AR Drone Identification and Navigation Control at CVG-UPM

Centre for Automation and Robotics, joint research centre CSIC-UPM, www.vision4uav.com

J. Pestana, J. L. Sanchez-Lopez, I. Mellado-Bataller, Changhong Fu and P. Campoy
jesus.pestana;jl.sanchez;fu.changhong@upm.es, ignacio.mellado@gmail.com, pascual.campoy@upm.es

Resumen

This article presents the proposal of the Computer Vision Group to the first phase of the international competition "Concurso de Ingeniería de Control 2012, Control Autónomo del seguimiento de trayectorias de un vehículo cuatrirrotor". This phase consists mainly of two parts: identifying a model and designing a trajectory controller for the AR Drone quadrotor. For the identification task, two models are proposed: a simplified model that captures only the main dynamics of the quadrotor, and a second model based on the physical laws underlying the AR Drone behavior. The trajectory controller design is based on the simplified model, whereas the physical model is used to tune the controller to attain a certain level of robust stability to model uncertainties. The controller design is simplified by the hypothesis that accurate positions sensors will be available to implement a feedback controller.

Keywords: multirotor UAV control, control engineering, system identification.

1 Introduction

Due to their unique features, multirotors are becoming a competitive platform to perform commercial civilian tasks. Their maintainance costs are lower than a traditional helicopter's, due to the fact that they do not have complex servomechanisms. The size of the propellers of a multirotor compared to a similar payload counterpart are smaller, which is less dangerous for humans. The capability to hover of the quadrotors empower their maneuverability near obstacles and in constrained spaces. These features along with the modern technology capabilities, in terms of computing power and the great quantity of information that can be sent to a ground station, are enabling the use of multirotors for multiple civilian tasks. Some example applications are aerial mapping, inspection tasks in construction sites, post-disaster damage assessment for insurance estimations and agriculture, environmental and wildlife monitoring.

Taking into account its size and weight, the AR Drone quadrotor, see Fig. 1, can be considered a Micro Unmanned Aerial Vehicle (MUAV) comparable in size to other commercial quadrotors that are being used for civilian tasks. Thus, the AR Drone is a cheap platform for research projects that aim to demonstrate the viability of civilian quadrotor applications. The "Concurso de Ingeniería de Control 2012, Control Autónomo del seguimiento de trayectorias de un vehículo cuatrirrotor" (CEA 2012) competition is a perfect opportunity for undergraduate and postgraduate students to be involved in MUAV research projects.

Commercial MUAV applications involve video and data streaming to a ground station, and task-level remote control by a human supervisor. In order to achieve a good user experience the UAV has to autonomously perform lower-level navigation tasks. These tasks can be classified in three groups: hovering in position, following a trajectory and obstacle avoidance. Each of them is receiving a great research effort, mainly due to the fact that mapping, localization and state estimation, and environment awareness are not fully solved robotics problems. An interesting approach is to divide this technical problem in three separate subparts:

1. Position and state estimation: the high and mid-level control laws require a certain amount of accurate information about the quadrotor speed and position relative to the environment. This problem is not fully solved, but specific approaches give an acceptable estimation for simplified environment models. Inside buildings, the most usual approach is the usage of laser range finders (such as Hokuyo URG-04LX-UG01) and state of the art SLAM algorithms [1]. Another approach is using a Kinect sensor and point cloud mapping algorithms [2, 3]. In outdoors applications, the research objective is to improve the GPS position measurements and decrease the dependence on GPS technology [4]. For example, regarding computer vision applications, 3D monocular vision inspired on Parallel Tracking and Mapping algorithms are being tested on MUAV [5, 6].

2. Control problem: the objective is to develop a control layer that offers an appropriate interface to the task-level supervisor, such as GPS checkpoint navigation. Latest research projects have demonstrated, using motion capture systems, that the quadrotor control problem can be solved successfully when accurate position measurements are available [7, 8, 9]. The issues with this approach are the requirement of a structured environtment, and that the control law is strongly dependent on the available sensors and positioning algorithms. On the other hand, the STARMAC project shows competitive outdoors performance[10, 11], relying only on GPS.

3. Environment awareness: the environment model utilized by a robot limits the hazardous situations to which it can react successfully, thus limiting the commercial applications of mobile robots in unstructured environments. For instance, the ability to avoid an obstacle is limited by the type of obstacles that the robot can detect [12].

In this article two general methodologies for the quadrotor model identification problem and a checkpoint navigation controller architecture are proposed. First, two quadrotor models along with two corresponding identification methodologies are presented in Sects. 2, 3 respectively. Second, the controller architecture is proposed in Sect. 4. And finally, the simulation results of both, the identification and control tasks of the competition, are discussed in Sect. 5.

2 Vehicle Dynamics modeling

Figura 1: AR Drone black-box model. The specified inputs match the real system's. The euler angles and the altitude outputs are available from the AR Drone telemetry data. To the contrary, the only available data from the drone about the X and Y directions are speed measurements obtained by means of optical flow algorithms.

The AR Drone quadrotor accepts attitude and altitude commands, as illustrated in Fig. 1. The task of the embedded controller is to perform the required data processing and calculate the necessary propellers commands to achieve the commanded attitude and altitude. This task usually utilizes a decoupling of the propeller commands and four separate control loops, an explanation of these low-level control laws can be found in [10, 7].

The existence of this low-level control layer permits us to work with a simplified black-box model. In this section, first, the complete quadrotor model is introduced, and second, it is used to infer a simplified model for the AR Drone.
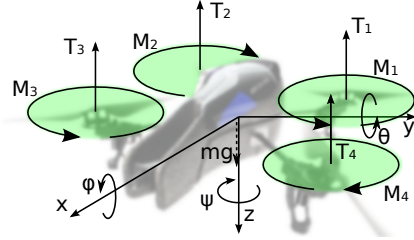
2.1 Physical model

Figura 2: Free body diagram of a quadrotor. The four propellers, and each of their performed thrust T_i and torque M_i are labeled 1-4. The euler angles are denoted $\{\phi, \text{roll}\}$, $\{\theta, \text{pitch}\}$ and $\{\psi, \text{yaw}\}$

A quadrotor consists of a rigid body driven by four propellers, each one of them developing a mechanical force T_i, and a torque M_i, see Fig. 2. It is generally accepted to approximate their mechanical effort as $T_i = k_T \cdot \omega^2$ and $M_i = k_M \cdot \omega^2$, where ω, k_M and k_T are the propeller speed and two propeller characteristic constants. The usual mid-level control approach involves using [10, 7]: the average thrust $\sum_{i=1}^{4} T_i$ as control variable for the altitude, the average torque $\sum_{i=1}^{4} M_i$ as control variable for the yaw heading of the vehicle, and the roll and pitch angles that tilt the average thrust to obtain two linear acceleration commands in the horizontal plane. The quadrotor rigid body dynamics model, see Eq. 1 and Fig. 2, is easy to infer from physical laws and is explained in multiple articles [10, 7].

$$
\begin{cases}
I_x \ddot{\phi} = \dot{\psi}\dot{\theta}(I_y - I_z) + l(T_4 - T_2) \\
I_y \ddot{\theta} = \dot{\phi}\dot{\psi}(I_z - I_x) + l(T_1 - T_3) \\
I_z \ddot{\psi} = \dot{\theta}\dot{\phi}(I_x - I_z) + \sum_{i=1}^{4} M_i \\
m\ddot{x} = (s\phi\, s\psi + c\phi\, s\theta\, c\psi) \sum_{i=1}^{4} T_i \\
m\ddot{y} = (c\phi\, s\theta\, s\psi - s\phi\, c\psi) \sum_{i=1}^{4} T_i \\
m\ddot{z} = mg - c\theta\, c\phi \sum_{i=1}^{4} T_i
\end{cases}
\tag{1}
$$

The following symbols are used in the equations and in later discussion:

- the euler angles are denoted $\{\phi, \text{roll}\}$, $\{\theta, \text{pitch}\}$ and $\{\psi, \text{yaw}\}$

- the quadrotor rigid body is characterized by its mass m and its three principal mass moments of inertia $\{I_x,\ I_y,\ I_z\}$

- the l constant is the distance between the centers of each pair of opposite propellers, such us propellers 1 and 3 in Fig. 2

To obtain a more accurate simulation, it may be necessary to model, as explained in [11, 7]: the saturation in the command variables, the propeller dynamics, the friction and other aerodynamic disturbances, the communication sampling time and delay between the UAV and the ground station; and the sensor noise and measurement errors. The damping coefficients in the horizontal plane, that model the aerodynamic friction, serve to have a good steady-state model fixing the correspondance between roll and pitch values to steady-state horizontal speed values.

In the case of the AR Drone, see Fig. 1, it is also necessary to simulate the low-level control laws. The goal is to have a simulated AR Drone that behaves similarly to the real platform in terms of response time in roll, pitch, yaw speed and altitude speed.

2.2 Simplified model

To simplify the identification and control problem, the organizers of the CEA 2012 competition have decided to constrain the quadrotor movement to the horizontal plane and also to fix the yaw heading. Thus, leaving roll and pitch as the only command variables. An inspection of the underlying dynamic model, from Eq. 1, reveals that the pitch angle controls the speed in the X direction, and the roll angle controls the speed in the Y direction.
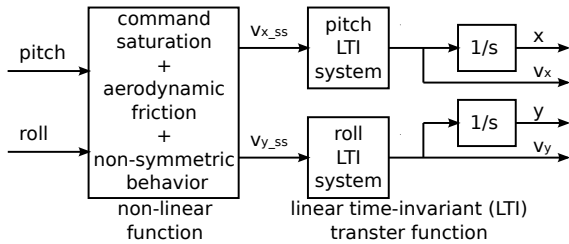

Figura 3: Quadrotor simplified model, only the pitch and roll inputs are kept, along with the main resulting quadrotor behavior.

Taking into account these contraints the simplified model depicted in Fig. 3 is proposed:

- A non-linear function models and fixes the steady-state correspondance between $\{\phi, \theta\}$ and $\{v_x, v_y\}$. The quadrotor characteristics modeled by this non-linear function are:

- aerodynamic friction: it can be expressed as a correspondance between constant input values and steady-state speed values $\{v_{x\_ss}, v_{y\_ss}\}$. This relationship can be non-symmetric, $[v_{x\_ss},\ v_{y\_ss}] = \mathbf{f}(\theta,\ \phi)$, i. e. presenting different maximum speed values for each direction.

- maximum tilt angle: expressed by a 2D command saturation $(\phi/\phi_{max})^2 + (\theta/\theta_{max})^2 \leq 1$
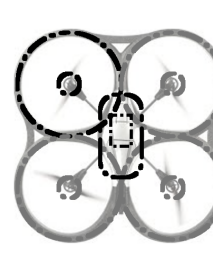
- The dynamics of the quadrotor are modeled by a linear time invariant (LTI) system, i. e. a transfer function. The LTI system models: the roll and pitch closed loop response, and the resultant damped speed response. Instead of including all the high frequency poles and the whole physical behavior of the drone, the actual transfer function can be simplified and just include the slowest poles/dynamics.

3 System identification

Designing and testing a controller on the same model is dangerous, because it does not check the stability robustness to uncertainties in the model parameters. This is the reason why two quadrotor models are utilized. The simplified model is used for the controller design. Then both, the simplified and the physical models, are used to check the closed loop system stability.

3.1 Physical model

This model includes the dynamic simulation of Eq. 1, and also approximately identifies the drone's low-level controller. The steps followed to identify this model are:

parameter	val.	units
m_{body}	104	g
$m_{battery}$	119	g
$m_{housing}$	62.0	g
m_{motor}	37.8	g
I_x	4.50	g m^2
I_y	5.10	g m^2
I_z	9.50	g m^2

Figura 4: Theoretical AR Drone mass distribution and related model parameters

1. estimation of mass and inertia: the mass of the drone has been measured using an adequate weight balance. The inertia is then estimated using the theoretical mass distribution shown in Fig. 4 where: some components of the drone were weighed separately, the motors are point masses, the body and battery

are parallelepipeds, and the housing protection mass is divided in four circumferences.

2. low-level attitude and altitude controller design: from experimental data it was estimated that the roll and pitch response times are approximately 150-300 ms. The model includes PID controllers, similar to those explained in [10, 7], that match these response times. The yaw and altitude loops were designed to obtain a speed response time of about 400-500 ms.

3. additional parameter tuning: the aerodynamic friction was fixed to approximately fit the data provided by the CEA competition organizers using only a simple friction law $F_x = -k_x \left| \mathbf{v} \right| v_x$, $F_y = -k_y \left| \mathbf{v} \right| v_y$.

3.2 Simplified model

The identification of the simplified model, depicted in Fig. 3, requires the realization of two separate parameter fitting tasks; which are carried out in the following order:

1. Non-linear steady-state correspondance, shown in Fig. 5:

    - for each step response, calculate the mean of steady-state data, i. e. $\{\theta_{ss\_i}, v_{x\_ss\_i}\}$, $\{\phi_{ss\_i}, v_{y\_ss\_i}\}$. This data is used to obtain a piecewise linear interpolant.
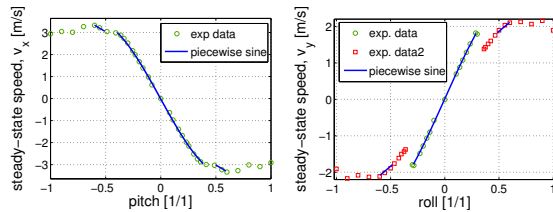   - use the Matlab's Curve Fitting Toolbox to obtain a piecewise sine regressor

Figura 5: The aerodynamic friction is modeled using a non-linear function that relates input values $\{\theta_{ss\_i}, \phi_{ss\_i}\}$ to steady-state output speed values $\{v_{x\_ss\_i}, v_{y\_ss\_i}\}$. The graph shows the samples used to obtain the piecewise linear interpolants and the piecewise sine regressors

2. Identification of the LTI block:

    - perform data preprocessing: first the input of the LTI block $\{v_{x\_ss}, v_{y\_ss}\}$ is calculated, see Fig. 3, using the previously obtained aerodynamic friction models. Then the experimental data is divided in step response experiments. This part

of the process involves: eliminating relatively long, compared to the system's response time, periods of zero input and also eliminating the initial output offset from all experiments

- use the Matlab's System Identification Toolbox to identify the LTI block of the simplified model, shown in Fig.3. The obtained model is summarized in Fig. 6

$$G(s) = \frac{K\left(\frac{s}{w_n}+1\right)}{\left(\frac{s}{w_n}\right)^2 + \left(\frac{2\xi}{w_n}\right)s + 1}$$

	θ, v_x	ϕ, v_y
K	.979	.980
ξ	.724	1.35
w_n	1.79	.852
w_z	-	6.33

Figura 6: The simplified model, Fig. 3, contains a LTI block to approximataly take into account the dynamic response of the real system. The figure shows: (left) LTI model (right) parameter values

The results of the identification process are shown in Figs. 5, 6, 7. A piecewise linear interpolant is used as non-linear function for the model simulation, and the inverse of the piecewise sine regressor is used inside the control law to compensate the non-linear aerodynamic friction.
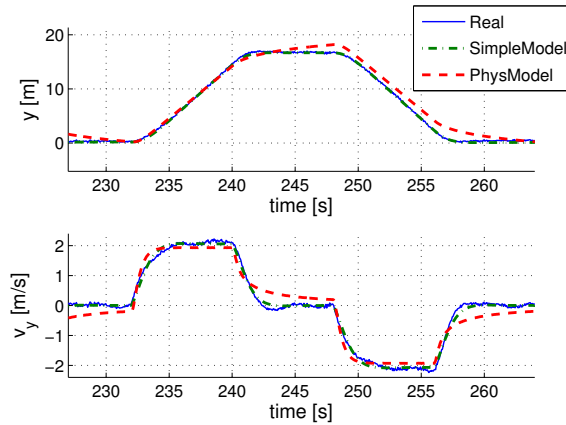

Figura 7: Simulations of both the simplified model and the physical model compared to step response system data, which was also used as train data.

Having an accurate steady-state model is critical for the proposed controller due to the fact that it has a speed planner. For example, in the case that non-reachable speeds were planned, the speed controller would saturate and the position controller would not work properly; thus, drastically increasing the trajectory tracking error.

The original competition identification dataset only provides reliable steady-state data along the X and Y axes, as shown on Fig. 8. The CVG team's additional petition dataset consists of a
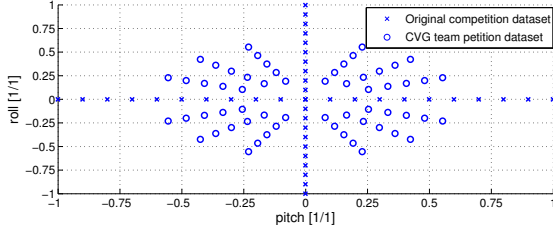
Figura 8: Domain of definition of the simplified model's aerodynamic friction. The data indicated that the inputs $\{\theta, \phi\}$ are decoupled, the curves shown in Fig. 5 were obtained. This graph shows the available samples from both: the original competition dataset, and the petition dataset.

sequence of step commands sampling twelve additional directions, see Fig. 8. Thus, it contains information about the system response in all directions, not only in the X and Y directions. The values of the step commands were selected in the boundary of the linear region of the curves shown in Fig. 5, and also in the linear region itself. The analysis of the new data shows that both commands $\{\theta, \phi\}$ are decoupled in steady-state, i. e. the model behavior can be modeled using the two separate functions $[v_x, \ v_y] \ = \ [f_1(\theta), \ f_2(\phi)]$ shown in Fig. 5.

4 Trajectory controller design

The goal for the controller design task is to make the AR Drone follow a path specified by a sequence of checkpoints. Due to the performance indicators defined by the CEA 2012 competition organizers the path has to be as close as possible to the straight segments connecting consecutive checkpoints. This fact has influenced the present proposal, particularly the path and speed planners. The general controller architecture, shown in Fig. 9, consists of: a mid-level controller, that calculates the actual commands that will be sent to the drone; and a high-level controller that tracks the quadrotor position relative to the desired path and calculates speed and position references for the mid-level controller.

In Figs. 9, 10, 11 and in the following discussions:

- $\{v_x, x, y, v_y\}$ are the AR Drone measurements and may contain noise

- $\{v_{xf}, x_f, y_f, v_{yf}\}$ are the filtered measurements, which are used inside the Finite-State Machine (FSM) and the mid-level controller

- $\{v_{xc}, dx_c, dy_c, v_{yc}\}$ are the mid-level controller references calculated by the FSM. They can also be considered position and speed commands.
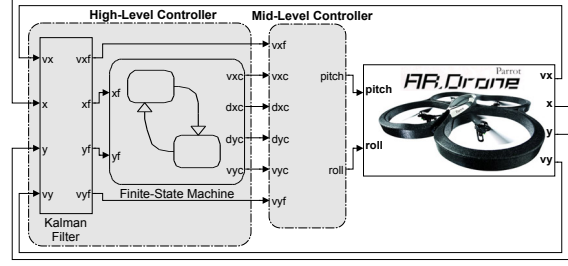

Figura 9: General architecture of the controller proposal. In order to minimize the route time, the controller includes a speed planner and a state machine that send optimized reachable speed commands to the mid-level controller

4.1 High-level controller

The first component of the high-level controller is a Kalman Filter (KF), see Fig. 9. Its purpose is to filter the signal noise from the measurements. The model used to obtain the KF is the simplified model, shown in Fig. 3. Nevertheless, the same KF works rather well with both models, the simplified and the physical models presented in Sects. 2.1, 2.2. It has been necessary to add a constant wind disturbance model to the KF model in order to make the KF work correctly in presence of wind disturbances.

The second component is a FSM, which uses the filtered signals to calculate the mid-level controller references; thus, generating filtered references. The FSM has three states, as shown in Fig. 10, corresponding to three control strategies: hover in a desired position, follow a straight segment of the trajectory and turn to head to the next trajectory segment. The work of the FSM can be summarized as the repetition of the following three steps:

1. Plannify the speed along the next straight segment of the trajectory, the actual algorithm is inspired on the work [10] and takes into account the initial speed and the radius of the next turn. The result is a planified speed value, $v_{plan}(s)$, as a function of the intrinsic path coordinate s

2. Follow the straight segment, accelerating at first, but then slowing down as the drone approaches the next turn

3. Perform the turn, controlling the speed direction to achieve a soft alignment with the next straight segment

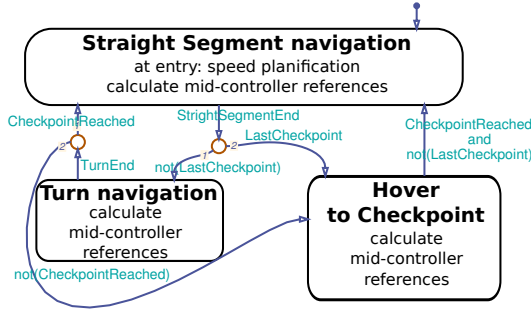The middle-level controller receives position and speed commands from the FSM, as shown in

Figura 10: High-level control finite state machine. The states correspond to navigation control strategies. The state transitions are activated depending on the current mission status and on the position of the drone relative to the trajectory.

Fig. 11. The calculation for the mid-level references is inspired on those presented in [10, 7]. These commands are calculated by the FSM in such a way that the relative position and speed commands are orthogonal. Depending on the current state of the FSM, see Fig. 10, these commands are calculated as follows:

- Hover to a desired position $\mathbf{r_{desired}}$:

$$\begin{cases} [dx_c, \, dy_c] & = & \mathbf{r_{desired}} - [x, \, y] \\ [v_{xc}, \, v_{yc}] & = & [0, \, 0] \end{cases} \quad (2)$$

- Follow straight segment in direction $\mathbf{u_r}$: the position error command is calculated relative to the nearest segment point, $\mathbf{r_{nsp}}$. The speed command is parallel to $\mathbf{u_r}$, its magnitude being the planned speed $v_{plan}(s)$ for the current position in the trajectory segment.

$$\begin{cases} [dx_c, \, dy_c] & = & \mathbf{r_{nsp}} - [x, \, y] \\ [v_{xc}, \, v_{yc}] & = & v_{plan}(s) \, \mathbf{u_r} \end{cases} \quad (3)$$

- Perform a turn: the speed command is constant in magnitude and tangent to a circle arc that joins both consecutive trajectory segments. The arc is calculated so that the trajectory passes at a distance $d = R_{conf}/2$ from the checkpoint. Where $d_{max} = R_{conf}$ is the maximum distance at which the checkpoint is considered reached. The position error command is calculated relative to the nearest point of the arc $\mathbf{r_{circ}}$.

$$\begin{cases} [dx_c, \, dy_c] & = & \mathbf{r_{circ}} - [x, \, y] \\ [v_{xc}, \, v_{yc}] & = & v_{turn}\mathbf{u_{circ}} \\ v_{turn} & = & \sqrt{R_{curve} \, a_{max}} \end{cases} \quad (4)$$

4.2 Middle-level controller

The mid-level controller is shown in Fig.11. Its general architecture is a cascade controller consisting of an inner speed loop and and outer position loop. But it also includes control laws to take
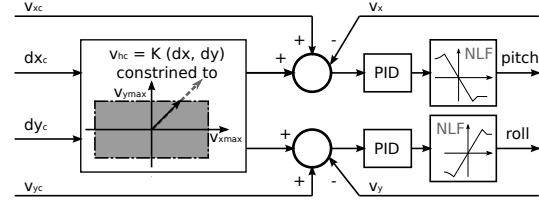

Figura 11: Mid-level controller architecture. The proposal is a cascade controller, which consists of an inner speed loop and an outer position loop. The controller includes non-linear laws to take into account the drone's maximum speed capabilities

into account the AR Drone non-linearities. This enhancements are the following, observe Fig. 11:

- The position controller loop only commands achievable speeds to the speed controller loop, see left block in Fig. 11

- The aerodynamic friction is partially linearized using the inverse of the simplified model aerodynamic friction, namely the piecewise sine functions shown in Fig. 5

- The controller is constrained to work on the most linear part of the model, i.e. the linear part of the $\{\theta, v_x\}$ and $\{\phi, v_y\}$ curves shown in Fig. 5

- The planned velocity is lower than the maximum velocity, thus, giving the relative position controller a speed margin to work on

The controller stability robustness was succesfully tested. This means that little parameter and model uncertainties will not unstabilize the controller. The performed tests are the following:

- uncertainty in the LTI blocks of the simplified model:
  - simultaneously introducing 5% changes on the model parameters $\{K, \xi, w_n, w_z\}$
  - adding one additional pair of underdamped fast poles to the LTI blocks, having a response times about 5-10 faster than the LTI corresponding block

- testing the controller with the physical model

5   Results

The identification task results are demonstrated, first, comparing the identified models with real system data in two simulations shown in Figs. 7, 12, and second, using the performance indicators shown in Table 1 proposed by the competition organizers. The portion of the identification

data shown in Fig. 12 is used only as test dataset, and represents 5% of the total available data for the identification task.

The models' step response is shown in Fig.7, and the response to an arbitrary sequence of step commands is shown in Fig. 12. The simple aerodynamic friction law used in the physical model does not fit well the speed steady-state data, thus, this model has more tendency to cumulate position error. As a consequence the position error indicators are not evaluated for the physical model, and in favour of a better visual comparison, the prior cumulative position error was substracted from the physical model position plot shown in Fig. 12. Table 1 shows the identification performance indicators for the simplified model, which were evaluated with the original dataset (set1) and with the CVG petition dataset (set2), both with a duration of 720 s. As a conclusion, the dynamic behavior of both models is good, as shown in Figs. 7, 12. And the worst mean estimation error is 41 cm, as shown in Table 1, which extensively satisfies our team's performance expectations.
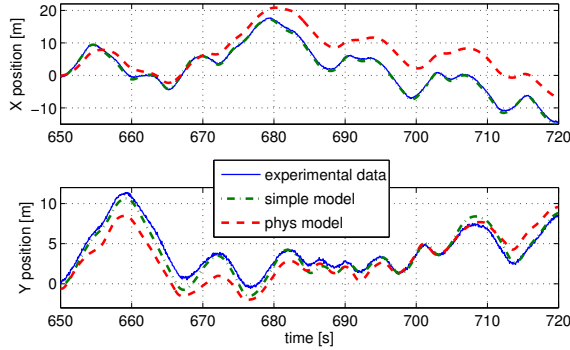

Figura 12: Simulations of both the simplified model and the physical model compared to response system data to an arbitrary sequence of step commands. The system data shown in this graph was only used as test data.

data	Δx_{mean}	Δy_{mean}	Δx_{max}	Δy_{max}
set1	0.41 m	0.40 m	1.61 m	1.73 m
set2	0.17 m	0.33 m	0.73 m	1.35 m

Tabla 1: Identification performance indicators, tests run only on the simplified model, (set1) original dataset (set2) petition dataset

The controller design is tested in simulation only: a path following example is shown in Fig. 13 and Table 2 contains performance indicators for the controller. It is necessary to consider the system kinematic saturations in order to properly judge the table values, which are: $v_{xmax} = 3.3$ m/s, $v_{ymax} = 2.2$ m/s, $a_{xmax} = 2$ m/s^2 and $a_{ymax} = 0.9$ m/s^2. In Fig. 13, the drone fol-

lows the trajectory shown in the bottom chart, the speed control loop variables are shown in the upper chart and the FSM state is specified between the speed plots. The acceleration and deceleration during the straight segment state and the lower velocity commands during turn states can be observed in the speed graphs.

When testing the controller with the simplified model there are two possible parameter configurations, one optimizing position precision and another minimising the route time. The controller performance indicators for both configurations following the 57 m length path shown in the bottom graph of Fig. 13 are shown in the first two rows of Table 2. Some of the FSM parameters are changed in order to optimize the regulated system response when the physical model is used to simulate the AR Drone. The resulting control indicators are shown in the third row of Table 2.
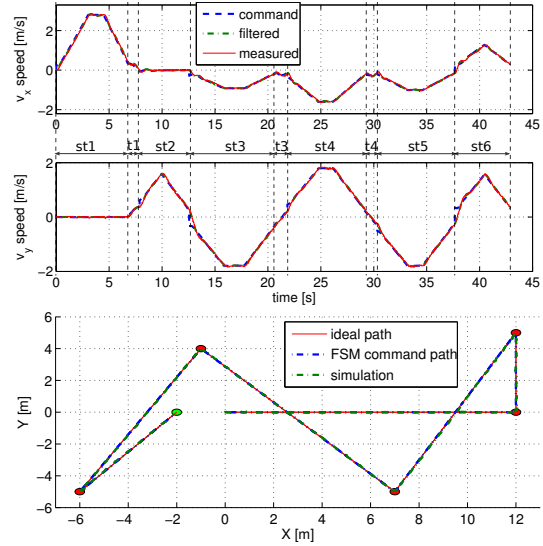

Figura 13: Trajectory controller performance using the high precision parameter configuration, the system is simulated using the simplified model. FSM states: {st, straight line} {t, turn}

configuration	d_{mean}	d_{max}	v_{mean}
high speed[1]	.037 m	.226 m	1.80 m/s
high precision[1]	.016 m	.187 m	1.33 m/s
physical model[2]	.165 m	.701 m	1.60 m/s

Tabla 2: Controller performance indicators, tests run on [1] simplified model, [2] physical model. d is the distance to the ideal trajectory and the mean route speed is calculated as $v_{mean} = \dfrac{\text{path length}}{\text{route time}}$

As a conclusion, the proposed trajectory controller can operate the drone assuring stability, as discussed at the end of Sect. 4.2. The quadrotor is operated in such a way that mean speeds of about 1.33 to 1.8 m/s are achieved with mean distances

to the ideal trajectory lower than 20 cm, as shown in Table 2. In order to achieve this values, the speed is preplanned with acceleration and deceleration phases in the straight segments of the trajectory; and navigating at sufficiently lower velocities during turns, as shown in Fig. 13. The position loop controller has to be optimized further, for it to work better with the drone's pyshical model.

6   Conclusion

This article presents the proposal of the CVG-Team to the first phase of the CEA 2012 international competition. More information about the competition can be found at *http://www.ceautomatica.es/og/ingenieria-de-control/benchmark-2011-2012* . A complete and a simplified model for the AR Drone quadrotor were presented, along with a methodology to identify the model parameters in both cases. A controller architecture to solve the checkpoint sequence navigation problem was presented and implemented in simulation using Matlab Simulink.

The control scheme presented in this article achieved the best controller score in phase 1 of the CEA 2012 competition. Moreover, this controller obtained the best score regarding average distance error, $0.10\ m$, and maximum distance error, $0.62\ m$, to the desired trajectory; and also the least roundtrip time with an average speed of $1.96\ m/s$. The model identification obtained the second best identification score. The team attained the second position in the first phase of the competition.

It is important to note that the controller has been designed assuming that accurate position measurements are available. Thus, the first task to be accomplished in future work is the design of a state estimation algorithm enabling this situation. The final goal of the presented work is providing the means to enable the use of MUAV in multiple civilian tasks.

Referencias

[1] A. Bachrach, R. He, and N. Roy, "Autonomous Flight in Unstructured and Unknown Indoor Environments," *International Journal of Micro Air Vehicles 1 (2009)*, pp. 217–228, 2011.

[2] F. Leberl, A. Irschara, T. Pock, P. Meixner, M. Gruber, S. Scholz, A. Wiechert, and S. Scholz, "Point Clouds : Lidar versus 3D Vision," *Photogrammetric Engineering & Remote Sensing*, vol. 76, no. 10, pp. 1123–1134, 2010.

[3] A. S. Huang, A. Bachrach, P. Henry, M. Krainin, D. Maturana, D. Fox, and N. Roy, "Visual odometry and mapping for autonomous flight using an rgb-d camera," in *Int. Symposium on Robotics Research (ISRR)*, (Flagstaff, Arizona, USA), Aug. 2011.

[4] A. Irschara, C. Hoppe, H. Bischof, and S. Kluckner, "Efficient Structure from Motion with Weak Position and Orientation Priors," in *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR), Workshop on Aerial Video Processing*, 2011.

[5] G. Klein and D. Murray, "Parallel Tracking and Mapping on a camera phone," in *2009 8th IEEE International Symposium on Mixed and Augmented Reality*, pp. 83–86, Oct. 2009.

[6] A. Wendel, M. Maurer, A. Irschara, and H. Bischof, "3D Vision Applications for MAVs: Localization and Reconstruction," in *International Symposium on 3D Data Processing, Visualization and Transmission (3DPVT)*, pp. 3–4, 2011.

[7] B. Y. N. Michael, D. Mellinger, and Q. Lindsey, "The GRASP Multiple Micro UAV Testbed," *IEEE Robotics & Automation Magazine*, no. September, pp. 56–65, 2010.

[8] L. S., S. A., S. M, and R. D'Andrea, "A Simple Learning Strategy for High-Speed Quadrocopter Multi-Flips," in *2010 IEEE International Conference on Robotics and Automation (ICRA 2010)*, pp. 1642–1648, 2010.

[9] A. Schölling, F. Augugliaro, S. Lupashin, and R. D'Andrea, "Synchronizing the Motion of a Quadrocopter to Music," in *IEEE International Conference on Robotics and Automation ICRA*, pp. 3355–3360, 2010.

[10] G. M. Hoffmann, S. L. Waslander, and C. J. Tomlin, "Quadrotor Helicopter Trajectory Tracking Control," *Electrical Engineering (2008)*, pp. 1–14, 2008.

[11] J. H. Gillula, H. Huang, M. P. Vitus, and C. J. Tomlin, "Design of Guaranteed Safe Maneuvers Using Reachable Sets : Autonomous Quadrotor Aerobatics in Theory and Practice," *IEEE Electrical Engineering (2010)*, pp. 1649–1654, 2010.

[12] L. Mejias and I. F. M. P. Campoy, "Omnidirectional bearing-only see-and-avoid for small aerial robots," in *Automation, Robotics and Applications (ICARA), 2011 5th International Conference on*, pp. 23 –28, dec. 2011.