OKVIS 2.0 for the FPV Drone Racing VIO Competition 2020

Stefan Leutenegger[1]*

Abstract
In the following, the Visual-Inertial SLAM approach OKVIS 2.0 is briefly described. It was used to compute the trajectories submitted to the 2020 FPV Drone Racing Competition. In summary, the estimator was designed as a sliding-window optimisation scheme, where error terms from the Inertial Measurement Unit (IMU) as well as visual reprojection errors are jointly minimised, along with relative pose error terms derived from marginalisation of old observations. The method further supports visual place recognition and respective loop-closure optimisation, where the edges of the pose graph correspond to the previously mentioned relative pose errors. Furthermore, the estimator makes use of online extrinsics estimation, which was found particularly useful in the setting at hand. The implementation was designed to be deployed in real-time on a state-of-the art mobile processor. Therefore, *causal* trajectories are reported, i.e. for the computation of any given state, no information from future measurements are taken into account – noting that the framework naturally extends to full trajectory smoothing including a full expansion into a visual-inertial Bundle Adjustment problem.

Keywords
Visual-Inertial SLAM - Drones

[1] *Department of Computing, Imperial College, London, United Kingdom*
***Corresponding author**: s.leutenegger@imperial.ac.uk

1. Introduction

For space constraints, a lengthy introduction and description of the state of the art are omitted. For an excellent summary of both the used UZH-FPV Drone Racing Dataset, as well as a survey of Visual-Inertial (VI) SLAM, the reader is referred to [1]. The datasets supports and encourages the use of the mounted event camera, naturally lending itself to tracking of highly aggressive motion, where conventional cameras suffer from motion blur and large image content change from frame to frame. While the author believes the inclusion of event data into OKVIS 2.0 would be very promising, it is left for future work. The interested reader is referred to [2] for a comprehensive overview of event cameras and their application to robot vision.

In the following, OKVIS 2.0 as applied to the FPV Challenge is described, characterised by the following contributions:

- A real-time capable VI SLAM system, the C++ implementation of which is scheduled for release in 2020.
- A novel tightly coupled sliding-window VI estimator based on non-linear least-squares optimisation of visual reprojection errors, pre-integrated IMU measurements, as well as relative pose error terms obtained from marginalisation of past observations.
- A place recognition module based on DBoW2 [3] using the same BRISK 2.0 [4, 5] visual keypoints and descriptors as in the frontend tracking.
- Background loop closure optimisation that contains the same visual, inertial, and relative pose error terms as the real-time tracking estimator, thus seamlessly integrating the two without the need for approximative pose graph edge creation.

2. Notation and Definitions

The VI SLAM problem consists of tracking a moving body with a mounted IMU and N cameras relative to a static World coordinate frame $\underrightarrow{\mathcal{F}}_W$. The IMU coordinate frame is denoted as $\underrightarrow{\mathcal{F}}_S$ and the camera frames as $\underrightarrow{\mathcal{F}}_{C_i}$, $i = 1 \dots N$.

Left-hand indices denote coordinate representation. Homogeneous position vectors can be transformed with $\boldsymbol{T}_{AB}$, meaning ${}_A\mathbf{r}_P = \boldsymbol{T}_{AB}\,{}_B\mathbf{r}_P$.

The following state representation is used:

$$\mathbf{x} = [{}_W\mathbf{r}_S^T, \mathbf{q}_{WS}^T, {}_W\mathbf{v}^T, \mathbf{b}_\mathrm{g}^T, \mathbf{b}_\mathrm{a}^T]^T, \tag{1}$$

where ${}_W\mathbf{r}_S$ denotes the position of the origin of $\underrightarrow{\mathcal{F}}_S$ relative to $\underrightarrow{\mathcal{F}}_W$, $\mathbf{q}_{WS}$ is the Hamiltonian Quaternion of orientation describing the attitude of $\underrightarrow{\mathcal{F}}_S$ relative to $\underrightarrow{\mathcal{F}}_W$, and ${}_W\mathbf{v}$ stands for the velocity of $\underrightarrow{\mathcal{F}}_S$ relative to $\underrightarrow{\mathcal{F}}_W$. Furthermore, the rate gyro biases $\mathbf{b}_\mathrm{g}$ and accelerometer biases $\mathbf{b}_\mathrm{a}$ are included.

The state is estimated at every time step k where frames from the camera(s) are obtained.

3. VI Estimator

In the following, the different components of the VI Estimator are outlined. The frontend, as well as visual and inertial error terms are largely adopted from OKVIS [6]. Jacobians are omitted for brevity and will be described in a full report describing OKVIS 2.0. The non-linear least squares costs as described below are minimised using Google's Ceres Solver [7].

Reprojection Error

We use the standard reprojection error $\mathbf{e}_\mathrm{r}^{i,j,k}$ of the j-th landmark ${}_W\mathbf{l}^j$ into the i-th camera image at time step k:

$$\mathbf{e}_\mathrm{r}^{i,j,k} = \tilde{\mathbf{z}}_\mathbf{r}^{i,j,k} - \mathbf{h}(\boldsymbol{T}_{SC_i}^{-1}\,\boldsymbol{T}_{S^kW}\,{}_W\mathbf{l}^j), \tag{2}$$

with the keypoint detection $\tilde{\mathbf{z}}_\mathbf{r}^{i,j,k}$ and $\mathbf{h}(.)$ denoting the camera projection, where in this work pinhole projection is used, optionally with distortion (radial-tangential or equidistant).

IMU Error

The IMU error $\mathbf{e}_\mathrm{s}^k$ between time instance k and r is used:

$$\mathbf{e}_\mathrm{s}^{k,r} = \mathbf{x}^r \boxminus \hat{\mathbf{x}}^r(\mathbf{x}^k, \tilde{\mathbf{z}}_\mathrm{s}^k)), \tag{3}$$

with $\hat{\mathbf{x}}^r(\mathbf{x}^k)$ denoting the predicted state at step $r = k + n$ based on the estimate $\mathbf{x}^k$ and the IMU measurements (rate gyro and accelerometer readings) $\tilde{\mathbf{z}}_\mathrm{s}^k$. A formulation of this error term using a pre-integration scheme adopted from [8] is used, rendering its evaluation tractable for any number of IMU samples used.

Relative Pose Error

Furthermore, relative pose errors $\mathbf{e}_\mathrm{p}^{k,r}$ between time steps k and r are used:

$$\mathbf{e}_\mathrm{p}^{k,r} = [{}_{S^k}\mathbf{r}_{S^r}^T, \log \mathbf{q}_{S^kS^r}]^T. \tag{4}$$

Realtime Estimation Problem

The realtime estimator running at least at camera frame rate will minimise the following non-linear least squares cost

$$J(\mathbf{x}) = \frac{1}{2}\sum_i \sum_{k\in\mathcal{K}} \sum_{j\in\mathcal{J}(i,k)} \mathbf{e}^{i,j,k}{}_\mathrm{r}^T \mathbf{W}_\mathrm{r} \mathbf{e}^{i,j,k}{}_\mathrm{r} + \frac{1}{2}\sum_{k,r\in\mathcal{S}} \mathbf{e}^{k,r}{}_\mathrm{s}^T \mathbf{W}_\mathrm{s} \mathbf{e}_\mathrm{s}^{k,r} + \frac{1}{2}\sum_{k,r\in\mathcal{P}} \mathbf{e}^{k,r}{}_\mathrm{p}^T \mathbf{W}_\mathrm{p} \mathbf{e}_\mathrm{p}^{k,r}. \tag{5}$$

Here, $\mathbf{W}_\mathrm{r}$ stands for the visual weight as the inverse covariance of the reprojection error, and the set $\mathcal{K}$ denotes all active frames, i.e. poses with observations of the respectively visible landmarks in the set $\mathcal{J}(i,k)$. $\mathcal{K}$ contains the T most recent frames as well as M keyframes in the past. A new frame is selected as a keyframe based on a co-visibility criterion with currently active keyframes.

IMU errors are considered for all (key-)frames k, r in succession (denoted by the set $\mathcal{S}$). The IMU error weight $\mathbf{W}_\mathrm{s}$ is obtained from linear error propagation as part of the IMU error (pre-)integration.

The set $\mathcal{P}$ contains the pose graph frames, i.e. those linked together with relative pose errors obtained from original co-observations. The weight $\mathbf{W}_\mathrm{p}$ is computed from marginalisation of old observations.

To keep the problem complexity bounded, frames exhibiting least co-visibility with the current frame or current keyframe are moved from $\mathcal{K}$ to $\mathcal{P}$ by marginalisation of common observations under insertion of relative pose errors. Furthermore, in order to keep the number of poses being estimated limited, only the A most recent frames are kept variable.

Note that we may leave the camera extrinsics $\boldsymbol{T}_{SC_i}$ as variables to be optimised, i.e. performing online calibration.

Place Recognition and Loop Closure

Whenever a query of the current frame to the DBoW2 [3] database returns a match, and geometric verification using 3D-2D RANSAC passes, the currently active window of states and landmarks is transformed to be re-aligned to the matched pose. Then, the co-observations with the matched frame are re-inserted as landmarks and observations, i.e. adding the recognised frame back as a keyframe into the set $\mathcal{K}$.

Then, the background optimisation is started, using the very same cost as (5), however, with different fixation of states (i.e. the states inside

the closed loop remain variable). Note how all the IMU error terms are also considered in this loop closure optimisation, adding further constraints, most notably regarding consistency of the orientation relative to gravity.

4. Experimental Results

We use the stereo-visual-inertial setup as provided by the Snapdragon sensor assembly, without the use of any information from the event camera.

All experiments were conducted with online extrinsics calibration enabled, as it was found to vary from dataset to dataset sufficiently to deteriorate accuracy (most notably the scale) when left fixed. All estimator parameters were left unchanged across the experiments; however, the BRISK 2 detection threshold was changed from 35 to 45 for the outdoor sequences, as naturally more texture occurs; furthermore, the Harris score noise rejection threshold was slightly varied between the three categories of datasets, as the image noise was found to vary. Both of these could, however, be selected automatically with some basic image processing engineering, if desired.

A maximum of 10 keyframes was used (maximum 5 loop-closure frames and 5 regular keyframes), plus $T = 3$ most recent frames; $A = 10$ variable states are used to optimise over.

Note that results are reported as *causal*, i.e. only using measurements up to the time of the computed pose. This leads to non-smooth jumps upon loop-closure that are easily visible.

Trajectory Accuracy

The main results regarding quantitative accuracy will be visible from the competition. For completeness, however, a representative example is reported here, namely the `indoor_forward_7` sequence. Please see Fig. 1 and Fig. 2 for indicative results obtained with the open-source implementation of [9]. As shown, position accuracies in the single-digit percentage of distance travelled can be obtained for challenging yet loopy trajectories.

Timings

The trajectories submitted to the competition were computed on a three-year old 13 inch ThinkPad laptop equipped with a quad-core Intel Core i7-7500U CPU @ 2.70GHz running Ubuntu 16.04. The multi-threaded C++ implementation was compiled with GCC 5.4.0. Timings include data loading, a simple visualiser, and writing of the submitted files. Full processing times per sequence are reported in Table 1. This is close to real-time and could be run at the necessary 26 Hz with some further tweaks or on a slightly better processor.

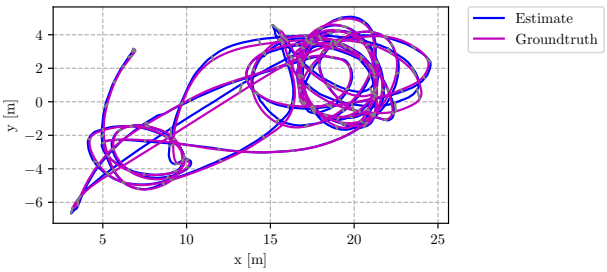

Figure 1. Overhead estimate and ground truth. Trajectory aligned in position and yaw.

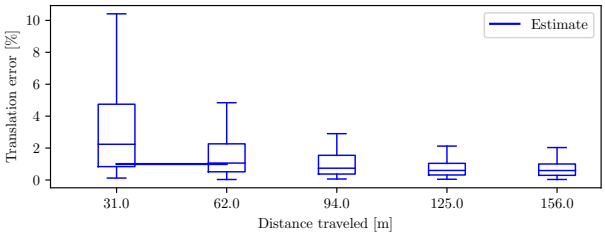

Figure 2. Translation errors as percentage of distance travelled.

Table 1. OKVIS 2 Timings

Sequence	Processing time
`indoor_forward_11`	130.19 s
`indoor_forward_12`	103.08 s
`indoor_45_3`	106.55 s
`indoor_45_16`	65.79 s
`outdoor_forward_9`	153.9 s
`outdoor_forward_10`	220.54 s

Conclusions

OKVIS 2.0, a VI SLAM system was presented as applied to the 2020 FPV Drone Racing Competition. It uses joint minimisation of visual, inertial, and relative pose error terms and features both a frame-by-frame component, as well as a place recognition part triggering background loop-closure optimisation using the same error terms. It can operate in real-time on a state-of-the-art CPU, as illustrated by the timings. The evaluation results on the UZH-FPV dataset with public ground truth are promising; but it will have to be seen how competitive they are in comparison to other submissions.

References

[1] Jeffrey Delmerico, Titus Cieslewski, Henri Rebecq, Matthias Faessler, and Davide Scaramuzza. Are we ready for autonomous drone racing? the uzh-fpv drone racing dataset. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 6713–6719. IEEE, 2019.

[2] Guillermo Gallego, Tobi Delbruck, Garrick Orchard, Chiara Bartolozzi, Brian Taba, Andrea Censi, Stefan Leutenegger, Andrew Davison, Joerg Conradt, Kostas Daniilidis, et al. Event-based vision: A survey. *arXiv preprint arXiv:1904.08405*, 2019.

[3] Dorian Gálvez-López and J. D. Tardós. Bags of binary words for fast place recognition in image sequences. *IEEE Transactions on Robotics*, 28(5):1188–1197, October 2012.

[4] Stefan Leutenegger, Margarita Chli, and Roland Y Siegwart. Brisk: Binary robust invariant scalable keypoints. In *2011 International conference on computer vision*, pages 2548–2555. Ieee, 2011.

[5] Stefan Leutenegger. *Unmanned solar airplanes: Design and algorithms for efficient and robust autonomous operation*. PhD thesis, ETH Zurich, 2014.

[6] Stefan Leutenegger, Simon Lynen, Michael Bosse, Roland Siegwart, and Paul Furgale. Keyframe-based visual–inertial odometry using nonlinear optimization. *The International Journal of Robotics Research*, 34(3):314–334, 2015.

[7] Sameer Agarwal, Keir Mierle, and Others. Ceres solver. http://ceres-solver.org.

[8] Christian Forster, Luca Carlone, Frank Dellaert, and Davide Scaramuzza. On-manifold preintegration for real-time visual–inertial odometry. *IEEE Transactions on Robotics*, 33(1):1–21, 2016.

[9] Zichao Zhang and Davide Scaramuzza. A tutorial on quantitative trajectory evaluation for visual(-inertial) odometry. In *IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS)*, 2018.