Universidade Federal do Pará
Instituto Tecnológico
Programa de Pós-Graduação em Engenharia Elétrica

SINTONIA DE CONTROLADOR FUZZY POR ALGORITMO GENÉTICO EM SISTEMA DE NÍVEL DE LÍQUIDOS

JOSÉ FLÁVIO BARBOSA DE ANDRADE

DM: 07/2014

UFPA / ITEC / PPGEE
Campus Universitário do Guamá
Belém-Pará-Brasil
2014

Universidade Federal do Pará
Instituto Tecnológico
Programa de Pós-Graduação em Engenharia Elétrica

JOSÉ FLÁVIO BARBOSA DE ANDRADE

SINTONIA DE CONTROLADOR FUZZY POR ALGORITMO GENÉTICO EM SISTEMA DE NÍVEL DE LÍQUIDOS

DM: 07/2014

UFPA / ITEC / PPGEE
Campus Universitário do Guamá
Belém-Pará-Brasil
2014

Universidade Federal do Pará
Instituto Tecnológico
Programa de Pós-Graduação em Engenharia Elétrica

JOSÉ FLÁVIO BARBOSA DE ANDRADE

SINTONIA DE CONTROLADOR FUZZY POR ALGORITMO GENÉTICO EM SISTEMA DE NÍVEL DE LÍQUIDOS

Dissertação de Mestrado submetida à Banca Examinadora do Programa de Pós-Graduação em Engenharia Elétrica como parte dos requisitos para obtenção do título de Mestre em Engenharia Elétrica. Área de concentração: **Sistema de Energia Elétrica**.

UFPA / ITEC / PPGEE
Campus Universitário do Guamá
Belém-Pará-Brasil
2014

Dados Internacionais de Catalogação-na-Publicação (CIP)

Andrade, José Flávio Barbosa de, 1983-
Sintonia de controlador fuzzy por algoritmo genético em sistema de nível de liquidos / José Flávio Barbosa de Andrade. - 2014.
Orientador: José Augusto Lima Barreiros.
Dissertação (Mestrado) - Universidade Federal do Pará, Instituto de Tecnologia, Programa de Pós-Graduação em Engenharia Elétrica, Belém, 2014.
1. Algoritmos genéticos. 2. controladores programáveis. 3. Lógica fuzzy. I. Título.
CDD 006.31. ed. 22

UNIVERSIDADE FEDERAL DO PARÁ
INSTITUTO DE TECNOLOGIA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

"SINTONIA DE CONTROLADOR FUZZY POR ALGORITMO GENÉTICO EM SISTEMA DE NÍVEL DE LÍQUIDO"

AUTOR: **JOSÉ FLÁVIO BARBOSA DE ANDRADE**

DISSERTAÇÃO DE MESTRADO SUBMETIDA À BANCA EXAMINADORA APROVADA PELO COLEGIADO DO PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA, SENDO JULGADA ADEQUADA PARA A OBTENÇÃO DO GRAU DE MESTRE EM ENGENHARIA ELÉTRICA NA ÁREA DE SISTEMA DE ENERGIA ELÉTRICA.

APROVADA EM: 31 / 03 / 2014

BANCA EXAMINADORA:

Prof. Dr. José Augusto Lima Barreiros
(Orientador – PPGEE/UFPA)

Prof. Dr. Walter Barra Júnior
(Avaliador Interno – PPGEE/UFPA)

Prof. Dr. André Maurício Damasceno Ferreira
(Avaliador Externo - IFPA)

VISTO:

Prof. Dr. Evaldo Gonçalves Pelaes
(Coordenador do PPGEE/ITEC/UFPA)

AGRADECIMENTOS

Agradeço a Deus por me dar força para continuar sempre.

Agradeço à minha esposa e minha família que são minha base para tudo.

Agradeço à Universidade Federal do Pará (UFPA) pela oportunidade de realizar este trabalho.

Ao meu orientador, Prof. Dr. José Augusto Lima Barreiros, pela oportunidade e apoio dado durante este trabalho de pesquisa.

Agradeço à Fundação de Amparo à Pesquisa do Estado do Pará (FAPESPA) pelo suporte financeiro durante o período de conclusão deste trabalho.

Dedico este trabalho à D. Vitória, Giselle Souza, Jerônima Moraes e amigos.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

RESUMO

O presente trabalho demonstra a aplicação de um Algoritmo Genético com o intuito de projetar um controlador *Fuzzy* MISO, através da sintonia de seus parâmetros, em um processo experimental de nivelamento de líquido em um tanque, cuja dinâmica apresenta características não-lineares. Para o projeto e sintonia do controlador, foi utilizado o suporte do software Matlab, e seus pacotes Simulink e *Global Optimization Toolbox*. O Controlador *Fuzzy* ora projetado teve seu desempenho avaliado através de ensaios em tempo real em um Sistema de Nível de Liquido.

Palavras Chaves: Algoritmo genético, Controlador Fuzzy, Controle de Nível

ABSTRACT

This work presents the application of a Genetic Algorithm in order to design a MISO Fuzzy Controller by tuning its parameters in an experimental process of leveling liquid in a tank whose dynamics has nonlinear characteristics. In the Fuzzy Controller design and tuning it was used the software Matlab and their packages Simulink and Global Optimization Toolbox. The Fuzzy Controller designed had its performance evaluated by a real time test in a Liquid Level System.

Key Words: Genetic Algorithm, Fuzzy Controller, Level Control.

CAPÍTULO 1
INTRODUÇÃO

1. INTRODUÇÃO

O Sistema *Fuzzy* atualmente desempenha um papel muito importante no projeto de sistemas de controle moderno, por permitir a interpretação da expressão ambígua e imprecisa das decisões e do pensamento humano através de um conjunto regras que relacionam as reações da máquina ao conhecimento do especialista humano. Além disso, os baixos requisitos computacionais tornam esse tipo de controle ideal para aplicações em tempo real.

Um sistema *fuzzy* consiste de uma lista de regras baseadas em condicionais se-então que se relacionam com conjuntos *fuzzy*, de entrada e saída, cuja precisão entre eles é interpolada. Normalmente, a seleção das regras se-então é feita através de sintonias heurísticas que expressam o conhecimento especializado das estratégias apropriadas para o problema. Entretanto, é difícil para especialistas humanos examinar todos os dados de entrada e saída, principalmente para sistemas complexos, onde há muitas entradas e muitas saídas, a fim para encontrar regras apropriadas para o sistema *fuzzy*. Para lidar com essa dificuldade, várias abordagens de geração de bases de regras *fuzzy* foram propostas em ZADEH (1965) e WANG (1997).

Recentemente, o desenvolvimento de abordagens evolucionárias permitiu outra possibilidade de projeto de controladores que contribuem para facilitar a construção de base de regras de projetos *Fuzzy* e de outros projetos. A principal abordagem é a que utiliza os algoritmos genéticos (AG) que representa procedimentos probabilísticos de busca de soluções inspiradas nos mecanismos de seleção natural e são boas alternativas em situações onde as informações sobre o espaço de soluções é limitado ou desconhecido.

Sistemas *Fuzzy* do tipo Takagi-Sugeno (TS) e Mandani foram projetados visando a modelagem ou controle utilizando variações do algoritmo genético originalmente proposto por Holland no início dos anos 60, sendo as primeiras aplicações implementadas apenas no início da década de 90, e que, até hoje, vem sendo experimentadas com série de adaptações (JAIN e JAIN, 1997). As pesquisas estão sendo desenvolvidas com o intuito de aperfeiçoar os parâmetros do sistema

Fuzzy com cada vez menos interação humana. Aos poucos foram analisados também outros critérios qualitativos para os sistemas obtidos, tais como interpretabilidade e simplicidade.

Dessa forma, o presente trabalho visa a aplicação de um algoritmo evolutivo, mais precisamente o algoritmo genético, na tentativa de projetar um controlador *Fuzzy* para um processo experimental de nivelamento de líquido em tanque cuja dinâmica apresenta características não lineares. Para isso, foi necessária a utilização do software MATLAB, com o apoio dos pacotes Simulink e Global Optimization Toolbox. O controlador *fuzzy* foi testado através de ensaios em tempo real em um Sistema de Nível de Líquido.

As principais contribuições desta pesquisa residem na proposta de representações cromossômicas baseadas na simetria de funções de pertinência, na otimização baseada na dinâmica de um sistema modelo auxiliar, na elaboração de uma função de *fitness* simplificada e na alteração do operador de crossover aritmético respeitando a representação cromossômica imposta às funções de cada gene.

1.1 OBJETIVO DO TRABALHO

A motivação deste trabalho é estudar a possibilidade de usar o algoritmo genético para sintonizar parâmetros de um controlador baseado em lógica *fuzzy* de forma que possa ser usado como um método de controle a ser aplicado em um processo não linear de nível de tanques. Para tal foi escolhido o modelo de uma planta de tanque simples que se comporta como um sistema de primeira ordem.

1.2 METODOLOGIA

O processo escolhido para estudo foi o processo de nível de tanque simples, modelado matematicamente através da física envolvida no processo (AGUIRRE, 2004), principalmente pelo seu aspecto não linear. Para esse sistema já foram estudadas em diversas oportunidade e abordagens incluindo os aspectos

multivariáveis e complexidade dinâmica, com interação de várias variáveis manipuladas. Entretanto neste trabalho o foco principal foi o desenvolvimento do controlador, cujos parâmetros são encontrados por técnica baseada em algoritmos evolucionários.

O primeiro passo para esta metodologia é estudar e conhecer o processo a ser controlado. No caso deste trabalho, usou-se principalmente a técnica de modelagem baseada na física do processo realizada em AGUIRRE (2004) e as bem sucedidas implementações em tempo real realizadas em BERNARDES *et al* (2006) e ANDRADE (2008).

Procedeu-se com a linearização e discretização do modelo. O próximo passo é o desenvolvimento do programa de apoio baseado em software MATLAB, com as ferramentas Simulink e Global Optimization Toolbox, para a aplicação do algoritmo genético no modelo *fuzzy*. Este programa configura e executa o algoritmo genético recursivamente, através de um simulador do sistema baseado em Simulink, e aplica seu resultado nos parâmetros do controlador *fuzzy*. A simulação pode parar até atingir um desempenho desejado ou atingir um número limite de gerações de indivíduos de forma que se tenha o desempenho mais próximo do desejado. O desempenho ótimo é obtido através da comparação com um modelo cuja dinâmica serviu de base para planta. Após a simulação, o terceiro passo constitui uma comparação entre os resultados simulados e resultados reais aplicados na planta do sistema de tanque do sistema *fuzzy* ora projetado.

1.3 ORGANIZAÇÃO DO TEXTO

Esta dissertação está organizada da seguinte forma. Após esta introdução, o Capítulo 2 apresenta um breve histórico e apresenta uma introdução simplificada sobre os Sistemas *Fuzzy*. Reserva-se o Capítulo 3 para a apresentação detalhada do protótipo em que foi realizado o ensaio em tempo real, enfatizando a modelagem matemática baseado na física do processo. O Capítulo 4, o principal da dissertação, é dedicado à apresentação dos algoritmos genéticos, seus princípios, aplicações e, principalmente como se funde com o Sistema *Fuzzy*, formando os sistemas

denominados genético-*fuzzy*. Após uma revisão dos conceitos básicos da ferramenta, detalha-se cada etapa do AG, primeiramente de forma geral e então a implementação específica para este trabalho, seguindo a ordem: representação cromossômica, avaliação dos indivíduos que codificam os modelos, operador de seleção e operadores de reprodução (crossover e mutação).

O Capítulo 5 se dedica a mostrar como foram realizadas as simulações e os ensaios em tempo real da metodologia mostrada no Capítulo 4. Os resultados obtidos indicam a eficiência da metodologia proposta, possibilitando o levantamento de conjecturas sobre a aplicabilidade de cada arquitetura na modelagem de sistemas dinâmicos não lineares. O Capítulo 6 finaliza demonstrando as conclusões da pesquisa e apresenta algumas perspectivas e sugestões para a continuação deste trabalho.

CAPÍTULO 2
INTRODUÇÃO AO CONTROLE *FUZZY*

2. INTRODUÇÃO AO CONTROLE *FUZZY*

2.1 INTRODUÇÃO

Neste capítulo apresenta-se uma introdução ao controlador *fuzzy* e aos sistemas não-lineares que são os principais assuntos necessários ao desenvolvimento desta dissertação. Inicialmente, será feito um breve histórico do controle nebuloso, sua teoria, alguns conceitos e algumas aplicações.

2.3 UM BREVE HISTÓRICO

O primeiro estudioso que se tem registro de tentar formalizar uma representação sistêmica do processo do pensamento foi Aristóteles (384-322 a.C.). A estrutura lógica de Aristóteles permaneceu intacta durante muitos séculos. Só no século XIX que começaram a surgir novas teóricas sobre estruturas lógicas que refutasse ou contribuísse com a logica clássica.

G. Leibniz (1646-1716), Saccheri (1667-1733), J. H. Lambert (1728-1777) e, em seguida, A. De Morgan (1806-1871), G. Frege (1848-1925) e Bertrand Russell (1872-1970) fizeram tentativas e realizaram grandes contribuições acerca do objetivo de criar uma teoria matemática associada à lógica, mas foi George Boole (1815-1864), que definiu a lógica através de uma estrutura matemática de forma contundente.

O grande trunfo desta técnica é que a lógica poderia ser manipulada algebricamente e os resultados operações obtidos por técnicas matemáticas, representaram um grande passo na ciência da computação, no sentido de auxiliar na simulação e implementação do processo de eletricidade e eletrônica e do raciocínio em máquinas. Entretanto, no Século XX, vários outros estudiosos contribuíram para definir melhor os princípios da logica já ora consolidados.

A lógica nebulosa, ou Lógica *Fuzzy*, como é internacionalmente conhecida, foi primeiramente publicada em um artigo por Lotfi A. Zadeh, da Universidade da Califórnia em Berkeley, em 1965. Ele próprio detalhou os conceitos da lógica

introduzindo o conceito de "variáveis linguísticas" em outro artigo de 1973. (ZADEH,1973). Zadeh afirmava que, até a década de 1960, devido às características ambíguas dos processos de decisão, as ferramentas lógicas disponíveis não eram capazes de modelar as atividades relacionadas a problemas de natureza industrial, uma vez que a lógica computacional clássica ou booleana eram limitadas nesse aspecto. Na década seguinte, uma grande contribuição prática veio através do Professor Mandani, que aplicou os conceitos na construção e controle de uma máquina a vapor. (WANG, 1997), (YONEYAMA e NASCIMENTO Jr, 2000).

A década de 1970 foi marcada por grandes implementações de controladores *fuzzy* em sistemas reais. Depois que Mandani e Assilian construíram e aplicaram um controlador *fuzzy* na máquina a vapor, Holmblad e Ostergaad, em 1978, desenvolveram o primeiro sistema *fuzzy* para processos industriais em larga escala aplicando em um forno de cimento. (YONEYAMA e NASCIMENTO Jr, 2000).

Na década de 1980, os avanços ocorreram de forma paulatina. Poucos pesquisadores se aprofundaram e houve poucas novas propostas de conceitos e abordagens. Entretanto, os mais entusiastas desta nova técnica foram as empresas japonesas que, não apenas mostraram grande interesse em sistemas *fuzzy*, como implementaram grandes projetos com sucesso em área de infraestrutura de suas cidades, definido o cenário de que a nova técnica tem excelentes resultados práticos.

Nesse contexto, Sugeno, ainda na década de 1980, cria a primeira aplicação *fuzzy* genuinamente japonesa – um sistema de purificação de água. Na mesma década Sugeno também criou um sistema autocontrolado de estacionamento por comando de voz. Yasubono e Yamamoto criam um sistema de controle para o metrô da cidade de Sendai, no Japão, no final da década de 1980. (MENDEL, 1995).

O sucesso japonês impressionou inclusive os pesquisadores mais críticos e céticos. A partir da década de 1990, do ponto de vista teórico, houve muitos avanços. Vieram em seguida, várias outras aplicações, destacando-se, por exemplo, os controladores *fuzzy* de plantas nucleares, refinarias, processos biológicos e

químicos, trocador de calor, máquina diesel, tratamento de água e sistema de operação automática de trens (MENDEL, 1995).

2.4 BREVE INTRODUÇÃO SOBRE SISTEMAS *FUZZY*

A lógica *fuzzy* foi desenvolvida com o intuito de satisfazer soluções plausíveis para problemas de grandezas imprecisas e incertas, que são totalmente aplicadas em questões envolvendo decisões dicotômicas, cujas formulações são incompletas e fazem pouco sentido ou lógica.

De forma mais clara e objetiva, podemos definir os Sistemas *Fuzzy*, como uma técnica de logica capaz de capturar e combinar informações vagas e imprecisas, descritas e definidas em uma linguagem natural, e transformá-las em dados de formato numérico, de fácil manipulação, cujo valor possui uma representatividade mais precisa.

Em termos de conjuntos clássicos, as variáveis de entrada e saída do sistema são definidas por divisões, mas precisamente como faixa de estado de grandeza hierárquica: “baixo”, “médio”, “alto”, ou então, "frio", "fresco", "moderado", "morno", "quente”, como na Figura 2.1.

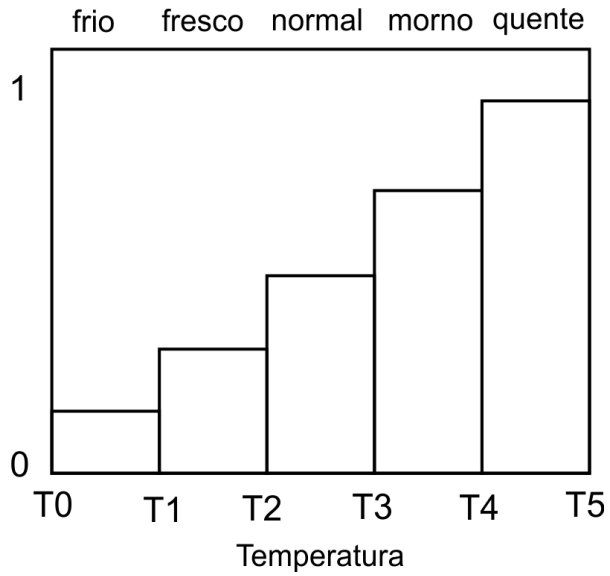


Figura 2.1 - Temperatura representada sob a escala dos conjuntos clássicos.

Contudo, para um conjunto desses valores pré-definidos, a transição ou um valor médio de um estado para o próximo é difícil de mensurar com precisão. Um valor arbitrário pode ser definido para dividir o "morno", do "quente". Entretanto, isso implicaria em uma descontinuidade entre as faixas.

A solução é criar os estados com mudanças suaves, que permitem uma transição gradual de um estado para o outro. Poderia se definir a temperatura de entrada usando funções características como na Figura 2.2.

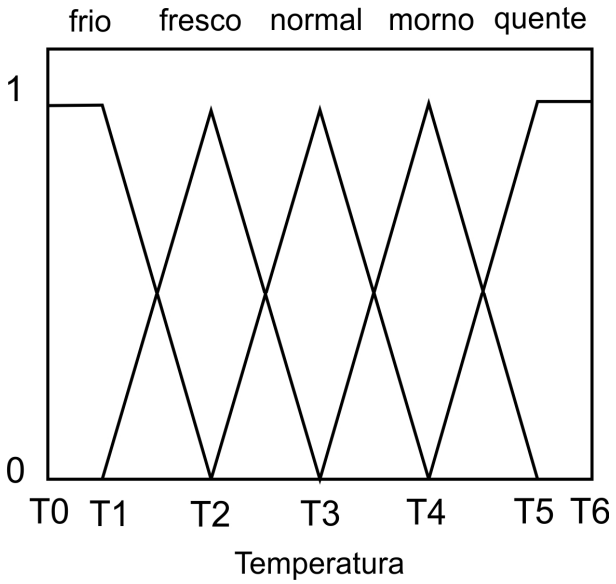


Figura 2.2 - Funções de características de temperatura.

Observa-se que, no exemplo da Figura 2.1, o estado da variável de entrada possui transição abrupta entre seus valores naturais. Na Figura 2.2, possui, agora, uma transição linear em relação aos estados vizinhos. Um conceito relacionado com conjuntos *fuzzy* é o de variável linguística. Entende-se por variável um identificador que pode assumir um dentre vários valores. Deste modo, uma variável linguística pode assumir um valor linguístico dentre vários outros em um conjunto de termos linguísticos. Cada termo linguístico é caracterizado por uma função de pertinência.

As variáveis de entrada em um sistema de controle nebuloso são, em geral, mapeadas dentro de conjuntos de funções de pertinência. Em princípio, as funções de pertinência podem ser qualquer função que produza valores entre 0 e 1.

De forma prática, o valor atual da variável é uma combinação percentual ou proporcional de estados de acordo com suas transições. O processo de conversão de um valor de entrada em um valor nebuloso é chamado de "codificação" ou “Fuzzificação”

Após a realização do mapeamento das variáveis de entrada com as funções de pertinência, a decisões sobre a saída do sistema são baseadas em um conjunto de regras do tipo condicional, com a seguinte estrutura:

SE {estado da variável de entrada} ENTÃO {estado da variável de saída}.

Para sistemas com estruturas de múltiplas entradas e múltiplas saídas, pode ocorrer regras condicionais como abaixo:

SE {estado da variável de entrada 1} E/OU { estado da variável de entrada 2} ENTÃO {estado da variável de saída 1} E/OU { estado da variável de saída 2}

No Sistema *Fuzzy*, uma ou todas as regras podem ser invocadas, usando as funções características e os valores verdadeiros obtidos das entradas, para determinar o resultado da regra. As bases de regras podem ser bem entendidas na seção 2.6. Essa Etapa é conhecida como inferência associativa paralela.

Da mesma forma como são as variáveis de entradas, as variáveis de saída são constituídas pelas funções de pertinência, que relacionam estados naturais com valores reais. Entretanto, as respostas das variáveis de saída possuem o processo inverso em relação às variáveis de entrada. O resultado das regras mapeiam os estados dentro das funções de pertinência de saída e depois estes resultados são combinados para gerar uma resposta específica, com valores reais de grandezas mensuráveis.

Este procedimento é conhecido como "decodificação" ou “Defuzzificação”. A combinação de operações nebulosas e regras descrevem um sistema nebuloso. A seguir serão detalhados estes conceitos de sistemas nebulosos.

2.5 CONJUNTOS *FUZZY*

Na teoria clássica de conjuntos, um elemento possui apenas duas possibilidades quanto a sua relação de pertinência com um conjunto. Ou o elemento pertence ao conjunto, ou não pertence. Na teoria de conjuntos nebulosos, formulada por ZADEH (1965), um objeto possui variados graus de pertinência com um determinado conjunto.

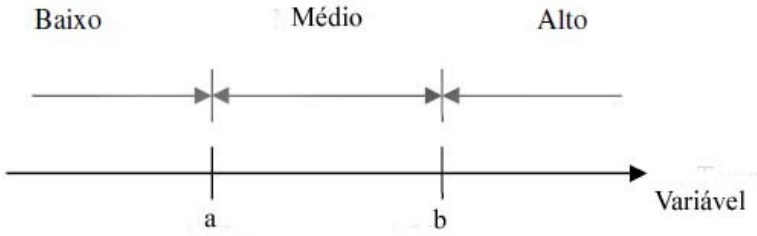


Figura 2.3 - Graus de pertinência de uma variável genérica.

A formalização do conceito de conjuntos nebulosos pode ser obtida estendendo a teoria clássica de conjuntos. Assim, na teoria clássica, um conjunto pode ser caracterizado pela sua função característica, ou seja, dado um conjunto A no universo X, define-se IA(x):X→(0,1) por

$$I_A(x) = \begin{cases} 1 & se \quad x \in A \\ 0 & se \quad x \notin A \end{cases} \tag{2.1}$$

Se X for um conjunto R+ e A um intervalo fechado, a função indicadora de A assume o aspecto ilustrado na Figura 2.4 abaixo:

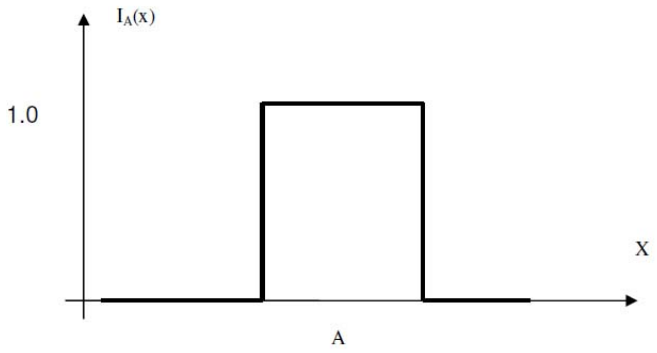


Figura 2.4 - Função característica de um conjunto clássico

Por outro lado, usando o conceito de conjuntos nebulosos, as incertezas quanto ao que seria um valor determinado de uma variável ficariam representadas por um grau de pertinência, como Figura 2.5.

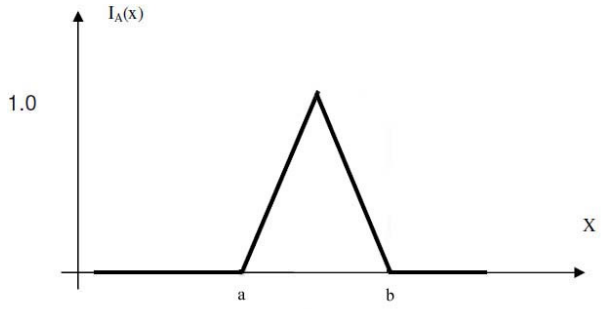


Figura 2.5 - Função característica de um conjunto *fuzzy*.

De forma análoga, os conjuntos nebulosos são definidos por:

$$\mu_A(x) : X \rightarrow [0,1] \tag{2.2}$$

Esta função é conhecida como a função de pertinência, que expressa o quanto um dado elemento X pertence a A. A funções de pertinência não precisam ter uma geometria definida e podem ser qualquer função que produza valores entre 0 e 1. Entretanto existem algumas formas já convencionadas pela literatura que são as mais utilizadas como, por exemplo, formas triangulares, trapezoidais, sigmóides e gaussianas (Figura 2.6).

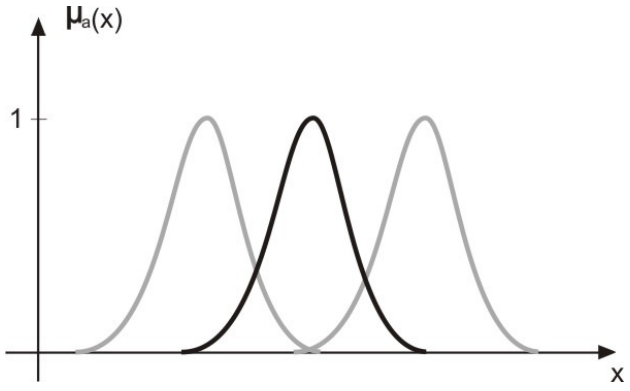


Figura 2.6 - Funções de Pertinência gaussianas

A teoria de conjuntos nebulosos busca, portanto, traduzir em termos formais a informação imprecisa que ocorre de maneira natural na representação dos fenômenos da natureza, como descrito por seres humanos utilizando uma linguagem corriqueira.

As notações de teoria clássica de conjuntos são amplamente estudadas e bastante difundidas. Essa teoria dispõe de diversas operações e propriedades que possibilitam a manipulação de elementos dentro de um universo.

A teoria dos conjuntos *fuzzy* ou nebulosos é uma extensão da teoria clássica de conjunto, inclusive, possibilitando fazer associações semânticas análogas e que resultariam em expressões do tipo:

$$(\alpha \vee \beta), (\alpha \wedge \beta), (\neg \alpha) \tag{2.3}$$

Assim, ao supor que existam dois conjuntos A e B, onde são caracterizadas pelas funções de pertinência μA e μB, sobre um universo X, é perfeitamente possível definir, portanto, para conjuntos nebulosos, as operações com conjuntos:

$$A \cup B, \quad A \cap B \quad e \quad A^c \tag{2.4}$$

a) União:

O conjunto nebuloso *C* = *A*U*B* pode ser caracterizado pela função de pertinência:

$$\forall x \in X, \mu_c(x) = \max\{\mu_A(x), \mu_B(x)\} \tag{2.5}$$

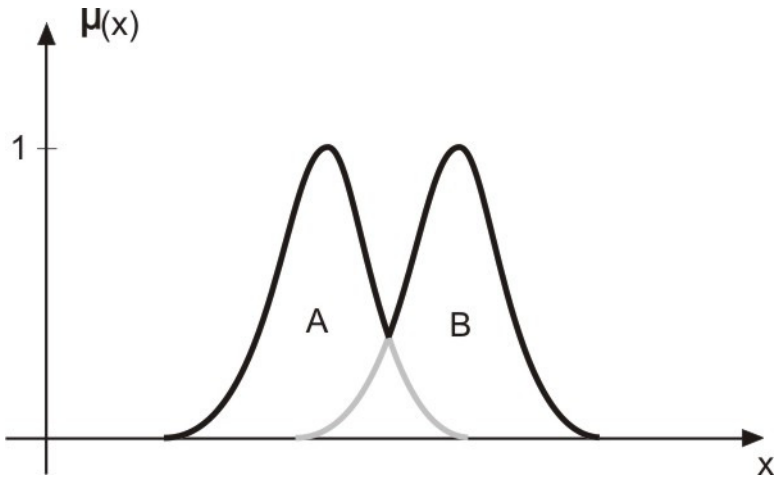


Figura 2.7 - Operação de União

b) Interseção:

O conjunto C = *A*∩*B* pode ser caracterizado pela função de pertinência:

$$\forall x \in X, \mu_D(x) = \min\{\mu_A(x), \mu_B(x)\} \tag{2.6}$$

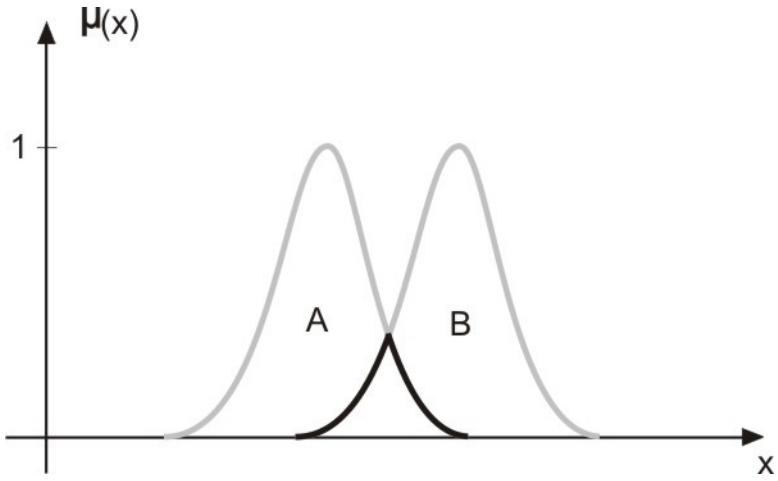


Figura 2.8 - Operação de Interseção

c) Complementação:

O conjunto C=A^c, diz-se que E é o complemento de A, pode ser caracterizado pela função de pertinência:

$$\forall x \in X, \mu_E(x) = 1 - \mu_A(x) \tag{2.7}$$

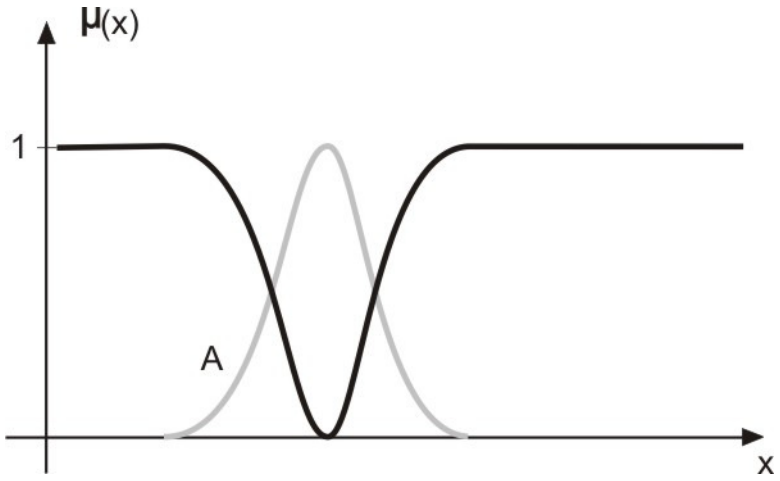


Figura 2.9 - Operação de Complementação

d) Produto Cartesiano:

O conjunto *C = A x B* pode ser caracterizado pela função de pertinência:

$$\forall x \in X, \mu_F(x) = \mu_A(x)\mu_B(x) \tag{2.8}$$

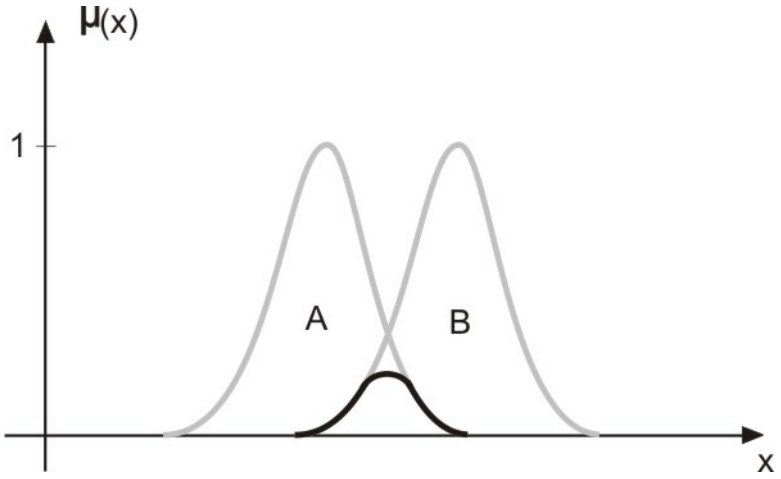


Figura 2.10 - Operação de Produto

Algumas propriedades também podem ser verificadas para conjuntos nebulosos (YONEYAMA, 2000).

a) $(A^c)^c = A$; Involução
b) $(A \cap B)^c = A^c \cup B^c$; De Morgan
c) $(A \cup B)^c = A^c \cap B^c$; De Morgan
d) $A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$; Distributividade da ∩
e) $A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$; Distributividade da U
f) $(A \cap B) \cup A = A$; Absorção
g) $(A \cup B) \cap A = A$; Absorção
h) $A \cup A = A$; Idempotência
i) $A \cap A = A$; Idempotência

Posteriormente, com o objetivo de generalização, foram definidos de base axiomática, baseados nos conceitos de norma triangular. Esta inferência consiste de dois passos: avaliação da premissa de cada regra (conjunção), através dos operadores t-norma, e em seguida a etapa de agregação, ponderando as diferentes conclusões das regras ativas sob o operador s-norma (SHAW e SIMÕES, 1999)(WANG, 1997). Os operadores de t-norma e s-norma são normas e co-normas triangulares que fornecem métodos genéricos para as operações de intersecção e união em conjuntos *fuzzy*. Alguns exemplos de t-normas são o produto e o operador de mínimo. Para as s-normas têm-se como exemplos o operador de máximo ou a soma drástica. (SHAW E SIMÕES, 1999) (WANG, 1997).

2.6 BASES DE REGRAS DA LÓGICA NEBULOSA

Uma afirmação frequentemente feita é a de que um sistema dinâmico qualquer pode ser interpretado como uma conversão de valores entre entradas e saídas. No item anterior, revisamos a forma com que cada entrada e saída se relacionam com o meio exterior ao sistema, capturando valores reais e transformando em dados do universo *fuzzy* e vice versa.

Como foram introduzidas anteriormente, estas conversões são realizadas a partir de um conjunto de implicações linguística na forma condicional, que relaciona

as funções de pertinência das variáveis de entrada com as funções de pertinência das variáveis de saída. (PADILHA, 2001).

A etapa de inferência consiste em se avaliar um conjunto de regras condicionais (a base de conhecimento) do tipo "*se* . . . *então* . . ." que descrevem a dependência entre as variáveis linguísticas de entrada e as de saída. Essas regras seguem o paradigma *modus ponens*, um mecanismo de inferência progressiva (ao contrário do *modus tollens*, regressivo, mais utilizado em sistemas especialistas), para fazer o mapeamento do conceito de implicação lógica. A mais simples base de regras tem a seguinte sintaxe (WANG, 1997):

$$A \Rightarrow B, \text{ isto é, } \textbf{SE } \{A(u)\} \textbf{ ENTÃO } \{B(v)\}$$

Para uma base de regras com N regras, lista-se as regras da seguinte forma:

$$R1: [\textbf{SE } \{A1(u)\} \textbf{ ENTÃO } \{B1(v)\}]$$
$$R2: [\textbf{SE } \{A2(u)\} \textbf{ ENTÃO } \{B2(v)\}]$$
$$\vdots$$
$$RN: [\textbf{SE } \{AN(u)\} \textbf{ ENTÃO } \{BN(v)\}]$$

Um sistema *fuzzy* contém um conjunto dessas regras, todas ativadas em paralelo. Assim, o sistema *fuzzy* funciona com inferência associativa paralela. Quando o sistema detecta uma entrada, as regras são disparadas paralelamente, com diferentes graus de ativação, para inferir um resultado ou saída.

Como pode acontecer de saídas distintas serem acionadas em um mesmo momento, com diferentes graus de ativação, deve-se encontrar o valor que melhor corresponda à distribuição de possibilidade da combinação dos conjuntos *fuzzy* de saída. Dessa forma, existem dois modelos de arquitetura de saída para sistemas *fuzzy*: os modelos são *fuzzy* **Mamdani** e **Takagi-Sugeno** (WANG, 1997).

A diferença entre os modelos *fuzzy* Mamdani e Takagi-Sugeno reside nos consequentes das regras da base de regras de inferência.

Em um modelo **Mamdani**, as regras são do tipo:

se (u1 é A1,1) e . . . e (un é A1,n) então y é B1

...

se (u1 é Am,1) e . . . e (un é Am,n) então y é Br

sendo B1, . . . ,Br os termos linguísticos, que correspondem a funções de pertinência, para a saída y (neste exemplo, a saída do sistema é monovariável). Cada regra acima define uma região *fuzzy* no espaço de entrada-saída. O conjunto de todas as regras particionam esse espaço em regiões *fuzzy* sobrepostas e combinadas paralelamente.

Esse modelo em particular requer, após o processo de inferência *fuzzy* da saída, a aplicação da etapa de "*defuzzificação*" citada na seção anterior. Há vários métodos de defuzzificação, sendo os principais: Centro de "Gravidade" e Centro dos Máximos.

O método do centro da gravidade é o mais comum, entretanto este requer um considerável esforço computacional, pois surge em sua definição o cálculo numérico de integrais, com os objetivos de encontrar o centro de "gravidade" da forma gerada pelas operações com as funções de pertinência.

O método dos centros dos máximos (SHAW E SIMÕES,1999) trata-se de calcular a média ponderada entre os valores de cada termo da variável linguística de saída. Este método atende um requisito essencial em aplicações em controle, a continuidade (SHAW E SIMÕES,1999). Isso significa que uma mudança infinitesimal em uma variável de entrada não causa uma variação abrupta em nenhuma das variáveis de saída.

No modelo **Takagi-Sugeno**, a base de regras é da forma:

se (u1 é A1,1) e . . . e (un é A1,n) então y = f1(u1, . . . , un)

...

se (u1 é Am,1) e . . . e (un é Am,n) então y = fr(u1, . . . , un)

Assim, os consequentes das regras são agora compostos por uma função qualquer das variáveis de entrada. Pode ser usado um número menor de variáveis nas premissas com o objetivo de diminuir o tamanho da base de regras. O modelo *fuzzy* Takagi-Sugeno combina uma descrição global baseada em regras com uma aproximação local que, no contexto de identificação de sistemas, é normalmente escolhida como um modelo de regressão linear. No caso de as funções *fi* serem funções afins, por exemplo, o consequente da i-ésima regra seria da forma:

$$y = a_{i,1}u_1 + \ldots + a_{i,n}u_n + a_{i,n+1} \tag{2.9}$$

No modelo *fuzzy* TS resultante cada regra descrever uma região *fuzzy* na qual as saídas dependem das entradas de forma linear. Os parâmetros dos consequentes ai,1, . . . , ai,n+1 podem ser facilmente estimados via Método de Mínimos Quadrados.

Como as saídas das regras não estão definidas através de termos linguísticos, não faz sentido falar de "*defuzzificação*" após o processo de inferência *fuzzy*. A saída final ˆy é calculada da mesma forma que na equação (2.9), com a diferença que o valor de cada yi é o resultado direto do consequente da i-ésima regra.

2.7 ESTRUTURA DE CONTROLADORES EMPREGANDO LÓGICA NEBULOSA

O problema típico de controle consiste em obter, a partir do conhecimento da dinâmica da planta, uma lei de controle que atenda as especificações desejadas. É muito comum, em sistemas de controle, o uso de controladores em série com a planta como mostra a Figura 2.11.

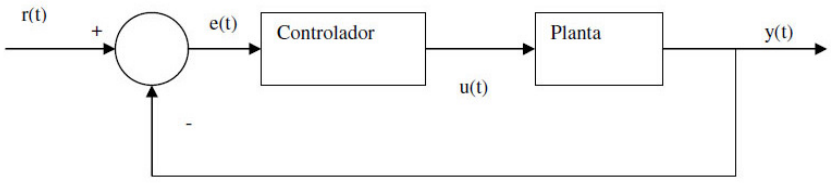


Figura 2.11 - Diagrama de um sistema de controle *fuzzy* típico.

Os controladores empregando lógica nebulosa possuem, através de um nível mais detalhado, três blocos, como especificado na Figura 2.11.

A interface de codificação é também chamada de “fuzzificador”. Como visto na figura 2.12, sua função é converter o sinal de entrada, de grandeza real, para um valor apropriado para inferência nebulosa. Os valores e limites das funções de pertinência devem ser adequados para cada um dos valores linguísticos associados à grandeza do sinal.

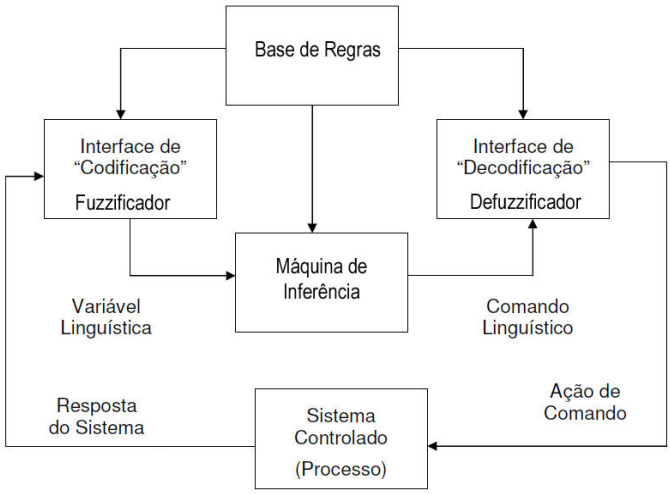


Figura 2.12 - Representação em blocos de um controlador de lógica *fuzzy*

Formalmente realiza as seguintes funções:

- Medida dos valores das variáveis de entrada.
- Mapeamento dos valores assumidos pelas variáveis de entrada no universo de referência correspondente.
- Conversão dos dados de entrada em valores linguísticos apropriados.

O sinal, já convertidos em graus de pertinência de conjuntos nebulosos, passa para a máquina de inferência e gera, a partir da base de regras desenvolvidas pelo projetista, ativações para as várias saídas, onde os valores ainda são definidos em graus de pertinência.

A máquina de inferência é o núcleo de um controlador nebuloso, e possui a capacidade de simulação do processo de decisão humano baseado em conceitos nebulosos e em ações de controle tomadas através das bases de regras de dedução nebulosa.

A base de regras caracteriza os objetivos e a política de controle do sistema o que compreende um conhecimento do domínio da aplicação e dos objetivos desejados. Ela fornece as definições necessárias para a correta execução das regras de controle e da manipulação de dados.

Os valores de saída, ainda definidas em graus de pertinência, através da combinação das funções de pertinência das saídas ativadas pela base de regra, são convertidas para valores numéricos de grandeza real. O defuzzificador efetua o mapeamento dos valores assumidos pelas variáveis de saída no seu universo de referência correspondente e a converte em uma ação de comando.

Esse processo de conversão, realizado no defuzzificador, é o responsável por transformar as variáveis linguísticas geradas pela máquina de inferência nebulosa em um valor real.

Os sistemas *fuzzy* já possuem uma base teórica bastante consistente e uma serie de aplicações já desenvolvidas com sucesso. Dessa forma, é interessante conhecer uma série de aspectos que podem ajudar a decidir se o sistema é adequado ao projeto. Um dos aspectos mais favoráveis é possibilidade de incorporar conhecimentos e experiência acumulada do operador humano sobre o controle manual da planta. Além disso, não há necessidade de um modelo do sistema a ser controlado e é totalmente aplicável em sistemas de dinâmica complexa e não-linear, possibilitando um maior desempenho nesse tipo de aplicação.

Os aspectos desfavoráveis iniciam com a impossibilidade de demonstração, nos casos gerais, de propriedades e critérios de projeto como a estabilidade, ausência de ciclos limite, etc. Outro problema é a ausência de diretrizes precisas para o projeto do controlador nebuloso, mesmo com uma base teórica bem fundamentada e consumada, resultando em uma sintonia que, na maioria das vezes,

exige um ajuste aspecto artesanal e heurístico. Existe ainda a possibilidade das regras não possuírem uma consistência garantida.

2.8 CONCLUSÃO DO CAPÍTULO

Este capítulo ressalta vários aspectos, a partir de um breve histórico seguido de um contexto matemático e incluindo algumas propriedades do sistema fuzzy, embora apresentada de forma resumida, pois, há diversas publicações específicas sobre o assunto. É importante destacar os tipos de controlador, os processos de inferência e a estrutura empregada nos sistemas de malha fechada que são comumente utilizadas em projetos de controle e automação, e que são utilizadas neste trabalho.

CAPÍTULO 3
PROJETO DE CONTROLADORES *FUZZY* UTILIZANDO ALGORITMOS GENÉTICOS

3. PROJETO DE CONTROLADORES *FUZZY* UTILIZANDO ALGORITMOS GENÉTICOS

3.1 INTRODUÇÃO

O Algoritmo Genético é uma instância da computação evolutiva que é constituída por métodos computacionais inspirados na teoria da evolução natural das espécies. Os métodos citados tem por base a geração, em um computador, de uma população de indivíduos representados por cromossomos, que na prática é um conjunto de vetores de números e que representam uma analogia aos cromossomas de quatro bases (timina, guanina, adenosina e citosina) existentes no DNA natural. Os indivíduos na população são então submetidos a um processo de evolução através de operadores de mutação, recombinação e seleção e sujeitos a uma avaliação com objetivo de se obter o melhor indivíduo. A cada iteração desse grupo de operadores damos o nome de geração.

Os algoritmos genéticos foram desenvolvidos por Holland no início dos anos 60 (HOLLAND, 1975). Foi projetado inicialmente como um sistema para adaptação e posteriormente utilizado como método de otimização. Suas características básicas eram a forte ênfase em recombinação (crossover), uso de um operador de seleção probabilístico e a interpretação da mutação como um operador secundário. A proposição inicial representava soluções através de cadeias binárias e um grande número de variantes foi desenvolvido para ampliar o âmbito de aplicações do algoritmo.

O algoritmo genético apresentado neste trabalho pode ser considerado padrão apesar de algumas adaptações. Entretanto descrevem o funcionamento da maioria dos algoritmos evolutivos. Sendo assim, no contexto deste trabalho, este capítulo faz uma descrição acerca da técnica dos algoritmos genéticos e sobre a elaboração para o projeto automático do sistema *Fuzzy* Mamdani. Em cada seção, a partir da seção 3.2, detalha-se uma característica ou componente dos algoritmos genéticos de um modo geral e então a sua configuração específica na metodologia ora proposta.

3.2 ALGORITMO GENÉTICO BÁSICO

Para iniciar um algoritmo genético básico devem-se definir os aspectos da representação da população, a ordem e forma de atuação dos operadores de seleção e reprodução.

O algoritmo básico do AG é constituído por quatro etapas principais:

- Sorteio da População;
- Avaliação;
- Seleção (Reprodução);
- Cruzamento e Mutação.

No Algoritmo Genético, o indivíduo pode ser representado por uma representação binária através de um vetor de 0s e 1s, que mais se aproxima do modelo real do cromossomo, ou por um vetor de números inteiros ou reais. Geralmente, cada indivíduo gerado em cada rodada representa um candidato à solução, embora possam existir situações em que a solução pode ser uma população dos indivíduos.

O operador “mutação” insere pequenas variações no cromossomo dos indivíduos normalmente por distribuição normal, mas podem ter distribuição uniforme ou, simplesmente, aleatoriamente factíveis.

O operador “recombinação” é caracterizado pela participação de dois indivíduos (pais) na geração de um ou mais filhos, onde os cromossomos possuem partes distintas de cada pai.

A seleção realiza a seleção dos melhores indivíduos e pode ser implementada de diversas maneiras, sendo as mais comuns a roleta e o torneio, e ainda uniforme ou estocástica, desempenhando um papel fundamental na manutenção da diversidade da população de indivíduos. Neste caso, o método da roleta se caracteriza por provocar alta pressão seletiva na população, convergindo a

população para ótimos locais mais rapidamente. Já o torneio permite um ajuste mais apurado da pressão seletiva, tornando possível a manutenção da diversidade da população.

A Figura 3.1 ilustra o elo entre estas etapas de forma básica, mas que será tomada como base na construção do AG que foi aplicado ao problema abordado neste trabalho.

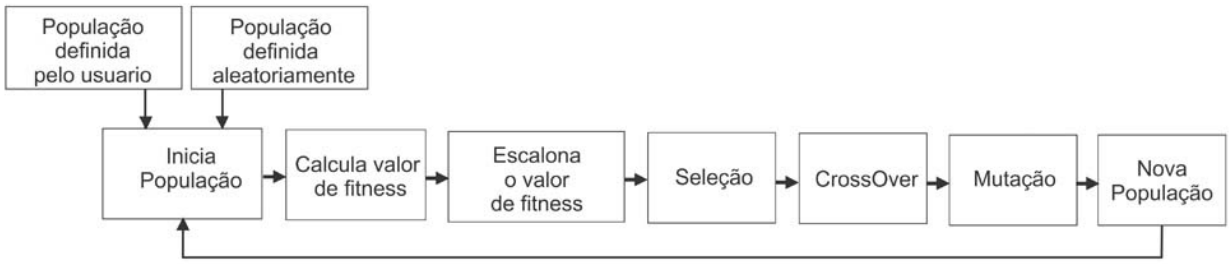


Figura 3.1 - Diagrama de um algoritmo genético básico

A priori, uma população de indivíduos é gerada de forma aleatória, mas é possível definir a população inicial. Entretanto, é necessário estar atento à representação do espaço de busca por parte dos indivíduos criados.

Segundo passo, é definir o tipo de dado que deverá estar contidos em cada indivíduo (representação cromossômica), que podem ser de cadeias binárias, numéricas ou mistas que dependem do tipo de aplicação e, principalmente, do resultado que se deve esperar.

Por fim, deve-se avaliar cada indivíduo da população. Um conceito muito importante em algoritmos genéticos é a função de *fitness* ou função objetivo. A função de *fitness* é utilizada para medir se cada indivíduo possui a maior aptidão possível para a solução do problema tratado pelo algoritmo. Dessa forma, através um operador de seleção, são escolhidos os indivíduos com os melhores fitness de forma privilegiada, que correspondem aos melhores indivíduos da população.

Além disso, o algoritmo genético submete os melhores indivíduos a operadores genéticos de recombinação e mutação para a composição da nova geração da população, passando suas características.

Essa nova população é submetida à avaliação, da mesma forma que a inicial ou anterior. O processo se repete até que uma condição de parada especificada seja atendida. A Figura 3.1 mostra um fluxograma do processo utilizado pelo algoritmo genético.

3.3 INTRODUÇÃO AO PROJETO AUTOMÁTICO DE MODELOS *FUZZY*

As pesquisas recentes têm mostrado que sistemas inteligentes híbridos fornecem métodos eficientes para aplicações práticas. Ao se compensar as deficiências de uma técnica com os benefícios de outra, criam-se estruturas de enorme potencial onde cada metodologia tem suas vantagens. (JAIN e JAIN, 1997)

Desde a última metade da década de 1980 têm sido realizadas intensas pesquisas nos campos das redes neurais, sistemas *fuzzy* e algoritmos genéticos. O crescente número de publicações nessa área indica que estudos combinando essas diferentes ferramentas têm aumentado bastante principalmente a combinação entre redes neurais e sistemas *fuzzy*, em que já foram incorporadas em diversos produtos e sistemas. (JAIN e JAIN, 1997).

Os algoritmos genéticos e as redes neurais possuem uma característica bem flexível de adaptabilidade e podem se aproximar de funções não lineares. Sistemas *fuzzy* podem ainda incorporar o conhecimento especializado e empírico de um ser humano de forma bastante eficiente. Já as redes neurais podem tratar dessas informações implicitamente, além da característica de aprendizagem.

Os algoritmos genéticos caracterizam-se pelas propriedades de busca local e global de pontos máximos e mínimos de uma função. Um exemplo de problema mais comumente usado é definir os parâmetros de um Controlado PID através de um Controlador *Fuzzy*, sendo este ultimo treinado por um algoritmo genético (SWIECH et al., 2004).

Outro exemplo de problema que motivaria a aplicação de um sistema inteligente híbrido relaciona-se à definição adequada dos parâmetros de um algoritmo genético. As taxas de ocorrência dos operadores de reprodução, o

tamanho da população, a precisão na representação dos indivíduos, podem, por exemplo, ser automaticamente calculadas utilizando-se um sistema *fuzzy (*HERRERA e LOZANO, 1996).

Normalmente a configuração oposta é mais comumente pesquisada, seja a utilização de algoritmos genéticos para a determinação dos parâmetros ótimos de um sistema *fuzzy*, ou mesmo de um sistema neuro-*fuzzy* (SENG et al.,1999).

Para este último caso, relata-se, por exemplo, o uso de algoritmos genéticos na definição adequada das funções de pertinência para um sistema *fuzzy* cujas regras de controle são aprendidas através do treinamento não supervisionado de uma rede neural, a qual também implementa as operações de inferência e defuzzificação.

No caso do uso de algoritmos evolutivos em sistemas *fuzzy*, a meta geralmente estabelecida é a definição de um conjunto de funções de pertinência (de entrada e/ou de saída) além da geração automática da base de regras de inferência (LEE e TAKAGI, 1993). A Figura 3.2 ilustra a forma com que o algoritmo genético interage com o sistema *fuzzy*.

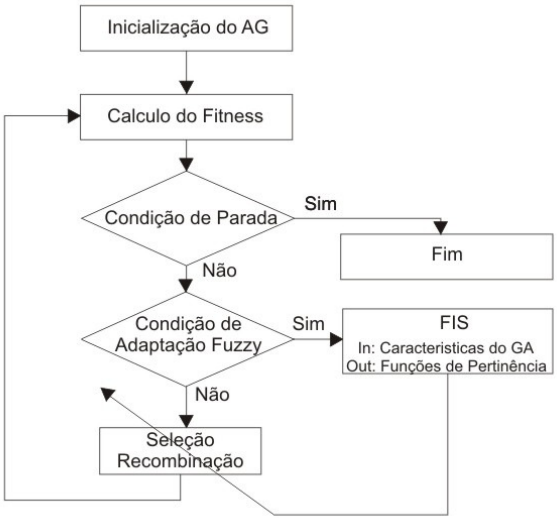


Figura 3.2. Interação entre o Algoritmo Genético e o Sistema *Fuzzy*

3.4 EVOLUÇÃO DE SISTEMA *FUZZY*

LEE e TAKAGI (1993) propuseram um método para projetar automaticamente uma série de elementos de um sistema de controle *fuzzy*. Neste trabalho são referenciados outros desde o ano de 1989, que não consideravam todos os componentes do sistema *fuzzy* simultaneamente. A proposta de Lee e Takagi determinava, utilizando algoritmos genéticos, as funções de pertinência (FP) para um número pré-determinado de variáveis de entrada, o número de regras e os parâmetros dos consequentes das regras de um sistema do tipo Takagi-Sugeno-Kang. A aplicação consistiu no controle de um pêndulo invertido.

LISKA e MELSHEIMER (1994) propuseram um AG para o projeto de um sistema *fuzzy*, englobando:

- Número de regras e suas estruturas.
- Parâmetros das funções de pertinência.
- Uso do método do gradiente conjugado para melhorar o resultado final.

A aplicação inicial foi a modelagem de um sistema dinâmico não linear, utilizando um modelo *fuzzy* do tipo Mamdani. NG e LI (1994) propuseram uma abordagem de projetos de AG para controladores *fuzzy* PI e PD. Um algoritmo genético foi aplicado para otimizar as funções de pertinência exponenciais parametrizadas pelas constantes α, β e σ, conforme a Equação (2.10).

$$\mu(x) = exp\left(-\frac{|x \pm \alpha|^{\beta}}{\sigma}\right) \tag{2.10}$$

Além disso, a base de regras completa do sistema do tipo Mamdani e os ganhos proporcional e integral dos controladores foram também obtidos com o algoritmo genético.

HOMAIFAR e MCCORMICK (1995) projetaram um sistema de controle *fuzzy* do tipo Mamdani, com uma base de regras completa. Um AG foi usado para calcular simultaneamente as FP triangulares de 2 variáveis de entrada e a base de regras.

Esta metodologia foi utilizada para o controle de posicionamento de um carro (Cart Controller) e de um caminhão (Truck-backing). SENG et al. (1999) projetaram um sistema neuro-*fuzzy* baseado em sintonia automática via algoritmos genéticos com funções de pertinência gaussianas e base de regras completa.

Diversos outros trabalhos envolvendo algoritmos genéticos foram utilizados, não apenas para controladores, mais para os mais diversos tipos de aplicações. TOSATTI et.al. (2008) desenvolveram um sistema híbrido genético *fuzzy*, para utilização em diagnósticos de câncer. Outros estudos relatam a aplicação de diversas configurações de algoritmos genéticos como parte de sistemas complexos de pré-processamento de dados em sistemas de potência, dimensionamento de sistema de sistema de energia e controle multi-agente de manipuladores robóticos.

Grande parte das publicações atuais foca em aplicações utilizando a abordagem genético-*fuzzy* dentro de um contexto complexo e maior, não sendo estes os destaques maiores da pesquisa científica.

Ao se utilizar algoritmos genéticos para solucionar um problema, deve-se analisar a influência dos parâmetros de ajuste dessa ferramenta ao sistema que será otimizado. Por isso é necessário formalizar os aspectos técnicos da planta para iniciar os processos de otimização via algoritmos genéticos.

3.5 CONCLUSÃO DO CAPÍTULO

Os algoritmos genéticos foram desenvolvidos por Holland no início dos anos 60 inicialmente como um sistema para adaptação e posteriormente utilizado como método de otimização. Nesse sentido, houve a experimentação de utilizar os algoritmos genéticos como forma de otimizar e automatizar a obtenção de parâmetros de controladores, entre eles, o controlador fuzzy. O objetivo deste capítulo é contextualizar este conhecimento ao objetivo desta dissertação uma vez que grande parte da literatura e pesquisas atuais focam em aplicações utilizando a abordagem genético-*fuzzy* de forma generalista.

CAPÍTULO 4
MODELAGEM E IMPLEMENTAÇÃO DO SISTEMA DE NÍVEL DE LÍQUIDO

4. MODELAGEM E IMPLEMENTAÇÃO DO SISTEMA DE NÍVEL DE LÍQUIDO

4.1 INTRODUÇÃO

O objetivo desse capítulo é descrever os principais conceitos relacionados ao contexto no qual se insere a presente dissertação. São apresentadas as bases teóricas que serão utilizadas em capítulos posteriores e os principais conceitos envolvidos. Dessa forma, parte-se das definições simples (como o conceito de sistema e modelo) e prossegue-se até a contextualização de todo o trabalho. Delega-se aos capítulos subsequentes a tarefa de aprofundamento dos principais temas abordados.

Neste trabalho, a técnica explanada nos capítulos anteriores será validada através de um sistema real de nível de líquidos de um tanque apenas. Esse sistema possui o comportamento de sistema de 1ª ordem. Entretanto apresenta características não-lineares, o que faz o mesmo ser um bom candidato para validação da técnica proposta.

São, também, vários os trabalhos de investigação e bibliografias que modelaram exaustivamente esse sistema GOSMANN (2002), SAGAZ (2003), AGUIRRE (2004), BERNARDES et al. (2006), ANDRADE (2008), RIKER Jr (2008), ACHY (2012) e que as utilizaram em diversas técnicas de controle.

Além disso, este tipo de modelo de sistema é também bastante comum na indústria como, por exemplo, na fabricação de refrigerantes em misturadores de gás e xarope, e na indústria petroquímica em colunas desbutanizadora (PADILHA, 2001).

4.2 DESCRIÇÂO DO SISTEMA

O projeto proposto prevê o uso de um tanque, posicionados verticalmente sobre um sorvedouro (ou tanque pulmão). A Figura 4.1 mostra o sistema de forma simplificada.

O transporte de água para o tanque é realizado por 3 bombas de esguicho d'água de corrente continua atrelado a tubulações de transporte de líquidos que adquire líquido do tanque auxiliar, que realiza o papel de sorvedouro. Para o controle da vazão, utilizamos o controle de velocidade do motor para gerar vazões proporcionais. Neste trabalho optou-se pela utilização destas últimas, pois, podem atuar de forma mais precisa no controle da vazão, sem precisar "ligar e desligar" os dispositivos de forma intermitente.

O projeto também utiliza um sensor de vazão para a realização da quantidade de líquidos que alimenta o tanque.

Devido sua configuração simples, pode-se modelar facilmente a dinâmica do tanque. Optou-se por realizar um modelo baseado na física do processo para poder proceder no controle do mesmo.

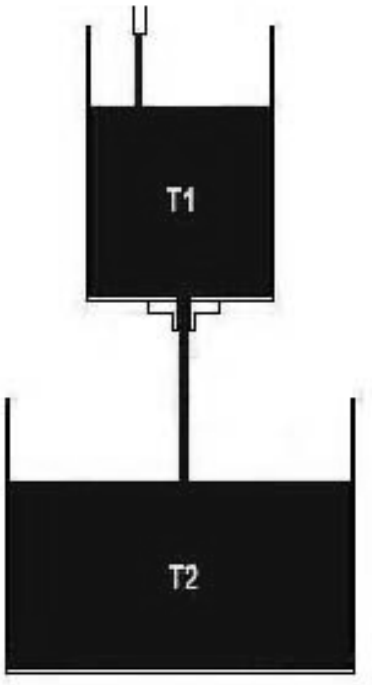


Figura 4.1. Proposta de Configuração do Projeto.

4.3 MODELAGEM MATEMÁTICA

O modelo será demonstrado de forma breve e resumida, para haver uma linha de raciocínio concisa, apesar do estudo teórico das equações básicas sobre este trabalho já ter sido demonstrado em OGATA (1998), AGUIRRE (2004), ANDRADE (2008) e RIKER Jr (2008).

Para que possamos trabalhar em um sistema dessa natureza, de início, devemos considerar que o fluido seja genérico e o sistema ideal. Isto significa que consideramos o fluido incompreensível e que não há perdas de carga nas tubulações do sistema. O sistema deve ter parâmetros concentrados e somente as variáveis relevantes entram no modelo.

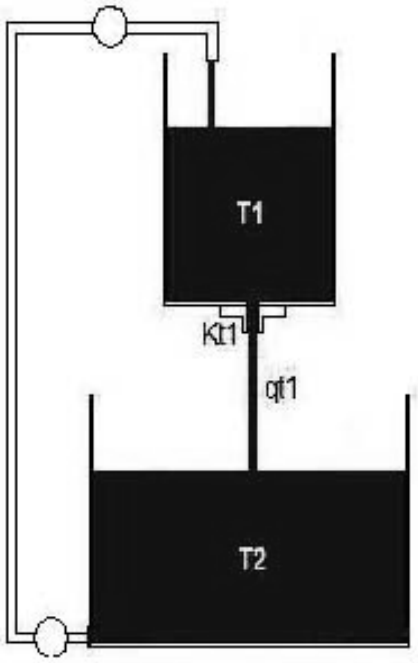


Figura 4.2 - Modelo esquemático do protótipo.

Os sistemas de nível de líquidos devem ser restringidos quanto ao modelo com base no Número de Reynolds (OGATA, 1998), onde se determina o regime de escoamento do fluido do sistema. Este regime se distingue em escoamento laminar e escoamento turbulento. Nas aplicações industriais dificilmente é encontrado algum sistema que faça uso de fluxo de líquidos em regime de escoamento laminar.

Entretanto, neste trabalho, consideramos que o sistema utilize o regime do escoamento laminar devido escala do protótipo, onde as vazões são pequenas comparadas aos processos industriais, e possibilidade da descrição dos fenômenos físicos do sistema ter forma simplificada. Desse modo, considera-se também, devido a escala diminuta do protótipo, que o escoamento turbulento do protótipo não traga grandes diferenças nos resultados finais.

O objetivo principal do sistema é a realização do controle do nível do tanque. A Figura 4.3 mostra a configuração do sistema de nível de liquido utilizada neste trabalho e as respectivas incógnitas do sistema. Aqui, o sistema apresenta um tanque onde a vazão é realizada por um conjunto de 3 bombas iguais, onde suas vazões variam conforme a tensão. Estas tem a finalidade de controlar e equilibrar a vazão de saída qv1 que possuem coeficiente de resistência hidráulica, kv1, variando conforme o nível do tanque. Isso ocorre devido a pressão hidrostática no fundo do tanque variar conforme o nível. A pressão altera-se linearmente conforme a Equação 4.11. Entretanto, a relação entre o nível do tanque e a vazão de saída não é linear, descrevendo uma curva como função que será mostrado na Equação 4.13. O tanque T1 possui uma vazão natural proporcionada por orifício com restrição constante.

Há trabalhos em que foi considerado um modelo com quatro válvulas como em RIKER Jr (2008) e ANDRADE (2008), sendo duas utilizadas como saída para dois tanques. Há outras modelagens com válvula na saída como em SAGAZ (2003), AGUIRRE (2004), BERNARDES et al (2006) e em BERNARDES e MELO (2006). A válvula na saída acarreta na modificação das características das restrições de saída de cada tanque, consequentemente na complexidade do sistema, uma vez que válvulas produzem perdas de carga que deixam o sistema ainda mais complexo. Embora seja necessário estudar a dinâmica dessas restrições para sistemas deste tipo, este trabalho não visa a obtenção de modelos mais complexos. O objetivo é apenas implementar a técnica proposta por este trabalho. Isso justifica o uso de um modelo mais simples que utiliza um orifício de saída com diâmetro fixo, já que deve possuir características que deverá ser conhecida para o controle do sistema. A identificação dos parâmetros referentes à restrição de saída será abordada adiante.

O ponto inicial do trabalho se dá através da modelagem matemática do sistema relacionando essas variáveis para o entendimento da natureza do processo. Nos sistemas onde há troca de materiais, é necessário equacionar somente as variáveis em alguns pontos específicos do processo, considerando-os parâmetros concentrados (AGUIRRE, 2004).

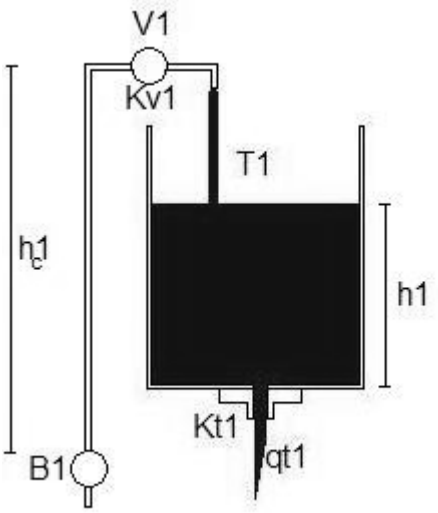


Figura 4.3 - Representação das variáveis do tanque 1. (ANDRADE, 2008).

Consideramos tanque T1, com vazão natural de líquido, q_{t1} (cm^3/s), como mostra a Figura 4.3. Para obter certo nível em T1, é necessário que haja uma entrada de volume de líquido, q_{v1}. Logo, a troca de material neste tanque pode ser equacionada através da equação de balanço ou troca de volume, considerando que sempre haverá a conservação do volume de líquido no sistema, que corresponde a soma de todas as vazões de entrada de líquido subtraído da soma de todas as vazões que saem do tanque. Generalizando, tem-se

$$\frac{dv}{dt} = \sum qi - \sum qo \tag{4.1}$$

E para o tanque T1,

$$\frac{dv}{dt} = q_{v1} - q_{t1} \tag{4.2}$$

A variável *v* corresponde ao volume de água que permanece no tanque. Pode-se observar que, para este sistema, torna-se mais conveniente e intuitivo

estimar a quantidade volumétrica de líquido que outra grandeza, como a massa de líquido, por exemplo. Além disso, é mais comum lidarmos cotidianamente com grandezas volumétricas de líquido.

Sabendo que,

$$v = A.h \tag{4.3}$$

Substituindo, temos

$$A.\frac{dh}{dt} = q_{v1} - q_{t1} \tag{4.4}$$

e

$$\frac{dh}{dt} = \frac{q_{v1} - q_{t1}}{A} \tag{4.5}$$

Desse modo, o controle pode simplesmente ser realizado com referência à altura da coluna de líquido do tanque uma vez que se torna fácil e rápido mensurar uma grandeza unidimensional.

Observe que q_{v1} e q_{t1} são, respectivamente, vazão de entrada e vazão de saída do tanque T1, em cm^3/s. Logo, obteve-se um modelo que relaciona vazão de entrada e de saída com a altura do tanque, e que a Equação 4.5 representa uma equação de estado do sistema, onde *h* é uma variável de estado. Porém ela ainda não explica de forma explicita o comportamento dinâmico do tanque 1.

Isso significa que se deve encontrar em q_{t1} e q_{v1} equações que se relacionem com *h*. Tem-se que as vazões q_{t1} e q_{v1} estão diretamente relacionadas com o coeficiente hidráulico k_{t1} e k_{v1} respectivamente. Ora, ao se alterar o coeficiente hidráulico, alteram-se as características da restrição, consequentemente, a vazão. Em OGATA (1998), identifica-se duas situações relevantes. A primeira refere-se ao regime laminar de escoamento do líquido onde

$$q_{t1} = k_{t1}.h \tag{4.6}$$

e segue que

$$k_{t1} = \frac{q_{t1}}{h} \tag{4.7}$$

.

No regime laminar, a relação entre vazão de saída e altura é constante o que torna o sistema bastante linear e de simples resolução. A segunda consideração, é a de que o regime de escoamento do fluido é turbulento que, devido a perda de carga na restrição, pode ser definido através da Lei de Bernoulli,

$$q_{t1} = k_{t1}\sqrt{\Delta P} \tag{4.8}$$

onde ΔP é o diferencial de pressão na restrição. Logo,

$$q_{t1} = k_{t1}\sqrt{P2 - P1} \tag{4.9}$$

sendo $P1 = P_{atm}$, já que é a extremidade de fora da restrição

$$q_{t1} = k_{t1}\sqrt{P2 - P_{atm}} \tag{4.10}$$

E P2 coincidindo com o fundo do tanque, tem-se que

$$P2 = \rho.g.h + P_{atm} \tag{4.11}$$

e

$$q_{t1} = k_{t1}\sqrt{\rho.g.h + P_{atm} - P_{atm}} \tag{4.12}$$

Logo,

$$q_{t1} = k_{t1}\sqrt{\rho.g}\sqrt{h} \tag{4.13}$$

Onde ρ é a massa específica da água e g é a gravidade. P_{atm} é a pressão atmosférica.

Para o sistema de nível de líquidos, é comum utilizar o regime de escoamento turbulento, pois há perdas de cargas decorrentes das restrições, em caso de distúrbios no sistema, como enchimento em vazão máxima ou turbilhonamento na vazão de saída.

A vazão da válvula V1, q_{v1}, também deve ser modelada de forma que possa gerar uma função da altura h do tanque T1. Porém, é importante lembrar que a válvula V1 é um dispositivo elétrico e depende de outra variável para seu acionamento, no caso, a tensão. Sua característica é realizar a variação da restrição, variando o coeficiente hidráulico que corresponda a valores de vazão, q_{v1}. Em princípio seu funcionamento pode ser também estudado pela Lei de Bernoulli, pois, a configuração em que foi instalada pode ser comparada a uma restrição comum. De igual maneira, temos que:

$$q_{v1} = k_{v1}\sqrt{\Delta P} \tag{4.14}$$

onde ΔP é o diferencial de pressão na restrição. Logo,

$$q_{v1} = k_{v1}\sqrt{P2 - P1} \tag{4.15}$$

Sendo, $P1 = P_{atm}$, já que é a extremidade de fora da restrição

$$q_{v1} = k_{v1}\sqrt{P2 - P_{atm}} \tag{4.16}$$

Entretanto, neste caso, P2 é a pressão do encanamento no nível da válvula, “sentida” na extremidade da mesma e sua direção é oposta a da gravidade. Como existe uma coluna de água de altura hc1, tem que:

$$P2 = Pb - \rho.g.hc - P_{atm} \tag{4.17}$$

e

$$q_{v1} = k_{v1}\sqrt{Pb - \rho.g.hc - 2.P_{atm}} \tag{4.18}$$

Onde Pb é a pressão na saída da bomba

Observa-se que o termo dentro da raiz é constante e que kv1 é variável e corresponde a uma função da tensão. Como qt1 e qv1 foram encontradas, pode-se substituir (4.15) e (4.18) em (4.5). Logo,

$$\frac{dh}{dt} = \frac{k_{v1}\sqrt{Pb - \rho.g.hc - 2.P_{atm}} - k_{t1}\sqrt{\rho.g}\sqrt{h}}{A} \tag{4.19}$$

A Equação 4.19 descreve a dinâmica do tanque T1, em termos de h instantâneo e que corresponde a equação de estado do respectivo tanque. A Equação 4.19 não explicita diretamente uma função de h do tanque 1, porém nos entrega a possibilidade de influenciar dh/dt através do fator kv1, que pode ser expressão por uma função da tensão, que constitui um elemento importante para a realização do controle. Adiante, será disponibilizada uma atenção especial sobre kv1.

4.3.1 LINEARIZAÇÃO POR APROXIMAÇÃO DA SÉRIE DE TAYLOR

Uma solução simples é realizar uma linearização aproximada em torno de um ponto de operação através da Série de Taylor.

A forma geral da série é definida por:

$$f(x) = f(a)(x-a)^0 + \frac{f'(a)(x-a)^1}{1!} + \frac{f''(a)(x-a)^2}{2!} + \ldots + \frac{f^{(n)}(a)(x-a)^n}{n!}. \tag{4.20}$$

A linearização para o tanque pôde ser realizada através de uma aproximação de primeira ordem da Série de Taylor. A dedução do modelo linearizado pode ser consultada em RIKER Jr (2008). Considerando que qv1 (consequentemente kv1) são também lineares em um ponto de operação, a Equação 4.21, correspondente ao tanque 1, na forma linearizada, é escrita na forma:

$$\frac{dh_1}{dt} = \frac{qv1}{A} - k1.\frac{h_1}{2.A.\sqrt{ho1}} \tag{4.21}$$

Onde,

$$h_1 = ho1 + \delta h \tag{4.22}$$

que h1 é a nova variável de estado e significa uma pequena excursão em torno do ponto de operação ho1.

4.3.2 DISCRETIZAÇÃO DO MODELO

Em projetos modernos de controle, para a realização de um processo de controle supervisionado, é necessária a implementação em circuito que possa realizá-la de forma coerente e precisa. É conveniente o uso de um circuito controlador digital para aquisição e processamento de sinais. Isso implica em transformar a Equação 4.21 em um modelo discreto de representação de forma que se torne prática a manipulação microprocessada da lógica digital.

O primeiro passo é realizar a obtenção da função de transferência a partir da Equação 4.19. A função de transferência já linearizada é mostrada na Equação 4.23 abaixo.

$$G(s) = \frac{1/A}{s + \frac{k_1}{2.A.\sqrt{h_{o1}}}} \tag{4.23}$$

Pode-se discretizar tal equação utilizando diversos métodos como método de Tustin e segurador de ordem zero (Zero Order Holder). Este última é mais simples e mais popular (PARASKEVOPOULOS, 1996), e pode-se modelar o sistema com mais rapidez. É importante ressaltar que sua dinâmica lenta em comparação com a resposta dos atuadores e que se assemelha a um sistema de primeira ordem, o que não exige-se utilizar grandes períodos de amostragem (AGUIRRE, 2004). Entretanto, optou-se por utilizar um período de amostragem relativamente rápido, de 300 ms, pois o tanque do protótipo possui uma altura útil pequena e a vazão de entrada possui um regime turbulento de escoamento, implicando a incidência de grande variações espontâneas que devem ser capturados pelo circuito.

O método do segurador de ordem zero (Zero Order Hold - ZOH) consiste em colocar um amostrador fictício em cascata com o sistema contínuo conforme a figura 4.4. O amostrador é dado pela Equação 4.24.

$$G_{zoh}(s) = \frac{1 - e^{-sT}}{s} \tag{4.24}$$

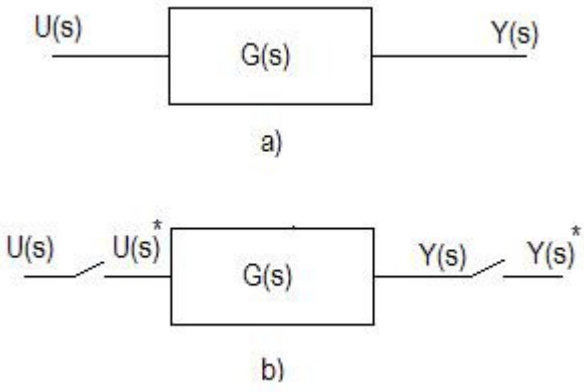


Figura 4.4: Amostragem fictícia de G(s). (NISE, 2000).

$$G(z) = G_{zoh}(s)G(s) = \frac{1-e^{-sT}}{s}G(s) \tag{4.25}$$

Ou

$$G(z) = 1 - e^{-sT}.Z\left\{\frac{G(s)}{s}\right\} \tag{4.26}$$

Aplicando em (4.25) temos:

$$g(z) = (1-z^{-1}).Z\left\{\frac{1}{s}\frac{1/A}{s+\frac{k1}{2.A.\sqrt{h_{o1}}}}\right\} \tag{4.27}$$

E fazendo o mapeamento do plano S para o plano Z, temos:

$$H(z) = \frac{1/A}{1-z^{-1}\exp(\frac{-k1}{2.A.\sqrt{h_{o1}}})}U(z) \tag{4.28}$$

A Equação 4.28 é a discretização da Equação 4.23. Como o objetivo é encontrar equações discretas no tempo, realizamos os cálculos da Transformada Z inversa para cada equação separadamente. Assim, para o tanque 1, após a organização das variáveis e convertendo-a em equação de diferenças, temos que:

$$h_1(k) = \frac{u(k)}{A} + \exp\left(\frac{-k1}{2.A.\sqrt{h_{o1}}}\right).h_1(k-1) \tag{4.29}$$

A Equação 4.29 mostra a dinâmica do tanque 1 através de uma equação de diferença.

4.4 CONSTITUIÇÃO DO PROTÓTIPO DE NÍVEL DE LÍQUIDOS

A Figura 4.5 mostra o sistema em diagrama de blocos do sistema.

O controlador adquire um valor de referência na entrada e realiza uma comparação do valor da referência com o valor adquirido pelo sensor de nível. Assim, o sistema ajusta a tensão de entrada das válvulas conforme o erro.

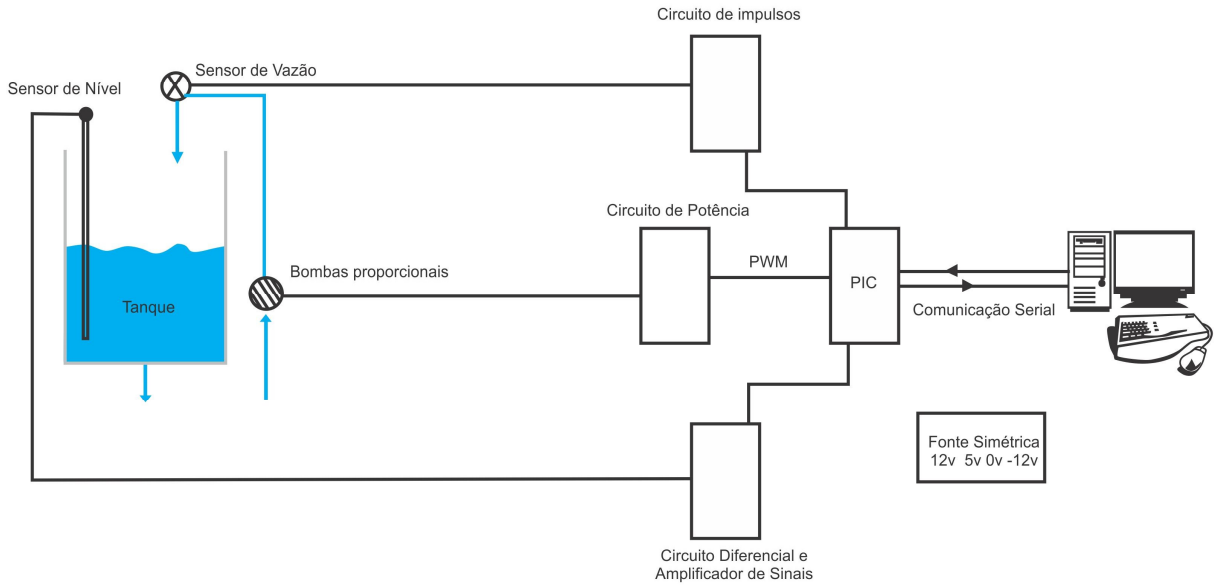


Figura 4.5. Diagrama em blocos do sistema. (Andrade, 2008)

O projeto é composto por dois tanques de plástico, um medindo 100 mm de altura sobre uma base de 100,3 mm² e outro medindo 70 mm de altura e 29400 mm² de base. Estes tanques estão dispostos um sobre o outro, separadamente, e se comunicam através de um orifício na base do tanque menor para permitir o transporte de líquido de um para o outro. Os tanques são apoiados através de um pequeno suporte sobre o sorvedouro. O tanque de maior volume é utilizado como sorvedouro (ou tanque pulmão) para suprir a quantidade de líquido necessário para o funcionamento da planta e localiza-se abaixo do tanque principal.

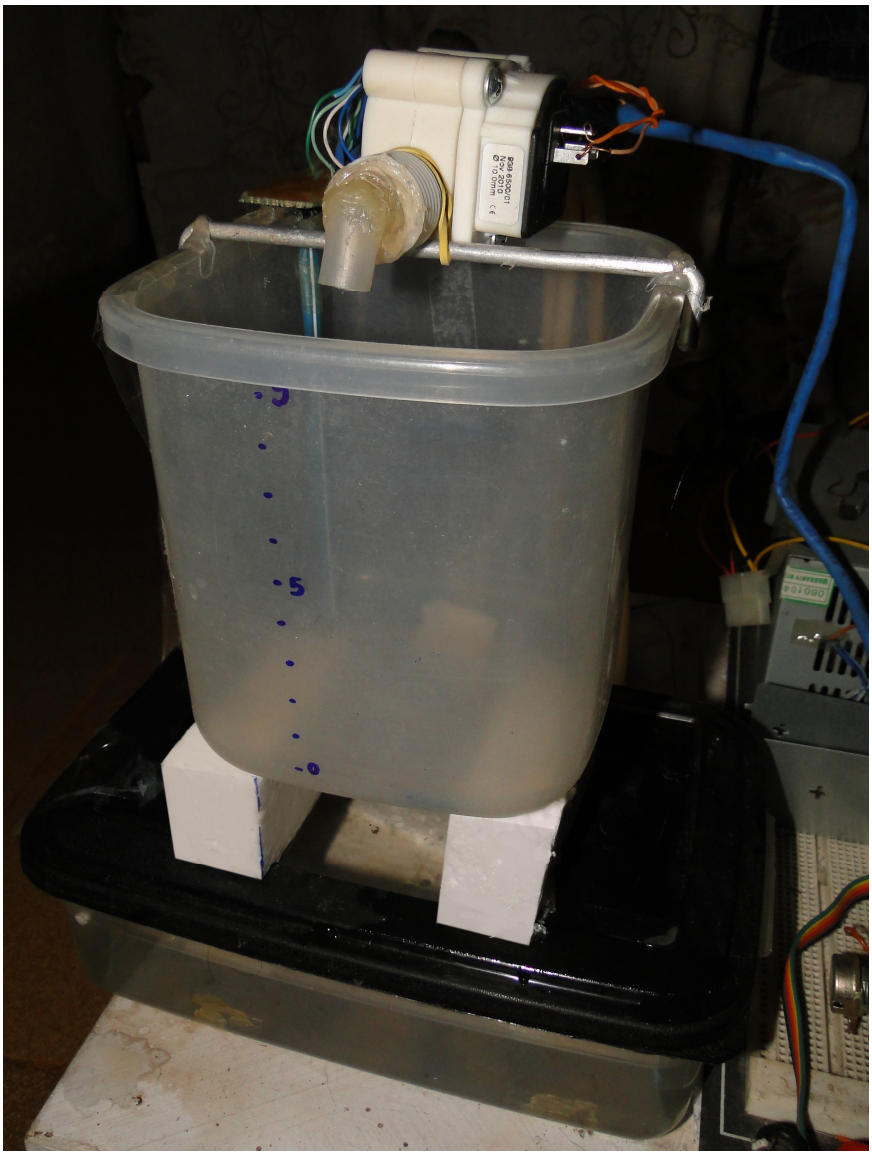


Figura 4.6. Tanque principal e sorvedouro do protótipo.

Para realizar o controle de vazão nas entradas dos tanques, o transporte de líquido é realizado através de três bombas de esguicho d'água. Em ANDRADE (2008), são utilizadas eletroválvulas proporcionais de corrente contínua de 24 V - 10 W da marca Burkert modelo 6213 para pressão máxima de 10 bar. A diferença deste projeto para o anterior, descrito em ANDRADE (2008), é a supressão desta válvula já que o conjunto de motores pode fornecer vazão variável ao tanque. A grande vantagem é que elas podem funcionar também com Modulação por Largura de Pulso – PWM – utilizando a tensão efetiva conforme o ciclo de trabalho (Duty Cicle) possibilitando estratégias de controle no uso com microcontroladores. As válvulas foram suprimidas para evitar a cavitação dos motores através da alta pressão submetida quando era exigida uma vazão muito pequena do sistema.

4.4.1 ACIONAMENTOS

4.4.1.1 BOMBAS

O sorvedouro comporta líquido para o tanque posicionado acima dele. Esse transporte é realizado através de três bombas de esguicho d'agua que bombeiam líquido para o tanque. Em BERNARDES et al. (2006) são utilizadas bombas semelhantes. As bombas são instaladas em paralelo para reagirem à mesma mudança de tensão durante o controle. As bombas são acionadas por corrente contínua com tensão máxima de 12 volts. O comando de acionamento e controle é realizado pelo microcontrolador utilizando um circuito de potência para corrente continua através de uma proteção óptica. As bombas são ligadas a uma fonte auxiliar dedicada e chaveadas através de um Transistor de potencia TIC122. O optoacomplador é o 4N25 e é utilizado para evitar a inversão de corrente para a parte de baixa potência do circuito, além de bloquear interferências parasitas da fonte.

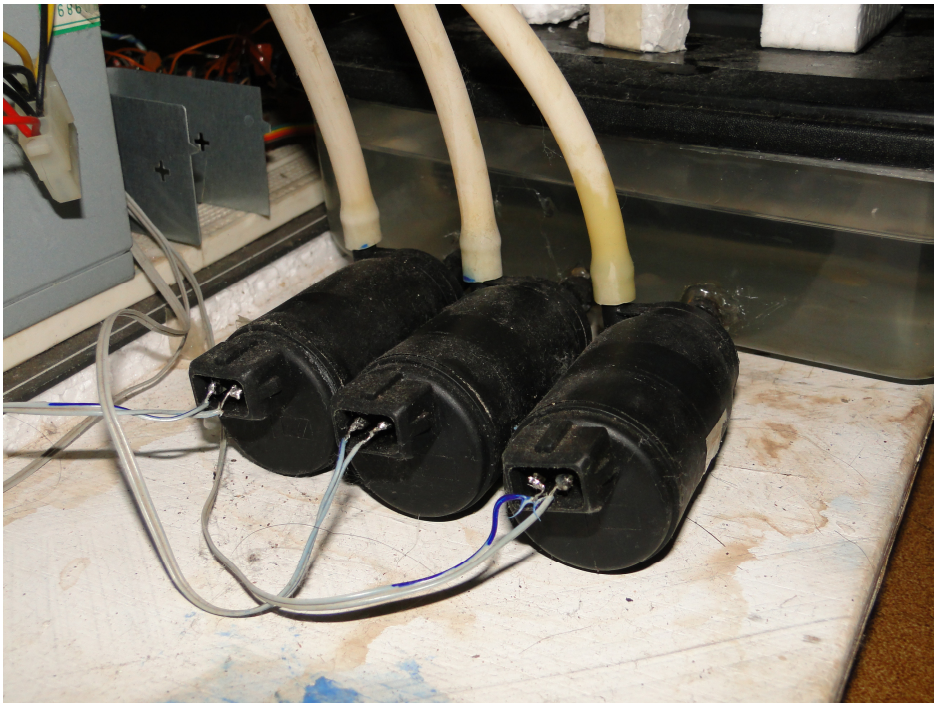


Figura 4.7 - Motobombas de esguicho.

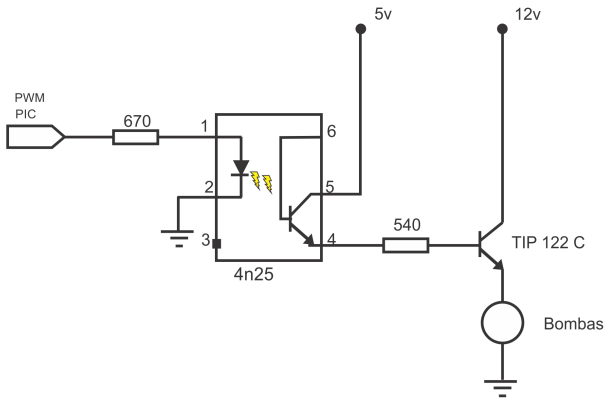


Figura 4.8 - Circuito de acionamento das motobombas.

A Figura 4.7 mostra como as bombas foram dispostas e se comunicam com o sorvedouro. A Figura 4.8 descreve o circuito de acionamento das bombas. O protótipo do SICONILI utiliza o conjunto de acionamento de bombas através do comando do microcontrolador PIC 30F2011. No Anexo 6, encontra-se o desenho da placa e o circuito completo.

4.4.2 INSTRUMENTAÇÃO

4.4.2.1 SENSOR DE VAZÃO

O Medidor de Vazão FHKU é um dispositivo de uso geral especialmente que suporta fluxos bem elevados de até aproximadamente 30 l / min (FLOW SENSOR DATASHEET). É empregado para a medição ou regulação e garante medição mais precisa de quantidades de fluidos. Além disso, um gerador de impulsos integrado no fluxômetro garante uma vida útil praticamente ilimitada. As características de destaque deste dispositivo são a capacidade de suportar alta temperatura e boa resistência a produtos químicos. Possui entrada e saída bastante linear e design compacto.

Os valores especificados devem ser considerados como valores aproximados. O número de impulsos por litro pode diferir dependendo do meio e instalação. O Fabricante sugere calibrar o número de impulsos por litro durante a instalação.

Este sensor gera como saída um trem de pulsos cuja largura de cada pulso determina a vazão atual que passa pelo sensor. A frequência do trem de pulso é diretamente proporcional à vazão. Isto significa que para vazões pequenas, maior será a largura entre os pulsos. A Figura 4.9 exemplifica tal comportamento onde, em “a”, a vazão é bem menor que em “c”.

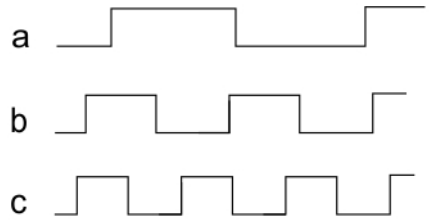


Figura 4.9. Gráficos de saída do sinal do sensor de vazão

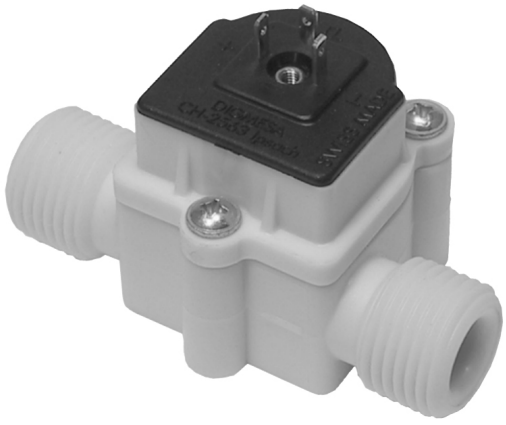


Figura 4.10. Sensor de Vazão DIGMESA

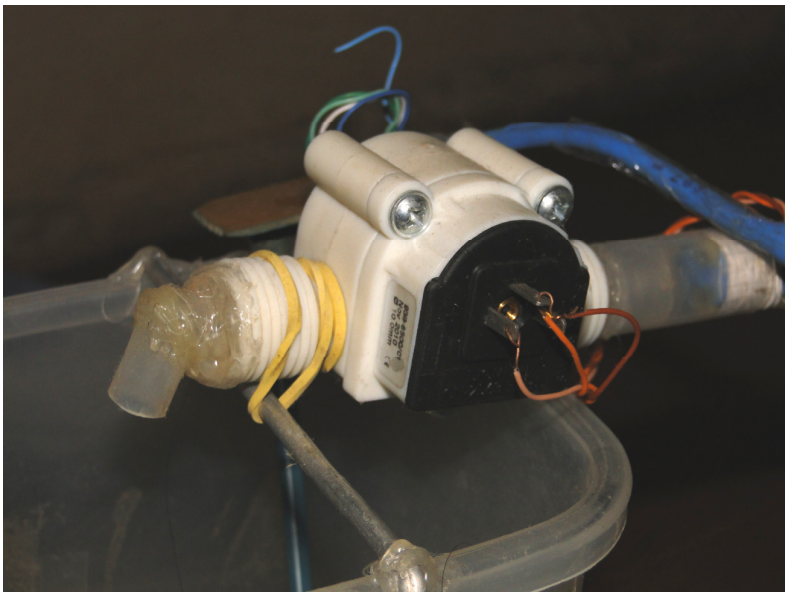


Figura 4.11 - Sensor de Vazão DIGMESA instalado no Protótipo.

Para o condicionamento de sinal pulsado, foi utilizado um circuito simples, compatíveis com as tensões TTL utilizadas pelo dsPIC 30F2011.

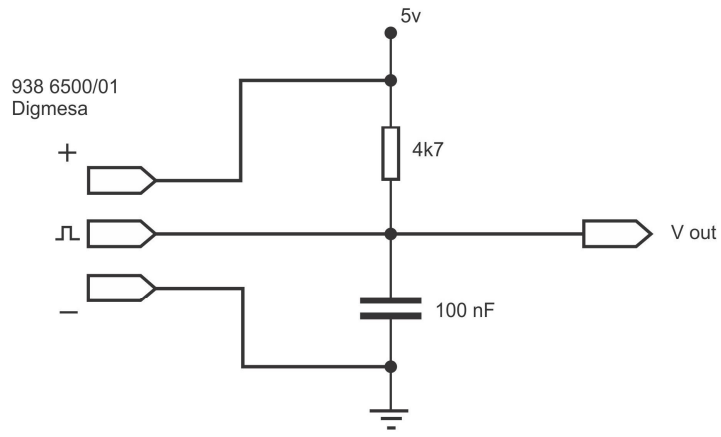


Figura 4.12 - Circuito de condicionamento do sensor de vazão.

4.4.2.2 CÁLCULO DE VAZÃO DO SENSOR.

Tabela 4.1 - Especificações do sensor de vazão

Tamanho do bico	Pulso/Litro	g/pulso	Vazão minima em litros/ min no início linear	Vazão máxima em litros / min	Perda de Pressão
Ø 10.00 mm	65	15.37	3.00	26.69	0.32

A partir das especificações da Tabela 4.1, extraída do *datasheet* do componente (FLOW SENSOR DATASHEET), pode-se mensurar com precisão os dados obtidos pelo sensor.

A informação mais importante é grandeza estimada de pulsos por litro do sensor, que é padronizada em 65 pulsos/litro. Isto significa que o sensor detecta 15,38 ml para cada pulso. Como a frequência do trem de pulso é diretamente proporcional à vazão (Figura 4.9), para realizar a medição, é necessário contar o tempo em que o pulso está ativo, isto é, o tempo entre duas bordas de subida (ou descida). Neste caso, foi utilizado um dos *timers* do microcontrolador, com incremento de 128 µs (Anexo 1 e Anexo 5). Quando a borda de um pulso é detectada na porta, inicia-se uma contagem com incremento a cada 128 µs até que a porta receba outra borda igual de outro pulso. Em seguida, o programa executa a Equação 4.30, reinicia o contador e aguarda novos pulsos.

$$Q(t) = 15.38/(t * 0.000128) \quad (4.30)$$

4.4.2.3 SENSOR DE NÍVEL

Para a medição de níveis de líquidos há diversos tipos de sensores, cada um apresentando suas características, vantagens e desvantagens. Neste trabalho, desejou-se utilizar um método capaz de realizar a medida dos níveis dos líquidos sem a necessidade de inspeção humana e que pudesse gerar um sinal de resposta de boa precisão que possa ser utilizada para controle do processo. O método que utilizamos nesse trabalho é o método indireto de medição que utiliza outra grandeza que se relaciona com o nível de líquido a ser medido.

Neste caso, os sensores piezoresistivos, além de ser o principal representante, são muito confiáveis.

O sensor utilizado neste trabalho é o MPXM2010GS como mostra a Figura 4.13. A motivação para utilização deste sensor é sua versatilidade. É um sensor de baixo custo e de tamanho reduzido, podendo ser utilizado de diversas maneiras. Outro fator é a precisão, pois apresenta uma alta resolução e larga faixa de linearidade. A Universidade de Brasília (UNB), através do LAVSI (Laboratório de Automação, Visão e Sistemas Inteligentes), já utilizou este sensor em vários projetos (BERNARDES et al., 2006) e consideramos que é uma boa fonte de referência para uso do mesmo.

O método de medição utilizado neste projeto é denominado de “ar no cano”, como mostra a Figura 4.14. A extremidade de uma cânula é fixada no orifício do sensor enquanto a outra extremidade permaneça aberta. A cânula é mergulhada na água de modo vertical, como mostram as Figuras 4.15 e 4.16. Sendo o nível da cânula o mesmo do nível do tanque, pressão exercida pelo ar dentro do cano causa uma tensão de saída no sensor que é proporcional ao nível do tanque.

O princípio para a medição de nível de líquido através da pressão é a diferença de pressão entre um ponto de referência e o ponto de medição. No caso deste projeto, o ponto de medição é a altura do líquido acima do ponto de referência do líquido. A pressão neste ponto também depende diretamente da densidade do líquido.

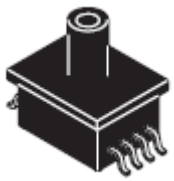


Figura 4.13: Sensor MPXM2010GS. (FREESCALE, 2007)

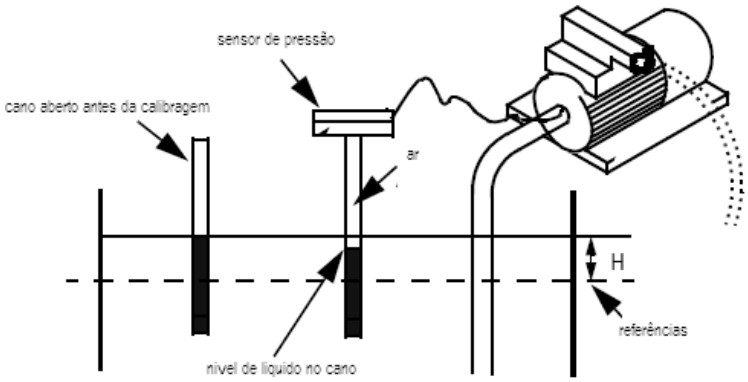


Figura 4.14: Monitoramento de nível. (FREESCALE, 2007.)

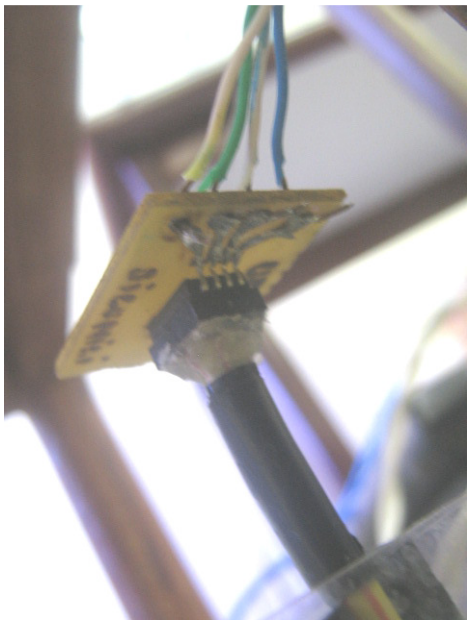
4.4.2.4 CARACTERÍSTICAS DO SENSOR DE PRESSÃO

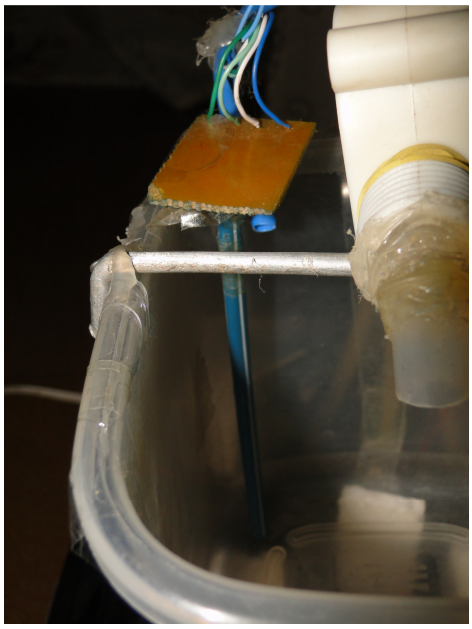
O sensor MPXM2010GS é um sensor de elemento piezoresistivo, implementado internamente em uma tradicional ponte de Wheatestone e com offset calibrado através de laser.

É alimentado com tensão máxima de 16 V e sua saída é composta por uma tensão diferencial de, no máximo, 25 mV quando existir uma taxa máxima de pressão sobre sensor. Esta pressão máxima suportada está em torno de 10 kPa.

Como a tensão de saída é muito pequena, especialmente para aplicações que utilizem frações desta tensão máxima de saída, deve ser implementado junto com um amplificador para que o sinal de saída seja utilizável para o fim que se deseja. Utiliza-se um amplificador diferencial, conhecido também como amplificador de instrumentação, cujo circuito é mostrado na Figura 4.18, para conseguir amplificar o sinal de 25 mV para 5 V que será adquirido pelo microcontrolador.

Consideramos que o circuito de amplificação possui linearidade suficiente para transpor os níveis de tensão de forma proporcional.





Figuras 4.15 e 4.16: Implementação do sensor no tanque. (ANDRADE, 2008)

4.4.2.5 CÁLCULO PARA O CIRCUITO DE CONDICIONAMENTO E SENSOR

Se uma cânula é colocada verticalmente na água, com uma das extremidades aberta para fora, o nível da cânula será exatamente o nível da água. Entretanto, se a extremidade externa da cânula for fechada, o volume de ar estará enclausurado e a pressão irá variar proporcionalmente com o nível da água.

Este sensor possui uma sensibilidade de 2mV/kPa. Sabendo que a pressão varia de 0,09806 kPa para cada 10 mm, para medir uma coluna de água de 25cm, equivalente a altura do tanque, tem-se:

$$2.0\ \text{mV/kPa} \times 10 \times 0{,}09806 = 1{,}9612\ \text{mV}. \qquad (4.31)$$

Caso se utilize circuitos com sensibilidade TTL ou microprocessados, o máximo de valor admitido para condicionamento é de 5V. Assim, o ganho do circuito condicionador de sinal é dado por:

$$G = 5\text{V}/(1.9612\ \text{mV}) \qquad (4.32)$$
$$G = 2549{,}45.$$

Pode-se arredondar este valor de ganho para G = 2550.

O amplificador de instrumentação é utilizado quando necessitamos amplificar sinais diferenciais de entrada. São considerados os mais úteis e versáteis circuitos com amplificadores operacionais para medições de precisão e controle de processos (FREIRE, 2002) e (SEDRA e SMITH, 2000). É constituído por uma entrada diferencial e uma realimentação por ganho de tensão onde consegue amplificar sinais muito pequenos de tensão superpostos a grandes tensões de modo comum ou ruídos de modo comum, onde estas últimas se anulam.

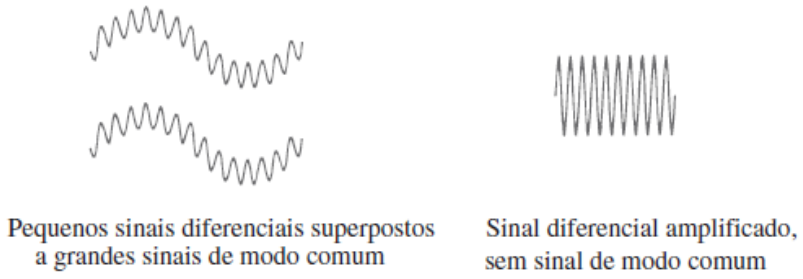


Figura 4.17: Condicionamento de sinais diferenciais. (FREIRE, 2002)

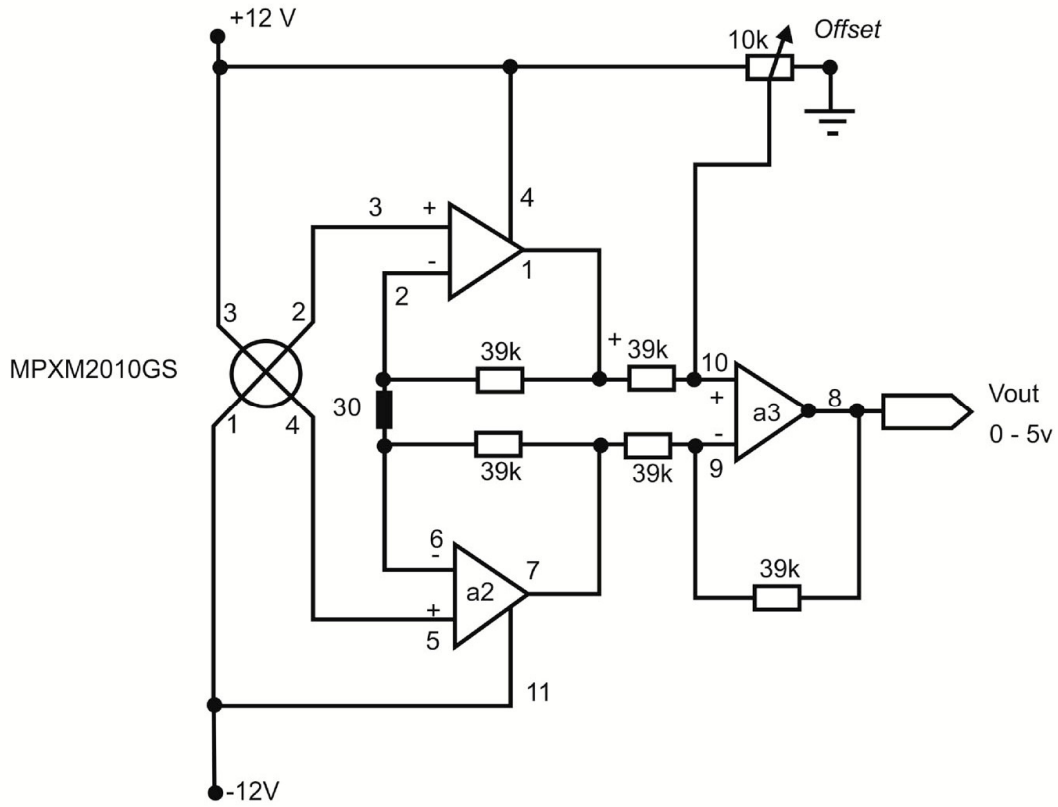


Figura 4.18. Circuito de condicionamento de sinal. (ANDRADE, 2008).

A equação abaixo nos mostra que o cálculo para este circuito é simples, através do ganho em função do resistor Rg. Utilizando todos os resistores iguais a 39kΩ e o ganho de 2550, podemos calcular Rg facilmente. Tem-se:

$$Av = 1 + \frac{2.R}{Rg} \tag{4.33}$$

$$2550 = 1 + \frac{2x39k}{Rg} \tag{4.34}$$

$$Rg = 30{,}60\Omega$$

O resistor com o valor comercial aproximado do valor calculado é de:

$$Rg = 30\Omega \tag{4.35}$$

4.4.3 LEVANTAMENTO DE CURVAS E DAS CONSTANTES DO SISTEMA

4.4.3.1 ESTIMAÇÃO DE PARÂMETROS

AGUIRRE (2004) refere-se à identificação de sistemas como sendo uma área do conhecimento que estuda maneiras de modelar e analisar sistemas a partir de observações. Em outras palavras, dados. Como visto no capítulo anterior, existem parâmetros do sistema que precisam ser estimados para que se possam validar os resultados teóricos com os práticos, além de possibilitar os projetos simulados de novos controladores para serem aplicados e testados na planta real.

Em OGATA (1998), afirma-se que em muitos casos práticos os valores dos coeficientes hidráulicos são desconhecidos. Uma técnica que pode ser utilizada é levantar uma curva entre a vazão de entrada e vazão de saída, relacionando com os sinais de controle e a altura de líquido no tanque, respectivamente, onde se calcula a inclinação da reta no ponto ou trecho de operação.

Para isso, é necessária a realização de alguns ensaios para obtenção de duas curvas características que determinem as características da entrada e as características da saída. Assim, pode ser realizado para determinar um vetor de entrada X com a saída Y. Na determinação das características da entrada, para cada valor de sinal de controle, medimos o valor de vazão de entrada em regime permanente. Na determinação das características da saída, para cada valor de vazão de entrada, medimos o valor da saída em regime permanente. Os ensaios são realizados em malha aberta. As variáveis X e Y se relacionam da forma Y = f(X). Assim, em f(X), podem ser encontrados diversos parâmetros que validarão a própria função.

Realiza-se tal procedimento, para as variáveis do sistema, a fim de se comparar com o sistema simulado. O sinal em azul mostra o nível em regime permanente, o sinal em lilás mostra o valor da vazão e o sinal em preto mostra o sinal de controle PWM. Os gráficos dos ensaios reais são mostrados a seguir:

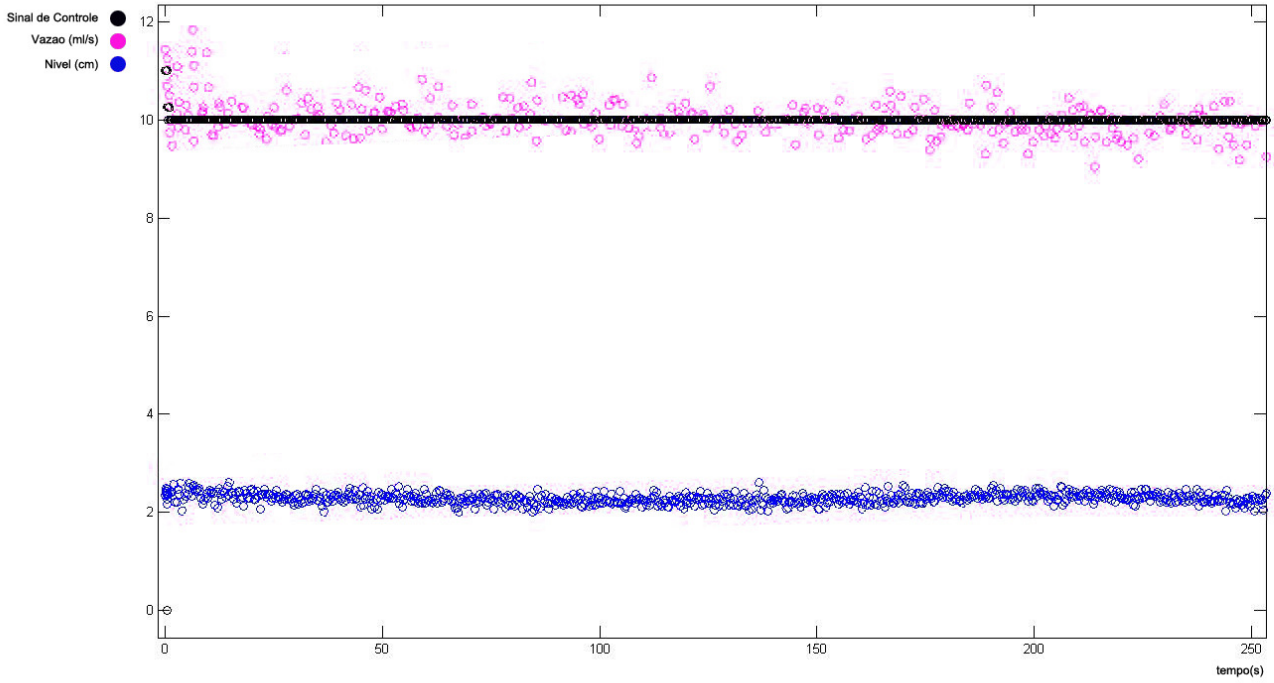


Figura 4.19. Resposta do sistema com PWM com *duty cucle* de 32%

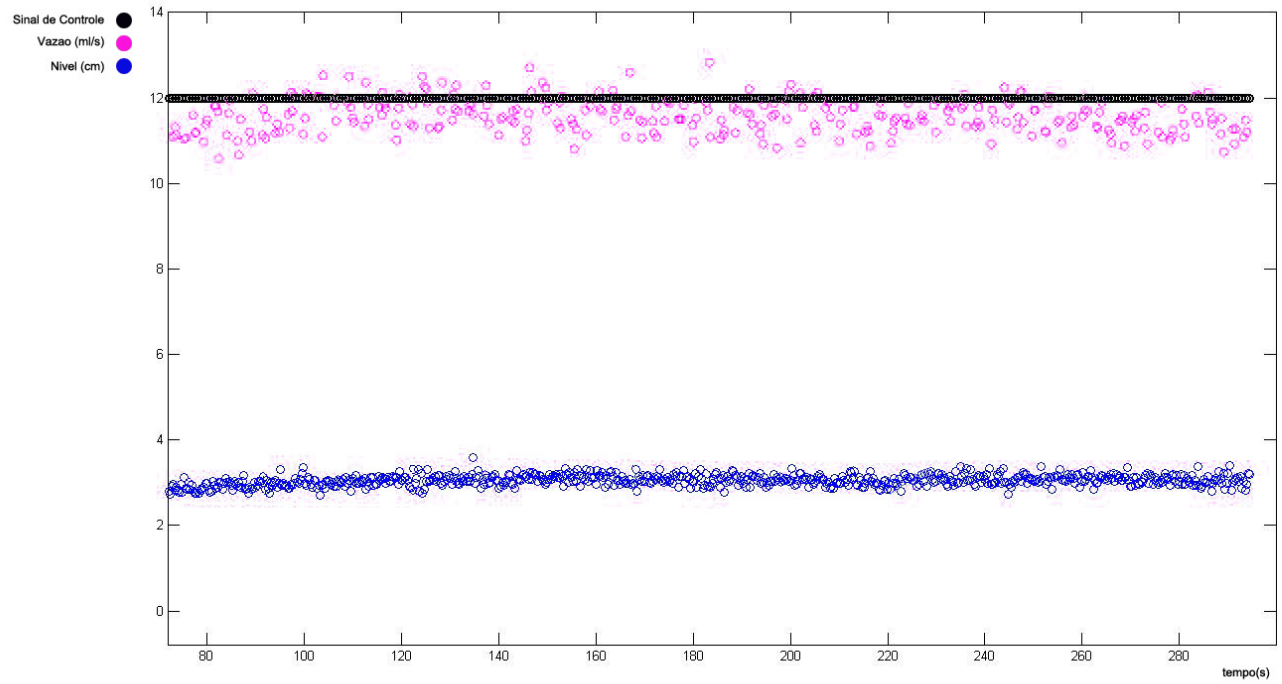


Figura 4.20. Resposta do sistema com PWM com *duty cucle* de 40%

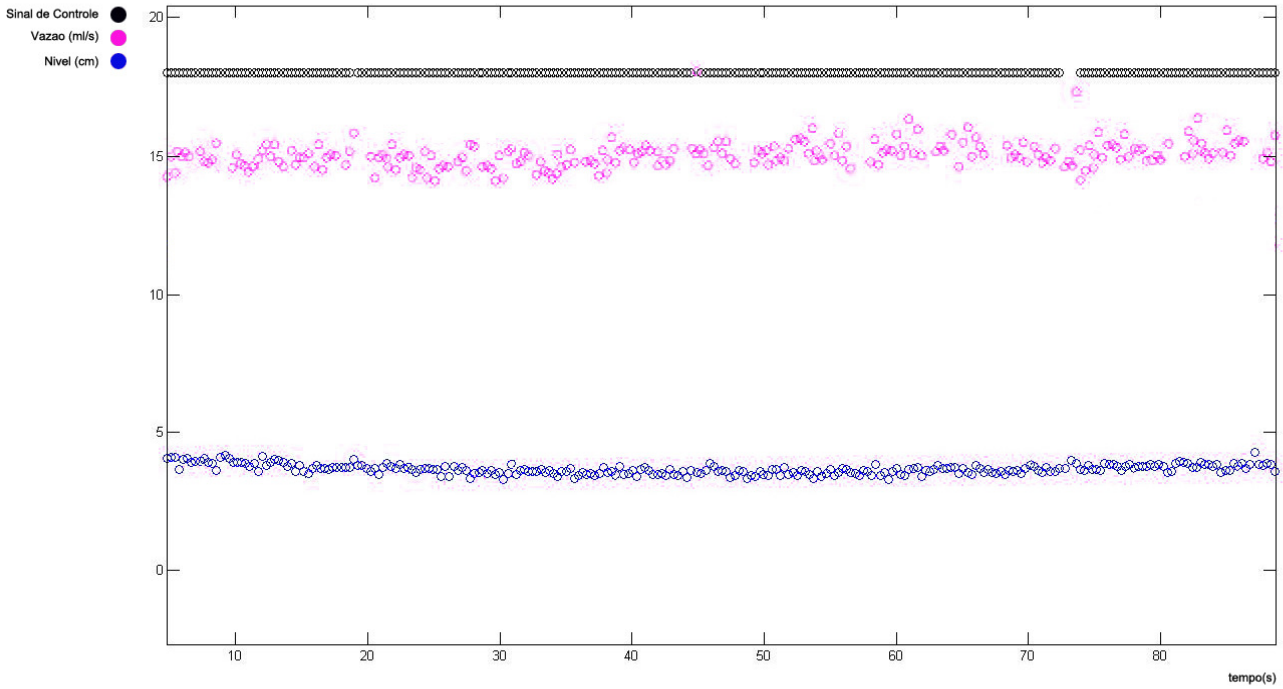


Figura 4.21 - Resposta do sistema com PWM com *duty cicle* de 60%

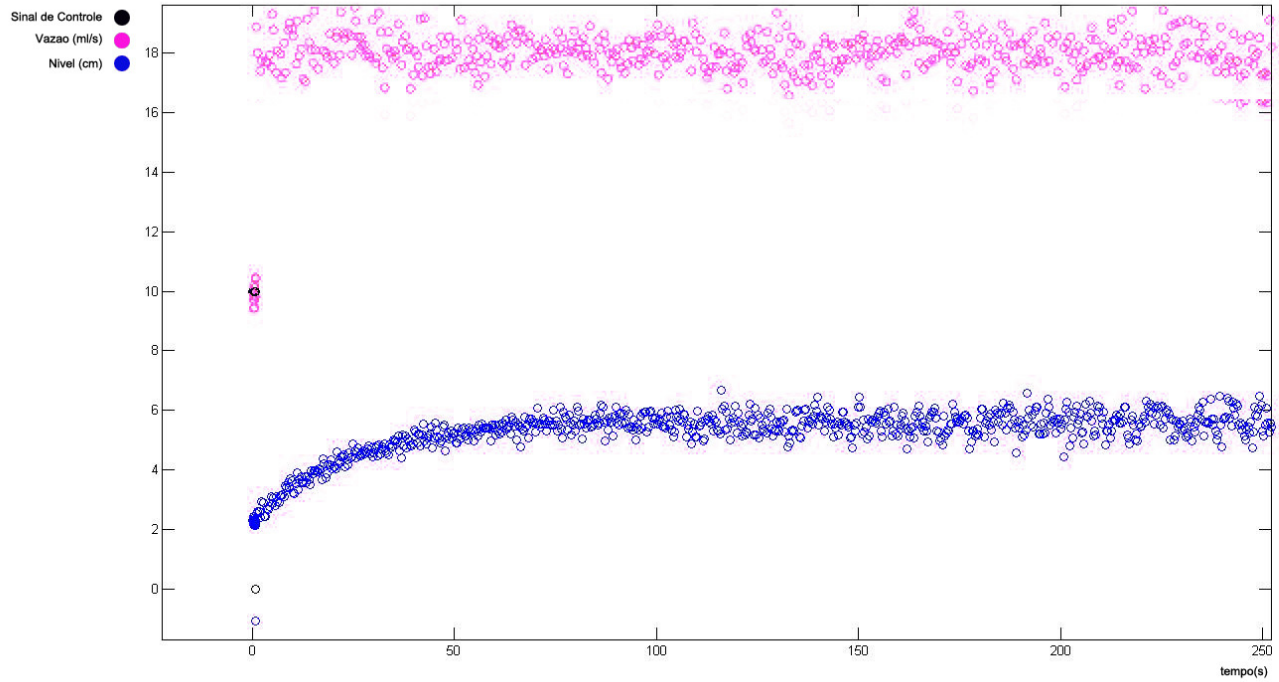


Figura 4.22 - Resposta do sistema com PWM com *duty cicle* de 80%

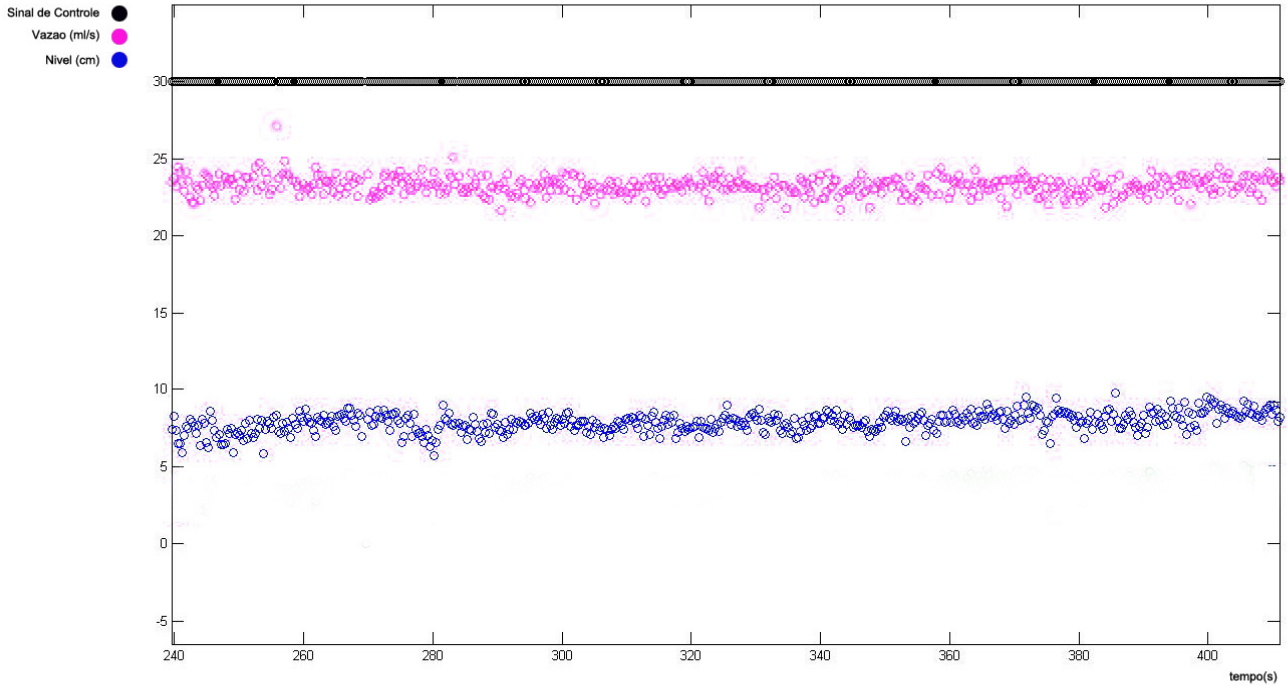


Figura 4.23 - Resposta do sistema com PWM com *duty cicle* de 100%

4.4.3.2 PARÂMETROS PARA CURVA CARACTERÍSTICA DA SAÍDA.

Observa-se no capítulo anterior que as vazões de saída dos tanques são dependentes de parâmetros que traduzem a sua natureza de saída.

Neste protótipo, o ensaio de saída é realizado por vários pontos de operação de nível. Dessa forma, enche-se o tanque para vários valores de vazão, aguardando-se o sistema atingir o valor de regime permanente do nível. Esses valores são confrontados em uma tabela e produz uma função vazão x nível através do Método de Mínimos Quadrados.

Observa-se que Equação 4.13 mostra que a vazão de saída é função da raiz quadrada do nível. O calculo é feito utilizando a vazão de saída pela raiz quadrada do nível, pois optou-se por considerar os efeitos da não linearidade do sistema.

Tabela 4.2 - Relação PWM x Nível

	Tanques Nível em Regime Permanente
Vazões média	h1
10	2,5 cm
11,5	3 cm
15	4 cm
18	5,5 cm
24	8 cm

As Figuras 4.19 a 4.23 mostram os ensaios realizados para aquisição de dados de saída. A Figura 4.24 mostra a curva correspondente aos dados coletados. A Equação 4.38 correspondente à Figura 4.24 representa a resistência hidráulica da restrição e também mostra o comportamento não linear do tanque.

Como o ensaio foi feito em regime permanente, isto é,

$$\frac{dh_1}{dt} = 0 \tag{4.36}$$

É fácil deduzir que qi = qo. Deduz-se, também que,

$$k_1 = \frac{q_1}{\sqrt{h_1}} \tag{4.37}$$

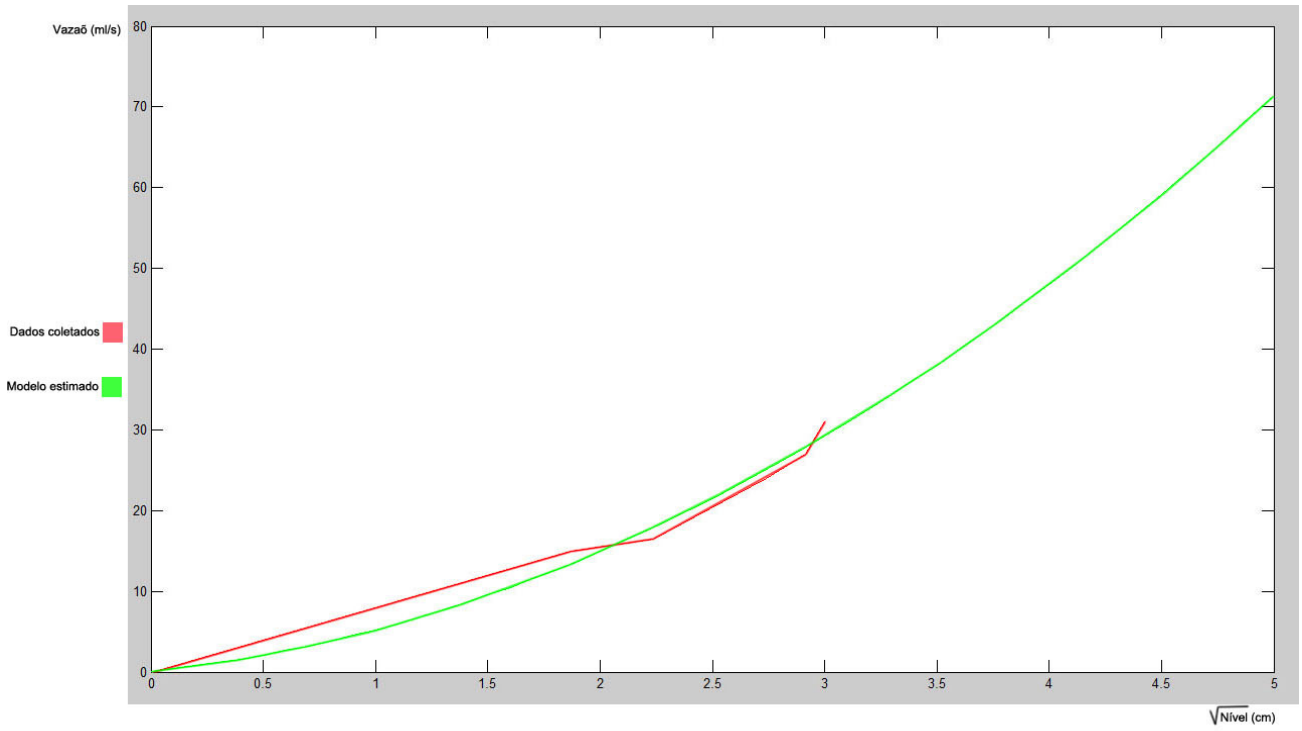


Figura 4.24 - Gráfico Vazão x Nível

O modelo que mais se adequou as características de Vazão x Sqrt(nível), foi o modelo quadrático.

A função de saída é

$$q_1 = 2{,}2861.\left(\sqrt{h}\right)^2 + 1{,}1324.\left(\sqrt{h}\right) + 0{,}1127 \tag{4.38}$$

4.4.3.3 PARÂMETROS DA CURVA CARACTERÍSTICA DA ENTRADA.

A determinação das características de entrada também é realizada experimentalmente. Como foi mencionado, os motores de esguicho d'água fornecem vazão de entrada ao tanque, cuja quantidade é controlada através da tensão. Observa-se que a saída da bomba é a vazão e que está relacionada com a tensão sendo esta sua entrada. Devido às dificuldades encontradas em obter esses parâmetros internos do dispositivo, optou-se em estimar seus parâmetros através de uma curva que representasse a saída em função da entrada. Uma das formas de realizar o controle deste tipo de dispositivo é utilizando a técnica de Modulação por Largura de Pulso comandado por um microcontrolador cujo código está no Anexo 5.

No processo de obtenção dos dados de forma semelhante a da saída, realizam-se vários ensaios determinando vários e diferentes valores de PWM no sistema. Para cada ensaio, enche-se o tanque registrando vários valores de vazão. Neste caso, não é necessário aguardar o sistema atingir o valor de regime permanente do nível, pois, existem circunstâncias em que o tanque pode transbordar. Novamente, esses valores são confrontados em uma tabela para produzir uma função PWM x Vazão através do Método de Mínimos Quadrados de função quadrática.

As Figuras 4.19 à 4.23 mostram os ensaios realizados para aquisição de dados de entrada que são os mesmos gráficos coletados para o calculo dos parâmetros de saída. A Figura 4.25 mostra a função correspondente aos dados coletados. Essa função representa a relação de PWM em relação à quantidade de líquido. A Figura 4.25 também mostra o comportamento não linear do desta relação.

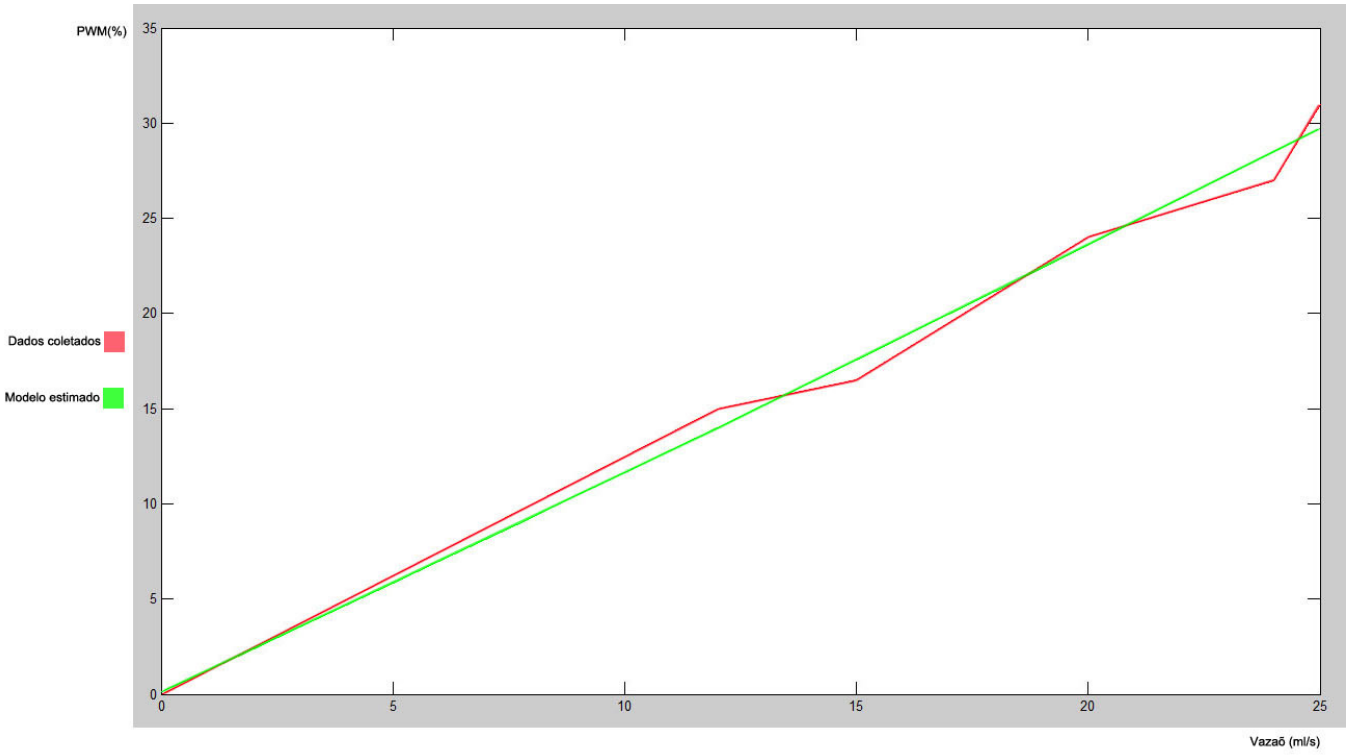


Figura 4.25 - Relação do PWM X Vazão de entrada

Tabela 4.3 - Relação PWM x Vazão de entrada em regime permanente

	Tanques Vazão em Regime Permanente
PWM (%)	Vazões Média
32	10
40	11,5
60	15
80	18
100	24

O modelo que mais se adequou as características de PWM x Vazão, foi o modelo quadrático.

A função de entrada é:

$$q_{v1} = 0{,}0021.(pwm)^2 + 1{,}1324.(pwm) + 0{,}1127 \qquad (4.39)$$

Observa-se que a saída do conjunto de motores é a vazão e que está relacionada com a tensão sendo esta sua entrada. Devido às dificuldades encontradas em obter esses parâmetros internos do dispositivo, considerou-se como um simples sistema “caixa preta”, optando-se em estimar seus parâmetros através de uma curva que representasse a saída em função da entrada.

4.5 CONCLUSÃO DO CAPÍTULO

Este capítulo descreveu a forma com que foi constituído o sistema de nível de líquido a partir da modelagem baseado na física do processo (AGUIRRE, 2004) e a implementação do protótipo, desde a especificação dos componentes até os circuitos de acionamento e de aquisição de dados, incluído os cálculos que são necessários para o bom funcionamento do projeto. O programa do microcontrolador esta no Anexo 5. Além disso, foi realizado um ensaio para determinação de parâmetros do sistema de forma que auxiliasse na validação do sistema e no projeto do controlador.

CAPÍTULO 5
CONFIGURAÇÃO DO ALGORITMO GENÉTICO

5 CONFIGURAÇÃO DO ALGORITMO GENÉTICO

5.1 INTRODUÇÃO

A configuração de um algoritmo genético é uma etapa critica para obtenção de bons resultados. Apesar disso, ainda não foi encontrado uma metodologia para a determinação ótima desse conjunto de parâmetros sendo que, uma determinada configuração pode ter um desempenho melhor que outra em um problema específico (BRAGA et al., 2000). As metodologias adotadas para a definição desses parâmetros normalmente são empíricas, isto é, baseadas na experiência prévia do projetista. A base de conhecimento utilizada neste sistema *fuzzy* poderia ser perfeitamente obtida diretamente de um projetista experiente, ou de forma analítica dos dados e sinais, ou automaticamente, através de algoritmo genético, como demonstra esse trabalho.

Com o relacionamento dos parâmetros de configuração é baseado em heurística, a primeira decisão importante é a escolha de uma representação cromossômica dos indivíduos de forma a obter o melhor desempenho do algoritmo. Além da arquitetura da representação cromossômica escolhida, é necessária determinar a precisão desejada. Dependendo do universo dos números utilizados (binários, reais inteiros, mistos) e da maneira que cada gene ou conjunto de genes são mapeados na solução, pode haver influência no tamanho do cromossomo e, por consequência, alterar o comportamento do desempenho do algoritmo e na qualidade da solução.

Adotou-se neste trabalho, uma representatividade utilizando números reais flutuantes, sendo que, para a base de regras, utilizou-se o arredondamento para o inteiro mais próximo. Esta configuração será apresentada na Seção 5.4.

Uma precisão maior do sistema implica num maior número de genes, aumentando o tamanho do cromossomo, seu espaço ocupado na memória e acarretando perda de performance do algoritmo. Pelo contrário, uma representação com baixa precisão pode mesmo impossibilitar a determinação de uma solução adequada do problema.

Para evitar esses problemas, adota-se neste trabalho uma representação utilizando números reais e inteiros, apresentado também na Seção 5.4.

A segunda decisão importante a ser tomada é o tamanho da população a ser submetida ao algoritmo genético. O desempenho global, a eficiência do algoritmo genético e o resultado da função de *fitness* são diretamente influenciados pelo tamanho da população. Populações pequenas fornecem um universo pequeno de representações na busca da solução do problema e pode provocar convergência prematura para soluções locais. Populações grandes, embora o universo de representações aumente, torna maior o tempo de convergência para a solução e exige maior capacidade computacional.

Outros fatores de configuração relacionados à população também exigem atenção especial. Sobre a taxa de elitização, onde os melhores indivíduos são selecionados para formar a nova população, o valor deve ser moderado para não possibilitar a substituição da maior parte da população por indivíduos de baixa adaptabilidade ou tornar o sistema lento por processar uma população de indivíduos redundantes. Sobre a taxa de recombinação (*crossover*), quanto maior, mais rapidamente novas estruturas serão introduzidas na população e elementos de boas representatividades poderão ser retirados de forma mais rápida do que a velocidade de geração dos melhores indivíduos. Entretanto, com um valor baixo, a evolução da população pode estagnar, convergindo para um mínimo local.

Em seguida, é necessário configurar a taxa de ocorrência da mutação que, em geral, é um valor baixo adotado. Define-se também uma taxa de recombinação adequada que é imprescindível para o funcionamento adequado do algoritmo genético. Esta taxa é diretamente proporcional à velocidade de introdução de novas estruturas na população. Entretanto, precisa ser aplicada com moderação, pois, se for um valor muito elevado, elementos com boas adaptabilidades poderão ser retirados mais rapidamente do que são gerados melhores indivíduos. Por outro lado, se for aplicado um valor baixo, a evolução da população pode estagnar, convergindo para um mínimo local.

5.2 PROPOSTA PARA OTIMIZAÇÃO DO SISTEMA *FUZZY*

Este trabalho apresenta uma proposta de uma arquitetura para o projeto automático de sistemas *fuzzy* para um sistema não linear utilizando algoritmos genéticos.

A arquitetura proposta baseia-se em um nível único de representação, ou seja, um cromossomo codifica toda a solução do problema. Como tarefa de se projetar automaticamente todo um sistema *fuzzy* é, de certa forma, muito complexa, algumas abordagens envolvem co-evolução adotando níveis hierárquicos de codificação pela combinação de indivíduos de outras populações e várias populações representam elementos e soluções distintas dos sistemas *fuzzy* onde a função de avaliação do *fitness* de um indivíduo é a mesma para todos da população (RAPOSO, 2000).

O presente trabalho se propõe a mostrar que é possível a obtenção de bons resultados utilizando apenas um nível de representação, desde que os operadores genéticos utilizados levem em conta a codificação elaborada, considerando as diferentes semânticas em diferentes seções do cromossomo.

Dessa forma, por exemplo, o operador de mutação por inversão (na Seção 5.4.4) está imediatamente descartado, uma vez que pode alocar a um gene um alelo incompatível com a semântica daquela posição.

Os parâmetros de projeto incluídos no algoritmo genético são apresentados neste Capítulo 5. A partir da próxima seção são detalhados os vários componentes e processos que integram um algoritmo genético e apresentadas as propostas específicas dessa dissertação para o projeto automático dos modelos *fuzzy.*

5.3 REPRESENTAÇÃO CROMOSSÔMICA

O primeiro passo para a utilização dos algoritmos genéticos é a definição da representação cromossômica dos indivíduos da população. Os cromossomos são compostos por genes, dígitos alfanuméricos, que serão, tal como na biologia, alterados quando houver reprodução.

Para este trabalho, na representação do modelo *fuzzy*, basta encontrar um individuo cujos valores de seu cromossomo seja ótimo para o controle eficiente da planta. Neste caso, a abordagem cromossômica utilizada é abordagem Pittsburgh onde cada elemento é uma possível solução do problema. Existe ainda outra abordagem relacionada com cromossomos que é a abordagem Michigan na qual a população como um todo é a solução. Essa ultima abordagem é mais utilizada, por exemplo, na solução do problema do caixeiro viajante (PEÑA-REYES, 2002) O conjunto de todas as configurações que o cromossomo pode assumir forma o espaço de busca do algoritmo genético. Assim, caso o cromossomo represente n parâmetros de uma função, o espaço de busca resultante é um espaço de n dimensões.

Tradicionalmente, os indivíduos são representados por um vetor binário onde cada elemento do vetor (gene) denota a presença (1) ou ausência (0) de uma determinada característica: o seu genótipo. Os elementos podem ser combinados formando as características reais do indivíduo, ou seja, o seu fenótipo. Teoricamente, cada problema não possui uma representação restrita pode ser manipulada conforme a necessidade.

Dependendo do problema a ser tratado, outras representações podem ser mais eficientes e, portanto, adotadas. Pode-se pensar em se utilizar números naturais, caracteres, ou mesmo uma combinação dos dois.

E ainda o uso de representações com valores reais, através de permutações ou por meio de árvores hierárquicas ou mesmo a representação binária apresentando a variante do código Gray. A escolha pela representação a ser utilizada é dependente da natureza do problema em estudo. Além disso, os

operadores genéticos, abordados na Seção 5.4.4, também estão intrinsecamente conectados ao tipo de representação adotada.

A primeira população de indivíduos normalmente é criada de maneira aleatória. É recomendado apenas o cuidado para que seus indivíduos tenham uma ampla representatividade no domínio do problema em estudo. Em caso contrário, corre-se o risco do processo de busca paralisar em um ponto de máximo local, em um problema de maximização, ou mínimo local, em um problema de minimização, fornecendo assim um resultado indesejável para a solução global do problema.

Um importante fator a ser considerado durante a elaboração da representação cromossômica é a manutenção da validade dos indivíduos durante o processo evolutivo. Idealmente, após a criação da população, todos os operadores genéticos devem manter a validade de um indivíduo, no sentido de que este continue representando uma solução factível do problema. Caso tal factibilidade possa vir a ser violada, algum procedimento deve ser executado para corrigir, ou eliminar, o indivíduo problemático.

A segunda opção é a implementação de uma função de penalização dos indivíduos que representam soluções infactíveis. Tal função deve ser cuidadosamente elaborada para que não faça com que o algoritmo genético forneça, ao final do processo evolutivo, soluções simplesmente factíveis, sem a otimização do critério de desempenho.

5.4 PROPOSTA DE REPRESENTAÇÃO CROMOSSÔMICA

Uma representação cromossômica foi elaborada, uma para a arquitetura *fuzzy,* com os seguintes objetivos:

1. A representação seja simples, transparente;
2. Os operadores genéticos sejam aplicados entre elementos com a mesma semântica. Por exemplo, durante um *crossover* não são combinados genes representando base de regras do modelo com genes representando o centro de uma função de pertinência;

3. Parâmetros que não tenham uma quantidade fixa pré-determinada (como o número de estados nas premissas das regras) possuem certo grau de liberdade para aumentar ou diminuir seu número;
4. Restrições de integridade sejam automaticamente preservadas, ou facilmente passíveis de correção, durante a aplicação dos operadores genéticos;
5. A representação não seja esparsa.

Aqui se optou por escolher funções de pertinência dos termos linguísticos em formato triangular, configurada por três pontos: valor inicial, centro e valor final. Foi escolhido por ser mais fácil definir os limites de vizinhança das funções de pertinências. Essa escolha não é crítica uma vez que tem sido confirmado que a maior importância na definição das partições das variáveis de entrada de sistemas *fuzzy* reside na disposição e grau de sobreposição das funções de pertinência, e não propriamente em seu formato (se gaussiana, triangular ou trapezoidal) (WANG, 1997).

Poderiam ser escolhidas funções de pertinência dos termos linguísticos como gaussianas, parametrizadas por dois valores: seu centro e sua abertura. De fato, esses modelos apresentam melhores resultados em uma série de exemplos de aproximação de funções, comparados a modelos utilizando funções de pertinência triangulares ou polinomiais. É fato também que a maioria dos resultados de aproximação universal em sistemas *fuzzy* assume o uso de funções de pertinência gaussianas (WANG, 1994).

A configuração da proposta deste trabalho permite estabelecer duas interpretações para a codificação das funções de pertinência. Na primeira, os valores de seus centros são absolutos. Na segunda, as posições de seus centros são relativas. Apenas o valor do centro de uma das funções de pertinência, utilizada como referencia, é absoluto. Nas outras funções de pertinência, o valor real do centro coincide com o valor limite de atuação da função de pertinência vizinha, ou seja, o parâmetro centro define o onde começa, ou termina a funções de pertinência vizinha, como mostram as Figuras 5.2, 5.3 e 5.4.

x1 x2 x3	x4 x5	x6 x7 x8 x9	x10 x11 x12 x13 x14 x15 x16 x17 x18 x19 x20 x21 x22 x23 x24
erro	derro	saida	regras

Figura 5.1 - Cromossomo e a disposição dos genes

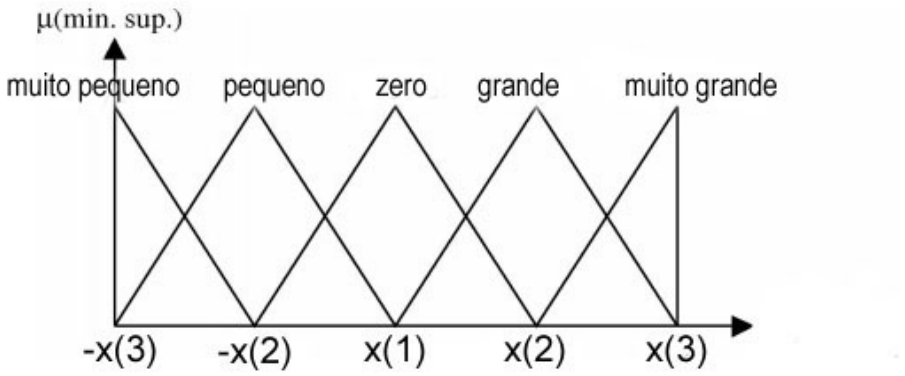


Figura 5.2 - Funções de pertinência da variável de entrada “erro”.

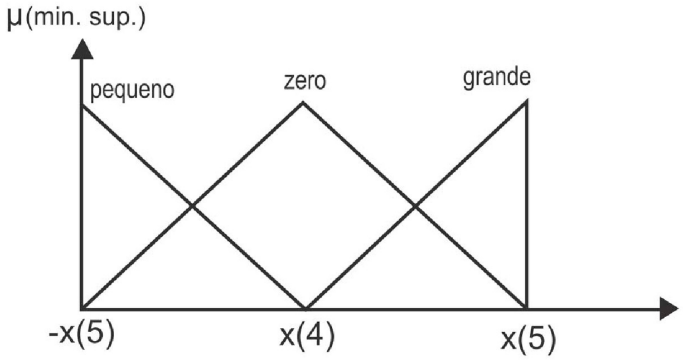


Figura 5.3 - Funções de pertinência da variável de entrada “variação do erro”.

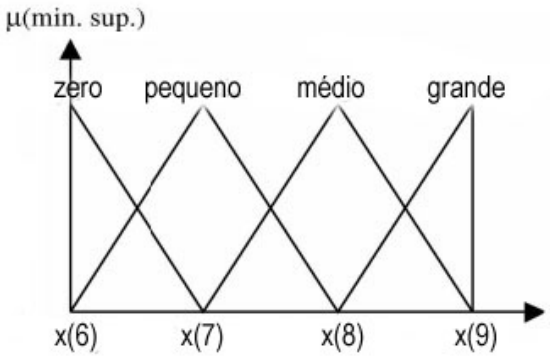


Figura 5.4 - Funções de pertinência da variável de saída.

Essa implementação possibilita o controle do grau mínimo ou máximo de sobreposição entre as funções de pertinência de forma mediana, bastando para isso impor esses limites diretamente aos respectivos genes do cromossomo, através da configuração dos limites de máximos e mínimos. Claramente, o controle das aberturas (ou larguras) das funções de pertinência pode ser feito independentemente da interpretação dada a seus centros. Sendo assim, a segunda interpretação apresentada será adotada neste trabalho.

Esta segunda interpretação (posição relativa dos centros) é uma opção natural da arquitetura proposta. Existem técnicas de medidas de similaridades, que correspondem a métodos eficientes no controle da sobreposição entre funções de pertinência, simplificando modelos *fuzzy*, e que não foram utilizadas neste trabalho.

A inicialização de todos os parâmetros codificados na população pode ser feita de duas maneiras: através de uma distribuição uniforme, já que em princípio não há informações sobre regiões do espaço de busca probabilisticamente mais promissoras, onde se poderia, por exemplo, centrar as funções de pertinência. Ou inicializar com valores aproximados, quando se conhece o universo de atuação do sistema. Esta etapa depende do problema sendo tratado e do conhecimento sobre as ordens de grandeza dos parâmetros codificados no algoritmo genético. Espera-se, também, que a inicialização da população seja representativa de todo o espaço de busca, desde que, para cada parâmetro sendo otimizado, sejam estipulados os limites de máximos e mínimos de seus universos.

No caso do controlador de nível de liquido, o cromossomo contém os parâmetros de função de pertinência e base de regras do controlador *Fuzzy* que serão otimizados no processo. Os conjuntos *fuzzy* utilizados possuem formato triangular. Para simplificar os cálculos utilizamos valores maiores ou iguais a zeros, e aplicamos a simetria dos valores para valores negativos das funções de pertinência (Figura 5.2 e Figura 5.3). Além disso, foram criadas 15 regras para a inferência do controlador. Sendo assim, o cromossomo de cada indivíduo possui 9 parâmetros das funções de erro, derivada do erro e saída, e 15 parâmetros para base de regras, totalizando 24 parâmetros, como mostra a Figura 5.1

Também é possível definir as restrições dos valores do cromossomo. Para isso, utilizam-se dois vetores. Um definindo limites máximos e o outro definindo os limites mínimos de cada elemento do cromossomo. Deve se lembrar que tais restrições interferem diretamente no resultado e desempenho do sistema. Portanto precisam ser estimadas com o máximo de cuidado e bom senso, de acordo com os limites da planta.

5.4.1 AVALIAÇÃO

Após a definição da representação cromossômica, é necessário elaborar uma estratégia de avaliação dos indivíduos da população. Esta função de avaliação depende diretamente do problema que está sendo tratado. Uma propriedade que ela deve possuir é explicitar quais cromossomos representam as melhores configurações para a solução do problema em questão, assim como apontar aqueles que geram elementos insatisfatórios. Convenciona-se que essa função, também chamada função de *fitness*, deve atribuir valores mais altos aos indivíduos melhores, mais aptos a resolver o problema de otimização, e valores menores às soluções pobres. Além disso, são comumente usadas funções não negativas.

Normalmente, o *fitness* (valor da função de avaliação) de um indivíduo é calculado a partir da simulação do algoritmo no ambiente do problema que está sendo tratado. É necessário decodificar o cromossomo e obter a proposta de solução, na maioria dos casos. Dessa forma, a avaliação do desempenho vai depende de cada caso. Se os cromossomos representam sistemas classificadores, sua avaliação vai consistir no teste de sua capacidade de classificação sobre um conjunto de padrões de teste. Quanto maior o número de padrões corretamente classificados, maior será seu *fitness*. Caso os indivíduos ou cromossomos representem os parâmetros de um controlador, durante a avaliação pode-se medir o desempenho deste em malha fechada frente uma trajetória de referência pré-estabelecida.

Assim, quanto menor a medida do erro quadrático médio entre o sinal de referência e a saída da planta, maior será o valor do *fitness*. Em um problema de roteamento de veículos, os indivíduos que codificam as rotas com menor custo apresentarão também maiores valores de *fitness*.(HERRERA e LOZANO, 1996). O mesmo acontece em problemas clássicos como o Problema do Caixeiro Viajante (MICHALEWICZ, 1996)

A função de *fitness* pode ainda incorporar outras informações que não estão relacionadas de forma íntima ao desempenho do indivíduo codificado. Medidas de penalização também podem ser incluídas e podem se referir à infactibilidade de uma solução, à violação de restrições mais brandas ou ao número excessivo de parâmetros em uma solução (diretamente relacionado à complexidade da solução). Em problemas de otimização multiobjetivo, durante a otimização de funções com múltiplos objetivos que sejam conflitantes, a definição da função de *fitness* exige ainda mais atenção, tornando-se necessário um tratamento analítico cuidadoso.

Dentro do contexto dessa dissertação, a avaliação é realizada comparando-se a resposta dos modelos obtida em série sintética (simulação do modelo sem a utilização da saída do sistema real como entrada do modelo) com o comportamento real dos sistemas sendo modelados, ambos sujeitos a um mesmo sinal de excitação. É nessa etapa que o método dos mínimos quadrados estima o vetor dos coeficientes dos consequentes das regras, completando assim o modelo para sua avaliação. A função de *fitness* desenvolvida considera tanto o desempenho do modelo quanto sua complexidade.

Para este trabalho utilizamos a abordagem descrita em PHAM e KARABOG (1997). Nessa abordagem, um modelo matemático aproximado do processo serve de base para a geração de dados iniciais de referência para o treino do algoritmo genético. O modelo de referência pode ser escolhido de acordo com a dinâmica desejada para o sistema a ser treinado.

Para cada passo da simulação, os dados obtidos pelo modelo de referência e pela planta simulada, em cada período de amostragem, são submetidos pelo algoritmo à equação (5.1) de forma que se minimize valor do índice J_m. Na equação

5.1, w é o numero de amostras de entrada-saída de dados. Y_p é a saída da planta e y_m é a saída do modelo de referência. Logo, J_m calcula o erro absoluto entre a saída da planta e a saída do modelo de referencia. A diferença elevada ao quadrado garante que o índice J_m não será negativo. Esse método é bastante semelhante ao integral do erro absoluto utilizado como critério de desempenho do sistema PID.

$$J_m = \sum_{k=1}^{w} \left(y_p[k] - y_m[k] \right)^2. \tag{5.1}$$

As vantagens e desvantagens de cada uma das abordagens não estão bem definidas na literatura, ainda sendo motivos de estudos. A escolha é justificada principalmente pela maior facilidade de cálculo, já que a Equação 5.1 tem caráter determinístico e não está baseada em métodos estatísticos, sem necessidade de estimar funções densidade de probabilidade, por exemplo.

5.4.2 OPERADORES DE SELEÇÃO

A utilização de um critério de seleção é necessária uma vez que os indivíduos da população foram avaliados. A grande parte dos métodos de seleção foi desenvolvida para escolher preferencialmente indivíduos com maiores fatores de aptidão (valores de *fitness*). Entretanto, embora não exclusivamente, também servem para manter a diversidade da população que é importante para a variabilidade de soluções. Basicamente, existem três tipos de seleção: determinística, estocástica e híbrida.

5.4.2.1 SELEÇÃO DETERMINÍSTICA

Contempla somente os indivíduos que atendem a determinadas características previamente estabelecidas como desejáveis. Essa característica desejável converge através da avaliação da aptidão ou *fitness*. Os indivíduos que não atenderem o critério de escolha são eliminados de forma imediata. Assim, não há chance de um indivíduo que não satisfaça os requisitos vir a ser escolhido para fazer parte da próxima geração.

5.4.2.2 SELEÇÃO ESTOCÁSTICA

Todos os indivíduos podem ser selecionados. Entretanto, os indivíduos que apresentarem maior adaptabilidade terão mais chance de serem escolhidos. Os indivíduos de menor adaptabilidade terão menos chance, mas poderão eventualmente ser selecionados.

5.4.2.3 SELEÇÃO HÍBRIDA

Este processo de seleção intercala os critérios de seleção determinísticos e estocásticos, já descritos acima. Dessa forma, há, por hipótese, a garantia de escolha de parte dos melhores indivíduos da população, porém mantendo a possibilidade de seleção também dentro de todo o espaço de soluções.

Nas novas abordagens de seleção, outros métodos também são discutidos: Método da Roleta, Seleção Baseada em Rank e Método do Torneio.

5.4.2.4 MÉTODO DA ROLETA.

O índice de aptidão é proporcional a representatividade de cada indivíduo da população na roleta. Isto é, os indivíduos com alta adaptabilidade têm uma proporção maior da roleta, enquanto aos que possuem menor adaptabilidade é dada uma proporção menor. Quando a roleta é girada um determinado número de vezes, dependendo do tamanho da população, os operadores de reprodução atuam sobre os sorteados na roleta, ou então, são copiados diretamente para a população da geração seguinte. É um processo predominantemente estocástico, portanto é probabilístico e variante com as gerações.

Assim como todos os métodos de seleção proporcionais ao *fitness*, o método da roleta possui desvantagens. O primeiro problema esta no tamanho de solução de fitness da população. Se a população contém uma solução de *fitness* proporcionalmente maior que as demais soluções na população, esta irá ocupar a maior parte da roleta, acarretando na escolha do mesmo resultado no sorteio, fazendo a população perder sua diversidade e provocando a convergência

prematura para soluções que podem não satisfazer o problema. O segundo problema é bem parecido com o primeiro e pode ocorrer após várias gerações. Conforme a roleta for sendo executada durante as gerações, a maioria dos membros da população pode assumir, aproximadamente ou exatamente, o mesmo *fitness*, possuindo a mesma probabilidade de serem selecionados. Além disso, o uso do mecanismo da roleta não permite o calculo de valores de *fitness* negativos caso seja necessário em algum projeto específico.

5.4.2.5 SELEÇÃO BASEADO EM *RANK.*

É bem semelhante ao método da roleta. A diferença reside nas proporções que são atribuídas aos indivíduos de acordo com sua posição após a ordenação ascendente ou descendente pelo valor do *fitness*. Só após a ordenação é que são utilizados mapeamentos lineares ou não lineares para a determinação da probabilidade de seleção de cada indivíduo. A seleção baseada em *rank* não apresenta nenhum dos problemas citados para o método da roleta.

5.4.2.6 MÉTODO DO TORNEIO

Neste método, um determinado indivíduo, dentro de uma população, é selecionado para pertencer à nova geração determinística ou probabilística, de acordo com seu valor de *fitness*. No mínimo, duas soluções são sorteadas e a melhor delas é escolhida.

Dois eventos podem ocorrer após o fim do torneio. Ou todos os participantes são colocados de volta na população e provavelmente voltam a participar de outros torneios, ou aguardam fora da população até que um determinado número de torneios volte a ocorrer. O desempenho do método é controlado pelo numero de indivíduos envolvidos em cada torneio e que é proporcional a taxa de convergência do algoritmo. A implementação do método do torneio é bem mais simples e eficiente que os outros, não requerendo o ordenamento da população (Seleção baseada em *rank*) e não requer tanta estrutura computacional, mostrando-se mais adequado a implementações paralelas.

Assim como na Seleção baseada em *Rank,* o Método do Torneio não possui os problemas apresentados no método de roleta. Além disso, aplica-se tanto a problemas de maximização quanto de minimização, enquanto o método da roleta se compromete naturalmente ao objetivo de maximização. Entretanto, não se trata de uma restrição grave, pois basta uma adaptação da função de *fitness* para que se resolvam problemas de minimização.

5.4.3 PROPOSTA DE OPERADOR DE SELEÇÃO

As definições dos outros parâmetros do algoritmo genético devem servir de base para a escolha do operador de seleção de forma que trabalhem em conjunto na execução do algoritmo genético. Isso diz respeito principalmente ao operador de reprodução, que será abordado em seguida. Uma implementação de algoritmo genético eficiente pode ser obtida aliando operadores de recombinação e mutação com alto poder de perturbação com um operador de seleção de desempenho seletivo moderado.

Tendo em vista as vantagens devido à simplicidade e eficiência, foi adotado, no presente trabalho, como operador de seleção, o Método de Torneio, conforme justificativas na seção anterior. Na configuração estabelecida, um total de cinco por cento da população é escolhido aleatoriamente para participar do torneio.

Deterministicamente é selecionado o indivíduo com o maior valor de *fitness*. O método é implementado com reposição, ou seja, todos os indivíduos participantes do torneio são recolocados na população, inclusive o selecionado. Foi implementado também elitismo, o qual consiste em selecionar o indivíduo com o maior *fitness* de uma geração e incluí-lo na geração seguinte.

5.4.4 REPRODUÇÃO

A reprodução é a fase do algoritmo genético onde indivíduos pais misturam parte de suas informações cromossômicas com o objetivo de criar novos indivíduos filhos para compor a nova população. Consiste na ação de operadores genéticos, os quais são aplicados sobre os indivíduos pais, gerando os filhos. A literatura expõe algumas técnicas utilizando diversos operadores genéticos.

A definição dos operadores de reprodução está intrinsecamente ligada ao tipo de representação adotada, a qual, por sua vez, depende da classe de problemas que se estuda. São dois os operadores genéticos básicos utilizados durante a fase de reprodução nos algoritmos genéticos:

5.4.4.1 RECOMBINAÇÃO E MUTAÇÃO.

O operador de recombinação (ou *crossover*) envolve a participação de dois indivíduos pais. A técnica da recombinação consiste na troca de material genético entre os pais, gerando dois candidatos a filhos.

Muitos operadores de recombinação já foram desenvolvidos e estão intimamente relacionados com o tipo de representação adotada. Os pontos de recombinação são escolhidos aleatoriamente respeitando uma região de combinação de genes. As variações desse método apenas dizem respeito à definição do número de pontos, a região desses pontos dentro do cromossomo e a maneira como os genes são trocados. De qualquer forma que se processe a recombinação, o resultado será sempre a geração de dois filhos. A principal motivação em usar esse tipo de operador é poder, através da combinação de soluções boas, gerar soluções melhores. Para representação binária, tem-se dois tipos principais: *crossover* de *n*-pontos e *crossover* uniforme.

O *crossover* de 1-ponto é o mais simples da família de *n*-pontos. Nesse tipo de operador de c*rossover*, um ponto do cromossomo dos pais é escolhido aleatoriamente e suas cadeias, usando esse ponto como corte, são combinadas gerando crias.

Uma das alternativas mais conhecidas para a família de tipos de *crossover* de *n*-pontos é o *crossover uniforme*. Nesse mecanismo, os bits são trocados individualmente entre os dois pais. Caso um número pseudoaleatório, gerado entre 0 e 1, seja maior que um determinado limiar, o alelo (ou gene no caso da representação da cria) será igual ao do pai 1. Caso contrário, do pai 2

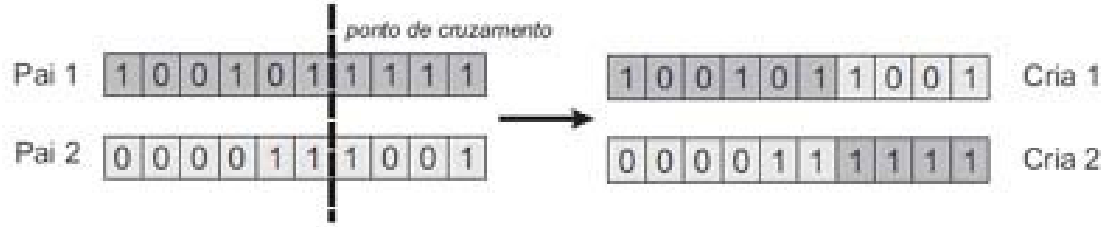


Figura 5.5 - *Crossover* de 1-ponto

A figura 5.5 ilustra esse mecanismo, o qual pode ser interpretado como um crossover de n pontos com n variáveis a cada cria gerada. O crossover uniforme também permite que a ordem de aparecimento dos atributos no cromossomo seja indiferente para a eficiência da busca.

Figura 5.6 - *Crossover* Uniforme

Outro operador de recombinação, só que dessa vez aplicado à representação dos atributos em ponto flutuante ou valore reais, é o *crossover* aritmético (MICHALEWICZ, 1996). Dados dois pais P1 e P2, esse mecanismo realiza uma combinação linear entre os cromossomos dos pais, gerando uma cria como mostra a Equação 5.2:

$$\hat{g}_i = \alpha_i(g_i^1) + (1 - \alpha_i)(g_i^2) \tag{5.2}$$

sendo ˆgi o i-ésimo gene de uma cria, g1 i o i-ésimo gene do pai 1 e g2, i o i-ésimo gene do pai 2. O parâmetro alfa é um número aleatório com distribuição uniforme dentro do intervalo [–δ; 1 + δ]. A proposta mais comumente adotada restringe o parâmetro alfa ao intervalo [0, 1]. No entanto, através do aumento do intervalo, definido aqui como um tipo de extrapolação, é possível aumentar a capacidade de busca do operador permitindo uma exploração mais efetiva do espaço de busca, dado que em sua versão original o valor de determinado gene de um indivíduo filho é limitado pelos valores dos respectivos genes dos indivíduos pais. Além disso, opera-se com um valor distinto de β para cada elemento. Entretanto, em alguns casos, pode ocorrer problemas de infactibilidade. Esta medida, aliada a uma alta taxa de ocorrência de *crossover*, contribui para a manutenção da diversidade da população durante sua evolução. Esse mecanismo é realizado até que todos os genes dos cromossomos da cria tenham sido recombinados.

Em algumas abordagens sugere-se que, destes dois filhos, apenas um sobrevive, ao qual se tornará um indivíduo na próxima geração. A escolha do filho sobrevivente pode ser feita de através de vários métodos. São eles: Sorteio, Elitista, Geométrico, Heurístico.

No método de seleção por Sorteio simplesmente é realizado um sorteio entre os dois candidatos a filho. No método Elitista, a escolha recai sobre o candidato de maior adaptabilidade. No método geométrico é utilizado para valores reais assumidos nos cromossomos.

Após a ação do operador de recombinação ocorre o processo de mutação. Na natureza a mutação ocorre naturalmente quando há replicação de material genético. Durante essa replicação ocorrem erros aleatórios no material genético, resultando na diferenciação do indivíduo replicado. Isso propicia a variabilidade genética dos indivíduos dentro de uma população. Na prática, atua, durante o processo de evolução/otimização, como o meio para exploração de novas áreas do espaço de busca para valores ótimos.

Dentro do Algoritmo Genético, uma mutação pode ser caracterizada por diversos critérios, sendo os principais: eliminação, duplicação, inversão ou deslocamento de um conjunto de genes.

Nos algoritmos genéticos, o operador genético de mutação envolve a participação de apenas um indivíduo. Esta técnica consiste, conforme a analogia à Biologia Genética, na alteração de um ou mais genes de um indivíduo após o *crossover*. O novo indivíduo gerado para a próxima geração pode assim possuir alguma característica adicional que lhe forneça melhor adaptação que seu gerador. Os pontos que identificarão as posições dos genes que sofrerão as transformações são escolhidos aleatoriamente.

Exemplifica-se com um caso de representação binária, se um bit é escolhido para mutação, ele é trocado pelo seu valor complementar, no processo denominado aqui de mutação pontual.

Figura 5.7 - Mutação Pontual

Na representação real, a forma mais comum é adicionar ao valor do gene uma perturbação aleatória com uma determinada distribuição. Outro exemplo de operador de mutação é a mutação por inversão. Após a escolha aleatória de dois pontos no cromossomo, o operador inverte os genes entre estes pontos. A Figura 5.8 ilustra o comportamento desse operador ao se sortear o primeiro e o último gene do cromossomo para a delimitação da inversão.

Cromossomo original

0,55 0,23 1,70 3,01 0,84 -1,22 6

Cromossomo após mutação

0,55 **-1,22 0,84 3,01 1,70 0,23** 6

Figura 5.8 - Operador de mutação por inversão.

Assim como no operador de crossover, razão para a existência do operador de mutação, além da coerência biológica, é a geração de uma diversidade na população. Isso acarreta em desvios de direção com foco no resultado ótimo global. Desta forma, a mutação assegura que a probabilidade de se chegar a qualquer ponto do espaço de busca nunca será nula. Também resolve o problema de encontrar ótimos locais através da leve alteração da direção da busca.

Cada operador genético pode ser configurado para reagir de forma conveniente. Tem associado a si, cada um, uma taxa de ocorrência. Os operadores de recombinação possuem altas taxas de ocorrência. Os de mutação, baixa, sob a pena da degeneração do algoritmo em uma busca aleatória.

5.4.5 PROPOSTA DE OPERADORES DE REPRODUÇÃO

5.4.5.1 *CROSSOVER*

Para a arquitetura elaborada o operador de recombinação utilizado é o crossover aritmético pelo detalhamento já mencionado.

A presente aplicação evolui o sistema *fuzzy* tanto na base de regras quanto na base de conhecimento (funções de pertinência), acarretando na ampliação do espaço de soluções possíveis e no aumento da complexidade da implementação.

Isto significa que o cromossomo inclui as funções de pertinência das entradas e da saída e a base de regras. Dado que a representação proposta é real (em contrapartida à representação binária), o uso do *crossover* aritmético é imprescindível. Os operadores de *crossover* tradicionais de um ponto ou de n-pontos, mesmo em codificações binárias, tem a dificuldade de não serem capazes de realizar uma exploração eficaz do espaço de busca. Para codificações reais, o único meio de se alcançar essa exploração é permitindo que os indivíduos filhos possuam genes que não estão presentes em nenhum de seus pais (LISKA E MELSHEIMER, 1994).

Os operadores de crossover e mutação, detalhada em seguida, atuam de forma separada na base de regras e na base de conhecimento, a fim de evitar os genes referentes a regras se misturem com os genes que representam as funções de pertinência. Essa independência nos operadores ajuda a evitar o surgimento de indivíduos inconsistentes e assegura que a base de regras evolua sem a interferência da base de conhecimento e vice versa. O *crossover* aritmético é aplicado entre todos os genes, observando-se apenas que as posições do cromossomo, referentes a base de regras, são número inteiros, sendo necessário então o arredondamento dos valores finais obtidos para o inteiro mais próximo.

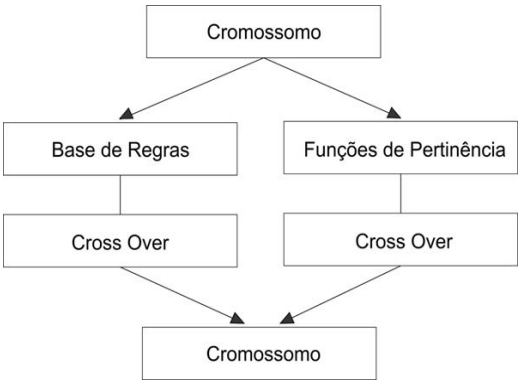


Figura 5.9 - Crossover atuando de forma separada no cromossomo.

O *crossover* aritmético opera, resumidamente, da seguinte maneira (MICHALEWICZ, 1996):

1.Para cada variável de entrada faz-se:

2.Para cada termo linguístico (função de pertinência) faz-se:

(a) Se os dois indivíduos possuem o termo linguístico, aplica-se o *crossover* às suas funções de pertinência, combinando os respectivos parâmetros (aberturas e centros);

(b) Se apenas um indivíduo possui o termo linguístico, copia-se tal termo para o respectivo indivíduo filho;

3.Aplica-se o *crossover* aritmético ao número de estados nos consequentes das regras, realizando o devido arredondamento.

Após a aplicação do operador, uma verificação de validade simples é necessária para a definição da abertura das funções de pertinência. Nesse caso, se a abertura da função de pertinência resultante for menor que o limite especificado, então atribui-se à abertura o valor deste limite mínimo. O caso contrário também é verdadeiro. Caso a abertura da função de pertinência resultante for maior que o limite especificado, então se atribui à abertura o valor deste limite máximo.

Temos dessa maneira 2 operadores de crossover e 4 operadores de mutação. O *crossover* das funções de pertinência é feito da seguinte forma: pai e mãe são escolhidos por Torneio, ou seja, os melhores indivíduos têm mais chance de serem selecionados para crossover. O filho tem uma probabilidade de 50% de herdar a função de pertinência de cada pai para uma dada entrada/saída. Aqui também não há mistura entre as funções de pertinência, o operador toma o cuidado de não permutar funções de pertinência de entrada com as de saída ou de uma entrada como a temperatura com a entrada de pressão.

No *crossover* da base de regras os pais também são escolhidos por Torneio e o filho tem a probabilidade de 50% de herdar uma regra de cada pai. No caso de pais com números de regras diferentes, o filho tem a possibilidade de 50% de herdar as regras sobressalentes do pai e de 50% de não herdar a regra. As regras não são quebradas em seus termos, ou seja, não há como um filho ter uma regra com o antecedente de um pai e o consequente do outro.

Assim como os operadores de *crossover*, os 4 operadores de mutação são selecionados por torneio, privilegiando os indivíduos mais adaptados. Mutação das funções de pertinência: As funções de pertinência são compostas por um número finito de pontos interpolados, por exemplo, uma função triangular necessita apenas de 3 pontos para ser representada, uma função trapezoidal necessita de 4 e assim por diante.

Na base de regras existem 2 tipos de mutação, a mutação do número de regras, que adiciona ou remove uma regra da base, e a mutação dos antecedentes e consequentes, que adiciona ou remove termos da regra.

Para a seleção decidimos por utilizar torneio de 3 indivíduos a fim de manter a diversidade, impedir a estagnação precoce da população e garantir sempre a sobrevivência do melhor indivíduo. A seleção é efetuada após a mutação e crossover. A taxa de mutação foi ajustada em 10% e a taxa de crossover em 25%.

Cada vez que uma operação de *crossover* deve ser efetuada, realiza-se um sorteio entre eles, sendo que o *crossover* aritmético tem probabilidade de ocorrência igual a 70%, e o uniforme de 30%.

Estes valores são justificados pela maior capacidade do operador de *crossover* aritmético de manter a diversidade da população (uma vez que tende a gerar indivíduos filhos distintos dos pais) por um número maior de gerações, fornecendo resultados médios melhores.

5.4.5.2 MUTAÇÃO

A mutação é útil para prevenir rápida convergência do valor de fitness e ajuda o algoritmo de busca a escapar de mínimos locais. Por outro lado, esta função é utilizada para preservar os diferentes estados de genes e criar uma variabilidade de individuos diferentes em uma população.

O operador de mutação implementado nesse trabalho é a Mutação de Adaptação Viável ou *mutation adapt feasible*. A região viável é delimitada pelas restrições e limites de desigualdade. A função de Mutação de adaptação viável gera aleatoriamente modificações nos genes de forma adaptativa com relação à última geração bem sucedida ou não. O tamanho do passo para a solução é escolhido ao longo de cada direção possível do problema até que as restrições e limites fiquem bem proximos de valores satisfatórios.

Esse metodo é o mais simples entre os metodos de mutação. Neste caso, não é necessário nenhuma configuração adicional, pois o próprio método de mutação se encarrega de realizar a adaptação genetica da forma mais viável possível, mesmo sendo prioritariamente aleatória.

5.4.6 CONDIÇÕES DE PARADA

Para encerrar a descrição dos componentes do algoritmo genético resta explicitar a condição de parada adotada. O algoritmo procede a evolução até que uma das duas condições seguintes seja atendida:

- Um número máximo de gerações definido pelo usuário é atingido;

- A diversidade da população, calculada em termos do desvio padrão dos valores codificados nos cromossomos (soma dos desvios padrão de cada gene), torna-se menor que um limite inferior pré-estabelecido.

Algumas possibilidades de condições de parada são:

- A n-ésima geração corresponde à quantidade máxima de gerações estabelecida na inicialização do algoritmo genético, sem que se tenha conseguido descobrir pelo menos um indivíduo que satisfaça a solução do problema;

- A n-ésima geração possui pelo menos um indivíduo que seguramente satisfaça a solução do problema, sendo n um número menor que a quantidade máxima de gerações fixada para o algoritmo genético;

- O melhor indivíduo da população se repete por um número pré-estabelecido de vezes. Esta situação caracteriza o encerramento do algoritmo por estagnação do melhor indivíduo;

- A média da adaptabilidade da população não se altera por um determinado número de gerações. Este caso é chamado de encerramento por estagnação da evolução da população.

- A diversidade da população atinge um limite inferior indicando convergência.

Neste trabalho realizou-se a configuração do algoritmo genético para 100 gerações. Assim, utilizou-se o primeiro critério como critério de parada. Esse critério é mais cômodo, pois há como estimar um tempo de execução do algoritmo. Entretanto, o mais correto para um melhor desempenho do sistema seria utilizar a estagnação do melhor indivíduo. Isso faria com que o sistema adotasse o individuo com parâmetros ótimos para o processo. Entretanto, o algoritmo poderia levar um tempo muito grande para convergir.

Com intuito de resumir o processo do algoritmo genético, mostra-se, a seguir, um pseudocódigo aplicado neste projeto. O código completo do processo esta no Anexo 2.

1. g = 0;
2. cria população P(g);
3. calcula diversidade da população inicial;
4. **fazer**
   - 4.1 Avaliação do fitness de P(g);
   - 4.2 Seleção de indivíduos para reprodução via Método de Torneio;
   - 4.3 Operação de crossover aritmético, gerando P'(g)
   - 4.4 Operação de mutação adaptação viável, gerando P(g+1)
   - 4.5 Medida de similaridade em P(g+1);
   - 4.6 Calcula diversidade da população P(g+1);
   - 4.7 g = g + 1;

enquanto (g < Máximo de Gerações)

5. Mostrar resultados;

5.5 CONCLUSÃO DO CAPÍTULO

Este capítulo demonstrou uma proposta de configuração de algoritmo genético para um controlador fuzzy com algumas adaptações importantes. Embora este algoritmo seja padrão do ponto de vista da literatura, a adequação deste processo ao projeto de controlador passou por decisões que estão diretamente ligadas ao desempenho e precisão do algoritmo. O importante foi destacar neste capítulo a forma de representação cromossômica que dá a garantia de que os

operadores genéticos sejam aplicados entre elementos com a mesma semântica. Além disso, destacou-se o método de avaliação, ou fitness, que explicita quais os indivíduos representam as melhores configurações para a solução do problema em questão, assim como aponta aqueles que geram elementos insatisfatórios, convergindo os parâmetros do controlador para valores mais coerentes com resposta do sistema.

CAPÍTULO 6
EXECUÇÃO DO A.G PARA SINTONIA DO CONTROLADOR *FUZZY*.

6 EXECUÇÃO DO ALGORITMO GENÉTICO PARA SINTONIA DO CONTROLADOR *FUZZY*.

6.1 INTRODUÇÃO

A criação do controlador *Fuzzy* exigiu a construção de uma plataforma recursiva de execução do sistema. A otimização dos parâmetros do controlador *Fuzzy* é realizado pelo algoritmo genético, que analisa os sinais do sistema baseado no critério da função de avaliação. A literatura menciona diversas formas de realizar a analise do sistema, porem há dois métodos mais comuns. O primeiro método de análise realiza o treinamento do controlador baseado no erro do sistema, comparando entrada e saída. O segundo método de análise realiza o treinamento do controlador baseando no erro existente entre o sistema treinado e um sistema usado como referência. Para este trabalho foi usado o segundo método tomando um sistema simples de primeira ordem como referência.

6.2 EXECUÇÃO DO ALGORITMO NA PLANTA SIMULADA

A metodologia de treinamento é realizada com o sistema não linear simulado em conjunto com as ferramentas *Fuzzy* Logic toolbox, Optimization toolbox e o ambiente Simulink, todos do Software MATLAB. Este procedimento foi preparado utilizando abordagem simples do controlador *fuzzy*, isto é, um controlador recebendo dados de erro e da derivada do erro como entrada e fornece como saída um sinal de controle aplicado diretamente na entrada da planta. O algoritmo foi realizado usando 100 gerações e com uma população inicial de 20 indivíduos. A Figura 6.2 ilustra o procedimento de treinamento. Foi aplicado o algoritmo no modelo simulado esperando que a função de *fitness* minimize entre um modelo a resposta de um modelo fictício e o sistema não linear a ser controlado.

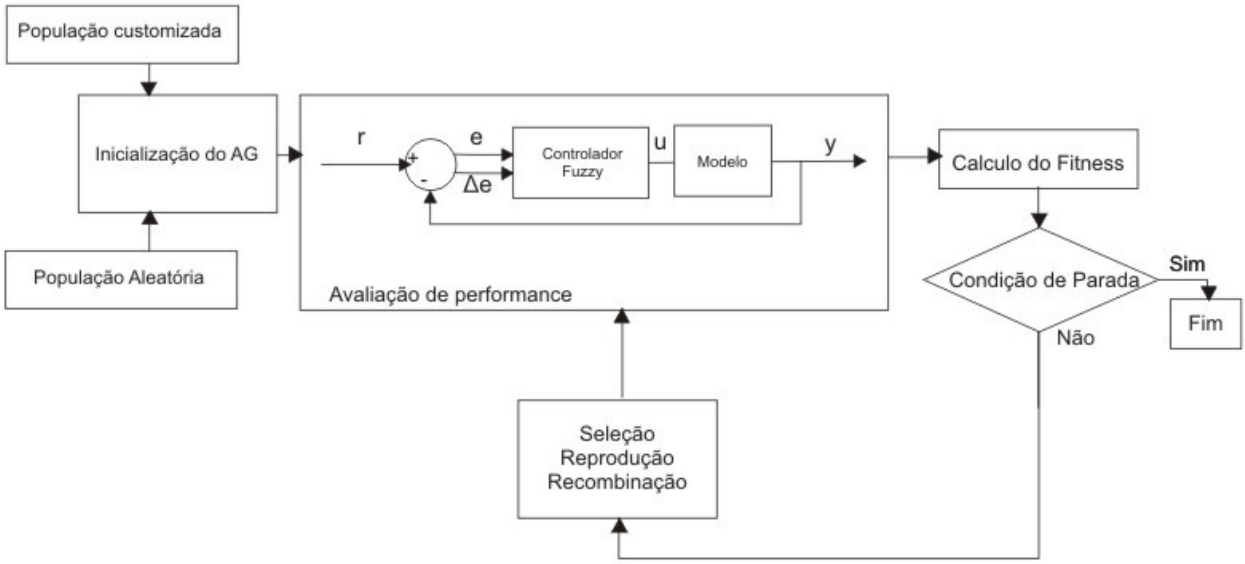


Figura 6.1 - Estrutura de Algoritmo mais usual em projeto de controladores

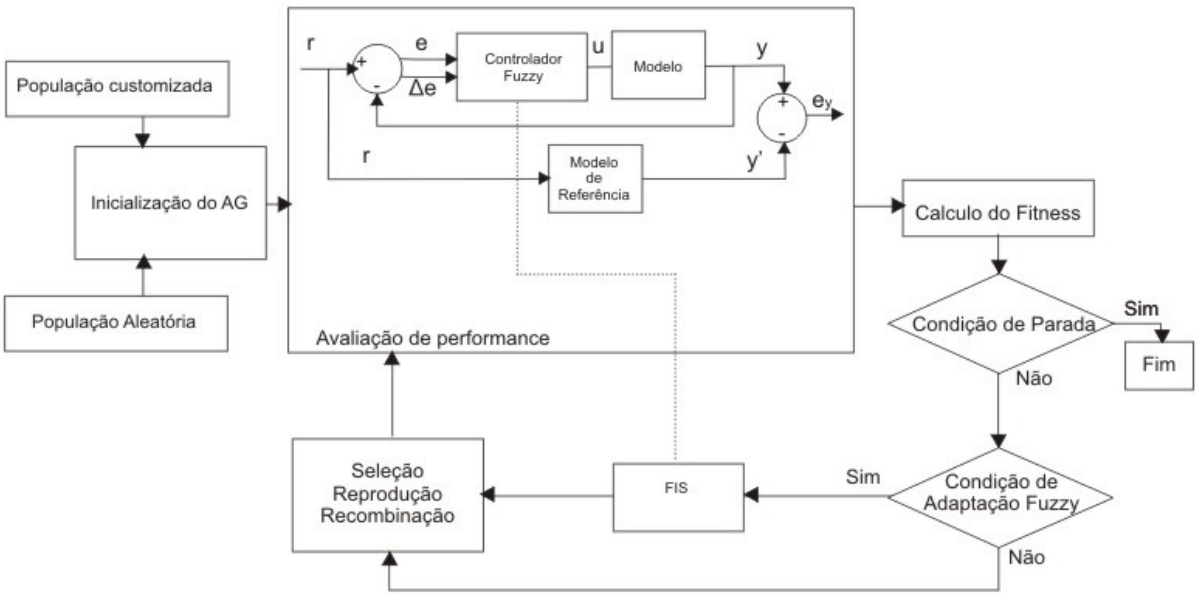


Figura 6.2 - Estrutura de Algoritmo proposta neste trabalho.

Durante a configuração, definiu-se um estado inicial para inicialização do treinamento. Como visto no Capítulo 5, o cromossomo possui 23 genes sendo 8 para otimizações de funções de pertinência e 15 para base de regras. O importante na inicialização é a definição dos limites de máximo e mínimo que cada gene pode atingir. Isso ajuda o sistema a escolher os melhores genes sem precisar realizar combinações infactíveis que apenas iriam realizar desvios durante o processo, onerar em tempo e utilizar recursos computacionais desnecessariamente.

Da mesma forma, para melhoria do desempenho do treinamento, valores dos genes do individuo inicial deve ser aproximada a valores factíveis. É a partir da combinação inicial que será criada a população inicial. Assim, a nova população terá indivíduos semelhantes ao individuo inicial convergindo à resposta do treinamento a um patamar aceitável.

Embora a configuração mostrada na Figura 6.1 seja mais usual em projetos de controlador fuzzy (SENG et al.,1999), a Figura 6.2 exemplifica bem o processo utilizado neste trabalho. O algoritmo simula o sistema diversas vezes, combinando os 20 indivíduos iniciais, durante 100 gerações, de forma a minimizar o erro existente entre o sistema de interesse e um modelo de referência, para que se tenham as melhores funções de pertinência e base de regras.

O modelo de referência utilizado possui o comportamento de um sistema de primeira ordem, cuja função de transferência é mostrada na Equação 6.1.

$$Y_{ref} = \frac{1}{10s+1} \tag{6.1}$$

A Figura 6.9 mostra o esquema do sistema implementado no Simulink. Este sistema foi simulado a uma taxa de amostragem de 0,3 s. Esta taxa é a mesma taxa utilizada pela planta real para realizar o envio e aquisição de dados. A simulação foi realizada em um microcomputador com processador Intel Dual Core de 3,5 Ghz cada núcleo, com 4GB de Memória RAM e durou em torno de 3 horas e meia. O resultado da execução é mostrado na Figura 6.3.

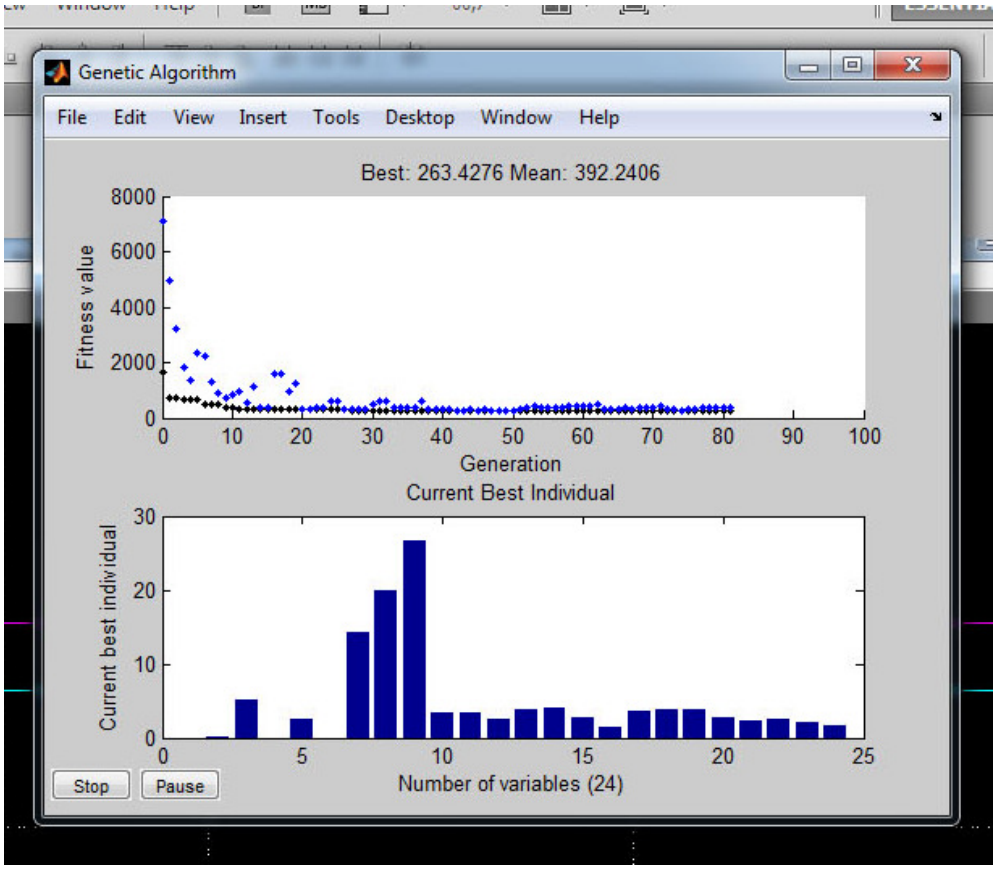


Figura 6.3 - Execução do algoritmo genético pelo MATLAB

Como mostrado no Capítulo 5, observa-se que o algoritmo genético formou funções de pertinência simétricas para a variável de entrada “erro”, mostrada na Figura 6.4, igualmente para a variável de entrada “variação do erro”, mostrada na Figura 6.5, e finalmente para a variável de saída, mostrada na Figura 6.6. A base de regra formada pelo algoritmo genético é mostrada na Figura 6.7 que formou uma superfície em 3D mostrada na Figura 6.8.

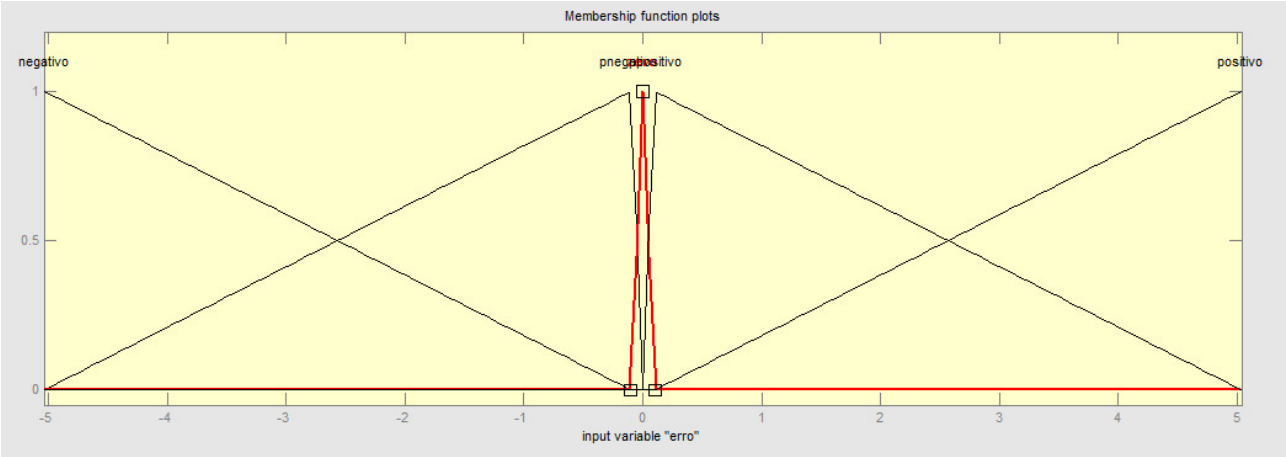


Figura 6.4 - Disposição das Funções de pertinência do erro

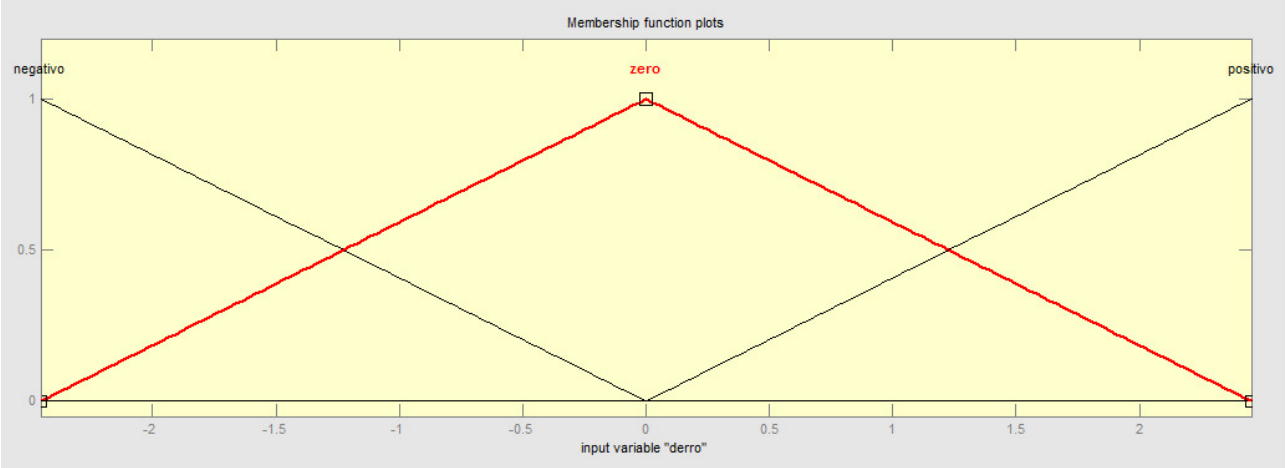


Figura 6.5 - Disposição das Funções de pertinência da variação do erro.

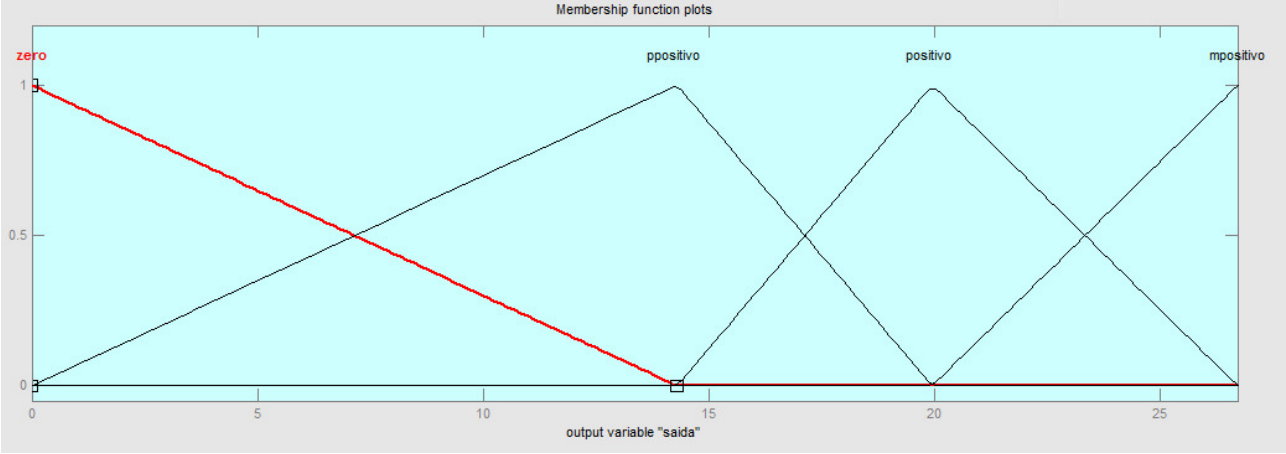


Figura 6.6 - Disposição das Funções de pertinência de saída.

1. If (erro is negativo) and (derro is negativo) then (saida is zero) (1)
2. If (erro is pnegativo) and (derro is negativo) then (saida is zero) (1)
3. If (erro is zero) and (derro is negativo) then (saida is ppositivo) (1)
4. If (erro is ppositivo) and (derro is negativo) then (saida is mpositivo) (1)
5. If (erro is positivo) and (derro is negativo) then (saida is mpositivo) (1)
6. If (erro is negativo) and (derro is zero) then (saida is ppositivo) (1)
7. If (erro is pnegativo) and (derro is zero) then (saida is zero) (1)
8. If (erro is zero) and (derro is zero) then (saida is zero) (1)
9. If (erro is ppositivo) and (derro is zero) then (saida is mpositivo) (1)
10. If (erro is positivo) and (derro is zero) then (saida is mpositivo) (1)
11. If (erro is negativo) and (derro is positivo) then (saida is positivo) (1)
12. If (erro is pnegativo) and (derro is positivo) then (saida is ppositivo) (1)
13. If (erro is zero) and (derro is positivo) then (saida is positivo) (1)
14. If (erro is ppositivo) and (derro is positivo) then (saida is ppositivo) (1)
15. If (erro is positivo) and (derro is positivo) then (saida is ppositivo) (1)

Figura 6.7 - Base de regras gerada pelo algoritmo genético.

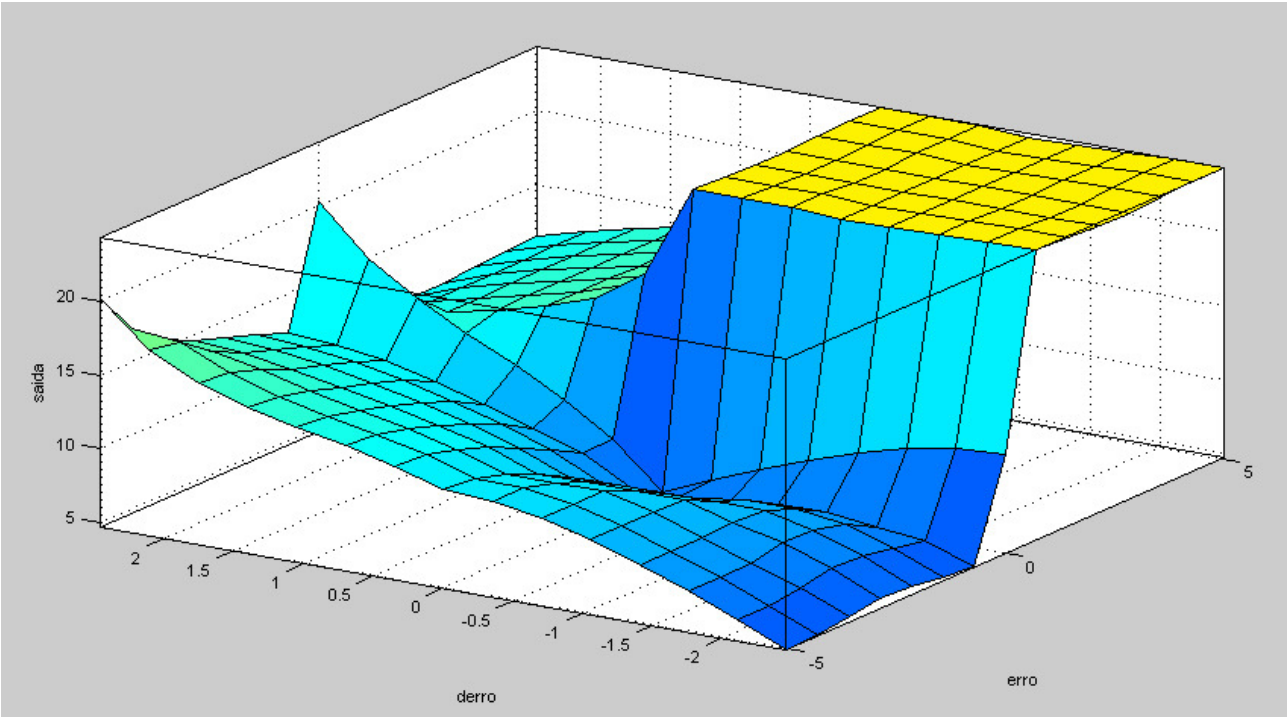


Figura 6.8 - Superfície 3D das regras.

6.3 RESPOSTA DO SISTEMA OBTIDAS NA PLANTA SIMULADA

A próxima etapa é realizar a validação do sistema, realizando alguns ensaios simulados no próprio Simulink/Matlab.

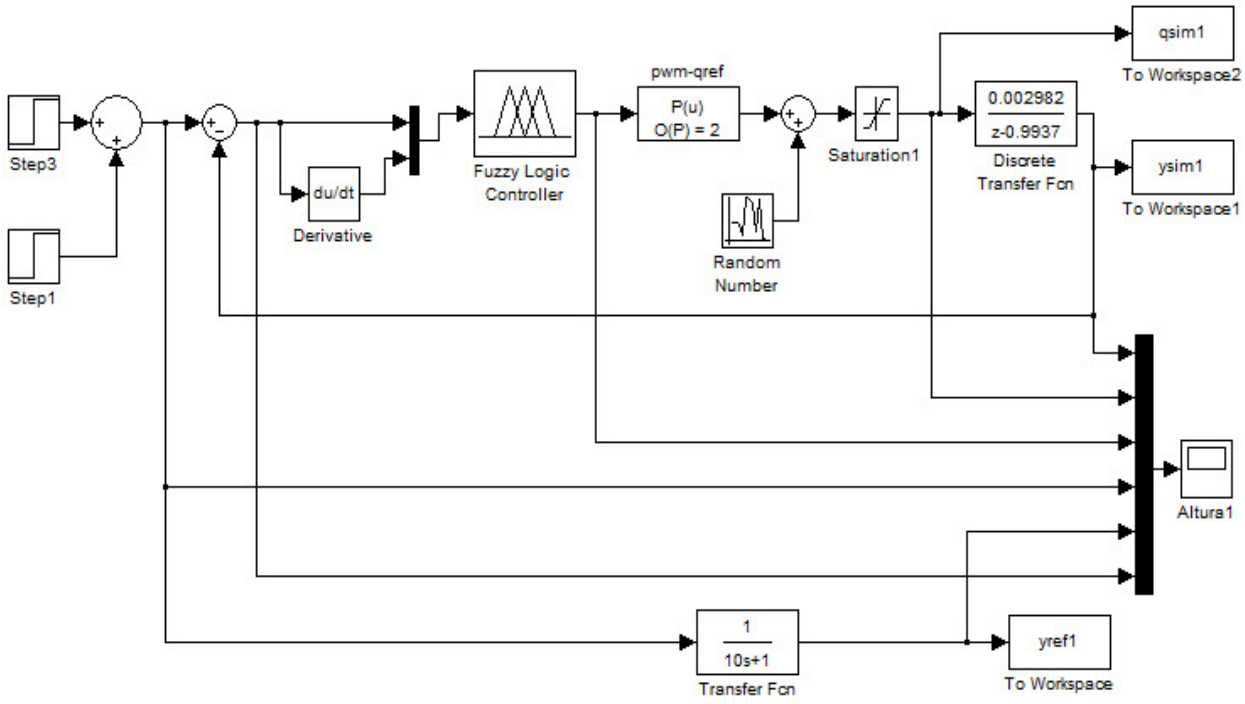


Figura 6.9 - Implementação do sistema completo no Simulink.

Para obtenção da resposta do sistema na planta simulada, o procedimento é muito simples. Basta executar a simulação, mostrada pela Figura 6.9. Para testar a regulação do controlador, adiciona-se um degrau na saída do sistema e, então, provoca-se um distúrbio, forçando o controlador a atuar sobre a planta corrigindo o erro. Foram realizadas simulações de rastreamento de referência e de distúrbios com características de vazamento e entupimento do sistema. É importante destacar que, para a simulação, o modelo de referência (Equação 6.1) é colocado paralelamente ao sistema para efeitos comparativos entre os sinais de saída

As próximas figuras mostram os resultados simulados do controlador já sintonizado. É importante frisar que ambos sistemas foram submetidos ao mesmo sinal de entrada. Nos resultados da simulação, os sinais são comparados para avaliar as semelhanças nos desempenhos.

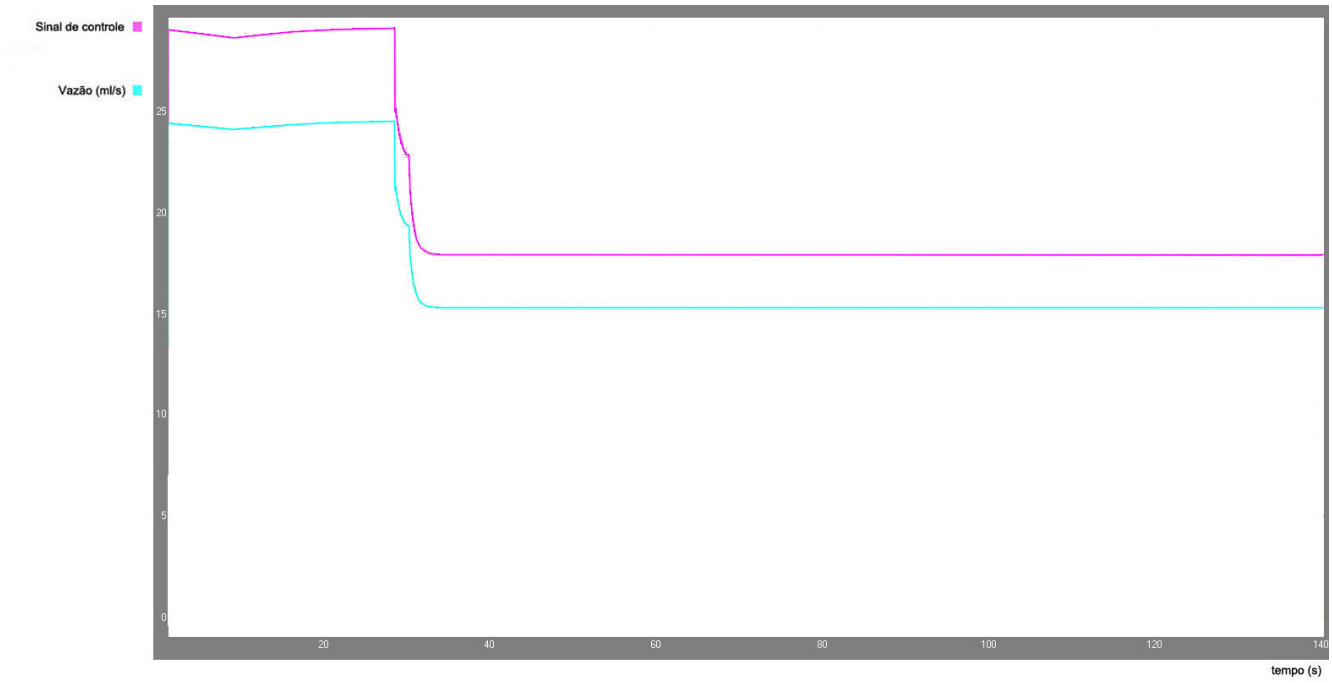


Figura 6.10.a – Sinal de controle do sistema em degrau de referência 5 cm

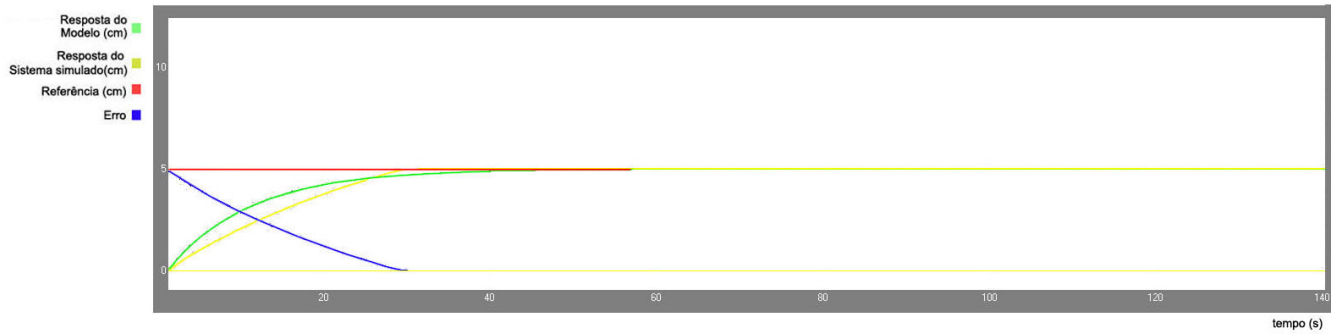


Figura 6.10.b - Resposta ao degrau de referência 5 cm

As Figuras 6.10.a e 6.10.b mostra os resultados da simulação aplicando um degrau de 5 cm no sistema. O mesmo degrau é aplicado no Modelo de Referência (Figura 6.10.b). Observa-se que o tempo de subida em ambos os sistemas é bastante semelhante, atingindo o regime permanente em torno de 30 segundos. Entretanto, no sistema simulado com controlador, a resposta é ligeiramente mais rápida. A Figura 6.10.a mostra a reação do sistema com base nos sinais de controle e vazão.

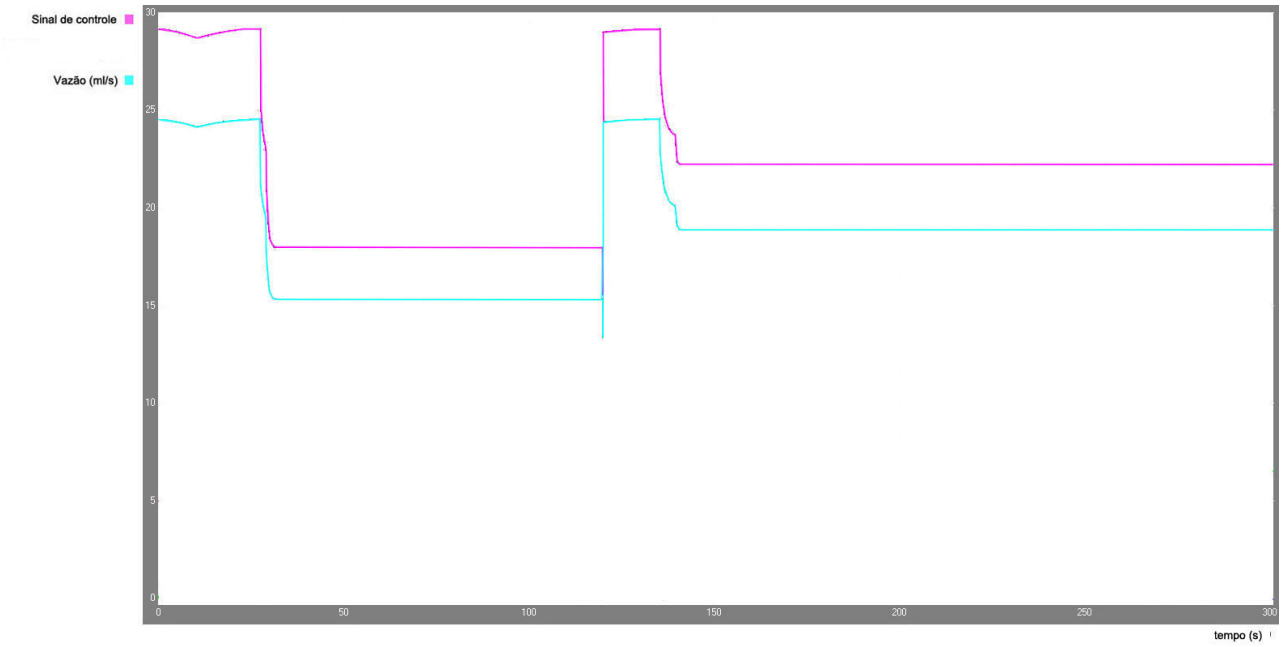


Figura 6.11.a – Sinal de controle e vazão do sistema em mudança de referência de 5 cm para 6 cm

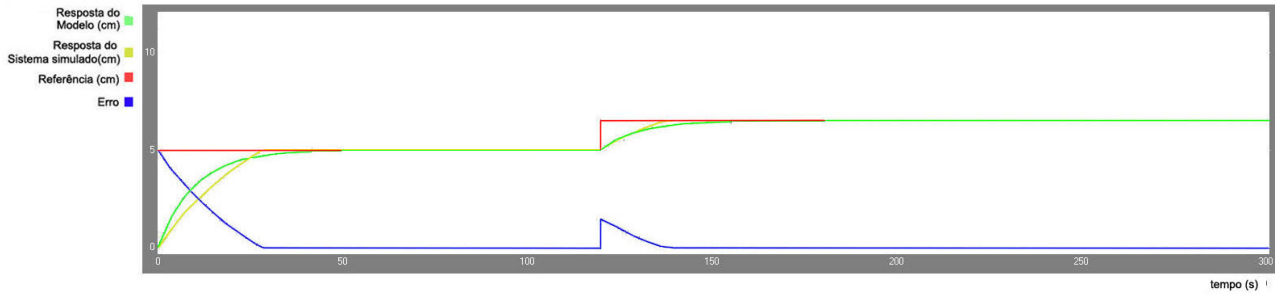


Figura 6.11.b - Resposta ao degrau com mudança de referência de 5 cm para 6 cm

As Figuras 6.11.a e 6.11.b são os resultados de uma simulação com mudança de referência. Um degrau de 5 cm é aplicado na entrada de ambos sistemas até atingirem o regime permanente. No instante 120 segundos, alterou-se o degrau, aumentando esse patamar para 6 cm. Neste instante, o sistema atua de forma a corrigir o erro e encontrar o novo patamar de referência.

Novamente, o sistema simulado se comportou de maneira semelhante ao modelo de referência, entretanto, com uma correção ligeiramente mais rápida, tanto nos instantes iniciais quanto na mudança de referência.

Uma simulação semelhante às Figura 6.11.a e 6.11.b tem os resultados mostrados na Figura 6.12.a e 6.12.b. O procedimento foi o mesmo, entretanto, a mudança de referência foi alterada para um patamar menor, de 5 cm para 4 cm. Um degrau de 5 cm é aplicado na entrada de ambos sistemas até atingirem o regime permanente. No instante 120 segundos, alterou-se o degrau, diminuindo esse patamar para 4 cm.

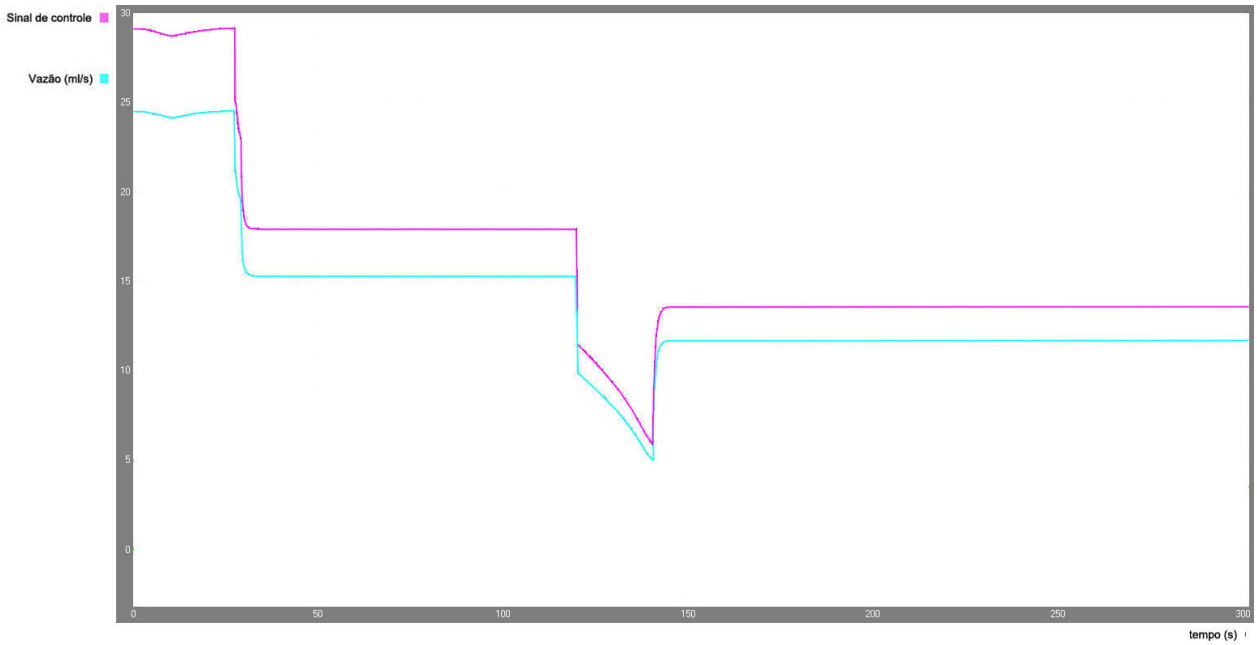


Figura 6.12.a - Sinal de controle e vazão do sistema em mudança de referência de 5 cm para 4 cm

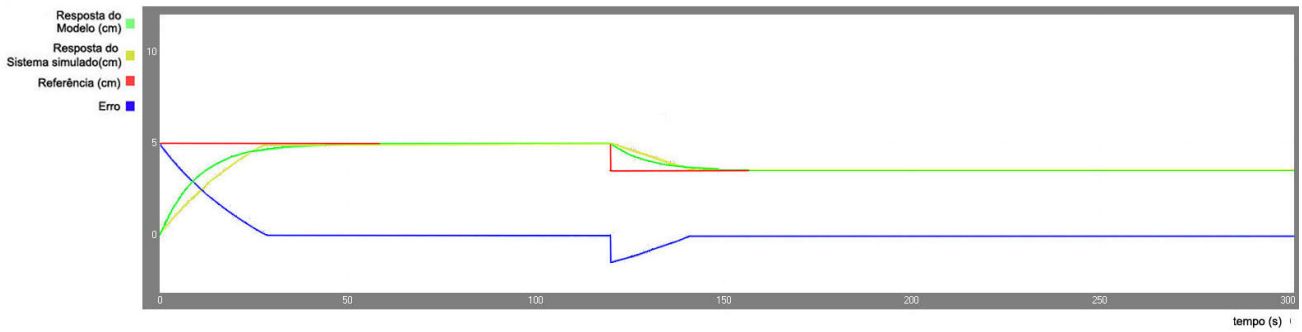


Figura 6.12.b - Resposta ao degrau com mudança de referência de 5 cm para 4 cm

Da mesma forma, assim como na Figura 6.11.b, o sistema conseguiu rastrear a nova referência, como mostra a Figura 6.12.b. O sistema simulado se comportou de maneira semelhante ao modelo de referência, com a ressalva de que foi ligeiramente mais rápido, tanto nos instantes iniciais, na subida do nível, quanto na mudança de referência, na descida do nível.

Observa-se que nos casos de mudança de referência das Figuras 6.11.a e 6.12.a, há a atuação do controlador (linhas lilás) e da clara mudança da vazão (linhas azuis) nos instantes em que foi exigido realizar a correção do nível.

Outros ensaios foram necessários para avaliar se o controlador é capaz de reagir de forma eficaz às perturbações externas. Assim, foram experimentados distúrbios de entupimento e vazamento nos sistemas simulados. Esses distúrbios forma aplicados somente no modelo com controlador (linha amarela) e não foram aplicados ao modelo de referência (linha verde). Por isso, as figuras da simulação com distúrbios não mostram quaisquer alterações no sinal.

Figura 6.13.a - Sinal de controle e vazão do sistema com perturbação de entupimento.

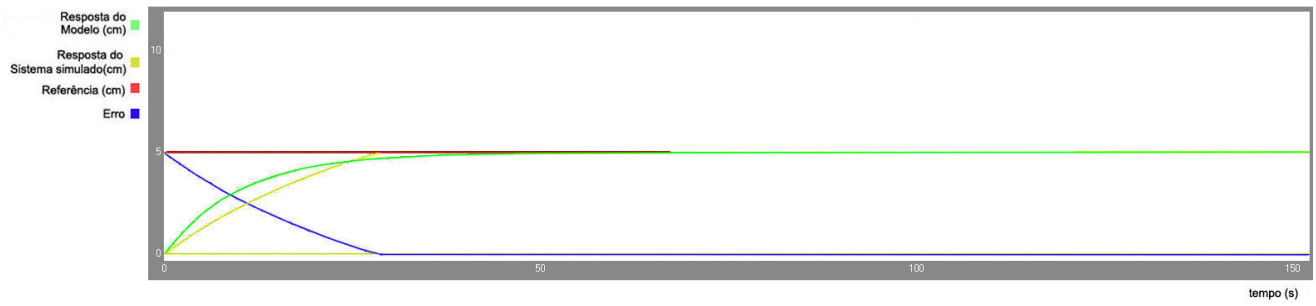


Figura 6.13.b - Resposta ao degrau na referência de 5 cm com perturbação de entupimento.

Nesta simulação, mostrada nas Figuras 6.13.a e 6.13.b, foi aplicado um degrau de referência 5 cm na entrada do sistema simulado. O modelo de referência recebe o mesmo degrau para efeitos de comparação. No instante 120 segundos, é realizado efeito de entupimento parcial abrupto na saída do sistema simulado, mantendo-se esse entupimento até o final da simulação. Antes disso, tanto o sistema simulado quanto o modelo de referência, inicialmente, responderam normalmente e estava em regime permanente.

Observa-se que, no instante 120 segundos, o sinal de controle atua de forma a diminuir a vazão, uma vez que não é mais necessário um volume maior para manter o nível. A Figura 6.14 mostra com mais detalhes somente os sinais de saída desta simulação.

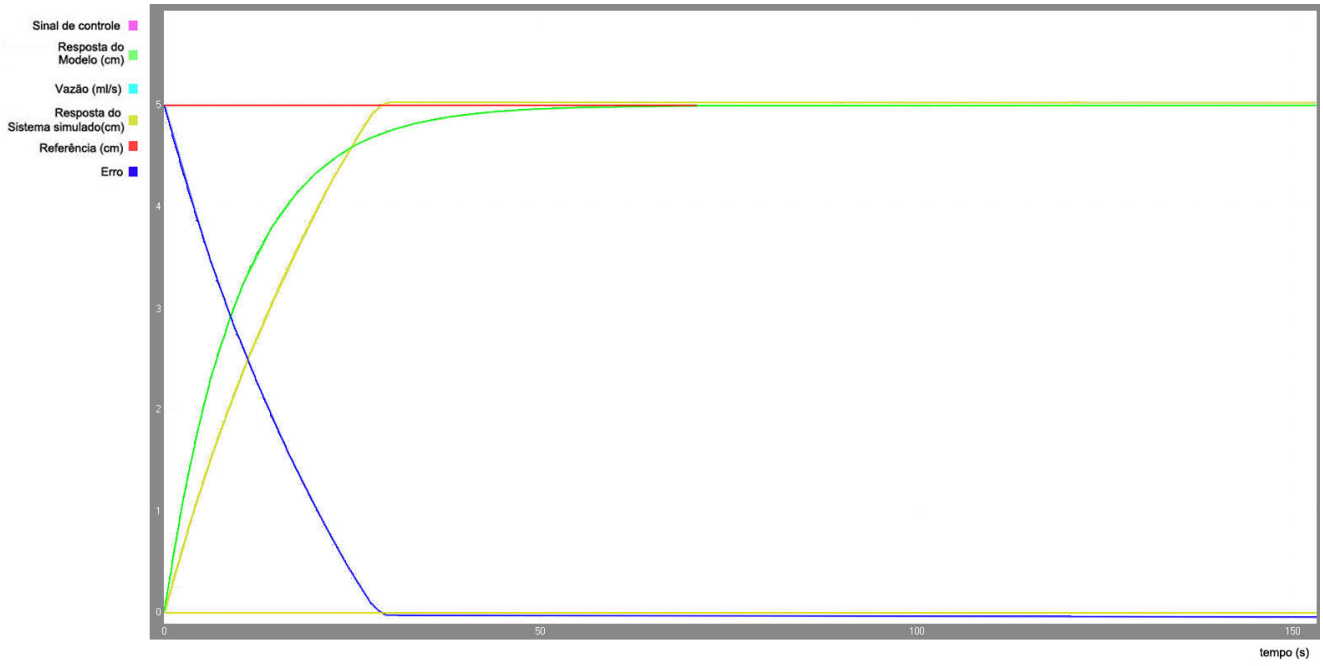


Figura 6.14 - Resposta em detalhe da perturbação de entupimento.

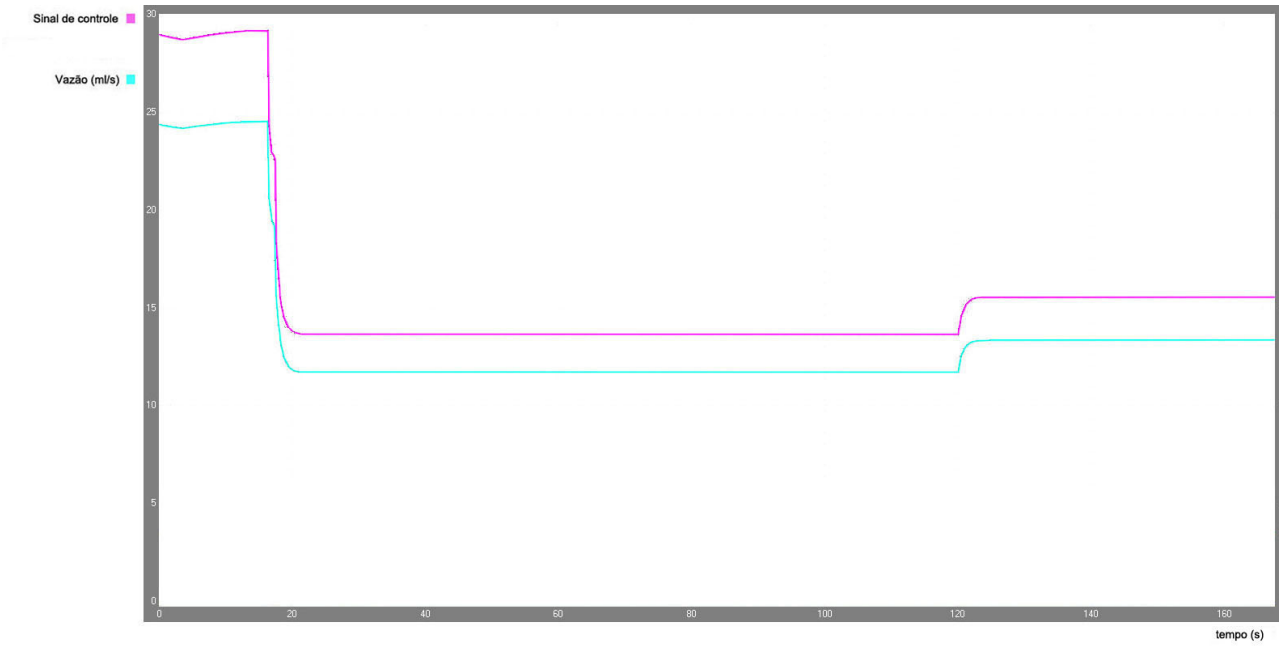


Figura 6.15.a - Sinal de controle e vazão do sistema com perturbação de vazamento.

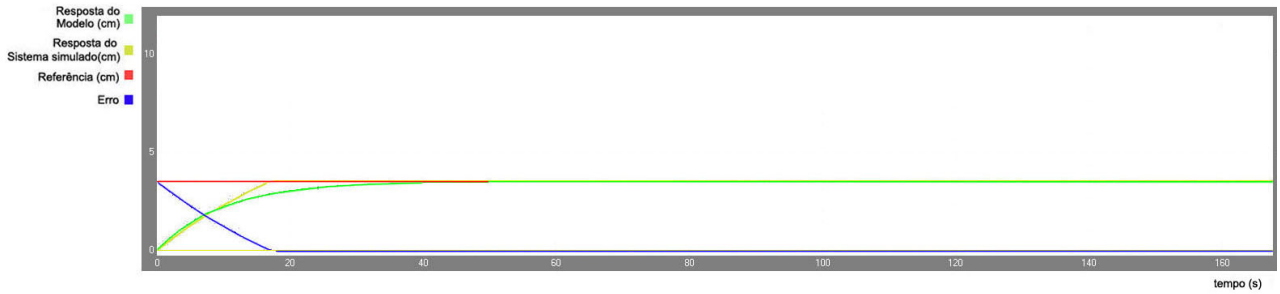


Figura 6.15.b - Resposta ao degrau na referência de 5 cm com perturbação de vazamento.

Realizou-se uma ultima simulação, mostradas nas Figura 6.13.a e 6.13.b, de forma semelhante à simulação nas figuras 6.15.a e 6.15.b, onde de foi aplicado um degrau de referência 5 cm na entrada do sistema simulado e no modelo de referência, que recebe o mesmo degrau para efeitos de comparação. Tanto o sistema simulado quanto o modelo de referência, inicialmente, responderam normalmente e estavam em regime permanente.

No instante 120 segundos, é realizado efeito de vazamento parcial abrupto na saída do sistema simulado, mantendo-se esse vazamento até o final da simulação.

Nota-se, no instante 120 segundos, o sinal de controle atuando de forma a aumentar a vazão, em um esforço de compensação do volume, tentando mantê-lo estável.

6.4 RESPOSTAS DO SISTEMA OBTIDAS NA PLANTA REAL

Para ter a certeza do real funcionamento da técnica, foi necessário implementar o sistema *Fuzzy* já treinado no sistema real de nível de líquido demonstrado no capitulo 4. Dessa forma, o arquivo *.fis do MATLAB utilizado na simulação, ora criada pelo Algoritmo Genético, foi o mesmo utilizado para ensaio real de forma que este ultimo pudesse validar a técnica gerada computacionalmente. Devido a praticidade técnica, optou-se em realizar o ensaio com o suporte de um microcomputador, em que, através de uma comunicação serial, realiza-se a captura e enviam-se os dados ao circuito. Isso é feito através de um programa simples, feito em linguagem MATLAB, e que é mostrado no Anexo 1. O fluxograma do sistema é mostrado na Figura 6.16.

Primeiramente, o programa solicita o nível de referência desejado. Em seguida, o sistema captura informações de nível e de vazão. Com essas informações, o programa calcula o erro, a derivada do erro, e introduz no sistema *Fuzzy* através do comando *evalfis*. Este último executa seus processos internos de avaliação e responde com um valor correspondente a uma escala proporcional ao valor de PWM utilizado para controlar as bombas. Esse processo é realizado com uma taxa de amostragem de 0,3 s.

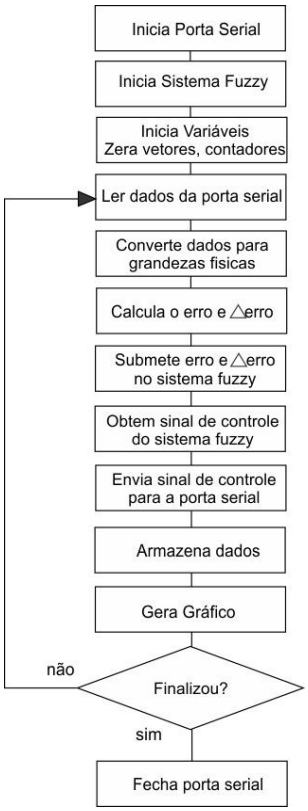


Figura 6.16 - Fluxograma do programa de comunicação serial entre controlador e planta.

Assim que a porta serial é habilitada e o programa iniciado, o protótipo começa a funcionar de acordo com a referência definida pelo usuário. Todos os ensaios realizados neste trabalho, foram feitos com referencia de 5 cm. A validação do sistema real é realizada de forma semelhante ao sistema simulado, com os mesmos ensaios. Os resultados estão demonstrados nas figuras a seguir.

Nelas os pontos em verde do sinal de controle PWM enviadas pelo controlador às bombas, os pontos em lilás correspondem à vazão, os pontos em azul representam o nível do tanque e em preto é mostrado o erro.

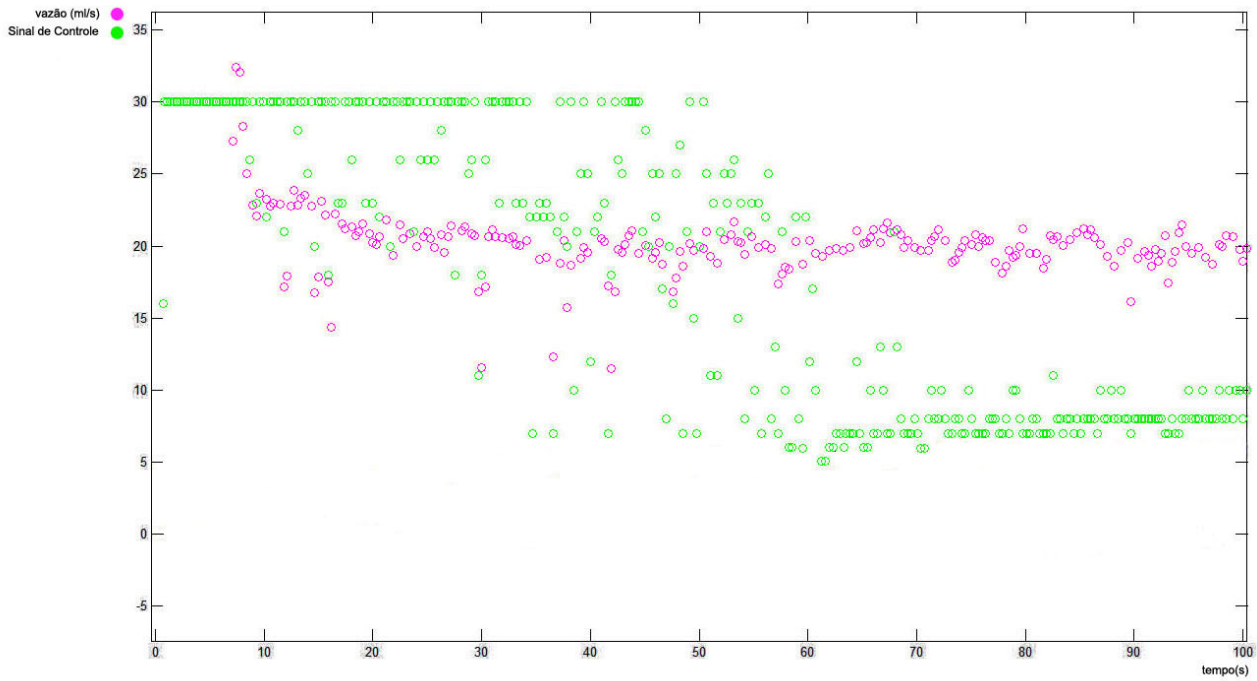


Figura 6.17.a - Sinal de controle e vazão em degrau de referência 5 cm.

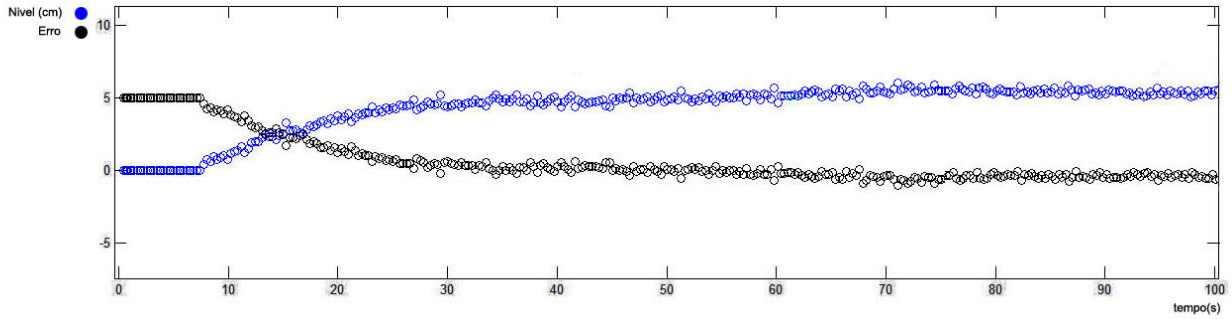


Figura 6.17.b - Resposta ao degrau de referência 5 cm

As Figura 6.17.a e 6.17.b correspondem à resposta do sistema após um degrau de referência de 5 cm. Observa-se que o sistema, representado pela circunferência em azul, responde de forma satisfatória chegando ao nível desejado em um tempo bastante semelhante ao tempo mostrado na simulação (Figura 6.10 e Figura 6.11). Observa-se que a vazão (círculos em lilás) diminui e se estabiliza conforme o nível do tanque se aproxima do nível desejado.

As Figuras 6.18.a e 6.18.b mostram a resposta ao degrau do sistema em uma situação de mudança de referência. O objetivo deste ensaio é observar a reação do sistema quando houver necessidade do modificar o nível do tanque. Neste ensaio, o

sistema foi inicialmente submetido a um degrau de 4 cm e, 60 segundos depois, o degrau muda de patamar e sobe para 6 cm.

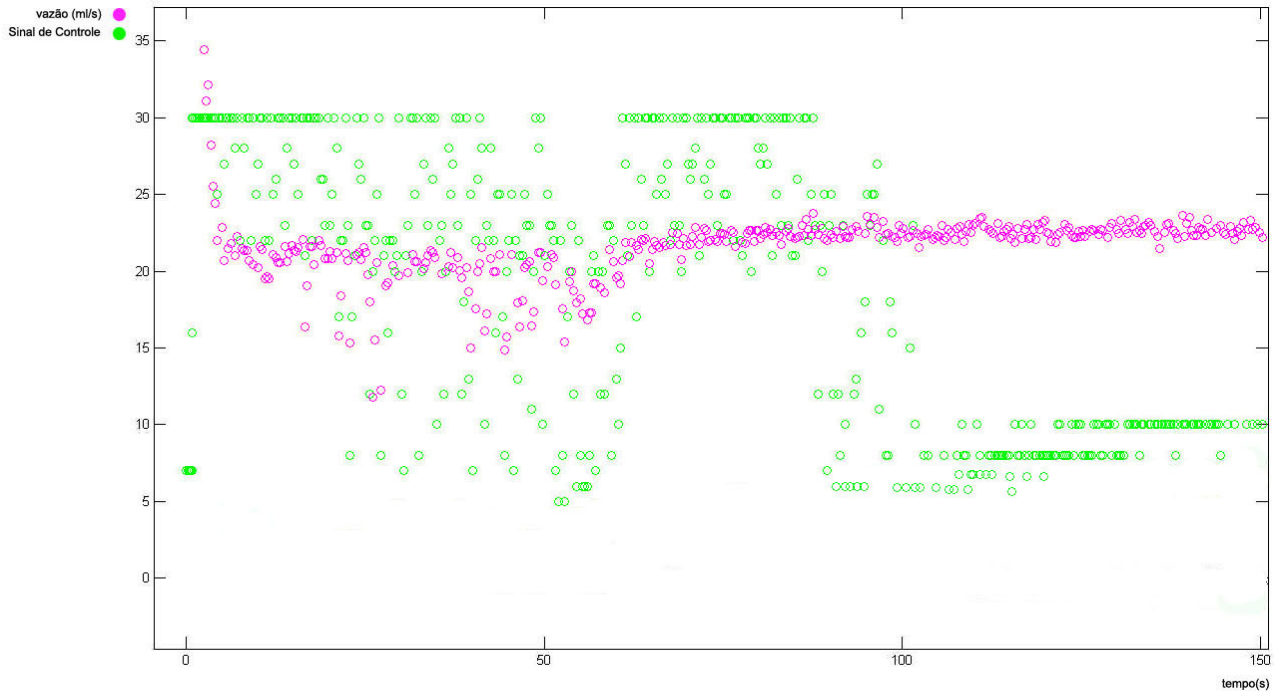


Figura 6.18.a – Sinal de controle e vazão ao degrau com mudança de referência de 4 cm para 6 cm.

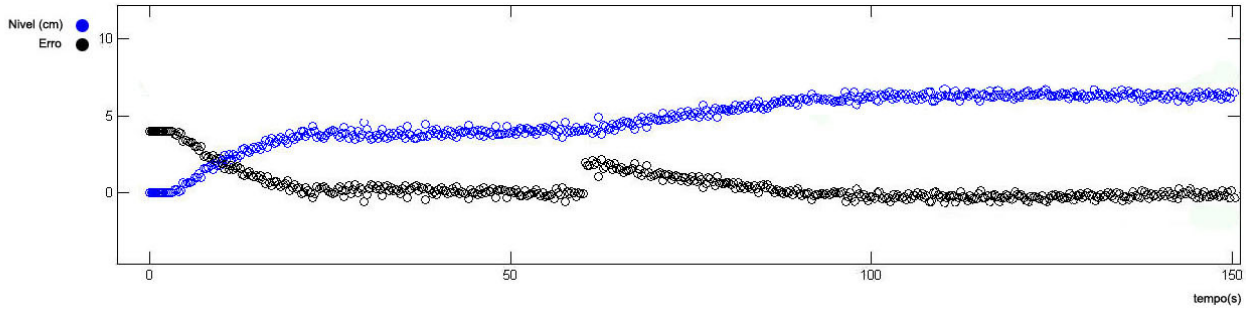


Figura 6.18.b - Resposta ao degrau com mudança de referência de 4 cm para 6 cm

Observa-se pela Figura 6.18.b que o nível do sistema (círculos em azul) altera-se satisfatoriamente, subindo a 4 cm em 30 segundos aproximadamente e mantendo-se estável neste nível até 60 segundos, instante em que há mudança de referência. Neste momento, o controlador detecta a mudança de referência através do erro e atua sobre as bombas aumentando a vazão, alterando o nível e estabilizando novamente em 6 cm.

Note-se a alteração no padrão do sinal de vazão na Figura 6.18.a (círculos em lilás) no momento em que há a detecção da mudança de referência antes e depois do instante 60 segundos significando que o controlador atuou de forma a alterar o estado do sistema.

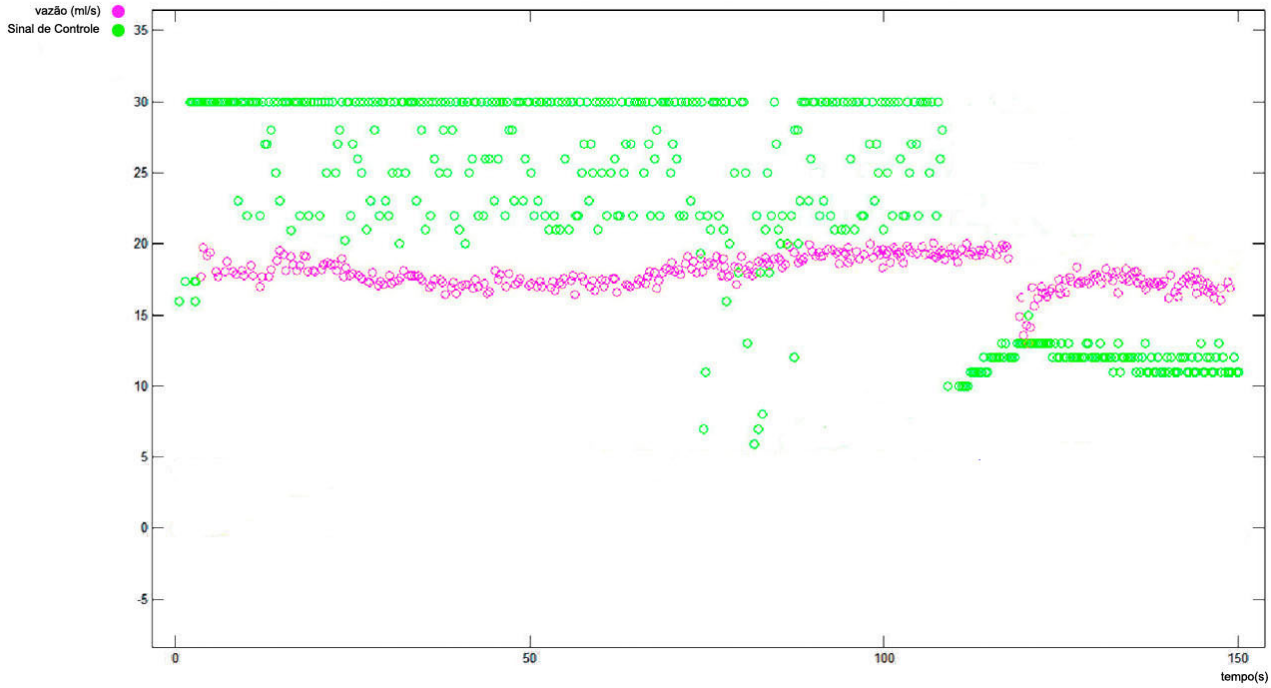


Figura 6.19.a – Sinal de controle e degrau em referência de 4 cm com perturbação de entupimento.

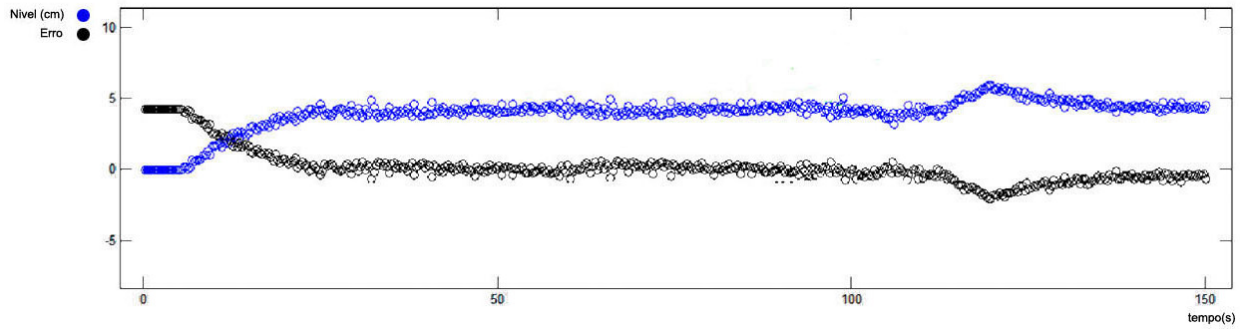


Figura 6.19.b - Resposta ao degrau: Referência de 4 cm com perturbação de entupimento.

As Figuras 6.19.a e 6.19.b correspondem à resposta do sistema dada uma determinada perturbação externa. Inicialmente, o ensaio foi realizado aplicando um degrau de 4 cm na entrada do sistema. O nível se estabiliza normalmente em 4 cm. A partir do instante 110 segundos, aplica-se uma perturbação de entupimento abrupto, simulando um degrau, na saída do sistema. O entupimento é mantido até o fim do ensaio. Observa-se que, neste instante, o nível do sistema eleva-se por um

curto período. O controlador detecta a perturbação através do erro, e atua de forma a corrigi-lo, estabelecendo novamente o nível na referência desejada.

Tem-se, também, a diminuição da vazão (círculos em lilás), e a atuação do controlador mudando o sinal PWM (círculos em verde), depois que foi detectado o entupimento de forma a corrigir o nível.

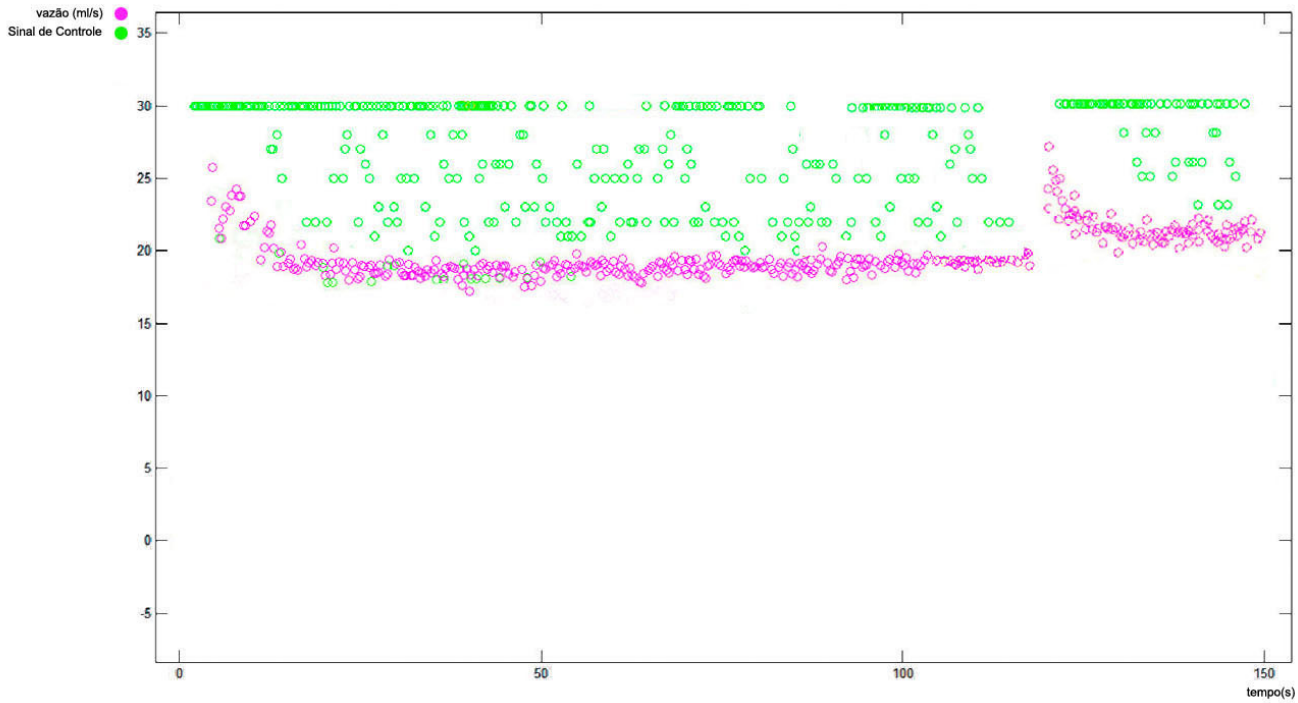


Figura 6.20.a – Sinal de controle e vazão em referência de 4 cm com perturbação de vazamento.

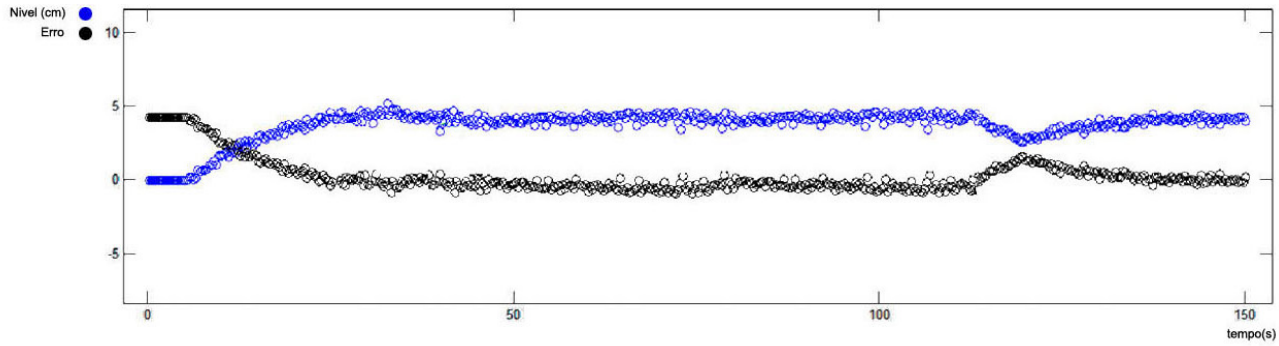


Figura 6.20.b - Resposta ao degrau: Referência de 4 cm com perturbação de vazamento.

As Figuras 6.20.a e 6.20.b correspondem à resposta do sistema dada outro tipo de perturbação externa. Novamente, o ensaio foi realizado aplicando um degrau de 4 cm na entrada do sistema e o nível se estabilizando normalmente em 4 cm. A partir do instante 110 segundos, aplica-se uma perturbação de vazamento abrupto, simulando um degrau, na saída do sistema. Este vazamento é mantido até o fim do ensaio. Neste instante, o nível do sistema declina-se por um curto período. O controlador detecta a perturbação através do erro, e atua de forma a corrigi-lo, estabelecendo novamente o nível na referência desejada.

De forma oposto ao ensaio de ensaio de entupimento das Figuras 6.19.a e 6.19.b, houve o aumento da vazão (círculos em lilás), e a atuação do controlador alterando o sinal PWM (círculos em verde), depois que foi detectado o vazamento (Figura 6.20.a).

6.5 CONCLUSÃO DO CAPÍTULO

O objetivo deste capítulo foi mostrar a execução do algoritmo genético e obtenção de sistema fuzzy, além da implementação do mesmo em ambiente virtual e em tempo real através de um protótipo. Este capítulo também realiza um comparativo entre o projeto simulado e o ensaio em tempo real, comparando os sinais e as respostas em situações de rastreamento simples e sob efeitos de distúrbios.

CAPÍTULO 7
CONCLUSÃO

7 CONCLUSÃO

O objetivo da presente dissertação foi sugerir um método mais simples e de acentuada utilização prática e funcional para a tarefa de projeto de controladores *fuzzy*, indicando uma possível direção para a realização desta última tarefa, ou mais especificamente, para a obtenção de parâmetros da estrutura de controladores *fuzzy* de forma automatizada.

O método adotado foi um Algoritmo Genético que consiste em um método de otimização portador de uma base teórica bem consolidada na literatura acadêmica e com uma vasta gama de aplicações práticas. Com uma configuração simples, e uma função fitness baseada em um conceito dos controladores PID, foi possível sintonizar e obter parâmetros para um controlador *fuzzy* com o objetivo de controlar um sistema dinâmico de tanque de característica não linear.

O protótipo foi construído pelo autor especialmente para fins didáticos. Entretanto, o método pode ser utilizado em outros tipos de planta, que aqui, não se teve oportunidade de realizar. Neste trabalho, foi utilizado o suporte computacional do Software MATLAB juntamente com seu pacote de simulação SIMULINK e a comunicação serial que possibilitou a interface entre o sistema implementado e a planta real. Não foi utilizada uma programação *fuzzy* embarcada neste projeto.

O protótipo foi identificado e modelado matematicamente para a aplicação do Algoritmo Genético no MATLAB. Ao finalizar, o método obteve um controlador pronto para ser utilizado. Primeiramente, o controlador foi testado no modelo matemático com o suporte do Simulink onde os resultados se aproximaram do esperado.
Finalmente, o controlador foi submetido à planta real com suporte da comunicação serial. Os ensaios, simulado e real, foram feitos com os objetivos de:

- rastreamento de referência
- mudança de referência
- entupimento abrupto
- vazamento abrupto.

Os resultados obtidos para o controle do protótipo foram bastante semelhantes aos resultados simulados, com grau de autonomia satisfatório. É conclusivo que, ao se utilizar a metodologia proposta neste trabalho, pode ser confirmado uma razoável eficiência.

A qualidade dos parâmetros do controlador foi avaliada tanto pela disposição das funções de pertinência, da base de regras obtida e, principalmente, pela comparação das respostas fornecidas através dos ensaios com o modelo simulado e com o sistema real, ambos sujeitos a um mesmo sinal de entrada.

Nos ensaios realizados, a confiabilidade nas implementações computacionais pode ser apoiada pela verificação do comportamento esperado do algoritmo genético (queda da diversidade da população e valor médio da função fitness com o passar das gerações) e pela obtenção de modelos satisfatórios para os sistemas estudados, resultado este que confirma ainda a eficiência da metodologia aqui proposta.

A otimização de alguns parâmetros dos modelos já foi abordada em diversos outros trabalhos na literatura acadêmica (LISKA e MELSHEIMER, 1994, NG e LI, 1994, HOMAIFAR e MCCORMICK, 1995, SENG et al, 1999) , incluindo a forma e localização das funções de pertinência e a base de regras de inferência.

Sendo assim, as contribuições deste trabalho no contexto acima são:

- Automatização do processo de sintonia e obtenção de parâmetros de controlador *fuzzy*.
- Funções de pertinência de formato predefinido.
- Comportamento final baseado em um modelo de referência de característica própria externo ao sistema.
- Utilização de uma função fitness baseada em um conceito PID.
- *Crossover* aritmético aplicado separadamente nas funções de pertinência e base de regras

- A utilização do mesmo controlador *fuzzy* implementado no modelo matemático e planta real.
- A comparação e obtenção de resultados satisfatórios no modelo matemático simulado e na planta real.

As motivações para continuidade deste trabalho podem ser as implementações deste método em processos de 2ª ordem, ou processos com polos instáveis. Outras possibilidades seriam as elaborações de representações cromossômicas flexíveis, que permitissem o aumento e diminuição do número de genes, o que alteraria a quantidade ou conjunto de funções de pertinência de tal forma que a base de regras completa fosse composta por um numero de regras proporcionais. Seria possível realizar um estudo implementando um conjunto de base de regras não completas. Para sistemas reais com regiões de operação distintas, seria de imensa utilidade a descoberta de sistemas que se adequem a modelos locais lineares e não lineares e que permitisse transição suave entre os modelos, para sistema *fuzzy* do tipo Takagi-Sugeno inclusive.

REFERÊNCIAS BIBLIOGRÁFICAS

REFERÊNCIAS BIBLIOGRÁFICAS

ACHY, A. R. A., FONTES, A. B., M. R. S. GARCIA. **Avaliação de Técnicas de Controle Utilizando uma Plataforma Multivariável de Controle de Nível desenvolvida para o Ensino e Pesquisa**. CBA 2012.

AGUIRRE, L. A. **Introdução a Identificação de Sistemas**, 2º Edição, Editora UFMG. 2004.

ANDRADE, J.F.B. **Controle Multivariável em Sistemas de Nível de Líquidos com Tanques Interligados.** Trabalho de Conclusão de Curso. UFPA. 2008.

BRAGA, A. P., LUDERMIR, T. B. E CARVALHO, A. C. P. L. F. **Redes Neurais Artificiais.** LTC - Livros Técnicos e Científicos Editora. 2000.

BERNARDES, M. C., MELO, G. A. F., BORGES G. A., FREITAS, A. A., BAUCHSPIESS, A., **Instrumentação e Identificação Não-Linear de um Sistema de Nível de Líquido com Quatro Tanques Interligados.** CBA, 2006.

BERNARDES, M. C., MELO, G. A. F., **Instrumentação e Controle de uma maquete de Nível de Líquido com Quatro Tanques Interligados.** Trabalho de Conclusão de Curso. UNB. 2006.

CAMPOS, M. C., **La simulation Dynamique et L' Intelligence Artificielle Delaboration de Strategies de Controle-Commande Multivariable Flou. Aplication a une Unite de Cracage Catalytique en Phase Fluide,** Tese de Doutorado, Paris, 1997.

DSPIC® LANGUAGE TOOLS LIBRARIES. Microchip Technology Inc. 2004.

DSPIC30F2011/2012/3012/3013 DATA SHEET. Microchip Technology Inc. 2004.

FLOW SENSOR FHKU 938-6500/01 G1/2" 100 ARNITE. DATASHEET. Digmesa. Switzerland.

FREESCALE SEMICONDUCTOR, **Liquid Level Control Using a Pressure Sensor.** Application Notes. 2007.

FREIRE, R. C. S. **Instrumentação eletrônica**, Technical report, Universidade Federal da Paraíba, 2002.

GOSMANN, H. L. **Um sistema Multivariável de Tanques Acoplados para a Avaliação de Técnicas de Controle**. Dissertação de Mestrado, Publicação 143/2002, Departamento de Engenharia Elétrica, Universidade de Brasília, 2002.

GREGA, W. AND MACIEJCZYK, A. **Digital Control of a Tank System.** IEEE Transactions on Education, Vol. 37, No. 3. 1994.

HERRERA, F. AND LOZANO, M. **Adaptation of Genetic Algorithm Parameters Based on Fuzzy Logic Controllers**. Genetic Algorithms and Soft Computing, Physica-Verlag. 1996.

HOMAIFAR, A. E MCCORMICK, E. **Simultaneous Design of Membership Functions and Rule Sets for Fuzzy Controllers Using Genetic Algorithms.** IEEE Transactions on Fuzzy Systems, Vol. 3, Issue 2, 129-139. 1995.

HOLLAND, J. H. **Adaptation in Natural and Artificial Systems**. Ann Arbor,MI: University of Michigan Press. 1975.

JAIN, L. C. E JAIN, R. K. **Advances in Fuzzy Systems - Applications and Theory.** Hybrid Intelligent Engineering Systems, Vol. 11, World Scientific. 1997.

LEE, M. A. E TAKAGI, H. **Integrating Design Stages of Fuzzy Systems using Genetic Algorithms**. Second IEEE International Conference on Fuzzy Systems, San Francisco, CA, USA, 1993.

LIN C.T. AND LEE C. S. G., **Neural Fuzzy Systems: A Neural-Fuzzy Synergism to Intelligent Systems**. Englewood Cliffs, NJ: Prentice-Hall, May 1996.

LISKA, J. E MELSHEIMER, S. S. **Complete Design Of Fuzzy Logic Systems Using Genetic Algorithms**. IEEE World Congress on Computational Intelligence, Proceedings of the Third IEEE Conference on Fuzzy Systems, Vol. 2, 1377-1382. 1994.

MATLAB USER`S GUIDE, Mathworks Inc, 1999.

MEDEIROS, A. V., MAITELLI, A. L. E GABRIEL FILHO, O. **Otimização das Funções de Pertinência de um Controlador Nebuloso utilizando Algoritmos Genéticos**. V Simpósio Brasileiro de Automação Inteligente, Canela, RS.2001.

MENDEL, J. M., **Fuzzy Logic Systems for Engeneering: A Tutorial**, Proceedings of the IEEE, vol. 83, nº 3, 1995.

MICHALEWICZ, Z. **Genetic Algorithms + Data Structures = Evolution Programs**. 3º Edição, Springer-Verlag.1996.

NISE, NORMAN S. **Engenharia de Sistemas de Controle.** 3ª Edição. Rio de Janeiro: Livros Técnicos e Científicos Editora S.A, 2002.

NG, K. C. E LI, Y. **Design of Sophisticated Fuzzy Logic Controllers Using Genetic Algorithms**. IEEE World Congress on Computational Intelligence, Proceedings of the Third IEEE Conference on Fuzzy Systems, Orlando, FL, USA, Vol. 3, 1708-1712. 1994.

OGATA, K., **Engenharia de controle moderno**, Prentice-Hall do Brasil, 1998.

PADILHA, P. C. C., **Desenvolvimento de uma metodologia de sintonia de controladores fuzzy utilizando redes neurais: aplicações em processos petroquímicos,** Tese de mestrado, Instituto Militar de Engenharia, 2001.

PARASKEVOPOULOS P.N, **Digital Control Systems.** Prentice-Hall. 1996.

PEDRYCZ, W. E GOMIDE, F. A. C. **An Introduction to Fuzzy Sets: Analysis and Design (Complex Adaptive Systems)**. MIT Press. 1998.

PEÑA-REYES, C. A. **Coevolutionary Fuzzy Modeling**. Tese de doutorado, Section de'informatique, École Polytechnique Fédérale de Lausanne, Lausanne, Suiça. 2002.

PHAM, D.T. KARABOG, D. **Self-tuning fuzzy controller design using genetic optimization and neural network modeling**. University of Wales College of Cardiff, 1997.

PINHEIRO, C. A. M. **"Analise e Projeto de sistemas de controle fuzzy: Uma abordagem no domínio da frequência"**. Tese de Doutorado. Unicamp. 2000.

RAPOSO, T. A., **Aplicação de Algoritmos Genéticos para o Otimização de Controladores Fuzzy em Processos Petroquímicos**, Dissertação de Mestrado, Instituto Militar de Engenharia, 2000.

RIKER JÚNIOR, F. M. **Desenvolvimento de um Protótipo Didático Multivariável: Sistema de Controle de Níveis de Líquidos**. Projeto de Conclusão de Curso. Universidade Federal do Pará, 2008.

SAGAZ, F. S. G. **Sistema Baseado em Lógica Nebulosa ao Controle dos Níveis de um Processo Multivariável com Quatro Tanques**. Dissertação de Mestrado. IME. 2003.

SEDRA, A. AND SMITH, K. **Microeletrônica,** Pearson Education do Brasil.. 2000.

SENG.T.L, KHALID.M, YUSOF. R, **Tuning of a Neuro-Fuzzy Controller by Genetic Algorithm**.IEEE Transactions on System, Man and Cybernetics.1999.

SHAW, I. S. E SIMÕES, M. G. **Controle e Modelagem Fuzzy**. Edgard Blücher.1999

SWIECH. M. C S, OROSKI. E., ARRUDA, L. V. R. **Sintonia de Controladores PID em Colunas de Destilação Através de Algoritmos Genéticos.** 3º Congresso Brasileiro de Petroleo e Gás. 2004.

TOSATTI. M. A., GALVÃO FILHO, J. C. A., FORBECK F. R., FONSECA A.B., KATAYAMA J. P. M. K., SILVA E. D., CASTANHO M. J. P., VENSKE S. M. G. S., DE RÉ A. M., HERNANDES F. **Algoritmo Híbrido Genético-*Fuzzy* aplicado em Câncer de Próstata**. Revistas Eletronicas, Vol. 32. Nº 62. PUC-RS. 2008.

USER GUIDE ICD2BR In Circuit Debugger. Mosaico. 2011.

WANG, L. X. **Adaptive Fuzzy Systems and Control**. Englewood Cliffs, NJ: Prentice-Hall, 1994.

WANG. LI-XIN**. A course in fuzzy system and control**. Prentice Hall. Inc. 1997

YONEYAMA, T. e NASCIMENTO JÚNIOR, C. L., **Inteligência artificial em controle e automação**, São Paulo, Edgard Blucher, 2000.

ZADEH, L. A., **Fuzzy Sets**, Information and Control, 8, 338-353, 1965.

ZADEH, L. A., **Outline of a new approach to the analysis of complex systems and decision process**, IEEE Transactions Systems, man and cybernetics, 1973.

ANEXOS

ANEXOS

ANEXO 1

PROGRAMA EM LINGUAGEM MATLAB PARA COMUNICAÇÃO SERIAL PARA ENSAIO REAL.

```
clc;
clear;
s = serial('COM1','BaudRate',9600);        % Create serial object (PORT Dependent)
fopen(s)                                   % Open the serial port for r/w
fis = readfis('siconili');
T = 0;
flag = 0;
tvazao=1;
contador=1;
myChar = '0';
pwm = '0';

prompt2 = 'Enter a um nivel2:';
prompt1 = 'Enter a um nivel1: ';
d = zeros(1,5);
q = 0;
nivel = 0;
nivel2 = input(prompt2);
nivel = input(prompt1);

if (nivel > 0)
tstart = tic;
end

while (T < 150)                    % While user hasn't typed 'q'

   if (isempty(fscanf(s)) == 0)
```

```
    [a,x] = fscanf(s);

    d = str2num(a)

    if  ((isempty(d)) | (size(d) ~= 5))
       d = zeros(1,5);
    end

q = 15.38/(d(4)*0.000128);  %ml/s    1 litro / 65 pulsos => logo cada pulso equivale a
15,38 ml
h = d(3)/200;  %calibragem da altura da coluna em relação ao valor do adc
    erro = nivel - h;
    if (contador == 1)
    derro = erro;
    end
    if (contador ~= 1)
    derro = erro - dados((contador-1),6);
    end

    pwm=evalfis([erro derro],fis);

    pwmchar = pwm+97;  %escala de a - z na tabela ascii
    myChar = char(pwmchar);
    fprintf(s, '%s', myChar);    % Write first char of user input to serial port

    T = toc(tstart);
    if (T > 70)
    nivel = nivel2;
    end

    dados(contador,1) = T;
    dados(contador,2) = d(1);
    dados(contador,3) = d(2);
```

```
 dados(contador,4) = h;
 dados(contador,5) = q;
 dados(contador,6) = erro;
 dados(contador,7) = derro;
 dados(contador,8) = pwm;
 contador = contador + 1;
 plot(T,d(1),'or',T,d(2),'og',T,h,'ob',T,q,'oy',T,erro,'ok');
 hold on;
end
end

 pwm = 0;
 fclose(s);
 delete(s)                          % Close the serial port
 fclose(instrfind)
 fclose(instrfind)
```

ANEXO 2

PROGRAMA DE CONSTRUÇÃO PARA AS CURVAS DE RESTRIÇÃO

Arquivo VETORP1.M

```
qi=[31 27 24 16.5 15 0];
pwm=[25 24 20 15 12 0];

pwm_=[0:1:25];
P1=polyfit(pwm,qi,2);

for i=1:length(pwm_)
   Pp(i)=P1(1)*pwm_(i)*pwm_(i)+P1(2)*pwm_(i)+P1(3);
   %Pp12(i)=P12(1)*h_(i)*h_(i)+P12(2)*h_(i)+P12(3);
end
plot(pwm,qi,'r')
hold on
plot(pwm_,Pp,'g')
P1
```

Arquivo VETORP2.M

```
qi=[31 27 24 16.5 15 0];
rhh=[3 2.9154 2.7386 2.2360 1.8708 0];   %raiz quadrada da altura analizada
h_=[0:0.1:5];
P2=polyfit(rhh,qi,2);
for i=1:length(h_)
      Pp2(i)=P2(1)*h_(i)*h_(i)+ P2(2)*h_(i) + P2(3);
end
plot(rhh,qi,'r')
hold on
plot(h_,Pp2,'g')
P2
```

ANEXO 3

PROGRAMA DE ALTERAÇÃO DAS PROPRIEDADES DO CONTROLADOR FUZZY

Arquivo CHANGEFUZZY.M

```
function a = changefuzzy(x1)

a = readfis('siconili');

a = setfield(a,'input',{1,1},'mf',{1,1},'params',[-x1(3) -x1(3) -x1(2)]);
a = setfield(a,'input',{1,1},'mf',{1,2},'params',[-x1(3) -x1(2) x1(1)]);
a = setfield(a,'input',{1,1},'mf',{1,3},'params',[-x1(2) x1(1) x1(2)]);
a = setfield(a,'input',{1,1},'mf',{1,4},'params',[x1(1) x1(2) x1(3)]);
a = setfield(a,'input',{1,1},'mf',{1,5},'params',[x1(2) x1(3) x1(3)]);
a = setfield(a,'input',{1,2},'mf',{1,1},'params',[-x1(5) -x1(5) x1(4)]);
a = setfield(a,'input',{1,2},'mf',{1,2},'params',[-x1(5) x1(4) x1(5)]);
a = setfield(a,'input',{1,2},'mf',{1,3},'params',[x1(4) x1(5) x1(5)]);
a = setfield(a,'output','mf',{1,1},'params',[x1(6) x1(6) x1(7)]);
a = setfield(a,'output','mf',{1,2},'params',[x1(6) x1(7) x1(8)]);
a = setfield(a,'output','mf',{1,3},'params',[x1(7) x1(8) x1(9)]);
a = setfield(a,'output','mf',{1,4},'params',[x1(8) x1(9) x1(9)]);

%regras

a = setfield(a,'input',{1,1},'range',{1,1},-x1(3));
a = setfield(a,'input',{1,1},'range',{1,2},x1(3));

a = setfield(a,'input',{1,2},'range',{1,1},-x1(5));
a = setfield(a,'input',{1,2},'range',{1,2},x1(5));

a = setfield(a,'output','range',{1,1},x1(6));
a = setfield(a,'output','range',{1,2},x1(9));
```

```
%usar setfield para modificar as regras
a = setfield(a,'rule',{1,1},'consequent',round(x1(10)));
a = setfield(a,'rule',{1,2},'consequent',round(x1(11)));
a = setfield(a,'rule',{1,3},'consequent',round(x1(12)));
a = setfield(a,'rule',{1,4},'consequent',round(x1(13)));
a = setfield(a,'rule',{1,5},'consequent',round(x1(14)));
a = setfield(a,'rule',{1,6},'consequent',round(x1(15)));
a = setfield(a,'rule',{1,7},'consequent',round(x1(16)));
a = setfield(a,'rule',{1,8},'consequent',round(x1(17)));
a = setfield(a,'rule',{1,9},'consequent',round(x1(18)));
a = setfield(a,'rule',{1,10},'consequent',round(x1(19)));
a = setfield(a,'rule',{1,11},'consequent',round(x1(20)));
a = setfield(a,'rule',{1,12},'consequent',round(x1(21)));
a = setfield(a,'rule',{1,13},'consequent',round(x1(22)));
a = setfield(a,'rule',{1,14},'consequent',round(x1(23)));
a = setfield(a,'rule',{1,15},'consequent',round(x1(24)));

writefis(a,'siconili');
```

ANEXO 4

PROGRAMA QUE EXECUTA O ALGORITMO GENÉTICO NO CONTROLADOR FUZZY

Arquivo CALCULA.M

```
clc;
clear;

vetorP1;
vetorP2;
siconili_siso;


x1=[0  2 8 0 2 0 12 15 20 1 1 3 2 3 2 3 3 3 4 3 4 4 1 1];

ub=[0  5 9 0 3 0 15 20 31 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4];
lb=[0 .1 5 0 2 0 12 15 20 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1];


a=newfis('siconili');
a=addvar(a,'input','erro',[-x1(3) x1(3)]);
a=addmf(a,'input',1,'negativo','trimf',[-x1(3) -x1(3) -x1(2)]);
a=addmf(a,'input',1,'pnegativo','trimf',[-x1(3) -x1(2) x1(1)]);
a=addmf(a,'input',1,'zero','trimf',[-x1(2) x1(1) x1(2)]);
a=addmf(a,'input',1,'ppositivo','trimf',[x1(1) x1(2) x1(3)]);
a=addmf(a,'input',1,'positivo','trimf',[x1(2) x1(3) x1(3)]);

a=addvar(a,'input','derro',[-x1(5) x1(5)]);
a=addmf(a,'input',2,'negativo','trimf',[-x1(5) -x1(5) x1(4)]);
a=addmf(a,'input',2,'zero','trimf',[-x1(5) x1(4) x1(5)]);
a=addmf(a,'input',2,'positivo','trimf',[x1(4) x1(5) x1(5)]);
%a=addmf(a,'input',2,'positivo','trimf',[x1(2) x1(3) x1(3)]);
```

```
a=addvar(a,'output','saida',[x1(6) x1(9)]);

a=addmf(a,'output',1,'zero','trimf',[x1(6) x1(6) x1(7)]);
a=addmf(a,'output',1,'ppositivo','trimf',[x1(6) x1(7) x1(8)]);
a=addmf(a,'output',1,'positivo','trimf',[x1(7) x1(8) x1(9)]);
a=addmf(a,'output',1,'mpositivo','trimf',[x1(8) x1(9) x1(9)]);

ruleList=[ ...
1 1 round(x1(10)) 1 1
2 1 round(x1(11)) 1 1
3 1 round(x1(12)) 1 1
4 1 round(x1(13)) 1 1
5 1 round(x1(14)) 1 1
1 2 round(x1(15)) 1 1
2 2 round(x1(16)) 1 1
3 2 round(x1(17)) 1 1
4 2 round(x1(18)) 1 1
5 2 round(x1(19)) 1 1
1 3 round(x1(20)) 1 1
2 3 round(x1(21)) 1 1
3 3 round(x1(22)) 1 1
4 3 round(x1(23)) 1 1
5 3 round(x1(24)) 1 1
];

a = addrule(a,ruleList);
writefis(a,'siconili');

options = gaoptimset('StallGenLimit',100,'Generations',100,'PlotFcns',{@gaplotbestf
@gaplotbestindiv},...
'SelectionFcn',@selectiontournament,'CrossoverFcn',@crossoverarithmetic,'Creation
Fcn',@gacreationuniform,'MutationFcn',@mutationadaptfeasible);

[x1 fval] = ga(@(x1)genfuzzy_siconili(x1,a),24,[],[],[],[],lb,ub,[],options);
```

```
a = setfield(a,'input',{1,1},'mf',{1,1},'params',[-x1(3) -x1(3) x1(2)]);
a = setfield(a,'input',{1,1},'mf',{1,2},'params',[-x1(3) -x1(2) x1(1)]);
a = setfield(a,'input',{1,1},'mf',{1,3},'params',[-x1(2) x1(1) x1(2)]);
a = setfield(a,'input',{1,1},'mf',{1,4},'params',[x1(1) x1(2) x1(3)]);
a = setfield(a,'input',{1,1},'mf',{1,5},'params',[x1(2) x1(3) x1(3)]);


a = setfield(a,'input',{1,2},'mf',{1,1},'params',[-x1(5) -x1(5) x1(4)]);
a = setfield(a,'input',{1,2},'mf',{1,2},'params',[-x1(5) x1(4) x1(5)]);
a = setfield(a,'input',{1,2},'mf',{1,3},'params',[x1(4) x1(5) x1(5)]);


a = setfield(a,'output','mf',{1,1},'params',[x1(6) x1(6) x1(7)]);
a = setfield(a,'output','mf',{1,2},'params',[x1(6) x1(7) x1(8)]);
a = setfield(a,'output','mf',{1,3},'params',[x1(7) x1(8) x1(9)]);
a = setfield(a,'output','mf',{1,4},'params',[x1(8) x1(9) x1(9)]);


%regras

a = setfield(a,'input',{1,1},'range',{1,1},-x1(3));
a = setfield(a,'input',{1,1},'range',{1,2},x1(3));

a = setfield(a,'input',{1,2},'range',{1,1},-x1(5));
a = setfield(a,'input',{1,2},'range',{1,2},x1(5));

a = setfield(a,'output','range',{1,1},x1(6));
a = setfield(a,'output','range',{1,2},x1(9));

%usar setfield para modificar as regras

a = setfield(a,'rule',{1,1},'consequent',round(x1(10)));
a = setfield(a,'rule',{1,2},'consequent',round(x1(11)));
```

```
a = setfield(a,'rule',{1,3},'consequent',round(x1(12)));
a = setfield(a,'rule',{1,4},'consequent',round(x1(13)));
a = setfield(a,'rule',{1,5},'consequent',round(x1(14)));
a = setfield(a,'rule',{1,6},'consequent',round(x1(15)));
a = setfield(a,'rule',{1,7},'consequent',round(x1(16)));
a = setfield(a,'rule',{1,8},'consequent',round(x1(17)));
a = setfield(a,'rule',{1,9},'consequent',round(x1(18)));
a = setfield(a,'rule',{1,10},'consequent',round(x1(19)));
a = setfield(a,'rule',{1,11},'consequent',round(x1(20)));
a = setfield(a,'rule',{1,12},'consequent',round(x1(21)));
a = setfield(a,'rule',{1,13},'consequent',round(x1(22)));
a = setfield(a,'rule',{1,14},'consequent',round(x1(23)));
a = setfield(a,'rule',{1,15},'consequent',round(x1(24)));

writefis(a,'siconili');
```

ANEXO 5

PROGRAMA EM C EMBARCADO NO MICROPROCESSADOR.

Arquivo MAIN.C

```
#include "p30f2011.h", #include "p30fxxxx.h", #include "uart.h", #include "adc12.h", #include "ports.h", #include "pwm.h", #include "timer.h", #include "incap.h", #include "dsp.h", #include "stdio.h", #include "stdlib.h", #include "math.h", #include "yvals.h", #include "outcompare.h"

_FWDT(WDT_OFF);                    // Watch Dog Timer // WDT disabilitado
_FOSC(CSW_FSCM_OFF & FRC_PLL8);
// Fail-Safe Clock Monitor Disabilidado // Frequencia do oscilador (8MHz)
// multiplicada por 8 (PLL8) // 7,37mhz *8(pll)/4 = 14.740mhz(clock interno)

// Brown-Out and Power-on Reset Fuse // Brown-Out Desabilitado //Master Clear Habilitado // Power-on Reset Desabilitado
_FBORPOR(MCLR_EN & PBOR_OFF & PWRT_OFF);

// Genneral (Code) Segment Fuse // Proteção desabilitada
_FGS(CODE_PROT_OFF);

unsigned int n1, n2, v1, v1_t, v2;  unsigned int aux13, aux15;
char  sinais[50]; char  sinais2[50];

unsigned int desligaT1, desligaT2, timer_1_edge, timer_3_edge, Int_flag;
int aux1 = 0; int togg = 0; int tempo = 0;
unsigned int y,b,s;
int Interrupt_Count = 0;
```

```
// INTERRUPÇÕES
void __attribute__((interrupt,no_auto_psv)) _ADCInterrupt(void)
{

v1 = ReadADC12(1);
v2 = ReadADC12(0);
_ADIF=0;
}

void __attribute__((__interrupt__)) _OC2Interrupt(void)
{
WriteTimer2(0);
IFS0bits.OC2IF = 0;
}

void __attribute__((__interrupt__)) _T1Interrupt(void)
{
tempo = TMR1;           // Pulse duration is displayed at port D
PORTCbits.RC13 =~ PORTCbits.RC13;
WriteTimer1(0);
IFS0 = IFS0 & 0xFFF7;
IFS0bits.T1IF = 0;
}

void __attribute__((__interrupt__)) _T2Interrupt(void)
{
WriteTimer2(0);
IFS0bits.T2IF = 0;
}
```

```
void __attribute__((__interrupt__)) _T3Interrupt(void)
{
unsigned int compare_reg;
compare_reg = ReadDCOC2PWM();
sprintf(sinais,"%d %d %d %d %d \n\r", s, b, v1, tempo, compare_reg);
putsUART1(sinais);
while(BusyUART1());
tempo = 0;
PORTCbits.RC15 =~ PORTCbits.RC15;
WriteTimer3(0);
_T3IF=0;
}

void inicia_ADC12()
{
unsigned int config_1; unsigned int config_2; unsigned int config_3; unsigned int
configport; unsigned int configscan;

CloseADC12();
ADCON1bits.ADON = 0;

SetChanADC12(
ADC_CH0_POS_SAMPLEA_AN0 &
ADC_CH0_NEG_SAMPLEA_NVREF &
ADC_CH0_POS_SAMPLEB_AN1 &
ADC_CH0_NEG_SAMPLEB_NVREF
);

ConfigIntADC12( ADC_INT_ENABLE &          // Habilita a interrupção do ADC12.
ADC_INT_PRI_3);                                  // Define prioridade 1.

config_1 = ADC_MODULE_ON &          // Habilita o ADC12
ADC_IDLE_CONTINUE &                 // Defifine a operação em modo Idle.
ADC_FORMAT_INTG &
```

```
ADC_CLK_AUTO &
ADC_AUTO_SAMPLING_ON;
ADC_SAMP_ON;

config_2 = ADC_VREF_AVDD_AVSS  &
ADC_SCAN_ON &
ADC_SAMPLES_PER_INT_4 &
ADC_ALT_BUF_ON &
ADC_ALT_INPUT_ON;

config_3 = ADC_SAMPLE_TIME_15 &
// Define o tempo de amostragem como 4*Tad.
ADC_CONV_CLK_SYSTEM &
// Seleção da fonte de clock de converção (Tcy)
//ADC_CONV_CLK_9Tcy;
// Define o Tad.   TAD = TCY * (0.5*(ADCS<5:0> + 1))
ADC_CONV_CLK_5Tcy;
configport = ENABLE_AN0_ANA & ENABLE_AN1_ANA;

configscan = SKIP_SCAN_AN2 & SKIP_SCAN_AN3 & SKIP_SCAN_AN4 &
SKIP_SCAN_AN5 & SKIP_SCAN_AN6 & SKIP_SCAN_AN7;
OpenADC12(config_1,config_2,config_3,configport,configscan);  // Inicia ADC12
ADCON1bits.ADON = 1;
}

void config_T1(unsigned int periodo)
{
CloseTimer1();
ConfigIntTimer1(T1_INT_ON & T1_INT_PRIOR_1);
OpenTimer1(T1_ON & T1_GATE_ON & T1_IDLE_STOP & T1_SYNC_EXT_OFF &
T1_SOURCE_INT & T1_PS_1_256, periodo);
EnableIntT1;
WriteTimer1(0);
}
```

```
void config_T2(unsigned int periodo)
{
periodo = 366;   // máximo valor dos timers

CloseTimer2();
IFS0bits.T2IF = 0;
ConfigIntTimer2(T2_INT_ON & T2_INT_PRIOR_2);

OpenTimer2(T2_ON & T2_GATE_OFF & T2_IDLE_STOP &  T2_SOURCE_INT &
T2_PS_1_1, periodo);
WriteTimer2(0);
EnableIntT2;
}

void config_T3()
{
unsigned int periodo = 7197;
CloseTimer3();
IFS0bits.T3IF = 0;
ConfigIntTimer3(T3_INT_ON & T3_INT_PRIOR_2);
OpenTimer3(T3_ON & T3_GATE_OFF & T3_IDLE_STOP & T3_SOURCE_INT &
T3_PS_1_256, periodo);
EnableIntT3;
}

void inicia_UART()
{
  unsigned int ubrg;    unsigned int config1;    unsigned int config2;

CloseUART1();
ubrg = 95;
//taxa de transmissão de 9600bps
//ubrg = (14740000/(16*(9600))-1
 //ubrg = 94.96
```

```
 ConfigIntUART1(UART_RX_INT_DIS & UART_RX_INT_PR2 & UART_TX_INT_DIS
&  UART_TX_INT_PR2);
config1    =   UART_EN &  UART_IDLE_CON & UART_RX_TX &
UART_DIS_WAKE & UART_DIS_LOOPBACK & UART_EN_ABAUD &
UART_NO_PAR_8BIT &  UART_1STOPBIT;

 config2 =
UART_INT_TX_BUF_EMPTY & UART_TX_PIN_NORMAL & UART_TX_ENABLE &
UART_INT_RX_BUF_FUL & UART_ADR_DETECT_DIS &
UART_RX_OVERRUN_CLEAR;

 OpenUART1(config1, config2, ubrg);
EnableIntU1RX; EnableIntU1TX;
}

void _ISR _U1TXInterrupt(void)
{
 IFS0bits.U1TXIF = 0;
}

void _ISR _U1RXInterrupt(void)
{
IFS0bits.U1RXIF = 0;
}

void inicia_PWM()
{
unsigned int pulse_start ;
unsigned int pulse_stop;
CloseOC2();
ConfigIntOC2(OC_INT_ON & OC_INT_PRIOR_1);
pulse_start = 0;
pulse_stop = 366;
```

```
PR2 = 366;
WriteTimer2(0);
OpenOC2(OC_IDLE_STOP & OC_TIMER2_SRC &
 OC_PWM_FAULT_PIN_DISABLE, pulse_stop, pulse_start);
}

int main(void)
{
   TRISCbits.TRISC13 = 0;
        TRISCbits.TRISC14 = 1;  //RC14 é entrada
        TRISCbits.TRISC15 = 0;  //RC15 é saida
        TRISBbits.TRISB7 = 0;
        TRISBbits.TRISB0 = 1;
        TRISBbits.TRISB1 = 1;
        v1_t = 0;

        PORTCbits.RC13 = 0;
        y = 0;

        inicia_PWM();
        config_T1(3907);
        config_T2(366);
        config_T3(3907);
        inicia_ADC12();
 inicia_UART();

while(1)
        {
                if(v1<=0)
                {
                v1=0;
                }
                if(v1>4095)
                {
```

```
            v1=4095;
        }
            if(tempo<=0)
                {
            tempo=0;
        }

if(DataRdyUART1())
    {
        s = ReadUART1();
        if(s>=122)              // s é sempre 122 - valor maximo 366;
            {   b = 366;    }

        if(s<122)
            {
            b = 14.64*(s-97);
              // s é um numero de a - z (97 a 122) na escala da tabela ascii
            }
         if(b<=0)
            {       b=0;            }
         if(b>366)
            {       b=366;      }


        SetDCOC2PWM(b); //corrige o sinal do PWM
    } //end if
}       // end while

}  //end main
```

ANEXO 6

CIRCUITO COMPLETO DA PLANTA DE NÍVEL DE LÍQUIDOS