FUZZY CONTROLLER FOR A SEMI-AUTONOMOUS QUAD-ROTOR HELICOPTER

Fernando Rios-Gutierrez
Mechanical & Electrical Engineering Technology Dept.
Georgia Southern University

Dinesh Baniya Kshatri
Electrical & Computer Engineering Dept.
University of Minnesota Duluth

Abstract

This paper describes the hardware and software design and implementation of a Fuzzy Logic Based Controller that can be used to control a quad-rotor flying device using a remote computer. The system allows the human operator to take manual control of the flying device or allows the Fuzzy Logic controller to takeover and fly the device autonomously. The flying device used in this project is a miniature four propeller helicopter whose remote control transmitter was modified to interface with the controller. An electronic interface was designed to serve as the platform for the hardware of the controller, and a MATLAB program that implements a Fuzzy logic control algorithm serves as the software of the controller. The hardware interface is connected to the computer using the printer's parallel port and a 24-pin connector cable.

The software provides a graphical user interface (GUI) that allows two modes to control the quad-rotor flying device: a Manual mode and an Autopilot mode. In manual mode, the controller is a closed-loop control system as the human operator can visualize the quad-rotor's position and the surrounding environment and steer the craft accordingly. In autopilot mode the Fuzzy Logic controller is used as an open-loop control system to operate the flying device.

Introduction

Flight control of autonomous flying vehicles has been a goal in the area of fuzzy controller design for some time. More recently, a growing interest for unmanned aerial vehicles has been shown among the research community due to security concerns and protection of national borders [4-5]. Fuzzy Logic has been successfully used in control applications that are difficult to implement using traditional control methods. With its alternative design methodology that includes qualitative as well as quantitative information that can be used to control the desired system. Fuzzy logic can be applied to develop both linear and non-linear systems for intelligent control. Some of the main advantages that Fuzzy Logic offers are: it reduces design development life cycle, simplifies design complexity, improves control performance, reduces hardware costs, and most importantly it is simpler and faster to implement than conventional techniques used in control systems. These features of Fuzzy Logic make it a good option to design complex automatic control systems.

In this paper we present the implementation of a Fuzzy Logic based controller that provides control mechanism to a quad-rotor vertical takeoff and landing (VTOL) flying device. Quadrotor VTOL devices are intrinsically unstable, complex, non-linear and time-varying systems.

The design and implementation of the controller is unique in that the fuzzy control algorithm is implemented in a remote computer therefore the need of local complex control in the flying device is not required. This approach has the main advantage of not adding extra weight to the flying device since most controllers implemented for this kind of application are always limited by the mechanical characteristics (weight and size) of the flying device or the power needs and computational capabilities of a local embedded controller.

System Description.

The vehicle used in this project has four independently controlled rotors (shown in Figure 1) that are located at the front, rear, left, and right ends of a cross frame that requires no cyclic or collective pitch. The quad-rotor vehicle can be highly maneuverable and has the potential to hover and to take off, fly, and land in small areas, and can be used to perform surveying and monitoring tasks in dangerous and/or inaccessible environments.

The flight of the quad-rotor is controlled by changing the speed of rotation of each rotor. The front and rear rotors rotate in a counter-clockwise direction while the left and right rotors rotate in a clockwise direction to balance the torque created by the spinning rotors. The relative speed of the left and right rotors is varied to control the roll rate of the vehicle. Increasing the speed of the left motor by the same amount that the speed of the right motor is decreased will keep the total thrust provided by the four rotors approximately the same. In addition, the total torque created by these two rotors will remain constant. Similarly, the pitch rate is controlled by varying the relative speed of the front and rear rotors. The yaw rate is controlled by varying the relative speed of the clockwise (right and left) and counter-clockwise (front and rear) rotors. The collective thrust is controlled by varying the speed of all the rotors simultaneously.

A quad-rotor has some advantages over other rotary wing vehicles (like standard helicopters) because it is mechanically simpler and its navigation is controlled by only changing the speed of rotation for the four rotors. Since the yaw rate is controlled by changing motor speed, a tail rotor is not required to control yaw rate and all thrust can be used to provide lift. A quad-rotor may also be able to fly closer to an obstacle than a conventional helicopter that has a larger single rotor, without fear of a rotor strike. The vehicle's dynamics are good for agility and its four rotors can allow increased payload. However, the dynamics of the quad-rotor can make the vehicle difficult to control. The challenge of controlling the vehicle can be even more difficult for a small, low cost quad-rotor because it is more sensitive to climatic changes.

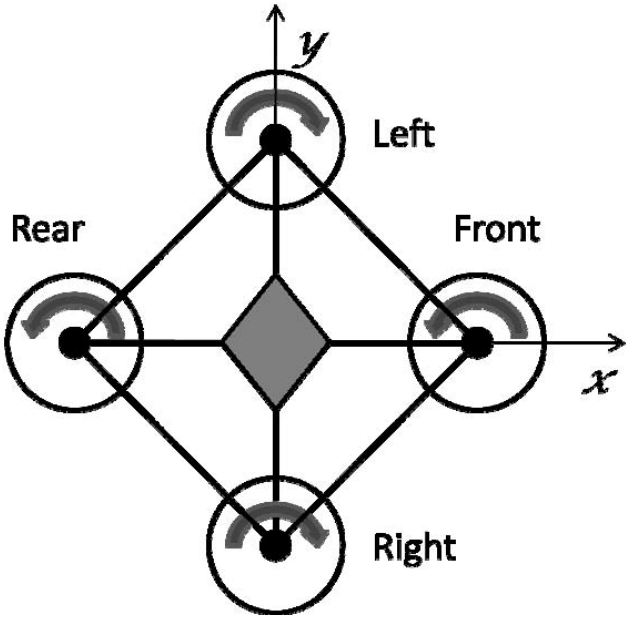

Figure1. Quad-rotor Helicopter

System Implementation

To implement our system we used a small remote controlled quad-rotor named the "DRAGANFLYER V Ti ", available from RC Toys. It has onboard electronics that include a receiver for pilot input, three piezo gyroscopes, a small microcontroller to control the quadrotor's motors and a rechargeable battery pack to power the Draganflyer's electronics and the four motors. The newest version of this product has four infrared heat sensors to allow the Draganflyer to level itself while it is flying outdoors. The Draganflyer is radio controlled operating at frequency of 72 MHz.

There are previous published works [1,2,3] that describe the implementation of controllers for quadrotor devices similar to the one we used. However, most of these control systems are implemented using a local controller (microcontroller) embedded in the flying device

itself. This approach has the disadvantage that the quad-rotor has to have enough lift to carry the embedded controller, its required power source and any other interfaces needed for the particular application. Also, because of the size and weight restriction imposed by the flying device, the type of controller and its computational power is limited, making the design of the controller more difficult and its operation less reliable.

Due to the above, for the realization of our system we decided to follow a different approach in the implementation of the controlling mechanism. We elected to use a remote larger computing system rather than using a local small processor. This approach allowed us to have access to a faster computer running more powerful software. In our case, we used a desktop PC computer running MATLAB to implement the controlling algorithm. This approach has the advantage to leave the original Draganflyer unmodified, and to use the already available radio transmitter of the quad-rotor to transmit the control signals from the controller to the flying device.

The only alteration to the original system was done to the radio transmitter, where modifications were performed to the connections of the potentiometers that mechanically control the aileron (roll), elevator (pitch), rudder (yaw), and throttle levels of the quad-rotor. These modifications were made such that they still allow the manual control of the flying device if it is so required by the human operator.

From the software point of view, using MATLAB allows a faster and more efficient implementation of the controlling algorithm, since we can use the toolboxes available in the MATLAB software, which reduces the developing time and makes the resulting program very efficient. Figure 2 shows a photo of the Draganflyer and its transmitter.

Implementation and Interfaces

Our main objective was to develop a control system for the quad-rotor through Fuzzy Logic. In order to do that we needed the following:

1) A hardware interface that would supply the appropriate communication mechanism between the computer and the hardware interface and is used to deliver the control signals to the flying device.
2) A software program where we could apply Fuzzy Logic techniques.

The implementation detail of each of these components is presented in the next sections.

Figure 2. The Draganflyer quad-rotor and transmitter.

Hardware Interface

The interface used to communicate the control commands generated by the computer to the radio transmitter is shown in Figure 3. We took advantage of the availability of a printer port in our computer, and used this port to output the control signals generated by the controller program. The connection to the PC was made through a standard DB-25 connector [6], while on the interface side we used a DIP 24-pin connector shown in Figure 3. The wiring of the circuit was color coded for ease of designing and debugging. The function of each colored wire is given next:

a) Red wires – They symbolize power lines, positive and negative voltages are needed to power the electronic devices present in the circuit. Three different voltages are needed to operate the circuit:

   i. +5 volts: Used to power up the digital circuitry. It also serves as a reference voltage for the digital to analog converter chips.

   ii. +15 and -15 volts: These voltages serve as the positive and negative power supplies for the digital to analog chips and the rail voltages for the operational amplifiers.

b) Green wires – They represent the ground connection for all the devices.

c) Orange wires – They represent the data, control and status signals received from the 24- pin solderless connector to the decoder, inverter, and octal latches.

d) White wires – They represent the wires used to connect the octal latches with the digital to analog chips and the operational amplifiers.

Figure 4 shows the complete circuit diagram of the interface. The circuit diagram begins at the DB-25 connector that receives the control signals from the computer to the hardware interface and ends at the four analog control signals - aileron, elevator, rudder, and throttle.

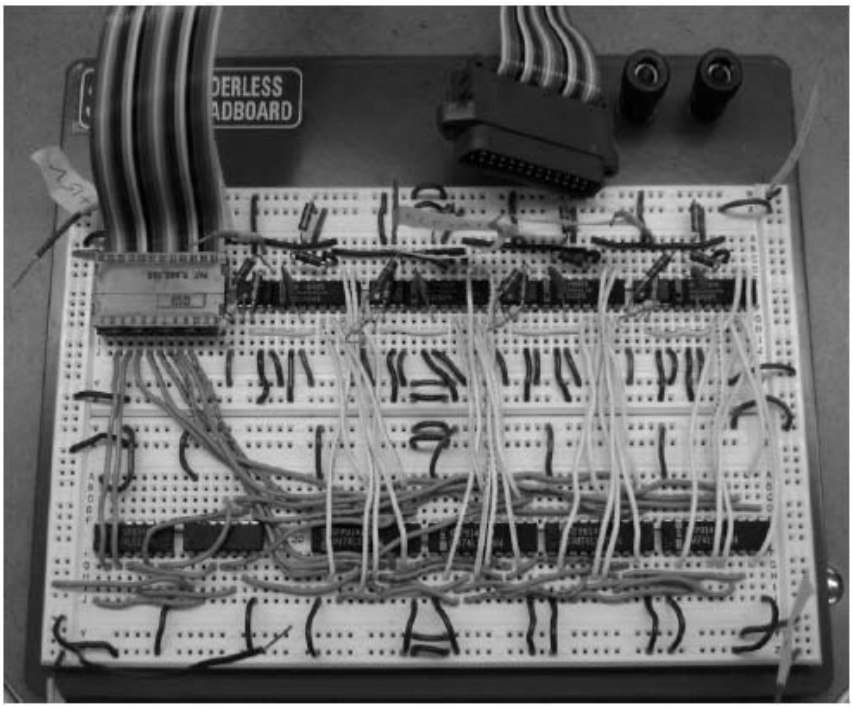

Figure 3. Hardware Interface Implemented on a Solderless Breadboard.

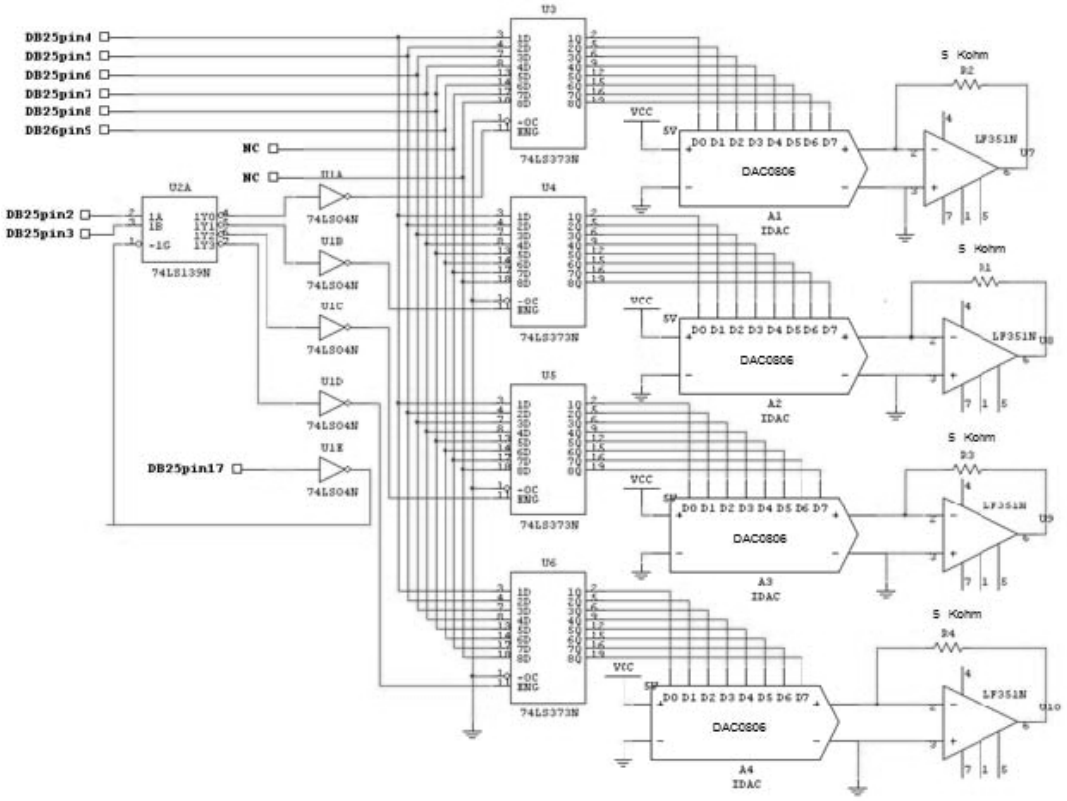

Figure 4. Controller Interface.

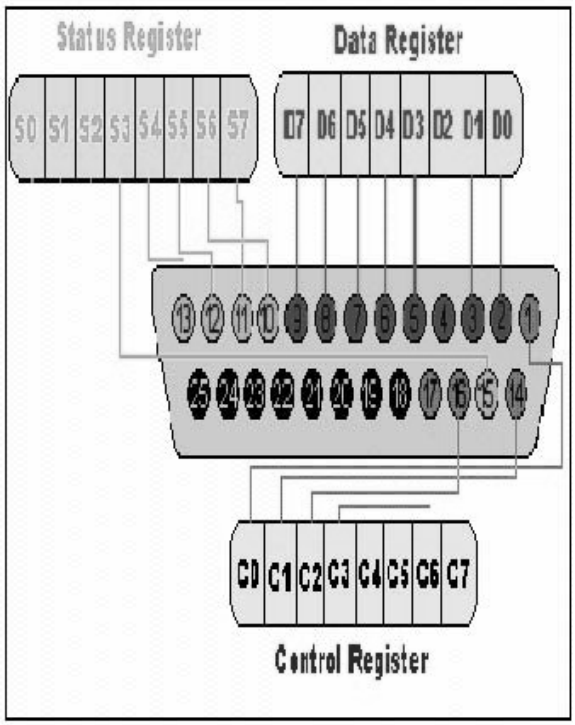

Figure 5. DB-25 Connector Groups.

Pin No DB 25	Signal Name	Direction	Register Bit	Inverted
1	Strobe	Out	Control	Yes
2	Data0	In/Out	Data 0	No
3	Data1	In/Out	Data 1	No
4	Data2	In/Out	Data 2	No
5	Data3	In/Out	Data 3	No
6	Data4	In/Out	Data 4	No
7	Data5	In/Out	Data 5	No
8	Data6	In/Out	Data 6	No
9	Data7	In/Out	Data 7	No
10	Ack	In	Status	No
11	Busy	In	Status	Yes
12	Paper Out	In	Status	No
13	Select	In	Status	No
14	Line Feed	Out	Control	Yes
15	Error	In	Status	No
16	Init	Out	Control	No
17	Select	Out	Control	Yes
18-25	Ground	-	-	-

Figure 6. DB-25 Pin Description.

The lines in DB-25 connector are divided into three groups (refer to Figures 5 and 6):

a) Data lines (data bus)
b) Control lines, and
c) Status lines.

The controller program generates digital control words that represent the commands that control the navigation of the quad-rotor. These digital encoded commands need to be translated into analog commands that are transmitted via the radio remote controller to the quad-rotor. The main purpose of the hardware interface is to perform the translation of the commands from digital to analog signals and to interconnect the computer and the radio controller. In order to perform these tasks three type of signals (data, control and status) are required to be managed by the interface. As the name refers, data is transferred over the Data lines, Control lines are used to control the peripherals operation and the status lines are used to the return status signals back to the computer from the peripherals. The Data, Control and Status lines values are controlled by variables in the MATLAB program. So, to manipulate these registers in the program, one can easily read or write to parallel port input/output instructions. The details of the signals connected through the parallel port lines are given in Figures 5 and 6.

Interface Signal Description

Pins 1-17 and 20 of the printer port were used for this project. Pins 2 through 9 are bidirectional data pins that were used only in the output direction. Pin 2 holds the least significant bit (LSB) and pin 9 holds the most significant bit (MSB). Pin 2 & 3 were used to select, using a decoder chip, the corresponding register to store the contents of the data lines. Pins 4 to 9 were used for data transmission, so only 6 significant bits were used for data. Pin 20 was used as the interface ground. Pin 17 which is an inverted output line was used to enable or disable the decoder chip. Pins 10-13 &15 were used to receive the status signals. Pins 1, 14, 16 & 17 were used to output control signals to the interface.

The purpose of using the 74LS139 decoder is to be able to select the desired flying device rotor whose speed has to be changed. This is important because it allows changing the intended rotor's speed, whereas the others remain unchanged.

To store the data for each control signals, four SN74LS373 tri-state octal latches were used. Output pins 4, 5, 6, and 7 of the 74LS139 were connected to the respective enable pins of the octal latches that hold the aileron speed, elevator speed, rudder speed, and the throttle speed respectively. The SN74LS373 acts as a gate that allows or disallows signals to be sent from its input pins to its output pins. Each of these chips controls one helicopter rotor. By disabling the latch of the chip, the previous value could be retained, whereas by enabling the latch of the chip new data could be passed on to the output. The data pins 4 (LSB), 5, 6, 7, 8, and 9 (MSB) of DB- 25 were connected to the input pins 3, 4, 7, 8, 13, and 14 of the octal latches. Since only 6 bits were used as data, the remaining two input pins 17 and 18 of the octal latches were not used. The output pins 2, 5, 6, 9, 12, and 15 of each octal latch were connected to the input pins of a separate digital to analog (DAC0806) chip to convert the digital control signals generated by the controller into their corresponding analog representation that will be transmitted to the quad-rotor, as shown in Figure 7.

The DAC design that is presented in Figure 7 was adapted and modified from the datasheet of the DAC0806 8-bit D/A converter. This design includes two chips: the DAC0806 and an operational amplifier (LF-351). The DAC0806 chip is able to intake an 8-bit digital input, however, in this project only six bits were used. Input pins 10 (LSB), 9 8, 7, 6, and 5 (MSB) of the DAC0806 were used to accept the signals being held by the octal latches. Input pins 11 and 12 of the DAC0806 were connected to ground. The DAC0806 converts the six bits into

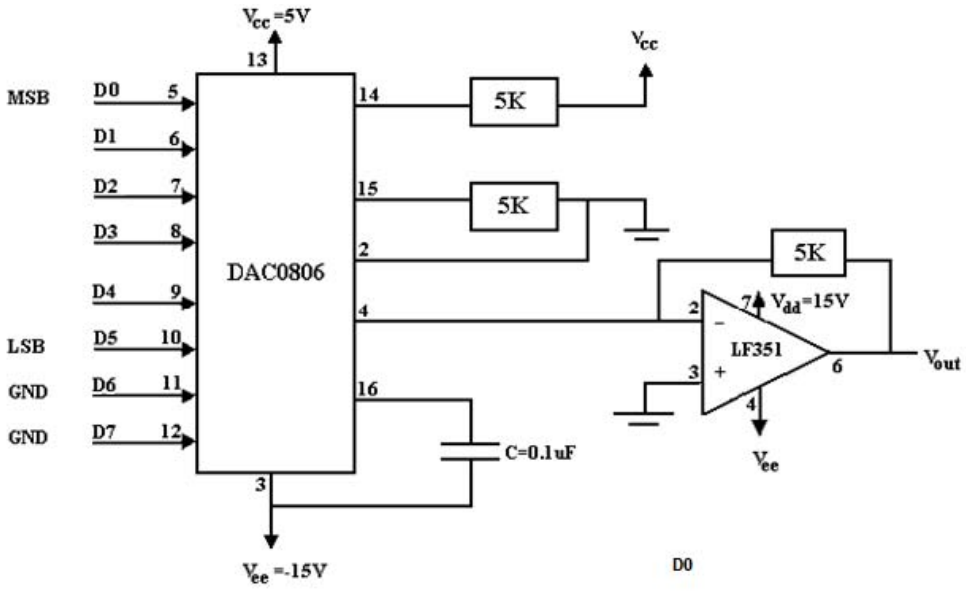

Figure 7. Digital to Analog Conversion Circuit.

analog currents I_{out}. The Op-Amp then converts these currents into analog voltages. The resulting voltage is V_{out}. The maximum output voltage desired for this project was 5 volts. So a referenc voltage of 5 volts was used at pin 14 of the DAC0806 chip. With six bits, the maximum decimal value obtainable is 63 which in binary form translates to 111111 and the minimum is 0 which I binary is 000000. So, with six bits a total of 64 numbers are possible. The voltage resolution o the converter can then be calculated as 5 volts/64 ≈ 0.08. This shows that voltage at the output o the op-amps can be increased in steps of 0.08 volts up to a maximum of 5 volts.

$$V_{out} = V_{ref} * \left(\frac{A_1}{2} + \frac{A_2}{4} + \frac{A_3}{8} + \frac{A_4}{16} + \frac{A_5}{32} + \frac{A_6}{64} + \frac{A_7}{128} + \frac{A_8}{256}\right) volts$$

Since, V_{ref} is 5 volts and A7 and A8 are connected to ground,

$$V_{out} = 5 * \left(\frac{A_1}{2} + \frac{A_2}{4} + \frac{A_3}{8} + \frac{A_4}{16} + \frac{A_5}{32} + \frac{A_6}{64} + \frac{0}{128} + \frac{0}{256}\right) volts$$

So,

$$V_{out} = 5 * \left(\frac{A_1}{2} + \frac{A_2}{4} + \frac{A_3}{8} + \frac{A_4}{16} + \frac{A_5}{32} + \frac{A_6}{64} + \right) volts$$

When all the bits i.e. A_1, A_2, A_3, A_4, A_5, and A_6 are HIGH, then:

$$V_{out} = 5 * \left(\frac{1}{2} + \frac{1}{4} + \frac{1}{8} + \frac{1}{16} + \frac{1}{32} + \frac{1}{64} + \right) volts = 4.92\ volts$$

The output is not exactly 5 volts because not all 8 bits are taken into consideration. When all the bits i.e. A_1, A_2, A_3, A_4, A_5, and A_6 are LOW, then:

$$V_{out} = 5 * \left(\frac{0}{2} + \frac{0}{4} + \frac{0}{8} + \frac{0}{16} + \frac{0}{32} + \frac{0}{64} + \right) volts = 0\ volts$$

The resolution can be calculated as:

$$V_{out} = 5 * \left(\frac{0}{2} + \frac{0}{4} + \frac{0}{8} + \frac{0}{16} + \frac{0}{32} + \frac{1}{64} + \right) volts = 0.08\ volts$$

A maximum of 5 volts was chosen because it was determined during testing that the transmitter knobs when moved horizontally or vertically produced a maximum voltage level which was close to 5 volts. The complete set of measurements are given in Table 1.

The voltages presented above were measured using a digital multimeter and mechanically moving the knobs of the transmitter. The

Transmitter Knobs and Output Voltages		
	Minimum Voltage	**Maximum Voltage**
Aileron Knob (Right control stick moved horizontally)	0.085 volts (Extreme Left)	4.83 volts (Extreme Right)
Elevator Knob (Right control stick moved vertically)	0.26 volts (Extreme Up)	4.88 volts (Extreme Down)
Rudder Knob (Left control stick moved horizontally)	0.080 volts (Extreme Left)	4.89 volts (Extreme Right)
Throttle Knob (Left control stick moved vertically)	0.25 volts (Extreme Up)	4.80 volts (Extreme Down)

Table 1 .- ADC Minimum and Maximum Voltages.

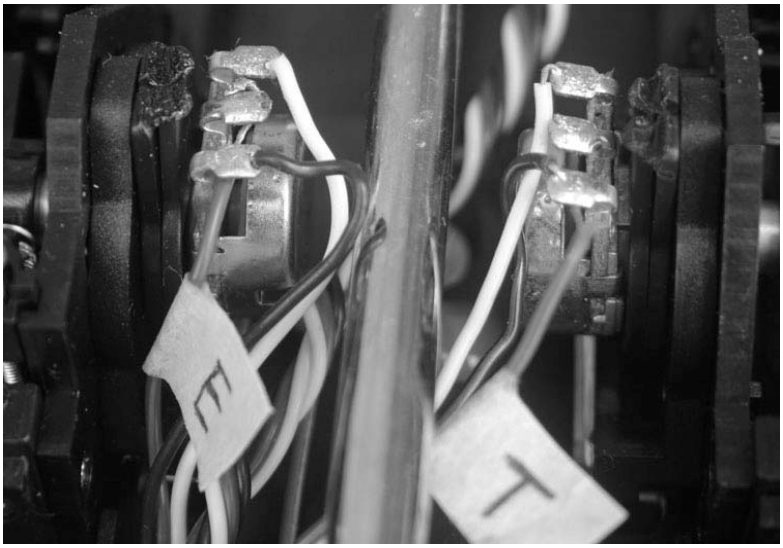

Figure 8. Potentiometer Detail of the Transmitter

transmitter has three different colored wires. Each knob has two potentiometers (Figure 8), and each potentiometer has the three colored wires. The descriptions of the wires are given below:

a) Black wire (left pin) = 5 volts (V_{cc})
b) White wire (right pin) = Ground (GND)
c) Red wire (middle pin) = Potentiometer Output.

The red wires controlled the amount of voltage supplied to the rotors of the helicopter. Since the objective of this project was to be able to control the helicopter via software, all four red wires were disconnected from their respective potentiometers. Each red wire was labeled with the function it provided such as aileron, elevator, rudder, and throttle and then connected to the output of the corresponding op-amp. The interconnections between the interface and transmitter are shown in Figures 9 and 10.

A 24-pin solderless connector was used to plug into the breadboard end. Pins 1 through 12 of the DB-25 matched perfectly with pins 1-12 in the 24-pin connector, pins 13-25 of the DB-25 were connected to the 24-pin connector as shown in Figure 6.

Figure 11, show the details of the connections between the computer, hardware interface and the remote control transmitter.

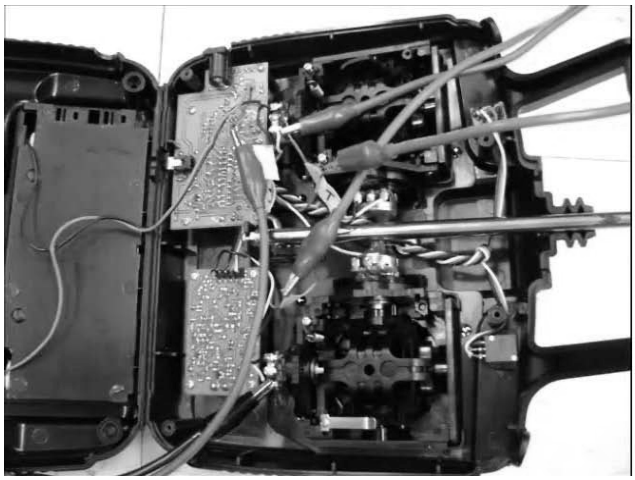

Figure 9. Transmitter Connections.

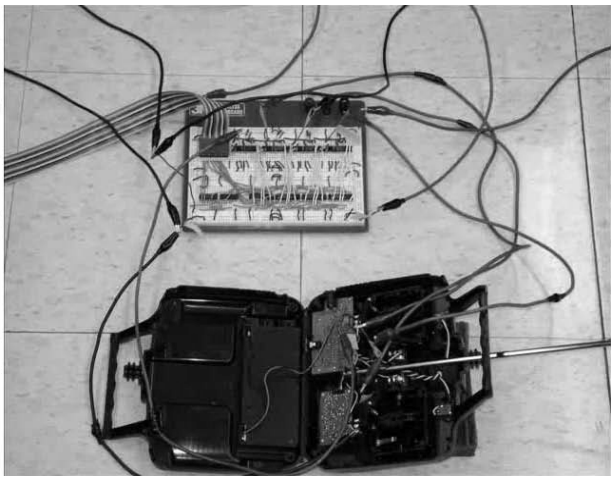

Figure 10. Interface Connections.

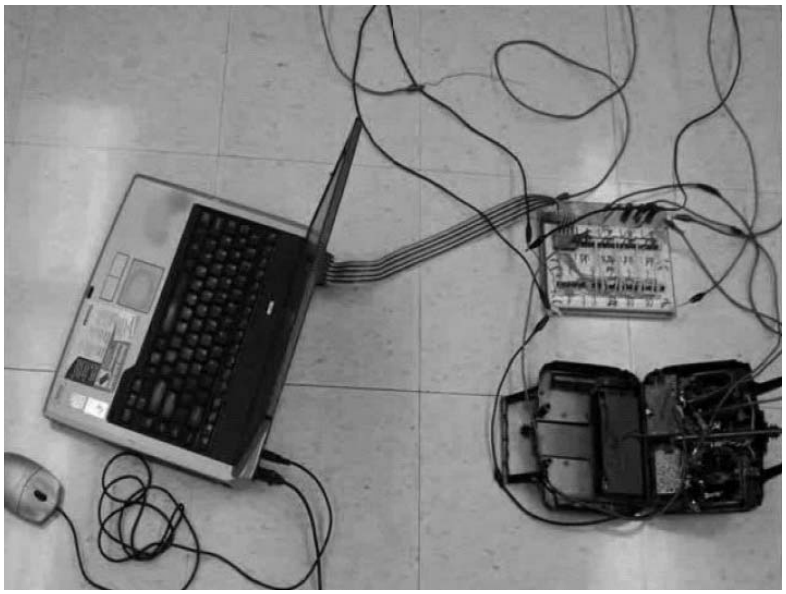

Figure 11. Complete Hardware Setup.

Software Description

The software of choice for this project was MATLAB 9.0. Because of the availability of input/output and Fuzzy Logic toolboxes on MATLAB it was chosen to create the controller algorithm for the system. To control the flight of the quad-rotor a Graphical User Interface (GUI) was developed. A screenshot of the GUI designed for this project is shown in Figure 12.

The GUI consists of different windows with menu buttons to select the flying mode. Two flying-control modes are available: Auto Pilot and Manual Mode. In Auto Pilot mode Fuzzy Logic rules are used to guide the helicopter whereas in Manual mode the flight operator can provide desired speeds to the helicopter's four rotors. The GUI's message box is used to indicate the current flying mode. It is also used to display the message "it is safe to turn off the helicopter" once the flight operator presses the GUI's QUIT button. In total the GUI consists of four windows. The Aileron window displays the speed that is currently applied to the Aileron rotor, the Elevator window displays the speed

that is currently applied to the Elevator rotor, the Rudder window displays the speed that is currently applied to the Rudder rotor, and the Throttle window display the speed that is currently applied to the Throttle rotor. Below each window there is a horizontal scroll bar which allows the flight operator to adjust the speed of the corresponding rotor. The range of values for the four windows is 0-63, however in practice a minimum value of 0 and a maximum value of 54 were actually used because these values facilitate the creation of symmetrical triangular membership functions in Fuzzy Logic.

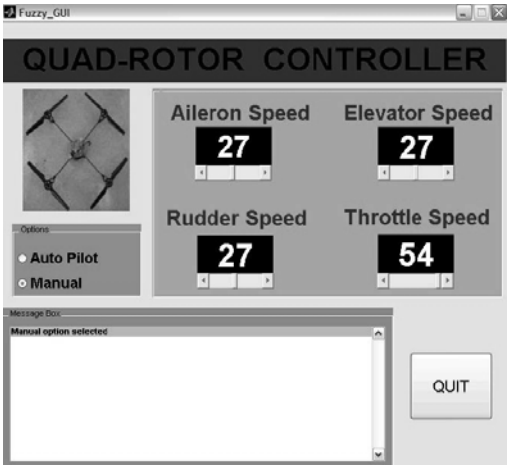

Figure 12.- GUI of Quad-rotor Controller.

The function of each flight control is given below:

1) Aileron Operation or Roll (Left-Right motion)

This operation allows the flight operator to control the left and right motion of the helicopter. When the horizontal scroll bar below the Aileron window is moved to the right, the helicopter tilts more to the right, and when the horizontal scroll bar is moved to the left, the helicopter tilts more to the left. The range of voltages and values for this control is shown in Figure 13.

Output Voltage and Helicopter Orientation due to Aileron Scrollbar Position		
Scroll bar position	**Op-amp Voltage**	**Helicopter Orientation**
Aileron Speed 54	High (~ 5 volts)	Left Tilt
Aileron Speed 0	Low (~ 0 volts)	Right Tilt

Figure 13.- GUI of Aileron Speed.

2) Elevator Operation or Pitch (Forward – Backward motion)

This operation allows the flight operator to control the backward and forward motion of the helicopter. When the horizontal scroll bar below the Elevator window is moved more to the right, the helicopter flies faster backwards, and when the scroll bar is moved more to the left, the helicopter flies faster forward. The range of voltages and values for this control is shown in Figure 14.

Output Voltage and Helicopter Orientation due to Elevator Scrollbar Position		
Scroll bar position	**Op-amp Voltage**	**Helicopter Orientation**
Elevator Speed 54	High (~ 5 volts)	Backward Flight
Elevator Speed 0	Low (~ 0 volts)	Forward Flight

Figure 14.- GUI of Elevator Speed.

3) Rudder Operation or Yaw (Clockwise – Counterclockwise Rotation)

This operation rotates or spins the helicopter. When the horizontal scroll bar below the Rudder window is moved more to the right, the helicopter will spin faster clockwise, and when the scroll bar is moved more to the left, the helicopter will spin faster counterclockwise. The range of voltages and values for this control is shown in Figure 15.

Output Voltage and Helicopter Orientation due to Rudder Scrollbar Position		
Scroll bar position	**Op-amp Voltage**	**Helicopter Orientation**
Rudder Speed 54	High (~ 5 volts)	Clockwise
Rudder Speed 0	Low (~ 0 volts)	Counterclockwise

Figure 15.- GUI of Rudder Speed.

4) Throttle Operation or Power (Ascend – Descend motion)

This operation allows the flight operator to make the helicopter rise or fall. When the horizontal scroll bar below the Throttle window is moved more to the right, the speed of all the rotors is reduced and the helicopter descends, however, when the scroll bar is moved more to the left, the speed of all the rotors is increased and the helicopter begins to climb. The range of voltages and values for this control is shown in Figure 16.

Output Voltage and Helicopter Orientation due to Throttle Scrollbar Position		
Scroll bar position	**Op-amp Voltage**	**Helicopter Orientation**
Throttle Speed 54	High (~ 5 volts)	Descend
Throttle Speed 0	Low (~ 0 volts)	Ascend

Figure 16.- GUI of Throttle Speed.

When the program is launched, the GUI starts up in default mode. The default settings include Manual mode with the speed value 27 for aileron, elevator, and rudder rotors and 54 for the throttle rotor. The value 27 is the medium value in the range from 0 to 54. This value was chosen as the start up value to keep the helicopter balanced. The throttle is kept at the minimum value (54) to avoid sudden lift off. In Manual mode the flight operator has full control over the helicopter and there is no use of the Fuzzy controller. However, in Auto Pilot mode Fuzzy Logic rules are used to govern the helicopter's flight.

Fuzzy Logic Controller Design

Some basic Fuzzy Logic rules were used to control the helicopter in Auto Pilot mode. Determining the output, which is the voltage that will be sent from the controller/signal conditioner/transistor to the helicopter rotor is time consuming when done by hand, as is done below, but this calculation takes only thousandths of a second when done by a computer. A human expert provides a description of how best to control the helicopter in some natural language (e.g., English). This "linguistic" description is then loaded into a fuzzy controller. Regardless, the choice of the linguistic variable has no impact on the way that the fuzzy controller operates; it is simply a notation that helps to facilitate the construction of a fuzzy controller via Fuzzy Logic. Such abbreviations help keep the linguistic descriptions short yet precise. Table 2 shows the linguistic variables used for this project. The linguistic variables were used to describe the positions of the horizontal scrollbar that is present in the GUI below each control window. Since the scrollbar moves horizontally, values such as left and right were used to describe its position.

Table 2: Linguistic Variables	
VLL	Very Large Left
LL	Large Left
ML	Medium Left
SL	Small Left
Z	Zero
SR	Small Right
MR	Medium Right
LR	Large Right
VLR	Very Large Right

Table 2. Linguistic Quantification Table.

Table 2 gives the linguistic quantifications that were used to specify a set of rules (a rule-base) that captures the knowledge needed to control the helicopter[7-8]. Fuzzy set theory defines Fuzzy Operators on Fuzzy Sets. The problem in applying this is that the appropriate

fuzzy operator may not be known. For this reason, Fuzzy logic usually uses IF-THEN rules, or constructs that are equivalent. Table 3 shows the IF-THEN rules used in our implementation.

In fuzzy logic, a membership function is used to represent the degree of truth that a particular parameter has as a member of a fuzzy set. Triangular Membership functions were used to quantify the meaning of the linguistic values.

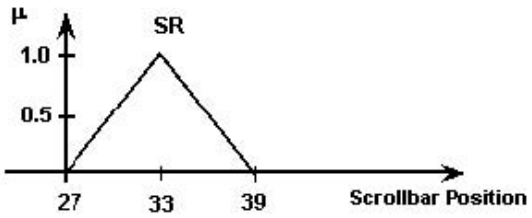

Figure 17. Membership Function for the Linguistic Value "Small Right".

Consider, for example, Figure 17. This is a plot of a function μ versus scrollbar position that takes on special meaning. The function μ quantifies the certainty that scrollbar position can be classified linguistically as "Small Right." To understand the way how a membership function works, it is best to perform a case analysis where it is shown how to interpret it for various values of *scrollbar position.*

• If scrollbar position = 20 then $\mu(20) = 0$, indicating that we are certain that scrollbar position = 20 is not "small right" (SR).

• If scrollbar position = 30 then $\mu(30) = 0.5$, indicating that we are halfway certain that scrollbar position = 30 is "small right" (we are only halfway certain since it could also be "zero" with some degree of certainty since this value is in a "gray area" in terms of linguistic interpretation).

• If scrollbar position = 33 then $\mu(33) = 1.0$, indicating that we are absolutely certain that scrollbar position = 33 is what we mean by "small right."

• If scrollbar position = 39 then $\mu(39) = 0$, indicating that we are certain that scrollbar position = 39 is not "small right" (actually, it is "medium right").

The membership function quantifies, in a continuous manner, whether values of a scrollbar position belong to (or are members of) the set of values that are "small right" and hence it quantifies the meaning of the linguistic statement "scrollbar position is small right." This is why it is called a membership function. It is important to recognize that the membership function in Figure 17 is only one of many possible definition of the meaning of "scrollbar position is small right". For example, a bell-shaped function, a trapezoid, or many others membership functions could also be used.

Table 3 **IF -THEN** Rules
1. **If** scrollbar position is "VLL", **Then** restoring signal needed is "VLR"
2. **If** scrollbar position is "LL", **Then** restoring signal needed is "LR"
3. **If** scrollbar position is "ML", **Then** restoring signal needed is "MR"
4. **If** scrollbar position is "SL", **Then** restoring signal needed is "SR"
5. **If** scrollbar position is "Z", **Then** restoring signal needed is "Z"
6. **If** scrollbar position is "VLR", **Then** restoring signal needed is "VLL"
7. **If** scrollbar position is "LR", **Then** restoring signal needed is "LL"
8. **If** scrollbar position is "MR", **Then** restoring signal needed is "ML"
9. **If** scrollbar position is "SR", **Then** restoring signal needed is "SL"

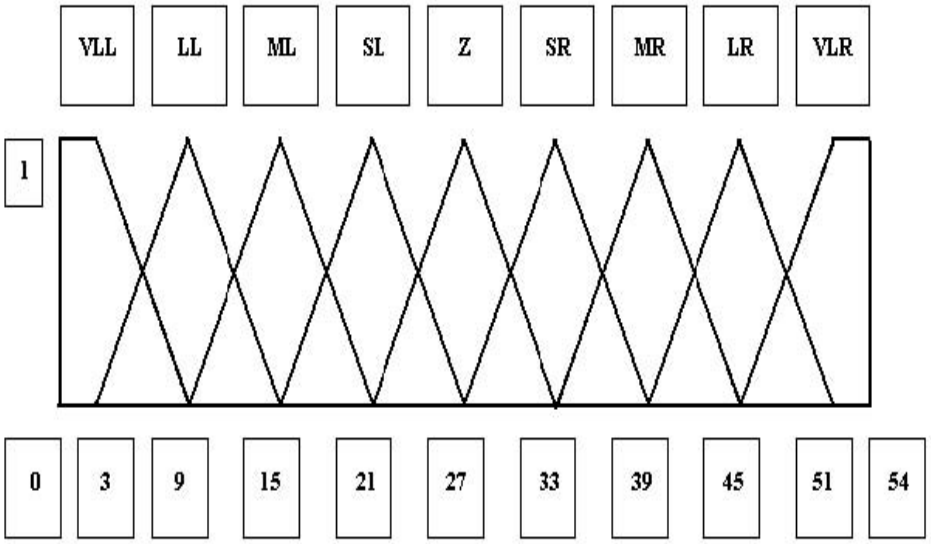

Figure 18. Triangular Membership Functions.

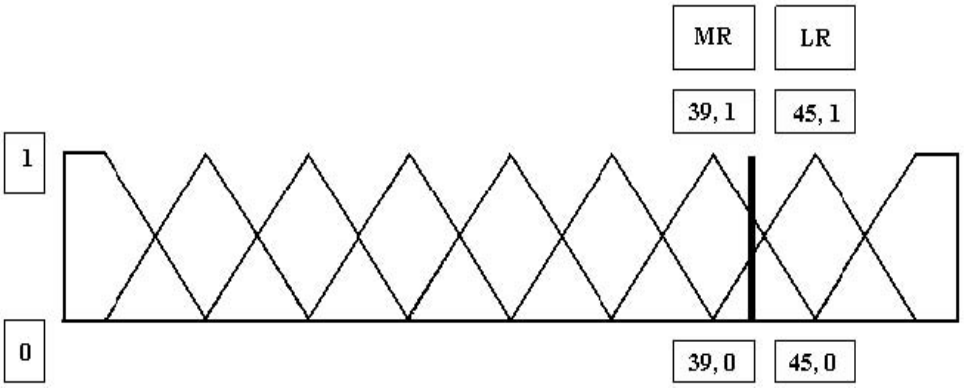

Figure 19. Speed Above Target Value.

Figure 18 shows the complete triangular membership function values and linguistic quantifications used in the implementation of the fuzzy controller.

In Auto Pilot mode, the throttle control was fixed at a value of 45. This value was selected because it provides enough voltage for the helicopter to hover above the ground. To test the controller behavior, random numbers were used to cause the scrollbar positions of the aileron, elevator, and rudder controls to be changed. This would change the output voltage of the Op-Amp whose scrollbar position was altered. As a result of this the corresponding rotor of the helicopter would sense a change in voltage, and the helicopter would become unbalanced. To correct this situation Fuzzy Logic rules were used. This situation is best explained with an example as presented next. Assume that a random number causes the rudder speed to increase from the target speed of 27 to 41. This represents an increase of 14 above the “set point”. Some action is needed to "pull" the speed back to 27. Intuitively, the voltage to the rotor needs to be reduced a little. The speed of 41 intersects the “Medium Right” triangle at 0.67 and the “Large Right” triangle at 0.33 (as shown in Figure 19). This is determined as follows:

The slope of the MR triangle where the vertical line intersects is:

$$Slope = \frac{0-1}{48-39} = -\frac{1}{6}$$

The point of intersection on the MR triangle is:

$$1 + \left(\left[-\frac{1}{6}\right] * (41-39)\right) = 0.67$$

Similarly, the slope of the LR triangle where the vertical line intersects is:

$$Slope = \frac{1-0}{48-39} = \frac{1}{6}$$

The point of intersection on the LR triangle is:

$$0 + \left(\left[\frac{1}{6}\right] * (41-39)\right) = 0.33$$

The next step is to draw output determining triangles with their height determined by the values obtained above. Only the LL and ML triangles are shown in Figure 20 because the vertical line intersects the LR and MR triangles. These "effect" triangles will be used to determine the controller output, which is the voltage that needs to be applied correct the rudder's horizontal scrollbar.

The result is affected by the widths of the triangles and will be calculated. The LL triangle has a height of 0.33 and the ML triangle has a height of 0.67, because these are the intersect points for their matching "cause" triangles. The control output is determined by calculating the point at which a fulcrum would balance the two triangles:

The Area of the LL triangle is: ½*12*0.33=1.98

The Area of the ML triangle is: ½*12*0.67=4.02

The controller output voltage is calculated by finding the point on the scrollbar position axis where the "weight" (area) of the triangles will balance. It is assumed that all the weight of the LL triangle is at 9 and all the weight of the ML triangle is at 15. We are looking for the balance point.

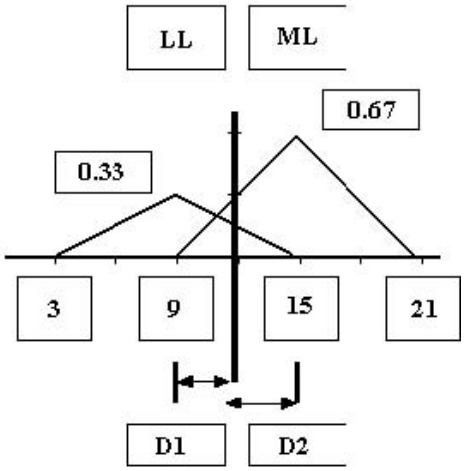

Figure 20.- Determination of Restoring Voltage

The position of the scrollbar i.e. the balance point is found out with the following two equations:

0.33 * D1 = 0.67 * D2 ---- [1]

D1 + D2 = 6 ---- [2]

Where D1 is the fulcrum distance from 2.4 V, and D2 is the fulcrum distance from 2.36 V.

So,

D1 = 6 - D2

Solving the system of equations by substituting (6 - D2) for D1 in Equation 1 gives D2 = 1.98 and D1 = 4.02, therefore the balance point is at 13 (taking only the integer value). To bring the rudder's scrollbar back to the optimum value of 27, the following calculation is done:

$$Correct\ Value = \frac{13+41}{2} = 27$$

Figure 21 .- Quad-rotor Taking Off

Figure 22 .- Quad-rotor in Mid-flight

Results

Figures 21 and 22, show the testing of the helicopter. In manual mode the helicopter did respond to signals sent by the human controller via the GUIs in the computer. The commands are given using either the computer's mouse or the left right arrows of the keyboard. At the beginning, it was a bit difficult to control the helicopter using the computer's mouse/left-right arrows because due to lack of matching in the hardware components, the helicopter reacted very quickly to small changes in voltage levels. Because of the above, even if the same numerical value was assigned to each op-amp, the resulting output voltage would be different. This caused the helicopter to become unstable and difficult to balance. After some offset corrections, the helicopter finally took off in manual mode and was able to fly as expected. The demonstration and a complete video of the flight can be found at http://video.yahoo.com/watch/1335099.

In Auto Pilot mode, the controller operates as a truly open-loop control system. This is due because the transmitter operates in one way direction only. Then, it is not possible at this time, for the fuzzy controller to receive feedback from the actual location of the helicopter. To have any idea about where the helicopter is actually flying and if whether or not the four rotors have their correct values or if the helicopter is approaching an obstacle the controller can be switched to manual control. Because of this limitation, the control of the helicopter in Autopilot mode was very unstable. Depending on the weather conditions (wind speed) the flight control of the helicopter was easier to achieve or not at all.

Future Improvements

In the near future a closed-loop and more stable system will be built by installing sensors on the helicopter that would continuously provide feedback about the helicopter's current location and obstacle positions to the fuzzy controller. This information would give the controller some estimation of where to guide the helicopter.

Conclusions

A fuzzy logic based controller has been designed with emphasis an easy to implement hardware and software platform. Testing of the system was performed and flight autonomy using fuzzy control was observed. Results showed how Fuzzy Logic makes the behavior of the controller more similar to how a human pilot would command a flying device. Other outcomes of this project include the design of an interface that allowed the implementation of a wireless controller. In addition, this project showed how Fuzzy Logic minimizes the use of complex mathematical models and equations to

achieve the control of complex systems, and the potential that Fuzzy Logic possesses to extend its applications to future control mechanisms.

References

1. Kreindel, E., Rothschild, D., "Design of Optimal Decoupling Controllers Application to VTOL", *International Journal of Systems Science*, Vol. 10, Issue 8.

2. Porltlock, J., Cubero, S., "Dynamics and Control of a VTOL Quad-Thrust Aerial Robot", in *Mechatronics and Machine Vision in Practice*, Springer, Berlin, 2007.

3. Kochersberger, K., Et Al, "Human Supervisory Control of an Autonomous VTOL using LIDAR and Stereovision", Proceedings of the *AUVSI Unmanned Systems North America*, San Diego, CA, June 10-12, 2008.

.

4. "Unmanned Air Vehicles Roadmap, 2002-2027," Office of the Secretary of Defense, 2002.

5. "Unmanned Air Systems Roadmap, 2005-2030," Office of the Secretary of Defense, 2005.

6. http://logix4u.net/Legacy_Ports/Parallel_Port/A_tutorial_on_Parallel_port_Interfacing.html

7. http://www.ece.osu.edu/~passino/FCbook.pdf.

8. http://www.fuzzy-logic.com/ch3.htm

Biographical Information

Fernando Rios-Gutierrez was born in Mexico City, Mexico. He graduated with a bachelor's degree in Electrical Engineering and Communications from the National Polytechnic Institute, Mexico City, in 1978. After graduating, he worked as a product designer engineer for the National Cash Register Company, Mexico, where he participated in the design of High-Frequency Switching Power Supplies. In 1983, he joined the Sciences Institute of the Autonomous University of Puebla, Mexico, as an Instrument Design Engineer and lecturer. In 1992, he accepted a tenured position as an Assistant Professor at the Electrical Engineering Department of the Universidad de las Americas, Puebla, Mexico. He pursued graduate studies at the Electrical Engineering and Computer Sciences Department of Tulane University, New Orleans LA, where he was awarded the M.S. degree in Computer Engineering in 1998, and a Ph.D. in Electrical Engineering in 2000. Since September 2008 he is tenure track Assistant Professor at the Mechanical and Electrical Engineering Technology department Georgia Southern University. His main research interests include robotics, remote sensing, robot's learning techniques, digital systems, and microprocessor applications.

Dinesh Baniya Kshatri was born in Nepal. He received a B.Sc. in Electrical and Computer Engineering from the University of Minnesota Duluth in 2008, and M.Sc. degree in Electrical Engineering from Yale University in 2009. Currently, he is a Ph. D. student at Stanford University. His research interests include Robotics, Fuzzy Logic and Digital Systems.