $\mathbf{H}_{\mathbf{f}}$ is the Jacobian with respect to the feature position, and $\mathbf{H}_{\mathbf{c}}$ is the Jacobian with respect to $\mathbf{x}_{\mathbf{c}}$. For details on the definition of the orientation error, $\tilde{\boldsymbol{\theta}}_B$, and the computation of the time-related Jacobians, please refer to [5].

Since (12) involves the errors of quantities at time $\hat{t}_n$, which do not appear in the EKF state, it cannot be used directly for an EKF update. We therefore proceed to express the errors at $\hat{t}_n$ as a function of the errors at $t + \hat{t}_d$. Starting with the position error, and using Taylor-series expansion, we obtain:

$${}^{G}\tilde{\mathbf{p}}_B(\hat{t}_n) = {}^{G}\tilde{\mathbf{p}}_B(t+\hat{t}_d) + \frac{n\hat{t}_r}{N}{}^{G}\tilde{\mathbf{v}}_B(t+\hat{t}_d) + \frac{(n\hat{t}_r)^2}{2N^2}{}^{G}\tilde{\mathbf{a}}_B + \dots$$

This expression would be exact if all terms were kept, but with a finite number of terms a truncation error is incurred. The key advantage here is that, since we have prior knowledge about the magnitude of the estimation errors, we can *predict* the worst-case modeling error incurred by any choice of truncation order. For example, if we only keep the first two terms, then the truncation error in each direction is upper bounded by $\frac{(nt_r)^2}{2N^2}\epsilon_a$ where ϵ_a is the worst-case acceleration error. Using $\epsilon_a = 2$ m/sec^2 (which is larger than any value seen in practice in our tests) and $t_r = 33$ msec, the truncation error is upper bounded by 6.9×10^{-5} m along each axis. For the characteristics of the camera used in our experiments, this translates to a worst-case unmodelled residual term of 0.08 pixels, when imaging objects at a depth of 2 m.

Thus if we model the position error using two terms, as

$${}^{G}\tilde{\mathbf{p}}_B(\hat{t}_n) \simeq {}^{G}\tilde{\mathbf{p}}_B(t + \hat{t}_d) + \frac{n\hat{t}_r}{N}{}^{G}\tilde{\mathbf{v}}_B(t + \hat{t}_d)$$

we only incur a minimal loss of modeling accuracy. A similar analysis for the orientation errors shows that even if we truncate the corresponding series after a single term, by using the approximation $\tilde{\boldsymbol{\theta}}_B(\hat{t}_n) \simeq \tilde{\boldsymbol{\theta}}_B(t + \hat{t}_d)$, the worst-case unmodelled residual terms remain below 0.33 pixels, even allowing for rotational velocity errors of 1 deg/sec.

Employing these approximations in (12), we obtain the following linearized equation for the measurement residual:

$$\mathbf{r} \simeq \mathbf{H}_{\boldsymbol{\theta}}\tilde{\boldsymbol{\theta}}_B(t + \hat{t}_d) + \mathbf{H}_{\mathbf{p}}\,{}^{G}\tilde{\mathbf{p}}_B(t + \hat{t}_d) + \frac{n\hat{t}_r}{N}\mathbf{H}_{\mathbf{p}}\,{}^{G}\tilde{\mathbf{v}}_B(t + \hat{t}_d) + \mathbf{H}_{\mathbf{c}}\tilde{\mathbf{x}}_{\mathbf{c}} + \mathbf{H}_{\mathbf{f}}\tilde{\mathbf{f}} + \mathbf{n} \quad (13)$$

This expression can be used for EKF updates, as it only involves the errors in the body state at $t + \hat{t}_d$, the feature, and the calibration parameters, which are all part of the EKF state vector.

VI. Simulations

We present results from two sets of Monte-Carlo simulations demonstrating the performance the proposed algorithm. To obtain realistic simulation setups, we generate the simulated trajectories and the IMU and camera measurements with characteristics (e.g., noise, feature properties) matching those of actual datasets.

First test: For the first test, we base our simulation on a dataset involving significant rotations and translations, while moving in a room-sized environment for a period of 45 sec. The data was collected with the rolling-shutter camera and the Invensense MPU-6050 IMU found on an

TABLE I: Simulations: RMS errors of the calibration parameters

Time (sec)	$\mathbf{b}_g$ (^{o}/sec)	$\mathbf{b}_a$ (m/sec^2)	${}^C\mathbf{p}_B$ (cm)	t_d (msec)	$\mathbf{T}_g$	$\mathbf{T}_a$	$\mathbf{T}_s$ (^{o}/sec/g)	a_u, a_v (pix.)	$\mathbf{p}_c$ (pix.)	k_i	t_i	t_r (msec)
0	0.334	0.187	0.610	9.450	0.0200	0.0200	0.216	5.146	4.941	0.082	0.0010	4.171
15	0.014	0.009	0.160	0.076	0.0007	0.0011	0.020	0.475	0.470	0.015	0.0004	0.190
45	0.013	0.007	0.110	0.062	0.0006	0.0008	0.010	0.400	0.332	0.012	0.0003	0.108

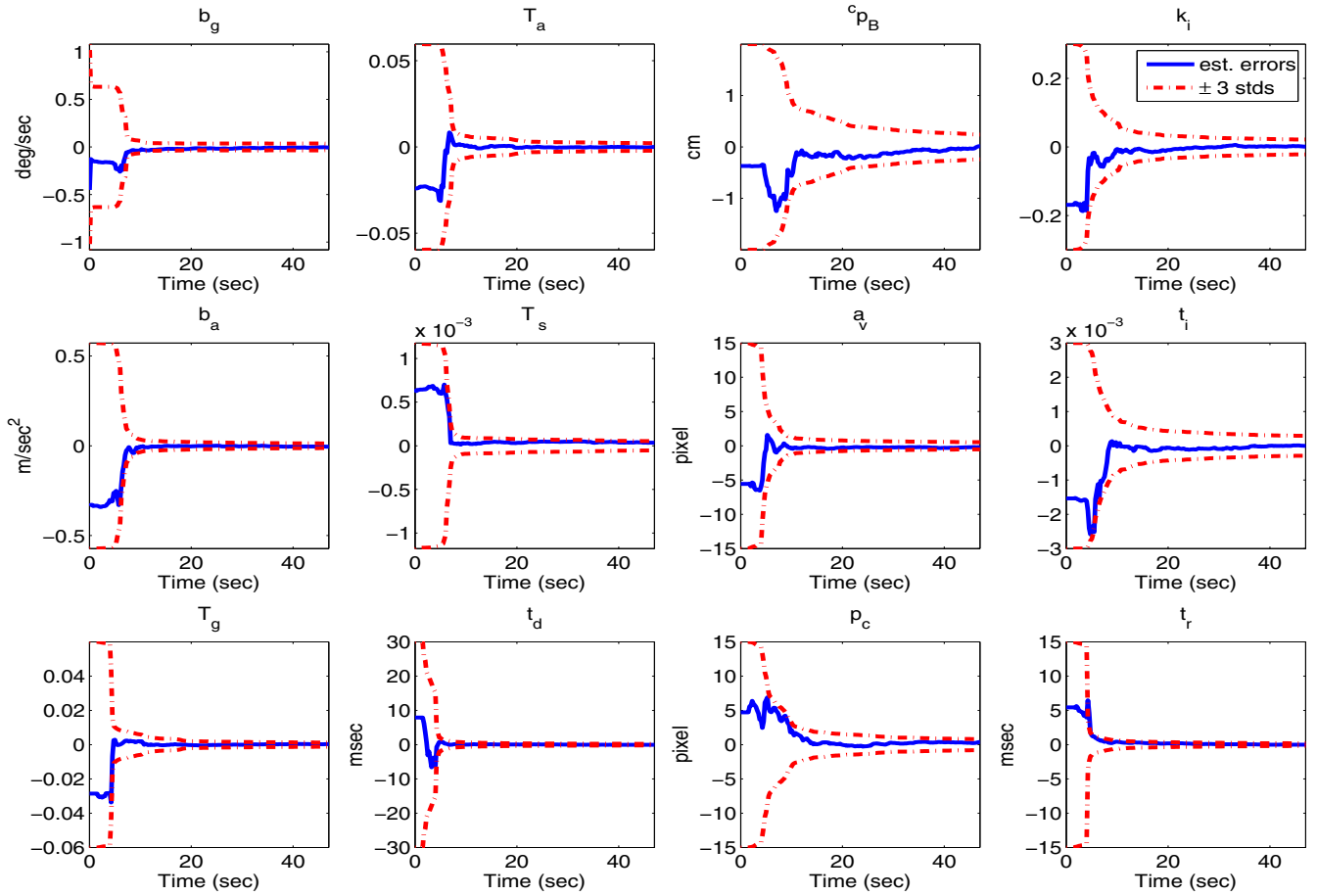

Fig. 1: Results of a representative simulation trial: estimation errors and the reported ± 3 standard deviations. (Top to bottom, left to right) (a) gyroscope biases, (b) accelerometer biases, (c) gyroscope misalignment/scale, (d) accelerometer misalignment/scale, (e) g-sensitivity, (f) camera-to-IMU time offset, (g) camera-to-IMU position, (h) camera focal lengths, (i) camera principal point, (j) camera radial distortion, (k) camera tangential distortion, and (l) rolling shutter readout time. For all vectorial parameters the least accurate element is shown.

LG Nexus 4 device. In each trial of the simulation tests, the calibration parameters were set equal to known nominal values, with the addition of errors drawn from zero-mean Gaussian distributions.

Table I shows the RMS errors over 100 Monte-Carlo simulations for all the parameters. The first line corresponds to the initial errors, the second line to the errors after 15 sec of motion, and the third line to the final errors. We note that the initial RMS error values for the $\mathbf{T}_a$ and $\mathbf{T}_g$ matrices correspond to misalignment errors with standard deviation of approximately 1.15^o and/or scale factors of 2%. These values (as well as most of the other initial errors shown in Table I) are larger than what we typically find in practice. From the results of this table we can clearly observe that the proposed online calibration approach can accurately estimate all the desired quantities. Moreover, we note that the estimation errors after 15 seconds are significantly smaller than the initial ones, and almost as same as the final errors, indicating quick convergence.

It is also useful to examine the results of a single representative trial, shown in Fig. 1. These plots show the estimation errors and ± 3 standard deviations reported by the filter for all the calibration parameters. Due to the limited space, for the vector parameters (e.g., biases, the $\mathbf{T}_a$, $\mathbf{T}_g$, $\mathbf{T}_s$ matrices, and so on) we only plot the least accurate element. These figures also demonstrate the rapid reduction in uncertainty, and show that the uncertainty reported by the EKF is commensurate with the actual errors, indicating consistency [3]. It is important to note that the results of both Fig. 1 and Table I suggest that, with sufficiently

TABLE II: Initial standard deviation of the calibration parameters in the second set of simulations

$\mathbf{T}_g$	$\mathbf{T}_a$	$\mathbf{T}_s$ (°/sec/g)	a_u, a_v (pix.)
0.0058	0.0058	0.169	0.350
$\mathbf{p}_c$ (pix)	k_i	t_i	t_r (msec)
0.450	0.01	0.001	0.500

TABLE III: Second simulation: RMS errors of pose estimation

	${}^{B}_{G}\bar{\mathbf{q}}$ (°)	${}^{G}\mathbf{p}_B$ (m)	${}^{G}\mathbf{v}_B$ (m/sec)
Full calibration	0.468	0.619	0.027
Partial calibration	2.045	2.914	0.092

exciting motion, the calibration parameters are *observable* (identifiable), a result that we currently seek to prove.

Second test: We now present the results of a second set of simulations, which demonstrates the improvement in performance gained by using high-fidelity IMU models and online calibration. We compare the proposed approach that uses full self-calibration against a "partial calibration" approach, in which only the IMU biases, camera-to-IMU calibration, and the time offset between the camera and IMU is estimated (similarly to [5]). In both cases, the IMU and camera are affected by *small* calibration errors, whose standard deviations are shown in Table II. Note that the chosen values for $\mathbf{T}_a$ and $\mathbf{T}_g$ correspond to scale factors of less than 1% and/or axis misalignment of approximately 0.3 degrees (these are similar to or smaller than the values we encountered in practice for MEMS sensors). Moreover, the camera parameters' errors are similar to what we obtain with offline calibration.

We conducted 100 Monte-Carlo tests, generated based on a 6-minute, 400-m long dataset. In each trial the calibration parameters are corrupted by random errors, drawn from zero-mean Gaussian pdfs with the standard deviations given in Table II. Table III shows the RMS error for the position, orientation, and velocity, in the case of full calibration (the proposed approach) vs. partial calibration. We can clearly observe a significant difference in accuracy in all state variables. Moreover, we should point out that the partial-calibration approach *failed* in 15% of the trials, due to the presence of unmodeled errors (the results of the failed trials are not included in the calculations). By properly modeling the uncertainty of the calibration, the proposed approach is more robust to the presence of errors (no simulation trials failed for the full-calibration approach). We thus see that by using high-fidelity models of the IMU and camera, and performing online estimation of the model parameters, we are able to obtain better precision as well as increased reliability of the estimator.

VII. Real-world experiment

We next present results from a real-world experiment, which was conducted using the sensors of a Nexus 4 device. The experiment involves motion in three floors of the UCR Engineering building, by a person holding the device. The total duration of the experiment is 9.8 min and the trajectory length is approximately 633 m. The IMU sample

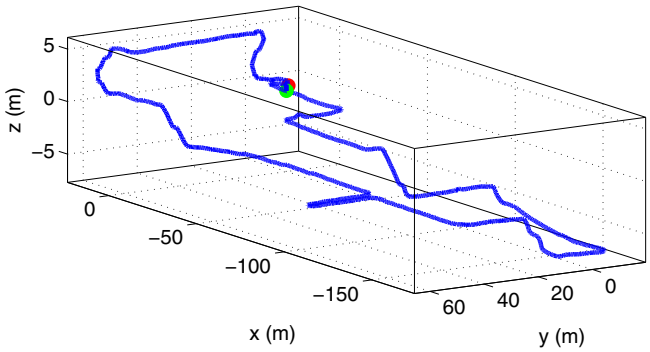

Fig. 2: Real world experiment: Estimated trajectory. The red mark represents the initial position, while the green mark represents the end position.

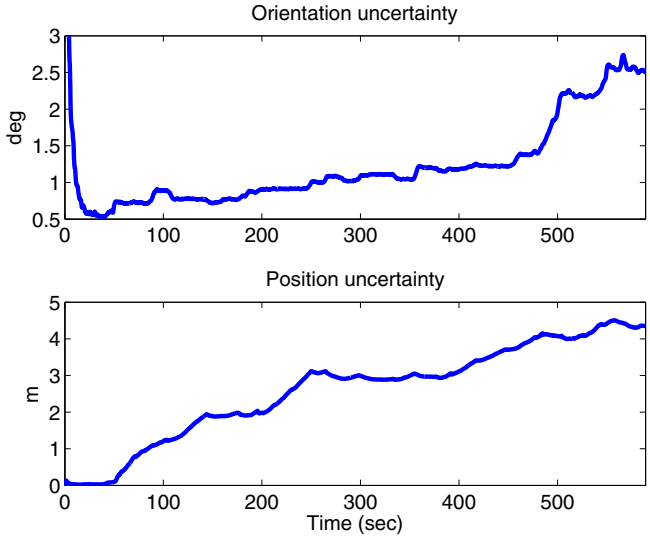

Fig. 3: Real world experiment: Orientation and position uncertainty (3σ) reported by the EKF during the experiment. The plotted values correspond to the major axis of the uncertainty ellipses. Note the sharp drop in the initial orientation uncertainty, which occurs as the calibration parameters become more certain during the first few seconds.

rate is 200 Hz, while the images are captured at 22 Hz (sample images are shown in Fig. 4). Shi-Tomasi features are extracted in the images [25], and matched by normalized cross-correlation. All data was processed in real-time on the phone, with the average time needed per EKF update being 18 msec. This time is significantly lower than the image period, which shows that the proposed method is capable of real-time operation.

For this experiment, prior estimates for the calibration parameters are obtained as follows: zero values are used for the biases and for the g-sensitivity matrix, $\mathbf{T}_s$; the priors for $\mathbf{T}_a$ and $\mathbf{T}_g$ are set to the identity matrix; the image readout time was set equal to the image period; the camera-to-IMU translation prior is set to zero; the time-offset prior was set to zero; and finally for the camera intrinsics the calibration parameters from a different device of the same type were used. This is done to demonstrate that even rough information about the camera parameters can be employed

Fig. 4: Sample images recorded during the experiment.

as an initial guess for the proposed method.

Fig. 2 shows the trajectory estimated by the proposed approach, while the reported uncertainties of the IMU orientation and position are shown in Fig. 3. Although an accurate ground truth for the entire trajectory cannot be obtained, the final position error is equal to 1.09 m, corresponding to 0.17% of the travelled distance. This error is commensurate with the reported uncertainty, indicating a consistent estimator. Due to limited space, the complete results of the online calibration cannot be included here, but are available online, along with the video from the experiment [26]. We point out that the largest misalignment of the IMU axes was estimated as 0.85^o, the maximum scale factor deviation from unity was estimated as 1.8%, and the maximum value of the g-sensitivity corresponds to 0.33^o/sec/g. As seen in the simulation results of the preceding section, these values are large enough to cause significant degradation of performance, if not properly modeled.

VIII. Conclusion

In this paper, we have presented an online approach to the joint self-calibration of a rolling-shutter camera and a low-cost IMU during vision-aided inertial navigation. Our approach employs detailed models of the IMU, the camera projection geometry, as well as of the relative spatial configuration and timing of the sensors. Our simulation results and experimental testing demonstrate that by including *all* the parameters of the sensor models in the state vector of the proposed EKF, it is possible to obtain precise estimates of their values, and to improve the accuracy of the pose estimates. These results suggest that by including all the parameters in the state vector, the state remains observable, and the additional parameters are identifiable in general trajectories. A formal proof (or refutation) of this result is the subject of ongoing work.

Acknowledgments

This work was supported by the National Science Foundation (grant no. IIS-1117957, IIS-1253314 and IIS-1316934).

References

[1] E. Jones and S. Soatto, "Visual-inertial navigation, mapping and localization: A scalable real-time causal approach," *International Journal of Robotics Research*, vol. 30, no. 4, pp. 407–430, Apr. 2011.

[2] J. Kelly and G. Sukhatme, "Visual-inertial sensor fusion: Localization, mapping and sensor-to-sensor self-calibration," *International Journal of Robotics Research*, vol. 30, no. 1, pp. 56–79, Jan. 2011.

[3] M. Li and A. I. Mourikis, "High-precision, consistent EKF-based visual-inertial odometry," *International Journal of Robotics Research*, vol. 32, no. 6, pp. 690–711, May 2013.

[4] 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 environment," in *Proceedings of the IEEE International Conference on Robotics and Automation*, St Paul, MN, May 2012, pp. 957–964.

[5] M. Li and A. I. Mourikis, "3-D motion estimation and online temporal calibration for camera-IMU systems," in *Proceedings of the IEEE International Conference on Robotics and Automation*, Karlsruhe, Germany, May 2013, pp. 5689–5696.

[6] J. Hedborg, P. Forssen, M. Felsberg, and E. Ringaby, "Rolling shutter bundle adjustment," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Providence, RI, June 2012, pp. 1434 –1441.

[7] M. Li, B. Kim, and A. I. Mourikis, "Real-time motion estimation on a cellphone using inertial sensing and a rolling-shutter camera," in *Proceedings of the IEEE International Conference on Robotics and Automation*, Karlsruhe, Germany, May 2013, pp. 4697–4704.

[8] L. Oth, P. Furgale, L. Kneip, and R. Siegwart, "Rolling shutter camera calibration," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Portland, OR, Jun. 2013.

[9] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. Cambridge, UK: Cambridge University Press, 2000.

[10] J. Civera, D. R. Bueno, A. Davison, and J. M. M. Montiel, "Camera self-calibration for sequential bayesian structure from motion," in *Proceedings of the IEEE International Conference on Robotics and Automation*, Kobe, Japan, May 2009, pp. 403–408.

[11] M. Meingast, C. Geyer, and S. Sastry, "Geometric models of rolling-shutter cameras," in *Proceedings of the 6th Workshop on Omnidirectional Vision, Camera Networks and Non-Classical Cameras*, Beijing, China, 2005.

[12] A. Karpenko, D. Jacobs, J. Back, and M. Levoy, "Digital video stabilization and rolling shutter correction using gyroscopes," Stanford University, Tech. Rep., 2011.

[13] A. I. Mourikis, N. Trawny, S. I. Roumeliotis, A. E. Johnson, A. Ansar, and L. H. Matthies, "Vision-aided inertial navigation for spacecraft entry, descent, and landing," *IEEE Transactions on Robotics*, vol. 25, no. 2, pp. 264–280, Apr. 2009.

[14] D. Strelow, "Motion estimation from image and inertial measurements," Ph.D. dissertation, Carnegie Mellon University, Nov. 2004.

[15] P.-J. Bristeau, F. Callou, D. Vissire, and N. Petit, "The navigation and control technology inside the AR.Drone Micro UAV," in *18th World Congress of the International Federation of Automatic Control*, Milano, Italy, August 2011, pp. 1477–1484.

[16] C. Krebs, "Generic IMU-camera calibration algorithm: Influence of IMU-axis on each other," ETH Zrich, Tech. Rep., December 2012.

[17] D. Zachariah and M. Jansson, "Joint calibration of an inertial measurement unit and coordinate transformation parameters using a monocular camera," in *International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, Zurich, September 2010, pp. 1–7.

[18] J. Kelly and G. S. Sukhatme, "A general framework for temporal calibration of multiple proprioceptive and exteroceptive sensors," in *Proceedings of the International Symposium of Experimental Robotics*, New Delhi, India, December 2010.

[19] M. Li and A. I. Mourikis, "Optimization-based estimator design for vision-aided inertial navigation," in *Proceedings of Robotics: Science and Systems*, Sydney, Australia, Jul. 2012.

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

[21] N. Trawny and S. I. Roumeliotis, "Indirect Kalman filter for 6D pose estimation," University of Minnesota, Dept. of Comp. Sci. & Eng., Tech. Rep. 2005-002, Jan. 2005.

[22] D. Titterton and J. Weston, Eds., *Strapdown Inertial Navigation Technology, 2nd Edition*. IEE, 2005.

[23] J. Heikkila and O. Silven, "A four-step camera calibration procedure with implicit image correction," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Washington, DC, 1997, pp. 1106–1113.

[24] J. Hedborg, E. Ringaby, P. Forssen, and M. Felsberg, "Structure and motion estimation from rolling shutter video," in *IEEE International Conference on Computer Vision Workshops*, Barcelona, Spain, Nov. 2011, pp. 17 –23.

[25] J. Shi and C. Tomasi, "Good features to track," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Seattle, WA, June 1994, pp. 593–600.

[26] www.ee.ucr.edu/∼mli/ICRA2014.