2018 IEEE International Conference on Robotics and Automation (ICRA)
May 21-25, 2018, Brisbane, Australia

Collaborative 6DoF Relative Pose Estimation for two UAVs with Overlapping Fields of View

Marco Karrer[1], Mohit Agarwal[1], Mina Kamel[2], Roland Siegwart[2] and Margarita Chli[1]

[1]Vision for Robotics Lab and [2]Autonomous Systems Lab, ETH Zurich, Switzerland

Abstract— **Driven by the promise of leveraging the benefits of collaborative robot operation, this paper presents an approach to estimate the relative transformation between two small Unmanned Aerial Vehicles (UAVs), each equipped with a single camera and an inertial sensor, comprising the first step of any meaningful collaboration. Formation flying and collaborative object manipulation are some of the few tasks that the proposed work has direct applications on, while forming a variable-baseline stereo rig using two UAVs carrying a monocular camera each promises unprecedented effectiveness in collaborative scene estimation.**

Assuming an overlap in the UAVs' fields of view, in the proposed framework, each UAV runs monocular-inertial odometry onboard, while an Extended Kalman Filter fuses the UAVs' estimates and common image measurements to estimate the metrically scaled relative transformation between them, in real-time. Decoupling the direction of the baseline between the cameras of the two UAVs from its magnitude, this work enables consistent and robust estimation of the uncertainty of the relative pose estimation. Our evaluation on both on simulated data and benchmarking datasets consisting of real aerial data, reveals the power of the proposed methodology in a variety of scenarios.

Video – `https://youtu.be/Amkk8X826oI`

I. Introduction

The capability of a robot to estimate its pose in a previously unknown environment, while simultaneously mapping the scene, commonly referred to as Simultaneous Localization and Mapping (SLAM), is a key enabler of autonomous navigation. While initially the problem of SLAM was addressed using range sensors, it was with the emergence of real-time capable SLAM systems using a monocular camera, such as [7] and [11] that SLAM onboard small Unmanned Aerial Vehicles (UAV) started being investigated. Using the combination of both visual measurements as well as readings from an Inertial Measurement Unit (IMU), [18] demonstrated the strength of this combination in a vision controlled flight using only onboard sensing capabilities, rendering Visual-Inertial (VI) sensing as the preferred choice for the control and the navigation of a small UAV. Current state of the art Visual Inertial Odometry (VIO) systems, such as OKVIS [13] and ROVIO [3], have reached a remarkable maturity and robustness and their publicly available implementations had great impact in the community. Most recently, the work of [14] presented a complete VI-SLAM system achieving global consistency of the map at metric scale.

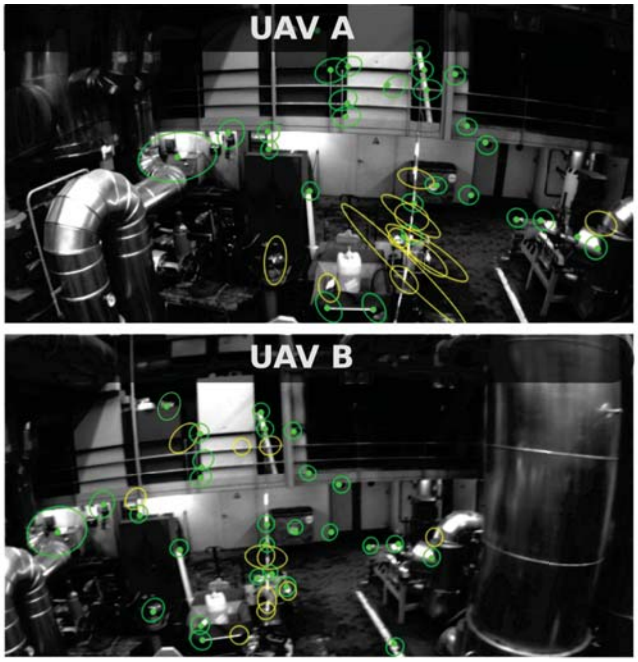


Fig. 1: A snapshot of the proposed system on the EuRoC [4] dataset, showing the camera views of the two UAVs. Green points correspond to map landmarks successfully detected in image space. The ellipses indicate the projected uncertainty of the predicted landmark positions in the image plane, with yellow corresponding to any unmatched landmarks. Fusing cues from monocular-inertial odometry running onboard each UAV with cross-camera correspondences into an EKF framework, the relative pose between the two UAVs gets estimated in real-time.

Alongside with the increased maturity of single-robot SLAM systems, the community started to gain interest in collaborative SLAM using multiple robots. The system proposed in [19] for example, leverages view overlap from multiple individual moving cameras to perform SLAM in challenging dynamic scenes. While the system introduces very powerful ideas, the reliance on a GPU and the absence of metric scale limits the applicability of the approach in robotic application, especially on UAVs. The approaches of [10] and [16] demonstrate the use of multiple UAVs, each equipped with a monocular camera to perform efficient mapping and collaborative SLAM, respectively. Whereas these systems use a centralized architecture, ongoing research has been studying the SLAM estimation process in a distributed fashion aiming for scalability of the approach with the number of agents. While [6] presented a method to achieve consistency using only peer-to-peer communication and as-

This research was supported by EC's Horizon 2020 Programme under grant agreement n. 644128 (AEROWORKS) and the Swiss National Science Foundation (SNSF, Agreement no. PP00P2_157585)

978-1-5386-3081-5/18/$31.00 ©2018 IEEE

suming known correspondences, [8] showed a distributed SLAM pipeline on multiple UAVs utilizing 2D laser scans for mapping.

Consistent mapping from multiple UAVs is necessary for planning the movements of a swarm of robots, e.g. coordinating their trajectories for efficient exploration, however, for a wide variety of tasks, such as collaborative manipulation of objects using multiple UAVs or formation flying, knowing the relative pose between UAVs is most important. The work of [9] aims at estimating the relative pose between a UAV and a ground robot by observing a set of LED markers mounted on the UAV with a monocular camera on the ground robot. While [9] relies on direct measurements between the robots, [1] presented an approach to fuse inertial readings together with scaled relative pose measurements obtained via overlap in the Field of View (FoV) in the images from two UAVs.

In this paper, we present an approach to estimate the relative transformation between two UAVs following the footsteps of [1] by utilizing overlap between the images captured from two UAVs in combination with inertial data. As demonstrated by [17], optimization based approaches to visual odometry allow the estimation over a larger set of state variables more efficiently than filter based methods. However, as shown by [3], including a low number of landmarks in the filter state enables a computationally inexpensive state estimation while exhibiting a considerable robustness. Furthermore, the employment of a filter as opposed to an optimization based method allows the extraction of a probability distribution without the need for additional computations, offering a significant advantage in a robotic scenario, e.g. to define safety margins to avoid collisions in a formation flight. Motivated by this, we base our approach on an Extended Kalman Filter (EKF) to fuse local odometry estimates from two UAVs together with the image measurements to estimate the metrically scaled relative transformation between the two UAVs in real-time. The proposed system is completely expressed in local coordinates, reducing the influence of drift stemming from the local odometry system. Furthermore, in contrast to [1], we demonstrate and evaluate the proposed approach both on simulated data as well as on a benchmarking dataset consisting of real data captured with a UAV.

The main contribution of this work is the presented EKF design for the relative pose estimation, where we express the baseline between the two UAVs using a bearing vector and an inverse distance parametrization for the magnitude of the baseline, allowing to reflect the uncertainty of the estimation problem in a natural and intuitive way. Finally, the presented system design is shown to be capable of running in real-time with two UAVs, only requiring peer-to-peer communication, while sharing some of the computational load.

II. Problem Setup

A. Notation

In this paper, small letters are used for scalars (x), capital letters for coordinate frames (X) according to their definition in Figure 2, bold capitals for matrices ($\boldsymbol{X}$), and bold small letters for vectors and unit quaternions ($\boldsymbol{x}$). An arbitrary variable $\boldsymbol{x}_{\mathrm{id}}^{\mathrm{k}}$ can be characterized by its timestamp k and some identifier id, which is used to either label the variable (e.g. $\boldsymbol{x}_1, \boldsymbol{x}_2$,...) or to distinguish the coordinate frame it is expressed in (e.g. $\boldsymbol{x}_{\mathrm{A}}, \boldsymbol{x}_{\mathrm{B}}$,...). In order to differentiate between predicted and updated state variables at each timestamp, we use the following convention:

- $\hat{\boldsymbol{x}}^{k+1}$: the predicted state variable at time $k+1$ given the posterior state and system input at time k.
- $\boldsymbol{x}^{k+1}$: the updated state variable at time $k+1$ given the measurements obtained at time $k+1$.

For the representation of rotations, we use unit quaternions, denoted by $\boldsymbol{q}$. Concatenations of two quaternions are denoted by $\boldsymbol{q}_1 \circ \boldsymbol{q}_2$ and the rotation of a vector $\boldsymbol{v}$ by the quaternion $\boldsymbol{q}$ is denoted by $\boldsymbol{q}(\boldsymbol{v})$.

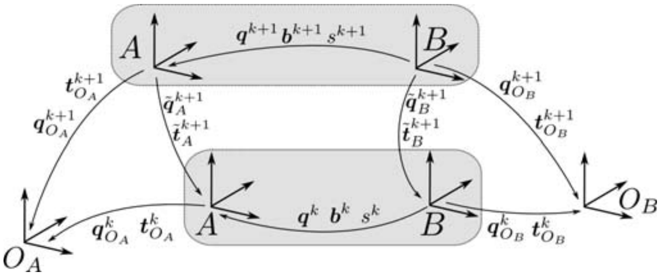

A
$\boldsymbol{q}^{k+1}\ \boldsymbol{b}^{k+1}\ s^{k+1}$
B
$\boldsymbol{t}_{O_A}^{k+1}$
$\boldsymbol{q}_{O_A}^{k+1}$
$\bar{\boldsymbol{q}}_A^{k+1}$
$\bar{\boldsymbol{t}}_A^{k+1}$
$\bar{\boldsymbol{q}}_B^{k+1}$
$\bar{\boldsymbol{t}}_B^{k+1}$
$\boldsymbol{q}_{O_B}^{k+1}$
$\boldsymbol{t}_{O_B}^{k+1}$
O_B
$\boldsymbol{q}_{O_A}^{k}\ \boldsymbol{t}_{O_A}^{k}$
A
$\boldsymbol{q}^{k}\ \boldsymbol{b}^{k}\ s^{k}$
B
$\boldsymbol{q}_{O_B}^{k}\ \boldsymbol{t}_{O_B}^{k}$
O_A

Fig. 2: The transformations and the coordinate frames used in this paper, with the shaded regions marking the pair of UAVs used as a setup here at two consecutive timestamps (k and $k+1$). The coordinate frames O_A and O_B correspond to the local odometry origins for the UAV A and UAV B, respectively.

B. System Requirements and Assumptions

The proposed system considers two UAVs, each equipped with one monocular camera onboard and a module producing metrically scaled 6-Degree of Freedon (DoF) egomotion estimation, such as Visual Inertial Odometry (VIO) and the ability to to exchange data amongst each other (e.g. over WiFi). While our system does not impose any general constraints on the movements of the UAVs, we make the following assumptions:

- the cameras experience overlap in their fields of view most of the time,
- the system clocks as well as the cameras of the two UAVs are synchronized, and
- the egomotion estimation for both UAVs is stable at all times.

Although a stable egomotion estimation is assumed, the odometry estimates can be noisy and drift over time and we do not require any prior knowledge about the transformation of their local origins.

III. Relative Pose Filter Setup

A. State Representation and Parametrization

As the goal of the filter is to estimate the relative pose of the two UAVs independently of their global position, the reference frame for all state variables is chosen to be the camera coordinate frame of UAV A (A). In this work, we co-estimate a number of landmarks from both UAVs enabling tight coupling between the updates arising from visual measurements and the predictions computed by odometry. The filter state, when there are N visual landmarks tracked, is:

$$\boldsymbol{x} := \begin{pmatrix} \boldsymbol{b} & s & \boldsymbol{q} & \boldsymbol{\mu} & \cdots & \boldsymbol{\mu} & \varrho_1 & \cdots & \varrho_N \end{pmatrix} \ , \tag{1}$$

where the following state variables are used:

- $\boldsymbol{b}$: the bearing vector of the baseline from B to A, expressed in A,
- s: the inverse of the metric magnitude of the baseline between the two monocular cameras mounted on the two UAVs,
- $\boldsymbol{q}$: the relative rotation from B to A,
- $\boldsymbol{\mu}_i$: the bearing vector of landmark i in frame A, and
- ϱ_i: the inverse depth of landmark i in frame A.

In order to avoid singularities in the state variables, we express rotations as $\boldsymbol{q} \in SO(3)$ and the bearing vectors as $\boldsymbol{b}, \boldsymbol{\mu} \in S^2$, while both are parameterized as quaternions. To obtain a minimal representation of the state covariance matrix, the lie algebra and the tangent space representations are used for rotations and bearing vectors, respectively. For a detailed explanation and analytical expressions of the used parameterization, we refer the interested reader to [2].

Note that we do not only represent the landmarks using the inverse distance parametrization, but also employ the same parametrization to represent the baseline of the relative transformation. This is especially advantageous at the initialization stage of the filter, as we can get a direct measurement on the direction from only one pair of frames, but not on the metrically scaled distance. The baseline of the relative transformation between UAV A and UAV B is intrinsically expressed by the decoupled nature of the (inverse) distance and the bearing vector formulation.

B. State Prediction

For the state prediction, we utilize the poses as obtained by the local odometry systems as

$$\tilde{\boldsymbol{t}}_i^{k+1} = \left(\boldsymbol{q}_{Oi}^k\right)^{-1}\left(\boldsymbol{t}_i^{k+1} - \boldsymbol{t}_i^k\right) + \boldsymbol{\delta}_t, \quad \boldsymbol{\delta}_t \sim \mathscr{N}(0, \boldsymbol{\Sigma}_t) \tag{2}$$

$$\tilde{\boldsymbol{q}}_i^{k+1} = \left(\left(\boldsymbol{q}_{Oi}^k\right)^{-1} \circ \boldsymbol{q}_{Oi}^{k+1}\right) \boxplus \boldsymbol{\delta}_q, \quad \boldsymbol{\delta}_q \sim \mathscr{N}(0, \boldsymbol{\Sigma}_q) \ , \tag{3}$$

where the subscript i is used to differentiate the inputs from UAV A and B. The $\boxplus$ operator denotes the generalization of the addition operation for rotations as in [2]. We assume the noise covariances $\boldsymbol{\Sigma}_t$ and $\boldsymbol{\Sigma}_q$ of the computed relative transformations to be diagonal. For simplicity, we summarize the odometry predictions to an input vector given by

$$\boldsymbol{u}^k := \begin{pmatrix} \tilde{\boldsymbol{t}}_A^{k+1} & \tilde{\boldsymbol{q}}_A^{k+1} & \tilde{\boldsymbol{t}}_A^{k+1} & \tilde{\boldsymbol{q}}_B^{k+1} \end{pmatrix} \ . \tag{4}$$

Using the obtained odometry estimates for the two UAVs, the state prediction is performed by closing the transformation loop:

$$\hat{\boldsymbol{b}}^{k+1} = \left(\tilde{\boldsymbol{q}}_A^{k+1}\right)^{-1}\left(\boldsymbol{q}^k(\tilde{\boldsymbol{t}}_B^{k+1}) + \frac{1}{s^k}\boldsymbol{b}^k - \tilde{\boldsymbol{t}}_A^{k+1}\right)/c_b \tag{5}$$

$$\hat{s}^{k+1} = \frac{1}{c_b} \tag{6}$$

$$\hat{\boldsymbol{q}}^{k+1} = \left(\tilde{\boldsymbol{q}}_A^{k+1}\right)^{-1} \circ \boldsymbol{q}^k \circ \tilde{\boldsymbol{q}}_B^{k+1} \tag{7}$$

$$\hat{\boldsymbol{\mu}}_i^{l+1} = \tilde{\boldsymbol{q}}_A^{k+1}\left(\frac{1}{\varrho_i^k}\boldsymbol{\mu}_i^k - \tilde{\boldsymbol{t}}_A^{k+1}\right)/c_i \tag{8}$$

$$\hat{\varrho}_i^{k+1} = \frac{1}{c_i} \ , \tag{9}$$

where the intermediate normalization constants c_b, c_i are given by

$$c_b := \left\| \boldsymbol{q}^k(\tilde{\boldsymbol{t}}_B^k) + \frac{1}{s^k}\boldsymbol{b}^k - \tilde{\boldsymbol{t}}_A^{k+1} \right\|_2, \quad c_i := \left\| \frac{1}{\varrho_i^k}\boldsymbol{\mu}_i^k - \tilde{\boldsymbol{t}}_A^{k+1} \right\|_2 \tag{10}$$

Since the used model is already discrete in time, the EKF equations for the covariance prediction can be directly applied:

$$\hat{\boldsymbol{P}}^{k+1} = \boldsymbol{F}^k \boldsymbol{P}^k \ {\boldsymbol{F}^k}^T + \boldsymbol{G}^k \boldsymbol{Q}^k \ {\boldsymbol{G}^k}^T \ , \tag{11}$$

where $\boldsymbol{Q}^k \in \mathbb{R}^{12\times12}$ is the covariance matrix of the odometry noise obtained by stacking $\boldsymbol{\Sigma}_q$ and $\boldsymbol{\Sigma}_t$ for both UAVs. The matrices $\boldsymbol{F}^k$ and $\boldsymbol{G}^k$ are the jacobians of the predicted state with respect to the state variables and the odometry input, respectively:

$$\boldsymbol{F}^k := \frac{\partial \hat{\boldsymbol{x}}^{k+1}}{\partial \boldsymbol{x}^k} \in \mathbb{R}^{6+3N\times 6+3N} \tag{12}$$

$$\boldsymbol{G}^k := \frac{\partial \hat{\boldsymbol{x}}^{k+1}}{\partial \boldsymbol{u}^k} \in \mathbb{R}^{6+3N\times 12} \ , \tag{13}$$

C. State Update

For every pair of images from the two UAVs, an update of the state is performed. Assuming that the camera intrinsics for both UAVs are known, the mapping between a bearing vector $\boldsymbol{\mu}$ in the camera's coordinate frame and the corresponding pixel coordinates $\boldsymbol{p}$ in the image is given by

$$\boldsymbol{p} = \pi\left(\boldsymbol{\mu}\right) \ . \tag{14}$$

Using the reprojection error in the image plane as a measurement, we can formulate the residuals as

$$\boldsymbol{r}_{i,j}^{k+1} = \boldsymbol{z}_{i,j}^{k+1} - h_j(\hat{\boldsymbol{x}}^{k+1}) \ , \tag{15}$$

where $\boldsymbol{z}_{i,j}^{k+1}$ denotes the measured detection of landmark i in image plane of UAV j and $h_j(\cdot)$ corresponds to the prediction of the measurement, given as

$$h_A(\hat{\boldsymbol{x}}^{k+1}) = \pi_A(\hat{\boldsymbol{\mu}}_i^{k+1}) \tag{16}$$

for the prediction in UAV A's image and

$$h_B(\hat{\boldsymbol{x}}^{k+1}) = \pi_B\left(\hat{\boldsymbol{q}}^{k+1}\left(\frac{1}{\hat{\varrho}_i^{k+1}}\hat{\boldsymbol{\mu}}_i^{k+1} - \frac{1}{\hat{s}^{k+1}}\hat{\boldsymbol{b}}^{k+1}\right)^{-1}\right) \tag{17}$$

for the measurement prediction in UAV B's image frame. The residual vector $\boldsymbol{r}^{k+1}$ is obtained by stacking all residual terms for both UAVs. Similarly, we obtain the observation matrix $\boldsymbol{H}^{k+1} \in \mathbb{R}^{4N\times 6+3N}$ by stacking the individual jacobians of the predicted measurements given by

$$\boldsymbol{H}_{i,j}^{k+1} = \frac{\partial h_i(\hat{\boldsymbol{x}}^{k+1})}{\partial \hat{\boldsymbol{x}}^{k+1}} \in \mathbb{R}^{2\times 6+3N} \ . \tag{18}$$

Together with the predicted covariance, we can compute the residual covariance given by

$$\boldsymbol{S}^{k+1} = \boldsymbol{H}^{k+1} \hat{\boldsymbol{P}}^{k+1} \, {\boldsymbol{H}^{k+1}}^T + \boldsymbol{O}^{k+1} \ , \tag{19}$$

with $\boldsymbol{O}^{k+1} = \operatorname{diag}\left(\dots, \sigma^2_{z_{j,i}}, \sigma^2_{z_{j,i}}, \dots\right) \in \mathbb{R}^{4N\times 4N}$ representing the stacked covariances of the measured landmark detections. At this stage, we perform a Mahalanobis distance based outlier detection, allowing to reject spurious matches by comparing the obtained residuals with the predicted residual covariance. Using the the residual covariance, the Kalman gain $\boldsymbol{K}$ is computed by

$$\boldsymbol{K}^{k+1} = \hat{\boldsymbol{P}}^{k+1} \, {\boldsymbol{H}^{k+1}}^T \, {\boldsymbol{S}^{k+1}}^{-1} \ . \tag{20}$$

Given the Kalman gain, the updated state is computed using

$$\boldsymbol{x}^{k+1} = \hat{\boldsymbol{x}}^{k+1} \oplus \boldsymbol{K}^{k+1} \boldsymbol{r}^{k+1} \ , \tag{21}$$

where we use the operator $\oplus$ to indicate the use of $+$ for vector and scalar states and the $\boxplus$ operation for rotations and bearing vectors. The updated state covariance is then computed using

$$\boldsymbol{U} = \boldsymbol{I} - \boldsymbol{K}^{k+1}\boldsymbol{H}^{k+1} \tag{22}$$

$$\boldsymbol{P}^{k+1} = \boldsymbol{U}\hat{\boldsymbol{P}}^{k+1}\boldsymbol{U}^T + \boldsymbol{K}^{k+1}\boldsymbol{O}^{k+1} \, {\boldsymbol{K}^{k+1}}^T \tag{23}$$

IV. System Design

In order to keep the computational complexity of the filter bounded, the proposed approach maintains a constant number of landmarks in the state for the estimation. Therefore, as landmarks can get out of the overlapping field of view during motion, we employ a heuristic strategy to dynamically initialize new and replace any old landmarks that have not been measured consistently during operation. The rest of the section describes these processes in detail.

A. Computation Architecture and Keypoint Detection

One of the main motivations of this work is to be able to run an odometry system onboard of each UAV independently, permitting fail-safe control regardless of communication issues of (e.g. network delays). In this way, in the case that collaboration is not possible, the UAVs can still be able to stabilize themselves, while when the UAVs experience sufficient overlap in their fields of view, the proposed filter estimates the relative pose of the two UAVs leading to collaborative mapping. We propose to perform the execution of the filter onboard UAV A, which forms our reference frame. However, keypoint detection can be performed independently on each camera feed and thus, can be run onboard the each UAV independently, distributing some of the computation. In the proposed system, the keypoints are detected using customized Harris corner detection used in [13] and described via an ORB descriptor extraction [15]. Therefore, UAV B forms messages only summarizing its local odometry pose estimate together with its keypoint locations and their descriptors and transmit these to UAV A. As a reference, in our setup each keypoint requires 40 Bytes, 32 for the ORB descriptor and 8 for the keypoint location. With a upper limit on the detected keypoints of 500 and 20Hz framerate, adding up to a bandwidth requirement of 0.4 MB/s, which is easily feasible using standard WiFi modules (e.g. IEEE 802.11g standard).

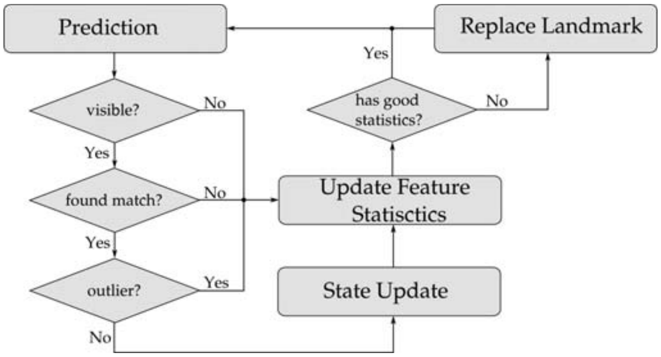


Fig. 3: Workflow of the dynamic landmark replacement in the filter state. All steps are performed for both UAVs and the decision for landmark replacement is based on the combined statistics.

B. State Initialization

For the initialization of the filter, we assume that a rough estimate of the magnitude of the UAVs' baseline is available. As both the bearing vector and the relative orientation can be estimated from a single pair of images, we establish a set of 2D correspondences via descriptor matching between the images of the two UAVs captured at the same time. As described in [13], the descriptor matching of binary descriptors can be performed very fast, permitting brute-force search for correspondences across the two images. Relative RANSAC is then performed using [12] on the matched keypoints to reject outliers and compute the relative transformation between the two UAVs. This transformation is scaled to the magnitude of the initial guess for the baseline. Finally, we attempt to triangulate any remaining matches using the scaled initial pose. If enough matches can be triangulated, we select a random subset for the initialization and set the state variables according to the relative pose and the selected triangulated landmarks, respectively. Initially, the covariance of the metric magnitude of the baseline s is set to a large nominal value, while the covariances of $\boldsymbol{b}$ and $\boldsymbol{q}$ are set to smaller values as in contrast to s, these are measured during the initialization. The initial covariances of the landmarks are obtained by considering the uncertainty of the (relative) pose used for the triangulation. In order to be able to re-detect landmarks, we associate each landmark to both of the keypoint descriptors used to triangulate it.

C. Matching and Landmark Management

Since landmark positions are estimated in the state, their predicted positions at the next timestamp are used to obtain

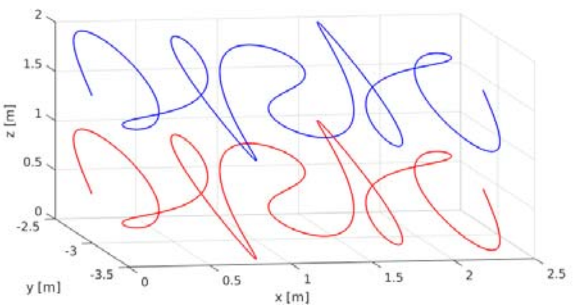


Fig. 4: Simulated trajectory of the two UAVs for the experiment with a constant relative transformation.

new landmark measurements for the state update. This is performed by projecting the landmarks into the corresponding image of each UAV and searching for candidate keypoint matches within a radius, which is chosen reflecting the projected uncertainty of the landmark. During matching, the descriptor used for each landmark is the one obtained during the initialization of this landmark. The candidate with the lowest descriptor distance to the query landmark that is smaller than a threshold is considered to be a match.

For the decision on when to replace a landmark, we employ a heuristic approach as illustrated in Figure 3 accounting for different failures of the detection of each landmark. In essence, each landmark has two counters associated to it; one for each UAV-agent, which gets incremented by a fixed value in case a test fails and decreases (until the minimum of zero) if a successful update is carried out. The increment for a failure at different cases is increasing, e.g. an outlier is weighted stronger than a landmark that is projected outside of the image. Both counters are independent from each other, however, at the decision whether a landmark needs to be replaced, the counter with the higher value is decisive. Upon the decision to replace landmarks, we perform again brute force matching on the keypoints without a landmark association, followed by checking the distance to the epipolar line in order to reject outliers. The remaining correspondences are used for triangulation, after which we select the landmark(s) used to newly insert in the state, at random. As described in Section IV-B, we associate the two corresponding feature descriptors to the landmark and insert it into the state.

V. Experimental Results

In this section, results using both simulated as well as real data using the EuRoC [4] benchmarking dataset are presented. For both experiments we set the number of landmarks estimated in the filter state to be 40.

A. Simulation Setup

In the simulation, we setup an environment consisting of a set of 3D-landmarks randomly distributed in a plane. We generate ground truth trajectories for both UAVs and construct artificial frames with ground truth for both the keypoints' positions obtained by projecting the map landmarks, as well as for the matching correspondences given by an identifier of the projected landmark. This allows to test the filter performance without any uncertainty introduced in matching. The odometry input is computed by disturbing the relative poses between two subsequent frames with a noise of $\sigma_t = 0.005m$ and $\sigma_q = 0.1°$. Furthermore, the keypoint positions are disturbed by zero-mean gaussian noise with 2 pixels standard deviation. In this setup, we performed two experiments: (i) one maintaining a constant relative transformation between the UAVs, while performing an constant movement along the x-axis and sinusoidal excitation both along the camera axis as well as the z-axis, as shown in Figure 4, and (ii) one, where UAV A performs the same motion as in (i), while UAV B moves in a similar manner, but with an offset resulting in a somewhat oscillating relative transformation.

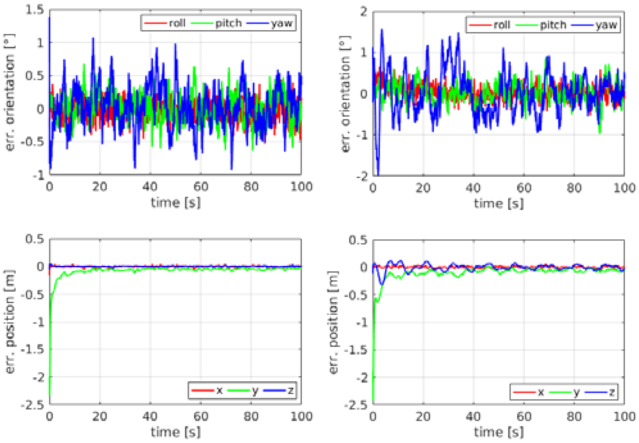
B. Real Data Experimental Setup

For the experiments using real-data, we utilize the publicly available EuRoC dataset [4], which consists of different sequences recorded from a UAV flying various trajectories in two different environments, both for a room sized scenario (Vicon Room) and in a larger industrial environment (Machine Hall). The onboard sensor suite captures stereo WVGA images at $20Hz$ from a global shutter camera along with inertial readings at $200Hz$ from a hardware-synchronized IMU.

The Vicon Room sequences are not suitable for these experiments, as the UAV is equipped with a forward-looking camera exhibiting predominantly fast rotations rendering it impossible to guarantee overlapping fields of view. Therefore, we evaluate the proposed system on the Machine Hall scenario, namely on the sequences *MH_01_easy* and *MH_02_easy*. As the dataset consists of single UAV trajectories and there is only sporadic view overlap amongst different sequences, we simulate the two UAVs using one EuRoC sequence for one UAV and the same one for the other UAV, but with a time-offset, choosing the left stereo camera of the sensor suite for UAV A and the right one for UAV B. We test with a time-offset of 1s for both sequences, as well as 2s time-offset for *MH_02_easy*, leading to reasonable baselines and sufficient variation, while maintaining sufficient view overlap throughout each test. All experiments are run on a single computer, however, using the same architecture as described in Section IV-A, i.e. data exchange is performed via internal memory access instead of WiFi. For the VIO estimation of both UAVs, we use ROVIO [3]. Note that the approach is agnostic to the used VIO system.

C. Results and Discussion

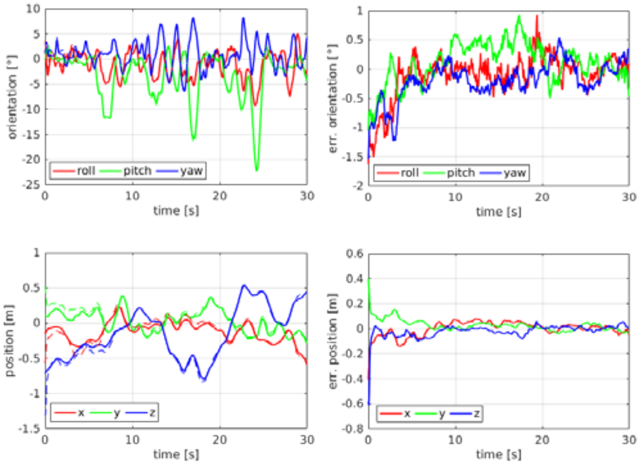
The results for both simulated experiments are shown in Figure 5, while the averaged errors are reported in Table I. Note that for the computation of the Root Mean Squared Error (RMSE), only estimates obtained after the convergence of the filter are considered. As it can be seen, the filter is able to converge in both cases. In contrast to the approach in [1], the proposed system is able to converge even without relative motion. However, as it is evident on the angular error for the full experiment, the relative orientation converges



(a) Constant distance

(b) Full motion

Fig. 5: The resulting errors converted into roll-pitch-yaw angles and Euclidean coordinates. (a) shows the error for the experiment with a constant relative transformation, while (b) shows the result for the experiment with some oscillating relative motion.



(a) Trajectory

(b) Error

Fig. 6: The initial part of the *MH_02_easy* sequence with $1s$ offset.(a) shows the relative transformation over time, where the estimated transformation is plotted with dashed line and the ground-truth with the continuous line, and (b) shows the resulting errors on the relative transformation. Note that the error in orientation is too small for the dashed line to be visible in the top left figure.

coupled with the position, which is related to the fact that in our system the relative poses of the UAVs are estimated via landmark estimation.

For the real-data experiments, a resulting error plot for the initial part of the sequence *MH_02_easy* with a time offset of $1s$ is shown in Figure 6. After an initial error on the relative distance of about $1m$, the error, both on translation and rotation decreases initially quickly and reaches a convergent state after approximately $8s$. However, during our experiments we could observe that the convergence rate usually is variable, depending on the initial configuration, but also on the initial matches, which are chosen at random as outlined in Section IV-B. The averaged RMSE for both rotation and translation are reported in Table I. The average errors for the $1s$ offset experiments are approximately $0.5°$ on the rotation and $0.06m$ on the translation. On the sequence *MH_02_easy* with a $2s$ offset, the errors increase compared to the experiments with $1s$ offset, which can be attained to two reasons: (a) the baseline is generally larger indirectly influencing the error, as, for example, any scale error in the estimation results in proportional errors in the absolute distance of the baseline, and (b) the viewpoint changes are larger, resulting to a smaller overlap, which in turn, results in shorter feature tracks (i.e. on average a larger number of landmarks gets replaced per frame pair).

	RMSE $\boldsymbol{q}$ [°]	RMSE $\boldsymbol{t}$ [m]
Sim. Constant	0.39	0.054
Sim. Dynamic	0.56	0.071
MH_01_easy (1s)	0.63	0.058
MH_02_easy (1s)	0.48	0.063
MH_02_easy (2s)	0.82	0.093

TABLE I: Average RMS errors of the relative transformations for both the simulation and the real experiments. The rotational errors are obtained by transforming in the roll-pitch-yaw formulation, while the translation errors $\boldsymbol{t}$ are obtained by transforming $\boldsymbol{b}$ and s in Euclidean coordinates. All results are recorded by averaging over 5 runs.

	Prediction	Matching	Update	Landmark Manag.
mean [ms]	1.31	0.15	7.51	4.49
std [ms]	0.29	0.05	0.95	4.41
max [ms]	3.60	1.03	13.76	12.66

TABLE II: The execution time of the proposed framework, broken down into the different steps. The reported statistics correspond to the real experiments, when using 40 landmarks in the state.

To evaluate the complexity of the proposed algorithm, the runtime was recorded over all real-data sequences. The statistics, broken down into the different parts of the algorithm are shown in Table II. All timings are obtained on an Intel Core i7-4710MQ with 16GB RAM running at 2.5GHz. On average, the total execution time per frame pair is approximately $13.5ms$, resulting in an average frame rate of $74Hz$, whereas the maximum time per frame pair overall is $25ms$. With a frame rate of $20Hz$ our algorithm can therefore, easily run in real-time. The largest variation in timings correspond to the landmark management, since the 2D-2D matching is only executed upon the initialization of new landmarks, as described in Section IV-C. Note that the timings for the keypoint detection and descriptor extraction, which together require $7.5ms$ on average per frame, are not included in Table II, as this process runs in parallel to the actual algorithm.

VI. Conclusion

This work presents an EKF design enabling real-time 6-degree-of-freedom relative pose estimation between two UAVs with overlapping fields of view, using local odometry estimates along with monocular vision measurements. Along with the filter design, we propose a new, minimal parametrization of the baseline between the UAVs' cameras as a bearing vector and an inverse-distance, enabling a consistent representation of the uncertainty of the estimation problem. The outlined lightweight system is designed to run onboard two UAVs only using peer-to-peer communication with a bandwidth requirement under 0.5MB/s, while distributing some of the computational load. We demonstrate the capability of the proposed system both on simulated data as well as on real data on the EuRoC benchmarking dataset.

Future work will address the landmark management system in order to improve the performance through longer feature tracks by storing a larger number of landmarks than in the actual filter state (e.g. as in [3]). Furthermore, we believe the robustness of the system could be boosted by incorporating more informed feature selection in the spirit of [5].

References

[1] M. W. Achtelik, S. Weiss, M. Chli, F. Dellaert, and R. Siegwart. Collaborative Stereo. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2011.

[2] M. Bloesch, M. Burri, S. Omari, M. Hutter, and R. Siegwart. Iterated extended Kalman filter based visual-inertial odometry using direct photometric feedback. In *International Journal of Robotics Research (IJRR)*, 2017.

[3] M. Bloesch, S. Omari, M. Hutter, and R. Siegwart. ROVIO: Robust Visual Inertial Odometry Using a Direct EKF-Based Approach. In *Proceedings of the IEEE/RSJ Conference on Intelligent Robots and Systems (IROS)*, 2015.

[4] M. Burri, J. Nikolic, P. Gohl, T. Schneider, J. Rehder, S. Omari, M. W. Achtelik, and R. Siegwart. The EuRoC micro aerial vehicle datasets. In *International Journal of Robotics Research (IJRR)*, 2016.

[5] L. Carlone and K. Sertac. Attention and anticipation in fast visual-inertial navigation. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2017.

[6] A. Cunningham, V. Indelman, and F. Dellaert. DDF-SAM 2.0: Consistent distributed smoothing and mapping. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2013.

[7] A. J. Davison, N. D. Molton, I. Reid, and O. Stasse. MonoSLAM: Real-time single camera SLAM. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 29(6):1052–1067, 2007.

[8] J. Dong, E. Nelson, V. Indelman, N. Michael, and D. Frank. Distributed real-time cooperative localization and mapping using an uncertainty-aware expectation maximization approach. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2015.

[9] M. Faessler, E. Mueggler, K. Schwalbe, and D. Scaramuzza. A Monocular Pose Estimation System based on Infrared LEDs. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2014.

[10] C. Forster, S. Lynen, L. Kneip, and D. Scaramuzza. Collaborative Monocular SLAM with Multiple Micro Aerial Vehicles. In *Proceedings of the IEEE/RSJ Conference on Intelligent Robots and Systems (IROS)*, 2013.

[11] G. Klein and D. Murray. Parallel Tracking and Mapping for Small AR Workspaces. In *Proceedings of the International Symposium on Mixed and Augmented Reality (ISMAR)*, 2007.

[12] L. Kneip, D. Scaramuzza, and R. Siegwart. OpenGV: A unified and generalized approach to real-time calibrated geometric vision. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2014.

[13] S. Leutenegger, P. Furgale, V. Rabaud, M. Chli, and R. Siegwart. Keyframe-based Visual-Inertial SLAM using Nonlinear Optimization. In *Proceedings of Robotics: Science and Systems (RSS)*, 2013.

[14] R. Mur-Artal and J. D. Tardós. Visual-Inertial Monocular SLAM with Map Reuse. In *IEEE Robotics and Automation Letters*, 2017.

[15] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski. ORB: An efficient alternative to SIFT or SURF. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2011.

[16] P. Schmuck and M. Chli. Multi-UAV Collaborative Monocular SLAM. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2017.

[17] H. Strasdat, J. Montiel, and A. J. Davison. Real-time monocular SLAM: Why filter? In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2010.

[18] S. Weiss, M. W. Achtelik, S. Lynen, M. C. Achtelik, L. Kneip, M. Chli, and R. Siegwart. Monocular Vision for Long-term MAV Navigation: A Compendium. *Journal of Field Robotics (JFR)*, 30:803–831, 2013.

[19] D. Zou and P. Tan. CoSLAM: Collaborative Visual SLAM in Dynamic Environments. In *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 2013.