We are IntechOpen, the world's leading publisher of Open Access books Built by scientists, for scientists

6,900
Open access books available

185,000
International authors and editors

200M
Downloads

Our authors are among the

154
Countries delivered to

TOP 1%
most cited scientists

12.2%
Contributors from top 500 universities



WEB OF SCIENCE™

Selection of our books indexed in the Book Citation Index in Web of Science™ Core Collection (BKCI)

Interested in publishing with us?
Contact book.department@intechopen.com

Numbers displayed above are based on latest data collected.
For more information visit www.intechopen.com



5

Fault Tolerant Flight Control Techniques with Application to a Quadrotor UAV Testbed

Youmin Zhang and Abbas Chamseddine
Department of Mechanical and Industrial Engineering, Concordia University
Canada

1. Introduction

Unmanned Aerial Vehicles (UAVs) are gaining more and more attention during the last few years due to their important contributions and cost-effective applications in several tasks such as surveillance, search and rescue missions, geographic studies, as well as various military and security applications. Due to the requirements of autonomous flight under different flight conditions without a pilot onboard, control of UAV flight is much more challenging compared with manned aerial vehicles since all operations have to be carried out by the automated flight control, navigation and guidance algorithms embedded on the onboard flight microcomputer/microcontroller or with limited interference by a ground pilot if needed.

As an example of UAV systems, the quadrotor helicopter is relatively a simple, affordable and easy to fly system and thus it has been widely used to develop, implement and test-fly methods in control, fault diagnosis, fault tolerant control as well as multi-agent based technologies in formation flight, cooperative control, distributed control, mobile wireless networks and communications. Some theoretical works consider the problems of control (Dierks & Jagannathan, 2008), formation flight (Dierks & Jagannathan, 2009) and fault diagnosis (Nguyen et al., 2009; Rafaralahy et al., 2008) of the quadrotor UAV. However, few research laboratories are carrying out advanced theoretical and experimental works on the system. Among others, one may cite for example, the UAV SWARM health management project of the Aerospace Controls Laboratory at MIT (SWARM, 2011), the Stanford Testbed of Autonomous Rotorcraft for Multi-Agent Control (STARMAC) project (STARMAC, 2011) and the Micro Autonomous Systems Technologies (MAST) project (MAST, 2011).

A team of researchers at the Department of Mechanical and Industrial Engineering of Concordia University, with the financial support from NSERC (Natural Sciences and Engineering Research Council of Canada) through a Strategic Project Grant and a Discovery Project Grant and three Canadian-based industrial partners (Quanser Inc., Opal-RT Technologies Inc., and Numerica Technologies Inc.) as well as Defence Research and Development Canada (DRDC) and Laval University, have been working on a research and development project on fault-tolerant and cooperative control of multiple UAVs since 2007 (see the Networked Autonomous Vehicles laboratory (NAV, 2011)). In addition to the work that has been carried out for the multi-vehicles case, many fault-tolerant control (FTC) strategies have been developed and applied to the single vehicle quadrotor helicopter UAV system. The objective is to consider actuator faults and to propose FTC methods to

accommodate as much as possible the fault effects on the system performance. The proposed methods have been tested either in simulation, experimental or both frameworks where the experimental implementation has been carried out using a cutting-edge quadrotor UAV testbed known also as Qball-X4. The developed approaches include the Gain-Scheduled PID (GS-PID), Model Reference Adaptive Control (MRAC), Sliding Mode Control (SMC), Backstepping Control (BSC), Model Predictive Control (MPC) and Flatness-based Trajectory Planning/Re-planning (FTPR), etc. Some of these methods require an information about the time of occurrence, the location and the amplitude of faults whereas others do not. In the former case, a fault detection and diagnosis (FDD) module is needed to detect, isolate and identify the faults which occurred.

Table 1 summarizes the main fault-tolerant control methodologies that have been developed and applied to the quadrotor helicopter UAV. The table shows which methods require an FDD scheme and whether they have been tested in simulation or experimentally. In the case of experimental application, one can also distinguish between two categories of faults: a simulated fault and a real damage. In the first case, a fault has been generated in an actuator by multiplying its control input by a gain smaller than one, thus simulating a loss in the control effectiveness. In the second case, the fault/damage has been introduced by breaking the tip of a propeller of the Qball-X4 UAV during flight.

Strategy	Passive/Active	Need for FDD	Simulation	Experiment	Real damage
GS-PID	Active	✓	✓	✓	X
MRAC	Active	X	X	✓	✓
SMC	Passive	X	✓	✓	✓
BSC	Passive	X	✓	X	X
MPC	Active	✓	✓	X	X
FTPR	Active	✓	✓	✓	X

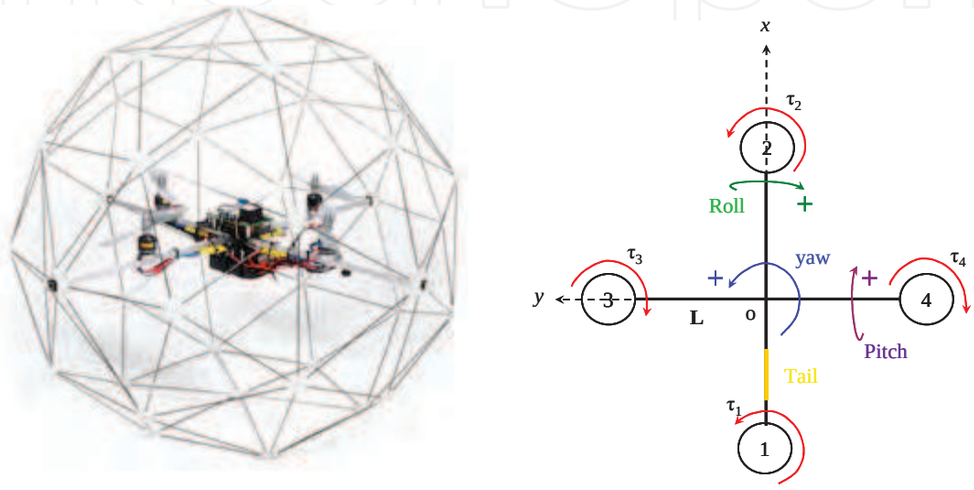
Table 1. Fault-tolerant control methods (Note: (✓) Yes/Done; (X) No/Not done yet).

These methods will be discussed in the subsequent sections. The remainder of this chapter is then organized as follows. Section 2 introduces the quadrotor UAV system that is used as a testbed for the proposed methods. Section 3 presents the FTC methods summarized in Table 1. The simulation as well as the experimental flight results are given and discussed in Section 4. Some concluding remarks together with future work are finally given. Note that due to space consideration, only some of the developed control methods will be included in this chapter. Interested readers can refer to the website of the NAV Lab on '`http://users.encs.concordia.ca/~ymzhang/UAVs.htm`' for more information.

2. Description and dynamics of the quadrotor UAV system

The quadrotor UAV of the Department of Mechanical and Industrial Engineering of Concordia University is the Qball-X4 testbed (Figure 1(a)) developed by Quanser Consulting Inc. The quadrotor UAV is enclosed within a protective carbon fiber round cage (therefore a name of Qball-X4) to ensure safe operation of the vehicle and protection to the personnel who is working with the vehicle in an indoor research and development environment. It uses

four 10-inch propellers and standard RC motors and speed controllers. It is equipped with the Quanser Embedded Control Module (QECM), which is comprised of a Quanser HiQ aero data acquisition card and a QuaRC-powered Gumstix embedded computer where QuaRC is Quanser's Real-time Control software. The Quanser HiQ provides high-resolution accelerometer, gyroscope, and magnetometer IMU sensors as well as servo outputs to drive four motors. The on-board Gumstix computer runs QuaRC, which allows to rapidly develop and deploy controllers designed in MATLAB/Simulink environment to real-time control the Qball-X4. The controllers run on-board the vehicle itself and runtime sensors measurement, data logging and parameter tuning is supported between the host PC and the target vehicle (Quanser, 2010).



(a) The Quanser Qball-X4 quadrotor UAV (b) Qball-X4 schematic representation

Fig. 1. The Quanser Qball-X4 quadrotor UAV and its schematic representation.

The block diagram of the entire UAV system is illustrated in Figure 2. It is composed of three main parts:

- The first part represents the Electronic Speed Controllers (ESCs) + the motors + the propellers in a set of four. The input to this part is $u = [u_1 \;\; u_2 \;\; u_3 \;\; u_4]^T$ which are Pulse Width Modulation (PWM) signals. The output is the thrust vector $T = [T_1 \;\; T_2 \;\; T_3 \;\; T_4]^T$ generated by four individually-controlled motor-driven propellers.
- The second part is the geometry that relates the generated thrusts to the applied lift and torques to the system. This geometry corresponds to the position and orientation of the propellers with respect to the system's center of mass.
- The third part is the dynamics that relate the applied lift and torques to the position (P), velocity (V) and acceleration (A) of the Qball-X4.

The subsequent sections describe the corresponding mathematical model for each of the blocks of Figure 2.

2.1 Qball-X4 dynamics

The Qball-X4 dynamics in a hybrid coordinate system are given hereafter where the position dynamics are expressed in the inertial frame and the angular dynamics are expressed in the

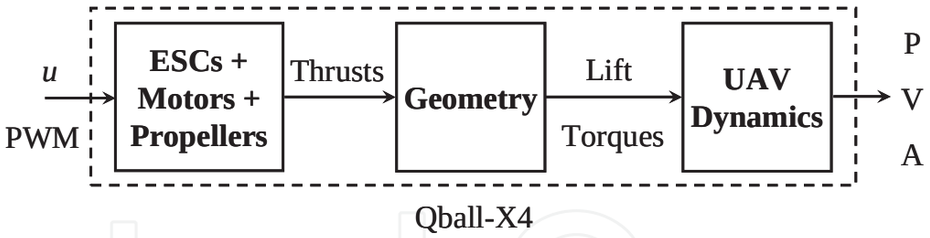


Fig. 2. The UAV system block diagram.

body frame (Bresciani, 2008):

$$\begin{aligned}
m\ddot{x} &= u_z\left(cos\phi\ sin\theta\ cos\psi + sin\phi\ sin\psi\right) - k_x\dot{x}\\
m\ddot{y} &= u_z\left(cos\phi\ sin\theta\ sin\psi - sin\phi\ cos\psi\right) - k_y\dot{y}\\
m\ddot{z} &= u_z\left(cos\phi\ cos\theta\right) - mg - k_z\dot{z}\\
J_x\dot{p} &= u_p + \left(J_y - J_z\right)qr - J_T q\,\Omega - k_p p\\
J_y\dot{q} &= u_q + \left(J_z - J_x\right)pr - J_T p\,\Omega - k_q q\\
J_z\dot{r} &= u_r + \left(J_x - J_y\right)pq - k_r r
\end{aligned} \tag{1}$$

where x, y and z are the coordinates of the quadrotor UAV center of mass in the inertial frame. m is the system mass and J_x, J_y and J_z are the moments of inertia along y, x and z directions respectively. θ, ϕ and ψ are the pitch, roll and yaw Euler angles and p, q and r are the angular velocities in the body-fixed frame. k_x, k_y, k_z, k_p, k_q and k_r are drag coefficients and are constant. J_T is the moment of inertia for each motor and Ω is the overall speed of propellers:

$$\Omega = -\Omega_1 - \Omega_2 + \Omega_3 + \Omega_4 \tag{2}$$

where Ω_i is the i^{th} propeller speed.

The angular velocities in the inertial frame (Euler rates) can be related to those in the body frame as follows:

$$\begin{bmatrix} p\\ q\\ r \end{bmatrix} = \begin{bmatrix} 1 & 0 & -sin\theta\\ 0 & cos\phi & cos\theta sin\phi\\ 0 & -sin\phi & cos\theta cos\phi \end{bmatrix} = \begin{bmatrix} \dot{\phi}\\ \dot{\theta}\\ \dot{\psi} \end{bmatrix} \tag{3}$$

Close to hovering conditions, the matrix in the above equation is close to identity matrix and therefore the angular velocities in the body frame can be seen as the angular velocities in the inertial frame. The model (1) can then be written as:

$$\begin{aligned}
m\ddot{x} &= u_z\left(cos\phi\ sin\theta\ cos\psi + sin\phi\ sin\psi\right) - k_x\dot{x}\\
m\ddot{y} &= u_z\left(cos\phi\ sin\theta\ sin\psi - sin\phi\ cos\psi\right) - k_y\dot{y}\\
m\ddot{z} &= u_z\left(cos\phi\ cos\theta\right) - mg - k_z\dot{z}\\
J_x\ddot{\theta} &= u_\theta + \left(J_y - J_z\right)\dot{\phi}\dot{\psi} - J_T\dot{\phi}\,\Omega - k_\theta\dot{\theta}\\
J_y\ddot{\phi} &= u_\phi + \left(J_z - J_x\right)\dot{\theta}\dot{\psi} - J_T\dot{\theta}\,\Omega - k_\phi\dot{\phi}\\
J_z\ddot{\psi} &= u_\psi + \left(J_x - J_y\right)\dot{\theta}\dot{\phi} - k_\psi\dot{\psi}
\end{aligned} \tag{4}$$

where u_p, u_q, u_r, k_p, k_q and k_r have been respectively changed to u_θ, u_ϕ, u_ψ, k_θ, k_ϕ, k_ψ for notation convenience. At low speeds, one can obtain a simplified nonlinear model of (4) by neglecting drag terms and gyroscopic and Coriolis-centripetal effects:

$$\begin{aligned} m\ddot{x} &= u_z\,(cos\phi\ sin\theta\ cos\psi + sin\phi\ sin\psi) \\ m\ddot{y} &= u_z\,(cos\phi\ sin\theta\ sin\psi - sin\phi\ cos\psi) \\ m\ddot{z} &= u_z\,(cos\phi\ cos\theta) - mg \\ J_x\ddot{\theta} &= u_\theta \\ J_y\ddot{\phi} &= u_\phi \\ J_z\ddot{\psi} &= u_\psi \end{aligned} \tag{5}$$

A further simplified linear model can be obtained by assuming hovering conditions ($u_z \approx mg$ in the x and y directions) with no yawing ($\psi = 0$) and small roll and pitch angles as follows:

$$\begin{aligned} \ddot{x} &= \theta g; & J_x\ddot{\theta} &= u_\theta \\ \ddot{y} &= -\phi g; & J_y\ddot{\phi} &= u_\phi \\ \ddot{z} &= u_z/m - g; & J_z\ddot{\psi} &= u_\psi \end{aligned} \tag{6}$$

The nonlinear model (4) will be used later on in the design of fault-tolerant control strategies such as the BSC, the MPC and the SMC. The simplified model (6) will be used for the MRAC design as well as for the trajectory planning and re-planning approach.

2.2 ESCs, motors and propellers

The motors of the Qball-X4 are outrunner brushless motors. The generated thrust T_i of the i^{th} motor is related to the i^{th} PWM input u_i by a first-order linear transfer function:

$$T_i = K\frac{\omega}{s+\omega}u_i \ \ ; \ \ i = 1, ..., 4 \tag{7}$$

where K is a positive gain and ω is the motor bandwidth. K and ω are theoretically the same for the four motors but this may not be the case in practice and therefore, this can be one of sources of modeling errors/uncertainties for fault-tolerant control design and trajectory planning/re-planning.

2.3 Geometry

A schematic representation of the Qball-X4 is given in Figure 1(b). The motors and propellers are configured in such a way that the back and front (1 and 2) motors spin clockwise and the left and right (3 and 4) spin counter-clockwise. Each motor is located at a distance L from the center of mass o and when spinning, a motor produces a torque τ_i. The origin of the body-fixed frame is the system's center of mass o with the x-axis pointing from back to front and the y-axis pointing from right to left. The thrust T_i generated by the i^{th} propeller is always pointing upward in the z-direction in parallel to the motor's rotation axis. The thrusts T_i and

the torques τ_i result in a lift in the z-direction (body-fixed frame) and torques about the x, y and z axis. The relation between the lift/torques and the thrusts is:

$$\begin{aligned} u_z &= T_1 + T_2 + T_3 + T_4 \\ u_\theta &= L(T_1 - T_2) \\ u_\phi &= L(T_3 - T_4) \\ u_\psi &= \tau_1 + \tau_2 - \tau_3 - \tau_4 \end{aligned} \tag{8}$$

The torque τ_i produced by the i^{th} motor is directly related to the thrust T_i via the relation of $\tau_i = K_\psi T_i$ with K_ψ as a constant. In addition, by setting $T_i \approx K u_i$ from (7), the relation (8) reads:

$$\begin{aligned} u_z &= K(u_1 + u_2 + u_3 + u_4) \\ u_\theta &= KL(u_1 - u_2) \\ u_\phi &= KL(u_3 - u_4) \\ u_\psi &= KK_\psi(u_1 + u_2 - u_3 - u_4) \end{aligned} \tag{9}$$

where u_z is the total lift generated by the four propellers and applied to the quadrotor UAV in the z-direction (body-fixed frame). u_θ, u_ϕ and u_ψ are respectively the applied torques in θ, ϕ and ψ directions (see Figure 1(b)).

3. Fault-tolerant control algorithms

Modern technological systems rely on sophisticated control systems to replace or reduce the intervention of human operators. In the event of malfunctions in actuators, sensors or other system components, a conventional feedback control design may result in an unsatisfactory performance or even instability of the controlled system. To prevent such situations, new control approaches have been developed in order to tolerate component malfunctions while maintaining desirable stability and performance properties. This is particularly important for safety-critical systems, such as aircrafts, spacecrafts, nuclear power plants, and chemical plants processing hazardous materials. In such systems, the consequences of a minor fault in a system component can be catastrophic. It is necessary then to design control systems which are capable of tolerating potential faults in these systems in order to improve the reliability and availability while providing a desirable performance. These types of control systems are often known as fault-tolerant control systems (FTCS). More precisely, FTCS are control systems which possess the ability to accommodate component failures automatically. They are capable of maintaining overall system stability and acceptable performance in the event of such failures (Zhang & Jiang, 2008).

Generally speaking, FTCS can be classified into two types: passive (PFTCS) and active (AFTCS). In PFTCS, controllers are fixed and are designed to be robust against a class of presumed faults. This approach needs neither FDD schemes nor controller reconfiguration, but it has limited fault-tolerant capabilities. In contrast to PFTCS, AFTCS react to the system component failures actively by reconfiguring control actions so that the stability and acceptable performance of the entire system can be maintained (Zhang & Jiang, 2008). An overall structure of a typical AFTCS is shown in Figure 3. In the FDD module, faults must be detected and isolated as quickly as possible, and fault parameters, system

state/output variables, and post-fault system models need to be estimated on-line in real-time. Based on the on-line information on the post-fault system model, the controller must be automatically reconfigured to maintain stability, desired dynamic performance and steady-state performance. To avoid potential actuator saturation and to take into consideration the degraded performance after fault occurrence, a command/reference governor may also need to be designed to adjust command input or reference trajectory automatically. Interested readers about FTCS may refer to the bibliographical review (Zhang & Jiang, 2008) and the recently published books (Noura et al., 2009) and (Edwards et al., 2010).

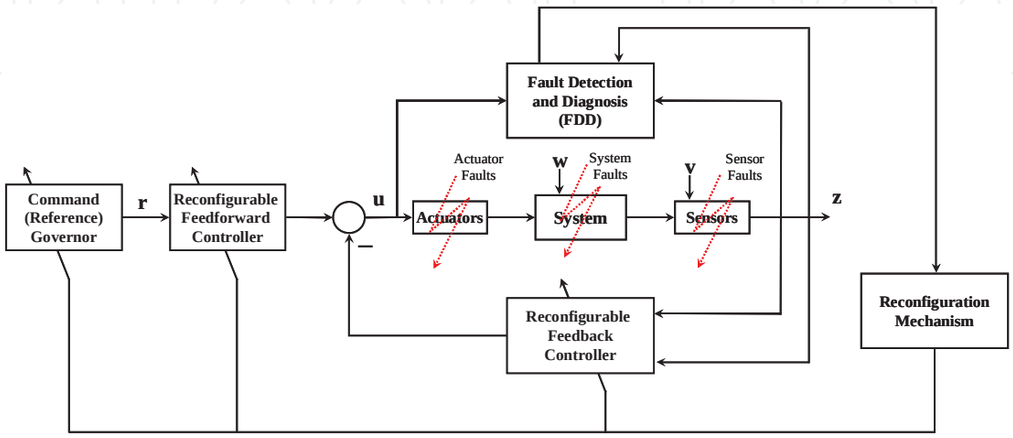


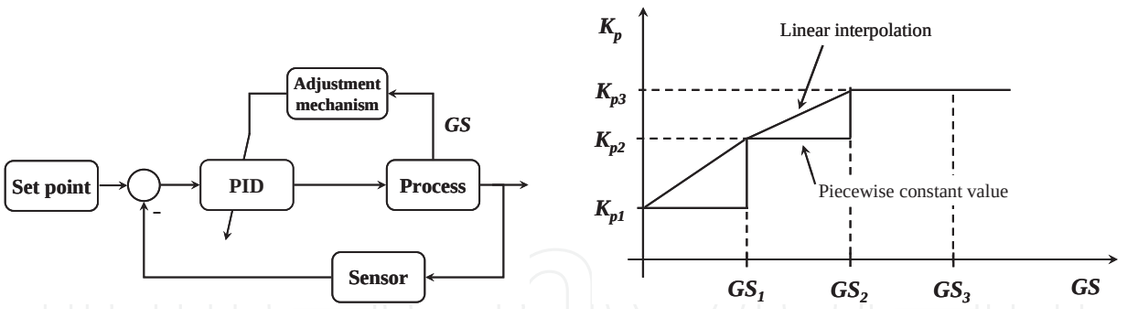
Fig. 3. A general structure of AFTCS (Zhang & Jiang, 2008).

The subsequent sections consider some of the FTC methods that have been developed for the quadrotor UAV. Some of these are classified as passive FTC methods where the fault tolerance is achieved thanks to the controller's robustness without changing the controller gains. Others are active FTC methods where control gains are updated in function of the occurring faults.

3.1 Gain-scheduled PID (GS-PID)

The Proportional-Integral-Derivative (PID) controller is the most widely used among controllers in industry. This is due to its unique features such as the simple structure, the ease of use and tunning. Its main advantages over other control strategies is that it does not require a mathematical model of the process/system to be controlled and thus allows to save time and effort by skipping the modeling phase. PID controllers are reliable and can be used for linear and nonlinear systems with certain level of robustness to model uncertainties and disturbances. Although one single PID controller can handle a wide range of system nonlinearities, better performance can be obtained when using multiple PIDs to cover the entire operation range of a nonlinear process/system. This is known as gain-scheduled PID (GS-PID) (Sadeghzadeh et al., 2011).

The operating principle of GS-PID is shown in Figure 4(a) where the controlled system may have varying dynamic properties as for example a varying gain (Haugen, 2004). The adjustment/scheduling of the PID controller gains is performed by using a gain scheduling variable GS. This is some measured process variable which at every instant of time expresses or represents the dynamic properties of the process.



(a) Operating principle of GS-PID (b) Constant and linear interpolation

Fig. 4. The operating principle of GS-PID and two rules for the interpolation of the proportional gain K_p.

There are several ways to express the PID parameters as functions of the GS variable such as the *piecewise constant controller parameters* and the *piecewise interpolation*. In the former method, an interval is defined around each GS value and the controller parameters are kept constant as long as the GS value is within the interval (see Figure 4(b) for an example of the proportional gain K_p). When the GS variable changes from one interval to another, the controller parameters are changed abruptly (Haugen, 2004). The same idea applies for the Integrator and Derivative gains. In the second method also shown in Figure 4(b), a linear function is found relating the controller parameter (output variable) and the GS variable (input variable). For the proportional gain, the linear function is of the form $K_p = aGS + b$ where a and b are two constants to be calculated.

Since faults can be seen as varying parameters, GS-PID can also be used to deal with possible fault conditions that may take place in the actuators or the system components. Some research works consider this problem: a GS-PID control strategy is proposed in (Bani Milhim, 2010a) and (Bani Milhim et al., 2010b) in simulation framework to achieve fault-tolerant control for the quadrotor helicopter UAV in the presence of actuator faults. In (Johnson et al., 2010), the authors investigate this technique for a Georgia Tech Twinstar fixed-wing research vehicle in the presence of partial damage of the wing. As continuation of the work presented in (Bani Milhim et al., 2010b), GS-PID has been considered for further investigation and most importantly for experimental implementation and application to the Qball-X4 UAV testbed for fault-tolerant trajectory tracking control (Sadeghzadeh et al., 2011). The GS-PID has been implemented for different sections of the entire flight envelope by properly tuning the PID controller gains for both normal and fault conditions. A set of PID controllers is then designed for the fault-free situation and each possible fault situation. The switching from one PID to another is then based on the actuator's health status (hence, the scheduling variable GS is the actuator's health status). A Fault Detection and Diagnosis (FDD) scheme is then needed to provide the time of fault occurrence as well as the location and the magnitude of the fault during the flight. Based on the information provided by the FDD module, the GS-PID controller will switch the controller gains under normal flight conditions to the pre-tuned and fault-related gains to handle the faults during the flight of the UAV. One of the main issues to consider in GS-PID is how fast to switch from the nominal PID gains to the pre-tuned fault-related gains after fault occurrence. This issue does not represent a problem when the scheduling variable GS is a measured variable since in most cases it is instantaneously provided by the sensors. However, when the GS is the health status of the actuators then

clearly, the switching time and gains depend on how fast and precise the FDD module is in detecting, isolating and identifying the faults. It is shown in Section 4 and through experimentations how the switching time affects the behavior of the system in handling the occurring faults.

3.2 Model reference adaptive control (MRAC)

Many adaptive control techniques for preserving stability have been proposed to deal with disturbances, unmodeled dynamics and time delays. Particularly, the concept of model reference adaptive control (MRAC) has gained significant attention and is now part of many standard textbooks on nonlinear and adaptive control. Two basic approaches for MRAC can be distinguished: the direct and the indirect approaches. In the direct method, the controller parameters are adjusted based on the error between the reference model describing the desired dynamics and the closed-loop dynamics of the physical plant. In the indirect approach, the parameters of the plant are estimated by updating them based on the identification error between the measured states and those provided by the estimation model. In this work, three MRAC techniques are implemented and applied to the Qball-X4, namely the MIT rule-based MRAC, the conventional MRAC and the modified MRAC (Chamseddine et al., 2011a).

3.2.1 MIT rule

This MRAC approach has been developed around 1960 at the Massachusetts Institute of Technology (MIT) for aerospace applications (Ioannou & Sun, 1995). For illustration, consider the plant:

$$\ddot{y}(t) = -a_1\dot{y}(t) - a_2 y(t) + bu(t) \tag{10}$$

where a_1, a_2 and b are the unknown plant parameters. The reference model to be matched by the closed-loop plant is:

$$y_m^{(3)}(t) = -a_{m_1}\ddot{y}_m(t) - a_{m_2}\dot{y}_m(t) - a_{m_3}y_m(t) + b_m r(t) \tag{11}$$

where $r(t)$ is the reference command and a_{m_i} $(i = 1, 2, 3)$ and b_m are constant and are chosen according to performance specifications. Let the control input $u(t)$ be defined as follows:

$$u(t) = -k_1\dot{y}(t) - k_2 y(t) - k_3 \int_0^t (y(\tau) - r(\tau))\, d\tau \tag{12}$$

By replacing (12) in (10) and differentiating with respect to time, one obtains:

$$y^{(3)}(t) = -(a_1 + bk_1)\,\ddot{y}(t) - (a_2 + bk_2)\,\dot{y}(t) - bk_3 y(t) + bk_3 r(t) \tag{13}$$

It is obvious that one can achieve perfect model following if k_1, k_2 and k_3 are chosen such that:

$$a_1 + bk_1 = a_{m_1},\ a_2 + bk_2 = a_{m_2} \ \text{ and } \ bk_3 = a_{m_3} = b_m \tag{14}$$

The control signal given by (12) cannot be implemented since the system parameters a_1, a_2 and b are assumed to be unknown. Nevertheless, one can use the following:

$$u(t) = -\hat{k}_1(t)\dot{y}(t) - \hat{k}_2(t) y(t) - \hat{k}_3(t) \int_0^t (y(\tau) - r(\tau))\, d\tau \tag{15}$$

where $\hat{k}_i$ are the estimates of k_i and are updated according to the MIT rule. The objective of the MIT rule is to adjust the parameters k_1, k_2 and k_3 so as to minimize a cost function J. This cost function can be chosen for example as follows:

$$J = \frac{1}{2}e^2 \tag{16}$$

where e is the tracking error between the system and the reference model, i.e. $e = y - y_m$. It is reasonable to adjust the parameters in the direction of the negative gradient of J:

$$\frac{d\hat{k}_i}{dt} = -\gamma \frac{\partial J}{\partial \hat{k}_i} = -\gamma e \frac{\partial e}{\partial \hat{k}_i} = -\gamma e \frac{\partial y}{\partial \hat{k}_i} \tag{17}$$

where $\gamma > 0$ is the adaptation rate. $\partial e/\partial \hat{k}_i$ is called the sensitivity derivative of the system and is evaluated under the assumption that $\hat{k}_i$ varies slowly. To calculate $\partial y/\partial \hat{k}_i$ in (17) one can use (13) to obtain:

$$\begin{aligned}
\frac{\partial y^{(3)}}{\partial \hat{k}_1} &= -a_1 \frac{\partial \ddot{y}}{\partial \hat{k}_1} - a_2 \frac{\partial \dot{y}}{\partial \hat{k}_1} - b\hat{k}_3 \frac{\partial y}{\partial \hat{k}_1} - b\ddot{y}(t) - b\hat{k}_1 \frac{\partial \ddot{y}}{\partial \hat{k}_1} - b\hat{k}_2 \frac{\partial \dot{y}}{\partial \hat{k}_1} \\
\frac{\partial y^{(3)}}{\partial \hat{k}_2} &= -a_1 \frac{\partial \ddot{y}}{\partial \hat{k}_2} - a_2 \frac{\partial \dot{y}}{\partial \hat{k}_2} - b\hat{k}_3 \frac{\partial y}{\partial \hat{k}_2} - b\dot{y}(t) - b\hat{k}_1 \frac{\partial \ddot{y}}{\partial \hat{k}_2} - b\hat{k}_2 \frac{\partial \dot{y}}{\partial \hat{k}_2} \\
\frac{\partial y^{(3)}}{\partial \hat{k}_3} &= -a_1 \frac{\partial \ddot{y}}{\partial \hat{k}_3} - a_2 \frac{\partial \dot{y}}{\partial \hat{k}_3} - b\hat{k}_3 \frac{\partial y}{\partial \hat{k}_3} - by(t) - b\hat{k}_1 \frac{\partial \ddot{y}}{\partial \hat{k}_3} - b\hat{k}_2 \frac{\partial \dot{y}}{\partial \hat{k}_3} + br(t)
\end{aligned} \tag{18}$$

With the assumption that the rate of adaptation is slow, i.e. $\hat{k}_i$ are small and the changes of $y^{(3)}$, $\ddot{y}$ and y with respect to $\hat{k}_1$, $\hat{k}_2$ and $\hat{k}_3$ are also small, one can interchange the order of differentiation to obtain:

$$\begin{aligned}
\frac{d^3}{dt^3}\frac{\partial y}{\partial \hat{k}_1} &= -(a_1 + b\hat{k}_1)\frac{d^2}{dt^2}\frac{\partial y}{\partial \hat{k}_1} - (a_2 + b\hat{k}_2)\frac{d}{dt}\frac{\partial y}{\partial \hat{k}_1} - b\hat{k}_3 \frac{\partial y}{\partial \hat{k}_1} - b\ddot{y}(t) \\
\frac{d^3}{dt^3}\frac{\partial y}{\partial \hat{k}_2} &= -(a_1 + b\hat{k}_1)\frac{d^2}{dt^2}\frac{\partial y}{\partial \hat{k}_2} - (a_2 + b\hat{k}_2)\frac{d}{dt}\frac{\partial y}{\partial \hat{k}_2} - b\hat{k}_3 \frac{\partial y}{\partial \hat{k}_2} - b\dot{y}(t) \\
\frac{d^3}{dt^3}\frac{\partial y}{\partial \hat{k}_3} &= -(a_1 + b\hat{k}_1)\frac{d^2}{dt^2}\frac{\partial y}{\partial \hat{k}_3} - (a_2 + b\hat{k}_2)\frac{d}{dt}\frac{\partial y}{\partial \hat{k}_3} - b\hat{k}_3 \frac{\partial y}{\partial \hat{k}_3} - by(t) + br(t)
\end{aligned} \tag{19}$$

These latter equations can be written as:

$$\begin{aligned}
\frac{\partial y}{\partial \hat{k}_1} &= \frac{-b}{p^3 + (a_1 + b\hat{k}_1)p^2 + (a_2 + b\hat{k}_2)p + b\hat{k}_3}\ddot{y}(t) \\
\frac{\partial y}{\partial \hat{k}_2} &= \frac{-b}{p^3 + (a_1 + b\hat{k}_1)p^2 + (a_2 + b\hat{k}_2)p + b\hat{k}_3}\dot{y}(t) \\
\frac{\partial y}{\partial \hat{k}_3} &= \frac{-b}{p^3 + (a_1 + b\hat{k}_1)p^2 + (a_2 + b\hat{k}_2)p + b\hat{k}_3}(y(t) - r(t))
\end{aligned} \tag{20}$$

where p is the differential operator. Because a_1, a_2 and b are unknown, the above sensitivity functions cannot be used. Using the MIT rule, we replace a_1, a_2 and b with their estimates $\hat{a}_1$,

$\hat{a}_2$ and $\hat{b}$ in the matching condition (14), i.e. we relate the estimates $\hat{a}_1$, $\hat{a}_2$ and $\hat{b}$ with $\hat{k}_1$, $\hat{k}_2$ and $\hat{k}_3$ using

$$\hat{a}_1 + \hat{b}\hat{k}_1 = a_{m_1},\ \hat{a}_2 + \hat{b}\hat{k}_2 = a_{m_2} \text{ and } \hat{b}\hat{k}_3 = a_{m_3} = b_m \tag{21}$$

and obtain the approximate sensitivity functions as:

$$\begin{aligned}
\frac{\partial y}{\partial \hat{k}_1} &\approx \frac{-b}{p^3 + a_{m_1}p^2 + a_{m_2}p + a_{m_3}}\ddot{y}(t) \\
\frac{\partial y}{\partial \hat{k}_2} &\approx \frac{-b}{p^3 + a_{m_1}p^2 + a_{m_2}p + a_{m_3}}\dot{y}(t) \\
\frac{\partial y}{\partial \hat{k}_3} &\approx \frac{-b}{p^3 + a_{m_1}p^2 + a_{m_2}p + a_{m_3}}(y(t) - r(t))
\end{aligned} \tag{22}$$

Finally, the adaptation of $\hat{k}_1$, $\hat{k}_2$ and $\hat{k}_3$ is:

$$\begin{aligned}
\frac{d\hat{k}_1}{dt} &= \gamma e \frac{1}{p^3 + a_{m_1}p^2 + a_{m_2}p + a_{m_3}}\ddot{y}(t) \\
\frac{d\hat{k}_2}{dt} &= \gamma e \frac{1}{p^3 + a_{m_1}p^2 + a_{m_2}p + a_{m_3}}\dot{y}(t) \\
\frac{d\hat{k}_3}{dt} &= \gamma e \frac{1}{p^3 + a_{m_1}p^2 + a_{m_2}p + a_{m_3}}(y(t) - r(t))
\end{aligned} \tag{23}$$

where the unknow parameter b in the numerator is absorbed by the gains $\hat{k}_i$.

The equations given by (22) are known as the *sensitivity filters or models*, and can be easily implemented to generate the approximate sensitivity functions for the adaptive law (23). It should be noted that the MRAC based on the MIT rule is locally stable provided the adaptive gain γ is small, the reference input signal r has a small amplitude and sufficient number of frequencies, and the initial conditions $\hat{k}_i(0)$ are close to the nominal values of k_i. For large γ and $\hat{k}_i(0)$ away from the nominal values of k_i, the MIT rule may lead to instability and unbounded signal response.

3.2.2 Conventional MRAC

Consider a multi-input multi-output (MIMO) uncertain linear system (Stepanyan & Krinshnakumar, 2010a):

$$\dot{x}(t) = Ax(t) + B\left[u(t) - f(t)\right],\ \ x(0) = x_0 \tag{24}$$

where $x \in \Re^n$ and $u \in \Re$ are the state and the control input of the system respectively. $f(t) \in \Re$ is a bounded and piece-wise continuous external disturbance. $A \in \Re^{n\times n}$ and $B \in \Re^{n\times p}$ are unknown constant matrices satisfying the following matching conditions.

Assumption 1. *Given a Hurwitz matrix $A_m \in \Re^{n\times n}$ and a matrix $B_m \in \Re^{n\times p}$ of full column rank, there exists a matrix $K_1 \in \Re^{p\times n}$ and a sign definite matrix $\Lambda \in \Re^{p\times p}$ such that the following equations hold*

$$\begin{aligned}
B &= B_m\Lambda \\
A &= A_m - BK_1
\end{aligned} \tag{25}$$

The sign definiteness of Λ corresponds to the conventional sign condition of the high frequency gain matrix of MIMO systems. Without loss of generality, we assume that Λ is positive definite. The rest of the conditions for the existence of an adaptive controller are given by (25).

The control objective is to design a control signal $u(t)$ such that the system tracks the reference model:

$$\dot{x}_m(t) = A_m x_m(t) + B_m N r(t), \;\; x_m(0) = x_0 \tag{26}$$

A_m and B_m are chosen according to performance specifications and satisfy Assumption 1 and $r(t)$ is a bounded and smooth external reference command. The matrix $N = -\left(CA_m^{-1}B_m^{-1}\right)^{-1}$ is chosen such that the output $y_m(t) = Cx_m(t)$ perfectly tracks the reference command $r(t)$. By using (25) and (26), one can note that system (24) can be written in the form:

$$\dot{x}(t) = A_m x(t) + B_m N r(t) + B_m \Lambda \left[u(t) - K_1 x(t) - K_2 r(t) - f(t)\right] \tag{27}$$

where $K_2 = \Lambda^{-1} N$. Hence, choosing the control input $u(t)$ as:

$$u(t) = K_1 x(t) + K_2 r(t) + f(t) \tag{28}$$

translates the system (24) into the reference model (26). The reference model (26) can always be specified from the performance perspective, however the control signal (28) cannot be implemented since the matrices K_1 and K_2 and the vector-function $f(t)$ are assumed to be unknown. Thus, the adaptive control is designed according to the MRAC architecture as:

$$u(t) = \hat{K}_1(t) x(t) + \hat{K}_2(t) r(t) + \hat{f}(t) \tag{29}$$

where $\hat{K}_1(t)$ and $\hat{K}_2(t)$ are the estimates of the ideal control gains K_1 and K_2, and $\hat{f}(t)$ is the estimate of a constant vector $\bar{f}$ that can be referred to as an average value of $f(t)$. These estimates are updated online according to robust adaptive laws:

$$\begin{aligned}
\dot{\hat{K}}_1(t) &= -\gamma B_m^T P \, e(t) x^T(t) + \Psi_1\left(x(t), e(t), \hat{K}_1(t)\right) \\
\dot{\hat{K}}_2(t) &= -\gamma B_m^T P \, e(t) r^T(t) + \Psi_2\left(r(t), e(t), \hat{K}_2(t)\right) \\
\dot{\hat{f}}(t) &= -\gamma B_m^T P \, e(t) + \Psi_3\left(e(t), \hat{f}(t)\right)
\end{aligned} \tag{30}$$

where $\gamma > 0$ is the adaptation rate, $P = P^T > 0$ is the solution of the Lyapunov equation:

$$A_m^T P + P A_m = -Q \tag{31}$$

for some $Q = Q^T > 0$. The terms Ψ_1, Ψ_2 and Ψ_3 represent the robust modifications such as σ-modification, e-modification, projection operator or dead-zone modification (Stepanyan & Krinshnakumar, 2010a). Here $e(t) = x(t) - x_m(t)$ is the tracking error between the system and the reference model.

3.2.3 Modified MRAC

The Modified MRAC (M-MRAC) is proposed in (Stepanyan & Krinshnakumar, 2010a;b). This approach is motivated by the fact that the initial large error in the control gains generates large transient excursions both in system's control and output signals. Therefore, driving the reference model toward the system proportional to the tracking error prevents the system's

attempt to aggressively maneuver toward the reference model. In this case, the modified reference model is defined as follows:

$$\dot{x}_m(t) = A_m x_m(t) + B_m N r(t) + \lambda(x(t) - x_m(t)),\ x_m(0) = x_0 \tag{32}$$

where $\lambda > 0$ is a design parameter that specifies the tracking error feedback into the reference model. As the tracking error approaches zero, the reference model (32) approaches its original form, which is called an ideal reference model:

$$\dot{x}^0(t) = A_m x^0(t) + B_m N r(t),\ \ x^0(0) = x_0 \tag{33}$$

The control input u as well as the adaptive laws for $\hat{K}_1$, $\hat{K}_2$ and $\hat{f}$ are the same as in the previous section. It has been shown that the error feedback term $\lambda(x(t) - x_m(t))$ determines the damping in the control signal. Increasing this term makes it possible to increase the learning rate for better transient performance without generating high frequency oscillations in the adaptive system.

3.3 Sliding mode control (SMC)

Closed-loop systems can have some fault-tolerance when the controller gains are carefully chosen to take care of effects of both faults and system uncertainties. Such systems are called passive fault-tolerant control (PFTC) systems. Passive approaches make use of robust control techniques to ensure that a closed-loop system remains insensitive to certain faults using constant controller parameters and without use of on-line fault information. In the case of PFTC, systems continue to operate with the same control gains and structures and the scheme effectiveness depends upon the robustness of the nominal (fault-free) closed-loop system. Systems are made robust to faults by assuming a restrictive repertoire of likely faults and the way in which they affect the control function.

Sliding mode control (SMC) techniques have a strong capability in handling uncertainties and disturbances, which makes them an excellent candidate for passive fault tolerant control system. The design of the SMC for the Qball-X4 starts by expressing the system model (4) in state-space representation (Li, 2011a), (Li et al., 2011b):

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \\ \dot{x}_5 \\ \dot{x}_6 \\ \dot{x}_7 \\ \dot{x}_8 \\ \dot{x}_9 \\ \dot{x}_{10} \\ \dot{x}_{11} \\ \dot{x}_{12} \end{bmatrix} = \begin{bmatrix} x_7 \\ x_8 \\ x_9 \\ x_{10} \\ x_{11} \\ x_{12} \\ u_z\left(cosx_5\ sinx_4\ cosx_6 + sinx_5\ sinx_6\right)/m - k_x x_7/m \\ u_z\left(cosx_5\ sinx_4\ sinx_6 - sinx_5\ cosx_6\right)/m - k_y x_8/m \\ u_z\left(cosx_5\ cosx_4\right)/m - g - k_z x_9/m \\ \left(u_\theta + \left(J_y - J_z\right)x_{11}x_{12} - J_T x_{11}\,\Omega - k_\theta x_{10}\right)/J_x \\ \left(u_\phi + \left(J_z - J_x\right)x_{10}x_{12} - J_T x_{10}\,\Omega - k_\phi x_{11}\right)/J_y \\ \left(u_\psi + \left(J_x - J_y\right)x_{10}x_{11} - k_\psi x_{12}\right)/J_z \end{bmatrix} \tag{34}$$

where $[x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_{10}, x_{11}, x_{12}]^T = [x, y, z, \theta, \phi, \psi, \dot{x}, \dot{y}, \dot{z}, \dot{\theta}, \dot{\phi}, \dot{\psi}]^T$ and u_z, u_θ, u_ϕ and u_ψ are defined in (9). In order to achieve tracking for the system, the tracking errors need to be defined as following where x_i^d is the desired path for the i^{th} state x_i:

$$e_i = x_i^d - x_i \quad ; \quad i = 1, ..., 6 \tag{35}$$

To guarantee the robustness of the passive fault tolerant control in both fault-free case and fault case, the faults need to be considered during the phase of controller design. Therefore, to design a controller that can handle both fault-free and fault situations, an integral sliding mode technique is employed to further enhance the robustness and to ensure a faster and smoother convergence. The sliding surface is then designed as:

$$s_i = \dot{e}_i + \lambda_i e_i + k_i \int e_i dt \quad ; \quad i = 1, ..., 6 \tag{36}$$

Consider the Lyapunov functions:

$$V_i = \frac{1}{2} s_i^2 \quad ; \quad i = 1, ..., 6 \tag{37}$$

The objective now is to design the control laws so that the time derivatives of the Lyapunov functions are negative. That is:

$$\dot{V}_i = \dot{s}_i s_i < 0 \quad ; \quad i = 1, ..., 6 \tag{38}$$

This latter condition states that the squared distance to the switching surface as measured by s^2 decreases along all system trajectories. However, this condition is not feasible in practice because the switching of real components is not instantaneous and this leads to an undesired phenomenon known as chattering in the direction of the switching surface. Thus, condition (38) is expanded by a boundary layer in which the controller switching is not required as:

$$\dot{s}_i s_i < -\eta \left| s \right| \quad ; \quad i = 1, ..., 6 \tag{39}$$

The control inputs should be chosen so that trajectories approach the sliding surface and then stay on it for all future time instants. Thus, the control inputs are expressed as the sum of two terms (Singh & Holé, 2004). The first one, called the equivalent control, is chosen so as to make $\dot{s}_i = 0$ when $s_i = 0$. Taking the time derivative of (36), the dynamics of the sliding surfaces can be obtained as following:

$$\begin{aligned}
\dot{s}_1 &= \ddot{x}_1^d - u_z u_x + k_x x_7/m + \lambda_1 \dot{e}_1 + k_1 e_1 \\
\dot{s}_2 &= \ddot{x}_2^d - u_z u_y + k_y x_8/m + \lambda_2 \dot{e}_2 + k_2 e_2 \\
\dot{s}_3 &= \ddot{x}_3^d - u_z \left(cosx_5 \; cosx_4 \right)/m + g + k_z x_9/m + \lambda_3 \dot{e}_3 + k_3 e_3 \\
\dot{s}_4 &= \ddot{x}_4^d - \left(u_\theta + \left(J_y - J_z \right) x_{11} x_{12} - J_T x_{11} \, \Omega - k_\theta x_{10} \right)/J_x + \lambda_4 \dot{e}_4 + k_4 e_4 \\
\dot{s}_5 &= \ddot{x}_5^d - \left(u_\phi + \left(J_z - J_x \right) x_{10} x_{12} - J_T x_{10} \, \Omega - k_\phi x_{11} \right)/J_y + \lambda_5 \dot{e}_5 + k_5 e_5 \\
\dot{s}_6 &= \ddot{x}_6^d - \left(u_\psi + \left(J_x - J_y \right) x_{10} x_{11} - k_\psi x_{12} \right)/J_z + \lambda_6 \dot{e}_6 + k_6 e_6
\end{aligned} \tag{40}$$

where $u_x = \left(cosx_5 \; sinx_4 \; cosx_6 + sinx_5 \; sinx_6 \right)/m$ and $u_y = \left(cosx_5 \; sinx_4 \; sinx_6 - sinx_5 \; cosx_6 \right)/m$ are defined as virtual inputs. By setting $\dot{s}_i = 0$ in

(40) the equivalent control inputs can be derived as:

$$\begin{aligned}
u_x^{eq} &= \frac{1}{u_z}\left(\ddot{x}_1^d + k_x x_7/m + \lambda_1\dot{e}_1 + k_1 e_1\right)\\
u_y^{eq} &= \frac{1}{u_z}\left(\ddot{x}_2^d + k_y x_8/m + \lambda_2\dot{e}_2 + k_2 e_2\right)\\
u_z^{eq} &= m\left(\ddot{x}_3^d + g + k_z x_9/m + \lambda_3\dot{e}_3 + k_3 e_3\right)/\left(cosx_5\ cosx_4\right)\\
u_\theta^{eq} &= J_x\left(\ddot{x}_4^d + \lambda_4\dot{e}_4 + k_4 e_4\right) - \left(J_y - J_z\right)x_{11}x_{12} + J_T x_{11}\,\Omega + k_\theta x_{10}\\
u_\phi^{eq} &= J_y\left(\ddot{x}_5^d + \lambda_5\dot{e}_5 + k_5 e_5\right) - \left(J_z - J_x\right)x_{10}x_{12} + J_T x_{10}\,\Omega + k_\phi x_{11}\\
u_\psi^{eq} &= J_z\left(\ddot{x}_6^d + \lambda_6\dot{e}_6 + k_6 e_6\right) - \left(J_x - J_y\right)x_{10}x_{11} + k_\psi x_{12}
\end{aligned} \tag{41}$$

The second term of the control inputs is chosen to tackle uncertainties in the system and to introduce reaching law. One can achieve a proportional $(l_i s_i)$ plus constant $(m_i sign(s_i))$ rate reaching law by selecting the second term as:

$$\begin{aligned}
u_x^* &= \frac{1}{u_z}\left(l_1 s_1 + m_1 sign(s_1)\right)\\
u_y^* &= \frac{1}{u_z}\left(l_2 s_2 + m_2 sign(s_2)\right)\\
u_z^* &= \frac{m}{\left(cosx_5\ cosx_4\right)}\left(l_3 s_3 + m_3 sign(s_3)\right)\\
u_\theta^* &= J_x\left(l_4 s_4 + m_4 sign(s_4)\right)\\
u_\phi^* &= J_y\left(l_5 s_5 + m_5 sign(s_5)\right)\\
u_\psi^* &= J_z\left(l_6 s_6 + m_6 sign(s_6)\right)
\end{aligned} \tag{42}$$

where l_i and m_i $(i = 1, ..., 6)$ are positive control gains. The total control inputs $u = u^{eq} + u^*$ are then (Li, 2011a), (Li et al., 2011b):

$$\begin{aligned}
u_x &= \frac{1}{u_z}\left(\ddot{x}_1^d + k_x x_7/m + \lambda_1\dot{e}_1 + k_1 e_1 + l_1 s_1 + m_1 sign(s_1)\right)\\
u_y &= \frac{1}{u_z}\left(\ddot{x}_2^d + k_y x_8/m + \lambda_2\dot{e}_2 + k_2 e_2 + l_2 s_2 + m_2 sign(s_2)\right)\\
u_z &= \frac{m}{\left(cosx_5\ cosx_4\right)}\left(\ddot{x}_3^d + g + k_z x_9/m + \lambda_3\dot{e}_3 + k_3 e_3 + l_3 s_3 + m_3 sign(s_3)\right)\\
u_\theta &= J_x\left(\ddot{x}_4^d + \lambda_4\dot{e}_4 + k_4 e_4 + l_4 s_4 + m_4 sign(s_4)\right) - \left(J_y - J_z\right)x_{11}x_{12} + J_T x_{11}\,\Omega + k_\theta x_{10}\\
u_\phi &= J_y\left(\ddot{x}_5^d + \lambda_5\dot{e}_5 + k_5 e_5 + l_5 s_5 + m_5 sign(s_5)\right) - \left(J_z - J_x\right)x_{10}x_{12} + J_T x_{10}\,\Omega + k_\phi x_{11}\\
u_\psi &= J_z\left(\ddot{x}_6^d + \lambda_6\dot{e}_6 + k_6 e_6 + l_6 s_6 + m_6 sign(s_6)\right) - \left(J_x - J_y\right)x_{10}x_{11} + k_\psi x_{12}
\end{aligned} \tag{43}$$

It is straightforward to verify the negativity of condition (39). For illustration purpose, taking for example u_x of (43) and plugging it into $\dot{s}_1$ of (40) gives:

$$\dot{s}_1 = -l_1 s_1 - m_1 sign(s_1) \tag{44}$$

Multiplying both sides of the previous equation by s_1:

$$s_1 \dot{s}_1 = -l_1 s_1^2 - m_1 |s_1| \tag{45}$$

and properly choosing control gains l_1 and m_1 allows to verify (39). The same procedure can be followed for the other control inputs.

The elimination of the chattering effect produced by the discontinuous function *sign* can be attained by using a saturation function *sat* instead of the *sign* function. This saturation function is defined as follows:

$$sat(s_i) = \begin{cases} \delta_b & \text{if} \quad s_i \geq \delta_b \\ s_i & \text{if} \quad -\delta_b < s_i < \delta_b \\ -\delta_b & \text{if} \quad s_i \leq -\delta_b \end{cases} \tag{46}$$

where δ_b is the boundary of the saturation function and is set small enough.

3.4 Backstepping control (BSC)

Backstepping design refers to "step back" to the control input, and a major advantage of backstepping design is its flexibility to avoid cancellation of useful nonlinearities and pursue the objectives of stabilization and tracking, rather than those of linearization. Recursively constructed backstepping controller (BSC) employs the control Lyapunov function to guarantee the global stability (Krstic et al., 1995).

In order to illustrate the BSC techniques, consider the following nonlinear system:

$$\begin{aligned} \dot{x} &= f(x) + g(x)\xi \\ \dot{\xi} &= u \end{aligned} \tag{47}$$

where x is the state vector and ξ is the control input. As a first step, define the tracking error between the actual value x and its desired value x^d as follows:

$$e = x^d - x \tag{48}$$

As a second step, define the following states:

$$\begin{aligned} x_1 &= x \\ x_2 &= \dot{x}_1 = \dot{x} \\ z_1 &= e = x^d - x = x^d - x_1 \end{aligned} \tag{49}$$

The key idea of BSC is to choose certain variables as virtual controls. Assuming x_2 is the virtual control variable and $a(z_1)$ is the function which makes $z_1 \to 0$, define $\bar{z}_1$ as:

$$\bar{z}_1 = a(z_1) - x_2 \tag{50}$$

The time derivative of z_1 is:

$$\dot{z}_1 = \dot{x}^d - \dot{x}_1 = \dot{x}^d - x_2 = \dot{x}^d + \bar{z}_1 - a(z_1) \tag{51}$$

Let us choose $a(z_1)$ as $a(z_1) = \dot{x}^d + k_1 z_1$ with $k_1 > 0$ and define the Lyapunov function:

$$V_1 = \frac{1}{2} z_1^2 \tag{52}$$

One can see that the time derivative of V_1 is:

$$\dot{V}_1 = z_1 \dot{z}_1 = z_1 \left(\dot{x}^d + \bar{z}_1 - a(z_1) \right) = z_1 \left(\bar{z}_1 - k_1 z_1 \right) = -k_1 z_1^2 + z_1 \bar{z}_1 \tag{53}$$

Clearly if $\bar{z}_1 = 0$, then $\dot{V}_1 = -k_1 z_1^2$ and z_1 is guaranteed to converge to zero asymptotically. Then, the next step is to define a Lyapunov function so as to make $\bar{z}_1 \to 0$:

$$V_2 = V_1 + \frac{1}{2} \bar{z}_1^2 \tag{54}$$

The time derivative of V_2 is:

$$\dot{V}_2 = \dot{V}_1 + \bar{z}_1 \dot{\bar{z}}_1 = -k_1 z_1^2 + z_1 \bar{z}_1 + \bar{z}_1 \left(\ddot{x}^d + k_1 \dot{z}_1 - \dot{x}_2 \right) \tag{55}$$

To obtain $\dot{V}_2 < 0$, $\dot{x}_2$ is chosen as:

$$\dot{x}_2 = \dot{x}^d + \left(1 - k_1^2 \right) z_1 + \left(k_1 + k_2 \right) \bar{z}_1 \tag{56}$$

with $k_1 > 0$, $k_2 > 0$, $z_1 = x^d - x_1$ and $\bar{z}_1 = \dot{x}^d + k_1 z_1 - x_2$.

The application of the method above to the state-space representation (34) of the Qball-X4 model gives the control inputs (Zhang, 2010a), (Zhang et al., 2010b):

$$\begin{aligned}
u_x &= \frac{1}{u_z} \left(\ddot{x}_1^d + k_x x_7 / m + \left(1 - k_1^2 \right) z_1 + \left(k_1 + k_2 \right) \bar{z}_1 \right) \\
u_y &= \frac{1}{u_z} \left(\ddot{x}_2^d + k_y x_8 / m + \left(1 - k_1^2 \right) z_2 + \left(k_1 + k_2 \right) \bar{z}_2 \right) \\
u_z &= \frac{m}{(cosx_5 \; cosx_4)} \left(\ddot{x}_3^d + g + k_z x_9 / m + \left(1 - k_1^2 \right) z_3 + \left(k_1 + k_2 \right) \bar{z}_3 \right) \\
u_\theta &= J_x \left(\ddot{x}_4^d + \left(1 - k_1^2 \right) z_4 + \left(k_1 + k_2 \right) \bar{z}_4 \right) - \left(J_y - J_z \right) x_{11} x_{12} + J_T x_{11} \, \Omega + k_\theta x_{10} \\
u_\phi &= J_y \left(\ddot{x}_5^d + \left(1 - k_1^2 \right) z_5 + \left(k_1 + k_2 \right) \bar{z}_5 \right) - \left(J_z - J_x \right) x_{10} x_{12} + J_T x_{10} \, \Omega + k_\phi x_{11} \\
u_\psi &= J_z \left(\ddot{x}_6^d + \left(1 - k_1^2 \right) z_6 + \left(k_1 + k_2 \right) \bar{z}_6 \right) - \left(J_x - J_y \right) x_{10} x_{11} + k_\psi x_{12}
\end{aligned} \tag{57}$$

with $u_x = \left(cosx_5 \; sinx_4 \; cosx_6 + sinx_5 \; sinx_6 \right) / m$ and $u_y = \left(cosx_5 \; sinx_4 \; sinx_6 - sinx_5 \; cosx_6 \right) / m$ defined as virtual inputs and $z_i = x_i^d - x_i$ and $\bar{z}_i = \dot{x}_i^d + k_1 z_i - \dot{x}_i$ $(i = 1, ..., 6)$.

3.5 Model predictive control (MPC)

Model Predictive Control (MPC) is a promising tool for fault tolerant control applications (Maciejowski & Jones, 2003) due to its prominent capabilities such as constraint handling, flexibility to changes in the process dynamics and applicability to nonlinear dynamics. Since MPC recalculates the control signal at each sampling time, any change in the process dynamics can be reflected simply into the control signal calculation. Also, the constraint handling capability of MPC allows system working close to the boundaries of the tight post-failure operation envelope. The drawback of MPC is that similarly to most of the control techniques, it needs an almost explicit model of the system to calculate a stabilizing control signal. On the other hand, the abrupt changes in the model parameters, due to failure, cannot be predicted beforehand and an online data-driven parameter estimation methodology is required to extract the post-failure model parameters from online input/output data. In other words, an FDD module is required to provide information about the occurring faults to allow MPC to consider faults.

3.5.1 Notation and terminology

In the MPC, also known as Receding Horizon Control (RHC), a cost function is minimized over a future prediction horizon time step denoted by N, subject to the dynamical constraints. The first control input in the sequence is applied to the plant until the next update is available. The discrete timing is shown by k where $k \in \mathbb{N}$. The state vectors are introduced as follows:

- $x(k)$: actual state vector at time step k.
- $x_k(j)$: predicted state vector at time step $k+j$ computed at time step k $(j \in \{0, 1, ..., N\})$.

Similar notation is used for the control input vector u. Also the sequence of the state/input vector over the prediction horizon is called the state/input trajectory and is shown as follows:

$$\begin{aligned} x_k(.) &= \{x_k(j) | j = 0, 1, 2, ..., N\} \\ u_k(.) &= \{u_k(j) | j = 0, 1, 2, ..., N-1\} \end{aligned} \tag{58}$$

3.5.2 Fault-free MPC formulation

The cost function at time step k is defined as follows:

$$J(x_k(.), u_k(.)) = \sum_{j=0}^{N-1} \left(\|x_k(j) - x^d\|_Q^2 + \|u_k(j)\|_R^2 \right) + \|x_k(N) - x^d\|_P^2 \tag{59}$$

where $\|x\|_Q^2 = x^T Q x$, and $P > 0$, $Q > 0$ and $R > 0$ are symmetric matrices. x^d is the desired or reference state of x. Then the MPC problem at each step time k is defined as follows:

Problem 1. *Calculate:*

$$J^*(x(k)) = \min_{\{u_k(.), x_k(.)\}} \quad J(x_k(.), u_k(.)) \tag{60}$$

Subject to:

$$x_k(j+1) = f(x_k(j), u_k(j)) \quad ; \quad x_k(0) = x(k) \tag{61}$$

$$x_k(j) \in \mathbb{X} \tag{62}$$

$$u_k(j) \in \mathbb{U} \tag{63}$$

$$x_k(N) \in \mathbb{X}_t \tag{64}$$

for $j = 0, 1, ..., N-1$.

$\mathbb{X} \subseteq \mathbb{R}^n$, $\mathbb{U} \subseteq \mathbb{R}^m$ and $\mathbb{X}_t \subseteq \mathbb{X}$ denote the set of admissible states, inputs and terminal states (terminal region) respectively. J^* denotes the optimal value of the cost function J. This MPC formulation is based on the quasi-infinite model predictive control (Chen & Allgower, 1998). At each time step k, MPC generates the input and state trajectories, by solving Problem 1. After generating these trajectories, MPC controller applies only the first computed control input, i.e. $u_k(0)$ to the system. The following algorithm presents the online implementation of Problem 1 (Izadi, 2009a), (Izadi et al., 2011b).

Algorithm 1. *Given* $x(0)$ *and* x^d*, do:*

1. $k = 0$.
2. *Measure* $x(k)$.

3. *Solve Problem 1 and generate* $u_k(.)$ *and* $x_k(.)$.
4. *Apply* $u_k(0)$ *to the system.*
5. *Set* $k = k + 1$ *and GOTO Step #2.*

This algorithm is repeated for $k = 0, 1, ..., \infty$. In step #2, if full state measurement is not available, then a state estimator (observer) can be used to provide the state estimate.

3.5.3 Fault-tolerant MPC

In the presence of an actuator fault, the system dynamics can be rewritten as follows:

$$x(k+1) = f(x(k), \alpha(k)u(k)) \tag{65}$$

where α captures the fault information and is called the fault parameter matrix which determines the fault severity. α is a diagonal matrix, i.e. $\alpha = diag(\alpha_1, \alpha_2, ..., \alpha_m)$ where m is the number of inputs ($m = 4$ for the quadrotor helicopter) and the scalar α_i denotes the amplitude of the fault in the i^{th} actuator. $\alpha_i = 1$ denotes a healthy actuator, $\alpha_i = 0$ denotes a complete loss of actuator effectiveness and $0 < \alpha_i < 1$ represents a partial loss. Taking into consideration the model representation (65) after fault occurrence, the fault-tolerant MPC problem can be defined as follows (Izadi et al., 2011b):

Problem 2. *Calculate:*

$$J^*(x(k)) = \min_{\{u_k(.), x_k(.)\}} J(x_k(.), u_k(.)) \tag{66}$$

Subject to:

$$x_k(j+1) = f(x_k(j), \alpha u_k(j)) \quad ; \quad x_k(0) = x(k) \tag{67}$$

$$x_k(j) \in \mathbb{X} \tag{68}$$

$$u_k(j) \in \mathbb{U} \tag{69}$$

$$x_k(N) \in \mathbb{X}_t \tag{70}$$

for $j = 0, 1, ..., N-1$.

One can see that the difference between Problem 1 and Problem 2 resides in the consideration of the fault matrix α in (67). Clearly, solving Problem 2 requires an FDD module for fault identification and thus the algorithm for solving Problem 2 is similar to Algorithm1 except that an FDD module is needed to run parallelly to provide an estimation of α.

3.6 Flatness-based trajectory planning/re-planning (FTPR)

This section does not present an FTC method but a trajectory planning/re-planning approach that can be combined with any FTC module to provide a better management of the system resources after fault occurrence. This is achieved by redefining the reference trajectories to be followed by the damaged system depending on the fault severity. This approach is equivalent to the reconfiguration of command governor block in Figure 3.

3.6.1 Flatness notion

The flatness can be defined as follows. A dynamic system:

$$\begin{aligned} \dot{x} &= f(x,u) \\ y &= h(x) \end{aligned} \tag{71}$$

with $x \in \Re^n$ and $u \in \Re^m$, is flat if and only if there exist variables $F \in \Re^m$ called the flat outputs such that: $x = \Xi_1(F, \dot{F}, ..., F^{(n-1)})$, $y = \Xi_2(F, \dot{F}, ..., F^{(n-1)})$ and $u = \Xi_3(F, \dot{F}, ..., F^{(n)})$. Ξ_1, Ξ_2 and Ξ_3 are three smooth mappings and $F^{(i)}$ is the i^{th} derivative of F. The parameterization of the control inputs u in function of the flat outputs F plays a key role in the trajectory planning problem: the nominal control inputs to be applied during a mission can be expressed in function of the desired trajectories. Thus allowing to tune the profile of the trajectories for keeping the applied control inputs below the actuator limits (Chamseddine et al., 2011b).

For the quadrotor UAV simplified model given in (6), one can see that the outputs to be controlled can be chosen as flat outputs. Thus, system (6) is flat with flat outputs $F_1 = z$, $F_2 = x$, $F_3 = y$ and $F_4 = \psi$. In addition to x, y, z and ψ, the parameterization of θ and ϕ in function of the flat outputs is:

$$\begin{aligned} \theta &= \frac{\ddot{F}_2}{g}; \\ \phi &= -\frac{\ddot{F}_3}{g} \end{aligned} \tag{72}$$

The parameterization of the control inputs in function of the flat outputs is:

$$\begin{aligned} u_z &= m\left(\ddot{F}_1 + g\right); \\ u_\theta &= J_x \frac{F_2^{(4)}}{g}; \\ u_\phi &= -J_y \frac{F_3^{(4)}}{g}; \\ u_\psi &= J_z \ddot{F}_4 \end{aligned} \tag{73}$$

3.6.2 Reference trajectory design

Let's define F_i^* as the reference trajectories for the flat output F_i. Several methods can be used to design F_i^*. In this work we employ the Bézier polynomial function. A general Bézier polynomial function of degree n is:

$$F = a_n t^n + a_{n-1} t^{n-1} + ... + a_2 t^2 + a_1 t + a_0 \tag{74}$$

where t is the time and a_i $(i = 0, ..., n)$ are constant coefficients to be calculated in function of the initial and final conditions. It is clear that the larger is n, the smoother is the reference trajectory. However, calculations for trajectory planning become heavier as n increases. For the quad-rotor UAV, it can be seen in (73) that u_z is function of $\ddot{F}_1$, u_θ of $F_2^{(4)}$, u_ϕ of $F_3^{(4)}$ and

u_ψ of $\ddot{F}_4$. The relative degrees are then $r_1 = 2$, $r_2 = 4$, $r_3 = 4$ and $r_4 = 2$. Smooth control inputs can be obtained if one can impose $F_i^{(0)}$ up to $F_i^{(n_i)}$ at the initial and final time (i.e. up to $F_i^{(n_i)}(t_0)$ and $F_i^{(n_i)}(t_f)$) where $n_i = 2$ for $i = 1, 4$ and $n_i = 4$ for $i = 2, 3$. To this end, we employ a Bézier polynomial function of degree 5 for F_1 and F_4 and a Bézier polynomial function of degree 9 for F_2 and F_3. The reference trajectories are then:

$$F_i^* = a_5^i t^5 + a_4^i t^4 + a_3^i t^3 + a_2^i t^2 + a_1^i t + a_0^i \ ; \ (i = 1, 4) \tag{75}$$

and

$$F_i^* = a_9^i t^9 + a_8^i t^8 + \cdots + a_2^i t^2 + a_1^i t + a_0^i \ ; \ (i = 2, 3) \tag{76}$$

The coefficients a_j^i ($i = 1, 4$ and $j = 0, ..., 5$) are calculated to verify the initial conditions $F_i(t_0)$, $\dot{F}_i(t_0)$, $\ddot{F}_i(t_0)$ and the final conditions $F_i(t_f)$, $\dot{F}_i(t_f)$, $\ddot{F}_i(t_f)$. The coefficients a_j^i ($i = 2, 3$ and $j = 0, ..., 9$) are calculated to verify the initial conditions $F_i(t_0)$, $\dot{F}_i(t_0)$, $\ddot{F}_i(t_0)$, $F_i^{(3)}(t_0)$, $F_i^{(4)}(t_0)$ and the final conditions $F_i(t_f)$, $\dot{F}_i(t_f)$, $\ddot{F}_i(t_f)$, $F_i^{(3)}(t_f)$, $F_i^{(4)}(t_f)$. t_0 and t_f are respectively the initial and the final instants of the mission.

3.6.3 Trajectory planning

The trajectory planning consists in driving the quadrotor from an initial to a final position. The initial and final conditions as well as the initial time t_0 are all known and the only unknown is the final time of the mission t_f. Thus, the trajectory planning tends to tune the profile of the trajectory (by tuning t_f) so that to drive the system along the desired trajectory without violating actuator constraints. Thanks to flatness, it is possible to find the relation between the applied control inputs and the reference trajectories. According to (73) the nominal control inputs to be applied along the reference trajectories are:

$$\begin{aligned} u_z^* &= m\left(\ddot{F}_1^* + g\right); \\ u_\theta^* &= J_x \frac{F_2^{(4)*}}{g}; \\ u_\phi^* &= -J_y \frac{F_3^{(4)*}}{g}; \\ u_\psi^* &= J_z \ddot{F}_4^* \end{aligned} \tag{77}$$

Since the quadrotor is assumed to have a zero yaw angle ($\psi = 0$), then $\ddot{F}_4^* = 0 \ \forall \, t \in [t_0, t_f]$. It is also assumed that the quadrotor is not changing altitude and thus $\ddot{F}_1^* = 0 \ \forall \, t \in [t_0, t_f]$. In this case, the nominal control inputs are:

$$\begin{aligned} u_z^* &= mg; \\ u_\theta^* &= J_x \frac{F_2^{(4)*}}{g}; \\ u_\phi^* &= -J_y \frac{F_3^{(4)*}}{g}; \\ u_\psi^* &= 0 \end{aligned} \tag{78}$$

The system is required to move from an initial position $F_i(t_0) = 0$ with initial conditions $\dot{F}_i(t_0) = \ddot{F}_i(t_0) = F_i^{(3)}(t_0) = F_i^{(4)}(t_0) = 0$ to a final position $F_i(t_f) \neq 0$ with final conditions $\dot{F}_i(t_f) = \ddot{F}_i(t_f) = F_i^{(3)}(t_f) = F_i^{(4)}(t_f) = 0$ $(i = 2,3)$. Without loss of generality, by setting $t_0 = 0$ one can write (76) as

$$F_i^* = 70F_i(t_f)\frac{t^9}{t_f^9} - 315F_i(t_f)\frac{t^8}{t_f^8} + 540F_i(t_f)\frac{t^7}{t_f^7} - 420F_i(t_f)\frac{t^6}{t_f^6} + 126F_i(t_f)\frac{t^5}{t_f^5} \;\; ; \;\; (i = 2,3) \quad (79)$$

Thus, it is possible to calculate $u_\theta^* = J_x F_2^{(4)*}/g$ and $u_\phi^* = -J_y F_3^{(4)*}/g$ since

$$F_i^{(4)*} = c_1 F_i(t_f)\frac{t^5}{t_f^9} - c_2 F_i(t_f)\frac{t^4}{t_f^8} + c_3 F_i(t_f)\frac{t^3}{t_f^7} - c_4 F_i(t_f)\frac{t^2}{t_f^6} + c_5 F_i(t_f)\frac{t}{t_f^5} \;\; ; \;\; (i = 2,3) \quad (80)$$

where c_j $(j = 1,...,5)$ are constants that can be easily derived from (79) and are not given here for simplicity. From (9), (78) and (80), it is possible to determine the nominal PWM inputs to be applied along the references trajectories, which are:

$$u_{1/2}^* = \frac{mg}{4K} \pm \frac{F_2(t_f)J_x\Delta}{2gKLt_f^9} \;\text{ and }\; u_{4/3}^* = \frac{mg}{4K} \pm \frac{F_3(t_f)J_y\Delta}{2gKLt_f^9} \quad (81)$$

where $\Delta = \left(c_1 t^5 - c_2 t_f t^4 + c_3 t_f^2 t^3 - c_4 t_f^3 t^2 + c_5 t_f^4 t\right)$. Since $F_2(t_f)$ and $F_3(t_f)$ are known, one must find the minimal t_f so that $u_i^* < u_{max}$ where u_{max} is the maximal PWM that corresponds to the maximal thrust that can be generated by a rotor. One way to determine t_f is to analytically calculate when the extrema of u_i^* take place and what are their values (where the extrema collectively denote the maxima and the minima of a function). For this purpose, it is necessary to calculate the derivative of u_i^* in (81) with respect to time and then solve for t the following equation:

$$\frac{du_i^*}{dt} = 0 \;\; ; \;\; (i = 1,...,4) \quad (82)$$

For a fixed i $(i = 1,...,4)$, each equation du_i^*/dt is a polynomial function of fourth degree and thus has four extrema which take place at:

$$\frac{t_f}{2}\left(1 \pm \sqrt{\frac{3}{7} - \frac{2}{35}\sqrt{30}}\right) \;\text{ and }\; \frac{t_f}{2}\left(1 \pm \sqrt{\frac{3}{7} + \frac{2}{35}\sqrt{30}}\right) \quad (83)$$

The values of the four extrema can be obtained by replacing t of (81) by the values obtained in (83). As an example, the values of the four extrema of u_1^* of the first rotor are:

$$u_{1_{Ext}}^* = \left(\frac{mg}{4K} \pm c_6\frac{F_2(t_f)J_x}{gKLt_f^4}; \frac{mg}{4K} \pm c_7\frac{F_2(t_f)J_x}{gKLt_f^4}\right) \quad (84)$$

with c_6 and c_7 two constants. The determination of t_f consists finally in solving (84) so that the extrema of the nominal PWM input verifies:

$$u_{1_{Ext}}^* \leq \rho u_{max} \quad (85)$$

Since the above study is based on the simplified model (6), the constant $0 \leq \rho \leq 1$ is introduced to create some safety margin and to robustify the obtained solution with respect to model uncertainties. Four solutions can then be obtained:

$$t_f \geq \left(\left(\pm c_6 \frac{F_2(t_f) J_x}{gKL\left(\rho u_{max} - \frac{mg}{4K}\right)} \right)^{1/4} ; \left(\pm c_7 \frac{F_2(t_f) J_x}{gKL\left(\rho u_{max} - \frac{mg}{4K}\right)} \right)^{1/4} \right) \tag{86}$$

The four extrema of u_2^*, u_3^* and u_4^* have the same structure as for u_1^* with different values for the constants c_6 and c_7. In conclusion, the quadrotor UAV will have 16 extrema (four per motor) and therefore 16 solutions are obtained and the maximal one among them is to be considered.

3.6.4 Trajectory re-planning

As for the fault-free case, the trajectory re-planning consists in determining the minimal time of the mission t_f so that the actuator constraints are not violated. For the damaged UAV, this is written as:

$$u_i^* \leq \rho(1 - \delta_i) u_{max} \;\; ; \;\; (i = 1, ..., 4) \tag{87}$$

where δ_i represents the loss of effectiveness in the i^{th} rotor. $\delta_i = 0$ denotes a healthy rotor, $\delta_i = 1$ denotes a complete loss of the rotor and $0 < \delta_i < 1$ represents a partial loss of control effectiveness. Clearly, trajectory re-planning in the fault case requires the knowledge of the fault amplitude δ_i. Thus, a fault detection and diagnosis module is needed to detect, isolate and identify the fault. Unlike trajectory planning, the initial conditions are not zero at the re-planning instant i.e. $\dot{F}_i(t_{rep}) \neq 0, \ddot{F}_i(t_{rep}) \neq 0, F_i^{(3)}(t_{rep}) \neq 0$ and $F_i^{(4)}(t_{rep}) \neq 0$ where t_{rep} is the re-planning time. Therefore, it is difficult to analytically calculate t_f as in the previous section since the expressions are much more complicated and thus another method is needed to solve the problem. Starting from the idea that the larger t_f is, the smaller are the PWM inputs u_i, we propose the following algorithm to calculate t_f:

1. Develop the expressions of $u_i^*(t)$ (similar to the procedure in the previous section) with nonzero initial conditions.
2. Start with an initial guess of the mission time t_f.
3. For $t = 0 : T_e : t_f$ calculate $u_i^*(t)$ and determine the maximum value $u_{i_{max}}^*$ among all the $u_i^*(t)$. T_e is a sampling period.
4. Determine the error E_i between $u_{i_{max}}^*$ and the maximal allowable PWM input (ρu_{max} for the healthy UAVs and $\rho(1 - \delta_i) u_{max}$ for the damaged one).
5. If the error is smaller than a predefined threshold, exit the algorithm and the solution is the current t_f. If not, update the current t_f as follows $t_f \leftarrow t_f + t_f E/\nu$ with ν is a positive constant.
6. Repeat from Step #3.

As can be seen in Step #5, if the error E_i is positive then $u_{i_{max}}^*$ is larger than the maximal allowable PWM input and thus it is necessary to increase t_f. If the error is negative then $u_{i_{max}}^*$ is smaller than the maximal allowable PWM input and thus it is necessary to decrease t_f. The constant ν must be carefully selected since it affects the convergence speed of the algorithm.

4. Simulation and experimental testing results

The approaches that have been experimentally tested on the Qball-X4 are built using Matlab/Simulink and downloaded on the Gumstix emdedded computer to be run on-board with a frequency of 200 Hz. The experiments are taking place indoor in the absence of GPS signals and thus the OptiTrack camera system from NaturalPoint is employed to provide the system position in the 3D space. Some photos of the NAV Lab of Concordia University are shown in Figure 5 and many videos related to the above approaches as well as other videos can be watched online on `http://www.youtube.com/user/NAVConcordia`.



Fig. 5. The NAV Lab of Concordia University.

4.1 GS-PID experimental results

Starting with the GS-PID, an 18% loss of overall power of all motors is considered where the Qball-X4 is requested to track a one meter square trajectory tracking. The fault-free case is shown in the left-hand side plot of Figure 6 whereas the middle one shows a deviation from the desired trajectory after the fault occurrence when the switching between the PID gains is taking place after 0.5 s of fault occurrence. Better tracking performance can be achieved with shorter switching time which requires fast and correct fault detection. The right-hand plot of Figure 6 demonstrates better performance when the switching is done without time delay.

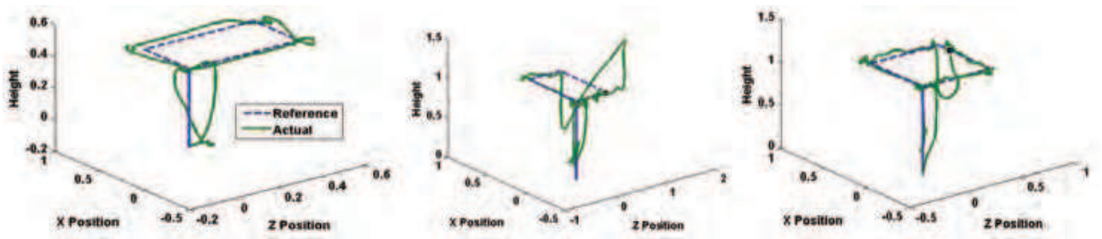


Fig. 6. Fault-free case and 18% loss in the total thrust with and without time delay.

4.2 MRAC experimental results

As for the MRAC, a partial effectiveness loss of 30% and 40% are simulated in the total thrust u_z. The faults are injected at $t = 40$ s and the experimental results are illustrated in Figure 7. This kind of fault induces a loss in the height z without significant effect on x, y or ψ directions. One can see that all the three MRAC techniques give better performance than the baseline LQR controller and that the conventional MRAC (C-MRAC) is the best among the MRAC techniques.

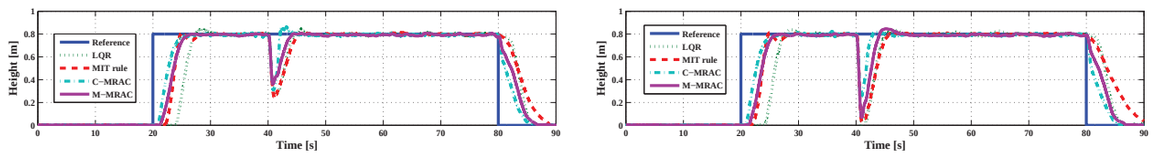


Fig. 7. System behavior with a simulated fault of 30% and 40% loss in the total thrust.

The MRAC has been also tested in the presence of a partial damage of the 4^{th} propeller. This type of faults is more realistic than the simulated faults given above. To accommodate a partial damage of the propeller, the controller tries to speed up the 4^{th} rotor so that to produce the same lift as for the fault-free case. Thus, the maximal speed that a rotor can reach is a critical factor that does not play a role in simulated faults. Figure 8 shows the system behavior along the z and y directions where the first row corresponds to 12% damage and the second to 16%. One can see that the performance obtained with the C-MRAC is better than that of the LQR.

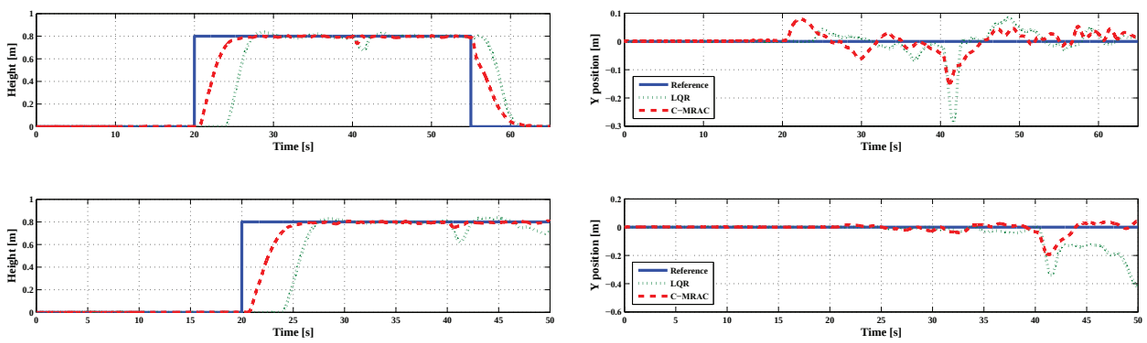


Fig. 8. System behavior with a damage of 12% and 16% of the fourth propeller.

4.3 SMC experimental results

The objective in the SMC is to track a square trajectory of 1.5 $m \times 1.5$ m. Figure 9 shows the system evolution along the x, y and z directions when the SMC-based PFTC is experimentally applied to the Qball-X4. The SMC-based PFTC is giving good performance where very small deviation in position z can be observed at 20 s due to a partial damage in the 4^{th} propeller.

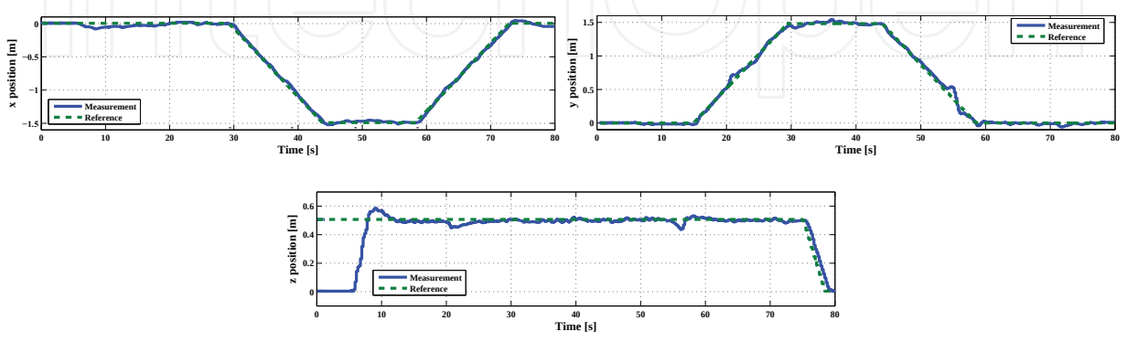
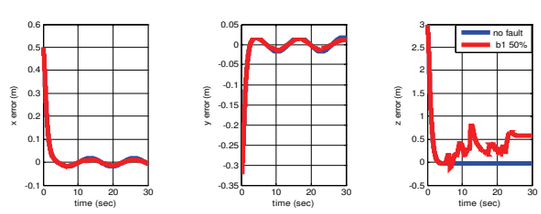


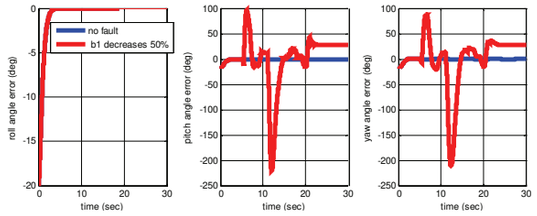
Fig. 9. Qball-X4 evolution along the x, y and z directions using SMC-based PFTC.

4.4 BSC simulation results

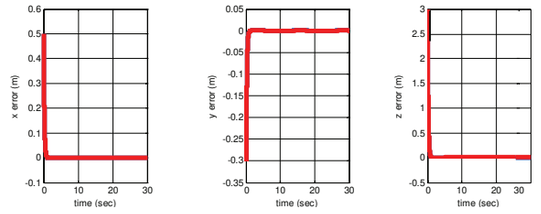
In the BSC approach, the system is required to follow a circular trajectory and the the actuator faults are injected at time $t = 5\ s$. Simulations are carried out with different control gains $k_1 = 1$, $k_2 = 3$ and $k_1 = 5$, $k_2 = 30$. Figures 10(a) and 10(b) show a comparison between the fault-free case and the fault-case of 50% loss of control effectiveness for both position and angle tracking errors. The control gains used in Figures 10(a) and 10(b) are $k_1 = 1$ and $k_2 = 3$. One can see that in the fault case, the position tracking errors in the x and y directions change slightly whereas the z tracking error is greatly affected. The roll, pitch and yaw tracking errors are also affected as can be seen in Figure 10(b). With higher control gains $k_1 = 5$ and $k_2 = 30$, it is possible to reduce fault effects on tracking errors as can be seen in Figures 10(c) and 10(d).



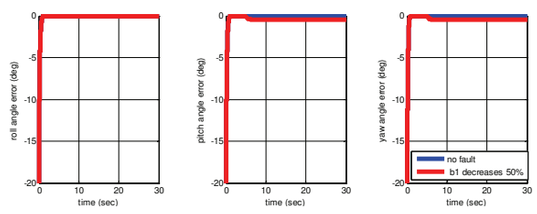
(a) Position errors with $k_1 = 1/k_2 = 3$



(b) Angle errors with $k_1 = 1/k_2 = 3$



(c) Position errors with $k_1 = 5/k_2 = 30$



(d) Angle errors with $k_1 = 5/k_2 = 30$

Fig. 10. Position and angle error comparison between fault-free and 50% control effectiveness loss in first actuator, with control gains $k_1 = 1/k_2 = 3$ and $k_1 = 5/k_2 = 30$.

4.5 MPC simulation results

To illustrate the MPC approach, the quadrotor is assumed to be on the ground initially and it is required to reach a hover height of 4 m and stay in that height while stabilizing the pitch and roll angles. The x^d, y^d and ψ^d are $x^d = 2\ m$, $y^d = 3\ m$ and $\psi^d = 0$. The upper left plot of Figure 11 shows the time history of states for fault-free condition (the desired position values are dotted and the velocities are dashed lines).

The fault is designed to happen at time $t = 5\ s$. At this time it is assumed that actuator faults occur which lead to multiple simultaneous partial loss of effectiveness of three actuators as follows: $\alpha_1 = 0.9$, $\alpha_2 = 0.7$, $\alpha_3 = 0.8$ and $\alpha_4 = 1.0$ (i.e. 10% loss of control effectiveness in the first motor, 30% in the second, 20% in the third and no fault in the fourth one). The upper right plot of Figure 11 shows the time history of the states when no fault estimation is performed. In fact in this case Algorithm 1 is used without any information about the occurred fault. One

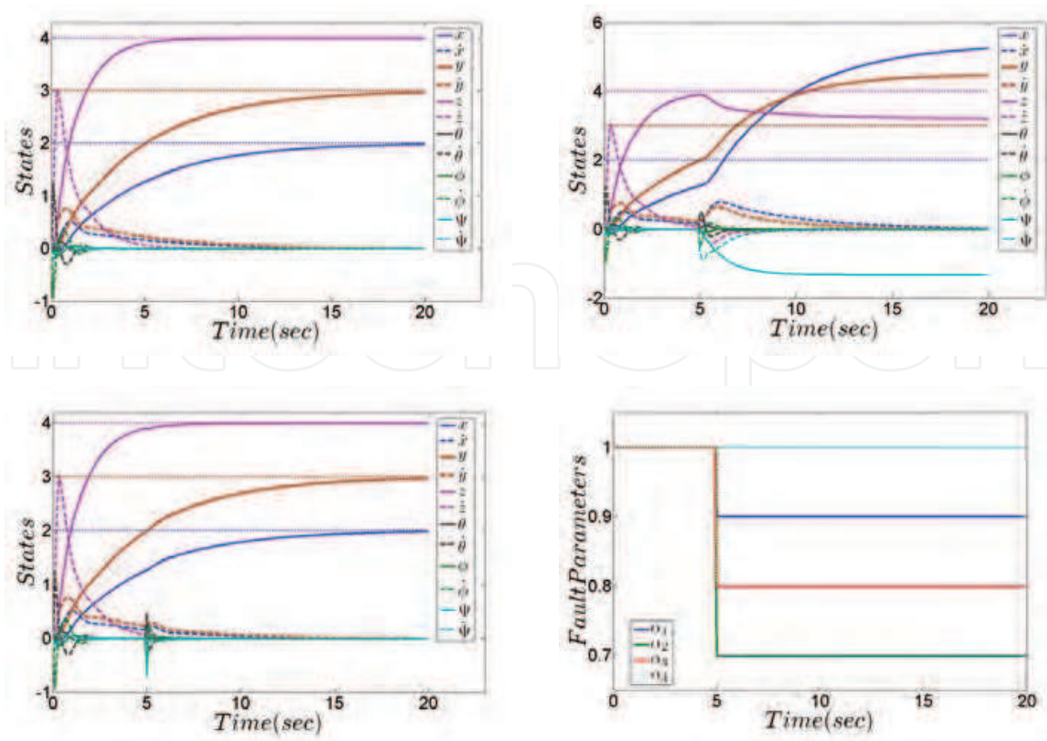


Fig. 11. MPC in fault-free and fault conditions without and with fault estimation.

can see that system states do not converge to their desired values and therefore it exhibits poor performance. The figure also shows that interestingly MPC exhibits some degree of fault tolerance inherently; all the linear and angular velocities are stabilized and there is only some error in the positions and orientations.

For the fault-tolerant MPC, Problem 2 is used where fault parameters estimation is provided by employing the Moving Horizon Estimator (MHE) Izadi et al. (2011b). The two lower plots of Figure 11 show fault parameters estimates as well as system states. It is clear that fault-tolerant MPC improves the system performance compared to the case without fault-tolerance.

4.6 FTPR simulation results

To illustrate the FTPR approach, let us assume that the quadrotor system is required to move from an initial position to a final one with $F_2(t_0) = 0$ and $F_3(t_0) = 0$, $F_2(t_f) = 20\ m$ and $F_3(t_f) = 30\ m$. It is also assumed that the maximal thrust that can be generated by each motor is $T_{max} = 8\ N$. ρ is set to 0.1 and the approach described is Section 3.6.3 gives the solution of $t_f = 6.81\ s$. The simulation results for this scenario are given in Figure 12 which shows the system trajectories along the x and y directions. The figure also shows that the four applied thrusts do not exceed ρT_{max}.

For the fault scenario, it is assumed that the Qball-X4 loses 25% of the control effectiveness of its fourth rotor at the time instant $t = 2\ s$. In this case and without trajectory re-planning, Figure 13(a) shows that the damaged system is not able anymore to reach the final desired

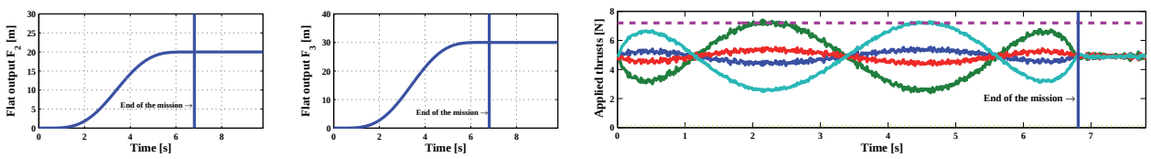
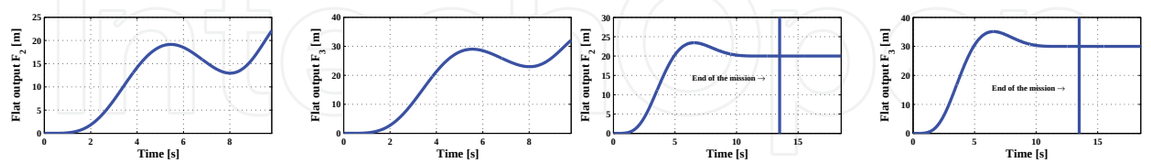
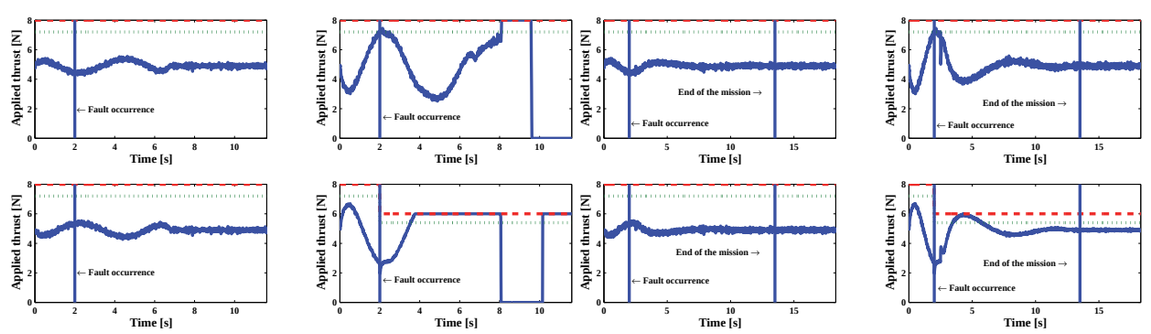


Fig. 12. System evolution in the x and y directions and the applied thrusts.



(a) Trajectories without re-planning (b) Trajectories with re-planning



(c) Control inputs without re-planning (d) Control inputs with re-planning

Fig. 13. Trajectories and control inputs without and with re-planning.

position. Figure 13(c) shows how the applied thrusts saturate when the system is forced to follow the pre-fault nominal trajectory. For the trajectory re-planning, once the fault is detected, isolated and identified, the trajectories are re-planned at $t = 2.5\ s$ where $0.5\ s$ is the time assumed to be taken by the FDD module. The solution obtained from Section 3.6.4 is $t_f = 13.3\ s$. Figure 13(b) shows that after re-planning, the UAV is able to reach the final desired position. The applied thrusts are illustrated in Figure 13(d) where it can be seen that the thrusts remain smaller than their maximal allowable limits. Thus, trajectory planning can help in keeping system stability by redefining the desired trajectories to be followed by taking into consideration the post-fault actuators limits.

5. Conclusion

This chapter presents some of the work that have been carried out at the Networked Autonomous Vehicles Laboratory of Concordia University. The main concern is to propose approaches that can be effective, easy to implement and to run on-board the UAVs. Many of the proposed methods have been implemented and tested with the Qball-X4 testbed and current work aims to propose and implement more advanced and practical techniques. As one of functional blocks in an AFTCS framework, FDD plays an important role for successful fault-tolerant control of systems (see Figure 3). Development on FDD techniques could not be included in the chapter due to space limit. Interested readers can refer to our recent work in

(Ma & Zhang, 2010a), (Ma & Zhang, 2010b), (Ma, 2011), (Gollu & Zhang, 2011), (Zhou, 2009) and (Amoozgar et al., 2011) for information on FDD techniques applied to UAV and satellite systems. Research work on fault-tolerant attitude control for spacecraft in the presence of actuator faults could not also be included but can be found in (Hu et al., 2011a), (Hu et al., 2011b), (Xiao et al., 2011a) and (Xiao et al., 2011b). Current and future work will focus more on the multi-vehicles cooperative control where preliminary works on cooperative control have been presented in (Izadi et al., 2009b), (Izadi et al., 2011a), (Sharifi et al., 2010), (Sharifi et al., 2011), (Mirzaei et al., 2011) and (Qu & Zhang, 2011).

6. Acknowledgements

Financial support of this work by the Natural Sciences and Engineering Research Council of Canada (NSERC) through a Strategic Project Grant (STPGP 350889-07) and Discovery Project Grants is highly acknowledged. Support from Quanser Inc. and colleagues from Quanser Inc. for the development of the Qball-X4 UAV testbed is also highly appreciated. Acknowledgements go also to the colleagues at the Department of Mechanical and Industrial Engineering of Concordia University, DRDC and other academic and industrial partners of the above grants, and all Postdoctoral Fellows/Research Associates, PhD, MASc/MEng students who contributed to this work.

7. References

Amoozgar, M. H., Gollu, N., Zhang, Y. M., Lee, J. & Ng, A. (2011). Fault detection and diagnosis of attitude control system for the JC2Sat-FF mission, *The 4th International Conference on Spacecraft Formation Flying Missions and Technology*, Saint Hubert, QC, Canada.

Bani Milhim, A. (2010a). *Modeling and Fault Tolerant PID Control of a Quad-rotor UAV*, Master's thesis, Concordia University, Montreal, QC, Canada.

Bani Milhim, A., Zhang, Y. M. & Rabbath, C. A. (2010b). Gain scheduling based PID controller for fault tolerant control of a quad-rotor UAV, *AIAA Infotech@Aerospace*, Atlanta, Georgia, USA.

Bresciani, T. (2008). *Modelling, Identification and Control of a Quadrotor Helicopter*, Master's thesis, Lund University, Sweden.

Chamseddine, A., Zhang, Y. M., Rabbath, C. A., Fulford, C. & Apkarian, J. (2011a). Model reference adaptive fault tolerant control of a quadrotor UAV, *AIAA Infotech@Aerospace*, St. Louis, Missouri, USA.

Chamseddine, A., Zhang, Y. M., Rabbath, C. A., Join, C. & Theilliol, D. (2011b). Flatness-based trajectory planning/re-planning for a quadrotor unmanned aerial vehicle, *IEEE Transactions on Aerospace and Electronic Systems (To appear)* .

Chen, H. & Allgower, F. (1998). A quasi infinitive horizon nonlinear model predictive control scheme with guaranteed stability, *Automatica*, 34(10): 1205–1217.

Dierks, T. & Jagannathan, S. (2008). Neural network output feedback control of a quadrotor UAV, *Proceedings of the 47th IEEE Conference on Decision and Control*, Cancun, Mexico, pp. 3633–3639.

Dierks, T. & Jagannathan, S. (2009). Neural network control of quadrotor UAV formations, *American Control Conference*, St. Louis, Missouri, USA, pp. 2990–2996.

Edwards, C., Lombaerts, T. & Smaili, H. (2010). *Fault Tolerant Flight Control: A Benchmark Challenge*, Lecture Notes in Control and Information Sciences, Springer.

Gollu, N. & Zhang, Y. M. (2011). Fault detection and diagnosis of a thruster controlled satellite for autonomous rendezvous and docking, *The 4th International Conference on Spacecraft Formation Flying Missions and Technology*, Saint Hubert, QC, Canada.

Haugen, F. (2004). *PID Control*, Tapir Academic Press.

Hu, Q.-L., Xiao, B. & Zhang, Y. M. (2011a). Fault-tolerant attitude control for spacecraft under loss of actuator effectiveness, *Journal of Guidance, Control, and Dynamics,* 3(34): 927–932.

Hu, Q.-L., Zhang, Y. M., Huo, X. & Xiao, B. (2011b). Adaptive integral-type sliding mode control for spacecraft attitude maneuvering under actuator stuck failures, *Chinese Journal of Aeronautics,* 1(24): 32–45.

Ioannou, P. A. & Sun, J. (1995). *Robust Adaptive Control*, Prentice Hall PTR.

Izadi, H. A. (2009a). *Decentralized Receding Horizon Control of Cooperative Vehicles with Communication Delays*, PhD thesis, Concordia University, Montreal, QC, Canada.

Izadi, H. A., Gordon, B. W. & Zhang, Y. M. (2009b). Decentralized receding horizon control for cooperative multiple vehicles subject to communication delay, *AIAA Journal of Guidance, Control, and Dynamics,* 6(32): 1959–1965.

Izadi, H. A., Gordon, B. W. & Zhang, Y. M. (2011a). Decentralized model predictive control for cooperative multiple vehicles subject to communication loss, *Special Issue on "Formation Flight Control" in International Journal of Aerospace Engineering,* 2011.

Izadi, H. A., Zhang, Y. M. & Gordon, B. W. (2011b). Fault tolerant model predictive control of quad-rotor helicopters with actuator fault estimation, *Proceedings of the 18th IFAC World Congress*, Milano, Italy.

Johnson, E. N., Chowdhary, G. V. & Kimbrell, M. S. (2010). Guidance and control of an airplane under severe structural damage, *AIAA Infotech@Aerospace*, Atlanta, Georgia, USA.

Krstic, M., Kanellakopoulos, I. & Kokotovic, P. V. (1995). *Nonlinear and Adaptive Control Design*, Wiley-Interscience, New York.

Li, T. (2011a). *Nonlinear and Fault-tolerant Control Techniques for a Quadrotor Unmanned Aerial Vehicle*, Master's thesis, Concordia University, Montreal, QC, Canada.

Li, T., Zhang, Y. M. & Gordon, B. (2011b). Fault tolerant control applied to a quadrotor unmanned helicopter, *Proceedings of the 7th ASME/IEEE International Conference on Mechatronics & Embedded Systems & Applications*, Washington DC, USA.

Ma, L. (2011). *Development of Fault Detection and Diagnosis Techniques with Applications to Fixed-wing and Rotary-wing UAVs*, Master's thesis, Concordia University, Montreal, QC, Canada.

Ma, L. & Zhang, Y. M. (2010a). Fault detection and diagnosis for GTM UAV with dual unscented Kalman filter, *AIAA Guidance, Navigation, and Control Conference*, Toronto, Ontario, Canada.

Ma, L. & Zhang, Y. M. (2010b). DUKF-based fault detection and diagnosis for GTM UAV using nonlinear and LPV models, *Proceedings of the 6th ASME/IEEE International Conference on Mechatronic & Embedded Systems & Applications*, Qingdao, P. R. China.

Maciejowski, J. M. & Jones, C. N. (2003). MPC fault-tolerant flight control case study: flight 1862, *IFAC Symposium on Safeprocess*, Washington D.C., USA.

MAST (2011). `https://www.grasp.upenn.edu/research/micro_autonomous_system_technologies_mast`.

Mirzaei, M., Sharifi, F., Gordon, B. W., Rabbath, C. A. & Zhang, Y. M. (2011). Cooperative multi-vehicle search and coverage problem in an uncertain environment, *Accepted*

by the 50th IEEE Conference on Decision and Control and European Control Conference, Orlando, Florida, USA.

NAV (2011). http://users.encs.concordia.ca/~ymzhang/UAVs.htm.

Nguyen, H. V., Berbra, C., Lesecq, S., Gentil, S., Barraud, A. & Godin, C. (2009). Diagnosis of an inertial measurement unit based on set membership estimation, *The 17th Mediterranean Conference on Control and Automation*, Thessaloniki, Greece, pp. 211–216.

Noura, H., Theilliol, D., Ponsart, J. C. & Chamseddine, A. (2009). *Fault-tolerant Control Systems: Design and Practical Applications*, Springer.

Qu, Y. H. & Zhang, Y. M. (2011). Cooperative localization against GPS signal loss in multiple UAVs flight, *Invited Special Issue on "Fault Detection, Diagnosis and Tolerant Control" in Journal of Systems Engineering and Electronics*, 1(22): 103–122.

Quanser (2010). Quanser Qball-X4 User Manual. Document number 829.

Rafaralahy, H., Richard, E., Boutayeb, M. & Zasadzinski, M. (2008). Simultaneous observer based sensor diagnosis and speed estimation of unmanned aerial vehicle, *Proceedings of the 47th IEEE Conference on Decision and Control*, Cancun, Mexico, pp. 2938–2943.

Sadeghzadeh, I., Mehta, A., Zhang, Y. M. & Rabbath, C. A. (2011). Fault-tolerant trajectory tracking control of a quadrotor helicopter using gain-scheduled PID and model reference adaptive control, *Annual Conference of the Prognostics and Health Management Society*, Montreal, QC, Canada.

Sharifi, F., Gordon, B. W. & Zhang, Y. M. (2010). Decentralized sliding control of cooperative multi-agent systems subject to communication delays, *AIAA Guidance, Navigation, and Control Conference*, Toronto, Ontario, Canada.

Sharifi, F., Zhang, Y. M. & Gordon, B. W. (2011). Voroni-based coverage control for multi-quadrotor UAVs, *Proceedings of the 7th ASME/IEEE International Conference on Mechatronic & Embedded Systems & Applications*, Washington, DC, USA.

Singh, G. K. & Holé, K. E. (2004). Guaranteed performance in reaching mode of sliding mode controlled systems, *Sadhana*, 29(1): 129–141.

STARMAC (2011). http://hybrid.stanford.edu/~starmac/project.htm.

Stepanyan, V. & Krinshnakumar, K. (2010a). Input and output performance of M-MRAC in the presence of bounded disturbances, *AIAA Guidance, Navigation, and Control Conference*, Toronto, Ontario, Canada.

Stepanyan, V. & Krinshnakumar, K. (2010b). MRAC revisited: guaranteed performance with reference model modification, *American Control Conference*, Baltimore, Maryland, USA, pp. 93–98.

SWARM (2011). http://vertol.mit.edu.

Xiao, B., Hu, Q.-L. & Zhang, Y. M. (2011a). Fault-tolerant attitude control for flexible spacecraft without angular velocity magnitude measurement, *Journal of Guidance, Control, and Dynamics*, 5(34): 1556–1561.

Xiao, B., Hu, Q.-L. & Zhang, Y. M. (2011b). Adaptive sliding mode fault tolerant attitude tracking control for flexible spacecraft under actuator saturation, *IEEE Transactions on Control Systems Technology (Available on-line on 25 October 2011)* .

Zhang, X. (2010a). *Lyapunov-based Fault Tolerant Control of Quadrotor Unmanned Aerial Vehicles*, Master's thesis, Concordia University, Montreal, QC, Canada.

Zhang, X., Zhang, Y. M., Su, C.-Y. & Feng, Y. (2010b). Fault tolerant control for quadrotor via backstepping approach, *The 48th AIAA Aerospace Sciences Meeting Including the New Horizons Forum and Aerospace Exposition*, Orlando, Florida, USA.

Zhang, Y. M. & Jiang, J. (2008). Bibliographical review on reconfigurable fault-tolerant control systems, *IFAC Annual Reviews in Control*, 32(2): 229–252.

Zhou, Q.-L. (2009). *Reconfigurable Control Allocation Design with Applications to Unmanned Aerial Vehicle and Aircraft*, Master's thesis, Concordia University, Montreal, QC, Canada.

IntechOpen

IntechOpen



Automatic Flight Control Systems - Latest Developments
Edited by Dr. Thomas Lombaerts

ISBN 978-953-307-816-8
Hard cover, 204 pages
Publisher InTech
Published online 18, January, 2012
Published in print edition January, 2012

The history of flight control is inseparably linked to the history of aviation itself. Since the early days, the concept of automatic flight control systems has evolved from mechanical control systems to highly advanced automatic fly-by-wire flight control systems which can be found nowadays in military jets and civil airliners. Even today, many research efforts are made for the further development of these flight control systems in various aspects. Recent new developments in this field focus on a wealth of different aspects. This book focuses on a selection of key research areas, such as inertial navigation, control of unmanned aircraft and helicopters, trajectory control of an unmanned space re-entry vehicle, aeroservoelastic control, adaptive flight control, and fault tolerant flight control. This book consists of two major sections. The first section focuses on a literature review and some recent theoretical developments in flight control systems. The second section discusses some concepts of adaptive and fault-tolerant flight control systems. Each technique discussed in this book is illustrated by a relevant example.

How to reference
In order to correctly reference this scholarly work, feel free to copy and paste the following:

Youmin Zhang and Abbas Chamseddine (2012). Fault Tolerant Flight Control Techniques with Application to a Quadrotor UAV Testbed, Automatic Flight Control Systems - Latest Developments, Dr. Thomas Lombaerts (Ed.), ISBN: 978-953-307-816-8, InTech, Available from: http://www.intechopen.com/books/automatic-flight-control-systems-latest-developments/fault-tolerant-flight-control-techniques-with-application-to-a-quadrotor-uav-testbed

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821

© 2012 The Author(s). Licensee IntechOpen. This is an open access article distributed under the terms of the Creative Commons Attribution 3.0 License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

IntechOpen

IntechOpen