Robotica (2014) volume 32, pp. 43–61. © Cambridge University Press 2013
doi:10.1017/S0263574713000611

Vision-based autonomous hovering for a miniature quad-rotor

J. E. Gomez-Balderas†, S. Salazar‡,*, J. A. Guerrero§ and R. Lozano‡¶

†*GIPSA-Lab, UMR 5216 CNRS, Grenoble, France*
‡*UMI LAFMIA 3175 CINVESTAV, Mexico*
§*HEUDIASYC UMR 7253 CNRS-UTC, France*
¶*HEUDIASYC UMR 7253 CNRS-UTC, France*

(Accepted May 30, 2013. First published online: July 19, 2013)

SUMMARY
In this paper, a vision-based scheme for the autonomous hovering of a miniature quad-rotor is developed. Cameras are used to estimate the position and the translational velocity of the vehicle. The dynamic model of the miniature quad-rotor is developed using the Newton–Euler approach. A nonlinear controller based on a separated saturation control strategy for a miniature quad-rotor is presented. To validate the theoretical results, an embedded control system for the miniature quad-rotor has been developed. Thus, the analytic results are supported by experimental tests. Experimental results have validated the proposed control strategy.

KEYWORDS: Vision-based position estimation; Optical flow; Quad-rotor UAV; Nonlinear control; Embedded control system.

1. Introduction

The study and development of unmanned aerial vehicles (UAVs) have matured over recent years due to the fact that they can be used in a wide variety of applications, ranging from environmental monitoring in civil applications to surveillance and homeland security in military applications. Hovering platforms have the operational flexibility of being able to hover as well as to take off and land vertically. Currently, many researchers from various disciplines are working to improve the performance of autonomous hovering rotorcrafts.

We are interested in the stabilization of a miniature quad-rotor using a vision-based control approach. The idea is to estimate the position of the miniature quad-rotor with respect to a visual target placed on a wall using a camera. A second camera pointing downward is used to estimate the translational velocity of the miniature quad-rotor. A nonlinear controller based on separated saturations using vision-based position and velocity estimation is proposed in this paper. When working with vision-based control, also known as visual servoing, useful information can be extracted from a vision sensor to control the motion of a mobile robot with respect to the environment. In this paper, we propose a control scheme capable of working at a sampling rate compatible with respect to the desired bandwidth of the closed-loop system, we have a visual system with slow sampling rate and the inertial measurements unit (IMU), which operates at a fast sampling rate. The main problem in natural landmark navigation is to detect and match characteristic features from sensory inputs. The selection of features is important since it will determine the complexity in features description, detection and matching. Researchers have used different kinds of patterns to be recognized by systems based on computer vision.[1–3] The use of a target for autonomous landing has been applied in refs. [4–6].

* Corresponding author. E-mail: sergio.salazar.cruz@gmail.com

https://doi.org/10.1017/S0263574713000611 Published online by Cambridge University Press

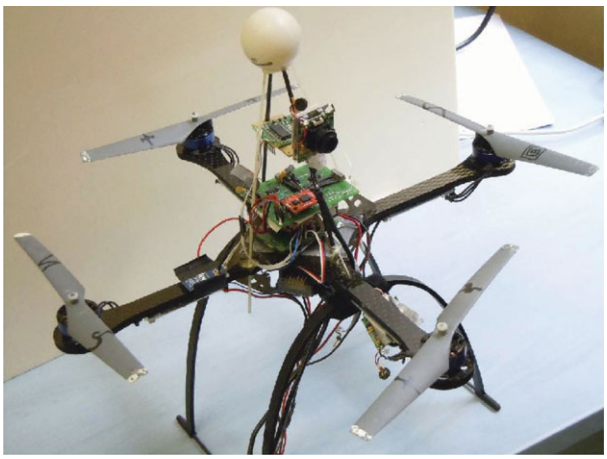


Fig. 1. (Colour online) Miniature quad-rotor with the embedded control system.

In ref. [4], a vision-based fixed-wing UAV landing system is proposed using a known marker. It has been shown that it is possible to obtain a precise altitude estimation using an on-board camera and a structured target.

A 3D vision system for estimating the height of a UAV is proposed in ref. [5]. This system is insensitive to the attitude for landing and the 3D vision system cannot measure distances smaller than 30 cm, producing a tracking error of 20 cm when landing. In ref. [6], a vision system consisting of a micro radio frequency transmitter and a miniature vision receiving antenna has been adapted to be used for acquiring real-time images without a vision control system.

A detailed summary of progress and several open problems in the vision-based estimation for UAVs may be found in refs. [7–9]. A different approach that uses video sequences to estimate the motion of UAV flight has been described in ref. [10]. Some tracking strategies using different targets are presented in refs. [11] and [12]. Caballero et al. [13] have shown that it is possible to determine the pose of UAVs using online mosaicking over the same area. In ref. [14], a two-camera method has been introduced for estimating the full six-degrees-of-freedom pose of the helicopter, one of these cameras is located on-board the helicopter, and the other camera is located on the ground.

Several approaches to nonlinear control of a miniature quad-rotor can be found in the literature.[15–18]

In this approach, the dynamical model is decoupled into lateral and longitudinal dynamics. This consideration is valid in stationary flights and small bounded angular displacements. The proposed control law is based on separated saturations in order to guarantee the state boundedness. Positive functions are used in the stability analysis. It has been shown in ref. [31] that this control algorithm converges faster than nested saturation control.

The present paper presents a vision-based control approach for the autonomous hovering of a quad-rotor shown in Fig. 1. The nonlinear controller is based on the separated saturations approach, which takes into account the *a priori* bound on the input amplitude. This approach uses a single camera to compute the 3D position of the vehicle with respect to a reference object placed on the wall. A second camera pointing downward has been mounted on the miniature quad-rotor to estimate the translational velocity of the vehicle using a Lucas–Kanade optical flow (OF) algorithm. Due to the limited computing capability on the UAV, on-board camera image processing was not an option. Therefore, the image processing computation has been carried out on the ground station. The approach used in this paper uses a webcam attached at the top of the UAV airframe pointing forward. The UAV position estimation algorithm uses a red color square in the image plane as input and computes the position of the UAV. The distance to the target on the wall is obtained with an accuracy of ± 2 cm. A real-time platform was developed and the miniature quad-rotor was able to hover autonomously.

This paper is organized as follows. Section 2 presents the dynamical model of the proposed architecture. In Section 3, the nonlinear control design is presented. The computer vision algorithm for position measurement and the velocity estimation using OF are described in Section 4. The embedded platform used to validate theoretical results is described in Section 5. Experimental results are presented in Section 6. Section 7 finally presents conclusions and future work.

https://doi.org/10.1017/S0263574713000611 Published online by Cambridge University Press

2. Dynamic Model of the Quad-Rotor

The UAV dynamical model can be represented as follows (see ref. [19]):

$$\dot{\Phi} = H(\Phi)\,\boldsymbol{\omega}^b_{b/e}, \tag{1}$$

$$\dot{\boldsymbol{\omega}}^b_{b/e} = \left(J^b\right)^{-1}\left[\mathbf{M}^b_A - \Omega^b_{b/e} J^b \boldsymbol{\omega}^b_{b/e}\right]. \tag{2}$$

The vehicle's center of mass, CM, is assumed to be at the body frame origin, F_b. The angular velocity in terms of the body system is given by $\boldsymbol{\omega}^b_{b/e} = [\,P\ Q\ R\,]^T$ and its cross-product matrix is denoted by $\Omega^b_{b/e}$. The angular velocity in the local inertial system has components $\dot{\Phi} = [\,\dot{\phi}\ \dot{\theta}\ \dot{\psi}\,]^T$. The rotation matrix from F_e to F_b is denoted by $\mathbf{C}_{b/n}$.

The rotation matrix from the inertial frame to the body frame is as follows:

$$\mathbf{C}_{b/n} = \begin{bmatrix} c\theta c\psi & -c\phi s\psi + s\phi c\theta c\psi & s\phi s\psi + c\phi s\theta c\psi \\ c\theta s\phi & c\phi c\psi + s\phi s\theta s\psi & -s\phi c\theta + c\phi s\theta s\psi \\ -s\theta & s\phi c\theta & c\phi c\theta \end{bmatrix}.$$

The set of attitude equations can be obtained using (1) and (2). The transformation of the components of the angular velocity generated by a sequence of Euler rotations from the body to the local reference system is written as follows:

$$H(\Phi) = \begin{bmatrix} 1 & t\theta s\phi & t\theta c\phi \\ 0 & c\phi & -s\phi \\ 0 & s\phi/c\theta & c\phi/c\theta \end{bmatrix}, \tag{3}$$

where s, c and t are used to denote the sin, cos and tan trigonometric functions, respectively. Then using (3), the *kinematic equation* (1) can be rewritten as

$$\begin{aligned} \dot{\phi} &= P + \tan\theta\,(Q\sin\theta + R\cos\phi), \\ \dot{\theta} &= Q\cos\phi - R\sin\phi, \\ \dot{\psi} &= (Q\sin\phi + R\cos\phi)/\cos\theta. \end{aligned}$$

The term $\mathbf{J}^b$ in (2) represents the inertia matrix and is defined by

$$\mathbf{J}^b = \begin{bmatrix} J_x & J_{xy} & J_{xz} \\ J_{yx} & J_y & J_{yz} \\ J_{zx} & J_{zy} & J_z \end{bmatrix}.$$

Since the miniature quad-rotor prototype is symmetrical in the xz and xy planes, the products of inertia J_{xy}, J_{yz} and J_{xz} vanish. Then, $\mathbf{J}^b$ can be written by

$$\mathbf{J}^b = \begin{bmatrix} J_x & 0 & 0 \\ 0 & J_y & 0 \\ 0 & 0 & J_z \end{bmatrix}.$$

https://doi.org/10.1017/S0263574713000611 Published online by Cambridge University Press

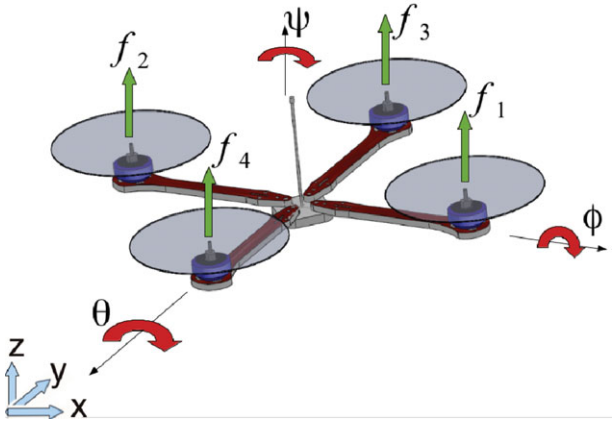


Fig. 2. (Colour online) The quad-rotor aircraft with the inertial and body frames.

The aerodynamics and thrust moments can be denoted by $\mathbf{M}^b_{A,T} = [\,\ell \quad m \quad n\,]^T$ and are shown in Fig. 2. Using the matrix of inertia and the moment vector, (2) yields

$$\dot{P} = \frac{\left(J_y - J_z\right) QR}{J_x} + \frac{\ell}{J_x},$$
$$\dot{Q} = \frac{\left(J_z - J_x\right) RP}{J_y} + \frac{m}{J_y},$$
$$\dot{R} = \frac{\left(J_x - J_y\right) PQ}{J_z} + \frac{n}{J_z}.$$

By differentiating (1) with respect to time, we get

$$\ddot{\Phi} = \dot{\mathbf{H}}\left(\Phi\right)\boldsymbol{\omega}^b_{b/e} + \mathbf{H}\left(\Phi\right)\dot{\boldsymbol{\omega}}^b_{b/e}. \tag{4}$$

Introducing (2) into (4),

$$\ddot{\Phi} = \dot{\mathbf{H}}\left(\Phi\right)\boldsymbol{\omega}^b_{b/e} + H\left(\Phi\right)\left(\mathbf{J}^b\right)^{-1}\left[\mathbf{M}^b_{A/T} - \Omega^b_{b/e}J^b\boldsymbol{\omega}^b_{b/e}\right]. \tag{5}$$

Let us propose the following control inputs:

$$\mathbf{M}^b_{A/T} \triangleq \Omega^b_{b/e}\mathbf{J}^b\boldsymbol{\omega}^b_{b/e} + \mathbf{J}^b\mathbf{H}\left(\Phi\right)^{-1}\left[\boldsymbol{\tau} - \dot{H}\left(\Phi\right)\boldsymbol{\omega}^b_{b/e}\right], \tag{6}$$

where $\tilde{\boldsymbol{\tau}} = [\,\tilde{\tau_\phi}\ \tilde{\tau_\theta}\ \tilde{\tau_\psi}\,]^T$. Then, (4) can be rewritten as

$$\ddot{\phi} = \tilde{\tau}_\phi, \tag{7}$$
$$\ddot{\theta} = \tilde{\tau}_\theta, \tag{8}$$
$$\ddot{\psi} = \tilde{\tau}_\psi. \tag{9}$$

Newton's second law is used to obtain the equations of translational motion in the inertial frame of reference as

$$\ddot{\mathbf{p}}^n_{CM} = \mathbf{C}_{n/b}\frac{\mathcal{T}}{m} + \mathbf{g}^n, \tag{10}$$

where the position of the CM in the NED (north–east–down) coordinate system with respect to the inertial frame origin, F_e, is given by $\mathbf{p}^n_{CM} = [\,x\ y\ z\,]^T$. The aerodynamic and thrust force vector in

https://doi.org/10.1017/S0263574713000611 Published online by Cambridge University Press

the body system is represented by $\mathcal{T} = [\, X_{A,T} \;\; Y_{A,T} \;\; Z_{A,T} \,]^T$. The aerodynamic and thrust forces in the body frame of reference are given by

$$\mathcal{T} = \sum_{i=1}^{4} f_i Z_{A,T}, \tag{11}$$

where $f_i = k_i \omega_i'^2$, where k_i is a constant related to each propeller and ω_i' is rotational velocity of each propeller. Thus, (10) can be rewritten as

$$m\ddot{x} = -\mathcal{T} \sin\theta, \tag{12}$$

$$m\ddot{y} = \mathcal{T} \cos\theta \sin\phi, \tag{13}$$

$$m\ddot{z} = \mathcal{T} \cos\theta \cos\phi - mg, \tag{14}$$

where m is the mass of the vehicle and g is the gravitational force.

3. Nonlinear Control Design

In this section, a nonlinear controller based on separated saturations[20] is presented to stabilize the quad-rotor in hover flight. The vertical control and heading can be linearized by using

$$\mathcal{T} \triangleq \frac{-k_1 \dot{z} - k_2 (z - z^d) + mg}{\cos\phi \cos\theta}, \tag{15}$$

$$\tilde{\tau}_\psi \triangleq -k_3 \dot{\psi} - k_4 (\psi - \psi^d), \tag{16}$$

where k_1, k_2, k_3 and k_4 are positive constants; z^d and ψ^d are the desired altitude and heading, respectively. Note that from (9) and (16), and assuming that $\psi^d = 0$, it follows that $\psi \to 0$. Using the above equations, the lateral dynamic model is represented by

$$\ddot{y} = mg \tan\phi, \tag{17}$$

$$\ddot{\phi} = \tilde{\tau}_\phi, \tag{18}$$

which represents four integrators in cascade. The technique based on separated saturations is used to stabilize the four integrators in cascade. The lateral controller is given by

$$\tilde{\tau}_\phi = -\sigma_{g_1}(\dot{\phi}) - \sigma_{g_2}(\dot{\phi} + \phi) - \sigma_{g_3}(\dot{\phi} + 2\phi + \dot{y}) - \sigma_{g_4}(\dot{\phi} + 3\phi + 2\dot{y} + y), \tag{19}$$

where σ_{g_i} represents a saturation function. This control law guarantees that $\dot{\phi}, \phi, \dot{y}$ and y converge to zero. See the next subsection.

Using (15), the longitudinal dynamic model is represented by the following set of equations:

$$\ddot{x} = \frac{-mg \tan\theta}{\cos\phi}, \tag{20}$$

$$\ddot{\theta} = \tilde{\tau}_\theta. \tag{21}$$

It is worth mentioning that using control law (19) implies that $\phi \to 0$, which in turns implies that $\cos\phi \to 1$. Then, assuming that $\tan\theta \approx \theta$, the above system has again four integrators in cascade and the corresponding longitudinal controller is given as

$$\tilde{\tau}_\theta = -\sigma_{g_1}(\dot{\theta}) - \sigma_{g_2}(\dot{\theta} + \theta) - \sigma_{g_3}(\dot{\theta} + 2\theta + \dot{x}) - \sigma_{g_4}(\dot{\theta} + 3\theta + 2\dot{x} + x). \tag{22}$$

The proposed control law guarantees that $\dot{\theta}, \theta, \dot{x}$ and x converge to zero.

https://doi.org/10.1017/S0263574713000611 Published online by Cambridge University Press

3.1. Stability control

The control strategy used in the previous section to control the longitudinal and lateral dynamics of the aircraft is based on separated saturations. In this subsection, the stability proof is presented. Equations (17), (18), (20) and (21) reduce to the following system of four integrators in cascade:

$$w^{(4)}(t) = u(t). \tag{23}$$

We will define the following change of variables:

$$z_1 = w^{(3)}, \tag{24}$$

$$z_2 = z_1 + \ddot{w}, \tag{25}$$

$$z_3 = z_2 + \ddot{w} + \dot{w}, \tag{26}$$

$$z_4 = z_3 + \ddot{w} + 2\dot{w} + w, \tag{27}$$

and the following control law:

$$u = -\sigma_a(z_1) - \sigma_b(z_2) - \sigma_c(z_3) - \sigma_d(z_4), \tag{28}$$

where σ_η is a saturation function defined as

$$\sigma_\eta(s) = \begin{cases} \eta & s > \eta \\ s & -\eta \leq s \leq \eta. \\ -\eta & s < -\eta \end{cases}$$

We will show that the control law (28) will globally asymptotically stabilize the system (23) around the origin.

3.2. State boundedness

We will first prove that the state of the system is bounded. Let us consider the following positive function $V_1 = \frac{1}{2}z_1^2$, where $z_1 = w^{(3)}$ (see (24)). Thus, $\dot{z}_1 = w^{(4)} = u(t)$ and so (see (28))

$$\dot{V}_1 = z_1 \left(-\sigma_a(z_1) - \sigma_b(z_2) - \sigma_c(z_3) - \sigma_d(z_4)\right). \tag{29}$$

Note that if $|z_1| > a$ and $a > b + c + d$, then $\dot{V}_1 < 0$. Thus, there exists a finite time T_1 such that for $t > T_1$ then $|z_1| < a$. For $t > T_1$, the control law (28) becomes

$$u = -z_1 - \sigma_b(z_2) - \sigma_c(z_3) - \sigma_d(z_4).$$

Since $z_2 = z_1 + \ddot{w}$ (see (25)), it follows that

$$\dot{z}_2 = w^{(4)} + z_1 = -\sigma_b(z_2) - \sigma_c(z_3) - \sigma_d(z_4).$$

Define the positive function $V_2 = \frac{1}{2}z_2^2$, then

$$\dot{V}_2 = -z_2 \left(\sigma_b(z_2) + \sigma_c(z_3) + \sigma_d(z_4)\right). \tag{30}$$

Note that if $|z_2| > b$ and $b > c + d$, then $\dot{V}_2 < 0$. We conclude that there exists a finite time T_2 such that for $t > T_2$ then $|z_2| < b$. For $t > T_2$, the control law becomes

$$u = -z_1 - z_2 - \sigma_c(z_3) - \sigma_d(z_4).$$

Given that $z_3 = z_2 + \ddot{w} + \dot{w}$ (see (26)), we then have

$$\dot{z}_3 = \dot{z}_2 + z_2 = -\sigma_c(z_3) - \sigma_d(z_4).$$

https://doi.org/10.1017/S0263574713000611 Published online by Cambridge University Press

Let us consider the positive function $V_3 = \frac{1}{2}z_3^2$, then

$$\dot{V}_3 = -z_3\left(\sigma_c(z_3) + \sigma_d(z_4)\right). \tag{31}$$

Similarly, if $|z_3| > c$ and $c > d$, then $\dot{V}_3 < 0$. Thus, there exists a finite time T_3 such that for $t > T_3$ we have $|z_3| < c$. For $t > T_3$, the control law becomes

$$u = -z_1 - z_2 - z_3 - \sigma_d(z_4).$$

Since $z_4 = z_3 + \ddot{w} + 2\dot{w} + w$ (see (27)), we have for $t > T_4$

$$\begin{aligned}\dot{z}_4 &= \dot{z}_3 + w^{(3)} + 2\ddot{w} + \dot{w}\\ &= \dot{z}_3 + z_2 + \ddot{w} + \dot{w}\\ &= \dot{z}_3 + z_3\\ &= -\sigma_d(z_4).\end{aligned}$$

Define the positive function $V_4 = \frac{1}{2}z_4^2$. Thus,

$$\dot{V}_4 = -z_4\left(\sigma_d(z_4)\right), \tag{32}$$

it follows that after a finite time T_4 we have $|z_4| < d$.

3.3. Convergence analysis

We will now prove that all the state w will converge to zero. Indeed, from (32) we conclude that $z_4 \to 0$. Similarly, from (31) it follows that $z_3 \to 0$. From (30), we conclude that $z_2 \to 0$ and finally from (29) it follows that $z_1 \to 0$.

4. Vision-Based Position Estimation

Computer vision plays an important role in a wide variety of applications such as text finding, color image segmentation,[21] and robotics.[22,23] In this section, we propose a computer vision algorithm to obtain the position of the miniature quad-rotor with respect to a visual target which in our case is a 28×28 cm red square. The algorithm proposed works in a 3D Euclidean space, in which measurements of lengths and angles do make sense.[24] We use a color extractor such that the detection of the target from the image is robust to noise and changes in luminance conditions. Using color segmentation, we avoid the problems caused by motion blur[25] in the image. Specially, we search the corner of our target and the image blur does not affect our estimation. Then, an edge detector algorithm is used to find the edges of the target in the image plane which is described next.

4.1. Edge detection algorithm

The Canny edge detection algorithm[26] produces thin fragments of image contours that can be controlled by a single smoothing parameter known as σ. The image is first smoothed with a Gaussian filter of spread σ and then the gradient magnitude and direction are computed at each pixel of the smoothed image. Gradient direction is used to thin the edges by suppressing any pixel response that is not higher than the two neighboring pixels on either side of it along the direction of the gradient; this is called non-maximum suppression. Once the gradient magnitudes are thinned, high magnitude contours are tracked. In the final aggregation phase, continuous contour segments are sequentially followed. Contour following is initiated only on edge pixels where the gradient magnitude meets a high threshold; however, once started, a contour may be followed through pixels whose gradient magnitude meets a lower threshold, usually about half of the higher starting threshold. After the edges are known, image regions can be detected from the segmented image. An image region is detected when edge segments or boundary segments close on themselves, forming a convex region. In particular, a rectangular object results in four straight line segments connected such that they form

https://doi.org/10.1017/S0263574713000611 Published online by Cambridge University Press

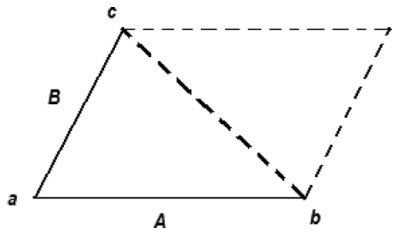


Fig. 3. Cross-product parallelogram.

a convex region. So far, we have settled the basis for image regions detection. We next describe the polygons obtained after color and edge detection algorithms and some useful polygon properties.

4.2. Polygons properties

We define a polygon as a region of a plane bounded by a finite collection of line segments forming a simple closed curve, let $v_0, v_1, v_2, \ldots, v_{n-1}$ be n points in the plane. Let $e_0 = v_0v_1, e_1 = v_1v_2, \ldots, e_i = v_iv_{i+1}, \ldots, e_{n-1} = v_{n-1}v_0$ be n segments connecting the points. Then these segments bound a convex polygon iff:

1. The intersection of each pair of adjacent segments in the cyclic ordering is the single point shared between them: $e_i \bigcap e_{i+1} = v_{i+1}$, for all $i = 0, \ldots, n-1$.
2. Nonadjacent segments do not intersect: $e_i \bigcap e_j = \emptyset$, for all $j \neq i + 1$.

The points v_i are called the vertices of the polygon, and the segments e_i are called its edges. Note that a polygon of n vertices has n edges.

The area of a polygon with vertices $v_0, v_1, \ldots, v_{n-1}$ labeled counterclockwise can be calculated as $A_P = A(v_0, v_1, v_2) + A(v_0, v_2, v_3) + \cdots + A(v_0, v_{n-2}, v_{n-1})$.

From linear algebra, we know that the magnitude of the cross product of two vectors is the area of the parallelogram they determine. If A and B are vectors with coordinates (A_x, A_y, A_z) and (B_x, B_y, B_z) in the x, y, and z directions, respectively, then $|A \times B|$ is the area of the parallelogram with sides A and B, as shown in Fig. 3.

For 3D vectors, $A_z = B_z = 0$, the cross product reduces to $(A_0B_1 - A_1B_0)$ and the area of a triangle A_T is given by

$$A_T = \frac{1}{2}(A_xB_y - A_yB_x). \tag{33}$$

Note that $A = b - a$ and $B = c - a$, where (a, b, c) are three arbitrary vertices, then (33) can be rewritten as

$$\begin{aligned} 2A_T &= a_xb_y - a_yb_x + a_yc_x - a_xc_y + b_xc_y - c_xb_y \\ &= (b_x - a_x)(c_y - a_y) - (b_y - a_y)(c_x - a_x). \end{aligned} \tag{34}$$

This is an expression for the area of the triangle as a function of the coordinates of its vertices. The area of a convex quadrilateral $Q = (a, b, c, d)$ may be written in two ways, depending on the two different triangulations as shown in Fig. 4:

$$A_Q = A(a, b, c) + A(a, c, d) = A(d, a, b) + A(d, b, c). \tag{35}$$

Writing out the expressions for the areas using Eq. (35) for the two terms of the first triangulation, we get

$$\begin{aligned} 2A_Q = {} & a_xb_y - a_yb_x + a_yc_x - a_xc_y + b_xc_y - c_xb_y \\ & + a_xc_y - a_yc_x + a_yd_x - a_xd_y + c_xd_y - d_xc_y. \end{aligned} \tag{36}$$

https://doi.org/10.1017/S0263574713000611 Published online by Cambridge University Press

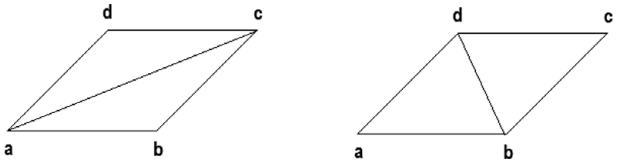


Fig. 4. The two triangulation of a convex quadrilateral.

4.3. Square detection algorithm

Once we have obtained four edge segments in the image plane, which intersect in four points, it is required to verify that the segments form a square. To verify this, the angle in each corner of the intersection of two edges is computed. By applying the triangulation of polygons, it is always possible to obtain two similar triangles from a square. Then, given the triangle Δ with edges A, B and C, the angle φ between the two edges B and C can be obtained by the following formula:

$$\varphi = \arccos \frac{|B|^2 + |C|^2 - |A|^2}{2 \cdot |B| \cdot |C|}, \tag{37}$$

where $|A|, |B|$ and $|C|$ denote the lengths of the edges A, B and C, respectively. Applying this formula at each vertex of the square, we obtain the four angles in the polygon. If these edges are perpendicular, we then verify the conditions for a convex region to be a square and its area can be obtained using formula (36). Once we have located the target on the image plane, we are interested in computing the distance to it and the translational velocity with respect to the target. The estimation of the distance to the target needs full knowledge of the camera parameters. The translational velocity is computed using an OF approach. The camera model and the OF estimation are described in the following sections.

4.4. Camera model

The classical geometric model of a camera is based on the pure perspective projection concepts. Camera calibration is composed basically in the computation of the extrinsic and intrinsic parameters.[32]

In the pinhole camera model, a projection point $\mathbf{p}_i = (x_i, y_i)^T$ in the camera image is related with a 3D point $\mathbf{P} = (X_W, Y_W, Z_W)^T$ relative to the world reference frame with coordinates $\mathbf{P}_c = (X, Y, Z)^T$ in the camera frame.

The perspective transformation matrix, which relates 3D world coordinates and 2D image coordinates (see Fig. 5), is given by

$$\lambda \begin{pmatrix} x_i \\ y_i \\ 1 \end{pmatrix} = \mathcal{M} \begin{pmatrix} X_W \\ Y_W \\ Z_W \\ 1 \end{pmatrix},$$

where λ is an arbitrary positive scalar and $\mathcal{M} = K(\mathbf{R} \quad \mathbf{t})$ is the standard projective matrix, also defined up to scale, with 11 free coefficients, 5 intrinsic parameters $(f_x, f_y, c_x, c_y, \eta)$ and 6 extrinsic parameters (the three angles defining the rotational matrix $\mathbf{R}$ and the three coordinates of the translational vector $\mathbf{t}$), which matches the number of independent coefficients of $\mathcal{M}$.

The matrix $\mathcal{M}$ can be written explicitly as a function of the intrinsic and extrinsic parameters of the camera:

$$\mathcal{M} = \begin{bmatrix} f_x R_1^T - f_x \cot\eta R_2^T + c_x R_3^T & f_x t_x - f_x \cot\eta t_y + c_x t_z \\ \frac{f_y}{\sin\eta} R_2^T + c_y R_3^T & \frac{lf}{\sin\eta} t_y + c_y t_z \\ R_3^T & t_z \end{bmatrix}, \tag{38}$$

where $\mathbf{R}_1^T, \mathbf{R}_2^T, \mathbf{R}_3^T$ denote the three rows of the $\mathbf{R}$ matrix and t_x, t_y and t_z are the coordinates of the vector $\mathbf{t}$ in the frame attached to the camera. Here, the angle η models the non-orthogonality between

https://doi.org/10.1017/S0263574713000611 Published online by Cambridge University Press

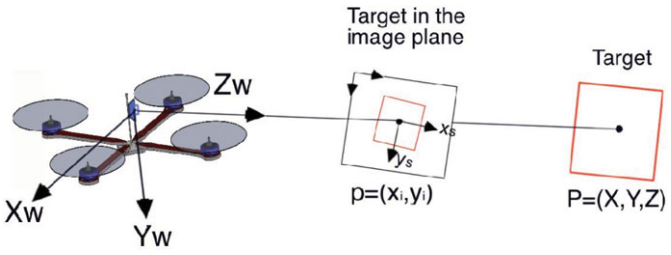


Fig. 5. (Colour online) Camera model of the quad-rotor.

the rows and columns in the image. The parameters (f_x, f_y) are horizontal and vertical scale factors expressed in pixels/unit length.

4.5. Optical flow measurement

We use the OF from image sequences to estimate the translational velocity of the quad-rotor. The estimate of the horizontal velocity will be used in the closed-loop feedback control to perform autonomous hover flights. In ref. [27], the rotational components of OF are compensated and ref. [28] uses an OF technique of phase correlation on images captured with a down-facing camera which is fitted with an omnidirectional lens and the images are transformed into the log-polar domain before the main computational step. There exist many different methods for computing the OF, such as intensity-based differential methods,[29] frequency-based filtering and correlation-based methods. In this paper, we have used the Lucas–Kanade pyramidal method, which is an intensity-based differential method.[30] This method has been chosen for its temporal persistence. To circumvent the problem of large and non-coherent motions, we can first track over larger spatial scales using an image pyramid and then refine the initial motion velocity assumptions by working our way down the levels of the image pyramid until we arrive at the raw image pixels. Furthermore, due to spatial coherence, the neighboring points in a scene on the floor belong to the same surface. They also have similar motion and project into nearby points on the image plane. A clear advantage of this method is that each residual OF vector $\mathbf{d}^L$ can be kept very small while computing a large overall pixel displacement vector $\mathbf{d}$. Assuming that each elementary OF computation step can handle pixel motions up to $d_{\max}$, then the overall pixel motion that the pyramidal implementation can handle becomes $d_{\max-\text{final}} = (2^{Lm+1} - 1)d_{\max}$.

Consider two discrete functions $I_1, I_2 \in R^{m_x \times n_y}$ representing two gray-scale images at different time instants, and let G_{p_i} be the gray-scale value of a particular pixel $p = (x_i, y_i)^T$. Then, the gray values for p_i which appear in two consecutive images are defined respectively as

$$G_{p_1} = I_1(x_1, y_1) \quad G_{p_2} = I_2(x_2, y_2),$$

where x_i and y_i are the row and column pixel coordinates, respectively. Given a specific image point $p_1 \in I_1$, the aim of the approach is to find another image point $p_2 \in I_2$ such that $G_{p_1} \approx G_{p_2}$. Moreover, the relationship between matched pixels p_1 and p_2 is given by

$$p_2 = p_1 + r = [\, x_1 + r_x \quad y_1 + r_y \,]^T,$$

where $r = [\, r_x \; r_y \,]^T$ defines the image displacement or OF and minimizes the following residual function:

$$\epsilon(r) = \sum_{x_p = x_{p_1} - w_x}^{x_{p_1} + w_x} \sum_{y_p = y_{p_2} - w_y}^{y_{p_2} + w_y} (I_1(p_1) - I_2(p_1 + r))^2,$$

where w_x and w_y are two integers that define the size of the integration window. The Lucas–Kanade OF algorithm has an adaptive integration window, which means that it is able to handle large pixel motions and this representation acts as a low-pass filter.

Let us assume that the camera is moving with translational velocity v and angular velocity ω while viewing an object at distance d and offset β from the direction of travel as depicted in Fig. 6. The OF

https://doi.org/10.1017/S0263574713000611 Published online by Cambridge University Press

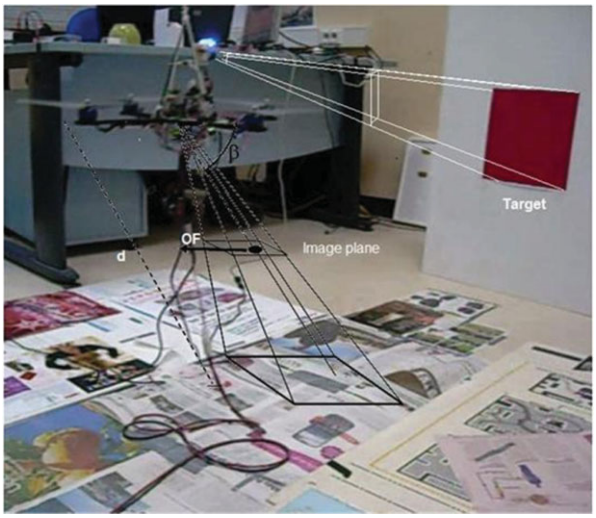


Fig. 6. (Colour online) Quad-rotor vision system using two cameras.

can be expressed as

$$OF = \frac{v}{d}\sin\beta - \omega. \tag{39}$$

The maximal OF is obviously generated when the plane that contains the features is perpendicular to the translational motion direction ($\beta = 90^\circ$). The velocity can be estimated by (39) as follows:

$$v = \frac{(OF + \omega)}{\sin\beta} d.$$

Note that singularities in the above equation appear when $\beta = 0^\circ$. Nevertheless, in our case the pitch and roll angles are very close to zero, which implies that $\beta = 90^\circ \pm 3^\circ$ as a maximal interval when the quad-rotor is appropriately stabilized at hover. The measurement of the angular velocity ω is obtained by using the gyro information on each axis. An altitude stabilization algorithm in the closed loop is used to keep the distance d constant and equal to some desired value. Since the IMU's sampling period is smaller (∼10 ms) than the OF sampling period (∼60 ms), a zero-order holder on the OF has been used in order to synchronize the estimation of the translational velocity. Note that no data fusion between the IMU and the visual system has been implemented.

4.6. Image rectification

The square target has four points named P_1, P_2, P_3 and P_4. The vector from the origin of the world coordinate system to the point P_i is labeled $\mathbf{P}_i$. The Euclidean distance between P_i and P_j is denoted by s_{ij}. The vector $(\mathbf{P}_i - \mathbf{P}_j)$ is denoted by $\mathbf{P}_{ij}$ for $i, j = 1, ..., 4, i \neq j$. The areas $A(P_i, P_j, P_k)$ of the triangles in this square can be computed as

$$\begin{aligned} A_1 &= A(P_2, P_1, P_4) = |\mathbf{P}_{21} \times \mathbf{P}_{24}|/2 \\ &= A(P_2, P_3, P_4) = |\mathbf{P}_{23} \times \mathbf{P}_{24}|/2, \\ A_2 &= A(P_1, P_4, P_3) = |\mathbf{P}_{14} \times \mathbf{P}_{13}|/2 \\ &= A(P_1, P_2, P_3) = |\mathbf{P}_{12} \times \mathbf{P}_{13}|/2. \end{aligned}$$

We call $\bar{p}_i^f$ the vector from the center of projection C to the point p_i in the image plane and the vector joining C to P_i is denoted $\bar{\mathbf{P}}_i^f$. The unit vector collinear to $\bar{\mathbf{P}}_i^f = (x_i, y_i, f)^t$ is

$$\mathbf{u}_i = (u_{ix}, u_{iy}, u_{iz})^t = \bar{\mathbf{p}}_i^f / \left\| \bar{\mathbf{p}}_i^f \right\| = (x_i, y_i, f)^t / F_i,$$

https://doi.org/10.1017/S0263574713000611 Published online by Cambridge University Press

where

$$F_i = \sqrt{x_i^2 + y_i^2 + f^2}.$$

Using the image coordinates of the points p_i, we can find the area of the triangle in the image plane using

$$B_1 = x_2(y_4 - y_1) + y_2(x_1 - x_4) + y_1x_4 - x_1y_4,$$
$$B_2 = x_2(y_3 - y_1) + y_2(x_1 - x_3) + y_1x_3 - x_1y_3.$$

The B_i values are twice the area of the triangle formed by three of the four image points corresponding to the subscripts of the coordinates. If the optical axis of the vision system is normal to the target plane, the target and its image are similar, i.e.

$$\frac{r_{12}}{s_{12}} = \frac{r_{13}}{s_{13}} = \frac{r_{14}}{s_{14}} = \frac{r_{23}}{s_{23}} = \frac{r_{24}}{s_{24}} = \frac{r_{34}}{s_{34}} = \frac{f}{f - Z_W},$$
$$\frac{B_1}{A_1} = \frac{B_2}{A_2},$$
$$Z_1^c = Z_2^c = Z_3^c = Z_4^c = Z_W.$$

The parameter r_{ij} denotes the distance between p_i and p_j. We have assumed that the vision system is always normal to the target plane, i.e. we have neglected projective and affine distortions from camera images. In our approach, the projective distortion was first removed and then the affine distortions were removed, as explained in the next section.

4.7. Removing projective distortion

It is possible to recover the affine properties from images by using the transformation matrix $\mathbf{H}_1$, which maps the vanishing line back into a line at infinity $\mathbf{l}^\infty$. We take two pairs of images lines $\mathbf{l}^{(1)}$, $\mathbf{l}^{(2)}$, $\mathbf{m}^{(1)}$, $\mathbf{m}^{(2)}$, which correspond to the lines connecting the points p_1, p_2 parallel to p_3, p_4, and p_1, p_3 parallel to p_2, p_4. In a perspective distorted image, these two sets of parallel lines will intersect at points $\mathbf{p}^{(1)}$ and $\mathbf{p}^{(2)}$. The line formed by connecting $\mathbf{p}^{(1)}$ and $\mathbf{p}^{(2)}$ is the vanishing line $\mathbf{l} = (l_1, l_2, l_3)^T$, in homogeneous coordinates:

$$\mathbf{p}^{(1)} = \mathbf{l}^{(1)} \times \mathbf{l}^{(2)},$$
$$\mathbf{p}^{(2)} = \mathbf{m}^{(1)} \times \mathbf{m}^{(2)},$$
$$\mathbf{l} = \mathbf{p}^{(1)} \times \mathbf{p}^{(2)}.$$

Therefore, by applying transformation $\mathbf{H}_1$,

$$\mathbf{H}_1 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ l_1 & l_2 & l_3 \end{pmatrix},$$

the vanishing line will be mapped into the line at infinity $\mathbf{l}^\infty = (0, 0, 1)^T$. This can be verified as

$$\mathbf{H}_1^{-T} = \begin{pmatrix} 1 & 0 & -l_1/l_3 \\ 0 & 1 & -l_2/l_3 \\ 0 & 0 & 1/l_3 \end{pmatrix}$$

and $\mathbf{H}_1^{-T}\mathbf{l} = (0, 0, 1)^T$; applying $\mathbf{H}_1$ to camera images, we obtain the affinely rectified image $\mathbf{X}_a = \mathbf{H}_1\mathbf{X}_c$.

4.8. Removing affine distortion

If we obtained the affinely rectified image $\mathbf{X}_a$, we should find the affine transformation matrix:

$$\mathbf{H}_2 = \begin{pmatrix} \mathbf{A} & \mathbf{t} \\ \mathbf{0} & 1 \end{pmatrix},$$

https://doi.org/10.1017/S0263574713000611 Published online by Cambridge University Press

such that $\mathbf{X}_a = \mathbf{H}_2\mathbf{X}_s$, where $\mathbf{X}_s$ is the scene image in the real world. In the target, we have a pair of real orthogonal lines, $\mathbf{l}\perp\mathbf{m}$ and $\mathbf{l}'$, $\mathbf{m}'$ are the transformed lines under affine transformation $\mathbf{H}_2$. By orthogonality, we know that

$$(l_1/l_3, l_2/l_3)(m_1/m_3, m_2/m_3)^T = 0$$

then

$$l_1m_1 + l_2m_2 = \mathbf{l}^T\mathbf{C}^*_\infty\mathbf{m} = 0,$$

where $\mathbf{C}^*_\infty$ is the dual degenerate conic. Since $\mathbf{C}^{*\prime}_\infty = \mathbf{H}_2\mathbf{C}^*_\infty\mathbf{H}_2^T$, we have

$$\mathbf{l}^T\mathbf{C}^*_\infty\mathbf{m} = \mathbf{l}'^T\mathbf{H}_2\mathbf{H}_2^{-1}\mathbf{C}'^*_\infty\mathbf{H}_2^{-T}\mathbf{H}_2^T\mathbf{m}' = \mathbf{l}'^T\mathbf{C}'^*_\infty\mathbf{m}' = 0.$$

Therefore,

$$\mathbf{l}'^T\mathbf{C}'^*_\infty\mathbf{m}' = \mathbf{l}'^T\mathbf{H}_2\mathbf{C}^*_\infty\mathbf{H}_2^T\mathbf{m}$$

$$= \mathbf{l}'^T\begin{pmatrix}\mathbf{A} & \mathbf{t}\\ \mathbf{0} & 1\end{pmatrix}\begin{pmatrix}I & \mathbf{0}\\ \mathbf{0} & 0\end{pmatrix}\begin{pmatrix}\mathbf{A}^T & \mathbf{0}\\ \mathbf{t}^T & 1\end{pmatrix}\mathbf{m}'$$

$$= \mathbf{l}'^T\begin{pmatrix}\mathbf{A}\mathbf{A}^T & \mathbf{0}\\ \mathbf{0} & 0\end{pmatrix}\mathbf{m}'.$$

We have

$$(l'_1, l'_2)\mathbf{A}\mathbf{A}^T(m'_1, m'_2)^T = 0. \tag{40}$$

In order to obtain $\mathbf{A}$, let $\mathbf{S} = \mathbf{A}\mathbf{A}^T$, where

$$\mathbf{S} = \begin{pmatrix}S_{11} & S_{12}\\ S_{12} & 1\end{pmatrix}, \tag{41}$$

since $\mathbf{S}$ is a symmetric matrix considering the scale problem. Introducing (41) into (40) we get

$$\left(l'_1m'_1, l'_1m'_2 + l'_2m'_1\right)\begin{pmatrix}s_{11}\\ s_{12}\end{pmatrix} = -l'_2m'_2. \tag{42}$$

From (42), it is clear that in order to solve the matrix S, with two unknown parameters, two pairs of orthogonal lines are required $\mathbf{l}^{(1)}\perp\mathbf{m}^{(1)}$ and $\mathbf{l}^{(2)}\perp\mathbf{m}^{(2)}$.

Since the symmetric matrix $\mathbf{S}$ can be written as $\mathbf{S} = \mathbf{U}\mathbf{D}\mathbf{U}^T$, where $\mathbf{U}^{-1} = \mathbf{U}^T$, we can get $\mathbf{A} = \mathbf{U}\sqrt{\mathbf{D}}\mathbf{U}^T$. Now, we know $\mathbf{H}_2$ and the restored image is $\mathbf{X} = \mathbf{H}_2^{-1}X_a$. In fig. 7 we show the result image after remove projective and affine distortions.

4.9. Solving the 3D localization problem

In order to estimate the miniature quad-rotor position and velocity, we propose to use two cameras orthogonally mounted on the miniature quad-rotor structure. The miniature quad-rotor position is obtained as follows. First, a color segmentation is carried out on the original RGB image taken by the frontal camera which is observing the red square target, then we rectify the image using the methods described above. Next, the Canny algorithm is applied for edge detection to obtain the boundaries of the regions of interest. The boundaries are formed by a finite collection of n line segments $e_0 = v_0v_1, e_1 = v_1v_2, \ldots, e_i = v_iv_{i+1}, \ldots, e_{n-1} = v_{n-1}v_0$, connected by n image points $v_0, v_1, v_2, \ldots, v_{n-1}$, where $v_i = (x_i, y_i)$. Using these points, we can obtain four vertices and four edges. The next step is to verify the angle on each corner of the quadrilateral, using (37). If this quadrilateral is a square, it is necessary to compute the square area using (33). Once the area of the square is known, and the intrinsic parameters values of the camera (38) are also known, we

https://doi.org/10.1017/S0263574713000611 Published online by Cambridge University Press

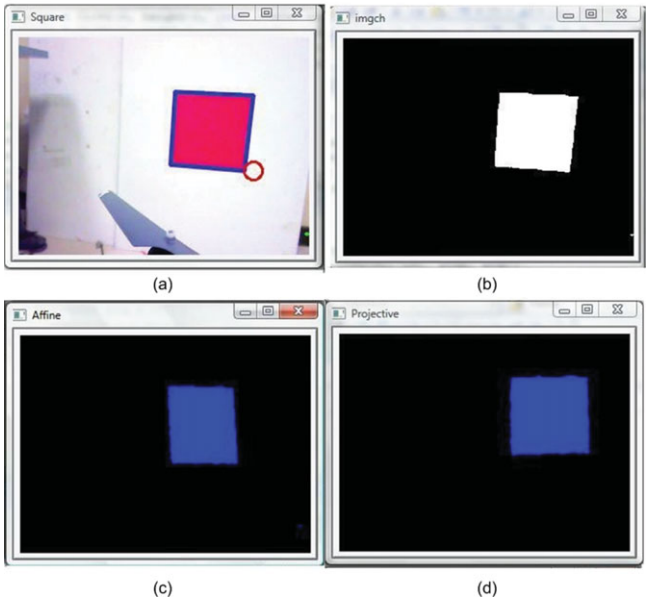


Fig. 7. (Colour online) Frontal camera distortion removal in (a) original image, (b) image after color filter, (c) image removing projective distortion and (d) image removing affine distortion.

use similar triangle properties to obtain the distance Z_W between the camera and the target. Using (38), we know that $f_x = f S_x$ and $f_y = \frac{f S_y}{\sin \eta}$ then $S_x = \frac{f_x}{f}$ and $S_y = \frac{f_y \sin \eta}{f}$, the four 3D points of our target are called P_1, P_2, P_3 and P_4 in the clockwise sense. These points have corresponding points $p_1 = (x_i, y_i)$, $p_2 = (x_{i+1}, y_i)$, $p_3 = (x_{i+1}, y_{i+1})$ and $p_4 = (x_i, y_{i+1})$ in image coordinates. Assuming knowledge of h, the height of the square target, and the focal length f, we can obtain

$$Z = \frac{f h}{(y_i - y_{i+1}) S_y}. \tag{43}$$

Other information can also be obtained, such as the central point of the square, which is equal to the point at which the two diagonals of the square intersect, called (x_{ci}, y_{ci}) in the image plane. Using perspective projection properties in the frontal pinhole camera model, we obtain

$$X = \frac{x_{ci} Z}{f} \tag{44}$$

and

$$Y = \frac{y_{ci} Z}{f}. \tag{45}$$

The miniature quad-rotor translational velocity is estimated by using a Lucas–Kanade OF algorithm on the camera that is pointing downward. The proposed position algorithm estimates the 3D position of the miniature quad-rotor $\mathbf{p}^n_{CM} = [\, x \;\; y \;\; z \,]^T = [\, X \;\; Z \;\; Y \,]^T$.

5. Embedded Architecture

The miniature quad-rotor is powered by four brushless motors with fixed pitch propellers. Altitude is controlled by the collective thrust of the motors. The pitch motion is achieved through the angular velocity difference between motors M_1 and M_2, as shown in Fig. 2. The vehicle roll position is controlled by the angular velocity difference between motors M_3 and M_4. The total weight of the vehicle is about 380 g, with a flight endurance of 15 min approximately.

https://doi.org/10.1017/S0263574713000611 Published online by Cambridge University Press

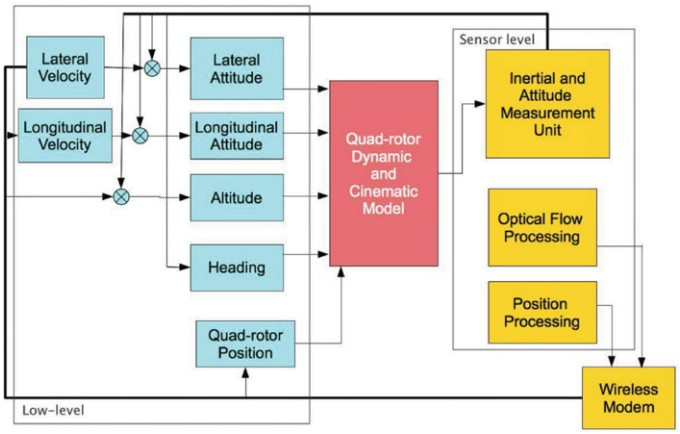


Fig. 8. (Colour online) Block diagram of the quad-rotor controller.

Theoretical results obtained were incorporated into an autopilot control system using an architecture based on a 29-MHz Rabbit microcontroller with 512-Kb flash and 512-Kb RAM. These microcontrollers are capable of handling floating point operations and multitasking processing virtually due to the enhancement compiler Dynamic C.[33] We have built an IMU using accelerometers, gyros and a compass to obtain the roll, pitch and yaw angles and angular rates.

The IMU information and visual information are sent to the microcontroller, which also reads reference control inputs from an operator through a serial wireless modem. The microcontroller subsequently combines this information to compute the control input and sends the control corrections to the motors through an I2C serial port. Hence, the brushless motor speed controller or booster is handled by the I2C port of the microcontroller. Figure 8 shows a block diagram of the basic architecture.

6. Experimental Results

As observed above, the control strategy for this vehicle is a state feedback control law where the angular position variables and its derivatives are obtained from the IMU. The position and velocity of the vehicle are estimated using a vision-based algorithm explained in the previous sections. Our algorithm was tested in a PC with an Intel Core Duo 2.10-GHz processor. We use the library OpenCV, with a frequency of 15 frames per second approximately. The closed-loop system includes a low-pass filter of the translational velocity, and a saturation function to limit the translational velocity. We have modified the contrast of the image to obtain an improved estimation of characteristic points. In these experiments, the reference value for $y^d = Z^d$ in frontal camera coordinates was set to 130 cm, which is the desired horizontal distance between the target and the miniature quad-rotor vehicle. The lateral reference was set to $x^d = X^d = 120$ cm, which is the desired displacement of the miniature quad-rotor over the roll axis. The desired altitude was set to $z^d = Y^d = 130$ cm. To adjust the control parameters, several flight tests were carried out until good performance of the vehicle was obtained. Figures 9–11 show the performance of the longitudinal and lateral subsystems using the nonlinear controller with separated saturations. Several perturbations have been induced to test the robustness of our system to mild perturbations such as light winds and uncertainties due to the camera and the environment. The experimental validation of the proposed method can be seen in refs. [34] and [35]. In Figs. 9–11, where the red line represents the desired positions of the quad-rotor, we show the result of our position algorithm in 60 s of hover flight. Figure 9 shows different disruptions caused manually over the x-axis of the quad-rotor. Figure 10 presents different disruptions caused manually over the y-axis of the quad-rotor. Figure 11 shows the quad-rotor manually perturbed in altitude. The velocities in the x- and y-axes of the miniature quad-rotor were estimated using OF and are shown in Figs. 12 and 13. Figure 12 shows the estimation velocity over the x-axis of the quad-rotor. Figure 13 shows velocity estimation over the y-axis of the miniature quad-rotor. The pitch and roll angles are shown in Figs. 14 and 15. This experiment shows that the nonlinear control law can recover from large orientation perturbations.

https://doi.org/10.1017/S0263574713000611 Published online by Cambridge University Press

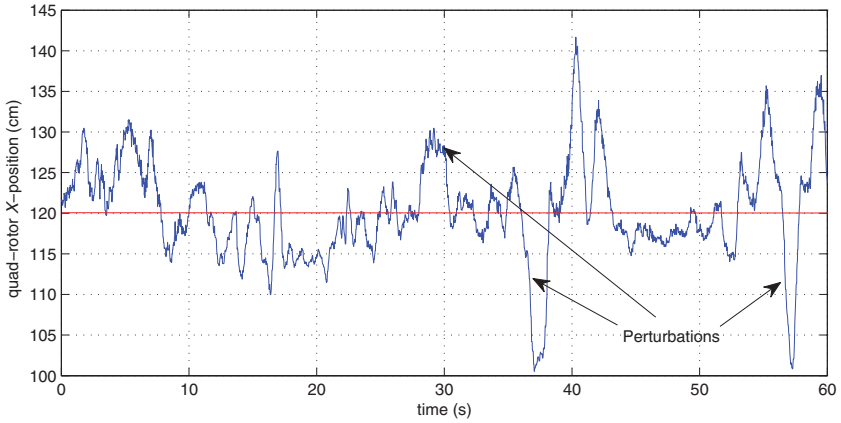


Fig. 9. (Colour online) X position of the quad-rotor. The quad-rotor is manually perturbed over its x-axis.

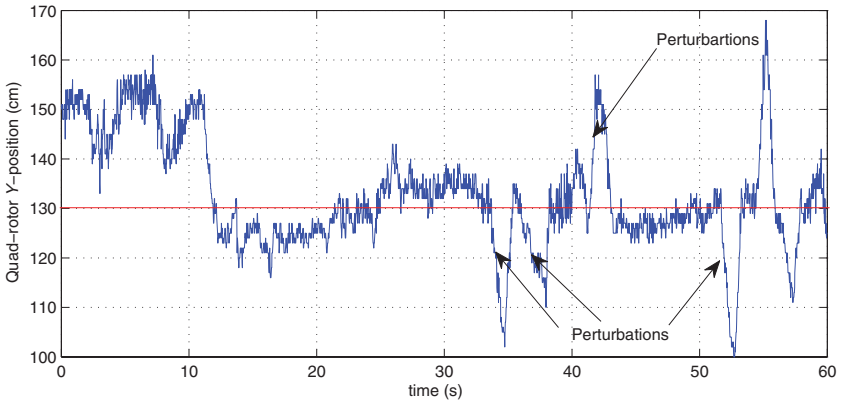


Fig. 10. (Colour online) Y position of the quad-rotor. The quad-rotor is manually perturbed over its y-axis.

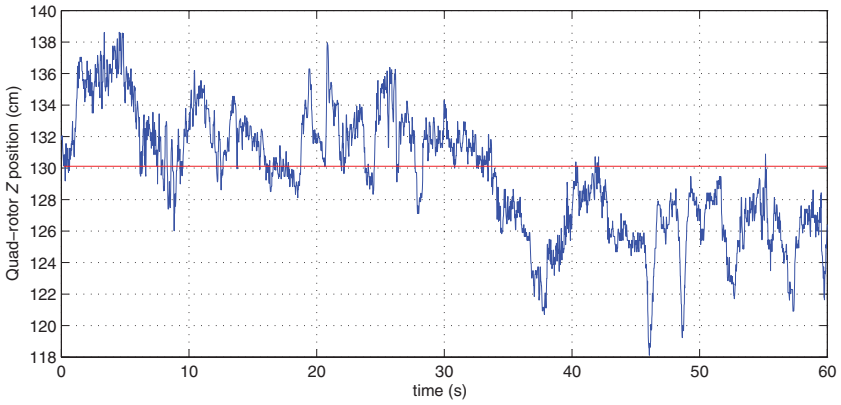


Fig. 11. (Colour online) The Z position of the quad-rotor. Quad-rotor altitude performance in the presence of manual perturbations.

7. Conclusion and Future Work

A nonlinear dynamical model of the miniature quad-rotor has been presented using the Newton–Euler formulation. A nonlinear control scheme based on separated saturation functions was proposed for stabilizing a miniature quad-rotor. Position was obtained using a frontal camera looking at a target placed on a wall. The miniature quad-rotor velocity was estimated using a camera pointing downward running an OF algorithm. Experimental tests showed that the miniature quad-rotor performed well at hover flight using the proposed vision-based control system. Future work in this area includes

https://doi.org/10.1017/S0263574713000611 Published online by Cambridge University Press

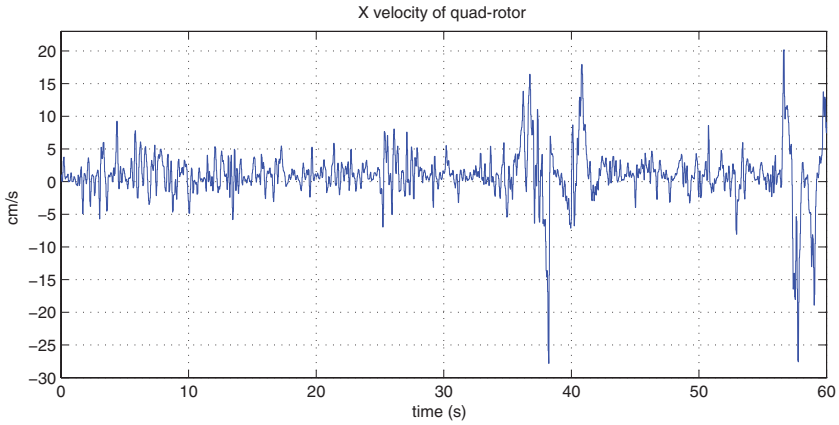


Fig. 12. (Colour online) X velocity of the miniature quad-rotor. The velocity estimations over the x-axis are shown with some perturbations during quad-rotor hover (cm/s).

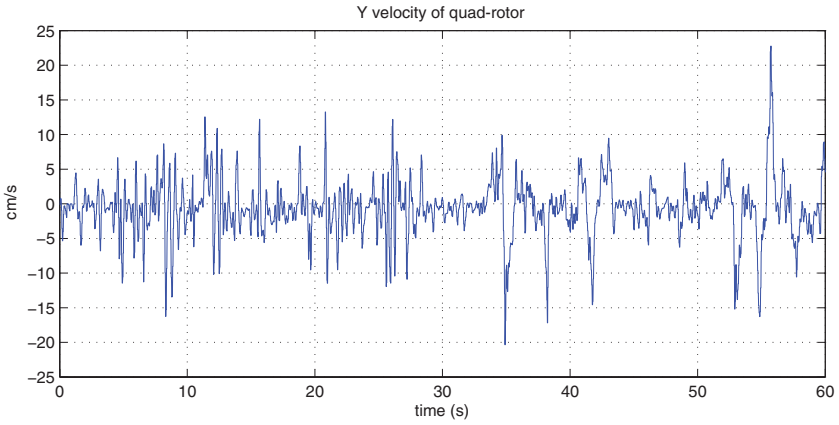


Fig. 13. (Colour online) Y velocity of the miniature quad-rotor. The velocity estimations over y-axes are showed with some perturbations during quad-rotor hover (cm/s).

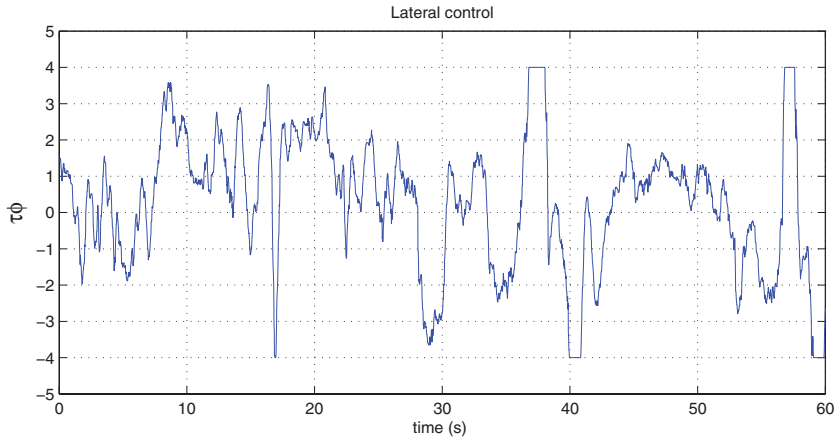


Fig. 14. (Colour online) Lateral control τ_ϕ in the presence of manual perturbations.

tracking a wall for inspection purposes using the vision system as well as tracking a moving visual target. This work highlights the potential of the computer vision-based control strategies for mini UAVs.

https://doi.org/10.1017/S0263574713000611 Published online by Cambridge University Press

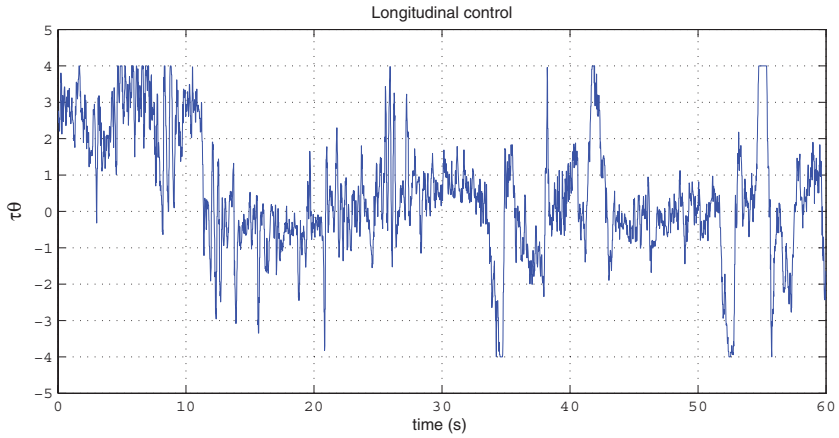


Fig. 15. (Colour online) Longitudinal control τ_θ in the presence of manual perturbations.

Acknowledgement

S.S. and R.L. are partially supported by CONACYT. This work was carried out and funded in the framework of the Labex MS2T. It was supported by the French Government, through the program Investments for the future managed by the National Agency for Research (Reference ANR-11-IDEX-0004-02)

References

1. J. Borenstein, H. R. Everett, L. Feng and D. Wehe, "Mobile robot positioning-sensors and techniques," *J. Robot. Syst.* **14**(4), 231–249 (2007).
2. H. Al-Helal and J. Sprinkle, "UAV Search: Maximizing Target Acquisition," *17th IEEE International Conference and Workshops on the Engineering of Computer-Based Systems*, Oxford (Mar. 2010) pp. 9–18.
3. M. Quigley, M. A. Goodrich, S. Griffiths, A. Eldredge and R. W. Beard, "Target Acquisition, Localization, and Surveillance Using a Fixed-Wing Mini-UAV and Gimbaled Camera," *Proceedings of ICRA 2005*, Barcelona, Spain (April 18–22, 2005).
4. H.-W. Schulz, M. Buschmann, L. Krüger, S. Winkler and P. Vörsmann, "Towards vision-based autonomous landing for small UAVs: First experimental results of the vision system," *J. Aerosp. Comput. Inf. Commun.* **4**, 785–797 (2007).
5. Z. Yu, K. Nonami, J. Shin and D. Celestino, "3D vision based landing control of a small scale autonomous helicopter," *Int. J. Adv. Robot. Syst.* **4**(1), 51–56 (2007).
6. H. Wu, D. Sun and Z. Zhou, "Micro air vehicle: Configuration, analysis, fabrication, and test," *IEEE/ASME Trans. Mechatronics* **9**, 108–117 (2004).
7. A. Kurdila, M. Nechyba, R. Prazenica, W. Dahmen, P. Binev, R. DeVore and R. Sharpley, "Vision-Based Control of Micro–Air–Vehicles: Progress and Problems in Estimation," *43rd IEEE Conference on Decision and Control*, Atlantis Paradise Island, Bahamas (2004) pp. 1635–1642.
8. C. Kemp, "Visual Control of a Miniature Quad-Rotor Helicopter," in *Churchill College.* Ph.D. Thesis (University of Cambridge, 2006).
9. J. Conroy, G. Gremillion, B. Ranganathan and J. S. Humbert, "Implementation of widefield integration of optic flow for autonomous quad-rotor navigation," *Auton. Robot.* **2**7, 189–198 (2009) .
10. Z. He, R.V. Iyer and P. R. Chandler, "Vision-Based UAV Flight Control and Obstacle Avoidance," *American Control Conference*, Minneapolis, MN (June 2006) pp. 2166–2170.
11. I. F. Mondragon, P. Campoy, J. F. Correo and L. Mejias, "Visual Model Feature Tracking For UAV Control," **In:** *IEEE International Symposium on Intelligent Signal Processing, 2007 (WISP 2007)*, Alcala de Henares, Spain (Oct. 2007) pp. 1–6.
12. V. Duchaine, S. Bouchard and M. Gosselin, "Computationally efficient predictive robot control," *IEEE/ASME Trans. Mechatronics* **12**(5), 570–578 (2007).
13. F. Caballero, L. Merino, J. Ferruz and A. Ollero, "Unmanned aerial vehicle localization based on monocular vision and online mosaicking," *Intell. Robot. Syst.* **55** 323–343 (2009).
14. E. Altug, J. P. Ostrowski and C. J. Taylor, "Control of a quad-rotor helicopter using dual camera visual feedback," *Int. J. Robot. Res.* **24**, 329–341 (2005).
15. P. Castillo, A. Dzul and R. Lozano, "Realtime stabilization and tracking of a four-rotor mini rotorcraft," *IEEE Trans. Control Syst. Technol.* **12**, 510–516 (2004).
16. P. Castillo, R. Lozano and A. Dzul, "Stabilization of a mini rotorcraft with four rotors: Experimental implementation of linear and nonlinear control laws," *IEEE Control Syst. Mag.* **25**, 45–55 (2005).

https://doi.org/10.1017/S0263574713000611 Published online by Cambridge University Press

17. S. Salazar-Cruz, R. Lozano and J. Escareno, "Stabilization and nonlinear control for a novel trirotor mini-aircraft," *Control Eng. Pract.* **17**, 886–894 (2009).
18. H. Romero, S. Salazar and R. Lozano, "Real-time stabilization of an eight-rotor UAV using optical flow," *IEEE Trans. Robot.* **25**, 809–817 (2009).
19. B. L. Stevens and F. L. Lewis, *Aircraft Control and Simulation* (John Wiley and Sons, New Jersey, USA, 2003).
20. A. R. Teel, "Global stabilization and restricted tracking for multiple integrators with bounded controls," *Syst. Control Lett.* **18**, 165–171 (1992).
21. D. Stein, E. R. Scheinerman and G. S. Chirikjian, "Mathematical models of binary spherical-motion encoders," *IEEE/ASME Trans. Mechatronics* **8**(2), 234–244 (2003).
22. W. E. Green, P. Y. Oh and G. L. Barrows, "Flying Insect Inspired Vision for Autonomous Aerial Robot Maneuvers in Near-Earth Environments," *Proceedings of IEEE International Conference on Robotics and Automation*, New Orleans, LA (2004) pp. 2347–2352.
23. L. Zhang, A. Dehghani, S. Zhenwei, T. King, B. Greenwood and M. Levesley, "Development of a mechatronic sorting system for removing contaminants from wool," *IEEE/ASME Trans. Mechatronics* **10**(3), 297–304 (2005).
24. O. Faugeras, "Stratification of three-dimensional vision: Projective, affine, and metric representations," *J. Opt. Soc. Am.* **12**(3), 465–484 (1995).
25. L. Bar, N. Sochen and N. Kiryati, "Variational Pairing of Image Segmentation and Blind Restoration," *Proceedings of the ECCV*, Prague, Czech Republic, Part II: LNCS 3022 (2004) pp. 166–177.
26. J. Canny, "A computational approach to edge detection," *IEEE Trans. Pattern Anal. Mach. Intell.* **PAMI-8**(6), 679–698 (1986).
27. L. Murateta, S. Doncieuxa, Y. Briereb and J. Meyera, "A contribution to vision-based autonomous helicopter flight in urban environments," *Robot. Auton. Syst.* **50**, 195–209 (2005).
28. J. Stowers, A. Bainbridge-Smith, M. Hayes and S. Mills, "Optical flow for heading estimation of a quad-rotor helicopter," *Int. J. Micro Air Vehicles* **1**(4), 229–239 (2009).
29. S. S. Beauchemin and J. L. Barron, "The computation of optical flow," *ACM Comput. Surv.* **27**, 433–467 (1995).
30. J. Y. Bouguet, "Pyramidal implementation of the Lucas–Kanade feature tracker," *Intel Corp. Microprocess. Res. Lab*., 1–9 (1999).
31. J. Peuteman, D. Aeyels and R. Sepulchre. "Boundedness properties for time-varying nonlinear systems," *SIAM J. Control Optim.* **39**(5), 1408–1422 (2000).
32. R. Hartley and A. Zisserman, *Multiple View Geometry*, 2nd ed. (Cambridge University Press, 2003), New York, NY, USA. ISBN: 0521540518.
33. Rabbit Semiconductors, Dynamics C user manual 2007, http://www.rabbitsemiconductor.com/.
34. X4 Hover with position control. http://www.youtube.com/watch?v=3DYTncI19x0
35. X4 and moving target. http://www.youtube.com/watch?v=LkWP6-2_Esk&feature=related

https://doi.org/10.1017/S0263574713000611 Published online by Cambridge University Press