Available online at www.sciencedirect.com

ScienceDirect

IFAC-PapersOnLine 48-3 (2015) 2435–2441

Experimental Fuzzy Logic Controller Type 2 for a Quadrotor Optimized by ANFIS

Pedro Ponce, Arturo Molina, Israel Cayetano, Jose Gallardo, Hugo Salcedo, Jose Rodriguez

pedro.ponce@itesm.mx, armolina@itesm.mx.

Tecnologico de Monterrey campus Ciudad de México –México City

Abstract: Quadrotors are nonlinear systems that can be controlled by human experts. Since the mathematical models of Quadrotors are quite complex, the expert knowledge may be one way to come to a solution for controlling Quadrotors. However, human experts make occasionally mistakes, and thus some linguistic rules used in the controller may be false or redundant. Hence, fuzzy logic type 2 optimized by ANFIS (Adaptive Neuro-Fuzzy Inference Systems), which can deal with uncertainties, could be applied to control Quadrotors. ANFIS can optimize the number of linguistic rules and the domain of membership functions could be adjusted automatically. In addition, ANFIS should also be capable of identifying bad rules so the digital system (micro-controller) can decrease the computational resources required for implementing the fuzzy logic controller type 2. The controller designed can accomplish excellent experimental results when it is reduced by an ANFIS system. To confirm robustness in the fuzzy logic controller a noise signal was added in the position control loop for the Quadrotor. Besides, a comparison between fuzzy logic controller type 2 tuned by an expert and Fuzzy Logic type 2 optimized by an ANFIS is illustrated. Experimental results confirmed the good response reached when fuzzy logic type 2 optimized by ANFIS is deployed.

Keywords: Quadrotor, Fuzzy Logic type 2, ANFIS, Human expert.

1. INTRODUCTION

The study of Fuzzy Logic controllers for Quadrotors have been developed highly in the last decade. Coza and Macnab presented in 2006 a new Robust Adaptive-Fuzzy control method for Quadrotor stabilization. They propose a method using control and center updates for each axis rotation to approximate the same nonlinear function as the e-modification method. Results on simulink showed that this new method offered a better center tuning than the e-method, obtaining less error in the steady state although oscillations were present in a range of ±0.2 radians (Coza C. M., 2006).

In 2010 Santos, Morata and López developed a PID-like type-1 fuzzy controller for the three axes of a Quadrotor and the altitude using trapezoid membership functions. The controller was tested in simulink showing a smooth, fast, stable response (Santos, 2010). The same year, Kirly et. al. presented their work of the design of a Fuzzy controller embedded into a TMS320F28335 micro processing unit, testing the axes separately. They obtained that starting from 3 and 12 degrees, the controller was able to reach its steady state of ±2 degrees near-horizontal and good response to perturbations. They note also that given the sensitiveness of their inclination sensor, vibrations given by environmental noises are to be studied extensively (Kirli, 2010).

In 2013 Sheikpour and Shouraki published their results of the design of a Fuzzy controller using the Parallel Distributed Compensation method obtained from a Takagi-Sugeno fuzzy model of a Quadrotor. They show the viability of this method obtaining a ~1 second response with a ~0.1 radians of overshoot. (Sheikhpour, 2013). The same year, Ilhan and Karakose presented their work in Type-2 Fuzzy Logic controller for a Quadrotor for position and altitude using triangular membership functions. It showed a very slightly better response than type-1 Fuzzy approach and significantly better than PID. However, it was not delved in the tuning and selection of different Footprints of Uncertainty (Ilhan, 2013).

In 2014, it was presented a Real Time Fuzzy controller embedded in a GUMSTIX Overo FIRESTORM COM microcontroller board. The controller was obtained using the ANFIS system from test data obtained from a first experiment. The results showed that fuzzy controller is easily capable of controlling the Quadrotor, with the advantage that it was self-tuned as opposed to the PD controller. Besides, Fuzzy outperformed PD in certain conditions (Bhatkhande, 2014). Finally, in 2014, a Hybrid method of backstepping and fuzzy adaptive PID is proposed by Qingji, Fengfa and Dandan. In it, a Fuzzy Inference System is used to tune the parameters K_p, K_i and K_d of a PID controller. Simulation and practical results showed that this hybrid controller performed achieved the stabilization, effectiveness and robustness desired, with variations of ±1 degree in the steady state and rejection of disturbances of ~5 degrees (Qingji, 2014).

Peer review under responsibility of International Federation of Automatic Control.
10.1016/j.ifacol.2015.06.453

1.1 Quadrotor Basic Principles

It is important to describe how the Quadrotor works in order to design its controller. The arrangement of motors in the Quadrotor is shown in Figure 1. Each rotor has a thrust and an angular momentum about its center of rotation, as well as a drag force opposite to the rotorcraft's direction of flight (Brito, 2009). To produce lift, the rotors have to spin at a certain speed to produce enough thrust. The quantity of thrust will determine the altitude and speed at which the Quadrotor rises. (Hoffmann, 2007). On the other hand, the spinning of each motor generates an angular momentum that will try to rotate the Quadrotor in its yaw angle such as it would happen in a helicopter without tail propeller. To avoid this effect, two intercalated rotors will spin clockwise and the other two counterclockwise so that the individual angular momentum cancel each other. Therefore, the propellers attached to each motor are different: two of them are pusher and two are puller, working in contra-rotation.

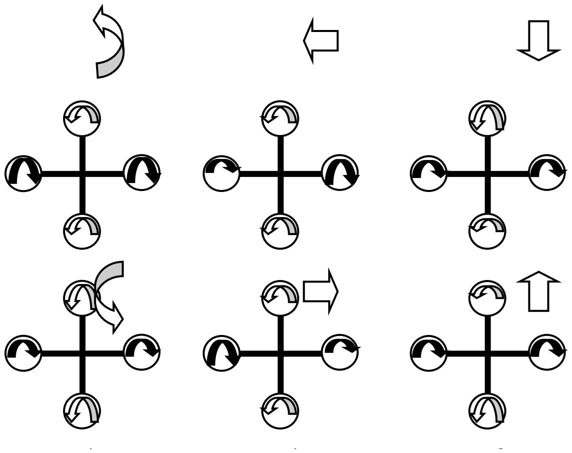

Figure 1. Quadrotor frame and functioning. (A: Clockwise Yaw, B: Anticlockwise Yaw, C: Anticlockwise Roll, D: Clockwise Roll, E: Anticlockwise Pitch, F: Clockwise Pitch)

In order to change the angle along pitch and roll axes, and therefore move it in a certain direction, depending on the desired speed of the displacement, the rotors orientated toward that direction must change their thrust as seen in the figure 1, but always taking care that the sum of angular momentum remains zero so that the Quadrotor remains in the same altitude and yaw angle.

1.2 Basic Diagram for Fuzzy Logic Type 2

Figure 2 shows the main block diagram for a Fuzzy Logic Controller Type- 2 (T2FS) which is similar to a traditional Fuzzy Logic Type 1.

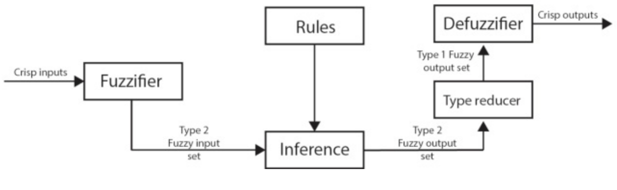

Figure 2. Block diagram of a type 2 fuzzy logic system

Fuzzifier: The fuzzifier maps the read value into the T2DS which activates the inference system.

Rule base: The antecedents and consequents are related each other by IF-ELSE language functions.

Inference: This block assign to the fuzzy input a fuzzy output according to the rules established and operators such as union (⊔) and intersection operators (⊓); those operators are equivalent to the union and intersection operations but are used in the secondary membership functions. They definition and explanation can be found in the paper presented by Mendel, 2006.

Type reduction: For some systems it is required to transform the type 2 fuzzy outputs from the inference engine into T1FS and the result is called a type reduced set. There most common method for doing this are the Mendel 2007 iteration algorithm and the (Mendel 2006) uncertainty bounds method. Both are based on the calculation of centroid of mass.

Defuzzification: Once the outputs have been reduced the defuzzification block determines the crisp value that will be send to the actuator (Mendel, 2007).

1.3 ANFIS

Fuzzy logic and neural networks are complementary tools for building intelligent systems. While neural networks are low-level computational structures that perform well when dealing with raw data, fuzzy logic deals with reasoning on a higher level, using linguistic information acquired from domain experts. However, fuzzy systems lack the ability to learn and cannot adjust themselves to a new environment. On the other hand, although neural networks can learn, they are black boxes to the user. When a training input-output example is presented to the system, the back-propagation algorithm can compute the system output and compares it with the desired output of the training example. The error is propagated backwards through the network from the output layer to the input layer. The neuron activation functions are modified as the error is propagated. To decide the necessary modifications, the back-propagation algorithm differentiates the activation functions of the neurons. Figure 3 depicts an ANFIS topology (J.-S. R. Jang 1993).

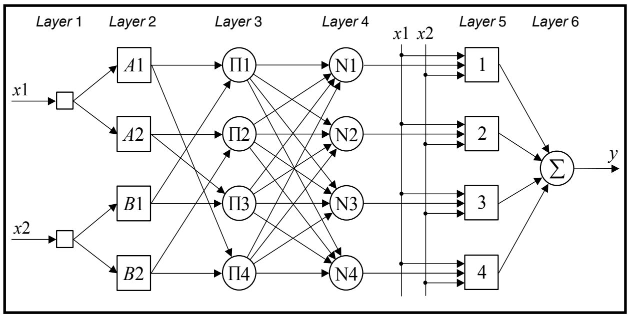

Figure 3. Depicts an ANFIS topology.