Visual Inertial Odometry Swarm: An Autonomous Swarm of Vision-Based Quadrotors

Aaron Weinstein, Adam Cho, Giuseppe Loianno, and Vijay Kumar

***Abstract*—In this letter, we present the system infrastructure for a swarm of quadrotors, which perform all estimation on board using monocular visual inertial odometry. This is a novel system since it does not require an external motion capture system or GPS and is able to execute formation tasks without inter-robot collisions. The swarm can be deployed in nearly any indoor or outdoor scenario and is scalable to higher numbers of robots. We discuss the system architecture, estimation, planning, and control for the multirobot system. The robustness and scalability of the approach is validated in both indoor and outdoor environments with up to 12 quadrotors.**

***Index Terms*—Aerial systems, applications, swarms, visual-based navigation.**

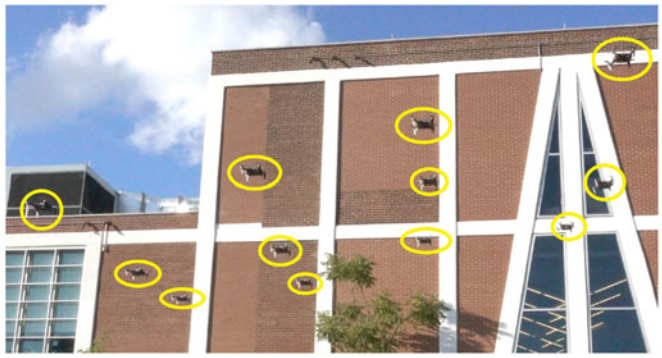


Fig. 1. The VIO Swarm during outdoor flight without the use of GPS or External Motion Capture system.

I. INTRODUCTION

QUADROTORS equipped with on-board sensors are popular Micro Aerial Vehicles (MAV) due to their size, ability to hover, and navigate complex 3D environments. They can be used for a variety of applications ranging from inspection to search and rescue. Collaborative teams of quadrotors, or Swarms, are able to cover larger areas, gather more information, and are resilient to agent failure.

Previous Swarm implimentations have relied heavily on external position feedback such as Motion Capture or Global Positioning Systems (GPS). Motion capture systems provide high precision robot tracking by merging images from fixed cameras positioned around a working volume. These systems have enabled prior works on the control and navigation of teams aerial vehicles [1]–[3]. While useful, these systems are expensive, time consuming to setup, and limit a swarm to operate within fixed tracking boundaries. Additionally, position information for all robots must be broadcast at high frequency from a central computer. This centralized communcation channel is a single point of system failure with a limited bandwith, which constrains the robustness and maximum size of a swarm. Alternatively, in other works [4] robots have individually used GPS to determine their position onboard. However, GPS information suffers from a lack of precision and is prone to interference in indoor settings, limiting these swarms to remain spaced far apart and only operate outdoors.

Visual Odometry offers an alternate solution to robot localization in which robots localize onboard by perceiving their environment with cameras. By performing localization onboard, individual robots do not rely on a central source of information to operate. Unlike Motion Capture or GPS, Visual Odometry can operate without boundaries in both indoor and outdoor environments.

Previous vision based multi robot systems have utilized fiducial markers placed in the environment to guide robot localization and obstacle avoidance [5]–[7]. In these works, robots must stay within an area where sufficient artificially placed tags are visible. Other works placed fiducial tags on robots, allowing each to determine its relative position within the swarm [8], [9]. In this approach, each robot must maintain a line of sight to a sufficient number of neighbors for formation stability. In [10], two MAVs map the same environment. Loop closure detection between different cameras, map merging, and concurrent map access are performed on a ground station. In [11], a framework based on RGB-D sensors for cooperative localization and mapping, is presented. In this case, a monocular vision algorithm is used and the scale factor problem discussed in [10]. The presented work also builds from the previous work [12] which was a first attempt at a vision based autonomous swarm. This solution was limited in the number of robots due to the large

Manuscript received September 10, 2017; accepted January 9, 2018. Date of publication January 31, 2018; date of current version March 12, 2018. This letter was recommended for publication by Associate Editor E. Johnson and Editor J. Roberts upon evaluation of the reviewers' comments. This work was supported in part by Qualcomm Research, in part by the Army Research Laboratory under Grants W911NF-08-2-0004 and W911NF-17-2-0181, in part by the Office of Naval Research under Grants N00014-07-1-0829 and N00014-14-1-0510, in part by the Army Research Office under Grant W911NF-13-1-0350, in part by the National Science Foundation under Grant IIS-1426840 and Grant IIS-1138847, and in part by the Defense Advanced Research Projects Agency under Grants HR001151626 and HR0011516850, *(Corresponding author: Aaron Weinstein.)*

The authors are with the General Robotics, Automation, Sensing, and Perception Laboratory, University of Pennsylvania, Philadelphia, PA 19104 USA (e-mail: aarow@seas.upenn.edu; choadam@seas.upenn.edu; loiannog@seas.upenn.edu; kumar@seas.upenn.edu).

Digital Object Identifier 10.1109/LRA.2018.2800119

TABLE I
LIST OF COMPONENTS FOR EACH PLATFORM WITH CORRESPONDING COSTS AND LINKS

Component	Cost	Link
Thunderpower ProLiteX Series 910 mAh 2S 25C LiPo Battery	$14.99	http://www.thunderpowerrc.com/Products/910-mAhX/TP910-2SPX25J
QualcommSnapdragon Flight Development Board	$399	https://shop.intrinsyc.com/collections/all/products/qualcomm-snapdragon-flight-sbc
QualcommSnapdragon Flight Stereo Vision Add-on Kit	$399	https://shop.intrinsyc.com/collections/all/products/qualcomm-snapdragon-flight-stereovision-add-on-kit
QualcommSnapdragon Flight Electronic Speed Controller	$144	https://shop.intrinsyc.com/collections/all/products/qualcomm-electronic-speed-control-board
Tiger Motor MT-1306-10 3100 kv Brushless Motors	$25	https://www.getfpv.com/motors/tiger-motor-mt-1306-10-3100kv.html

footprint of each vehicle as well as the requirement to make custom vehicles.

Other recent works utilized a commercially available vehicle which can perform fast motions as presented in [13]. This platform performs Visual Inertial Odometry (VIO) by combining information from an Inertial Measurement Unit (IMU) and a downward facing camera. This robot requires only a sufficiently textured ground surface and can operate under varied lighting conditions both indoors and outdoors.

The VIO-Swarm, shown in Fig. 1, leverages advancements in vision based MAVs and extends them to multi robot formation flight. The primary setup requirement of the swarm is that robots must takeoff from known locations. Otherwise, the robots can fly anywhere there is sufficient visual texture. Compared to previous works, our solution presents the following main differences. First, we developed a scalable and extensible architecture for controlling multiple vision based quadrotors. We use the term scalable to refer to the ease of adding additional agents to the system without sacrificing overall performance. Second, this is the first time that perception, planning and control are combined for autonomous navigation of multiple interchangeable aerial vehicles (up to 12 quadrotors) without relying on GPS or an external motion capture system. Finally, we use commercially available components and provide our source code online.[1] We believe at the time of publishing that this is the largest swarm of autonomous quadrotors that does not rely on motion capture or GPS position.

The letter is organized as follows. In Section II, we discuss individual vehicle architecture, which is broken down into quadrotor selection, control, and estimation. In Section III, we discuss the system architecture of each individual robot and the centralized ground station. Section IV describes the trajectory planning methods used. Section V presents experimental results from a single robot, indoor multi-robot, and outdoor multi-robot experiments. Section VI concludes the work and discusses future applications.

II. VEHICLE ARCHITECTURE

A. Quadrotor Selection

Choosing the right quadrotor was crucial to the success of the VIO-Swarm. We chose a platform with precise visual localization which has been utilized for lightweight agile autonomous applications in past works such as [13], [14].

Fig. 2. A Single robot in flight.

This chosen platform, shown in Fig. 2, strikes a balance between flight time, size, sensing capabilities, computational power, and agility. Each robot features advanced processing power and a versatile sensing suite. Computations are handled by a Qualcomm Snapdragon Flight computation board, which has a Qualcomm Hexagon DSP, Bluetooth connectivity, 802.11n Wi-Fi. Of the 4 available cores on the CPU, only 1.5 cores were used for planning, estimation, and control, with attitude control performed entirely on the DSP board. The remaining computation power can be utilized in the future to perform more complex calculations. The robots were equipped with a downward facing VGA camera with 160° field of view, a forward facing VGA stereo camera pair, and a 4K video camera. However, only the downward facing camera was used in this work. The quadrotor frame is comprised of carbon fiber balsa wood composite with 3D printed PLA landing feet. The quadrotor has Tiger motors, 4" propellers, Snapdragon flight electronic speed controllers, and a 2S Thunder Power ProLite battery, which provided up to 8 minutes of flight time. Each robot weighs 250 g and has a tip-to-tip diameter of 32 cm. Table I provides a detailed list of all the components and corresponding links to build the proposed system. However, for less experienced users, we suggest the use of the commercially available platform released by World Electronics,[2] which utilizes the same computation and sensing componenets and is supported by our software framework.

[1] https://github.com/orgs/MultiRobotUPenn/dashboard

[2] https://worldsway.com/product/dragon-drone-development-kit/

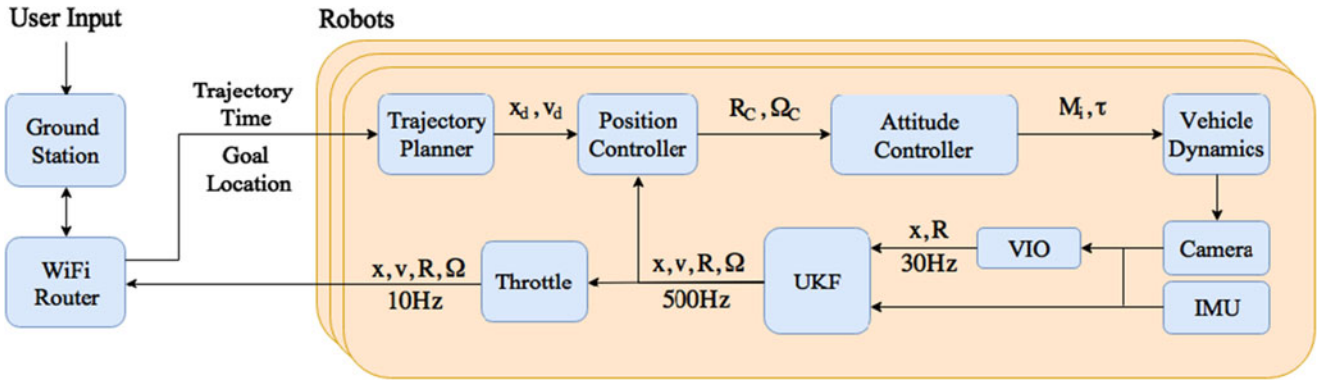


Fig. 3. Diagram of major system components.

The majority of calculations were performed on-board each robot including: planning, control, and estimation. This lightens the computation load of the centralized ground station, which is particularly important when scaling swarm size. An overview of the distribution of the computation and communication is shown in Fig. 3.

B. Control

In this work, we consider vehicles moving in agile manners for rapid formation motion and stability in high winds. In many previous works, a back-stepping approach and linearized controllers are used for both position and attitude loops [15], [16]. We employ a nonlinear controller based on [17], [18] to model and control for large excursions from hover. The control inputs are the thrust τ and moments $\mathbf{M}$ and are calculated as

$$\begin{aligned} \boldsymbol{M} = & -k_R \boldsymbol{e}_R - k_\Omega \boldsymbol{e}_\Omega + \boldsymbol{\Omega} \times J\boldsymbol{\Omega} \\ & - \boldsymbol{J}\left(\hat{\boldsymbol{\Omega}}\boldsymbol{R}^\top \boldsymbol{R}_C \boldsymbol{\Omega}_C - \boldsymbol{R}^\top \boldsymbol{R}_C \dot{\boldsymbol{\Omega}}_C\right), \\ \tau = & \left(-k_x \boldsymbol{e}_x - k_v \boldsymbol{e}_v + mg\boldsymbol{e}_3 + m\ddot{\boldsymbol{x}}_d\right) \cdot \boldsymbol{R}\boldsymbol{e}_3, \end{aligned} \tag{1}$$

with $\ddot{\boldsymbol{x}}_d$ the desired acceleration, k_x, k_v, k_R, k_Ω positive definite terms. The subscript C denotes a commanded value, $\boldsymbol{x} \in \mathbb{R}^3$ is the Cartesian position of the vehicle expressed in the inertial frame, $m \in \mathbb{R}$ is the mass, $\boldsymbol{\Omega} \in \mathbb{R}^3$ is the angular velocity and $\boldsymbol{J} \in \boldsymbol{R}^{3\times 3}$ is the inertia matrix both with respect to the body frame. The hat symbol $\hat{\cdot}$ denotes the skew-symmetry operator according to $\hat{\boldsymbol{x}}\boldsymbol{y} = \boldsymbol{x} \times \boldsymbol{y}$ for all $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^3$, g is the standard gravitational acceleration. The quantities $\boldsymbol{e}_R, \boldsymbol{e}_\Omega, \boldsymbol{e}_x, \boldsymbol{e}_v$ are the orientation, angular rate, and translation errors respectively, defined in [17]–[19].

C. Estimation

To maintain the autonomy and scalability of the swarm, each robot estimates its 6DOF pose onboard using dead reckoning style monocular Visual Inertial Odometry (VIO) from the downward facing camera and IMU. Since their estimation only tracks relative to their start locations, the robots must start at known offsets. As described in our recent work [13], we use two nested filters to achieve high quality tracking. An EKF combining visual and IMU data provides initial pose estimates. The prediction step is based on IMU integration, while the update step relies on the residual between visible landmarks in the environment and the one by obtained reprojecting the corresponding 3D point stored into the filter's state. A second filter, a UKF, is able to estimate the full state of each vehicle at 500 Hz. We use an UKF, instead of an EKF because of the need to operate over a large operating envelope with significant excursions in attitude and position. The second filter has a more compact state since we do not incorporate all the landmarks in the state, the state vector for the UKF is represented by

$$\mathbf{x}_f = \begin{bmatrix} \boldsymbol{x}^\top & \boldsymbol{v}^\top & \Phi^\top & \boldsymbol{b}_a^\top \end{bmatrix}^\top, \tag{2}$$

where $\boldsymbol{x}$, $\boldsymbol{v}$ are the position and velocity of the quadrotor, Φ is the attitude quaternion and $\mathbf{b}_a$ the accelerometer biases. The prediction step uses as inputs the linear acceleration and angular velocity measurements given by the IMU. The VIO pose estimates, are then used to update the state estimate using a linear measurement model. This estimation method produces fast and computationally tractable pose estimates. However it is prone to errors in linear scale as well as drift over time. The effect of these errors was quantified for our robotic platform in Section V-A.

III. System Architecture

A. Individual Robot

Each robot runs its own ROS[3] network to tie together a high level interface with low level estimation and control. The robots respond to ROS services to perform basic actions such as takeoff and land as well as plan and execute dynamically feasible trajectories as described in Section IV. While the onboard estimation and control loops ran at 500 Hz to ensure stable flight, the ROS network includes a throttling node to broadcast odometry updates at 10 Hz on the shared network. This throttled odometry is fast enough for the ground station to perform coordination calculations and slow enough to not jam communication channels.

B. Ground Station

While the majority of computation is performed onboard, a Ground Station computer was used for user interface and

[3]http://www.ros.org

multi-robot coordination. The ground station was a laptop running Linux 16.04 connected via Ethernet to a Ubiquiti Nanostation[4] Router. The router broadcasts a 5 GHz WiFi network which the robots connect to. The Chrony NTP Suite[5] was used on robot startup to synchronize all clocks with the ground station. Multi-Robot coordination was performed via a ROS network, where the ground station computer ran the following ROS nodes:

- Master Discovery - Detected the ROS masters running on all of the robots.
- Master Sync Fkie - Synchronized throttled robot odometry messages and position commands between the robots and the ground station.
- Motion Capture - Recorded odometry information from external motion capture system for formation error post processing.
- Multi Mav Manager - Interfaced with the ROS network of each robot. The user interfaced with this node via ROS services to perform formation actions such as takeoff and land, as well as command formation shapes. These formation requests include a form factor, such as circle or rectangle, inter-robot spacing, centroid location, and rotation angles. Upon request, the node computed individual goal locations for the robots and performed the multi-robot trajectory assignment as described in Section IV.

IV. Trajectory Planning

The goal of trajectory planning is to generate trajectories in 3D space to safely guide robots to a set of goal locations, while avoiding inter-robot collisions. The Centralized Concurrent Assignment and Planning of Trajectories Algorithm (C-CAPT)[3] was chosen for this task because it provides a complete and scalable solution to the problem. Using C-CAPT, the ground station calculated assignments of goals to robots by minimizing the sum of the distances traveled squared using the Hungarian Algorithm. This computation has a time complexity of $O(n^3)$ and is performed once per formation change. C-CAPT guarantees that robots remain a minimum distance of D apart if initial and goal locations are spaced at least $\Delta = \sqrt{2}D$ apart. In our experiments D was chosen to be large enough to account for the radius of the robot as well as estimation errors. We chose to keep this distance fixed for simplicity, but future works could increase this distance over time to account for drift.

In this algorithm, robot trajectories must be synchronized, so the start time of was offset to account for communication and computation time, and the duration was determined by a trapezoidal approximation heuristic for a minimum jerk trajectory. All of the trajectories utilized the shared duration of the longest individual trajectory. The ground station then transformed the goal locations from the global frame into the frame of the robots, as offset by their starting locations. Then, the ground station sent each robot its assigned goal, trajectory start time, and duration as a ROS service.

The robots then generated trajectories on-board to travel to their assigned goal at the required time. For C-CAPT, trajectories are required to be identically time parameterized straight lines. Previous work has shown that the angular velocity of a quadrotor is directly related to the jerk (third order derivative of position) of the vehicle. Since vision based systems are generally sensitive to fast rotations. minimum jerk trajectories were chosen to maintain high quality odometry tracking during flight. Since the robots generate and track their trajectories on-board, the system doesn't depend heavily on WiFi communication to execute formation changes.

[4]https://www.ubnt.com/airmax/nanostationm/

[5]https://chrony.tuxfamily.org/index.html

TABLE II
Single Robot Performance

Maximum Velocity	Total Distance Travelled	Drift Percentage	Maximum Odometry Error
4.5 m/s	124.47 m	0.37%	0.60 m

V. Experimental Results

In order to validate the robustness and scalability of the system, three experiments were performed. All experiments took place at the PERCH lab (Penn Engineering Research Collaboration Hub) in the Pennovation Center at the University of Pennsylvania.

A. Estimation Error of a Single Robot

The first experiment sought to quantify the estimation performance of a single robot. In this experiment, one robot flew a 10 m straight line trajectory back and forth six times reaching a maximum velocity of 4.5 m/s. This experiment was conducted in the indoor motion capture space at the PERCH lab. The motion capture flying area measures 20 m $\times$ 6 m $\times$ 4 m and uses 20 ViconVantage V16 cameras to track models at 100 Hz. Table II shows the odometry performance of the robot in this trial. Drift percentage was calculated as the odometry error at the end of the trial divided by the total distance traveled by the robot. This drift was very small, showing that the position estimation functioned well.

Fig. 4 shows the magnitude of the estimated position error as a function of total distance traveled. We note that the odometry error stems from 2 main factors. First, the odometry error generally increases with total distance traveled due to the inherent drift associated with dead reckoning style VIO. Second, the fluctuations in the error around this upward trend correspond with the distance from the origin. This is due to scale and yaw errors causing larger error magnitudes further from the origin. While more complicated position estimators could be implemented with techniques such as loop closure and global map merging, the robot position estimate error remained within 0.6 m over long distances. This deviation was accounted for in the multi-robot experiments.

B. Indoor Swarm

The first trial including multiple robots was performed in the same indoor PERCH flying area as the single robot experiments. Six robots were used due to space constraints of

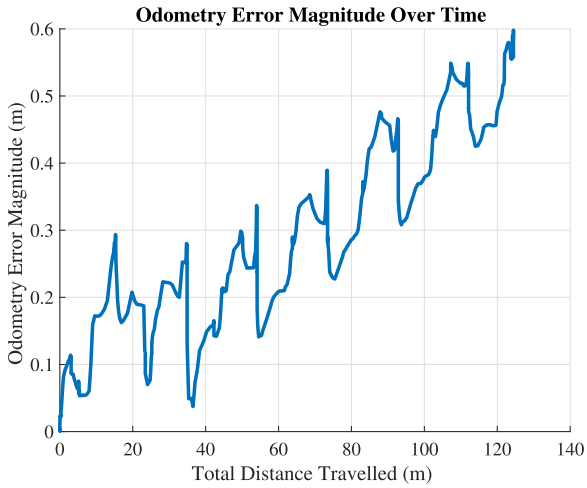


Fig. 4. Estimation error of a single robot traveling back and forth in a 10 m straight line trajectory.

Fig. 5. Robot on takeoff plate used to initialize spatial location prior to takeoff.

the flying area. Considering the maximum estimation error from the single robot experiment as well as the robot radius, a nominal inter-robot spacing of 1.5 m was chosen to avoid collisions caused by estimation error. The starting location of the robots was initialized by installing takeoff plates on the ground which ensure that the robots takeoff from in the same location between trials, as shown in Fig. 5. These plates were designed to precisely locate the feet of the robot without interfering with takeoff. Precise positioning of the launch plates served to minimize the effect of initial yaw and offset errors throughout the formation flight. In addition, chalk and colored tape was applied to the floor to provide ample features for VIO tracking. The formation flight consisted of transitions between a set of formations which included changes of shape and bulk motion. The set of formation commands used during indoor trials was:

- Takeoff
- Form a Rectangle at (0, 0, 1.5) m, spaced by 1.5 m
- Form a Circle at (0, 0, 1) m, spaced by 1.5 m, pitched 0.1 rad
- Form a Line at (−2, 0, 1) m, spaced by 1.5 m
- Form a Rectangle at (0, 0, 1) m, spaced by 1.5 m, yawed $\frac{\pi}{2}$ rad
- Form a Circle at (−2, 0, 1) m, spaced by 1.5 m
- Form a Rectangle at (0, 0, 1) m, spaced by 1.5 m
- Land

TABLE III
SWARM TESTING RESULTS

Trial	Maximum Velocity	Mean Distance Travelled	Mean Formation Error	Std. Deviation Error
Indoor Slow	2.25 m/s	13.54 m	0.09 m	0.05 m
Outdoor Slow	3.74 m/s	28.00 m	0.32 m	0.11 m
Outdoor Medium	4.68 m/s	27.95 m	0.29 m	0.13 m
Outdoor Fast	5.37 m/s	26.74 m	0.28 m	0.15 m

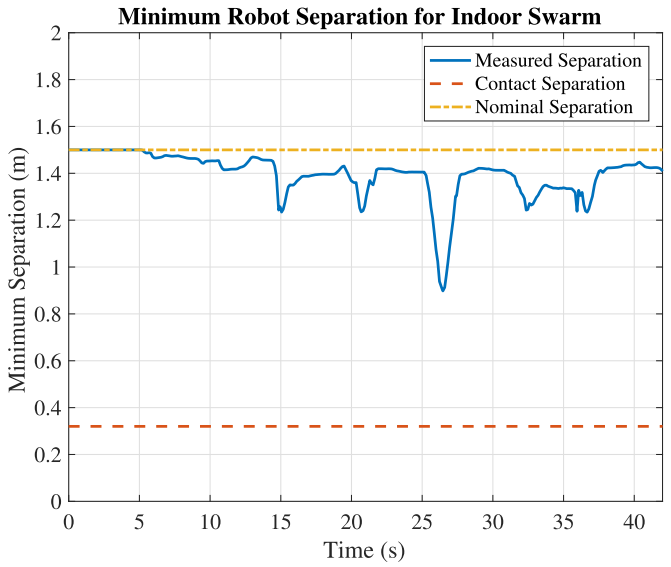
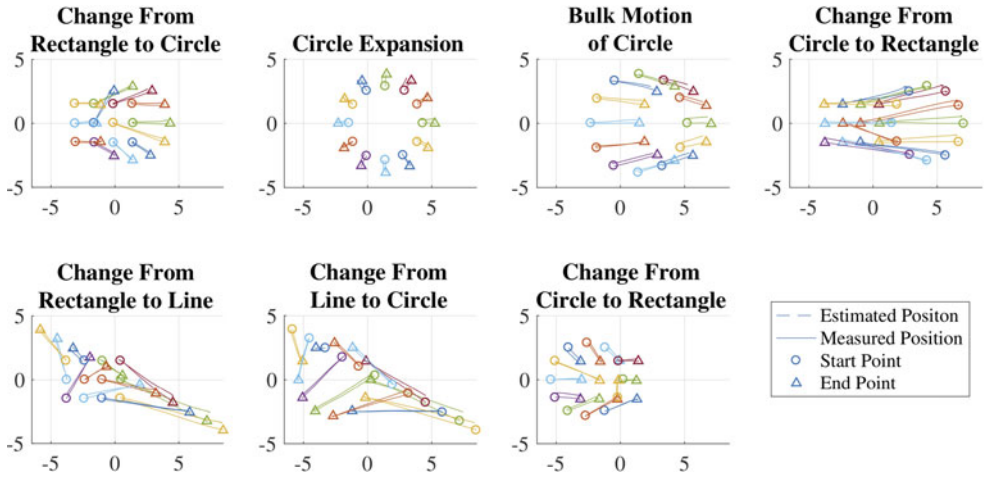


Fig. 6. Minimum inter-robot separation in 6 robot indoor trial. Contact separation is the distance at which the robots collide. Nominal Separation is the commanded separation for formations, however during formation changes the commanded trajectories get closer than this.

Fig. 7. Perch Outdoor flying space.

The performance of the swarm is detailed in Table III. The Formation Error is calculated as the distance between each robot and its desired location as measured by motion capture. Fig. 6 shows the minimum distance between the robots during the trial. This trial occurred at slower speeds and covered shorter distances than the single robot experiments, causing the



(a)

Formation	Circle	Expanded Circle	Moved Circle	Rectangle	Line	Circle	Rectangle
Mean Error	0.36 m	0.35 m	0.53 m	0.29 m	0.40 m	0.30 m	0.30 m
Std. Deviation	0.10 m	0.12 m	0.14 m	0.08 m	0.21 m	0.09 m	0.10 m

(b)

Fig. 8. Swarm performance during the Slow Outdoor Trial. (a) Desired and actual positions of the swarm during formation transitions. (b) Evaluation of position errors at transition endpoints.

odometry errors to remain small. Additionally, the drift percentage was the similar to the single robot trial, confirming that the earlier trial was indicative of robot performance in a swarm. Footage from the indoor trial is included in the attached video.

C. Outdoor Swarm

In order to accommodate the full 12 robots of our swarm, the next set of trials were performed at PERCH outdoor flying space, shown in Fig. 7. The outdoor motion capture flying area measures 30 m $\times$ 15 m $\times$ 15 m and uses 34 Qualisys Oqus cameras to track models at 100 Hz. The swarm performed similar formation changes as in the indoor trials, however the experiments were repeated at three different speeds. The set of formation commands used during outdoor trials was:

- Takeoff
- Form a Circle at (2.25, 0, 1) m, spaced by 1.5 m
- Form a Circle at (2.25, 0, 1) m, spaced by 2.0 m
- Form a Circle at (5.25, 0, 2.5) m, spaced by 1.5 m, pitched 0.3 rad
- Form a Rectangle at ($-$1.25, 0, 1.5) m, spaced by 1.5 m, pitched $-$0.3 rad
- Form a Line at (2.25, 0, 2) m, spaced by 1.5 m, pitched 0.1 rad and yawed 0.5
- Form a Circle at ($-$1.75, 0, 1) m, spaced by 1.5 m
- Form a Rectangle at (0, 0, 1) m, spaced by 1.5 m
- Land

Formation results are shown in Fig. 8. As detailed in Table III, the performance of the swarm slightly decreased at higher speeds. Additionally, estimation errors were higher outdoors than indoors. It was not possible to achieve the same level of launch plate precision outdoors as indoors because the ground outside was neither flat nor level. Odometry errors were higher outdoors primarily due to the imprecision of these takeoff locations, as well as lighting differences across the utilized flying space. Despite these inaccuracies, the robots were able to successfully reach all formations without colliding. Flying outdoors also validated the robustness of the swarm to high winds and varied lighting conditions. During testing, the average wind speed was 10 mph with gusts reaching up to 18 mph. Experiments were performed at various times of day and included a variety of lighting conditions such as direct sunlight, cloud coverage, and even low light after sunset as pictured in Fig. 9.

Fig. 9. Swarm flying in low light conditions.

D. Scalability of the Architecture

Finally, we would like to analyze the extension of our approach to larger teams. The proposed architecture is semi-distributed, allowing users to easily increase the number of robots since the main components such as estimation, control

and planning pipeline, are the same on each robot. The key challenges for larger teams are threefold. First, the assignment problem will take a longer time to be executed. Second, to fly more vehicles, due to space constraints, it is necessary to reduce the vehicles size. Third, the archtecture needs to take into account the drift of each platform. The second challenge is a design problem and will be investigated in the future work. Regarding the first challenge, the assignment has cubic complexity and it can be solved in less than half second for hundreds of vehicles. Since this challenge does not affect the main estimation and control components, the architecture easily allows to increase the number of elements in the formation. The third challenge will require the ability to add multi-robot loop closure algorithms, when similar areas are visited by the platforms and perform pose graph optimization. The proposed architecture allows anyone interested to this additional feature to add it to the existing framework. However, this feature will require more frequent and larger amounts of data communication between the platforms and the ground station. Specifically, the descriptors from each vehicles should be sent to the ground station, which sends the corrections back to the interested vehicles once the loop closures constraints are identified. This task opens up new research opportunities related to limited communication bandwidth and decentralized approaches to solve the problem using limited inter-vehicle communication instead of a centralized ground station.

VI. CONCLUSION

We have demonstrated what we believe at the time of publication is the largest swarm of quadrotors, which does not rely on motion capture or GPS information. Our experiments show that the vehicle and system framework is robust in a variety of indoor and outdoor scenarios making it deployable nearly anywhere. Additionally, the framework minimally relies on wireless communication and is scalable to larger swarms of robots. The vehicles are constructed from off the shelf components and all software is released online. Future works will focus on increasing the size of the swarm, exhibiting more complex formation behaviors, and implementing loop closure detection and shared global feature mapping.

REFERENCES

[1] A. Kushleyev, D. Mellinger, C. Powers, and V. Kumar, "Towards a swarm of agile micro quadrotors," *Auton. Robots*, vol. 35, no. 4, pp. 287–300, Nov. 2013.

[2] J. A. Preiss, W. Hönig, G. S. Sukhatme, and N. Ayanian, "Crazyswarm: A large nano-quadcopter swarm," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2017, pp. 3299–3304.

[3] M. Turpin, N. Michael, and V. Kumar, "Capt: Concurrent assignment and planning of trajectories for multiple robots," *Int. J. Robot. Res.*, vol. 33, no. 1, pp. 98–112, Jun. 2014.

[4] K. Mohta, M. Turpin, A. Kushleyev, D. Mellinger, N. Michael, and V. Kumar, *QuadCloud: A Rapid Response Force with Quadrotor Teams*. Cham, Switzerland: Springer International Publishing, 2016, pp. 577–590.

[5] M. Afanasov, L. Mottola, and K. Whitehouse, "Poster: Testbed for aerial drone applications," in *Proceedings of the 2017 International Conference on Embedded Wireless Systems and Networks* (EWSN ’17). Uppsala, Sweden, USA: Junction Publishing, 2017, pp. 192–193.

[6] J. L. Sanchez-Lopez, J. Pestana, P. de la Puente, A. Carrio, and P. Campoy, *Visual Quadrotor Swarm for the IMAV 2013 Indoor Competition*. Cham, Switzerland: Springer International Publishing, 2014, pp. 55–63.

[7] J. L. Sanchez-Lopez, J. Pestana, P. de la Puente, and P. Campoy, "A reliable open-source system architecture for the fast designing and prototyping of autonomous multi-UAV systems: Simulation and experimentation," *J. IntellRobot. Syst.*, vol. 84, no. 1, pp. 779–797, Dec. 2016.

[8] J. Faigl, T. Krajnk, 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.

[9] P. Lightbody, T. Krajník, and M. Hanheide, "A versatile high-performance visual fiducial marker detection system with scalable identity encoding," in *Proc. Symp. Appl. Comput.* (SAC '17). New York, NY, USA: ACM, 2017, pp. 276–282.

[10] C. Forster, S. Lynen, L. Kneip, and D. Scaramuzza, "Collaborative monocular SLAM with multiple micro aerial vehicles," in *Proc. IEEE/RSJ Conf. Intll. Robots Syst.*, Tokyo, Japan, 2013, pp. 3962–3970.

[11] G. Loianno, J. Thomas, and V. Kumar, "Cooperative localization and mapping of MAVs using RGB-D sensors," in *IEEE Int. Conf. Robot. Autom.*, May 2015, pp. 4021–4028.

[12] G. Loianno *et al.*, "A swarm of flying smartphones," in *Proc. 2016 IEEE/RSJ Int. Conf., Intell. Robots Syst..*, 2016, pp. 1681–1688.

[13] G. Loianno, C. Brunner, G. McGrath, and V. Kumar, "Estimation, control, and planning for aggressive flight with a small quadrotor with a single camera and IMU," *IEEE Robot. Autom. Lett.*, vol. 2, no. 2, pp. 404–411, Apr. 2017.

[14] J. Thomas, J. Welde, G. Loianno, K. Daniilidis, and V. Kumar, "Autonomous flight for detection, localization, and tracking of moving targets with a small quadrotor," *IEEE Robot. Autom. Lett.*, vol. 2, no. 3, pp. 1762–1769, Jul. 2017.

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

[16] S. Weiss, D. Scaramuzza, and R. Siegwart, "Monocular-SLAM-based navigation for autonomous micro helicopters in GPS denied environments," *J. Field Robot.*, vol. 28, no. 6, pp. 854–874, 2011.

[17] T. Lee, M. Leok, and N. H. McClamroch, "Nonlinear robust tracking control of a quadrotor UAV on SE(3)," *Asian J. Control*, vol. 15, no. 2, pp. 391–408, 2013.

[18] D. Mellinger and V. Kumar, "Minimum snap trajectory generation and control for quadrotors," in *Proc. IEEE Int. Conf. Robot. Autom.*, Shangai, China, 2011, pp. 2520–2525.

[19] G. Loianno *et al.*, "Smartphones power flying robots," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Hamburg, Germany, Sep. 2015, pp. 1256–1263.