3D model-based tracking for UAV position control

Céline Teulière, Laurent Eck, E. Marchand, Nicolas Guenard

3D model-based tracking for UAV position control

Céline Teulière, Laurent Eck, Eric Marchand, Nicolas Guénard

***Abstract*— This paper presents a 3D model-based tracking suitable for indoor position control of an unmanned aerial vehicle (UAV). Given a 3D model of the edges of its environment, the UAV locates itself thanks to a robust multiple hypothesis tracker. The pose estimation is then fused to inertial data to provide the translational velocity required for the control. A hierarchical control is used to achieve positioning tasks. Experiments on a quad-rotor aerial vehicle validate the proposed approach.**

I. INTRODUCTION

Unmanned aerial vehicles (UAVs) have a large range of indoor or outdoor applications such as surveillance, search and rescue, inspection for maintenance, etc. To achieve such tasks as autonomously as possible, UAVs are usually equipped with at least one embedded camera. The visual information received is used for the vehicle control, combined or not with other available sensors like an inertial measurement unit (IMU).

For safe vision-based navigation or servoing tasks, the choice of the visual features to use is crucial since their ability to be robustly matched and tracked will determine the good realization of the task. This choice mainly depends on the knowledge we have about the environment. Feature points, (Sift points, Harris corners...) can be extracted and tracked in any textured environment, which makes them particularly suitable for outdoor applications, in unknown areas. For this reason, they have been used in various aerial applications, from structure from motion [13] [1] to optical flow computation [8]. However, in indoor structured environments with nude floor and walls, feature points can be less frequent, leading to robustness issues. They are also sensitive to the noise produced by transmission interferences.

In most image-based visual servoing (IBVS) approaches, the target is assumed to be known [3] [6], and the error to regulate is expressed directly in the image. On the other hand, position-based visual servoing (PBVS) uses visual information to retrieve the relative pose (position and orientation) between the embedded camera and the target, to use it in the control loop [16], [17]. The main difficulty for this kind of approach is then to get a robust estimate of the pose.

In this paper we propose to use a robust model-based tracking to estimate the pose of the UAV through its motion. The tracking algorithm requires a 3D CAD model of the edges of the environment. This condition is obviously not suitable for unstructured outdoor missions, but could perfectly be fulfilled in indoor inspection tasks. Moreover, edges are very frequent in such structured environments, and robust to some illumination changes, or even noise due to transmission interferences. In [9] a model-based tracking was already applied to UAV navigation, with a different tracking approach. Payload constraints prevented the author to consider fusion with inertial sensors which makes the system more sensitive to tracking errors. In our work, the pose estimate obtained using a 3D tracking is fused with inertial data to estimate the translational velocity of the vehicle, required for the control. Position-based visual servoing has been experimented on a quad-rotor aerial vehicle developped by CEA LIST [6], which is capable of stationary or quasi-stationary flight. The next section gives an overview of the overall system.

Fig. 1. Quad-rotor in indoor environment.

II. OVERVIEW

In this paper we propose a model-based vision system for the position control of a quadrotor (see Figure 3). The vehicle considered is equipped with camera attached to the airframe of the UAV. In our experiments the camera is pointing downward. The vehicle is also equipped with an inertial measurement unit (IMU). Except for the low level embedded attitude control, the computations are deported to a ground station. The data are transmitted between the ground station and the UAV via a radio transmission. For clarity purpose, the system is divided into 3 parts which are more precisely described in the next sections (see Figure 2).

- *The control scheme* is presented in section III. Estimations of the position and translational velocity of the vehicle are used to achieve a positioning task.
- *The visual tracking subsystems* aims at providing an estimate of the 3D pose of the camera with respect to the world frame. Given a 3D CAD model composed of

C. Teulière, N. Guénard and L. Eck are with CEA, LIST, Interactive Robotics Unit, 18 route du Panorama, BP6, Fontenay-aux-Roses, F- 92265, France firstname.name@cea.fr

E. Marchand is with Université de Rennes 1, IRISA, INRIA, Lagadic Project, Rennes, France eric.marchand@irisa.fr

This work was realized in the context of the French ANR national project SCUAV (ANR Psirob 06_174032 SCUAV project ref ANR-06-ROBO-0007-02)

edges of the environment, its role is to evaluate which camera pose provides the best alignment between the model projected edges and the edges detected in the image. To achieve this in a robust way, the proposed approach considers multiple hypothesis. This tracking system is presented in section IV.

- *The velocity estimator* relies on an extended Kalman filter which fuses inertial data with the pose estimate given by the vision subsystem to estimate the translational velocity of the vehicle. It also filters the tracked pose. Section V describes the estimation process.

Experiments using the quad-rotor UAV are presented in section VI.

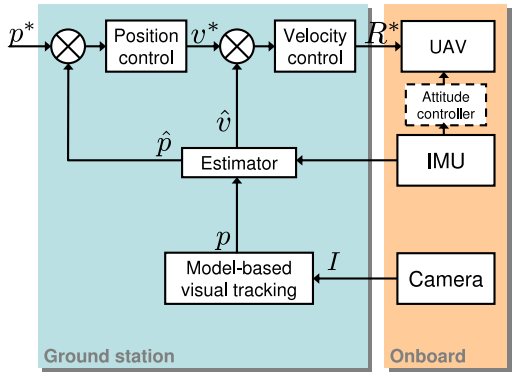

Fig. 2. Overview of the system.

III. Control scheme

In this section we present the control scheme used for the position control. First the motion equations for the UAV in quasi-stationary flight conditions are given. Then the control laws are presented.

A. UAV Modelling

The UAV is represented by a rigid-body of mass m and of tensor of inertia $\mathbf{I} \in \mathbb{R}^{3\times3}$.

Let us define the frame $\Re_c$ attached to the vehicle in its centre of mass, and assume it coincides with the camera frame (see figure 4). The position of the centre of mass of the vehicle relative to the world frame ${}^w\mathbf{p}_c$ is denoted by $\mathbf{p}$. For simplicity of notation the rotation ${}^w\mathbf{R}_c$ of the body frame $\Re_c$ relative to $\Re_w = (\mathbf{e}_x, \mathbf{e}_y, \mathbf{e}_z)$ is denoted by $\mathbf{R}$. Let $\mathbf{v}$ (respectively $\mathbf{\Omega}$) be the linear (resp. angular) velocity of the center of mass expressed in the world frame $\Re_w$ (resp. in $\Re_c$). The control inputs to send to the vehicle are: T, a scalar input termed thrust or heave, applied in direction $\mathbf{e}_z$ and $\mathbf{\Gamma} = [\mathbf{\Gamma}_x \mathbf{\Gamma}_y \mathbf{\Gamma}_z]^\top$ the control torques relative to the Euler angles. Assuming the world frame is Galilean, Newton's equations of motion yield the following:

$$\begin{cases} \dot{\mathbf{p}} = \mathbf{v} \\ m\dot{\mathbf{v}} = T\mathbf{R}\mathbf{e}_z - f\mathbf{v}^2\mathbf{u}_v + mg\mathbf{e}_z \\ \dot{\mathbf{R}} = [\mathbf{\Omega}]_\times \\ \mathbf{I}\dot{\mathbf{\Omega}} = -\mathbf{\Omega} \times \mathbf{I}\mathbf{\Omega} + \mathbf{\Gamma} \end{cases} \tag{1}$$

where g is the gravity constant. $f\mathbf{v}^2\mathbf{u}_v$ is a friction force opposed to the direction of motion $\mathbf{u}_v$, with f a friction coefficient. The notation $[\mathbf{a}]_\times$ denotes the skew-symmetric matrix associated with any vector $\mathbf{a} \in \mathbb{R}^3$ such that for any vector $\mathbf{b} \in \mathbb{R}^3$, $[\mathbf{a}]_\times \mathbf{b} = \mathbf{a} \times \mathbf{b}$.

The quad-rotor UAV is an underactuated system with 4 inputs. Its translational motion results from the rotations. In this work we assume that the system's attitude is already controlled onboard. Therefore, our control scheme acts as a controller sending orientation commands to a low-level controller which is responsible for robust flight.

Fig. 3. Quad-rotor aerial vehicle used in our experiments.

B. Yaw control

The yaw angle ψ is controlled by using a proportional controller.

$$\mathbf{\Omega}_z^* = -K_{p\psi}(\psi^* - \psi) \tag{2}$$

ψ^* is the desired yaw angle. The yaw velocity $\mathbf{\Omega}_z$ is then controlled onboard using gyrometers.

C. Translational control

The translational control architecture is illustrated in Figure 2. The position and velocity errors are defined by:

$$e_\mathbf{p} = \mathbf{p}^* - \mathbf{p} \tag{3}$$

$$e_\mathbf{v} = \mathbf{v}^* - \mathbf{v} \tag{4}$$

where $\mathbf{p}^*$ is the desired position, that is the position we want the vehicle to reach.

We use a hierarchical control. The inner-loop is a PI controller on the velocity, and the outer-loop a simple proportional control on the position:

$$\mathbf{v}^* = -K_p e_\mathbf{p} \tag{5}$$

The inner-loop on the velocity is required to ensure the stability of the system. It acts as a damping in the UAV control. This control scheme thus requires good estimates of the position (at least $\mathbf{p}$ and ψ) and velocity $\mathbf{v}$ of the vehicle. The next sections describe how they are obtained.

IV. Model-based tracking system

In this section the visual tracking system is presented. The role of this system is to provide an estimate of the relative pose between the moving camera and the fixed environment using a 3D model of linear edges.

The issue of model-based tracking has been widely investigated in the past years [11] and different approaches have been proposed to address it. These approaches can be divided into two classes:

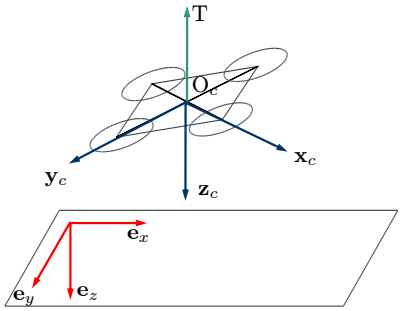

Fig. 4. Frame definitions

- *Registration methods* use non linear optimization techniques (Newton minimization, virtual-visual servoing,...) to find the pose which minimizes a given reprojection error between the model and the image edges [12], [5], [4]. The robustness of these methods has been improved by using robust estimation tools [2], [5], [4]. However, they can fail in case of ambiguities (different edges can have very similar appearances) or large displacements.
- *Bayesian methods*, on the other hand, have been recently proposed to achieve the same task by estimating the probability density associated to the pose [15], [10], [14]. In particle filtering, this density is represented by a set of samples (the particles) each of them corresponding to a potential pose. Each particle is associated with a weight depending on how well it reprojects the model with regard to what is observed in the image.

For the considered application, the tracking algorithm has to be robust to noise and interferences (See Figure 6), to large displacements and run fast enough for the control to be effective. In a previous work [18], we proposed a multiple hypothesis registration process and embedded it in a particle filtering framework. This method provided a robust pose estimate. However, it is too computationnaly expensive for using it directly in the control loop of a UAV.

In this paper, we propose a simplified version, adapted to real-time capabilities. The multi-hypothesis registration approach is similar to [18] without particle filtering. Instead, the filtering is achieved in the Kalman filter which fuses inertial and visual measurements (see section V). This section recalls the principles of our multiple hypothesis tracker.

A. Considering multiple low level hypothesis

In order to track the relative pose between the camera and the modelled environment, our multiple hypothesis tracker relies on a similar basis than the one used in [4], [5] and [19]. Assuming the camera parameters and an estimate of the pose are known, the CAD model is first projected into the image according to that pose, which can be the previous one or a prediction obtained from a filter (see section V). The projection of an edge l_i of the 3D model according to the pose ${}^c\mathbf{M}_w$ is denoted by $E_i = l_i({}^c\mathbf{M}_w)$. Each projected edge E_i is then sampled, giving a set of points $\{e_{i,j}\}$ (see Figure 8). From each sample point $e_{i,j}$ a search is performed along the edge normal to find strong gradients.

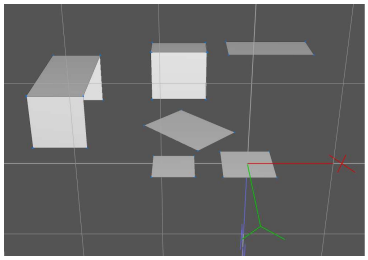

Fig. 5. Example of 3D model.

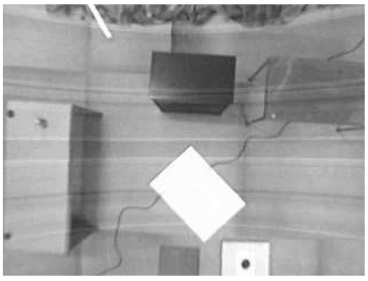

Fig. 6. Corresponding scene (undistorted).

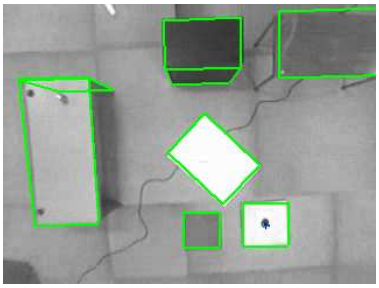

Fig. 7. Tracking result.

As illustrated in Figure 8, two close edges can lead to ambiguities when several strong edges are found along the normal to the contour. In [4] and [5], the point of maximum likelihood with regard to the initial point $e_{i,j}$ is selected. It is denoted by $e'_{i,j}$ in the following. An optimization method is then used to find the camera pose which minimizes the errors between the selected points and the projected edges [12], [4] and [5], that is:

$$\widehat{{}^c\mathbf{M}_w} = argmin_{{}^c\mathbf{M}_w} \sum_{i,j} d_\perp(E_i, e'_{i,j}) \tag{6}$$

where $d_\perp(E_i, e'_{i,j}) = d_\perp(l_i({}^c\mathbf{M}_w), e'_{i,j})$ is the squared distance between the point $e'_{i,j}$ and the projection E_i of the linear segment l_i of the model. The quantity to minimize is then expressed by:

$$S = \frac{1}{N_e} \sum_i \sum_j \rho\left(d_\perp(E_i, e'_{i,j})\right) \tag{7}$$

where N_e is the total number of sampled points, and ρ is a robust estimator.

In our approach, we keep several low level hypothesis $\{e'_{i,j,l}\}$ corresponding to local extrema of the image gradient along the edge normal in $e_{i,j}$. Instead of performing one single minimization from these points like in [19] resulting in one single pose, we go from these multiple low level

hypothesis to multiple hypothesis on the camera pose itself. The next section explains how this is achieved.

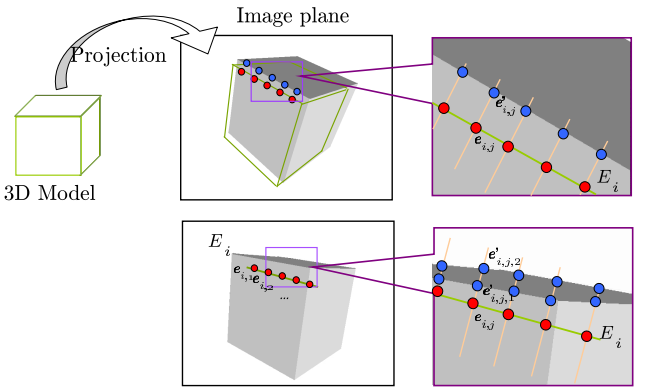

Fig. 8. In classic edge based tracking, the model is projected into the image plane and points are sampled on the projected edges. A search is performed along the normal (top). When multiple strong edges are close in the image, ambiguities can occur when searching along the normal (bottom).

B. Segmenting the low level hypothesis into edge hypothesis

In order to get multiple hypothesis on the camera pose corresponding to the detected low level hypothesis, several minimizations can be performed, using different sets of points in (7). Since considering all the possible combinations of points is obviously not an option, we first determine the underlying lines of the set of points $\{e'_{i,j,l}\}$, to group the points into different sets corresponding to potential edges (see Figure 9). This is achieved using a k-mean classification algorithm [7]. For each projected edge E_i, the algorithm segments the candidate points $\{e'_{i,j,l}\}$ into k_i sets of points or classes $(c^i_1, ..., c^i_{k_i})$. The mean of each of the k_i classes is in our case the line which best fits the points of that class, obtained by a robust least square minimization. The number k_i of classes for the edge E_i is set to the maximum number of candidate points detected, that is: $k_i = max_j\{n_{i,j}\}$. The classes $(c^i_1, ..., c^i_{k_i})$ are initialized using the order in which the hypothesis have been found on the normal. That is for each class c^i_m: $c^i_m = \{e'_{i,j,m}\}_j$. At each iteration of the algorithm, the mean line of each class is computed. Each point is then assigned to the class with the nearest mean line. The algorithm is deemed to have converged when the assignments no longer change.

Finally, the k-mean algorithm corresponding to the initial edge E_i provides us with a set of classes $c^i_m = (\{e'_{i,j,m}\}_j, r^i_m)$ where r^i_m is the residue of the least square minimization, and represents a likelihood criterium that will be used in the next step. In practice, only lines with a sufficient number of points are taken into account. In most cases k_i does not exceed two or three. Figure 9 illustrates this process.

C. From edge hypothesis to pose hypothesis

Once candidates have been obtained for each edge in the form of sets of points associated to a residue, random weighted draws are performed. Weights w^i_m considered for each candidates are deduced from the residues by:

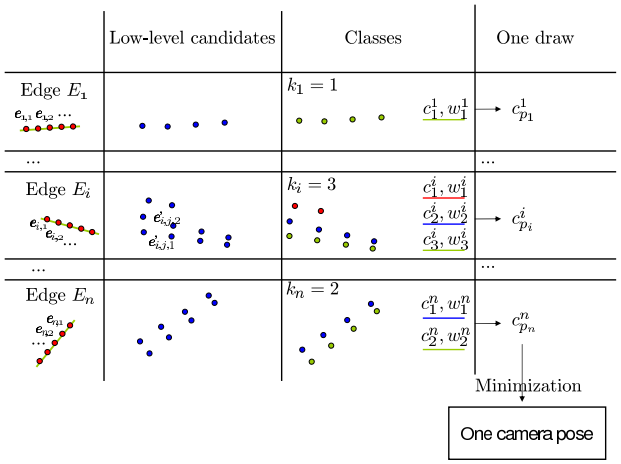

Fig. 9. Classes of points are extracted from low level hypothesis,. For each projected edge a random weighted draw is performed among the classes to determine the points that will be used for the minimization process. The minimization provides a candidate pose.

$$w^i_m = \begin{cases} e^{-\lambda\left(\frac{r^i_m - r^i_{min}}{r^i_{max} - r^i_{min}}\right)^2} & \text{if } r^i_{max} \neq r^i_{min} \\ 1 & \text{otherwise.} \end{cases} \quad (8)$$

where λ is a parameter that can be tuned according to the selectivity that is desired.

For each edge E_i a class $c^i_{p_i}$ is drawn from the k_i classes. From the resulting set of points, a numerical non-linear minimization is performed according to equation (9), resulting in a camera pose. Different set of points, built from different draws among the low level detected hypothesis, thus lead to different potential camera poses.

$$S = \frac{1}{N_e} \sum_i \sum_{e'_{i,j,l} \in c^i_{p_i}} \rho\left(d_\perp(E_i, e'_{i,j,l})\right) \quad (9)$$

The weighted draw allows to favour among all the possible combinations the ones with the candidates of lowest residue, which are more likely to correspond to real edges. The process is illustrated in Figure 9.

In practice, since the number of candidate lines per edge is small, so will be the number of optimizations to be performed and thus the number of pose candidates obtained. In [18] those hypothesis were integrated in a particle filter. Here, the pose giving the smallest residue is selected as the current estimate, and sent to the velocity estimator which is described in the next section.

V. Translational velocity estimation

The visual tracking system provides an estimate of the full 3D pose of the vehicle. However, in order to build a control scheme, an estimate of the translational velocity is also required (see Figure 2 and section III).

To take advantage of the pose estimate from the vision system and the data from the IMU, an extended Kalman filter (EKF) is used. The state to estimate is $\mathbf{X} = \left(\mathbf{p}^\top \mathbf{v}^\top \mathbf{a}^\top\right)^\top$ where $\mathbf{a}$ is the acceleration of the UAV. The proposed

model equations are derived from the translational dynamics equation in (1) with the following simplifying assumptions:

- the x and y velocities are assumed to be decoupled
- the friction coefficient f is assumed to be constant, independant of the direction of the motion
- the small angles assumption is made, which is reasonnable in quasi-stationary flight
- T is assumed to be quasi-constant. In indoor applications, without wind perturbations, the thrust has small variations which can be ignored in first approximation. Errors will be compensated by the visual observation.

From equation (1) the following discrete model equations are derived:

$$ {a_x}^{(t+\delta t)} = \frac{T}{m}\phi^{(t-\tau)} - \text{sign}(v_x)\frac{f}{m}{v_x}^2 + n_{ax} \quad (10) $$

$$ {a_y}^{(t+\delta t)} = \frac{T}{m}\theta^{(t-\tau)} - \text{sign}(v_y)\frac{f}{m}{v_y}^2 + n_{ay} \quad (11) $$

where τ is a possible time delay. The attitude angles θ and ϕ are respectively the pitch and roll angles, obtained from the IMU. For the vertical translation, a constant acceleration model has been used:

$$ {a_z}^{(t+\delta t)} = {a_z}^{(t)} + n_{az} \quad (12) $$

n_{ax}, n_{ay}, n_{az} are assumed to be the components of a white noise $\mathbf{n_a} = \mathcal{N}(0, \mathbf{Q_a})$. $\mathbf{Q_a}$ is the covariance matrix associated with the acceleration model. Then, the position and velocity are simply deduced by:

$$ \begin{cases} v^{(t+\delta t)} = v^{(t)} + a^{(t+\delta t)}\delta t \\ p^{(t+\delta t)} = p^{(t)} + v^{(t+\delta t)}\delta t. \end{cases} \quad (13) $$

Note that the constant τ, $\alpha = \frac{T}{m}$ and $\beta = \frac{f}{m}$ have been learnt, in our case from a genetic algorithm, but could be derived experimentally from any other estimation technique.

Since the model equations (13) are non-linear, an extended Kalman filter (EKF) is used. The Jacobian matrix $\mathbf{J_x}$ is given by:

$$ \mathbf{J_x} = \begin{bmatrix} \mathbf{I} & \delta t\mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{I} & \delta t\mathbf{I} \\ \mathbf{0} & -\text{sign}(v)2\beta v a & \mathbf{0} \end{bmatrix} \quad (14) $$

The prediction of the covariance matrix of the state $\mathbf{P}$ is then given by:

$$ \mathbf{P}_{(t+\delta t)|t} = \mathbf{J_x}\mathbf{P}_{t|t}{\mathbf{J_x}}^T + \mathbf{J_n}\mathbf{Q}{\mathbf{J_n}}^T \quad (15) $$

where

$$ \mathbf{Q} = \begin{bmatrix} \mathbf{I} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{Q_a} \end{bmatrix} \quad (16) $$

$\mathbf{J_n}$ is the matrix of the derivative of the model equation relative to the noise components.

The observation is simply the position estimate $\hat{\mathbf{p}}$ given by the visual tracking system.

$$ \mathbf{X}_{t+\delta t} = \begin{bmatrix} \mathbf{I} & \mathbf{0} & \mathbf{0} \end{bmatrix}^\top \begin{bmatrix} \hat{\mathbf{p}} \end{bmatrix} \quad (17) $$

VI. Experimental validation

This section presents the experiments conducted on the quad-rotor (X4-flyer) developped by the CEA LIST (Figure 3). The UAV sends the images from its embedded camera to the ground station (PC) via a wireless analogical link of 2.4GHz. In parallel, it also sends inertial data from the IMU at a frequency of 15Hz. The data is processed on the ground station and the desired orientation and thrust are sent back to the quad-rotor vehicle. Onboard, the exponential stability of the orientation toward the desired one is ensured by a 'high gain' controller (in the DSP running at 166Hz) [6]. One of the difficulties of such systems comes from the time latency between the inertial and visual data. On the ground station, the overall system (visual tracking, velocity estimation and control computation) runs with a framerate of 20Hz. There is a time delay between the time the image is acquired on the embedded camera and the time the desired attitude is computed and reached by the vehicle. This delay is roughly estimated and used in the prediction of the acceleration (equation (10)).

A. Velocity estimation

Figure 10 and 11 show the velocity obtained with our filter (in red) as compared to a simple differentation of the positions given by the tracking system (in green) and the velocity obtained with the prediction model alone (in blue). The differentiation between consecutive frames gives poor results. The filtered velocity is smoothed with little time lag thanks to the prediction model.

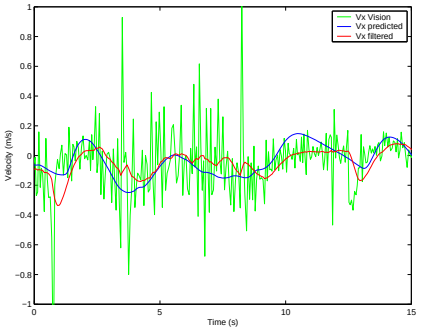

Fig. 10. Velocity on x axis.

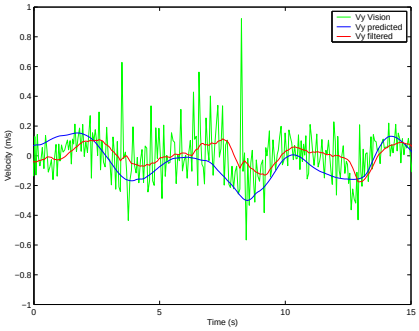

Fig. 11. Velocity on y axis.

B. Position control

The proposed approach was validated on positioning tasks. A scene was built, combining planar and 3D objects (see

Figure 5). The tracking initialization has not been considered in this paper. During the experiment, the tracking is automatically initialized by detecting a black dot in its first position (Figure 6). Then the vehicle can locate itself thanks to the model-based tracking, without using the dot anymore. The task considered was to autonomously reach several set points successively: first the vehicle is stabilised 2 meters above the dot target ($\mathbf{p} = (0, 0, -2)$). Then the set points are successively set to $(0, -2, -2)$, $(-2, -2, -2)$, $(0, -2, -2)$, $(0, 0, -2)$, $(-2, 0, -2)$. The desired yaw angle is set to 0 all along the sequence. A video of the quad-rotor performing this sequence is submitted with this paper.

Figure 12 and 13 show the position error (3) on each axis. Peaks on x and y axis correspond to the manual change of the set points. At convergence, the UAV stays within 15cm of the desired position on x and y axis, and up to 30cm on z axis. Figure 14 gives the velocity error (4). Small oscillations on the velocity error are mainly due to latencies.

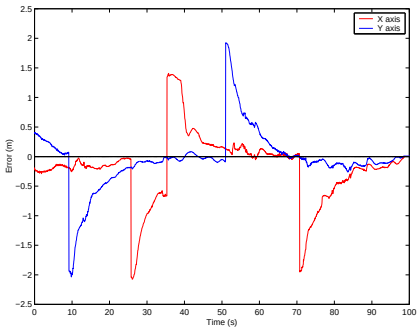

Fig. 12. Positioning error on x (red) and y (blue) axis.

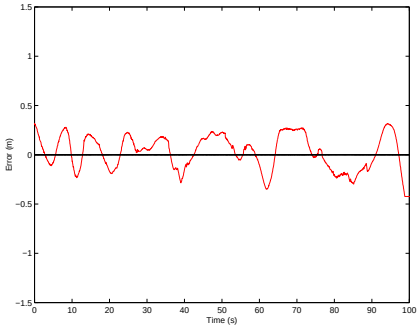

Fig. 13. Positioning error on z axis.

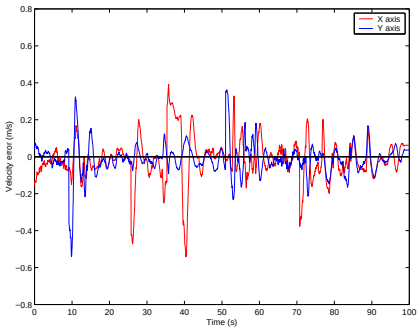

Fig. 14. Velocity error on x (red) and y (blue) axis.

The small velocities observed confirm that the small angles hypothesis is satisfied. The aerial vehicle was able to achieve the task with a satisfactory behavior.

VII. CONCLUSIONS

In this paper, we proposed a model-based tracking suitable for the position control of a UAV in indoor environment. A multiple hypothesis tracker provides an estimate of the relative pose between the vehicle and its environment in real time. It is also robust to the noise produced by transmission interferences. By fusing this pose with inertial measurement, a velocity estimate is obtained, and the position is filtered. These two estimates allow to perform the position control of the UAV. Although no ground truth was available in the experiment to evaluate the precision of the position and velocity estimates, which we plan to do next, the experiment shows the feasibility of the proposed approach in indoor structured environments.

REFERENCES

[1] A. Angeli, D. Filliat, S. Doncieux, and J.-A. Meyer. 2D simultaneous localization and mapping for micro aerial vehicles. In *Proceedings of the European Micro Aerial Vehicles (EMAV 2006) conference*, 2006.

[2] M. Armstrong and A. Zisserman. Robust object tracking. In *Proc. Asian Conference on Computer Vision*, volume 1, pp 58–61, 1995.

[3] O. Bourquardez, R. Mahony, N. Guenard, F. Chaumette, T. Hamel, and L. Eck. Image-based visual servo control of the translation kinematics of a quadrotor aerial vehicle. *IEEE TRO*, 25(3), June 2009.

[4] A.I. Comport, E. Marchand, M. Pressigout, and F. Chaumette. Real-time markerless tracking for augmented reality: The virtual visual servoing framework. *IEEE Trans. on visualization and computer graphics*, 12(4):615–28, July/August 2006.

[5] R. Drummond, T. Cipolla. Real-time visual tracking of complex structures. *IEEE PAMI*, 24(7):932–946, 2002.

[6] N. Guenard, T. Hamel, and Mahony. A practical visual servo control for an unmanned aerial vehicle. *IEEE T. on Robotics*, 24(2):331–340, April 2008.

[7] J.A. Hartigan. *clustering algorithms*. John Wiley & Sons, 1975.

[8] B. Herisse, T. Hamel, R. Mahony, and F-X. Russotto. A nonlinear terrain-following controller for a vtol unmanned aerial vehicle using translational optical flow. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, St Louis, USA, October 2009.

[9] Christopher Kemp. *Visual control of a miniature quad-rotor helicopter*. PhD thesis, University of Cambridge, 2006.

[10] G. Klein and D. Murray. Full-3d edge tracking with a particle filter. In *British Machine Vision Conf.*, volume 3, pp 1119–1128, 2006.

[11] V. Lepetit and P. Fua. Monocular model-based 3d tracking of rigid objects: A survey. In *Foundations and Trends in Computer Graphics and Vision*, pp 1–89, 2005.

[12] D.G. Lowe. Fitting parameterized three-dimensional models to images. *IEEE PAMI*, 13:441–450, 1991.

[13] M. Meingast, C. Geyer, and S. Sastry. Vision based terrain recovery for landing unmanned aerial vehicles. In *IEEE CDC*, vol 2, pp 1670–1675, dec. 2004.

[14] S. Nuske, J. Roberts, and G. Wyeth. Visual localisation in outdoor industrial building environments. In *IEEE Int. Conf. on Robotics and Automation*, pp 544–550, 2008.

[15] M. Pupilli and A. Calway. Real-time camera tracking using known 3d models and a particle filter. In *Int. Conf. on Pattern Recognition*, pp 199–203, August 2006.

[16] S. Saripalli, J.F. Montgomery, and G.S. Sukhatme. Visually guided landing of an unmanned aerial vehicle. *IEEE Trans. on Robotics and Automation*, 19(3):371–380, june 2003.

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

[18] C. Teuliere, E. Marchand, and L. Eck. Using multiple hypothesis in model-based tracking. In *IEEE Int. Conf. on Robotics and Automation*, Anchorage, Alaska, May 2010.

[19] L. Vacchetti, V. Lepetit, and P. Fua. Combining edge and texture information for real-time accurate 3d camera tracking. In *IEEE/ACM Int. Symp. on Mixed and Augmented Reality*, pp 48–57, 2004.