A Survey of Quadrotor Design and Control

Rainer Trummer
rtrummer@cs.uni-salzburg.at

Department of Computer Sciences
University of Salzburg, Austria

June 26, 2007

Abstract

Flying a quadrotor helicopter requires at least a means for stabilization control. Although efficiently stabilizing a quadrotor is itself an extensive research field, adding autonomous control poses even more challenging problems: the underlying quadrotor platform must provide sufficiently precise sensors with high update rates, the mechanical parts should not incur large update delays, and the controller should operate properly even under the influence of external perturbations. The aim of this survey is to shade some light on existing quadrotor projects and provide a brief overview about the current status.

1 Introduction

Quadrotor helicopters can be considered not belonging to state-of-the-art helicopters. This is primarily due to the fact that, except for some historical models that can be found in designated museums, they are not being built in full scale anymore. Despite the reasons responsible for this specific evolution in flight history, quadrotors are on their way back into publicity again, at least in the sense of miniature models.

Compared to the most commercially-used helicopter design that suggests a single main rotor combined with a small rear rotor for compensating a spin around the main-rotor axis, quadrotor helicopters are different in many aspects. First, the term "quadrotor" implies that there are four rotors involved. They are arranged in a cross-like shape, where one pair of opposite rotors spins clockwise and the other one counter-clockwise. Second, especially regarding miniature models, there does not exist the need for pitching the rotor blades. More precisely, the blades are mounted in some fixed position that does not allow any change of the actual pitch angle. Consequently, differences in the amount of thrust generated by such a fixed-pitch rotor is achieved by merely regulating its rotational speed. In fact, any physically possible movement of a fixed-pitch quadrotor can be accomplished by regulating the rotational speed of its four rotors. Although this sounds fairly simple, in reality it is quite difficult to successfully control a quadrotor, since they are inherently instable. This, of course, poses very challenging problems in the field of control engineering.

Many different control concepts for quadrotors have been introduced in the past decades, however, there does not seem to exist an optimal solution so far.

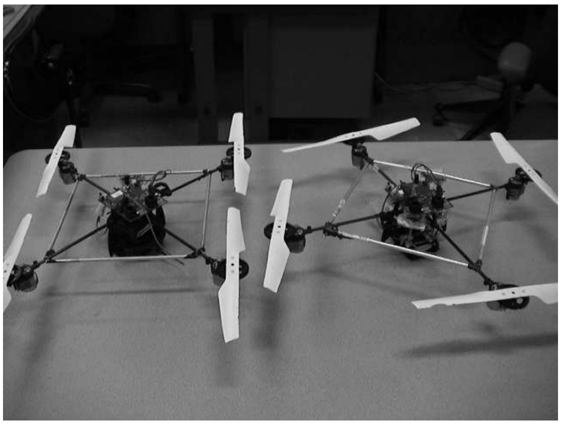
2 Selected Quadrotor Projects

2.1 Toward Obstacle Avoidance on Quadrotors [1].

The aim of this project is to develop an active control system for quadrotors that enables fully autonomous flight facilitated by a means for obstacle recognition. The quadrotor used is a self-made light-weight construction with a total diameter of 800mm and total mass of 520g. It is equipped with an IMU (Inertial Measurement Unit), a differential GPS reciever, a mini-computer carrying a 266MHz Geode 1200 processor, and five ultrasonic sensors, one pointing down for measuring the altitude and four in the plane for obstacle recognition. The onboard system is running a Debian-based minimal Linux distribution and connected to the ground station using wireless LAN. Placing the focus on autonomous flight with obstacle avoidance, five distinct algorithms were implemented and tested by simulation before conducting experiments with the real helicopter. It turned out that obstacle avoidance on a quadrotor is possible but also restricted in many ways due to the inherent instability of such platforms. More precisely, aerodynamical constraints demand a complex combination of stabilization, trajectory, and avoidance-maneuver control.

2.2 The Stanford Testbed of Autonomous Rotorcraft for Multi-Agent Control (STARMAC) [2].

The STARMAC project is based on a commercially available quadrotor, called the Draganflyer III, and mainly concerned with control engineering and multi-agent control. They started by modifying the base model, which is capable of carrying an additional payload of approximately 300g, by augmenting it with an IMU, differential GPS, and one ultrasonic sensor required for the altitude. The original onboard electronics was replaced by a custom-built printed-circuit board that contains two PIC18F6520 microchips responsible for all flight activities. The communication link to the ground station is implemented by a Bluetooth Class II device, which offers a larger range at an operating frequency of 2.4GHz. The controller is designed as a single loop that polls all sensors for data, as well as the communication channel, and performs all necessary computations with each iteration using floating-point arithmetic. In order to regulate the contribution of the roll, pitch, yaw, and altitude portion, a penalty strategy was implemented. Due to this, it is possible to adjust the contribution of each individual controller in dependence of the others by penalizing the occurring deviations.

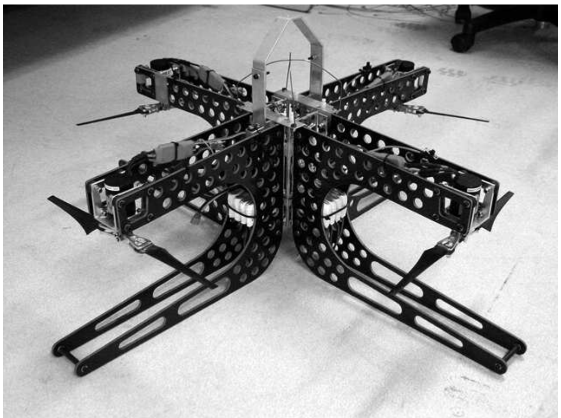


2.3 Multi-Agent X4-Flyer Testbed Control Design: Integral Sliding Mode vs. Reinforcement Learning [3].

As an extension to the previously introduced STARMAC project, two different approaches of altitude control were implemented in order to improve the testbed stability regarding altitude deviations. Because classical linear techniques failed to provide sufficient stability performance, mainly due to the complex airflow induced by the four interacting rotors, ISM (Integral Sliding Mode) and RL (Reinforcement Learning) were applied to accommodate non-linear disturbances. Compared to standard linear-control design techniques, ISM causes a significant improvement, especially when incorporating an integral error term to the control policy. RL on the other hand stands out with its ease of implementation, but requires several hours of training with a simulator before it can be applied to the real helicopter. One drawback of RL is its sensitivity to system disturbances it was not trained for. For instance, the ground effect that demands much higher thrust levels while taking off or landing, varying battery levels, as well as rotor blade degradation, can easily lead to instable flight behavior. Regarding step response, both ISM and RL show similar response time and stable performance.

2.4 Towards Dynamically-Favorable Quadrotor Aerial Robots [4].

In this project, a custom-built quadrotor is introduced that stands out with an aero-elastic rotor blade design. The rotors are inverted and placed downwards with the aim to obtain favorable stability properties. The blades itself are mounted with a sprung teetering rotor hub, first, to compensate for natural flapping that occurs with thin blades, and second, to allow an adjustment of the blade flapping characteristics. They have been modelled in accordance to blade element and momentum theory in order to achieve maximal thrust performance. The chassis is constructed from carbon fiber with a central body and four rotor arms, where each arm consists of two similar plates with some distance between. Compared to most other quadrotor model helicopters, the present one is much heavier with a total weight of 4kg and is designed to carry an additional payload of 1kg. The body holds an IMU located exactly in the center of gravity and a mini-computer equipped with an HC-12 microprocessor. Nothing is said about senors used for measuring the distance to ground. Placing the focus on thrust generation and dynamic stability, measurements that proof the efficiency of the individual rotor design are presented, but no flight tests have been conducted at this point.



2.5 The JAviator Project: A High-Precision Quadrotor [5].

The JAviator (Java Aviator) is a custom-built high-precision quadrotor aimed to serve as flying software laboratory. Its frame design is based on a bicycle wheel which, first, offers more stability compared to designs based on one-point rotor axis mountings, and second, allows the usage of very thin and light-weight materials. The main portion of the frame is built from carbon fiber, whereas aircraft aluminium and medical titanium was chosen for the connecting parts and propulsion groups. The JAviator has a total diameter of 1.1m and total mass of 1.6kg. It can carry an additional payload of 1kg due to custom-built brushless motors, which are much stronger than state-of-the-art brushless motors that come at a weight of 35g. The sensor equipment consists of an IMU and one ultrasonic sensor for measuring the distance to ground. The computational devices comprise a robostix board carrying an Atmel Atmega 128 processor that serves for data acquisition and generating the PWM motor signals, a gumstix board carrying an Intel XScale 400 processor that runs the controller algorithm, and a WIFI daughter card for providing WLAN connectivity to the ground station. Except for the low-level sensing and actuating software, running on the Atmega 128 and written in C, all high-level software running on the XScale 400 is written in Java. Referring to the successful first all-Java flight conducted in October 2006, this was the first time that the programming language Java was used for controlling a quadrotor.

References

[1] Samir Bouabdallah, Marcelo Becker, Vincent de Perrot, and Roland Siegwart, Computer Obstacle Avoidance on Quadrotors, *Proceedings of the XII International Symposium on Dynamic Problems of Mechanics*, 2007.

[2] Gabe Hoffmann, Dev Gorur Rajnarayan, Steven L. Waslander, David Dostal, Jung Soon Jang, and Claire J. Tomlin, The Stanford Testbed of Autonomous Rotorcraft for Multi-Agent Control (STARMAC), *Proceedings of the 23rd Digital Avionics Systems Conference*, 2004.

[3] Steven L. Waslander, Gabriel Hoffmann, Jung Soon Jang, and Claire J. Tomlin, Multi-Agent X4-Flyer Testbed Control Design: Integral Sliding Mode vs. Reinforcement Learning, *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2005.

[4] Paul Ponds, Robert Mahony, Joel Gresham, Peter Corke, and Jonathan Roberts, Towards Dynamically-Favourable Quad-Rotor Aerial Robots, *Proceedings of the Australasian Conference on Robotics and Automation*, 2004.

[5] The JAviator Project: `http://javiator.cs.uni-salzburg.at/`.