Chapter 6: Design and Control of a Miniature Quadrotor[1]

This Chapter presents a method for designing and controlling a miniature quadrotor. It starts by presenting an accurate quadrotor simulation model that is used for initial testing and validation via a simulator built using *MATLAB*. A design methodology is then discussed along with an iterative algorithm derived to choose quadrotor components based on design constraints. Testing using several control techniques follows the design phase; the result is adoption of the integral backstepping techniques that achieves superior quadrotor performance. Stability analysis and obstacle avoidance are also discussed. Simulation and experimental results are included to demonstrate effectiveness of the design.

6.1 Introduction

Miniature aerial vehicles (MAVs) have attracted major research interest during the last decade. Recent advances in low power processors, miniature sensors and control theory have contributed to system miniaturization and creation of new application fields.

Miniature flying robots (MFRs) offer major advantages when used for aerial surveillance in complex or cluttered environments like office buildings and commercial centers. MFRs may assist in search and rescue missions after earthquakes, explosions and other natural disasters.

As such, an MFR that is capable of flying in narrow spaces and fit through small openings and collapsed buildings, maneuver around pillars and destroyed wall structures could quickly and systematically search for accident victims or inspect disaster areas without risking human lives. An MFR could, after localization, provide coordinates of potential victims to rescuers guiding and assisting them to complete their mission.

In such hazardous environments, wireless communication is difficult if not impossible because of natural obstacles. Aerial relay could be possible

[1] Written by S. Bouabdallah, R. Siegwart

using unmanned aerial vehicles equipped with transceivers, ensuring constant communication.

Reviewing closely related research, the development of a miniature autonomous flight control system and the creation of a multi-vehicle platform for experimentation and validation of multi-agent control algorithms are discussed in [11], while [14] presents results in centimeter-scale quadrotor design and analysis. A recent result reported in [18] is a 13.6 cm micro-helicopter capable of hovering for 3 minutes.

The Autonomous Systems Lab (ASL) of the Swiss Federal Institute of Technology is participating in similar research efforts [1] concentrating in MFRs. The major focus is on fully autonomous MFRs that may be developed through coordinated efforts in system-level optimization, design and control.

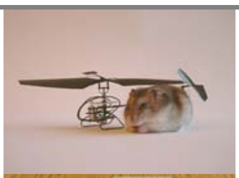
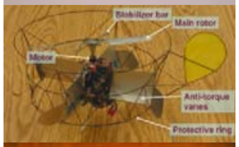
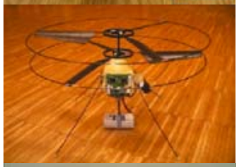
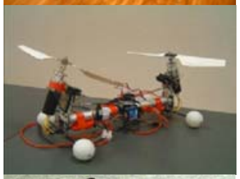
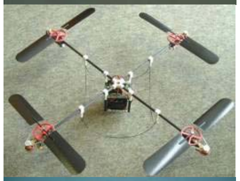
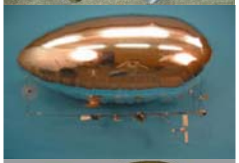
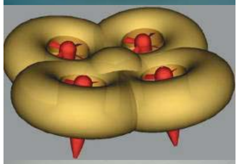
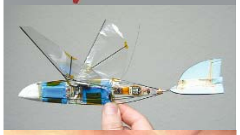
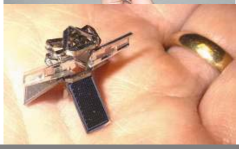
The approach that is followed is to design vehicles with optimized mechanics and innovative control techniques. The goal is to keep on miniaturizing the MFR in every redesign step following the latest technological advancements.

The objective of the ASL MFR project is to optimally design and control aerial systems for navigation in cluttered environments. A quadrotor (OS4) and a coaxial (CoaX) vertical take-off and landing (VTOL) systems are considered because of their challenging control problems and their broad field of applications. This is accomplished via a detailed design methodology presented below, based on which the best components are chosen following imposed constraints.

As widely known, when compared with other aerial vehicles, VTOL vehicle systems have specific characteristics like flying in very low altitudes and being able to hover that make them suitable for applications that may be impossible to complete using fixed-wing vehicles.

Different configurations of commonly used MAVs for research purposes and in industry are shown in Table 6.1 along with related advantages and drawbacks. Table 6.1 offers a pictorial comparison that may be used when a new design is proposed.

Configuration	Picture	Advantages	Drawbacks
Fixed-wing (AeroVironment)		- Simple mechanics - Silent operation	- No hovering

Single (A.V de Rostyne)		- Good controllability and maneuverability	- Complex mechanics - Large rotor - Long tail boom
Axial rotor (Maryland Univ.)	 Stabilizer bar Main rotor Motor Anti-torque vanes Protective ring	- Compactness - Simple mechanics	- Complex control - Weak maneuvera-bility
Coaxial rotors (ETHZ)		- Compactness - Simple mechanics	- Complex aerodynamics
Tandem rotors (Heudiasyc)		- Good controllability and maneuverability - No aerodynamics Interference	- Complex mechanics - Large size
Quadrotors (ETHZ)		- Good maneuverability - Simple mechanics - Increased payload	- High energy consumption - Large size
Blimp (EPFL)		- Low power consumption - Auto-lift	- Large size - Weak maneuverability
Hybrid (MIT)		- Good maneuverability - Good survivability	- Large size - Complex design
Bird-like (Caltech)		- Good maneuverability - Low power Consumption	- Complex mechanics - Complex control
Insect-like (UC Berkeley)		- Good maneuverability - Compactness	- Complex mechanics - Complex control

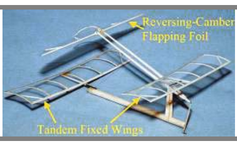
Fish-like (US Naval Lab)		- Multimode mobility - Efficient Aerodynamics	- Complex control - Weak maneuverability

Table 6.1. Common UAV-MAV configurations.

Further, Table 6.2 presents a short and not exhaustive comparison between different VTOL vehicle concepts. It may be observed from Table 6.2 that the quadrotor and the coaxial helicopter are among the best configurations if used as MFRs.

	A	B	C	D	E	F	G	H
Power cost	2	2	2	2	1	4	3	3
Control cost	1	1	4	2	3	3	2	1
Payload/volume	2	2	4	3	3	1	2	1
Maneuverability	4	2	2	3	3	1	3	3
Mechanics simplicity	1	3	3	1	4	4	1	1
Aerodynamics complexity	1	1	1	1	4	3	1	1
Low speed flight	4	3	4	3	4	4	2	2
High speed flight	2	4	1	2	3	1	3	3
Miniaturization	2	3	4	2	3	1	2	4
Survivability	1	3	3	1	1	3	2	3
Stationary flight	4	4	4	4	4	3	1	2
Total	**24**	**28**	**32**	**24**	**33**	**28**	**22**	**24**

A=Single rotor, B=Axial rotor, C=Coaxial rotors, D=Tandem rotors, E=Quadrotor, F=Blimp, G=Bird-like, H=Insect-like.

Table 6.2. VTOL concept comparison (1=Bad, 4=Very good).

6.2 Modeling for Simulation

The OS4 quadrotor simulation model was developed through several successive steps as presented in [2]. Major improvements of the most recent version include hub forces (*H*), rolling moments (R_m) and variable aerodynamic coefficients. This makes the model more realistic, particularly in forward flight. Previous model versions required often to slightly adjust control parameters for successful experimentation with the OS4 testbed [3]. The current version of the model used in the OS4 simulator incorporates a newly designed Integral Backstepping (IB) controller where simulated control parameters may be directly used on the real OS4 helicopter to demonstrate successful autonomous flights.

The dynamics of a rigid body subject to external forces applied to the center of mass and expressed in the body-fixed reference frame [17] in Newton-Euler formalism are:

$$\begin{bmatrix} mI_{3\times3} & 0 \\ 0 & I \end{bmatrix}\begin{bmatrix} \dot{V} \\ \dot{\omega} \end{bmatrix} + \begin{bmatrix} \omega \times mV \\ \omega \times I\omega \end{bmatrix} = \begin{bmatrix} F \\ \tau \end{bmatrix} \tag{6.1}$$

where, $I \in \Re^{(3\times3)}$ is the inertia matrix, V is the body linear velocity and ω is the body angular velocity. F and τ are, respectively, the body force and torque, while m is the system mass. Consider the earth-fixed frame E and the body-fixed frame B as seen in Figure 6.1. Using Euler angle parameterization, the airframe orientation in space is given by a rotation R from B to E, where R in $SO3$ is the rotation matrix. The frame system is slightly different compared to previous versions in order to conform with the N, E, D (North, East, Down) standard.

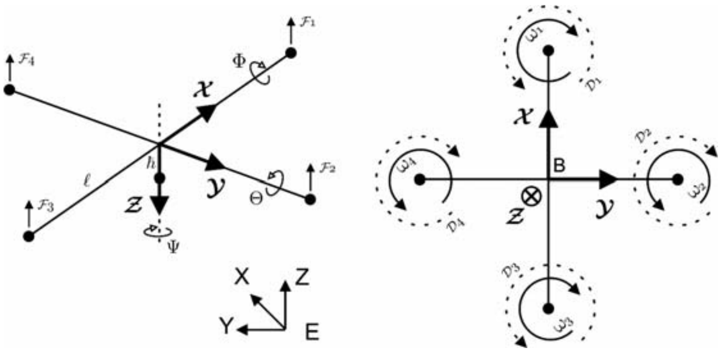


Fig. 6.1. The OS4 coordinate system.

6.2.1 Aerodynamic Forces and Moments

The aerodynamic forces and moments are derived using a combination of momentum and blade element theory [15] based on previously reported research [8]. Equations use, among others, the abbreviated symbols shown next:

Air density	ρ	Rotor radius	R
Solidity ratio	σ	Rotor speed	Ω
Lift slope	a	Rotor area	A
Rotor advance ratio	μ	Pitch of incidence	θ_0
Inflow ratio	λ	Twist pitch	θ_{tw}
Induced velocity	υ	Average drag coefficient	$\bar{C}_d$

Thrust Force

Thrust force is the result of vertical forces acting on all blade elements of one propeller calculated as:

$$\begin{cases} T = C_T \rho A (\Omega R_{rad})^2 \\ \dfrac{C_T}{\sigma a} = (\dfrac{1}{6} + \dfrac{1}{4}\mu^2)\theta_0 - (1+\mu^2)\dfrac{\theta_{tw}}{8} - \dfrac{1}{4}\lambda \end{cases} \tag{6.2}$$

Hub Force

The hub force is the result of horizontal forces acting on all blade elements defined as:

$$\begin{cases} H = C_H \rho A (\Omega R_{rad})^2 \\ \dfrac{C_H}{\sigma a} = \dfrac{1}{4a}\mu \overline{C_d} + \dfrac{1}{4}\lambda\mu(\theta_0 - \dfrac{\theta_{tw}}{2}) \end{cases} \tag{6.3}$$

Drag Moment

Drag moment about the rotor shaft is caused by aerodynamic forces acting on blade elements. Horizontal forces act on the rotor, multiplied by the moment arm and integrated over the rotor. Drag moment is important because it determines the power required to spin the rotor calculated as:

$$\begin{cases} Q = C_Q \rho A (\Omega R_{rad})^2 R_{rad} \\ \dfrac{C_Q}{\sigma a} = \dfrac{1}{8a}(1+\mu^2)\overline{C_d} + \lambda(\dfrac{1}{6}\theta_0 - \dfrac{1}{8}\theta_{tw} - \dfrac{1}{4}\lambda) \end{cases} \tag{6.4}$$

Rolling Moment

The rolling moment of a propeller is present in forward flight when the advancing blade produces more lift than the retreating one. It is defined in terms of integration over the entire rotor of the lift of each section acting at a given radius – it is not to be confused with the overall rolling moment that is caused by a number of other effects. The rolling moment is:

$$\begin{cases} R_m = C_{R_m} \rho A (\Omega R_{rad})^2 R_{rad} \\ \dfrac{C_{R_m}}{\sigma a} = -\mu (\dfrac{1}{6}\theta_0 - \dfrac{1}{8}\theta_{tw} - \dfrac{1}{8}\lambda) \end{cases} \tag{6.5}$$

Ground Effect

Helicopters operating near the ground are subject to thrust augmentation due to better rotor efficiency. This is related to reduction of the induced airflow velocity called In Ground Effect (IGE). One may find in the literature different approaches to deal with this effect, for instance by using adaptive techniques [10]. In this research, the goal is to find a model for IGE that improves autonomous take-off and landing. As such, the idea is to obtain a simple model capturing mainly the variation of the induced inflow velocity. The images method discussed in [15] is used in [5] to state that at constant power $T_{OGE}\, \upsilon_{i.OGE} = T_{IGE}\, \upsilon_{i.IGE}$. The velocity induced at the rotor center by its image is $\delta\upsilon_i = A\upsilon_i / 16\pi z^2$. The simple equation shown below was obtained in [5] by assuming that υ and $\delta\upsilon_i$ are constant over the disk allowing for $\upsilon_{i,IGE} = \upsilon_i - \delta\upsilon_i$, with z being the altitude:

$$\frac{T_{IGE}}{T_{OGE}} = \frac{1}{1 - \dfrac{R_{rad}^2}{16z^2}} \tag{6.6}$$

Alternatively, one may consider that the inflow ratio IGE is $\lambda_{IGE} = (\upsilon_{i,OGE} - \delta\upsilon_i - \dot{z}) / \Omega R_{rad}$, where the variation of the induced velocity is $\delta\upsilon_i = \upsilon_i / (4z / R_{rad})^2$. Then, from the previous equations, the thrust coefficient IGE may be re-written as follows:

$$\begin{cases} T_{IGE} = C_T^{IGE} \rho A (\Omega R_{rad})^2 \\ \dfrac{C_T^{IGE}}{\sigma a} = \dfrac{C_T^{OGE}}{\sigma a} + \dfrac{\delta\upsilon_i}{4\Omega R_{rad}} \end{cases} \tag{6.7}$$

The variation of the inflow velocity in and out of the ground effect was also calculated using the OS4 simulator. As such, Figure 6.2 illustrates the plot of the ratio ($\delta\upsilon_i / \upsilon_i$) obtained by simulation and analytically. The influence is minimal, very close to zero when $z / R_{rad} \approx 2$ but it becomes important for values of $z / R_{rad} \leq 1$. It looks like in the case of a quadrotor the ground effect influence is already present at one rotor diameter and becomes important at one rotor radius.

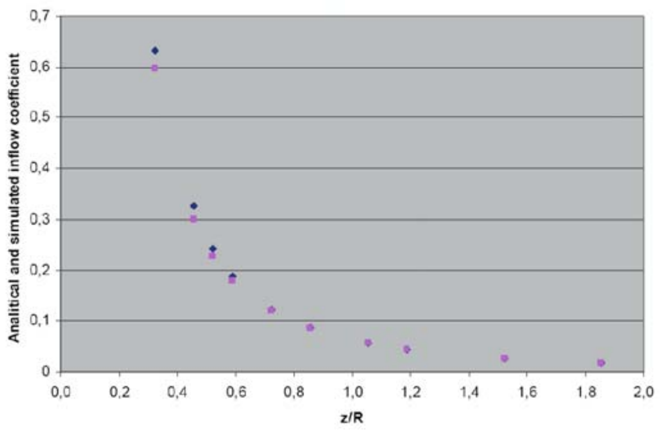


Fig. 6.2. Ground effect influence on the inflow velocity.

In order to empirically verify this assumption, a simple experiment was conducted, which proved that a quadrotor deprived of altitude control can hover at a constant altitude at nearly one rotor diameter from the ground. It should be clear that this result is only an indication of validity and it does not constitute a formal proof.

6.2.2 Moments and Forces

Quadrotor motion is caused by a series of forces and moments. Defining the roll angle as φ, the pitch angle as θ and the yaw angle as ψ, the rotor inertia as J_r, the vertical distance prop./center of gravity as h and the horizontal distance prop./center of gravity as l, the following are defined with respect to rolling, pitching and yawing moments:

Rolling Moments

Body gyro effect:	$\dot{\theta}\dot{\psi}(I_{yy}-I_{zz})$
Propeller gyro effect:	$J_r\dot{\theta}\Omega_r$
Roll actuators action:	$l(-T_2+T_4)$
Hub moment due to sideward flight:	$h\left(\sum_{i=1}^{4}H_{yi}\right)$
Rolling moment due to forward flight:	$(-1)^{i+1}\sum_{i=1}^{4}R_{mxi}$

Pitching Moments

Body gyro effect:	$\dot{\theta}\dot{\psi}(I_{zz}-I_{xx})$
Propeller gyro effect:	$J_r\dot{\theta}\Omega_r$
Roll actuators action:	$l(T_1-T_3)$
Hub moment due to forward flight:	$h\left(\sum_{i=1}^{4}H_{xi}\right)$
Rolling moment due to sideward flight:	$(-1)^{i}\sum_{i=1}^{4}R_{myi}$

Yawing Moments

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

The forces along the *x*, *y* and *z* axes are defined, respectively as:

Forces along the z- axis

Body gyro effect:	$\cos\psi\cos\phi\left(\sum_{i=1}^{4}T_i\right)$
Weight:	mg

Forces along the x- axis

Actuators action:	$(\sin\psi\sin\phi+\cos\psi\sin\theta\cos\phi)\left(\sum_{i=1}^{4}T_i\right)$
Hub force in y axis:	$\left(-\sum_{i=1}^{4}H_{xi}\right)$
Friction:	$\frac{1}{2}C_x A_c \rho \dot{x}\,\lvert\dot{x}\rvert$

Forces along the y- axis

Actuators action:	$(-\cos\psi\sin\phi+\sin\psi\sin\theta\cos\phi)\left(\sum_{i=1}^{4}T_i\right)$
Hub force in y axis:	$\left(-\sum_{i=1}^{4}H_{yi}\right)$
Friction:	$\frac{1}{2}C_y A_c \rho \dot{y}\,\lvert\dot{y}\rvert$

6.2.3 Equations of Motion

The equations of motion may be derived from (6.1) and the previously stated equations of forces and moments acting on the quadrotor as:

$$\begin{cases} \ddot{\phi} = \sum \tau_x / I_{xx} \\ \ddot{\theta} = \sum \tau_y / I_{yy} \\ \ddot{\psi} = \sum \tau_z / I_{zz} \\ \ddot{z} = g - \sum F_z / m \\ \ddot{x} = \sum F_x / m \\ \ddot{y} = \sum F_y / m \end{cases} \tag{6.8}$$

6.2.4 Rotor Dynamics

The OS4 helicopter is equipped with four fixed-pitch rotors (no swash plate) each including a BrushLess Direct Current (BLDC) motor, a one-stage gearbox and a propeller. The entire rotor dynamics have been identified and validated using the *MATLAB* Identification Toolbox. A first-order transfer function is reasonable to reproduce the dynamics between the propeller's speed set-point and its true speed. It was set to be:

$$G(s) = \frac{0.936}{0.178s + 1} \tag{6.9}$$

It is essential to mention the non-unity gain in (6.9) that is shown in Figure 6.3. Figure 6.3 shows the model output (actuator's output) in red and the sensor data (measured propeller speed) in blue to a step input in green. Since sensorless BLDC motors require a minimum speed to run, the set point does not start from zero.

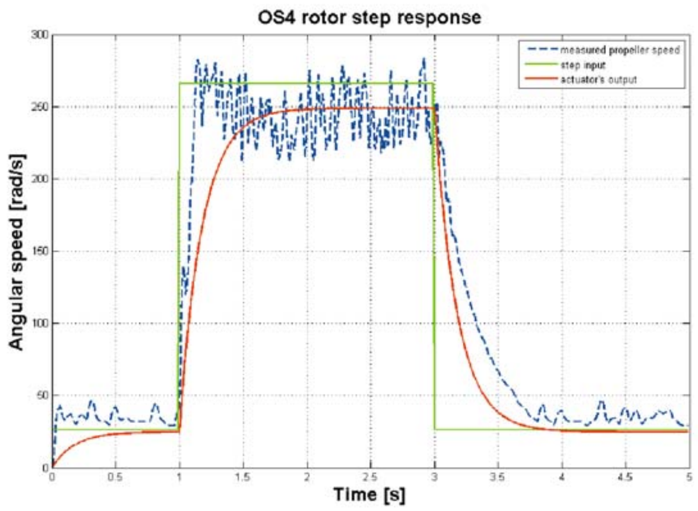


Fig. 6.3. Rotor and model step response, measured at the propeller shaft.

6.3 The OS4 Simulator

The most recent version of the OS4 simulator includes the identified actuator's dynamics, the aerodynamics block, obstacle avoidance controller (OAC) and a high level planner for waypoint definition. Each block is described by one or more *MATLAB* files; the OS4 simulator may be easily incorporated into other simulators. A block diagram of the simulator is depicted in Figure 6.4. Simulation starts by defining the initial state from the dedicated block Initial Conditions. Data is subject to potential delay, white noise filtering before used in the Control block and sent to the Motor Dynamics block. The estimated rotors' speed feeds the Aerodynamics block, which outputs the forces and moments of each propeller, sent to the System Dynamics block, along with the state and the actual rotors' speed to process the new state.

The Control block refers to attitude, altitude and position controllers as detailed in Figure 6.5. Each control loop is simulated at the sampling time of its respective sensor.

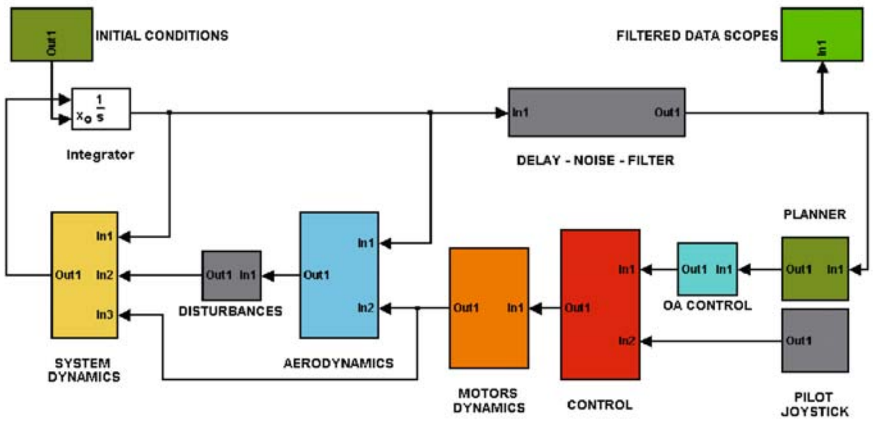


Fig. 6.4. The OS4 simulator block diagram.

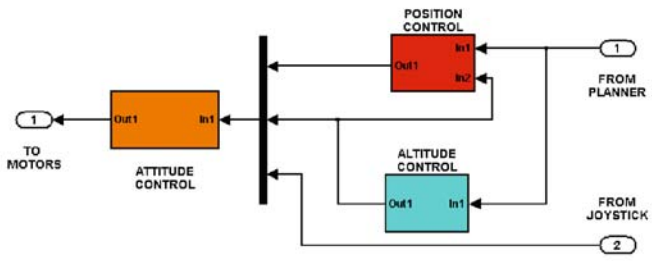


Fig. 6.5. Individual controller blocks in the OS4 simulator.

6.4 Design

In principle, interdependency of components during any design phase makes each component choice to depend on choice of all others and vice-versa. In order to best decide about the OS4 quadrotor design variables and components, a practical method was developed suitable for a small scale rotorcraft. The method described below combines models and databases of relevant components resulting in the best possible selection including battery selection to comply with the total mass constraint.

6.4.1 The General Method

The design process starts by defining three important design constraints for the small scale helicopter: Maximum mass $m_{\max}$, maximum span $s_{\max}$ and target thrust/weight ratio T_w. The three constraints give a good idea about the desired propeller diameter d_{prop}. In practice, the propeller span defines the overall span of the helicopter. Using the propeller diameter d_{prop}, one can estimate the characteristics of the propeller in terms of thrust, drag and power for a range of angular speeds. Based on the analysis in [19], it follows that the following relations should hold:

$$T \propto \Omega^2 L^4 ,\ D \propto \Omega^2 L^5 \text{ and } P \propto \Omega^3 L^5$$

L is a reference dimension, for example, the center of the blade. So, the mass $m_{\max}$, the drag moment D and the thrust/weight ratio T_w are sufficient to fully define the motor power requirements. This allows for selecting from a database a list of candidate actuators that offer the required power. Then, a rough estimation of the mass of the airframe m_{af} and avionics m_{av} is required to calculate a first estimate of the total mass without considering the battery as shown in Figure 6.6. Considering the mass of the battery, the maximum mass $m_{\max}$ may be defined as:

$$m_{\max} = m_{af} + m_{av} + m_{pg} + m_{bat}$$

where m_{pg} is the propulsion group (propeller, gearbox and motor) mass and m_{bat} is the mass of the battery. The iterative algorithm to calculate m_{pg} and m_{bat} is described next.

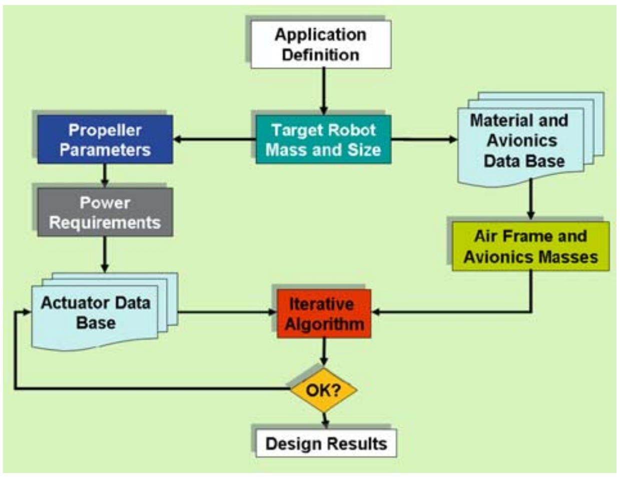


Fig. 6.6. The flowchart of the design method. The user has to define a target mass and size of the system in addition to airframe and avionics mass.

The Iterative Algorithm

A block diagram of the iterative algorithm is shown in Figure 6.7. The algorithm starts by choosing one candidate actuator from the database, its mass being $m_{pg}(i)$. Next, an initial value $m_{bat}(j_0)$ is given to the m_{bat} variable in order to calculate $m_{\max}(i,j) = m_{af} + m_{av} + m_{pg}(i) + m_{bat}(j)$. This 'temporary' total mass of the helicopter $m_{\max}(i,j)$ is used to estimate other variables at the two operational points of hovering and maximum thrust as shown in Figure 6.7: For each candidate actuator, the variable $m_{bat}(j)$ is incremented until $m_{\max}$ is reached. This process allows for estimating under hovering and at maximum thrust, for each candidate actuator and for each increment of $m_{bat}(j)$ the following variables:

• Total mass:	m_{bat}
• Total power consumption:	P_{tot}
• Propulsion group efficiency:	η_{gb}
• Propulsion group cost factor:	$C = P_{el} /(T - m_{pg})$
• Propulsion group quality factor:	$Q = B_W T_w / \Omega C$

(B_W: PG bandwidth, T_w: thrust/weight ratio)

- Operational time (autonomy): $Au = m_{bat}C_{bat} / P_{el}$
- Design quality index: $Q_{in} = Au / P_{tot}$

The propulsion group cost factor C describes the cost in power of the lifted mass. The propulsion group quality factor Q describes the quality of the mass lifting. The propulsion group quality factor is necessary to take into account actuator bandwidth and thrust to weight ratio. On the other hand, the design quality index constraints the operational time with regards to total power consumption.

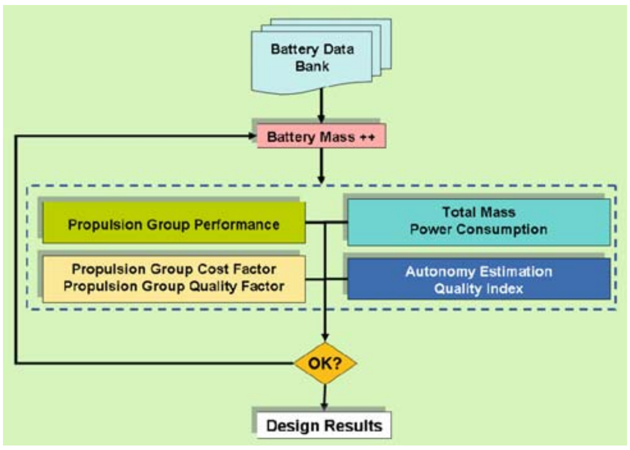

Battery Data Bank
Battery Mass ++
Propulsion Group Performance
Total Mass Power Consumption
Propulsion Group Cost Factor Propulsion Group Quality Factor
Autonomy Estimation Quality Index
OK?
Design Results

Fig. 6.7. The iterative algorithm flowchart.

6.4.2 The OS4 Quadrotor

The design of the OS4 quadrotor has been based on two main constraints: maximum mass of 500 gr and a maximum span of 800 mm. A propeller with a 300 mm diameter was selected that satisfied the span constraint. Table 6.3 shows the main design variables of a propulsion group that have been used in the models shown in Table 6.4.

Propeller		OS4	Unit
Mass	m_p	5.2	gr
Thrust coefficient	b	3.13e-5	N s ^{2}
Drag Coefficient	d	7.5e-7	Nm s ^{2}
Inertia	J_r	6e-5	kg.m ^{2}
Gearbox		**OS4**	**unit**
Efficiency	η	90	%
Mass	m_{gb}	7	gr
Max torque		0.15	Nm
Max speed		1000	rad/s
Reduction ratio	r	4:1	
Motor		**OS4**	**unit**
Efficiency at hover	η_m	64	%
Mass	m_m	12	gr
Max power	P_{el}	35	W
Internal resistance	R_{mot}	0.6	Ω
Inertia	J_m	4e-7	kgm ^{2}
Torque coefficient	k	5.2	mNm/A

Table 6.3. The OS4 propulsion group design variables.

The choice of the propulsion group is determined by the iterative algorithm that returns a classification based on the cost and quality factors. It returns a battery mass m_{bat}=230 gr (11 V, 3.3 Ah) of Lithium-Polymer. On the rotor side, the tests revealed that a gearbox is necessary. In fact, a direct-drive propulsion group would allow only a thrust to weight ratio of T_w=0.75, which is obviously not enough for lifting purposes. The selected motor is a brushless DC motor (12 gr, 35 W) with a high power to weight ratio, which justifies the choice even including its control electronics. A 6 gr I^2C controller was specially designed for the sensorless out runner LRK195.03 motor as shown in Figure 6.8. Obviously, BLDC motors offer high life time and low electromagnetic noise. The ready to plug propulsion group weighs 40 gr and lifts more than 260 gr.

Component	Model
Propeller	$(b,d)\Omega^2 = (T,D)$
Gearbox	$P_{in}\eta = P_{out}$
DC motor	$-\frac{k^2}{R_{mot}}\omega - D + \frac{k}{R_{mot}}u = J\frac{d\omega}{dt}$
PG cost	$P_{el}/(T - m_{pg}) = C$
PG quality	$B_W T_w/\Omega C = Q$

Table 6.4. Propulsion group (motor, gearbox, propeller) component models.

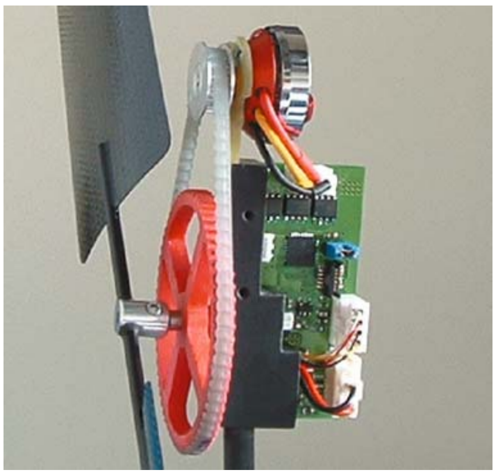


Fig. 6.8. OS4 propulsion group. The module is interfaced through I^2C bus and has a local PI speed controller.

Figure 6.9 illustrates the OS4 component block diagram. Embedding the controller is definitely advisable because it avoids delays and interruptions with wireless connections. A miniature computer module, based on a Geode 1200 processor running at 266MHz with 128M of RAM and flash memory was used. The computer module is x86 compatible and offers all standard PC interfaces (see Figure 6.10). The whole computer has a weight of 44 gr, it is 56 mm by 71 mm in size and runs Linux.

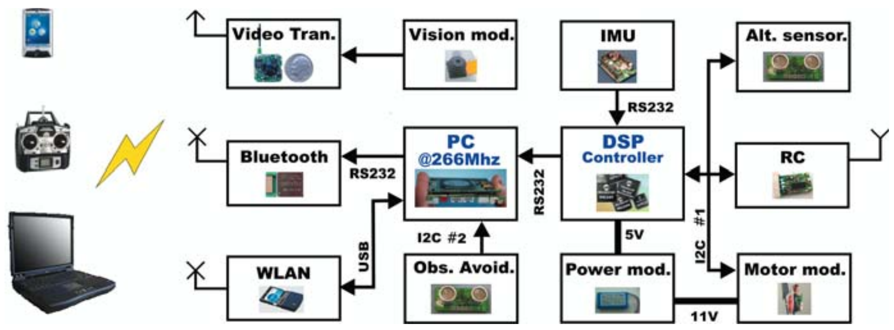


Fig. 6.9. The OS4 block diagram. A DSP processor handles attitude and altitude control. A miniature PC (x-board) handles obstacle avoidance control and communication tasks. The helicopter communicates through a WiFi interface and accepts standard remote control signals.

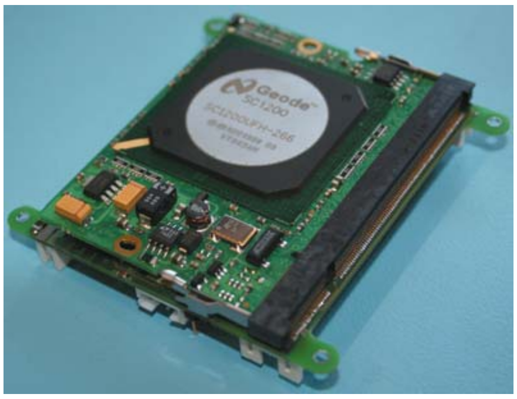


Fig.6.10. The x-board based, 40 gr, 56x71 mm computer module.

The controller includes an MCU for interfacing Bluetooth with the computer module. The same chip is used to decode the PPM signal picked from a 1.6 gr, 5 channel commercially available RC receiver. This decoding on the MCU makes it possible to interface the RC receiver to an I^2C bus and at the same time detects any problem in the channels. It is also possible to control the helicopter using a standard remote control.

Position Sensor

The OS4 position sensor is based on an on-board down-looking CCD camera and a simple pattern on the ground. The camera provides a motion blur free image of 320x240 at up to 25 fps. The algorithm detects the pattern, estimates the pose and provides the camera position (*x, y*) and heading angle (ψ). The image is primarily sent to an off-board computer for processing and then the position data is sent back to the helicopter for control purposes as shown in Figure 6.11.

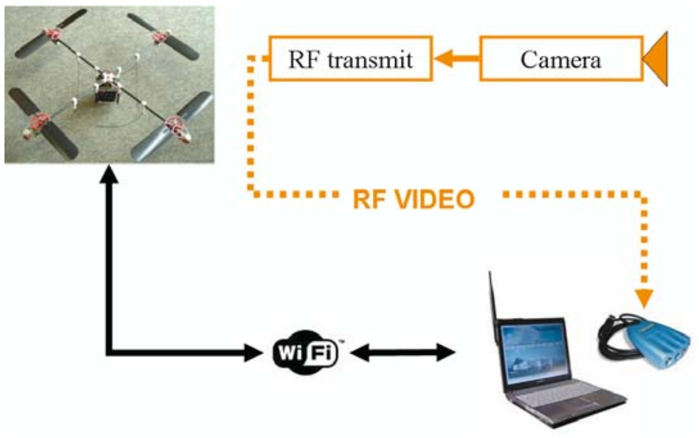


Fig. 6.11. Position sensing setup on the OS4.

Several alternatives have been considered for pattern detection. The first method that was tested refers to detecting five red dots on an A4 paper; this method suffers from sensitivity to lighting conditions. For the second test four LEDs were considered with different colors on an A4 size board; it was hard to tune LED intensity for the overall working volume. Then, a red A4 paper was used with a white spot shifted from the pattern center; this time the pattern was robustly detected using the Canny edge detector and Douglas-Peucker algorithms already implemented in OpenCV. In addition, a least-square based linear regression was used to refine detection. Pose estimation was performed using PnP algorithm [6]. The sensor algorithm was then enhanced to incorporate different situations where the pattern was not or it was partially detected.

All processing takes about 7ms. Image capture takes 1ms with a PCI acquisition card and almost 20ms with a USB 1.1 device on a Pentium 4, 2.4GHz. The algorithm is limited to 25Hz by the camera frame rate (25 fps). The errors obtained in x and y position sensing were about 2 cm at 0.5 m/s. The error on the yaw was about 3° at 180°/s.

Obstacle Detection Setup

Four ultrasound range finders are mounted on the OS4 quadrotor for obstacle detection, one under each propeller as shown in Figure 6.12. Two short plastic tubes are mounted on each sensor in order to reduce the beam width.

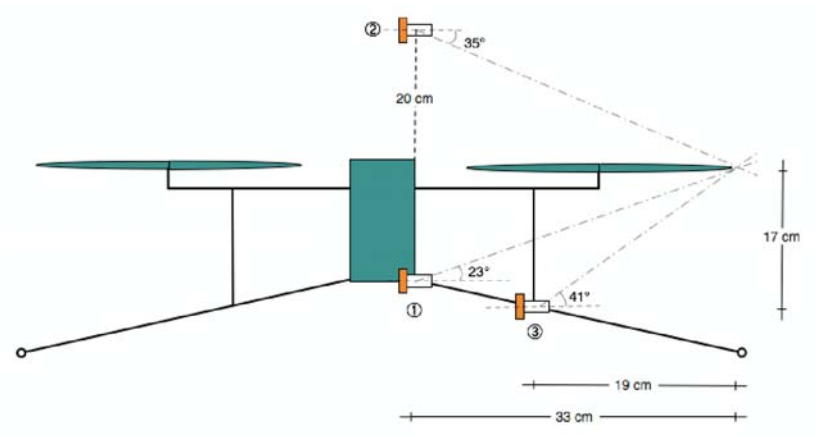


Fig. 6.12. Possible ultrasound sensor arrangement on the OS4. Position (3) was adopted after testing.

Design Results

Figure 6.13 shows the real quadrotor configuration. The quadrotor mass and power distribution are shown in Figure 6.14. The total mass in the initial design was about 520gr, with the battery accounting for almost one-half of the total mass and the actuators accounting for only one-third thanks to BLDC technology. All actuators account for 60W of 66W average power consumption. However, the latter depends on flight conditions and represents a weighted average between the equilibrium (40W) and the worst possible inclination state (120W) without loosing altitude.

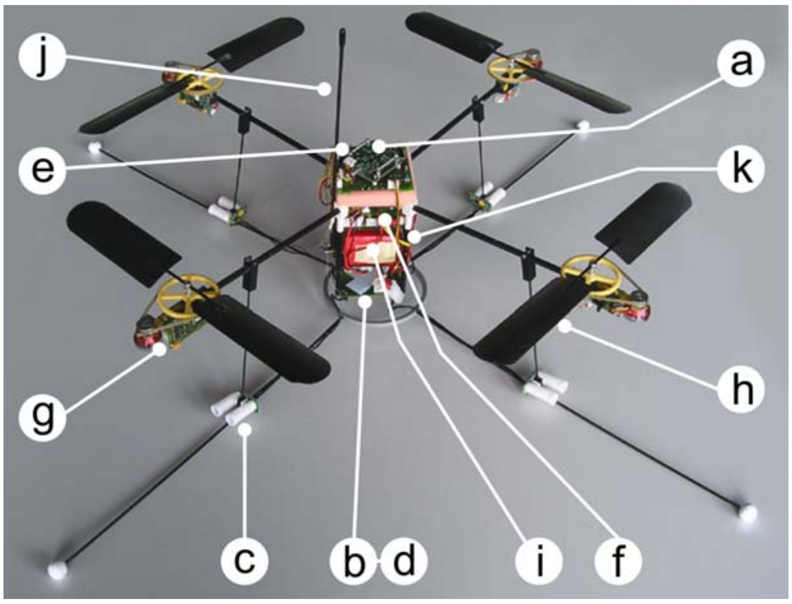


Fig. 6.13. Sensors, actuators and electronics of the OS4 quadrotor. *(a)* inertial measurement unit, *(b)* altitude sensor below the OS4, *(c)* obstacle avoidance sensor with tubes, *(d)* mini camera below the OS4, *(e)* DSP, *(f)* mother board, *(g)* motor module, *(h)* propeller, *(i)* battery, *(j)* RC antenna, *(k)* WiFi dongle.

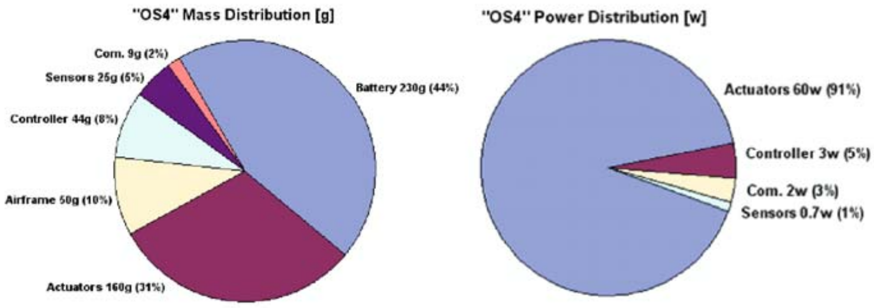


Fig.6.14. Mass and power distribution of the OS4 quadrotor. The battery mass accounts for almost one-half of the total mass while the actuators sink about 90% of the total power.

6.5 Simulation and Control

Improved performance expected from the new generation of MFRs is possible through derivation and implementation of specific control techniques incorporating limitations related to sensors and actuators. For the project under consideration, several control approaches were explored, from theoretical developments to final experimentation.

As a first attempt, two linear controllers were tested, a PID and an LQR that was based on a simplified model. The main result was autonomous hover already presented in [2]. However, strong disturbances (strong wind conditions) were poorly rejected.

In the second attempt control was reinforced using backstepping techniques that allowed for strong disturbance rejection; however, stabilization in hover flight was rather delicate [3].

An additional improvement was to introduce integral backstepping [13]. The idea of using integral action in the backstepping design was first proposed in [12] and applied in [21]. This controller design is derived based on [21]. The result is that the OS4 quadrotor is able to perform autonomous hovering with altitude control and autonomous take-off and landing.

The OS4 controller is structured in terms of six different controllers as illustrated in Figure 6.15. The take-off and landing controller outputs the desired altitude (z_d) to the altitude controller, which outputs the desired overall thrust (T_d) based on sonar data. The position controller receives the OS4 position (x, y) and desired thrust and outputs the desired roll (ϕ_d) and pitch (θ_d) while the desired yaw (ψ_d) comes directly from the user. The attitude controller outputs the desired motor speed to the motor controllers. The integral backstepping technique is used for attitude, altitude and position control. This results in a powerful and flexible control structure.

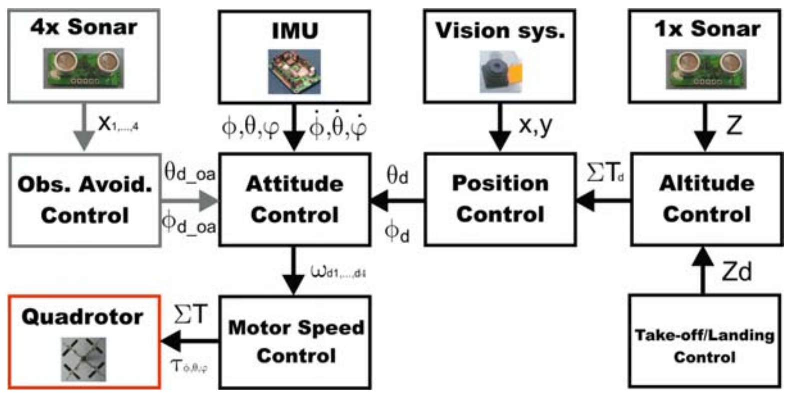


Fig. 6.15. The control structure implemented on the OS4 quadrotor.

6.5.1 Model Simplifications

The differential equations of motion of the quadrotor are shown in (6.8). However, it is advisable to simplify the model in order to comply with real-time constraints of the embedded control loop. Hence, hub forces and rolling moments are neglected, while the thrust and drag coefficients are considered to be constant. The equations of motion may be rewritten in state-space form $\dot{X} = f(X,U)$ with U representing the input vector and X the state vector, chosen as follows:

$$X = [\phi\ \dot\phi\ \theta\ \dot\theta\ \psi\ \dot\psi\ z\ \dot z\ x\ \dot x\ y\ \dot y]^T \tag{6.10}$$

$$U = [U_1\ U_2\ U_3\ U_4]^T \tag{6.11}$$

$$\begin{cases} U_1 = b(\Omega_1^2 + \Omega_2^2 + \Omega_3^2 + \Omega_4^2) \\ U_2 = b(-\Omega_2^2 + \Omega_4^2) \\ U_3 = b(\Omega_1^2 - \Omega_3^2) \\ U_4 = d(-\Omega_1^2 + \Omega_2^2 - \Omega_3^2 + \Omega_4^2) \end{cases} \tag{6.12}$$

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 a unity matrix if perturbations from hover 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 (6.8), (6.10) and (6.11) one obtains:

$$f(X,U) = \begin{pmatrix} \dot{\phi} \\ \dot{\theta}\dot{\psi}a_1 + \dot{\theta}a_2\Omega_r + b_1U_2 \\ \dot{\theta} \\ \dot{\phi}\dot{\psi}a_3 + \dot{\phi}a_4\Omega_r + b_2U_3 \\ \dot{\psi} \\ \dot{\theta}\dot{\phi}a_5 + b_3U_4 \\ \dot{z} \\ -g + (\cos\phi\cos\theta)\frac{1}{m}U_1 \\ \dot{x} \\ u_x\frac{1}{m}U_1 \\ \dot{y} \\ u_y\frac{1}{m}U_1 \end{pmatrix} \tag{6.13}$$

$$\begin{pmatrix} a_1 = (I_{yy} - I_{zz})/I_{xx} \\ a_2 = -J_r/I_{xx} \\ a_3 = (I_{zz} - I_{xx})/I_{yy} \\ a_4 = J_r/I_{yy} \\ a_5 = (I_{xx} - I_{yy})/I_{zz} \\ b_1 = l/I_{xx} \\ b_2 = l/I_{yy} \\ b_3 = 1/I_{zz} \end{pmatrix} \tag{6.14}$$

$$\begin{cases} u_x = (\cos\phi\sin\theta\cos\psi + \sin\phi\sin\psi) \\ u_y = (\cos\phi\sin\theta\sin\psi - \sin\phi\cos\psi) \end{cases} \tag{6.15}$$

It is worthwhile to mention that in the latter system the angles and their time derivatives do not depend on translation components. On the other hand, translations depend on the angles. One may ideally imagine the overall system described by (6.13) as being composed of two subsystems, the angular rotations and the linear translations.

6.5.2 Attitude Control

The attitude controller is the heart of the overall control system; it keeps the 3-D orientation of the helicopter at the desired value. Usually roll and pitch angles are forced to zero allowing for hovering. The attitude control loop runs at 76Hz, the update rate of the Microstrain 3DM-GX1 IMU. The latter provides the rates of turn and orientation around (x, y, z) axes with a dynamic accuracy of $\pm 2^{\circ}$. The first step in the IB controller design is to consider the tracking error $e_1 = \phi_d - \phi$ and its dynamics:

$$\frac{de_1}{dt} = \dot{\phi}_d - \omega_x \tag{6.16}$$

The angular speed ω_x is not considered a control input; it has its own dynamics, so it is set at a desired behavior interpreted as virtual control:

$$\omega_{xd} = c_1 e_1 + \dot{\phi}_d + \lambda_1 \chi_1 \tag{6.17}$$

c_1 and λ_1 are positive constants and $\chi_1 = \int_0^t e_1(\tau) d_\tau$ is the integral of the roll tracking error. So, the integral term is now introduced in (6.17). Since ω_x has its own error e_2, its dynamics are computed using (6.17) as follows:

$$\frac{de_2}{dt} = c_1(\dot{\phi}_d - \omega_x) + \ddot{\phi}_d + \lambda_1 e_1 - \ddot{\phi} \tag{6.18}$$

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

$$e_2 = \omega_{xd} - \omega_x \tag{6.19}$$

Using (6.17) and (6.19) the roll tracking error dynamics are re-written as:

$$\frac{de_1}{dt} = -c_1 e_1 - \lambda \chi_1 + e_2 \tag{6.20}$$

Replacing $\ddot{\phi}$ in (6.18) by its corresponding expression from (6.13), the real control input U_2 appears in equation:

$$\frac{de_2}{dt} = c_1(\dot{\phi}_d - \omega_x) + \ddot{\phi}_d + \lambda_1 e_1 - \dot{\theta}\dot{\psi} a_1 - \dot{\theta} a_2 \Omega_r - b_1 U_2 \tag{6.21}$$

Therefore, using (6.16), (6.20) and (6.21) tracking errors of the position e_1, angular speed e_2 and integral position tracking error χ_1 are combined to obtain:

$$\frac{de_2}{dt} = c_1(-c_1e_1 - \lambda_1\chi_1 + e_2) + \ddot{\phi}_d + \lambda_1 e_1 - \tau_x / I_{xx} \tag{6.22}$$

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

$$\frac{de_2}{dt} = -c_2e_2 - e_1 \tag{6.23}$$

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

$$U_2 = \frac{1}{b_1}[(1-c_1^2+\lambda_1)e_1 + (c_1+c_2)e_2 - c_1\lambda_1\chi_1 + \ddot{\phi}_d - \dot{\theta}\dot{\psi}a_1 - \dot{\theta}a_2\Omega_r] \tag{6.24}$$

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

$$\begin{cases} U_3 = \frac{1}{b_2}[(1-c_3^2+\lambda_2)e_3 + (c_3+c_4)e_4 - c_3\lambda_2\chi_2 + \ddot{\theta}_d - \dot{\phi}\dot{\psi}a_3 + \dot{\phi}a_4\Omega_r] \\ U_4 = \frac{1}{b_3}[(1-c_5^2+\lambda_3)e_5 + (c_5+c_6)e_6 - c_5\lambda_3\chi_3] \end{cases} \tag{6.25}$$

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.

Stability Analysis

To prove stability, the following candidate Lyapunov function is chosen:

$$V = \lambda\frac{1}{2}\chi_1^2 + \frac{1}{2}e_1^2 + \frac{1}{2}e_2^2 \tag{6.26}$$

It includes the position tracking error e_1, its integration χ_1 and velocity tracking error e_2. From (6.26) and using (6.20) and (6.23) one obtains:

$$\dot{V}(e_1, e_2) < 0, \forall(e_1, e_2) \neq 0 \text{ or } dV/dt = -c_1e^2 - c_2e^2 \tag{6.27}$$

The definition of (6.26) and because of (6.27) guarantee that e_1, χ_1 and e_2 are bounded. The desired position reference ϕ_d is bounded by assumption and $e_1 = \phi_d - \phi$ is also bounded, so, position ϕ is also bounded. This implies that the virtual control ω_x is bounded. Finally, the boundedness of the overall control torque is due to the choice of the control law in (6.24). The system is also globally asymptotically stable from the positive definition of V and the fact that $\dot{V}(e_1,e_2) < 0, \forall (e_1,e_2) \neq 0$ and $\dot{V}(0) = 0$.

Results

Attitude control performance is of crucial importance; it is directly linked to the performance of the actuators. OS4 is equipped with brushless sensorless motors that are powerful enough to avoid amplitude saturation. However, they suffer from low dynamics and, thus, from bandwidth saturation. This was taken into account in control design. Simulation results are shown in Figure 6.16 performed with a model including the actuators' dynamics and amplitude saturation. The simulation takes into account the delay and the noise inherent to sensors. The task was to stabilize roll, pitch and yaw angles at zero. Control parameters used for simulation were $C_1 = 10$, $C_2 = 2$, $C_3 = 10$, $C_4 = 2$, $C_5 = 2$, $C_6 = 2$.

The experimental results shown in Figure 6.17 correspond to a free flight with attitude references set at zero. One may observe in the roll and pitch plots a bounded oscillation of 0.1 rad in amplitude. This oscillation is not felt during flight; nevertheless, it is due to the slow dynamics of the OS4 actuators coupled with the differences between the four propulsion groups. Control parameters used in this experiment were $C_1 = 10.5$, $C_2 = 2$, $C_3 = 10$, $C_4 = 2$, $C_5 = 2$, $C_6 = 2$. These are very close to the parameters used for simulation purposes, which highlights the quality of the model.

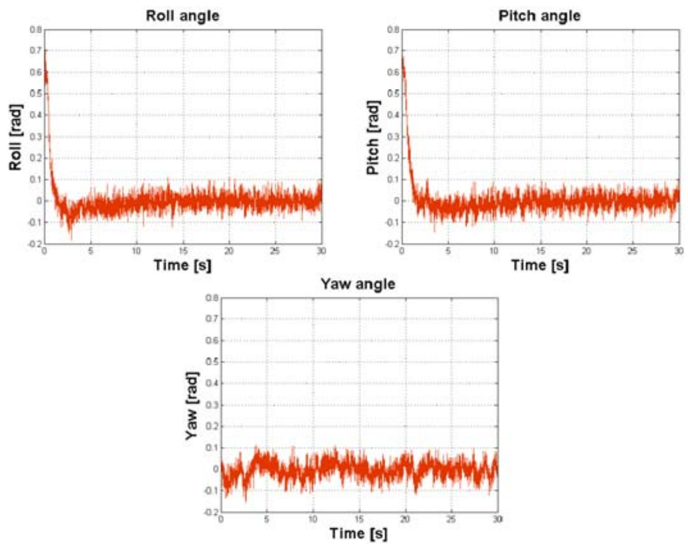


Fig. 6.16. Simulation: Integral backstepping attitude controller has to maintain roll, pitch and yaw angles to zero. Despite of the hard initial conditions and the white noise, the helicopter is quickly brought back to equilibrium.

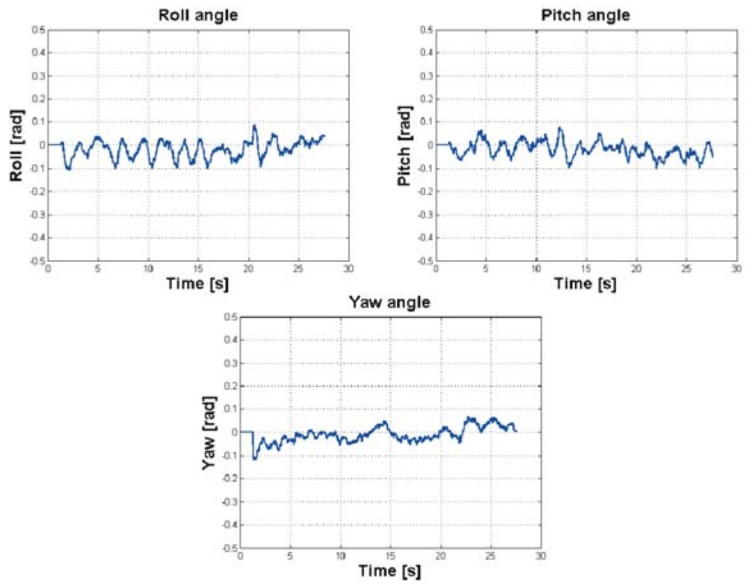


Fig. 6.17. Experiment: Integral backstepping attitude controller maintaining attitude angles to zero during flight. The helicopter is stabilized despite disturbances due to yaw drift, sensor noise and unmodeled effects.

6.5.3 Altitude Control

The altitude controller keeps the helicopter distance from the ground at a desired value. It is accomplished using a sonar (Devantech SRF10) that gives the range to the closest obstacle at 15 Hz. The accuracy depends on the distance, with an error of about 1-2 cm at 1 m. The necessary altitude rate of change is estimated based on the range. For control purposes, the altitude tracking error and the speed tracking error are defined as:

$$e_7 = z_d - z \tag{6.28}$$

$$e_8 = c_7 e_7 + \dot{z}_d + \lambda_4 \chi_4 - \dot{z} \tag{6.29}$$

The control law is then:

$$U_1 = \frac{m}{cos\phi cos\theta}[g + (1 - c_7^2 + \lambda_4)e_7 + (c_7 + c_8)e_8 - c_7\lambda_4\chi_4] \tag{6.30}$$

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

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. One can see in Figure 6.18 that the desired altitude reference is gradually reduced by a fixed step k ($k > 0$) that 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. Ground effect was not implemented because the landing skids are long enough to keep the propellers out of ground effect even after touchdown.

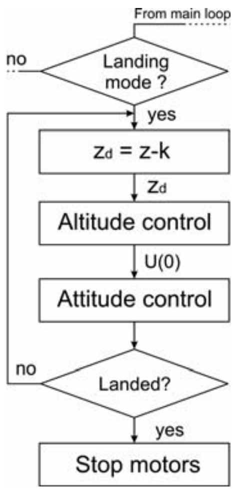


Fig. 6.18. Autonomous landing flowchart. Altitude reference is gradually reduced taking into account the dynamics of the quadrotor.

Results

Altitude control works surprisingly well despite all sonar limitations. Figure 6.19 shows an altitude reference profile (green) followed by the simulated controller (red) and the real controller (blue). The task was to climb to 0.5 m, hover, and then land. Chosen control parameter values were set to $C_7 = 3.5$, $C_8 = 1.5$ for simulation and $C_7 = 4$, $C_8 = 2$ for the experiments. The slight deviation between simulations and experimentation in take-off and landing phases is due to 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 deviation of 3 cm from the reference.

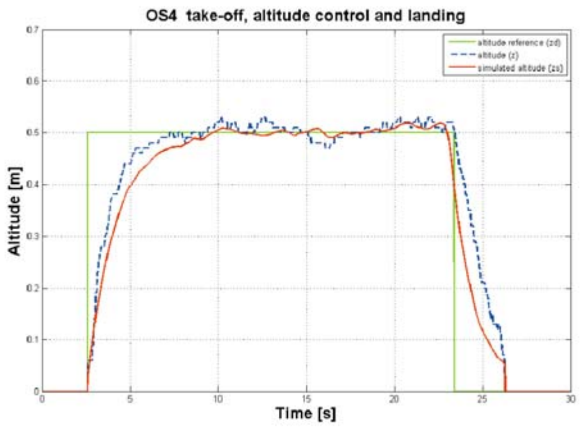


Fig. 6.19. Autonomous take-off, altitude control and landing by simulation and in real flight.

6.5.4 Position Control

Position control keeps the helicopter over the desired point, in this case, the (x, y) horizontal position with respect 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 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 Figure 6.15). The thrust vector orientation in the earth-fixed frame is given by R, the rotation matrix. Applying small angle approximation to R gives:

$$R = \begin{bmatrix} 1 & \psi & \theta \\ \psi & 1 & -\phi \\ -\theta & \phi & 1 \end{bmatrix} \tag{6.31}$$

From (6.13) and using (6.31) one can simplify horizontal motion dynamics to:

$$\begin{bmatrix} m\ddot{x} \\ m\ddot{y} \end{bmatrix} = \begin{bmatrix} -\theta U_1 \\ \phi U_1 \end{bmatrix} \tag{6.32}$$

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

$$\begin{cases} e_9 = x_d - x \\ e_{11} = y_d - y \end{cases} \tag{6.33}$$

Accordingly, speed tracking errors are:

$$\begin{cases} e_{10} = c_9 e_9 + \dot{x}_d + \lambda_5 \chi_5 - \dot{x} \\ e_{12} = c_{11} e_{11} + \dot{y}_d + \lambda_6 \chi_6 - \dot{y} \end{cases} \tag{6.34}$$

The control laws are then:

$$\begin{cases} U_x = \dfrac{m}{U_1}[(1 - c_9^2 + \lambda_5)e_9 + (c_9 + c_{10})e_{10} - c_9 \lambda_5 \chi_5] \\ U_y = -\dfrac{m}{U_1}[(1 - c_{11}^2 + \lambda_6)e_{11} + (c_{11} + c_{12})e_{12} - c_{11} \lambda_6 \chi_6] \end{cases} \tag{6.35}$$

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

Results

The main result in position control was obtained by simulation. Observing again Figure 6.15 one sees how the different controllers are cascaded. In fact, only the attitude is driven by the position, the altitude controller is simply feeding signal 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 chosen for simulation purposes were $C_9 = 2, C_{10} = 0.5, C_{11} = 2, C_{12} = 0.5$; results are shown in Figure 6.20.

Waypoint Following

The planner block in Figure 6.4 defines the waypoints, and hence, the trajectories OS4 has to follow. The position of the next waypoint is sent to the position controller that directs the helicopter towards the goal. A waypoint is declared reached when the helicopter enters a sphere around this point. The radius of this sphere (defined to be 0.1 m) is the maximum allowable error. Figure 6.21 shows a square trajectory defined by four waypoints. The task was to climb to 1 m from the ground and then follow the four waypoints of a square of 2 m side.

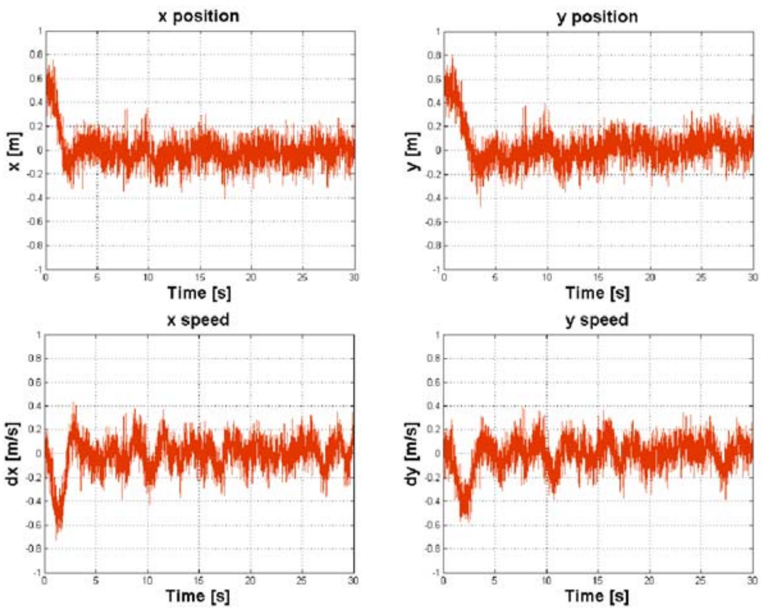


Fig. 6.20. Simulation: Integral backstepping position controller drives attitude controller in order to keep the helicopter over a given point.

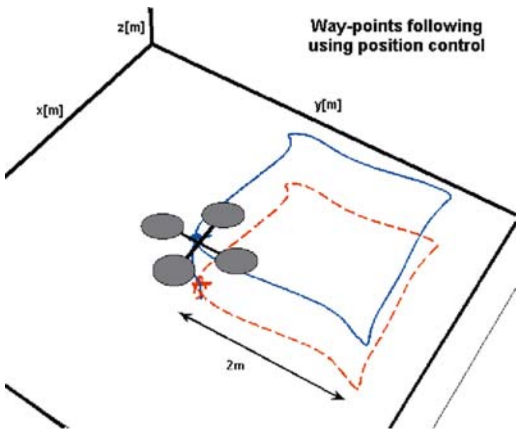


Fig. 6.21. Four waypoints for a square trajectory tracked by OS4.

In order to track the square trajectory, the planner generates the (x_d, y_d) position references, and consequently the position controller generates the (ϕ_d, θ_d) attitude references. Figure 6.22 depicts these signals and shows that the 2 m side square is tracked with about 10% overshoot (20cm), while the trajectory is completed in 20s.

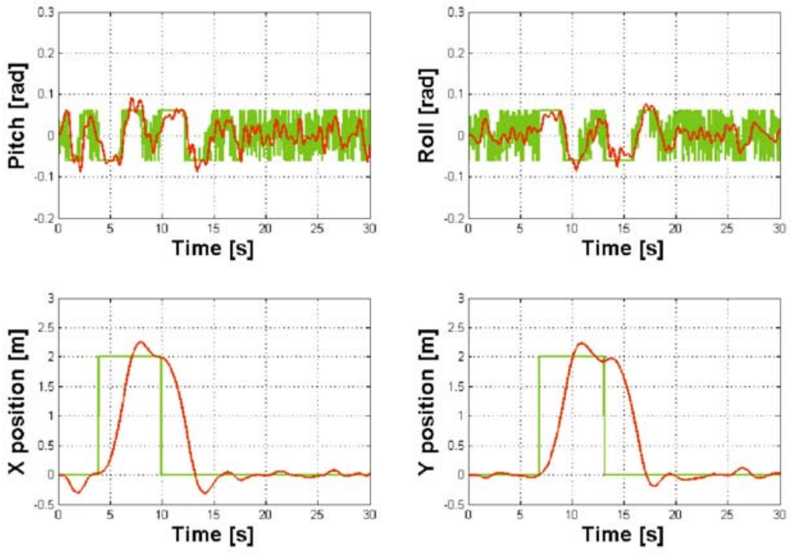


Fig. 6.22. Simulation: The position and attitude signals generated to track the square trajectory.

6.5.5 Obstacle Avoidance

OS4 is equipped with a sonar-based obstacle avoidance system composed of four miniature ultrasound range finders in cross configuration. The obstacle avoidance controller is incorporated into the *Simulink* model along with the environment model and sensor libraries. Aiming at simplifying the procedure, it was decided to keep the altitude constant during evasive maneuvers. This would reduce the path planning complexity to a 2-D problem. Direction of flight was also restricted; the OS4 could move only on four directions where the ultrasound sensors were placed. To increase the flight safety, a 90 cm radius security zone was constantly maintained between the helicopter and the environment (see Figure 6.23). This zone assured a 50 cm distance between the helicopter rotors and any obstacle. If an obstacle was detected inside the security zone, a safety loop (that ran in parallel to the obstacle avoidance controller) intervenes with the helicopter flight controller and generates an evasive maneuver. This maneuver is obtained by selecting pre-defined pitch and/or roll angles that would avoid a collision between helicopter and obstacles.

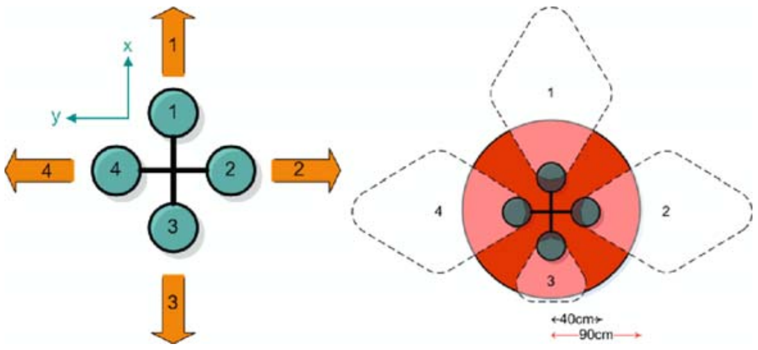


Fig. 6.23. The 4 flight directions (left) and the security zone (right).

Several algorithms were simulated with various results in a 100 m^2 environment, with obstacles represented as columns of 20 cm in diameter and 3 m in height. The approaches followed can be classified as relative position and speed-based approaches. The first obstacle avoidance controller uses the position controller to act on the relative position of the helicopter with respect to the closest obstacle (X_{oa}, Y_{oa}). The second one uses the speed controller to act on the speed of the vehicle $(\dot{X}, \dot{Y})$ if an obstacle is detected. The latter approach was used to develop the main obstacle avoidance controller algorithm. The idea was to act on $\dot{X}$ and $\dot{Y}$ while keeping the heading and altitude constant. When an obstacle is detected its distance from the helicopter is classified based on a given threshold as "far", "close" or "too close". If the obstacle is "far", no avoidance action is needed and there is no intervention with the helicopter normal flight. On the other hand, if the obstacle distance is "close" or "too close" then the obstacle avoidance controller informs the helicopter flight control, reduces its speed, and generates evasive maneuvers using pre-defined flight directions as shown in Figure 6.24. The selection of the direction of the evasive flight depends on the stimulated sensor and the desired flight direction previously selected by the user. However, if the quadrotor is surrounded by obstacles that are "too close", it reduces the speed and keeps a hovering behavior.

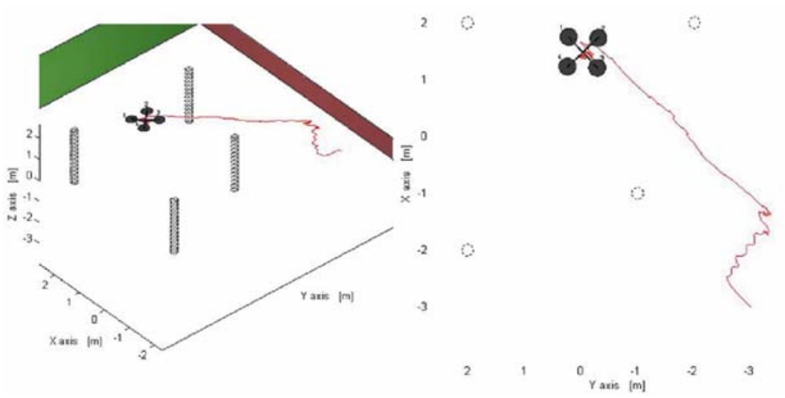


Fig. 6.24. Simulation: OS4 avoiding static obstacles.

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 affecting the OS4 stability. This was mainly due to interference between the five sonars and the effect of the propellers on the ultrasound waves. Figure 6.25 shows detection of an obstacle with and without the filter. The latter is based on the variation of successive samples and gives a reliable detection signal usable in flight.

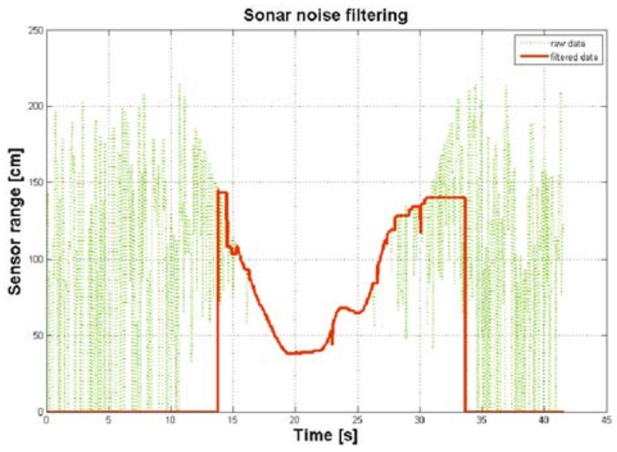


Fig. 6.25. Obstacle detection with and without the filter.

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 6.26 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.

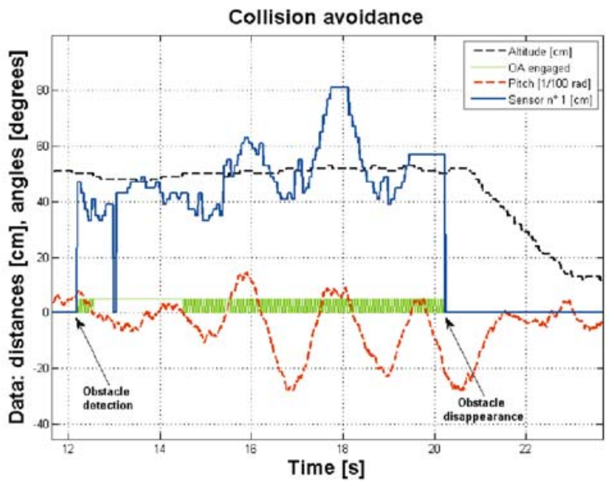


Fig. 6.26. Experiment: Collision avoidance with OS4. The helicopter flies back until the obstacle disappears

6.6 Conclusion and Future Work

This Chapter presented the latest developments of the ASL MFR project. A quadrotor simulation model was introduced. It included major aerodynamic effects modeled with blade element and momentum theory. In addition, the actuator's model was identified and all sensor delays and noise were taken into account. Moreover, a *MATLAB / Simulink* based simulator was developed for testing the controllers. Real experiments were conducted with the same control parameters tuned during simulation. A practical methodology was introduced for miniature rotorcraft design. It was the only tool used to design the OS4 helicopter achieving 100% thrust margin for 30 min autonomy (in the initial design). The last version of OS4 embeds all necessary avionics and energy devices for a fully autonomous flight. This comprises a low cost IMU, a vision based position sensor specifically developed for this project and an obstacle detection setup. The control approach was then presented to design the main controllers: Attitude, altitude, position and the auxiliary ones like take-off and landing, obstacle avoidance and waypoint following. The latter was demonstrated using simulation. Experiments have shown that the OS4 is currently able to take-off, hover, land and avoid collisions automatically thanks to model-based control. Future work is to enhance the propulsion group towards more reliability and high bandwidth, improve the vision sensor in order to get rid of the external pattern and to provide stable vision-based yaw information. It is also possible to reduce the mass of the helicopter by using the tiny single board computers available now. It will be interesting to practically test a waypoint following maneuver with obstacle avoidance capabilities. The OS4 is undoubtedly one of the most integrated quadrotor ever designed. It is the first one practically capable of a collision avoidance maneuver.

References

1. AERO-EPFL, http://aero.epfl.ch/.
2. Bouabdallah S. et al, "PID vs LQ Control Techniques Applied to an Indoor Micro Quadrotor", *Proceedings, IEEE/RSJ International Conference on Intelligent Robots and Systems,* Sendai, Japan, 2004.
3. Bouabdallah S., Siegwart R., "Backstepping and Sliding-Mode Techniques Applied to an Indoor Micro Quadrotor", *Proceedings, IEEE International Conference on Robotics and Automation*, Barcelona, Spain 2005.
4. Bouabdallah S. et al., "Autonomous Miniature Flying Robots: Coming Soon", *IEEE Robotics and Automation Magazine*, to appear.
5. Cheeseman I., Bennett W., "*The Effect of the Ground on a Helicopter Rotor in Forward Flight*", Aeronautical Research Council, No. 3021, 1957.
6. DeMenthon D., Davis L., "Model-Based Object Pose in 25 lines of Code", *International Journal of Computer Vision*, Vol. 15, 2005.
7. Deng X., et al., "Attitude Control for a Micromechanical Flying Insect Including Thorax and Sensor Models", *Proceedings, International Conference on Robotics and Automation*, Teipei, Taiwan, 2003.
8. Fay G., *Derivation of the Aerodynamic Forces for the Mesicopter Simulation*, Stanford University, 2001.
9. Griffiths D., Leishman J., "A Study of Dual-Rotor Interference and Ground Effect Using a Free-Vortex Wake Model", *Proceedings, 58th Annual Forum of the American Helicopter Society*, Montréal, Canada, 2002.
10. Guenard N., et al., "Control Laws for the Teleoperation of an Unmanned Aerial Vehicle Known as an x4-Flyer", *Proceedings, IEEE/RSJ International Conference on Intelligent Robots and Systems*, Beijing, China, 2006.

11. Hoffmann G., et al., “The Stanford Testbed of Autonomous Rotorcraft for Multi Agent Control”, *Proceedings, 23rd Digital Avionics Systems Conference*, Salt Lake City, Utah, 2004.
12. Kanellakopoulos I., Krein P., “Integral-Action Nonlinear Control of Induction Motors”, *Proceedings, 12th IFAC World Congress*, Sydney, Australia, 1993.
13. Krstic M., et al., *Nonlinear and Adaptive Control Design*, Wiley Interscience, 1995.
14. Kroo I., et al., *The Mesicopter: A Miniature Rotorcraft Concept - Phase 2*, Interim Report, 2000.
15. Leishman J. G., *Principles of Helicopter Aerodynamics*, Cambridge University Press.
16. MARV, http://www.enae.umd.edu/AGRC/Design00/MARV.html.
17. Murray R., et al**.,** *A Mathematical Introduction to Robotic Manipulation*, CRC, Boca Raton, FL 1994.
18. Wang W., et al., “Autonomous Control for Micro-Flying Robot and Small Wireless Helicopter”, *Proceedings, IEEE/RSJ International Conference on Intelligent Robots and Systems*, Beijing, China, 2006.
19. Nicoud J. D., Zufferey J. C., “Toward Indoor Flying Robots”, *Proceedings, IEEE/RSJ International Conference on Intelligent Robots and Systems,* Lausanne, Switzerland, 2002.
20. OpenCV, http://www.intel.com/research/mrl/research/opencv/.
21. Tan Y., et al., “Advanced Nonlinear Control Strategy for Motion Control Systems”, *Proceedings, IEEE Power Electronics and Motion Control Conference*, Beijing, China, 2000.