VI-RPE: Visual-Inertial Relative Pose Estimation for Aerial Vehicles

Journal Article

Author(s):
Teixeira, Lucas; Maffra, Fabiola; Moos, Marco; Chli, Margarita

Publication date:
2018-10

Permanent link:
https://doi.org/10.3929/ethz-b-000264548

Rights / license:
In Copyright - Non-Commercial Use Permitted

Originally published in:
IEEE Robotics and Automation Letters 3(4), https://doi.org/10.1109/LRA.2018.2837687

Funding acknowledgement:
644128 - Collaborative Aerial Robotic Workers (SBFI)
157585 - Collaborative vision-based perception for teams of (aerial) robots (SNF)

VI-RPE: Visual-Inertial Relative Pose Estimation for Aerial Vehicles

Lucas Teixeira, Fabiola Maffra, Marco Moos and Margarita Chli

***Abstract*—With a large body of literature dedicated to ego-motion estimation and perception of a robot's workspace, the Robotics community has seen some impressive advances in self-localization and mapping, however, we are still far from general applicability of such approaches in real scenarios. Driven by the need for portable and low-cost solutions to relative pose estimation between Unmanned Aerial Vehicles (UAVs), in this work we propose a new framework to track a master UAV in real-time, carrying a known constellation of LED markers, from a slave UAV without any other pose estimation capability. This setup is especially interesting to aerial manipulation and close-up inspection of structures with low or no texture. Our approach is able to fuse the estimated master's pose with the slave's onboard inertial readings, supporting intermittent communication between the UAVs. Evaluation on both simulation and real indoor and outdoor experiments reveals that the proposed approach achieves unprecedented robustness to noise and occlusion, accuracy and speed of computation. All the code to reproduce this work is publicly available.**

***Index Terms*—Aerial Systems: Perception and Autonomy; Localization; Visual-Based Navigation; Sensor Fusion**

Supplementary Material

The code, videos and simulations are available at:
www.v4rl.ethz.ch/research/datasets-code.html

I. Introduction

With the booming interest in Unmanned Aerial Vehicles (UAVs) for a variety of applications, such as infrastructure inspection and maintenance[1,2], there has been an increasing body of research dedicated in developing more complex aerial manipulation algorithms and collaborative approaches. With unique agility UAVs can swiftly reach high-rise places that previously were only accessible using special equipment, such as suspended platforms or bucket trucks. However, pose estimation comprises a key obstacle that still prevents the employment of UAVs in real scenarios.

With works most often relying on motion capture systems for indoor demonstrations, for outdoor functionality, research focuses mainly on computationally-expensive sensor fusion using Differential GPS (DGPS), cameras and inertial sensors to address pose estimation. However, these approaches do not apply to a wide range of tasks, such as, aerial navigation or manipulation, as UAVs have typically a small payload and consequently limited computational capabilities resulting to added challenges in robustness of operation. Often, such tasks require flying only a few centimetres away from large structures, such as buildings or wind turbines, and in these cases most of the existing pose estimation approaches fail as the sensors' behaviour deteriorates. Typically, GPS signal gets occluded by large structures and magnetometers are disturbed by the presence of metals. Moreover, the otherwise very promising infra-red depth sensors (e.g. Microsoft Kinect) perform poorly outdoors due to the presence of sunlight. and high-power laser sensors, such as the Velodyne Puck are typically too heavy for most aerial platforms. On the other hand, vision-based approaches perform well provided that the environment is sufficiently textured and inertial sensors are still reliable outdoors, rendering visual-inertial pose estimation as the main approach employed in recent systems besides the computational cost.

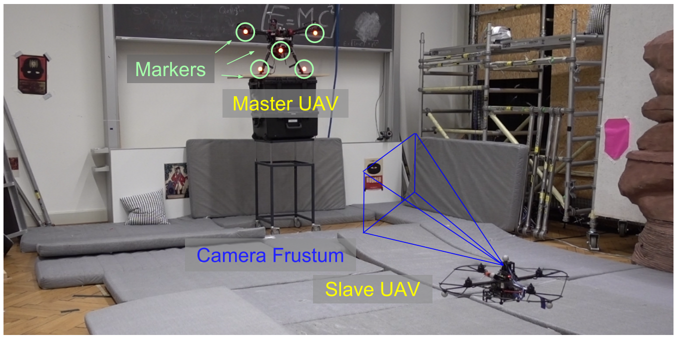

Fig. 1: Master/Slave setup used to collaborative pose estimation between a Master UAV carrying LED markers and a second Slave UAV carrying a camera pointing to the direction of the Master.

Aerial inspection of tree cavities was shown to be possible [1] employing visual-inertial Simultaneous Localization And Mapping (SLAM) to continuously estimate the pose of the UAV as it introduces its manipulator arm inside the cavity. Such tasks can only be accomplished thanks to the high-textured nature of the tree. For texture-less objects, pointing the camera to the ground can keep the estimation process working. However, as the triangulation of landmarks for depth estimation becomes ill-posed as the camera moves further away from the textured surface (e.g. ground), visual-inertial SLAM becomes very imprecise at large distances, rendering

Manuscript received: February, 24, 2018; Accepted April, 23, 2018.

This paper was recommended for publication by Editor Jonathan Roberts upon evaluation of the Associate Editor and Reviewers comments. This work was supported by the Swiss National Science Foundation (SNSF, Agreement no. PP00P2_157585) and EC's Horizon 2020 Programme under grant agreement n. 644128 (AEROWORKS)

Authors are with Vision for Robotics Lab, ETH Zurich, Switzerland lteixeira@mavt.ethz.ch; fmaffra@mavt.ethz.ch; mamoos@student.ethz.ch; chlim@ethz.ch

Digital Object Identifier (DOI): see top of this page.

[1] www.aeroworks2020.eu

[2] www.aeroarms-project.eu

pose estimation based on traditional methods very inaccurate. For reference, the pose accuracy of high-performing state-of-the-art visual inertial SLAM systems, has been evaluated when employed onboard a fully autonomous UAV by using a Leica Total Station for ground-truth; values are reported in [2] for the open sourced ROVIO [3] and in [4] for OKVIS [5]. Inspired by the need for a robust pose estimation for outdoor aerial manipulation at high-altitudes and at low computational cost, in this paper, we present a novel and efficient master/slave based method to perform collaborative pose estimation to enable flights close to structures including the ones with insufficient texture on the target. The proposed method makes use of an additional UAV in order to indirectly estimate the pose of the UAV flying close to the structure. The first UAV (Master) is positioned few meters away from the structure, such that its onboard visual-inertial SLAM can still achieve high accuracy, while a second UAV (Slave) is positioned close to the target surface. The latter, indirectly evaluates its own pose using the Master as reference. Using an onboard monocular camera and an inertial sensor, the Slave assesses its pose using a robust relative pose estimation approach based on small LEDs (i.e. markers) carried by the Master platform. Figure 1 illustrates the Master/Slave setup used in this work.

Our open-source system employs the open-source autonomous drone framework [6], rendering this work fully reproducible onboard a wide range of UAV platforms.

In summary, the main contributions of this paper are:

- a novel and efficient open-source relative pose estimation (VI-RPE) that cascades onboard visual-inertial sensors between multiple UAVs,
- a robust multi-hypothesis monocular pose estimation using visible LED markers that works in challenging outdoors scenarios, and
- an evaluation on simulated photo-realistic environments, indoor motion capture system and real outdoor scenarios.

II. Related Work

Marker tracking: Multi-camera motion capture systems (Mocap), such as VICON[3], make use of several calibrated cameras and identical reflective spherical markers illuminated by infra-red light in order to accurately estimate the markers position. The known location of the cameras allows to determine the position (in 3 degrees of freedom) of each marker by triangulation. A 6-degrees-of-freedom pose of a known constellation of spherical markers can also be estimated if the markers are attached to a rigid body. The small size and light weight of the markers make them easily attachable to UAVs. Moreover, the high accuracy of these tracking systems make this the most popular approach to accurate pose tracking in indoor confined spaces. April-tag-like [7] planar markers are also commonly used, however, mounting planar markers onboard a UAV, severally influence its dynamics and only very small tags can be placed on UAVs, as in [8], leading to inaccuracies and a small operational range. In practice, planar markers are only used on non-aerial vehicles or static objects, such as walls or on the ground.

Aerial relative pose estimation: A far more versatile approach to Mocap is to attach a known constellation of infra-red or visible-light LEDs markers on the UAV. Such markers are detectable under a wide variety of illumination conditions. Correlating the prior on the relative positioning of these markers on the UAV and the position of the marker detections in the image obtained from an external camera, the pose of the UAV can be calculated. This problem is typically approached in the literature either by using the Perspective-n-Point (PnP) formulation [9] as in [10] and [11] or a solver specifically designed for the known marker constellation as in [12]. The main problem of these approaches is the association between the detected markers and the ones in the known constellation. MPE [10], tries all combinations for initialization and then runs a tracking algorithm to avoid the association step. As reported by Wilson et al. [13], MPE tracking is not good enough to track one UAV from the other. In Section IV, we propose a novel multi-hypothesis tracker based on P3P and particle filtering that outperforms MPE and works well while tracking one UAV from another, widening the applicability of the method to more generic and outdoor scenarios, albeit posing greater robustness challenges to noise due to significantly more candidate detections per frame. Dias et al. [14] also proposed aerial relative pose estimation. Their system shares the same marker tracking algorithm as MPE, but they extend the methodology to support multiple robots using blinking LEDs for differentiation and add a high-level tracking algorithm that also takes in consideration inertial sensors and thrust commands. However, their results were only demonstrated in a similar setting to MPE's original work, of about 3 × 3m indoor environments.

Autonomous close-formation flight: The recent work in [13] proposes a leader-follower control and pose estimation using two autonomous unmanned planes where the leader features infra-red markers and the follower has a camera embedded. They propose a simpler and faster marker association algorithm that takes in consideration an initial pose from the planes, showing large scale outdoor results. Their controlling and tracking strategy leads to unprecedented results on close-range flight formation for planes, however, this is not applicable to multi-copters. The omni-directional often erratic motions from two multi-copter UAVs pose a much harder problem for tracking, while the fact that the controller is part of the tracking algorithm, renders this approach difficult to use with other generic controllers. Aiming to overcome this, the framework in this paper, does not couple the pose estimation tightly to any controller. The proposed system is developed using the open-source drone stack [6], with the controller outside the loop. The controller available in this stack can perform model-based prediction of the UAV's pose in the near future. To keep better relative pose between both UAVs the controller could be easily added to our pipeline, however, it would limit its generality of applicability.

III. VI-RPE: Visual-Inertial Relative Pose Estimation

Our method proposes a monocular-inertial relative pose estimation between a Master UAV with LED markers attached

[3] www.vicon.com

on it and a second UAV (Slave) carrying a camera pointing to the direction of the Master, as shown in Figure 1.

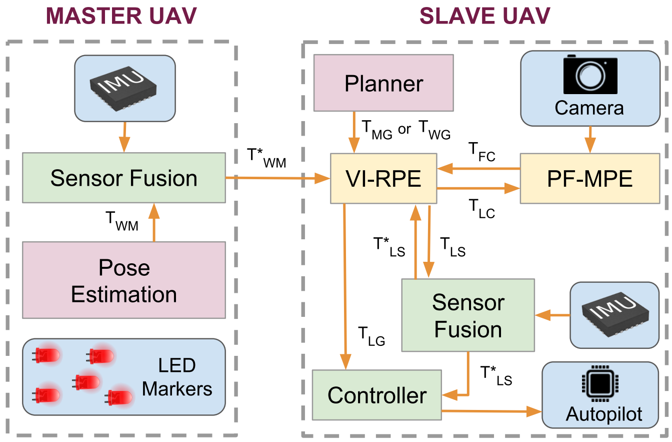

Fig. 2: This diagram shows the data flow of our system. We do not assume any synchronization between the components. VI-RPE is the responsible for the maintenance of a smooth odometry input for the controller while still pursuing the goal requested by the planner. The blue blocks are hardware. The green blocks are part of the open-drone stack [6]. We are using the non-linear MPC controller [15]. The pink blocks are also external implementations. In addition, the accuracy of different global position systems are reported in the results section. VI-RPE is described in this section and PF-MPE in Section IV.

A. Inertial and Relative Pose Fusion Problem

Relative pose estimation with a moving reference has an intrinsic incompatibility with inertial fusion, making this a very challenging problem. While inertial sensors inform acceleration in absolute values, the relative pose estimation reports the motion with respect to the moving target. Conventional approaches typically have both an odometry estimation with relation to a static reference and a relative pose estimation on the same robot. In this case, only the transformation between these two coordinate systems needs to be estimated. In contrast to traditional methods, the proposed approach tasks the Master UAV with global pose estimation and the Slave with computing only its relative pose to the Master. Great benefits are acquired when compared with the usual configuration, where the Master holds the camera and the Slave the markers. Firstly, and most important, in case of communication failures, the Slave UAV can still compute its relative pose. Secondly, multiple Slave UAVs can be added to the system without increasing the computational workload of the Master.

B. Coordinate frames and transformations

In our system, we define several coordinate frames and transformations. The rigid transformation T_{AB} transform from frame B to frame A and T^* represents the transformation after been fused with the inertial reading.

The coordinate frames used here are defined as:

- World [**W**]: Origin of the Master's pose estimation.
- Master Body [**M**]: Center of the Master UAV.
- LED Fiducial Markers [**F**]: Origin of the constellation.
- Slave Body [**S**]: Center of the Slave UAV.
- Slave Camera [**C**]: Centered on the Slave Camera
- Slave Goal [**G**]: Aligned with the Slave-Goal's Pose
- Local [**L**]: VI-RPE internal frame used to absorb inconsistencies between the relative pose estimation and the inertial measurements.

In addition, the transformation T_{CS} between the Slave and the camera and the transformation between Master and the fiducial marker, T_{MF}, are known from calibration.

C. Workflow

The main goal of our approach is to provide a smooth odometry and relative trajectory, which the Slave UAV can use for safe navigation at very close distances from the target structure and from each other even in the event of communication failures. In the traditional approach the pose of the Slave UAV at certain time can be estimated using equation 1. After fusing this pose with the IMU measurings, T^*_{WS} is fed to the controller. In this way, the goal can be defined by T_{WG} or T_{MG} using Equation 2. However, in case of a loop-closure detection or communication link failures for a short period of time, the pose of the Master with relation to the world frame, T_{WM}, can suddenly change, resulting to instabilities in the pose estimation.

$$T_{WS} = T^*_{WM} * T_{MF} * T_{FC} * T_{CS} \tag{1}$$

$$T_{WG} = T_{WM} * T_{MG} \tag{2}$$

With this problem in mind, our Visual-inertial Relative Pose Estimation (VI-RPE) introduces a novel artificial Slave Local coordinate frame L. The Local frame is static for each Slave UAV and the planning algorithm is not aware of this coordinate frame. Our software works as a mediator between all the other blocks of the system as show in Figure 2. In this configuration, the state of the VI-RPE algorithm consists of three dynamic transformations: the transformation between the Local frame and the World (T^{latest}_{LW}); the Local frame and the Master (T^{latest}_{LM}); and the Local frame and the Slave(T^{latest}_{LS}). In our case, T_{LS} is fed to the controller after being fused to the IMU and can be calculated using Equation 3 every time a new relative pose measurement is available. In addition, the Master pose in Local frame is updated using Equation 4 every time a new pose of the Master in the world is available.

$$T_{LS} = T^{latest}_{LM} * T_{MF} * T_{FC} * T_{CS} \tag{3}$$

$$T^{latest}_{LM} = T_{LW} * T_{WM} \tag{4}$$

As in the traditional approach, the user is only allowed to define the goal either in the World or the Master's frame. As a result, in our system, we need to keep an internal goal, T_{LG}, as the controller receives T_{LS}. These goals are converted using Equations 5 and 6.

$$T_{LG} = T_{LW} * T_{WG} \tag{5}$$

$$T_{LG} = T_{LM}^{latest} * T_{MG} \tag{6}$$

In case of a big change on T_{WM} or when a communication failure is restored, the World pose T_{LW} and the Master pose T_{LM} in the Local frame are changed at the same time in order to keep the pose of the Slave in the Local frame T_{LS} constant, while maintaining Equation 3 valid. During the initialization we set $T_{LW} = (T_{WM})^{-1}$.

IV. PF-MPE: Particle-Filter-Enhanced Monocular Pose Estimation

The core of the algorithm presented on this section was initially proposed by Breitenmoser et al. [16] and after improved and open-sourced by Faessler et al. [10] named as Monocular Pose Estimation (MPE). In this section we present a further improved version that was implemented based on Faessler et al. code. The VI-RPE is also compatible with Faessler's code or any other visual-based relative pose estimation algorithm as long as they are able to provide a 6D pose estimation. However, in the Section V, we show that our improvements in Faessler's algorithm is essential for real scenarios. While [10] offers a detailed explanation of the base algorithm, in this section we give focus to the proposed improvements.

A. Markers setup and detection

In contrast to [10], here we use RGB LEDs as markers, instead of infra-red ones. The configuration of the markers can be chosen arbitrarily, however, it has to be considered that three markers should not be collinear in 3D. We will assume five markers in this section and they are assigned to specific labels $\mathcal{M} : \{M_1, M_2, M_3, M_4, M_5\}$. The individual marker position on the UAV Master is known by the algorithm running in the UAV Slave.

In each camera frame the areas of interest correspond to the really bright image intensities. We use a threshold function followed by a Gaussian smoothing step before clustering neighbouring bright pixels in order to detect the markers in the image. The center of each cluster is computed using its first image moments. For a cluster be classified as a detection D_j, its shape and size should reflect the expected values with respect to the estimated distance d between the UAV (and hence each marker) and the camera. We check these by imposing constraints on the distortion of the circular nature of the marker and its expected size in the image. If the UAV is close to the camera, the markers are assumed to be bigger in the image, but as the distance between the UAV and the camera increases this distance slowly decreases.

B. Initialization of the Particle filter

When the UAV appears for the first time in the field of view of the camera or when the UAV tracking algorithm loses track, this (re-)initialization is performed. Our initialization consists of two steps. The first one follows the same initialization proposed in [10], where all possible associations between detections and a subset of 3 markers are tested. In constrast to [10] where the most probable hypothesis is selected and the initialization is finalized, our approach has a second step. Instead of pick only the most probable one, in this step we collect all the hypotheses passing in a cascade of filters. This set of hypotheses is used to create the initial particles. In such a way our particle cloud reflects directly the ambiguity of the UAV pose initialization problem at the beginning of tracking.

First Step: Since the marker labels are unknown following the detection stage, we want to consider as many UAV poses as possible. In order to get valid hypotheses we use the P3P algorithm [17], which suggests up to four possible poses by evaluating triplets of correspondences between detections and marker labels. The evaluation of all possible triplet of correspondences is the most expensive part of the algorithm given that the total number of poses suggested in each image during the initialization is

$$\mathcal{N}_{\mathcal{P}} = 4 \cdot \binom{n_D}{3} \cdot \frac{n_M!}{(n_M - 3)!} \ ,$$

where n_D and n_M stand for the number of detections and the number of total marker labels, respectively. In order to evaluate all the different correspondence possibilities, we maintain a voting matrix V with the rows spanning all detections D_j and the columns spanning all marker labels M_i. V is initialized as a zero matrix. The markers not considered for the P3P estimation are then reprojected onto the image plane. If at least one of them is close enough to a detection, i.e its distance is less than λ_r (here, 5 pixels), we increment the votes $\mathrm{V}(D_j, M_i)$ for all $D_j M_i$ correspondences participating in this hypothesis.

Second Step: Due to noise and the measurement uncertainties the obtained voting matrix is not only showing the optimal correspondences, but also other likely and some very unlikely possibilities. This is why taking in consideration multiples hypotheses is important.

So, here, we first weight each cell of matrix V according to the probabilities that a certain marker belongs to a certain detection and vice versa using Equation (7). If the obtained weight $w_{V_{ij}}$ is below a threshold λ_{vote} (here, $2/(n_D \cdot n_M)$) then it is set to zero.

$$w_{V_{ij}} = \frac{V_{ij}}{\sum_a V_{aj}} \cdot \frac{V_{ij}}{\sum_b V_{ib}} \tag{7}$$

The correspondences with a non-zero weight (i.e. $w_{V_{ij}} \neq 0$) are used to create possible combinations C, for finding the true pose. The set of possible combinations C is determined by all potential combinations, which use every marker label exactly once (i.e. $M_1 D_2, M_2 D_1, M_3 D_1, M_4 D_6, M_5 D_7$). Note that for the initialization stage all n_M markers are needed, but in the tracking stage also solutions containing fewer markers are accepted. After, for every combination in C the P3P problem is solved again. In contrast to the first step, a pose is valid if and only if the distance between the predicted reprojections of all marker labels, which were not used in the P3P, and their *corresponding* detection is smaller than the threshold λ_r. Every obtained valid pose is then stored for later use in the Particle Filter. In addition, the most likely combination is used to calculate the initial UAV pose and optimized according to Section IV-C.

C. Optimization of suggested UAV pose

As mentioned above the UAV pose is the result of an optimization. Starting with an initial guess coming from the initialization or the Particle Filter, the pose is optimized to minimize the error between the projected marker labels M_i and the corresponding detections D_j as expressed by Equation (8).

$$P_{opt} = \underset{P}{\operatorname{argmin}} \sum_{(M_i, D_j) \in C} \| \pi(M_i, P_{guess}) - D_j \| \tag{8}$$

The function $\pi : \mathbb{R}^3 \times SE(3) \rightarrow \mathbb{R}^2$ denotes the projection of the marker label M_i onto the image plane. For optimizing the pose it is parametrized according to the Lie Group theory and then the Gauss-Newton algorithm is applied.

D. Particle Filter for frame-to-frame UAV pose tracking

For the frame-to-frame UAV pose tracking we use a Particle Filter for maintaining multiple hypotheses. A particle contains an expected pose and its weight ($p : \{\mathrm{P}, w\} \in SE(3) \times \mathbb{R}_{[0,1]}$). If the Particle Filter is used for the first time after the initialization, each particle pose P is determined by a valid pose, obtained from the initialization stage. In case there are too many particles, some of them may be initialized at the same pose. On the other hand, if there are too many valid poses, only the most likely ones are used for the particle initialization. If the Particle Filter has already been running in the last frame, the particles are determined by the poses obtained by the last resampling. The set of particles is denoted by Ψ, ($p = \{\mathrm{P}, w\}$; $\Psi = \{p_k\} \; \forall \; k = 1, ...$). In order to keep generality, the particle update predicts the pose using a constant velocity model, which is based on the UAV poses from the last two frames (P_{i-1}, P_{i-2}).

The pose updates P^u of the individual particles p are obtained by applying the extrapolated change in pose T_i and some uniformly distributed noise $\mathcal{W}_i$ to the actual particle pose P (Equation (9)). The noise contains a translation part $t \in \mathbb{R}^3_{[-20mm, 20mm]}$ and a rotational part $R \in \mathbb{R}^{3 \times 3}$ which angles are bounded by ±30° for each axis,

$$P^u = T_i * P * \mathcal{W}_i \text{ , where } \mathcal{W}_i \in \left[\begin{array}{c|c} R & t \\ \hline 0 & 1 \end{array}\right]. \tag{9}$$

After updating the particles they are weighted. Each updated particle pose P^u is used to project the five marker labels onto the image. For each projection the distance Δ_{min} to the *closest* detection is calculated. This distance is needed to determine the number of projections (n_{PR}) which are close enough to a detection (i.e. $\Delta_{min} \leq \lambda_{r_{PF}}$, here, $\lambda_{r_{PF}} = 5pixel$). These two numbers ($n_{PR}$ and Δ_{min}), together with the number of self-occlusions ($n_{selfOcl}$), are used to calculate the weight w of the individual particles using Equation (10). The number of self-occlusions is defined as the number of projections which share a 'closest' detection. This value is included to suppress poses which are very far away (many or all markers projected on the same detection). Multiplying number of projections n_{PR} by number of markers n_M does ensure that combinations with more valid correspondences are preferred.

$$w = n_{PR} \cdot n_M + \sum_{k=1}^{n_{PR}} \Big(\frac{\lambda_{r_{PF}} - \Delta_{min,k}}{\lambda_{r_{PF}}} \Big)^2 - 3 \cdot n_{selfOcl} \tag{10}$$

If the weight of at least one particle is bigger than a threshold ($\lambda_{w_{PF}} \geq 5$) the particles will be resampled. The threshold $\lambda_{w_{PF}}$ changes according to the number of detected markers. The more markers are detected, the higher is the threshold. This adaptive change allows our method to track UAV poses with one or multiple markers (self-) occluded or poses when the UAV is not completely inside the field of view of the camera. In case the weights of all particles are smaller than $\lambda_{w_{PF}}$, the particle update and its weighting is repeated until at least one particle weight is bigger than the threshold or the maximum number of iterations is reached. In this case the particle update including the biggest weight is chosen. An overview of this step is shown by Algorithm 1.

Algorithm 1 Particle Update (for a single frame)

```
iter := 0, $w_{max} := 0$
while $w_{max} < \lambda_{w_{PF}}$ && iter < maxIter do
    $\Psi^u := \emptyset$
    ▷ update $p$ and build a set of updated particles $\Psi^u$
    for all $\{\mathrm{P}, w\} \in \Psi$ do
        $\mathrm{P}^u \leftarrow$ update(P, $T$)
        $\mu \leftarrow$ project($\mathcal{M}$, $\mathrm{P}^u$, $\mathcal{W}$)
        $w \leftarrow$ weight($\mu, \mathcal{D}$)
        $p^u = \{\mathrm{P}^u, w\}$
        $\Psi^u = \Psi^u \cup p^u$
    end for
    if $\max_w \Psi^u > w_{max}$ then
        $w_{max} = \max_w \Psi^u$
        $\Psi^u_{max} = \Psi^u$
    end if
    iter++
end while
$\Psi = \Psi^u_{max}$
```

After the updating stage the obtained weights are normalized and all particles are resampled. This step is done to reduce the number of unlikely particles and equalize their weights. The resampling is done using the stratified resampling method of [18]. Compared to the traditional multinomial approach, this method results to more uniformly distributed particles and smaller variance of the particles' distribution, which improves the quality of the resampling. Similarly to the initialization, the final pose for the UAV is obtained via optimization using Equation (8). Here the initial pose guess for the optimization is set to the pose with more particles after the resampling.

V. RESULTS

Our experiments were designed to assess the performance of our algorithm according to the estimation time needed, the robustness to challenging scenarios and the achieved accuracy.

The outdoor experiments demonstrate that our relative pose estimation works in a wide range of illumination conditions. We also use indoor experiments for comparisons with millimeter precision using a motion capture system, as well as a photo-realistic physical simulation for large-scale experiments.

A. *Accuracy of the Master's Pose Estimate*

The core of the contribution of this work is to handle the challenges arising by the use of a reference frame defined by the Master UAV that even when in hovering mode, can never be assumed to be entirely still. Using the UAV simulator Rotors [19] with metric photo-realistic models, shown in figure 3, we tested the scenario of aerial manipulation in front of a large building. First, we tested two state-of-the-art visual-inertial SLAM algorithms, ROVIO [3] and OKVIS [5] on a sequence where the UAV hovers at 5 meters away from a facade. Figure 4 shows that OKVIS exhibits smaller errors in the range of 10 cm, but with more high-frequency changes. ROVIO exhibits bigger errors on average, but error fluctuates smoothly, which is favourable if the Slave runs visual servoing similarly to [1].

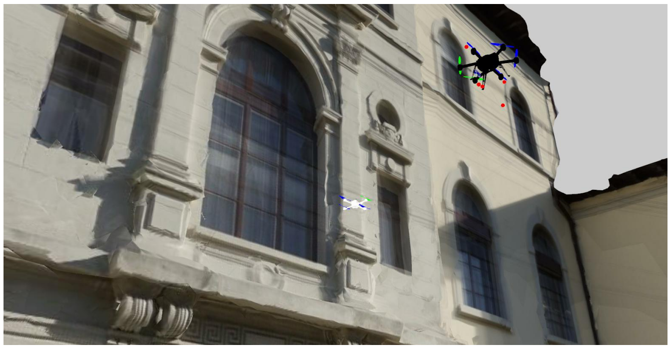

Fig. 3: UAV Master, in black with red marker, and the Slave, in white, in front of a large 3D Reconstructed facade

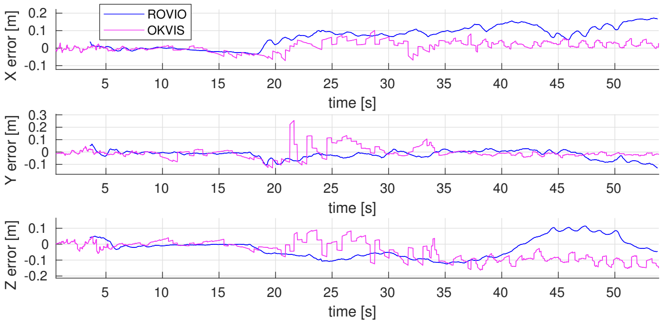

Fig. 4: The absolute error in the pose estimate of the Master UAV by two state of the art VI-SLAM systems: ROVIO and OKVIS.

Global position estimation is a challenging element in GPS denied scenarios. In this work we also tested global pose estimation in the recently published localization algorithm [20], especially designed to facades. The global pose error is on average 0.55m in x- and y-axis, and 0.22m in the z-axis and nearly zero error in yaw. With the disadvantage of requiring extra sensors in the ground and calibration, similar results can be obtained using Ultra Wide Band position estimation. Guo et al. [21] report errors of 0.175 m in a range of up to 100 m from the beacon.

In conclusion, combining the global localization and the VI-SLAM pose estimation, it is safe to assume that the Master will be hovering with a slow drift while the error with the visual localization algorithm can be bound to a maximum of about 50cm.

B. *Comparison of the propose PF-MPE to MPE for marker-based pose estimation*

The performance of the proposed PF-MPE method to localize the Master from the Slave is of crucial importance in order to enable accurate pose estimation of the Slave. Figure 5 shows a comparison between both Faessler et al. MPE [10] and the proposed approach in the same sequence. Highlighted in yellow are the areas where MPE loses track and besides a completely wrong angle estimation, the error in position still low. This happens because of the self similarity of the marker constellation. Sometimes the pose is flipped, so the error should be around 180 degrees but the position still correct. These poses are easily discarded by the visual inertial algorithm as it has access to the gravity orientation and it is known that it is not possible to see the marker from behind.

The PF-MPE does not improve or degrade the quality of the relative transformation. It only keeps the computation time very low to be able to be used onboard computers by avoiding the costly re-initialization described on IV-B. Figure 6 shows an outdoor experiment featuring several outliers all the time and Figure 7 shows the times of both algorithms. It is clear that it is not possible to use MPE algorithm as it takes over one second per frame, while we are required to run the algorithm at least 20 frames per second for good UAV stabilization. In the video accompanying this work we show examples of how our modification was essential for its outdoor deployment.

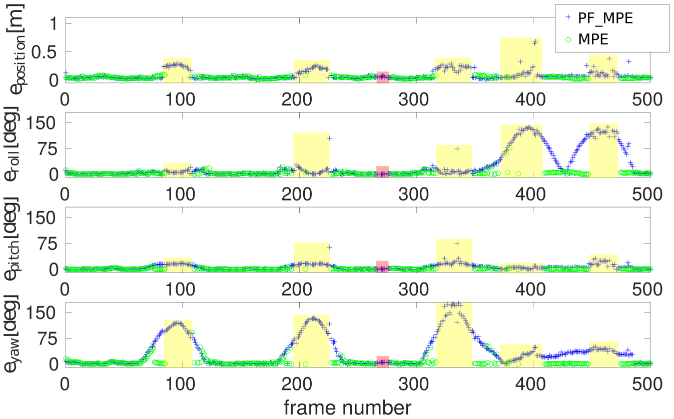

Fig. 5: The evolution of the position and orientation accuracy of MPE with respect to the proposed method on an indoor sequence. The shaded regions denote areas where MPE fails.

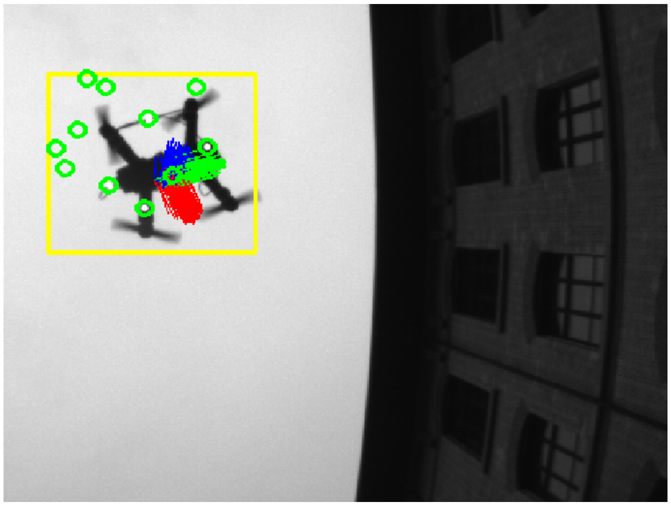

Fig. 6: An example image from an outdoor sequence denoting with green circles all candidate marker detections occurring inside the yellow region of interest. The proposed UAV tracker is able to successfully estimate the center of the UAV and its orientation, with the help of the particle cloud (RGB color-code denotes the orientation-axes suggested by each particle).

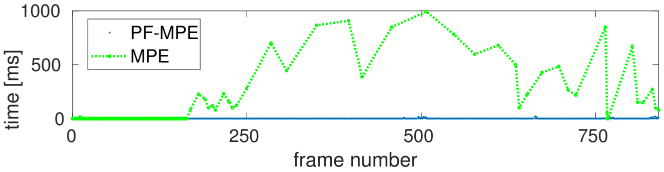

Fig. 7: The number of detections can have a significant influence on the computational time as evident in this sequence. Solving the P3P problem considering all possible detection triplets causes such a high computational time and that is why MPE is not able to estimate a pose for each frame of the sequence. The particle filter in the proposed method only evaluates all obtained detections against the projected marker labels suggested by each particle and can still estimate a UAV for each frame.

C. *Hovering using Motion Capture System*

Aiming to verify the error in the relative pose estimation while running the proposed VI-RPE algorithm onboard the UAV, in this experiment we use the Mocap system, VICON, as input to the controller. An Asctec Hummingbird equipped with an Intel Atom computer and a 1.2MP camera is used as the Slave UAV and a DJI S900 as the Master. Figure 8 shows the error of our VI-RPE algorithm w.r.t. the ground-truth while the Slave UAV flies as stable as possible.

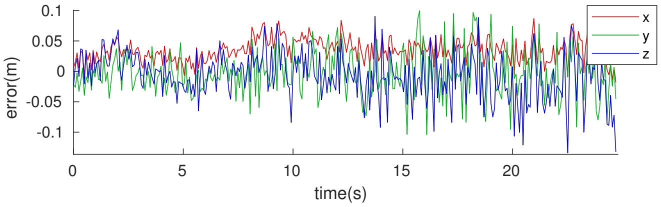

Fig. 8: The error of our VI-RPE algorithm w.r.t. the ground-truth while the Slave UAV flies as stable as possible.

D. *Autonomous Flight with Moving Slave and Static Master*

This experiment and all the next sections use our VI-RPE algorithm as input to the controller. The Mocap data is only recorded for analysis. Figure 9 shows the error of our algorithm w.r.t. the ground-truth.

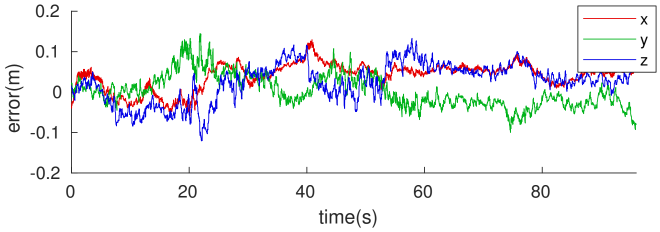

Fig. 9: The error of our VI-RPE algorithm w.r.t. the ground-truth during autonomous flight with moving Slave and static Master.

E. *Autonomous Flight of the Slave UAV with a Moving Master UAV*

On this set of experiments, we test our approach when the Master is willing moving in the direction on purpose. First we test the system using a goal pose in the Master frame (T_{MG}) and after in the Absolute World frame (T_{WG}).

1) Slave Hovering with static ***relative*** *pose :* On this experiment the Slave's goal is setted to (x:5,y:0,z:-1.3,yaw:180) in the Master frame (T_{MG}). Figure 10 shows the 3D position of the UAVs during the experiment, while the Master UAV moves back and forward four times. Figure 11 shows the difference between the goal and the actual position of the Slave UAV measured by the Mocap system in the x and z axis. So it does not represent the error of the pose estimation algorithm but the capacity of the whole solution in keeping the right position. Figure 12 shows the position of both UAVs in time. The ideal result for formation algorithms would be one graph on top of the other, but here we are interested in good platform stabilization. This figure shows that the Slave had a very stable flight using our VI-RPE algorithm while stayed behind only 2-3 seconds in average from the Master.

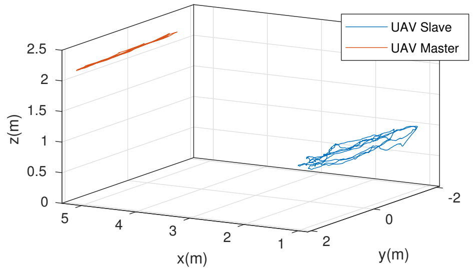

Fig. 10: The 3D position of the UAVs with the Slave hovering with static relative pose and a moving Master.

2) Slave Hovering with static ***absolute*** *pose :* This test is similar from the one before, but now the pose is setted to (x:0,y:0,z:1,yaw:180) in world frame (T_{WG}). This is the center of the room in the y direction. The Slave UAV is about 5 meters from the Master. Figure 13 shows that the UAV is capable to keep a position inside of a sphere of 20 cm radius.

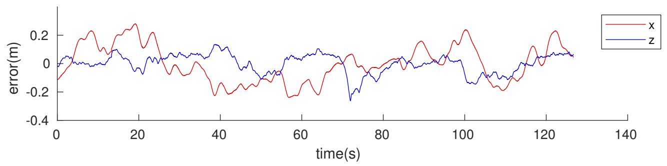

Fig. 11: Difference between the goal and the actual position of the Slave UAV measured by the Mocap system in the x and z axis with the Slave hovering with static relative pose and a moving Master.

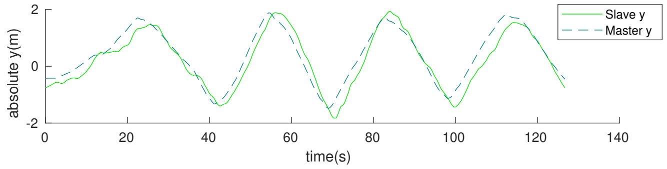

Fig. 12: Absolute position of both UAVs with the Slave hovering with static relative pose and a moving Master.

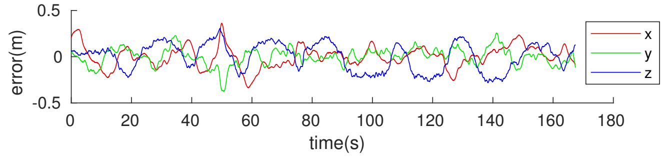

Fig. 13: Distance between the goal and current position measured by the Mocap system while the Slave is trying to keep the same position and the Master moving freely.

VI. Conclusions

In this paper we present a method to enable aerial manipulation and close inspection of potentially textureless structures. By indirectly evaluating the global pose of the inspecting/manipulating UAV, the proposed approach can provide accurate pose estimation in cases where traditional pose estimation approaches are not operational. Employing a Master UAV with a known constellation of LED markers and the ability to run visual-inertial SLAM, the Slave UAV operating in close range to the structure of interest, estimates its relative pose to the Master by tracking the LED markers. We evaluate the proposed approach with respect to scene ground truth obtained using a motion capture system with relative poses up to 5 m distance and in simulation up to 15 m with error ranging between 2 and 15 cm. On the experiments with autonomous flight, the Slave UAV was able to hover while the Master was moving with average error of 20 cm. The robustness of our particle-filter-enhanced monocular pose estimation was also tested in challenging outdoor scenarios.

Future work involves research in a fully communication-less approach between both aerial vehicles and also improvements using less generic information from the UAVs, such as the output of a specific controller.

Acknowledgment

The authors would like to thank Mina Kamel (Autonomous Systems Lab, ETH Zurich) for helping with the controller.

References

[1] K. Steich, M. Kamel, P. Beardsley, M. K. Obrist, R. Siegwart, and T. Lachat, "Tree cavity inspection using aerial robots," in *Conference on Intelligent Robots and System*, 2016.

[2] R. Bhnemann, D. Schindler, M. Kamel, R. Siegwart, and J. Nieto, "A decentralized multi-agent unmanned aerial system to search, pick up, and relocate objects," in *2017 IEEE International Symposium on Safety, Security and Rescue Robotics (SSRR)*, Oct 2017, pp. 123–128.

[3] M. Bloesch, S. Omari, M. Hutter, and R. Siegwart, "Robust visual inertial odometry using a direct ekf-based approach," in *Conference on Intelligent Robots and System*, 2015.

[4] L. Teixeira, I. Alzugaray, and M. Chli, "Autonomous aerial inspection using visual-inertial robust localization and mapping," in *Field and Service Robotics*, M. Hutter and R. Siegwart, Eds., 2018.

[5] S. Leutenegger, S. Lynen, M. Bosse, R. Siegwart, and P. Furgale, "Keyframe-based visual–inertial odometry using nonlinear optimization," *The International Journal of Robotics Research*, 2015.

[6] I. Sa, M. Kamel, M. Burri, M. Bloesch, R. Khanna, M. Popovic, J. Nieto, and R. Siegwart, "Build your own visual-inertial drone: A cost-effective and open-source autonomous drone," *IEEE Robotics Automation Magazine*, vol. PP, no. 99, pp. 1–1, 2017.

[7] E. Olson, "AprilTag: A robust and flexible visual fiducial system," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, May 2011, pp. 3400–3407.

[8] R. Hoogervorst, S. Stramigioli, H. W. Wopereis, and M. Fumagalli, "Vision-imu based collaborative control of a blind uav," in *2015 Workshop on Research, Education and Development of Unmanned Aerial Systems (RED-UAS)*, Nov 2015, pp. 53–61.

[9] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, 1981.

[10] M. Faessler, E. Mueggler, K. Schwabe, and D. Scaramuzza, "A monocular pose estimation system based on infrared leds," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2014.

[11] K. E. Wenzel, A. Masselli, and A. Zell, "Automatic take off, tracking and landing of a miniature UAV on a moving carrier vehicle," *Journal of Intelligent & Robotic Systems*, vol. 61, no. 1-4, pp. 221–238, 2011.

[12] H. Tjaden, F. Stein, E. Schömer, and U. Schwanecke, "High-speed and robust monocular tracking," in *International Conference on Computer Vision Theory and Applications*, 2015.

[13] D. B. Wilson, A. H. Gktoan, and S. Sukkarieh, "Vision-aided guidance and navigation for close formation flight," *Journal of Field Robotics*, 2016.

[14] D. Dias, R. Ventura, P. Lima, and A. Martinoli, in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*.

[15] M. Kamel, T. Stastny, K. Alexis, and R. Siegwart, "Model predictive control for trajectory tracking of unmanned aerial vehicles using robot operating system," in *Robot Operating System (ROS) The Complete Reference, Volume 2*, A. Koubaa, Ed. Springer.

[16] A. Breitenmoser, L. Kneip, and R. Siegwart, "A monocular vision-based system for 6d relative robot localization," in *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2011.

[17] L. Kneip, D. Scaramuzza, and R. Siegwart, "A novel parametrization of the perspective-three-point problem for a direct computation of absolute camera position and orientation," in *IEEE Computer Vision and Pattern Recognition (CVPR)*, 2011.

[18] J. D. Hol, T. B. Schon, and F. Gustafsson, "On resampling algorithms for particle filters," in *IEEE Nonlinear Statistical Signal Processing Workshop*, 2006.

[19] F. Furrer, M. Burri, M. Achtelik, and R. Siegwart, *Robot Operating System (ROS): The Complete Reference (Volume 1)*. Springer International Publishing, 2016, ch. RotorS—A Modular Gazebo MAV Simulator Framework.

[20] F. Maffra, L. Teixeira, Z. Chen, and M. Chli, "Loop-closure detection in urban scenes for autonomous robot navigation," in *International Conference on 3D Vision (3DV)*, 2017.

[21] K. Guo, Z. Qiu, C. Miao, A. H. Zaini, C.-L. Chen, W. Meng, and L. Xie, "Ultra-wideband-based localization for quadcopter navigation," *Unmanned Systems*, vol. 4, no. 01, pp. 23–34, 2016.