Honours Degree in Computing

Computer intelligence Assessment: 1
Fuzzy Pump Controller

Submitted by: Stephen Blaney, B00076157

Submission date 9h March 2018

Declaration

I declare that this material, which I now submit for assessment, is entirely my own work and has not been taken from the work of others, except where otherwise stated. I have identified and included the source of all facts, ideas, opinions, and viewpoints of others in the assignment references. Direct quotations from books, journal articles, internet sources, module text, or any other source whatsoever are acknowledged, and the source cited are identified in the assignment references.

I understand that plagiarism, collusion, and copying are grave and serious offences and accept the penalties that would be imposed should I engage in plagiarism, collusion or copying. I acknowledge that copying someone else's assignment, or part of it, is wrong, and that submitting identical work to others constitutes a form of plagiarism. I have read and understood the colleges plagiarism policy 3AS08 (available here).

This material, or any part of it, has not been previously submitted for assessment for an academic purpose at this or any other academic institution.

I have not allowed anyone to copy my work with the intention of passing it off as their own work.

Name: Stephen Blaney Dated: 26/10/2017

Background information

The objective of this assignment is to design and implement a fuzzy logic pump controller. The purpose of the pump is to keep the level of water in the tank between 40% and 70%. It must also keep this water in this range while demand from appliances continues to drain water, as well as increase water level from rainfall. The appliance further down the line can consume water from the tank and can also feed water back into the tank. These are represented by checkboxes within the application and should be used to ensure the system works under all conditions (pump, rain, demand) The problem will be simulated using a java applet the was created using the fuzzyLite API within the eclipse IDE.

Creation of linguistic variables

The first step in implementing this project was the creation of linguistic variables and terms that represented our input and output variables. It's important that we set a domain range for our variables as we are trying to model a real-world system. Both Input and output variable are given to us in the project brief along with their associated domain ranges.

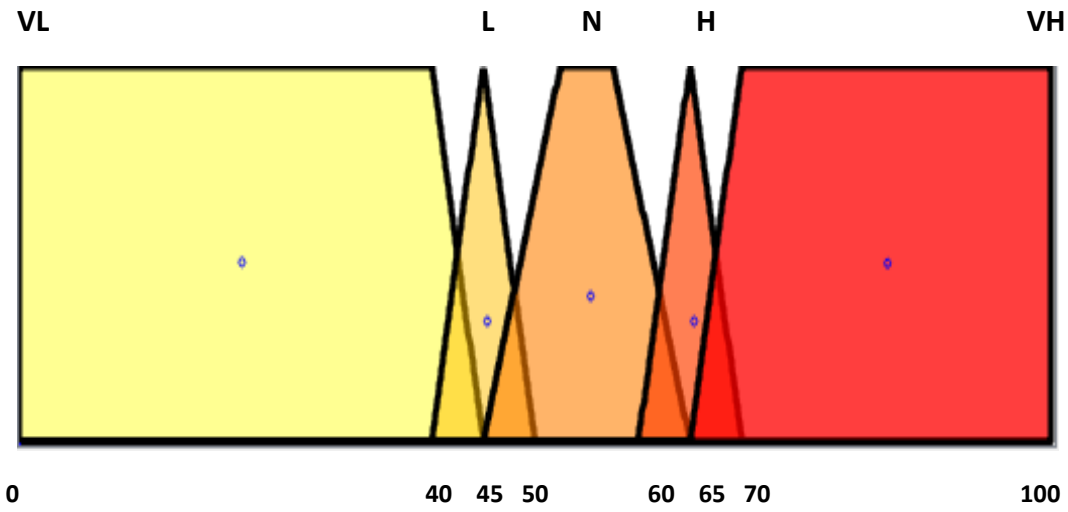
Inputs	Outputs
Level between 0 and 100	Command between -1 and 1
Demand between -1 and 1.5	

After setting the domain range for each variable a set up term is drawn up to describe each variable. All variables have the same term names in this case. For each variable we split the ranges so that it's a lot easier to understand

Term	Description
VL	Very_low
L	Low
N	Normal
H	High
VH	Very_high

Level membership function (input 1)

Next step in step in our implementation is to create membership functions for two of our input variables and one output variable. The following consist of screenshots from the fuzylite GUI application that illustrates our membership functions for level, demand and command.

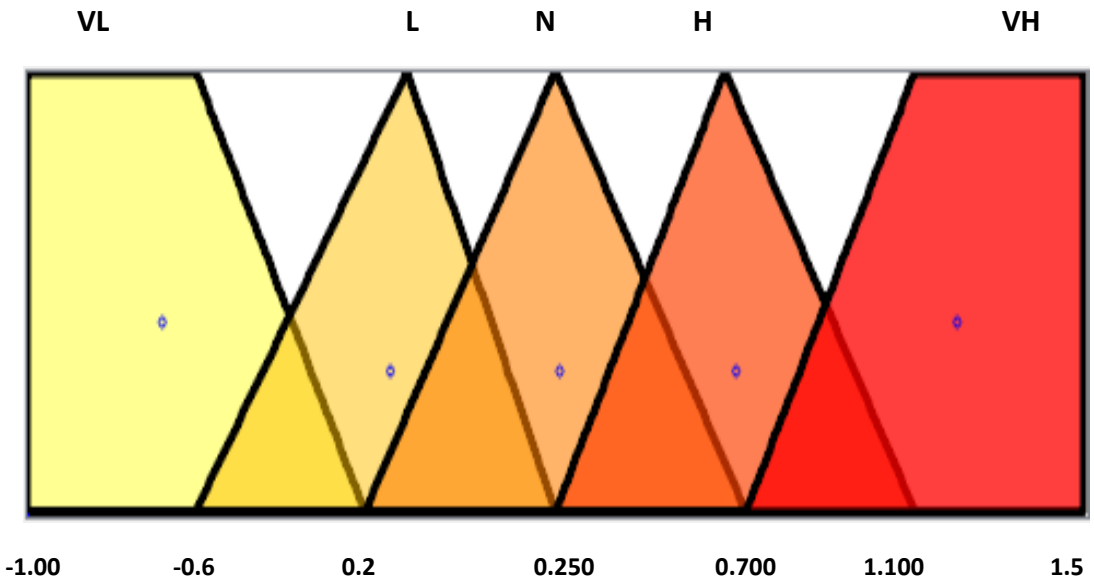


Term	Range
VL	Trapezoid 0.00, 0.00, 40.00, 45.00
L	Triangle 40.00, 45.00, 50.00
N	Trapezoid 45.00, 52.5 ,57.5, 65.00
H	Triangle 60.00, 65.00, 70.00
VH	Trapezoid 65.00, 70.00, 100.00, 100.00

The first thing to consider when designing the membership functions is dividing each of the terms into different ranges. In the project specs we are given the range where the water should be between (40%-70%) so we have a trapezoid shape that reaches out to these values (As in the screenshot above).

This was calculated by splitting the ranges initialling in half example the range 0-100 halved would give a value of fifty this would be our middle value. Then we do the same thing again by getting the middle range between 40-50 which gives 45. The shapes generated are arbitrary and where based on trial and error

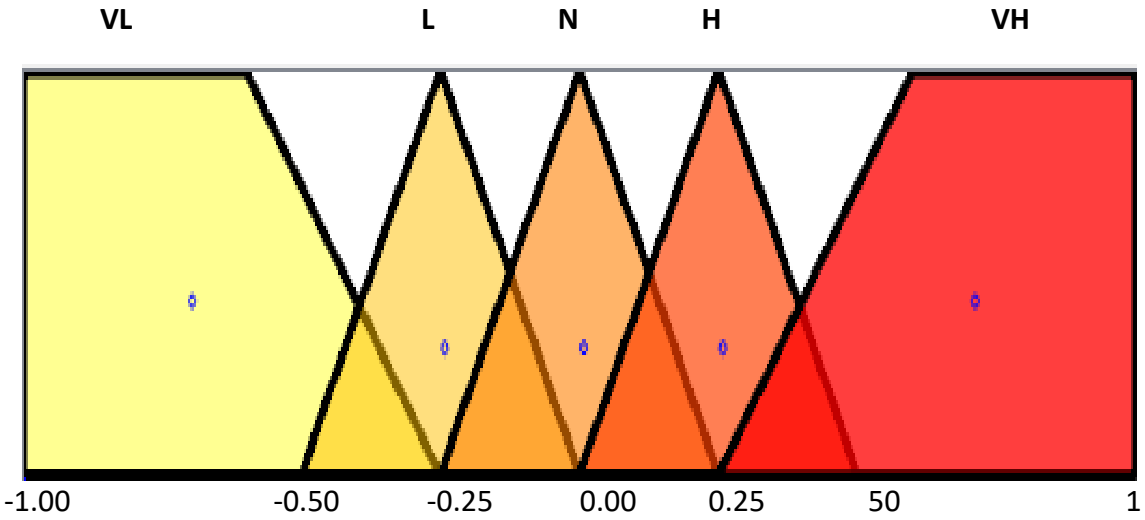
Demand membership function (input 2)



Term	Range
VL	Trapezoid -1.00, -1.00, -0.6, -0.200
L	Triangle -0.600, -0.100, 0.250
N	Triangle -0.200, 0.250, 0.700
H	Triangle 0.250, 0.650, 1.100
VH	Trapezoid 0.700, 1.0, 1.5, 1.5

Most of the remaining membership functions where derived the same way as the first one by halving values between a range (half way between -0.6 and 0.250 = 0.2) In the case of demand this a bit difficult as it's domain range was between -1.00 and 1.5 and it wasn't exactly easy to get ab even spilt between them.

Command membership function (output variable)



Term	Range
VL	Trapezoid -1.00, -1.00, -0.60, -0.250
L	Triangle -0.500, -0.250, 0.00
N	Triangle -0.250, 0.00, 0.250
H	Triangle 0.00, 0.250, 0.500
VH	Trapezoid 0.250, 0.600, 1.00, 1.00

Fuzzy Rules

Now that the linguistic variables and terms have been defined the number of rules will need to be written that take the two-input variable level and demand as input in order to decide on an output for our command variable. Initially the program had 17 rules. It was decided that this was too much so in the end it was reduced to 10 rules. This was tested under all conditions (Rain and demand) to show the pump keeping the water level in between 40 and 70 percent

Level	VL	L	N	H	VH
Demand					
VL	VL	VL	**VL**	**L**	N
L	VL	L	**L**	**N**	**VH**
N	VL	L	**N**	H	VH
H	VL	N	**H**	VH	**VH**
VH	**N**	H	**VH**	VH	VH

Defuzzification

Our rules contain connective statements like AND we will have to define how these Boolean operators are handled in our fuzzy logic system. It is important to select an accumulation and defuzzifier function for the output variable. In fuzzylite we do this by clicking on defuzzifier and selecting centroid we also click on the Accumulation and select maximum this is how our fuzzy system produces CRISP value which we can use to keep the level in the tank in between the specified range.