Visual Servoing of Quadrotors for Perching by Hanging From Cylindrical Objects

Justin Thomas, Giuseppe Loianno, Kostas Daniilidis, and Vijay Kumar

***Abstract*—This letter addresses vision-based localization and servoing for quadrotors to enable autonomous perching by hanging from cylindrical structures using only a monocular camera. We focus on the problems of relative pose estimation, control, and trajectory planning for maneuvering a robot relative to cylinders with unknown orientations. We first develop a geometric model that describes the pose of the robot relative to a cylinder. Then, we derive the dynamics of the system, expressed in terms of the image features. Based on the dynamics, we present a controller, which guarantees asymptotic convergence to the desired image space coordinates. Finally, we develop an effective method to plan dynamically feasible trajectories in the image space, and we provide experimental results to demonstrate the proposed method under different operating conditions such as hovering, trajectory tracking, and perching.**

***Index Terms*—Aerial Robotics, Visual Servoing.**

I. Introduction

MICRO Aerial Vehicles (MAVs) are becoming ubiquitous and are being employed and considered for a wide range of tasks such as aerial photography, environmental monitoring, and disaster response. We see increasing autonomy in such areas, but the effectiveness of multi-rotor MAVs is, nevertheless, significantly limited by flight time. This is because the expensive power demands for hover as well as the limited energy density of batteries combine to limit flight durations, for example, to 10 to 20 minutes if the battery mass is 30% of the total mass [1]. However, there are many persistent operations requiring monitoring that do not require vehicles to be actively flying once a target location has been reached. Therefore, it is appealing to develop solutions that might allow robots to perch to conserve energy and/or recharge their batteries.

In this article, we propose a novel approach to perching by hanging for an autonomous quadrotor equipped with only a single camera and an onboard Inertial Measurement Unit (IMU). We focus on the problem of vision-based control to enable grasping a cylinder with an overhead gripper, allowing a robot to hang from branches, poles (see Fig. 1), or power lines, opening up possibilities for energy saving and inductive charging. Specifically, we consider the problem of obtaining reliable pose estimates relative to a cylinder and the synthesis of a control law that allows the vehicle to converge to stable perching positions using the apparent contour of the cylinder as feedback.

Fig. 1. A sample outdoor perch location for a quadrotor on a cantilevered cylindrical light post. Note that the axis of the cylinder can generally oriented.

Many other research groups have recognized the benefits of pairing a camera with an IMU. In particular, the two sensors have complementary characteristics, are lightweight, and are relatively inexpensive. In [2], a stereo camera configuration is employed for autonomous navigation and mapping of the environment while in [3], a monocular approach is used. Estimation using stereo cameras has proven to be effective [2], but requires heavy processing capabilities and a pair of calibrated cameras, which increases the weight and cost of the vehicle while decreasing the agility. Recent work has shown the feasibility of using a single monocular camera and an IMU for real-time localization [4]–[6] and autonomous flight [3], [7]–[9]. In these letters, information from the two sensors is fused using an Extended Kalman Filter (EKF) or an Unscented Kalman Filter (UKF) in order to provide localization that is viable for real-time control. These estimates include the positions and velocities of the robots in the inertial frame and possibly the 3-D positions of features observed by the camera.

However, common visual odometry techniques do not provide the ability to control the vehicle with respect to a specific object and therefore are not directly applicable to scenarios such as perching. Thus, it is necessary to leverage different approaches. In this paper, we are concerned with the direct control relative to commonly occurring objects such as cylinders, and not with the control of absolute position and orientation.

There is a foundational body of literature covering control using monocular vision for visual servoing applications, which discusses the differences between Position Based Visual Servoing (PBVS) and Image Based Visual Servoing (IBVS) [10]–[12]. The key difference between these approaches is that with PBVS, the pose of the robot is estimated

Manuscript received August 31, 2015; accepted November 14, 2015. Date of publication December 4, 2015; date of current version January 11, 2016. This paper was recommended for publication by Associate Editor E. Johnson and Editor J. Roberts upon evaluation of the reviewers' comments. This work was supported in part by the ARL under Grant W911NF-08-2-0004, in part by the ONR under Grants N00014-07-1-0829, N00014-14-1-0510, N00014-09-1-1051, and N00014-09-1-103, and in part by the NSF under Grants IIP-1113830, IIS-1426840, and IIS-1138847.

The authors are with the GRASP Laboratory, University of Pennsylvania, Philadelphia, PA USA (e-mail: jut@seas.upenn.edu; loiannog@seas.upenn.edu; kostas@seas.upenn.edu; kumar@seas.upenn.edu).

Digital Object Identifier 10.1109/LRA.2015.2506001

based on the image features, and the control law exists in the cartesian space. On the other hand, with IBVS, the control law is computed directly from features observed in the image [12]. Each has its benefits; for example, PBVS systems can use common filters in the MAV literature for the pose estimate while IBVS is considered to be more robust to camera calibration errors, making it appealing for low cost and low quality systems.

In the IBVS literature, most research analyzes first order systems [13] and proves stability for fully actuated systems [14]. Recently, a Port-Hamiltonian approach has been used to formulate a general IBVS approach [15], however an improvement in accuracy over traditional IBVS approaches is not demonstrated. Further, works such as [16]–[19] use a backstepping approach for control, while [20] and [21] use a geometric-based control approach. However, with backstepping, it is necessary to assume that the inner control loops are significantly faster than the outer ones. Furthermore, it is possible to design controllers for quadrotor MAVs which do not require these assumptions and can guarantee convergence from almost any point on SE(3), the Euclidean motion group [22].

Position-based Visual Servoing is directly related to computer vision work on recovering pose from the projection of known structures, which is widely used in 3D-tracking and augmented reality. In the context of this paper, approaches trying to recover pose from conics, cylinders, or quadrics are closely related. An extensive survey can be found in [23], but the closest and most specialized treatment is in [24] and [25]. The projection geometry of conics and the decomposition of conic projections in intrinsic and extrinsic parameters is introduced in [26], [27], and Augmented Reality based tracking using circles and cylinder projections was accomplished in [28].

The authors in [29] leverage a Structure from Motion (SFM) approach to estimate the parameters of a cylinder if the velocity of the camera is known, and in [30], an IBVS approach is presented for visual inspection of vertical cylinders at fixed distances, but requires a pilot or magnetometer to control the yaw and an additional sensor to estimate the height. Grasping of cylindrical objects is considered in [20] and [21], where the cylinder's axis is assumed to be perpendicular to gravity and an external motion capture system is used (in addition to onboard sensing) to control the motion along the cylinder axis.

In this paper, we develop the first fully autonomous vision-based quadrotor for perching, providing experimental results and a formulation of control and second order system dynamics in the image plane. We address the full 3D visual servoing problem of grasping a cylinder with an overhead gripper, allowing the robot to hang from branches or poles. Specifically, several key contributions are presented:

- A second order dynamic model of features in a virtual image plane is derived using a diffeomorphism between image features and robot's location in 3D space.
- An IBVS control law is proposed in which the stability is proved via Lyapunov theory.
- The system is shown to be differentially flat with respect to a set of given outputs in a virtual image space, which enables a simple method for planning dynamically feasible trajectories directly in the image space.
- The control law and trajectory planning method are demonstrated through successful perches of a quadrotor on real-world objects.

The paper is organized as follows: in Section II and Section III, the geometric and dynamic models are developed, respectively. An IBVS control law is presented in Section IV, and a method for trajectory planning is developed in Section V. Finally, experimental results are reported in Section VI, and Section VII concludes the paper.

II. Geometric Model

Consider a cylinder with a known radius, r, and an unknown axis, $\mathbf{a} \in \mathbb{S}^2$ defined in the camera frame, $\mathcal{C}$. In this section, all vectors will be defined with respect to the camera frame unless otherwise noted with a superscript. We assume that the radius of the cylinder is known (perhaps using the approach in [29]), and we focus on the problem of estimating the pose of the cylinder in a camera fixed frame. The closest concise treatment of a relative cylinder pose can be found in [25].

A. Projection of Quadric

Let $\mathbf{X} \in \mathbb{R}^3$, $M \in \mathbb{R}^{3\times 3}$, and

$$\mathbf{X}^T M \mathbf{X} + \mathbf{m}^T \mathbf{X} + \mu = 0 \tag{1}$$

be the equation of a quadric surface in $\mathbb{R}^3$ expressed in the coordinate system of a camera. Its projection $\mathbf{x} = (x, y, 1)$ to the image plane $Z = 1$ of a calibrated camera can be obtained by letting $\mathbf{X} = \lambda \mathbf{x}$ be the ray of projection and requiring that this ray is tangent to the surface. This means that the equation

$$\lambda^2 \mathbf{x}^T M \mathbf{x} + \lambda \mathbf{m}^T \mathbf{x} + \mu = 0 \tag{2}$$

must have exactly one solution for λ. For this to happen, the discriminant must vanish

$$(\mathbf{m}^T \mathbf{x})^2 - 4\mu \mathbf{x}^T M \mathbf{x} = 0. \tag{3}$$

Then, for any point $\mathbf{x}$ (recall that $\lambda \mathbf{x}$ is tangent to the surface) satisfying (3), we can recover its depth as

$$\lambda = -\frac{\mathbf{m}^T \mathbf{x}}{2\mathbf{x}^T M \mathbf{x}}. \tag{4}$$

Let us find the equation of a cylinder of arbitrary pose in camera coordinates. Let the cylinder be expressed as

$$X_o^2 + Y_o^2 = r^2 \tag{5}$$

in the object coordinates $(X_o, Y_o, Z_o) \in \mathbb{R}^3$. Then, in the camera frame, let the axis of the cylinder be the unit vector $\mathbf{a}$, and let the direction perpendicular to the plane spanned by the origin and the axis be the unit vector $\mathbf{b}$ as displayed in Fig. 2. Finally, let the vector from the origin to the closest point on the axis be $\beta \mathbf{c}$ with $||\mathbf{c}|| = 1$. Choosing the closest point of the axis as the origin of the object coordinate system, the transformation from the object to the camera becomes

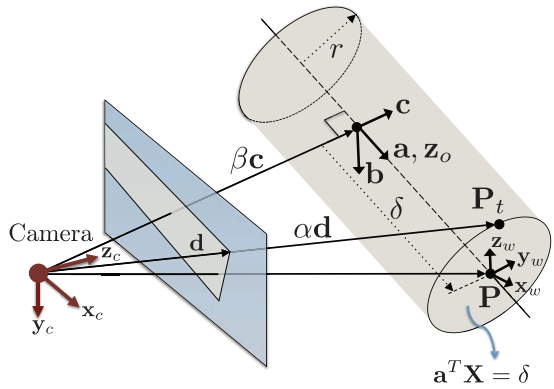

Fig. 2. The geometry of the camera-cylinder system. The coordinate systems of the camera, cylinder object, and world are given by $(\mathbf{x}_c, \mathbf{y}_c, \mathbf{z}_c)$, $(\mathbf{b}, \mathbf{c}, \mathbf{a})$, and $(\mathbf{x}_w, \mathbf{y}_w, \mathbf{z}_w)$, respectively. The axis of the cylinder $\mathbf{a}$ and the point on the axis closest to the camera, $\beta\mathbf{c}$, can be estimated directly from the lines representing the boundaries of the cylinder in the image. Further, we assume that the projection $\mathbf{d}$ of a ray passing through the camera and tangent to the cylinder at the $\mathbf{a}^T\mathbf{X} = \delta$ plane can be observed.

$$\mathbf{X} = \begin{pmatrix}\mathbf{c} & \mathbf{b} & \mathbf{a}\end{pmatrix}\begin{pmatrix}X_o \\ Y_o \\ Z_o\end{pmatrix} + \beta\mathbf{c} \tag{6}$$

or, rearranging,

$$\begin{pmatrix}X_o \\ Y_o \\ Z_o\end{pmatrix} = \begin{pmatrix}\mathbf{c}^T\mathbf{X} - \beta \\ \mathbf{b}^T\mathbf{X} \\ \mathbf{a}^T\mathbf{X}\end{pmatrix}. \tag{7}$$

Then, the cylinder can be expressed in camera coordinates as

$$(\mathbf{c}^T\mathbf{X} - \beta)^2 + (\mathbf{b}^T\mathbf{X})^2 = r^2 \tag{8}$$

and following (3), we obtain the projection and factorize

$$0 = (\beta\mathbf{c}^T\mathbf{x})^2 - (\beta^2 - r^2)((\mathbf{c}^T\mathbf{x})^2 + (\mathbf{b}^T\mathbf{x})^2) \tag{9}$$

$$= (\sqrt{\beta^2 - r^2}\mathbf{b} + r\mathbf{c})^T\mathbf{x}\,(\sqrt{\beta^2 - r^2}\mathbf{b} - r\mathbf{c})^T\mathbf{x}, \tag{10}$$

which shows that the projection of a cylinder is the union of two lines. Let us write these two lines as

$$\mathbf{n}_1^T\mathbf{x} = 0 \quad \text{and} \quad \mathbf{n}_2^T\mathbf{x} = 0 \tag{11}$$

with $\mathbf{n}_{i=1,2}$ being the unit vectors

$$\mathbf{n}_{1,2} = \frac{1}{\beta}\left(\sqrt{\beta^2 - r^2}\mathbf{b} \pm r\mathbf{c}\right). \tag{12}$$

We see that $\mathbf{n}_{1,2}$ contains all the information about the orientation of the cylinder:

$$\mathbf{a} \sim \mathbf{n}_1 \times \mathbf{n}_2, \qquad \mathbf{b} \sim \mathbf{n}_1 + \mathbf{n}_2, \qquad \mathbf{c} \sim \mathbf{n}_1 - \mathbf{n}_2, \tag{13}$$

and the distance of the camera to the closest point β can be recovered from the inner product

$$\mathbf{n}_1^T\mathbf{n}_2 = 1 - 2\frac{r^2}{\beta^2}, \tag{14}$$

which has a plausible geometric interpretation as $\cos\phi = 1 - 2\cos^2\frac{\phi}{2}$ where $\phi/2$ is the angle between $\mathbf{c}$ and the radial vector to the tangential point in the same plane.

1) A World-Fixed Point: The main objective is to determine the relative position of the robot, but also considering translation along the axis of the cylinder relative to some fixed point in the inertial frame. Thus, we are interested in determining the location of a fixed point, $\mathbf{P}$, that lies on the axis of the cylinder. Note that a tangent point cannot be used as the fixed point because it would move in the inertial frame based on the relative pose. We could, however, use points from features on the surface of the cylinder, but would then have to be concerned about preventing the feature from becoming occluded.

If we are given the projection $\mathbf{d}$ of a tangent point $\mathbf{P}_t$ as in Fig. 2, we first recover the depth α of the point from (4)

$$\alpha = \frac{\beta\mathbf{c}^T\mathbf{d}}{(\mathbf{c}^T\mathbf{d})^2 + (\mathbf{b}^T\mathbf{d})^2} \tag{15}$$

The plane that is perpendicular to the cylinder axis and passes through $\delta\mathbf{a}$ and $\mathbf{P}_t$ can be represented by

$$\mathbf{a}^T\mathbf{X} = \delta. \tag{16}$$

Since $\alpha\mathbf{d}$ is known from eq. (15), we can obtain δ from

$$\delta = \mathbf{a}^T(\alpha\mathbf{d}) = \frac{(\mathbf{a}^T\mathbf{d})\beta\mathbf{c}^T\mathbf{d}}{(\mathbf{c}^T\mathbf{d})^2 + (\mathbf{b}^T\mathbf{d})^2}. \tag{17}$$

Knowing δ, it is possible to compute $\mathbf{P}$ as

$$\mathbf{P} = \beta\mathbf{c} + \delta\mathbf{a}. \tag{18}$$

2) An Attitude Estimate: To control the robot, it is important to know the orientation relative to a fixed frame. We assume that we can estimate the gravity vector, $\mathbf{g}^{\mathcal{B}}$, in the robot body frame using the onboard IMU. The transformation from the robot frame to the camera frame is fixed and known by design, so we can determine the gravity vector in the camera frame, $\mathbf{g}^{\mathcal{C}}$. In addition, we have already concluded that we can estimate the axis of the cylinder, $\mathbf{a}$, also in the camera frame, which provides an anchor for the rotation about the vertical axis. If the cylinder is not vertical (i.e.$\mathbf{g}^{\mathcal{C}} \times \mathbf{a} \neq 0$), then the rotation between the camera frame and the world frame is

$$R_{\mathcal{W}}^{\mathcal{C}} = \begin{bmatrix}\mathbf{x}_{\mathcal{W}}^{\mathcal{C}} & \mathbf{y}_{\mathcal{W}}^{\mathcal{C}} & \mathbf{z}_{\mathcal{W}}^{\mathcal{C}}\end{bmatrix} \tag{19}$$

where $R_{\mathcal{W}}^{\mathcal{C}}$ rotates vectors with coordinates in $\mathcal{W}$ to $\mathcal{C}$ and

$$\mathbf{z}_{\mathcal{W}}^{\mathcal{C}} = -\frac{\mathbf{g}^{\mathcal{C}}}{\|\mathbf{g}^{\mathcal{C}}\|},\ \mathbf{y}_{\mathcal{W}}^{\mathcal{C}} = \frac{\mathbf{z}_{\mathcal{W}}^{\mathcal{C}} \times \mathbf{a}}{\|\mathbf{z}_{\mathcal{W}}^{\mathcal{C}} \times \mathbf{a}\|},\ \mathbf{x}_{\mathcal{W}}^{\mathcal{C}} = \mathbf{y}_{\mathcal{W}}^{\mathcal{C}} \times \mathbf{z}_{\mathcal{W}}^{\mathcal{C}}. \tag{20}$$

B. Image Features in a Fixed-Orientation Virtual Frame

The observed features are in the camera frame, but we are interested in a fixed-orientation frame located at the Center of Mass (COM) in order to simplify the control and planning problems. In practice, this could be achieved using a gimbal that would keep the camera at a fixed orientation in the world frame or, as in this case, we can re-project the observed cylinder onto a virtual image based on the known orientation of the robot.

The virtual image plane is parallel to the axis of the cylinder, and its orientation can be expressed as

$$R_{\mathcal{V}}^{\mathcal{C}} = \begin{bmatrix}\mathbf{a} & \mathbf{y}_{\mathcal{W}}^{\mathcal{C}} & \mathbf{a} \times \mathbf{y}_{\mathcal{W}}^{\mathcal{C}}\end{bmatrix}. \tag{21}$$

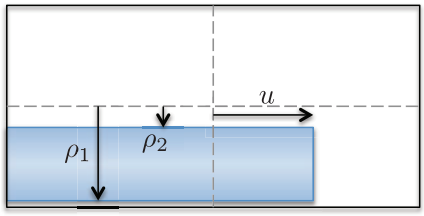

Fig. 3. The features in the virtual image are the coordinates (ρ_1 and ρ_2) of the two lines and the coordinate, u, representing the projection of the $\mathbf{a}^T\mathbf{X} = \delta$ plane.

In this case, we have formulated the virtual image plane as being in the $\mathbf{a}$-$\mathbf{y}_{\mathcal{W}}$ plane. However, if the user would prefer to operate beside (in contrast to above or below) the cylinder, a vertical plane would be more appropriate (*e.g.* the $\mathbf{a}$-$\mathbf{z}_{\mathcal{W}}$ plane). The translational offset of the camera from the COM of the system is known in the camera frame, which allows us to determine

$$\mathbf{P}^{\mathcal{V}} = T_{\mathcal{C}}^{\mathcal{V}}(\mathbf{P}) \tag{22}$$

where $T_{\mathcal{C}}^{\mathcal{V}}$ is a known transformation. The feature vector in the virtual image (see Fig. 3) is

$$\mathbf{s} \equiv \begin{bmatrix} \rho_1 \\ \rho_2 \\ u \end{bmatrix} = \Gamma\left(\mathbf{P}^{\mathcal{V}}, r\right) \tag{23}$$

where ρ_1 and ρ_2 can be computed from the lines of the cylinder and

$$u = \frac{\mathbf{e}_1^T\mathbf{P}^{\mathcal{V}}}{\mathbf{e}_3^T\mathbf{P}^{\mathcal{V}}} \tag{24}$$

where $\mathbf{e}_i$ denotes the i^{th} standard basis vector. Now that we have computed the image features in the virtual image, we will explore their dynamics to provide insight for control.

III. Dynamic Model in the Virtual Frame

As the robot moves, the image features will also move in the virtual image. Since we are ultimately interested in driving the features to a desired goal, it is important to understand how the control inputs of the system influence the dynamics and can be used to manipulate the image features. This will enable simulations as well as provide insight for the development of an image based control law.

The velocities of the image features are (see [12])

$$\dot{\mathbf{s}} = \frac{\partial\Gamma\left(\mathbf{P}^{\mathcal{V}}\right)}{\partial\mathbf{P}^{\mathcal{V}}}\dot{\mathbf{P}}^{\mathcal{V}}. \tag{25}$$

Let the position of the robot in the world frame be $\mathbf{x}_q$ so that the velocity of $\mathbf{P}$ can be computed to be

$$\dot{\mathbf{P}}^{\mathcal{V}} = -R_{\mathcal{C}}^{\mathcal{V}}R_{\mathcal{W}}^{\mathcal{C}}\dot{\mathbf{x}}_q. \tag{26}$$

Then, we can express the image feature velocities in terms of the robot velocity and $\mathbf{P}^{\mathcal{V}}$

$$\dot{\mathbf{s}} = -\frac{\partial\Gamma\left(\mathbf{P}^{\mathcal{V}}\right)}{\partial\mathbf{P}^{\mathcal{V}}}R_{\mathcal{C}}^{\mathcal{V}}R_{\mathcal{W}}^{\mathcal{C}}\dot{\mathbf{x}}_q \equiv J\dot{\mathbf{x}}_q. \tag{27}$$

We assume that $R_{\mathcal{W}}^{\mathcal{V}}$ is constant (*i.e.* the cylinder's orientation is fixed in the world) and known since $R_{\mathcal{W}}^{\mathcal{C}}$ is known from (19) and $R_{\mathcal{C}}^{\mathcal{V}}$ is known from (21). Since we have assumed that $r > 0$ and that the robot can not pass through the cylinder, it can be shown that the Jacobian is nonsingular.

The translational dynamics of a quadrotor in the world frame can be expressed as

$$m\ddot{\mathbf{x}}_q = -mg\mathbf{e}_3 + fR_{\mathcal{B}}^{\mathcal{W}}\mathbf{e}_3 \tag{28}$$

where m is the mass of the vehicle, g is gravitational acceleration, $\mathbf{e}_3 = \begin{bmatrix}0 & 0 & 1\end{bmatrix}^T$, f is the net thrust, which only acts vertically (*i.e.* in the $\mathbf{e}_3$ direction) in the body frame of the robot, and $R_{\mathcal{B}}^{\mathcal{W}}$ is the orientation of the vehicle's body with respect to the word frame. Further, the angular dynamics are given by

$$\dot{R}_{\mathcal{B}}^{\mathcal{W}} = R_{\mathcal{B}}^{\mathcal{W}}\hat{\boldsymbol{\Omega}} \tag{29}$$

$$\mathcal{I}\dot{\boldsymbol{\Omega}} + \boldsymbol{\Omega}\times\mathcal{I}\boldsymbol{\Omega} = \mathbf{M} \tag{30}$$

where $\boldsymbol{\Omega} \in \mathbb{R}^3$ is the body-frame angular velocity of the robot, $\mathcal{I}$ is the inertial tensor, $\mathbf{M} \in \mathbb{R}^3$ is the control moments, $\hat{\cdot}: \mathbb{R}^3 \mapsto \mathfrak{so}(3)$ is the "hat" map defined such that $\hat{\mathbf{a}}\mathbf{b} = \mathbf{a}\times\mathbf{b}$, and $\mathfrak{so}(3)$ is the Lie algebra of SO(3), the 3D rotation group. The control inputs are related to the rotor speeds through

$$\begin{bmatrix} f \\ \mathbf{M} \end{bmatrix} = \begin{bmatrix} k_f & k_f & k_f & k_f \\ 0 & lk_f & 0 & -lk_f \\ -lk_f & 0 & lk_f & 0 \\ k_m & -k_m & k_m & -k_m \end{bmatrix}\begin{bmatrix} \omega_1^2 \\ \omega_2^2 \\ \omega_3^2 \\ \omega_4^2 \end{bmatrix} \tag{31}$$

where k_f and k_m are the thrust and moment coefficients of the rotors, respectively, l is the length from the COM to the rotors, and ω_i is the angular speed of the i^{th} rotor [31]. Since this is invertible, the rotor speeds can be computed directly from the desired control inputs.

Using (27), the acceleration of the robot can be expressed in terms of J, J^{-1}, $\mathbf{s}$, and their derivatives

$$m\left(J^{-1}\ddot{\mathbf{s}} + \dot{J^{-1}}\dot{\mathbf{s}}\right) = fR_{\mathcal{B}}^{\mathcal{W}}\mathbf{e}_3 - mg\mathbf{e}_3. \tag{32}$$

Rearranging, the dynamics of the image features are

$$\ddot{\mathbf{s}} = \frac{1}{m}J\left(fR_{\mathcal{B}}^{\mathcal{W}}\mathbf{e}_3 - mg\mathbf{e}_3 - mJ^{\dot{-1}}\dot{\mathbf{s}}\right) \tag{33}$$

and can be used for simulation and the development of the control law in the next section.

IV. An Image-Based Control Law

The goal of our controller is to drive observed image features, $\mathbf{s}$, to desired values, $\mathbf{s}_{des}$. In our case, we want to allow the desired values to change with time (*i.e.* $\mathbf{s}_{des}(t)$) so that the robot can transition smoothly from one state to another. From the dynamics in (33), we propose the following control law. Let the desired thrust vector in the world frame be

$$\mathbf{f}_{des} = m\left(g\mathbf{e}_3 + J^{-1}\left(k_x\mathbf{e}_s + k_v\dot{\mathbf{e}}_s + \ddot{\mathbf{s}}_{des}\right) + \dot{J^{-1}}\dot{\mathbf{s}}\right) \tag{34}$$

where

$$\mathbf{e}_s = \mathbf{s}_{des} - \mathbf{s}, \quad \dot{\mathbf{e}}_s = \dot{\mathbf{s}}_{des} - \dot{\mathbf{s}} \tag{35}$$

are the position and velocity errors in the image coordinates, and k_x and k_v are positive gains. Note that $R_{\mathcal{B}}^{\mathcal{W}}$ can be computed as $R_{\mathcal{B}}^{\mathcal{W}} = R_{\mathcal{C}}^{\mathcal{W}} R_{\mathcal{B}}^{\mathcal{C}}$ where $R_{\mathcal{B}}^{\mathcal{C}}$ is a known, constant rotation and $R_{\mathcal{C}}^{\mathcal{W}}$ can be determined from the real image as in (19). For ease of notation, in the rest of this section, let $R \equiv R_{\mathcal{B}}^{\mathcal{W}}$.

Proposition 1: Let the commanded thrust and moments be

$$f = \mathbf{f}_{des} \cdot R\mathbf{e}_3, \tag{36}$$

$$\mathbf{M} = -K_R\mathbf{e}_R - K_\Omega\mathbf{e}_\Omega + \boldsymbol{\Omega} \times \mathcal{I}\boldsymbol{\Omega} \tag{37}$$

where K_R and K_Ω are positive gains. Also, $\mathbf{e}_R$ and $\mathbf{e}_\Omega$ represent the attitude errors (see [22])

$$\mathbf{e}_R = \frac{1}{2}\left(R_{des}^T R - R^T R_{des}\right)^\vee, \quad \mathbf{e}_\Omega = \boldsymbol{\Omega} - R^T R_{des}\boldsymbol{\Omega}_{des} \tag{38}$$

where $\cdot^\vee : \mathfrak{so}(3) \mapsto \mathbb{R}^3$ is the opposite of the hat map and the subscript "des" indicates a desired value. Then, the closed loop system is asymptotically stable.

Proof: A sketch is given below.

Stability of the Attitude Dynamics: For the attitude controller, let a Lyapunov candidate be

$$\mathcal{V}_R = \frac{1}{2}\mathbf{e}_\Omega \cdot \mathcal{I}\mathbf{e}_\Omega + K_R\Psi\left(R, R_{des}\right) + c_2\mathbf{e}_R \cdot \mathbf{e}_\Omega, \tag{39}$$

with $\Psi\left(R, R_{des}\right) = \frac{1}{2}\text{tr}\left[I - R_{des}^T R\right]$ (for properties, see [22]) and c_2 a positive scalar. Also, recall that $\mathcal{I}$ is the inertial tensor. Then, it can be shown that

$$\mathbf{z}_\theta^T M_\theta \mathbf{z}_\theta \leq \mathcal{V}_R \leq \mathbf{z}_\theta^T M_\Theta \mathbf{z}_\theta, \tag{40}$$

$$\dot{\mathcal{V}}_R \leq -\mathbf{z}_\theta^T W_\theta \mathbf{z}_\theta \leq 0, \tag{41}$$

where $\mathbf{z}_\theta = \left[\|\mathbf{e}_R\|, \|\mathbf{e}_\Omega\|\right]^T$, and M_θ, M_Θ, and W_θ are positive definite, guaranteeing the asymptotic stability of the attitude dynamics.

Translational Dynamics: Using (27), we can determine the image errors

$$\ddot{\mathbf{e}}_s = \ddot{\mathbf{s}}_{des} - \frac{1}{m}J\left(fR\mathbf{e}_3 - mg\mathbf{e}_3 - m\dot{J^{-1}}\dot{\mathbf{s}}\right). \tag{42}$$

Let a Lyapunov candidate be, with $c_1 > 0$,

$$\mathcal{V}_s = \frac{1}{2}k_x\|\mathbf{e}_s\|^2 + \frac{1}{2}m\|\dot{\mathbf{e}}_s\|^2 + c_1\mathbf{e}_s \cdot \dot{\mathbf{e}}_s. \tag{43}$$

Then, it is possible to show that

$$\dot{\mathcal{V}}_s \leq -\mathbf{z}_s^T W_s \mathbf{z}_s + \mathbf{z}_s^T W_{s\theta}\mathbf{z}_\theta, \tag{44}$$

where $\mathbf{z}_s = \left[\|\mathbf{e}_s\|, \|\dot{\mathbf{e}}_s\|\right]^T$ and W_s is positive definite.

Stability of the Underactuated System: Now, we consider the combined Lyapunov candidate for the translational and rotational error dynamics, $\mathcal{V} = \mathcal{V}_s + \mathcal{V}_R$. From (40) and (44),

$$\mathbf{z}_s^T M_s \mathbf{z}_s + \mathbf{z}_\theta^T M_\theta \mathbf{z}_\theta \leq \mathcal{V} \leq \mathbf{z}_\theta^T M_\Theta \mathbf{z}_\theta + \mathbf{z}_s^T M_S \mathbf{z}_s \tag{45}$$

where the matrices are positive definite, and it holds that

$$\dot{\mathcal{V}} \leq -\mathbf{z}_s^T W_s \mathbf{z}_s + \mathbf{z}_s^T W_{s\theta}\mathbf{z}_\theta - \mathbf{z}_\theta^T W_\theta \mathbf{z}_\theta \leq 0, \tag{46}$$

which guarantees that the entire underactuated system exhibits asymptotic stability. For the complete proof, see [32]. ■

V. Trajectory Planning

While the controller does guarantee stability, perching performance can be further improved by varying the desired setpoints in time ($\mathbf{s}_{des}(t)$) in a feasible manner. For a dynamic system, it is expected that a smooth transition between states (*e.g.* not a step input) is required, but with the state vector represented by image features, the exact requirements are not obvious. In the rest of this section, we present a planning method that clarifies the requirements for a dynamically feasible trajectory in the image feature space and also provides a method for minimizing the inputs to the system after being expressed as a *differentially flat* system.

A system is called *differentially flat* if there exists a change of coordinates which allows the state and control inputs to be written as functions of the flat outputs and their derivatives ($\mathcal{Y}$, $\dot{\mathcal{Y}}$, $\ddot{\mathcal{Y}}$, ...) [33]. One of the key benefits of the differential flatness property is that the task of trajectory planning for underactuated, dynamic systems can be simplified greatly, as will be demonstrated in the rest of this section. For more details regarding differentially flat systems, we refer the reader to [33], which provides examples and further explains the benefits of such systems.

For our system, let

$$\mathcal{Y} = \begin{bmatrix} \rho_1 \\ \rho_2 \\ u \\ \psi \end{bmatrix} = \begin{bmatrix} \mathbf{s} \\ \psi \end{bmatrix} \tag{47}$$

where $\mathbf{s}$ is the image feature vector and ψ is the yaw angle describing a rotation about the vertical axis of the inertial frame, $\mathbf{z}_w$. We will show that $\mathcal{Y}$ is a valid set of flat outputs for our system and that trajectories can be planned for $\mathcal{Y}(t)$ such that they are dynamically feasible. From a given trajectory, the thrust can be expressed considering (32) as

$$f = \left\| mg\mathbf{e}_3 + m\left(J^{-1}\ddot{\mathbf{s}} + \dot{J^{-1}}\dot{\mathbf{s}}\right)\right\| \tag{48}$$

which is dependent only on $\mathcal{Y}$ since J is a function of $\mathcal{Y}$ (assuming $R_{\mathcal{W}}^{\mathcal{V}}$ is known and constant). Since the orientation of the robot can be expressed as

$$R_{\mathcal{B}}^{\mathcal{W}} = \left[\, \mathbf{x}_{\mathcal{B}}^{\mathcal{W}} \;\; \mathbf{y}_{\mathcal{B}}^{\mathcal{W}} \;\; \mathbf{z}_{\mathcal{B}}^{\mathcal{W}} \,\right], \tag{49}$$

the $R_{\mathcal{B}}^{\mathcal{W}}\mathbf{e}_3$ term in (32) is equal to $\mathbf{z}_{\mathcal{B}}^{\mathcal{W}}$ and the dynamic model can be rewritten accordingly. Since $f \in \mathbb{R}$ and, by definition, $\left\|\mathbf{z}_{\mathcal{B}}^{\mathcal{W}}\right\| = 1$, we can solve for $\mathbf{z}_{\mathcal{B}}^{\mathcal{W}}$:

$$\mathbf{z}_{\mathcal{B}}^{\mathcal{W}} = \frac{m\left(J^{-1}\ddot{\mathbf{s}} + \dot{J^{-1}}\dot{\mathbf{s}}\right) + mg\mathbf{e}_3}{\left\|m\left(J^{-1}\ddot{\mathbf{s}} + \dot{J^{-1}}\dot{\mathbf{s}}\right) + mg\mathbf{e}_3\right\|}. \tag{50}$$

Defining a vector in the world, $\mathbf{x}_\psi = \left[\,\cos\psi \;\sin\psi \;0\,\right]^T$, we can write the other two components of the rotation matrix $R_{\mathcal{B}}^{\mathcal{W}}$ as

$$\mathbf{y}_{\mathcal{B}}^{\mathcal{W}} = \frac{\mathbf{z}_{\mathcal{B}}^{\mathcal{W}} \times \mathbf{x}_\psi}{\left\|\mathbf{z}_{\mathcal{B}}^{\mathcal{W}} \times \mathbf{x}_\psi\right\|}, \quad \mathbf{x}_{\mathcal{B}}^{\mathcal{W}} = \mathbf{y}_{\mathcal{B}}^{\mathcal{W}} \times \mathbf{z}_{\mathcal{B}}^{\mathcal{W}}. \tag{51}$$

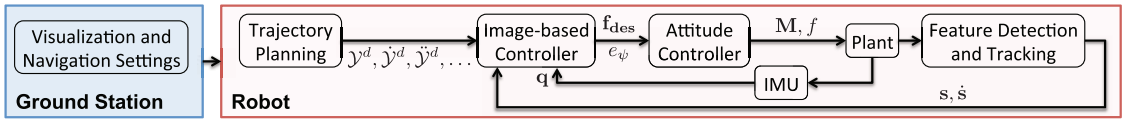

Fig. 4. The architecture for planning, control, and estimation. The ground station is only responsible for visualization and sending high level commands. A desired trajectory $(\mathcal{Y}^d, \dot{\mathcal{Y}}^d, \ddot{\mathcal{Y}}^d, \ldots)$ is computed and sent to the IBVS controller. The controller uses IMU feedback at 200 Hz (in the form of a quaternion, $\mathbf{q}$) and image feature feedback at 75 Hz to compute a desired force $\mathbf{f}_{des}$ and a yaw error e_ψ, which are sent to the attitude controller. Finally, the attitude controller computes the necessary moment control inputs $\mathbf{M}$ and the thrust f.

Since $\mathbf{z}_{\mathcal{B}}^{\mathcal{W}}$ and $\mathbf{x}_\psi$ are dependent on the flat outputs and the corresponding derivatives, we have shown that the body orientation is dependent on those quantities.

Next, we demonstrate that the angular velocity in the body frame is also dependent on the chosen set of flat outputs. Taking the derivative of (32), we obtain

$$m\left(2\dot{J^{-1}}\ddot{\mathbf{s}} + J^{-1}\mathbf{s}^{(3)} + \dot{J^{-1}}\dot{\mathbf{s}}\right) = \boldsymbol{\Omega}^{\mathcal{W}} \times f\mathbf{z}_{\mathcal{B}}^{\mathcal{W}} + \dot{f}\mathbf{z}_{\mathcal{B}}^{\mathcal{W}}, \tag{52}$$

and an inner product with $\mathbf{z}_B$ reveals that

$$\dot{f} = m\left(2\dot{J^{-1}}\ddot{\mathbf{s}} + J^{-1}\mathbf{s}^{(3)} + \dot{J^{-1}}\dot{\mathbf{s}}\right) \cdot \mathbf{z}_{\mathcal{B}}^{\mathcal{W}}. \tag{53}$$

Similarly, solving (52) for the $\boldsymbol{\Omega}^{\mathcal{W}} \times \mathbf{z}_{\mathcal{B}}^{\mathcal{W}}$ term and independently projecting onto $\mathbf{x}_{\mathcal{B}}^{\mathcal{W}}$ and $\mathbf{y}_{\mathcal{B}}^{\mathcal{W}}$ provides the first two terms of $\boldsymbol{\Omega}^{\mathcal{B}}$ by leveraging the circular shift property of the scalar triple product and the fact that $\boldsymbol{\Omega}^{\mathcal{W}} = R_{\mathcal{B}}^{\mathcal{W}}\boldsymbol{\Omega}^{\mathcal{B}}$. Then, the third component of $\boldsymbol{\Omega}^{\mathcal{B}}$ can be determined using an appropriate parameterization of $R_{\mathcal{B}}^{\mathcal{W}}$ and because $\dot{\psi}$ is known.

With another derivative, the angular acceleration of the robot appears as a function of $\mathcal{Y}$ (and higher derivatives) and allows for the computation of the control moments from (30). Thus, the control inputs are dependent on $\mathbf{s}^{(2)}, \mathbf{s}^{(3)}, \mathbf{s}^{(4)}$, $\dot{\psi}$, and $\ddot{\psi}$, and the dynamic system could be rewritten as a chain of integrators for each of the elements of $\mathcal{Y}$ with the appropriate derivative as the input to the flat system. For this reason, we plan trajectories that minimize the fourth derivative (*i.e.* snap) of the image features and of the second derivative of the yaw, ψ.

The trajectory planning problem can be formulated as a Quadratic Program (QP) with the requirement that the trajectories are $\mathcal{C}^4$ (or $\mathcal{C}^2$ for ψ) and solved using the method outlined in [31], which allows for the implementation of constraints formulated in terms of the flat outputs.

VI. Experimental Results

The experiments were conducted in the GRASP Lab [34] at the University of Pennsylvania using an Ascending Technologies Hummingbird quadrotor. An onboard computer (ODROID-XU3 from Hardkernel) handles all image processing and control, and the onboard sensors compose of the quadrotor's IMU and a Matrix Vision mvBlueFOX-MLC camera with a 98° horizontal and a 73° vertical field of view. The image processing and control use ROS, and the ODROID-XU3 is able to provide a rectified 752×480 image at frame-rate. A schematic and more detailed description of the system is provided in Fig. 4. As depicted in the figure, all of the visual processing, control, estimation, and planning is done onboard without human intervention or an external motion capture system. The user can remotely change navigation and visualization settings from a base station.

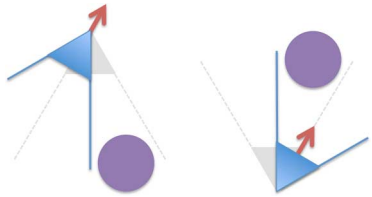

Fig. 5. Consider the case when the goal is to keep the robot aligned with the cylinder, but the robot starts slightly to the left (gray). Then, the robot must rotate clockwise to accelerate towards the right (blue). The red arrow indicates the desired direction of the thrust and also the desired direction of the $\mathbf{z}_{\mathcal{B}}$ axis. The left hand side depicts the camera and field of view if the camera is facing downwards. When the camera faces the same direction as the thrust (*i.e.* upwards), the robot more naturally keeps the cylinder in the field of view (right). In particular, this is important when maneuvering close to a cylinder such as when perching.

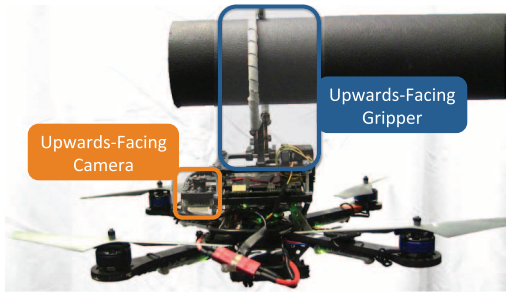

Fig. 6. The robot has an upwards facing camera and gripper to enable perching on objects above the robot.

We have intentionally pointed the camera upwards because it more naturally keeps the cylinder within the field of view (see Fig. 5). Additionally, the robot is equipped with an upwards facing $1 - DOF$ gripper, depicted in Fig. 6, which is actuated using a small servo.

A. Visual Processing

Efficient image processing is crucial since the algorithm must run at a sufficient rate for closed loop control. A sample of the result is given in Fig. 7. The image processing algorithm is mainly divided into two steps: 1) Line detection and 2) Line tracking.

1) Line Detection: The goal of the first task, which is executed only at the first frame, is to reliably identify the lines representing the image of the cylinder. To accomplish this, the detection algorithm leverages the OpenCV library and a Hough

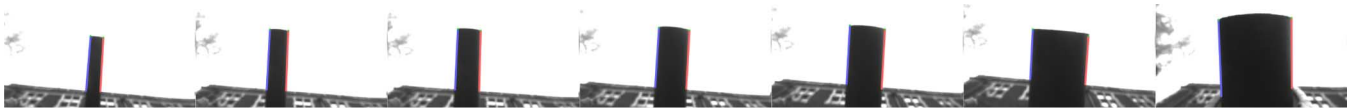

Fig. 7. A sample image sequence from the onboard camera while perching. The time increases from left to right. As the robot gets closer to the cylinder, the cylinder becomes larger in the image. The blue (left) and red (right) lines indicate the detected lines.

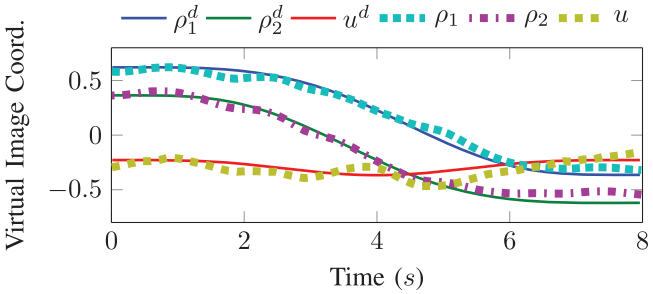

Fig. 8. A sample trajectory starting below and to the side of the cylinder, passing directly underneath, and ending on the other side of the cylinder. The desired trajectory is denoted by a "d" superscript.

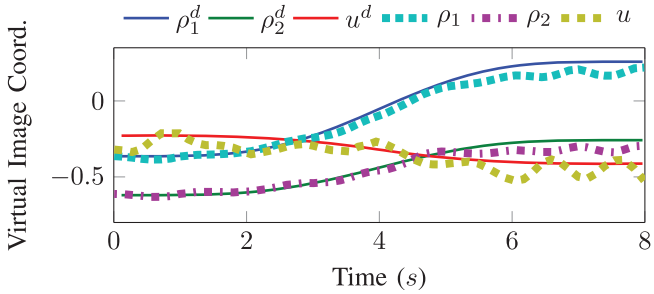

Fig. 9. The desired (computed from ρ_1^d, ρ_2^d, and u^d) and computed estimate of the quadrotor's position in the virtual frame $\mathcal{V}$ for a perching maneuver.

transform [35] to detect lines in the image. From the set of lines detected, we identify two with similar length and orientation while enforcing a minimum distance between the two lines.

2) Line Tracking: In the second task, the two detected lines are tracked in the image. We use the Visual Servoing Platform (ViSP) [36] library to track the moving edges identified in the previous step. Each sample point along a line is tracked from one image to another along the normal to the edge passing through the point. To improve real-time image processing and control, two threads are instantiated to simultaneously track the two lines, which leverages multiple processor cores. If the tracking is lost, the two steps are repeated again. The cylinder tracking code runs onboard at $75Hz$, and the computed parameters in the virtual plane are filtered to obtain velocity estimates.

B. Trajectory Planning and Perching

Now, we leverage the trajectory planning in Section V to smoothly transition between states. For experimental results of a sample trajectory, see Fig. 8. The image coordinates track the desired values using the control law proposed in Section VI. These results demonstrate the effectiveness of the proposed control law.

In many situations for aerial robots, it will beneficial for the robot to perch. Similar to the motivation for an upwards-facing camera, we also use an upwards facing gripper. In this way, corrections of the robot position keep the gripper facing towards the target. Using this setup, the robot can successfully perch as displayed in Fig. 6 using a trajectory in the first three components of $\mathcal{Y}$ as in Fig. 9. The estimate of $\beta\mathbf{c}$ in the camera frame provides the distance of the vehicle from the cylinder and can be used to determine when to close the gripper. The estimate of $\mathbf{x}_q^{\mathcal{V}}$ throughout the trajectory is shown in Fig. 10, and Table I shows the Root Mean Square Error (RMSE) of the vehicle with respect to the desired values. The first component is the largest because, in this maneuver, $\mathbf{x}_{\mathcal{V}}$ is closely aligned with the axis

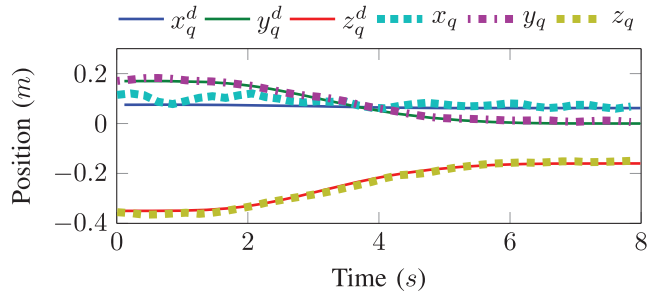

Fig. 10. Desired (computed from ρ_1^d, ρ_2^d, and u^d) and computed estimate of the quadrotor's position in the virtual frame $\mathcal{V}$ for a perching maneuver.

TABLE I
POSITION RMSE DURING A PERCHING MANEUVER.

Coordinate in frame $\mathcal{V}$	Position Root Mean Square Error (m)
x	0.020
y	0.008
z	0.009

of the cylinder, and the measurement of the end of the cylinder is noisy. The average Euclidean error is 2.1 cm for the position confirming the effectiveness of the controller. Finally, a video of the experimental results is provided in the supplementary multimedia material and online (see [32]), showing that the vehicle can hover, perch from different initial configurations, and handle various cylinder orientations.

VII. CONCLUSION

In this work, we presented the first vision-based control approach enabling a quadrotor, without the aid of GPS or a motion capture system, to perch on generally-oriented cylindrical objects. We developed a geometric model to determine the pose relative to a cylinder, used a virtual camera frame to simplify the image feature space, expressed the dynamics of the robot in terms of the image features, developed an image-based controller, and presented a method to plan dynamically feasible trajectories directly in the image plane. Finally, we provided

experimental results, demonstrating the success of our methods using a fully autonomous platform with all computation and sensing onboard.

There are still opportunities for further development. Our future work will include a sensitivity analysis of the control law, subject to radius uncertainty estimates, and will include a comparison between the current IBVS approach and a PBVS approach. The gripper will accommodate perching using a smaller contact area, not by caging the cylinder, which will enable perching on larger cylinders using a smaller gripper. We also will improve the cylinder detection to consider objects that are cylindrical in only some sections, such as a crooked tree branch. Finally, we aim to achieve vision-based perching on smooth, inclined surfaces, such as windows (see, for example, [37]), which will require the development of a new control law and detection algorithm.

ACKNOWLEDGMENT

We thank Terry Kientz for designing and building the gripper.

REFERENCES

[1] V. Kumar and N. Michael, "Opportunities and challenges with autonomous micro aerial vehicles," *Int. J. Robot. Res.*, vol. 31, no. 11, pp. 1279–1291, Aug. 2012.

[2] S. Shen, Y. Mulgaonkar, N. Michael, and V. Kumar, "Vision-based state estimation and trajectory control towards high-speed flight with a quadrotor," in *Robotics: Science and Systems*, Berlin, Germany, 2013.

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

[4] J. A. Hesch, D. G. Kottas, S. L. Bowman, and S. I. Roumeliotis, "Camera-IMU-based localization: Observability analysis and consistency improvement," *Int. J. Robot. Res.*, vol. 33, pp. 182–201, 2013.

[5] E. S. Jones and S. Soatto, "Visual-inertial navigation, mapping and localization: A scalable real-time causal approach," *Int. J. Robot. Res.*, vol. 30, no. 4, pp. 407–430, 2011.

[6] J. Kelly and G. S. Sukhatme, "Visual-inertial sensor fusion: Localization, mapping and sensor-to-sensor self-calibration," *Int. J. Robot. Res.*, vol. 30, pp. 56–79, 2011.

[7] S. Shen, N. Michael, and V. Kumar, "Tightly-coupled monocular visual-inertial fusion for autonomous flight of rotorcraft MAVs," in *Proc. Int. Conf. Robotics Autom.*, 2015, pp. 5303–5310.

[8] C. Forster, M. Pizzoli, and D. Scaramuzza, "SVO: Fast semi-direct monocular visual odometry," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2014, pp. 15–22.

[9] D. Scaramuzza *et al.*, "Vision-controlled micro flying robots: From system design to autonomous navigation and mapping in GPS-denied environments," *IEEE Robot. Autom. Mag.*, vol. 21, no. 3, pp. 26–40, Sep. 2014.

[10] F. Chaumette and S. Hutchinson, "Visual servo control. I. Basic approaches," *IEEE Robot. Autom. Mag.*, vol. 13, no. 4, pp. 82–90, Dec. 2006.

[11] F. Chaumette and S. Hutchinson, "Visual servo control. II. Advanced approaches [Tutorial]," *IEEE Robot. Autom. Mag.*, vol. 14, no. 1, pp. 109–118, Mar. 2007.

[12] S. Hutchinson, G. Hager, and P. Corke, "A tutorial on visual servo control," *IEEE Trans. Robot. Autom.*, vol. 12, no. 5, pp. 651–670, Oct. 1996.

[13] C. J. Taylor and J. P. Ostrowski, "Robust vision-based pose control," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2000, vol. 3, pp. 2734–2740.

[14] N. J. Cowan, J. D. Weingarten, and D. E. Koditschek, "Visual servoing via navigation functions," *IEEE Trans. Robot. Autom.*, vol. 18, no. 4, pp. 521–533, Aug. 2002.

[15] R. Mahony and S. Stramigioli, "A port-Hamiltonian approach to image-based visual servo control for dynamic systems," *Int. J. Robot. Res.*, vol. 31, pp. 1303–1319, 2012.

[16] T. Hamel and R. Mahony, "Visual servoing of an under-actuated dynamic rigid-body system: An image-based approach," *IEEE Trans. Robot. Autom.*, vol. 18, no. 2, pp. 187–198, Apr. 2002.

[17] T. Hamel and R. Mahony, "Image based visual servo control for a class of aerial robotic systems," *Automatica*, vol. 43, no. 11, pp. 1975–1983, Nov. 2007.

[18] N. Guenard, T. Hamel, and R. Mahony, "A practical visual servo control for an unmanned aerial vehicle," *IEEE Trans. Robot.*, vol. 24, no. 2, pp. 331–340, Apr. 2008.

[19] H. Jabbari, G. Oriolo, and H. Bolandi, "Dynamic IBVS control of an underactuated UAV," in *Proc. IEEE Int. Conf. Robot. Biomimetics*, Dec. 2012, pp. 1158–1163.

[20] J. Thomas, G. Loianno, K. Sreenath, and V. Kumar, "Toward image based visual servoing for aerial grasping and perching," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2014, pp. 2113–2118.

[21] J. Thomas, G. Loianno, J. Polin, K. Sreenath, and V. Kumar, "Toward autonomous avian-inspired grasping for micro aerial vehicles," *Bioinspiration Biomimetics*, vol. 9, no. 2, p. 025010, Jun. 2014.

[22] T. Lee, M. Leoky, and N. H. McClamroch, "Geometric tracking control of a quadrotor UAV on SE(3)," in *Proc. IEEE Conf. Decis. Control*, Dec. 2010, pp. 5420–5425.

[23] C. Doignon, *An Introduction to Model-Based Pose Estimation and 3-D Tracking Techniques*. Rijeka, Croatia: INTECH Open Access Publisher, 2007 [Online]. Available: http://www.intechopen.com/books/howtoreference/scene_reconstruction_pose_estimation_and_tracking/an_introduction_to_model-based_pose_estimation_and_3-d_tracking_techniques

[24] N. Navab, "Canonical representation and three view geometry of cylinders," *Int. Arch. Photogramm. Remote Sens. Spatial Inf. Sci. Comm. III*, vol. XXXIV, Part 3A, pp. 218–224, 2002.

[25] N. Navab and M. Appel, "Canonical representation and multi-view geometry of cylinders," *Int. J. Comput. Vis.*, vol. 70, no. 2, pp. 133–149, 2006.

[26] K. Kanatani, "Computational projective geometry," *CVGIP: Image Understand.*, vol. 54, no. 3, pp. 333–348, 1991.

[27] L. Quan, "Conic reconstruction and correspondence from two views," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 18, no. 2, pp. 151–160, Feb. 1996.

[28] É. Marchand and F. Chaumette, "Virtual visual servoing: A framework for real-time augmented reality," in *Computer Graphics Forum*, vol. 21, no. 3. Hoboken, NJ, USA: Wiley, 2002, pp. 289–297.

[29] R. Spica, P. R. Giordano, and F. Chaumette, "Active structure from motion for spherical and cylindrical targets," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2014, pp. 5434–5440.

[30] I. Sa, S. Hrabar, and P. Corke, "Inspection of pole-like structures using a vision-controlled VTOL UAV and shared autonomy," in *Proc. IEEE Int. Conf. Intell. Robots Syst.*, 2014, pp. 4819–4826.

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

[32] J. Thomas, G. Loianno, K. Daniilidis, and V. Kumar. (2015). *Visual Servoing of Quadrotors for Perching by Hanging from Cylindrical Objects: Supplementary Material* [Online]. Available: http://repository.upenn.edu/meam_papers/300/

[33] R. Murray, M. Rathinam, and W. Sluis, "Differential flatness of mechanical control systems: A catalog of prototype systems," in *Proc. ASME Int. Congr. Expo.*, 1995, pp. 1–9.

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

[35] J. Matas, C. Galambos, and J. Kittler, "Robust detection of lines using the progressive probabilistic hough transform," *Comput. Vis. Image Understand.*, vol. 78, no. 1, pp. 119–137, Apr. 2000.

[36] E. Marchand, F. Spindler, and F. Chaumette, "ViSP for visual servoing: A generic software platform with a wide class of robot control skills," *IEEE Robot. Autom. Mag.*, vol. 12, no. 4, pp. 40–52, Dec. 2005.

[37] J. Thomas *et al.*, "Aggressive Flight for Perching on Inclined Surfaces," *J. Mech. Robotics*, Dec. 2015, doi: 10.1115/1.4032250.