Solin, Arno; Cortés Reina, Santiago; Rahtu, Esa; Kannala, Juho

PIVO

Published in:
Proceedings - 2018 IEEE Winter Conference on Applications of Computer Vision, WACV 2018

DOI:
10.1109/WACV.2018.00073

Published: 01/01/2018

Document Version
Peer-reviewed accepted author manuscript, also known as Final accepted manuscript or Post-print

Please cite the original version:
Solin, A., Cortés Reina, S., Rahtu, E., & Kannala, J. (2018). PIVO: Probabilistic Inertial-Visual Odometry for Occlusion-Robust Navigation. In *Proceedings - 2018 IEEE Winter Conference on Applications of Computer Vision, WACV 2018* (pp. 616-625). (IEEE Winter Conference on Applications of Computer Vision). IEEE. https://doi.org/10.1109/WACV.2018.00073

PIVO: Probabilistic Inertial-Visual Odometry for Occlusion-Robust Navigation

Arno Solin
Aalto University
arno.solin@aalto.fi

Santiago Cortés
Aalto University
santiago.cortesreina@aalto.fi

Esa Rahtu
Tampere Univ. of Tech.
esa.rahtu@tut.fi

Juho Kannala
Aalto University
juho.kannala@aalto.fi

Abstract

This paper presents a novel method for visual-inertial odometry. The method is based on an information fusion framework employing low-cost IMU sensors and the monocular camera in a standard smartphone. We formulate a sequential inference scheme, where the IMU drives the dynamical model and the camera frames are used in coupling trailing sequences of augmented poses. The novelty in the model is in taking into account all the cross-terms in the updates, thus propagating the inter-connected uncertainties throughout the model. Stronger coupling between the inertial and visual data sources leads to robustness against occlusion and feature-poor environments. We demonstrate results on data collected with an iPhone and provide comparisons against the Tango device and using the EuRoC data set.

1. Introduction

Accurate tracking of the orientation and location of mobile devices is important for many applications. Examples include robot navigation, pedestrian navigation and wayfinding, augmented reality, games, safety and rescue (*e.g.* for firefighters). Motivated by the practical relevance of the problem, there is plenty of recent research on device tracking based on various kinds of sensors. Perhaps one of the most promising approaches for precise real-time tracking is visual-inertial odometry, which is based on fusing measurements from inertial sensors (*i.e.* an accelerometer and gyroscope) with visual feature tracking from video. This kind of approach is practical since both video cameras and MEMS-based inertial measurement units are nowadays low-cost commodity hardware which are embedded in most smartphones and tablet devices. Further, the two sensor modalities complement each other; visual odometry can provide good precision in visually distinguishable environments whereas inertial navigation is robust to occlusion and able to provide absolute metric scale for the motion.

Visual-inertial odometry methods have already been demonstrated in many research papers and in some commercial devices, such as the Tango device by Google. However, most published methods are heavily camera-based in the sense that they need unobstructed visibility and plenty of visual features in the scene all the time. For example, the tracking of the Tango device fails if the camera lens is fully occluded. Further, since pure inertial navigation has been considered challenging or impossible with low-cost consumer grade inertial sensors [28], so far there are no occlusion-robust visual-inertial odometry demonstrated using conventional smartphone hardware.

In this paper, we propose a probabilistic inertial-visual odometry technique which allows occlusion-robust navigation on standard smartphone hardware. Our approach is able to utilize consumer-grade MEMS-based inertial sensors in a more robust manner than previous approaches and we therefore call our approach *inertial-visual* odometry instead of the commonly used term 'visual-inertial' odometry. By taking the full advantage of the inertial sensors we are able to demonstrate precise odometry with a standard iPhone in use cases where the camera is temporarily fully occluded (*e.g.* in a bag or by people in a crowd) for extended periods of time. A visual example of such a path is shown in Figure 1.

We aim at maximizing the information throughput from both the visual and inertial modality. This means both accurate tracking and insensitivity to partial and complete occlusion for extended periods of time. More exactly, this goal is covered by the following three contributions of this paper novel to visual-inertial schemes: (i) An efficient IMU propagation model in discrete-time, which keeps the error sources limited to the linearization error; (ii) A trailing pose augmentation through a series of linear Kalman updates which preserve the cross-covariance between current and past poses; (iii) A visual update step where the estimated feature point coordinate is integrated out, and thus the only approximation error comes from the linearization in the EKF. For all of the methodology a core design criterion is formulating the methods in sequential manner suitable for real-time running on current mobile hardware and building upon noisy sensor sources available in standard smartphones.

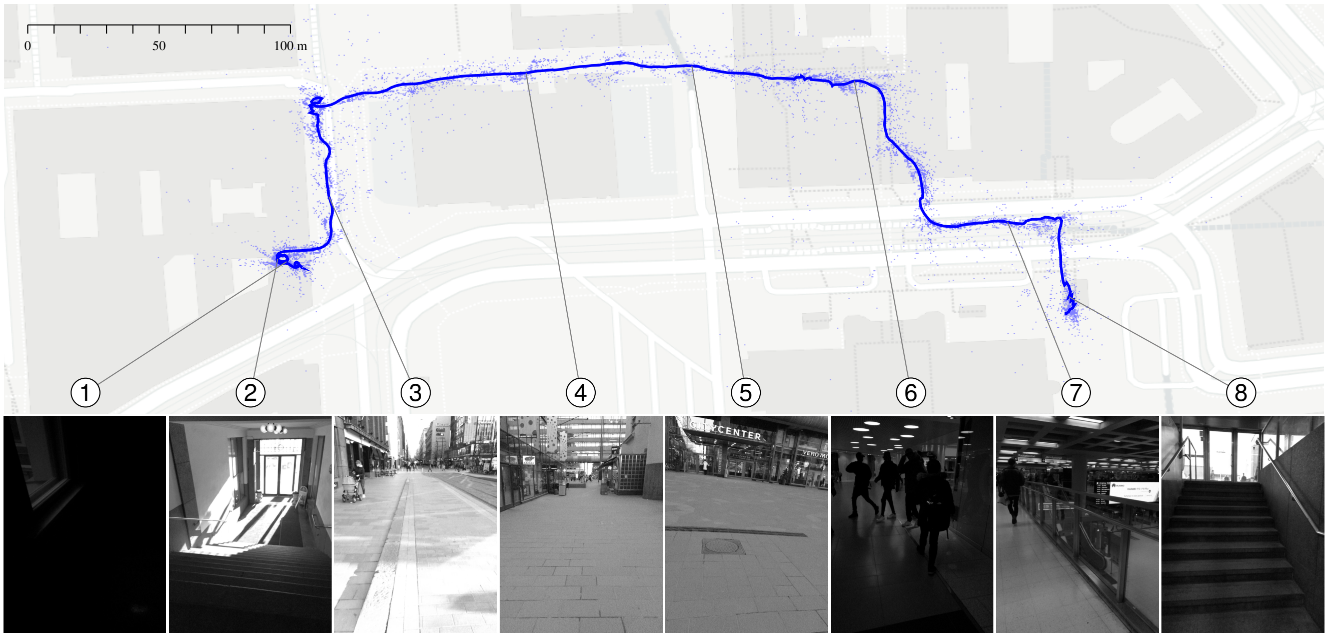


Figure 1: PIVO tracking on a smartphone (iPhone 6) starting from an office building (1–2), through city streets (3–5), a shopping mall (6), and underground transportation hub (7–8).

2. Related work

Methods for tracking the pose of a mobile device with six degrees of freedom can be categorized based on (i) the input sensor data, or (ii) the type of the approach. Regarding the latter aspect the main categories are simultaneous localization and mapping (SLAM) approaches, which aim to build a global map of the environment and utilize it for loop-closures and relocalization, and pure odometry approaches, which aim at precise sequential tracking without building a map or storing it in memory. SLAM is particularly beneficial in cases where the device moves in a relatively small environment and revisits the mapped areas multiple times, since the map can be used for removing the inevitable drift of tracking. However, accurate low-drift odometry is needed in cases where the device moves long distances without revisiting mapped areas.

Regarding the types of input data there is plenty of literature using various combinations of sensors. Monocular visual SLAM and odometry techniques, which use a single video camera, are widely studied [5, 8, 18] but they have certain inherent limitations which hamper their practical use. That is, one can not recover the absolute metric scale of the scene with a monocular camera and the tracking breaks if the camera is occluded or there are not enough visual features visible all the time. For example, homogenous textureless surfaces are quite common in indoor environments but lack visual features. Moreover, even if the metric scale of the device trajectory would not be necessary in all applications, monocular visual odometry can not keep a consistent scale if the camera is not constantly translating—that is, pure rotations cause problems [12]. Scale drift may cause artifacts even without pure rotational motion if loop-closures can not be frequently utilized [33].

Methods that use stereo cameras are able to recover the metric scale of the motion, and consistent tracking is possible even if the camera rotates without translating [6, 25]. Still, the lack of visually distinguishable texture and temporary occlusions (*e.g.* in a crowd) are problem for all approaches that utilize only cameras. Recently, due to the increasing popularity of depth sensing RGB-D devices (either utilizing structured light or time-of-flight), also SLAM and odometry approaches have emerged for them [14, 17, 27]. These devices provide robustness to lack of texture but they also require unobstructed line of sight, and hence occlusions may still be a problem. In addition, many of the cameras have a limited range for depth sensing and do not work outdoors because they utilize infrared projectors.

Thus, in order to make tracking more robust and practical for consumer applications on mobile devices both indoors and outdoors, it has become common to combine video cameras with inertial measurement units (IMUs) [2, 7, 13, 24, 26, 35]. Examples of hardware platforms that provide built-in visual-inertial odometry are Google Tango and Microsoft Hololens devices. However, both of these devices contain custom hardware components (*e.g.* a fisheye lens camera), which are not common in conventional smartphones. In addition, there are several research papers which utilize IMUs with custom stereo camera setups [9, 11, 20, 37].

In this paper we present a probabilistic approach for fusing information from consumer grade inertial sensors (*i.e.* 3-axis accelerometer and gyroscope) and a monocular video camera for accurate low-drift odometry. This is practically the most interesting hardware setup as most modern smartphones contain a monocular video camera and an IMU. Despite the wide application potential of such hardware platform, there are not many previous works which demonstrate visual-inertial odometry using standard smartphone sensors. This is most likely due to the relatively low quality of low-cost IMUs which makes inertial navigation challenging. The most notable papers covering the smartphone use case are [18, 21, 34]. However, all these previous approaches are either *visual-only* or *visual-heavy* in the sense that tracking breaks if there is complete occlusion of camera for short periods of time. This is the case also with the visual-inertial odometry of the Google Tango device.

3. Inertial-visual information fusion

Consider a device with a monocular camera, an IMU with 3-axis gyroscope/accelerometer, and known camera-to-IMU translational and rotational offsets—a characterization that matches modern day smartphones. In the following, we formulate the PIVO approach for fusing information from these data sources such that we maintain all dependencies between uncertain information sources—up to the linearization error from the non-linear filtering approach. The outline of the PIVO method is summarized in Algorithm 1.

3.1. Non-linear filtering for information fusion

In the following, we set the notation for the non-linear filtering approach (see [29] for an overview) for information fusion in the paper. We are concerned with non-linear state-space equation models of form

$$\mathbf{x}_k = \mathbf{f}_k(\mathbf{x}_{k-1}, \boldsymbol{\varepsilon}_k), \tag{1}$$

$$\mathbf{y}_k = \mathbf{h}_k(\mathbf{x}_k) + \boldsymbol{\gamma}_k, \tag{2}$$

where $\mathbf{x}_k \in \mathbb{R}^n$ is the state at time step t_k, $k = 1, 2, \ldots$, $\mathbf{y}_k \in \mathbb{R}^m$ is a measurement, $\boldsymbol{\varepsilon}_k \sim \mathrm{N}(\mathbf{0}, \mathbf{Q}_k)$ is the Gaussian process noise, and $\boldsymbol{\gamma}_k \sim \mathrm{N}(\mathbf{0}, \mathbf{R}_k)$ is the Gaussian measurement noise. The dynamics and measurements are specified in terms of the dynamical model function $\mathbf{f}_k(\cdot, \cdot)$ and the measurement model function $\mathbf{h}_k(\cdot)$, both of which can depend on the time step k. The extended Kalman filter (EKF, [1, 15, 29]) provides a means of approximating the state distributions $p(\mathbf{x}_k \mid \mathbf{y}_{1:k})$ with Gaussians: $p(\mathbf{x}_k \mid \mathbf{y}_{1:k}) \simeq \mathrm{N}(\mathbf{x}_k \mid \mathbf{m}_{k|k}, \mathbf{P}_{k|k})$.

The linearizations inside the extended Kalman filter cause some errors in the estimation. Most notably the estimation scheme does not preserve the norm of the orientation quaternions. Therefore after each update an extra quaternion normalization step is added to the estimation scheme.

Algorithm 1: Outline of the PIVO method.

```
Initialize the state mean and covariance
foreach IMU sample pair $(\mathbf{a}_k, \boldsymbol{\omega}_k)$ do
    Propagate the model with the IMU sample            see Sec. 3.2
    Perform the EKF prediction step
    if new frame is available then
        track visual features
        foreach feature track do
            Jointly triangulate feature using poses in state and
              calculate the visual update proposal      see Sec. 3.4
            if proposal passes check then
                Perform the EKF visual update
        Update the trail of augmented poses              see Sec. 3.3
```

In case either the dynamical (1) or measurement model (2) is linear (*i.e.* $\mathbf{f}_k(\mathbf{x}, \boldsymbol{\varepsilon}) = \mathbf{A}_k\,\mathbf{x} + \boldsymbol{\varepsilon}$ or $\mathbf{h}_k(\mathbf{x}) = \mathbf{H}_k\,\mathbf{x}$, respectively), the prediction/update steps reduce to the closed-form solutions given by the conventional Kalman filter.

3.2. IMU propagation model

The state variables of the system hold the information of the current system state and a fixed-length window of past poses in the IMU coordinate frame:

$$\mathbf{x}_k = (\mathbf{p}_k, \mathbf{q}_k, \mathbf{v}_k, \mathbf{b}^{\mathrm{a}}_k, \mathbf{b}^{\omega}_k, \mathbf{T}^{\mathrm{a}}_k, \boldsymbol{\pi}^{(1)}, \boldsymbol{\pi}^{(2)}, \ldots, \boldsymbol{\pi}^{(n_{\mathrm{a}})}), \tag{3}$$

where $\mathbf{p}_k \in \mathbb{R}^3$ is the device position, $\mathbf{v}_k \in \mathbb{R}^3$ the velocity, and $\mathbf{q}_k \in \mathbb{R}^4$ the orientation quaternion at time step t_k. Additive accelerometer and gyroscope biases are denoted by $\mathbf{b}^{\mathrm{a}}_k \in \mathbb{R}^3$ and $\mathbf{b}^{\omega}_k \in \mathbb{R}^3$, respectively. $\mathbf{T}^{\mathrm{a}}_k \in \mathbb{R}^{3\times3}$ holds the multiplicative accelerometer bias. The past device poses are kept track of by augmenting a fixed-length trail of poses, $\{\boldsymbol{\pi}^{(i)}\}_{i=1}^{n_{\mathrm{a}}}$, where $\boldsymbol{\pi}^{(i)} = (\mathbf{p}_i, \mathbf{q}_i)$, in the state.

Contrary to many previous visual-inertial methods, we seek to define the propagation method directly in discrete-time following Solin *et al.* [32]. The benefits are that the derivatives required for the EKF prediction are available in closed-form, no separate ODE solver iteration is required, and possible pitfalls (see, *e.g.*, [30]) related to the traditional continuous-discrete formulation [15] can be avoided.

The IMU propagation model is given by the mechanization equations

$$\begin{pmatrix} \mathbf{p}_k \\ \mathbf{v}_k \\ \mathbf{q}_k \end{pmatrix} = \begin{pmatrix} \mathbf{p}_{k-1} + \mathbf{v}_{k-1}\Delta t_k \\ \mathbf{v}_{k-1} + [\mathbf{q}_k(\tilde{\mathbf{a}}_k + \boldsymbol{\varepsilon}^{\mathrm{a}}_k)\mathbf{q}^{\star}_k - \mathbf{g}]\Delta t_k \\ \boldsymbol{\Omega}[(\tilde{\boldsymbol{\omega}}_k + \boldsymbol{\varepsilon}^{\omega}_k)\Delta t_k]\mathbf{q}_{k-1} \end{pmatrix}, \tag{4}$$

where the time step length is given by $\Delta t_k = t_k - t_{k-1}$, the acceleration input is $\tilde{\mathbf{a}}_k \in \mathbb{R}^3$ and the gyroscope input by $\tilde{\boldsymbol{\omega}}_k \in \mathbb{R}^3$. Gravity $\mathbf{g}$ is a constant. The quaternion rotation is denoted by $\mathbf{q}_k[\cdot]\mathbf{q}^{\star}_k$, and the quaternion update is given by $\boldsymbol{\Omega} : \mathbb{R}^3 \to \mathbb{R}^{4\times4}$ (see, *e.g.*, [36]). The process noises associated with the IMU data are treated as i.i.d. Gaussian noise $\boldsymbol{\varepsilon}^{\mathrm{a}}_k \sim \mathrm{N}(\mathbf{0}, \boldsymbol{\Sigma}^{\mathrm{a}}\Delta t_k)$ and $\boldsymbol{\varepsilon}^{\omega}_k \sim \mathrm{N}(\mathbf{0}, \boldsymbol{\Sigma}^{\omega}\Delta t_k)$.

Low-cost IMU sensors suffer from misalignment and scale errors in addition to white measurement noise. These are accounted for in the dynamic model by including estimation for multiplicative and additive scale bias for the accelerometer, $\tilde{\mathbf{a}}_k = \mathbf{T}^{\mathrm{a}}_k \, \mathbf{a}_k - \mathbf{b}^{\mathrm{a}}_k$, and an additive bias for the gyroscope, $\tilde{\boldsymbol{\omega}}_k = \boldsymbol{\omega}_k - \mathbf{b}^{\omega}_k$, where $\mathbf{a}_k$ and $\boldsymbol{\omega}_k$ are the accelerometer and gyroscope sensor readings at t_k, and $\mathbf{b}^{\mathrm{a}}_k$ and $\mathbf{b}^{\omega}_k$ are the additive biases. Miscalibrations in the accelerometer scale are handled by the diagonal scale error matrix $\mathbf{T}^{\mathrm{a}}_k$.

In this work, we ignore the slow random walk of the sensor biases, and thus the biases and scale error terms are considered fixed and estimated as a part of the state. Similarly, the augmented poses $\boldsymbol{\pi}^{(i)}$ also have no temporal dynamics. The complete dynamical model must be differentiated both in terms of the state variables and process noise terms for the EKF estimation scheme. The derivatives are available in closed-form, which helps preserving the numerical stability of the system. As a part of the estimation setup we deal with zero-velocity and velocity pseudo-measurement updates similarly as in [32].

3.3. Camera pose augmentation

Once a new camera frame is acquired at time $t_\star$ (the $t_\star$ is matched against the IMU clock such that $t_\star$ is assigned the most recent t_k), the current camera pose given by the state variables $\boldsymbol{\pi}^\star = (\mathbf{p}_\star, \mathbf{q}_\star)$ is augmented into the state by a linear Kalman update.

The camera pose augmentation is a two-step process. First, the pose trail propagation step takes care of moving the past poses forward, discarding the oldest pose, and initiating a prior for the current pose. As the state distributions are all Gaussians, in probabilistic sense this corresponds to the following linear Kalman prediction step with the following dynamic model

$$\mathbf{A}_\star = \begin{pmatrix} \mathbf{I}_{19} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{I}_{7(n_{\mathrm{a}}-1)} & \mathbf{0} \end{pmatrix} \tag{5}$$

and $\mathbf{Q}_\star = \mathrm{blkdiag}(\mathbf{0}_{19}, \sigma_{\mathrm{p}}^2 \, \mathbf{I}_3, \sigma_{\mathrm{q}}^2 \, \mathbf{I}_4, \mathbf{0}_{7(n_{\mathrm{a}}-1)})$, where the prior variances for the new pose position and orientation, σ_{p}^2 and σ_{q}^2, are set large to keep the prior uninformative. $\mathbf{I}_d$ and $\mathbf{0}_d$ denote identity and zero matrices with dimension d, respectively.

The second step of the pose augmentation takes care of including the current system pose frontmost in the pose history trail in the state. As the state distribution is Gaussian, the state update corresponds to a linear Kalman update step with the following measurement model matrix

$$\mathbf{H}_\star = \begin{pmatrix} \mathbf{I}_7 & \mathbf{0}_{7\times 19} & -\mathbf{I}_7 & \mathbf{0}_{7\times 7(n_{\mathrm{a}}-1)} \end{pmatrix} \tag{6}$$

and a measurement noise covariance $\mathbf{R} = \sigma_\star^2 \, \mathbf{I}$ very close to zero and representing the small residual error related to the misalignment of the sensor and frame timestamps. Setting $\mathbf{y}_\star = \mathbf{0}$, defines a Kalman update that enforces the current pose to match the first augmented state slot $\boldsymbol{\pi}^{(1)}$.

The length of the augmented pose trail n_{a} is a selectable parameter, it can be tuned according to the available computing resources or the accuracy requirements. In theory the length of the filter state as well as the frame rate of the camera could also be adaptively controlled during the estimation.

Even though the state only holds a fixed number of poses and the oldest pose is discarded, formally the poses are not removed from the model. Our formulation means that the result is the same as it would have been if *every pose* would remain in the state, even though the information is used only in the trailing window. This would of course be computationally infeasible and totally unnecessary as the information is not used later on. However, it means that the state variables are altered only in a fashion that preserves the cross-covariances between future and past states.

3.4. Visual update model

The visual measurement model formulation bears resemblance to the model used in MSCKF [24], but the linearization for the EKF update is handled differently. The motivation is that tracking a visual feature over several camera frames imposes a long time-range constraint on the device movement together with the camera intrinsics. What we underline is that not only the feature points, but also all interconnected uncertainties between *all* state variables need to be bundled in the update. See Figure 2a for an outline of the elements involved in the visual update model. This makes the derivation of the visual update step slightly more complicated than that in [24].

For feature tracking, we use the Shi–Tomasi *Good features to track* approach [31] to determine strong corners in the image. These features are tracked across frames by a pyramidal Lucas–Kanade tracker [3, 22]. Lost features are replaced by new features and with preference of having the features evenly scattered over the field of view.

In this work the visual update is performed per tracked feature, which keeps the computational complexity constant per feature. The feature update for a given track j of feature point pixel coordinates $\mathbf{y}^{(j)} = (u_1, v_1, u_2, v_2, \ldots, u_m, v_m)$ seen in m past frames with augmented poses $\boldsymbol{\pi}^{(i)} = (\mathbf{p}^{(i)}, \mathbf{q}^{(i)})$ still in the state ($m \leq n_{\mathrm{a}}$).

We construct a 'full' measurement model representing any projection of an arbitrary feature point coordinate to the observed image pixel coordinates:

$$\mathbf{y}_k^{(j)} = \mathbf{h}_k^{(j)}(\mathbf{x}) + \boldsymbol{\gamma}_k, \quad \boldsymbol{\gamma}_k \sim \mathrm{N}(\mathbf{0}, \sigma_{\mathrm{uv}}^2 \, \mathbf{I}_{2m}), \tag{7}$$

where σ_{uv} is the measurement noise standard deviation in pixel units. In the following we drop the time index subscript k and will write out the function $\mathbf{h}$ per frame i (*i.e.*

for pixel pairs $\mathbf{y}_i = (u_i, v_i)$ in $\mathbf{y}$). In our formulation the feature global coordinate $\mathbf{p}_*^{(j)} \in \mathbb{R}^3$ will be integrated out in the final model, which differs from previous approaches. We, however, write out the derivation of the model by including the estimation of $\mathbf{p}_*^{(j)}$.

$$\mathbf{h}^{(j,i)}(\mathbf{x}) = \mathbf{g}\big(\mathbf{R}({}^{\mathrm{C}}\mathbf{q}^{(i)})\,(\mathbf{p}_*^{(j)} - {}^{\mathrm{C}}\mathbf{p}^{(i)})\big), \qquad (8)$$

where the rotation and translation in the global frame corresponds to the camera extrinsics calculated from the device pose and known rotational and translational offsets between IMU and camera coordinate frames (denoted by the superscript 'C' in Eq. 8). The camera projection is modeled by a standard perspective model $\mathbf{g} : \mathbb{R}^3 \to \mathbb{R}^2$ with radial and tangential distortion [10] and calibrated off-line.

To estimate the position $\mathbf{p}_*^{(j)}$ of a tracked feature we employ a similar approach as in [24], where the following minimization problem is set up: $\boldsymbol{\theta}_* = \arg\min_{\boldsymbol{\theta}} \sum_{i=1}^{m} \|\boldsymbol{\varphi}_i(\boldsymbol{\theta})\|$, where we use the inverse depth parametrization, $\boldsymbol{\theta} = 1/p_{\mathrm{z}}(p_{\mathrm{x}}, p_{\mathrm{y}}, 1)$, to avoid local minima and improve numerical stability [23]. The target function $\boldsymbol{\varphi}_i : \mathbb{R}^3 \to \mathbb{R}^2$ can be defined as follows on a per frame basis:

$$\boldsymbol{\varphi}_i(\boldsymbol{\theta}) = \tilde{\mathbf{y}}_i - h_{i,3}^{-1}\begin{pmatrix} h_{i,1} & h_{i,2} \end{pmatrix}^{\mathsf{T}}, \qquad (9)$$

$$\mathbf{h}_i = \mathbf{C}_i \begin{pmatrix} \theta_1 & \theta_2 & 1 \end{pmatrix}^{\mathsf{T}} + \theta_3\,\mathbf{t}_i, \qquad (10)$$

$$\mathbf{C}_i = \mathbf{R}({}^{\mathrm{C}}\mathbf{q}^{(i)})\,\mathbf{R}^{\mathsf{T}}({}^{\mathrm{C}}\mathbf{q}^{(1)}), \qquad (11)$$

$$\mathbf{t}_i = \mathbf{R}({}^{\mathrm{C}}\mathbf{q}^{(i)})\,\big(\mathbf{p}^{(1)} - \mathbf{p}^{(i)}\big), \qquad (12)$$

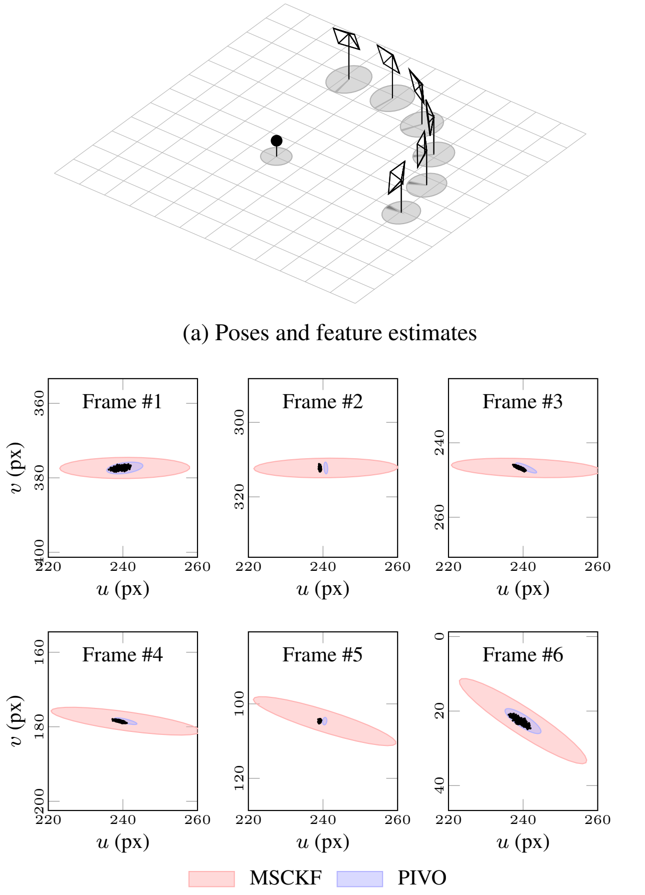
where feature pixel coordinates $\tilde{\mathbf{y}}_i$ are undistorted from $\mathbf{y}_i$. For solving the minimization problem a Gauss–Newton minimization scheme is employed:

$$\boldsymbol{\theta}^{(s+1)} = \boldsymbol{\theta}^{(s)} - (\mathbf{J}_{\boldsymbol{\varphi}}^{\mathsf{T}}\mathbf{J}_{\boldsymbol{\varphi}})^{\mathsf{T}}\,\mathbf{J}_{\boldsymbol{\varphi}}^{\mathsf{T}}\,\boldsymbol{\varphi}(\boldsymbol{\theta}^{(s)}), \qquad (13)$$

where $\mathbf{J}_{\boldsymbol{\varphi}}$ is the Jacobian of $\boldsymbol{\varphi}$. The iteration is initialized by an intersection estimate $\boldsymbol{\theta}^{(0)}$ calculated just from the first and last pose.

The beef of this section is that in order to do a precise EKF update with measurement model (7) the entire procedure described after Equation (7) needs to be differentiated in order to derive the closed-form Jacobian $\mathbf{H}_{\mathbf{x}} : \mathbb{R}^n \to \mathbb{R}^{2m \times n}$. This includes differentiating the entire Gauss–Newton scheme iterations with respect to *all* state variables.

The effect of taking into account all the cross-derivatives is illustrated in Figure 2b. The figure illustrates how the visual update for a three frame long feature track shows in the extended Kalman filter, where the estimated feature location must be summarized into a multivariate Gaussian distribution. The black dots depict the true distribution calculated by a Monte Carlo scheme, the red patch show the 95% confidence ellipses for the MSCKF visual update, and the blue patch the 95% confidence ellipses for the PIVO visual update. The MSCKF approximation is coarse, but the approximate density covers the true one with high certainty. Taking all cross-correlation into account and successfully accounting for the sensitivity of the estimated feature location, makes the PIVO update model more accurate. The directions of the correlations (tilt of the distributions) are interpreted right. The nature of the local linearization can still keep the mean off.



(a) Poses and feature estimates

(b) Comparison between PIVO and MSCKF visual update model

Figure 2: (a) A visual feature observed by a trail of camera poses with associated uncertainties. In (b) the black dots are the 'true' distributions for the visual update model. The red patch shows shape of the Gaussian approximation used by the MSCKF visual update, and blue shows the shape of the approximation used by PIVO.

When proposing a visual update, for robustness against outlier tracks, we use the standard chi-squared innovation test approach (see, *e.g.*, [1]), which takes into account both the predicted visual track and the estimate uncertainty.

4. Results

In the following we present a number of experiments which aim to demonstrate the proposed method to be com-

Table 1: Comparison results on the EuRoC MAV dataset in absolute trajectory error RMSE (median) in meters. The other tabulated results for other methods are collected from Krombach *et al.* [19]† and Mur-Artal and Tardós [25]‡, [26]*.

Dataset	PIVO (ours)	HYBRID†	LIBVISO2†	LSD-SLAM†	ORB-SLAM†	S-PTAM†	ORB-SLAM2‡	VI-SLAM*
V1 01 (easy)	0.82 (0.31)	0.25 (0.18)	0.31 (0.31)	0.19 (0.10)	0.79 (0.62)	0.28 (0.19)	0.035	0.027
V1 02 (medium)	—	0.24 (0.16)	0.29 (0.27)	0.98 (0.92)	0.98 (0.87)	0.50 (0.35)	0.020	0.028
V1 03 (difficult)	0.72 (0.48)	0.81 (0.76)	0.87 (0.64)	—	2.12 (1.38)	1.36 (1.09)	0.048	—
V2 01 (easy)	0.11 (0.07)	0.22 (0.13)	0.40 (0.31)	0.45 (0.41)	0.50 (0.42)	2.38 (1.78)	0.037	0.032
V2 02 (medium)	0.24 (0.15)	0.31 (0.25)	1.29 (1.08)	0.51 (0.48)	1.76 (1.39)	4.58 (4.18)	0.035	0.041
V2 03 (difficult)	0.51 (0.23)	1.13 (0.97)	1.99 (1.66)	—	—	—	—	0.074
Mean	0.48 (0.25)	0.49 (0.41)	0.85 (0.71)	0.53 (0.48)	1.23 (0.94)	1.82 (1.52)	0.035	0.040

parable with the current state-of-the-art, to provide added value in comparison to the Google Tango device, and to provide useful results in challenging real-world environments with smartphone hardware.

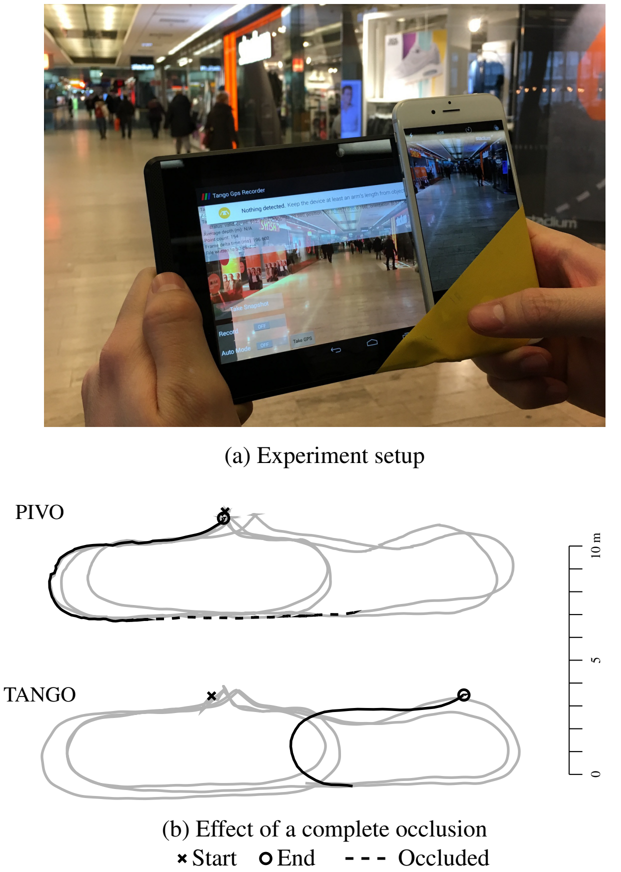
4.1. EuRoC MAV data

We compare the proposed PIVO method to other recent methods by using the publicly available EuRoC MAV dataset [4]. This dataset contains sequences of data recorded from an IMU and two synchronized video cameras on board a micro-aerial vehicle. We consider a subset of the data which contains ground-truth trajectories from a VICON system, and we only use camera '0'.

The EuRoC MAV sequences contain typically several passes through the same scene and thus are best suited for SLAM approaches where a map is built on the fly, or methods employing automatic loop-closures. The inertial information contains very pronounced high-frequency noise from the rotors and relatively soft movement slowing down sensor bias estimation. We present our results on the dataset to show that our system performs on par with recently published methods despite this use not being its core purpose.

Our results are shown in Table 1. The values for other methods are obtained from [19, 25, 26]. All other methods in Table 1, except ours and VI-SLAM [26], are visual SLAM methods for monocular or stereo cameras. The results for LIBVISO2, LSD-SLAM, S-PTAM, and ORB-SLAM have been computed by Krombach *et al.* using open-source software from original authors with default parameters [19]. Our results are comparable with [19] but worse than in [25, 26]. While both [26] and our method use inertial sensors, ours is not a SLAM approach, that is we do not utilize loop-closures or re-localization like [25, 26].

It is important to note that map reuse is highly beneficial in the case of EuRoC MAV datasets since the camera moves in a small environment only some meters in size. However, the situation would be different in use cases where the device moves long distances without re-visiting previous areas. In fact, instead of SLAM methods, our results should be compared to other visual-inertial odometry approaches, such as [37]. Unfortunately, [37] does not report the global RMS error on EuRoC data and detailed comparison is therefore not possible.



(a) Experiment setup

(b) Effect of a complete occlusion

Figure 3: Comparison study between PIVO (iPhone 6) and a Tango device. The Tango is sensitive to occlusions and failed to return to the starting point at the end.

4.2. Demonstration of robustness to occlusion

In the following experiments, we use an off-the-shelf Apple iPhone 6. The data acquisition was implemented in Objective-C, while the reconstruction was done off-line in Mathworks Matlab with performance critical components written in C++. The IMU data is sampled at 100 Hz. The experiments use the rear-facing camera at 10 fps, with a resolution of 480×640 (portrait orientation), grayscale images, shutter speed $1/60$, an ISO value of 200, and locked focus. The IMU and camera data were acquired time-synced on the device. The camera distortion parameters were estimated off-line prior to the data acquisition.

There are not many publicly available visual-inertial odometry methods, and none of them is shown to be robust to occlusion and to work with standard MEMS-based inertial sensors and rolling shutter cameras of current smartphones (namely those of iPhone 6) in challenging conditions (*e.g.* with rapid motions and moving obstacles). Therefore, the proprietary visual-inertial odometry technology of the Google Tango device was the only baseline which we could use for comparisons. We attached the iPhone and the Tango device together, and captured similar motion trajectories with both devices. We observed that both approaches, PIVO and Tango, produced comparable and accurate trajectories in normal capture conditions without occlusions, but only PIVO was robust to occlusion.

During data acquisition the iPhone was firmly attached to the Tango in order to keep the tracks aligned (see Fig. 3a). We ran the built-in visual-inertial odometry method on the Tango. Figure 3b shows an example of estimated tracks both for PIVO and the Tango device using data acquired in a shopping mall. After walking for some 110 m, the camera was completely occluded for the dashed part of the path. While the Tango track exhibits a discontinuity at the occlusion, PIVO is able to track movement even though being completely occluded. At the end of the path PIVO is horizontally off from the starting point 29 cm (0.23% rel. error to traveled distance).

In the second occlusion experiment, which is also illustrated in Figure 4 above. In this experiment, the Tango device was again rigidly attached to the iPhone 6, and a walk was performed between seven known points in a building. The points were close to supporting pillars of the building and their location was precisely determined from the architectural floor plan. The camera lenses of both Tango and iPhone were occluded by hand simultaneously between points 3 and 5. The original iPhone video and the PIVO result are shown in the supplementary video. At each known point the devices were firmly placed to the ground truth location so that the corresponding time was easy to recognise from the stationary part in sensor recordings. The estimated PIVO trajectory was aligned onto the floorplan image by fitting a least-squares 2D rotation and translation that maps the seven points of the trajectory to the floor plan. The residual RMS distance to the known points was 0.74 meters after fitting and the result is illustrated in Figure 4. Due to the occlusion the tracking of the Tango device failed completely between points 4 and 5 as illustrated in Figure 4. Thus, the Tango trajectory was aligned onto the floorplan image using just the first four points before divergence.

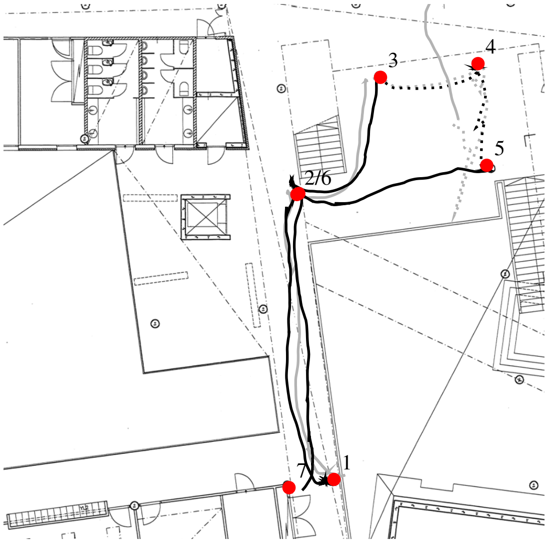


Figure 4: Occlusion experiment: A motion sequence through seven known points (●) was recorded with an iPhone 6 and Tango device so that the cameras were occluded between points 3 and 5 (dashed lines). The trajectories estimated by the Tango device (——) and our PIVO method (——) from iPhone data are shown. Tracking on the Tango device fails due to occlusion but the PIVO path is still accurate (RMS error 0.74 meters, also confirmed by visual inspection).

Without occlusions, Tango and PIVO performances are close to each other. In general, the fisheye lens in the Tango gives it an upper hand, but with enough movement for exciting the sensors with the narrow field of view in the iPhone, PIVO keeps up. We performed an experiment where a set of seven ground-truth points were visited along a 68 meter path. The Tango trajectory gave an RMSE of 0.26 m, while PIVO gave 0.59 m. This suggests that comparable results to the Tango can be gained without its specialized hardware.

4.3. Tracking a smartphone through city center

Figure 1 shows a large-scale odometry example on an iPhone, where the path is started by descending from the fourth floor of an office building, then walking through city streets to a shopping mall, descending to a metro station, and finally re-surfacing. Ill-lit indoor spaces, direct sunlight, occluding crowds, passing cars, and feature-poor environments make the visual environment challenging. To better balance the camera settings between indoor and outdoor, we used a shutter speed of $1/1000$, ISO value 400, and $n_{\mathrm{a}} = 40$. The reconstructed path, about 600 m in length, is overlaid on a city map. A video demonstrating the reconstruction is included in the supplementary material[1].

[1] Also available at http://arno.solin.fi.

The supplementary video shows the original iPhone video and PIVO result for the experiment illustrated in Figure 1. The captured image data contains many challenges that usually hamper tracking, such as low-light illumination conditions, over-exposure due to direct sunlight, and moving people in front of the camera, which cause that several times during the course of the trajectory there are effectively no visual feature tracks or most of them are not compatible with the device motion.

For example, these kind of situations happen in the beginning (at 01:14 from the start of the video) and in the end of the video (at 05:35 from the start). Further, the frame rate is relatively low (10 frames per second) and there are rapid motions causing motion blur and possibly also rolling shutter effects. Despite the aforementioned challenges PIVO performs accurately and robustly during the whole path of several hundreds of meters. The visual inspection, performed after aligning the trajectory onto the map image by rigid transformation (Fig. 1), shows that the estimated trajectory matches the street map very accurately (including scale).

It should be noted that most of the solely camera-based odometry methods will not work in cases like the one illustrated in the attached video. In fact, we tried the publicly available open-source implementation of ORB-SLAM2 for several of our iPhone videos and could not get it working in such challenging cases as those shown in our attached videos. However, ORB-SLAM2 worked fine when the camera motion was less rapid and when the camera was observing mostly the same small scene during the entire trajectory (*i.e.* not moving constantly and rapidly forward to new previously unseen areas).

5. Discussion and conclusion

In this paper we have derived a method for fusing inertial odometry with visual data. These two modalities are complementary in the sense that the inertial data provides fast-sampled high-accuracy estimates in short time-windows, while the visual data provide relative info of movement between consecutive frames. Therefore the visual data is ideal for correcting long-term drifting and bias effects introduced by low-cost IMU sensors. On the other hand, the IMU data is completely occlusion and use-case invariant, while the visual data is sensitive to occlusions and feature-poor environments.

Our PIVO method is able produce impressive results in challenging visual environments with few or no features (occlusion), rapid motion, changing lighting conditions, rolling shutter, and small field-of-view. We are not aware of previous work able to do this on a standard smartphone.

There are three key factors in the presented PIVO method. First, we formulate the IMU propagation model directly in discrete-time, which gains us some efficiency. Second, the trailing pose augmentation is accomplished through a series of linear Kalman update tricks which preserve the cross-covariance between past and future poses. Third, the visual update resembles that of MSCKF [24], but as they mention at the end of their appendix: "*. . . the camera pose estimates are treated as known constants, and their covariance matrix is ignored. As a result, the minimization can be carried out very efficiently, at the expense of the optimality of the feature position estimates*". We on the other hand take all the cross-terms into account, and our scheme is optimal in the first-order EKF linearization sense. Therefore we call our method 'probabilistic'.

The sequential formulation of PIVO makes the computational complexity scale linearly with respect to time. Computationally the IMU propagation and pose augmentation updates are extremely fast on an iPhone and negligible with respect to the total computation time. Most of the computation time is spent in feature tracking (OpenCV) and the visual update. The slowest separate part is calculating the visual update (including derivatives of Gauss-Newton). Benchmarks (on a MacBook laptop) of our implementation show that calculating the visual update takes on average 0.079 s (for a visual track of 40 frames) or 0.0052 s (for a track of 20 frames). The algorithm is capable of running on, for example, any recent iPhone.

A computational detail of the PIVO method is that in terms of the last augmented pose in the state, our model gives the same result as a fixed-lag extended RTS smoother would give: the pose estimate conditioned on all the data seen up to the most recent augmented pose (see, *e.g.*, [16] for smoothing methods for visual odometry).

In the scope of this paper, we only consider sequential odometry, and the methodology could be extended with loop-closures which would bring it closer to recently published SLAM methods, and help with slow crawling. On the other hand, SLAM methods are only beneficial if the same location is re-visited several times, and in cases such as Figure 1 SLAM would not bring any added value. The current sequential formulation of PIVO also means that computational complexity does not grow over time. Furthermore, visual-heavy methods perform poorly on narrow field-of-view cameras such as the one in iPhones. Also the PIVO method could be extended to perform even better on low-cost camera hardware—*e.g.* by taking rolling-shutter effects into account. Other future directions include delayed pose disposal for loop-closures and fusing additional data (barometric height, GPS locations) with the model.

Acknowledgements

Authors acknowledge funding from the Academy of Finland (grant numbers 277685, 295081, 308640, 310325).

References

[1] Y. Bar-Shalom, X.-R. Li, and T. Kirubarajan. *Estimation with Applications to Tracking and Navigation*. Wiley-Interscience, New York, 2001. 3, 5

[2] M. Blösch, S. Omari, M. Hutter, and R. Siegwart. Robust visual inertial odometry using a direct EKF-based approach. In *Proceedings of the International Conference on Intelligent Robots and Systems (IROS)*, pages 298–304, Hamburg, Germany, 2015. 2

[3] J.-Y. Bouguet. Pyramidal implementation of the affine Lucas Kanade feature tracker: Description of the algorithm. Technical report, Intel Corporation, 2001. 4

[4] M. Burri, J. Nikolic, P. Gohl, T. Schneider, J. Rehder, S. Omari, M. W. Achtelik, and R. Siegwart. The EuRoC micro aerial vehicle datasets. *International Journal of Robotics Research*, 35:1157–1163, 2016. 6

[5] J. Engel, T. Schöps, and D. Cremers. LSD-SLAM: Large-scale direct monocular SLAM. In *Proceedings of European Conference on Computer Vision (ECCV)*, pages 834–849, Zurich, Switzerland, 2014. 2

[6] J. Engel, J. Stückler, and D. Cremers. Large-scale direct SLAM with stereo cameras. In *Proceedings of the International Conference on Intelligent Robots and Systems (IROS)*, pages 1935–1942, Hamburg, Germany, 2015. 2

[7] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza. On-manifold preintegration for real-time visual–inertial odometry. *Transactions on Robotics*, 33(1):1–21, 2017. 2

[8] C. Forster, M. Pizzoli, and D. Scaramuzza. SVO: Fast semi-direct monocular visual odometry. In *Proceedings of the International Conference on Robotics and Automation (ICRA)*, pages 15–22, Hong Kong, China, 2014. 2

[9] C. Forster, Z. Zhang, M. Gassner, M. Werlberger, and D. Scaramuzza. SVO: Semidirect visual odometry for monocular and multicamera systems. *Transactions on Robotics*, 33(2):249–265, 2017. 2

[10] J. Heikkila and O. Silven. A four-step camera calibration procedure with implicit image correction. In *Proceedings of the Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1106–1112, San Juan, Puerto Rico, 1997. 5

[11] L. Heng and B. Choi. Semi-direct visual odometry for a fisheye-stereo camera. In *Proceedings of International Conference on Intelligent Robots and Systems (IROS)*, pages 4077–4084, Daejeon, Korea, 2016. 2

[12] D. Herrera, K. Kim, J. Kannala, K. Pulli, and J. Heikkilä. DT-SLAM: Deferred triangulation for robust SLAM. In *International Conference on 3D Vision (3DV)*, pages 609–616, Tokyo, Japan, 2014. 2

[13] J. A. Hesch, D. G. Kottas, S. L. Bowman, and S. I. Roumeliotis. Consistency analysis and improvement of vision-aided inertial navigation. *Transactions on Robotics*, 30(1):158–176, 2014. 2

[14] S. Izadi, R. A. Newcombe, D. Kim, O. Hilliges, D. Molyneaux, S. Hodges, P. Kohli, J. Shotton, A. J. Davison, and A. W. Fitzgibbon. KinectFusion: Real-time dynamic 3D surface reconstruction and interaction. In *Proceedings of the International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*, page 23, Vancouver, BC, Canada, 2011. 2

[15] A. H. Jazwinski. *Stochastic Processes and Filtering Theory*. Academic Press, New York, 1970. 3

[16] M. Kaess, H. Johannsson, R. Roberts, V. Ila, J. J. Leonard, and F. Dellaert. iSAM2: Incremental smoothing and mapping using the Bayes tree. *International Journal of Robotics Research*, 31(2):216–235, 2012. 8

[17] C. Kerl, J. Stückler, and D. Cremers. Dense continuous-time tracking and mapping with rolling shutter RGB-D cameras. In *Proceedings of the International Conference on Computer Vision (ICCV)*, pages 2264–2272, Santiago, Chile, 2015. 2

[18] G. Klein and D. W. Murray. Parallel tracking and mapping on a camera phone. In *International Symposium on Mixed and Augmented Reality (ISMAR)*, Science & Technology Proceedings, pages 83–86, Orlando, FL, USA, 2009. 2, 3

[19] N. Krombach, D. Droeschel, and S. Behnke. Combining feature-based and direct methods for semi-dense real-time stereo visual odometry. In *International Conference on Intelligent Autonomous Systems (IAS)*, pages 855–868, Shanghai, China, 2016. 6

[20] S. Leutenegger, S. Lynen, M. Bosse, R. Siegwart, and P. Furgale. Keyframe-based visual–inertial odometry using nonlinear optimization. *International Journal of Robotics Research*, 34(3):314–334, 2015. 2

[21] M. Li, B. Kim, and A. I. Mourikis. Real-time motion tracking on a cellphone using inertial sensing and a rolling shutter camera. In *Proceedings of the International Conference on Robotics and Automation (ICRA)*, pages 4712–4719, Karlsruhe, Germany, 2013. 3

[22] B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of the International Conference on Artificial Intelligence (IJCAI)*, pages 674–679. Vancouver, BC, Canada, 1981. 4

[23] J. M. M. Montiel, J. Civera, and A. J. Davison. Unified inverse depth parametrization for monocular SLAM. In *Proceedings of Robotics: Science and Systems*, Philadelphia, USA, 2006. 5

[24] A. I. Mourikis and S. I. Roumeliotis. A multi-state constraint Kalman filter for vision-aided inertial navigation. In *Proceedings of the International Conference on Robotics and Automation (ICRA)*, pages 3565–3572, Rome, Italy, 2007. 2, 4, 5, 8

[25] R. Mur-Artal and J. D. Tardós. ORB-SLAM2: An open-source SLAM system for monocular, stereo and RGB-D cameras. *arXiv preprint arXiv:1610.06475*, 2016. 2, 6

[26] R. Mur-Artal and J. D. Tardós. Visual-inertial monocular SLAM with map reuse. *Robotics and Automation Letters*, 2(2):796–803, 2017. 2, 6

[27] R. A. Newcombe, D. Fox, and S. M. Seitz. DynamicFusion: Reconstruction and tracking of non-rigid scenes in real-time. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 343–352, Boston, MA, USA, 2015. 2

[28] D. Sachs. Sensor fusion on Android devices: A revolution in motion processing. https://www.youtube.com/watch?v=C7JQ7Rpwn2k, 2010. [Online; accessed 17-March-2017]. 1

[29] S. Särkkä. *Bayesian Filtering and Smoothing*. Institute of Mathematical Statistics Textbooks. Cambridge University Press, 2013. 3

[30] S. Särkkä and A. Solin. On continuous-discrete cubature Kalman filtering. In *Proceedings of SYSID 2012, 16th IFAC Symposium on System Identification*, pages 1210–1215, Brussels, Belgium, 2012. 3

[31] J. Shi and C. Tomasi. Good features to track. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 593–600, 1994. 4

[32] A. Solin, S. Cortes, E. Rahtu, and J. Kannala. Inertial odometry on handheld smartphones. *arXiv preprint arXiv:1703.00154*, 2017. 3, 4

[33] H. Strasdat, J. M. M. Montiel, and A. J. Davison. Scale drift-aware large scale monocular SLAM. In *Proceedings of Robotics: Science and Systems*, Zaragoza, Spain, 2010. 2

[34] P. Tanskanen, K. Kolev, L. Meier, F. Camposeco, O. Saurer, and M. Pollefeys. Live metric 3D reconstruction on mobile phones. In *Proceedings of the International Conference on Computer Vision (ICCV)*, pages 65–72, Sydney, Australia, 2013. 3

[35] P. Tanskanen, T. Naegeli, M. Pollefeys, and O. Hilliges. Semi-direct EKF-based monocular visual-inertial odometry. In *Proceedings of the International Conference on Intelligent Robots and Systems (IROS)*, pages 6073–6078, Hamburg, Germany, 2015. 2

[36] D. H. Titterton and J. L. Weston. *Strapdown Inertial Navigation Technology*. The Institution of Electrical Engineers, 2004. 3

[37] V. Usenko, J. Engel, J. Stückler, and D. Cremers. Direct visual-inertial odometry with stereo cameras. In *Proceedings of the International Conference on Robotics and Automation (ICRA)*, pages 1885–1892, Stockholm, Sweden, 2016. 2, 6