J Intell Robot Syst (2013) 70:65–78
DOI 10.1007/s10846-012-9747-9

Tracking a Ground Moving Target with a Quadrotor Using Switching Control

Nonlinear Modeling and Control

J. E. Gomez-Balderas · G. Flores · L. R. García Carrillo · R. Lozano

Received: 30 June 2012 / Accepted: 16 July 2012 / Published online: 12 August 2012

Abstract An unmanned aerial vehicle (UAV) stabilization strategy based on computer vision and switching controllers is proposed. The main goal of this system is to perform tracking of a moving target on ground. The architecture implemented consists of a quadrotor equipped with an embedded camera which provides real-time video to a computer vision algorithm where images are processed. A vision-based estimator is proposed, which makes use of 2-dimensional images to compute the relative 3-dimensional position and translational velocity of the UAV with respect to the target. The proposed estimator provides the required states measurements to a microcontroller for stabilizing the vehicle during flight. The control strategy consists of switching controllers, which allows making decisions when the target is lost temporarily or when it is out of the camera's field of view. Real time experiments are presented to demonstrate the performance of the target-tracking system proposed.

Keywords Switching controllers · Tracking · Control strategy

J. E. Gomez-Balderas (✉) · G. Flores · R. Lozano
Heudiasyc UMR 7253,
Université de Technologie de Compiegne,
Centre de Recherche de Royalieu,
60200 Compiegne, France
e-mail: jgomezba@hds.utc.fr

G. Flores
e-mail: gfloresc@hds.utc.fr

R. Lozano
e-mail: rlozano@hds.utc.fr

L. R. Garcia Carillo
Department of Electrical and Computer Engineering,
University of California, Santa Barbara,
CA 93106-9560, USA
e-mail: lrgc@ece.ucsb.edu

1 Introduction

Tracking moving targets (m.t.) using unmanned aerial vehicles (UAV) allows performing important tasks like surveillance, reconnaissance, and intelligence missions. UAV hovering over a desired position requires information coming from a group of sensors; more precisely, it requires data coming from the UAV attitude as well as of its surrounding environment. A sensor capable of obtaining abundant information from the UAV environment is a vision system. Many results related to this topic have been presented recenttly. Most of them are related to identification and classification of multiple targets, see for example [1, 9, 15, 16, 25] and [22]. A circular pattern navigation algorithm for autonomous target tracking has been studied in [24] and [30], showing a good performance. Other work concerning trajectory generation from video sensors includes particle

filters for moving cameras without stabilization [20], and the Joint Probabilistic Data Association Filter (JPDAF) for tracking multiple targets with unreliable target identification [3]. If a model for the object's motion is known, an observer can be used to estimate the object's velocity [10]. In [13], an observer for estimating the object velocity was utilized; however, a description of the object's kinematics must be known. In [18] an autoregressive discrete time model is used to predict the location of features of a moving object. In [2], trajectory filtering and prediction techniques are utilized to track a moving object. In [26], object-centered models are utilized to estimate the translation and the center of rotation of the object. Several interesting works have been presented concerning visual tracking of targets using UAVs. In [29] a color-based tracker is proposed to estimate the target position, while in [14] thermal images are correlated with a geographical information system (GIS) towards the same goal. In [12] a vision-based control algorithm for stabilizing a UAV equipped with two cameras is presented. The same system has been used in [11] to track a line painted in a wall using a vanishing points technique. Some methods for designing UAV trajectories that increase the amount of information available are presented in [23].

In this paper we describe the use of a vision system designed to observe a visual target located over a ground vehicle and to track it by using an UAV platform of quadrotor type (X-4). The main challenge involved in target localization include maintaining the target inside the camera's field of view. In order to achieve this requirement, we propose a control schema that develops a X-4 tracking, such that the target localization estimation error is minimized. To successfully perform this task, we have developed a control strategy consisting of three principals objectives. First, the hovering UAV tracks the target over a desired position, this stage is known as take-off mode (TO). In case of a temporary lose of sight of the target, we use a second control schema named target localization (TL), which consists on increasing the UAV altitude to obtain a better view of the scene. Once the vision system has located the target, our third control schema called loss of moving target (LOMT) moves the UAV until it reaches a desired position with respect to the target. All this control strategies switch at every time during the test. The input of the image processing algorithm is a set of images taken during real flight tests, the output of our algorithm is the 3-dimensional target position and its translational velocity. However, one missing part, as far as our interest is concerned, is that there are no physical obstacles which hinder UAVs from tracking a target. This paper focuses on the UAV implementation strategies of tracking a target over a ground vehicle using vision.

This paper is organized as follows. Computer vision algorithm for position measurement and velocity estimation using optical flow are described in Section 2. In Section 3, the control design is presented. The experimental platform used to validate theoretical results is described in Section 4. Experimental results are presented in Section 5. Finally, Section 6 presents conclusions and future work.

2 Image Processing

The quadrotor's computer needs to be able to chase the on ground m.t. using the information given by the camera, i.e., position and relative velocity. Regarding this, it is required the estimation of the relative $x - y$ position, altitude and velocity to the moving target with respect to the quadrotor. An on board camera is implemented for such goal and the schema describing such system is shown in Fig. 1. At this respect, we use some processing methods: edge detection, target detection and optical flow measurement which will be further discussed in this section.

2.1 Edge Detection

In the present approach a red square object must be tracked using a camera. For this reason, an edge detector to find the edges of the target in the image plane is proposed. We use a Canny edge detection algorithm [7] which is known to have the following characteristics: good detection, localization, and only one response to a single edge. The Canny edge detection algorithm uses the good detection criterion: there should be a low probability

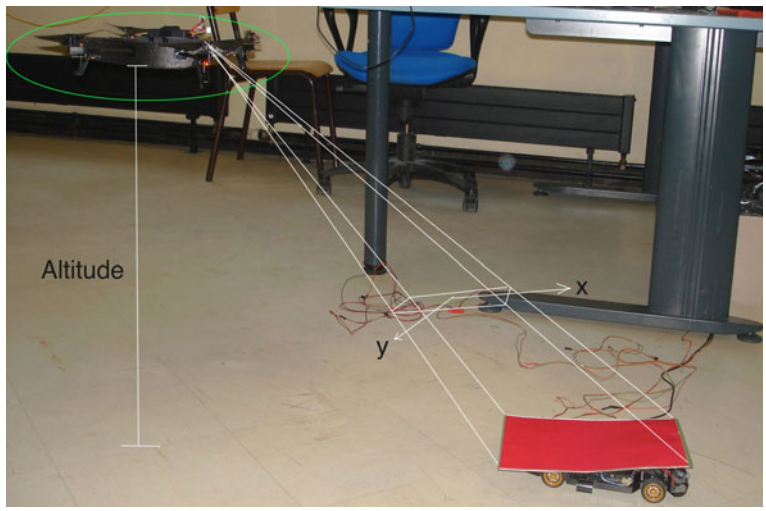


Fig. 1 Basic scheme

of failing to mark real edge points, and low probability of falsely marking non edge points. This criterion corresponds to maximize signal-to-noise ratio. In the good localization criterion the points marked as edge points by the operator should be as close as possible to the center of the true edge. That corresponds to maximize the position interval, passing by zero and correspond to the inverse of the expected value between the true edge point and the maximum output of the operator. The last criterion to only obtain one response to a single edge is implicitly captured in the first criterion, since when there are two responses to the same edge, one of them must be considered false. This is necessary to limit the number of peaks in the response so that there will be a low probability of declaring more than one edge.

Our vision-based position estimation algorithm uses a red colored square target, for this reason we propose to use an edge detector to find the edges of the square target in the image plane. The Canny edge detection algorithm, locates the edges of the target in the image and 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, then, gradient magnitude and direction are computed at each pixel of the smoothed image. Gradient direction is used to thin 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. A good operation to use with any edge operator when thin boundaries are wanted. The two 8-neighbors of a pixel $[x, y]$ that are to be compared are found by rounding off the computed gradient direction to yield one neighbor on each side of the center pixel. 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 meet a lower threshold, usually about half of the higher starting threshold. Image regions can sometimes be detected when boundary segments close on themselves. In our case the boundary of a rectangular building might result in four straight line segments.

2.2 Target Detection

Once we have four rays which intersect in four points it is required to verify that the rays form a square. To verify this, the angle in each corner of the intersection of vertices are computed. By applying the triangulation of polygons, we can obtain two similar triangles in one square, given the triangle Δ with edges A, B and C, the angle β

Fig. 2 Target detection

between the two edges B and C can be obtained by the following formula:

$$\beta = \arccos \frac{|B|^2 + |C|^2 - |A|^2}{2 \cdot |B| \cdot |C|} \tag{1}$$

where $|A|, |B|$ and $|C|$, denotes the lengths of the edges A, B, and C respectively. Applying this formula at each one of the square vertices, we obtain the four angles in the polygon if this edges are perpendiculars. Figure 2 shows the result of target detection algorithm using four vertices; the 4 edges are painted in blue color.

2.3 Optical Flow Measurement

We use the optical flow obtained from image sequences to estimate the translational speed of the X-4. The horizontal speed estimation is used to perform autonomous hover flights as well as for avoiding lateral displacement. Many different methods for computing the optical flow are available [4]. We can mention for example intensity-based differential methods, frequency-based filtering methods or correlation-based methods. In this paper we implement the Lucas–Kanade pyramidal method, which is an intensity-based differential method [6].

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

$$G_{p_1} = I_1(u_1, v_1) \quad G_{p_2} = I_2(u_2, v_2) \tag{2}$$

where u_i and v_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

$$p_2 = p_1 + r = [\, u_1 + r_u \;\; v_1 + r_v \,]^T \tag{3}$$

where $r = [\, r_u \; r_v \,]^T$ defines the image displacement or optical flow and minimizes the following residual function

$$\varepsilon(r) = \sum_{u_p = u_{p_1} - w_u}^{u_{p_1} + w_u} \sum_{v_p = v_{p_2} - w_v}^{v_{p_2} + w_v} (I_1(p_1) - I_2(p_1 + r))^2 \tag{4}$$

where w_u and w_v are two integers that define the size of the integration window. The Lucas–Kanade optical flow algorithm has an adaptive integration window, therefore it is capable of handling large pixel motions and acts as a low pass filter. For a complete description of the algorithm see [6].

Optical flow can be generated by two kinds of observer motion: translational motion (F_t) and rotational motion (F_r). 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. 1. The optical flow (OF) can be mathematically expressed as follows:

$$OF = \frac{v}{d} \sin\beta - \omega \tag{5}$$

The maximal optical flow is obviously generated when the plane that contains the features is perpendicular to translational motion direction ($\beta = 90°$) [4]. The velocity can be estimated from Eq. 5 as follows

$$v = \frac{(OF + \omega)}{\sin\beta} d \tag{6}$$

Notice that singularities in the above equation appear when $\beta = 0°$. Nevertheless, in our case the roll and pitch angles are very close to zero which implies that $\beta = 90° \pm 3°$ as a maximal interval when the quadrotor is appropriately stabilized at hover. The measurement of the angular speed ω is obtained by using the gyro information on each axis. An altitude stabilization algorithm in closed loop is used to keep the distance d constant and equal to some desired value.

2.4 Solving the Problem

Translational velocity was obtained using the optic flow algorithm described in the last section. To obtain X-4 position, we use a target detection with a red color segmentation algorithm to obtain only the red channel image. After this, we apply the Canny algorithm for edge detection, attaining the interior and exterior boundaries in this region. The exterior boundaries are formed by a finite collection of n line segments of the following way $e_0 = v_0v_1, e_1 = v_1v_2, ..., e_i = v_iv_{i+1}, ..., e_{n-1} = v_{n-1}v_0$, connected by n image points $v_0, v_1, v_2, ..., v_{n-1}$ where $v_i = (x_i, y_i)$. Using this points, we can obtain four vertices with four edges, the next step is to verify the right angle in each corner of the quadrilateral, using Eq. 1. With the area of the square founded and using the pinhole camera model with the intrinsic parameters values, we use similar triangles properties to obtain the distance Z_W (altitude) between the camera and the target. Also, other information is obtained such as the central point of the square, that is equal to the point where the two diagonals of the square intersect, called (x_{ci}, y_{ci}) in the image plane. Using this information we can obtain the 3D coordinates of this point:

$$X_W = f\frac{x_{ci}}{Z_W} \tag{7}$$

$$Y_W = f\frac{y_{ci}}{Z_W} \tag{8}$$

where f is the focal length of the camera. Knowing the four coordinates of target v_0, v_1, v_2, and v_3 vertex of the target and the center image coordinates (x_{ci}, y_{ci}), we construct a line l_1 going from the vertex v_0 to the central point $p_{ci} = (x_{ci}, y_{ci})$, $l_1 = v_0 + \alpha_1(p_i)$. In case of horizontal displacement of the target we need to know the target direction in order to indicate to the UAV a change in yaw angle. This is achieved by using a reference coordinate located in the image corner $p_0 = (0, 0)$. We construct a line from p_0 to p_{ci}, $l_2 = p_0 + \alpha_2(p_{ci})$; using l_1 and l_2 we obtained the yaw angle reference of the UAV using dot product equation:

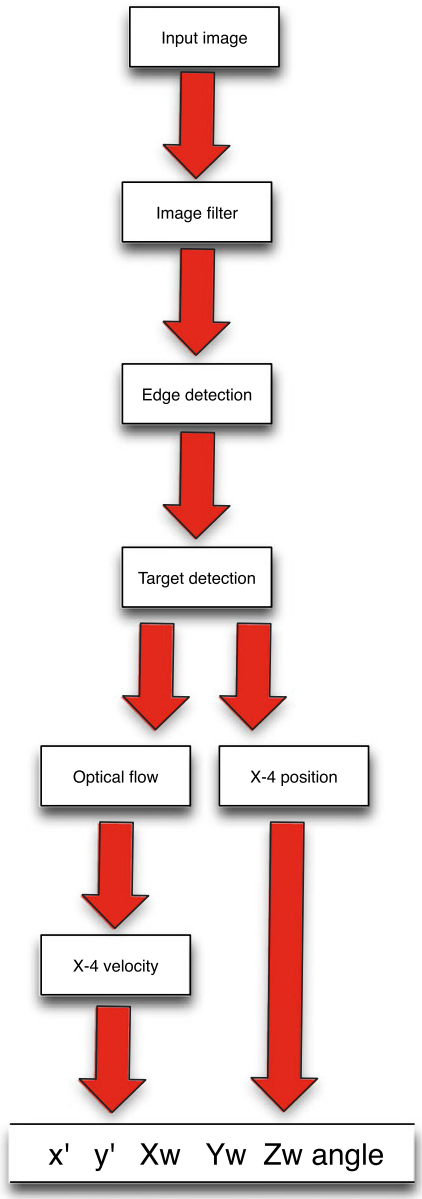


Fig. 3 Flowchart of image processing

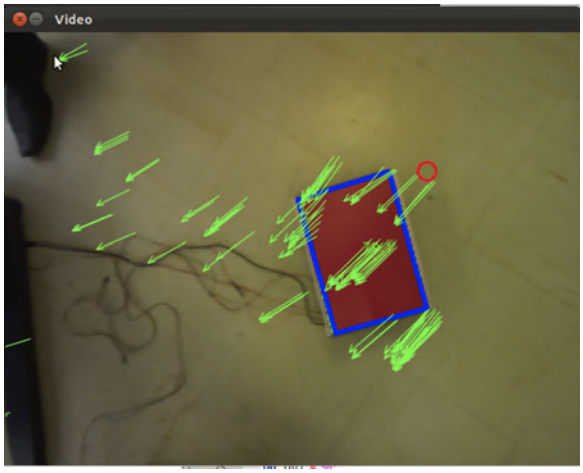


Fig. 4 X-4 real time image

$\psi = \arccos(l_1 \cdot l_2)$. To calculate X-4 velocity estimation we use a Lucas–Kanade based optical flow algorithm to obtain the translational velocity of the camera placed under X-4 structure. It should be noticed that optical flow gives the velocity in the image plane, which in this case corresponds to the velocity of the X-4 displacements when using the floor as reference.

The proposed algorithm obtain (X_W, Y_W, Z_W, ψ) that is, the 3-dimensional position orientation of the target with respect to the X-4 camera. Using the optical flow algorithm we acquire $(\dot{x}, \dot{y})$, which is the X-4 velocity estimation corresponding respectively to $(\dot{x_{ci}}, \dot{y_{ci}})$ in world coordinates. The

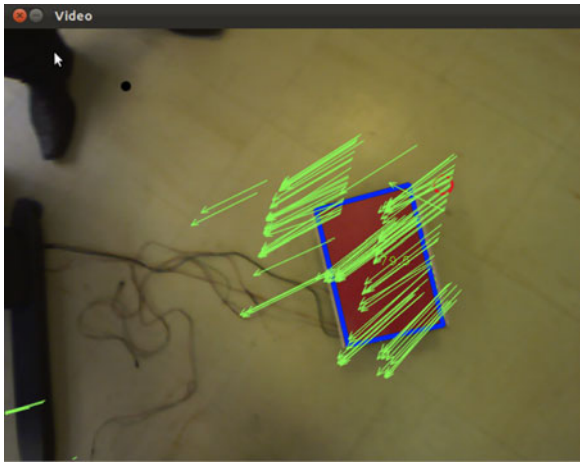


Fig. 5 X-4 real time displacement

flowchart in Fig. 3 explains the image processing algorithms used to obtain position, orientation and velocity data. Once we obtain this 6-tuple, this information is sent to the X-4 by means of a wireless modem with a baud rate equal to 38400. Our X-4 vehicle is equipped with a reception modem connected to the serial port of a Rabbit microprocessor, embedded in the X-4. The output of these methods during real flight tests are shown in Figs. 4 and 5.

3 Control

A common mathematical model for the quad-rotor is implemented aiming at developing the navigation control strategy [8]:

$$\begin{aligned}
\ddot{x} &= -u_1(\cos\phi \sin\theta \cos\psi + \sin\phi \sin\psi) \\
\ddot{y} &= -u_1(\cos\phi \sin\theta \sin\psi - \sin\phi \cos\psi) \\
\ddot{z} &= 1 - u_1(\cos\phi \cos\theta) \\
\ddot{\theta} &= u_2 \\
\ddot{\phi} &= u_3 \\
\ddot{\psi} &= u_4
\end{aligned} \tag{9}$$

The quad-rotor model, presented in Eq. 9, can be written in a state-space form by introducing $\dot{X} = (x_1, ..., x_6)^T \in \Re^6$, $\dot{Z} = (z_1, ..., z_6)^T \in \Re^6$, with states defined by

$$\begin{array}{l|l}
x_1 = x & z_1 = \theta \\
x_2 = \dot{x} & z_2 = \dot{\theta} \\
x_3 = y & z_3 = \phi \\
x_4 = \dot{y} & z_4 = \dot{\phi} \\
x_5 = z & z_5 = \psi \\
x_6 = \dot{z} & z_6 = \dot{\psi}
\end{array} \tag{10}$$

Using the linear representation of the model in Eq. 9 and the notation from Eq. 10, one has

$$\begin{array}{l|l}
\dot{x_1} = x_2 & \dot{z_1} = z_2 \\
\dot{x_2} = -z_1 u_1 & \dot{z_2} = u_2 \\
\dot{x_3} = x_4 & \dot{z_3} = z_4 \\
\dot{x_4} = z_3 u_1 & \dot{z_4} = u_3 \\
\dot{x_5} = x_6 & \dot{z_5} = z_6 \\
\dot{x_6} = 1 - u_1 & \dot{z_6} = u_4
\end{array} \tag{11}$$

Due to the fact that, in general, the vehicle never works in areas where $|\theta| \geq \pi/2$ and $|\phi| \geq \pi/2$,

the linear model is chosen for being implemented instead of its nonlinear version. Such working domains are satisfied even in research works where the nonlinear model is used together with a feedback control [5].

3.1 Operating Modes of the Moving Target Mission

The navigation mission has been divided into different sub-missions or stages:

- **Take-off (TO)** the objective is to achieve the desired altitude z_d and it will be supposed that the moving target is initially inside of the camera's field of view.
- **Target localization (TL)** in this mode, the vehicle has achieved the desired altitude. The task to be accomplished here is to align the vehicle's center of gravity (CG) w.r.t the moving target.
- **Loss of moving target (LOMT)** the vehicle is required to change its altitude in order to find the moving target, i.e. until the moving target is inside of the camera's field of view. Once the mini-UAV has found the moving target, the vehicle should go back to the desired altitude and pursue the moving target.

The navigation control is structured in different controllers for each sub-mission as illustrated in Fig. 6.

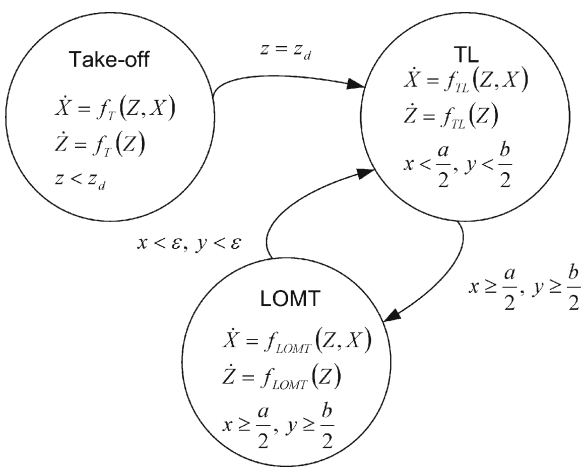


Fig. 6 Navigation control system

3.2 Control Laws in Each Operating Mode

The control strategy proposed in all different modes is based on the idea that the global system, presented in Eq. 11, is constituted of two subsystems, the attitude dynamics and the position dynamics, existing a time-scale separation between them [19]. From this fact, it is possible to propose a hierarchical control scheme, where the positioning controller provides the reference attitude angles (θ_d, ϕ_d and ψ_d), which are the angles that must be tracked by the orientation controllers. For the complete system analysis, the error dynamics of the model in Eq. 11 are represented by the errors $\tilde{x}_i = x_i - x_{i_d}$ and $\tilde{z}_i = z_i - z_{i_d}$, with $i \in \{1, ..., 6\}$.

3.2.1 Attitude Control

For the present studies, attitude dynamics have the same controller among all operating modes. An integral sliding mode control (ISMC) is proposed and implemented on the platform, which is explained next.

For the pitch dynamics case, the error equation is defined as $\tilde{z}_1 = z_1 - z_{1_d}$. As shown in [28], lets select the switching function

$$s(z, t) = \dot{\tilde{z}}_1 + 2\lambda\tilde{z}_1 + \lambda^2 \int_0^t \tilde{z}_1(\tau)\, \mathrm{d}\tau \tag{12}$$

which depends on the pitch dynamics states. The λ parameter in Eq. 12 is the slope of the sliding line and should be greater than zero to ensure the asymptotic stability of the sliding mode. Computing the time derivative of Eq. 12 one has

$$\dot{s} = z_1 u_1 + 2\lambda z_2 + \lambda^2 \tilde{z}_1 \tag{13}$$

Considering the sliding mode condition $\dot{s} = 0$, and using Eq. 13 one finds the equivalent control

$$z_{1_{eq}} = -2\lambda z_2 - \lambda^2 \tilde{z}_1 \tag{14}$$

With the purpose of obtaining a control law such that the state vector $\tilde{z}_1$ remains on the sliding surface $s(z, t) = 0$, $\forall t > 0$, let's use the Lyapunov function candidate

$$v(s) = \frac{1}{2}s^2 \tag{15}$$

An efficient condition for the stability of the pitch sub-system can be satisfied if one can ensure that the reached condition

$$\dot{v}(s) = \frac{1}{2}\frac{d}{dt}s^2 \leq \eta|s|, \qquad \eta \geq 0 \tag{16}$$

holds. Then, the system remains on the sliding surface and the states go to the origin. Thus $s\dot{s} \leq -\eta|s|$ and the controller must be chosen such that

$$z_1 = z_{1_{eq}} - Ksign(s) \tag{17}$$

where K is a positive real number. Following a similar approach, it is possible to obtain the yaw and roll angles controllers.

3.3 Position Control

For each sub-mission or stage, the position control have well defined objectives, previously explained. Motion in the $x-y$ plane is accomplished by orientating the vehicle's thrust vector in the direction of the displacement desired. As consequence, the angles θ_d and ϕ_d act as virtual controllers for the position dynamics. The control laws proposed for the x an y positions, respectively, are expressed as

$$\theta_d = \frac{k_{vx}(x_2 - x_{2_d}) + k_{px}(x_1 - x_{1_d})}{u_1} \tag{18}$$

$$\phi_d = -\frac{k_{vy}(x_4 - x_{4_d}) + k_{py}(x_3 - x_{3_d})}{u_1} \tag{19}$$

with k_{vx}, k_{px}, k_{vy} and k_{py} being positive real numbers.

3.3.1 Altitude Control

The z-position control is composed by two different controllers, one for the situation when the m.t. is being detected and one for the situation when it is not. When the m.t. is inside of the camera's field of view, the proposed control law is formed by a state feedback taking into account only the states involving the altitude dynamics. On the other hand, when the vehicle looses the image of the target, a switch to a different method for measuring the vehicle's ψ angle occurs, and, at the same time, the z controller changes. In both of them, the control objective is to regulate the x and y states to the origin, i.e. $x_{1d} = x_{3d} = 0$.

There are two possible situations where the m.t. is lost by the mini-UAV. The first one occurs when the camera's field of view is disturbed by some objects between the m.t. and the UAV. The second one occurs when the m.t. presents one acceleration large enough to disallow the tracking of the target. In this paper we focus only in the second case, considering the camera's field of view is never obstructed by some objects. Thus, we will focus in the second case where is impossible to track the m.t. due to lack of information in the visual system. Then we propose a searching process consisting on changing the altitude until the m.t. is found again.

– Control schema when the m.t. is detected: in this case, the feedback control law proposed is given by

$$u_{1_{mt}} = k_{pz}\left(x_5 - x_{5_d}\right) + k_{vz}\left(x_6 - x_{6_d}\right) - 1 \tag{20}$$

where k_{pz} and k_{vz} are positive real numbers. Then, using the controllers 19 and 20, the closed-loop system of the position dynamics (left hand side of Eq. 11)is given by

$$\begin{aligned}
\dot{x}_1 &= x_2 \\
\dot{x}_2 &= -k_{px}x_1 - k_{vx}x_2 \\
\dot{x}_3 &= x_4 \\
\dot{x}_4 &= -k_{py}x_3 - k_{vy}x_4 \\
\dot{x}_5 &= x_6 \\
\dot{x}_6 &= -k_{pz}x_5 - k_{vz}x_6
\end{aligned} \tag{21}$$

The Eq. 21 can be represented as $\dot{X} = A_{mt}X$ where

$$A_{mt} = \begin{pmatrix}
0 & 1 & 0 & 0 & 0 & 0 \\
-k_{px} & -k_{vx} & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & -k_{py} & -k_{vy} & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & -k_{pz} & -k_{vz}
\end{pmatrix} \tag{22}$$

– Control schema when the m.t. is not detected: in this case, the proposed control schema is given by

$$u_{1_{\overline{mt}}} = k_{pz}(x_5 - x_{5_d}) + k_{vz}(x_6 - x_{6_d}) - k_{Zx}x_1 - k_{Zy}x_3 - 1 \quad (23)$$

where k_{Zx} and k_{Zy} are positive real numbers. Then, the closed-loop system is represented as $\dot{X} = A_{\overline{mt}}X$ where $A_{\overline{mt}} =$

$$\begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ -k_{px} & -k_{vx} & -k_{Zx} & 0 & -k_{Zy} & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & -k_{py} & -k_{vy} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & -k_{pz} & -k_{vz} \end{pmatrix} \quad (24)$$

3.4 Stability Analysis of the Position Sub-System

This section focuses on the system's stability across switching boundaries, i.e., where altitude dynamics switches between the pair of controllers Eqs. 20 and 23.

Following a similar approach than the one presented in [21], it is possible to find a common Lyapunov function for the closed-loop system formed applying the two controllers of the position dynamics. However, working in this way, the same pole locations have to be chosen for both cases, when the m.t. is detected and were it is not detected, which in fact is not the case.

Let's define d_x and d_y as the distances measured from the vehicle's center of gravity projection to the point where the camera loses the image of the road, see Fig. 7. Thus, when the error is bigger than d_x for x dynamics or d_y for y dynamics, the control law is changed. A change of coordinates can be made using the known constants d_x and d_y. Thus, without loss of generality, a state dependent switched linear system can be defined, given by the closed-loop system together with the switching conditions

$$\dot{X} = \begin{cases} A_{mt}X & \text{if } |e_x| < d_x \text{ and } |e_y| < d_y \\ A_{\overline{mt}}X & \text{if } |e_x| \geq d_x \text{ or } |e_y| \geq d_y \end{cases} \quad (25)$$

It is clear that each individual system in Eq. 25 is stable, since the matrices A_{mt} and $A_{\overline{mt}}$ are Hurwitz.

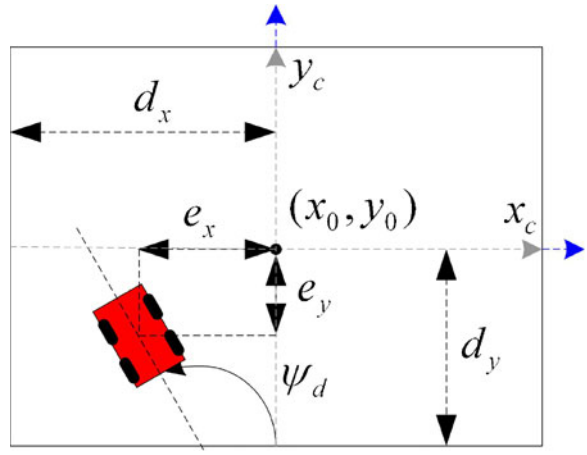


Fig. 7 Camera's view

Suppose that there is a family A_p, $p \in \mathscr{P}$ of functions from $\Re^n$ to $\Re^n$, with $\mathscr{P} = 1, 2, ..., m$ defining the finite index set. For the case of linear systems, this results in a family of systems $\dot{x} = A_p x$ with $A_p \in \Re^{n\times n}$. Let's define a piecewise constant function $\sigma : [0, \infty) \to \mathscr{P}$ with finite number of discontinuities (switching times) on every bounded time interval. This function takes a constant value on every interval between two consecutive switching times. Then σ gives the index $\sigma(t) \in \mathscr{P}$ of the system that is actually active, at each instant of time t.

Theorem 1 *Consider vectors t_{pq}, symmetric matrices S_p with $\Omega_p \in \{x : x^T S_p x \geq 0\}$, $\forall p \in \mathscr{P}$ having non-negative entries and symmetric matrices P_p such that:*

$$A_p^T P_p + P_p A_p + \beta_p S_p < 0, \qquad \beta_p \geq 0 \quad (26)$$

$$0 < P_p - P_q + f_{pq}t_{pq}^T + t_{pq}f_{pq}^T \text{ for some } t_{pq} \in \Re^n \quad (27)$$

With the boundary between Ω_p and Ω_q of the form $\{x : f_{pq}^T = 0\}$, $f_{pq} \in \Re^n$. Then every continuous, piecewise $\mathscr{C}^1$ trajectory of the system $\dot{x} = A_\sigma x$ tends to zero exponentially.

See the Appendix.

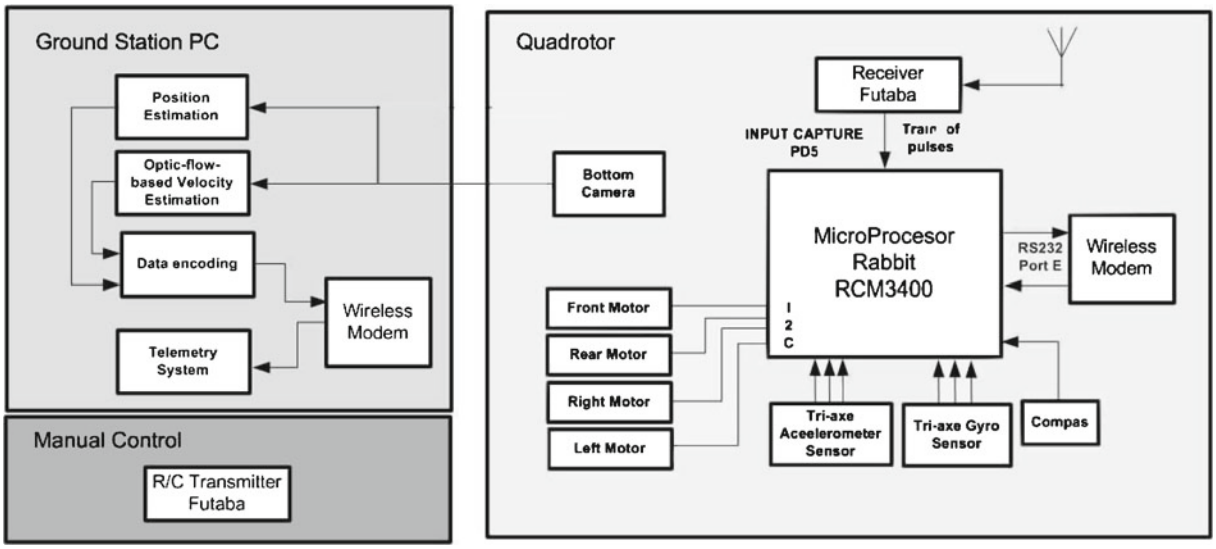
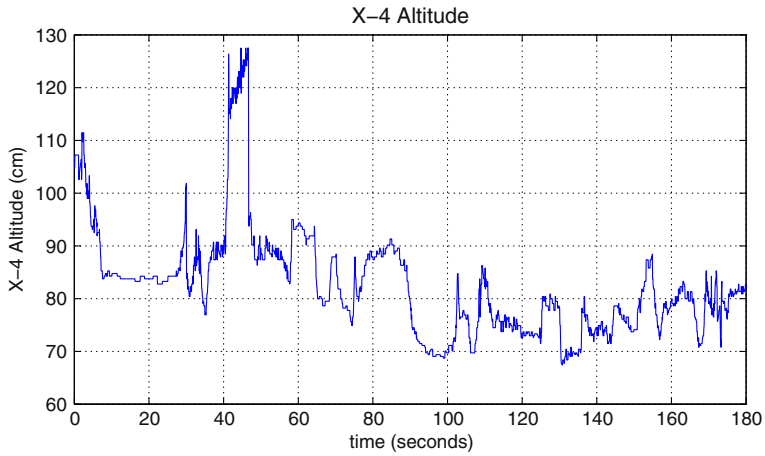


Fig. 8 Embedded control architecture

4 Experimental Platform

The total weight of the vehicle is about 500 gr, with a flight endurance of 10 min approximately. Theoretical results obtained were incorporated into an autopilot control system using an architecture based on a 29 MHz Rabbit micro controller with 512 Kb Flash and 512 Kb RAM. These micro controllers are capable of handling floating point operations and multitasking processing virtually due to the enhancement compiler Dynamic C [27]. We have built our own inertial measurement unit (IMU) using accelerometers, gyros and a compass to obtain the roll, pitch, and yaw angles, as well as the angular rates. The IMU information is sent to the micro controller which also reads reference control inputs from an operator through a serial wireless modem. The micro controller subsequently combines this information to compute the control input and sends the control corrections to the motors through a I2C serial port. The vision sensor is composed of a wifi camera placed onboard the X-4 platform, the camera is pointing downwards, allowing to perform the optical

Fig. 9 X-4 altitude



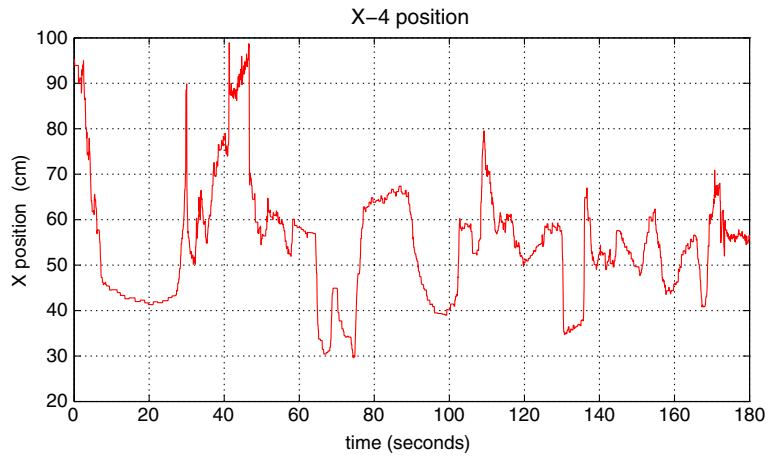


Fig. 10 X-4 position over X-axis

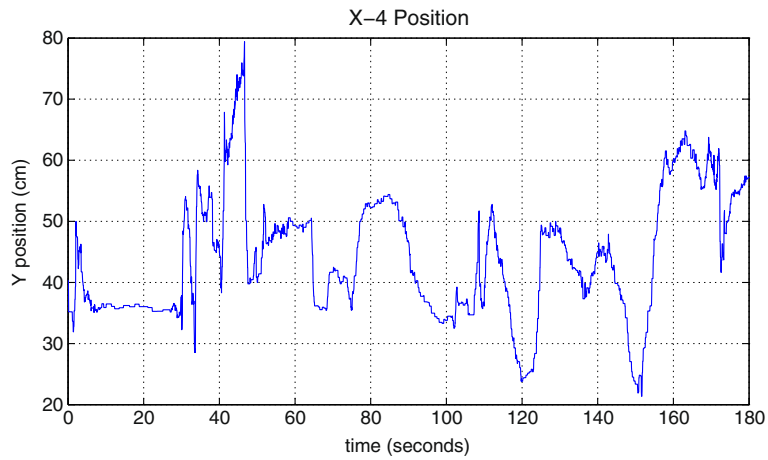


Fig. 11 X-4 position over Y-axis

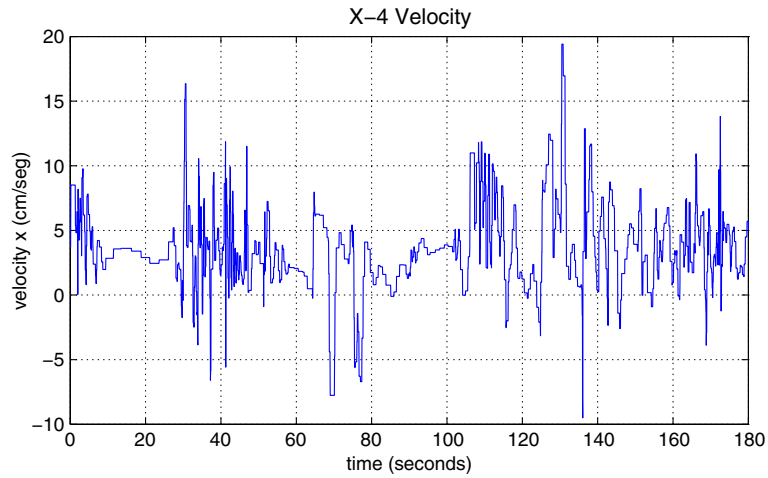


Fig. 12 X-4 velocity over X-axis

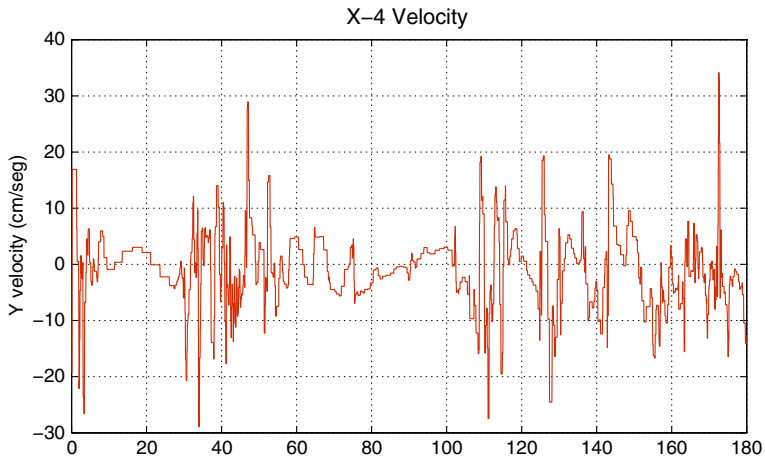


Fig. 13 X-4 velocity over Y-axis

flow and positioning algorithms. The algorithm processes images coming the camera at a rate of 18 frames/seconds. The image has a size of 320 width pixels and 240 height pixels. The 3D position and displacement velocity estimation are computed in a PC with Intel Core 2 duo processor 2.10 GHz using OpenCV libraries. Figure 8 shows a block diagram of the basic architecture.

5 Real Time Experiments

To test the proposed algorithms we realize several experiments. In this article we describe an experience in real flight consisting of 180 s of tracking a car with a translation motion at constant velocity. X-4 position and velocity estimation are obtained by means of the image processing algorithm. In the first 10 s we observe the initialization of the algorithms, at this time the target detection algorithms identifies the target in the field of view of the camera. When the target is detected we obtain the position of the camera, next, the X-4 tries to go over the target. After the initialization the control algorithm uses the take-off (TO) navigation method with the desired altitude $z_d = 90$ cm. At 55 s the target move faster than X-4 and the camera lost the target. With this information the X-4 navigation method change to loss of moving target (LOMT) mode, increasing its altitude over 120 cm. At this altitude the X-4 camera localizes the target, and target localization mode (TL) try to displace the X-4 toward a desired altitude. It is observed in Fig. 9 how the X-4 moves to desired altitude using a embedded switch control. Figure 10 shown the X-4 displacements estimation over its X-axis on 180 s of real flight. It can be seen the initialization steps during the first 10 s of flight. After that, we observe the displacement of the X-4 center of gravity and how the X-4 embedded control tries to move it to a desired orientation close to 60 cm over X-4 axis. In Fig. 11 we shown X-4 displacements estimation over its Y-axis on 180 s of real flight. After 10 s of initialization step, the X-4 embedded control moves it to a desired orientation over Y-axis, in our case 50 cm. Figures 12 and 13 show the X-4 velocity over the X-axis and Y-axis, respectively, on 180 s of flight. Velocity estimation was obtained using optical flow algorithm.

6 Conclusion

In this paper we presented a UAV tracking a target placed over a ground vehicle. The strategy makes use of a vision system that estimates position, orientation and displacement velocity of the UAV with tespect to the moving target. The navigation mission uses different control algorithms: take-off (TO) mode for achieving a desired altitude; target localization (TL) mode has for goal aligning the vehicle's center of gravity with respect to the moving target; and loss of moving target (LOMT) mode where the vehicle is required to

change its altitude in order to find the moving target. The UAV tracking algorithm has been developed and tested using our experimental X-4 platform in real-time flights. The real-time experiments have shown an acceptable performance of the UAV applying the control navigation schema proposed.

Future work will concern extending the strategies presented in this paper, with the purpose of working in different conditions, such as a different target forms. Pattern recognition methods, like machine learning, are being tested with the main objective of providing a more robust estimation of the form and location of the target.

Acknowledgements This work was partially supported by the Mexican National Council for Science and Technology (CONACYT), the Institute for Science & Technology of Mexico City (ICyTDF) and the French National Center for Scientific Research (CNRS).

Appendix

Before proving Theorem 1, let's use the following theorem.

Theorem 2 *The system* $\dot{x} = f(t, x)$, $f(t, x) \equiv 0$, *is exponentially stable on the region* $D = \{x \in \Re^n | \|x\| < r\}$ *if there exists a Lyapunov function* $V(t, x)$ *and some positive constants* c_1, c_2, c_3, *such that* $\forall (t, x) \in [0, \infty) \times D_0$, $D_0 = \{x \in \Re^n | \|x\| < r/m\}$

$$c_1 \|x\|^2 \leq V(t, x) \leq c_2 \|x\|^2 \tag{28}$$

$$\frac{\partial V}{\partial t} + \frac{\partial V}{\partial x} \leq -c_3 \|x\|^2 \tag{29}$$

where m is the overshot from definition of exponential stability.

See [17], pp. 169.

Proof of Theorem 1 The proof relies on the Theorem 2, then using the Lyapunov function candidate $V(x) = x^T P_p x$ and assuming that $x(t)$ is continuous and piecewise $\mathscr{C}^1$, hence, $V(t)$ has the same characteristics. Premultiplying and postmultiplying the condition 27 by x, the inequality on the left side of Eq. 28 is satisfied. In the same way, inequality 29 follows if we premultiply and postmultiply both sides of Eq. 26 by x.

References

1. Aguiar, A.P., Hespanha, J.P.: Minimum-energy state estimation for systems with perspective outputs. IEEE Trans. Automat. Contr. **51**(2), 226–241 (2006)
2. Allen, P.K., Timcenko, A., Yoshimi, B., Michelman, P.: Trajectory filtering and prediction for automated tracking and grasping of a moving object. In: Proc. IEEE Conference on Robotics and Automation, pp. 1850–1856 (1992)
3. Bar-Shalom, Y., Fortmann, T.E.: Tracking and Data Association. Academic Press, Boston, MA (1988)
4. Beauchemin, S.S., Barron, J.L.: The computation of optical flow. ACM Comput. Surv. **27**, 433–467 (1995)
5. Bouabdallah, S., Siegwart, R.: Backstepping andd sliding-mode techniques applied to an indoor micro quadrotor. In: Proceedings of the IEEE International Conference on Robotics and Automation, Barcelona, Spain, pp. 2259–2264 (2005)
6. Bouguet, J.Y.: Pyramidal implementation of the lucas kanade feature tracker description of the algorithm. Tech. Rep., Intel Corporation Microprocessor Research Labs (2000)
7. Canny, J.: A computational approach to edge detection. IEEE Trans. Pattern Anal. Mach. Intell. **PAMI-8**, 679–698 (1986)
8. Carrillo, L.G., Rondon, E., Sanchez, A., Dzul, A., Lozano, R.: Stabilization and trajectory tracking of a quad rotor using vision. J. Intell. Robot. Syst. **61**(1–4), 103–118 (2011)
9. Chitrakaran, V.K., Dawson, D.M., Dixon, W.E., Chen, J.: Identification of a moving objects velocity with a fixed camera. Automatica **41**(3), 553–562 (2005)
10. Ghosh, B.K., Loucks, E.P.: A realization theory for perspective systems with application to parameter estimation problems in machine vision. IEEE Trans. Automat. Contr. **41**, 1706–1722 (1996)
11. Gomez-Balderas, J., Castillo, P., Guerrero, J., Lozano, R.: Vision based tracking for a quadrotor using vanishing points. J. Intell. Robot. Syst. **65**, 361–371 (2012)
12. Gomez-Balderas, J., Salazar, S., Guerrero, J., Lozano, R.: Vision based autonomous hover of a mini-rotorcraft. In: Unmanned Aerial Vehicles Symposium, Dubai (2010)
13. Hashimoto, K., Noritsugu, T.: Observer-based control for visual servoing. In: Proc. 13th IFAC World Congress, San Francisco, California, pp. 453–458 (1996)
14. Heintz, F., Rudol, P., Doherty, P.: From images to traffic behavior - a uav tracking and monitoring

application. In: 10th International Conference on Information Fusion, 9–12 July 2007
15. Hong, L., Cui, N., Pronobis, M.T., Scott, S.: Simultaneous ground moving target tracking and identification using q wavelets features from hrr data. Inf. Sci. **162**, 249–274 (2004)
16. Hwang, I., Roy, K., Balakrishnan, H., Tomlin, C.: A distributed multiple-target identity management algorithm in sensor networks. In: Proceedings of the 43rd IEEE Conference on Decision and Control (2010)
17. Khalil, H.K.: Nonlinear Systems. Prentice Hall, New York (2002)
18. Koivo, A.J., Houshangi, N.: Real-time vision feedback for servoing robotic manipulator with self-tuning controller. IEEE Trans. Syst. Man Cybern. **21**, 134–142 (1991)
19. Kokotovic, P., Khalil, H.K., O'Reilly, J.: Singular Perturbation Methods in Control: Analysis and Design. Academic Press, Siam, London (1999)
20. Lee, J., Huang, R., Vaughn, A., Xiao, X., Hedrick, J.K.: Strategies of path-planning for a uav to track a ground vehicle. In: Proceedings of the 2nd annual Autonomous Intelligent Networks and Systems Conference (2003)
21. Liberzon, D.: Switching in Systems and Control. Birkhuser, Boston (2003)
22. Martinez, S., Bullo, F.: Optimal sensor placement and motion coordination for target tracking. Automatica **42**(4), 661–668 (2006)
23. Ponda, S.S., Kolacinski, R.M., Frazzoli, E.: Trajectory optimization for target localization using small unmanned aerial vehicles. In: AIAA Guidance, Navigation, and Control Conference, Chicago, Illinois, USA, pp. 10–13 (2009)
24. Rafi, F., Khan, S., Shafiq, K., Shah, M.: Autonomous target following by unmanned aerial vehicles. Unmanned systems technology. Conference No. 8 USA **6230**(2) (2006)
25. Schmitt, T., Hanek, R., Beetz, M., Buck, S., Radig, B.: Cooperative probabilistic state estimation for vision-based autonomous mobile robots. IEEE Trans. Robot. Autom. **18**(5), 670–684 (2002)
26. Shariat, H., Price, K.: Motion estimation with more than two frames. In: IEEE Transactions on PAMI, vol. 12, pp. 417–434 (1990)
27. Shell, M.: Rabbit semiconductors, dynamics c user manual. Digi International Inc. (2011). http://www.rabbitsemiconductor.com
28. Slotine, J., Li, W.: Applied Nonlinear Control. Prentice Hall (1990)
29. Teuliere, C., Eck, L., Marchand, E.: Chasing a moving target from a flying uav. In: Int. Conference on Intelligent Robots and Systems, IROS, San Francisco, CA, pp. 4929–4934, 25–30 Sept 2011
30. Wise, R., Rysdyk, R.: Uav coordination for autonomous target tracking. In: Proceedings of the AIAA Guidance, Navigation and Control (2006)