

UNIVERSIDADE ESTADUAL DE MARINGÁ
CENTRO DE TECNOLOGIA
DEPARTAMENTO DE ENGENHARIA QUÍMICA
PROGRAMA DE DOUTORADO EM ENGENHARIA QUÍMICA

**APLICAÇÃO DE ESTRATÉGIAS DE CONTROLE EM COLUNA DE
DESTILAÇÃO**

Claiton Moro Franchi

Eng^o Eletricista, UFSM, 2002.

Msc. Eng. Química, 2007.

Orientador: Mauro A. S. S. Ravagnani

Tese de Doutorado submetida à
Universidade Estadual de Maringá,
como parte dos requisitos necessários à
obtenção do Grau de Doutor em
Engenharia Química na área de
Desenvolvimento de Processos.

Maringá – PR – Brasil

Setembro de 2010

Dados Internacionais de Catalogação-na-Publicação (CIP)
(Biblioteca Central - UEM, Maringá – PR., Brasil)

F816a Franchi, Claiton Moro
Aplicação de estratégias de controle em coluna de destilação / Claiton Moro Franchi. -- Maringá, 2010.
181 f. : il. color., figs., tabs., retrs.

Orientador : Prof. Dr. Mauro A. S. S. Ravagnani.
Tese (doutorado) - Universidade Estadual de Maringá,
Programa de Pós-Graduação em Engenharia Química, área de
concentração: Desenvolvimento de Processos, 2010.

1. Etanol - Química - Controle de processos. 2.
Etanol - Coluna de destilação. 3. Etanol - Coluna de
destilação - Instrumentação. 4. Etanol - Coluna de
destilação - Identificação. I. Ravagnani, Mauro A. S.
S., orient. II. Universidade Estadual de Maringá.
Programa de Pós-Graduação em Engenharia Química. Área de
concentração: Desenvolvimento de Processos. III. Título.

CDD 21.ed.662.6692

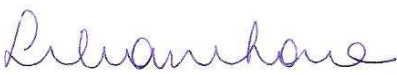
UNIVERSIDADE ESTADUAL DE MARINGÁ
CENTRO DE TECNOLOGIA
DEPARTAMENTO DE ENGENHARIA QUÍMICA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA QUÍMICA

Esta é a versão final da Tese de Doutorado apresentada por Claiton Moro Franchi perante a Comissão Julgadora do Curso de Doutorado em Engenharia Química em 24 de setembro de 2010.

COMISSÃO JULGADORA


Prof. Dr. Mauro Antonio da Silva Sá Ravagnani
Orientador


Prof. Dr. Fernando Palú
Membro


Profª Drª Liliane Maria Ferrareso Lona
Membro


Prof. Dr. Cid Marcos Gonçalves Andrade
Membro


Prof. Dr. Flávio Faria de Moraes
Membro


Prof. Dr. Oswaldo Curty da Motta Lima
Membro

DEDICATÓRIA

Aos meus pais, Calixtro e Lurdes, por todo apoio, educação e confiança em mim depositados;

Com carinho especial, à minha esposa Eliane pela paciência, amor e apoio que recebi durante a elaboração deste trabalho.

AGRADECIMENTOS

Ao Professor Mauro pela confiança em mim depositada, paciência, apoio e auxílio, indispensáveis para a elaboração do trabalho.

Aos funcionários do DEQ pelo auxílio na realização dos experimentos.

Aos colegas do DEQ que sempre me incentivaram durante a elaboração do trabalho.

Aos demais amigos e colegas não citados que contribuíram de forma direta ou indireta com a realização deste trabalho.

APLICAÇÃO DE ESTRATÉGIAS DE CONTROLE EM COLUNA DE DESTILAÇÃO

AUTOR: CLAITON MORO FRANCHI

ORIENTADOR: PROF. DR. MAURO ANTONIO DA SILVA SÁ RAVAGNANI

Tese de Doutorado; Programa de Pós-Graduação em Engenharia Química; Universidade Estadual de Maringá; Av. Colombo, 5790, BL E46 - 09; CEP: 87020-900 Maringá - PR, Brasil, defendida em setembro de 2010. (181 páginas)

RESUMO

A destilação é o processo de separação mais utilizado na indústria química. Devido ao fato de ser um processo que consome uma grande quantidade de energia, é fundamental conhecer adequadamente o seu funcionamento, para que se possa aplicar técnicas eficientes de controle, obtendo-se assim, o rendimento máximo do processo, bem como um produto final de elevada pureza, com menor consumo energético.

O presente trabalho tem como objetivo o desenvolvimento e a implementação de uma placa de aquisição de dados e controle de processos, juntamente com um sistema de supervisão de dados. O sistema desenvolvido foi implementado em uma coluna de destilação utilizada como módulo didático para disciplinas do curso de graduação em Engenharia Química da Universidade Estadual de Maringá.

Para a aplicação das estratégias de controle na coluna de destilação foi primeiramente obtido um modelo aproximado do processo por meio de técnicas de identificação de processos utilizando redes neurais artificiais e modelos clássicos de identificação ARX, ARMAX, OE, BJ e Espaço de Estados. Para implementar as técnicas de identificação foram desenvolvidos algoritmos com o auxílio do software MATLABTM, tendo como resultado um modelo obtido por meio de identificação usando redes neurais artificiais com um erro médio quadrático de 0,66% em relação ao conjunto de dados de entrada-saída.

Com o modelo obtido foram feitas simulações no *software* MATLABTM, aplicando estratégias de controle convencional (PID), *fuzzy* e *neuro-fuzzy*. O *software* de simulação foi empregado para sintonia dos respectivos controladores, bem como na avaliação do desempenho dos mesmos quando sujeitos a perturbações. Nesta etapa de projeto, o controlador *neuro-fuzzy* apresentou os melhores resultados, com tempos de resposta, *overshoot* e tempo de acomodação menores, bem como menores índices de erro (*ITAE*, *IAE* e *ISE*), mediante aplicações de perturbações na carga e no *setpoint*.

Após a fase de simulação, as estratégias de controle foram implementadas na coluna de destilação com o auxílio da placa da aquisição de dados e de controle e do sistema de supervisão, integrados ao *software* MATLABTM por meio de uma interface serial. A aplicação dos controladores, implementados e configurados no *software* MATLABTM na coluna de destilação serviram para confirmar o melhor desempenho do controlador *neuro-fuzzy*, perante os controladores PID e *fuzzy*. Os resultados experimentais comprovaram a eficácia do modelo do processo e da sintonia dos controladores obtidas via simulação, melhorando a operação da coluna de destilação com um consumo energético adequado para a produção de um produto final dentro das especificações requeridas.

CONTROL STRATEGIES APPLICATION IN DISTILLATION COLUMN

AUTHOR: CLAITON MORO FRANCHI

SUPERVISOR: PROF. DR. MAURO ANTONIO DA SILVA SÁ RAVAGNANI

Doctor Thesis, Chemical Engineering Graduate Program; State University of Maringá; Av. Colombo, 5790, BL E46 - 09; CEP: 87020-900 Maringá - PR, Brazil, presented in september 2010 (181 pages).

ABSTRACT

Distillation is the most widely used separation process in industry. It is a process that consumes a large amount of energy and it is essential to understand its operation properly to apply efficient control techniques, thus obtaining the maximum process performance, as well as an end high purity product, with lower energy consumption.

This work aims to develop and implement a data acquisition and control card, along with a data monitoring system. The system has been implemented in a distillation column used as a module for Chemical Engineering undergraduate course teaching at State University of Maringá.

To apply the control strategies in the distillation column, was first obtained an approximate process model using identification process techniques with artificial neural networks and classical identification methods ARX, ARMAX, OE, BJ and State Space. To implement the identification techniques, algorithms were developed with MATLABTM software, resulting in a model obtained by identification using artificial neural networks with a mean square error of 0,66% on the input-output set.

Simulations were made with obtained model in MATLABTM software, applying conventional control strategies (PID), fuzzy and neuro-fuzzy. The simulation software was used to tune controllers, as well to evaluate the performance when applied disturbances. Neuro-fuzzy controller presents the best results, with smaller response times, overshoot and settling time, as well as lower performance criteria (*ITAE*, *IAE*, and *ISE*) by the disturbance application in the load and setpoint.

The control strategies were implemented in the distillation column with control and data acquisition card and supervisory system, integrated with MATLABTM software by one serial interface. The controllers application in the distillation column, tuned and implemented in MATLABTM software play the role to confirm the better neuro-fuzzy controller performance comparing with PID and fuzzy controllers. The experimental results confirm the process model effectiveness and the controllers tuned via simulation, improving the distillation column operation with an appropriate energy consumption to produce a final product within the required specifications.

ÍNDICE DE FIGURAS

Figura 2.1 - Operação de pratos do tipo perfurado com baixos fluxos de líquido e de vapor (NISENFELD e SEEMAN, 1981).	8
Figura 2.2 - Equilíbrio líquido-vapor para mistura etanol-água a 1 atm (JACQUES, 1995).	9
Figura 2.3 - Diagramas T-x para mistura binária ideal, com azeótropo de mínimo e azeótropo de máximo (PERRY e GREEN, 1997).	11
Figura 2.4 – Variáveis controladas e manipuladas em uma coluna de destilação típica.	13
Figura 2.5 - Configuração típica de uma coluna de destilação utilizando sensores de concentração.	14
Figura 2.6 - Configuração típica de uma coluna de destilação utilizando sensores de temperatura para inferência da concentração.	15
Figura 2.7 - Controle da coluna de destilação por meio da variação da taxa de aquecimento do refeedor.	16
Figura 2.8 - Fluxo de calor através de uma parede.	17
Figura 2.9 - Superfície da parede da coluna de destilação.	19
Figura 3.1 - Placa de aquisição de dados desenvolvida.	22
Figura 3.2 - Esquema elétrico da placa de aquisição de dados proposta.	23
Figura 3.3 - Instrumentação aplicada na coluna de destilação.	23
Figura 3.4 - Microcontrolador PIC 16F877A e descrição de pinagem.	25
Figura 3.5 - Estrutura interna do microcontrolador utilizado.	26
Figura 3.6 - Termômetro digital DS 18B20.	28
Figura 3.7 - Sequência de operação dos sensores DS18B20.	30
Figura 3.8 - Descrição da pinagem do DS18B20.	30
Figura 3.9 - Diagrama de blocos do DS18B20.	31
Figura 3.10 - Formato do registrador de temperatura.	32
Figura 3.11 - 64 bits de código ROM.	33
Figura 3.12 - Mapa de memória do sensor de temperatura DS18B20.	34
Figura 3.13 - Banho térmico utilizado para aferição dos sensores.	34
Figura 3.14 - Gráfico representativo da temperatura real medida e reta de calibração.	36
Figura 3.15 - Configuração do <i>hardware</i>	37
Figura 3.16 - Sequência de inicialização.	41
Figura 3.17 - Inicialização do barramento 1-Wire.	41

Figura 3.18 - Leitura de dados no barramento 1-Wire.	43
Figura 3.19 - Fluxo de leitura no barramento.	43
Figura 3.20 - Fluxo de escrita no barramento.	44
Figura 3.21 - Diagrama de blocos placa de controle de potência.	45
Figura 3.22 - Circuito placa de controle de potência desenvolvida.	45
Figura 3.23 - Circuito de <i>zero crossing</i>	46
Figura 3.24 - Fluxograma do circuito <i>zero crossing</i>	46
Figura 3.25 - Circuito de controle de tensão.	47
Figura 3.26 - Fluxograma do circuito de controle de tensão.	47
Figura 3.27 - Fluxograma do protocolo de comunicação.	48
Figura 3.28 - Tela inicial do sistema de supervisão.	49
Figura 3.29 - Tela principal do sistema de supervisão.	50
Figura 3.30 - Cadastro de usuários.	50
Figura 3.31 - Gráfico representando a tendência de temperatura nos pratos.	51
Figura 3.32 – Opção de saída de relatórios.	52
Figura 3.33 - Relatório do perfil de temperaturas.	52
Figura 3.34 - Controle manual.	53
Figura 3.35 - Sistema típico utilizado para aquisição de temperaturas.	55
Figura 4.1 - Coluna de destilação binária.	58
Figura 4.2 - Procedimentos necessários para obtenção de modelos de identificação.	66
Figura 4.3 - Representação de sinais do modelo ARX.	68
Figura 4.4 - Representação de sinais do modelo ARMAX.	68
Figura 4.5 - Representação de sinais do modelo OE.	69
Figura 4.6 - Representação de sinais do modelo BJ.	69
Figura 4.7 - Modelo básico de um neurônio artificial.	71
Figura 4.8 - Modelo básico de um neurônio artificial com múltiplas entradas.	72
Figura 4.9 - Modelo básico de um neurônio artificial com múltiplas entradas e múltiplas camadas.	72
Figura 4.10 - Etapa de treinamento propagação do algoritmo <i>backpropagation</i>	76
Figura 4.11- Etapa de treinamento retropropagação do algoritmo <i>backpropagation</i>	76
Figura 4.12- Fluxograma de treinamento do algoritmo <i>backpropagation</i>	77
Figura 4.13 - Conexão da coluna de destilação com sistema de aquisição de dados.	79
Figura 4.14 - Etapas da obtenção do modelo do processo por meio de identificação.	80
Figura 4.15 - Dados de temperatura do topo coluna e refeedor.	80

Figura 4.16 - Comparação entre dados de entrada e saída e modelo ARX.	81
Figura 4.17 - Comparação entre dados de entrada e saída e modelo Espaço de Estados....	82
Figura 4.18 - Comparação entre dados de entrada e saída e modelo OE	83
Figura 4.19 - Comparação entre dados de entrada e saída e modelo ARMAX.....	84
Figura 4.20 - Comparação entre dados de entrada e saída e modelo BJ.	85
Figura 4.21 - Representação do esquema de identificação utilizando redes neurais.....	87
Figura 4.22 - Comparação entre dados de entrada e saída e modelo com redes neurais artificiais (treinamento).	87
Figura 4.23 - Dados aplicados no modelo para teste e validação.....	88
Figura 5.1 - Passos necessários para desenvolvimento de um sistema de controle.....	90
Figura 5.2 - Resposta da malha de controle com ganho e períodos críticos.	95
Figura 5.3 - Arquitetura representativa da lógica <i>fuzzy</i>	98
Figura 5.4 - Arquitetura de um controlador <i>fuzzy</i>	99
Figura 5.5 - Entradas e saídas do controlador <i>fuzzy</i>	100
Figura 5.6 - Funções de pertinência típicas.	102
Figura 5.7 - Estrutura dos controladores (a) <i>fuzzy</i> -PD, (b) <i>fuzzy</i> -PI e (c) <i>fuzzy</i> -PID.	104
Figura 5.8 - Procedimento de sintonia de controlador <i>fuzzy</i> -PID.....	105
Figura 5.9 - Diagrama representativo de um sistema neuro- <i>fuzzy</i>	106
Figura 5.10 - Estrutura de um sistema neuro- <i>fuzzy</i>	108
Figura 5.11 - Representação de uma rede adaptativa.	109
Figura 5.12 – Representação da estrutura do ANFIS.	111
Figura 5.13 – Exemplo de regras para o modelo <i>fuzzy</i> Takagi-Sugeno de primeira ordem.	112
Figura 5.14 – Índices para avaliação de desempenho de malhas de controle.	116
Figura 5.15 – Simulação das estratégias de controle mediante modelo obtido via identificação.....	118
Figura 5.16 - Funções de pertinência para as variáveis (a) erro (b) variação do erro (c) saída.....	120
Figura 5.17 - Diagrama de superfície da resposta do controlador <i>fuzzy</i>	121
Figura 5.18 - Descrição das entradas e saídas do controlador neuro- <i>fuzzy</i>	122
Figura 5.19 - Etapas para parametrização do controlador neuro- <i>fuzzy</i>	122
Figura 5.20 - Modelo ANFIS elaborado.....	123
Figura 5.21 - Funções de pertinência das entradas e respectivas funções de saída.....	124
Figura 5.22 - Arquitetura de rede ANFIS com as regras geradas.....	124

Figura 5.23 - Erro médio quadrático por iteração do modelo obtido.	125
Figura 5.24 – Dados de entrada/saída e modelo obtido com a simulação.....	125
Figura 5.25 - Comparação de controladores para perturbação degrau unitário no <i>setpoint</i> (a) <i>fuzzy</i> -PD/PD (b) <i>fuzzy</i> -PI/PI.....	126
Figura 5.26 - Comparação de controladores PID/ <i>fuzzy</i> -PID para perturbação degrau unitário no <i>setpoint</i>	127
Figura 5.27 - Comparação de controladores PID/ <i>neuro-fuzzy</i> para perturbação degrau unitário no <i>setpoint</i>	127
Figura 5.28 - Comparativo de instante de pico para os controladores.	128
Figura 5.29 - Comparativo de <i>overshoot</i> máximo para os controladores.....	128
Figura 5.30 - Comparativo de tempo de acomodação para os controladores.....	128
Figura 5.31 - Comparativo de tempo de atraso para os controladores.	128
Figura 5.32 - Comparativo de tempo de subida para os controladores.	129
Figura 5.33 - Comparação de controladores PID/ <i>fuzzy</i> -PID para perturbação degrau na carga (a) +1 (b) +2 (c) -1 (d) -2.	130
Figura 5.34 - Comparação de controladores PID/ <i>neuro-fuzzy</i> para perturbação degrau na carga (a) +1 (b) +2 (c) -1 (d) -2	131
Figura 5.35 - Comparação do índice <i>ITAE</i> para os controladores PID, <i>fuzzy</i> -PID e <i>neuro-</i> <i>fuzzy</i> para perturbação degrau na carga (a) +1 (b) +2 (c) -1 (d) -2.....	131
Figura 5.36 - Comparação do índice <i>ISE</i> para os controladores PID, <i>fuzzy</i> -PID e <i>neuro-</i> <i>fuzzy</i> para perturbação degrau na carga (a) +1 (b) +2 (c) -1 (d) -2.....	132
Figura 5.37 - Comparação do índice <i>IAE</i> para os controladores PID, <i>fuzzy</i> -PID e <i>neuro-</i> <i>fuzzy</i> para perturbação degrau na carga (a) +1 (b) +2 (c) -1 (d) -2.....	132
Figura 6.1 - Arquitetura proposta para monitoração e controle da coluna de destilação estudada.	134
Figura 6.2 - Coluna de destilação integrada ao sistema de controle, aquisição de dados e supervisão.	136
Figura 6.3 - Pontos de operação e temperaturas da coluna de destilação.....	138
Figura 6.4 - Perfil de temperaturas médias na coluna de destilação.....	139
Figura 6.5 - Malha de controle proposta para coluna de destilação.	140
Figura 6.6 - Resposta do controlador PID para perturbação degrau na carga (a) +20 mL/min (b) +40 mL/min (c) -20 mL/min (d) -40 mL/min.	142
Figura 6.7 - Resposta do controlador <i>fuzzy</i> -PID para perturbação degrau na carga (a) +20 mL/min (b) +40 mL/min (c) -20 mL/min (d) -40 mL/min.	144

Figura 6.8 - Resposta do controlador neuro- <i>fuzzy</i> para perturbação degrau na carga (a) +20 mL/min (b) +40 mL/min (c) -20 mL/min (d) -40 mL/min.	145
Figura 6.9 - Comparação do índice <i>ITAE</i> para os controladores PID, <i>fuzzy</i> -PID e neuro- <i>fuzzy</i> para perturbação degrau na carga (a) +20 mL/min (b) +40 mL/min (c) -20 mL/min (d) -40 mL/min.	146
Figura 6.10 - Comparação do índice <i>ISE</i> para os controladores PID, <i>fuzzy</i> -PID e neuro- <i>fuzzy</i> para perturbação degrau na carga (a) +20 mL/min (b) +40 mL/min (c) -20 mL/min (d) -40 mL/min.	146
Figura 6.11 - Comparação do índice <i>IAE</i> para os controladores PID, <i>fuzzy</i> -PID e neuro- <i>fuzzy</i> para perturbação degrau na carga (a) +20 mL/min (b) +40 mL/min (c) -20 mL/min (d) -40 mL/min.	146
Figura 6.12 - Resposta do controlador PID para perturbação degrau no <i>setpoint</i> (a) + 2 ⁰ C (b) -2 ⁰ C.	148
Figura 6.13 - Resposta do controlador <i>fuzzy</i> -PID para perturbação degrau no <i>setpoint</i> (a) + 2 ⁰ C (b) -2 ⁰ C.	149
Figura 6.14 - Resposta do controlador neuro- <i>fuzzy</i> para perturbação degrau no <i>setpoint</i> (a) + 2 ⁰ C (b) -2 ⁰ C.	150
Figura 6.15 - Comparação do índice <i>ITAE</i> para os controladores PID, <i>fuzzy</i> -PID e neuro- <i>fuzzy</i> para perturbação degrau no <i>setpoint</i> (a) + 2 ⁰ C (b) -2 ⁰ C.	150
Figura 6.16 - Comparação do índice <i>ISE</i> para os controladores PID, <i>fuzzy</i> -PID e neuro- <i>fuzzy</i> para perturbação degrau no <i>setpoint</i> (a) + 2 ⁰ C (b) -2 ⁰ C.	151
Figura 6.17 - Comparação do índice <i>IAE</i> para os controladores PID, <i>fuzzy</i> -PID e neuro- <i>fuzzy</i> para perturbação degrau no <i>setpoint</i> (a) + 2 ⁰ C (b) -2 ⁰ C.	151

ÍNDICE DE TABELAS

Tabela 2.1 - Dados à pressão constante do equilíbrio vapor-líquido para a mistura etanol-água.....	10
Tabela 2.2 - Perfil corrigido de temperaturas nos pratos da coluna de destilação.	20
Tabela 3.1 - Quadro comparativo entre sensores típicos de temperatura e o DS18B20.	30
Tabela 3.2 - Valores obtidos na calibração do sensor 1.	35
Tabela 3.3 - Retas de calibração para os sensores de temperatura.	36
Tabela 3.4 - Protocolo de recepção de dados.	48
Tabela 3.5 - Descrição dos elementos utilizados para a confecção do sistema de aquisição, controle e supervisão de dados (Fonte: Eletrônica Projesol Ltda.).	54
Tabela 3.6 - <i>Hardware</i> e <i>software</i> necessários para o sistema de controle e aquisição de dados (Fonte: Eletronor distribuidora de materiais elétricos Ltda).	55
Tabela 4.1 - Forma geral de representação de modelos autorregressivos.	70
Tabela 4.2 - Erro médio quadrático obtido para os modelos de identificação.	85
Tabela 5.1 - Equações de Ziegler e Nichols segundo método.....	96
Tabela 5.2 - Constantes dos controladores obtidas utilizando o método de sintonia de Ziegler-Nichols.	96
Tabela 5.3 - Caracterização matemática da função de pertinência triangular.	102
Tabela 5.4 – Etapas relativas ao algoritmo de aprendizagem híbrido da ANFIS.....	115
Tabela 5.5 - Base de regras para controlador <i>fuzzy</i>	120
Tabela 5.6 - Valores iniciais das constantes dos controladores <i>fuzzy</i>	121
Tabela 5.7 - Valores finais das constantes dos controladores <i>fuzzy</i>	121
Tabela 5.8 - Índices de desempenho para controladores PI/ <i>fuzzy</i> -PI e PD/ <i>fuzzy</i> -PD para perturbação degrau unitário no <i>setpoint</i>	126
Tabela 5.9 - Índices de desempenho para controladores PID/ <i>fuzzy</i> -PID para perturbação degrau unitário no <i>setpoint</i>	129
Tabela 5.10 - Índices de desempenho para controladores PID/ <i>fuzzy</i> -PID/neuro- <i>fuzzy</i> para perturbação na carga.	132
Tabela 6.1 - Médias aritméticas das temperaturas e valores de temperatura aferidos.....	138
Tabela 6.2 - Condições de operação da coluna de destilação.....	141
Tabela 6.3 - Índices de desempenho dos controladores PID, <i>fuzzy</i> -PID e neuro- <i>fuzzy</i> mediante perturbação na vazão de alimentação.	147

Tabela 6.4 - Índices de desempenho dos controladores PID, <i>fuzzy</i> -PID e neuro- <i>fuzzy</i> mediante perturbação no <i>setpoint</i>	151
--	-----

LISTA DE SÍMBOLOS

A	Ampère
$\mathbf{A}$	Matriz do sistema para Espaço de Estados
$A(q)$	Polinômio da variável de saída dos modelos de função de transferência
$\tilde{A}_i^j$	J -ésimo valor linguístico da variável $\tilde{u}_i$
$\mathbf{B}$	Matriz do sistema para Espaço de Estados
B	Vazão mássica do produto de fundo (ft^3/s)
Bias	<i>Offset</i>
Bx	Concentração do produto de fundo (GL)
$B(q)$	Polinômio da variável de entrada dos modelos de função de transferência
b	<i>Bias</i> (redes neurais)
$b(k)$	Variável de média móvel
b_j^h	Termo de <i>bias</i> da j -ésima camada oculta para modelo NARMAX
$\mathbf{C}$	Matriz do sistema para espaço de estados
$C(q)$	Polinômio da variável de média móvel nos modelos ARMAX
$\mathbf{D}$	Matriz do sistema para espaço de estados
D	Vazão mássica do destilado (ft^3/s)
DQ	Entrada de dados/Saída de dados (pino de interface 1-Wire)
$D(q)$	Polinômio aplicado na função de erro do modelo BJ
E	Função erro do algoritmo <i>backpropagation</i>
$e(n)$	Erro do sistema para espaço de estados
$e(t)$	Termo de ruído no modelo de função de transferência no tempo t
F	Vazão mássica de alimentação (mol/min)
F_L	Taxa de escoamento do líquido pelo vertedouro (ft^3/s)
Fm	Vazão de entrada (mol/min)
$F(q)$	Polinômio da variável de saída dos modelos OE e BJ
f	Função de transferência (redes neurais)
f_j^h	Função de transferência para o modelo NARMAX
f_l^q	Função de ativação para nós de saída para modelo NARMAX
GL	Grau Gay Lussac

GND	Pino de conexão a terra
h	Coeficiente de transferência de calor por convecção ($\text{W}/\text{m}^2 \cdot ^\circ\text{C}$)
h_{ow}	Nível do líquido no vertedouro (ft)
j_n	N-ésima camada oculta para modelo NARMAX
K	Matriz do sistema para Espaço de Estados
K_c	Ganho proporcional
K_{cu}	Ganho crítico de um controlador proporcional
k	Coeficiente de transferência de calor por condução ($\text{W}/\text{m} \cdot ^\circ\text{C}$)
L	Número de camadas de uma rede adaptativa
L	Vazão mássica de líquido (ft^3/s)
L_{NF}	Vazão mássica de líquido no prato de alimentação (ft^3/s)
L_{NT}	Vazão mássica de líquido do prato de topo (ft^3/s)
L_w	Líquido no vertedouro (ft)
l	l -ésima unidade de saída do modelo NARMAX
M	Vapor de topo (moles)
M_B	Número de moles do produto de fundo
M_D	Número de moles do destilado
M_n	Número de moles do n-ésimo prato
M_{nF}	Número de moles do prato de alimentação
$m(t)$	Ação de controle
N_T	Número de pratos
n_a	Ordem do polinômio $A(q)$
n_b	Ordem do polinômio $B(q)$
n_c	Ordem do polinômio $C(q)$
n_d	Ordem do polinômio $D(q)$
n_f	Ordem do polinômio $F(q)$
n_k	Número de atrasos da saída para a entrada nos modelos de função de transferência
n_u	Ordem do polinômio da variável de média móvel nos modelos NARMAX
n_y	Ordem do polinômio da variável autorregressiva nos modelos NARMAX
O_i^k	Função de saída de um nó de uma rede adaptativa
$O_{m,p}^L$	N-ésima componente do atual do vetor de saída produzida pelo vetor p de entrada em uma rede adaptativa

p	Escalar de entrada (redes neurais)
q	Número de unidades de saída do modelo NARMAX
R	Vazão mássica de refluxo (ft^3/s)
$\mathbf{R}$	Vetor de entrada (redes neurais)
T_{amb}	Temperatura ambiente ($^{\circ}\text{C}$)
T_{in}	Temperatura interna na coluna ($^{\circ}\text{C}$)
T_{sen}	Temperatura medida pelo sensor ($^{\circ}\text{C}$)
T_u	Período crítico de oscilação
$T_{m,p}$	N-ésima componente de saída do vetor p^n de entrada de uma rede adaptativa
U_i	Universo de discurso para variável de entrada
$u(k)$	Variável exógena (entrada do sistema)
$u(n)$	Entrada do sistema para espaço de estados
$u(t)$	Dados de entrada do modelo
$u^T(k)$	Vetor de entrada para modelo NARMAX
$\tilde{u}_i$	Variável linguística de entrada
V	Vazão mássica (ft^3/s)
V_{dd}	Alimentação 5Vcc
V_{y_B}	Vazão mássica de vapor do produto de fundo (ft^3/s)
$V_{y_{NF}}$	Vazão mássica do prato de alimentação (ft^3/s)
$V_{y_{NT}}$	Vazão mássica de vapor no n-ésimo prato (ft^3/s)
$v(t)$	Constantes de ajuste de erro de modelo
$\hat{v}(k)$	Nós de saída modelo NARMAX
$\mathbf{W}$	Matriz de pesos sinápticos (redes neurais)
W	Potência ativa em Watts
W_{ji}^h	Peso das conexões entre a i -ésima entrada e a j -ésima camada oculta para o modelo NARMAX
W_{ij}^q	Peso das conexões entre a i -ésima entrada e a q -ésima camada de saída para modelo NARMAX
$W_{lj}^q(k)$	Peso da conexão entre a j -ésima camada oculta e l -ésima unidade de saída para o modelo NARMAX
w	Peso sináptico (redes neurais)

$w_{j,i}$	Pesos sinápticos algoritmo <i>backpropagation</i>
$\hat{\mathbf{w}}^T(k)$	Vetor de regressão para modelo NARMAX
$\mathbf{x}(n)$	Vetor de estados
$\mathbf{x}(t)$	Dados de entrada de um sistema dinâmico
x_B	Concentração do produto de fundo (GL)
x_D	Concentração do destilado (GL)
x_n	Concentração do líquido no n -ésimo prato (GL)
x_{NF}	Concentração no prato de alimentação (GL)
x_{NT}	Concentração do prato de topo (GL)
Y_i	Universo de discurso para variável de saída
y	Fração molar de vapor
y_n	Concentração do vapor no n -ésimo prato (GL)
$y(k)$	Variável autorregressiva
$y(n)$	Saída do sistema para espaço de estados
$y(t)$	Dados de saída de um sistema dinâmico
y_q^{crisp}	Saída fuzzificada
$\tilde{y}_i$	Variável linguística de saída
$\hat{y}$	Saída do modelo identificado por meio de redes neurais
z_i	Concentração do prato de alimentação (GL)
z_j	Saída para rede neural modelo NARMAX
α	Constante de momento algoritmo <i>backpropagation</i>
α_r	Volatilidade relativa
Δx	Espessura da coluna em metros
ε	Tolerância de erro algoritmo <i>backpropagation</i>
η	Constante de aprendizagem algoritmo <i>backpropagation</i>
η_a	Taxa de aprendizagem de uma rede adaptativa
θ	Vetor de parâmetros do polinômio do modelo de função de transferência
μ	Função de pertinência
τ_d	Tempo derivativo
τ_i	Tempo integral
Φ	Função não-linear modelo NARMAX
$\varphi(t)$	Vetor de regressão no tempo t

ABREVIATURAS E SIGLAS

A/D	Analógico/Digital
ANFIS	Sistema de Inferência <i>Fuzzy</i> para Rede Adaptativa (<i>Adaptive Network-based Fuzzy Inference System</i>)
AR	Modelo Autorregressivo
ARMAX	Modelo Autorregressivo com Média Móvel e Entrada Exógena
ARX	Modelo Autorregressivo com Entrada Exógena
BJ	Modelo Box-Jenkins
Bps	<i>Bits</i> por Segundo
CC	Controlador de Concentração
CLP	Controlador Lógico Programável
CMOS	Semicondutor de Metal-Óxido Complementar (<i>Complementary Metal Oxide Semiconductor</i>)
CRC	Verificação de Redundância Cíclica (<i>Cyclic Redundancy Check</i>)
DDE	Intercâmbio Dinâmico de Dados (<i>Dynamic Data Exchange</i>)
EEPROM	<i>Electrically Erasable Programmable Read Only Memory</i>
FIR	Modelo de Resposta Finita (<i>Finite Impulse Response</i>)
I/O	Entrada/Saída (<i>Input/Output</i>)
I ² C	Protocolo de Integração de Circuitos (<i>Inter-Integrated Circuit</i>)
IAE	Integral do Erro Absoluto (<i>Integrated Absolutted Error</i>)
IHM	Interface Homem Máquina
ISE	Integral do Erro Quadrático (<i>Integral of the Square of the Error</i>)
ITAE	Integral do Erro Absoluto Ponderado no Tempo (<i>Integral of Time and Absolute Error</i>)
LC	Controlador de Nível (<i>Level Controller</i>)
MA	Média móvel (<i>Move Average</i>)
MIMO	Sistema com Múltiplas Entradas e Múltiplas Saídas (<i>Multiple-Input, Multiple-Output</i>)
MISO	Sistema com Múltiplas Entradas e Única Saída (<i>multiple- input, single-output</i>)
NARMAX	Modelo Não linear Autorregressivo com Média Móvel e Entrada Exógena

OBF	Modelo de Base Ortonormal (<i>Orthonormal Basis Functions</i>)
OE	Modelo de Erro de Saída (<i>Output Error</i>)
PC	Controlador de Pressão (<i>Pressure Controller</i>)
PD	Controlador Proporcional Derivativo
PI	Controlador Proporcional Integral
PID	Controlador Proporcional Integral Derivativo
PWM	Modulação por Largura de Pulso (<i>Pulse Width Modulation</i>)
QC	Controlador de Concentração
RAM	<i>Random Access Memory</i>
RBF	Funções de Base Radial (<i>Radial Basis Functions</i>)
RISC	Conjunto Reduzido de Instruções (<i>Reduced Instruction Set Computer</i>)
ROM	<i>Read Only Memory</i>
RS232	<i>Recommended Standard 232</i>
RTD	Termoresistências (<i>Resistance Temperature Devices</i>)
SCADA	Sistema de Supervisão, Controle e Aquisição de Dados (<i>Supervisory Control and Data Acquisition</i>)
SCI	Interface de Comunicação Serial (<i>Serial Communication Interface</i>)
SIMO	Sistema com Única Entrada e Múltiplas Saídas (<i>Single-Input, Multiple-Output</i>)
SISO	Sistema com Múltiplas Entradas e Múltiplas saídas (<i>Single-output, single-input</i>)
SPI	Periférico de Interface Serial (<i>Serial Peripheral Interface</i>)
TM	<i>Trade Mark</i> (Marca Registrada)
TTL	Lógica Transistor Transistor (<i>Transistor Transistor Logic</i>)
ULA	Unidade Lógica Aritmética
USART	Receptor/Transmissor Universal Síncrono/Assíncrono
Vac	Tensão em corrente alternada
Vcc	Tensão em corrente contínua

Índice de Texto

CAPÍTULO 1 – INTRODUÇÃO	1
1.1 Objetivos da Tese.....	1
1.2 Introdução Geral	2
1.3 Organização da Tese	2
1.4 Contribuições do Trabalho.....	3
CAPÍTULO 2 - PRINCÍPIOS DE DESTILAÇÃO BINÁRIA E CARACTERÍSTICAS DA COLUNA ESTUDADA	5
2.1 Descrição do Processo de Destilação	5
2.2 Descrição da Coluna de Destilação	6
2.3 Configurações Internas	7
2.4 Estratégias de Controle Típicas de Colunas de Destilação	12
2.5 Perda de Calor na Parede da Coluna de Destilação.....	16
2.5.1 Cálculo da perda de calor na parede da coluna de destilação.....	18
2.5.2 Aplicação do cálculo da perda de calor na coluna estudada.....	19
CAPÍTULO 3 - SISTEMA DE CONTROLE, AQUISIÇÃO DE DADOS E SUPERVISÃO.	21
3.1 Instrumentação em Processos Industriais.....	21
3.2 Placa de Aquisição de Dados	22
3.3 Descrição do Sistema de Supervisão, Controle e Aquisição de Dados e Instrumentação aplicados à Coluna de Destilação.	23
3.4 Microcontrolador Pic 16F877A.....	24
3.4.1 Estrutura interna	25
3.4.2 Comunicação serial - USART	26
3.4.2.1 Modo de comunicação assíncrono.....	27
3.5 Termômetro Digital Dallas DS18B20	28
3.5.1 Descrição do sensor de temperatura	30
3.5.2 Operação do DS18B20	32

3.5.3 Código ROM de 64 <i>Bits</i>	32
3.5.4 Memória	33
3.5.5 Calibração dos sensores de temperatura	34
3.5.6 Sistema do barramento 1- Wire	36
3.5.6.1 Configuração do <i>hardware</i>	37
3.5.6.2 Sequência da comunicação	37
3.5.6.2.1 Inicialização	38
3.5.6.2.2 Comandos ROM	38
3.5.6.2.2.1 <i>Search</i> ROM [F0h]	38
3.5.6.2.2.2 <i>Read</i> ROM[33h]	38
3.5.6.2.2.3 <i>Match</i> ROM	39
3.5.6.2.2.4 <i>Skip</i> ROM [CCh]	39
3.5.6.3 Comandos de função do DS18B20	39
3.5.6.3.1 <i>Convert T</i> [44h]	40
3.5.6.3.2 <i>Read scratchPad</i> [BEh]	40
3.6 Protocolo 1 – Wire	40
3.6.1 Procedimento de inicialização: pulso de presença e <i>reset</i>	40
3.6.2 Leitura de dados	42
3.7 Placa de Controle de Potência	44
3.7.1 Circuito de <i>zero crossing</i>	45
3.7.2 Circuito de controle de tensão	46
3.8 Software de Supervisão	48
3.9 Comparativo entre o Sistema Desenvolvido e um Sistema Convencional de Controle, Aquisição de Dados e Supervisão	53
 CAPÍTULO 4 - OBTENÇÃO DO MODELO DA COLUNA DE DESTILAÇÃO POR MEIO DE TÉCNICAS DE IDENTIFICAÇÃO	56
 4.1 Aspectos Inerentes à Modelagem do Sistema	56
4.2 Modelagem Matemática de uma Coluna de Destilação Binária	57
4.3 Identificação de Processos	63
4.3.1 Elementos básicos para elaboração de modelos	65
4.3.2 Tipos de modelos	66
4.3.3 Família de modelos de função de transferência	67

4.4 Conceitos Básicos de Redes Neurais Artificiais	70
4.5 Identificação de Processos Utilizando Redes Neurais Artificiais	72
4.5.1 Implementação da rede neural artificial	74
4.5.1.1 Algoritmo <i>backpropagation</i>	75
4.6 Aplicação das Técnicas de Identificação para Obtenção do Modelo da Coluna de Destilação.....	78
4.6.1 Modelo ARX	80
4.6.2 Modelo Espaço de Estados	81
4.6.3 Modelo de Erro de saída (OE).....	82
4.6.4 Modelo ARMAX.....	83
4.6.5 Modelo Box Jenkins (BJ)	84
4.6.6 Redes neurais artificiais.....	86
 CAPÍTULO 5 – SIMULAÇÃO DAS ESTRATÉGIAS DE CONTROLE	 89
5.1 Estratégias de Controle	89
5.1.1 Aplicação de estratégias de controle em processos químicos	89
5.1.2 Estratégias empregadas em controle de colunas de destilação.....	91
5.1.3 Controle clássico	92
5.1.3.1 Controlador proporcional (P)	93
5.1.3.2 Controlador proporcional integral (PI).....	93
5.1.3.3 Controlador proporcional derivativo (PD)	93
5.1.3.4 Controlador proporcional integral derivativo (PID).....	94
5.1.3.5 Métodos de sintonia de controladores PID.....	94
5.1.3.5.1 Método de Ziegler e Nichols do ganho crítico	95
5.2.3 Lógica <i>fuzzy</i>	96
5.2.3.1 Projeto de um controlador <i>fuzzy</i>	98
5.2.3.1.1 Variáveis linguísticas.....	100
5.2.3.1.2 Valores lingüísticos	100
5.2.3.1.3 Regras linguísticas.....	101
5.2.3.1.4 Conjuntos <i>fuzzy</i> , lógica <i>fuzzy</i> e base de regras	101
5.2.3.1.5 Funções de pertinência	101
5.2.3.1.6 Fuzzificação.....	102
5.2.3.1.7 Defuzzificação	103

5.2.3.2 Estrutura dos controladores <i>fuzzy</i>	103
5.2.3.3 Sintonia do controlador <i>fuzzy</i>	104
5.2.4 Lógica neuro- <i>fuzzy</i>	106
5.2.4.1 ANFIS.....	107
5.2.4.1.1 Arquiteturas ANFIS e algoritmos de aprendizagem.....	108
5.3 Arquitetura ANFIS	111
5.3.1 Algoritmo de aprendizagem híbrido.....	114
5.4 Índices de Desempenho das Malhas de Controle.....	116
5.4.1 Análise de desempenho dos controladores sujeitos a perturbação na <i>carga</i>	117
5.4.1.1 Integral do erro absoluto (<i>IAE</i>).....	117
5.4.1.2 Integral do erro quadrático (<i>ISE</i>).....	117
5.4.1.3 Integral do erro absoluto ponderado no tempo (<i>ITAE</i>).....	118
5.5 Simulação das Estratégias de Controle	118
5.5.1 Parametrização do controlador <i>fuzzy</i>	119
5.5.1.1 Funções de pertinência	119
5.5.1.2 Determinação de regras	119
5.5.1.3 Sintonia dos controladores <i>fuzzy</i>	121
5.5.1.4 Parametrização do controlador neuro- <i>fuzzy</i>	122
5.6 Resultados Obtidos a partir das Simulações.....	125
5.6.1 Avaliação de desempenho dos controladores mediante perturbação no <i>setpoint</i>	125
5.6.2 Avaliação de desempenho dos controladores mediante perturbação na carga.	129
CAPÍTULO 6 – APLICAÇÃO DAS ESTRATÉGIAS DE CONTROLE	134
6.1 - Aplicação do Sistema à Coluna de Destilação do LabEQ2 (DEQ/UEM).....	135
6.2 – Aplicação das Estratégias de Controle na Coluna de Destilação	139
6.2.1 Desempenho dos controladores mediante perturbação na carga.	141
6.2.2 Desempenho dos controladores mediante perturbação no <i>setpoint</i>	147
CAPÍTULO 7 - CONCLUSÕES E PERSPECTIVAS	151
7.1 - Perspectivas para Trabalhos Futuros	156
REFERÊNCIAS	157

8. APÊNDICE – PROGRAMAS DESENVOLVIDOS	166
8.1 – Programa Microcontrolador Pic16f877A (Placa de Controle e Aquisição de Dados	166
8.2 – Programas Identificação (<i>Software</i> MATLAB™).....	179

CAPÍTULO 1 – INTRODUÇÃO

O processo de separação mais amplamente usado na indústria química é a destilação. A separação dos componentes da mistura está baseada nas diferenças de volatilidade dos componentes. A fase vapor entra em contato com a fase líquida e há transferência de massa do líquido para o vapor e vice versa, pois o líquido e o vapor contêm, em geral, os mesmos componentes em quantidades relativas diferentes. Como a destilação consome uma grande quantidade de energia, é fundamental conhecer adequadamente o seu funcionamento, para que ela possa atuar com a máxima eficiência.

Se for considerada a operação típica de uma coluna de destilação em um processo industrial, a coluna será um dos equipamentos do processo e estará sujeita a distúrbios inerentes ao processo de separação. Para quantificar os efeitos das perturbações na coluna de destilação e atuar no processo de forma que estes distúrbios não afetem a qualidade do produto final a ser obtido, é fundamental o uso de instrumentação aliada a um sistema de controle eficaz que garanta as condições operacionais do equipamento.

1.1 Objetivos da Tese

O presente trabalho tem por objetivo a implementação de diferentes estratégias de controle, como o convencional (PID), *fuzzy* e *neuro-fuzzy* em uma coluna de destilação atualmente utilizada como módulo didático no Laboratório de Engenharia Química II (LabEQ2) do Departamento de Engenharia Química da Universidade Estadual de Maringá, avaliando o desempenho das mesmas para posterior aplicação em colunas de porte industrial.

Também é objetivo do trabalho o desenvolvimento e a implementação de um sistema de aquisição de dados e controle, constituído de uma placa desenvolvida com o emprego de um microcontrolador PIC de 14 *bits* e sensores de temperatura inteligentes interligados via protocolo 1-Wire e *software* de supervisão de dados desenvolvido em linguagem orientada a objetos (Delphi).

1.2 Introdução Geral

Processos de separação são amplamente utilizados nas indústrias químicas. Nesta categoria merece destaque a destilação, atuando como um elemento chave em indústrias de transformação de matérias-primas para a obtenção de produtos finais dentro de especificações requeridas. Assim, é essencial o emprego de sistemas de instrumentação e controle para assegurar a qualidade do produto final, devido ao fato do processo estar sujeito a perturbações. Particularidades comuns à destilação, como: dinâmica lenta e acoplamento entre as malhas de controle, podem levar a elevados períodos de transição desde a perturbação até a obtenção do regime permanente, afetando a qualidade do produto final que se deseja obter.

A aplicação de controladores convencionais PID tem sido feita sistematicamente em um grande número de processos químicos, sendo que, com este controlador, os resultados de desempenho foram melhores em processos lineares e dinâmicas de baixa ordem. No caso do controle da coluna de destilação, existe uma grande dificuldade de ajuste dos parâmetros do controlador, devido ao fato deste equipamento necessitar de controle multivariável e de apresentar malhas com um alto grau de acoplamento.

Desta forma, a aplicação de técnicas de controle avançado, como *fuzzy* e *neuro-fuzzy*, permite a aplicação dos sistemas de controle denominados especialistas, melhorando significativamente o desempenho do processo.

1.3 Organização da Tese

A descrição detalhada das etapas que compõem este trabalho, desde a concepção da placa de aquisição de dados e controle, simulação, sintonia até a implementação dos controladores, está descrita nos capítulos que compõem a tese.

A tese está organizada em capítulos para uma melhor apresentação da metodologia empregada, resultados obtidos, conclusões e perspectivas para trabalhos futuros.

No capítulo 2 são expostos os conceitos teóricos básicos do processo de destilação, suas características e particularidades, além de apresentar uma descrição detalhada da coluna de destilação estudada.

No capítulo 3 é apresentada a placa de aquisição de dados desenvolvida utilizando um microcontrolador RISC de 16 bits, responsável pela captura dos sinais de temperatura

da coluna de destilação e interligação de sensores inteligentes DS18B20. Também é descrita a placa de controle de potência desenvolvida para controlar o aquecimento do refulvedor, sendo esta placa integrada de uma forma modular com a placa de aquisição de dados. O sistema de supervisão foi desenvolvido em linguagem orientada a objetos, no qual é feita a interface homem máquina (IHM), para supervisionar o processo e a interface para coleta e armazenamento de dados e integração com as placas de aquisição de dados e de potência.

O capítulo 4 dedica-se à descrição dos modelos clássicos de identificação ARX, ARMAX, BJ, OE e Espaço de Estados, à obtenção do modelo a partir destes modelos e à comparação com o modelo obtido a partir de redes neurais artificiais.

No capítulo 5 é realizada uma análise comparativa das estratégias de controle PID, *fuzzy* e neuro-*fuzzy*, utilizando índices de desempenho, mediante perturbações na carga e no *setpoint* com o auxílio de simulações com o modelo obtido via identificação realizada no capítulo 4.

No capítulo 6 as estratégias de controle PID, *fuzzy* e neuro-*fuzzy* simuladas e parametrizadas no *software* MATLABTM são implementadas na coluna de destilação estudada. São ilustrados os resultados alcançados e a análise de desempenho dos controladores no processo, quando este é sujeito a perturbações na carga e no *setpoint*.

No capítulo 7 são avaliados os resultados obtidos, sendo apresentadas as conclusões finais e sugestões para trabalhos futuros.

Finalmente, são relacionadas as principais referências que contribuíram para a concretização deste trabalho e, por último, nos apêndices, apresentam-se os programas desenvolvidos para o microcontrolador PIC da placa de aquisição e controle de dados e os programas desenvolvidos no *software* MATLABTM para a obtenção do modelo da coluna de destilação por meio de técnicas de identificação.

1.4 Contribuições do Trabalho

- Desenvolvimento de *hardware* de baixo custo para aquisição de dados e controle de processos e sua aplicação numa coluna de destilação;
- Desenvolvimento de um sistema de supervisão de baixo custo em comparação aos sistemas tradicionais empregados em colunas de destilação de porte industrial;

- Obtenção do modelo da coluna de destilação por meio do emprego de técnicas de identificação de processos;
- Implementação e discussão dos resultados da aplicação de técnicas de controle avançado a uma coluna de destilação piloto para possivelmente no futuro aplicá-las em uma coluna de porte industrial;
- Possibilidade de aplicação e difusão de estratégias de controle avançado em colunas de destilação.

CAPÍTULO 2 - PRINCÍPIOS DE DESTILAÇÃO BINÁRIA E CARACTERÍSTICAS DA COLUNA ESTUDADA

Neste capítulo, é apresentada uma breve descrição do processo de destilação, levando em consideração suas características e particularidades. Também é apresentada uma revisão bibliográfica das estratégias típicas de controle aplicadas em colunas de destilação.

2.1 Descrição do Processo de Destilação

A operação unitária destilação também é denominada fracionamento ou destilação fracionada. A separação dos componentes da mistura está baseada nas suas diferenças de volatilidade. Na destilação, uma fase vapor entra em contato com uma fase líquida e há transferência de massa do líquido para o vapor e vice versa, pois o líquido e o vapor contêm, em geral, os mesmos componentes em quantidades relativas diferentes. Primeiramente, uma substância em estado líquido é aquecida até atingir a temperatura de ebulição, até transformar-se em vapor. Para aumentar a eficiência do processo, são utilizados dispositivos chamados pratos ou bandejas, espaçados em seqüência em uma coluna vertical e encapsulados em um invólucro cilíndrico, de modo a provocar um contato íntimo entre duas fases (PERRY e GREEN, 1997). A tendência natural é o componente mais leve (mais volátil) se concentrar na fase vapor e os componentes mais pesados (menos voláteis) permanecerem na fase líquida.

Para promover a separação são requeridas grandes quantidades de energia para promover a vaporização e a condensação da mistura a ser destilada. De acordo com WILSON (2000), a destilação tem ainda três grandes vantagens sobre outros métodos de separação para misturas de fluidos homogêneos:

- 1- Possibilidade de trabalhar com uma larga variação de vazão de alimentação. Muitos dos processos alternativos para separação de misturas de fluidos podem trabalhar somente com baixas vazões de alimentação, enquanto a destilação pode ser projetada para a separação de vazões extremamente grandes ou pequenas de alimentação.

- 2- A capacidade de separar a carga alimentada dentro de uma larga faixa de concentrações de alimentação. Na maioria dos métodos de separação, consegue-se separar fluidos que são relativamente puros.
- 3- A capacidade de separar um produto destilado de alta qualidade, enquanto a maioria dos outros métodos de separação realiza uma separação parcial e não consegue obter componentes puros.

O vapor dentro da coluna é gerado a partir do aquecimento da mistura por meio de um refulvedor (trocador de calor) situado na sua base. Assim, o vapor tende a subir enquanto a fase líquida desce, possibilitando assim a troca de calor e massa. O vapor se enriquece no componente mais volátil, enquanto o líquido se enriquece no componente menos volátil em cada um dos estágios de equilíbrio.

2.2 Descrição da Coluna de Destilação

Uma coluna de destilação pode operar nas formas contínua ou batelada. Na forma contínua, o equipamento opera em regime permanente, sendo alimentado continuamente. Na forma batelada, a coluna é carregada com toda a mistura a ser separada e o processamento é efetuado. Após ter sido concluído, os produtos são removidos (KISTER, 1992).

Quando são feitas operações de pequena escala, ou em operações em que o processo corrosivo é muito acentuado, utiliza-se o processo em batelada. Entretanto, na maior parte das operações de processamento, é economicamente viável manter o equipamento em operação contínua, com o mínimo de perturbações ou de paradas. Assim, pode-se obter uma maior produtividade e, por consequência, um menor custo unitário do produto final. Desta forma, o tempo não é uma variável na análise do processo em regime permanente.

A maioria das colunas de destilação em processos industriais pode levar horas, ou até mesmo dias para atingir o regime permanente.

Entretanto, na indústria, as condições operacionais não são constantes, devido às perturbações existentes no processo. Para contorná-las, é necessário o projeto de sistemas de controle robustos, capazes de manter o processo dentro de um regime de operação para obter a qualidade do produto final desejada.

A coluna de destilação é geralmente fabricada para funcionar como um equipamento contínuo. Dependendo do seu tamanho, ela é usualmente um conjunto de seções soldadas. Em algumas aplicações, nas quais manutenções frequentes são requeridas, a coluna é fabricada em seções flangeadas aparafusadas para permitir sua desmontagem. Como a coluna de destilação é vista como um cilindro no qual o vapor deve escoar para cima e o líquido para baixo, e onde estão os estágios ou bandejas, deve possuir um diâmetro tal que permita manter as velocidades de vapor a níveis aceitáveis.

As velocidades de líquido e de vapor são função do projeto do equipamento e das razões de vapor-líquido (L/V). Estas são variáveis importantes que devem ser levadas em consideração para promover a separação ao longo das seções da coluna.

2.3 Configurações Internas

A destilação necessita sucessivos estágios de vaporização-condensação para realizar a separação desejada. Quanto mais íntimo o contato entre vapor e líquido que entram no estágio, mais eficiente é o estágio e menor o seu número para obter a separação requerida.

2.3.1 Pratos de destilação: Um prato de destilação é uma placa horizontal que mantém uma acumulação de líquido descendente, geralmente com alguns centímetros, e através da qual flui o vapor ascendente. Para permitir esta passagem de vapor, o prato deve possuir furos de algum tipo. De acordo com o projeto, o líquido pode descer através das mesmas aberturas, ou, ainda, por vertedouros laterais. A coluna estudada neste trabalho possui pratos perfurados sem vertedouros laterais.

Quando não se utilizam vertedouros, os pratos podem ser chamados de pratos perfurados (regadeiras). A Figura 2.1 ilustra a operação deste tipo de prato com um pequeno fluxo de líquido e de vapor.

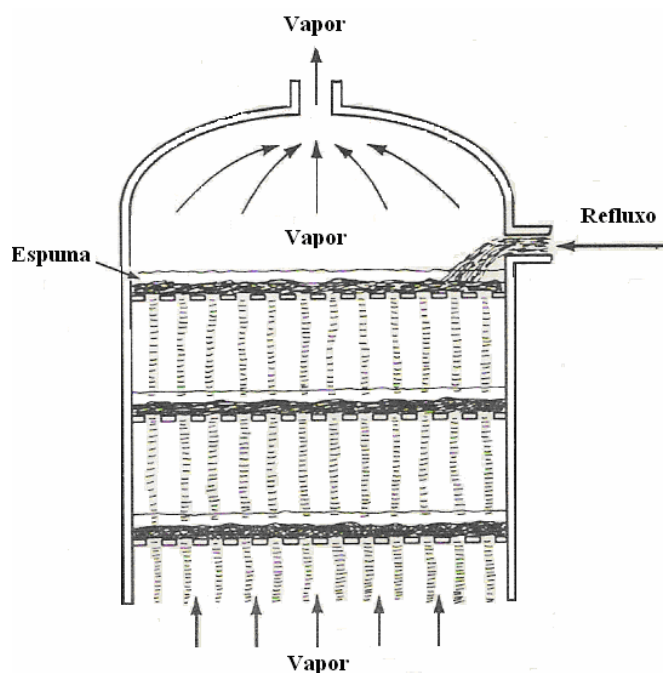


Figura 2.1 - Operação de pratos do tipo perfurado com baixos fluxos de líquido e de vapor (NISENFELD e SEEMAN, 1981).

O nível indicado no líquido em cada um dos pratos é resultado da queda de pressão ocasionada pelo fluxo em contracorrente das vazões de líquido e de vapor. O contato eficiente entre líquido e vapor requer um fluxo turbulento de vapor através do líquido, isto é, com turbulência suficiente para que a superfície interfacial criada permita aos materiais uma rápida transferência de massa entre as fases. Os vapores ascendentes dão aeração ao líquido ocasionando formação de espuma.

Associado com a formação da espuma, aparece o arraste de líquido pelos vapores ascendentes, de maneira que o líquido arrastado é conduzido ao prato imediatamente superior. Dentro de certos limites, a quantidade de líquido arrastado diminui com a distância conforme o vapor ascende. Consequentemente, o arraste diminui conforme se aumenta a distância entre os pratos.

2.3.2 Condensadores: A função do condensador é condensar o vapor que está no topo da coluna para fornecer refluxo suficiente para a separação. Condensadores resfriados a água são os mais comumente usados.

2.3.3 Refervedores: Para fornecer à coluna calor suficiente para o processo de destilação, é necessário a utilização de refervedores. Sua função é vaporizar o produto de fundo para fornecer vapor ao processo de separação.

2.3.4 Pré-aquecedores: A temperatura de alimentação da coluna é escolhida para prevenir restrições relativas à sua seção de esgotamento, na qual vapor extra teria que ser fornecido ao processo para suprir suas necessidades térmicas devido à temperatura da alimentação. Dessa forma, pode ser mais econômico pré-aquecer a alimentação, reduzindo assim a carga na seção de esgotamento.

2.3.5 Tanque acumulador: O tanque acumulador de alimentação existe para possibilitar uma alimentação constante para a coluna, mantendo-se um fluxo constante.

O caso estudado no presente trabalho trata da separação da mistura etanol-água, sendo essa mistura considerada um sistema não ideal.

A Figura 2.2 mostra o processo de separação da mistura a partir de divisões do equilíbrio líquido-vapor em 3 zonas distintas de processo e de equipamentos: esgotamento, retificação e desidratação.

Esta divisão é fundamental para projetos de equipamentos e sistemas no desenvolvimento das etapas da destilação.

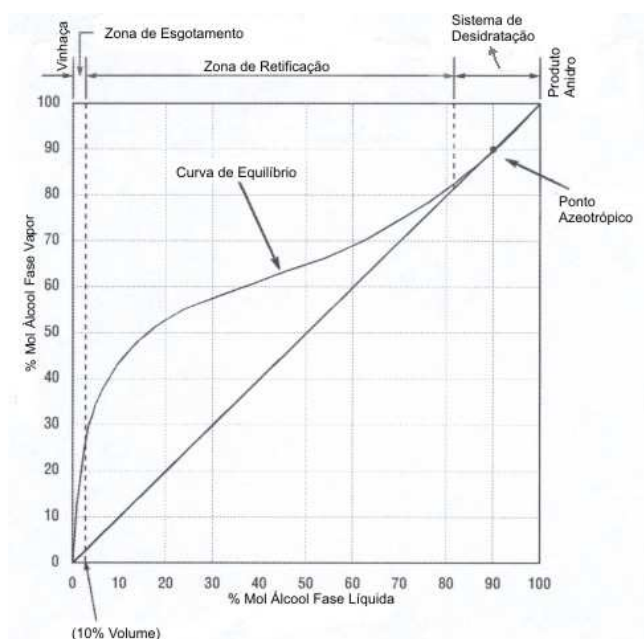


Figura 2.2 - Equilíbrio líquido-vapor para mistura etanol-água a 1 atm (JACQUES, 1995).

Para uma mistura binária, a pressão e a temperatura determinam as composições do equilíbrio entre o vapor e líquido. Assim, dados experimentais podem ser apresentados na forma de tabelas de fração molar de vapor y e fração molar de líquido x para um constituinte, em uma escala de temperatura T , para uma pressão fixa P , ou em uma escala de pressão P , para uma temperatura fixa T . Os dados para o sistema binário etanol-água na pressão de 1 atm são apresentados na Tabela 2.1.

Tabela 2.1 - Dados à pressão constante do equilíbrio vapor-líquido para a mistura etanol-água.

Temperatura (°C)	Fração molar	
	Líquido	Vapor
95,5	0,0190	0,1700
89,0	0,0721	0,3891
86,7	0,0966	0,4375
58,3	0,1238	0,4704
84,1	0,1661	0,5089
82,7	0,2337	0,5445
82,3	0,2608	0,5580
81,5	0,3273	0,5826
80,7	0,3965	0,6122
79,8	0,5079	0,6564
79,7	0,5198	0,6599
79,3	0,5732	0,6841
78,74	0,6763	0,7385
78,41	0,7472	0,7815
78,15	0,8943	0,8943

Assim, pode-se verificar o ponto de azeotropia na temperatura de 78,15°C, no qual as concentrações na fase vapor e na fase líquido são iguais a 0,8943.

O azeótropo é uma mistura de duas ou mais substâncias que, a uma certa composição, possui um ponto de ebulição constante e fixo, como se fosse uma substância pura. Um azeótropo pode ferver a uma temperatura inferior, intermediária ou superior às temperaturas dos componentes da mistura. Quando é inferior, chama-se azeótropo de mínimo ponto de ebulição ou azeótropo negativo. Quando é superior, azeótropo de máximo ponto de ebulição ou azeótropo positivo. A Figura 2.3 mostra os diagramas T-x para mistura binária ideal, com azeótropo de mínimo e azeótropo de máximo.

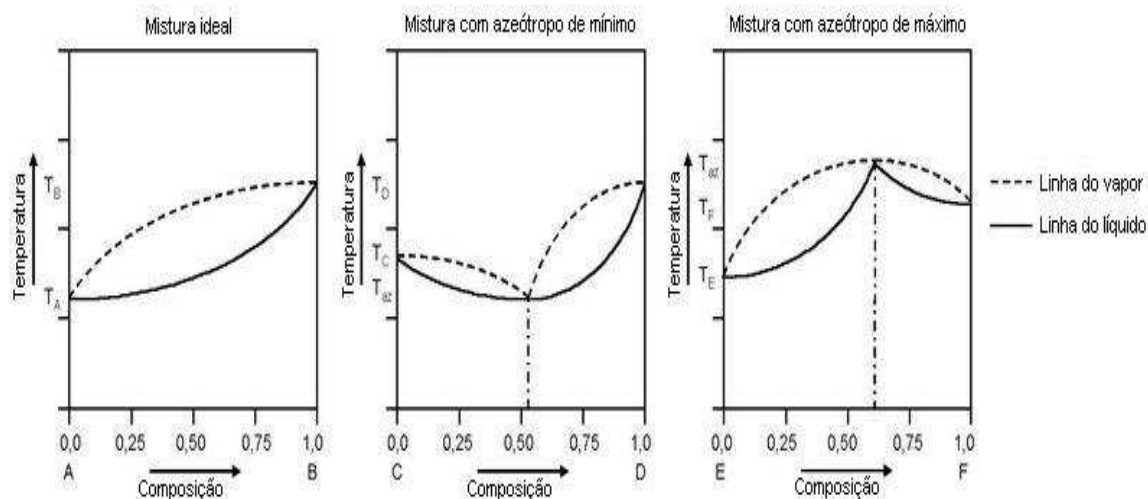


Figura 2.3 - Diagramas T-x para mistura binária ideal, com azeótropo de mínimo e azeótropo de máximo (PERRY e GREEN, 1997).

Na Figura 2.3, o gráfico à esquerda é de uma mistura ideal de duas substâncias A e B, sem formação de azeótropo, com as linhas de líquido e vapor saturados variando desde a temperatura de ebulição de A, a mais baixa, até a de B, a mais alta. No gráfico central, tem-se uma mistura binária de C e D que forma um azeótropo de mínimo, na qual a temperatura no azeótropo é menor do que as temperaturas de ebulição de C e de D, quando puros. À direita, é representada uma mistura binária E e F que forma um azeótropo de máximo. A temperatura no azeótropo é maior do que as temperaturas de ebulição de E e F, quando puros.

A destilação foi desenvolvida com o intuito de separar misturas de componentes que possuem diferenças significativas nas volatilidades relativas e pontos de ebulição, caso contrário, a separação por destilação convencional torna-se difícil e com alto custo devido à necessidade da colocação de um grande número de pratos para obter a separação requerida. Como os azeótropos não possuem diferenças em suas volatilidades, por meio da destilação convencional, nenhuma mudança na concentração é possível. Desta forma, deve-se levar em consideração a ocorrência da azeotropia no projeto e operação da coluna de forma a evitar o ponto de azeotropia, ou operar até o ponto de azeotropia e adicionar um solvente capaz de quebrar o azeótropo, sendo assim necessária uma coluna adicional para remover e reciclar o solvente. Os azeótropos são bem conhecidos e documentados, como, por exemplo, a mistura etanol-água, que forma um azeótropo de mínimo com etanol como maior componente.

Segundo WILSON (2000), estudos recentes têm sido focados na previsão da ocorrência de uma mistura azeotrópica (ou de misturas com muito pouca diferença na volatilidade de seus componentes) e, também, no projeto de colunas de destilação com o objetivo de produzir uma separação ótima em termos de especificação de produto, mínimo uso de solvente e, conseqüentemente, redução de custos de operação.

2.4 Estratégias de Controle Típicas de Colunas de Destilação

A operação de um sistema de controle de uma coluna de destilação de forma incorreta terá, como conseqüência, a ocorrência de instabilidades no processo. Estas instabilidades podem afetar a pureza do produto, capacidade de produção da coluna, aspectos econômicos da operação, podendo ainda afetar a segurança do processo e danificar a coluna de destilação. A ideia de se aplicar uma estratégia de controle à coluna de destilação é selecionar um dentre os diversos esquemas de controle possíveis que atenda as necessidades operacionais do processo, por meio de ações corretivas, possibilitando a operação do equipamento mesmo com a ocorrência de distúrbios.

A previsão de um comportamento dinâmico da coluna de destilação é bastante difícil, sendo que o conhecimento obtido com colunas semelhantes pode ser uma ótima ferramenta para possibilitar a implementação de estratégias de controle.

De uma maneira geral, um sistema de controle aplicado a uma coluna de destilação possui os seguintes objetivos:

- Promover condições de operação estáveis para a coluna de destilação;
- Ajustar variáveis manipuladas do equipamento de forma que os produtos sempre alcancem as especificações solicitadas;
- Alcançar os objetivos da forma mais eficiente. Isto significa obter o produto final com o máximo rendimento do processo, com o menor consumo de energia possível.

A instrumentação em uma coluna de destilação, aliada a um sistema de controle, é fundamental para quantificar os efeitos das perturbações e atuar no processo de forma que estas variações não afetem a qualidade do produto final a ser obtido.

A Figura 2.4 mostra as variáveis tipicamente controladas em uma coluna de destilação, exceto vazões mássicas.

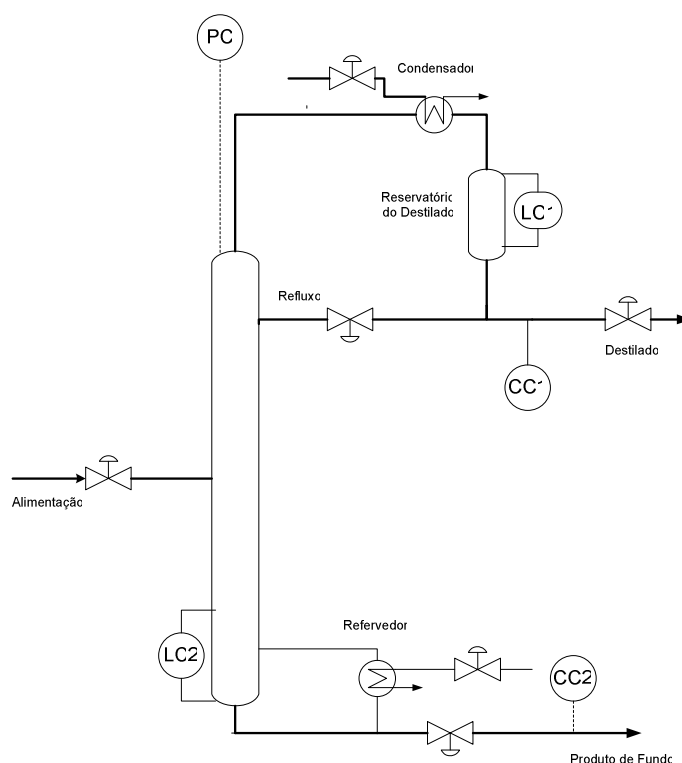


Figura 2.4 – Variáveis controladas e manipuladas em uma coluna de destilação típica.

Estas variáveis incluem: pressão da coluna (PC), níveis de produto de fundo (LC2), nível do acumulador (LC1), concentração do produto de topo (CC1) e concentração do produto de fundo (CC2). Segundo KISTER (1992), as variáveis podem ser classificadas em dois grupos:

- **Variáveis *single-loop*:** Este grupo inclui pressões e níveis. Estas variáveis são controladas a fim de obter a operação estável para a coluna. Os valores desejados destas variáveis são definidos para atender considerações de estabilidade do processo, sem levar em consideração a especificação dos produtos finais.

O controle de pressões e níveis regula a acumulação de matéria na coluna de destilação. Mantendo o nível constante previne-se acumulação de líquido, enquanto que o controle da pressão mantém o acúmulo de vapor a vazões mássicas desejáveis.

- **Variáveis de objetivo:** Este grupo inclui as concentrações de topo e de fundo, estas variáveis são controladas para se atingir os objetivos de operação da coluna.

Os controles de concentração podem ser feitos diretamente, isto é, usando sensores de concentração nas correntes da coluna, ou indiretamente, usando propriedades físicas representativas da concentração do produto, como: índice de refração, densidade, pressão de vapor ou temperatura de equilíbrio dos pratos.

Em resumo, uma coluna de destilação típica possui cinco variáveis controladas: três variáveis (pressão e dois níveis) controlados para promover uma operação estável da coluna e as duas concentrações (topo e fundo) para obter a pureza dos produtos finais especificada. As variáveis manipuladas são as vazões mássicas que podem ser controladas pela abertura das suas respectivas válvulas de controle.

De acordo com BEQUETTE (2003), uma configuração típica para o controle de uma coluna de destilação utilizando sensores de concentração é mostrada na Figura 2.5.

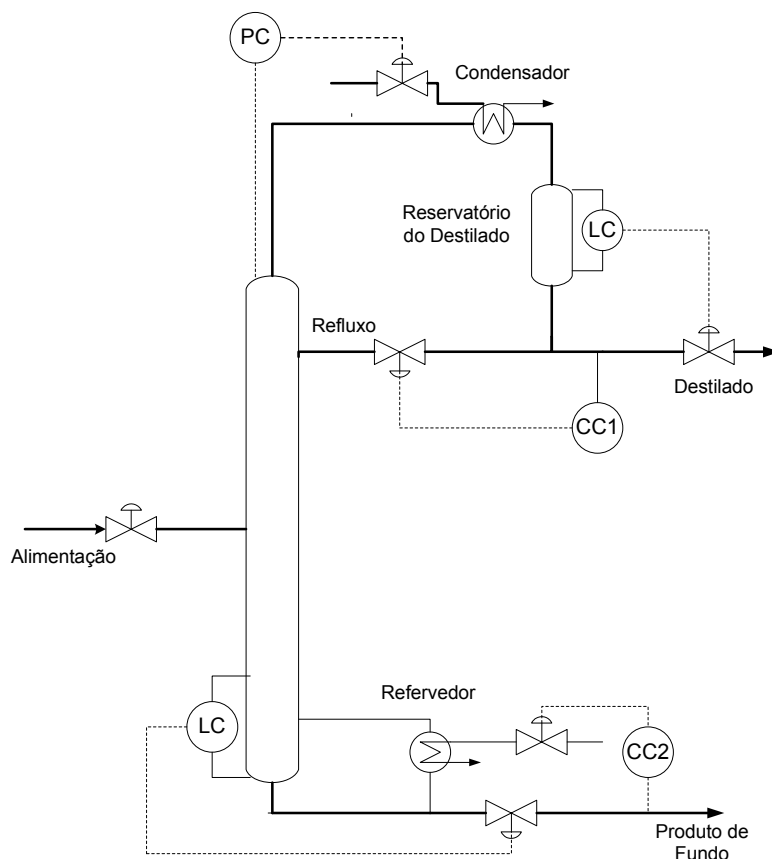


Figura 2.5 - Configuração típica de uma coluna de destilação utilizando sensores de concentração.

Nesta configuração existem cinco variáveis medidas (pressão, nível do produto destilado, nível de fundo, composição do destilado e composição de fundo) e cinco variáveis manipuladas (taxa de resfriamento do condensador, vazão de refluxo, vazão de destilado, vazão do produto de fundo, e taxa de aquecimento do refrervador). Além disso, se a alimentação da coluna é controlada, então haverá uma medição adicional (vazão de alimentação) e outra variável manipulada (% de abertura da válvula de alimentação).

Normalmente, considera-se que a pressão é controlada pela manipulação da taxa de resfriamento do condensador, dada pelo fluxo de água refrigerante no condensador. O nível

do destilado é controlado pela manipulação da vazão do destilado e o nível do fundo é controlado pela manipulação da vazão do produto de fundo, assim como a concentração de fundo pode ser manipulada pela taxa de aquecimento do refeedor e a concentração de topo pela razão de refluxo.

A idéia do controle de processo é fazer a atuação em conjunto dos pares de variáveis manipuladas e controladas. Assim, 5! (120) malhas de controle seriam possíveis. Entretanto, a implementação de todas estas estratégias de controle seria inviável dos pontos de vista de controle de processo e econômico.

Devido à dificuldade da medição de composição de uma forma contínua, podem ser instalados sensores de temperatura para inferir a concentração (Figura 2.6).

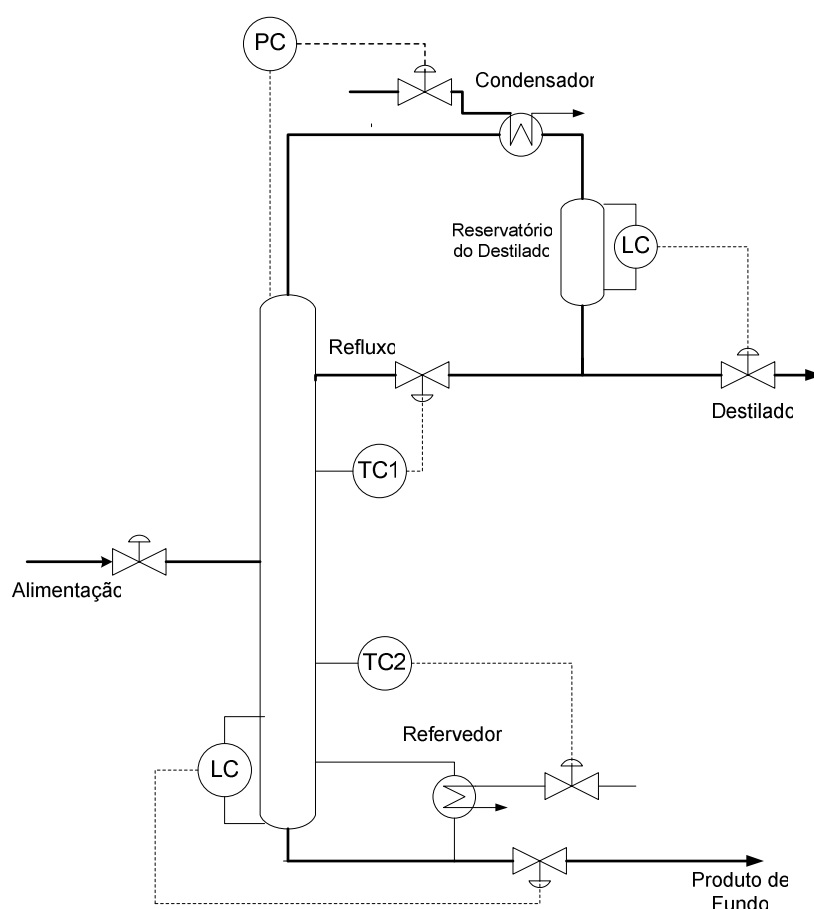


Figura 2.6 - Configuração típica de uma coluna de destilação utilizando sensores de temperatura para inferência da concentração.

Considerando a coluna de destilação estudada, a qual possui as válvulas com operação manual, não seria possível um controle automático do equipamento, alterando as vazões mássicas das correntes de produto. Como o objetivo do controle da coluna de destilação é controlar a concentração do produto de topo, será feito um controle de forma

indireta, ou seja, o controle não irá atuar nas correntes de produto e sim na taxa de aquecimento do refeedor, como ilustra a Figura 2.7.

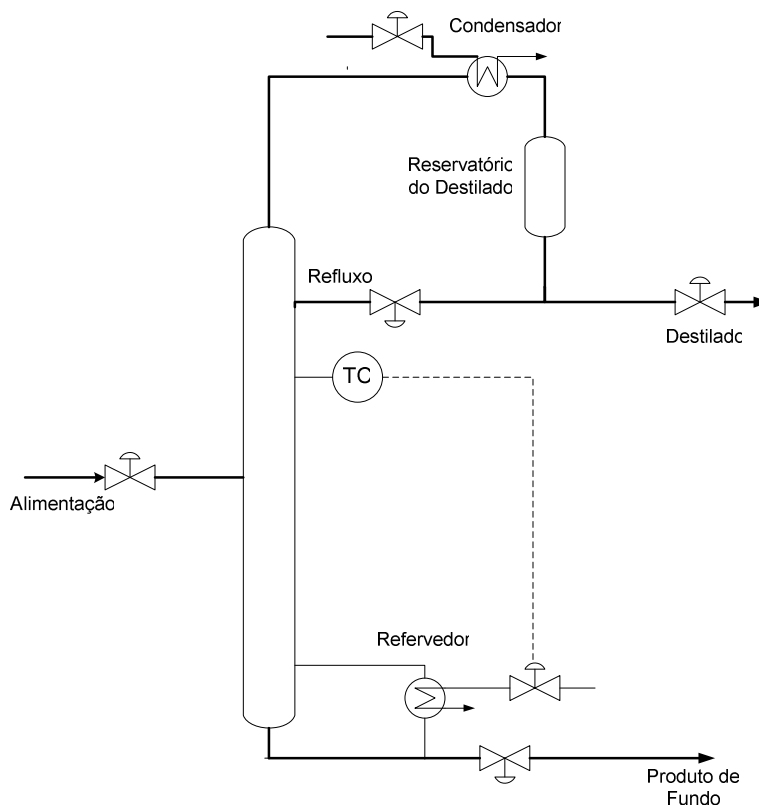


Figura 2.7 - Controle da coluna de destilação por meio da variação da taxa de aquecimento do refeedor.

A taxa de aquecimento do refeedor é controlada a fim de obter uma desejada pureza de um produto, em uma vazão mássica constante. Para controlar a taxa de aquecimento do refeedor, é levada em consideração a temperatura de um prato da coluna, ou a leitura de um sensor de concentração alocado no topo da coluna.

No caso específico da coluna de destilação estudada, foram utilizados sensores de temperatura ao longo da coluna para efetuar o controle do refeedor. Desta forma, o controle da coluna de destilação atua de forma automática para manter a concentração do produto final em face às perturbações.

2.5 Perda de Calor na Parede da Coluna de Destilação

Segundo HOLMAN (2002), a transferência de calor é a ciência que busca prever a transferência de energia que ocorre em corpos tendo, como força motriz, a diferença de

temperatura. A transmissão de calor, conforme a Segunda Lei da Termodinâmica, ocorre sempre de uma fonte de temperatura mais elevada para um receptor de temperatura mais baixa. Há três mecanismos básicos de transferência de calor: condução, convecção e radiação. Na maior parte dos casos e nas aplicações na engenharia, a transferência de energia ocorre devido à combinação de dois ou três destes mecanismos e nunca de forma isolada. No caso em estudo, tem-se a combinação dos mecanismos de condução e de convecção.

No mecanismo de condução, o fenômeno de transferência de energia ocorre através de um material fixo, como no caso da parede mostrada na Figura 2.8.

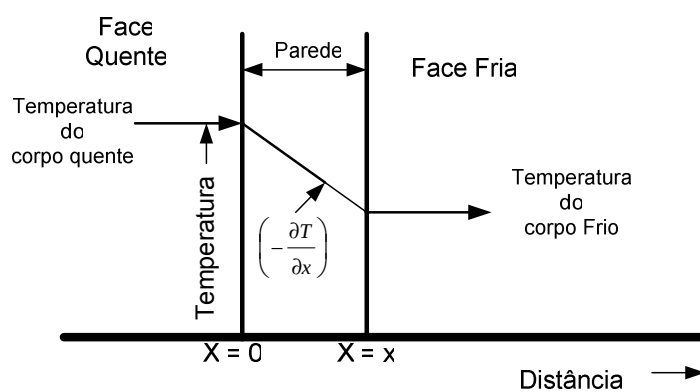


Figura 2.8 - Fluxo de calor através de uma parede.

Se existe um gradiente de temperatura em um corpo, existirá transferência de energia da região de temperatura mais alta para a região de temperatura mais baixa. A energia transferida por condução, por unidade de área, é proporcional ao gradiente de temperatura:

$$\left(\frac{q}{A} \right) \sim \frac{\partial T}{\partial x} \quad (2.1)$$

Considerando a direção do fluxo de calor ortogonal à parede e o corpo homogêneo e isotrópico. O fluxo de calor por unidade de tempo q é proporcional à variação de temperatura através da parede e à sua área A , T é a temperatura da parede em $^{\circ}\text{C}$ e x a espessura da parede na direção do fluxo de calor. O calor que flui na parede é dado pela chamada equação de Fourier:

$$q = kA \left(\frac{-\partial T}{\partial x} \right) \quad (2.2)$$

O termo $\left(\frac{-\partial T}{\partial x} \right)$ representa a variação da temperatura na direção de x e possui sinal negativo quando se supõe que a temperatura mais alta seja a da face da parede para $x=0$ e a mais baixa em $x=L$. Assim, pode-se deduzir que a quantidade de calor transferida instantaneamente é proporcional à área e à diferença de temperatura ΔT . A constante de proporcionalidade k é definida como a condutividade térmica do material. Baseado nesta definição, medidas experimentais podem ser feitas para determinar a condutividade térmica de diferentes materiais.

O mecanismo de transmissão de calor por convecção está associado ao movimento do fluido. Um fluido frio adjacente a uma superfície quente recebe calor, que é transmitido para todo o volume do fluido misturando-se com ele. Sendo assim, o mecanismo da convecção também está associado ao processo de mistura.

Quando o movimento do fluido não é incrementado por agitação mecânica, tem-se convecção natural ou livre. No caso em que exista agitação mecânica, a convecção é dita forçada.

O calor transferido por convecção pode ser descrito por uma equação que imita a forma da equação de condução, conhecida com lei de Newton de resfriamento:

$$q = hA\Delta T \quad (2.3)$$

A constante de proporcionalidade h , chamada coeficiente de transferência de calor por convecção, ou simplesmente, coeficiente de filme ou coeficiente de película, é uma variável influenciada pela natureza do fluido e pela natureza da agitação, e deve ser avaliada experimentalmente.

2.5.1 Cálculo da perda de calor na parede da coluna de destilação

Para a instrumentação da coluna de destilação os sensores de temperatura foram colocados na sua parte externa sem contato direto com o líquido em cada um dos pratos. Desta forma, existe uma diferença entre o valor lido experimentalmente e o valor real da temperatura no interior da coluna. Para corrigir este valor, foi desenvolvido um

procedimento para estimar a perda de calor no vidro da coluna. Para tanto, considerou-se a transmissão de calor entre o vidro e o sensor de temperatura na forma de condução e entre o sensor e o ar externo como convecção natural.

Como o fluxo de calor por área é constante, pode-se igualar as duas equações:

$$\frac{Q}{A} = k \left(\frac{\Delta T_{cond}}{\Delta x} \right) = h \Delta T_{conv} \quad (2.4)$$

A partir dessa igualdade, pode-se calcular a temperatura no interior da coluna de destilação. Para o cálculo de h , uma vez que a superfície na qual está instalado o sensor pode ser aproximada como uma placa plana, foi utilizada uma correlação para a convecção natural em uma placa plana vertical em regime laminar, segundo HOLMAN (2002):

$$h = 1,32 \left(\frac{\Delta T}{L} \right)^{0,25} \quad (2.5)$$

Nesta equação, ΔT é a diferença de temperatura entre a superfície quente e o fluido frio e L , a altura em metros.

2.5.2 Aplicação do cálculo da perda de calor na coluna estudada

Para estimar a quantidade de perda de calor no vidro da parede da coluna, a mesma foi considerada como uma superfície plana, como está ilustrado na Figura 2.9.

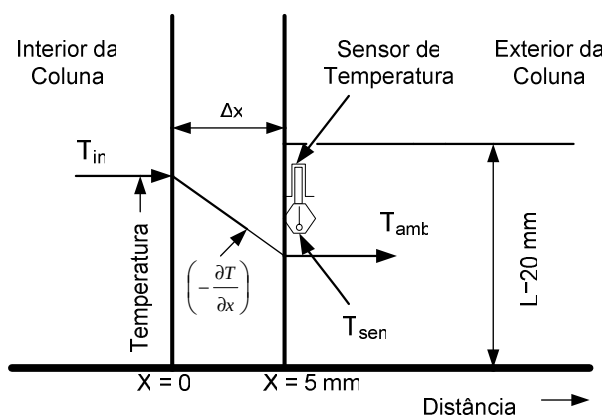


Figura 2.9 - Superfície da parede da coluna de destilação.

A seguir, são apresentados os parâmetros utilizados para o cálculo da perda de calor na parede da coluna de destilação:

$k = 1,09$ (HOLMAN, 2002) para vidro borosilicato ($\text{W/m}^\circ\text{C}$);

$$\Delta T_{\text{cond}} = T_{\text{in}} - T_{\text{sen}}$$

Sendo:

T_{in} = Temperatura interna na coluna em $^\circ\text{C}$

T_{sen} = Temperatura medida pelo sensor $^\circ\text{C}$

$$\Delta T_{\text{conv}} = T_{\text{sen}} - T_{\text{amb}}$$

Com:

T_{amb} = Temperatura ambiente dada em $^\circ\text{C}$

Para o cálculo do h (em $\text{W/m}^2\ ^\circ\text{C}$), foram consideradas as alturas de cada segmento anular da coluna (2 cm): $L = 0,02$ m

A espessura da coluna corresponde a 5 mm. Assim: $\Delta x = 0,005$ m.

O perfil de temperatura foi calculado de acordo com o trabalho desenvolvido por FRANCHI (2007) e o perfil corrigido de temperaturas é apresentado na Tabela 2.2.

Tabela 2.2 - Perfil corrigido de temperaturas nos pratos da coluna de destilação.

Pratos	Temperatura Medida ($^\circ\text{C}$)	Temperatura Interna ($^\circ\text{C}$)
2	66,28	67,86
7	68,28	69,96
9	71,12	72,95
10	73,08	75,01
11	72,11	73,99
12	73,53	75,49
13	73,91	75,89
14	75,11	77,15

Conforme o esperado, a Tabela 2.2 apresentou os valores de temperaturas internas superiores às medidas externamente por meio dos sensores colocados na face externa da coluna de destilação. Esta discrepância nos valores de temperatura é relevante e deve-se a resistência a transferência de calor imposta pela parede de vidro.

CAPÍTULO 3 - SISTEMA DE CONTROLE, AQUISIÇÃO DE DADOS E SUPERVISÃO.

Para alcançar os objetivos do trabalho proposto, foram desenvolvidos e implementados uma placa de aquisição de dados e controle, juntamente com um sistema de supervisão de dados aplicados a uma coluna de destilação localizada no Departamento de Engenharia Química da Universidade Estadual de Maringá. Para tanto, foi utilizado um sistema de supervisão com uma linguagem visual orientada a objetos, um microcontrolador PICTM de arquitetura RISC de 14 *Bits* conectado a sensores de temperatura DS18B20 DallasTM em redes *1-Wire*TM.

Neste capítulo estão contidas a placa de aquisição de dados e controle, bem como a descrição do *software* de supervisão, elaborado em linguagem orientada a objetos, juntamente com a descrição de suas características e particularidades.

3.1 Instrumentação em Processos Industriais

Um sensor pode ser definido com um dispositivo que recebe um estímulo e envia um sinal elétrico, sendo que o estímulo pode ser uma quantidade, propriedade ou condição.

Para que o sensor funcione apropriadamente, deve ser parte de um grande sistema de aquisição de dados que pode incorporar condicionadores e processadores de sinal, dispositivos de memória, atuadores, etc.

Assim, pode-se definir instrumentação como o estudo teórico e prático dos equipamentos responsáveis pelo controle, medição e registro e indicação de variáveis de um processo produtivo (instrumentos), que podem ser usados para monitorar de forma contínua ou discreta as variáveis de interesse do processo.

A partir da evolução tecnológica, foi possível a introdução em larga escala dos microcomputadores nos diversos processos produtivos, possibilitando o acompanhamento e controle por meio de *software* e controladores específicos dentro de ambientes industriais (ALVES, 2005).

Isto possibilitou uma melhoria na qualidade do produto, aumento em quantidade, segurança e melhoria do meio ambiente, com o auxílio do controle e monitoração do processo.

Com o desenvolvimento dos microprocessadores, uma nova categoria de sensores, chamada sensores inteligentes, começou a ser desenvolvida. De acordo com SINCLAIR (2001), este tipo de sensor usa uma miniatura de um sensor que é integrado em um único *chip* com um processador. No presente trabalho, são utilizados sensores inteligentes de temperatura da família DS18B20, com protocolo de comunicação digital *1-Wire*TM

3.2 Placa de Aquisição de Dados

Para a coleta de dados de temperatura e interligação com o sistema de supervisão, foi desenvolvida uma placa de aquisição de dados utilizando o microcontrolador RISC de 14 Bits PICTM 16F877A de 40 pinos e 33 portas de I/O's. É possível a conexão de até 16 sensores de temperatura digital DS18B20, em um barramento de comunicação *1-Wire*TM. Os dados são captados pelo microcontrolador e enviados a um conversor de níveis TTL/RS232, até chegar ao microcomputador que contém o *software* de supervisão.

A placa de aquisição de dados é responsável por fazer o controle da placa de controle de potência, com utilização de um controle automático ou manual efetuado pelo usuário em conjunto com o *software* de supervisão.

Além do controle manual ou automático de potência, a placa de aquisição de dados efetua a coleta de temperaturas do processo utilizando sensores digitais DS18B20 (DALLAS SEMICONDUCTOR, 2001). Este tipo de sensor trabalha com um protocolo digital de comunicação conhecido como *1-Wire*. A Figura 3.1 apresenta a placa de aquisição de dados desenvolvida.

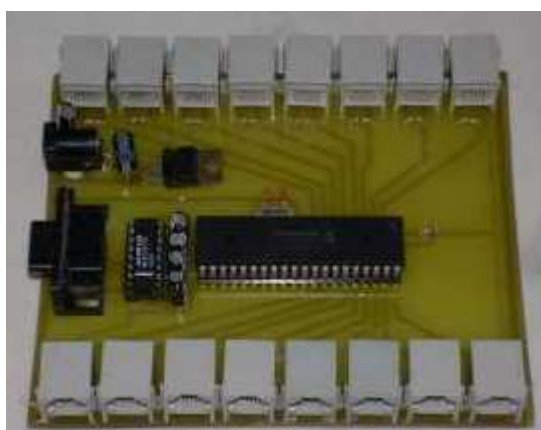


Figura 3.1 - Placa de aquisição de dados desenvolvida.

Durante a operação, a placa de aquisição de dados troca constantemente dados com o *software* de supervisão, tanto enviando valores de temperatura ao *software*, quanto

recebendo sinais que serão enviados à variável manipulada do processo de destilação, por meio de uma conversão de sinais TTL para RS232. A Figura 3.2 mostra o esquema elétrico da placa de controle e aquisição de dados.

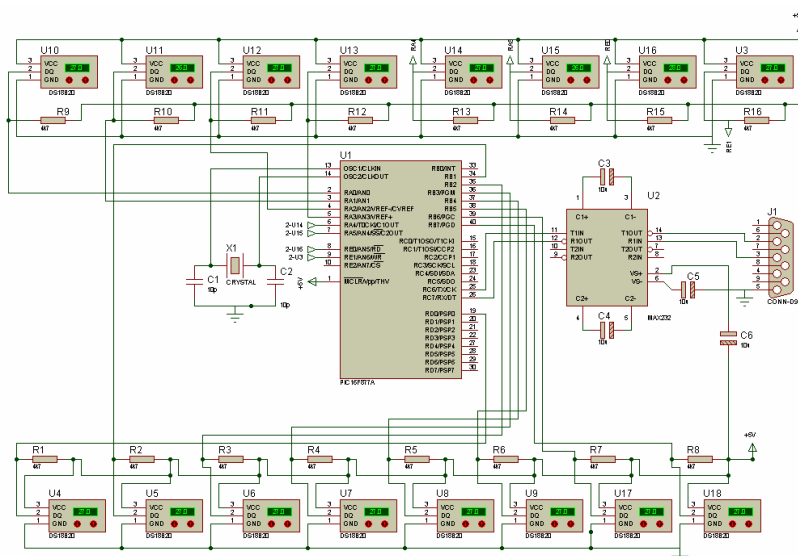


Figura 3.2 - Esquema elétrico da placa de aquisição de dados proposta.

3.3 Descrição do Sistema de Supervisão, Controle e Aquisição de Dados e Instrumentação aplicados à Coluna de Destilação.

A Figura 3.3 representa a instrumentação aplicada na coluna de destilação desenvolvida no presente trabalho, juntamente com a placa de aquisição de dados, placa de controle de potência e sistema de supervisão.

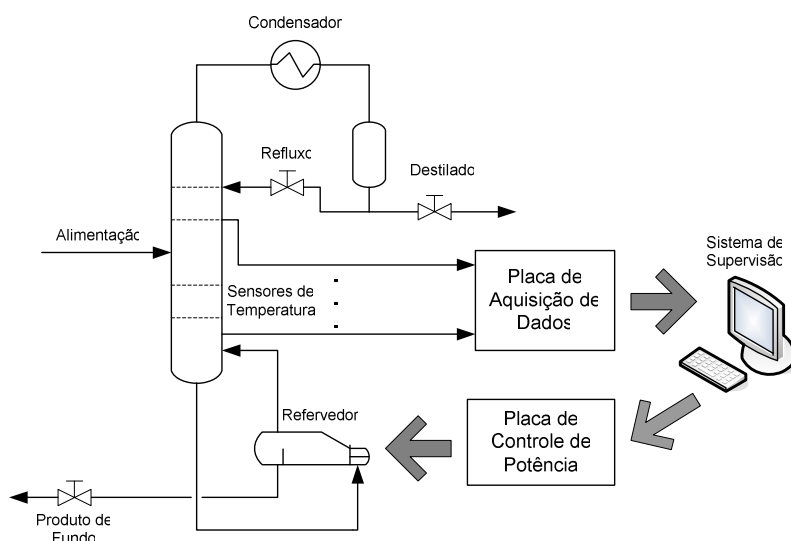


Figura 3.3 - Instrumentação aplicada na coluna de destilação.

No sistema desenvolvido, a comunicação dos sinais de temperatura com a placa de aquisição de dados foi realizada utilizando o barramento 1-WireTM. O sistema 1-WireTM desenvolvido pela empresa Dallas Semiconductor tem mostrado grande eficiência em aplicações remotas de monitoramento de temperatura em ambientes que possuam ruídos elétricos, nos quais o uso de sensores de temperatura analógicos é difícil (DESHPANDE, 1985). A interligação entre os sensores e o microcontrolador é feita por meio do protocolo 1-WireTM, sendo o sinal da placa de aquisição de dados enviado a um conversor de níveis TTL/RS232 (Max 232) para ser interligado até chegar ao microcomputador que contém o sistema de supervisão. Nas seções seguintes, serão apresentados com detalhes os elementos que compõem o sistema de aquisição de dados, controle e de supervisão.

3.4 Microcontrolador Pic 16F877A

O microcontrolador utilizado na realização do trabalho é um microcontrolador PICTM 16F877A, com as seguintes características (DALLAS SEMICONDUCTOR, 2001):

- ✓ Microcontrolador de 40 pinos, o que possibilita a montagem de um *hardware* complexo e capaz de interagir com diversos recursos e funções ao mesmo tempo;
- ✓ Via de programação com 14 *bits* e 35 instruções;
- ✓ 33 portas configuráveis como entrada ou saída;
- ✓ 15 interrupções disponíveis;
- ✓ Memória de programação EEPROM *FLASH*, que permite a gravação rápida do programa diversas vezes no mesmo chip, sem a necessidade de apagá-lo por meio de luz ultravioleta, como acontece nos microcontroladores de janela;
- ✓ Memória de programa com 8 *keywords*, com capacidade de escrita e leitura pelo próprio código interno;
- ✓ Memória EEPROM (não-volátil) interna com 256 *bytes*;
- ✓ Memória RAM com 368 *bytes*;
- ✓ Três *timers* (2x8 *bits* e 1x16 *bits*);
- ✓ Comunicações seriais: SPI, I²C e USART;
- ✓ Conversores analógicos de 10 *bits* (8x) e comparadores analógicos (2x);
- ✓ Programação *in-circuit* (alta e baixa tensão);
- ✓ *Power-on Reset* (POR) interno;
- ✓ *Brown-out Reset* (BOR) interno.

A Figura 3.4 apresenta o microcontrolador PIC 16F877A e a descrição da nomenclatura dos pinos do mesmo.

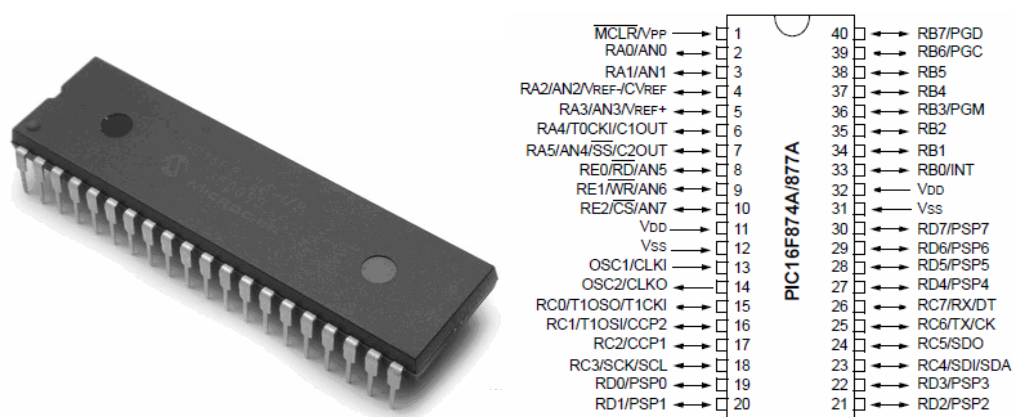


Figura 3.4 - Microcontrolador PIC 16F877A e descrição de pinagem.

3.4.1 Estrutura interna

De acordo com SOUZA (2003), no diagrama de blocos podem ser visualizadas as diversas partes que compõem o microcontrolador PIC16F877A. No centro encontra-se a ULA, que é a unidade de processamento e está diretamente ligada ao registrador Work (W reg). No canto superior esquerdo tem-se a memória de programa (*FLASH*). A partir desse bloco existe um barramento de 14 bits (*program Bus*). Também está presente a memória de dados (RAM) que possui um barramento de 8 bits (*Data Bus*). Neste microcontrolador existem 5 portas (PORTs), de PORTA a PORTE. Na parte inferior podem ser encontrados os demais periféricos, tais como a EEPROM (memória de dados não-volátil), os *timers* (TMRO, TMR1 e TMR2), os A/D's de 10 bits, os modos CCP (*Compare, Capture e PWM*), as comunicações seriais (SPI, I²C e USART), os comparadores e a tensão de referência. Observa-se que, entre todos os periféricos, a comunicação é feita por um barramento de oito vias. A Figura 3.5 mostra com detalhes a estrutura interna do microcontrolador utilizado.

No registrador de status (*STATUS reg*), estão alocadas informações importantes sobre as operações aritméticas da ULA ficam armazenadas, assim como os demais SFRs (registradores especiais). Na parte superior tem-se ainda o contador de linha de programa (*Program Counter*) e a pilha de oito níveis (*Stack*), em conjunto com os circuitos internos de reset, POR, BOR, osciladores, *Watchdog Timer* (WDT) e sistema de programação.

Os PIC's utilizam uma arquitetura conhecida como *Harvard*, segundo SOUZA (2003), que prevê várias vias de comunicação entre a *CPU* e os periféricos, permitindo a realização de várias operações simultaneamente, o que implica em um aumento considerável na velocidade de execução de uma instrução e permite ainda que a memória de dados e de programa tenha tamanhos diferentes. Esta facilidade permite ainda que em uma única palavra de 14 *bits* se tenha o código da instrução, onde a mesma vai atuar com o eventual operando ou dado, se houver. Criou-se uma terminologia chamada *RISC* (Computador com Set de Instruções Reduzido) que faz com que existam poucas instruções (aproximadamente 35 instruções).

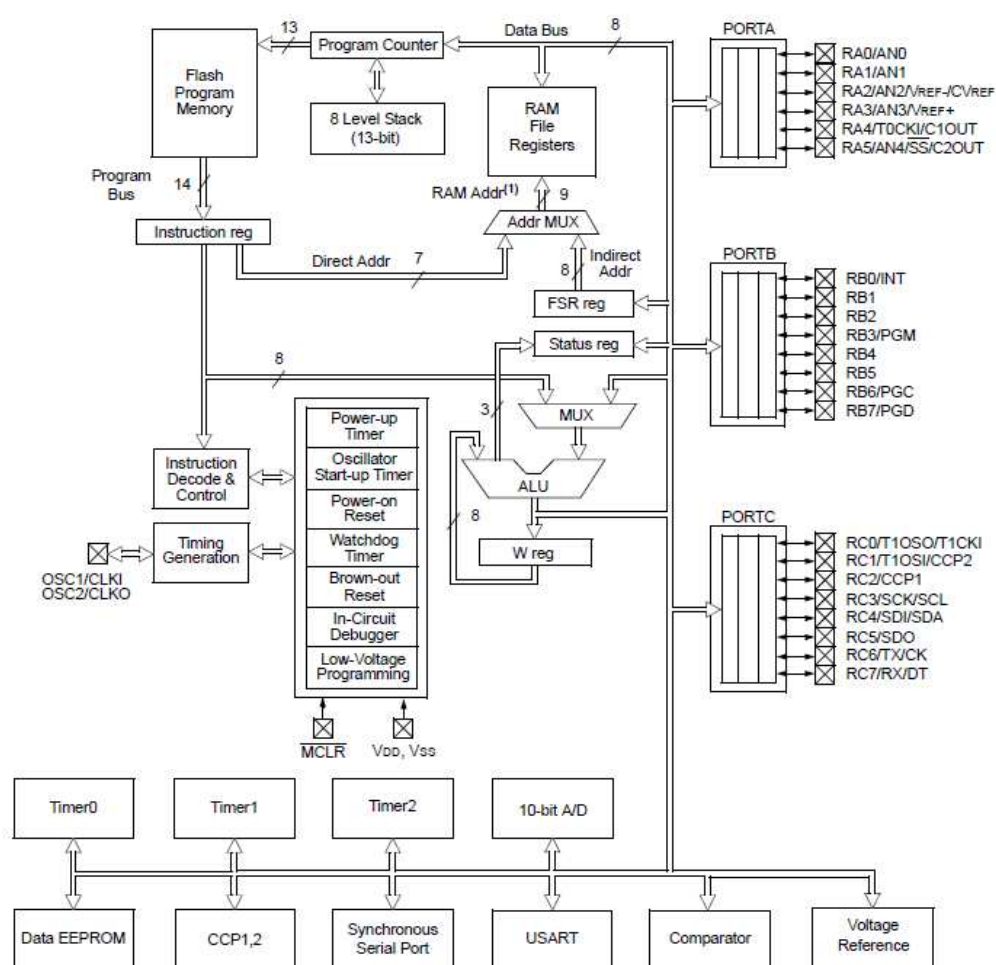


Figura 3.5 - Estrutura interna do microcontrolador utilizado.

3.4.2 Comunicação serial - USART

Para a conexão entre a placa de aquisição de dados e o *software* de supervisão foi utilizada a USART (*Universal Synchronous Receiver Transmitter*), também conhecida

como SCI (*Serial Communications Interface*). É um protocolo de comunicação universal que possui dois modos distintos de trabalho: o modo síncrono e o modo assíncrono. No caso estudado no presente trabalho a comunicação assíncrona que foi a escolhida como protocolo de comunicação.

3.4.2.1 Modo de comunicação assíncrono

A comunicação é feita somente com duas vias. Entretanto, como este modo não é sincronizado, essas duas vias são utilizadas para dados. Uma delas para transmissão (T_X) e outra para recepção (R_X). Isso possibilita que as informações sejam enviadas e recebidas ao mesmo tempo, cada qual na sua via. Este recurso é chamado de comunicação *Full Duplex*. A idéia de se usar este modo de comunicação serve para implementar a comunicação com a porta serial de computadores no padrão RS 232.

Em outros sistemas de comunicação, há um sincronismo entre os dados por meio de um *clock* para informar o momento correto de transmissão de cada bit por uma via de dados específica. No caso desta comunicação serial esta via não existe, sendo a sincronização feita pela própria via de dados.

Para compreender este sistema de comunicação é preciso entrar na definição de *Baud Rate*. Para que o sistema opere o tamanho dos dados (intervalo de cada bit) deve ser completamente padronizado, e ambos os lados devem estar ajustados para o mesmo valor. Como essa comunicação trabalha sempre com base nos *bits*, essa velocidade é normalmente indicada em *bits* por segundo, ou bps. Assim, é possível calcular o tempo de duração de cada bit.

$$T_{BIT} = \frac{1}{BaudRate} \quad (3.1)$$

Existe somente um sincronismo de tempo feito para a transmissão/recepção sendo esta responsabilidade ficando a cargo do *Start Bit*. Ambas as vias de comunicação (R_X , T_X) devem ser tratadas da mesma forma, pois o T_X de um dispositivo deve estar conectado ao R_X de outro dispositivo e vice-versa.

Ambas as vias possuem seu estado padrão como sendo o nível alto. Tem-se então uma situação de *stand by*. Quando é iniciada a transmissão, o T_X força para nível baixo mantendo-o assim pelo T_{BIT} . Esta borda de descida é reconhecida pelo outro lado (em R_X) e é suficiente para iniciar o processo de sincronização para recebimento desse *byte*. Este pulso de nível baixo é chamado de *Start Bit*. Após isto, inicia-se a comunicação com o T_X

enviando 8 *bits* de dados, todos eles com o mesmo tamanho do *Start Bit*. Assim, R_X sabe o momento do *Start Bit* e deixa transcorrer o tempo e depois coleta os 8 *bits*, mais ou menos no meio do tempo do T_{BIT} . Para finalizar o processo, T_X envia um *Stop Bit*, que nada mais é do que outro Bit com valor fixo em '1', garantindo que a linha voltará ao seu estado padrão e o sistema voltará ao *stand by*, ficando apto a receber um novo dado. O lado R_X deve considerar a leitura do *Stop Bit* para garantir que nenhum erro aconteça com a recepção.

Este padrão aceita também a comunicação com 9 *bits*, sendo que o bit adicional pode ser usado para dado paridade ou endereçamento.

É importante que ambos os lados (T_X e R_X) sejam configurados para operar com a mesma quantidade de *bits*. A configuração mais comum é 8 *bits* de dados (sem paridade), com um bit de *stop* e é chamada 8N1. O *Baud Rate* também obedece a valores pré-ajustados como: 300, 1200, 2400, 9600, 19200 bps, entre outros.

3.5 Termômetro Digital Dallas DS18B20

O sensor utilizado no trabalho é o termômetro digital Dallas DS18B20 da empresa Dallas Semiconductor, é um sensor de temperatura incorporado em um circuito integrado. Seu encapsulamento é o TO-92 com um tamanho ligeiramente maior do que um transistor de uso geral, não necessitando de componentes externos para a medição de temperatura, como é mostrado na Figura 3.6.



Figura 3.6 - Termômetro digital DS 18B20.

Este sensor é um termômetro digital que possui de 9 até 12 *bits* de resolução para medidas de temperatura. O sensor comunica-se em um barramento *1-WireTM* que necessita somente de uma linha de comunicação (e terra) para a comunicação com um microprocessador. O sensor possui uma escala de -55°C a $+125^{\circ}\text{C}$ e sua precisão é de $\pm$

0,5°C, na escala de 10°C a + 85°C. Também possui uma função de alarme com dados não voláteis, com valores programados pelo usuário.

Esse sistema pode ser denominado como uma rede de transmissão de dados que possibilita a comunicação digital entre um computador e dispositivos da série *1-Wire™*. O barramento *1-Wire™* é um protocolo mestre/escravo com um barramento simples para o mestre. Os dispositivos em *1-Wire™* são fornecidos pela Dallas Semicondutores e o mestre é definido pelo usuário como um microcontrolador ou microprocessador. O sistema *1-Wire™* é constituído, basicamente, por três elementos: um computador com programa de controle e gerenciamento das atividades, um protocolo específico de comunicação e condutores (meio físico de comunicação) e dispositivos remotos da série *1-Wire™*. Normalmente, a transmissão de dados ocorre por meio de um condutor e, além desse, existe um condutor específico para fornecer energia aos sensores e outro para o aterramento da rede. Por definição, o sistema *1-Wire™* possui apenas um condutor no qual são conectados todos os dispositivos da série, já que, por convenção, o condutor de referência ou aterramento não é considerado.

O fornecimento de energia para o sistema *1-Wire™* pode se dar de duas maneiras: alimentação parasita (derivada do condutor de transmissão de dados), sendo necessários somente dois condutores, ou alimentação externa, em que são necessários três condutores, além de uma fonte de alimentação regulada de 5 Vcc.

Uma condição prévia para o funcionamento de qualquer rede de transmissão de dados que empregue diversos dispositivos semelhantes é a existência de endereços ou códigos de identificação. Todo dispositivo *1-Wire™* possui um código de identificação único de 64 *bits* sequenciais, o que exclui a possibilidade de conflitos durante a transmissão de dados em redes com diversos dispositivos. O protocolo de comunicação do sistema *1-Wire™* utiliza níveis lógicos convencionais CMOS/TTL, nos quais o nível lógico 0 (zero) é representado por uma tensão máxima de 0,8 Vcc e o nível lógico 1 (um) por uma tensão mínima de 2,2 Vcc (DALLAS SEMICONDUCTOR, 2001).

No trabalho desenvolvido, são conectados ao microcontrolador 12 sensores DS18B20 por meio de um barramento *1-Wire™* que faz a comunicação entre um mestre e escravos por meio de uma conexão simples. Na primeira parte da comunicação, o mestre envia um sinal de *reset* para sincronizar todo o barramento. Um escravo então é selecionado para a comunicação. Uma vez selecionado para comunicação, o mestre envia um comando para leitura ou escrita, e o processo segue como mostrado na Figura 3.7. É muito importante que se tenha esta sequência em todo o momento em que o sensor seja

acessado, pois o mesmo não responderá se algum dos passos nesta sequência estiver faltando ou fora de ordem.

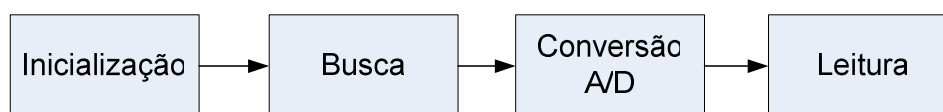


Figura 3.7 - Seqüência de operação dos sensores DS18B20.

Cada DS18B20 tem um código serial de 64 *bits* próprio, que permite colocar vários DS18B20 no mesmo barramento 1-Wire, e assim, com um único microprocessador, pode-se controlar uma grande quantidade de sensores.

Na Tabela 3.1 é apresentado um comparativo do sensor DS18B20 com sensores típicos de temperatura utilizados na indústria (termoresistências e termopares).

Tabela 3.1 - Quadro comparativo entre sensores típicos de temperatura e o DS18B20.

Característica	Termopar/Termoresistência	DS18B20
Condicionamento de Sinal	Necessita de amplificação, filtragem, excitação e compensação de sinal.	Incorporado no sensor.
Digitalização	Necessita de conversor A/D.	Conversor A/D incorporado com resolução de 12 bits.
Cabeamento	Termopar = 2 cabos; Termoresistência = 2, 3 ou 4 cabos.	Rede 1-Wire de 3 cabos.
Sinal de saída	Analógico; Limitação de distância; Menor integridade do sinal; Mais suscetível à ruído.	Digital; Possibilidade de conexão em rede; Preserva a integridade do sinal.
Endereçamento	Não é possível.	Detecção automática do endereço do sensor.
Memória	Não possui.	Memória <i>on board</i> .
Custos	Alto devido a necessidade de condicionadores de sinal e cabos específicos (termopar).	Baixo, somente controlador.

3.5.1 Descrição do sensor de temperatura

O DS18B20 utilizado é o modelo TO-92, mostrado na Figura 3.8.



Figura 3.8 - Descrição da pinagem do DS18B20.

Sendo:

GND - Terra;

DQ - Entrada de Dados/Saída de Dados (pino de interface 1-Wire);

V_{DD} - Alimentação 5 Vcc .

A Figura 3.9 apresenta um diagrama de blocos descritivo do DS18B20

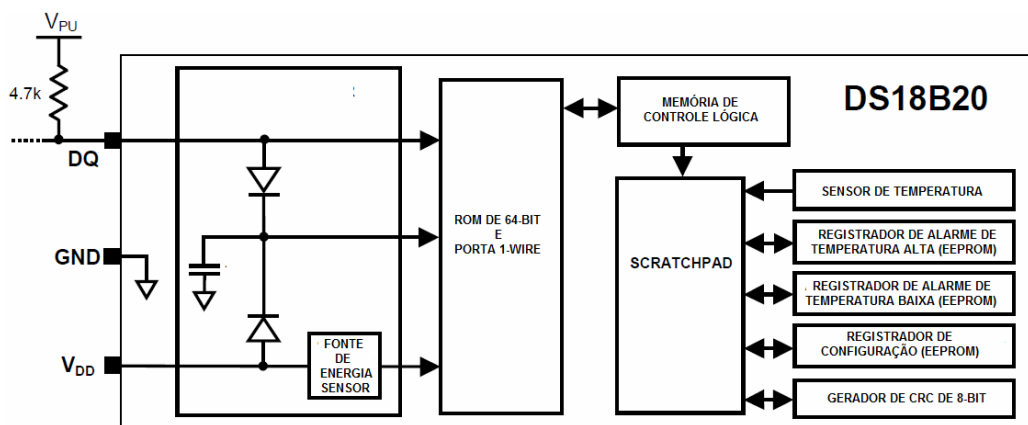


Figura 3.9 - Diagrama de blocos do DS18B20.

A ROM de 64 *bits* armazena o código serial único do sensor. O *Scratchpad* de memória contém os registradores de temperatura de 2 Bytes e armazena a saída digital do sensor de temperatura. Além disso, o *Scratchpad* fornece acesso a um *byte* de alarme para temperatura alta e baixa (*TH* e *TL*) e um registrador de 1 *byte* de configuração. A configuração deste registrador permite ao usuário ajustar a resolução da conversão analógico-digital da temperatura para 9, 10, 11 ou 12 *bits*. Os registradores configuráveis *T_H* e *T_L* possuem uma memória não volátil (EEPROM) e, dessa forma, mantém a configuração mesmo quando a alimentação do equipamento é desligada.

O DS18B20 usa o protocolo proprietário *1-Wire*TM que implementa a comunicação no barramento usando um sinal de controle. A linha de controle necessita de um pequeno resistor de *Pullup*, pois todos os equipamento estão conectados no barramento via 3- *State* ou porta *Open – drain* (Pino DQ). Neste sistema de barramento, o microprocessador (mestre) identifica e endereça os sensores usando cada código de 64 *bits* do sensor. Devido ao fato de cada um dos sensores possuírem um código único, o número de sensores que podem ser endereçados no barramento é virtualmente ilimitado.

3.5.2 Operação do DS18B20

O princípio de funcionamento do DS18B20 é o de um sensor com conversão de temperatura direta para digital. A resolução da temperatura do sensor é configurável pelo usuário de 9, 10, 11 ou 12 *bits*, o que corresponde a incrementos de 0,5 °C, 0,25 °C, 0,125 °C e 0,0625 °C, respectivamente. A resolução padrão na energização é 12 *bits*. O DS18B20 inicia em um estado de baixo consumo. Para iniciar a medição da temperatura e a conversão analógico/digital, o mestre precisa enviar um comando *Convert T*[44h]. Seguindo a conversão, o resultado é armazenado em um registrador de temperatura de 2 *bits* na memória *scratchpad* e o DS18B20 retorna ao estado repouso. Quando o DS18B20 é alimentado por uma fonte externa, o mestre pode enviar o comando “*read time slots*” após o comando *Convert T*, e o sensor irá responder enviando ‘0’, se a conversão de temperatura estiver ocorrendo e ‘1’, se estiver pronta.

A saída de temperatura é calibrada em graus centígrados. Para aplicações em outras unidades, uma tabela ou rotina de conversão deve ser utilizada. Os dados de temperatura são armazenados em um registrador de 16 *bits* com sinal. Os *bits* de sinal (S) indicam se a temperatura é positiva ou negativa: para números positivos, S=0 e para números negativos S=1. Se o sensor for configurado para 12 *bits* de resolução, todos os *bits* no registrador terão dados válidos. Para 11 *bits* de resolução, bit 0 é indefinido, para 10 *bits* de resolução, *bits* ‘0’ e ‘1’ são indefinidos, como mostra a Figura 3.10.

	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
LS Byte	2 ³	2 ²	2 ¹	2 ⁰	2 ⁻¹	2 ⁻²	2 ⁻³	2 ⁻⁴
	bit 15	bit 14	bit 13	bit 12	bit 11	bit 10	bit 9	bit 8
MS Byte	S	S	S	S	S	2 ⁶	2 ⁵	2 ⁴

Figura 3.10 - Formato do registrador de temperatura.

3.5.3 Código ROM de 64 Bits

Cada sensor contém um código único de 64 *bits* como mostra a Figura 3.11, gravados em ROM. Os primeiros menos significativos 8 *bits* da ROM contém o código da família *1-Wire*TM e os próximos 48 *bits* contém o número serial. Os 8 mais significativos *bits* contém a *cyclic redundant check (CRC) byte*, que é calculado a partir dos primeiros 56 *bits* do código da ROM. Os 64 *bits* de código ROM associados com as funções lógicas de controle permitem com que o sensor opere como um equipamento *1-Wire*TM.

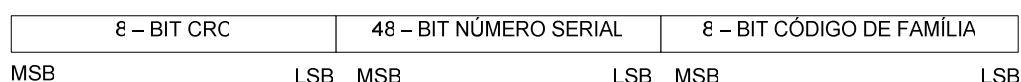


Figura 3.11 - 64 bits de código ROM.

3.5.4 Memória

A memória do sensor é organizada como mostra a Figura 3.12, e consiste de um SRAM (*scratchpad* com EEPROM não volátil) para os registradores de alarme alto e baixo (T_H e T_L) e registros de configuração. Se as funções de *alarm* não forem usadas, os registradores T_H e T_L podem ser usados como memória de uso geral. *Byte 0* e *byte 1* do *scratchpad* contém LSB e MSB dos registradores de temperatura, respectivamente. Estes *bytes* são somente leitura, os *bytes 2* e *3* fornecem o acesso para os registradores T_H e T_L . O *byte 4* contém a configuração do registrador de dados e os *bytes 5, 6* e *7* são reservados para uso interno do sensor e não podem ser sobrescritos, e retornando ‘1’ sempre que forem lidos. E, finalmente, o *byte 8* do *scratchpad* é somente leitura e contém a (CRC) para os *bytes 0* ao *7* do *scratchpad*.

Dados são escritos nos *bytes 2, 3* e *4* do *scratchpad* usando o comando *Write Scratchpad* [4Eh]. O dado deve ser transmitido começando com o bit menos significativo do *byte 2*. Para verificar a integridade dos dados, o *scratchpad* pode ser lido usando o comando *Read Scratchpad* [BEh] após o dado ser escrito. Quando se estiver lendo o *Scratchpad*, o dado é transferido pelo barramento *1-Wire™* começando com o bit menos significativo do *byte 0*. Para transferir T_H e T_L e os dados de configuração do *scratchpad* para a EEPROM, o mestre deve enviar o comando *copy scratchpad* [48h].

Os dados nos registradores da EEPROM são retidos quando a alimentação é desligada. Após a alimentação ser religada, os dados da EEPROM são recarregados em suas correspondentes locações no *scratchpad*. Os dados também podem ser recarregados da EEPROM para o *scratchpad* a qualquer momento, a partir do comando *Recall E²* [B8h].

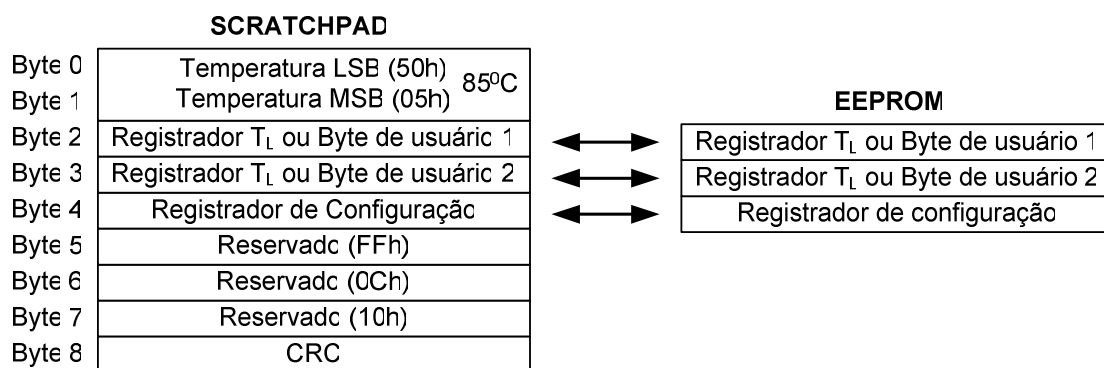


Figura 3.12 - Mapa de memória do sensor de temperatura DS18B20.

3.5.5 Calibração dos sensores de temperatura

Os sensores de temperatura foram calibrados com o auxílio de um banho térmico tipo Seco, Modelo T 25N da Presys – Instrumentos e Sistemas, como mostra a Figura 3.13.



Figura 3.13 - Banho térmico utilizado para aferição dos sensores.

O calibrador produz valores de temperatura no bloco de prova de forma a possibilitar a calibração de termopares, termoresistências, entre outros. O calibrador possui como principais características:

- o Gera temperaturas de – 25°C até 125°C;
- o Dispensa o uso de termômetro padrão externo;
- o Realiza calibrações totalmente automáticas, com ou sem o uso do computador;
- o Exatidão de 0,1°C, estabilidade de 0,02°C e resolução de 0,01°C.

Para a realização do procedimento de calibração, os sensores de temperatura DS18B20 foram inseridos no banho térmico, um por vez, sendo considerada a faixa de temperatura de 50^o C até 100^o C.

O procedimento de aferição iniciou-se colocando os sensores no banho térmico um a um, sendo a temperatura inicial aplicada no banho de 50^oC. A temperatura foi lida no sistema de supervisão desenvolvido e para sua estabilização nos sensores aguardou-se um período igual a 10 minutos. Em seguida, iniciou-se um procedimento de aumento gradual de temperatura em passos de 5 °C. Em cada passo, foi aguardado um período de tempo para estabilização da temperatura interna. Após a verificação da estabilização, a partir da indicação mostrada no banho térmico, foi feita uma nova leitura em cada sensor. A temperatura foi incrementada em passos de 5^oC, até a temperatura final de aferição de 100^oC.

Todo o procedimento foi repetido três vezes, sendo obtidos três conjuntos de leituras para verificar a sua repetibilidade e precisão. Com os valores obtidos para cada sensor, calcularam-se os coeficientes de reta de regressão linear para cada conjunto de leituras. Depois, para cada sensor, foi feita uma média dos coeficientes encontrados nos três ensaios e o resultado (um coeficiente linear e um coeficiente angular) foi introduzido no *software* a fim de apresentar as leituras corrigidas. A Tabela 3.2 mostra os valores obtidos com as leituras, juntamente com a temperatura do banho térmico para o sensor 1. A Figura 3.14 mostra os dados medidos e a reta de calibração (função de primeiro grau) em forma gráfica. A reta apresentada é a calculada por meio da regressão linear que gerou os coeficientes de correção aplicados ao *software*.

Tabela 3.2 - Valores obtidos na calibração do sensor 1.

Temp. Indicada no Sistema(^o C)	Temp. Real(^o C)
49,84	50
54,83	55
59,96	60
65,02	65
70,18	70
75,31	75
80,4	80
85,56	85
90,62	90
95,68	95
100,96	100

Seguindo o mesmo procedimento anterior, os coeficientes obtidos para as retas de calibração dos demais sensores estão representados na Tabela 3.3.

Tabela 3.3 - Retas de calibração para os sensores de temperatura.

Sensor	Reta de Calibração
1	$Y = 1,34 + 0,98 x$
2	$Y = 1,29 + 0,98 x$
3	$Y = 1,25 + 0,97 x$
4	$Y = 1,18 + 0,98 x$
5	$Y = 1,42 + 0,97 x$
6	$Y = 0,97 + 0,99 x$
7	$Y = 0,89 + 0,99 x$
8	$Y = 1,23 + 0,98 x$
9	$Y = 1,26 + 0,98 x$

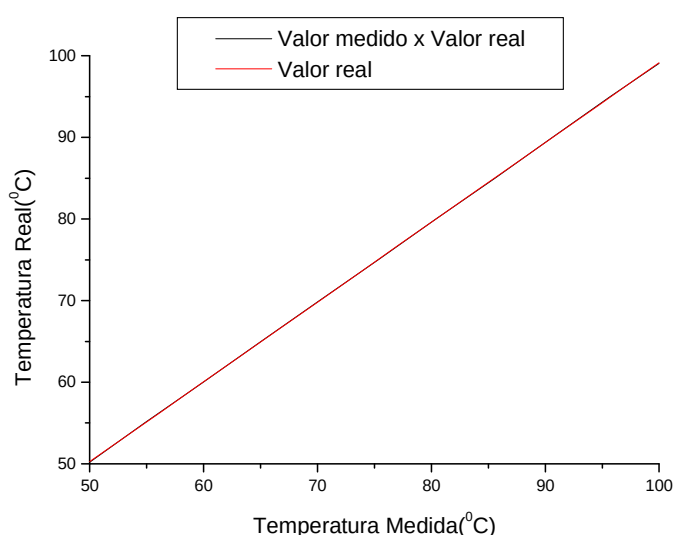


Figura 3.14 - Gráfico representativo da temperatura real medida e reta de calibração.

3.5.6 Sistema do barramento 1- Wire

O sistema *1-WireTM* usa um barramento mestre simples para controlar um ou mais equipamento escravos. O DS18B20 é sempre escravo. Quando há somente um escravo no barramento, o sistema é chamado *single-drop* e é chamado *multidrop*, se existe mais de um escravo no barramento.

Dentro do trabalho desenvolvido, o microcontrolador PIC 16F877A é o mestre, com os sensores de temperatura conectados às suas portas de entrada com escravos. Todos os dados têm a transmissão sendo iniciada pelo bit menos significativo no barramento 1-

Wire™. A descrição do barramento *1-Wire™* é dividida em três etapas: Configuração do *hardware*, sequência de comunicação e tipos e tempos dos sinais *1-Wire™*.

3.5.6.1 Configuração do *hardware*

O barramento *1-Wire™* tem por definição somente uma linha de dados. Cada equipamento (mestre ou escravo) interfaceia com a linha de dados via *open drain* ou porta 3 *State*. Isto permite que cada equipamento libere a linha de dados quando o equipamento não estiver transmitindo dados, permitindo que o barramento fique disponível para o uso de outro equipamento. A porta *1-Wire* do DS 18B20 (pino DQ) é *open drain* com um circuito interno equivalente mostrado na Figura 3.15. O barramento *1-Wire* necessita de um resistor de *pullup* de aproximadamente 5 k Ω . O estado repouso para o barramento *1-Wire* é nível alto. Se o barramento ficar em nível baixo por mais de 480 μ S todos os componentes do barramento serão reiniciados.

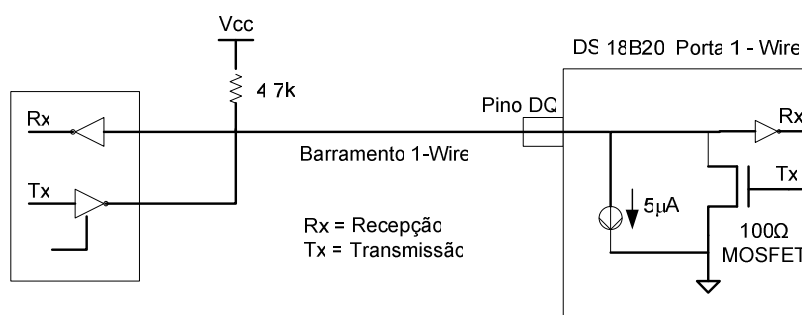


Figura 3.15 - Configuração do *hardware*.

3.5.6.2 Sequência da comunicação

A sequência da transação para acessar o DS18B20 é descrita da seguinte maneira:

- ✓ Inicialização;
- ✓ Comando ROM (seguido por qualquer troca de dados);
- ✓ Função de Comando DS18B20 (seguido por qualquer troca de dados solicitada);

É muito importante que esta sequência seja utilizada a todo o momento que o sensor for acessado, pois o mesmo não responderá se algum dos passos nesta sequência estiver faltando ou fora de ordem. Exceções para esta regra são os comandos *Search Rom* [F0h] e *Alarm Search* [ECh]. Depois de fazer estes comandos, o mestre deve retornar para o passo 1 na sequência.

3.5.6.2.1 Inicialização

Todas as transações no barramento 1-Wire iniciam com uma sequência de inicialização. A sequência de inicialização consiste de um pulso de *reset* transmitido pelo mestre do barramento seguido por pulso(s) de presença transmitido pelo escravo(s). O pulso de presença faz com que o mestre saiba que os escravos estão no barramento e prontos para operar.

3.5.6.2.2 Comandos ROM

Depois que o mestre detectou o pulso de presença, ele pode enviar um comando ROM. Estes comandos operam nos códigos de 64 *bits* de cada escravo e permitem ao mestre especificar um sensor dentre os diversos contidos no barramento. Estes comandos também permitem ao mestre determinar quantos e quais tipos de dispositivos estão presentes no barramento ou se algum possui uma condição de alarme. Existem 5 comandos ROM e cada comando é de 8 *bits* de comprimento. O mestre deve enviar um comando ROM apropriado antes de enviar um comando de função.

3.5.6.2.2.1 *Search* ROM [F0h]

Quando um sistema é ligado, o mestre deve identificar os códigos ROM de todos os escravos no barramento, o que lhe permite determinar o número de escravos e o tipo de dispositivos. O mestre aprende os códigos ROM por meio de um processo de eliminação que requer ao mestre promover um ciclo de *Search* ROM (isto é, comando *Search* ROM seguindo por troca de dados) quantas vezes sejam necessárias para identificar todos os dispositivos escravos. Se houver somente um dispositivo no barramento pode-se usar simplesmente o comando *Read* ROM no lugar do *Search* ROM.

3.5.6.2.2.2 *Read* ROM[33h]

Este comando somente pode ser usado quando há somente um escravo no barramento. Ele permite ao mestre do barramento ler o código de 64 *bits* do escravo sem usar a procedimento *Search* ROM. Se este comando for usado quando houver mais de um

escravo presente no barramento, uma colisão de dados pode ocorrer quando todos os escravos tentarem responder ao mesmo tempo.

3.5.6.2.2.3 *Match* ROM

O comando *Match* ROM, seguido pela sequência de 64 *bits* ROM, permite ao mestre endereçar um escravo específico em um barramento *multidrop* ou *single drop*. Somente o escravo que possuir a sequência de código de 64 *bits* irá responder ao comando de função enviado pelo mestre. Todos os outros escravos do barramento irão esperar até um pulso de *reset*.

3.5.6.2.2.4 *Skip* ROM [CCh]

O mestre pode usar este comando para endereçar todos os dispositivos no barramento sem enviar nenhuma informação de código ROM. Por exemplo, o mestre pode fazer com que todos os DS18B20 no barramento façam a conversão de temperatura simultânea enviando o comando *Skip* ROM, seguido do comando *Convert T* [44h]. O comando *Read ScratchPad* [BEh] pode seguir o comando *Skip* ROM somente se houver um único dispositivo escravo no barramento. Neste caso, o tempo gasto será reduzido, permitindo o mestre ler do escravo sem enviar o código de 64 *bits* da ROM. Um comando *Skip* ROM seguido por um comando *Read Scratchpad* irá causar uma colisão de dados no barramento se existirem mais de um escravo, pois eles tentarão transmitir dados simultaneamente.

3.5.6.3 Comandos de função do DS18B20

Depois que o mestre do barramento usou um comando ROM para endereçar o DS18B20 com o qual iria se comunicar, o mestre pode enviar um dos comandos de função. Estes comandos permitem ao mestre escrever e ler da memória *scratchpad* do escravo, iniciar conversões de temperatura e determinar o modo de fornecimento de energia.

3.5.6.3.1 Convert T [44h]

Este comando inicia a conversão da temperatura. Depois desta conversão, o resultado é armazenado no registrador de temperatura de 2 *bytes* na memória *scratchpad* e o dispositivo retorna à condição de baixo consumo no estado repouso. Com o dispositivo alimentado por uma fonte externa, o mestre pode enviar a leitura da temperatura após a conversão e o dispositivo irá responder transmitindo um ‘0’, enquanto a conversão da temperatura estiver ocorrendo e um ‘1’, quando a conversão for completada.

3.5.6.3.2 Read scratchPad [BEh]

Este comando permite ao mestre ler o conteúdo do *scratchpad*. A transferência de dados inicia com o menos significativo *bit* do *byte* 0 e continua até o nono *byte* (CRC) ser lido. O mestre pode enviar um *reset* para terminar a leitura a qualquer tempo se somente parte do *scratchpad* precisa ser lida.

3.6 Protocolo 1 – Wire

O DS18B20 usa o protocolo de comunicação 1-WireTM para assegurar a integridade dos dados. Alguns sinais são definidos por este protocolo: pulso de *reset*, pulso de presença, *write* 0, *write* 1, *read* 0 e *read* 1. O mestre inicia todos estes sinais com exceção do pulso de presença.

3.6.1 Procedimento de inicialização: pulso de presença e *reset*

O sensor de temperatura irá comunicar-se com o PIC por intermédio de uma conexão 1-WireTM, sendo o PIC definido como mestre, neste caso e o sensor, como escravo. Como toda a comunicação é efetuada com um único cabo e não é utilizado nenhum *clock* para sincronização, é necessário um tipo de comunicação assíncrona para enviar os dados, chamada de *Time Slots*.

Todas as comunicações com DS18B20 iniciam com uma sequência que consiste de um pulso de *reset* do mestre seguido por um pulso de presença do sensor. Quando o sensor

enviar o pulso de presença em resposta ao *reset*, ele indica ao mestre que está no barramento e pronto para operar.

Durante a sequência de inicialização, o mestre transmite (TX) o pulso de *reset* colocando o barramento 1-Wire em nível baixo no mínimo de 480 μ S. O mestre do barramento então libera o barramento e entra em modo de recepção (RX). Quando o barramento é liberado, o resistor de 5 k Ω *pullup* coloca o barramento em nível alto. Quando o sensor detecta a borda de subida, ele espera de 15 μ s a 60 μ s e então transmite um pulso de presença colocando o barramento 1-Wire em nível baixo de 60 μ s a 240 μ s, como mostra a Figura 3.16.

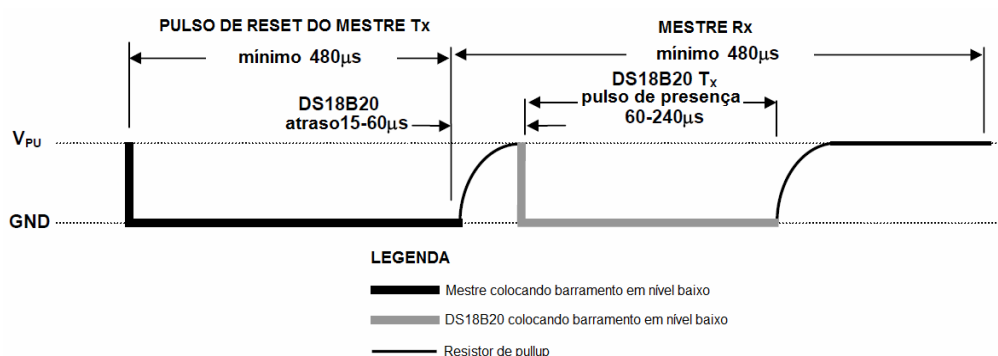


Figura 3.16 - Sequência de inicialização.

De uma forma geral, pode-se resumir as operações de inicialização com o diagrama de blocos ilustrado na Figura 3.17.



Figura 3.17 - Inicialização do barramento 1-Wire.

A fim de ler o valor de um equipamento conectado como escravo, ou transmitir uma instrução do mestre, o mestre e o escravo devem manipular a linha DQ em intervalos definidos de tempo para definir se o dado trocado é ‘1’ ou ‘0’.

Todas as instruções têm 8 *bits*. Assim, para transmiti-las no barramento 1-WireTM, o mestre deve sincronizar todos 8 *bits* de instrução, sendo 1 por vez (bit menos significativo primeiramente) e enviar um ‘1’ ou ‘0’, de acordo com o escravo. Assim os dados são transmitidos para o escravo de acordo com as seguintes regras de temporização:

Um ‘1’ é transmitido colocando a linha DQ em nível baixo por menos de 15 μ s. Após este tempo, libera-se o barramento, fazendo com que o resistor de *pullup* coloque o barramento em nível alto. Para escrever ‘0’ no barramento, após colocar o barramento 1-Wire em nível baixo, o mestre deve continuar mantendo o barramento 1-Wire em nível baixo por um tempo de pelo menos 60 μ s.

O DS18B20 faz a amostragem do barramento 1-Wire durante um tempo que dura de 15 μ s a 60 μ s após o mestre ter iniciado a escrita do dado. Se o barramento estiver alto durante a amostragem, um ‘1’ é escrito para o sensor e se a linha estiver em nível baixo um ‘0’ é escrito no sensor.

3.6.2 Leitura de dados

O sensor pode transmitir somente dados para o mestre quando este inicia a leitura de dados. Assim, o mestre pode fazer a leitura imediatamente após enviar o comando *Read Scratchpad* [BEh]. Dessa forma, o sensor pode enviar o dado solicitado. O mestre também irá fazer a leitura após o envio dos comandos *Convert T*[44h] ou *Recall E*²[B8h].

Todos os dados de leitura devem ter o mínimo de 60 μ s de duração com um mínimo de 1 μ s de tempo de recuperação entre *slots*. Um *slot* de leitura é iniciado pelo mestre colocando o barramento em nível baixo por um tempo mínimo de 1 μ s e, então liberando o barramento. Após, o mestre inicia a leitura do *slot* de tempo e o sensor irá iniciar transmitindo um ‘1’ ou um ‘0’ no barramento. O sensor transmite um ‘1’, colocando o barramento em nível alto ou um ‘0’, colocando o barramento em nível baixo. Quando transmite um ‘0’, o sensor irá liberar o barramento no final do *slot* de tempo e o barramento será colocado de volta em seu estado alto em repouso pelo resistor de *pullup*. Dados de saída do sensor são válidos para 15 μ s após a borda de descida que iniciou o *slot* de tempo de leitura. Então, o mestre deve liberar o barramento e amostrá-lo dentro de 15 μ s a partir do início do *slot*. A Figura 3.18 ilustra esta operação.

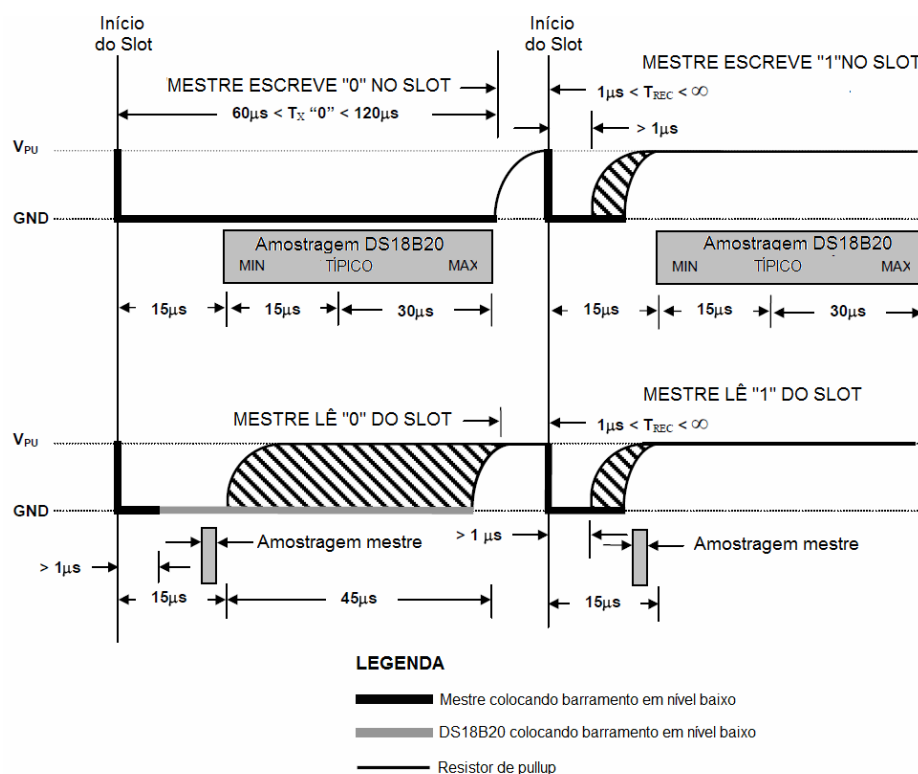


Figura 3.18 - Leitura de dados no barramento 1-Wire.

Os diagramas de bloco das Figuras 3.19 e 3.20 mostram como ocorrem as operações de leitura e escrita no barramento 1-Wire:

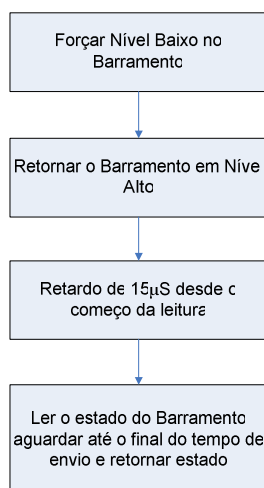


Figura 3.19 - Fluxo de leitura no barramento.

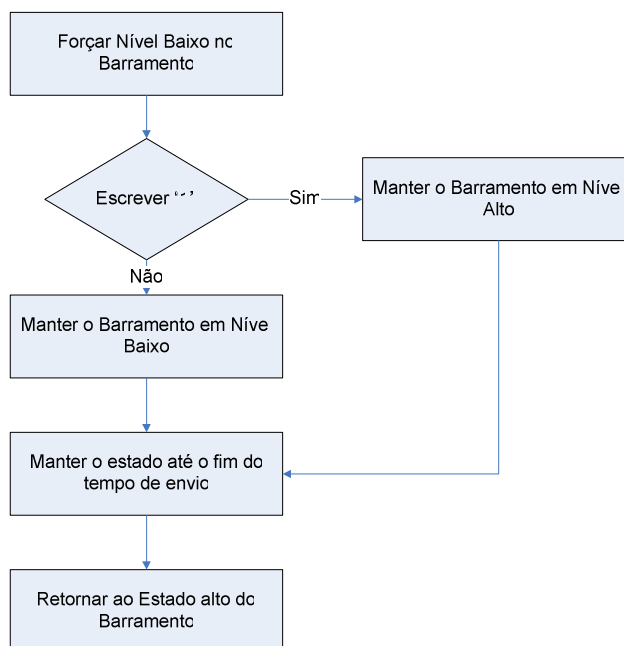


Figura 3.20 - Fluxo de escrita no barramento.

A sequência de eventos é mostrada abaixo:

- ✓ O barramento 1-Wire é inicializado;
- ✓ Envia o comando *Skip Rom* [CCh].
- ✓ Envia o comando *Convert Command* [44h].
- ✓ O barramento 1-Wire é inicializado novamente;
- ✓ Envia o comando *Skip Rom* [CCh] novamente;
- ✓ Envia o comando *Read Scratch Pad* [BEh].

3.7 Placa de Controle de Potência

A placa de controle de potência proposta tem a função de controlar a tensão em corrente alternada que alimentará a resistência do refervedor. Para um melhor entendimento do circuito da placa de controle, este foi dividido em duas partes, cada uma responsável por funções específicas: a leitura do sinal da tensão de entrada em forma senoidal (*zero crossing*), e o controle de tensão aplicada à resistência. A Figura 3.21 apresenta um diagrama de blocos com a representação das partes integrantes da placa de controle de potência.

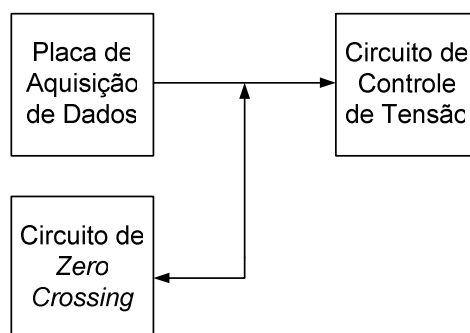


Figura 3.21 - Diagrama de blocos placa de controle de potência.

Na Figura 3.22 é representado o circuito elétrico relativo a placa de controle de potência do refeedor da coluna de destilação.

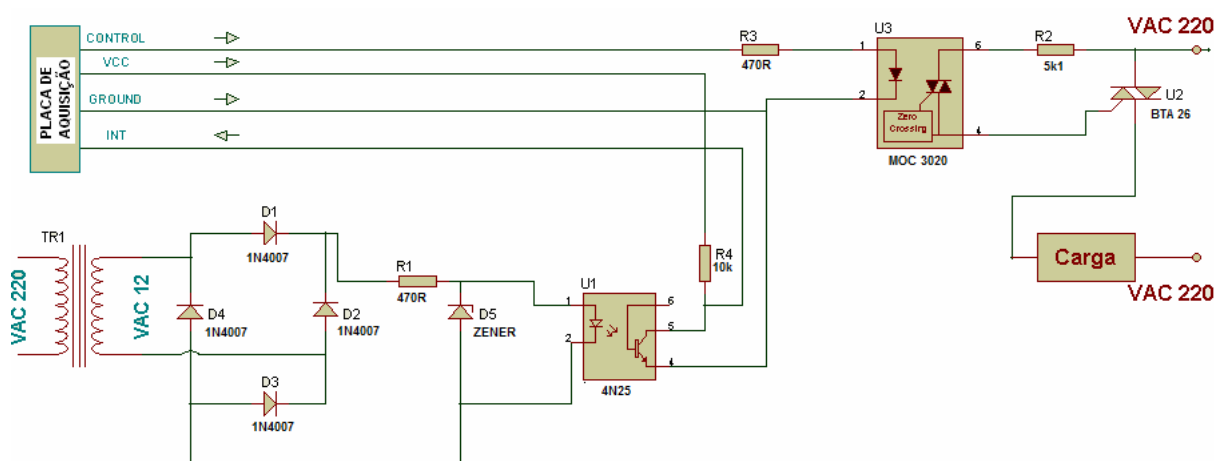


Figura 3.22 - Circuito placa de controle de potência desenvolvida.

3.7.1 Circuito de zero crossing

De forma a tornar possível o controle da potência em corrente alternada, é necessário efetuar a leitura do sinal de tensão em corrente alternada que será utilizado para alimentação da resistência elétrica. Sendo este circuito nomeado de *zero crossing*, conforme representado na Figura 3.22.

A finalidade do circuito de *zero crossing*, é identificar o instante de cruzamento do sinal de potência alternada senoidal em um valor igual a zero, sendo este sinal enviado ao microcontrolador, a cada ocorrência deste cruzamento, considerando a frequência da rede de alimentação de 60 Hz (período de 16,6 ms). Neste circuito, o sinal é retificado em onda completa, resultando na frequência de oscilação do sinal de 120 Hz (período de 8,33 ms). Desta forma, o circuito enviará um pulso à placa de aquisição a cada 8,33 ms, informando

que houve um cruzamento do sinal da origem por zero. A Figura 3.23 apresenta o circuito de *zero crossing*.

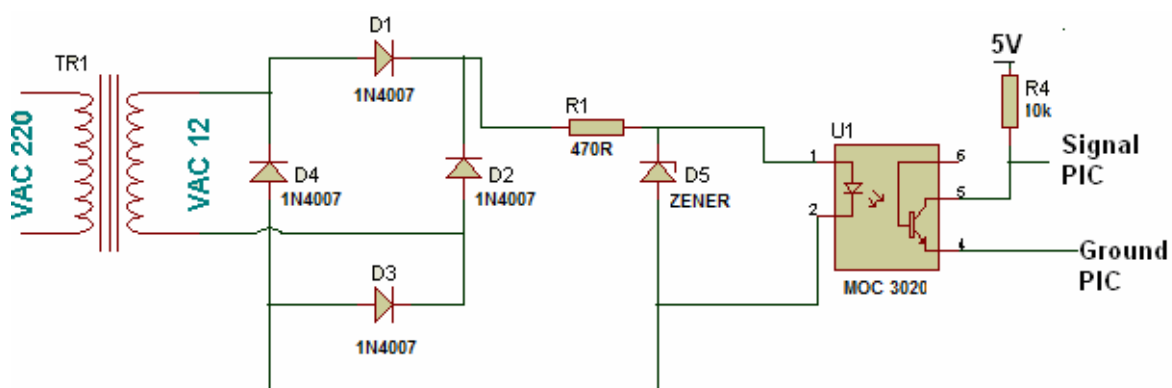


Figura 3.23 - Circuito de *zero crossing*.

O diagrama de blocos representado na Figura 3.24 representa o funcionamento do circuito de *zero crossing*.

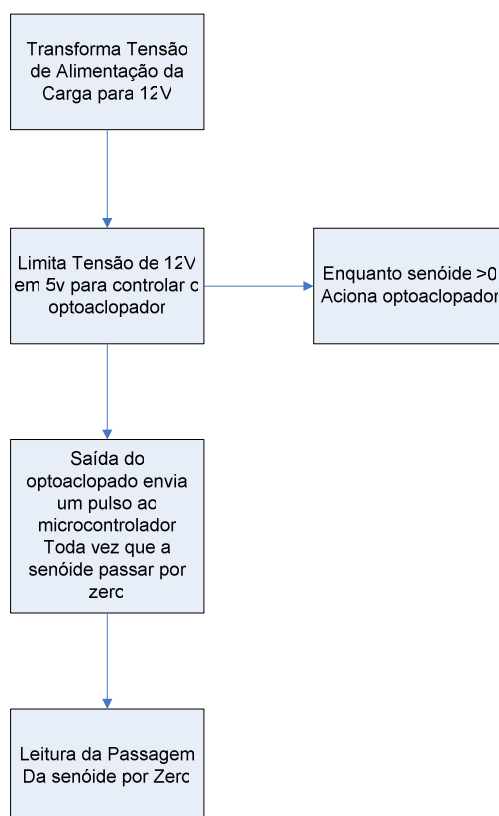


Figura 3.24 - Fluxograma do circuito *zero crossing*.

3.7.2 Circuito de controle de tensão

Na Figura 3.25, é apresentado o circuito responsável pelo controle da tensão aplicada na resistência do refervedor da coluna de destilação.

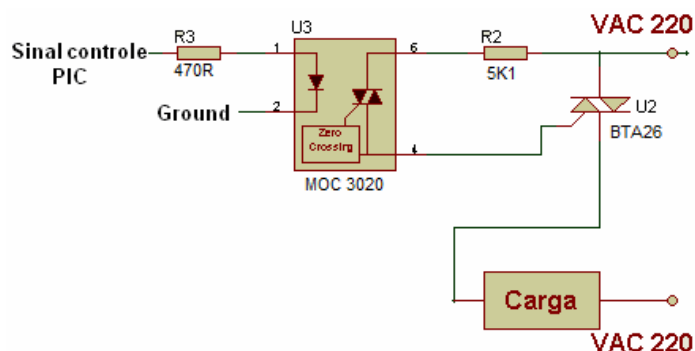


Figura 3.25 - Circuito de controle de tensão.

Este circuito tem por finalidade receber um sinal de controle oriundo do microcontrolador para efetuar o controle da variação de tensão em corrente alternada. Para efetuar este controle com eficácia, é necessária a utilização de um optotriac, para promover a isolamento entre o circuito de corrente contínua do microcontrolador e o de corrente alternada do controle de potência, possibilitando, assim, o chaveamento de um tiristor (Triac BTA 26) em corrente alternada.

Para elaboração da placa de potência, foi utilizado o Triac BTA26, devido ao fato do mesmo trabalhar com uma corrente de operação por volta de 26 A, possibilitando que esta placa de controle opere com uma margem de segurança para utilização na resistência do refulvedor, que possui em torno de 3000 watts de potência e, como o sistema é alimentado na tensão de 220 V, e irá gerar uma corrente de 13,23 A.

O diagrama de blocos, ilustrado na Figura 3.26 descreve o funcionamento do circuito de controle de tensão.

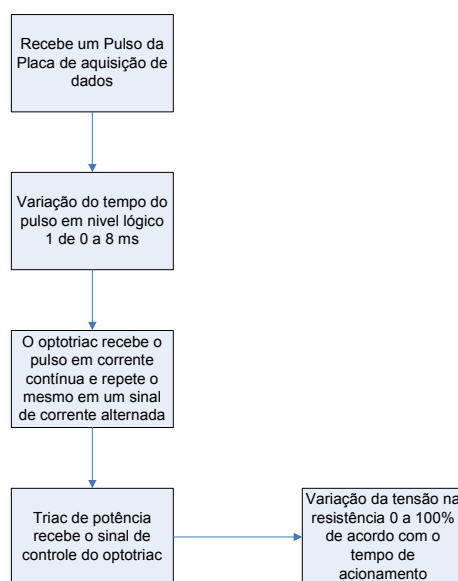


Figura 3.26 - Fluxograma do circuito de controle de tensão.

Para que a placa de aquisição de dados possa receber diversos parâmetros simultaneamente vindos do *software* de supervisão, foi necessário o desenvolvimento de um protocolo de comunicação serial enviado pelo *software* de supervisão com os valores dos parâmetros localizados neste protocolo, conforme a Tabela 3.4.

Tabela 3.4 - Protocolo de recepção de dados.

<i>String</i> de modo Automático (Parâmetros)	OXXHXXZXXF (Descrição)
O	Saída para a variável manipulada
A/H	Valor que indica controle auto/manual
F	Fim de transmissão
Z	Valor de controle manual

O funcionamento do protocolo de comunicação é detalhado no diagrama de blocos apresentado na Figura 3.27.

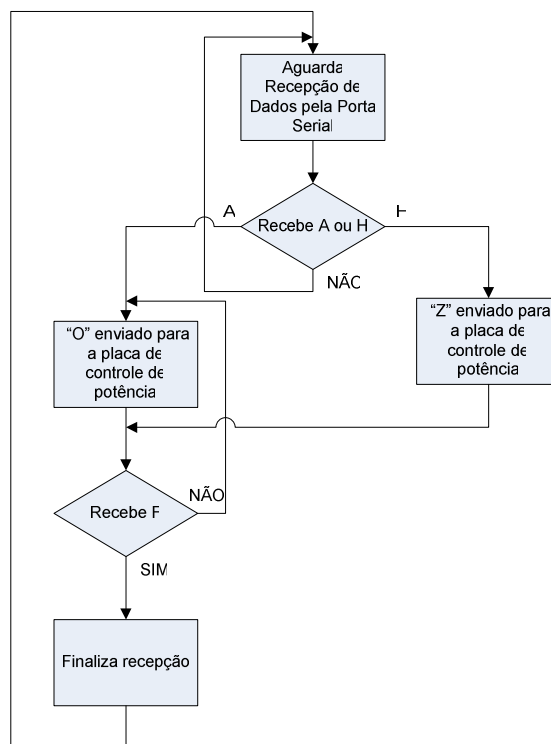


Figura 3.27 - Fluxograma do protocolo de comunicação.

3.8 *Software* de Supervisão

Para a supervisão do sistema, foi desenvolvido um *software* em linguagem orientada a objetos para a monitoração das temperaturas. O ambiente de desenvolvimento escolhido para a realização do trabalho foi o Delphi, pois se constitui em uma ótima

solução para desenvolvimento de projetos orientados a objetos e devido ao seu ambiente gráfico e à facilidade de se criar os objetos do código. Alia-se a isso, o fato da facilidade de integração do programa a bancos de dados diversos.

No *software* de supervisão, foram desenvolvidas diversas funcionalidades similares às empregadas em *softwares* de supervisão usados em nível industrial, como: controle com níveis de acesso a usuários; gráficos de temperatura em tempo real; armazenamento de temperaturas do processo; bem como dados relativos ao controle do processo. Também possibilita o registro de acesso de todos os usuários em um banco de dados, com possibilidade de apresentação destes dados em uma forma de relatórios. Para uma melhor compreensão da estrutura do *software* de supervisão desenvolvido, serão demonstradas as principais telas desenvolvidas e funcionalidades do *software* de supervisão. Na Figura 3.28, é apresentada a tela inicial do *software*, na qual o usuário irá efetuar o *login* e senha. Desta forma, de acordo com o nível de acesso deste usuário, o *software* irá habilitar telas específicas a este determinado usuário, fazendo com que um usuário comum não tenha privilégios de configuração e operação de um administrador de sistema, assegurando, assim, a segurança necessária ao processo a ser controlado.



Figura 3.28 - Tela inicial do sistema de supervisão.

Após a entrada com usuário e senha, é mostrada a tela principal do sistema de supervisão, onde está representada a coluna de destilação com os seus respectivos pratos, bem como as demais funcionalidades do *software*. De acordo com o nível de acesso de usuário, nesta tela é possível: cadastrar usuários, visualizar relatórios e gráficos de temperatura em tempo real e, principalmente, efetuar o controle manual ou automático do processo. A Figura 3.29 mostra a tela de supervisão, sendo a tela de cadastro de usuário apresentada na Figura 3.30.

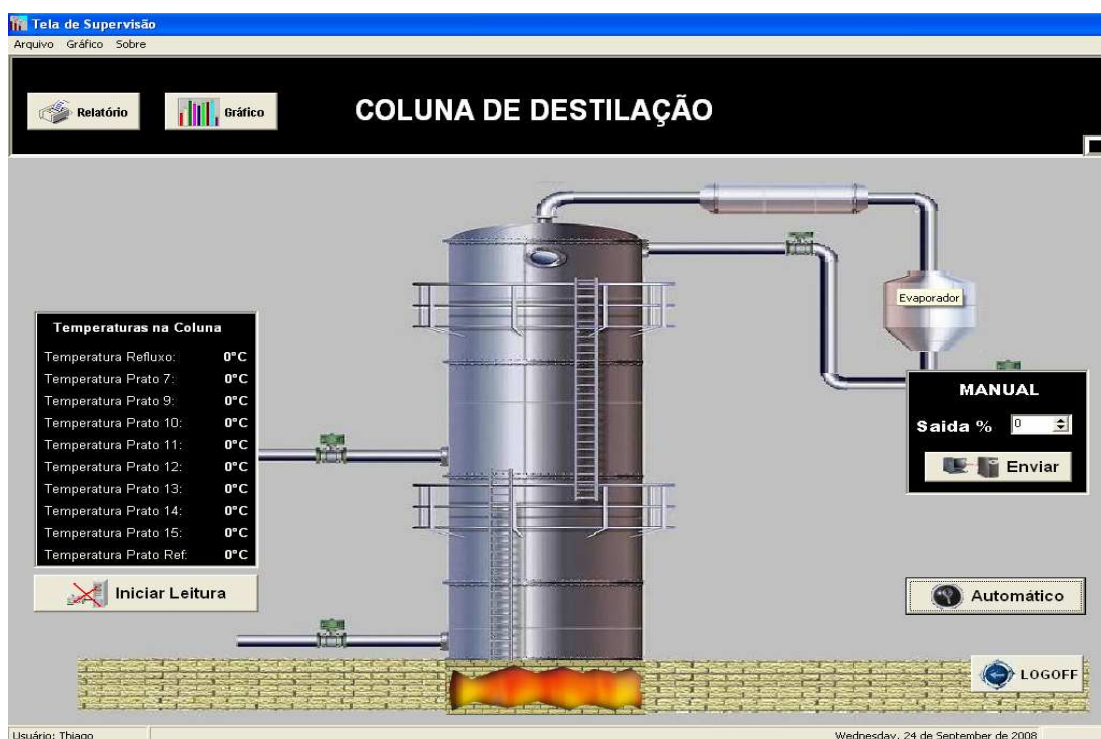


Figura 3.29 - Tela principal do sistema de supervisão.

A imagem mostra a interface gráfica do sistema de cadastro, intitulada "Tela de Cadastro".

À esquerda, há uma seção "Usuários Cadastrados" com uma lista de usuários: "Thiago", "Claiton" e "Operador". O usuário "Thiago" está selecionado.

À direita, há campos para "Usuário" (contendo "Thiago"), "Senha" (contendo "*") e "Nível de Acesso" (contendo "0").

Na barra de ferramentas inferior, há quatro botões: "Gravar", "Excluir", "Novo" e "Sair".

Figura 3.30 - Cadastro de usuários.

O monitoramento do perfil de temperaturas é feito *on-line* na tela principal, sendo visualizadas as temperaturas de todos os sensores instalados na coluna, como mostrado na Figura 3.31. Para tanto, os dados do processo foram levantados com uma taxa de aquisição de 9 segundos entre as medidas. Este tempo de aquisição consegue representar a variação da temperatura com grande fidelidade, visto que a dinâmica de um processo de destilação é bastante lenta, sendo todos os dados coletados enviados a um banco de dados Access para armazenamento e posterior análise.

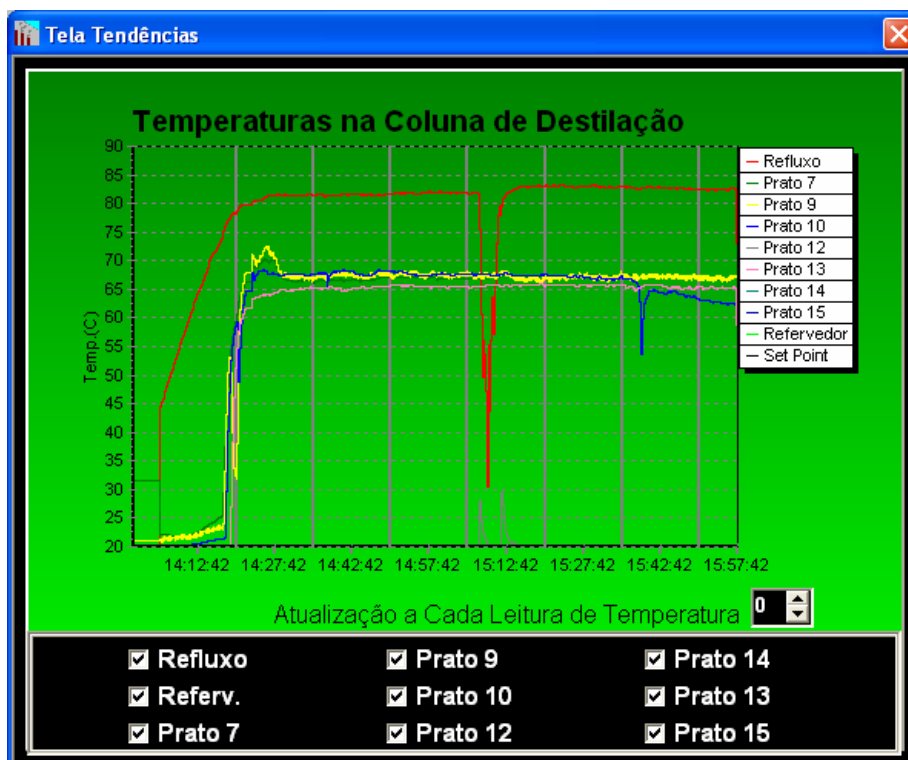


Figura 3.31 - Gráfico representando a tendência de temperatura nos pratos.

Na opção arquivo => Abrir tem-se a visualização dos relatórios do perfil de temperaturas. Estes podem ser visualizados e arquivados nos formatos. .rtf, .txt, .html, .doc, e .pdf. Cada relatório possui informações, como data e temperatura em cada um dos pratos da coluna nos respectivos tempos. As Figuras 3.32 e 3.33 mostram a representação da opção de saída e formato dos relatórios, respectivamente.

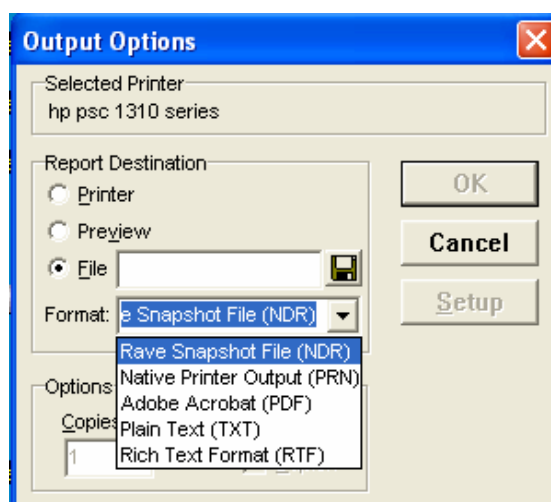


Figura 3.32 – Opção de saída de relatórios.

Report Preview											
File Page Zoom											
Page 1 of 15 Zoom 100.0 %											
Temperaturas Coluna de Destilação(C)											
Hora	Refluxo	Prato 7	Prato 9	Prato 10	Prato 12	Prato 13	Prato 14	Refer.	Alim.	Manual	Usuário
14:00:26	20,25	31,56	21	19,87	19,62	19,62	0	0	0		Thiago
14:05:16	44,56	22,25	21,43	20,06	19,75	19,75	0	0	0	0	Thiago
14:05:25	44,87	22,25	21,43	20,06	19,68	19,75	0	0	0	0	Thiago
14:05:34	45,25	22,18	20,93	20,06	19,75	19,75	0	0	0	0	Thiago
14:05:42	45,62	22,12	21,62	20,06	19,75	19,75	0	0	0	0	Thiago
14:05:51	45,87	22,12	21	20,12	19,68	19,75	0	0	0	0	Thiago
14:05:59	46,25	22,12	21,81	20,12	19,68	19,75	0	0	0	0	Thiago
14:06:08	46,62	22,06	21,18	20,18	19,68	19,75	0	0	0	0	Thiago
14:06:16	47,06	22,12	21,75	20,18	19,68	19,81	0	0	0	0	Thiago
14:06:25	47,43	22,12	21,43	20,18	19,68	19,81	0	0	0	0	Thiago
14:06:33	47,93	22,12	21,62	20,18	19,68	19,81	0	0	0	0	Thiago
14:06:42	48,43	22,06	21,68	20,18	19,68	19,81	0	0	0	0	Thiago
14:06:50	48,87	22,06	21,18	20,25	19,68	19,81	0	0	0	0	Thiago
14:06:59	49,18	22,06	21,81	20,25	19,68	19,81	0	0	0	0	Thiago
14:07:07	49,56	22	21,12	20,18	19,68	19,81	0	0	0	0	Thiago
14:07:16	49,93	21,93	21,93	20,18	19,68	19,81	0	0	0	0	Thiago
14:07:24	50,31	21,93	21,25	20,18	19,68	19,81	0	0	0	0	Thiago
14:07:33	50,68	21,87	21,75	20,18	19,68	19,81	0	0	0	0	Thiago
14:07:41	51,06	21,93	21,5	20,18	19,68	19,81	0	0	0	0	Thiago
14:07:50	51,5	21,93	21,56	20,18	19,68	19,81	0	0	0	0	Thiago
14:07:58	51,93	21,93	21,68	20,25	19,68	19,81	0	0	0	0	Thiago
14:08:07	52,37	21,93	21,18	20,25	19,62	19,81	0	0	0	0	Thiago
14:08:16	52,87	21,93	21,93	20,25	19,68	19,81	0	0	0	0	Thiago
14:08:24	53,37	21,93	21,25	20,31	19,68	19,81	0	0	0	0	Thiago
14:08:33	53,56	21,93	22	20,25	19,68	19,81	0	0	0	0	Thiago
14:08:41	54	21,87	21,37	20,25	19,68	19,81	0	0	0	0	Thiago
14:08:50	54,37	21,87	21,87	20,31	19,68	19,81	0	0	0	0	Thiago

Figura 3.33 - Relatório do perfil de temperaturas.

Dentro do sistema de relatórios, existe uma conexão com bancos de dados Dbase que permite exportar todos os dados para um programa com planilha, como, por exemplo, o Excel. O *software* de supervisão permite ao usuário visualizar os relatórios desejados,

como relatório dos parâmetros de controladores empregados no sistema, temperaturas na coluna e usuários. Este modelo de relatórios foi desenvolvido na interface gráfica de desenvolvimento de relatórios do Delphi chamada *Rave Reports*, sendo que esse gerador de relatórios traz uma interface independente de desenvolvimento de relatórios. Os relatórios podem ser gravados em arquivos à parte ou incorporados em seus sistemas de banco de dados.

O *software* de supervisão permite ao usuário visualizar os relatórios desejados do processo, bem como permite salvar o mesmo em diversos formatos, como pdf, txt, entre outros e, também, a opção de imprimir o relatório desejado.

Existe também a possibilidade do usuário optar por um controle manual da potência elétrica do refrervedor, sendo possível a variação deste valor de potência de 0 a 100% do valor de potência máxima, como é ilustrado na Figura 3.34.



Figura 3.34 - Controle manual.

3.9 Comparativo entre o Sistema Desenvolvido e um Sistema Convencional de Controle, Aquisição de Dados e Supervisão.

Para comprovar a viabilidade do desenvolvimento do sistema de controle, aquisição de dados e sistema de supervisão, é apresentada uma comparação de custos relativos ao sistema desenvolvido em comparação a um sistema de aquisição de dados convencional, desenvolvido com uma plataforma constituída de um CLP (Controlador Lógico programável) *Allen Bradley* e um sistema de Supervisão SCADA (*Supervisory Control And Data Acquisition*) RS VIEW SE STATION e sensores de temperatura do tipo termoresistência (PT100). A Tabela 3.5 apresenta uma planilha de custos relativa aos materiais empregados no sistema de aquisição, controle e supervisão de dados.

Tabela 3.5 - Descrição dos elementos utilizados para a confecção do sistema de aquisição, controle e supervisão de dados (Fonte: Eletrônica Projesol Ltda.).

Item	Quant.	Custo Unitário(R\$)	CustoTotal(R\$)
Regulador de Tensão 7805	1	2,50	2,50
Capacitor Cerâmico 33nF	2	0,20	0,40
Soquete DIP 20 Pinos	1	2,20	2,20
Soquete Dip 15 Pinos	1	1,5	1,5
Conector RJ 45 Macho	16	0,5	8
Capacitor Eletrolítico 0,22 μ F 24V	1	0,20	0,20
Capacitor Eletrolítico 1 μ F 24V	1	0,20	0,20
Microcontrolador PIC 16F877A	1	28,70	28,70
Conversor TTL/RS232	1	8,70	8,70
Conector Rj 45	16	0,5	8,00
Capacitor eletrolítico 10 μ F/6V	4	0,20	0,80
Resistor 4,7K Ω /1/4W	16	0,1	1,6
Conector DB9 Fêmea	1	2,00	2
Conector Pino Fêmea	1	2,00	2
Conversor Rs 232/ USB	1	74,90	74,90
Cabo Telefônico 4 x 0,5mm ²	30 m	0,20	6,00
Placa de Circuito impresso	1	5,00	5
Fonte de Alimentação 12Vcc	1	15,00	15,00
Sensor de Temperatura DS18B20	10	14,50	145,00
Licença Delphi 7.0	1	3270	3270
Cristal 4 MHz	1	2,00	2
Triac BTA 26	1	4,75	4,75
Total			3589,45

O sistema equivalente pode ser implementado com *hardware* Allen Bradley e foi desenvolvido com o auxílio do *software* RA_IAB (*Integrated Architecture Builder*), sendo apresentado na Figura 3.35.

Neste sistema, tem-se o CLP Micro Logix 1500: 1764-24AWA, juntamente com 3 cartões de leitura de sensores tipo RTD (termoresistência), cada um com 6 entradas. O CLP está conectado ao computador de supervisão utilizando o protocolo RS232. Os dados de temperatura são enviados ao cartão em níveis de resistência (Ω), sendo que os mesmos interpretam o sinal por meio do CLP e enviam os dados serialmente ao *Software* de Supervisão instalado no Microcomputador de Supervisão.

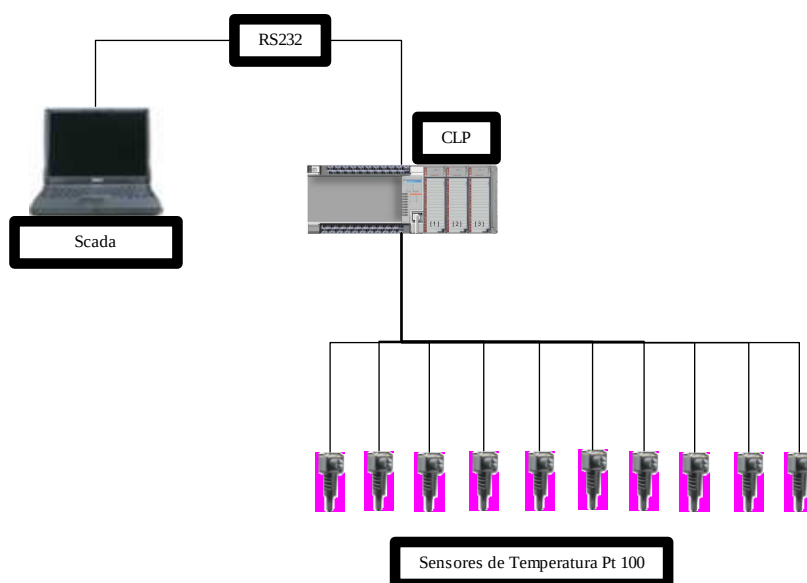


Figura 3.35 - Sistema típico utilizado para aquisição de temperaturas.

Os *softwares* utilizados para o sistema de aquisição e supervisão de dados são os seguintes:

- RsLogix: responsável pela programação da lógica implementada no CLP;
- RSLinx OEM: responsável pela conexão do CLP com o sistema de supervisão;
- RS View SE Station: *software* utilizado para confecção do sistema de supervisão.

Na Tabela 3.6, tem-se uma descrição do *hardware* e *softwares* de desenvolvimento utilizados e seu respectivo custo.

Tabela 3.6 - *Hardware* e *software* necessários para o sistema de controle e aquisição de dados (Fonte: Eletronor distribuidora de materiais elétricos Ltda).

Item	Quant.	Custo Unitário (R\$)	Custo Total (R\$)
Micro Logix 1500: 1764-24AWA	1	2.062,09	2.062,09
Modulo RTD1769-IR6	2	4.038,27	8.076,54
Terminador: 1769-ECR	1	126,43	126,43
Modulo 1769 –IF4I	1	3.498,19	3.498,19
Cabo de Comunicação RS232	1	219,00	219,00
Sensor de Temperatura Pt 100	10	320,00	3.200,00
RS View SE Station	1	11.343,47	11.343,47
RS View SE Studio	1	10.851,47	10.851,47
RsLogix 500 Professional	1	5.865,00	5.865,00
Rslinx OEM	1	3.860,00	3.860,00
Total			49102,19

CAPÍTULO 4 - OBTENÇÃO DO MODELO DA COLUNA DE DESTILAÇÃO POR MEIO DE TÉCNICAS DE IDENTIFICAÇÃO

A existência de um modelo de um processo químico é necessidade fundamental para tornar possível seu controle efetivo, bem como ajuste de parâmetros de sintonia dos controladores em um simulador antes da aplicação no processo.

Embora seja possível obter um modelo representativo fenomenológico, por meio de balanços de massa, energia e quantidade de movimento, estes modelos podem ser difíceis de obter e de implementar. Devido a esta dificuldade, é possível o desenvolvimento de um modelo por meio de um conjunto de entradas-saídas representativas de um processo físico. O procedimento da modelagem matemática é diretamente influenciado pelo conhecimento acerca do processo. Desta forma, se a finalidade é obter um modelo para projeto e simulação de sistemas de controle avançados, como, por exemplo, o controle preditivo/adaptativo, a melhor opção é a identificação de processos, pois fornece modelos relativamente simples e necessários para simulação e aplicação dos controladores.

De acordo com os objetivos deste trabalho, neste capítulo apresenta-se os resultados obtidos por meio da aplicação de algumas técnicas de identificação, que irão fornecer um modelo com erro percentual baixo em relação aos dados de entrada/saída, o que comprova todo o potencial da identificação de processos como ferramenta para modelagem.

O objetivo deste capítulo é fazer uma análise de dados da coluna de destilação e, a partir dos mesmos, obter um modelo para sua resposta. A obtenção do modelo do processo por meio de identificação será elaborada com o auxílio do *software* MATLABTM a partir da implementação de redes neurais artificiais e comparando-as com modelos clássicos de identificação: ARX, ARMAX, OE, BJ e Espaço de Estados.

4.1 Aspectos Inerentes à Modelagem do Sistema

Em um sistema de controle a planta apresenta não-linearidades que devem ser descritas precisamente a fim de se projetar um controlador que leve uma operação efetiva do processo.

Para se obter o modelo do processo, o projetista pode trabalhar de duas maneiras. A primeira utilizando conhecimentos de física, química e outras ciências para descrever o processo envolvido, na forma de um modelo matemático.

A segunda maneira é utilizar dados experimentais obtidos via perturbação. Este modelo é chamado de identificação de processos e é recomendado quando o processo envolvido é extremamente complexo e apresenta fortes não-linearidades.

Assim, será feita uma descrição do modelo fenomenológico da coluna de destilação para efeito de comparação e comprovação da eficácia e simplicidade relativa dos modelos obtidos via identificação de processos.

4.2 Modelagem Matemática de uma Coluna de Destilação Binária

A seguir será apresentada a modelagem de uma coluna de destilação para a separação da mistura binária etanol-água. Para tanto, serão feitas simplificações e idealizações para facilitar a compreensão dos fenômenos que ocorrem no interior da coluna.

Será suposto um sistema binário, com volatilidade relativa constante e pratos com eficiência de 100 %, isto é, o vapor que deixa o prato está em equilíbrio com o líquido no prato. Isto significa que uma relação de equilíbrio vapor-líquido mais simples pode ser usada:

$$y_n = \frac{\alpha_r x_n}{1 + (\alpha_r - 1)x_n} \quad (4.1)$$

Em que:

x_n = Concentração do líquido no n -ésimo prato (fração molar do componente mais volátil);

y_n = Concentração do vapor no n -ésimo prato (fração molar do componente mais volátil);

α_r = Volatilidade relativa.

Considera-se uma única corrente de alimentação na coluna, no estado de líquido saturado, no prato de alimentação N_F , como mostra a Figura 4.1.

A vazão de entrada é F (mol/min) e sua concentração é z (fração molar do componente mais volátil). O vapor de topo é totalmente condensado e escoar até um reservatório de refluxo cuja quantidade de líquido é M_D (moles). O conteúdo do

reservatório é suposto ser perfeitamente miscível, com concentração x_D . O líquido no reservatório encontra-se no seu ponto de bolha. O refluxo é bombeado de volta para o prato de topo da coluna, N_T , à vazão R . O produto de topo destilado é removido a uma taxa D .

Na base da coluna, o líquido é removido a uma taxa B e com uma concentração x_B . O vapor é gerado pelo aquecimento do refeedor através de um termo-sifão na taxa de V . O líquido circula a partir do fundo da coluna através da seção circular da coluna devido à menor densidade da mistura vapor-líquido no refeedor decorrente do processo de aquecimento promovido pelo refeedor. Supõe-se que o fluido no refeedor e na base da coluna tem a mesma concentração x_B e nível M_B (moles).

A concentração do vapor que deixa a base da coluna e no prato 1 é y_1 , e está em equilíbrio com o líquido com concentração x_B .

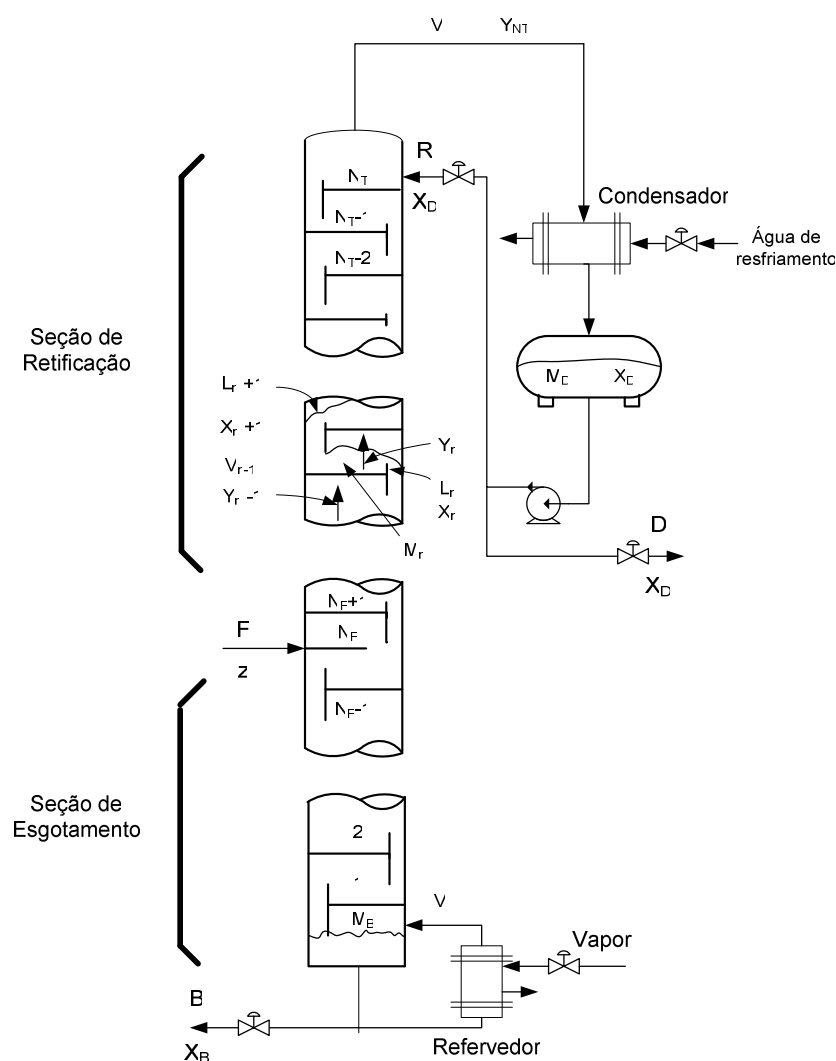


Figura 4.1 - Coluna de destilação binária.

A coluna possui um total de N_T pratos teóricos. O acúmulo de líquido em cada prato é M_n . O líquido em cada prato é considerado perfeitamente misturado, com a concentração x_n . O acúmulo de vapor suposto é desprezível no sistema. Embora o volume de vapor seja muito grande, o número de moles é geralmente pequeno, devido ao fato da densidade do vapor ser muito menor que a densidade do líquido.

Outra consideração a ser feita é a de fluxo equimolar. As variações de concentração devido às perdas de calor e a temperatura que mudam de prato a prato são consideradas desprezíveis. Com base nestas considerações, pode-se afirmar que as vazões mássicas de vapor e líquido através das seções de retificação e esgotamento serão constantes em condições de regime permanente.

Desta forma, a vazão mássica do vapor através de todos os pratos da coluna será a mesma, dinamicamente ou em regime estacionário.

$$V = V_1 = V_2 = V_3 = \dots = V_{NT} \quad (4.2)$$

As vazões mássicas de vapor não são constantes com o tempo, devido ao fato de que o calor aplicado ao refeedor pode ser manipulado dinamicamente, alterando-as. O efeito matemático de se assumir fluxo equimolar é que não é necessário implementar uma equação de energia para cada prato.

As vazões mássicas de escoamento do líquido através da coluna não serão as mesmas dinamicamente, pois irão depender da mecânica do fluido de cada prato. Considerando um prato com vertedouro, pode ser usada a relação de Francis. Segundo LUYBEN (1996), para o acúmulo de líquido no prato esta relação é utilizada para relacionar o acúmulo no prato (M) com a taxa de escoamento do fluido que está deixando o prato (L_w):

$$F_L = 3,33.L_w(h_{ow})^{1.5} \quad (4.3)$$

Sendo:

F_L - taxa de escoamento do líquido pelo vertedouro (ft^3/s).

L_w - líquido no vertedouro (ft).

h_{ow} - nível do líquido no vertedouro (ft).

Podem ser utilizadas relações mais rigorosas a partir do detalhamento das equações hidráulicas, incluindo efeitos da taxa de escoamento de vapor, densidades, composições,

etc. Para o estudo desenvolvido, será suposta uma simples relação funcional entre acúmulo de líquido e taxa de escoamento do líquido:

$$M_n = f(L_n) \quad (4.4)$$

Outra simplificação a ser considerada é desprezar a dinâmica do condensador e do refeedor, pois em colunas de escala comercial, a resposta dinâmica destes trocadores de calor é geralmente muito mais rápida que a resposta da coluna.

Com estas considerações, é possível modelar o sistema por meio do equacionamento matemático e utilizando as equações de continuidade.

Condensador e reservatório do refluxo:

Equação da continuidade total:

$$\frac{dM_D}{dt} = V - R - D \quad (4.5)$$

Equação da continuidade para o componente mais volátil:

$$\frac{d(M_D x_D)}{dt} = Vy_{NT} - (R + D)x_D \quad (4.6)$$

Prato do topo ($n=N_T$)

Equação da continuidade total:

$$\frac{dM_{NT}}{dt} = R - L_{NT} \quad (4.7)$$

Equação da continuidade para o componente mais volátil:

$$\frac{d(M_{NT} x_{NT})}{dt} = Rx_D - L_{NT} x_{NT} + Vy_{NT-1} - Vy_{NT} \quad (4.8)$$

Prato sequencial ao prato de topo ($n=N_T-1$)

Equação da continuidade total:

$$\frac{d(M_{NT-1} x_{NT-1})}{dt} = Rx_D - L_{NT} x_{NT} + Vy_{NT-1} - Vy_{NT} \quad (4.9)$$

Equação da continuidade para o componente mais volátil:

$$\frac{d(M_{NT-1} x_{NT-1})}{dt} = L_{NT} x_{NT} - L_{NT-1} x_{NT-1} + Vy_{NT-2} - Vy_{NT-1} \quad (4.10)$$

N-ésimo prato

Equação da continuidade total:

$$\frac{dM_n}{dt} = L_{n+1} - L_n \quad (4.11)$$

Equação da continuidade para o componente mais volátil:

$$\frac{d(M_n x_n)}{dt} = L_{n+1} x_{n+1} - L_n x_n + V y_{n-1} - V y_n \quad (4.12)$$

Prato de alimentação ($n = N_F$)

Equação da continuidade total:

$$\frac{dM_{NF}}{dt} = L_{NF+1} - L_{NF} + F \quad (4.13)$$

Equação da continuidade para o componente mais volátil:

$$\frac{d(M_{NF} x_{NF})}{dt} = L_{NF+1} x_{NF+1} - L_{NF} x_{NF} + V y_{NF-1} - V y_{NF} + F z \quad (4.14)$$

Primeiro prato ($n=1$)

Equação da continuidade total:

$$\frac{dM_1}{dt} = L_2 - L_1 \quad (4.15)$$

Equação da continuidade para o componente mais volátil:

$$\frac{d(M_1 x_1)}{dt} = L_2 x_2 - L_1 x_1 + V y_B - V y_1 \quad (4.16)$$

Refrervador e base da coluna

Equação da continuidade total:

$$\frac{dM_B}{dt} = L_1 - V - B \quad (4.17)$$

Equação da continuidade para o componente mais volátil:

$$\frac{d(M_B x_B)}{dt} = L_1 x_1 - V y_B - B x_B \quad (4.18)$$

Cada prato e a base da coluna têm equações de equilíbrio. Cada prato tem uma equação hidráulica. Desta forma, também é necessário representar os controladores para os níveis dos controladores na base da coluna e no refluxo.

$$D = f_{1(MD)} \quad B = f_{2(MB)} \quad (4.19)$$

Assim, é possível analisar os graus de liberdade do sistema. A vazão de alimentação F e a concentração z são conhecidas.

Número de variáveis:

Concentração dos pratos (x_n e y_n)	$= 2N_T$
Escoamento de líquido nos pratos (L_n)	$= N_T$
Nível de líquido nos pratos	$= N_T$
Concentração do reservatório de refluxo (X_D)	$= 1$
Vazão do reservatório de refluxo (R e D)	$= 2$
Nível do reservatório de refluxo	$= 1$
Composições da base da coluna (x_B e y_B)	$= 2$
Vazões da base (V e B)	$= 2$
Acúmulo da base (M_B)	$= 1$
Total de variáveis	$= 4.N_T+9$

Número de equações:

		Equação
Continuidade dos componentes nos pratos	$= N_T$	(4.8)
Continuidade total dos pratos	$= N_T$	(4.11)
Equilíbrio (pratos mais base)	$= N_{T+1}$	(4.1)
Hidráulica	$= N_T$	(4.4)
Controladores de nível	$= 2$	(4.19)
Continuidade de componentes no reservatório de refluxo	$= 1$	(4.6)
Total continuidade do reservatório do refluxo	$= 1$	(4.5)
Continuidade do componente de base	$= 1$	(4.18)
Continuidade total da base	$= 1$	(4.17)
Total de equações	$4 N_T+7$	

Fazendo uma análise do número de equações e de variáveis, verifica-se que o sistema está subespecificado por duas equações. Estas duas equações referem-se à razão de refluxo (R) e à taxa de aquecimento do refeedor, que podem ser consideradas constantes.

Considerando a coluna de destilação estudada, que é constituída de 16 pratos ($N_T = 16$), existem 73 variáveis e 71 equações.

Apesar da modelagem desenvolvida ser de fácil implementação, a coluna estudada não possui isolamento térmico, bem como existem pequenos vazamentos entre os pratos, não promovendo uma perfeita separação em cada estágio. Desta forma, mesmo que se trabalhe com o modelo completo da coluna, a resposta do controlador pode não ser satisfatória. Sendo assim, é perfeitamente justificável empregar uma metodologia de

identificação para obter o modelo por meio de dados de entrada e saída, o que irá possibilitar uma análise com uma fidelidade maior do processo, tornando mais efetivo o ajuste dos controladores via simulação.

4.3 Identificação de Processos

A identificação de processos iniciou-se nos anos 70, e teve um grande desenvolvimento com o método do erro de predição (PEM) e da formalização dos métodos proposta por LJUNG (1987).

A modelagem de sistemas dinâmicos pode ser feita a partir de modelos do tipo ARX (*auto-regressive with exogenous input*), em que a saída do sistema em dado instante de tempo discreto é representada utilizando amostras passadas dos seus sinais de entrada e saída (SJÖBERG et al., 1995; NELLES, 2001). Apesar destes modelos apresentarem bons resultados, têm como desvantagem o fato de que a característica autorregressiva, geralmente, leva a uma grande sensibilidade à escolha da ordem dos modelos, causando a recursão de erros, correndo o risco de comprometer a qualidade de previsão destes erros, principalmente em horizontes de previsão longos.

Segundo NELLES (2001), pode-se utilizar uma subcategoria de modelos com estrutura FIR (*Finite Impulse Response*), na qual a saída do sistema em que um dado instante de tempo discreto é representada, utiliza amostras passadas apenas dos seus sinais de entrada. Entretanto, a ausência da recursão de saída acarreta uma quantidade elevada de termos no vetor de regressão, principalmente em sistemas com dinâmicas lentas. A quantidade de termos de regressão é proporcional à quantidade de parâmetros desconhecidos no modelo. Esta relação pode ser considerada exponencial, para a grande maioria dos modelos não lineares de propósito geral (OLIVEIRA et al., 2007a).

Outra subcategoria de modelos sem realimentação de saída são os modelos OBF (*Orthonormal Basis Functions*), que são desenvolvidas com o auxílio de bases de funções ortonormais (WAHLBERG e MÄKILÄ, 1996). Estas funções têm sido largamente utilizadas no contexto de modelagem e controle de sistemas lineares e não lineares (HEUBERGER et al., 2005; OLIVEIRA et al., 2007b). Estas funções apresentam:

- i) Ausência de realimentação de saída, o que leva a modelos mais precisos, reduzindo erros de previsão, fazendo com que ocorra o desacoplamento das saídas em modelos multivariáveis e leva também a um conjunto de propriedades

que tornam favoráveis a estimação numérica de modelos lineares via algoritmo de mínimos quadrados;

- ii) Não há a necessidade de conhecer os termos passados dos sinais de entrada e saída do sistema;
- iii) Pode-se aumentar a capacidade de representação dos modelos incrementando o número de funções ortonormais na base;
- iv) Garante-se a representação de um sistema estável e a robustez a estimativas pouco precisas do atraso de transporte do sistema e tolerância a dinâmicas não modeladas.

Em relação à modelagem de sistemas não lineares, algumas das técnicas mais difundidas são:

- (i) Redes neurais (MASRI et al., 1993);
- (ii) Funções de base radial, RBF (CASDAGLI, 1989);
- (iii) Séries de Volterra (BILLINGS, 1980);
- (iv) Wavelets (STRANG, 1989);
- (v) Funções polinomiais e racionais (NOSHIRO et al., 1993; JANG e KIM, 1994);
- (vi) Equações diferenciais polinomiais (GOUESBET e LETELLIER, 1994).

Outra alternativa é a aplicação de estruturas NARMAX (não-lineares autorregressivas com média móvel e entrada exógena). Segundo CHEN e BILLINGS (1989), as propriedades gerais deste tipo de modelos constituem uma representação natural para uma grande classe de sistemas não-lineares. Uma vantagem da representação polinomial é a facilidade com que a informação analítica sobre a dinâmica do modelo pode ser obtida (AGUIRRE e MENDES, 1996). Os polinômios não-lineares são também funções lineares nos parâmetros, o que permite a utilização de algoritmos de estimação de parâmetros para modelos lineares (KORENBERG et al., 1988; CHEN et al., 1989).

Entretanto, os modelos lineares nos parâmetros requerem mais parâmetros do que modelos que são não lineares nos parâmetros. Assim, o que vai determinar o número de parâmetros de num modelo linear nos parâmetros é o tipo de regressores incluídos no modelo. Se for feita a escolha de regressores adequados, mesmo um modelo linear nos parâmetros com poucos coeficientes pode modelar a dinâmica não linear satisfatoriamente. Pode-se ainda determinar os regressores adequados. Entretanto, é um problema não trivial e que está longe de ser satisfatoriamente resolvido (AGUIRRE et al., 1998). Uma limitação dos modelos polinomiais lineares nos parâmetros é a modelagem de sistemas com não

linearidade estática que não possa ser adequadamente aproximada por polinômios de baixa ordem (AGUIRRE, 1997).

4.3.1 Elementos básicos para elaboração de modelos

Segundo LJUNG (1987), a construção de modelos a partir de dados envolve três elementos básicos:

- Conjunto de dados: Os dados de entrada/saída são obtidos a partir de experimentos exclusivos para a obtenção destes dados, determinando quais sinais medir e quando medir, ou seja, encontrar os dados que são representativos para um referido sistema.
- Conjunto de Modelos ou Estrutura de Modelos: Para escolha de um modelo é necessário um conhecimento inicial do modelo e do processo referido, sendo que o modelo pode ser de três tipos: caixa preta, cinza e branca;
- Determinação do melhor modelo: Baseado nos dados de entrada/saída é determinado o melhor modelo de um determinado sistema. É considerado o melhor modelo aquele que melhor reproduz os dados de saída a partir dos dados de entrada.

Após a execução das três tarefas anteriores, espera-se chegar a um modelo do processo que melhor descreva os dados de acordo com um critério escolhido. Para comprovar o modelo, é necessário fazer a sua validação. É importante salientar que um modelo não pode ser aceito com uma escolha final e definitiva do sistema, e sim, como uma escolha boa o suficiente para um determinado propósito.

Assim, é possível definir uma sequência lógica para identificação do processo: primeiramente coleta-se os dados, depois escolhe-se um conjunto de modelos e, finalmente, utiliza-se o modelo que melhor se adapta ao processo. A Figura 4.2 apresenta uma descrição gráfica dos procedimentos necessários para obtenção dos modelos.

Com relação ao tempo de duração do experimento para coleta de dados de entrada e saída, são consideradas as condições operacionais, custos, precisão e finalidade do modelo desenvolvido (ANDRADE, 2000).

Como os processos químicos possuem constantes de tempo bastante elevadas, a identificação geralmente é executada *off-line*, ou seja, os dados de entrada e saída são armazenados em um banco de dados apropriado para posterior análise.

Segundo LJUNG (1987), os dados de entrada e saída devem ser analisados de forma que se verifique o comportamento do processo, bem como se o sistema apresenta

não-linearidades. Deve-se selecionar uma parcela dos dados para a geração do modelo e uma outra parcela para sua validação.

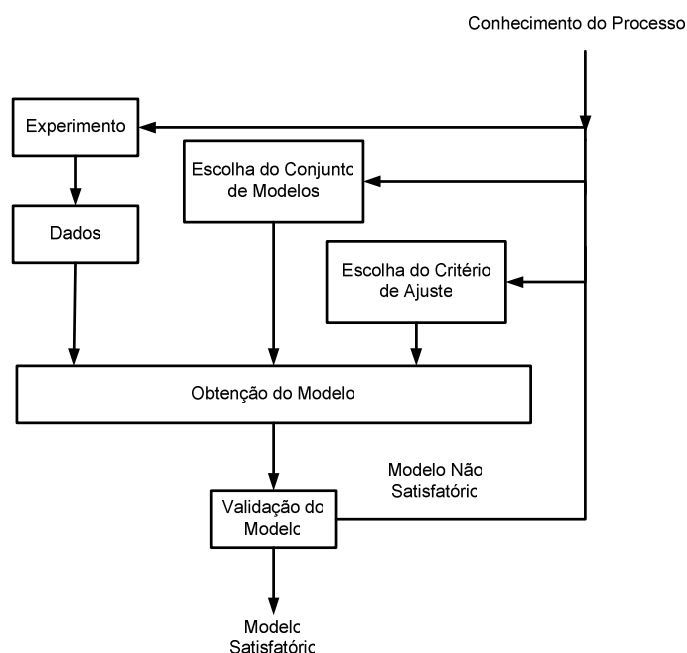


Figura 4.2 - Procedimentos necessários para obtenção de modelos de identificação.

4.3.2 Tipos de modelos

O conhecimento prévio de um modelo permitirá classificá-lo em três níveis (SJÖBERG et al., 1995):

- Modelos caixa branca (*white box*): Neste caso o modelo é obtido com a utilização das equações fenomenológicas, como balanços de massa e de energia e de um modelo físico.
- Modelos caixa cinza (*gray box*): Este modelo é recomendado quando existe um conhecimento pequeno do processo, sendo que os parâmetros dos modelos são desconhecidos e precisam ser determinados.
- Modelos caixa preta (*black box*): Não existe nenhum conhecimento físico do processo e o modelo é obtido por um conjunto de dados experimentais de entrada-saída.

Podem ser utilizados modelos discretos na identificação de sistemas, sendo os dados experimentais coletados por meio de um procedimento de amostragem. Desta forma, o problema de identificação estudado é o descrito por SJÖBERG et al. (1995).

A partir dos dados de entrada $u(t)$ e saída $y(t)$ observados de um sistema dinâmico:

$$u' = [u(1) \ u(2) \ \dots u(t)] \quad (4.20)$$

$$y' = [y(1) \ y(2) \ \dots y(t)] \quad (4.21)$$

A idéia é encontrar uma relação entre as observações passadas $[u^{t-1}, y^{t-1}]$ e as saídas futuras $y(t)$:

$$y(t) = g(u^{t-1}, y^{t-1}) + v(t) \quad (4.22)$$

O termo $v(t)$ envolve o fato de que a próxima saída $y(t)$ não será uma função exata dos dados passados. O desejável é que $v(t)$ seja pequeno de modo que $g(u^{t-1}, y^{t-1})$ possa ser uma boa predição de $y(t)$ para os dados passados.

4.3.3 Família de modelos de função de transferência

A estrutura mais simples que relaciona entrada-saída para a modelagem de sistemas lineares de modelos caixa preta é descrita pela equação (LJUNG, 1987):

$$A(q)y(t) = B(q)u(t - n_k) + e(t) \quad (4.23)$$

Expandindo:

$$\begin{aligned} y(t) + a_1 y(t-1) + \dots + a_{n_a} y(t - n_a) &= b_1 u(t - n_k) + \\ &+ b_2 u(t - n_k - 1) + \dots + b_{n_b} u(t - n_k - n_b + 1) + e(t) \end{aligned} \quad (4.24)$$

Na equação (4.24), n_a e n_b são as ordens dos polinômios $A(q)$ e $B(q)$, n_k é o número de atrasos da saída para a entrada. Pode-se apresentar uma versão mais compactada, introduzindo os vetores:

$$\theta = [a_1, a_2 \dots a_{n_a} \ b_1 \dots b_{n_b}]^T \quad (4.25)$$

$$\varphi(t) = [-y(t-1) \dots -y(t-n_a) \ u(t-1) \dots u(t-n_b)]^T \quad (4.26)$$

Assim, a equação (4.24) pode ser reescrita como:

$$y(t) = \varphi^T(t) \theta \quad (4.27)$$

Para análise é considerado o termo de ruído $e(t)$ com um erro direto na equação, e o modelo é chamado de modelo da equação do erro, sendo os parâmetros ajustáveis:

$$A(q) = 1 + a_1 q^{-1} + \dots + a_{n_a} q^{-n_a} \quad (4.28)$$

$$B(q) = 1 + b_1 q^{-1} + \dots + b_{n_b} q^{-n_b} \quad (4.29)$$

Assim, tem-se:

$$y(t) = G(q, \theta)u(t) + H(q, \theta)e(t) \quad (4.30)$$

Existe a seguinte correspondência com a equação (4.29):

$$G(q, \theta) = \frac{B(q)}{A(q)} \quad H(q, \theta) = \frac{1}{A(q)} \quad (4.31)$$

O modelo da equação (4.30) é conhecido como ARX, sendo que AR se refere à parte autorregressiva ($A(q)y(t)$) e X refere-se a uma entrada extra ($B(q)u(t)$), chamada de variável exógena. O fluxo de sinais para o modelo é descrito na Figura 4.3.

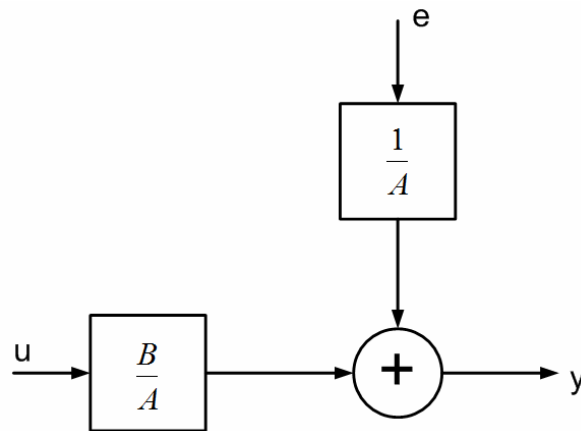


Figura 4.3 - Representação de sinais do modelo ARX.

Se for acrescentado um elemento de média móvel (MA), é obtido um modelo ARMAX, dado por:

$$A(q)y(t) = B(q)u(t - n_k) + C(q)e(t) \quad (4.32)$$

Sendo:

$$C(q) = 1 + c_1 q^{-1} + \dots + c_{nc} q^{-nc} \quad (4.33)$$

Expandindo:

$$y(t) + a_1 y(t-1) + \dots + a_{na} y(t - n_a) = b_1 u(t - n_k) + b_2 u(t - n_k - 1) + \dots + b_{nb} u(t - n_k - n_b + 1) + e(t) + c_1 e(t-1) + \dots + c_{nc} e(t - n_c) \quad (4.34)$$

Na Figura 4.4 é apresentada a descrição dos sinais para o modelo ARMAX.

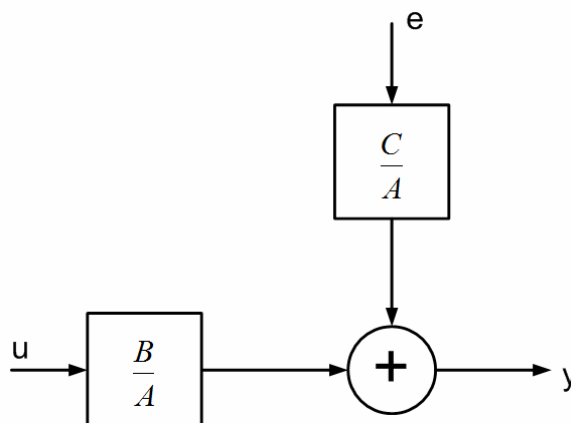


Figura 4.4 - Representação de sinais do modelo ARMAX.

Nos modelos ARX e ARMAX, são utilizados pólos comuns nas funções $G(q,\theta)$ e $H(q,\theta)$. No caso do modelo de Erro na Saída (OE), isto não acontece, sendo o erro modelado separadamente. A equação do modelo OE é apresentada a seguir:

$$y(t) = \frac{B(q)}{F(q)}u(t - n_k) + e(t) \quad (4.35)$$

Sendo:

$$F(q) = 1 + f_1 q^{-1} + \dots + c_{nf} q^{-nf} \quad (4.36)$$

A Figura 4.5 representa o fluxo de sinais para o modelo de Erro na Saída.

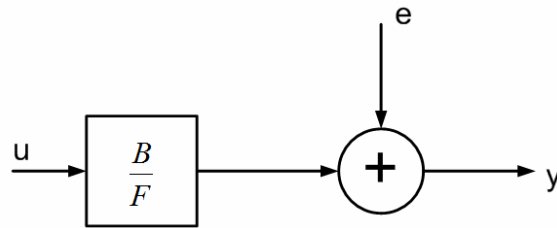


Figura 4.5 - Representação de sinais do modelo OE.

A estrutura do Modelo Box-Jenkins (BJ) fornece um modelo completo do sistema, com a dinâmica do distúrbio modelada separadamente da dinâmica do sistema, mostrada a seguir:

$$y(t) = \frac{B(q)}{F(q)}u(t - n_k) + \frac{C(q)}{D(q)}e(t) \quad (4.37)$$

Sendo:

$$D(q) = 1 + d_1 q^{-1} + \dots + d_{nd} q^{-nd} \quad (4.38)$$

A Figura 4.6 representa o modelo BJ na forma de sinais.

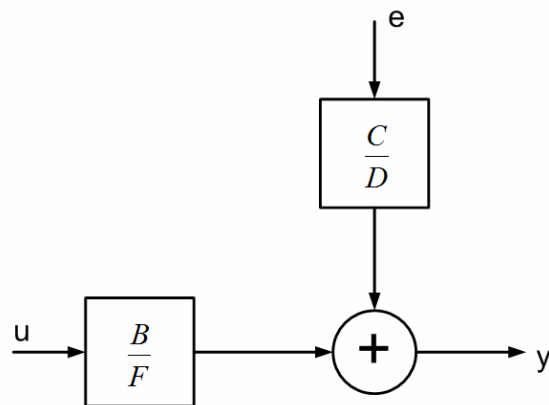


Figura 4.6 - Representação de sinais do modelo BJ.

Existe uma forma geral para representar os modelos autorregressivos, descritos anteriormente, pela seguinte equação:

$$A(q)y(t) = \frac{B(q)}{F(q)}u(t - n_k) + \frac{C(q)}{D(q)}e(t) \quad (4.39)$$

Na Tabela 4.1, estão apresentados os coeficientes empregados para obtenção das equações relativas aos modelos autorregressivos apresentados.

Tabela 4.1 - Forma geral de representação de modelos autorregressivos.

Modelo	Polinômios utilizados
ARX	A B
ARMAX	A B C
OE	B F
BJ	B F C D

Um sistema dinâmico que consiste de um número finito de elementos concentrados pode ser escrito por equações diferenciais ordinárias em que o tempo é a variável independente. Utilizando uma notação matricial e vetorial, uma equação diferencial de ordem n pode ser representada por uma equação matricial vetorial diferencial de primeira ordem. Considerando n elementos do vetor como um conjunto de variáveis de estado, a equação matricial vetorial diferencial será chamada de equação de estado. As seguintes equações descrevem o modelo de Espaço de Estados:

$$\begin{aligned} x(n+1) &= Ax(n) + Bu(n) + Ke(n) \\ y(n) &= Cx(n) + Du(n) + e(n) \end{aligned} \quad (4.40)$$

Sendo: $x(n)$ o vetor de estados, $y(n)$ a saída do sistema, $u(n)$ a entrada do sistema e $e(n)$ o erro considerado. **A**, **B**, **C**, **D**, e **K** são as matrizes do sistema.

A dimensão da matriz do vetor de estados $x(n)$ deve ser ajustada de acordo com o modelo do processo.

4.4 Conceitos Básicos de Redes Neurais Artificiais

Segundo HAYKIN (2001), as redes neurais artificiais foram projetadas para emular o comportamento do cérebro humano, ou seja, ter a capacidade de aprender e se adaptar

quando submetidas a estímulos (treinamento). Um modelo de um simples neurônio é mostrado na Figura 4.7. Um escalar de entrada p é multiplicado por um peso w para formar wp , um dos termos que é enviado para o somador. Existe ainda uma outra entrada que é multiplicada por um *bias* b e enviada ao somador, sendo a saída do mesmo conectada como entrada a uma função de transferência f , que produz a saída escalar do neurônio.

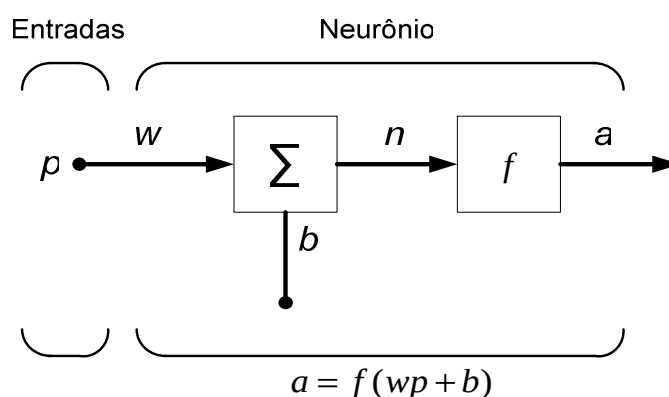


Figura 4.7 - Modelo básico de um neurônio artificial.

Tipicamente um neurônio pode possuir múltiplas entradas, como mostra Figura 4.8, sendo p o vetor de entrada, representado por uma barra sólida vertical à esquerda. As dimensões de p são mostradas a seguir, com a variável $\mathbf{R} \times 1$. As entradas são um vetor de $\mathbf{R}$ elementos. Estas entradas são ponderadas pela matriz dos pesos W , que possui $\mathbf{R}$ colunas e uma linha. A entrada para a função de transferência f é n , que é a soma de b e o produto Wp . É usual trabalhar com redes com diversas camadas, o que proporciona a possibilidade de resolução de um espectro maior de problemas através da adição das camadas escondidas, incluindo problemas discretos e regressões não-lineares. Cada camada possui sua própria matriz W , vetor b , vetor de entrada n e de saída a . Para distinguir as variáveis a matriz de peso da primeira camada será chamada de W e da segunda W^2 e assim, sucessivamente.

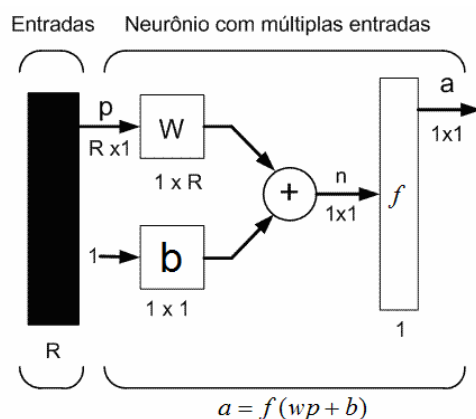


Figura 4.8 - Modelo básico de um neurônio artificial com múltiplas entradas.

A última camada é chamada de camada de saída, sendo as outras camadas chamadas de camadas escondidas. A rede mostrada na Figura 4.9 possui uma saída (camada 3) e 2 camadas escondidas (1 e 2).

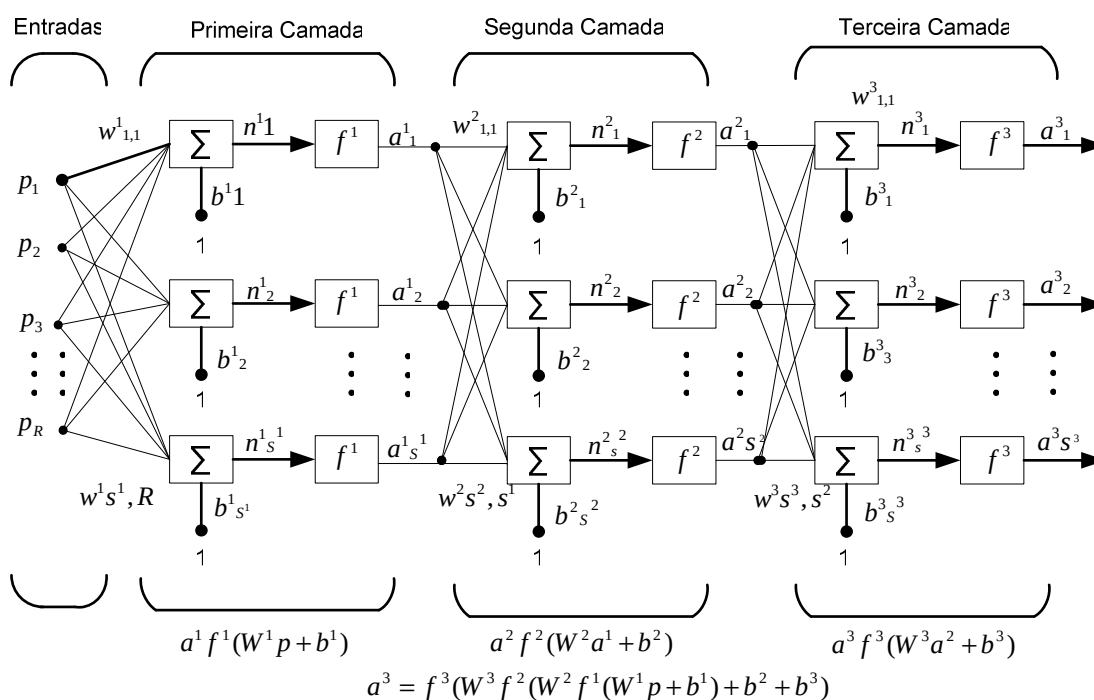


Figura 4.9 - Modelo básico de um neurônio artificial com múltiplas entradas e múltiplas camadas.

4.5 Identificação de Processos Utilizando Redes Neurais Artificiais

O uso de redes neurais artificiais tem como vantagens a possibilidade de modelar sistemas não-lineares, que resolvidos por métodos convencionais são bastante trabalhosos

e fornecem uma solução muito complexa para ser implementada em aplicações de simulação e controle. Além disso, as redes neurais são capazes de trabalhar com dados ruidosos ou incompletos e possui a capacidade de resolver problemas sem a necessidade da definição de regras ou de modelos precisos.

Os procedimentos clássicos de identificação de sistemas que utilizam os dados históricos do processo estão baseados em assumir como verdadeiras diversas hipóteses a respeito do sistema. Algumas destas hipóteses são: a linearidade, invariança no tempo, conhecer a ordem do sistema, a constante de tempo, entre outros. Os requerimentos para o uso de uma rede neural em comparação com os procedimentos clássicos de identificação são descritos a seguir (PADILHA et al., 2008).

Nos procedimentos clássicos o usuário deve especificar a característica da estrutura da relação entre as entradas e as saídas, a qual, muitas vezes, é bastante complexa, enquanto que na identificação utilizando redes neurais artificiais o fundamental é identificar a topologia da rede e a relação entre as entradas e saídas obtida durante o treinamento da rede. Assim, a função da rede é criar um modelo interno capaz de reproduzir valores numéricos que se aproximem do sistema físico dentro de um erro aceitável.

Para a identificação de uma planta em que a dinâmica do processo é considerada, cuja principal característica é a dependência com o tempo, assume-se que o sistema físico tenha uma estrutura matemática geral expressa por uma função linear ou não-linear $f(y,u)$ dos valores atuais e dos valores passados de suas entradas e saídas.

De acordo com ASSIS (2001), existem três procedimentos distintos para identificação de um sistema dinâmico utilizando redes neurais:

1- Por meio da utilização de redes neurais cujos neurônios possuem características dinâmicas em arquiteturas recorrentes, um exemplo típico desta arquitetura são as redes de Hopfield, nas quais a dinâmica do processo é capturada por uma operação de diferenciação nos neurônios, e a rede deve ser integrada ao longo do tempo para se obter as saídas. Desta forma, o treinamento equivale a uma estimativa de parâmetros em um sistema de equações diferenciais ordinárias não lineares.

2- Por meio de redes neurais artificiais com neurônios estáticos que prevêm a derivada do modelo dinâmico de um sistema em relação ao tempo e por integração numérica é calculado $y(t+1)$ a partir de $y(t)$ (THE MATHWORKS, 1998).

3- Por meio da representação de uma aproximação discreta do sistema dinâmico na forma de um modelo ARMAX, com o uso de neurônios estáticos.

A fim de obter uma melhor aproximação para o comportamento dinâmico do processo de destilação, foi aplicado um modelo NARMAX, obtido utilizando uma rede *multilayer perceptron*, com redes neurais artificiais. O modelo de um sistema finito dimensional, com ordem (n_y, n_u, n_b) e os escalares y, u e b é definido por:

$$y(k) = \Phi(y(k-1), \dots, y(k-n_y), u(k-n_u), \dots, u(k-n_u-m), b(k-1), \dots, b(k-n_b)) \quad (4.41)$$

Sendo $y(k)$ a variável autorregressiva (AR) ou a saída do sistema; $u(k)$ a variável exógena (X) ou entrada do sistema e $b(k)$ a variável de média móvel (MA) ou ruído branco.

As ordens de AR, MA e X são dadas por n_y, n_u, n_b , respectivamente e n_u e m são, respectivamente, o atraso de tempo e o máximo de atraso de tempo. A função não-linear é representada por Φ .

4.5.1 Implementação da rede neural artificial

As etapas descritas a seguir apresentam o cálculo da saída das redes neurais artificiais baseadas no vetor de entrada (CHETOUANI, 2007):

1 – Considera-se $\hat{w}^T(k)$ para o vetor de entrada $u^T(k)$ e aplica-se o mesmo nas unidades de entrada nas quais $\hat{w}^T(k)$ é o vetor de regressão dado pela seguinte equação:

$$\hat{w}^T(t) = [y(t-1), \dots, y(t-n_y), u(t-1-n_k), \dots, u(t-n_u-n_k)] \quad (4.42)$$

2- Calcula-se a entrada para as camadas escondidas:

$$net_j^h(k) = \sum_{i=1}^p W_{ji}^h(k) x_i + b_j^h \quad (4.43)$$

Sendo p o número de nós de entrada da rede, isto é $p = n_y + n_u + n_b$; j a n -ésima camada oculta; W_{ji}^h o peso das conexões entre a i -ésima entrada e a j -ésima camada oculta e b_j^h o termo de *bias* da j -ésima camada oculta.

3 – Obtêm-se a saída a partir de um nó da camada de entrada:

$$z_j = f_j^h(net_j^h(k)) \quad (4.44)$$

Sendo f_j^h a função de transferência tan-sigmoid.

4 – Calcula-se a entrada para os nós de saída:

$$net_l^q(k) = \sum_{j=1}^h W_{lj}^q(k) z_j(k) \quad (4.45)$$

Sendo l a n -ésima unidade de saída e $W_{lj}^q(k)$ o peso da conexão entre a j -ésima camada oculta e a l -ésima unidade de saída.

5- Calcula-se as saídas a partir dos nós de saída:

$$\hat{v}(k) = f_l^q(net_l^q(k)) \quad (4.46)$$

Sendo f_l^q a função de ativação, definida por:

$$f_l^q(net_l^q(k)) = net_l^q(k) \quad (4.47)$$

4.5.1.1 Algoritmo *backpropagation*

A rede neural não-recorrente com treinamento do tipo supervisionado visa ajustar os pesos e *bias* pelo gradiente descendente, de forma a minimizar o erro quadrático da rede. Durante o processo de treinamento de uma rede não recorrente, com treinamento via retropropagação, os padrões de entrada e de respostas desejadas correspondentes são apresentados à rede neural artificial. Um algoritmo de adaptação ajusta, automaticamente, os pesos de tal forma que as saídas fiquem próximas dos valores desejados. O objetivo do treinamento via retropropagação é minimizar um erro em torno de um conjunto de padrões de treinamento, utilizando o método do gradiente descendente. O algoritmo retropropagação ajusta os pesos na direção oposta ao erro do gradiente instantâneo.

O processo de aprendizagem de uma rede neural é resultado do ajuste dos pesos sinápticos de tal maneira que a diferença entre os valores de saída desejadas e da rede seja mínima (MAITELLI e FILHO, 2003).

O algoritmo de treinamento de retropropagação é feito em duas etapas:

1º- Propagação: Após a apresentação do padrão de entrada, a resposta de uma unidade é propagada como entrada para as unidades na camada seguinte, até a camada de saída, onde é obtida a resposta da rede e o erro é calculado, como ilustra a Figura 4.10.

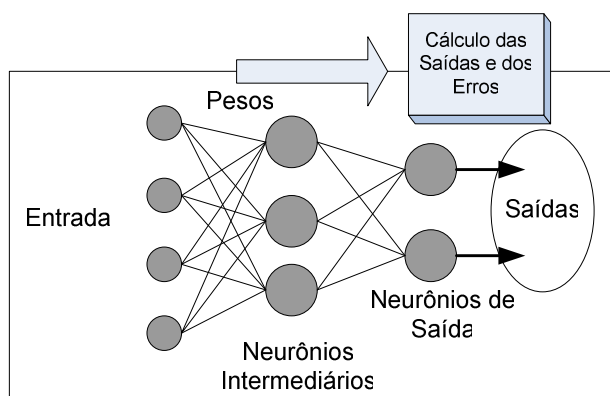


Figura 4.10 - Etapa de treinamento propagação do algoritmo *backpropagation*.

2º - Retropropagação (*backpropagation*): A partir da camada de saída até a camada de entrada, são feitas alterações nos pesos sinápticos, de forma a minimizar o erro, como mostra a Figura 4.11.

O algoritmo de retropropagação do erro é baseado no método de Descida do Gradiente (*Steepest Descent Gradient*). O fluxograma mostrado na Figura 4.12 descreve o processo de treinamento (MOGHADASSI et al., 2009).

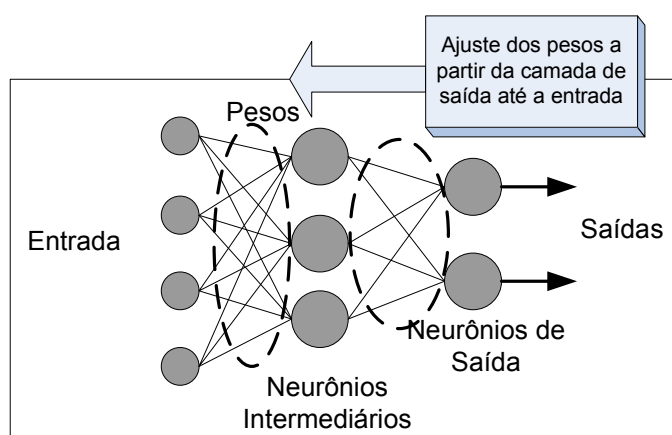


Figura 4.11- Etapa de treinamento retropropagação do algoritmo *backpropagation*.

O ajuste dos pesos sinápticos $w_{j,i}$, dá-se no sentido inverso, como mostram as expressões a seguir.

A função erro E é definida por:

$$E = \frac{1}{2} \sum_{l=1}^q (v_l(k) - \hat{v}_l(k))^2 \quad (4.48)$$

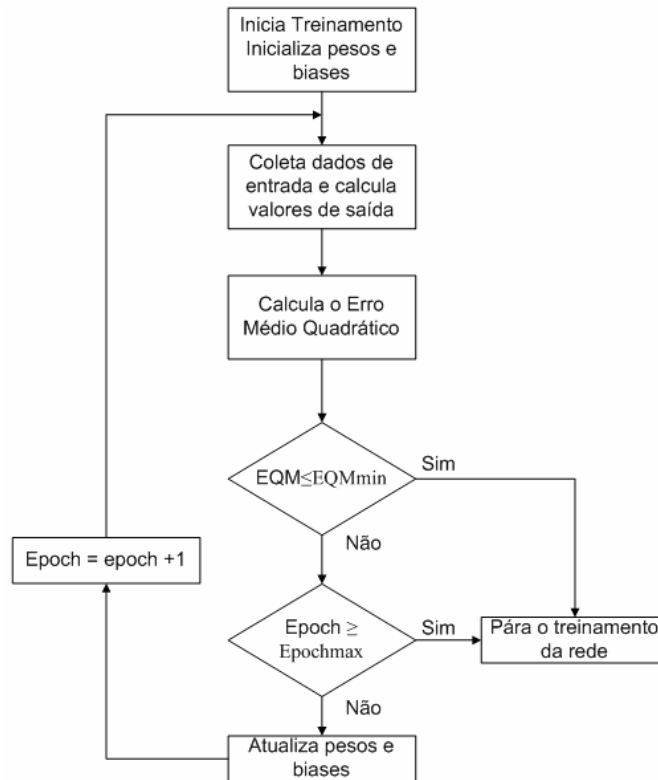


Figura 4.12- Fluxograma de treinamento do algoritmo *backpropagation*.

Sendo q o número de unidades de saída e $v_l(k)$ o l -ésimo elemento do vetor de saída da rede, compreendido em cada intervalo de tempo de k para $k+1$.

Considerando E a função erro (função custo), $\hat{v}_l(k)$ é o valor corrente assumido pelo mesmo neurônio após o processamento no sentido direto da rede. A ideia do algoritmo é ajustar os valores dos pesos sinápticos de cada conexão existente entre um neurônio j da camada escondida e um neurônio l da camada de saída, de maneira que o erro global E seja igual ou menor que uma tolerância pré-estabelecida ε ($E \leq \varepsilon$).

O algoritmo *backpropagation* busca minimizar o erro para cada valor de saída definido por E pelo ajuste dos pesos das conexões das redes, isto é, $W_{ji}^h(k)$ e $W_{lj}^q(k)$, com a aplicação das seguintes equações:

$$W_{ji}^h(k+1) = W_{ji}^h(k) + \alpha \Delta W_{ji}^h(k) - \eta \frac{\partial E}{\partial W_{ji}^h(k)} \quad (4.49)$$

$$W_{lj}^q(k+1) = W_{lj}^q(k) + \alpha \Delta W_{lj}^q(k) - \eta \frac{\partial E}{\partial W_{lj}^q(k)} \quad (4.50)$$

Sendo: η e α a constante de aprendizagem e a constante de momento, respectivamente; αW_{ji}^h e $\alpha W_{lj}^q(k)$ são as quantidades ponderadas das mudanças dos fatores η e α ; $\partial E / \partial W_{ji}^h(k)$ e $\partial E / \partial W_{lj}^q(k)$, são dados por:

$$\frac{\partial E}{\partial W_{ji}^h} = -[z_j(k)(1 - z_j(k))x_i(k)] \sum_{l=1}^q [(v_l(k) - \hat{v}_l(k))\hat{v}_l(k)W_{lj}^h(k)] \quad (4.51)$$

$$\frac{\partial E}{\partial W_{lj}^q} = -(v_l(k) - \hat{v}_l(k))z_f(k) \quad (4.52)$$

O procedimento de implementação da rede neural é feito da seguinte forma:

- 1 - Inicializa-se os pesos utilizando pequenos valores randômicos e ajusta-se a taxa de aprendizagem e o fator de momento para a rede neural;
- 2 - Aplica-se o vetor de entradas dado pela equação (4.42) para as unidades de entrada;
- 3 - Calcula-se o valor previsto do erro utilizando os dados disponíveis como n-ésima amostra ($k-1$), por meio das equações (4.42, 4.43, 4.44, 4.45, 4.46 e 4.47)
- 4 - Calcula-se o erro entre o valor previsto e o medido;
- 5 - Propaga-se o erro para trás para atualizar os pesos equações (4.49, 4.50, 4.51 e 4.52);
- 6 - Retorna-se ao passo 2.

4.6 Aplicação das Técnicas de Identificação para Obtenção do Modelo da Coluna de Destilação

O sistema de supervisão, controle e aquisição de dados foi aplicado à coluna de destilação piloto do Departamento de Engenharia Química da Universidade Estadual de Maringá.

Para a realização do experimento foram colocados sensores de temperatura do tipo DS18B20 externamente à coluna, nos pratos de refluxo, 7, 9, 10, 12, 13, 14 (de cima para baixo) e no refeedor (*reboiler*) da coluna de destilação.

Esta coluna é constituída de 16 pratos perfurados, sendo o aquecimento do refeedor feito por meio de resistências elétricas. A variação da quantidade de energia utilizada se dá por meio da variação da potência elétrica empregada nas resistências. A mistura a ser destilada é colocada em um reservatório na parte superior do módulo e, por gravidade, é introduzida no prato de alimentação da coluna de destilação.

O trabalho desenvolvido foi aplicado em um experimento na coluna de destilação para separação de uma mistura composta por etanol e água. As condições de operação da coluna foram: Vazão e concentração de entrada de 220 mL/min e 33 GL, respectivamente; temperatura de alimentação de 80 °C; Vazão de saída e concentração de destilado de 53 mL/min e 93 GL, respectivamente; razão de refluxo igual a 3,7 e a vazão e concentração do produto de fundo 168 mL/min e 4 GL, respectivamente.

Por meio do *software* de supervisão e da placa de aquisição de dados desenvolvidos foram coletadas as temperaturas nos pratos, além das temperaturas da alimentação e do refeedor.

Para obtenção do modelo foram utilizados dados de leitura da temperatura do refeedor (variável manipulada) e da temperatura de topo da coluna, representada pelo prato 7 (variável de processo). A Figura 4.13 mostra a interligação da coluna de destilação no sistema de aquisição de dados.

Os dados coletados na coluna de destilação foram enviados a uma placa de aquisição de dados, na qual os dados de entrada/saída eram armazenados em um banco de dados.

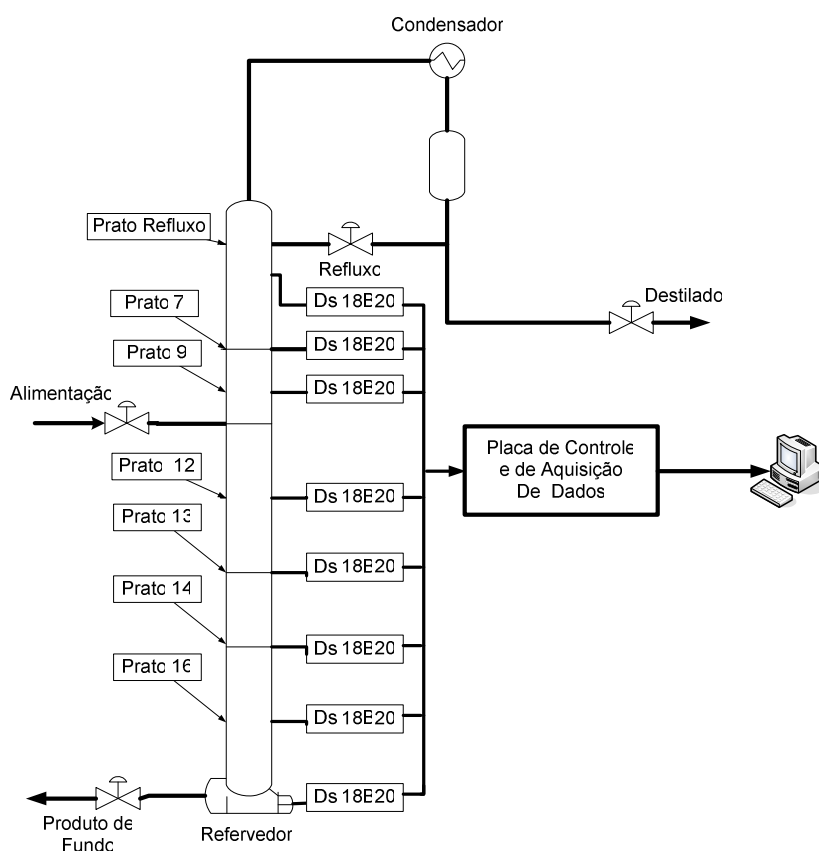


Figura 4.13 - Conexão da coluna de destilação com sistema de aquisição de dados.

Os dados armazenados são utilizados para a obtenção do modelo do processo utilizando as estratégias de identificação, como ilustra a Figura 4.14.

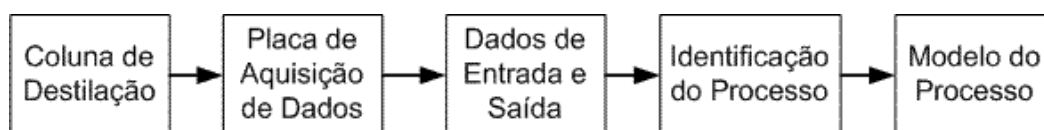


Figura 4.14 - Etapas da obtenção do modelo do processo por meio de identificação.

Para a estimativa do modelo matemático foram levantados 658 pares de dados de entrada/saída (refervedor/topo), como é mostrado na Figura 4.15.

Para a identificação do processo foram aplicados os seguintes modelos: *Output Error* (OE), ARX, Espaço de Estados, BJ e ARMAX. Os modelos desenvolvidos foram aplicados com o auxílio do *software* de simulação MATLABTM, sendo os programas elaborados para obtenção dos modelos apresentados no apêndice da tese. Os resultados obtidos por meio das simulações são expostos nas seções a seguir.

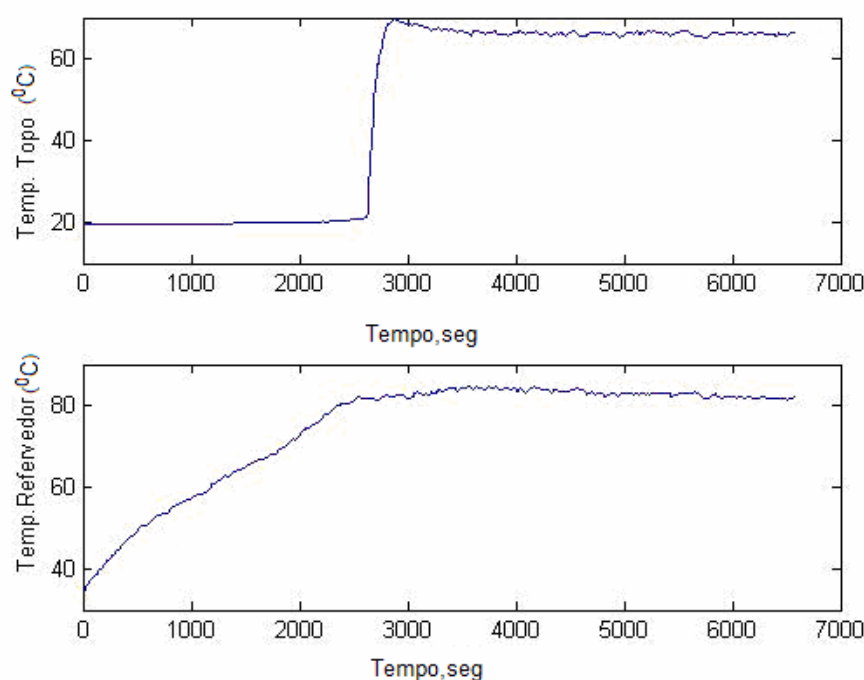


Figura 4.15 - Dados de temperatura do topo coluna e fervedor.

4.6.1 Modelo ARX

Considerando o modelo de identificação ARX desenvolvido e implementado no *software* MATLABTM, obteve-se um modelo de ordem elevada (arx620). A melhor

resposta obtida com a aplicação deste modelo foi a apresentada por um sistema composto de 6 pólos, 2 zeros e um número de atrasos da entrada em relação a saída igual a zero.

A equação (4.53) representa o modelo ARX com os seus respectivos coeficientes.

$$A(q)y(t) = B(q)u(t) + e(t) \quad (4.53)$$

$$A(q) = 1 - 1.346 q^{-1} + 0.1372 q^{-2} - 0.0878 q^{-3} + 0.1807 q^{-4} + 0.2231 q^{-5} - 0.00677 q^{-6}$$

$$B(q) = -0.2389 q^{-2} + 0.2412 q^{-3}$$

Na Figura 4.16, está representada a relação entre os dados de entrada-saída e o modelo ARX obtido.

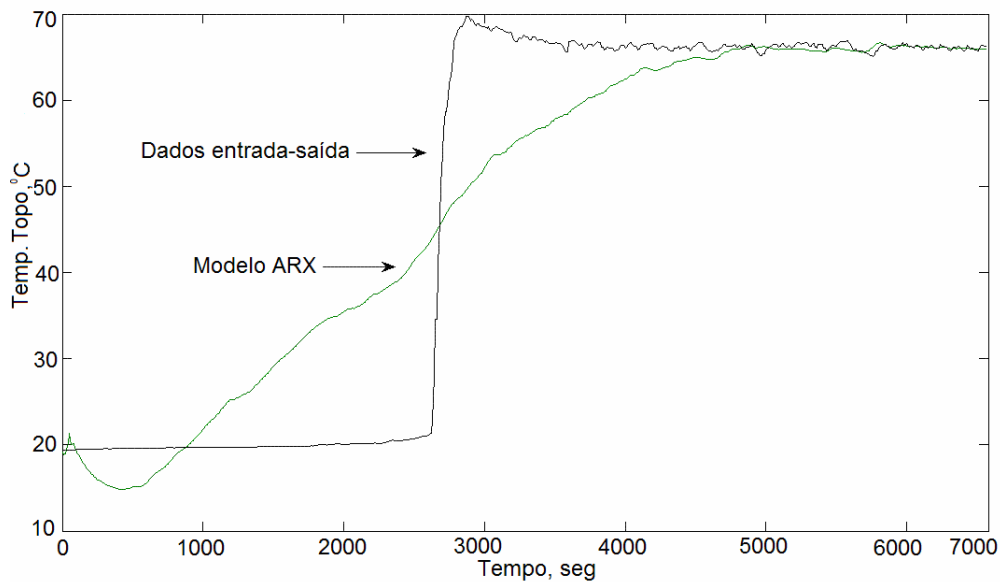


Figura 4.16 - Comparação entre dados de entrada e saída e modelo ARX.

4.6.2 Modelo Espaço de Estados

O modelo de Espaço de Estados é representado pela equação (4.54), sendo os coeficientes encontrados com auxílio da simulação a partir do modelo de identificação desenvolvido são apresentados a seguir.

$$x(t+Ts) = \mathbf{A}x(t) + \mathbf{B}u(t) + \mathbf{K} e(t) \quad (4.54)$$

$$y(t) = \mathbf{C}x(t) + \mathbf{D}u(t) + e(t)$$

$$A = \begin{bmatrix} 0.99379 & -0.044851 & 0.019981 & 0.00052 & -0.00809 & -0.00376 & -0.00540 \\ 0.04449 & 0.89543 & 0.25141 & 0.12785 & 0.01894 & -0.26078 & -0.12352 \\ 0.01694 & -0.14448 & -0.52331 & 0.47852 & 0.34836 & -0.27954 & -0.062814 \\ -0.00075 & -0.05728 & 0.24534 & -0.48798 & 0.80009 & -0.057069 & -0.03560 \\ -0.00244 & 0.03009 & -0.41503 & -0.63015 & -0.34712 & -0.50286 & -0.21003 \\ -0.013899 & 0.04529 & -0.11374 & 0.00473 & 0.06603 & 0.60082 & -0.81085 \\ 0.01387 & 0.04684 & -0.1889 & -0.17988 & 0.11114 & 0.17654 & 0.29047 \end{bmatrix}$$

$$B = \begin{bmatrix} -0.02270 \\ -0.06750 \\ 1.90951 \\ -0.56308 \\ 0.71297 \\ 0.74985 \\ 0.45399 \end{bmatrix} \quad C = \begin{bmatrix} -0.02270 \\ -0.06750 \\ 1.90951 \\ -0.56308 \\ 0.71297 \\ 0.74985 \\ 0.45399 \end{bmatrix} \quad D = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \quad K = \begin{bmatrix} 0.00383 \\ -0.03147 \\ -0.02748 \\ -0.00740 \\ -0.00342 \\ 0.00077 \\ 0.0039552 \end{bmatrix}$$

Na Figura 4.17 está representada a relação entre os dados de entrada-saída e o modelo de Espaço de Estados obtido.

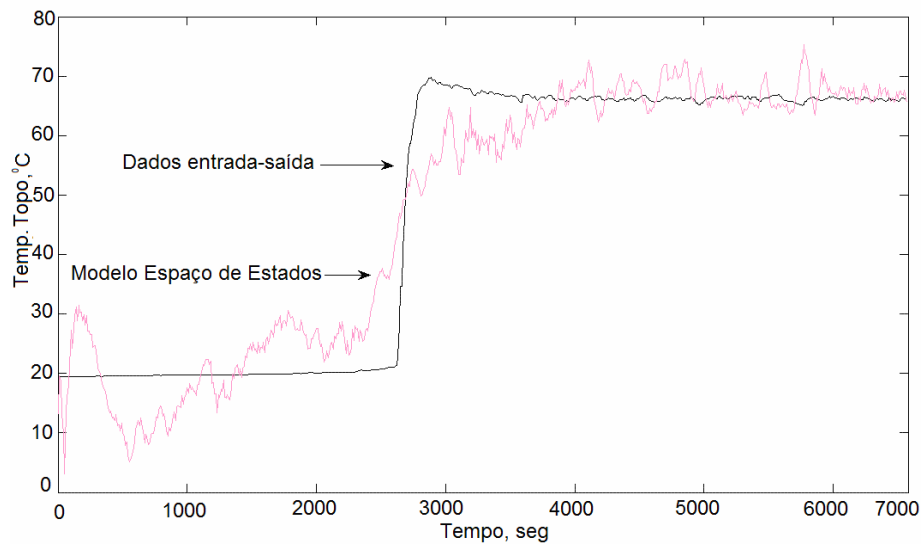


Figura 4.17 - Comparação entre dados de entrada e saída e modelo Espaço de Estados.

4.6.3 Modelo de Erro de saída (OE)

O modelo OE é representado pela equação (4.55).

$$y(t) = [B(q)/F(q)]u(t) + e(t) \quad (4.55)$$

A melhor resposta obtida com a aplicação deste modelo foi a apresentada com um sistema composto de 6 pólos e 2 zeros, sendo os coeficientes encontrados, representados a seguir:

$$B(q) = 0.09238 q^{-1} - 0.09257 q^{-2}$$

$$F(q) = 1 - 1.308q^{-1} - 0.414q^{-2} + 1.761q^{-3} - 1.72q^{-4} - 0.04582q^{-5} + 1.138q^{-6}$$

Na Figura 4.18 está ilustrada a relação entre os dados de entrada-saída e o modelo OE obtido.

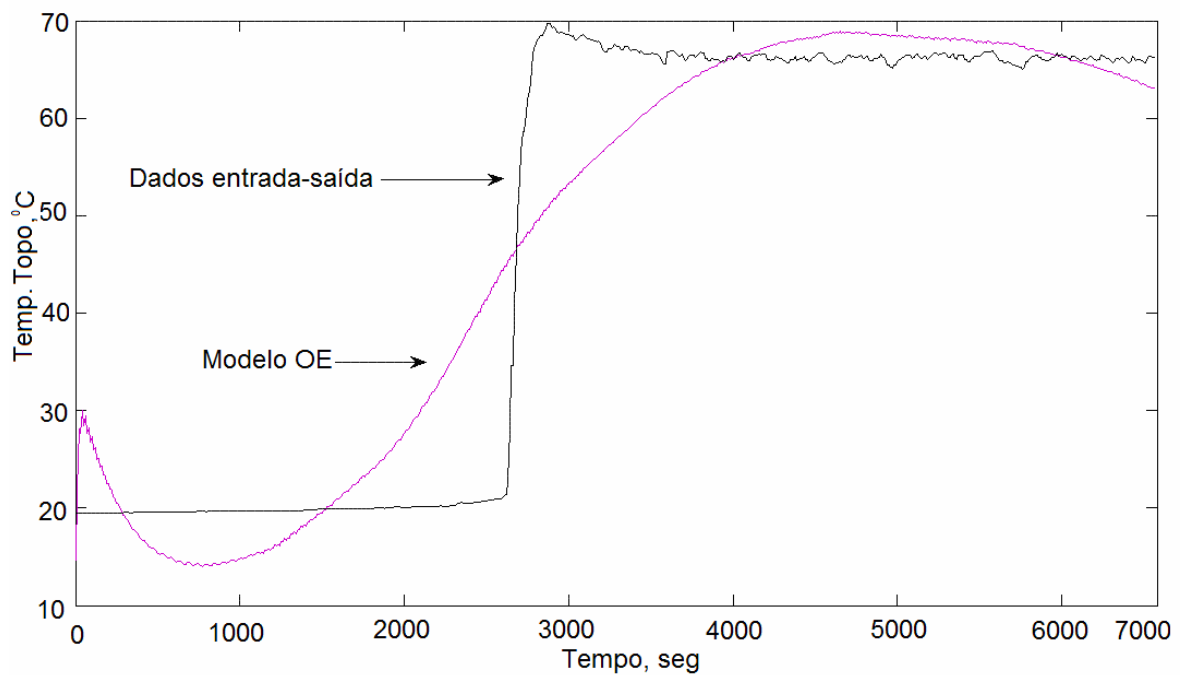


Figura 4.18 - Comparação entre dados de entrada e saída e modelo OE

4.6.4 Modelo ARMAX

O modelo ARMAX é descrito pela equação (4.56).

$$A(q)y(t) = B(q)u(t) + C(q)e(t) \quad (4.56)$$

A resposta obtida com a aplicação deste modelo foi a apresentada por um sistema composto de 5 pólos e 2 zeros, sendo os coeficientes encontrados por meio da simulação estão representados a seguir:

$$\begin{aligned}
 A(q) &= 1 - 1.412q^{-1} + 1.237q^{-2} - 1.245q^{-3} + 0.1765q^{-4} + 0.1334q^{-5} \\
 B(q) &= 0.00611q^{-1} + 0.007057q^{-2} \\
 C(q) &= 1 - 0.06815q^{-1} + 0.9806q^{-2}
 \end{aligned}$$

Na Figura 4.19 está representada a relação entre os dados de entrada-saída e o modelo ARMAX obtido.

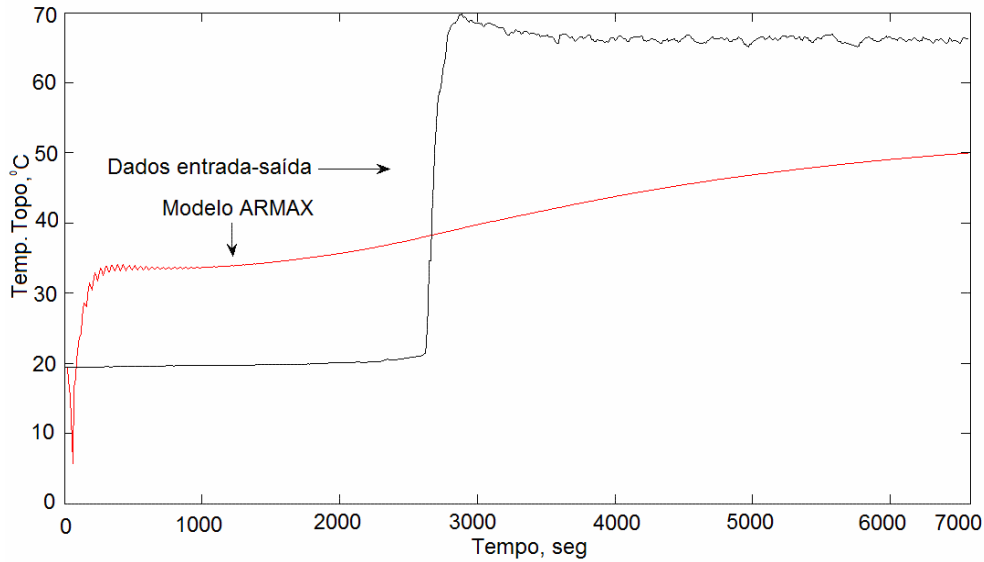


Figura 4.19 - Comparação entre dados de entrada e saída e modelo ARMAX.

4.6.5 Modelo Box Jenkins (BJ)

Segundo HAGAN et al., (1996) o modelo BJ é representado pela equação (4.57), sendo apresentada a resposta obtida com a aplicação deste modelo a um sistema composto de 6 pólos e 2 zeros. Os coeficientes encontrados por meio de simulação são representados a seguir.

$$y(t) = [B(q)/F(q)]u(t) + [C(q)/D(q)]e(t) \quad (4.57)$$

$$B(q) = 1 - 1.412q^{-1} + 1.237q^{-2} - 1.245q^{-3} + 0.1765q^{-4} + 0.1334q^{-5} - 0.009581q^{-6} + 0.1207q^{-7}$$

$$C(q) = 1 - 0.5007q^{-1} - 0.1289q^{-2}$$

$$D(q) = 1 - 1.88q^{-1} + 0.8798q^{-2}$$

$$F(q) = 1 - 2.153q^{-1} + 2.598q^{-2} - 3.841q^{-3} + 3.809q^{-4} - 2.574q^{-5} + 2.114q^{-6} - 0.9511q^{-7}$$

Na Figura 4.20 está representada a relação entre os dados de entrada-saída e o modelo BJ obtido.

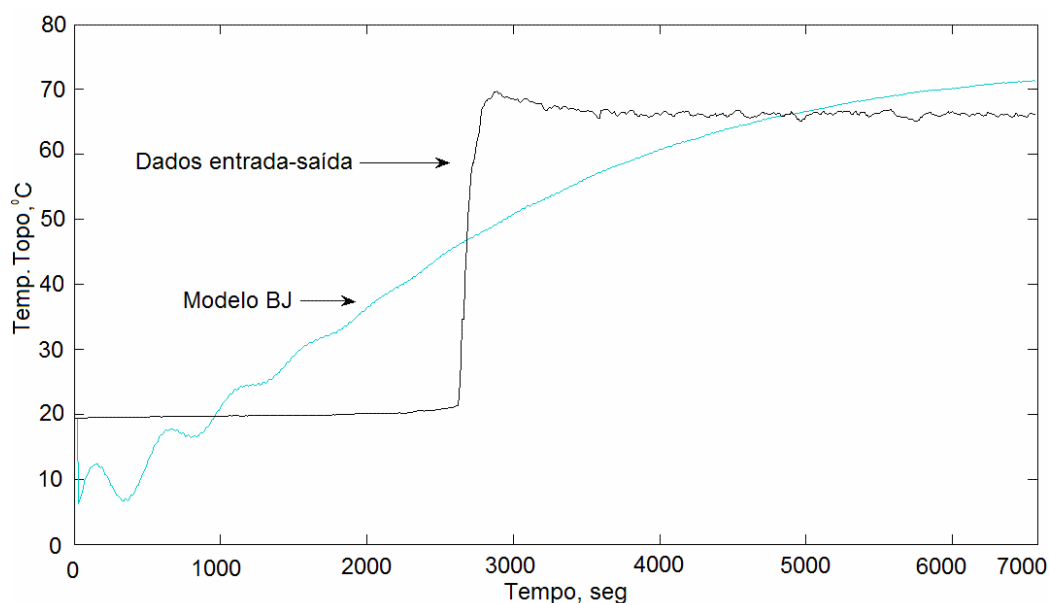


Figura 4.20 - Comparação entre dados de entrada e saída e modelo BJ.

Na Tabela 4.2 foram tabulados os dados relativos à qualidade dos modelos obtidos, representados pelo erro médio quadrático.

Tabela 4.2 - Erro médio quadrático obtido para os modelos de identificação.

Modelo	Erro Médio Quadrático (%)
OE	22,86
Espaço de Estados	29,26
ARX	39,28
BJ	45,07
ARMAX	82,79

Apesar de serem empregados modelos de ordens elevadas para os modelos OE, ARX, BJ e ARMAX, com o número de pólos de 7, 6 e 5, respectivamente, a magnitude dos erros médios quadráticos resultantes com a aplicação dos modelos de identificação indicam que os modelos de identificação empregados não forneceram uma resposta satisfatória.

A magnitude dos erros médios quadráticos obtidos entre o conjunto de dados de entrada-saída e os modelos foi na ordem de 10^1 . Segundo PEÑA et al. (2007) são aceitáveis erros médios quadráticos para modelos de processos químicos na ordem

de 10^{-1} . Assim, os resultados obtidos por meio de simulações indicam que os modelos de identificação empregados não apresentaram uma resposta satisfatória devido ao fato de possuírem erros médios quadráticos bastante elevados. Existe a possibilidade da redução dos erros obtidos a partir dos modelos, aumentando-se a ordem dos mesmos. Entretanto, para a obtenção de um modelo viável para a aplicação de estratégias de controle é fundamental se fazer uma análise levando-se em consideração o compromisso entre complexidade e desempenho do modelo obtido, pois seria necessário um grande aumento na ordem dos modelos para se obter uma pequena melhoria de performance dos modelos.

Desta forma, será apresentada a seguir, uma técnica de identificação por meio de redes neurais artificiais, mais adequada para representar as não-linearidades da coluna estudada.

4.6.6 Redes neurais artificiais

De acordo com os resultados expostos na Tabela 4.2, os modelos de identificação OE, Espaço de Estados, ARX, BJ e ARMAX não apresentaram uma boa resposta para identificação do processo da coluna de destilação. Desta forma, foi aplicada a identificação via redes neurais artificiais utilizando um modelo NARMAX.

As redes neurais artificiais aplicadas para identificação de sistemas dinâmicos utilizam um laço de realimentação com as saídas anteriores da planta, considerando-as como entrada adicional às entradas existentes.

A identificação dos parâmetros da planta foi realizada em uma rede neural artificial de acordo com o esquema proposto por VAS (1999), sendo y_p a saída da planta e a saída da rede neural artificial é $\hat{y} = y_p''$. O erro é dado por: $e(t) = y_p - \hat{y}$, que é utilizado para alimentar o algoritmo de treinamento da rede neural. Este é considerado um treinamento supervisionado, sendo a planta responsável por fornecer os valores desejados (saídas da planta) para que a rede possa ser treinada para realizar seu objetivo de identificação. Na Figura 4.21, tem-se um esquema representando a metodologia utilizada para o treinamento da rede neural.

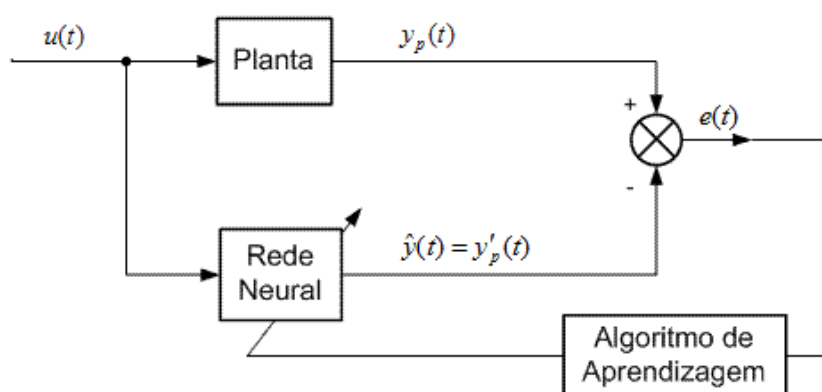


Figura 4.21 - Representação do esquema de identificação utilizando redes neurais.

A estrutura básica da rede neural foi definida com uma camada de entrada, 4 camadas ocultas e uma camada de saída. O número de neurônios empregados foi: 4, 2 e 1 neurônios nas camadas de entrada, oculta e de saída, respectivamente. Foi utilizada uma rede *feedforward* com o algoritmo de treinamento *backpropagation*, sendo empregada a função de ativação Tansig.

Com a utilização da topologia descrita e com o auxílio do software MATLABTM, cujo programa desenvolvido encontra-se no apêndice da tese, foi realizado o treinamento da rede neural, sendo alcançado o resultado mostrado na Figura 4.22.

Um conjunto adicional de dados foi utilizado para testar e validar o modelo obtido, como mostra a Figura 4.23.

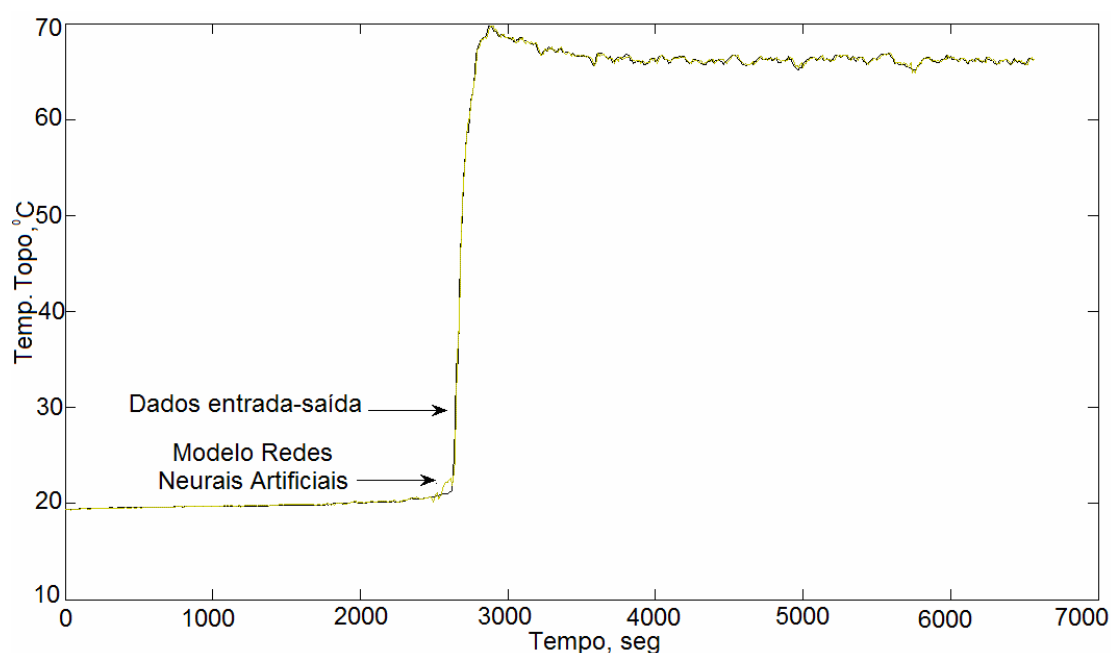


Figura 4.22 - Comparação entre dados de entrada e saída e modelo com redes neurais artificiais (treinamento).

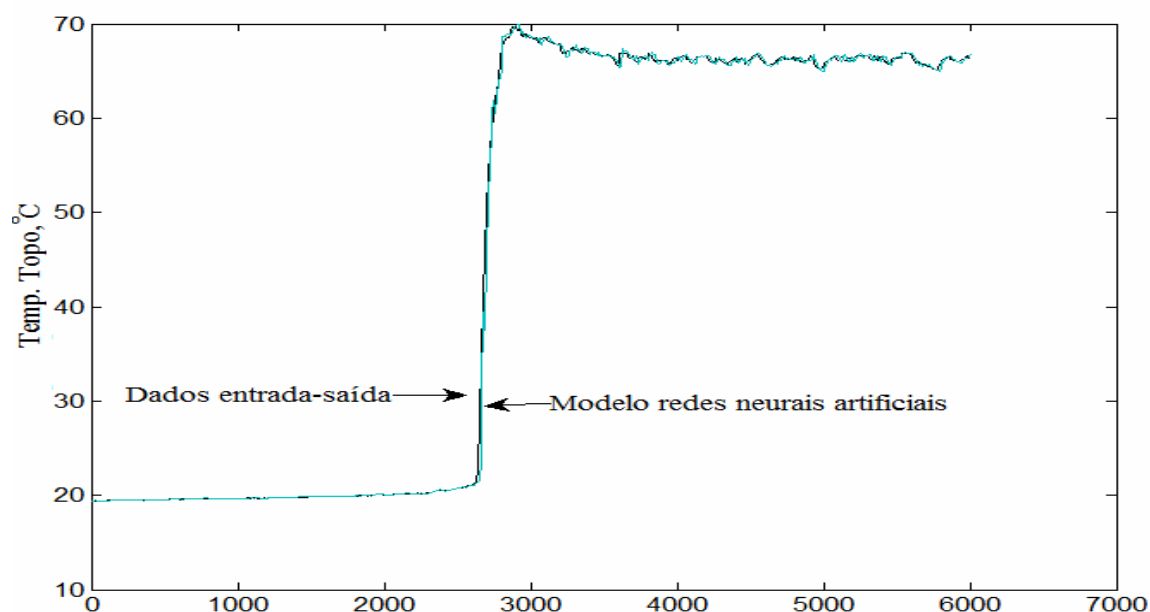


Figura 4.23 - Dados aplicados no modelo para teste e validação.

Como é possível observar a partir das Figuras 4.22 e 4.23, o modelo obtido por meio da identificação utilizando de redes neurais artificiais trouxe resultados muito superiores aos apresentados pelas técnicas ARX, ARMAX, BJ, OE e Espaço de Estados, fornecendo um erro médio quadrático com percentual de aproximadamente 0,66. O erro obtido com o uso deste modelo é menor em aproximadamente 34, 44, 60, 68 e 125 vezes em relação aos modelos OE, Espaço de Estados, ARX, BJ e ARMAX, respectivamente. O erro médio quadrático encontrado com o modelo de redes neurais artificiais enquadra-se dentro da faixa de 10^{-1} , aceitável para processos químicos (PEÑA et al., 2007). Desta forma, é possível o uso do modelo obtido para sintonia dos controladores mediante aplicação do mesmo em um ambiente de simulação, sendo esta etapa apresentada no capítulo 5 a seguir.

CAPÍTULO 5 – SIMULAÇÃO DAS ESTRATÉGIAS DE CONTROLE

Efetuada a análise dos dados de entrada/saída da coluna de destilação, foi obtido um modelo aproximado por meio de técnicas de identificação. Para avaliar a performance das estratégias de controle, foram feitas simulações com o modelo obtido.

De acordo com os objetivos deste trabalho, neste capítulo estão contidas as simulações das estratégias de controle PID, *fuzzy* e *neuro-fuzzy*, bem como os resultados e sintonia de parâmetros obtidos utilizando o *software* MATLABTM.

As seções iniciais trazem a descrição dos fundamentos utilizados para confecção e sintonia das estratégias de controle.

O objetivo deste capítulo é demonstrar as etapas relativas à configuração dos controladores, bem como prever a sua eficácia e viabilidade para aplicação em malhas de controle de colunas de destilação.

5.1 Estratégias de Controle

A necessidade da utilização de estratégias de controle em processos químicos é fundamental para garantir a obtenção de objetivos operacionais. Estes objetivos podem ser obtidos com o uso racional do arranjo de equipamentos (equipamentos de medição, válvulas, controladores, computadores) e intervenção humana (engenheiros de processo, operadores), que juntos constituem o sistema de controle.

Segundo SHINSKEY (1996), existem três classes gerais de necessidades que um sistema de controle precisa atender:

- Eliminar a influência de perturbações externas;
- Garantir a estabilidade de um processo químico;
- Otimizar a performance de um processo químico.

5.1.1 Aplicação de estratégias de controle em processos químicos

Para a aplicação de estratégias de controle a obtenção do modelo do processo a ser estudado pode ser útil, devido às seguintes funcionalidades:

- (i) Utilização como base para métodos de projeto de controladores clássicos;
- (ii) Pode ser incorporado diretamente na lei de controle, que é uma aproximação para muitas técnicas de controle avançado;
- (iii) Empregado para desenvolver uma simulação computacional do processo de modo a permitir a exploração de estratégias de controle alternativas e calcular valores preliminares de ajustes de controladores.

Segundo SEBORG et al. (2003), os passos a serem seguidos para projetar e instalar sistemas de controle com o auxílio de um modelo do processo são apresentados na Figura 5.1.

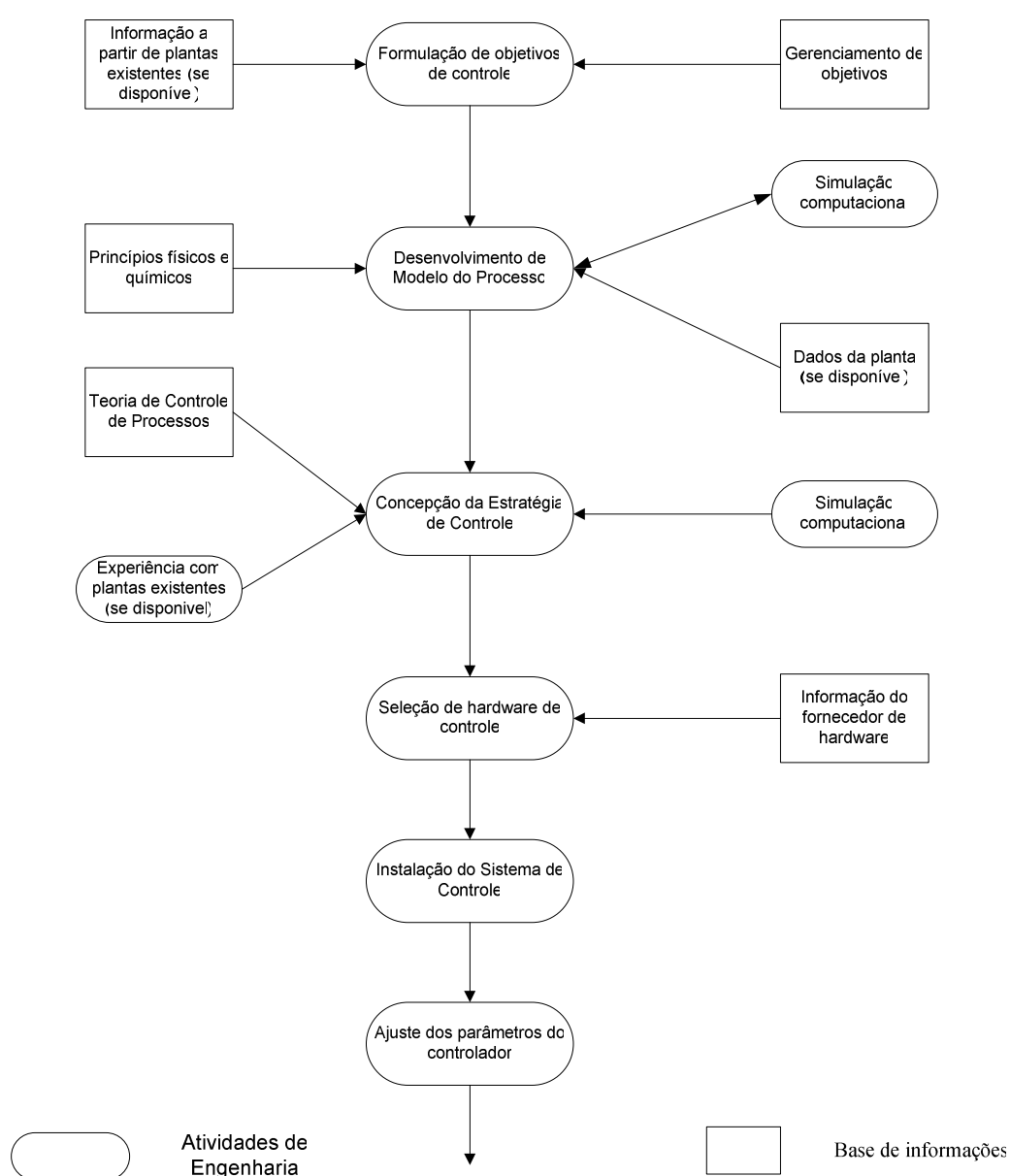


Figura 5.1 - Passos necessários para desenvolvimento de um sistema de controle.

O primeiro passo é a formulação dos objetivos de controle, levando-se em consideração sempre as restrições do processo.

Um modelo dinâmico é desenvolvido após a definição dos objetivos de controle. O modelo do processo pode ter uma base teórica, utilizando princípios químicos e físicos, como conservação de energia e massa, ou o modelo pode desenvolvido empiricamente a partir de dados experimentais.

O próximo passo no projeto do sistema de controle é aplicar uma estratégia de controle apropriada, que irá atingir os objetivos de controle, respeitando as restrições do processo.

Ferramentas de simulação para o processo controlado são utilizadas para apresentar alternativas de estratégias de controle e determinar ajustes preliminares dos controladores. Finalmente, o *hardware* do sistema de controle é selecionado e aplicado no processo. Os controladores são sintonizados na planta, usando estimativas preliminares obtidas nos passos iniciais do projeto de controle.

5.1.2 Estratégias empregadas em controle de colunas de destilação

A partir de 1960, uma variedade de técnicas matemáticas e computacionais foram introduzidas. Nos últimos anos, foram largamente utilizados controles adaptativos e preditivos. Existem muitas técnicas adaptativas que vão desde um simples controlador PID, que se ajusta de acordo com uma relação entre a variável medida e os parâmetros de ganho do controlador (*gain scheduling*) até métodos estatísticos. Para estas técnicas, um modelo interno é estatisticamente ajustado para a resposta medida e os parâmetros de ajuste são determinados para alcançar alguns critérios de performance como mínima variação.

O controle preditivo descreve algoritmos de controle que usam uma convolução interna discreta para prever a saída do processo e, então, ajustar os parâmetros do controlador de refluxo até o processo seguir para uma trajetória especificada. Este modelo é indicado para processos que são difíceis de modelar, multivariáveis e têm restrições.

O principal objetivo da destilação é separar componentes da mistura, a fim de obter um produto final com a pureza dentro de uma determinada especificação (SEBORG et al., 2003). Autores como REMBERG et al. (1994) destacam que o controle de uma coluna de destilação deve ser feito observando-se cinco aspectos: (1) especificação dos produtos, (2) variações na qualidade do produto, (3) recuperação do produto, (4) custo das utilidades e (5) práticas de operação. Desta forma, as estratégias de controle devem ser definidas para

cada tipo de processo de destilação. Conforme estudos (RADEMAKER et al., 1975; REMBERG et al., 1994; SHINSKEY, 1996), a grande maioria das colunas de destilação é tratada como um sistema 5 x 5, tendo como variáveis controladas os níveis na base e no acumulador, a pressão da coluna e as concentrações das correntes de produto. Para atender estas especificações, diversas combinações de pares de variáveis controladas e manipuladas podem ser estabelecidas. O controle preditivo é uma técnica bastante utilizada devido a sua flexibilidade no controle de mudanças de cargas nas colunas de destilação, conforme estudos de BRABILLA e D'ELIA (1992) e DOMA et al. (1996). Com a utilização do controlador PID a maior dificuldade encontra-se no ajuste dos parâmetros. Autores como ALVAREZ et al. (2000), ADERSEN e KÜMMEL (1989), HILL (2001), PEDRET et al. (2002), SHUNTA (1995), VERMEER et al. (1998) apresentam novas configurações de um controlador PID como autoajustáveis ou baseados em observadores de estado. Também existem técnicas aplicadas que combinam PID com sistemas multivariáveis (NOORAI et al., 1999; SANDELIN et al., 1991) e com controle não linear (VESTER et al., 1993). Também foram abordadas técnicas baseadas em lógica difusa (*fuzzy*) (LU et al., 2001; RAVI et al., 1997). Uma técnica neuro-*fuzzy* adaptativa para controle de colunas de destilação foi proposta por CAÑETE et al. (2004). Outros estudos utilizam lógica difusa com controladores preditivos (MAHFOUF et al., 2002). MARANGONI (2005) propôs uma nova estratégia de controle com ação distribuída nos pratos, diferentemente dos métodos convencionais normalmente utilizados, conseguindo diminuir os transientes de operação. JANA et al. (2007) propuseram dois controladores lineares chamados DMC (*Dynamic Matrix Controller*) juntamente com um controlador convencional PI (Proporcional Integral), obtendo performance superior quando o processo foi submetido a perturbações externas.

5.1.3 Controle clássico

Existem, basicamente, duas configurações de malhas de controle de acordo com o número de variáveis de entrada e de saída:

- SISO – *single-input, single-output*. Nesta configuração está representada uma variável de entrada (variável de processo) e uma variável de saída (variável manipulada);
- MIMO – *multiple-input, multiple-output*. Nesta configuração existe mais de uma variável de entrada e mais de uma variável de saída.

Existem ainda algumas variações destas nomenclaturas, como: SIMO (*single-input, multiple-output*): uma variável de entrada e mais de uma variável de saída; MISO (*multiple-input, single-output*): mais de uma variável de entrada e uma variável de saída.

A comparação do valor de referência (*setpoint*) da variável de processo com o valor atual irá gerar um sinal de erro ao controlador, produzindo um valor de saída com o objetivo de anular este erro. COUGHANOWR e LEBLANC (2008) descrevem os controladores nas seguintes categorias apresentadas a seguir.

5.1.3.1 Controlador proporcional (P)

No controlador P, a ação de controle ($m(t)$) é proporcional ao produto do erro e uma constante (K_c), chamada de ganho proporcional. O erro ($e(t)$) é dado pela diferença entre o valor da variável de processo e o valor de referência. A equação que rege este tipo de controlador é:

$$m(t) = K_c \cdot e(t) \quad (5.1)$$

Tem como característica apresentar uma resposta com pequena oscilação. Entretanto, possui sempre um erro em regime permanente da variável de processo.

5.1.3.2 Controlador proporcional integral (PI)

Neste controlador, a ação de controle é definida por:

$$m(t) = K_c \left(e(t) + \frac{1}{\tau_i} \int_0^t e(t) dt \right) \quad (5.2)$$

Sendo:

τ_i = Tempo integral

Este controlador tem como característica não possuir erro (*offset*) em regime permanente. Contudo, apresenta uma sobrelevação (*overshoot*), possuindo um considerável período de oscilação em comparação ao controlador proporcional.

5.1.3.3 Controlador proporcional derivativo (PD)

Neste controlador a ação de controle é proporcional ao erro e à taxa de variação do erro, sendo descrito pela equação:

$$m(t) = Kc \left(e(t) + \tau d \cdot \frac{d}{dt} e(t) \right) \quad (5.3)$$

Sendo:

τd = Tempo derivativo

Neste tipo de controlador o processo atinge o regime permanente em um tempo mais curto, com menor oscilação em relação ao PI. Entretanto, apresenta um considerável *offset* em regime permanente.

5.1.3.4 Controlador proporcional integral derivativo (PID)

Neste controlador existe a combinação das vantagens das ações de controle proporcional integral (PI) e proporcional derivativa (PD). A equação (5.4) descreve o controlador PID.

$$m(t) = Kc \left(e(t) + \frac{1}{\tau i} \int_0^t e(t) dt + \tau d \cdot \frac{d}{dt} e(t) \right) \quad (5.4)$$

No controlador PID a ação proporcional-integral tem como função reduzir o erro em regime permanente, sendo a ação proporcional-derivativa responsável pela redução do *overshoot*, bem como pela eliminação da oscilação eventualmente provocada pelo controlador PI.

5.1.3.5 Métodos de sintonia de controladores PID

A sintonia de um controlador é o ajuste dos seus parâmetros, para se obter uma resposta desejada em malha fechada. Os valores de sintonia dos parâmetros dependem de características dinâmicas do processo e dos elementos da malha de controle.

Neste trabalho o método de sintonia utilizado foi o de Ziegler e Nichols, devido a sua larga utilização e excelentes resultados obtidos nos ambientes acadêmico e industrial (SPANDRI, 2003).

Antes de apresentar o método de sintonia Ziegler e Nichols é importante salientar que existem duas variações do referido método. Na primeira é obtida a resposta a um degrau unitário em malha aberta, e na segunda o processo é colocado em malha fechada sendo denominado método do ganho crítico, descrito a seguir (OGATA, 2003).

5.1.3.5.1 Método de Ziegler e Nichols do ganho crítico

Este foi um dos primeiros métodos de sintonia de controladores PID, também conhecido com método de sintonia *on-line* ou em malha fechada, sendo proposto por Ziegler e Nichols em 1942 (SMITH e CORRIPIO, 2008). Este método de sintonia pode ser descrito por dois passos:

Passo 1. Determinação das características dinâmicas da malha de controle;

Passo 2. Estimação dos parâmetros de sintonia do controlador que produzem uma saída desejada, para atender uma característica do processo em particular.

Neste método as características dinâmicas do processo são representadas pelo ganho crítico de um controlador proporcional (K_{cu}) e o período crítico de oscilação (T_u) da malha de controle. Estes períodos e ganhos são obtidos experimentalmente da seguinte forma:

- No controle PID, os ganhos integral e derivativo, são considerados zero, tornando o controlador proporcional;
- O ganho proporcional é aumentado até a malha de controle oscilar com amplitude constante.

A Figura 5.2 ilustra a resposta da malha de controle, com o ganho e períodos críticos, K_{cu} , T_u , em que $c(t)$ representa a variável de processo a ser controlada.

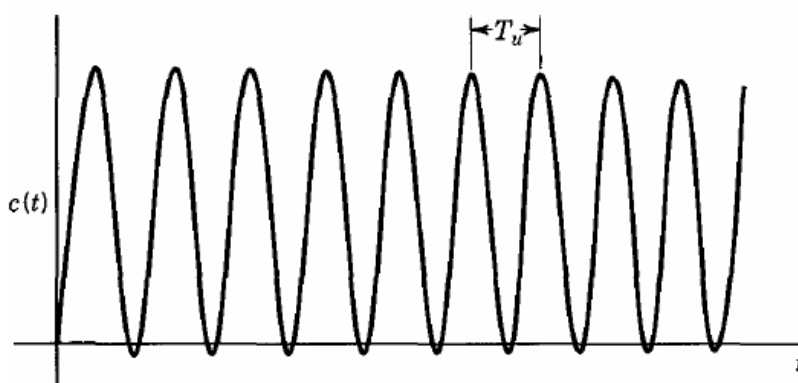


Figura 5.2 - Resposta da malha de controle com ganho e períodos críticos.

Com o valor do ganho crítico do ganho proporcional é possível encontrar os parâmetros de sintonia do controlador para as outras constantes, com o auxílio da Tabela 5.1.

Tabela 5.1 - Equações de Ziegler e Nichols segundo método.

Tipo de Controlador	K_c	T_i	T_d
P	$0,5.K_{cu}$	∞	0
PI	$0,45.K_{cu}$	$\frac{1}{1,2}.T_u$	0
PID	$0,6.K_{cu}$	$0,5.T_u$	$0,125.T_u$

O controlador PID foi aplicado ao modelo obtido com o uso das técnicas de identificação e os parâmetros do controlador foram obtidos por meio do método de sintonia de Ziegler-Nichols de ganho crítico, sendo obtidos os valores de: $K_{cu} = 20$ e $T_u = 0.8$. Desta forma, por meio do emprego das equações relativas a este método, foram obtidos os valores das constantes dos controladores mostradas na Tabela 5.2.

Tabela 5.2 - Constantes dos controladores obtidas utilizando o método de sintonia de Ziegler-Nichols.

Tipo de Controlador	K_c	T_i	T_d
PD	10		0,1
PI	9	0,666	
PID	12	0,4	0,1

Existe também um outro método de sintonia proposto por Ziegler e Nichols chamado de método da curva de reação (ASTRÖM e HÄGGLUND, 1995). Neste método é utilizada a resposta à perturbação a um degrau para encontrar os ganhos do controlador e sua vantagem está na aplicação em processos em que oscilações no sistema de malha fechada não podem ser toleradas.

5.2.3 Lógica *fuzzy*

Na aplicação de sistemas inteligentes na automação e controle de processos existem técnicas largamente difundidas como lógica *fuzzy* (difusa) e redes neurais, além de algoritmos genéticos, estratégias de busca adaptativas baseadas em modelos de evolução biológica (KUO e GOLNARAGHI, 2009).

Os sistemas *fuzzy* atualmente são uma realidade em diversas áreas do conhecimento, como: sistemas de energia, telecomunicações, robótica, processos químicos e industriais, medicina, entre outros. Os sistemas *fuzzy* são também largamente utilizados em sistemas de controle de processo, devido ao fato de que uma das características dos sistemas de controle é de possuírem: não-linearidades, parâmetros que variam com o tempo e erros inerentes à etapa de modelagem, sendo sujeitos a perturbações que não foram previstas no projeto do controlador. Assim, os controladores tradicionais podem não ser suficientes para atender estas exigências.

Como os sistemas de controle clássico são baseados em um modelo matemático segundo o qual o sistema de controle é descrito por meio de uma ou mais equações diferenciais que definem a resposta do sistema às entradas. Estes sistemas são produtos de décadas de análise teórica e desenvolvimento tecnológico, sendo altamente eficazes em uma vasta gama de aplicações. Entretanto, em vários casos, o modelo matemático do processo não existe, não é conhecido ou é muito complexo, como no caso da coluna de destilação. Nestas situações, um sistema baseado em regras empíricas como o *fuzzy* pode ser bem mais eficaz.

A lógica *fuzzy* está alicerçada em aspectos que caracterizam a imperfeição da informação, que são a imprecisão e a incerteza, sendo ferramentas essenciais para tratar de informações fornecidas por seres humanos. Foi desenvolvida a partir da teoria dos conjuntos *fuzzy*, de acordo com ZADEH (1965), que trata do aspecto vago da informação. A teoria descreve que um conjunto *fuzzy* específico (*crisp*) permite objetos que possuam graus de pertinência a determinados conjuntos, o que permite a representação de conceitos imprecisos sem perder o formalismo matemático. Desta forma, o uso de controladores baseados em lógica *fuzzy* torna-se uma alternativa interessante quando o modelo matemático de um determinado processo está sujeito a incertezas.

Além disso, a lógica *fuzzy* é apropriada para implementações de baixo custo computacional com sensores de baixo custo e utilizando conversores A/D de baixa resolução, sendo que estes podem ser aperfeiçoados adicionando-se regras para melhorar o desempenho ou melhorar sistemas de controle já existentes.

O projeto conceitual de controladores *fuzzy* é descrito da seguinte forma:

- o O estágio de entrada mapeia sensores e outras entradas (como chaves) para funções de pertinência apropriadas e valores verdade.
- o O estágio de processamento invoca as regras apropriadas e gera os resultados para cada uma, depois combina os resultados das regras.

- o Finalmente, o estágio de saída converte o resultado da combinação em um valor de saída preciso para controle.

Na Figura 5.3 está apresentada uma arquitetura representativa da lógica *fuzzy*.

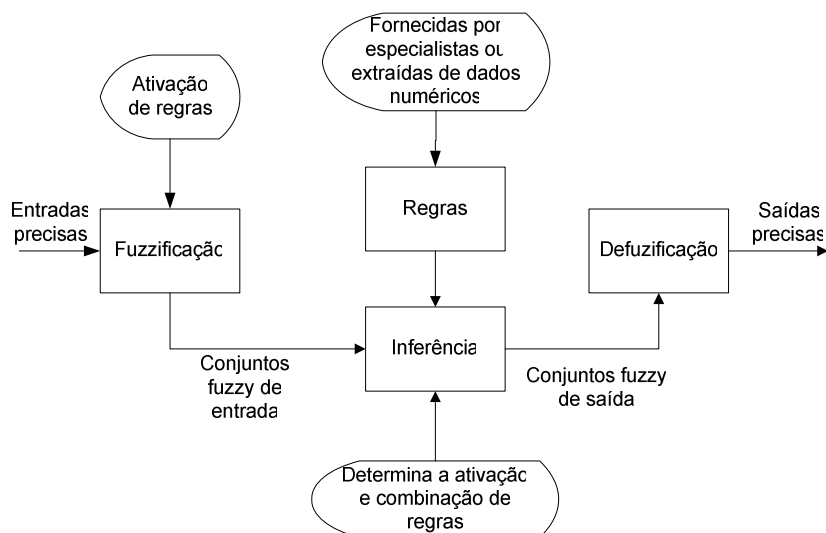


Figura 5.3 - Arquitetura representativa da lógica *fuzzy*.

O controlador *fuzzy* é um sistema composto de regras do tipo SE-Então, que definem ações de controle em função das diversas faixas de valores que as variáveis do processo podem assumir. Estas faixas de valores podem ser modeladas com o uso de funções de pertinência (*membership*). É importante enfatizar que não existe um procedimento formal de otimização geral para a determinação das funções de pertinência dos conjuntos *fuzzy* para um determinado modelo, sendo propostos diferentes modelos para resolver este problema.

Uma das alternativas para a determinação das funções de pertinência é a utilização de modelos neuro-*fuzzy*. Neste caso os parâmetros são obtidos pelo treinamento de uma rede neural adequada, que irá aproximar as funções que relacionam os sinais de entrada-saída (JANG et al., 1997).

5.2.3.1 Projeto de um controlador *fuzzy*

De acordo com PASINO e YURKOVICH (1997), é possível estabelecer uma metodologia formal para representar e manipular a heurística do conhecimento humano de como controlar um sistema. Um diagrama de blocos de um controlador *fuzzy* inserido em

uma malha de controle é mostrado na Figura 5.4. As saídas são chamadas de $y(t)$, as entradas $u(t)$ e o sinal de referência de entrada (*setpoint*) é chamado de $r(t)$.

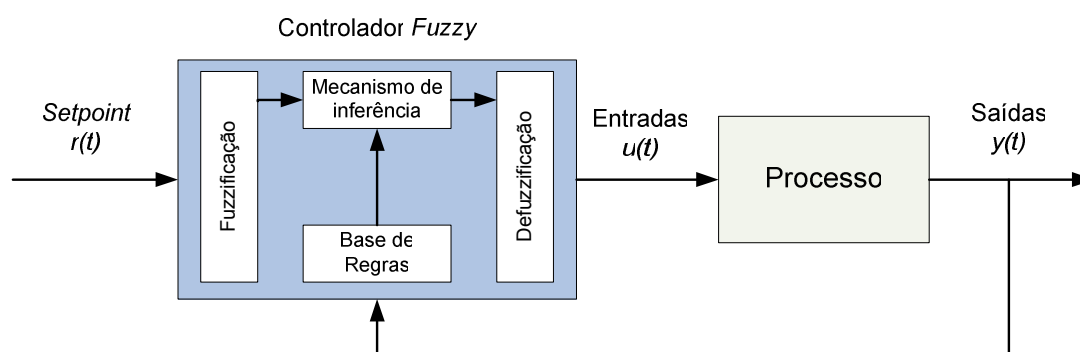


Figura 5.4 - Arquitetura de um controlador *fuzzy*.

O controlador *fuzzy* tem 4 componentes principais:

- 1- Base de regras, que possui o conhecimento na forma de um conjunto de regras de como melhor controlar o sistema;
- 2- O mecanismo de inferência, que avalia quais regras de controle são relevantes em um determinado tempo e escolhe a entrada para o processo;
- 3- A interface de fuzzificação, que modifica as entradas do mundo externo para que as mesmas possam ser interpretadas e comparadas com as regras na base de regras;
- 4- A interface de defuzzificação converte as informações do controle *fuzzy* para as entradas da planta.

Desta forma, pode-se considerar o controlador *fuzzy* como um elemento de controle que opera em um sistema de malha fechada em tempo real, coletando os dados de saída da planta e comparando-os com uma referência de entrada $r(t)$ e decidindo quais entradas da planta $u(t)$ devem assegurar os objetivos do controle requeridos.

Um sistema *fuzzy* é um mapeamento não linear entre entradas e saídas. Considera-se que o sistema *fuzzy* possui entradas $u_i \in U_i$, sendo $i = 1, 2, \dots, n$ e saídas $y_i \in Y_i$, sendo $i = 1, 2, \dots, m$, como é mostrado na Figura 5.5.

As entradas e saídas são números reais chamados de “*crisp*”. O bloco de fuzzificação converte as entradas *crisp* em conjuntos *fuzzy*. O mecanismo de inferência utiliza as regras *fuzzy* na base de regras para fornecer resultados para o controlador e o bloco de defuzzificação converte as informações *fuzzy* do controlador para as saídas reais (*crisp*).

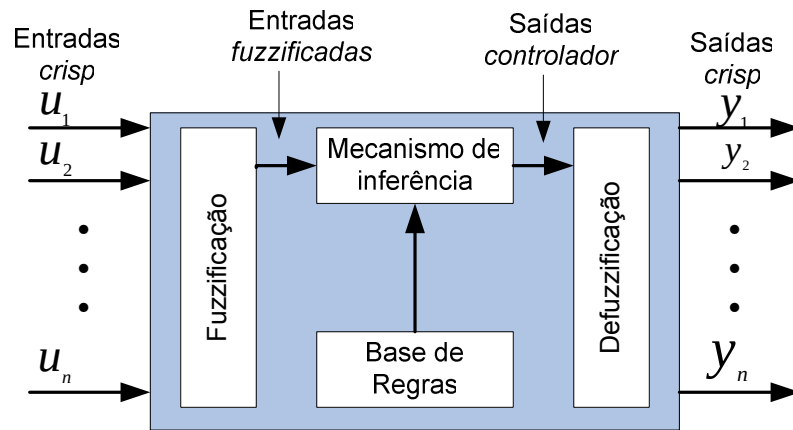


Figura 5.5 - Entradas e saídas do controlador *fuzzy*.

Os conjuntos U_i e Y_i são chamados de universos de discurso para u_i e y_i , respectivamente, sendo os conjuntos ou subconjuntos de números reais utilizados para as variáveis reais de entrada e de saída.

5.2.3.1.1 Variáveis linguísticas

Para especificar as regras que devem ser inseridas na base de regras, o projetista utiliza uma descrição linguística. Desta forma, expressões linguísticas são necessárias para descrever as características das entradas e das saídas. Desta forma, serão utilizadas variáveis linguísticas, que são descrições de constantes simbólicas para quantidades variáveis no tempo. As variáveis linguísticas são denominadas por $\tilde{u}_i$ para as entradas u_i e $\tilde{y}_i$ para as saídas y_i .

5.2.3.1.2 Valores lingüísticos

Da mesma forma que u_i e y_i fornecem valores para cada universo de discurso U_i e Y_i , respectivamente, as variáveis linguísticas $\tilde{u}_i$ e $\tilde{y}_i$ fornecem valores lingüísticos utilizados para descrever as características das variáveis. Considere-se $\tilde{A}_i^j$ para denotar o

j^{a} valor linguístico da variável $\tilde{u}_i$ definida no universo de discurso U_i . Desta forma, o conjunto de valores linguísticos é descrito como:

$$\tilde{A}_i = \{\tilde{A}_i^j : j = 1, 2, \dots, n_i\}$$

Os valores linguísticos são geralmente descritos como “muito grande”, “zero”, entre outros.

5.2.3.1.3 Regras linguísticas

O mapeamento das entradas em relação as saídas para um sistema *fuzzy* é em parte caracterizado por um conjunto de regras de *condição* $\rightarrow$ *ação* ou como condições (SE-ENTÃO), na seguinte forma:

Se premissa **Então** consequente

As entradas são associadas com as premissas e as saídas com o consequente. Para um sistema de múltiplas entradas e única saída (MISO), a forma da regra linguística é:

$$\text{Se } \tilde{u}_1 \text{ é } \tilde{A}_1^j \text{ e } \tilde{u}_2 \text{ é } \tilde{A}_2^k \text{ e } \dots \text{ e } \tilde{u}_n \text{ é } \tilde{A}_n^l \text{ então } \tilde{y}_q \text{ é } \tilde{A}_q^p$$

Este tipo de método de regras é utilizado largamente, sendo denominado método de inferência Mamdani.

5.2.3.1.4 Conjuntos *fuzzy*, lógica *fuzzy* e base de regras

Os conjuntos *fuzzy* e a lógica *fuzzy* são usados heurísticamente para quantificar o significado de variáveis, valores e regras linguísticas que são especificadas por especialistas.

5.2.3.1.5 Funções de pertinência

Considere-se que U_i e $A_i^j \in A_i$, descrevem um valor linguístico específico para a variável linguística $\tilde{u}_i$. A função $\mu(u_i)$ associada com $\tilde{A}_i^j$ que mapeia U_i para $[0,1]$ é chamada de função de pertinência.

Existem diversas formas de funções de pertinência possíveis, como, por exemplo, formas triangulares, gaussianas, trapezoidais, entre outras, sendo que cada uma fornece um diferente significado para os valores linguísticos que quantificam. Na Figura 5.6, é

apresentada uma ilustração gráfica de diferentes funções de pertinência. Para o controlador *fuzzy* aplicado no sistema de controle do trabalho desenvolvido na tese, é aplicada a função de pertinência triangular, cuja representação matemática é descrita na Tabela 5.3.

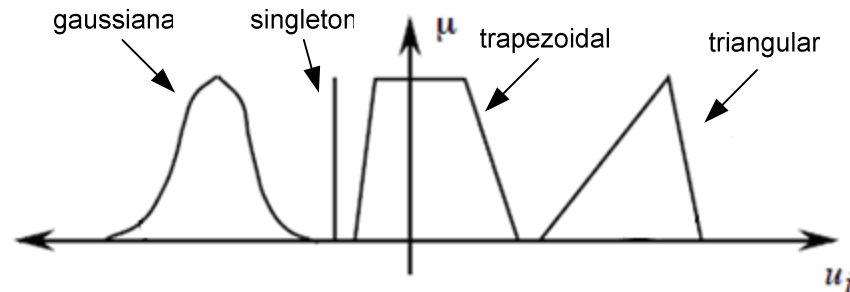


Figura 5.6 - Funções de pertinência típicas.

Tabela 5.3 - Caracterização matemática da função de pertinência triangular.

Porção da Função de Pertinência	Equação Descritiva
Esquerda	$\mu^L(u) = \begin{cases} 1 & \text{se } u \leq e^L \\ \max\left\{0, 1 + \frac{e^L - u}{0.5w^L}\right\} & \text{senão} \end{cases}$
Centro	$\mu^C(u) = \begin{cases} \max\left\{0, 1 + \frac{u - c}{0.5w}\right\} & \text{se } u \leq c \\ \max\left\{0, 1 + \frac{c - u}{0.5w}\right\} & \text{senão} \end{cases}$
Direita	$\mu^R(u) = \begin{cases} \max\left\{0, 1 + \frac{u - e^R}{0.5w^R}\right\} & \text{se } u \leq c^R \\ 1 & \text{senão} \end{cases}$

5.2.3.1.6 Fuzzificação

Os conjuntos *fuzzy* são utilizados para quantificar a informação na base de regras e o mecanismo de inferência opera de forma a produzir conjuntos *fuzzy*. Desta forma deve-se especificar como o sistema *fuzzy* irá converter as entradas numéricas $u_i \in U_i$ em conjuntos *fuzzy* em um processo chamado de fuzzificação.

Se U_i^* for considerado o conjunto de todos os possíveis conjuntos *fuzzy* que podem ser definidos por U_i , dado $u_i \in U_i$, a fuzzificação transforma u_i em um conjunto *fuzzy*

denominado por $\hat{A}_i^{fuz}$, definido no universo de discurso U_i . Esta transformação é produzida pelo operador de fuzzificação F definido por:

$$F : U_i \rightarrow U_i^*$$

Sendo: $F(u_i) = \hat{A}_i^{fuz}$

O mecanismo de inferência tem como premissas básicas: determinar a extensão para a qual cada regra é relevante para a situação atual como caracterizada pelas entradas u_i , $i=1,2,\dots,n$; e obter conclusões utilizando as entradas atuais u_i e a informação na base de regras (etapa de inferência).

5.2.3.1.7 Defuzzificação

Existe um grande número de estratégias de defuzzificação, sendo que cada uma fornece um meio de escolher uma única saída, chamada de y_q^{crisp} , baseada nos conjuntos *fuzzy* obtidos, sendo utilizado o método do centro de gravidade. Neste método y_q^{crisp} é obtido por meio do centro de área de cada conjunto *fuzzy* dado por:

$$y_q^{crisp} = \frac{\sum_{i=1}^R b_i^q \int y_q \mu_{\hat{B}_i^q}(y_q) dy_q}{\sum_{i=1}^R \int y_q \mu_{\hat{B}_i^q}(y_q) dy_q} \quad (5.5)$$

Sendo: R o número de regras, b_i^q o centro de área da função de pertinência de B_q^p associado com o conjunto *fuzzy* $\hat{B}_q^i$ para i -ésima regra ($j, k, \dots, l; p, q$) i , e

$$\int y_q \mu_{\hat{B}_i^q}(y_q) dy_q \quad (5.6)$$

descreve a área sobre $\mu_{\hat{B}_i^q}(y_q)$.

5.2.3.2 Estrutura dos controladores *fuzzy*

De acordo com LI (1997), de forma análoga ao controlador PID, para o controlador *fuzzy* também existem os controladores PI, PD e PID para a lógica *fuzzy*, que serão

chamados de *fuzzy*-PI, *fuzzy*-PD e *fuzzy*-PID, respectivamente. Na Figura 5.7, estão representados estes controladores.

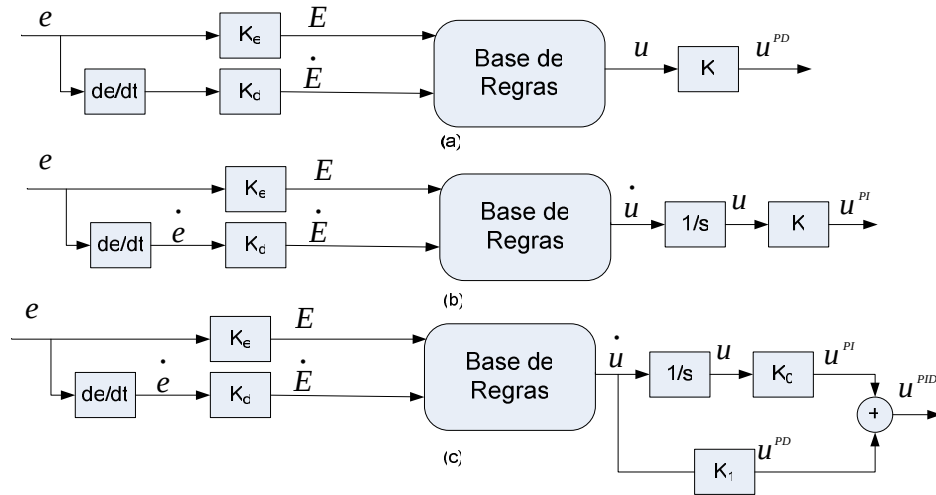


Figura 5.7 - Estrutura dos controladores (a) *fuzzy*-PD, (b) *fuzzy*-PI e (c) *fuzzy*-PID.

Para o projeto do controlador *fuzzy*, é necessário torná-lo não linear e substituir o controlador PID por um controlador linear *fuzzy*. As fontes de não-linearidades do controlador *fuzzy* são:

- **Base de regras:** A posição, forma e número de conjuntos *fuzzy*, bem como o escalonamento das entradas não linear causam transformações não lineares. As regras geralmente expressam uma estratégia de controle não-linear.
- **Mecanismo de Inferência:** Se os conectivos **and** e **or** são implementados como, por exemplo: min e max, respectivamente, são não lineares.
- **Defuzzificação:** Os diversos métodos de defuzzificação são não lineares.

5.2.3.3 Sintonia do controlador *fuzzy*

Devido ao fato dos controladores *fuzzy* serem não lineares, a tarefa de ajustar os ganhos de um controlador da mesma forma que para um controlador PID torna-se uma tarefa bastante complexa. JANTZEN (1998) propôs um procedimento de projeto e sintonia que transpõem a ideia de sintonia de um controlador PID convencional para um controlador PID tipo *fuzzy*. A ideia proposta é iniciar com um controlador PID convencional sintonizado e substituí-lo por um controlador não linear *fuzzy*.

Os controladores PID podem ser sintonizados de diferentes maneiras, que incluem sintonia manual, Zigler-Nichols, métodos analíticos, por otimização, colocação de pólos

(*pole placement*) ou autoajuste (ASTRÖM e HÄGGLUND, 1995). Desta forma, a tarefa de sintonia será mais trivial se for utilizado um destes métodos de sintonia e comparado o controlador sintonizado com um controlador tipo *fuzzy*. O procedimento para o ajuste do controlador *fuzzy*-PID é descrito na Figura 5.8.

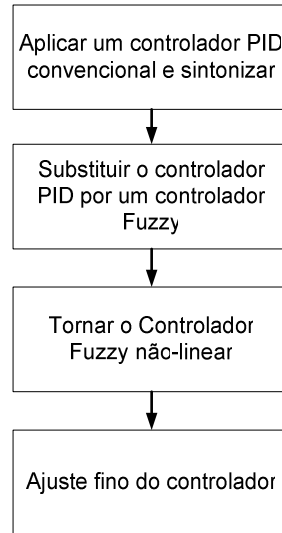


Figura 5.8 - Procedimento de sintonia de controlador *fuzzy*-PID.

A proposta deste procedimento é projetar um controlador PID convencional, estabilizar o sistema em malha fechada e sintonizá-lo para uma performance satisfatória. Com este controlador e o modelo da planta linear é possível projetá-lo utilizando os sólidos conceitos da teoria de controle linear e movê-los para o domínio *fuzzy*.

De acordo com o procedimento de sintonia proposto por LI (1997), os ganhos dos controladores são dados por:

$$\text{Fuzzy-PD} \quad K=K_c, \quad \alpha \approx T_d \quad (5.7)$$

$$\text{Fuzzy-PI} \quad K \geq K_c \quad \alpha \approx T_i \quad (5.8)$$

$$\text{Fuzzy-PID} \quad \alpha + \beta \approx T_i, \quad \beta = \frac{K_1}{K_0} \quad (5.9)$$

$$\alpha\beta \approx T_i.T_d \quad (5.10)$$

$$K_0 \geq \frac{K_p}{T_i} \quad (5.11)$$

5.2.4 Lógica neuro-fuzzy

A lógica neuro-fuzzy é uma tecnologia da inteligência artificial que gera regras diretamente de dados numéricos e associa probabilidades a estas regras. A lógica neuro-fuzzy combina a lógica fuzzy com o aprendizado e capacidades adaptativas das redes neurais artificiais.

Uma rede neuro-fuzzy é um sistema que usa um algoritmo de aprendizado derivado ou inspirado em teoria de redes neurais artificiais para determinar seus parâmetros, conjuntos e regras fuzzy, obtidos por meio do processamento de exemplos de dados.

Em redes neuro-fuzzy os pesos da conexão, a propagação e as funções de ativação diferem de redes neurais comuns. O termo sistema neuro-fuzzy é usado para denominar sistemas neuro-fuzzy baseados em um sistema fuzzy treinado por um algoritmo de aprendizagem derivada de uma teoria de redes neurais artificiais.

O procedimento de aprendizagem (heurística) opera em informações locais, e causa apenas modificações locais no fundamento do sistema fuzzy. Um sistema neuro-fuzzy pode ser visto como uma rede neural de três camadas:

- Camada 1: representa as variáveis de entrada;
- Camada 2: representa as regras fuzzy;
- Camada 3: representa as variáveis de saída.

Os conjuntos fuzzy são codificados como pesos de conexões. Os sistemas neuro-fuzzy combinam os atributos positivos destas abordagens produzindo sistemas fuzzy com habilidade de aprender e se adaptar ao mundo real. Estes sistemas são ideais para aplicações como modelagem e controle de processos dinâmicos em tempo real.

Um sistema neuro-fuzzy é composto de componentes convencionais de um sistema fuzzy, exceto que a computação de cada estágio é realizada por uma camada de neurônios ocultos e a capacidade de aprendizagem da rede neural possibilita o melhoramento do sistema de conhecimento. Existem inúmeras arquiteturas de sistemas neuro-fuzzy. Uma possível arquitetura de um sistema neuro-fuzzy é mostrada na Figura 5.9

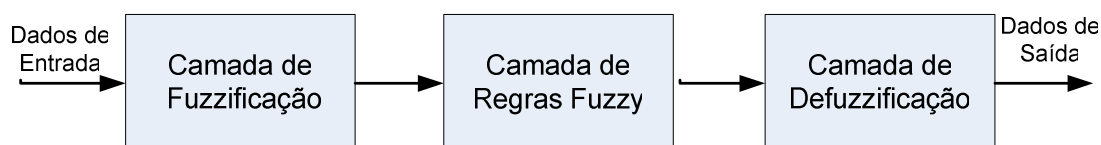


Figura 5.9 - Diagrama representativo de um sistema neuro-fuzzy

. O sistema contém três diferentes camadas, sendo elas:

- Camada de Fuzzificação;
- Camada de Inferência (Regras *fuzzy*);
- Camada de Defuzzificação.

Na camada de Fuzzificação cada neurônio representa uma função "*membership*" de entrada do antecedente de uma regra *fuzzy*. Na camada de Inferência as regras são ativadas e os valores ao final de cada regra representam o peso inicial da regra, e serão ajustados ao seu próprio nível ao final do treinamento. Na camada de Defuzzificação cada neurônio representa uma proposição consequente e suas funções *membership* podem ser projetadas combinando uma ou duas funções *sigmoid* e lineares.

O peso de cada saída representa, aqui, o centro de gravidade de cada saída da função *membership*. Após adquirir a saída correspondente, o ajuste é feito na conexão do peso e nas funções *membership* visando compensar o erro e produzir um novo sinal de controle. A estrutura da rede neuro-*fuzzy* foi projetada com a finalidade de se obter melhores resultados do que aqueles obtidos com o uso de uma rede neural convencional. A estrutura de rede neuro-*fuzzy* utiliza um algoritmo de aprendizado híbrido para identificar parâmetros de sistemas de inferência nebulosa do tipo Takagi-Sugeno. Este algoritmo facilita o processo de computação do vetor gradiente que relaciona a redução de uma função erro para uma modificação nos parâmetros de um FIS (*Fuzzy Inference System*). Assim que o vetor gradiente é obtido, um número de rotinas de otimização pode ser aplicado para reduzir o erro entre a saída real e a obtida. A Figura 5.10 representa a estrutura de um sistema neuro-*fuzzy*.

5.2.4.1 ANFIS

O sistema neuro-*fuzzy* escolhido foi o sistema criado por J. S. Roger Jang denominado ANFIS (*Adaptive Network-based Fuzzy Inference System*) que faz uma adaptação dos valores de entrada e saída para uma base de regra que interliga todas as entradas e saídas, formando assim uma base de regras robusta que cria um sistema de inferência *fuzzy* que contempla todas as possíveis entradas. Os sistemas neuro-*fuzzy* podem ser analisados como sistema de inferência *fuzzy*, implementados sob a arquitetura das redes neurais artificiais. O objetivo fundamental é obter as vantagens da lógica nebulosa no que diz respeito ao processo de raciocínio, e com a capacidade de aprendizagem das redes neurais. Desta forma o ANFIS é considerado uma poderosa ferramenta de modelagem e

controle de sistemas, ideal para processos com dinâmica complexas demais para serem controlados por meio de técnicas convencionais de controle (HAYKIN, 1994).

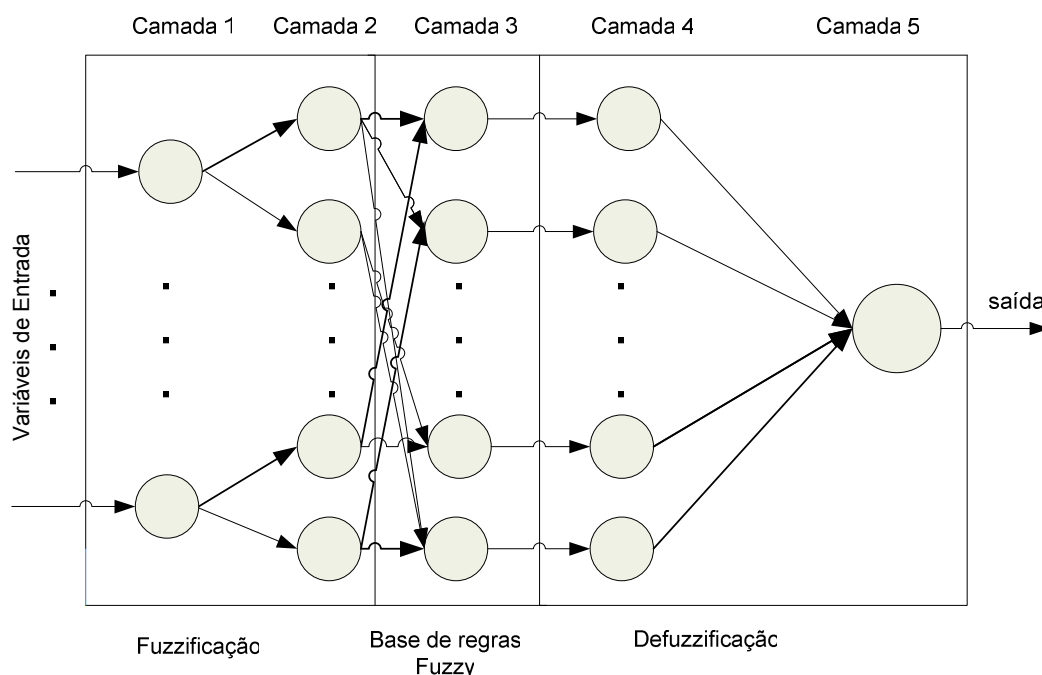


Figura 5.10 - Estrutura de um sistema neuro-fuzzy.

5.2.4.1.1 Arquiteturas ANFIS e algoritmos de aprendizagem

Segundo SU e MCAVOY (1993), a arquitetura e o procedimento de aprendizagem de uma rede adaptativa são um conjunto de redes neurais *feedforward* com capacidade de aprendizagem supervisionada. Como uma rede adaptativa, a estrutura da rede é composta de nós e de *links* direcionais por meio dos quais os nós são interligados. Parte dos nós é adaptativa, o que significa que suas saídas dependem dos parâmetros pertinentes destes nós e a regra de aprendizagem especifica como estes parâmetros devem ser alterados para minimizar um erro de medição definido.

A regra básica de aprendizagem de redes adaptativas é baseada no gradiente descendente e na regra da cadeia (WERBOS, 1974).

Uma rede adaptativa é uma rede *feedforward* multicamadas na qual cada nó executa uma determinada função nos sinais de entrada, bem como um conjunto de parâmetros contidos em cada nó. As equações para cada função executada pelo nó podem variar de nó para nó e a escolha de cada função do nó depende da função global de entrada-saída que a rede adaptativa exerce. As ligações de uma rede adaptativa somente indicam a direção do

fluxo dos sinais entre os nós, não sendo associados pesos com os *links*. Uma representação de uma rede adaptativa é mostrada na Figura 5.11.

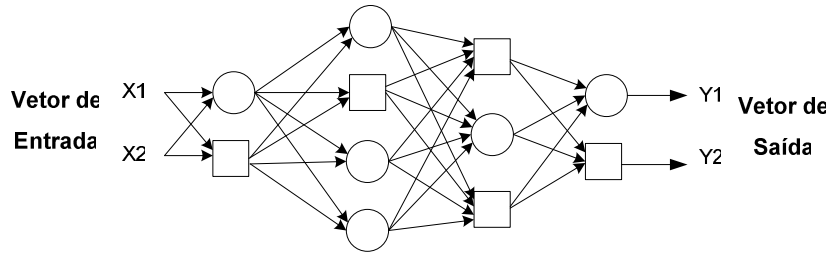


Figura 5.11 - Representação de uma rede adaptativa.

Para descrever capacidades adaptativas diferentes serão utilizados nós em formatos cilíndricos e quadrados. O nó em formato quadrado representa o nó adaptativo e o nó em formato cilíndrico representa o nó fixo. O conjunto de parâmetros em uma rede adaptativa é a união dos conjuntos de parâmetros de cada nó adaptativo. Com o objetivo de descrever uma função entre as entradas e saídas, estes parâmetros são atualizados de acordo com um conjunto de dados de treinamento especificados em um procedimento de treinamento baseado no gradiente.

Se for considerada uma dada rede adaptativa com L camadas, a n -ésima camada possui $\#(k)$ nós. Pode-se denotar o nó contido na posição k^n por (k, i) e a função de saída do nó por O_i^k . Como a saída do nó depende do sinal de entrada e de seu conjunto de parâmetros, tem-se:

$$O_i^k = O_i^k(O_1^{k-1}, \dots, O_{\#(k-1)}^{k-1}, a, b, c, \dots) \quad (5.12)$$

Sendo: a, b, c , etc., os parâmetros pertencentes a este nó.

Considerando um dado conjunto de dados de treinamento, com P elementos, pode-se definir o erro de medida (ou função de energia) para p ($1 \leq p \leq P$) para os dados de entrada como a soma dos erros quadráticos:

$$E_p = \sum_{m=1}^{\#(L)} (T_{m,p} - O_{m,p}^L)^2 \quad (5.13)$$

Sendo $T_{m,p}$ a n -ésima componente de saída do vetor p de entrada e $O_{m,p}^L$ a n -ésima componente do atual do vetor de saída produzida pelo vetor p de entrada. Desta forma, o erro global é dado por:

$$E = \sum_{p=1}^p E_p \quad (5.14)$$

O procedimento de aprendizagem é obtido por meio da aplicação do gradiente descendente em E sobre o espaço de parâmetros. Primeiramente, calcula-se a taxa de erro ($\partial E_p / \partial O$) para p dados de treinamento para cada nó de saída O . A taxa de erro para o nó de saída em (L, i) pode ser calculada por:

$$\frac{\partial E_p}{\partial O_{i,p}^L} = -2(T_{i,p} - O_{i,p}^L) \quad (5.15)$$

Para um nó interno em (k,i) a taxa de erro pode ser obtida pela regra da cadeia:

$$\frac{\partial E_p}{\partial O_{i,p}^k} = \sum_{m=1}^{\#(k+1)} \frac{\partial E_p}{\partial O_{m,p}^{k+1}} \frac{\partial O_{m,p}^{k+1}}{\partial O_{i,p}^k} \quad (5.16)$$

Sendo: $1 \leq k \leq L-1$.

Assim, considera-se que a taxa de erro de um nó interno, pode ser expressa como uma combinação linear das taxas de erro dos nós das camadas anteriores. Desta forma, para todo $1 \leq k \leq L$ e $1 \leq i \leq \#(k)$, pode-se encontrar $\partial E_p / \partial O_{i,p}^k$ por (5.15) e (5.16).

Ao utilizar o método do gradiente para minimizar o erro de medição, é necessário, primeiramente calcular o gradiente do vetor. Para tanto, considera-se que uma mudança em um determinado parâmetro irá causar a mudança na saída do nó contendo o referido parâmetro, que por sua vez irá causar uma mudança na saída da última camada, alterando a medida do erro de saída.

Se α for considerado um parâmetro de uma dada rede adaptativa, tem-se:

$$\frac{\partial E_p}{\partial \alpha} = \sum_{O^* \in S} \frac{\partial E_p}{\partial O^*} \frac{\partial O^*}{\partial \alpha} \quad (5.17)$$

Sendo S o conjunto de nós nos quais a saída depende de α . Então, a derivada do erro medido E , com respeito a α é:

$$\frac{\partial E_p}{\partial \alpha} = \sum_{p=1}^P \frac{\partial E_p}{\partial \alpha} \quad (5.18)$$

A taxa de aprendizagem η_a pode ser expressa como:

$$\eta_a = \frac{k}{\sqrt{\sum \alpha \left(\frac{\partial E}{\partial \alpha} \right)^2}} \quad (5.19)$$

Sendo k o tamanho do passo de cada transição do gradiente no espaço de parâmetros, o valor de k pode ser variado para alterar a taxa de convergência.

5.3 Arquitetura ANFIS

Para descrever a estrutura do ANFIS, considera-se a arquitetura baseada no modelo Takagi-Sugeno (TS) de primeira ordem, mostrada na Figura 5.12.

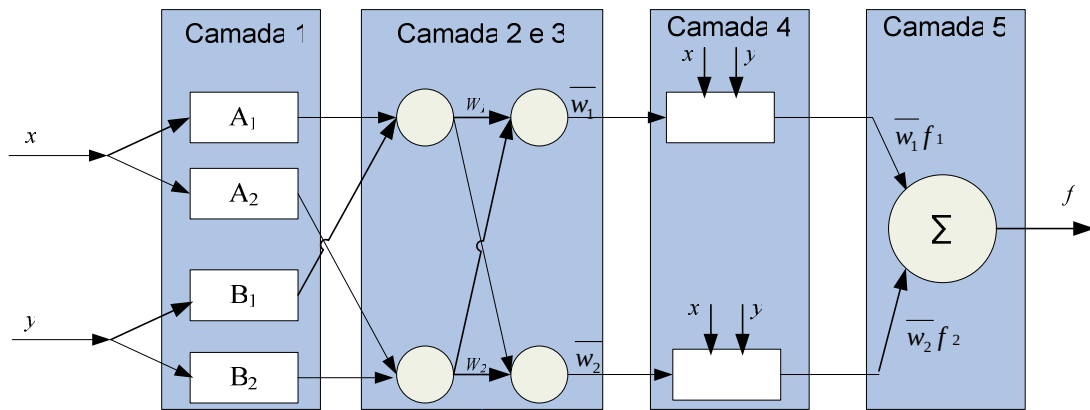


Figura 5.12 – Representação da estrutura do ANFIS.

Sendo consideradas duas variáveis de entrada x e y , uma saída f e as seguintes regras:

Regra 1: Se x é A_1 e y é B_1 Então $f_1 = p_1x + q_1y + r_1$

Regra 2: Se x é A_2 e y é B_2 Então $f_2 = p_2x + q_2y + r_2$

A Figura 5.13 ilustra um exemplo de regras para o modelo *fuzzy* Takagi-Sugeno de primeira ordem.

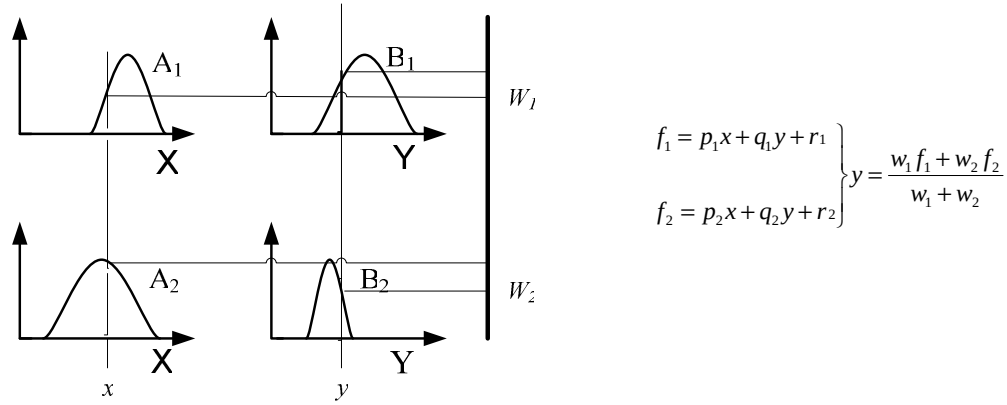


Figura 5.13 – Exemplo de regras para o modelo *fuzzy* Takagi-Sugeno de primeira ordem.

Sabendo que o modelo TS é a base da estrutura ANFIS, é sugerido que os parâmetros p , q e r sejam identificados pelo método de estimação de acordo com os seguintes passos:

Camada 1 - Calcula o grau de pertinência com que as entradas satisfazem os valores ou termos linguísticos associados a estes nós. Cada nó i desta camada é um nó que realiza a seguinte função:

$$O_i^1 = \mu A_i(x) \quad (5.20)$$

Sendo x a entrada do nó i e A_i a variável linguística associada com a função do nó. Assim, pode-se dizer que O_i^1 é a função de pertinência de A_i e especifica o grau no qual um dado x satisfaz A_i . Para este caso, uma das funções mais utilizadas para $\mu A_i(x)$ é a *bell-shaped* com máximo igual a 1 e mínimo igual a 0, dada por:

$$\mu A_i(x) = \exp \left\{ - \left(\frac{x - c_i}{a_i} \right)^2 \right\} \quad (5.21)$$

Sendo a_i e c_i o conjunto de parâmetros.

À medida que os valores destes parâmetros se alteram a função *bell-shaped* varia exibindo várias formas de função de pertinência nos rótulos linguísticos. Qualquer função contínua e diferenciável pode ser utilizada para função de nó nesta camada, sendo os parâmetros nesta camada chamados de parâmetros de premissa.

Camada 2 - Cada nó desta camada corresponde a uma regra, calculando com que grau o consequente da regra está sendo atendido, ou seja, são as implicações das premissas. Cada um dos blocos multiplica os sinais de entrada e envia o produto para a saída, representado pela equação:

$$w_i = \mu A_i(x) \times \mu B_i(y), \quad i = 1, 2. \quad (5.22)$$

Camada 3 - Esta camada realiza uma normalização dos valores da camada anterior.

$$\overline{w_i} = \frac{w_i}{w_1 + w_2} \quad (5.23)$$

Camada 4 – Nesta camada as saídas dos neurônios são calculadas de acordo com a seguinte função:

$$O_i^4 = \overline{w_i} f_i = \overline{w_i} (p_i x + q_i y + r_i) \quad (5.24)$$

Sendo $\overline{w_i}$ a saída da camada 3 e $\{p_i, q_i, r_i\}$ o conjunto de parâmetros. Os parâmetros desta camada são chamados de consequentes.

Camada 5 – Os nós desta última camada calculam a saída da ANFIS (f), dada pela saída global como a soma de todos os sinais de entrada:

$$O_1^5 = \sum_i \overline{w_i} f_i = \frac{\sum_i w_i f_i}{w_i} \quad (5.25)$$

Assim, esta estrutura pode ser treinada por qualquer mecanismo de aprendizagem empregado nas redes neurais artificiais.

A parte central do algoritmo de aprendizagem concentra-se em como obter um vetor gradiente no qual cada parâmetro é definido como a derivada de um erro de medida em relação a um parâmetro. O procedimento de encontrar um vetor gradiente é baseado em uma estrutura de rede chamado de retropropagação, pois o gradiente do vetor é calculado na direção oposta do fluxo de saída de cada nó. Uma vez o gradiente obtido, um número de técnicas de otimização baseadas em derivadas e técnicas de regressão são disponíveis.

Uma classe de métodos baseados em métodos de gradiente pode ser aplicada para otimização de modelos neuro-*fuzzy* não lineares, como os métodos de *steepest descent* e gradiente conjugado (JANG et al., 1997). Para o trabalho desenvolvido foi utilizado o algoritmo de aprendizagem híbrido, descrito a seguir.

5.3.1 Algoritmo de aprendizagem híbrido

O algoritmo de aprendizagem híbrido é descrito em duas etapas, caminho direto e caminho de retropropagação (ENGIN et al., 2004):

a) Passagem direta: No caminho direto de um algoritmo de aprendizagem híbrido, os nós de saída são avaliados diretamente até a camada 4 e os consequentes são identificados pelo método dos mínimos quadrados. Quando os valores dos parâmetros das premissas são fixados, a saída geral pode ser expressa como uma combinação linear dos parâmetros dos consequentes.

$$\begin{aligned} f &= \frac{w_1}{w_1 + w_2} f_1 + \frac{w_2}{w_1 + w_2} f_2 \\ &= \bar{w}_1 f_1 + \bar{w}_2 f_2 \\ &= (\bar{w}_1 x) p_1 + (\bar{w}_1 y) q_1 + (\bar{w}_1) r_1 + (\bar{w}_2 x) p_2 + (\bar{w}_2 y) q_2 + (\bar{w}_2) r_2 \end{aligned} \quad (5.26)$$

A equação (5.26) é linear nos parâmetros consequentes p_1, q_1, r_1, p_2, q_2 e r_2 , assim:

$$f = XW \quad (5.27)$$

Se a matriz for inversível, então:

$$W = X^{-1} f$$

b) Passagem por retropropagação: Nesta etapa os sinais de erro se propagam para trás e os parâmetros das premissas são atualizados pelo gradiente descendente.

$$a_{ij}(t+1) = a_{ij}(t) - \eta \cdot \frac{\partial E}{\partial a_{ij}} \quad (5.28)$$

A regra da cadeia é utilizada para calcular as derivadas parciais utilizadas para atualizar os parâmetros das funções de pertinência.

$$\frac{\partial E}{\partial a_{ij}} = \frac{\partial E}{\partial f} \cdot \frac{\partial f}{\partial f_i} \cdot \frac{\partial f_i}{\partial w_i} \cdot \frac{\partial w_i}{\partial \mu_{ij}} \cdot \frac{\partial \mu_{ij}}{\partial a_{ij}} \quad (5.29)$$

As derivadas parciais são definidas como:

$$E = \frac{1}{2}(f - f^i)^2 \text{ assim, } \frac{\partial E}{\partial f} = (f - f^i) = e \quad (5.30)$$

$$f = \sum_{i=1}^n f_i \text{ assim, } \frac{\partial f}{\partial f_i} = 1 \quad (5.31)$$

$$f_i = \frac{w_i}{\sum_{i=1}^n w_i} (p_i x + q_i y + r_i) \text{ assim, } \frac{\partial f_i}{\partial w_i} = \frac{(p_i x + q_i y + r_i) - f}{\sum_{i=1}^n w_i} \quad (5.32)$$

$$w_i = \prod_{j=1}^m \mu A_{ji} \text{ assim, } \frac{\partial w_i}{\partial \mu_{ij}} = \frac{w_i}{\mu_{ij}} \quad (5.33)$$

A última derivada parcial depende do tipo de função de pertinência utilizada. O gradiente é então obtido:

$$\frac{\partial E}{\partial a_{ij}} = e \frac{(p_i x + q_i y + r_i) - f}{\sum_{i=1}^n w_i} \frac{w_i}{\mu A_{ij}} \frac{\partial \mu A_{ij}}{\partial a_{ij}} \quad (5.34)$$

A Tabela 5.4 resume a atividade de cada uma das etapas.

Tabela 5.4 – Etapas relativas ao algoritmo de aprendizagem híbrido da ANFIS.

	Passagem direta	Passagem por retropropagação
Parâmetros de premissa	Fixos	Gradiente descendente
Parâmetros consequentes	Métodos dos mínimos quadrados	Fixos
Sinais envolvidos	Saídas dos nós	Sinais de Erros

5.4 Índices de Desempenho das Malhas de Controle

Em sistemas de controle é fundamental utilizar-se de índices para medida de desempenho das malhas de controle. Segundo OGATA (2003), geralmente, os elementos básicos que medem o desempenho de uma malha de controle são estabelecidos em função da resposta em regime transiente do sistema mediante a aplicação de um degrau unitário na referência (*setpoint*), devido ao fato de este ser o método mais simples e fácil de ser implementado. Na Figura 5.14, estão apresentados os índices empregados para medir o desempenho de uma malha de controle:

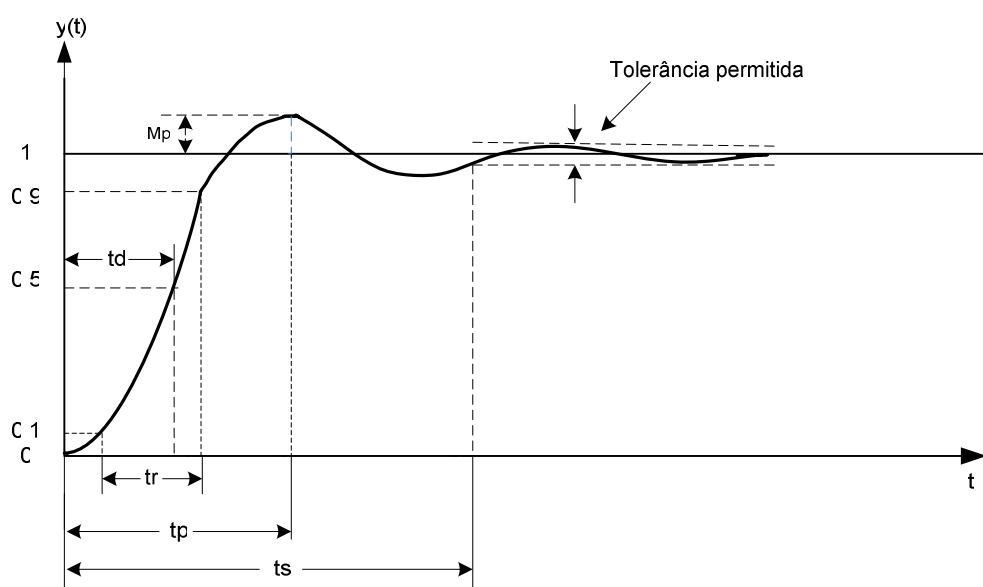


Figura 5.14 – Índices para avaliação de desempenho de malhas de controle.

-Tempo de atraso, t_d (delay time): tempo transcorrido pela resposta do sistema para alcançar 50% do valor final;

-Tempo de subida, t_r (rise time): é o tempo necessário para a resposta passar de 10% a 90% do seu valor final;

-Instante do pico, t_p (peak time): tempo necessário para a resposta atingir o primeiro pico do sobre-sinal (*overshoot*) máximo;

-Sobre-sinal (*overshoot*) máximo (M_p): valor máximo da resposta, a partir do valor unitário da resposta, sendo definido pela equação:

$$M_p = \frac{c(t_p) - c(\infty)}{c(\infty)} * 100 \quad (5.35)$$

-Tempo de acomodação, t_s (settling time): é o tempo transcorrido para a curva de resposta alcançar e permanecer dentro de uma faixa em torno do valor final, especificada em uma porcentagem do valor absoluto final entre 2 e 5%.

5.4.1 Análise de desempenho dos controladores sujeitos a perturbação na carga

Para avaliar a performance das malhas de controle mediante perturbação na carga podem ser considerados diversos critérios. Os critérios baseados na integração do erro do sinal de saída são os mais utilizados para análise de desempenho de malhas de controle submetidas a perturbação na carga são: *ISE*, *IAE*, *ITAE* (OGATA, 2003).

5.4.1.1 Integral do erro absoluto (*IAE*)

Corresponde a integral do valor absoluto do sinal do erro no tempo, sendo equivalente à soma das áreas formadas entre a curva da variável de processo e a curva do *setpoint*. É considerado um critério de parâmetros intermediários, e assim a sua resposta estará entre os critérios *ISE* e *ITAE* apresentando, por consequência, uma resposta também intermediária aos dois critérios.

$$IAE = \int_0^{\infty} [e(t)] dt \quad (5.36)$$

5.4.1.2 Integral do erro quadrático (*ISE*)

Este critério é baseado no erro ao quadrado, desta forma quanto maior o erro, maior o termo somado na integral. Neste critério, à medida que se aumenta o tempo, a tendência é de ocorrer uma diminuição no erro. Deste forma, a análise utilizando este critério leva em consideração os erros maiores que ocorrem no início da perturbação.

$$ISE = \int_0^{\infty} [e(t)]^2 dt \quad (5.37)$$

5.4.1.3 Integral do erro absoluto ponderado no tempo (ITAE)

Baseia-se na ponderação do erro durante o tempo transcorrido. Desta forma, quanto maior o tempo, maior o produto do valor absoluto do erro versus tempo ($t \cdot |e(t)|$). Neste critério, no início da perturbação como o erro tende a ser maior, é feita a ponderação no tempo. Desta forma, o termo somado na integral tende a ser pequeno, fazendo com que quanto maior o tempo, maior o termo somado na integral do erro. Este critério irá considerar com maior peso os erros que persistem por um grande período de tempo.

$$ITAE = \int_0^{\infty} t \cdot |e(t)| dt \quad (5.38)$$

Para integração considera-se desde $t=0$, ponto em que ocorre o início da perturbação e o tempo infinito (∞) seria o tempo em que ocorre a estabilização da resposta final. Estes critérios de erro (*IAE*, *ITAE* e *ISE*) são utilizados para evitar a anulação das parcelas de erro relativas ao tempo integral onde ocorre uma resposta oscilatória, alternando entre valores positivos e negativos de erro. A ocorrência da perturbação é a situação de maior erro, sendo que o mesmo tende a se reduzir durante o decorrer da resposta do sistema.

5.5 Simulação das Estratégias de Controle

Nas seções a seguir serão feitas as simulações das estratégias de controle, aplicadas ao modelo obtido a partir da identificação realizada por meio de redes neurais artificiais como mostra a Figura 5.15.

A partir das simulações com o modelo identificado será possível obter valores das constantes dos controladores antes de se iniciar os testes na coluna de destilação.

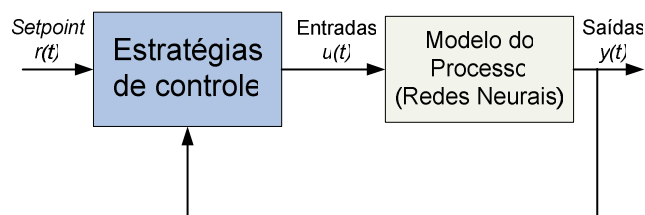


Figura 5.15 – Simulação das estratégias de controle mediante modelo obtido via identificação.

5.5.1 Parametrização do controlador *fuzzy*

Nas seções a seguir será descrita a etapa relativa a simulação do controlador *fuzzy* aplicada ao modelo obtido por meio da identificação. Também são apresentadas as etapas necessárias para o desenvolvimento do controlador *fuzzy*, bem como a etapa de simulação e apresentação dos resultados obtidos em comparação com um controlador PID convencional.

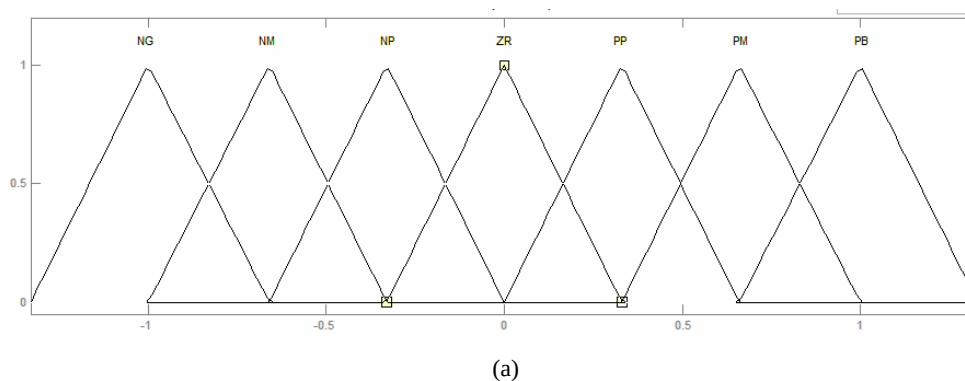
5.5.1.1 Funções de pertinência

Para iniciar o projeto do controlador *fuzzy*, foi escolhida a função de pertinência triangular, sendo que o ponto de cruzamento entre estas funções é aquele em que as funções de pertinência possuem 50% do valor final ($\mu = 0,5$).

Para a determinação dos universos de discurso de cada função de pertinência do controlador *fuzzy*, foi feita uma minuciosa observação do comportamento dinâmico da variável de processo (temperatura do prato de topo) sujeita à perturbações na variável manipulada (temperatura do refeedor).

5.5.1.2 Determinação de regras

As regras foram definidas e aplicadas no *Rule Editor* do *Toolbox Fuzzy* do software MATLABTM. As regras utilizadas tanto para o erro quanto para a variação no erro foram: grande positiva “PG”, média positiva “PM”, pequena positiva “PP”, zero “Z”, pequena negativa “NP”, média negativa “NM” e grande negativa “NG”. Seguindo o procedimento sugerido por LI (1997), são obtidos os seguintes intervalos para as variáveis de entrada, erro [-1,1] e variação de erro [-0.01, 0.01], e de saída [-1,1], como mostram as Figuras 5.16 (a), (b) e (c), respectivamente.



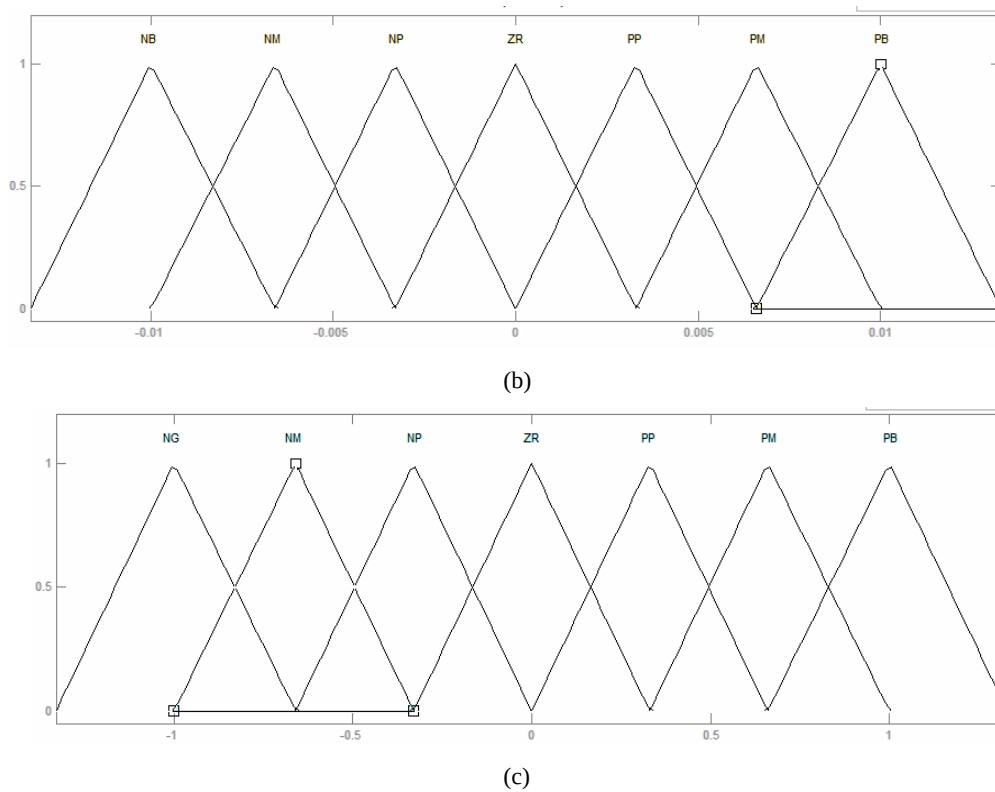


Figura 5.16 - Funções de pertinência para as variáveis (a) erro (b) variação do erro (c) saída.

As regras são combinadas em uma matriz, resultando em 49 diferentes regras, que são apresentadas na Tabela 5.5. Com base nestas regras, foi possível obter o diagrama de superfície de resposta para a variável de saída, obtida a partir da matriz de base de regras que tem como entradas, o erro e a variação do erro, ilustrado na Figura 5.17.

Tabela 5.5 - Base de regras para controlador *fuzzy*.

	Δerro						
Erro	NG	NM	NS	Z	PP	PM	PG
PG	Z	NP	NM	NG	NG	NG	NG
PM	PP	Z	NS	NM	NG	NG	NG
PP	PM	OS	Z	NS	NM	NG	NG
Z	PG	PM	PP	Z	NP	NM	NG
NP	PG	PG	PM	PP	Z	NP	NM
NM	PG	PG	PG	PM	PP	Z	NP
NG	PG	PG	PG	PG	PM	PP	Z

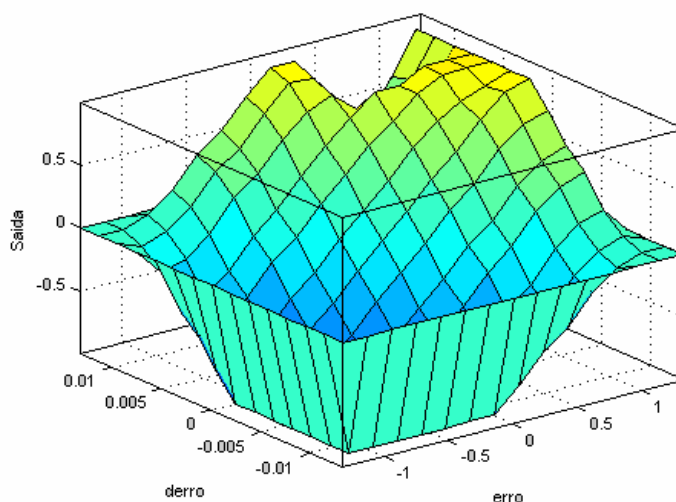


Figura 5.17 - Diagrama de superfície da resposta do controlador *fuzzy*.

5.5.1.3 Sintonia dos controladores *fuzzy*

Utilizando os valores das constantes dos controladores PI, PD e PID, obtidos pela sintonia por meio do método de Ziegler-Nichols e as equações (5.7, 5.8, 5.9, 5.10 e 5.11), foram obtidos os valores das constantes para os controladores *fuzzy* mostradas na Tabela 5.6.

Tabela 5.6 - Valores iniciais das constantes dos controladores *fuzzy*.

	Ke	α	K	β	Kd	K_0	K_1
<i>Fuzzy</i> -PD	1	0,1	10	-	0,1	-	-
<i>Fuzzy</i> -PI	1	0,66	9	-	0,66	-	-
<i>Fuzzy</i> -PID	1	0,1	12	0,3	0,1	30	9

Os valores apresentados na Tabela 5.6 são os parâmetros iniciais de sintonia. No entanto, para se fazer o ajuste fino dos controladores foi aplicada uma pequena variação nestes valores e observada a resposta do sistema, sendo obtidos os seguintes valores finais apresentados na Tabela 5.7.

Tabela 5.7 - Valores finais das constantes dos controladores *fuzzy*.

	Ke	α	K	β	Kd	K_0	K_1
<i>Fuzzy</i> -PD	2,4	0,1	10	-	0,24	-	-
<i>Fuzzy</i> -PI	3,6	0,66	9	-	2,376	-	-
<i>Fuzzy</i> -PID	1,7	0,1	12	0,3	0,17	30	9

5.5.1.4 Parametrização do controlador neuro-fuzzy

Os dados utilizados para parametrização do controlador neuro-fuzzy foram obtidos por meio da simulação de um controlador PID, sintonizado por meio do método de Ziegler-Nichols. Desta forma, o controlador neuro-fuzzy é sintonizado por meio de três entradas distintas: erro, variação do erro e integral do erro, sendo considerada a variável de saída a resposta do controlador PID, como mostra a Figura 5.18.

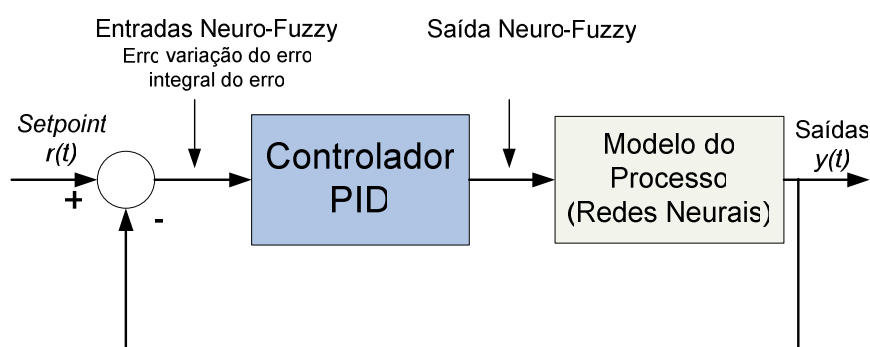


Figura 5.18 - Descrição das entradas e saídas do controlador neuro-fuzzy.

O intervalo de coleta dos dados é de 0,01s, sendo o tempo total de simulação igual a 1000s. Os dados armazenados são utilizados para a obtenção do modelo matemático do processo por meio de uma estratégia adaptativa neuro-fuzzy de inferência (ANFIS), como ilustra a Figura 5.19.

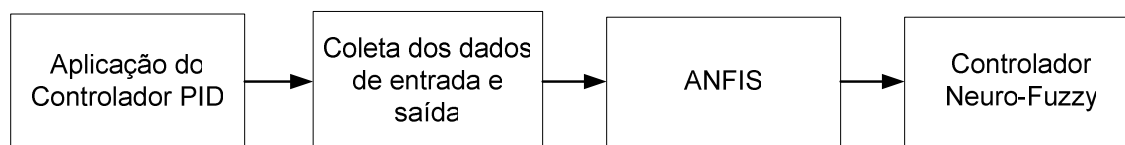


Figura 5.19 - Etapas para parametrização do controlador neuro-fuzzy.

O sistema foi implementado no software MATLABTM, no qual foi empregado o método de inferência Takagi-Sugeno. Os parâmetros de entrada são representados por conjuntos *fuzzy* (variáveis linguísticas). Por meio da ANFIS a faixa das funções de pertinência foi treinada, as regras *fuzzy* foram criadas e seus pesos ajustados a fim de se obter o melhor modelo a partir dos dados de entrada/saída. A Figura 5.20 mostra as variáveis de entrada aplicadas ao modelo neuro-fuzzy, enviando um sinal de controle para a saída.

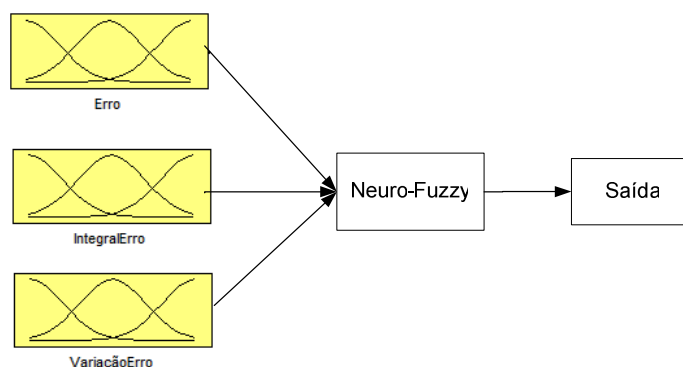


Figura 5.20 - Modelo ANFIS elaborado.

Para a estimativa do modelo matemático foram levantados 1000 pares de dados de entrada/saída, sendo colocados na interface de identificação ANFIS do MATLABTM. Para este processo foram utilizadas três funções de pertinência para cada entrada do tipo triangular e para a saída como linear, resultando em 27 regras, como é ilustrado na Figura 5.21.

Com estes elementos foi obtida a seguinte estrutura de modelo ANFIS, descrita na Figura 5.22.

Foram feitas 20 iterações (épocas) sendo o erro médio quadrático por iteração mostrado na Figura 5.23.

A partir das premissas citadas anteriormente foi obtido o modelo a seguir (vermelho), a partir de um conjunto de dados de entrada/saída, sendo obtido um erro médio final de 0,0003 %. A Figura 5.24 mostra os dados de entrada/saída e o modelo obtido com a aplicação do controlador neuro-*fuzzy*.

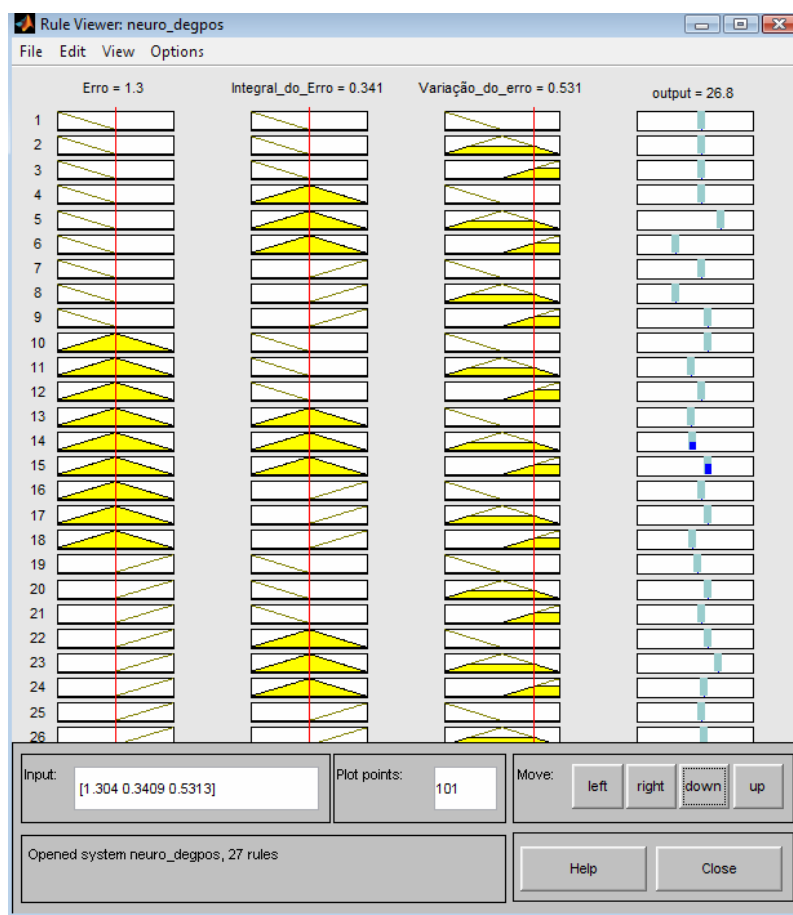


Figura 5.21 - Funções de pertinência das entradas e respectivas funções de saída.

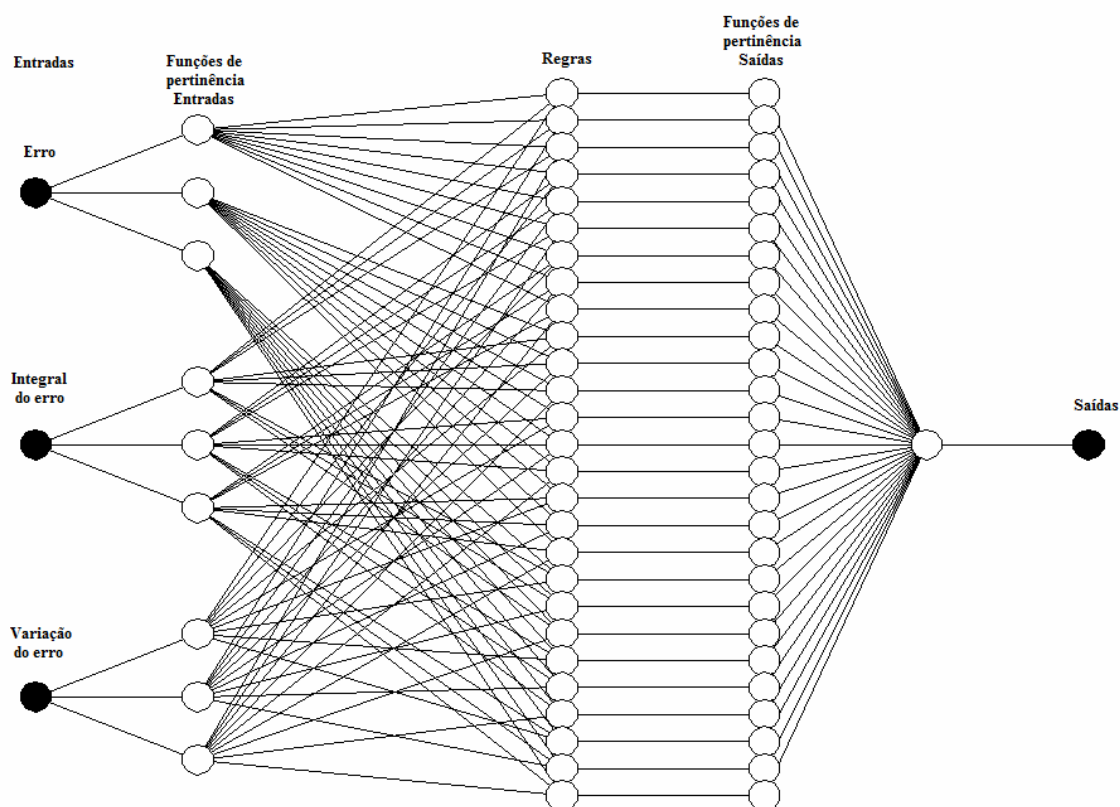


Figura 5.22 - Arquitetura de rede ANFIS com as regras geradas.

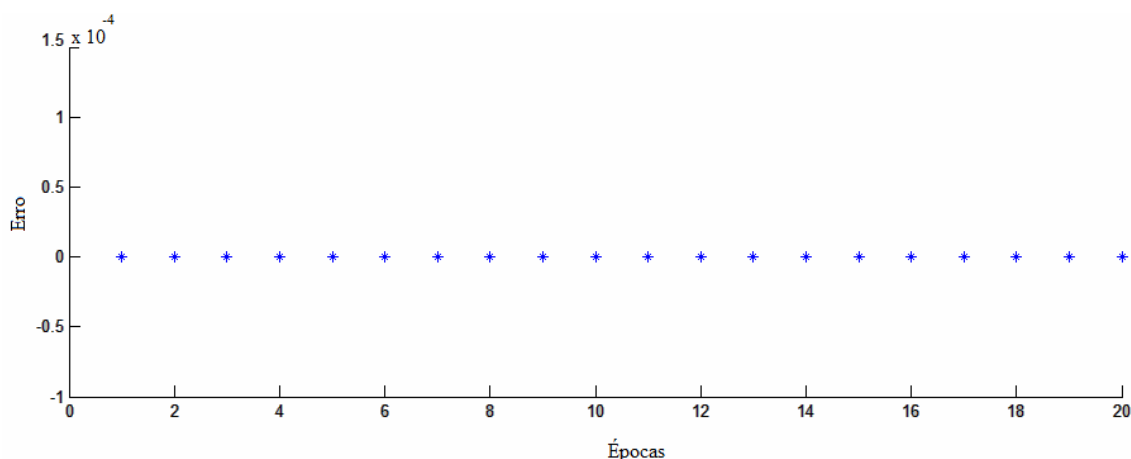


Figura 5.23 - Erro médio quadrático por iteração do modelo obtido.

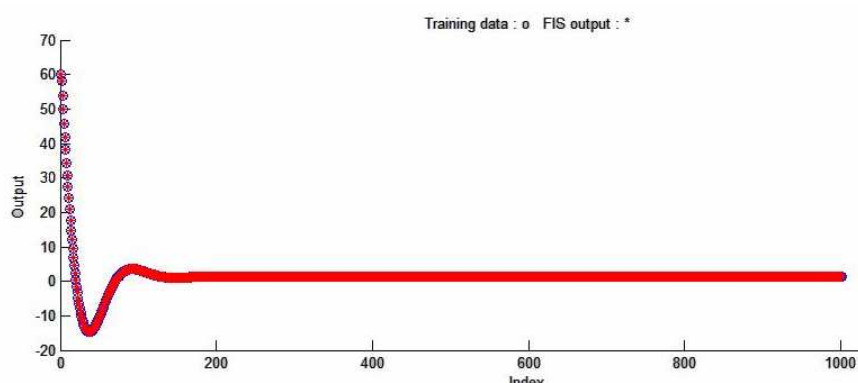


Figura 5.24 – Dados de entrada/saída e modelo obtido com a simulação.

5.6 Resultados Obtidos a partir das Simulações

Nas seções a seguir serão apresentadas as respostas dos controladores mediante a perturbações no *setpoint* e na carga para controladores PID, *fuzzy*-PID e *neuro-fuzzy*.

5.6.1 Avaliação de desempenho dos controladores mediante perturbação no *setpoint*

De acordo com os índices apresentados, foi feita uma análise de performance mediante a aplicação de uma perturbação degrau unitário no *setpoint* para os controladores *fuzzy*-PI e *fuzzy*-PID e sendo feita uma comparação com os controladores convencionais (PI e PD), sendo as respostas ilustradas na Figura 5.25.

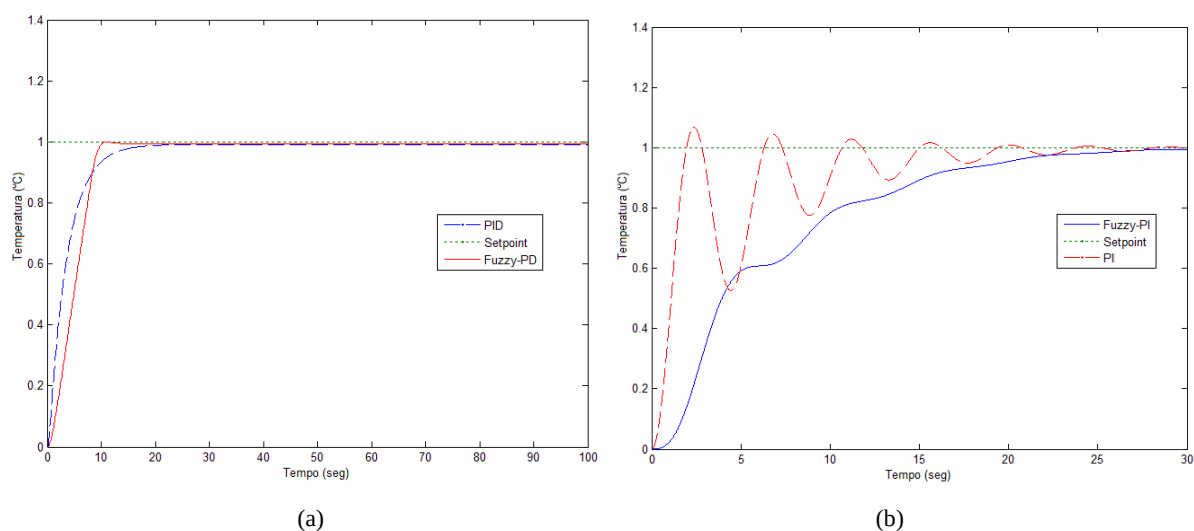


Figura 5.25 - Comparação de controladores para perturbação degrau unitário no *setpoint*
(a) *fuzzy-PD/PD* (b) *fuzzy-PI/PI*.

Na Tabela 5.8, são expostos os índices de desempenho dos controladores *fuzzy-PI* e *fuzzy-PD*, comparados com os controladores convencionais PD e PI.

Tabela 5.8 - Índices de desempenho para controladores PI/*fuzzy-PI* e PD/*fuzzy-PD* para perturbação degrau unitário no *setpoint*.

ÍNDICES	FUZZY-PD	PD	FUZZY-PI	PI
Instante de pico (seg)	10,896	20,89	27,73	2,36
<i>Overshoot</i> máximo	1,0004	0,993	0,996	1,068
Tempo de Acomodação (2%)	10,50	10,55	26,72	28,58
Tempo de atraso (seg)	4,85	2,61	3,902	1,0823
Tempo de subida (seg)	7,45	8,1	14,44	1,28

Na sequência do trabalho foi feita uma simulação com o controlador *fuzzy-PID* em comparação com um controlador convencional PID, como mostra a Figura 5.26.

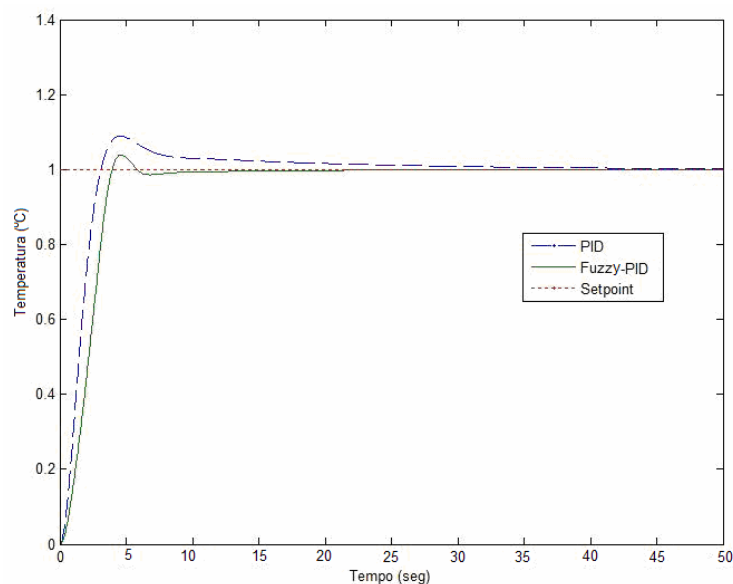


Figura 5.26 - Comparação de controladores PID/fuzzy-PID para perturbação degrau unitário no *setpoint*.

Nota-se que o controlador *fuzzy*-PID obteve uma performance bastante superior nos índices de desempenho em relação aos controladores *fuzzy*-PI e *fuzzy*-PD, sendo esta a motivação do uso deste controlador nas seções posteriores.

Na simulação seguinte verificou-se a resposta dos controladores *neuro-fuzzy* mediante a uma perturbação no *setpoint*. Na Figura 5.27 está ilustrada a resposta deste controlador em comparação ao controlador convencional PID.

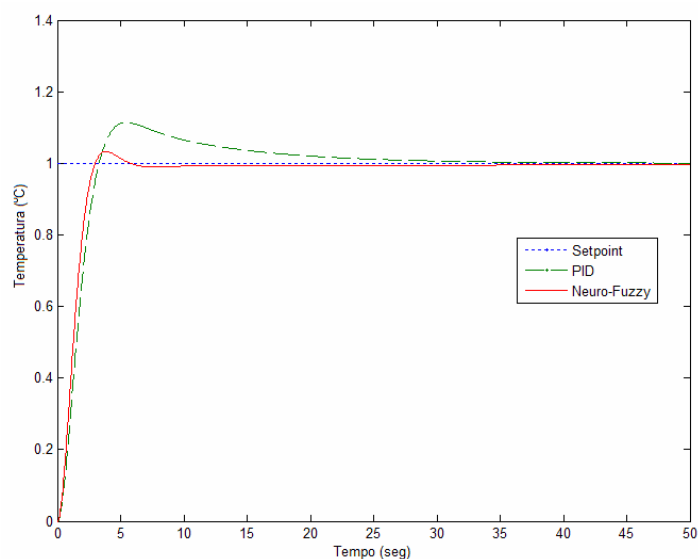


Figura 5.27 - Comparação de controladores PID/*neuro-fuzzy* para perturbação degrau unitário no *setpoint*.

Nas Figuras 5.28, 5.29, 5.30, 5.31 e 5.32 estão apresentados de uma forma comparativa os índices de desempenho para os controladores PID, *fuzzy*-PID e neuro-*fuzzy*.

A partir da análise da resposta dos gráficos expostos, foram tabulados os índices de desempenho e apresentados na Tabela 5.9.

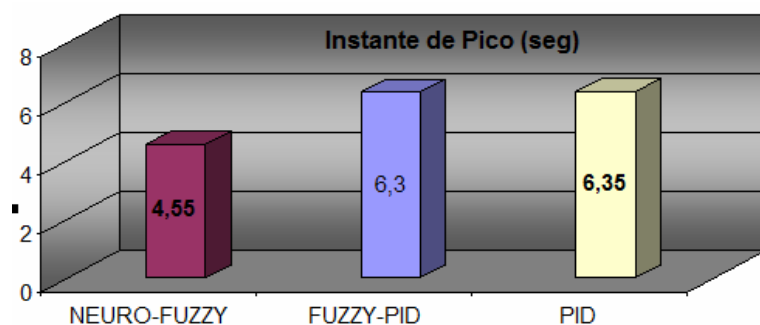


Figura 5.28 - Comparativo de instante de pico para os controladores.

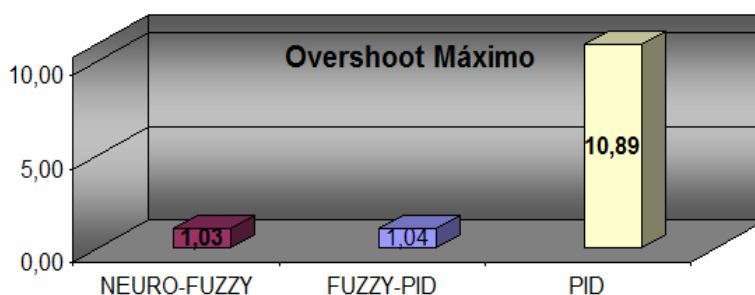


Figura 5.29 - Comparativo de *overshoot* máximo para os controladores.

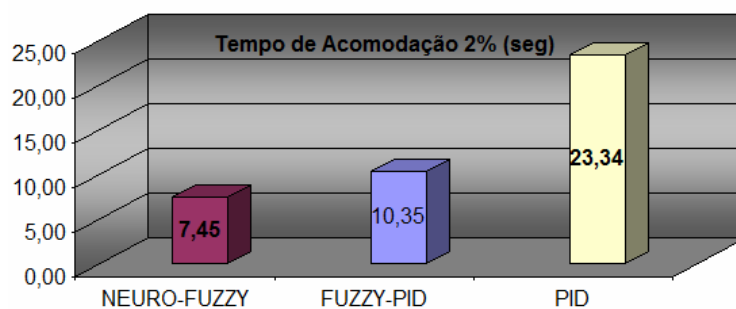


Figura 5.30 - Comparativo de tempo de acomodação para os controladores.

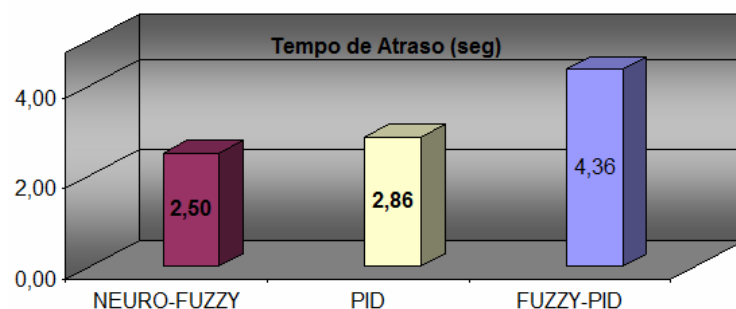


Figura 5.31 - Comparativo de tempo de atraso para os controladores.

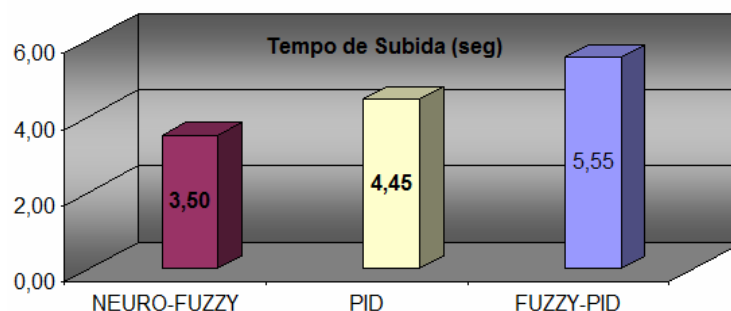


Figura 5.32 - Comparativo de tempo de subida para os controladores.

Tabela 5.9 - Índices de desempenho para controladores PID/fuzzy-PID para perturbação degrau unitário no *setpoint*.

ÍNDICES	FUZZY-PID	NEURO-FUZZY	PID
Instante de pico (seg)	6,30	4,55	6,35
<i>Overshoot</i> máximo	1,037	1,033	1,0892
Tempo de Acomodação (2%)	10,35	7,45	23,335
Tempo de atraso (seg)	4,36	2,5	2,86
Tempo de subida (seg)	5,55	3,5	4,45

Os valores dos parâmetros de desempenho obtidos comprovaram que o controlador *fuzzy*-PID apresentou uma resposta inferior ao controlador PID nos quesitos tempo de resposta e tempo de atraso, entretanto obteve uma performance muito superior no tempo de acomodação o que comprovou um melhor desempenho deste controlador em relação ao PID. Por sua vez, o controlador *neuro-fuzzy* apresentou uma resposta melhor em todos os índices de desempenho, sendo estes elementos que confirmam a superioridade deste controlador para perturbações no *setpoint*.

5.6.2 Avaliação de desempenho dos controladores mediante perturbação na carga

Para avaliar a performance dos controladores mediante perturbação na carga foram utilizados os índices *ITAE*, *ISE* e *IAE*. Nesta análise da resposta dos controladores foram aplicadas perturbações na carga para os controladores *fuzzy*-PID e *neuro-fuzzy* no tempo de 50s, com os seguintes valores: +1, +2, -1 e -2. As Figuras 5.33a, 5.33b, 5.33c, 5.33c, apresentam a resposta destes controladores em comparação com o controlador PID.

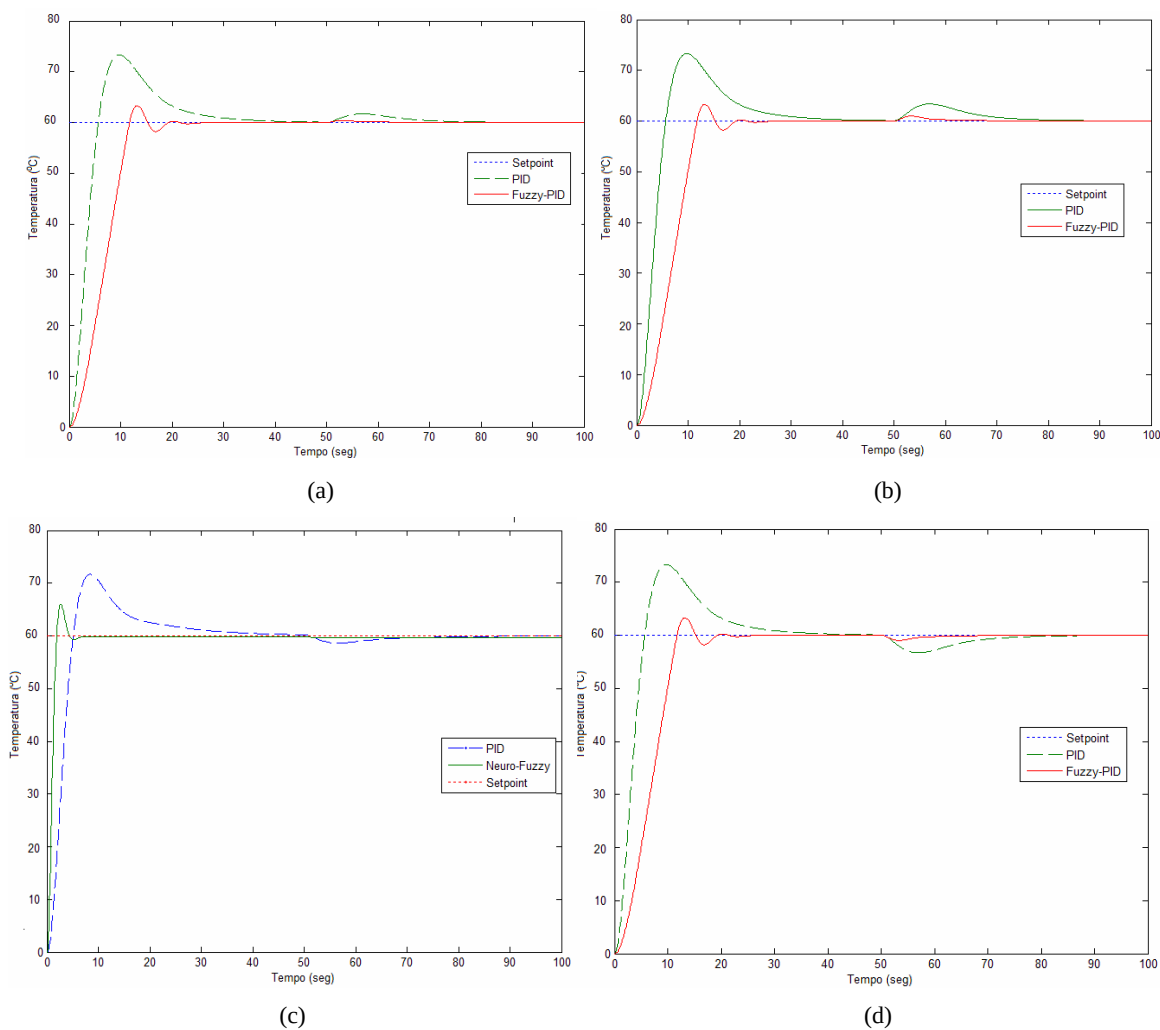
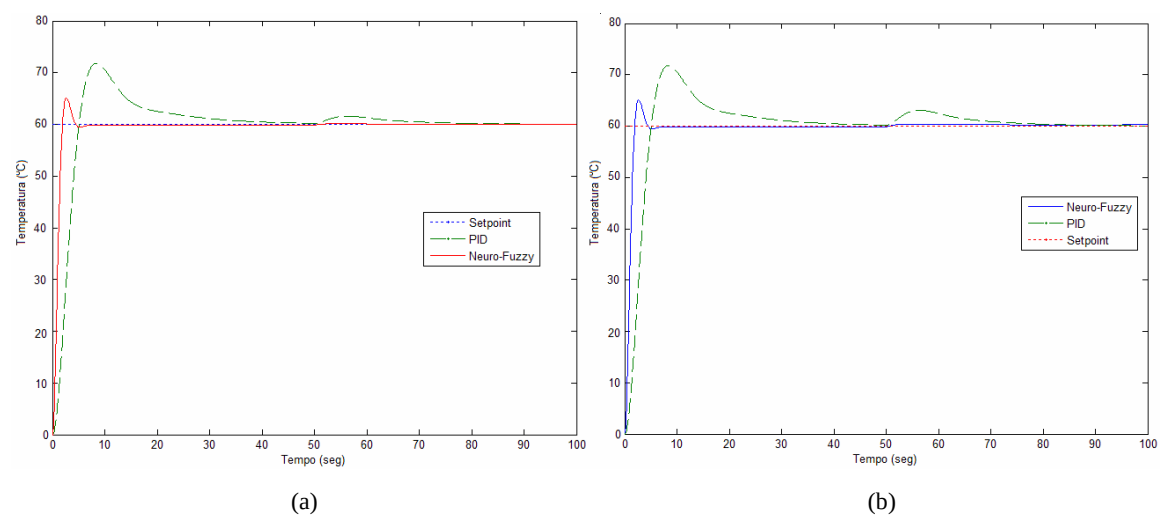


Figura 5.33 - Comparação de controladores PID/fuzzy-PID para perturbação degrau na carga (a) +1 (b) +2 (c) -1 (d) -2.

A seguir são descritos os resultados obtidos com a simulação do controlador neuro-fuzzy, bem com os índices de desempenho e comparação com o controlador convencional PID, como ilustra a Figura 5.34.



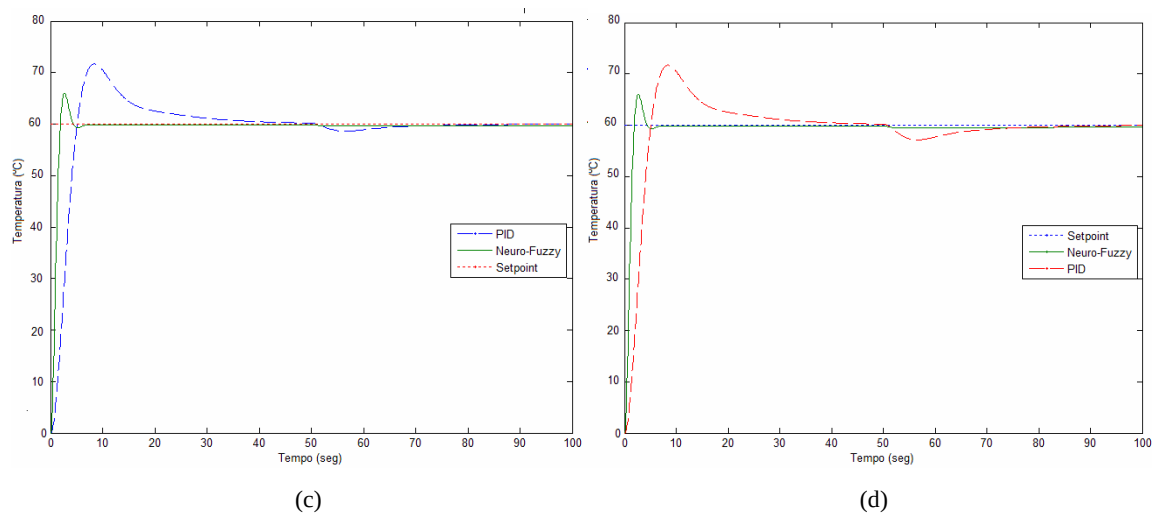


Figura 5.34 - Comparação de controladores PID/neuro-*fuzzy* para perturbação degrau na carga (a) +1 (b) +2 (c) -1 (d) -2

Nas Figuras 5.35, 5.36 e 5.37, está representada a comparação mediante os índices de desempenho para os controladores PID, *fuzzy*-PID, neuro-*fuzzy* para as perturbações na carga especificadas.

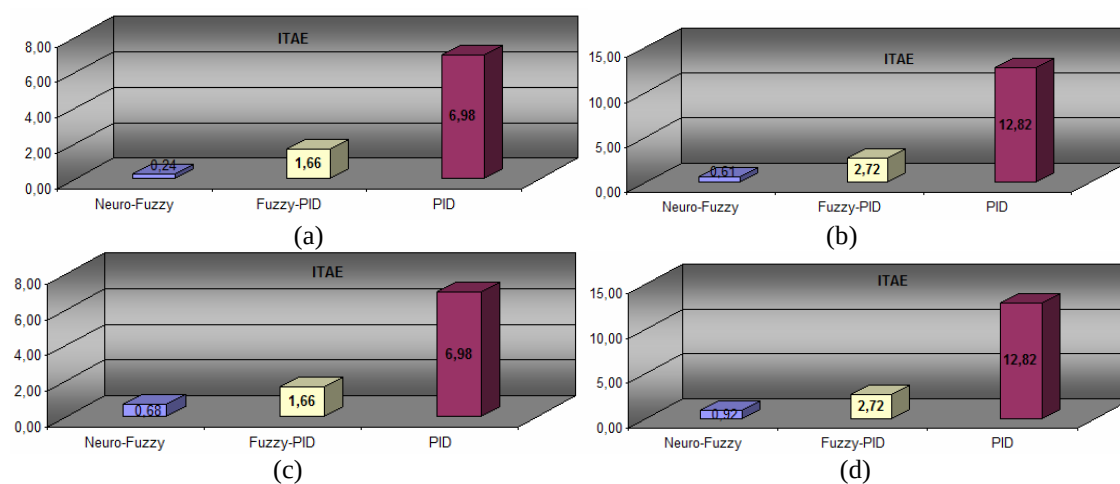
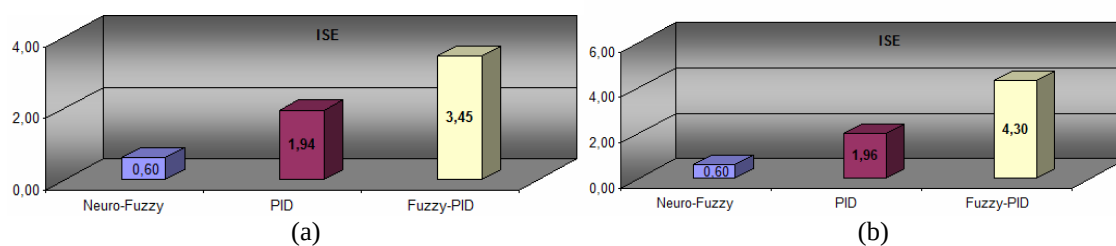


Figura 5.35 - Comparação do índice *ITAE* para os controladores PID, *fuzzy*-PID e neuro-*fuzzy* para perturbação degrau na carga (a) +1 (b) +2 (c) -1 (d) -2.



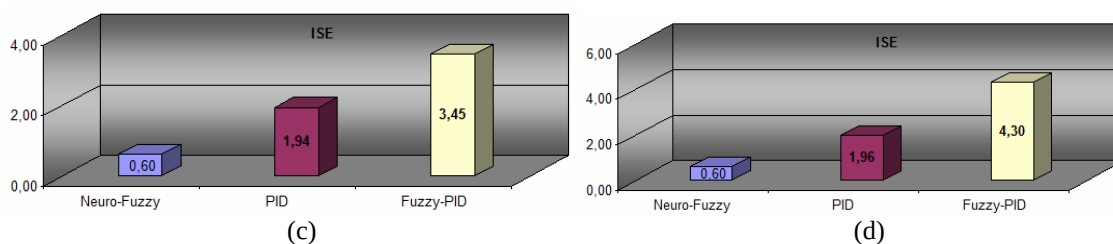


Figura 5.36 - Comparação do índice *ISE* para os controladores PID, *fuzzy*-PID e neuro-*fuzzy* para perturbação degrau na carga (a) +1 (b) +2 (c) -1 (d) -2.

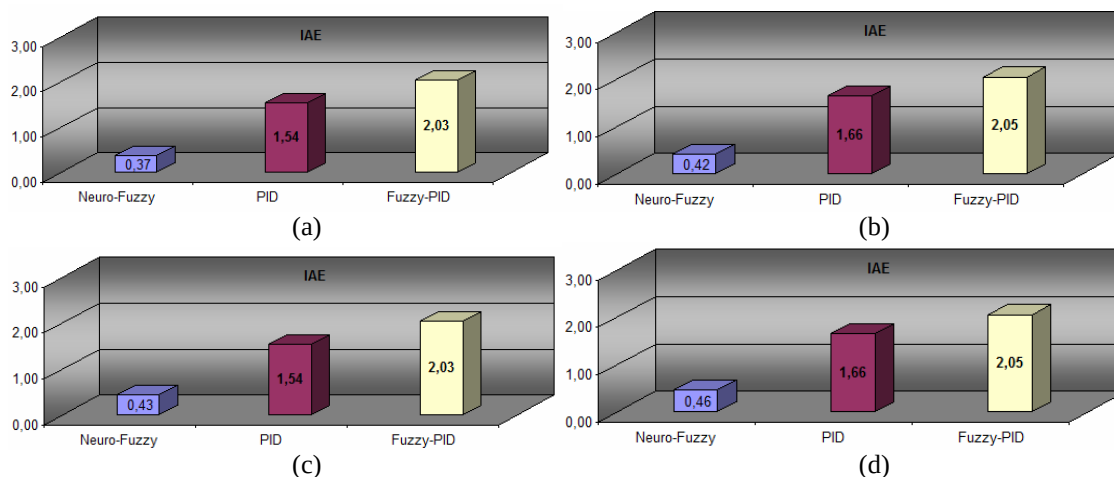


Figura 5.37 - Comparação do índice *IAE* para os controladores PID, *fuzzy*-PID e neuro-*fuzzy* para perturbação degrau na carga (a) +1 (b) +2 (c) -1 (d) -2.

Na Tabela 5.10 estão tabulados os índices de desempenho *ITAE*, *IAE* e *ISE* para os controladores PID, *fuzzy*-PID e neuro-*fuzzy*.

Tabela 5.10 - Índices de desempenho para controladores PID/*fuzzy*-PID/neuro-*fuzzy* para perturbação na carga.

Perturbação	PID			<i>Fuzzy</i> -PID			Neuro- <i>Fuzzy</i>		
	<i>ITAE</i>	<i>IAE</i>	<i>ISE</i>	<i>ITAE</i>	<i>IAE</i>	<i>ISE</i>	<i>ITAE</i>	<i>IAE</i>	<i>ISE</i>
+1	6,979	1,542	1,941	1,662	2,032	3,453	0,2355	0,3662	0,5973
+2	12,82	1,656	1,955	2,724	2,052	4,298	0,6079	0,4152	0,5981
-1	6,979	1,542	1,941	1,662	2,032	3,453	0,6786	0,4285	0,5984
-2	12,82	1,656	1,955	2,724	2,052	4,298	0,9186	0,4573	0,5995

Da mesma forma que na perturbação no *setpoint*, o controlador neuro-*fuzzy* apresentou uma resposta melhor em todos os índices de desempenho, sendo estes elementos que vêm novamente confirmam a eficácia deste controlador mediante perturbações na carga. O inconveniente de se utilizar o controlador neuro-*fuzzy* é a

necessidade de um conjunto de dados que devem ser coletados e armazenados, bem como o tempo necessário para confecção e treinamento da rede neural.

Para estas simulações foram feitas variações hipotéticas na carga em degraus positivos e negativos como forma de aproximar as reais perturbações possíveis na coluna de destilação (vazão de entrada, vazão de saída e razão de refluxo). Levando em consideração estes aspectos, apesar do controlador *neuro-fuzzy* obter um melhor desempenho nas simulações em todos os índices de desempenho, os controladores PID e *fuzzy*-PID obtiveram um bom desempenho com pequeno tempo de resposta e *overshoot*, bem como erros em regime permanente considerados pequenos.

CAPÍTULO 6 – APLICAÇÃO DAS ESTRATÉGIAS DE CONTROLE

Neste capítulo estão contidos os resultados obtidos com a aplicação das estratégias de controle na coluna de destilação didática do Departamento de Engenharia Química da Universidade Estadual de Maringá.

Para implementação das estratégias de controle na coluna de destilação foi elaborado o sistema apresentado na Figura 6.1.

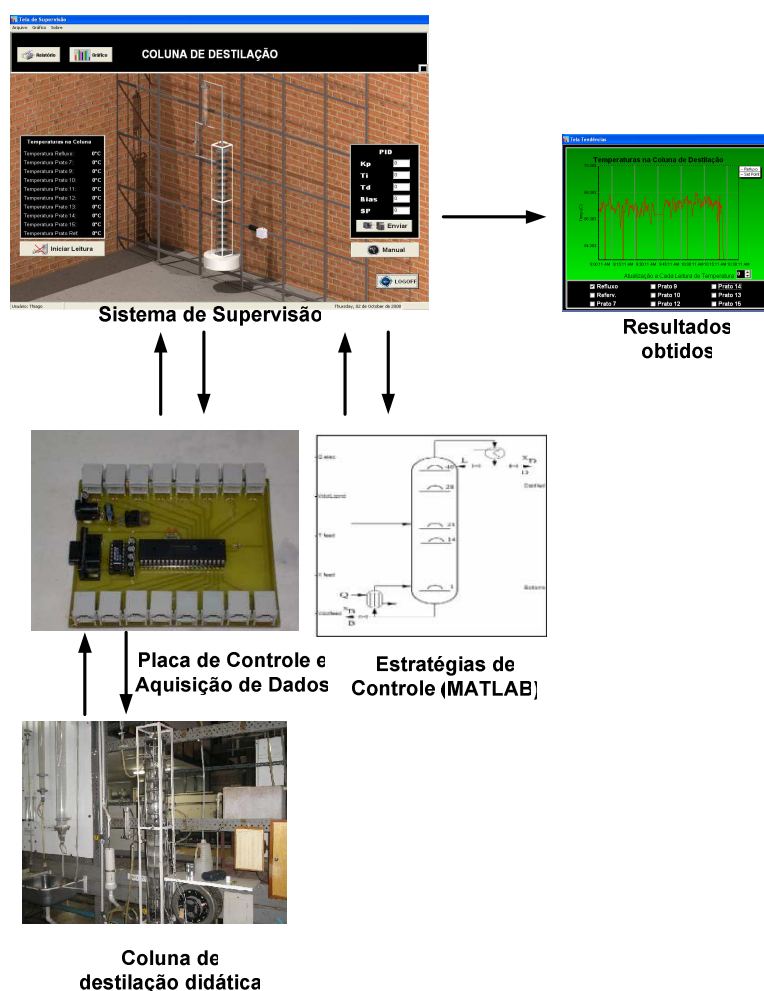


Figura 6.1 - Arquitetura proposta para monitoração e controle da coluna de destilação estudada.

O sistema empregado é constituído das seguintes partes:

- o *Software* de simulação MATLABTM, responsável pela aplicação das estratégias de controle PID, *fuzzy* e *neuro-fuzzy*;
- o Sistema de supervisão, responsável pela interface homem máquina (IHM) e visualização de variáveis do processo e controladores, bem como conexão com banco de dados para geração de relatórios dos controladores.
- o Placas de aquisição de dados e de controle de potência, encarregadas de coletar dados de temperatura, integrá-los ao sistema de supervisão e receber os sinais de saída das estratégias de controle e enviá-las a variável manipulada (resistência elétrica do refeedor);
- o Coluna de destilação piloto, equipamento em que são aplicadas as estratégias de controle.

Na topologia do sistema empregado, as temperaturas foram coletadas por meio de sensores de temperatura e enviados à placa de aquisição de dados, que tem por função enviar os sinais de entrada da variável de processo ao *software* MATLABTM, responsável pela implementação das estratégias de controle. O sinal de saída dos controladores é enviado à placa de controle de potência que tem como função enviar o sinal à variável manipulada (potência elétrica do refeedor).

6.1 - Aplicação do Sistema à Coluna de Destilação do LabEQ2 (DEQ/UEM)

Para implementar as estratégias de controle foi utilizada uma coluna de destilação alocada no Laboratório de Engenharia Química II do Departamento de Engenharia Química constituída de 16 pratos perfurados, sendo o prato de alimentação o prato de número 10, considerando a numeração de cima para baixo.

Neste módulo a coluna foi construída em vidro transparente, para que seja possível visualizar os fenômenos de transferência de energia e massa que ocorrem em cada prato. O aquecimento do refeedor é feito por meio de resistências elétricas e a variação da quantidade de energia utilizada se dá por meio da variação da potência elétrica empregada nas resistências, sendo que a potência máxima que pode ser obtida é 5500 W.

O condensador utilizado foi construído em aço e utiliza água como líquido refrigerante. Trata-se de um condensador total, ou seja, todo o vapor do topo da coluna é condensado. Uma parte do vapor condensado retorna à coluna como refluxo, sendo

enviado a um vaso acumulador acoplado na saída do condensador, recebendo a fase líquida e tendo como objetivo manter a vazão de refluxo para o processo de destilação. A outra parte consiste na corrente de produto de topo.

A mistura a ser destilada é colocada em um reservatório na parte superior do módulo e, por gravidade, é pré-aquecida na base da coluna, juntamente com o refeedor, sendo introduzida no prato de alimentação da coluna de destilação.

O material utilizado para a construção da coluna foi o vidro boro silicato, com 250 cm de altura, 15 cm de diâmetro e espessura de 5 mm.

Para a coleta dos sinais de temperatura, foram colocados sensores de temperatura do tipo DS18B20 externamente à coluna, devido à impossibilidade da sua colocação no seu interior, nos pratos de refluxo (2), 7, 9, 10, 11, 12, 13, 14 e 15 (de cima para baixo) e no refeedor (*reboiler*). Os sinais provenientes destes sensores são enviados à placa de aquisição de dados, diretamente conectada ao sistema de supervisão contido em um microcomputador. A coluna de destilação e a interligação com o sistema de aquisição, controle e supervisão são apresentados na Figura 6.2.

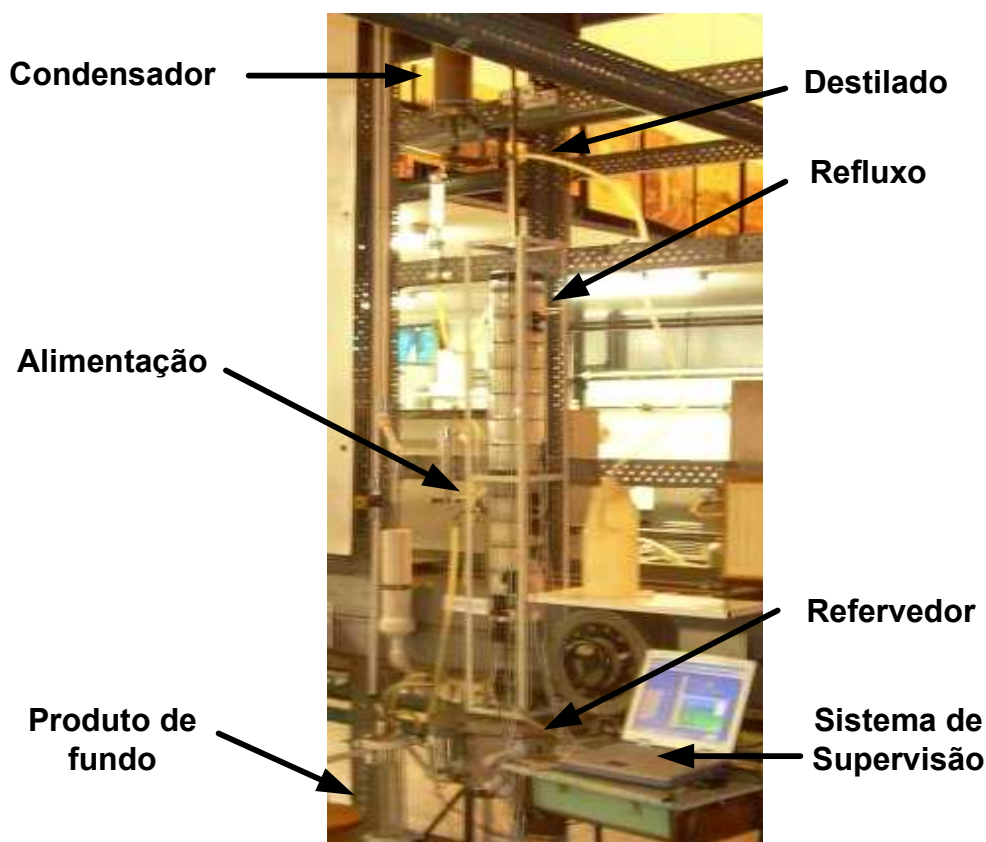


Figura 6.2 - Coluna de destilação integrada ao sistema de controle, aquisição de dados e supervisão.

O trabalho desenvolvido foi aplicado à coluna de destilação na separação de uma mistura etanol-água. Este sistema foi escolhido devido ao baixo custo e facilidade de aquisição destes componentes, além da baixa viscosidade e da facilidade de limpeza que a mistura oferece.

O procedimento de partida da coluna foi tem início com o aquecimento da coluna de destilação por meio do refeedor, até que o equipamento apresente um aquecimento uniforme ao longo do seu comprimento. Após esta etapa efetua-se a alimentação na coluna, que por gravidade, dirige-se ao refeedor, sendo então vaporizada assim que se aplica calor ao refeedor. O vapor ascende até o condensador, de onde retorna como refluxo condensado. A coluna é operada em refluxo total durante certo intervalo de tempo até que a composição do destilado se aproxime da composição desejada em regime permanente. Assim que a composição desejada é atingida, inicia-se a retirada de produto de fundo e a alimentação da coluna, para se fazer a transição do modo de partida para o modo estacionário.

Assim que o balanço de massa da coluna é fechado, ou seja, a vazão da entrada é igual a soma da vazão do produto de fundo com a vazão do destilado, inicia-se a coleta de amostras do produto de topo (destilado) e do produto de fundo para a determinação das concentrações finais.

Durante todo o processo são coletadas as temperaturas, por meio do *software* de supervisão e da placa de aquisição de dados. Foram coletadas as temperaturas nos pratos 2, 7, 9, 10, 11, 12, 13, 14 e 15, além das temperaturas do refeedor. Os valores de temperatura em cada um dos sensores foi armazenado em um banco de dados do sistema de supervisão. A Figura 6.3 mostra a variação das temperaturas na coluna de destilação nos pratos desde o momento de sua partida, alimentação até sua parada.

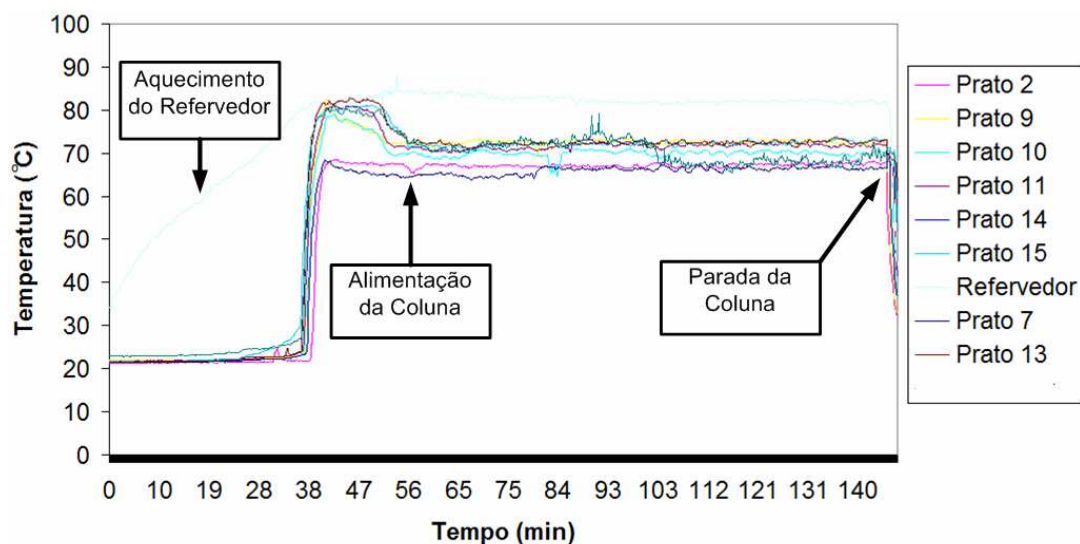


Figura 6.3 - Pontos de operação e temperaturas da coluna de destilação.

A partir dos dados coletados de temperatura (considerando o estado estacionário) foram feitas as médias aritméticas das temperaturas e também a aplicação da função de correção da aferição dos sensores apresentadas na Tabela 6.1.

Tabela 6.1 - Médias aritméticas das temperaturas e valores de temperatura aferidos.

Elemento	Temperatura Lida no Sensor(°C)	Temperatura Corrigida(°C)
Refervedor	82,25	81,94
Prato 16	77,51	77,25
Prato 14	75,32	75,11
Prato 13	74,21	73,91
Prato 12	73,82	73,53
Prato 11	72,87	72,11
Prato 10	71,91	72,08
Prato 9	71,31	71,12
Prato 7	68,38	68,28
Prato Refluxo	66,12	66,28

Com base nos dados de regime permanente, foi plotado o gráfico das médias das temperaturas (perfil de temperatura) nos pontos da coluna, como mostra a Figura 6.4.

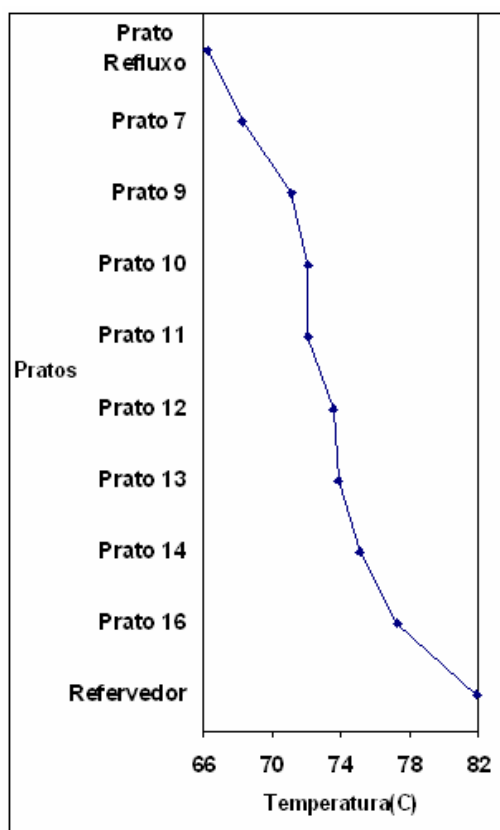


Figura 6.4 - Perfil de temperaturas médias na coluna de destilação.

As medidas de concentração foram realizadas durante os experimentos por meio da retirada de amostras de misturas da base e do topo da coluna e mensuradas com um densímetro para álcoois. As medidas foram realizadas a uma temperatura de 20°C, e são apresentadas em base volumétrica.

6.2 – Aplicação das Estratégias de Controle na Coluna de Destilação

Para aplicar as estratégias de controle à coluna de destilação considerou-se uma malha de controle, sendo que a variável de processo é a temperatura de topo e a variável manipulada a potência elétrica do refervedor, como ilustra a Figura 6.5.

O objetivo do sistema é controlar a concentração do produto de topo. Entretanto, para esta finalidade é necessário implementar instrumentos capazes de executar tal tarefa. Analisadores de composição *online* demandam grandes investimentos e custos de manutenção, além de terem grandes atrasos de medição. Segundo KISTER (1992), a melhor forma de se obter a concentração em uma coluna de destilação, de uma maneira fácil, rápida e de baixo custo é por meio da medição de temperatura.

Segundo POIANI (1993), a alternativa utilizada em implementações usuais de controle é medir as temperaturas e inferenciar a composição utilizando dados termodinâmicos de equilíbrio líquido-vapor. Esta metodologia se adapta bem a misturas binárias ideais, devido ao fato que para sistemas com múltiplos componentes ou não ideais a complexidade e o esforço computacional do modelo termodinâmico de equilíbrio líquido-vapor que necessita ser utilizado é sensivelmente aumentada.

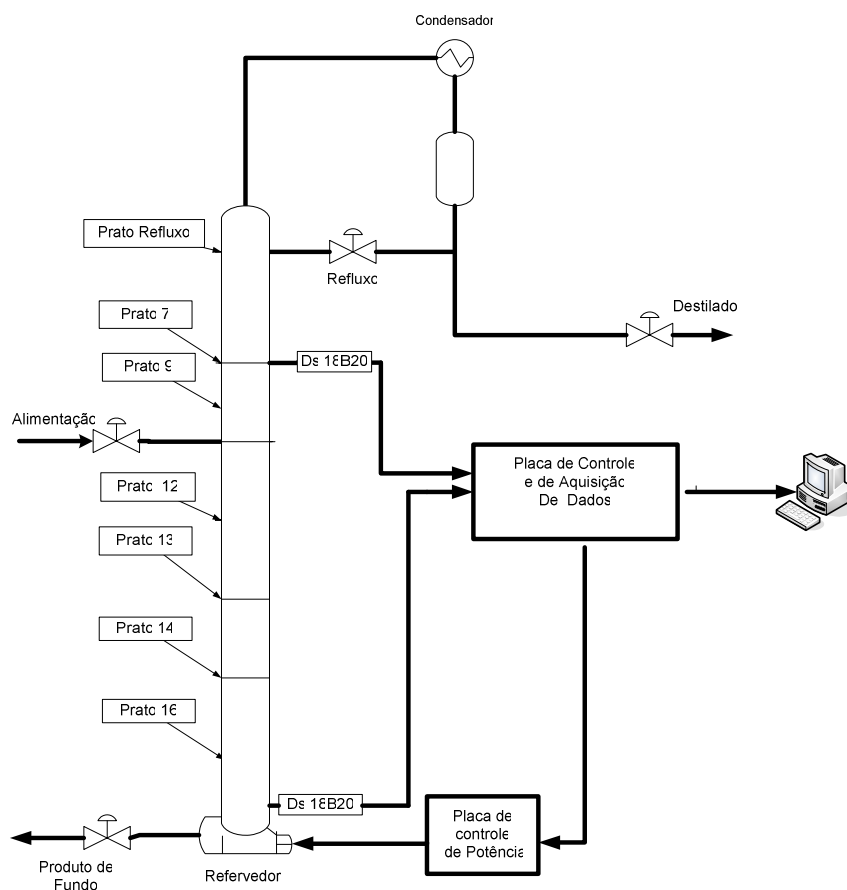


Figura 6.5 - Malha de controle proposta para coluna de destilação.

Segundo PEDROSA (1998), o uso de uma rede neural artificial utilizando razões de polinômios e modelos fenomenológicos com etapas de treinamento adequadas e faixas de operação bem definidas, apresentou resultados satisfatórios para a inferência da concentração a partir da temperatura.

Para analisar a performance dos controladores foram realizados experimentos para destilação da mistura. A seguir é apresentado um dos experimentos com as seguintes condições de operação ilustradas na Tabela 6.2.

Tabela 6.2 - Condições de operação da coluna de destilação.

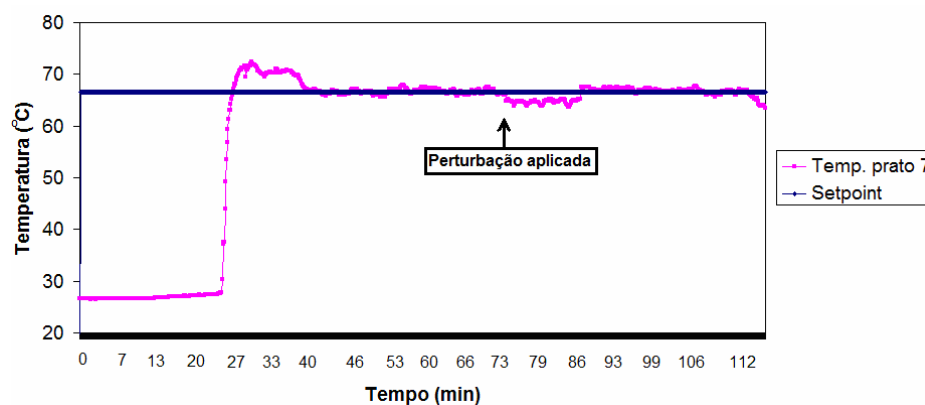
Variável da Mistura Destilada	Valor
Concentração de entrada	32 GL
Concentração de destilado	93 GL
Concentração do produto de fundo	4 GL
Vazão de entrada	210 mL/min
Vazão de destilado	70 mL/min
Vazão do produto de fundo	150 mL/min
Vazão de refluxo	180 mL/min
Temperatura da Alimentação	80°C

Com estas condições de operação, a coluna de destilação foi colocada em regime permanente, para aplicação das estratégias de controle mediante as perturbações na carga e no *setpoint*.

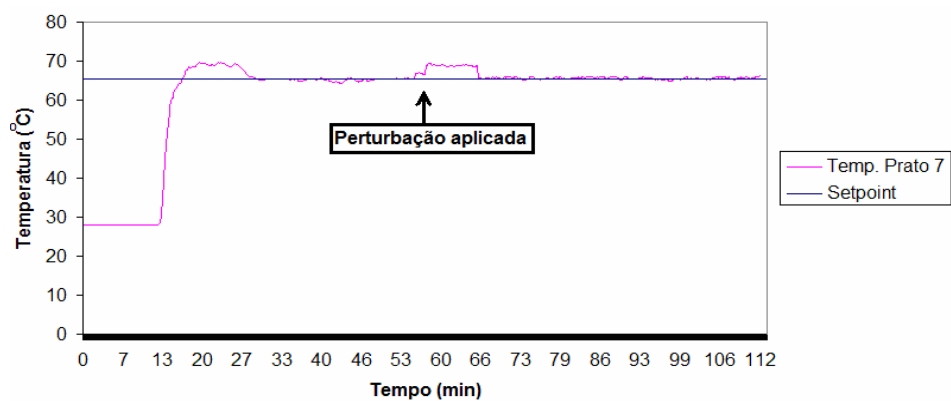
6.2.1 Desempenho dos controladores mediante perturbação na carga.

Para avaliar o desempenho dos controladores em condições de funcionamento da coluna de destilação, a mesma foi colocada em regime permanente, sendo feitas perturbações na carga e avaliados os índices de desempenho *ISE*, *ITAE* e *IAE*.

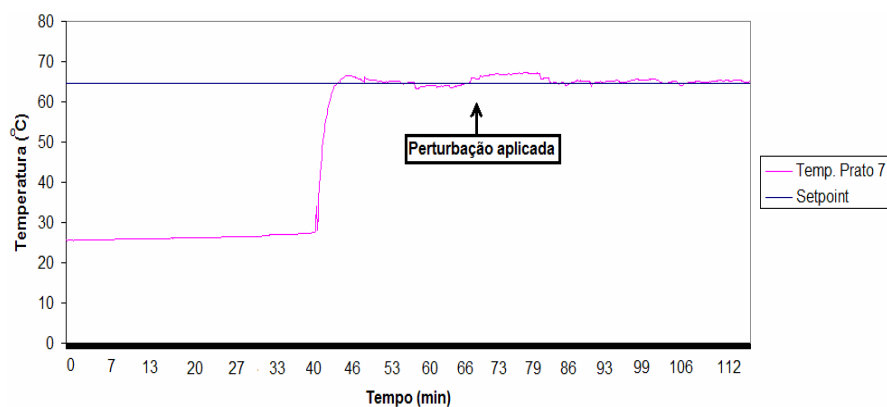
O valor desejado (*setpoint*) de temperatura no prato 7 foi de 65°C, que segundo os experimentos realizados corresponde ao valor de saída de 94 GL. Devido às características construtivas da coluna de destilação, foram feitas perturbações na vazão de entrada e razão de refluxo. A vazão de entrada sofreu perturbações de 20 mL/min e 40 mL/min, positivas e negativas em forma de degrau, sendo os resultados obtidos para os controladores PID, *fuzzy*-PID e *neuro-fuzzy* apresentados nas Figuras 6.6, 6.7 e 6.8, respectivamente.



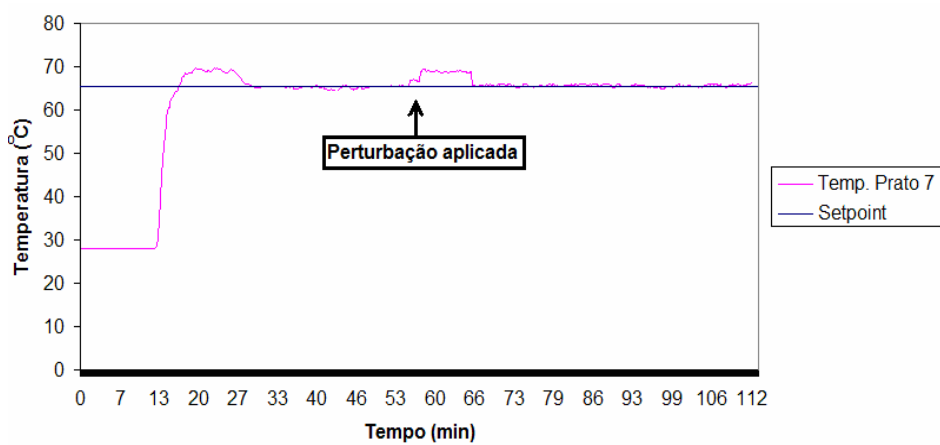
(a)



(b)

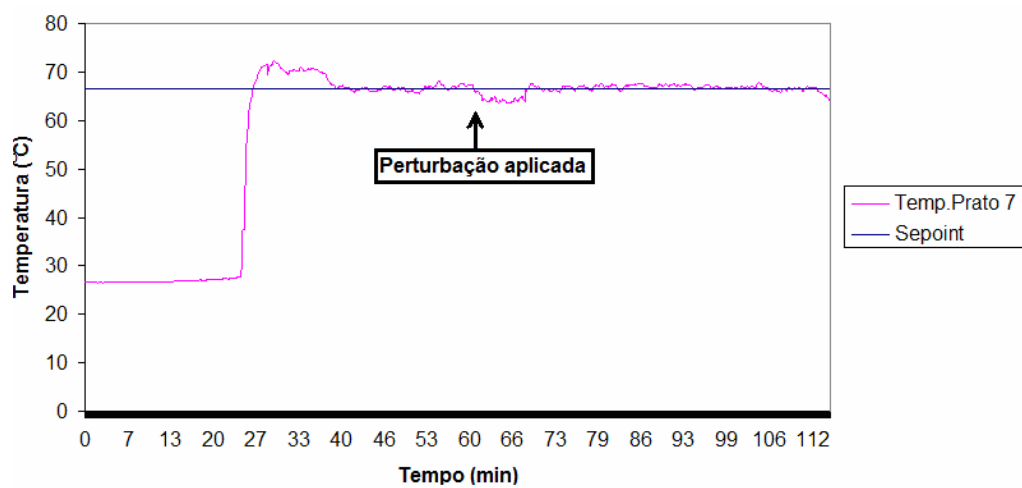


(c)

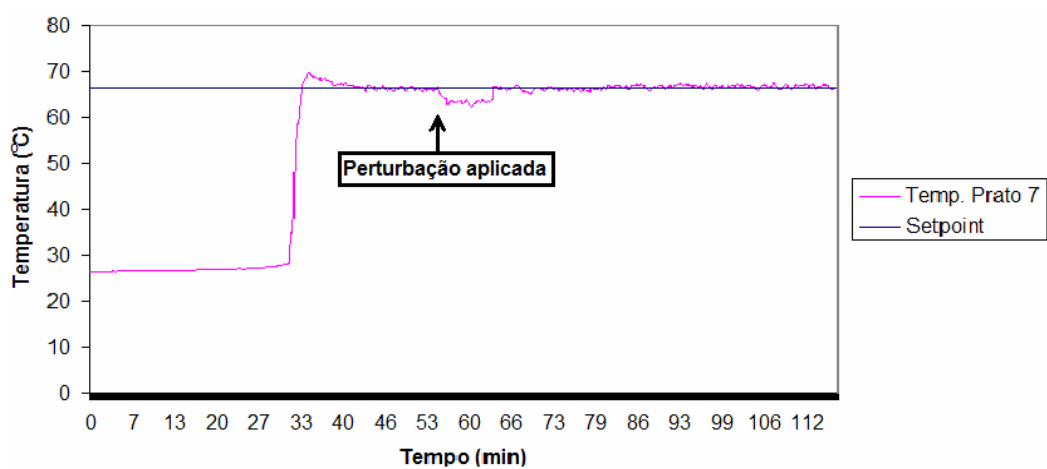


(d)

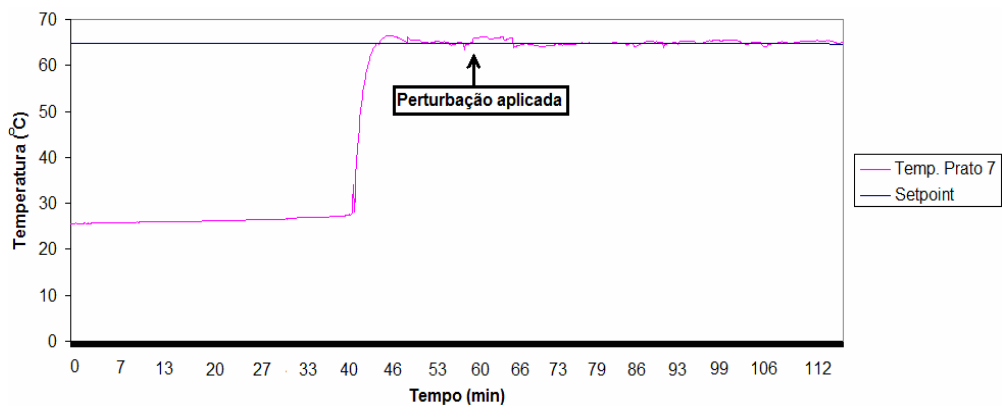
Figura 6.6 - Resposta do controlador PID para perturbação degrau na carga
 (a) +20 mL/min (b) +40 mL/min (c) -20 mL/min (d) -40 mL/min.



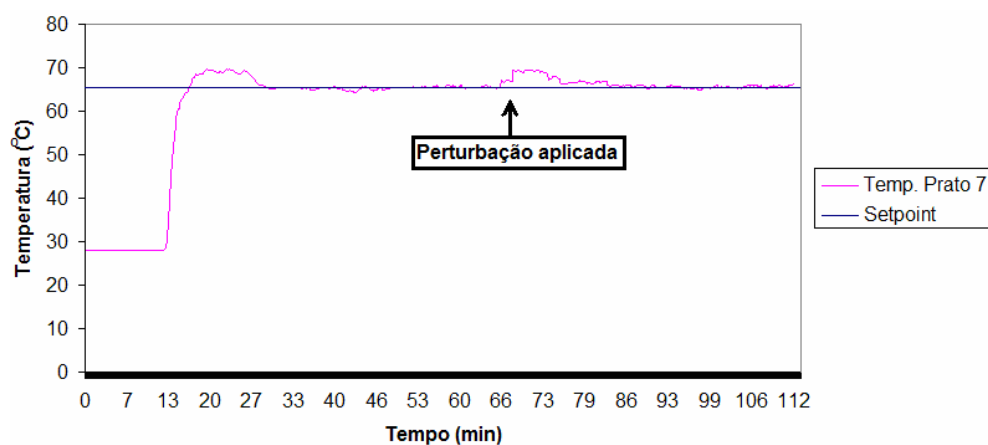
(a)



(b)

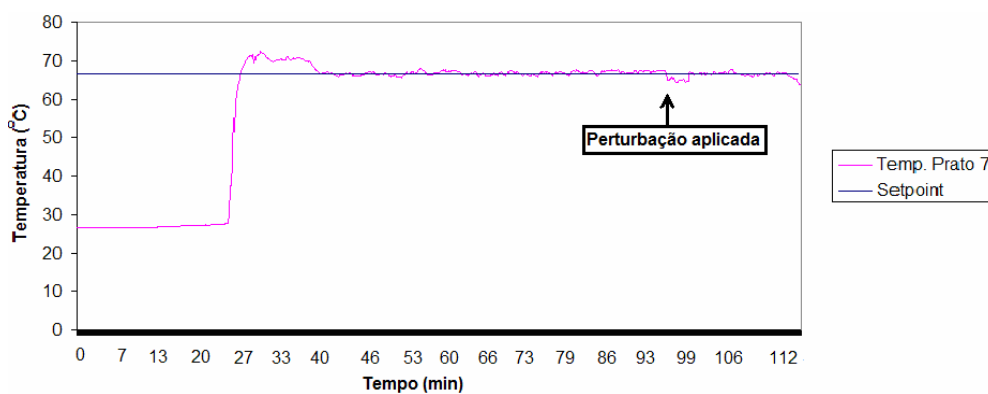


(c)

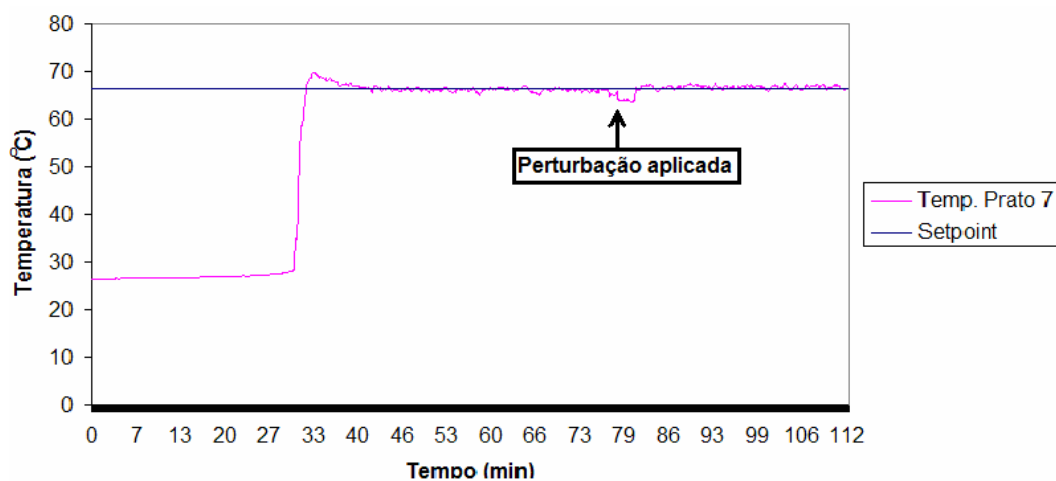


(d)

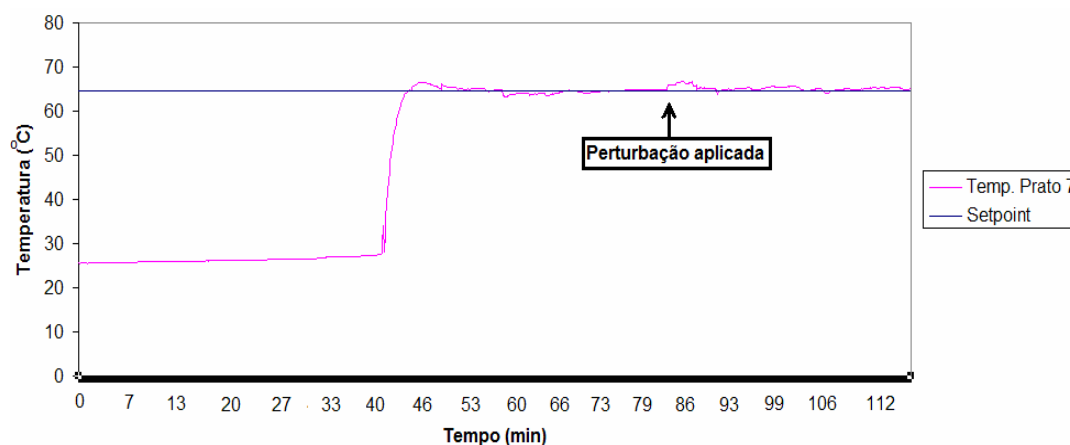
Figura 6.7 - Resposta do controlador *fuzzy*-PID para perturbação degrau na carga
 (a) +20 mL/min (b) +40 mL/min (c) -20 mL/min (d) -40 mL/min.



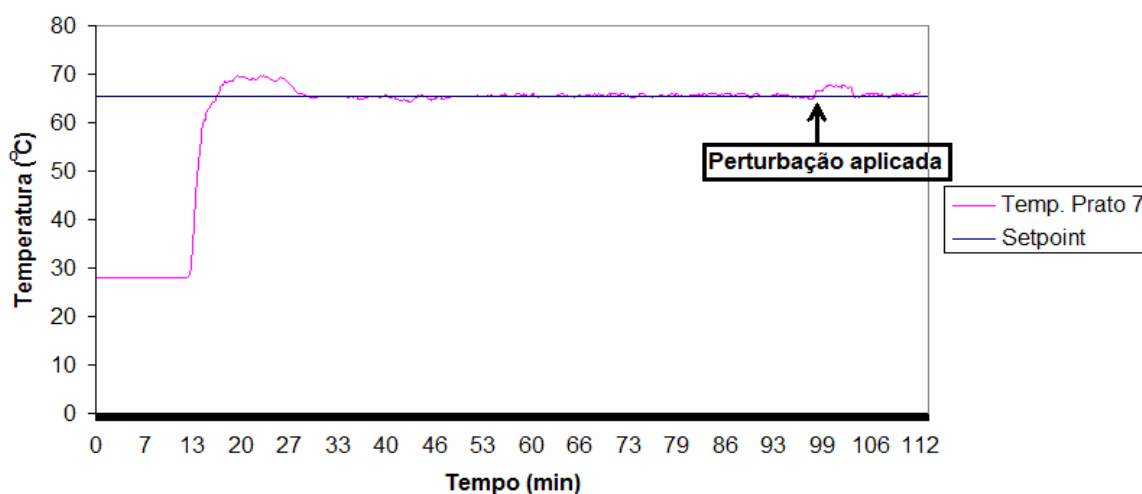
(a)



(b)



(c)

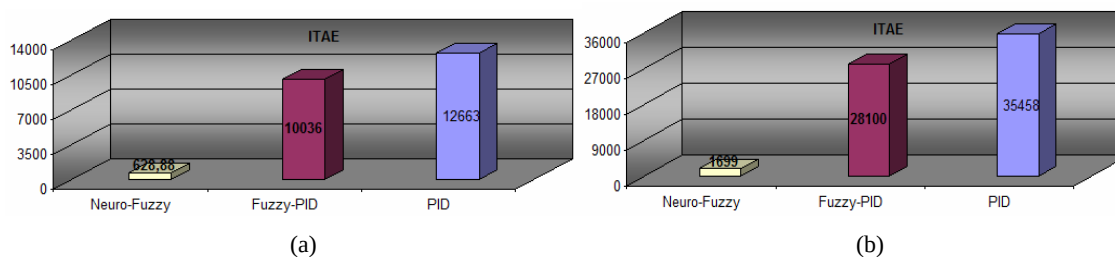


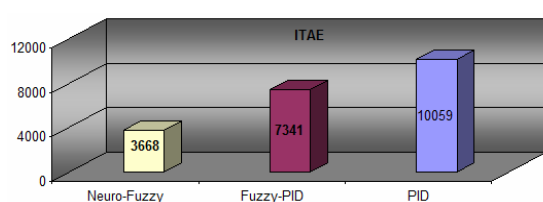
(d)

Figura 6.8 - Resposta do controlador neuro-fuzzy para perturbação degrau na carga
(a) +20 mL/min (b) +40 mL/min (c) -20 mL/min (d) -40 mL/min.

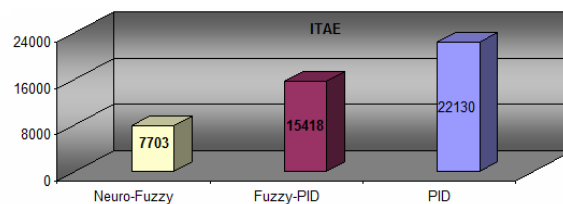
Para facilitar a análise de performance dos controladores nas Figuras 6.9, 6.10 e 6.11, são apresentados gráficos comparativos dos controladores segundo os índices de desempenho *ITAE*, *ISE* e *IAE*.

Com base na resposta de cada controlador mediante as perturbações, os dados relativos aos índices de desempenho foram tabulados e apresentados na Tabela 6.3.



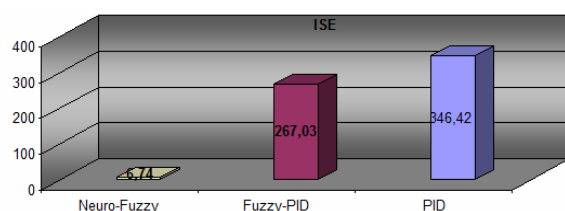


(c)

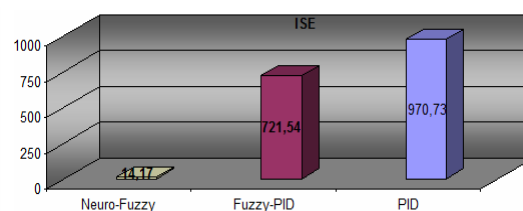


(d)

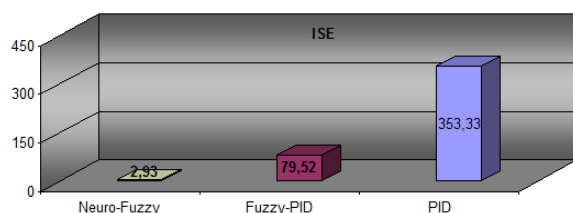
Figura 6.9 - Comparação do índice *ITAE* para os controladores PID, *fuzzy*-PID e *neuro-fuzzy* para perturbação degrau na carga (a) +20 mL/min (b) +40 mL/min (c) -20 mL/min (d) -40 mL/min.



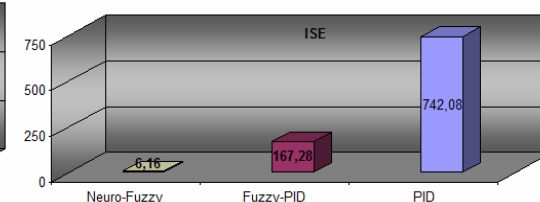
(a)



(b)

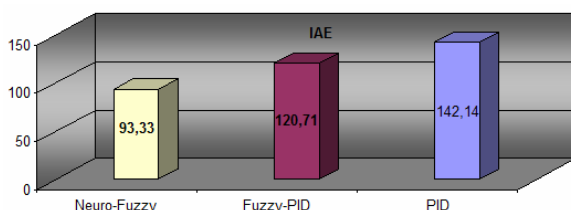


(c)

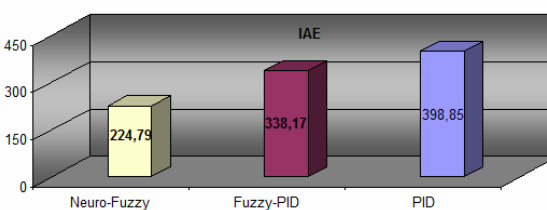


(d)

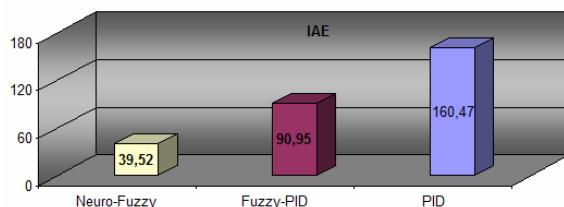
Figura 6.10 - Comparação do índice *ISE* para os controladores PID, *fuzzy*-PID e *neuro-fuzzy* para perturbação degrau na carga (a) +20 mL/min (b) +40 mL/min (c) -20 mL/min (d) -40 mL/min.



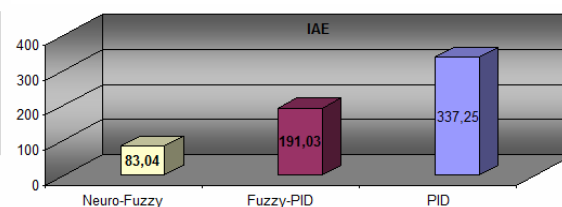
(a)



(b)



(c)



(d)

Figura 6.11 - Comparação do índice *IAE* para os controladores PID, *fuzzy*-PID e *neuro-fuzzy* para perturbação degrau na carga (a) +20 mL/min (b) +40 mL/min (c) -20 mL/min (d) -40 mL/min.

Tabela 6.3 - Índices de desempenho dos controladores PID, *fuzzy*-PID e neuro-*fuzzy* mediante perturbação na vazão de alimentação.

Perturbação	PID			Fuzzy-PID			Neuro-Fuzzy		
	ITAE	IAE	ISE	ITAE	IAE	ISE	ITAE	IAE	ISE
+20 mL/min	12663,57	142,14	346,42	10035,71	120,71	267,03	628,88	93,33	6,74
+40 mL/min	35458,91	398,85	970,73	28100,12	338,17	721,54	1698,84	224,79	14,17
-20 mL/min	10059,09	160,47	353,33	7341,42	90,95	79,52	3668,09	39,52	2,93
-40 mL/min	22130,12	337,25	742,08	15417,97	191,03	167,28	7703,04	83,04	6,16

De acordo com os dados apresentados na Tabela 6.3, comprova-se experimentalmente os resultados obtidos via simulação, sendo o controlador neuro-*fuzzy* aquele que proporcionou a melhor performance mediante às perturbações aplicadas. Para tanto, considera-se as perturbação na carga (vazão de alimentação) +20 mL/min e +40 mL/min. Para a perturbação na vazão de alimentação de + 20 mL/min o controlador PID apresentou os índices de erro *ITAE*, *IAE* e *ISE*, com valores de 2013, 52,22 e 5139 % superiores ao controlador neuro-*fuzzy*, respectivamente. Já o controlador *fuzzy*-PID teve os índices de erro *ITAE*, *IAE* e *ISE*, com valores de 1595, 29,33 e 3961,86% superiores ao controlador neuro-*fuzzy*, respectivamente.

Na situação em que a perturbação da vazão de alimentação de + 40 mL/min, o controlador PID apresentou os índices de erro *ITAE*, *IAE* e *ISE*, com valores de 2087, 77,43% e 6850% superiores ao controlador neuro-*fuzzy*, respectivamente. Já o controlador *fuzzy*-PID, teve os índices de erro *ITAE*, *IAE* e *ISE*, com valores de 1654, 50,43 e 5092% superiores ao controlador neuro-*fuzzy*, respectivamente.

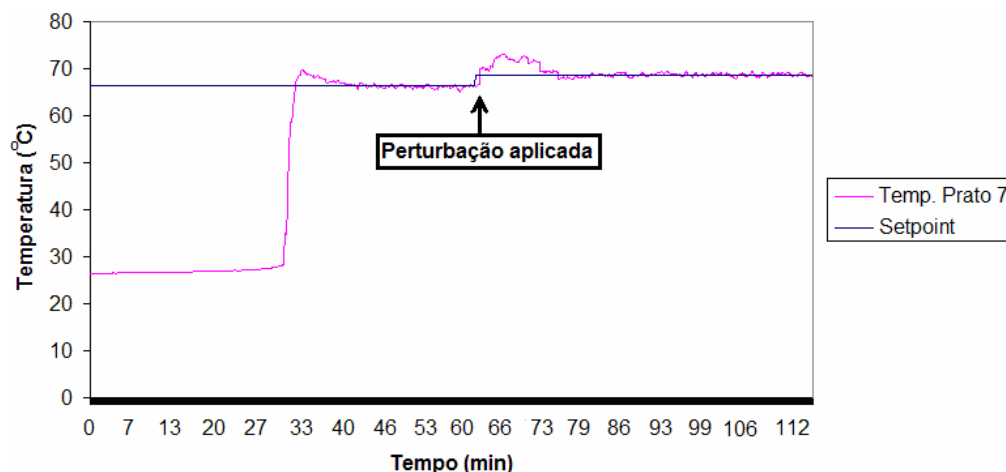
Desta forma os resultados obtidos na etapa de simulação, quando o controlador neuro-*fuzzy* apresentou os melhores resultados, são confirmados na etapa experimental.

6.2.2 Desempenho dos controladores mediante perturbação no *setpoint*.

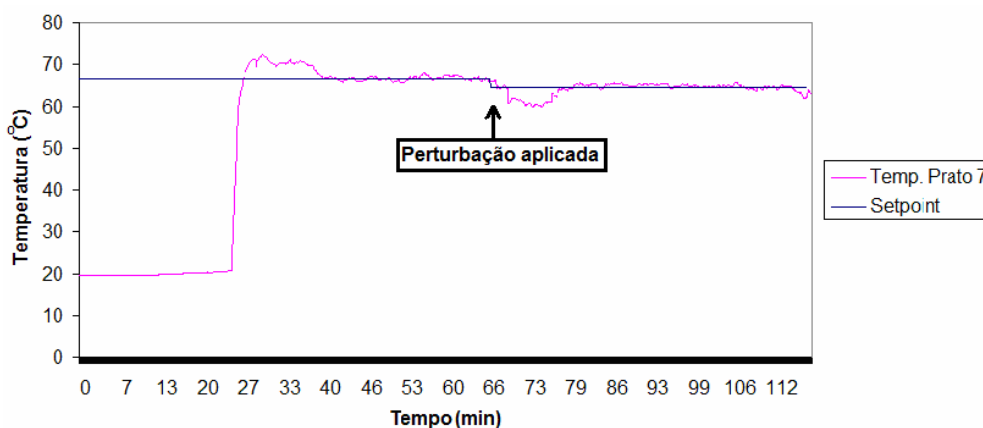
Uma possibilidade típica para a coluna de destilação é alterar o *setpoint* ao longo do processo de destilação. Esta é uma alternativa de operação da coluna de destilação, pois operando com *setpoints* diferentes, são obtidas concentrações de saídas diferentes. No caso da coluna de destilação, esta configuração é largamente utilizada industrialmente. Um exemplo típico é a coluna de destilação de etanol que pode operar com concentrações diferentes de produto final, como por exemplo, para aplicações em combustíveis e produtos de limpeza.

Segundo POIANI (1993), a mudança de valor de *setpoint* também é de extrema importância no caso de destilação multicomponente, quando o objetivo é separar os componentes intermediários da mistura. Esta alternativa é largamente utilizada na prática, pois permite a obtenção de várias frações de destilado de purezas diferentes, permitindo uma maior flexibilidade a operação da coluna de destilação. Nas Figuras 6.12, 6.13 e 6.14 são ilustradas as respostas dos controladores PID, *fuzzy*-PID e *neuro-fuzzy* mediante à perturbações de $+2^{\circ}\text{C}$ e -2°C no *setpoint*, respectivamente.

Com base nos índices de desempenho *ITAE*, *ISE* e *IAE*, a resposta dos controladores é descrita de uma forma comparativa nas Figuras 6.15, 6.16 e 6.17, respectivamente, sendo os dados relativos ao desempenho dos controladores segundo os índices *ITAE*, *IAE* e *ISE* expostos na Tabela 6.4.



(a)

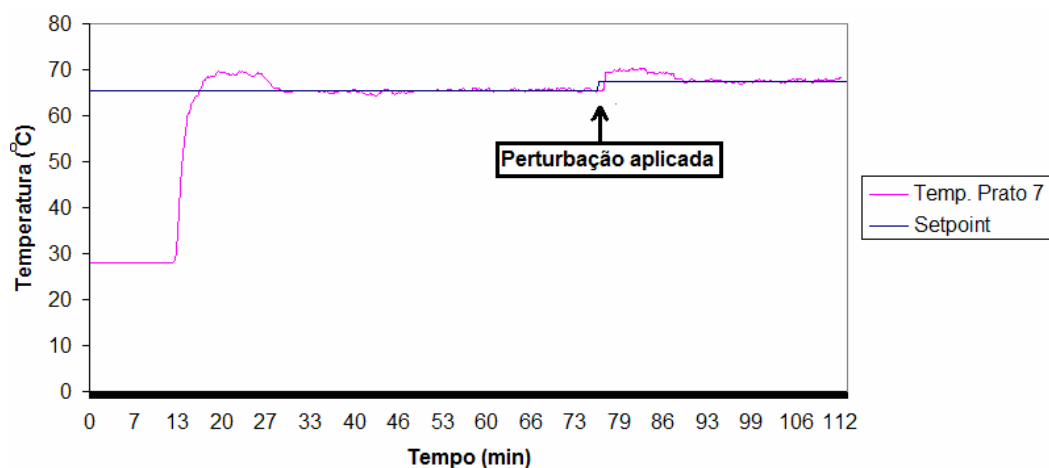


(b)

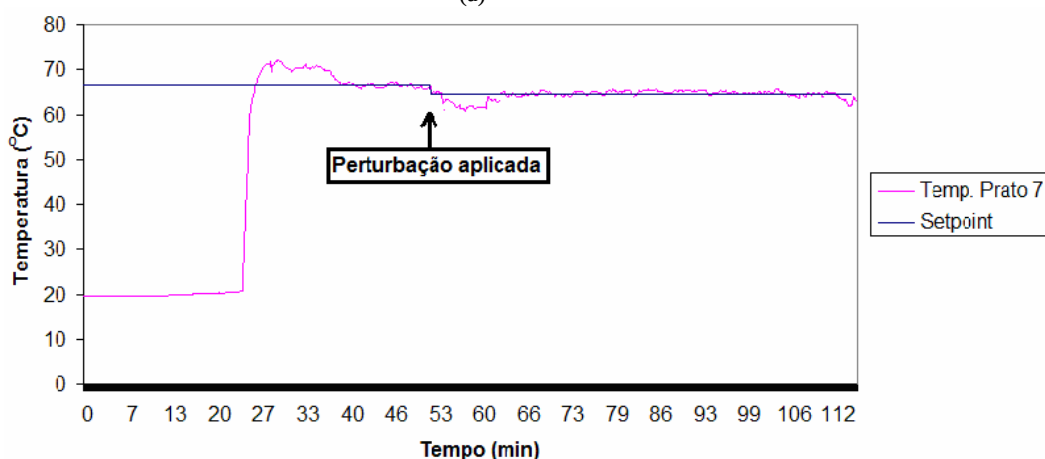
Figura 6.12 - Resposta do controlador PID para perturbação degrau no *setpoint* (a) $+2^{\circ}\text{C}$
(b) -2°C .

Observa-se claramente que para todos os índices de desempenho descritos, o controlador *neuro-fuzzy* apresentou uma melhor resposta em relação aos controladores PID

e *fuzzy*-PID, proporcionando a manutenção da variável controlada mais próximo do setpoint, o que fará com que possa ser produzido um produto final dentro das especificações e com um consumo energético menor.

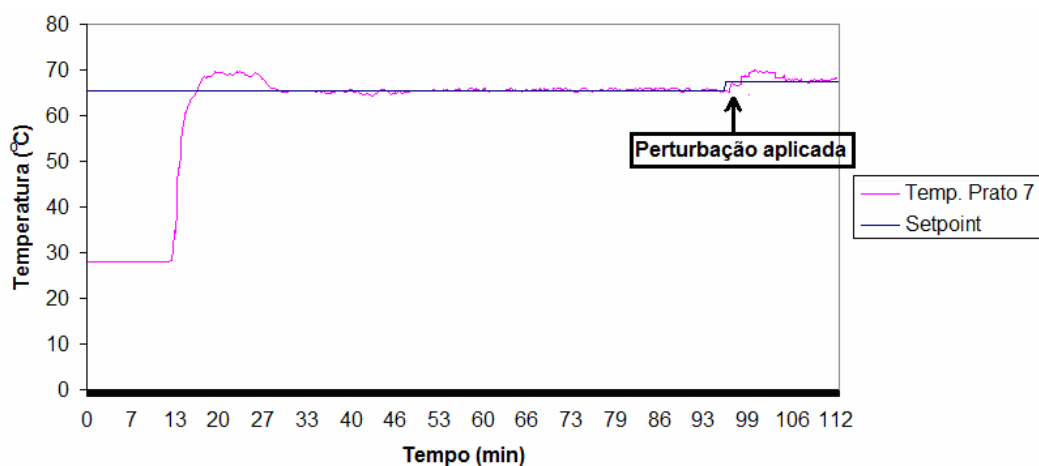


(a)

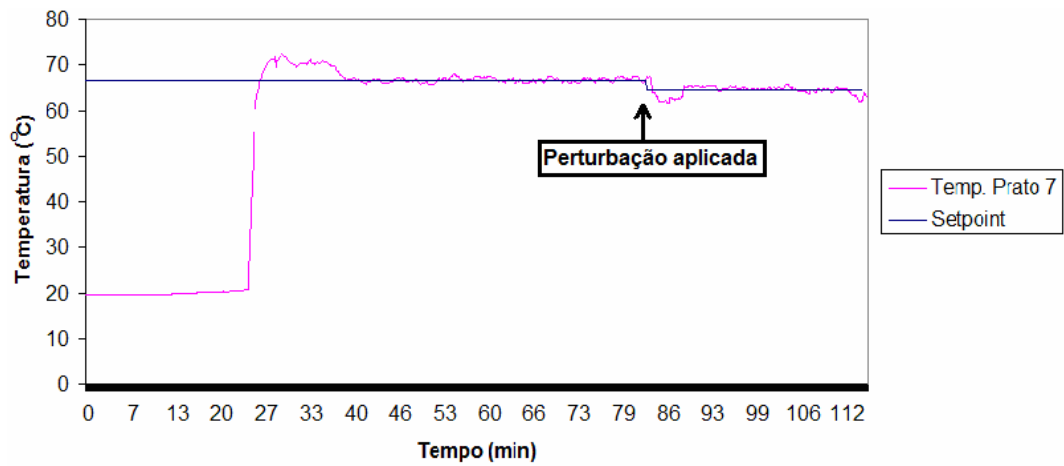


(b)

Figura 6.13 - Resposta do controlador *fuzzy*-PID para perturbação degrau no *setpoint*
(a) + 2 °C (b) -2 °C.

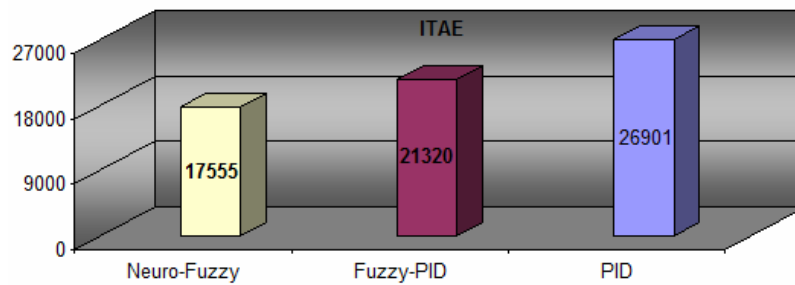


(a)

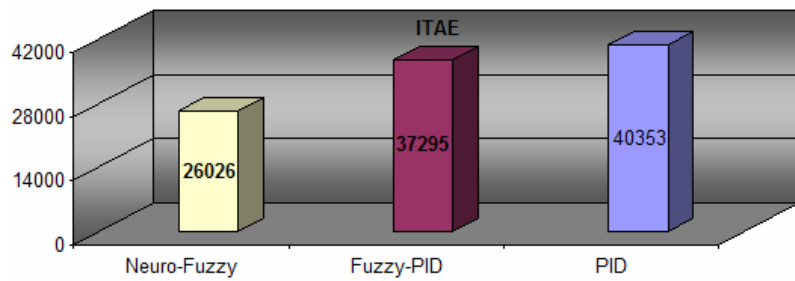


(b)

Figura 6.14 - Resposta do controlador neuro- *fuzzy* para perturbação degrau no *setpoint* (a) $+ 2^{\circ}\text{C}$ (b) -2°C .

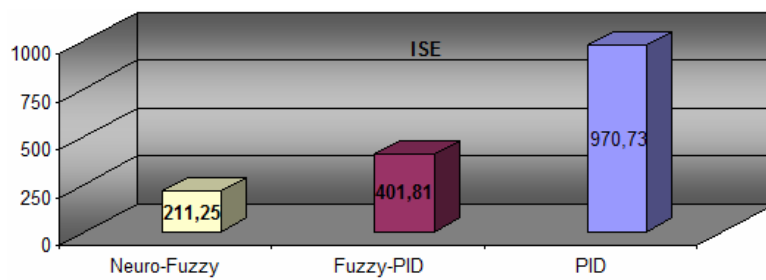


(a)



(b)

Figura 6.15 - Comparação do índice *ITAE* para os controladores PID, *fuzzy*-PID e neuro-*fuzzy* para perturbação degrau no *setpoint* (a) $+ 2^{\circ}\text{C}$ (b) -2°C .



(a)

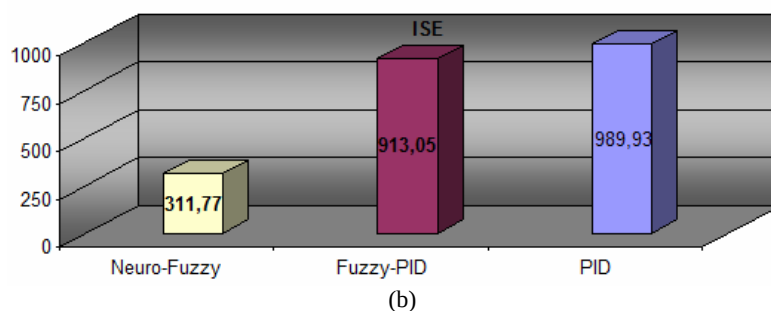


Figura 6.16 - Comparação do índice *ISE* para os controladores PID, *fuzzy*-PID e neuro-*fuzzy* para perturbação degrau no *setpoint* (a) + 2°C (b) -2 °C.

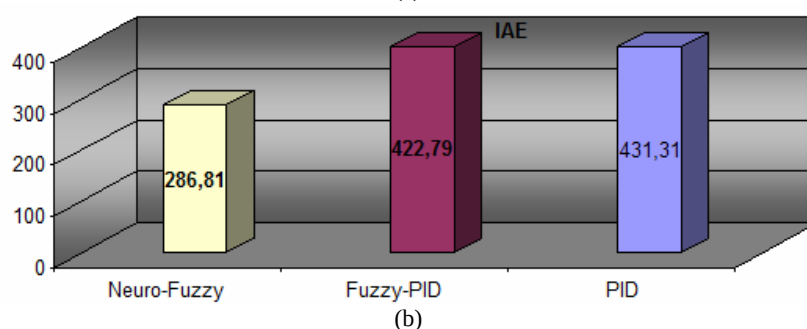
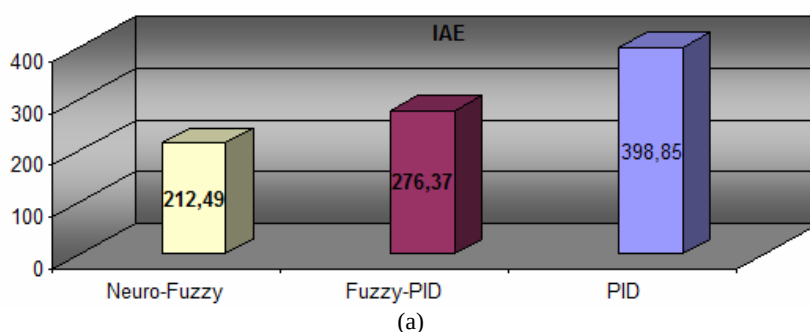


Figura 6.17 - Comparação do índice *IAE* para os controladores PID, *fuzzy*-PID e neuro-*fuzzy* para perturbação degrau no *setpoint* (a) + 2°C (b) -2 °C.

Tabela 6.4 - Índices de desempenho dos controladores PID, *fuzzy*-PID e neuro-*fuzzy* mediante perturbação no *setpoint*.

Perturbação	PID			Fuzzy-PID			Neuro-Fuzzy		
	ITAE	IAE	ISE	ITAE	IAE	ISE	ITAE	IAE	ISE
+2	26901	398,85	970,73	21319	276,37	401,81	17555	212,49	211,25
-2	40352	431,31	989,93	37294	422,79	613,05	26025	286,81	311,77

A partir da análise da resposta dos controladores ilustradas na Tabela 6.4, mediante a perturbação +2 no *setpoint*, o controlador PID apresentou os índices de erro *ITAE*, *IAE* e *ISE*, com valores de 53,23, 87,7 e 459 % superiores ao controlador neuro-*fuzzy*, respectivamente. Já o controlador *fuzzy*-PID, teve os índices de erro *ITAE*, *IAE* e

ISE, com valores de 21,44, 30,06 e 90% superiores ao controlador neuro-*fuzzy*, respectivamente. Para a perturbação -2 no *setpoint*, o controlador PID apresentou os índices de erro *ITAE*, *IAE* e *ISE*, com valores de 55,05, 50,38 e 317,51% superiores ao controlador neuro-*fuzzy*, respectivamente. O controlador *fuzzy*-PID apresentou os índices de erro *ITAE*, *IAE* e *ISE*, com valores de 43,30, 47,41 e 96,63% superiores ao controlador neuro-*fuzzy*, respectivamente.

CAPÍTULO 7 - CONCLUSÕES E PERSPECTIVAS

A partir do trabalho desenvolvido foi possível um controle automático do aquecimento da coluna, além de um monitoramento de temperaturas, capaz de transmitir dados de temperaturas a partir dos pontos chave de operação da coluna: alimentação, fundo, topo, refluxo e destilado, bem como nos pratos. Desta forma, o sistema de supervisão permitirá o acompanhamento em tempo real das temperaturas na coluna de destilação além do armazenamento das temperaturas coletadas em um banco de dados para sua posterior análise, sem que ocorram perdas, distorções inerentes a um sistema manual de operação.

As estratégias de controle desenvolvidas neste trabalho com o auxílio do *software* MATLABTM podem ser implementadas em tempo real, juntamente com o *software* de supervisão desenvolvido, por meio de uma interface gráfica com o usuário permitindo uma interação amigável com o usuário.

A interface do sistema de supervisão com o banco de dados permitiu a coleta e aquisição de dados para a obtenção do modelo do processo por meio de técnicas de identificação. As técnicas de obtenção de modelos por identificação em colunas de destilação são ferramentas auxiliares para a aplicação das estratégias de controle, pois neste tipo de processo, geralmente a obtenção de um modelo fenomenológico requer um grande esforço e pode apresentar um modelo final impreciso devido às características construtivas da coluna.

Os resultados obtidos via identificação por meio de redes neurais foram bastante satisfatórios, resultando em erros médios quadráticos de aproximadamente 0,66%. O erro médio quadrático obtido é pequeno em comparação aos obtidos pelas técnicas de identificação OE, Espaço de Estados, ARX, BJ e ARMAX que foram de: 22,86, 29,26, 39,28, 45,07, 82,79%, respectivamente. O baixo erro médio quadrático encontrado comprova que as redes neurais artificiais representam uma ferramenta matemática com excelente potencial para a identificação de sistemas complexos.

O modelo obtido poderá ser utilizado para simulação do processo juntamente com controladores, servindo como ferramenta auxiliar para a sintonia dos mesmos antes de colocá-los na planta real. O modelo também pode ser utilizado em estratégias avançadas de controle (preditivo/adaptativo) que necessitam de um modelo fiel do processo para promover um controle adequado do processo.

Com a obtenção de um modelo de processo confiável e que represente a dinâmica do processo é possível melhorar a sintonia de parâmetros de controladores via simulação e aplicar técnicas de controle avançado, fazendo com que a coluna de destilação obtenha uma significativa melhora em sua performance. Como o volume de produto processado por uma coluna de destilação e os custos envolvidos no processo são elevados, melhorias no desempenho, mesmo que pequenas, podem representar ganhos significativos a médio e longo prazo.

Os valores dos parâmetros de desempenho obtidos mostraram que o controlador neuro-*fuzzy* apresentou uma melhor resposta em todos os índices de desempenho, confirmando a sua superioridade. O inconveniente de sua utilização está na necessidade de um conjunto de dados que devem ser coletados e armazenados, bem como o tempo necessário para confecção e treinamento da rede neural artificial.

Levando em consideração estes aspectos, apesar do controlador neuro-*fuzzy* apresentar a melhor performance, os controladores PID e *fuzzy*, obtiveram um bom desempenho com pequeno tempo de resposta e *overshoot*, bem como reduzidos erros em regime permanente.

A melhor resposta dos controladores avançados (*fuzzy* e neuro-*fuzzy*) perante aos controladores tradicionais, deve-se ao fato dos controladores tradicionais utilizarem parâmetros fixos, ajustados de acordo com a média do comportamento ao longo da duração do processo. As principais vantagens dos controladores *fuzzy* e neuro-*fuzzy* advêm do fato de possuírem a capacidade de incorporar o conhecimento de especialistas humanos para especificação das regras de controle, permitindo o tratamento da imprecisão dos sinais dos sensores empregados, podendo também ser expandidos e aperfeiçoados, adicionando regras que venham melhorar o desempenho do controlador.

Os resultados experimentais vêm a confirmar os dados obtidos na simulação das estratégias de controle com o software MATLABTM em relação à eficiência dos controladores. O controlador neuro-*fuzzy* obteve uma melhor resposta mediante às perturbações aplicadas nos índices de desempenho baseados nos erros *IAE*, *ITAE* e *ISE*. Desta forma, a estratégia de controle neuro-*fuzzy* se destaca como a mais adequada para o controle da coluna de destilação, pois apresentou menor consumo de energia, mantendo o produto final (destilado) dentro das especificações por uma maior quantidade de tempo.

Este estudo comprova que ferramentas de simulação de processo em Engenharia Química são úteis para apresentar alternativas de estratégias de controle e determinar ajustes preliminares dos controladores. Finalmente, o *hardware* do sistema de controle é

selecionado e aplicado no processo. Desta forma, os controladores são sintonizados na planta, usando estimativas preliminares obtidas nos passos iniciais do projeto de controle com o auxílio de *softwares* de simulação.

Este trabalho tem como contribuições: o desenvolvimento de um sistema de supervisão de baixo custo em comparação aos sistemas tradicionais empregados na indústria, o emprego de técnicas de identificação de processos para obtenção do modelo de uma coluna de destilação, implementação e discussão dos resultados da aplicação de técnicas de controle avançado a uma coluna de destilação piloto para posterior aplicação em uma coluna de porte industrial.

Outra contribuição do trabalho proposto é em relação ao *hardware* e *software* desenvolvidos para o controle do sistema. De acordo com os levantamentos de custos descritos no capítulo 3, um sistema equivalente composto de um controlador lógico programável (CLP) convencional, teria um custo aproximado 11 vezes maior em relação ao sistema desenvolvido, o que torna este sistema extremamente viável para aplicação proposta. Se for considerada a licença do programa Delphi existente no Departamento de Engenharia Química ou a possibilidade do sistema ser implementado com *software* livre, tem-se um sistema com um custo de aproximadamente 153 vezes menor que um sistema convencional de aquisição de dados industriais.

Neste trabalho, foram identificados alguns aspectos construtivos e relativos à instrumentação que podem ser aprimorados, a fim de se obter uma melhor performance da coluna de destilação piloto estudada, melhorando suas características de operação e possibilitando o controle do processo.

Uma das dificuldades da instalação do sistema foi a colocação dos sensores, pois a natureza do material empregado na construção da coluna de destilação (vidro borosilicato) impossibilitou a colocação dos instrumentos no interior do equipamento, em contato direto com a mistura a ser destilada, sendo os sensores colocados na sua parte externa. Como a medição da temperatura interna dos pratos do equipamento é de fundamental importância para o trabalho, foi feita uma estimativa da perda de calor da parte interna para a parte externa, e a partir desta estimativa, um ajuste nas temperaturas, para que os dados coletados fossem coerentes com as temperaturas no interior da coluna.

A partir deste trabalho, desenvolveu-se uma base para o estudo e a implementação de técnicas avançadas de controle e instrumentação na coluna, pois sua construção permite alterações no projeto, possibilitando o estudo de diversos tipos de mistura em menor escala. Desta forma, torna-se possível a aplicação de estratégias de controle na coluna, o

que permitirá a minimização de transientes e do tempo de produção de produto final fora da especificação desejada.

7.1 - Perspectivas para Trabalhos Futuros

Na presente tese foram propostas e implementadas estratégias de controle em uma coluna de destilação piloto, entretanto o sistema desenvolvido possui algumas limitações, tanto na parte de *hardware* quanto no módulo didático utilizada para aplicação do controlador. Desta forma, a seguir são apresentadas sugestões para trabalhos futuros que possam acrescentar melhorias a pesquisa desenvolvida:

- Implementação de outras estratégias de controle, como controle antecipativo, bem como implementação de controladores avançados como preditivo, possibilitando conclusões acerca deste tipo de controladores para futuras aplicação em plantas de maior escala.

- Desenvolvimento de um sistema de inferência de composição por meio da temperatura, por meio do desenvolvimento de um *soft sensor*, que possibilitará a predição da concentração em cada um dos pratos, bem como na corrente de destilado.

- A instalação de válvulas eletro-pneumáticas proporcionais, juntamente com sensores de pressão e de vazão, possibilitará a implementação de estratégias de controle em outras malhas de controle. Podem ser implementadas estratégias de controle em malhas contendo razão de refluxo, vazão de alimentação, vazão de destilado e taxa de resfriamento do condensador. A aplicação de estratégias de controle em outras malhas possibilitará um controle da coluna de destilação com uma performance superior à apresentada neste trabalho, onde foi possível, por restrições operacionais, trabalhar somente com uma única malha de controle.

- Controle da temperatura do fluido de alimentação com a instalação de um pré-aquecedor para o aquecimento preliminar da mistura que ingressa na coluna. Atualmente, a mistura de alimentação é aquecida no interior do refeedor por meio de uma serpentina, não tornando possível o desacoplamento entre a temperatura do refeedor e do pré-aquecedor.

REFERÊNCIAS

ADERSEN, H. W.; KÜMMEL, M., 1989, "Discrete-time control of a binary distillation column", *Chemical Engineering Science*, v. 44, n. 11, pp. 2583-2595.

AGUIRRE, L. A.; MENDES, E. M. A. M., 1996, "Global nonlinear polynomial models: Structure, term clusters and fixed points", *International Journal of Bifurcation and Chaos*, v.6, n.2, pp. 279-294.

AGUIRRE, L. A.; RODRIGUES, G. G.; JÁCOME, C. R.F., 1998, "Identificação de Sistemas Não-Lineares Utilizando Modelos NARMAX Polinomiais - Uma Revisão e Novos Resultados", *SBA Controle & Automação*, v. 9, n.2, pp. 90-160.

AGUIRRE, L.A., 1997, "Recovering map static nonlinearities from chaotic data using dynamical models", *Physica D*, v.100, n.1. pp. 41-57.

ALVAREZ, R.J.; MONROY, R. L.; CERVANTES, I.; MORALES, A., 2000, "Novel proportional-integral derivative control configuration with application to the control of batch distillation", *Industrial Engineering Chemical Research*, v. 39, n. 2, pp. 432 - 440.

ALVES, J. L. L., 2005, *Instrumentação e Automação de Processos*. Rio de Janeiro, LTC.

ANDRADE, C. M. G., 2000, *Identificação de Processos Químicos em Malha Fechada*. Tese de doutorado, UNICAMP, Campinas, SP, Brasil.

ASSIS, A. J., 2001, *Identificação e Controle de Processos não Lineares Utilizando Redes Neurais Artificiais*. Tese de doutorado, Campinas, SP, Brasil.

ASTRÖM, K. J.; HÄGGLUND, T., 1995, *PID Controllers – Theory, Design, and Tuning*. 2nd ed. North Carolina, ISA.

BEQUETTE, B. W., 2003, *Process Control: Modeling, Design and Simulation*. Prentice Hall.

BILLINGS, S. A., 1980, "Identification of nonlinear systems – A survey", *IEEE Proceedings, Part D*, v. 127, n. 6, pp. 272-285.

BRABILLA, A.; D'ELIA, L., 1992, "Multivariable controller for distillation columns in the presence of strong directionality and model error", *Industrial Engineering Chemical Research*, v. 32, n. 2, pp. 536-543.

CAÑETE, J.F.; CORDERO, T.; GUIJAS D.; ALARCON, J., 2004, "An Adaptive Neuro-Fuzzy Approach to Control a Distillation Column", *Computer Science*, v.9, n.2, pp. 112-118.

CASDAGLI, M., 1989, "Nonlinear prediction of chaotic time series", *Physica D*, v.35, n.4, pp. 335-356.

COUGHANOWR, D. R.; LEBLANC, L. B., 2008, *Análise e Controle de Processos*. 3rd ed. MacGraw-Hill.

CHEN, S.; BILLINGS, S. A., 1989, "Representation of non-linear systems: the NARMAX model", *International Journal of Control*, v. 49, n. 3, pp. 1013-1032.

CHEN, S.; BILLINGS, S. A.; LUO, W., 1989, "Orthogonal least squares methods and their application to non-linear system identification", *International Journal of Control*, v. 50, n. 5, pp. 1873-1896.

CHETOUANI, Y., 2007, "Using Artificial Neural networks for the modeling of a distillation column", *International Journal of Computer Science & Applications Technomathematics Research Foundation*, v. 4, n. 3, pp. 119-133.

DALLAS SEMICONDUCTOR., 2001, *DS1820: 1-wire digital thermometer*. Disponível em: <<http://www.maxim-ic.com>>.

DESHPANDE, P. B., 1985, *Distillation dynamics and control*. ISA-Instrument Society of America.

DOMA, M. J.; TAYLOR, P. A.; VERMEER, P.J., 1996, "Closed loop identification of MPC models for MIMO process using genetic algorithms and dithering one variable a time: application to an industrial distillation tower", *Computers & Chemical Engineering*, v. 20, n. 1, pp. 1035-1040.

ENGİN, S. N.; KUVULMAZ, J.; OMURLU, V. E., 2004, "Fuzzy control of an ANFIS model representing a nonlinear liquid-level system", *Neural Computer & Application*, v.13, pp. 202-210.

FRANCHI, C. M., 2007, *Instrumentação de uma Coluna de Destilação*. Dissertação de Mestrado, UEM, Maringá, PR, Brasil.

GOUESBET, G.; LETELLIER, C., 1994, "Global vector field reconstruction by using a multivariate polynomial L2 approximation on nets", *Physical Review E*, v.49, n. 6, pp. 4955-4972.

HAGAN, M. T.; DEMUTH, H. B.; BEAL, M., 1996, *Neural Network Design*. Boston, PWS Publishing Company.

HAYKIN, S., 1994, *Neural Networks: A Comprehensive Foundation*. New York, Macmillan.

HAYKIN, S., 2001, *Redes Neurais. Princípios e Prática*. Porto Alegre, Bookman.

HEUBERGER, P. S. C.; VAN DEN HOF, P. M. J.; WAHLBERG, B., 2005. *Modelling and Identification with Rational Orthogonal Basis Functions*. Springer.

HILL, A. G., 2001, "Avoiding split-personality in 2 multiplied by 2 multivariable control", *ISA Tech. Expo. Technology Update Conference*, v. 413, pp. 79-90.

HOLMAN, J. P., 2002, *Heat transfer*. 9th ed. New York, McGraw-Hill.

JACQUES, K. A., 1995, *The Alcohol textbook*. 4th ed. England, Nottingham University Press.

JANA, A. K.; GANGULY, S.; SAMANTA, A. N., 2007, "Non-linear control of a distillation column coupled with MPC and state observer", *International Journal of Modeling, Identification and Control*, v. 2, n. 2, pp. 88-99.

JANG, H.; KIM, K., 1994, "Identification of loudspeaker nonlinearities using the NARMAX modeling technique", *Journal of Audio Engineering Society*, v. 42, n. 1, pp. 50-59.

JANG, R. J. S.; SUN, C. T.; MIZUTANI, E., 1997, *Neuro-Fuzzy and Soft Computing: A Computational Approach to Learning and Machine Intelligence*. Prentice Hall.

JANTZEN, J., 1998, *Tuning of Fuzzy PID Controllers*. Lyngby, Technical University of Denmark Press.

KISTER, H. Z., 1992, *Distillation Design*. California, MacGraw-Hill.

KORENBERG, M.; BILLINGS, S. A.; LIU, Y. P.; MCILROY, P. J., 1988, "Orthogonal parameter estimation algorithm for non-linear stochastic systems", *International Journal of Control*, v. 48, n. 1, pp. 193-210.

KUO, B.; GOLNARAGHI, F., 2009, *Automatic Control Systems*. 9th edition, Wiley.

LI, H. X., 1997, "A Comparative Design and Tuning for Conventional Fuzzy Control", *Ieee Transactions On Systems, Man, and Cybernetics—Part B: Cybernetics*, v. 27, n. 5, pp. 230-242.

LJUNG, L., 1987, *System Identification – Theory for the User*. New Jersey, Prentice Hall.

LU, J.; CHEN, G.; YING, H., 2001, "Predictive fuzzy PID control: theory design and simulation", *Information Sciences*, v. 137, pp. 157-187.

LUYBEN, W.L., 1996, *Process Modeling, Simulation and Control for Chemical Engineers*. 2nd ed. New York, McGraw-Hill.

MAHFOUF, M.; KANDIAH, S.; LINKENS, D. A., 2002, "Fuzzy model-based predictive control using an ARX structure with feedforward", *Fuzzy sets and systems*, v. 125, pp. 39-59.

MAITELLI, A. L.; FILHO O. G., 2003, "Controlador Híbrido Indireto Baseado em Redes Neurais-Parte I: Desenvolvimento e Implementação". In: *Anais do VI Simpósio Brasileiro de Automação Inteligente - SBAI*, pp. 183-188, Bauru-SP, Setembro.

MARANGONI, C., 2005, *Implementação de uma Estratégia de Controle com Ação Distribuída em uma coluna de Destilação*. Tese de Doutorado. CPGENQ - Departamento de Engenharia Química - Universidade Federal de Santa Catarina - UFSC, Santa Catarina, SC, Brasil.

MASRI, S.F.; CHASSIAKOS, A.G.; CAUGHEY, T. K., 1993, "Identification of nonlinear dynamic systems using neural networks. Transactions", *ASME - Journal of Applied Mechanics*, v. 60, n. 1, pp. 123-133.

MOGHADASSI, A. R.; PARVIZIAN, F.; HOSSEINI, S. M.; FAZLALI, A. R., 2009, "A New Approach for Estimation of PVT Properties of Pure Gases Based on Artificial Neural Network Model", *Braz. J. Chem. Eng.* [online], v. 26, n. 1, pp. 199-206.

NELLES, O., 2001. *Nonlinear System Identification*. Springer-Verlag.

NISENFELD, E. A.; SEEMAN, R. C., 1981, *Distillation Columns*. ISA Monograph.

NOORAI, A.; ROMAGNOLI, J. A. FIGUEROA, J., 1999, "Process, identification, uncertainty characterization and robustness analysis of a pilot scale distillation column", *Journal of Process Control*, v. 9, n. 3. pp. 247-264.

NOSHIRO, M.; FURUYA, M.; LINKENS, D.; GOODE, K., 1993, "Nonlinear identification of PCO₂ control system in man", *Computer Methods and Programs in Biomedicine*, v. 40, n. 1, pp. 189-202.

OGATA, K., 2003, *Engenharia de Controle Moderno*. 4ª ed. Prentice Hall.

OLIVEIRA, G. H. C.; CAMPELLO, R. J. G. B.; AMARAL, W. C., 2007a, "Identificação e Controle de Processos via Desenvolvimentos em Séries Ortonormais. Parte A: Identificação", *Controle & Automação*, v. 18, n. 3, pp. 301-320.

OLIVEIRA, G. H. C.; CAMPELLO, R. J. G. B.; AMARAL, W. C., 2007b, "Identificação e controle de processos via desenvolvimentos em séries ortonormais. Parte B: Controle preditivo", *Controle & Automação*, v. 18, n. 3, pp. 322–336.

PADILHA, P. A.; MUMAN, F. C.; ALBERTO, M. T., 2008, *Métodos de Identificación Dinâmica*, Universidade de Oriente, Facultad de Ingenieria Eléctrica, Santiago de Cuba.

PASINO, K.; YURKOVICH, S., 1997, *Fuzzy Control*. Addison Wesley Publishing Company.

PEDRET, C.; VILANOVA, R.; MORENO, R. SERRA, I., 2002, "A refinement procedure for PID controller tuning", *Computers & Chemical Engineering*, v. 26, n. 2, pp. 903-908.

PERRY, R.H; GREEN, D. H., 1997, *Perry's Chemical engineers handbook*, 6th ed. New York, McGraw-Hill.

PEDROSA, L. S., 1998, *Controle Adaptativo de uma Coluna Piloto de Destilação em Batelada com Inferenciação de Composição através de Redes Neurais Artificiais*, dissertação de mestrado, UNICAMP, Campinas, SP, Brasil.

PEÑA, S. R. S.; CASÍN, J. Q.; CAYUELA, V. P., 2007, *Identification and Control - The Gap between Theory and Practice*. Califórnia, Springer.

POIANI, L. M., 1993 *Dinâmica e Controle de Colunas de Destilação – Aplicações a*

Sistemas de Elevada Pureza, Tese de doutorado, UNICAMP, Campinas, SP, Brasil.

RADEMAKER, O.; RIJNSDORP, J.E.; MAARLEVELD, A., 1975, *Dynamics and control of continuous distillation units*. New York, Elsevier.

RAVI, V.; REDDY, P. J.; DUTTA, D., 1997, "Application of *fuzzy non linear goal programming* to a refinery model", *Computers & Chemical Engineering*, v. 22, pp. 709-712.

REMBERG, C.; INTEMANN, K; FETT, F. N.;WOZNY, G., 1994, "Decision supporting system for the design of control system for distillation columns", *Computers & Chemical*, v. 18, n. 1, pp. 409-413.

SANDELIN, P. M.; TOIVONEN, H.T.; OSTERAS, M.; WALLER, K. V., 1991, "Robust multiobjective linear quadratic control of distillation using low-order controllers", *Chemical Engineering Science*, v. 46, n. 11, pp. 2815-2827.

SEBORG, D.E.; EDGAR, T. F.; MELLICHAMP, D. A., 2003, *Process Dynamics and Control*, 2nd ed. John Wiley & Sons.

SINCLAIR, I. R., 2001, *Sensors and transducers*. 4th ed. Newnes.

SHINSKEY, F.G., 1996, *Process Control Systems*. 4th edition. New York, McGraw-Hill.

SHUNTA, J. P., 1995, "Linearized temperature control of distillation columns with nonlinear temperature profiles", *Advanced. Instrumentation and Control*, v. 50, n. 3 pp. 905-916.

SJÖBERG, J.; ZHANG, Q.; LJUNG, L.; BENVENISTE, A.; DEYLON, B.; GLORENNEC, P.; HJALMARSSON H.; JUDITSKY A., 1995, "Nonlinear Black-Box Models in System Identification: a Overview", *Automatica*, v. 3, n. 4, pp. 1691-1725.

SMITH, C. A.; CORRIPIO A., 2008, *Princípios e Prática do Controle Automático de Processo*. 3ª ed. LTC.

SOUZA, D. J., 2003, *Desbravando o PIC: ampliado e atualizado para PIC 16F628A*. 6ª ed. São Paulo, Érica.

SPANDRI, R., 2003, “Sintonia de controladores regulatórios”, *Boletim técnico da Petrobrás- RJ*, v. 46, n. 314, pp. 383-410.

STRANG, G., 1989, “Wavelets and dilation equations: a brief introduction”, *SIAM Review*, v. 31, n. 4, pp. 614-627.

SU, H. T.; MCAVOY, T.J., 1993, “Neural Model Predictive Control of Nonlinear Chemical Processes”, In: *IEEE, Proceedings of the International Symposium on Intelligent Control*, pp. 358-363.

THE MATHWORKS, 1998, *Neural Toolbox User’s Guide – Version 3.0*, USA.

VAS, P., 1999, *Artificial-Intelligence-Based Electrical Machines and Drives: Application of Fuzzy, Neural, Fuzzy-Neural and Genetic-Algorithm-Based Techniques*. Oxford Science Publications.

VERMEER, P. J.; MORRIS, A. J; SHAH, S. L., 1988, "Adaptive PID control. A pole placement algorithm with a single controller tuning parameter", *IFAC – Proceedings Series*, v. 6, n. 4, pp. 159-164.

VESTER, M. T.; VAN DER LINDEN, R. J. P. ; PANGELS, J. L. A., 1993, “Control Design for an Industrial Distillation Column”, *Computers Chem. Eng.*, v.17, n. 5, pp. 609-615.

WAHLBERG, B.; MÄKILÄ, P. M., 1996, “Approximation of stable linear dynamical systems using Laguerre and Kautz functions”, *Automatica*, v. 32, n. 5, pp. 693-708.

WERBOS, P., 1974, *Beyond regression: New tools for prediction and analysis in the behavioral sciences*, Tese de Doutorado, Harvard University, Cambridge, MA, USA.

WILSON, I.D., 2000, *Encyclopedia of separation science*, Academic Press Inc.

ZADEH, L. A., 1965, "Fuzzy sets", *Information and Control*, v. 8, n. 2, pp. 338-353.

8. APÊNDICE – PROGRAMAS DESENVOLVIDOS

8.1 – Programa Microcontrolador Pic16f877A (Placa de Controle e Aquisição de Dados)

```
#include ".\teste.h"
#include <math.h>
#include <regs_16f87x.h>
#include <stdlib.h>
#include <input.c>
int16 i,i2=0,i3,i4=0,i5=0,i6=0,i7=0;
char y[52]={""},prop[10]={""},in[10]={""},der[10]={""},set[10]={""},aux=' ',manual[4]={""},ind[2]={""};
char banda2[3]={""};
int dt=9,pb=0,indicador=1;
signed int32 integral=0,aatual=0,proporcional=0;
int16 setp=0,setp2=0,banda=0;
signed int16 erro=0,erroant=0,man=0,derivativo=0,derian=0;
float sensor1=0, k=0,ti=0,td=0,td2=0,ti2=0;
signed int32 out=0,out2=57535;
#priority RDA,timer1,EXT
#int_RDA

void RDA_isr()
{ char x;
  disable_interrupts(INT_RDA);
  x=getc();
  y[i2]=x;
  if(y[0]=='z')
  {
    if(y[i2]=='f')
    {
      for(i6=1;i6<5;++i6)
      {
        if(y[i6]!='f')
          manual[i7]=y[i6];
        else
          i6=5;
        i7++;
      }
      for(i6=i7-1;i6<5;++i6)//zera string
        manual[i6]=aux;
      man=atol(manual);
      indicador=0;
      i7=0;
      i2=0;
    }
  }
  if(y[0]=='p')
  {
    if(y[i2]=='f')
    {
      for(i2=1;i2<11;++i2)// proporcional
      {
        if(y[i2]!='i')
          prop[i3]=y[i2];
        else
          i2=11;
        i3++;
      }
      for(i5=i3-1;i5<11;++i5)// zera string
        prop[i5]=aux;
      i2=0;
      i4=i3+1;
      i3=0;
      for(i2=i4;i2<22;++i2)// integral
      {
        if(y[i2]!='d')
          in[i3]=y[i2];
        else
          i2=22;
        i3++;
      }
    }
  }
}
```

```

    }
    i4=i4+i3;
    i2=0;
    for(i5=i3-1;i5<11;++i5)// string
    in[i5]=aux;
    i3=0;
    for(i2=i4;i2<33;++i2)// derivativo
    {
        if(y[i2]!='s')
            der[i3]=y[i2];
        else
            i2=33;
        i3++;
    }
    i2=0;
    for(i5=i3-1;i5<11;++i5)//zera o resto da string
    der[i5]=aux;
    i4=i4+i3;
    i3=0;

    for(i2=i4;i2<44;++i2)//pega valor do set point
    {
        if(y[i2]!='b')
            set[i3]=y[i2];

        else
            i2=44;
        i3++;
    }
    i2=0;

    for(i5=i3-1;i5<11;++i5)//zera o resto da string
    set[i5]=aux;
    i4=i4+i3;
    i3=0;

    for(i2=i4;i2<48;++i2)//pega valor da banda
    {
        if(y[i2]!='h')
            banda2[i3]=y[i2];
        else
            i2=48;
        i3++;
    }
    for(i5=i3-1;i5<3;++i5)//zera o resto da string
    banda2[i5]=aux;
    i4=i4+i3;
    i3=0;
    for(i2=i4;i2<51;++i2)//pega valor do indicador
    {
        if(y[i2]!='f')
            ind[i3]=y[i2];
        else
            i2=51;
        i3++;
    }
    for(i5=i3-1;i5<2;++i5)//zera o resto da string
    ind[i5]=aux;
    i3=0;
    i2=0;
    k=atof(prop);
    ti=atof(in);
    td=atof(der);
    setp=atol(set);
    pb=atoi(banda2);
    indicador=atoi(ind);
    ti2=1/ti;
    td2=td*1000;
}
}
i2++;
enable_interrupts(INT_RDA);
}
#endif
void TIMER1_isr()
{
    output_high(pin_c2);

```

```

set_timer1(0);
}
#int_EXT
void EXT_isr()
{
    out2=57535+out;
    set_timer1(out2);
    output_low(pin_c2);
}
void controle()
{
    // inicia controlador PID
    // buffer PID
    i3=0;
    i2=0;
    i4=0;
    i5=0;
    // Conversão do valores k=prop,ti=in,td=der
    setp2=setp*40; //0 a 150°C variação de 0 a 6000 para tensão 0 a 180v
    //Sensor do refeedor
    erro=setp2-sensor1;// pid reverso
    if(erro<0)
        erro=0;
    proporcional=k*erro;
    aatual=(erroant+erro)*dt;
    //aatual=aatual+aant;
    //aatual=aatual+aant;
    if(aatual>6000)
        aatual=6000;
    if(aatual<0)
        aatual=0;
    integral=(k*ti2)*aatual;
    if(integral>6000)
        integral=6000;
    if(integral<0)
        integral=0;
    if(proporcional>6000)
        proporcional=6000;
    if(proporcional<0)
        proporcional=0;
    derian=(erro-erroant)/dt;
    derivativo=k*td2*derian;
    derivativo=derivativo/1000;
    erroant=erro;
    if(derivativo>0)
        derivativo=0;
    if(derivativo<-6000)
        derivativo=-6000;
    banda=setp/pb;
    out=proporcional+integral+derivativo;

    if(banda>0)
        out=out/banda;
    if(out>6000)
        out=6000;

    if(out<0)
        out=0;
    out=out*0.8;// limitando 80%
    set_pwm1_duty(out);
}
boolean reset_1w(int8 sens)
// reseta os dispositivos no barramento
{
    boolean presente;
    if (sens==1) { output_low (pin_d0);
    enable_interrupts(INT_EXT);
        enable_interrupts(INT_TIMER1);
        delay_us(480);
        output_float (pin_d0);
        delay_us(60);
        disable_interrupts(INT_EXT);
        disable_interrupts(INT_TIMER1);
        presente = input (pin_d0);
    }
    if (sens==2) { output_low (pin_b1);

```

```

enable_interrupts(INT_EXT);
    enable_interrupts(INT_TIMER1);
    delay_us(480);
    output_float (pin_b1);
    delay_us(60);
    disable_interrupts(INT_EXT);
    disable_interrupts(INT_TIMER1);
    presente = input (pin_b1);
}
if (sens==3) { output_low (pin_b2);
enable_interrupts(INT_EXT);
    enable_interrupts(INT_TIMER1);
    delay_us(480);
    output_float (pin_b2);
    delay_us(60);
    disable_interrupts(INT_EXT);
    disable_interrupts(INT_TIMER1);
    presente = input (pin_b2);
}
if (sens==4) { output_low (pin_b3);
enable_interrupts(INT_EXT);
    enable_interrupts(INT_TIMER1);
    delay_us(480);
    output_float (pin_b3);
    delay_us(60);
    disable_interrupts(INT_EXT);
    disable_interrupts(INT_TIMER1);
    presente = input (pin_b3);
}
if (sens==5) { output_low (pin_b4);
enable_interrupts(INT_EXT);
    enable_interrupts(INT_TIMER1);
    delay_us(480);
    output_float (pin_b4);
    delay_us(60);
    disable_interrupts(INT_EXT);
    disable_interrupts(INT_TIMER1);
    presente = input (pin_b4);
}
if (sens==6) { output_low (pin_b5);
enable_interrupts(INT_EXT);
    enable_interrupts(INT_TIMER1);
    delay_us(480);
    output_float (pin_b5);
    delay_us(60);
    disable_interrupts(INT_EXT);
    disable_interrupts(INT_TIMER1);
    presente = input (pin_b5);
}
if (sens==7) { output_low (pin_b6);
enable_interrupts(INT_EXT);
    enable_interrupts(INT_TIMER1);
    delay_us(480);
    output_float (pin_b6);
    delay_us(60);
    disable_interrupts(INT_EXT);
    disable_interrupts(INT_TIMER1);
    presente = input (pin_b6);
}
if (sens==8) { output_low (pin_b7);
enable_interrupts(INT_EXT);
    enable_interrupts(INT_TIMER1);
    delay_us(480);
    output_float (pin_b7);
    delay_us(60);
    disable_interrupts(INT_EXT);
    disable_interrupts(INT_TIMER1);
    presente = input (pin_b7);
}
if (sens==9) { output_low (pin_a0);
    enable_interrupts(INT_EXT);
    enable_interrupts(INT_TIMER1);
    delay_us(480);
    output_float (pin_a0);
    delay_us(60);
    disable_interrupts(INT_EXT);

```

```

        disable_interrupts(INT_TIMER1);
        presente = input (pin_a0);
    }
    if (sens==10) { output_low (pin_a1);
        enable_interrupts(INT_EXT);
        enable_interrupts(INT_TIMER1);
        delay_us(480);
        output_float (pin_a1);
        delay_us(60);
        disable_interrupts(INT_EXT);
        disable_interrupts(INT_TIMER1);
        presente = input (pin_a1);
    }
    if (sens==11) { output_low (pin_a2);
        delay_us(480);
        output_float (pin_a2);
        delay_us(60);
        presente = input (pin_a2);
    }
    if (sens==12) { output_low (pin_a3);
        delay_us(480);
        output_float (pin_a3);
        delay_us(60);
        presente = input (pin_a3);
    }
    if (sens==13) { output_low (pin_a4);
        delay_us(480);
        output_float (pin_a4);
        delay_us(60);
        presente = input (pin_a4);
    }
    if (sens==14) { output_low (pin_a5);
        delay_us(480);
        output_float (pin_a5);
        delay_us(60);
        presente = input (pin_a5);
    }
    if (sens==15) { output_low (pin_e0);
        delay_us(480);
        output_float (pin_e0);
        delay_us(60);
        presente = input (pin_e0);
    }
    if (sens==16) { output_low (pin_e1);
        delay_us(480);
        output_float (pin_e1);
        delay_us(60);
        presente = input (pin_e1);
    }
    // delay_us (240);
    return (presente);
    // 0 = dispositivo presente
    // 1 = nenhum dispositivo detectado
}

void alimenta_barramento_1w (int8 sens)
// força o barramento em nível alto
// utilizado com dispositivos alimentados no modo parasita
{
    if (sens==1) { output_high (pin_d0);
        enable_interrupts(INT_EXT);
        enable_interrupts(INT_TIMER1);
        delay_ms(1000);
        disable_interrupts(INT_EXT);
        disable_interrupts(INT_TIMER1);
        output_float (pin_d0);
    }
    if (sens==2) { output_high (pin_b1);
        enable_interrupts(INT_EXT);
        enable_interrupts(INT_TIMER1);
        delay_ms(1000);
        disable_interrupts(INT_EXT);
        disable_interrupts(INT_TIMER1);
        output_float (pin_b1);
    }
    if (sens==3) { output_high (pin_b2);

```

```

        enable_interrupts(INT_EXT);
        enable_interrupts(INT_TIMER1);
        delay_ms(1000);
        disable_interrupts(INT_EXT);
        disable_interrupts(INT_TIMER1);
        output_float (pin_b2);
    }
    if (sens==4) { output_high (pin_b3);
        enable_interrupts(INT_EXT);
        enable_interrupts(INT_TIMER1);
        delay_ms(1000);
        disable_interrupts(INT_EXT);
        disable_interrupts(INT_TIMER1);
        output_float (pin_b3);
    }
    if (sens==5) { output_high (pin_b4);
        enable_interrupts(INT_EXT);
        enable_interrupts(INT_TIMER1);
        delay_ms(1000);
        disable_interrupts(INT_EXT);
        disable_interrupts(INT_TIMER1);
        output_float (pin_b4);
    }
    if (sens==6) { output_high (pin_b5);
        enable_interrupts(INT_EXT);
        enable_interrupts(INT_TIMER1);
        delay_ms(1000);
        disable_interrupts(INT_EXT);
        disable_interrupts(INT_TIMER1);
        output_float (pin_b5);
    }
    if (sens==7) { output_high (pin_b6);
        enable_interrupts(INT_EXT);
        enable_interrupts(INT_TIMER1);
        delay_ms(1000);
        disable_interrupts(INT_EXT);
        disable_interrupts(INT_TIMER1);
        output_float (pin_b6);
    }
    if (sens==8) { output_high (pin_b7);
        enable_interrupts(INT_EXT);
        enable_interrupts(INT_TIMER1);
        delay_ms(1000);
        disable_interrupts(INT_EXT);
        disable_interrupts(INT_TIMER1);
        output_float (pin_b7);
    }
    if (sens==9) { output_high (pin_a0);
        enable_interrupts(INT_EXT);
        enable_interrupts(INT_TIMER1);
        delay_ms(1000);
        disable_interrupts(INT_EXT);
        disable_interrupts(INT_TIMER1);
        output_float (pin_a0);
    }
    if (sens==10) { output_high (pin_a1);
        enable_interrupts(INT_EXT);
        enable_interrupts(INT_TIMER1);
        delay_ms(1000);
        disable_interrupts(INT_EXT);
        disable_interrupts(INT_TIMER1);
        output_float (pin_a1);
    }
    /* if (sens==11) { output_high (pin_a2);
        delay_ms(1000);
        output_float (pin_a2);
    }
    if (sens==12) { output_high (pin_a3);
        delay_ms(1000);
        output_float (pin_a3);
    }
    /* if (sens==13) { output_high (pin_a4);
        delay_ms(1000);
        output_float (pin_a4);
    }
    if (sens==14) { output_high (pin_a5);

```

```

        delay_ms(1000);
        output_float (pin_a5);
    }
    if (sens==15) { output_high (pin_e0);
        delay_ms(1000);
        output_float (pin_e0);
    }
    if (sens==16) { output_high (pin_e1);
        delay_ms(1000);
        output_float (pin_e1);
    }
}
boolean le_bit_1w (int8 Sens)
// lê um bit do barramento 1-wire
{
    boolean df;
    // dá um pulso na linha, inicia quadro de leitura
    if(sens==1) { output_low (pin_d0);
        output_float (pin_d0);
        enable_interrupts(INT_EXT);
        enable_interrupts(INT_TIMER1);
        delay_us (15);
        disable_interrupts(INT_EXT);
        disable_interrupts(INT_TIMER1);    // aguarda o dispositivo colocar o dado na saída
        df = (input(pin_d0)); // retorna o dado
    }
    if(sens==2) { output_low (pin_b1);
        output_float (pin_b1);
        enable_interrupts(INT_EXT);
        enable_interrupts(INT_TIMER1);
        delay_us (15);
        disable_interrupts(INT_EXT);
        disable_interrupts(INT_TIMER1);    // aguarda o dispositivo colocar o dado na saída
        df = (input(pin_b1)); // retorna o dado
    }
    if(sens==3) { output_low (pin_b2);
        output_float (pin_b2);
        enable_interrupts(INT_EXT);
        enable_interrupts(INT_TIMER1);
        delay_us (15);
        disable_interrupts(INT_EXT);
        disable_interrupts(INT_TIMER1);    // aguarda o dispositivo colocar o dado na saída
        df = (input(pin_b2)); // retorna o dado
    }
    if(sens==4) { output_low (pin_b3);
        output_float (pin_b3);
        enable_interrupts(INT_EXT);
        enable_interrupts(INT_TIMER1);
        delay_us (15);
        disable_interrupts(INT_EXT);
        disable_interrupts(INT_TIMER1);    // aguarda o dispositivo colocar o dado na saída
        df = (input(pin_b3)); // retorna o dado
    }
    if(sens==5) { output_low (pin_b4);
        output_float (pin_b4);
        enable_interrupts(INT_EXT);
        enable_interrupts(INT_TIMER1);
        delay_us (15);
        disable_interrupts(INT_EXT);
        disable_interrupts(INT_TIMER1);    // aguarda o dispositivo colocar o dado na saída
        df = (input(pin_b4)); // retorna o dado
    }
    if(sens==6) { output_low (pin_b5);
        output_float (pin_b5);
        enable_interrupts(INT_EXT);
        enable_interrupts(INT_TIMER1);
        delay_us (15);
        disable_interrupts(INT_EXT);
        disable_interrupts(INT_TIMER1);    // aguarda o dispositivo colocar o dado na saída
        df = (input(pin_b5)); // retorna o dado
    }
    if(sens==7) { output_low (pin_b6);
        output_float (pin_b6);
        enable_interrupts(INT_EXT);
        enable_interrupts(INT_TIMER1);
        delay_us (15);
        disable_interrupts(INT_EXT);
    }
}

```

```

        disable_interrupts(INT_TIMER1);    // aguarda o dispositivo colocar o dado na saída
        df = (input(pin_b6)); // retorna o dado
    }
    if(sens==8) { output_low (pin_b7);
        output_float (pin_b7);
        enable_interrupts(INT_EXT);
        enable_interrupts(INT_TIMER1);
        delay_us (15);
        disable_interrupts(INT_EXT);
        disable_interrupts(INT_TIMER1);    // aguarda o dispositivo colocar o dado na saída
        df = (input(pin_b7)); // retorna o dado
    }
    if(sens==9) { output_low (pin_a0);
        output_float (pin_a0);
        enable_interrupts(INT_EXT);
        enable_interrupts(INT_TIMER1);
        delay_us (15);
        disable_interrupts(INT_EXT);
        disable_interrupts(INT_TIMER1);    // aguarda o dispositivo colocar o dado na saída
        df = (input(pin_a0)); // retorna o dado
    }
    if(sens==10) { output_low (pin_a1);
        output_float (pin_a1);
        enable_interrupts(INT_EXT);
        enable_interrupts(INT_TIMER1);
        delay_us (15);
        disable_interrupts(INT_EXT);
        disable_interrupts(INT_TIMER1);    // aguarda o dispositivo colocar o dado na saída
        df = (input(pin_a1)); // retorna o dado
    }
    if(sens==11) { output_low (pin_a2);
        output_float (pin_a2);
        delay_us (15);    // aguarda o dispositivo colocar o dado na saída
        df = (input(pin_a2)); // retorna o dado
    }
    if(sens==12) { output_low (pin_a3);
        output_float (pin_a3);
        delay_us (15);    // aguarda o dispositivo colocar o dado na saída
        df = (input(pin_a3)); // retorna o dado
    }
    if(sens==13) { output_low (pin_a4);
        output_float (pin_a4);
        delay_us (15);    // aguarda o dispositivo colocar o dado na saída
        df = (input(pin_a4)); // retorna o dado
    }
    if(sens==14) { output_low (pin_a5);
        output_float (pin_a5);
        delay_us (15);    // aguarda o dispositivo colocar o dado na saída
        df = (input(pin_a5)); // retorna o dado
    }
    if(sens==15) { output_low (pin_e0);
        output_float (pin_e0);
        delay_us (15);    // aguarda o dispositivo colocar o dado na saída
        df = (input(pin_e0)); // retorna o dado
    }
    if(sens==16) { output_low (pin_e1);
        output_float (pin_e1);
        delay_us (15);    // aguarda o dispositivo colocar o dado na saída
        df = (input(pin_e1)); // retorna o dado
    }
    return (df);
}
void escreve_bit_1w (int8 sens, boolean bit)
// escreve um bit no barramento 1-wire
{
    if (sens==1) { output_low (pin_d0);
        if (bit) output_float (pin_d0);
        enable_interrupts(INT_EXT);
        enable_interrupts(INT_TIMER1);
        delay_us (120);
        disable_interrupts(INT_EXT);
        disable_interrupts(INT_TIMER1);
        output_float (pin_d0);
    }
    if (sens==2) { output_low (pin_b1);
        if (bit) output_float (pin_b1);
    }
}

```

```

        enable_interrupts(INT_EXT);
        enable_interrupts(INT_TIMER1);
        delay_us (120);
        disable_interrupts(INT_EXT);
        disable_interrupts(INT_TIMER1);
        output_float (pin_b1);
    }
    if (sens==3) { output_low (pin_b2);
        if (bit) output_float (pin_b2);
        enable_interrupts(INT_EXT);
        enable_interrupts(INT_TIMER1);
        delay_us (120);
        disable_interrupts(INT_EXT);
        disable_interrupts(INT_TIMER1);
        output_float (pin_b2);
    }
    if (sens==4) { output_low (pin_b3);
        if (bit) output_float (pin_b3);
        enable_interrupts(INT_EXT);
        enable_interrupts(INT_TIMER1);
        delay_us (120);
        disable_interrupts(INT_EXT);
        disable_interrupts(INT_TIMER1);
        output_float (pin_b3);
    }
    if (sens==5) { output_low (pin_b4);
        if (bit) output_float (pin_b4);
        enable_interrupts(INT_EXT);
        enable_interrupts(INT_TIMER1);
        delay_us (120);
        disable_interrupts(INT_EXT);
        disable_interrupts(INT_TIMER1);
        output_float (pin_b4);
    }
    if (sens==6) { output_low (pin_b5);
        if (bit) output_float (pin_b5);
        enable_interrupts(INT_EXT);
        enable_interrupts(INT_TIMER1);
        delay_us (120);
        disable_interrupts(INT_EXT);
        disable_interrupts(INT_TIMER1);
        output_float (pin_b5);
    }
    if (sens==7) { output_low (pin_b6);
        if (bit) output_float (pin_b6);
        enable_interrupts(INT_EXT);
        enable_interrupts(INT_TIMER1);
        delay_us (120);
        disable_interrupts(INT_EXT);
        disable_interrupts(INT_TIMER1);
        output_float (pin_b6);
    }
    if (sens==8) { output_low (pin_b7);
        if (bit) output_float (pin_b7);
        enable_interrupts(INT_EXT);
        enable_interrupts(INT_TIMER1);
        delay_us (120);
        disable_interrupts(INT_EXT);
        disable_interrupts(INT_TIMER1);
        output_float (pin_b7);
    }
    if (sens==9) { output_low (pin_a0);
        if (bit) output_float (pin_a0);
        enable_interrupts(INT_EXT);
        enable_interrupts(INT_TIMER1);
        delay_us (120);
        disable_interrupts(INT_EXT);
        disable_interrupts(INT_TIMER1);
        output_float (pin_a0);
    }
    if (sens==10) { output_low (pin_a1);
        if (bit) output_float (pin_a1);
        enable_interrupts(INT_EXT);
        enable_interrupts(INT_TIMER1);
        delay_us (120);
        disable_interrupts(INT_EXT);
    }

```

```

        disable_interrupts(INT_TIMER1);
        output_float (pin_a1);
    }
    if (sens==11) { output_low (pin_a2);
        if (bit) output_float (pin_a2);
        delay_us (120);
        output_float (pin_a2);
    }
    if (sens==12) { output_low (pin_a3);
        if (bit) output_float (pin_a3);
        delay_us (120);
        output_float (pin_a3);
    }
    if (sens==13) { output_low (pin_a4);
        if (bit) output_float (pin_a4);
        delay_us (120);
        output_float (pin_a4);
    }
    if (sens==14) { output_low (pin_a5);
        if (bit) output_float (pin_a5);
        delay_us (120);
        output_float (pin_a5);
    }
    if (sens==15) { output_low (pin_e0);
        if (bit) output_float (pin_e0);
        delay_us (120);
        output_float (pin_e0);
    }
    if (sens==16) { output_low (pin_e1);
        if (bit) output_float (pin_e1);
        delay_us (120);
        output_float (pin_e1);
    }
}

byte le_byte_1w (int8 sen)
// lê um byte do barramento 1-wire
{
    byte i, dado = 0;
    boolean df;
    // lê oito bits iniciando pelo bit menos significativo
    for (i=0;i<8;i++)
    {
        // sens=1;
        if (sen==1) { output_low (pin_d0);
            output_float (pin_d0);
            enable_interrupts(INT_EXT);
            enable_interrupts(INT_TIMER1);
            delay_us (15); // aguarda o dispositivo colocar
            disable_interrupts(INT_EXT);
            disable_interrupts(INT_TIMER1);
            df = (input(pin_d0)); // retorna o dado
        }
        if (sen==2) { output_low (pin_b1);
            output_float (pin_b1);
            enable_interrupts(INT_EXT);
            enable_interrupts(INT_TIMER1);
            delay_us (15); // aguarda o dispositivo colocar
            disable_interrupts(INT_EXT);
            disable_interrupts(INT_TIMER1);
            df = (input(pin_b1)); // retorna o dado
        }
        if (sen==3) { output_low (pin_b2);
            output_float (pin_b2);
            enable_interrupts(INT_EXT);
            enable_interrupts(INT_TIMER1);
            delay_us (15); // aguarda o dispositivo colocar
            disable_interrupts(INT_EXT);
            disable_interrupts(INT_TIMER1);
            df = (input(pin_b2)); // retorna o dado
        }
        if (sen==4) { output_low (pin_b3);
            output_float (pin_b3);
            enable_interrupts(INT_EXT);
            enable_interrupts(INT_TIMER1);
            delay_us (15); // aguarda o dispositivo colocar
            disable_interrupts(INT_EXT);

```

```

        disable_interrupts(INT_TIMER1);
        df = (input(pin_b3)); // retorna o dado
    }
    if (sen==5) { output_low (pin_b4);
        output_float (pin_b4);
        enable_interrupts(INT_EXT);
        enable_interrupts(INT_TIMER1);
        delay_us (15); // aguarda o dispositivo colocar
        disable_interrupts(INT_EXT);
        disable_interrupts(INT_TIMER1);
        df = (input(pin_b4)); // retorna o dado
    }
    if (sen==6) { output_low (pin_b5);
        output_float (pin_b5);
        enable_interrupts(INT_EXT);
        enable_interrupts(INT_TIMER1);
        delay_us (15); // aguarda o dispositivo colocar
        disable_interrupts(INT_EXT);
        disable_interrupts(INT_TIMER1);
        df = (input(pin_b5)); // retorna o dado
    }
    if (sen==7) { output_low (pin_b6);
        output_float (pin_b6);
        enable_interrupts(INT_EXT);
        enable_interrupts(INT_TIMER1);
        delay_us (15); // aguarda o dispositivo colocar
        disable_interrupts(INT_EXT);
        disable_interrupts(INT_TIMER1);
        df = (input(pin_b6)); // retorna o dado
    }
    if (sen==8) { output_low (pin_b7);
        output_float (pin_b7);
        enable_interrupts(INT_EXT);
        enable_interrupts(INT_TIMER1);
        delay_us (15); // aguarda o dispositivo colocar
        disable_interrupts(INT_EXT);
        disable_interrupts(INT_TIMER1); // o dado na saída
        df = (input(pin_b7)); // retorna o dado
    }
    if (sen==9) { output_low (pin_a0);
        output_float (pin_a0);
        enable_interrupts(INT_EXT);
        enable_interrupts(INT_TIMER1);
        delay_us (15); // aguarda o dispositivo colocar
        disable_interrupts(INT_EXT);
        disable_interrupts(INT_TIMER1); // o dado na saída
        df = (input(pin_a0)); // retorna o dado
    }
    if (sen==10) { output_low (pin_a1);
        output_float (pin_a1);
        enable_interrupts(INT_EXT);
        enable_interrupts(INT_TIMER1);
        delay_us (15); // aguarda o dispositivo colocar
        disable_interrupts(INT_EXT);
        disable_interrupts(INT_TIMER1);
        // o dado na saída
        df = (input(pin_a1)); // retorna o dado
    }
    /* if (sen==11) { output_low (pin_a2);
        output_float (pin_a2);
        delay_us (15); // aguarda o dispositivo colocar // o dado na saída
        df = (input(pin_a2)); // retorna o dado
    }
    if (sen==12) { output_low (pin_a3);
        output_float (pin_a3);
        delay_us (15); // aguarda o dispositivo colocar // o dado na saída
        df = (input(pin_a3)); // retorna o dado
    }
    if (sen==13) { output_low (pin_a4);
        output_float (pin_a4);
        delay_us (15); // aguarda o dispositivo colocar // o dado na saída
        df = (input(pin_a4)); // retorna o dado
    }
    if (sen==14) { output_low (pin_a5);
        output_float (pin_a5);
        delay_us (15); // aguarda o dispositivo colocar // o dado na saída

```

```

        df = (input(pin_a5)); // retorna o dado
    }
    if (sen==15) { output_low (pin_e0);
        output_float (pin_e0);
        delay_us (15); // aguarda o dispositivo colocar // o dado na saída
        df = (input(pin_e0)); // retorna o dado
    }
    if (sen==16) { output_low (pin_e1);
        output_float (pin_e1);
        delay_us (15); // aguarda o dispositivo colocar // o dado na saída
        df = (input(pin_e1)); // retorna o dado
    }*/
    if (df==1) dado|=0x01<<i;
        delay_us(90); // aguarda o fim do quadro de leitura
        // do bit atual
    }
    return (dado);
}

void escreve_byte_1w (int8 se, char dado)
// escreve um byte no barramento 1-wire
{
    byte i, temp;
    // envia o byte iniciando do bit menos significativo
    for (i=0; i<8; i++)
    {
        temp = dado>>i; // desloca o dado 1 bit à direita
        temp &= 0x01; // isola o bit 0 (LSB)
        if (se==1) escreve_bit_1w (1,temp); // escreve o bit no barramento
        if (se==2) escreve_bit_1w (2,temp); // escreve o bit no barramento
        if (se==3) escreve_bit_1w (3,temp); // escreve o bit no barramento
        if (se==4) escreve_bit_1w (4,temp); // escreve o bit no barramento
        if (se==5) escreve_bit_1w (5,temp); // escreve o bit no barramento
        if (se==6) escreve_bit_1w (6,temp); // escreve o bit no barramento
        if (se==7) escreve_bit_1w (7,temp); // escreve o bit no barramento
        if (se==8) escreve_bit_1w (8,temp); // escreve o bit no barramento
        if (se==9) escreve_bit_1w (9,temp); // escreve o bit no barramento
        if (se==10) escreve_bit_1w (10,temp); // escreve o bit no barramento
        // if (se==11) escreve_bit_1w (11,temp); // escreve o bit no barramento
        // if (se==12) escreve_bit_1w (12,temp); // escreve o bit no barramento
        /* if (se==13) escreve_bit_1w (13,temp); // escreve o bit no barramento
        if (se==14) escreve_bit_1w (14,temp); // escreve o bit no barramento
        if (se==15) escreve_bit_1w (15,temp); // escreve o bit no barramento
        if (se==16) escreve_bit_1w (16,temp); // escreve o bit no barramento*/
    }
}

float le_18B20 (int8 ss)
{
    float Temp;
    int8 Conta,
        Escala,
        Buffer[9] ;
    disable_interrupts(INT_EXT);
    disable_interrupts(INT_TIMER1);
    reset_1w (ss);
    escreve_byte_1w (ss,0xcc); // comando skip ROM
    escreve_byte_1w (ss,0x44); // inicia conversão da temperatura
    enable_interrupts(INT_EXT);
    enable_interrupts(INT_TIMER1);
    delay_ms(700); // rotina delay_ms(1000);
    disable_interrupts(INT_EXT);
    disable_interrupts(INT_TIMER1);

    reset_1w (ss); // reseta o dispositivo
    escreve_byte_1w (ss,0xcc); // comando skip ROM
    escreve_byte_1w (ss,0xbe); // comando de leitura da memória de rascunho , efetua a leitura dos nove bytes da memória de
    rascunho
    for (conta = 0; conta<9; conta++) buffer[conta]=le_byte_1w (ss);
    Temp = ((buffer[0] & 240)>>4 | buffer[1]<<4) + (buffer[0] & 15 )* 0.0625;
    reset_1w(ss);
    escreve_byte_1w (ss,0xcc);
    escreve_byte_1w (ss,0xb4);
    escala = (buffer[1] & 240)>>4;
    enable_interrupts(INT_EXT);
    enable_interrupts(INT_TIMER1);
    return (Temp) ;
}

```

```

void main()
{
    float Iinput[16];
    setup_adc_ports(NO_ANALOGS);
    setup_adc(ADC_OFF);
    setup_psp(PSP_DISABLED);
    setup_spi(FALSE);
    setup_timer_0(RTCC_INTERNAL|RTCC_DIV_2);
    //setup_timer_1(T1_DISABLED);
    //setup_timer_2(T2_DIV_BY_1,255,4);
    setup_timer_1(T1_INTERNAL|T1_DIV_BY_1);
    //setup_ccp1(CCP_PWM);
    setup_comparator(NC_NC_NC_NC);
    setup_vref(FALSE);
    enable_interrupts(INT_EXT);
    enable_interrupts(INT_TIMER1);
    enable_interrupts(INT_RDA);
    enable_interrupts(GLOBAL);
    //set_pwm1_duty(0);
    set_timer1(0);
    //output_low(pin_c2);
    // TODO: USER CODE!!
    while (true)
    {
        enable_interrupts(INT_EXT);
        enable_interrupts(INT_TIMER1);
        i2=0;
        for (i=1; i< 11; ++i) { IInput[i] = le_18B20(i);}
        sensor1=(IInput[1])*40;
        enable_interrupts(INT_EXT);
        enable_interrupts(INT_TIMER1);
        i2=0;
        printf("< ");
        for (i=1; i<11; ++i) { printf(" S%2ld@%03.2f ", i,IInput[i]); } // escreve na serial
        printf(" >\r\n");
        //printf("Teste teste >\r\n");
        //printf(" %ld %f %ld %ld %ld %ld %ld>\r\n",out,sensor1,proporcional,integral,derivativo,erro);
        enable_interrupts(INT_EXT);
        enable_interrupts(INT_TIMER1);
        if(indicador==1)
            controle();
        else
        {
            out=man*60;
            if(out>6000)
                out=6000;
            if(out<0)
                out=0;
            out=out*0.8;
        }
        i2=0;
        printf(" %lf %lf %lf %lf %d z=%ld w=%d>\r\n",k,Ti,td,setp,pb,man,indicador);
    }
}

```

8.2 – Programas Identificação (*Software* MATLAB™)

```
% Filtragem dos sinais

n=length(a);

minval = nan(1,n-window+1);
maxval = nan(1,n-window+1);

L = initwedge(window+1);
U = initwedge(window+1);

for i = 2:n
    if i > window
        if ~wedgeisempty(U)
            maxval(i-window) = a(getfirst(U));
        else
            maxval(i-window) = a(i-1);
        end
        if ~wedgeisempty(L)
            minval(i-window) = a(getfirst(L));
        else
            minval(i-window) = a(i-1);
        end
    end % i>window

    if a(i) > a(i-1)
        L = pushback(L, i-1);
        if i==window+getfirst(L), L = popfront(L);
    end
    while ~wedgeisempty(U)
        if a(i)<=a(getlast(U))
            if i==window+getfirst(U), U = popfront(U);
            end
            break
        end
        U = popback(U);
    end % while loop
    else
        U = pushback(U, i-1);
        if i==window+getfirst(U), U = popfront(U);
    end
    while ~wedgeisempty(L)
        if a(i)>=a(getlast(L))
            if i==window+getfirst(L), L = popfront(L);
            end
            break
        end
        L = popback(L);
    end % while loop
    end % if
end % for-loop

i=n+1;
if ~wedgeisempty(U)
    maxval(i-window) = a(getfirst(U));
else
    maxval(i-window) = a(i-1);
end
if ~wedgeisempty(L)
    minval(i-window) = a(getfirst(L));
else
    minval(i-window) = a(i-1);
end

end
function X = initwedge(sz)
X = struct('buffer', zeros(1, sz), ...
    'sz', sz, ...
    'n', 0, ...
    'first', 1, ...
    'last', 0, ...
    'mxn', 0);
end
```

```
function b = wedgeisempty(X)
b = X.n <= 0;
end
```

```
function X = pushback(X, v)
X.last = mod(X.last,X.sz) + 1;
X.buffer(X.last) = v;
X.n = X.n+1;
X.mxn = max(X.mxn,X.n);
end
```

```
function X = popback(X)
X.n = X.n-1;
X.last = mod(X.last-2,X.sz) + 1;
end
```

```
function X = popfront(X)
X.n = X.n-1;
X.first = mod(X.first,X.sz) + 1;
end
```

```
function v = getfirst(X)
v = X.buffer(X.first);
end
```

```
function v = getlast(X)
v = X.buffer(X.last);
end
```

```
function dispw(X)
A = X.buffer(mod(X.first+(-1:X.n-2), X.sz) + 1);
disp(A)
end
```

%Modelagem do atraso do tempo

```
function response=delayt(k,nstep)

[nvar,j,nlag]=size(k);
response=zeros(nvar,nvar,nstep);
A0=k(:,j,1);
if cond(A0)>1e200
    disp 'singular A0'
    response=[];
else
    A0i=inv(A0);
    Aplus=k(:,j,2:nlag);
    Aplus=A0\Aplus(:,j);
    Aplus=reshape(Aplus,[nvar,nvar,nlag-1]);
    response(:,j,1)=A0i;
    for it=1:nstep, for ilag=1:min(nlag-1,it-1)
        response(:,j,it)=response(:,j,it)+Aplus(:,j,ilag)*response(:,j,it-ilag);
    end, end
end
```

```
function y = model_func_lin(x, p, k)
```

```
if k==0
y = p(1)+p(2)*x;
elseif k==1
y = 1;
elseif k==2
y = x;
end
```

%Redes Neurais

```
ts=0.5;
epoca = 20000;
error= 0;
lr = 0.01;
gradiente = 1e-10;
ID=[1];
S1=[10 1];
```

```

net = newfftd(a,T,ID,S1);

net.inputWeights{1,1}.delays = [1 2 3 4];
net.trainParam.epochs = epoca;
net.trainParam.goal = error;
net.trainParam.min_grad = gradiente;
net.trainParam.alpha = lr;
net = train(net,P,T);
Y = sim(net,P)
gensim(net,ts)
plot(z);
hold on
plot(zp,'r-.');
title('temperaturas');
legend('modelo arx, dados entrada saida');
xlabel('tempo (seg)');
nk = delayt(z);

arx620 = arx(a, [6 2 nk]);          % modelo ARX.
plot(a);
hold on
plot(a,'r-.');
title('temperaturas');
legend('modelo arx, dados entrada saida');
xlabel('tempo (seg)');

armax52 = armax(a, [5 2 nk]);       %Modelo BJ
plot(a);
hold on
plot(a,'r-.');
title('temperaturas');
legend('modelo BJ, dados entrada saida');
xlabel('tempo (seg)');

oe73 = oe(a, [7 3 nk]);            %Modelo OE
plot(a);
hold on
plot(a,'r-.');
title('temperaturas');
legend('modelo OE, dados entrada saida');
xlabel('tempo (seg)');

bj62 = bj(a, [6 2 nk]);            %Modelo BJ
plot(a);
hold on
plot(a,'r-.');
title('temperaturas');
legend('modelo BJ, dados entrada saida');
xlabel('tempo (seg)');

th=n4s2;                           % Modelo Espaço de Estados

[num, den] = tfdata(th) ;
trfn = tf(num, den, 0.01)
con = d2c(trfn)
plot(a);
hold on
plot(zp,'r-.');
title('temperaturas');
legend('modelo Espaço de Estados, dados entrada saida');
xlabel('tempo (seg)');

```