An Onboard Monocular Vision System for Autonomous Takeoff, Hovering and Landing of a Micro Aerial Vehicle

Shaowu Yang, Sebastian A. Scherer and Andreas Zell

***Abstract*— In this paper, we present an onboard monocular vision system for autonomous takeoff, hovering and landing of a Micro Aerial Vehicle (MAV). Since pose information with metric scale is critical for autonomous flight of a MAV, we present a novel solution to six degrees of freedom (DOF) pose estimation. It is based on a single image of a typical landing pad which consists of the letter "H" surrounded by a circle. A vision algorithm for robust and real-time landing pad recognition is implemented. Then the 5 DOF pose is estimated from the elliptic projection of the circle by using projective geometry. The remaining geometric ambiguity is resolved by incorporating the gravity vector estimated by the inertial measurement unit (IMU). The last degree of freedom pose, yaw angle of the MAV, is estimated from the ellipse fitted from the letter "H". The efficiency of the presented vision system is demonstrated comprehensively by comparing it to ground truth data provided by a tracking system and by using its pose estimates as control inputs to autonomous flights of a quadrotor.**

I. INTRODUCTION

Autonomous flight of Unmanned Aerial Vehicles (UAVs) has attracted much research during the past years. Takeoff, hovering and landing are three basic phases for autonomous flight of rotorcrafts. Among them, autonomous landing on a specified target is especially complex because it requires robust recognition of the landing pad and precise position control. Micro unmanned Aerial Vehicles (MAVs) are limited in onboard computational power and payload, thus the ability for onboard sensing and control of these phases turns out to be a challenge.

Compared with other sensors, cameras have a superior potential for environment perception and are usually lightweight. Thus, many UAVs rely on vision systems for autonomous flight tasks, especially in GPS-denied environments, e.g. in indoor applications. For the control of autonomous flight of a MAV, metric pose estimation is essential, which could be provided by stereo cameras. But for a MAV, monocular vision is more compact and less computationally intensive. Monocular vision, however, poses more challenges for pose estimation with metric scale, especially for hovering above or landing on a specified target.

In this paper, we present an onboard monocular vision solution for the full 6 DOF pose estimation of a MAV. A typical landing pad for rotorcrafts with the letter "H" surrounded by a circle can be robustly recognized by the proposed object recognition algorithm, even in cluttered environments. Then the 5 DOF pose of the MAV, its 3D position, roll and pitch angles, is estimated from the projection of the known circle, and the remaining degree of freedom pose, the yaw angle, from the projection of the letter "H". No additional metric scale measurement sensor is needed for pose estimation. As MAVs are nearly always equipped with an IMU, we use its gravity vector measurement as reference data in the 5 DOF pose estimation step. The 3D position and yaw angle estimates are then used as input to a nested PID controller which is used to control hovering of the MAV above the landing pad. Takeoff and landing on the landing pad are achieved by using the proposed point setting approach.

S. Yang and S. A. Scherer are PhD students with the Chair of Cognitive Systems, Department of Computer Science, University of Tübingen, Sand 1, D-72076 Tübingen, Germany {shaowu.yang, sebastian.scherer}@uni-tuebingen.de

A. Zell is full professor with the Chair of Cognitive Systems, Department of Computer Science, University of Tübingen, Sand 1, D-72076 Tübingen, Germany andreas.zell@uni-tuebingen.de

II. RELATED WORK

Previous work on visually guided takeoff, hovering and landing of UAVs is mainly focused on the landing problem, as it is the most difficult of the three phases. Two main categories of visual landing systems exist for rotorcraft UAVs, being used both on large scale helicopters with a high payload and on MAVs with a very limited payload. The first one is for landing a UAV on a predefined target [20][17][14][23], which requires precise pose estimation of the UAV relative to the target. The second category is for landing on a suitable area [11][3], which uses vision for the detection of a good enough place for landing.

Saripalli et al. [20] solved the landing task of a helicopter quite early by using image moments for object recognition and estimating the relative position to the landing pad with precise height of the helicopter provided by differential GPS. But the proposed approach would hardly work in cluttered or GPS-denied environments. A special landing pad with five circle triplets in different sizes was designed by Merz et al. [17]. And three ellipses in the image, i.e. the projections of three circles of this pad are used for estimating the relative pose in a coarse-to-fine process. Lange et al. [14] achieved autonomous landing and position control of a MAV by estimating the 3D position from a landing pad consisting of several concentric circles, assuming that the UAV is flying exactly parallel to the landing pad and the ground plane. Wenzel et al. [22] presented a low-cost solution for visual tracking a landing place by using a Wii remote infrared camera and four infrared LED markers. In [23], they achieved takeoff, hovering and landing of a MAV on a moving platform. Xu et al. [24] also use a cooperative object and an infrared camera for pose estimation of UAVs, but only the yaw angle is computed.

The core of these vision systems for takeoff, hovering and landing of UAVs with respect to a predefined target is to provide pose estimation with metric scale for the position control of the UAVs. This can actually be done with one circular pad. The geometry of the image projection of a circle, which is an ellipse in the general case, is well studied in the area of projective geometry [10][13][5]. But this does not mean that the pose estimation from the previous work can be directly used for control of autonomous flight applications because there is a common geometric ambiguity in these approaches, where two possible solutions can be obtained, i.e. the absolute solution cannot be achieved from one image projection of one circle. This is similar to the work of Chen et al. [4] for camera calibration, which uses two coplanar circles. Recently, Eberli et al. [7] presented a vision algorithm which is able to estimate the 5 DOF pose of a MAV by using two concentric circles with very different radii. But this algorithm does not work when the camera is in an upright position above the circle mark [7], so the attitude estimation from the IMU is directly used for the 3D position computation in their work.

In our work, we solve the 5 DOF pose estimation problem based on one image projection of one circle, using a projective geometry method with a simple algebraic form. The gravity vector estimated by the IMU is used only as a reference for solving the geometric ambiguity inherited from the projective geometry. Furthermore, we extend this to 6 DOF pose estimation by using the ellipse fitted to the contour of the projected letter "H" for yaw angle computation. The presented algorithm is demonstrated to be sufficient for the control of autonomous takeoff, hovering and landing of a MAV.

III. EXPERIMENTAL SETUP

A. Quadrotor Platform

In this work, we use the open source MAV platform developed by the Pixhawk team at ETH Zürich described in [15]. As shown in Fig. 1, it is a quadrotor equipped with four motors and 10" propellers, enabling it to lift about 400g of payload at a total system weight of about 1.2 kg, including battery. Its onboard computer is a Kontron microETXexpress computer-on-module (COM) featuring an Intel Core 2 DUO 1.86 GHz CPU, 2 GB DDR3 RAM and a 16 Gb SSD. The pxIMU inertial measurement unit/autopilot board we use mainly consists of a MCU and sensors including an accelerometer and a gyroscope. The MCU is a 60 MHz ARM7 microcontroller for sensor readout and fusion as well as position and attitude control. The accelerometer and the gyroscope provide the 3D linear acceleration (±6g) and the 3D angular velocity (±500 deg/s). A PointGrey Firefly MV monochrome camera of only 37g weight, with the resolution of 640×480, a maximum frame rate of 60 fps, and a lens with view angle of 90 degrees, is mounted looking downwards.

B. Landing Pad

The landing pad is printed on an A4 paper for the experiments, as shown in Fig. 1. The radius of the outer boundary of the circle is 90 mm. A larger radius would lead to a larger working distance for the vision system, but would lead to a larger "blind" range, in which the camera cannot observe the circle.

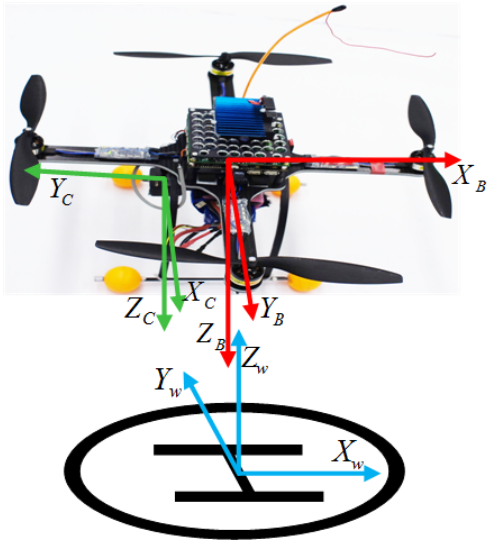


Fig. 1: The quadrotor platform and the landing pad. The corresponding coordinate systems are also plotted.

C. External Tracking System

To measure ground truth data of the 6 DOF pose of the quadrotor, we use an external tracking system, "Optitrack" by Naturalpoint[1], which includes 12 infrared cameras. After we attach six markers to the quadrotor, it can provide 6 DOF pose estimates of the quadrotor with a speed of up to 100 fps. The deviations of position estimates for the static quadrotor is in the order of few millimeters according to our tests.

D. Coordinate Systems and Their Calibration

In general, let $\mathbf{T}_M^N$ be the homogeneous transformation matrix from frame N to frame M, which combines a rotation $\mathbf{R}_M^N$ with a translation $\mathbf{t}_M^N$:

$$\mathbf{T}_M^N = \begin{pmatrix} \mathbf{R}_M^N & \mathbf{t}_M^N \\ \mathbf{0} & 1 \end{pmatrix} \tag{1}$$

The world frame (W), camera frame (C) and quadrotor body frame (B) are defined in the way shown in Fig. 1. W is assumed to be the inertial frame, and the roll, pitch and yaw angles (ϕ, θ and ψ) of the quadrotor are the Euler decomposition of $\mathbf{R}_W^B$ using the Tait-Bryan convention. Due to the symmetric nature of the letter "H", we define the X_W axis along the forward direction of the quadrotor when it starts. In frame C, the origin is the optical center and the Z_C axis is the optical axis of the camera. We assume that the center of gravity of the quadrotor coincides with the center of its mechanical frame.

As the IMU does not directly provide position information, we assume the origin of the IMU frame (U) to coincide

[1] http://www.naturalpoint.com/optitrack/products/tracking-tools-bundles

with the body frame (B). If the IMU was perfectly mounted, U would completely coincide with B. This is usually not the case, however, because it is impossible to mount the IMU with perfect accuracy. The actual rotation between IMU frame and body frame is calibrated by measuring the normalized gravity vector $\mathbf{g}_U$ using the accelerometer when the quadrotor is placed on a horizontal ground plane, so that the gravity vector should be parallel to the Z_B axis, which means $\mathbf{g}_B = (0,\ 0,\ 1)^T$. The rotation $\mathbf{R}_B^U$ can be found as the shortest rotation that satisfies $\mathbf{g}_B = \mathbf{R}_B^U \mathbf{g}_U$.

We obtain $\mathbf{T}_B^C$ by calibrating the corresponding two frames with respect to another external frame (E). This frame E is fixed on a planar calibration pattern as used in the camera calibration Matlab toolbox by Bouguet [1]. We place both the calibration pattern and the quadrotor within the view field of the external tracking system and put markers on both of them. Then $\mathbf{T}_W^B$ and $\mathbf{T}_W^E$ can be obtained from both of their 6 DOF poses measured by the tracking system. In order to obtain $\mathbf{T}_E^C$ we perform extrinsic parameter calibration of the camera using the Matlab toolbox mentioned above. The missing transform $\mathbf{T}_B^C$ can finally be computed as:

$$\mathbf{T}_B^C = \left(\mathbf{T}_W^B\right)^{-1} \mathbf{T}_W^E \mathbf{T}_E^C \tag{2}$$

IV. VISION ALGORITHM

Robustness and real time performance are important issues for the onboard vision algorithm of a MAV, which usually have to be traded off against each other. In this paper, we aim at achieving both of them, as well as accurate pose estimation of the quadrotor. With the landing pad being chosen, the idea of our vision algorithm is straightforward, which mainly consists of landing pad recognition, ellipse fitting, and 6 DOF pose estimation from ellipses.

A. Landing Pad Recognition

Most regular landing pads are usually not particularly textured, whereas their background might be very cluttered. This is a difficult scenario for feature-based object detection methods, as these will find only few interest points on the landing pad itself compared to other objects in the background. Since our landing pad consists of the letter "H" surrounded by a circle printed in black on white background, we decided to treat this as a sign detection problem and solve it similarly as in [21] by binarization of the camera image, finding connected components, and then classifying connected components using an artificial neural network. The results of applying different steps are illustrated in Fig. 2. This allows us to detect the landing pad in real-time even on computationally constrained hardware.

1) Binarization: We use adaptive thresholding to binarize each camera image and allow for different lighting conditions in different parts of the image. This can be computed efficiently using the integral image for fast computation of average pixel values of the window surrounding each pixel.

2) Extracting Connected Components: We extract connected components of the possible patterns based on the run-based two-scan labeling algorithm by He et. al. [12], disregarding connected components that are too small to be reliably classified. After that we end up with a set of connected components which might be parts to the landing pad.

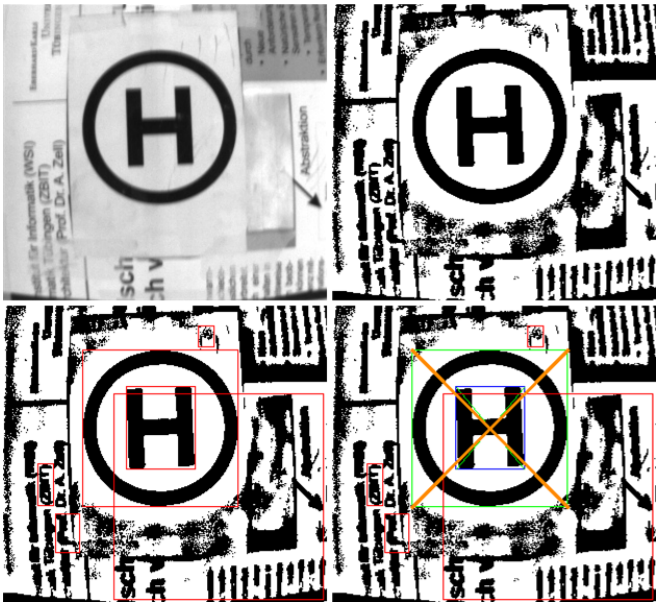


Fig. 2: Different steps of landing pad recognition illustrate: the original image (top left), the binary image (top right), connected components labelled with their bounding boxes (bottom left), and the classification result (bottom right).

3) Classification of Connected Components: We classify each of the connected components detected in the image using an artificial neural network. This neural network assigns one of three classes to each connected component: *Circle*, *letter "H"* or *other*. As shown in Fig. 2, the circles and the letter "H"s are marked with green and blue bounding boxes, respectively, and other objects with red ones. We implement this by using the 1-of-3 target coding scheme, which requires one output neuron per class: For each input sample, we expect the output neuron corresponding to its true class to return 1 and the others to return 0.

The structure of the neural network is a multilayer perceptron with 196 input units (one per pixel of patterns resized to 14×14), only one hidden layer consisting of 20 hidden units and three output units. All units use the logistic activation function, except the output layer which uses the softmax activation function so we can interpret the output value of each output unit as the posterior probability of the input patch belonging to its corresponding class. We used standard backpropagation to train the network, based on a labeled dataset containing approx. 7,000 samples of background clutter and 3,000 samples each of both parts of the landing pad (circle and letter "H"), taken from different perspectives. After training the network, we can generate efficient C code using the snns2c module of Stuttgart Neural Network Simulator (SNNS) [25].

4) Enforcing Geometric Relationship Constraint: Once all connected components are classified by the neural network, we can suppress false positives by enforcing the geometric constraint that each letter "H" which is part of our landing pad has to be surrounded by a circle. We can therefore disregard all connected components classified as letter "H"s that do not lie within the bounding box of a connected component classified as a circle. The relative sizes and center positions of their bounding boxes are considered in this constraint. Let $p_1,\ p_2$ be the false positive rates of the circle and the letter "H" detections. Due to this geometric consistency check, the false positive rate for the final landing pad detection will be lower than $\min(p_1,\ p_2)$, usually by a large amount. In fact, we seldom encountered a single false positive of the landing pad detection during our flight experiments.

If our system at this point is still certain that it has detected the landing pad, it extracts the corresponding gray scale pattern for further processing. In Fig. 2, the finally detected landing pad is marked with an orange cross.

B. Ellipse Fitting

In the general case, the perspective projection of a circle is an ellipse. To make our vision system reliable in a large range of perspective views, this general case is considered in our work. Thus, accurate ellipse fitting turns out to be a critical issue for the later geometric computation. In this section, we obtain the ellipses corresponding to the inner and outer boundary of the landing pad circle and that of the letter "H". It is achieved by performing edge detection on the gray scale image pattern corresponding to the landing pad and fitting ellipses to these edges. Lens distortion effect to the ellipse parameters is taken into account during this process.

First, the well known Canny edge detector [2] is applied to the gray scale image pattern. The edge contours retrieved from the detected edges are then used to fit the ellipses by implementing a so called direct least square fitting algorithm [9], which is extremely robust to image noise and efficient. A comprehensive comparison of this algorithm with some other ellipse fitting algorithms can be found in [9]. An implementation of this algorithm exists in OpenCV [19].

Due to lens distortion, especially when using a wide angle lens, the assumption of perspective projection no longer applies. To eliminate the effect of this, we apply a correction step to the edge contours before applying the ellipse fitting algorithm: We transform the edge contour from the image frame into an undistorted image frame. For this step, the camera model in [26] with only the first two terms of radial distortion being considered is adopted, and Newton's iterative method is used to calculate this projection with known intrinsic camera parameters. To further improve the efficiency, a look-up table mapping distorted image coordinates to undistorted image coordinates is precomputed at the initialization phase of the vision system.

Even though we can detect two ellipses for the circle of the landing pad in this paper, we only use the ellipse corresponding to its outer boundary for further pose estimation. Fitting an ellipse to the projected contour of the letter "H" provides us with the orientation of the landing pad, which will be described in Sect. IV-C.3.

C. 6 DOF Pose from Ellipses

In this paper, we use one single ellipse originating from the projection of a known circle for the 5 DOF pose estimation of the quadrotor within the world frame (W), including its 3D position $\mathbf{t}_W^B$ and the roll and pitch angles ($\phi,\ \theta$) of the quadrotor. ϕ and θ are derived from the normal vector of the plane on which the circle lies. Chen et al. [4] also described this problem for camera calibration with two arbitrary coplanar circles. We briefly introduce it for our pose estimation here, and then use an IMU-aided approach to resolve the ambiguity inherited from this problem. The yaw angle of the quadrotor (ψ) is estimated as the orientation of the major axis of the ellipse fitted to the projection of the letter "H".

1) 5 DOF Pose From One Ellipse: Once the effect of lens distortion has been corrected, we can consider the vision geometry to obey perspective projection, in which a pinhole camera model applies. An ellipse in the image frame can then be described by the following quadratic equation,

$$Ax^2 + 2Bxy + Cy^2 + 2Dx + 2Ey + F = 0, \tag{3}$$

or if we define the augmented vector $\mathbf{X} = (x,\ y,\ 1)^T$, we get

$$\mathbf{X}^T \begin{bmatrix} A & B & D \\ B & C & E \\ D & E & F \end{bmatrix} \mathbf{X} = 0. \tag{4}$$

Let f be the focal length of the camera, we can define the image plane to be at $z = f$. A bundle of straight lines passing through the optical center and the ellipse define an oblique elliptical cone, of the form

$$\mathbf{P} = k(x,\ y,\ f)^T, \tag{5}$$

where k is a scale factor describing the distance from the origin to $\mathbf{P}$. Combining (4) and (5), the equation to describe the oblique elliptical cone is:

$$\mathbf{P}^T \mathbf{Q} \mathbf{P} = 0, \tag{6}$$

where

$$\mathbf{Q} = \begin{bmatrix} A & B & \frac{D}{f} \\ B & C & \frac{E}{f} \\ \frac{D}{f} & \frac{E}{f} & \frac{F}{f^2} \end{bmatrix} \tag{7}$$

is called a conic in [13].

We directly derive the results for 5 DOF pose of the circle in the camera frame from the conic $\mathbf{Q}$. Detailed proof of this result can be found in [13] and [4]. Let r be the radius of the original circle, which is projected as the ellipse, λ_1, λ_2, and λ_3 be the eigenvalues of $\mathbf{Q}$, and $\mathbf{u}_2$ and $\mathbf{u}_3$ the unit eigenvectors for eigenvalues λ_2 and λ_3, respectively. As $\mathbf{Q}$ has a signature of (2, 1) [13], without loss of generality, we can assume that $\lambda_3 < 0 < \lambda_1 \leq \lambda_2$. Then following the world frame definition in III-D, the unit vector of the Z_W axis and

the origin of the world frame described in the camera frame (denoted by $\mathbf{n}$ and $\mathbf{t}_C^W$) are given by

$$\mathbf{n} = S_1 \sqrt{\frac{(\lambda_2 - \lambda_1)}{(\lambda_2 - \lambda_3)}}\mathbf{u}_2 + S_2 \sqrt{\frac{(\lambda_1 - \lambda_3)}{(\lambda_2 - \lambda_3)}}\mathbf{u}_3, \tag{8}$$

$$\mathbf{t}_C^W = z_0 \left(S_1 \lambda_3 \sqrt{\frac{(\lambda_2 - \lambda_1)}{(\lambda_2 - \lambda_3)}}\mathbf{u}_2 + S_2 \lambda_2 \sqrt{\frac{(\lambda_1 - \lambda_3)}{(\lambda_2 - \lambda_3)}}\mathbf{u}_3 \right), \tag{9}$$

where $z_0 = S_3 \frac{r}{\sqrt{-\lambda_2 \lambda_3}}$, and S_1, S_2 and S_3 are undetermined signs. (8) and (9) give us a clear algebraic formula for the 5 DOF pose estimation. As $\mathbf{n}$ faces to the camera, and the center of the circle is in front of the camera in our definition, we can get two constraints for the undetermined signs,

$$\mathbf{n} \cdot (0,\ 0,\ 1)^T < 0, \tag{10}$$

$$\mathbf{t}_C^W \cdot (0,\ 0,\ 1)^T > 0. \tag{11}$$

Only two of these three signs can be determined by (10) and (11), so two possible solutions for $\mathbf{n}$ and $\mathbf{t}_C^W$ remain. When $\lambda_1 = \lambda_2$, only one solution exists. In general case, let us denote these two solutions to be $\mathbf{n}_1$, $\mathbf{t}_1$ and $\mathbf{n}_2$, $\mathbf{t}_2$. Further disambiguation is needed to obtain the absolute 5 DOF pose estimation.

2) Resolving the Ambiguity: The gravity vector described in the camera frame can also be calculated from the roll and pitch angles (ϕ and θ) estimated by the IMU. We use it as a reference to resolve the geometric ambiguity.

Two assumptions are introduced for the disambiguation step: The error of the attitude estimates by the IMU is small, and the landing pad is placed horizontally oriented. Since the attitude estimates by the IMU is used for high frequency attitude control, the first assumption should be met, otherwise a MAV could not fly properly. Fortunately, this assumption does apply to our IMU and most commercial ones. In this case, the error of the gravity vector estimation would be small correspondingly. Also, we usually want a MAV to land on level ground. Therefore, it is reasonable to make the second assumption.

Following the second assumption, the gravity vector $\mathbf{g}$ would be antiparallel to the Z_W axis. While estimating the 5 DOF geometry, we assume the yaw angle to be $\psi = 0$. Then the rotation matrix from frame B to frame W can be derived as,

$$\mathbf{R}_{W1}^B = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi & \cos\phi \end{bmatrix}. \tag{12}$$

Let $\mathbf{z}_W$ be the unit vector of the Z_W axis, then in frame B, $\mathbf{z}_W$ can be expressed as

$$\mathbf{z}_B^W = \left(\mathbf{R}_{W1}^B\right)^{-1} \cdot \mathbf{z}_W.$$

In the camera frame C, we have

$$\mathbf{z}_C^W = \mathbf{R}_C^B \cdot \mathbf{z}_B^W. \tag{13}$$

$\mathbf{z}_C^W$ is used as the final reference vector for resolving the ambiguity of the 5 DOF pose estimation according to the angles of the vectors $\mathbf{n}_1, \mathbf{n}_2$ to $\mathbf{z}_C^W$, denoted by θ_1 and θ_2. According to our assumptions, the correct vision measurement should be close to the IMU measurement. So we choose the vector with smaller angle relative to $\mathbf{z}_C^W$ to be the final estimate of the unit vector $\mathbf{n}$, which means

$$\mathbf{n} = \begin{cases} \mathbf{n}_1 & \text{if } \theta_1 < \theta_2 \\ \mathbf{n}_2 & \text{if } \theta_1 > \theta_2. \end{cases} \tag{14}$$

Thus, the last undetermined sign in (8) and (9) is now determined, and the 5 DOF pose can be derived.

It should be noted that if the error of the gravity vector estimated from the IMU is higher than the angles spanned by the two possible solutions to the true gravity vector, the disambiguation may fall to the false result according to (14). But as predicted by our assumptions, this seldom happened during our experiments. And if this ever results in a large jump between the previous and current estimate, the current one can simply be disregarded as an outlier.

3) Yaw Angle From the Letter "H": For yaw angle (ψ_C) estimation, previous methods use image moments to estimate the orientation of a letter "H", as described in [20]. In our work, the orientation of the major axis of the ellipse fitted to the letter "H" is used as an approximation. We tested these two methods by using a synthetic image of a letter "H", and rotating it with a step size of one degree. Errors of less than 3 degrees were achieved by using the ellipse fitting method in this case, and less than 1 degree for the image moments method. But when fitting the ellipse to a solid rectangle, even smaller errors were achieved. The errors from both these two methods would be larger in real applications with noise and perspective projection. But we adopt the ellipse fitting approach because of the following advantages: Additional computational cost can be avoided, and we can unify the 6 DOF pose estimation by using the ellipse fitting method. Furthermore, we can use other patterns with rectangular shape to replace the letter "H" as long as we train the neural network with this new pattern, making the vision algorithm more flexible. This approach is demonstrated to be sufficient for controlling the yaw angle of the quadrotor in our experiments.

Due to the symmetric nature of the letter "H", the yaw angle of the quadrotor has two solutions, with a difference of 180°. As this is inherited from the symmetric configuration of the landing pad, and does not effect the autonomous flight, we simply ignore this issue and assume $-90° < \psi_C < 90°$.

4) 6 DOF Pose of the Quadrotor: Until now, the 6 DOF pose estimated is describing the world frame (W) in the camera frame (C). The 6 DOF pose of the quadrotor in W is obtained by performing the following transforms.

Let $\mathbf{R}_n^C$ be the rotation matrix transforming the vector $\mathbf{n}$ in (14) to the Z_C axis, which can be calculated by using Rodrigues formula [8]. From $\mathbf{R}_n^C$, ψ_C and (9), we get the 6 DOF pose of the camera related to the world frame in a form of

$$\mathbf{R}_W^C = \begin{bmatrix} \cos\psi_C & -\sin\psi_C & 0 \\ \sin\psi_C & \cos\psi_C & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \mathbf{R}_n^C, \tag{15}$$

$$\mathbf{t}_C^W = z_1 \left(\lambda_3 \sqrt{\frac{(\lambda_2 - \lambda_1)}{(\lambda_2 - \lambda_3)}} \mathbf{u}_2 + S\lambda_2 \sqrt{\frac{(\lambda_1 - \lambda_3)}{(\lambda_2 - \lambda_3)}} \mathbf{u}_3 \right), \quad (16)$$

where $z_1 = S' \frac{r}{\sqrt{-\lambda_2 \lambda_3}}$, and S, S' are determined by (10), (11) and (14).

Finally we get the 6 DOF pose of the quadrotor in the world frame as follows,

$$\mathbf{R}_W^B = \mathbf{R}_W^C \cdot \mathbf{R}_C^B, \quad (17)$$

$$\mathbf{t}_W^B = -\mathbf{R}_W^B \cdot \left(\mathbf{R}_B^C \cdot \mathbf{t}_C^W + \mathbf{t}_B^C \right). \quad (18)$$

$\mathbf{t}_W^B$ is the 3D position of the quadrotor in the world frame, and the rotation matrix $\mathbf{R}_W^B$ gives the three individual roll, pitch and yaw angles of the quadrotor, which can be calculated by the Euler decomposition of $\mathbf{R}_W^B$ [6].

V. FLIGHT CONTROL ALGORITHM

Michael and Mellinger et al. [18][16] developed a nested controller which consists of an attitude and a position controller, achieving precise hovering and 3D trajectory control based on a basic dynamic model of the quadrotor and using accurate 6 DOF pose estimation from an external tracking system. To prove our vision algorithm, we used a very similar nested PID controller, which is implemented in the pxIMU by Pixhawk, for hovering control of our quadrotor. In our case, we set the desired yaw angle to a constant value ($\psi^{des} = 0$). The 3D position estimates from the onboard vision system is used as feedback to the position controller after applying a basic Kalman Filter without taking the IMU information into account. Since the IMU can provide roll and pitch estimates with a frequency of 200 Hz, much higher than that of the onboard vision system, we use these estimates provided by the IMU for attitude control, and only the yaw angle is provided by the onboard vision system.

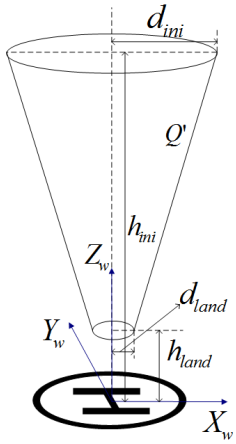


Fig. 3: The satisfactory truncated cone of the desired trajectory.

We designed a point setting approach for takeoff and landing of the quadrotor. The desired trajectory is defined to be along the Z_W axis for both phases. The same nested PID controller is then used for trajectory following. For the landing phase, the set point is initialized as $\mathbf{P}_{set} = (0,\ 0,\ h_{ini})^T$. We define a satisfactory truncated cone Q' as shown in Fig. 3. If the quadrotor stays within Q' with an height error $e_h < h_{err}$ for a duration $t_{sat} > t_0$, we assume that the current set point is reached, and $\mathbf{P}_{set}$ is decreased by a constant vector $\left(0,\ 0,\ h_{step}\right)^T$. This process is repeated until the point $\mathbf{P}_{set} = (0,\ 0,\ h_{land})^T$ is reached, where the quadrotor can be safely landed by blindly powering down the motors. For the takeoff phase, the target height is simply increased at a constant rate until the quadrotor reaches the desired hovering height.

VI. EXPERIMENTS AND RESULTS

In this section, the performance of our vision system and the control algorithm for takeoff, hovering and landing is evaluated. The tracking system mentioned in Sect. III-C is used for providing ground truth data of the 6 DOF pose of the quadrotor during indoor experiments.

A. Landing Pad Recognition and Ellipse fitting

Fig. 4 shows an exemplary image processing result for landing pad recognition, with the same color labels as in Fig. 2. Besides the landing pad, we use some other circles and letter "H"s with different orientations and stretched shapes attached to some posters which are rich of texture features. All our latter indoor experiments are done in the same cluttered environment. It shows that different circles and letter "H"s can be efficiently detected even if the perspective changes dramatically. False positives for the individual circle class or letter "H" class may appear, but would not be finally classified as a landing pad.

Fig. 5 shows the results of fitting ellipses to the corresponding gray scale image patches depicted above, detected from various perspectives. Some of them clearly exhibit the effect of motion blur. The major and minor axis of the ellipses are also plotted. The orientation of the major axis fitted to the letter "H" provides an approximation for the yaw angle of the camera.

B. 6 DOF pose from Ellipses

We compare the 6 DOF pose estimates of the onboard vision system with ground truth data from the tracking system, both recorded at a frequency of 60Hz. The 3D position and the roll, pitch and yaw angles are compared separately.

1) Hand-held Case: First, we manually rise and hover the quadrotor above the landing pad so that a large range of perspective changes of the camera can be tested. As shown in Fig. 6, estimates of the onboard vision system are plotted in red line, and ground truth data in green. The 5 DOF onboard vision pose estimates are well in line with ground truth data, without many obvious outliers, even though the position and attitude of the quadrotor changes in a large range. When the landing pad gets further away from the camera, its image projection will get smaller respectively, and image noise will cause larger deviations to the pose estimates, which can be

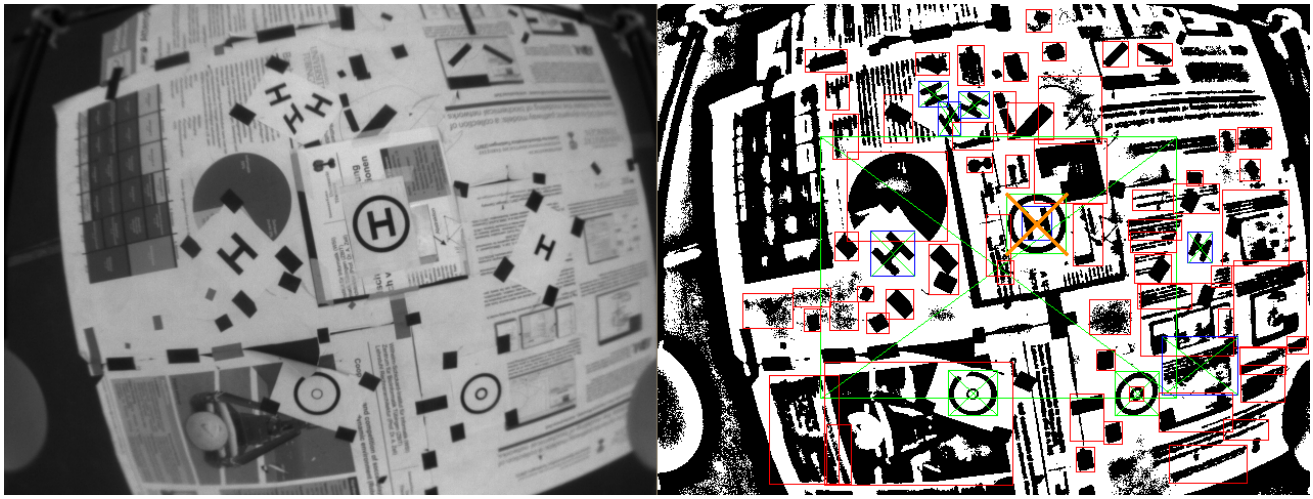


Fig. 4: Landing pad recognition results when the quadrotor hover above the landing pad in cluttered environment.

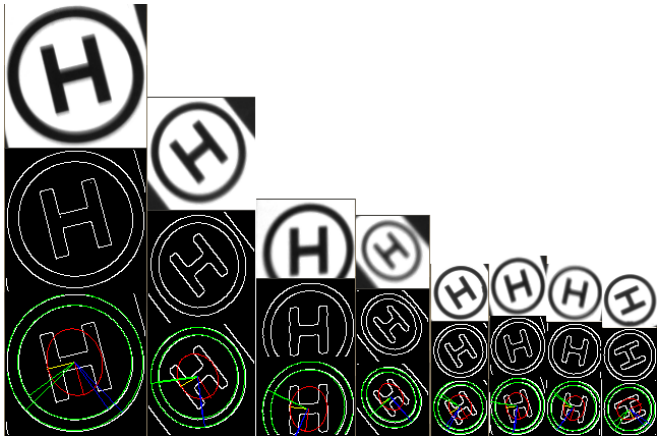


Fig. 5: Image patches of the landing pad from different perspectives and the corresponding edge detection and ellipse fitting results.

found in Fig. 6 when the quadrotor was hovering at a height of around 1.5 meters.

Deviations of the yaw angle are larger than those of the roll and pitch angles, especially when the quadrotor is at poses where the image projections of the letter "H" are warped much. This is because we use the approximation method as described in Sect. IV-C.3, where only the rotation of the letter "H" is considered. Since the IMU, the onboard vision system and the tracking system perform very similarly for the roll and pitch estimation, to clearly demonstrate the performance of the onboard vision system, we omit those estimates provided by the IMU in the figures, even though they are required by both the vision algorithm and attitude control of the quadrotor.

In Fig. 6, outliers of the ground truth data are mainly caused by human occlusion to the markers. We initialize the onboard vision pose estimates with the 3D position $(0,\ 0,\ 300)^T$ and the identity matrix for the rotation matrix,

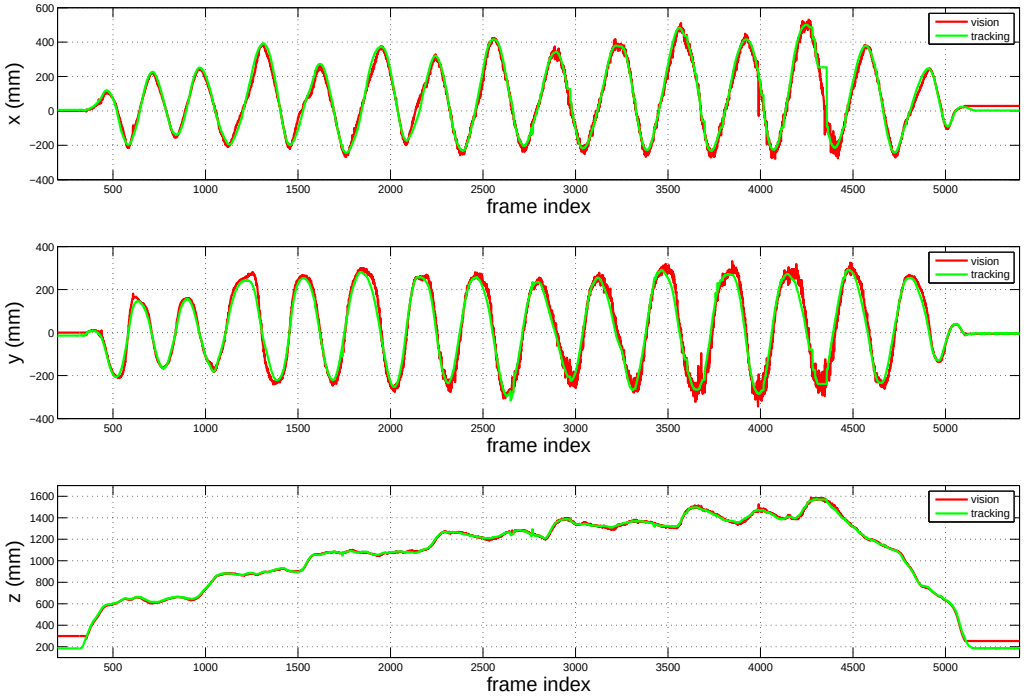
TABLE I: RMSEs in different cases.

	Hand-held	Hovering	Auto
$X_W Y_W$ RMSE (mm)	43.1	38.8	33.7
Z_W RMSE (mm)	7.9	5.7	6.8
3D RMSE (mm)	43.8	39.2	34.4
ϕ RMSE (deg)	1.4	1.3	1.4
θ RMSE (deg)	1.2	1.4	1.5
ψ RMSE (deg)	7.2	4.8	2.7

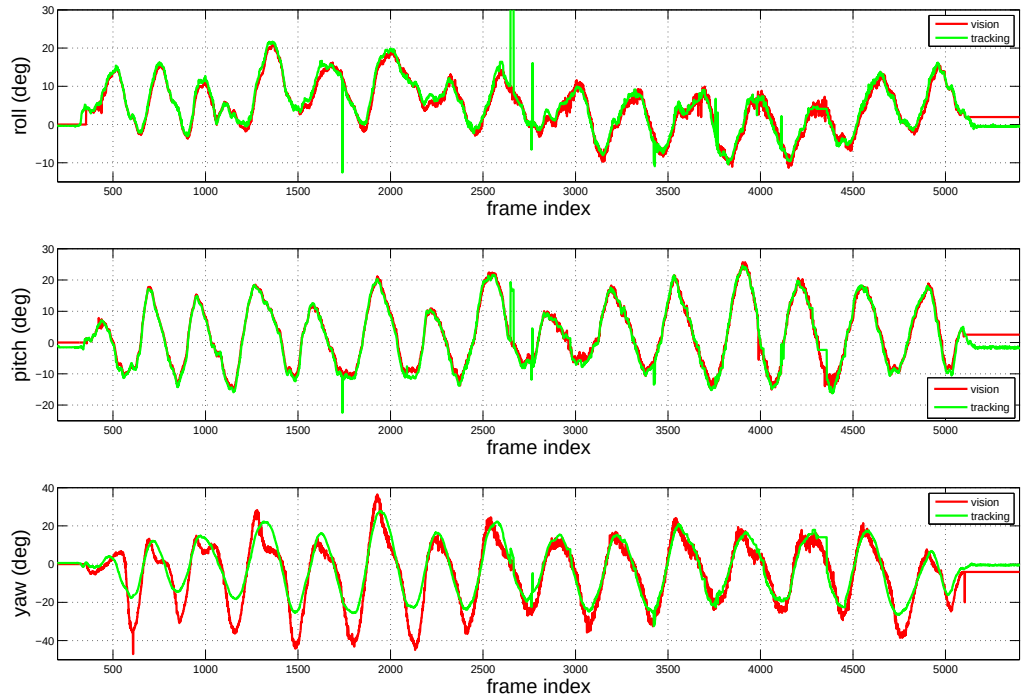
as the circle is not fully visible when the camera is very close to it. We compute the root-mean-square errors (RMSEs) of the onboard 6D pose estimation for the whole trajectory by comparing them with the ground truth data. The raw of 3D RMSE in Table I is for the distance of the onboard vision 3D position estimates to the ground truth data, and $X_W Y_W$ is for the distance on the $X_W Y_W$ plane.

2) Hovering Case: Fig. 7 shows an autonomous hovering flight with a set point of $(0,\ 0,\ 1000)^T$ (mm) for about 66 seconds. More motion blur from the vibration of the quadrotor is introduced in real flight, which will increase errors to the pose estimates. We still manually added disturbances to the control command to let the quadrotor hover in a larger area, so that a relatively large range of perspectives could be tested in this case. Compared to the hand-held case, the performance is not much worse and the RMS error is even smaller. This is not surprising as in autonomous hovering flight, the three Euler angles are normally very small, and the quadrotor usually will not reach such poses where large deviations may be introduced.

3) Autonomous Flight: A full trajectory of the quadrotor during an autonomous taking off, hovering and landing flight is shown in Fig. 8. In this flight, it took the quadrotor about 6 seconds to reach the set point for hovering state. After 5 seconds of hovering around set point $(0,\ 0,\ 1000)^T$ (mm), it started the landing phase, which took about 3 seconds to

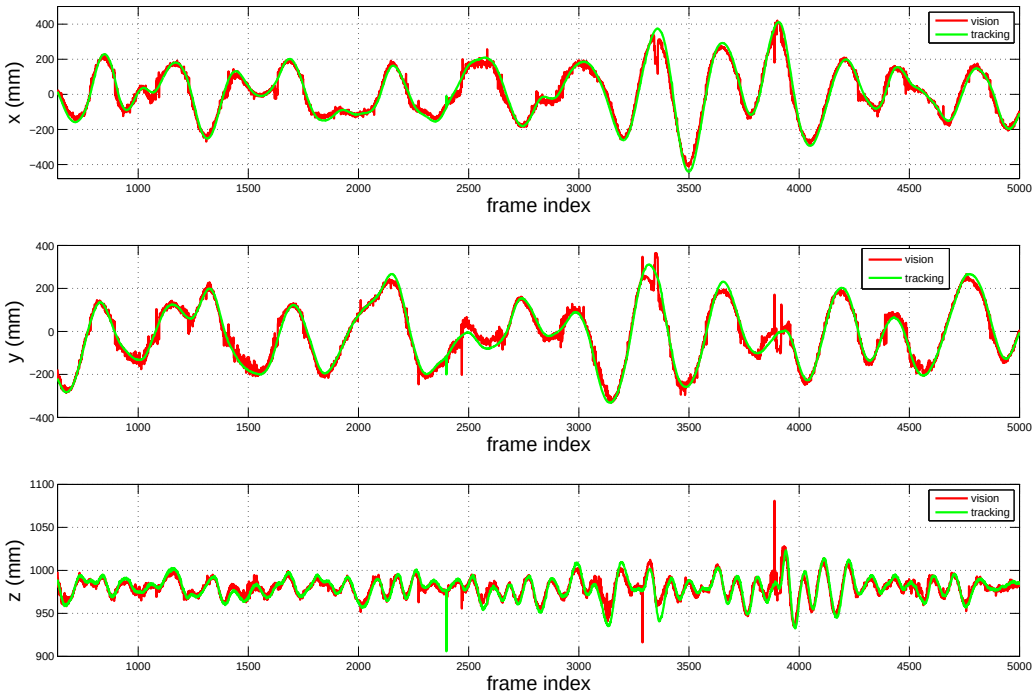


(a)

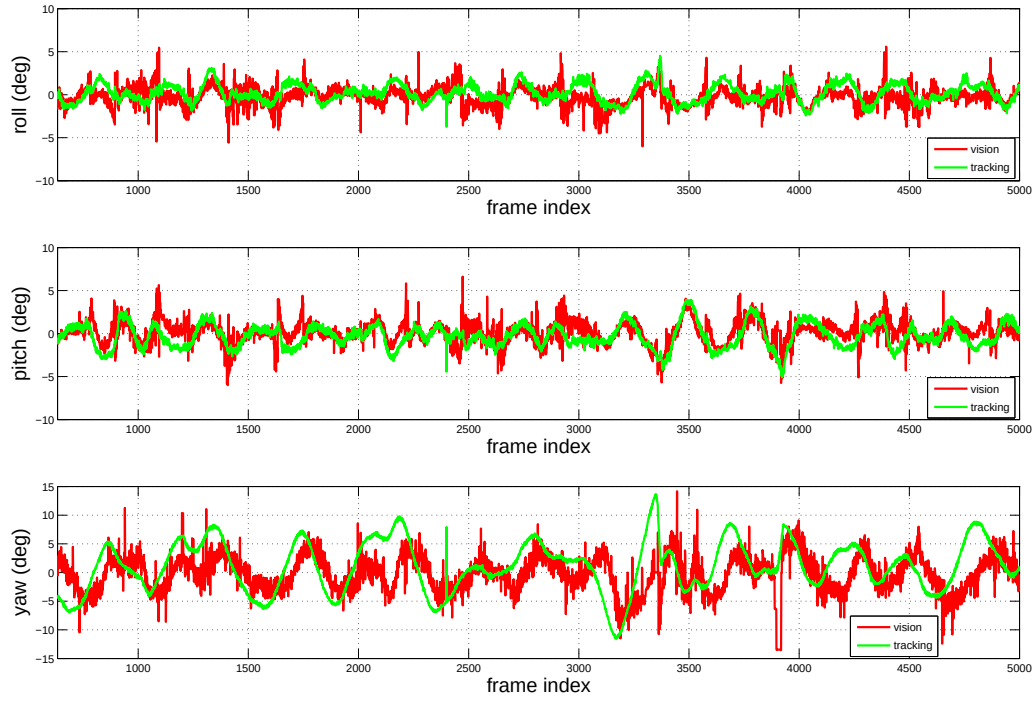


(b)

Fig. 6: (a) Position and (b) attitude estimates with hand-held quadrotor.

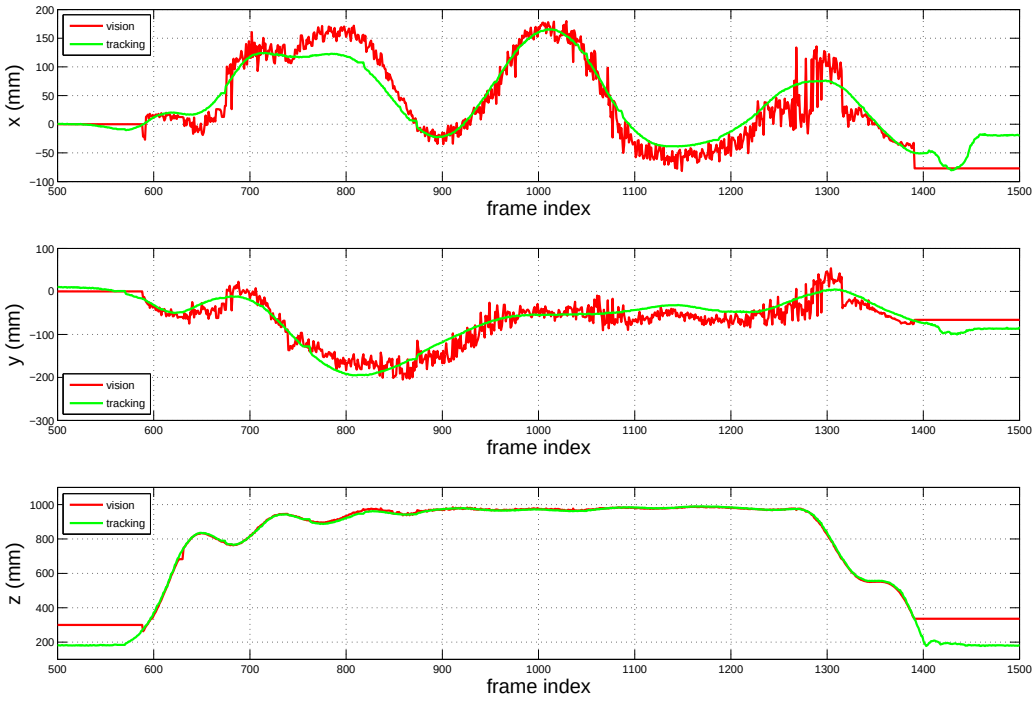


(a)

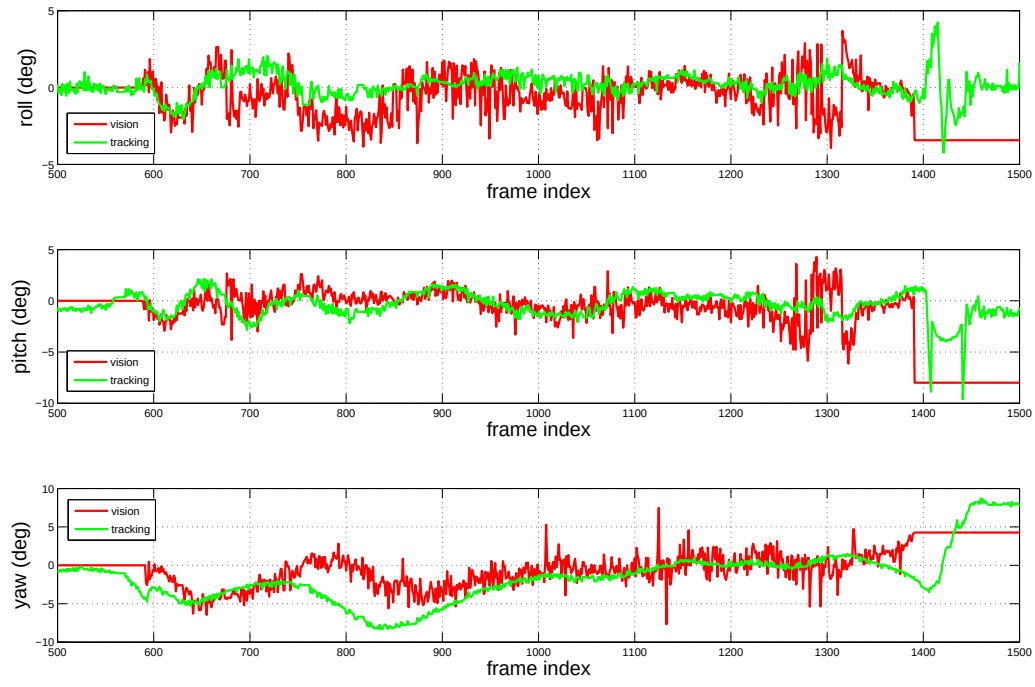


(b)

Fig. 7: (a) Position and (b) attitude estimates during a hovering flight with remotely added disturbances to the controller.



(a)



(b)

Fig. 8: (a) Position and (b) attitude estimates during an autonomous takeoff, hovering and landing flight.

land on the landing pad. For the start of the takeoff phase, the quadrotor just ascends with open loop control until it observes the circle in the landing pad. Based on ground truth data of the final states of the quadrotor, the deviations of final landing positions can be found. In our tests of 10 continuous landing flights, the mean deviation of the position on X_W, Y_W and yaw angle are about $(24,\ 86)^T$ (mm) and 6 degrees. The deviations are partially caused by the "blind" range of the camera, which makes the landing not soft enough for the mechanism of the quadrotor landing gear.

4) Computation Time: The computation time of our vision system during the flight in Fig. 8 is shown in Fig. 9. The main part of it comes from the landing pad recognition phase, which has an average cost of less than 10 ms and a maximum of about 18 ms. Peaks in Fig. 9 may be mainly caused by the performance of the onboard computer, as they did not occur when we tested video logfiles on an off-board PC. While hovering on the set point $(0,\ 0,\ 1000)^T$ (mm), the geometry computation time, including edge detection, ellipse fitting and the 6 DOF pose computation, is around 1 ms. When the landing pad gets very close to the camera, its image projection may nearly occupy the whole image, which causes a longer time for geometry computation. When the quadrotor hovers above the landing pad at a height of about 300 mm, the geometry computation time reaches a maximum of about 11 ms. The average time cost of the vision algorithm is less than 11 ms/frame, making full use of our 60 fps camera.

5) 5 DOF Accuracy Evaluation: We evaluate the 5 DOF RMSE of the quadrotor pose estimated by the onboard vision system at different distances to the landing pad and attitudes. The results are shown in Fig. 10. We manually fix the quadrotor above the landing pad and record the onboard estimates and ground truth data from the tracking system at each pose for about 10 seconds, i.e. 600 frames, and calculate the RMSEs of these measurements. Since roll and pitch angles have the same effect to pose estimation in our geometrical method, we set the pitch angle of the quadrotor to zero degrees and only change its roll angle. The yaw angle is set to be about zero degree. The optical axis of the camera nearly coincides with the Z_B axis, with a small tilt in pitch angle. All onboard pose estimates for this evaluation are obtained within the same system configuration, which means that the systematic deviation with respect to ground truth data is constant throughout this experiment.

Fig. 10a demonstrates that the RMSE of the 3D position estimates grows with increasing distance to the landing pad and increasing roll angle. When the roll angle is nearly zero, the two possible solutions mentioned in Sect. IV-C.1 are very close to each other, which may cause our disambiguation method to fail and result in larger deviations. This can explain why the RMSEs for $roll = 0$ may exceed those for $roll = 10$. Fig. 10b shows that there is no obvious relationship between the attitude angle and its RMSE even if it increases to up to 40 degrees, and its RMSE tends to grow with the increase of the working distance. Attitude estimates of our vision system are overall very accurate, with RMSEs below 1.5 degrees for all tested poses. We do not expect our ground truth data to be much more accurate than this, which could explain why the decrease in accuracy for higher distances and angles in fig. 10b is not so obvious.

C. Outdoor Autonomous Flight

In our outdoor experiment, we locate the landing pad on grass with some other objects beside it, like a small path, in cloudy and windless weather. In such scenario, the background of the landing pad is still textured, and the assumption of the landing pad being parallel to the ground plane in Sect. IV-C.2 is also not strictly satisfied. The autonomous flight achieved similar overall performance as in the previous indoor scenario. Fig. 11 shows the position estimation of the onboard vision system during a successful takeoff, hovering and landing outdoor flight.

VII. CONCLUSIONS

We have presented an onboard vision system that can detect a landing pad consisting of the letter "H" surrounded by a circle, from images captured by a monocular camera on a MAV and determine the 6 DOF pose of the MAV relative to the landing pad using projective geometry.

Our algorithms are computationally efficient enough to process up to 60 frames per second on our onboard computer. We have shown that the whole system produces robust and accurate pose estimates, which were evaluated using an external tracking system. We used these pose estimates to enable a completely autonomous helicopter to reliably start from, hover above, and land on the landing pad, even if this is located within a challenging environment, i.e. in front of a visually cluttered background.

A video demonstrating our autonomous quadrotor flying using this vision system can be found online[2].

In future work, we plan to use the prestented vision system at the beginning and at the end of autonomous flights of our MAVs. We also want to use the pose estimates produced by this system to initialize the otherwise unknown scale factor of a monocular visual SLAM system. It may be interesting to investigate whether sensor fusion of both attitude estimates provided by the IMU and our vision system can further improve the attitude-estimation accuracy.

VIII. ACKNOWLEDGMENTS

The authors would like to thank the Pixhawk team at ETH Zürich for the release of the open source Pixhawk quadrotor platform, and Prof. Andreas Schilling from the computer graphics group in our faculty for providing us with the access to the tracking system. We would also like to thank Karl E. Wenzel in our group for the discussions about this paper.

REFERENCES

[1] J.Y. Bouguet, "Camera Calibration Toolbox for Matlab", *http://www.vision.caltech.edu/bouguetj/calib_doc*, 2001.

[2] J. Canny, "A computational approach to edge detection", *IEEE Transactions on Pattern analysis and Machine Intelligence*, Vol. 8(6), 1986, pp. 679-698.

[2] http://www.youtube.com/watch?v=yvyzvttuNsQ

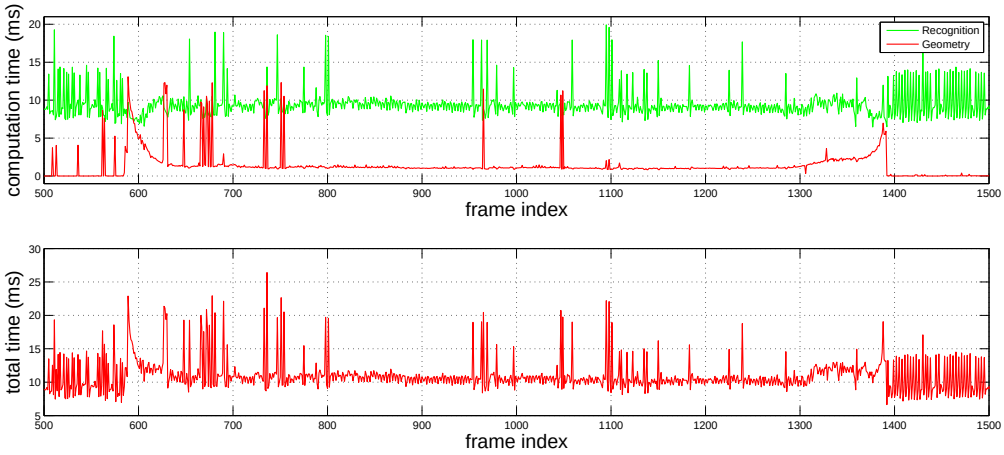
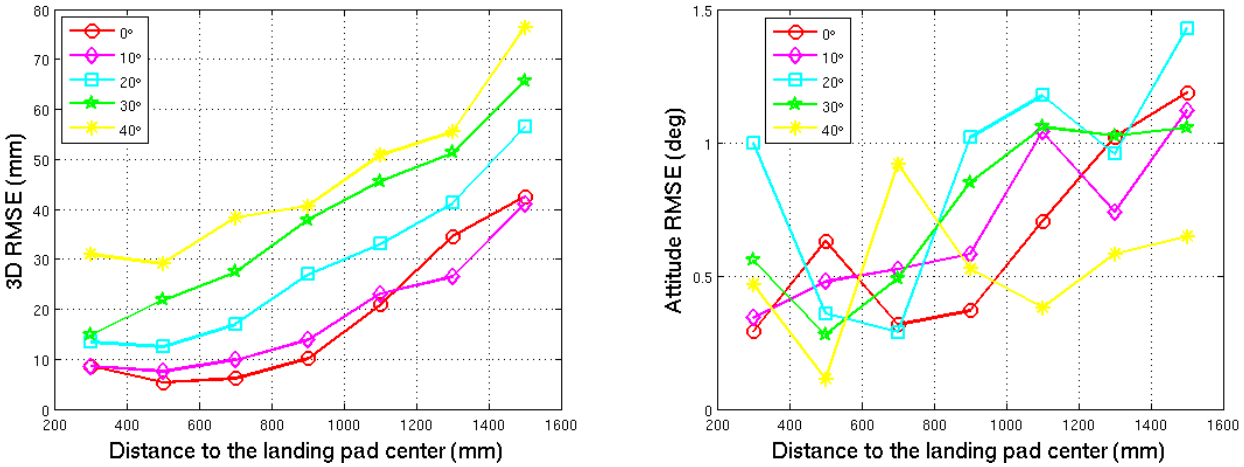


Fig. 9: Computation time during the autonomous takeoff, hovering, and landing flight.



(a) (b)

Fig. 10: (a) 3D Position RMSE and (b) attitude RMSE within different distances and attitudes to the landing pad.

[3] A. Cesetti, E. Frontoni, A. Mancini, P. Zingaretti, S. Longhi, "A Vision-Based Guidance System for UAV Navigation and Safe Landing using Natural Landmarks", *Journal of Intelligent & Robotic Systems*, Vol. 57, No. 1-4, 2010, pp. 233-257.

[4] Q. Chen, H. Wu, T. Wada, "Camera Calibration with Two Arbitrary Coplanar Circles", *ECCV-2004, LNCS*, Vol. 3023/2004, 2004, pp. 521-532.

[5] Z. Chen, and J. B. Huang, "A vision-based method for the circle pose determination with a direct geometric interpretation", *IEEE Transactions on Robotics and Automation*, Vol. 15(6), 1999, pp. 1135-1140.

[6] J. Diebel, "Representing attitude: Euler angles, unit quaternions, and rotation vectors", *Technical report, Stanford University*, Stanford, California 94301-9010, October 2006.

[7] D. Eberli, D. Scaramuzza, S. Weiss, R. Siegwart, "Vision Based Position Control for MAVs Using One Single Circular Landmark", *Journal of Intelligent & Robotic Systems*, Vol. 61, No. 1-4, 2011, pp. 495-512.

[8] O. Faugeras, *Three-Dimensional Computer Vision: a Geometric Viewpoint*, MIT Press, 1993.

[9] Fitzgibbon, A., Pilu, M., Fisher, R.B., "Direct least square fitting of ellipses", *IEEE Transaction on Pattern Analysis and Machine Intelligence*, Vol. 21(5), 1999, pp. 476-480.

[10] D. Forsyth, J.L. Mundy, A. Zisserman, C. Coelho, A. Heller, C. Rothwell, "Invariant descriptors for 3D object recognition and pose", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 13(10), 1991, pp. 971-991.

[11] P.J. GarciaPardo, G.S. Sukhatme, J.F. Montgomery, "Towards vision-based safe landing for an autonomous helicopter", *Robotics and Autonomous Systems*, Vol. 38(1), 2002, pp. 19-29.

[12] L. He, Y. Chao, and K. Suzuki, "A run-based two-scan labeling algorithm", *IEEE Transactions on Image Processing*, Vol. 17(5), May 2008, pp. 749-756.

[13] K.Kanatani and L.Wu, "3D Interpretation of Conics and Orthogonality", *Image Understanding*, Vol. 58, 1993, pp. 286-301.

[14] S. Lange, N. Sünderhauf, P. Protzel, "A Vision Based Onboard Approach for Landing and Position Control of an Autonomous Multirotor UAV in GPS-Denied Environments", *Proceedings 2009 International Conference on Advanced Robotics*, Munich, June 2009, pp. 1-6.

[15] L. Meier, P. Tanskanen, F. Fraundorfer, M. Pollefeys, "PIXHAWK: A system for autonomous flight using onboard computer vision", *Proceedings 2011 IEEE International Conference on Robotics and Automation*, Shanghai, May 2011, pp. 2992-2997.

[16] D. Mellinger, N. Michael, V. Kumar, "Trajectory generation and control for precise aggressive maneuvers with quadrotors", *The International Journal of Robotics Research*, Jan. 2012 online first,

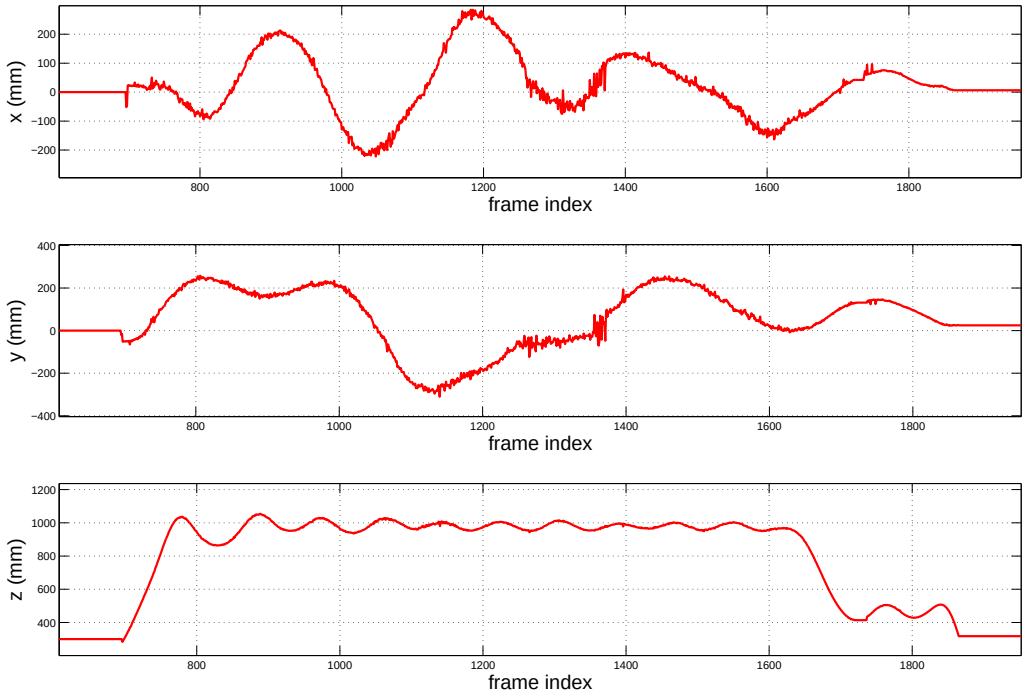


Fig. 11: Position estimates during an outdoor autonomous takeoff, hovering and landing flight.

doi:10.1177/0278364911434236.

[17] T. Merz, S. Duranti, G. Conte, "Autonomous landing of an unmanned helicopter based on vision and inertial sensing", *Experimental Robotics IX*, STAR, Vol. 21, 2006, pp. 343-352.

[18] N. Michael, D. Mellinger, Q. Lindsey, V. Kumar, "The GRASP Multiple Micro UAV Testbed", *Robotics and Automation Magazine*, Vol. 17(3), 2010, pp. 56-65.

[19] G. Bradski, "The OpenCV Library", *Dr. Dobb's Journal of Software Tools*, 2000.

[20] S. Saripalli, J.F. Montgomery, G.S. Sukhatme, "Visually guided landing of an unmanned aerial vehicle ", *IEEE Transactions on Robotics and Automation*, Vol. 19(3), 2003, pp. 371-380.

[21] S.A. Scherer, D. Dube, P. Komma, A. Masselli, A. Zell, "Robust Real-Time Number Sign Detection on a Mobile Outdoor Robot", *In Proceedings of the 6th European Conference on Mobile Robots (ECMR 2011)*, Orebro, Sweden, September 2011.

[22] K. E. Wenzel, P. Rosset, A. Zell., "Low-Cost Visual Tracking of a Landing Place and Hovering Flight Control with a Microcontroller", *Journal of Intelligent & Robotic Systems*, 2009, Vol. 57(1-4), pp. 297-311.

[23] K. E. Wenzel, A. Masselli, A. Zell, "Automatic Take Off, Tracking and Landing of a Miniature UAV on a Moving Carrier Vehicle", *Journal of Intelligent & Robotic Systems*, Vol. 61, 2011, pp. 221-238.

[24] G. Xu, Y. Zhang, S. Ji, Y. Cheng, Y. Tian, "Research on computer vision-based for UAV autonomous landing on a ship", *Pattern Recognition Letters*, Vol. 30(6), 2009, pp. 600-605.

[25] A. Zell, N. Mache, R. Hübner, G. Mamier, M. Vogt, M. Schmalzl, and K.-U. Herrmann, "SNNS (stuttgart neural network simulator)", *In Josef Skrzypek, editor, Neural Network Simulation Environments, volume 254 of The Springer International Series in Engineering and Computer Science*, Kluwer Academic Publishers, Norwell, MA, USA, February 1994. Chapter 9.

[26] Z. Zhang, "A Flexible New Technique for Camera Calibration", *IEEE Transaction on Pattern Analysis and Machine Intelligence*, Vol. 22(11), Nov. 2000, pp. 1330-1334.