IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received April 2, 2021, accepted April 14, 2021, date of publication April 20, 2021, date of current version April 30, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3074537

Collaborative Localization for Micro Aerial Vehicles

SAI H. VEMPRALA[1], (Member, IEEE), AND SRIKANTH SARIPALLI[2], (Senior Member, IEEE)
[1]Microsoft Corporation, Redmond, WA 98052, USA
[2]Department of Mechanical Engineering, Texas A&M University, College Station, TX 77840, USA
Corresponding author: Sai H. Vemprala (sai.vemprala@microsoft.com)

ABSTRACT We present a framework for performing collaborative localization for groups of micro aerial vehicles (MAV) that use vision-based sensing. The vehicles are each assumed to be equipped with a monocular camera, and to be capable of communicating with each other. This collaborative localization approach is developed as a decentralized algorithm and built in a distributed fashion where individual and relative pose estimation techniques are combined for the group to localize against surrounding environments. The MAVs initially detect and match salient features between each other to create a sparse reconstruction of the observed environment, which acts as a global map. Once a map is available, each MAV individually performs feature detection and tracking with a robust outlier rejection process to estimate its own pose in 6 degrees of freedom. When needed, one or more MAVs can compute poses for another MAV through relative measurements, which is achieved by exploiting multiple view geometry concepts. These relative measurements are then fused with individual measurements in a consistent fashion to result in more accurate pose estimates. We present the results of the algorithm on image data from MAV flights both in simulation and real life, and discuss the advantages of collaborative localization in improving pose estimation accuracy.

INDEX TERMS Unmanned aerial vehicles, computer vision, localization, multi-robot systems.

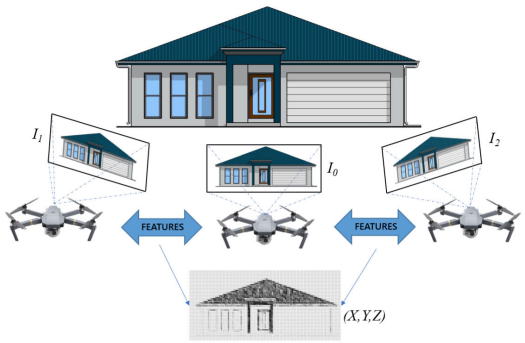
I. INTRODUCTION

Micro aerial vehicles (MAV) are a special class of unmanned aerial vehicles that have gained both general attention and research focus in recent years. The term MAV is generally applied to small multirotor configurations: typically measuring $0.1 - 0.5$m in dimension and between $0.1 - 0.5$ kg in mass [1], such as quadrotor platforms. MAVs possess several desirable properties: agile navigation in six degrees of freedom, a small size that allows them to fly within cluttered spaces, inexpensiveness and a relative ease of prototyping and so on. Currently, MAVs enjoy widespread popularity in many application domains: aerial photography, precision agriculture, search and rescue, delivery, inspection etc.

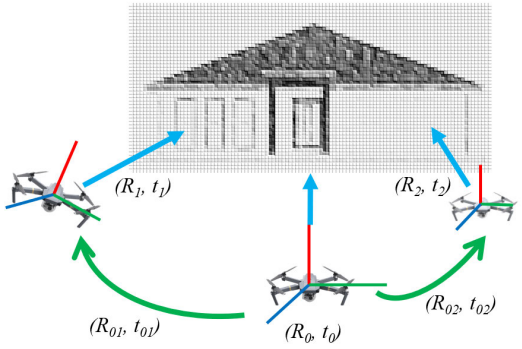
In the context of autonomous operations, MAVs require onboard sensing and computation for reliable localization and planning. The choice of sensors for MAV is typically constrained due to the requirement of small size: which results in a trade-off between fidelity of sensory information and size/power requirements. In this regard, vision sensors have shown great potential as exteroceptive sensors for MAVs. Specifically, monocular cameras can be seen as a particularly good fit for MAVs: as opposed to stereo cameras, which for instance, could possibly have baseline limitations and require more processing. Today, monocular cameras are almost ubiquitous on both hobby and research grade MAV platforms.

At the same time, the small size of MAVs usually creates constraints such as low computational power and smaller energy sources, which in turn limit their ability to perform complex tasks while maintaining sufficient flight time. Given these challenges, a single MAV in a complex autonomous operation can always run the risk of being resource limited with no backup in conditions that may lead to its failure. As a solution to this problem, it would be more desirable to employ multiple small, low-power MAVs as a team: an idea that can boost mission efficiency by allowing larger spatial coverage, larger distributed payloads etc. Multiple MAVs can also be leveraged for task distribution, helping reduce the computational burden on individual vehicles compared to single vehicle implementations. Collaboration can also help enhance localization accuracy as multiple sources of information can be fused for robust estimation. This is especially useful in the case of monocular vision sensing: while a single monocular camera cannot resolve the depth of a scene, depth

The associate editor coordinating the review of this manuscript and approving it for publication was Li He.



(a) Initial phase of collaboration between multiple MAVs involves matching feature points and creating a sparse reconstruction of the map.



(b) Subsequently, MAVs can localize either individually (intra-MAV), or through assistance from others (inter-MAV)

FIGURE 1. Vision based collaborative localization : a conceptual representation.

can be computed using information from cameras on other vehicles in a group.

In this paper, we present a framework for vision based collaborative localization (VCL) for a group of MAVs as an extension to our previous work [2], [3]. We assume that each MAV is equipped with a monocular camera, and is capable of communicating with the other MAVs to transmit or receive information. In the first step of the algorithm, the MAVs capture images of the environment visible through their cameras, upon which feature detection and matching are performed to isolate common salient features. These common features are then triangulated to form a sparse reconstruction that acts as a global map: which is then shared to all the vehicles (Figure 1(a)). Once the MAVs start moving, each MAV performs feature tracking to observe which features from the map are still visible, and uses these 2D-3D correspondences to perform its own individual pose estimation, which we call intra-MAV localization. When required, one or more MAVs can generate relative pose measurements to a target MAV and these estimates can be fused with the target's own individual estimate in a consistent way: and this process is known as inter-MAV localization (Figure 1(b)). As the MAVs continue to navigate, if the number of tracked features for the MAVs consistently falls below a threshold, the MAVs can match features between themselves to update the global map. Between the intra-MAV and inter-MAV modes of operation, the VCL algorithm operates in a mostly decentralized way. Because of the existence of these two modes of estimation, communication between members of the group is optional, there is no need for it to be continuous or synchronous. When communication is required, the algorithm has been designed in a way such that network bandwidth requirements are reduced. In the next few sections, we present the details of our algorithm along with results from realistic simulation imagery and data from real MAV flights. Although the system has been tested offline on pre-recorded image datasets, we also discuss for the applicability of the algorithm to real-time deployment.

In this article, we build upon and extend our previous work [2], [3] where we presented initial versions of our collaborative localization method leveraging multiple-view geometry, and data fusion through covariance intersection. We extend our previous work by enabling multi-vehicle fusion (more than one vehicle acting as measurement source), introducing guided matching and outlier rejection for more robust mapping and estimation, conducting an extensive ablation study discussing the advantages of fusion, along with validation of our full pipeline with data from real life. We direct the reader to the author's thesis [4] for an extended discussion on the topic.

II. RELATED WORK

Vision based localization has been studied extensively in the literature. Initially, it was achieved through external camera placement such as in professional motion capture systems [5] and [6]. When vision sensors were used as onboard exteroceptive sensors, RGBD sensors were one of the initially investigated setups. Microsoft Kinect sensors were used for altitude estimation [7], in tandem with a 2D laser rangefinder for mapping and localization [8] and visual odometry [9]. Even more recently, full six degree-of-freedom localization was demonstrated using RGBD sensors [10].

The ubiquity and compactness of monocular cameras have had a significant influence on their popularity and applicability for estimation in both computer vision and robotics communities. Many monocular camera based localization and mapping methods have been developed over the last decade such as Parallel Tracking and Mapping (PTAM) [11], Semi Direct Visual Odometry (SVO) [12], ORB-SLAM2 [13] and Large-scale direct SLAM (LSD-SLAM) [14], among which some were successfully implemented on UAV platforms. When applying these techniques onboard MAVs, a specific focus lies on removing the scale ambiguity. Various algorithms were proposed in the last few years that try to remove scale ambiguity either by fusing vision data with an IMU [15], using ultrasonic rangefinders in conjunction with optical flow such as in the commercial autopilot PIXHAWK [16]. Many other promising monocular visual-inertial systems have been proposed, such as MSCKF [17], visual-inertial ORB-SLAM2 [18], VINS-MONO [19] etc.

Photogrammetry using UAVs, which commonly involves feature detection and matching has also been used widely in areas such as topographic monitoring and precision agriculture [20], [21], although the focus here is often map building and image stitching while the UAVs localize using GPS.

Collaborative localization has been of interest recently as well, with the general idea being that of fusing different measurements to result in a more accurate fused state. Martinelli *et al.* [22] present a localization approach that uses an extended Kalman filter to fuse proprioceptive and exteroceptive measurements, applied to multi-robot localization. Nerurkar *et al.* [23] present a distributed cooperative localization algorithm through maximum aposteriori estimation, under the condition that continuous synchronous communication exists within a robot group. Carrillo-Arce *et al.* [24] present a decentralized cooperative localization approach where robots need to communicate only during the presence of relative measurements, which was tested in simulation and on Pioneer ground robots. We use this algorithm in our framework to facilitate inter-MAV data fusion. Indelman *et al.* [25] propose a multi robot localization algorithm that can handle unknown initial poses and solves the data association problem through expectation maximization. Knuth and Barooah [26] propose a distributed algorithm for GPS-denied scenarios, where the robots fuse each other's information and average the relative pose data in order to achieve cooperative estimation.

Subsequently, collaborative localization ideas were fused into the realm of aerial vehicles and vision based localization. Faigl *et al.* [27] proposed a method involving onboard cameras and observation of black and white markers for relative localization within a MAV swarm. Indelman *et al.* [28] propose a technique for cooperative localization for camera-equipped vehicles inspired by multi-view geometry ideas such as the trifocal tensor that estimates transformation between images. Zou and Tan [29] present a collaborative monocular SLAM system with a focus on handling dynamic environments, with multiple vehicles helping each other isolate moving features from constant ones, but requiring constant communication between cameras. In [30], the authors present an approach where two UAVs equipped with monocular cameras and IMUs estimate relative poses along with absolute scale, thus acting as a collaborative stereo camera. Piasco *et al.* [31] also present a distributed stereo system with multiple UAVs for collaborative localization with a focus on formation control. Forster *et al.* [32] show a structure-from-motion based collaborative SLAM system, which contains a centralized ground station whose function is to merge maps created by various vehicles. In this framework, the vehicles do not benefit from additional information from other vehicles. Similarly, Schmuck and Chli [33], [34] present a collaborative monocular SLAM pipeline for MAVs where each MAV runs the ORB-SLAM2 algorithm for SLAM and a central server focuses on place recognition, optimization and map fusion. Karrer *et al.* [35] present a similar technique for visual-inertial collaborative SLAM, combining advances in visual inertial odometry with a multi-agent scheme where the agents and a central server communicate both ways to result in globally consistent maps.

In our paper, we propose a vision-only based collaborative approach focused on localization of a group of MAVs. The main contribution of this work lies in the extension of vision based 6-DoF pose estimation for MAVs to a cooperative estimation scheme by combining individual and relative estimation. In addition, our relative localization approach does not require direct visual recognition or explicit range measurements between vehicles, and instead, this information is inferred through feature overlap. We list out the main features of our proposed approach, some of which exhibit differences compared to other existing works in this domain.

1) Our approach presents a systematic pipeline for collaborative localization in groups of MAVs, with vehicles directly communicating relative measurements and fusing relative and individual pose estimates. MAVs can receive corrections from multiple members in the group to enhance their own pose accuracy.
2) We show that collaborative estimation is possible through a vision-only approach with no requirement of additional sensors for collaboration. Both individual and relative pose estimation are performed using only visual (feature) data, making it suitable for the occasional issues with visual-inertial fusion due to IMU drift in cluttered spaces.
3) We present a consistent method for individual-relative pose estimate fusion through covariance intersection. Leveraging epipolar error and camera reprojection error, we estimate the uncertainties of vision based pose estimation and subsequently present a method to fuse these estimates even under unknown correlations.
4) The proposed algorithm is aimed to be decentralized, as pose estimation is attempted individually by all MAVs. Communication between vehicles is only required when relative measurements are requested, either as part of pose correction or map construction. This helps reduce network/bandwidth requirements during constrained operation.
5) Compared to distributed SLAM approaches, our method is aimed at applications where overlap between vehicles is guaranteed such as large-scale structure from motion. Our approach emphasizes direct vehicle-vehicle cooperation such as relative pose estimates as opposed to tasks such as map-merging.

III. PROBLEM STATEMENT

The main goal of the proposed framework is to estimate six degree-of-freedom poses of multiple micro aerial vehicles (MAV). Each micro aerial vehicle is a multirotor platform capable of moving in all three translational axes with roll, pitch and yaw capabilities, thus navigating in $\mathbb{R}^3 \times SO(3)$, and is equipped with an intrinsically calibrated monocular camera. In the world frame of reference, as the vehicles are usually small in size, we consider the position of the camera

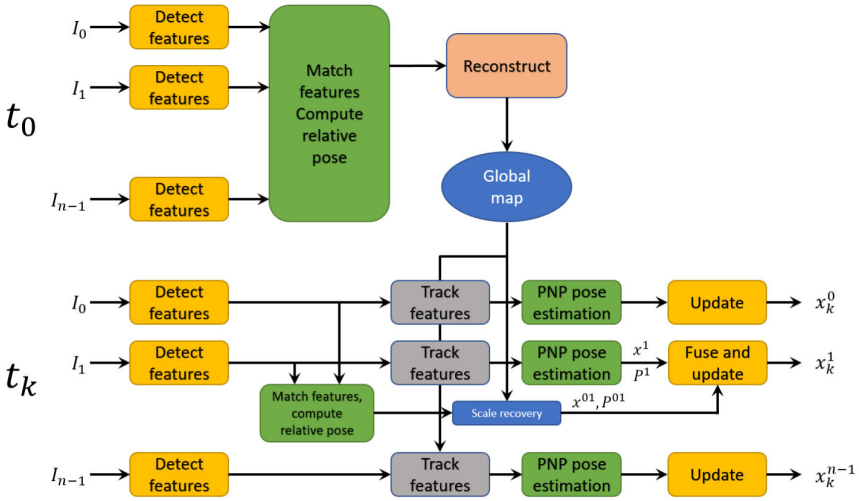


FIGURE 2. Flow of the vision based collaborative localization (VCL) algorithm. At any time step k, all MAVs in a group capture images $I_0 \ldots I_{n-1}$. With the VCL algorithm, an MAV j computes intra-MAV state and covariance estimates x^j and P^j respectively, while also being able to fuse it with inter-MAV estimates x^{ij}, P^{ij} from its neighbor j to produce a final pose estimate.

to be equivalent to the position of the vehicle. Thus, for k MAVs in a group, the goal is to estimate a system state $\mathbf{X}$ (equation 1), comprising of individual states: where each state is the 3D position of the respective MAV's onboard camera (x, y, z) and the roll, pitch, yaw angles (ϕ, θ, ψ), all in a predefined frame of reference (equation 2). The reference frame can be chosen according to the application: it can represent a specific location in the world frame, or all the vehicles in a group can be made to localize relative to a leader MAV. which can be chosen according to the task - for exmaple, one with the highest computational capacity can act as a leader. As the problem is formulated as localization as opposed to full SLAM, the system state does not account for map points or any uncertainties associated with them.

$$\mathbf{X} = \begin{bmatrix} \mathbf{X_0} & \mathbf{X_1} & \ldots & \mathbf{X_n} \end{bmatrix} \tag{1}$$

$$\mathbf{X_m} = \begin{bmatrix} x_m & y_m & z_m & \phi_m & \theta_m & \psi_m \end{bmatrix} \tag{2}$$

The cameras are assumed to adhere to the central projection model; each of which has a known intrinsics matrix $\mathbf{K}$ and is capable of producing 2D projections for each 3D point it observes through a projective mapping $\pi : \mathbb{R}^3 \implies \mathbb{R}^2$. At every time step, we assume the availability of images from each camera from which salient feature points are observed as 2D projections. Given this information, the responsibility of the algorithm is to estimate the 6-DoF poses of all MAVs to form the state matrix specified above.

To facilitate pose estimation, we make three important assumptions:

1) All the cameras onboard the vehicles are calibrated, and the intrinsics (and distortion coefficients, if any) are known.
2) The true distance between at least two of the vehicles in the group is known prior to commencement of flight.
3) Communication delays between vehicles are ignored in the current scope of the work. Any information transmitted between vehicles is assumed to be received without delay or loss.
4) During localization, we assume that the real world locations of the landmarks that constitute the map stay constant.

IV. VISION BASED COLLABORATIVE LOCALIZATION

In this section, we detail the individual steps of the vision-based collaborative localization (VCL) algorithm. For reference, the general flow of the algorithm is depicted in figure 2. At every time step, the first phase of the algorithm involves detecting salient features in the images and matching them for correspondences. Initially, these correspondences are used to build a sparse 3D map, and subsequently, new correspondences from the vehicles are a) matched with the map to result in pose estimates for each vehicle, and b) matched with each other to result in inter-MAV estimates. Inter and intra-MAV estimates are fused for increased accuracy.

A. FEATURE DETECTION AND MATCHING

The collaborative localization framework works on the basis of feature data: hence, the first step in the algorithm is to detect and describe salient features visible in the field of view of each camera. In our pipeline, we have utilized two kinds of detection/description methods which were empirically chosen:

1) CPU implementation: AKAZE - Accelerated KAZE features and M-LDB descriptors.
2) GPU implementation: KORAL - Multi-scale FAST features and LATCH descriptors.

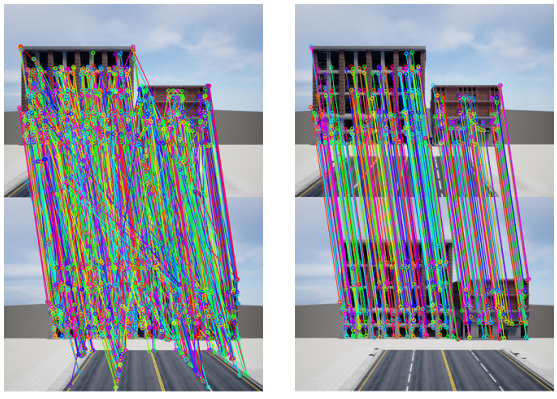
AKAZE features [36], an extension of KAZE [37] are multi-scale features which are faster to compute compared to SIFT [38] and demonstrate better accuracy than ORB [39]. In the second combination, we use a system known as KORAL [40], which contains a modified version of the FAST

corner detection algorithm [41] that is robustified by applying it to every layer in a non-linear scale space pyramid. The features detected in this step are described through 512-bit binary vectors from the LATCH description algorithm [42]. As the next step to both these techniques, we utilize brute force matching on a Hamming distance metric in order to find matches between different sets of these binary descriptors. The VCL framework is method-agnostic and can be adapted to any feature extraction or matching algorithm.

B. RELATIVE POSE ESTIMATION

The detection/matching step of the algorithm results in feature data: a combination of keypoint location coordinates in the image plane, and a binary descriptor data vectors corresponding to each location; and a list of common features between two or more views. The common feature data is then used to estimate relative poses between the vehicles as well as to create a sparse reconstruction of the environment.

We use epipolar geometry principles to estimate the essential matrix relating both views (specifically, the matched feature points) using the 5-point algorithm [43]. In order to avoid false matches during estimation that naturally arise due to high speed, repetitive texture etc, we use a modified RANSAC scheme known as the a-contrario RANSAC (AC-RANSAC) [44], [45]. Unlike traditional RANSAC, AC-RANSAC does not require predetermining a value of the error threshold. Figure 3 demonstrates how AC-RANSAC helps during essential matrix estimation by filtering out matches that do not adhere to the right epipolar geometry.



(a) Raw feature matches (b) AC-RANSAC filtered matches

FIGURE 3. Demonstration of how AC-RANSAC helps with filtering out outliers in feature matches.

C. MAP BUILDING

Once a relative pose is known between two camera views, this information can be coupled with the feature matches to compute a sparse 3D reconstruction of the matched features. We achieve this using the DLT triangulation method [46] over all the feature matches after computing a relative pose as described in IV(B). This map can then be made available to all the MAVs and is meant to be reused for feature tracking, 3D-2D correspondence computing as well as a source for performing scale factor recovery for subsequent reconstructions. We note here that for the first ever reconstruction, having access to the distance between two MAVs helps remove the scale ambiguity problem (assumption 2). The initial map, thus, is assumed to be metrically accurate: and this scale can be propagated through future updates. The globally available map data consists of two parts: one, a set of the 3D locations of all the points in the map, and two, the feature descriptor for each point from the keyframes. For a group containing more than two vehicles, we use an incremental reconstruction procedure. The vehicle pair with the maximum feature overlap is considered a seed pair and a first reconstruction is attempted. Feature observations of the other MAVs are then incrementally included in this reconstruction. Finally, we use a fast bundle adjustment scheme to jointly optimize the poses and the scene landmarks [47].

D. INTRA-MAV LOCALIZATION

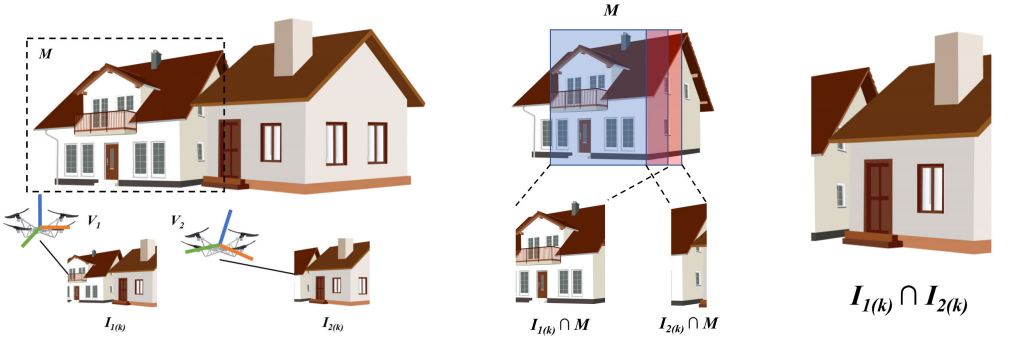
Intra-MAV localization is the process that is performed by each MAV independently once a global map is distributed to all agents. Every time the MAV's onboard camera captures an image, the intra-MAV algorithm performs feature detection on the image and attempts to track points from the 3D map that are still visible. Once these 2D-3D matches are isolated, a six degree-of-freedom pose for the camera (and by extension, the vehicle) can be computed using the perspective-N-Point (PNP) algorithm. The PNP algorithm estimates the relationship between a set of 3D points (tracked from a map) and their projections on the image plane for a calibrated camera. In our implementation, we combine an efficient perspective-3-point algorithm [48] with another AC-RANSAC scheme, and apply it to the tracked correspondences between the image and the 3D map, which results in estimates of the position and orientation of the MAV. Once a pose estimate is computed, it is further refined by minimizing the reprojection error as defined below:

$$\theta^* = \arg\min_{\theta} \sum_{i} \|\mathbf{x_i} - \pi(\mathbf{X_i}, \theta)\| \tag{3}$$

π encodes the camera projective transformation of a 3D point $\mathbf{X}_i$ onto the image plane for the pre-computed pose θ, whereas $\mathbf{x}_i$ is the actual observation from the image at that time step.

E. INTER-MAV LOCALIZATION

The success and the accuracy of intra-MAV localization depends entirely on the amount of overlap between the features in view of a certain vehicle and the features that comprise the global map. While theoretically 5-8 common points are sufficient to estimate the essential/fundamental matrices between multiple views [46], in practice this requirement tends be higher due to noisy matches or low quality features. If an insufficient number of features are tracked by a vehicle, the error in pose estimation would increase drastically, which could then lead to drift. To assist with this, we capitalize on



(a) Example case with two vehicles V_1 and V_2 with varying overlap with map, thereby varying localization accuracy.

(b) V_2 does not have sufficient overlap with map, whereas V_1 does

(c) V_2 has significant overlap with V_1, hence a 'local', relative pose estimate is possible.

FIGURE 4. Example scenario where intra-MAV localization may fail and inter-MAV localization can be beneficial.

the existence of more than one vehicle, and attempt to use the collaborative nature to the advantage of localization. We call this step the 'inter-MAV' localization step.

Figure 4 shows a possible scenario where inter-MAV localization can be helpful. Let us consider a case with two MAVs: V_1 and V_2. At time instant k, vehicle V_1 has a good amount of overlap with the map $\mathbf{M}$ (blue region in Figure 4(b)), which gives it higher confidence in its own pose; whereas vehicle V_2 does not (red region in Figure 4(b)). On the other hand, there is sufficient overlap between V_1 and V_2 independent of the map; with a non-zero overlap with the $\mathbf{M}$ (Figure 4(c)). Hence, the high confidence of V_1 in its own pose can be used to the advantage of V_2: we create a pipeline that enables V_1 to estimate the metric pose of V_2, which can then be fused with V_2's own onboard estimate. We discuss the individual steps of this process below.

1) RELATIVE POSE ESTIMATION

As described in the IV(A) and IV(B), first, the features visible from V_1 and V_2 are isolated and matched, and the essential matrix is used to compute a relative rotation and translation. This measurement of translation will only result in a unit vector with an arbitrary scale factor, thus is not sufficient for an accurate metric pose. This unit translation also results in a reconstruction of the features common to V_1 and V_2: a reconstruction that is scaled with an arbitrary, yet unknown scale factor.

2) SCALE FACTOR ESTIMATION

The local map obtained through the common points between the images of V_1 and V_2 can be referred to as $\mathbf{M}'$. If an assumption is made that both V_1 and V_2 were able to localize using intra-MAV localization, albeit with varying degrees of accuracy, both V_1 and V_2 have a non-zero overlap with the existing map. Hence, it follows that there is a non-zero overlap of features between $\mathbf{M}'$ and $\mathbf{M}$.

In order to compute the right scale factor λ for this reconstruction (one that matches the true scale of the global map), the frame of reference for the local map has to be considered. Within this local frame, let us assume V_1 to be the 'host' MAV, located at the origin $[\mathbf{I}|\mathbf{0}]$ and V_2 as the 'client' MAV at $[\mathbf{R}|\mathbf{t}]$, where $\mathbf{R}$ and $\mathbf{t}$ are the estimated relative rotation and translation between the host and client. If any two pairs of common features can be identified between the local and the global map, as their true 3D coordinates are already known from the global map, the ratio of the length of a line connecting the local coordinates to the length of one connecting the global coordinates is the true scale factor for the new map. Once this scale factor is known from the ratios, the relative pose is scaled to its right value, and the reprojection error is minimized to obtain a better estimate.

While comparing the inter-MAV local map and the global map, any wrong matches between the two sets of points can affect the estimation of the scale factor greatly. Although the goal of the AC-RANSAC scheme used during relative/individual pose estimation is typically to solve this very problem of removing outliers, in case of matching two maps during inter-MAV localization, the total number of points could be too low for a RANSAC scheme to act effectively. Hence, the VCL algorithm uses guided matching to ensure accuracy of matching between the two point sets.

3) GUIDED MATCHING

The goal of this step is to find accurate matches between two maps: one being the typically denser global map, and the other, a sparse temporary map obtained by the matches between V_1 and V_2. Now, it can be recalled that the host MAV, which is responsible for generating relative poses has an acceptable degree of confidence in its own pose, which means that both the global map and local map are generated from confident poses. If the descriptors of the features that form the global and local maps can be assumed to represent two (virtual) images, the transformation between these two views is already known. Given this known transformation, it is possible to 'guide' the matching towards inliers that adhere to the transformation [46]. As the rotation and translation

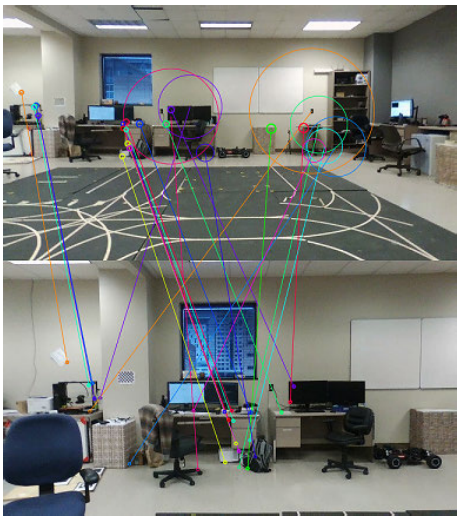


FIGURE 5. Guided matching to remove outliers: false matches are seen to have a high epipolar error, corresponding circles seen to have a much larger radius than those of the inlier matches.

between the two maps $\mathbf{R}'$ and $\mathbf{t}'$ are known, it is trivial to compute the fundamental matrix $\mathbf{F}$ relating the two views, and then discard any matches (x_1, x_2) that are not adherent to the proper epipolar geometry $\mathbf{x}_2^\top \mathbf{F}\mathbf{x}_1 = 0$.

Intra-MAV estimation usually suffers in accuracy when there are not enough features to be tracked from the original map. In such cases, inter-MAV estimation can be helpful as it utilizes common features between the MAVs at that instant and does not require multiple observations over time. Scale recovery in the inter-MAV estimation step requires a minimum of only two pairs of accurate matches between the local and global map, as opposed to intra-MAV estimation, which requires a significantly higher number of tracked features for better accuracy.

F. UNCERTAINTY ESTIMATION

One of the critical parts of localization is to estimate not only the position and orientation of a vehicle, but also estimate the uncertainty of the estimated pose, usually described through a covariance matrix. Such state estimates and covariances are conventionally propagated through an optimal filtering framework to predicting and updating poses. To enable this process, it is important to describe the accuracy of each measurement received.

In both the inter and intra-MAV estimation steps, a final refinement step is performed through a non-linear least squares method where the algorithm attempts to correct the pose by minimizing the reprojection error. For a projective transformation π, the cost function that is being optimized, i.e, sum of all squared reprojection errors can be written for m features, each with 2D pixel coordinates x and 3D coordinates X as

$$\begin{aligned} f(\theta) &= \Sigma_i \, \|r_i(\theta)\|^2 \\ \text{where } r_i(\theta) &= x_i - \pi(X_i, \theta) \end{aligned} \tag{4}$$

The Hessian of this function can be evaluated to be:

$$\nabla f(\theta) = \sum_i r_i(\theta) \nabla r_i(\theta) = \mathbf{J}(\theta)^T \mathbf{r}(\theta)$$

$$\nabla^2 f(\theta) = \sum_i \nabla r_i(\theta) \nabla r_i(\theta)^T + \sum_i \nabla^2 \, r_i(\theta) \tag{5}$$

$$\approx \mathbf{J}(\theta)^T \mathbf{J}(\theta) \tag{6}$$

When a solution is close to a local minimum, the effect of the second order terms in eq. 5 is minimized, and hence they can be ignored. It can then be seen that the outer product of this Jacobian matrix of the reprojection error at the final optimum with itself is an approximation of the Hessian matrix of the solution (eq. 6). For a non-linear multidimensional least squares error near the optimum, the inverse of this Hessian matrix is an approximation of the covariance matrix of the reprojection errors [49]. Hence, the approximate covariance of the solution can be expressed as

$$\Sigma = (\mathbf{J}^\top \mathbf{J})^{-1} \tag{7}$$

We note here that Σ in eq. 7 does not necessarily translate into an uncertainty in the real position/orientation values directly: it merely expresses the quality of the solution and the possible uncertainty around the local surface at the point of convergence. This value is still a function of the reprojection errors and not of the rotation/translation parameters: in case the solution is a local minimum, the estimated covariance could still be low although the pose estimate is not very true to the actual value. So in order to express the pose uncertainty more accurately, we scale this covariance artificially with the reprojection error obtained for that pose estimate.

$$\mathbf{R} = (\mathbf{J}^\top \mathbf{J})^{-1} * \epsilon_r \tag{8}$$

G. KALMAN FILTER AND OUTLIER REJECTION

As a final step, the VCL algorithm propagates raw measurements through a Kalman filter framework for state and covariance estimation. The VCL scheme being a purely vision based localization system without augmentation from other sensors like IMUs, the cameras are essentially considered to be replacements for the vehicles in terms of poses. Due to this reason, the primary function of the Kalman filter in the VCL framework is for smoothing and outlier rejection, utilizing a constant velocity model as the process for simplicity. This filtering scheme can be modified to incorporate a more complex vehicle model, or include IMU measurements as an extension to this work.

Each MAV is responsible for running its own internal recursive estimation scheme through a Kalman filter which maintains a running estimate of the mean and covariance of its state. At every instant an image is received, it is expected that the MAV computes an intra-MAV pose estimate for itself. Once computed, the obtained measurement is used to correct the state and covariance of that particular MAV according to

conventional Kalman Filter predict-correct equations.

$$\begin{aligned}\mathbf{P}^i_{k|k-1} &= \mathbf{A}^i_k \mathbf{P}^i_{k-1|k-1} \mathbf{A}^{i^\top}_k + \mathbf{Q}^i_k \\ \mathbf{S}^i_k &= \mathbf{H}^i_k \mathbf{P}^i_{k|k-1} \mathbf{H}^{i^\top}_k + \mathbf{R}^i_k \\ \mathbf{P}^i_{k|k} &= (\mathbf{I} - \mathbf{KH})\mathbf{P}_{k|k-1}\end{aligned} \tag{9}$$

The measurement noise covariance $\mathbf{R}$ corresponding to each measurement is computed as discussed in eq. 8. Matrices $\mathbf{A}$, $\mathbf{P}$, $\mathbf{H}$, and $\mathbf{Q}$ denote the state transition model, the initial state covariance, observation model and the process noise covariance are chosen as identity matrices for simplicity, each scaled by a certain factor.

Before using the obtained pose value in the measurement update, it is beneficial to determine the likelihood of the measurement being an outlier. We achieve this by propagating new measurements through a Chi-squared gating test to detect and reject noisy observations. For each new measurement, the algorithm computes the Mahalanobis distance between the predicted state and the measurement as

$$\gamma_k = (\mathbf{z}^i_k - \hat{\mathbf{x}}^i_k)^\top \mathbf{S}^{-1} (\mathbf{z}^i_k - \hat{\mathbf{x}}^i_k) \tag{10}$$

If γ_k exceeds the α-quantile of the Chi-squared distribution for six degrees of freedom, the measurement can be treated as an outlier and discarded ([50], [51]).

In the context of inter-MAV estimation, the responsibility of computing both the state and the covariance of a client MAV is taken up by the host MAV. Once a relative measurement between the host and client, denoted as $\mathbf{z}^{i,j}_k$ is available, host MAV V_i uses this relative measurement in conjunction with its own state estimate to compute the state of MAV V_j as follows.

$$\mathbf{x}^{j'}_k = \mathbf{x}^i_k + \mathbf{M}^{i,j}_k \mathbf{z}^{i,j}_k \tag{11}$$

In the VCL scheme, the measurement computed by the host is the pose of the client directly, i.e., there is no additional observation model - hence M in eq. 11 is also an identity matrix. We note that when V_i attempts to compute the uncertainty of V_j, this is in combination with V_i's own uncertainty. Hence, the covariance matrix that V_i has estimated for itself should be propagated into any other relative measurements attempted by V_i. When V_i computes a relative measurement to V_j at time instant k with measurement noise covariance $\mathbf{R}^{ij}_k$, the corresponding state covariance for V_j: $\mathbf{P}^{j'}_k$ can be calculated as:

$$\mathbf{P}^{j'}_k = \mathbf{H}^{j'}_k \mathbf{P}^i_{k|k} \mathbf{H}^{j'^\top}_k + \mathbf{R}^{ij}_k \tag{12}$$

H. DATA FUSION

One of the core features of the VCL routine is the ability to fuse estimates from multiple sources. Let us consider two vehicles V_i and V_j, and a case where the pose computed by V_i for V_j needs to be fused with the onboard estimate of MAV V_j itself (an example scenario is depicted in Figure 6). A conventional way of data fusion is to include both sources as two measurements in a Kalman filter update phase - but here, the relative measurement and the intra-MAV measurement have common sources of information, i.e., the map data and the features. The two MAVs could also have communicated pose data in the past, which makes these estimates correlated. But because these cross-correlation parameters are not kept track of, the correlations are treated as unknown: which makes the conventional Kalman filter update step result in inconsistent and erroneous estimates.

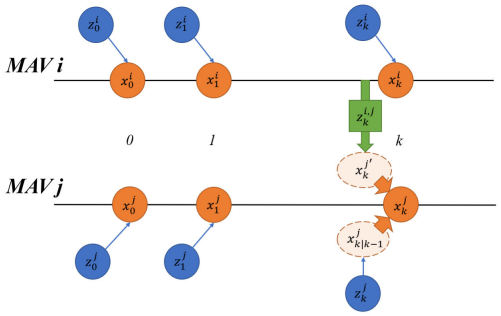


FIGURE 6. A pictorial representation of data fusion between inter-MAV and intra-MAV localization. At time step k, MAV i attempts to correct the pose of MAV j by generating a relative measurement, which is then fused by MAV j with its own onboard estimate.

Covariance intersection (CI) is an elegant solution for the fusion of estimates with unknown correlations [24], [52] . The CI algorithm expresses the covariance of the fused estimate as a combination of the individually estimated covariances. Depending on the confidence in each estimate or a desired final statistic, each individual covariance can be weighted by a scalar value. In the case of the VCL framework, each estimate is already described by its own confidence coming either from onboard the same vehicle requiring fusion, or the host vehicle that generated a pose for the client. At time instant k, assume that the individual estimate of V_j is a state-covariance pair with state $\hat{\mathbf{x}}^j_k$ and covariance $\hat{\mathbf{P}}^j_k$. For the same time instant, V_i computes another state-covariance pair for the pose of V_j, represented as $\hat{\mathbf{x}}^{ij}_k$ and $\hat{\mathbf{P}}^{ij}_k$. Then the CI algorithm can be used to compute a state and covariance pair of a fused estimate as below.

$$\mathbf{P}^j_k = \left[\omega(\mathbf{P}^j_k)^{-1} + (1-\omega)(\mathbf{P}^{ij}_k)^{-1}\right]^{-1} \tag{13}$$

$$\mathbf{x}^j_k = \mathbf{P}^j_k \left[\omega(\mathbf{P}^j_k)^{-1}\mathbf{x}^j_k + (1-\omega)(\mathbf{P}^{ij}_k)^{-1}\mathbf{x}^{ij}_k\right] \tag{14}$$

where ω is a parameter that is computed such that the trace of the combination of the covariances being fused is minimized.

$$\arg\min_{\omega} Tr\left[\omega(\mathbf{P}^j_k)^{-1} + (1-\omega)(\mathbf{P}^{ij}_k)^{-1}\right]^{-1} \tag{15}$$

Another elegant property of the CI algorithm is that, while being derived from a geometric viewpoint, it can also be expressed as a matrix- and scalar-weighted optimization problem. This property allows the CI algorithm to be extended to the fusion of higher dimensional state vectors and thus, an arbitrary number of estimates. Consequently, in the VCL framework, fusion can be performed between more than two sources of data, where for k sources, the covariance matrices are weighted by an array of k weighting factors $\omega_1, \omega_2, \ldots, \omega_k$ such that

$$\omega_1 + \omega_2 + \ldots + \omega_k = 1 \tag{16}$$

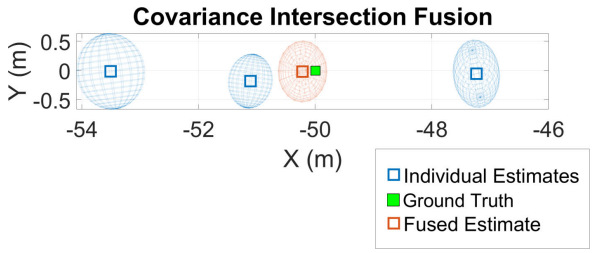


FIGURE 7. Example of covariance intersection fusion of data from three sources.

For k sources of data, eq. 15 can be extended into a multi-variable minimization problem of finding a set of weights that minimize the weighted sum of the traces of all covariance matrices involved. Figure 7 is a visual depiction of covariance intersection fusion for data from 3 sources, from an experiment where 3 vehicles were responsible for estimating the position of a fourth on the X axis. At that particular time step, none of the estimates are particularly close to the ground truth, but the level of confidence exhibited by the closest estimate is higher compared to the others; which results in the covariance intersection algorithm computing the right combination of weights to result in a fairly accurate fused estimate.

I. MAP UPDATES

While the formulation and focus of the VCL algorithm is mainly on localization, mapping is an essential part of the process. In certain situations that could be part of the application, all the MAVs may have to move away from an initial map, which would necessitate an update of the global map in order to maintain localization. While inter-MAV localization is able to assist with the case of specific MAVs leaving the area of the mapped scene, all the vehicles navigating to a different area would require a new map. Hence, the same principles that allow for inter-MAV localization, i.e, relative pose estimation and scale recovery, can be used for building a new global map. This process can be invoked when the tracked feature count falls under a certain threshold for all the vehicles in the group or any other condition that is deemed appropriate. The steps involved in a map update follow the steps in the inter-MAV localization closely:

1) Perform feature matching and relative pose estimation to result in a new, scale-ambiguous reconstruction.
2) Perform matching between the new reconstruction and the existing global map in order to scale the new reconstruction accurately.
3) Perform fast bundle adjustment to jointly optimize poses and map points and then, either replace the global map with the new one, or append new points to existing map.

V. IMPLEMENTATION

The collaborative localization framework has been written in C++, where we utilize various open-source libraries available in OpenCV [53] and OpenMVG [54] in order to implement feature detection, matching, AC-RANSAC and PNP pose estimation. Ceres libraries [55] were utilized to refine reconstructions and estimated poses, as well as for estimating covariances of the solutions, and dlib [56] was used to perform optimization for the covariance intersection.[1] The algorithm was run on an Intel NUC computer with an Intel i7-6770HQ processor, 32 GB of RAM and an NVIDIA GTX 1080 GPU used as an external GPU.

The collaborative localization algorithm has been tested on image data obtained from both simulated and real flight tests. For the simulations, we use Microsoft AirSim as our platform [57]. AirSim is a photorealistic UAV simulator built as a plugin for Unreal Engine, with capabilities such as high resolution textures, realistic lighting, soft shadows etc. As our technique is heavily dependent on computer vision,

Algorithm 1 VCL Algorithm: Sample for Two MAVs

procedure BUILDMap(x_i, I_1, I_2)
  $i_1, i_2 \leftarrow$ detectFeatures(I_1, I_2)
  $\bar{i_1}, \bar{i_2} \leftarrow$ matchFeatures(i_1, i_2)
  $\mathbf{E} \leftarrow$ ACRANSAC($\bar{i_1}, \bar{i_2}, \mathbf{K_1}, \mathbf{K_2}$)
  $\mathbf{R}, \mathbf{t} \leftarrow$ SVD($\mathbf{E}$)
  $\mathbf{M} \leftarrow$ reconstruct($\bar{i_1}, \bar{i_2}, [I|0], [\mathbf{R}, \mathbf{t}]$)
  return $\mathbf{M}$ ▷ $\mathbf{M}$:= Global map
procedure localizeIntraMAV(I_k, $\mathbf{K_k}$, $\mathbf{M}$)
  $i_k \leftarrow$ detectFeatures(I_k)
  $\bar{i_1} \leftarrow$ trackFeatures($i_k, \mathbf{M}$)
  $\mathbf{R}, \mathbf{t} \leftarrow$ PNP($\bar{p_k}, \mathbf{M}, \mathbf{K_1}$)
  $\mathbf{z_k}, \mathbf{R_k} \leftarrow$ refinePose($\mathbf{R}, \mathbf{t}, \mathbf{M}$)
  $\mathbf{x}_k, \mathbf{P}_k \leftarrow$ updateState($\mathbf{z}_k, \mathbf{R}_k$)
  return $\mathbf{x}_k^j, \mathbf{P}_k^j$
procedure localizeInterMAV($\mathbf{x}_i$, I_i, I_j)
  $i_i, i_j \leftarrow$ detectFeatures(I_i, I_j)
  $\bar{i_i}, \bar{i_j} \leftarrow$ matchFeatures(i_i, i_j)
  $\mathbf{E} \leftarrow$ ACRANSAC($\bar{i_1}, \bar{i_2}, \mathbf{K_i}, \mathbf{K_j}$)
  $\mathbf{R}, \mathbf{t} \leftarrow$ SVD($\mathbf{E}$)
  $\mathbf{M}' \leftarrow$ reconstruct($\bar{i_i}, \bar{i_j}, [I|0], [\mathbf{R}, \mathbf{t}]$) ▷ Local map
  $\mathbf{m}_{map} \leftarrow$ matchFeatures($\mathbf{M}', \mathbf{M}$)
  $\lambda \leftarrow$ recoverScale($\mathbf{M}', \mathbf{M}, \mathbf{m}_{map}$)
  $\mathbf{t} \leftarrow \mathbf{t} * \lambda$
  $\mathbf{z}_k^{i,j} = [\mathbf{R}, \mathbf{x_i} + \mathbf{t}]$
  $\mathbf{z}_k^{i,j}, \mathbf{R}_k^{i,j} \leftarrow$ refinePose($\mathbf{R}, \mathbf{t}, \mathbf{M}'$)
  $\mathbf{x}_k^{j'}, \mathbf{P}_k^{j'} \leftarrow eqn(18), (19)$
  return $\mathbf{x}_k^{j'}, \mathbf{P}_k^{j'}$
procedure FUSEINTERINTRA($(\mathbf{x}_k^j, \mathbf{P}_k^j), (\mathbf{x}_k^{j'}, \mathbf{P}_k^{j'})$)
  $\mathbf{P_A} \leftarrow \mathbf{P}_k^j$
  $\mathbf{P_B} \leftarrow \mathbf{P}_k^{j'}$
  $\omega \leftarrow \arg\min_\omega Tr(\omega \mathbf{P_A}^{-1} + (1-\omega)\mathbf{P_B}^{-1})$
  $\mathbf{P}_k^{j*} \leftarrow (\omega \mathbf{P_A}^{-1} + (1-\omega)\mathbf{P_B}^{-1})^{-1}$
  $\mathbf{x}_k^{j*} \leftarrow \mathbf{P}_k^{j*}(\omega \mathbf{P_A}^{-1}\mathbf{x}_k^j + (1-\omega)\mathbf{P_B}^{-1}\mathbf{x}_k^{j'})$
  return $\mathbf{x}_k^{j*}, \mathbf{P}_k^{j*}$

[1] The source code can be found at https://github.com/saihv/coloc

(a) Simulated city environment in Unreal Engine/AirSim

(b) Parrot Bebop 2 quadrotors used in real flight experiments

FIGURE 8. Implementation details for simulation and real experiments.

using AirSim enabled creating high fidelity, close-to-real-life situations. Each MAV simulated within AirSim had a forward facing monocular camera, and onboard images were captured at approximately 5 Hz with a resolution of 640×480. We created an urban environment in Unreal Engine within which the MAVs were flown through different trajectories (example picture from the environment can be seen in Figure 8(a)). The images from the onboard cameras and ground truth poses were recorded. For the real life tests, we used two Parrot Bebop 2 quadrotors (Figure 8(b)). Videos from the forward facing monocular cameras were recorded onboard the vehicles at a 1280×720 resolution, and then processed offline. Due to limitations of the Bebop 2 platform, it was not possible to capture both GPS/IMU updates along with high-frequency images in a timestamped way, hence, we limit our quantitative error analysis to simulation. The assumption regarding initial distance between the drones was satisfied by starting the flights from pre-marked positions. A quick summary of our implementation and results can be found in a supplementary video.[2]

VI. RESULTS AND DISCUSSION

A. INTRA-MAV LOCALIZATION: SIMULATION

As a first step, we initialize three MAVs in the simulation environment, which were then commanded to take off and fly in square-like trajectories at different altitudes, while onboard camera images were recorded. With these images from the three vehicles, the algorithm builds a global map and performs only intra-MAV localization for each vehicle. Figure 9 shows the three estimated trajectories of the vehicles along with the ground truth. Each MAV covers approximately 120m in total distance. Table 1 shows the RMS errors of the VCL estimates for the three vehicles compared to the respective ground truth positions.

B. INTRA-MAV LOCALIZATION: REAL EXPERIMENTS

Similarly, we also evaluate the performance of intra-MAV localization in real scenarios with the Bebop 2 quadrotors. Figure 10 show the localization results for trajectories navigated by two MAVs. The first test (10(a)) was meant to evaluate the accuracy of the algorithm against ground truth, so the

[2]Supplementary video can be found at https://www.youtube.com/watch?v=LvaTOWuTOPo

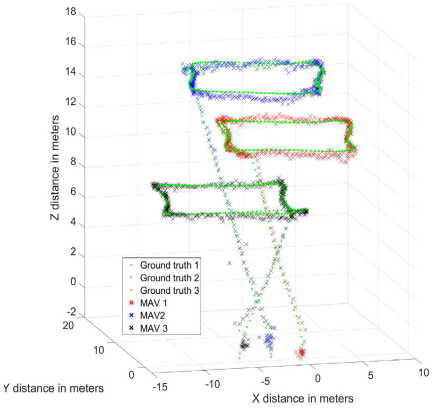
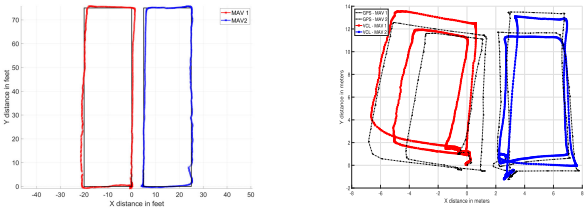


FIGURE 9. VCL position estimates for three MAVs navigating within AirSim - intra-MAV localization only.



(a) X-Y positions of two Bebop 2 quadrotors moved through a trajectory of known, marked dimensions. Ground truth plotted in black.

(b) X-Y positions of two Bebop 2 quadrotors flown through rectangular trajectories. GPS position estimates plotted in dotted black.

FIGURE 10. Intra-MAV localization in real experiments.

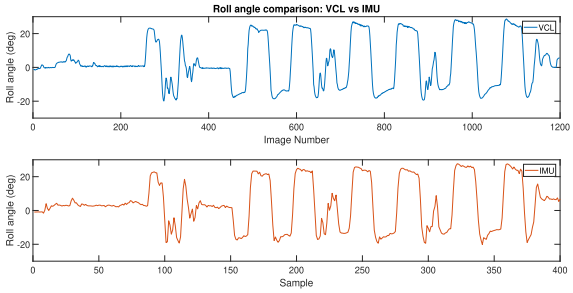
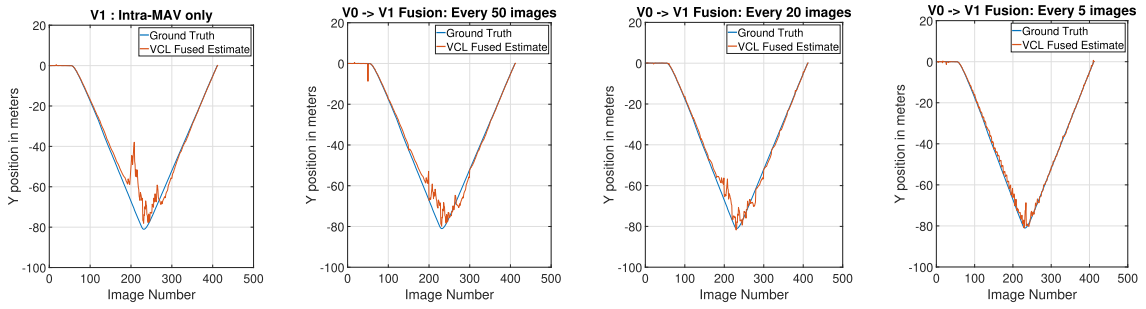


FIGURE 11. Roll angle comparison between VCL estimates and onboard IMU (ground truth) for a fast side-to-side flight shows good tracking.

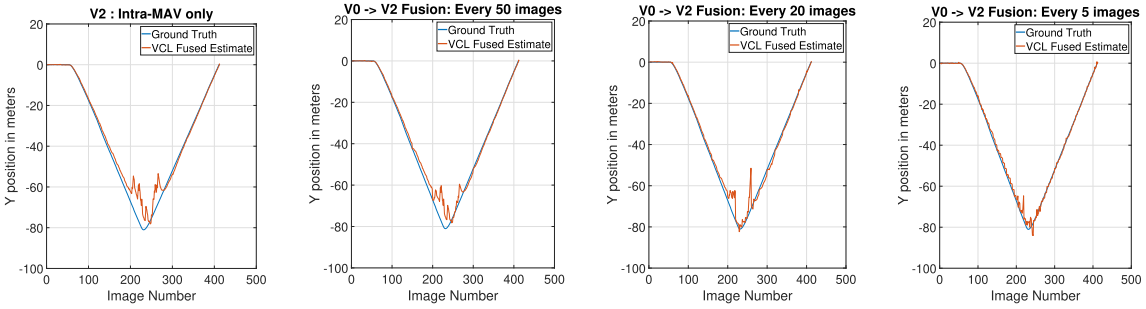
TABLE 1. RMS/maximum absolute errors for position estimates of three MAVs in AirSim.

MAV ID	Error	X (cm)	Y (cm)	Z (cm)
1	RMSE	2.63	4.75	4.94
	Max	66.45	52.45	98.23
2	RMSE	2.77	4.88	4.70
	Max	62.41	65.43	71.28
3	RMSE	2.39	3.85	4.49
	Max	44.99	65.29	84.11

MAVs were moved by hand along two pre-marked rectangles of dimensions 75×20 feet each. The VCL estimates were seen to track the ground truth fairly well, while exhibiting a slight drift at the far edges (which can be attributed to changes in feature distribution/appearances compared to the keyframes with which maps were generated).



(a) V1 with no fusion (b) V1 with fusion every 50 images (c) V1 with fusion every 20 images (d) V1 with fusion every 5 images



(e) V2 with no fusion (f) V1 with fusion every 50 images (g) V2 with fusion every 20 images (h) V2 with fusion every 5 images

FIGURE 12. Effect of frequency of inter-MAV data fusion on localization for both clients: Only Y axis positions shown for simplicity. Accuracy of estimation increases with higher frequency of fusion, whereas unfused estimates (leftmost) exhibit large errors.

The second test of the intra-MAV localization involved manual flight of two MAVs in an outdoor area, and comparison with the GPS position estimates. In this test, we observe that the VCL estimates were closer to the real trajectories taken by the MAVs than the GPS, which could be attributed to low altitudes, demonstrating the requirement for vision based navigation. Comparison of GPS estimates, VCL estimates and ground truth are shown in 10(b).

Finally, we also evaluate the accuracy of orientation estimates in real flights. In one particular test, the two Bebop MAVs were made to fly side to side at high speeds (up to 5 m/s), and the estimates of the roll angles were compared to the estimates coming from the IMU data onboard the vehicle. The VCL estimates and the IMU estimates of the angles are close to each other (Figure 11), demonstrating an accurate estimation of angles even through fast flights. The roll angles coming from the IMU and the VCL are plotted separately instead of a common X axis due to the varying frequencies of the data sources.

C. INTER-MAV LOCALIZATION: SIMULATION

1) EFFECT OF FREQUENCY OF RELATIVE MEASUREMENTS

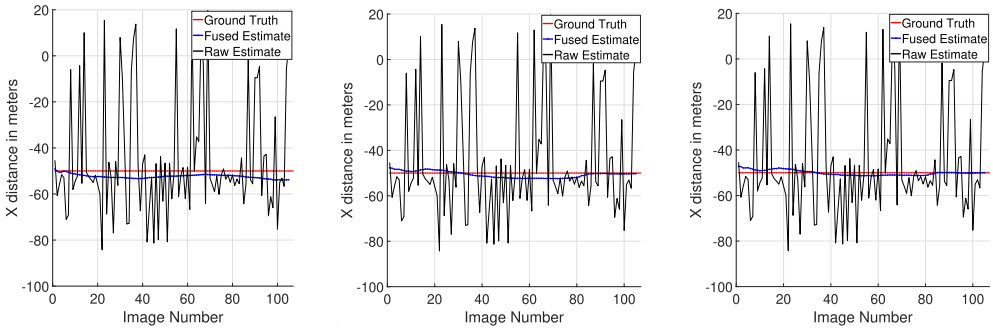
Inter-MAV localization was first tested and evaluated in the simulation, by creating a sparsely populated environment with three vehicles - flying in backward-forward formation. Three images from this trajectory are shown in Figure 13 to demonstrate the change in image appearances. Between the backward and forward motions, each MAV traverses a total of 160 meters in the environment.

(a) Initial position (b) Midpoint (c) Final position

FIGURE 13. Sample images from backward and forward trajectory.

We recall here that the biggest advantage of inter-MAV localization is when one vehicle has better localization than the others, so to simulate such a condition, we assume the MAV in the center has access to better estimates. We simulate this condition by allowing this MAV to use ground truth positions corrupted with a small amount of zero mean noise as its pose estimate, whereas the other MAVs use the normal ('intra') vision-based localization routine at each timestep. We then attempt to analyze the effect of inter-MAV localization between this host MAV and the others.

With no inter-MAV measurements, the VCL estimates for the clients show significant error at the midpoint, where the vehicles are the farthest from the scene. But once inter-MAV measurements are obtained and fused, the errors decrease, and increasing the number of times inter-MAV measurements are fused also decreases the error significantly (Figures 12 and 15). In this specific case, increasing the frequency of the relative measurements to once in 5 images brings the fused position estimates of V_1 and V_2 closer to the ground truth, exhibiting a RMS error of 2.3m in position.



(a) One host (b) Two hosts (c) Three hosts

FIGURE 14. Client X axis position estimate with varying number of sources for fusion. Having more number of hosts results in more accurate estimation (fused estimate closer to ground truth in (c) compared to (a)), whereas the unfused raw estimate is very erratic.

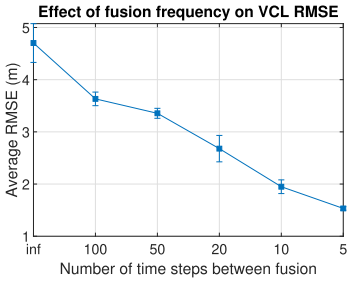


FIGURE 15. Effect of frequency of inter-MAV data fusion on RMS error. Error bar shows variation of position RMSE between clients V1 and V2.

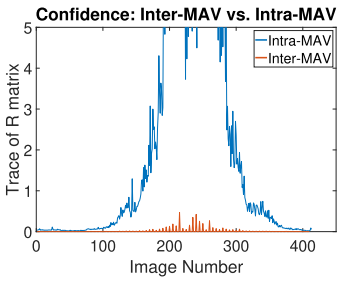


FIGURE 16. Comparison of inter vs intra-MAV measurement covariances. Inter-MAV measurements are usually seen to have significantly lower solution covariance.

(a) V_0 (b) V_1 (c) V_2 (d) V_3

FIGURE 17. Images from the starting positions for four MAVs in AirSim.

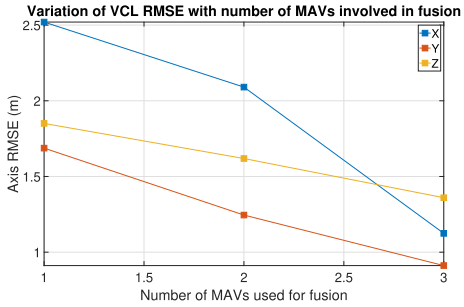


FIGURE 18. Change in position RMSE of one MAV with the number of participants providing relative measurements.

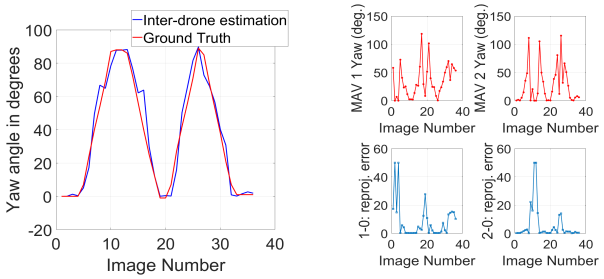
Over 160m of navigation, this equates to about 1.4% of RMS error.

Figure 16 shows a comparison of the measurement confidences of intra-MAV measurements versus inter-MAV measurements by plotting the trace of the measurement covariance **R**. It is evident here that as the MAV moves away from the map features, the feature appearance deviates from the keyframes and the intra-MAV measurement covariance rises rapidly; but the inter-MAV covariance stays relatively low throughout, as it only depends on the amount of feature overlap and distribution of points at every particular instant.

2) EFFECT OF NUMBER OF VEHICLES IN GROUP

In a second experiment involving inter-MAV localization and fusion, we attempt to evaluate the effect of having more than one host vehicle that is able to assist with inter-MAV localization for a client vehicle. For this experiment, we initialize four MAVs in another simulation environment, which are spaced apart by a distance of 25m between each pair of vehicles, greatly reducing feature overlap. We treat V_3 as the client that needs localization assistance, because it can be seen from Figure 17 that V_3 has the least overlap with the scene, thus has the most potential to exhibit inaccuracy in the intra-MAV localization scheme. To maintain this condition of low overlap, the vehicles were then made to execute trajectories along the Z axis, while keeping the X axis positions mostly constant.

We observe that all three hosts combining their inter-MAV estimates results in the most accurate estimate for the client V_3, as multiple sources of information help robustify the final estimate in the covariance intersection scheme. In Figure 14, we show the X-axis position estimates of V_3 compared to the ground truth, with the fused estimate in blue and the raw estimate in dotted black. Although V_3's internal estimate is very noisy, fusing data from one other 'host', as seen in 14(a) allows V_3 to track the ground truth with an RMSE of 2.52 m. The error reduces even further with an increase in the number of vehicles taking part in the fusion, as evidenced by the



(a) Fused yaw angle estimate for a rotating MAV.

(b) Yaw angle from relative estimates from the other two vehicles, and associated reprojection errors.

FIGURE 19. Estimation of yaw angle of a client MAV through inter-MAV localization when other MAVs are able to contribute.

estimates in figures 14(b) and 14(c), with three hosts bringing the error down to 1.12 m. Figure 18 plots the variation of the RMSE on all three position axes as the number of vehicles participating in data fusion changes.

3) HANDLING PURE ROTATION

One of the problems that is evident in single monocular-camera localization is the issue with purely rotational movement in yaw, which is a very common maneuver for MAVs. Pure rotation usually causes a large portion of existing map points to go out of view suddenly, while the fact that there is no translation by the camera means it is not possible to triangulate new feature points through a single camera without any additional information. In contrast, the VCL algorithm is able to handle this problem by capitalizing upon the inter-MAV localization feature points between what is one rotating MAV and another MAV that has map points in common.

As a test case, we simulate an environment containing two perpendicular buildings being observed by an MAV (MAV 0) that performs periodic 90-degree rotations, trying to observe both. MAV 0's rotation speed was set to 45°/s. Two other MAVs (1 and 2) are also present in the proximity in hover mode, each observing one of the buildings from a distance. To assist with MAV 0's pose estimation during fast rotations, for every image captured by these three MAVs, we compute relative poses between vehicles 1-0 and 2-0 and attempt fusion with the estimate of MAV 0. Due to the way the fusion algorithm is framed, the final fused yaw angle estimate is computed based on which estimate has the least uncertainty. In figure 19(a), we show the fused estimates of the yaw angle, where we see that the estimated angle closely matches the ground truth. A careful analysis of figure 19(b) shows how either MAV 1 or MAV 2 is chosen as the better source of the yaw information based on the reprojection error (on which the uncertainty estimate depends). Swift rotations such as this are typically problematic for single-camera localization, but the collaborative aspect can maintain localization through data fusion from other, possibly more reliable, sources of information.

D. INTER-MAV LOCALIZATION: REAL EXPERIMENTS

After validating the efficacy of inter-MAV localization and data fusion in simulation, we test the VCL algorithm with fusion on data from real flights. As a first experiment, we fly two Bebop MAVs manually indoors. In this experiment, MAV 1 hovers in the middle of a room, whereas the other (MAV 2) navigates a square trajectory around the first. This trajectory was traversed in a way that MAV 2 encounters one part of the room where the feature overlap with the initial map is relatively low. As a result of this, relying solely on intra-MAV localization results in a high amount of drift and measurement outliers in that particular part of the environment, creating an inaccurate trajectory that can be seen in Figure 20(a).

To alleviate this problem, we test inter-MAV localization between MAVs 1 and 2, whenever the number of features tracked by MAV 2 falls under a threshold. The covariance of inter-MAV localization being much lower, the fusion results in a significantly more accurate trajectory estimate, as shown in Figure 20(b).

E. MAP UPDATES

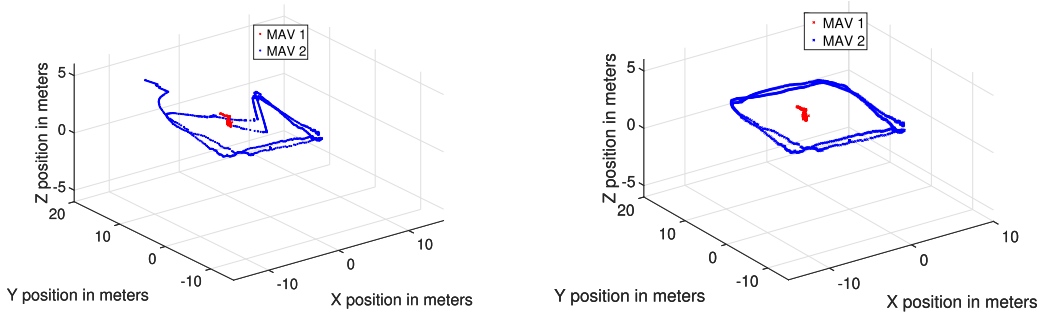
The principles behind relative pose estimation and scale recovery can also be used for performing map updates for the whole group, when the navigation is over large spaces. Figure 21 shows the process of map updates as two MAVs navigate an environment in AirSim. Table 2 offers some information regarding the accuracy of localization (compared to ground truth) while the vehicles transition through different 3D maps.

TABLE 2. RMS errors for position estimates of two MAVs localizing through map updates.

MAV	Total Distance (m)	RMSE error (m)
1	170	1.35
2	140	1.55

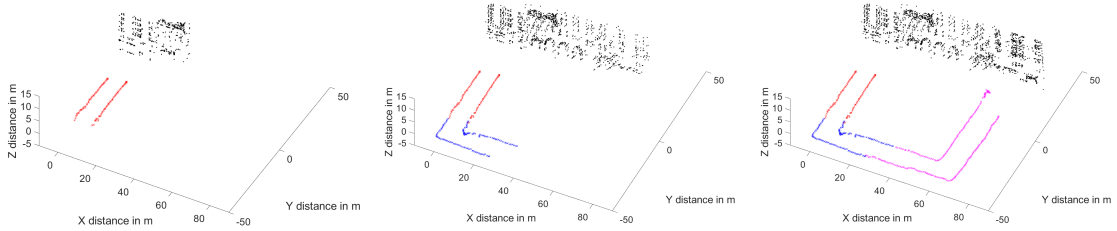
F. ALGORITHMIC REQUIREMENTS AND DISCUSSION

While in the experiments shown in the paper, we test the algorithm in an offline fashion, we perform some analysis on the possible computational and communication requirements for an online deployment. We recall that intra-MAV localization is expected to be performed on each MAV individually, whenever a new image is received, whereas inter-MAV localization is only performed in an on-demand fashion (for instance, when the uncertainty of the intra-MAV localization is deemed high according to some threshold). In order to keep communication bandwidth requirements low, the algorithm was designed in a way that it does not require images to be transferred between vehicles whenever inter-MAV localization is performed. Instead, when a client vehicle needs inter-MAV localization data from a host, a packet containing the feature keypoint locations and descriptors from the image obtained by the client is transmitted to the host. The size required per 'feature' would be a sum of the size for storing a single keypoint location: two pixel coordinates, thus a combination



(a) Intra-MAV only localization (b) Inter+intra fused localization

FIGURE 20. Indoor flight with two MAVs: intra-only localization vs. inter+intra fused localization.



(a) Step 1 (b) Step 2 (c) Step 3

FIGURE 21. Process of map updates while maintaining localization: newly captured features are appended to the global map when the number of tracked features becomes low.

of two double precision numbers and thus 16 bytes and a 512-bit descriptor vector, i.e. 64 bytes. The total requirement per feature is under 100 bytes, which stays the same for both CPU and GPU implementations of the feature algorithms. For a typical image, the total information that needs to be transferred would be in the order of kilobytes, an amount that can be handled with ease by conventional Wi-Fi networks. After the host vehicles finishes its computation of the client's pose, the pose data transmission involves sending six double precision values: three positions and three orientations, again only about 50 bytes, through a simple $\mathcal{O}(1)$ update (per host).

Delays in communication were not explicitly considered in the formulation of this problem, but one possible way of adapting to delays can be easily identified. If an inter-MAV localization process is delayed but received at a later time, it is possible to continue localizing using intra-MAV measurements, but when the inter-MAV measurement is received, the system can revert to the previous time, update with the inter-MAV measurement and then re-propagate to the present. While this would involve keeping track of not only the posterior belief of the Kalman filter, but also the measurements: because the measurements are considered to be essentially just the state vector, the space requirements for storing these are significantly lower than those of an implementation that considers map points as part of the state.

In table 3, we show a sample breakdown of how long the various modules that form the VCL algorithm took to execute in our implementation. As mentioned before, in our implementation, the algorithm was run offline: and the data

TABLE 3. Sample computational time required for each module in the VCL algorithm.

Procedure	Percentage computation	Nominal time (ms)
Map construction	35	31
Intra-MAV estimation	17	15.6
Inter-MAV estimation	19	16.8
Fusion	2	1.8
Map update	27	24

in table 3 was obtained by averaging from a pipeline that performs four functions using data from two vehicles.

1) Detect features in two images (GPU method), construct an initial map.
2) Perform intra-MAV localization for two new images I_0 and I_1.
3) Perform inter-MAV localization between I_0 and I_1 and fuse with intra estimate computed for I_1.
4) Periodically, detect features in two new images and update existing map.

The average times taken for executing these modules is shown in the table. We note here that these numbers are purely for indicative purposes: to help gauge the relative differences between, for example, inter and intra-MAV localizations. The true computational load on any given hardware depends on multiple factors such as existing load on the computer, size of images, number of features being handled during matching/estimation etc. At the same time, the algorithm can also be improved through code optimization, usage of multithreading etc.

VII. CONCLUSION

In this paper, we present a collaborative localization framework aimed at vision-based micro aerial vehicles: particularly those equipped with monocular cameras. Feature detection and matching between the MAVs enables the creation of a 3D map that is then shared between them. Once a map is available, the MAVs alternate between two modes: *intra-MAV* localization, where each MAV attempts to track features from the map and estimate its own pose; and *inter-MAV* localization, which comes into picture when the feature tracking suffers: where one/more 'host' MAVs attempt to correct the pose of a client MAV using a relative pose measurement under the assumption that feature overlap exists between the host(s) and the client. Intra-MAV and inter-MAV measurements can also be combined in a consistent fashion to result in fused localization estimates that were shown to be consistently more accurate than just intra-MAV estimates. Assuming sufficient feature overlap exists, some of the major factors influencing the effect of collaboration in this framework were shown to be the frequency at which relative measurements are obtained, as well as the number of other vehicles contributing to fusion. This algorithm was tested for image datasets coming from the MAV simulator Microsoft AirSim as well as some from real flights, which validate the accuracy of pose estimation as well as the effectiveness of the contribution of multiple MAVs, thus demonstrating that collaboration has a positive effect on localization through reduction of position/orientation estimation error.

APPLICATIONS AND FUTURE WORK

Given the formulation of the localization problem, we identify two applications where this algorithm could be a good fit. Through the collaborative nature of multiple cameras through robust relative and individual pose estimation, the VCL algorithm can be used to create what we refer to as a 'decoupled aerial stereo' system. In this system, a combination of small, cheap MAVs equipped with a camera can be converted into a variable, high-baseline stereo imaging system. These MAVs, as they are not restricted in baseline or positioning can be used to image large natural structures, buildings etc. or reconstruction through structure from motion. An application such as this would usually guarantee sufficient feature overlap between the MAVs, thus the inter-MAV localization can be used to its full potential. Another possible application for this algorithm would be in cooperative assembly as it requires precise localization between multiple MAVs as they attempt to lift/carry objects. This VCL algorithm also forms the base for our work on collaborative uncertainty-aware path planning for MAVs [58].

There are numerous possibilities for extension and future improvements of this work. We identify that adapting the current software into a real-time framework, capable of running onboard separate vehicles with communication capabilities would be an important extension. Such a deployment could also investigate the ability to handle communication delays, as well as the creation of a feedback control loop to allow for higher level features such as planning and control to take advantage of this collaborative localization scheme. The algorithms used in the VCL framework could also be extended to match a more robust SLAM framework: for instance, generating and including map uncertainty could help in better mapping. It is also possible to use a more accurate system model within the Kalman filter to match the MAV dynamics. Another direction of work could involve integrating IMU measurements into the vision based framework in order to relax the assumption of knowing an initial estimate of scale: utilizing visual-inertial data would allow for online scale estimation and propagation.

REFERENCES

[1] V. Kumar and N. Michael, "Opportunities and challenges with autonomous micro aerial vehicles," *Int. J. Robot. Res.*, vol. 31, no. 11, pp. 1279–1291, Sep. 2012.

[2] S. Vemprala and S. Saripalli, "Monocular vision based collaborative localization for micro aerial vehicle swarms," in *Proc. Int. Conf. Unmanned Aircr. Syst. (ICUAS)*, Jun. 2018, pp. 315–323.

[3] S. Vemprala and S. Saripalli, "Vision based collaborative localization for multirotor vehicles," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2016, pp. 1653–1658.

[4] S. H. Vemprala, "Vision based collaborative localization and path planning for micro aerial vehicles," Ph.D. dissertation, Dept. Mech. Eng., Texas A&M Univ., College Station, TX, USA, 2019.

[5] G. Ducard and R. D'Andrea, "Autonomous quadrotor flight using a vision system and accommodating frames misalignment," in *Proc. IEEE Int. Symp. Ind. Embedded Syst.*, Jul. 2009, pp. 261–264.

[6] N. Michael, D. Mellinger, Q. Lindsey, and V. Kumar, "The GRASP multiple micro-UAV testbed," *IEEE Robot. Autom. Mag.*, vol. 17, no. 3, pp. 56–65, Sep. 2010.

[7] J. Stowers, M. Hayes, and A. Bainbridge-Smith, "Altitude control of a quadrotor helicopter using depth map from microsoft kinect sensor," in *Proc. IEEE Int. Conf. Mechatronics*, Apr. 2011, pp. 358–362.

[8] S. Shen, N. Michael, and V. Kumar, "Autonomous indoor 3D exploration with a micro-aerial vehicle," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2012, pp. 9–15.

[9] A. Bachrach, S. Prentice, R. He, P. Henry, A. S. Huang, M. Krainin, D. Maturana, D. Fox, and N. Roy, "Estimation, planning, and mapping for autonomous flight using an RGB-D camera in GPS-denied environments," *Int. J. Robot. Res.*, vol. 31, no. 11, pp. 1320–1343, Sep. 2012.

[10] Z. Fang and S. Scherer, "Real-time onboard 6DoF localization of an indoor MAV in degraded visual environments using a RGB-D camera," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2015, pp. 5253–5259.

[11] G. Klein and D. Murray, "Parallel tracking and mapping for small AR workspaces," in *Proc. 6th IEEE ACM Int. Symp. Mixed Augmented Reality*, Nov. 2007, pp. 225–234.

[12] C. Forster, M. Pizzoli, and D. Scaramuzza, "SVO: Fast semi-direct monocular visual odometry," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2014, pp. 15–22.

[13] R. Mur-Artal and J. D. Tardos, "ORB-SLAM2: An open-source SLAM system for monocular, stereo, and RGB-D cameras," *IEEE Trans. Robot.*, vol. 33, no. 5, pp. 1255–1262, Oct. 2017.

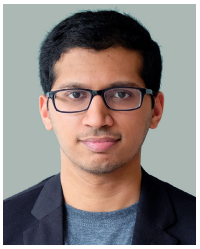
[14] J. Engel, T. Schöps, and D. Cremers, "LSD-SLAM: Large-scale direct monocular SLAM," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*. Springer, 2014, pp. 834–849.

[15] E. S. Jones and S. Soatto, "Visual-inertial navigation, mapping and localization: A scalable real-time causal approach," *Int. J. Robot. Res.*, vol. 30, no. 4, pp. 407–430, Apr. 2011.

[16] L. Meier, P. Tanskanen, L. Heng, G. H. Lee, F. Fraundorfer, and M. Pollefeys, "PIXHAWK: A micro aerial vehicle design for autonomous flight using onboard computer vision," *Auto. Robots*, vol. 33, nos. 1–2, pp. 21–39, Aug. 2012.

[17] A. I. Mourikis and S. I. Roumeliotis, "A multi-state constraint Kalman filter for vision-aided inertial navigation," in *Proc. IEEE Int. Conf. Robot. Autom.*, Apr. 2007, pp. 3565–3572.

[18] R. Mur-Artal and J. D. Tardos, "Visual-inertial monocular SLAM with map reuse," *IEEE Robot. Autom. Lett.*, vol. 2, no. 2, pp. 796–803, Apr. 2017.
[19] T. Qin, P. Li, and S. Shen, "VINS-mono: A robust and versatile monocular visual-inertial state estimator," *IEEE Trans. Robot.*, vol. 34, no. 4, pp. 1004–1020, Aug. 2018.
[20] F. Remondino, L. Barazzetti, F. Nex, M. Scaioni, and D. Sarazzi, "UAV photogrammetry for mapping and 3D modeling–current status and future perspectives," *ISPRS-Int. Arch. Photogramm., Remote Sens. Spatial Inf. Sci.*, vols. 38, p. C22, Sep. 2012.
[21] D. C. Tsouros, S. Bibi, and P. G. Sarigiannidis, "A review on UAV-based applications for precision agriculture," *Information*, vol. 10, no. 11, p. 349, Nov. 2019.
[22] A. Martinelli, F. Pont, and R. Siegwart, "Multi-robot localization using relative observations," in *Proc. IEEE Int. Conf. Robot. Autom.*, Jun. 2005, pp. 2797–2802.
[23] E. D. Nerurkar, S. I. Roumeliotis, and A. Martinelli, "Distributed maximum a posteriori estimation for multi-robot cooperative localization," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2009, pp. 1402–1409.
[24] L. C. Carrillo-Arce, E. D. Nerurkar, J. L. Gordillo, and S. I. Roumeliotis, "Decentralized multi-robot cooperative localization using covariance intersection," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Nov. 2013, pp. 1412–1417.
[25] V. Indelman, E. Nelson, N. Michael, and F. Dellaert, "Multi-robot pose graph localization and data association from unknown initial relative poses via expectation maximization," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2014, pp. 593–600.
[26] J. Knuth and P. Barooah, "Distributed collaborative localization of multiple vehicles from relative pose measurements," in *Proc. 47th Annu. Allerton Conf. Commun., Control, Comput. (Allerton)*, Sep. 2009, pp. 314–321.
[27] J. Faigl, T. Krajnik, J. Chudoba, L. Preucil, and M. Saska, "Low-cost embedded system for relative localization in robotic swarms," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2013, pp. 993–998.
[28] V. Indelman, P. Gurfil, E. Rivlin, and H. Rotstein, "Distributed vision-aided cooperative localization and navigation based on three-view geometry," *Robot. Auto. Syst.*, vol. 60, no. 6, pp. 822–840, Jun. 2012.
[29] D. Zou and P. Tan, "CoSLAM: Collaborative visual SLAM in dynamic environments," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 2, pp. 354–366, Feb. 2013.
[30] M. W. Achtelik, S. Weiss, M. Chli, F. Dellaert, and R. Siegwart, "Collaborative stereo," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Sep. 2011, pp. 2242–2248.
[31] N. Piasco, J. Marzat, and M. Sanfourche, "Collaborative localization and formation flying using distributed stereo-vision," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2016, pp. 1202–1207.
[32] C. Forster, S. Lynen, L. Kneip, and D. Scaramuzza, "Collaborative monocular SLAM with multiple micro aerial vehicles," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Nov. 2013, pp. 3962–3970.
[33] P. Schmuck and M. Chli, "Multi-UAV collaborative monocular SLAM," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2017, pp. 3863–3870.
[34] P. Schmuck and M. Chli, "CCM-SLAM: Robust and efficient centralized collaborative monocular simultaneous localization and mapping for robotic teams," *J. Field Robot.*, vol. 36, no. 4, pp. 763–781, Jun. 2019.
[35] M. Karrer, P. Schmuck, and M. Chli, "CVI-SLAM—Collaborative visual-inertial SLAM," *IEEE Robot. Autom. Lett.*, vol. 3, no. 4, pp. 2762–2769, Oct. 2018.
[36] P. F. Alcantarilla and T. Solutions, "Fast explicit diffusion for accelerated features in nonlinear scale spaces," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 7, pp. 1281–1298, Oct. 2011.
[37] P. F. Alcantarilla, A. Bartoli, and A. J. Davison, "Kaze features," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*. Springer, 2012, pp. 214–227.
[38] D. G. Lowe, "Object recognition from local scale-invariant features," in *Proc. 7th IEEE Int. Conf. Comput. Vis.*, vol. 2, Sep. 1999, pp. 1150–1157.
[39] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "ORB: An efficient alternative to SIFT or SURF," in *Proc. Int. Conf. Comput. Vis.*, Nov. 2011, pp. 2564–2571.
[40] C. P. K. Omar. (2016). *Koral*. [Online]. Available: https://github.com/komrad36/KORAL
[41] E. Rosten and T. Drummond, "Machine learning for high-speed corner detection," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*. Springer, 2006, pp. 430–443.
[42] G. Levi and T. Hassner, "LATCH: Learned arrangements of three patch codes," in *Proc. IEEE Winter Conf. Appl. Comput. Vis. (WACV)*, Mar. 2016, pp. 1–9.
[43] D. Nister, "An efficient solution to the five-point relative pose problem," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 2, Jun. 2003, pp. 2–195.
[44] L. Moisan and B. Stival, "A probabilistic criterion to detect rigid point matches between two images and estimate the fundamental matrix," *Int. J. Comput. Vis.*, vol. 57, no. 3, pp. 201–218, May 2004.
[45] L. Moisan, P. Moulon, and P. Monasse, "Automatic homographic registration of a pair of images, with a contrario elimination of outliers," *Image Process. Line*, vol. 2, pp. 56–73, May 2012.
[46] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. Cambridge, U.K.: Cambridge Univ. Press, 2003.
[47] B. Triggs, P. F. McLauchlan, R. I. Hartley, and A. W. Fitzgibbon, "Bundle adjustment—A modern synthesis," in *Proc. Int. Workshop Vis. Algorithms*. Springer, 1999, pp. 298–372.
[48] T. Ke and S. I. Roumeliotis, "An efficient algebraic solution to the perspective-three-point problem," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 7225–7233.
[49] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes in C*, vol. 2. Cambridge, U.K.: Cambridge Univ. Press, 1996.
[50] G. Chang, "Robust Kalman filtering based on mahalanobis distance as outlier judging criterion," *J. Geodesy*, vol. 88, no. 4, pp. 391–401, Apr. 2014.
[51] F. M. Mirzaei and S. I. Roumeliotis, "A Kalman filter-based algorithm for IMU-camera calibration: Observability analysis and performance evaluation," *IEEE Trans. Robot.*, vol. 24, no. 5, pp. 1143–1156, Oct. 2008.
[52] S. J. Julier and J. K. Uhlmann, "A non-divergent estimation algorithm in the presence of unknown correlations," in *Proc. Amer. Control Conf.*, vol. 4, Jun. 1997, pp. 2369–2373.
[53] G. Bradski, "The OpenCV library," *Dr. Dobb's J. Softw. Tools*, 2000. [Online]. Available: https://github.com/opencv/opencv
[54] P. Moulon, P. Monasse, R. Perrot, and R. Marlet. (2014). *Openmvg. An Open Multiple View Geometry Library*. [Online]. Available: https://github.com/openMVG/openMVG
[55] S. Agarwal and K. Mierle. (2010). *Ceres Solver*. [Online]. Available: http://ceres-solver.org
[56] D. E. King, "Dlib-ml: A machine learning toolkit," *J. Mach. Learn. Res.*, vol. 10, pp. 1755–1758, Jan. 2009.
[57] S. Shah, D. Dey, C. Lovett, and A. Kapoor, "Airsim: High-fidelity visual and physical simulation for autonomous vehicles," in *Field and Service Robotics*. Cham, Switzerland: Springer, 2017.
[58] S. Vemprala and S. Saripalli, "Vision based collaborative path planning for micro aerial vehicles," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2018, pp. 1–7.



SAI H. VEMPRALA (Member, IEEE) received the B.Tech. degree from JNTU-Hyderabad, India, in 2011, the M.S. degree from Arizona State University, in 2013, and the Ph.D. degree from Texas A&M University, in 2019. He is currently a Senior Researcher with Microsoft Corporation. His research interests lie at the intersection of robotics and machine learning. His research work has focused on robot localization and navigation, representation learning, and robust computer vision. His doctoral thesis was on the topic of collaborative localization and path planning for vision-based unmanned aerial vehicles.

SRIKANTH SARIPALLI (Senior Member, IEEE) received the B.E. degree (Hons.) from the Birla Institute of Technology and Sciences, Pilani, India, in 1999, and the M.S. and Ph.D. degrees from the University of Southern California, in 2002 and 2007, respectively. He is currently a Professor of mechanical engineering and the Director of the Center for Autonomous Vehicles and Sensor Systems, Texas A&M University. He is also a Roboticist with research interests in unmanned systems in general and aerial vehicles in particular. His research interest includes robotic exploration particularly in air and ground vehicles and necessary foundations in perception, planning, and control for this domain. He was the Program Chair with the International Conference on Unmanned Aerial Systems 2015 and a member of AIAA.