Model-Based Global Localization for Aerial Robots Using Edge Alignment

Kejie Qiu, Tianbo Liu, and Shaojie Shen

***Abstract*—Robust state estimation is the core capability for autonomous aerial robots operating in complex environments. Global navigation satellite system and visual odometry/SLAM are popular methods for state estimation. However, there exist scenarios, such as when flying between tall buildings or in the middle of deep canyons, that all these methods fail due to obstructed sky view and high altitude. in this letter, inspired by the fact that offline-generated three-dimensional (3-D) models of cities and natural scenes are readily available, we propose a global localization method for aerial robots by using 3-D models and measurements from a monocular fisheye camera and an inertial measurement unit (IMU). Due to the fact that 3-D models are generated by different cameras at different times, traditional feature-based or direct registration methods usually fail to perform, we therefore propose to use an edge alignment-based method for image-to-model registration under strong changes in lighting conditions and camera characteristics. We additionally aid our model-based localization with electronic image stabilization for better tracking performance, and extended Kalman filter (EKF)-based sensor fusion for closed-loop control. Our method runs onboard an embedded computer in real time. We verify both local accuracy and global consistency of the proposed approach in real-world experiments with comparisons against ground truth. We also show closed-loop control using the proposed method as state feedback.**

***Index Terms*—Aerial robotics, localization, omnidirectional vision.**

I. Introduction

Aerial robots are attractive for a wide range of applications due to their superior mobility in complex environments. We are interested in equipping aerial robots with autonomous navigation capabilities for delivery and infrastructure inspection applications in large-scale urban environments. One of the core requirements for such applications is reliable global localization, GNSS-based solutions have been widely deployed for this purpose. However, GNSS can easily be disrupted or disabled due to obstructed sky view, which are very common in urban environments. Global localization solutions based on place recognition [1], [2] can only obtain topological localization which is not accurate enough for closed-loop flight control. Majdik *et al.* proposed an appearance-based urban localization system using air-ground matching in which topological and metric localizations are combined [3]. One may also rely on relative localization methods including visual odometry and SLAM-based approaches using monocular camera [4]–[7], stereo cameras [8], RGBD sensors [9], and laser scanner [10]. However, odometry-based methods suffer from long-term drifting. SLAM-based approaches do not guarantee global consistency before major loop closures are detected. Fusion of odometry, SLAM and GNSS may resolve the localization in most cases, but it still does not guarantee drift-free localization at all times. All these issues drive us to find a lightweight global localization solution that is always available in urban environments.

Fortunately, we can now easily obtain 3D models of major urban areas through multiple online resources or online model construction services, such as Google Maps, Altizure, Skycatch,[1] etc. These dense 3D models are constructed using well-studied offline structure from motion (SfM) techniques, in which all the camera poses and pixels are optimized jointly to achieve globally dense consistency. This motivates us to utilize these 3D models to achieve global localization.

To this end, global localization is essentially an image registration problem which aims to align the image capture by the onboard camera and a virtual image rendered from the 3D model. This requires fast rendering onboard embedded computers. Luckily, today's high performance mobile devices for graphics computation make real-time rendering a reality. The rendering process provides both a virtual RGB image and a depth image with nearly unlimited range, from which localization can be done by using only the sensory input from one camera. We further improve the performance by using a fisheye camera aided with inertial measurement unit (IMU)-based electronic image stabilization. This enables us to use only one camera to capture a large field of view rather than just looking downwards, thus fully utilize the 3D information provided by the model at all altitudes. The EIS generates images that are as stable as those from mechanical gimbals and enables online selection of arbitrary viewing angles. It reduces image motions thus gives robust pose tracking performance. The global pose estimations from image-to-model alignment are further smoothed

Manuscript received September 10, 2016; accepted January 17, 2017. Date of publication January 26, 2017; date of current version March 10, 2017. This letter was recommended for publication by Associate Editor F. Kendoul and Editor J. Roberts upon evaluation of the reviewers' comments. This work was supported by the WeChat-HKUST Joint Laboratory on Artificial Intelligence Technology (WHAT LAB).

The authors are with the Department of Electronic and Computer Engineering, Hong Kong University of Science and Technology, Hong Kong (e-mail: kqiuaa@connect.ust.hk; tliuam@connect.ust.hk; eeshaojie@ust.hk).

This letter has supplemental downloadable multimedia material available at http://ieeexplore.ieee.org, provided by the authors. The Supplementary Materials contain an MP4 file showing the quadrotor trajectory tracking using the model+EKF state estimation as controller feedback. This material is 10 MB in size.

Color versions of one or more of the figures in this letter are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/LRA.2017.2660063

[1]www.altizure.com, www.skycatch.com

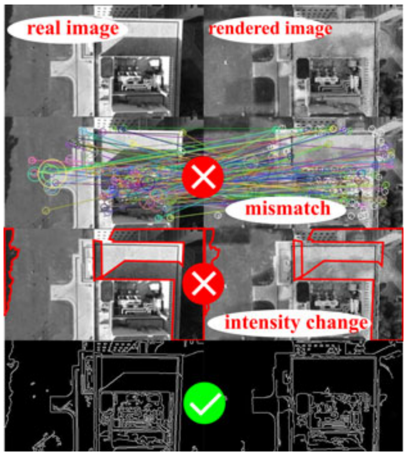


Fig. 1. A comparison of three image alignment methods. The left column shows the captured images while the right column denotes rendered images for alignment. Feature-based and dense alignment methods can not work due to significant changes in lighting condition and camera characteristics. Feature descriptors do not preserve due to the rendering process, Lighting variations, such as the shadow areas marked in red, have major negative impact on direct (dense) alignment-based methods. Our edge alignment-based method functions properly as edges can be preserved even after rendering, and intensity changes does not cause major variations to the distribution of edges (Section IV).

by extended Kalman filter (EKF)-based IMU fusion in order to achieve low latency and high bandwidth state estimation for closed-loop feedback control. The whole system is sufficiently lightweight and runs real-time on an embedded computer. The local estimation accuracy is validated by comparison against a ground truth provided by a motion capture system, the global localization consistency is also proved through the comparison.

Related work on 3D edge-based tracking with multiple hypotheses are proposed years ago [11], [12], but only Computer-Aided Design (CAD) model or simple geometric model like cubes are used for edge extraction. The most relevant work is a direct alignment-based tracking system on a known mesh model constructed by Kinect fusion for indoor environments [13]. However, such method cannot be applied to our application due to significant changing illumination conditions and camera characteristics. In fact, comparing to images used for online localization, images used for constructing the 3D model construction are captured by another aerial robot at different time and date, with different cameras, from different view points, and even under different weather conditions. This challenging situation is evidenced by a comparison of different image alignment methods shown in Fig. 1. We first studied different methods for image-to-model registration. Feature-based method definitely does not work due to major changes to feature descriptors resulting from the reconstruction-rendering process. Direct method works well in indoor environment but also suffers in outdoor cases due to the likely substantial changes in illumination.

Fortunately, not all image features are destroyed during the reconstruction-rendering process, and we see from Fig. 1 that edges are mostly preserved. We therefore opt to utilize our recently proposed edge-based method [14] that geometrically align edges on a distance transform field to achieve pose estimation. This method was inspired by the classical ICP algorithm [15] that make use of the geometry information for frame alignment. [16] uses the distance transform to model the point correspondence function to align 2D or 3D point cloud. Our method generalizes [16] such that it is able to handle the 3D-2D motion estimation problem. Comparing to direct methods, ICP and edge-based methods are significantly more robust against changes to lighting conditions since the photo-consistency assumption for direct methods is no longer necessary.

To this end, we address application scenarios that GNSS, visual odometry, and SLAM are unable to solve by utilizing the power of readily available 3D models. We identify our contributions as follows:

1) We propose a practical global localization system for aerial robots operating in large-scale urban environments by utilizing offline-generated 3D models.
2) We design block rendering scheme, electronic image stabilization system, and edge-based image-to-model registration method, to achieve real-time and robust pose estimation.
3) We implement the complete autonomous aerial robot system platform to verify localization performance and demonstrate closed-loop control capability.

II. Overview

The overall structure of the proposed approach is shown in Fig. 2. Given an initial guess of the camera pose, the rendering thread generates a virtual RGB image and its depth map accordingly (Section III). This process is identical to those in computer graphics where a local view is generated by putting a virtual camera into the scene. For every fisheye image captured by the camera, it first goes through the EIS module to obtain a cropped and stabilized pinhole image (Section IV-B). The EIS enables us to handle low altitude and high altitude cases by changing the viewing angle of the stabilized image. It also improves the robustness under agile motions. Edges are extracted from the stabilized image to eliminate impacts from different illumination conditions, shadows, etc. This edge image is further turned into a distance transform for constructing the cost function for edge alignment. At the same time, edges of the rendered RGB image are also extracted and further converted into 3D edges since the depth values of the virtual image are known. The 3D edges can be reprojected onto the distance transform field, and relative translation and rotation between the actual camera view and the rendered view (from initial guess) can be derived by minimizing the reprojection error (Section IV-C). The global pose is computed following standard pose compounding of relative transformations. The updated global pose is fed back to the rendering thread for the next rendering and registration. The global pose is fused with IMU using an EKF to obtain state estimates for closed-loop control (Section IV-D). Section V gives the experimental results with comparison to ground truth for indoor case and comparison to GPS and a state-of-the-art visual-inertial state estimator (VINS) [7] for outdoor case. Conclusion and directions for future work are presented in Section VI.

III. 3D Model Composition and Rendering

Structure from motion (SfM) is a well studied problem in computer vision. Commercial image-based modeling software

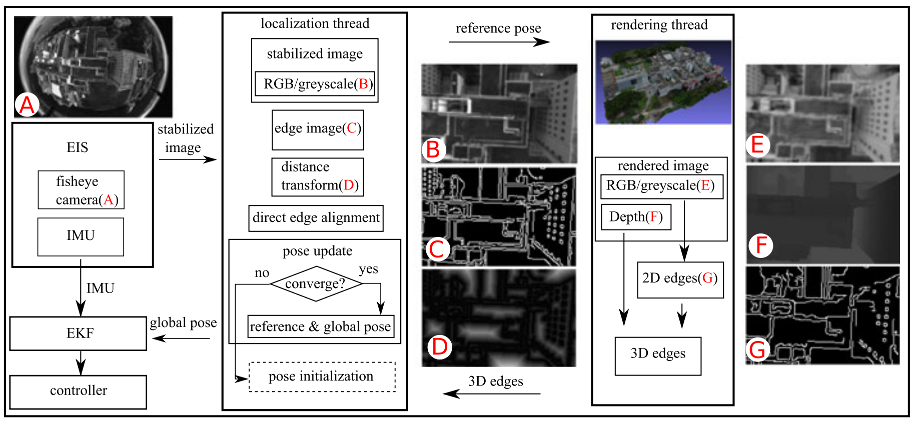


Fig. 2. The overall structure of the proposed model-based global localization system.

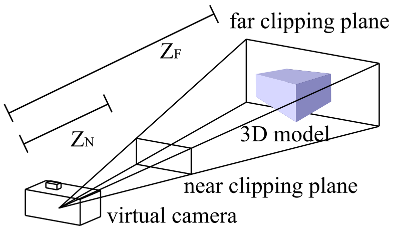


Fig. 3. View frustum culling.

is available for converting photos taken by drones into accurate 3D models [17]. Similar to computer graphics applications, we use textured mesh model directly for view rendering. Three points comprise a basic face with texture attached to it. Rendering on such a mesh model provides a vivid virtual image view for image alignment.

We use wavefront OBJ [18] for OpenGL-based real-time rendering. OpenGL is a standard application programming interface for drawing 2D and 3D graphics. The scale of the model and the transformation between world frame and the model frame are calibrated offline. Given the pose of a virtual camera view point in terms of the model frame, the basic rendering process for RGB image and depth image is detailed in [13]. Both the RGB buffer and the depth buffer store the rendered data ranging from the near clip plane ($\mathrm{Z_N}$) to the far clip plane ($\mathrm{Z_F}$) using frustum culling, a rendering technology in 3D computer graphics, as shown in Fig. 3. The rendered color image can be directly read from the RGB buffer, which will be converted into grayscale image for further use, while the depth buffer ($\mathrm{D}(i,j)$) is actually non-linear. The actual values stored in the depth buffer memory are related to the real depth Z of the object in the following manner:

$$\mathrm{Z} = \mathrm{Z_N} + \mathrm{c}\left(\frac{\mathrm{Z_N}}{\mathrm{Z_F}/\mathrm{D}(i,j) - \mathrm{c}}\right), \tag{1}$$

where $\mathrm{c} = (\mathrm{Z_F} - \mathrm{Z_N})/2^N$, and N is the number of bits of the depth buffer. We get the rendered depth image according to Equation (1). For more efficient rendering in a real localization process, we split the original model into several overlapped blocks and select the nearest block for rendering.

IV. Edge Alignment-Based Global Localization

A. Notations

We consider $(\cdot)^m$ as the model frame, $(\cdot)^w$ as the world frame, $(\cdot)^b$ as the IMU-body frame, $(\cdot)^c$ as the fisheye camera frame, $(\cdot)^s$ as the EIS-stabilized image frame, and $(\cdot)^v$ as the virtual camera frame. $(\cdot)_k$ means timestamp k. We denote $(\cdot)^{b_k}$ as the IMU-body frame at timestamp k. $\mathbf{p}_y^x$, $\mathbf{v}_y^x$ and $\mathbf{R}_y^x$ are the 3D translation, velocity and rotation of frame $(\cdot)^y$ respectively with respect to frame $(\cdot)^x$. The skew-symmetric matrix operator is denoted as $\lfloor(\cdot)\times\rfloor$. The focal lengths of the view camera are denoted as f_x and f_y respectively, and the corresponding image gradients of the distance transform are represented by G_u and G_v. The rendered image is denoted as $\mathbf{I} : \Omega \subset \mathbb{R}^2 \longmapsto \mathbb{R}$, where Ω represents the image domain. The rendered depth image is denoted as $\mathbf{Z} : \Omega \subset \mathbb{R}^2 \longmapsto \mathbb{R}$. $\mathbf{P} \in \mathbb{R}^3$ denotes a 3D scene point in the coordinate system of the camera optical center. As for the rigid transformation description, we use $\mathbf{R}_s^v$ and $\mathbf{p}_s^v$ together to denote the relative transformation between the virtual camera frame and the stabilized image frame, and $\mathbf{R}_v^w$ and $\mathbf{p}_v^w$ together to denote the global transformation from the world frame to the virtual camera frame. Further, we define the camera projection function $\pi : \mathbb{R}^3 \longmapsto \mathbb{R}^2$ projects the 3D points onto the image domain. And the inverse projection function $\pi^{-1} : (\mathbb{R}^2, \mathbb{R}) \longmapsto \mathbb{R}^3$ back-projects a pixel coordinate at this pixel coordinate given the depth value:

$$\mathbf{u} = \pi(\mathbf{P}), \tag{2}$$

$$\mathbf{P} = \pi^{-1}(\mathbf{u}, \mathrm{Z}(\mathbf{P})), \tag{3}$$

where $\mathbf{u} \in \mathbb{R}^2$ denotes the image coordinate of a 3D point $\mathbf{P}$. Particularly, we use $\mathbf{e} \in \mathbb{R}^2$ to denote an edge point. Let $\mathbf{P}_v^i$ denote the ith 3D scene point of the virtual camera frame and $\mathbf{u}_s^j$ denote the jth image point of the stabilized image frame. $\mathrm{X}(\mathbf{P})$, $\mathrm{Y}(\mathbf{P})$, $\mathrm{Z}(\mathbf{P})$ are the specific position values of the 3D point $\mathbf{P}$ in the x, y and z axes respectively.

The relationship between relevant frames is shown in Fig. 4. After edge-based motion estimation, we use the estimated pose of the stabilized image as the next pose for the virtual camera

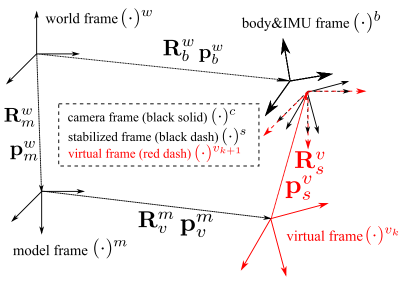

world frame $(\cdot)^w$
body&IMU frame $(\cdot)^b$
$\mathbf{R}_b^w\ \mathbf{p}_b^w$
$\mathbf{R}_m^w$
$\mathbf{p}_m^w$
camera frame (black solid) $(\cdot)^c$
stabilized frame (black dash) $(\cdot)^s$
virtual frame (red dash) $(\cdot)^{v_{k+1}}$
$\mathbf{R}_s^v$
$\mathbf{p}_s^v$
$\mathbf{R}_v^m\ \mathbf{p}_v^m$
model frame $(\cdot)^m$
virtual frame $(\cdot)^{v_k}$

Fig. 4. Coordinate frames in the system. We set the body frame as the IMU frame. The transformation between the model frame and the world frame are calibrated offline using least square minimization with ground truth or good GNSS signal reception. The pose of the current stabilized view will be used for next (k + 1) virtual image rendering, thus their frames are identical.

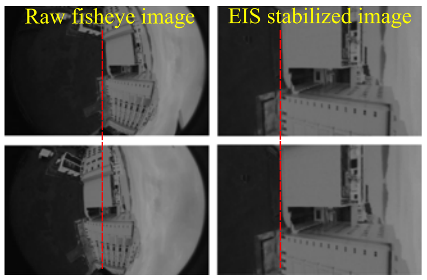


Fig. 6. The image stabilization results from EIS. Similar stabilized images can be obtained even with different raw fisheye images. This gives the motion estimation smoother and more robust tracking performance. Note that the EIS functions well even if there is slight rotation error in the IMU-based rotation estimation.

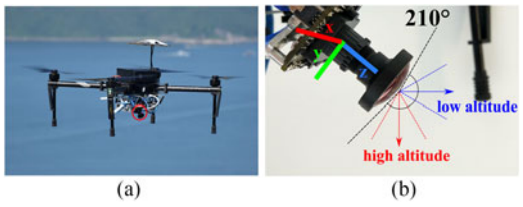


Fig. 5. The quadrotor used for experiments and the ultra wide FOV fisheye camera for model-based localization. The electronic image stabilization system operates on the ultra wide (210 degrees) FOV fisheye camera. It stabilizes the image and minimizes pixel movements even under agile motions (Fig. 6). Different orientations for downward and forward-facing viewing angles, which cover both high and low altitude situations, can also be realized by using a single camera. (a) Quadrotor testbed. (b) Fisheye camera.

in the model frame,

$$\mathbf{R}_{v_{k+1}}^m = \mathbf{R}_{s_k}^m = \mathbf{R}_{v_k}^m \mathbf{R}_{s_k}^{v_k}, \tag{4}$$

$$\mathbf{p}_{v_{k+1}}^m = \mathbf{p}_{s_k}^m = \mathbf{p}_{v_k}^m + \mathbf{R}_{v_k}^m \mathbf{p}_{s_k}^{v_k}. \tag{5}$$

where, $\mathbf{R}_s^v$ and $\mathbf{p}_s^v$ are the motion tracking results and $\mathbf{R}_c^s$ is the rotation compensation matrix for EIS. Finally, the updated global location can be derived as

$$\mathbf{R}_b^w = (\mathbf{R}_w^m)^{-1} \mathbf{R}_v^m \mathbf{R}_s^v \mathbf{R}_c^s (\mathbf{R}_c^b)^{-1}, \tag{6}$$

$$\mathbf{p}_b^w = (\mathbf{R}_w^m)^{-1} (\mathbf{p}_v^m + \mathbf{R}_v^m \mathbf{p}_s^v - \mathbf{p}_w^m) - \mathbf{R}_b^w \mathbf{p}_c^b. \tag{7}$$

B. Electronic Image Stabilization

EIS is essentially rotation compensation on a cropped view of the original fisheye image (Fig. 6). This allows our estimator to handle more agile motions of the aerial platform due to smaller pixel movements after rotation compensation. In addition to the image stabilization, EIS also provides a way to select desired viewing angles to adapt to different flight conditions, as shown in Fig. 5. EIS utilizes the 210 degree field of view (FOV) of an super-fisheye lens. The module receives synchronized attitude information from the onboard IMU and raw fisheye images from the camera, and converts some regions of the distorted fisheye image into a stabilized pinhole image with no lens distortion. Every time a raw image and the corresponding attitude information arrive, EIS computes how much the platform rotates around its x-axis and y-axis. These two rotations are compensated such that the optical axis of the stabilized camera always points toward the desired global orientation, regardless the actual direction of optical axis direction of the fisheye camera.

The roll and pitch angles that needs to be compensated, as well as the desired viewing direction, are assembled into a rotation matrix $\mathbf{R}_s^c$ that rotates a vector in the stabilized image frame into the physical camera frame. The pixel relation between the stabilized pinhole image and the actual image can be computed as follows:

$$\mathbf{u}_c = \pi_c \left(\mathbf{R}_s^c \pi_s^{-1} (\mathbf{u}_s, 1) \right), \tag{8}$$

where π_c is the projection function of the fisheye camera (inclusive of lens distortion), $\mathbf{u}_c$ is the pixel coordinate in the fisheye image. π_s is the projection function of the stabilized camera with no lens distortion, and $\mathbf{u}_s$ is its pixel coordinate. The depth for inverse projection is set to one since we are only compensating for rotation, not translation. As the pixel coordinates are calculated, a stabilized pinhole image can be obtained using bilinear interpolation (Fig. 6). The stabilized image and corresponding compensating rotation of the stabilized camera are pairwise packed in order to provide an exact extrinsic parameter for every stabilized image. We use the fisheye camera model from [19], and use CamOdoCal [20] as the camera calibration software.

We stress that while the EIS module requires the rotation estimate from the IMU, its performance is not affected by slight error in the IMU rotation estimate. In fact, as far as the rotation can be roughly compensated, the pixel movements between two consecutive images will be greatly removed. This already helps a lot to achieve better convergence for the edge-based motion estimation module (Section IV-C).

C. Edge Alignment-Based Motion Estimation

We construct a formulation based on the geometric error between edges, rather than the photometric error used in direct methods, for estimating the relative pose between the current stabilized image frame and the virtual image frame rendered from the 3D model [14]. The proposed residual is the sum of the distances of the re-projections of edge points from the virtual camera image and the nearest edge points in the current stabilized image:

$$r(\mathbf{R}_s^v, \mathbf{p}_s^v) = \sum_i \min_j \mathbf{D}^2(\pi[\mathbf{R}_s^{vT}(\mathbf{P}_v^i - \mathbf{p}_s^v)], \mathbf{u}_s^j), \quad (9)$$

where $\mathbf{D} : (\mathbb{R}^2, \mathbb{R}^2) \longmapsto \mathbb{R}$ denotes the Euclidean distance between those points. The optimal estimations for the relative transformation can be obtained by:

$$\begin{aligned} &\text{minimize } r(\mathbf{R}_s^v, \mathbf{p}_s^v) \\ &\text{subject to } \mathbf{R}_s^v \in \mathrm{SO}(3). \end{aligned} \quad (10)$$

We use the minimal representation, namely $\xi \in \mathrm{se}(3)$ instead of $(\mathbf{R}, \mathbf{p})$ for optimization implementation. The constrained optimization problem can be converted to an unconstrained one:

$$\xi^* = \underset{\xi}{\operatorname{argmin}} \sum_i \min_j \mathbf{D}^2(\pi[\tau(\mathbf{P}_v^i, \xi)], \mathbf{u}_s^j), \quad (11)$$

where $\tau(\mathbf{P}_v^i, \xi) = \mathbf{P}_s^i = [\exp(\xi)]^{-1}\mathbf{P}_v^i = \mathbf{R}_s^{vT}(\mathbf{P}_v^i - \mathbf{p}_s^v)$. We also observe that if the image points corresponding to edge points($\mathbf{e}_v^i \in \mathbb{R}^2$) in the virtual camera image are preselected then the minimization object function is exactly the definition of the Distance Transform [21]. We use $\mathbf{V}^{(n)} : \mathbb{R}^2 \longmapsto \mathbb{R}$ to denote the distance transform of the edge-map of the current stabilized image. The energy term for an edge-pixel of the virtual camera frame will be

$$\mathbf{v}_{\mathbf{e}_v^i}(\xi) = \mathbf{V}^{(n)}(\pi[\tau(\pi^{-1}(\mathbf{e}_v^i, \mathrm{Z}(\mathbf{e}_v^i)), \xi)]), \quad (12)$$

The optimal value ξ^* is given by:

$$\xi^* = \underset{\xi}{\operatorname{argmin}} \sum_{\forall \mathbf{e}_v^i} (\mathbf{v}_{\mathbf{e}_v^i}(\xi))^2. \quad (13)$$

To be specific, we apply Gaussian Newton method to solve this optimization problem. The update equation will be

$$\mathbf{J}^T\mathbf{W}\mathbf{J}\Delta\xi = -\mathbf{J}^T\mathbf{W}r(0), \quad (14)$$

where $\mathbf{J}$ is the stacked matrix of all $\mathbf{J}_i$ pixel-wise Jacobians and $\mathbf{W}$ is the diagonal matrix of weights with $\mathbf{W}_{ii} = w(r_i)$. Here we adopt the weight function derived from the t-distribution residual assumption [9], and

$$\mathbf{J}_i = \frac{\partial \mathbf{v}_{\mathbf{e}_v^i}}{\partial \mathbf{P}_v^i}\frac{\partial \mathbf{P}_v^i}{\partial \xi}, \quad (15)$$

where

$$\frac{\partial \mathbf{v}_{\mathbf{e}_v^i}}{\partial \mathbf{P}_v^i} = \begin{bmatrix} \mathrm{G}_u f_x/\mathrm{Z}(\mathbf{P}_v^i) \\ \mathrm{G}_v f_y/\mathrm{Z}(\mathbf{P}_v^i) \\ -\mathrm{G}_u f_x \mathrm{X}(\mathbf{P}_v^i)/\mathrm{Z}(\mathbf{P}_v^i)^2 - \mathrm{G}_v f_y \mathrm{Y}(\mathbf{P}_v^i)/\mathrm{Z}(\mathbf{P}_v^i)^2 \end{bmatrix},$$

$$\frac{\partial \mathbf{P}_v^i}{\partial \xi} = \mathbf{R}_s^{vT}[-\mathbf{I} \mid \lfloor(\mathbf{P}_v^i - \mathbf{p}_s^v)\times\rfloor].$$

The relative pose ξ is updated by $\xi = \log(\exp(\xi)\exp(\Delta\xi))$, the one with the smallest residual value will be set as ξ^*, and we can recover the relative translation $\mathbf{p}_s^v$ and rotation $\mathbf{R}_s^v$ from it to get the relative transformation between the virtual camera frame and the current stabilized frame. We only project the edges of the virtual camera image onto the stabilized image for residual computation, thus the edges come from the shadows of the stabilized image will not contribute to the alignment result by using such a single direction checking scheme. As shown in Fig. 1, this is also a major difference from the photometric error-based methods.

TABLE I
PERFORMANCE COMPARISON OF ALIGNMENT METHODS

	Feature-based	Dense method	Edge-based
indoor	×	✓	✓
outdoor	×	✓̸	✓

×: does not work; ✓: works well; ✓̸: depends on illumination condition

D. EKF-Based Sensor Fusion

The model-only pose estimation results are not suitable for flight control because of the relatively low frame rate and the noisy estimation output, as shown in Fig. 8. We combine the global localization result $\mathbf{p}_b^w$ and $\mathbf{R}_b^w$ with IMU measurements using an loosely-coupled EKF framework for velocity ($\mathbf{v}_b^w$) estimation, delay compensation, and frame rate boosting. We apply the ring-buffer scheme for delayed measurement update by aligning a visual pose estimation with an earlier measurement [22]. Meanwhile, $\mathbf{p}_v^m$ and $\mathbf{R}_v^m$ will be sent to the rendering thread as the new virtual camera pose for the next round of view rendering, as shown in Fig. 4. The output of this EKF can directly be used for closed-loop feedback control. If additional visual odometry or similar relative state estimation modules are available, we can also conveniently fuse them to further improve the state estimation accuracy.

V. EXPERIMENTAL RESULTS

Please note that the main contribution of our method is to provide a possibility, that 'global' localization results can be obtained all the time during flight, GPS cannot work in some cases while incremental localization, e.g., VINS suffers from the intrinsic long-term drift before a good loop closure detection. Thus we do not focus here on the localization accuracy of the proposed method, but the robustness under the situations when other global localization reference can not work well. In fact, the proposed method itself has comparable localization accuracy, while higher accuracy can easily be achieved by fusion with any incremental methods. In that way, all-the-time high local accuracy and global consistency can both be satisfied, which, however, is not the topic discussed in this letter because at this stage our method its own can already be used for control feedback when fusing with IMU in an EKF framework.

Before the decision to utilize edge-based alignment as the motion tracking method is made, we first studied the other two widely-used methods by implementing them as the tracking method in our system. The results are summarized in Table I. Due to the mentioned reasons at the very beginning, feature-based method cannot work at all because of the destroyed feature descriptors after the reconstruction-rendering process on a rough 3D model. Dense method works well in indoor environments

TABLE II
TIMING STATISTICS OF THE WHOLE SYSTEM

	Modules	Time (ms)	Rate (Hz)	Thread
	Rendering	9	16	1
Localization	EIS	4	16	2
	Motion Estimation	25	16	3
EKF	Prediction	0.2	100	4
	Update	0.4	16	4

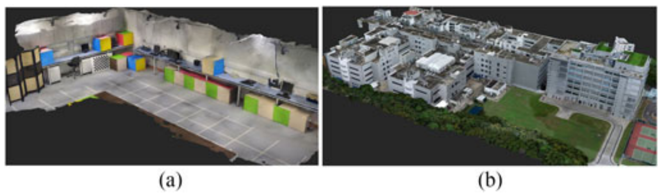


Fig. 7. The 3D models we use for virtual view rendering. They are constructed offline using the service from Altizure.com. (a) Indoor case. (b) Outdoor case.

thanks to the stable illumination condition, but the performance of outdoor case depends on the specific illumination condition. It actually works at a cloudy day but fails at a sunny day or just under the rain (such data is included in our provided dataset[2]). Only edge-based method works well for all kinds of situations, which is naturally adopted in our system.

A. Implementation Details

We use the service by Altizure.com for offline 3D model construction. The experimental testbed (Fig. 5) is a DJI Matrice 100 quadrotor equipped with a mvBlueFOX-MLC200w grayscale camera with a fisheye lens that is installed at a 45-degree tilted angle. The transformation between the body frame and camera frame is calibrated offline, which is also included in the provided dataset together with the camera intrinsic parameters. The onboard IMU runs at 100 Hz, while the camera captures 752 by 480 images at 16 Hz. The size of the stabilized image after EIS is 188 by 120. Onboard computation is a NVIDIA Tegra K1. We use the Linux OS with ROS as the robotics middleware. Our implementation is able to process the stabilized images at 16 Hz.

The initial pose for the model-based estimator is given by ground truth for indoor case, but it is initialized by searching 12 m by 12 m by 12 m cube space centered at a GPS reported location for outdoor case. The convergence of the edge alignment identifies the exact starting point of the localization process. After this point, ground truth or GPS is not needed except for performance comparison purpose. The timing statistics for the whole system, including rendering, localization and EKF are given in Table II. To achieve closed-loop feedback control, we use the same trajectory generation and control scheme described in [23]. The controller receives state commands from the trajectory generation and state estimates from the EKF, and compute the required thrust and the desired attitude of the quadrotor.

B. Indoor Localization Performance

The proposed method is mainly designed for autonomous navigation in large-scale outdoor environment where no ground truth data is provided (GPS is not accurate enough and it's also error-prone). However, in order to validate the localization accuracy of the proposed method, we first construct a 3D model of an indoor environment where a motion capture system[3] is deployed to provide ground truth references, as shown in Fig. 7(a). Limited by the space and safety consideration, we move the drone manually in the indoor environment with different motion patterns. The total trajectory length is about 80 meters. The position, orientation and velocity comparison is shown in Fig. 8. Since the model-only result is quite noisy, we here present quantitative comparison between model+EKF and Ground Truth, as we can see that results from the proposed approach matches well with the ground truth, with a standard deviation of $\{0.0104, 0.0139, 0.0139\}$ (rad) in yaw, pitch and roll, a standard deviation of $\{0.0654, 0.0438, 0.0521\}$ (m) in the x, y and z positions, and a standard deviation of $\{0.1023, 0.0907, 0.0605\}$ (m/s) in the x, y, z velocities, respectively. Thanks to the use of a pre-constructed 3D model as the global localization reference, the state estimates are drift-free. Note that part of the environment is missing in the 3D model, thus the localization accuracy will be affected when the rendered view is not complete. However, this also suggests that our method is able to estimate the sensor pose even if the environment is partially modeled. Finally, we have to note that the localization accuracy depends on the distance between the camera and the objects in the environment due to the projective nature of cameras, scenes at close-proximity lead to relative higher accuracy since small camera motions will cause large differences in the captured view.

[2]https://www.dropbox.com/s/mu2fxgisgp506kn/benchmark_model.zip?dl=0
[3]http://www.optitrack.com/

C. Outdoor Localization Performance

To show the feasibility of the proposed method in large-scale environments, we demonstrate the outdoor case in a 120 m by 90 m outdoor area ranging from 10 m to 90 m in altitude, as shown in Fig. 7(b). The total length of the motion path is about 600 m lasting about 140 s, Results from a state-of-the-art visual-inertial state estimator (VINS) [7] is also included in the comparison, which is reported to have only 0.44$\%$ drift distance of the total trajectory length. Note that the inclusion of GPS and VINS in the experiment is only for qualitative evaluation due to the lack of high-accuracy ground truth. The trajectory comparison is presented in Fig. 9(a), from which we can see the global consistency of the proposed approach, and the drifting of VINS.

The indoor case has shown the localization performance of our system in GPS-denied environments, here we again include the special case when GPS cannot work well in outdoor environments, as shown in Fig. 9(b). Notice that GPS trajectory is on and off because of bad GPS status now and then, while model-based method works well all the time. To further validate the practicability of our method, we implement quadrotor trajectory tracking using the model+EKF state estimation as controller

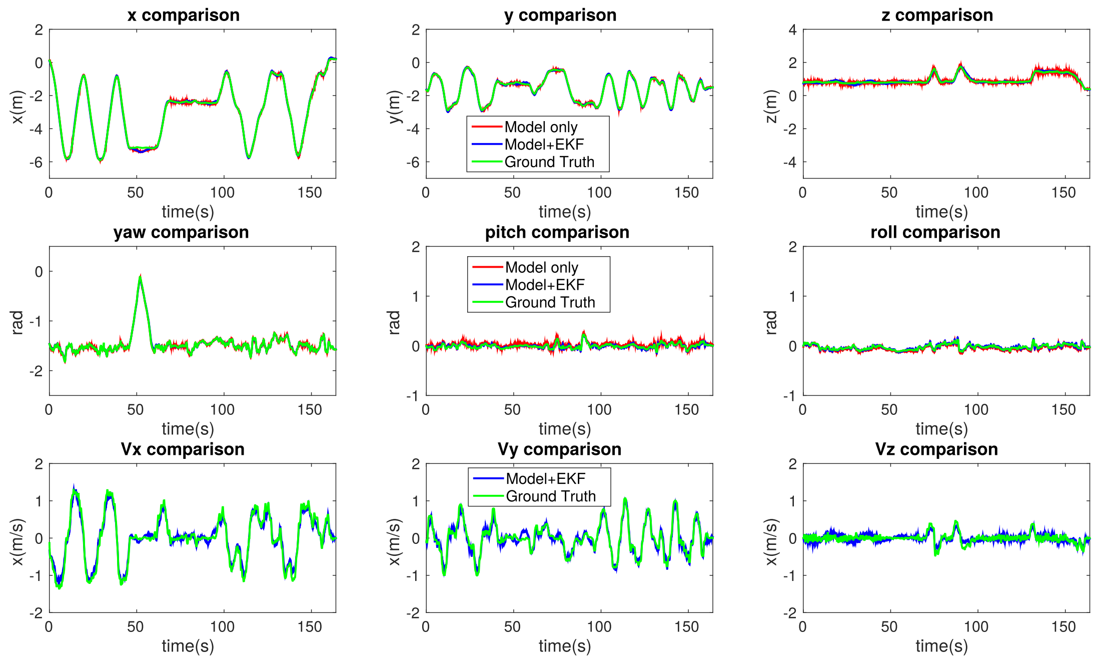
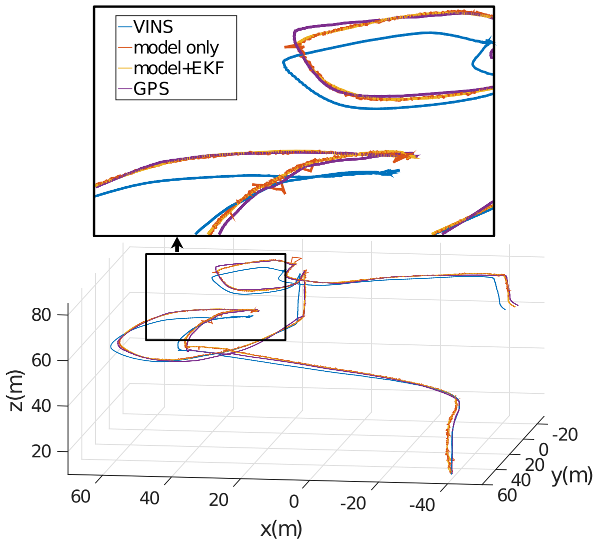
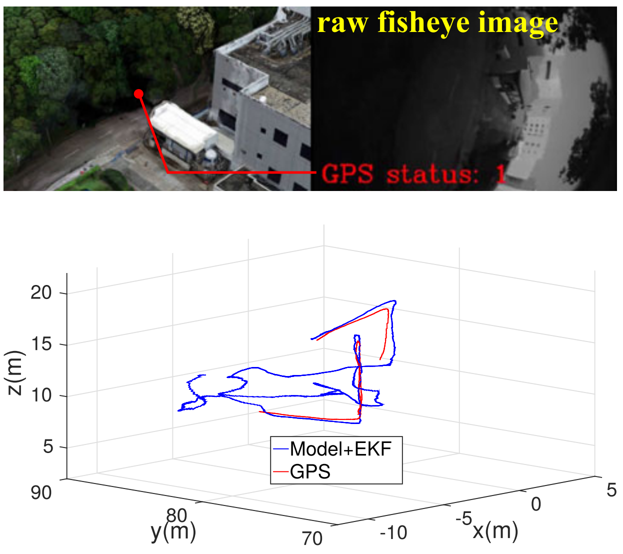


Fig. 8. The comparison of position, orientation and linear velocity. For velocity, we only compare model+EKF and the ground truth. Although the model-only result is noisy, we can see that after EKF fusion, our model-based state estimation is sufficiently smooth and accurate for closed-loop control of aerial robots.



(a)



(b)

Fig. 9. Trajectory comparison between the model-based method and other localization results in outdoor environments. For trial 1, comparison between model-only localization, model+EKF fusion, VINS and GPS is given. We can clearly tell drift in VINS and the noisy results of the model-only method. On the other hand, model+EKF is both globally consistent and locally smooth. Trial 2 shows the flight case when GPS cannot work well. The upper left image shows one example when flying between buildings and woods and the upper right image shows the raw fisheye image with bad GPS status (range from 1 (bad) to 5 (good)) on it. The corresponding trajectory shows the comparison between model+EKF fusion and GPS. Only the GPS segments with good status (status 4 and 5) are plotted, model-based method works well all the time while the GPS data is on and off. (a) Trial 1 with stable GPS. (b) Trial 2 with unstable GPS.

feedback. More details are shown in the video attachment provided in supplementary material for this paper.

VI. Conclusion and Future Work

In this letter, we proposed a novel global localization system utilizing a known 3D textured model. We integrate fast model view rendering, image stabilization, edge-based image alignment and EKF sensor fusion framework to solve the challenging state estimation problem for aerial robots in complex environments. Experimental results show both the global consistency and local accuracy of our system. This estimator can fuse with any additional visual odometry method to achieve better local accuracy. Our solution runs real-time onboard a computationally-constrained quadrotor. In the future, we will improve the initialization/relocalization module. We may also refine the 3D model online using onboard visual information to account for the minor differences between the model and the environment.

VIDEO SUPPLEMENT

The attached video contains three parts; The first part presents the localization accuracy and global consistency by comparing with the ground truth provided in the indoor environment. Three trajectories of model-only, model+EKF and ground truth are shown in different colors. The 3D model used for localization is shown as dense point cloud. The second part shows the real-time localization results in outdoor case. Four trajectories of model-only, model+EKF, VINS and GPS are represented in different colors. The 3D model used for localization is shown as sparse point cloud for display efficiency consideration. The special outdoor case with unstable GPS is also included. Three image views including the raw fisheye image, the electronically stabilized image, and the rendered image are shown simultaneously. Besides comparing different trajectories, it is intuitive to tell the localization performance by visually comparing the similarity between the stabilized image and the rendered image. The third part of the video shows the closed-loop control by a trajectory tracking experiment using the proposed method for state feedback.

REFERENCES

[1] G. Schindler, M. Brown, and R. Szeliski, "City-scale location recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2007, pp. 1–7.

[2] M. Cummins and P. Newman, "FAB-MAP: Probabilistic localization and mapping in the space of appearance," *Int. J. Robot. Res.*, vol. 27, pp. 647–665, 2008.

[3] A. L. Majdik, D. Verda, Y. Albers-Schoenberg, and D. Scaramuzza, "Air-ground matching: Appearance-based GPS-denied urban localization of micro aerial vehicles," *J. Field Robot.*, vol. 32, pp. 1015–1039, 2015.

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

[5] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse, "MonoSLAM: Real-time single camera slam," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 9, pp. 1052–1067, Jun. 2007.

[6] R. Mur-Artal, J. Montiel, and J. D. Tardos, "ORB-SLAM: A versatile and accurate monocular slam system," *IEEE Trans. Robot.*, vol. 31, no. 5, pp. 1147–1163, Oct. 2015.

[7] Z. Yang and S. Shen, "Monocular visual-inertial state estimation with online initialization and camera-IMU extrinsic calibration," *IEEE Trans. Autom. Sci. Eng.*, vol. 14, no. 1, pp. 39–51, Jan. 2017.

[8] S. Leutenegger, S. Lynen, M. Bosse, R. Siegwart, and P. Furgale, "Keyframe-based visual–inertial odometry using nonlinear optimization," *Int. J. Robot. Research*, vol. 34, pp. 314–334, 2015.

[9] C. Kerl, J. Sturm, and D. Cremers, "Robust odometry estimation for RGB-D cameras," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2013, pp. 3748–3754.

[10] D. M. Cole and P. M. Newman, "Using laser range data for 3D SLAM in outdoor environments," in *Proc. 2006 IEEE Int. Conf. Robot. Autom.*, 2006, pp. 1556–1563.

[11] C. Teuliere, E. Marchand, and L. Eck, "3-D model-based tracking for UAV indoor localization," *IEEE Trans. Cybern.*, vol. 45, no. 5, pp. 869–879, May 2015.

[12] G. Klein and D. W. Murray, "Full-3d edge tracking with a particle filter," in *Proc. Brit. Mach. Vis. Conf.*, 2006, pp. 1119–1128.

[13] K. Ok, W. N. Greene, and N. Roy, "Simultaneous tracking and rendering: Real-time monocular localization for MAVs," in *Proc. IEEE Int. Conf. Robot. Autom.,*Stockholm, Sweden, 2016, pp. 4522–4529.

[14] M. P. Kuse and S. Shen, "Robust camera motion estimation using direct edge alignment and sub-gradient method," in *Proc. IEEE Int. Conf. Robot. Autom.,* Stockholm, Sweden, 2016, pp. 573–579.

[15] J. Stückler and S. Behnke, "Model learning and real-time tracking using multi-resolution Surfel maps," in *Proc. AAAI Conf. Artif. Intell.*, 2012, pp. 2081–2087.

[16] A. W. Fitzgibbon, "Robust registration of 2D and 3D point sets," *Image Vis. Comput.*, vol. 21, pp. 1145–1153, 2003.

[17] M. Lhuillier and L. Quan, "A quasi-dense approach to surface reconstruction from uncalibrated images," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 3, pp. 418–433, Mar. 2005.

[18] K. McHenry and P. Bajcsy, "An overview of 3D data content, file formats and viewers," *National Center Supercomputing Appl.*, Urbana, IL, USA, Rep. no. isda08-002, 2008.

[19] C. Mei and P. Rives, "Single view point omnidirectional camera calibration from planar grids," in *Proc. 2007 IEEE Int. Conf. Robot. Autom.*, 2007, pp. 3945–3950.

[20] L. Heng, B. Li, and M. Pollefeys, "CamOdoCal: Automatic intrinsic and extrinsic calibration of a rig with multiple generic cameras and odometry," in *Proc. 2013 IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2013, pp. 1793–1800.

[21] P. Felzenszwalb and D. Huttenlocher, "Distance transforms of sampled functions," Cornell Univ., Ithaca, NY, USA, Tech. Rep. 2004-1963, 2004.

[22] S. Lynen, M. W. Achtelik, S. Weiss, M. Chli, and R. Siegwart, "A robust and modular multi-sensor fusion approach applied to MAV navigation," in *Proc. 2013 IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2013, pp. 3923–3929.

[23] Y. Ling, T. Liu, and S. Shen, "Aggressive quadrotor flight using dense visual-inertial fusion," in *Proc. 2016 IEEE Int. Conf. Robot. Autom.*, 2016, pp. 1499–1506.