AN ADAPTIVE SCHEME FOR IMAGE-BASED VISUAL SERVOING OF AN UNDERACTUATED UAV

Hamed Jabbari Asl,* Giuseppe Oriolo,** and Hossein Bolandi*

Abstract

An image-based visual servoing (IBVS) method is proposed for controlling the 3D translational motion and the yaw rotation of a quadrotor. The dynamic model of this unmanned aerial vehicle (UAV) is considered at the design stage to account for its underactuation. In contrast with previous IBVS methods for underactuated UAVs, which used spherical image moments as visual features, the proposed controller makes use of appropriately defined perspective moments. As a consequence, we gain a clear improvement in performance, as satisfactory trajectories are obtained in both image and Cartesian space. In addition, an adaptation mechanism is included in the controller to achieve robust performance in spite of uncertainties related to the depth of the image features and to the dynamics of the robot. Simulation results in both nominal and perturbed conditions are presented to validate the proposed method.

Key Words

Unmanned aerial vehicle, quadrotor, underactuation, visual servoing, image-based visual servoing, adaptive control, image moments

1. Introduction

Vision-based control of robots has been the subject of intensive research since the late 1980s. The initial focus was on robot manipulators and led to identify two major approaches. The first is position-based visual servoing (PBVS), where the control law is computed from 3D Cartesian information reconstructed from 2D image data. Since in practice it may be difficult or impossible to reconstruct the 3D model of the observed scene from an image, most PBVS methods use estimation techniques that assume a priori knowledge of a geometric model of the observed object. Typically, PBVS schemes are quite sensitive to initial conditions and noise corrupting the image data [1]. Moreover, their computational load may become prohibitive in the presence of fast variations in the image data, *e.g.*, when the camera is mounted on an agile robot such as an unmanned aerial vehicle (UAV).

The second approach is image-based visual servoing (IBVS), in which the control law is computed using only visual features. Since IBVS does not need the reconstruction of full 3D information, it is naturally lighter and more robust than PBVS. However, the depth of the observed image features is still required; in fact, it is shown in [2] that inaccurate estimation of the feature depth can make IBVS systems unstable.

For high-speed tasks, it was suggested in [3] that it might be beneficial to take into account the dynamics of the robot when designing visual servoing control schemes. Actually, this is not essential for fully actuated robotic manipulators, whose dynamics can be reasonably well approximated by simple integrators [4]. However, some works have considered the dynamics of the manipulator in the control design. In [5], an asymptotically stable IBVS controller has been designed for the manipulator dynamic model. In [6] and [7], respectively, robust and adaptive IBVS schemes are proposed to cope with uncertainties on the vision system and the robot dynamic model.

In recent years, the steady increase in applications of UAVs has motivated researchers to consider vision-based control of these systems. Since UAVs are typically underactuated and move fast, it is advisable to take into account their dynamics in the design of visual control laws. PBVS approaches include [8], where the dynamic model of a helicopter is controlled using estimates of the camera pose and ego-motion derived from image data; [9], which uses a special camera configuration to reconstruct the pose of a quadrotor for dynamic control purposes; and [10], where visual servoing is used to automatically land a quadrotor.

Designing IBVS schemes that take into account the underactuated dynamics of UAVs (simply called "dynamic" IBVS schemes in the following) has proved to be challenging. In [11], a cascade nonlinear IBVS controller is proposed for a quadrotor: based on the observation that perspective moments would destroy the triangular cascade structure of the robot dynamics and the associated passivity-like properties, spherical image moments are used as visual features. However, simulation results of this

* Department of Electrical Engineering, Iran University of Science and Technology, Iran; e-mail: {hjabbari, h_bolandi}@iust. ac.ir

** Department of Computer, Control, and Management Engineering, Sapienza University of Rome, Italy; e-mail: oriole@dis. uniroma1.it

Recommended by Prof. B. Shirinzadeh

(DOI: 10.2316/Journal.206.2014.1.206-3942)

control law, together with experimental results of a modified version presented in [12], have shown that convergence of the quadrotor to the desired vertical position may be very slow, essentially due to the largely different sensitivity of the visual features in the three directions of motion. This performance problem is only slightly alleviated by using rescaled spherical image moments as proposed in [13].

In all the above IBVS controllers, an inertial measurement unit (IMU) provides the quadrotor orientation (roll, pitch, yaw angles) required to compute a suitable image error. An exception is [14], which also uses spherical image moments but lumps the unknown orientation in an uncertainty term, which is then taken into account in the stability analysis. However, the controlled system is not globally stable, and therefore this approach cannot be used for high-speed tasks where the robot executes aggressive maneuvers.

Recently, an approach based on virtual springs has been proposed in [15] to design a dynamic IBVS controller for quadrotors using perspective image moments. However, the dynamics of the translation along and rotation around the vertical axis is only locally stable and requires the restrictive assumption that the image plane is always parallel to the planar object where the visual features are located.

In this paper, we present a dynamic IBVS scheme for controlling 3D translational and yaw rotational motions of a quadrotor helicopter. Our control law makes use of perspective image features reconstructed in a suitably defined virtual image plane. As a consequence, the performance degradation associated to the use of spherical moments is avoided, and efficient trajectories are obtained in both image and Cartesian space. Another difference with respect to previous approaches is that only the roll and pitch angles of the UAV are required to compute the visual control law. With respect to our previous work [16], where this approach was first developed for the ideal case, an adaptation mechanism is added to the IBVS control law to guarantee robust performance in the presence of uncertainties related to the image (depth of the visual features) and to the dynamics of the robot (mass parameters).

The rest of the paper is organized as follows. Section 2 presents the kinematic and dynamic model of the considered UAV. In Section 3, we introduce the image features used by our control law and derive their dynamics. Our main result is contained in Section 4, which presents an adaptive IBVS method for controlling the traslational motion and the yaw angle of the quadrotor. Simulation results in both nominal and perturbed conditions are presented in Section 5, while a concluding discussion is given in Section 6.

2. Equations of Motion of the Quadrotor UAV

When IBVS is applied to fully actuated mechanical systems (*e.g.*, robot manipulators), it is possible to select the image features so as to achieve independent control of all the degrees of freedom. An underactuated robot, however, has fewer actuators than degrees of freedom, and therefore these cannot be controlled independently.

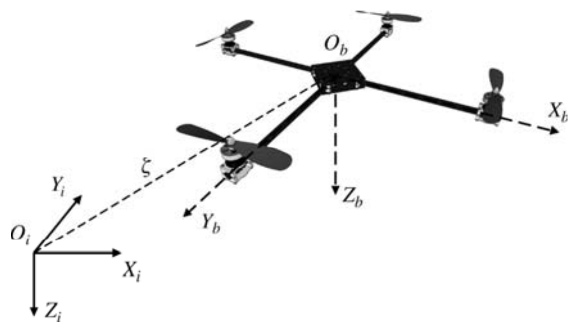

Figure 1. A quadrotor helicopter with its body frame. Also shown is an inertial frame.

Also, the interdependence among degrees of freedom may cause undesired image feature behaviours which, in the end, would hinder the performance of the IBVS scheme.

The underactuated UAV considered in this paper is a quadrotor, *i.e.*, a helicopter consisting of a rigid cross frame equipped with four rotors (see Fig. 1). The front and rear rotors rotate clockwise whereas the right and left rotors rotate counterclockwise. These opposite rotations cancel out reactive torques generated by the rotors and can compensate yaw motion, as done by the tail rotor in conventional helicopters [17]. Vertical motion can be achieved by controlling the total thrust generated by the rotors, *i.e.*, increasing or decreasing the angular velocity of the rotors. By increasing the thrust of one pair of rotors and decreasing the other while preserving total thrust, one can generate a yaw motion in the direction of the induced reactive torque. Roll (pitch) motion can be achieved by producing different thrusts with the left and right (front and rear) rotors. For a quadrotor, the interdipendence among degrees of freedom is seen in the fact that to generate forward/backward (left/right) motion, the quadrotor must pitch (roll).

Refer to Table 1 for a list of symbols used throughout the paper. To describe the equations of motion of the quadrotor, we consider two coordinate frames, *i.e.*, an inertial frame $\mathcal{I} = \{O_i, X_i, Y_i, Z_i\}$ and a body frame $\mathcal{B} = \{O_b, X_b, Y_b, Z_b\}$ attached to the centre of mass of the UAV. The position and orientation of $\mathcal{B}$ with respect to $\mathcal{I}$ are respectively expressed by vector $\zeta = (x, y, z)$ and the rotation matrix R associated to the three Euler angles ϕ, θ and ψ, respectively the roll, pitch and yaw angle.

Denoting by $V \in \Re^3$ and $\Omega = [\Omega_1\ \Omega_2\ \Omega_3]^T \in \Re^3$ respectively the linear and angular velocity of the UAV expressed in the body frame, the kinematic equations of the quadrotor as a 6-DOF rigid body can be written as follows:

$$\dot{\zeta} = RV \tag{1}$$

$$\dot{R} = Rsk(\Omega) \tag{2}$$

where $sk(\Omega)$ is the skew-symmetric matrix such that $sk(\Omega)b = \Omega \times b$, with $\times$ the vector cross-product and b a

Table 1
List of Main Symbols used in the Paper

$\mathcal{B}$	body frame
$\mathcal{C}$	camera frame
$\mathcal{I}$	inertial frame
$\mathcal{V}$	virtual frame
R	rotation matrix
$R_{\mathcal{C}}^{\mathcal{B}}$	rotation matrix between frames $\mathcal{B}$ and $\mathcal{C}$
$d_{\mathcal{C}}^{\mathcal{B}}$	position vector of $\mathcal{C}$ frame from $\mathcal{B}$ frame
Ω	angular velocity of the rigid body in $\mathcal{B}$ frame
V	linear velocity of the rigid body in $\mathcal{B}$ frame
${}^{\mathcal{I}}V$	linear velocity of the rigid body in $\mathcal{I}$ frame
v	linear velocity of the rigid body in $\mathcal{V}$ frame
F	input force of the rigid body in $\mathcal{B}$ frame
${}^{\mathcal{I}}F$	input force of the rigid body in $\mathcal{I}$ frame
f	input force of the rigid body in $\mathcal{V}$ frame
${}^{\mathcal{I}}p$	coordinates of point P in $\mathcal{I}$ frame
${}^{\mathcal{C}}p$	coordinates of point P in $\mathcal{C}$ frame
λ	focal length of camera
u, n	coordinates of a point in image plane
${}^{\mathcal{V}}u, {}^{\mathcal{V}}n$	coordinates of a point in virtual image plane
m	mass of quadrotor
m_{ij}	image moments
${}^{\mathcal{V}}m_{ij}$	image moments in virtual image plane
μ_{ij}	centered image moments
${}^{\mathcal{V}}\mu_{ij}$	centered image moments in virtual image plane
q_x, q_y, q_z, α	image features
U_1	thrust input
k_1, k_2, k_3, k_4	control gains
σ, γ, η	adaptation gains

vector in $\Re^3$. The relation between the time derivative of the Euler angles and the angular velocity Ω is given by:

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} 1 & \sin\phi\tan\theta & \cos\phi\tan\theta \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi/\cos\theta & \cos\phi/\cos\theta \end{bmatrix} \begin{bmatrix} \Omega_1 \\ \Omega_2 \\ \Omega_3 \end{bmatrix} \quad (3)$$

The matrix in (3) is singular when $\theta = \pm 90°$. This singularity is not relevant in image-based control, where image features should be kept in any case inside the field of view of the camera and therefore such pitch angles are not allowed.

Let m be the mass of the quadrotor and $J \in \Re^{3\times3}$ its body frame inertia matrix. The Newton-Euler equations describe the dynamics of the quadrotor as follows:

$$m\dot{V} = -m\,\Omega \times V + F \quad (4)$$

$$J\dot{\Omega} = -\Omega \times J\,\Omega + \tau \quad (5)$$

where $F \in \Re^3$ and $\tau \in \Re^3$ are respectively the force and torque vectors, which results from the action of the quadrotor actuators and of the gravitational field (for simplicity, we neglect other contributions such as gyroscopic effects produced by the rotors). In particular, the four rotors can only generate a single thrust force U_1 and a full torque vector $\tau = [U_2\ U_3\ U_4]^T$. Denoting by ω_1, ω_2, ω_3, ω_4 the angular velocity of the front, right, rear and left rotor, respectively, we can write

$$\begin{aligned} F &= -U_1 E_3 + mg\,R^T e_3 \\ &= \begin{bmatrix} 0 \\ 0 \\ b(\omega_1^2 + \omega_2^2 + \omega_3^2 + \omega_4^2) \end{bmatrix} + m\,gR^T e_3 \end{aligned} \quad (6)$$

$$\tau = \begin{bmatrix} U_2 \\ U_3 \\ U_4 \end{bmatrix} = \begin{bmatrix} bl(\omega_4^2 - \omega_2^2) \\ bl(\omega_3^2 - \omega_1^2) \\ d(\omega_2^2 + \omega_4^2 - \omega_1^2 - \omega_3^2) \end{bmatrix} \quad (7)$$

where $E_3 = e_3 = [0\ \ 0\ \ 1]^T$ are special unit vectors in the body frame and the inertial frame, respectively, b and d are the thrust and drag factors, g is the gravity acceleration and l is the distance between each rotor centre and the centre of mass of the quadrotor.

The kinematic and the dynamic equations (1) and (2) and (4) and (5) are written in body frame, but it is more common to use a hybrid formulation in which (4) is expressed in inertial frame, by replacing V with ${}^{\mathcal{I}}V = RV$ and F with ${}^{\mathcal{I}}F = RF$, whereas (5) is kept in body frame. By doing so, the equations assume the triangular cascade form widely used in the literature to develop nonlinear control schemes (*e.g.*, see [11]).

3. Image Dynamics

The choice of image features is critical for the performance of a visual servoing system. Typical features include point coordinates, parameters for line and circle segments, and image moments, see [4] and [20] for comprehensive treatments. As for image formation, typical models used in visual servoing are perspective and spherical projection.

It is shown in [13] that perspective image moments although have satisfactory practical results for IBVS control of the quadrotors but global stability of the system cannot be guaranteed, essentially due to the underactuated nature of these robots that must perform pitching or rolling motions to achieve Cartesian displacements. Thanks to their passivity properties, spherical image moments have been used to develop globally stable dynamic IBVS schemes for quadrotors, see [11], [12], [14], [18]. However, the performance of these methods in terms of UAV Cartesian motion is not satisfactory, as acknowledged in [12] and [15].

In view of the above, in this paper we will revisit the use of perspective image moments to design an IBVS scheme for the quadrotor. The roll and pitch angles of the UAV are used to reproject perspective image data in a suitable "virtual" image plane. Based on selected image features in this new image plane, it will be possible to design a globally stable dynamic IBVS scheme.

3.1 Reprojection of Image Points

As shown in Fig. 2, the quadrotor considered in this paper is equipped with a fixed camera that looks down. The camera frame $\mathcal{C} = \{O_c, X_c, Y_c, Z_c\}$ is attached to its centre of projection. The optical axis Z_c is orthogonal to the image plane, which is located at a distance λ (the focal length) from O_c.

Denote by ${}^{\mathcal{I}}p$ the coordinates of a fixed point P in the inertial frame and by ${}^{\mathcal{C}}p(t) = [{}^{\mathcal{C}}x\ {}^{\mathcal{C}}y\ {}^{\mathcal{C}}z]^T$ the coordinates of P in the camera frame. We have:

$${}^{\mathcal{C}}p(t) = R^T(t)\,[{}^{\mathcal{I}}p - O_c(t)]$$

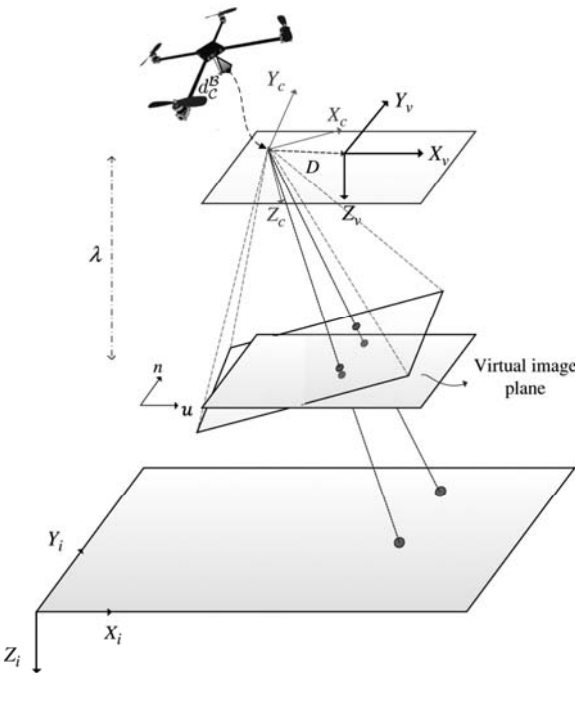

Figure 2. The camera frame $\mathcal{C} = \{O_c, X_c, Y_c, Z_c\}$ with the associated image plane and the virtual frame $\mathcal{V} = \{O_v, X_v, Y_v, Z_v\}$ with the associated virtual image plane.

where $R(t)$ is the rotation matrix defining the orientation of the camera frame relative to the inertial frame at time t. Using the well-known perspective projection equations, the image plane coordinates of the P are expressed as:

$$u = \lambda\frac{{}^{\mathcal{C}}x}{{}^{\mathcal{C}}z}, \qquad n = \lambda\frac{{}^{\mathcal{C}}y}{{}^{\mathcal{C}}z}$$

Now consider a virtual camera frame $\mathcal{V} = \{O_v, X_v, Y_v, Z_v\}$, which moves with the camera frame but in such a way that the $X_v - Y_v$ plane is always horizontal, *i.e.*, parallel to the $X_i - Y_i$ plane. There is a constant displacement $D = \left[{}^{\mathcal{C}}x_D\ {}^{\mathcal{C}}y_D\ {}^{\mathcal{C}}z\right]^T$ between O_c and the origin O_v of $\mathcal{V}$. Associated to $\mathcal{V}$, let us also define a virtual image plane, whose position and orientation w.r.t. $\mathcal{V}$ is the same as the actual image plane w.r.t. $\mathcal{C}$.

The coordinates of point P in the virtual frame will be denoted by ${}^{\mathcal{V}}p(t) = \left[{}^{\mathcal{V}}x\ {}^{\mathcal{V}}y\ {}^{\mathcal{V}}z\right]^T$ and are expressed instead as:

$${}^{\mathcal{V}}p(t) = R_{\psi}^T(t)[{}^{\mathcal{I}}p - O_v\,(t)]$$

where ψ is the UAV yaw angle and $R_{\psi}(t)$ is the rotation matrix of angle ψ around Z_i. Then, the time derivative of ${}^{\mathcal{V}}p$ will be:

$$\begin{aligned} {}^{\mathcal{V}}\dot{p} &= \frac{d}{dt}\left(R_{\psi}^T\left({}^{\mathcal{I}}p - O_v\right)\right) - R_{\psi}^T\dot{O}_v \\ &= -sk(\dot{\psi}\,e_3)R_{\psi}^T\left({}^{\mathcal{I}}p - O_v\right) - R_{\psi}^T\dot{O}_v \\ &= -sk(\dot{\psi}\,e_3){}^{\mathcal{V}}p - v \end{aligned} \tag{8}$$

where, for simplicity, time dependency is omitted. In (8), $\dot{O}_v(t) = \left[{}^{\mathcal{C}}v_x\ {}^{\mathcal{C}}v_y\ {}^{\mathcal{C}}v_z\right]^T$ is the linear velocity of the camera frame or of the virtual frame (they coincide) in the inertial frame, whereas $v(t) = \left[{}^{\mathcal{V}}v_x\ {}^{\mathcal{V}}v_y\ {}^{\mathcal{V}}v_z\right]^T$ is the same vector expressed in the virtual frame.

The image coordinates (u, n) can be reprojected in the virtual image plane. Denoting by $R_{\phi\theta}$ the matrix associated to a rotation of ϕ (roll) around X_i and then θ (pitch) around Y_i, we have:

$$\begin{bmatrix} {}^{\mathcal{V}}u \\ {}^{\mathcal{V}}n \\ \lambda \end{bmatrix} = \beta R_{\phi\theta}\begin{bmatrix} u \\ n \\ \lambda \end{bmatrix} - \begin{bmatrix} u_D \\ n_D \\ 0 \end{bmatrix} \tag{9}$$

where $u_D = \lambda{}^{\mathcal{C}}x_D/{}^{\mathcal{C}}z$, $n_D = \lambda{}^{\mathcal{C}}y_D/{}^{\mathcal{C}}z$, and multiplication by:

$$\beta = \lambda/\left([0\ 0\ 1]R_{\phi\theta}\begin{bmatrix} u \\ n \\ \lambda \end{bmatrix}\right)$$

ensures reprojection in an image plane with focal length equal to λ.

$\left({}^{\mathcal{C}}x_D, {}^{\mathcal{C}}y_D\right)$ are the coordinates of the distance of the virtual frame from the camera frame obtained in desired position of the robot. This displacement is considered in order to have proper image space error in order to design full dynamic IBVS in Section 4. To compute u_D and n_D, vertical distance and relative yaw rotation of the camera from the observed object are required. This information can be available from visual features. However, in most applications of visual servoing the task is to bring the observed object to the center of image plane where for the defined features in the following we will have $u_D = n_D = 0$.

To derive the dynamics of image features in the virtual image plane, we use the corresponding perspective projection equations. We have:

$$\begin{cases} {}^{\mathcal{V}}u = \lambda \frac{{}^{\mathcal{V}}x}{{}^{\mathcal{V}}z} \\ {}^{\mathcal{V}}n = \lambda \frac{{}^{\mathcal{V}}y}{{}^{\mathcal{V}}z} \end{cases} \Rightarrow \begin{cases} {}^{\mathcal{V}}\dot{u} = \lambda \left(\frac{{}^{\mathcal{V}}\dot{x}}{{}^{\mathcal{V}}z} - \frac{{}^{\mathcal{V}}x\,{}^{\mathcal{V}}\dot{z}}{{}^{\mathcal{V}}z^2} \right) \\ {}^{\mathcal{V}}\dot{n} = \lambda \left(\frac{{}^{\mathcal{V}}\dot{y}}{{}^{\mathcal{V}}z} - \frac{{}^{\mathcal{V}}y\,{}^{\mathcal{V}}\dot{z}}{{}^{\mathcal{V}}z^2} \right) \end{cases} \tag{10}$$

Using (8), we obtain:

$$\begin{cases} {}^{\mathcal{V}}\dot{u} = \lambda \left(\frac{{}^{\mathcal{V}}y\dot{\psi} - {}^{\mathcal{V}}v_x}{{}^{\mathcal{V}}z} + \frac{{}^{\mathcal{V}}x\,{}^{\mathcal{V}}v_z}{{}^{\mathcal{V}}z^2} \right) \\ {}^{\mathcal{V}}\dot{n} = \lambda \left(\frac{-{}^{\mathcal{V}}x\dot{\psi} - {}^{\mathcal{V}}v_y}{{}^{\mathcal{V}}z} + \frac{{}^{\mathcal{V}}y\,{}^{\mathcal{V}}v_z}{{}^{\mathcal{V}}z^2} \right) \end{cases}$$

Then, we can write the relationship between the velocity of a point in the virtual image plane and the velocity of the camera frame as:

$$\begin{bmatrix} {}^{\mathcal{V}}\dot{u} \\ {}^{\mathcal{V}}\dot{n} \end{bmatrix} = \begin{bmatrix} -\frac{\lambda}{z} & 0 & \frac{{}^{\mathcal{V}}u}{z} \\ 0 & -\frac{\lambda}{z} & \frac{{}^{\mathcal{V}}n}{z} \end{bmatrix} \begin{bmatrix} {}^{\mathcal{V}}v_x \\ {}^{\mathcal{V}}v_y \\ {}^{\mathcal{V}}v_z \end{bmatrix} + \begin{bmatrix} {}^{\mathcal{V}}n \\ -{}^{\mathcal{V}}u \end{bmatrix} \dot{\psi} \tag{11}$$

where $z = {}^{\mathcal{V}}z$ (considering the transformation between the virtual and the camera frame).

3.2 Image Features and Their Dynamics

Since our objective is to design an IBVS scheme for controlling the translational movements and the yaw rotation of the quadrotor, the selected image features should be sensitive to these motions. In IBVS, to have satisfactory trajectories in both image space and Cartesian coordinates, it is desirable to have a decoupled linear relationship between image space and Cartesian space velocities. Also, since the UAV may perform aggressive motions, it is convenient to use features that do not need to be tracked through the sequence of images.

In [19], a set of image features based on perspective image moments is proposed for IBVS of robotic manipulators. These features satisfy the criteria mentioned above; moreover, by knowing some information about the model of the observed object, the feature dynamics does not depend on the point depth. However, this information (which is similar to a priori knowledge in pose estimation algorithms) is not always available. Therefore, we will use the same idea without assuming preliminary information about the model of the observed object. For this purpose, we first define the features in the virtual plane and then derive their dynamics which will be used for visual servoing.

In IBVS control of UAVs, it is common to use planar objects as target [11]–[13], [15]. Under this assumption, the image moments m_{ij} of an object with N image points are defined as follows [19]:

$$m_{ij} = \sum_{k=1}^{N} u_k^i n_k^j \tag{12}$$

while the centered moments are given by:

$$\mu_{ij} = \sum_{k=1}^{N} (u_k - u_g)^i (n_k - n_g)^j$$

where $u_g = m_{10}/m_{00}$ and $n_g = m_{01}/m_{00}$. Differentiation of (12) with respect to time gives:

$$\dot{m}_{ij} = \sum_{k=1}^{N} i u_k^{i-1} n_k^j \dot{u}_k + j u_k^i n_k^{j-1} \dot{n}_k \tag{13}$$

For the considered object, the depth of any 3D point can be expressed as follows [20]:

$$\frac{1}{z} = Au + Bn + C \tag{14}$$

Since we are interested in determining the moment dynamics in the virtual image plane, and this plane is always parallel to the object, we have $A = B = 0$. Hence, by substituting (14) in (11) and using it in (13) together with (12), we obtain:

$$\begin{aligned} {}^{\mathcal{V}}\dot{m}_{ij} &= \begin{bmatrix} -\frac{i\lambda}{z}\,{}^{\mathcal{V}}m_{i-1,j} & -\frac{j\lambda}{z}\,{}^{\mathcal{V}}m_{i,j-1} & \frac{(i+j)}{z}\,{}^{\mathcal{V}}m_{i,j} \end{bmatrix} \begin{bmatrix} {}^{\mathcal{V}}v_x \\ {}^{\mathcal{V}}v_y \\ {}^{\mathcal{V}}v_z \end{bmatrix} \\ &\quad + \dot{\psi} \left(i\,{}^{\mathcal{V}}m_{i-1,j+1} - j\,{}^{\mathcal{V}}m_{i+1,j-1} \right) \end{aligned} \tag{15}$$

A similar procedure can be followed to determine the dynamics of the centered moments in the virtual image plane. We obtain:

$$\begin{aligned} {}^{\mathcal{V}}\dot{\mu}_{ij} &= \begin{bmatrix} 0 & 0 & \frac{(i+j)}{z}\,{}^{\mathcal{V}}\mu_{i,j} \end{bmatrix} \begin{bmatrix} {}^{\mathcal{V}}v_x \\ {}^{\mathcal{V}}v_y \\ {}^{\mathcal{V}}v_z \end{bmatrix} \\ &\quad + \dot{\psi} \left(i\,{}^{\mathcal{V}}\mu_{i-1,j+1} - j\,{}^{\mathcal{V}}\mu_{i+1,j-1} \right) \end{aligned} \tag{16}$$

The image features used for controlling the translational motion of the quadrotor are defined as follows:

$$q_x = q_z \frac{{}^{\mathcal{V}}u_g}{\lambda}, \quad q_y = q_z \frac{{}^{\mathcal{V}}n_g}{\lambda}, \quad q_z = \sqrt{\frac{a^*}{a}} \tag{17}$$

where a is defined as follows:

$$a = {}^{\mathcal{V}}\mu_{20} + {}^{\mathcal{V}}\mu_{02} \tag{18}$$

and a^* is the desired value of a. By using (15) and (16), and knowing that $z\sqrt{a} = z^*\sqrt{a^*}$, where z^* is the desired normal distance of the camera from the object, the dynamics of the features in the virtual image plane will be:

$$\begin{bmatrix} \dot{q}_x \\ \dot{q}_y \\ \dot{q}_z \end{bmatrix} = \begin{bmatrix} -\frac{1}{z^*} & 0 & 0 \\ 0 & -\frac{1}{z^*} & 0 \\ 0 & 0 & -\frac{1}{z^*} \end{bmatrix} \begin{bmatrix} {}^{\mathcal{V}}v_x \\ {}^{\mathcal{V}}v_y \\ {}^{\mathcal{V}}v_z \end{bmatrix} + \begin{bmatrix} q_y \\ -q_x \\ 0 \end{bmatrix} \dot{\psi} \tag{19}$$

In particular, these equations define the relationship between the velocity of the image features in the virtual image plane and the velocity of the virtual frame. Note that z^* in (19) is unknown; to compensate this uncertainty, an adaptive controller will be designed in Section 4. We can rewrite (19) as follows:

$$\dot{q} = -sk(\dot{\psi}\, e_3) \begin{bmatrix} q_x \\ q_y \\ q_z^D \end{bmatrix} - \frac{1}{z^*}\, v \tag{20}$$

where $q = [q_x\ q_y\ q_z]^T$ and q_z^D is an arbitrary value which will be properly chosen in Section 4. Equation (19) provides the required decoupled linear map between the velocity of image features and the linear velocity of the camera; also, this features dynamics possesses the passivity property mentioned in [11].

Finally, to control the yaw rotation of the quadrotor based on image features, we can use the object orientation α defined by [20]:

$$\alpha = \frac{1}{2}\arctan\left(\frac{2\,{}^{\mathcal{V}}\mu_{11}}{{}^{\mathcal{V}}\mu_{20} - {}^{\mathcal{V}}\mu_{02}}\right) \tag{21}$$

Using (16), the dynamics of angle α in the virtual image plane will be:

$$\dot{\alpha} = -\dot{\psi} \tag{22}$$

4. The Adaptive Dynamic IBVS Controller

In this section, we present a dynamic IBVS scheme for quadrotors which is adaptive with respect to (i) the unknown depth value z^* and (ii) the uncertain mass m of the system. The proposed approach can be extended to other parametric uncertainties in the dynamic model of the robot, *e.g.*, in the inertia matrix.

The control task is to move the quadrotor so that the image features observed by the on-board camera on a stationary object are driven to match some desired value obtained in advance. First, we will design a visual controller of translational motion based on the image features (17); because of underactuated nature of the quadrotor, this will require controlling the roll and pitch dynamics to achieve horizontal motions. Then, we will design a visual controller for the yaw angle of the quadrotor based on the image feature (21).

In designing the controller, it is assumed for generality that the camera frame $\mathcal{C}$ does not coincide with the body frame $\mathcal{B}$. In particular, they are related by a known constant homogeneous transformation expressed as follows:

$$T_{\mathcal{C}}^{\mathcal{B}} = \begin{bmatrix} R_{\mathcal{C}}^{\mathcal{B}} & d_{\mathcal{C}}^{\mathcal{B}} \\ 0 & 1 \end{bmatrix} \tag{23}$$

As shown in the following, the relative roll and pitch between the two frames will be compensated by reprojecting the image features in the virtual image plane through (9); the relative yaw, denoted by $R_{\mathcal{C}\psi}^{\mathcal{B}}$, will be used in the control design.

4.1 Control of the Translational Motion

To design the controller, we need to define an appropriate error in image space. With this assumption that depth information of the object is not available, based on our discussion in Section 3, we will consider the case that our desired image features are as follows:

$$q^d = \begin{bmatrix} q_x^d & q_y^d & q_z^d \end{bmatrix}^T = \begin{bmatrix} 0 & 0 & q_z^d \end{bmatrix}^T$$

where we have considered that a common IBVS objective is to place the barycenter of the observed object in the centre of the image plane. For the image features (17), this leads to $q_x^d = q_y^d = 0$. Note that the desired image features in the actual and virtual image planes are the same.

Define the image error as follows:

$$q_1 = q - [0\ \ 0\ \ q_z^d]^T$$

Using (20) and letting $q_z^D = q_z - q_z^d$, the dynamics of the image error are written as:

$$\dot{q_1} = -\mathrm{sk}\left(\dot{\psi}\, e_3\right) q_1 - \frac{1}{z^*} v$$

Now we can write the full dynamics (*i.e.*, including the quadrotor motion) of the image features. To this end, using (23) and $[R_{\mathcal{C}\psi}^{\mathcal{B}}]^T \dot{\psi}\, e_3 = \dot{\psi}\, e_3$, we write the translational dynamics (4) in the virtual frame and the attitude dynamics (5) in the body frame. We obtain:

$$\dot{q_1} = -sk(\dot{\psi}\, e_3) q_1 - \delta\, v \tag{24}$$

$$\dot{v} = -sk(\dot{\psi}\, e_3) v - \frac{f}{m} + G + D_d \tag{25}$$

$$\dot{R}_{\phi\theta} = R_{\phi\theta}\, sk\,(\Omega) - sk\left(\dot{\psi}\, e_3\right) R_{\phi\theta} \tag{26}$$

$$J\dot{\Omega} = -\Omega \times J\Omega + \tau \tag{27}$$

In (24), it is $\delta = 1/z^*$, while in (25) we have let:

$$f = [R^{\mathcal{B}}_{\mathcal{C}\psi}]^T R_{\phi\theta} U_1 E_3 \tag{28}$$

and also $D_d = -[R^{\mathcal{B}}_{\mathcal{C}\psi}]^T sk\left(\dot{\psi}\, e_3\right) d^{\mathcal{B}}_{\mathcal{C}}$, $G = [R^{\mathcal{B}}_{\mathcal{C}\psi}]^T g\, R^T_{\psi}\, e_3 = g\, e_3$. Equation (26) is obtained using (2) and the definition of angular velocity.

To design a controller for the translational motion of the quadrotor, we will use an adaptive backstepping approach with tuning functions to avoid over-parameterization of estimated parameters.

We assume that accurate measurements of the angular velocities of the robot are available, so that it is possible to implement a high gain control on them. As mentioned in [12], since these measurements are typically available at high rate, it is possible to ignore their dynamics in comparison to the dynamics of the other parts of the system. Therefore, our control design will only[1] consider (24)–(26).

From (26) and (28), we have:

$$\begin{aligned} R^T_{\phi\theta}\dot{f} + sk\left(R^T_{\phi\theta}\dot{\psi}\, e_3\right) U_1 e_3 \;&=\; sk(\Omega) U_1 e_3 + \dot{U}_1 e_3 \\ &= \begin{bmatrix} U_1\Omega_2 \\ -U_1\Omega_1 \\ \dot{U}_1 \end{bmatrix} \end{aligned} \tag{29}$$

Hence, we will consider the thrust jerk $\dot{U}_1$ and the angular velocities Ω_1 and Ω_2 as the available control inputs. To develop the controller, define the new parameter $\rho = 1/m$ (in order to avoid possible division by zero) and the following new error signals:

$$q_2 = \frac{v}{k_1} - q_1 \tag{30}$$

$$q_3 = f - \hat{m} k_1 A_p \tag{31}$$

where $A_p = \left(k_2 q_2 + \hat{\delta} k_1 q_2 + G/k_1 + D_d\right)$, k_1 and k_2 are constant gains and $\hat{m}$, $\hat{\delta}$ are the estimated values of parameters m, δ. The following quantities are also defined:

$$\begin{aligned} B_p \;=\; -\hat{m}k_1(k_2 + \hat{\delta}k_1)\bigg(&-\frac{\hat{\rho}}{k_1}q_3 - sk(\dot{\psi}\, e_3)q_2 - \hat{\rho}\hat{m}A_p \\ &+ \frac{G}{k_1} + \hat{\delta}k_1 q_1 + \hat{\delta}k_1 q_2 + D_d\bigg) \end{aligned}$$

$$C_p = -\hat{m}k_1^2 q_2$$

$$D_p = -\hat{m}k_1(k_2 + \hat{\delta}k_1)(k_1 q_1 + k_1 q_2)$$

$$E_p = \hat{m}k_1(k_2 + \hat{\delta}k_1)\left(\frac{q_3}{k_1} + \hat{m}A_p\right)$$

[1] However, the proposed procedure can be easily extended to the full dynamics of the system including (27).

where $\hat{\rho}$ is the estimated value of the parameter ρ. Our main result is the following.

Theorem 1. *Consider the system dynamics (24)–(26) and the control action:*

$$\dot{f} = -k_3 q_3 + \frac{\hat{\rho}}{k_1} q_2 + \sigma k_1 q_2 A^T_p A_p - B_p - C_p \dot{\hat{\delta}} \tag{32}$$

where $k_1, k_2, k_3 > 0$, together with the following update laws for parameters $\hat{m}$, $\hat{\delta}$ and $\hat{\rho}$:

$$\dot{\hat{m}} = \sigma q^T_2 A_p \tag{33}$$

$$\dot{\hat{\delta}} = \gamma\left(q^T_3 D_p + k_1 q^T_2 q_2\right) \tag{34}$$

$$\dot{\hat{\rho}} = \eta\left(q^T_3 E_p - \frac{q^T_2 q_3}{k_1}\right) \tag{35}$$

where σ, γ and η are positive constants. Then the errors q_1, q_2 and q_3 will converge to zero and the parameters estimates $\hat{m}$ and $\hat{\rho}$ will converge to their true values.

Proof: The controller defined by (32) is designed via backstepping. Hence, we start considering (24) as the dynamics that should be stabilized at the first step. Define the following Lyapunov function for (24):

$$L_1 = \frac{1}{2} q^T_1 q_1$$

so that:

$$\dot{L}_1 = q^T_1\left(-sk\left(\dot{\psi}\, e_3\right) q_1 - \delta v\right) = -\delta q^T_1 v$$

By considering v as control input, choose:

$$v = k_1 q_1$$

with $k_1 > 0$, which yields,

$$\dot{L}_1 = -\delta k_1 q^T_1 q_1$$

Since δ is a positive constant, if v was actually available as a control input then the origin of (24) would be globally asymptotically stable. Following the standard approach, we consider the error term q_2 defined in (30), which is the scaled difference between the desired and the actual value of v. The time derivative of q_1 is:

$$\dot{q}_1 = -sk\left(\dot{\psi}\, e_3\right) q_1 - \delta k_1 q_1 - \delta k_1 q_2 \tag{36}$$

and the time derivative of L_1 becomes:

$$\dot{L}_1 = -\delta k_1 q^T_1 q_1 - \delta k_1 q^T_1 q_2 \tag{37}$$

In view of (25) and (36), we can write:

$$\dot{q}_2 = \frac{\dot{v}}{k_1} - \dot{q}_1 = -sk(\dot{\psi}\, e_3)q_2 + \delta k_1 q_1 + \delta k_1 q_2 - \frac{f}{mk_1} + \frac{G}{k_1} + D_d \tag{38}$$

To stabilize (38), consider the following Lyapunov function:

$$L_2 = L_1 + \frac{1}{2} q_2^T q_2 + \frac{1}{2\sigma}\rho \tilde{m}^2 + \frac{1}{2\gamma}\tilde{\delta}^2$$

where $\tilde{m} = m - \hat{m}$ and $\tilde{\delta} = \delta - \hat{\delta}$. It is:

$$\dot{L}_2 = -\delta k_1 q_1^T q_1 + q_2^T \left(-\rho \frac{f}{k_1} + \frac{G}{k_1} + \hat{\delta} k_1 q_2 + D_d\right) - \frac{\rho \tilde{m}\dot{\hat{m}}}{\sigma} + \tilde{\delta}\left(k_1 q_2^T q_2 - \frac{\dot{\hat{\delta}}}{\gamma}\right) \tag{39}$$

With f as input, choose the following controller:

$$f = \hat{m} k_1 A_p \tag{40}$$

Substituting (40) in (39) and using $\rho \hat{m} = (1 - \rho\, \tilde{m})$, we get:

$$\dot{L}_2 = -\delta k_1 q_1^T q_1 - k_2 q_2^T q_2 + \rho \tilde{m}\left(q_2^T A_p - \frac{\dot{\hat{m}}}{\sigma}\right) + \tilde{\delta}\left(k_1 q_2^T q_2 - \frac{\dot{\hat{\delta}}}{\gamma}\right)$$

Following the standard backstepping approach with tuning functions, we now use the update law for the parameter $\hat{m}$ defined by (33), to cancel the effect of $\tilde{m}$, and leave the error term $\tilde{\delta}$ for the next step.

Since the quadrotor is an underactuated system, the same is clearly true for the dynamics of the camera. Hence, it is not possible to achieve independent control of its translational degrees of freedom through the force input; it is therefore necessary to continue the backstepping procedure until the actual control inputs are reached. To this end, the error signal (31) is used. The time derivative of q_2 is:

$$\dot{q}_2 = -sk(\dot{\psi}\, e_3)q_2 - \rho \hat{m} A_p + \frac{G}{k_1} + \delta k_1 q_1 + \delta k_1 q_2 - \frac{\rho}{k_1} q_3 + D_d \tag{41}$$

and the time derivative of L_2 is:

$$\dot{L}_2 = -\delta k_1 q_1^T q_1 - k_2 q_2^T q_2 - \frac{\rho}{k_1} q_2^T q_3 + \tilde{\delta}\left(k_1 q_2^T q_2 - \frac{\dot{\hat{\delta}}}{\gamma}\right) \tag{42}$$

Using (41), we can write:

$$\dot{q}_3 = \dot{f} - \sigma k_1 q_2^T A_p A_p + B_p + C_p \dot{\hat{\delta}} + D_p \tilde{\delta} + E_p \tilde{\rho} \tag{43}$$

To stabilize the origin of (43), consider the following Lyapunov function:

$$L_3 = L_2 + \frac{1}{2} q_3^T q_3 + \frac{1}{2\eta}\tilde{\rho}^2$$

where $\tilde{\rho} = \rho - \hat{\rho}$. Using (42) and (43), we obtain:

$$\dot{L}_3 = -\delta k_1 q_1^T q_1 - k_2 q_2^T q_2 + q_3^T\left(-\frac{\hat{\rho}}{k_1} q_2 + \dot{f} - \sigma k_1 q_2 A_p^T A_p + B_p + C_p \dot{\hat{\delta}}\right) + \tilde{\rho}\left(q_3^T E_p - \frac{q_2^T q_3}{k_1} - \frac{\dot{\hat{\rho}}}{\eta}\right) + \tilde{\delta}\left(q_3^T D_p + k_1 q_2^T q_2 - \frac{\dot{\hat{\delta}}}{\gamma}\right)$$

Substituting the control action (32) and the update laws (34) and (35) in the above formula, we get:

$$\dot{L}_3 = -\delta k_1 q_1^T q_1 - k_2 q_2^T q_2 - k_3 q_3^T q_3$$

Then, the equilibrium point $[q_1\ q_2\ q_3\ \tilde{m}\ \tilde{\delta}\ \tilde{\rho}]^T = [0\ 0\ 0\ 0\ 0\ 0]^T$ is stable and according to Theorem 4.12 of [21], q_1 q_2 and q_3 will converge to zero. To study the convergence of estimated parameters, consider that (41) on the invariant set $(q_1 = \dot{q}_1 = q_2 = \dot{q}_2 = q_3 = \dot{q}_3 = 0)$ reduces to:

$$\frac{\tilde{m}}{m}\left(\frac{G}{k_1} + D_d\right) = 0$$

showing that $\tilde{m}$ will converge to zero. Also, substituting (32) into (43), we obtain:

$$\tilde{\rho}\, k_1 \hat{m}^2\left(k_2 + \hat{\delta} k_1\right)\left(\frac{G}{k_1} + D_d\right) = 0$$

Since $k_1 \hat{m}^2 G(k_2 + \hat{\delta} k_1)(G/k_1 + D_d) \neq 0$, then also $\tilde{\rho}$ will converge to zero. Note that, according to (40), when $k_2 + \hat{\delta} k_1 = 0$ the force f will be constant, hence the system will be in a stable position and all the system states have converged to the final values. According to (36), (41) and (43), in the invariant set $\hat{\delta}$ will converge to the $\hat{\delta}$-axis. To avoid possible large values for $\hat{\delta}$, we can modify the update law (34) using projection [22] to ensure that $\hat{\delta}$ will remain in a predefined range $[\delta_{\min}, \delta_{\max}]$. In particular, assume that $\delta_{\min} + \epsilon < \hat{\delta} < \delta_{\max} + \epsilon$, where ϵ is a positive constant. The following projection operator is used to define the update law:

$$\dot{\hat{\delta}} = \mathrm{proj}(\hat{\delta}, \mu) = \begin{cases} \gamma\mu & \text{if } \delta_{\min} < \hat{\delta} < \delta_{\max} \text{ or} \\ & \text{if } \hat{\delta} \geq \delta_{\max} \text{ and } \mu \leq 0 \text{ or} \\ & \text{if } \hat{\delta} \leq \delta_{\min} \text{ and } \mu \geq 0 \text{ or} \\ \gamma\left(1 + \frac{\delta_{\max} - \hat{\delta}}{\epsilon}\right)\mu & \text{if } \hat{\delta} \geq \delta_{\max} \text{ and } \mu > 0 \\ \gamma\left(1 + \frac{\hat{\delta} - \delta_{\min}}{\epsilon}\right)\mu & \text{if } \hat{\delta} \leq \delta_{\min} \text{ and } \mu < 0 \end{cases} \tag{44}$$

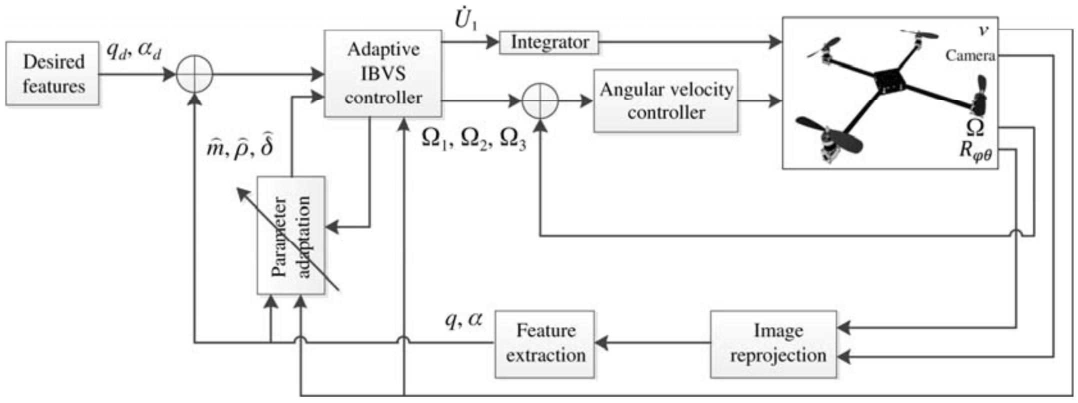

Figure 3. Block diagram of the adaptive scheme for IBVS controller of translational motion and yaw rotation.

where $\mu = q_3^T D_p + k_1 q_2^T q_2$. This operator will ensure that $\tilde{\delta}\left(q_3^T D_p + k_1 q_2^T q_2 - \frac{\dot{\hat{\delta}}}{\gamma}\right) \leq 0$ and so the stability properties of the system will be preserved.

A block diagram of our adaptive IBVS controller is shown in Fig. 3. The presented control law regulates the position of the camera through the error terms q_1 and q_2. Moreover, the feedback action on the error term q_3 will change the roll and pitch angles so as to generate the desired horizontal motions. The control inputs $\left(\dot{U}_1,\ \Omega_1,\ \Omega_2\right)$ which are depicted in Fig. 3 and designed in (32) can be computed by (29). As noted earlier, it would also be possible to continue the adaptive backstepping procedure so as to reach the actual control inputs of the quadrotor in (27).

4.2 Yaw Rotation Control

To control the yaw rotation of the quadrotor, define the following image error:

$$q_4 = \alpha - \alpha_d$$

where α_d is the desired value of the image feature. Then, according to (22), we get:

$$\dot{q}_4 = -\dot{\psi} \tag{45}$$

Similar to the control of translational dynamics, at this stage we consider Ω_3 as control input for system (45). Our control design is completed by the following result.

Theorem 2. *Consider system (45) with the control law:*

$$\Omega_3 = \frac{\cos\theta}{\cos\phi}\left(k_4 q_4 - \Omega_2 \frac{\sin\phi}{\cos\theta}\right) \tag{46}$$

with $k_4 > 0$. Then the error signal q_4 will converge to zero exponentially.

Proof: Using the relationship between $\dot{\psi}$ and Ω_3 entailed by (3) and substituting (46) in (45), we obtain:

$$\dot{q}_4 = -k_4 q_4$$

which implies the thesis.

5. Simulation Results

In this section, we report results of some MATLAB® simulations to show the performance of the proposed IBVS controller.

The dynamic parameters of the simulated quadrotor are $m = 2\,\text{kg}$, $g = 9.8\,\text{m/s}^2$ and $J = \text{diag}\{0.0081,\ 0.0081,\ 0.0142\}\,\text{kg}\cdot\text{m}^2/\text{rad}^2$. The torque τ is simply generated by proportional control (with unit gains) of the angular velocities, whose reference values are those produced by the IBVS controller and the dynamics of the rotors is assumed to be negligible compared to the airframe dynamics. The camera frame is set at 50 Hz, while the sampling time for

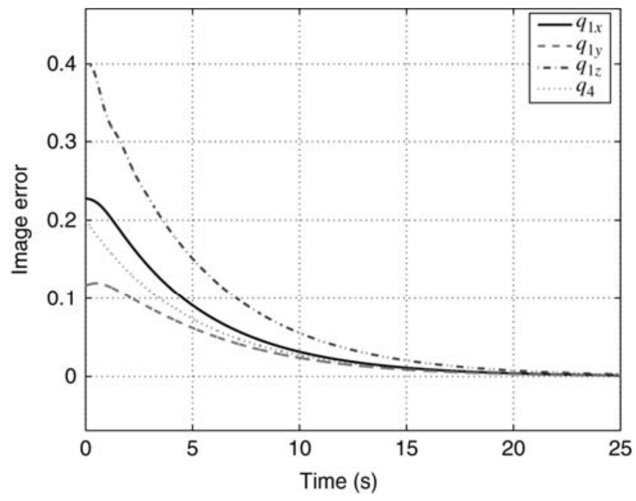

Figure 4. Simulation 1: time evolution of image space errors.

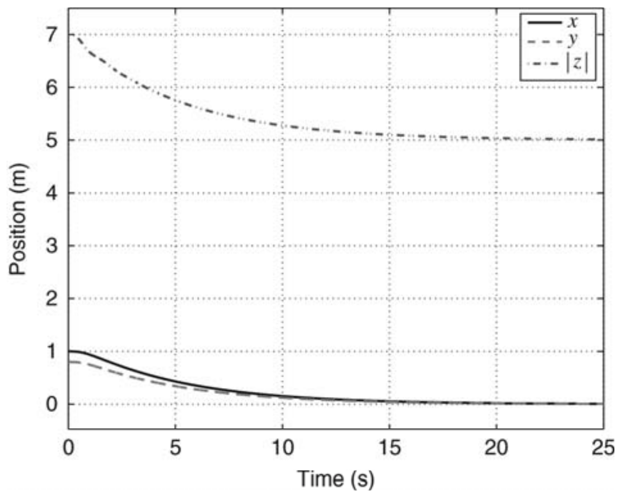

Figure 5. Simulation 1: time evolution of the quadrotor Cartesian coordinates.

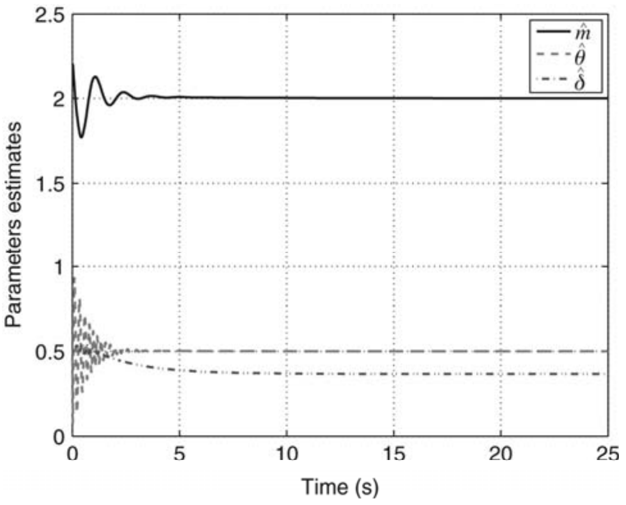

Figure 6. Simulation 1: parameter estimates.

the rest of the system is 1 ms. For simplicity, the camera frame and the body frame are assumed to coincide.

In all simulations, the robot is assumed to start in a hover position with the desired object (a rectangle) in the camera field of view. Visual information includes the image coordinates of four points corresponding to the four vertexes of the rectangle; these are used to calculate the image features defined by (17) and (21). The vertexes of the rectangle are located at $(0.25, 0.2, 0)$, $(0.25, -0.2, 0)$, $(-0.25, 0.2, 0)$, $(-0.25, -0.2, 0)$ with respect to the inertial frame. These points are transformed via perspective projection on a digital image plane with focal length divided by pixel size (identical in both u and n directions) equal to 213 and the principal point located at (80,60) (for more details see *e.g.*, [23]).

In the first simulation, the quadrotor initial position is $(1, 0.8, -7)$ while the desired image features were computed at the reference position $(0, 0, -5)$. The values of the image feature α at these positions are 0.2 and 0, respectively. The control gains are chosen as $k_1 = 1$, $k_2 = 2$, $k_3 = 3$ and $k_4 = 0.2$. The initial values of the parameter estimates are $\hat{m}(0) = 2.2$, $\hat{\rho}(0) = 1/\hat{m}(0)$ and $\hat{\delta}(0) = 0.5$. The gains for the parameter update laws are $\sigma = \gamma = 0.5$ and $\eta = 0.1$ while the parameters of the projection operator (44) are $\delta_{\max} = 5$, $\delta_{\min} = 0.01$ and $\epsilon = 0.005$. Figure 4 shows the evolution of the image space errors q_1 and q_4. The translational motion of the robot in Cartesian coordinates is shown in Fig. 5. The parameter estimates $\hat{m}$, $\hat{\delta}$ and $\hat{\rho}$ are illustrated in Fig. 6. Finally, the image space trajectories of the observed object in the camera image plane and in the virtual image plane are shown in Fig. 7. As expected, the results show that satisfactory trajectories have been achieved in both Cartesian coordinates and the virtual image plane. Also, parameters $\hat{m}$ and $\hat{\rho}$ converge to their true values, while parameter $\hat{\delta}$ is inside the range defined by projection operator.

For comparison, we have run the same simulation using the spherical image moments used in [11] and [14] as visual information for the IBVS controller designed in [11]. Figure 8 shows the evolution of the quadrotor Cartesian

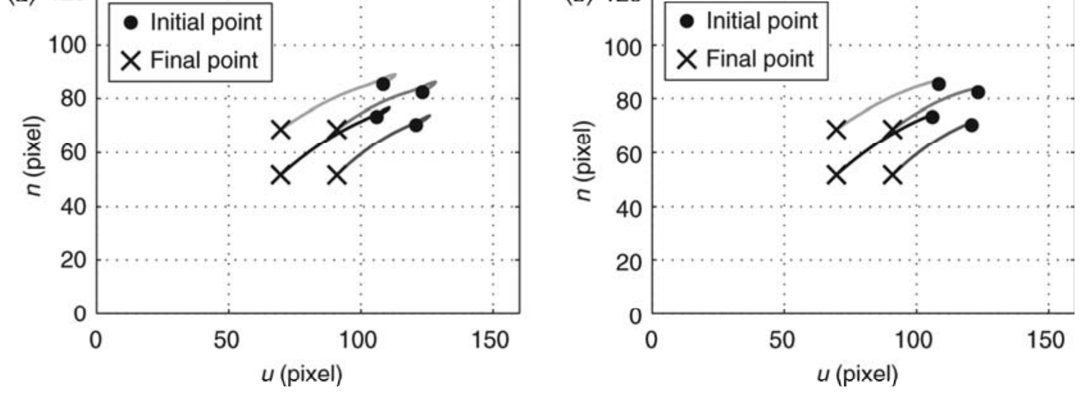

Figure 7. Simulation 1: object point trajectories in the image plane (a) in the virtual image plane (b).

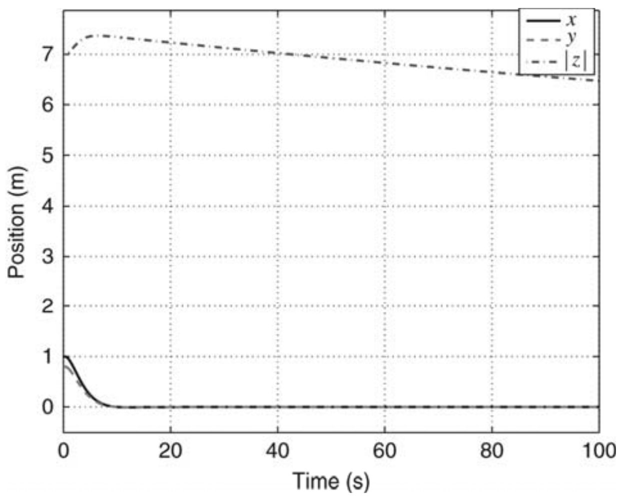

Figure 8. Simulation 1 using spherical moments and IBVS controller of [11]: time evolution of the quadrotor Cartesian coordinates.

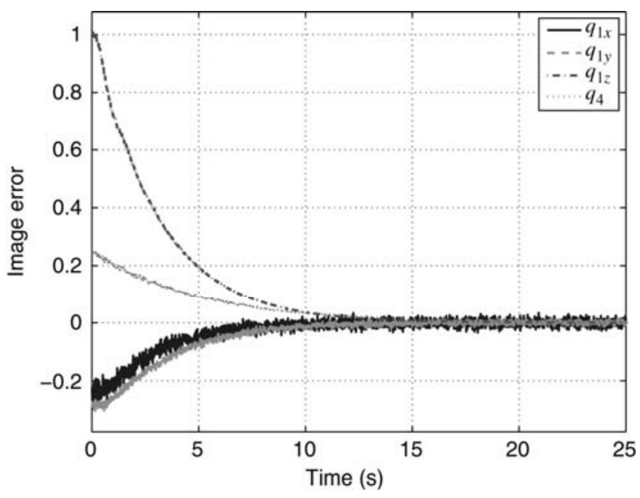

Figure 9. Simulation 2: time evolution of image space errors.

coordinates, confirming that the behaviour of the z coordinate is unsatisfactory in view of its very slow convergence.

In the second simulation, to evaluate the controller performance we have used noisy data for both attitude and visual information. The robot's initial position is assumed to be $(-0.5, -1, -6)$ and the desired image features are obtained at $(0, 0, -3)$. The value of the visual feature α at these positions is 0.25 and 0, respectively. Control gains, update laws gains, initial values for parameters estimates and parameters for the projection operator are the same as before. The covariance of noise is 10^{-4} and 1 for attitude and visual information, respectively. Image space errors and Cartesian coordinates are shown in Figs. 9 and 10, respectively. The noisy Euler angles of the robot are plotted in Fig. 11. Figure 12 shows the parameters estimate and Fig. 13 depicts the object trajectories in the camera image plane and the virtual image plane. In spite of the

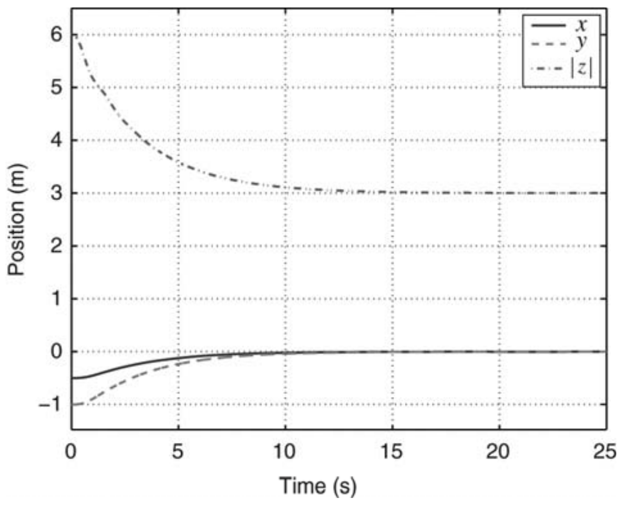

Figure 10. Simulation 2: time evolution of the quadrotor Cartesian coordinates.

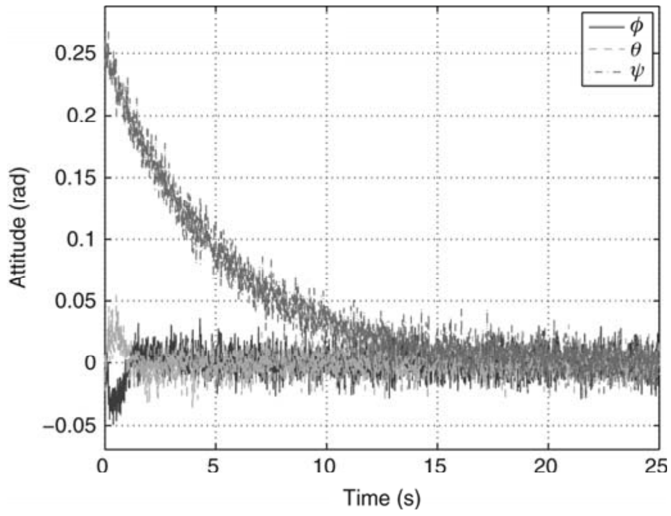

Figure 11. Simulation 2: noisy Euler angles of the quadrotor.

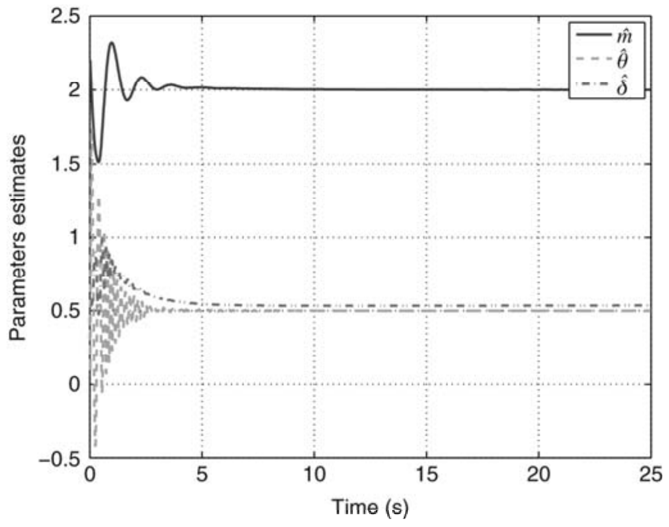

Figure 12. Simulation 2: parameter estimates.

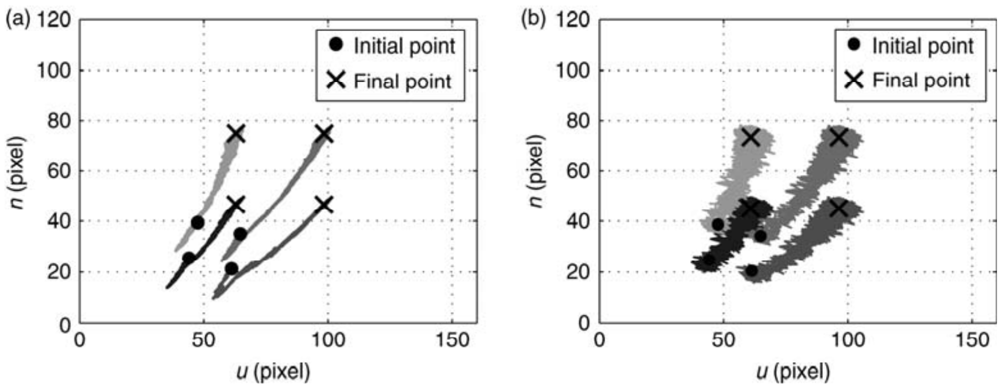

Figure 13. Simulation 2: object point trajectories in the image plane (a) in the virtual image plane (b).

presence of noise, the performance of the proposed IBVS controller is satisfactory.

6. Conclusion

In this paper, an adaptive IBVS scheme has been developed for controlling the translational motion and the yaw angle of a quadrotor helicopter equipped with an on-board camera. The proposed scheme makes use of suitably defined perspective image moments as visual features, ultimately resulting in a clear improvement of the transient response in Cartesian space with respect to other methods in the literature that are based on spherical moments. Our nonlinear controller is designed following the backstepping approach and includes an adaptation mechanism that copes with the presence of uncertainty in the observed scene (the feature depth) as well as in the robot dynamics (the quadrotor mass). Simulations results for the task of positioning the quadrotor by observation of a stationary object show the satisfactory performance of the proposed method, even in the presence of noise corrupting inertial and image data.

However, since the control is designed based on the dynamics of the image features in the virtual image plane, it is not guaranteed that the visual features will be inside the field of view of the camera during the whole mission. Also, the proposed controller for translational motion assumes that the linear velocity of the quadrotor is known. While in principle this requirement could be satisfied by specific sensors, the use of such expensive equipment does not seem appropriate for low-cost UAVs. Therefore, future work will be aimed at designing an observer-based visual servoing scheme to avoid the need of the linear velocity information beside considering the visibility of image features during the task. We also intend to verify the effectiveness of the proposed control algorithm by experimental validation on a real quadrotor retrofitted with a camera.

References

[1] F. Janabi-Sharifi and M. Marey, A Kalman-filter-based method for pose estimation in visual servoing, *IEEE Transactions on Robotics*, *26*, 2010, 939–947.

[2] E. Malis, Y. Mezouar, and P. Rives, Robustness of image-based visual servoing with a calibrated camera in the presence of uncertainties in the three-dimensional structure, *IEEE Transactions on Robotics*, *26*, 2010, 112–120.

[3] F. Chaumette and S. Hutchinson, Visual servo control. II. Advanced approaches, *IEEE Robotics and Automation Magazine*, *14*, 2007, 109–118.

[4] S. Hutchinson, G. Hager, and P. Corke, A tutorial on visual servo control, *IEEE Transactions on Robotics and Automation*, *12*, 1996, 651–670.

[5] R. Kelly, R. Carelli, O. Nasisi, B. Kuchen, and F. Reyes, Stable visual servoing of camera-in-hand robotic systems, *IEEE/ASME Transactions on Mechatronics*, *5*, 2000, 39–48.

[6] C.-S. Kim, E.-J. Mo, S.-M. Han, M.-S. Jie, and K.-W. Lee, Robust visual servo control of robot manipulators with uncertain dynamics and camera parameters, *International Journal of Control, Automation and Systems*, *8*, 2010, 308–313.

[7] L. Hsu and P. Aquino, Adaptive visual tracking with uncertain manipulator dynamics and uncalibrated camera, *38th IEEE Conf. on Decision and Control*, 1999, 1248–1253.

[8] O. Shakernia, Y. Ma, T. J. Koo, T. John, and S. Sastry, Landing an unmanned air vehicle: vision based motion estimation and nonlinear control, *Asian Journal of Control*, *1*, 1999, 128–145.

[9] J.P. Ostrowski and C.J. Taylor, Control of a quadrotor helicopter using dual camera visual feedback, *The International Journal of Robotics Research*, *24*, 2005, 329–341.

[10] L. Garca Carrillo, E. Rondon, A. Sanchez, A. Dzul, and R. Lozano, Stabilization and trajectory tracking of a quadrotor using vision, *Journal of Intelligent and Robotic Systems*, *61*, 2011, 103–118.

[11] T. Hamel and R. Mahony, Visual servoing of an under-actuated dynamic rigid-body system: an image-based approach, *IEEE Transactions on Robotics and Automation*, *18*, 2002, 187–198.

[12] N. Guenard, T. Hamel, and R. Mahony, A practical visual servo control for an unmanned aerial vehicle, *IEEE Transactions on Robotics*, *24*, 2008, 331–340.

[13] O. Bourquardez, R. Mahony, N. Guenard, F. Chaumette, T. Hamel, and L. Eck, Image-based visual servo control of the translation kinematics of a quadrotor aerial vehicle, *IEEE Transactions on Robotics*, *25*, 2009, 743–749.

[14] T. Hamel and R. Mahony, Image based visual servo control for a class of aerial robotic systems, *Automatica*, *43*, 2007, 1975–1983.

[15] R. Ozawa and F. Chaumette, Dynamic visual servoing with image moments for a quadrotor using a virtual spring approach, *2011 IEEE Int. Conf. on Robotics and Automation*, 2011, 5670–5676.

[16] H. Jabbari, G. Oriolo, and H. Bolandi, Dynamic IBVS control of an underactuated UAV, *IEEE Int. Conf. on Robotics and Biomimetics*, 2012, 1158–1163.

[17] A. Tayebi and S. McGilvray, Attitude stabilization of a vtol quadrotor aircraft, *IEEE Transactions on Control Systems Technology*, *14*, 2006, 562–571.

[18] F. Le Bras, T. Hamel, R. Mahony, and A. Treil, Output feedback observation and control for visual servoing of VTOL

UAVs, *International Journal of Robust and Nonlinear Control*, *21*, 2011, 1008–1030.

[19] O. Tahri and F. Chaumette, Point-based and region-based image moments for visual servoing of planar objects, *IEEE Transactions on Robotics*, *21*, 2005, 1116–1127.

[20] F. Chaumette, Image moments: a general and useful set of features for visual servoing, *IEEE Transactions on Robotics*, *20*, 2004, 713–723.

[21] I.K.M. Krstic and P.V. Kokotovic, *Nonlinear and adaptive control design* (John Wiley & Sons, 1995).

[22] H. Khalil, Adaptive output feedback control of nonlinear systems represented by input-output models, *33th IEEE Conf. on Decision and Control*, 1994, 199–204.

[23] M. Spong, S. Hutchinson, and M. Vidyasagar, *Robot modeling and control* (John Wiley & Sons, 2006).

Biographies

Hamed Jabbari Asl received the M.Sc. in control engineering in 2007 from Tabriz University. From 2011 to 2012, he was a visiting scholar at the Department of Computer, Control, and Management Engineering, La Sapienza, University of Rome. Currently, he is a Ph.D. student in the Department of Electrical Engineering, Iran University of Science and Technology. His areas of interest include robotics, vision-based control, adaptive and robust control.

Giuseppe Oriolo received the Ph.D. in control engineering in 1992 from Sapienza University of Rome. From 1990 to 1991 he was visiting scholar at the Center for Robotic Systems, University of California at Santa Barbara. In 1994, he joined the Department of Computer, Control, and Management Engineering, La Sapienza, University of Rome, where he is currently an associate professor of automatic control and robotics, as well as the director of the Robotics Lab. He was an associate editor of the IEEE Transactions on Robotics from 2001 to 2005, and the organization chair for the 2007 IEEE International Conference on Robotics and Automation. Since January 2009, he has been the editor of the IEEE Transactions on Robotics.

Hossein Bolandi received the Ph.D. in control engineering in 1990 from George Washington University. Currently, he is an associate professor in the Department of Electrical Engineering at Iran University of Science and Technology.