Real-time Onboard 6DoF Localization of an Indoor MAV in Degraded Visual Environments Using a RGB-D Camera

Zheng Fang, Sebastian Scherer

***Abstract*— Real-time and reliable localization is a prerequisite for autonomously performing high-level tasks with micro aerial vehicles(MAVs). Nowadays, most existing methods use vision system for 6DoF pose estimation, which can not work in degraded visual environments. This paper presents an onboard 6DoF pose estimation method for an indoor MAV in challenging GPS-denied degraded visual environments by using a RGB-D camera. In our system, depth images are mainly used for odometry estimation and localization. First, a fast and robust relative pose estimation (6DoF Odometry) method is proposed, which uses the range rate constraint equation and photometric error metric to get the frame-to-frame transform. Then, an absolute pose estimation (6DoF Localization) method is proposed to locate the MAV in a given 3D global map by using a particle filter. The whole localization system can run in real-time on an embedded computer with low CPU usage. We demonstrate the effectiveness of our system in extensive real environments on a customized MAV platform. The experimental results show that our localization system can robustly and accurately locate the robot in various practical challenging environments.**

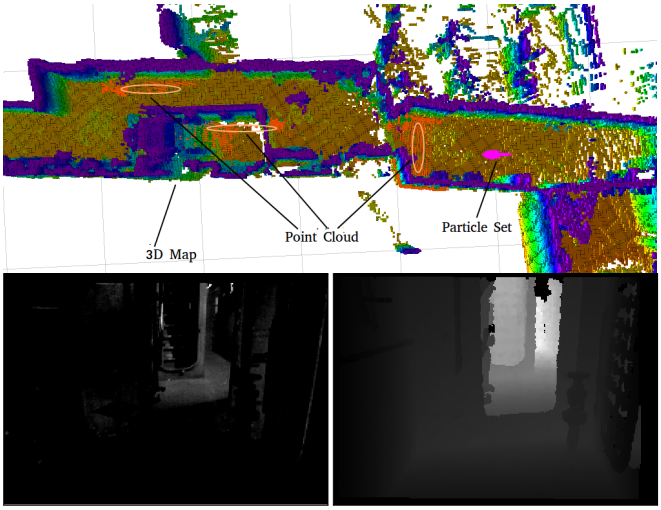

Fig. 1. Localization in degraded visual environments: The top picture shows the 3D map, depth point cloud and particle set. The bottom picture shows the color and depth images output from RGB-D camera

I. INTRODUCTION

Micro Aerial Vehicles (MAVs) rely on accurate location information for a variety of purposes including navigation, motion planning, control and mission completion. Nowadays, most outdoor MAVs obtain their location from the global positioning system (GPS). However, in indoor environments or GPS-denied environments, MAVs must locate themselves using onboard sensors. Due to the payload and power limitations, only a few lightweight sensors can be carried on MAVs for pose estimation. Among all kinds of sensors, cameras are the most popular sensors due to their advantages such as light weight, low power consumption and rich information. In recent years, several vision [1] [2] based pose estimation methods have been proposed for MAVs. Those methods can work very well in feature rich environments. However, they cannot work in featureless or degraded visual environments. Beside vision sensors, lightweight 2D laser scanners are also very popular for indoor MAV pose estimation [3] [4]. However, these methods usually only work in 2D or 2.5D environments since the 2D laser scanner can only detect a plane in a single scan. Recently, consumer-level RGB-D cameras have also become very popular for visual navigation of indoor MAVs [5] [6]. Unfortunately, most methods still heavily rely on either sparse visual features or dense visual information from RGB images. Therefore, they also do not work in featureless or degraded visual environments.

There are several challenges need to be considered for the pose estimation of MAVs in our confined degraded visual environments. First, the vehicle should be small enough to navigate in the narrow environments (Width $<$ 65cm). Therefore, accurate laser scanners, such as Hokuyo UTM-30LX, cannot be used due to MAV's payload restrictions. Second, the onboard computational resources are very limited while pose estimation methods should run in real-time together with other tasks like control, path planning and obstacle avoidance. Third, in degraded visual environments, there are almost no or very few visual features available. Therefore, most existing RGB-D visual odometry or localization methods which use visual information would not work in such environments. Though there are some depth-based RGB-D pose estimation methods, such as ICP [7] or NDT [8] based methods, they are either too slow or computationally too heavy to run on a small MAV.

In this paper, we propose a real-time 6DoF localization system for an indoor MAV that mainly uses depth information from a RGB-D camera in degraded visual environments. An illustrative picture is shown in Fig. 1. To achieve this goal, we first propose a fast and robust relative pose estimation (6DoF Odometry estimation) method that mainly uses depth images. Then, a real-time absolute pose estimation (6DoF Localization) method is proposed to locate the MAV

Zheng Fang is with State Key Laboratory of Synthetical Automation for Process Industries, Northeastern University, Shenyang 110189, China fangzheng@mail.neu.edu.cn

Sebastian Scherer is with the Robotics Institute at Carnegie Mellon University, Pittsburgh, PA 15213, USA basti@cmu.edu

in a given 3D global map with a particle filter. The whole localization system can run in real-time on an embedded computer with low CPU usage. The experiment results show that our localization system can robustly and accurately locate the robot in practical challenging environments.

The rest of this paper is organized as follows. In section II, we discuss the related work. Section III describes the proposed direct RGB-D odometry estimation method. Section IV describes the particle filter based localization algorithm. We validate the performance of our methods by using real datasets in section V and we conclude in section VI.

II. RELATED WORK

The state estimation of MAVs is mainly composed of two sub-problems, namely odometry estimation (relative pose estimation) and localization (absolute pose estimation).

For relative pose estimation, many odometry estimation methods have been proposed with stereo cameras, monocular cameras, RGB-D cameras and 2D laser scanners. In [9], a stereo camera and 2D laser scanner based odometry estimation is proposed for indoor MAVs. This method uses sparse visual feature matching and scan matching algorithm to compute odometry. However, this method doesn't run on the onboard computer. A monocular visual odometry method which can run very fast on an embedded computer is proposed in [1]. However, monocular visual odometry can only estimate the odometry up to an unknown scale. To solve the unknown scale problem, IMU information is used in [2], [10] to estimate the absolute metric scale. In recent years, many RGB-D visual odometry methods have also been proposed. For example, Huang et al [5] propose the *Fovis* RGB-D odometry estimation method for MAVs. In [11], a dense RGB-D visual odometry which minimizes the photometric error between two RGB images is proposed. Pomerleau [7] develops an ICP-based odometry estimation method which only uses depth information. However, though this method can run on our embedded computer (Odroid XU) at 10Hz, the CPU usage is very high.

For absolute pose estimation, there are several ways to locate a robot in a given map. The first kind is 2D method [12] [3]. However, those methods usually only work in structured or 2.5D environments. Some people also use a floor plan for the localization using a RGB-D camera [13]. This method is efficient and fast, but limited to environments with many line features. The second kind is 3D method. A common idea is to create a global point cloud map, and then use ICP or NDT based methods to match the current point cloud to global map. However, those methods are usually very slow. Some researchers also try to use 3D planes as the global map to locate the robot [14] [15]. For example, Fallon [15] proposes Kinect Monte Carlo Localization (KMCL) method. However, this method only estimates x, y and yaw using the particle filter and needs powerful GPU to run in real-time. Oishi [16] uses particle filter to track the robot's pose in a known 3D NDT map. However, this method is still too slow to run on a MAV. Another method is an Octomap based method [17]. But in their paper they have a relatively accurate and robust odometry from the encoders, and everything is running on a remote desktop. Bry [4] also proposes a real-time localization algorithm based on an Octomap for a fix-wing MAV. However, they use a high accurate 2D laser scanner for sensing, which is too heavy to be used on our small quadrotor MAV.

III. ROBUST DIRECT RGB-D ODOMETRY ESTIMATION

In this paper, we use a direct method to compute the frame to frame motion from depth images directly, which is robust and much faster than state-of-the-art ICP based methods. However, if only depth images are used, the odometry will suffer from degeneration problems in some challenging environments. In our method, when severe degeneration happens, we try to use dense visual odometry method to calculate the frame-to-frame motion. By doing so, our odometry method can be computationally efficient and robust in degraded visual environments.

A. Direct Motion Estimation from Depth Images

Most existing depth-based motion estimation methods are based on registration algorithms, such as ICP [7], 3D NDT [8] or 3D geometric feature based methods [18]. Those methods can get very accurate pose estimation if a dense point cloud is available. However, they are too slow and computationally heavy to run on a MAV. In this paper, a direct method based on the idea of [19] is used to calculate the frame-to-frame motion estimation. It directly works on the depth image without detecting any features.

Let a 3D point $R = (X, Y, Z)^T$(measured in the depth camera's coordinate system) is captured at pixel position $r = (x, y)^T$in the depth imageZ_t. This point undergoes a 3D motion $\Delta R = (\Delta X, \Delta Y, \Delta Z)^T$, which results in an image motion Δr between frames t_0 and t_1. Given that the depth of the 3D point will have moved by ΔZ, the depth value captured at this new image location $r + \Delta r$ will have consequently changed by this amount:

$$Z_1(r + \Delta r) = Z_0(r) + \Delta Z \tag{1}$$

This equation is called **range change constraint equation**. Taking the first-order Taylor expansion of the term $Z_1(r + \Delta r)$ generates a pixel-based constraint relating the gradient of the depth image ∇Z_1 and the temporal depth difference to the unknown pixel motion and the change of depth as follows:

$$Z_1(r + \Delta r) = Z_1(r) + \nabla Z_1(r) * \Delta r = Z_0(r) + \Delta Z \tag{2}$$

For a pin hole camera model, any small 2D displacement Δr in image can be related directly to the 3D displacement ΔR which gave rise to it by differentiating the perspective projection equation with respect to the components of the 3D position R

$$\frac{\partial r}{\partial R} = \frac{\Delta r}{\Delta R} = \begin{bmatrix} \frac{f_x}{Z} & 0 & -X\frac{f_x}{Z^2} \\ 0 & \frac{f_y}{Z} & -Y\frac{f_y}{Z^2} \end{bmatrix} \tag{3}$$

where f_x and f_y are the normalised focal lengths.

Substituting equation 3 into equation 2, we can get a linear constraint equation of three unknowns relating the motion ΔR of a 3D point imaged at pixel r to the gradient of the depth image $\nabla Z_1 = (Z_x, Z_y)$ and the temporal depth difference:

$$(Z_x, Z_y, -1)\begin{pmatrix}\Delta r \\ \Delta Z\end{pmatrix} = Z_0(r) - Z_1(r) \tag{4}$$

Under small rotation assumption, if the sensor moves with instantaneous transnational velocity v and instantaneous rotational velocity ω with respect to the environment, then the point R appears to move with a velocity

$$\frac{dR}{dt} = -v - \omega \times R \tag{5}$$

with respect to the sensor. Substituting equation 5 into equation 4, we can get:

$$A\xi = Z_0(r) - Z_1(r) \tag{6}$$

where $A = \begin{bmatrix} -Y - Z_y f_y - Z_x XY\frac{f_x}{Z^2} - Z_y Y^2\frac{f_y}{Z^2} \\ X + Z_x f_x + Z_x X^2\frac{f_x}{Z^2} + Z_y XY\frac{f_y}{Z^2} \\ -Z_x Y\frac{f_x}{Z} + Z_y X\frac{f_y}{Z} \\ Z_x\frac{f_x}{Z} \\ Z_y\frac{f_y}{Z} \\ -1 - Z_x X\frac{f_x}{Z^2} - Z_y Y\frac{f_y}{Z^2} \end{bmatrix}^T$ and $\xi = [\omega_x, \omega_y, \omega_z, v_x, v_y, v_z]^T$. Here, $\omega_x, \omega_y, \omega_z$ and v_x, v_y, v_z are components of the rotation and translation vectors.

This equation generates a pixel-based constraint relating the gradient of the depth image ∇Z_1 and the temporal depth difference to the unknown pixel motion and the change of depth. If there are n pixels in the image, then we can get n such equations with only six unknowns. Here, we use a least-squares error minimization technique to solve the set of equations. In practice, in order to improve the computation speed, the depth image is downsampled to 80×60 which is sufficient to get an accurate estimation.

B. Dealing with Degeneration Problem

The direct depth-based method can estimate the frame-to-frame transform very fast. However, in environments with few geometric features, this method will suffer from the degeneration problem, for example when the camera can only see a ground plane or parallel walls. In these "ill-conditioned" cases which are really common in indoor environments, the direct depth-based method will produce wrong estimates. In such cases, the only way to solve the problem is to try to use additional information, such as RGB or IMU information.

In our algorithm, we try to detect when the degeneration happens. And, when the degeneration happens, we try to use visual information to calculate the frame-to-frame transform. Since we want to get a very fast odometry estimation with low CPU usage, we do not always incorporate the visual information into the depth image based odometry estimation method. Though joint optimization methods [20], [21] may get more accurate estimation, it is much more expensive to compute. In our system, we are not too concern with the accuracy of odometry estimation method since our localization algorithm can correct the drift of visual odometry. Compared to accuracy, robustness is more important for localization since a sudden odometry failure or wrong estimation will influence the localization performance much more than an inaccurate odometry estimation.

Here, we try to analyse the eigenvalues of equation 6 to determine whether the equation is "ill-conditioned". In mathematics, we can use a measure called *condition number*, which is the ratio of the largest to the smallest eigenvalue, to detect the degeneration. We use the condition number to measure the degeneration degree of 6. When severe degeneration happens (condition number > 1000), we try to incorporate RGB information to estimate the frame to frame motion. If the RGB information is not available, then our method output a failure signal to indicate the odometry estimation fails.

C. Direct Motion Estimation from Color Images

The dense visual odometry method [11] is used to estimate the frame-to-frame motion when degeneration happens in the depth based method described in section III-A. In our previous study [22], we found that dense visual odometry is more robust than sparse feature based visual odometry methods in some challenging environments.

Compared to sparse visual feature based methods, this approach is based on the photo-consistency (also called **brightness change constraint equation**) assumption, which means a world point p observed by two cameras is assumed to yield the same brightness in both images.

$$I_1(X) = I_2(\tau(\xi, X)) \tag{7}$$

where $\tau(\xi, X)$ is the warping function that maps a pixel coordinate $X \in \Re^2$ from the first image to a coordinate in the second image given the camera motion $\xi \in \Re^6$. The goal is to find the camera motion ξ that minimizes the photometric error over all pixels. For dense visual odometry method, more details can be found in [11].

IV. Particle Filtering Based Localization

From section III, we can get a robust odometry estimation, however it will definitely drift after a long time of running. In order to get an accurate absolute pose in the environment, we need a localization algorithm to locate the robot in a given 3D environment.

For 6DoF absolute localization in a given 3D Map, there are two important things should be considered. First, what kind of 3D map representation should be used? Second, what kind of localization algorithm should be used? There are several 3D map representation approaches, such as point cloud, planes, NDT map [16], 3D octomap [17] and 3D Polygonal Map [15]. Some of them are raw data based maps, while some of them are feature-based maps. Here, 3D octomap is selected since it is compact and can represent

many kinds of environments. For localization, a particle filter algorithm (also known as Monte Carlo Localization, MCL) is selected since it is very robust, which has already verified extensively on ground mobile robots [23]. Though the MCL has been successfully used on ground mobile robots, 6DoF pose $S = (x, y, z, \phi, \theta, \psi)$ which needs to be estimated for MAVs increases the complexity of the problem. In this paper, we show that by carefully designing the motion and observation model, MCL can work very well on an embedded computer.

A. Particle Filtering Algorithm

Particle Filtering Localization is a Bayes filtering technique which recursively estimates the posterior about the robot's pose S_t at time t:

$$p(S_t|O_{1:t}, u_{1:t-1}) = \eta \cdot p(O_t|S_t) \cdot \int_{s_{t-1}} p(S_t|S_{t-1}, u_t) \cdot p(S_{t-1}|O_{1:t-1}, u_{1:t-1}) dS_{t-1}$$

Here, η is a normalization constant resulting from Bayes' rule, $u_{1:t}$ is the sequence of all motion commands up to time t, and $O_{1:t}$ is the sequence of all observations. $p(S_t|S_{t-1}, u_t)$ is called motion model, which means the probability that the robot ends up in state S_t given it executes the motion command u_t in state S_{t-1}. And, $p(O_t|S_t)$ is the observation model, which means the likelihood of obtaining observation O_t given the robot's current pose is S_t. The particle filter approximates the belief of the robot with a set of particles:

$$S_t = \left\{ \left(s_t^1, w_t^1\right), ..., \left(s_t^n, w_t^n\right) \right\} \tag{8}$$

where, each s_t^i is one pose hypothesis and w_t^i is the corresponding weight. The particle set is updated iteratively by sampling those particles from the motion model and compute a weight according to the observation model. Particles are then resampled according to this weight and the process iterates.

B. Motion Model

For each subsequent frame, we propagate the previous state estimate according to the motion model $p(S_t|S_{t-1}, u_t)$ using the odometry computed by the fast direct RGB-D odometry proposed in section III. The propagation equation is of the form:

$$S_t = S_{t-1} + u_t + e_t \qquad e_t \sim N(0, \sigma^2) \tag{9}$$

where u_t is relative transform estimated from visual odometry and e_t is a small amount of normally distributed noise. For smooth and continuous motion, usually the above noise model works well. However, during abrupt accelerations or sharp turning close to the wall (the minimum and maximum measurement range are around 50cm and 700cm respectively) or in ill-conditioned cases, the odometry algorithm may suffer from periods of total failure. In such cases, we will propagate the particle set using a noise-driven dynamical model replacing Eq 9 with

$$S_t = S_{t-1} + e'_t \qquad e'_t \sim N(0, \sigma'^2) \tag{10}$$

where σ' is much bigger than σ.

C. Observation Model

The belief of vehicle's 6DoF state is updated according to several different sources of sensor information in one observation O_t, namely depth measurements d_t from depth camera, roll $\tilde{\theta}_t$, pitch $\tilde{\phi}_t$ and height measurement $\tilde{z}_t$ from ground plane detection or onboard sensors (IMU and Sonar). Therefore, the final observation model is:

$$\begin{aligned} p(O_t|S_t) = p(d_t, \tilde{z}_t, \tilde{\phi}_t, \tilde{\theta}_t|S_t) = \\ p(d_t|S_t) \cdot p(\tilde{z}_t|S_t) \cdot p(\tilde{\phi}_t|S_t) \cdot p(\tilde{\theta}_t|S_t) \end{aligned} \tag{11}$$

The likelihood formulation $p(\cdot|S_t)$ is given by a Gaussian distribution. Here, the ground plane is used in two ways. First, the ground plane is detected to get the roll, pitch and height measurement. Then, since the ground plane has no contribution for determining the x,y, yaw, it is filtered out when updating the particle's position weight using depth measurement.

In order to detect the ground plane from the point cloud, a RANSAC based method is used. We assume that the ground plane is the furthest plane to the MAV and the closest to horizontal. After detecting the ground plane, roll, pitch and height values can be easily computed from the ground plane equation. Then, the weight of each particle is updated according to the observed measurement and predicted measurement by using following equations.

$$\begin{aligned} p(\tilde{z}_t|S_t) &= \rho(z_t - \tilde{z}_t, \sigma_z) \\ p(\tilde{\phi}_t|S_t) &= \rho(\phi_t - \tilde{\phi}_t, \sigma_\phi) \\ p(\tilde{\theta}_t|S_t) &= \rho(\theta_t - \tilde{\theta}_t, \sigma_\theta) \end{aligned} \tag{12}$$

where $\tilde{z}_t, \tilde{\phi}_t$ and $\tilde{\theta}_t$ are calculated from the detected ground plane, and σ_z, σ_ϕ and σ_θ are determined by the noise characteristics of the ground plane.

In order to evaluate the depth sensing likelihood $p(d_t|S_t)$, we use a sparse subset of beams from the point cloud. From our experiment, we found that how one selects the subset of beams really influences the robustness and accuracy of the localization algorithm. In order to efficiently use the points with most constraints, we try two ways to select points. First, the point cloud is segmented into ground and non-ground point clouds. Since the ground part has little importance to determine the x, y and yaw of the MAV, only very few points from the ground part is selected. For the non-ground part, we found that most time in indoor environments especially in long corridors, there are only few points on the wall are useful for determining the forward translation. If we use a uniform downsampling, then we will miss this valuable information. In order to use this information, we select those points using a Normal Space Sampling method [24]. By doing so, we can select those points with most constraints.

We assume that the sampled measurements are conditionally independent. Here, the likelihood of a single depth measurement $d_{t,k}$ depends on the distance d of the corresponding beam endpoint to the closest obstacle in the map:

$$p(d_{t,k}|S_t) = \rho(d, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} exp(-\frac{d^2}{2\sigma^2}) \tag{13}$$

where σ is the standard deviation of the sensor noise and d is the distance. Since a point cloud measurement consists of K beams $d_{t,k}$, the integration of a full scan is computed as the product of the each beam likelihood. To improve the computation efficiency, an endpoint observation model [23] is used for calculating $p(d_t|S_t)$.

Another issue that should be considered is that depth values of the RGB-D camera are very noisy when the measurement range is bigger than 4 meters. For example, when the measurement distance is less than 3 meters, usually the measurement error is less than 2.5cm. However, when the measurement distance is at 5 meters, the measurement error could be around 7cm. Therefore, the sensor noise is quite different at different distances. In order to include this characteristic into our observation model, we use a changing σ which increases with the measurement distance.

V. Experiments and Analyses

In order to realize localization in a given 3D map, we need to create the global map. In our system, LOAM [25] is used to create the 3D map. In all the experiments, we set our map resolution to 4cm. We test the odometry and localization algorithms in different kinds of environment by carrying or semi-autonomously flying our customized MAV. Our customized quadrotor is equipped with a forward-looking Asus Xtion Pro Live RGB-D camera and Odroid XU embedded computer. The RGB-D camera is used for odometry estimation and localization. We develop our localization system using ROS Indigo, PCL 1.7, OpenCV 2.4 and C++ language. In all experiments, the RGB-D images are streamed at frame rate of 15Hz with QVGA resolution. The experiment video can be found in the attached video file.

A. Illustrative Localization Examples

In this part, the localization algorithm is tested in visual degraded environment and natural office environment. In the degraded visual environments, some areas are very dark and some areas have very few visual or geometric features. In the natural office environment, there are some long clear corridors which pose great challenge for odomtry estimation and localization. We show that our localization system can work well in those environments.

1) Degraded Visual Environment: The first experiment is in a narrow and cluttered environment, which has a size of 16m $\times$ 25.6m $\times$ 4.04m. In this environment, most of the time the RGB images are very dark as shown in Fig. 2, while the depth images are still very good. However, there are some locations that the robot can only see one flat wall in front of it. For example, when the robot turns left at corner (a), since the corridor is very narrow (less than 1m), the robot can only see the wall in front of it. Another example is that when the robot is in the spacious room (b), the depth camera can only see the ground plane and cannot see the wall in front of it. In both scenarios, if just depth images are used for odometry estimation, it will suffer from the degeneration problem. In our system, when the degeneration is severe, RGB information is considered to estimate the odometry. By doing so, our odometry method will avoid suffering from a severe degeneration problem. The localization result in this environment is shown in Fig. 2.

Fig. 2. Localization in degraded visual environment: Pink: Odometry, Red: Localization. The bottom pictures show some snapshots of the environment. The top figure shows the odometry, localization results with the 3D octomap. Note that ceiling and ground are cropped for visualization purpose (same in the later figures).

The second experiment is in a structured but almost completely dark environment, which has size of 11.8m $\times$ 19.2m $\times$ 2.8m. In this environment, we cannot get any useful information from RGB images. There are also some challenging locations where RGB-D camera can only see the ground plane, one wall or two parallel walls, or even detect nothing when it is very close to the wall(Minimum measurement range of the RGB-D camera is around 0.5 meters). In such situations, the depth-based odometry will also suffer from the degeneration problem. In this experiment, if the degeneration is severe, the odometry estimation method will output a odometry failure indicator. Then, our localization algorithm will use the noise-driven motion model to propagate particle set. In our experiment, we find that if the odometry failure is relatively short in duration, it is possible for the localization algorithm to overcome this failure entirely. The localization result in this experiment is shown in Fig. 3.

2) Typical Office Environment: In this experiment, we want to show that our localization system not only works in degraded visual environments, but also works well in normal challenging environments. The test environment is a typical office environment with long clear corridors, which has a size of 64.2m $\times$ 21.2m $\times$ 3.9m. In this environment, the illumination is very good. However, there are also several challenges in this environment for odometry estimation and localization using RGB-D camreas. First, the corridors are very clear. Therefore, there are only few visual features

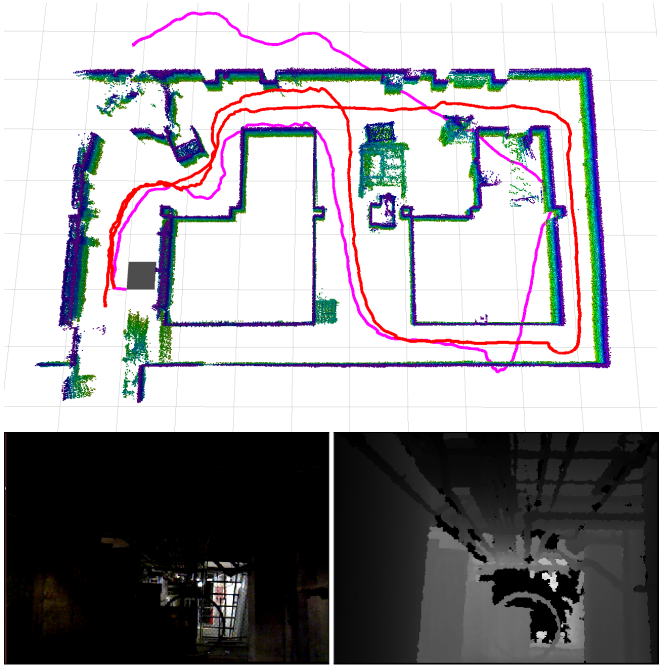

Fig. 3. Localization in completely dark environment: Pink: Odometry, Red: Localization. The top figures shows the odometry, localization results with the 3D octomap. The bottom pictures show some snapshots of the environment.

and geometric features in the corridors, which poses big challenges for odometry estimation and localization. Second, the corridors are very narrow, therefore when the robot turns from one corridor to another corridor, the RGB-D camera can only see a part of the wall. Therefore, the localization system must be robust enough, otherwise it will easily fail around each corner. The third challenge is that the maximum measurement range of RGB-D camera is about 6∼7m and the measurement noise increases along with the distance. However, in this environment there are several corridors whose length are longer than 10 meters. Both the odometry estimation method and localization method should find useful constraints for estimation. In our experiment, we found our localization system can robustly locate the robot in the map. Fig. 4 shows the localization results.

B. Localization Accuracy

In this part, we compare the localization accuracy with ground truth from LOAM mapping system. We attached an Xtion RGB-D camera to the LOAM system and recorded the datasets for offline comparison. Since the estimation accuracy of LOAM system is very high, we could consider its trajectory as ground truth. We test our localization algorithm in two environments. One is a general office environment, where there are many chairs, long tables, long corridors and a lot of office furnitures. This environment is much easier for odometry estimation and localization, since there are lots of visual and geometric features. The other one is in a long tunnel, which is very difficult for odometry estimation and localization using a RGB-D camera since it is very clear. For both experiments, the map resolution is 4cm and the particle

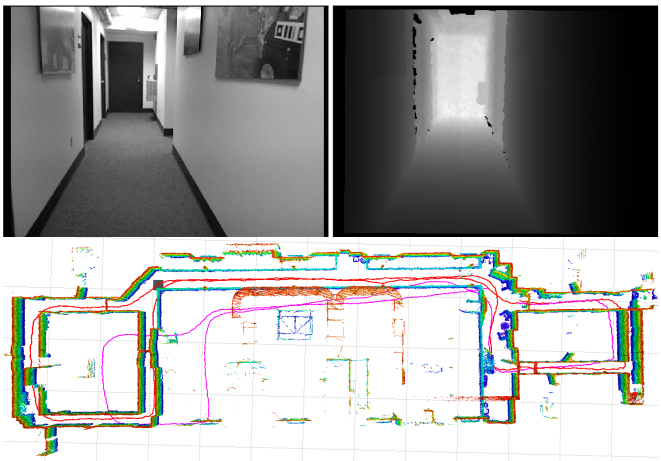

Fig. 4. Localization in typical office environment: Pink: Odometry, Red: Localization. The top pictures show some snapshots of the environment. The bottom figure shows the odometry, localization results with the 3D octomap.

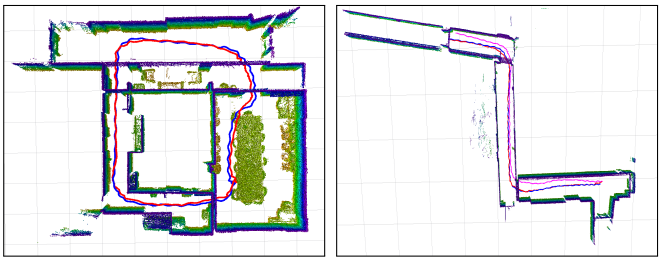

Fig. 5. Accuracy comparison with ground truth in two different kind of environments: Pink: RGB-D Odometry Red: Localization Blue: Ground truth.

number is set to 500. The localization algorithm updates the pose when the robot moves every 10cm or turns 0.1 radians. The experimental results are shown in Table. I. From the experimental results, we also can see that localization accuracy in office environment is better than in long tunnel environment. In long tunnel environment, the biggest error is in the x direction since sometimes there are not enough constraints to determine its position. But our localization algorithm can quickly converge to the true position once there are enough constraints available. The accuracy of our localization algorithm is better than others work [15] and [13]. In their work, their mean localization error is about 40cm, while ours is about 17cm. It should be noted that the localization accuracy changes in different environments or moving at different speeds because it influences the accuracy of odometry estimation dramatically.

TABLE I

LOCALIZATION ACCURACY FOR DATASETS SHOWN IN FIG.5

Environments	Distance	RSME	Mean	Std
Office	47.2m	0.161m	0.152m	0.056m
Tunnel	46.1m	0.235m	0.194m	0.107m

C. Runtime Performance Evaluation

We test the runtime performance of our algorithms on the Odroid XU system, which has two CPUs. One is a quad core 1.6GHz CPU. The other one is a quad core 1.2GHz CPU. Each core has one thread. Our odometry and localization algorithms are both single-threaded programs. Therefore, each algorithm takes only one core. For the experiment in Fig. 2, the runtime performance is shown in Table. II (including drivers). In our experiment, we use 300 particles. Our algorithm can run up to 30Hz on the embedded system. When it is running at 15Hz, the CPU usage is very low which leaves many computation resources for path planning and obstacle avoidance.

TABLE II
RUNTIME PERFORMANCE ON AN EMBEDDED COMPUTER

Name	Algorithm Runtime			
	Mean	Min	Max	StdDev
Odometry	30.3ms	5ms	110ms	20.2ms
Localization	65.8ms	45.8ms	97ms	16.5ms
Total CPU Usage	34.5%	30.5%	44%	2.80%

VI. CONCLUSIONS

This paper presents a localization algorithm for an indoor MAV by using a RGB-D camera. Though our method is designed for degraded visual environments, it also works well in general indoor environments. Our system is based on a fast direct RGB-D odometry estimation method and robust particle filtering localization algorithm. Our localization algorithm can locate the robot robustly and accurately using the onboard RGB-D sensor and embedded computer. Though the system can be made to fail in extreme conditions, such as very fast motion or a lack of both visual and depth features, our system has performed very well in extensive experiments in various indoor environments. In the future, we will test our methods in a fully autonomous navigation experiment (including odometry, localization, obstacle mapping, motion planning and real-time control) in a dark and smoky environment. Furthermore, we will consider to fuse IMU information into the odometry estimation and localization algorithms.

ACKNOWLEDGMENT

Research presented in this paper was supported by United Technology Research Center, China Scholarship Council, NSFC(No.61040014) and Fundamental Research Funds for the Central Universities(No.N120408002). The authors would also like to thank J. Zhang, S. Jain, Y. Zhang and G. Dubey for their help.

REFERENCES

[1] C. Forster, M. Pizzoli, and D. Scaramuzza, "SVO: Fast Semi-Direct Monocular Visual Odometry," *2014 IEEE Int. Conf. Robot. Autom.*, 2014.

[2] 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. - IEEE Int. Conf. Robot. Autom.*, pp. 957–964, 2012.

[3] S. Grzonka, G. Grisetti, and W. Burgard, "Towards a navigation system for autonomous indoor flying," in *2009 IEEE Int. Conf. Robot. Autom.*, pp. 2878–2883, IEEE, May 2009.

[4] A. Bry, A. Bachrach, and N. Roy, "State estimation for aggressive flight in GPS-denied environments using onboard sensing," in *2012 IEEE Int. Conf. Robot. Autom.*, pp. 1–8, IEEE, May 2012.

[5] A. Huang and A. Bachrach, "Visual odometry and mapping for autonomous flight using an RGB-D camera," *Int. Symp. Robot. Res.*, pp. 1–16, 2011.

[6] A. Bachrach, S. Prentice, R. He, P. Henry, a. S. Huang, M. Krainin, D. Maturana, D. Fox, and N. Roy, "Estimation, planning, and mapping for autonomous flight using an RGB-D camera in GPS-denied environments," *Int. J. Rob. Res.*, vol. 31, pp. 1320–1343, Sept. 2012.

[7] F. Pomerleau, F. Colas, R. Siegwart, and S. Magnenat, "Comparing ICP variants on real-world data sets," *Auton. Robots*, vol. 34, pp. 133–148, Feb. 2013.

[8] T. Stoyanov, M. Magnusson, H. Andreasson, and A. J. Lilienthal, "Fast and accurate scan registration through minimization of the distance between compact 3D NDT representations," *Int. J. Rob. Res.*, vol. 31, pp. 1377–1393, Sept. 2012.

[9] M. Achtelik, A. Bachrach, R. He, S. Prentice, and N. Roy, "Stereo vision and laser odometry for autonomous helicopters in GPS-denied indoor environments," in *Proc. SPIE Int. Soc. Opt. Eng.*, vol. 1, pp. 733219–733219–10, 2009.

[10] S. Weiss and R. Siegwart, "Real-time metric state estimation for modular vision-inertial systems," in *Proc. - IEEE Int. Conf. Robot. Autom.*, pp. 4531–4537, 2011.

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

[12] G. Angeletti and J. Valente, "Autonomous indoor hovering with a quadrotor," in *Int. Conf. Simulation, Model. Program. Auton. Robot.*, pp. 472–481, 2008.

[13] J. Biswas and M. Veloso, "Depth camera based indoor mobile robot localization and navigation," in *IEEE Int. Conf. Robot. Autom.*, pp. 1697–1702, 2012.

[14] R. Cupec, E. K. Nyarko, D. Filko, A. Kitanov, and I. Petrović, "Global Localization Based on 3D Planar Surface Segments Detected by a 3D Camera," in *Proc. Croat. Comput. Vis. Work. Year 1*, pp. 31–36, 2013.

[15] M. F. Fallon, H. Johannsson, and J. J. Leonard, "Efficient scene simulation for robust monte carlo localization using an RGB-D camera," in *Proc. - IEEE Int. Conf. Robot. Autom.*, pp. 1663–1670, 2012.

[16] S. Oishi, Y. Jeong, R. Kurazume, Y. Iwashita, and T. Hasegawa, "ND voxel localization using large-scale 3D environmental map and RGB-D camera," in *2013 IEEE Int. Conf. Robot. Biomimetics*, no. December, pp. 538–545, IEEE, Dec. 2013.

[17] D. Maier, A. Hornung, and M. Bennewitz, "Real-time navigation in 3D environments based on depth camera data," in *IEEE-RAS International Conference on Humanoid Robots*, pp. 692–697, 2012.

[18] K. Pathak, A. Birk, N. Vaskevicius, and J. Poppinga, "Fast Registration Based on Noisy Planes With Unknown Correspondences for 3-D Mapping," *IEEE Trans. Robot.*, vol. 26, pp. 424–441, June 2010.

[19] B. K. Horn and J. G. Harris, "Rigid body motion from range image sequences," *CVGIP Image Underst.*, vol. 53, pp. 1–13, Jan. 1991.

[20] P. Henry, M. Krainin, E. Herbst, X. Ren, and D. Fox, "RGB-D mapping: Using Kinect-style depth cameras for dense 3D modeling of indoor environments," *Int. J. Rob. Res.*, vol. 31, pp. 647–663, Feb. 2012.

[21] G. Jones, "Accurate and Computationally-inexpensive Recovery of Ego-Motion using Optical Flow and Range Flow with Extended Temporal Support," in *Procedings of the British Machine Vision Conference 2013*, pp. 75.1–75.11, British Machine Vision Association, 2013.

[22] Z. Fang and S. Scherer, "Experimental Study of Odometry Estimation Methods using RGB-D Cameras," *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Sept. 2014.

[23] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. The MIT Press, 2005.

[24] S. Rusinkiewicz and M. Levoy, "Efficient variants of the ICP algorithm," in *Proc. Third Int. Conf. 3-D Digit. Imaging Model.*, pp. 145–152, IEEE Comput. Soc, 2001.

[25] J. Zhang and S. Singh, "LOAM : Lidar Odometry and Mapping in Real-time," *Robotics: Science and Systems Conference (RSS)*, 2014.