Universidade do Estado do Rio de Janeiro
Departamento de Engenharia Eletrônica e Telecomunicações
Faculdade de Engenharia

David Ricardo de Mendonça Soares

Sistema inteligente com entrada e saída remota sem fio

Rio de Janeiro
2010

David Ricardo de Mendonça Soares

Sistema inteligente com entrada e saída remota sem fio

Dissertação apresentada, como requisito parcial para a obtenção do título de Mestre, ao Programa de Pós-Graduação em Engenharia Eletrônica, da Universidade do Estado do Rio de Janeiro. Área de concentração: Sistemas Inteligentes e Automação.

Orientador: Prof. Dr. José Franco Machado do Amaral

Coorientador: Prof. Dr. Jorge Luís Machado do Amaral

Rio de Janeiro

2010

CATALOGAÇÃO NA FONTE
UERJ / REDE SIRIUS / BIBLIOTECA CTC/B

S676 Soares, David Ricardo de.
Sistema inteligente com entrada e saída remota sem fio / David Ricardo de Mendonça Soares. – 2010.
116 f. Il.

Orientador: José Franco Machado do Amaral.
Co-orientador: Jorge Luís Machado do Amaral.
Dissertação (Mestrado) – Universidade do Estado do Rio de Janeiro, Faculdade de Engenharia.

1. Sistemas inteligentes de controle – Teses. 2. Redes de sensores sem fio – Teses. 3. Engenharia Eletrônica e Telecomunicações – Teses. I. Amaral, José Franco Machado do. II. Universidade do Estado do Rio. III. Título.

CDU 681.51

Assinatura

01.09.2010

Data

David Ricardo de Mendonça Soares

Sistema inteligente com entrada e saída remota sem fio

Dissertação apresentada, como requisito parcial para a obtenção do título de Mestre, ao Programa de Pós-Graduação em Engenharia Eletrônica, da Universidade do Estado do Rio de Janeiro. Área de concentração: Sistemas Inteligentes e Automação.

Aprovado em 15 de julho de 2010

Banca Examinadora:

Prof. Dr. José Franco Machado do Amaral (Orientador)
Faculdade de Engenharia, UERJ

Prof. Dr. Jorge Luís Machado do Amaral (Co-orientador)
Faculdade de Engenharia, UERJ

Prof. Dr. Luiz Biondi Neto
Faculdade de Engenharia, UERJ

Profª. Drª. Karla Tereza Figueiredo Leite
Departamento de Engenharia Elétrica, PUC-Rio

Rio de Janeiro
2010

DEDICATÓRIA

A minha esposa Eliane, que me acompanhou e me deu força a cada passo do Curso de Mestrado e às nossas crianças, Igor e Tayane, que me alegram e me acompanharam nesta etapa, tão importante, de minha vida.

AGRADECIMENTOS

Aos professores José Franco Machado do Amaral e Jorge Luís Machado do Amaral, os quais, acreditando no meu potencial, abriram as portas da Universidade para esta oportunidade.

À Infraero – Empresa Brasileira de Infra-Estrutura Aeroportuária que me liberou no horário das aulas para que eu pudesse concluir este curso.

Ao Ricardo Guedes Machado, Ex-Gerente de Manutenção do Aeroporto Santos Dumont, à Patrícia Ramos Cardoso Guimarães, Coordenadora de Manutenção do Aeroporto Santos Dumont e à Patrícia Angélica Silva, Coordenadora de Planejamento e Gestão da Manutenção do Aeroporto Santos Dumont, que, graças a competência desta equipe, o trabalho pode ser implementado.

Não preciso de modelos.
Não preciso de heróis.
Eu tenho meus amigos.
Renato Russo

RESUMO

SOARES, David Ricardo de. Sistema inteligente com entrada e saída remota sem fio. 2010. 116f. Dissertação (Mestrado em Engenharia Eletrônica) – Faculdade de Engenharia, Universidade do Estado do Rio de Janeiro, Rio de Janeiro, 2009.

Este trabalho avalia o desempenho de um controlador fuzzy (tipo Takagi-Sugeno-Kang) quando, utilizando tecnologia sem fio para conectar as entradas e a saída do controlador aos sensores/atuadores, sofre perda das informações destes canais, resultado de perdas de pacotes. Tipicamente são utilizados controladores PID nas malhas de controle. Assim, o estudo realizado compara os resultados obtidos com os controladores fuzzy com os resultados dos controladores PID. Além disso, o trabalho visa estudar o comportamento deste controlador implementado em uma arquitetura microprocessada utilizando números inteiros nos cálculos, interpolação com segmentos de reta para as funções de pertinência da entrada e *singletons* nas funções de pertinência da saída. Para esse estudo foi utilizado, num ambiente Matlab®/Simulink®, um controlador fuzzy e o aplicativo *True Time* para simular o ambiente sem fio. Desenvolvido pelo Departamento de Controle Automático da Universidade de Lund, o *True Time* é baseado no Matlab®/Simulink® e fornece todas as ferramentas necessárias para a criação de um ambiente de rede (com e sem fio) virtual. Dado o paradigma de que quanto maior for a utilização do canal, maior a degradação do mesmo, é avaliado o comportamento do sistema de controle e uma proposta para diminuir o impacto da perda de pacotes no controle do sistema, bem como o impacto da variação das características internas da planta e da arquitetura utilizada na rede. Inicialmente são realizados ensaios utilizando-se o controlador fuzzy virtual (Simulink®) e, posteriormente, o controlador implementado com dsPIC. Ao final, é apresentado um resumo desses ensaios e a comprovação dos bons resultados obtidos com um controlador fuzzy numa malha de controle utilizando uma rede na entrada e na saída do controlador

Palavras-chave: Fuzzy. Rede de Sensores sem Fio. *True Time*.

ABSTRACT

This work evaluates the performance of a fuzzy controller (Takagi-Sugeno-Kang) that uses wireless technology to connect the inputs and the output of the controller to sensors / actuators, and with the loss of information from these channels, the result of packet loss. PID controllers are typically used in control loops. Thus, the study compares the results obtained with the fuzzy controllers with the results of PID controllers. Moreover, the work aims to study the behavior of this controller implemented in a microprocessor architecture using integer calculations, interpolation with straight line segments for the membership functions of input and singletons in the output membership functions. For this study it was used in a Matlab®/Simulink®, a fuzzy controller and the application True Time to simulate wireless environment (Developed by the Department of Automatic Control at Lund University). It is based on MATLAB®/Simulink® and provides all the tools necessary to create a virtual network environment (wired and wireless). When we increase the occupation of the channel we increase the degradation of it. Under this conditions, is rated the behavior of the control system and is evaluated, and actions were proposal to reduce the impact of packet loss in the control system, as well as the impact of variations in the internal characteristics of plant and architecture used in the network. Initially, tests are conducted using the virtual fuzzy controller (Simulink ®) and thereafter, the controller implemented with dsPIC. Finally, a summary of testing and verification of results are presented.

Keywords: Fuzzy. Wireless Sensor Network. True Time.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

ASIC	*Application-Specific Integrated Circuit*
CAN	*Controller Area Network*
CISC	*Complex Instruction Set Computer*
CSMA/CD	*Carrier Sense Multiple Access with Collision Detection*
CSMA/AMP	*Carrier Sense Multiple Access / Arbitration by Message Priority*
DUT	Device Under Test
FDMA	*Frequency Division Multiple Access*
FPAA	*Field-Programmable Analog Array*
FPGA	*Field-Programmable Gate Array*
IHM	Interface Homem-Máquina
NCS	*Networked Control Systems*
PIL	*Processor in loop co-simulation*
RISC	*Reduced Instruction Set Computer*
TDMA	*Time Division Multiple Access*
USB	*Universal Serial Bus*
WLAN	*Wireless Local Area Network*

SUMÁRIO

INTRODUÇÃO

A necessidade de controlar um processo é bastante antiga. Ela surgiu no momento em que o homem passou a manufaturar bens para suas necessidades. No processo de manufatura, o homem era o responsável pelo controle e pela execução de todos os procedimentos envolvidos no processo. Com a invenção do regulador mecânico de velocidade para a máquina a vapor, concebido por James Watt, a máquina passou a ter um uso industrial importante, pois agora a velocidade permanecia constante e era regulada automaticamente por um dispositivo, permitindo que uma máquina pudesse realizar uma tarefa ou um processo. Com o passar do tempo, novos componentes, como sensores, atuadores e controladores, foram desenvolvidos, passando por várias tecnologias (mecânica, elétrica, eletro-mecânica, eletrônica e inteligente) na medida em que os processos fabris foram se tornando mais complexos.

O desenvolvimento dos Controladores Lógicos Programáveis e da instrumentação eletrônica viabilizou um alto grau de sofisticação no controle de processos industriais. A partir dos anos 80, a eletrônica acelerou o processo de integração, permitindo que um único equipamento fosse capaz de controlar mais de um processo. A partir daí uma central de controle passou a fazer parte das linhas de produção e uma enorme quantidade de cabos começou a circular pelas fábricas. Esta situação permaneceu assim até o surgimento dos sensores inteligentes. Estes sensores, que inicialmente foram desenvolvidos para facilitar os procedimentos de manutenção, acabaram incorporando outras funções, dentre elas a capacidade de se comunicar. Esta característica alavancou a criação das redes de sensores, cuja grande vantagem é a redução da quantidade de cabos para interligar os pontos de medição, a possibilidade de atuação nos diversos processos de uma fábrica e melhor qualidade dos sinais (melhor relação sinal/ruído).

Em paralelo, várias tecnologias para a comunicação sem fio foram desenvolvidas para uso exclusivamente militar. Quando a indústria teve acesso a estas tecnologias, surgiram as primeiras redes de sensores sem fio que, apesar de apresentar uma grande facilidade de instalação, apresentavam características que limitavam suas aplicações pois o canal de transmissão não era confiável. Apesar de ser um problema de difícil solução, várias técnicas para contornar esta limitação

foram apresentadas objetivando atender as necessidades de uma comunicação satisfatória.

Em meados da década de 60, Zadeh observou que os recursos tecnológicos disponíveis eram incapazes de automatizar as atividades humanas relacionadas a problemas de natureza industrial, biológica ou química, que compreendessem situações ambíguas, ou que, segundo suas próprias palavras, apresentassem "sentimentos matemáticos humanísticos" (Weber et al., 2003). Procurando solucionar este problema, o Prof. Zadeh publicou, em 1965, um artigo apresentando os conceitos dos conjuntos fuzzy. Em 1973, Zadeh apresentou o *Princípio da Incompatibilidade*:

"*À medida que a complexidade de um sistema aumenta nossa habilidade para fazer afirmações precisas e que sejam significativas acerca deste sistema diminui até que um limiar é atingido, além do qual precisão e relevância tornam-se quase que características mutuamente exclusivas*" (Weber et al., 2003).

Esta lógica tem se mostrado muito atraente aos projetistas de sistemas de controle. Isto se deve, basicamente, ao fato desta lógica apresentar robustez e facilidade de programação. Sem a necessidade de uma modelagem matemática complexa, devido ao uso de uma linguagem quase coloquial, a lógica fuzzy proporciona uma ótima ferramenta de desenvolvimento na implantação de sistemas de controle, em especial em aplicações não-lineares (Júnior et al., 2000).

Como os controladores fuzzy são mais flexíveis, no sentido de que se pode codificar regras para tratar os problemas causados pelas falhas nas entradas e saídas, eles merecem ser investigados como alternativa para ultrapassar as dificuldades impostas pela rede sem fio.

Objetivos

O objetivo principal deste trabalho é estudar o comportamento de um controlador fuzzy (tipo Takagi-Sugeno-Kang) quando utiliza tecnologia sem fio para conectar as entradas e saída do controlador aos sensores e ao atuador de um processo e propor soluções para contornar os problemas inerentes ao uso destes sensores remotos.

Além disso, o trabalho visa avaliar o desempenho deste controlador implementado fisicamente em uma arquitetura microprocessada utilizando números inteiros nos cálculos, interpolação com segmentos de reta para as funções de pertinência da entrada e *singletons* nas funções de pertinência da saída.

Esta dissertação contribui para a área de sistemas inteligentes e automação através da apresentação dos resultados obtidos a partir dos ensaios realizados com controladores fuzzy (teórico e real) e avaliando as propostas de solução para os problemas inerentes as redes sem fio, de forma a:

- Enfatizar a confiabilidade do controle fuzzy em uma rede sem fio;
- Determinar o tipo de aplicação que pode utilizar sensores sem fio;
- Identificar quais os parâmetros que devem ser utilizados para avaliar este tipo de controle;
- Identificar alternativas para contornar os problemas de uma rede sem fio no controle fuzzy.

Metodologia

Para se alcançar os objetivos propostos nesta dissertação, foram seguidas as seguintes etapas:

- Apresentação do(s):
  - Controle realimentado simples;
  - Controles: PID e fuzzy;
  - Formas de implementação de controladores fuzzy;
  - Redes industriais;
  - Problemas existentes (atrasos variáveis), perda de pacotes, etc.;
  - Redes sem fio;
- Identificação dos parâmetros que mais influenciam no controle;
- Sugestões de soluções;
- Implementação de uma plataforma real microprocessada;
- Descrição da plataforma - real e simulador (*True Time*);
- Descrição dos testes;
- Discussão dos resultados.

Desenvolvimento

Este trabalho está dividido, além desta introdução, em 08 capítulos.

O primeiro capítulo apresenta um breve histórico da automação, bem como o estudo da estabilidade de uma malha de controle analógica e digital.

O segundo capítulo apresenta um breve histórico dos comandos à distância (remotos) e como isto contribuiu para as redes de sensores. Em seguida, são apresentados alguns tipos de redes industriais e os desafios deste tipo de arquitetura. Por fim, apresenta-se uma rede industrial sem fio do tipo Zigbee.

O terceiro capítulo descreve os Sistemas de Controle em Rede e os aspectos que influenciam na estabilidade do controle. Posteriormente é estudado o impacto desses fatores na estabilidade da malha de controle, a identificação dos parâmetros que influenciam na instabilidade da malha, algumas soluções que contornam esse problema e o comportamento da malha com um sistema fuzzy. Ao final, é apresentada uma proposta para contornar este problema em malhas de controle fuzzy e uma avaliação da mesma.

O quarto capítulo descreve um sistema fuzzy e aborda algumas alternativas para implementação eletrônica.

O quinto capítulo aborda, minuciosamente, a implementação do controlador fuzzy utilizado neste trabalho, apresentando os detalhes do *hardware* e do *software*, assim como as técnicas de aproximações utilizadas no cálculo das funções de pertinência das entradas e no cálculo da inferência e da defuzzificação. Ao final, é apresentada uma avaliação do controlador implementado.

No sexto capítulo são apresentados os estudos de casos realizados.

No sétimo capítulo são apresentadas as conclusões e são sugeridos alguns trabalhos futuros.

1. CONTROLE DE PROCESSO

1.1 Um breve histórico da automação

O controle automático tem desempenhado um papel fundamental no avanço da engenharia e da ciência (Ogata, 2005). O primeiro trabalho significativo de controle automático foi o regulador centrifugo (Figura 1) construído por James Watt para o controle de velocidade de uma máquina a vapor, no século XVIII.

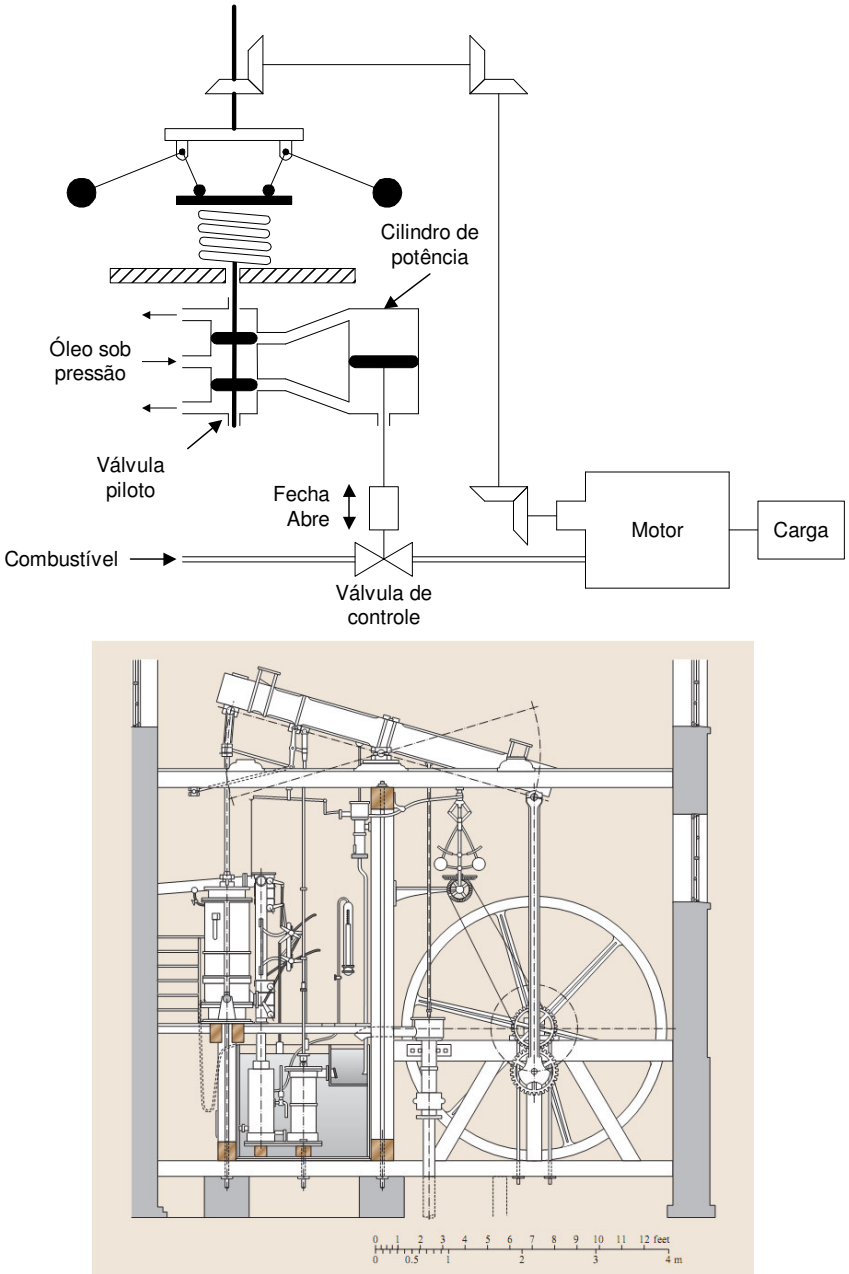


Figura 1 – Controle de velocidade para máquina a vapor (Ogata, 2005)

Neste engenhoso dispositivo, a quantidade de combustível fornecida ao motor é ajustada de acordo com a diferença entre a velocidade esperada e a velocidade efetiva do motor. Quando a velocidade do motor está estabilizada na velocidade correta, a válvula piloto encontra-se na posição central, bloqueando a passagem do óleo pressurizado para ambas as saídas. Se a velocidade real do motor diminuir, os pesos, pela redução da força centrífuga, se aproximam do eixo de rotação do dispositivo e a mola pré-comprimida move o eixo da válvula piloto para cima. Este movimento permite que o óleo pressurize a câmara posterior do cilindro de potência e abra a válvula de controle, injetando mais combustível no motor. À medida que a rotação do motor aumenta, a força centrífuga afasta os pesos do eixo de rotação, empurrando o eixo da válvula piloto para baixo fechando a passagem do óleo pressurizado, fazendo com que o cilindro de potência pare na posição que garante a velocidade de rotação correta.

A partir desta invenção, muitos pesquisadores forneceram valiosas contribuições ao estudo dos sistemas de controle. Minorsky, em 1922, demonstrou que a estabilidade de um sistema poderia ser determinada a partir de equações diferenciais de descrevem o sistema. Nyquist, em 1932, desenvolveu um procedimento para a determinação da estabilidade de sistemas de malha fechada com base na resposta de malha aberta a excitações senoidais estacionárias. Hazen, em 1934, introduziu o termo *servomecanismos* para sistemas de controle de posição e discutiu o projeto de um servomecanismo a relé, capaz de acompanhar de perto uma variação na entrada. Durante a década de 40, métodos de resposta em freqüência (diagramas de Bode) tornaram possível aos engenheiros projetar sistemas de controle linear de malha fechada que satisfizessem o desempenho desejado. Do final da década de 40 ao início da década de 50, Evans, desenvolve o método de lugar das raízes, essência da teoria clássica de controle. A partir de 1960, com a disponibilidade dos computadores digitais, foi possível a analise de sistemas complexos diretamente no domínio do tempo e lidar com a crescente complexidade dos sistemas modernos e seus rigorosos requisitos. No período de 1960 a 1980, o controle ótimo de sistemas determinísticos e estocásticos, bem como o controle adaptativo e de aprendizagem de sistemas complexos, foram amplamente pesquisados. De 1980 em diante, os desenvolvimentos na teoria de controle moderno se voltaram para o controle robusto, o controle H_∞ e tópicos associados.

Agora que os computadores digitais (Figura 2) se tornaram mais baratos e compactos, eles são utilizados como parte integrante dos sistemas de controle.

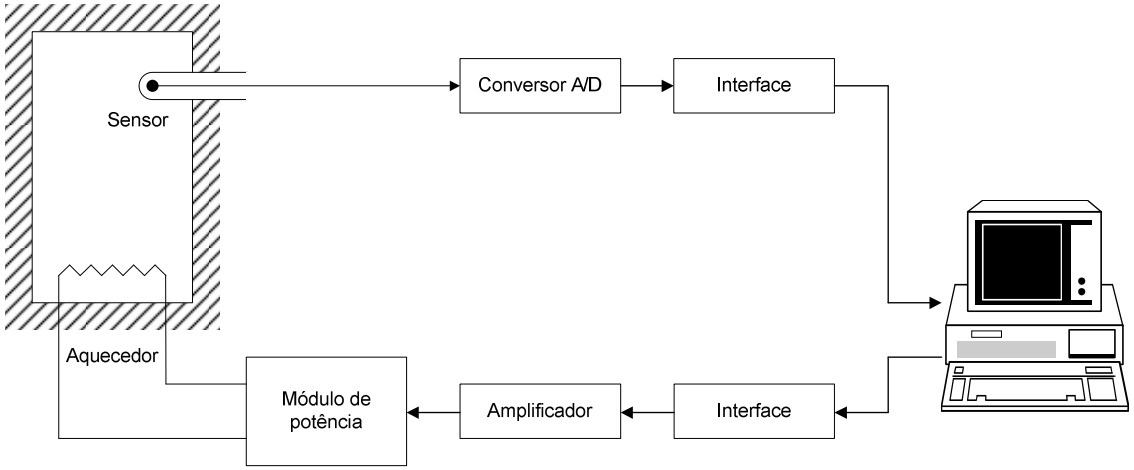


Figura 2 – Exemplo de sistema de controle moderno (Ogata, 2005)

1.2 Tipos de controle

Um sistema de controle consiste em subsistemas ou processos reunidos com o propósito de controlar as saídas dos processos (Nise, 2002). No estudo de sistemas de controle, deve-se ser capaz de modelar sistemas e analisar características dinâmicas (Ogata, 2005). O modelo matemático de um sistema dinâmico é definido como um conjunto de equações que representa a dinâmica do sistema. Um modelo matemático não é único e pode assumir diferentes formas dependendo do sistema considerado e das circunstâncias particulares. Na obtenção de um modelo matemático deve-se estabelecer uma conciliação entre a simplicidade do modelo e a precisão dos resultados na análise.

Como um sistema de controle possui vários componentes, para que se possa mostrar as funções que são executadas por cada um desses componentes, normalmente são utilizados diagrama de blocos (Figura 3).

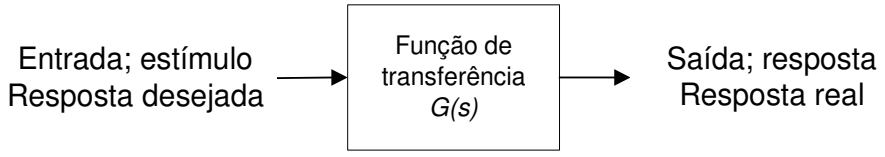


Figura 3 – Representação gráfica de uma função de transferência

A função de transferência de um sistema representado por uma equação diferencial linear invariante no tempo (Equação (1) é definida como a relação entre a transformada de Laplace da saída e a transformada de *Laplace* da entrada, admitindo-se as condições iniciais nulas [Ogata, 2005].

$$G(s) = \frac{\mathcal{L}[saída]}{\mathcal{L}[entrada]} = \frac{Y(s)}{X(s)} = \frac{b_0.s^m + b_1.s^{m-1} + \cdots + b_{m-1}.s + b_0}{a_0.s^n + a_1.s^{n-1} + \cdots + a_{n-1}.s + a_0} \quad (1)$$

Um diagrama de blocos de uma instalação industrial típica é constituído de um sensor (elemento de medida), um atuador e um controle automático (Figura 4). O controlador automático compara o valor real de saída da instalação com o valor de referência (valor desejado), determina o desvio e produz um sinal de controle que vai reduzir o desvio a zero ou a um valor pequeno. A maneira pela qual um controlador produz o sinal de controle é chamada ação de controle.

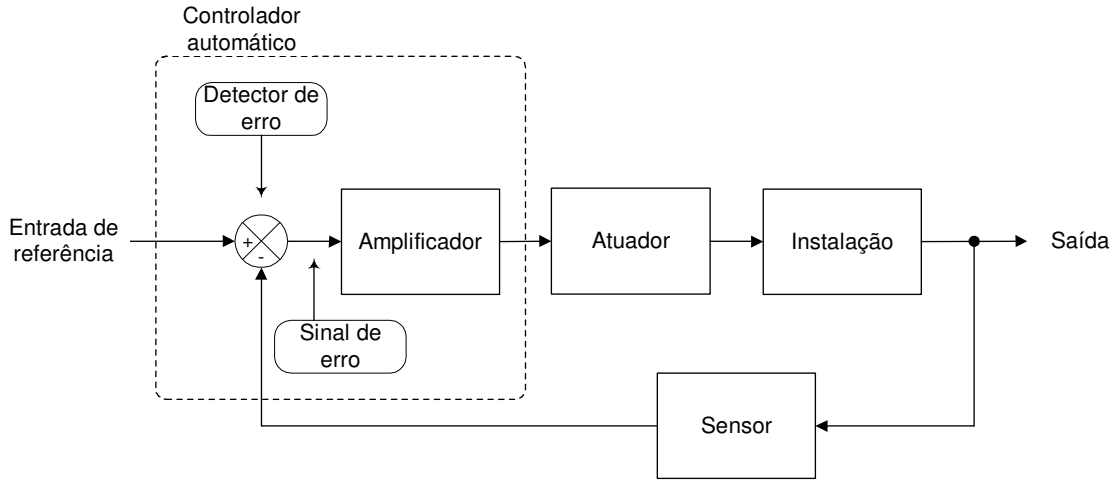


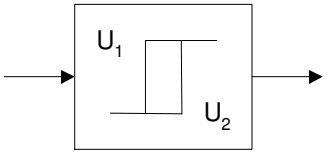
Figura 4 – Diagrama de blocos de uma instalação industrial típica

Os controladores industriais mais utilizados podem ser classificados de acordo com suas ações de controle:

- Controladores on-off;
- Controladores proporcionais (P);
- Controladores integrais (I);
- Controladores proporcional-integrais (PI);
- Controladores proporcional-derivativos (PD);
- Controladores proporcional-integral-derivativos (PID).

Controladores on-off

Em sistemas de controle do tipo *on-off*, o elemento de controle possui somente duas posições (*on* e *off*). Este tipo de controle é simples e barato e é bastante utilizado em sistemas de controle domésticos e industriais. Tendo como entrada o sinal e(t) e como saída os valores u(t), o sistema *on-off* possui a função de transferência apresentada na Figura 5. A curva de resposta de um sistema realimentado com um controlador proporcional é apresentada na Figura 6.



$$u(t) = \begin{cases} U1, & \text{para } e(t) > 0 \\ U2, & \text{para } e(t) < 0 \end{cases}$$

Figura 5 – Diagrama de blocos de um controlador *on-off*

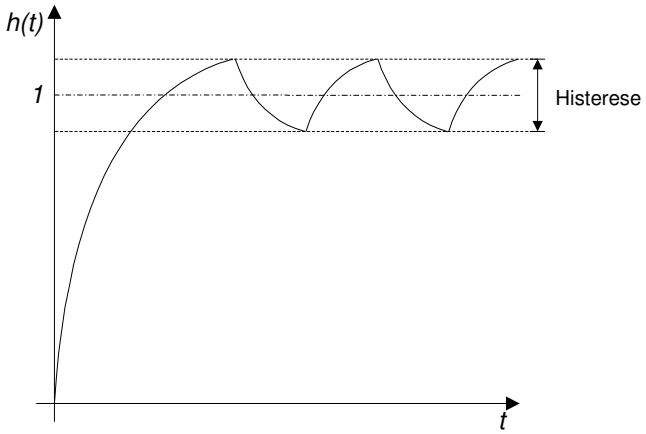


Figura 6 – Resposta do sistema realimentado com um controlador do tipo *on-off*

Uma das características deste tipo de controle é a existência de uma histerese. Esta histerese, em muitos casos é o resultado de um atrito não intencional e da perda de movimentos, entretanto, outras vezes é provocado intencionalmente, para prevenir a operação muito freqüente do mecanismo de *on-off*.

Controladores de ação proporcional (P)

Nos controladores de ação proporcional, a relação entre a entrada e a saída é dada pela Equação (2) e a curva de resposta ao degrau unitário é apresentada na Figura 7.

Domínio do tempo: $u(t) = K_p . e(t)$

Domínio da freqüência: $U(s) = K_p . E(s)$ (2)

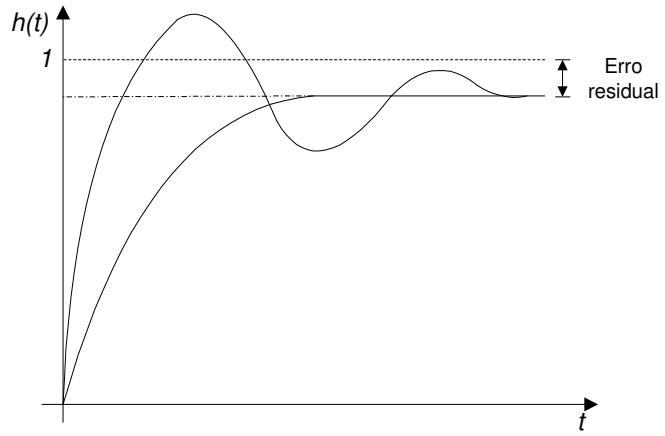


Figura 7 – Respostas típicas de um sistema realimentado com um controlador de ação do tipo P a um degrau unitário

O controlador proporcional possui como característica a existência de um erro residual. Este erro é conseqüência, em sistemas reais com perda, na necessidade de manutenção de uma atuação para manter o sistema estabilizado. Para um sistema de primeira ordem realimentado por um controlador proporcional o erro estacionário, e_{ss} (t $\rightarrow \infty$), é dado pela Equação (3) – (Ogata, 2001).

$$e_{ss} = \frac{1}{K_p + 1} \tag{3}$$

Controladores de ação integral (I)

Nos controladores de ação integral a relação entre a entrada e a saída do controlador é dada pela Equação (4) e a curva de resposta típica é apresentada na Figura 8.

Domínio do tempo: $u(t) = K_i \int_o^t e(t)dt$

Domínio da freqüência: $U(s) = \frac{K_i}{s} . E(s)$ (4)

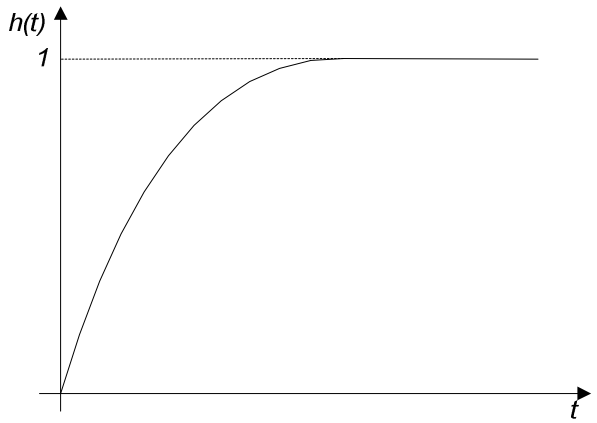


Figura 8 – Resposta do sistema realimentado com um controlador de ação do tipo I a um degrau unitário

Para este caso, um sistema primeira ordem realimentado por um controlador integral o erro estacionário (e_{ss}) é dado pela Equação (5).

$$e_{ss} = \lim_{s \to 0} \frac{s^2(Ts+1)}{Ts^2 + s + K} \frac{1}{s} = 0 \tag{5}$$

Controladores de ação proporcional-integral (PI)

Nos controladores de ação proporcional-integral a relação entre a entrada e a saída do controlador é dada pela Equação (6) e a curva de resposta típica é apresentada na Figura 9.

Domínio do tempo: $u(t) = K_p . e(t) + \frac{K_p}{T_i} \int_o^t e(t)dt$

Domínio da freqüência: $U(s) = K_p . \left(1 + \frac{1}{T_i . s}\right) . E(s)$ (6)

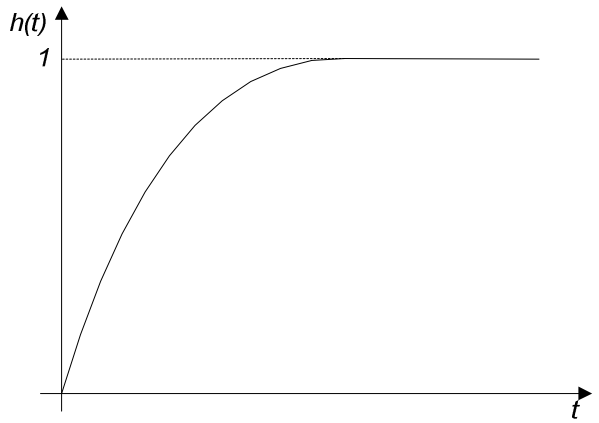


Figura 9 – Resposta do sistema realimentado com um controlador de ação do tipo PI a um degrau unitário

Controladores de ação proporcional-derivativos (PD)

Nos controladores de ação proporcional-derivativo a relação entre a entrada e a saída do controlador é dada pela Equação (7) e a curva de resposta típica é apresentada na Figura 10.

Domínio do tempo: $$u(t) = K_p.e(t) + K_pT_d\frac{de(t)}{dt}$$

Domínio da freqüência: $$U(s) = K_p.(1 + T_d.s).E(s) \quad (7)$$

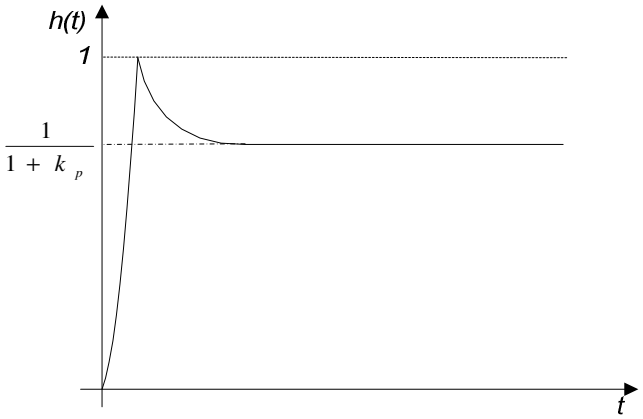


Figura 10 – Resposta do sistema realimentado com um controlador de ação do tipo PD a um degrau unitário

Controladores de ação proporcional-integral-derivativos (PID)

Nos controladores de ação proporcional-integral-derivativo a relação entre a entrada e a saída do controlador é dada pela Equação (8) e a curva de resposta típica é apresentada na Figura 11.

Domínio do tempo:

$$u(t) = K_p.e(t) + \frac{K_p}{T_i}\int_o^t e(t)dt + K_p T_d \frac{de(t)}{dt} \quad (8)$$

Domínio da freqüência:

$$U(s) = K_p.\left(1 + \frac{1}{T_i.s} + T_d.s\right).E(s)$$

Em função dos parâmetros Kp, Ti e Td escolhidos, o sistema pode apresentar um comportamento sub amortecido, superamortecido ou com amortecimento crítico (Figura 11).

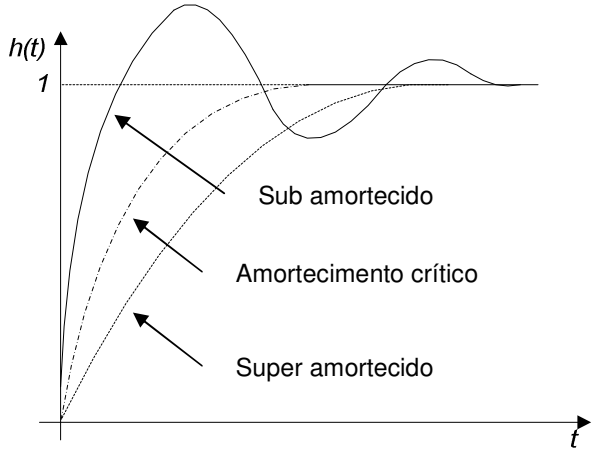


Figura 11 – Respostas típicas de um sistema realimentado com um controlador de ação do tipo PID a um degrau unitário

Um sistema estável é definido como aquele que, para uma entrada limitada, possui uma resposta limitada (Figura 12).

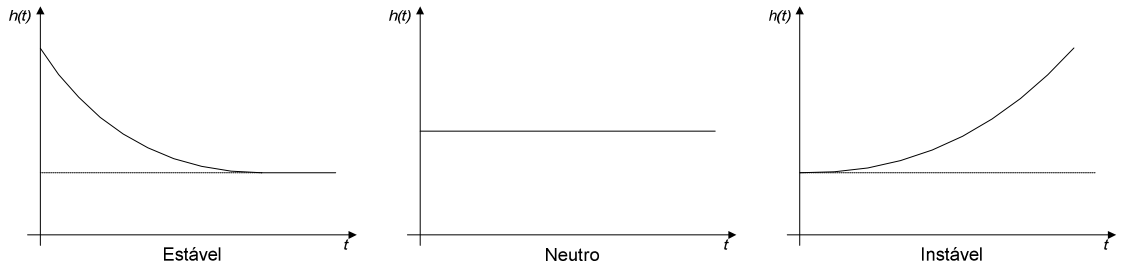


Figura 12 – Estabilidade de um sistema

Para averiguar a estabilidade de um sistema com realimentação, pode-se determinar as raízes do polinômio característico. Conseqüentemente, vários métodos têm sido desenvolvidos para fornecer a resposta, sim ou não, requerida para a pergunta sobre a estabilidade (Dorf, 2009). As três abordagens para essa questão da estabilidade são:

- Abordagem do plano *S*;
- Abordagem no domínio da freqüência (*jw*) e
- Abordagem no domínio do tempo.

A estabilidade de um sistema de controle em malha fechada está diretamente relacionada com a posição das raízes, em malha fechada, da equação característica no plano S. Freqüentemente é necessário ajustar-se um ou mais parâmetros do sistema para se obter as posições adequadas das raízes. Assim, a resposta transitória de um sistema de controle com realimentação em malha fechada pode ser descrita em termos da posição dos pólos da função de transferência (Dorf, 2009).

1.3 Controle Digital

O uso de computadores digitais na malha de controle possui as seguintes vantagens, se comparados a um controlador analógico:

- Redução no custo;
- Flexibilidade para realizar mudanças de projeto;
- Imunidade ao ruído.

Os sistemas de controle modernos requerem o controle de numerosas malhas (pressão, posição, velocidade, tensão, etc.). As malhas contendo sinais analógicos e digitais devem coexistir numa malha de controle e isto acontece através dos dispositivos de conversão *digital-analógica* (D/A) e de conversão *analógico-digital* (A/D). A conversão digital-analógica é simples e efetuada através de soma de tensões ponderadas. A conversão analógico-digital tipicamente é um processo de duas etapas. Na conversão A/D o sinal da entrada é primeiramente convertido em sinal amostrado e depois transformado em uma seqüência de números binários, ou seja, o sinal é amostrado em intervalos periódicos e mantido constante durante o período de amostragem por meio de um dispositivo chamado *amostrador-retentor de ordem zero (z.o.h.)* até a quantificação. Este processo é ilustrado na Figura 13(a), aonde a chave é comutada segundo uma taxa de amostragem uniforme. Este processo também pode ser considerado como o produto de uma forma de onda no

domínio do tempo a ser amostrada, $f(t)$, por uma função de amostragem, $s(t)$, Figura 13(b). Se $s(t)$ for uma seqüência de pulsos de largura T_w, amplitude constante e taxa uniforme, a saída amostrada $f_{T_w}^*(t)$, consistirá de uma seqüência de segmentos de $f(t)$ nos intervalos regulares, idêntica a saída apresentada na Figura 13(a).

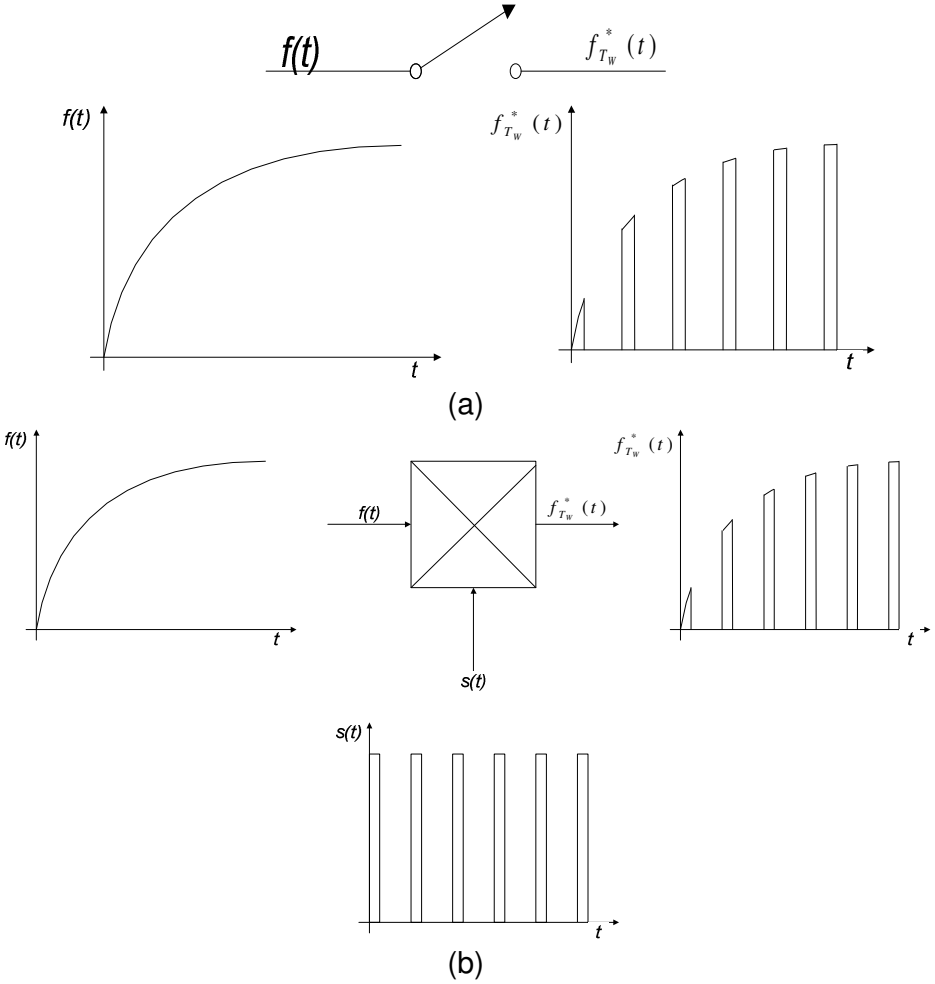


Figura 13 – Amostrador (a) e respectivo modelo (b)

Modelagem de um controlador digital

O fato de sinais serem amostrados e retidos com valor constante em intervalos específicos faz com que o desempenho do sistema se altere ao se mudar a taxa de amostragem. Basicamente, então, o efeito do computador sobre o sinal decorre dessa amostragem e da retenção. Portanto, para modelar sistemas de controle digital, deve-se falar primeiramente de uma representação matemática desse processo de amostragem e retenção (*sample-and-hold*).

O amostrador, Equação (9), é dividido em duas partes (Nise, 2002):

1. Um amostrador ideal descrito pela parte da Equação (9) que não é dependente das características da forma de amostragem e
2. Uma parte dependente das características da forma de onda de amostragem, T_w.

$$f_{T_w}^*(t) = T_w \sum_{k=-\infty}^{\infty} f(kT)\,\delta(t - kT) \tag{9}$$

O passo final na modelagem do computador digital é modelar o extrapolador de ordem zero que segue o amostrador (Figura 14), que consiste em reter o último valor amostrado de $f(t)$. O extrapolador de ordem zero é modelado pela Equação (10).

$$G_h(s) = \frac{1 - e^{-Ts}}{s} \tag{10}$$

O resultado final é apresentado na Figura 14.

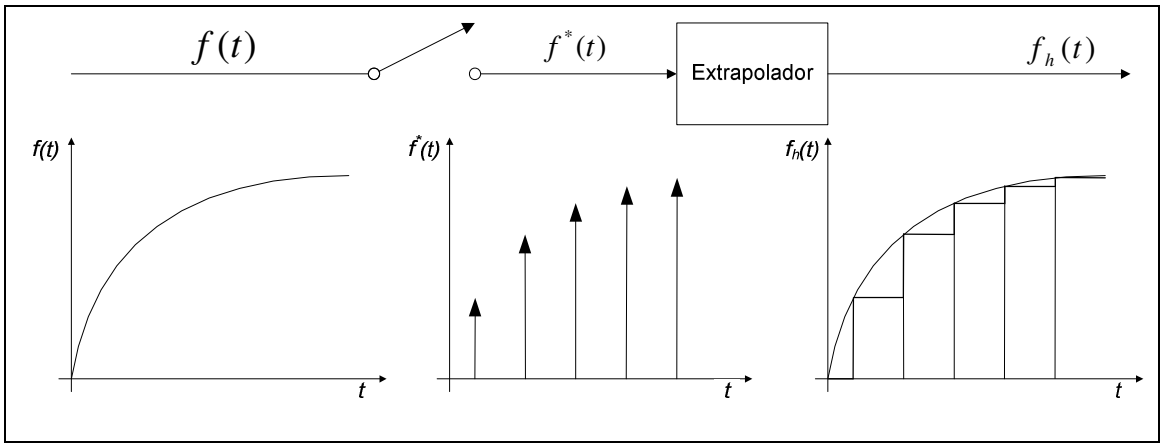


Figura 14 – Amostrador/Estrapolador

Estabilidade num controle digital

A diferença evidente entre os sistemas de controle com retroação analógicos e os sistemas com retroação digitais é devido ao efeito que a taxa de amostragem tem sobre a resposta transitória. Uma mudança na taxa de amostragem não somente altera a natureza da resposta de superamortecida para subamortecida,

mas também pode transformar um sistema estável em um sistema instável (Nise, 2002).

No plano *s*, a região de estabilidade é o semiplano da esquerda (Figura 15). Se a função de transferência, *G(s)*, for transformada na função de transferência com dados amostrados, *G(z)*, a região de estabilidade sobre o plano *z* pode ser calculada a partir da definição apresentada da Equação (11):

$$z = e^{Ts} \tag{11}$$

Se $s = \alpha + j\omega$, obtemos a Equação (12):

$$z = e^{\alpha T} \angle \omega T \tag{12}$$

Assim, cada região do plano s pode ser mapeada na região correspondente sobre o plano *z* (Figura 15). Os pontos que tem valores positivos de α estão no semiplano à direita do plano *s*, região C. Baseado na Equação (12), a magnitude destes pontos mapeados são maiores que 1 ($\alpha > 0$). Portanto, os pontos localizados à direita do semiplano *s* são mapeados no exterior do círculo de raio unitário no plano *z*.

Os pontos sobre o eixo $j\omega$, região B, têm valores nulos de α e produzem pontos no plano *z* com magnitude = 1, o círculo unitário. Portanto, os pontos sobre o eixo $j\omega$ no plano *s* são mapeados nos pontos sobre o círculo unitário do plano *z*.

Finalmente, os pontos do plano *s* que levam a valores negativos de α (raízes do semiplano da esquerda, região A são mapeados no interior do círculo unitário do plano *z*.

Por conseguinte, o sistema de controle digital é:

- Estável se todos os pólos da função de transferência a malha fechada *T(z)* estiverem dentro do círculo unitário do plano *z*;
- Instável se algum pólo estiver fora do círculo unitário e/ou se existir pólos de multiplicidade maior que um sobre o círculo unitário;
- Marginalmente estável se pólos de multiplicidade maior que um estiver sobre o círculo unitário e todos os outros pólos dentro do círculo unitário.

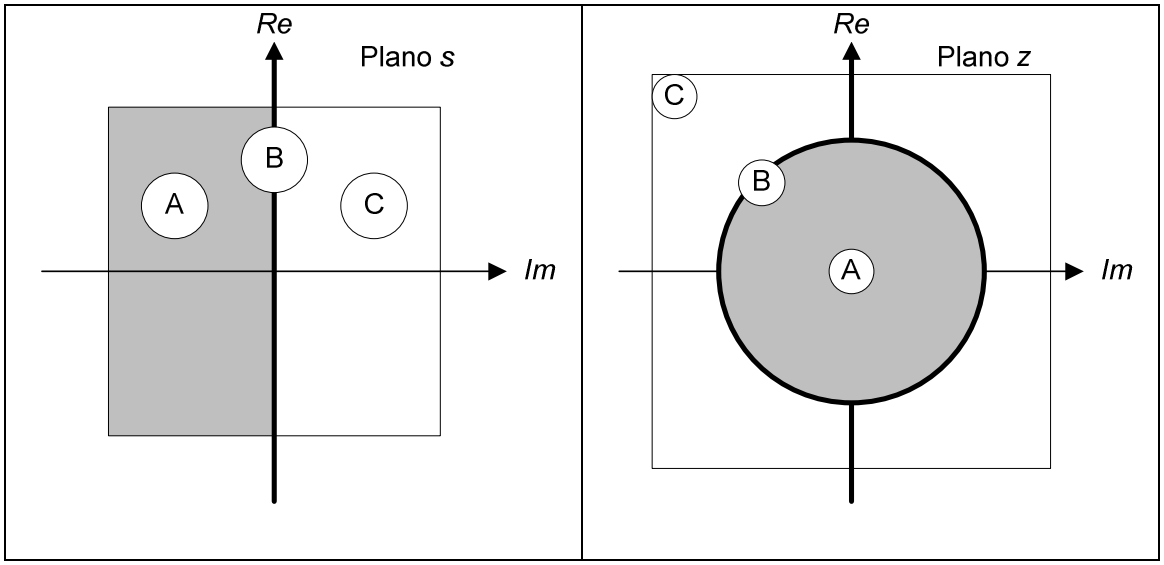


Figura 15 – Correlação entre a estabilidade de um sistema contínuo (a) e um sistema discreto (b)

A maioria dos processos de controle industrial hoje no mundo opera com controladores do tipo proporcional – integral – derivativo (PID) (Ferreira, 2008). Isto se deve muitas vezes aos fornecedores de equipamentos que disponibilizam apenas esta opção aos operadores. Assim, processos que possuem não-linearidades, atrasos de transporte e/ou parâmetros variantes no tempo acabem sendo controlados de forma insatisfatória pelos controladores PID clássicos. Controladores auto-sintonizáveis e controladores baseados em lógica fuzzy têm apresentado resultados promissores quando comparados aos obtidos até então com os controladores PID clássicos (Ferreira, 2008).

2. DISPOSITIVOS REMOTOS

Atualmente, informação é a palavra-chave em muitas empresas mundo afora. Não só as que trabalham diretamente com Informática, mas também as do ramo industrial estão sendo afetadas pelos avanços nas tecnologias de transmissão de dados (Bordim, 2006). A integração entre os diversos níveis de equipamentos e sistemas de controle tem se tornado essencial para alcançar o aumento de eficiência, de flexibilidade e da confiabilidade nos sistemas produtivos.

Tal como nos outros mercados de comunicação de dados (Telefonia, Rádios, Emissoras de Televisão, Internet, etc), os sistemas de transmissão de dados nas indústrias começaram de forma bastante simples, utilizando conexões do tipo serial RS-232 e RS-422/485. Com o passar do tempo, as indústrias foram desenvolvendo sistemas de comunicações mais complexos, utilizando tecnologias próprias (protocolos, softwares e hardwares) apropriados para suas necessidades. Redes industriais são essencialmente sistemas distribuídos, ou seja, diversos elementos trabalham de forma simultânea a fim de supervisionar e controlar um determinado processo. Tais elementos (sensores, atuadores, CLP's, CNC's, PC's, etc), necessitam estar interligados para trocar informações de forma rápida e precisa. Um ambiente industrial é, geralmente, hostil, de maneira que os dispositivos e equipamentos pertencentes a uma rede industrial devem ser robustos, além de confiáveis e rápidos.

Para implementar um sistema de controle distribuído, baseado em redes, há a necessidade de estudos detalhados acerca do processo a ser controlado, buscando-se o sistema que melhor se adeque às necessidades do usuário.

Os fabricantes de sistemas de integração industrial tendem a lançar produtos compatíveis com sua arquitetura proprietária, o que leva a graves problemas de compatibilidade entre as diversas redes e sub-redes presentes nos sistemas, em diversos níveis, equipamentos, dispositivos, hardware e software.

As arquiteturas de sistemas abertos tendem a seguir padrões, de maneira que o usuário pode encontrar diversas soluções diferentes para o mesmo problema.

Uma possível classificação para as redes industriais é apresentada por Lian (Lian et al., 1999), onde tais redes são padronizadas em três níveis hierárquicos responsáveis pela interconexão de diferentes tipos de equipamentos.

O nível mais alto é, geralmente, o que interliga os equipamentos responsáveis pelo planejamento da produção, *scheduling*, controles de estoque, estatísticas de qualidade, previsões de vendas, etc. Geralmente é implementado utilizando-se softwares gerenciais, tais como sistemas SAP, Arena, etc. O protocolo TCP/IP com padrão ethernet é o mais utilizado nesse nível.

No nível intermediário, onde temos os CLP's e CNC's, principalmente, trafegam informações de controle no nível de máquinas contendo informações a respeito do estado dos equipamentos tais como robôs, máquinas-ferramentas, transportadores, etc.

O terceiro nível, mais baixo, é o que se refere à parte física da rede, onde se localizam os sensores, atuadores, contatores, etc.

2.1 Redes de sensores

As redes industriais possuem três níveis e o tipo de equipamento conectado à rede industrial é que determina a classificação, conforme abaixo.

Rede *sensorbus* - dados no formato de bits

A rede *sensorbus* conecta equipamentos simples e pequenos diretamente à rede. Os equipamentos deste tipo de rede necessitam de comunicação rápida em níveis discretos e são tipicamente sensores e atuadores de baixo custo. Estas redes não almejam cobrir grandes distâncias, sendo sua principal preocupação manter os custos de conexão tão baixos quanto for possível. Exemplos típicos de rede sensorbus incluem Seriplex, ASI e INTERBUS Loop.

Rede *devicebus* - dados no formato de bytes

A rede devicebus preenche o espaço entre redes *sensorbus* e *fieldbus* e pode cobrir distâncias de até 500 m. Os equipamentos conectados a esta rede terão mais pontos discretos, alguns dados analógicos ou uma mistura de ambos. Além disso, algumas destas redes permitem a transferência de blocos em uma menor prioridade comparada aos dados no formato de bytes. Esta rede tem os mesmos requisitos de transferência rápida de dados da rede de sensorbus, mas consegue gerenciar mais equipamentos e dados. Alguns exemplos de redes deste tipo são DeviceNet, Smart Distributed System (SDS), Profibus DP, LONWorks e INTERBUS-S.

Rede *fieldbus* - dados no formato de pacotes de mensagens

A rede *fieldbus* interliga os equipamentos de I/O mais inteligentes e pode cobrir distâncias maiores. Os equipamentos acoplados à rede possuem inteligência para desempenhar funções específicas de controle tais como loops PID, controle de fluxo de informações e processos. Os tempos de transferência podem ser longos, mas a rede deve ser capaz de comunicar-se por vários tipos de dados (discreto, analógico, parâmetros, programas e informações do usuário). Exemplos de redes *fieldbus* incluem IEC/ISA SP50, *Fieldbus Foundation* e *Profibus PA*.

2.2 Redes sem fio

Os primeiros dispositivos a serem operados por controle remoto foram desenvolvidos pela Marinha alemã durante a 1ª Guerra Mundial. Tratava-se de barcos controlados por rádio que eram utilizados para atacar os navios inimigos. Durante a 2ª Guerra Mundial, bombas e outras armas de acionamento remoto tiveram uso em larga escala.

Após o término da 2ª Guerra Mundial, a empresa Zenith (Zenith, 2008). lançou, em 1954, a 1ª TV comandada por controle remoto. O sistema era denominado *Lazy Bones* e utilizava fios entre o controle e o aparelho. Em 1955, a mesma empresa, aperfeiçoou o sistema para o uso de luz no lugar de fios, *Flash-o-Matic*, e, em 1957, lançou o sistema *Space Command*, que utilizava luz infravermelha.

Nos anos seguintes, as indústrias desenvolveram várias soluções para a comunicação sem fio, cujo enfoque sempre foi a transferência de informações à distância sem o uso de fios.

As Redes de Sensores Sem Fio (RSSF) ganham maior viabilidade a cada ano, e sua utilização é cada vez mais imprescindível no sensoriamento dos mais diversos ambientes. Estas redes podem ser instaladas em praticamente todos os tipos de ambientes, graças ao seu tamanho reduzido, sua facilidade de comunicação e seu baixo custo. Uma característica importante é a capacidade de transportar os dados através de nós que utilizam algoritmos de roteamento e, por este motivo, podem cobrir grandes áreas geográficas apresentando uma boa tolerância a falhas.

O conhecimento das principais arquiteturas e dos protocolos envolvidos na comunicação, bem como o comportamento destas redes em ambientes reais é de extrema importância para a compreensão do funcionamento da rede e dos

fenômenos que prejudicam o desempenho. Outros aspectos, como a durabilidade das fontes de energia e o conhecimento das principais fontes de interferências eletromagnéticas, também contribuem para o bom funcionamento da rede.

Estes dispositivos remotos, que começam a ocupar uma fatia do mercado mundial, fizeram com que vários fabricantes já buscassem uma padronização dos protocolos de comunicação, através de alianças internacionais.

Os sensores que compõem uma rede sem fio (RSSF) são compostos (Khemapech et al., 2007), basicamente, de:

- Unidade sensora;
- Unidade de processamento;
- Unidade transceptora;
- Fonte de alimentação.

Na unidade sensora, a medida física (alvo de interesse) é transformada em informação elétrica proporcional ao fenômeno e, em seguida, convertida em sinal digital através de um conversor A/D. Este sinal é disponibilizado para a unidade de processamento.

Na unidade de processamento ocorre a maior parte do processamento do sensor. Esta unidade é implementada por microprocessadores, microcontroladores ou FPGAs. Nesta unidade também existe uma memória não-volátil (para armazenamento de dados ou parâmetros do dispositivo) e, em alguns casos, o próprio conversor A/D ou uma interface para este conversor.

A unidade transceptora é responsável pela interface entre o sensor e o meio físico da rede, isto inclui comunicações óticas (laser), infravermelho e rádio-freqüência (RF).

O consumo de energia é o ponto fraco de um sensor remoto, pois está diretamente ligado a vida útil das baterias, as quais podem ser recarregáveis ou não. O gerenciamento de energia do sensor pode ser realizado pela fonte de alimentação.

2.3 Vantagens e desafios da rede sem fio

Tendo como principal característica o fato de não necessitar de infra-estrutura, as redes sem fio apresentam uma grande vantagem na implementação pois possuem um custo inferior aos sensores convencionais, principalmente em

instalações cujos sensores estão constantemente sofrendo mudanças no *layout* ou em movimento.

Porém, as redes sem fio possuem vulnerabilidades, como, por exemplo, interferências eletromagnéticas e alcance variável, pois dependem das condições topológicas e ambientais. Além dessas vulnerabilidades, existe a preocupação com infiltração de usuários que não pertencem à rede e que podem adquirir informações sigilosas ou interferir de forma danosa nas informações que trafegam pela rede.

Como resultado do crescente uso da tecnologia sem fio e da estrutura de TI (Tecnologia de Informação), cresce também a vulnerabilidade do sistema que normalmente são alvos de:

- Vírus e *Worms* – programas mal intencionados que controlam e propagam-se pelo sistema, consumindo recursos computacionais e transmitindo informações para usuários não autorizados;
- D.o.S. – *Denial of Service*, ou Degradação do Serviço, consiste de um "bombardeio" de solicitações de recursos ou interrupções que fazem com que o sistema fique sobrecarregado, impossibilitando o gerenciamento da "planta";
- Portas Abertas ou *Back Orificies* – surgem quando computadores, CLPs ou dispositivos de rede são instalados e não são configurados corretamente ou simplesmente não são configurados, mantendo os padrões de fábrica, tais como portas de comunicação ou senhas de administrador;
- Espionagem – a espionagem industrial está em ampla expansão graças a vários programas facilmente encontrados na Internet que conseguem drenar informações da rede e
- Hackers – trata-se de um indivíduo que, por diversão ou contratado, invade a rede a fim de interferir ou roubar informações.

A fim de preservar a estabilidade e informações da rede, surgiu no mercado um novo tipo de serviço: Segurança de Controle de Processo. Trata-se de um novo ramo de serviços que está crescendo junto com a expansão da rede sem fio industrial (Verhappen, 2004).

2.4 Utilização de redes sem fio

Uma das tendências na área de controle de processos e automação industrial é o uso em larga escala da comunicação digital. Isto se torna evidente ao se observar a quantidade, cada vez maior, do uso de sistemas em rede nos sistemas de controle.

Outra tendência é o uso de tecnologia sem fio, especificamente as baseadas na IEEE 802 (Wi-Fi, WiMAX, Bluetooth ou, mais recentemente, no Wireless Hart e ZigBee). Esta última, por utilizar a faixa de freqüências ISM (*Industrial, Scientifical and Medical*), tem atraído um número cada vez maior de adeptos, pois não necessita de licença para a operação (Stojmenovic, 2005).

Uma rede de sensores sem fio é composta de dispositivos sem fio capaz realizar medições e transmiti-las através de um canal sem fio para um dispositivo de tomada de decisões com base nas leituras dos sensores.

O projetista de uma rede de sensores sem fio se depara com vários desafios, que vão desde a necessidade de alta taxa de transmissão, com tolerância a falhas, à limitação de energia dos nós, o que significa baixa potência de transmissão, baixa capacidade de processamento e memória reduzida.

Para que as aplicações de rede de sensores sem fio tenham durabilidade razoável, é obrigatória uma política agressiva de gerenciamento de energia. Este é atualmente o maior desafio nos projetos, para qualquer aplicação, de rede de sensores sem fio. Considerando que o custo energético associado à transmissão de um byte no transmissor é substancialmente maior do que o necessário para realizar computação local, os projetistas devem priorizar a capacidade de processamento local para minimizar o consumo da bateria, bastante exigida nas comunicações via rádio. Existem várias diferenças importantes entre as mais tradicionais redes de sensores sem fios.

- Os nós individuais em uma rede de sensores sem fio têm poder computacional, energia e capacidade de armanezamento limitados. Eles operam em fontes de energia não-renováveis e empregam um transceptor de curto alcance para enviar e receber mensagens.
- O número de nós de uma rede de sensores sem fio pode ser muito grande. Assim, a capacidade do algoritmo de tratar escalabilidade da rede é um critério de projeto importante para aplicações de rede de sensores.

- Os nós sensores são, geralmente, densamente implantados na área de interesse. Estas informações podem ser tratadas pela aplicação local, uma vez que estes nós que se encontram próximos podem colaborar localmente antes de veicular a informação de volta à estação base.
- Redes de sensores são passíveis de mudanças freqüentes na topologia. Isto é devido a várias razões, tais como falha de hardware, pilhas gastas, interferência nos sinais de rádio, fatores ambientais ou a adição de sensores. Como resultado, estas aplicações exigem um grau de tolerância a falhas e a capacidade de se reconfigurar à medida que a topologia da rede evolui ao longo do tempo.

2.5 ZigBee

ZigBee é o nome dado para um conjunto de protocolos para comunicação sem fio. Oriundo do *Home RF Lite*, um sistema desenvolvido pela PHILIPS para rede de sensores, este sistema foi batizado comercialmente de ZigBee devido a forma com que a informação trafega pela rede. Assim como as abelhas voam em *zig-zag* pelas flores indicando o caminho dos alimentos para as outras abelhas da colméia, na rede original, os dados trafegavam pela rede, do tipo malha, até que um caminho seja encontrado entre o módulo coordenador e o dispositivo final (*ZigBee Alliance*, 2009).

Após a criação do grupo TG4 da IEEE e a formação da *ZigBee Alliance*, uma aliança formada por mais de 200 empresas de mais de 20 paises, o sistema teve a primeira versão oficial em 27 de julho de 2005 (Figura 16).

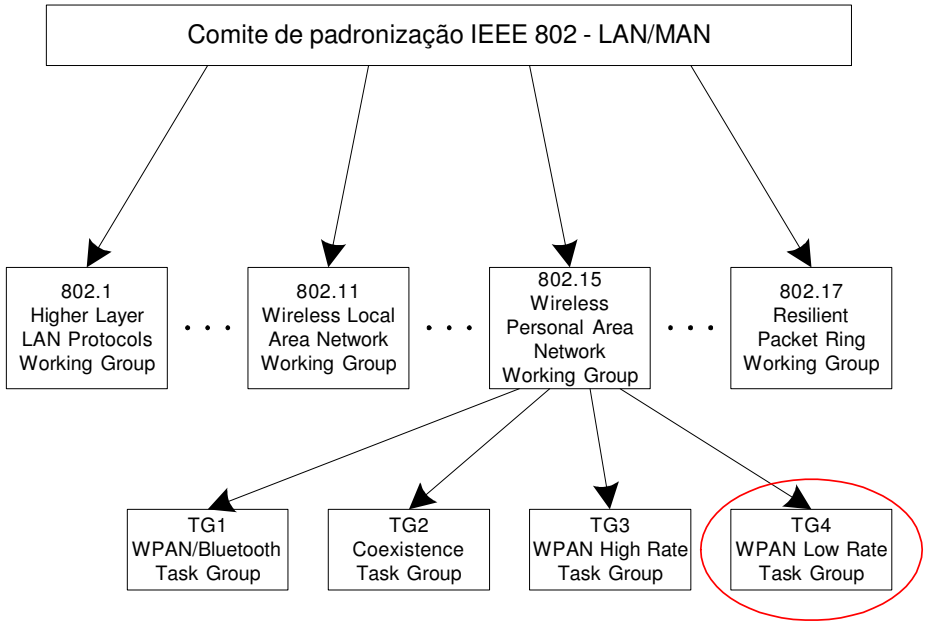


Figura 16 – TG4 na estrutura IEEE

Utilizando rádios de baixa potência baseados no padrão IEEE 802.15.4, estes rádios configuram uma *Wireless Personal Area Networks* (WPANs) e operam nas faixas de freqüência ISM (*Industrial, Scientifical and Medical*), as quais não necessitam de licença.

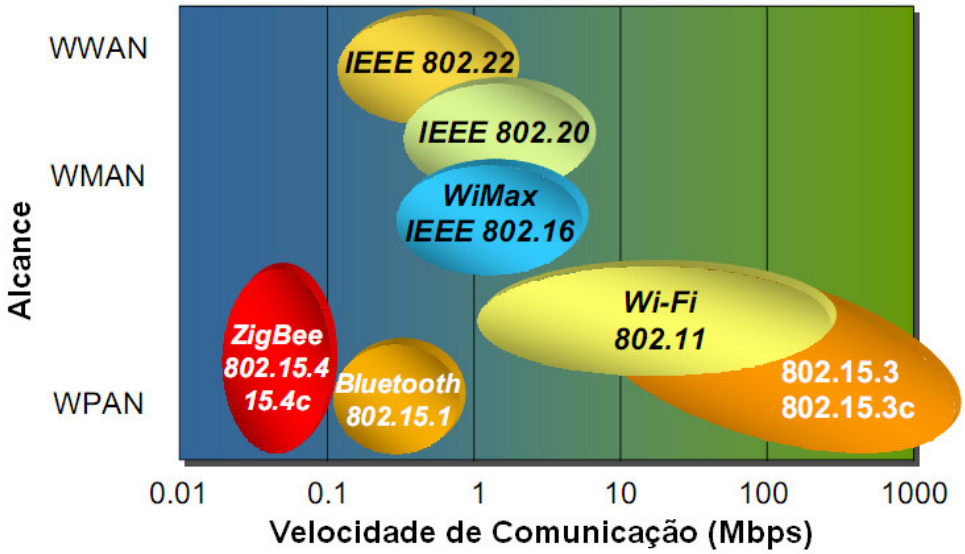


Figura 17 – Espectro de padrões *wireless*.

A aliança classifica os dispositivos em dois tipos, os FFD (*Full Function Devices*) e os RFD (*Reduced Function Devices*). Apesar de só existirem dois tipos de dispositivos físicos, foram definidos três tipos de dispositivos lógicos, a saber:

- Coordenador;
- Roteador;
- Elemento final.

As atribuições destes dispositivos estão resumidas na Tabela 1.

Tabela 1 – Atribuições dos dispositivos normatizados pela *ZigBee Alliance*.

Dispositivo	Tipo de dispositivo físico	Função/Características
Coordenador	FFD	✓ Só existe um por rede; ✓ Forma a rede, atribui endereços e suporta *binding table*.
Roteador	FFD	✓ Existência opcional; ✓ Permite que mais nós se juntem à rede (aumentam o alcance físico); ✓ Pode efetuar ações de controle ou monitoração.
Elemento Final	RFD ou FFD	✓ Efetua ação de controle ou monitoração através do dispositivo a ele associado (sensor, atuador, ...).

Na rede ZigBee, apesar da existência de um coordenador, responsável pela inicialização e controle da rede, a filosofia é do tipo *ad hoc*, ou seja, não existe uma topologia predeterminada e é capaz de se reconfigurar dinamicamente. A rede ZigBee suporta três topologias: estrela, árvore e malha (Figura 18).

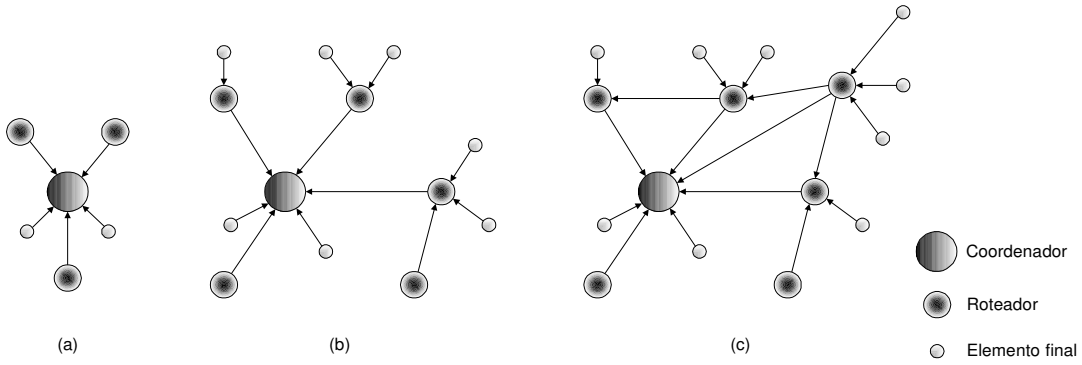


Figura 18 – Topologia da rede: (a) estrela, (b) árvore e (c) malha.

Na topologia estrela (Figura 18a) cabe ao coordenador todo o controle da rede, assumindo a comunicação direta com todos os dispositivos.

Na topologia árvore (Figura 18b) os roteadores são utilizados para expandir a rede (maior alcance físico) e os dispositivos finais (RFD) podem se comunicar tanto com o coordenador quanto com o roteador, em função do alcance. Porém, os roteadores não se comunicam entre sí.

Na topologia malha (Figura 18c) os roteadores, assim como na topologia em árvore, além de aumentar o alcance da rede, podem se comunicar entre si ou com o coordenador. Isto reduz significativamente o fluxo de informações que passam pelo coordenador, o qual, por sua vez, já não assume um papel tão predominante.

Na Figura 19, na Figura 20 e na Figura 21 são apresentados alguns exemplos de uma RSSF (Zigbee) em uma aplicação na área agrícola, de pecuária e domótica (automação doméstica) (Rogercom, 2007). Na área agrícola e pecuária a eficiência da rede é mais favorecida, pois os dispositivos encontram-se em campo aberto (longe das interferências dos grandes centros urbanos) e o alcance da comunicação entre os dispositivos não é prejudicado por obstáculos (estruturas metálicas, pontes, viadutos e prédios). Já na domótica, apesar da existência de obstáculos pertinentes a cada ambiente, os dispositivos encontram-se tão próximos que a redução de alcance não é percebida pelo usuário.

Na Figura 19 é apresentada uma aplicação na pecuária onde a rede ZigBee é utilizada para a monitoração do nível de água dos açudes, rios, ou bebedouros, detecção de arames rompidos na cerca, saber o local onde os animais permanecem mais tempo pastando, controlar a irrigação do pasto, controlar o abre/fecha de cancelas, etc.

Figura 19 – Aplicação na pecuária (Rogercom, 2007)

Na Figura 20 é apresentada mais uma aplicação, agora na agricultura. Através de uma Rede ZigBee de sensores tais como: umidade relativa do ar, umidade do solo, pressão atmosférica, temperatura do ar, temperatura do solo, luminosidade, velocidade do vento, direção do vento e quantidade de chuva num certo intervalo de tempo, é possível após a obtenção dos dados, cruzar os mesmos com informações do tipo: data, hora, estação do ano, tipo de plantação, tipo do solo da região, fases da lua, entre outras, e assim gerar um relatório de informações precisas sobre o por que e quando certas pragas se proliferaram na plantação. Após as análises das informações, fica fácil para um profissional agrônomo, detectar e dar uma solução ao problema na plantação (Agricultura de Precisão, 2009).

Na Figura 21 é apresentada uma aplicação em domótica. A Figura 21(a) apresenta uma central ZigBee de climatização doméstica de fabricação da empresa Ecobee (Ecobee, 2007) e a Figura 21(b) uma central de uso geral aonde podem ser vistos os sensores tipo *reed*, de incêndio, tomadas de acionamento remoto (Zigbee, 2007).

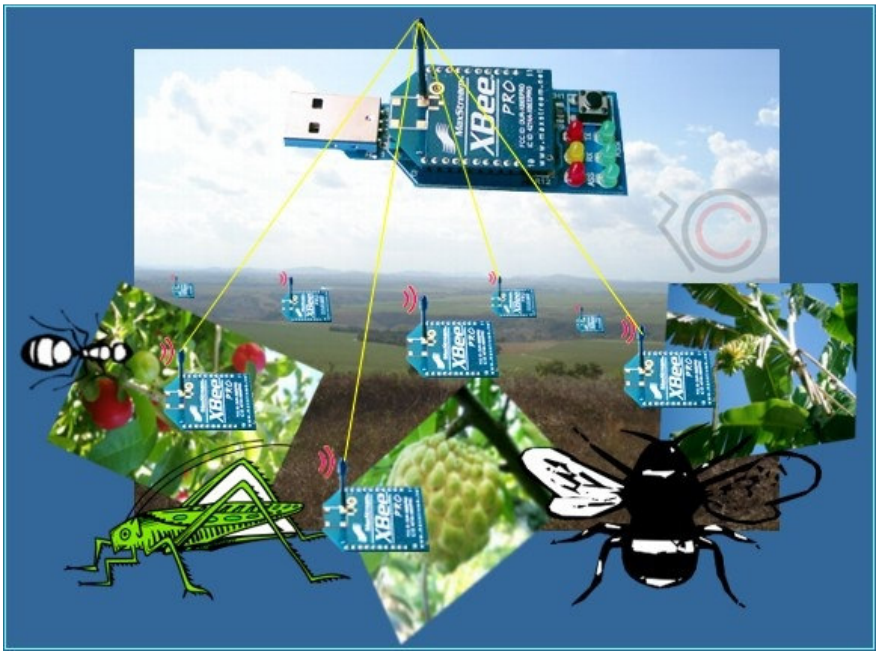


Figura 20 – Aplicação na agricultura no controle de pragas (Rogercom, 2007)

(a) (b)

Figura 21 – Aplicação na domótica – automação residencial (Ecobee, 2007)

3. O CONTROLE DIGITAL EM UMA REDE DE SENSORES

Com o crescimento da complexidade dos sistemas de automação modernos, os Sistemas de Controle em Redes (NCS – *Networked Control Systems*) ganham cada vez mais importância devido à modularidade e simplicidade de diagnósticos. Num NCS, o controlador e o processo encontram-se fisicamente distantes e o laço de controle é fechado através de uma rede de comunicação (Figura 22). Entretanto, uma rede de comunicação introduz um atraso no laço que degrada o controle e, em alguns casos, levam a instabilidade [Matiakis et. al, 2008].

Os atrasos introduzidos pela rede são inevitáveis, devido ao processo de "agendamento" de mensagens. As perdas de pacotes acontecem, esporadicamente, devidas, por exemplo, ao congestionamento da rede. Há que se considerar também que uma taxa de comunicação infinita não existe, devido à largura de banda limitada disponível [Teng et. al, 2008].

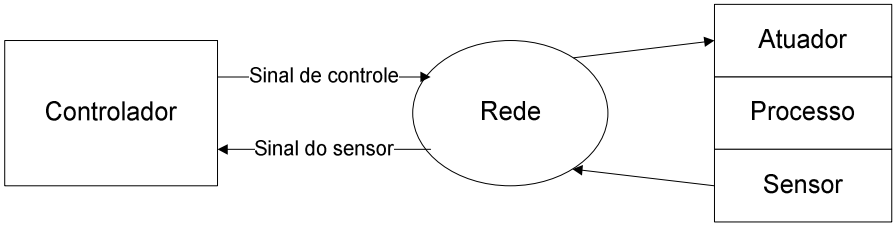


Figura 22 – Arquitetura de uma rede de sensores genérica [Tipsuwan, 2003]

Ao analisarmos os modelos apresentados na Figura 23 e Figura 24, cujas funções de transferência são representadas pelas equações (13) e (14), respectivamente, observa-se que tanto o atraso da rede (*a*) quanto a perda de pacotes (*T*) alteram o(s) pólo(s) do sistema.

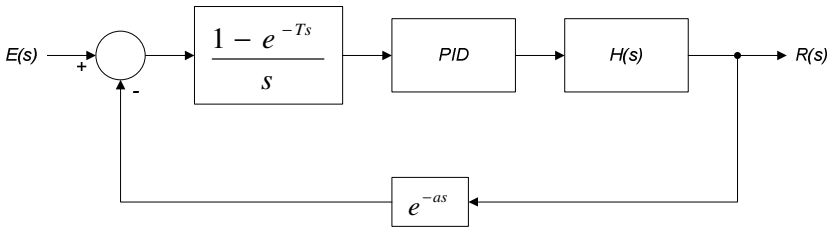


Figura 23 – Modelo de um NCS com atraso na realimentação

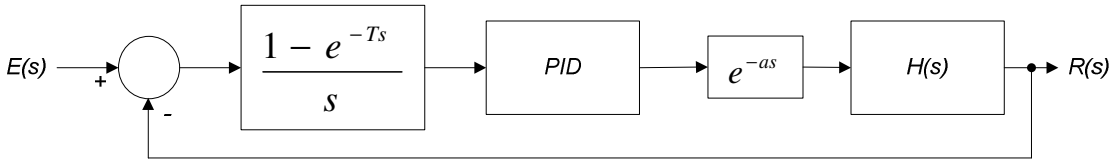


Figura 24 - Modelo de um NCS com atraso na atuação

$$T(s) = \frac{R(s)}{E(s)} = \frac{G(s)}{1 + e^{-as}.G(s)} \Rightarrow T(z) = \cdots \quad (13)$$

$$T(s) = \frac{R(s)}{E(s)} = \frac{e^{-as}.G(s)}{1 + e^{-as}.G(s)} \Rightarrow T(z) = \cdots \quad (14)$$

Tendo em vista que o mapeamento dos pólos da função de transferência no plano z depende dos pólos originais da função de transferência do sistema no domínio do tempo, da taxa de amostragem e do atraso na malha de realimentação, uma alteração da taxa de amostragem ou a inserção de mais atrasos na malha de controle podem fazer com que o controle passe de superamortecido para sub-amortecido ou de estável para instável (Nise, 2002).

3.1 Temporização de um NCS

Quando uma informação é transferida de um dispositivo para outro numa rede de sensores, várias etapas são necessárias. Inicialmente um pulso de sincronismo dispara a rotina de amostragem, envelopamento e transmissão desta informação. Quando esta informação é recebida pelo outro dispositivo da rede de sensores, ele é convertido novamente em sinal analógico ou aproveitado diretamente pelo controlador do processo. Neste processo, as etapas necessitam de tempo para processamento, os quais são discriminados na Figura 25(a) para a transmissão e na Figura 25(b) para a recepção. No *toolbox True Time*, utilizado neste trabalho, todos estes tempos são configuráveis pelo usuário, a saber:

- t_{mi} – tempo morto inicial
  Tempo necessário para a conclusão da instrução ou operação em curso pelo processador do dispositivo.
- t_{conv} – tempo de conversão
  Tempo necessário para a conversão do sinal analógico em digital.

- t_{exec} – tempo de execução da rotina
  Tempo necessário para o envelopamento da informação, incluindo o cálculo de *checksum*, CRC, etc., além da inserção dos códigos de início e término de pacote.
- t_{mf} – tempo morto final
  Tempo existente entre o envio do pacote ao módulo de transmissão e o inicio do envio do pacote.
- t_{lib} – tempo de liberação do canal
  Tempo entre a primeira tentativa de transmissão e o início efetivo da transmissão. Este tempo expressa o grau de ocupação do canal.
- $t_{amostra}$ – tempo de amostragem do sinal
  Tempo entre amostras do sinal.
- t_{tx} – tempo de transmissão
  Tempo necessário para a transmissão da informação pelo canal. Este parâmetro é função da taxa de transmissão suportada pelo canal e o tamanho do pacote.
- t_{rx} – tempo de recepção
  Tempo necessário para a recepção da informação pelo canal. Este parâmetro também é função da taxa de transmissão suportada pelo canal e o tamanho do pacote.

Alguns protocolos possuem um mecanismo de confirmação de recebimento de mensagem, o que implica na transmissão de uma mensagem do dispositivo destinatário para o dispositivo emissor informando o recebimento de uma mensagem válida.

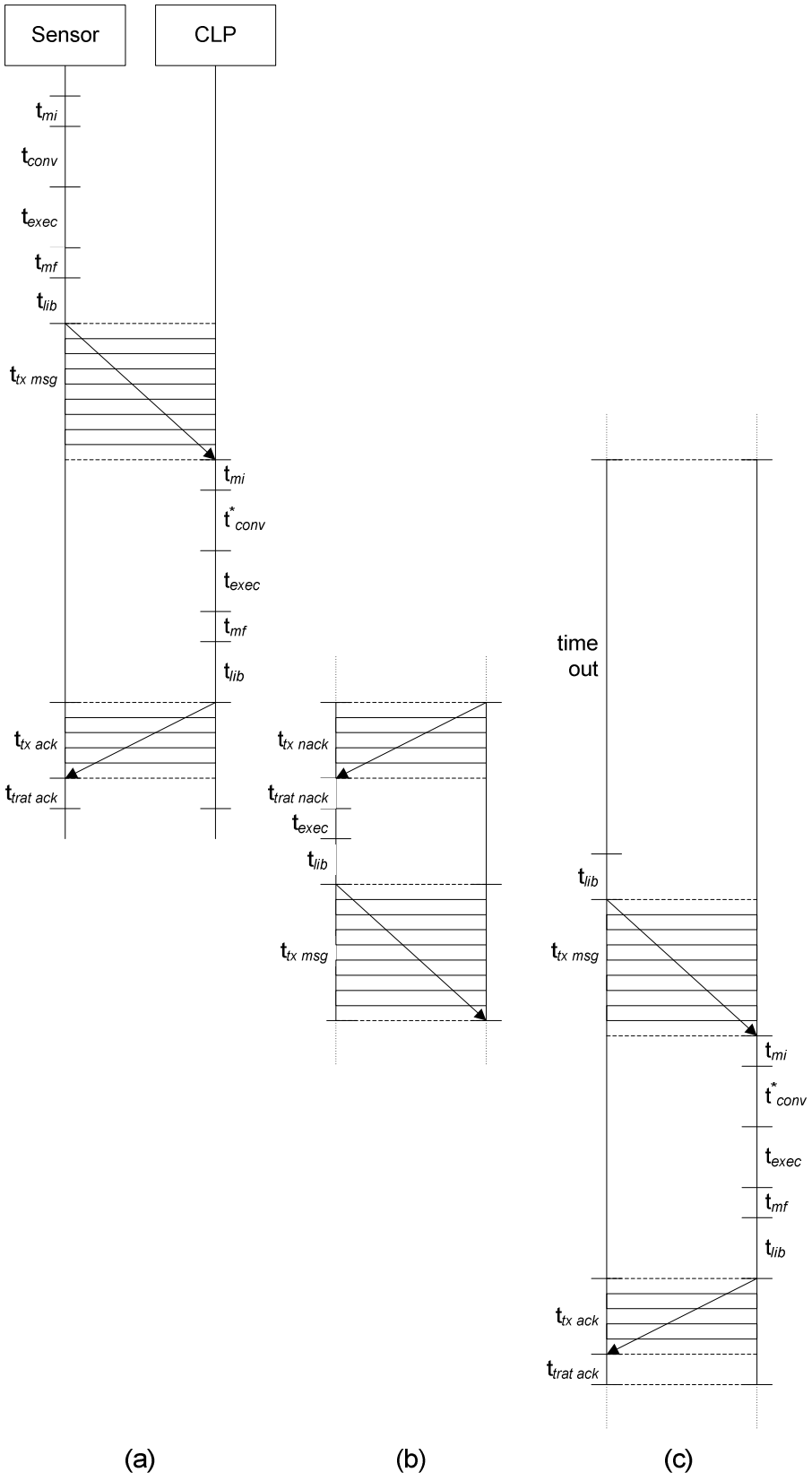


Figura 25 – *Temporização de dados num NCS*

Conforme o canal começa a ficar congestionado ou ruidoso, os pacotes podem não chegar ou chegar com erros. Estes erros obrigam o dispositivo de destino a solicitar o re-envio da mensagem, quer pela falta de retorno de reconhecimento da mensagem quer pela solicitação do dispositivo controlador. Também é possível a chegada de uma mensagem válida, mas a ocorrência de perda do pacote de reconhecimento, implicando no re-envio da mensagem desnecessariamente. Todas estas situações implicam em aumento da ocupação do canal (aumento de t_{lib}), piorando o desempenho do sistema de controle. Um exemplo desta situação pode ser observado na Figura 26. A situação torna-se crítica quando o tempo necessário para o envio de um pacote ultrapassa o tempo de amostragem do sinal ($t_{amostra}$), caracterizando uma redução da taxa de amostragem do sinal.

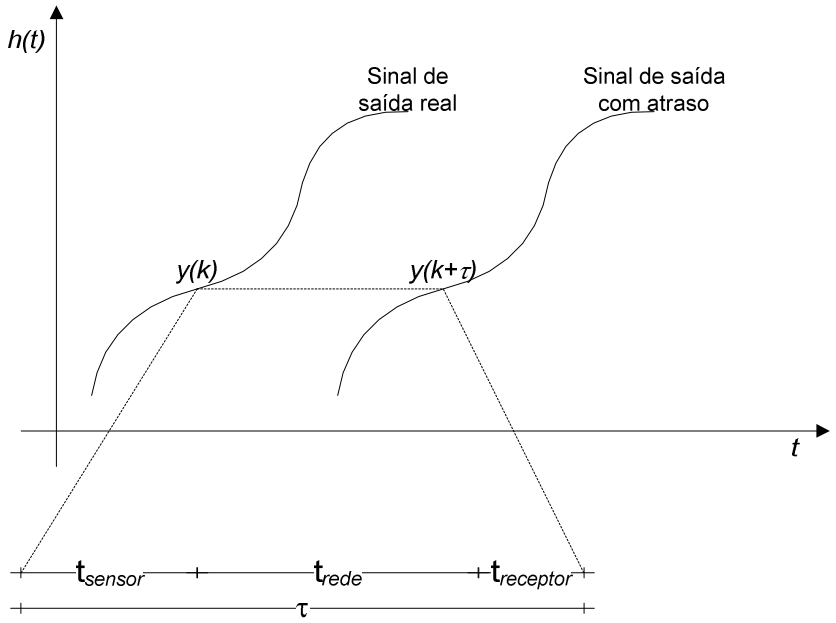


Figura 26 – Atraso de um sinal devido à temporização do NCS [Tipsuwan, 2003]

3.2 Toolbox True Time

O simulador *True Time*, desenvolvido pelo Departamento de Automação e Controle da Universidade de Lund, na Suécia, funciona no Matlab/Simulink e possui ferramentas para facilitar a co-simulação no estudo da execução, em tempo real, de *Kernels*, transmissões em redes e da dinâmica de instalações (Ohlin et. al., 2007).

O simulador é composto por seis blocos (Figura 27), cujas descrições encontram-se abaixo:

- *True Time Kernel*

  O bloco de *Kernel* do *True Time* é utilizado para especificar o número de entradas e saídas do bloco, definir a política de agendamento das mensagens, criação das tarefas, tratamentos de interrupções, eventos, monitoração e etc. na simulação. Isto é efetuado através dos parâmetros de inicialização existentes para cada bloco de *Kernel*.

- *True Time Network*

  O bloco de rede do *True Time* simula o acesso ao meio e a transmissão dos pacotes no canal da rede. Quando um nó inicia a transmissão de uma mensagem (utilizando a primitiva *ttSendMsg)*, um sinal de *trigger* é enviado para o canal de entrada correspondente no bloco de rede. Quando a simulação de transmissão termina, o bloco de rede envia um novo sinal de *trigger* na saída correspondente do nó de recepção. A mensagem transmitida é então colocada no *buffer* de recepção do nó. A mensagem contém informações sobre os nós de transmissão e recepção, dados arbitrados pelo usuário (normalmente sinais de medição ou de controle), tamanho da mensagem e, opcionalmente, atributos opcionais de execução tais como prioridade ou *time out* de mensagem.

  O *True Time* possui suporte para seis tipos de rede:

  - CSMA/CD – *Carrier Sense Multiple Access with Collision Detection* (ex. Ethernet);
  - CSMA/AMP – *Carrier Sense Multiple Access / Arbitration by Message Priority* (ex. CAN – *Controller Area Network*);
  - *Round Robin* (ex. Token Bus);
  - FDMA – *Frequency Division Multiple Access*;
  - TDMA – *Time Division Multiple Access* e
  - *Switched* Ethernet.

  O atraso da propagação é ignorado, já que, tipicamente, este tempo é muito pequeno numa rede local. Só existe suporte até a camada enlace (OSI), já que o co-simulador supõe que os níveis superiores do protocolo de *Kernel* do nó dividem grandes mensagens em pequenos pacotes.

- *True Time Wireless Network*

  O uso do bloco de rede sem fio é similar ao bloco da rede com fio e funciona da mesma forma. A fim de ser possível a simulação da perda de sinal de rádio, existem duas entradas (x e y) que são utilizadas para especificar a localização dos nós. O *True Time* suporta dois protocolos de rede atualmente: IEEE 802.11b/g (WLAN – *Wireless Local Area Network*) e IEEE 802.15.4 (ZigBee). Maiores detalhes sobre a rede sem fio podem ser encontrados em [Ohlin, 2007].

  O pacote é capaz de modelar:

  - Redes sem fio *ad-hoc*;
  - Antenas isotrópicas;
  - Incapacidade de envio e recebimento de mensagens ao mesmo tempo;
  - Perda dos sinais de rádio modelados como $\frac{1}{\mathrm{d}^{\mathrm{a}}}$, onde $\boldsymbol{d}$ é a distância, medida em metros, e $\boldsymbol{\alpha}$ é o parâmetro escolhido para se modelar o sistema e
  - Interferência de outros terminais através da introdução de outros dispositivos na mesma rede.

- *True Time Battery*

  O bloco de bateria possui um único parâmetro, a carga inicial. Este parâmetro pode ser configurado utilizando-se a tela de configuração correspondente. Para se utilizar o bloco de bateria, basta habilitá-lo na mascara de configuração de *Kernel* e conectar a saída da bateria na entrada *E* do bloco de *Kernel*. Este bloco utiliza um integrador para modelar o comportamento da bateria (tanto para a carga, quanto para a descarga).

  Conectam-se cada ponto de consumo de energia, tais como os blocos padrão do Simulink, na saída *P* do bloco de *Kernel* e os blocos da rede sem fio na entrada *P* da bateria.

  Deve-se ter em mente de que o *Kernel* não irá executar nenhum código se estiver configurado para utilizar bateria e a entrada P do bloco de *Kernel* estiver com valor zero.

- *ttGetMsg e ttSendMsg*

  Os blocos da rede, denominados *ttSendMsg* e *ttGetMsg* **(**Figura 27**)**, podem ser utilizados para envio de mensagens através dos blocos de rede sem o uso dos blocos de *Kernel*. Isto torna possível criar simulações no *True Time* sem a necessidade de se inicializar o *Kernel*, de se criar e instalar tratamentos de interrupções, etc.

  Em outras palavras, pode-se criar uma rede inteira no Simulink sem nenhum arquivo do tipo *m* do Matlab/Simulink. Também é possível misturar estes blocos, de modo que algumas estações trabalhem com blocos de rede *Stand Alone* enquanto que outros utilizem as primitivas *ttSendMsg* e *ttGetMsg* de uma função executada pelo bloco de *Kernel*.

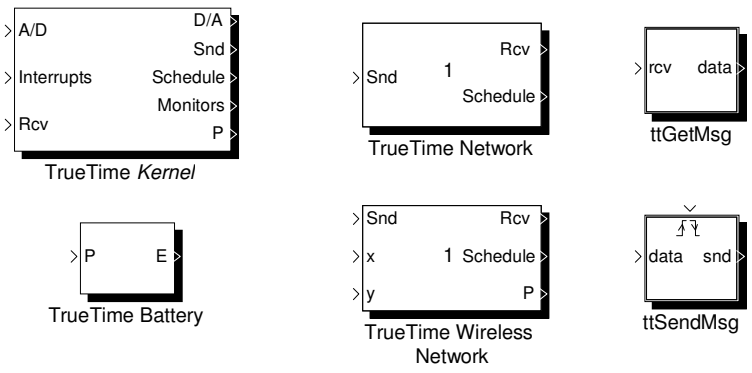


Figura 27 – Blocos do co-simulador *True Time*

O pacote também possui outras características, a saber:

- O sinal que trafega entre a entrada e a saída da rede é do tipo *float*;
- O número de bits especificado no bloco de *Kernel* é única e exclusivamente para a simulação do atraso da transmissão da mensagem. Caso seja necessário o estudo do impacto da quantificação introduzido pela rede, deve-se utilizar o bloco específico do próprio Simulink.

A utilização de uma rede de sensores insere uma não-linearidade na malha de controle (Sadjadi et. al., 2003). Estas características degradam a estabilidade do sistema de controle PID. Vários estudos vêm sendo realizados no intuito de se contornar este problema. A seguir são apresentados alguns destes trabalhos.

3.3 Estudos de soluções

Song et. al. (2006) propõe a arquitetura apresentada na Figura 28 e um conjunto de equações (Equações (15 e (16) para minimizar os efeitos das não-linearidades, conseqüência dos atrasos e perdas de pacotes, na malha de controle.

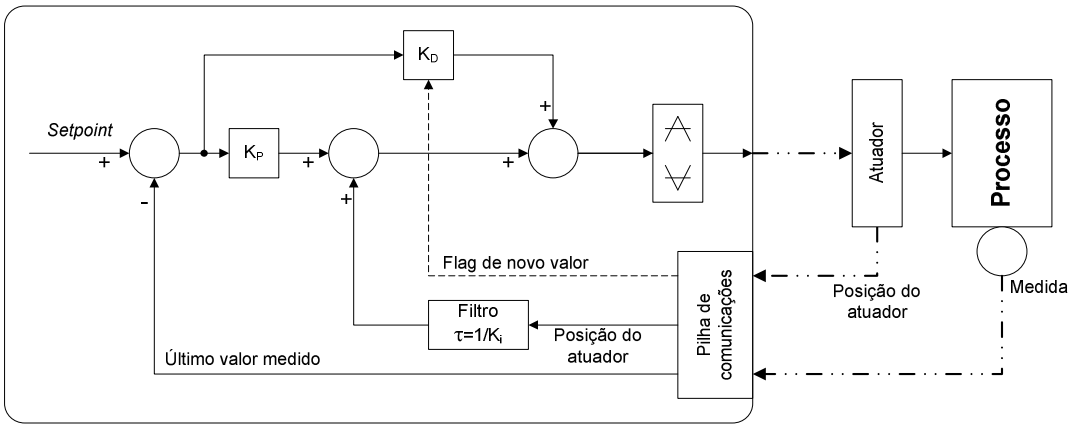


Figura 28 – Algoritmo PID aprimorado

$$F_n = F_{n-1} + (O_{n-1} - F_{n-1}) * \left(1 - e^{\frac{-\Delta t}{t_{reset}}}\right) \quad (15)$$

Onde:

- F_n = novo valor de saída do filtro;
- F_{n-1} = valor da saída do filtro na última execução;
- O_{n-1} valor da saída do controlador na última execução;
- Δt = tempo decorrido até a chegada no novo valor.

$$O_D = K_D * \frac{e_n - e_{n-1}}{\Delta t} \quad (16)$$

Onde:

- O_D = termo da derivada do controlador;
- e_{n-1} = valor do último erro;
- e_n = valor do erro atual;
- Δt = tempo decorrido até a chegada no novo valor.

Toroghi et al. (2009) propõe uma solução que utiliza um controlador fuzzy para sintonizar o controlador PID (Figura 29). O resultado apresentado no estudo é significativamente melhor que o resultado apresentado com o controlador com sintonia fixa.

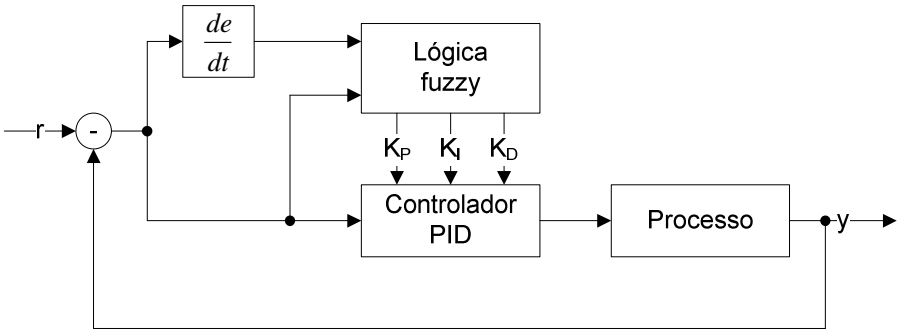


Figura 29 – Controlador PID com sintonia fuzzy

Karasakal et al. (2005) propõe uma arquitetura que utiliza um controlador fuzzy com ganhos ajustados através de um *Estimador de derivada de erro* (Figura 30).

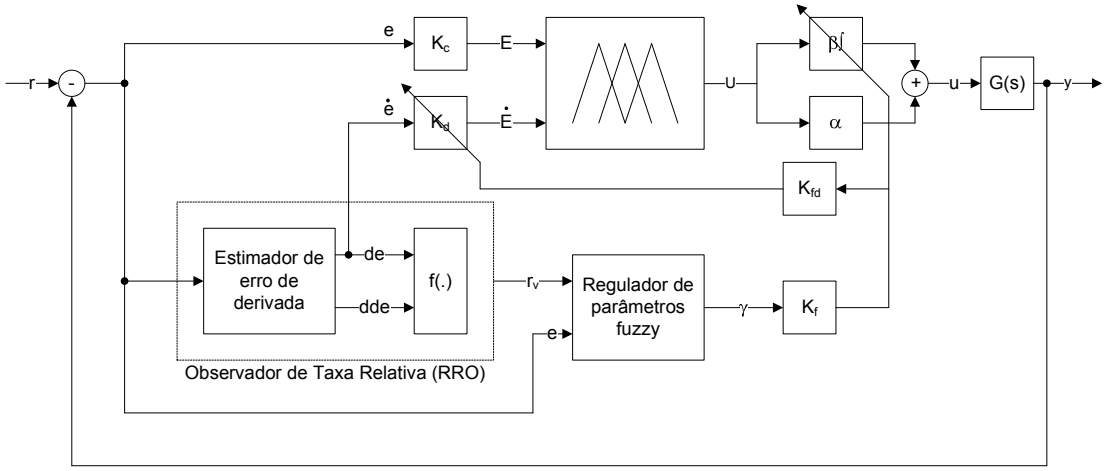


Figura 30 – Controlador adaptativo *fpid* com observador de taxa relativa

Todas as propostas estudadas apresentam excelentes resultados, mas exigem a utilização de dois controladores na malha de controle. A análise abordada neste trabalho estuda o comportamento de um único controlador fuzzy na malha de controle de um NCS e verifica qual é o limite desta utilização.

3.4 Estudo do comportamento do controlador Fuzzy

Inicialmente vamos montar um ambiente padrão para que se possa criar um índice de comparação entre os controladores. Jantzen (1998) propõe uma arquitetura simples e uma metodologia para a migração dos parâmetros PID, calculados pelos mais diversos métodos existentes, para o controle fuzzy. No referido trabalho também são sugeridos alguns controladores fuzzy padrão. A arquitetura proposta por Jantzen (Jantzen, 1998) é apresentada na Figura 31.

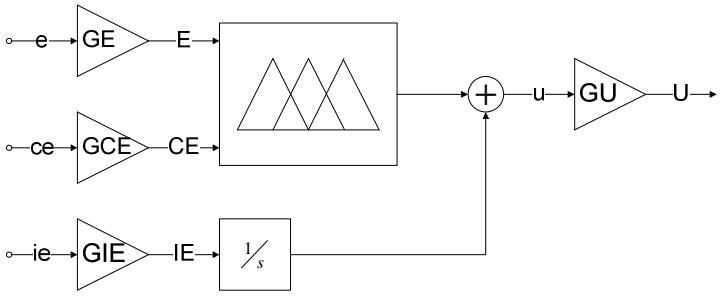


Figura 31 – Controlador fuzzy PD+I

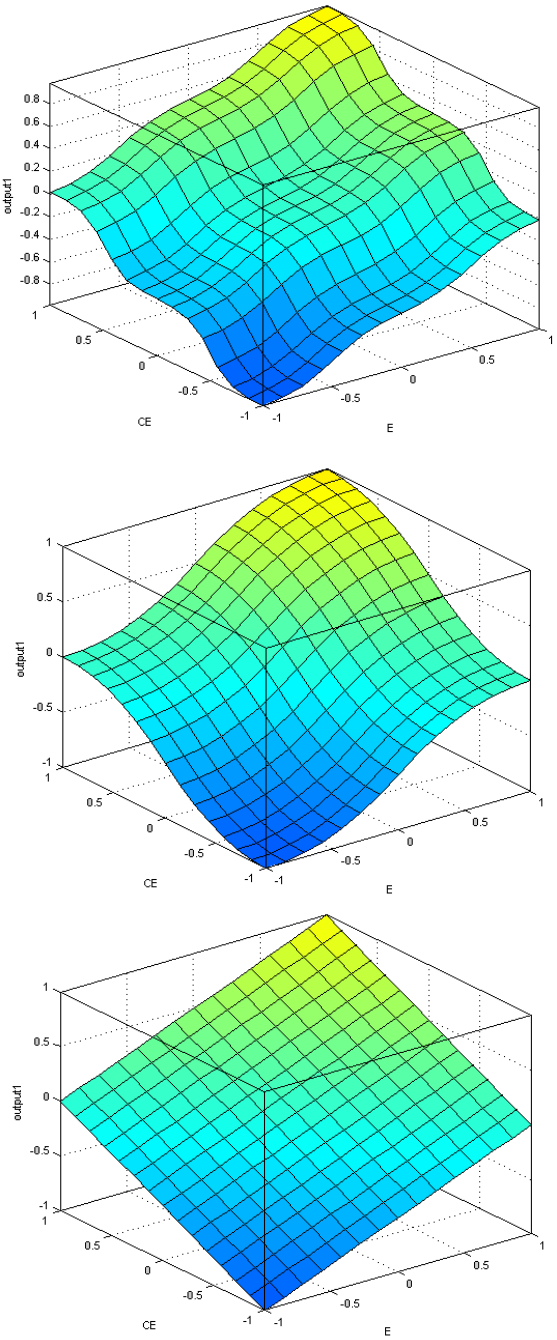
Para a migração de valores, o autor propõe as equações:

$$GE * GU = K_p \tag{17}$$

$$\frac{GCE}{GE} = T_d \tag{18}$$

$$\frac{GIE}{GE} = \frac{1}{T_i} \tag{19}$$

Dos sistemas fuzzy propostos no trabalho do autor vários são similares e foram suprimidos deste estudo. As superfícies dos controladores utilizados são apresentadas na Figura 32.



(a)

(b)

(c)

Figura 32 – Superfície do controlador (a) *bumpy*, (b) *steep* e (c) *linear*

Foram avaliados vários processos, varrendo-se os casos de pólos positivos e negativos, com e sem atraso, além das superfícies acima.

4. IMPLEMENTAÇÃO DE SISTEMAS FUZZY

Para a implementação de um sistema fuzzy (Figura 33) são necessárias, basicamente, três etapas:

- Fuzzificação;
- Inferência e
- Defuzzificação.

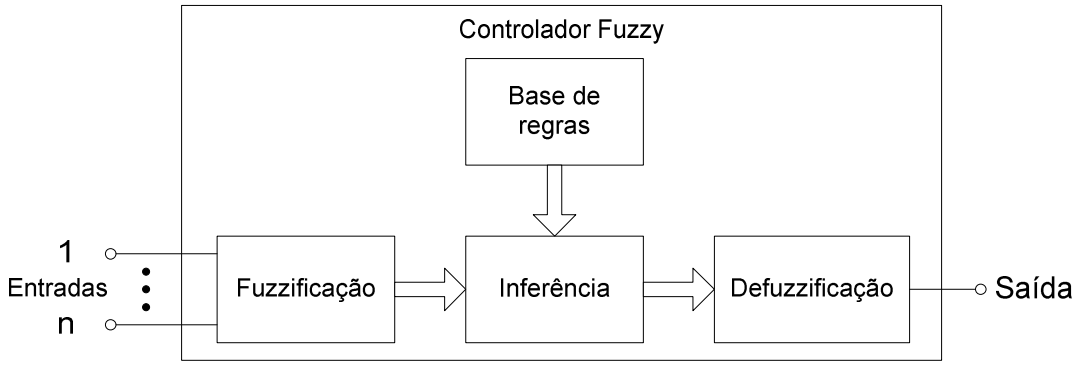
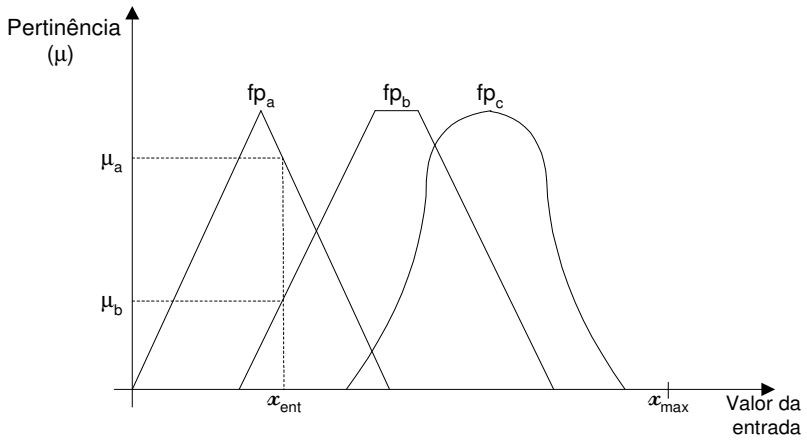


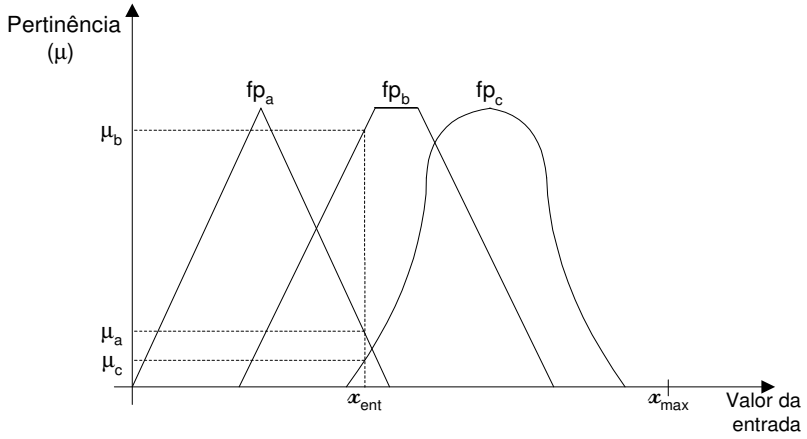
Figura 33 – Controlador Fuzzy de n entradas e uma saída.

Na fuzzificação (Tanscheit, 1999) as entradas não-fuzzy, ou precisas, resultantes de medições ou observações são transformadas em sinais fuzzy, ou seja, o valor do sinal da entrada do controlador é correlacionado a um número que corresponde ao grau de pertinência deste valor a cada condição ou função de pertinência, para posterior ativação das regras.

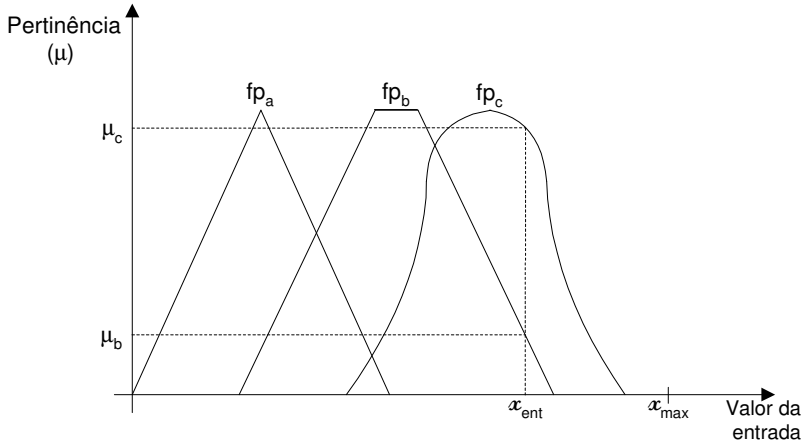
A Figura 34 exemplifica um processo de fuzzificação. Na Figura 34(a) o valor da entrada x_{ent} recebe pertinências μ_a e μ_b, respectivamente. Na Figura 34(b) o valor da entrada x_{ent} recebe pertinências μ_a, μ_b e μ_c, respectivamente. Finalmente, na Figura 34(c) o valor da entrada x_{ent} recebe pertinências μ_b e μ_c, respectivamente.



(a)



(b)



(c)

Figura 34 – Fuzzificação de uma entrada

Na inferência ocorrem as operações com conjuntos fuzzy propriamente ditas (*modus ponens generalizado*). As relações contidas na base de regras do controle são avaliadas (IF precedentes THEN conseqüentes) e o valor de cada função de

saída é calculado através de um OU dos resultados dos respectivos conseqüentes. O resultado, um vetor com os pesos para cada valor de saída, é então calculado.

Na defuzzificação, ocorre a interpretação do processo de inferência. O resultado numérico ou resposta do controlador (não-fuzzy ou precisa) é calculado utilizando-se métodos matemáticos específicos para cada caso. Como exemplo, podemos citar o método dos máximos, do centro de massa, da média ponderada, dentre outros. Abaixo são explicados alguns destes métodos para defuzzificação, normatizados pela IEC 1131-7. Também são apresentadas algumas características de cada método.

Método do Máximo

Neste método (Figura 35), examina-se o conjunto fuzzy da saída e escolhe-se, como valor preciso, o maior valor da faixa da variável de saída (*y*) para o qual o grau de pertinência $\mu(y)$ é máximo. No exemplo da Figura 35(a) o valor preciso da saída é uma faixa $[y_4, y_{max}]$. Apesar da norma definir como valor de saída y_{max}, a escolha do valor preciso ideal é muito difícil. Já no exemplo da Figura 35(b) o valor preciso, y_{max}, é facilmente aceito.

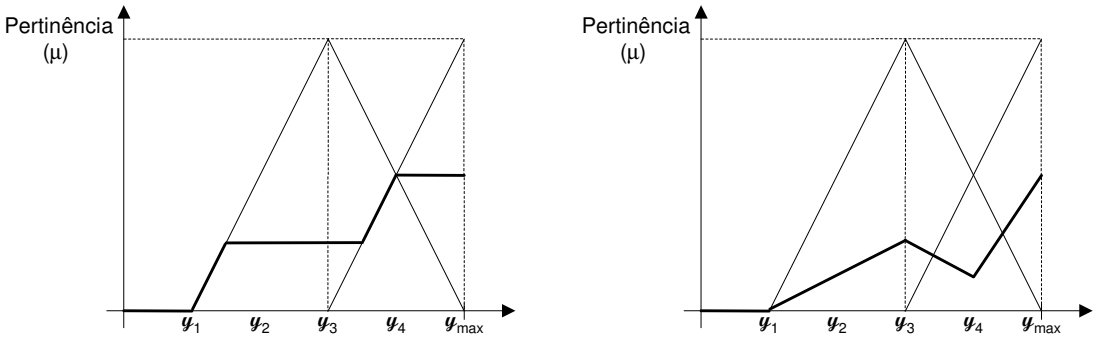


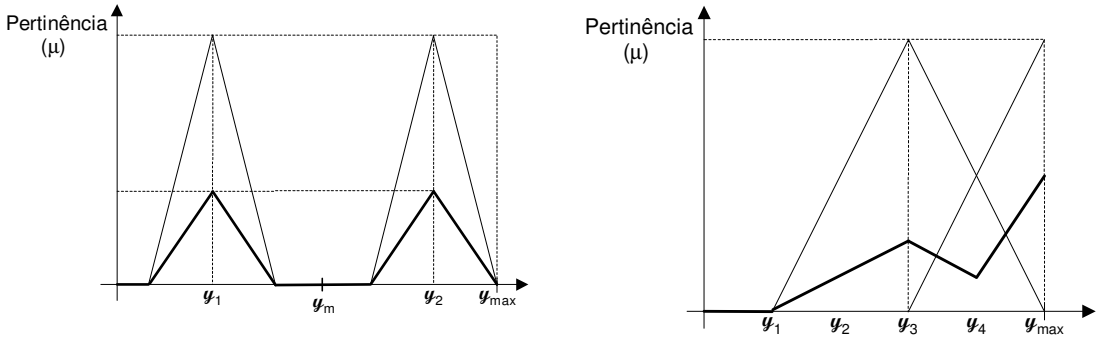
Figura 35 – Defuzzificação - Método do Máximo

Método da Média dos Máximos

Neste método (Figura 36), a saída precisa é obtida tomando-se a média entre os dois elementos extremos da faixa da variável de saída (*y*) que correspondem aos maiores valores da função de pertinência do conjunto de saída $\mu(y)$. No exemplo da Figura 36(a) o valor preciso é dado pela Equação (20). O curioso é que, exatamente

neste ponto, o grau de pertinência é zero. Na Figura 36(b), como só existe um valor máximo, o valor preciso é y_{max}.

$$y_m = \frac{y_1 + y_2}{2} \tag{20}$$



(a) (b)

Figura 36 – Defuzzificação - Método Média dos Máximos

Método do Centróide

Neste método, a saída precisa, y_c (Figura 39), é o valor no universo que corresponde ao centro de gravidade do conjunto fuzzy de saída $\mu(y)$. Este valor é calculado utilizando-se as equações Equação (21), para sistemas contínuos, e Equação (22), para sistemas discretos.

$$y_c = \frac{\int y.\mu_B(y)dy}{\int \mu_B(y)dy} \tag{21}$$

$$y_c = \frac{\sum y_i.\mu_B(y_i)}{\sum \mu_B(y_i)} \tag{22}$$

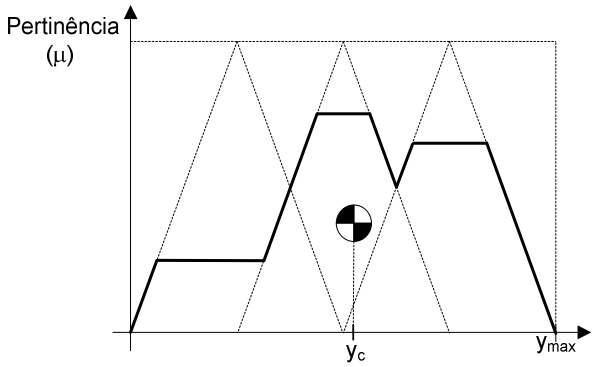


Figura 37 – Defuzzificação - Método do Centróide

Método da Altura

Neste método, a saída precisa, y_h, é a média ponderada dos centróides de cada função de pertinência de saída B^l, associado ao grau de ativação da regra R^l – Equação (23) – Figura 38.

$$y_h = \frac{\sum_l y^l . \mu_{B^l}(y^l)}{\sum_l \mu_{B^l}(y^l)} \tag{23}$$

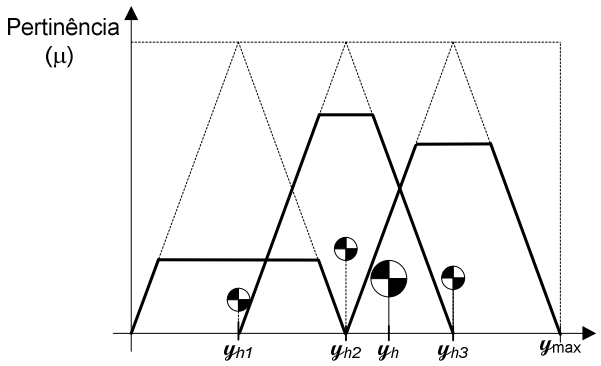


Figura 38 – Defuzzificação - Método da Altura

A vantagem deste método é a simplicidade dos cálculos, pois o valor do centro de gravidade das funções mais comuns é conhecido *a priori*:

- Triangular (simétrica) $\Rightarrow$ ápice do triangulo;
- Gaussiana $\Rightarrow$ valor central de função;
- Trapezoidal (simétrica) $\Rightarrow$ ponto médio do suporte.

O problema deste método é qualquer que seja a largura da função de pertinência, o método oferece o mesmo resultado.

Método da Altura Modificada

Neste método, a saída precisa (y_{mh}) é o valor no universo correspondente a medida da extensão do suporte do conseqüente da regra R^l – Equação (24).

$$y_{mh} = \frac{\sum_l \frac{y^l . \mu_{B^l}(y^l)}{(\delta^l)^2}}{\sum_l \frac{\mu_{B^l}(y^l)}{(\delta^l)^2}} \tag{24}$$

Para funções de pertinência triangulares e trapezoidais δ^{ℓ} é o suporte do conjunto e para funções gaussianas δ^{ℓ} é o desvio padrão.

Etapas na implementação

O primeiro passo na implementação de um controlador fuzzy (Campos et al., 2004) é a definição das variáveis controladas (entradas) e manipuladas (saída). Em seguida, deve-se definir o universo de referência (faixa de trabalho) e o número de valores lingüísticos necessários (número de funções de pertinência). Deve-se dar especial atenção a quantidade de valores lingüísticos, pois um valor baixo pode não controlar o processo e um valor alto pode sobrecarregar o controle.

A aquisição dos conhecimentos necessários à elaboração da base de conhecimentos é a parte mais importante, demorada e crítica durante o desenvolvimento de um controlador fuzzy. Alguns métodos utilizados são:

- Obtenção manual;
- Modelagem do comportamento do operador;
- Modelagem do processo;
- Extração automática do conhecimento (Ex. sistemas neuro-fuzzy).

As implementações de controladores fuzzy podem ser realizadas em software ou em hardware.

4.1 Implementação em software

As implementações em software são desenvolvidas para funcionar com processadores de uso geral e podem utilizar bibliotecas de um programa de desenvolvimento ou utilizar rotinas desenvolvidas pelo próprio usuário. A seguir são apresentados alguns aplicativos comerciais.

Matlab (Mathworks)

Esta implementação, que roda em ambiente Matlab/Simulink, possui uma interface gráfica amigável (GUI - Graphics User Interface) com o usuário (Figura 39). Disponibilizada pelo fabricante através de um *toolbox*, a implementação fornece um ambiente virtual para simulação e possui capacidade de controle a nível laboratorial utilizando hardware específico do fabricante (xPC – Figura 40).

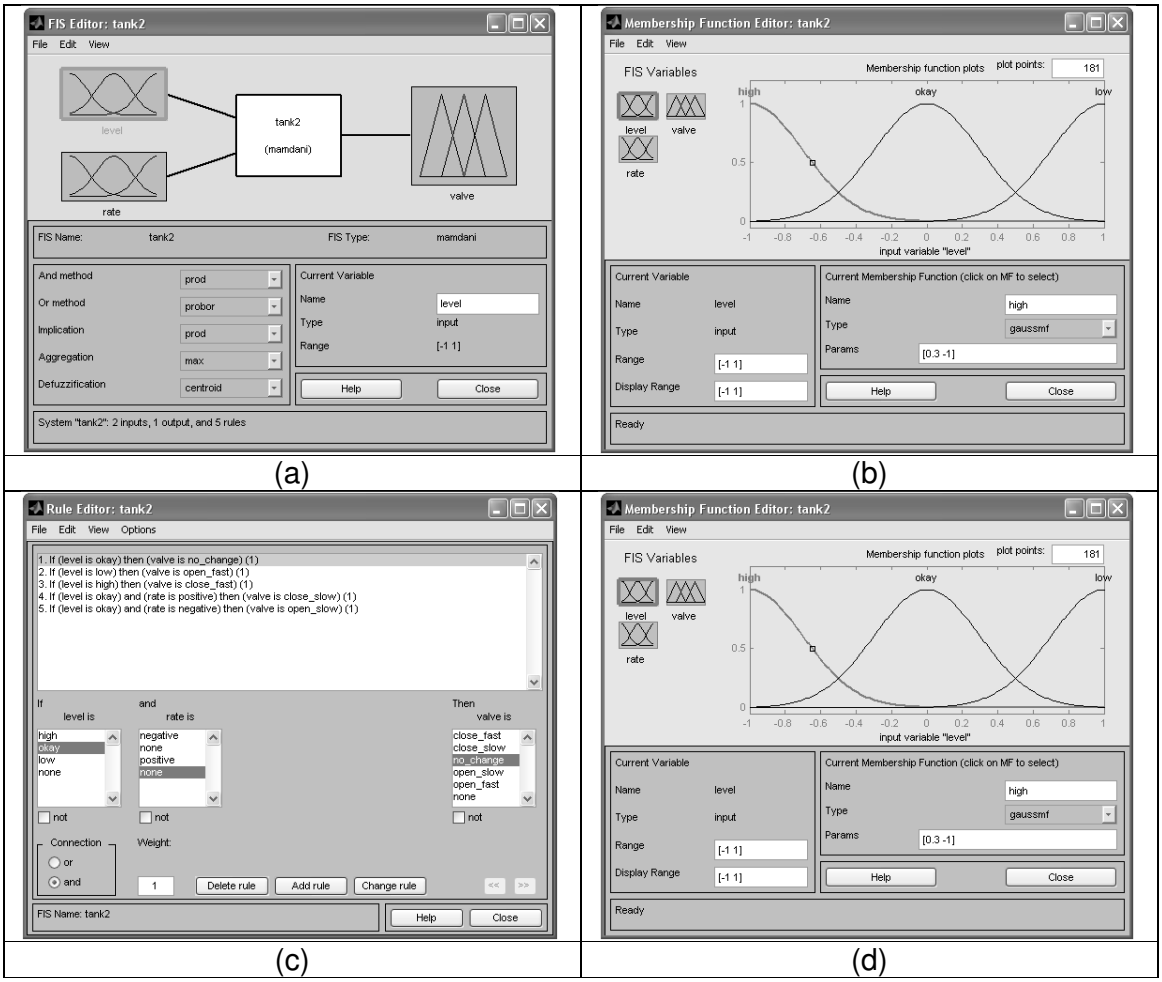


Figura 39 – Aplicativo *Fuzzy* do Matlab para implementação de um controlador fuzzy.
(a) Diagrama em blocos, (b) Fuzzificação, (c) Inferência e (d) Defuzzificação

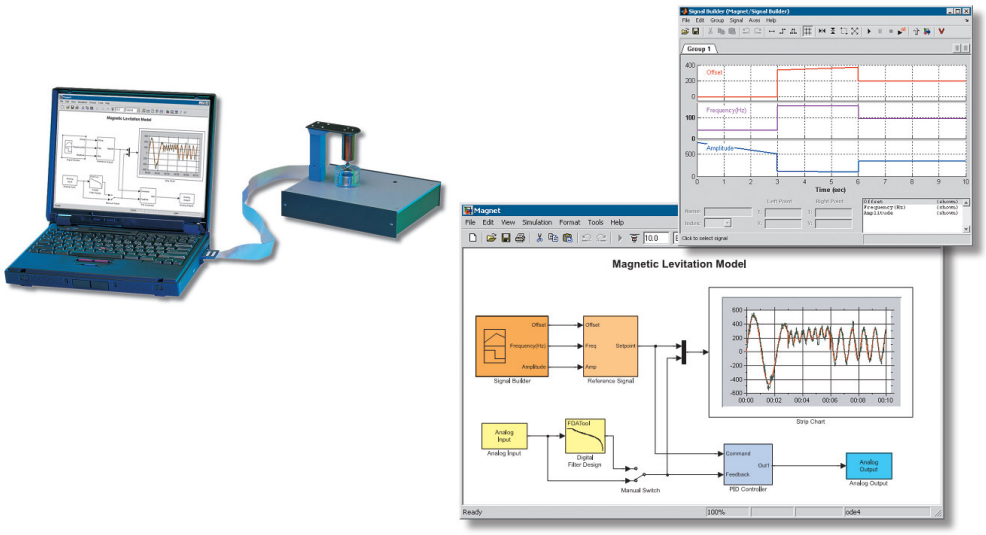


Figura 40 – Interface xPC da Mathworks para uso com Matlab/Simulink.

FuzzyTECH

Desenvolvido pala Empresa Alemã fuzzyTECH, o programa, consonante com a norma IEC 1131-7, oferece suporte para diversos aplicativos comerciais, tais como, InTouch™, InControl™, CiTECT™, LabVIEW™, Matlab/Simulink™, WinFACT™, WinCC™, FIX™, BridgeVIEW™, dentre outros (Figura 41). Também é possível, segundo o site do fabricante, adquirir o pacote *neuro-fuzzy*.

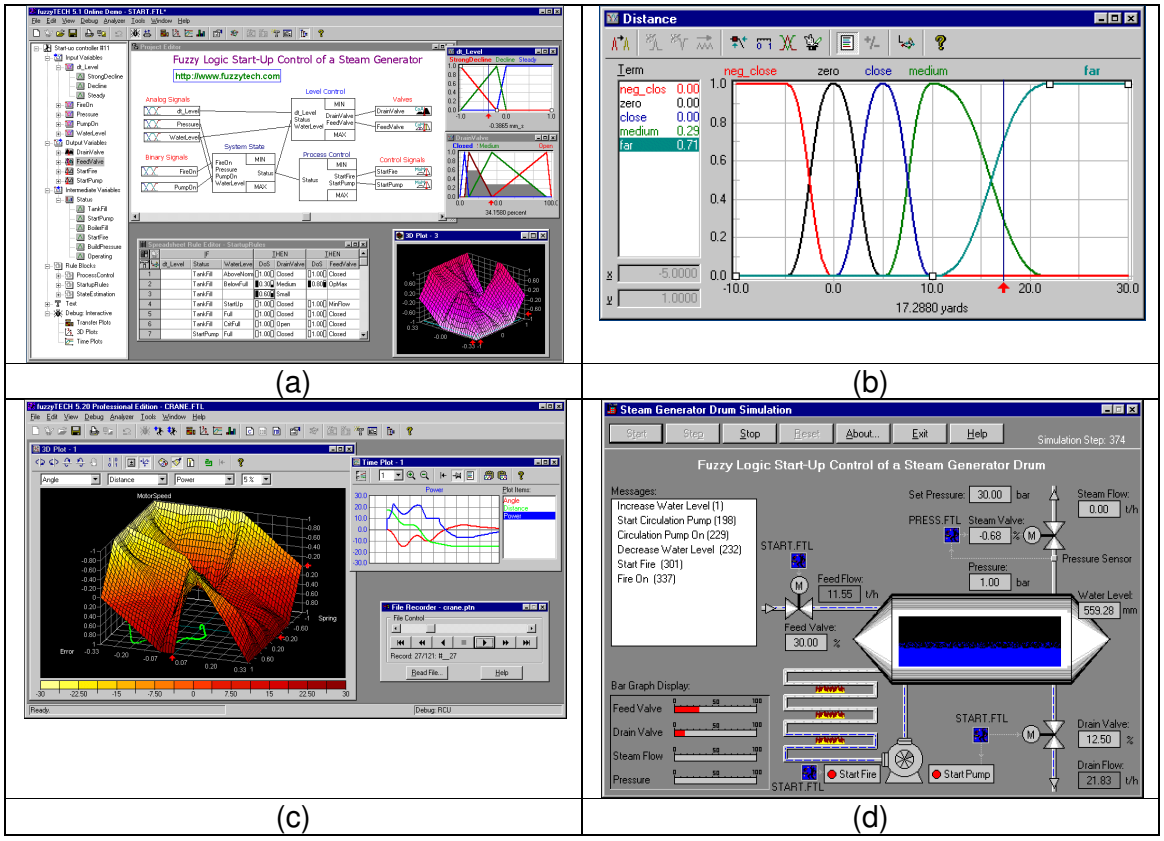


Figura 41 – Aplicativo fuzzyTECH para implementação de um controlador fuzzy.
Módulo de edição: (a) do sistema e (b) das funções de pertinência
Módulo de análise: (c) do sistema e (d) Simulação

Labview (National Instruments)

Utilizando a interface gráfica G (Figura 42), padrão do *LabView*, o pacote *Fuzzy* oferecido pela *National Instruments*, permite ao projetista o desenvolvimento de um controlador fuzzy tanto virtual quanto real. Através de hardware específico do fabricante (módulos DAQ) ou de hardware de terceiros, cuja interface com o programa é feita via comunicação específica (RS232, GPIB, USB, etc.), o aplicativo

permite o controle de processos industriais tanto via plataforma PC quanto via hardware específico do fabricante.

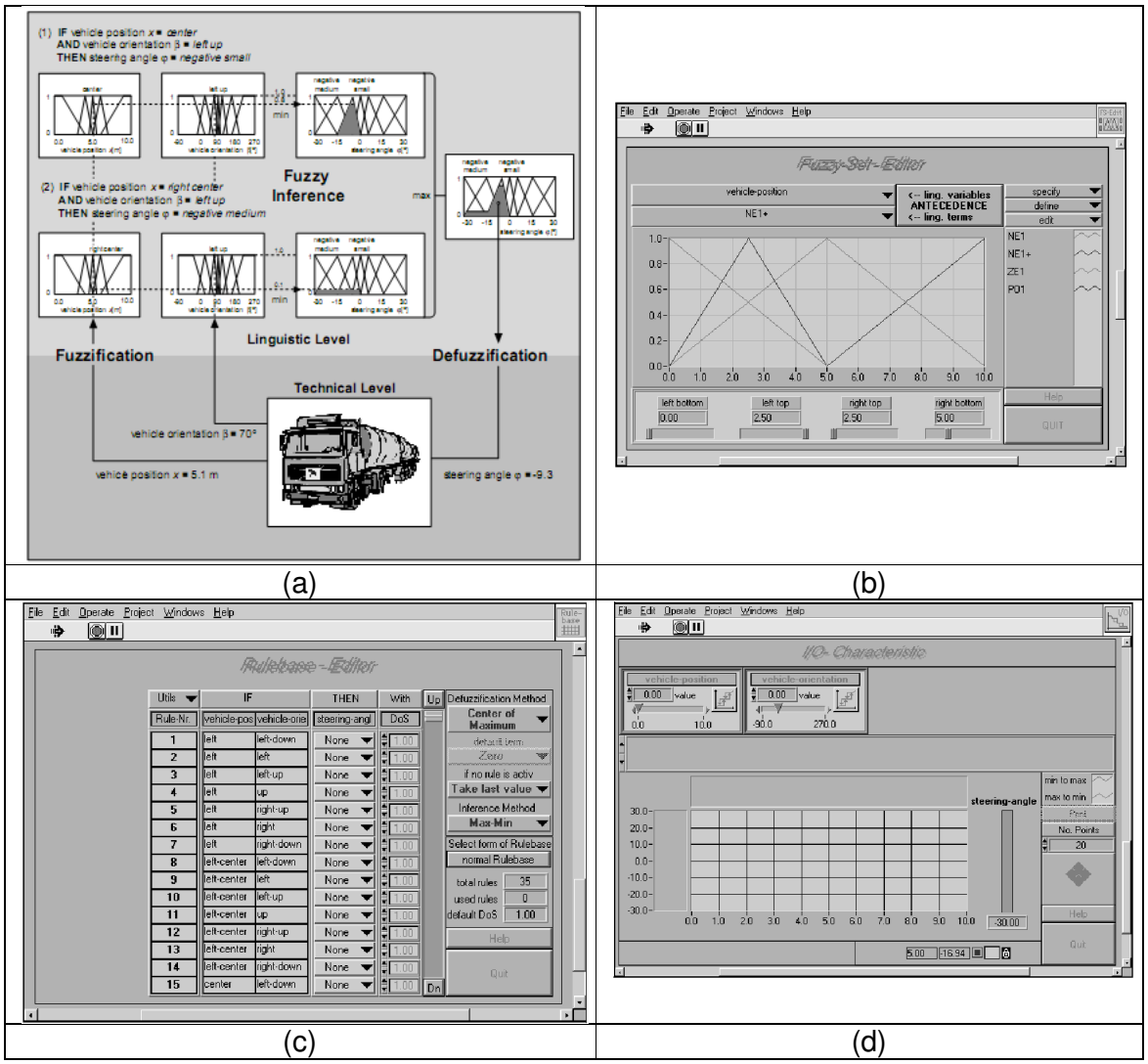


Figura 42 – Ferramenta de programação gráfica da National Instruments para implementação de um controlador Fuzzy.
(a) Diagrama em blocos, (b) Fuzzificação, (c) Inferência e (d) Defuzzificação

As soluções por software, além de proporcionar flexibilidade para definir a base do conhecimento fuzzy, também tornam possível a seleção dos operadores fuzzy e a escolha dos algoritmos de inferência. Entretanto, estas soluções se tornam inadequados para problemas que demandam alta velocidade de inferência. Neste caso, soluções de hardware devem ser adotadas.

Existem também algumas soluções fuzzy desenvolvidas para rodar em hardware de uso geral, como os micros controladores PIC, e que podem ser vistos

em diversos bens de consumo, tais como, máquinas de fazer arroz (muito comum no Japão), máquinas fotografias (para evitar foto tremida), etc

4.2 Implementação em hardware

As implementações em hardware, por sua vez, utilizam circuitos eletrônicos para realizar as funções existentes no controlador fuzzy e podem utilizar técnicas analógicas ou digitais.

4.2.1 Hardware analógico

Durante o projeto de um controlador fuzzy utilizando hardware analógico pode-se lançar mão de duas abordagens. A primeira, conhecida como granularidade grossa, utiliza blocos com funções já pré-definidas, concentrando, basicamente, o foco do projeto na parametrização e a interligação destes blocos – FPAAs (Amaral, 2003). Em (Embabi et al., 1998) é apresentada uma implementação de um controlador fuzzy utilizando FPAA, no qual são utilizadas células de fonte de corrente.

Numa segunda abordagem, faz-se uso de componentes eletrônicos discretos, tais como transistores, diodos, etc. para a realização das funções do controlador. (Castillo et al., 1997) apresentam alguns circuitos analógicos que foram utilizados para a implementação das principais funções que podem ser empregadas em um controlador fuzzy (Figura 44).

4.2.2 Hardware digital

Assim como em outras áreas de aplicação, a implementação de um sistema fuzzy conduz a uma discussão sobre o tipo de implementação: implementação em arquitetura de uso geral *versus* sistemas dedicados (Costa et. al., 1995). Várias propostas já foram desenvolvidas em cima deste tema, sem que nenhuma fosse aceita como definitiva. A estrutura relativamente simples do controlador fuzzy favorece uma implementação em hardware dedicado. O problema está na decisão do quão flexível a implementação do controlador fuzzy deve ser (em termos de alteração de funções de pertinência, métodos de inferência e defuzzificação, precisão e formato das regras, dentre outros). Obviamente, uma solução que aborde todas as possíveis aplicações não existe e cada implementação possui algumas soluções mais apropriadas. Os fatores que mais influenciam na escolha são:

- Velocidade;
- Complexidade;
- Restrições de tempo real;
- Interação entre os sistemas fuzzy e não-fuzzy.

Podemos identificar quatro classes principais, entre as diferentes alternativas de implementação de sistemas fuzzy (Figura 43):

- Soluções de software com componentes de hardware de uso geral;
- Processadores de uso geral com instruções especiais para cálculos;
- Coprocessadores fuzzy dedicados;
- ASIC's fuzzy com capacidade de operações stand-alone.

Soluções de software com componentes de hardware de uso geral

Implementações em software de algoritmos *fuzzy* em microcontroladores são hoje as técnicas mais largamente utilizadas. As seguintes vantagens e desvantagens que esta abordagem possui são:

- Flexibilidade completa;
- Suporte a processamento não fuzzy;
- Disponibilidade de sistemas de desenvolvimento;
- Disponibilidade de ferramentas de sistemas de suporte;
- Baixa velocidade.

Uma abordagem alternativa que requer apenas componentes padrões é representada por mapeamento de memória ou *look-up tables* (Garcia, 2009). Nesta abordagem, os valores de saída são pré-computados para muitos valores de entrada e armazenados em RAM. Neste caso a velocidade de inferência é muito alta, mas há crescimento exponencial da memória requerida se o número de variáveis de entrada e saída, ou a resolução, incrementam. Normalmente esta técnica é limitada para aplicações com 2 a 3 entradas e 1 a 2 saídas, com baixa resolução.

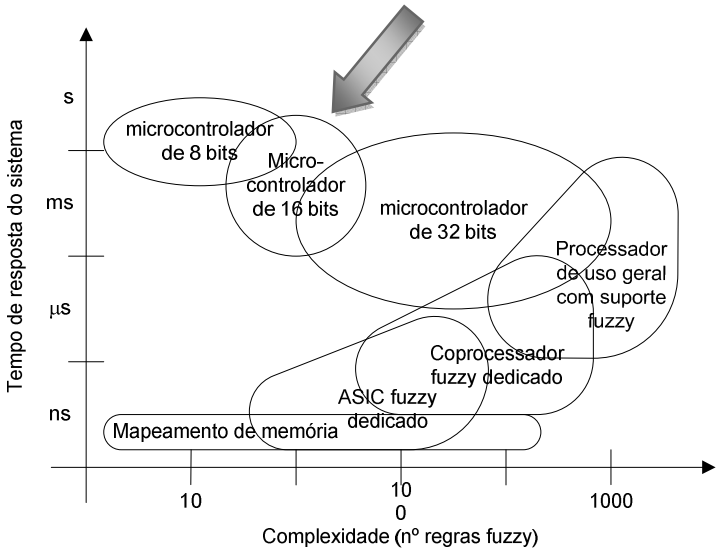


Figura 43 – Figura comparativa dos tipos de implemtenação fuzzy.

Processadores de uso geral com instruções especiais para cálculos

Uma abordagem alternativa, que ainda mantém capacidade de computação de uso geral, emprega processadores de uso geral com a adição de pequenas instruções especializadas (ASIC) para acelerar tarefas *fuzzy*. As vantagens e desvantagens desta abordagem são:

- Flexibilidade completa;
- Alta velocidade comparada aos microcontroladores padrões;
- Simples extensão do núcleo existente;
- Suporte automático de computação não fuzzy;
- Alto custo comparado aos microcontroladores padrões devido ao pequeno volume de produção;
- Desempenho limitado a algumas aplicações.

Podem-se distinguir duas técnicas alternativas que correspondem às diferentes filosofias de projeto do núcleo do microcontrolador:

- Para conjunto de instruções complexas (arquitetura CISC) é possível adicionar firmware dedicado ao suporte de lógica fuzzy, modificando, apropriadamente, o microprograma do controlador.
- Para processadores de instruções reduzidas (arquitetura RISC), uma possibilidade interessante é somar poucas instruções dedicadas para um

conjunto de instruções de uso geral que permitem ao compilador otimizar aplicações fuzzy.

Coprocessadores fuzzy dedicados

Muitos processadores *fuzzy* dedicados são desta classe. Existem processadores dedicados a computação fuzzy que não podem implementar sozinhos um sistema de controle devido a falta de capacidade computacional de uso geral, mas ainda permitem alguma flexibilidade e configurabilidade. São as seguintes vantagens e desvantagens desta abordagem:

- Alta velocidade;
- Modularidade;
- Alguma flexibilidade;
- Regras e faixa de funções de pertinência limitadas;
- Alto custo;
- Necessidade de processador de uso geral para suportar partes não fuzzy.

Estes coprocessadores dedicados normalmente adotam funções de pertinência triangulares, formato de regras flexíveis (número de antecedentes e conseqüentes), método de inferência max-min, 8 bits de resolução, e método de desfuzzificação centróide. A Tabela 2 apresenta dois desses coprocessadores comerciais.

Tabela 2 – Exemplos de coprocessadores fuzzy comerciais

Fabricante	Modelo	Características	Observações
Siemens	81C99	256 entradas 64 saídas 64 módulos (com 256 regras)	Qualquer tipo de função de pertinência
Omron	PFX000	8 entradas 4 saídas 128 regras 12 bits	Formato da regra é fixa (8 antecedentes e 2 conse-qüentes)

O Processador de Inferência Fuzzy SAE 81C99 possui, além das características apresentadas na tabela, a capacidade de utilização *stand alone* ou em conjunto com processadores de 8 e 16 bits. Utilizando uma freqüência de *clock*

de 20 MHz, atinge a velocidade máxima de inferência de 10 milhões de regras por segundo.

ASIC's fuzzy com capacidade de operações *stand-alone*

A última solução para aumentar o desempenho em aplicações fuzzy é o desenvolvimento de uma arquitetura dedicada. Devido a natureza particular dos processos de inferência fuzzy, é possível implementar diretamente um algoritmo fuzzy com estrutura de hardware dedicado de modo a maximizar a velocidade e minimizar a área de silício, usando técnicas de alto nível. O resultado da solução é, obviamente, para uma aplicação simples e os parâmetros de configuração são reduzidos a um conjunto mínimo, o que não compromete a área e a velocidade. São as seguintes as vantagens e desvantagens desta abordagem:

- Processamento muito rápido, direcionado a aplicação;
- Projeto rápido (síntese automática);
- Baixo custo em termos de área de silício (5 a 10 mil portas);
- Disponível como um bloco em um sistema de controle de chip único;
- Implementável em FPGA para prototipação;
- Baixo custo para alto volume de produção;
- Sem flexibilidade (a aplicação é fixa);
- Complexidade limitada;
- Apenas para sistemas baseados em ASIC;
- Pode necessitar de microprocessador para suportar partes não fuzzy.

Na implementação em hardware digital, não é comum ver a utilização de dispositivos discretos (portas AND, OR, XOR, etc.). Neste tipo de implementação é mais usual a utilização de blocos como conversores A/D, somadores, multiplicadores, etc., possibilitando ao projetista o uso de dispositivos como FPGAs, microprocessadores e microcontroladores (com os respectivos programas).

(Mesquita et al., 2006) apresentam uma implementação de um controlador fuzzy baseado em FPGA. Nesta implementação os autores inicialmente apresentam uma arquitetura que tanto pode ser implementada em hardware, quanto em software ou, ainda, de forma híbrida. Baseada nesta arquitetura os autores desenvolvem a aplicação em FPGA.

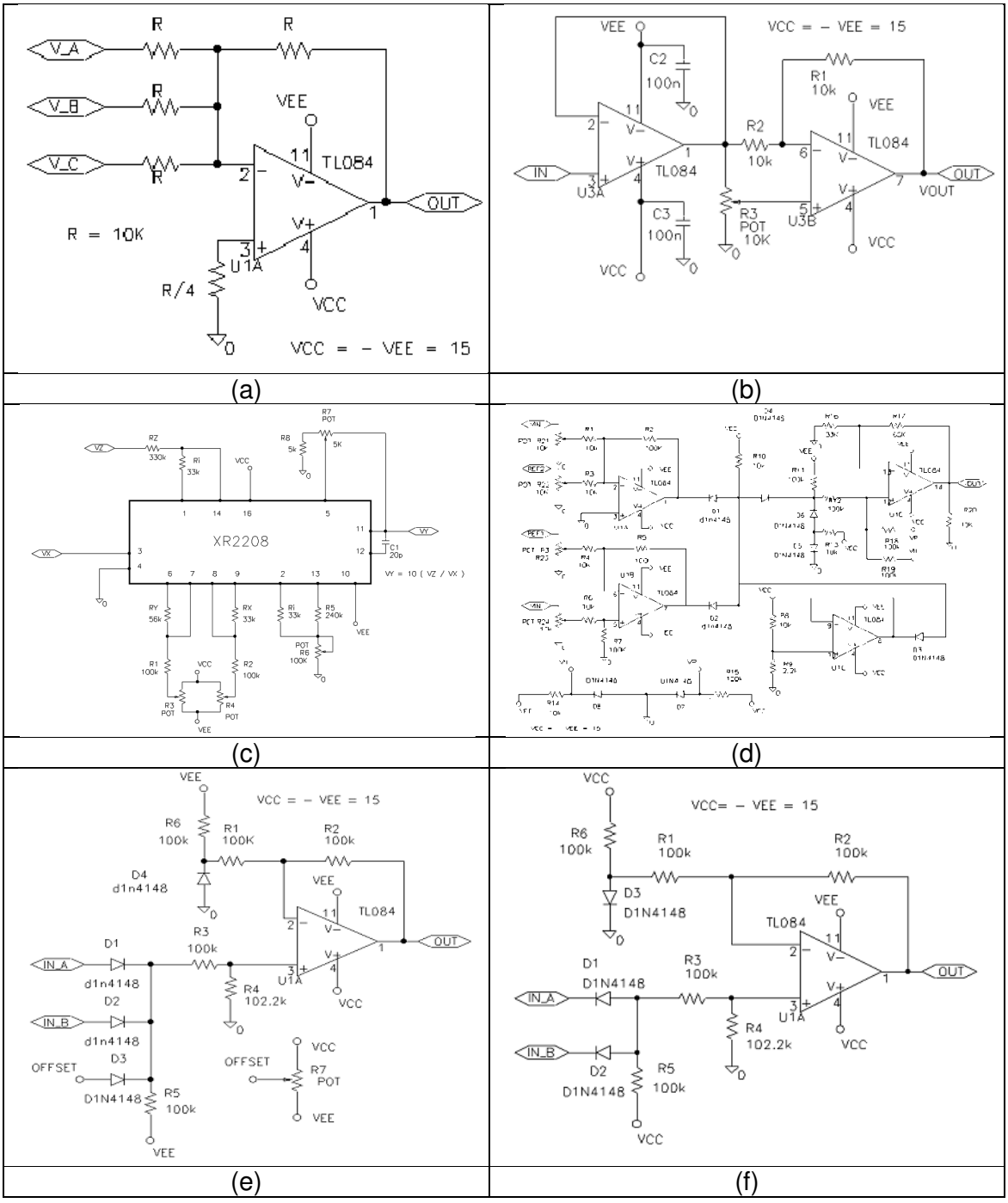


Figura 44 – Implementação em hardware
(a) Somador, (b) Multiplicador por constante, (c) Divisor, (d) Fuzzificação, (e) Função máximo e (f) Função mínimo (Castillo et al., 1997)

Como as arquiteturas de implementação com o uso de microprocessadores e microcontroladores são as mesmas. No decorrer deste capítulo, só será utilizado o termo microprocessador para descrever ambas as implementações.

5. CONTROLADOR FUZZY COM ENTRADA E SAÍDA REMOTA

5.1 Controlador fuzzy

Para a implementação de um controlador baseado em arquitetura microprocessada, são necessárias, além do algoritmo de controle propriamente dito, rotinas para:

- Condicionamento do sinal de entrada - para compatibilizar os níveis de tensão e/ou corrente do(s) sinal(is) de entrada com o nível de tensão do(s) conversor(es) A/D;
- Conversão A/D;
- Conversão D/A;
- Condicionamento do sinal de saída - para compatibilizar os níveis de tensão e/ou corrente do(s) sinal(is) do conversor D/A com o nível de tensão da(s) saídas;
- Rotinas de comunicação (serial, USB, TCP/IP, ...);
- Rotinas diversas (conversão hex → ascii e ascii → hex, envio de dados para display, leitura de teclado, ...).

Visando a flexibilização do controlador fuzzy, tornando-o capaz de implementar uma vasta gama de controladores fuzzy, as informações necessárias, tais como: número de entradas, número de funções de pertinência e número de saídas, etc utilizam uma padronização.

5.2 Projeto do controlador fuzzy

Tendo em vista que o foco deste trabalho é o estudo da implementação de um sistema fuzzy utilizando microcontroladores de uso geral, optou-se por um controlador com a seguinte arquitetura:

- Número de entradas configurável (programável);
- Uma saída;
- Cada entrada possui seu próprio número de funções de pertinência;
- Cada função de pertinência possui sua própria quantidade de pontos de interpolação.

Com o intuito de verificar o comportamento de um controlador de baixo custo, a estratégia utilizada neste trabalho optou pela seguinte abordagem:

- O cálculo do grau de pertinência utiliza aproximação por segmentos de retas;
- As funções de saída são *singletons*;
- Nos cálculos do controlador foram utilizados números inteiros.

Aproximação por segmentos de reta

Esta técnica é baseada na aproximação da(s) curva(s) da(s) função(ões) de pertinência por vários segmentos de retas. Para isso, é necessário que se escolha tantos pontos quantos forem necessários para a melhor representação da função (Figura 45).

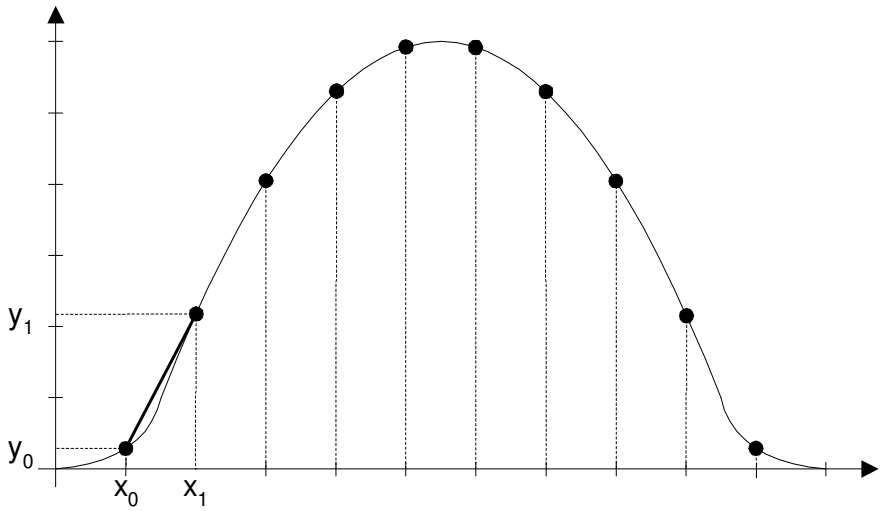


Figura 45 – Seleção dos pontos para segmentação da curva

Os valores das ordenadas e das abscissas obtidas na curva são utilizadas para compor uma tabela, conforme apresentado na Tabela 3. Cabe ressaltar que, para se obter uma maior precisão neste método, deve-se concentrar uma quantidade maior de pontos nas regiões da curva que possui variação significativa da derivada $\left(\frac{d^2y}{dt^2}\right)$.

Tabela 3 – Valores dos pontos obtidos na função de pertinência.

X_{in}	$\sigma(X_{in})$
X_0	Y_0
X_1	Y_1
⋮	⋮
X_n	Y_n

Quando existe a necessidade do cálculo do valor da pertinência de um valor da entrada, busca-se, na tabela, o intervalo ao qual o ponto pertence. Uma vez localizado o intervalo, por exemplo, $x_{ent} \in [x_i, x_{i+1}]$, calcula-se os valores dos coeficientes (m e h) da equação da reta do intervalo – Equação (25).

$$\begin{aligned} y &= m.x + h \\ y_i &= m.x_i + h \quad (a) \\ y_{i+1} &= m.x_{i+1} + h \quad (b) \end{aligned} \xrightarrow{(b)-(a)} \begin{aligned} y_{i+1} - y_i &= m.(x_{i+1} - x_i) \\ \boldsymbol{m} &= \frac{\boldsymbol{y_{i+1} - y_i}}{\boldsymbol{(x_{i+1} - x_i)}} \\ \boldsymbol{h} &= \boldsymbol{y_i - m.x_i} \end{aligned} \qquad (25)$$

Uma vez calculados os coeficientes da equação, o valor da entrada é substituído nesta equação e o valor da pertinência é obtido.

Tendo em vista o fato de o programa trabalhar somente com números inteiros, o valor de *m*, o qual representa o valor da tangente do ângulo que a reta fez com eixo *x*, possui passo unitário. Assim, os valores de *m* possuem a seqüência: 0, 1,..., n, significando que o ângulo da reta muda abruptamente de 0° para 45°, introduzindo um erro significativo quando o valor real de *m* encontra-se no intervalo [0,1]. Para reduzir este efeito, foi utilizado o recurso matemático de multiplicar ambos os lados da equação por um fator *k*, o que permitiu reduzir o passo inicial de *m* para 1/k. Na Tabela 4 são apresentados alguns valores do comportamento do ângulo $\theta = tg^{-1} m$ para k = 1 e 10.

Tabela 4 – Comportamento do ângulo da reta com o eixo x em função do fator multiplicador k

k = 1	
m	$\theta = tg^{-1}m$
0	0°
1	45°
2	63°
...	...

k = 10	
m	$\theta = tg^{-1}m$
0,0	0,0°
0,1	5,7°
0,2	11,3°
...	...

Verificou-se também que devido ao fato de estar-se trabalhando com números inteiros, os valores do coeficiente *h* apresentavam uma diferença significativa dependendo dos valores que eram utilizados para o seu cálculo (Figura 46). Para se contornar este problema, foram calculados dois valores de h (calculados a partir dos extremos do intervalo). No cálculo da pertinência foi utilizado o valor médio

$$h_m = \frac{h_1 + h_2}{2}$$

A equação final ficou com o seguinte aspecto:

$$y = \frac{m_k * x}{k} + h_m$$, onde $m_k = k * m$

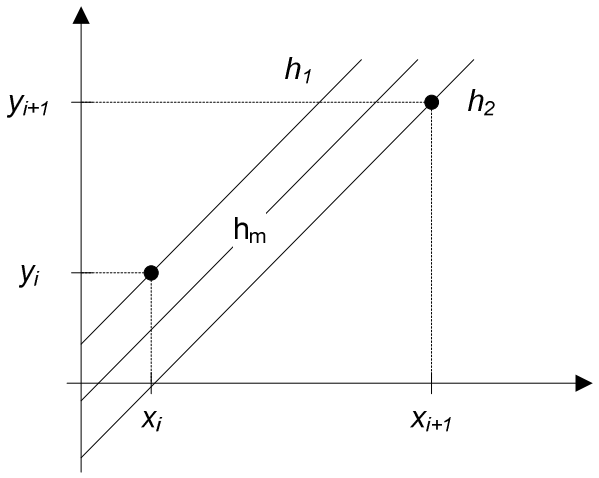


Figura 46 – Erro no cálculo da constante ***h*** devido às operações com números inteiros.

Além disso, também é necessária a tabulação das regras de inferência, da quantidade de *singletons* e respectivos valores. Estes pontos formarão a(s) tabela(s) que são gravadas na memória não-volátil do controlador.

Rotina de Fuzzificação

Na rotina de fuzzificação, o valor de entrada é correlacionado ao valor de pertinência para cada função de entrada.

Este procedimento é repetido até que todos os valores de pertinência de todas as entradas tenham sido calculados. Estes valores são armazenados numa tabela na memória do controlador (Tabela 5). Esta tabela é construída com os valores das pertinências calculadas pelo controlador, configurado para n entradas, sendo a primeira com r funções de pertinência, a segunda com s funções de pertinência e assim sucessivamente até a enésima entrada com t funções de pertinência.

Tabela 5 – Tabela para mapeamento das funções de pertinência do controlador.

Entrada	Valor da Pertinência (μ)			
$Valor_{E1}$	$\mu_{E1.1}$	$\mu_{E1.2}$	...	$\mu_{E1.r}$
$Valor_{E2}$	$\mu_{E2.1}$	$\mu_{E2.2}$	...	$\mu_{E2.s}$
⋮	⋮	⋮	...	⋮
$Valor_{En}$	$\mu_{En.1}$	$\mu_{En.2}$	...	$\mu_{En.t}$

Rotina de inferência

Na rotina de inferência, o controlador, a partir das regras gravadas numa tabela na memória, monta o vetor com os pesos para cada *singleton* da saída. Na Tabela 6 estão resumidas as equivalências das operações que foram utilizadas para cada sentença.

Tabela 6 – Correlação entre os operadores fuzzy e/ou e as operações min/max do controlador.

Relação fuzzy	Equivalente no programa do controlador
a e b	min(a,b)
a ou b	max(a,b)

Assim, a sentença fuzzy:

SE Entrada1 é FP_n E Entrada2 é FP_m, ENTÃO Saída$_k$

Ficará com o seguinte equivalente no programa:

Nova Saída$_k$ = Max[Antiga Saída$_k$, min($\mu_{E1.n}$, $\mu_{E2.m}$)]

E a sentença fuzzy:

SE Entrada1 é FP_n OU Entrada2 é FP_m, ENTÃO Saída$_k$

Ficará no programa:

Nova Saída$_k$ = Max[Antiga Saída$_k$, max($\mu_{E1.n}$, $\mu_{E2.m}$)]

Rotina de defuzzyficação

Para o cálculo do valor de saída (resposta do controlador) foi adotada a média ponderada dos valores dos *singletons* (Figura 47), cujos pesos são os valores das pertinências calculadas na rotina de inferência ((26)

$$Saída = \frac{\mu_1 * S_1 + \mu_2 * S_2 + ... + \mu_n * S_n}{\mu_1 + \mu_2 ... + \mu_n} \tag{26}$$

A fim de evitar erro de divisão, o programa, antes de realizá-la, verifica se o valor do denominador é zero, caracterizando que o ponto possui grau de pertinência zero em todas as funções de pertinência, e, em caso positivo, atribui o valor 0 na saída.

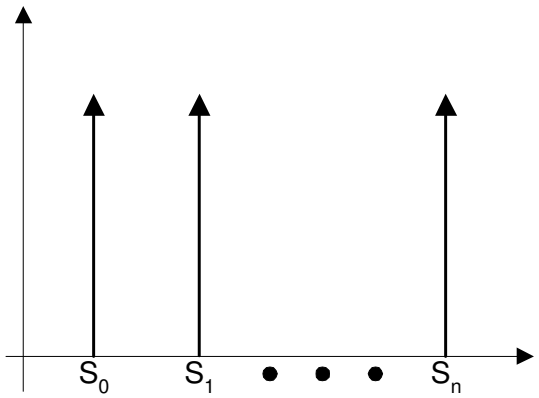


Figura 47 – *Singletons* da saída

5.2.1 Diagrama funcional

Esta parte do trabalho foi dividida em duas etapas. A primeira foi a implementação de um controlador fuzzy com A/D local para o teste do controlador

(Figura 48a) e a segunda, foi a implementação de um controlador com A/D remoto (Figura 48b), no qual foi utilizado o conversor A/D do módulo sem fio.

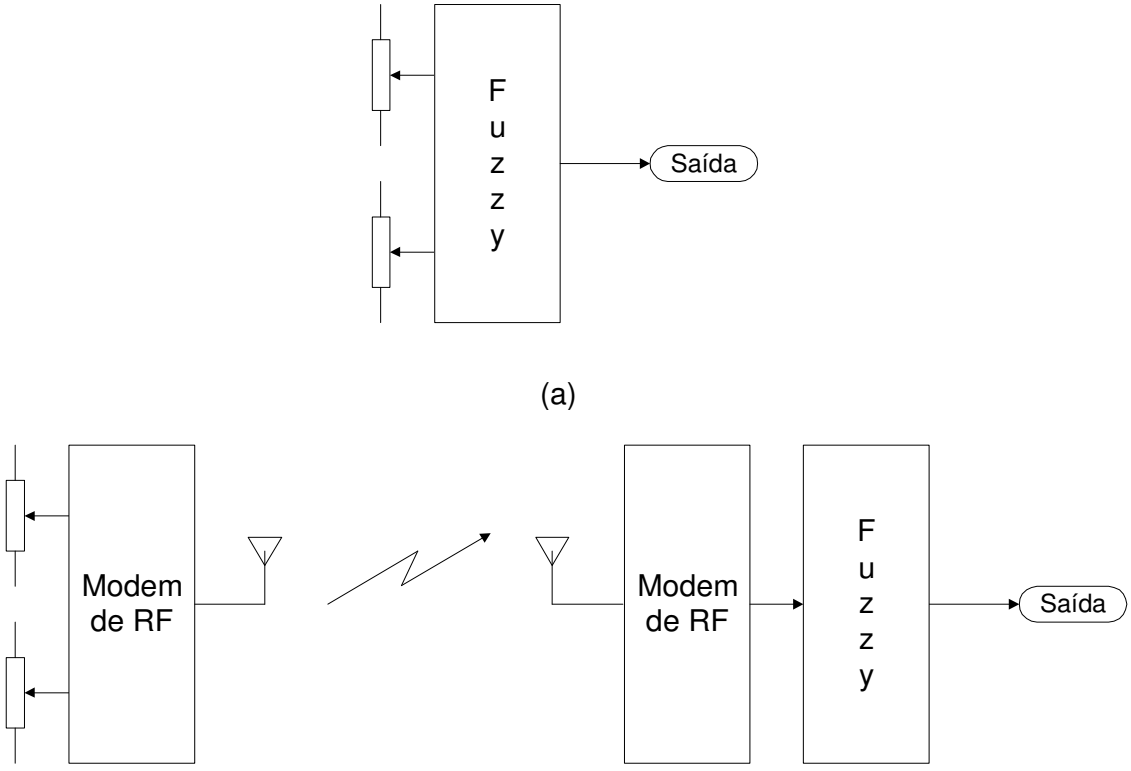


Figura 48 – Implementação de um controlador fuzzy com (a) A/D local e (b) A/D remoto

5.2.2 Diagrama de blocos

Para a implementação deste trabalho, foi escolhido um microcontrolador dsPIC de 16 bits devido a facilidade de programação, farta bibliotecas de rotinas, velocidade de processamento e conversor A/D incorporado. Para facilitar a implementação do controlador, foi priorizada a utilização de kits de desenvolvimento, pois os kits já possuem diversas facilidades incorporadas, como display de LCD (para a apresentação de mensagens e/ou valores), circuitos e conectorização para as diversas interfaces (serial, PS2, CAN, relés, etc.). A Figura 49 e a Figura 50 representam as duas arquiteturas das etapas de implementação que foram utilizadas neste trabalho, já com a concepção de kit de desenvolvimento.

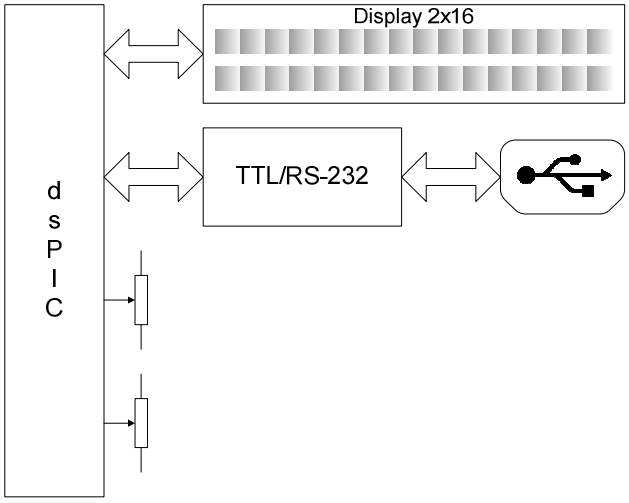


Figura 49 – Implementação de um controlador fuzzy com A/D local

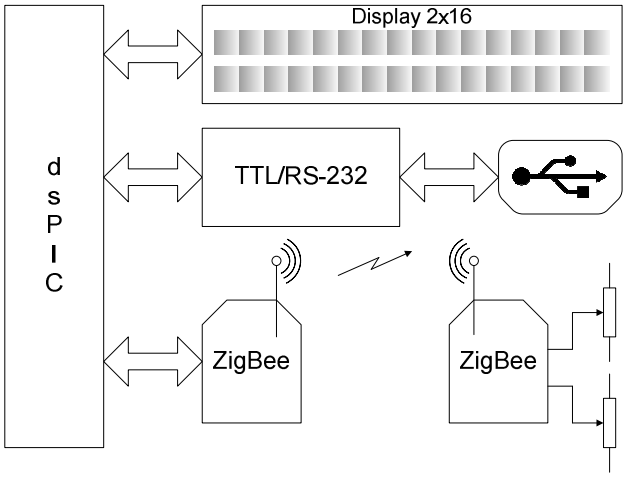


Figura 50 – Implementação de um controlador fuzzy com A/D remoto

5.3 Concepção de E/S remota

Um sistema com E/S remota é todo aquele em que as entradas e/ou saídas encontram-se distantes da CPU do CLP. As principais características são (Pereira et al., 1998):

- A/D e D/A remotos (no próprio dispositivo);
- Leitura e escrita dos dados via rede de comunicação (proprietárias ou abertas);
- Tipos de atualização:
  - Pooling (varredura dos sensores a cada *t* seg.);

- Timing (um dado a cada *t* seg.);
- Evento (mudança no estado lógico – Low → High ou High → Low);
- Variação (variação do sinal de entrada superior a um valor prédeterminado);

o Compartilhamento do canal com outros processos.

5.4 Apresentação da proposta de E/S remota

Para a escolha dos dispositivos que seriam utilizados neste trabalho, foram observados, inicialmente, quais os módulos comercializados no mercado nacional e, posteriormente, se as características destes módulos incluíam, pelo menos, a conversão A/D. Sob este aspecto, foram utilizados os módulos XBee, de fabricação da empresa DIGI, cuja especificações encontram-se no Apêndice.

5.4.1 Arquitetura

Neste trabalho foram utilizadas duas arquiteturas.

A primeira (Figura 50) foi escolhida porque nos permite, utilizando o modo *pass through* do módulo XBee, testar a comunicação entre os módulos, além de validar as rotinas do controlador fuzzy. Nesta arquitetura pode-se, através de dois potenciômetros, variar as entradas do controlador fuzzy, que correspondem ao erro e a variação do erro. O display de LCD é usado como IHM (Interface Homem-Máquina) e informa o valor das entradas e da saída do controlador.

A segunda arquitetura (Figura 49) foi utilizada para inserir o controlador fuzzy implementado em dsPIC no *loop* do Simulink. Conhecida como PIL *(Processor-in-loop co-simulation*), esta arquitetura, durante a co-simulação de hardware, permite que o Simulink, a cada *step* da simulação, envie dados para o dispositivo externo, ficando em espera até a chegada dos dados provenientes deste dispositivo, quando então o *loop* é concluído, iniciando-se a simulação do próximo *step*. Cada fabricante desenvolve uma solução para fornecer estas interfaces (proprietárias, serial, USB, Ethernet, etc.). No caso deste trabalho, foi utilizado, no controlador fuzzy implementado, uma interface desenvolvida para este fim. Tendo como componente principal o chip FT232BM fabricado pela FTDI, o qual já possui internamente a função de *host* USB, a interface disponibiliza ao *dsPIC* um canal de comunicação serial. Este mesmo fabricante disponibiliza em sua página na internet os drivers necessários ao interfaceamento da plataforma utilizada (Windows, Linux, Mac e etc.)

com o referido chip, transformando a porta USB em uma porta serial virtual. Através do *toolbox* de comunicação serial *RS232 Blockset* (Daga, 2010), tornou-se possível a utilização da ferramenta *PIL* no *Simulink*.

5.5 Implementação

Para a implementação do controlador fuzzy utilizado neste trabalho foram necesárias quatro etapas:

- Avaliação e escolha do microcontrolador;
- Avaliação e escolha dos módulos sem fio (Zigbee);
- Desenvolvimento e testes de um sistema operacional e das rotinas fuzzy;
- Projeto e montagem de interfaces Zigbee, necessárias à alimentação e condicionamento de sinais para o módulo Zigbee.

5.5.1 Apresentação da placa microncontroladora

Para a implementação foi utilizado um microcontrolador dsPIC30F4011 de fabricação da Microchip. Para tanto, várias plataformas de desenvolvimento da família 30/33 foram avaliadas e, dentro dos critérios desejados para esta plataforma, tais como display de LCD de, pelo menos, 2 linhas x 16 colunas, interface RS-232, capacidade de programação *on board* e preço, foram selecionados três fabricantes de kits (Microgênios, Cerne e Labtools). Dentre eles, foi selecionada a plataforma de desenvolvimento da Microgênios pelo seu menor preço (Figura 51).

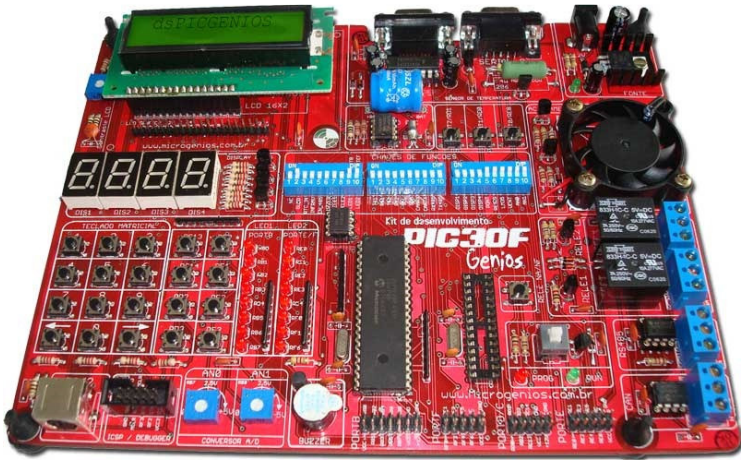


Figura 51 – Kit de desenvolvimento da Microgênios para o dsPIC30F4011

5.5.2 Apresentação dos módulos de comunicação sem fio

O módulo XBee (série 802.15.4), um dos primeiros disponíveis no mercado brasileiro, ainda não contava com a função de roteamento implementada, o que limita nosso estudo numa rede do tipo estrela. Os módulos foram adquiridos da empresa Rogercom (Rogercom, 2007) junto com uma interface USB/ZigBee (Figura 52). A partir destes módulos foram montados os dispositivos utilizados para teste deste trabalho, os quais serão tratados posteriormente.

Figura 52 – Kit da COM-USBBEE

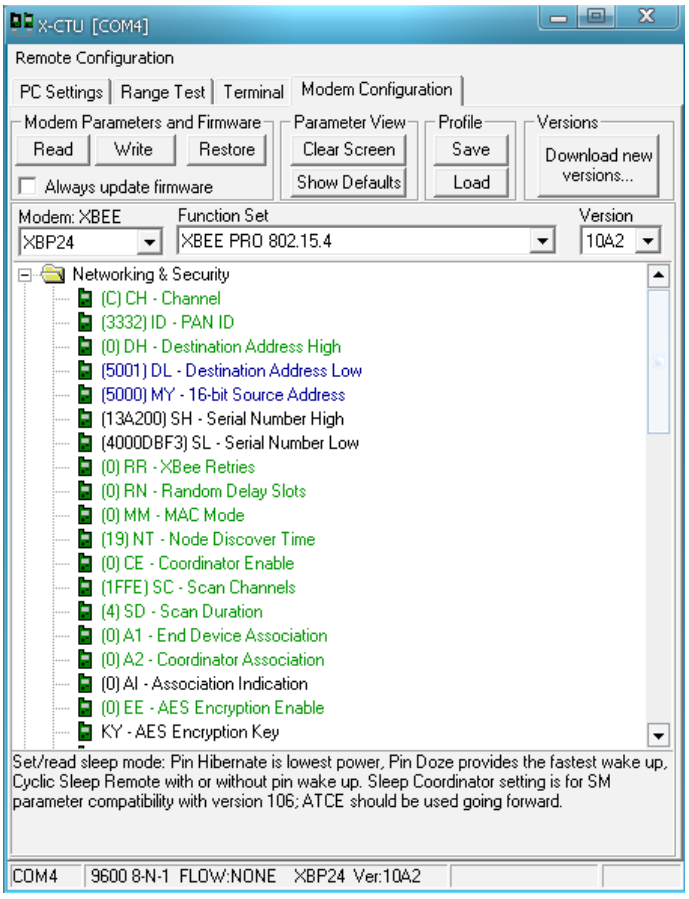


Figura 53 – Programa X-CTU

Para a configuração dos módulos, foi utilizado o programa X-CTU (Figura 53), o qual foi obtido no site do fabricante do módulo (Digi, 2008). Através de um drive da empresa FTDI (FTDI, 2008) (fabricante do chip que compõe a interface USB/RS232), o módulo, ao ser conectado numa das portas USB, passa a ser tratado pelo Windows como uma porta serial (COM).

Para atender aos objetivos deste trabalho, os módulos foram configurados para a formação de uma rede, cujo coordenador, localizado no dsPIC, recebe os valores (conversores A/D e I/Os digitais) vindos do elemento final, a cada 500ms.

5.5.3 Apresentação das rotinas

Para facilitar a implementação do controlador fuzzy, foi desenvolvido um mini sistema operacional que conta com rotinas básicas para um ambiente de programação mais amigável. Para isso, foram desenvolvidas rotinas para:

- Gerenciamento da temporização de envio de mensagens para o display, cuja função é enviar byte a byte de cada mensagem para o display segundo uma temporização pré-estabelecida ($clk_{dsPIC} \approx 8ns$ e $clk_{display} \approx 1ms$). Esta rotina também é responsável pelo gerenciamento da fila de mensagens;
- Gerenciamento da temporização de envio e recebimento de mensagens via porta serial (comunicação com o módulo coordenador da rede ZigBee) cuja função é equivalente a rotina do display. Além de enviar as mensagens para o coordenador ZigBee, a rotina recebe e trata as mensagens que chegam dos módulos da rede;
- Rotinas de conversão A/D para o dsPIC;
- Rotinas para fuzzificação, inferência e defuzzificação.

5.5.4 Módulos de interfaceamento e testes

Para a implementação deste controlador três interfaces foram construídas. A primeira foi necessária para interligar o dsPIC ao módulo coordenador da rede ZigBee (Figura 54 e Figura 55). A segunda foi necessária para a leitura dos valores das entradas do controlador fuzzy, os quais foram gerados através de dois potenciômetros, cujos cursores encontravam-se ligados às portas AD0 e AD1 do módulo ZigBee e, uma vez convertidos pelo módulo, foram transmitidos ao

coordenador da rede (Figura 56 e Figura 57). A terceira foi utilizada para permitir a comunicação entre o dsPIC e o Simulink, utilizando a comunicação serial via USB e é baseada no chip FT232 da FTDI (Figura 58 e Figura 59).

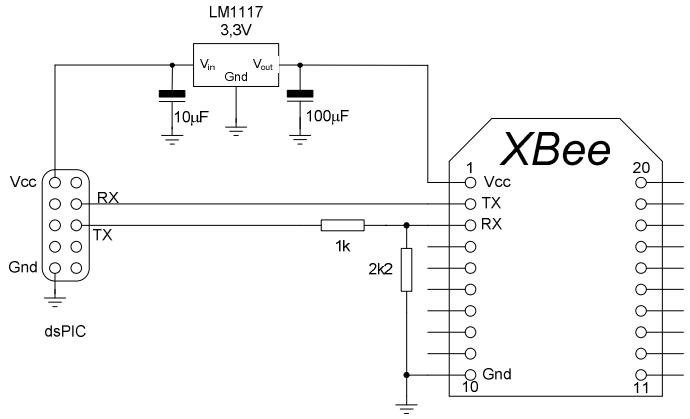


Figura 54 – Interface serial (esquemático)

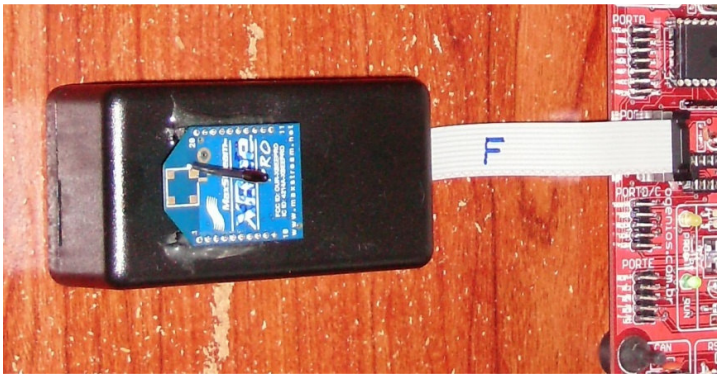


Figura 55 – Interface serial XBee (montado)

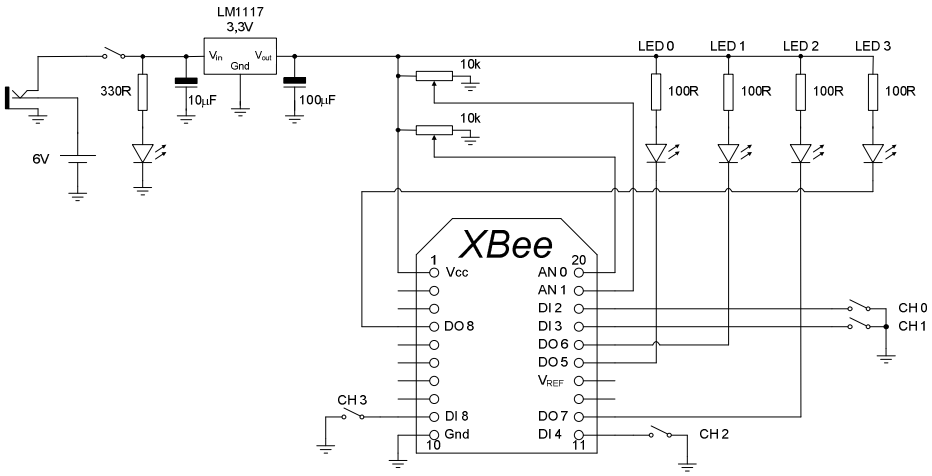


Figura 56 – Módulo de teste (esquemático)

Figura 57 – Módulo de teste (montado)

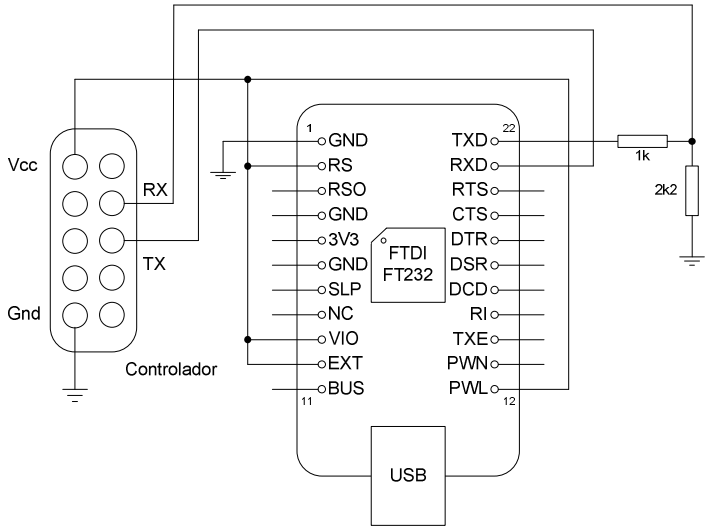


Figura 58 – Módulo USB/RS232 (esquemático)

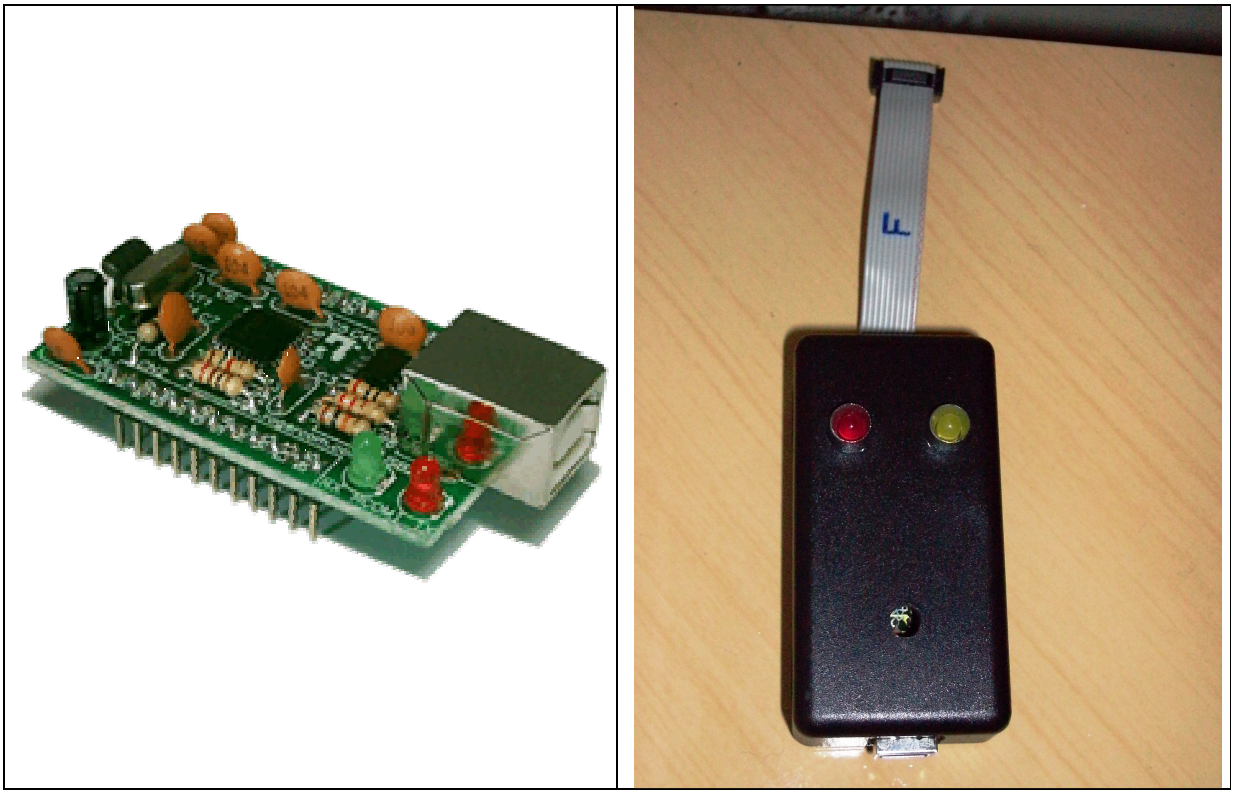


Figura 59 – Módulo USB/RS232 (a) Placa e (b) Montado

6. ESTUDO DE CASOS

O estudo de casos foi organizado em três partes.

Primeiramente foi realizado um teste funcional do controlador fuzzy implementado em dsPIC.

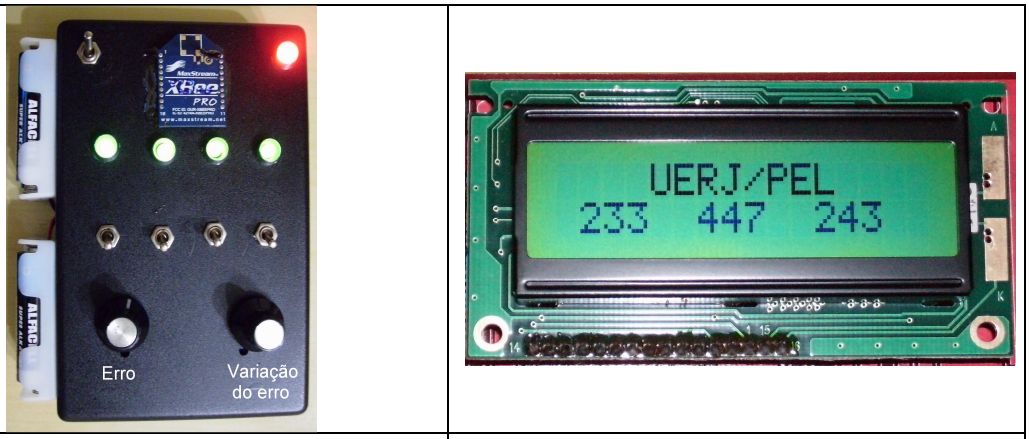
Em seguida foi realizado um mapeamento deste controlador e avaliado seu funcionamento numa malha de controle implementada no Simulink. Para isso, o controlador implementado ou DUT (*Device Under Test*), foi introduzido na malha através de uma comunicação serial/USB.

Finalmente foi verificada a estabilidade de um sistema fuzzy em comparação com um sistema PID em diversos ambientes.

6.1 Teste funcional do controlador implamentado

Para verificar o funcionamento do controlador fuzzy implementado, foi utilizado o módulo de teste (Figura 57) e a interface serial XBee (Figura 55). Para isso, o controlador implementado foi configurado com as mesmas características do o controlador fuzzy definido no exemplo *tank* do Simulink, o qual se encontra minuciosamente explicado no item 6.2.

Através dos potenciômetros foram simulados os valores correspondentes ao erro e variação do erro e assim foi possível avaliar-se alguns pontos (Figura 60) e compará-los com o valor teórico (Matlab), cujos erros permaneceram inferior a 4 em 1000, ou seja, 0,4%.



(a) (b)

Figura 60 – Teste do controlador fuzzy dsPIC – (a) interface de entrada (b) Resultados

6.2 Teste do controlador implementado

Para teste do controlador implementado foi utilizado um modelo apresentado no exemplo de controle de nível de um tanque existente no Matlab. Neste exemplo, um tanque que possui um tubo na entrada e outro na saída foi modelado utilizando-se o Simulink. No modelo, pode-se mudar a vazão de entrada, mas a vazão de saída depende do diâmetro da tubulação de saída (o qual é constante), assim como a pressão do tanque (que varia com o nível do tanque). Com estas características, o sistema apresenta algumas não-linearidades (Figura 61).

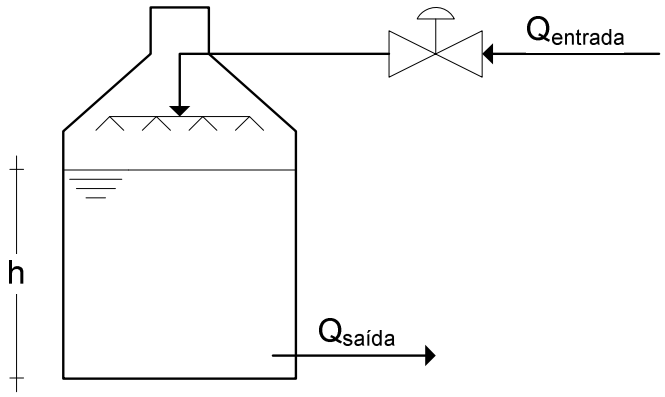


Figura 61 – Modelo do controle de nível de um tanque

A vazão de saída ($Q_{saída}$) através da tubulação de área *a* é dada pela Equação 27, onde *h* é a altura entre a lâmina d'água e a tubulação de saída e *g* é a aceleração da gravidade.

$$Q_{saída} = a\sqrt{2gh} \tag{27}$$

Utilizando-se a lei de conservação de massa, obtemos a Equação (28), onde *A* é a área do tanque.

$$A\frac{dh}{dt} = q_{entrada} - q_{saida} = q_{entrada} - a\sqrt{2gh} \tag{28}$$

Manipulando-se a equação acima, chega-se a equação final da modelagem que será utilizada no Simulink (Equação (29).

$$h(t) = h(0) + \int_0^t \frac{1}{A}[q_{entrada}(t') - q_{saída}(t')]dt'$$
$$h(0) + \int_0^t \left[q_{entrada}(t') - a\sqrt{2gh(t')}\right]dt' \quad (29)$$

O controlador de nível do tanque necessita informações sobre o nível atual do tanque e deve ser capaz de ajustar a válvula de controle de vazão da entrada. A entrada do controlador é alimentada com o valor do erro no nível (nível desejado – o nível real) e a saída apresenta o valor da "velocidade" que a válvula deverá abrir ou fechar. O resultado final do ambiente para o Simulink apresentado na Figura 62. Neste ambiente pose-se escolher, através do bloco *Switch*, qual controlador será utilizado na simulação.

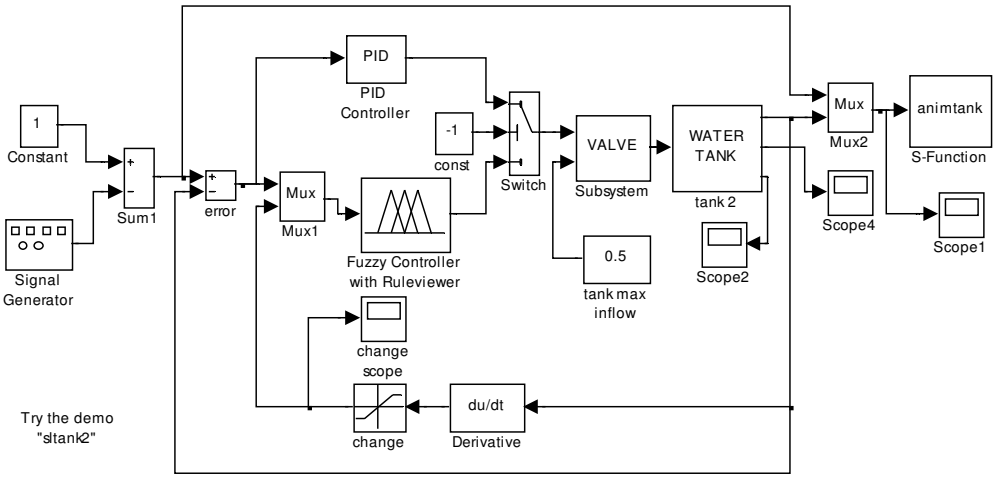


Figura 62 – Controle de nível de um tanque (exemplo retirado do Simulink)

Um gerador de sinais altera periodicamente o *set point* do sistema para se avaliar a resposta do sistema.

Inicialmente, o exemplo foi adaptado para que as funções de pertinência da saída, que originalmente eram triangulares, passassem a ser *singletons* e pudessem ser utilizadas como parâmetro de comparação (*benchmark*). Visando avaliar se a alteração impactou na qualidade do controle, o novo controlador fuzzy foi inserido no lugar do controlador original e o programa do Simulink foi executado.

6.2.1 Mapeamento comparativo do controlador implementado

Para a geração de um banco de dados, foi desenvolvida, no MATLAB, uma rotina que “varresse” os valores das entradas entre 0.0 e 100.0, com passo de 0.1 unidade. Ao final, foi gerado um gráfico de mapeamento (Figura 63).

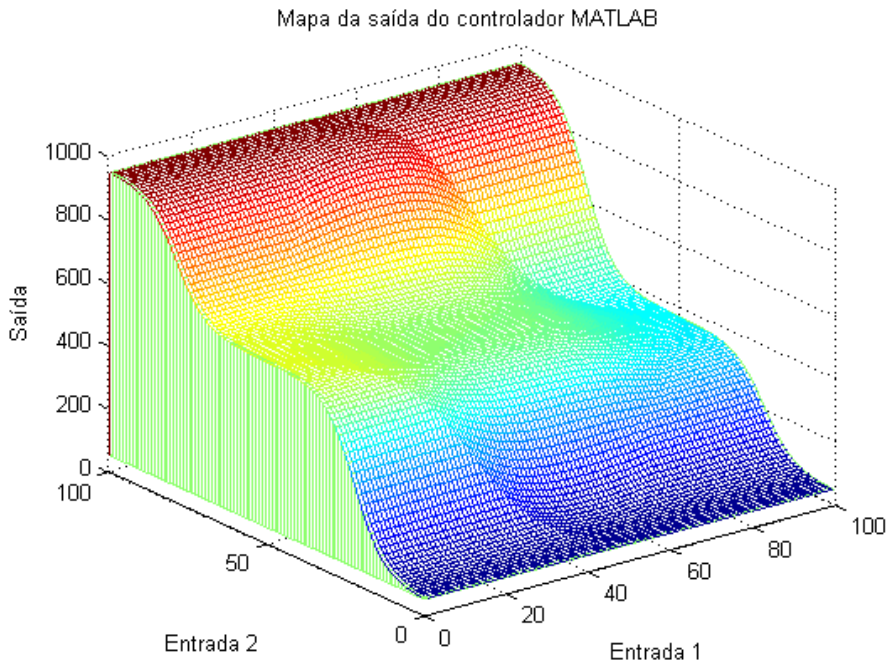


Figura 63 – Mapa do controlador fuzzy utilizado no exemplo do tanque do MATLAB

Também foi gerado um banco de dados com os resultados do controlador fuzzy implementado neste trabalho utilizando os mesmos critérios. Para a gravação deste banco de dados, os dados gerados pelo controlador dsPIC fuzzy foram transmitidos via porta serial/USB do controlador para um PC via *hyperterminal*. Este banco de dados também serviu para geração de um gráfico de mapeamento das entradas com a saída (Figura 64) e um gráfico de mapeamento do erro (Figura 65).

Tendo em vista que o controlador implementado com o dsPIC só trabalha com números inteiros, a Figura 63, Figura 64 e Figura 65 diferem de um multiplicador 10.

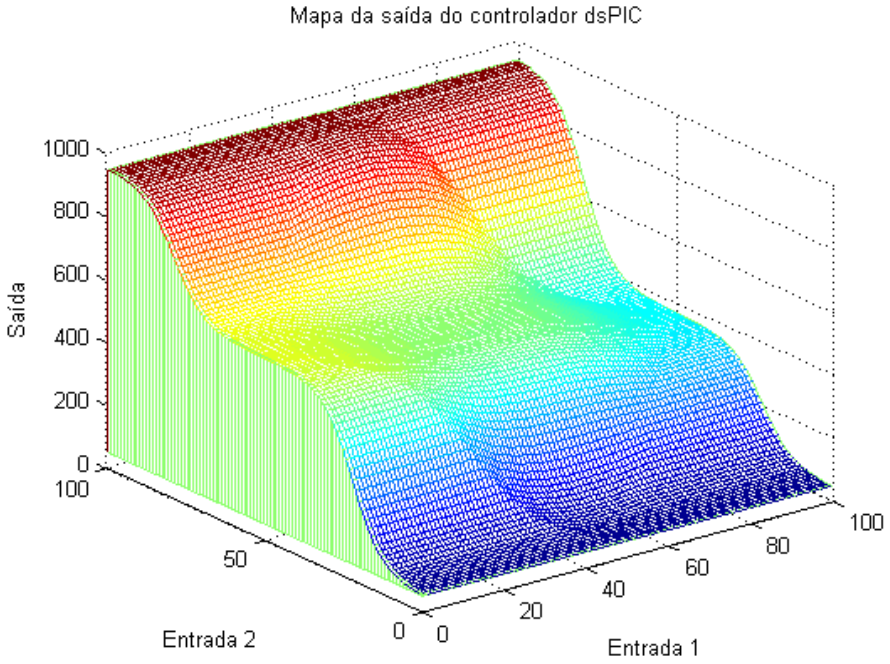


Figura 64 – Mapa do controlador fuzzy do exemplo do tanque do MATLAB implementado no dsPIC

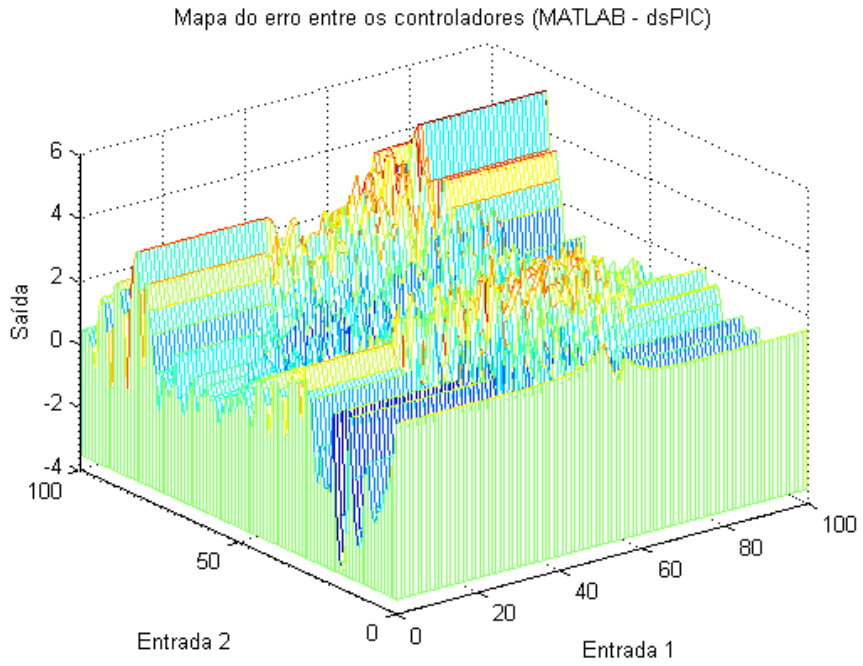


Figura 65 – Mapa do erro entre o controlador fuzzy utilizado no exemplo do tanque do MATLAB e o implementado no dsPIC

Os valores obtidos apresentaram um erro máximo de 06 (seis) unidades em 1000, ou seja, 0,6%. Este resultado é perfeitamente aceitável para uma implementação “simplificada” como a do objetivo deste estudo.

6.2.2 Avaliação do controlador no Simulink

Para a avaliação do controlador na malha do Simulink, foi utilizado o *toolbox RS232 Blockset* (Daga, 2010). Este *toolbox* possui vários blocos para a configuração de uma comunicação no ambiente do Simulink (Figura 66).

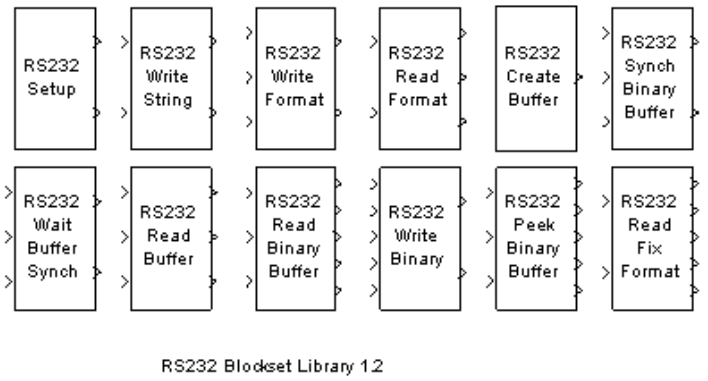


Figura 66 – Toolbox para comunicação serial para o Simulink

A partir destes blocos, foi montado o ambiente da Figura 67. O contorno ① demarca as operações de condicionamento do sinal do Simulink ($-1{,}0 \leq x_{ent} \leq +1{,}0$) para a faixa do controlador dsPIC ($0 \leq x_{ent} \leq 1000$). O mesmo ocorre na saída (contorno ②), quando este sinal de saída ($0 \leq x_{sai} \leq 1000$) é condicionado à faixa do Simulink ($-1{,}0 \leq x_{sai} \leq +1{,}0$).

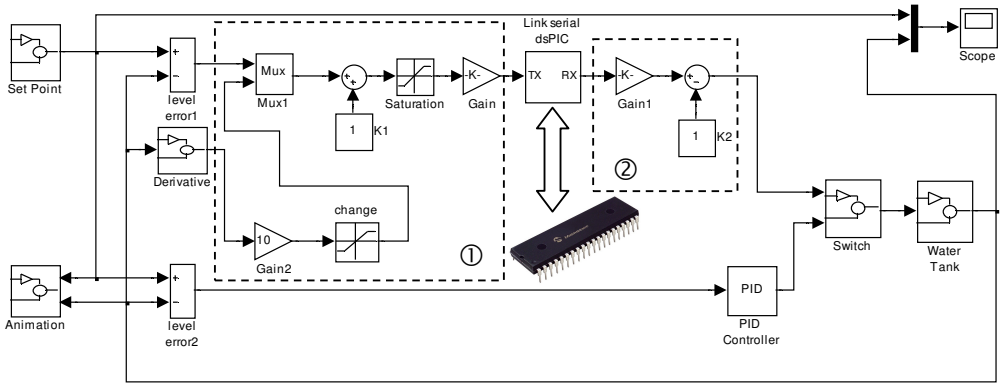


Figura 67 – Ambiente para teste do controlador implementado com dsPIC

O bloco intitulado *Link serial dsPIC* é apresentado explodido na Figura 68 e é responsável pela comunicação entre o ambiente virtual do Simulink e o controlador real implementado com dsPIC.

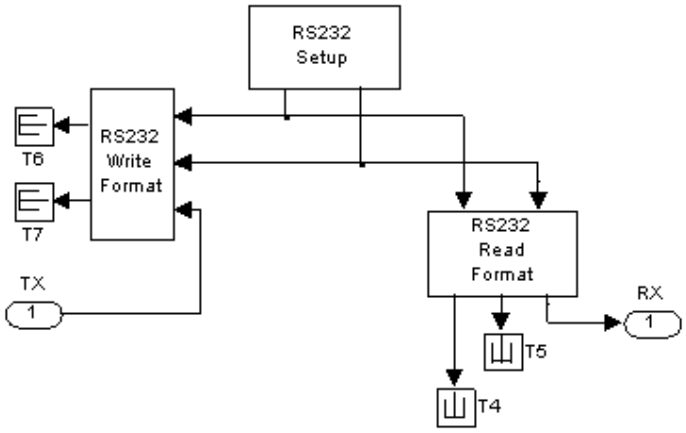


Figura 68 – Conteúdo do bloco de comunicação serial para dsPIC.

O resultado teórico, resultante da simulação, é apresentado na Figura 69 para o controlador PID e na Figura 70 para um controlador fuzzy.

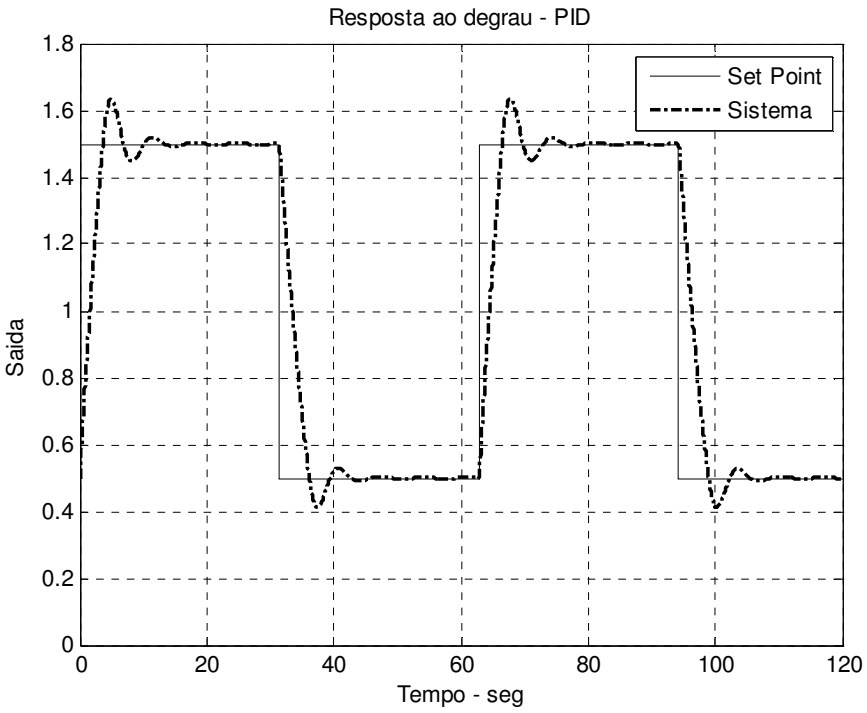


Figura 69 – Resposta teórica do controlador PID.

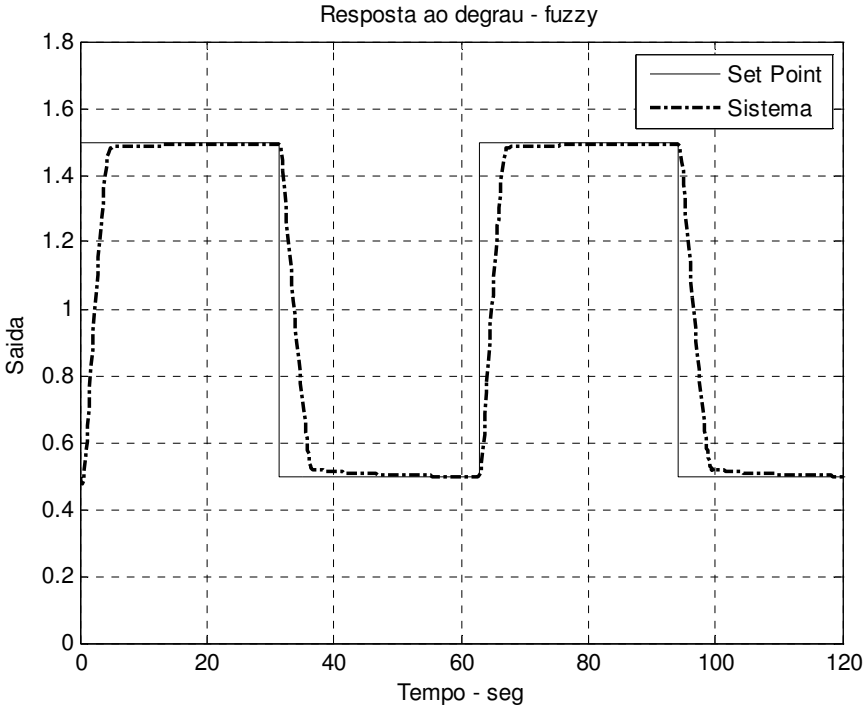


Figura 70 – Resposta teórica do controlador fuzzy.

O resultado do controlador real implementado com dsPIC encontra-se na Figura 71.

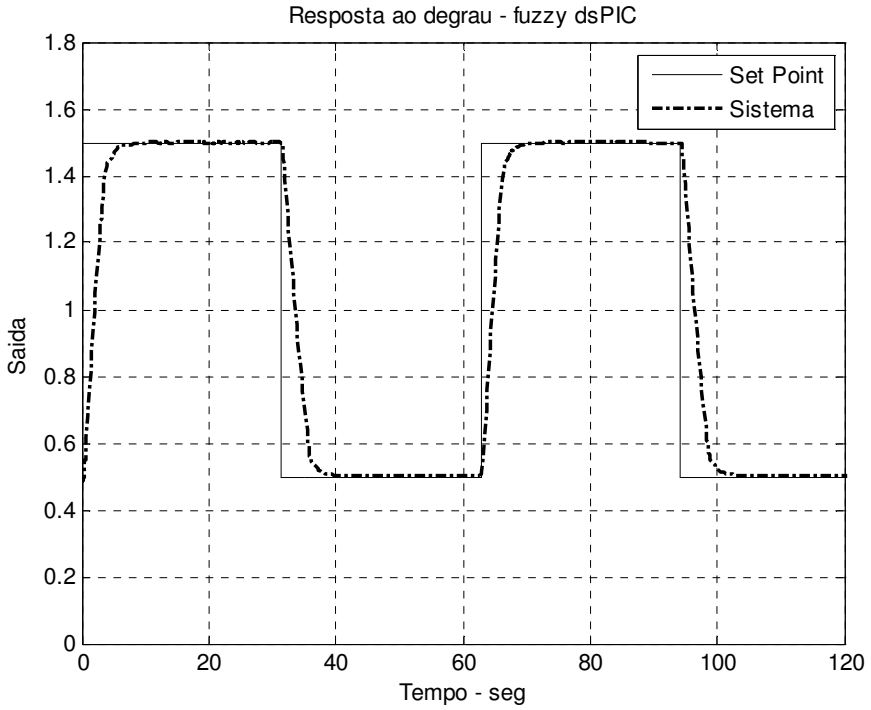


Figura 71 – Resposta do controlador fuzzy implementado com dsPIC

O resultado apresentado na Figura 71 demonstra que o controlador fuzzy implementado com dsPIC e inserido na malha de controle do Simulink (PIL) obteve o mesmo resultado que o controlador fuzzy teórico e que o erro de 0,6% apontado no item 6.2.1 não afetou o desempenho do controlador.

6.2.3 Avaliação do controlador com a rede

Nesta avaliação do sistema, foi utilizado o controlador implementado e a rede virtual, *True Time* (Figura 72). Novamente os contornos ① e ② foram utilizados para condicionar os sinais de entrada e saída, respectivamente. O bloco TX é responsável pela simulação da conversão A/D, envelopamento da mensagem e envio pela rede. O bloco RX simula o caminho inverso, ou seja, recebimento, demodulação da mensagem e conversão D/A. O bloco de REDE simula o meio físico (no caso o IEEE 802.15.4).

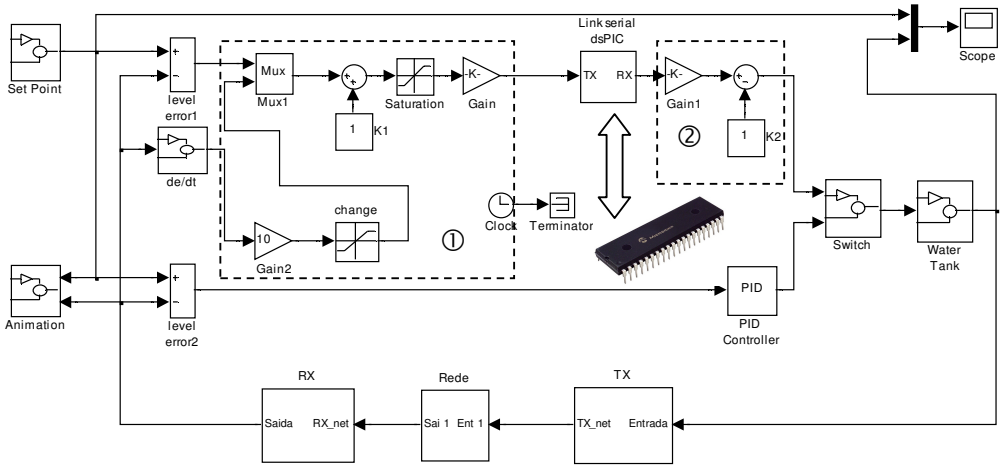
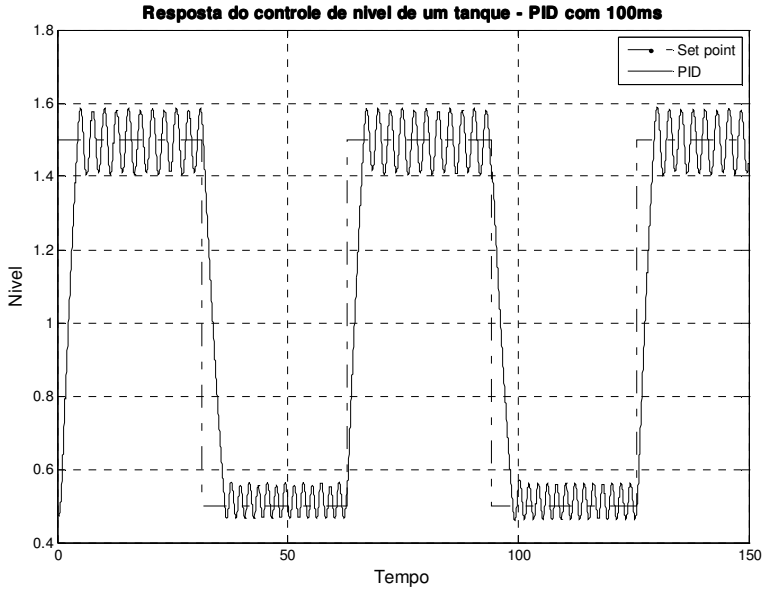


Figura 72 – Controle de nível de um tanque com controlador fuzzy (dsPIC) e rede

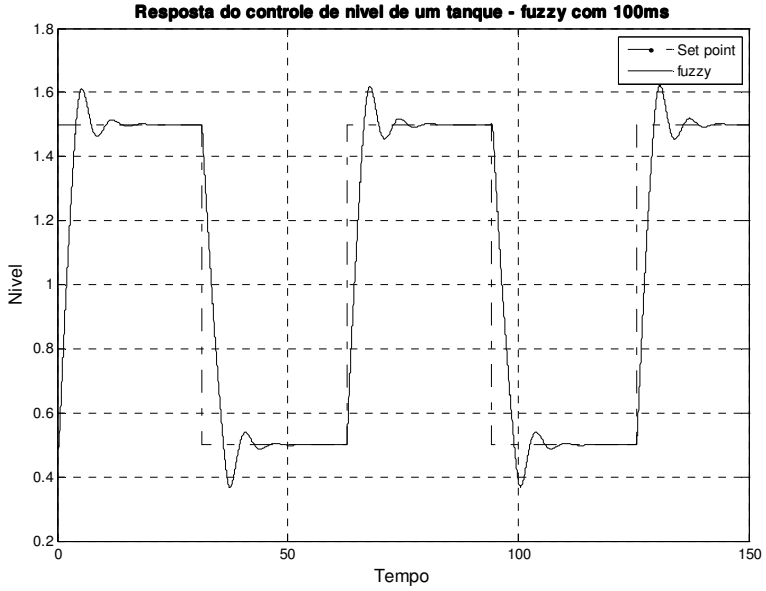
Com o intuito de se simular os módulos ZigBee da DIGI (XBee), base do estudo proposto, a rede foi configurada para operar com os seguintes parâmetros:

- taxa de transferência de 1,0 Mbit/seg;
- pacotes de 10 bits;
- taxa de amostragem do sensor de 100ms.

Os resultados mostram uma instalibização devido à inserção desta rede na malha de controle (Figura 73). Esta constatação inspirou a realização da próxima seção deste trabalho.



(a)



(b)

Figura 73 – Resposta do controle de nível de um tanque (rede)

6.3 Estudo do controle da Entrada e Saída Remota

Para este estudo, escolheu-se um controlador PID, já consagrado na indústria, e sintonizado utilizando-se métodos clássicos. Através do método proposto por (Jantsen, 1998), sem a preocupação de qualquer melhoria do controlador, foi construído um controlador fuzzy e introduzido no mesmo ambiente.

6.3.1 Primeiro cenário – Sistemas sem atraso

Para o primeiro teste deste cenário foi utilizado um ambiente com uma função de transferência com um pólo positivo e sem atraso. O bloco ① foi introduzido no sistema para a retenção do valor referente à diferença durante o *step* da simulação.

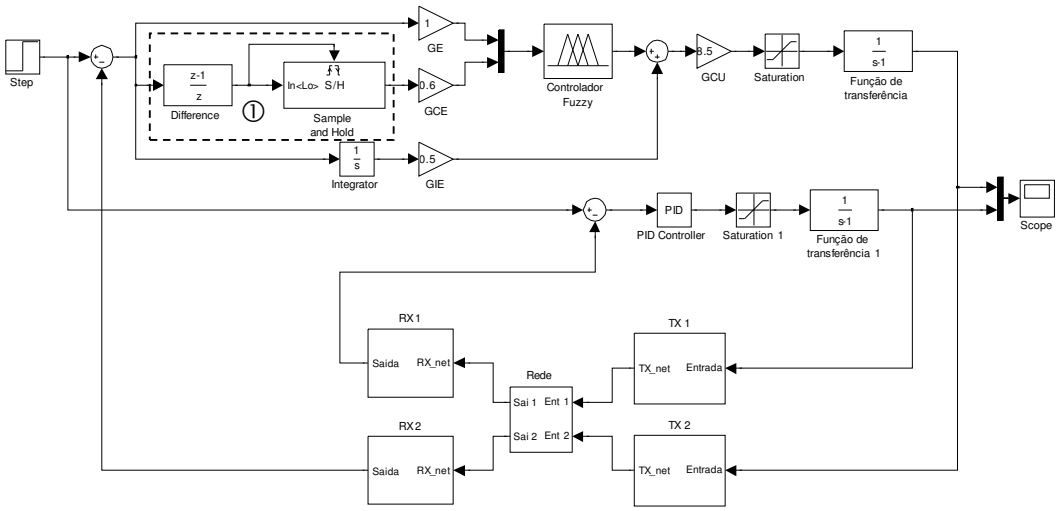


Figura 74 – Cenário com 01 pólo real positivo

Os parâmetros utilizados nos controladores PID e fuzzy encontram-se explicitadas na tabela abaixo. Fruto de ensaios realizados foi verificado que o melhor resultado foi conseguido quando se atribuía o ganho unitário ao erro e, quando possível, um ganho maior que o unitário na atuação do controlador fuzzy, ou seja, GU > 1,0.

$$\begin{array}{l} P = 8,5 \\ I = 5,0 \\ D = 4,23 \end{array} \quad \Rightarrow \quad \begin{array}{l} GE = 1,00 \\ GCE = 0,85 \\ GIE = 0,59 \\ GU = 5,0 \end{array}$$

Esta arquitetura resultou na resposta típica apresentada na Figura 75.

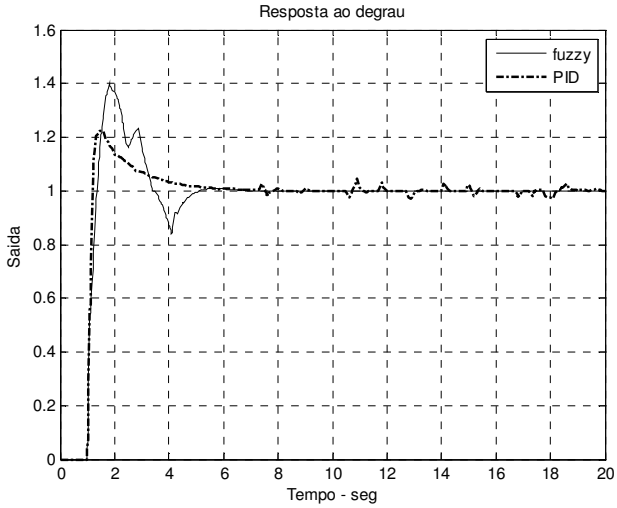


Figura 75 – Resposta ao degrau obtida com um sistema com 01 pólo real positivo

Nesta figura pode-se constatar que, apesar do controlador PID apresentar uma resposta mais “bem comportada” no período transitório, o controlador fuzzy apresentou uma estabilidade melhor em regime permanente.

Para o segundo teste deste cenário foi utilizado um ambiente com uma função de transferência com um pólo negativo e sem atraso. O bloco de saturação também foi utilizado para avaliar o comportamento do sistema em função da energia disponível para a atuação do controle. Novamente o bloco ① foi introduzido no sistema para a retenção do valor referente à diferença durante o *step* da simulação.

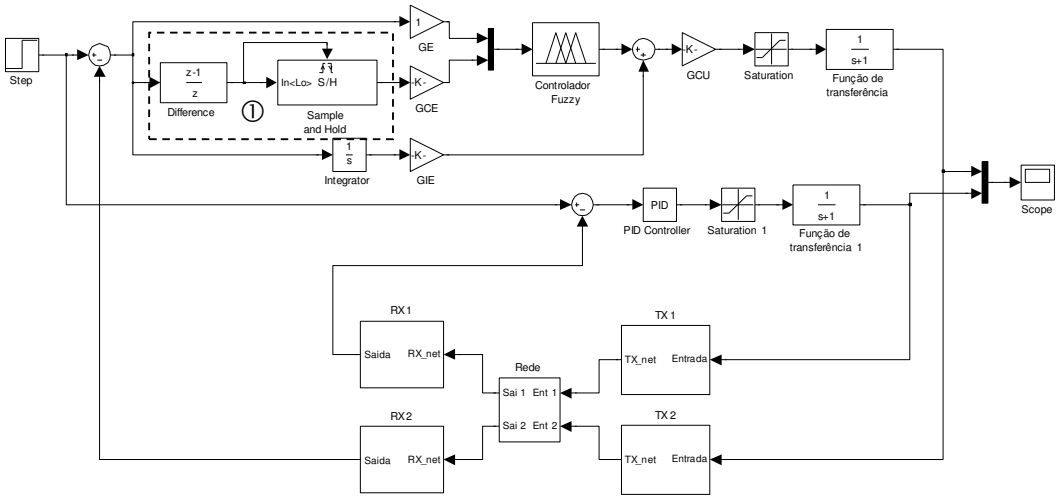


Figura 76 – Cenário com 01 pólo real negativo

Os parâmetros utilizados nos controladores PID e fuzzy encontram-se explicitadas na tabela abaixo.

P = 1,25		GE = 1,00
I = 2,18	⇒	GCE = 0,14
D = 0,17		GIE = 1,75
		GU = 1,25

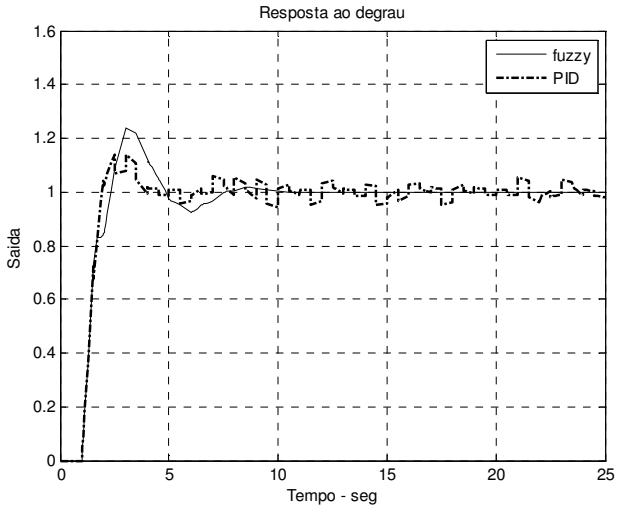


Figura 77 – Resposta ao degrau obtida com um sistema com 01 pólo real negativo

Nesta figura pode-se constatar que, apesar do controlador PID apresentar uma resposta mais “bem comportada” no período transitório, o controlador fuzzy apresentou uma estabilidade significativamente melhor em regime permanente.

6.3.2 Segundo cenário – Sistemas com atraso

Para o primeiro teste deste cenário foi utilizado um ambiente com uma função de transferência com um pólo positivo e com atraso. O bloco de saturação foi utilizado para avaliar o comportamento do sistema em função da energia disponível para a atuação do controle. Mais uma vez o bloco ① foi introduzido no sistema para a retenção do valor referente à diferença durante o *step* da simulação.

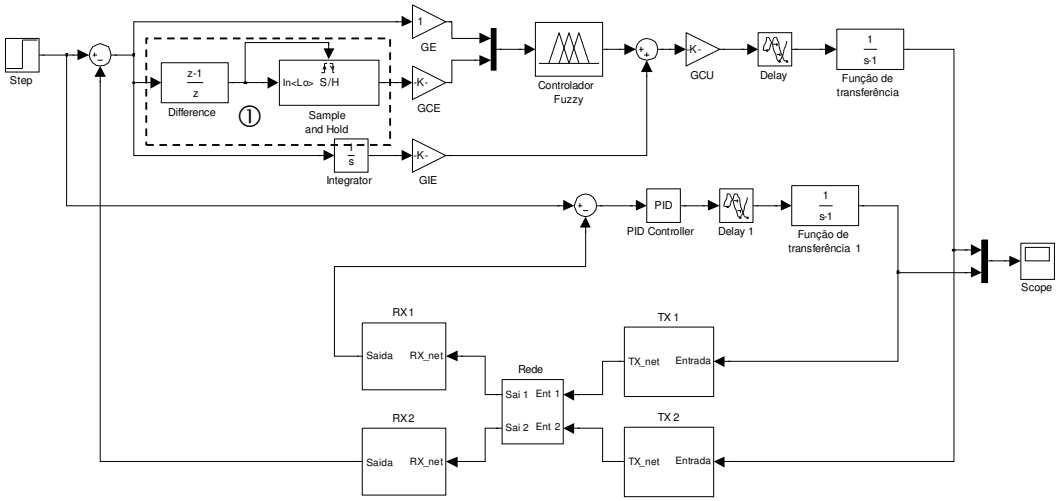


Figura 78 – Cenário com 01 pólo real positivo

Os parâmetros utilizados nos controladores PID e fuzzy encontram-se explicitadas na tabela abaixo.

$$\begin{matrix} P = 7{,}05 \\ I = 39{,}2 \\ D = 0{,}47 \end{matrix} \quad \Rightarrow \quad \begin{matrix} GE = 1{,}00 \\ GCE = 0{,}07 \\ GIE = 5{,}56 \\ GU = 7{,}05 \end{matrix}$$

O controlador PID apresentou ótimos resultados em ambientes sem rede. Porém, quando a rede foi inserida na malha o sistema PID instabilizou (Figura 79), impossibilitando qualquer análise. Com o intuito de gerar um gráfico comparativo, o valor de K_D foi zerado na simulação. Esta arquitetura resultou na resposta típica apresentada na Figura 79.

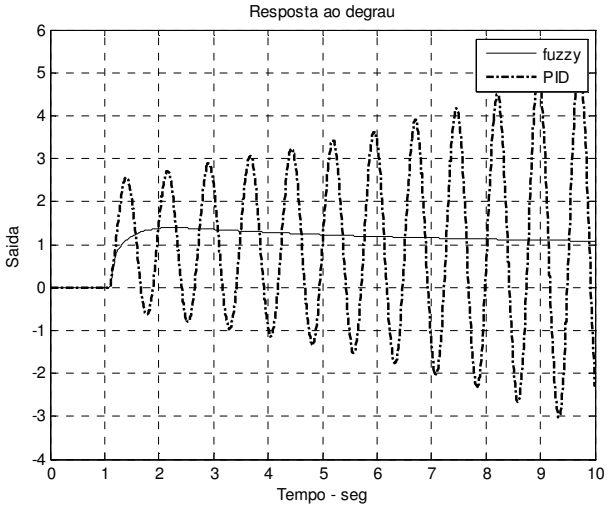


Figura 79 – Resposta típica obtida de um sistema com 01 pólo real positivo

Para o segundo teste deste cenário foi utilizado um ambiente com uma função de transferência com um pólo negativo e com atraso. Os blocos de saturação também foram utilizados para avaliar o comportamento do sistema em função da energia disponível para a atuação do controle. Novamente o bloco ① foi introduzido no sistema para a retenção do valor referente à diferença durante o *step* da simulação.

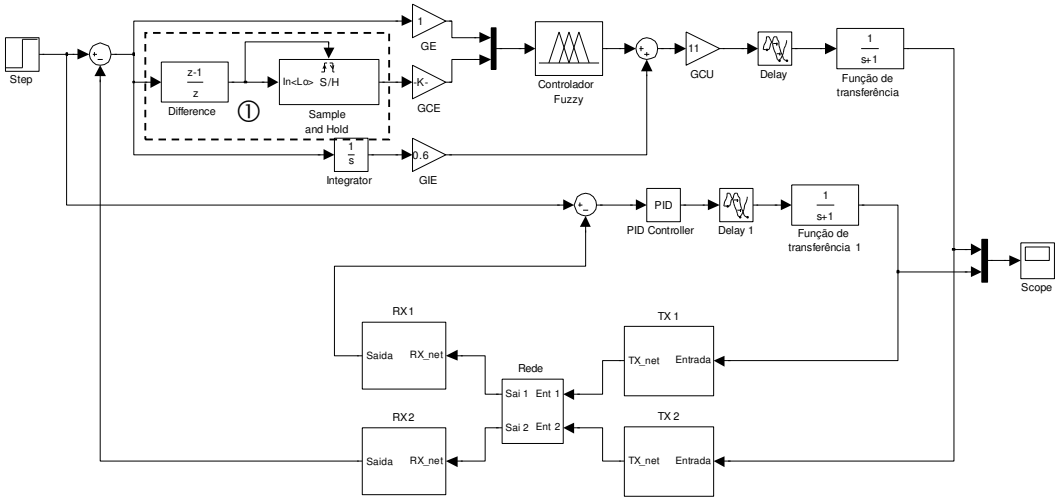


Figura 80 – Cenário com 01 pólo real negativo

Esta arquitetura resultou na resposta típica apresentada na Figura 81.

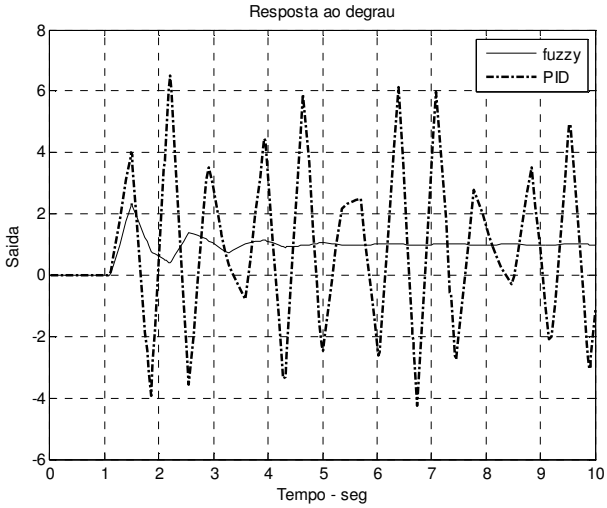


Figura 81 – Resposta obtida de um sistema com 01 pólo real negativo

Os parâmetros utilizados nos controladores PID e fuzzy encontram-se explicitadas na tabela abaixo.

P = 11,0		GE = 1,00
I = 6,65	$\Rightarrow$	GCE = 0,05
D = 0,50		GIE = 0,60
		GU = 11,0

Assim como no exemplo do pólo positivo, foi constatado que o controlador PID só pode ser utilizado sem a rede, pois com ela o sistema torna-se extremamente instável. Para a plotagem acima, o valor de K_D utilizado foi o zero.

6.4 Avaliação da robustez dos sistemas de controle

Visando o estudo da robustez do controlador fuzzy em comparação a um controlador PID, foram realizados ensaios que avaliam o impacto de alterações que ocorrem posteriormente ao projeto e implementação do sistema de controle.

Desgastes de peças ou defeitos podem provocar distúrbios na resposta do controle de controle devido a flutuações do pólo do sistema ou limitação da atuação do controle. Todas estas situações são avaliadas a seguir.

6.4.1 Estudo da variação do pólo

Neste cenário foi avaliada a robustez de um sistema, que inicialmente foi projetado para apresentar um comportamento ótimo para um pólo de valor unitário,

passa a apresentar variações no valor deste pólo de ±10%. A Figura 82 e Figura 83 apresentam os valores encontrados para o *overshoot* e tempo de estabilização, tanto para o controlador PID quanto para o fuzzy, para pólos *s = -0,9*, *s = -1,0* (valor original do projeto) e *s = -1,1*.

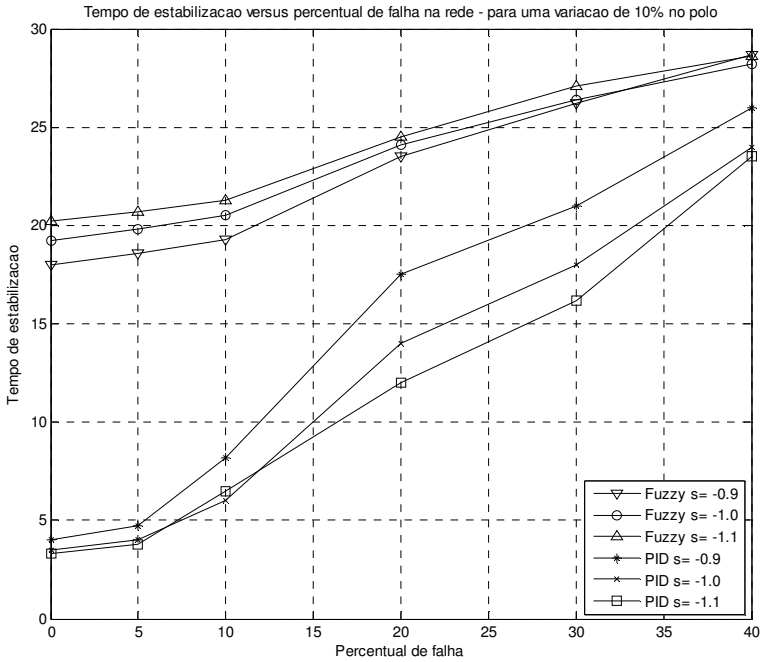


Figura 82 – Variação do tempo de estabilização em função do pólo num NCS fuzzy

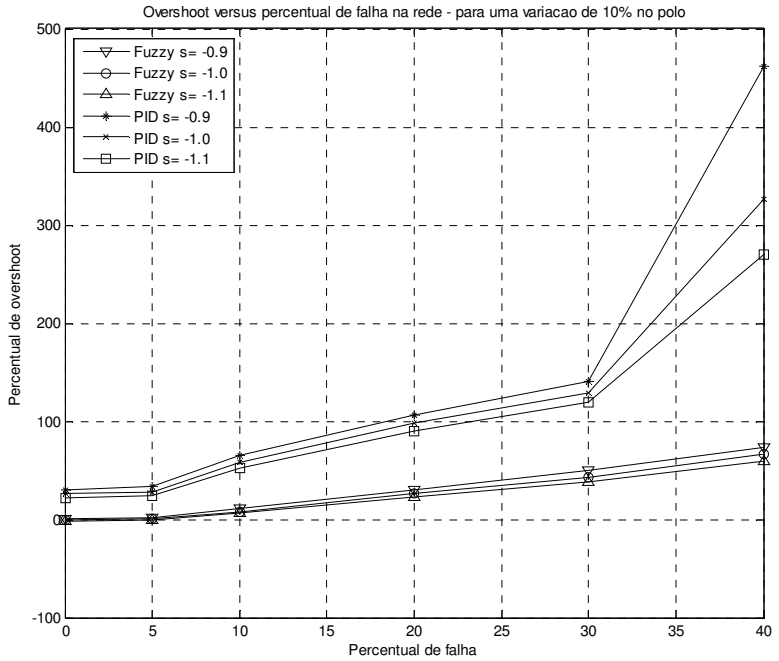


Figura 83 – Variação do *overshoot* em função do pólo num NCS fuzzy

Neste estudo pode-se verificar que o controlador fuzzy, apesar de apresentar um tempo de estabilidade superior ao encontrado com o controlador PID, possui uma maior robustez no *overshoot*.

6.4.2 Estudo da resposta a um degrau unitário

Neste cenário foi avaliada a resposta dos sistemas PID e fuzzy quando excitados por um degrau unitário. Um bloco de saturação foi utilizado para também se avaliar o comportamento do sistema em função da energia disponível para a atuação do controle. Foi utilizada uma rotina do Matlab para calcular as métricas (*overshoot* e tempo de estabilização) do sistema implantado. Ambos os sistemas (fuzzy e PID) foram submetidos às mesmas condições e os resultados, frutos da média de 20 leituras provenientes deste programa, foram *plotados* e avaliados.

Para o primeiro cenário foi utilizado um sistema com pólo negativo.

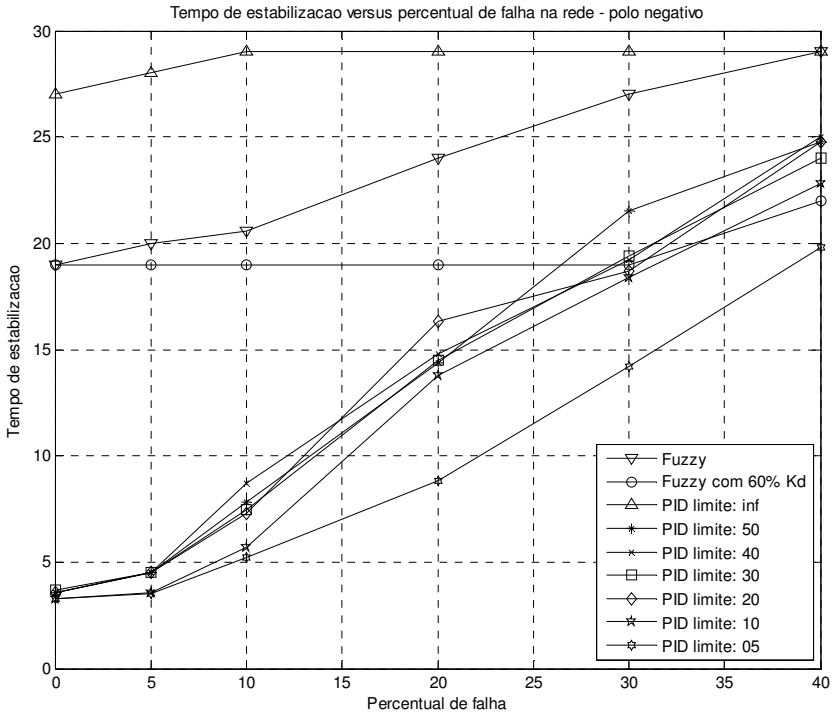


Figura 84 – Variação do tempo de estabilização em função da perda de pacotes num NCS fuzzy

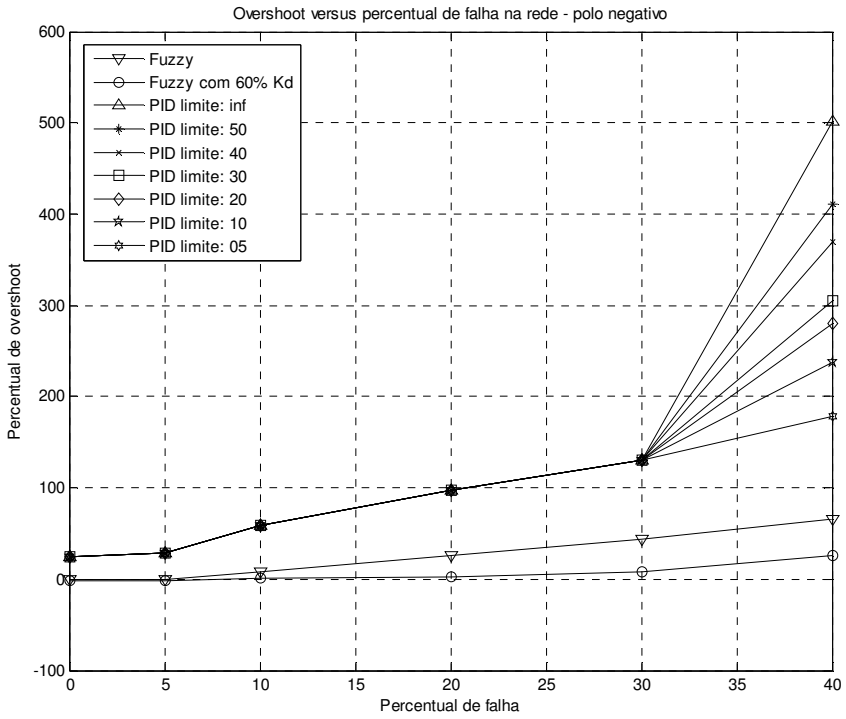


Figura 85 – Variação do *overshoot* em função da perda de pacotes num NCS fuzzy

Novamente, pode-se verificar que o controlador fuzzy, apesar de apresentar um tempo de estabilidade superior ao encontrado com o controlador PID, possui uma maior robustez no *overshoot*.

Para o segundo cenário foi utilizado um sistema com pólo positivo.

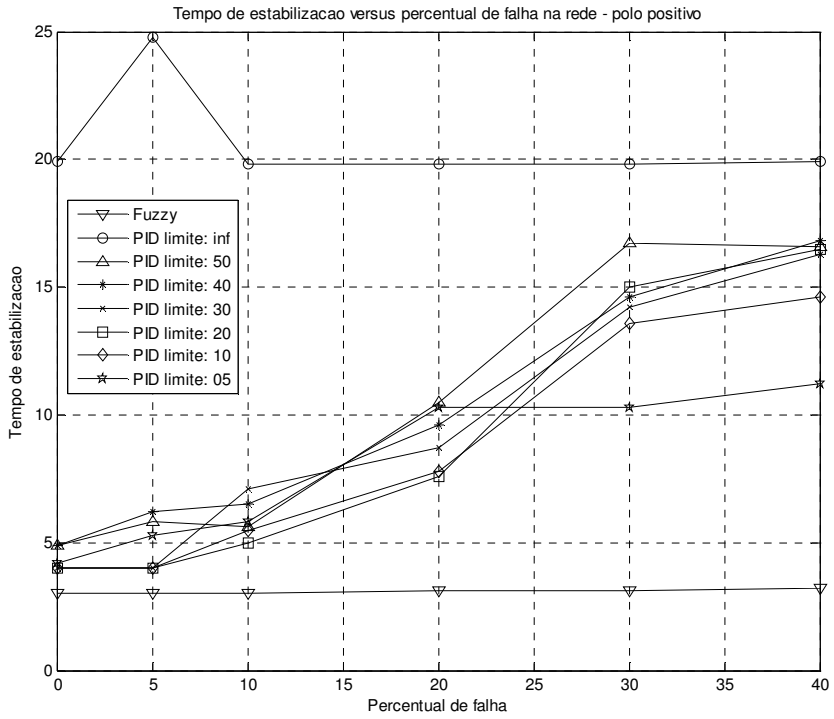


Figura 86 – Variação do tempo de estabilização em função da perda de pacotes num NCS fuzzy

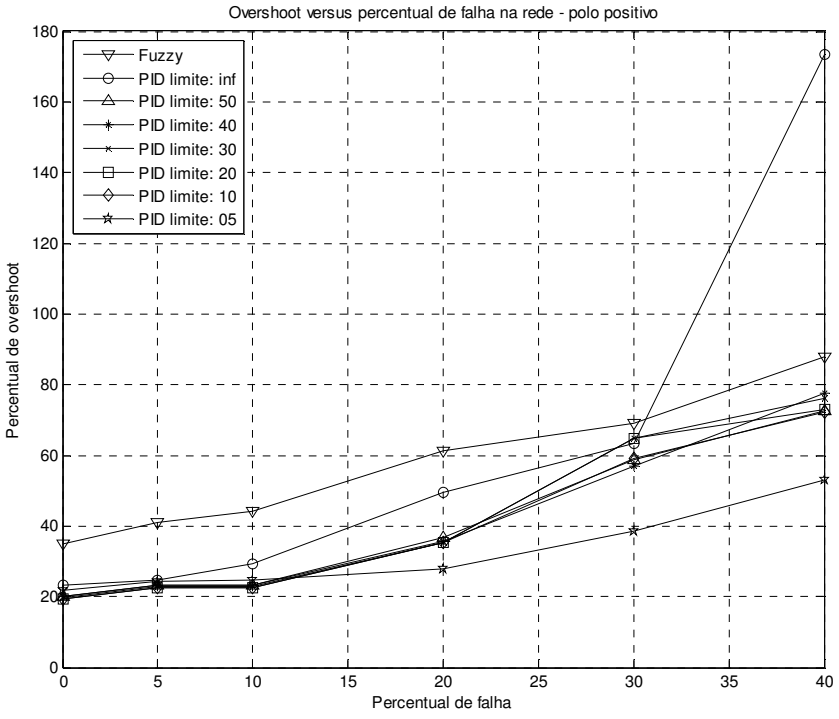


Figura 87 – Variação do *overshoot* em função da perda de pacotes num NCS fuzzy

Neste caso, o controlador fuzzy não apresentou variações significativas à limitação da atuação, ao contrário do controlador PID que apresentou uma maior sensibilidade a esta limitação do valor máximo.

6.4.3 Estudo do impacto da localização da rede numa malha de controle

Song (2006) apresenta em seu trabalho que a localização da rede de sensores numa malha de controle pode impactar no comportamento da mesma. Neste trabalho também foram realizados testes com controladores PID e fuzzy e avaliada a resposta a esta situação (Figura 88 e Figura 89).

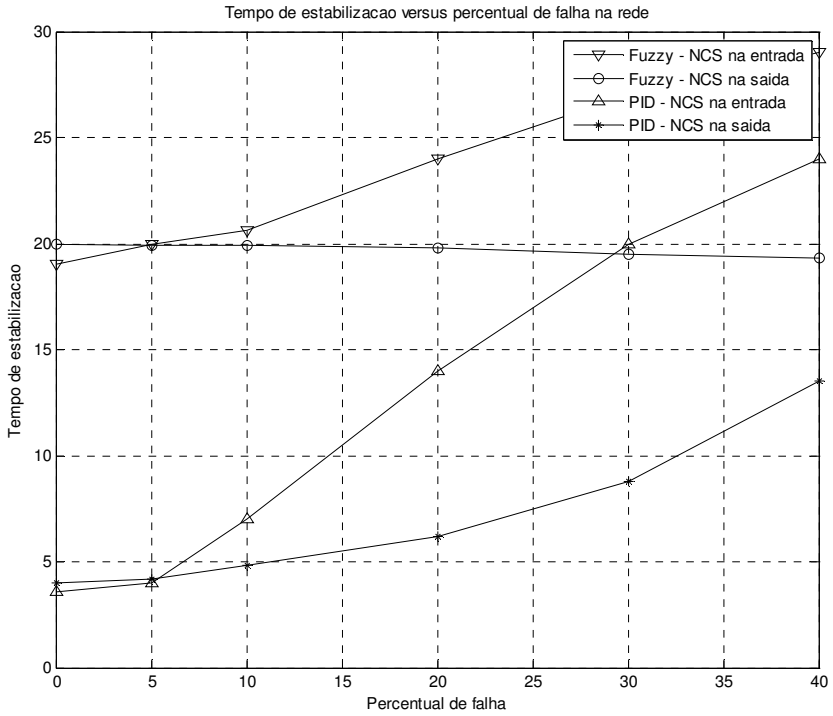


Figura 88 – Variação do tempo de estabilização em função da perda de pacotes e localização da rede de sensores num NCS fuzzy

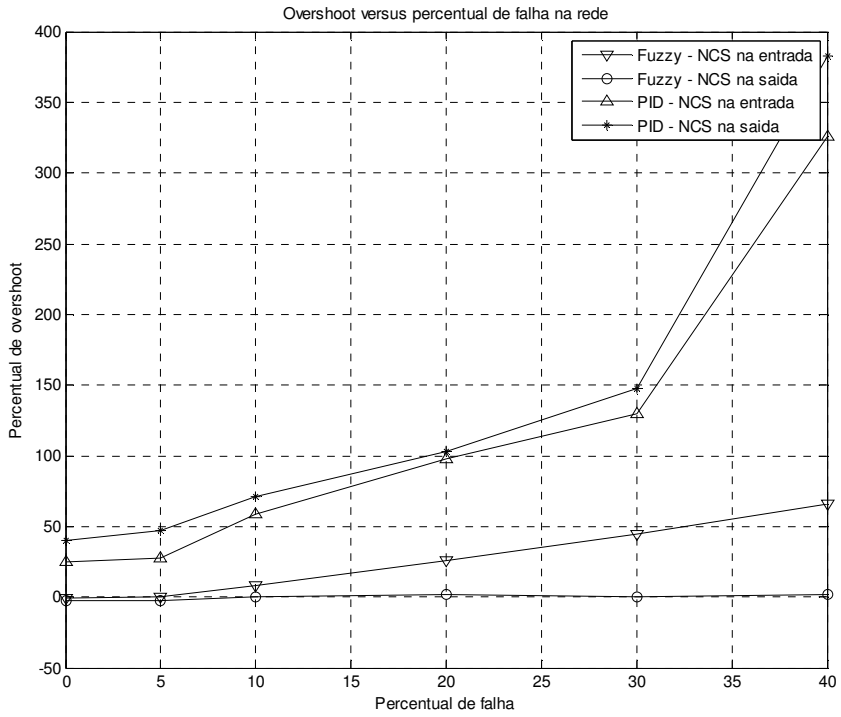


Figura 89 – Variação do *overshoot* em função da perda de pacotes e localização da rede de sensores num NCS fuzzy

Neste estudo pode-se verificar que o controlador fuzzy, apesar de apresentar um tempo de estabilidade superior ao encontrado com o controlador PID, possui uma maior robustez no *overshoot*.

7. CONCLUSÕES E TRABALHOS FUTUROS

7.1 Conclusões

Esta dissertação apresentou um estudo de um controlador fuzzy, implementado com um microcontrolador dsPIC, no qual foi utilizado um protótipo funcional. Para viabilizar este estudo utilizou-se um ambiente Simulink e a ferramenta PIL (*processor in loop*), onde o protótipo foi inserido na malha de controle do Simulink através de uma interface de comunicação serial/USB desenvolvida para este fim.

Além disso, apresentou um estudo do comportamento de uma malha de controle que utiliza uma rede de sensores e um controlador fuzzy como fonte de tomada de decisão.

A implementação dsPIC apresentou resultados bastante satisfatórios e o microcontrolador dsPIC utilizado (30F4011) demonstrou ser uma ótima opção para desenvolvimento, pois o compilador C, disponibilizado gratuitamente pelo fabricante, facilitou bastante o desenvolvimento das rotinas.

Concluiu-se que sistemas que utilizam rede de sensores na malha de controle apresentam melhores resultados quando utilizam um controlador fuzzy, excetuando os sistemas que possuem pólo negativo, os quais poderão ser objeto de estudo para um aprimoramento do método de configuração do controlador fuzzy.

Concluiu-se também que o controlador fuzzy apresenta características bastante robustas e é menos sensível a variações das partes que compõe a malha de controle. A robustez do controlador fuzzy, em face à perda de pacotes, atrasos e localização da rede de sensores na malha de controle, pode ser comprovada pelos resultados obtidos nos cenários apresentados.

Por fim, foi verificado que o controlador fuzzy se apresentou como uma excelente opção para substituir o controlador PID quando se pretende introduzir uma rede de sensores (com ou sem fio) numa malha de controle já existente.

7.2 Trabalhos futuros

Como uma opção de trabalho futuro, pode-se estudar o ajuste dos ganhos do controlador fuzzy em função das perdas dos pacotes, quantificando-se o tempo entre os pacotes válidos.

Os estudos de falhas na comunicação também merecem um aprofundamento, pois apesar de tentarmos cobrir uma ampla gama de cenários, cada deles apresenta características próprias, as quais necessitam de uma solução dedicada para o melhor comportamento do sistema.

Por fim também se pode estudar o comportamento em outros tipos de malhas de controle, como por exemplo, em sistemas de controle preditivo.

REFERÊNCIAS

AMARAL, José Franco Machado. *Síntese de Sistemas Fuzzy por Computação Evolucionária*. Tese (Doutorado em Engenharia Eletrônica) - Pontifícia Universidade Católica do Rio de Janeiro, p. 60-78, 2003.

AGRICULTURA DE PRECISÃO. *World Wide Web - Universidade Católica Dom Bosco do Mato Grosso do Sul*. Disponível em: <http://www.agriculturadeprecisao.org.br/>. Acesso em: Jun. de 2009.

BORDIM, Jacir L.; NAKANO, Koji. *Fundamental Protocols to Gather Information in Wireless Sensor Networks*. Editora CRC Press, 2005.

CAMPOS, Mário Massa; SAITO, Kaku. *Sistemas Inteligentes em Controle e Automação de Processos*. Editora Ciência Moderna, p. 5-16, 2004.

CASTILLO, J.; PÉREZ, J. L.; PAREDES, S. A. *Design of an Analog Computer for for Fuzzy Processes*. Instrumentation and Development, vol. 3 nº. 8, 1997.

COSTA, Alessandra; GLORIA, Alessandro de; FARABOSCHI, Paolo; PAGNI, Andrea; RIZZOTTO, Gianguido. *Hardware Solutions for Fuzzy Control*. Proceedings of IEEE, vol. 83, nº 3, 1995.

DAGA, Leonardo. *World Wide Web – RS232 Blockset for use with Simulink*. Disponível em: <http://leonardodaga.insyde.it/index.htm>. Acesso em: Jan/2010.

DIGI. *World Wide Web – DIGI Homepage*. Disponível em: <http://www.digi.com>. Acesso em: 2008.

DORF, Richard C.; BISHOP, Robert H. *Sistemas de Controle Modernos*. Editora LTC, 2009.

ECOBEE. *World Wide Web – Ecobee Homepage*. Disponível em: <http://www.ecobee.com/>. Acesso em: 2007.

EMBABI, S. H. K.; QUAN X.; OKI, N., MANJREKAR, A.; SINENCIO, Sanchez E. *A Current-Mode based Field-Programmable Analog Array for Signal 24Processing Applications*. Analog Integrated Circuits and Signal Processing, nº 17, p. 125-142, 1998.

FTDI. *World Wide Web – FTDI Homepage*. Disponível em: <http://www.ftdi.com>. Acesso em: 2008.

GARCIA, André Luís Jorge. *Implementação Eletrônica de Sistemas Fuzzy*. Dissertação (Mestrado em Engenharia Eletrônica) - Universidade do Estado do Rio de Janeiro, 2009.

JANTZEN, Jan. *Tuning of Fuzzy PID Controllers*. Technical University of Denmark , Report 98-H 871, 1998.

JUNIOR, Cairo L. Nascimento; YONEYAWA, Takashi. *Inteligência Artificial em Controle e Automação*. Editora Edgard Blücher Ltda., 2000.

KARASAKAL, Onur; YESIL, Engin; GÜZELKAYA, Müjde, EKSIN, Ibraim. *Implementation of a New Self-Tuning Fuzzy PID Controller on PLC*. Turk J Elec Engin, p. 182-195, 2005.

KHEMAPECH, I.; MILLER, A. DUNCAN, I. *A Survey of Transmission Power Control in Wireless Sensor Networks*. 8th Annual Postgraduate Symposium on The Convergence of Telecommunications, Networking and Broadcasting (PGNet 2007), p.15-20, June 2007.

LIAN, Feng-Li; MOYNE, James R., TILBURY, Dawn M. *Performance Evaluation of Control Networks: Ethernet, ControlNet, and DeviceNet*. IEEE Control Systems Maganize, 1999.

MATIAKIS, Tilemachos; HIRCHE, Sandra; BUSS, Martin. *Netowrked Controls Systems with Time-Varing Delay – Stability through Input-Output Transformation*. Automatisierungstecknik 56, 2008.

MESQUITA, Leonardo; JUNIOR, Galdenoro Botura; ROCHA, Paloma Maria Silva. *Implementação de controladores lógicos difusos usando FPGA ou ASIC*. Rev. ciênc. exatas, Taubaté, v. 12, n. 2, p. 9-17, 2006.

NISE, Norman S. *Engenharia de Sistemas de Controle*. Editora LTC, 2002.

OGATA, Katsuhiko. *Engenharia de Controle Moderno*. Editora Prentice Hall, 2005.

OHLIN, Martin; HENRIKSSON, Dan; CERVIN, Anton. *TRUETIME 1.5 — Reference Manual*. Department of Automatic Control, Lund University, 2007.

PEREIRA, Marluce R.; AMORIM, Cláudio L. de; CASTRO, Maria Clícia Stelling de. *Tutorial sobre Rede de Sensores*. Cadernos do IME/UERJ, 1998).

ROGERCOM. *World Wide Web – Rogercom – RCOM MESHBEE*. Disponível em: <http://www.zigbee.org>. Acesso em: 2007.

SADJADI, Babak Azimi. *Stability of Network Control Systems in the Presence of Packet Losses*. Proceedings of IEEE Conference on Decision and Control, 2003.

SONG, Jianping; MOK, Aloysius K.; CHEN, Deji; NIXON, Mark; BLEVINS, Terry; WOJSZNIS, Willy. *Improving PID Control with Unreliable Communicatios*. ISA EXPO 2006.

STOJMENOVIC, Ivan. *Handbook of Sensor Networks – Algorithms and Architectures*. Ed. John Wiley & Sons, 2005.

TANSCHEIT, Ricardo. *Sistemas Fuzzy*. DEE-PUC-Rio, 1999.

TENG, Lanzhi; WEN, Peng; XIANG, Wei. *Model-Based Networked Control System Stability Based on Packet Drop Distributions*. 10th International Conference on Control, Automation, Robotics and Vision (ICARCV 2008), p.17-20 Dec 2008, Hanoi, Vietnam.

TOROGHI, Masoud Khaksar; FANAEI, Mohammad Ali; SHAHROKI, Mohammad. *Fuzzy Self-tuning PID Control of an Unstable Continuous Stirred Tank Reactor*. The 6th International Chemical Engineering Congress & Exhibition, 2009.

VERHAPPEN, Ian. *Process Control Security - Industry's New Challenge*. Foundation Fieldbus End Users Council Australia Inc., 2004.

WEBER, Leo; KLEIN, Pedro Antonio Trierweiler. *Aplicação da Lógica Fuzzy em Software e Hardware*: Editora ULBRA, 2003.

ZENITH. *From Lazy Bones to RedEye: A Brief History of the TV Remote*. Disponível em <http://morecontrol.com/2009/08/lazy-bones-to-redeye-a-brief-history-of-the-tv-remote/>. Acesso em 15 Jun. 2010.

ZIGBEE. *World Wide Web – Zigbee Alliance Homepage*. Disponível em: <http://www.zigbee.org/>. Acesso em: 2007.

APENDICE A – Especificações técnicas dos módulos Xbee

Performance

- Rendimento da Potência de saída: 60 mW (18 dBm), 100 mW EIRP;
- Alcance em ambientes internos/zonas urbanas: 100m;
- Alcance de RF em linha visível para ambientes externos: 1,6Km;
- Sensibilidade do receptor: -100 dBm (1% PER);
- Freqüência de operação: ISM 2.4 GHz;
- Taxa de dados de RF: 250.000 bps;
- Taxa de dados da Interface (Data Rate): 115.200 bps;

Alimentação

- Tensão de alimentação: 2.8 à 3.4v;
- Corrente de transmissão (típico): 215 mA @ 3.3 V;
- Corrente de Recepção (típico): 55 mA @ 3.3 V;
- Corrente de Power-down Sleep: <10 µA;

Propriedades físicas

- Dimensões: (2.438cm **x** 3.294cm**);**
- Peso: 0.10 oz (3g);
- Temperatura de operação: -40 to 85º C (industrial);
- Opções de antena: Conector U.FL RF, Chip ou Chicote (whip);

Rede

- Tipo de espalhamento espectral: DSSS (Direct Sequence Spread Spectrum);
- Manipulação de erro: Retransmite novamente (Retries) & reconhecimento (acknowledgements);
- Topologia de Rede: Peer-to-peer(Par-a-par), ponto-a-ponto, ponto-a-multiponto e malha;
- Endereçamento: 65.000 endereços de rede disponíveis para cada canal;
- Opções de filtros: PAN ID, canais e endereços;
- Criptografia: 128-bit AES;
- Número de canais selecionáveis via software: 12 canais de seqüência direta;

Geral

- Faixa de freqüência: 2.4000 2.4835 GHz;