mathematics

MDPI

Article

Optimization of Fuzzy Logic Based Virtual Pilot for Wargaming

Alexandr Stefek * and Petr Frantis

Department of Informatics and Cyber Operations, Faculty of Military Technology, University of Defence, Kounicova 65, 662 10 Brno, Czech Republic; petr.frantis@unob.cz

* Correspondence: alexandr.stefek@unob.cz

Abstract: This paper deals with the design of an autopilot based on a set of fuzzy controllers. The model of the aircraft that the autopilot controls is defined as a model with 6 degrees of freedom, where the inputs to this model are the settings of the engine thrust (DX), rudder rotation (Dl) and elevators (Dm and Dn). The fuzzy controllers are of the Mamdani type, where the set of parameters defining the controller allow the derivation of membership functions of the input variables and membership functions of the output variables. The parameters of fuzzy controllers are determined by the optimization process. For the purpose of optimization, a fitness function is defined, which derives the simulation parameters from its parameter (vector), and the simulation subsequently performed and evaluated determines whether it is a feasible solution in addition the value of this solution. By this optimization process, the sub-optimal solution is found and is then used to define the settings of the fuzzy controllers and, therefore, the autopilot. This paper contains a description of each step of the solution of the described problem, and with the help of the obtained results, it is determined that the presented procedure allows us to find an autopilot capable of controlling the defined model of the aircraft. In addition, there is a brief description regarding the misconceptions explored during the development of the experiment.

Keywords: autopilot; fuzzy controller; flight simulation model; optimization

Citation: Stefek, A.; Frantis, P. Optimization of Fuzzy Logic Based Virtual Pilot for Wargaming. *Mathematics* **2021**, *9*, 3169. https://doi.org/10.3390/math9243169

Academic Editor: Giampaolo Liuzzi

Received: 31 October 2021
Accepted: 6 December 2021
Published: 9 December 2021

1. Introduction

Modelling is one of the fundamental disciplines that aid human cognition. Very often, its results are used in simulation, which also has a strong place in innovation. In the modern concept of the armed forces, simulation and simulation tools are an integral part of training and decision support systems. The quality and accuracy of these simulation systems are determined by the quality and accuracy of the implemented simulation models. In order for these simulators to be approved for use in a military environment, they must pass a system of military tests to ensure that their results meet the required level of conformance with real systems used in the military. These complex simulation systems consist of many models that interact with each other so that the results of one model affect the inputs of another model. The models used are selected according to the level of accuracy required. The level of accuracy affects the intended use of the simulation system.

This paper deals with a part of the simulation system for air defence purposes. Air defence has sensors and effectors which need a sufficiently accurate model of the aircraft they are to counteract. In terms of sensors, this includes, for example, a reflecting surface, which may change during the course of the aircraft's mission. This is determined by the roll of the aircraft and its position relative to the sensor—the radar. Effectors and their defensive actions are affected by the maneuver of the aircraft.

For this system, a general model of an aircraft moving along a predetermined path must be constructed. Since detailed models of the means of detection and tracking of airborne targets will be implemented in the developed simulation system, it is necessary that the developed model allows us to simulate the aircraft as a detailed three-dimensional

model that moves realistically in space and contains not only positional data but also orientation around all axes and control surface deflections.

In our case, it is, therefore, a simulation model of an aircraft originally developed for a flight simulator. The choice of the model defines the requirements for the pilot who must be able to fly the aircraft in the simulation and thus accomplish the specified mission. The pilot cannot be realized or represented by a human in conditions where a repeated number of simulations are involved. Therefore, the question is: "how can a system be created that would realize the tasks set by the mission?".

Thus, the problem to be solved consists of the simulation model of an aircraft and the simulation model of a pilot flying along a specified trajectory.

Previously published aircraft control systems that satisfy the above requirements have used expertly designed pilot models. The aim of this paper is to show the possibility of an automated control system–pilot design using optimization methods.

2. Literature Review

In general terms, we can look at this task as the creation of a simulation model of an autopilot piloting a simulated aircraft. One possible approach to autopilot design is to model the behavior of a human pilot. In [1], many possible approaches to human pilot models can be found. One of the described models is the quasi-linear model that utilizes the servomechanism theory. This model (and its variants) are easy-to-implement for single input—single output scenarios, but for complex multi-input handling, it becomes very complex to implement. The autopilot problem is usually addressed using automatic control theory as in [2] or [3], which focuses on UAV control issues. The PID assisted control approach is also mentioned in [4], which focuses on the automatic control of a model aircraft simulated in the X-plane flight simulator.

The modelling of a plane and air defence during a particular conflict and its solution has been published in [5–7]. However, in this case, quite a simple aircraft model has been used, and thus some of the attributes mentioned above cannot be considered.

Another approach to solving this problem is the use of fuzzy logic to optimize the fuzzy controller and genetic algorithms, such as in [8], but in which authors are focusing on the control of the ship. Similarly, an adaptive fuzzy controller is used for trajectory control purposes in [9], but with a limitation to its application in ship control.

The determination of the fuzzy logic controller, with the help of optimization, has been published in [10]. In this case, the optimal robot movement, in short-range, was the goal.

The abovementioned fuzzy logic controllers assume a disturbance-free environment. In case of dealing with disturbances, the fuzzy controller should be combined with other methods such as an active disturbance rejection control with a proportional-derivative Takagi–Sugeno fuzzy control as described in [11].

The solution described in this paper focuses on the application of a fuzzy controller in a turbulence-free environment with optimization using genetic algorithms for the purpose of modelling a virtual pilot controlling a simulation model of an aircraft.

3. Proposed System

The proposed system consists of three parts:

- path planner,
- pilot model,
- airplane model.

The input to the path planner is a description of the flight path in the form of a list of waypoints. This block passes these values to the pilot model, which simulates the movements of the aircraft controlled by a real pilot. The outputs from this model are the positions of the individual control elements of the aircraft and are inputs to the airplane simulator model. The output of the simulator model combined with path planner data are used as feedback to the pilot model to simulate "human-in-the-loop" airplane control (Figure 1).

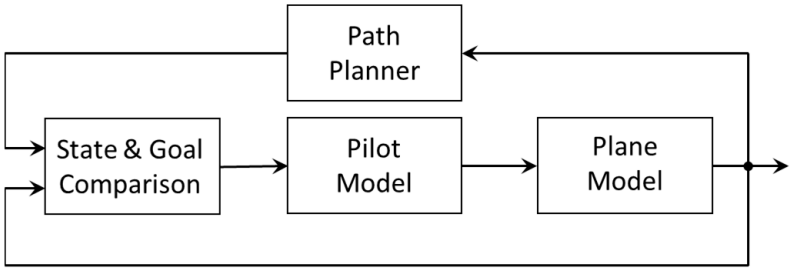

Figure 1. Block scheme of the proposed system.

Figure 1 depict a close loop control with the input taken from a path planner block, which compares the preplanned (war) mission (fly route) with the current output of the plane model block, which contains the aircraft position. If the aircraft finishes a stage of preplanned mission (flies through a waypoint), the path planner block changes its output to the next waypoint. The state and goal comparison block calculates inputs used by the pilot model block controlling the plane model (block).

3.1. Airplane Simulation Model

The simulation of aircraft motion is one of the fundamental problems that must be addressed in military simulations that consider the air domain. The simulation model used is described in detail in [12]. The model is defined in the standard form as:

$$\dot{x} = f(t, x) \tag{1}$$

where x is model state and t is time.

The state of the model consists of variables:

$$\begin{pmatrix} x \\ y \\ z \\ \psi \\ \theta \\ \phi \\ \alpha \\ \beta \\ p \\ q \\ r \\ V \end{pmatrix} \tag{2}$$

where x, y and z are positions, ψ, θ and φ are angular rotations of plane (Figure 2) and α, β are angle of attack and angle of sideslip, p, q and r are angular velocities of the plane relative to the earth in the plane frame of reference, and V is the velocity.

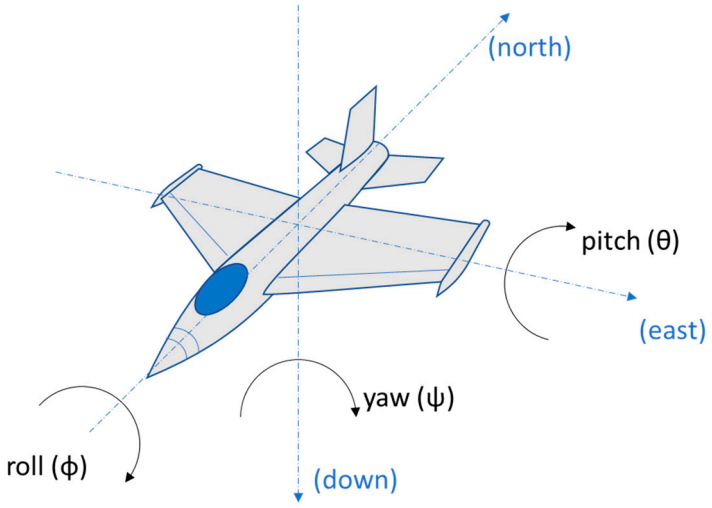

Figure 2. Plane and earth frames of reference.

The state variables can be computed using following equations:

$$\begin{gathered}
\dot{x} = v(-\cos\phi\sin\psi + \sin\phi\sin\theta\cos\psi) + u(\cos\theta\cos\psi) + w(\sin\theta\cos\psi + \sin\theta\cos\phi\cos\psi) \\
\dot{y} = v(\cos\phi\cos\psi + \sin\theta\sin\phi\sin\psi + u\cos\theta\sin\psi) + w(-\sin\phi\cos\psi + \sin\theta\cos\phi\sin\psi) \\
\dot{z} = v\sin\phi\cos\theta - u\sin\theta + w\cos\theta\cos\phi \\
\dot{\psi} = \frac{q\sin\phi + r\cos\phi}{\cos\theta} \\
\dot{\theta} = p + \tan\phi(q\sin\theta - r\sin\theta) \\
\dot{\phi} = q\cos\phi - r\sin\phi \\
\dot{\alpha} = \frac{1}{V\cos\alpha\cos\beta}\left(\frac{Z}{m} + g\cos\theta\cos\phi - \dot{V}\sin\alpha\cos\beta + V\dot{\beta}\sin\alpha\sin\beta - V(p\sin\beta - q\cos\alpha\cos\beta)\right) \\
\dot{\beta} = \frac{1}{V\cos\beta}\left(\frac{Y}{m} + g\cos\theta\sin\phi - \dot{V}\sin\beta - V(r\cos\alpha\cos\beta - p\sin\alpha\cos\beta)\right) \\
\dot{p} = \frac{1}{A - \frac{E^2}{C}}(L + \frac{E}{C}(N + (A - B)pq - Erq) + (B - C)qr + Epq) \\
\dot{q} = \frac{1}{B}(M + (C - A)rp - E(p^2 - r^2)) \\
\dot{r} = \frac{1}{C}(N + E\dot{p} + (A - B)pq - Erq) \\
\dot{V} = \frac{1}{\cos\alpha\cos\beta}(\frac{X + F_T}{m} - g\sin\theta - V(q\sin\alpha\cos\beta - r\sin\beta))
\end{gathered} \tag{3}$$

Aerodynamics coefficients can be computed as follows:

$$\begin{gathered}
C_x = C_{x0} + k_i {C_z}^2 \\
C_y = C_{y\beta}\beta + C_{y\delta l}\delta l + C_{y\delta n}\delta n \\
C_z = C_{z\alpha}(\alpha - \alpha_0) + C_{zq}\frac{ql}{V} + C_{z\delta m}\delta m \\
C_l = C_{l\beta}\beta + C_{lp}\frac{pl}{V} + C_{lr}\frac{rl}{V} + C_{l\delta l}\delta l + C_{l\delta n}\delta n \\
C_m = C_{m0} + C_{m\alpha}(\alpha - \alpha_0) + C_{mq}\frac{ql}{V} + C_{m\delta m}\delta m \\
C_n = C_{n\beta}\beta + C_{np}\frac{pl}{V} + C_{nr}\frac{rl}{V} + C_{n\delta l}\delta l + C_{n\delta n}\delta n \\
L = \frac{1}{2}Slv^2C_l \\
M = \frac{1}{2}Slv^2C_m \\
N = \frac{1}{2}Slv^2C_n \\
u = V\cos\alpha\cos\beta \\
v = V\sin\beta \\
w = \sin\alpha\cos\beta
\end{gathered} \tag{4}$$

where

- δl, δm and δn are the position of the roll, pitch, and yaw control surfaces in radians, respectively (Figure 3),

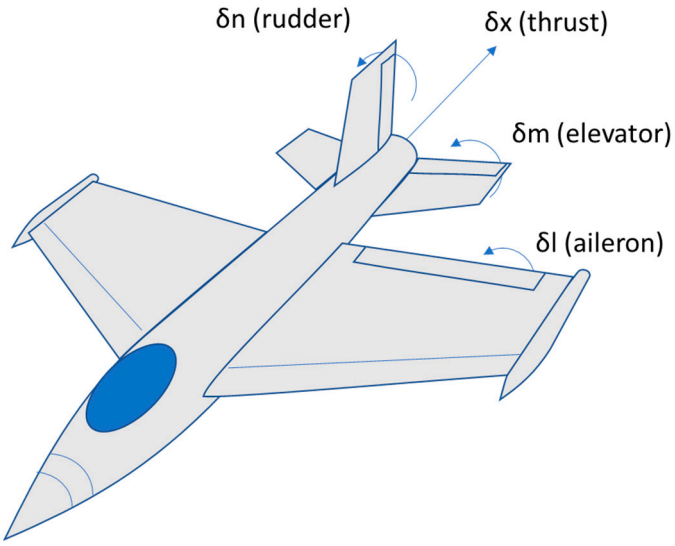

Figure 3. Airplane control surfaces.

- $C_{i\alpha}$ and $C_{i\beta}$ are moment coefficients;
- α_0 is the zero-lift angle of attack;
- C_{ip}, C_{iq} and C_{ir} are rolling, pitching or yawing damping coefficients;
- $C_{i\delta l}$, $C_{i\delta m}$ and $C_{i\delta n}$ are coefficients representing the influences of steering (dihedral effect, inducted rolling and adverse yaw);
- C_{x0} is the zero-lift drag coefficient and k_i is the induced drag coefficient;
- S is the wing surface, l is mean aerodynamic chord and C_L, C_M and C_N are roll, pitch, yaw moment coefficients;
- F_0 is the full thrust of the engine;
- A, B, C, E are inertial momentums of the aircraft;
- m is a mass of the airplane;
- L, M, N are moments due to the aerodynamic force in the center of the airplane mass;
- u, v, w are velocities in the plane of reference.

Aerodynamic force $F_R\left(R_x, R_y, R_z\right)$ can be computed as follows:

$$\begin{aligned} R_x &= -\tfrac{1}{2}\rho S V^2 C_x \\ R_y &= -\tfrac{1}{2}\rho S V^2 C_y \\ R_z &= -\tfrac{1}{2}\rho S V^2 C_z \end{aligned} \tag{5}$$

and transformed into the plane of reference:

$$\begin{aligned} X &= R_x \cos\alpha \cos\beta - R_y \cos\alpha \sin\beta - R_z \sin\alpha \\ Y &= R_x \sin\beta + R_y \cos\beta \\ Z &= R_x \sin\alpha \cos\beta - R_y \sin\alpha \sin\beta + R_z \cos\alpha. \end{aligned} \tag{6}$$

The actual thrust of the engine F_T can be computed from the position of the engine control lever δ_x (Figure 3) in range of $(0,0–1,0)$.

$$F_T = \frac{\rho}{1.225} F_0 \delta_x \tag{7}$$

The non-state variables are:

- g is gravitational force;
- ρ is density of the air and assuming it is constant relatively to the altitude and time, thus $\rho = 1.225\ \mathrm{kg/m^3}$.

3.2. Airplane Simulation Model Implementation

The airplane simulation model has been implemented in the Python programming language [13] using the Runge–Kutta numerical integration method. In addition, the SciPy library [14] has been used for the numerical solution. The implementation of this model has been verified via the original implementation in [12]. As the programming code must use a standard ASCII character set, the δl, δm, δn and δ_x symbols have been replaced by Dl, Dm, Dn and DX variable names. In order to make the text clearer, they will be referred to as such below.

4. Fuzzy Controller

The fuzzy controller used is to be one of the Mamdani [15] or Sugeno [16] type. For the problem discussed in this article, the Mamdani fuzzy system has been chosen. This means that all input variables are fuzzified, then a rule set is applied and the result of the inference system is defuzzified to get a crisp value (see Figure 4).

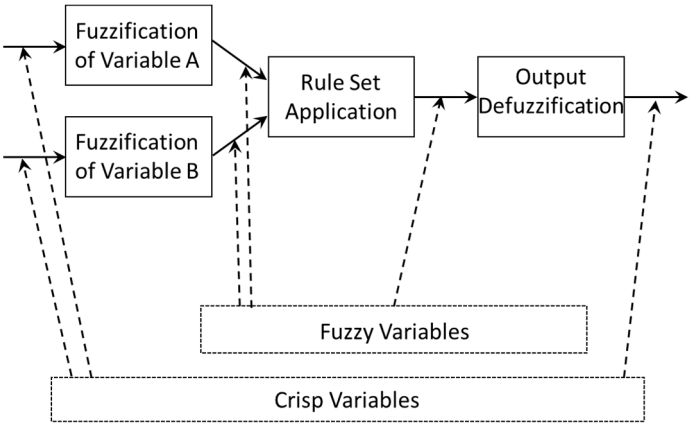

Figure 4. Structure of fuzzy controller.

4.1. Fuzzy Controller as a Part of Pilot

The fuzzy controller used for autopilot modelling has been implemented up to four times, each time for one autopilot output. The variability of the number of the used fuzzy system originated in the option defining a pilot's output by a constant. Such fixation means that the appropriate pilot's output is constant (does not change during the entire simulation).

Figure 5 depict the inner structure of the pilot model block (four fuzzy blocks DX, Dm, Dl, and Dn) taken from Figure 1. (Block scheme of the proposed system). Each of the fuzzy blocks has this inner structure (see Figure 4).

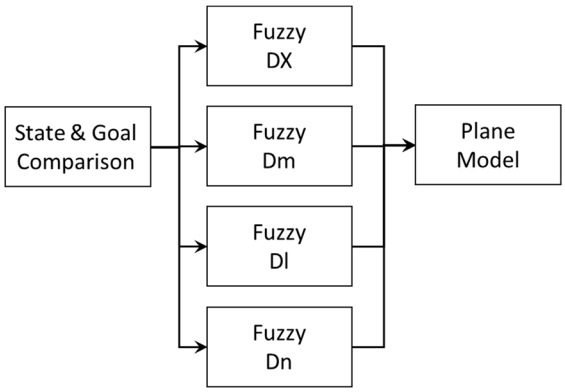

Figure 5. Incorporation of fuzzy controller into pilot subsystem.

Figure 6 is a composition of Figures 1, 4 and 5. It depicts a full three-level structure of the pilot model based on four fuzzy systems connected in close loop control.

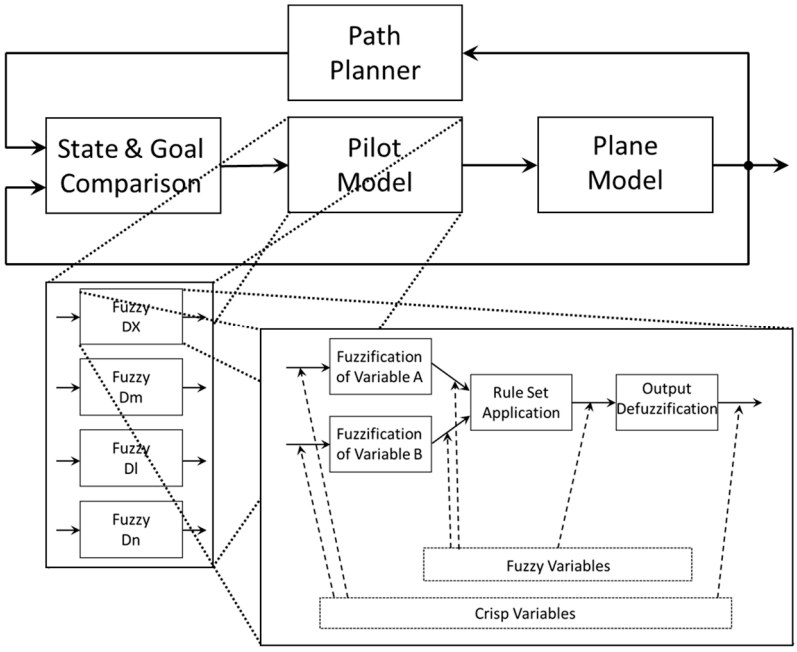

Figure 6. Close loop schema of control with fuzzy controllers.

To simplify the autopilot definition, Dn and Dl output variables have been fixed to value 0. This leads to the inability of the aircraft to make any maneuver on a horizontal plane.

4.2. The Input Variables for Fuzzy Controllers

It has been stated that for the determination of DX and Dm output pilot's variables, the difference between angles α and β and distance $|DP|$ (see Figure 7) should be enough. For the determination of Dn and Dl, the difference between angles γ and δ and distance $|DP|$ (see Figure 7) should be enough. Thus, for the fuzzy controller, there are three different variables on the input of fuzzy systems, but a single fuzzy system uses just two of them. It should be noted that input variables do not have to have the same definition for different fuzzy controllers, even if they are based on the same state values.

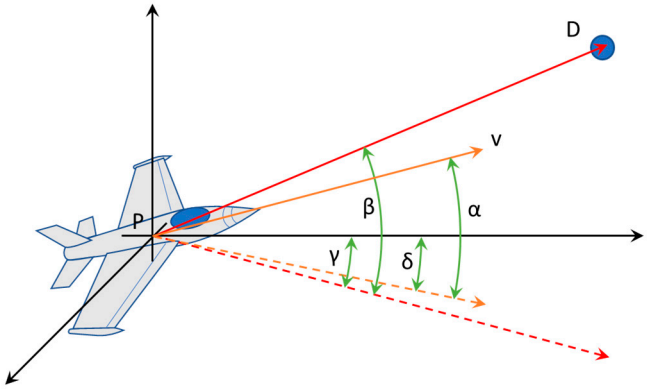

Figure 7. Description of relation of aircraft current state and the destination.

5. Optimization Process

For the design of the autopilot based on fuzzy control systems, an optimization process can be used. Generally, the optimization process is an algorithm where a fitness function is evaluated multiple times and the best value found is considered as an optimum.

5.1. Types of Optimization

There are two basic sets of optimizations: deterministic optimization and stochastic optimization. The second set (stochastic optimization) contains several algorithms developed over the last few decades [17]. It is important to understand that the application of stochastic optimization gives a result that must not be an optimal solution and often is not.

For the problem discussed in this paper, the genetic algorithm has been used.

5.2. Genetic Algorithm Brief Description

The genetic algorithm [18] is based on observations of the behavior of nature where successful individuals have offspring while others do not. The offspring inherit the behavior (or properties) of parents, and this behavior is encoded into their chromosomes. During the production of the offspring, the chromosomes of the parents are combined. Sadly (or luckily), the combination could be imperfect, and thus sometimes, the resulting chromosome is damaged (mutated).

The genetic algorithm, when implemented in a programming language such as Python, has a generation where every member represents a feasible solution with its fitness function value. Such a generation is replaced with the new generation in two steps:

- reproduction (creation of new offspring) of the best members;
- mutation applied with a probability.

The discussed problem needs to find the best (optimal) setup of autopilot. Thus, to get a single fitness function value, a simulation has to be computed. The parameters of the fitness function have to define the autopilot's parameters. The evaluation of the fitness function means that parameters must be translated into a simulation description, the simulation must be executed, observed values must be evaluated during the simulation, and when the simulation ends, the result of the fitness function is derived from the gathered values (see Figure 8). Due to the fact that this simulation can fail, this must be detected, and then parameters given to the fitness function must be marked as an unacceptable solution.

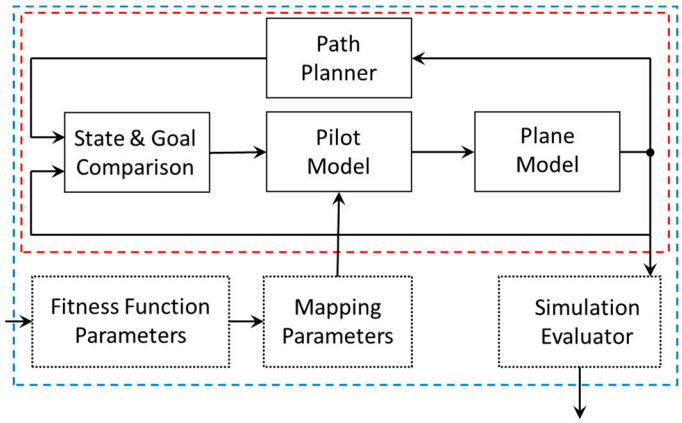

Figure 8. Fitness function based on simulation result.

5.3. Mapping a Chromosome to Simulation

The connection between the chromosome and a full set of autopilot descriptions is crucial. The set autopilot description can be defined in different ways. Below, three of them are mentioned.

5.3.1. The Lowest Chromosome Count

For a fuzzy controller, it is quite common to have five membership functions. To slightly simplify the problem, only the triangular membership function is considered. As the triangular membership function is defined by three parameters (see Figure 9), the full set of parameters is 15 (for five membership functions).

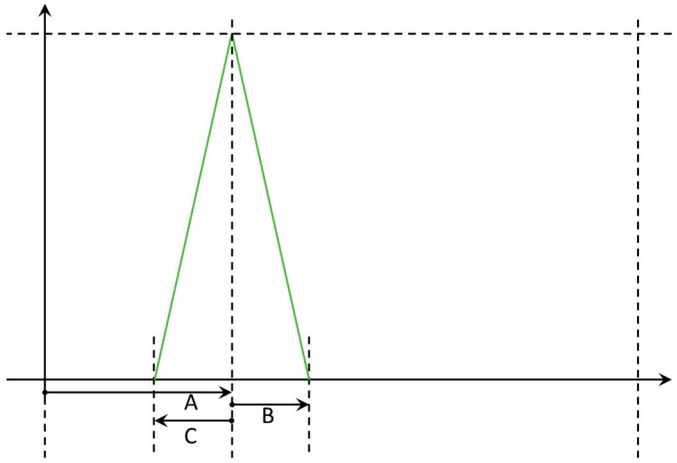

Figure 9. Triangular membership function.

If the leftmost and rightmost membership functions are expected to have their maximum bounds, four parameters could be omitted. The next simplification could be conducted by binding the left bound of the triangular membership function to the maximum of the left neighbor (membership function) and the right bound of the membership function to the maximum of the right neighbor (membership function). Such simplification leads to the ability to describe a full set of membership functions with just three parameters (see Figure 10).

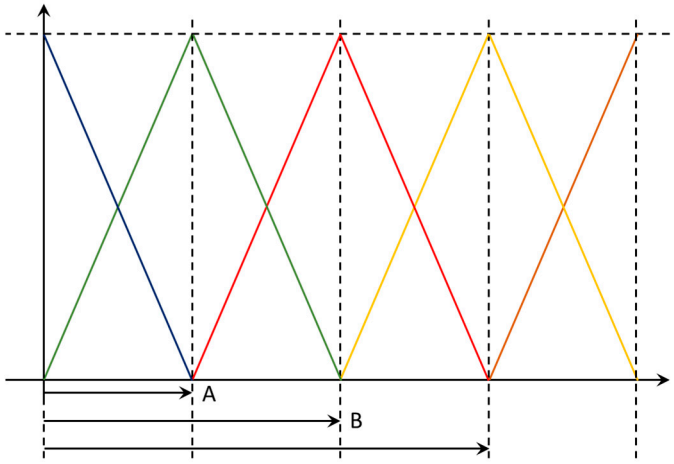

Figure 10. Minimized set of parameters of membership functions.

5.3.2. The Optimal Chromosome Count

In the previous section, the description of the full set of membership functions has been reduced to three parameters. However, such simplification can significantly limit the feasible setup of fuzzy controllers. Therefore, it can be useful to allow the definition of the left and right bound of triangular membership functions (exclude the leftmost and rightmost). A special case is an expectation that the triangular function can be symmetric. If this symmetry is not expected, then the full set of membership functions can be defined by 11 parameters (see Figure 11), otherwise, 8 parameters are able to describe the full set of membership functions.

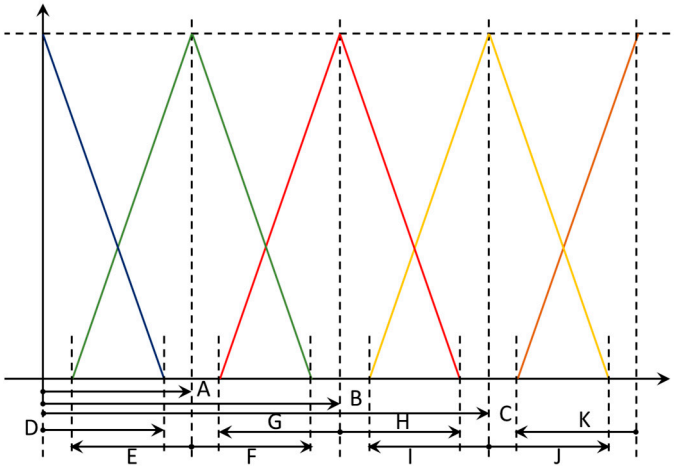

Figure 11. Optimal set of parameters of membership functions.

This setup (11 parameters) has been used in experiments published in [10].

5.3.3. The Extended Chromosome Count

As is noted below, it can be practical to unbind the leftmost and rightmost membership functions. This leads to the requirement of 15 parameters for the description of the full set of membership functions (see Figure 12).

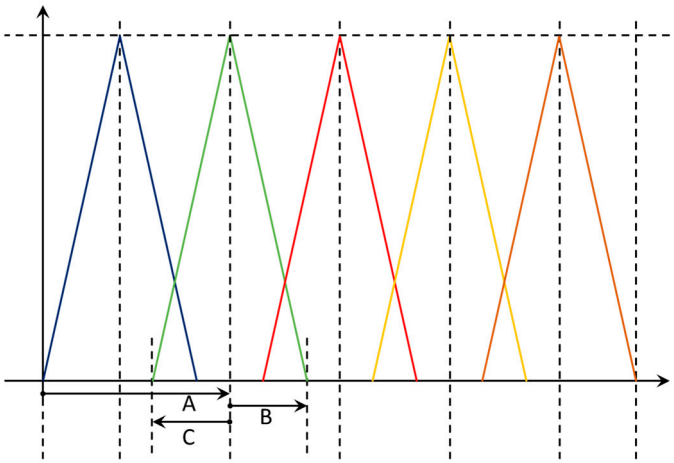

Figure 12. Extended set of parameters of membership functions.

For a fuzzy controller, three variables are expected (two of them as the input and the last one as the output). Such a number of variables leads to 9, 24, 44 or 60 parameters. As the autopilot can have up to four fuzzy controllers, the problem can have up to 240 dimensions. This is the main reason why the problem discussed in this article has been reduced only to movement on the vertical plane.

5.3.4. Rule Sets

The fuzzy controller needs a rule set that allows it to derive the value of the fuzzy variable at the output from input variables. For the autopilot problem, the form of a full cartesian product has been chosen. The full cartesian product means that all fuzzy lexical representations from the first input are combined with all fuzzy lexical representations from the second input, and there is one fuzzy lexical representation of output as a result. Such operations can be represented by a table (see Tables 1 and 2).

Table 1. Dm rules.

Dm\Angle	AHN	ANN	AZZ	APP	AHP
DRC	DMRN	DMMN	DMNN	DMMP	DMRN
DQC	DMRN	DMMN	DMNN	DMMP	DMRN
DMD	DMMN	DMMN	DMNN	DMMP	DMMP
DQF	DMMN	DMMN	DMNN	DMMP	DMMP
DFA	DMMN	DMNN	DMNN	DMMN	DMMP

Table 2. DX rules.

DX\Angle	AHN	ANN	AZZ	APP	AHP
DRC	DXRF	DXRF	DXMM	DXMM	DXQS
DQC	DXRF	DXQF	DXMM	DXQS	DXQS
DMD	DXRF	DXQF	DXMM	DXQS	DXQS
DQF	DXQF	DXQF	DXMM	DXQS	DXRS
DFA	DXQF	DXMM	DXMM	DXRS	DXRS

Both rule sets have been defined by an expert.

The appropriate definitions of abbreviations describing fuzzy lexical representations are:

AHN—Angle high negative
ANN—Angle negative (negative)
AZZ—Angle zero (zero)
APP—Angle positive (positive)
AHP—Angle high positive
DRC—Distance really close
DQC—Distance quite close
DMD—Distance medium (distance)
DQF—Distance quite far
DFA—Distance far away
DMRN—Dm really negative
DMMN—Dm moderate negative
DMNN—Dm neutral (neutral)
DMMP—Dm moderate positive
DMRP—Dm really positive
DXRF—DX really fast
DXQF—DX quite fast
DXMM—DX medium (medium)
DXQS—DX quite slow
DXRS—DX really slow

For maximums of triangular membership functions should be true that:

AHN < ANN < AZZ < APP < AHP
DRC < DQC < DMD < DQF < DFA
DMRN < DMMN < DMNN < DMMP < DMRP
DXRF < DXQF < DXMM < DXQS < DXRS

If the optimization process finds a setup (marked as the best option) where relations above are not valid, rules should be considered as wrongly defined, which has happened in this discussed set of experiments.

5.4. Set of Feasible Solutions of Optimization Problem

For the standard optimization problem, a fitness function and a set of feasible solutions are needed. Fitness function and its evaluation are defined above. The set of feasible solutions is usually defined by volume and function, which excludes values from this volume.

For linear problems, such a set is defined/named as a simplex. If the optimization problem is being solved with the help of stochastic optimization methods such as genetic algorithm, the set of feasible solutions are right n-prism, where n is the number of variables in the optimization problem. For such a case, if all n variable intervals are defined, the right n-prism is also defined. If variables are directly derived from real objects, it is quite common that the left and right bounds of the interval reflect the limits (abilities or properties) of a real object.

The optimization problem, which is mentioned in this article, works with angle, distance, throttle lever position and elevator angle. The elevator angle is limited by the aircraft design to $-\frac{\pi}{4}$ [rad] and $+\frac{\pi}{4}$ [rad]. The throttle lever position has limits naturally described as "idle" and "full throttle", which are projected to interval $\langle 0; 1 \rangle$. As the variables values define the parameters of the triangular function, their limits are taken from appropriate spaces. The angle variable naturally has limits derived from schema depicted in Figure 7. The distance variable has been bound to the interval $\langle 0;\ 30,000 \rangle$ [km] according to expert advice.

Table 3 summarizes intervals of all optimization problem variables.

Table 3. Optimization problem variables limits.

Optimization Variables	Low Bound	High Bound
Related to triangular functions for distance	0 km	30,000 km
Related to triangular functions for angle	$-\pi$ rad	$+\pi$ rad
Related to triangular functions for throttle lever position	0	1
Related to triangular functions for elevator	$-\frac{\pi}{4}$ rad	$+\frac{\pi}{4}$ rad

6. Results and Its Discussion

6.1. Usage of a Different Description for Membership Functions

During the experiments, it has been found that the implementation of autopilot with the help of a fuzzy system the definitions provided by three and eight parameters are inappropriate. After a couple of experiments' hypotheses, it appeared that those rule sets could be wrong. At that stage, the extended set of parameters came into play. With such a setup, the fuzzy pilot adjustment was found, and it can be stated that the extended set of parameters can discover an error in the fuzzy rule definitions.

The simulation time was limited to 500 s. If the airplane went to loopback, the appropriate autopilot setup was marked as an unacceptable solution.

In the section below, the behavior of the (sub)optimal fuzzy autopilot is presented.

6.2. Genetic Algorithm Setup

For the optimization, the Python language has been chosen. This choice has been determined by the availability of different tools needed for this problem. Python has a library for numerical solutions of differential equations, which comes in handy for simulations. In addition, there is the Pymoo library [19] which has already implemented a couple of optimization algorithms. This could also be useful that the new version of Pymoo (0.5) integrated support for distributed optimization employs the Dask [20]. For optimization, the genetic algorithm implemented in the Pymoo library has been chosen; the size of the population has been set to 200 members and the generation count has been set to 500.

6.3. Selected State Variables Evolution in Time

The autopilot should keep an appropriate altitude level. As can be read out from Figure 13, the altitude has been kept with errors less than 20 m. The figure does not keep the aspect ratio.

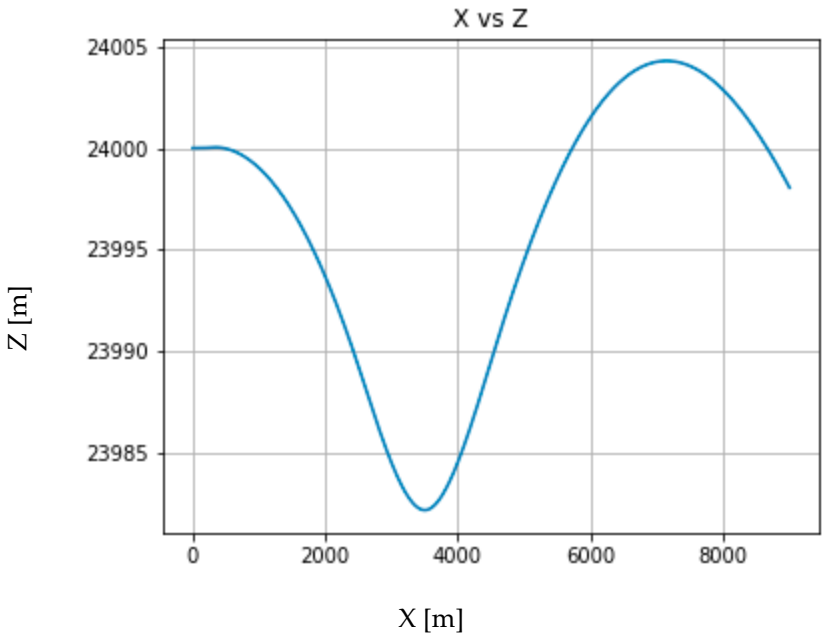

Figure 13. Vertical plane of aircraft movement (plane X [m], Z [m]).

For the aircraft movement, it is also important to keep an appropriate velocity. Figure 13 shows that the velocity [m s^{-1}] during the flight (41 s) has a maximum error of less than 2 m s^{-1} (seed Figure 14).

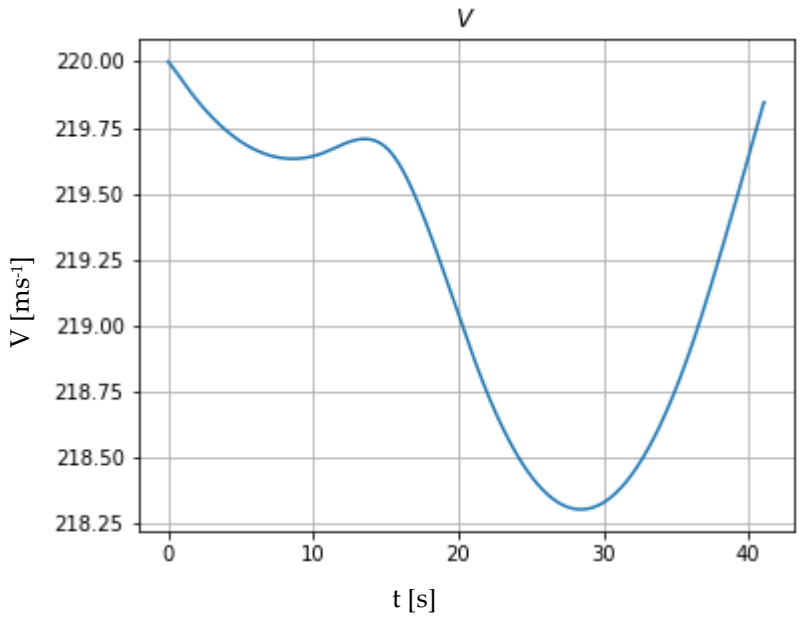

Figure 14. Airplane's velocity V [m s^{-1}] evolution in time t [s].

6.4. Selected Autopilot Outputs Evolution in Time

During the flight, the Dm and DX autopilot's outputs were observed. The results of this observation can be found in Figures 15 and 16. All figures have the time [s] on axis x, and axis y holds values of appropriate autopilot's output.

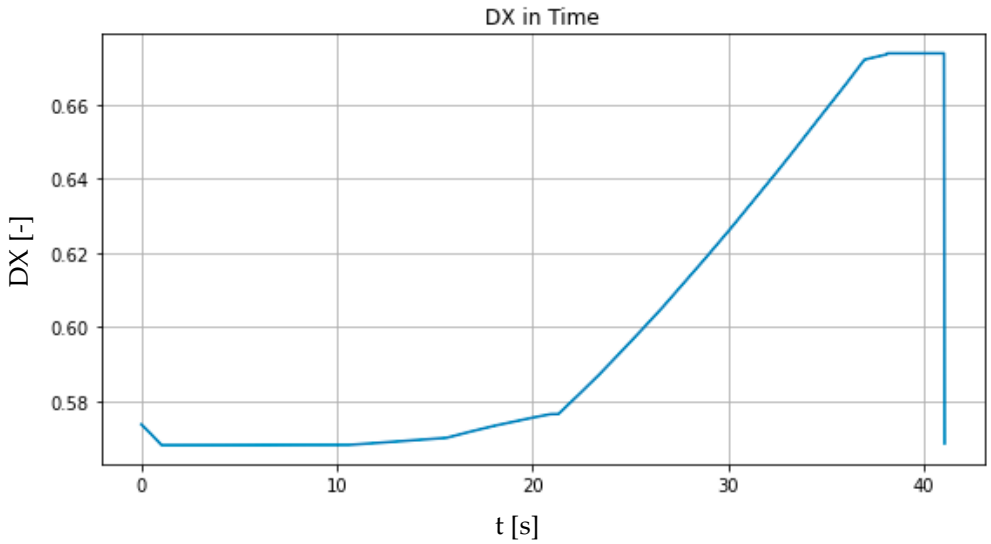

Figure 15. Autopilot DX output [-] evolution in time [s].

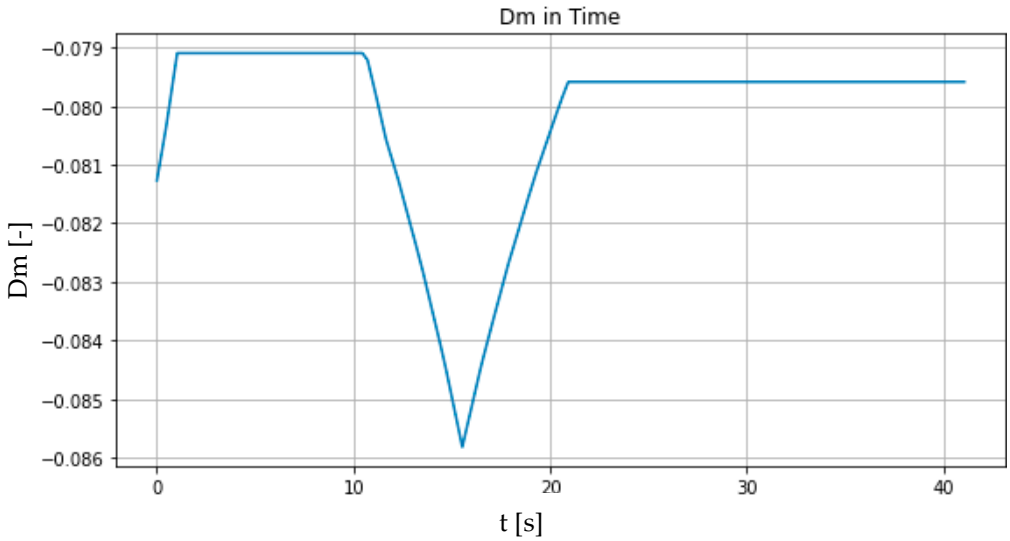

Figure 16. Autopilot Dm output [-] evolution in time [s].

In the simulation, the 15th s is interesting because the autopilot 'noticed' the change in height and changed the Dm. At the 21st s, it increased the DX and adapted the Dm accordingly. At the 41st s, the waypoint was been reached.

6.5. Fuzzy Controller Setup

The fuzzy controller is the main result of the optimization process. As has been stated above, the ruleset defined by an expert has made several optimizations unsuccessful. When it was hypothesized that the ruleset is imperfect (or limiting) and the extension of parameters defining the full set of membership functions, the solution was found. Below, Figures 17–19 show the fuzzy controller setup. There are several important aspects. One of the aforementioned aspects is that the relation between the maximums of membership functions has not been fulfilled. As a reminder, for angle input, the relations should be:

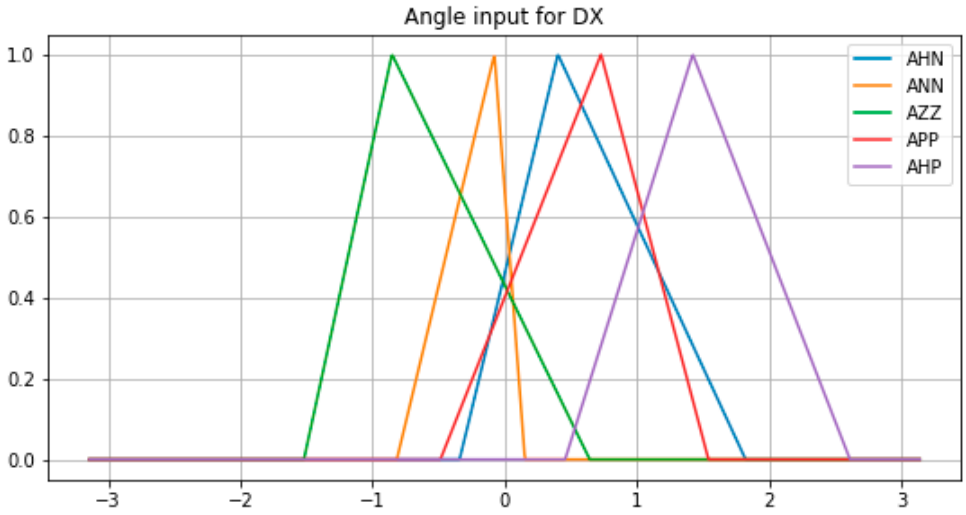

Figure 17. Membership functions resulted by optimization process for angle input fuzzy variable.

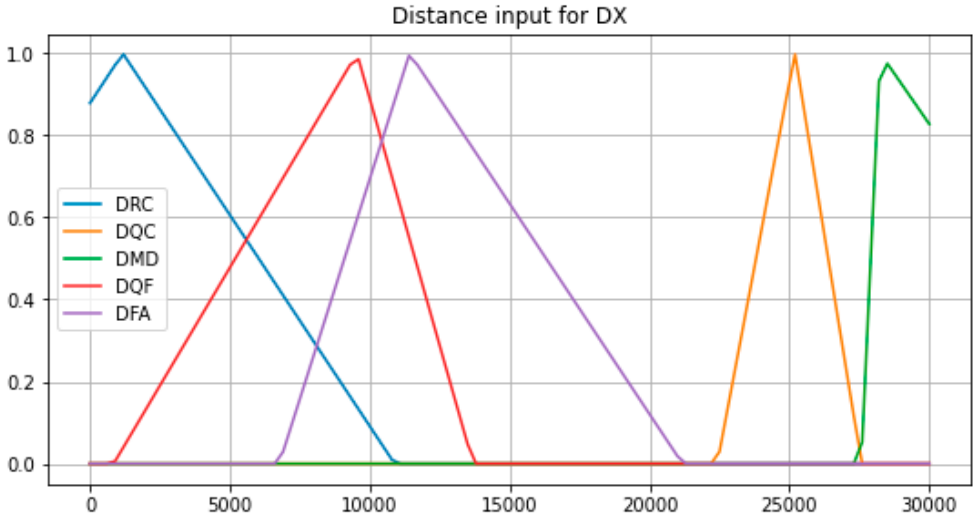

Figure 18. Membership functions resulted by optimization process for distance input fuzzy variable.

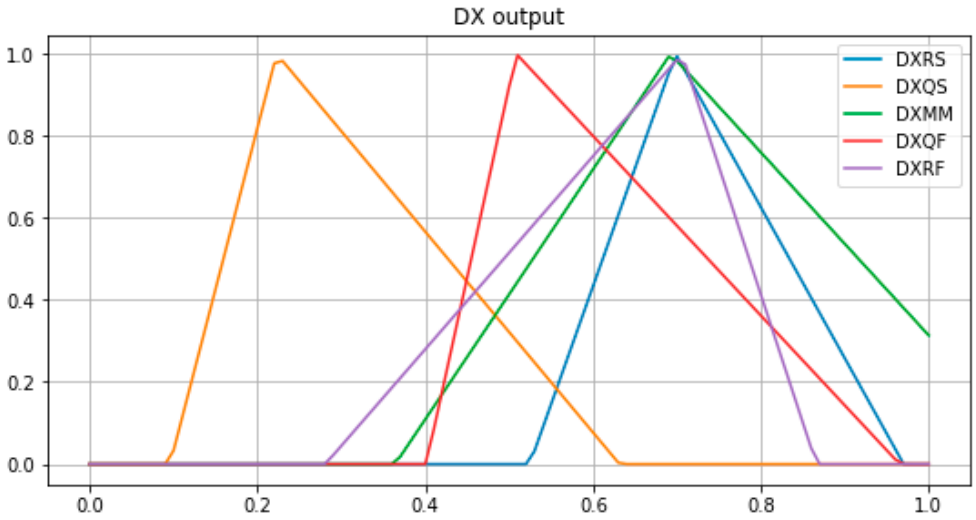

Figure 19. Membership functions resulted by optimization process for DX output fuzzy variable.

AHN < ANN < AZZ < APP < AHP,
but Figure 16 shows that
AZZ < ANN < AHN < APP < AHP.
Moreover, Distance should have
DRC < DQC < DMD < DQF < DFA,
but (see Figure 17)
DRC < DQF < DFA < DQC < DMD.

In Figure 17, it must be noted that there is missing control for distances in the interval from 21,000 m to 22,000 m.

7. Conclusions

In the previous chapters, a model of the fuzzy-based virtual pilot for wargaming and its development has been introduced. With the help of a genetic algorithm, the optimization of fuzzy logic controllers has been made. The sequence of experiments that have been conducted reveal that setting up the fuzzy rules for a complex system is a difficult problem that can be underestimated, and if the expert's proposal is not tested appropriately, it can mislead the research. As has been stated above, a definition able to check the inaccuracy of the ruleset can be used.

The proposed system was developed as a special subsystem for experimentation with the scenario of a battle aircraft against air defence systems where aircraft (pilots) should follow a mission. The models available in the literature have been determined too complicated for such a task. Unfortunately, modelling the aircraft as a point with constant velocity from one waypoint to the next miss some important aspects such as elevations of the plane, and thus it is not possible to model the detection of an approaching airplane by a radar, appropriately. Therefore, the main advantage of the proposed system is the perfect level of model details.

This article describes the first phase of implementing a synthetic pilot using fuzzy controllers. At this stage, the authors have limited themselves to simulating the control of the throttle lever and the elevator. The conclusions described above will be used to refine the proposed model and extend it to control other controls of the simulated aircraft.

There is a plan for the next iteration. The fuzzy rule setup will be revised with the help of an expert. In addition, a more flexible setup for the identification of an expert's mistake can be chosen. Such a setup can be based on increased membership functions employed in the output definition. If an extended setup version is achieved, 25 membership functions can be used. However, this case has a very large count of variables, and thus it will be quite hard to optimize it. Moreover, 25 membership functions for an output variable can help to point out that the use of the Sugeno fuzzy system could be more appropriate. It is also possible to perform a minimization process on the set of 25 membership functions, such as identifying clusters of membership functions and their replacement with one membership function.

Author Contributions: Conceptualization, A.S. and P.F.; methodology, A.S.; software, A.S.; validation, A.S. and P.F.; writing—original draft preparation, A.S. and P.F.; writing—review and editing, A.S. and P.F. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

References

1. Lone, M.; Cooke, A. Review of pilot models used in aircraft flight dynamics. *Aerosp. Sci. Technol.* **2014**, *34*, 55–74. [CrossRef]
2. Laskar, A.R.; Alam, S.; Islam, T.; Garg, A. Modeling and Simulation of Longitudinal Autopilot for General Aviation Aircraft. In Proceedings of the 5th International Conference on Informatics, Electronics and Vision (ICIEV), Dhaka, Bangladesh, 13–14 May 2016; IEEE: Piscataway, NJ, USA, 2016; pp. 490–495. [CrossRef]
3. Eichel-Streiber, J.; Weber, C.; Altenburg, J. Design and Implementation of an Autopilot for an UAV. In Proceedings of the 17th International Conference on Computer Aided Systems Theory, Las Palmas de Gran Canaria, Spain, 17–22 February 2019; Revised Selected Papers, Part I, pp. 3–10. [CrossRef]
4. Florencio Henriques, L.C.; Protasio De Souza, C. Development of an Embedded Longitudinal Flight Control Based on X-Plane Flight Simulator. *IEEE Lat. Am. Trans.* **2021**, *19*, 1684–1691. [CrossRef]
5. Štefek, A.; Časar, J.; Starý, V.; Gacho, L. Air Defence Command and Control System Modelling and Simulation for War-gaming. In Proceedings of the 8th International Conference on Military Technologies, Brno, Czech Republic, 8–11 June 2021; IEEE: Piscataway, NJ, USA, 2021.
6. Štefek, A.; Časar, J.; Starý, V. Flight route generator for simulation-supported wargaming. In Proceedings of the 2020 19th International Conference on Mechatronics–Mechatronika (ME), Praha, Czech Republic, 2–4 December 2020; IEEE: Piscataway, NJ, USA, 2020; p. 9286646.
7. Hamtil, I.; Šebela, M.; Štefek, A. Radar information creation with use of a simulation environment. *IET Radar Sonar Navig.* **2013**, *7*, 333–341. [CrossRef]
8. Sedova, N.; Sedov, V.; Bazhenov, R.; Bogatenkov, S. Neural network classifier for automatic course-keeping based on fuzzy logic. *J. Intell. Fuzzy Syst.* **2021**, *40*, 4683–4694. [CrossRef]
9. Zhu, L.; Li, T. Observer-Based Adaptive Fuzzy Control for Intelligent Ship Autopilot with State Constraint. In Proceedings of the 11th International Conference on Information Science and Technology, Chengdu, China, 21–23 May 2021; IEEE: Piscataway, NJ, USA, 2021; pp. 106–113. [CrossRef]
10. Štefek, A.; Pham, V.T.; Křivánek, V.; Pham, K.L. Optimization of Fuzzy Logic Controller Used for a Differential Drive Wheeled Mobile Robot. *Appl. Sci.* **2021**, *11*, 6023. [CrossRef]
11. Roman, R.C.; Precup, R.E.; Petriu, E.M. Hybrid data-driven fuzzy active disturbance rejection control for tower crane systems. *Eur. J. Control.* **2021**, *58*, 373–387. [CrossRef]
12. Frantis, P.; Cuzzolin, A. Real-time flight model for embedded simulator. *Adv. Mil. Technol.* **2014**, *9*, 59–68.
13. Van Rossum, G.; Drake, F.L. *Python Reference Manual*; Centrum voor Wiskunde en Informatica: Amsterdam, The Netherlands, 1995.
14. Virtanen, P.; Gommers, R.; Oliphant, T.E.; Haberland, M.; Reddy, T.; Cournapeau, D.; Burovski, E.; Peterson, P.; Weckesser, W.; Bright, J.; et al. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nat. Methods* **2020**, *17*, 261–272. [CrossRef] [PubMed]
15. Mamdani, E.H. Advances in the linguistic synthesis of fuzzy controllers. *Int. J. Man-Mach. Stud.* **1976**, *8*, 669–678. [CrossRef]
16. Takagi, T.; Sugeno, M. Fuzzy Identification of Systems and Its Applications to Modeling and Control. *IEEE Trans. Syst. Man Cybern.* **1985**, *15*, 116–132. [CrossRef]
17. Yang, X.S. *Nature-Inspired Optimization Algorithms*; Elsevier Inc.: Amsterdam, The Netherlands, 2014. [CrossRef]
18. Goldberg, D.E.; Korb, B.; Deb, K. Messy Genetic Algorithms: Motivation Analysis, and First Results. *Complex Syst.* **1989**, *5*, 493–530.
19. Blank, J.; Deb, K. Pymoo: Multi-Objective Optimization in Python. *IEEE Access* **2020**, *8*, 89497–89509. [CrossRef]
20. Dask Development Team. Dask: Library for Dynamic Task Scheduling. Available online: https://dask.org (accessed on 16 August 2021).