HAL
open science

Tracking a Ground Moving Target with a Quadrotor Using Switching Control

Jose-Ernesto Gomez-Balderas, Gerardo Flores, Luis-Rodolfo Garcia Carrillo, Rogelio Lozano

To cite this version:

Jose-Ernesto Gomez-Balderas, Gerardo Flores, Luis-Rodolfo Garcia Carrillo, Rogelio Lozano. Tracking a Ground Moving Target with a Quadrotor Using Switching Control. International Conference on Unmanned Aircraft Systems (ICUAS 2012), Jun 2012, Philadelphia, United States. hal-00776094

HAL Id: hal-00776094

https://hal.science/hal-00776094v1

Submitted on 15 Jan 2013

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Tracking a ground moving target with a quadrotor using switching control

J.E. Gomez-Balderas†, G. Flores†, L.R. García Carrillo‡ and R. Lozano⋆

***Abstract*—An 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 images to a computer vision algorithm where images are processed. A vision-based estimator is proposed, which makes use of 2-dimensional images in order to compute the relative 3-dimensional position and translational velocity of the UAV with respect to the target. The proposed estimator provides the required measurements to a micro-controller 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.**

I. 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 varied sensors, more precisely, it requires data coming from UAV environment. A sensor capable to obtain abundant information of UAV environment is a vision system. Many results related to this topic have been presented in the last few years. Most of them are related to identification and classification of multiple targets [1], [2], [3][4], [5] and [6]. A circular pattern navigation algorithm for autonomous target tracking has been studied in [7] and [8] showing a good performance supported by numerical simulations. Other work trajectory acquisition from video includes particle filters for moving cameras without stabilization [9] and the Joint Probabilistic Data Association Filter (JPDAF) for tracking multiple targets with unreliable target identification [10]. If a model for the object's motion is known, an observer can be used to estimate the object's velocity [11]. In [12], an observer for estimating the object velocity was utilized; however, a description of the object's kinematics must be known. In [13] an autoregressive discrete time model is used to predict the location of features of a moving object. In [14], trajectory filtering and prediction techniques are utilized to track a moving object. In [15], 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 [16] a color based tracker is proposed to estimate the target position, while in [17] thermal images are correlated with a geographical information system (GIS) towards the same goal. In [18] a vision-based control algorithm for stabilizing a UAV equipped with two cameras is presented. The same system has been used in [19] 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 [20].

In this paper we describe the use of a vision system to observe a visual target 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 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

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).

† are with the Heudiasyc UMR 6599 Laboratory, University of Technology of Compiègne, France. Jose-Ernesto Gomez-Balderas (E-mail: jgomezba@hds.utc.fr), G. Flores (email: gfloresc@hds.utc.fr),

‡ is with Department of Electrical and Computer Engineering, University of California, Santa Barbara, CA 93106-9560, USA. E-mail: lrgc@ece.ucsb.edu

⋆ is with the Heudiasyc UMR 6599 Laboratory, UTC CNRS France and LAFMIA UMI 3175, Cinvestav, México. (email: rlozano@hds.utc.fr)

called loss of moving target (LOMT) that 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, the output of our algorithm is the 3D target position and its displacement 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 II. In section III, the control design is presented. The experimental platform used to validate theoretical results is described in section IV. Experimental results are presented in section V. Section VI presents conclusions and future work.

II. Image processing

Estimation of position and velocity of the quadrotor relative to the moving target is needed.

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 needed the estimation of $x - y$ position, altitude and relative velocity to the moving target of the quadrotor. An on board camera is used and the schema is shown in Figure 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.

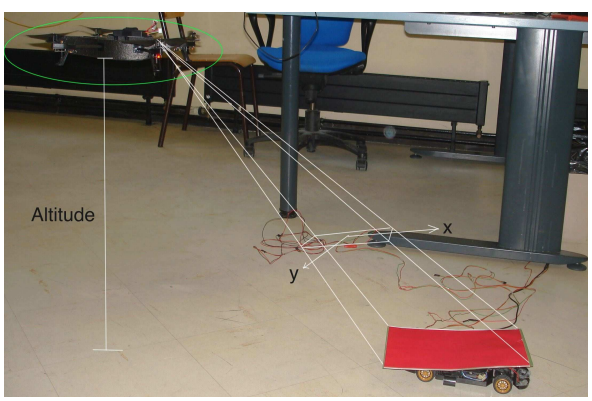

Fig. 1. Basic scheme

A. 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 [21] which is known to have the following characteristics: good detection, good 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 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

Fig. 2. Target detection

in four straight line segments.

B. 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 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 shown the result of target detection algorithm using four vertices, the 4 edges are painted in blue color.

C. Optical Flow measurement

In this paper, we use the optical flow from image sequences to estimate the translational speed of the X-4. The estimate of the horizontal speed will be used to perform autonomous hover flights without lateral displacement. There exist many different methods for computing the optical flow [22]. There exist 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 [23].

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

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

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

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

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

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

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, it is capable of handling large pixel motions and acts as a low pass filter. See [23] for a complete description of the algorithm.

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 Figure 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 ($\beta = 90°$) [22]. The velocity can be estimated of (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 mini-helicopter 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.

D. 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 (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 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 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 to indicate to UAV an change in yaw angle, to do that we use 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: $\psi = \arccos(l_1 \cdot l_2)$. 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 position algorithm obtain (X_W, Y_W, Z_W, ψ) that is the three dimensional position of X-4 and target orientation with respect to the X-4 camera, using optical flow algorithm we acquire $(\dot{x}, \dot{y})$ which is X-4 velocity estimation corresponding respectively to $(\dot{x_{ci}}, \dot{y_{ci}})$ in world coordinates. The

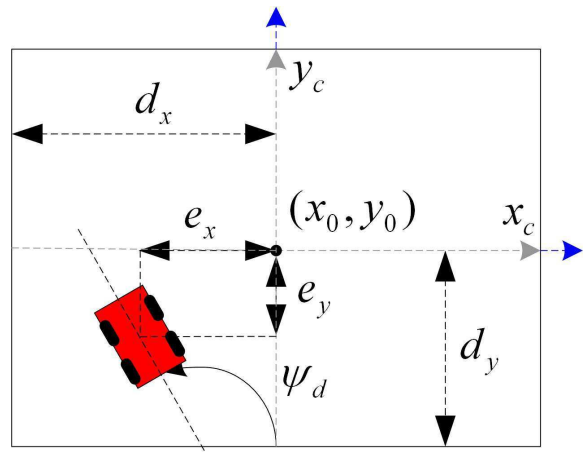

Fig. 3. Camera's view.

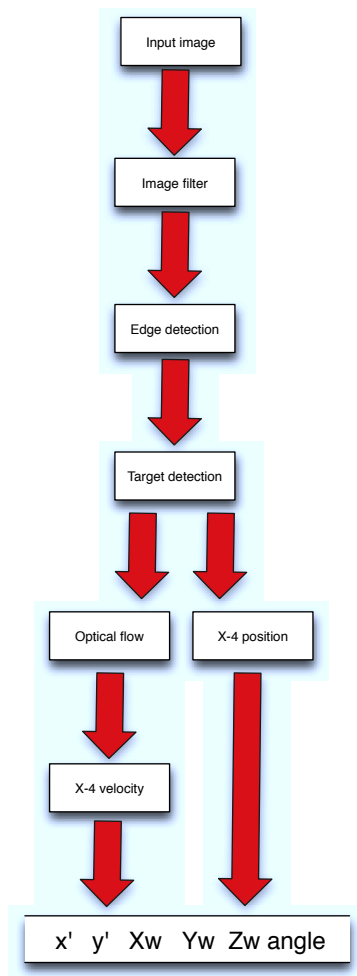

Fig. 4. Flowchart of image processing.

flowchart in Figure 4 explains the image processing algorithms used to obtain position, orientation and velocity data. Once we obtain this 6-tuple, we send this 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 our methods in real flight are shown in Figures 7 and 8.

III. Control

A common mathematical model for the quad-rotor is implemented aiming at developing the navigation control strategy [24]:

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

The quad-rotor model, presented in equation (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 equation (9) and the notation from equation (10), one has

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

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 [25].

A. 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.

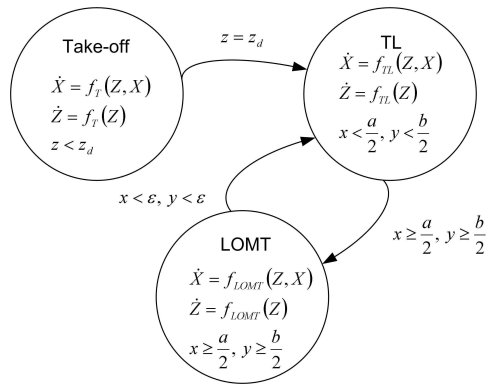

Fig. 5. Navigation control system

The navigation control is structured in different controllers for each sub-mission as illustrated in Figure 5.

B. 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 equation (11), is constituted of two subsystems, the attitude dynamics and the position dynamics, existing a time-scale separation between them [26]. 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 equation (11) are represented by the errors $\tilde{x}_i = x_i - x_{i_d}$ and $\tilde{z}_i = z_i - z_{i_d}$, with $i \in \{1, ..., 6\}$.

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 [27], lets select

the switching function

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

which depends on the pitch dynamics states. The λ parameter in equation (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 equation (12) one has

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

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

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

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

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

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

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

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

$$z_1 = z_{1_{eq}} - K sign(s) \qquad (17)$$

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

C. 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 a 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

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

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

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

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 behind 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 in change the altitude until the m.t. is found.

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

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

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 (11))is given by

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

The equation (21) can be represented as $\dot{X} = A_{mt}X$ where

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

- 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 $A_{\overline{mt}} =$

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

D. 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 (20,23).

Following a similar approach than the one presented in [28], 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 Figure 3. 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 easily 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 equation (25) is stable, since the matrices A_{mt} and $A_{\overline{mt}}$ are Hurwitz.

Suppose that there is a family A_p, $p \in \mathcal{P}$ of functions from $\Re^n$ to $\Re^n$, with $\mathcal{P} = 1, 2, ..., m$ 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 $\sigma : [0, \infty) \to \mathcal{P}$ 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 $\sigma(t) \in \mathcal{P}$ of the system that is actually active, at each instant of time t.

Theorem 1: Consider vectors t_{pq}, symmetric matrices S_p with $\Omega_p \in \{x : x^T S_p x \geq 0\}$, $\forall p \in \mathcal{P}$ 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 $\mathcal{C}^1$ trajectory of the system $\dot{x} = A_\sigma x$ tends to zero exponentially.

Proof: See the Appendix. ■

IV. Experimental platform

The total weight of the vehicle is about 500 gr, with a flight endurance of 10 minutes 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 512Kb RAM. These micro controllers are capable of handling floating point operations and multitasking processing virtually due to the enhancement compiler Dynamic C [29]. We have built our own inertial measurement unit (IMU) using accelerometers, gyros and a compass to obtain the roll, pitch and yaw angles and 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. Hence, the brushless motor speed controller or booster is handled by the I2C port of the micro controller. Visual sensor is composed of a wifi camera placed over X-4 platform, the camera is putting downwards to process an optical flow algorithm and position algorithm, our vision algorithm processes images coming from two cameras, at a rate of 18 frames/seconds. Each 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 2.10GHz using OpenCV libraries. Figure 6 shows a block diagram of the basic architecture.

V. Real time experiments

To test the proposed algorithms we realize several experiments. In this article we describe an experience in real flight over 180 seconds tracking a car with a translation motion at constant velocity. X-4 position and velocity estimation are obtained by image processing algorithm. In the first 10 seconds we observe the initialization of the algorithms, at this time the target detection

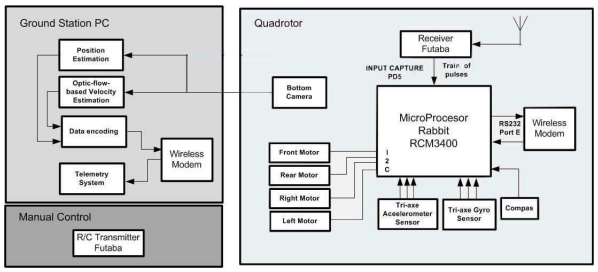

Fig. 6. Embedded Control Architecture

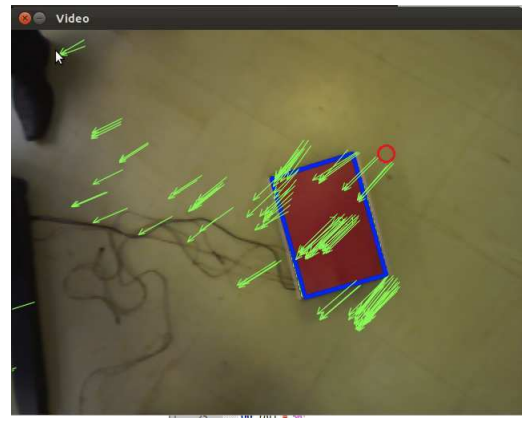

Fig. 7. X-4 Real time image

algorithms detect the target in the field of view of the camera, when the target was detected we obtain the position of the camera next the X-4 try to go over the target. After the initialization the control algorithm uses take-off (TO) navigation method with the desired altitude $z_d = 90$cm at 55 seconds, our target move faster than X-4 and the camera lost the target, with this information X-4 navigation method change to loss of moving target (LOMT) method increasing its altitude over 120cm. 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, we observe in Figure 11 how X-4 moves to desired altitude using a embedded switch control. Figure 9 shown X-4 displacements estimation over its X-axis on 180 seconds of real flight, we can see a initialization steps during the first 10 seconds of flight, after that, we observe the displacement of X-4 center of gravity and how X-4 embedded control try to move it to a desired orientation close to 60 cm over X-4 axis. In Figure 10 we shown X-4 displacements estimation over its Y-axis on 180 seconds of real flight. After 10 seconds of initialization step, X-4 embedded control moves it to a desired orientation over Y-axis, in our case 50 cm. Figure 12 and 13 shown X-4 velocity over X-axis and Y-axis respectively on 180 seconds of flight. Velocity estimation was obtained using optical flow algorithm.

VI. Conclusion

In this paper we have presented an UAV tracking a target over a ground vehicle based on a vision system that estimates position, orientation and displacement velocity of an UAV. The navigation mission uses different control algorithms: take-off (TO) trying to achieve to the desired altitude, target localization (TL) has like a goal to align the vehicle's center of gravity (CG) with respect to the moving target and loss of moving target (LOMT) 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 flight. The control strategy proposed can be used in absence oft he possibility of changing the field of view of the camera. The real-time experiments have shown an acceptable performance of the UAV applying the control navigation schema proposed. The authors are currently working on extending the strategies presented in this paper to work in different conditions such as a different form of the target, to do that some methods of pattern recognition like machine learning will be tested to provide more robust estimation of the form and location of the target.

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.

Proof: See [30], 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 $\mathcal{C}^1$, hence, $V(t)$ has the same characteristics. Premultiplying and postmultiplying the condition (27) by x, the inequality on the left side of (28) is satisfied. In the same way, inequality (29) follows if we premultiply and postmultiply both sides of (26) by x.

■

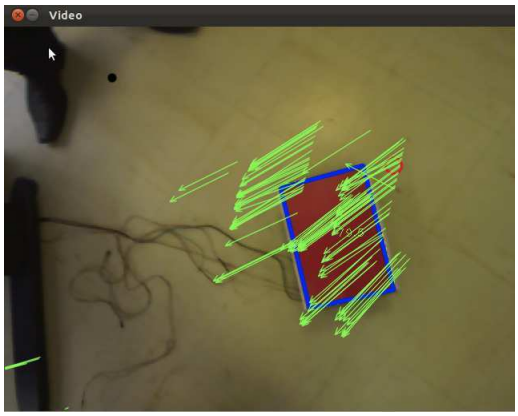

Fig. 8. X-4 Real time displacement

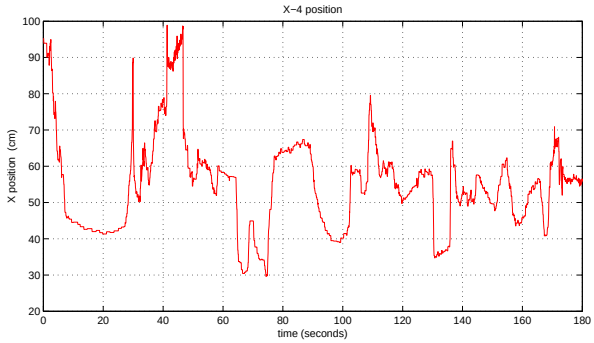

Fig. 9. X-4 Position over X-axis

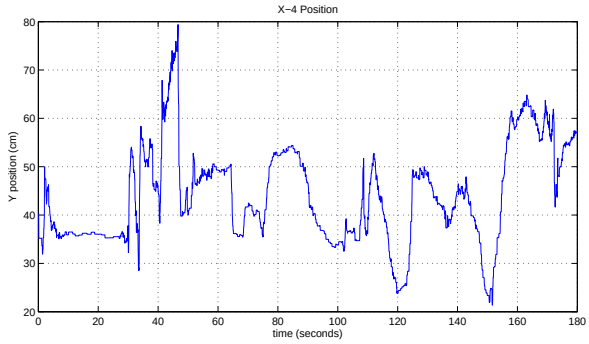

Fig. 10. X-4 Position over Y-axis

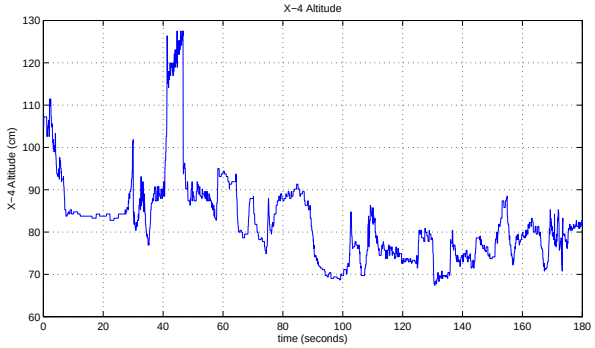

Fig. 11. X-4 Altitude

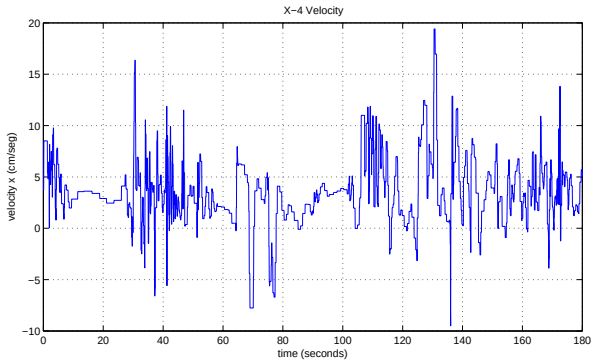

Fig. 12. X-4 velocity over X-axis

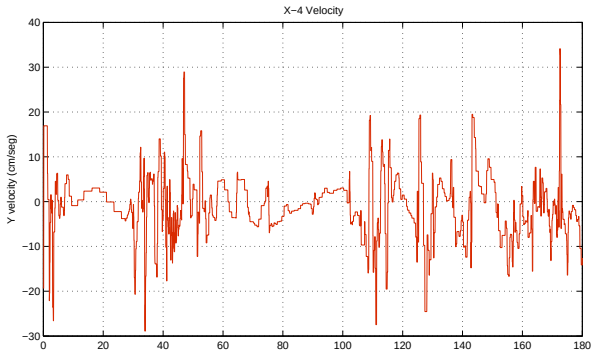

Fig. 13. X-4 velocity over Y-axis

REFERENCES

[1] T. Schmitt, R. Hanek, S. B. Michael Beetz, and B. Radig, "Cooperative probabilistic state estimation for vision-based autonomous mobile robots," *IEEE Transactions on robotics and automation*, vol. 18, no. 5, pp. 670–684, October 2002.

[2] I. Hwang, H. B. K. Roy, and C. Tomlin, "A distributed multiple-target identity management algorithm in sensor networks," in *Proceedings of the 43rd IEEE Conference on Decision and Control*, 2010.

[3] L. Hong, N. Cui, M. T. Pronobis, and S. Scott, "Simultaneous ground moving target tracking and identification using q wavelets features from hrr data," *Information Science*, vol. 162, pp. 249–274, 2004.

[4] V. K. Chitrakaran, D. M. Dawson, W. E. Dixon, and J. Chen, "Identification of a moving objects velocity with a fixed camera," *Automatica*, vol. 41, no. 3, pp. 553–562, 2005.

[5] A. P. Aguiar and J. P. Hespanha, "Minimum-energy state estimation for systems with perspective outputs," *IEEE Transactions on Automatic Control*, vol. 51, no. 2, pp. 226–241, February 2006.

[6] S. Mart?nez and F. Bullo, "Optimal sensor placement and motion coordination for target tracking," *Automatica*, vol. 42, no. 4, pp. 661–668, 2006.

[7] F. Rafi, S. Khan, K. Shafiq, and M. Shah, "Autonomous target following by unmanned aerial vehicles," in *Proceedings of the*, May 2006.

[8] R. A. and R. T. Rysdyk, "Uav coordination for autonomous target tracking," in *Proceedings of the AIAA Guidance, Navigation and Control*, 2006.

[9] J. Lee, R. Huang, A. Vaughn, X. Xiao, and J. K. Hedrick, "Strategies of path-planning for a uav to track a ground vehicle," in *Proceedings of the 2nd annual Autonomous Intelligent Networks and Systems Conference*, Menlo Park, CA, June 2003.

[10] Y. Bar-Shalom and T. E. Fortmann, *Tracking and Data Association*. MA: Academic Press, Boston, 1988.

[11] B. K. Ghosh and E. P. Loucks, "A realization theory for perspective systems with application to parameter estimation problems in machine vision," in *IEEE Transactions on Automatic Control*, vol. 41, 1706-1722, 1996, p. 12.

[12] K. Hashimoto and T. Noritsugu, "Observer-based control for visual servoing," in *Proc. 13th IFAC World Congress*, San Francisco, California, 1996, pp. 453–458.

[13] A. J. Koivo and N. Houshangi, "Real-time vision feed- back for servoing robotic manipulator with self-tuning controller," in *IEEE Transactions Systems, Man, and Cybernetics*, vol. 21, 1991, pp. 134–142.

[14] P. K. Allen, A. Timcenko, B. Yoshimi, and P. Michelman, "Trajectory filtering and prediction for automated tracking and grasping of a moving object," in *Proc. IEEE Conference on Robotics and Automation*, 1992, pp. 1850–1856.

[15] H. Shariat and K. Price, "Motion estimation with more than two frames," in *IEEE Transactions on PAMI*, vol. 12, 1990, pp. 417–434.

[16] C. Teuliere, L. Eck, and E. Marchand, "Chasing a moving target from a flying uav," in *Int. Conference on Intelligent Robots and Systems, IROS*, San Francisco, CA, 25-30 Sept., 2011, pp. 4929 – 4934.

[17] F. Heintz, P. Rudol, and P. Doherty, "From images to traffic behavior - a uav tracking and monitoring application," in *10th International Conference on Information Fusion*, 9-12 July, 2007.

[18] J. Gomez-Balderas, S. Salazar, J. Guerrero, and R. Lozano, "Vision based autonomous hover of a mini-rotorcraft," in *Unmanned Aerial Vehicles Symposium*, Dubai, Juin, 2010.

[19] J. Gomez-Balderas, P. Castillo, J. Guerrero, and R. Lozano, "Vision based tracking for a quadrotor using vanishing points," *Journal of Intelligent and Robotic Systems*, vol. 65, no. 1-4, pp. 361–371, January, 2012.

[20] S. S. Ponda, R. M. Kolacinski, and E. Frazzoli, "Trajectory optimization for target localization using small unmanned aerial vehicles," in *AIAA Guidance, Navigation, and Control Conference*, Chicago, Illinois, USA, 10-13 August, 2009.

[21] J. Canny, "A computational approach to edge detection," *IEEE Trans on Pattern Analysis and Machine Intelligence*, vol. PAMI-8, no. 6, pp. 679 – 698, Nov. 1986.

[22] S. S. Beauchemin and J. L. Barron, "The computation of optical flow," *ACM Computing Surveys*, vol. 27, no. 4, pp. 433–467, 1995.

[23] J.-Y. Bouguet, "Pyramidal implementation of the lucas kanade feature tracker description of the algorithm," Intel Corporation Microprocessor Research Labs, Tech. Rep., 2000.

[24] L. G. Carrillo, E. Rondon, A. Sanchez, A. Dzul, and R. Lozano, "Stabilization and trajectory tracking of a quad rotor using vision," *Journal of Intelligent and Robotic Systems*, vol. 61, no. 1-4, pp. 103–118, January, 2011.

[25] S. Bouabdallah and R. Siegwart, "Backstepping andd sliding-mode techniques applied to an indoor micro quadrotor," in *Proceedings of the IEEE International Conference on Robotics and Automation*, Barcelona, Spain, 2005, pp. 2259–2264.

[26] P. Kokotović, H. K. Khalil, and J. O'Reilly, *Singular Perturbation Methods in Control: Analysis and Design*. London: Academic Press: Siam, 1999.

[27] J. Slotine and W. Li, *Applied Nonlinear Control*. Prentice Hall, 1990.

[28] D. Liberzon, *Switching in Systems and Control*. Boston: Birkhuser, 2003.

[29] M. Shell. (2007) Rabbit semiconductors, dynamics c user manual. [Online]. Available: http://www.rabbitsemiconductor.com/

[30] H. K. Khalil, *Nonlinear Systems*. New York: Prentice Hall, 2002.