2011 IEEE International Conference on Robotics and Automation
Shanghai International Conference Center
May 9-13, 2011, Shanghai, China

Real-Time Metric State Estimation for Modular Vision-Inertial Systems

Stephan Weiss and Roland Siegwart

Abstract— **Single camera solutions - such as monocular visual odometry or monoSLAM approaches - found a wide echo in the community. All the monocular approaches, however, suffer from the lack of metric scale. In this paper, we present a solution to tackle this issue by adding an inertial sensor equipped with a three-axis accelerometer and gyroscope. In contrast to previous approaches, our solution is independent of the underlying vision algorithm which estimates the camera poses. As a direct consequence, the algorithm presented here operates at a constant computational complexity in real time. We treat the visual framework as a black box and thus the approach is modular and widely applicable to existing monocular solutions. It can be used with any pose estimation algorithm such as visual odometry, visual SLAM, monocular or stereo setups or even GPS solutions with gravity and compass attitude estimation. In this paper, we show the thorough development of the metric state estimation based on an Extended Kalman Filter. Furthermore, even though we treat the visual framework as a black box, we show how to detect failures and estimate drifts in it. We implement our solution on a monocular vision pose estimation framework and show the results both in simulation and on real data.**

I. Introduction

Nowadays the camera pose estimation topic is well studied in the community and a variety of ready-to-use solutions are available. In recent years the community launched several visual odometry and visual SLAM frameworks to avoid bulky sensor suites like laser scanners. These visual solutions carry a very lightweight, cheap and yet powerful sensor suite and are still capable of running in real time. In [1] and [2] the authors reduced the sensor suite to one single camera.

One camera is probably the only viable sensor solution for devices and robots with a very limited weight and power budget. We aim in particular at applications on small flying platforms. In previous work [3] we implemented a visual position controller on a quadrotored rotorcraft based on the visual framework PTAM [1]. This showed the possibility to control a robot in its 6 Degrees of Freedom (DoF) by only using one single camera. The drawback was that every time before take-off the visual scale had to be manually measured since one camera cannot recover the metric scale of the scene. Moreover, a scale drift, while the robot was flying, eventually led to instability and crash of the system. Thus, in order to use a monocular solution for controlling a robot, it is crucial to recover the absolute scale and scale drift in real time. This information immediately leads to an absolute position and also velocity estimation of the robot. Then, well known position and velocity based controllers may be applied to robustly control the vehicle.

The research leading to these results has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement n. 231855 (sFly). Stephan Weiss is currently PhD student at the ETH Zurich (email: stephan.weiss@mavt.ethz.ch). Roland Siegwart is full professor at the ETH Zurich and head of the Autonomous Systems Lab (email: r.siegwart@ieee.org).

II. Related Work

Fusing Inertial Measurement Units (IMUs) with a visual sensor is a well known topic in the community. Recent work in [4] shows the fusion of IMU and vision in its 6DoF on a robot arm. The authors compare the performance of an Extended Kalman Filter (EKF) and an Unscented Kalman Filter (UKF). The scale is recovered from the known dimensions of the observed patterns in the image frames. Thus the working area is limited to the area the pattern can be observed. The authors state that the UKF approach yields better results at the cost of calculation power. We set a high priority on speed of the algorithm as we aim at real time full 6DoF state estimation. Thus we implemented our solution in an EKF framework only.

The work of Mirzaei and Roumeliotis [5] presents a Kalman Filter (KF) based calibration solution to estimate rotation and translation between a camera and an inertial sensor mounted on a rigid body. They retrieve the scale from a feature pattern with known dimension. In contrast to their work, however, we focus also on the unknown scale.

A similar approach for camera-IMU calibration was chosen by Kelly et al. [6]. The authors present a target free calibration method where the visual scale to arbitrary but fixed features is estimated during the calibration procedure. As the procedure requires fix features, the operation area is restricted to the area of sight of the features. The same spirit has the work of Jones [7] and Pinies [8]. The authors coupled tightly the IMU into an EKF SLAM. This is in essence a similar approach as Kelly followed but with the extension to new features. Strelow has already earlier shown a detailed analysis in this direction [9]. From the filter point of view these approaches are more robust compared to our solution as they take all cross couplings of the observed features into account. However, it is known that the computational cost of EKF SLAM is $O(m^2)$ at its theoretical best, with m being the amount of features. This issue prevents long term runs or requires sub-mapping at the cost of cross couplings. Furthermore in [10] the authors state the need of many features for a robust pose estimations and favor, thus, a keyframe-based SLAM approach in most scenarios. The latter approach runs linear in the number of features (navigating in a previously visited area).

In contrast to the tightly coupled EKF-SLAM approaches using an IMU (that we call here IMU-EKF-SLAM), our solution decouples the two frameworks visual pose estimate and metric scaled state estimation. Figure 1 depicts the two different approaches. The novelty of our solution is that the metric state estimation is independent of the visual framework. More precisely, our work is independent of the visual pose estimation algorithm and is, thus, not limited to SLAM. Also visual odometry, stereo setups, or even GPS (unit scale, magnetometer, and gravity for attitude) can be used as pose input for the filter. This makes our approach a very versatile solution to control a robot efficiently and at high frequency in metric position and speed. Of course, due to the absence of the landmarks in the filter, we lose cross-coupling effects in the covariance matrix to a certain amount. We show here, however, that even without these cross-couplings the filter yields precise results while having the benefit of constant computational complexity. Using, for example, a keyframe-based visual framework the whole state estimation framework - visual pose and our filter - has a complexity of $O(m)$ instead of $O(m^2)$ as in the IMU-EKF-SLAM proposed in other work. As we treat the visual framework as a black box, a second contribution of this paper is to determine when the visual pose estimator fails and also to estimate the corresponding drifts.

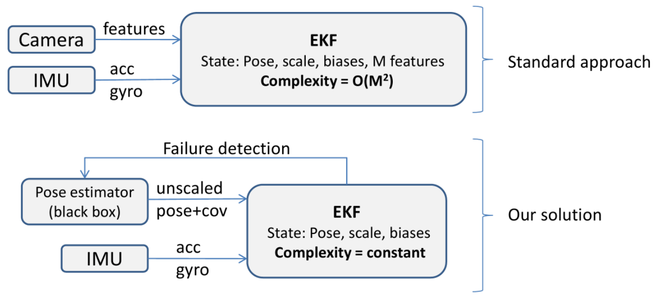

Fig. 1. In contrast to tightly coupled solutions (top schematic) our filter runs at constant computational complexity. Moreover, treating the pose estimation part as a black box, we are independent of the underlying pose estimation method. The whole framework has thus the complexity of the chosen pose estimator. As we show in this work, we can still detect failures and drifts of the black box.

The paper is structured as follows. In the main section III we explain first the noise models for the Extended Kalman Filter, then we give a thorough explanation of the state, of its discretized propagation and finally of its update. In section IV we describe how to detect failure modes and drifts of the black box visual part. Section V is dedicated to the results we obtained in simulation and with real data. We conclude the paper in section VI.

We use standard notation, in particular, we speak about the units of a variable a when writing it in brackets $[a]$. The measured accelerations a_m for example have thus $[a_m] = \frac{m}{s^2}$. Also, the skew symmetric matrix of a is $\lfloor a_x \rfloor$.

III. Extended Kalman Filter

The inertial sensor yields the acceleration and rotational velocity in 3 axes. The visual sensor provides the filter with unscaled 3D position and scale free attitude estimations with respect to a visual frame set at the initialization. Figure 2 shows the situation of the camera-IMU setup with its corresponding coordinate frames.

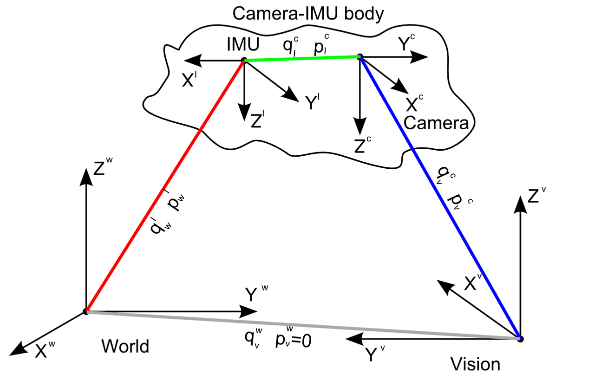

Fig. 2. Visualization of the different coordinate frames in the setup. Between every frame there is a rotation q and translation p. Constant values are in green such as the IMU-cam transformation. Red values denote the variables used for robot control. Blue values are the measurements and in grey we denote the short term stationary transformations.

A. Inertial Sensor Model

We assume that the inertial measurements contain a certain bias b and white Gaussian noise n. Thus, for the real angular velocities ω and the real accelerations a we have

$$\omega = \omega_m - b_\omega - n_\omega \qquad a = a_m - b_a - n_a \tag{1}$$

the subscript m denotes the measured value and the dynamics of the non-static bias b are modeled as a random process

$$\dot{b_\omega} = n_{b_\omega} \qquad \dot{b_a} = n_{b_a} \tag{2}$$

B. State Representation

The state of the filter is composed of p_w^i the position of the IMU in the inertial world frame, its velocity v_w^i and its attitude quaternion q_w^i describing a rotation from the inertial frame to the IMU frame. We add also the gyro and acceleration biases b_ω and b_a as well as the visual scale factor λ. The calibration states are the rotation from the IMU frame to the camera frame q_i^c and the distance between these two sensors p_i^c. We note that the calibration states can be omitted and set to a calibrated constant making the filter more robust [7]. For completeness we keep them here in the filter state. This yields a 24-elements state vector X:

$$X = \{p_w^{i^T} \; v_w^{i^T} \; q_w^{i^T} \; b_\omega^T \; b_a^T \; \lambda \; p_i^c \; q_i^c\} \tag{3}$$

The following differential equations govern the state:

$$\begin{aligned} \dot{p}_w^i &= v_w^i && (4)\\ \dot{v}_w^i &= C_{(q_w^i)}^T(a_m - b_a - n_a) - g && (5)\\ \dot{q}_w^i &= \frac{1}{2}\Omega(\omega_m - b_\omega - n_\omega)q_w^i && (6) \end{aligned}$$

$$\dot{b}_\omega = n_{b_\omega} \quad \dot{b}_a = n_{b_a} \quad \dot{\lambda} = 0 \quad \dot{p}_i^c = 0 \quad \dot{q}_i^c = 0 \tag{7}$$

With g as the gravity vector in the world frame and $\Omega(\omega)$ as the quaternion multiplication matrix of ω. We assume the scale to drift very slowly, thus $\dot{\lambda} = 0$. Taking the expectations of the above derivatives and the beforehand discussed noise model for the filter state propagation we obtain

$$\begin{aligned} \dot{\hat{p}}_w^i &= \hat{v}_w^i & (8)\\ \dot{\hat{v}}_w^i &= C^T_{(\hat{q}_w^i)}(a_m - \hat{b}_a) - g & (9)\\ \dot{\hat{q}}_w^i &= \frac{1}{2}\Omega(\omega_m - \hat{b}_\omega)\hat{q}_w^i & (10)\end{aligned}$$

$$\dot{\hat{b}}_\omega = 0 \quad \dot{\hat{b}}_a = 0 \quad \dot{\hat{\lambda}} = 0 \quad \dot{\hat{p}}_i^c = 0 \quad \dot{\hat{q}}_i^c = 0 \tag{11}$$

With $C_{(q)}$ as the rotational matrix corresponding to the quaternion q.

C. Error State Representation

In the above described state representation, we use quaternions as attitude description. It is common that in such a case we represent the error and its covariance not in terms of an arithmetic difference but with the aid of an error quaternion. This increases numerical stability and handles the quaternion in its minimal representation [11].

Thus, we define the 22-elements error state vector

$$\tilde{x} = \{\Delta p_w^{i^T} \ \Delta v_w^{i^T} \ \delta\theta_w^{i^T} \ \Delta b_\omega^T \ \Delta b_a^T \ \Delta\lambda \ \Delta p_i^{c^T} \ \delta\theta_i^{c^T}\} \tag{12}$$

as the difference of an estimate $\hat{x}$ to its quantity x, i.e. $\tilde{x} = x - \hat{x}$. We apply this to all state variables except the error quaternions which are defined as follows

$$\begin{aligned} \delta q_w^i &= q_w^i \otimes \hat{q}_w^{i^{-1}} \approx [\frac{1}{2}\delta\theta_w^{i^T} \ 1]^T & (13)\\ \delta q_i^c &= q_i^c \otimes \hat{q}_i^{c^{-1}} \approx [\frac{1}{2}\delta\theta_i^{c^T} \ 1]^T & (14)\end{aligned}$$

The differential equations for the continuous time error state are

$$\begin{aligned} \Delta\dot{p}_w^i &= \Delta v_w^i & (15)\\ \Delta\dot{v}_w^i &= -C^T_{(\hat{q}_w^i)}\lfloor \hat{a}_x \rfloor \delta\theta - C^T_{(\hat{q}_w^i)}\Delta b_a - C^T_{(\hat{q}_w^i)} n_a & (16)\\ \delta\dot{\theta}_w^i &= -\lfloor \hat{\omega}_x \rfloor \delta\theta - \Delta b_\omega - n_\omega & (17)\end{aligned}$$

$$\Delta\dot{b}_\omega = n_{b_\omega} \quad \Delta\dot{b}_a = n_{b_a} \quad \Delta\dot{\lambda} = 0 \quad \Delta\dot{p}_c^i = 0 \quad \Delta\dot{\theta}_i^c = 0 \tag{18}$$

with $\hat{\omega} = \omega_m - \hat{b_\omega}$ and $\hat{a} = a_m - \hat{b_a}$. This can be summarized to the linearized continuous-time error state equation

$$\dot{\tilde{x}} = F_c\tilde{x} + G_c n \tag{19}$$

with n being the noise vector $n = [n_a^T, n_{b_a}^T, n_\omega^T, n_{b_\omega}^T]^T$. In the here presented solution, we are in particular interested in the speed of the algorithm. This is why we assume F_c and G_c to be constant over the integration time step between two consecutive state propagations. For the discretization we may therefore write

$$F_d = \exp(F_c\Delta t) = \mathbf{I_d} + F_c\Delta t + \frac{1}{2!}F_c^2\Delta t^2 + \ldots \tag{20}$$

Careful analysis of the matrix exponents reveals a repetitive and sparse structure. This allows us to express F_d exact without approximation and to use sparse matrix operations later in the implementation of the filter. The matrix F_d has the following structure

$$F_d = \begin{bmatrix} \mathbf{I_{d_3}} & \Delta t & A & B & -C^T_{(\hat{q}_w^i)}\frac{\Delta t^2}{2} & \mathbf{0_{3x7}} \\ \mathbf{0_3} & \mathbf{I_{d_3}} & C & D & -C^T_{(\hat{q}_w^i)}\Delta t & \mathbf{0_{3x7}} \\ \mathbf{0_3} & \mathbf{0_3} & E & F & \mathbf{0_3} & \mathbf{0_{3x7}} \\ \mathbf{0_3} & \mathbf{0_3} & \mathbf{0_3} & \mathbf{I_{d_3}} & \mathbf{0_3} & \mathbf{0_{3x7}} \\ \mathbf{0_3} & \mathbf{0_3} & \mathbf{0_3} & \mathbf{0_3} & \mathbf{I_{d_3}} & \mathbf{0_{3x7}} \\ \mathbf{0_{7x3}} & \mathbf{0_{7x3}} & \mathbf{0_{7x3}} & \mathbf{0_{7x3}} & \mathbf{0_{7x3}} & \mathbf{I_{d_7}} \end{bmatrix}$$

We use the small angle approximation for which $|\omega| \to 0$, apply de l'Hopital rule and obtain for the six matrix blocks A, B, C, D, E, F a compact solution [11]:

$$\begin{aligned} A &= -C^T_{(\hat{q}_w^i)}\lfloor \hat{a}_x \rfloor \left(\frac{\Delta t^2}{2} - \frac{\Delta t^3}{3!}\lfloor \omega_x \rfloor + \frac{\Delta t^4}{4!}\lfloor \omega_x \rfloor^2\right)\\ B &= -C^T_{(\hat{q}_w^i)}\lfloor \hat{a}_x \rfloor \left(\frac{-\Delta t^3}{3!} + \frac{\Delta t^4}{4!}\lfloor \omega_x \rfloor - \frac{\Delta t^5}{5!}\lfloor \omega_x \rfloor^2\right)\\ C &= -C^T_{(\hat{q}_w^i)}\lfloor \hat{a}_x \rfloor \left(\Delta t - \frac{\Delta t^2}{2!}\lfloor \omega_x \rfloor + \frac{\Delta t^3}{3!}\lfloor \omega_x \rfloor^2\right)\\ D &= -A\\ E &= \mathbf{I_d} - \Delta t\lfloor \omega_x \rfloor + \frac{\Delta t^2}{2!}\lfloor \omega_x \rfloor^2\\ F &= -\Delta t + \frac{\Delta t^2}{2!}\lfloor \omega_x \rfloor - \frac{\Delta t^3}{3!}\lfloor \omega_x \rfloor^2 \end{aligned}$$

We can now derive the discrete time covariance matrix Q_d [12] as

$$Q_d = \int_{\Delta t} F_d(\tau)G_cQ_cG_c^T F_d(\tau)^T d\tau \tag{21}$$

with Q_c being the continuous time system noise covariance matrix $Q_c = \mathrm{diag}(\sigma^2_{n_a}, \sigma^2_{n_{b_a}}, \sigma^2_{n_\omega}, \sigma^2_{n_{b_\omega}})$

With the discretized error state propagation and error process noise covariance matrices we can propagate the state as follows:

1) propagate the state variables according to their differential equations (8) to (11). For the quaternion, we used the first order integration described in [11].
2) calculate F_d and Q_d
3) compute the propagated state covariance matrix according to the filter equation

$$P_{k+1|k} = F_dP_{k|k}F_d^T + Q_d \tag{22}$$

D. Update

1) Covariances: From [13] and [14] we assume to be able to obtain the uncertainties of the camera translation and rotation axis and magnitude using one of the described methods. We distinguish between the position $\mathbf{n_p}$ and rotational $\mathbf{n_q}$ measurement noise. This yields the six-vector

$$n_m = [n_p \; n_q]^T \tag{23}$$

with its measurement covariance matrix R.

E. Measurement Model

1) Measurement: For the camera position measurement p_v^c we obtain from the visual algorithm, we have the following measurement model

$$z_p = p_v^c = C^T_{(q_v^w)}(p_w^i + C^T_{(q_w^i)} p_i^c)\lambda + n_p \tag{24}$$

with $C_{(q_w^i)}$ as the IMU's attitude and $C_{(q_v^w)}$ the rotation from the visual frame to the inertial world frame.

We define the position error as

$$\begin{aligned} \tilde{z}_p &= z_p - \hat{z}_p \\ &= C^T_{(q_v^w)}(p_w^i + C^T_{(q_w^i)} p_i^c)\lambda + n_p - \\ &\quad C^T_{(q_v^w)}(\hat{p}_w^i + C^T_{(\hat{q}_w^i)} \hat{p}_i^c)\hat{\lambda} \end{aligned} \tag{25}$$

which can be linearized to

$$\tilde{z}_{p_l} = H_p \tilde{x} \tag{26}$$

with

$$H_p^T = \begin{bmatrix} C^T_{(q_v^w)}\hat{\lambda} \\ \mathbf{0_3} \\ -C^T_{(q_v^w)} C^T_{(\hat{q}_w^i)} \left\lfloor \hat{p^c_{i_x}} \right\rfloor \hat{\lambda} \\ \mathbf{0_3} \\ \mathbf{0_3} \\ C^T_{(q_v^w)} C^T_{(\hat{q}_w^i)} \hat{p_i^c} + C^T_{(q_v^w)} \hat{p} \\ C^T_{(q_v^w)} C^T_{(\hat{q}_w^i)} \hat{\lambda} \\ \mathbf{0_3} \end{bmatrix}$$

using the definition of the error quaternion

$$\begin{aligned} q_w^i &= \delta q_w^i \otimes \hat{q}_w^i && (27) \\ C_{(q_w^i)} &= C_{(q_{\hat{i}}^i)} C_{(q_w^{\hat{i}})} && (28) \\ C_{(q_{\hat{i}}^i)} &\approx \mathbf{I_d} - \left\lfloor \delta\theta^{\mathbf{i}}_{\mathbf{w_x}} \right\rfloor && (29) \end{aligned}$$

For the rotation measurement we again apply the notion of an error quaternion. The vision algorithm yields the rotation from the vision frame to the camera frame q_v^c. We can model this as

$$z_q = q_v^c = q_i^c \otimes q_w^i \otimes q_v^w \tag{30}$$

which yields for the error measurement

$$\begin{aligned} \tilde{z}_q &= z_q - \hat{z}_q \\ &= q_i^c \otimes q_w^i \otimes q_v^w \otimes (q_i^c \otimes \hat{q}_w^i \otimes q_v^w)^{-1} \\ &= \delta q_i^c \otimes \hat{q}_i^c \otimes \delta q_w^i \otimes {q_i^c}^{-1} \\ &= H_q^{wi} \delta q_w^i = H_q^{ic} \delta q_i^c \end{aligned} \tag{31}$$

Where H_q^{wi} and H_q^{ic} are the matrices when the error measurement is linearized versus the filter error states δq_w^i and δq_i^c respectively. Finally the measurements can be stacked together as

$$\begin{aligned} \begin{bmatrix} \tilde{z}_p \\ \tilde{z}_q \end{bmatrix} &= \begin{bmatrix} H_p \\ \mathbf{0_{3x6}} \; \tilde{H}_q^{wi} \; \mathbf{0_{3x10}} \; \tilde{H}_q^{ic} \end{bmatrix} \tilde{x} \\ \tilde{\mathbf{z}} &= \mathbf{H}\tilde{\mathbf{x}} \end{aligned} \tag{32}$$

Using the approximation of $H_q^{xy} = \begin{pmatrix} 1 & 0_{1x3} \\ 0_{3x1} & \tilde{H}_q^{xy} \end{pmatrix}$. This is justified since the expectation of the error quaternions δq_w^i and δq_i^c are unit quaternions.

2) Update: Once we obtained the measurement matrix $\mathbf{H}$ we can update our estimate according to the well known Kalman Filter procedure:

1) compute the residual $\tilde{z} = z - \hat{z}$
2) compute the innovation $S = HPH^T + R$
3) compute the Kalman gain $K = PH^T S^{-1}$
4) compute the correction $\hat{\tilde{x}} = K\tilde{z}$

From the correction $\hat{\tilde{x}}$ we can compute the updated state variables in our state X. The error quaternion is calculated by following eq. (13) and ensuring its unit length. The error state covariance is updated as following

$$P_{k+1|k+1} = (\mathbf{I_d} - KH)P_{k+1|k}(\mathbf{I_d} - KH)^T + KRK^T \tag{33}$$

IV. Drift and False Pose Estimates

Unlike tightly coupled approaches with our solution, we can treat the visual part (i.e. the visual pose estimation) as a black box. Of course, one desires to detect failures and drifts of the black box to ensure ongoing functionality of the whole framework. The scale drift is handled by the scale estimate in real time. We explain the detection of failures of the vision part in the following.

We note that in section III we considered the rotation between the inertial world frame and the frame of the visual framework q_v^w as constant during a filter step. At each filter step k we can measure this rotation as

$$q_v^w(k) = {\hat{q}_w^i}^{-1}(k) \otimes {\hat{q}_i^c}^{-1}(k) \otimes q_v^c(k) \tag{34}$$

Since the drift is slow compared to the filter frequency, we can smooth a sequence of measurements of $q_v^w(k)$. We suggest a median filter as the vision part is better modeled with non-zero mean outlier jumps. The estimation of the rotation between inertial and visual frame $\hat{q}_v^w(k)$ using a window of size N is then

$$\hat{q}_v^w(k) = \text{med}(q_v^w(i)) \quad i = k - N \rightarrow k \tag{35}$$

Due to the fact that the drift is slow we can identify abrupt jumps in the measured orientation $q_v^w(k)$ w.r.t. the smoothed estimate $\hat{q}_v^w(k)$ as failures of the visual pose estimation. A typical plot of the measurement of $q_v^w(k)$ is shown in figure 3. The sequences where the visual part failed to estimate a correct pose are clearly visible. As soon as a measurement $q_v^w(k)$ lies outside the 3σ error bounds of the past M

estimates $\hat{q}_v^w(k)$ it is considered as false pose estimate. Note that, analyzing the rotation for wrong measurements, this is less sensitive to loop closure jumps. Usually these jumps affect greatly the position but less the attitude.

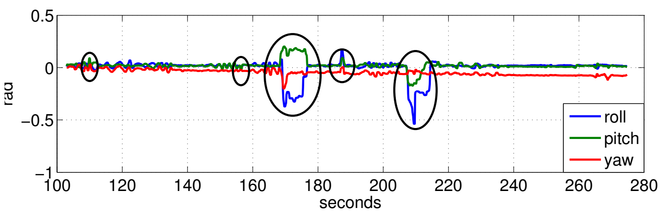

Fig. 3. Roll, pitch, yaw representation of the calculated rotation between the black box visual frame and the world frame. Note the five encircled areas which depict a failure of the vision algorithm. Even though we treat the vision algorithm as a black box we can clearly identify failures.

During such failure periods neither $\hat{q}_v^w(k)$ nor the biases and scale are updated since the measured data is corrupted. Note that we do update the IMU orientation, the velocity and position to bound the error during long dropouts. Low cost accelerometers suffer from large drifts and should not be integrated over longer periods of time without updates. In contrast, gyroscope integration is fairly robust also over longer periods of time. Thus we increase the visual measurement noise for the position much less than for the attitude. The increasing of the measurement noise during failure sequences bounds the filter error yet trusts the simple integration to a large extent. As figure 4 shows, the attitude is still estimated reliably.

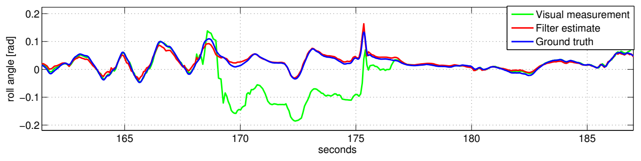

Fig. 4. Example for the roll angle estimate evolution during a failure of the black box visual framework. During a failure mode biases and scale are not updated. The measurement noise for the attitude is highly increased as gyroscopes are reliable for long term integration. The noise for the position measurement is less increased, because accelerometers suffer from larger drifts. This results in an slight erroneous position estimate of the filter but still very reliable attitude estimate.

With the above approaches we showed both, the handling of scale drift and failures of the visual part treated as a black box.

V. Results

In this section we demonstrate the use of our solution based on both simulated and real data. We use the Crossbow VG400CC IMU running at 75Hz and a uEye WVGA monochrome camera with global shutter. As a vision system black box we use the PTAM monocular SLAM framework [1]. We process images at around 20Hz. We assume to have accurate timestamps from the sensors to deal with time delays. For ground truth measurements of the real data we use a Vicon pose measurement system providing us the device's position with millimeter accuracy and attitude up to a tenth of a degree both at 200Hz.

The faster sampling rate of the IMU compared to the vision system demands for a multi rate handling in the practical implementation. We propagate the filter state and its error covariance each time an IMU measurement is obtained as described in section III. We perform an update on a vision reading. Our simulated data reflects this multi rate aspect as well as the noises given by the manufacturer of the real hardware.

A. *Simulated Data*

In simulations we evaluate the filter with respect to different hardware configurations and their error in the initialization. For each simulation run we modify one parameter and its error in the initialization and keep the others constant with correct initialization. More precisely we analyze the distance between IMU and camera p_i^c, their rotation q_i^c, the scale factor λ, and the acceleration and rotational rate biases of the IMU b_a and b_w. The default values are randomly chosen and are listed in table I. A simulation run lasted 80 simulated seconds with accelerations of max $4.5\frac{m}{s^2}$ and rotational velocities of max $\frac{\pi}{2}\frac{rad}{s}$.

TABLE I
DEFAULT SIMULATION VALUES

$p_i^c[m]$	[-1.1 0.5 1.05]^T
$q_i^c[rad]$	rpy[0.7 -1.2 0.1]^T
λ	0.5
$b_a[\frac{m}{s^2}]$	[-0.1 -0.2 0.15]^T
$b_w[\frac{rad}{s}]$	[0.01 0.02 -0.015]^T

For the distance between IMU and camera p_i^c we changed the setup between 0.1m, 0.6m and 1.1m for each single axis and then all axis together. An initial error of 10% to 100% was applied for each configuration. Figure 5 shows the error of the results of the filter compared to ground truth after 80 seconds. Note that the filter is still converging after 80 seconds for large absolute initialization errors on all three axis together. Be aware of the different y-axis scales in the plots.

For the rotation between IMU and camera q_i^c we changed the setup between 0rad, 1.5rad and 3rad for each single axis and then all axes together. An initial error of 0.1rad to 1rad was applied for each configuration since an error in percentage is not meaningful due to the circular nature of rotations. The fix offset resulted in a similar behavior for each configuration. We summarize the performance in figure 6. It shows the error of the results of the filter compared to ground truth after 80 seconds of simulation. Naturally the error increases with larger initialization errors. Note that with an initialization error of 1rad (i.e. 57.3°) after 80 seconds we only have an error of less than 2.3° in roll, pitch, and yaw.

We also altered the scales from 0.1 to 5.5 in steps of 0.5 and apply at each value initialization errors from 10% to 100% of the real value. Note that a good initial estimate of the scale is crucial to not break the filter. This my be

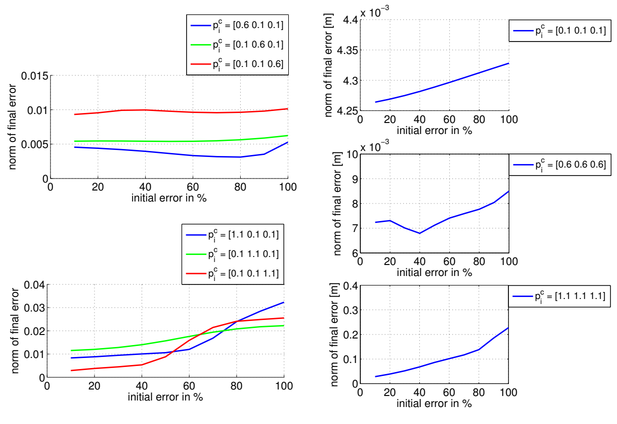

Fig. 5. The graphs read as follows: we changed the initial true setup for the IMU-camera distance p_i^c to 0.1m, 0.6m and 1.1m on each axis. We distorted the initial filter values by adding 10% to 100% of the true value. We measured the norm of the filter value $\hat{p}_i^c$ after 80 seconds of simulated filtering as performance evaluation. Graph a) depicts the performance when p_i^c has different magnitudes on the axis. In contrast to [6] the error is in the same range no matter which axis is more elongated. This is because we can move freely and no particular features need to remain in the given field of view. Graph b) shows the error with equally elongated axis of p_i^c. Note that for large initial errors the filter did not yet converge fully (i.e. bottom plot with 0.2m final error and 90% error added to the initial value. All units are in [m]

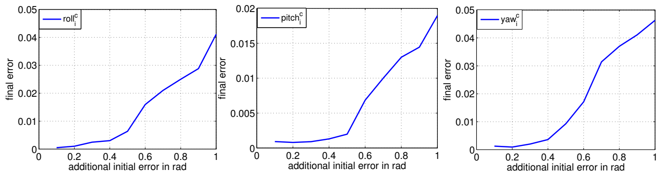

Fig. 6. The three plots show the error of the estimated rotation between IMU and camera. We run simulations for 80 seconds each and changed the initial filter state by adding from 0.1rad up to 1rad of error to the correct value.

achieved by a comparison of the visual speed with a short-period acceleration integration during initialization. Figure 7 shows the error of the estimated scale after 80 seconds of filtering time. Note that this is a single run experiment to also demonstrate outliers (i.e. no mean filtering is applied).

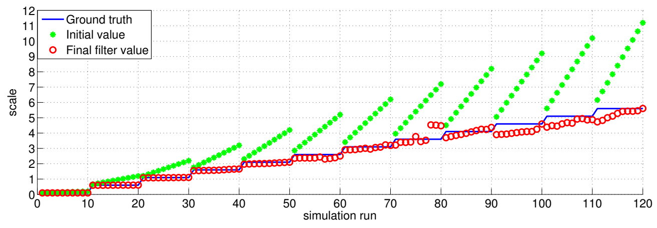

Fig. 7. The solid blue line depicts the ground truth. We added at each scale an error of 10% to 100% of the true value for the initial filter state (green stars). After 80 seconds of simulated filtering we plot the filter value for the scale in red circles. Note that this is a single run experiment. We like to draw the attention to the convergence behavior after simulation run 70. The outliers occurring every now and then after run 70 underline that a good initial guess of the scale is important.

Finally, we analyzed the initialization errors of the acceleration and angular rate biases. We used the true values 0, 1.5 and 3 $\frac{m}{s^2}$ for the accelerometer and 0.1, 0.6 and 1.1 $\frac{rad}{s}$ for the gyro biases. For each value we tested the filter adding an initial error of 10% to 100% of the true value. We did not experience any different convergence behaviors by distorting only one axis or all three axes together. We evaluate the filter as fairly insensitive to wrong initial bias estimates. The resulting estimates after 80 seconds are summarized in table II.

TABLE II

SIMULATED FILTER RESULTS ON DIFFERENT BIAS VALUES

	b_a	b_a	b_a
Ground truth	$0\frac{m}{s^2}$	$1.5\frac{m}{s^2}$	$3\frac{m}{s^2}$
Absolute error	$0.0024\frac{m}{s^2}$	$0.005\frac{m}{s^2}$	$0.02\frac{m}{s^2}$
	b_ω	b_ω	b_ω
Ground truth	$0.1\frac{rad}{s}$	$0.6\frac{rad}{s}$	$1.1\frac{rad}{s}$
Absolute error	$0.0006\frac{rad}{s}$	$0.0008\frac{rad}{s}$	$0.0009\frac{rad}{s}$

B. Real data

Using the results of the above simulations we ensured that the filter is robust enough against some initialization errors. We mounted a camera on an IMU as described in the beginning of this section and performed hand-held movements. The maximal excitations in accelerations are $4\frac{m}{s^2}$ in z-direction, $3\frac{m}{s^2}$ in y-direction and below $1\frac{m}{s^2}$ in x-direction. In angular velocity the maximal excitations are about $2\frac{rad}{s}$ in all directions. Note that this is less excitation than the values used in the simulations. The results are summarized in table III. Note that the estimated scale has an error of just under 2% compared to the calculated scale aid of ground truth data.

TABLE III

FILTER RESULTS ON REAL DATA

	p_i^c xyz[m]	q_i^c rpy[rad]	Scale
Filter	-0.039 -0.026 0.018	-0.038 0.072 3.115	0.774
Measured	-0.0381 -0.0381 0.02	0 0 π	0.789

The novelty of this work is also to have the full metric scaled state in real time at constant calculation power (from the filter point of view). We evaluate first the RMS of the position. We analyzed the case where the estimated position is rescaled with the true scale $\lambda = 0.789$ to show errors in the pure position estimate. Next we analyze the same using the scale estimated by the filter at the end $\hat{\lambda}_{final} = 0.774$ analyzing the filter's position error after convergence. Last we compute the RMS of the filter's position estimate $\hat{p}(t)$ and the true position taking also the evolution of the scale estimate $\hat{\lambda}(t)$ into account. Table IV summarizes the results. In the same table we do the same calculations for the estimated velocity.

In our work, we treat the vision part as a black box providing the unscaled 6DoF pose only. We are, however,

TABLE IV
FILTER RESULTS ON REAL DATA

RMS	p_x[m]	p_y[m]	p_z[m]	v_x[m]	v_y[m]	v_z[m]
λ	0.0445	0.0604	0.0708	0.0680	0.0731	0.0794
$\hat{\lambda}_{final}$	0.0373	0.0622	0.0755	0.0689	0.0740	0.0806
$\hat{\lambda}(t)$	0.1115	0.1005	0.1239	0.0694	0.0743	0.0812

able to detect failures as figure 8 shows. Failure modes are correctly detected at seconds 109, 156, 168-176, 187 and 207-214 using the algorithm explained in section IV.

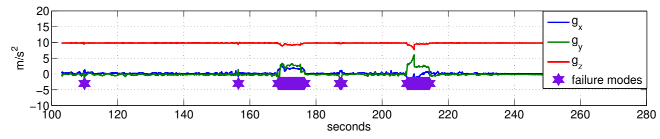

Fig. 8. In real data, failure modes of the black box visual pose estimator are correctly detected. For better visualization, the gravity vector is rotated from the world frame to the estimated visual frame. Abrupt changes (i.e. failures) in the visual pose estimate result in similar abrupt changes in the rotation between visual and world frame q_v^w. These changes are detected according to section IV. Visual failures are marked in magenta stars. Note that the failure sections may be several seconds long, pure sensor integration is thus not feasible. Short dropouts usually reflect bundle adjustment steps of the pose estimator while longer ones reflect wrong tracking.

A typical evolution of the estimated position, velocity and scale is shown in figure 9.

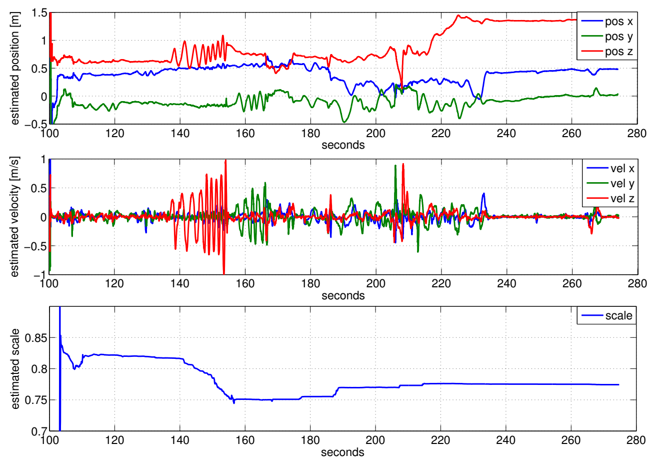

Fig. 9. The top graph shows the evolution of the estimated position. Only from second 137 on we perform enough excitation to let the scale fully converge. We note, however, that already before the scale converged to about 4% of the true value. That is, we actually need remarkably low excitations to estimate roughly the scale factor of the black box visual framework.

VI. CONCLUSION

In this paper, we showed an approach to estimate the full and metric scaled state of a camera-IMU device in real time. The novelty of our solution lies in the decoupling of the (visual) pose estimate and the filter state estimation. This makes our approach highly modular to be used with any pose estimation algorithm such as visual odometry, visual SLAM, monocular or stereo setups or even GPS solutions with gravity and compass attitude estimation. The pose estimation part can be treated as a black box which makes our solution usable for closed third party algorithms as well. The decoupling of the algorithms leads directly to a constant computational complexity of the state estimation filter. It consists of multiplications of 22×22 matrices and an inversion of a 6×6 matrix only. That is, with our solution the whole framework takes the complexity of the used pose estimation algorithm. Moreover, we propose a second contribution to address failure and drift estimate issues arising when treating the pose estimation part as a black box. We carefully implemented and analyzed the state estimation filter and evaluated its sensitivities to initial state errors. We also demonstrated successfully the functioning of the filter on real data and showed that we reliably can identify failures and drifts of the pose estimation algorithm. This ensures proper functioning of the filter only having access to the pose output of the black box.

REFERENCES

[1] G. Klein and D. Murray, "Parallel tracking and mapping for small AR workspaces," in *Proc. Sixth IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR'07)*, Nara, Japan, November 2007.

[2] A. J. Davison, "Real-time simultaneous localisation and mapping with a single camera," in *ICCV '03: Proceedings of the Ninth IEEE International Conference on Computer Vision*. Washington, DC, USA: IEEE Computer Society, 2003.

[3] M. Bloesch, S. Weiss, D. Scaramuzza, and R. Siegwart, "Vision based mav navigation in unknown and unstructured environments," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2010, pp. 21–28.

[4] L. Armesto, J. Tornero, and M. Vincze, "Fast ego-motion estimation with multi-rate fusion of inertial and vision," *Int. J. Rob. Res.*, vol. 26, no. 6, pp. 577–589, 2007.

[5] F. Mirzaei and S. Roumeliotis, "A kalman filter-based algorithm for imu-camera calibration: Observability analysis and performance evaluation," *IEEE Transactions on Robotics*, vol. 24, no. 5, pp. 1143 –1156, oct. 2008.

[6] J. Kelly and G. S. Sukhatme, "Visual-inertial sensor fusion: Localization, mapping and sensor-to-sensor self-calibration," *International Journal of Robotics Research*, vol. to appear, no. CRES-08-005, Nov 2010.

[7] E. Jones, "Large scale visual navigation and community map building," Ph.D. dissertation, University of California at Los Angeles, June 2009.

[8] P. Pinies, T. Lupton, S. Sukkarieh, and J. D. Tardós, "Inertial aiding of inverse depth slam using a monocular camera." in *ICRA*. IEEE, 2007, pp. 2797–2802.

[9] D. Strelow, "Motion estimation from image and inertial measurements," Ph.D. dissertation, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, November 2004.

[10] H. Strasdat, J. M. M. Montiel, and A. J. Davison, "Real-time monocular slam: Why filter?" in *IEEE International Conference on Robotics and Automation (ICRA)*, 2010.

[11] N. Trawny and S. Roumeliotis, "Indirect kalman filter for 3d attitude estimation," University of Minnesota, Department of Computing Science and Engineering, Tech. Rep., 2005.

[12] P. S. Maybeck, *Stochastic Models, Estimation and Control*. New York: Academic Press, 1979, vol. vol. 1.

[13] C. Beder and R. Steffen, *Determining an initial image pair for fixing the scale of a 3d reconstruction from an image sequence*, ser. LNCS, K. Franke, K.-R. Müller, B. Nickolay, and R. Schäfer, Eds. Springer, 2006, no. 4174.

[14] A. Eudes and M. Lhuillier, *Error propagations for local bundle adjustment*. Los Alamitos, CA, USA: IEEE Computer Society, 2009, vol. 0.