Vision-Based State Estimation and Trajectory Control Towards High-Speed Flight with a Quadrotor

Shaojie Shen
GRASP Lab
University of Pennsylvania
shaojie@seas.upenn.edu

Yash Mulgaonkar
GRASP Lab
University of Pennsylvania
yashm@seas.upenn.edu

Nathan Michael
The Robotics Institute
Carnegie Mellon University
nmichael@cmu.edu

Vijay Kumar
GRASP Lab
University of Pennsylvania
kumar@seas.upenn.edu

***Abstract*—This paper addresses the development of a lightweight autonomous quadrotor that uses cameras and an inexpensive IMU as its only sensors and onboard processors for estimation and control. We describe a fully-functional, integrated system with a focus on robust visual-inertial state estimation, and demonstrate the quadrotor's ability to autonomously travel at speeds up to 4 m/s and roll and pitch angles exceeding 20°. The performance of the proposed system is demonstrated via challenging experiments in three dimensional indoor environments.**

I. Introduction

Aerial robots have great potential for applications in search and rescue and first response. They can, in principle, navigate quickly through 3-D unstructured environments, enter and exit buildings through windows, and fly through collapsed buildings. However, it has proved to be challenging to develop small (less than 1 meter characteristic length, less than 1 kg mass) aerial robots that can navigate autonomously without GPS. In this work, we take a significant step in this direction by developing a quadrotor that uses a pair of cameras and an IMU for sensing and a netbook class processor for state estimation and control. The robot weights less than 750 grams and is able to reach speeds of over 10 body lengths/second. The paper describes the design of the system and the algorithms for estimation and control, and provides experimental results that demonstrate the performance of the system.

The literature on autonomous flight in GPS-denied environments is extensive. Laser-based autonomous flight approaches for micro-aerial vehicles (MAVs) frequently require a partially-structured environment to enable incremental motion calculations [1, 18] or mechanized panning laser-scanners that add considerable payload mass [10]. Vision-based approaches (monocular-, stereo-, and RGB-D camera-based) enable full 6-DOF state estimation but operate at limited update rates due to computational complexity and limited onboard processing [5, 6, 23]. We are interested in pursuing high-speed flight and therefore require accurate 6-DOF state estimation with low latency in general, unstructured, and unknown environments. Previous work toward this goal includes a laser-based approach for state estimation toward high-speed flight in general known 3-D environments [2].

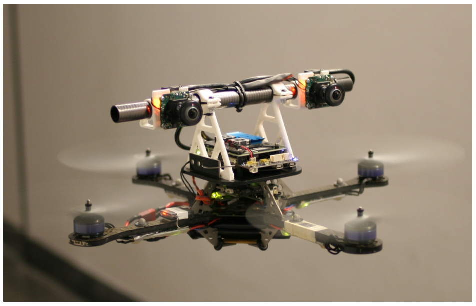


Fig. 1. The experimental platform with limited onboard computation (Intel Atom 1.6 GHz processor) and sensing (two cameras with fisheye lenses and an off-the-shelf inexpensive IMU). The platform mass is 740 g.

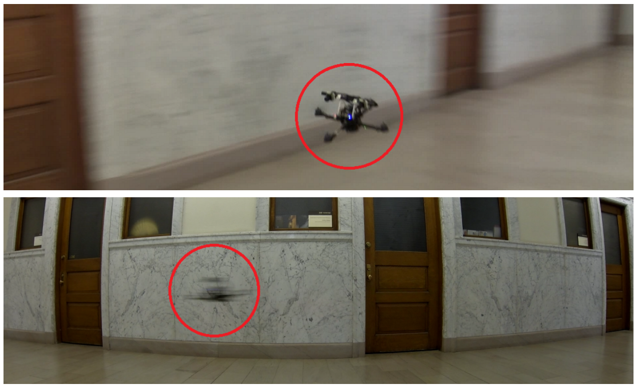


Fig. 2. Snapshots from two vantage points of the quadrotor autonomously tracking a line trajectory at 4 m/s. We highlight the position of the robot with a red circle. Videos of the experiments are available at: http://youtu.be/erTk71643Ro.

The contributions of this work are twofold. First, we develop a vision-inertial (VINS) state estimator that is able to handle high-speed motion with linear velocities up to 4 m/s. The proposed state estimator adaptively fuses the information from monocular and stereo camera subsystems in order to avoid a drift in scale while requiring a limited computational overhead. The estimator runs onboard a 1.6 GHz Intel Atom processor with 20 Hz vision processing and provides 100 Hz state es-

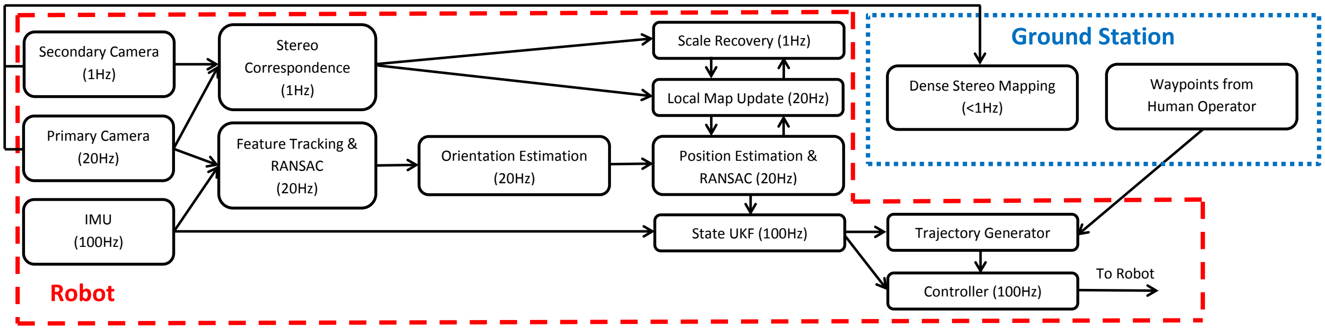

Secondary Camera (1Hz)
Primary Camera (20Hz)
IMU (100Hz)
Stereo Correspondence (1Hz)
Feature Tracking & RANSAC (20Hz)
Orientation Estimation (20Hz)
Scale Recovery (1Hz)
Local Map Update (20Hz)
Position Estimation & RANSAC (20Hz)
State UKF (100Hz)
Trajectory Generator
Controller (100Hz)
To Robot
Robot
Ground Station
Dense Stereo Mapping (<1Hz)
Waypoints from Human Operator

Fig. 3. System architecture.

timation after fusion with the IMU. The resulting integrated system enables autonomous high-speed maneuvers in GPS-denied environments using a quadrotor that weighs only 740 g (Fig. 1). Second, we pursue experimentation by integrating our VINS estimator with a nonlinear tracking controller [11] and by generating smooth polynomial trajectories that minimize the image jitter caused by excessive angular velocities. We study the performance of the system through rigorous experiments in multiple environments with maximum vehicle speeds of 4 m/s (Fig. 2). The architecture diagram in Fig. 3 depicts the integration of the perception, state estimation, trajectory design, and control modules onboard as well as the offboard operator interface.

II. Visual-Inertial State Estimation

The key component technology in our system is a robust visual-inertial state estimator that accurately tracks the pose and velocity of the quadrotor in 3-D environments. The problem of monocular VINS state estimation is well studied in the literature [7, 8, 9]. A nonlinear observability analysis of the estimation problem shows the presence of unobservable modes that can only be eliminated through motions that involve non-zero linear accelerations [7, 8]. Thus it may be difficult to directly use state-of-the-art VINS systems such as the ones described in [9] on hover-capable platforms such as quadrotors.

In [23], an optical flow-based velocity estimator, in conjunction with a loosely coupled filtering framework, successfully enables autonomous quadrotor flight via a downward-facing camera. However, this approach assumes a slowly-varying visual scale, which can be difficult to enforce during fast motions at low-altitudes with potentially rapid changes in the observed environment and large variations in scene depth (height). A downward-facing camera also severely limits the application of vision-based obstacle detection for planning and control purposes.

Stereo vision-based state estimation approaches for autonomous MAVs such as those proposed in [5, 6] do not suffer from the problem of scale drift as seen in monocular systems or limit the observable camera motion. However, we found that the overhead to compute state estimates using these methods exceeds the limited onboard computation budget at the frame-rates required to enable high-speed operation.

Based on the above evaluation we choose to equip our quadrotor platform with two forward-facing fish-eye cameras and develop a loosely-coupled, combined monocular-stereo approach. A primary forward facing fisheye camera operates at a high rate and supports pose estimation and local mapping, while a secondary camera operates at a low-rate and compensates for the limitations of monocular vision-based approaches. The pose estimate derived from the VINS is fused with IMU information to enable feedback control. Note that we do not address the full vision-based SLAM problem [4, 21] due to computational constraints. We require that visual pose estimation and map update be done at frame rate (20 Hz) in order to maximize robustness to rapid changes in observable features during fast maneuvers. The proposed VINS estimator builds upon our earlier work [19] with the following improvements: 1) an orientation estimation approach to reduce drifting; 2) online scale recovery using low-rate stereo measurements; and 3) system optimizations that enable onboard processing with a limited computation budget.

A. Feature Detection, Tracking, and Outlier Rejection

Both cameras in the system are modeled as spherical cameras and calibrated using the Omnidirection Calibration Toolbox [17]. For the primary camera that runs at 20 Hz, we detect Shi-Tomasi corners [20] and track them using the KLT tracker [12]. Due to the limited motion between image frames, We are able to perform the feature detection and tracking calculations on the distorted fisheye camera image, reducing the overall computational burden. All features are transformed into unit length feature observation vectors $\mathbf{u}_{ij}$ using calibration parameters. Here we denote $\mathbf{u}_{ij}$ as an observation of the i^{th} feature in the j^{th} image in the camera body frame.

Following the method in [23], we remove tracking outliers by using the the estimated rotation (from short term integration of gyroscopic measurements) between two consecutive frames and unrotate the feature observation prior to applying the epipolar constraint in the unrotated frame:

$$(\mathbf{u}_{ij-1} \times \Delta R \mathbf{u}_{ij})T = 0$$

where ΔR is the rotation between two consecutive images estimated by integrating gyroscope measurements, and T is the translation vector with unknown scale. Only two correspondences are required to solve an arbitrary scaled T, thus a

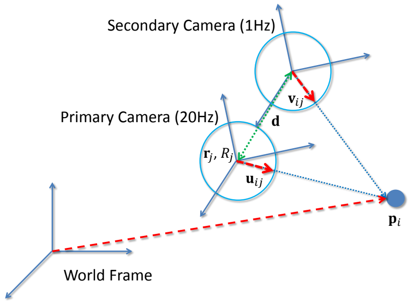


Fig. 4. Camera geometry notation. $\mathbf{r}_j$ and R_j represent the j^{th} primary camera pose in the world frame and $\mathbf{p}_i$ is the position vector of the i^{th} feature in the world frame. $\mathbf{u}_{ij}$ and $\mathbf{v}_{ij}$ are unit length feature vectors in the body frame of the primary and the secondary cameras, respectively. $\mathbf{d}$ is the baseline line of the calibrated stereo cameras.

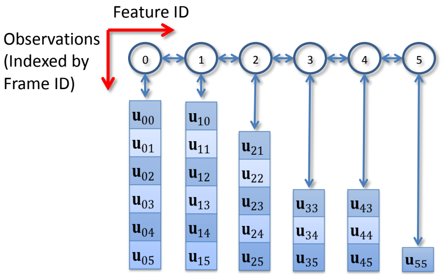


Fig. 5. Data structure for feature storage. Features are managed in a linked list and newly added features are added to the end of the list. For every feature, all observations are recorded in pre-allocated memory. Feature deletion, addition, as well as the lookup of observations of a given feature can be performed in constant time.

2-point RANSAC can be used to reject outliers. This approach reaches a consensus with much fewer hypotheses compared to the traditional 5-point algorithm [15].

For the low rate stereo subsystem, candidate correspondences can be found using the KLT tracker. With calibrated stereo cameras, outlier rejection of these candidates is possible via applying epipolar geometry constraint. We denote the observation of the i^{th} feature by the secondary camera as $\mathbf{v}_{ij}$. and define the inter-camera baseline as $\mathbf{d}$.

B. Pose Estimation

In general, and especially for a monocular system, the number of features with good 3-D position estimates is much smaller than the number of tracked features. Even in a stereo setting, a large number of features cannot be triangulated due to scene ambiguity. Although these "low quality" features cannot be used for position estimation, they do carry information about the orientation. Therefore, similar to [3], we decouple the orientation and position estimation subproblems.

1) Orientation Estimation: Orientation estimation is traditionally computed via the essential matrix between two consecutive images and compounding incremental rotation [22]. However, we wish to minimize rotation drift, especially for the case of hovering when the same set of features can be observed over an extended period of time. We store the index of each frame k in which feature i is observed in the set $\mathcal{J}_i \subset \mathbb{Z}_{\geq 0}$ and record its observation, $\mathbf{u}_{ik}$, and the corresponding camera orientation R_k. M_i denotes the frame of the index of the first observation of i and j is the current frame index. Note that M_i may be different for each feature. We maintain all features in an ascending order according to M_i (Fig. 5).

We pick all features that have at least T_j observations for orientation estimation. T_j is determined by:

$$T_j = T_{j-1} + 1 - D_j$$

where the integer $D_j \in [0, T_{j-1}]$ is the minimum number of observation reduction that makes the estimated essential matrix well-posed. In other words, we require the singular values of the essential matrix to be close to $[\sqrt{2}, \sqrt{2}, 0]$. D_j is found in a brute force manner. However, if the robot is hovering, D_j is likely to be zero as there are no large changes in the distribution of feature observations. On the other hand, fast motions can result in $D_j = T_{j-1}$ and only consecutive frames can be used for orientation estimation due to rapid changes in the feature distribution. As T_{j-1} is likely to be one in this case, the computation overhead of this brute force search is limited.

We denote the index of the last feature that has at least T_j observations as n. The image index M_n and its corresponding camera orientation R_{M_n} are used as a reference, via the 8-point algorithm [13], to estimate the essential matrix $E_{M_n,j}$ and then the rotation $R_{M_n,j}$ between the M_n^{th} image and the current image j. Therefore the current orientation can be written as:

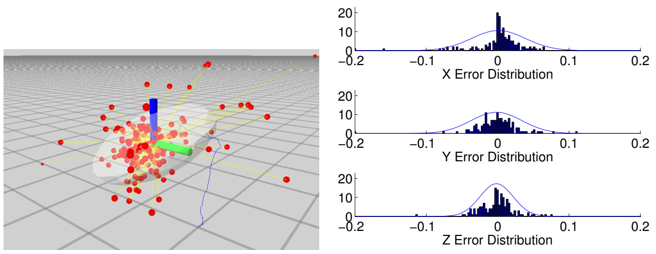
$$R_j = R_{M_n} R_{M_n,j}.$$

We require that the onboard attitude estimate be aligned with the inertial frame and therefore employ a common IMU design strategy where drift in the vision-based attitude estimate (roll and pitch) is eliminated via fusion with accelerometer measurements. This approach assumes that the vehicle state is near hover or at a constant velocity. However, fast vehicle motions can invalidate these assumptions. In this work, we find that applying small weightings to accelerometer measurements yields a reasonable estimate (Fig. 9(c)) given small drift in the vision-based attitude estimate.

2) Position Estimation: We begin by assuming a known 3-D local feature map and describe the maintenance of this map in the next subsection. Given observations of a local map consisting of known 3-D features, and assuming that this local map is *noiseless*, the 3-D position of the camera can be estimated by minimizing the sum-of-square sine of angle error of the observed features:

$$\mathbf{r}_j = \underset{\mathbf{r}_j}{\operatorname{argmin}} \sum_{i \in \mathcal{I}} \left\| \frac{\mathbf{r}_j - \mathbf{p}_i}{\|\mathbf{r}_j - \mathbf{p}_i\|} \times R_j \mathbf{u}_{ij} \right\|^2 \qquad (1)$$

where, as shown in Fig. 4, $\mathbf{r}_j$ is the 3-D position of primary camera in the world frame when the j^{th} image is captured, $\mathcal{I}$ represents the set of features observed in the j^{th} image, and $\mathbf{p}_i$ is the 3-D position of the i^{th} feature in the world frame.



(a) (b)

Fig. 6. The 3-D distribution of $\mathbf{e}_{ij}$ and the ellipsoid of the best fit Gaussian distribution (Fig. 6(a)). The error distribution histogram for each axis is shown in Fig. 6(b))

R_j, which is the estimated rotation of the primary camera, is treated as a known quantity while doing position estimation. Note that (1) is nonlinear. However, if we assume that the change of feature distance between two consecutive images is small, we can approximate (1) and solve the camera position $\mathbf{r}_j$ via the following linear system:

$$\left(\sum_{i\in\mathcal{I}} \frac{\mathbb{I}_3 - \mathbf{u}^r_{ij}\mathbf{u}^r_{ij}{}^{\mathrm{T}}}{d_i^2}\right)\mathbf{r}_j = \sum_{i\in\mathcal{I}} \frac{\mathbb{I}_3 - \mathbf{u}^r_{ij}\mathbf{u}^r_{ij}{}^{\mathrm{T}}}{d_i^2}\mathbf{p}_i \qquad (2)$$

where $d_i = \|\mathbf{r}_{j-1} - \mathbf{p}_i\|$, $\mathbf{u}^r_{ij} \triangleq R_j\mathbf{u}_{ij}$. Equation (2) always represents three equations in three unknowns, regardless of the number of observed features. Therefore, the position estimation problem can be solved efficiently in constant time.

In our formulation, position estimation is essentially an intersection of multiple rays. We can therefore represent the localization error via the statistical distribution of the ray-to-robot distance. We experimentally verify that this distribution can be be approximate by a 3-D Gaussian distribution (Fig. 6). The covariance for position estimation at the j^{th} frame is obtained as:

$$\Sigma_{\mathbf{r}_j} = \frac{1}{|\mathcal{I}|}\sum_{i\in\mathcal{I}} \mathbf{e}_{ij}\mathbf{e}_{ij}{}^{\mathrm{T}}$$

$$\mathbf{e}_{ij} \triangleq (\mathbf{r}_j - \mathbf{p}_i) \times \mathbf{u}^r_{ij} \times \mathbf{u}^r_{ij}.$$

We apply a second RANSAC to further remove outliers that cannot be removed from the epipolar constraint check (Sect. II-A). A minimum of two feature correspondences are required to solve this linear system. As such, an efficient 2-point RANSAC can be applied for outlier rejection.

C. Local Map Update

As stated in Sect. II-B2, a map consisting of 3-D features is required to estimate the position of the camera. We approach the local mapping problem as an iterative procedure where the pose of the camera ($\mathbf{r}_j$ and R_j) is assumed to be a *noiseless* quantity. We do not perform optimizations for the position of both the camera and the features at the same time (like traditional SLAM approaches) due to CPU limitations.

We define the *local* map as the set of currently tracked features and cull features with lost tracking. New features are introduced into the local map when the current number of tracked features falls below a pre-defined threshold. Given $\mathcal{J}_i$, the set of observations of the i^{th} feature up to the j^{th} frame, we can formulate the 3-D feature location $\mathbf{p}_i$ via triangulation as:

$$\mathbf{p}_i = \underset{\mathbf{p}_i}{\operatorname{argmin}} \sum_{k\in\mathcal{J}_i} \left\|(\mathbf{p}_i - \mathbf{r}_k) \times R_k\mathbf{u}_{ik}\right\|^2 + \mathbf{1}_{\mathcal{J}_i}(\mathbf{v}_{ik})\left\|(\mathbf{p}_i - \mathbf{r}_k - R_k\mathbf{d}) \times R_k\mathbf{v}_{ik}\right\|^2$$

where

$$\mathbf{1}_{\mathcal{J}_i}(\mathbf{v}_{ik}) \triangleq \begin{cases}\mathbf{1} & \mathbf{v}_{ik} \text{ exists} \\ \mathbf{0} & \mathbf{v}_{ik} \text{ does not exist.}\end{cases}$$

Note that $\mathbf{v}_{ik}$ may not be available for every $k \in \mathcal{J}_i$ due to the slower frame rate of the secondary camera. The feature position $\mathbf{p}_i$ up to the j^{th} frame can be solved via the following linear system:

$$A_{ij}\mathbf{p}_i = \mathbf{b}_{ij} \qquad (3)$$

where

$$A_{ij} \triangleq \left(\sum_{k\in\mathcal{J}_i} A^{\mathbf{u}}_{ik} + A^{\mathbf{v}}_{ik}\right)$$

$$\mathbf{b}_{ij} \triangleq \sum_{k\in\mathcal{J}_i} A^{\mathbf{u}}_{ik}\mathbf{r}_k + A^{\mathbf{v}}_{ik}(\mathbf{r}_k + R_k\mathbf{d})$$

$$A^{\mathbf{u}}_{ik} \triangleq \left(\mathbb{I}_3 - \mathbf{u}^r_{ik}\mathbf{u}^r_{ik}{}^{\mathrm{T}}\right)$$

$$A^{\mathbf{v}}_{ik} \triangleq \mathbf{1}_{\mathcal{J}_i}(\mathbf{v}_{ik})\left(\mathbb{I}_3 - \mathbf{v}^r_{ik}\mathbf{v}^r_{ik}{}^{\mathrm{T}}\right).$$

Again, it can be seen that regardless of the number of observations of a specific feature, the dimensionality of (3) is always three. This enables multi-view triangulation with constant computation complexity. Also, this system is memoryless, meaning that for the i^{th} feature up to the j^{th} frame, only A_{ij} and $\mathbf{b}_{ij}$ need to be stored, removing the need of repeated summation of observations. Moreover, the condition number or the ratio between the maximum and minimum eigenvalues of the matrix A_{ij} gives us information about the quality of the estimate of $\mathbf{p}_i$. We evaluate every feature based on the condition number and reject those features with high condition numbers as unsuitable for position estimation.

D. Scale Recovery

One drawback of the above pose estimation approach is the drifting of scale due to accumulated error in the monocular-based triangulation and the low measurement rate from the stereo subsystem. Here we propose a methodology that makes use of the instant stereo measurement to compensate scale drift. Using current observations from the primary and secondary cameras only, we can perform stereo triangulation and obtain a set of 3-D points $\mathbf{p}^s_k$ in the reference frame of the primary camera, where $k \in \mathcal{K}$ is the set of features that gives valid stereo correspondences in the current image. The ratio:

$$\tilde{\gamma} = \frac{1}{|\mathcal{K}|}\sum_{k\in\mathcal{K}} \frac{\|\mathbf{p}_k - \mathbf{r}_j\|}{\|\mathbf{p}^s_k\|} \qquad (4)$$

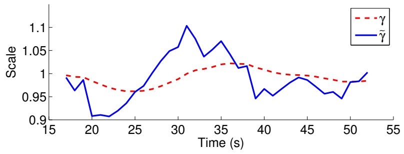


Fig. 7. Scale changes during the flight of a trajectory in Sect. IV-C.

measures the *drift* in scale (from $\gamma = 1$). Scaling all features according to the inverse of this ratio preserves scale consistency. However, as this measurement can be noisy, we apply a complementary filter to estimate the scale drift:

$$\gamma = (1 - \alpha)\gamma + \alpha\tilde{\gamma} \tag{5}$$

where $0 < \alpha \ll 1$. Hence, the proposed approach assumes that the scale *drifts* slowly. This is a major differentiation between our approach and [23], which requires that the scale *changes* slowly. As such, our approach is able to accommodate variations in the visual scene and resulting scale changes that can arise during fast indoor flight with a forward-facing camera.

Figure 7 shows changes in γ and $\tilde{\gamma}$ during the flight of a figure eight pattern (Sect. IV-C). The new position of the feature $\mathbf{p}_i$ can be updated by modifying $\mathbf{b}_{ij}$ as (6) and solve the linear system (3) again.

$$\mathbf{b}_{ij}^{+} = \frac{1}{\gamma}\mathbf{b}_{ij} - \frac{1}{\gamma}A_{ij}\mathbf{r}_j + A_{ij}\mathbf{r}_j \tag{6}$$

E. UKF-Based Sensor Fusion

The 20 Hz pose estimate from the vision system alone is not sufficient to control the robot. We therefore employ a UKF (Unscented Kalman filter) framework with delayed measurement compensation to estimate the pose and velocity of the robot at 100 Hz [14]. The system state is defined as:

$$\mathbf{x} = \left[\mathbf{r}, \dot{\mathbf{r}}, \mathbf{\Phi}, \mathbf{a}_b\right]^{\mathrm{T}}$$

where $\mathbf{\Phi} = [\phi, \theta, \psi]^{\mathrm{T}}$ is the roll, pitch, and yaw Euler angles that represent the 3-D orientation of the robot; and $\mathbf{a}_b = \left[a_{b_x}, a_{b_y}, a_{b_z}\right]^{\mathrm{T}}$ is the bias of the accelerometer measurement in the body frame. We avoid the need to estimate the metric scale in the filter (as in [23]) through the stereo-based scale recovery noted above.

1) Process Model: We consider an IMU-based process model:

$$\mathbf{u} = [\omega, \mathbf{a}]^{\mathrm{T}} = [\omega_x, \omega_y, \omega_z, a_x, a_y, a_z]^{\mathrm{T}}$$
$$\mathbf{v} = [\mathbf{v}_\omega, \mathbf{v}_a, \mathbf{v}_{\mathbf{a}_b}]^{\mathrm{T}}$$
$$\mathbf{x}_{t+1} = f(\mathbf{x}_t, \mathbf{u}_t, \mathbf{v}_t)$$

where $\mathbf{u}$ is the body frame angular velocities and linear accelerations from the IMU. $\mathbf{v}$ represents additive Gaussian noise associated with the gyroscope, accelerometer, and accelerometer bias.

2) Measurement Model: The pose estimate from the vision system is first transformed to the IMU frame before being used for the measurement update. The measurement model is linear and can be written as:

$$\mathbf{z} = H\mathbf{x} + \mathbf{n}$$

where H extracts the 6-DOF pose in the state and $\mathbf{n}$ is additive Gaussian noise. Since the measurement model is linear, the measurement update can be performed via a KF update step.

III. Trajectory Generation and Control

The equation of motion of a quadrotor is given by:

$$m\ddot{\mathbf{r}} = -mg\mathbf{z}_W + f\mathbf{z}_B \tag{7}$$
$$M = J\dot{\Omega} + \Omega \times J\Omega$$

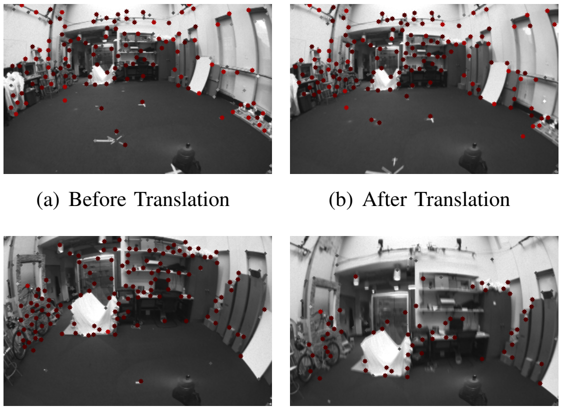
where $\mathbf{z}_W$ $\mathbf{z}_B$ are vertical axes in the world and the body frame, respectively. Ω is the angular velocity in the body frame. J is the inertial matrix. f and M are thrust and moment from all four propellers. We choose to use a nonlinear tracking controller [11] due to its superior performance in highly dynamical motions. The 100 Hz state estimate is used directly as the feedback of the controller.

Given a set of waypoints specified by the human operator, we would like to have the quadrotor smoothly pass through all waypoints as fast as possible, while at the same time maintaining a high quality state estimate. A crucial condition that determines the quality of the vision-based estimate is the tracking performance. With our fisheye cameras setup, it can be seen from Fig 8 that fast translation has little effect on the tracking performance due to the large field of view. However, fast rotation can blur the image easily, causing the failure of the KLT tracker. This observation motivates us to design trajectories that minimize the angular velocities of the platform. By differentiating (7), it can be seen that the angular velocity of the body frame is affinely related to the jerk ($\dddot{\mathbf{r}}$, derivative of the linear acceleration). Therefore, we generate trajectories that minimize the jerk of the quadrotor, and utilize a polynomial trajectory generation algorithm [16] that runs onboard the robot with a runtime on the order of 10 ms for a set of 10 waypoints.

IV. Experimental Results

A. Offboard Mapping

The system presented up to this point is capable of autonomous following of online generated trajectories given user specified waypoints, however, there is no guarantee on global localization consistency, nor obstacle-free trajectories. Although it is not the focus of this work, we wish to provide the human operator an intuitive map for sending waypoints to the robot. To this end, we stream the low-rate stereo images to a ground station, project the stereo pointcloud based on the onboard state estimates, and generate dense voxel grid maps in which the human operator is able to select collision-free goals for the robot. This module does not generate any global corrections for the onboard system.



(a) Before Translation (b) After Translation

(c) Before Rotation (d) After Rotation

Fig. 8. Effects on feature tracking performance due to fast translation (Figs. 8(a)–8(b)) and fast rotation (Figs. 8(c)–8(d)). The number of tracked features significantly reduced after rotation.

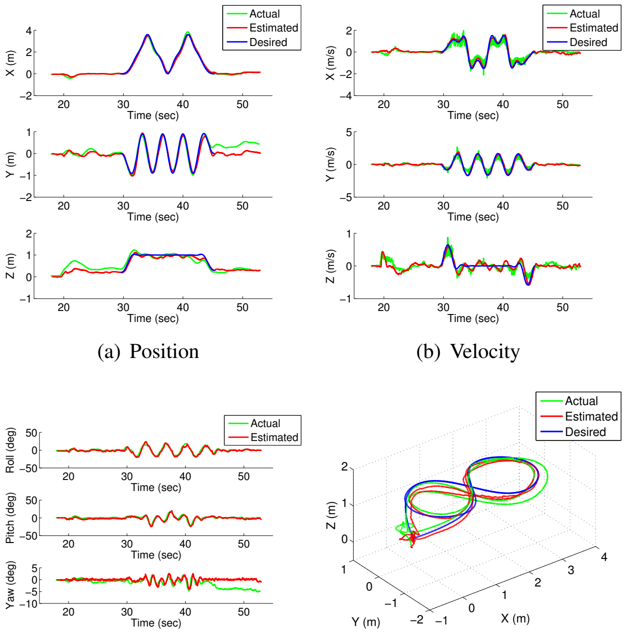
B. *Experiment Design and Implementation Details*

The experimental platform (Fig. 1) is based on the Hummingbird quadrotor from Ascending Technologies[1]This off-the-shelf platform comes with an AutoPilot board that is equipped with an IMU and an user-programmable ARM7 microcontroller. The high level computer onboard includes an Intel Atom 1.6GHz processor and 1GB RAM. The only new additions to this platform are two grayscale mvBlueFOX-MLC200w cameras with fisheye lenses. We use one camera to capture images at 20 Hz as the primary camera. The secondary camera captures images at 1Hz. All camera images are at 376×240 resolution. The synchronization between cameras is ensured via hardware triggering. The total mass of the platform is 740 g. All algorithm development is in C++ using ROS[2] as the interfacing robotics middleware. We utilize the OpenCV library for corner extraction and tracking. The maximum number of features is set to be 300.

The experiment environment includes a laboratory space equipped with a sub-millimeter accurate Vicon motion tracking system[3], a long hallway, and a regular indoor environment. The Vicon system is only used for ground truth. In all experiments, the robot is autonomously controlled using its onboard state estimate. A study of the hover performance of the proposed vision-based state estimator yields similar results to those presented in our earlier work [19]. Three experiments are presented: (1) fast tracking of a figure eight trajectory with ground truth comparison; (2) high speed straight line navigation in a long hallway; and (3) autonomous flight in complex indoor environments.

C. *Autonomous Trajectory Tracking with Ground Truth Comparison*

In this experiment, the robot is programmed to fly through a figure eight pattern in which each circle in the pattern is 0.9 m in radius. The maximum speed of this flight is approximately 2 m/s. Performance is evaluated against the ground truth from Vicon. The estimated, actual, and desired values of the trajectory, position, and velocity are shown in Fig. 9. Large and frequent attitude changes can be seen in Fig. 9(c), as well as in the snapshots (Fig. 10).



(a) Position (b) Velocity

(c) Orientation (d) Trajectory

Fig. 9. The robot is commanded to follow a figure eight pattern at high speed.

Our focus is on generating state estimates that are suitable for high-speed flight, rather than generating globally consistent maps. Therefore, it makes less sense to discuss the drift in absolute position. The onboard velocity estimate, on the other hand, compares well with the Vicon estimates with standard deviation of $\{\sigma_{v_x}, \sigma_{v_y}, \sigma_{v_z}\} = \{0.1105, 0.1261, 0.0947\}$ (m/s). We can also see that the velocity profile matches well with the desired velocity. Note that the Vicon velocity estimate is obtained by a one-step numerical derivative of the position and in fact nosier than the onboard velocity estimate. It is likely that the actual velocity estimation errors are smaller than the values reported above. It should also be pointed out that the tracking error is the result of a combination of the noise of the estimator and the tracking error of the controller.

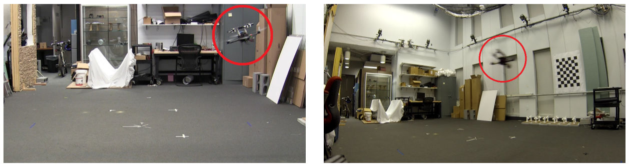
D. *High Speed Straight Line Navigation*

This experiment represents the highest speed that our system is able to handle. The robot is commanded to follow an approximately 15 m long straight line trajectory with a maximum speed of 4 m/s. The estimated and desired trajectory, position, and velocity are shown in Fig. 11. Figure 2 shows snapshots of this flight. It can be seen that the estimated covariance scales with respect to the speed of the robot. Although we do not have ground truth for this experiment, we measure

[1] http://www.asctec.de

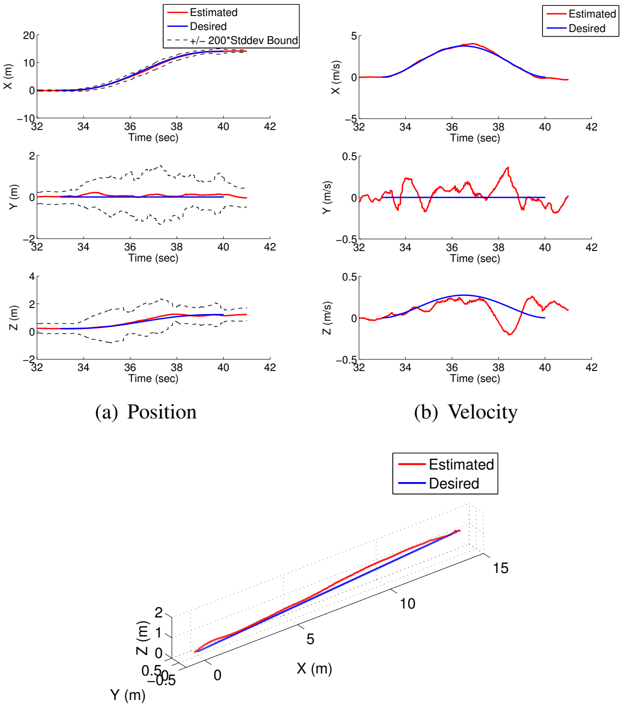
[2] http://www.ros.org

[3] http://www.vicon.com



(a) (b)

Fig. 10. Snapshots taken from different cameras of the quadrotor autonomously tracking a figure eight pattern at 2m/s . Note the large rotation of the robot.

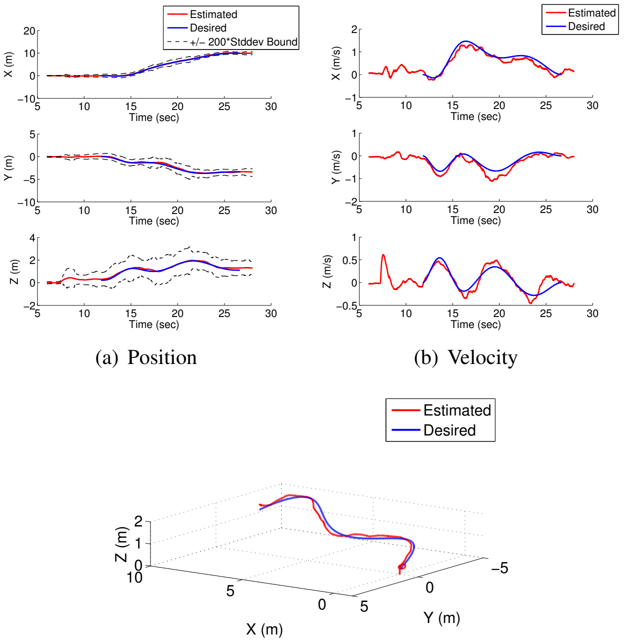


(a) Position (b) Velocity

(c) Trajectory

Fig. 11. The robot is commanded to track a straight line at high speed. A estimated position error standard deviation is presented in Fig. 11(a). Note that we do not have ground truth in this figure. The plot of the estimated covariance (multiplied by 200) shows that the covariance scales with the speed of the robot.

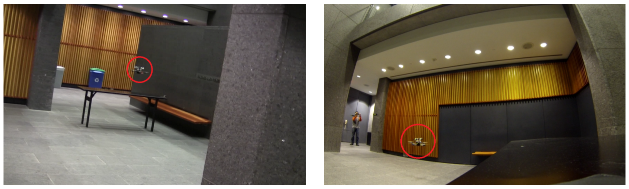
estimator performance by initially placing the robot in the middle of the hallway and visually verifying the drift after the trajectory is completed. The rough measurement of the drift is $\{0.5, 0.1, 0.3\}$ (m) in X, Y, and Z axes, respectively.



(a) Position (b) Velocity

(c) Trajectory

Fig. 12. The robot tracks a trajectory in a complex indoor environment. A scaled estimated position error standard deviation is presented in Fig. 12(a). There is drift in the vertical direction caused by the dominant horizontal texture due to the vertical wood paneling and the absence of any texture in the vertical direction (Fig. 13). This leads to the significantly larger covariance in the Z-direction as seen in the figure.



(a) (b)

Fig. 13. Snapshots taken from different cameras of the quadrotor flying a challenging trajectory in complex environments. Note the repetitive patterns on the wall, causing significant drifting in the vertical direction.

E. Autonomous Flight in Complex Indoor Environments

In this experiment, the robot autonomously flies through a challenging environment with a maximum speed of 1.5 m/s. The robot successfully completes the trajectory and avoids two obstacles (a wall and a table). Note the effect of the very repetitive patterns in the wood paneling on the wall (Fig. 13), as the robot approaches this pattern, the robot estimate drifts significantly in the vertical direction. This is because the wood paneling has limited features to estimate changes in the vertical direction. The fact that we lack features for state estimation is reflected in the increase in the error covariance (Fig. 12(a)).

V. CONCLUSION AND FUTURE WORK

In this paper, we presented a less-than-750 gram, fully autonomous quadrotor and described the algorithms for robust autonomous flight. The main contributions of our work include robust vision-based state estimation with inexpensive IMUs and its integration with a nonlinear controller on an Intel 1.6 GHz Atom processor to enable high-speed flight in 3-D complex environments. The estimator adaptively fuses information from a high frame rate monocular-based estimator and a stereo-based subsystem to provide robust performance even in maneuvers that cause the feature set to change rapidly. We also present experimental results navigating at speeds up to 4 m/s with roll and pitch angles that exceed 20°.

Our future work addresses the development of robust control algorithms that allow the robot to adapt its controller to the uncertainty in the state estimate. Secondly, although our work in this paper was less concerned with global consistency, we plan to expand the functionality of the system by incorporating visual loop closing techniques in order to generate globally consistent environment representations in an online setting.

Acknowledgments

We gratefully acknowledge the support of ONR Grants N00014-07-1-0829 and N00014-09-1-1051, ONR MURI Grant N00014-08-1-0696, and ARL Grant W911NF-08-2-0004.

References

[1] A. Bachrach, S. Prentice, R. He, and N. Roy. RANGE-robust autonomous navigation in gps-denied environments. *J. Field Robotics*, 28(5):644–666, 2011.

[2] A. Bry, A. Bachrach, and N. Roy. State estimation for aggressive flight in gps-denied environments using onboard sensing. In *Proc. of the IEEE Intl. Conf. on Robot. and Autom.*, pages 1–8, Saint Paul, MN, May 2012.

[3] D. Burschka and E. Mair. Direct pose estimation with a monocular camera. In *RobVis*, pages 440–453, 2008.

[4] M. Cummins and P. Newman. Appearance-only SLAM at large scale with FAB-MAP 2.0. *Intl. J. Robot. Research*, 30(9):1100–1123, August 2011.

[5] F. Fraundorfer, L. Heng, D. Honegger, G. H Lee, L. Meier, P. Tanskanen, , and M. Pollefeys. Vision-based autonomous mapping and exploration using a quadrotor MAV. In *Proc. of the IEEE/RSJ Intl. Conf. on Intell. Robots and Syst.*, Vilamoura, Algarve, Portugal, October 2012.

[6] A. S. Huang, A. Bachrach, P. Henry, M. Krainin, D. Maturana, D. Fox, and N. Roy. Visual odometry and mapping for autonomous flight using an RGB-D camera. In *Proc. of the Intl. Sym. of Robot. Research*, Flagstaff, AZ, August 2011.

[7] E. S. Jones and S. Soatto. Visual-inertial navigation, mapping and localization: A scalable real-time causal approach. *Intl. J. Robot. Research*, 30(4):407–430, April 2011.

[8] J. Kelly and G. S. Sukhatme. Visual-inertial sensor fusion: Localization, mapping and sensor-to-sensor self-calibration. *Intl. J. Robot. Research*, 30(1):56–79, January 2011.

[9] D. G. Kottas, J. A. Hesch, S. L. Bowman, and S. I. Roumeliotis. On the consistency of vision-aided inertial navigation. In *Proc. of the Intl. Sym. on Exp. Robot.*, Quebec, Canada, June 2012.

[10] A. Kushleyev, B. MacAllister, and M. Likhachev. Planning for landing site selection in the aerial supply delivery. In *Proc. of the IEEE/RSJ Intl. Conf. on Intell. Robots and Syst.*, pages 1146–1153, San Francisco, CA, September 2011.

[11] T. Lee, M. Leoky, and N.H. McClamroch. Geometric tracking control of a quadrotor uav on SE(3). In *Proc. of the Intl. Conf. on Decision and Control*, pages 5420–5425, Atlanta, GA, December 2010.

[12] B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proc. of the Intl. Joint Conf. on Artificial Intelligence*, pages 24–28, Vancouver, Canada, August 1981.

[13] Y. Ma, S. Soatto, J. Kosecka, and S. Sastry. *An invitation to 3D vision: from images to geometric models*. Springer, 2001.

[14] R. V. D. Merwe, E. A. Wan, and S. I. Julier. Sigma-point Kalman filters for nonlinear estimation: Applications to integrated navigation. In *Proc. of AIAA Guidance, Navigation, and Controls Conf.*, Providence, RI, August 2004.

[15] D. Nister. An efficient solution to the five-point relative pose problem. In *Proc. of IEEE Computer Society Conf. on Computer Vision and Pattern Recognition*, volume 2, pages 195–202, Madison, WI, June 2003.

[16] C. Richter, A. Bry, and N. Roy. Polynomial trajectory planning for quadrotor flight. In *RSS Workshop on Resource-Efficient Integration of Perception, Control and Navigation for MAVs*, Berlin, Germany, 2013.

[17] D. Scaramuzza, A. Martinelli, and R R. Siegwart. A flexible technique for accurate omnidirectional camera calibration and structure from motion. In *Proc. of IEEE Intl. Conf. of Vision Systems*, New York, NY, January 2006.

[18] S. Shen, N. Michael, and V. Kumar. Autonomous multi-floor indoor navigation with a computationally constrained MAV. In *Proc. of the IEEE Intl. Conf. on Robot. and Autom.*, pages 20–25, Shanghai, China, May 2011.

[19] S. Shen, Y. Mulgaonkar, N. Michael, and V. Kumar. Vision-based state estimation for autonomous rotorcraft MAVs in complex environments. In *Proc. of the IEEE Intl. Conf. on Robot. and Autom.*, Karlsruhe, Germany, May 2013.

[20] J. Shi and C. Tomasi. Good features to track. In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition*, pages 593–600, Seattle, WA, June 1994.

[21] H. Strasdat, J. M. M. Montiel, and A. J. Davison. Scale drift-aware large scale monocular SLAM. In *Proc. of Robot.: Sci. and Syst.*, Zaragoza, Spain, June 2010.

[22] J.-P. Tardif, Y. Pavlidis, and K. Daniilidis. Monocular visual odometry in urban environments using an omnidirectional camera. In *Proc. of the IEEE/RSJ Intl. Conf. on Intell. Robots and Syst.*, pages 2531–2538, Nice, France, September 2008.

[23] S. Weiss, M. W. Achtelik, S. Lynen, M. Chli, and R. Siegwart. Real-time onboard visual-inertial state estimation and self-calibration of mavs in unknown environments. In *Proc. of the IEEE Intl. Conf. on Robot. and Autom.*, pages 957–964, Saint Paul, MN, May 2012.