UNIVERSIDADE ESTADUAL DE CAMPINAS
Faculdade de Engenharia Elétrica e de Computação

APLICAÇÃO DE TÉCNICAS DE FUSÃO DE SENSORES NO MONITORAMENTO DE AMBIENTES

Autor: Rogério Esteves Salustiano
Orientador: Prof. Dr. Carlos Alberto dos Reis Filho

Dissertação de Mestrado apresentada à Faculdade de Engenharia Elétrica e de Computação como parte dos requisitos para obtenção do título de Mestre em Engenharia Elétrica. Área de concentração: **Eletrônica, Microeletrônica e Optoeletrônica**.

Banca Examinadora

Sérgio Santos Mühlen, Dr. ..... DEB/FEEC/UNICAMP
Guido Costa Souza de Araújo, Ph.D. ..... DSC/IC/UNICAMP

Campinas, SP
Janeiro/2006

Este exemplar corresponde à redação final da tese defendida por: ROGÉRIO ESTEVES SALUSTIANO
e aprovada pela Comissão
Julgada em 16 / 01 / 2006
Orientador

FICHA CATALOGRÁFICA ELABORADA PELA
BIBLIOTECA DA ÁREA DE ENGENHARIA - BAE - UNICAMP

Salustiano, Rogério Esteves
Sa38a Aplicação de técnicas de fusão de sensores no monitoramento de ambientes / Rogério Esteves Salustiano.--Campinas, SP: [s.n.], 2006.

Orientador: Carlos Alberto dos Reis Filho
Dissertação (Mestrado) - Universidade Estadual de Campinas, Faculdade de Engenharia Elétrica e de Computação.

1. Monitoramento ambiental. 2. Fusão de dados de múltiplos sensores. I. Reis Filho, Carlos Alberto. II. Universidade Estadual de Campinas. Faculdade de Engenharia Elétrica e de Computação. III. Título.

Titulo em Inglês: Application of sensor fusion techniques in the environmental monitory
Palavras-chave em Inglês: Sensor Fusion, Consensus Sensors, Environmental monitory, Multisensor data fusion
Área de concentração: Semicondutores, Instrumentação e Fotônica
Titulação: Mestre em Engenharia Elétrica
Banca examinadora: Sérgio Santos Mühlen e Guido Costa Souza de Araújo
Data da defesa: 16/01/2006

Resumo

Este trabalho propõe um sistema computacional no qual são aplicadas técnicas de Fusão de Sensores no monitoramento de ambientes. O sistema proposto permite a utilização e incorporação de diversos tipos de dados, incluindo imagens, sons e números em diferentes bases. Dentre os diversos algoritmos pertinentes a um sistema como este, foram implementados os de Sensores em Consenso que visam a combinação de dados de uma mesma natureza. O sistema proposto é suficientemente flexível, permitindo a inclusão de novos tipos de dados e os correspondentes algoritmos que os processem. Todo o processo de recebimento dos dados produzidos pelos sensores, configuração e visualização dos resultados é realizado através da Internet.

Palavras-chave: Fusão de Sensores, Sensores em Consenso, Fusão de Sensores em Níveis, Monitoramento de Ambientes, Múltiplos sensores.

Abstract

This work proposes a computer system in which Sensor Fusion techniques are applied to monitoring the environment. The proposed system allows the use and incorporation of different data types, including images, sounds and numbers in different bases. Among the existing algorithms that pertain to a system like this, those, which aim to combine data of the same nature, called Consensus Sensors, have been particularly implemented. The proposed system is flexible enough and allows the inclusion of new data types and their corresponding algorithms. The whole process of receiving the data produced by the sensors, configuration of produced results as well as their visualization is performed through the Internet.

Keywords: Sensor Fusion, Consensus Sensors, Sensor Fusion Levels, Environmental Monitory, Multisensor.

“O Senhor é meu pastor, nada me faltará”.
Salmo 22

“Não podes ensinar nada a um homem; podes apenas ajudá-lo a encontrar as respostas dentro dele mesmo".
Galileu Galilei

“Aprender sem pensar é inútil. Pensar sem aprender, perigoso”.
Confúcio

“Nossas dúvidas são traiçoeiras e com freqüência nos fazem perder o bem que poderíamos conseguir sem o medo de tentar”.
William Shakespeare

Aos meus pais,
Pedro Martins Salustiano e
Maria Albertina Esteves Salustiano,
pelo suporte material e principalmente
pelo insubstituível suporte imaterial.

Agradecimentos

Primeiramente à Deus, por ter me dado a vida e a capacidade de raciocínio.

Ao professor Reis pela oportunidade e confiança depositada em mim e no meu trabalho, além do incentivo a novas descobertas no decorrer das minhas pesquisas.

Aos colegas da UNICAMP, em especial aos amigos-pesquisadores do LPM/FEEC. Destaco os nomes de Ricardo Pureza Coimbra, Donato Manzan Júnior e João Paulo Cerquinho Cajueiro pela ajuda na parte experimental e questões de Eletrônica.

Aos professores da FEEC e do Instituto de Computação (IC), especialmente à profa. Thelma Cecília Chiossi (DSI/IC) pela ajuda no projeto e documentação do Banco de Dados.

À UNICAMP, meu segundo lar, e à CAPES pelo auxílio financeiro.

Índice Geral

Índice de Figuras

Índice de Tabelas

Glossário

ASCII	Sigla de *American Standard Code for Information Interchange*. Trata-se de uma padronização da indústria de computadores para a representação de números, letras, pontuação e outros caracteres. Cada caractere é representado sob a forma de código binário de 8 bits.
AST	Sigla de *Abstract Syntax Tree* (Árvore sintática abstrata). É uma estrutura na forma de árvore cujos nós representam operações ou contém os operandos das operações. As AST são utilizadas na construção de compiladores e representam a estrutura interna de um programa.
API	Sigla de *Application Programming Interface*. É um conjunto de rotinas e padrões estabelecidos por um software para a utilização de suas funcionalidades. Em geral, a API é composta por uma série de funções acessíveis somente por programação.
Applet	*Applets* são pequenos aplicativos escritos em **JAVA** que são executados no ***browser*** da máquina do cliente.
AU	Formato de arquivos de som do Sistema Operacional **UNIX**.
Base 64	É uma forma de conversão de uma seqüência de bits em 64 caracteres imprimíveis da tabela **ASCII**. São utilizados os caracteres A-Z, a-z, 0-9, / e + na representação dos bits, sendo esses dois últimos variável de sistema para sistema. Essa conversão permite o envio de arquivos binários no formato texto, utilizado principalmente em e-mails e no padrão **XML**.
BGP	Sigla de *Bizantine General Problem* (Problema dos Generais Bizantinos). Ver capítulo 4.

BLOB	Sigla de *Binary Large Object* (Objeto Binário Grande). É uma coleção de dados binários armazenados em uma única entidade em um **SGBD**. Geralmente, os BLOBs são utilizados para armazenar imagens, áudio e outros objetos multimídia.
Browser	Também conhecido como *web browser* ou navegador, é um programa que permite os usuários interagirem com documentos **HTML**.
Certificado digital	Também conhecido como ID Digital, permite a codificação e assinatura de mensagens para assegurar a sua autenticidade, integridade e inviolabilidade. Os certificados digitais são muito utilizados em páginas comerciais da **Internet**.
Criptografia	É o estudo dos princípios e das técnicas pelas quais a informação pode ser transformada da sua forma original para outra ilegível.
Chave primária	Termo utilizado em Banco de Dados para identificar o atributo (ou coluna) de uma entidade (ou tabela) que identifica de forma única uma instância da entidade (ou linha da tabela).
Chave estrangeira	Termo utilizado em Banco de Dados para identificar que a coluna de uma tabela deve conter elementos que pertencem a uma coluna que é definida como **chave primária** de outra tabela.
DOM	Sigla de *Document Object Model*. É uma **API** destinada a acesso e modificação de documentos **XML**.
Driver	É um programa que possibilita a comunicação entre os Sistemas Operacionais e dispositivos periféricos ligados a um computador ou entre um software e um Banco de Dados.
DTD	Sigla de *Document Type Definition*. Trata-se de um texto estruturado que define a gramática de um documento **XML**. Nele são definidos quais elementos devem estar presentes no **XML**, quais os seus atributos e suas hierarquias.
Garbage Collector	Também conhecido como Coletor de Lixo. Trata-se de um mecanismo de limpeza de memória, feita pelo programa que está em execução, que identifica dados que não são mais utilizados pelo programa e que podem ser eliminados, disponibilizando, assim, mais memória para o programa. Não são todas as linguagens de programação que implementam coletores de lixo. **JAVA** é uma linguagem que implementa esse recurso.

GIF	Sigla de *Graphics Interchange Format* (Formato para Intercâmbio de Gráficos). É um formato de imagem de mapa de bits muito utilizado na **Internet**. Uma imagem GIF pode ser estática ou conter uma animação (GIF animado).
GPIB	Sigla de *General Purpose Instrumentation Bus* (Barramento de Instrumentação de Propósito Geral). Padronização IEEE 408, trata-se de um barramento utilizado para comunicação entre instrumentos de medição como multímetros e microcomputadores.
Hipertexto	Trata-se de um sistema para a visualização de informações cujos documentos contêm referências internas (chamadas de *hiperlinks* ou, simplesmente, *links*) para outros documentos ou para outras partes do documento.
HTML	Sigla de *HyperText Markup Language*. Trata-se de uma linguagem textual de marcação utilizada para produzir páginas na **Internet**. Esses textos podem ser interpretados pelos ***browsers*** para exibir páginas **WWW**.
HTTP	Sigla de *HyperText Transfer Protocol*. É um protocolo da camada de aplicação **OSI** utilizado para transferência de dados na *World Wide Web* (**WWW**).
HTTPS	Sigla de *HyperText Transfer Protocol Secure*. É uma implementação do protocolo **HTTP** sobre uma camada adicional de segurança **SSL** ou **TSL**. Essa implementação permite uma conexão que realize a **criptografia** dos dados transmitidos e verifique a autenticidade do servidor e do cliente através de um **certificado digital**.
Internet	É uma rede de computadores em escala mundial que utiliza o protocolo **IP** como forma de comunicação.
ISO	Sigla de *International Organization for Standardization* (Organização Internacional para Padronização). É uma organização que aprova padrões internacionais em todos os campos técnicos.
IP	É um acrônimo para a expressão inglesa *Internet Protocol* (ou Protocolo da Internet). Trata-se de um protocolo de comunicação utilizado entre duas máquinas em rede para encaminhamento de dados ou mensagens.
Jakarta	É o nome de um projeto colaborativo que oferece uma diversidade de soluções **JAVA** e faz parte da ASF (*The Apache Software Foundation*).

JAR	É a extensão do nome de um arquivo compactado contendo um conjunto de classes **JAVA**. É utilizado para armazenar classes compiladas e metadados associados que podem constituir um programa.
JAVA	É uma linguagem de programação orientada a objetos desenvolvida pela *Sun Microsystems*.
JAVADOC	É uma ferramenta da *Sun Microsystems* que gera uma documentação **API** no formato **HTML** através de comentários estruturados escritos nos programas **JAVA**.
Javascript	É uma linguagem de programação que executa nos ***browsers*** dos clientes e é interpretada em tempo de execução. *Javascript* não é uma linguagem de programação que possa ser compilada.
JSP	Sigla de *Java Server Pages*. É uma tecnologia para o desenvolvimento de aplicações de **Internet**. Ela reúne o potencial da interface **HTML** com a tecnologia **JAVA**, permitindo o acesso a Banco de Dados, arquivos texto e o recebimento e processamento de informações vindas em formulários.
JVM	Sigla de *Java Virtual Machine* (Máquina Virtual Java). É um programa que carrega e executa os aplicativos **JAVA**, convertendo o código binário dos arquivos .class em código executável da máquina.
JPG ou JPEG	Acrônimo de *Joint Photographic Experts Group***.** É um formato de compressão de imagem especialmente utilizado para comprimir imagens fotográficas. Trata-se de uma compressão com perdas, sendo que as perdas são proporcionais ao fator de compressão utilizado para gerar o arquivo.
LabVIEW	Acrônimo de "**Lab**oratory **V**irtual **I**nstrument **E**ngineering **W**orkbench". É uma linguagem de programação gráfica pertencente à *National Instruments*.
MER	Modelo entidade-relacionamento (em inglês ERM – *entity-relacionship model*). Trata-se de um modelo utilizado para o modelamento de Banco de Dados.
Multi-thread	Característica dos sistemas operacionais modernos que permite repartir a utilização do processador entre várias tarefas, dando a idéia que elas executam simultaneamente.

MySQL	É um sistema de gerenciamento de Bando de Dados (**SGBD**) que utiliza a linguagem **SQL** como interface. É um software livre.
OSI	Sigla de *Open Systems Interconnection* (Interconexão de Sistemas Abertos). Trata-se de um conjunto de padrões **ISO** relativo à comunicação de dados.
PNG	Sigla de *Portable Network Graphics*. É um formato de dados utilizado para imagens com o intuito de substituir o formato **GIF**. Este formato é recomendado pela **W3C**. Permite a compressão de imagens sem perda de qualidade.
SAX	Sigla de *Simple API for XML*. É um mecanismo de acesso a documentos XML baseado no acesso serial dos dados e execução de eventos.
SGBD	Sigla de Sistema de Gerenciamento de Banco de Dados.
SGML	Sigla de *Stantard Generalized Markup Language*. É uma metalinguagem através da qual se pode definir outras linguagens de marcação para documentos texto.
Socket	Um *socket* (ou soquete) é utilizado para estabelecer ligações bidirecionais entre dois programas que rodam em redes de computadores. Um *socket* é determinado por um número **IP** e uma porta disponível na interface de rede do computador.
String	Seqüência de caracteres alfanuméricos, ou seqüência de caracteres imprimíveis da tabela **ASCII**.
SQL	Sigla de *Structure Query Language* (Linguagem de Questões Estruturadas). É uma linguagem de pesquisa declarativa para acesso a dados em Banco de Dados relacionais. Além de oferecer comandos de pesquisa, a linguagem oferece comandos de inclusão de dados, criação de tabelas e exclusão de dados.
SSL	Sigla de *Secure Sockets Layer* (Camada de Socket Segura). Protocolo desenvolvido pela Netscape™ para promover o tráfego segura de dados na **Internet**.
Tag	Dentro do contexto de um documento **XML**, uma *tag* (etiqueta) é o identificador de um elemento. Em <NOME>Rogério</NOME>, NOME é uma *tag*.

TLS — Sigla de *Transport Layer* Security (Camada de Transporte Segura). Corresponde à Camada de Transporte segura do protocolo **SSL**.

TCP — Sigla de *Transmission Control Protocol* (Protocolo de Controle de Transmissão). Trata-se do protocolo sob o qual se assenta a comunicação na **Internet**. O TCP realiza a verificação se os dados foram enviados de forma correta, na seqüência correta e sem erros.

Tomcat — É um servidor **HTTP** ou **HTTPS** de aplicações **JAVA**. É distribuído como software livre e desenvolvido em código aberto. Oficialmente ele é endossado pela *Sun Microsystems* e é referência para a tecnologia **JSP**.

Token — Seqüência de caracteres que pode ser tratada como uma unidade dentro da gramática de uma linguagem de programação. Uma palavra (seqüência de caracteres), um número, um caractere de marcação são exemplos de *tokens*.

UNIX — É um Sistema Operacional portátil, multitarefa e multiusuário originalmente criado por um grupo de programadores da AT&T. Hoje este Sistema Operacional é mantido e comercializado pela *Sun Microsystems*.

XLS — Formato (e extensão do nome) de arquivos cujo conteúdo é composto de planilhas eletrônicas e gráficos.

XML — Sigla de *eXtensible Markup Language* (Linguagem Extensiva de Marcação). É uma recomendação da **W3C** para gerar linguagens de marcação para necessidades especiais. É um subtipo de **SGML**.

URL — Sigla de *Universal Resource Locator* (Localizador Uniforme de Recursos). É o endereço eletrônico de um recurso (uma pasta, uma impressora, uma página na **Internet**, etc). Uma URL tem a seguinte estrutura: `protocolo://máquina/caminho/recurso`.

W3C — É um acrônimo para *World Wide Web Consortium* (Consórcio da rede mundial de computadores). É um consórcio de empresas de tecnologia com o objetivo de desenvolver protocolos comuns e fóruns abertos para a evolução da **WWW**.

Web — Termo que significa “teia”. É utilizado para designar de forma curta a sigla WWW.

WWW — Sigla de **Wold Wide Web**. É uma rede de computadores na **Internet** que fornece informação em **hipertexto**. Para se visualizar as informações, utiliza-se um software chamado navegador (***browser***).

1

Introdução

"É impossível para um homem aprender aquilo que ele acha que já sabe".
Epíteto

A utilização de sensores no controle de sistemas e monitoramento de ambientes vem crescendo a cada dia. Em busca do aumento da confiabilidade e da integridade dos dados sensoriados, a utilização de mais de um sensor é freqüentemente realizada.

O processo de combinação dos dados provindos de múltiplos sensores de mesma natureza ou de naturezas diferentes é denominado **Fusão de Sensores** (*Sensor Fusion*). Seu principal objetivo é fornecer aos sistemas dados de maior qualidade, permitindo assim reduzir as falhas nos processos decisórios.

Quando combinados de maneira adequada, os dados provenientes de sensores com exatidão e resolução limitadas podem fornecer aos sistemas informações mais aperfeiçoadas, aumento da exatidão e da resolução no valor resultante da combinação dos dados.

A utilização de sensores de mesma natureza operando de forma redundante permite a redução dos dados incorretos (suspeitos de discordarem do

conjunto) e amplia a exatidão do valor final. Sensores de naturezas diferentes ou sensores de mesma natureza utilizados de forma cooperativa permitem a obtenção de informações impossíveis de serem obtidas através de sensores isolados (como a visão 3D, por exemplo).

Os mecanismos de Fusão de Sensores envolvem diferentes áreas da Computação e da Engenharia. Estrutura de dados, operações matemáticas e algébricas, processamento de imagens, modelos estatísticos, inteligência artificial e outros campos do conhecimento são utilizados em conjunto.

Em alguns sistemas, um sensor de maior exatidão e custo elevado pode ser substituído por diversos sensores de menor custo, permitindo estabelecer um grau de precisão entre eles, apesar de haver perdas na exatidão. Além disso, a falha de um sensor não compromete o funcionamento do sistema visto que os demais sensores garantem o fornecimento do dado do sistema que está sendo sensoriado.

As técnicas de Fusão de Sensores podem ser utilizadas em uma grande variedade de aplicações. Determinação da trajetória de robôs, reconhecimento de alvos, sensoriamento remoto, monitoramento ambiental, controle de tráfego, diagnóstico médico e navegação aérea são alguns exemplos no qual a Fusão de Sensores pode ser utilizada como método de melhoria do desempenho e da qualidade dos resultados.

Além do desenvolvimento de mecanismos de Fusão de Sensores, isto é, a forma com que os dados dos sensores são combinados, um outro desafio que aparece é a determinação de estruturas de dados capazes de armazenar os dados de forma eficiente e permitir uma fácil manipulação dos diferentes tipos de dados: imagens, sons, números, etc.

Este trabalho propõe um sistema computacional que permite o monitoramento de ambientes utilizando mecanismos de Fusão de Sensores.

O sistema desenvolvido é suficientemente flexível, permitindo a manipulação de dados de naturezas diversas e fornece uma estrutura de programação que facilita a incorporação de diferentes algoritmos de Fusão de Sensores. Todo o processo de recebimento dos dados produzidos pelos sensores, configuração e visualização dos resultados é realizado através da Internet.

Uma atenção especial é dada aos algoritmos de Sensores em Consenso dentro do conjunto de algoritmos de Fusão de Sensores, por se tratar da combinação de dados de mesma natureza com o objetivo de se obter um valor consensual a partir de um conjunto de medidas. Esses algoritmos são úteis em aplicações que dispõem de diversos sensores de mesma natureza, mas que são de baixa exatidão ou são suscetíveis a falhas, buscando, através da aplicação dos algoritmos, melhorar a qualidade do valor final e descartar os valores incorretos dentro do conjunto de medidas. Contudo, se todos os sensores (ou a maioria deles) fornecerem medidas erradas ao sistema, o resultado será errado, independentemente do algoritmo utilizado.

Este trabalho está organizado em 6 capítulos. O capítulo 2 apresenta os conceitos da Fusão de Sensores e analisa as aplicações das técnicas de combinação de dados de sensores em diferentes áreas. O capítulo 3 descreve o sistema computacional desenvolvido com o propósito de monitoramento de ambientes. Nesse capítulo, são descritos todos os componentes do sistema e a comunicação entre eles, assim como é feito um detalhamento das configurações necessárias para o funcionamento do sistema.

O capítulo 4 descreve os algoritmos de Sensores em Consenso e traça um perfil comparativo entre eles. O capítulo 5 descreve duas aplicações experimentais do sistema: monitoramento da temperatura de um ambiente através de 10 sensores de temperatura e monitoramento de uma planta através de 4 sensores de naturezas diferentes (um sensor de luminosidade, um sensor de temperatura, um sensor de umidade e uma câmera de vídeo).

No capítulo 6 são apresentadas as conclusões a respeito do Sistema de Monitoramento de Ambientes desenvolvido e das técnicas de Fusão de Sensores.

2

Fusão de Sensores

"What is "the unit of the senses"? Simply stated, it is the thesis that the senses have a lot in common. Different senses often assist one another in the perception of objects and events. Different senses often share common phenomenological attributes. Different senses often obey similar laws, often employ similar or common mechanisms".[1]

Lawrence Marks - psicofísico

Este capítulo descreve o processo de Fusão de Sensores, relacionando suas vantagens e limitações. São abordadas algumas configurações dos sensores no ambiente e o local onde a fusão dos dados dos sensores pode ocorrer.

Um resumo sobre os termos relacionados à fusão se propõe a definir a Fusão de Sensores e os seus principais objetivos, apesar das divergências e inconsistências encontradas na literatura [ELM02].

Dentre as técnicas de fusão de sensores estudados, destaca-se o modelo de fusão em níveis por ser o único a classificar os algoritmos de acordo com os tipos de dados a serem combinados.

[1] "O que é "a unidade dos sentidos"? Simplesmente formalizando, é a tese de que os sentidos têm muito em comum. Sentidos diferentes freqüentemente se auxiliam na percepção de objetos e eventos. Sentidos diferentes freqüentemente compartilham atributos fenomenológicos comuns. Sentidos diferentes freqüentemente obedecem a leis similares, freqüentemente utilizam mecanismos similares ou comuns".

A parte final deste capítulo se destina a exemplos de aplicações das técnicas de Fusão de Sensores, tentando encaixar os algoritmos propostos dentro do modelo de níveis de fusão aos problemas que podem ser beneficiados utilizando este processo.

2.1 – Termos relacionados à Fusão

A literatura sobre os processos de Fusão de Sensores é recente (década de 1980) e ainda existe muita imprecisão e usos divergentes dos termos relacionados à fusão (ou combinação) de dados de sensores.

Os termos **Fusão de Dados** (*Data Fusion*), **Fusão de Sensores** (*Sensor Fusion*), **Integração de Múltiplos Sensores** (*Multi-sensor Integration*), **Fusão de (Dados de) Múltiplos Sensores** (*Multi-sensor (Data) Fusion*) e **Fusão da Informação** (*Information Fusion*) são utilizados para se referir a uma variedade de técnicas, tecnologias, sistemas e aplicações que visam à combinação de informações originárias de múltiplas fontes de dados.

Em busca de uma padronização nos termos, diferentes autores buscaram definir e classificar em suas obras os termos relacionados ao processo de combinação de dados e às técnicas envolvidas nesses processos. Apesar de ainda não haver um consenso nas definições, segue uma breve coletânea de definições e principais áreas nas quais os termos são empregados.

O termo **Fusão de Dados** (*Data Fusion*) é definido em [KLE99] como um processo de vários estágios que trata da detecção, associação, correlação, estimação e combinação de dados de diversas fontes. Contudo, em alguns modelos de fusão, o termo é empregado como um conjunto de métodos e ferramentas com o objetivo de obter informações de melhor qualidade através da fusão de dados de baixa qualidade, sendo que a definição específica de “melhor qualidade” depende da aplicação [ELM02].

A utilização do termo **Fusão de Dados** segundo a definição de [KLE99] é realizada principalmente na área de Geociências e Sensoriamento Remoto, na área de processamento de imagens e em diversos artigos relacionados ao rastreamento e localização de objetos em movimento.

Em diversos artigos e livros, o termo **Fusão de Dados** é expandido para **Fusão de Dados de Múltiplos Sensores** e é definido como a tecnologia interessada em combinar os dados de diversos sensores (Figura 2.1) com o intuito de estabelecer inferências sobre eventos físicos, atividades ou situações. Em [BRO98] e [JOS99], o termo **Fusão de Dados de Múltiplos Sensores** é expresso da forma acima, acrescentado da idéia de que os dados combinados dos múltiplos sensores devem ser apresentados e organizados de uma forma única.

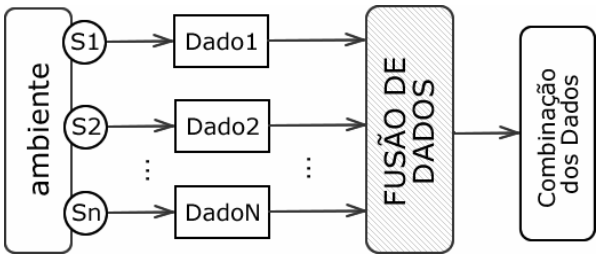


Fig. 2.1 – Processo de Fusão de Dados de Múltiplos Sensores. Adaptação de [JOS99]

O termo **Fusão da Informação** é bem amplo e refere-se à fusão de qualquer tipo de informação (Figura 2.2). Ele está relacionado à minimização do volume de informações e à integração delas, cobrindo todos os aspectos no campo da fusão [ELM02]. Esse termo engloba teoria, técnicas e ferramentas destinadas à exploração das informações obtidas de múltiplas fontes (sensores, banco de dados, informações fornecidas por homens, etc).

Fig. 2.2 – Processo de Fusão da Informação.

O processo de **Integração de Múltiplos Sensores** ou **Integração de Sensores** é definido como a utilização de múltiplos sensores em um sistema e comumente está vinculado a um uso sinergético dos mesmos [JOS99]. A definição em

[LUO95] para esse termo é bem semelhante, referindo-se ao uso sinergético da informação provinda de múltiplos dispositivos sensores com o objetivo de ajudar na realização de uma tarefa do sistema; é o meio com que múltiplos sensores são integrados na operação de uma máquina inteligente ou sistema. A literatura relacionada ao processo de **Integração de Múltiplos Sensores** divide o processo em três categorias: planejamento, arquitetura e fusão dos sensores (Figura 2.3).

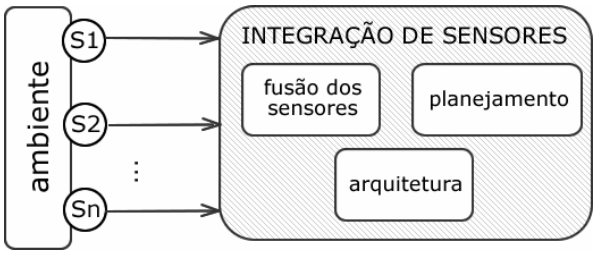


Fig. 2.3 – Processo de Integração de Múltiplos Sensores. Adaptação de [JOS99]

O planejamento refere-se ao modo com que é feita a aquisição dos dados sensoriados; a arquitetura refere-se à organização do controle e fluxo de dados em um sistema de múltiplos sensores; e a fusão dos sensores está relacionada ao processo de combinação dos dados dos sensores. O foco da **Integração de Sensores** está nos aspectos do sistema, como a modularização, escalonamento, coordenação, robustez e comunicação de dados pelos dispositivos sensores.

Fusão de Sensores ou **Fusão de Múltiplos Sensores** é o termo mais utilizado na literatura técnica (principalmente nos artigos) e refere-se à combinação de dados sensoriados ou de dados derivados de dados sensoriados de tal maneira que a informação resultante é de alguma maneira melhor (qualitativamente) do que seria possível se os dados de origem fossem usados individualmente. O processo de Fusão de Sensores tem como objetivo criar uma representação, isto é, um modelo, do ambiente real dentro do sistema computacional (Figura 2.4).

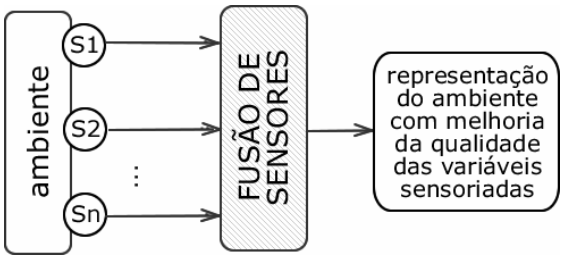


Fig. 2.4 – Processo de Fusão de Sensores.

A definição de **Fusão de Sensores** dada por [LUO95] não restringe que os dados sejam produzidos em múltiplos sensores, apenas diz que os dados devem ser combinados de forma a melhorar a sua interpretação. A técnica de **Fusão de Sensores** pode ser realizada com um único sensor efetuando-se leitura de dados seqüenciais ao longo do tempo e posteriormente esses dados são combinados. Os sensores utilizados na percepção do ambiente podem ser de mesma natureza (vários sensores de temperatura, por exemplo) ou de naturezas diferentes (sensores de temperatura, sensores umidade e sensores de pressão, por exemplo).

Embora a distinção dos termos ainda apareça de forma confusa, a Tabela 2.1 tenta resumi-las conforme suas utilizações na literatura.

Tabela 2.1 – Definições dos termos relacionados à fusão

Termo	Definição	Objetivo
Fusão de Dados Fusão de Dados de Múltiplos Sensores	Métodos e ferramentas utilizados na combinação de dados de baixa qualidade	Melhoria na qualidade da informação gerada pela combinação dos dados sensoriados
Fusão da Informação	Termo amplo. Refere-se à fusão de qualquer tipo de dado	Minimização do volume de dados e integração
Integração de Sensores	Uso sinergético das informações sensoriadas por diferentes dispositivos sensores (envolve fusão dos dados, planejamento e arquitetura do sistema)	Potencializar a realização de tarefas do sistema
Fusão de Sensores Fusão de Múltiplos Sensores	Processo de combinação dos dados obtidos por sensores de mesma natureza ou de naturezas diferentes.	Obter uma representação (modelo) do ambiente sensoriado com melhoria na qualidade das variáveis sensoriadas

Dentro do contexto desta dissertação, o termo Fusão de Sensores será utilizado segundo a definição abaixo, resultado do estudo da utilização do termo na literatura e da comparação com os demais termos relacionados à fusão.

> *A Fusão de Sensores refere-se ao processo que autonomamente reúne e combina as observações de múltiplos sensores de mesma natureza ou de naturezas diferentes com o objetivo de fornecer aos sistemas de controle e monitoramento uma melhor percepção do ambiente, ou seja, dados e informações mais refinadas.*

Um dos grandes ganhos proporcionados pelas técnicas de Fusão de Sensores é fazer com que algumas informações emirjam indiretamente a partir dos dados dos sensores, sem a necessidade da utilização de sensores específicos, possivelmente mais caros. Um outro ponto de relevância nas técnicas de fusão está na melhoria da qualidade dos dados sensoriados que, conseqüentemente, permite o desenvolvimento de sistemas de decisão mais confiáveis.

2.2 – Vantagens no uso da Fusão de Sensores

A proposta da Fusão de Sensores é obter um sistema que forneça informações úteis sobre alguma característica de interesse do ambiente, assim como permitir ter uma visão global do ambiente sensoriado. As vantagens desse tipo de integração é que as informações obtidas são mais refinadas e mais próximas das reais, podendo apresentar características que são impossíveis de serem obtidas com apenas um sensor, assim como num intervalo mais curto de tempo e a menor custo. Esses atributos são conseguidos através das seguintes características que o processo de fusão possui:

Tolerância a falha: O uso de múltiplos sensores aumenta a tolerância a falhas do sistema. Mesmo que alguns sensores falhem, a informação obtida pelos

elementos sensores que não falharam é garantida. Se um sensor falhar, outros providenciam uma informação similar.

Confiabilidade: A medição de um sensor é confirmada pela medição de outros sensores que efetuam a leitura de um mesmo ambiente sob um mesmo domínio.

Sinergia: Os dados de um único sensor podem ser insuficientes ou incompletos, mas sensores complementares que "observam" diferentes aspectos do ambiente podem ser usados para gerar suposições que seriam impossíveis de serem feitas com a utilização de apenas um elemento sensor.

Expansão da cobertura espacial e temporal: Enquanto alguns sensores fazem a tarefa de obter informações do ambiente, outros podem executar suas tarefas internas de preparo para obtenção dos dados e/ou cobrirem outras áreas do ambiente. As suposições feitas pelo sensoriamento realizado por múltiplos sensores podem ser mais convenientes porque os sensores podem trabalhar de forma integrada para o preenchimento de escalas de tempo de leitura que seriam impossíveis se serem preenchidas com as limitações dos sensores empregados.

Redução na ambigüidade e na incerteza: A união de informações reduz a ambigüidade na interpretação do valor medido. Valores dispersos dentro de um conjunto de medidas podem ser descartados diante os critérios adotados.

Robustez contra interferência: Com o aumento da dimensão do espaço medido, isto é, utilizando sensores que atuam em diferentes faixas do espectro de energia, o sistema fica menos vulnerável à interferência.

Aumento da resolução: quando múltiplas medidas independentes da mesma propriedade são fundidas, a resolução do valor resultante é melhor do que a medida realizada por um sensor único.

Custo: Pode ser menos custosa (em termos de tempo de processamento, recursos computacionais ou materiais) a utilização de múltiplos sensores. Ao invés de se utilizar um sensor de alta exatidão e custo elevado, a utilização de vários sensores de menor custo pode ser satisfatória em alguns sistemas. Isto se aplica, naturalmente, àqueles casos em que a redução da exatidão seja tolerada. Em contrapartida, pode haver aumento da precisão.

2.3 – Sensores e atuadores

Todo sistema automatizado deve ser capaz de reagir a mudanças no seu ambiente. Os sensores e atuadores (comumente denominados transdutores) são a interface entre o sistema computacional e o ambiente caótico ao qual o sistema tem o objetivo de “perceber” e “atuar”.

Para que se estabeleçam métodos de mapeamento do mundo real é preciso separar o dispositivo sensor físico da tarefa que ele executa. O sensor analisado sob o ponto de vista da tarefa que ele executa é chamado de **sensor abstrato** ou **sensor lógico** [BRO98].

Essa abstração da entidade sensora permite estabelecer limites teóricos para o sensor, sem a necessidade da implementação de detalhes particulares para cada um dos vários tipos de sensores existentes e suas tecnologias e formas de aquisição dos dados.

Deve-se ressaltar que sensores abstratos não necessariamente correspondem a sensores concretos. Leituras de diferentes sensores físicos podem se tornar uma única leitura de um sensor abstrato.

Nessa dissertação todos os sensores serão considerados abstratos, visto que a importância no processo de combinação dos dados sensoriados está na especificação da natureza do dado do sensor e o que se espera dele e não como ele é composto e de que forma interage com o ambiente.

2.4 – Configurações dos sensores

A construção de uma rede de sensores pode ser realizada utilizando-se sensores de tecnologias e naturezas diferentes ou ainda utilizando-se sensores iguais que se auxiliam na observação do ambiente.

As configurações dos sensores em uma rede podem ser divididas nas seguintes categorias: **sensores complementares**, **sensores competitivos** e **sensores cooperativos**. Essa divisão de configurações está baseada no papel de cada sensor em relação aos outros no ambiente sensoriado e, para que todo tipo de configuração pertença a uma categoria de rede, criou-se uma quarta configuração denominada **sensores independentes** que engloba as redes de múltiplos sensores que não pertençam às três outras categorias mencionadas [BRO98].

Essas redes podem estar configuradas em diferentes níveis de complexidade e arquiteturas, valendo-se de diferentes modos de comunicação e necessitando cada qual de complexidade computacional específica até o momento da fusão dos dados. Para cada configuração de rede é necessário traçar distinções e escolher métodos específicos para tratamento e combinação dos dados.

Segue a descrição das possíveis configurações entre os sensores.

Complementar: Uma configuração de sensores é dita complementar se os sensores não dependem diretamente um do outro, mas podem ser combinados de maneira a fornecerem um diagnóstico mais completo do fenômeno que está sendo observado. Um exemplo seria a utilização de alguns radares de tal maneira que cada um cubra uma região geográfica diferente. Os dados são combinados para fornecer uma visão equivalente de toda a região.

Competitiva: Os sensores são configurados de forma competitiva quando cada sensor capta medidas independentes da mesma propriedade. São possíveis duas configurações competitivas: a fusão de medidas de diferentes sensores ou a fusão de medidas de um único sensor obtidas em instantes diferentes. A

configuração competitiva de sensores é também chamada de configuração **redundante**. Essa configuração é importante no caso de tolerância a falha, visando robustez e confiabilidade no sistema. Um exemplo da utilização de sensores competitivos é o emprego de mais de um sensor de temperatura na determinação dessa grandeza em um determinado ambiente.

Cooperativa: Uma rede de sensores cooperativa usa a informação proveniente de diversos sensores independentes para derivar uma informação que não poderia ser obtida através de um único sensor. A visão estereoscópica é um exemplo de configuração cooperativa de sensores ópticos, através da combinação de duas imagens bidimensionais de duas câmeras posicionadas em pontos levemente diferentes, que resulta em uma imagem tridimensional da cena observada. Um outro exemplo importante de rede de sensores cooperativa é a triangulação das informações de latitude e longitude fornecidas pelos satélites e combinadas no aparelho de GPS. Em contraste com a configuração competitiva, a configuração cooperativa geralmente diminui a exatidão e a confiança no sistema.

Independente: Uma rede de sensores independente é aquela que não se enquadra nas três categorias discutidas acima. Um exemplo para essa configuração é uma rede de radares que visa sensoriar a posição de aeronaves e ao mesmo tempo obter dados da temperatura. Não seria nada realista encontrar uma relação entre a temperatura e a posição da aeronave, mas as duas variáveis possuem informações que representam um único ambiente que está sendo observado.

Essas categorias não são mutuamente exclusivas. Muitas aplicações apresentam a utilização de mais de uma dessas categorias, configurando uma arquitetura híbrida. Um exemplo desse tipo de arranjo é a utilização de câmeras de vídeo no monitoramento de uma área: existem regiões cobertas por duas ou mais

câmeras onde a informação é competitiva ou cooperativa e existem áreas observadas por apenas uma câmera onde a informação é complementar.

A figura 2.5 ilustra os diferentes modos de configurações de sensores e os resultados (ou conquistas) alcançados através da associação.

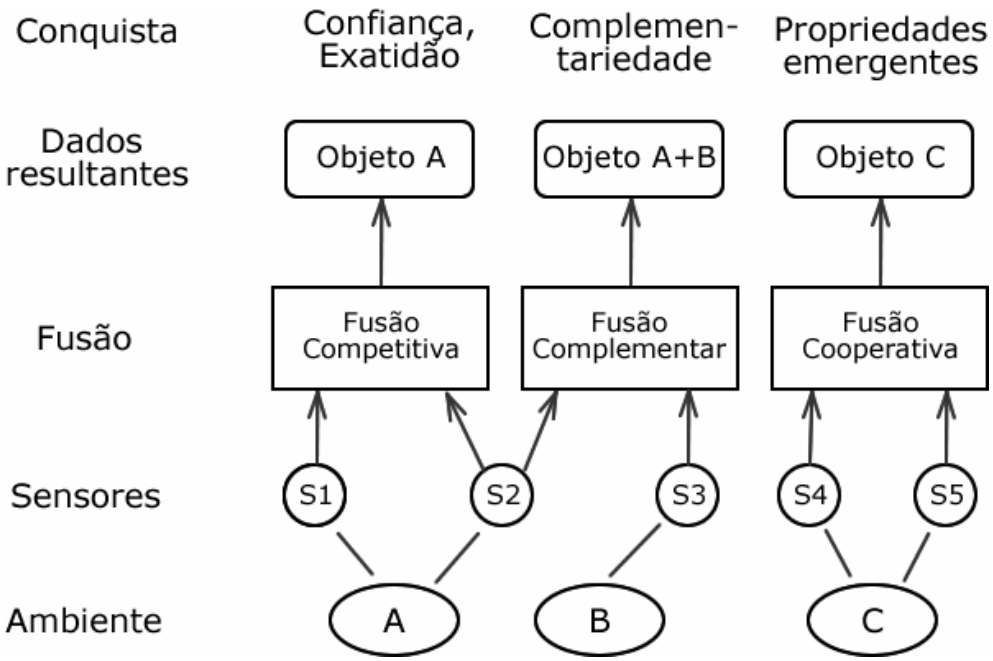


Fig. 2.5 – Fusão competitiva, complementar e cooperativa. Adaptação de [ELM02]

A categoria mais difícil para se realizar a fusão dos dados é a competitiva [BRO 98]. Devido à redundância dos dados, aparecem inconsistências entre as leituras dos sensores e o sistema deve evitar realizar suposições incertas baseado na coincidência da similaridade dos dados. Quando a combinação dos dados é realizada de forma apropriada, a fusão dos dados aumenta a robustez do sistema, mas se feita incorretamente ou sem critérios bem elaborados e com objetivos definidos, a fusão de dados competitivos pode ter conseqüências desastrosas.

2.5 – Limitações e erros no processo de Fusão de Sensores

O processo de Fusão de Sensores apresenta algumas limitações na sua utilização, muitas vezes gerando inconsistências nos resultados e encontrando algumas características do ambiente sensoriado que o sistema não é capaz de "entender".

Alguns algoritmos de Fusão de Sensores que registram dados de mesma natureza necessitam de dados de mais de três sensores para que a fusão de fato possa ocorrer. Essa limitação impede a utilização desses algoritmos em ambientes onde há pouca disponibilidade de sensores.

Alguns erros no processo de fusão são ressaltados em [LUO95] e têm o intuito de auxiliar na configuração e programação dos sistemas. Grande parte desses erros pode ser eliminada pela escolha adequada dos algoritmos de Fusão de Sensores ou contornados através de políticas de manutenção e monitoramento dos sistemas.

Erro no processo de fusão: O maior problema na integração e fusão de informações redundantes de múltiplos sensores é a determinação de que a informação de cada sensor se refere à mesma característica do ambiente. Esse problema é conhecido como correspondência e associação de dados.

Erro na informação do sensor: O erro na informação do sensor é comumente causado por um processo de ruído aleatório que pode ser adequadamente modelado como uma distribuição de probabilidade. O ruído geralmente é assumido como não tendo correlação com o tempo e o espaço, não possui distribuição Gaussiana e é independente. Se o ruído apresentasse algum tipo de correlação ele poderia ser tratado com algum tipo de filtro.

Erro na operação do sistema: Quando um erro ocorre durante a operação de um sistema devido ao efeito de acoplamento entre os componentes do sistema, é possível assumir que a medição do sensor é independente da calibração. Em ambientes bem conhecidos, a calibração dos sensores não

é um problema difícil, mas quando os múltiplos sensores são utilizados em ambientes desconhecidos, a calibração dos sensores pode não ser possível de ser executada. A solução para esse problema pode ser a criação de uma base de conhecimento detalhada de cada tipo de sensor de maneira que o sistema seja capaz de se autocalibrar.

2.6 – Fusão centralizada *versus* Fusão nos sensores

Existem duas arquiteturas possíveis no que diz respeito ao local onde a combinação (fusão) dos dados pode ser realizada: no sensor (*Sensor-level Fusion*) ou na Central do Sistema (*Central-level Fusion*).

A Central do Sistema sempre executa o processamento mais refinado (o qual necessita de maior quantidade de memória e que gera informações com maior nível de abstração), mas podem existir arquiteturas que executem o processamento no próprio sensor, processando sua própria medição e recebendo dados de outros sensores configurados na forma de rede.

A fusão dos dados no sensor requer um gasto de energia adicional no circuito sensor, necessitando uma maior complexidade no seu desenvolvimento e a rede formada pelos sensores precisa lidar com os problemas típicos de um sistema distribuído.

A Tabela 2.2 resume as vantagens do uso da Fusão no sensor e da Fusão centralizada.

Existem sistemas chamados híbridos, que apresentam tanto a Fusão no sensor como a Fusão centralizada, aproveitando as vantagens de ambas as implementações. Contudo, a decisão para a escolha do local onde ocorre a fusão dos dados é uma característica típica de cada sistema.

Tabela 2.2 – Comparação das vantagens da Fusão no sensor e da Fusão centralizada [KLE99]

Fusão no sensor	Fusão centralizada
Possibilidade de uma plataforma de processamento distribuída, sem a necessidade de grande capacidade de processamento na central do sistema.	Maior integração entre os sensores, uma vez que o dado bruto pode ser trabalhado de forma conjunta com outros dados provindos de outros sensores do sistema, sem a necessidade de gasto de comunicação entre os sensores.
Otimização no processamento, visto que o mesmo é específico para cada sensor.	Redução no peso, volume, potência e custo de produção dos sensores.
Flexibilidade no número e tipos de sensores com a possibilidade de adição, remoção e substituição de sensores sem necessidade de modificação na estrutura do algoritmo de fusão centralizada.	Aumento na confiança do processamento no sistema, uma vez que o processamento estando no sensor fica sujeito às condições físicas e mecânicas que podem deteriorar o resultado do processamento.

2.7 – Modelo de Fusão de Sensores em Níveis

A utilização do processo de Fusão de Sensores depende do sistema do qual ele fará parte: depende das características dos sensores e das tarefas específicas que cada sistema em particular necessita.

Alguns modelos de Fusão de Sensores foram propostos com aplicações específicas como o **Modelo de Fusão JDL** (*US Joint Directors of Laboratories*) proposto em 1985 pelo Departamento de Defesa Americano (DoD) [KLE99] para aplicações militares e civis, o **Modelo de Fusão Boyd** (ou **OODA** – *Observer, Orientate, Decide and Act*) proposto em 1987 [BED99], o **Modelo de Fusão em Cascata** proposto em 1997 [BED99] utilizado pelo departamento de defesa da Grã-Bretanha, o **Modelo de Fusão LAAS** (*Laboratoire d´Analyse et d´Architecture des Systèmes*) proposto em 1998 [ALA98] e aplicado na área de robótica, o **Modelo de Fusão Omnibus** apresentado em 1999 [ELM02] e o **Modelo de fusão *Time-Triggered*** desenvolvido em 2001 voltado para sistemas robóticos [ELM01].

Cada um desses modelos possui vantagens e desvantagens na sua estrutura e concepção, mas compartilham a desvantagem de não possuírem uma

classificação dos algoritmos de fusão no que diz respeito à natureza (ou tipo) do dado de entrada e do dado de saída da fusão, ou seja, os modelos acima tratam os dados de forma genérica, sem haver uma distinção entre dado numérico, imagem, objeto, etc. Além disso, a maioria desses modelos está mais preocupada com a organização e controle do fluxo dos dados no sistema do que com o bloco responsável pela fusão dos dados dos sensores.

O **Modelo de Fusão em Níveis** descrito no trabalho de Ren C. Luo e Michael G. Kay [LUO 90] e [LUO95] apresenta uma clara divisão dos processos de fusão de acordo com a natureza do dado a ser combinado, além de classificar os tipos de fusão que podem ocorrer com os dados.

Nesse modelo, os dados dos sensores podem ser combinados em quatro níveis de representação (ou níveis de fusão): **nível de sinal** (*signal-level*), **nível de *pixel*** (*pixel-level*), **nível de característica** (*feature-level*) e **nível de símbolo** (*symbol-level*). Dependendo da necessidade de cada sistema e do grau de similaridade dos sensores utilizados, um desses níveis é utilizado.

Muitos dos sistemas sensores são capazes de realizar a combinação dos dados em mais de um nível de fusão. Além da classificação levar em conta o tipo de representação, ela considera também como a informação sensoriada está modelada, o grau de registro necessário para a fusão, os métodos utilizados para a fusão e o modo pelo qual o processo de fusão melhora a qualidade da informação provida ao sistema. A Tabela 2.3 traça uma comparação dessas considerações com os níveis de fusão.

2.7.1 – Fusão no nível de sinal

A fusão no nível de sinal refere-se à combinação dos sinais provenientes de um grupo de sensores de mesma natureza (temperatura ou pressão ou altitude, coordenadas geográficas ou luminosidade e assim por diante) ou de naturezas diferentes, mas que haja uma relação de dependência entre as variáveis (posição e velocidade, por exemplo).

Tabela 2.3 – Comparação entre os Níveis de Fusão

	Nível de Sinal	Nível de *Pixel*	Nível de Característica	Nível de Símbolo
Tipo de informação sensoriada	Sinais uni- ou multi-dimensionais	Imagens	Características extraídas de sinais e imagens	Símbolos que possuem características capazes de representar decisões
Grau de complexidade na representação da informação	Baixo	Baixo a Médio	Médio	Alto
Métodos de Fusão	• Média Aritmética • Média Ponderada • Filtragem Kalman • Sensores em Consenso	• Filtros Lógicos • Morfologia Matemática • Álgebra de Imagens • Recozimento Simulado	• Transformações Geométricas • Segmentação • Inteligência Artificial	• Estimativa Bayesiana • Teoria dos Conjuntos • Lógica Booleana • Lógica Fuzzy
Melhoria obtida pela fusão	Redução na variância esperada	Melhoria no desempenho do processamento da imagem	Redução no processamento e aumento da precisão nas medidas	Aumento da veracidade ou da probabilidade dos valores

O objetivo da fusão no nível de sinal é gerar um sinal que geralmente é da mesma natureza que os sinais originais, mas que seja mais representativo do ambiente. A melhoria na qualidade no dado resultante em comparação aos valores registrados pelos sensores individualmente vai depender do algoritmo que se está utilizando para a fusão.

Os dados obtidos pelos sensores necessitam estar sincronizados para haver uma fusão no nível de sinal eficiente. Caso o sistema de aquisição não seja síncrono, há a possibilidade de registrar o instante da obtenção dos dados juntamente com o valor da medição do sensor para posterior combinação com os demais dados dos outros sensores no mesmo instante.

Dependendo do tipo de integração de sensores que se esteja utilizando no sistema, podem-se utilizar técnicas distribuídas de relógios globais para sincronizar as medições realizadas nos sensores [LUO90].

Um dos modos mais simples de se realizar a fusão dos sinais provenientes de sensores de mesma natureza é aplicar a **Média Aritmética** aos valores. Contudo, pode-se refinar essa média ponderando-a (**Média Ponderada**) pelo inverso da exatidão (*accuracy*) de cada sensor, dando assim um maior peso aos dados obtidos por sensores mais exatos.

Uma outra forma de se realizar a fusão de sinais é através da **Filtragem Kalman** (publicada em 1960 por R. E. Kalman). A filtragem Kalman consiste em um conjunto de equações matemáticas recursivas que estimam o estado de uma variável (ou de um processo) com o objetivo de minimizar o seu erro quadrático médio [WEL04]. Esse filtro é muito utilizado para estimar o passado, o presente e o futuro das variáveis de um sistema, sendo a aquisição temporal dos dados dos sensores um requisito fundamental.

A Filtragem Kalman vem sido utilizada amplamente em aplicações de localização e reconhecimento de trajetórias nas quais as variáveis combinadas são a posição e a velocidade do objeto que está em movimento. Trata-se, portanto, de uma fusão no nível de sinal com variáveis de natureza diferente, mas que se relacionam entre si.

Um terceiro modo de realizar a fusão no nível de sinal é através de um método denominado **Sensores em Consenso** (*Consensus Sensors*) que visa a eliminação dos dados de alguns sensores que aparentam representar erros diante os valores registrados pelos outros sensores. O resultado desse tipo de fusão é uma estimação matemática da variável sensoriada pelo grupo de sensores de mesma natureza, eliminando os valores suspeitos de divergir dos demais valores registrados no grupo.

Um algoritmo bem elaborado dentro do método de Sensores em Consenso deve possuir critérios de seleção bem elaborados e bem definidos para justificar a exclusão da medida de um ou mais sensores dentro do grupo e também identificar quais dispositivos sensores estão adquirindo dados discordantes para realizar testes e possíveis substituições desses dispositivos. Alguns algoritmos de Sensores em Consenso serão analisados detalhadamente no capítulo 4.

2.7.2 – Fusão no nível de *pixel*

A fusão no nível de *pixel* pode ser utilizada para melhorar a informação associada a cada *pixel* de uma imagem resultante da associação de múltiplas imagens. O aumento da qualidade no resultado da fusão no nível de *pixel* está diretamente relacionado ao ganho resultante da aplicação de técnicas de processamento de imagens (segmentação, restauração, etc) às imagens captadas pelos sensores.

As imagens são obtidas comumente através de sensores ópticos que registram diferentes faixas do espectro visível ou invisível. Pode-se utilizar uma câmera de infravermelho, vermelho visível, verde visível, azul visível ou ainda câmeras compostas (sistema RGB, por exemplo). A partir da combinação das imagens geradas por essas câmeras, que registram imagens de naturezas diferentes, é possível produzir imagens combinadas e extrair características delas. Além de combinar imagens, os algoritmos de fusão no nível de *pixel* podem atuar sobre as imagens de um único sensor com o objetivo de melhorar o contraste, a luminosidade, extrair características ou detectar movimento.

A fusão no nível de *pixel* deve verificar se as imagens registradas pelos sensores estão na mesma resolução, caso contrário há a necessidade de mapear as regiões correspondentes das imagens para haver fusão das mesmas.

O modo mais intuitivo de combinar duas ou mais imagens é a aplicação de **Filtros Lógicos** aos *pixels* das imagens. As operações *E*, *OU*, *NÃO* e *OU-exclusivo* são aplicadas aos *pixels* de forma a fundir uma ou mais imagens em uma única imagem. Através desse processo pode-se obter uma imagem combinada de sensores que registram faixas diferentes do espectro luminoso, aumentar o tamanho da imagem pela junção de imagens complementares e verificar a mudança de uma imagem em relação à imagem anteriormente capturada pelo mesmo sensor.

A **Morfologia Matemática** é um método de fusão no nível de *pixel* que aplica às imagens (ou a um conjunto de *pixels* das imagens) as operações da Teoria dos Conjuntos (união, intersecção, diferença, etc.). O principal objetivo desse tipo de fusão é a detecção de bordas e possivelmente fornecer ao sistema a extração de características presentes nas imagens.

Através da **Álgebra de Imagens** é possível a transformação de um conjunto de *pixels* em objetos geométricos (retângulos, hexágonos, triângulos, etc). Essa é outra técnica de fusão no nível de *pixel* que permite realizar a transformação em imagens de vários sensores e agrupar os objetos geométricos reconhecidos nas imagens em grupos que poderão ser interpretados de alguma maneira pelo sistema.

O **Recozimento Simulado** (*Simulated Annealing*) é uma técnica que processa imagens através da análise de cada *pixel* e sua vizinhança, como se os *pixels* fossem átomos que formam moléculas com os *pixels* vizinhos. O objetivo dessa técnica de fusão no nível de *pixel* é estabelecer relações entre os *pixels* de uma imagem ou entre os *pixels* de diversas imagens.

2.7.3 – Fusão no nível de característica

A fusão no nível de característica pode ser entendida como a extração de feições, similaridades, padrões ou aspectos em destaque a partir dos dados dos sensores. Tipicamente esse tipo de fusão é realizado em imagens (previamente

processadas ou não), mas ela pode ser efetua em outros tipos de dados com o objetivo de proporcionar um significado semântico aos dados sensoriados.

O objetivo final da fusão no nível de característica é potencializar as informações obtidas a partir do dado de um sensor pela análise dos dados de sua vizinhança ou fronteiras. Através desse tipo de análise é possível não somente reconhecer padrões nos dados dos sensores, mas também eliminar possíveis interpretações duvidosas efetuando fusões competitivas nos dados dos sensores.

As características obtidas nesse nível de fusão podem ser divididas em primárias e secundárias. Uma característica primária é caracterizada pela aplicação de regras semânticas aos dados coletados pelos sensores, enquanto uma característica composta (secundária) é originada pela combinação de outras características já conhecidas no sistema. Assim, uma característica composta necessita de dados, informações ou conjunto de regras auxiliares que devem estar disponíveis no sistema.

A **Transformação Geométrica** é uma técnica de fusão no nível de característica. Através da rotação e translação de porções de uma imagem é possível realizar o reconhecimento de padrões e objetos.

A técnica de **Segmentação** realiza a divisão geométrica e de características dos elementos (objetos) de uma imagem com o intuito de combinar as características encontradas com outras características já registradas no sistema, como padrões e formas de objetos.

Uma outra ferramenta muito poderosa para a fusão no nível de característica é a **Inteligência Artificial** que pode auxiliar na tarefa de reconhecimento de padrões tanto de objetos como de letras em uma imagem (processo conhecido como OCR – *Optical Character Recognition*) ou ainda na leitura de códigos de barra.

2.7.4 – Fusão no nível de símbolo

A fusão no nível de símbolo permite que a informação dos múltiplos sensores seja realmente usada de forma conjunta no nível mais alto de abstração dentro do sistema. O nível de símbolo é o único nível em que se pode realizar fusão de informações provenientes de sensores de naturezas diferentes que não tenham nenhuma relação direta entre si ou fusão de regiões diferentes do ambiente.

Os símbolos usados para a fusão podem ter origem tanto nas informações fornecidas pelos sensores como podem fazer uso de informações obtidas *a priori* do modelo do ambiente ou de origens externas ao sistema. O símbolo resultante da informação fundida representa a decisão que o sistema deve tomar a respeito de algum aspecto do ambiente. Em algumas referências, a fusão no nível de símbolo é denominada de **fusão no nível de decisão**.

As ferramentas mais utilizadas nesse nível são a **Estimativa Bayesiana**, a **Teoria dos Conjuntos** aplicada aos símbolos do sistema, a **Lógica Booleana** e a **Lógica Fuzzy**.

A **Estimativa Bayesiana** revela o grau de crença na decisão obtida pela análise e combinação dos símbolos do sistema. Esse método envolve a coleção de evidências que os símbolos associados a alguma hipótese que o sistema tenta provar representam. A hipótese geralmente nunca é totalmente aceita (igual a 1), mas na medida que as evidências fornecidas pelos símbolos se acumulam, o grau de crença aumenta (cresce de 0, ou seja, nenhuma evidência, para 1) e a aceitação da hipótese aumenta.

A **Teoria dos Conjuntos** é um outro método de fusão no nível de símbolos e é muito útil no processo de fusão de símbolos complementares e redundantes (competitivos). Trata-se de uma teoria matemática desenvolvida pelo russo Georg Cantor (1845-1918) que utiliza as operações de união, intersecção, adição e subtração, entre outras, com o objetivo de determinar regiões de similaridade ou discriminação de objetos a partir das características dos símbolos do sistema.

A **Lógica Booleana** é um conjunto de estruturas algébricas que implementam as lógicas básicas *E*, *OU* e *NÃO* nas características dos símbolos do sistema. Esse tipo de fusão no nível de símbolo é útil para verificar se determinada propriedade é verdadeira (valor 1) ou falsa (valor 0), desencadeando uma tomada de decisão pelo sistema.

Uma extensão da Lógica Booleana é a **Lógica Fuzzy** (ou Lógica Difusa) que determina o grau de verdade na análise de uma propriedade, verificando o quanto ela é verdadeira ou falsa. Em outras palavras, a Lógica Fuzzy permite valores

fracionários entre 0 e 1 (falso e verdadeiro, respectivamente), indicando o quanto a propriedade do símbolo é mais próxima da verdade.

2.8 – Exemplos de aplicações da Fusão de Sensores

A utilização da Fusão de Sensores pode ser realizada em diversos cenários. Foram selecionados quatro exemplos para ilustrar o emprego dos níveis de Fusão de Sensores no processo de fusão de sensores e possíveis combinações desses níveis.

O primeiro cenário trata do reconhecimento de objetos, o segundo é um exemplo de controle do conforto térmico de ambientes, o terceiro trata das possibilidades de emprego das técnicas de Fusão de Sensores na área de Geociências, especificamente em Sensoriamento Remoto e o quarto o emprego de Fusão de Sensores no auxílio no diagnóstico médico.

2.8.1 – Reconhecimento de objetos

A figura 2.6 é uma adaptação de um exemplo de aplicação da Fusão de Sensores em Níveis do trabalho de Ren C. Luo e Michael G. Kay [LUO95].

São utilizados cinco sensores com o objetivo do reconhecimento de objetos, no caso, um veículo. Dois radares operando em freqüências diferentes, um sensor de infravermelho, uma câmera de vídeo e um detector de ondas de rádio capaz de identificar os sons característicos de um veículo.

Os dados dos radares são processados e posteriormente é realizada uma fusão no nível de sinal com o objetivo de combinar os dados, levando em consideração as freqüências distintas de cada sensor. Ocorre, portanto, um refinamento no sinal pela combinação dos dados dos dois radares.

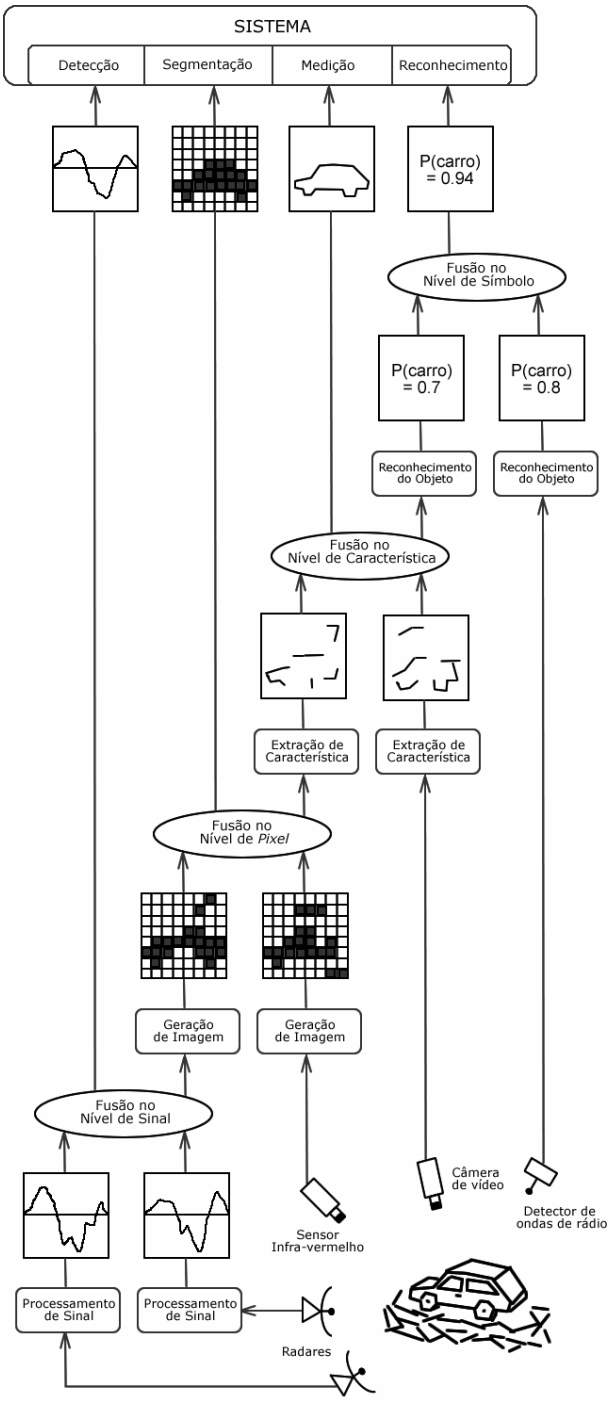


Fig. 2.6 – Níveis de fusão na tarefa de reconhecimento de um carro. Adaptação de [LUO90]

A partir dos dados dos radares também é possível gerar uma imagem do objeto sensoriado que pode ser combinada através de algoritmos de fusão de sensores no nível de *pixel* com a imagem registrada pelo sensor infravermelho. O objetivo dessa fusão é obter uma imagem de melhor qualidade que a imagem gerada pelos dados dos

radares e pelo sensor de infravermelho isoladamente, tratando-se, pois, de uma combinação de duas tecnologias diferentes.

Através dos mecanismos de fusão no nível de característica é possível extrair formas e contornos das imagens obtidas pela câmera de vídeo e pela imagem produzida no processo de fusão no nível de *pixel*. A combinação dos dados neste nível garante uma melhora na determinação das características reconhecidas pelos algoritmos.

O último estágio de fusão do esquema, fusão no nível de símbolo, realiza a combinação dos objetos identificados pela análise das características obtidas pelo processo de fusão no nível de característica e pelo detector de ondas de rádio. Neste estágio final de fusão, o sistema consegue reconhecer o objeto sensoriado de forma mais apurada e completa.

2.8.2 – Conforto térmico

A definição de conforto térmico pode ser feita através de dois pontos de vista: pessoal e ambiental. Considerando apenas o ponto de vista pessoal, o conforto térmico pode ser definido como sendo “uma condição mental que expresse satisfação com o ambiente térmico” [CON05]; considerando o ponto de vista físico, um ambiente confortável é aquele “cujas condições permitam a manutenção da temperatura interna sem a necessidade de serem associados os mecanismos termo-reguladores” [CON05].

Os parâmetros pessoais podem ser atribuídos ao metabolismo do corpo humano e ao vestuário. Quanto aos parâmetros ambientais, eles podem ser mapeados através da temperatura, umidade e velocidade do ar.

Para determinar se um ambiente é confortável ou não, os parâmetros pessoais são descartados, pois não se podem verificar as condições metabólicas e de vestimenta de cada indivíduo presente no ambiente. Entretanto, as variáveis físicas podem ser controladas através de máquinas de condicionamento de temperatura e umidade.

Como um único sensor de temperatura não é capaz de determinar a temperatura geral de um ambiente, mais de um sensor deve ser empregado. Da mesma forma, vários sensores de umidade devem ser espalhados pelo ambiente de forma a

determinar a umidade relativa consensual do ambiente. A velocidade do ar pode ser medida em diferentes pontos, contudo, uma atenção maior deve ser dada às áreas mais próximas aos dutos de saída de ar do sistema de condicionamento por possuírem maior velocidade no fluxo gasoso.

A figura 2.7 ilustra uma possível configuração de um conjunto de sensores de temperatura (T1, T2, T3, T4 e T5), sensores de umidade (U1, U2 e U3) e sensores de velocidade do ar (V1, V2, V3 e V4) monitorando um ambiente fechado. A temperatura e a umidade do ambiente deverão ser condicionadas através dos três condicionadores de ar (CA1, CA2 e CA3).

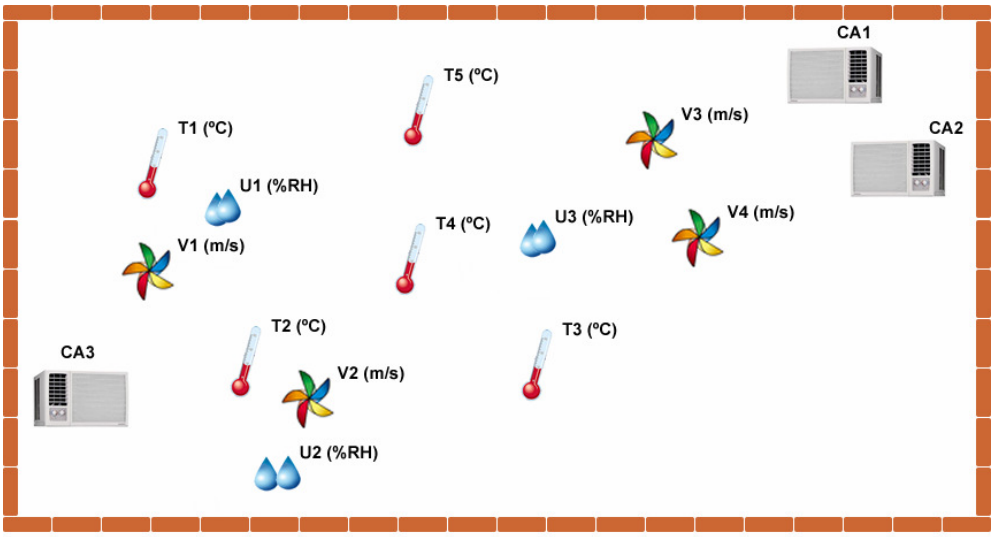


Fig. 2.7 – Ambiente monitorado por sensores de umidade, temperatura e velocidade do ar. T1, T2, T3, T4 e T5 são sensores de temperatura; U1, U2 e U3 são sensores de umidade; V1, V2, V3 e V4 são sensores de velocidade do ar; CA1, CA2 e CA3 são condicionadores de ar.

Diferentes modelos de conforto térmico podem ser adotados para o monitoramento deste sistema, tanto para determinação da posição dos sensores, como para a fusão dos dados. O enfoque do exemplo será na maneira com que os dados são combinados para o controle da potência dos ventiladores, da temperatura e da umidade dos condicionadores de ar, utilizando o posicionamento dos sensores e dos condicionadores de acordo com a figura 2.7. A figura 2.8 mostra um esquema de fusão dos dados para o ambiente da figura 2.7.

A primeira fusão ocorre no nível de sinal entre os sensores de velocidade do ar V3 e V4 (resultando no valor Vs) e entre os sensores de temperatura T3, T4 e T5

(resultando no valor Ts). Os algoritmos utilizados nessa fusão podem pertencer à classe de Sensores em Consenso (estudados no capítulo 4), com o objetivo de encontrar um valor representativo das medidas realizadas pelos sensores.

A temperatura, umidade e velocidade do ar podem determinar diversos índices que estimam o conforto térmico. O índice *Windchill*, o índice de Desconforto [CON05] e o índice de Sensação Térmica [SCH94] relacionam a temperatura e a velocidade do ar. Já o índice de Temperatura Efetiva [CON05] relaciona a temperatura do bulbo seco, a umidade e velocidade do ar como forma de estimar a sensação térmica. Este último será utilizado no mecanismo de fusão dos dados.

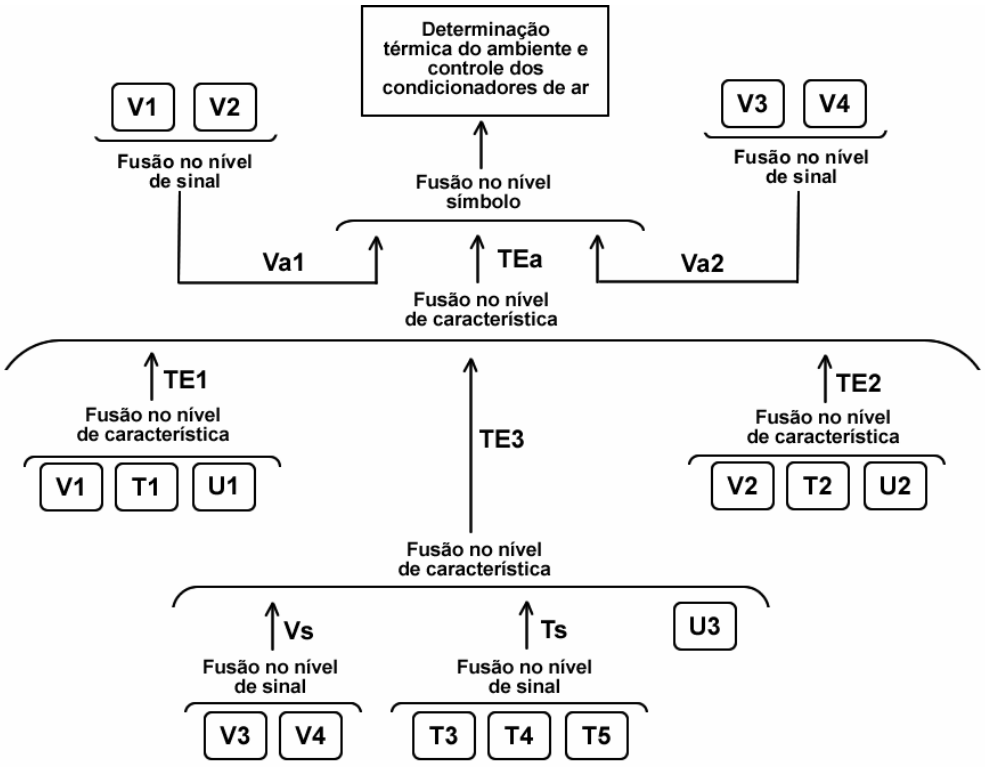


Fig. 2.8 – Diagrama de fusão dos dados dos sensores para o monitoramento do conforto térmico de um ambiente

Através dos grupos de valores dos sensores {V1, T1, U1}, {V2, T2, U2} e {Vs, Ts, V3} pode-se realizar uma fusão no nível de característica obtendo os valores do índice de Temperatura Efetiva dos três grupos (TE1, TE2 e TE3). Cada índice dá um valor semântico (um valor pertencente à escala Índice de Temperatura Efetiva), ou seja, um significado para os dados dos sensores combinados. Esses três valores podem ser

combinados de forma redundante com o objetivo de se obter um único índice no sistema de controle (TEa).

Devido à posição dos condicionadores de ar, a potência dos ventiladores deve ser determinada utilizando os dados dos sensores de velocidade do ar V1 e V2 para o controle da potência do ventilador do condicionador CA3 e os valores de V3 e V4 para o controle de CA1 e CA2. Os dados V1 e V2 são combinados no nível de sinal no valor Va1 e os dados V3 e V4 no valor Va2.

Uma fusão no nível de símbolo pode ser obtida pela aplicação de uma lógica *Fuzzy* aos valores de Va1, Va2 e TEa em níveis como “Ambiente agradável” ou “Ambiente impróprio”, regulando, então os parâmetros dos condicionadores de ar.

Pode-se, adicionalmente, incluir um sensor de luminosidade no ambiente, ampliando ao conforto térmico o conforto visual.

2.8.3 – Sensoriamento Remoto

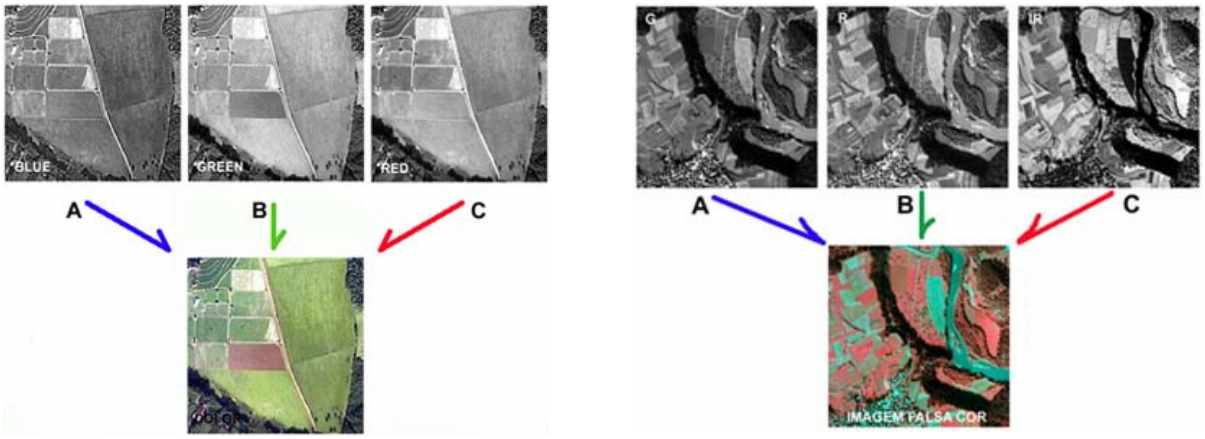
O Sensoriamento Remoto é uma técnica de identificação de feições específicas existentes na superfície terrestre através de imagens de satélites ou imagens obtidas por câmeras instaladas em aeronaves. O objetivo na identificação das feições nas imagens é a elaboração de mapas temáticos e o reconhecimento das características dos terrenos imageados para aplicações ambientais, comerciais e militares, entre outras.

Através da combinação de imagens obtidas por diferentes bandas espectrais é possível obter uma única imagem capaz de revelar informações não “visíveis” nas imagens originais ou realçá-las.

A figura 2.9 mostra duas composições coloridas obtidas pela combinação de três imagens obtidas por satélites. No exemplo (a), foram utilizadas fotografias aéreas com filmes sensíveis à luz azul, verde e vermelha. A combinação resultante das três imagens é uma combinação colorida real, isto é, igual à imagem formada na retina do olho humano e interpretada pelo córtex visual.

A partir das três imagens originais é possível extrair características sobre a vegetação da área sensoriada. Na composição B (verde) a refletância é maior visto que a clorofila absorve menos a radiação verde.

No exemplo (b) da figura 2.9 existe uma combinação chamada de falsa-cor porque as imagens combinadas foram obtidas com os filtros verde (A), vermelho (B) e infravermelho (C). Através dessa combinação outras características podem ser obtidas das imagens, destacando, por exemplo, a cor negra adquirida pelo rio ou a diferenciação de um objeto verde no meio da vegetação. Esta última identificação pode ser feita utilizando as imagens A e C, verificando os pontos que são verdes (em destaque na imagem A), mas que não são destacados no infravermelho (que capta as ondas de calor geradas pela vegetação).



(a) (b)

Fig. 2.9 – Imagens obtidas por sensores remotos. (a) Combinação de três imagens obtidas com filtros azul (A), verde (B) e vermelho (C); (b) Combinação de três imagens obtidas com filtros verde (A), azul (B) e infravermelho (C), caracterizando uma composição falsa-cor.[2]

O processo de Sensoriamento Remoto nada mais é, portanto, do que a combinação de imagens através de algoritmos de Fusão de Sensores no nível de *pixel* e extração de características (tipo de solo, tipo de plantação, área da superfície de um rio, área ocupada pela malha urbana, etc.) através de algoritmos de fusão no nível de características e no nível de símbolos.

[2] Fonte: *http://www.herbario.com.br/fotomicrografia07/inntrodsensoramentoremoto.htm* (acessado em 15/12/2005)

2.8.4 – Auxílio no diagnóstico médico

A área médica utiliza diversos sensores no diagnóstico de pacientes. Sistemas computacionais capazes de realizar pré-diagnósticos podem ser construídos utilizando técnicas específicas de Fusão de Sensores.

A área de estudo na automatização dos diagnósticos médicos ainda gera muita controvérsia. Contudo, cada vez mais são utilizados aparelhos sensores no diagnóstico de pacientes e a integração desses dados tende a ser um processo que poderá beneficiar a área da saúde, auxiliando os médicos na obtenção de diagnósticos mais exatos.

A figura 2.10 ilustra diversos tipos de dados utilizados na área médica: pressão arterial, temperatura corpórea do paciente, imagem de ultra-som e eletrocardiograma.

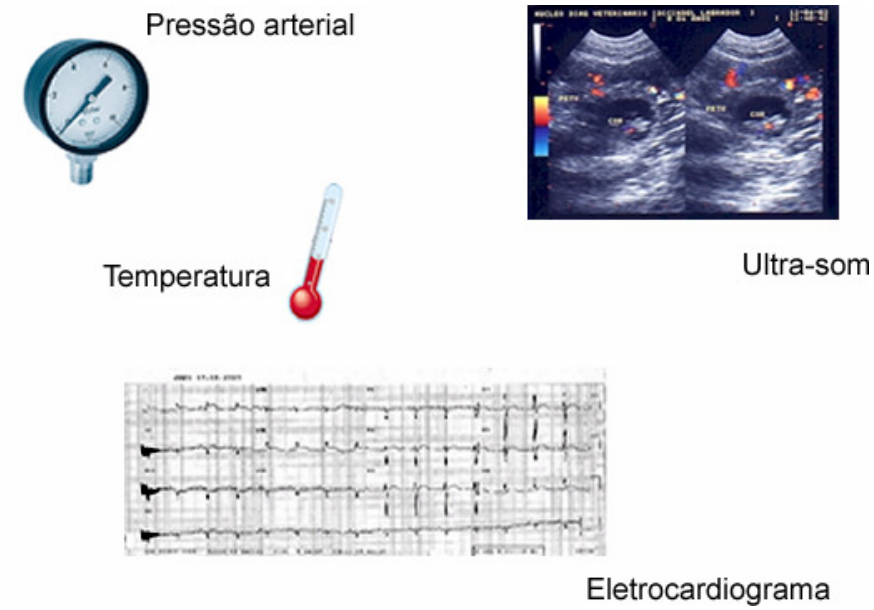


Fig. 2.10 – Dados obtidos por sensores na área médica.

Uma possível fusão dos dados no nível de característica seria a extração de elementos (ou padrões) nas imagens de ultra-som e associá-las a uma análise do sinal resultante do eletrocardiograma. A temperatura do paciente poderia ser medida por diferentes termômetros em diferentes partes do corpo e níveis de febre poderiam ser estabelecidos utilizando a lógica *Fuzzy.* Uma combinação dos níveis de febre com a

pressão sangüínea poderiam determinar uma formulação específica para um medicamento antitérmico.

Resumo

O capítulo 2 cobriu os diferentes aspectos relacionados à Fusão de Sensores, desde os termos relacionados à fusão (ainda não bem definidos na literatura) até algumas aplicações nas áreas de reconhecimento de objetos, sensoriamento remoto e perspectivas na área médica.

Foram citadas as vantagens e as limitações no processo de Fusão de Sensores e as possíveis configurações dos sensores (complementar, competitiva, cooperativa e independente).

Dentre os modelos de Fusão de Sensores, o Modelo de Fusão em Níveis se destaca por classificar as fusões em níveis: nível de sinal, nível de *pixel*, nível de característica e nível de símbolo.

O capítulo 3 descreve o sistema implementado com o objetivo de realizar fusão de dados com sensores de naturezas iguais e diferentes, permitindo a inclusão de novos algoritmos de fusão e novos tipos de dados. O sistema tem como principal objetivo o monitoramento de ambientes.

3

Sistema de Fusão de Sensores para o monitoramento de ambientes

*"*A imaginação é mais importante que o conhecimento*".*
Albert Eistein

A fusão dos dados de sensores pode ser realizada de diferentes formas e em diferentes lugares do sistema. Os algoritmos, os tipos de dados (formatação e natureza), a topologia da rede de sensores e os diferentes modos de combinação das informações precisam ser estruturados para que o processo de fusão execute corretamente e novos sensores e mecanismos de combinação de dados possam ser incorporados ao sistema sem muita dificuldade.

Este capítulo descreve os componentes e as funcionalidades de um sistema de Fusão de Sensores desenvolvido com o objetivo de monitorar diversos ambientes.

Trata-se de um sistema de fusão centralizada capaz de receber, armazenar e processar dados provenientes de sensores de naturezas diversas e que permite a incorporação de novos tipos de dados e novas formas de processamento (combinação) dos dados dos sensores.

A interface de monitoramento e configuração do sistema foi toda implementada em páginas HTML (*HyperText Markup Language*), facilitando o acesso de usuários cadastrados no sistema em qualquer local onde haja disponibilidade de acesso à Internet. Mecanismos de realimentação, isto é, formas de envio de avisos ou comandos para entidades (humanas ou máquinas) capazes de atuar no ambiente sensoriado foram implementados no sistema e podem ser configurados de acordo com a necessidade de cada ambiente.

3.1 – Descrição geral do sistema

O sistema implementado baseia-se no modelo de realimentação sensoriamento-atuação (figura 3.1) adaptado aos algoritmos do Modelo de Fusão em Níveis (discutidos no item 2.7) e à comunicação baseada em troca de mensagens pela Internet.

Neste sistema, o Servidor, um computador (ou uma rede de computadores), recebe os dados dos sensores, armazena-os de forma apropriada para posterior consulta, processa os dados (aplica os algoritmos de fusão nos dados) e aciona mecanismos de decisão que farão com que os atuadores ajam no ambiente.

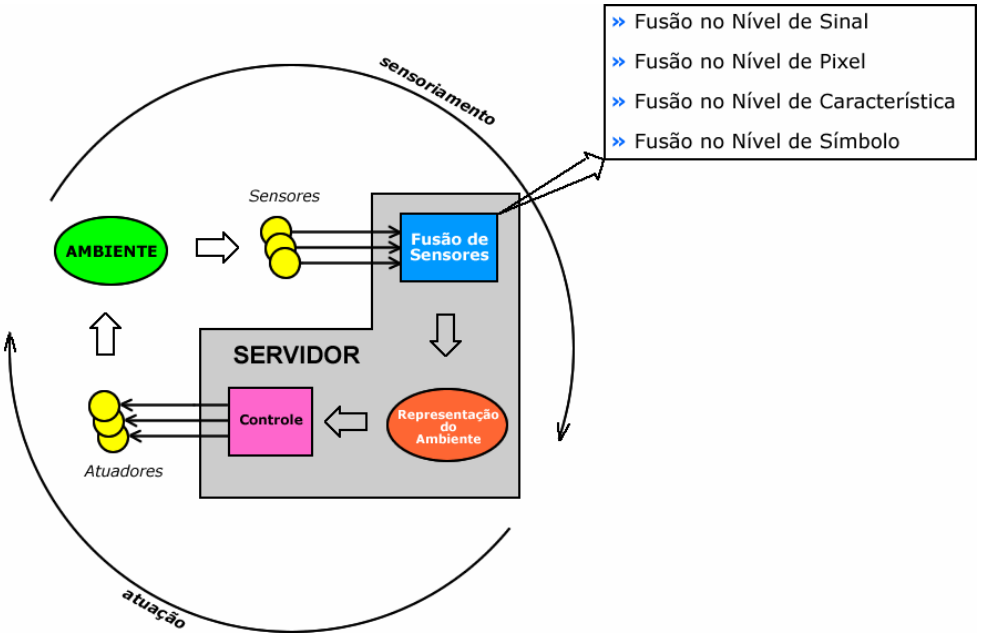


Fig. 3.1 – Modelo de sensoriamento-atuação do Sistema de Fusão de Sensores.

Podem-se diferenciar três estágios no processo a partir do diagrama: o estágio sensor responsável pela captação das informações do ambiente e fornecer os dados para o sistema (sensoriamento), o estágio de combinação dos dados (fusão de sensores e representação do ambiente) e o estágio de controle responsável pela intervenção no ambiente através dos nós atuadores de acordo com o resultado da combinação dos dados.

Contudo, o sistema desenvolvido pode ser utilizado com o intuito de apenas monitorar um ou mais ambientes e nenhum laço de realimentação (execução de tarefas) precisa ser configurado.

3.2 – Detalhamento do sistema

O detalhamento do Sistema de Fusão de Sensores pode ser visualizado na figura 3.2, focando os aspectos de comunicação das partes do sistema.

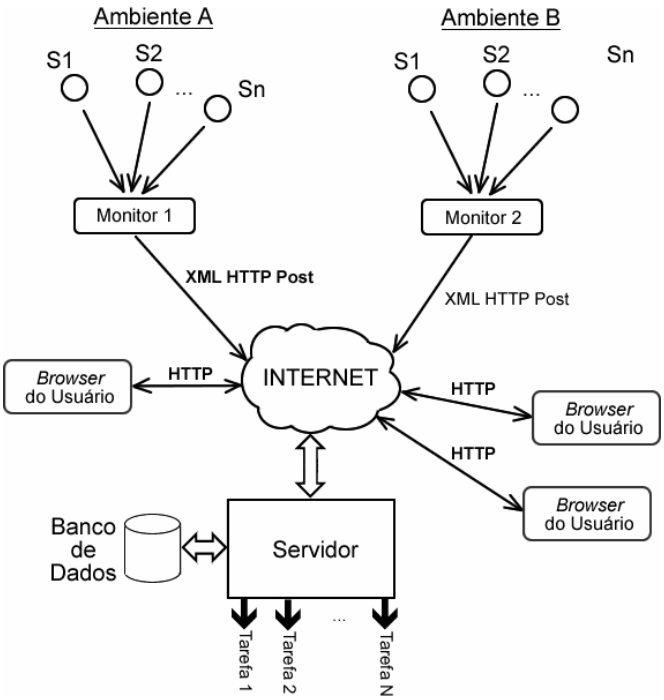


Fig. 3.2 – Arquitetura detalhada do Sistema de Fusão de Sensores: Monitores, Servidor e o Aplicativo (Interface do Usuário) sendo executado nos navegadores dos usuários

Existem três partes do sistema interconectadas pela Internet: os **monitores**, o **servidor** e a **interface do usuário** (que é executada nos navegadores de Internet). Cada uma dessas partes possui componentes auxiliares que auxiliam na realização das suas atividades. Um detalhamento dessas três partes e dos seus componentes será realizado nos próximos itens.

Os **monitores** (concentradores de dados dos sensores) enviam os dados dos sensores de um ambiente em monitoramento para o servidor que armazena (Banco de Dados), processa os dados (procedimentos de Fusão de Sensores) e executa as tarefas relacionadas às decisões do processamento específicas do ambiente sensoriado. A interface do usuário envia requisições de consulta e monitoramento ao servidor que por sua vez fornece os dados requisitados pelos usuários. O cadastramento de ambientes, as configurações do processo de fusão, de consulta e monitoramento também são realizados através da Interface HTML fornecida pelos navegadores.

A troca de mensagens é toda realizada via Internet através do envio de mensagens pelo protocolo de comunicação HTTP (*HyperText Transfer Protocol*) ou HTTPS (*HyperText Transfer Protocol Secure*). Cada ambiente sensoriado deve possuir um **monitor** que concentra os dados dos sensores e os envia para o servidor. O dado de um sensor é identificado através de um **canal** de dados no monitor, ou seja, cada sensor envia suas leituras para um canal específico do monitor. Cada leitura (conjunto de leituras dos canais) possui uma data e uma hora específica. O sistema não permite que um mesmo monitor possua duas leituras na mesma data e hora para o mesmo canal.

Os usuários podem cadastrar no Sistema vários monitores (ambientes) com vários canais. Para cada canal deve-se identificar o tipo de dado que o sensor conectado ao canal coleta: um número, uma imagem, um som, etc. Essas configurações – dos monitores e dos canais – devem ser realizadas antes do início do envio de dados pelos monitores.

Uma vez que um monitor esteja cadastrado, ele receberá um número, único no sistema (idMonitor), que o identificará. Cada canal também possuirá um número de identificação que poderá ser escolhido pelo usuário.

O usuário tem a possibilidade de configurar o sistema para salvar ou não os dados recebidos dos canais no Banco de Dados. Essa funcionalidade é útil no caso de ambientes no qual se deseja apenas efetuar o monitoramento e não se necessita armazenar um histórico do ambiente. Uma outra possibilidade é armazenar apenas canais numéricos (que ocupam pouco espaço no armazenamento) e utilizar os canais de imagem e som apenas para o monitoramento e execução de tarefas. No caso dos dados não serem guardados no Sistema, os dados recebidos são combinados e fornecidos para os usuários que estão executando o monitoramento naquele momento através da interface do usuário, além de serem utilizados na execução das tarefas programadas no sistema.

Todo o processo de combinação de dados é realizado configurando-se tabelas no sistema. Assim, um monitor pode possuir diversos canais e diversas tabelas. Entende-se por tabela um conjunto de operações que são realizadas sobre os dados dos canais e sobre colunas já processadas da tabela.

Por exemplo, um monitor que possui seis canais: três canais de temperatura (numérico), um de pressão (numérico) e duas imagens registradas por câmeras (arquivos JPG). Pode-se configurar uma **tabela** para que a primeira coluna exiba a média dos valores das temperaturas dos três canais que recebem os valores dos sensores de temperatura, a segunda coluna recebe o valor da pressão vezes 100, a terceira coluna exiba uma única imagem resultante da média dos valores de cada *pixel* das imagens, a quarta coluna calcule a quantidade de *pixels* com nível de vermelho entre 123 e 158 da imagem combinada (imagem da quarta coluna) e a quinta coluna relacione de alguma maneira através de uma lógica *Fuzzy* a quantidade de *pixels* calculada na coluna quatro com a média das temperaturas e a pressão, resultante na classificação do ambiente numa escala entre o “Tipo A” (nível 0) ou “Tipo B” (nível 1).

No exemplo descrito foram realizados três tipos de fusão: uma fusão no nível de sinal realizada nos canais dos sensores de temperatura (Média Aritmética), uma fusão no nível de *pixel* nas imagens e posterior extração de uma característica da imagem resultante (o número de *pixels* com nível vermelho entre 123 e 158) e uma

fusão no nível de símbolo (lógica *Fuzzy*) para combinar a média das temperaturas, a pressão e o número de *pixels* em uma classificação do ambiente (Tipo A ou B).

A configuração das tabelas permite, portanto, a execução de diversos tipos de algoritmos sobre os dados brutos (vindos dos canais dos monitores) e sobre os dados já processados em outras colunas da tabela.

Dependendo do tipo de ambiente e das possibilidades de atuação nele, o usuário pode configurar uma tarefa para ser executada. Por exemplo, se a classificação do ambiente for “Tipo A”, o sistema pode enviar um e-mail para o usuário e se for do “Tipo B”, o sistema pode realizar o envio de uma mensagem (o *POST* de um texto XML, por exemplo) contendo um comando para acionar um alarme ou para acertar a temperatura do ambiente sensoriado.

Tanto o monitoramento como as consultas dos dados dos canais (histórico) devem ser realizadas através da interface do usuário. O usuário tem a possibilidade realizar a consulta transformando as tabelas em gráficos (excetuando-se as colunas do tipo imagem, som e outros tipos de dados nos quais não há a possibilidade de gerar gráfico). Todas as tabelas são ordenadas (de forma crescente ou decrescente, de acordo com a escolha do usuário) pela data e hora da leitura.

O processo de execução das tarefas cadastradas é executado automaticamente quando o Servidor recebe os dados do monitor, independentemente se os dados são ou não armazenados no Banco de Dados.

A seguir será realizada uma descrição detalhada das responsabilidades e da organização interna de cada parte do sistema e seus componentes, assim como das tecnologias e protocolos de comunicação utilizados no desenvolvimento do Sistema.

3.2.1 – Os monitores

O papel dos monitores no Sistema de Fusão de Sensores é concentrar os dados provenientes dos sensores que estão monitorando um determinado ambiente e agrupá-los em um formato específico que deverá ser enviado através de uma conexão de rede (TCP/IP) para o Servidor. O monitor deve, portanto, ser configurado para enviar os dados para um *socket* (IP e porta) específicos do servidor.

O monitor pode ser um computador conectado à rede e aos sensores (através de uma interface serial, interface USB ou placas de aquisição de dados específicas para os módulos sensores) ou um dispositivo eletrônico que implemente uma comunicação de rede TCP/IP.

A comunicação entre os monitores e o servidor é realizada através do protocolo HTTP que originalmente foi desenvolvido com a finalidade de envio de documentos HTML através da Internet, mas que pode ser utilizado para o envio e recebimento de *streams* de bytes com outra formatação e finalidade.

O protocolo HTTP define oito métodos de ação (GET, HEAD, POST, PUT, DELETE, TRACE, OPTIONS e CONNECT). O Servidor está configurado para receber documentos XML através do método POST, que é o método do protocolo HTTP para submissão (envio) de dados.

A segurança no transporte pode (e em muitos casos deve) ser realizada utilizando-se o protocolo HTTPS que realiza a cifração dos dados através da implementação de protocolos de segurança SSL (*Security Socket Layer*) e TLS (*Transport Layer Security*), garantindo assim uma razoável segurança no transporte dos dados, evitando ataques do tipo “homem do meio” [TAN03].

Os dados dos sensores a serem enviados para o Servidor são organizados dentro de um documento texto utilizando a linguagem XML que será descrita no próximo item (3.2.1.1).

Diversas tecnologias podem ser utilizadas para se desenvolver um Monitor. Desde a estrutura física (placas, circuitos, etc) até os componentes de software (Sistema Operacional, pré-processamento das informações, etc) podem ser desenvolvidos utilizando diferentes componentes. Contudo, as regras de comunicação (POST via HTTP) e formatação da mensagem (linguagem XML) devem ser respeitadas para que o Servidor possa “entender” a mensagem contendo os dados dos sensores que está sendo enviada pelo Monitor.

3.2.1.1 – XML

XML (*Extensible Markup Language*) é uma linguagem de marcação simples e flexível que tem o objetivo de descrever dados de uma maneira estruturada. Essa linguagem é derivada da especificação SGML (*Standard Generalized Markup Language*), linguagem utilizada como referência para a criação de novas linguagens de marcação [WOR05A], recomendada pela W3C (*World Wide Web Consortium*).

A especificação SGML foi definida em 1986 pela padronização ISO 8879, originalmente com o objetivo de troca, armazenamento e processamento de documentos eletrônicos complexos, mas tem sido largamente utilizada na concepção de linguagens para troca de mensagens. XML é um subconjunto do padrão SGML que pode ser utilizado para marcação de fórmulas matemáticas, estruturas moleculares em Química, gráficos, dados e mensagens de diferentes tipos [DEI00] e sua utilização vem crescendo a cada dia em aplicações na rede mundial de computadores.

Diferentemente da linguagem HTML utilizada para visualização de hipertextos nos navegadores e outros aplicativos, a linguagem XML permite a criação de *tags* próprias, criando uma estrutura organizada de dados que pode ser facilmente lida pelo computador e entendida pelo usuário.

A figura 3.3 apresenta um exemplo de documento, utilizando a formatação XML. Nesse documento é possível identificar com clareza dois produtos de um estoque (identificação do produto, descrição, código do fornecedor, data de recebimento, quantidade e preço unitário). Através de um analisador de documentos XML é possível processar esse documento buscando o valor dos atributos identificados pelas *tags*.

Um documento XML deve respeitar algumas regras na sua formatação. Todo elemento deve ser representado por uma *tag* de abertura (`<ELEMENTO>`) e uma de fechamento (`</ELEMENTO>`), sempre havendo correspondência entre elas, isto é, no documento devem existir tantas *tags* de fechamento quanto forem as *tags* de abertura. É possível indicar atributos nos elementos definindo o nome do atributo e o seu valor dentro da *tag* do referido elemento. No exemplo da figura 3.3, `unid` é um atributo do elemento `QUANTIDADE`.

```
<?xml version="1.0"?>
<!DOCTYPE ESTOQUE SYSTEM "estoque.dtd">

<ESTOQUE>
       <PRODUTO>
              <PRODUTO_ID> 1204234 </PRODUTO_ID>
              <DESCRICAO> RESISTOR 100 Ohm </DESCRICAO>
              <COD_FORNECEDOR> 8964-34 </COD_FORNECEDOR>
              <DATA_RECEBIMENTO> 25/10/2005 </DATA_RECEBIMENTO>
              <QUANTIDADE unid="unidades"> 12500 <QUANTIDADE>
              <PRECO_UNITARIO> 0.01 </PRECO_UNITARIO>
       </PRODUTO>
       <PRODUTO>
              <PRODUTO_ID> 1204246 </PRODUTO_ID>
              <DESCRICAO> RESISTOR 1KOhm </DESCRICAO>
              <COD_FORNECEDOR> 8964-34 </COD_FORNECEDOR>
              <DATA_RECEBIMENTO> 25/10/2005 </DATA_RECEBIMENTO>
              <QUANTIDADE unid="unidades"> 23800 <QUANTIDADE>
              <PRECO_UNITARIO> 0.01 </PRECO_UNITARIO>
       </PRODUTO>
</ESTOQUE>
```

Fig. 3.3 - Exemplo de um documento XML

Todo documento XML deve conter apenas um elemento raiz. No exemplo, **<ESTOQUE>** é o elemento raiz. Uma *tag* vazia pode ser representada por **<ELEMENTO/>**, podendo esta ter atributos ou não.

Um documento XML bem formado é aquele que atende todas as regras estruturais e notacionais da linguagem. Qualquer tentativa de leitura de um documento XML que não siga essas regras deve ser rejeitado. As três principais regras são:

- Cada *tag* de abertura deve possuir uma de fechamento, ou seja, para cada <ELEMENTO> deve existir um </ELEMENTO> no documento;
- Uma *tag* que é aberta dentro de outra *tag* deve ser fechada antes do fechamento da *tag* externa, mantendo uma estrutura hierárquica no documento;
- Os valores dos atributos dos elementos devem estar entre aspas.

Contudo, um documento bem formado não necessariamente é um documento válido. Para que isso ocorra, é necessário que o documento esteja de acordo com uma estrutura definida em outro documento denominado DTD (*Document Type Definition*). Dentro de um DTD são definidas informações como quais elementos devem estar presentes no documento XML, quais os atributos dos elementos, qual a

hierarquia dos elementos, etc.. Pode-se dizer que o DTD é a gramática de um documento XML.

O uso de um DTD é opcional, mas ele é recomendável para garantir que a estrutura do documento corresponde à sua especificação e não haverá erros ou falhas no processo de leitura do mesmo.

A figura 3.4 mostra o documento DTD referente ao XML da figura 3.3. A linha `<!DOCTYPE ESTOQUE SYSTEM "estoque.dtd">` do XML da figura 3.3 indica que o documento deve seguir as especificações DTD contidas no arquivo estoque.dtd, que não precisa ser enviado junto com o XML, pois tanto a parte que gerou o XML como a parte que o receberá deverão possuir *a priori* a especificação DTD.

```
<!ELEMENT ESTOQUE(PRODUTO)>
<!ELEMENT PRODUTO(PRODUTO_ID, DESCRICAO, COD_FORNECEDOR,
DATA_RECEBIMENTO, QUANTIDADE, PRECO_UNITARIO)>
<!ELEMENT PRODUTO_ID (#PCDATA)>
<!ELEMENT DESCRICAO (#PCDATA)>
<!ELEMENT COD_FORNECEDOR (#PCDATA)>
<!ELEMENT DATA_RECEBIMENTO (#PCDATA)>
<!ELEMENT QUANTIDADE (#PCDATA)>
<!ATTLIST TITULO unid CDATA #REQUIRED>
<!ELEMENT PRECO_UNITARIO (#PCDATA)>
```

Fig. 3.4 – Exemplo de uma especificação DTD (arquivo estoque.dtd)

As definições de um documento DTD devem seguir as seguintes especificações:

- As *tags* dos elementos devem ser definidas por `<!ELEMENT...>`. A linha `<!ELEMENT ESTOQUE(PRODUTO)>` define a *tag* `<ESTOQUE>` que contém a *tag* `<PRODUTO>` como sub-elemento;
- O conteúdo de cada elemento deve ser especificado como na linha `<!ELEMENT DESCRICAO(#PCDATA)>`, na qual o elemento `DESCRICAO` pode conter qualquer tipo de dado;
- Os atributos de determinada *tag* devem ser indicados por `<!ATTLIST...>`. A linha `<!ATTLIST TITULO unid CDATA #REQUIRED>` define que o elemento `TITULO` deve ter um atributo chamado `unid`. Um atributo pode ser `#REQUIRED` quando a sua presença no elemento é obrigatória, `#IMPLIED` quando o atributo não precisa ser

utilizado (interpretador XML pode utilizar qualquer valor para esse atributo) e **#FIXED** quando o valor no documento XML não pode ser diferente do especificado no DTD. Nesse último caso, o valor fixo deve ser especificado no DTD entre aspas após a palavra **#FIXED**.

Um programa denominado *parser* (analisador) é utilizado para verificar se um documento XML é bem formado e válido. As principais APIs (*Application Programming Interface*) que processam arquivos XML são a DOM e a SAX.

DOM (*Document Object Model*) é uma interface independente de plataforma que manipula documentos XML representando-o como uma árvore cujos nós são os elementos [WOR05B]. O analisador DOM monta a estrutura inteira do documento XML na memória, sendo por esse motivo indicado para aplicações que necessitam modificar a estrutura do documento (como ordenar os elementos do documento).

A interface SAX (*Simple API for XML*) lê um documento XML e gera eventos a partir dos elementos lidos. Esses eventos podem ser programados de acordo com o conteúdo de cada elemento. Originalmente a interface SAX foi desenvolvida para a linguagem de programação JAVA, mas com o tempo ela foi adotada por outras linguagens [SAX05].

A forma de leitura dos documentos XML através da interface SAX é mais simples e rápida, requerendo menos memória que a API DOM, uma vez que ela não aloca memória para construir toda a árvore hierárquica do documento. SAX é recomendado para aplicações que manipulam objetos grandes (que muitas vezes não cabem na memória) e processamentos rápidos de leitura do documento XML que não precisam alterar a sua estrutura (por exemplo, contar o número de elementos de um documento).

3.2.1.2 – Utilização da linguagem XML no Sistema de Fusão de Sensores

O envio dos dados do Monitor para o Servidor obedece às regras especificadas no DTD do arquivo **leitura.dtd** descrito no Apêndice A.

Um exemplo de XML de envio de dados está presente na figura 3.5. Cada monitor recebe um número que o identifica unicamente dentro do sistema. Esse número é incorporado ao XML na *tag* **<MONITOR></MONITOR>** e é responsável pela identificação da origem dos dados a serem recebidos.

```
<?xml version="1.0"?>
<!DOCTYPE LEITURA SYSTEM "leitura.dtd">
<LEITURA>
       <MONITOR> 1 </MONITOR>
       <DATA> 08/10/2005 </DATA>
       <HORA> 15:01:00 </HORA>
       <CANAL>
              <ID_CANAL> 0 </ID_CANAL>
              <VALOR> 3.46 </VALOR>
       </CANAL>
       <CANAL>
              <ID_CANAL> 1 </ID_CANAL>
              <VALOR> 3.78 </VALOR>
       </CANAL>
       <CANAL>
              <ID_CANAL> 2 </ID_CANAL>
              <VALOR> 3.20 </VALOR>
       </CANAL>
       <CANAL>
              <ID_CANAL> 3 </ID_CANAL>
              <VALOR> 3.63 </VALOR>
       </CANAL>
</LEITURA>
```

Fig. 3.5 - Exemplo de um documento XML utilizado no envio de dados para Servidor

A data e a hora da leitura dos valores podem ou não ser preenchidas nos elementos **DATA** e **HORA**. Se elas estiverem vazias (**<DATA></DATA>** e **<HORA></HORA>** ou **<DATA/>** e **<HORA/>**), quando o servidor receber o documento, ele considerará a data e hora no instante do recebimento como a data e hora de leitura dos dados. Isso pode ser útil para não haver a necessidade de se implementar um relógio sofisticado na parte do monitor.

Como os dados dos sensores são enviados para os canais do monitor, cada canal deve ser mapeado em um elemento **CANAL**. Como se pode notar no exemplo e na especificação DTD do Apêndice A, o elemento **CANAL** possui dois sub-elementos: **ID_CANAL** e **VALOR**. O valor de **ID_CANAL** é o número identificador do canal e o valor de **valor** é a leitura do sensor.

No XML da figura 3.5, o monitor 1 possui 4 canais (0, 1, 2 e 3) que estão registrando os valores 3,46, 3,78, 3,20 e 3,63, respectivamente.

A identificação da natureza do dado, da unidade em que ele se encontra e outras informações sobre os canais são configuradas no servidor através da interface HTML. Essa configuração, que será descrita no item 3.2.3.1, identificará o modo como o dado do canal deve ser lido e armazenado no sistema.

Como cada dado originado pelos sensores é diferente: pode ser um número (decimal, binário, hexadecimal, etc.), uma imagem (arquivo BMP, JPG, etc.) ou um arquivo de áudio (WAV, por exemplo), há a necessidade de adaptar os dados para que eles se tornem textos (caracteres imprimíveis) para poderem ser transmitidos através de um documento XML.

Quando se tratar de um número ou de uma cadeia alfa-numérica, não há necessidade de transformar os dados. Em caso de arquivos binários como imagens, sons ou outros tipos de conteúdo que possuam caracteres não imprimíveis, deve-se utilizar o artifício de transformar os dados binários para Base 64, permitindo assim que eles sejam incorporados ao documento XML.

A conversão para a Base 64 transforma uma seqüência de bits em 64 caracteres da tabela ASCII (*American Standard Code for Information Interchange*). A tabela ASCII possui 95 caracteres imprimíveis (figura 3.6), mas somente 64 são utilizados: A-Z, a-z, 0-9, / e +.

```
!"#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNO
PQRSTUVWXYZ[\]^_`abcdefghijklmnopqrstuvwxyz{|}~
```

Fig. 3.6 – Os 95 caracteres imprimíveis da tabela ASCII

A figura 3.7 ilustra a transformação de uma imagem (logotipo de 50 x 57 *pixels* da Wikipedia – A Enciclopédia Livre[1]) em um texto na Base 64.

Existem diversos softwares disponíveis na Internet que realizam a transformação de arquivos para a Base 64. As bibliotecas da Sun Microsystems **sun.misc.BASE64Encoder** e **sun.misc.BASE64Decoder** são nativas do JAVA [JAV05] e foram utilizadas no processo de codificação de arquivos nos Monitores e decodificação no Servidor.

[1] http://www.wikipedia.com

(a)

/9j/4AAQSkZJRgABAQEASABIAAD/2wBDAAYEBQYFBAYGBQYHBwYIChAKCgkJChQODwwQFxQYGBcU
FhYaHSUfGhsjHBYWICwgIyYnKSopGR8tMC0oMCUoKSj/2wBDAQcHBwoIChMKChMoGhYaKCgoKCgo
KCgoKCgoKCgoKCgoKCgoKCgoKCgoKCgoKCgoKCgoKCgoKCgoKCgoKCj/wAARCAA5ADIDASIA
AhEBAxEB/8QAGwAAAgMBAQEAAAAAAAAAAAAAAAAAAAYEBQcDAgj/xAA1EAACAQMDAgQEAwcFAAAAAAAB
AgMEBREAEiExQQYTIlEUMmGRcXKBBxYjMzRCUmKCobHB/8QAFAEBAAAAAAAAAAAAAAAAAAAAAAAAAP/E
ABQRAQAAAAAAAAAAAAAAAAAAAAAA/2gAMAwEAAhEDEQA/APqkkAEnppN8aVtbWV1usFpq6ikqq9y0
ksCKXjplH8STLghaK9gyeTzlweAcE/QH7/AG1l/j63+JmuVRcaSpkitNvaBaOlpVLT
SPjBZsKW6ydC2MKCdBoVxuNNYLKSPiKhaREiKxq08vTC7gMnJ9zx3Jxzpepr3dZykaOPIKITNINs
hJUnpjHPBIHTd9qKweHqqLyr3ca2oNxbMvw8yHcxAIw2Tk8Hj8Rq28SVz24qRQ1NW05AzEPSCDwO
MkHv9cYznAIQoL3eLVdDJJWGqoGzvhmwWBBHynryCfsPfWg2q50t1pfiKKUSJnaw6FT7Eazaq3GP
dPHsLICwAOMkDkk+xyPpqt8B3KeyeLBBPUCSmr5TERjAQn5O55zx/u7cDQbPo0aNBR3qkFXXUrNI
8QppfMyo6+nAyfbnUW8xGttpipqgQTxzpIkyhgpwwyCQehXcvXv21K8QrVJPTS01QYYy480BAdyj
k89uNRKxppBNHTEHkkB8bQ2NwPHPzY/50EejpjTLMks81VIJXbKkqEVjuCctkgdudQqioZrbHJI+
yMKJGkZuqdecfTr01FtM16hqna8LTpA0eSUIwHwvfOSOWHJ7D31zneGRn+UKjkKqjJxj39uTwNBR
tdaW4yVdNTNKpWNJcMo2lXB2kY6/Kcj79dKt+mmN1oZoZNjLUrIVHfDAjH66vKiaitopqKE00Egh
2R04YbgBzwOpHzd++qix0NRf/G9mpKdmWCKYVFScdY0O7GfYkKP10H0Xo0aNBGudGlwoZqWR5IxI
pUPG210Pup7EaS5YZrbN5dxCvglVfaPUpPBz3x99SvF3i2s8PV9UklHA9JFSCuSTe254kcCowMfN
GrK4H9wOOMa6i/pPBi+UsHwzxbh5WZPWIjKy/pHg5HUkgdOQUpqq4xT+j4dliOxGIwChPc7sj7HX
KsrwkjFzlinLZ6YPToPfV9cLTaJYoJ6aproY6lzHF6BIrERtIQM+yq3OeoI66qKKwWW41kcL3avm
d1zsSERnGIz1Oe00Z4/y+hwCRXyCpqlprfSCarnkLLHCg3M5yM8DrgnJ1sP7OvCf7uUDzVm17pUg
ecQchB2QH8eSe5/Aa8eH5fDVlpaeW3U7xPWJE6SOhMsyyOqKcntudAR23DIGdWkHiSCXxjPYdoEi
U3nJJvB3spXzEx22iSI577j7aC+0aNGgXvF9vjrDbJmoxVSUtQZowUZtp8tskhSMjGRg8EldL0Vv
2+G7bQUtnraeOiMkPlYLeloH3MueSPVtGT1OOo1oWjQICUQhoDBSWy6wCkqBJFE6hkVn3QuUAHTa
0jYHHIPfXumeniagC2i6QvSR/CxbYgu5TtYvjGefKXJ6jjudPmvB/nL+U/8AmgzqIwRpR08llvEj
UqxrGyrl4YklV4/TjoTGu44ySgHIHE+xUor7pRNU0l1p5qGeoqoZpU2BhIfUHJX1bg59PGMf6VId
F/qpPyL/ANnXXQGjRo0H/9k=

(b)

Fig. 3.7 - Exemplo de uma imagem (a) no formato JPG (50x57 *pixels*) e (b) sua representação na Base 64.

Uma vez que um XML é enviado ao servidor, um outro XML de resposta é recebido pelo monitor contendo a situação da submissão dos dados. O XML pode estar bem formado e o monitor recebe o XML da figura 3.8. Se houver algum tipo de erro na estrutura do documento, não haverá processamento do mesmo e uma mensagem com *status* de erro será enviada para o monitor. Na figura 3.9, o erro reportado pelo XML de resposta é que o XML de envio possuía uma *tag* **"LEIURA"** que não faz parte da especificação DTD do documento. A mensagem de erro é gerada automaticamente pelo *parser* SAX do JAVA.

```
<RESPOSTA>
        <STATUS> OK </STATUS>
        <MENSAGEM> XML recebido com sucesso </MENSAGEM>
</RESPOSTA>
```

Fig. 3.8 – Documento XML de resposta para uma mensagem bem formada.

```
<RESPOSTA>
       <STATUS> ERRO </STATUS>
       <MENSAGEM> ERRO: Element type "LEIURA" must be declared.</MENSAGEM>
</RESPOSTA>
```

Fig. 3.9 - Exemplo de um documento XML de resposta para uma mensagem que apresenta erro na sua formação.

3.2.2 – O servidor

O papel do servidor está relacionado com o recebimento dos dados dos monitores, o armazenamento das tabelas (com as regras de Fusão de Sensores e a formatação dos dados dos sensores), a execução da combinação dos dados e das tarefas relacionadas ao ambiente monitorado, a consulta do histórico dos dados dos sensores e o fornecimento dos dados para o monitoramento dos ambientes.

Todas as informações dos módulos, formas de fusão, tarefas e os dados dos sensores disponíveis para consulta estão armazenados em um Banco de Dados.

Um *socket* deve ser reservado para a execução de um programa desenvolvido para o recebimento dos XML (HttpServer) contendo os dados do sensor; um outro deve ser utilizado para a execução de um servidor HTTP (no caso, o Tomcat) que permite o acesso dos usuários à interface de configuração, consulta e monitoramento.

A execução de tarefas é realizada toda vez que o programa HttpServer receber um XML, dependendo da condição de execução da tarefa. Uma tarefa geralmente utiliza a comunicação disponível pela Internet para o envio de e-mails de aviso ou envio de comandos para a intervenção no ambiente sensoriado.

A comunicação do Servidor com as outras partes do sistema é realizada através da Internet. Portanto, é extremamente importante garantir uma conexão de rede adequada para máquina servidora. Políticas de controle no acesso à máquina (segurança) também devem ser adotadas (bloqueio do *firewall* por exemplo).

A figura 3.10 detalha o fluxo dos dados e as etapas do processamento. Primeiramente os monitores enviam os documentos XML através de uma conexão HTTP utilizando o método POST. O programa HttpServer recebe o XML através da porta 9988 do servidor (vonbraun.lpm.fee.unicamp.br) e valida o documento enviando

uma resposta (item 2 do diagrama) para o monitor com a situação da operação, ou seja, com o *status* da operação de envio do XML.

O terceiro estágio é inserir os dados do XML na fila de processamento. Essa fila é responsável pela leitura e interpretação dos dados de acordo com as configurações de cada canal, isto é, o dado de um canal numérico será lido como um número, um dado imagem será decodificado da Base 64 para sua forma binária original e assim por diante.

O quarto estágio é verificar se os dados dos canais devem ou não ser gravados no Banco de Dados. Essa alternativa é verificada na configuração de cada canal.

Em seguida (item 5 do diagrama), são verificadas as condições dos dados através da configuração dos mecanismos de Fusão de Sensores e as tarefas relacionadas às condições são executadas.

Toda vez que um dado entra na fila de processamento ele deve ser disponibilizado para o monitoramento e para a consulta. O processo de monitoramento disponibiliza sempre os últimos dados recebidos dos canais (item 6 do diagrama) e não acumula dados de recebimentos anteriores. Assim, quando um usuário acessa a porta 8080 do servidor e faz a requisição de um monitoramento através de uma tabela ou gráfico na interface do usuário, o servidor HTTP acessa a porta 9596 requisitando os últimos dados do monitor selecionado pelo usuário. Nesse estágio (item 8 do diagrama), ocorre a Fusão dos Dados de acordo com as regras da tabela selecionada.

A acumulação dos dados durante o monitoramento é feita no Tomcat, que armazena em memória os dados processados de acordo com as regras de combinação dos dados das tabelas. Dessa maneira, não é preciso processar todos os dados acumulados durante o processamento, havendo a necessidade apenas de processar os últimos dados recebidos na requisição feita na porta 9596. Isso faz com que o processo de monitoramento seja agilizado.

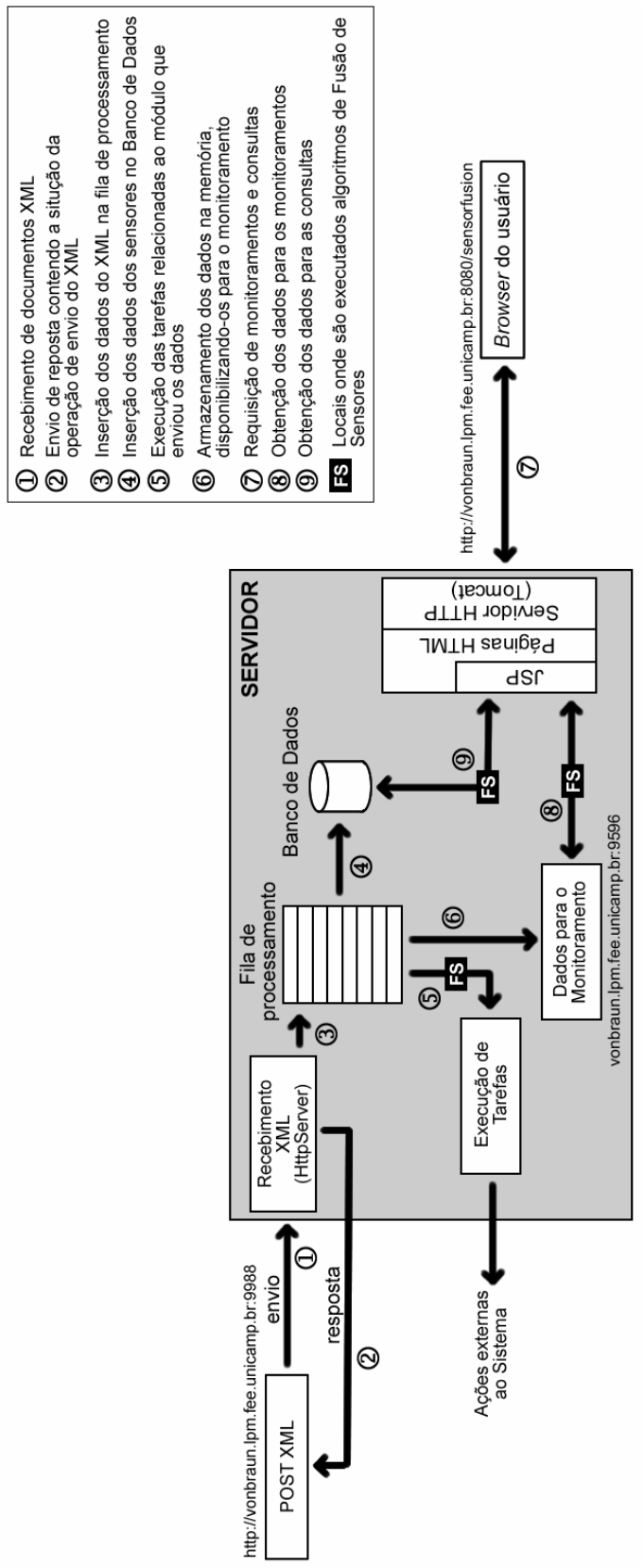


Fig. 3.10 – Fluxo dos dados e etapas de processamento no Servidor.

A consulta também é requisitada pela interface do usuário acessada pela porta 8080 do servidor. O servidor HTTP faz uma requisição ao Banco de Dados dos dados requisitados para a consulta (item 9 do diagrama) e são realizados os mecanismos de Fusão de Sensores pertinentes à tabela escolhida para a consulta. Dependendo do número de leituras (conjunto de valores dos canais) e da natureza dos dados dos canais (imagem, som, número, etc.), o procedimento de consulta pode demorar pela necessidade de processar todos os dados consultados pelo mecanismo de Fusão de Dados. O tempo necessário para gerar as tabelas está diretamente relacionado à complexidade dos algoritmos executados nas colunas e ao número de linhas resultantes da consulta.

Tanto as consultas como os monitoramentos são baseados em tabelas que contêm as regras de fusão, podendo existir diversas tabelas para um mesmo módulo. Portanto, não seria razoável guardar os dados processados por cada tabela – agilizando, assim, as consultas – visto que haveria a necessidade de uma quantidade de espaço em disco muito grande para tal.

A tabela 3.1 relaciona as portas utilizadas pelo servidor, a sua utilização e os programas que as utilizam. O número das portas pode ser alterado sem maiores dificuldades, contudo tendo impacto externo no recebimento dos dados através dos documentos XML (porta 9988) e acesso à interface do usuário (porta 8080).

Tabela 3.1 – Portas utilizadas no Sistema

Porta	Utilização	Programa
8080	Acesso à interface do usuário (páginas HTML/JSP)	Tomcat
9988	Recebimento dos documentos XML com os dados dos sensores	HttpServer
9596	Fornecimento dos últimos dados recebidos dos sensores para o monitoramento (porta utilizada internamente ao Sistema)	HttpServer

3.2.2.1 – Estrutura de software

Todo o projeto de software foi desenvolvido baseando-se na linguagem de programação Orientada a Objetos JAVA. A escolha dessa linguagem pode ser explicada pelos seguintes fatores:

• Portabilidade. A independência de plataforma permite a interpretação das classes compiladas em diferentes arquiteturas;

• Disponibilidade de recursos de redes. A existência de classes na API JAVA que implementam protocolos TCP/IP, HTTP e HTTPS facilita e agiliza o desenvolvimento de aplicativos que necessitam comunicação de rede;

• Disponibilidade de API para manipulação de documentos XML. A API do JAVA possui classes para leitura e manipulação de documentos XML tanto no esquema SAX quanto no DOM;

• Possibilidade de execução de vários processos paralelamente com baixo custo para o Sistema Operacional (*Multi-thread*);

• Integração com Servidor HTTP Tomcat através da tecnologia JSP;

• Documentação facilitada e organizada através do JAVADOC;

• Reuso de classes e facilidade de incorporação de novas classes ao projeto.

As classes do projeto do Sistema de Fusão de Sensores foram organizadas em 11 pacotes de acordo com as suas funcionalidades. A figura 3.11 lista os pacotes.

```
br.eng.rsalustiano.sensorfusion.business
br.eng.rsalustiano.sensorfusion.chart
br.eng.rsalustiano.sensorfusion.database
br.eng.rsalustiano.sensorfusion.element
br.eng.rsalustiano.sensorfusion.expression
br.eng.rsalustiano.sensorfusion.expression.function
br.eng.rsalustiano.sensorfusion.http
br.eng.rsalustiano.sensorfusion.structure
br.eng.rsalustiano.sensorfusion.task
br.eng.rsalustiano.sensorfusion.util
br.eng.rsalustiano.sensorfusion.xml
```

Fig. 3.11 – Pacotes da arquitetura de software do Sistema.

O pacote **business** possui as classes que serão utilizadas nos arquivos JSP. Toda interface do usuário está relacionada às classes desse pacote. Dentro de um arquivo JSP não é utilizada nenhuma outra classe de outro pacote. Dessa forma, um

outro tipo de interface com o usuário pode ser implementada alterando-se apenas as classes desse pacote.

O pacote **chart** possui as classes destinadas a criação de imagens gráficas baseadas no pacote da API JAVA **java.awt**. As classes do pacote **chart** são responsáveis por criar os gráficos que permitem a visualização dos dados de sensores numéricos.

O pacote **database** possui as classes de acesso ao Banco de Dados. Elas utilizam o *driver* de acesso ao MySQL desenvolvido para a linguagem JAVA.

As classes destinadas à comunicação em rede através do protocolo HTTP estão reunidas no pacote **http**. A classe HttpServer, pertencente a este pacote, contém o programa responsável por receber os documentos XML, colocar os dados na fila de processamento, inseri-los no Banco de Dados, executar as Tarefas e disponibilizar os dados para o Monitoramento (conforme esquema da figura 3.10).

Os pacotes **structure** e **util** contêm classes que auxiliam as classes de outros pacotes. São classes que possuem métodos de criptografia para inserção das senhas dos usuários no Banco de Dados, codificam e decodificam dados na Base 64, apresentam estruturas para organização dos dados na forma de tabelas, transformam tabelas no formato de arquivo XLS, convertem números de uma base numérica para outra, entre outras funções.

O pacote **task** compreende as classes destinadas à execução de tarefas. Essas classes possuem métodos que enviam dados para fora do sistema, como e-mail, por exemplo.

Os principais pacotes para se entender a arquitetura de Fusão de Sensores desenvolvida são: **expression**, **expression.function** e **element**. Elas serão discutidas no item 3.2.2.2.1, que fará um detalhamento das classes e das suas relações, dentro da explicação de como os dados dos canais são combinados.

A tabela 3.2 relaciona os pacotes com as principais funcionalidades de suas classes.

Tabela 3.2 – Pacotes e descrição das funcionalidades das classes do Sistema

Pacote	Descrição das funcionalidades das classes
business	Classes que organizam os dados (módulos, canais, descrição das tabelas, etc) para serem utilizados nos arquivos JSP.
chart	Classes relacionadas à criação de Gráficos.
database	Classes de acesso ao Banco de Dados.
element	Classes para a organização dos Elementos do Sistema. Um Elemento pode ser uma imagem, um áudio, um número, uma *string*, etc.
expression	Classes destinadas à resolução de expressões. Utiliza os Elementos (pacote element) e as funções agrupadas no pacote expression.function para resolver os mecanismos de fusão.
expression.function	Funções disponíveis no sistema que implementam os algoritmos de fusão.
http	Classes que executam a comunicação em rede através do protocolo HTTP.
structure	Classes auxiliares contendo as estruturas utilizadas no sistema, como tabelas, por exemplo.
task	Classes contendo as tarefas disponíveis no sistema.
util	Classes de utilidades, como a codificação e decodificação na Base 64, métodos criptográficos, etc..
xml	Classes voltadas para a leitura de documentos XML através do esquema SAX.

3.2.2.2 – Tabelas, colunas e a Fusão de Sensores

Todo o processo de Fusão de Sensores está baseado na configuração de tabelas. Cada coluna de uma tabela pode ser configurada para exibir o valor de um canal, uma formatação específica do valor de um canal ou a execução de algoritmos tendo como entrada valores dos canais ou outras colunas da tabela.

Cada módulo possui uma lista de canais cadastrados no Sistema. Cada canal é identificado com a letra “C” seguida do número do canal (o número que o usuário cadastrou o canal). O canal número 10 é identificado por C10, por exemplo. Essa identificação dos canais será utilizada para realizar a fusão e formatação dos dados dos sensores.

Cada coluna de uma tabela possui os seguintes campos: nome, unidade, conteúdo e faixa de valores ideais. O nome é o identificador do conteúdo que será

exibido na coluna (por exemplo, temperatura, pressão, umidade, imagem da câmera A, Conforto Térmico, etc.); a unidade refere-se a própria unidade do valor exibido na coluna (por exemplo, ℃, mmHg,%RH, etc); o conteúdo é uma expressão matemática que indicará qual valor ou qual o resultado de uma operação será exibido nas células da coluna (por exemplo, a expressão SIN(C1) exibirá o seno do valor do canal 1); e a faixa de valores ideais é uma expressão lógica que permite fazer com que o valor exibido nas células da coluna sejam destacados visualmente na interface do usuário se a expressão lógica for verdadeira (por exemplo, a expressão C1≥100 fará com que as células da coluna fiquem em destaque se o valor do canal 1 for menor ou igual a 100).

De modo análogo à identificação dos canais, as colunas são identificadas pela sigla “COL” seguida do número da coluna. COL2 refere-se, portanto, à segunda coluna da tabela.

As tabelas 3.3, 3.4 e 3.5 ilustram um exemplo de configuração de uma tabela. A tabela 3.3 mostra os 8 canais disponíveis no monitor **Lux-Planta**; na tabela 3.4 são exibidas as configurações das colunas da tabela **Lux-Planta01**; e a tabela 3.5 é a visualização dos dados dos canais combinados de acordo com as regras de fusão determinadas na tabela 3.4. O objetivo do exemplo é utilizar os dados dos 6 sensores de temperatura (C1, C2, C3, C4, C5 e C6) combinados com o sensor de luminosidade (C7) e a imagem da câmera (C10) para obter um valor para a luminosidade incidente em uma planta.

Tabela 3.3 – Exemplo de definições de canais e tipos de canais

Monitor: Lux-Planta [idMonitor: 19]	
Canal	**Tipo de Sensor**
C1	Sensor de Temperatura
C2	Sensor de Temperatura
C3	Sensor de Temperatura
C4	Sensor de Temperatura
C5	Sensor de Temperatura
C6	Sensor de Temperatura
C7	Sensor de Luminosidade
C10	Câmera RGB

Tabela 3.4 – Exemplo de configuração de uma tabela

Tabela: Lux-Planta01	
COL1	**Nome:** Temperatura-Fusão **Unidade:** ℃ **Valor:** CCA(C1, C2, C3, C4, C5, C6) **Faixa de Valores Ideais:** COL1<40
COL2	**Nome:** Luminosidade **Unidade:** lux **Valor:** C7*1.2 **Faixa de Valores Ideais:** C7<200
COL3	**Nome:** Imagem CamA **Unidade:** **Valor:** C10 **Faixa de Valores Ideais:**
COL4	**Nome:** Luminosidade-Fusão **Unidade:** lux **Valor:** LUXFUSION(COL1, COL2, C10) **Faixa de Valores Ideais:** COL4<200

Os 6 sensores de temperatura são combinados em um único valor de temperatura através de um algoritmo de Sensores em Consenso chamado Concordância Inexata (função CCA) (ver capítulo 4) e esse valor é combinado com o valor da luminosidade recebida pelo canal C7 multiplicado por uma constante de calibração (igual a 1,2) e a imagem do canal C10 através de um algoritmo que funde essas três informações e retorna um valor de luminosidade mais representativo das condições do ambiente.

Tabela 3.5 – Exemplo de visualização de uma tabela de acordo com a sua configuração (Tabela 3.4)

Data e Hora	Temperatura-Fusão (℃)	Luminosidade (lux)	Imagem CamA	Luminosidade-Fusão (lux)
25/11/2005 10:00:00	23.8	69.3		68.5
25/11/2005 10:15:00	24.0	128.7		130.2
25/11/2005 10:30:00	24.6	**239.0**		**238.7**

Ainda sobre o exemplo, verifica-se que a terceira leitura (25/11/2005 10:30:00) apresenta tanto a coluna Luminosidade como a coluna Luminosidade-Fusão fora dos valores ideais, por isso estão em destaque na visualização.

A resolução das expressões é o mecanismo chave de Fusão dos Sensores neste Sistema de Monitoramento de Ambientes. Trata-se de um programa que "lê" a expressão e os valores dos canais e realiza as operações programadas. O próximo item explica como é realizada a resolução das expressões.

3.2.2.2.1 – Resolução das expressões

A resolução de expressões matemáticas, lógicas ou composições mistas dessas funções deve seguir os três passos a seguir [APP98]:

- Análise léxica
- Análise sintática
- Análise semântica

Na **análise léxica** as palavras e os símbolos das expressões são identificados de forma a classificá-los dentro de uma gramática de tipos. São identificados os números, as palavras, os sinais de pontuação, etc..

Uma *token* é uma seqüência de caracteres que pode ser tratada como uma unidade dentro da gramática de uma linguagem. A tabela 3.6 mostra os tipos de *tokens* determinados na gramática para se realizar a análise léxica das expressões utilizadas para a fusão dos dados dos sensores. A tabela 3.7 mostra as expressões regulares que determinam os tipos.

Como o mecanismo de Fusão de Sensores permite a manipulação de números em diferentes bases numéricas, foi estabelecido um padrão de representação de valores numéricos, conforme indicado na figura 3.12. A base 10, por ser a mais utilizada, não precisa do termo "0x10x" antes do número, ou seja, quando se escrever um número na base 10 não há a necessidade de explicitar que o número está nessa base.

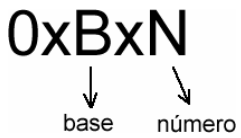


Fig. 3.12 – Forma de representação numérica.

Assim, para a resolução de expressões e para todo o Sistema (inclusive recebimento de valores de canais numéricos em documentos XML), os valores 25, 0x2x11001, 0x7x34, 0x10x25, 0x16x19, 0x21x14, 0x25x10 são iguais. O apêndice C mostra os algoritmos utilizados para a conversão de número entre bases.

Tabela 3.6 – Gramática utilizada para a resolução de expressões. Tipos de *tokens* e exemplos

Tipo	Exemplos
NUMÉRICO	10 2.5 32.675 0x16x4AF 0x2x1001.001 0x23xA3G.H2D
OPERADOR UNÁRIO DIREITO	! (fatorial)
OPERADOR UNÁRIO ESQUERDO	! (negação) - (sinal de negativo)
OPERADOR BINÁRIO	^ (operação de exponenciação) / (operação de divisão) * (operação de multiplicação) - (operação de subtração) + (operação de adição) % (operação de módulo) < (menor) > (maior) <= (menor ou igual) >= (maior ou igual) != (diferente) == (igual) || (ou lógico) && (e lógico) : (intervalo de valores) +- (exatidão)
VÍRGULA	,
STRING	“abracadabra” “vai-e-vem” “isto é um teste”
BOOLEANO	true false
ABRE PARÊNTESES	(
FECHA PARÊNTESES	)
VARIÁVEL	C1 C25 COL10
FUNÇÃO	SIN COS CCA FCA
DESCONHECIDO	outros tipos de seqüências de caracteres

Tabela 3.7 – Expressões regulares utilizadas na análise léxica para a resolução de expressões

Tipo	Expressão regular
NUMÉRICO	([0-9]* "." [0-9]+) | ([0-9]+ "." [0-9]*) | ("0x" [0-9]+ "x" ([0-9] | [A-Z])+ "." ([0-9] | [A-Z])*) ("0x" [0-9]+ "x" ([0-9] | [A-Z])* "." ([0-9] | [A-Z])+)
OPERADOR UNÁRIO DIREITO	"!"
OPERADOR UNÁRIO ESQUERDO	"!" | "-"
OPERADOR BINÁRIO	"/" | "*" | "-" | "+" | "^" | "%" | "<" | ">" | "<=" | ">=" | "!=" | "==" | "||" | "&&" | ":" | "+-"
VÍRGULA	","
STRING	Qualquer seqüência alfa-numérica entre aspas duplas
BOOLEANO	true | false
ABRE PARÊNTESES	"("
FECHA PARÊNTESES	")"
VARIÁVEL	[A-Za-z]+ [A-Za-z0-9]*
FUNÇÃO	Qualquer nome de função cadastrada no sistema
DESCONHECIDO	Outras combinações de caracteres não descritas

Os operadores unários são aqueles que aparecem antes (à esquerda) ou depois (à direita) de números ou variáveis. Os operadores binários são aqueles que necessitam de dois operandos; são as operações de soma, subtração, divisão, etc e os operadores lógicos maior, menor, e, ou, etc..

A ordem de execução dos operadores binários segue a ordem apresentada na tabela 3.6. Quase todos os operadores são conhecidos de outras linguagens de programação, com exceção do "+-" e ":".

Uma vez que cada *token* é identificada na expressão, deve ser realizada uma **análise sintática** na expressão, ou seja, deve ser verificada a forma com que as *tokens* se relacionam na expressão. Por exemplo, (45.4+6)^2 é uma expressão que não apresenta problemas sintáticos, mas a expressão (45.4+)^2 apresenta, pois falta um operando para a operação de soma. Apesar de todos os caracteres poderem ser agrupados em *tokens*, o operador "+" necessita de dois operandos (um à direita e outro à esquerda) para que a expressão possa ser calculada.

O operador "+-" indica a exatidão de um número, ou seja, **10+-2** indica que o número representado pode estar entre 8 e 12. O operador ":" expande os valores dos seus operadores para uma lista de valores. Por exemplo, a expressão **C1:C4** se torna

C1,C2,C3,C4. Esse tipo de operação é útil quando se quer escrever uma expressão com os dados de vários canais que sejam identificados com número seqüenciais. Pode-se utilizar composições de seqüências e valores, como por exemplo: **C1:C3,C5,C7:C9**, resultando: **C1,C2,C3,C5,C7,C8,C9**.

A terceira fase, **análise semântica**, é a fase na qual o significado (ou resultado) da expressão é calculado. Nesta fase podem ocorrer erros de significado (uma divisão por zero, por exemplo).

As classes dos pacotes **expression**, **expression.function** e **element** são as destinadas à representação dos dados dos sensores e resolução das expressões.

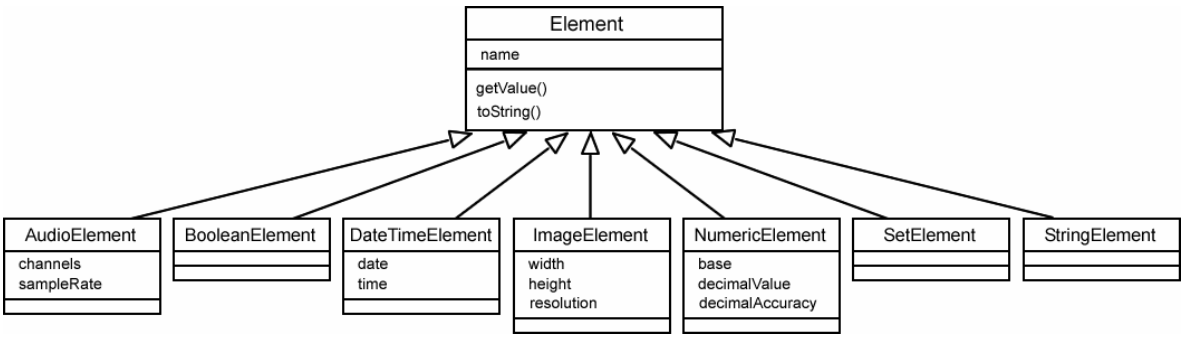


Fig. 3.13 – Diagrama de classes do pacote element.

Um elemento (objeto instanciado das subclasses de **Element** pertencente ao pacote **element**) no sistema corresponde a um valor numérico, uma imagem, um som, um conjunto de elementos, etc..

A figura 3.13 é um Diagrama de classes (notação UML – *Unified Modeling Language* [MUL97]) que evidencia a herança das classes **AudioElement** (elementos tipo áudio), **BooleanElement** (elementos tipo *booleano*, ou seja, verdadeiro ou falso), **DateTimeElement** (elemento tipo data e hora), **ImageElement** (elemento tipo imagem), **NumericElement** (elemento tipo numérico), **SetElement** (elemento contendo um conjunto de elementos) e **StringElement** (elemento tipo *string*, ou seja, uma seqüência de caracteres) em relação aos atributos e métodos da classe **Element**.

Assim sendo, um elemento, seja qual for a sua subclasse, herda o atributo **name** (nome do elemento) e os métodos **getValue()** e **toString()** da classe **Element**. Além disso, cada subclasse possui atributos e métodos específicos de acordo com as

necessidades do tipo de dado que ela está armazenando. Um canal do tipo imagem, por exemplo, deve ser mapeado em um **ImageElement**, possuindo assim os atributos altura (**height**), largura (**width**) e resolução (**resolution**) da imagem.

A figura 3.13 mostra as 7 subclasses implementadas no Sistema, mas outras classes podem ser inseridas no pacote **element**, acrescentando outros tipos de elementos no Sistema.

Quando uma expressão do tipo **SIN(C1)+(230*10)** (sendo C1 igual a um valor numérico) começa a ser resolvida, todos os valores numéricos e variáveis são substituídos por identificadores que são relacionados com seu valor através de uma Tabela de Símbolos [AHO77].

A expressão acima tomaria a forma **SIN(E1)+(E2*E3)** e uma tabela de símbolos identificaria E1=C1, E2=NumericElement (valor 230) e E3=NumericElement (valor 10). Como o valor de C1 também é um símbolo e supondo que seu valor seja 180, a tabela de símbolos ficaria da seguinte forma: E1=NumericElement (valor 180), E2=NumericElement (valor 230) e E3=NumericElement (valor 10).

Se o canal for uma imagem, um som ou outro tipo de elemento, a tabela de símbolos conterá um objeto instanciado da classe pertinente.

A identificação das funções por parte da **análise semântica** é realizada observando as funções disponíveis nas classes do pacote **expression.function**. As classes desse pacote são subclasses da classe **Function** e implementam a classe **FunctionInterface**. A herança da classe **Function** determina que todas as subclasses devem ter um nome (atributo **name**) que é o nome que o usuário deve utilizar nas expressões. O método **solve()**, explicitado na classe de interface **FunctionInterface** deve ser implementado por todas classes de função do pacote.

Através das implementações do método **solve()**, pode-se programar qualquer algoritmo de Fusão de Sensores, pois a entrada do método é um conjunto (*array*) de objetos do tipo **Element** e a resposta do método é um único objeto do tipo **Element**.

Portanto, pode-se implementar qualquer tipo de combinação com qualquer tipo de elemento (NumericElement, StringElement, ImageElement, etc.) dentro do método **solve()**. Se a combinação dos dados resultar em mais de um elemento, pode-

se retornar um objeto da classe **SetElement** que agrega um conjunto de objetos instanciados das subclasses de **Element**.

A figura 3.14 mostra o Diagrama de classes do pacote **expression.function**. Nele não estão representadas todas as subclasses de **Function**, pois são muitas e tendem a crescer na medida em que o Sistema incorporar novos algoritmos de fusão. Uma lista das funções implementadas pode ser vista no Apêndice D, com os tipos de elementos de entrada e saída e qual modificação a função realiza nos elementos. Neste mesmo apêndice encontra-se uma lista com as constantes definidas no Sistema que podem ser utilizadas nas expressões (PI, por exemplo).

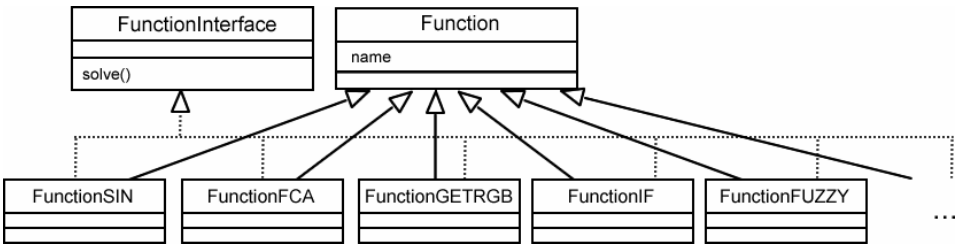


Fig. 3.14 – Diagrama de classes do pacote expression.function.

A figura 3.15 mostra o código JAVA da classe **FunctionInterface** e a figura 3.16 mostra o código da classe **Function**. A figura 3.17 ilustra a implementação da função seno através da classe **FunctionSIN**: o vetor de entrada (**elements**) deve possuir apenas um elemento e a resposta do método é um elemento do tipo **NumericElement** com o seu valor igual ao seno do valor do elemento de entrada.

```
package br.eng.rsalustiano.sensorfusion.expression.function;

import br.eng.rsalustiano.sensorfusion.element.Element;

public interface FunctionInterface {

	public abstract Element solve( Element[] elements );

}
```

Fig. 3.15 – Código JAVA da interface FunctionInterface.

```
package br.eng.rsalustiano.sensorfusion.expression.function;

public abstract class Function extends Object {

        private String name = new String();

        public Function( String name ) {
                this.setName( name );
        }

        public String getName() {
                return this.name;
        }

        public Function setName( String name ) {
                this.name = name;
                return this;
        }

        public abstract Element solve( Element[] elements );

}
```

Fig. 3.16 – Código JAVA da superclasse Function.

```
package br.eng.rsalustiano.sensorfusion.expression.function;

import br.eng.rsalustiano.sensorfusion.element.*;

public class FunctionSIN extends Function implements FunctionInterface {

        public FunctionSIN() {
                super( "SIN" );
        }

        public Element solve( Element[] elements ) {
                if ( elements.length != 1 )
                        return null;
                if ( elements[ 0 ] instanceof NumericElement )
                        return new NumericElement( Math.sin( ((NumericElement)
                                                elements[0]).getDoubleValue() ) );
                return null;
        }
}
```

Fig. 3.17 – Exemplo de classe que implementa a função seno.

Vistas as estruturas dos objetos, suas relações e a gramática para as expressões, pode-se agora entender quais os passos programados na classe **Expression** que realizam a análise léxica, sintática e semântica das expressões.

Os passos 1 a 14 a seguir mostram como é realizada a resolução das expressões:

1) Remover os espaços em branco fora das aspas duplas, isto é, fora das *tokens* do tipo STRING;

2) Executar o analisador léxico para identificar as *tokens* de acordo com a gramática da Tabela 3.7;

3) Verificar se os parênteses encontram-se em pares, ou seja, para cada "(" deve existir um ")";

4) Substituir os operadores binários ":" pelos intervalos de valores correspondentes;

5) Substituir as variáveis e os números identificados pelo analisador léxico por símbolos (letra "E" seguida de um número) que devem ser inseridos na Tabela de Símbolos com seus respectivos elementos (objetos das subclasses de **Element**);

6) Inserir um "(" antes e um ")" depois de cada função;

7) Inserir um "(" antes e um ")" depois de cada símbolo da expressão;

8) Inserir um "(" antes dos operadores binários de esquerda e um ")" depois de cada operando da operação correspondente;

9) Inserir um "(" antes dos operandos binários de direita e um ")" depois de cada operador da operação correspondente;

10) Inserir um "(" antes do operando esquerdo de uma operação binária e ")" depois de cada operando da operação correspondente;

11) Partindo da primeira *token* da expressão até a última, atribuir o número 1 (valor **cont**) para a primeira *token* e em seguida incrementar esse valor de uma unidade para todo abre parênteses e decrementar de uma unidade para todo fecha parênteses;

12) Remover todas as *tokens* abre e fecha parênteses;

13) Iniciar o processo de resolução da expressão resolvendo os agrupamentos com o mesmo número **cont** partindo do maior. A cada resolução, seja de uma função, de uma operação binária ou unária, decrementar de uma unidade o valor do **cont** do elemento de retorno da resolução.

14) Se houver algum erro em alguns dos itens acima, reportar o erro; caso contrário, devolver o último (e único) objeto **Element** que sobrou (com contador 0) da resolução da expressão do item 13.

A expressão **SIN(C1*2)+IF(C2>30.2,2,3)** será utilizada como exemplo, dados os valores dos canais C1 = 31,5 e C2 = 31. As figuras 3.18, 3.19, 3.20 e 3.21 mostram a seqüência de execução da expressão acima. A figura 3.22 mostra a execução da expressão de exemplo através da *Abstract Syntax Tree* (AST), ou seja, a árvore sintática da expressão.

PASSOS 1, 2, 3 e 4			**PASSO 5**			**PASSOS 6, 7, 8, 9, 10 e 11**			**PASSO 12**			**PASSO 13**		
N	*token*	cont	N	*token*	cont	N	*token*	cont	N	*token*	cont	N	*token*	cont
1	SIN	0	1	SIN	0	1	(	1	1	SIN	3	1	SIN	3
2	(	0	2	(	0	2	(	2	2	E8	6	2	E8	5
3	C1	0	3	E8	0	3	SIN	3	3	*	5	3	*	5
4	*	0	4	*	0	4	(	3	4	E21	6	4	E21	6
5	2	0	5	E21	0	5	(	4	5	+	2	5	+	2
6	)	0	6	)	0	6	(	5	6	IF	3	6	IF	3
7	+	0	7	+	0	7	E8	6	7	E9	6	7	E9	6
8	IF	0	8	IF	0	8	)	6	8	>	5	8	>	5
9	(	0	9	(	0	9	*	5	9	E22	6	9	E22	6
10	C2	0	10	E9	0	10	(	5	10	,	4	10	,	4
11	>	0	11	>	0	11	E21	6	11	E23	5	11	E23	5
12	30.2	0	12	E22	0	12	)	6	12	,	4	12	,	4
13	,	0	13	,	0	13	)	5	13	E24	5	13	E24	5
14	2	0	14	E23	0	14	)	4						
15	,	0	15	,	0	15	)	3						
16	3	0	16	E24	0	16	+	2						
17	)	0	17	)	0	17	(	2						
						18	IF	3						
						19	(	3						
						20	(	4						
						21	(	5						
						22	E9	6						
						23	)	6						
						24	>	5						
						25	(	5						
						26	E22	6						
						27	)	6						
						28	)	5						
						29	,	4						
						30	(	4						
						31	E23	5						
						32	)	5						
						33	,	4						
						34	(	4						
						35	E24	5						
						36	)	5						
						37	)	4						
						38	)	3						
						39	)	2						

Fig. 3.18 – Passos do algoritmo para resolução da expressão de exemplo SIN(C1*2)+IF(C2>30.2,2,3) – Parte 1.

PASSO 13			PASSO 13			PASSO 13			PASSO 13			PASSO 13		
N	*token*	cont	N	*token*	cont	N	*token*	cont	N	*token*	cont	N	*token*	cont
1	SIN	3	1	SIN	3	1	SIN	3	1	SIN	3	1	SIN	3
2	E8	5	2	E8	5	2	**E8**	**5**	2	E26	4	2	E26	4
3	*	5	3	*	5	3	*****	**5**	3	+	2	3	+	2
4	E21	5	4	E21	5	4	**E21**	**5**	4	IF	3	4	IF	3
5	+	2	5	+	2	5	+	2	5	**E9**	**5**	5	E27	4
6	IF	3	6	IF	3	6	IF	3	6	**>**	**5**	6	,	4
7	**E9**	**6**	7	E9	5	7	E9	5	7	**E22**	**5**	7	**E23**	**5**
8	>	5	8	>	5	8	>	5	8	,	4	8	,	4
9	E22	6	9	**E22**	**6**	9	E22	5	9	E23	5	9	E24	5
10	,	4	10	,	4	10	,	4	10	,	4			
11	E23	5	11	E23	5	11	E23	5	11	E24	5			
12	,	4	12	,	4	12	,	4						
13	E24	5	13	E24	5	13	E24	5						

Fig. 3.19 – Passos do algoritmo para resolução da expressão de exemplo SIN(C1*2)+IF(C2>30.2,2,3) – Parte 2.

PASSO 13			PASSO 13			PASSO 13			PASSO 13			PASSO 13		
N	*token*	cont	N	*token*	cont	N	*token*	cont	N	*token*	cont	N	*token*	cont
1	SIN	3	1	SIN	3	1	SIN	3	1	**SIN**	**3**	1	E28	2
2	E26	4	2	**E26**	**4**	2	E26	3	2	**E26**	**3**	2	+	2
3	+	2	3	+	2	3	+	2	3			3	**IF**	**3**
4	IF	3	4	IF	3	4	IF	3	4	IF	3	4	**E27**	**3**
5	E27	4	5	E27	4	5	**E27**	**4**	5	E27	3	5	**E23**	**3**
6	,	4	6	,	4	6	**,**	**4**	6	E23	3	6	**E24**	**3**
7	E23	4	7	E23	4	7	**E23**	**4**	7	E24	3			
8	,	4	8	,	4	8	**,**	**4**						
9	**E24**	**5**	9	E24	4	9	**E24**	**4**						

Fig. 3.20 – Passos do algoritmo para resolução da expressão de exemplo SIN(C1*2)+IF(C2>30.2,2,3) – Parte 3.

PASSO 13			PASSO 13			PASSO 14		
1	**E28**	**2**	1	**E30**	**1**	1	**E30**	**0**
2	**+**	**2**						
3	**E29**	**2**						

Fig. 3.21 – Passos do algoritmo para resolução da expressão de exemplo SIN(C1*2)+IF(C2>30.2,2,3) – Parte 4.

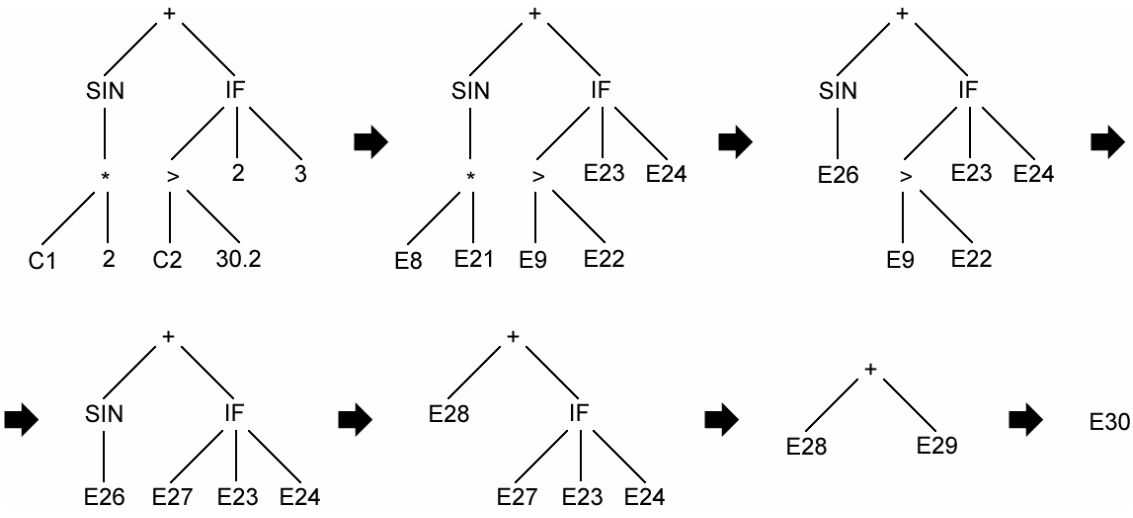


Fig. 3.22 – Execução da expressão de exemplo SIN(C1*2)+IF(C2>30.2,2,3) através da sua *Abtract Syntax Tree*.

A resolução dos agrupamentos de *tokens* é realizada da seguinte forma. Se o agrupamento tiver apenas um elemento, o retorno é o mesmo elemento (com o valor **cont** diminuído de uma unidade). Se o agrupamento iniciar com uma *token* de função, o analisador semântico irá resolver a função de acordo com seu nome e as demais *tokens* (parâmetros da função) agrupadas, retornando um novo elemento – resultado da função – na pilha de elementos. Se o agrupamento iniciar ou terminar com um operador unário, o analisador irá resolver a operação e retornar um novo elemento. O último caso válido é se a segunda *token* for um operador binário. Nesse caso, o analisador semântico resolve a operação com o primeiro e o terceiro elementos do agrupamento.

3.2.2.3 – Banco de Dados

O objetivo principal do Banco de Dados no Sistema é armazenar as informações dos módulos e seus canais, as tabelas contendo os mecanismos de fusão dos dados dos sensores e os dados recebidos dos monitores.

O MySQL[2] foi o sistema de gerenciamento de Banco de Dados (SGBD) utilizado no Sistema. A escolha pela utilização deste gerenciador pode ser explicada por ele ser de distribuição livre, fácil instalação e manutenção, ser *multithread* (múltiplos acessos à base de dados são realizados em paralelo), multi-usuário e o esquema de consulta ser baseado na linguagem SQL (*Structure Query Language*). A sua integração com a tecnologia JAVA também favorece e facilita a sua utilização através do uso de um *driver* de acesso denominado mysql-connector[3].

A modelagem da estrutura das tabelas utilizadas no Banco de Dados seguiu três fases: 1) Determinação das entidades envolvidas no sistema e seus atributos; 2) Criação de um diagrama entidade-relacionamento (MER); 3) Definição da estrutura física das tabelas.

Estas três fases para o projeto de Banco de Dados é uma simplificação do modelo proposto em [ELM89]. Na fase de determinação das entidades envolvidas, e seus atributos, são considerados os requisitos do sistema e é feita uma análise dos

[2] http://www.mysql.com
[3] http://www.mysql.com/products/connector/j/

objetivos que se quer atingir, no caso deste sistema, armazenar as informações dos usuários, dos módulos e seus canais, a descrição das tabelas contendo os mecanismos de fusão e o armazenamento dos dados dos sensores.

Na segunda fase do projeto, deve-se criar um diagrama evidenciando o relacionamento entre as entidades. Para isso, utilizou-se o Modelo entidade-relacionamento (MER) que é uma descrição gráfica em alto nível que representa as relações entre as entidades determinadas na primeira fase do projeto. São definidos as chaves primárias das entidades (atributos que identificam as tuplas de forma única na tabela) e o tipo de relacionamento entre elas.

A terceira e última fase do projeto é a determinação das tabelas (colunas e tipo de dados de cada coluna) do Banco de Dados a partir do MER. Através da linguagem SQL, as tabelas são modeladas e inseridas no Banco de Dados.

A figura 3.23 ilustra o diagrama entidade-relacionamendo utilizado no projeto do Banco de Dados do Sistema. As instruções SQL para a criação das tabelas podem ser vistas no Apêndice B.

Um usuário do sistema possui um ou mais monitores aos quais ele pode realizar a configuração e as operações de consulta e monitoramento. Cada usuário possui um nome, um nome de usuário (*login*) e uma senha. Para acessar a interface do usuário através da Internet é preciso que o usuário forneça o nome do usuário e a senha.

Os monitores possuem nomes únicos que os identificam para o usuário. Relacionado a cada monitor existem canais e podem existir tarefas, tabelas e dados armazenados dos sensores (leituras). Cada monitor tem um ou mais canais (CANAL MONITOR) que podem ser do tipo áudio, imagem ou numérico. Cada uma dessas especializações de canal possui atributos específicos: os canais de imagem possuem a altura e largura da imagem e a sua resolução, enquanto que os canais numéricos possuem a base numérica (2 a 36), exatidão, valor máximo e valor mínimo.

Os canais de áudio e de imagem possuem tipos específicos (GIF, JPG, etc para imagens e WAV, AU, etc. para áudio) que são cadastrados nas tabelas TIPO AUDIO e TIPO IMAGEM, respectivamente.

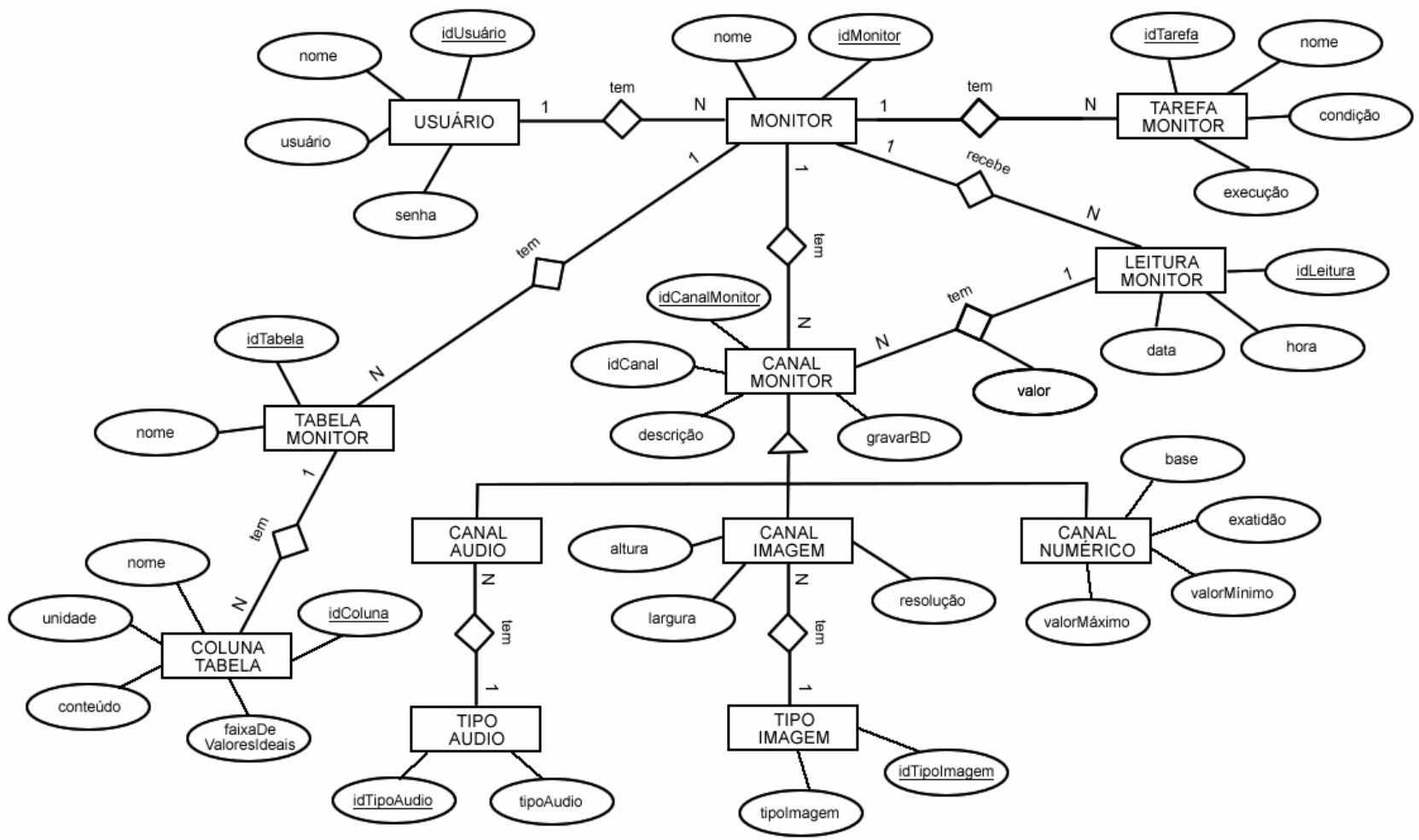


Fig. 3.23 – Modelo entidade-relacionamento (MER) utilizado no projeto do Banco de Dados do Sistema.

Cada tabela possui uma ou mais colunas (COLUNA TABELA) que tem os atributos nome, uma unidade, o conteúdo que é a expressão de manipulação do dado do sensor e a faixa de valores ideais que é uma expressão *booleana* que faz com que a visualização do dado durante a montagem da tabela fique em destaque caso o resultado da expressão seja falso. Um monitor recebe uma ou mais leituras (LEITURA MONITOR) que consta de um conjunto de dados identificados pelo número do monitor (idMonitor), número do canal, data e hora. Como cada canal recebe um dado de tipo diferente (número, imagem, som, etc.), o atributo *valor* que armazena os dados dos canais deve ser do tipo BLOB (*Binary Large Object*) e a forma com que o conteúdo desse atributo é lido depende do tipo de especialização utilizado na especificação do canal (CANAL ÁUDIO, CANAL IMAGEM, CANAL NUMÉRICO e outros que possam ser adicionados ao Sistema).

Através do atributo gravarDB da entidade CANAL MONITOR é possível determinar se o valor recebido do canal do monitor através do documento XML deve ou não ser armazenado no Banco de Dados.

3.2.2.4 – Servidor HTTP

O servidor HTTP é responsável por disponibilizar a interface do usuário na Internet. O servidor utilizado é o Apache Tomcat[4], uma distribuição gratuita que permite uma integração com a arquitetura JAVA através da tecnologia JSP.

Um arquivo JSP nada mais é do que um arquivo HTML que possui uma *tag* especial (<% %>) que permite que códigos JAVA sejam executados do lado do servidor antes do envio da página HTML para o navegador do usuário. Isso permite que as páginas HTML sejam dinâmicas, isto é, possam ser “escritas” de acordo com as requisições e as configurações do usuário.

Dentro da *tag* <% %> pode-se escrever qualquer código na linguagem JAVA, como o exemplo da figura 3.24. A figura 3.25 mostra o código HTML resultante do processamento do JSP da figura 3.24.

[4] http://tomcat.apache.org/

A *tag* **<%@ page import="" %>** deve ser utilizada no início do arquivo JSP para importar as classes que serão utilizadas no código JAVA dentro do arquivo JSP.

```
<%@ page import="br.eng.rsalustiano.sensorfusion.business.Monitor" %>
<HTML>
	<HEAD>
		<TITLE> Página de teste de JSP </TITLE>
	</HEAD>
	<BODY>
		<%
			Monitor m = new Monitor( 10 );
			out.write( "Monitor: " + m.getName() );
		%>
	</BODY>
</HTML>
```

Fig. 3.24 – Arquivo JSP de exemplo.

```
<HTML>
	<HEAD>
		<TITLE> Página de teste de JSP </TITLE>
	</HEAD>
	<BODY>
		Monitor: Planta-Ficus
	</BODY>
</HTML>
```

Fig. 3.25 – Arquivo HTML resultado do processamento do arquivo JSP da figura 3.26.

A configuração do Tomcat é toda realizada no arquivo **server.xml** presente no diretório **conf** da raiz da instalação do servidor. A figura 3.26 mostra a *tag* Connector, uma das mais importantes do arquivo de configuração do Tomcat. A porta configurada nesse exemplo para a execução do servidor é a 8080, o número máximo de usuários conectados simultaneamente é 150 e o *timeout* da conexão é 20 minutos.

```
<Connector	port="8080"
		maxHttpHeaderSize="8192"
		maxThreads="150"
		minSpareThreads="25"
		maxSpareThreads="75"
		enableLookups="false"
		redirectPort="8443"
		acceptCount="100"
		connectionTimeout="20000"
		disableUploadTimeout="true"
/>
```

Fig. 3.26 – *Tag* Connector do arquivo de configuração server.xml.

Os arquivos JSP e HTML devem estar dentro de um diretório indicado no arquivo de configuração **server.xml**. Dentro desse diretório deve existir três outros diretórios: **WEB-INF**, **classes** e **lib**.

Todas as classes JAVA (arquivos .class) utilizadas pelos arquivos JSP devem estar no diretório **classes**. O diretório **lib** deve conter todas os pacotes de classes compactados (arquivos .jar). Dentro do diretório **WEB-INF** deve existir o arquivo **web.xml** que contém especificações sobre o projeto.

Os arquivos JAR necessários para o funcionamento das classes do projeto do Sistema de Monitoramento de Ambientes estão relacionados na Tabela 3.8 juntamente com uma descrição do seu conteúdo. Todas as API fornecidas através desses arquivos são de utilização livre.

Tabela 3.8 – Arquivos JAR utilizados no Sistema

Arquivo	Descrição
activation.jar **mail.jar** **smtp.jar**	API contendo classes utilizadas para o envio de e-mails[5].
flanagan.jar	API contendo classes que possuem métodos que realizam cálculos avançados (derivadas, integrais, etc) e cálculos estatísticos[6].
poi-2.5-final-20040302.jar	API contendo classes para criação e manipulação de arquivos XLS[7].
mysql-connector-java-3.1.7-bin.jar	Arquivo contendo o conector que faz a ligação entre a arquitetura JAVA e o Banco de Dados MySQL[8].

3.2.3 – Interface do usuário

A interface do usuário é composta de um conjunto de páginas HTML que podem ser acessadas através da Internet com o auxílio de navegadores. Através delas, o usuário pode configurar os módulos, canais, tabelas, colunas e tarefas, além de realizar consulta aos dados dos sensores e executar o monitoramento dos ambientes.

O usuário deve seguir os passos da Tabela 3.9 para configurar e utilizar o Sistema de Monitoramento de Ambientes. Antes de iniciar o envio dos dados dos

[5] http://java.sun.com/products/javamail/
[6] http://www.ee.ucl.ac.uk/~mflanaga/java/
[7] http://jakarta.apache.org/poi/
[8] http://dev.mysql.com/downloads/connector/

monitores para o servidor, é preciso configurar os monitores e os seus respectivos canais.

Tabela 3.9 – Passos para a configuração e utilização do Sistema

Passo	Ação	Local
1	Autenticação do Usuário no Sistema (processo de *login*)	Interface do usuário
2	Cadastro dos Monitores e dos canais	Interface do usuário
3	Configuração das mensagens (XML) para envio dos dados dos Monitores para o Servidor, verificando o número do monitor (idMonitor) fornecido pelo Sistema e o número dos canais indicados na configuração dos monitores (item 2).	Monitores
4	Configuração das Tabelas e Tarefas	Interface do usuário
5	Monitoramento e Consulta dos dados armazenados no Banco de Dados (Histórico dos sensores)	Interface do usuário

Cada usuário possui um nome de usuário (*login*) e uma senha para poder utilizar a interface. Esses valores devem ser fornecidos toda vez que o usuário acessar a página através do endereço eletrônico http://vonbraun.lpm.fee.unicamp.br:8080/sensorfusion.

Ao acessar o conteúdo da interface *web*, o usuário tem quatro opções: **Configurações**, **Monitoramento**, **Consultas** e **Sair** presentes no canto direito superior, conforme detalhe da figura 3.27.

3.2.3.1 – Menu Configurações

Acessando o menu de **Configurações**, os monitores do usuário, suas tabelas e tarefas são exibidas na coluna esquerda da interface. O usuário pode configurar os monitores acessando o item **Configurar** de cada módulo; pode configurar as tabelas clicando sobre o nome da tabela (TAB01 e TAB02 no detalhe da figura 3.27) ou criar uma nova tabela no item **Nova Tabela**; e pode configurar uma tarefa clicando no nome da tarefa (E-mail no detalhe da figura 3.27) ou criar uma nova tabela através do item **Nova Tarefa**.

Fig. 3.27 – Interface do usuário do Sistema de Monitoramento de Ambientes.
Destaque para o menu do usuário (parte superior direita) e configuração dos módulos (esquerda).

A figura 3.28 mostra um exemplo de configuração de um monitor e seus canais. O monitor é identificado como **Lumi-Moni** e possui 4 canais: três numéricos e um do tipo imagem. Os canais numéricos são todos números de base 10 e devem ser gravados no Banco de Dados. O canal imagem tem largura de 640 *pixels* e altura de 480 *pixels* e não deve ser gravado no Banco de Dados. Nenhum canal tipo áudio foi cadastrado.

Na figura 3.29 é exibida uma configuração de tabela, no caso, a tabela **Temperaturas** do monitor **Câmara Climática**.

Para facilitar a configuração de tabelas e o conteúdo das colunas, pode-se visualizar os canais cadastrados de cada monitor com suas descrições clicando no botão **Canais disponíveis** na interface de configuração da tabela. Uma nova janela se abrirá contendo a lista de canais. Através do botão **Funções disponíveis**, pode-se visualizar a lista de funções cadastradas no Sistema.

A figura 3.30 ilustra a interface de configuração de uma tarefa. A tarefa selecionada tem o nome de **E-mail** e, toda vez que o servidor receber uma mensagem do módulo **Lumi-Moni**, se a condição enunciada for verdadeira, a função SENDMAIL é executada, enviado um e-mail de acordo com a programação da função. Um detalhe sobre a função SENDMAIL e outras funções disponíveis no Sistema pode ser vista no Apêndice D.

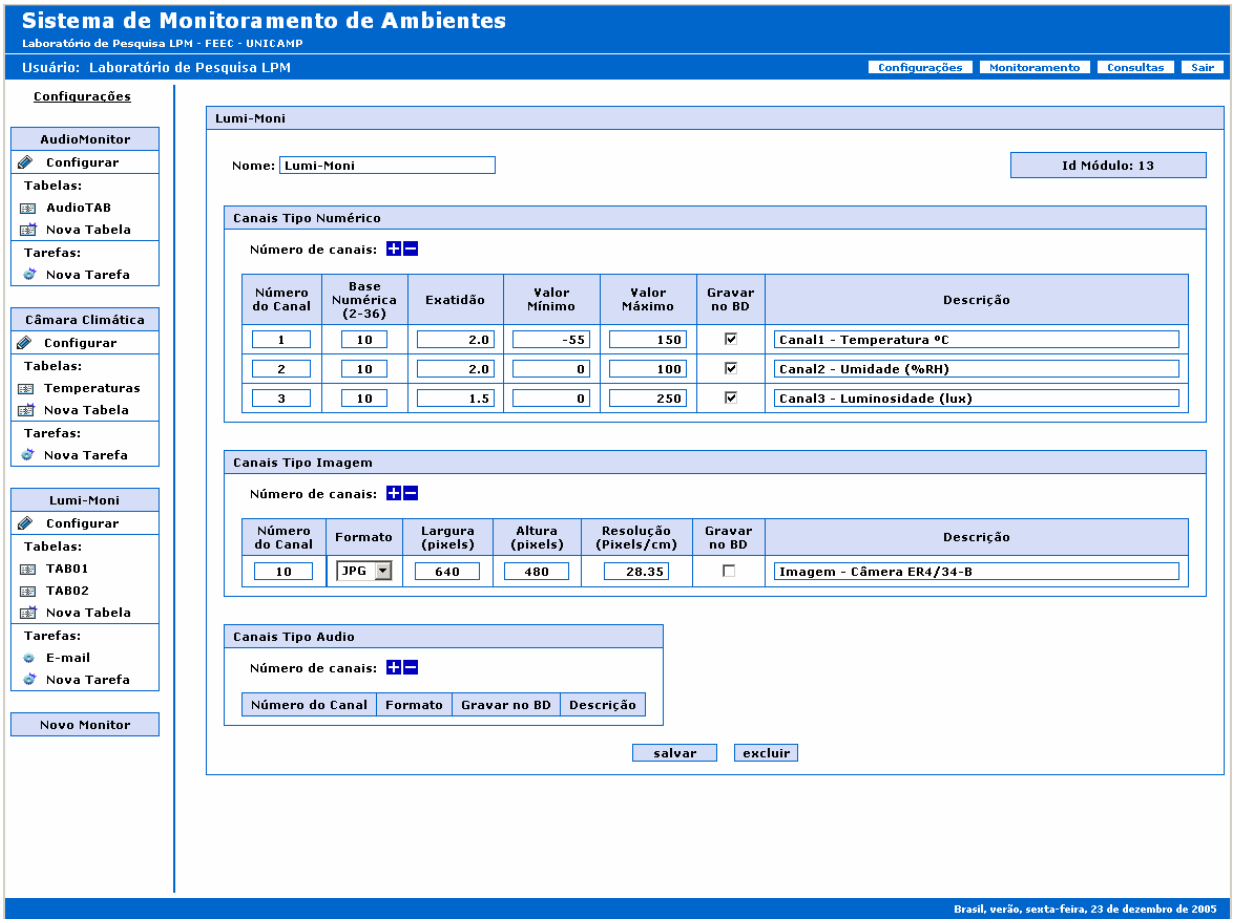


Fig. 3.28 – Interface do usuário do Sistema de Monitoramento de Ambientes. Configuração de um monitor e seus canais.

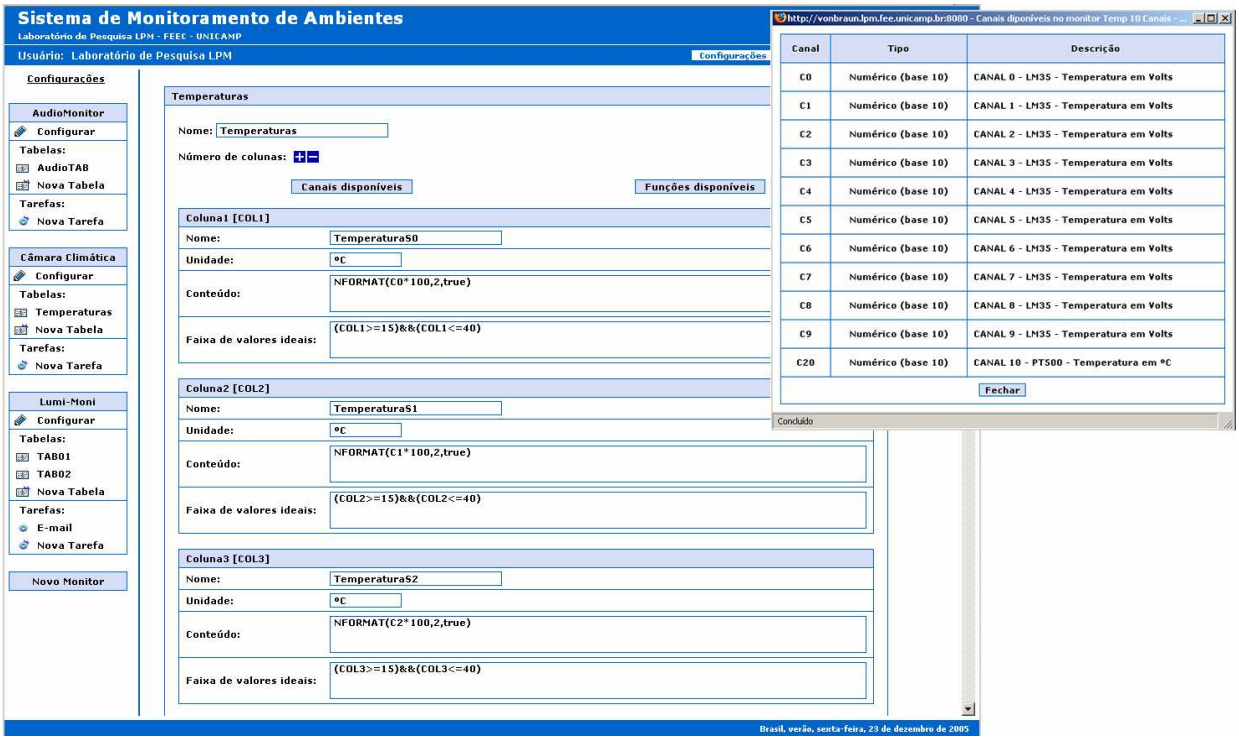


Fig. 3.29 – Interface do usuário do Sistema de Monitoramento de Ambientes. Configuração de tabelas. Detalhe para a janela com a lista de canais disponíveis para o módulo selecionado.

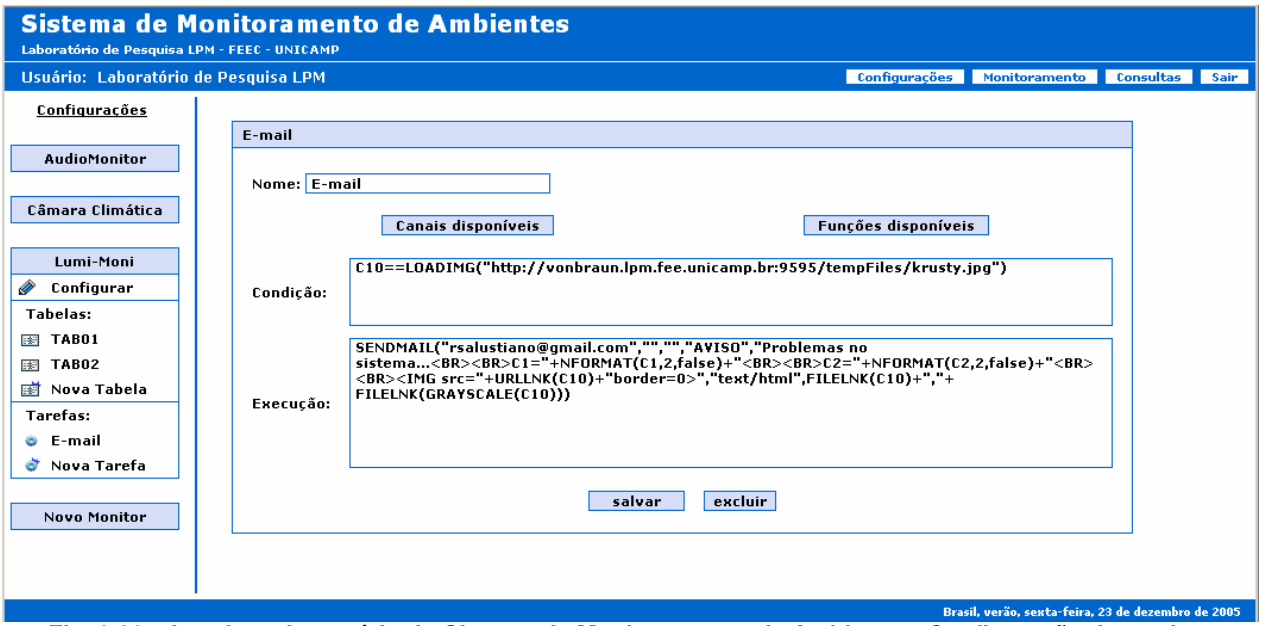


Fig. 3.30 – Interface do usuário do Sistema de Monitoramento de Ambientes. Configuração de tarefas.

3.2.3.2 – Menu Monitoramento

O menu de **Monitoramento** permite que o usuário acompanhe o recebimento de dados dos monitores, executando, ao mesmo tempo, os algoritmos de Fusão de Sensores programados nas tabelas.

A figura 3.31 mostra a interface de configuração de um monitoramento para a tabela **Fícus-TAB01** do monitor **Fícus**. Diferentemente dos exemplos anteriores, somente o monitor **Fícus** está disponível na interface porque se trata de outro usuário do Sistema, que só possui esse monitor cadastrado.

Para iniciar o monitoramento, deve-se escolher a quantidade de leituras que se quer monitorar por vez: ir acumulando os registros (leituras) indefinidamente (isto é, ir exibindo todas as leituras a partir do momento de início do monitoramento) ou exibir apenas as últimas N leituras, onde N pode ser configurado de acordo com a necessidade do usuário.

Outro parâmetro que deve ser identificado é o intervalo de monitoramento, ou seja, o intervalo que o *browser* do usuário faz a checagem se existe uma nova leitura disponível no servidor. Esse parâmetro poderia ser eliminado se a interface fosse programada em uma *applet* em JAVA, que permite a implementação de uma conexão direta entre o cliente e o servidor. Contudo, optou-se por não executar nenhum tipo de aplicativo no *browser* do cliente, pois muitas vezes a instalação da JVM na máquina do cliente requer privilégios do administrador da rede e em alguns computadores antigos a velocidade de processamento da JVM não é satisfatória.

O último passo para se dar início ao monitoramento é escolher se a visualização dos dados será feita através de tabelas ou gráfico. Se for feita por tabela, deve-se identificar qual a ordem (crescente ou decrescente) que os dados deverão aparecer e quais as colunas; em seguida é só clicar no botão **iniciar monitoramento por Tabela** que uma nova janela contendo a tabela selecionada se abrirá. Se o usuário preferir realizar o monitoramento através de um gráfico, ele deverá selecionar a variável do eixo X e uma ou mais variáveis para o eixo Y; em seguida é só clicar no botão **iniciar monitoramento por Gráfico** que uma nova janela contendo o gráfico se abrirá. Também é possível optar pelo tamanho do gráfico, indicando a altura e a largura da imagem que representa o gráfico.

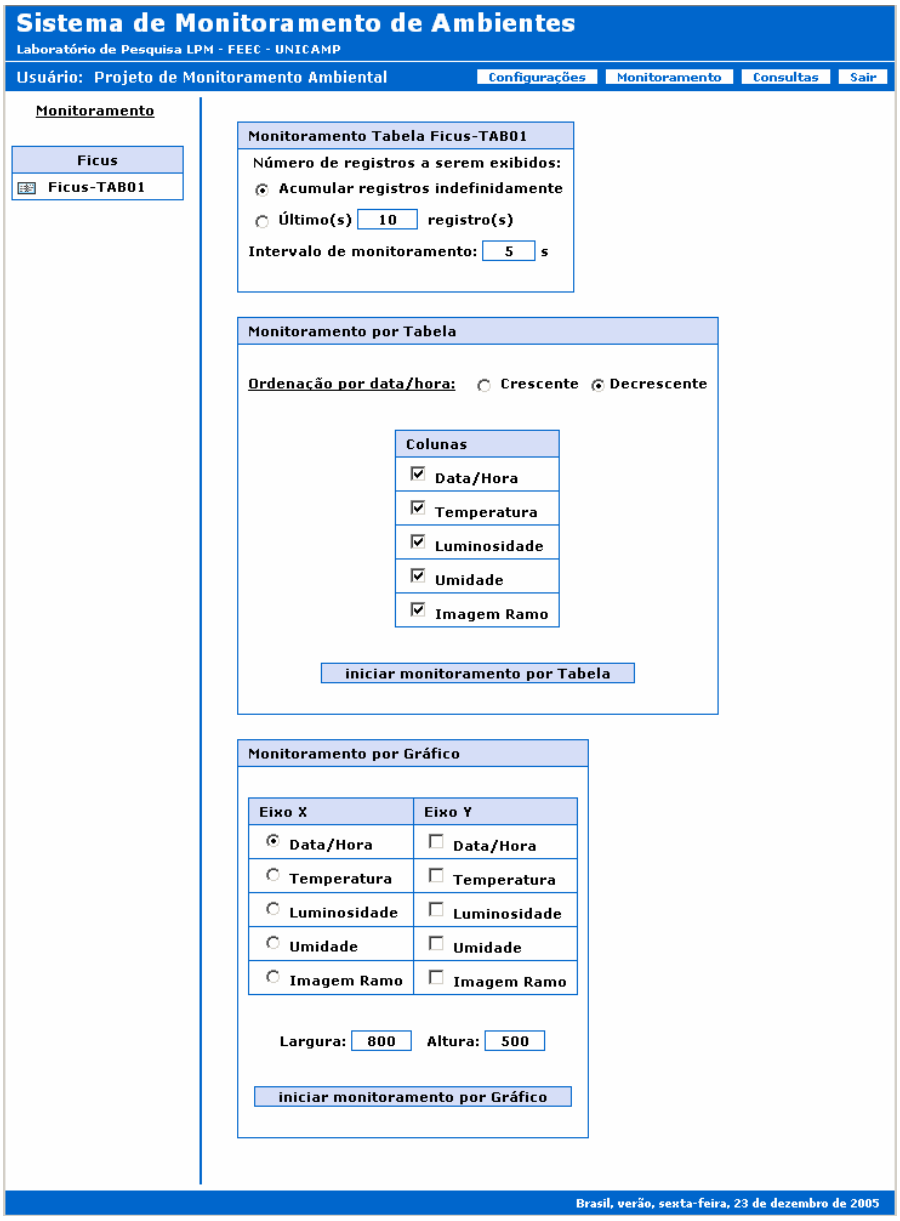


Fig. 3.31 – Interface do usuário do Sistema de Monitoramento de Ambientes. Monitoramento.

As tabelas e os gráficos gerados através do Monitoramento são idênticos aos gerados na consulta, mas durante o Monitoramento há atualização dos dados. Exemplos de gráficos e tabelas serão vistas a seguir, no item Menu Consulta, assim como um detalhamento de como os dados são organizados nos gráficos.

O sistema de monitoramento é suficientemente flexível, permitindo a realização de diversos monitoramentos simultâneos através de tabelas e gráficos diferentes, com intervalos de monitoramento diferentes.

3.2.3.3 – Menu Consulta

O menu de **Consulta** permite que o usuário realize pesquisas nos dados dos sensores recebidos e armazenados no Banco de Dados do servidor.

O procedimento de consulta é semelhante ao de monitoramento. A figura 3.32 ilustra a interface do usuário no estado de consulta dos dados do monitor **Fícus** através da tabela **Fícus-TAB01**. Pode-se realizar a consulta tanto através do fornecimento de datas e horas como realizá-la requisitando as últimas N leituras registradas no Banco de Dados.

Da mesma forma que o procedimento de Monitoramento, deve-se optar por realizar consultas através de tabelas ou gráficos. No caso das consultas através de tabelas, há a possibilidade de escolher o formato da tabela: HTML ou XLS (formato de arquivo de planilhas eletrônicas). Tabelas HTML geradas nas consultas são abertas em uma nova janela do *browser*, enquanto que uma consulta por tabelas XLS gera um arquivo neste formato que pode ser salvo na máquina do cliente.

A figura 3.33 mostra uma tabela gerada através de uma consulta das últimas 20 leituras do monitor **Câmara Climática** através da tabela **Temperaturas**. A figura 3.34 ilustra uma consulta feita através de um gráfico com os mesmos dados da consulta da figura 3.33.

O gráfico é uma imagem do tipo PNG que agrupa os dados selecionados para o eixo Y de acordo com as unidades. Desse modo, como todas as unidade das variáveis do gráfico da figura 3.34 são iguais (°C), elas ficaram agrupadas no mesmo eixo.

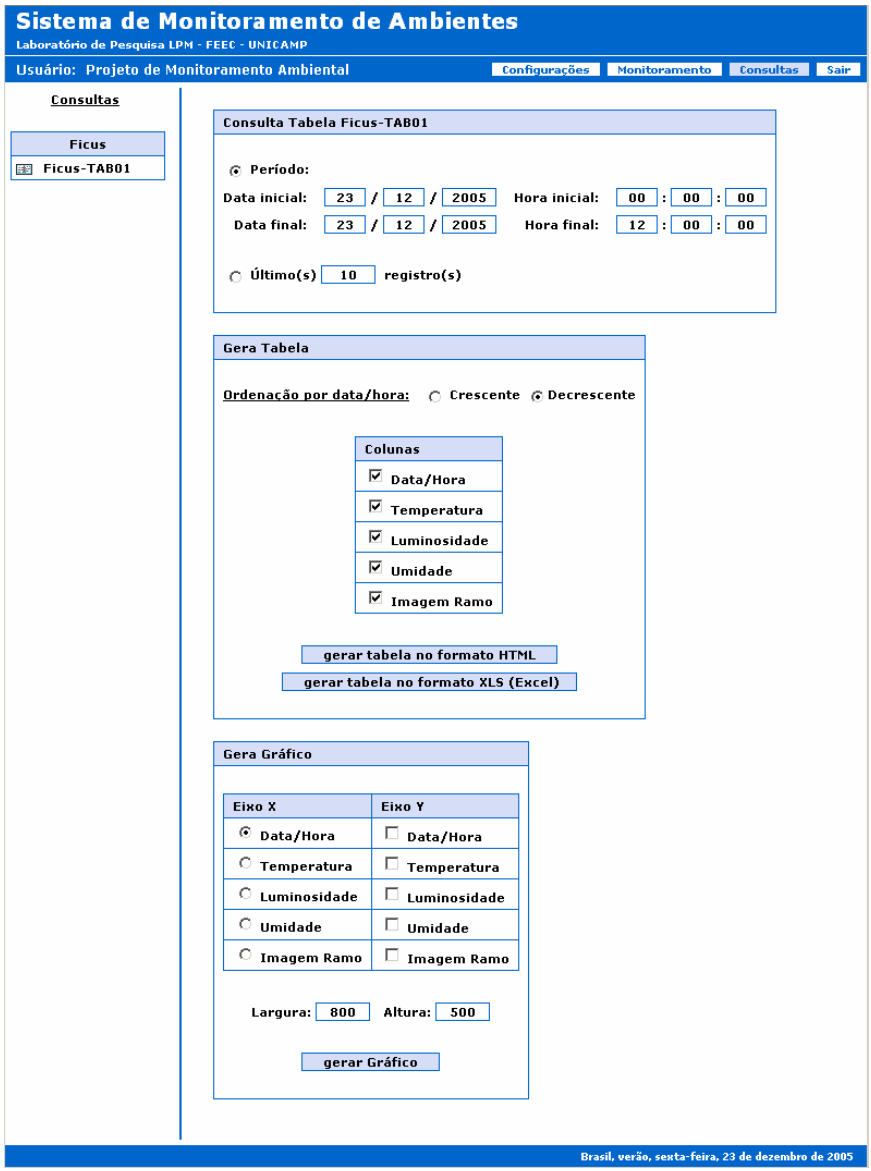
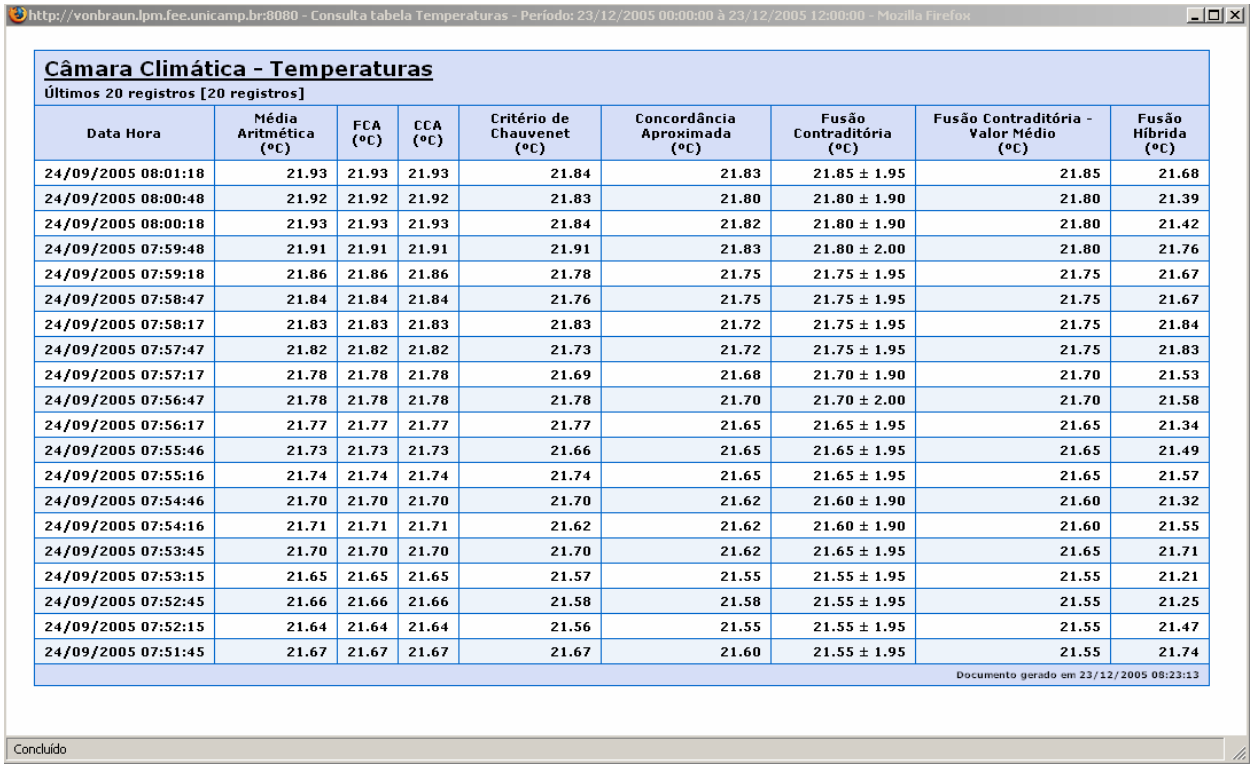


Fig. 3.32 – Interface do usuário do Sistema de Monitoramento de Ambientes. Consulta.

No exemplo da figura 3.35, as variáveis **Canal1** e **Canal2** estão em ℃, a variável **Canal3** em lux e a variável **Cor** não apresenta unidades. Este gráfico ilustra o agrupamento das variáveis nos eixos pelas suas unidades e abaixo de cada eixo existe uma barra colorida relacionando a cor da curva com o eixo no qual os valores devem ser lidos. Nesse mesmo gráfico, verifica-se a possibilidade de incluir na consulta (ou monitoramento) colunas de tabelas que resultam em valores discretos (no caso desse exemplo, a coluna **Cor**, que pode ser vermelho, roxo, azul, etc.). Uma outra observação

nesse gráfico é a visualização das barras de erro nos valores das colunas que produzem resultados numéricos com exatidão.


http://vonbraun.lpm.fee.unicamp.br:8080 - Consulta tabela Temperaturas - Período: 23/12/2005 00:00:00 à 23/12/2005 12:00:00 - Mozilla Firefox

Câmara Climática - Temperaturas

Últimos 20 registros [20 registros]

Data Hora	Média Aritmética (ºC)	FCA (ºC)	CCA (ºC)	Critério de Chauvenet (ºC)	Concordância Aproximada (ºC)	Fusão Contraditória (ºC)	Fusão Contraditória - Valor Médio (ºC)	Fusão Híbrida (ºC)
24/09/2005 08:01:18	21.93	21.93	21.93	21.84	21.83	21.85 ± 1.95	21.85	21.68
24/09/2005 08:00:48	21.92	21.92	21.92	21.83	21.80	21.80 ± 1.90	21.80	21.39
24/09/2005 08:00:18	21.93	21.93	21.93	21.84	21.82	21.80 ± 1.90	21.80	21.42
24/09/2005 07:59:48	21.91	21.91	21.91	21.91	21.83	21.80 ± 2.00	21.80	21.76
24/09/2005 07:59:18	21.86	21.86	21.86	21.78	21.75	21.75 ± 1.95	21.75	21.67
24/09/2005 07:58:47	21.84	21.84	21.84	21.76	21.75	21.75 ± 1.95	21.75	21.67
24/09/2005 07:58:17	21.83	21.83	21.83	21.83	21.72	21.75 ± 1.95	21.75	21.84
24/09/2005 07:57:47	21.82	21.82	21.82	21.73	21.72	21.75 ± 1.95	21.75	21.83
24/09/2005 07:57:17	21.78	21.78	21.78	21.69	21.68	21.70 ± 1.90	21.70	21.53
24/09/2005 07:56:47	21.78	21.78	21.78	21.78	21.70	21.70 ± 2.00	21.70	21.58
24/09/2005 07:56:17	21.77	21.77	21.77	21.77	21.65	21.65 ± 1.95	21.65	21.34
24/09/2005 07:55:46	21.73	21.73	21.73	21.66	21.65	21.65 ± 1.95	21.65	21.49
24/09/2005 07:55:16	21.74	21.74	21.74	21.74	21.65	21.65 ± 1.95	21.65	21.57
24/09/2005 07:54:46	21.70	21.70	21.70	21.70	21.62	21.60 ± 1.90	21.60	21.32
24/09/2005 07:54:16	21.71	21.71	21.71	21.62	21.62	21.60 ± 1.90	21.60	21.55
24/09/2005 07:53:45	21.70	21.70	21.70	21.70	21.62	21.65 ± 1.95	21.65	21.71
24/09/2005 07:53:15	21.65	21.65	21.65	21.57	21.55	21.55 ± 1.95	21.55	21.21
24/09/2005 07:52:45	21.66	21.66	21.66	21.58	21.58	21.55 ± 1.95	21.55	21.25
24/09/2005 07:52:15	21.64	21.64	21.64	21.56	21.55	21.55 ± 1.95	21.55	21.47
24/09/2005 07:51:45	21.67	21.67	21.67	21.67	21.60	21.55 ± 1.95	21.55	21.74

Documento gerado em 23/12/2005 08:23:13

Concluído

Fig. 3.33 – Resultado de uma consulta por tabela.

Todo o projeto visual e organizacional dos gráficos foi desenvolvido sem nenhuma API adicional além das nativas do JAVA. Foi testada a possibilidade de gerar o gráfico do lado do cliente, ao invés de enviar uma imagem, através de uma biblioteca criada em *javascript* denominada jsgraphics[9]. Contudo, o desempenho e limitação de memória dos *browsers* não permitia que os gráficos possuíssem muitos dados, fazendo, muitas vezes, com que o *browser* do usuário travasse ou emitisse mensagens de falta de memória.

[9] http://www.walterzorn.com/jsgraphics/jsgraphics_e.htm

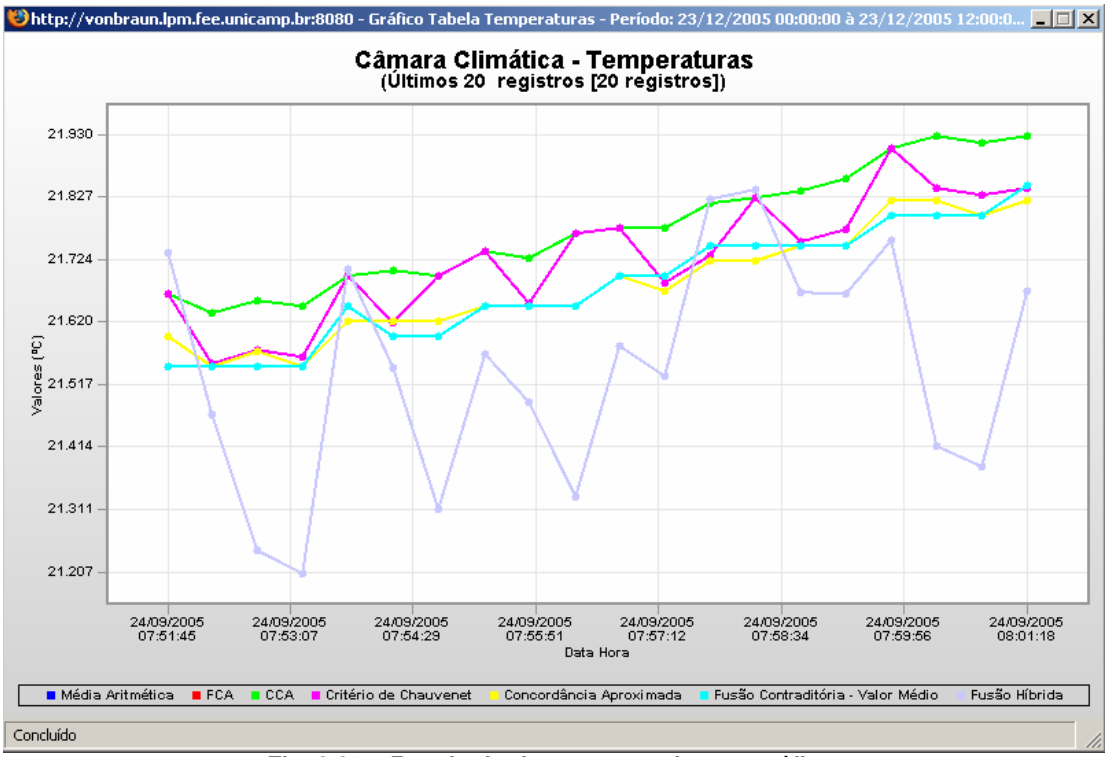


Fig. 3.34 – Resultado de uma consulta por gráfico.

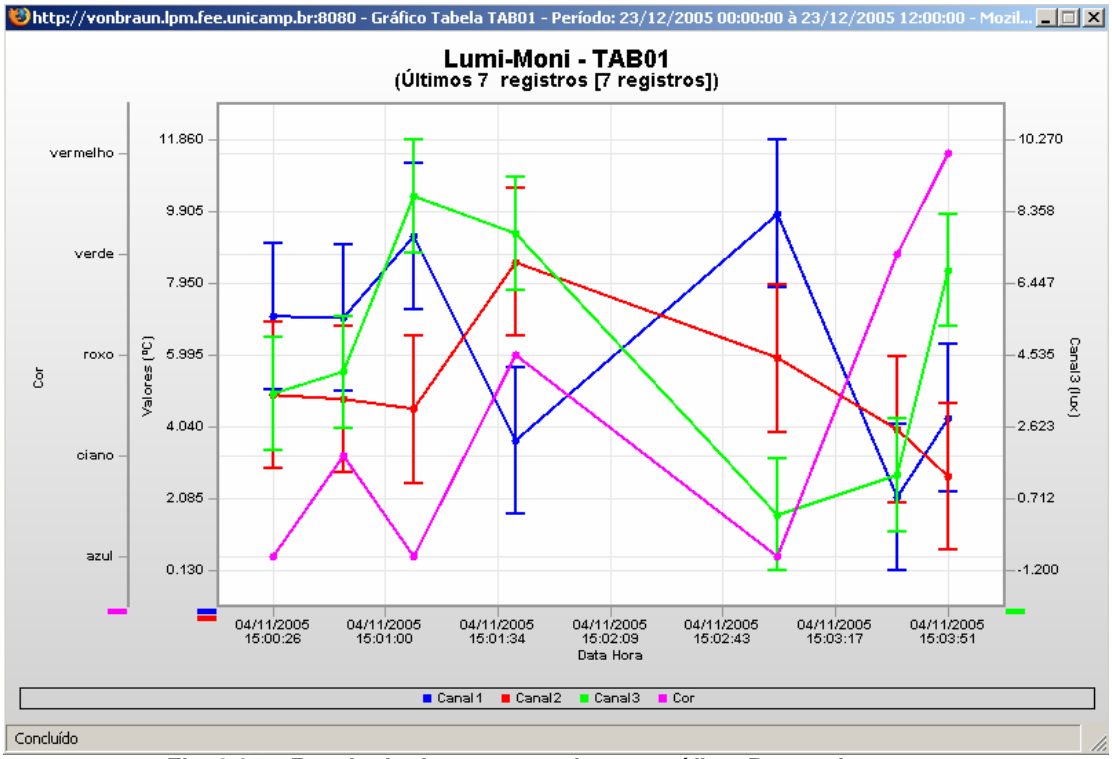


Fig. 3.35 – Resultado de uma consulta por gráfico. Barras de erros e agrupamentos de colunas nos eixos determinados pelas unidades.

As figuras 3.36 e 3.37 ilustram consultas realizadas em tabelas que contêm imagens e áudios. As imagens são exibidas nas colunas das tabelas que tenham como resultado uma imagem e os elementos do tipo áudio podem ser baixados como arquivos de áudio e o usuário visualiza a forma de onda do áudio nas colunas da tabela. A figura 3.36 mostra um áudio e o seu reverso (reprodução do fim para o início) e a figura 3.37 mostra a imagem de um personagem de desenho animado e a sua respectiva imagem preto e branco. Nessa mesma tabela da figura 3.36 observa-se que na última coluna foi realizada uma interpretação da cor da camiseta do personagem, isto é, obtenção de uma característica através de uma imagem.

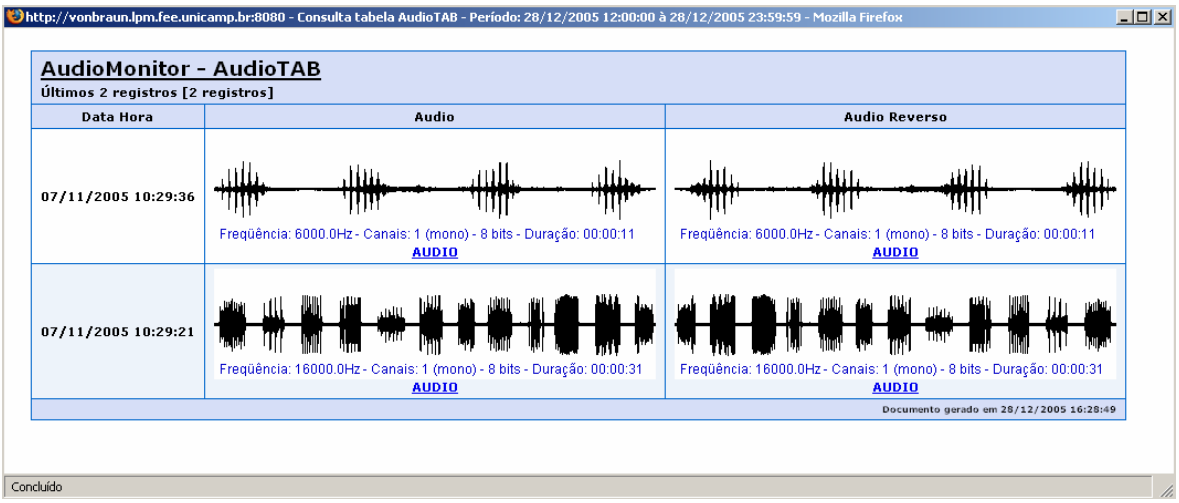


Fig. 3.36 – Resultado de uma consulta com colunas de áudio.

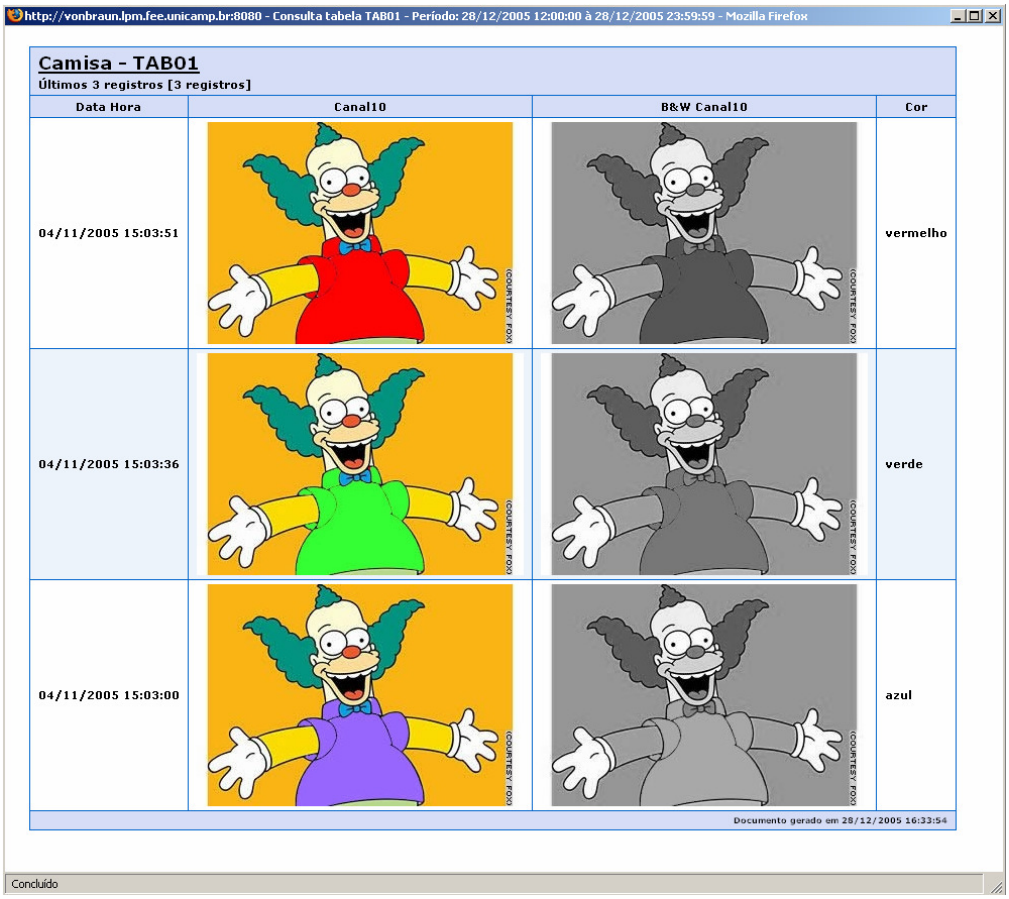


Fig. 3.37 – Resultado de uma consulta com resultado em imagens.

Resumo

Este capítulo descreveu de forma detalhada a implementação do Sistema de Monitoramento de Ambientes desenvolvido. Foram abordados todos os componentes: os monitores, o servidor e a interface do usuário.

Foram descritas as tabelas que são os mecanismos pelo qual o usuário realiza a programação (e posteriormente consulta e monitoramento) dos mecanismos de Fusão dos dados dos Sensores.

A estrutura de software foi detalhada, descrevendo todo o mecanismo de resolução das expressões que realizam a combinação dos dados dos sensores e a organização das classes programadas em JAVA.

O MER do Banco de Dados foi comentado e o processo de configuração do servidor HTTP Tomcat foi explicado.

Algumas imagens ilustraram a interface do usuário, evidenciando as possibilidades de configuração, consultas e monitoramento dos ambientes.

O capítulo 4 explicará os algoritmos de Sensores em Consenso utilizados para combinar dados de mesma natureza (a figura 3.34 mostra a execução desses algoritmos com os dados de dez sensores de temperatura) e o capítulo 5 mostrará aplicações dos procedimentos de Fusão de Sensores no Sistema de Monitoramento de Ambientes.

4

Algoritmos de Sensores em Consenso

"Fusion or confusion: knowledge or nonsense?"[1]
Peter L. Rothman e Richard V. Denton

Os algoritmos de fusão de sensores indicados na descrição dos níveis de fusão no capítulo 2 apresentam diferentes graus de complexidade e a utilização deles abrange um número quase incontável de aplicações.

Em particular, os algoritmos de fusão no nível de *pixel* e no nível de característica são objetos de estudos de alta complexidade nas áreas de processamento de imagens e inteligência artificial. A Filtragem Kalman é outro processo de combinação de dados que vem rendendo vários trabalhos acadêmicos e artigos, sendo objeto de estudo principalmente nas áreas de robótica e navegação aérea.

Cada classe de algoritmos possui um estudo dedicado e foge do propósito desse trabalho discutir e analisar todas elas.

Os algoritmos de Sensores em Consenso (ou Algoritmos de Concordância de Dados), pertencentes ao nível de fusão de sinais segundo a classificação de Ren C. Luo e Michael G. Kay, possuem uma característica especial dentro da área de fusão de

[1] "Fusão ou confusão: conhecimento ou absurdo?" – Título de um artigo publicado pelos autores Peter L. Rothman e Richard V. Denton.

sensores porque tratam especificamente da combinação de dados de mesma natureza com o objetivo de se obter um valor consensual dentro de um conjunto de medidas, melhorando, assim, a qualidade do valor que se quer monitorar no ambiente. Trata-se, portanto, de uma fusão do tipo competitiva (ou redundante).

O fato dos sensores estarem sujeitos a diversas interferências e ao desgaste promovido pelo ambiente, muitas vezes hostil, é outro fator que potencializa a utilização de mais de um sensor de mesma natureza no monitoramento de ambientes e a posterior aplicação de algoritmos de fusão competitiva como forma de eliminação dos dados discordantes, além de possibilitar a detecção dos possíveis sensores falhos.

O primeiro algoritmo estudado neste capítulo é o **Critério de Chauvenet** [TAY97], um critério estatístico para eliminação de dados discordantes do conjunto. O algoritmo de **Concordância Aproximada** [DOL85], o de **Concordância Inexata (FCA** [MAH85] **e CCA** [DOL81]**)**, o de **Fusão de Dados Contraditórios** [JAY94] e o algoritmo **Híbrido** [BRO98] são propostas que utilizam a manipulação de intervalos numéricos para a determinação do valor consensual no conjunto de medidas.

4.1 – Problema dos Generais Bizantinos

A fusão de sensores no nível de sinal para variáveis de mesma natureza pode ser realizada de diversas formas, desde a realização da média aritmética com os valores obtidos pelos sensores até a execução de algoritmos sofisticados que eliminam valores discordantes do conjunto ou consideram a exatidão (*accuracy*) dos elementos sensores na fusão dos dados.

Grande parte dos algoritmos que efetuam a fusão dos dados sensoriados deriva do Problema dos Generais Bizantinos (*Bizantine General Problem* – BGP) formulado por Lamport *et al*, em 1982 [LAM82]. Resumidamente, o BGP pode ser enunciado da seguinte forma:

> *O comandante de um exército tem algumas tropas posicionadas ao redor da cidade. Cada tropa está sob o comando de um general. O comandante dá as ordens para as tropas avançarem ou recuarem, sabendo que existem alguns generais e mensageiros que são traidores.*

Os generais discutem suas decisões através dos mensageiros. Se todos os generais seguirem as ordens do comandante, eles terão uma grande chance de sucesso. Se algumas tropas atacarem enquanto outras recuam, o fracasso é iminente.

Como os generais garantem que através da troca de mensagens entre eles, todos os generais tomarão a mesma decisão e que essa decisão coincide com a ordem dada pelo comandante?

O problema romanceado através do BGP é encontrado tipicamente em Sistemas Distribuídos. Uma mensagem é enviada para todos os nós de uma rede e deve-se garantir que todos os nós da rede recebam a mesma mensagem, sem alterações no seu conteúdo.

Para que isso ocorra, algumas questões devem ser resolvidas para garantir a confiabilidade do sistema:

- As falhas de comunicação deverão ser contornadas;
- As possíveis alterações nas mensagens deverão ser detectadas;
- Uma ação baseada no conteúdo de uma mensagem só deve ser executada por um nó uma vez que haja concordância com as mensagens recebidas por todos os outros nós da rede ou, pelo menos, que a mensagem seja considerada correta baseando-se em algum critério.

Em ambientes onde há a possibilidade de perda de mensagens ou alterações no conteúdo das mensagens enviadas através de uma rede é fundamental a elaboração de mecanismos que garantam a concordância entre os nós da rede e, assim, a confiabilidade da rede.

Dentro do contexto de rede de sensores e fusão de sensores, o Problema dos Generais Bizantinos pode ser traduzido da seguinte maneira:

Dado um sistema com N elementos sensores (ES), sendo que t elementos sensores podem estar falhos ou que haja falha na comunicação desses t elementos sensores com os outros elementos sensores da rede, busca-se garantir que os processadores não falhos presentes em todos os elementos sensores da rede concordem com os dados provenientes de cada outro elemento sensor da rede.

4.2 – Adaptação do problema BGP

O Problema dos Generais Bizantinos foi formulado dentro do contexto de uma rede de sensores cuja característica é que cada um dos N elementos sensores recebe e envia suas leituras para os outros $N-1$ elementos sensores da rede. O objetivo é que mesmo na presença de falhas na comunicação entre os nós sensores ou na existência de problemas na aquisição de dados dos sensores (funcionamento incorreto do sensor), todos os elementos sensores da rede, uma vez de posse do seu valor sensoriado e dos demais valores dos outros sensores, possam convergir para um único valor.

Dentro de um ambiente computacional onde não há comunicação entre os elementos sensores, mas eles enviam suas medições para uma central de processamento, há a possibilidade da utilização dos algoritmos propostos na resolução do BGP de forma que a central de processamento represente um nó da rede no qual não há medição própria e é o único nó onde há o processamento.

Como as medições realizadas por grande parte dos sensores é um valor real (tensão, corrente, etc.), a proposição do BGP da transmissão da mensagem com a ordem do comandante para as tropas avançarem ou recuarem (variável *booleana* 0 ou 1) deve ser substituída por um número real.

Com essas adaptações, não faz sentido pensar em convergência dos valores nos nós sensores, uma vez que não há comunicação entre os sensores. O que se procura é uma melhoria na exatidão e na precisão do sistema baseado em algoritmos de concordância dos dados sensoriados.

4.3 – Concordância de Dados

O principal objetivo dos algoritmos de Concordância de Dados dentro do contexto deste trabalho está relacionado à idéia de exatidão e precisão em um conjunto de medidas.

A **exatidão** (*accuracy*) pode ser definida como a diferença entre a medida efetuada de uma variável e o seu valor real. Em outras palavras, a exatidão representa o grau com que a variável medida realmente representa a verdade. Quanto maior a

diferença entre a medida da variável e o valor real dela, menor é a exatidão do elemento sensor.

Para determinar a exatidão de um sistema ou de um conjunto de medidas é necessário conhecer o valor verdadeiro da grandeza que se está medindo. Este valor, que é a referência das medidas, provém de um padrão.

A exatidão de um conjunto de medidas é a máxima distância encontrada das medidas do conjunto e o valor real da variável.

Em um conjunto de medidas, define-se **precisão** como a máxima distância encontrada entre duas medidas desse conjunto. Quanto menor a diferença entre as medidas, melhor é a precisão do sistema. A figura 4.1 ilustra as idéias de precisão e exatidão em um sistema de alvo.

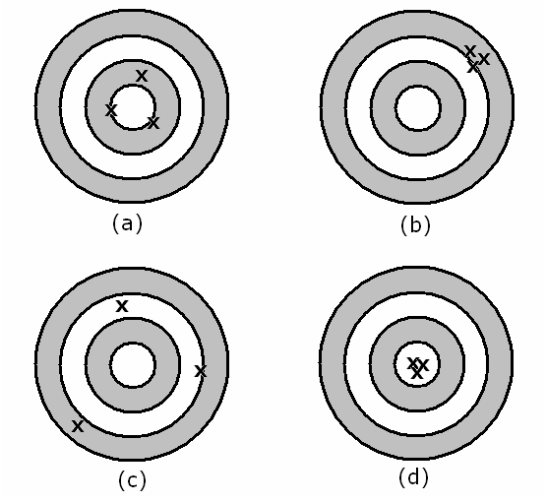


Fig. 4.1 – Comparação entre precisão e exatidão. (a) Conjunto de medidas não precisas e exatas; (b) Conjunto de medidas precisas e não exatas; (c) Conjunto de medidas não precisas e não exatas; (d) Conjunto de medias precisas e exatas.

Os algoritmos de Concordância Aproximada, Concordância Inexata e Fusão de Dados Contraditórios são variações do BGP. O algoritmo Híbrido é uma mistura das idéias propostas pelos algoritmos Concordância Inexata e a Fusão de Dados Contraditórios.

O algoritmo de descarte de dados pelo critério de Chauvenet é uma proposta estatística para o tratamento das medições realizadas pelos sensores. Os algoritmos estatísticos valem-se de um modelo que prevê a distribuição das medidas de

uma variável; se alguma das medidas não satisfizer o modelo de distribuição, essa medida é considerada “estranha” e deve ser descartada. O Critério de Chauvenet utiliza o modelo de distribuição normal como critério de seleção das medidas.

Os algoritmos de Sensores em Consenso podem ser executados com medidas de mesma natureza que são obtidas ao mesmo tempo, mas podem ser aplicados aos dados obtidos por medições temporais sucessivas de um único elemento sensor que atua sobre um evento repetitivo (como forma de aperfeiçoamento da média aritmética em medidas experimentais de laboratórios, por exemplo).

4.4 – O problema da rejeição de dados

O problema da rejeição de dados também é conhecido como recusa de dados contraditórios. A questão que envolve esse tema é “quando”, ou seja, em quais circunstâncias, um dado pode ser rejeitado dentro de um conjunto de medidas.

Ainda há muitas controvérsias sobre esse tópico [TAY97]. Alguns cientistas acreditam que um dado nunca deve ser descartado sem que haja evidências externas que a medição esteja incorreta. Às vezes uma medida dentro de um conjunto discorda de forma tão grosseira de todas as outras que fica claro que ela deve ser rejeitada. Quando um sistema não está projetado para detectar e indicar a falha de seus componentes, algum critério deve ser aplicado aos valores para que se justifique a retirada do valor anômalo do conjunto.

Uma medição legítima de certa variável desvia naturalmente, dentro de certos limites, do valor real [BUS87, LEH97, TAY97]. Entretanto, é improvável acontecer uma discrepância muito grande em um conjunto de medidas de mesma natureza realizadas por elementos sensores destinados à medição de uma mesma variável do mundo real dentro de um mesmo contexto (condições ambientais, espaciais e/ou temporais).

Contudo, a recusa de dados contraditórios sempre deve ser reportada dentro do sistema. A ocorrência de dados discrepantes indica que o sistema apresenta problemas em algum de seus componentes e, apesar do resultado dos algoritmos eliminarem as medições “suspeitas” dentro do conjunto de leituras, é fundamental a

execução da manutenção do sistema efetuando-se substituições de componentes e/ou reparo de algum subsistema. A recusa de dados serve, pois, como uma forma indicativa de falhas no sistema.

Se a maioria dos sensores estiver registrando valores incorretos e eles concordarem entre si no erro, o resultado do algoritmo será os dados incorretos, mas mesmo assim, o descarte das medidas corretas servirá como aviso de que o sistema apresenta algum problema.

A não rejeição de dados nem sempre é indicadora de que todos os elementos sensores estão funcionando corretamente. O fato dos valores adquiridos pelos elementos sensores concordarem diante algum critério indica apenas que as medições concordam entre si, mas todo o sistema pode estar corrompido (pode ter ocorrido uma falha generalizada no sistema de sensoriamento) e as medições podem não estar representando o valor real da variável sensoriada.

4.5 – Formulação do problema da Concordância de Dados

O problema que os algoritmos de Sensores em Consenso se propõem a resolver pode ser formulado da seguinte maneira:

> Dado um conjunto de medidas de uma mesma grandeza física obtidas por sensores de tecnologias e exatidões diferentes, busca-se um valor consensual, qualitativamente melhor que as medidas isoladas, de forma que esse valor represente a grandeza física medida no ambiente. Se alguma medida for descartada na computação do valor concordante, o sensor que obteve a medida deve ser identificado, pois há a possibilidade dele estar com problemas no seu funcionamento ou no canal de comunicação.

Cada algoritmo estudado resolve esse problema de uma forma diferente. Alguns levam em consideração as exatidões dos sensores, outros só consideram as medidas realizadas por eles.

4.6 – O critério de Chauvenet

Dado um conjunto M contendo N medidas $M=\{x_0, x_1, ..., x_{N-1}\}$ obtidas por diferentes elementos sensores, nas quais há a possibilidade de existir medidas

contraditórias, o critério de Chauvenet fornece um forma de decidir se os valores discordantes do conjunto devem ou não ser descartados.

O critério estatístico de Chauvenet inicia calculando a média aritmética $\bar{x}$ e o desvio padrão σ das N medidas do conjunto M (equações 4.1 e 4.2, respectivamente).

$$\bar{x} = \frac{1}{N}\sum_{i=0}^{N-1} x_i \tag{4.1}$$

$$\sigma = \sqrt{\frac{1}{N-1}\sum_{i=0}^{N-1}(x_i - \bar{x})^2} \tag{4.2}$$

Tendo a média aritmética $\bar{x}$ e o desvio padrão σ calculados, para cada valor $x_i \in M$, com $0 \leq i \leq N\text{-}1$, deve-se verificar se x_i é suspeito de discordar das demais medidas do conjunto através do cálculo da diferença entre o valor x_i e média aritmética $\bar{x}$ das medidas (equação 4.3).

$$t_i = \frac{|x_i - \bar{x}|}{\sigma} \tag{4.3}$$

O critério de Chauvenet considera que as medidas do conjunto M representam uma variável contínua sujeita a erros aleatórios, enquadrando-se no modelo da distribuição normal (ou Gaussiana). Portanto, deve-se verificar a probabilidade de se encontrar uma medida como a x_i dentro do conjunto de medidas M segundo a distribuição normal.

A equação 4.4 é a função da distribuição normal, na qual X é o valor real da medida x e σ é o desvio padrão.

$$G_{X,\sigma}(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-(x-X)^2/2\sigma^2} \tag{4.4}$$

A integral de $G_{X,\sigma}(x)$ é denominada Integral do Erro Normal. A equação 4.5 exibe essa integral com x se entre a e b.

$$Prob(a \leq x \leq b) = \int_a^b G_{X,\sigma}(x)dx \tag{4.5}$$

A integral da equação 4.5 calculada no intervalo de a=X-$t\sigma$ e b=X+$t\sigma$ fornece a probabilidade da medida x estar contida dentro do desvio padrão de t em um ou outro lado de X (equação 4.6).

$$Prob(\text{ dentro } t\sigma) = Prob(X\text{-}t\sigma \leq x \leq X\text{+}t\sigma) = \int_{X-t\sigma}^{X+t\sigma} G_{X,\sigma}(x)dx = \frac{1}{\sqrt{2\pi}}\int_{-t}^{t} e^{-z^2/2}dz \tag{4.6}$$

De modo complementar, a probabilidade de uma medida pertencer à parte exterior (fora $t\sigma$) da curva, para o mesmo intervalo é dada pela equação 4.7.

$$Prob(\text{ fora } t\sigma) = 1 - Prob(\text{ dentro } t\sigma) \tag{4.7}$$

Para o critério de Chauvenet, deve-se calcular a $Prob$(fora $t_{sus}\sigma$) que indica a probabilidade do ponto x_{sus} estar fora do intervalo $X \pm t\sigma$ (figura 4.2).

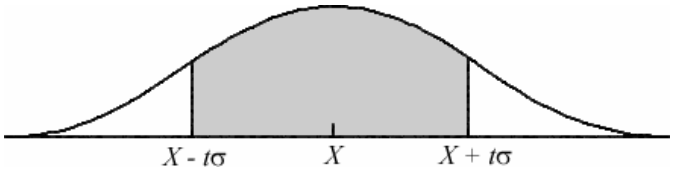


Fig. 4.2 – Curva Normal.

O passo seguinte do critério de eliminação é calcular o número de medidas esperadas que estejam fora do intervalo $x \pm \sigma$, multiplicando a probabilidade $Prob(\text{ fora } t_i\sigma)$ pelo número de elementos do conjunto M, ou seja, N=|M| (equação 4.8).

$$n = N \times Prob(\text{ fora } t_i\sigma) \tag{4.8}$$

Se $n<1/2$, de acordo com o critério de Chauvenet, x_i deve ser descartado do conjunto M.

Com a exclusão das medidas consideradas “ruins” pelo critério de Chauvenet, somente as restantes devem ser consideradas em cálculos posteriores (como média aritmética e desvio padrão, por exemplo).

Algoritmo Critério de Chauvenet

1 - Cada elemento sensor envia seu valor para a central;
2 - A central recebe os valores dos ***N*** sensores da rede e os insere no vetor ***v***;
3 - Calcula-se a média aritmética ***m*** dos valores do vetor ***v***;
4 - Calcula-se o desvio padrão ***dp*** dos valores do vetor ***v***;
5 - Cria-se o vetor **v'**;
6 - Para cada $v[i]$, $0 \leq i \leq N-1$, calcula-se o valor

$$cc = N * (1 - Prob((v[i] - m)/dp))$$

(***Prob*** é a função integral da Equação 4.6)
Se $cc \geq 0,5$, $v[i]$ é inserido no vetor **v'**;
7 - A resposta do algoritmo é o vetor **v'**, para o qual pode ser calculada a média aritmética com os seus valores.

4.7 – Algoritmo de Concordância Aproximada

O algoritmo de Concordância Aproximada é uma extensão do Problema dos Generais Bizantinos (BGP). Este algoritmo foi proposto por Danny Dolev *et al* em 1985 [DOL85].

A solução proposta pelo algoritmo é ordenar o conjunto de medidas $M_{S1}=\{x_{S1,0}, x_{S1,1}, ..., x_{S1,N-1}\}$ em cada elemento sensor S_k $(0 \leq k \leq N-1)$ e considerar a possibilidade das t maiores medidas e das t menores medidas estarem incorretas, ou seja, são descartados os valores extremos do conjunto ordenado das medidas. O valor de t é igual a um terço da quantidade de elementos sensores da rede $(N/3)$.

Para que o algoritmo consiga descartar todos os valores com erro e haja uma concordância adequada dos valores, o número de sensores falhos ou comunicações falhas na rede não deve ser maior que $(N-1)/3$, ou seja, $t \leq (N-1)/3$.

A figura 4.3 ilustra de forma gráfica o Algoritmo de Concordância Aproximada. Num conjunto de seis valores, apenas dois são considerados aceitos.

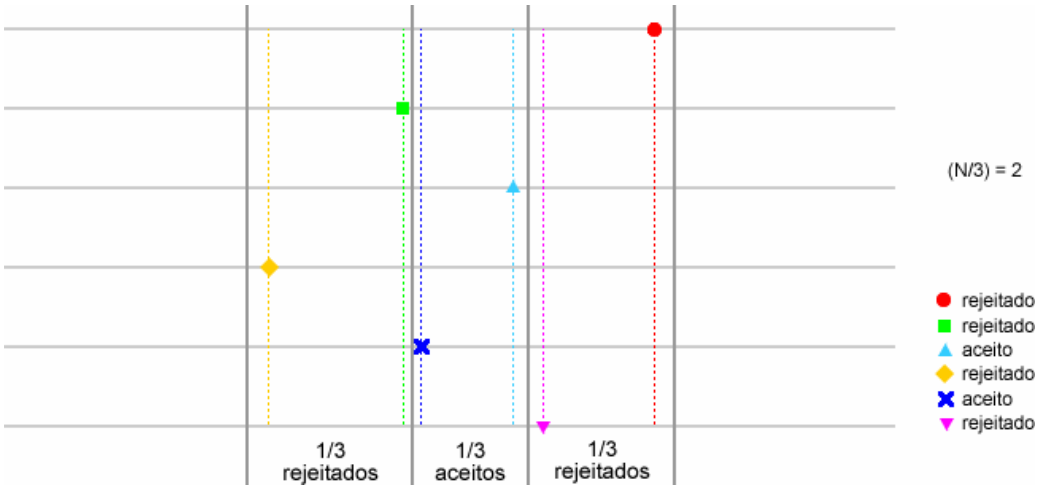


Fig. 4.3 – Visualização gráfica do algoritmo de Concordância Aproximada.

O algoritmo de Concordância Aproximada melhora a exatidão da média das medidas, uma vez que ele elimina os valores mais discrepantes (os mais discrepantes para mais e os mais discrepantes para menos) do conjunto.

Algoritmo de Concordância Aproximada

1 - Cada elemento sensor envia seu valor para a central;
2 - A central recebe os valores dos ***N*** sensores da rede;
3 - Os valores recebidos são ordenados e inseridos em um vetor ***v***;
4 - Calcula-se o número **t** = ⌊**N**/3⌋ de elementos a ser descartado em cada extremo do vetor;
5 - Cria-se um novo vetor **v'** de tamanho (**N**-2**t**), contendo os valores **v**[**i**], para **t** ≤ i ≤ **N**-1-**t**;
6 - A resposta do algoritmo é o vetor **v'**, para o qual pode ser calculada a média aritmética dos seus valores.

4.8 – Algoritmo de Concordância Inexata

O algoritmo de Concordância Inexata proposto por Stephen R. Mahaney e Fred B. Schneider em 1985 [MAH85] também é uma variante do problema dos Generais Bizantinos. Este algoritmo também é conhecido por Algoritmo de Convergência Rápida (*Fast Convergente Algorithm – FCA*). Um segundo algoritmo de Concordância Inexata foi proposto por Danny Dolev [DOL81] e é conhecido como Algoritmo de Convergência Cruzada (*Crusaders Convergence Algorithm – CCA*).

Dado um conjunto de medidas $M=\{x_0, x_1, ..., x_{N-1}\}$ e um segundo conjunto contendo a exatidão dos elementos sensores $A=\{a_0, a_1, ..., a_{N-1}\}$ que originaram as medidas do primeiro de tal forma que dada uma medida $x_k \in M$ a sua exatidão é $a_k \in A$, para $0 \leq k \leq N-1$, um valor x_k é considerado aceitável dentro do conjunto, isto é, ele não é "suspeito" de discordar dos demais valores, se x_k estiver contido dentro de $N-t$ intervalos de exatidão dos outros elementos sensores do conjunto, onde t é o número de elementos sensores falhos na rede. Ou seja, se x_k $(0 \leq k \leq N-1)$, pertencer a pelo menos $N-t$ intervalos $[x_w-a_w, x_w+a_w]$ para $0 \leq w \leq N-1$ e $w \neq k$, ele é considerado aceito dentro do conjunto. Segundo o algoritmo FCA, os elementos discordantes do conjunto devem ser substituídos pela média aritmética dos valores do conjunto de medidas M; o algoritmo CCA propõe que esses elementos devem ser eliminados do conjunto M. Dolev denomina essa eliminação de elementos discordantes como processo de "purificação" do conjunto.

A figura 4.4 mostra a visualização gráfica do algoritmo de Concordância Inexata. Apenas os pontos que tiveram 4 ou mais interseções com os intervalos dos outros pontos são considerados aceitos.

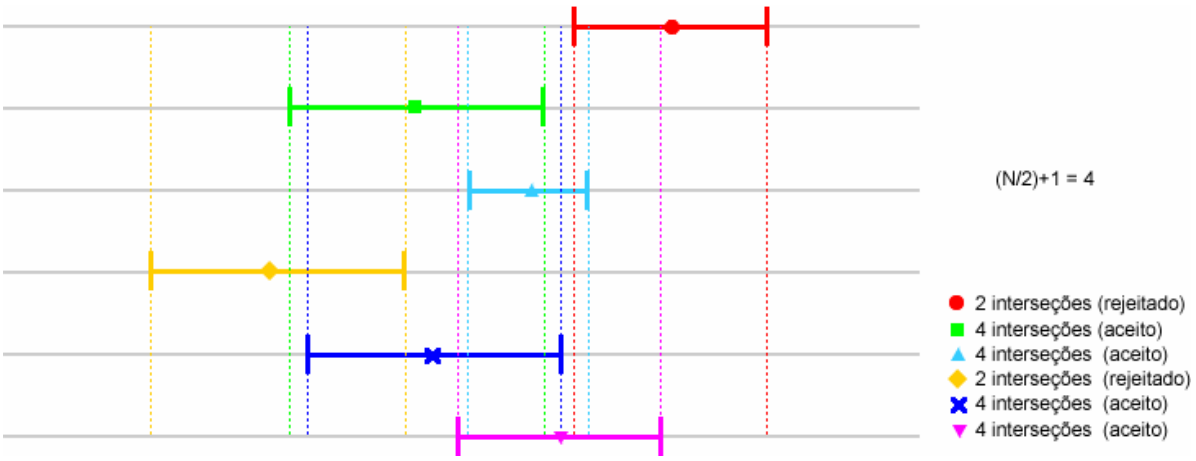


Fig. 4.4 – Visualização gráfica do algoritmo de Concordância Inexata.

Como não se sabe de antemão a quantidade de elementos sensores falhos na rede ou quantas falhas de comunicação existem entre os sensores, o valor de t utilizado no cálculo do número de intervalos que a medida de um sensor deve pertencer para a medida ser aceita $(N-t)$ é igual a $(N/2)+1$.

Algoritmo de Concordância Inexata FCA

1 - Cada elemento sensor envia seu valor para a central;
2 - A central recebe os valores dos ***N*** sensores da rede;
3 - Os valores recebidos dos sensores são inseridos no vetor ***v***;
4 - Os valores das precisões dos sensores são inseridos no vetor ***a***, de forma que para cada elemento sensor S_i ($0 \leq i \leq N-1$), o valor da sua medida seja ***v*[*i*]**, o valor da sua precisão seja ***a*[*i*]**;
5 - Cria-se um novo vetor ***v'*** de tamanho ***N***;
7 - Calcula-se a média dos valores do vetor ***v*** e atribui-se o resultado na variável ***media***;
6 - Para cada ***v*[*i*]**, $0 \leq i \leq N-1$:
  s = 0;
  Para cada ***v*[*j*]**, $0 \leq j \leq N-1$ e $j \neq i$:
    Se (***v*[*i*]** $\leq$ ***v*[*j*]+*a*[*j*]**) e (***v*[*i*]** $\geq$ ***v*[*j*]-*a*[*j*]**)
      então **s** = **s** + 1;
  Se **s** $\geq$ (**N**/2)+1
    então ***v'*[*i*]** = ***v[i]***;
    senão ***v'*[*i*]** = ***media***;
7 - A resposta do algoritmo é o vetor **v'**, para o qual pode-se calcular a média aritmética dos seus valores.

Algoritmo de Concordância Inexata CCA

1 - Cada elemento sensor envia seu valor para a central;
2 - A central recebe os valores dos ***N*** sensores da rede;
3 - Os valores recebidos dos sensores são inseridos no vetor ***v***;
4 - Os valores das precisões dos sensores são inseridos no vetor ***a***, de forma que para cada elemento sensor S_i ($0 \leq i \leq N-1$), o valor da sua medida seja ***v*[*i*]**, o valor da sua precisão seja ***a*[*i*]**;
5 - Cria-se um novo vetor ***v'*** de tamanho ***N***;
6 - ***j*** = 0;
  Para cada ***v*[*i*]**, $0 \leq i \leq N-1$:
    s = 0;
    Para cada ***v*[*j*]**, $0 \leq j \leq N-1$ e $j \neq i$:
      Se (***v*[*i*]** $\leq$ ***v*[*j*]+*a*[*j*]**) e (***v*[*i*]** $\geq$ ***v*[*j*]-*a*[*j*]**)
        então **s** = **s** + 1;
  Se **s** $\geq$ (**N**/2)+1
    então ***v'*[*j*]** = ***v[i]***;
      j = ***j*** + 1;
7 - A resposta do algoritmo é o vetor **v'**, para o qual pode-se calcular a média aritmética dos seus valores.

Assim como o algoritmo de Concordância Aproximada, o algoritmo de Concordância Inexata também tem como objetivo aumentar a precisão no valor final da medida, mas neste último, a exatidão pode ser prejudicada. Isso ocorre porque o descarte de valores (no caso do algoritmo CCA) ou substituição dos valores (no caso do algoritmo FCA) é realizado considerando a relação das medidas e seus intervalos de

exatidão entre si, não havendo garantia da aceitação dos valores mais exatos no conjunto final.

4.9 – Algoritmo de Fusão de Dados Contraditórios

O algoritmo de Fusão de Dados Contraditórios aborda a questão da determinação de uma medida que representa um conjunto de medidas (onde podem existir medidas falhas e discordantes) de uma maneira diferenciada dos demais algoritmos. A busca deste algoritmo, também denominado Algoritmo da Média Tolerante a Falhas (*Fault-tolerant Averaging Algorithm*), não é um único valor resultante da combinação dos valores do conjunto de medidas e sim um intervalo de valores que representa o conjunto de medidas.

O problema, então, pode ser enunciado da seguinte forma: dado um conjunto de N elementos sensores, todos com exatidão limitada e podendo existir t elementos sensores defeituosos (ou existir em falhas de comunicação na rede para esses nós sensores), qual é a menor faixa de valores que contém o valor real da variável sensoriada pelos elementos sensores?

Uma proposta de algoritmo para solução deste problema foi desenvolvida por Keith Marzullo [MAR90], que levou em consideração a existência de comunicação entre os sensores da rede. D. N. Jayasimba [JAY94] fez a abordagem considerando que os elementos sensores da rede se comunicam exclusivamente com a central de processamento. Como nos demais algoritmos apresentados, não há necessidade de adaptações no Algoritmo da Média Tolerante a Falhas para utilizá-lo em redes nas quais os sensores não se comunicam entre si.

A idéia do Algoritmo de Fusão de Dados Contraditórios está baseada na interseção de intervalos. Dado um conjunto de medidas $M=\{x_0, x_1, ..., x_{N-1}\}$ e um conjunto $A=\{a_0, a_1, ..., a_{N-1}\}$ que contém a exatidão dos elementos sensores que originaram as medidas de tal forma que para cada medida $x_k \in M$ a sua exatidão é $a_k \in A$, com $0 \leq k \leq N-1$, criam-se os conjuntos L e U contendo, respectivamente, os menores e maiores valores para cada medida dentro dos seus intervalos de exatidão, ou seja: $L = \{l_0=x_0-a_0, ..., l_{N-1}=x_{N-1}-a_{N-1}\}$ e $U = \{u_0=x_0+a_0, ..., u_{N-1}=x_{N-1}+a_{N-1}\}$. Os conjuntos L e U são unidos

formando o conjunto $H=\{ h_0=l_0, ..., h_{N-1}=l_{N-1}, h_N=u_0, ..., h_{2N-1}=u_{N-1}\}$, que deve ser ordenado de forma crescente e eliminando os elementos repetidos.

O passo seguinte do algoritmo é verificar para cada intervalo $[h_k,h_{k+1}]$, com $0 \leq k \leq 2N-2$, quantas interseções existem deste intervalo com os intervalos $[x_w- a_w, x_w+a_w]$, com $0 \leq w \leq N-1$. Se o número de interseções for maior que $(N/2)+1$, o valor máximo e mínimo do intervalo $[h_k,h_{k+1}]$ devem ser inseridos no conjunto R.

O último passo do algoritmo é ordenar os elementos do conjunto R. A resposta do algoritmo é o intervalo determinado pelo menor e pelo maior elemento do conjunto R, ou seja, o valor real da variável que foi sensoriada pelos N elementos sensores encontra-se nesse intervalo.

A figura 4.5 ilustra de forma gráfica a Fusão de Dados Contraditórios. O número de interseções é o número de intervalos criados pelas exatidões das medidas que o intervalo de cada medida intercepta.

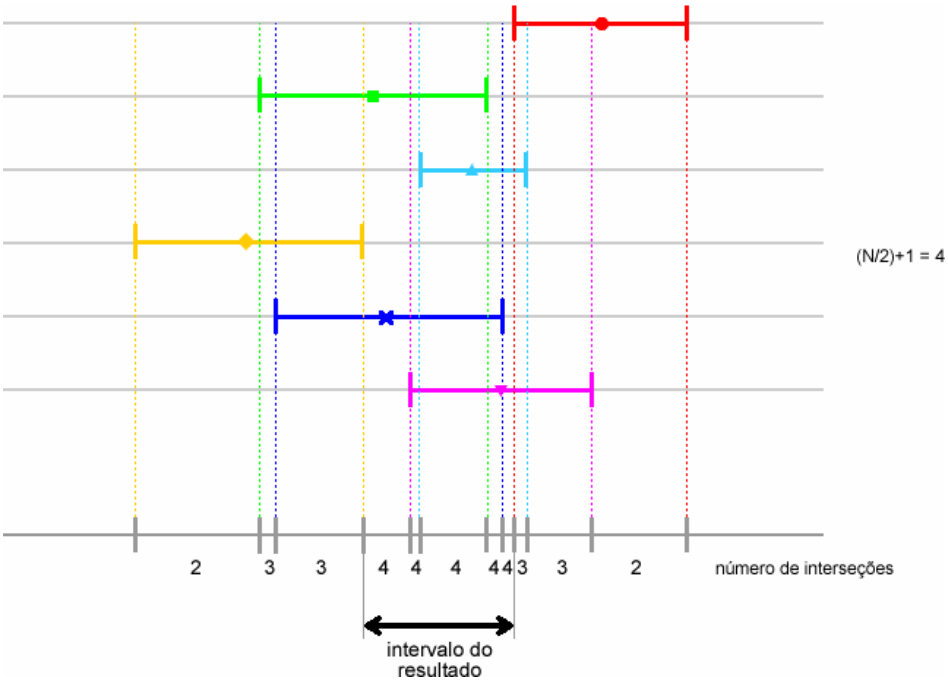


Fig. 4.5 – Visualização gráfica do algoritmo de Fusão de Dados Contraditórios.

Algoritmo de Fusão de Dados Contraditórios

1 - Cada elemento sensor envia seu valor para a central;
2 - A central recebe os valores dos ***N*** sensores da rede e os insere no vetor ***v***;
3 - Os valores das exatidões dos sensores são inseridos no vetor ***a***, de forma que para cada elemento sensor S_i ($0 \leq i \leq N-1$), o valor da sua medida seja ***v*[*i*]** e o valor da sua exatidão seja ***a*[*i*]**;
4 - Cria-se o vetor ***l*** contendo os mínimos valores de cada medida, ou seja, ***l*[*i*]=*v*[*i*]-*a*[*i*]**;
5 - Cria-se o vetor ***u*** contendo os máximos valores de cada medida, ou seja, ***u*[*i*]=*v*[*i*]+*a*[*i*]**;
6 - Cria-se o vetor ***h*** concatenando os vetores ***l*** e ***u***, ou seja, $\boldsymbol{h} \leftarrow \boldsymbol{l} \cup \boldsymbol{u}$;
7 - Ordena-se o vetor ***h***;
8 - Eliminam-se os elementos repetidos do vetor ***h***;
9 - Criam-se as variáveis ***retMin*** e ***retMax*** com valor inicial nulo (***null***);
10 - Para cada ***h*[*i*]**, $0 \leq \boldsymbol{i} \leq \boldsymbol{N}-2$:
  s = 0;
  Para $0 \leq \boldsymbol{j} \leq \boldsymbol{N}-1$:
    Se **!**((***h*[*i*]** ≥ ***l*[*j*]**) ou (***u*[*j*]** ≥ ***h*[*i*+*1*]**))
      então **s** = **s** + 1;
  Se **s** ≥ (**N**/2)+1, então:
    retMin = min(**retMin**, ***h*[*i*]**);
    retMax = max(**retMax**, ***h*[*i*+1]**);
11 - A resposta do algoritmo são os valores ***retMin*** e ***retMax***, que contêm os valores mínimo e máximo, respectivamente, do intervalo no qual o valor real da medida está contido. Para se obter um único valor, pode-se calcular a média aritmética dos limites do intervalo: (***retMin***+***retMax***)/2 .

4.10 – Algoritmo Híbrido

O algoritmo de Fusão de Dados Contraditórios tem a propriedade de aumentar a exatidão em detrimento da precisão do valor final da medida, se comparado aos demais algoritmos de Sensores em Consenso. O algoritmo de Concordância Inexata, pelo contrário, aumenta a precisão em detrimento da exatidão. R. R. Brooks e S. S. Iyengar [BRO98] propuseram um algoritmo Híbrido que une a melhora na exatidão efetuada pelo algoritmo de Fusão de Dados Contraditórios com a melhora na precisão proporcionada pelo algoritmo FCA.

A utilização do algoritmo CCA em detrimento do algoritmo FCA pode ser realizada sem problemas. A precisão conseguida será melhorada com a utilização do FCA.

Os passos do algoritmo Híbrido são bem semelhantes aos do algoritmo de Fusão de Dados Contraditórios. A diferença está na fase final que determina um único

valor e não um intervalo de valores, como é o caso do algoritmo de Fusão de Dados Contraditórios. Esse valor único é determinado através da média ponderada que é baseada no número de ocorrências de sobreposição de intervalos.

Dado um conjunto de medidas $M=\{x_0, x_1, ..., x_{N-1}\}$ e um conjunto $A=\{a_0, a_1, ..., a_{N-1}\}$ que contém a exatidão dos elementos sensores que originaram as medidas de tal forma que para cada medida $x_k \in M$ a sua exatidão é $a_k \in A$, com $0 \leq k \leq N-1$, criam-se os conjuntos L e U contendo, respectivamente, os menores e maiores valores para cada medida dentro dos seus intervalos de exatidão, ou seja: $L = \{l_0=x_0-a_0, ..., l_{N-1}=x_{N-1}-a_{N-1}\}$ e $U = \{u_0=x_0+a_0, ..., u_{N-1}=x_{N-1}+a_{N-1}\}$. Os conjuntos L e U devem unidos formando o conjunto $H=\{ h_0=l_0, ..., h_{N-1}=l_{N-1}, h_N=u_0, ..., h_{2N-1}=u_{N-1}\}$, que deve ser ordenado de forma crescente e eliminando os elementos repetidos.

O passo seguinte do algoritmo é verificar para cada intervalo $[h_k,h_{k+1}]$, com $0 \leq k \leq 2N-2$, quantas interseções existem deste intervalo com os intervalos $[x_w- a_w, x_w+a_w]$, com $0 \leq w \leq N-1$. Se o número de interseções ($q_{w,k}$) encontrado for maior que $(N/2)+1$ (da mesma forma que o algoritmo de Fusão de Dados Contraditórios), o valor $q_{w,k}$ e os extremos do intervalo $[x_w- a_w, x_w+a_w]$ deverão ser guardados para serem utilizados no cálculo da média ponderada; caso contrário, esses valores devem ser ignorados.

O valor final do algoritmo (v) será dado pela média ponderada dos valores da média aritmética dos extremos dos intervalos $[x_w- a_w, x_w+a_w]$, que passaram pelo teste da interseção dos intervalos, ponderados através dos pesos $q_{w,k}$, conforme a equação 4.9.

$$v = \frac{\sum q_{w,k}\left(\frac{(x_w + a_w)+(x_w - a_w)}{2}\right)}{\sum q_{w,k}} \tag{4.9}$$

Como o algoritmo Híbrido é derivado do Algoritmo de Fusão de Dados Contraditórios, o valor retornado por aquele necessariamente pertence ao intervalo retornado por este. Portanto, o valor resultante do algoritmo Híbrido pode ser

considerado um refinamento do procedimento de se calcular a média aritmética com os extremos do intervalo retornado pelo algoritmo de Fusão de Dados Contraditórios.

Algoritmo Híbrido

1 - Cada elemento sensor envia seu valor para a central;
2 - A central recebe os valores dos ***N*** sensores da rede e os insere no vetor ***v***;
3 - Os valores das exatidões dos sensores são inseridos no vetor ***a***, de forma que para cada elemento sensor S_i ($0 \leq i \leq N-1$), o valor da sua medida seja ***v*[*i*]** e o valor da sua exatidão seja ***a*[*i*]**;
4 - Cria-se o vetor ***l*** contendo os mínimos valores de cada medida, ou seja, ***l*[*i*]=*v*[*i*]-*a*[*i*]**;
5 - Cria-se o vetor ***u*** contendo os máximos valores de cada medida, ou seja, ***u*[*i*]=*v*[*i*]+*a*[*i*]**;
6 - Cria-se o vetor ***h*** concatenando os vetores ***l*** e ***u***, ou seja, ***h*** $\leftarrow$ ***l*** $\cup$ ***u***;
7 - Ordena-se o vetor ***h***;
8 - Eliminam-se os elementos repetidos do vetor ***h***;
9 - Cria-se a variável ***ret*** que conterá a resposta do algoritmo e a variável ***w*** que conterá a somatória dos pesos, ou seja, a somatória do número de interseções dos intervalos de valores;
10 - Para cada ***h*[*i*]**, $0 \leq i \leq N-2$:
  s = 0;
  Para $0 \leq j \leq N-1$:
    Se **!**((***h*[*i*]** $\geq$ ***l*[*j*]**) ou (***u*[*j*]** $\geq$ ***h*[*i*+*1*]**))
      então **s** = **s** + 1;
  Se **s** $\geq$ (**N**/2)+1, então:
    ret = ***ret*** + (***s*** * ((***h***[***i***+1] + ***h***[***i***]) / 2));
    w = ***w*** + ***s***;
  ret = (***ret*** / ***w***);
11 - A resposta do algoritmo é o valor ***ret***.

4.11 – Processamentos adicionais

Como cada elemento sensor é considerado uma entidade abstrata para o sistema, é possível impor restrições para os dados sensoriados na medida que os elementos sensores são limitados com relação ao intervalo de valores que eles conseguem registrar suas medições de forma satisfatória.

Algumas medidas podem ser descartadas pelo simples fato de estarem fora da faixa operacional do elemento sensor. Esse descarte (ou filtragem) deve ser efetuado antes dos valores sensoriados serem processados pelos algoritmos.

A central de processamento deve possuir, portanto, uma descrição dos elementos sensores que fornecerão medidas para as entradas dos algoritmos, constando a exatidão e o intervalo de operação de cada sensor.

Dado um conjunto de medidas $M=\{x_0, x_1, ..., x_{N-1}\}$ e os conjuntos $L=\{l_0, l_1, ..., l_{N-1}\}$ e $U=\{u_0, u_1, ..., u_{N-1}\}$ que contém, respectivamente, os valores mínimos e máximos da faixa de operação de cada sensor, sendo que para cada medida $x_k \in M$ o seu valor mínimo é $l_k \in L$ e o seu valor máximo é $u_k \in U$, com $0 \leq k \leq N-1$, o valor x_k deve ser eliminado do conjunto M se $x_k < l_k$ ou se $x_k > u_k$.

Algoritmo de eliminação por intervalo de operação dos sensores

1 - A central recebe os valores dos ***N*** sensores da rede e os insere no vetor ***v***;
2 - A central obtém os valores de máximo de mínimo da faixa de operação de cada sensor e os coloca, respectivamente nos vetores ***l*** e ***u***;
4 - Cria-se o vetor **v'**;
5 - s = 0;
Para 0 ≤ ***i*** ≤ ***N***-1:
Se **!**((***v***[***i***] < ***l***[***i***]) ou (***v***[***i***] > ***u***[***i***]))
então ***v'***[***s***] = ***v***[***i***];
s = ***s*** + 1;
6 - A resposta do algoritmo é o vetor ***v'***.

Um outro processamento que pode ser utilizado para a melhoria do resultado final é a aplicação da média ponderada nas respostas dos algoritmos que retornam vetores, ponderando os valores com o inverso da exatidão dos elementos sensores ao invés da utilização da média aritmética. Essa prática daria um peso maior aos sensores mais exatos e peso menor a sensores menos exatos.

Dado um conjunto de medidas $M=\{x_0, x_1, ..., x_{N-1}\}$ e um conjunto $A=\{a_0, a_1, ..., a_{N-1}\}$ que contém a exatidão dos elementos sensores que originaram as medidas de tal forma que para cada medida $x_k \in M$ a sua exatidão é $a_k \in A$, com $0 \leq k \leq N-1$, o valor final (v) resultante da média ponderada dos N valores com os pesos determinados pelo inversos dos valores do conjunto A está representado na equação 4.10.

$$v = \frac{\sum_{i=0}^{N-1} \frac{1}{a_k} x_k}{\sum_{i=0}^{N-1} \frac{1}{a_k}} \quad (4.10)$$

Algoritmo da média ponderada pelo inverso da exatidão dos sensores

1 - Recebe-se os ***N*** valores das medidas no vetor ***v***;
2 - Recebe-se os ***N*** valores das exatidões dos sensores que originaram as medidas do vetor ***v*** no vetor ***a***;
3 - Cria-se a variável ***ret*** que conterá a resposta do algoritmo;
4 - ***s*** = 0;
Para 0 ≤ ***i*** ≤ ***N***-1:
ret = ***ret*** + (***v*[*i*]** * (1/***a*[*i*]**));
s = s + (1/***a*[*i*]**);
ret = (***ret*** / ***s***);
5 - A resposta do algoritmo é o valor de ***ret***.

A utilização do algoritmo de média ponderada pelo inverso da exatidão do sensor não deve ser realizada utilizando o conjunto de valores brutos obtidos pelos sensores, pois um sensor com boa exatidão pode estar realizando medidas falhas (discordante das demais) e nesse caso, o seu peso será o maior, inviabilizando o resultado.

4.12 – Comparação dos Algoritmos de Sensores em Consenso

Os algoritmos de Sensores em Consenso podem ser comparados quanto ao tipo de processamento, a consideração ou não da exatidão do elemento sensor e o tipo de resposta. A tabela 4.1 faz essa comparação.

Todos algoritmos têm como entrada um vetor de tamanho N contendo as medições de cada sensor em cada posição do vetor. Alguns algoritmos também têm como entrada a exatidão dos elementos sensores.

O processamento dos dados pode ser realizado através do descarte (exclusão) de alguns valores do vetor seguindo certos critérios ou através da análise dos valores e seus intervalos, determinados pela exatidão de cada elemento sensor.

Tabela 4.1 – Comparação das características dos algoritmos de Sensores em Consenso

Algoritmo	***Critério de Chauvenet* [TAY97]**	***Concordância Aproximada* [DOL85]**	***Concordância Inexata***		***Fusão de Dados Contraditórios* [JAY94]**	***Híbrido* [BRO98]**
			***FCA* [MAH85]**	***CCA* [DOL81]**		
Tipo de Processamento	descarte de dados	descarte de dados	análise de intervalos e substituição de dados	análise de intervalos e descarte de dados	análise de intervalos	análise de intervalos
Considera a exatidão do Elemento Sensor	Não	Não	Sim		Sim	Sim
Tipo de Resposta	vetor numérico	vetor numérico	vetor numérico		intervalo numérico	valor numérico

O resultado dos algoritmos pode ser um vetor contendo os valores considerados “aproveitáveis” do conjunto de medidas da entrada, um intervalo de valores ou apenas um único valor que corresponde à fusão dos dados. No caso do resultado ser um conjunto ou um intervalo, pode-se obter um único valor numérico efetuando-se a média aritmética com os valores do vetor resultante; ou a média aritmética dos valores extremos do intervalo no caso da resposta ser um intervalo de valores.

A escolha de um dos algoritmos depende do objetivo da aplicação. O Critério de Chauvenet é a melhor opção quando o objetivo é aproximar o valor obtido pelo algoritmo com a média aritmética em ambientes que contenham sensores falhos (a não ser quando o valor registrado pelo sensor falho coincidentemente fica próximo aos demais valores).

Apesar de todos algoritmos eliminarem os valores discordantes, os algoritmos de Fusão de Dados Contraditórios e o Híbrido são os que proporcionam uma maior concordância entre os dados, tomando como parâmetros a precisão e a exatidão. Enquanto o algoritmo de Fusão de Dados Contraditórios aumenta a precisão em detrimento da exatidão, o algoritmo Híbrido realiza o oposto (desconsiderando o seu comportamento anômalo que é discutido no capítulo 5), ou seja, aumenta a exatidão em detrimento da precisão.

Os algoritmos de Concordância Aproximada e os de Concordância Inexata (FCA e CCA) são os que menos necessitam recursos computacionais, sendo indicados para sistemas com pouca memória ou baixa capacidade de processamento.

Resumo

O capítulo 4 descreveu os principais algoritmos de Sensores em Consenso, classe de algoritmos pertencentes à fusão de sensores no nível de sinal que tem como objetivo calcular uma medida consensual a partir de um conjunto de valores obtidos pelos sensores.

Foi discutida a problemática da rejeição dos dados em um conjunto de valores no qual há a possibilidade de existirem medidas discordantes e explicada a diferença entre exatidão e precisão em uma medida.

Algumas particularidades de cada algoritmo foram destacadas no final do capítulo 4 com o intuito de auxiliar na escolha de um dos algoritmos de Sensores em Consenso, conforme o objetivo que se quer alcançar com a fusão.

Na primeira aplicação experimental do capítulo 5 serão aplicados os algoritmos de Sensores em Consensos à sensores de temperatura e será realizada uma análise dos resultados e características particulares de cada algoritmo.

5

Aplicações Experimentais

"É lento ensinar por teorias, mas breve e eficaz fazê-lo pelo exemplo".
Sêneca

A aplicação do Sistema de Monitoramento de Ambientes utilizando técnicas de Fusão de Sensores é praticamente ilimitada. Podem-se criar tantos monitores e tabelas quantos forem necessários e efetuar diferentes tipos de fusão nos dados dos sensores.

Com o intuito de testar o Sistema e alguns algoritmos de Fusão de Sensores implementados, foram realizados dois experimentos.

No primeiro experimento utilizaram-se dez sensores de temperatura colocados em uma câmara térmica com o propósito de testar os algoritmos de Sensores em Consenso na obtenção de uma temperatura que representasse a temperatura interna da câmara.

O segundo experimento foi realizado utilizando sensores de naturezas diferentes (temperatura, umidade, luminosidade e uma câmera de vídeo) no monitoramento de uma planta. O objetivo desse experimento é aplicar algoritmos de

fusão de sensores que fornecem dados de naturezas distintas como possibilidade de estudo da influência das variáveis ambientais no desenvolvimento da planta.

5.1 – Aplicação dos Algoritmos de Sensores em Consenso

Para se determinar as condições de um ambiente, a utilização de apenas um sensor para cada variável não é suficiente. Um ponto de uma sala, por exemplo, pode estar mais quente que outro e a posição do sensor de temperatura teria grande influência. Além disso, se houver falha no único sensor empregado, a leitura da variável é prejudicada.

A distribuição de diversos sensores de mesma natureza por um ambiente é uma forma adotada como meio de se obter uma medida representativa da variável sensoriada e eliminar possíveis valores discordantes obtidos pelos sensores falhos.

Nesse experimento, foram utilizados dez sensores de temperatura LM35D (ver Apêndice E) com o objetivo de obter uma temperatura consensual dentro de uma câmara climática. Buscou-se, também, através da introdução de falhas em um sensor, não comprometer a medida final.

O experimento foi realizado variando-se a temperatura da câmara climática de 10 a 60°C, estabilizando-a durante 20 minutos em intervalos de 5°C. Os dados dos sensores foram "lidos" através de uma unidade de chaveamento (*Switch Control Unit* – HP3488A) associada a um multímetro (HP3458A). A saída do multímetro foi conectada através da interface GPIB com o programa LabVIEW[1] executado em um microcomputador conectado à Internet. O programa acumulava os dados dos dez canais a cada 5 segundos e enviava o documento XML com os valores para o servidor do Sistema de Monitoramento de Ambientes. A figura 5.1 ilustra o arranjo experimental utilizado.

O sensor LM35D utilizado possui uma exatidão de ±2°C, de acordo com as especificações do fabricante. Essa foi a exatidão inserida no sistema para utilização nos algoritmos de Sensores em Consenso. Uma outra característica do LM35D é que a tensão medida no terminal de leitura multiplicada pelo fator 10 fornece a temperatura

[1] http://www.ni.com/labview

medida em ºC. Há, portanto, uma transformação na unidade do valor recebido no servidor de volt para ºC.

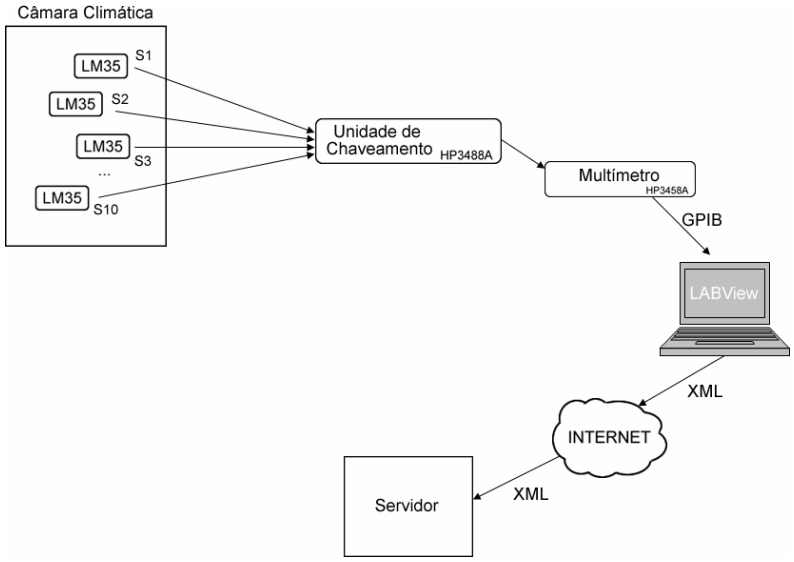


Fig. 5.1 – Montagem experimental utilizada para a aquisição das temperaturas de 10 sensores dentro de uma câmara climática.

Como cada algoritmo de Sensores em Consenso explora um tipo de combinação dos dados, buscou-se, através desse experimento, verificar as particularidades de cada algoritmo realizando duas baterias de medidas: uma com os dez sensores funcionando corretamente e uma com um sensor (S5) apresentando falhas. Os gráficos das figuras 5.2 e 5.3 mostram as duas baterias de medidas realizadas.

A falha no sensor S5 foi introduzida experimentalmente deixando o dispositivo sensor (LM35D) mal conectado à placa na qual os sensores estavam fixados (figura 5.4). Dessa forma, a vibração da câmara térmica auxiliada pela conexão incorreta do sensor permitiu que as leituras do sensor de temperatura S5 discordassem de forma grosseira das demais temperaturas registradas pelos outros sensores.

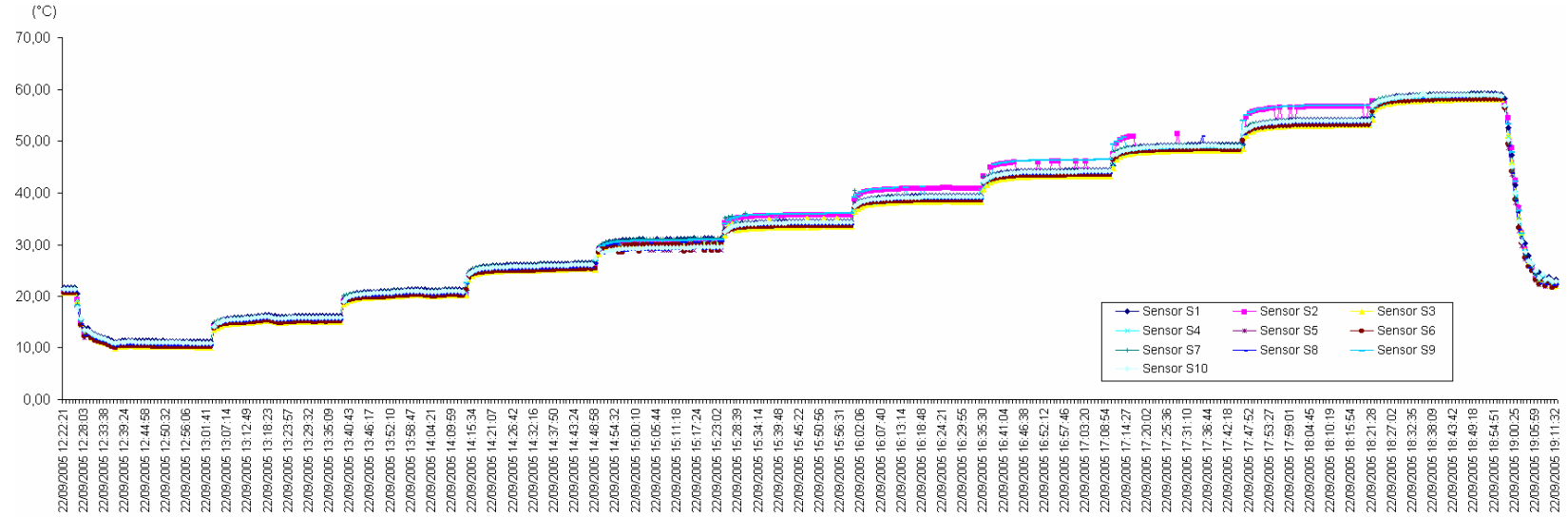


Fig. 5.2 – Gráfico das temperaturas registradas por dez sensores ao longo do tempo.

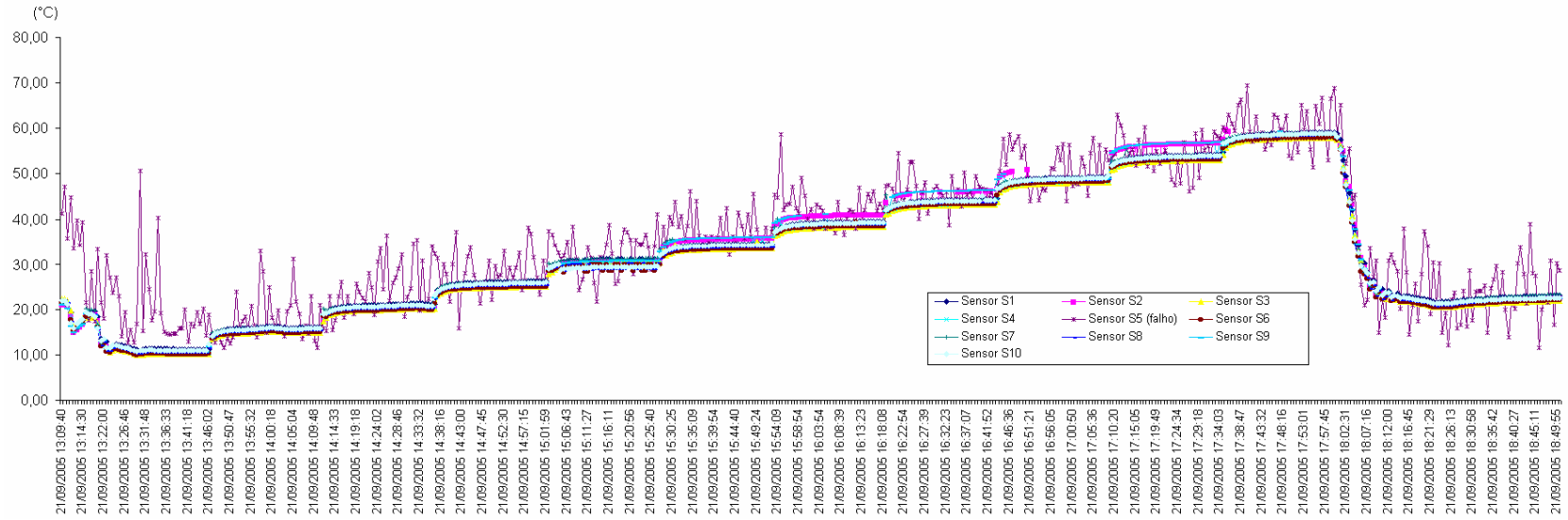


Fig. 5.3 – Gráfico das temperaturas registradas por dez sensores ao longo do tempo. Sensor S5 apresentando falhas.

Fig. 5.4 – Placa contendo os dez sensores de temperatura dentro da câmara climática.

Foram extraídas dos gráficos das figuras 5.2 e 5.3 algumas medidas pertencentes aos patamares formados durante os 20 minutos após a estabilização da temperatura na câmara climática. A tabela 5.1 mostra nove leituras seqüenciais obtidas do gráfico da figura 5.2 durante a estabilização da temperatura na câmara perto de 20°C e a tabela 5.3 mostra doze leituras do gráfico da figura 5.3 (com o sensor S5 falho) com a temperatura estabilizada em 25°C.

A primeira análise feita foi a do comportamento do algoritmo Híbrido. Seu resultado é muito próximo ao da Média Aritmética, quando não há sensores falhos. Mas existem alguns pontos onde há uma diferença considerável entre os resultados desses dois algoritmos.

Na tabela 5.1 observa-se que algumas medidas obtidas pelos sensores foram coincidentes (mais “concordantes”), fazendo com que o algoritmo Híbrido aproximasse o seu resultado dessas medidas. No caso das medições 5 e 6, os sensores S1 e S7 concordaram com a mesma medida (20,79°C na medição 5 e 20,81ºC na medição 6) e os sensores S9 e S10 também concordaram entre si (20,45°C na medição 5 e 20,49°C na medição 6).

Esse comportamento do algoritmo Híbrido se justifica porque os sensores utilizados possuem a mesma exatidão (±2°C). Além disto, como algumas medidas registradas coincidiram, o intervalo gerado pelos extremos dessas medidas também

coincidiram, diminuindo o número total de intervalos no algoritmo, concentrando as interseções nesses intervalos.

As coincidências de valores nas medições 3 (S4 e S5 com temperatura de 19,89°C) e 7 (S1 e S7 com temperatura de 20,84°C) não foram suficientes para deslocar o valor obtido pelo algoritmo Híbrido de forma abrupta em relação à Média Aritmética. Contudo, se o número de sensores utilizados for menor, a quantidade de intervalos criados no algoritmo será menor. Conseqüentemente, a influência das coincidências entre valores medidos será maior no resultado.

Tabela 5.1 – Nove medições realizadas nos dez sensores de temperatura.

N	Sensor S1	Sensor S2	Sensor S3	Sensor S4	Sensor S5	Sensor S6	Sensor S7	Sensor S8	Sensor S9	Sensor S10
1	20,45	19,81	19,30	19,68	19,74	19,49	20,58	19,90	20,12	20,14
2	20,61	19,95	19,46	19,82	19,84	19,60	20,69	20,01	20,26	20,28
3	20,68	20,01	19,53	19,89	19,89	19,66	20,74	20,06	20,34	20,35
4	20,75	20,08	19,61	19,96	19,92	19,71	20,77	20,09	20,40	20,41
5	20,79	20,11	19,65	20,00	19,95	19,73	20,79	20,12	20,45	20,45
6	20,81	20,13	19,67	20,01	19,96	19,75	20,81	20,14	20,49	20,49
7	20,84	20,15	19,70	20,04	19,99	19,79	20,84	20,18	20,52	20,53
8	20,86	20,18	19,73	20,06	20,00	19,80	20,85	20,18	20,53	20,55
9	20,89	20,20	19,75	20,08	20,02	19,81	20,86	20,19	20,55	20,56

Tabela 5.2 – Aplicação dos algoritmos de Sensores em Consenso aos dados da Tabela 5.1.

N	Média Aritmética	Critério de Chauvenet	Concordância Aproximada	Concordância Inexata (FCA)	Concordância Inexata (CCA)	Fusão Contraditória	Híbrido
1	19,92	19,92	19,89	19,92	19,92	19,85 ± 1,95	19,93
2	20,05	20,05	20,02	20,05	20,05	19,98 ± 1,97	20,06
3	20,12	20,12	20,08	20,12	20,12	20,03 ± 1,98	20,13
4	20,17	20,17	20,13	20,17	20,17	20,09 ± 1,99	20,18
5	20,20	20,20	20,17	20,20	20,20	20,11 ± 2,00	20,43
6	20,22	20,22	20,19	20,22	20,22	20,14 ± 1,99	20,45
7	20,26	20,26	20,22	20,26	20,26	20,17 ± 1,99	20,27
8	20,27	20,27	20,24	20,27	20,27	20,18 ± 2,00	20,29
9	20,29	20,29	20,26	20,29	20,29	20,20 ± 1,99	20,31

O aumento ou a permanência da exatidão no resultado da Fusão Contraditória só é notado porque todos os sensores possuem a mesma exatidão (±2°C). Quando se utilizam sensores com exatidões diferentes, pode ou não ocorrer aumento da exatidão, dependendo do grau de precisão das medidas.

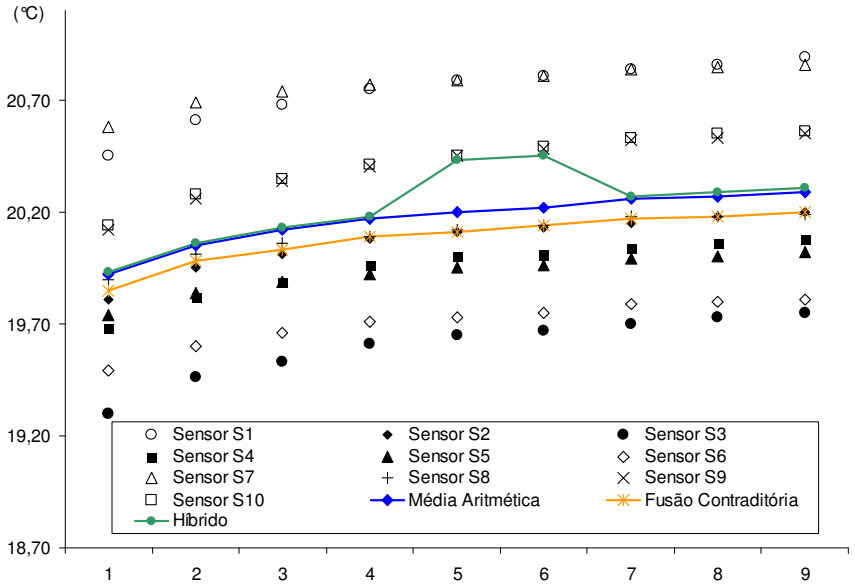


Fig. 5.5 – Comportamento do algoritmo Híbrido (dados das tabelas 5.1 e 5.2).

Apesar do algoritmo de Fusão Contraditória utilizar o mesmo princípio do algoritmo Híbrido, no segundo é realizado uma média ponderada com os valores extremos dos intervalos obtidos pelo algoritmo (ver seção 4.10), explicando assim a diferença no resultado de ambos. O gráfico da figura 5.5 ilustra o comportamento do algoritmo Híbrido em comparação com a Média Aritmética e o Algoritmo de Fusão Contraditória.

O algoritmo de Fusão Contraditória apresenta comportamento semelhante ao algoritmo da Média Aritmética no caso de não ocorrerem sensores falhos. A diferença entre eles está que o algoritmo de Fusão Contraditória leva em consideração a proximidade dos valores obtidos pelos sensores, dando maior “peso” às regiões onde há maior concentração. A Média Aritmética não faz esse tipo de análise.

Os demais algoritmos (FCA, CCA, Critério de Chauvenet e Concordância Aproximada) não apresentaram nenhuma característica importante que poderia ser aproveitada de alguma forma no cenário em que todos os sensores estão em perfeito funcionamento. O Critério de Chauvenet não descartou praticamente nenhum valor registrado pelos sensores. Os algoritmos FCA e CCA não apresentaram nenhuma diferença substancial em relação à Média Aritmética e o algoritmo de Concordância

Aproximada revelou ligeiro afastamento da Média Aritmética por sempre eliminar os valores dos extremos.

Os dados da Tabela 5.3 foram extraídos da segunda bateria de medidas, no cenário em que foram introduzidas falhas no sensor S5. A Tabela 5.4 mostra a aplicação dos algoritmos de Sensores em Consenso aos dados da Tabela 5.3.

O gráfico da figura 5.6 ilustra a influência da presença de sensores falhos no cálculo da Média Aritmética. Observa-se nitidamente a diferença entre a Média Aritmética computada com e sem a presença do sensor S5.

Tabela 5.3 – Doze medições realizadas nos dez sensores de temperatura. S5 apresentando falhas.

N	Sensor S1	Sensor S2	Sensor S3	Sensor S4	Sensor S5	Sensor S6	Sensor S7	Sensor S8	Sensor S9	Sensor S10
1	26,01	25,43	24,88	25,19	**25,85**	24,93	26,07	25,31	25,74	25,71
2	26,04	25,45	24,91	25,22	**29,14**	24,95	26,10	25,34	25,77	25,73
3	26,06	25,48	24,93	25,24	**27,00**	24,96	26,11	25,35	25,79	25,75
4	26,07	25,49	24,93	25,25	**29,19**	24,98	26,12	25,36	25,79	25,76
5	26,07	25,49	24,94	25,25	**32,42**	24,97	26,11	25,35	25,79	25,75
6	26,09	25,50	24,95	25,27	**24,33**	24,99	26,14	25,38	25,82	25,77
7	26,10	25,52	24,96	25,28	**26,42**	25,01	26,15	25,40	25,83	25,79
8	26,12	25,54	24,99	25,30	**38,08**	25,02	26,16	25,41	25,84	25,81
9	26,14	25,55	25,00	25,32	**36,85**	25,05	26,19	25,43	25,87	25,83
10	26,16	25,57	25,02	25,34	**31,44**	25,06	26,21	25,45	25,89	25,84
11	26,17	25,59	25,04	25,35	**26,92**	25,08	26,22	25,46	25,90	25,86
12	26,19	25,60	25,05	25,36	**23,32**	25,09	26,23	25,48	25,91	25,87

Tabela 5.4 – Aplicação dos algoritmos de Sensores em Consenso aos dados da Tabela 5.3.

N	Média Aritmética	Critério de Chauvenet	Concordância Aproximada	Concordância Inexata (FCA)	Concordância Inexata (CCA)	Fusão Contraditória	Híbrido
1	25,51	25,51	25,55	25,51	25,51	25,57 ± 1,86	25,50
2	25,86	25,86	25,57	25,86	25,86	25,59 ± 1,86	25,73
3	25,67	25,67	25,59	25,67	25,67	25,62 ± 1,86	25,69
4	25,89	25,89	25,60	25,89	25,89	25,62 ± 1,87	25,76
5	26,21	26,21	25,60	25,59	25,52	25,55 ± 1,80	25,53
6	25,42	25,42	25,48	25,42	25,42	25,44 ± 1,94	25,41
7	25,65	25,65	25,64	25,65	25,65	25,66 ± 1,86	25,65
8	26,83	26,83	25,65	25,70	25,58	25,61 ± 1,80	25,58
9	26,72	26,72	25,67	25,71	25,60	25,63 ± 1,80	25,60
10	26,20	26,20	25,69	25,67	25,62	25,64 ± 1,80	25,62
11	25,76	25,76	25,70	25,76	25,76	25,72 ± 1,87	25,78
12	25,41	25,41	25,58	25,41	25,41	25,54 ± 1,94	25,36

A meta é que o valor consensual esteja dentro da faixa de valores limitada pela menor e pela maior medida obtida pelos sensores corretos, isto é, pelos sensores que não apresentam medidas demasiadamente espúrias. Na figura 5.5 observa-se que os três algoritmos se mantiveram dentro da faixa esperada. Pode-se observar pelas

Tabela 5.1 e 5.2 que todos os algoritmos também obtiveram dados dentro da faixa esperada.

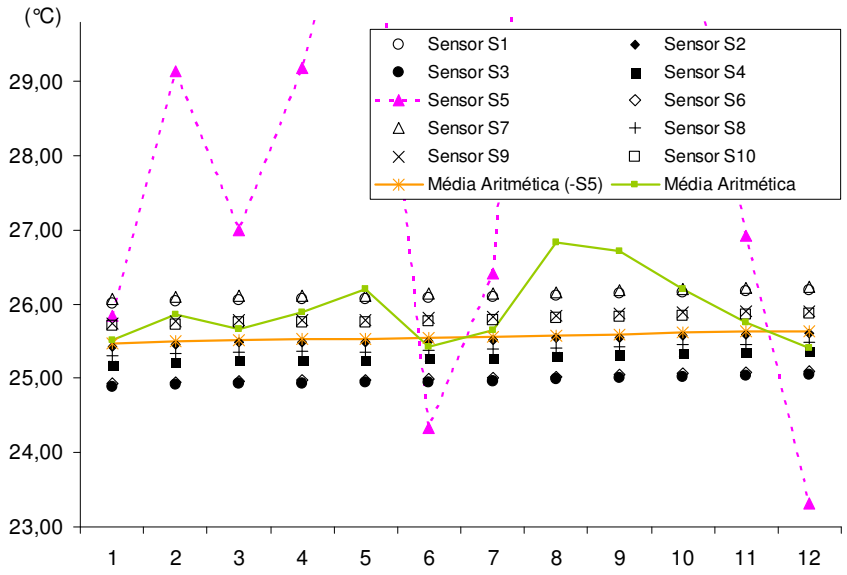


Fig. 5.6 – Influência da presença de sensores falhos na Média Aritmética (dados das tabelas 5.3 e 5.4).

A Média Aritmética obtém resultados fora da faixa esperada na presença de medidas falhas. Isso foi comprovado pelos resultados mostrados na figura 5.6. As medições 5, 8 e 9 obtiveram um valor para a Média Aritmética fora da faixa determinada pelas medidas mínima e máxima. As características dessa técnica estatística até podem ser utilizadas como critério de consenso das medidas quando não há valores discordantes, mas na presença deles, a utilização da Média Aritmética se torna imprópria.

O gráfico da figura 5.7 ilustra o resultado dos algoritmos de Sensores em Consenso na presença de falha do sensor S5 (ver dados nas tabelas 5.3 e 5.4). Observa-se que todos os algoritmos obtiveram resultados entre aproximadamente 25,3 e 26,0°C. Pelo gráfico da figura 5.6, verifica-se que os resultados estão dentro da faixa esperada (entre o maior e menor valor de temperatura obtido pelos sensores).

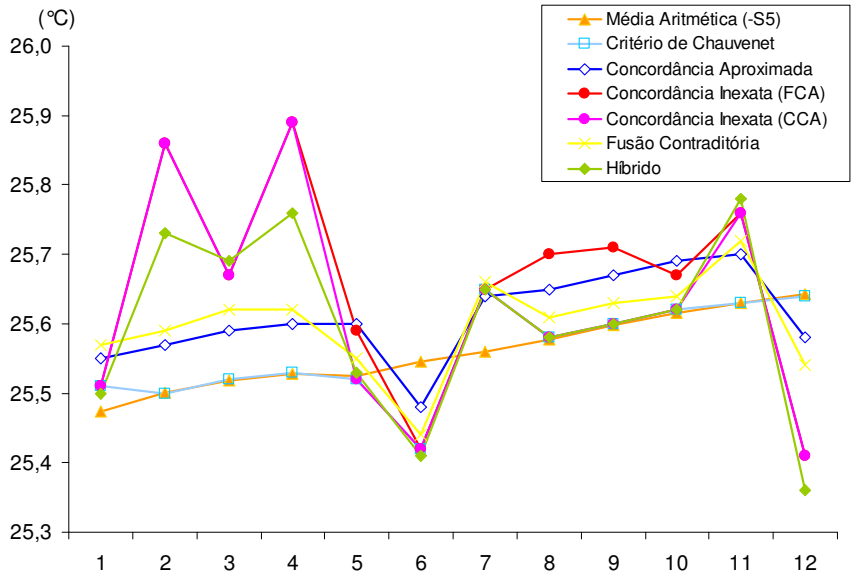


Fig. 5.7 – Resultados dos algoritmos de Sensores em Consenso na presença de um sensor falho (dados da tabela 5.4).

Apesar de cada algoritmo ter obtido um valor diferente dentro da faixa esperada, o Critério de Chauvenet eliminou os valores do sensor falho de forma a coincidir em diversos pontos o resultado final com a Média Aritmética calculada excluindo o valor do sensor S5. O Critério de Chauvenet só não realiza essa coincidência quando o valor do sensor falho coincidentemente é registrado próximo aos valores dos sensores corretos, pois, nesse caso, o valor do sensor falho não fica fora da distribuição normal.

Da mesma forma, os algoritmos de Concordância Aproximada e Fusão Contraditória aplicados no cenário onde o sensor S5 apresenta falhas apresentam "saltos" quando o sensor falho obteve valores próximos aos valores registrados pelos outros sensores.

Durante o experimento, os dados foram monitorados através de gráficos e tabelas na interface do usuário e, posteriormente, os dados foram consultados na forma de um arquivo XLS. As figuras 5.8 e 5.9 ilustram, respectivamente, a configuração do Monitor utilizado na câmara climática e a configuração da tabela (Temperaturas) criada para a visualização dos dados.

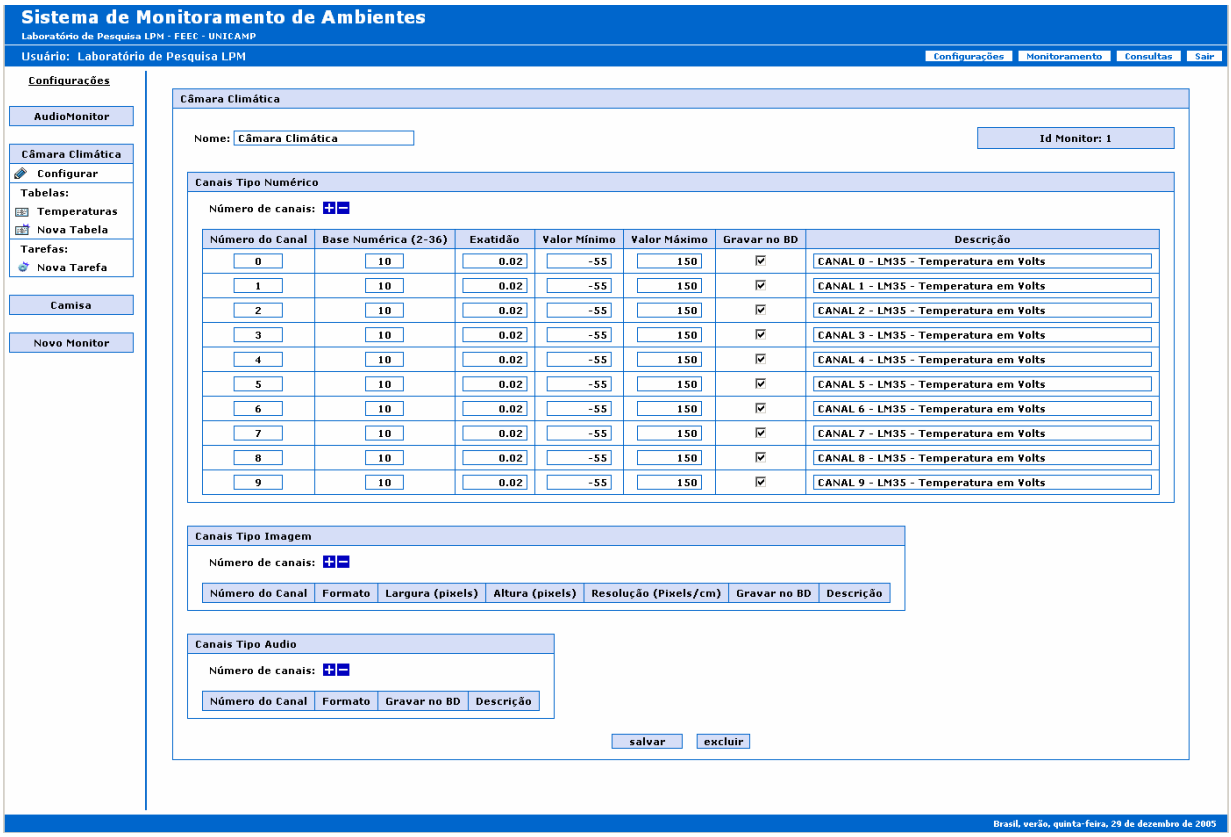


Fig. 5.8 – Configurações do módulo Câmara Climática

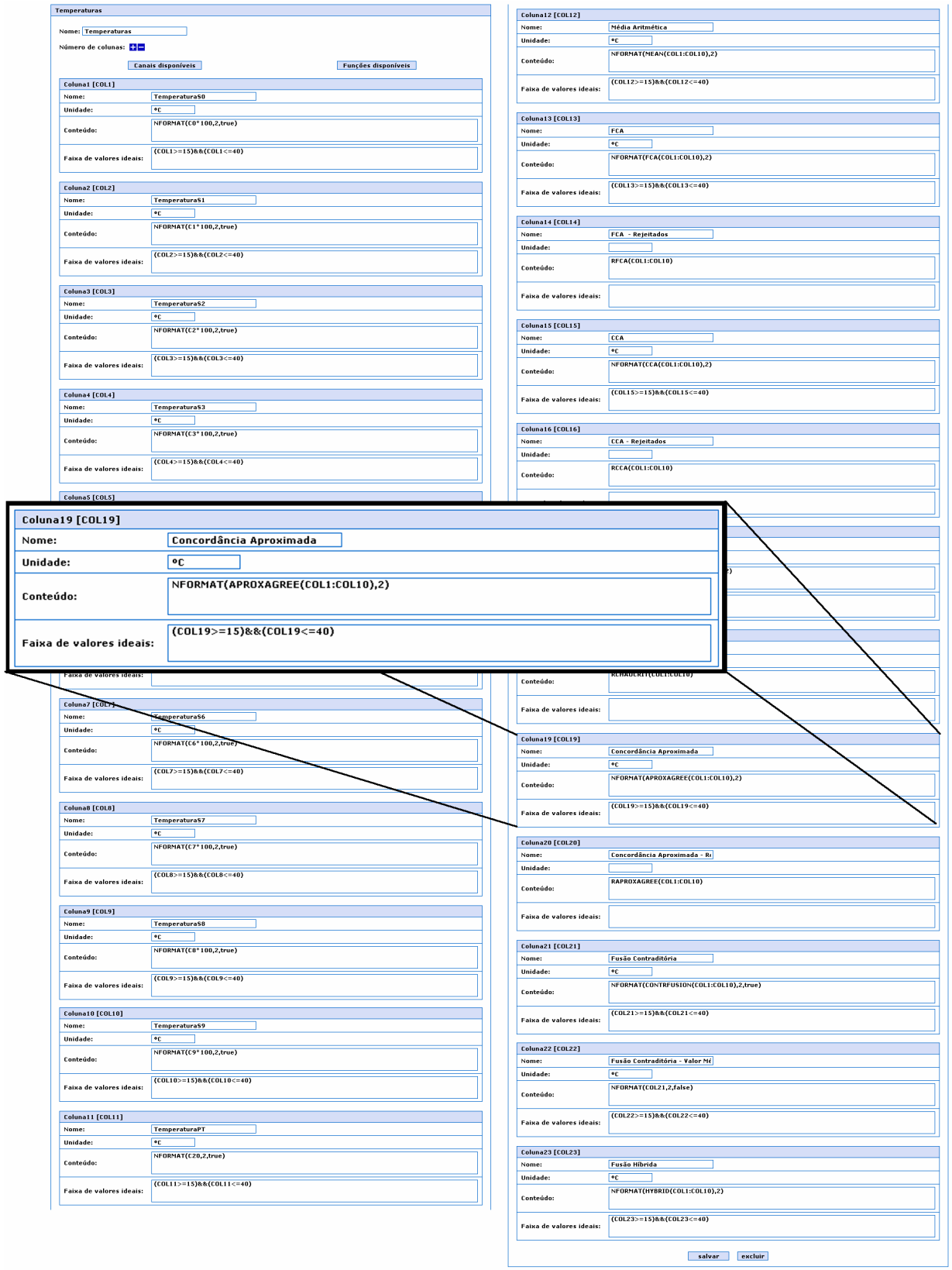


Fig. 5.9 – Configurações da tabela "Temperaturas" para visualização dos dados do módulo Câmara Climática.

5.2 – Monitoramento de uma planta

A segunda aplicação experimental do Sistema de Monitoramento de Ambientes foi realizada com o objetivo de monitorar o ambiente no qual uma planta do gênero *Schefflera arboricola* se encontrava. Com o auxílio de quatro sensores de naturezas diferentes, buscou-se determinar o conforto térmico da planta e a percentagem de manchas da folha de um ramo como possível forma de detecção de contaminação por fungos.

Os parâmetros ideais de temperatura, umidade e luminosidade para o cultivo da planta *Schefflera arboricola* ainda não foram determinados experimentalmente[2]. Considerou-se que a temperatura ideal de cultivo estaria entre 18 e 20°C e umidade relativa ideal entre 75 e 80%RH. A luminosidade adequada para a planta estaria entre 20.000 e 50.000 lux.

As manchas típicas das folhas da *Schefflera arboricola* foram consideradas para a análise de um possível ataque por fungos nas folhas de plantas. Nesta espécie vegetal as manchas não caracterizam fungos, mas podem ser utilizadas como um teste de aplicação do Sistema de Monitoramento na detecção de fungos em folhas de outras espécies vegetais.

Através de sensores de temperatura, umidade, luminosidade e uma câmera de vídeo buscou-se combinar os dados coletados de forma a estabelecer um diagnóstico da planta. A figura 5.10 mostra o arranjo físico utilizado no monitoramento.

O emprego dos sensores de umidade e luminosidade no monitoramento foi realizado de forma direta, analisando se seus valores estão dentro das faixas de valores ideais. O sensor de temperatura foi utilizando aplicando-se uma lógica *Fuzzy* que determina que se a temperatura estiver entre 18 e 20°C ela está adequada e abaixo de 13 e acima de 25, inadequada (figura 5.11).

[2] Informação obtida através de consulta aos professores do departamento de Botânica do Instituto de Biologia da UNICAMP

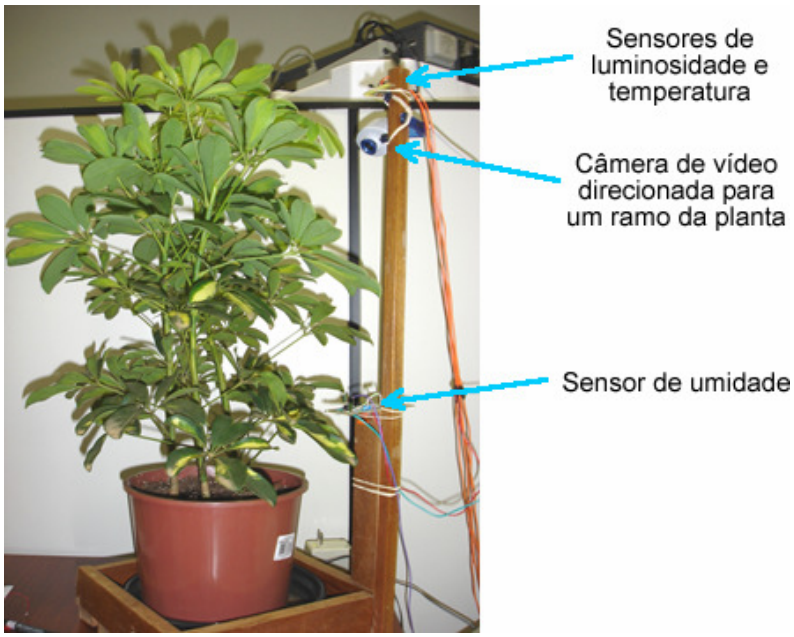


Fig. 5.10 – Arranjo de sensores utilizados no monitoramento de uma planta *Schefflera arboricola*.

Com o intuito de aprimorar os valores registrados pelos sensores, seria interessante a inserção de vários sensores de temperatura, umidade e luminosidade espalhados ao redor da planta e a aplicação dos algoritmos de Sensores em Consenso na determinação de valores consensuais representativos do ambiente.

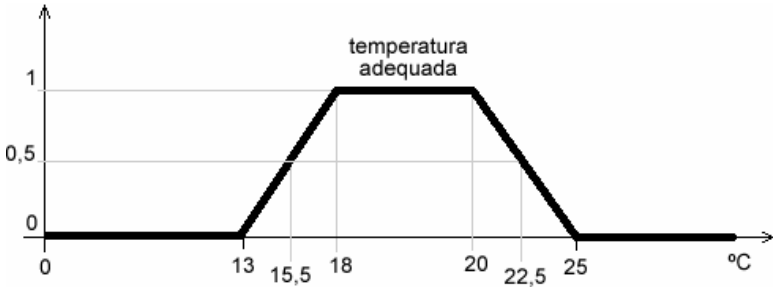


Fig. 5.11 – Lógica *Fuzzy* aplicada à temperatura

Juntamente com a análise isolada da temperatura e umidade, calculou-se um dos índices de conforto térmico que leva em consideração apenas a umidade a temperatura [CON05], sem a necessidade da velocidade do ar. Esse índice revela uma temperatura mais refinada, levando em consideração a umidade ambiente.

O sensor de luminosidade também foi utilizado como auxílio na regulação da intensidade luminosa da imagem registrada pela câmera de vídeo. Foram aplicados dois filtros à imagem: o primeiro, denominado Filtro *Gamma*, fez a correção do brilho da imagem de acordo com a intensidade luminosa registrada pelo sensor de luminosidade; o segundo, Filtro *Lookup*, realizou uma re-coloração da imagem convertendo-a de preto e branco para tons de verde. Em seguida, pode-se determinar a quantidade de *pixels* da imagem que apresentavam tonalidades de verde entre 200 e 250 (padrão RGB, com cada componente da cor variando entre 0 a 255), correspondente, aproximadamente, à tonalidade das manchas.

A análise final, ou seja, o diagnóstico da planta, foi realizado unindo as condições de luminosidade (pouca, muita ou boa), temperatura (adequada ou inadequada), umidade (baixa, alta, adequada), presença ou ausência de fungos e o índice de conforto térmico. Se a luminosidade estiver abaixo de 20.000 lux, ela é considerada “pouca”; se estiver acima de 50.000 lux, “muita”; e entre 20.000 e 50.000 lux, ela é considerada “boa”. Temperaturas com resultado da lógica *Fuzzy* (figura 5.11) maiores de 0,5 (ou seja, $15,5 \leq T \leq 22,5$) são consideradas “adequadas” e se menores que este valor são “inadequadas”. A umidade é considerada “baixa”, se ela estiver abaixo de 70%RH, “alta” se estiver acima de 80%RH e “adequada” se estiver entre esses dois valores.

As figuras 5.12, 5.13, 5.14 e 5.15 mostram as configurações do monitor, as configurações da tabela de visualização e a resposta de uma consulta com os resultados da Fusão dos Sensores, respectivamente.

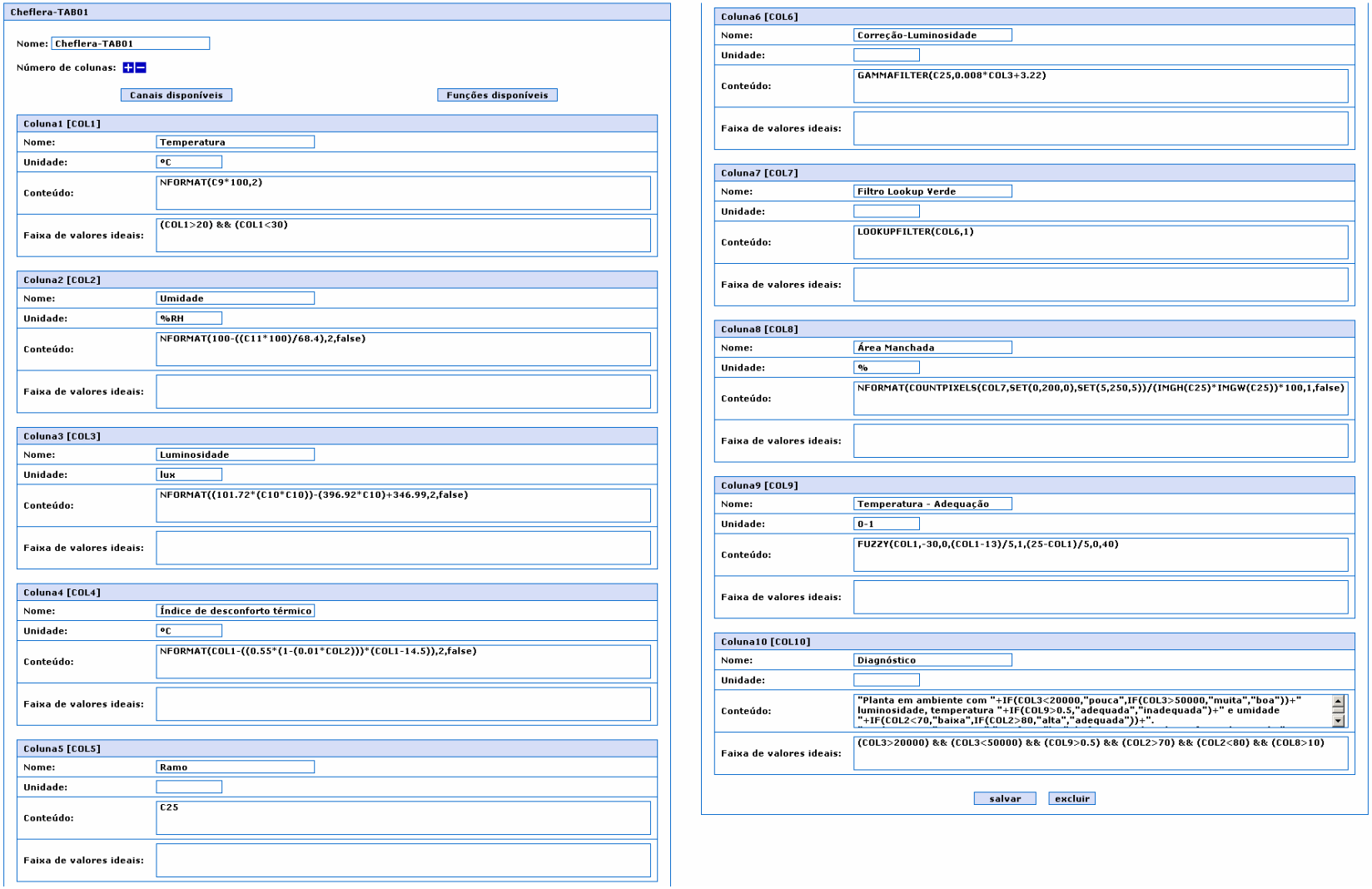


Fig. 5.12 – Configuração da tabela "*Schefflera*-TAB01" para o monitoramento da planta *Schefflera arboricola*.

Chefera - Cheflera-TAB01

Últimos 5 registros [5 registros]

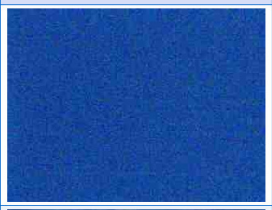
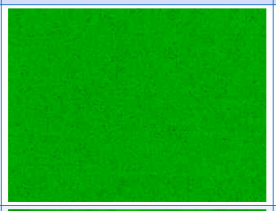
Data Hora	Temperatura (°C)	Umidade (%RH)	Luminosidade (lux)	Índice de desconforto térmico (°C)	Ramo	Correção-Luminosidade	Filtro Lookup Verde
28/12/2005 15:09:18	28.00 ± 2.00	46.01	28.36	23.99			
28/12/2005 15:09:03	29.00 ± 2.00	46.01	89.91	24.69			
28/12/2005 15:08:47	29.00 ± 2.00	45.00	140.00	24.61			
28/12/2005 15:08:32	29.00 ± 2.00	45.00	171.02	24.61			
28/12/2005 15:08:17	29.00 ± 2.00	46.01	220.53	24.69			

Fig. 5.13 – Resultado de uma consulta dos dados da planta *Schefflera arborícola* – Parte 1.

Área Manchada (%)	Temperatura - Adequação (0-1)	Diagnóstico
0.0	0.0	Planta em ambiente com pouca luminosidade, temperatura inadequada e umidade baixa. Ausência de fungos. Índice de conforto térmico de 23.99ºC
15.9	0.0	Planta em ambiente com pouca luminosidade, temperatura inadequada e umidade baixa. Presença de fungos. Índice de conforto térmico de 24.69ºC
31.4	0.0	Planta em ambiente com pouca luminosidade, temperatura inadequada e umidade baixa. Presença de fungos. Índice de conforto térmico de 24.61ºC
31.7	0.0	Planta em ambiente com pouca luminosidade, temperatura inadequada e umidade baixa. Presença de fungos. Índice de conforto térmico de 24.61ºC
34.0	0.0	Planta em ambiente com pouca luminosidade, temperatura inadequada e umidade baixa. Presença de fungos. Índice de conforto térmico de 24.69ºC

Documento gerado em 09/01/2006 13:18:52

Fig. 5.14 – Resultado de uma consulta dos dados da planta *Schefflera arborícola* – Parte 2.

O exemplo do monitoramento da planta *Schefflera arboricola* é apenas uma forma de ilustrar as possibilidades de Fusão de Sensores no Sistema de Monitoramento de Ambientes. Os resultados foram coerentes, visto que a planta se encontrava dentro de uma sala onde havia baixa luminosidade e ar condicionado ligado, mas os critérios utilizados na determinação das condições ambientes não tem embasamento científico profundo.

Resumo

O capítulo 5 descreveu duas aplicações experimentais elaboradas com intuito de validar o sistema desenvolvido.

A primeira aplicação possibilitou uma análise mais profunda dos algoritmos de Sensores em Consenso através dos resultados obtidos na combinação dos dados de dez sensores de temperatura em uma câmara climática.

A segunda aplicação utilizou diferentes tipos de sensores (temperatura, umidade, luminosidade e câmera de vídeo) no monitoramento de uma planta *Schefflera arboricola*. Apesar dos critérios de monitoramento não serem determinados cientificamente por biólogos ou agrônomos, o experimento permitiu ilustrar a possibilidade de se utilizar o sistema com diferentes tipos de sensores.

O capítulo 6 fará uma conclusão do trabalho realizado, avaliando o desempenho e limitações do sistema desenvolvido e traçará perspectivas para o campo de Fusão de Sensores.

6

Discussão e Conclusões

"Be wary of proposals for synergistic systems.
Most of the time when you try to make 2 + 2 = 5,
you end up with 3... and sometimes 1.9." [1]
Charles A. Fowler

O sistema implementado para Monitoramento de Ambientes com aplicações de técnicas de Fusão de Sensores apresentou resultados satisfatórios dentro do que se espera em um ambiente computacional que depende da Internet. Apesar do atraso provocado pela comunicação, monitoramentos que não precisem de uma resposta instantânea podem se beneficiar do sistema desenvolvido.

O sistema pode ser utilizado tanto para realizar uma análise de um ambiente, extraindo informações e verificando o seu comportamento, como para realizar o controle de ambientes dos quais se conheça *a priori* o comportamento e a dependência das variáveis sensoriadas.

O estudo e a aplicação dos algoritmos de Sensores em Consenso evidenciaram que sua utilização depende do objetivo que se quer atingir com a combinação dos dados. Em ambientes onde não há referências, a participação de

[1] "Seja cauteloso com a proposta para sistemas sinergéticos. A maioria das vezes quando você tenta fazer 2 + 2 = 5, você termina com 3... e às vezes 1,9".

diversos sensores na determinação de um único valor que representa a variável sensoriada é fundamental para evitar a consideração de medidas obtidas por sensores falhos.

Apesar do sistema não ter sido projetado para realizar a fusão dos dados de modo temporal, ou seja, não se consegue combinar dados de uma determinada data e hora com dados de outra data e hora, é possível concatenar dados seqüências em um mesmo documento XML com data e hora específicos e realizar a combinação dos dados seqüências como se eles pertencessem à mesma data e hora. Contudo, essa prática requer uma certa quantidade de processamento no elemento monitor.

O envio de documentos XML através do protocolo HTTP mostrou-se extremamente prático na organização dos dados e facilmente implementável nos monitores. Contudo, a interface de configuração dos monitores e das tabelas não se mostrou prática, principalmente no que diz respeito à escrita das expressões que realizam as fusões dos dados dos sensores nas colunas das tabelas e às expressões que realizam tarefas. Uma outra forma de interface deve ser elaborada para facilitar o entendimento do usuário e agilizar o processo de cadastramento de módulos.

Este trabalho faz parte do projeto de uma rede de sensores sem fio ainda a ser concebida. Tanto a parte teórica como as aplicações experimentais do sistema desenvolvido forneceram elementos essenciais para a parte computacional de uma rede de sensores inteligente capaz de combinar os dados de diferentes sensores e interagir com o ambiente sensoriado.

Muita pesquisa ainda deve ser realizada para se entender melhor os mecanismos Fusão de Sensores presentes na natureza e aqueles de relevância em sistemas de controle.

A interdisciplinaridade é fundamental na área da fusão, visto que os sistemas computacionais que realizam a Fusão de Sensores fornecem as ferramentas e a estrutura da combinação dos dados, mas os parâmetros de controle e as necessidades de fusão devem ser estabelecidos pelas áreas que empregarão o sistema.

Ainda há muito a ser feito com relação aos mecanismos de fusão. Por mais que se possam classificar as fusões em níveis e desenvolver ferramentas que

permitam diferentes tipos de combinações de dados, a “essência” da combinação de informações distintas ainda não está bem explicada.

A própria combinação dos sentidos no ser humano (tato, paladar, audição e olfato) não é bem conhecida, mas sabemos que ela funciona, pois uma pessoa se locomove utilizando uma combinação da audição, visão e do tato, observa uma paisagem em profundidade utilizando a fusão de duas imagens geradas pelos olhos e tem a sensação do gosto de um alimento através da combinação do paladar e do olfato.

Apesar da singeleza desse trabalho, ele conseguiu reunir os principais aspectos da Fusão de Sensores e propôs uma estrutura de dados que permite a inserção de dados de naturezas diversas e diferentes algoritmos de fusão com o objetivo de monitorar ambientes através da Internet.

Referências Bibliográficas

[ALA98] ALAMI, R et al. An Architecture for Autonomy. **International Journal of Robotics Research**, Toulouse. 1998.

[AHO77] AHO, Alfred; ULLMAN, Jeffrey. **Principles of Compiler Design.** London: Addison-Wesley Publishing Company, 1977.

[APP98] APPEL, Andrew W.. **Modern Compiler Implementation in Java.** United Kingdom: Cambridge University Press, 1998.

[BED99] BEDWORTH, Mark D.. **Source Diversity and Feature-Level Fusion.** United Kingdom: British Crown, 1999.

[BRO98] BROOKS, R. R.; IYENGAR, S. S.. **Multi-Sensor Fusion:** Fundamentals and Applications with Software. New Jersey: Prentice Hall, 1998.

[BUS87] BUSSAB, Wilson O.; MORETTIN, Pedro A.. **Estatística Básica:** Métodos Quantitativos. São Paulo: Editora Atual, 1987.

[CON05] CONFORTO Térmico. Departamento de Ciências Atmosféricas da Universidade de São Paulo. Disponível em: <http://www.master.iag.usp.br/conforto/index.html>. Acesso em: 15 dez. 2005.

[DEI00] DEITEL, H. M. et al. **XML How to program.** New Jersey: Prentice Hall, 2000. 934p.

[DOL81] DOLEV, Danny. **The Byzantine Gerenals Strike Again.** Stanford: National Science Foundation, 1981.

[DOL86] DOLEV, Danny et al. Reaching Approximate Agreement in the Presence of Faults. **Journal of the Association for Computing Machinery**, p. 499-516. jul. 1986.

[ELM89] ELMASRI, Ramez; NAVATHE, Sham. **Fundamentals of database systems.** California: The Benjamin/Cummings Publishing Company Inc., 1989.

[ELM01] ELMENREICH, W.; PITZEK, S.. The Time-Triggered Sensor Fusion Model. **Proceedings of the 5th IEEE International Conference on Intelligent Engeneering Systems**, Vienna, 14 set. 2001.

[ELM02] ELMENREICH, Wilfried. **Sensor Fusion in Time-Triggered Systems.** 2002. 157 f. Doutorado - Institut Für Technische Informatik, Wien, 2002.

[KLE99] KLEIN, Lawrence A.. **Sensor and Data Fusion Concepts and Applications.** 2. ed. Washington: SPIE Optical Engeneering Press, 1999. 226 p.

[LAM82] LAMPORT, Leslie; SHOSTAK, Robert; PEASE, Marshall. The Byzantine Generals Problem. **ACM Transactions on Programming Language and Systems**, California, v. 4, n. 3, p.382-401, jul. 1982.

[LEH97] LEHMANN, Kevin. **Rejection of Data.** New Jersey: Department Of Chemistry Princeton University, 1997.

[LUO90] LUO, Ren C.; KAY, Michael G.. A Tutorial on Multisensor Integration and Fusion. In: IECON '90, 1990, Raleigh. **16th Annual Conference of IEEE.** Pacific Grove: IEEE, 1990. p. 707 - 722.

[LUO95] LUO, Ren C.; KAY, Michael G.. **Multisensor Integration and Fusion for Intelligent Machines and Systems.** Norwood: Ablex Publishing Corporation, 1995. 688 p.

[JAV05] JAVA Home Page. Sun Microsystems. Disponível em: <http://java.sun.com>. Acesso em: 5 out. 2005.

[JAY94] JAYAUMHA, D. N.. Fault Tolerance in a Multisensor Environment. **Proceedings of the 13th Symposium on Reliable Distibuted System**, IEEE, 1994.

[JOS99] JOSHI, Rajive; SANDERSON, Arthur C.. **Multisensor Fusion:** A minimal representation framework. Singapore: World Scientific Publishing Co., 1999. 315 p.

[MAH85] MAHANEY, Stephen; SCHNEIDER, Fred B.. **Inexact Agreement: Accuracy, Precision and Graceful Degradation**. In: SYMPOSIUM ON PRINCIPLES OF DISTRIBUTED COMPUTING, 1985, p. 237 - 249.

[MAR90] MARZULLO, Keith. Tolerating Failures of Continuous-Valued Sensors. **ACM Transactions on Computer Systems**, v. 8, n. 4, p.284-304, nov. 1990.

[MUL97] MULLER, Pierre-Alain. **Instant UML.** Paris: Wrox Press Ltd., 1997. 345 p.

[SAX05] SAX Project. Simple API for XML. Disponível em: <http://www.saxproject.org>. Acesso em: 10 nov. 2005.

[SCH94] SCHUCH, Luiz Alexandre. Apêndice A - Noções sobre sensação térmica. In: SCHUCH, Luiz Alexandre. **Operação Antártica:** Uma experiência vivenciada. São Paulo: Edição Conjunta da Universidade Federal de Santa Maria (UFSM) e do Instituto Nacional de Pesquisas Espaciais (INPE), 1994. 178 p.

[TAN03] TANENBAUM, Andrews S.. **Redes de Computadores.** Rio de Janeiro: Campus, 2003. 945 p.

[TAY97] TAYLOR, John R.. **An Introduction to Error Analysis:** The Study of Uncertainties in Physical Measurements. 2. ed. California: University Science Books, 1997. 327 p.

[WEL04] WELCH, Greg; BISHOP, Gary. **An Introduction to the Kalman Filter.** Chapel Hill: Departament of Computer Science, University of North Carolina, 2004.

[WOR05A] WORLD Wide Web Consortium. Extensible Markup Language (XML). Disponível em: <http://www.w3.org/XML>. Acesso em: 10 nov. 2005.

[WOR05B] WORLD Wide Web Consortium. Document Object Model (XML). Disponível em: <http://www.w3.org/XML>. Acesso em: 10 nov. 2005.

Apêndice A

Especificação DTD

O DTD abaixo especifica a estrutura dos elementos que o documento XML contento as leituras dos sensores deve apresentar. No sistema, essa definição DTD está contida no arquivo **leitura.dtd**.

```
<!ELEMENT LEITURA (MONITOR, DATA, HORA, CANAL*)>
<!ELEMENT MONITOR (#PCDATA)>
<!ELEMENT DATA (#PCDATA)>
<!ELEMENT HORA (#PCDATA)>
<!ELEMENT CANAL (ID_CANAL, VALOR)>
<!ELEMENT ID_CANAL (#PCDATA)>
<!ELEMENT VALOR (#PCDATA)>
```

O XML abaixo é um exemplo de um monitor identificado pelo número 6 que recebe o valor de dois canais, 1 e 2, com os valores 32.656 e 32.637, respectivamente. Os elementos **DATA** e **HORA** não estão preenchidos, fazendo com que o sistema considere a data e a hora de chegado do documento no Servidor como a data e hora de leitura dos dados dos canais.

```
<?xml version="1.0"?>
<!DOCTYPE LEITURA SYSTEM "leitura.dtd">
<LEITURA>
	<MONITOR> 6 </MONITOR>
	<DATA></DATA>
	<HORA></HORA>
	<CANAL>
		<ID_CANAL> 1 </ID_CANAL>
		<VALOR> 32.656 </VALOR>
	</CANAL>
	<CANAL>
		<ID_CANAL> 3 </ID_CANAL>
		<VALOR> 32.637 </VALOR>
	</CANAL>
</LEITURA>
```

Apêndice B

Criação das Tabelas do Banco de Dados

```
CREATE TABLE Usuario (
   idUsuario INT NOT NULL AUTO_INCREMENT,
   usuario VARCHAR(50),
   senha TEXT,
   nome VARCHAR(255),
   UNIQUE (usuario),
   INDEX (usuario),
   PRIMARY KEY (idUsuario),
   INDEX (idUsuario)
) TYPE=INNODB CHARSET=latin1;


CREATE TABLE Monitor (
   idMonitor INT NOT NULL AUTO_INCREMENT,
   idUsuario INT NOT NULL,
   nome VARCHAR(255),
   PRIMARY KEY (idMonitor),
   FOREIGN KEY (idUsuario) REFERENCES Usuario(idUsuario) ON DELETE CASCADE,
   UNIQUE (idUsuario, nome),
   INDEX (idMonitor),
   INDEX (idMonitor, idUsuario)
) TYPE=INNODB CHARSET=latin1;


CREATE TABLE CanalMonitor (
  idCanalMonitor INT NOT NULL AUTO_INCREMENT,
  idMonitor INT NOT NULL DEFAULT '0',
  idCanal INT NOT NULL DEFAULT '0',
  descricao TEXT,
  gravarBD INT(1) NOT NULL DEFAULT '1',
  PRIMARY KEY (idCanalMonitor),
  FOREIGN KEY (idMonitor) REFERENCES Monitor(idMonitor) ON DELETE CASCADE,
  UNIQUE (idMonitor, idCanal),
  INDEX (idMonitor, idCanalMonitor)
) TYPE=INNODB CHARSET=latin1;


CREATE TABLE TipoAudio (
  idTipoAudio INT NOT NULL DEFAULT '0',
  tipoAudio VARCHAR(255) DEFAULT NULL,
  PRIMARY KEY (idTipoAudio),
  UNIQUE (tipoAudio)
) TYPE=INNODB CHARSET=latin1;
```

```
CREATE TABLE CanalAudio (
  idCanalMonitor INT NOT NULL DEFAULT '0',
  idTipoAudio INT DEFAULT '0',
  PRIMARY KEY (idCanalMonitor),
  FOREIGN KEY (idCanalMonitor) REFERENCES CanalMonitor(idCanalMonitor) ON DELETE CASCADE,
  FOREIGN KEY (idTipoAudio) REFERENCES TipoAudio(idTipoAudio) ON DELETE CASCADE,
  INDEX (idCanalMonitor)
) TYPE=INNODB CHARSET=latin1;


CREATE TABLE TipoImagem (
  idTipoImagem INT NOT NULL DEFAULT '0',
  tipoImagem VARCHAR(255) DEFAULT NULL,
  PRIMARY KEY (idTipoImagem),
  UNIQUE (tipoImagem)
) TYPE=INNODB CHARSET=latin1;


CREATE TABLE CanalImagem (
  idCanalMonitor INT NOT NULL DEFAULT '0',
  idTipoImagem INT DEFAULT '0',
  altura INT DEFAULT '0',
  largura INT DEFAULT '0',
  resolucao DOUBLE DEFAULT '0',
  PRIMARY KEY (idCanalMonitor),
  FOREIGN KEY (idCanalMonitor) REFERENCES CanalMonitor(idCanalMonitor) ON DELETE CASCADE,
  FOREIGN KEY (idTipoImagem) REFERENCES TipoImagem(idTipoImagem) ON DELETE CASCADE,
  INDEX (idCanalMonitor)
) TYPE=INNODB CHARSET=latin1;


CREATE TABLE CanalNumerico (
  idCanalMonitor INT NOT NULL DEFAULT '0',
  base INT NOT NULL DEFAULT '10',
  exatidao VARCHAR(255) NOT NULL DEFAULT '0',
  valorMinimo VARCHAR(255) DEFAULT '0',
  valorMaximo VARCHAR(255) DEFAULT '0',
  PRIMARY KEY (idCanalMonitor),
  FOREIGN KEY (idCanalMonitor) REFERENCES CanalMonitor(idCanalMonitor) ON DELETE CASCADE,
  INDEX (idCanalMonitor)
) TYPE=INNODB CHARSET=latin1;


CREATE TABLE TabelaMonitor (
  idTabela INT NOT NULL AUTO_INCREMENT,
  nome VARCHAR(255) NOT NULL DEFAULT '',
  idMonitor INT NOT NULL,
  PRIMARY KEY (idTabela),
  UNIQUE (nome,idMonitor),
  FOREIGN KEY (idMonitor) REFERENCES Monitor(idMonitor) ON DELETE CASCADE,
  INDEX (idTabela)
) TYPE=INNODB CHARSET=latin1;


CREATE TABLE ColunaTabela (
  idColuna INT NOT NULL AUTO_INCREMENT,
  idTabela INT NOT NULL,
  nome VARCHAR(255) NOT NULL DEFAULT '',
  unidade VARCHAR(10) NOT NULL DEFAULT '',
  conteudo TEXT NOT NULL DEFAULT '',
  faixaDeValoresIdeais TEXT NOT NULL DEFAULT '',
  PRIMARY KEY (idColuna),
  FOREIGN KEY (idTabela) REFERENCES TabelaMonitor(idTabela) ON DELETE CASCADE,
  INDEX (idTabela,idColuna)
) TYPE=INNODB CHARSET=latin1;
```

```
CREATE TABLE TarefaMonitor (
  idTarefa INT NOT NULL AUTO_INCREMENT,
  nome VARCHAR(255) NOT NULL DEFAULT '',
  condicao TEXT NOT NULL DEFAULT '',
  execucao TEXT NOT NULL DEFAULT '',
  idMonitor INT NOT NULL,
  PRIMARY KEY (idTarefa),
  FOREIGN KEY (idMonitor) REFERENCES Monitor(idMonitor) ON DELETE CASCADE,
  INDEX (idTarefa)
) TYPE=INNODB CHARSET=latin1;


CREATE TABLE LeituraMonitor (
  idLeitura INT NOT NULL AUTO_INCREMENT,
  idMonitor INT NOT NULL,
  dataHora DATETIME NOT NULL,
  PRIMARY KEY (idLeitura),
  FOREIGN KEY (idMonitor) REFERENCES Monitor(idMonitor) ON DELETE CASCADE,
  INDEX (idMonitor),
  INDEX (idLeitura,idMonitor),
  INDEX (idLeitura,idMonitor,dataHora)
) TYPE=INNODB CHARSET=latin1;


CREATE TABLE CanalLeitura (
  idLeitura INT NOT NULL,
  idCanal INT NOT NULL,
  valor MEDIUMBLOB NOT NULL,
  PRIMARY KEY (idLeitura,idCanal),
  FOREIGN KEY (idLeitura) REFERENCES LeituraMonitor(idLeitura) ON DELETE CASCADE,
  INDEX (idLeitura,idCanal)
) TYPE=INNODB CHARSET=latin1;
```

Apêndice C

Conversão de Bases Numéricas

Para converter um número de uma base b para uma base b' qualquer, um processo prático utilizado é converter o número da base b para a base 10 e em seguida converter da base 10 para a base b' desejada.

Os números utilizados no Sistema podem estar em qualquer base de 2 a 36. Os caracteres que representam os algarismos numéricos, na seqüência utilizada pelas bases, são: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, F, G, H, I, J, K, L, M, N, O, P, Q, R, S, T, U, V, W, X, Y e Z. Assim, a base 20 utiliza os algarismos de 0 a K para representar os números. Todos os números, independente da base utilizada, têm exatidão de 10 casas decimais.

Para indicar qual base um número está, deve-se utilizar a notação indicada na figura 3.14: 0xBxN, onde B é a base do número de N é o número.

• Conversão de um número de uma base qualquer para a base 10

Dado um número em uma base b qualquer, a sua representação na base 10 é dada pela expressão C.1, onde I é o número de algarismos da parte inteira do número, F o número de algarismos da parte fracionária e a_x o algarismo da posição x do número, sendo $x \geq 0$, a_x é da parte inteira do número e se $x<0$, ax é da parte fracionária.

$$N_{10} = a_{I-1}b^{I-1} + a_{I-2}b^{I-2} + \ldots + a_2b^2 + a_1b^1 + a_0b^0 + a_{-1}b^{-1} + a_{-2}b^{-2} + a_{-F-1}b^{-F-1} + a_{-F}b^{-F} \qquad \text{(C.1)}$$

Por exemplo, considerando o número 0x16x3F5.3A (isto é, 3F5,3A na base 16), a sua representação na base 10 será dada pela expressão $3\times16^2 + 15\times16^1 + 5\times16^0 + 3\times16^{-1} + 10\times16^{-2}$. Portando, 0x16x3F5.3A = 0x10x1013.2265625.

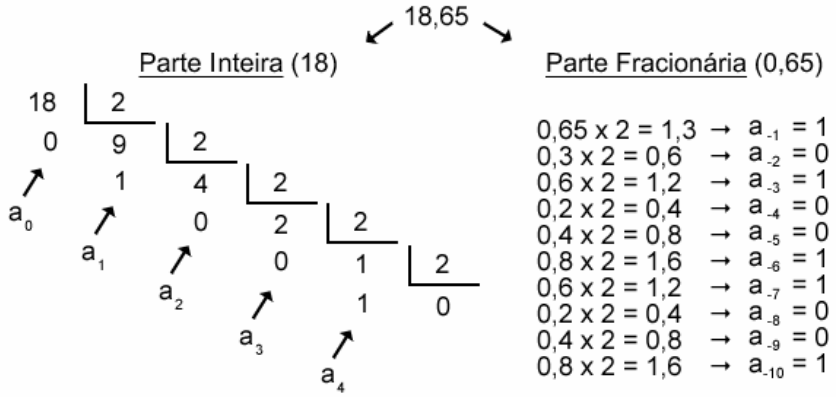
• Conversão de um número na base 10 para uma base qualquer

A conversão de um número na base 10 para outro número em uma base qualquer deve ser realizada dividindo o número na parte inteira e na parte fracionária.

A parte inteira do número deverá ser dividida pela base sucessivas vezes e o resto de cada divisão ocupará as posições de ordem 0, 1, 2 e assim por diante até o resto da última divisão, isto é, a divisão que resultar em um quociente igual zero. Este resto será o algarismo do número de maior ordem, ou seja, mais significativo no número.

Se o número tiver parte fracionária, o algoritmo para esta parte do número consta de uma série de multiplicações do número fracionário pela base. A parte inteira do resultado dessa multiplicação será o valor da primeira casa fracionária do número convertido e a parte fracionária será multiplicada pela base, e assim sucessivamente até a parte fracionária resultar em zero ou encontrarmos o número de casas decimais desejado. No caso deste Sistema, teremos 10 casas decimais como limite na execução do algoritmo.

O exemplo abaixo converte o número 18,65 na base 10 para a base 2.



Logo, 0x10x18.65 = 0x2x10010.1010011001.

Apêndice D

Constantes e funções incluídas no Sistema

As constantes incluídas no Sistema de Monitoramento de Ambientes estão listadas abaixo. Elas podem ser utilizadas em qualquer expressão escrita pelo usuário.

PI = Valor do pi (π = 3.141592653592)

E = Valor do número de Napier (e = 2.718281828459)

NINF = Infinito negativo

PINF = Infinito positivo

As funções apresentadas abaixo foram as implementadas no Sistema de Monitoramento de Ambientes. Outras funções podem (e devem) ser implementadas e adicionadas ao pacote **br.eng.rsalustiano.sensorfusion.expression.function** (ver detalhes no capítulo 3).

Nas funções declaradas abaixo, as siglas utilizadas para argumentos de entrada e retorno estão listadas abaixo:

NumElem = Elemento numérico (ex: 10.45, 0x16xAF.5D, 0.461)

StrElem = Elemento tipo seqüência alfanumérica (ex: “abacadabra”)

ImgElem = Elemento imagem (ex: um arquivo de imagem JPG)

AudElem = Elemento áudio (ex: um arquivo de som WAV)

BoolElem = Elemento *booleano* (true ou false)

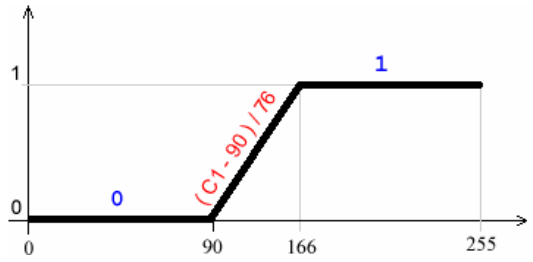
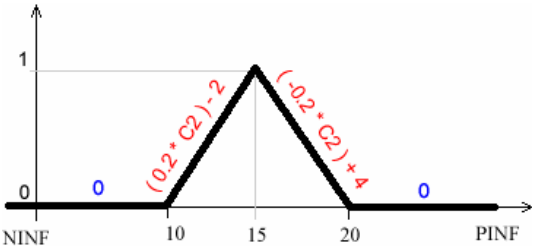
Elem = Qualquer tipo de elemento disponível no Sistema

• FUNÇÕES MATEMÁTICAS

ACOS	Utilização: Resultado: Descrição:	**ACOS**(*NumElem1*) *NumElemR* A função retorna um elemento numérico (*NumElemR*) resultado do arco cosseno de *NumElem1*.
ASIN	Utilização: Resultado: Descrição:	**ASIN**(*NumElem1*) *NumElemR* A função retorna um elemento numérico (*NumElemR*) resultado do arco seno de *NumElem1*.
ATAN	Utilização: Resultado: Descrição:	**ATAN**(*NumElem1*) *NumElemR* A função retorna um elemento numérico (*NumElemR*) resultado do arco tangente de *NumElem1*.
CHANGEBASE	Utilização: Resultado: Descrição:	**CHAGEBASE**(*NumElem1, NumElem2*) *NumElemR* A função retorna um elemento numérico (*NumElemR*) com o mesmo valor de *NumElem1*, mas na base indicada por *NumElem2*. O elemento numérico *NumElem2* deve ser um número inteiro entre 2 e 36.
COS	Utilização: Resultado: Descrição:	**COS**(*NumElem1*) *NumElemR* A função retorna um elemento numérico (*NumElemR*) resultado do cosseno de *NumElem1*.
EXP	Utilização: Resultado: Descrição:	**EXP**(*NumElem1*) *NumElemR* A função retorna um elemento numérico (*NumElemR*) com o valor exponencial do número *NumElem1*, isto é, *NumElemR* = $e^{NumElem1}$, onde e é o número de Napier.
LN	Utilização: Resultado: Descrição:	**LN**(*NumElem1*) *NumElemR* A função retorna um elemento numérico (*NumElemR*) com o valor do logaritmo neperiano do elemento numérico *NumElem1*, ou seja, logaritmo na base e (número de Napier).
LOG	Utilização: Resultado: Descrição:	**LOG**(*NumElem1, NumElem2*) *NumElemR* A função retorna um elemento numérico (*NumElemR*) com o valor do logaritmo do elemento numérico *NumElem1* na base *NumElem2*, isto é, *NumElemR* = $log_{NumElem2}(NumElem1)$.
POW	Utilização: Resultado: Descrição:	**POW**(*NumElem1, NumElem2*) *NumElemR* A função retorna um elemento numéric o (*NumElemR*) resultado do valor *NumElem1* elevado à potência *NumElem2*, isto é, *NumElemR* = $(NumElem1)^{NumElem2}$.
SIN	Utilização: Resultado: Descrição:	**SIN**(*NumElem1*) *NumElemR* A função retorna um elemento numérico (*NumElemR*) resultado do seno de *NumElem1*.
SQRT	Utilização: Resultado: Descrição:	**SQRT**(*NumElem1*) *NumElemR* A função retorna um elemento numérico (*NumElemR*) com a raiz quadrada de *NumElem1*, isto é, *NumElemR* = $\sqrt{NumElem1}$.

SUM	Utilização: Resultado: Descrição:	**SUM**(*NumElem1, NumElem2, ..., NumElemN*) *NumElemR* A função retorna um número *NumElemR* contendo a soma dos números *NumElem1, NumElem2, ..., NumElemN*, ou seja, *NumElemR = NumElem1 + NumElem2 + ... + NumElemN.*
TAN	Utilização: Resultado: Descrição:	**TAN**(*NumElem1*) *NumElemR* A função retorna um elemento numérico (*NumElemR*) resultado da tangente de *NumElem1*.

• FUNÇÕES LÓGICAS E ESTATÍSTICAS

IF	Utilização: Resultado: Descrição:	**IF**(*BoolElem1, Elem2, Elem3*) *ElemR* A função IF retorna o elemento *Elem2* se o valor da expressão *booleana BoolElem1* for verdadeira (true); se o valor da expressão *BoolElem1* for falsa (false), então a função retorna o elemento *Elem1*.
FUZZY	Utilização: Resultado: Descrição:	**FUZZY**(*NumElemV, NumElemP1, NumElemV12, NumElemP2, NumElemV23, NumElemP3, ..., NumElemP[N-1], NumElemV[N-1][N], NumElemP[N]*) *NumElemR* A função FUZZY implementa a lógica *Fuzzy*. O primeiro argumento da função (*NumElemV*) é a variável que está sendo verificada na lógica. A seqüência de elementos *NumElemP[N-1], NumElemV[N-1][N] e* NumElemP*[N]* indica os pontos extremos da parte da lógica *Fuzzy* (*NumElemP[N-1]* e *NumElemP[N]*) e o valor lógico *Fuzzy* do intervalo (*NumelemV*). O valor de retorno é número *NumElemR* contento o resultado da lógica, ou seja, um valor entre 0 e 1. Exemplos:  FUZZY(C1, 0, 0, 90, (C1-90)/76, 166, 1, 255)  FUZZY(C2, NINF, 0, 10, (0.2*C2)-2, 15, (-0.2*C2)+4, 20, 0, PINF)

MEAN	Utilização: Resultado: Descrição:	**MEAN**(*NumElem1, NumeElem2, ..., NumeElemN*) *NumElemR* A função retorna um número *NumElemR* contendo a Média Aritmética dos números *NumElem1, NumElem2, ..., NumElemN*, ou seja, *NumElemR = (NumElem1 + NumElem2 + ... + NumElemN)/N*, onde N é o número de elementos da chamada da função.

• FUNÇÕES RELACIONADAS A IMAGENS

GETRGB	Utilização: Resultado: Descrição:	**GETRGB**(*ImgElem1*, *NumElem2, NumElem3*) *SetElemR* A função GETRGB retorna um conjunto de três elementos numéricos (*SetElemR*) contendo os valores de 0 a 255 correspondentes ao vermelhor (R), verde (G) e azul (B), respectivamente, do *pixel* cujas coordenadas são *NumElem2 (x) e NumElem3 (y)* do elemento imagem *ImgElem1*.
GRAYSCALE	Utilização: Resultado: Descrição:	**GRAYSCALE**(*ImgElem1*) *ImgElemR* A função retorna a imagem *ImgElemR* contento a imagem *ImgElem1* com os pixels convertidos em escala de cinza.
IMGH	Utilização: Resultado: Descrição:	**IMGH**(*ImgElem1*) *NumElemR* A função retorna um número *NumElemR* contento a altura da imagem *ImgElemR.*
IMGW	Utilização: Resultado: Descrição:	**IMGH**(*ImgElem1*) *NumElemR* A função retorna um número *NumElemR* contento a largura da imagem *ImgElemR.*
COUNTPIXELS	Utilização: Resultado: Descrição:	**COUNTPIXELS**(*ImgElem1, SetElem1, SetElem2*) *NumElemR* A função retorna um número *NumElemR* contento o número de *pixels* com cor entre a cor definida pelos *NumElem* (conjunto RGB, nesta ordem) do conjunto *SetElem1* e a cor definida pelo conjunto *SetElem2* na imagem *ImgElem1*. Exemplo: COUNTPIXELS(LOADIMG(“img.jpg”), SET(0, 10, 0), SET(0, 255, 0)) O exemplo retorna a quantidade de *pixels* verdes, ou seja, *pixels* com a cor RGB entre (0,10,0) e (0,255,0).
GAMMAFILTER	Utilização: Resultado: Descrição:	**GAMMAFILTER**(*ImgElem1, NumElem1*) *ImgElemR* A função retorna uma imagem (*ImgElemR*) contendo o valor dos seus *pixels* modificados com aumento ou diminuição da luminosidade. O fator *gamma* é indicado pelo argumento *NumElem1*. Valores de *gamma* entre 0,0 e 1,0 reduzem a luminosidade da imagem e entre 1.0 e 3,0 (tipicamente), aumentam a luminosidade da imagem.

LOOKUPFILTER	Utilização: Resultado: Descrição:	**LOOKUPFILTER**(*ImgElem1, NumElem1*) *ImgElemR* A função retorna uma imagem (*ImgElemR*) contendo o valor do seus *pixels* coloridos com uma das três cores básicas (vermelho, verde ou azul). O algoritmo por traz da função cria uma imagem em preto e branco e em seguida altera o valor dos *pixels* de acordo com a cor selecionada. A seleção da cor é feita pelo argumento *NumElem1*: 0 para o vermelho, 1 para o verde e 2 para o azul.
LOADIMG	Utilização: Resultado: Descrição:	**LOADIMG**(*StrElem1*) *ImgElemR* A função LOADIMG carrega uma imagem externa ao Sistema. O retorno da função é *ImgElemR* com a imagem apontada pelo *link* da *string StrElem1*. Por exemplo: LOADIMG(“http://www.lpm.fee.unicamp.br/images/01.jpg”)

• FUNÇÕES RELACIONADAS A ÁUDIOS

LOADAUDIO	Utilização: Resultado: Descrição:	**LOADAUDIO**(*StrElem1*) *AudElemR* A função LOADAUDIO carrega um arquivo de áudio externo ao Sistema. O retorno da função é *AudElemR* com o áudio apontado pelo *link* da *string StrElem1*. Por exemplo: LOADIMG(“http://www.lpm.fee.unicamp.br/audios/01.wav”)
REVERSEAUDIO	Utilização: Resultado: Descrição:	**REVERSEAUDIO**(*AudElem1*) *AudElemR* A função retorna o elemento áudio *AudElemR* contendo o conteúdo do *AudElem1* invertido no tempo, ou seja, há uma inversão da ordem dos bytes do elemento áudio *AudElem1*.

• FUNÇÕES PARA OPERAÇÕES COM CONJUNTOS

GETSETINDEX	Utilização: Resultado: Descrição:	**GETSETINDEX**(*SetElem1*, *NumElem1*) *ElemR* A função retorna o elemento *ElemR* pertencente a posição *NumElem1* do conjunto de elementos *SetElem1*. O número *NumElem1* deve ser inteiro.
NSET	Utilização: Resultado: Descrição:	**NSET**(*SetElem1*) *NumElemR* A função retorna o elemento numérico *NumElemR* contendo o número de elementos do conjunto *SetElem1*.
SET	Utilização: Resultado: Descrição:	**SET**(*Elem1, Elem2, ..., ElemN*) *SetElemR* A função SET cria um conjunto (*SetElemR*) com os elementos de entrada *Elem1, Elem2, ..., ElemN*.

• FUNÇÕES PARA MANIPULAÇÃO E FORMATAÇÃO NUMÉRICA

ACCURACY	Utilização: Resultado: Descrição:	**ACCURACY**(*NumElem1*) *NumElemR* A função retorna um elemento numérico *NumElemR* contendo a exatidão do número *NumElem1*, isto é, dado um número 10,5±1,3, a função retorna o número 1,3.
MAXVAL	Utilização: Resultado: Descrição:	**MAXVAL**(*NumElem1*) *NumElemR* A função MAXVAL retorna um elemento numérico *NumElemR* contendo o valor máximo do número *NumElem1*, de acordo com a sua exatidão. Para o número 10,5±1,3, a função retorna o número 11,8.
MINVAL	Utilização: Resultado: Descrição:	**MINVAL**(*NumElem1*) *NumElemR* A função MINVAL retorna um elemento numérico *NumElemR* contendo o valor mínimo do número *NumElem1*, de acordo com a sua exatidão. Para o número 10,5±1,3, a função retorna o número 9,2.
NFORMAT	Utilização: Resultado: Descrição:	**NFORMAT** (*NumElem1, NumElem2, BoolElem1*) ou **NFORMAT** (*NumElem1, NumElem2*) *NumElemR* A função retorna o número *NumElem1*, formatando-o com *NumElem2* casas decimais e exibindo, ou não, a exatidão do número. Se o valor de *BoolElem1* for verdadeiro (true), a exatidão é exibida; caso contrário ela não será. Caso o número de casas decimais indicado por *NumElem2* for menor que o número de casas decimais do número *NumElem1*, o número do resultado será arredondado. Por exemplo: NFORMAT(12.5038, 2) = 12.504, e NFORMAT(3.7, 4) = 3.7000.
ROUND	Utilização: Resultado: Descrição:	**ROUND**(*NumElem1, NumElem2*) *NumElemR* A função retorna o valor de *NumElem1* arredondado de *NumElem2* casas decimais. Por exemplo, ROUND(10.58, 1) = 10.6.
TRUNC	Utilização: Resultado: Descrição:	**TRUNC**(*NumElem1, NumElem2*) *NumElemR* A função retorna o valor de *NumElem1* truncado de *NumElem2* casas decimais. Por exemplo, TRUNC(10.58, 1) = 10.5.
MINNUM	Utilização: Resultado: Descrição:	**MINNUM**(*NumElem1, NumElem2, ..., NumElemN*) *NumElemR* A função retorna o número *NumElemR* contendo o valor do menor número dos elementos *NumElem1, NumElem2, ...* e *NumElemN*.
MAXNUM	Utilização: Resultado: Descrição:	**MAXNUM**(*NumElem1, NumElem2, ..., NumElemN*) *NumElemR* A função retorna o número *NumElemR* contendo o valor do maior número dos elementos *NumElem1, NumElem2, ...* e *NumElemN*.

• ALGORITMOS DE SENSORES EM CONSENSO

CHAUCRIT	Utilização: Resultado: Descrição:	**CHAUCRIT**(*NumElem1, NumElem2, ..., NumElemN*) *NumElemR* Função que executa o algoritmo do Critério de Chauvenet nas entradas *NumElem1, NumElem2, ..., NumElemN*. O resultado é um número *NumElemR* contendo a Média Aritmética com os valores aceitos pelo critério. Ver detalhes do algoritmo no capítulo 4.
ACHAUCRIT	Utilização: Resultado: Descrição:	**ACHAUCRIT**(*NumElem1, NumElem2, ..., NumElemN*) *SetElemR* Função que executa o algoritmo do Critério de Chauvenet nas entradas *NumElem1, NumElem2, ..., NumElemN*. O resultado é o conjunto de valores (*SetElemR*) aceitos pelo critério. Ver detalhes do algoritmo no capítulo 4.
RCHAUCRIT	Utilização: Resultado: Descrição:	**RCHAUCRIT**(*NumElem1, NumElem2, ..., NumElemN*) *SetElemR* Função que executa o algoritmo do Critério de Chauvenet nas entradas *NumElem1, NumElem2, ..., NumElemN*. O resultado é o conjunto de valores (*SetElemR*) rejeitados pelo critério. Ver detalhes do algoritmo no capítulo 4.
APROXAGREE	Utilização: Resultado: Descrição:	**APROXAGREE**(*NumElem1, NumElem2, ..., NumElemN*) *NumElemR* Função que executa o algoritmo de Concordância Aproximada nas entradas *NumElem1, NumElem2, ..., NumElemN*. O resultado é um número *NumElemR* contendo a Média Aritmética com os valores aceitos pelo critério. Ver detalhes do algoritmo no capítulo 4.
AAPROXAGREE	Utilização: Resultado: Descrição:	**AAPROXAGREE**(*NumElem1, NumElem2, ..., NumElemN*) *SetElemR* Função que executa o algoritmo de Concordância Aproximada nas entradas *NumElem1, NumElem2, ..., NumElemN*. O resultado é o conjunto de valores (*SetElemR*) aceitos pelo critério. Ver detalhes do algoritmo no capítulo 4.
RAPROXAGREE	Utilização: Resultado: Descrição:	**RAPROXAGREE**(*NumElem1, NumElem2, ..., NumElemN*) *SetElemR* Função que executa o algoritmo de Concordância Aproximada nas entradas *NumElem1, NumElem2, ..., NumElemN*. O resultado é o conjunto de valores (*SetElemR*) rejeitados pelo critério. Ver detalhes do algoritmo no capítulo 4.
FCA	Utilização: Resultado: Descrição:	**FCA**(*NumElem1, NumElem2, ..., NumElemN*) *NumElemR* Função que executa o algoritmo de Concordância Inexada FCA nas entradas *NumElem1, NumElem2, ..., NumElemN*. O resultado é um número *NumElemR* contendo a Média Aritmética com os valores aceitos pelo critério. Ver detalhes do algoritmo no capítulo 4.
AFCA	Utilização: Resultado: Descrição:	**AFCA** (*NumElem1, NumElem2, ..., NumElemN*) *SetElemR* Função que executa o algoritmo de Concordância Inexada FCA nas entradas *NumElem1, NumElem2, ..., NumElemN*. O resultado é o conjunto de valores (*SetElemR*) aceitos pelo critério. Ver detalhes do algoritmo no capítulo 4.

RFCA	Utilização: Resultado: Descrição:	**RFCA**(*NumElem1, NumElem2, ..., NumElemN*) *SetElemR* Função que executa o algoritmo de Concordância Inexada FCA nas entradas *NumElem1, NumElem2, ..., NumElemN*. O resultado é o conjunto de valores (*SetElemR*) rejeitados pelo critério. Ver detalhes do algoritmo no capítulo 4.
CCA	Utilização: Resultado: Descrição:	**CCA**(*NumElem1, NumElem2, ..., NumElemN*) *NumElemR* Função que executa o algoritmo de Concordância Inexada CCA nas entradas *NumElem1, NumElem2, ..., NumElemN*. O resultado é um número *NumElemR* contendo a Média Aritmética com os valores aceitos pelo critério. Ver detalhes do algoritmo no capítulo 4.
ACCA	Utilização: Resultado: Descrição:	**ACCA** (*NumElem1, NumElem2, ..., NumElemN*) *SetElemR* Função que executa o algoritmo de Concordância Inexada CCA nas entradas *NumElem1, NumElem2, ..., NumElemN*. O resultado é o conjunto de valores (*SetElemR*) aceitos pelo critério. Ver detalhes do algoritmo no capítulo 4.
RCCA	Utilização: Resultado: Descrição:	**RCCA**(*NumElem1, NumElem2, ..., NumElemN*) *SetElemR* Função que executa o algoritmo de Concordância Inexada CCA nas entradas *NumElem1, NumElem2, ..., NumElemN*. O resultado é o conjunto de valores (*SetElemR*) rejeitados pelo critério. Ver detalhes do algoritmo no capítulo 4.
CONTRFUSION	Utilização: Resultado: Descrição:	**CONTRFUSION** (*NumElem1, NumElem2, ..., NumElemN*) *NumElemR* Função que executa o algoritmo de Fusão Contraditória nas entradas *NumElem1, NumElem2, ..., NumElemN*. O resultado é um número *NumElemR* contendo o resultado do algoritmo. Ver detalhes do algoritmo no capítulo 4.
HYBRID	Utilização: Resultado: Descrição:	**HYBRID**(*NumElem1, NumElem2, ..., NumElemN*) *NumElemR* Função que executa o algoritmo Híbrido nas entradas *NumElem1, NumElem2, ..., NumElemN*. O resultado é um número *NumElemR* contendo o resultado do algoritmo. Ver detalhes do algoritmo no capítulo 4.

• FUNÇÕES QUE REALIZAM TAREFAS

POST	Utilização: Resultado: Descrição:	**POST**(*StrElem1, StrElem2*) Envio de uma mensagem através do método POST do protocolo HTTP. A função realiza o envio da mensagem contida no elemento *StrElemt2* para o *socket* (IP+porta) indicado no elemento *StrElem1.* Exemplo: POST(“http://vonbraun.lpm.fee.unicamp.br:9877”, “<XMLSTS> Mensagem OK </XMLSTS>”)

SENDMAIL	Utilização:	**SENDMAIL**(*StrElem1, StrElem2, StrElem3, StrElem4, StrElem5, StrElem6, StrElem7*)
	Resultado:	Envio de uma mensagem eletrônica (e-mail).
	Descrição:	A função SENDMAIL realiza o envio de um e-mail com os seguintes parâmetros: *StrElem1* = campo destinatário (TO) do e-mail; *StrElem2* = campo cópia (CC) do e-mail; *StrElem3* = campo cópia oculta (BCC) do e-mail; *StrElem4* = campo assunto (*subject*) do e-mail; *StrElem5* = campo conteúdo (corpo)do e-mail; *StrElem6* = campo tipo do conteúdo do e-mail (text/plain ou text/html) *StrElem7* = campo dos arquivos anexados ao e-mail. Todos os elementos da chama da função são alfanuméricos (*strings*) e podem ser vazios, com exceção do elemento *StrElem1*. Exemplo: SENDMAIL(“rsalusti@lpm.fee.unicamp.br”, “”, “”, “Mensagem de exemplo”, “<HTML><BODY>Canal 1 = ” + C1 + ” Canal 2 em anexo</BODY></HTML>”, “text/html”, FILELNK(C2)).

• FUNÇÕES PARA MANIPULAÇÃO DE ARQUIVOS

FILELNK	Utilização:	**FILELNK**(*ImgElem1*) ou **FILELNK**(*AudElem1*)
	Resultado:	*StrElemR*
	Descrição:	Cria um arquivo imagem (*ImgElem1)* ou áudio (*AudElem1*) e o disponibiliza para acesso interno ao sistema através da pasta e arquivo indicado por *StrElemR*. É útil para se incluir anexos nos e-mails.
URLLNK	Utilização:	**URLLNK**(*ImgElem1*) ou **URLLNK**(*AudElem1*)
	Resultado:	*StrElemR*
	Descrição:	Cria um arquivo imagem (*ImgElem1)* ou áudio (*AudElem1*) e o disponibiliza para acesso pela Internet através da url *StrElemR*. É útil para ser utilizado com *link* de imagens e arquivos de áudio no corpo de e-mails.

Apêndice E

Especificações do sensor de temperatura LM35

As especificações do sensor de temperatura LM35 estão anexadas a seguir. Elas foram extraídas do endereço eletrônico da *National Instruments*, fabricante do componente, acessado em 22/11/2005: http://cache.national.com/ds/LM/LM35.pdf.

Apesar da descrição geral do componente indicar que a exatidão (*accuracy*) do LM35 varia de ¼°C a ¾°C, observa-se pelo gráfico da Exatidão *versus* a Temperatura (*Accuracy vs. Temperature*) da página 5 da folha de dados que a exatidão do componente LM35D, utilizado no experimento descrito no capítulo 5, varia até ±2°C, dependendo da temperatura.

National Semiconductor

November 2000

LM35
Precision Centigrade Temperature Sensors

General Description

The LM35 series are precision integrated-circuit temperature sensors, whose output voltage is linearly proportional to the Celsius (Centigrade) temperature. The LM35 thus has an advantage over linear temperature sensors calibrated in ° Kelvin, as the user is not required to subtract a large constant voltage from its output to obtain convenient Centigrade scaling. The LM35 does not require any external calibration or trimming to provide typical accuracies of ±¼°C at room temperature and ±¾°C over a full −55 to +150°C temperature range. Low cost is assured by trimming and calibration at the wafer level. The LM35's low output impedance, linear output, and precise inherent calibration make interfacing to readout or control circuitry especially easy. It can be used with single power supplies, or with plus and minus supplies. As it draws only 60 µA from its supply, it has very low self-heating, less than 0.1°C in still air. The LM35 is rated to operate over a −55° to +150°C temperature range, while the LM35C is rated for a −40° to +110°C range (−10° with improved accuracy). The LM35 series is available packaged in hermetic TO-46 transistor packages, while the LM35C, LM35CA, and LM35D are also available in the plastic TO-92 transistor package. The LM35D is also available in an 8-lead surface mount small outline package and a plastic TO-220 package.

Features

- Calibrated directly in ° Celsius (Centigrade)
- Linear + 10.0 mV/°C scale factor
- 0.5°C accuracy guaranteeable (at +25°C)
- Rated for full −55° to +150°C range
- Suitable for remote applications
- Low cost due to wafer-level trimming
- Operates from 4 to 30 volts
- Less than 60 µA current drain
- Low self-heating, 0.08°C in still air
- Nonlinearity only ±¼°C typical
- Low impedance output, 0.1 Ω for 1 mA load

Typical Applications

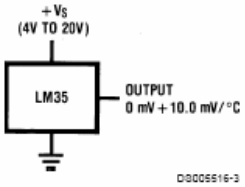
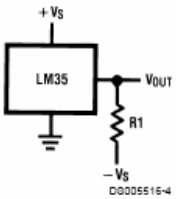


FIGURE 1. Basic Centigrade Temperature Sensor (+2°C to +150°C)



Choose $R_1 = -V_S/50\ \mu A$
V_{OUT} = +1,500 mV at +150°C
= +250 mV at +25°C
= −550 mV at −55°C

FIGURE 2. Full-Range Centigrade Temperature Sensor

Absolute Maximum Ratings (Note 10)

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/ Distributors for availability and specifications.

Supply Voltage	+35V to −0.2V
Output Voltage	+6V to −1.0V
Output Current	10 mA
Storage Temp.;	
TO-46 Package,	−60°C to +180°C
TO-92 Package,	−60°C to +150°C
SO-8 Package,	−65°C to +150°C
TO-220 Package,	−65°C to +150°C
Lead Temp.:	
TO-46 Package, (Soldering, 10 seconds)	300°C
TO-92 and TO-220 Package, (Soldering, 10 seconds)	260°C
SO Package (Note 12)	
Vapor Phase (60 seconds)	215°C
Infrared (15 seconds)	220°C
ESD Susceptibility (Note 11)	2500V
Specified Operating Temperature Range: T_{MIN} to T_{MAX} (Note 2)	
LM35, LM35A	−55°C to +150°C
LM35C, LM35CA	−40°C to +110°C
LM35D	0°C to +100°C

Electrical Characteristics

(Notes 1, 6)

Parameter	Conditions	LM35A			LM35CA			Units (Max.)
		Typical	Tested Limit (Note 4)	Design Limit (Note 5)	Typical	Tested Limit (Note 4)	Design Limit (Note 5)	
Accuracy	T_A=+25°C	±0.2	±0.5		±0.2	±0.5		°C
(Note 7)	T_A=−10°C	±0.3			±0.3		±1.0	°C
	T_A=T_{MAX}	±0.4	±1.0		±0.4	±1.0		°C
	T_A=T_{MIN}	±0.4	±1.0		±0.4		±1.5	°C
Nonlinearity	T_{MIN}≤T_A≤T_{MAX}	**±0.18**		**±0.35**	**±0.15**		**±0.3**	°C
(Note 8)								
Sensor Gain	T_{MIN}≤T_A≤T_{MAX}	**+10.0**	**+9.9, +10.1**		**+10.0**		**+9.9, +10.1**	mV/°C
(Average Slope)								
Load Regulation	T_A=+25°C	±0.4	±1.0		±0.4	±1.0		mV/mA
(Note 3) 0≤I_L≤1 mA	T_{MIN}≤T_A≤T_{MAX}	**±0.5**		**±3.0**	**±0.5**		**±3.0**	mV/mA
Line Regulation	T_A=+25°C	±0.01	±0.05		±0.01	±0.05		mV/V
(Note 3)	4V≤V_S≤30V	**±0.02**		**±0.1**	**±0.02**		**±0.1**	mV/V
Quiescent Current	V_S=+5V, +25°C	56	67		56	67		µA
(Note 9)	V_S=+5V	**105**		**131**	**91**		**114**	µA
	V_S=+30V, +25°C	56.2	68		56.2	68		µA
	V_S=+30V	**105.5**		**133**	**91.5**		**116**	µA
Change of	4V≤V_S≤30V, +25°C	0.2	1.0		0.2	1.0		µA
Quiescent Current	4V≤V_S≤30V	**0.5**		**2.0**	**0.5**		**2.0**	µA
(Note 3)								
Temperature Coefficient of Quiescent Current		**+0.39**		**+0.5**	**+0.39**		**+0.5**	µA/°C
Minimum Temperature for Rated Accuracy	In circuit of *Figure 1*, I_L=0	+1.5		+2.0	+1.5		+2.0	°C
Long Term Stability	T_J=T_{MAX}, for 1000 hours	±0.08			±0.08			°C

LM35

Electrical Characteristics

(Notes 1, 6)

Parameter	Conditions	LM35			LM35C, LM35D			Units (Max.)
		Typical	Tested Limit (Note 4)	Design Limit (Note 5)	Typical	Tested Limit (Note 4)	Design Limit (Note 5)	
Accuracy,	T_A=+25°C	±0.4	±1.0		±0.4	±1.0		°C
LM35, LM35C	T_A=−10°C	±0.5			±0.5		±1.5	°C
(Note 7)	T_A=T_{MAX}	±0.8	±1.5		±0.8		±1.5	°C
	T_A=T_{MIN}	±0.8		±1.5	±0.8		±2.0	°C
Accuracy, LM35D	T_A=+25°C				±0.6	±1.5		°C
(Note 7)	T_A=T_{MAX}				±0.9		±2.0	°C
	T_A=T_{MIN}				±0.9		±2.0	°C
Nonlinearity (Note 8)	T_{MIN}≤T_A≤T_{MAX}	**±0.3**		**±0.5**	**±0.2**		**±0.5**	°C
Sensor Gain	T_{MIN}≤T_A≤T_{MAX}	**+10.0**	**+9.8, +10.2**		**+10.0**		**+9.8, +10.2**	mV/°C
(Average Slope)								
Load Regulation	T_A=+25°C	±0.4	±2.0		±0.4	±2.0		mV/mA
(Note 3) 0≤I_L≤1 mA	T_{MIN}≤T_A≤T_{MAX}	**±0.5**		**±5.0**	**±0.5**		**±5.0**	mV/mA
Line Regulation	T_A=+25°C	±0.01	±0.1		±0.01	±0.1		mV/V
(Note 3)	4V≤V_S≤30V	**±0.02**		**±0.2**	**±0.02**		**±0.2**	mV/V
Quiescent Current	V_S=+5V, +25°C	56	80		56	80		µA
(Note 9)	V_S=+5V	**105**		**158**	**91**		**138**	µA
	V_S=+30V, +25°C	56.2	82		56.2	82		µA
	V_S=+30V	**105.5**		**161**	**91.5**		**141**	µA
Change of	4V≤V_S≤30V, +25°C	0.2	2.0		0.2	2.0		µA
Quiescent Current	4V≤V_S≤30V	**0.5**		**3.0**	**0.5**		**3.0**	µA
(Note 3)								
Temperature Coefficient of Quiescent Current		**+0.39**		**+0.7**	**+0.39**		**+0.7**	µA/°C
Minimum Temperature for Rated Accuracy	In circuit of *Figure 1*, I_L=0	+1.5		+2.0	+1.5		+2.0	°C
Long Term Stability	T_J=T_{MAX}, for 1000 hours	±0.08			±0.08			°C

Note 1: Unless otherwise noted, these specifications apply: −55°C≤T_J≤+150°C for the LM35 and LM35A; −40°≤T_J≤+110°C for the LM35C and LM35CA; and 0°≤T_J≤+100°C for the LM35D. V_S=+5Vdc and I_{LOAD}=50 µA, in the circuit of *Figure 2*. These specifications also apply from +2°C to T_{MAX} in the circuit of *Figure 1*. Specifications in **boldface** apply over the full rated temperature range.

Note 2: Thermal resistance of the TO-46 package is 400°C/W, junction to ambient, and 24°C/W junction to case. Thermal resistance of the TO-92 package is 180°C/W junction to ambient. Thermal resistance of the small outline molded package is 220°C/W junction to ambient. Thermal resistance of the TO-220 package is 90°C/W junction to ambient. For additional thermal resistance information see table in the Applications section.

Note 3: Regulation is measured at constant junction temperature, using pulse testing with a low duty cycle. Changes in output due to heating effects can be computed by multiplying the internal dissipation by the thermal resistance.

Note 4: Tested Limits are guaranteed and 100% tested in production.

Note 5: Design Limits are guaranteed (but not 100% production tested) over the indicated temperature and supply voltage ranges. These limits are not used to calculate outgoing quality levels.

Note 6: Specifications in **boldface** apply over the full rated temperature range.

Note 7: Accuracy is defined as the error between the output voltage and 10mv/°C times the device's case temperature, at specified conditions of voltage, current, and temperature (expressed in °C).

Note 8: Nonlinearity is defined as the deviation of the output-voltage-versus-temperature curve from the best-fit straight line, over the device's rated temperature range.

Note 9: Quiescent current is defined in the circuit of *Figure 1*.

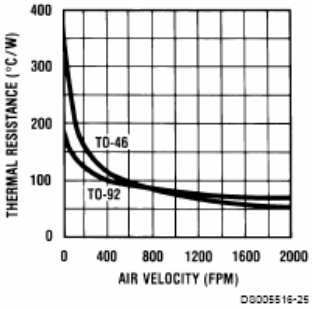
Note 10: Absolute Maximum Ratings indicate limits beyond which damage to the device may occur. DC and AC electrical specifications do not apply when operating the device beyond its rated operating conditions. See Note 1.

Note 11: Human body model, 100 pF discharged through a 1.5 kΩ resistor.

Note 12: See AN-450 "Surface Mounting Methods and Their Effect on Product Reliability" or the section titled "Surface Mount" found in a current National Semiconductor Linear Data Book for other methods of soldering surface mount devices.

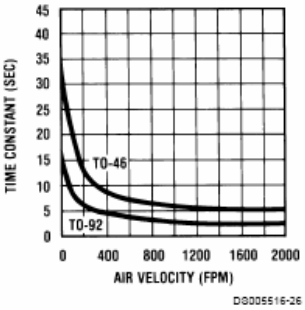
Typical Performance Characteristics

Thermal Resistance
Junction to Air



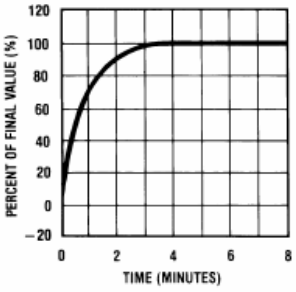
DS005516-25

Thermal Time Constant



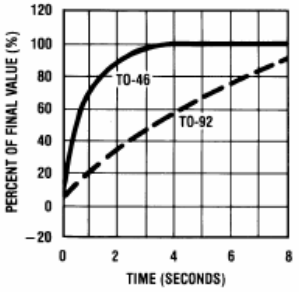
DS005516-26

Thermal Response
in Still Air



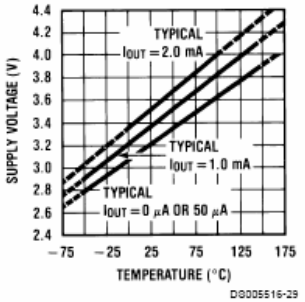
DS005516-27

Thermal Response in
Stirred Oil Bath



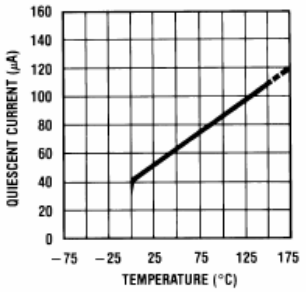
DS005516-28

Minimum Supply
Voltage vs. Temperature



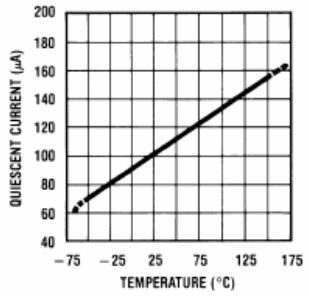
DS005516-29

Quiescent Current
vs. Temperature
(In Circuit of *Figure 1*.)



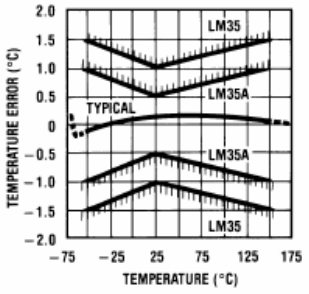
DS005516-30

Quiescent Current
vs. Temperature
(In Circuit of *Figure 2*.)



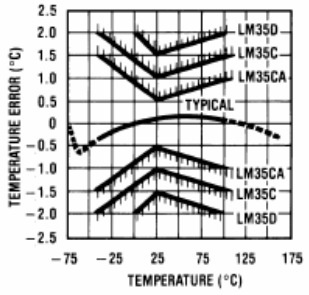
DS005516-31

Accuracy vs. Temperature
(Guaranteed)



DS005516-32

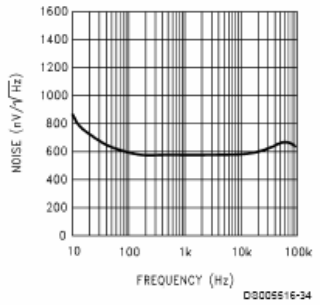
Accuracy vs. Temperature
(Guaranteed)



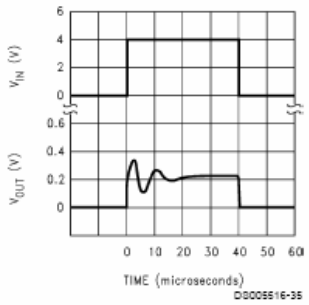
DS005516-33

Typical Performance Characteristics (Continued)

Noise Voltage



Start-Up Response



Applications

The LM35 can be applied easily in the same way as other integrated-circuit temperature sensors. It can be glued or cemented to a surface and its temperature will be within about 0.01°C of the surface temperature.

This presumes that the ambient air temperature is almost the same as the surface temperature; if the air temperature were much higher or lower than the surface temperature, the actual temperature of the LM35 die would be at an intermediate temperature between the surface temperature and the air temperature. This is expecially true for the TO-92 plastic package, where the copper leads are the principal thermal path to carry heat into the device, so its temperature might be closer to the air temperature than to the surface temperature.

To minimize this problem, be sure that the wiring to the LM35, as it leaves the device, is held at the same temperature as the surface of interest. The easiest way to do this is to cover up these wires with a bead of epoxy which will insure that the leads and wires are all at the same temperature as the surface, and that the LM35 die's temperature will not be affected by the air temperature.

The TO-46 metal package can also be soldered to a metal surface or pipe without damage. Of course, in that case the V− terminal of the circuit will be grounded to that metal. Alternatively, the LM35 can be mounted inside a sealed-end metal tube, and can then be dipped into a bath or screwed into a threaded hole in a tank. As with any IC, the LM35 and accompanying wiring and circuits must be kept insulated and dry, to avoid leakage and corrosion. This is especially true if the circuit may operate at cold temperatures where condensation can occur. Printed-circuit coatings and varnishes such as Humiseal and epoxy paints or dips are often used to insure that moisture cannot corrode the LM35 or its connections.

These devices are sometimes soldered to a small light-weight heat fin, to decrease the thermal time constant and speed up the response in slowly-moving air. On the other hand, a small thermal mass may be added to the sensor, to give the steadiest reading despite small deviations in the air temperature.

Temperature Rise of LM35 Due To Self-heating (Thermal Resistance, θ_{JA})

	TO-46, no heat sink	TO-46*, small heat fin	TO-92, no heat sink	TO-92**, small heat fin	SO-8 no heat sink	SO-8** small heat fin	TO-220 no heat sink
Still air	400°C/W	100°C/W	180°C/W	140°C/W	220°C/W	110°C/W	90°C/W
Moving air	100°C/W	40°C/W	90°C/W	70°C/W	105°C/W	90°C/W	26°C/W
Still oil	100°C/W	40°C/W	90°C/W	70°C/W			
Stirred oil	50°C/W	30°C/W	45°C/W	40°C/W			
(Clamped to metal, Infinite heat sink)	(24°C/W)				(55°C/W)		

*Wakefield type 201, or 1" disc of 0.020" sheet brass, soldered to case, or similar.
**TO-92 and SO-8 packages glued and leads soldered to 1" square of 1/16" printed circuit board with 2 oz. foil or similar.