Fuzzy Logic Approach in Real-time UAV Control

Eusebiu Marcu

**Polytechnics University of Bucharest, Bucharest, 060042 Romania (Tel: +40768516413; e-mail: marcueusebiu@gmail.com)*

Abstract: The current article presents a fuzzy logic approach for creating an artificial intelligent and autonomous pilot for unmanned aerial vehicle. The article presents a new way of identifying the position of the UAV relative to a destination point, based on two angles: deflection angle and relative angle. The major advantage of the presented method (over other artificial intelligent methods like genetic programming or neural networks) is the way of taking decisions in real-time, without the need of presenting any domain specific data to the system before running the simulation.
In this article, a professional flight simulator – Microsoft Flight Simulator X (FSX) – is used and its software development kit (SDK), and therefore, all computations are specific to this flight simulator. The aircraft that is used is a real UAV – General Atomics RQ/MQ-1 Predator – an FSX add-on.

Keywords: expert systems, fuzzy logic, unmanned aerial vehicle (UAV), flight simulators (FS), flight dynamics model (FDM).

1. INTRODUCTION

Aircraft engineering is an engineering domain in a continuous change. The need of smarter aircrafts that does not need human intervention has grown along with the technology. Once the computing power has grown, aerospace engineers start wondering if it is possible that an aircraft can fly by itself. Many researches are building models for aircrafts that can fly by them self's with respect to other entities (radar, aircraft) that are near the its position, models that have an input from a number of sensors. Recent research on this area includes multiple approaches which include fuzzy control (Ervin et al., Wu et al., Smith et al.), adaptive control, genetic algorithms control and even combinations of these (Kurnaz et al.). This article presents a new way of identifying the position of the UAV relative to a destination point, based on two angles: deflection angle and relative angle. The major advantage of the presented method (over other artificial intelligent methods like genetic programming or neural networks) is the way of taking decisions in real-time, without the need of presenting any domain specific data to the system before running the simulation. Also, the system described in this article sustains all flight phases: take-off, climb, cruise, descend and land.

This kind of research can make flying safer than traditional way of flying by reducing the human factor to zero. These models may also be applied to other industries like rail or car transport.

This article will provide a method (based on fuzzy logic) of computing the aircraft's commands that will make an aircraft fly by itself from start point to destination point. Both initial and final conditions for start and destination points are given. The initial and destination point in this article are two runways. Each runway is given by its latitude, longitude, elevation, length and heading.

2. UNMANNED AERIAL VECHICLE (UAV)

An UAV is an aerial vehicle with no human crew on-board. From the control point of view, there are two main categories of UAV. The first category consists of models that are controlled remotely and the second one consists of models that are self-controlled based on a predefined flight path. The present paper will present a method to control an UAV that does not require remote controlling nor a predefined path to achieve its goal – Drew (2005), Fahlstrom, Gleason (2009).

Some UAV examples include:

Boeing Condor - controlled through satellites

MQ-1 Predator - sensors and ground control center

TAM-5 - classic autopilot

TIHA - ground controlled.

The UAVs have a wide area of functions like transport, scientific research, armed attacks, surveillance, reconnaissance, and search and rescue.

The United States Department of Defense (DoD) introduced the term Unmanned Aircraft System (UAS) in order to replace the term Unmanned aerial vehicle. A typical UAS consists of a Unmanned Aircraft (UA), a Control System (CS), a Data-link and some other support equipment.
This study uses as UAV the General Atomics RQ/MQ-1 Predator – Fig. 1.

Fig. 1. General Atomics RQ/MQ-1 Predator

2.1 Flight simulators and flight dynamics model

Very important parts in designing a UAV are simulations and flight simulators typically do them. A flight simulator tries to recreate artificially the flight of an aircraft and flight environment. From the software point of view, it consists of a number of modules used to model the actual aircraft and environment. These modules are built using a number of graphics libraries like Microsoft DirectX or OpenGL to have a visual on the aircraft and environment. The second part of a flight simulator is the flight dynamics model. This includes the equations that govern the aircraft during flying, how the aircraft responds the control commands or external environment.

The input consists of control commands (aileron for rolling, elevator for elevation/descend, rudder for yaw and throttle for controlling the aircraft's engine + autopilot corrections), atmospheric disturbances and flight condition. Based on this input and equations of motion, the FDM produces an output – aircraft position/attitude, velocity, acceleration, etc.

The most notable flight simulators are Flight Gear, Outerra, OpenEaagles and Microsoft Flight Simulator X. This article is focused on Microsoft Flight Simulator X and its software development kit (SDK).

3. FUZZY LOGIC

Fuzzy logic – Zadeh (1965) – is a form of first-order logic that can determine the value of a statement as being true or false as a number between 0 and 1 – True = 1, False = 0. In fuzzy logic applications, a number of fuzzy sets and linguistic variables are defined.

A fuzzy set is a pair of a set *S* and a membership function *m*. The membership function is defined as:

$$m : S \to [0,1] \tag{1}$$

Fuzzy set theory defines fuzzy operators on fuzzy sets. Usually, fuzzy logic uses IF-THEN rules in order to apply the fuzzy operator that might be unknown. A fuzzy logic rule can be expressed as:

IF variable IS value

THEN execute_action

or

IF variable1 IS NOT value1 AND variable2 IS value2

THEN execute_action.

One main advantage over some artificial intelligence techniques (like genetic programming – Marcu (2010), Barlow (2004) – or neural networks) is that the computation is being made in real-time. Therefore, there is no need for supplying specific domain data to the system.

4. VARIABLES AND CONSTANTS

Using Microsoft Flight Simulator X SDK, the flight data can be retrieved in real time.

Usually, initial and final conditions are given. Both initial and final conditions can be the location (latitude and longitude), the attitude, the heading – heading is the angle of the aircraft direction and True North – and air speed of the aircraft.

The final conditions are used for determining the current location of the aircraft in our system. To fly an aircraft, the pilot can execute a number of commands that control the aircraft. These controls are executed using the yoke, rudder pedal and the throttle lever. The yoke and rudder pedal modifies the deflection of control surfaces - aileron, elevator and rudder (three commands) and the throttle lever is used to control the power or thrust of the aircraft. For these four commands, correspond a number of four *fuzzy linguistic variables*: the first three values are defined in *(-1, 1)* interval and the fourth is defined in *(0, 1)* interval. These four variables are called control parameter variables.

The control parameters are modifying the state parameters. The state parameters are altitude, heading, and distance to destination. The pseudo-position parameters (which are a form of state parameters) – deflection angle and relative angle – are defined using the state parameters.

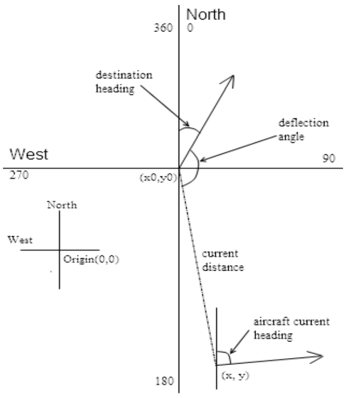

Fig. *2.* Pseudo-position parameters/angles

The relative angle (ρ) is defined as the difference between the aircraft's current heading – h_c and the destination heading – h_d. The deflection angle (δ) is defined as the angle formed by the destination direction and aircraft's current position vector. Therefore, one can write – based on Fig. 2:

$$\rho \in [0,360), \delta \in [0,360) \tag{2}$$

5. MATHEMATICAL BACKGROUND

In order to compute deflection angle and the relative angle, some mathematical calculation has to be performed.

The relative angle is given by the formula:

$$\rho = h_c - h_d \tag{3}$$

If $\rho < 0$, then $\rho \leftarrow 360 + \rho$.

For deflection angle, the following steps will determine the value:

translate current aircraft position by destination vector – so the destination point becomes the origin;

$$(x^1, y^1) = (x - x_0, y - y_0) \tag{4}$$

where (x, y) - current position of the aircraft

rotate the aircraft position by destination heading h_d degrees using the rotation matrix $R(h_d)$ – so the destination heading h_d becomes 0;

$$\begin{cases} x_r^1 = x^1 \cos(h_d) - y^1 \sin(h_d) \\ y_r^1 = x^1 \sin(h_d) + y^1 \cos(h_d) \end{cases} \tag{5}$$

apply the *asin* function.

$$\delta = 180 - \sin^{-1}\left(\frac{x_r^1}{d}\right) \tag{6}$$

where d is the distance to the destination in the current position.

The destination – a runway (or any other point in space) – will be retrieved from the list of possible runways of the airport. The airport and the destination runway are found by checking both the distance and the runway length. Therefore, the airport that is the closest of the aircraft's current position will be chosen and based on its identification number – in Microsoft FSX that is the ICAO string – the longest runway is found. This runway is given by its heading and altitude. This information is critical when the aircraft is preparing it's descend and landing procedure.

6. FUZZY LOGIC MODEL

6.1 Fuzzy logic controller design

A classic fuzzy logic system is built in two phases. The first phase is defining the systems variables (linguistic variables 5.3), the fuzzy database and the inference engine (rules database 5.3) –this phase can be viewed as preparing the meta-data of the fuzzy logic controller. The second phase is building the fuzzy logic controller which reads the sensors raw data, pre-process the raw data which transforms into fuzzy logic specific data in order to perform the fuzzy logic operations: fuzzification (transforming the fuzzy logic specific data into degrees of membership), inferencing the data(the rules are applied over the data) and defuzzification. After this last operation, the output data is determined and presented to the simulator using its software interface.

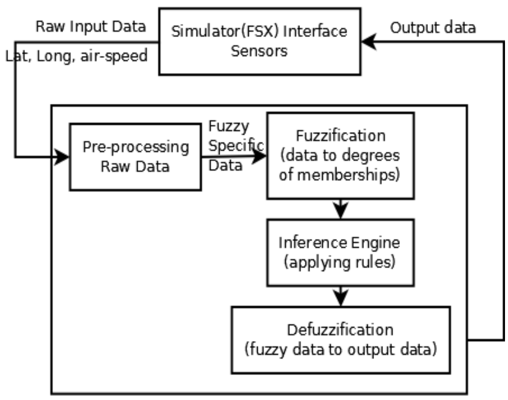

Fig. 3. Fuzzy logic controller design

The raw input data read from the sensors is aircraft's location (spatial coordinates that include the altitude), aircraft's heading, distance to the destination (mid/end-point) and destination spatial coordinates. Based on these inputs the pre-processing operation computes the fuzzy logic specific data. To get a better view of the aircraft's orientation in space, the attitude data is also needed. The specific data (pseudo-position) is computed (according to 4) using the heading and location of the aircraft and destination point.

6.2 Fuzzy sets

"A fuzzy set is a class of objects with a continuum of grades of membership. Such a set is characterised by a membership (characteristic) function which assigns to each object a grade of membership ranging between zero and one" – see Zadeh (1965), Goguen (1967).

Given (2), the space around the destination and the aircraft can be partitioned in 16 zones – Fig. 4. Therefore, one can define 17 fuzzy sets for each specific angle (near 0 and 360 there are defines two zones). Given two values for relative and deflection angle, one can tell in which zone is aircraft located and what is it's heading. Based on this information, one can define the fuzzy sets for each specific angle.

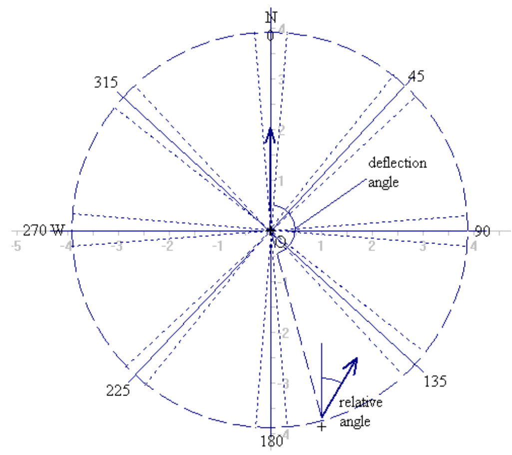

Fig. 4. Space partitioning

The fuzzy sets are defined as in Fig 5.

```
FuzzySet _0_5 = new FuzzySet("_0_5" + s,
    new TrapezoidalFunction(0, 5,
                TrapezoidalFunction.EdgeType.Right));
FuzzySet _0_45 = new FuzzySet("_0_45" + s,
    new TrapezoidalFunction(0, 5, 40, 45));
...
FuzzySet _355_360 =
    new FuzzySet("_355_360" + s,
    new TrapezoidalFunction(355, 360,
                TrapezoidalFunction.EdgeType.Left));
```

Fig. 5. Fuzzy sets definitions using AForge .NET Framework

Beside the sets for the specific angles, the sets for other linguistic variables are defined – for altitude, distance, yoke, rudder, throttle lever.

6.3 Linguistic variables

In fuzzy logic applications, linguistic variables are used in rules definitions and they express facts about system's state and how the system should react – Zadeh et al (1996). The linguistic variables must not overflow or underflow the fuzzy sets data definitions.

Once the fuzzy sets are defined, the linguistic variables can be defined. In this research, there are defined a number of eight linguistic variables: Altitude, Distance, DeflectionAngle, RelativeAngle, YokeX, YokeY, Rudder, and ThrottleLever. These linguistic variables are of two kinds: **conditional** and **action** linguistic variable. Altitude, Distance, DeflectionAngle, and RelativeAngle are condition variables; YokeX, YokeY, Rudder, and ThrottleLever are action variables.

The Altitude linguistic variable is defined using three fuzzy sets: Low, Medium and High. The Distance linguistic variable is also defined using three fuzzy sets: Near, Medium and Far.

The specific angles – DeflectionAngle and RelativeAngle – are defined using 17 fuzzy sets – Fig. 5.

Yoke (X and Y), Rudder are defined using three fuzzy sets – Positive, Zero and Negative. The ThrottleLever linguistic variable is defined using three fuzzy sets: Idle, Medium, Full.

```
// Specific angle (Input)
LinguisticVariable angle = new LinguisticVariable(name, 0, 360);
angle.AddLabel(_0_5); angle.AddLabel(_0_45);
angle.AddLabel(_40_50); angle.AddLabel(_45_90);
angle.AddLabel(_85_95); angle.AddLabel(_90_135);
angle.AddLabel(_130_140); angle.AddLabel(_135_180);
angle.AddLabel(_175_185); angle.AddLabel(_180_225);
angle.AddLabel(_220_230); angle.AddLabel(_225_270);
angle.AddLabel(_265_275); angle.AddLabel(_270_315);
angle.AddLabel(_310_320); angle.AddLabel(_315_360);
angle.AddLabel(_355_360);
return angle;
```

Fig. 6. Adding a linguistic variable using AForge.NET Framework

Each fuzzy linguistic variable has a name, a minimum value, and a maximum value.

For Altitude and Distance linguistic variables, the minimum is 0 and the maximum is infinity.

For DeflectionAngle and Relative angle, the minimum is 0 and the maximum is 360.

For YokeX and YokeY, the minimum is -1 and the maximum is 1. For ThrottleLever, the minimum is 0 and the maximum is 1 – in this case, the values multiplied by 100 represent percents of the throttle lever when pushed forward.

Using these fuzzy linguistic variables and given the space partitioning – Fig. 5 – rules can be defined. Also, they are added to the fuzzy database of linguistic variables.

6.4 Fuzzy Rules

The fuzzy rules are expressing the way of how the system should react in different states of the system.

```
_InferenceSystem = new InferenceSystem(
    fuzzyDB, new CentroidDefuzzifier(1000));

// climb 1
_InferenceSystem.NewRule(
    "1", "IF Height IS Low THEN YokeY IS YokeYPositive");
```

Fig. 7. Creating an inference system and adding rules using AForge.NET Framework

In Fig. 7, the rules for Altitude conditional variable are defined – for Distance variable, three rules are also defined. For DeflectionAngle and RelativeAngle, there are $289\left(17 \cdot 17\right)$ rules defined.

All these rules are added to the inference system together with the method (defuzzification) of obtaining the output value for each input value: CentroidDefuzzier – see AForge.NET (2010).

6.5 System workflow

In order to build a fuzzy agent that controls the aircraft, an application was build that connects to both main modules: flight simulator (Microsoft FSX) and Fuzzy Module.

The main application – host application – is written entirely in C#.NET 2.0 using Microsoft Visual Studio 2010 Express Edition. The connection to Microsoft FSX was accomplished using Microsoft FSX SDK – SimConnect class.

The Microsoft FSX data structures are defined as classic structures and each member of the structure is decorated using a custom attribute – FsxVariable – that has a name, unit and an internal Microsoft FSX type – Fig. 8.

```
[StructLayout(LayoutKind.Sequential, CharSet = CharSet.Ansi, Pack = 1)]
internal struct AircraftPosition
{
    [MarshalAs(UnmanagedType.ByValTStr, SizeConst = 256)]
    [FsxVariable(
        "Title", null, SIMCONNECT_DATATYPE.STRING256)]
    public String Title;
    [FsxVariable(
        "Plane Latitude", "degrees", SIMCONNECT_DATATYPE.FLOAT64)]
    public double Latitude;
    [FsxVariable(
        "Plane Longitude", "degrees", SIMCONNECT_DATATYPE.FLOAT64)]
    public double Longitude;
    [FsxVariable(
        "Plane Altitude", "feet", SIMCONNECT_DATATYPE.FLOAT64)]
    public double Altitude;
    [FsxVariable(
        "PLANE ALT ABOVE GROUND", "feet", SIMCONNECT_DATATYPE.FLOAT64)]
    public double GroundAltitude;
    [FsxVariable(
        "PLANE HEADING DEGREES TRUE", "radians", SIMCONNECT_DATATYPE.FLOAT64)]
    public double HeadingToTrueNorth;
}
```

Fig. 8. Defining a data type for connecting to Microsoft FSX

When connecting to the simulator, each structure's members is checked for these custom attribute to get the necessary data to register it in the flight simulator.

Fig. 9 displays the diagram of the main activities. The host application connects to Microsoft FSX using its SDK. After a successful connection, the fuzzy module is defining the linguistic variables – Altitude, Distance, DeflectionAngle, RelativeAngle, YokeX, YokeY, Rudder and ThrottleLever. After the creation of variables the database, inference system and rules are created or defined. At this time the infrastructure of the system – structural information/metadata – has been built. At each step – usually a simulation frame – the values for deflection angle and relative angle are determined. These two numbers gives the exact position and orientation to both aircraft and destination relative to each other. These angles and the other data – distance and Altitude – are set as inputs in the inference system that was previously created. The inference system is executed in order to get the output of the fuzzy agent. If the position of the aircraft is the desired one – a solution has been found – the simulation stops. Otherwise, based on the response of the agent, the commands of the aircraft are modified to get the next position – see **action** variables.

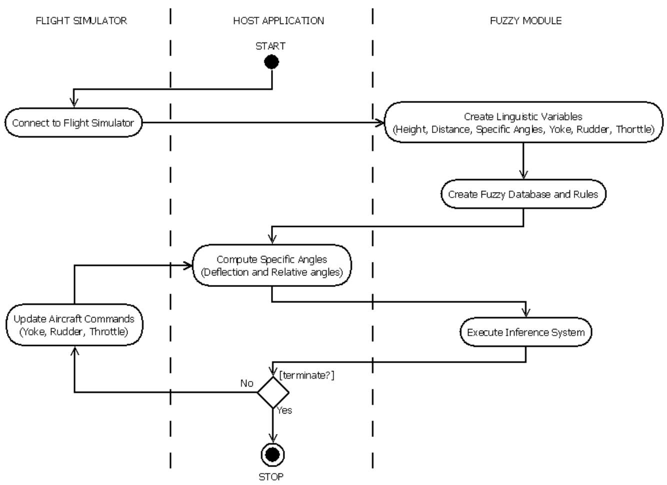

Fig. 9. System-workflow activity diagram

```
// Setting inputs
_InferenceSystem.SetInput("Altitude", Altitude);
_InferenceSystem.SetInput("Distance", Distance);
_InferenceSystem.SetInput("RelativeAngle", RelativeAngle);
_InferenceSystem.SetInput("DeflectionAngle", DeflectionAngle);

// Setting outputs
YokeY = _InferenceSystem.Evaluate("YokeY");
YokeX = _InferenceSystem.Evaluate("YokeX");

if (YokeChanged != null)
    YokeChanged(null, EventArgs.Empty);
```

Fig. 10. Running the inference systems on yoke

The host application is subscribed to different events of fuzzy module. Fig. 10 is an example of the YokeChanged event that is triggered when the position of the Yoke action variables has changed. Once the event was triggered, the host application will modify the control parameters of the aircraft – Fig. 11.

```
internal static void ModifyYoke(AircraftControl yokeData)
{
    if(AircraftSim.IsConnected)
        AircraftSim.SetDataOnSimObject(
            Definitions.AircraftControl,
            SimConnect.SIMCONNECT_OBJECT_ID_USER,
            SIMCONNECT_DATA_SET_FLAG.DEFAULT, yokeData);
}
```

Fig. 11. Modifying the control parameters for Microsoft FSX

7. TEST DATA AND RESULTS

In this article, the initial and destination point in this article are two runways. Each runway is given by its location – latitude, longitude – elevation, length and heading. The conditions for this test are ideal – no atmospheric disturbances.

Initial runway – starting point:

Table 1. Initial runway data

Variable	Value
Latitude	44°34′16″N
Longitude	026°05′06″E
Elevation	288 meters
Runway length	3500 meters

Final runway – destination point:

Table 2. Destination runway data

Variable	Value
Latitude	45°12′59′N
Longitude	26°58′42″E
Elevation	344 meters
Runway length	2734 meters

For this test, following plots are displayed: longitude-latitude – this is the actual path of the aircraft during flight and altitude-time.

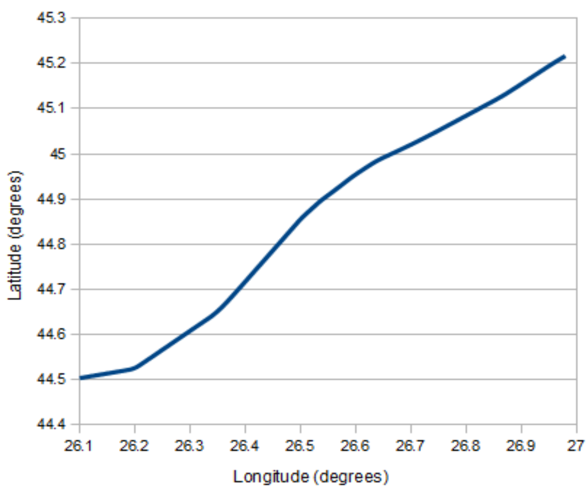

Fig. 12. Longitude-latitude plot

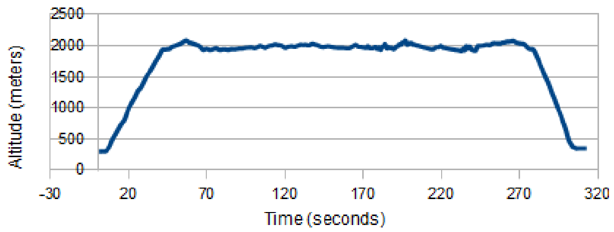

Fig. 13. Altitude-time plot – cruise at 2000 meters

8. CONCLUSIONS

This paper introduces a new fuzzy logic approach in UAV/aircraft control. The main enhancement of this method is that the decisions are made in real-time, without the need of providing specific domain data to the system.

The paper introduces a way of finding the accurate position of the aircraft's current position using two angles: deflection angle and relative angle. Also, the paper introduces a new way of dividing the linguistic variables in two kinds: conditional and action linguistic variables. Also, this paper can be used as a tutorial on how to define and use a fuzzy logic application in a flight simulator of other simulation applications.

Fig. 14 shows the take-off of the GA MQ-1 Predator using a self-controlled fuzzy logic agent.

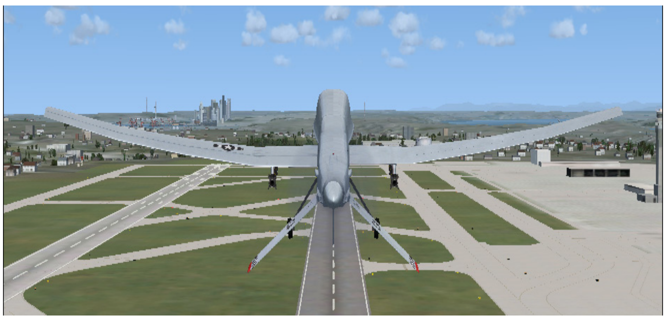

Fig. 14. GA MQ-1Predator in Microsoft FSX controlled by a *self-controlled* fuzzy agent – take off

Fig. 15. GA MQ-1 Predator in Microsoft FSX controlled by a fuzzy agent – turn left

AKNOWLEDGEMENTS

The work has been funded by the Sectoral Operational Programme Human Resources Development 2007-2013 of the Romanian Ministry of Labour, Family and Social Protection through the Financial Agreement POSDRU/6/1.5/S/19.

REFERENCES

Drew, J.G. (2005), *Unmanned Aerial Vehicle End to End Support Considerations*, RAND Corporation

Goguen, Joseph A. (1967) *L-fuzzy sets*. Journal of Mathematical Analysis and Applications, 18: 145–174

Zadeh, L. A. (1965), *Fuzzy sets*. Information and Control 8 (3) 338–353.

Zadeh, L. A. et al. (1996). *Fuzzy Sets, Fuzzy Logic, Fuzzy Systems*, World Scientific Press

Marcu, E (2010) – *New Genetic Programming Approach in UAV Control*, Unpublished article

Barlow, G. J. (2004). *Design of Autonomous Navigation Controllers for Unmanned Aerial Vehicles Using Multi-objective Genetic Programming*. North Carolina State University. Raleigh, NC

Fahlstrom, P. G., Gleason, Thomas J., (2009), *Introduction to UAV Systems*, 3rd edition

Ervin, J. C., Alptekin, S. E., DeTurris D. J., *Optimization of the Fuzzy Logic Controller for an Autonomous UAV*, e Department of the Navy, Office of Naval Research (N00014-01-1-1049 and N00014-02-1-0754)

Wu, P., Narayan, P., Campbell, D., Lees, M., Walker R., (2006), *A High Performance Fuzzy Logic Architecture for UAV Decision Making*, Proceedings IASTED International Conference on Computational Intelligence, San Francisco, ACTA Press

Smith III, J. F., Nguyen, T. V., (2006), *Fuzzy Logic Based UAV Allocation and Coordination*, Informatics in Control Automation and Robotics, 15, 81-94

Kurnaz, S., Cetin, O., Kaynak, O., (2010), *Adaptive Neuro-Fuzzy Iinference System Based Autonomous Fight Controlof Unmanned Air Vehicles*, Expert Systems with Applications 37, 1229–1234

Microsoft FSX – Microsoft Flight Simulator FSX, http://www.microsoft.com/games/pc/flightsimulatorx.aspx

AForge .NET (2010), http://www.aforge.com.