A ROBUST AND MODULAR MULTI-SENSOR FUSION APPROACH APPLIED TO MAV NAVIGATION

Citation for published version:

Digital Object Identifier (DOI):
10.1109/IROS.2013.6696917

Link:
Link to publication record in Edinburgh Research Explorer

Document Version:
Peer reviewed version

Published In:

General rights
Copyright for the publications made accessible via the Edinburgh Research Explorer is retained by the author(s) and / or other copyright owners and it is a condition of accessing these publications that users recognise and abide by the legal requirements associated with these rights.

Take down policy
The University of Edinburgh has made every reasonable effort to ensure that Edinburgh Research Explorer content complies with UK legislation. If you believe that the public display of this file breaches copyright please contact openaccess@ed.ac.uk providing details, and we will remove access to the work immediately and investigate your claim.
A Robust and Modular Multi-Sensor Fusion Approach Applied to MAV Navigation

Simon Lynen¹, Markus W. Achtelik¹, Stephan Weiss², Margarita Chli¹ and Roland Siegwart¹

Abstract—It has been long known that fusing information from multiple sensors for robot navigation results in increased robustness and accuracy. However, accurate calibration of the sensor ensemble prior to deployment in the field as well as coping with sensor outages, different measurement rates and delays, render multi-sensor fusion a challenge. As a result, most often, systems do not exploit all the sensor information available in exchange for simplicity. For example, on a mission requiring transition of the robot from indoors to outdoors, it is the norm to ignore the Global Positioning System (GPS) signals which become freely available once outdoors and instead, rely only on sensor feeds (e.g., vision and laser) continuously available throughout the mission. Naturally, this comes at the expense of robustness and accuracy in real deployment. This paper presents a generic framework, dubbed Multi-Sensor-Fusion Extended Kalman Filter (MSF-EKF), able to process delayed, relative and absolute measurements from a theoretically unlimited number of different sensors and sensor types, allowing self-calibration of the sensor-suite. The modularity of MSF-EKF allows seamless handling of additional/lost sensor signals online during operation while employing an state buffering scheme augmented with Iterated EKF (IEKF) updates to allow for efficient re-linearization of the propagation to get near optimal linearization points for both absolute and relative state updates. We demonstrate our approach in outdoor navigation experiments using a Micro Aerial Vehicle (MAV) equipped with a GPS receiver as well as visual, inertial, and pressure sensors.

I. INTRODUCTION
Precise and consistent localization is a core problem in many areas of mobile robotics, in both research and industrial applications. Driven by the need for effective solutions, the literature is currently host to an abundance of approaches to state estimation, addressing different choices of on-board sensor suites tailored to the task at hand. The use of GPS feeds, for example, is a common and convenient approach to localization for platforms operating in open (GPS-accessible) spaces. Conversely, in GPS-denied environments, vision or laser based approaches are often employed instead. The transition, however, across domains with different sensor-signal availability and suitability, remains a challenging problem.

In this paper, we present an effective approach to tackle the problem of seamless sensor-feed integration within state estimation. We put the focus on rotor-based Micro Aerial Vehicles (MAVs), as they are most capable of acting in and traversing across different domains, while imposing delicate challenges due to their high agility and limitations on both payload and computational power. Building on our earlier work [16], [17], we propose a highly generic, open source c++ state estimation framework which comprises:

- Modular support for an unlimited number of sensors.
- Estimation of calibration states between sensors.
- Compensation for unknown and changing delays.
- Handling of relative and absolute measurements.
- Efficient tracking of cross covariance terms for relative updates.
- Rere-linearization of the propagation after delayed updates.
- Iterated EKF over variable sized windows of updates.
- Measurement outlier rejection module.
- Efficient implementation for estimation rates of several kHz.

Following an analysis of the limitations of our earlier work, we also present a derivation to include relative poses from key-frame based Simultaneous Localization And Mapping (SLAM) system, which is essential when employing visual/laser odometric sensors. Finally, we demonstrate the MSF-EKF framework in real experiments, flying trajectories of more than 800 m with speeds of up to 4 m/s.¹

A. Sensor Fusion for State Estimation

Autonomous MAV navigation and control has seen great success over the last couple of years, demonstrating impressive results with the aid of external motion capture systems. However, the complex preparation of the operation space required with such systems is clearly not an option in large scale missions in unknown environments. Tackling this challenge is core in enabling operation for common tasks such as industrial inspection, search-and-rescue and surveillance. As

* This work has been supported by the European Commission’s Seventh Framework Programme (FP7) under grant agreements n. 285417 (ICARUS) and n.266470 (myCopter).
1 Autonomous Systems Laboratory, ETH Zurich, Switzerland
2 Computer Vision Group, Nasa JPL, California, USA

¹A video of the experiments is available at http://youtu.be/neG8iEf8XIQ
a result, a series of approaches have been proposed, using on-board sensors such as laser-range finders [13], visib-
light [1] or depth cameras, typically fused with readings from
an Inertial Measurement Unit (IMU) to provide information
about the state of the vehicle.

Most often, these state estimation approaches are designed
to use a specific sensor setup for the domain space of the task
at hand. While showing unmatched accuracy and consistency
[5], [9] they are commonly designed for a particular sensor
setup with limited modularity. Despite that GPS signals
become available once the robot moves outdoors, they are
often ignored in the state estimation [1], [17] as they require
a more complex control strategy when moving from a local
to global frame of reference. As a result, many current
frameworks used on board MAVs fail to utilize all available
information, limiting both accuracy and robustness of the
state estimate. In [8] sensor outages were advocated in the
context of fixed-wing aerial navigation with one position
sensor, demonstrating successful state estimation during sim-
ulated temporary outage of GPS signal. In [11] GPS and
visual measurements where used for different periods of the
experiment. Here, we present a generic framework, which
permits handling online with the effects of a multitude of
different sensors. Fusing multiple sensors was also addressed
recently in [6] where both relative and absolute sensors
are included in a factor graph formulation using non-linear
optimization. While fixed-lag smoothers based on non-linear
optimization have potentially higher accuracy due to the
ability to re-linearize the state propagation, the authors of [6]
report a computational cost which is higher by two orders
of magnitude compared to the framework presented here.

B. Self-Calibration of Sensors and Scale estimation

In navigation frameworks, any vehicle states essential for
robot control are commonly estimated at high rates, which
is especially critical for platforms like MAVs. In a typical
scenario, inertial measurements arriving at rates of several
100 Hz to 2 kHz are fused with lower rate exteroceptive
updates (~5 – 90 Hz), coming from e.g., GPS or visual
odometry, to mitigate drifts. Common fusion approaches
are based on indirect formulations of Extended (EKF) [13],
or Unscented (UKF) Kalman Filters [14]. In [10], it was
shown that additional quantities of interest can be estimated
in the same manner; for example, the intrinsic calibration of
the proprioceptive sensors, the extrinsic calibration between
proprio- and exteroceptive sensors, as well as unknown
quantities from the exteroceptive-sensor process such as the
scale and drifts of a monocular SLAM system. For the study
of inter sensor calibration we refer to our earlier work [17].

The accuracy of monocular visual-inertial frameworks is
dominated by the correct estimation of the scale. In Fig. 1
we show the first 350 m of an 800 m flight of a MAV flying
with speeds of up to 4 m/s in circles over grass. To highlight
the error in scale we plotted estimate and ground-truth in the
x and y directions versus the traveled distance. The left plot
highlights the error in the scale estimation of about 5 % while
the right plot shows the same data when the scale error is
minimized. This demonstrates the potential benefits of fusing
additional sources of metric information which then leads to
more accurate estimates also in long range missions.

Here, we adopt this idea to achieve online self-calibration
of the sensor-suite. Furthermore we adapt our framework to
handle relative measurements to avoid the shortcomings of
our previous work: In [16], the local map is considered as
noise free which leads to an inconsistent state estimate.

C. Relative and absolute pose measurements

In [16], we discussed the un-observability of states such
as the relative position and yaw between the SLAM-frame
and the world-frame in a visual-inertial navigation system.
This problem is commonly addressed by fixing the respective
states in the estimation process and applying pose estimates
from the visual SLAM algorithms as pseudo-absolute meas-
urements [3], [14], [15].

However, it has been shown [11] that applying the (rel-
ative) pose estimates from a visual odometry system as
pseudo-absolute measurements leads to sub-optimal esti-
mates, as the uncertainty of the pose computed by a visual
odometry system (or key frame based SLAM with a limited
number of key frames) is a relative and not an absolute
quantity. This leads to inconsistencies and does not allow the
estimator to correct for drifts in the visual SLAM system.
Here, we circumvent this problem by adopting Stochastic
Cloning[12] which allows us to include relative measure-
ments in a relative context only, which also means, that
we no longer incorporate local estimates of the scale factor
(typically effected by drift and jumps) to the global position
estimation. This contrasts with our previous work where the
latest scale estimate was applied to the global pose update,
which means that a small local drift in scale would falsely
result to large changes in global position estimates.

II. COMBINING MULTIPLE SENSORS

Our framework is based on the indirect formulation of an
iterated EKF where the state prediction is driven by IMU
measurements. The estimated state consists of number of
core states:

\[ x_{\text{core}}^T = \left[ p_{u,w}^T, v_{u,w}^T, q_{u,w}^T, b_{u,w}^T, b_{u,w}^T \right] \]  

(1)

Namely, these correspond to the relative position \( p_{u,w} \), velocity
\( v_{u,w} \), and attitude\(^2 \) \( q_{u,w} \) of the imu w.r.t. the inertial frame.
Furthermore we estimate IMU acceleration and gyroscope
biases \( b_{u} \) and \( b_{w} \), respectively. Additional sensors can then
be added modularly with respect to the IMU frame.

We use the insight from the observability analysis carried
out in [16] for the implementation of our framework in
order to design a sensor suite for a particular platform and
mission. For example, if we were to use a (differential) GPS
receiver in addition to the IMU, we would need to add the
translation between these two sensors as extrinsic calibration
state. Similarly, a pressure sensor introduces a translational

\(^2\)Rotations are parameterized as Quaternions of rotation in Hamilton
notation.
we define the 15-element error state vector for core states as providing relative 6-DoF pose estimates) – a loosely coupled camera (whose feeds are processed in a visual SLAM update formulation of a common setup in MAV navigation calibration state and an additional bias state in the global z axis. A more complex example is a camera yielding a six-degrees of freedom (DoF) pose, measuring w.r.t. a separate frame of reference which drifts in all six dimensions. As with the pressure sensor, we account for these drifts by adding a six-DoF state describing the drift between the intertial frame and the frame the sensor measurements are expressed in. Furthermore, we need to add a six-DoF extrinsic calibration state with respect to the IMU. A camera could as well measure optical flow representing a 3D body velocity sensor [17]. Since these are no global position measurements, we measure optical flow representing a 3D body velocity sensor state with respect to the IMU. A camera could as well Furthermore, we need to add a six-DoF extrinsic calibration and the frame the sensor measurements are expressed in. Additionally, we need to add a six-DoF extrinsic calibration state added by this sensor are:

\[
\tilde{\mathbf{x}}_{\text{core}} = \begin{bmatrix} \Delta p_w^T, \Delta v_w^T, \delta \Theta_w^T, \Delta b_w^T, \Delta b_a^T \end{bmatrix},
\]

with \( \tilde{x} \) representing the difference of an estimate \( \hat{x} \) to its true value \( x \), which is defined as \( \delta \hat{q} = q \otimes \hat{q} \approx \left[ \frac{1}{2} \delta \Theta^T \right] \) for quaternions. Additional to these core states, every sensor adds a number of auxiliary states which relate the measured quantities to the core states. Therefore the full EKF state is assembled from the core states \( \tilde{x}_{\text{core}} \) and a series of additional states \( \tilde{x}_{s_i} \) which are defined by the sensor type:

\[
\tilde{x} = \{ \tilde{x}_{\text{core}}, \tilde{x}_{s_1}, \tilde{x}_{s_2}, ..., \tilde{x}_{s_n} \}.
\]

Below, we derive the EKF equations for the visual-inertial-pressure suite.

1) Pressure sensor: In order to obtain height estimates from a pressure sensor we need to account for the bias \( b_{\text{press}} \) resulting from the changes in ambient pressure:

\[
\tilde{x}_{\text{press}} = [\Delta b_{\text{press}}].
\]

For the pressure measurement \( z_{\text{press}} \), the following measurement model applies:

\[
z_{\text{press}} = p_{\text{press}} - b_{\text{press}} + n_{\text{press}}
\]

with \( n_{\text{press}} \) denoting the measurement noise modeled as zero-mean, white and Gaussian. We define the error in the \( z \)-position generally as \( \tilde{z} = z - \hat{z} \). Which can be linearized to \( \tilde{z} = H_{\text{press}} \hat{x} + \eta \), where \( H_{\text{press}} \) denotes the Jacobian of the pressure measurement w.r.t. the (16-dimensional) error state.

2) Monocular visual SLAM sensor: The quantities to be estimated, are the scale \( \lambda \) and the drifts in position \( p_{vi}^C \) and attitude \( q_{vi}^w \) of the visual SLAM system w.r.t. the world-frame as well as the camera-to-IMU rotation \( \hat{q}_c \). We do not estimate for the camera-to-IMU translational offset \( p_{ci}^w \) online, as it is unlikely to change significantly during the mission. The states added by this sensor are:

\[
\tilde{x}_{\text{vis}}^T = [\Delta \lambda, \delta \Theta_{\text{vis}}^T, \Delta p_{vi}^w \Theta_{\text{vis}}^T].
\]

For the camera pose measurement \( z_{\text{vis}} \), the following measurement model applies [16]:

\[
z_{\text{vis}} = \begin{bmatrix} p_{wi}^C \, q_{wi}^C \end{bmatrix} = \begin{bmatrix} C(q_{vi}^w) (p_{wi}^C + C^T(q_{vi}^w) p_{vi}^C) \lambda + p_{vi}^w + n_{pv} \end{bmatrix} + q_{wi}^v \otimes q_{vi}^w - 1 + n_{qv},
\]

with \( C(q_{vi}^w) \) as the rotation matrix corresponding to the IMU’s attitude and \( C(q_{vi}^w) \), \( p_{vi}^w \), the rotation and translation of the world frame to the vision frame, respectively. The visual measurement is corrupted by noise \( n_{pv} \) and \( n_{qv} \) which we model as zero mean, white, Gaussian.

We define the position and attitude error of the vision measurement as

\[
\tilde{z}_{\text{vis}} = \begin{bmatrix} \tilde{z}_p \\ \tilde{q}_v \end{bmatrix} = C^T(q_{vi}^w) (p_{vi}^w + C^T(q_{vi}^w) p_{vi}^C) \lambda + p_{vi}^w \otimes q_{vi}^w - 1 + n_{qv}
\]

which can be linearized to \( \tilde{z} = H_{\text{vis}} \hat{x} + \eta \), where \( H_{\text{vis}} \) holds the Jacobian of the (visual-) pose measurement w.r.t. the error state.

IV. GENERIC AND MODULAR IMPLEMENTATION

A. Processing delayed measurements

The proprioceptive sensor readings are used to predict the state at IMU rate in real time, which is crucial for for MAV attitude control. This estimate is updated with other sensor readings (e.g. from SLAM) which are available only at lower rates and arrive with significant (and potentially unknown) time delay. In our previous work [17], we proposed to use a ring buffer for the states so that we could apply state updates in the past. After applying an update to the corresponding state in the past we re-propagate the state to the current time to keep the best state prediction available for control at high rate, see Fig. 2.

B. Process multiple delayed measurements

In order to integrate multiple sensor readings, we extend our previous approach in [17]: we maintain a potentially infinitely long buffer for states and measurements, inside which the elements are sorted by time. Whenever new IMU readings become available, a new state-object is instantiated, the state is propagated using the proprioceptive measurements and then inserted into the state buffer, illustrated at the bottom row in Fig. 3.
Fig. 2: The ring-buffer scheme we proposed in [16] to compensate for delayed measurements. a) The current state is used for control. The covariance is not required for control and therefore only propagated on-demand. b) A delayed measurement arrives upon which the corresponding state is queried in the buffer. The covariance propagation is carried out to the state and both quantities are corrected by the measurement. c) The updated state is predicted to the current time to provide the most recent estimate for the controller. The covariance is propagated to the state where we anticipate the next measurement to arrive.

As in Section III, we consider the typical MAV sensor-setup of an IMU, a pressure sensor (one DoF biased estimate) and a monocular visual-SLAM system (six DoF pose with unknown scale). Fusing the metric pressure sensor readings on IMU feeds ensures faster scale convergence and more accurate state estimates, as discussed in Section VI.

Fig. 3: a) shows how pressure measurements ($z_p$) are applied to the closest state in the buffer. Interpolation allows us to use the best available linearization point. In b), delayed vision measurements ($z_v$) are applied by querying the closest state in the buffer, and any subsequent measurements are re-applied to the updated state. Finally, in c), sensors with different rates can be directly integrated to the framework. The covariance is only propagated on-demand (bold-dash arrows), while the state is always predicted to the current time for control after all pending measurements have been applied (point-dash arrows).

Commonly, the pressure sensor has (constant) measurement rates in the order of 50 Hz, while the visual-SLAM system usually operates at a (varying) rate between 30 and 90 Hz. In the upper row in Fig. 3 a) we illustrate how the pressure measurements arrive and are applied to the state closest to their respective measurement time. In cases where no state is available at the measurement time, we interpolate the respective proprioceptive measurements to get the best available linearization point Fig. 3 a).

Fig. 3 b) illustrates the case where the delayed measurement of the visual-SLAM system becomes available. As before, the closest state (in the past) is retrieved from the state-buffer, which is updated using the data from the visual-SLAM system. Subsequently, the measurement buffer is queried recursively for later measurements, between which both the state and covariance are propagated accordingly.

After applying the last measurement in the measurement-buffer, all subsequent states in the state-buffer are propagated so that the most recent estimate is available for vehicle control Fig. 3 c).

C. Delayed state initialization

In multiple sensor fusion, it is typical that some sensor feeds might be missed for some time or switch between available and unavailable during the mission. For example, indoors the MAV can navigate using the on-board visual and inertial onboard sensors (global position and yaw unobservable) but with the transition to outdoors, GPS measurements become available (rendering global position and yaw observable).

D. Relinearization of the propagation and IEKF window updates

Keeping path states and covariances in a buffer also allows us to employ an IEKF scheme over a window of measurements. When our state a-priori state estimate was far from the a-posteriori state estimate, we can hold the update back and first iterate over a window of measurements and refine the linearization points before. Given the little computational cost for state and covariance propagation we can thus relinearize the propagation multiple times in order to reduce linearization errors for highly non-linear systems. Naturally this extends to applying a set of updates as batch non-linear optimization.

E. Separation of core states and auxiliary states

In our implementation we separate the state in core and auxiliary parts (which were added e.g., as bias and calibration terms for a particular sensor) already at the state level (see (3)). This allows us to perform optimizations by exploiting this knowledge in the propagation steps of the EKF. The code related to the core states can therefore be completely separated from the sensor specific implementations, rendering our MSF-EKF implementation very easy to extend. The definition of the current sensor suite is done at one place, and then used to unfold the full state and compute the dimensionality of the full state all at compile time. This allows transparent sensor integration and highly efficient matrix operations.

The design of a new filter setup therefore consists only in the implementation of the measurement Jacobian and residual for the sensor. Additional states can be added with a single line at a central place from which the framework computes all indices and derives the necessary calculations at compile time. This state definition also specifies the local parameterizations for both Quaternion error state Jacobian and update functions so that the filter can apply the correct parameterization automatically. Since our open source implementation includes a large set of sensor implementations, in most cases only the specification of the state is necessary.

F. Outlier rejection

The framework allows the modular addition of outlier rejection methods to each measurement module. The filter core module then performs then e.g., a Mahalanobis test before applying the update to the state.

G. Compile time calculation of all indices and matrix dimensions

Because the current sensor suite and the respective state is known at compile time, we make extensive use of template
meta-programming techniques to let the compiler compute all required matrix dimensions and all indices required in the EKF computation. This allows us to exploit the full efficiency of the linear algebra framework Eigen\(^3\) we employ.

V. Processing Relative Measurements

This formulation of the visual update has drawbacks when using measurements from a visual odometry framework: The latest estimate of the visual scale is used to scale the whole visual odometry path, no taking into account intermediate changes in scale. In addition to that visual odometry systems (i.e. performing SLAM with a limited number of key-frames to bound computational complexity), provide a pose measurement which denotes a measurement between time-instants \(k\) and \(k + m\) rendering the measured quantities dependent on the state values \(x_k\) and \(x_{k+m}\), as well as the previous measurement. Nevertheless many recent publications ([14], [17], [15]) compute visual updates from a local map with fixed landmarks and apply them as absolute pose measurements leading to inconsistent estimates, prohibiting fusion with absolute measurements like GPS.

By the Markov assumption, in an EKF all information about past states is contained in the latest state estimate and both the state \(\hat{x}_k\) and the corresponding covariance \(\hat{P}_k\) are available. The standard EKF formulation, however, does not allow for direct consideration of the correlations between the states at different time steps and therefore, applying a relative measurement is not straightforward.

Similar to the Stochastic Cloning approach of [12], we adapt the EKF update equations to handle absolute measurements as in [17] and then to handle the relative measurements, relating two states. The measurement equation for the relative update is:

\[
\tilde{z}_k = H_{k+m}\hat{x}_{k+m|k} + H_k\hat{x}_{k|k} + \eta,
\]

where the subscript \(k + m|k\) denotes the predicted quantities at time \(t_{k+m}\) and \(k|k\) corresponds to the posterior at time-step \(t_k\). Moreover, \(\hat{x}\) is state vector and \(H\) the corresponding measurement Jacobian.

A. Updating the state with a relative measurement

In order to account for relative measurements, the authors of [12] propose to add a clone of the state for each sensor providing relative measurements as well as the respective errors for the landmarks used to compute the relative measurements. Since we want to use the sensors in a loosely-coupled manner abstracting the internal algorithms (e.g. SLAM) we don't include the measurement errors for landmarks relating both states in our state vector. To keep the computational complexity low and because in general not all measurements in a multi-sensor setup denote relative quantities we do not in general clone every state but rather make use of our framework design, according to which, we can access the pair \(\hat{X}\) of past states we want to relate by a given relative measurement, at any time.

\[
\hat{X} = \begin{bmatrix} \hat{z}_k^T & \hat{z}_{k+m}^T \end{bmatrix}^T.
\]

We then build the full covariance matrix of this state pair \(\hat{X}\):

\[
\hat{P}_{k+m|k} = \begin{bmatrix} P_{k|k} & P_{k|k}\mathcal{F}_{k+m,k}^T \\ \mathcal{F}_{k+m,k}P_{k|k} & P_{k+m|k} \end{bmatrix}
\]

with \(\mathcal{F}_{k+m,k} = \prod_{i=k}^m F_{k+i}\) corresponding to the concatenation of the linearized system dynamic matrix. We store the \(F_k\) with the states (given the respective best available linearization point) in the buffer and only carry out the product accumulation and multiplication with \(\hat{P}_{k|k}\) when we want to apply a relative measurement. Thereby all additional measurements arriving within the time spanned by the relative measurements are considered and improve the respective linearization points for \(F_k\). The residual \(r_{k+m}\) and the covariance of a relative measurement \(\hat{S}_{k+m}\) are given by:

\[
r_{k+m} = \hat{z}_{k+m} - \hat{z}_{k} \approx \hat{H}\hat{X},
\]

\[
\hat{S}_{k+m} = \hat{H}\hat{P}_{k+m|k}\hat{H}^T + R_r,
\]

where \(R_r\) is the covariance of the relative pose coming from the employed SLAM framework and \(\hat{H} = [H_k|k\ H_{k+m|k}]\) comprises the two corresponding measurement Jacobians. The Kalman gain calculation then is straightforward:

\[
\hat{K} = \hat{P}_{k+m|k}\hat{H}^T\hat{S}_{k+m}^{-1} = \begin{bmatrix} K_k^T & K_{k+m}^T \end{bmatrix}^T.
\]

The final step is the correction of the state and the covariance at time step \(t_{k+m}\) given the residual \(r_{k+m}\):

\[
\hat{x}_{k+m|k+m} = \hat{x}_{k+m|k} + \hat{K}_{k+m}r_{k+m},
\]

\[
P_{k+m|k+m} = P_{k+m|k} - \hat{K}_{k+m}\hat{S}_{k+m}\hat{K}_{k+m}^T.
\]

Given the better estimate of \(\hat{x}_{k+m|k+m}\) we can now re-apply all measurements that arrived after the relative measurements and re-propagate the state using the new linearization points. Since we perform relinearization of the propagation multiple relative measurements are always applied using the best available linearization point.

The main question left is the derivation of the measurement covariance for the relative update.

B. Pose estimation covariance in key-frame based visual SLAM systems

Instead of deriving the uncertainty in the camera pose from the propagation of the pose and the visible landmarks as in EKF SLAM, the covariance of the camera-pose is obtained from bundle adjustment. This non-linear optimization technique solves simultaneously for both 3D-map points in the world and the camera locations over time. This is done by minimizing the weighted-least-squares re-projection errors of the 3D map points across all images, which then provides an estimate of the pose covariance [4]. Since the solution of the bundle adjustment problem is costly, the real-time pose estimate (pose tracking) is commonly calculated from

\(^3\)http://eigen.tuxfamily.org
an approximation where the map-points are kept fix and the current camera pose is recovered solving the perspective n point (PnP) problem. The loosely coupled SLAM systems employed in related work [14], [17] therefore provide a covariance of the current pose $T_{\text{Map}}^{k+1}$ w.r.t. the local fixed map which is not the quantity we would need to apply relative measurements. Furthermore by applying these local transformations $T_{k+1}^{k+1}$ as relative updates annihilates one of the main benefits of key-frame based SLAM system, namely the absence of temporal drift as shown in Fig. 4 a).

C. Stochastic Cloning for key-frame based visual SLAM systems

The covariance calculated w.r.t. the fixed map actually denotes the uncertainty of the current pose $T_{\text{Map}}^{k}$ w.r.t. the fixed key-frame $K_{n}^{\text{fixed}}$ in the bundle adjustment problem as shown in Fig. 4 b). Therefore to correctly integrate the relative-measurements, we need to apply stochastic-cloning to the EKF-state corresponding to the time the current fixed key-frame became fixed. Since for long term missions we have to drop old key frames to keep computational demands low, which key-frame is fixed in the bundle-adjustment problem is changing over time. This means that the current pose needs to be computed relative to a changing reference in the map as shown in Fig. 4 b).

The uncertainty of the fixed key-frame w.r.t. the world frame is changing whenever the fixed key-frame changes. At this moment the uncertainty of the past fixed key-frame w.r.t. the world frame is augmented with the uncertain transformation of the new fixed key-frame which is obtained from bundle-adjustment.

The pose measurement is therefore computed via the uncertain transformation chain from the world frame to the fixed key-frame $T_{\text{World}}^{K_{n}^{\text{fixed}}}$ and from there to the current camera pose estimate $T_{\text{map}}^{k}$.

This allows us to take advantage of the non temporal drifting estimate of key-frame based SLAM but at the same time correctly account for the uncertainties in the pose estimate which then allows fusion with additional exteroceptive sensors.

VI. EXPERIMENTS

In this section we present a series of experiments which we carried out by flying $\sim 350$ m with an Asctec Firefly MAV equipped with a Intel Core2Duo computer\(^4\). All computation is done on-board and the flights are purely vision based on grass with speeds of up to 4 m/s. Ground truth for all experiments is provided by a sub-mm precision Leica TotalStation 15i.

A. Covariance of pose estimate for sensor fusion

As discussed earlier, most visual SLAM systems based on key-frames (e.g. PTAM [7]) provide a pose estimate and respective covariance w.r.t. a local map following bundle adjustment. The covariance of such visual pose estimates is most often over-confident since the uncertainty in the pose is estimated assuming a fixed map for computational efficiency (in contrast to EKF SLAM where a full covariance matrix of the pose and the map is jointly maintained). When fusing this visual estimate as an absolute measurement the covariance of the global position does not increase with the distance traveled as shown in Fig. 5 (The oscillations show how applying local PnP based estimates as relative updates annihilates one of the main benefits of key-frame based SLAM system, namely the absence of temporal drift as shown in Fig. 4 a)).

The actual experiment was 800 m but ground truth is only available for a sub-part.
Online sensor switching

While the improvement of the estimated state is one benefit of including additional sensors, the capability of seamless switching between the elements of the sensor-suite online adds both additional fail-safety and versatility. One common case is the change from local vision-based navigation (indoor) to GPS based global position estimates (outdoor). Current experimental results for MAV navigation are primarily bound by battery life and therefore rather small in scale of operation (usually trajectories of around 1 km). While drifts in position and visual-scale become important at this trajectory length, the local estimate provided by vision systems is still superior to GPS. To demonstrate the capability of online sensor switching, we added alternating drop-outs of both GPS and vision measurements. Some dropouts of the visual update take place when the MAV was passing a group of trees, where the GPS is highly corrupted by multi-pass and high dilution of precision. In this area, the state estimate then follows the corrupted GPS measurements adding errors as large as 5 m (Fig. 7).

C. Processing time

While the main motivation for the heavy use of template meta-programming was to keep the framework generic and transparent to the employed sensor-suite, there are also significant improvements in terms of computational efficiency as detailed in the table below. The state corresponds to a visual inertial fusion filter estimating both rotation and translation of the extrinsic imu to camera calibration.

<table>
<thead>
<tr>
<th>Cost of function call (EKF: 31 states)</th>
<th>mean</th>
<th>stddev</th>
</tr>
</thead>
<tbody>
<tr>
<td>IMU handling and state propagation</td>
<td>44 µs</td>
<td>23 µs</td>
</tr>
<tr>
<td>Covariance propagation</td>
<td>31 µs</td>
<td>24 µs</td>
</tr>
<tr>
<td>Get state for delayed measurement</td>
<td>79 µs</td>
<td>53 µs</td>
</tr>
<tr>
<td>Apply measurement</td>
<td>65 µs</td>
<td>37 µs</td>
</tr>
<tr>
<td>Re-propagate state after measurement</td>
<td>21 µs</td>
<td>22 µs</td>
</tr>
<tr>
<td>Additional fixed overhead</td>
<td>11 µs</td>
<td>9 µs</td>
</tr>
</tbody>
</table>

References


