Design of Fuzzy Logic Controller for Auto Landing Applications

***K. David Solomon Raj, **Goutham Tattikota**

*Dept of Avionics, Jawaharlal Nehru Technological University – Kakinada
**Dept of Avionics, Jawaharlal Nehru Technological University - Kakinada

Abstract- It is a well-known fact about the degree of difficulty associated during the landing phase of an Aircraft. An efficient and a reliable auto-landing controller are therefore specially needed for auto landing purpose. A good number of parameters have to be controlled during the landing phase like the approach velocity, Heading, Heading offset, vertical velocity, flare, altitude, alignment of the aircraft etc., which makes the application of conventional controllers expensive for the purpose. This is precisely where fuzzy logic can be used. Using fuzzy sets and fuzzy set operation, it is possible to design a fuzzy reasoning system, which can act as a controller. Potentially, applications of such knowledge-based approach to controller design can result in efficiency, time and cost savings. The aircraft used in this project is a Personnel launching system (PLS) model given in the Aerosim adds in Matlab. The model of the PLS itself is unstable and before any simulation is made with it, there is a need to stabilize it. The stabilization of the Aerosim add model in MATLAB is performed by using classical techniques and the errors has been corrected and obtained smooth landing using fuzzy controller techniques.

I. INTRODUCTION

The scope of this paper is to design a fuzzy logic controller for landing of an air vehicle with various rule sets using program languages and to perform a design simulation that is done by MATLAB programming. The idea is to design and display the simulation of fuzzy logic controller for landing system control and the result of this will be display by using Rule viewer and Surface viewer which are parts of the graphical user interface (GUI) in Fuzzy Logic Toolbox in MATLAB program. This paper is designed to make use of the advantages of the Fuzzy Logic Toolbox and integrate it with SIMULINK which is also in MATLAB programmed.

This controller, based on fuzzy logic has been designed for a flight vehicle where it tracks the predetermined path trajectory for safe landing. This fuzzy logic tool box has the ability to take fuzzy systems into Simulink directly and test them in a simulation environment. This will display the animation of the landing phase for all the given parameters that controlled based on the rules of fuzzy sets. This system will be also tested with the Personal launching system by using various methods and different membership functions. The main purpose is to find the best way to get the result as close as the requirement for stability of the level control for the landing system.

Introduction to PLS: The Personnel Launch System (PLS), also known as HL-20, is a lifting body re entry vehicle designed to complement the Space Shuttle orbiter. It was developed originally as and a limited cargo. Low-cost solution for getting to and from low Earth orbit. It can carry up to 10 people a limited cargo.

The HL-20 lifting body can be placed in orbit either by launching it vertically with booster rockets or by transporting it in the payload bay of the Space Shuttle orbiter. Using a small onboard propulsion system the HL-20 lifting body deorbits. Its reentry profile is nose first, horizontal, and unpowered. Although the HL-20 program is not active currently, NASA projects used the aerodynamic data from HL-20 tests.

II. FUZZY LOGIC CONTROLLER

A. Basic idea

The basic idea of a fuzzy logic controller (FLC) is to imitate the control actions of a human operator, which can generally be represented as a collection of if-then rules. As an example, consider the rule:

If error1 is negative medium and error2 is zero, **then** control action is positive small.

The first part of the rule specifies the conditions under which the rule holds, called antecedent. The second part, called the consequent, prescribes the corresponding control action. Both parts contain vague, linguistic terms like small, medium, low, etc., that reflect the operators knowledge of the process. In the fuzzy control, the linguistic terms are represented by fuzzy sets, while **AND**, **OR** operations are used for combing the linguistic terms.

B. Fuzzy sets

A fuzzy set is an ordered set which associates each value of a variable to its grade of membership in the set. The grades of membership are represented by the membership function. The position and shape

(typically triangular or trapezoidal shaped) of the membership function depend on the particular application.

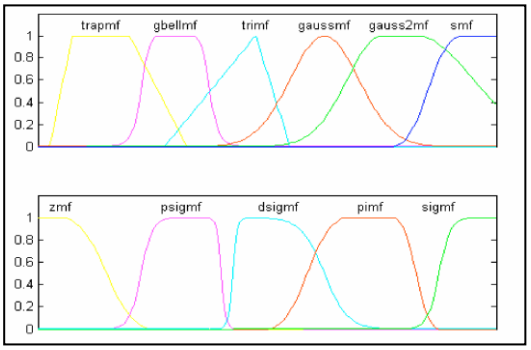

Fig 2.1: Various types of membership functions

C. *Fuzzy set operations*

Fuzzy set operations are performed by logical connectives such as **AND** (conjunction), **OR** (disjunction), or **NOT** (compliment). The most commonly used conjunction operators are the minimum and the product operators. Usually the maximum operator is used for the disjunction.

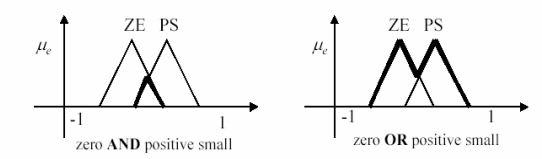

Fig 2.2: Conjunction and disjunction of 2 fuzzy sets by minimum and maximum operator

D. *Fuzzy logic control*

Using fuzzy sets and fuzzy set operation, it is possible to design a fuzzy reasoning system, which can act as a controller. The control strategy is stored in the form of it-then rules in the rule base. They represent as an approximate static mapping from inputs (e.g. errors).

Typically, the computational mechanism of a linguistic controller or a Mamdani type of FLC proceeds in five steps:

- Fuzzification**:** the membership degrees of the antecedent variables are computes (e.g. small (e), µmedium (Δe), µbig (Δe)).
- Degree of fulfilment**:** The degree of fulfillment for the antecedent of each rule is computed using fuzzy logic operators. The degree of fulfilment 'ωi' determines to which degree the 'ith' rule is valid.
- Implication**:** The degree of fulfilment is used to modify the consequent of the corresponding rule accordingly. This operation represents the **if-then** implication defines as a conjunction operator (e.g. product).
- Aggregation**:** The scale consequents of all rules are combined into a single fuzzy set. The aggregation operator depends on the implication function used; for the conjunctions, it is a disjunction operator (e.g. max).
- Defuzzification: The resulting fuzzy set is defuzzified to yield a crisp value. There exist a number of defuzzification methods, such as the centre of area method and others shown in the figure.

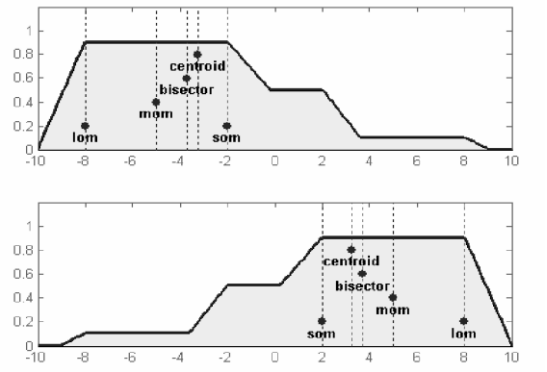

Fig2.3: Examples of Different Defuzzification Methods

For Mamdani-style inference, we can choose centroid, bisector, MOM (middle of maximum), SOM (smallest of maximum), LOM (largest of maximum),for a custom operation. For Sugeno-style inference, can choose either wtaver (weighted average) or wtsum (weighted sum).

III. FUZZY LOGIC CONCEPT IN LANDING APPLICATION

To construct and simulation of the final descent and landing approach of an aircraft, the desired profile is shown in below Figure.

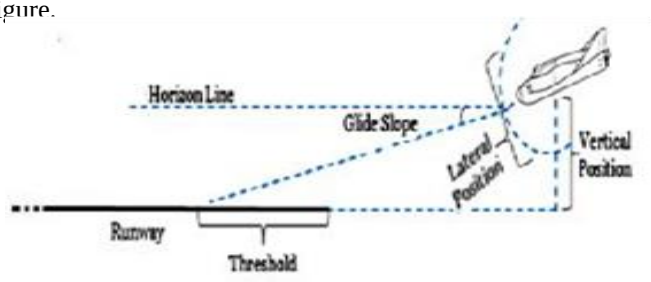

Fig3.1: Desired final approach path

The desired downward velocity is proportional to the square of the air vehicle height. So, at higher altitudes, a large downward velocity is desired. As the height diminishes, the required downward velocity gets smaller and smaller. In the limit, as the height becomes vanishingly small, the downward velocity also becomes zero. This is the way, the aircraft will descend from altitude promptly but will touchdown very gently to avoid damage.

The two state variables for this simulation will be the height above ground, h, and the vertical velocity of the aircraft, v. The control output will be a force that, when applied to the aircraft, will alter its height (h) and velocity (v).The differential control

equations are loosely derived as follows. Mass m moving with velocity v has momentum p (p=mv). If there is no external force was applied, then the mass will continue in the same direction at the same velocity, v. If over a time interval Δt,a force f is applied then there is a change in velocity of Δv=fΔt/m will result. If we let Δt=1.0 (sec) and m=1.0 (lb-sec2/ft), we obtain Δv=f (lb), or the change in velocity is proportional to the applied force.

In difference notation we get

$$v_{i+1} = v_i + f_i$$
$$h_{i+1} = h_i + v_i(1)$$

Where v_{i+1} is the new velocity, v_i is the old velocity, h_{i+1} is the new height, and h_i is the old height. These two "control equations" define the new value of the state variables v and h in response to control input and the previous state variables

	Output force (lbs)												
	-30	-25	-20	-15	-10	-5	0	5	10	15	20	25	30
(UL)	0	0	0	0	0	0	0	0	0	0.5	1	1	1
(US)	0	0	0	0	0	0	0	0.5	1	0.5	0	0	0
(Z)	0	0	0	0	0	0.5	1	0.5	0	0	0	0	0
(DS)	0	0	0	0.5	1	0.5	0	0	0	0	0	0	0
(DL)	1	1	1	0.5	0	0	0	0	0	0	0	0	0

IV. FUZZY CONTROLLER DESIGN PROCEDURE

Step 1: Define all the membership functions for state variables as shown in tables.

Step 2: As shown in tables, define a membership function for the control output.

Step 3: Define the rules and summarize them in an FAM table. The values in the FAM table are the control outputs.

Step 4: Define the initial conditions, and then conduct a simulation for four cycles. Since the task at hand is to control the aircraft's vertical descent during approach and landing, we will start with the aircraft at an altitude of 1000 feet, with a downward velocity of -20 ft/s. We will use the following equations to update the state variables for each cycle:.

	Height (ft)										
	0	100	200	300	400	500	600	700	800	900	1000
(L)	0	0	0	0	0	0	0.2	0.4	0.6	0.8	1
(M)	0	0	0	0	0.2	0.4	0.6	0.8	1	0.8	0.6
(s)	0.4	0.6	0.8	1	0.8	0.6	0.4	0.2	0	0	0
(NZ)	1	0.8	0.6	0.4	0.2	0	0	0	0	0	0

$$v_{i+1} = v_{i+1} + f_i$$
$$h_{i+1} = h_i + v_i$$

TABLES AND FUNCTION OUTPUTS:

TABLE 4.1: Membership values for height

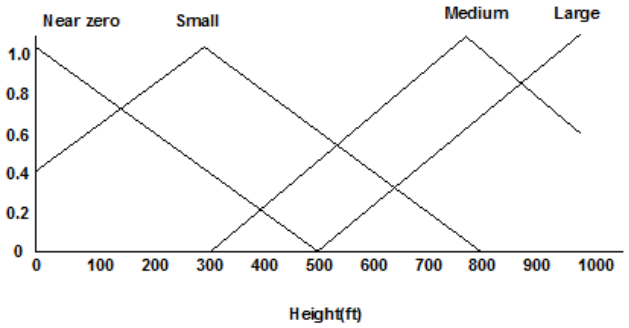

Fig 4.1: Membership Functions for Height

TABLE 4.2: Membership values for velocity

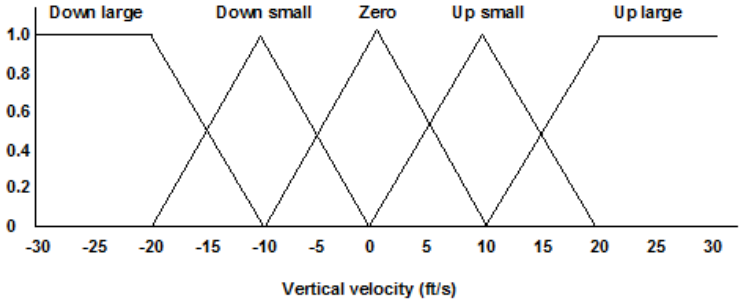

Fig 4.2: Membership Functions for Velocity

	Output force (lbs)												
	-30	-25	-20	-15	-10	-5	0	5	10	15	20	25	30
Up Large(UL)	0	0	0	0	0	0	0	0	0	0.5	1	1	1
Up small(US)	0	0	0	0	0	0	0	0.5	1	0.5	0	0	0
Zero(Z)	0	0	0	0	0	0.5	1	0.5	0	0	0	0	0
Down small(DS)	0	0	0	0.5	1	0.5	0	0	0	0	0	0	0
Down large (DL)	1	1	1	0.5	0	0	0	0	0	0	0	0	0

TABLE 4.3: Membership values for control force

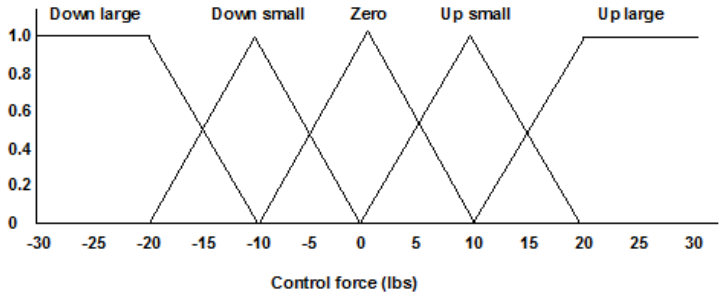

Fig 4.3: Membership Functions for Control Force

Height	Velocity				
	DL	DS	Zero	US	UL
L	Z	DS	DL	DL	DL
M	US	Z	DS	DL	DL
S	UL	US	Zero	DS	DL
NZ	UL	UL	Zero	DS	DS

TABLE 4.4: FAM table

SUMMARY AND SIMULATION RESULT

	Cycle 0	Cycle 1	Cycle 2	Cycle 3	Cycle 4
Height,ft	1000.0	980.0	965.8	951.1	936.0
Velocity,ft/s	-20.0	-14.2	-14.7	-15.1	-14.8
Control force	5.8	0.5	-0.4	0.3	

TABLE 4.5: Summary of four-cycle simulation results

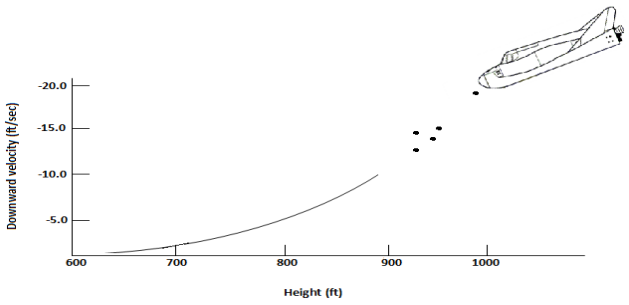

Fig 4.4: The profile of downward velocity, v, vs. height, h, using fuzzy logic control

V. FUZZY LOGIC CONTOL DESIGN USING MATLAB

Fuzzy inferences systems can be create and edit with Fuzzy Logic Toolbox software. We can create these systems using graphical tools or command-line functions, or we can generate them automatically using either clustering or adaptive neuro-fuzzy techniques. If we have access to Simulink software, we can easily test our fuzzy system in a block diagram simulation environment.

The toolbox also let us runs our own stand-alone C programs. This is made possible by a stand-alone Fuzzy Inference Engine that reads the fuzzy systems saved from a MATLAB session of the program. Then we can customize the stand-alone engine to build fuzzy inference into our own code. ANSI compliant code is provided. Because of the integrated nature of the MATLAB environment, we can create our own tools to customize the toolbox or harness it with another toolbox, like a Control System Toolbox, Optimization Toolbox software, Neural Network Toolbox.

A. FIS EDITOR

The FIS Editor GUI tool allows us to edit the fuzzy inference system highest level features, such as the number of output and input variables, the defuzzification method used, and so on. The FIS Editor is the high-level display for any fuzzy logic inference system. It allows calling the various other editors to operate on the FIS. This allows convenient access to all other editors with an emphasis on maximum flexibility for interaction with the fuzzy system.

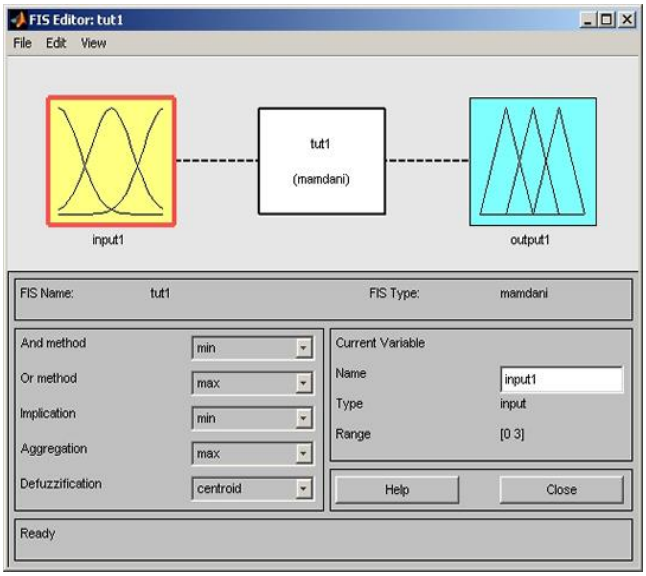

Fig5.1: FIS Editor

1. *The diagram*

The diagram displayed at the top of the window shows a central fuzzy rule processor, inputs and outputs, . Click one of the variable boxes to make the selected box the current variable. We should see the box highlighted in red. Give a Double-click on one of the variables to bring up the Membership Function Editor. Then Double-click the fuzzy rule processor to bring up the Rule Editor. A variable exists but is not mentioned in the rule base, in that case it is connected to the rule processor block with a dashed rather than a solid line.

2. *Menu items*

The FIS Editor displays a menu bar that allows us to open related GUI tools, open and save systems, and so on.

- Under **File** select

New FIS > **Mamdani** to open a new Mamdani-style system with no variables and no rules called Untitled.

New FIS > **Sugeno** to open a new Sugeno-style system with no variables and no rules called Untitled.

Import > **From workspace** to load a system from a specified FIS structure variable in the workspace.

Import > **From file** to load a system from a specified .fis file.

Export > **To workspace.** to save the system to a FIS structure variable in the workspace.

Export > **To file** to save the current system to a .fis file.

Print to print what is displayed in the GUI.

Close to close the GUI.

- Under **Edit** select

Undo to undo the most recent change.

Add variable > **Input** to add another input to the current system.

Add variable > **Output** to add another output to the current system.

Remove Selected Variable to delete a selected variable.

Membership functions to invoke the Membership Function Editor.

Rules to invoke the Rule Editor.

- Under **View** select

Rules to invoke the Rule Viewer.

Surface to invoke the Surface Viewer.

3. *Interface method pop-up menus*

And method: Choose min, prod, or Custom, for a custom operation.

Or method: Choose max, probor (probabilistic or), or Custom, for a custom operation.

Implication: Choose min, prod, or Custom, for a custom operation. This selection is not available for Sugeno-style fuzzy inference.

Aggregation: Choose max, sum, probor, or Custom, for a custom operation. This selection is not available for Sugeno-style fuzzy inference.

Defuzzification: For Mamdani-style inference, choose centroid, bisector, MOM (middle of maximum), SOM(smallest of maximum), LOM (largest of maximum), or Custom, for a custom operation. For Sugeno-style inference, choose between wtaver (weighted average) or wtsum (weighted sum).

B. RULE EDITOR

On the Rule Editor, there is a menu bar that allows to open related GUI tools, open and save systems, and so on. The **File** menu for the Rule Editor is the same as the one found on the FIS Editor.

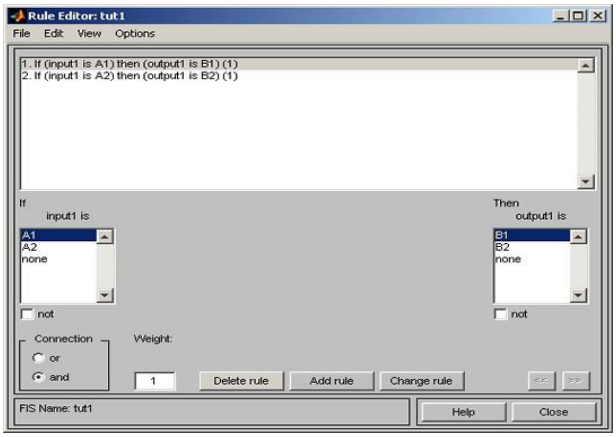

Fig 5.2: Rule Editor

C. RULE VIEWER

On the Rule Viewer, there is a menu bar that allows to open related GUI tools, open and save systems, and so on. The **File** menu for the Rule Viewer is the same as the one found on the FIS Editor. Refer to fuzzy for more information.

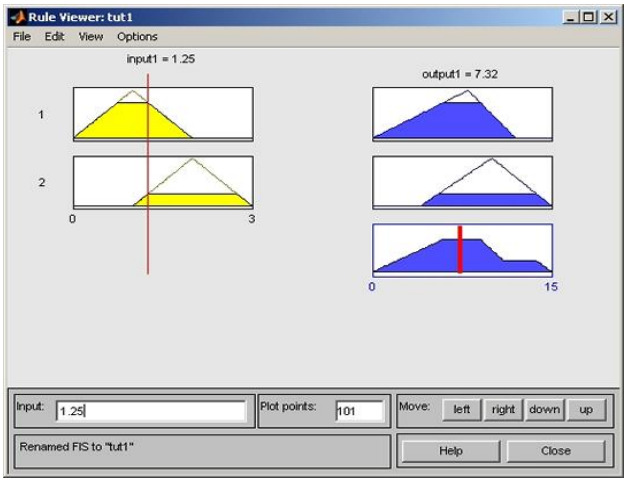

Fig5.3 : Rule Viewer

- Use the **Edit** menu items:

Undo to undo the most recent action

FIS properties to invoke the FIS Editor

Membership functions to invoke the Membership Function Editor

Rules to invoke the Rule Editor

- Use the **View** menu item:

Surface to invoke the Surface Viewer

- Use the **Options** menu item:

Format to set the format in which the rule appears: **Verbose, Symbolic,** or **Indexed**

D. SURFACE VIEWER

Upon opening the Surface Viewer, we can see a three-dimensional curve. This curve represents a two-input one-output case, from this we can see the entire mapping in one plot. If we move beyond three dimensions overall, then we start to encounter trouble displaying the results.

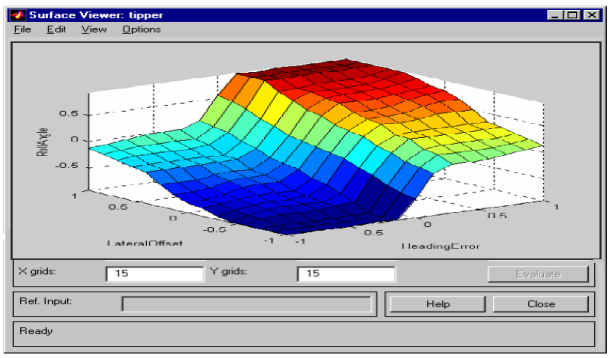

Fig 5.4: Surface Viewer

Accordingly, the Surface Viewer is equipped with drop-down menus X (input): Y (input): and Z (output): that let us select any two inputs and any one output to plot. Below these menus are two input fields X grids: and Y grids: that let us specify how many x-axis and y-axis grid lines you want to include in it. This capability allows you to keep the calculation time reasonable for complex problems.

If we want to create a smoother plot, we can use the Plot point's field to specify the number of points on which the membership functions are evaluated in the output or input range. By default, the value of this field is 101. Clicking Evaluate initiates the calculation, then the plot is generated after the calculation is complete. In order to change the x-axis or y-axis grid after the view, change the appropriate input field, press Enter. The surface plot is updated to reflect the new grid settings.

The Surface Viewer has a special capability that is very helpful in cases with two (or more) inputs and one output: you can grab the axes, by using the mouse and reposition them to get a different three-dimensional view on the data we have. The Reference Input field is used in situations when there are more inputs required by the system than the surface is mapping. We can edit this field to explicitly set inputs not specified in the surface plot.

Suppose we have a four-input one-output system and would like to see the output surface.Surface Viewer can generate a three-dimensional output surface where any two of the inputs vary, but the two of the inputs must be held constant because computer monitors cannot display a five-dimensional shape.If such a case occurs, the input is a four-dimensional vector with NaNs holding the place of the varying inputs while numerical values indicates those values that remain fixed. Here NaN is the IEEE® symbol for Not a Number.

The menu items allow us to open, close, save and edit a fuzzy system using the five basic GUI tools. We can access information about the Surface Viewer by clicking Help and close the GUI using Close.

VI. COORDINATE SYSTEM FOR MODELLING PLS

The model for airframe incorporates several key assumptions and limitations:

- Here the airframe was assumed to be rigid and have constant mass, centre of gravity, and inertia, since the model represents only the unpowered re entry portion of a mission.
- HL-20 is taken as a laterally symmetric vehicle.
- Compressibility (Mach) effects are to be assumed negligible.
- Control effectiveness is assumed to vary nonlinearly with angle of attack and linearly with angle of deflection. and Control effectiveness is not dependent on sideslip angle.

Coordinate system allows us to keep track of an aircraft or spacecraft's position and orientation in space. Aerospace Block set coordinate systems are based on these underlying concepts from astronomy, geodesy and physics.

Modelling aircraft and spacecraft is simplest if we use a coordinate system fixed in the body itself. In case of aircraft, the forward direction is modified by the presence of wind, the craft's motion through the air is not the same as its motion relative to the ground.

A. BODY COORDDINATES

The non inertial body coordinate system is fixed in both origin and orientation to the moving craft. This craft is assumed to be rigid.

The orientation of the flight body coordinate axes is fixed in the shape of body.

1. The x-axis points the aircraft through the nose.
2. The y-axis points the right of the x-axis (facing in the pilot's direction of view), perpendicular to the x-axis.
3. The z-axis points down through the bottom the aircraft, perpendicular to the xy plane and satisfying the RH rule.

TRANSLATIONAL DEGREES OF FREEDOM

These are defined by moving along these axes by distances *x*, *y*, and *z* from the origin.

ROTATIONAL DEGREES OF FREEDOM

Rotations are defined by the Euler angles (given *P*, *Q*, *R* or Φ, Θ, Ψ) They are:

P or Φ Roll about X-axis
Q or Θ Roll about Y-axis
R or Ψ Roll about Z-axis

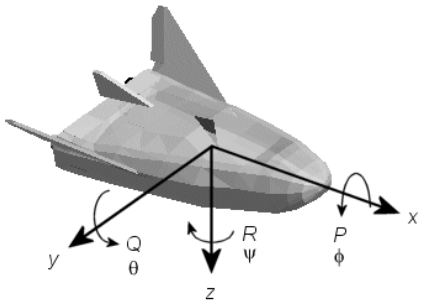

Fig6.1: Body Co-ordinate System

B. WIND COORDINATES

The non inertial wind coordinate system has its origin fixed in the rigid air vehicle. The coordinate system orientation is defined relative to the craft's velocity V.

The different orientation of the wind coordinate axes is fixed by the velocity V.

- X-axis points in the direction of V.
- Y-axis is points to the right of the x-axis (facing in the direction of V), perpendicular to the x-axis.
- Z-axis is point's perpendicular to the xy plane in whatever way needed to satisfy the RH rule with respect to the x- and y axes.

TRANSLATIONAL DEGREES OF FREEDOM

The Translations are defined by moving along these axes by distances x, y, and z from the origin.

ROTATIONAL DEGREES OF FREEDOM

Rotations are defined by the Euler angles (given Φ, γ, χ). They are:

Φ Bank angle about the x-axis

Γ Flight path about y-axis

X Heading angle about z-axis.

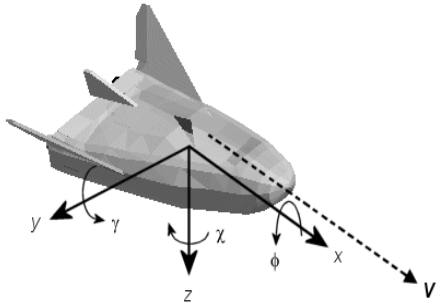

Fig6.2: Wind Co-ordinate System

C. COORDINATE SYSTEM FOR NAVIGATION

Modelling aerospace trajectories requires positioning and orienting the aircraft or spacecraft with respect to the rotating Earth. The Navigation coordinates are defined with respect to the centre and surface of the Earth.

GEOCENTRIC AND GEODETIC LATITUDES

The λ, geocentric latitude on the Earth's surface is defined by the angle subtended by the radius vector from the Earth's centre to the surface point with the equatorial plane.

The μ, geodetic latitude on the Earth's surface is defined by the angle subtended by the surface normal vector n and the equatorial plane.

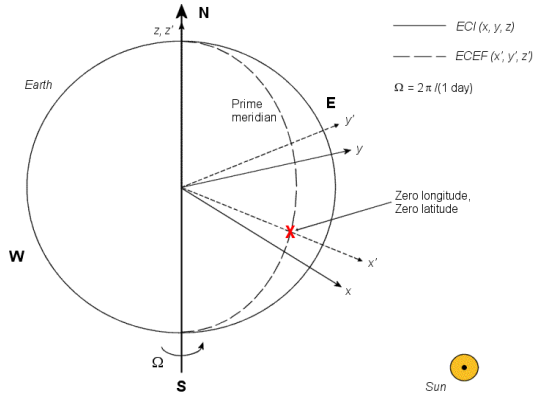

Fig6.3: Earth-centered coordinates

D. ECEF COORDINATES

The Earth-center, Earth-fixed (ECEF) system is a non inertial system that rotates with the Earth. The origin is fixed at the center of the Earth.

- The z′-axis points to northward along the Earth's rotation axis.
- The x′-axis points to outward along the intersection of the Earth's equatorial plane and prime meridian.

The y′-axis points into the eastward quadrant that is perpendicular to the x-z plane so as to satisfy the RH rule

E. COORDINATE SYSTEM FOR DISPLAY

Different display tools are available for use with the Aerospace Block set product, which has a specific coordinate system for rendering motion.

F. MATLAB GRAPHICS COORDINATES

MATLAB Graphics uses the default coordinate axis orientation:

- The *x*-axis points out of the screen.
- The *y*-axis points to the right.
- The *z*-axis points up.

G. FLIGHTGEAR COORDINATES

Flight Gear is an open-source, third-party flight simulator with an interface supported by the block set. The Flight Gear coordinates form a special body-fixed system, which rotated from the standard body coordinate system about the *y*-axis by -180 degrees:

- The *x*-axis is +ve toward the back of the vehicle.
- The *y*-axis is +ve toward the right of the vehicle.
- The *z*-axis is +ve upward, e.g., wheels typically have the lowest *z* values.

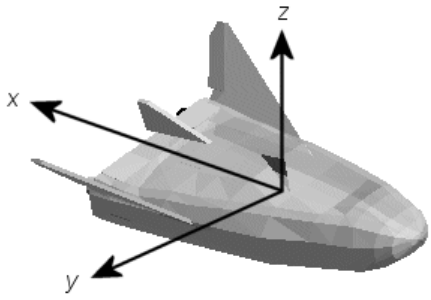

Fig6.4: Flight Gear Co-ordinates

VII. INTEGRATING FUZZY CONTROLLER WITH PLS

A. BASIC IDEA

The Fuzzy Controller was integrated to PLS model for performance and robustness evaluation. Evaluation was made for landing phase. An Auto-land button was provided, initialization of which activates the fuzzy controller to take over and make a safe landing.

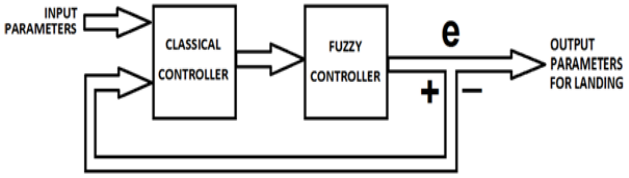

Fig7.1:Basic idea

In this paper, a controller based on fuzzy logic methodology has been designed for a flight vehicle that enables it to track pre-determined path trajectory for the safe landing. The controller has been designed with the available conventional autopilot. The fuzzy controller will be compare the error of the classical controller during landing with the pre setup program and it will correct the errors for safe landing.

B. INTEGRATION TECHNIQUE

Fuzzy Logic Toolbox software is a collection of functions built on the MATLAB technical computing environment itself. It provides tools to create and edit fuzzy inference systems within the framework of MATLAB. We can integrate our fuzzy system into simulations with Simulink software. We can even build stand-alone C programs that call on fuzzy systems we build with MATLAB tool box. This toolbox relies heavily on graphical user interface (GUI) tools to help us to accomplish work, although we can work entirely from the command line if we prefer.

This toolbox provides three categories of tools:

- Command line functions
- Graphical interactive tools
- Simulink blocks and examples

The first category of tools is made up of functions that we can call from the command line or from our own applications. Most of these functions are MATLAB M-files, and series of MATLAB statements that implement specialized fuzzy logic algorithms. Here we can view the MATLAB code for these functions using the statement.

We can change the way any toolbox function works by copying and renaming the M-file, it then modifying our copy. We can also extend the toolbox by adding our own M-files.

Secondly, the toolbox provides a number of interactive tools that let us access many of the functions through GUI.The GUI-based tools provide an environment for fuzzy inference system design, its analysis, and implementation.

The third category of tools is a set of blocks for use with Simulink. Those are specifically designed for high speed fuzzy logic inference in the Simulink environment.

What makes the toolbox so powerful is the fact that most of human reasoning and concept formation is linked to the use of rules of fuzzy. By providing a systematic framework for computing with fuzzy rules, that toolbox greatly amplifies the power of human reasoning. The amplification results from the use of MATLAB and graphical user interfaces, areas in which The Math Work has unparalleled expertise.

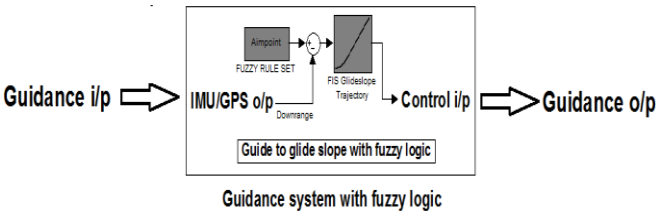

Fig7.2: Integration of Fuzzy logic guidance system

C. GUIDANCE SYSTEM WITH FUZZY LOGIC

The Fuzzy Logic Controller block implements a fuzzy inference system (FIS) in Simulink. The Fuzzy Logic Controller with Rule viewer block implements a fuzzy inference system (FIS) with the Rule Viewer in Simulink.

To build your own Simulink systems that use fuzzy logic, copy the Fuzzy Logic Controller block out of sltank (or any of the other Simulink demo systems available with the toolbox) and place it in your own block diagram. Here you can also find the Fuzzy Logic Controller blocks in the Fuzzy Logic Toolbox library. In this you can open the library by selecting Fuzzy Logic Toolbox in the Simulink Library Browser window, otherwise by typing Fuzblock at the MATLAB prompt.

The Fuzzy Logic Toolbox library contains the Fuzzy Logic Controller and Fuzzy Logic Controller with Rule Viewer blocks. Also it includes a Membership Functions sub library that contains Simulink blocks for the built-in membership functions.

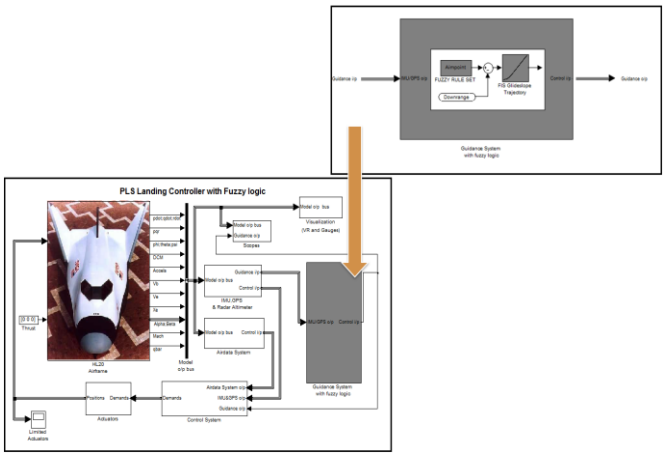

Fig7.3: PLS Landing Controller with Fuzzy Logic

VIII. SIMULATION

In addition to computer modelling of vehicle controllability during entry, flight simulator was set up at Langley to permit pilots to study the final landing phase of aircraft. Starting at an altitude of 4,600 meters, the simulation presented the pilot with a realistic view of the approach to a runway landing.

Simulation A duplication, working replica, or representation of a system or complex equipment in controlled environments for purpose of evaluation, analysis of problems,

Demonstration or training - at no risk to man or machine.

Flight Simulator Creates realistic cockpit environment, pilot feel and external world cues to enable the pilot to interact realistically with the flying controls.

IX. RESULT

From simulation we find that fuzzy controllers have given us better results than conventional controllers.

- We achieved good result for roll, Angle of attack and Angle of sight against demand by using fuzzy logic concept. In the scope named "demand versus achieved" shows the result.
- From them Guidance Performance, we had taken the parameters of Height, Flight path angle, Velocity and Vertical speed as inputs shown in the "Guidance Performance" scope.
- We got the smooth values of Attitudes Accelerations and Mach number with the fuzzy logic controller.

Addition to computer modeling of vehicle controllability starting at an altitude of 10,000 m, the simulation presented the realistic view of the approach to a runway landing.

X. CONCLUSION

A fuzzy logic based controller has been designed and tested on the HL20 PLS for the landing phase in the simulation using Aerospace block set of MATLAB. The results are adequate and quite promising.

Present evaluation was made for HL 20 PLS and the evaluation of the same for different aircraft can be taken up in the future.

This fuzzy logic concept can be useful for the developing Personnel Launching System designs and it can be a mile stone for the space applications.

REFERENCES

[1] Fuzzy-Logic Based Auto land Controller, Dr Poh Eng Kee and Dr Li Dong, Unmanned system Programme, Unmanned System centre, DSO National laboratories.
[2] Automatic Controls of Aircraft and Missiles, by John H. Blake lock.
[3] Jackson, E.B., and C.L. Cruz, "Preliminary Subsonic Aerodynamic model for Simulation Studies of the HL-20 Lifting Body", NASA TM4302 (August 1992).This document is included in the HL-20 Lifting Body.zip available from MATLAB central.
[4] Morring, F., Jr., "ISS Lifeboat Study Includes ELVs," aviation week and space technology (May 20, 2002).
[5] Fu Fuzzy logic with engineering applications by, Timothy J.Ross, Mc Graw Hill International editions.
[6] Control systems Engineering By., A. Nagoorkani.

AUTHORS

First Author – K David Solomon Raj, Lecturer, Institute of Science and Technology (IST), JNT University – Kakinada. Email id: k.rajusolomon@gmail.com

Second Author – Goutham Tattikota, P.G. Scholar, Institute of Science and Technology (IST), JNT University – Kakinada. Email id: eng.goutham@gmail.com

Correspondence Author – Goutham Tattikota, P.G. Scholar, Institute of Science and Technology (IST), JNT University – Kakinada. Email id: eng.goutham@gmail.com