Full control of a quadrotor

Author(s):
Bouabdallah, Samir; Siegwart, Roland Y.

Publication Date:
2007

Permanent Link:

https://doi.org/10.3929/ethz-a-010039365 →

Originally published in:

http://doi.org/10.1109/IROS.2007.4399042 →

Full Control of a Quadrotor

Samir Bouabdallah and Roland Siegwart
Autonomous Systems Lab
Swiss Federal Institute of Technology, ETHZ
Zürich, Switzerland
samir.bouabdallah@ieee.org
rsiegwart@ethz.ch

***Abstract*— The research on autonomous miniature flying robots has intensified considerably thanks to the recent growth of civil and military interest in Unmanned Aerial Vehicles (UAV). This paper summarizes the final results of the modeling and control parts of OS4 project, which focused on design and control of a quadrotor. It introduces a simulation model which takes into account the variation of the aerodynamical coefficients due to vehicle motion. The control parameters found with this model are successfully used on the helicopter without re-tuning. The last part of this paper describes the control approach (Integral Backstepping) and the scheme we propose for full control of quadrotors (attitude, altitude and position). Finally, the results of autonomous take-off, hover, landing and collision avoidance are presented.**

I. INTRODUCTION

Flying objects have always exerted a great fascination on man encouraging all kinds of research and development. This project started in 2003, a time at which the robotics community was showing a growing interest in Unmanned Aerial Vehicles (UAV) development. The scientific challenge in UAV design and control in cluttered environments and the lack of existing solutions was very motivating. On the other hand, the broad field of applications in both military and civilian markets was encouraging the funding of UAV related projects. It was decided from the beginning of this project to work on a particular configuration: the quadrotor. The interest comes not only from its dynamics, which represent an attractive control problem, but also from the design issue. Integrating the sensors, actuators and intelligence into a lightweight vertically flying system with a decent operation time is not trivial.

A. State of the Art

The state of the art in quadrotor control has drastically changed in the last few years. The number of projects tackling this problem has considerably and suddenly increased. Most of these projects are based on commercially available toys like the Draganflyer [1], modified afterwards to have more sensory and communication capabilities. Only few groups have tackled the MFR design problem. The thesis [2] lists some of the most important quadrotor projects of the last 10 years. Mesicopter project [3], started in 1999 and ended in 2001. It aimed to study the feasibility of a centimeter scale quadrotor. E. Altuğ presented in his thesis a dual camera visual feedback control [4] in 2003. The group of Prof. Lozano has also a strong activity on quadrotors design and control [5]. N. Guenard from CEA (France) is also working on autonomous control of indoor quadrotors [6]. Starmac, another interesting project, it targets the demonstration of multi agent control of quadrotors of about 1 kg.

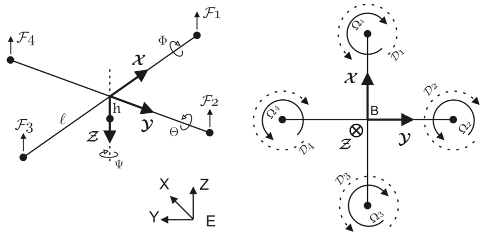


Fig. 1. OS4 coordinate system.

II. SYSTEM MODELING

This simulation model was developed through some successive steps as presented in papers [7], [8]. This last version includes hub forces H, rolling moments R_m and variable aerodynamical coefficients. This makes the model more realistic particularly in forward flight. This section presents the model used for the last version of the OS4 simulator with which the Integral Backstepping (IB) controller was developed. The simulated control parameters were directly used on the real helicopter for successful autonomous flights. Let us consider earth-fixed frame E and a body-fixed frame B as seen in Fig. 1. Using Euler angles parametrization, the airframe orientation in space is given by a rotation R from B to E, where $R \in SO3$ is the rotation matrix.

A. Aerodynamic Forces and Moments

The aerodynamic forces and moments are derived using a combination of momentum and blade element theory [9]. This is based on the work of Gary Fay in Mesicopter project [3]. For an easier reading of the equations below, we recall some symbols. σ: solidity ratio, a: lift slope, μ: advance ratio, λ: inflow ratio, υ: induced velocity, ρ: air density, R_{rad}: rotor radius, l: distance propeller axis-CoG, θ_0: pitch of incidence, θ_{tw}: twist pitch, $\overline{C_d}$: drag coefficient at 70% radial station. See [2] for numerical values. Thrust force T is the resultant of the vertical forces acting on all the blade elements.

$$\begin{cases} T = C_T \rho A (\Omega R_{rad})^2 \\ \frac{C_T}{\sigma a} = (\frac{1}{6} + \frac{1}{4}\mu^2)\theta_0 - (1+\mu^2)\frac{\theta_{tw}}{8} - \frac{1}{4}\lambda \end{cases} \tag{1}$$

Hub force H is the resultant of the horizontal forces acting on all the blade elements.

$$\left\{\begin{array}{l} H = C_H \rho A (\Omega R_{rad})^2 \\ \frac{C_H}{\sigma a} = \frac{1}{4a}\mu \overline{C_d} + \frac{1}{4}\lambda\mu(\theta_0 - \frac{\theta_{tw}}{2}) \end{array}\right. \qquad (2)$$

Drag moment Q is caused by the aerodynamic forces acting on the blade elements. It determines the power required to spin the rotor.

$$\left\{\begin{array}{l} Q = C_Q \rho A (\Omega R_{rad})^2 R_{rad} \\ \frac{C_Q}{\sigma a} = \frac{1}{8a}(1+\mu^2)\overline{C_d} + \lambda(\frac{1}{6}\theta_0 - \frac{1}{8}\theta_{tw} - \frac{1}{4}\lambda) \end{array}\right. \qquad (3)$$

Rolling moment R_m is the integration over the entire rotor of the lift of each section acting at a given radius (not to confuse with the overall rolling moment).

$$\left\{\begin{array}{l} R_m = C_{R_m} \rho A (\Omega R_{rad})^2 R_{rad} \\ \frac{C_{R_m}}{\sigma a} = -\mu(\frac{1}{6}\theta_0 - \frac{1}{8}\theta_{tw} - \frac{1}{8}\lambda) \end{array}\right. \qquad (4)$$

Ground effect is related to a reduction of the induced airflow velocity. The principal need is to find a model of this effect for OS4 to improve the autonomous take-off and landing controllers. A simple way to proceed is to consider that the inflow ratio In Ground Effet (IGE) is $\lambda_{IGE} = (\upsilon_{i,OGE} - \delta\upsilon_i - \dot{z})/\Omega R_{rad}$, where the variation of the induced velocity is $\delta\upsilon_i = \upsilon_i/(4z/R_{rad})^2$. We can then rewrite the thrust coefficient (1) IGE as follows:

$$\left\{\begin{array}{l} T_{IGE} = C_T^{IGE} \rho A (\Omega R_{rad})^2 \\ \frac{C_T^{IGE}}{\sigma a} = \frac{C_T^{OGE}}{\sigma a} + \frac{\delta\upsilon_i}{4\Omega R_{rad}} \end{array}\right. \qquad (5)$$

B. General Moments and Forces

Quadrotor motion is obviously caused by a series of forces and moments coming from different physical effects. This model considers the following ones (with J_r: rotor inertia):

1) Rolling Moments:

body gyro effect	$\dot{\theta}\dot{\psi}(I_{yy} - I_{zz})$
rolling moment due to forward flight	$(-1)^{i+1}\sum_{i=1}^{4} R_{mxi}$
propeller gyro effect	$J_r\dot{\theta}\Omega_r$
hub moment due to sideward flight	$h(\sum_{i=1}^{4} H_{yi})$
roll actuators action	$l(-T_2 + T_4)$

2) Pitching Moments:

body gyro effect	$\dot{\phi}\dot{\psi}(I_{zz} - I_{xx})$
hub moment due to forward flight	$h(\sum_{i=1}^{4} H_{xi})$
propeller gyro effect	$J_r\dot{\phi}\Omega_r$
rolling moment due to sideward flight	$(-1)^{i+1}\sum_{i=1}^{4} R_{myi}$
pitch actuators action	$l(T_1 - T_3)$

3) Yawing Moments:

body gyro effect	$\dot{\theta}\dot{\phi}(I_{xx} - I_{yy})$
hub force unbalance in forward flight	$l(H_{x2} - H_{x4})$
inertial counter-torque	$J_r\dot{\Omega}_r$
hub force unbalance in sideward flight	$l(-H_{y1} + H_{y3})$
counter-torque unbalance	$(-1)^i \sum_{i=1}^{4} Q_i$

4) Rotor Dynamics:

OS4 is equipped with four fixed-pitch rotors, each one includes a BrushLess DC (BLDC) motor, a one-stage gearbox and a propeller. The entire rotor dynamics was identified and validated using the Matlab Identification Toolbox. A first-order transfer function (6) is sufficient to reproduce the dynamics between the propeller's speed set-point and its true speed.

$$G(s) = \frac{0.936}{0.178s + 1} \qquad (6)$$

C. Equations of Motion

The equations of motion are derived from the dynamic model already developed in the author's thesis [2] and all the forces and moments listed in Subsection II-B.

$$\left\{\begin{array}{l} I_{xx}\ddot{\phi} = \dot{\theta}\dot{\psi}(I_{yy} - I_{zz}) + \ldots - h(\sum_{i=1}^{4} H_{yi}) + (-1)^{i+1}\sum_{i=1}^{4} R_{mxi} \\ I_{yy}\ddot{\theta} = \dot{\phi}\dot{\psi}(I_{zz} - I_{xx}) - \ldots + h(\sum_{i=1}^{4} H_{xi}) + (-1)^{i+1}\sum_{i=1}^{4} R_{myi} \\ I_{zz}\ddot{\psi} = \dot{\theta}\dot{\phi}(I_{xx} - I_{yy}) + \ldots + l(H_{x2} - H_{x4}) + l(-H_{y1} + H_{y3}) \\ m\ddot{z} = mg - (c\psi c\phi)\sum_{i=1}^{4} T_i \\ m\ddot{x} = (s\psi s\phi + c\psi s\theta c\phi)\sum_{i=1}^{4} T_i - \sum_{i=1}^{4} H_{xi} \\ m\ddot{y} = (-c\psi s\phi + s\psi s\theta c\phi)\sum_{i=1}^{4} T_i - \sum_{i=1}^{4} H_{yi} \end{array}\right. \qquad (7)$$

III. SYSTEM CONTROL

The model (7) is used to write the system in state-space form $\dot{X} = f(X, U)$ with U inputs vector and X state vector chosen as follows:

$$X = [\phi \quad \dot{\phi} \quad \theta \quad \dot{\theta} \quad \psi \quad \dot{\psi} \quad z \quad \dot{z} \quad x \quad \dot{x} \quad y \quad \dot{y}]^T \qquad (8)$$

$$\begin{array}{l|l} x_1 = \phi & x_7 = z \\ x_2 = \dot{x}_1 = \dot{\phi} & x_8 = \dot{x}_7 = \dot{z} \\ x_3 = \theta & x_9 = x \\ x_4 = \dot{x}_3 = \dot{\theta} & x_{10} = \dot{x}_9 = \dot{x} \\ x_5 = \psi & x_{11} = y \\ x_6 = \dot{x}_5 = \dot{\psi} & x_{12} = \dot{x}_{11} = \dot{y} \end{array} \qquad (9)$$

$$U = [U_1 \quad U_2 \quad U_3 \quad U_4]^T \qquad (10)$$

where the inputs are mapped by:

$$\left\{\begin{array}{l} U_1 = b(\Omega_1^2 + \Omega_2^2 + \Omega_3^2 + \Omega_4^2) \\ U_2 = b(-\Omega_2^2 + \Omega_4^2) \\ U_3 = b(\Omega_1^2 - \Omega_3^2) \\ U_4 = d(-\Omega_1^2 + \Omega_2^2 - \Omega_3^2 + \Omega_4^2) \end{array}\right. \qquad (11)$$

With b thrust coefficient and d drag coefficient. The transformation matrix between the rate of change of the orientation angles $(\dot{\phi}, \dot{\theta}, \dot{\psi})$ and the body angular velocities (p, q, r) can be considered as unity matrix if the perturbations from hover flight are small. Then, one can write $(\dot{\phi}, \dot{\theta}, \dot{\psi}) \approx (p, q, r)$. Simulation tests have shown that this assumption is reasonable. From (7), (8) and (10) we obtain after simplification:

$$f(X,U) = \begin{pmatrix} \dot{\phi} \\ \dot{\theta}\dot{\psi}a_1 + \dot{\theta}a_2\Omega_r + b_1U_2 \\ \dot{\theta} \\ \dot{\phi}\dot{\psi}a_3 - \dot{\phi}a_4\Omega_r + b_2U_3 \\ \dot{\psi} \\ \dot{\theta}\dot{\phi}a_5 + b_3U_4 \\ \dot{z} \\ g - (\cos\phi\cos\theta)\frac{1}{m}U_1 \\ \dot{x} \\ u_x\frac{1}{m}U_1 \\ \dot{y} \\ u_y\frac{1}{m}U_1 \end{pmatrix} \tag{12}$$

With:

$$\begin{array}{l|l} a_1 = (I_{yy} - I_{zz})/I_{xx} & b_1 = l/I_{xx} \\ a_2 = J_r/I_{xx} & b_2 = l/I_{yy} \\ a_3 = (I_{zz} - I_{xx})/I_{yy} & b_3 = 1/I_{zz} \\ a_4 = J_r/I_{yy} & \\ a_5 = (I_{xx} - I_{yy})/I_{zz} & \end{array} \tag{13}$$

$$\begin{aligned} u_x &= (\cos\phi\sin\theta\cos\psi + \sin\phi\sin\psi) \\ u_y &= (\cos\phi\sin\theta\sin\psi - \sin\phi\cos\psi) \end{aligned} \tag{14}$$

It is worthwhile to note in the latter system that the angles and their time derivatives do not depend on translation components. On the other hand, the translations depend on the angles. One can ideally imagine the overall system described by (12) as constituted of two subsystems, the angular rotations and the linear translations.

A. *Control Approach Selection*

During the OS4 project we explored several control approaches from theoretical development to final experiments. As a first attempt, we tested on OS4 two linear controllers, a PID and an LQR based on a simplified model. The main result was an autonomous hover flight presented in [7]. However, strong disturbances were poorly rejected as in presence of wind. In the second attempt we reinforced the control using backstepping techniques. This time, we were able to elegantly reject strong disturbances but the stabilization in hover flight was delicate (see [8]). Another improvement is now introduced thanks to integral backstepping (IB) [10]. The idea of using integral action in the backstepping design was first proposed in [11] and applied in [12] from which this control design was derived. After the evaluation of all the control approaches tested in this work, it became clear that the way to follow is a combination between PID and Backstepping into the so-called Integral Backstepping. In fact, the backstepping is well suited for the cascaded structure of the quadrotor dynamics. Moreover, the controller design process can be straight forward if done properly. In addition, this method guarantees asymptotic stability and has robustness to some uncertainties, while the integral action cancels the steady state errors. After a phase of extensive simulation and experimentation, Integral Backstepping was proposed as a single approach for attitude, altitude and position control. This process is presented in the next section. Thanks to this technique, OS4 is able to perform autonomous hovering with altitude control and autonomous take-off and landing. The derivation is similar for attitude, altitude and position controllers. So, only roll angle controller derivation will be presented. The OS4 control system is structured in six different controllers as illustrated in Fig. 2. Take-off and landing controller outputs the desired altitude (z_d) to altitude controller which outputs the desired overall thrust (T_d) based on sonar data. Position controller receives OS4 position (x, y) and desired thrust, it outputs desired roll (ϕ_d) and pitch (θ_d) while desired yaw (ψ_d) comes directly from the user. Attitude controller outputs then the desired motor speed to the motor controllers. Integral backstepping technique is used for attitude, altitude and position control. This permits a powerful and flexible control structure.

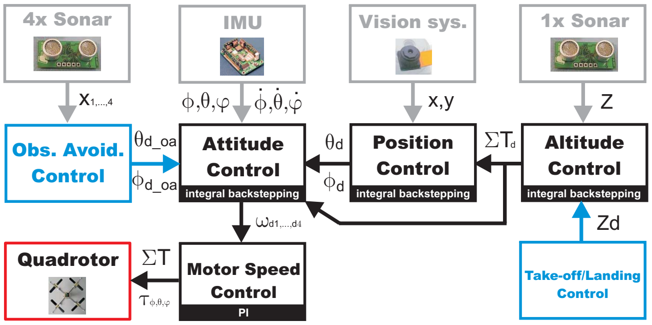


Fig. 2. The control structure implemented on OS4.

B. *Attitude Control*

Attitude control is the heart of the control system, it keeps the 3D orientation of the helicopter to the desired value. Attitude control loop runs at 76 Hz which is the update rate of the IMU (Microstrain 3DM-GX1). The first step in IB control design is to consider the tracking-error $e_1 = \phi_d - \phi$ and its dynamics:

$$\frac{de_1}{dt} = \dot{\phi}_d - \omega_x \tag{15}$$

The angular speed ω_x is not our control input and has its own dynamics. So, we set for it a desired behavior and we consider it as our virtual control:

$$\omega_{xd} = c_1e_1 + \dot{\phi}_d + \lambda_1\chi_1 \tag{16}$$

with c_1 and λ_1 positive constants and $\chi_1 = \int_0^t e_1(\tau)d_\tau$ the integral of roll tracking error. So, the integral term is now introduced in (16).

Since ω_x has its own error e_2, we compute its dynamics using (16) as follows:

$$\frac{de_2}{dt} = c_1(\dot{\phi}_d - \omega_x) + \ddot{\phi}_d + \lambda_1e_1 - \ddot{\phi} \tag{17}$$

where e_2, the angular velocity tracking error is defined by:

$$e_2 = \omega_{xd} - \omega_x \tag{18}$$

Using (16) and (18) we rewrite roll tracking error dynamics as:

$$\frac{de_1}{dt} = -c_1 e_1 - \lambda\chi_1 + e_2 \tag{19}$$

By replacing $\ddot{\phi}$ in (17) by its corresponding expression from model (12), the control input U_2 appears in (20):

$$\frac{de_2}{dt} = c_1(\dot{\phi}_d - \omega_x) + \ddot{\phi}_d + \lambda_1 e_1 - \dot{\theta}\dot{\psi}a_1 - \dot{\theta}a_2\Omega_r - b_1 U_2 \tag{20}$$

The real control input has appeared in (20). So, using equations (15), (19) and (20) we combine the tracking errors of the position e_1, of the angular speed e_2 and of the integral position tracking error χ_1 to obtain:

$$\frac{de_2}{dt} = c_1(-c_1 e_1 - \lambda_1\chi_1 + e_2) + \ddot{\phi}_d + \lambda_1 e_1 - \tau_x/I_{xx} \tag{21}$$

where τ_x is the overall rolling torque. The desirable dynamics for the angular speed tracking error is:

$$\frac{de_2}{dt} = -c_2 e_2 - e_1 \tag{22}$$

This is obtained if one chooses the control input U_2 as:

$$\begin{aligned} U_2 &= +\frac{1}{b_1}[(1 - c_1^2 + \lambda_1)e_1 + (c_1 + c_2)e_2 \\ &\quad -c_1\lambda_1\chi_1 + \ddot{\phi}_d - \dot{\theta}\dot{\psi}a_1 - \dot{\theta}a_2\Omega_r] \end{aligned} \tag{23}$$

where c_2 is a positive constant which determines the convergence speed of the angular speed loop. Similarly, pitch and yaw controls are:

$$\begin{aligned} U_3 &= +\frac{1}{b_2}[(1 - c_3^2 + \lambda_2)e_3 + (c_3 + c_4)e_4 \\ &\quad -c_3\lambda_2\chi_2 + \ddot{\theta}_d - \dot{\phi}\dot{\psi}a_3 + \dot{\phi}a_4\Omega_r] \end{aligned} \tag{24}$$

$$U_4 = +\frac{1}{b_3}[(1 - c_5^2 + \lambda_3)e_5 + (c_5 + c_6)e_6 - c_5\lambda_3\chi_3] \tag{25}$$

with $(c_3, c_4, c_5, c_6, \lambda_2, \lambda_3) > 0$, and (χ_2, χ_3) the integral position tracking error of pitch and yaw angles respectively.

1) Results: Attitude control performance is of crucial importance, it is directly linked to the performance of the actuators. OS4 is equipped with motors powerful enough to avoid amplitude saturation. However, they suffer from low dynamics and thus from bandwidth saturation. The experiment shown in Fig. 4 is a free flight were attitude references are zero. One can see in Fig. 3 that roll and pitch plots show a bounded oscillation of $0.1\,rad$ in amplitude. This oscillation is not perceptible in flight, nevertheless it is due to the slow dynamics of OS4's actuators coupled with the differences between the four propulsion groups. Control parameters were in this experiment $C_1 = 10.5$, $C_2 = 2$, $C_3 = 10$, $C_4 = 2$, $C_5 = 2$, $C_6 = 2$. These are really close to the parameters used in simulation which highlights the quality of the model.

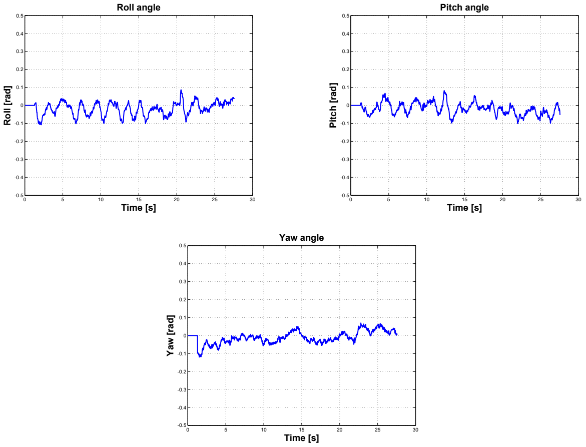


Fig. 3. Experiment: Integral backstepping attitude controller has to maintain attitude angles to zero in flight. The helicopter is stabilized despite the numerous disturbances due to yaw drift, sensors noise and unmodeled effects.

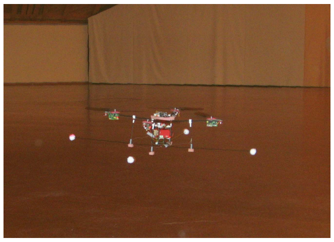


Fig. 4. OS4 in hover. A training frame was added for safety.

C. Altitude Control

The altitude controller keeps the distance of the helicopter to the ground at a desired value. It is based on a sonar (Devantech SRF10) which gives the range to the closest obstacle at 15 Hz. On the control law side, altitude tracking error is defined as (ground effect is neglected):

$$e_7 = z_d - z \tag{26}$$

The speed tracking error is:

$$e_8 = c_7 e_7 + \dot{z}_d + \lambda_4\chi_4 - \dot{z} \tag{27}$$

The control law is then:

$$U_1 = \frac{m}{cos\phi cos\theta}[g + (1 - c_7^2 + \lambda_4)e_7 + (c_7 + c_8)e_8 - c_7\lambda_4\chi_4] \tag{28}$$

where (c_7, c_8, λ_4) are positive constants.

1) Take-off and Landing: The autonomous take-off and landing algorithm adapts the altitude reference z_d to follow the dynamics of the quadrotor for taking-off or landing. The desired altitude reference is gradually reduced by a fixed step k ($k > 0$) which depends on the vehicle dynamics and the desired landing speed. Moreover, the fact that the control loop is much faster than the vehicle dynamics, makes the landing very smooth.

2) Results: Altitude control works surprisingly well despite all the limitations of the sonar. Figure 5 shows an altitude reference profile (green) followed by the simulated controller (red) and the real controller (blue). The task was

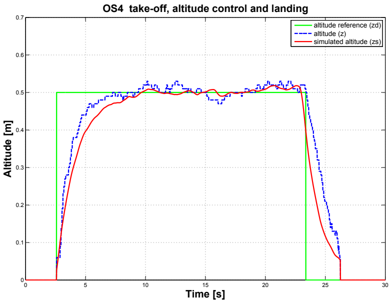


Fig. 5. Autonomous take-off, altitude control and landing in simulation and in real flight.

to climb to $0.5\,m$, hover and then land. Control parameters where $C_7 = 3.5$, $C_8 = 1.5$ in simulation and $C_7 = 4$, $C_8 = 2$ in experiment. The slight deviation between simulation and reality in take-off and landing phases is inherited from actuators' dynamics where the model was slightly slower in the raising edge, and slightly faster in the falling one. Take-off is performed in 2 s (0-0.5 m) and landing in 2.8 s (0.5-0 m). Altitude control has a maximum of 3 cm deviation from the reference.

D. Position Control

Position control keeps the helicopter over the desired point. It is meant here the (x, y) horizontal position with regard to a starting point. Horizontal motion is achieved by orienting the thrust vector towards the desired direction of motion. This is done by rotating the vehicle itself in the case of a quadrotor. In practice, one performs position control by rolling or pitching the helicopter in response to a deviation from the y_d or x_d references respectively. Thus, the position controller outputs the attitude references ϕ_d and θ_d, which are tracked by the attitude controller (see Fig. 2). The thrust vector orientation in the earth fixed frame is given by R, the rotation matrix. Applying small angle approximation to R gives:

$$R = \begin{bmatrix} 1 & \psi & \theta \\ \psi & 1 & -\phi \\ -\theta & \phi & 1 \end{bmatrix} \tag{29}$$

From (12) and using (29) one can simplify horizontal motion dynamics to

$$\begin{bmatrix} m\ddot{x} \\ m\ddot{y} \end{bmatrix} = \begin{bmatrix} -\theta U_1 \\ \phi U_1 \end{bmatrix} \tag{30}$$

The control law is then derived using IB technique. Position tracking errors for x and y are defined as:

$$\begin{cases} e_9 = x_d - x \\ e_{11} = y_d - y \end{cases} \tag{31}$$

Accordingly speed tracking errors are:

$$\begin{cases} e_{10} = c_9 e_9 + \dot{x}_d + \lambda_5 \chi_5 - \dot{x} \\ e_{12} = c_{11} e_{11} + \dot{y}_d + \lambda_6 \chi_6 - \dot{y} \end{cases} \tag{32}$$

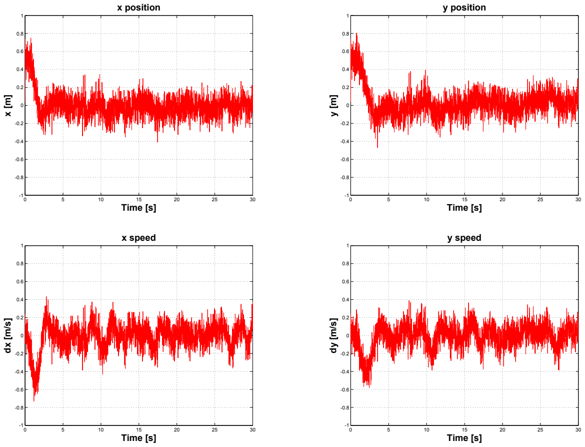


Fig. 6. Simulation: Integral backstepping position controller drives attitude controller in order to maintain the helicopter over a given point.

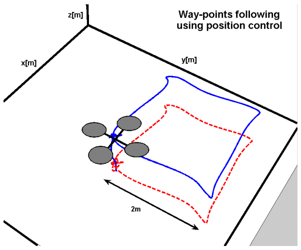


Fig. 7. Four way-points for a square trajectory tracked by OS4.

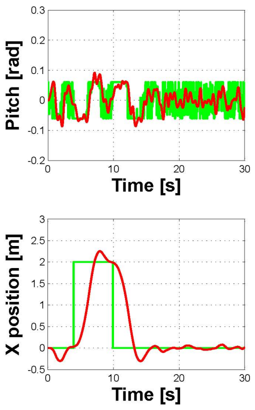
The control laws are then:

$$\begin{cases} U_x = \frac{m}{U_1}[(1 - c_9^2 + \lambda_5)e_9 + (c_9 + c_{10})e_{10} - c_9\lambda_5\chi_5] \\ U_y = -\frac{m}{U_1}[(1 - c_{11}^2 + \lambda_6)e_{11} + (c_{11} + c_{12})e_{12} - c_{11}\lambda_6\chi_6] \end{cases} \tag{33}$$

where $(c_9, c_{10}, c_{11}, c_{12}, \lambda_5, \lambda_6)$ are positive constants.

1) Results: The main result in position control was obtained in simulation. Fig. 2 shows how the different controllers are cascaded. In fact, only the attitude is driven by the position, altitude controller is simply feeding them with U_1. Attitude and position loops run at 76 Hz and 25 Hz respectively. This spectral separation is necessary to avoid a conflict between the two loops; it is often accompanied with gain reductions in the driving loop. Control parameters were $C_9 = 2, C_{10} = 0.5, C_{11} = 2, C_{12} = 0.5$ in the simulation of Fig. 6.

2) Way-points following: The planner block in the simulator defines the way-points and hence the trajectories OS4 has to follow. The position of the next way-point is sent to position controller which directs the vehicle towards the goal. A way-point is declared reached when the helicopter enters a sphere around this point. The radius of this sphere (0.1 m) is the maximum admitted error. Figure 7 shows a square trajectory defined by four way-points. The task was to climb to 1 m from the ground and then follow the four way-points of a square of 2 m side. In order to track the square trajectory, the planner generates the (x_d, y_d) position references, and consequently the position controller generates



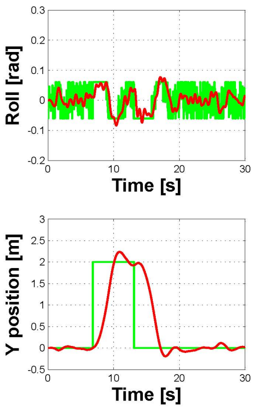


Fig. 8. Simulation: The position and attitude signals generated to track the square trajectory.

the (ϕ_d, θ_d) attitude references for every cycle. Figure 8 depicts these signals and shows that the 2 m side square is tracked with about 10% overshoot (20 cm), while the trajectory is completed in 20 s.

E. Obstacle Avoidance

OS4 is equipped with a sonar-based obstacle avoidance system composed of four miniature ultrasound range finders in cross configuration. Several algorithms were simulated with various results presented in details in [13]. The lack of precise sensors for linear speed made the implementation of this approach difficult. A simple collision avoidance algorithm was then developed. The idea was to avoid collision with walls or persons present in the flight area. The inherent noise of the sonar especially in absence of obstacles was threatening OS4 stability. This is mainly due to the interferences between the five sonar and the effect of the propellers on the ultrasound waves.

1) Results: A collision avoidance behavior was practically obtained after numerous tests and tuning. Once the obstacle is detected, a pitch reference is given to fly away the helicopter from the obstacle. Figure 9 shows the reaction of OS4 to an obstacle at 40 cm, one can see the distance to the obstacle increasing until the latter disappears, then OS4 recovers a normal flight.

IV. CONCLUSIONS AND FUTURE WORKS

This paper presented the final developments in OS4 project. A quadrotor simulation model was introduced. It includes major aerodynamical effects modeled with blade element and momentum theory. In addition, the actuator's model was identified and all sensor delays and noises were taken into account. Real experiments were conducted with the same control parameters tuned in simulation. A control approach was proposed it permitted the design of the main controllers: Attitude, altitude, position and the auxiliary ones: Take-off and landing, obstacle avoidance and way-point following. The latter was demonstrated in simulation. The experiment has shown that OS4 is currently able to take-off, hover, land and avoid collisions automatically thanks to model-based control. The future work is to firstly enhance the propulsion group towards more reliability and high bandwidth. Secondly, it is necessary to improve the vision sensor in order to get rid of the external pattern. Thirdly, it would be very interesting to practically test a way-points following maneuver with obstacle avoidance capability. OS4 is undoubtedly one of the most integrated quadrotor ever designed. As far as we know, it is the first one practically capable of a collision avoidance maneuver.

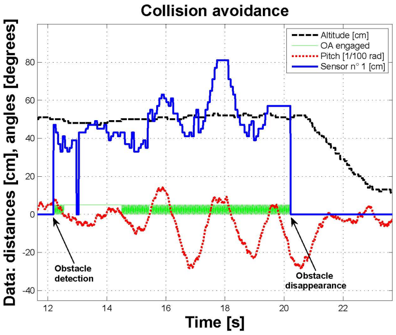


Fig. 9. Experiment: Collision avoidance with OS4. The helicopter flies back until the obstacle disappears.

V. ACKNOWLEDGMENTS

The authors gratefully acknowledge all the people that contributed directly or indirectly to this work.

REFERENCES

[1] RCtoys, http://www.rctoys.com/.

[2] S. Bouabdallah, "Design and control of quadrotors with application to autonomous flying," Ph.D. dissertation, EPFL, 2006.

[3] G. Fay, "Derivation of the aerodynamic forces for the mesicopter simulation," Stanford University, USA, 2001.

[4] E. Altuğ, "Vision based control of unmanned aerial vehicles with applications to an autonomous four rotor helicopter, quadrotor," Ph.D. dissertation, University of Pennsylvania, 2003.

[5] P. Castillo *et al.*, *Modelling and Control of Mini-Flying Machines*. Springer.

[6] N. Guenard *et al.*, "Control laws for the tele operation of an unmanned aerial vehicle known as an x4-flyer," in *Proc. (IEEE) International Conference on Intelligent Robots (IROS'06)*, Beijing, China, 2006.

[7] S. Bouabdallah *et al.*, "Design and control of an indoor micro quadrotor," in *Proc. (IEEE) International Conference on Robotics and Automation (ICRA'04)*, New Orleans, USA, 2004.

[8] S. Bouabdallah and R. Siegwart, "Backstepping and sliding-mode techniques applied to an indoor micro quadrotor," in *Proc. (IEEE) International Conference on Robotics and Automation (ICRA'05)*, Barcelona, Spain, 2005.

[9] J. G. Leishman, *Principles of Helicopter Aerodynamics*. Cambridge University Press.

[10] M. Krstić *et al.*, *Nonlinear and Adaptive Control Design*. New York, USA: Wiley Interscience, 1995.

[11] I. Kanellakopoulos and P. Krein, "Integral-action nonlinear control of induction motors," in *Proc. of the 12th IFAC World Congress*, Sydney, Australia, 1993.

[12] Y. Tan *et al.*, "Advanced nonlinear control strategy for motion control systems," in *Proc. of (IEEE) Power Electronics and Motion Control Conference, (PIEMC'00)*, Beijing, China, 2000.

[13] S. Bouabdallah *et al.*, "Toward obstacle avoidance on quadrotors," in *Proc. of the XII International Symposium on Dynamic Problems of Mechanics (DYNAME'07)*, Ilhabela, Brazil, Feb. 2006.