Metrics for Real-Time Mono-VSLAM Evaluation including IMU induced Drift with Application to UAV Flight

Alexander Hardt-Stremayr[1][0000−0002−0464−5303], Matthias Schörghuber[2][0000−0001−7629−0348], Stephan Weiss[1][0000−0001−6906−5409], and Martin Humenberger[3][0000−0003−0600−9164] ⋆

[1] Alpen-Adria Universität Klagenfurt
[2] Austrian Institute of Technology
[3] NAVER LABS Europe

Abstract. Vision based algorithms became popular for state estimation and subsequent (local) control of mobile robots. Currently a large variety of such algorithms exists and their performance is often characterized through their drift relative to the total trajectory traveled. However, this metric has relatively low relevance for local vehicle control/stabilization. In this paper, we propose a set of metrics which allows to evaluate a vision based algorithm with respect to its usability for state estimation and subsequent (local) control of highly dynamic autonomous mobile platforms such as multirotor UAVs. As such platforms usually make use of inertial measurements to mitigate the relatively low update rate of the visual algorithm, we particularly focus on a new metric taking the expected IMU-induced drift between visual readings into consideration based on the probabilistic properties of the sensor. We demonstrate this set of metrics by comparing ORB-SLAM, LSD-SLAM and DSO on different datasets.

1 Introduction

Robot pose estimation in unknown environments is an important and active field. A proven method is simultaneous localization and mapping (SLAM), which estimates the robots pose within a self-constructed map using one or more sensors (e.g. camera, laser).

A subset of SLAM, which uses cameras to estimate the pose is known as Visual-SLAM (VSLAM) or, without loop-closure, Visual-Odometry (VO). Both methods can be used in either a stereo or a monocular (only up to scale) setup. The latter being of particular interest for payload constrained robots like unmanned aerial vehicles (UAV).

⋆ The research leading to these results has received funding from the ARL within the BAA W911NF-12-R-0011 under grant agreement W911NF-16-2-0112 and from the Austrian Ministry for Transport, Innovation and Technology (BMVIT) under the grant agreement 855468 (Forest-IMATE) and under the grant agreement 848518 (AVIS).

Because of their agility, multirotor UAVs need pose estimation at higher frequency than a camera based algorithm might provide (also limited by the camera frame rate and the processing time). To overcome this problem, an inertial measurement unit (IMU) is usually used to estimate the pose between two camera measurements at high frequencies (>100Hz).

However, MEMS-based IMUs typically used in multirotors (because of price, weight, and energy consumption) have a significant amount of noise and bias in the acceleration and angular velocity measurements. As this noise and bias are integrated for position twice (or even more in the case of the angular velocity noise and bias for non-holonomic platforms like multicopters), it will generate noticeable position drift over time. This can lead to mission failure and, in the worst case, to accidents.

1.1 Problem Statement

When it comes to evaluation of the performance of VSLAM/VO algorithms, researchers often only use the drift relative to the total trajectory traveled as metric. In contrast to ground vehicles, UAVs are operating in 3D space and cannot stop and wait for the result of pose estimation during flight (even hovering needs continuous pose estimation).

Furthermore, IMU noise as well as gravity acceleration erroneously aligned in x or y direction create correctional movements once the next vision based update occurs. Such movements cause additional energy consumption and, even worse, may destabilize the flight. Thus, for the decision about the best visual pose estimation algorithm to be combined with inertial readings for closed loop vehicle control, the *time* components of the vision system (execution time, latency, etc.) are crucial rather than the global drift.

1.2 Contribution

We introduce a new metric based on statistical integration and the IMU noise characteristics that penalizes abrupt position corrections of the state estimate (which e.g. may happen after a long period of pure IMU integration and subsequent visual correction).

While this metric may be applied to all pose estimation algorithms using an IMU as core propagation sensor, we focus on visual odometry and VSLAM in section 5.

Our approach makes use of the probabilistic properties of the IMU noise and drifts such that the suggested statistical integration can provide the expected IMU noise and gravity-misalignment induced errors over time. This allows us to predict the discontinuities occurring on a visual correction step based on the characteristics of a potential IMU.

We develop a method to calculate the pose error based on IMU noise as well as the gravity alignment (roll and pitch) error depending on time. The advantage over existing metrics is twofold. First, time constraints are included in the metric

in a meaningful way: while a slow algorithm may provide more accurate results, the calculation time required to compute those results can decrease the overall performance. Second, the roll and pitch errors can be mapped to the position error. This enables evaluation of the pose error instead of only the position upon a correction step.

Furthermore, we show that this metric can easily be extended to include not only effects induced by IMU noise/integration errors but also to include the (more classical) distance to ground truth.

Finally, we apply our metric on a set of state-of-the-art algorithms, namely ORB-SLAM, LSD-SLAM and DSO and provide a comparison in order to demonstrate usability and benefit of this proposed metric.

2 Related Work

A qualitative evaluation compares different approaches in terms of the used methods, while a quantitative evaluation compares the results of the methods using certain metrics on different datasets.

Younes et al. [15] provide a thorough overview of monocular visual SLAM systems. They provide a historical overview as well as descriptions of techniques used in different SLAM algorithms until 2015. Current state-of-the-art algorithms are described and compared qualitatively.

Zia et al. [16] compare two different (semi-)dense algorithms quantitatively, KinectFusion [12] and LSD-SLAM [4]. The metrics used are time per frame, absolute trajectory error and energy consumption of the algorithms per frame. Furthermore, the resulting (semi-)dense depth information is compared to ground truth. Fuentes-Pacheo et al. [5] describe different VSLAM algorithms and compare them qualitatively. While [15] sets its focus on the overall SLAM structure, [5] elaborates on the feature extraction component of sparse algorithms and mapping challenges like loop closure or large scale mapping. Huletski et al. [6] provide a qualitative and quantitative evaluation of ORB-SLAM [10], LSD-SLAM [4], and OpenRatSLAM on the TUM-RGB-D dataset. As a metric, both the root mean square error of the trajectory and a number of successful trajectory estimations were used. In their work, ORB-SLAM performed best. Cadena et al. [2] provide a thorough overview of general SLAM systems, common architectures, history of SLAM, different available sensors, semantic interpretation of the results, and possible next steps to tackle the problem of SLAM as a whole. However, no evaluations of specific algorithms are included. Platinsky et al. [13] compare sparse and semi-dense algorithms by constructing a new VO algorithm with two approaches, covering a set of state-of-the-art direct algorithms.

Quantitative evaluations focus either on the positional error, other error metrics or on the energy/resources consumption of the device running the algorithm. Recent work by Delmerico et al. [7] analyzed a number of the presently most prominent visual-inertial estimators in view of a number of different metrics. However, no focus is set on their use for closed loop control nor temporal aspects. Rather, the classical overall drift and memory usage is discussed.

As an extension of the previous evaluations sections, the following papers have a stronger focus on metrics itself.

Sturm et al. [14] investigate the absolute trajectory error (ATE) and the relative pose error (RPE). Together with the overall drift error, these metrics represent the current standard regarding SLAM evaluation. Nardi et al. [11] introduce energy consumption by frame as a metric.

Kümmerle et al. [9] provide a metric used in the KITTI dataset, which, similar to [14], tackles the problem that the overall drift error does not sufficiently cover anything about the error during the flight, especially if loop closing is involved. It considers the error introduced in each pose and disregards cumulative error. It does not consider not temporal properties.

3 IMU Drift Function

In this section, we derive the statistical integration for the expected position error based on both IMU noise and gravity misalignment to provide the mean drift error over time originating from IMU sensor characteristics. This error can be interpreted as the expected position correction a visual-inertial state estimator will perform upon a visual reading. This naturally results in a discontinuity on the time evolution of the estimated state and leads to abrupt behaviours in the closed loop controlled systems.

First, we provide a calculation of the expected position error based on accelerometer noise and bias drift. Second, we include the error induced due to gravity misalignment originating from a wrong attitude estimation due to integration of gyroscope noise and drift. The gravity misalignment error results in a position error due to the wrong subtraction of gravity from and the subsequent wrong integration of the acceleration readings. Thus, the noise and bias drift of the gyroscope contribute to the position error by integrating the error over time 4 or 5 times respectively. We compare our model to both simulated and real-world data. Last, we include the effect on position (i.e. the position error) due to the wrongly calculated attitude, given by the VO algorithm. As roll and pitch are observable in such frameworks, we expect (in the statistical sense) a gravity aligned attitude measurement by the VO. However, the jitters (i.e. noise) with which the VO algorithm estimates the attitude around the gravity aligned mean upon a specific realization of the measurement will introduce additional gravity alignment induced errors in the position until the next measurement.

3.1 IMU Noise Model

To estimate the IMU induced drift, the assumed system dynamics model is described in equation 1.

$$\dot{\boldsymbol{p}} = \boldsymbol{v}, \quad \dot{\boldsymbol{v}} = \boldsymbol{R}^T \boldsymbol{a} - \boldsymbol{g}, \quad \boldsymbol{R} = e^{\boldsymbol{\Omega}_\times}, \quad \dot{\boldsymbol{\Omega}} = \boldsymbol{\omega} \tag{1}$$

With $\boldsymbol{p}$ being the position, $\boldsymbol{v}$ being the velocity, $\boldsymbol{a}$ being the acceleration, $\boldsymbol{g}$ being the gravity vector, $\boldsymbol{R}$ being the rotation in $SO(3)$ from the world frame

to the body frame, $\boldsymbol{\Omega}$ being the rotation in the tangent space $so(3)$ and $\boldsymbol{\omega}$ being the rotational velocity. $\boldsymbol{\Omega}_\times$ is the skew-symmetric 3×3 matrix based on the 3×1 vector $\boldsymbol{\Omega}$.

The true acceleration $\boldsymbol{a}$ can be expressed as $\boldsymbol{a} = \boldsymbol{a_b} + \boldsymbol{Rg}$ with $\boldsymbol{a_b}$ being the acceleration in the body frame and $\boldsymbol{Rg}$ being the gravity in the world frame rotated into the body frame. $\boldsymbol{a_b} = \boldsymbol{Ra_w}$, with $\boldsymbol{a_w}$ being the acceleration of the IMU in the world frame.

Neither $\boldsymbol{a}$ nor $\boldsymbol{\omega}$ can directly be measured, as measurement noise always exists, especially on low-cost IMUs used in UAVs. The model of the error used here is shown in equations 2-4 with a_m and ω_m being the one-dimensional measured linear acceleration and angular velocity respectively (with a and ω as unknown true values). $b_{a,\omega}$ are biases and $n_{a,\omega,b_a,b_\omega}$ are noise parameters. The noise models are assumed to be equal in all axes. The noise model vector $\boldsymbol{a_m}$ is equal to $[a_m, a_m, a_m]^T$ and $\boldsymbol{\omega_m}$ is equal to $[\omega_m, \omega_m, \omega_m]^T$.

$$a_m = a - b_a - n_a, \quad \omega_m = \omega - b_\omega - n_\omega \tag{2}$$

$$\dot{b_a} = n_{b_a}, \quad \dot{b_\omega} = n_{b_\omega}, \quad n_a \sim N(0, \sigma_a^2), \quad n_\omega \sim N(0, \sigma_\omega^2) \tag{3}$$

$$n_{b_a} \sim N(0, \sigma_{b_a}^2), \quad n_{b_\omega} \sim N(0, \sigma_{b_\omega}^2) \tag{4}$$

Note that the bias b is modelled as a Wiener process and the noise n is modeled as zero mean white Gaussian noise. Both have their mean at 0. Because of this, adding or subtracting the error is equivalent in the model.

Based on the IMU and the error model, the position error $\tilde{\boldsymbol{p}}$ and rotation error $\tilde{\boldsymbol{R}}$ can be modeled in equations 5-7.

$$\dot{\tilde{\boldsymbol{p}}} = \tilde{\boldsymbol{v}} \tag{5}$$

$$\dot{\tilde{\boldsymbol{v}}} = \tilde{\boldsymbol{R}}^{\boldsymbol{T}}\boldsymbol{R}^{\boldsymbol{T}}(\boldsymbol{Ra_w} + \boldsymbol{Rg} + \boldsymbol{n_a} + \boldsymbol{b_a}) - \boldsymbol{a_w} - \boldsymbol{g} \tag{6}$$

$$\tilde{\boldsymbol{R}}^{\boldsymbol{T}} = e^{\tilde{\boldsymbol{\Omega}}_\times}, \dot{\tilde{\boldsymbol{\Omega}}} = \boldsymbol{n_w} + \boldsymbol{b_w} \tag{7}$$

For $\dot{\tilde{\boldsymbol{v}}}$, $\boldsymbol{a_w}$ and $\boldsymbol{g}$ are both subtracted so only the erroneous values remain. $\dot{\tilde{\boldsymbol{v}}}$ can be reformulated in equation 8.

$$\dot{\tilde{\boldsymbol{v}}} = \tilde{\boldsymbol{R}}^{\boldsymbol{T}}\boldsymbol{a_w} - \boldsymbol{a_w} + \tilde{\boldsymbol{R}}^{\boldsymbol{T}}\boldsymbol{g} - \boldsymbol{g} + \tilde{\boldsymbol{R}}^{\boldsymbol{T}}\boldsymbol{Rn_a} + \tilde{\boldsymbol{R}}^{\boldsymbol{T}}\boldsymbol{Rb_a} \tag{8}$$

In this equation, $\tilde{\boldsymbol{R}}^{\boldsymbol{T}}\boldsymbol{Rn_a}$ can be reduced to $\boldsymbol{n_a}$ and $\tilde{\boldsymbol{R}}^{\boldsymbol{T}}\boldsymbol{Rb_a}$ to $\boldsymbol{b_a}$. It is assumed that the σ_a is equal for x, y and z direction of $\boldsymbol{n_a}$. This results in the 1σ distance of $\boldsymbol{n_a}$ being a sphere, which is invariant against rotation.

3.2 Average IMU Drift

Having an IMU model, the next step is to estimate the average IMU drift. For this, first, the average of the normally distributed error has to be derived. The probability density function of the normal distribution $f(x|0,1)$ assigns a probability to each possible error value x with the most probable error being 0. With x being an error measure, the absolute error $|x|$ can be used as impact

measure (i.e. how much an error affects the true value, as this is the case for positional errors). Although $x = 0$ is the most probable case, it is also the only case where the impact $|x|$ is 0. While the error $|x|$ for $x > 6\sigma$ is large, it is also very unlikely to occur. Therefore, we suggest to weight the impact of the error with the probability of its occurrence: $|x| \cdot f(x)$. The expectation is then:

$$\mu = \int_{-\infty}^{\infty} |x| \cdot f(x)dx \approx 0.7979 \tag{9}$$

Using this, the average IMU drift can be derived. Reformulating equation 8 for $\tilde{\boldsymbol{p}}$ yields

$$\tilde{\boldsymbol{p}} = \tilde{\boldsymbol{p}}_{\boldsymbol{n_a}} + \tilde{\boldsymbol{p}}_{\boldsymbol{b}_{n_a}} + \tilde{\boldsymbol{p}}_{\boldsymbol{g}} \tag{10}$$

$$\ddot{\tilde{\boldsymbol{p}}}_{\boldsymbol{n_a}} = \boldsymbol{n_a}, \quad \ddot{\tilde{\boldsymbol{p}}}_{\boldsymbol{b}_{n_a}} = \boldsymbol{b}_{\boldsymbol{n_a}}, \quad \ddot{\tilde{\boldsymbol{p}}}_{g} = \tilde{\boldsymbol{R}}^{\boldsymbol{T}}\boldsymbol{g} - \boldsymbol{g} \tag{11}$$

The position error has been split up into the translational noise error $\tilde{\boldsymbol{p}}_{\boldsymbol{n_a}}$, the translational bias error $\tilde{\boldsymbol{p}}_{\boldsymbol{b_a}}$ and the rotational error $\tilde{\boldsymbol{p}}_{\boldsymbol{g}}$ originating from an erroneous allocation of the gravity vector to the wrong axes. The rotational error with respect to the current acceleration $\tilde{\boldsymbol{R}}^{\boldsymbol{T}}\boldsymbol{a_w} - \boldsymbol{a_w}$ is being ignored as $\boldsymbol{a_w}$ is assumed to be significantly smaller than $\boldsymbol{g}$.

With $\boldsymbol{n_a}$ being zero mean white Gaussian noise, the integration over time $\tilde{\boldsymbol{v}}_{\boldsymbol{n_a}}$ is a corresponding Wiener process. Using stochastic integration, $\tilde{\boldsymbol{p}}_{\boldsymbol{n_a}}$ can be found. Similarly, with $\boldsymbol{n}_{\boldsymbol{b_a}}$ being the zero mean white Gaussian noise, $\boldsymbol{b}_{\boldsymbol{n_a}}$ is a Wiener process and both $\tilde{\boldsymbol{v}}_{\boldsymbol{n}_{\boldsymbol{b_a}}}$ and $\tilde{\boldsymbol{p}}_{\boldsymbol{n}_{\boldsymbol{b_a}}}$ are stochastic integrals. t is the time in seconds in the following equations. All of those random variables can be modelled to be Gaussian with the same mean of zero. To estimate the average error, X can be replaced by μ in these equations.

$$\boldsymbol{n_a} \sim N(0, \sigma_a^2) \Rightarrow \boldsymbol{\sigma_a} \cdot X, \quad X \sim N(0, 1) \tag{12}$$

$$\tilde{\boldsymbol{v}}_{\boldsymbol{n_a}} = \boldsymbol{\sigma_a} \cdot \sqrt{t} \cdot X, \quad \tilde{\boldsymbol{p}}_{\boldsymbol{n_a}} = \boldsymbol{\sigma_a} \cdot \sqrt{\frac{t^3}{3}} \cdot X, \quad \tilde{\boldsymbol{p}}_{\boldsymbol{n}_{\boldsymbol{b_a}}} = \boldsymbol{\sigma}_{\boldsymbol{b_a}} \cdot \sqrt{\frac{t^5}{20}} \cdot X \tag{13}$$

As the average error uses the standard deviation for scaling and as both $\tilde{\boldsymbol{p}}_{\boldsymbol{n_a}}$ and $\tilde{\boldsymbol{p}}_{\boldsymbol{n}_{\boldsymbol{b_a}}}$ are modelled as random Gaussian variables, their variances can be added. $\tilde{\boldsymbol{p}}_{\boldsymbol{t}}$ is the resulting average position integrating both n_a and n_{b_a}.

$$\tilde{\boldsymbol{p}}_{\boldsymbol{t}}^2 = \tilde{\boldsymbol{p}}_{\boldsymbol{n_a}}^2 + \tilde{\boldsymbol{p}}_{\boldsymbol{n}_{\boldsymbol{b_a}}}^2 = \boldsymbol{\sigma}_{\boldsymbol{a}}^2 \cdot \frac{t^3}{3} \cdot \mu^2 + \boldsymbol{\sigma}_{\boldsymbol{b_a}}^2 \cdot \frac{t^5}{20} \cdot \mu^2 \tag{14}$$

The rotational error does not contribute directly to the integrated position error. Instead, an erroneous rotation leads to a wrong rotation and subsequent subtraction of the measured gravity. This leads to wrongly perceived body accelerations which, through integration, lead to position errors.

We assume small errors and, thus, that the small angle approximation holds. We can then express the acceleration error due to gravity misalignment originating from rotational errors as linearly dependent on the angular error. An example for pure rotation around the y-axis is as follows (with $\boldsymbol{g} = [0, 0, 9.81]^T$):

$$\begin{bmatrix} cos(\alpha) & 0 & sin(\alpha) \\ 0 & 1 & 0 \\ -sin(\alpha) & 0 & cos(\alpha) \end{bmatrix} \cdot \boldsymbol{g} - \boldsymbol{g} \approx \begin{bmatrix} 1 & 0 & \alpha \\ 0 & 1 & 0 \\ -\alpha & 0 & 1 \end{bmatrix} \cdot \boldsymbol{g} - \boldsymbol{g} = \begin{bmatrix} \alpha \cdot 9.81 \\ 0 \\ 0 \end{bmatrix} \tag{15}$$

Similarly, a linearized dependency can be derived for the gyroscope noise and bias drift. Based on this, it is possible to calculate the integrated coefficient for the gyroscope noise and bias noise depending on time for the expected error. Here, $\tilde{\boldsymbol{\Omega}}_{\boldsymbol{n_\omega}}$ is the integrated linearized position error based on gyroscope noise and $\tilde{\boldsymbol{\Omega}}_{\boldsymbol{n_{b_\omega}}}$ is the integrated linearized position error based on gyroscope bias noise. $\tilde{\boldsymbol{\Omega}}$ is the result of both error sources integrated concurrently.

$$\tilde{\boldsymbol{\Omega}}^2_{\boldsymbol{n_\omega}} = \boldsymbol{\sigma}^2_{\boldsymbol{\omega}} \cdot \frac{t^5}{20} \cdot \mu^2, \quad \tilde{\boldsymbol{\Omega}}^2_{\boldsymbol{n_{b_\omega}}} = \boldsymbol{\sigma}^2_{\boldsymbol{b_\omega}} \cdot \frac{t^7}{252} \cdot \mu^2 \tag{16}$$

$$\tilde{\boldsymbol{\Omega}}^2 = \boldsymbol{\sigma}^2_{\boldsymbol{\omega}} \cdot \frac{t^5}{20} \cdot \mu^2 + \boldsymbol{\sigma}^2_{\boldsymbol{b_\omega}} \cdot \frac{t^7}{252} \cdot \mu^2 \tag{17}$$

We can extend the above notion to the more general $so(3)$ tangent space of the $SO(3)$ group. $so(3)$ consists of three distinct elements in the vector $\tilde{\boldsymbol{\Omega}}$, each of them corresponding to the amount of rotation around one axis in radiants. Using the matrix exponential $e^{\tilde{\boldsymbol{\Omega}}_\times}$, it can be converted into the rotation matrix representation $\tilde{\boldsymbol{R}}$.

This will give an correct estimation up until $\sigma^2_{b_\omega} \cdot \frac{t^7}{252} < \pi/2$. The main reason for the small-angle approximated derivation being valid even for larger angles lies in the fact that the integration can be seen as the repeated application of subsequent infinitesimal rotations. This also corresponds to the property of the $so(3)$ tangent space of $SO(3)$ that multiplications in $SO(3)$ can be reduced to additions in $so(3)$. This derivation has been tested in simulations as well as with real-world data (see Figure 2).

To extract the erroneous acceleration resulting from the rotational misalignment of gravity, the gravity vector $\boldsymbol{g}$ is used with the resulting error rotation matrix $e^{\tilde{\boldsymbol{\Omega}}_\times}$ (note that the standard deviation is used here):

$$\tilde{\boldsymbol{p}}_{\boldsymbol{g}} = e^{\tilde{\boldsymbol{\Omega}}_\times} \boldsymbol{g} - \boldsymbol{g} \tag{18}$$

The average position error $\tilde{\boldsymbol{p}}$ and the average rotation error $\tilde{\boldsymbol{\Omega}}$ depending on time are therefore

$$\tilde{\boldsymbol{\Omega}}(t) = \sqrt{\boldsymbol{\sigma}^2_{\boldsymbol{\omega}} \cdot \frac{t^5}{20} \cdot \mu^2 + \boldsymbol{\sigma}^2_{\boldsymbol{b_\omega}} \cdot \frac{t^7}{252} \cdot \mu^2} \tag{19}$$

$$\tilde{\boldsymbol{p}}(t) = \sqrt{\boldsymbol{\sigma}^2_{\boldsymbol{a}} \cdot \frac{t^3}{3} \cdot \mu^2 + \boldsymbol{\sigma}^2_{\boldsymbol{b_a}} \cdot \frac{t^5}{20} \cdot \mu^2 + (e^{\tilde{\boldsymbol{\Omega}}_\times} \boldsymbol{g} - \boldsymbol{g})^2} \tag{20}$$

In Figure 1 the development of the positional error over time can be seen, based on the values for a MEMS IMU sensor used in a commercial UAV. While the positional noise error e_{n_a} is larger than the positional bias error $e_{b_{n_a}}$ at the beginning, the faster growth of the bias error can be seen at 3 seconds. The

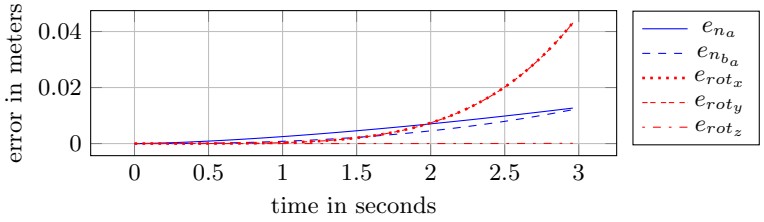

Fig. 1. Growth of different position error sources over time in seconds. Errors are in meters. The following values have been used: $\sigma_a = 0.0054, \sigma_{b_a} = 0.0045, \sigma_\omega = 0.00079, \sigma_{b_\omega} = 0.00008$. e_{rot_x} and e_{rot_y} are overlapping.

positional error based on the wrong handling of the gravity vector is the main source of error in both x and y directions while it is negligible in z direction.

This confirms real-world experiences as a very small drift in z direction compared to the x and y direction has been observed in experiments. The main reason for this difference lies in the structure of the gravity vector: $\boldsymbol{g} = [0, 0, -9.81]^T$. For roll and pitch errors, the sine is applied to the z entry of the gravity vector in y and x direction respectively within the rotation matrix $\tilde{\boldsymbol{R}}^T$ while 1 minus the cosine is applied to the z direction. For the yaw error, the error rotation does not result in a wrong application of parts of the gravity vector and therefore does not contribute to $\tilde{\boldsymbol{p}}_{\boldsymbol{g}}$.

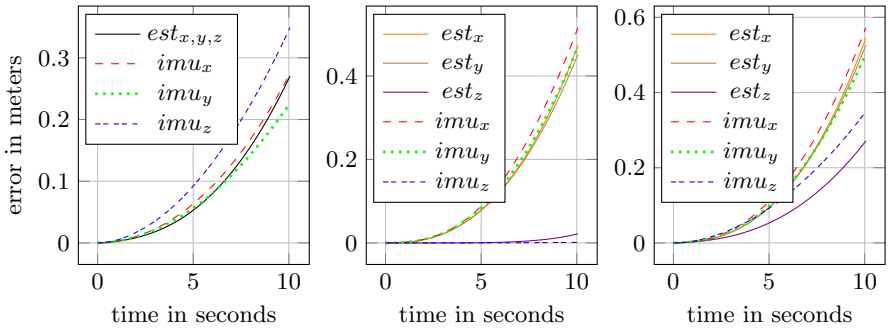

Fig. 2. Comparison of predicted and real-world data. *est* is the predicted data (values from Figure 1) while *imu* is the measured and integrated data over 350 IMU readings. Left is the translational error, to the center is the rotational error and to the right are both errors.

In Figure 2, the predicted positional error compared to the measured positional error can be seen. For the measured positional error, an IMU has collected data while being stationary. For every 10 seconds segment of this data, the biases

have been calculated at the beginning of the segment. The values shown in the graphs are the mean of 350 segments with 10 seconds each.

The left part shows the error introduced by accelerometer noise while the center part shows the error introduced by first integrating the gyroscope noise and then multiplying the gravity vector with the rotational error. The right part shows the result of integrating both errors simultaneously.

3.3 Visual Odometry induced Gravity Misalignment

An additional error source is the inaccurate attitude computation by the visual part, or more generally, the attitude error that remains after an update step. This rotational difference $\tilde{\boldsymbol{R}}_{\boldsymbol{up}}$ multiplied by the gravity vector will introduce further drift in addition to the above discussed mean IMU-induced drift. As only the next (visual) update can correct this error – there is only dead reckoning in between update steps – we model this error as being constant between two update steps.

In our metric, we are interested in the difference of the estimated position before and after the update since the discontinuity occurring at (i.e. right after) the update step affects the underlying controller. Due to the dead-reckoning in between update steps the maximum drift will have happened right before the update has been processed.

$\tilde{\boldsymbol{a}}_{\boldsymbol{up}}$ is the erroneous acceleration resulting from a misaligned gravity vector after an update step. Integrating twice over time results in the position error $\tilde{\boldsymbol{p}}_{\boldsymbol{up}}$.

$$\tilde{\boldsymbol{a}}_{\boldsymbol{up}} = |\tilde{\boldsymbol{R}}_{\boldsymbol{up}} \cdot \boldsymbol{g} - \boldsymbol{g}|, \quad \tilde{\boldsymbol{p}}_{\boldsymbol{up}}(t) = \frac{t^2}{2} \cdot \tilde{\boldsymbol{a}}_{\boldsymbol{up}} = \frac{t^2}{2} \cdot |\tilde{\boldsymbol{R}}_{\boldsymbol{up}} \cdot \boldsymbol{g} - \boldsymbol{g}| \tag{21}$$

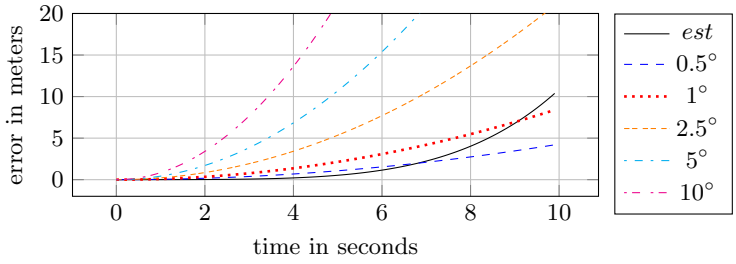

Fig. 3. Position error in x direction assuming an erroneous rotation around the y axis. *est* shows the estimated IMU drift error with values from figure 1.

Figure 3 shows typical position errors evolving over time assuming different magnitudes of remaining angular errors after an update step. It can be seen that it may take tens of seconds or even minutes until the above discussed IMU drift error surpasses the gravity misalignment error due to the errors remaining after an update step.

4 IMU-Drift based Metrics

4.1 Modified Relative Pose Error

As mentioned earlier, we assume that a visual odometry algorithm is combined with an IMU on a quadcopter. In such a system, discontinuities in the state estimation occur upon each visual measurement. Such discontinuities in the estimated state result in fast corrective actions of the underlying controller which could break both the controller and the VO/VSLAM algorithm (e.g. due to motion blur because of the sudden movements). Thus, we assume – from a closed loop control point of view – that the smaller the discontinuities are, the better the performance of the estimator is.

A metric with a similar goal already exists in literature. The relative pose error (RPE) [14] can be described as the difference between the position change according to ground truth and the position change according to the VO algorithm. In the equation 22, $\boldsymbol{p_a}$ is the estimated position while $\boldsymbol{p_{gt}}$ is the ground truth position.

$$m_{rpe_{VO}} = \frac{1}{n}\sum_{i=1}^{n} ||abs(\boldsymbol{p_{gt_i}} - \boldsymbol{p_{gt_{i-1}}}) - abs(\boldsymbol{p_{a_i}} - \boldsymbol{p_{a_{i-1}}})||_2 \tag{22}$$

For our purposes, we model that the correct part of the estimated pose change has been handled by the correct part of the IMU measurements read between two subsequent updates. Between two poses, IMU drift as well as drift based on gravity misalignment occurs. We extend the existing metric with these two values. This extended metric has three advantages:

- The time component is brought into the metric, preferring updates with a higher frequency over those with a lower frequency. Furthermore, it takes the processing time to generate an update into account as the correction can only be used when it is available, even if it is applied retroactively based on the time stamp of the camera image.
- The roll and pitch errors results in a position error, making both comparable in a meaningful way. Instead of only measuring the position error, the whole pose error excluding yaw is used.
- Because of the estimated drifts, the incorporation of time is weighted so that not the fastest algorithm is preferred, but instead the one resulting in the lowest overall error producing the least amount of distance change before and after an update.

The extension itself builds upon the work in the previous chapter with Δt being the time between two updates and T being the overall time:

$$m_{rpe_{drift}}(t) = abs(abs(\boldsymbol{p_{gt_i}} - \boldsymbol{p_{gt_{i-1}}}) - abs(\boldsymbol{p_{a_i}} - \boldsymbol{p_{a_{i-1}}})) \tag{23}$$

$$m_{rpe_{drift}} = \sum \frac{\Delta t}{T} ||m_{rpe_{drift}}(\Delta t) + \tilde{\boldsymbol{p}}(\Delta t) + \tilde{\boldsymbol{p}}_{\boldsymbol{up}}(\Delta t)||_2 \tag{24}$$

4.2 Modified Absolute Trajectory Error

While our modified RPE calculates the magnitude of the discontinuity between two pose estimates, it does not take the difference between estimated value and ground truth into account. It may be interesting to see the mean divergence from ground truth over time.

For this, we extend another metric, the absolute trajectory error (ATE) [14] which can be described as the translational difference between the estimated and ground truth trajectory, assuming both trajectories have been aligned using a linear least squares approach to minimize the error.

$$m_{ate_{VO}} = \frac{1}{n}\sum_{i=1}^{n} ||\boldsymbol{p_{gt_i}} - \boldsymbol{p_{a_i}}||_2 \tag{25}$$

$$m_{ate_{drift}} = \sum \frac{\Delta t}{T} ||abs(\boldsymbol{p_{gt_{i-1}}} - \boldsymbol{p_{a_{i-1}}}) + \tilde{\boldsymbol{p}}(\Delta t) + \tilde{\boldsymbol{p}}_{\boldsymbol{up}}(\Delta t)||_2 \tag{26}$$

Note that the above developed metrics are not constrained to vision based estimators. In fact, as long as the IMU is used as a state propagation sensor together with one or more measurement/update sensors, these metrics can be applied. In the following section, however, we limit the demonstration to application of the metrics to loosely coupled vision based sensor fusion systems.

5 Applying the Metrics

To evaluate our proposed metrics we selected multiple camera pose estimation algorithms and performed tests in real-world environment. The algorithms were executed on an Odroid-XU4. This computing device offers 8 CPU cores (4x2GHz, 4x1.4GHz), ARM architecture and the low weight from 38g makes is suitable for usage on micro aerial vehicles.

As testdata we use the *EuRoC* [1] dataset. It was created with a multi-rotor UAV equipped with a VI-Sensors. It offers camera (stereo 752x480, ∼20Hz), IMU (∼200Hz) and ground truth measurements for a total of 11 sequences in 3 scenarios. Because of the similarity of our assumed scenario - multi-rotor UAV flight in GPS denied scenarios - we chose it as our main evaluation dataset. From the sequences we selected four sequences where 6 DoF ground truth is available from a motion capture system. We selected these particular four sequences because all tested algorithms provided acceptable failure rate for multiple runs.

We selected 3 state-of-the-art visual odometry or SLAM algorithms with different approaches to camera pose estimation to evaluate the influence of our new metric on the results.

ORB-SLAM by Mur-Artal et al. [10] combines current state-of-the-art components to create a sparse SLAM algorithm (similar to PTAM [8]). It uses feature detection with binary descriptors and bag-of-words for mapping and loop closure. For our evaluations we used ORB-SLAM2 which is a computationally improved version of ORB-SLAM still adhering to the same principles and components.

LSD-SLAM by Engel et al. [4] is a direct, semi-dense, graph-based algorithm. Image alignment is achieved using a Gauss-Newton optimization to minimize the photometric error using depth information. This depth map is kept in a keyframe-based approach with each measurement improving the depth estimation within a keyframe.

DSO by Engel et al. [3] is a direct, sparse, graph-based method. It has similarities to LSD-SLAM with the main difference being sparsity. Instead of selecting all available image gradients, a set of points is chosen by splitting the image into $d \times d$ blocks with d being an adaptive factor. In each block, the pixel with the largest gradient is selected as long as it surpasses a certain threshold. This point is tracked and intensity differences are used to estimate the position in a Gauss-Newton procedure.

For each algorithm, we carried out 12 runs and averaged the results which are shown in table 1. Runs in which the algorithms failed to generate pose estimation for the majority of the sequences were discarded. The generated trajectories were scaled, rotated and aligned to the ground truth trajectory so that the squared error is minimized. All metrics are calculated on the aligned trajectories.

We considered the latency m_l in seconds, the absolute trajectory error m_{ate} for both the original version $m_{ate_{VO}}$ and our drift extension $m_{ate_{drift}}$ as well as the relative pose error m_{rpe} for both $m_{rpe_{VO}}$ and $m_{rpe_{drift}}$, all of them in millimeters. The latency is the time needed by the algorithm from receiving an image to the estimated pose.

Each entry in the table was generated by using the mean over all updates for multiple runs. DSO has the lowest latency of the compared algorithms, followed by ORB2. In terms of the absolute trajectory error $m_{ate_{VO}}$, ORB2 performed better than the other algorithms. This also fits to the fact that ORB2 has the most sophisticated mapping component compared to the other two algorithms. While LSD and DSO both use a windowed bundle adjustment approach, ORB2 fits the data against an iteratively filled map which reduced the trajectory drift.

Compared to the first sequences of each scenario, $v1\_02$ and $v2\_02$ have faster movements and rotations. Because of the limited computational power of the target platform the VO algorithms perform worse when facing more challenging movements.

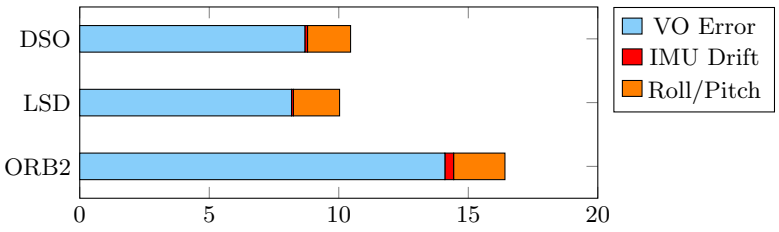

Fig. 4. Error sources for different algorithms on the v1_01 scenario according to the $m_{rpe_{drift}}$ metric in mm. IMU Drift based on IMU noise, Roll/Pitch on VO rotational error based gravity misalignment (rpe_{up}).

	m_l $[ms]$	$m_{ate_{VO}}$ $[mm]$	$m_{ate_{drift}}$ $[mm]$	$m_{rpe_{vo}}$ $[mm]$	$m_{rpe_{drift}}$ $[mm]$
			DSO		
v1_01	92	144.803	146.648	7.528	8.729
v1_02	81	391.209	392.446	32.549	33.869
v2_01	90	131.584	133.571	13.323	16.085
v2_02	80	157.105	158.317	19.370	20.623
			ORB2		
v1_01	186	25.237	27.635	11.989	14.149
v1_02	164	28.500	29.438	21.268	22.237
v2_01	172	100.727	106.129	21.154	27.418
v2_02	161	147.211	152.946	42.061	47.435
			LSD		
v1_01	156	319.002	324.386	7.846	13.164
v2_01	46	217.178	218.859	8.400	9.964
			Aggregation		
DSO	84	304.423	306.474	20.247	22.335
ORB2	175	104.980	110.257	25.351	30.595
LSD	108	269.520	271.648	9.562	11.837

Table 1. Results over multiple runs per sequence and algorithm. m_l is latency, $m_{ate_{VO}}$ and $m_{ate_{drift}}$ are ATE without and with our drift extension, $m_{rpe_{VO}}$ and $m_{rpe_{drift}}$ are RPE without and with our drift extension.

Figure 4 shows the $m_{rpe_{drift}}$ error for the $v1\_01$ scenario for DSO, ORB2 and LSD, separated into the visual odometry based position error $m_{rpe_{VO}}$, the IMU drift error and the gravity misalignment error based on roll and pitch error. The IMU drift error is very small compared to the roll and pitch error. This corresponds to the $100-200$ milliseconds needed to generate a new VO measurement. In such a time frame, the IMU noise was not yet able to accumulate enough drift to have an impact compared to the other two error sources.

Using our extended metric, it can be seen that the costs roughly increase by a tenth compared to only using the VO position error. This mostly corresponds to the roll and pitch error over the duration of the latency. Because the differences between the algorithms are larger than the additionally introduced error, this does not change the order. It also fits to the observation that with at least ten frames per second, smooth flight is possible.

Selecting less frames per second would increase the impact of our metric. However, we saw in practice that the algorithms used are already quite fragile regarding tracking failure, which is also the main reason for using the easier sequences. Lower frame rate further decreases the amount of successful runs.

Comparing the error increase of the other three metrics, DSO performs best. This is mainly because the low latency results in less time available to apply position drift.

6 Conclusion

In this paper we introduced novel metrics to evaluate the performance of VSLAM/VO algorithms for IMU aided state estimation and subsequent closed loop control of agile mobile platforms.

Existing work predominantly focuses on the pose estimation performance of such algorithms without considering their use for closed loop control in autonomous navigation scenarios. Thus, with the proposed metrics, VSLAM/VO algorithms can be evaluated according to their real performance in a visual-inertial setup for vehicle control.

We particularly focused on the newly defined and suggested metric of IMU induced drift which accounts for the position offset due to IMU integration errors based on the probabilistic properties of the inertial sensors. This metric favors fast yet accurate vision updates at all time and, thus, combines at the same time algorithm robustness (few measurement drops), high algorithmic speed (low computational complexity and thus low latency), and estimation precision in a single metric. The latter results from the requirement of propagating the state (aid of IMU readings) to the current time in order to feed the vehicle controller with most recent state estimates. This even holds for delay compensated frameworks applying the visual measurement at the correct time in the past.

Using our set of metrics, we tested and compared three different state-of-the-art algorithms on multiple scenarios. We showed that the proposed set of metrics is general and applicable on any framework generating time-discrete pose updates. Regarding the IMU-drift metric, our example evaluations have shown that a multitude of information previously shown in different metrics (latency or framerate, position error, rotation error) has been combined in a single, meaningful way including temporal information.

References

1. Burri, M., Nikolic, J., Gohl, P., Schneider, T., Rehder, J., Omari, S., Achtelik, M.W., Siegwart, R.: The EuRoC micro aerial vehicle datasets. The International Journal of Robotics Research (2016). https://doi.org/10.1177/0278364915620033, http://ijr.sagepub.com/content/early/2016/01/21/0278364915620033.abstract
2. Cadena, C., Carlone, L., Carrillo, H., Latif, Y., Scaramuzza, D., Neira, J., Reid, I., Leonard, J.J.: Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age. IEEE Transactions on Robotics **32**(6), 1309–1332 (2016)
3. Engel, J., Koltun, V., Cremers, D.: Direct sparse odometry. IEEE Transactions on Pattern Analysis and Machine Intelligence **40**(3), 611–625 (2018). https://doi.org/10.1109/TPAMI.2017.2658577
4. Engel, J., Schöps, T., Cremers, D.: LSD-SLAM: Large-scale direct monocular SLAM. In: European Conference on Computer Vision (ECCV) (September 2014)
5. Fuentes-Pacheco, J., Ruiz-Ascencio, J., Rendón-Mancha, J.M.: Visual simultaneous localization and mapping: a survey. Artificial Intelligence Review **43**(1), 55–81 (2015)

6. Huletski, A., Kartashov, D., Krinkin, K.: Evaluation of the modern visual slam methods. In: 2015 Artificial Intelligence and Natural Language and Information Extraction, Social Media and Web Search FRUCT Conference (AINL-ISMW FRUCT). pp. 19–25 (Nov 2015). https://doi.org/10.1109/AINL-ISMW-FRUCT.2015.7382963
7. J. Delmerico, D.S.: A benchmark comparison of monocular visual-inertial odometry algorithms for flying robots. In: Robotics and Automation (ICRA), IEEE International Conference on. IEEE (2018)
8. Klein, G., Murray, D.: Parallel tracking and mapping for small ar workspaces. In: IEEE and ACM International Symposium on Mixed and Augmented Reality. pp. 225–234 (2007). https://doi.org/10.1109/ISMAR.2007.4538852
9. Kümmerle, R., Steder, B., Dornhege, C., Ruhnke, M., Grisetti, G., Stachniss, C., Kleiner, A.: On measuring the accuracy of SLAM algorithms. Autonomous Robots **27**(4), 387–407 (2009)
10. Mur-Artal, R., Montiel, J.M.M., Tards, J.D.: Orb-slam: A versatile and accurate monocular slam system. IEEE Transactions on Robotics **31**(5), 1147–1163 (Oct 2015). https://doi.org/10.1109/TRO.2015.2463671
11. Nardi, L., Bodin, B., Zia, M.Z., Mawer, J., Nisbet, A., Kelly, P.H.J., Davison, A.J., Lujn, M., O'Boyle, M.F.P., Riley, G., Topham, N., Furber, S.: Introducing SLAMBench, a performance and accuracy benchmarking methodology for SLAM. In: IEEE International Conference on Robotics and Automation. pp. 5783–5790 (May 2015). https://doi.org/10.1109/ICRA.2015.7140009
12. Newcombe, R.A., Izadi, S., Hilliges, O., Molyneaux, D., Kim, D., Davison, A.J., Kohli, P., Shotton, J., Hodges, S., Fitzgibbon, A.: Kinectfusion: Real-time dense surface mapping and tracking. In: Proceedings of the 2011 10th IEEE International Symposium on Mixed and Augmented Reality. pp. 127–136. ISMAR '11, IEEE Computer Society, Washington, DC, USA (2011). https://doi.org/10.1109/ISMAR.2011.6092378, http://dx.doi.org/10.1109/ISMAR.2011.6092378
13. Platinsky, L., Davison, A.J., Leutenegger, S.: Monocular visual odometry: Sparse joint optimisation or dense alternation? In: IEEE International Conference on Robotics and Automation. pp. 5126–5133 (2017). https://doi.org/10.1109/ICRA.2017.7989599
14. Sturm, J., Engelhard, N., Endres, F., Burgard, W., Cremers, D.: A benchmark for the evaluation of rgb-d slam systems. In: IEEE/RSJ International Conference on Intelligent Robots and Systems. pp. 573–580 (2012). https://doi.org/10.1109/IROS.2012.6385773
15. Younes, G., Asmar, D.C., Shammas, E.A.: A survey on non-filter-based monocular visual SLAM systems. CoRR **abs/1607.00470** (2016), http://arxiv.org/abs/1607.00470
16. Zia, M.Z., Nardi, L., Jack, A., Vespa, E., Bodin, B., Kelly, P.H., Davison, A.J.: Comparative design space exploration of dense and semi-dense slam. In: Robotics and Automation (ICRA), 2016 IEEE International Conference on. pp. 1292–1299. IEEE (2016)