Abstract – Water level control is highly important in industrial applications such as boilers in nuclear power plants. This paper analyzes the effectiveness of water level controller using fuzzy logic. The fuzzy controller is implemented in MATLAB and then simulated in SIMULINK to test the behaviour of the system. The response of the fuzzy controller is then compared with a conventional PID controller. The results are shown and the effectiveness of the controller is illustrated. In many industrial processes, control of liquid level is required. It was reported that about 25% of emergency shutdowns in the nuclear power plant are caused by poor control of the steam generator water level. Such shutdowns greatly decrease the plant availability and must be minimized. Water level control system is a very complex system, because of the nonlinearities and uncertainties of a system. Currently, constant gain PI controllers are used in nuclear organisations for boiler water level control at high power operations. However, at low power operations, PI controllers can not maintain water level properly. A need for performance improvement in existing water level regulators is therefore needed.

Keywords – Fuzzy logic, MATLAB, PID Controller, Sensor, SIMULINK, Water level controller.

I. INTRODUCTION

Combined water tank systems are used in many resident areas. Most of the time they are functioning well, but there are some condition that the system faces the problem of overflows because of the system cannot detect whether the water level has properly reached the desired level or the flow-in rate of the water is not proportional with the flow-out rate. It can cause the water tank system empties faster than it fills up. Consequently, the pressure of the water cannot support the distribution of the water to the resident area. Construction of new tank, valve or system could be the solution to completely eliminate the problems. However, such schemes are expensive and can also be extremely disruptive because the system networks may extend across wide geographical areas. Alternatively, controlling the flow between parts of the system which are under different loading can reduce the overflow spills. Actuated valve can be opened or closed to control the flow past a certain point. The implementation of the fuzzy logic controller in water tank level control system can be used to overcome this problem. By doing some modification of this paper, the problem can be solved in the simplest and cheapest possible controller for a given application. The basic idea is still the same and the modification will be made depending on the design expectation. One interesting feature of this water tank system is that the tank empties much more slowly than it fills up because of the specific value of the outflow diameter pipe. It can deal with this by setting of the valve membership function. Although this paper is only focusing on software simulation not really on the hardware, but this is the best step that should be considered before any implementation of the system be constructed. By testing the system in this simulation area, the expected output from the input can be set earlier based on the rules set. The use of MATLAB card is most required in order to integrate the software and hardware parts of the system and the cost of MATLAB card is quite high. This paper is concentrated on the software simulation part only.

Fuzzy logic is a form of many valued logic which deals with reasoning that is approximate rather than fixed and exact. Compared to traditional binary sets (where variables may take on true or false values), fuzzy logic variables may have a truth value that ranges in degree between 0 and 1. Fuzzy logic has been extended to handle the concept of partial truth, where the truth value may range between completely true and completely false. Fuzzy logic has been applied to many field, from control theory to artificial intelligence. Fuzzy logic has rapidly became one of the most successful of today’s technologies for developing sophisticated control systems. Fuzzy logic addresses such applications perfectly as it resembles human decision making with an ability to generate precise solutions from certain or approximate information. It fills an important gap in engineering design methods left vacant by purely mathematical approaches (e.g. linear control design), and purely logic based approaches (e.g. expert systems) in system design. While other approaches require accurate equations to model real world behaviours, fuzzy design can accommodate the ambiguities of real world human language and logic. It provides both an intuitive method for describing systems in human terms and automates the conversion of those system specifications into effective models.

Fuzzy logic offers several unique features that makes it a particularly good choice for many control problems. These are:

- It is inherently robust since it does not require precise, noise free inputs and can be programmed to fail safely if a feedback sensor quits or is destroyed. The output control is a smooth control function despite a wide range of input variations.
Since the fuzzy logic controller processes user defined rules governing the target control system, it can be modified and tweaked easily to improve or drastically alter system performance. New sensors can easily be incorporated into the system simply by generating appropriate governing rules.

Fuzzy logic is not limited to a few feedback inputs and one or two control outputs, nor it is necessary to measure or compute rate of change parameters in order for it to be implemented. Any sensor data that provides some indication of a system’s actions and reactions is sufficient. This allows the sensors to be inexpensive and imprecise thus keeping the overall system cost and complexity low.

Because of the rule based operation, any reasonable number of inputs can be processed (1-8 or more) and numerous outputs (1-4 or more) generated, although defining the rule base quickly becomes complex if too many inputs and outputs are chosen for a single implementation since rules defining their interrelations must also be defined. It would be better to break the control system into smaller chunks and use several smaller fuzzy logic controllers distributed on the system, each with more limited responsibilities.

MATLAB (matrix laboratory) is a multiparadigm numerical computing environment and fourth generation programming language. Developed by MathWorks, MATLAB allows matrix manipulations, plotting of functions and data, implementation of algorithms, creation of user interfaces, and interfacing with programs written in other languages, including C, C++, Java, Fortran and Python.

Although MATLAB is intended primarily for numerical computing, an optional tool box uses the MuPAD symbolic engine, allowing access to symbolic computing capabilities. An additional package, Simulink, adds graphical multi-domain simulation and Model-Based Design for dynamic and embedded systems.

Simulink, developed by MathWorks, is a data flow graphical programming language tool for modelling, simulating and analyzing multi domain dynamic systems. Its primary interface is a graphical block diagramming tool and a customizable set of block libraries. It offers tight integration with the rest of the MATLAB environment and can either drive MATLAB or be scripted from it. Simulink is widely used in control theory and digital signal processing for multi domain simulation and Model-Based Design.

A sensor is a transducer whose purpose is to sense (that is to detect) some characteristic of its environs. It detects events or changes in quantities and provides a corresponding output, generally as an electrical or optical signal.

Sensors are used in everyday objects such as touch-sensitive elevator buttons and lamps which dim or brighten by touching the base, besides innumerable applications of which most people are never aware. With advances in micro machinery and easy to use micro controller platforms, the uses of sensors have expanded beyond the more traditional fields of temperature, pressure or flow measurement.
A proportional integral derivative controller (PID Controller) is a control loop feedback mechanism widely used in industrial control systems. A PID controller calculates an error value as the difference between a measured process variable and a desired set point. The controller attempts to minimize the error by adjusting the process through use of a manipulated variable.

II. METHODOLOGY & RESULT

How is fuzzy logic used?

- Define the control objectives and criteria: What am I trying to control? What do I have to do to control the system? What kind of response do I need? What are the possible (probable) system failure modes?

- Determine the input and output relationships and choose a minimum number of variables for input to the fuzzy logic engine (typically error and rate of change of error).

- Using the rule based structure of fuzzy logic, break the control problem down into a series of IF X AND Y THEN Z rules that define the desired system output response for given system input conditions. The number and complexity of rules depends on the number of input parameters that are to be processed and the number fuzzy variables associated with each parameter. If possible, use at least one variable and its time derivative. Although it is possible to use a single, instantaneous error parameter without knowing its rate of change, this cripples the system’s ability to minimize overshoot for a step inputs.

- Create fuzzy logic membership functions that define the meaning (values) of Input/Output terms used in the rules.

- Create the necessary pre and post processing fuzzy logic routines if implementing in S/W, otherwise program the rules into the fuzzy logic H/W engine.

Test the system, evaluate the results, tune the rules and membership functions, and retest until satisfactory results are obtained.

Defining the structure: In FIS editor first of all the structure of the system is defined. The structure is shown below. There are 2 inputs and 1 output. The 2 inputs are: i) Tank inflow, ii) Tank inflow rate. The output is Tank level.

It can be said that the FIS editor displays the information about the fuzzy inference system.

Defining the membership function and membership degree for all the input and output variables: After defining the structure in FIS editor the GUI tool that comes into play in membership editor. We can choose any type of membership function among the various types like trapMF, triMF, GuassMF etc..
For the first input variable i.e. ‘Level’ in this paper the type of membership function used is triMF. Also we can use any number of membership degrees for the membership function of a variable.

The membership function is a graphical representation of the magnitude of participation of each input. It associates a weighting with each of the inputs that are processed, define functional overlap between inputs, and ultimately determines an output response. The rules use the input membership values as weighting factors to determine their influence on the fuzzy output sets the final output conclusion. Once the functions are inferred, scaled, and combined, they are defuzzified into a crisp output which drives the system. There are different membership functions associated with each input and output response.

There are 5 membership degrees have been used for the first input variable. They are:

- Very low
- Low
- Normal
- High
- Very high

For the second input variable i.e. ‘Rate’ in this paper the type of membership function used is triMF. There are 5 membership degrees have been used for the second input variable. They are:

- Very low
- Low
- Normal
- High
- Very high

Similarly here also the range of the membership degree can be varied according to the requirement.
For the output variable i.e. ‘Tank Level’ in this paper the type of membership function used is triMF. There are 5 membership degrees have been used for the first output variable. They are:

- Very low
- Low
- Normal
- High
- Very high

Similarly here also the range of the membership degree can be varied according to the requirement.

Rule Matrix: The fuzzy parameters of error (command ~ feedback) and error ~ dot (rate ~ of ~ change ~ of ~ error) are modified by the adjectives negative, zero and positive. To picture this, imagine the simplest practical implementation, a 3 by 3 matrix. The columns represent negative error, zero error and positive error inputs from left to right. The rows represent negative, zero, and positive error dot input from top to bottom. This planar construct is called a rule matrix. It has two input conditions, error and error dot. One output response conclusion (at the intersection of each row and column). In this case there are nine possible logical product (AND) output response conclusions.

Although not absolutely necessary, rule matrices usually have an odd number of rows and columns to accommodate a zero center row and column region. This may not be needed as long as the functions on either side of the center overlap somewhat and continuous dithering of the output is acceptable since the zero regions correspond to no change output responses, the lack of this region will cause the system to continually hunt for zero. It is also possible to have a different number of rows than columns. This occurs when numerous degrees of inputs are needed. The maximum number of possible rules is simply the product of the number of rows and columns, but definition of all of these rules may not be necessary since some input conditions may never occur in practical operation. The primary objective of this construct is to map out the universe of possible inputs while keeping the system sufficiently under control.
Possibilities of fuzzy logic in water level controller: As a more advanced look at the possibilities of fuzzy logic we will look at the control of a water tank. This water tank has a pipe flowing in and a pipe flowing out. The input flow rate is variable by a control valve. The output flow rate is dependent on the amount of water in the tank; more water in the tank results in a faster output flow rate.

The water will drain out of the tank faster when there is more water in the tank and slower when there is less water in the tank. The goal of the control system is to take a set value and change the input valve so that the inflow rate will compensate for the outflow. The tank can be represented in Simulink using the following block.
This block can then be used in developing a control system for the tank using fuzzy logic. The Simulink model that we will be using to control the tank can be seen in the following illustration. The fuzzy controller inputs are the amount of error in the tank water level and the rate of change for the tank water level. These two inputs will be considered when the fuzzy logic controller determines the proper input valve setting.

Fig.10: Block diagram of water level controller by using fuzzy logic

The inflow max is in place since a given pipe can only provide a limited amount of water. The parameters for the water tank are shown below:

Fig.11: Block diagram of water level controller by using fuzzy logic & PID controller together

The inflow max is in place since a given pipe can only provide a limited amount of water. The parameters for the water tank are shown below:

Fig.12: Functions block parameters
Now that the simulink model has been made the fuzzy control system needs to be designed. In MATLAB, we are opening the FIS Editor to develop a new fuzzy system. The default FIS has only one input, however the system we will be using has two inputs so we need to add another input. To do this click: Edit > Add Variable… > Input. This will add another input variable to the system. These two inputs can be named Tank Inflow and Tank Inflow Rate and the output can be named Tank Level. The system should now look like this:

![Fig.13: FIS editor displaying the overall structure](image)

The membership functions used for this system now need to be defined for each input and for the output. The functions for each can be seen in the following images. The tank inflow input will be in the range of 0 to 1 and the tank inflow rate input will be in the range of 0 to 0.1 while the tank level output will be in the range of 0 to 1.

![Fig.14: Membership editing of the first input variable](image)

There are five membership functions for very low, low, normal, high and very high. The very low function is a triangular membership function which has the following parameters: [0.00296 0.1002 0.2]. The low membership function is triangular with the following parameters: [0.2 0.3 0.4]. The normal membership function is triangular and it has parameters [0.4 0.5 0.6]. The high membership function is triangular with the following parameters: [0.6 0.7 0.8]. The very high membership function is triangular with the following parameters: [0.8 0.91].
The tank inflow rate input also has five membership functions: very low, low, normal, high and very high. The very low function is a triangular membership function which has the following parameters: [ 0.00296 0.1002 0.2 ]. The low membership function is triangular with the following parameters: [ 0.2 0.3 0.4 ]. The normal membership function is triangular and it has parameters: [ 0.4 0.5 0.6 ]. The high membership function is triangular with the following parameters: [ 0.6 0.7 0.8 ]. The very high membership function is triangular with the following parameters: [ 0.8 0.91 ].

There are five output membership functions for the Tank Level output on the system: very low, low, normal, high and very high. They are all triangular functions with the following parameters:

- Very Low: [ 0.00221 0.1 0.1999 ]
- Low: [ 0.2 0.3 0.4 ]
- Normal: [ 0.4 0.5 0.6 ]
- High: [ 0.6 0.7 0.8 ]
- Very High: [ 0.8 0.9 ]

The rules then that we will use in the rule editor are shown below:
Another way to visualize the rules is to look at the rules surface for the system. This can be accomplished by clicking on View > Surface in the FIS Editor.

The rule surface shows the output value for any combination of the two input values. Now that all of the rules have been defined and all of the membership functions have been defined the FIS can be exported to the workspace. To do this click: File > Export > To Workspace... and then enter in the workspace variable, for this example we can use tank.
Now in the Simulink model double click the fuzzy logic controller and enter tank for the FIS matrix parameter. This is important or else the Simulink model will now know where to look for the fuzzy control system.

The Simulink model can now be simulated for 10 seconds with all of the previous water tank parameters and the FIS matrix imported to the workspace. The resulting water level for the system can be seen through a scope on the output:

![Fig.20: Output of fuzzy logic controller](image1)

![Fig.21: Output of PID controller](image2)

![Fig.22: Graphical comparison between fuzzy logic & PID controller](image3)
III. CONCLUSION

Fuzzy Logic was conceived as a better method for sorting and handling data but has proven to be an excellent choice for many control system applications since it mimics human control logic. It can be built into anything from small, hand-held products to large computerized process control systems. It uses an imprecise but very descriptive language to deal with input data more like a human operator. It is very robust and forgiving of operator and data input and often works when first implemented with little or no tuning.

Fuzzy systems are indicating good promise in consumer products, industrial and commercial systems, and decision support systems. The term fuzzy refers to the ability of dealing with imprecise of vague inputs. Instead of using complex mathematical equations, fuzzy logic uses linguistic descriptions to define the relationship between the input information and the output action. In engineering systems, fuzzy logic provides a convenient and user friendly front end to develop control programs, helping designers to concentrate on the functional objectives, not on the mathematics.

REFERENCES