*Tópicos Especiais em Instrumentação I, 2020, UFRGS, DELET*

***Tópicos Especiais em Instrumentação I***

*Relatório*

**Introdução à Lógica Fuzzy para Instrumentação Inteligente**

**Frederico May de Lize Matheus Quevedo Sivelli**

Universidade Federal do Rio Grande do Sul, Departamento de Engenharia Elétrica, Curso de

Engenharia Elétrica, Instrumentação A, Prof. Dr. Tiago Oliveira Weber

E-Mails: fredericoliz25@gmail.com (F.M.L); Matheus\_sivelli@hotmail.com (M.Q.S)

*Data Início: 20/06/2020; Data Final: 10/07/2020*

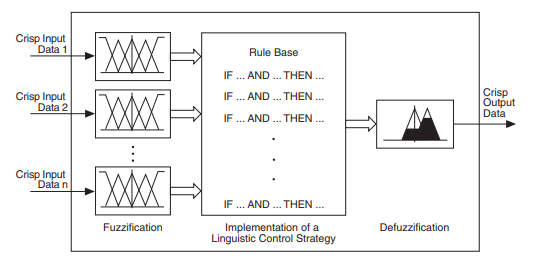
**Resumo:** O presente trabalho tem como principal objetivo de utilizar as técnicas aprendidas de aprendizado de máquina juntamente com a lógica Fuzzy para construirmos um controlador Fuzzy. Para atingirmos tal objetivo, faremos uso de um simulador de corrida obtido na internet e customizado para nós melhor atender dentro do nosso problema. O objetivo do nosso controlador é deixar o nosso carro no piloto automático dentro do nosso perímetro permitido, ou seja, caso ocorra algum acidente, quando o carro sair da pista, o jogo é automaticamente fechado. Por fim, iremos comparar e discutir a abordagem do controlador Fuzzy com outro controlador tradicional que iremos implementar.

**Abstract:** This work aims to know the method of Fuzzy controllers for intelligent instrumentation. It is used the Matlab© tool with its Fuzzy Toolbox tool in the systems solution and understand its concepts. In addition, the work will use concepts of interval logic and development of Fuzzy controllers. These objectives will be achieved by simulating three different Fuzzy systems using Matlab© and LabView computational tools along with their theoretical analysis. The first system will mainly address the analysis of computational tools. The remaining two systems will focus on the development of a Fuzzy controller for a hydroponic vegetable cultivation automation system and for temperature control, which will be interconnected with the course's laboratory.

**Palavras Chaves:** Fuzzy; Python; Instrumentação; Machine Learning;

**1. Introdução**

A lógica fuzzy é uma abordagem para o processamento de dados baseado no grau de pertencimento em detrimento do comum verdadeiro ou falso (1 ou 0) da lógica booleana. Esta lógica é mais intuitiva e acessível pois processa dados de forma parecida com a forma que nosso cérebro interpreta no cotidiano. Isso acontece de forma natural, com a associação de dados com intervalos de verdades parciais, de modo que nem todo dado está dentro de um intervalo apenas, podendo ter dupla interpretação. Assim a lógica fuzzy abre mão da precisão, porém facilita a resolução de problemas, trata de maneira mais clara das imprecisões e é mais intuitivo, principalmente se a aplicação será utilizada por uma parcela da população que não tem conhecimento intermediário sobre engenharia. A Figura 1 demonstra o processo de descrição de um controlador fuzzy, com a exibição de suas principais partes.



**Figura 1.** Processos de um controlador *Fuzzy*.

**Fonte –** LabVIEW: PID and Fuzzy Logic Toolkit User Manual.

Após um controlador fuzzy ‘fuzzyficar’ os valores de entrada de um sistema, o controlador fuzzy usa os termos linguísticos de entrada correspondentes e a base de regras para determinar os termos linguísticos resultantes das variáveis linguísticas de saída. A lógica fuzzy nós permite descrever sistemas extremamente complexos apenas com valores qualitativos, e que não necessitam de amplo conhecimento prévio, por exemplo, conseguimos implementar lógicas extremamente complexas no ponto de vista físico, mas que partam de uma premissa até então simplória, como é o caso de um piloto automático, que utilizando valores qualitativos como ‘devagar’, ‘rápido’ e ‘descentralizado’ conseguem nós trazer um resultado satisfatório em comparação com os controladores tradicionais que requerem uma matemática mais avançada da planta e da função de transferência aplicada no sinal de controle.

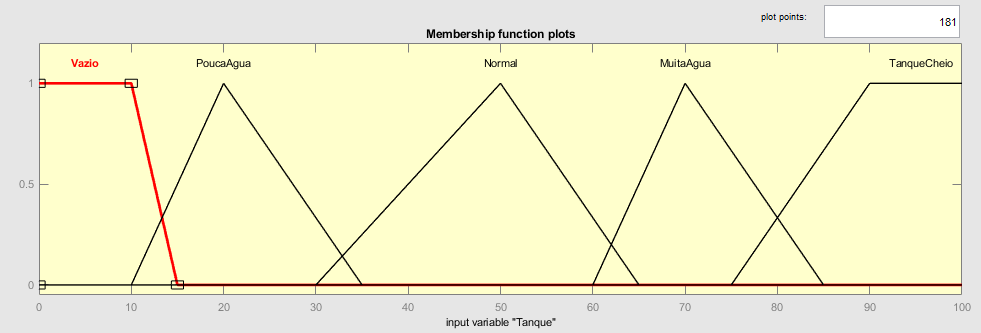
Neste relatório é utilizaremos a teoria por trás dos controladores fuzzy para criarmos um piloto automático para nosso carro. Inicialmente, construímos sensores laterais e frontais para conseguirmos armazenar os dados conforme íamos dirigindo o carro manualmente, assim que obtivemos uma quantidade considerável de dados, partimos para a abordagem de implementar um controlador sob as premissas fuzzys.

Após a implementação do nosso controlador fuzzy, buscamos implementar um controlador sob as teorias tradicionais de controle para compararmos o desempenho de cada um.

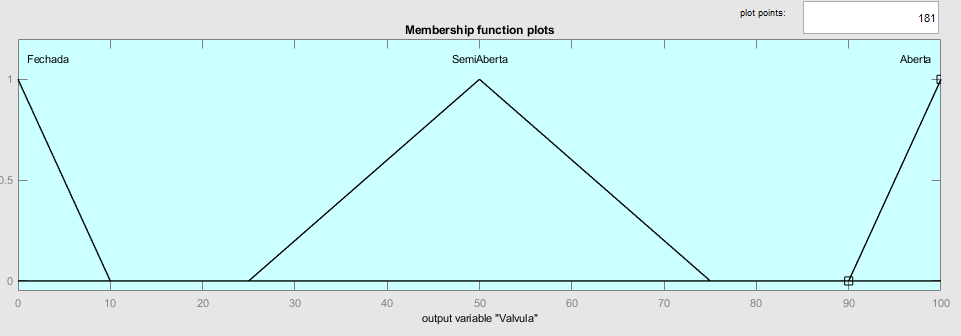
**2. Metodologia Experimental**

*2.1 Exercício 1: Implementação de um sistema fuzzy utilizando Matlab© e LabVIEW©*

Neste exercício, será realizado uma discussão e implementação ao mesmo tempo de diversos fatores nos softwares Matlab© e LabVIEW. Para isso, será implementado um sistema fuzzy para controlar um tanque de água genérico. Considerando que a variável de entrada é a quantidade de água e o atuador de saída será a válvula de escape. Para a variável de entrada da quantidade de água, os valores serão definidos entre 0L e 100L. Serão criados 5 conjuntos linguísticos: vazio, pouca água, normal, muita água e tanque cheio. Já para o atuador de saída, será definido uma porcentagem de abertura para a válvula de escape. Serão criados 3 conjuntos linguísticos: válvula fechada, válvula semi aberta e válvula totalmente aberta. Os graus de pertencimento do tanque e da válvula estão apresentados na Figura 2 e 3.



**Figura 2.** Grau de Pertinência da variável de entrada “Tanque” pela interface FIS do *software* Matlab© (MathWorks Inc., Massachusetts, EUA).



**Figura 3.** Grau de Pertinência da variável de saída “Válvula” pela interface FIS do *software* Matlab© (MathWorks Inc., Massachusetts, EUA).

O Quadro 1 apresenta a matriz de regras Fuzzy (FAM) para relacionar a variável de quantidade de água no tanque com o comportamento dos atuadores de saída do sistema, nesse caso a válvula.

**Quadro 1.** Matriz de regras Fuzzy (FAM) para exemplificar o exercício 1.

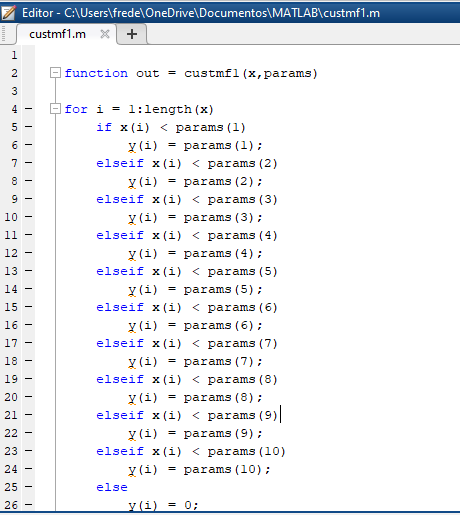
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Quantidade de Água no Tanque | | | | |
| Vazio | Pouca Água | Normal | Muita Água | Tanque Cheio |
| Fechada | Fechada | Semi Aberta | Semi Aberta | Aberta |

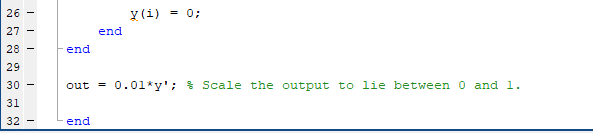
Com a utilização desse sistema fuzzy como exemplo, abordar-se-á primeiramente as utilidades via Matlab© e posteriormente via LabVIEW.

*2.1.1 Construir um sistema fuzzy usando funções customizadas*

Ao criar um sistema de inferência fuzzy no *software* Matlab© (MathWorks Inc., Massachusetts, EUA), você pode substituir os graus de pertinência, funções de inferência ou ambas por funções personalizadas. Será analisado como criar um sistema de inferência fuzzy usando funções personalizadas no aplicativo *Fuzzy Logic Designer*.

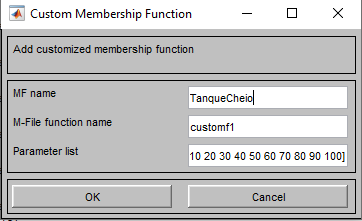
Pode-se criar graus de pertinência personalizadas e usá-las no processo de inferência *fuzzy* diretamente pelo script. Os valores dessas funções devem estar entre 0 e 1. Para criar um grau de pertinência personalizado e substituir a função de pertinência interna, deve-se criar uma função e salvar na pasta de trabalho atual. A Figura 4 mostra um exemplo de função personalizada de várias etapas, que depende de dez parâmetros entre 0 e 10, para o exemplo do tanque.



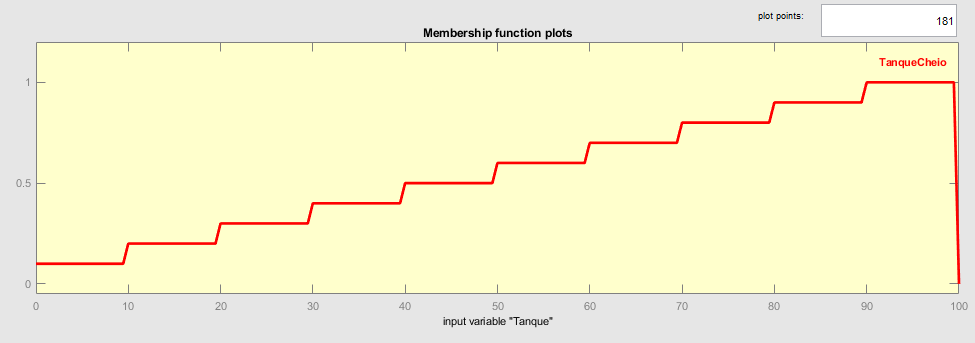


**Figura 4.** Código de criação de uma função personalizada.

Para substituir a função de associação padrão por uma função personalizada no Membership Function Editor, deve-se entrar em “Edit” e em “Add Custom MF” para finalmente poder adicionar a função personalizada como na Figura 5, para esse caso, se o grau de pertinência para o tanque cheio fosse uma função que aumentaria em degrau. Na Figura 6 é possível analisar a função de pertinência criada para o Tanque Cheio.



**Figura 5.** Janela do Custom Membership Function..



**Figura 6.** Função de pertinência para o Tanque Cheio.

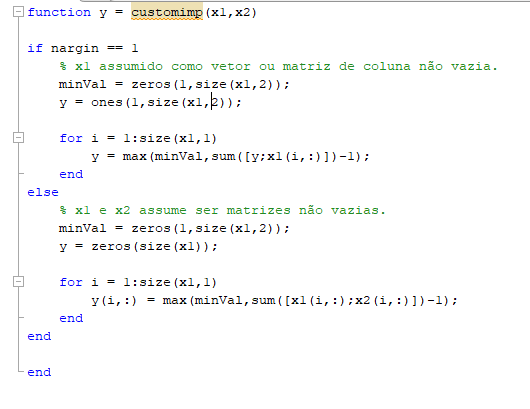
É possível também adicionar uma função personalizada utilizando a linha de comando do Matlab©, como visto pela Figura 7.



**Figura 7.** Código para adicionar a função de pertinência para o Tanque Cheio.

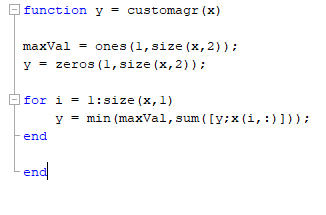
As funções de inferência AND e OR personalizadas devem operar em colunas em uma matriz, da mesma maneira que as funções Matlab© *max, min* ou *prod*. Para um vetor de linha ou coluna x, *min(x)* retorna o elemento mínimo. A função *min(x, y)* retorna uma matriz que é do mesmo tamanho que x e y com os elementos mínimos de x ou y. Qualquer um dos argumentos de entrada pode ser escalar. Funções como *max* e *prod* operam de maneira semelhante. Na caixa de ferramentas, os métodos de implicação AND executam uma operação de matriz elemento a elemento, semelhante à função Matlab© *min(x, y)*.

As funções de implicação personalizadas devem operar da mesma maneira que as funções Matlab© max, min ou prod. Uma função de implicação deve suportar uma ou duas entradas, porque o software chama a função para: calcular os valores definidos difusos de saída usando a força de disparo de todas as regras e as funções de associação de saída correspondentes; Para calcular o valor difuso da saída usando a força de disparo de uma única regra e a função de associação de saída correspondente, para uma determinada amostra. A Figura 8 é apresentado um código de uma função de implicação customizada de produto delimitado com mapeamento binário T(a, b) = max {0, a + b-1}.



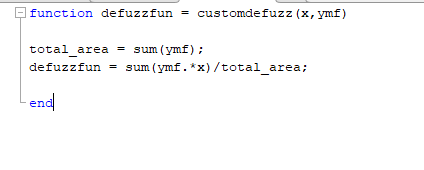
**Figura 8.** Código de uma função de implicação customizada.

As funções de agregação customizadas, como as funções de implicação, devem operar da mesma maneira que as funções *max, min* ou *prod* e devem ter a forma *y=customagr(x).* Sua função de implicação customizada deve ser uma operação de interseção fuzzy da norma T-conorm (S-norm). Supondo que x é uma matriz *nv* por *nr*, que é a lista de funções de saída truncadas retornadas pelo método de implicação para cada regra. *nv* é o número de variáveis de saída e *nr* é o número de regras. A saída do método de agregação é um conjunto difuso para cada variável de saída. Na Figura 9 é apresentado um exemplo de uma função de agregação personalizada de soma limitada com mapeamento binário S(a, b) = min{a + b, 1}.



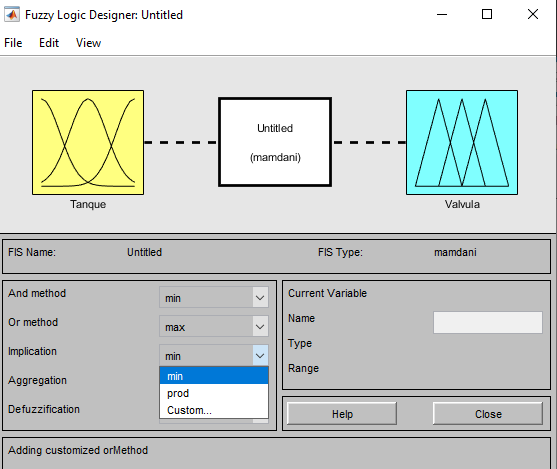
**Figura 9.** Código de uma função de agregação customizada.

As funções de defuzzificação personalizadas devem ter o formato y=customdefuzz (x,ymf), em que *x* é o vetor de valores no intervalo de entrada da função de associação e *ymf* contém os valores da função de associação para cada valor de x. Na Figura 10 observa-se um exemplo de uma função de defuzzificação.

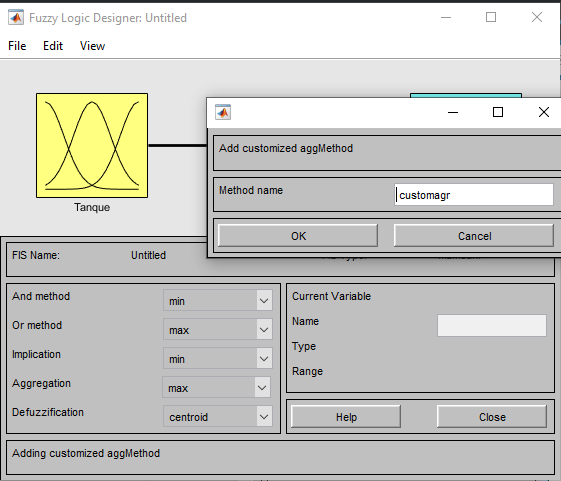


**Figura 10.** Código de uma função de defuzzyficação customizada.

Para adicionar as funções customizadas criadas por código e implementar no sistema fuzzy do exemplo, basta apertar em cada uma das abas de opções e apertar em “custom” como visto na Figura 11. Após isso, basta escrever o nome da função criada para cada opção como na Figura 12.



**Figura 11.** Opção ‘custom’ para o Fuzzy Logic Designer.



**Figura 12.** Adição da função customizada para o Fuzzy Logic Designer.

É possível também adicionar as funções personalizadas diretamente pela linha de comando do Matlab©, como pode se observar nas Figuras 13, 14 e 15.



**Figura 13.** Adição de defuzzificação customizada via linha de comando.



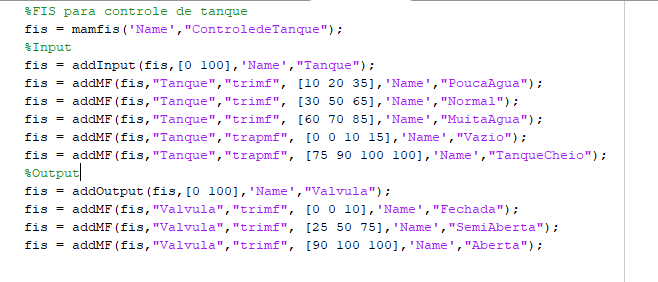
**Figura 14.** Adição de implicação customizada via linha de comando.



**Figura 15.** Adição de agregação customizada via linha de comando.

*2.1.1 Como trabalhar com linhas de comando para fuzzy no Matlab©*

Como alternativa ao uso do aplicativo *Fuzzy Logic Designer*, você pode construir um FIS inteiramente a partir da linha de comando. Primeiro, crie um Mamdani FIS, especificando seu nome. Para o exemplo do tanque de água, adicione a primeira variável de entrada para o tanque usando addInput. Será adicionado funções de associação para cada um dos níveis de água usando o addMF conforme a Figura 16.

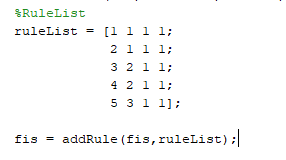


**Figura 16.** Código para criação de grau de pertinência das variáveis de entrada e saída por linha de comando.

Para a criação das regras via linha de comando, é necessário criar uma matriz numérica em que cada linha da matriz contém uma regra seguindo o formato:

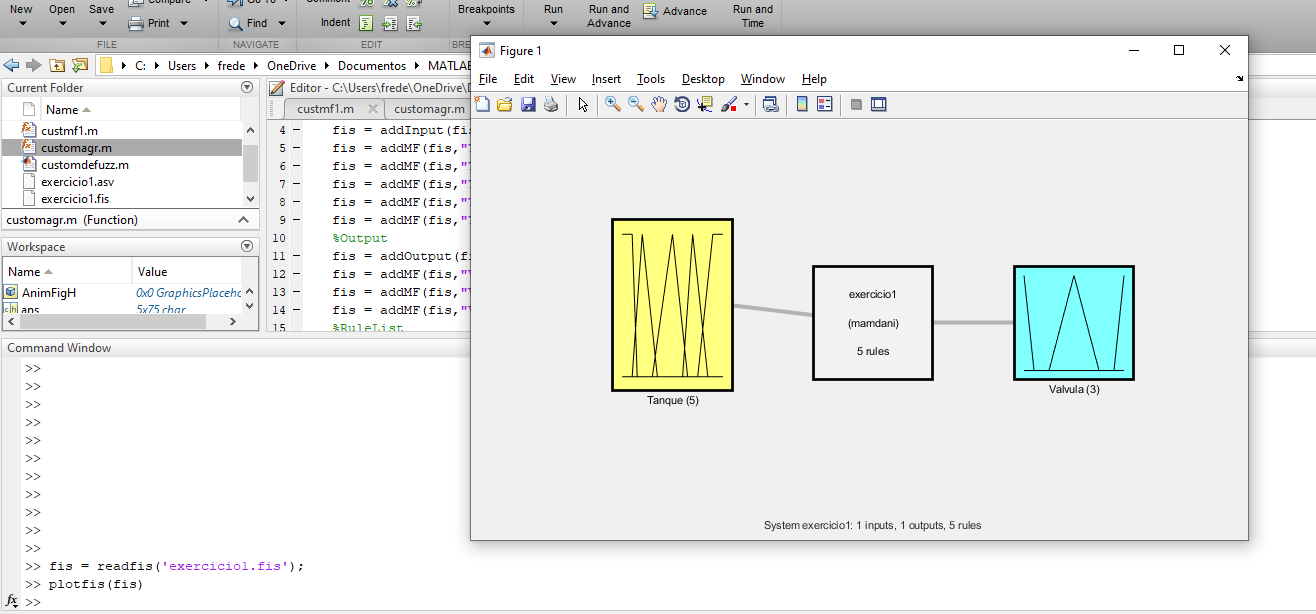
* Coluna 1 - Índice da função de associação para a primeira entrada
* Coluna 2 - Índice da função de associação para saída
* Coluna 3 - Peso da regra (de 0 a 1)
* Coluna 4 - Operador fuzzy (1 para AND, 2 para OR)

Na Figura 17 é possível observar as regras criadas anteriormente no Diagrama 1 em linha de comando do Matlab©.



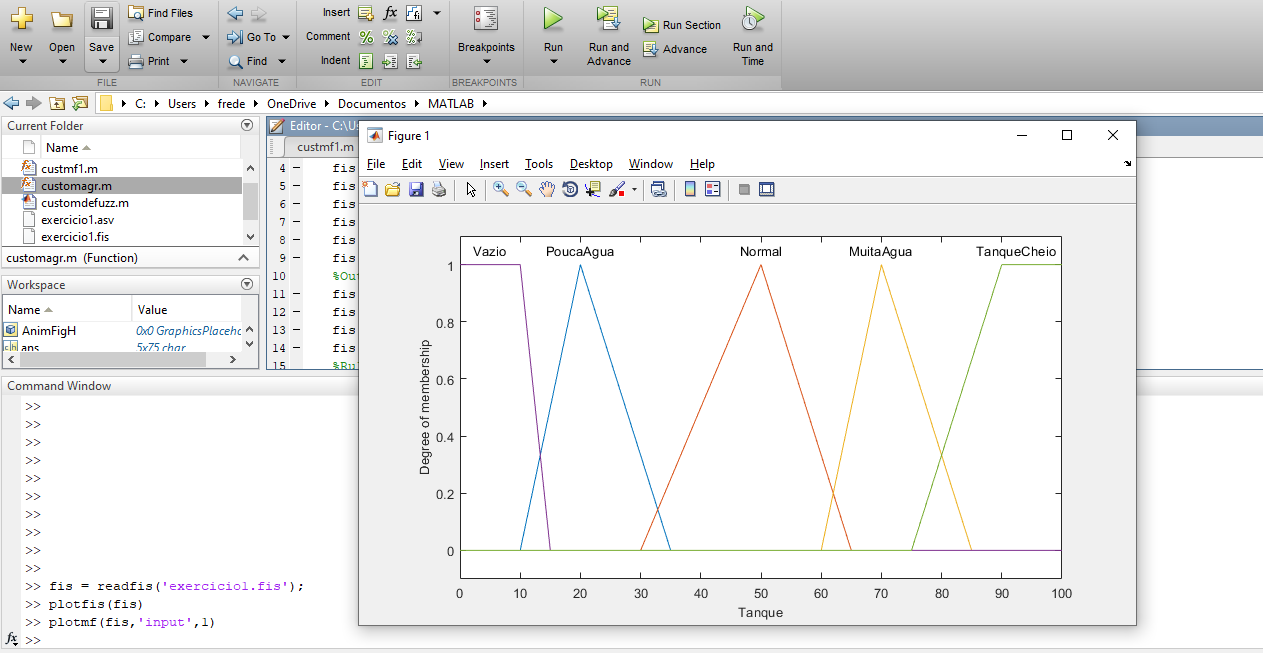
**Figura 17.** Código das listas de regras por linha de comando.

Para obter uma visão de alto nível do seu sistema fuzzy na linha de comando, usa-se as funções *plotfis*, *plotmf* e *gensurf*. O *plotfis* exibe todo o sistema como um diagrama de blocos, conforme mostrado na Figura 18.

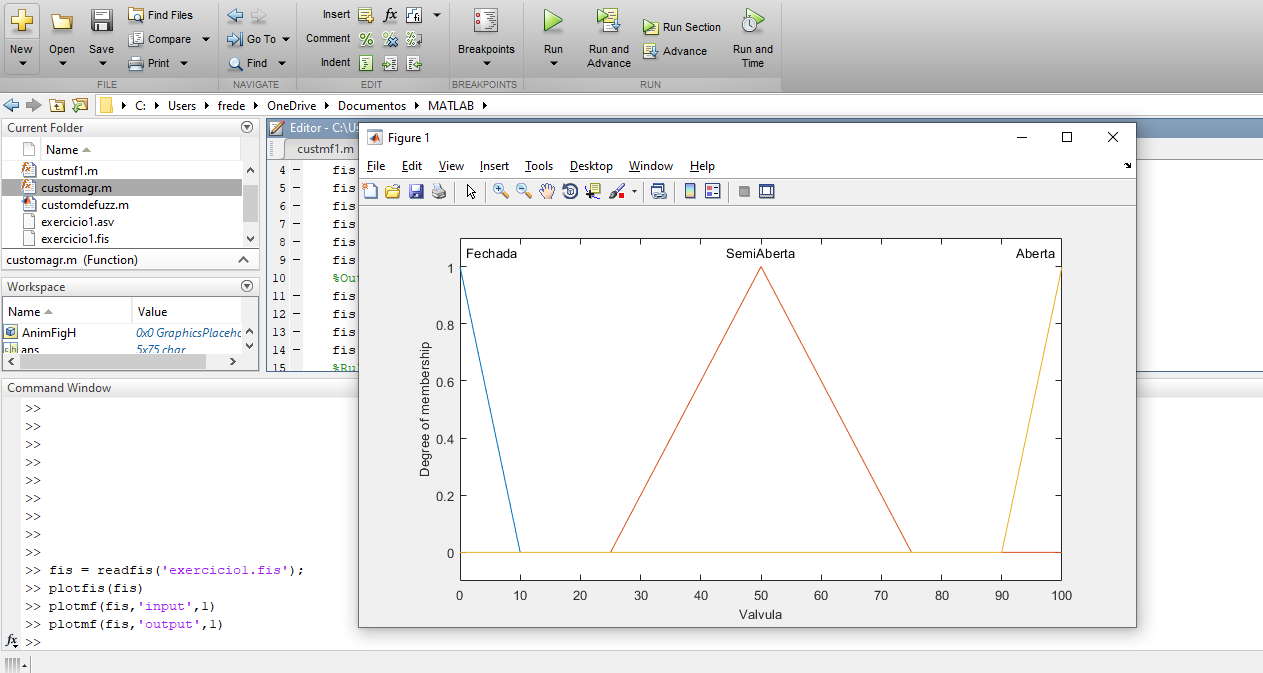


**Figura 18.** Utilização do comando *plotfis* em linha de comando.

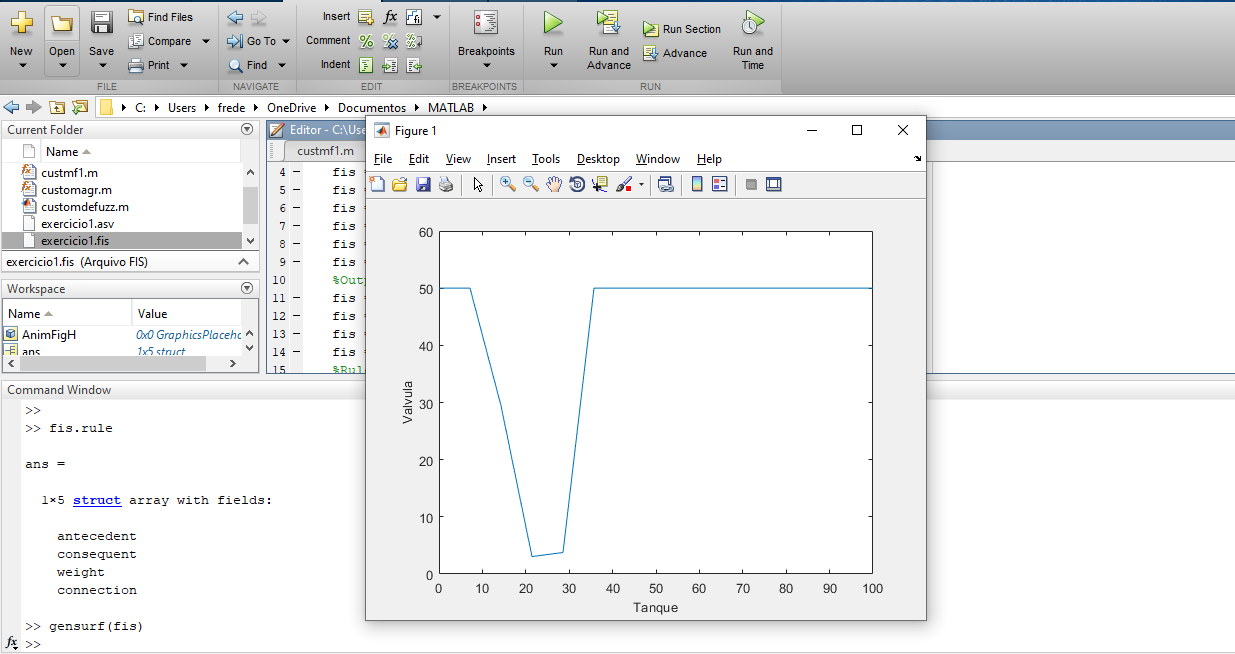
A função *plotmf* plota todas as funções de pertinência associadas a uma determinada variável. Nas Figuras 19 e 20 é possível verificar as funções de pertinência para as variáveis de entrada e saída respectivamente.



**Figura 19.** Utilização do comando *plotfis* em linha de comando para as variáveis de entrada.

**Figura 20.** Utilização do comando *plotfis* em linha de comando para as variáveis de saída.

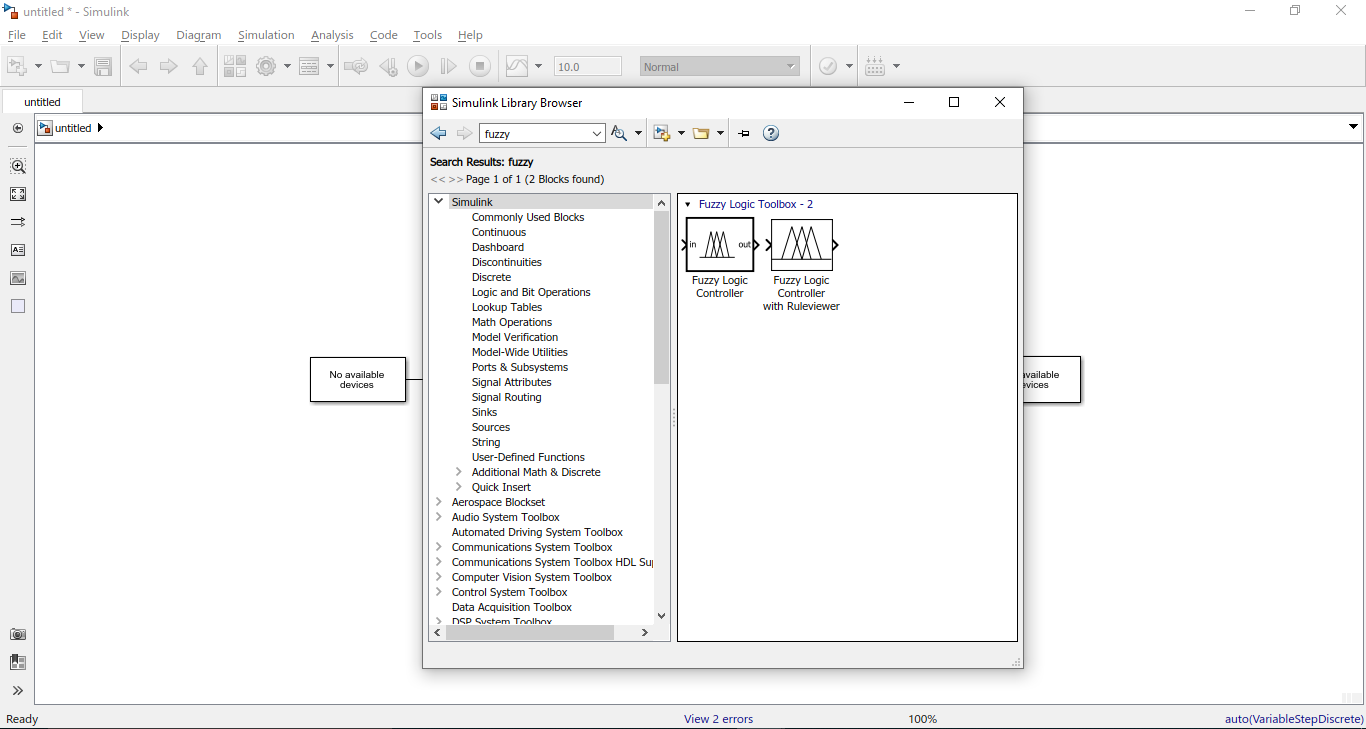
Para verificar as regras do sistema se utiliza o comando *fis.rule* e a função *gensurf* plota a saída do FIS para qualquer uma ou duas variáveis de entrada. A Figura 21 exemplifica a utilização desses comandos para o sistema fuzzy de controle de tanque.



**Figura 21.** Utilização do comando *fis.rule e gensurf(fis)* em linha de comando.

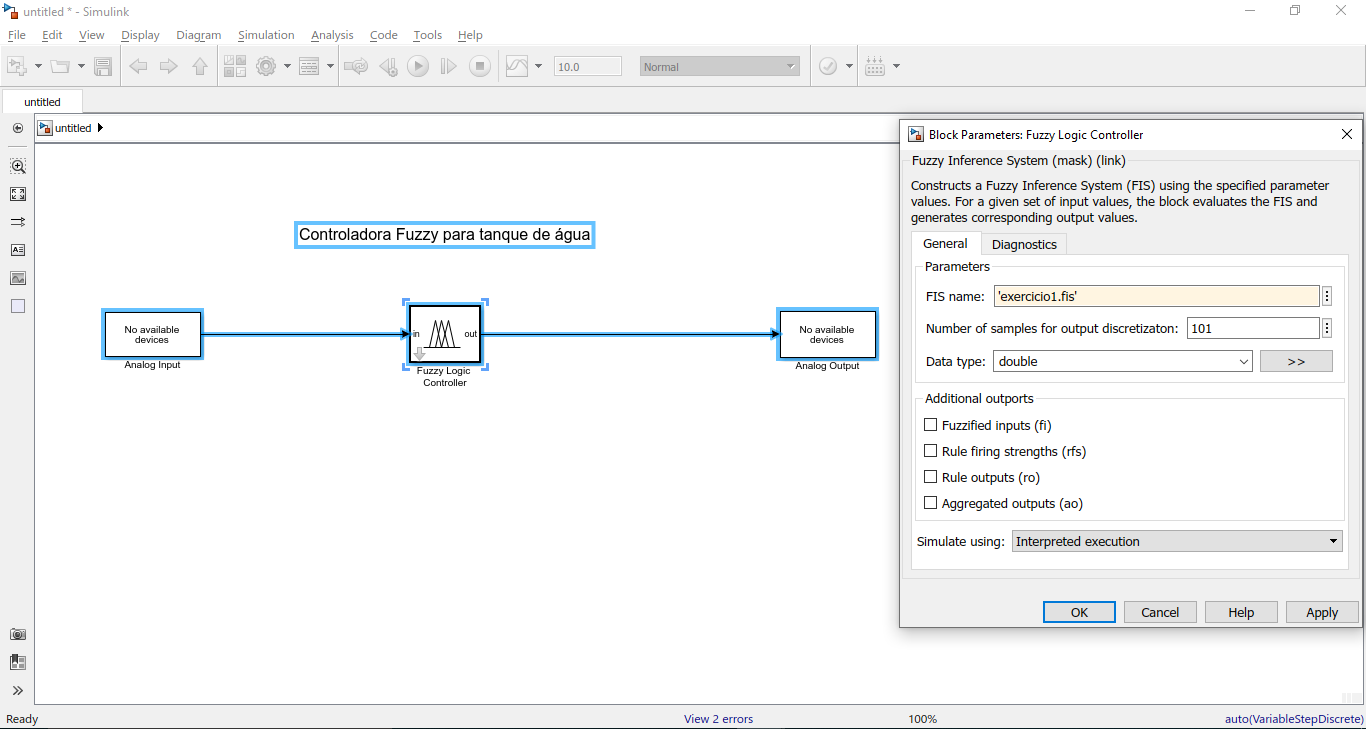
*2.1.3 Como trabalhar no ambiente SIMULINK com fuzzy*

Depois de implementar um sistema de inferência fuzzy usando o Fuzzy Logic Designer ou na própria linha de comando, você pode simular o sistema no Simulink. Para isso, basta adicionar no diagrama de blocos da simulação do Simulink o bloco Fuzzy Logic Controller da biblioteca do próprio programa, como visto na Figura 22.

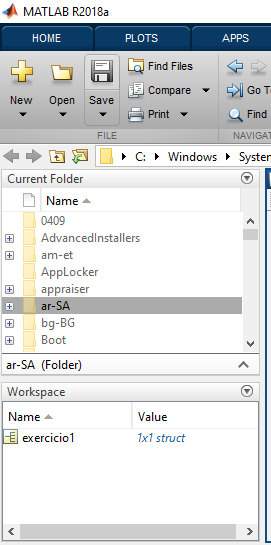


**Figura 22.** Biblioteca do Simulink com os blocos Fuzzy.

Para implementar um sistema fuzzy, se especifica o parâmetro de nome FIS do bloco Fuzzy Logic Controller como o nome de um objeto FIS na área de trabalho do Matlab©. Para o exemplo do projeto do tanque, o bloco usa o objeto *mamfis* ‘exercicio1’, como visto na figura 23 e 24.



**Figura 23.** Diagrama de blocos do exemplo juntamente com a janela de ajuste de parâmetros do bloco *Fuzzy Logic Controller*.

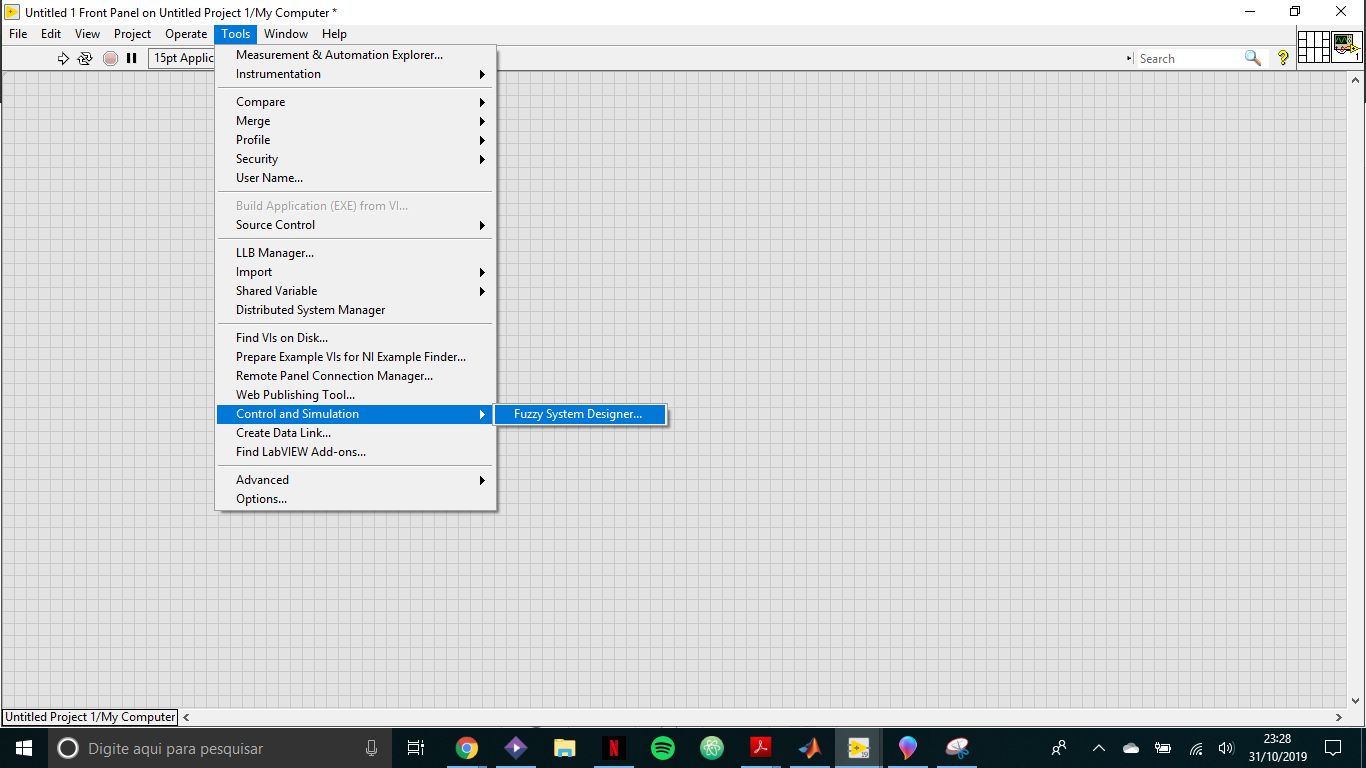


**Figura 24.** Objeto carregado no ambiente do Matlab© para a simulação.

Após a configuração do bloco, é possível facilmente simular a controladora criada com o Fuzzy System Designer com o Simulink. Ademais, pode-se também simular sistemas fuzzy usando o *Fuzzy Logic Controller* com o bloco *Ruleviewer* onde também pode ser observado na Figura 22*.* A grande diferença desse bloco é que durante a sua simulação, o bloco exibe o *Rule Viewer* no aplicativo *Fuzzy Logic Designer*, sendo uma vantagem para verificação se as regras estão sendo acionadas do jeito que o usuário realmente projetou.

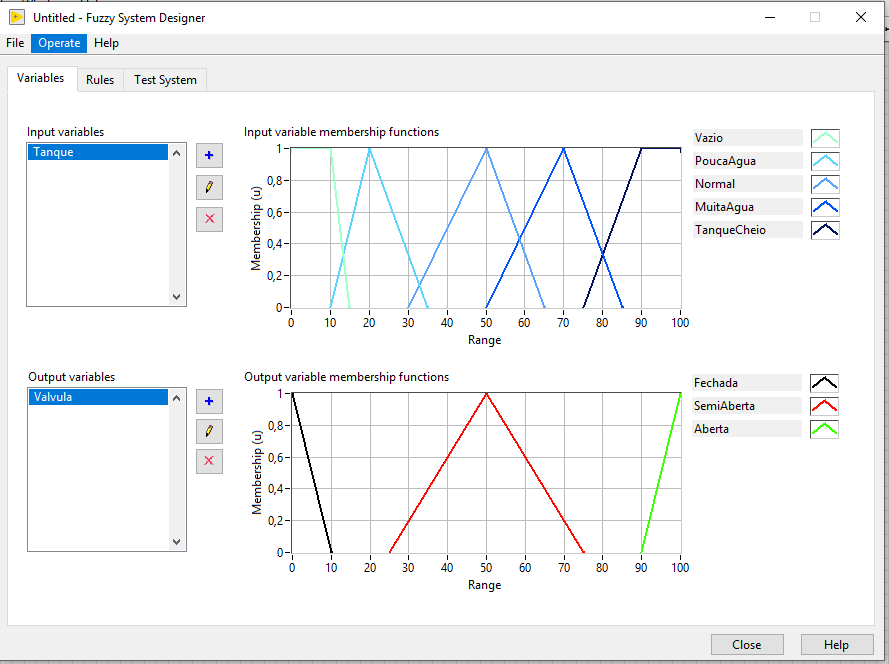
*2.1.4 Como trabalhar com lógica fuzzy no LabVIEW*

É possível também projetar um sistema fuzzy pelo LabVIEW utilizando o Fuzzy System Designer como visto pela Figura 25.

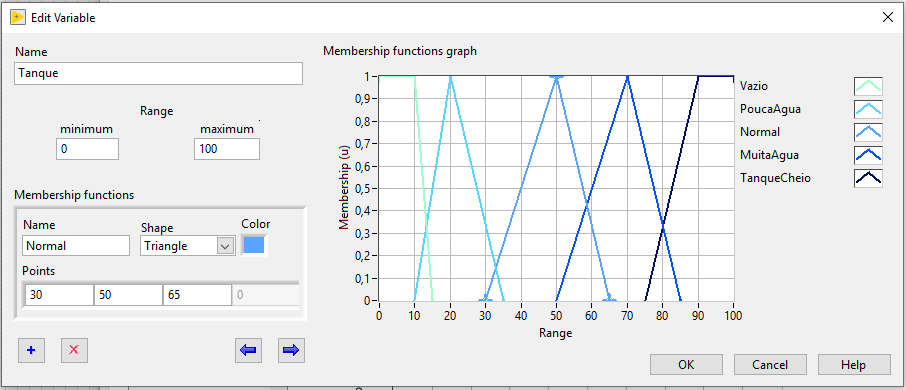


**Figura 25.** localização do *Fuzzy Logic Designer*.

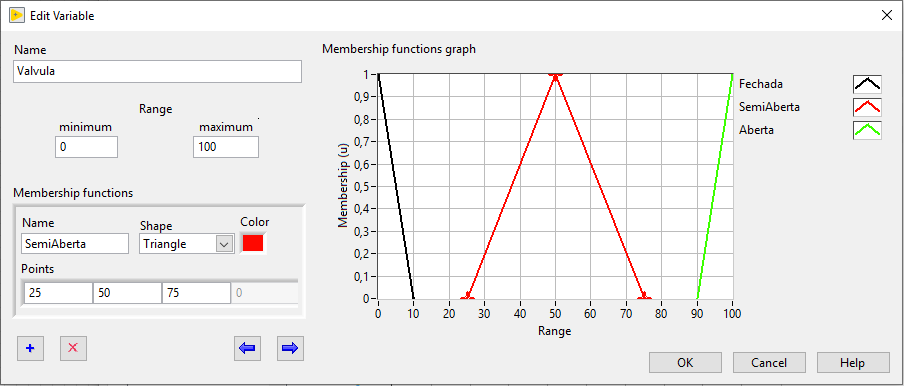
A primeira etapa no design de um sistema fuzzy com o Fuzzy System Designer é criar as variáveis linguísticas de entrada e saída para o sistema. Para os exemplos, será utilizado o mesmo modelo de controlador fuzzy para um tanque de água. A janela é muito intuitiva, sendo possível adicionar a quantidade necessária que precisa para o sistema desenvolvido e sendo possível alterar as cores das funções para aumentar a facilidade em verificar cada função em si. As Figuras 26, 27 e 28 demonstram cada janela de entrada e saída das variáveis para o exemplo proposto no início do exercício.



**Figura 26.** Janela de Variáveis do *Fuzzy Logic Designer*.

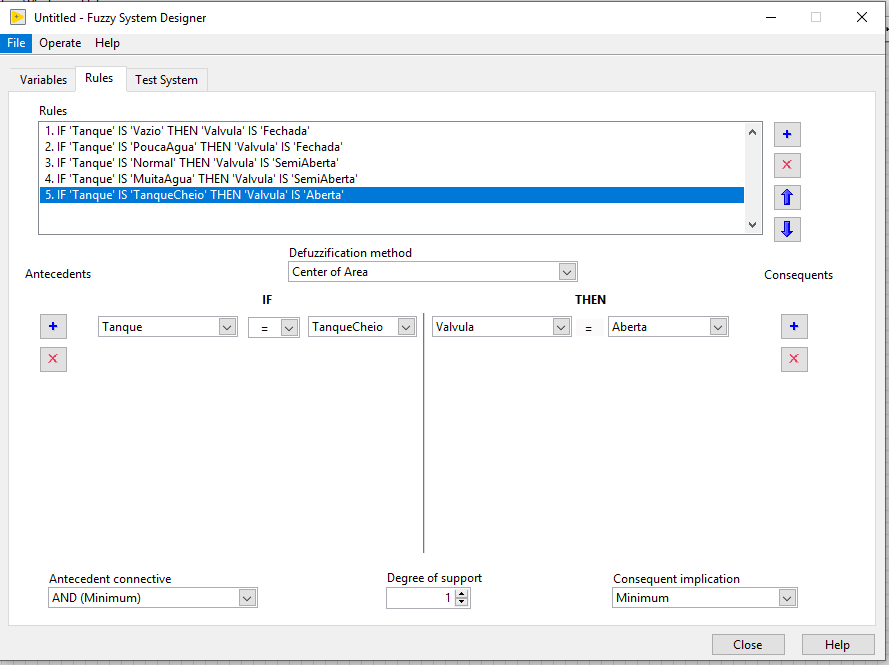


**Figura 27.** Janela de edição de variável de entrada do *Fuzzy Logic Designer*.



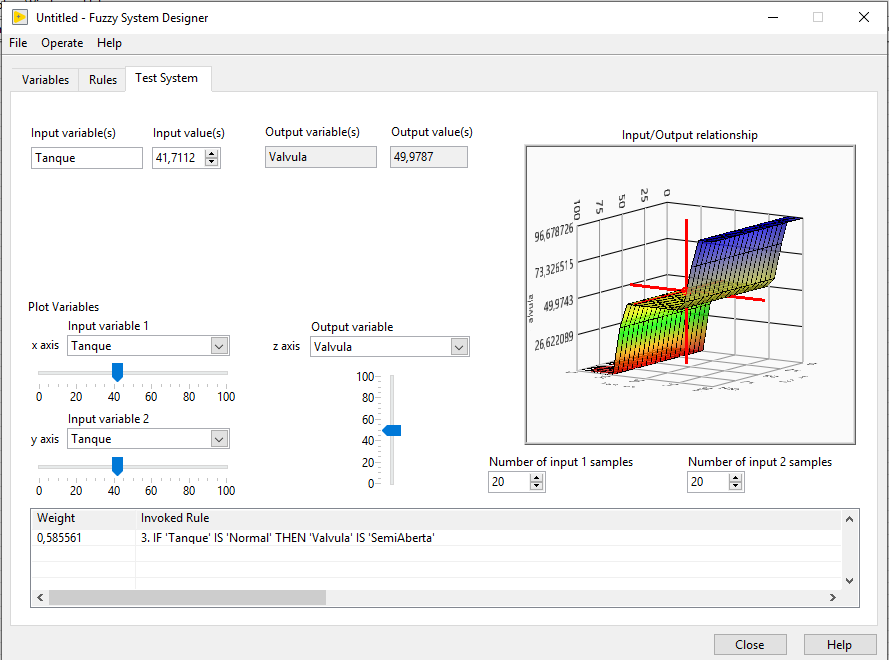
**Figura 28.** Janela de edição de variável de saída do *Fuzzy Logic Designer*.

Para adicionar as regras, se seleciona a aba de regras do Fuzzy system Designer e se adiciona as regras necessárias para o projeto. Ademais, é possível alterar nessa aba o método de defuzzyficação, método de implicação e os tipos de conectivos AND ou OR para o dado sistema. Pela Figura 29 é possível observar a janela com a aba de regras selecionada com as regras propostas no exercício.



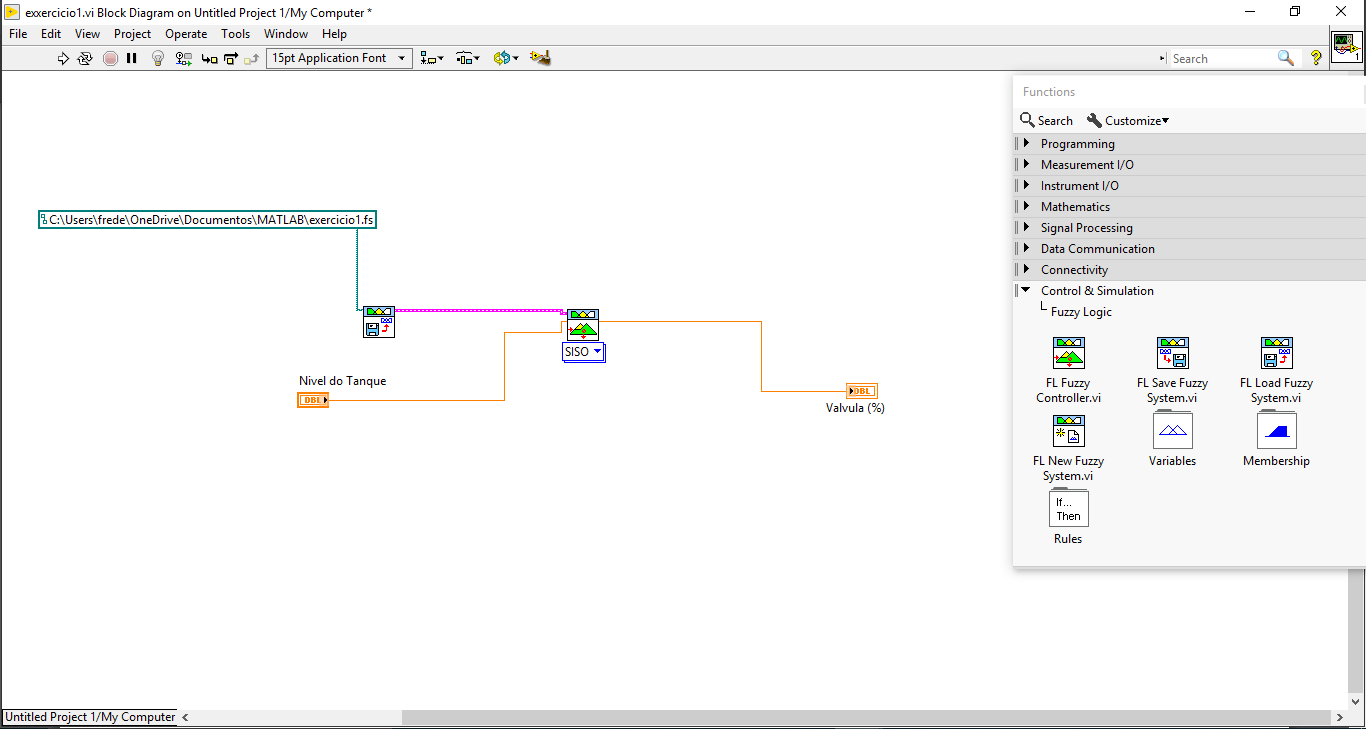
**Figura 29.** Aba de edição de regras do *Fuzzy Logic Designer*.

Para realizar testes no sistema criado com diferentes valores de entrada e verificação de saída, se seleciona a aba “Test System” do Fuzzy Logic Designer. Nessa aba, é possível simular um dado valor de entrada e o programa apresenta o valor da variável de saída. Juntamente com os dados simulados, é possível também avaliar o gráfico de saída e alterar os dados de input e output do sistema. Na Figura 30 pode se analisar o resultado do sistema fuzzy do controlador proposto pela aba “Test System”.

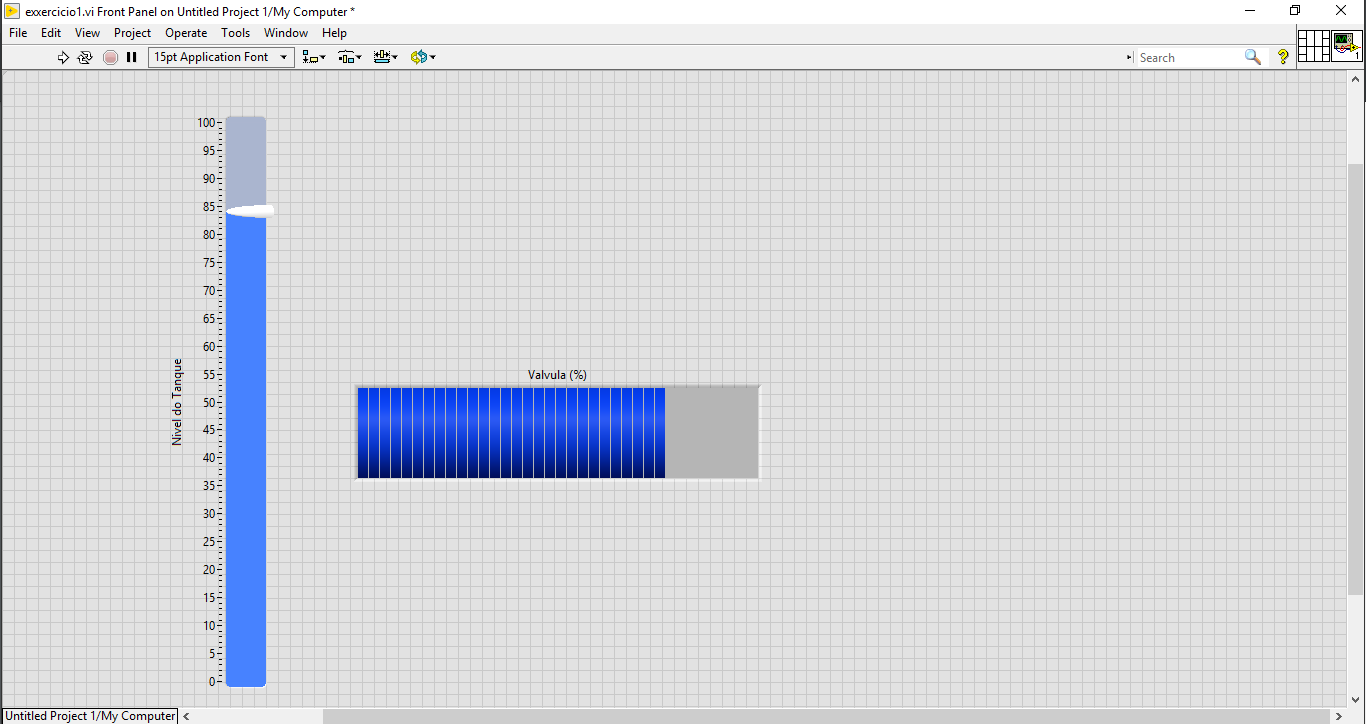


**Figura 30.** Aba de testes “Test System” do *Fuzzy Logic Designer*.

Depois de criar um sistema fuzzy no Fuzzy System Designer, é possível usar o FL Fuzzy Controller VI para implementar um controlador para o sistema fuzzy via diagrama de blocos do LabVIEW. Para isso, basta arrastar o “FL Fuzzy Controller.vi” e o “FL Load Fuzzy System.vi” para o diagrama e conectar o endereço do sistema criado com a extensão .fs no bloco de carregamento e realizar o controle após o processo. Na Figura 31 está apresentado um exemplo simples de utilização dos VIs de Fuzzy para o controle do tanque e na Figura 32 se encontra o painel de controle para o sistema.



**Figura 31.** Diagrama de blocos para o controle do tanque no LabVIEW.

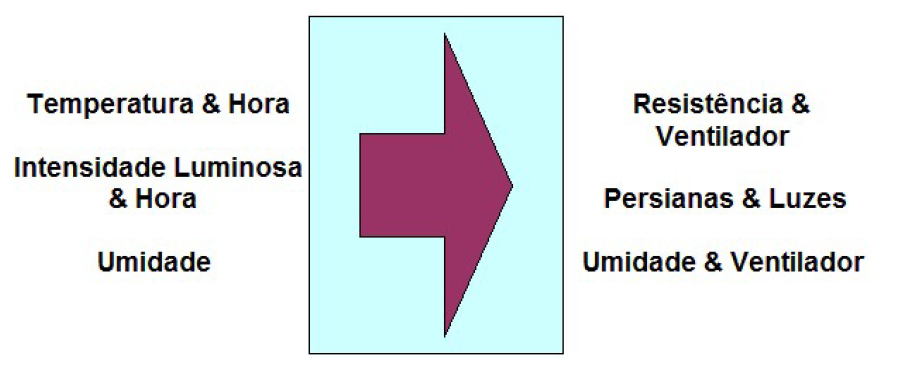


**Figura 32.** Painel de controle para o tanque no LabVIEW.

Além das opções já citadas, é possível também utilizar outros VIs para ajustar ou aprimorar o controlador fuzzy conforme sua necessidade pelo LabVIEW. O programa recomenda para que primeiramente seja criado o sistema fuzzy via Fuzzy Logic Designer e posteriormente alterar algo específico utilizando os VIs de fuzzy. Pode-se usar os VIs de pertinência por exemplo para modificar as funções de pertinência para conjuntos linguísticos em um sistema fuzzy ou pode usar os VIs de regras para modificar as regras no sistema.

*2.2 Exercício 2 : Controlador Fuzzy para Cultivo Hidropônico de Vegetais*

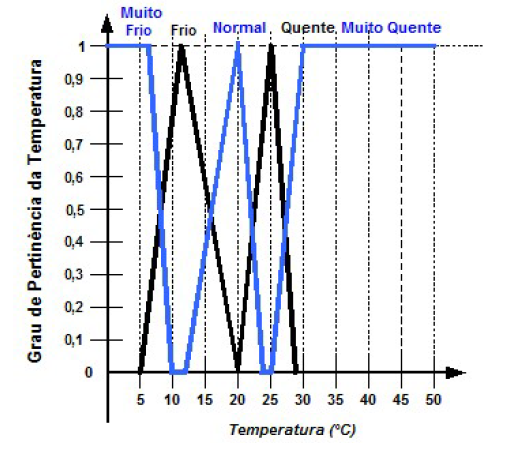
Neste exercício, o foco é projetar um sistema fuzzy para automatizar um cultivo hidropônico de vegetais. As variáveis de interesse deste sistema será considerar que para cada tipo de vegetal existem diferentes requerimentos de temperatura, umidade e intensidade luminosa. Além disso, dependendo da hora do dia é necessário que o controlador tenha quatro entradas correspondentes a cada uma das variáveis mencionadas pela Figura 33. Sendo assim, as variáveis de entrada e saída são: temperatura, umidade, intensidade luminosa e dependem do horário do dia, os atuadores de saída serão resistências, persianas, luzes, umidificador e ventilador, eles vão proporcionar as características ambientais requeridas para o cultivo.



**Figura 33.** Representação das variáveis de entrada e saída do cultivo

**Fonte –** Slides NA04C

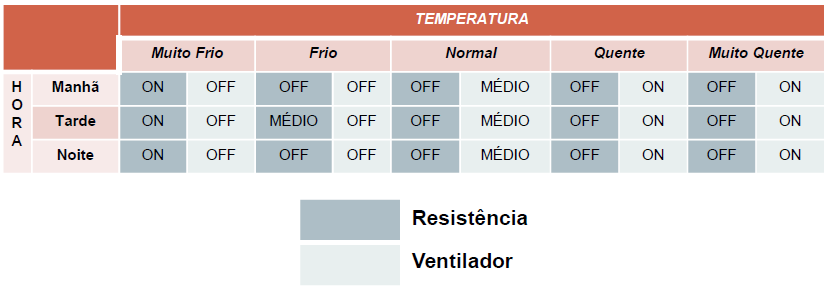
Para a variável de entrada da temperatura, medido pelo ambiente do local do cultivo, os valores serão definidos pelo exercício entre 0°C e 50°C. Serão criados 5 conjuntos linguísticos: muito frio, frio, normal, quente e muito quente. A Figura 34 demonstra as funções de pertinência da variável em função da tensão elétrica (obtida dos sensores e já na escala adequada).



**Figura 34.** Grau de pertinência da temperatura

**Fonte –** Slides NA04C

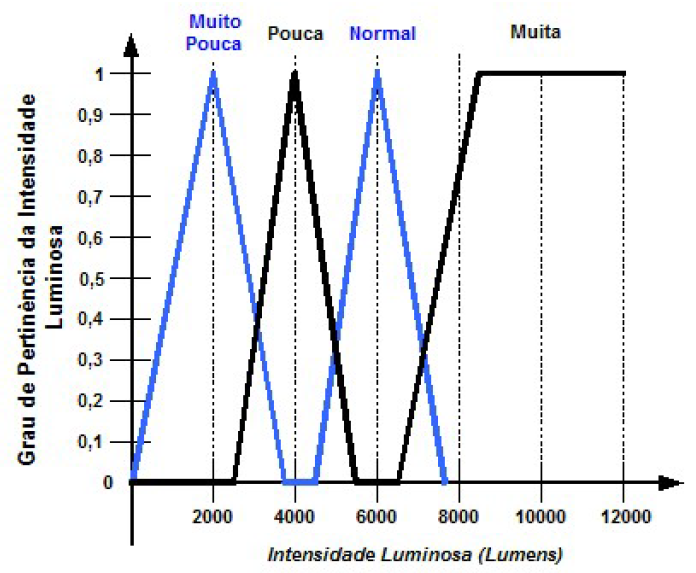
O controle da temperatura será determinado de acordo com os níveis de temperatura e o momento do dia e os atuadores para o controle serão a resistência e o ventilador. A Figura 35 apresenta a matriz de regras Fuzzy (FAM) para relacionar a variável de temperatura com o comportamento dos atuadores de saída do sistema, nesse caso a resistência e o ventilador.



**Figura 35.** Matriz de regras Fuzzy para controle de temperatura.

**Fonte –** Slides NA04C

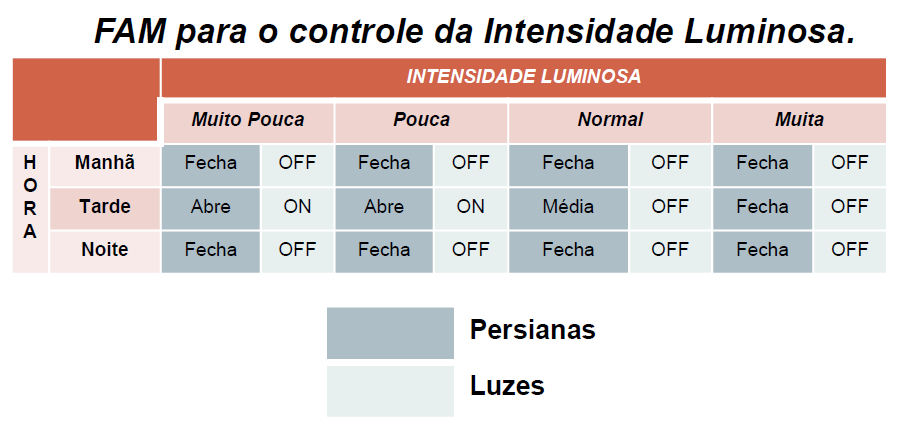
A variável de entrada de intensidade luminosa medida no local do cultivo hidropônico de alimentos possui uma faixa de 0 a 12000 lúmens. Serão criados 4 conjuntos linguísticos: muito pouca, pouca, normal e muita intensidade luminosa. A Figura 36 ilustra esses conjuntos linguísticos com suas respectivas faixas de intensidade luminosa e graus de pertinência.



**Figura 36.** Grau de pertinência da intensidade luminosa.

**Fonte –** Slides NA04C

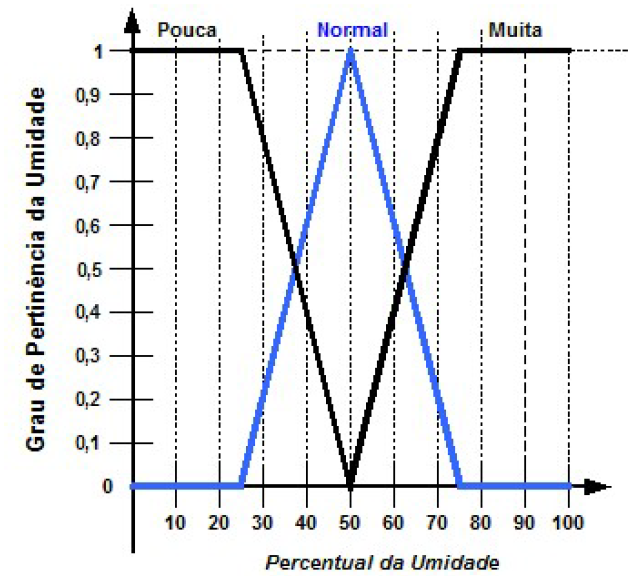
O controle da intensidade luminosa será determinado pelo nível de intensidade e o momento do dia e os atuadores para o controle serão as persianas e as luzes. A Figura 37 apresenta a matriz de regras Fuzzy (FAM) para relacionar o comportamento das luzes e da persiana de acordo com a intensidade luminosa e a hora do dia.



**Figura 37.** Matriz de regras Fuzzy para controle da intensidade luminosa..

**Fonte –** Slides NA04C

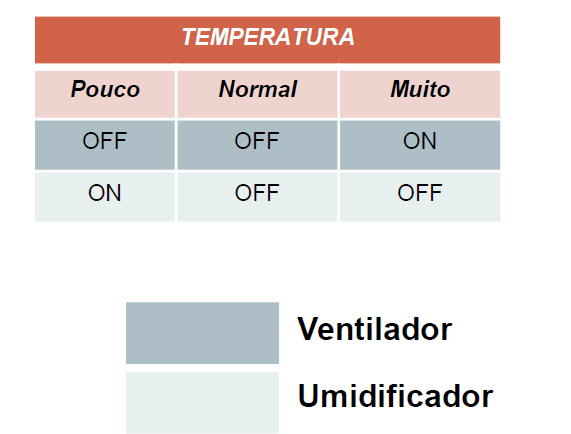
A variável Percentual de umidade presente no local do cultivo hidropônico de alimentos é definida a uma faixa de 0% a 100%. Serão definidos 3 conjuntos linguísticos: pouca, normal e muita. A Figura 38 ilustra esses conjuntos linguísticos com as faixas de cada percentual de umidade de cada um e seus respectivos graus de pertinência da umidade.



**Figura 38.** Grau de pertinência do percentual da umidade.

**Fonte –** Slides NA04C

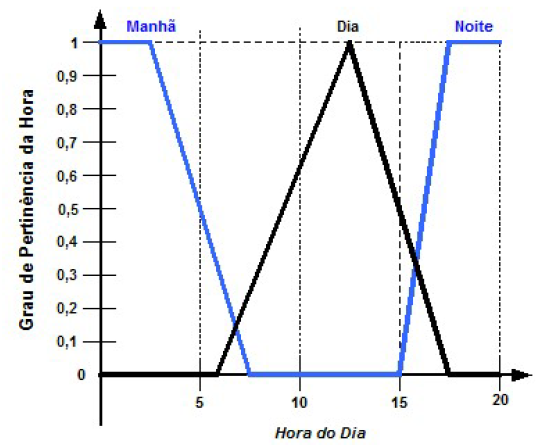
O controle de umidade será determinado pelo nível de umidade e os atuadores para o controle serão o umidificador e o ventilador. A Figura 39 apresenta a matriz de regras Fuzzy (FAM) para relacionar o comportamento do ventilador e do umidificador de acordo com o percentual de umidade.



**Figura 39.** Matriz de regras Fuzzy para controle do percentual de umidade.

**Fonte –** Slides NA04C

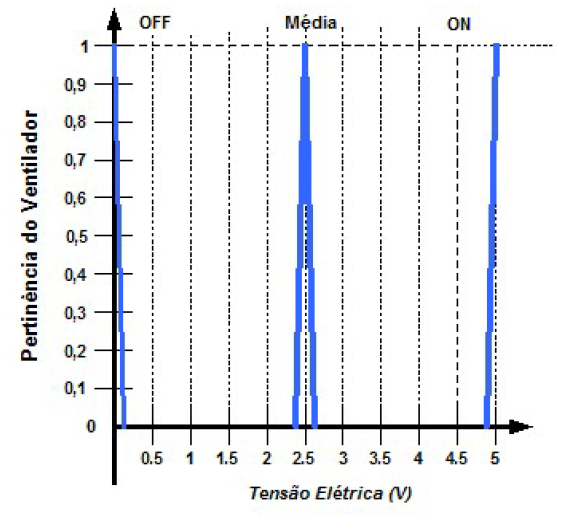
A variável hora do dia é definida com uma faixa entre 0 a 20 horas. Serão definidos 3 conjuntos linguísticos: manhã, tarde e noite. A Figura 40 ilustra esses conjuntos linguísticos com as faixas de hora do dia de cada um e seus respectivos graus de pertinência da hora.



**Figura 40.** Grau de pertinência da hora.

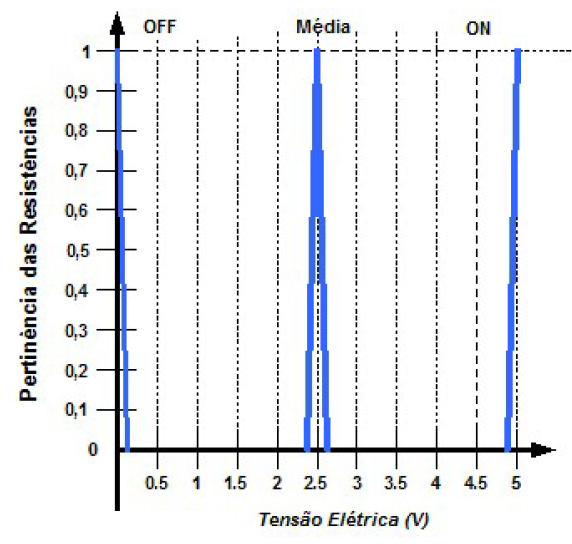
**Fonte –** Slides NA04C

Para a etapa de *defuzzyficação*, será empregado as Funções de Pertinência do tipo barra para simplificar o processo na obtenção do valor nítido de tensão elétrica para os atuadores, com uma faixa entre 0V a 5V. Nas Figuras 41, 42, 43, 44 e 45 estão apresentadas os graus de pertinência de tensão para cada um dos atuadores com seus 3 conjuntos linguísticos: ON, Média e OFF. Para encontrar o valor nítido será utilizado o método do Centróide, em que se calcula o centróide da área composta que representa o termo de saída fuzzy que é composto pela união de todas as contribuições das regras fuzzy determinadas.



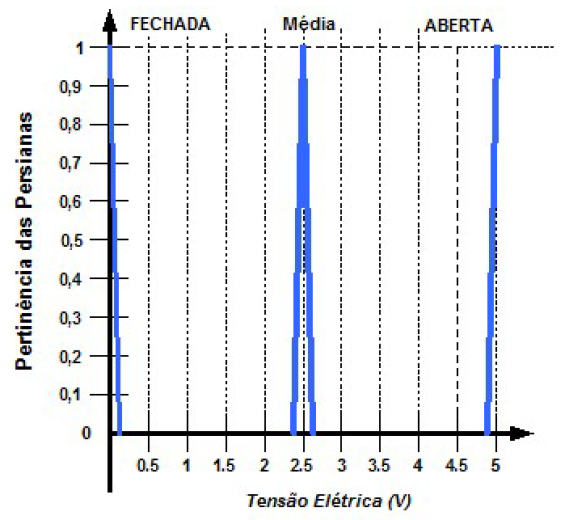
**Figura 41.** Grau de pertinência do ventilador.

**Fonte –** Slides NA04C



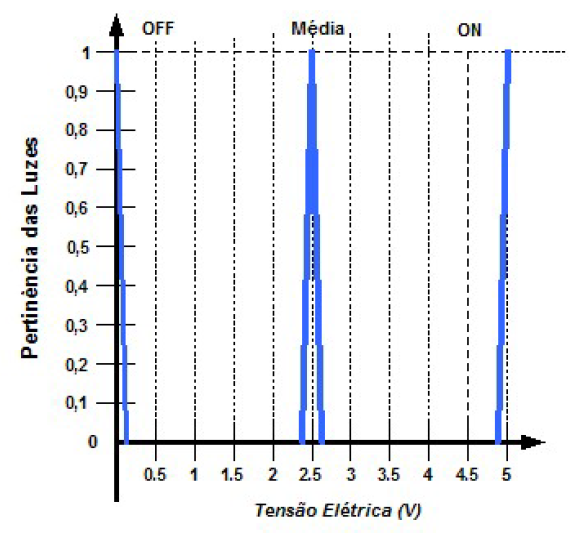
**Figura 42.** Grau de pertinência das resistências.

**Fonte –** Slides NA04C



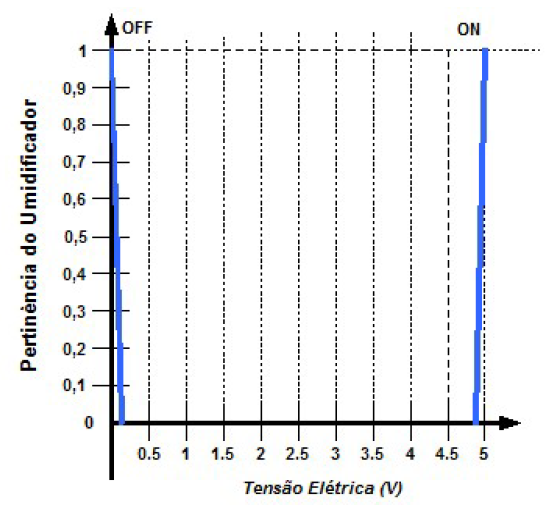
**Figura 43.** Grau de pertinência das persianas.

**Fonte –** Slides NA04C



**Figura 44.** Grau de pertinência das luzes.

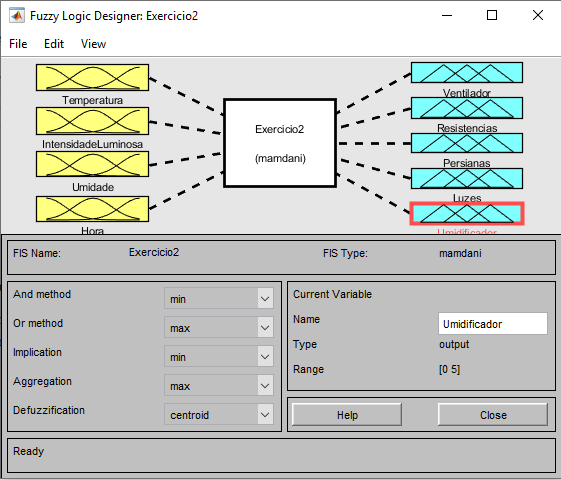
**Fonte –** Slides NA04C



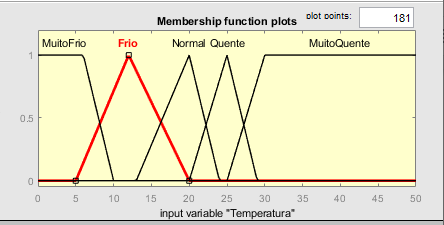
**Figura 45.** Grau de pertinência do umidificador.

**Fonte –** Slides NA04C

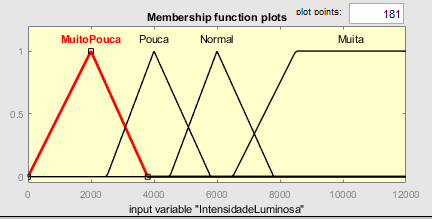
Para o projeto da controladora fuzzy, será utilizado a plataforma FIS do *software* Matlab© (MathWorks Inc., Massachusetts, EUA). Todas as variáveis de entrada e saída apresentadas anteriormente foram inseridas na plataforma, assim como a faixa de cada uma e seus respectivos graus de pertinência. Na Figura 46 é possível verificar o sistema criado pelo *software* e nas Figuras 47, 48, 49, 50, 51, 52, 53, 54 e 55 estão apresentadas os graus de pertinência inseridos em cada uma das variáveis de interesse.



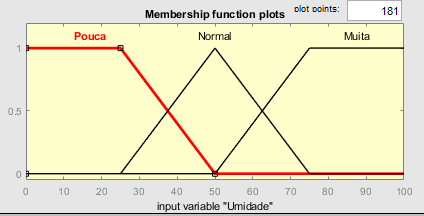
**Figura 46.** Interface FIS do *software* Matlab© (MathWorks Inc., Massachusetts, EUA).



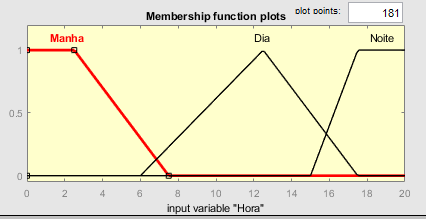
**Figura 47.** Grau de Pertinência da variável de entrada “Temperatura” pela interface FIS do *software* Matlab© (MathWorks Inc., Massachusetts, EUA).



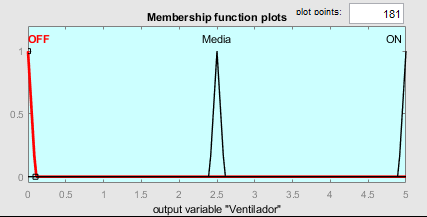
**Figura 48.** Grau de Pertinência da variável de entrada “Intensidade Luminosa” pela interface FIS do *software* Matlab© (MathWorks Inc., Massachusetts, EUA).



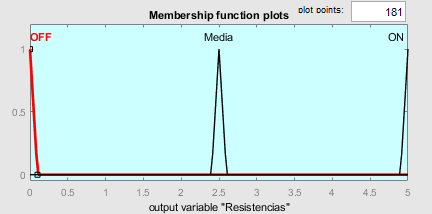
**Figura 49.** Grau de Pertinência da variável de entrada “Umidade” pela interface FIS do *software* Matlab© (MathWorks Inc., Massachusetts, EUA).



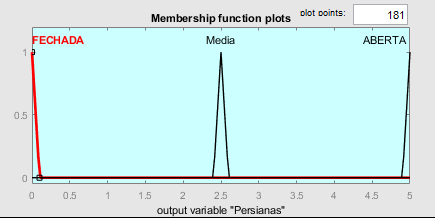
**Figura 50.** Grau de Pertinência da variável de entrada “Hora” pela interface FIS do *software* Matlab© (MathWorks Inc., Massachusetts, EUA).



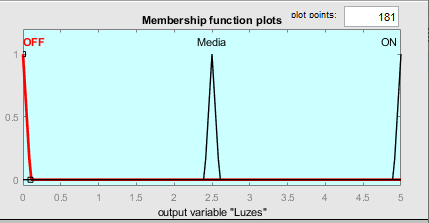
**Figura 51.** Grau de Pertinência da variável de saída “Ventilador” pela interface FIS do *software* Matlab© (MathWorks Inc., Massachusetts, EUA).



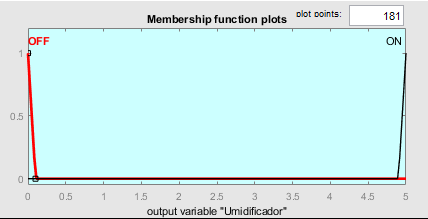
**Figura 52.** Grau de Pertinência da variável de saída “Resistências” pela interface FIS do *software* Matlab© (MathWorks Inc., Massachusetts, EUA).



**Figura 53.** Grau de Pertinência da variável de saída “Persianas” pela interface FIS do *software* Matlab© (MathWorks Inc., Massachusetts, EUA).



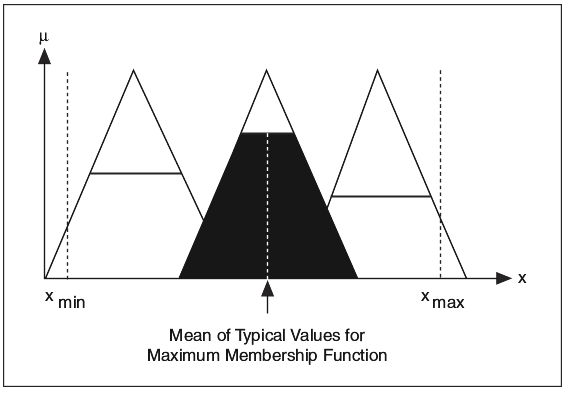
**Figura 54.** Grau de Pertinência da variável de saída “Luzes” pela interface FIS do *software* Matlab© (MathWorks Inc., Massachusetts, EUA).



**Figura 55.** Grau de Pertinência da variável de saída “Umidificador” pela interface FIS do *software* Matlab© (MathWorks Inc., Massachusetts, EUA).

Após realizar a análise com o software MatLab, será comparado com as equações teóricas do sistema.

Para concluir a análise, será alterado o método de *defuzzyficação* para o método MOM (*Mean of Maximum*) conforme a Figura 56 e analisar a resposta de saída do sistema. Ao usar o método de defuzzyficação *Mean of Maximum*, calcula-se o resultado mais plausível. O controlador fuzzy usa o valor típico do termo consequente da regra mais válida como o valor de saída nítido. Esse comportamento resulta em características de saída escalonadas. Em comparação com esse método, o método da centróide primeiro determina o valor numérico típico para cada função de associação em escala. O valor numérico típico é a média dos valores numéricos correspondentes ao grau de associação no qual a função de associação foi dimensionada.

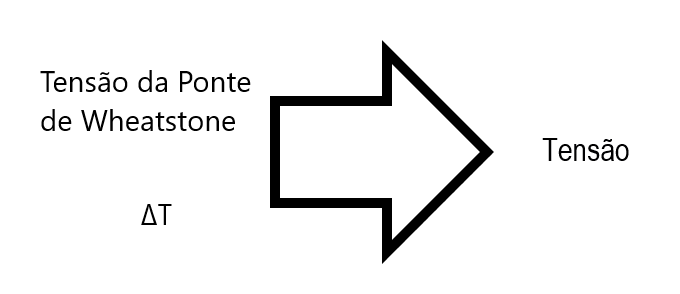


**Figura 56.** Método MOM (*Mean of Maximum*) de saída para o sistema.

**Fonte.** National Instruments.

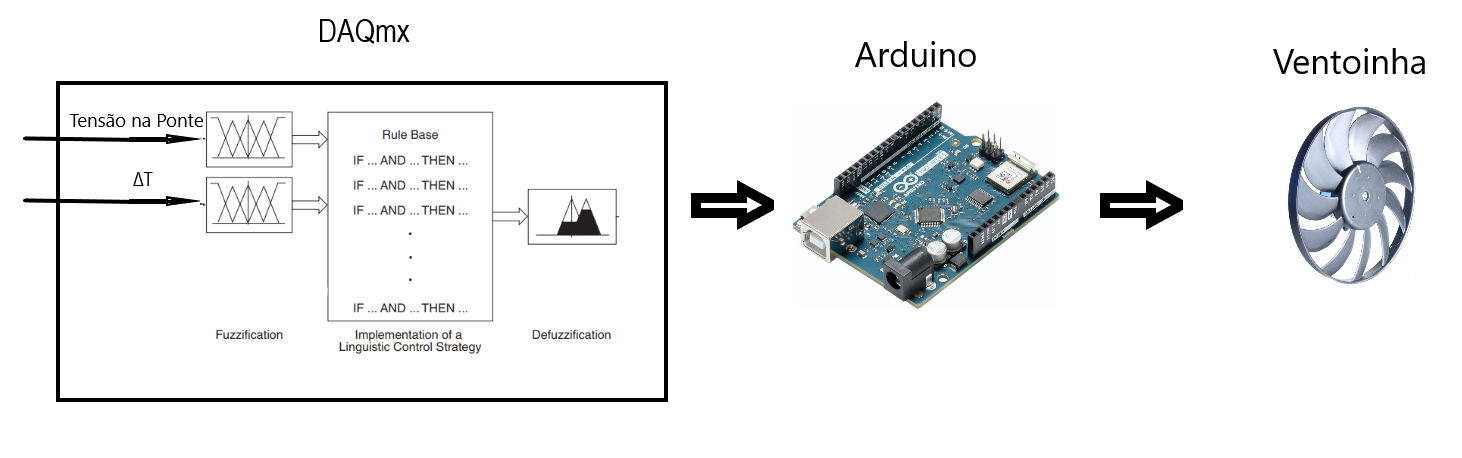
*2.3 Exercício 3*

Neste exercício, o foco é projetar um sistema fuzzy para controlar algum experimento realizado no laboratório. Será realizado, portanto, uma controladora fuzzy para controlar a temperatura da água no bécker com o sensor Pt100 em ponte, com as medidas já calculadas no laboratório, conforme a Figura 57.



**Figura 57.** Representação das variáveis de entrada e saída do controlador.

Para realizar a controladora fuzzy, será utilizado o DAQmx 6009 do laboratório juntamente com o software LabVIEW. Após a aquisição dos dados e da etapa de defuzzyficação, o DAQ enviará uma tensão na entrada analógica do Arduino que estará controlando uma ventoinha, resfriando a água do bécker. A Figura 58 demonstra o projeto do controlador elaborado para esse experimento.



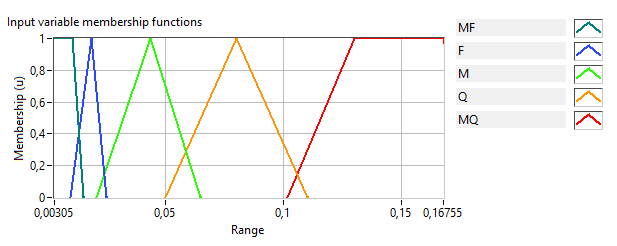
**Figura 58.** Esquemático completo do sistema de controle.

As variáveis de entrada serão a tensão de entrada e a variação da tensão de entrada em um certo período de tempo. O atuador de saída do sistema será uma tensão especificada no projeto. Para a variável de entrada de tensão, será utilizado os dados das medições já coletadas no laboratório de termometria para a ponte de Wheatstone do Pt100, conforme observado no Quadro X.

**Quadro 2.** Medidas realizadas em laboratório para a ponte de Wheatstone com o Pt100.

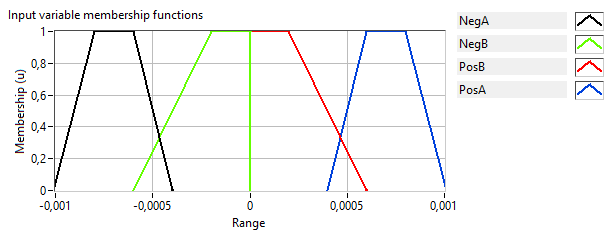
|  |  |
| --- | --- |
| Temperatura (°C) | Tensão (mV) |
|
| 16,5 | 3,05 |
| 18,5 | 8,8 |
| 20,5 | 14,85 |
| 22,5 | 20,6 |
| 24,5 | 26,55 |
| 26,5 | 32,1 |
| 28,5 | 38,05 |
| 30,5 | 43,8 |
| 32,5 | 49,6 |
| 34,5 | 55,3 |
| 36,5 | 61,3 |
| 38,5 | 67,3 |
| 40,5 | 72,7 |
| 42,5 | 78,4 |
| 44,5 | 84,35 |
| 46,5 | 90 |
| 48,5 | 95,4 |
| 50,5 | 101,55 |
| 52,5 | 107,05 |
| 54,5 | 112,55 |
| 56,5 | 118,35 |
| 58,5 | 123,8 |
| 60,5 | 129,35 |
| 62,5 | 134,65 |
| 64,5 | 140,65 |
| 66,5 | 146,15 |
| 68,5 | 151,5 |
| 70,5 | 157,15 |
| 72,5 | 162,25 |
| 74,5 | 167,55 |

Como os valores medidos estão nesse intervalo, os valores de entrada para a tensão serão definidos entre 3,05mV e 167,55mV. Serão criados 5 conjuntos linguísticos: muito frio (MF), frio (F), normal (M), quente (Q) e muito quente (MQ). A Figura 59 demonstra as funções de pertinência da variável em função da tensão elétrica (obtida dos sensores e já na escala adequada).



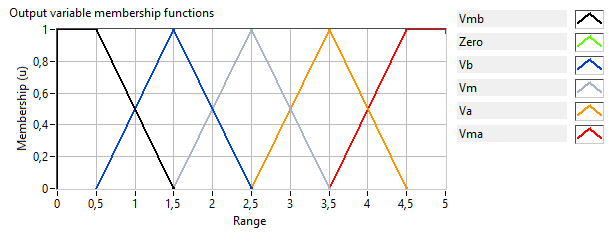
**Figura 59.** Grau de pertinência da tensão na ponte

A variável de entrada da diferença de tensão por um dado valor de tempo possui uma faixa de 0mV a 1mV. Serão criados 4 conjuntos linguísticos: variação negativa alta (NegA), variação negativa baixa (NegB), variação positiva baixa (PosB) e variação positiva alta (PosA). A Figura 60 ilustra esses conjuntos linguísticos com suas respectivas faixas de tensão e graus de pertinência.



**Figura 60.** Grau de pertinência da diferença de tensão.

Para a etapa de *defuzzyficação*, será empregado as Funções de Pertinência com uma faixa entre 0V a 5V. Serão criados 6 conjuntos linguísticos: sem tensão (Zero), tensão muito baixa (Vmb), tensão baixa (Vb), tensão média (Vm), tensão alta (Va) e tensão muito alta (Vma). A Figura 61 ilustra esses conjuntos linguísticos com suas respectivas faixas de tensão e graus de pertinência.



**Figura 61.** Grau de pertinência da saída de tensão do DAQmx.

O controle da temperatura para o nosso sistema será determinado de acordo com os níveis de tensão da ponte de Wheatstone e da diferença de tensão para cada 500 milissegundos. O DAQmx atuará com a saída de tensão em uma faixa de 0V a 5V para uma entrada analógica no arduino e posteriormente controlar a intensidade de uma ventoinha, de forma a resfriar a água do sistema. O Quadro 3 apresenta a matriz de regras Fuzzy (FAM) para relacionar a variável de temperatura com o comportamento dos atuadores de saída do sistema, nesse caso a resistência e o ventilador.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| V | MF | F | M | Q | MQ |
| ΔV |
| NegA | - | - | Vmb | Vm | Va |
| NegB | - | - | Vb | Va | Va |
| PosB | - | - | Va | Va | Vma |
| PosA | - | - | Vma | Vma | Vma |

**Quadro 3.** Matriz de Regras Fuzzy para o controlador de temperatura de água.

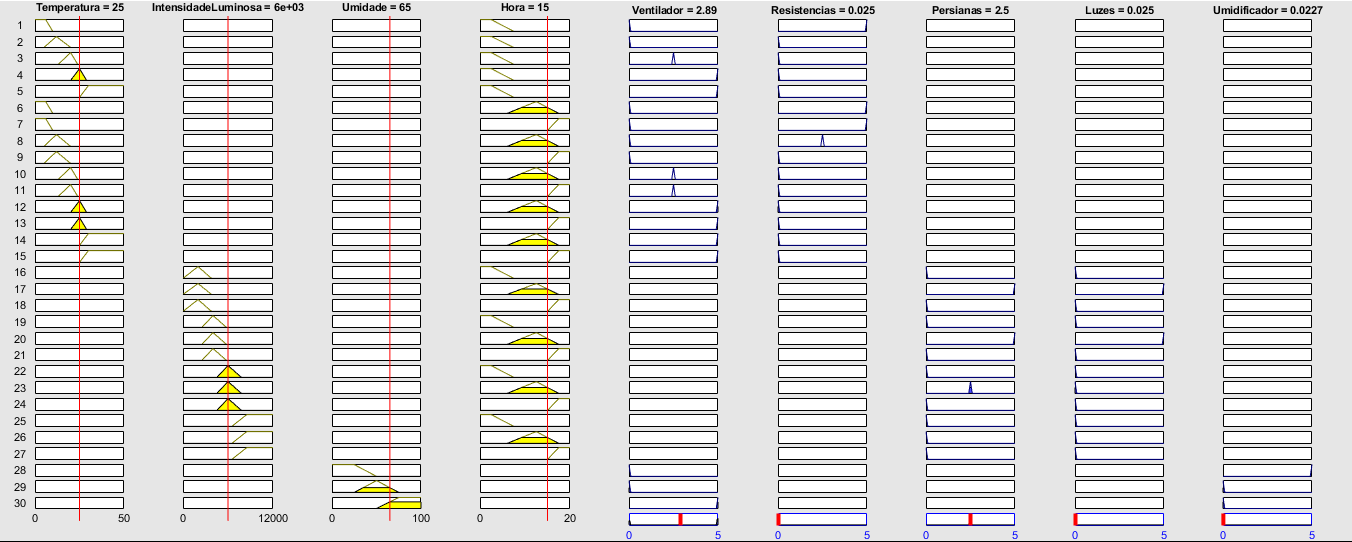
Para o projeto da controladora fuzzy, será utilizado a plataforma Fuzzy System Designer do *software* LabVIEW. Todas as variáveis de entrada e saída apresentadas anteriormente foram inseridas na plataforma, assim como a faixa de cada uma e seus respectivos graus de pertinência.

**3. Resultados e Discussões**

*3.1 Exercício 2 : Controlador Fuzzy para Cultivo Hidropônico de Vegetais*

Para a análise dos resultados do sistema, será utilizado o ponto com T=25°C, intensidade luminosa de 6000 lúmens, umidade em 65% e às 15 horas como exemplo. O método consiste em encontrar o ponto médio entre as áreas que convergem com o ponto avaliado. Observando na figura 11, sabe-se que a temperatura se encontra apenas no conjunto linguístico “Quente”, sendo os demais zero para aquele ponto em específico e possui grau de pertinência igual a 1. Já a hora do dia, 15 horas remete para o conjunto linguístico “Dia”, com grau de pertinência de aproximadamente 0,45. Pode-se concluir então pelos conjuntos de regras criado que o ventilador deverá ser ligado no ON e a resistência permanecerá desligada. Para 6000 lúmens observa-se que ele se encontra no conjunto linguístico normal, em seu pico, isso significa que as persianas serão ligadas no médio devido ao grau de pertinência da hora do dia e as luzes estão desligadas. Por fim, para a umidade em 50% ele deverá se encontrar no conjunto linguístico normal, portanto o umidificador fica desligado.

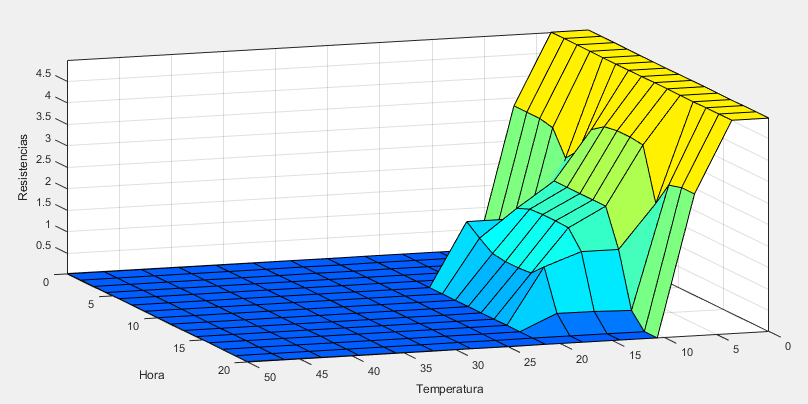
Para verificar o sistema e o exemplo citado acima, é possível observar pela Figura 62 o *Rule Viewer* do FIS, que demonstra o resultado do controlador.



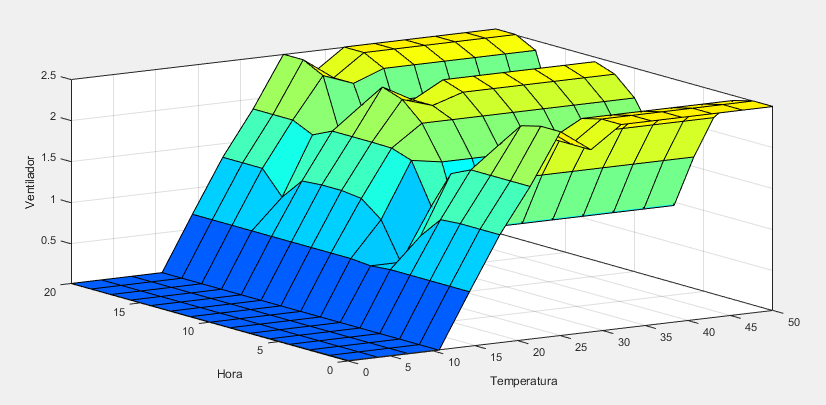
**Figura 62.** *Rule Viewer* para o exemplo realizado pela interface FIS do *software* Matlab© (MathWorks Inc., Massachusetts, EUA).

Pela Figura 62 nota-se que a controladora fuzzy criada pelo *software* Matlab© (MathWorks Inc., Massachusetts, EUA) faz sentido com as regras delimitadas pelo exercício, sabendo que todos dependem da hora exceto pelas 3 regras finais da umidade. Para a entrada de temperatura observa-se na saída resistência em ON para temperaturas mais baixas. Já para temperaturas altas observa-se o ventilador sendo acionado. Observa-se também as persianas e as luzes sendo ligadas apenas à tarde quando há baixa luminosidade. Para a entrada de umidade observa-se o umidificador sendo acionado apenas uma vez quando é baixa e o ventilador acionado uma vez quando é alta. Em suma, o comportamento de todos os atuadores condizem com as regras fuzzy criadas para o sistema.

Pelo Surface Viewer do FIS, é possível analisar a superfície que incluem as regras fuzzy adicionadas na controladora. Nas Figuras 63 e 64 é observado o comportamento da saída da resistência e do ventilador em relação à temperatura e a hora do dia.



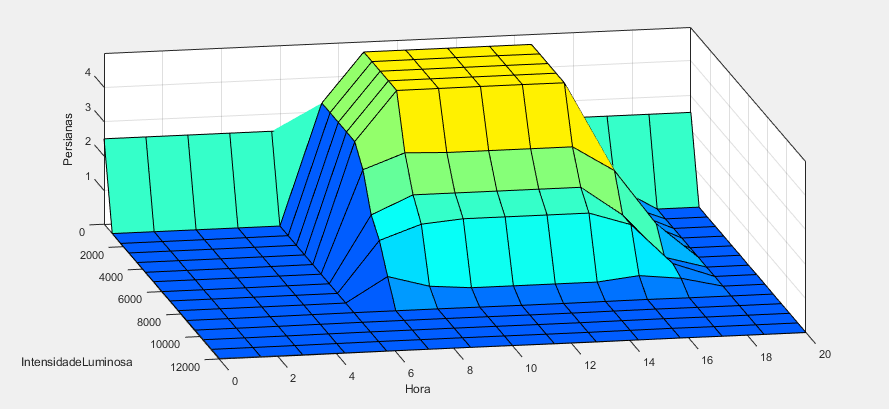
**Figura 63.** Superfície de saída da resistência realizado pela interface FIS do *software* Matlab© (MathWorks Inc., Massachusetts, EUA).



**Figura 64.** Superfície de saída do ventilador realizado pela interface FIS do *software* Matlab© (MathWorks Inc., Massachusetts, EUA).

Os resultados obtidos estão de acordo com o esperado, pois a resistência é acionada em temperaturas de muito frio que são até 10°C e é ligada também no médio de tarde em temperaturas frias. Já para o ventilador, ele é ligado somente na temperatura considerada “normal” no estado “médio”, entre 15°C e 25°C. Já para temperaturas quentes e muito quentes ele é ligado sempre, independente da hora do dia. Sendo assim, os gráficos apresentados estão condizentes com as regras da condicionadora fuzzy.

Nas Figuras 65 e 66 é observado o comportamento da saída das persianas e das luzes respectivamente em relação à intensidade luminosa e a hora do dia.



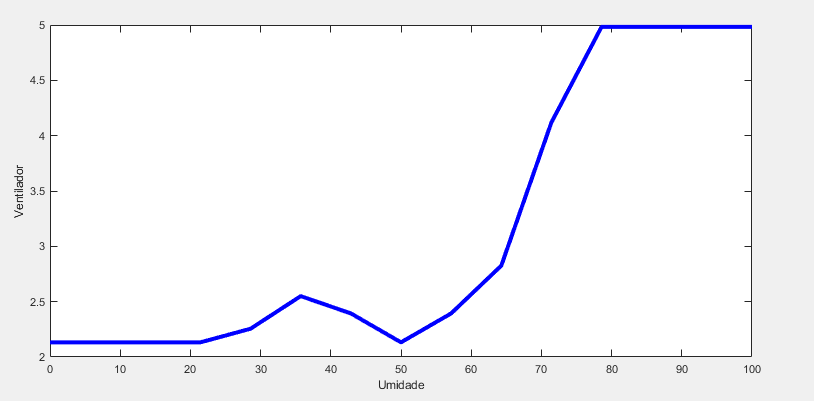
**Figura 65.** Superfície de saída das persianas realizado pela interface FIS do *software* Matlab© (MathWorks Inc., Massachusetts, EUA).



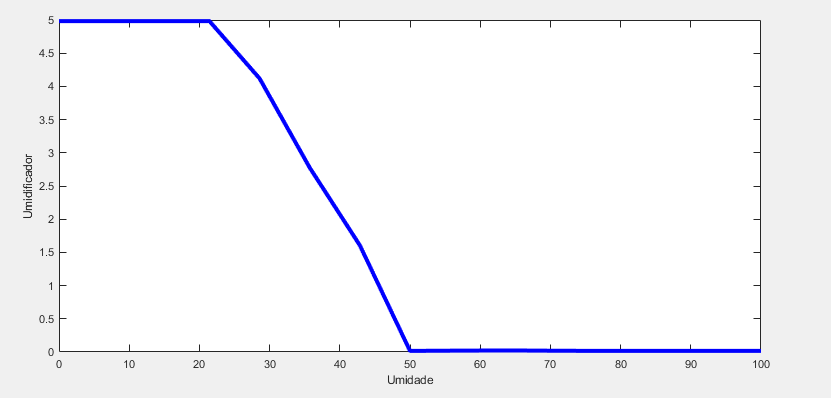
**Figura 66.** Superfície de saída das luzes realizado pela interface FIS do *software* Matlab© (MathWorks Inc., Massachusetts, EUA).

Avaliando os gráficos é possível verificar que o comportamento está conforme o esperado às regras impostas, já que as persianas só abrem no período da tarde e quando tem muito pouca ou pouca luminosidade. Ademais, é ligado no estado “médio” com média luminosidade à tarde. As luzes apresentam um comportamento parecido com excessão da média luminosidade onde só as persianas são ligadas.

Nas Figuras 67 e 68 é observado o comportamento do ventilador e do umidificador respectivamente em relação à umidade.



**Figura 67.** Superfície de saída da umidade realizado pela interface FIS do *software* Matlab© (MathWorks Inc., Massachusetts, EUA).



**Figura 68.** Superfície de saída do umidificador realizado pela interface FIS do *software* Matlab© (MathWorks Inc., Massachusetts, EUA).

Como o ventilador atua em outras variáveis da controladora, há um comportamento um pouco diferente em relação à umidade, pois ele deveria ser acionado somente ao chegar em 50%. Em contrapartida, o umidificador atua conforme o esperado, já que ele desliga quando chega aos 50%, sendo plausível conforme as regras criadas.

Agora para comparar com a análise teórica é utilizado o mesmo ponto com T=25°C, intensidade luminosa de 6000 lúmens, umidade em 65% e às 15 horas. Para esses valores temos as funções de pertinência descritas na Equação (1) até a Equação (4) e avaliadas pelas regras abaixo..

(1)

(2)

(3)

(4)

* If (Temperatura is Quente) and (Hora is Dia) then (Ventilador is ON)(Resistencias is OFF)  
  min[1;0,5] = 0,5;
* If (IntensidadeLuminosa is Normal) and (Hora is Dia) then (Persianas is Media)(Luzes is OFF)   
  min[1;0,5] = 0,5;
* If (Umidade is Normal) then (Ventilador is OFF)(Umidificador is OFF) = 0,4;
* If (Umidade is Muita) then (Ventilador is ON)(Umidificador is OFF) = 0,6

Considerando agora os valores de saída nas Equações (5), (6), (7), (8) e (9).

(5)

(6)

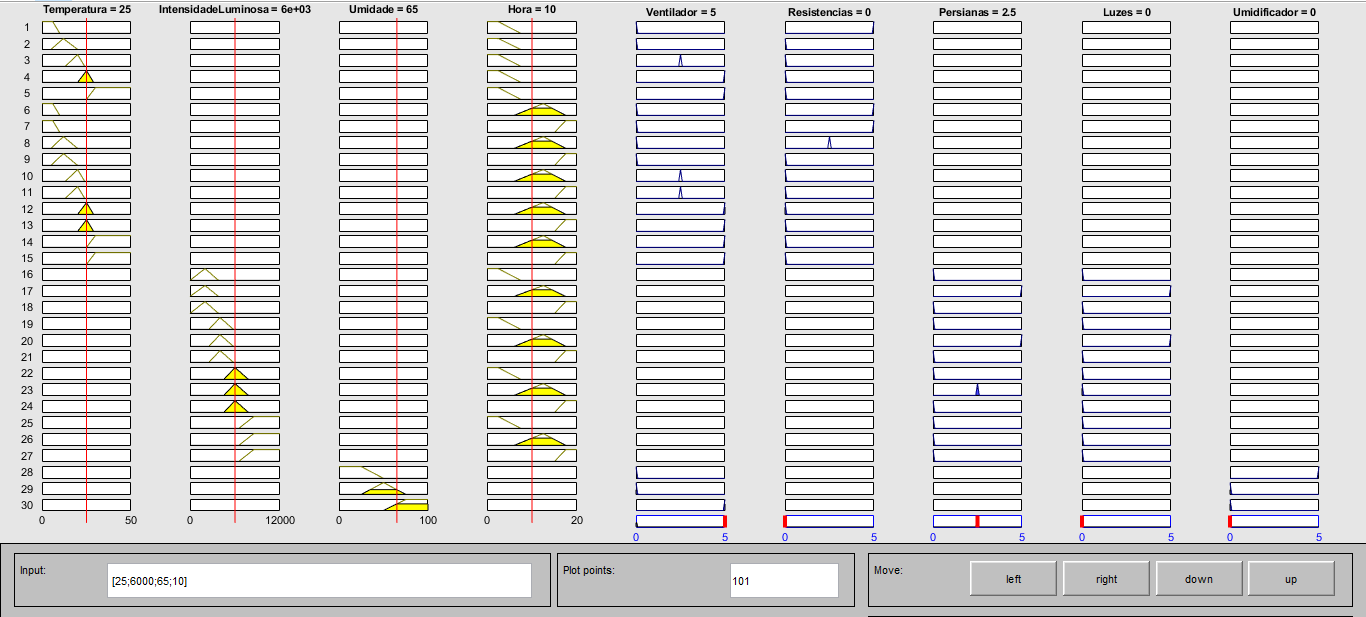
(7)

(8)

(9)

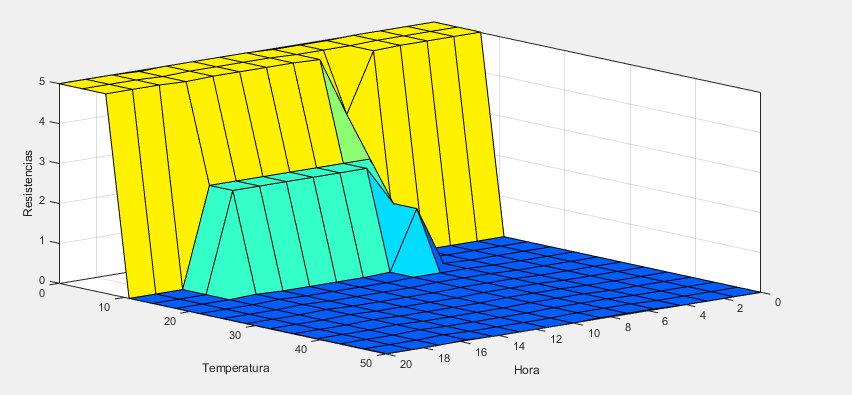
Obtém-se o valor dos atuadores de 3,6V no ventilador, 0V na resistência, 2,5V nas persianas, 0V nas luzes e 0V no umidificador. Comparando com as resposta analíticas temos uma aproximação muito boa ao ver que o ventilador foi acionado com o valor maior que a média e a persiana foi acionada na média. A imprecisão da analítica com a criada pelo software FIS pode ser ocasionada do fato do grau de pertinência ter sido definido ao observar o gráfico, com baixa resolução de medida.

Caso haja a alteração do método de defuzzyficação para o método MOM (*Mean of Maximum*), obtém-se um resultado diferente, como pode ser observado na Figura 69. Como o controlador fuzzy usa o valor da regra mais válida como o valor de saída, pode-se notar que os atuadores estão posicionados ou nos extremos ou no meio, conforme as regras criadas para os atuadores na saída. Não há um meio termo para eles pois o método MOM (*Mean of Maximum*) acaba gerando uma saída escalonada. Sendo assim, o ventilador não irá mais atuar com 2,91V e sim com 5V, seu valor total. Tanto os atuadores de resistência, luzes e umidificador irão zerar sua saída completamente, em contrapartida com o método da centróide, em que se possuía um valor de saída próximo de zero. Já a persiana, como já atuava anteriormente com seu valor no meio, continuará o mesmo valor.



**Figura 69.** *Rule Viewer* com método MOM de defuzzificação realizado pela interface FIS do *software* Matlab© (MathWorks Inc., Massachusetts, EUA).

Pelo Surface Viewer do FIS, é possível analisar a superfície que incluem as regras fuzzy adicionadas na controladora utilizando o método de defuzzyficação MOM. Nas Figuras 70, 71, 72, 73, 74 e 75 é observado o comportamento das saídas.



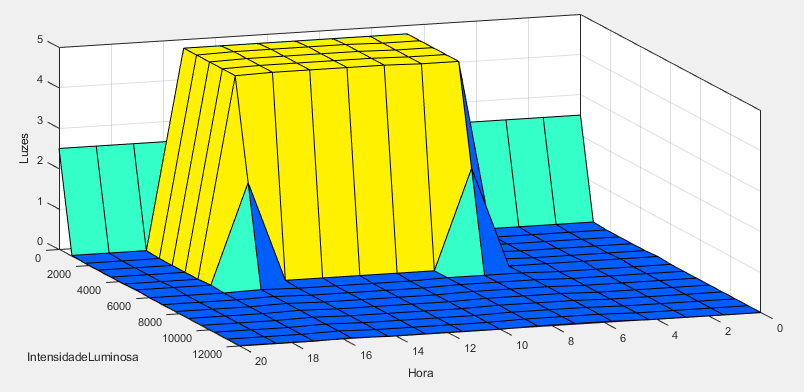
**Figura 70.** Superfície de saída da resistência realizado pela interface FIS do *software* Matlab© (MathWorks Inc., Massachusetts, EUA).



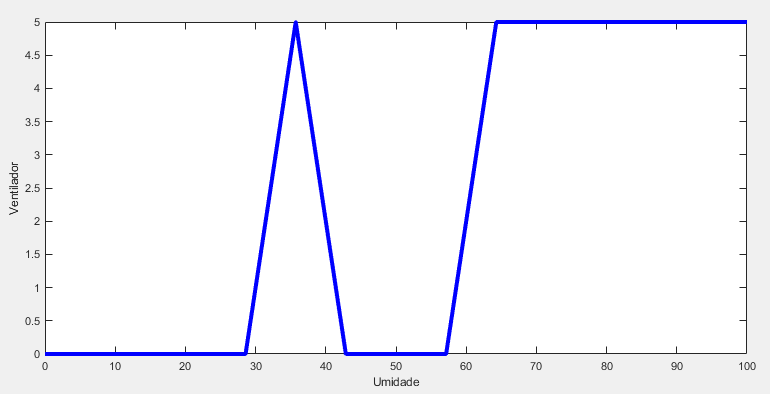
**Figura 71.** Superfície de saída do ventilador realizado pela interface FIS do *software* Matlab© (MathWorks Inc., Massachusetts, EUA).



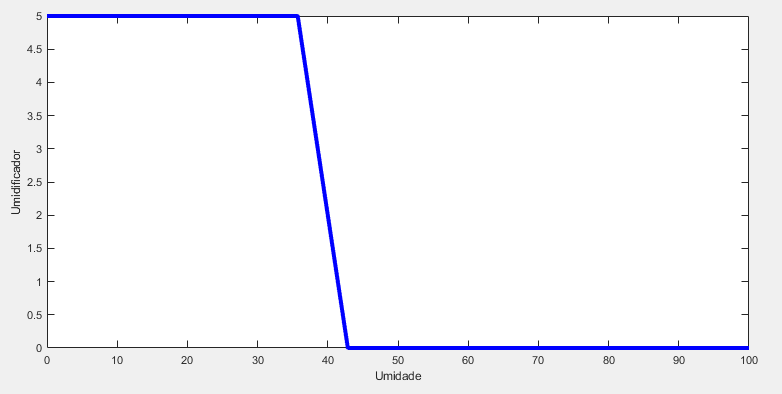
**Figura 72.** Superfície de saída das persianas realizado pela interface FIS do *software* Matlab© (MathWorks Inc., Massachusetts, EUA).



**Figura 73.** Superfície de saída das luzes realizado pela interface FIS do *software* Matlab© (MathWorks Inc., Massachusetts, EUA).



**Figura 74.** Superfície de saída da umidade realizado pela interface FIS do *software* Matlab© (MathWorks Inc., Massachusetts, EUA).



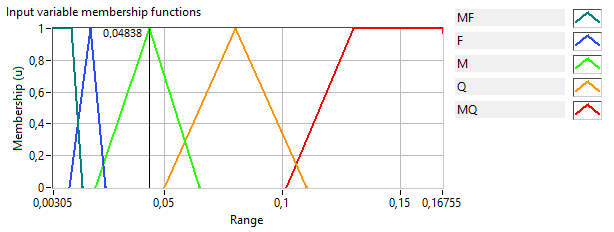
**Figura 75.** Superfície de saída do umidificador realizado pela interface FIS do *software* Matlab© (MathWorks Inc., Massachusetts, EUA).

Como pode ser observado e comparado com os gráficos de saída do método da centróide, o método MOM realmente possui uma resposta mais abrupta na sua saída. Para decidir qual método de *defuzzyficação* utilizar, cabe ao projetista verificar qual deve ser exatamente sua resposta no sistema para assim escolher o melhor método que aplica ao seu controlador.

*3.2 Exercício 3: Controladora Fuzzy para controle de temperatura no Pt100 na Ponte de Wheatstone*

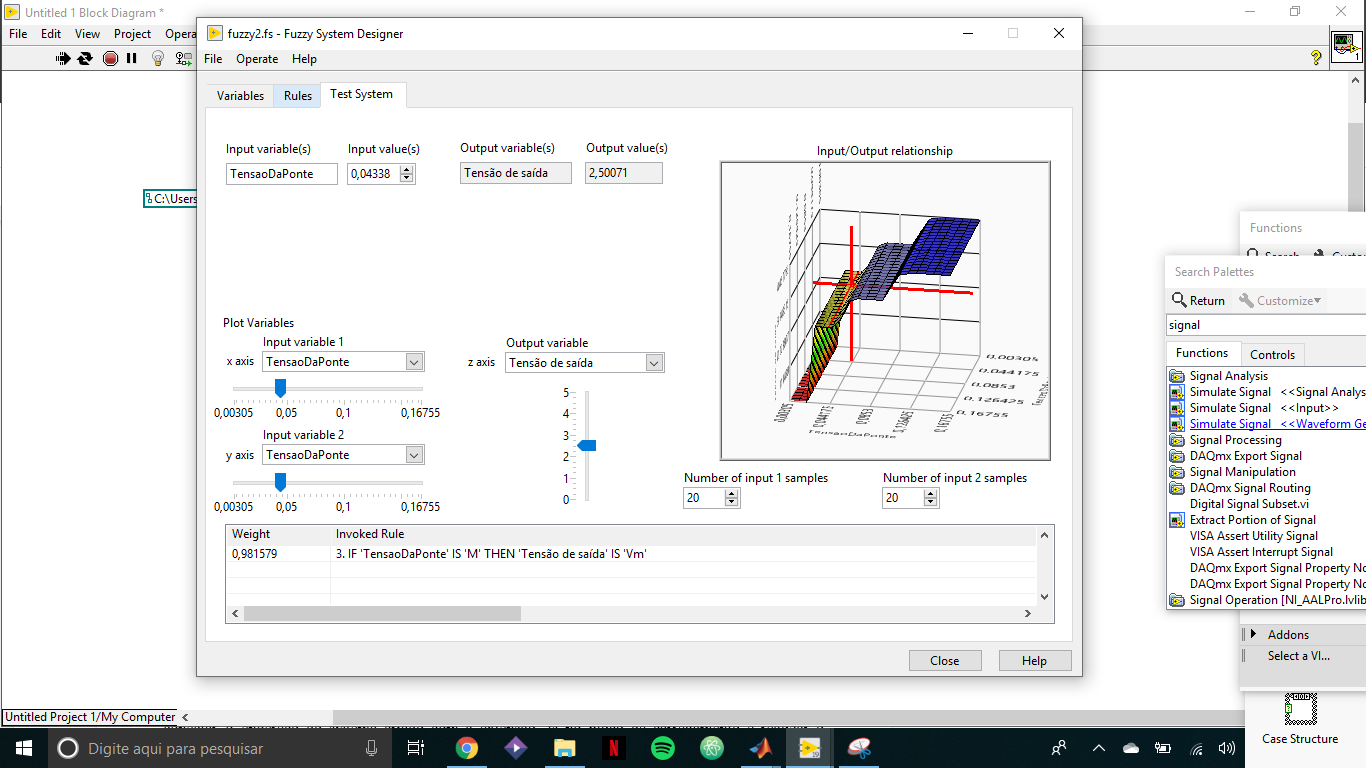
Para esta análise, o ponto que será analisado é com a Temperatura em 30°C, correspondendo à uma tensão de entrada de . Como no exercício 2, o método de análise será encontrar o ponto médio entre as duas áreas que convergem para o ponto avaliado.

Quando observa-se a figura 76, nota-se que o ponto de tensão indicado está apenas no conjunto linguístico M (Morno), com um grau de pertinência igual a 1, e nos demais sendo zero. Pode-se concluir então pelos conjuntos de regras criado que a Tensão de saída deverá ser ligado com um valor de 2,5V. O que faz sentido, pois o sistema se encontra com uma temperatura acima da setada, indicando que o sistema de resfriamento atua de forma significativa.



**Figura 76.** Análise da entrada tensão no ponto escolhido.

Para verificar o sistema e o exemplo citado acima, é possível observar na Figura 78 os resultados obtidos no Test System do Fuzzy System Designer. Nele, também apresenta-se as regras com seus graus de pertinência para os valores de .

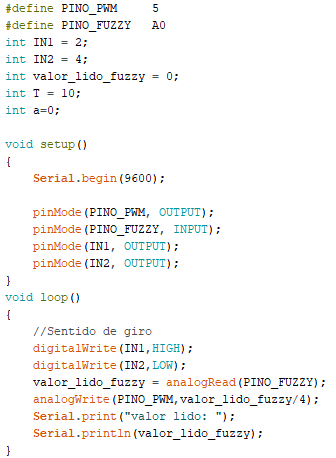


**Figura 78.** Análise das saídas em função das entradas escolhidas.

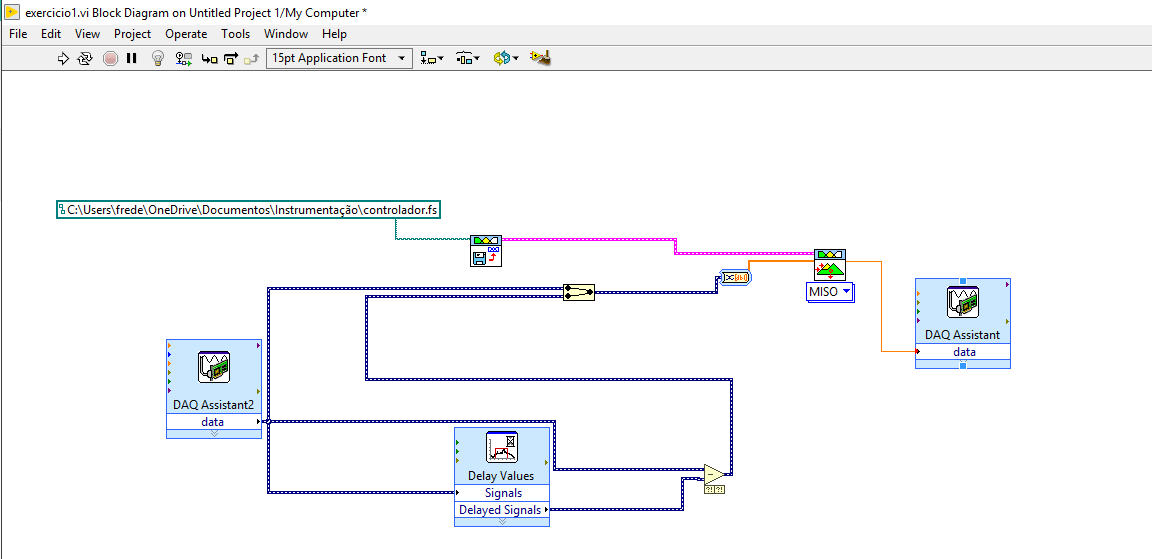
Pela Figura 78 nota-se que a controladora fuzzy criada faz sentido com as regras delimitadas no quadro 3, sabendo que a saída sempre depende de como está a tensão em dado momento (o que representa uma temperatura).

Pelas regras criadas, observa-se sempre que quando a tensão de entrada está muito elevada (temperatura alta), a saída do sistema será com uma tensão alta, proporcionando um resfriamento mais rápido. No entanto, quando a temperatura está perto da temperatura de interesse, a tensão aplicada na ventoinha será pequena, chegando gradativamente na temperatura. Dessa mesma forma, quando a temperatura está abaixo da de interesse, ocasionando uma tensão de entrada abaixo da de interesse, a tensão aplicada na ventoinha será nula, uma vez que o sistema de saída apenas resfria, e assim não fazendo sentido mantê-lo ligado.

A figura 79 mostra a rotina desenvolvida para o microcontrolador. Ela é feita para o controle PWM do sinal de saída CC para o motor, que consequentemente controla sua velocidade linearmente com o sinal aplicado. O microcontrolador recebe o valor do DAQ em sua porta ADC, transforma para uma string de 10 bits que interpreta o sinal de tensão e depois aplica proporcionalmente na saída PWM. Para o acionamento da ventoinha, foi usado um braço de uma ponte H, capaz de fornecer a corrente necessária para o motor. O programa desenvolvido para rodar a rotina no labview está exposto na Figura 80.



**Figura 79.** Script de programação do microcontrolador.



**Figura 78.** Rotina Fuzzy no LabWiew..

**4. Conclusões**

O projeto de controladores utilizando o FIS toolkit do MATLAB permite o desenvolvimento do projeto de controladores fuzzy de forma ágil, sendo possível configurar de forma intuitiva características de qualquer uma das partes constituintes de um sistema fuzzy: variáveis linguísticas, funções de pertinência e regras. Ademais, o ambiente de criação de uma controladora fuzzy no LabVIEW permite também criar sistemas inteligentes de forma muito simples, sendo possível a criação do projeto do grupo com alguns blocos e utilização do DAQmx.

Conclui-se também que pode ser projetados controladores fuzzy utilizando-se outras abordagens, como no próprio MatLab através de linha de comando ou através do ambiente de simulação SIMULINK. Os resultados obtidos com as variáveis de saída após a aplicação da metodologia utilizada por esses sistemas são muito mais ágeis aos valores obtidos algebricamente.

**Referências Bibliográficas**

1. Balbinot, A; Slides na04a, 2019.

2. Balbinot, A; Slides na04b, 2019.

3. Balbinot, A; Slides na04c, 2019.

4. National Instruments (org.). NI USB-6008/6009 User Guide. [S. l.], 2015. pdf.

5. LabVIEW PID and Fuzzy Logic Toolkit User Manual [S. l.], 2009. Disponível em: <http://www.ni.com/pdf/manuals/372192d.pdf>. Acesso em: 20 out. 2019.

6. Simulate Fuzzy Inference Systems in Simulink [S. l.], 2019. Disponível em: <https://www.mathworks.com/help/fuzzy/simulate-fuzzy-inference-systems-in-simulink.html>. Acesso em: 20 out. 2019.

7. Build Fuzzy Systems at the Command Line [S. l.], 2019. Disponível em: <https://www.mathworks.com/help/fuzzy/working-from-the-command-line.html>. Acesso em: 30 out. 2019.

8. Build Fuzzy Systems Using Custom Functions [S. l.], 2019. Disponível em: <https://www.mathworks.com/help/fuzzy/building-fuzzy-inference-systems-using-custom-functions.html>. Acesso em: 30 out. 2019.

©2019 Filipe Wermann, Frederico May de Liz e Natália Rovaris; disciplina de Instrumentação A, UFRGS, DELET, RS, Brasil.