Vision Based Collaborative Localization for Multirotor Vehicles

Sai Vemprala and Srikanth Saripalli[1]

***Abstract*— We present a framework for vision based localization for two or more multirotor aerial vehicles relative to each other. This collaborative localization technique is built upon a relative pose estimation strategy between two or more cameras with the capability of estimating accurate metric poses between each other even through fast motion and continually changing environments. Through synchronized feature detection and tracking with a robust outlier rejection process, classical multiple view geometry concepts have been utilized for obtaining scale-ambiguous relative poses, which are then refined through reconstruction and pose optimization to provide a metric estimate. Furthermore, we present the implementation details of this technique followed by a set of results which involves evaluation of the accuracy of the pose estimates through test cases in both simulated and real experiments. Test cases include keeping one camera stationary as the other is mounted on a quadrotor which is then flown through various types of trajectories. We also perform a quantitative comparison with a GPS/IMU localization technique to demonstrate the accuracy of our method.**

I. INTRODUCTION

Micro aerial vehicles (MAV) have recently started becoming the platforms of choice for many robotic applications. Especially when considering multirotor vehicles, their small size, light weight, relative inexpensiveness and prototyping ease; as well as their hover ability make them good options for navigating in tight spaces and in applications such as disaster management, reconnaissance, aerial imaging and such. As the intended applications for MAVs become complex, a need for precise localization becomes essential. Hence, state estimation and control of micro aerial vehicles are currently active research areas.

Conventionally, the task of determining the pose of multirotors has been handled by the combination of a global positioning system (GPS) receiver and an inertial measurement unit (IMU). This particular solution, although simple and straightforward to implement, is prone to several cases of inaccuracy: GPS reception cannot always be guaranteed, it can suffer from obstructions and multi-path while in cluttered environments and/or low altitude flights, and is completely inapplicable for indoor flight. Considering these disadvantages of GPS/IMU fused navigation, past research has tried to identify other kinds of sensing such as vision sensors and laser scanners that can enhance localization accuracy. In this regard, vision sensors have demonstrated potential for both inferring localization information and simultaneous mapping. This, coupled with the decreasing size of vision sensors, have made them viable options for exteroceptive sensing in micro aerial vehicles.

In this paper, we present the theoretical framework and implementation of a new methodology for collaborative localization of two or more MAVs using cameras as the primary sensors. Figure 1 shows the concept of this localization approach with two MAVs. Given two vehicles equipped with monocular cameras observing a scene, one of the vehicles is considered as the reference. For the first pair of images obtained from the two cameras, common features visible from both are isolated and matched, which are then be used to reconstruct the scene in 3D by estimating the relative pose between the vehicles. This 3D reconstruction is stored as a point cloud and as new images are captured, the new feature matches are tracked to determine how many original feature points are still visible. These tracked 3D points and the respective 2D projections for each camera are used to obtain the position and orientation of that particular vehicle. Hence, one MAV acts as a reference or the origin for the coordinate system, and the second vehicle (and possibly more) is localized with respect to the first. As the common features currently visible are tracked over images, once the number of features tracked falls under a certain threshold, it is adaptively decided that it is time to clear the current point cloud and create a new one, so that localization will not fail due to changes in the scene and/or fast motion of the MAVs.

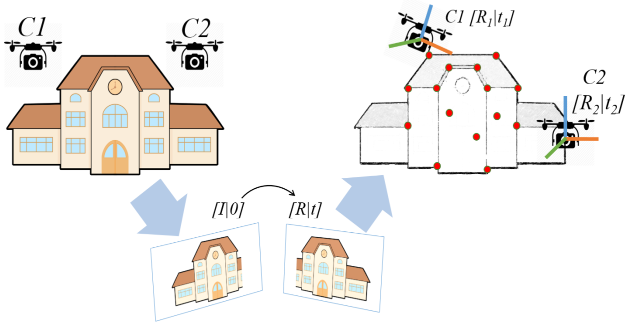

Fig. 1: Collaborative localization between two MAVs

This collaborative localization mechanism provides two advantages compared to other localization approaches. Visual odometry and SLAM techniques have been discussed extensively in the literature, but applying separate algorithms independently on all vehicles both increases the computational load on each one; as well as introducing multiple sources of error independent of each other. Hence, it can be more intuitive to perform localization relative to one another, thus keeping all vehicles in one frame of reference. Secondly,

[1]Sai Vemprala and Srikanth Saripalli are with the School of Earth and Space Exloration, Arizona State University, Tempe, Arizona, USA. svemprala@asu.edu, Srikanth.Saripalli@asu.edu

MAV swarms focus on small, lightweight vehicles and decentralized control and communication between individual agents, hence most vehicles are computationally constrained and are intended for only low-level local sensing rather than containing expensive sensors and performing complex sensor fusion on each platform [1]. Collaborative localization also does not need to be limited to vision, and can be extended to other metrics like wireless signal strength.

II. PREVIOUS WORK

Previous work on localization of multirotor vehicles can be classified into different groups based on the mode of sensing used. Solutions involving GPS/IMU fused estimation are currently available as commercial products. Initial work on non-GPS localization that focused on precise navigation concentrated on using advanced external tracking systems, or artificial marker distribution [2] in the flight space, thus simplifying pose estimation.

Augmentation of GPS/INS navigation with vision has also been discussed in the literature. [3] presents a technique of georeferencing images obtained from the downward facing camera of a UAV and improving pose estimation through a Kalman filter. Further vision-specific work has utilized stabilization mechanisms using optical flow, as in the commercially available PX4Flow camera [4]. More recently, monocular SLAM strategies have been investigated onboard multirotor vehicles, which have aimed to remove scale ambiguity either by fusing vision data with an IMU [5] or using multiple initial views [6] to obtain metric scale information. Similar SLAM strategies have been studied using stereo cameras [7] and RGBD sensors such as the Microsoft Kinect [8] with intended applications onboard robotic vehicles. [9] presents a relative pose estimation technique for a single ground vehicle which enhances accuracy by using the relative rotation angle from a different sensor such as an IMU as an estimate.

Collaborative/relative localization has been studied in the literature as well, although mainly in the realm of ground robots. Cooperative localization strategies using IR LEDs has been presented in [10], and using fiducial markers in [11]. Another cooperative estimation framework using RGBD sensors on ground robots has been discusssed in [12]. Relative localization using robot to robot distance measurements optimized using an iterative least squares method has been presented in [13], which computes 3 DoF relative transformations.

III. METHODOLOGY

The main goal of this research is to develop the framework to obtain relative metric poses of the vehicles with respect to each other when multiple frames of images are provided from two sources. In order to achieve this, we make two important assumptions:

1) Both of the cameras are calibrated, and the camera intrinsics and distortion coefficients are known.
2) The initial distance and difference between heading between the cameras are accurately known.

A high level description of the algorithm pipeline is given in figure 2, and the rest of this section describes the various modules in detail.

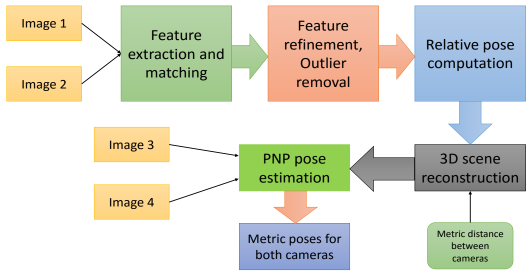
Image 1
Image 2
Feature extraction and matching
Feature refinement, Outlier removal
Relative pose computation
Image 3
Image 4
PNP pose estimation
3D scene reconstruction
Metric poses for both cameras
Metric distance between cameras

Fig. 2: Brief schematic of algorithm

A. Feature tracking and matching

To proceed with pose estimation, the first step is to identify common features being observed by both cameras through a robust and error-free feature identification and matching framework. Multi-scale feature detection approaches such as SIFT and SURF have been in extensive use in the past, yet one important limitation of SIFT or SURF is their computational complexity [14]. When dealing with long-term feature detection using cameras capable of producing a number of frames per second, these conventional approaches can cause an issue for on-board implementations. Given the recent increase in the popularity of mobile and computationally constrained devices and robotic platforms, new feature detection mechanisms such as ORB and BRISK have been developed that are faster than SIFT and SURF [14].

In this research, our feature detection algorithm works by computing what are known as Accelerated KAZE (AKAZE) features that are faster to compute than SIFT/SURF and demonstrate better accuracy than ORB/BRISK [15]. AKAZE is a multi-scale feature detection and description approach in non-linear scale spaces. AKAZE features also help maintain a low computational demand by using Modified Local Difference Binary (M-LDB) descriptors. Once features and their descriptors have been identified and constructed, the next step is to match the features obtained from different views accurately. We perform feature matching using a brute force matching technique, utilizing a Hamming distance metric, as brute force matching results in more precise matching than a nearest neighbor matching algorithm.

B. Feature refinement and relative pose

To initialize the algorithm with a reconstructed representation of the environment around, the relative pose between the cameras needs to be computed. In this algorithm, we achieve this step using the essential matrix E, that encodes the transformation between the multiple views. We use the five-point algorithm to compute the essential matrix [16]. Unlike the algorithms such as the 8-point algorithm that are used to compute the fundamental matrix, 5-point algorithm does not have a degeneracy case if there are coplanar points [16].

Automated vision based pose estimation using hundreds of frames of image data requires an accurate and reliable estimation method, yet, the essential matrix can be susceptible to noise [17]. We create a more robust approach for relative pose estimation by initializing the five point algorithm through a RANSAC scheme to filter outliers. The RANSAC scheme solves for the distance error metric between the putative point correspondences, i.e., for a given point correspondence pair x_i and x'_i, with $\hat{x}_i$ and $\hat{x}'_i$ defined with respect to a pair of matching epipolar lines, the RANSAC algorithm examines the number of correspondences that lie within a threshold of the distance error. The distance error metric can be expressed as follows.

$$e = d(x_i, \hat{x_i})^2 + d(x'_i, \hat{x'_i})^2 \tag{1}$$

Minimizing e thus is equivalent to minimizing the sum of squared distances (SSD) of the matched points from the epipolar lines. Once the essential matrix is obtained, it can be decomposed to obtain the rotation matrix and the translation vector.

Algorithm 1 Two-camera localization

```
1: procedure CONSTRUCTSCENE(im1, im2, scale)
2:     $f_1$, $f_2 \leftarrow$ Detect features in images im1, im2
3:     $M \leftarrow$ Brute force matching between $f_1$ and $f_2$
4:
5:     for each $m_i \in M$ do:
6:         $\bar{m_i}(\bar{x}, \bar{x}') \leftarrow min(d(x_i, \hat{x_i})^2 + d(x'_i, \hat{x'_i})^2)$
7:     $\bar{M} = (\bar{m_1}...\bar{m_n})$          ▷ Pairs of inlier feature points
8:     $\bar{i_1}$, $\bar{i_2} \leftarrow \bar{M}$          ▷ Individual feature points
9:     $E \leftarrow$ 5-point algorithm($\bar{I_1}, \bar{I_2}, K_1, K_2, d_1, d_2$)
10:    $R, t \leftarrow svd(E)$
11:    $P_1$, $P_2 \leftarrow [I|0]$, $[R|t]$
12:    $O \leftarrow$ triangulate($\bar{i_1}, \bar{i_2}, P_1, P_2$)    ▷ 3D object points
13:    return $O * scale, i_1, i_2$
14: Initial scale = s
15: loop
16:     Read images im1, im2 from cameras C1, C2
17:     $O, i_1, i_2 \leftarrow$ constructScene(im1, im2, s)
18:     while featureCountFlag is not set do
19:         Read next set of images
20:         $f_1$, $f_2 \leftarrow$ Detect features in images im1, im2...
21:         $\bar{f_1}$, $\bar{f_2} \leftarrow$ track($f_1, f_2, i_1, i_2$)
22:         if $size(f_1) < threshold$ then
23:             featureCountFlag $\leftarrow 1$
24:             break
25:         $R_1, t_1 \leftarrow$ PNP($O, \bar{f_1}$)
26:         $R_2, t_2 \leftarrow$ PNP($O, \bar{f_2}$)
27:         Compute new scale factor $s$
```

C. Triangulation and metric pose estimation

Once estimates of the rotation and translation of the second view have been obtained relative to the first, the next step would be to reconstruct the scene through triangulation. We use the Hartley-Sturm optimal triangulation procedure [18] to create a point cloud of the triangulated features. When the algorithm is initialized for the first time, the metric scale between the cameras (vehicles) is known and provided (in the context of multirotors, assumption 2 can be satisfied by placing the vehicles in specific spots before the flight begins), which allows for the construction of an accurate point cloud. This point cloud is then used for the successive images to estimate the poses of all cameras.

To estimate the pose of any camera viewing the same scene, we perform the first step of feature detection and track which features from the constructed point cloud are still visible. Once the set of tracked features is obtained, as the positions of the cameras (vehicles) would not be constant; to obtain the most current metric estimate of the pose, we implement the process of estimating the pose of a camera given a set of 3D-2D correspondences, which is known as the perspective-n-point (PnP) problem. While dealing with noisy feature points and rapidly changing environments, a reasonably accurate solution is required to this problem. Hence, we run a Levenberg-Marquardt optimization procedure coupled with a simple Kalman filter for position and orientation, which obtains the pose through iteratively minimizing the reprojection error for the views, and then propagates it through the Kalman filter to reduce noise. Depending on how the MAVs are moving, transient occurrences such as features going out of view could happen, and the Kalman filter helps obtain a reasonable estimate of the pose in case of such issues by reducing the noise. The Kalman filter is defined over an 18-element state vector that consists of the 3D position and its first and second derivatives, as well as the roll, pitch, yaw angles and their first and second derivatives.

Until the end of a certain window in time, this point cloud is stored and used to estimate the metric poses using the 3D to 2D correspondences, based on the number of features still visible from the images at any given instant. Once the number of features that are both part of the point cloud and are still visible from the cameras falls under a certain threshold, feature matching is performed again and a new point cloud is constructed. It has to be noted that until a new point cloud is reconstructed, the only reference is a point cloud from the past, which necessitates PNP pose estimation for both cameras as the vehicles are in motion. As every instant of image capture has its own value of metric pose, the most recent value of the scale factor is computed from the translation vectors and is used to scale the point cloud up to its real value. Pseudocode for the algorithm can be seen in algorithm 1.

IV. IMPLEMENTATION

We have implemented the collaborative localization algorithm (CLA) described in the previous section on datasets both from simulation and from real cameras. The simulated experiments were used for preliminary validation of the algorithm and were conducted in Gazebo in the RotorS simulator [19] using models of an AscTec Firefly multirotor that were modified to include monocular cameras. Three

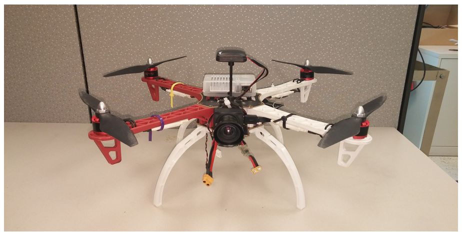
Fig. 3: DJI F450 test platform with a PointGrey camera

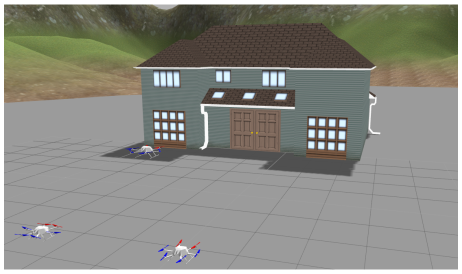
Fig. 4: Simulation environment

multirotors were used in this experiment and images from the vehicles were used to obtain the pose estimates. The simulator also contains support for odometry sensors for each multirotor and data from that was used as ground truth for comparison.

For the real experiments, we used a two-camera setup of Point Grey Chameleon3 USB3 cameras [20], with one camera held stationary and the other one installed on a quadrotor. The images were captured at 15 Hz, and with an effective resolution of 1288x964. The quadrotor used was a custom built DJI F450 platform, as seen in figure 3, with the 3DRobotics PIXHAWK [21] acting the main flight controller. The vehicle was equipped with a UBlox GPS receiver, the data from which is fused with the internal IMU by the PIXHAWK and propagated through an extended Kalman filter (EKF) to provide pose estimates for comparison. An ODROID-U3 computer was installed on the quadrotor to perform synchronized image logging. The pose estimation was performed offline. OpenCV [22] was utilized as the base level framework for the algorithm as well as the implementations of AKAZE detection and the PNP algorithm.

V. RESULTS AND DISCUSSION

A. Simulation

A screenshot of the simulation test case can be seen in figure 4. The vehicle in the center was considered as the leader, and the others were made to localize with respect to the initial position of the leader. The vehicles were flown through square shaped trajectories for a total of approximately 8m about all axes. The simulator provides ground truth estimates from a simulated odometry sensor, which were compared with the CLA results. Figure 5 shows a plot of the comparison. The average RMS error between the ground truth and CLA estimates was 5.83 cm.

B. Real flight data

1) Handheld experiment: To test the accuracy of the metric distances computed by the CLA while using the

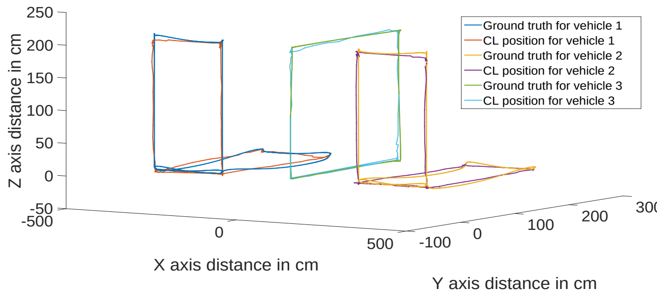

Fig. 5: Positions of three multirotors from simulation

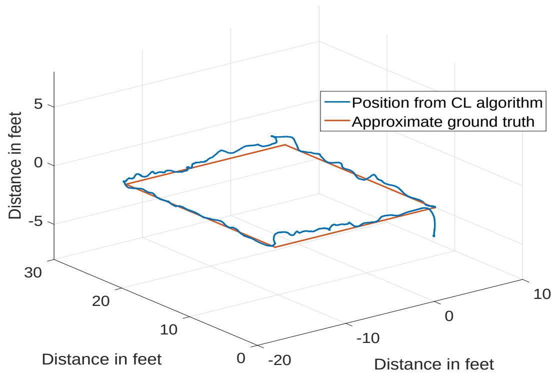

Fig. 6: Position of handheld quadrotor

real cameras, the first experiment we carried out was where the quadrotor containing the camera was held by hand and moved approximately along a rectangle of dimensions 18ftx22ft, then put on the ground. Figure 6 shows the plot of the positions obtained by the CLA, compared approximately with a rectangle of dimensions mentioned above, which shows that accurate metric estimates were maintained through motion.

2) Quadrotor flights: As the next step, we kept one camera stationary while the other was installed on the F450 quadrotor, which was flown manually using teleoperation through different kinds of trajectories. A picture of the setup can be seen in figure 7. Some cases examined here are:

1) Quadrotor flown forward, left, backwards and right towards the start point, repeated twice.
2) Quadrotor flown forwards and backwards twice, and then left, forward, right and back towards the start point.
3) Quadrotor flown forwards and backwards twice with an

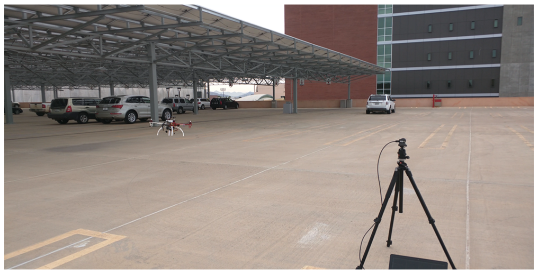
Fig. 7: Stationary camera and flying quadrotor

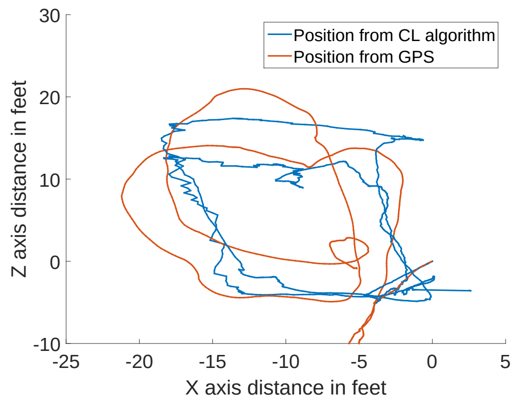

Fig. 8: Positions from CLA and GPS for case 1

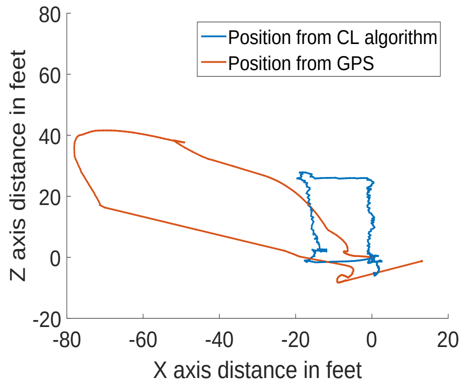

Fig. 9: Positions from CLA and GPS for the second part of case 2

increase and subsequent decrease in altitude for each leg and then flown upwards and down to the start point.

The flight time for each case was approximately 60-90 seconds. The distance from the observed scene from the locations of the cameras was around 200-250 feet. Vehicle poses obtained for these trajectories from the CLA were compared to the data obtained from the EKF processed GPS/IMU fused estimation from the PIXHAWK. Before the flights began, the quadrotor was left stationary until it could receive satellite based augmentation, and thus be able to perform corrections to the GPS measurements. The graphs in figures 8-13 show the performance of the CLA as compared with the EKF propagated pose data from the PIXHAWK.

It has been noticed that the positions of the quadrotor were more reliably tracked by the vision based localization than the GPS. The initialized point clouds contained around 250-300 feature matches, and CLA pose estimation began to suffer when the number of matches fell below 50, which was used as a threshold for reinitialization. In case 1, shown through figure 8, as the quadrotor was taken through the rectangular pattern twice, the GPS estimates resulted in a noticeable amount of accumulated error as well as deviations from the actual path taken by the quadrotor. Figure 9 represents the last part of case 2, where the quadrotor was flying faster than 2-3 m/s. This specific trajectory caused both a significant amount of tilt on the vehicle as well as it flying close to structures, which caused a large amount of drift in the GPS localization, whereas the CLA estimates stayed true to the actual position. Case 3 involved altitude changes, and these changes were tracked more faithfully by the CLA than the barometer and GPS fused altitude from the PIXHAWK's EKF (figure 10). The latter estimates were smoother and were slower in reaching their final value as the vehicle repeated its motions.

The orientation data obtained for case 2 is presented through graphs in figures 11, 12 and 13 where it's compared to the roll, pitch and yaw angles obtained from the PIXHAWK. The Kalman filtering of the CLA reduces the noise on the data to a higher extent than the PIXHAWK, but is still reliably able to track big changes in the angles given they're consistently present for a few frames. The roll angles increase and decrease towards the end of the graph as the quadrotor

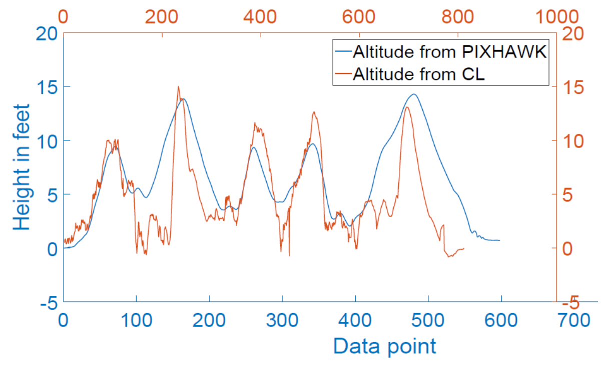

Fig. 10: Altitude from CLA and PIXHAWK for case 3

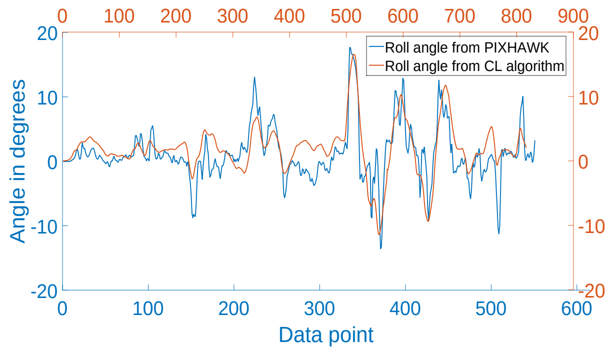

Fig. 11: Comparison of roll angles

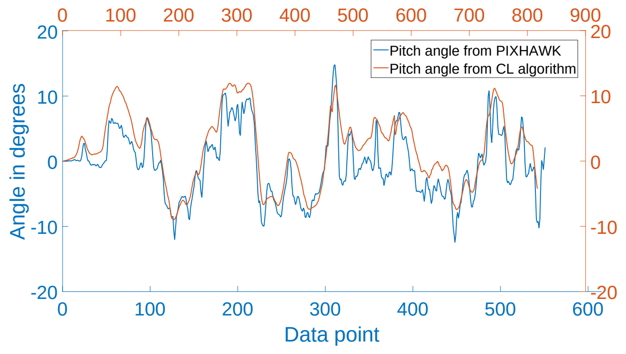

Fig. 12: Comparison of pitch angles

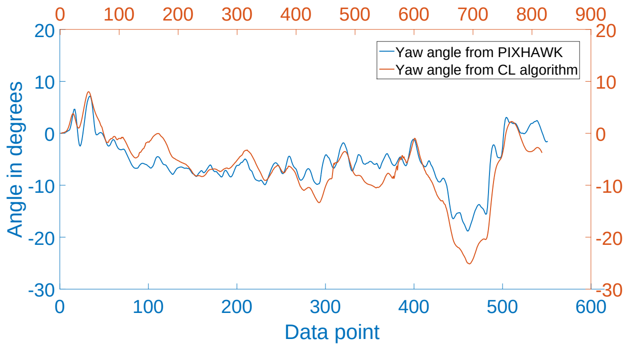

Fig. 13: Comparison of yaw angles

moves through the rectangular trajectory, and the pitch angles can be seen to be changing noticeably throughout as the first part involved a forward-backward motion.

VI. CONCLUSIONS AND FUTURE WORK

In this paper, we show that relative pose estimation between a decoupled camera setup can be used for accurate localization of the cameras individually. This algorithm is aimed at performing collaborative localization between multirotor vehicles and shows significant improvement over the GPS/IMU localization technique and can be used for any platforms with monocular cameras installed. We have validated the algorithm through tests in simulation and real experiments involving a quadrotor platform. The algorithm does not depend on any other sensory inputs, and can be easily extended to groups of more than two based on the computational/communication capacity of the platforms. There is room for further optimization in terms of speed.

Our future work aims at online implementation of this algorithm and possible speed optimizations. One application for this system as part of our future work is what we refer to as a decoupled aerial stereo (DAS) system. DAS is a system containing two multirotor MAVs, each with its own monocular camera, made to behave as a conventional stereo rig would, with the decoupled cameras used for variable baseline stereo imaging and structure from motion (SFM) applications where a normal stereo camera setup is impossible to use due to mechanical and/or access limitations, because of wide baselines and/or high altitudes. In case the cameras lose common feature points, a fallback to backup sensing such as GPS/IMU could be performed. Every time the point cloud is updated with sufficient number of features, the most current positions would be stored as backup GPS waypoints such that in case of a lack of common features, the vehicles can navigate to the last known waypoints in order to reinitialize. We also plan to investigate collaborative localization strategies adapted to other modes of sensing such as thermal cameras, wireless signal strength or scientific equipment, thus capitalizing on the kind of sensors that are more readily available rather than creating demand for a specific type of sensor.

REFERENCES

[1] Jin Yining, Wu Yanxuan, and Fan Ningjun. Research on distributed cooperative control of swarm uavs for persistent coverage. In *Control Conference (CCC), 2014 33rd Chinese*, pages 1162–1167, July 2014.

[2] S. Chhaniyara, K. Althoefer, Y.H. Zweiri, and L.D. Seneviratne. A novel approach for self-localization based on computer vision and artificial marker deposition. In *Networking, Sensing and Control, 2007 IEEE International Conference on*, pages 139–144, April 2007.

[3] Fernando Caballero, Luis Merino, Joaquin Ferruz, and Aníbal Ollero. Vision-based odometry and slam for medium and high altitude flying uavs. *Journal of Intelligent and Robotic Systems*, 54(1-3):137–161, 2009.

[4] Dominik Honegger, Lorenz Meier, Petri Tanskanen, and Marc Pollefeys. An open source and open hardware embedded metric optical flow cmos camera for indoor and outdoor applications. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pages 1736–1741. IEEE, 2013.

[5] Eagle S Jones and Stefano Soatto. Visual-inertial navigation, mapping and localization: A scalable real-time causal approach. *The International Journal of Robotics Research*, 30(4):407–430, 2011.

[6] Georg Klein and David Murray. Parallel tracking and mapping for small ar workspaces. In *Mixed and Augmented Reality, 2007. ISMAR 2007. 6th IEEE and ACM International Symposium on*, pages 225–234. IEEE, 2007.

[7] Taihú Pire, Thomas Fischer, Javier Civera, Pablo De Cristóforis, and Julio Jacobo Berlles. Stereo parallel tracking and mapping for robot localization. In Institute of Electrical and Electronics Engineers (IEEE), editors, *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, September 2015.

[8] Ji Zhang, Michael Kaess, and Sushil Singh. Real-time depth enhanced monocular odometry. In *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*, pages 4973–4980. IEEE, 2014.

[9] Bo Li, Liang Heng, Gim Hee Lee, and Marc Pollefeys. A 4-point algorithm for relative pose estimation of a calibrated camera with a known relative rotation angle. In *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*, pages 1595–1601. IEEE, 2013.

[10] Matthias Faessler, Elias Mueggler, Karl Schwabe, and Davide Scaramuzza. A monocular pose estimation system based on infrared leds. In *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pages 907–913. IEEE, 2014.

[11] Vikas Dhiman, Julian Ryde, and Jason J Corso. Mutual localization: Two camera relative 6-dof pose estimation from reciprocal fiducial observation. In *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*, pages 1347–1354. IEEE, 2013.

[12] Xiaoqin Wang, Y Ahmet Şekercioğlu, and Tom Drummond. Vision-based cooperative pose estimation for localization in multi-robot systems equipped with rgb-d cameras. *Robotics*, 4(1):1–22, 2014.

[13] Xun S Zhou, Stergios Roumeliotis, et al. Robot-to-robot relative pose estimation from range measurements. *Robotics, IEEE Transactions on*, 24(6):1379–1393, 2008.

[14] O. Miksik and K. Mikolajczyk. Evaluation of local detectors and descriptors for fast feature matching. In *Pattern Recognition (ICPR), 2012 21st International Conference on*, pages 2681–2684, Nov 2012.

[15] Pablo F Alcantarilla and TrueVision Solutions. Fast explicit diffusion for accelerated features in nonlinear scale spaces. *IEEE Trans. Patt. Anal. Mach. Intell*, 34(7):1281–1298, 2011.

[16] David Nistér. An efficient solution to the five-point relative pose problem. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 26(6):756–770, 2004.

[17] Quan-Tuan Luong and Olivier D Faugeras. The fundamental matrix: Theory, algorithms, and stability analysis. *International journal of computer vision*, 17(1):43–75, 1996.

[18] Richard I Hartley and Peter Sturm. Triangulation. *Computer vision and image understanding*, 68(2):146–157, 1997.

[19] Fadri Furrer, Michael Burri, Markus Achtelik, and Roland Siegwart. Robot operating system (ros). *Studies Comp.Intelligence Volume Number:625*, The Complete Reference (Volume 1)(978-3-319-26052-5):Chapter 23, 2016. ISBN:978-3-319-26052-5.

[20] Point Grey Inc. Firefly mv camera. `http://www.ptgrey.com/firefly-mv-usb2-cameras`, 2015.

[21] Lorenz Meier, Petri Tanskanen, Friedrich Fraundorfer, and Marc Pollefeys. Pixhawk: A system for autonomous flight using onboard computer vision. In *Robotics and automation (ICRA), 2011 IEEE international conference on*, pages 2992–2997. IEEE, 2011.

[22] G. Bradski. Opencv library. *Dr. Dobb's Journal of Software Tools*, 2000.