

UNIVERSIDADE ESTADUAL DE PONTA GROSSA
DEPARTAMENTO DE INFORMÁTICA
MESTRADO EM COMPUTAÇÃO APLICADA NA AGRICULTURA

ANDERSON ESTEVAM DA ROSA

DISSERTAÇÃO
"SISTEMA DE NAVEGAÇÃO POR GEORREFERENCIAMENTO COM CORREÇÃO
DE ROTA POR LÓGICA FUZZY APLICADO À ROBÓTICA AGRÍCOLA"

PONTA GROSSA
2012

ANDERSON ESTEVAM DA ROSA

DISSERTAÇÃO

"SISTEMA DE NAVEGAÇÃO POR GEORREFERENCIAMENTO COM CORREÇÃO
DE ROTA POR LÓGICA FUZZY APLICADO À ROBÓTICA AGRÍCOLA"

Dissertação apresentada ao Programa de Pós-Graduação em Computação Aplicada da Universidade Estadual de Ponta Grossa, como requisito para obtenção do título de Mestre.

Orientação: Prof. Dr. Ariangelo Hauer Dias

PONTA GROSSA

2012

Ficha Catalográfica Elaborada pelo Setor Tratamento da Informação Belém/UEPG

R788s Rosa, Anderson Estevam da
Sistemas de navegação por georreferenciamento com correção de rota por lógica Fuzzy aplicado à robótica agrícola / Anderson Estevam da Rosa. Ponta Grossa, 2012.
82 f.

Dissertação (Mestrado em Computação Aplicada), Universidade Estadual de Ponta Grossa.
Orientador: Prof. Dr. Ariangelo Hauer Dias.

1. Lógica Fuzzy. 2. Inteligência artificial. 3. Sistema computacional. 4. Sistema embarcado. I. Dias, Ariangelo Hauer. II. Universidade Estadual de Ponta Grossa. Mestrado em Computação Aplicada. III. T.

CDD: 006.3

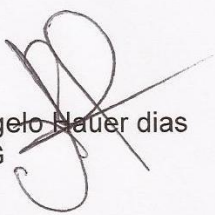
TERMO DE APROVAÇÃO

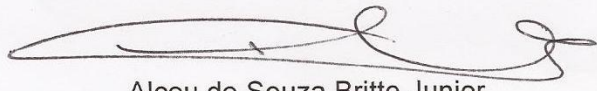
ANDERSON ESTEVAM DA ROSA

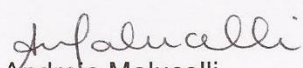
"SISTEMA DE NAVEGAÇÃO POR GEORREFERENCIAMENTO COM CORREÇÃO DE ROTA POR LÓGICA FUZZY APLICADO À ROBÓTICA AGRÍCOLA"

Dissertação aprovada como requisito parcial para obtenção do grau de Mestre no Programa de Pós-Graduação em Computação Aplicada da Universidade Estadual de Ponta Grossa, pela seguinte banca examinadora.

Orientador:


Ariangelo Hauser dias
UEPG


Alceu de Souza Britto Junior
UEPG


Andreia Malucelli
PUC-PR

Ponta Grossa, 23 de Julho de 2012.

AGRADECIMENTOS

Agradeço a Deus em primeiro lugar por ter me mantido de pé, pelas oportunidades e pelas pessoas que Ele colocou em meu caminho.

Aos meus pais Jamil Estevam da Rosa e Vera Regina da Rosa pelo apoio incondicional, pelo investimento em meu futuro, educação e por sempre acreditar em mim.

A minha irmã Audrey Cristine da Rosa, minha fonte de inspiração e exemplo de como lidar com as dificuldades e aos meus sobrinhos que contribuem em deixar minha vida mais leve.

Agradeço também a todas as pessoas que diretamente ou indiretamente contribuíram para o êxito desta pesquisa. Principalmente ao professor Dr. Ariangelo Hauer Dias, pela paciência, atenção e por toda a ajuda dispensada na orientação durante a elaboração desta dissertação.

Por fim, agradeço todos meus amigos e colegas de trabalho, em especial a Andressa Maria de Souza Derbis, Elvira Piegat do Amaral e principalmente ao Murilo Felipe da Silva pelo grande apoio e suporte emocional que foi fundamental durante todo o decorrer do mestrado.

RESUMO

A agricultura de precisão ainda é um tema considerado novo no Brasil, até mesmo pela Empresa Brasileira de Pesquisa Agropecuária (Embrapa), que em 2012 lançou um livro sobre tema e teve mais de 4500 acessos em menos de um mês de publicação. Isso demonstra que esta é uma área promissora, de interesse geral e pode ser o diferencial para que cada vez mais a agricultura no Brasil seja competitiva sem deixar de ser sustentável. Sendo assim, a substituição do processo manual por um processo automatizado e georreferenciado faz com que a coleta de dados no campo se torne mais rápida, padronizada e precisa. Pode-se observar também que o uso de dispositivos móveis talvez seja uma das chaves para o futuro, uma vez que são aparelhos de considerável capacidade computacional, relativo baixo custo e, obviamente, portáteis. Desta forma, o presente trabalho tem por objetivo desenvolver um sistema computacional para computadores embarcados ou celulares, de marcação de pontos e rotas georreferenciadas e correção da rota utilizando Lógica Fuzzy. A correção da rota se faz necessária devido aos possíveis desvios, causados por elementos comuns no campo, como pedras, buracos e vegetação. Este sistema será utilizado pelo robô de campo que está sendo desenvolvido no Laboratório de Automação e Robótica da Universidade Estadual de Ponta Grossa-PR, o projeto AGROBOT. A validação do sistema computacional foi feita manualmente, ou seja, ainda sem o robô, pelo autor do trabalho, em trajetos realizados a pé com o computador embarcado em mãos. Durante o percurso do trajeto foram feitas coletas de dados para estudo dos resultados, fazendo um comparativo entre a correção obtida pela Lógica Fuzzy e a correção matemática dada pela subtração do ângulo inicial e atual do dispositivo com o ponto alvo. Como resultado foi demonstrado graficamente que o uso da lógica Fuzzy mostrou-se mais eficaz em comparação com a simples subtração matemática, principalmente em resposta ao erro de localização dos dispositivos GPS.

Palavras-Chave: lógica Fuzzy, inteligência artificial, sistema computacional, sistema embarcado, georreferenciamento.

SUMMARY

Precision agriculture is still considered a new topic in Brazil, even by the Brazilian Agricultural Research Corporation (Embrapa), which, in 2012, published a book on the subject and had more than 4.5 million hits in less than a month of its publication. It demonstrates that this is a promising area of interest and can be the difference for Brazil's agriculture be increasingly competitive still in a sustainable way. Thus, the replacement of the manual for an automated process and georeferenced data collection in the field becomes more rapid, standardized, and accurate. One can also observe that the use of mobile devices is perhaps one of the keys to the future, since they are of considerable computational capacity on low cost, and obviously portable. Therefore, this study aims to develop a software for embedded computers or phones, with georeferenced points and routes and route correction using Fuzzy Logic. The correction of the route is necessary due to possible deviations caused by common elements in the field, such as rocks, holes and vegetation. This system will be used by the robot field being developed in the Laboratory of Automation and Robotics at the State University of Ponta Grossa, PR, the AGROBOT. The validation of the computer system was done manually, ie, yet without the robot, the author of this work, performed on foot paths with embedded computer in hand. During the course, the data were collected to study the results, making a comparison between the correction obtained by Fuzzy Logic and mathematics correction given by subtracting the initial and current angle of the device to the target point. As a result it was graphically demonstrated that the use of fuzzy logic has proved more effective in comparison with the simple subtraction mathematics, mainly in response to the error location GPS devices

Key-words: fuzzy logic, artificial intelligence, computer system, embedded systems, georeferencing

LISTA DE ILUSTRAÇÕES

Figura 1 - Marcação de rotas dos dispositivos de navegação GPS	13
Figura 2 - Marcação de rota em uma área agrícola (campo), sem ruas pré definidas	14
Figura 3 - Exemplos de hardware	16
Figura 4 - Arquitetura Android	23
Figura 5 - Estrutura Controlador Fuzzy	26
Figura 6 - Variáveis linguísticas e valores numéricos	28
Figura 7 - Grau de pertinência para febre de 36,6 °C	29
Figura 8 - (a) União entre conjuntos Fuzzy (b) Interseção entre conjuntos Fuzzy (c) Complemento de um conjunto Fuzzy	30
Figura 9 - Exemplo do método de centro de áreas	34
Figura 10 - Exemplo aritmético Defuzzificação	35
Figura 11 - Posicionamento Absoluto.....	36
Figura 12 - Posicionamento Relativo.....	37
Figura 13 - Estrutura básica para sistemas de navegação	39
Figura 14 - Fotografia do Friendly ARM Mini 6410.....	43
Figura 15 - Receptor GPS Gstar GS-87 Fonte: o autor.....	44
Figura 16 - Exemplo da versão Android SDK and AVD para Eclipse	47
Figura 17 - Máquina Android Emulada em Ambiente Windows Fonte: o autor	48
Figura 18 - Obtenção do MD5 Fingerprint para uso dos serviços do Google Maps ..	50
Figura 19 - Página para registro da chave para utilização do Google Maps	50
Figura 20 - O Google Maps, em um dispositivo Android com dois pontos marcados	51
Figura 21 - Tela onde o usuário vê onde está (ponto azul) e marca onde quer chegar (estrela)	54
Figura 22 - Construção do primeiro triângulo retângulo	56
Figura 23 - Termos linguísticos para X.....	56
Figura 24 - Termos linguísticos para Y.....	57
Figura 25 - Gráfico dos termos linguísticos da Fuzzy em X (cateto adjacente).....	57
Figura 26 - Gráfico dos termos linguísticos da Fuzzy em Y (cateto oposto)	58
Figura 27 - Desvio de rota 1	58
Figura 28 - Desvio de rota 2	59
Figura 29 - Gráfico dos termos linguísticos de saída da Fuzzy	60

Figura 30 - Rota seguida na primeira coleta de dados	63
Figura 31 - Capturas de telas: demonstração no mapa do lugar onde foi realizada a primeira coleta e os resultados obtidos pelo Navigo durante o percurso	63
Figura 32 - Gráfico das correções dadas pela Fuzzy por tempo	64
Figura 33 - Gráfico das correções dadas por subtração matemática	65
Figura 34 - Gráfico comparativo entre a correção da rota por Lógica Fuzzy e por subtração matemática	65
Figura 35 - Rota seguida na segunda coleta de dados	66
Figura 36 - Gráfico de comparação entre a correção por lógica fuzzy e por subtração matemática dos ângulos na segunda coleta de dados	67
Figura 37 - Capturas de telas e o local onde foram realizadas (Coleta 2).....	68
Figura 38 - Rota seguida na terceira coleta de dados	70
Figura 39 - Gráfico de comparação entre a lógica fuzzy e subtração matemática na terceira coleta de dados	71
 Quadro 1 - Termos nas telecomunicações.....	27
Quadro 2 - Operadores para lógica Fuzzy	30
Quadro 3 - Exemplo de mapa de regras	32
Quadro 4 - Regras usadas para Fuzzy	59

LISTA DE TABELAS

Tabela 1 - Tabela verdade Fuzzy	31
Tabela 2 - Tabela de valores para cada termo linguístico	60
Tabela 3 - Tabela com o formato do arquivo texto gerado para validação dos resultados	61
Tabela 4 - Leituras realizadas com o dispositivo parado.....	72

LISTA DE SIGLAS

ARM	<i>Advanced Risc Machine</i>
ADT	<i>Android Development Tools</i>
AVD	<i>Android Virtual Device</i>
CDA	Centro De Área
Cos	Cosseno
CPU	<i>Central Processing Unit</i>
DDR	<i>Double-Data-Rate</i>
Gb	<i>Gigabyte</i>
GPS	<i>Global Positioning System</i>
GSM	<i>Groupe Special Mobile</i>
IDE	<i>Integrated Development Environment</i>
JVM	<i>Java Virtual Machine</i>
Lat	Latitude
LCD	<i>Liquid Crystal Display</i>
Ln	Logaritmo Natural
Lon	Longitude
MAX	Crítério Dos Máximos
MDM	Média Dos Máximos
MMU	<i>Memory Management Unit</i>
Sen	Seno
SMS	<i>Short Message Service</i>
Tan	Tangente
wiFi	<i>Wireless Fidelity</i>
XML	<i>Extensible Markup Language</i>

SUMÁRIO

1	INTRODUÇÃO	12
1.1	OBJETIVO GERAL	14
1.2	OBJETIVOS ESPECÍFICOS.....	15
2	REVISÃO DE LITERATURA	16
2.1	HARDWARE LIVRE.....	16
2.2	ARQUITETURA RISC.....	17
2.3	ADVANCED RISC MACHINES - ARM	18
2.4	SOFTWARE LIVRE	19
2.5	ANDROID	20
2.5.1	Arquitetura Android	22
2.6	LÓGICA FUZZY.....	24
2.6.1	Controlador Fuzzy	25
2.6.2	Processo de Fuzzyficação	26
2.6.3	Operações entre conjuntos Fuzzy	29
2.6.4	Base de conhecimento	31
2.6.5	Defuzzificação	33
2.7	GEOREFERENCIAMENTO E GPS.....	35
2.8	SISTEMA DE NAVEGAÇÃO	37
2.9	ROBÓTICA AGRÍCOLA.....	39
2.10	TRABALHOS RELACIONADOS.....	40
3	METODOLOGIA	42
3.1	MATERIAL.....	42
3.1.1	Biblioteca GeoUtils	44
3.1.2	Biblioteca Fuzzy Engine.....	46

3.1.3 Eclipse e Android	47
3.1.4 Biblioteca Google Maps	48
3.2 MÉTODOS.....	52
4 RESULTADOS E DISCUSSÕES.....	54
4.1 SISTEMA COMPUTACIONAL NAVIGO	54
4.2 RESULTADOS E AVALIAÇÃO DO APLICATIVO	61
4.2.1 Primeira coleta de resultados	62
4.2.2 Segunda coleta de resultados	66
4.2.3 Terceira coleta de resultados.....	69
5 CONCLUSÕES.....	73
REFERÊNCIAS BIBLIOGRÁFICAS	74
GLOSSÁRIO.....	80

1 INTRODUÇÃO

O georreferenciamento está incorporando-se no cotidiano de diversas maneiras, seja para o mapeamento de percursos como, por exemplo, em sistemas embarcados de veículos, ou para delimitação e descrição de imóveis, como no censo rural já realizado pelo governo. Dentro da área agrícola, georreferenciamento vem como uma ferramenta de suporte para a agricultura de precisão, com o papel de auxiliar no monitoramento e avaliação das condições climáticas, dos solos ou das plantas, e para que isso se torne eficiente é necessário saber a exata localização do material utilizado para a amostragem em questão.

O georreferenciamento pode ser um dos insumos para a Agricultura de Precisão, tópico de relevância para que a agricultura brasileira ganhe um diferencial competitivo diante da globalização e ainda sem deixar de ser sustentável.

O Laboratório de Automação e Robótica da Universidade Estadual de Ponta Grossa-PR está desenvolvendo o projeto AGROBOT. Esse projeto visa à criação de um robô agrícola (*Field Robot*) que percorra talhões no campo com o intuito de coletar informações climáticas, amostras ou dados de solo e vegetação. Essas coletas serão feitas de maneira georreferenciada, para que se obtenha o posicionamento global do robô no momento exato de cada coleta.

O projeto AGROBOT se valerá de uma arquitetura, tanto em hardware como software, baseada em um modelo *OpenSource*, ou seja, com o código aberto para que a comunidade usuária possa usar, contribuir e customizar de acordo com as necessidades específicas.

Dentro desse contexto o presente trabalho traz o desenvolvimento de um sistema computacional para dispositivos móveis ou embarcados, onde seja possível e intuitivo ao usuário marcar pontos georreferenciados em um mapa virtual. Ainda, esse sistema computacional se encarregará de apontar correções de rotas dentro de um talhão, para eventuais desvios da rota início que o robô de campo venha a sofrer, devido a impasses

no caminho, como pedras, troncos e vegetação. Para isso é usada a lógica Fuzzy, a fim de corrigir os desvios de rota ocorridos, evitando assim que o robô ande em uma sequência de segmentos lineares alternados quanto à direção, ou seja, em ziguezague, e sim faça um percurso mais suavizado. Ainda, o sistema desenvolvido será, em futuros trabalhos, utilizado pelo robô agrícola do Projeto Agrobot.

Diferentemente dos dispositivos de navegação com GPS comerciais, onde os sistemas computacionais baseiam-se em um banco de dados com o mapeamento das ruas para correção da rota previamente salvo nos dispositivos ou em uma base na internet, o sistema proposto é focado para o campo, em um domínio rural. Isso quer dizer que o sistema de navegação tratado neste trabalho foi desenvolvido para aceitar rotas aleatórias, pois não se guiará por ruas ou estradas e sim, estará solto em um talhão da propriedade agrícola.

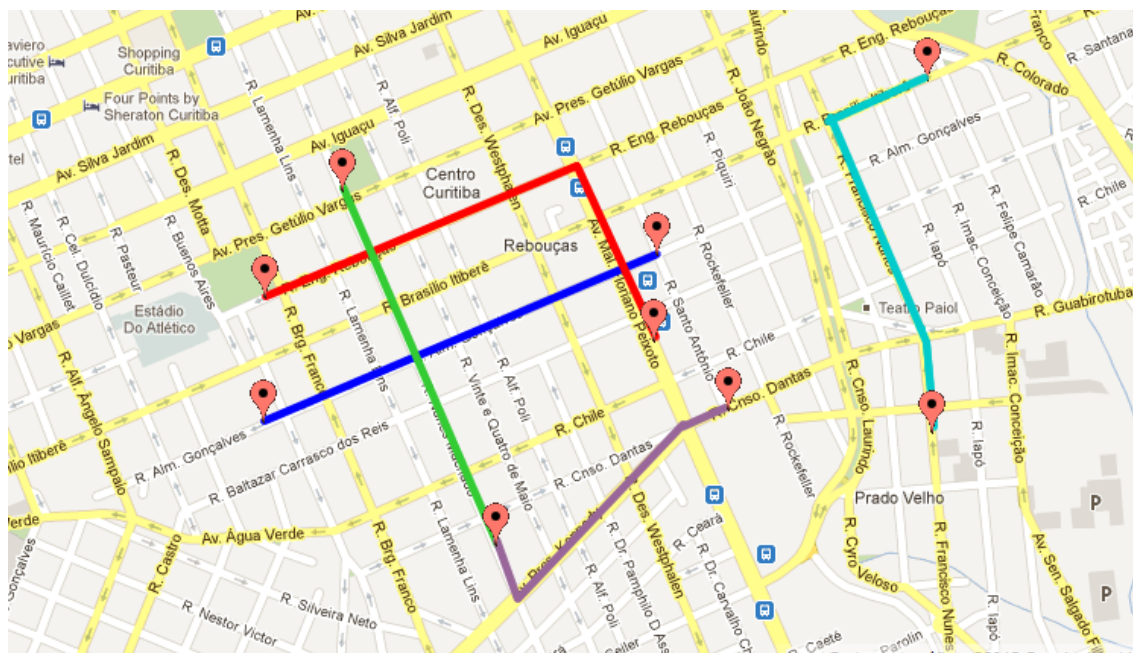


Figura 1 - Marcação de rotas dos dispositivos de navegação GPS
Fonte: o autor

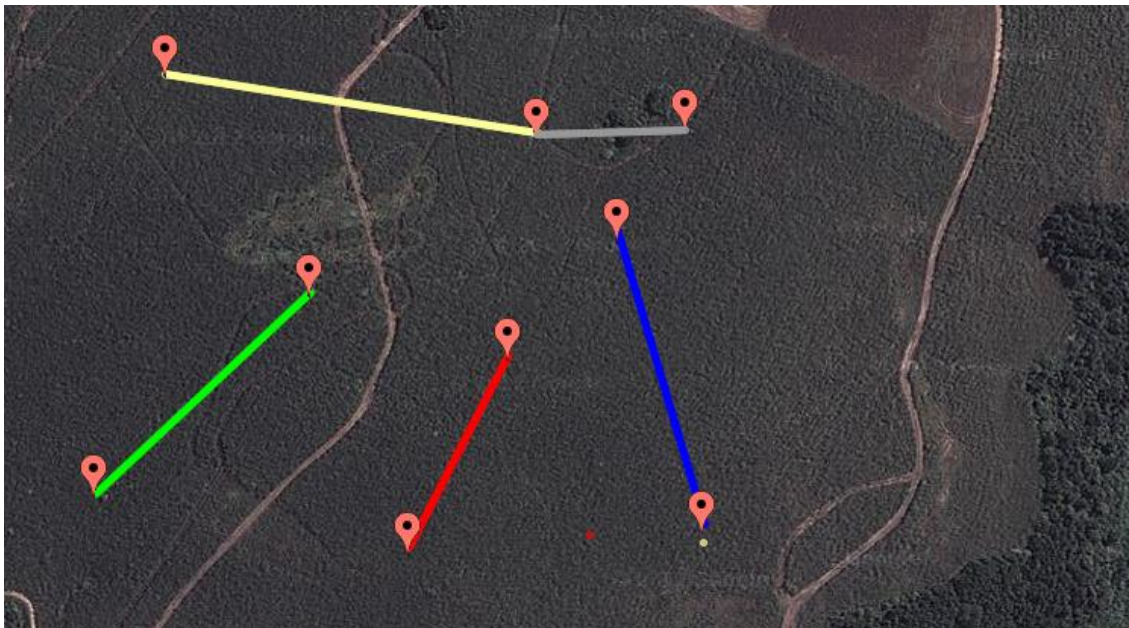


Figura 2 - Marcação de rota em uma área agrícola (campo), sem ruas pré definidas

Fonte: o autor

A Figura 1 e Figura 2 demonstram respectivamente dois sistemas de navegações. O primeiro, é baseado em mapas com ruas pré delimitadas e o segundo é a adaptação para uma aplicação rural, onde não se usa ruas e as rotas podem ser feitas de maneira arbitrária, pelo usuário especialista. No caso agrícola a adequação do posicionamento não se dará por dados pré cadastrados de caminhos fixos (ruas) como na maioria dos softwares comerciais. No sistema computacional, chamado Navigo, desenvolvido neste trabalho, essa adequação se dará pelo uso da lógica Fuzzy a fim de que o robô de campo siga o caminho marcado pelo usuário, mesmo havendo desvios de percurso.

1.1 OBJETIVO GERAL

O presente trabalho tem como objetivo a criação de um sistema computacional para computadores embarcados ou celulares, de marcação de pontos e rotas georreferenciadas com correção de rota por lógica Fuzzy.

1.2 OBJETIVOS ESPECÍFICOS

- Desenvolver um sistema para computadores embarcados ou aparelhos celulares
- Criar uma interface gráfica no sistema para traçar uma rota georreferenciada apenas com o toque do usuário
- Fazer a modelagem, criação de variáveis linguísticas e mapa de regras para a aplicação da Lógica Fuzzy para a correção da rota
- Utilizar o sistema desenvolvido para correção do desvios de percursos, simulando a pé uma possível rota traçada para o robô de campo.
- Avaliar o desempenho da correção dada pela lógica Fuzzy com a correção matemática dos ângulos: inicial e atual.

2 REVISÃO DE LITERATURA

Neste capítulo será apresentada uma revisão bibliográfica dos tópicos relevantes que foram utilizados para alcançar os objetivos da presente pesquisa. Os tópicos sobre *hardware* livre, arquitetura Risc, Advanced Risc Machine - ARM, *software* livre e Android também tem como objetivo contextualizar o leitor que pode não ter conhecimento sobre a tecnologia ou termo e justificar o seu uso no projeto, mostrando o modo de funcionamento e alguns de seus benefícios. Os tópicos de Lógica Fuzzy, Sistema de Navegação, Georreferenciamento e GPS serviram de alicerce fundamental para o desenvolvimento do projeto.

2.1 HARDWARE LIVRE

Maziero (2006) define *hardware* como um termo geral para equipamentos compostos por circuitos eletrônicos (processador, memória, portas de entrada/saída, etc.) e periféricos eletro-óptico-mecânicos (teclados, mouses, discos rígidos, unidades de disquete, CD ou DVD, dispositivos USB, etc.). A Figura 3 ilustra alguns exemplos de dispositivos que podem ser classificados como hardware.

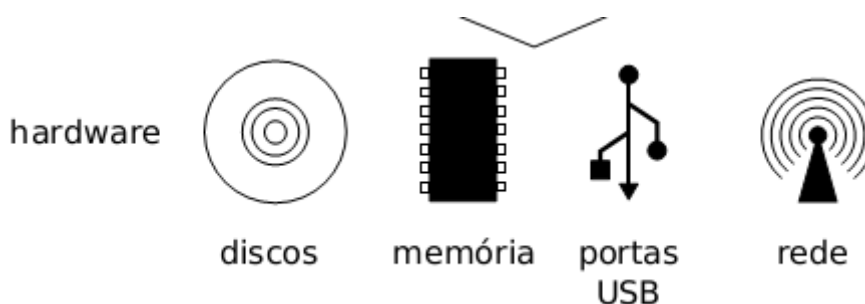


Figura 3 - Exemplos de hardware
Fonte: MAZIERO, 2006

Hardware livre é um hardware projetado e ofertado da mesma maneira que um *software* de código livre. O termo foi primeiramente empregado para refletir o lançamento irrestrito de informação sobre o projeto

de *hardware*, tal como um diagrama, estrutura de produtos e dados de leiaute de uma placa de circuito impresso.

2.2 ARQUITETURA RISC

O chamado *Reduced Instructions Set Computer* - RISC, ou Computador com Conjunto de Instruções Reduzidas, surgiu como uma opção de competição para a arquitetura convencional, *Complex Instruction Set Computer* - CISC, ou Computador com Conjunto de Instruções Complexas. Aletan et al (1992) aponta a origem da arquitetura RISC como resultado dos avanços tecnológicos e de um estudo histórico de como os conjuntos de instruções são usados pelos compiladores.

Para Sharshunov (2004) esse tipo de arquitetura pode ser utilizada em microcontroladores de grandes projetos de super-desempenho, assim como, em microcontroladores de baixo custo e simples. Empresas consolidadas no mercado como IBM, Motorola, Apple desenvolvem projetos com a arquitetura RISC desde 2004. Segundo Sharshunov (2004) existem três princípios fundamentais na arquitetura RISC:

- I) Operações de processamento de dados devem ser implementadas apenas no formato registro a registro. Com isso haverá um grande número de registros e a troca de dados entre registradores e memória é implementada por comandos de “*load/write*” especiais.
- II) O sistema de comando deve conter um pequeno número de comandos que são frequentemente usados com tamanho idêntico, formato regular dois ou três tipos de endereçamento simples.
- III) Cada comando, independentemente do seu tipo, deve ser executado em um ciclo. (SHARSHUNOV, 2004)

Alguns autores como Esponda (1991) e Patterson (1985), nos pioneiros estudos sobre RISC, consideravam os princípios de design de arquitetura mais que uma receita especial de arquitetura e sim uma nova filosofia.

Martins (2005) explica de maneira simples a diferença fundamental da arquitetura RISC com a arquitetura Von-Neumman, ou CISC, atribuindo essa diferença ao barramento do microcontrolador. Barramento é um caminho compartilhado que faz a comunicação entre componentes de um sistema computacional, podendo ser barramento de dados, de endereços e

controles, descreve Murdocca e Heuring (1999). Conforme, ainda, Martins (2005), a arquitetura CISC utiliza o mesmo barramento para fazer a busca de instruções na memória de programa quanto para acessar (escrever ou ler) a memória de dados. Já na arquitetura RISC são utilizados dois barramentos de endereços distintos para acessar instruções e dados.

2.3 ADVANCED RISC MACHINES - ARM

As chamadas *Advanced Risc Machines* (ARM) são processadores de arquitetura RISC de 32 bits, atualmente desenvolvidos pela *ARM Corporation*. O autor Ryzhyk (2006) conta um pouco da história desse processador, que começou em 1983 quando a empresa Acorn Computer procurava um microprocessador de 16 bits para suas máquinas *desktop*. Depois de anos de estudo e com o advento da arquitetura RISC, a empresa conseguiu lançar em 1985 o primeiro processador RISC comercial no mundo. O autor ainda afirma que em meados de 1990 a empresa Apple tomou a decisão estratégica de usar processador ARM em seus projetos, fundindo-se então com a Acorn Computer, dando origem a empresa ARM, com a terceira versão da arquitetura ARM, de endereçamento de 32 bits, suporte MMU (*Memory Management Unit*, ou em português, Unidade de Gerenciamento de Memória) e 64-bit de instruções multi-acumuladas e entrou para o mercado de processadores embarcados.

Sistemas embarcados, para Gajski e Vahid (1995) são sistemas que possuem funcionalidades fixas e previamente determinadas de acordo com as interações com o usuário e o ambiente. De modo geral, esses sistemas possuem a estrutura de um computador, mas sua especificidade faz com que sejam tratados de maneira diferente que computadores em geral.

O autor Bates (1996) já previa os micro controladores ARM como o futuro ideal para produtos portáteis, como telefones celulares, devido a facilidade de uso, confiabilidade, desempenho e baixo custo. Bates (1996) cita ainda como três principais razões as quais esse tipo de microprocessador se adapta bem para dispositivos portáteis:

(...) eles são projetados para baixo consumo de energia por unidade computação, essa baixa potência de consumo leva à queda da

temperatura de funcionamento sem a necessidade de sistemas de refrigeração, reduzindo o custo, a complexidade e tamanho. Temperaturas operacionais mais baixas e menor complexidade levam ao aumento da confiabilidade. Os dispositivos ARM são pequenos (...) e eles proporcionam alto desempenho. Alguns dispositivos ARM são tão poderosos quanto máquinas desktop e ainda são capazes de executar a partir de 2 pilhas AA (BATES, 1996).

Em 1985, segundo Furber (2008), o primeiro processador ARM executava aproximadamente seis milhões de instruções por segundo e usava 0.1 watts. Em 2008, ainda citando Furber (2008), um típico computador embarcado, como o ARM968, ocupa 0.4 mm² na superfície do *chip* de silício e executa perto de 200 milhões de instruções por segundo consumindo 20 mW.

Na página da internet da empresa ARM¹ é possível observar a arquitetura dos micro-controladores ARM atuais. Sobre o aspecto da evolução desses processadores nota-se que já existe suporte para o ARM de 32 bits, assim como ARM com extensões de arquitetura as quais fornecem suporte para linguagem Java e uma maior segurança. Estas melhorias na arquitetura RISC básica são justificadas pela empresa para que os processadores ARM alcancem um bom equilíbrio entre alto desempenho, com tamanho reduzido do código e um baixo consumo de energia.

2.4 SOFTWARE LIVRE

Uma maneira de conceituar o que é software livre pode ser a descrita por Von Hippel e Von Krogh (2003) que o caracteriza como sendo uma aplicação para computador com o código-fonte disponível para inspeção, alteração e utilização por qualquer pessoa, seja ela física ou jurídica. A Free Software Foundation² também conceitua de maneira semelhante Software

¹ <http://www.arm.com>

² Organização sem fins lucrativos com uma missão global de promover a liberdade de usuários de computadores e de defender os direitos de todos os usuários de software livre.
Site oficial: <http://www.fsf.org/>

Livre, como software que pode ser usado, copiado, estudado, modificado e redistribuído sem restrição.

É difícil falar sobre software livre sem ao menos mencionar o sistema operacional Unix. Por definição, pode-se dizer que Unix é um sistema operacional de propósito geral, multiusuário e interativo, que oferece uma série de características, tais quais, sistema de arquivos hierárquico incorporando volumes desmontáveis, capacidade de iniciar processos assíncronos, linguagem de comando do sistema selecionável por base de usuários, compatibilidade com arquivos, dispositivos e processos de Entrada e Saída (I/O) entre outras (RITCHIE e THOMPSON, 1974).

2.5 ANDROID

Meier (2009) descreve o sistema operacional Android como é uma plataforma de software completa para diferentes tipos de dispositivos. Isso quer dizer que é sistema operacional e base completa para rodar softwares específicos para Android. Foi apresentado ao mundo pela empresa Google em 2007 e com o objetivo de mudar a maneira com que aplicações para dispositivos móveis são desenvolvidas.

O sistema é baseado no kernel 2.6 do Linux. Assim como em qualquer outro sistema Linux o Android fornece funcionalidades principais do sistema em sua maioria ligados à abstração de hardware (MEIER, 2009).

“A base de desenvolvimento do Android é Java” (LECHETA, 2009). Contudo, Meier (2009), Lecheta (2009) até mesmo o site oficial dos desenvolvedores de Android³ e outros autores afirmam que o Android não usa a Java Virtual Machine, ao invés disso usa seu próprio mecanismo, chamado Dalvik junto com seu próprio conjunto de instruções. Ainda sobre o Dalvik, Google (2012) a descreve como uma máquina virtual especializada

³ Site oficial dos desenvolvedores Android: <http://developer.android.com/index.html>

desenvolvida para uso em dispositivos móveis, o que permite que os programas sejam distribuídos em formato binário e possam ser executados em qualquer dispositivo Android, independente do processador utilizado.

Lecheta(2009) e Google (2012) citam algumas das vantagens do Android:

- Framework de desenvolvimento de aplicações: reutilização de códigos e facilidade de acessos a recursos exclusivos e manutenção;
- Máquina Virtual Dalvik: a JVM (Java Virtual Machine), do Android, criada e otimizada pensando exclusivamente em dispositivos móveis e suas limitações;
- Navegador Web integrado: Baseado no projeto Open Source WebKit⁴ , o mesmo do iPhone e Nokia Série 60;
- Biblioteca de gráficos otimizada para dispositivos móveis: Exclusiva biblioteca para gráficos 2D e 3D baseada na especificação OpenGL ES 1.0 e pode ter aceleração por hardware como opcional;
- SQLite: Armazenamento de dados estruturados;
- Suporte multimídia: Compatibilidade com os principais formatos do mercado(MPEG4, H.264, MP3, AAC, AMR, JPG, PNG, GIF);
- Telefonia com tecnologia GSM: As aplicações podem manipular operações telefônicas, caso o fabricante permita esse acesso.
- Bluetooth, EDGE, 3G, e WiFi: Foco nas principais tecnologias de transmissão de dados sem fio, também depende da permissão do fabricante para acesso;
- Câmera e GPS: Pensando em transformar o celular em uma ferramenta para interação com redes sociais, também depende da permissão do fabricante para acesso.

⁴ Disponível documentação em <http://www.webkit.org>

- Ambiente de desenvolvimento com plug-in para IDE de programação Eclipse: Inclui emulador, ferramentas para debug e supervisão de memória e desempenho.

2.5.1 Arquitetura Android

A Google (2012) define a plataforma Android como arquitetura com várias camadas de programas, as chamadas pilhas de softwares. Cada uma das camadas tem uma função específica para o sistema operacional.

A Figura 4 descreve essas camadas e a posição que cada um delas se encontra. É possível verificar que são quatro: Kernel (Núcleo) Linux, bibliotecas, framework para aplicações, as aplicações propriamente dizendo, e *runtime* do Android.

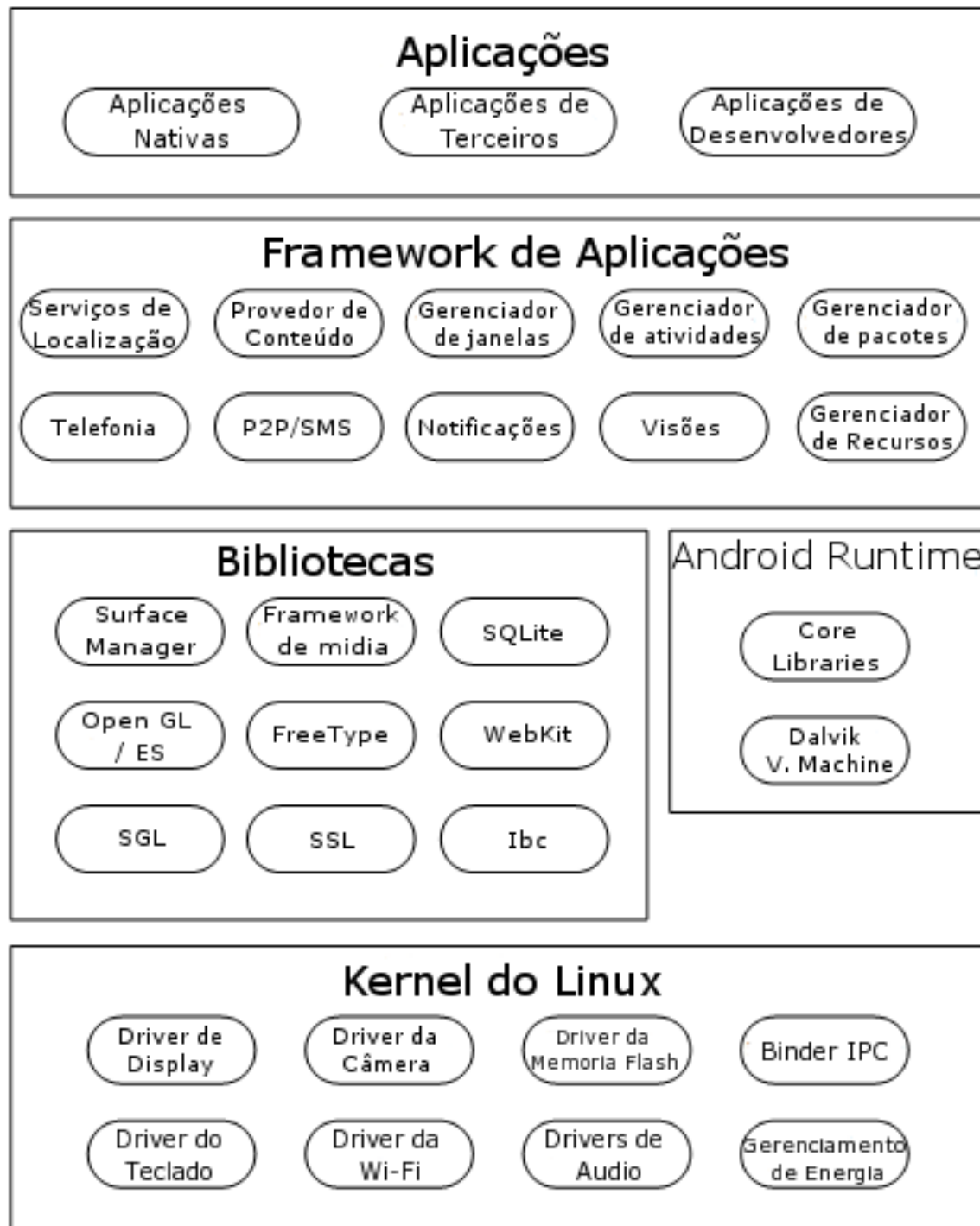


Figura 4 - Arquitetura Android
 Fonte: OLSSON e AKESSON, 2009

Olsson e Akesson (2009) descreve cada uma dessas camadas:

- **Aplicações (*Applications*):** softwares que são nativos do sistema operacional para funções consideradas básicas do Android, como

cliente de e-mail, calendário, mapas e funções para fazer chamadas ou enviar mensagens SMS.

- Framework de Aplicações (*Application Framework*): programas que gerenciam alocação de recursos, aplicações de telefone, mudança entre processos ou programas.
- Android Runtime: bibliotecas do núcleo Java.
- Bibliotecas (*Libraries*): bibliotecas para que desenvolvedores tenham acesso a componentes gráficos, sistema de banco de dados, navegadores web e outros.
- Kernel do Linux: programas de configuração de segurança, gerenciamento de memória e drivers dos dispositivos.

2.6 LÓGICA FUZZY

Os estudos sobre lógica Fuzzy, ou lógica Nebulosa, começaram na década de 60, na Universidade da Califórnia por Zadeh (1965) com intuito de tratar os aspectos vagos de uma informação.

Ao contrário dos sistemas lógicos binários tradicionais, onde o valor verdade só pode assumir dois valores: verdadeiro (1) ou falso (0), na lógica Fuzzy, os valores verdade são expressos linguisticamente como, por exemplo: verdade, muito verdade, não verdade, falso, muito falso. Cada termo linguístico é interpretado como um subconjunto *Fuzzy* do intervalo unitário (GOMIDE; GUDWIN e TANSCHKEIT, 1995).

Mendel (1995) descreve um sistema de lógica *Fuzzy* sendo um sistema capaz de manipular ao mesmo tempo com dados numéricos e conhecimentos linguísticos, que são normalmente, informações imprecisas. Ainda nesse sentido, segundo Sandri e Correa (1999), a lógica *Fuzzy* pode se tornar bastante útil para programar requisitos complexos em controladores simples, de fácil manipulação e custo, onde o modelo matemático do problema está sujeito a incertezas.

A ideia de aplicar lógica *Fuzzy* é para criar condições de tratar informações lógicas seguindo regras naturais de raciocínio, por exemplo: se

tensão está muito baixa então desligue motor (BILOBROVEC; MARÇAL e KOVALESKI, 2004).

Barros (2002) ilustra como funciona a lógica dos conjuntos *Fuzzy*. Se for considerado, por exemplo, o conjunto dos números naturais pequenos:

$$F = \{n \in \mathbb{N} : n \text{ é pequeno}\}$$

Nesse conjunto é válido dizer que o zero pertence a ele? E o número 1000? Usando a lógica *Fuzzy*, se diria que ambos pertencem a F , porém com diferentes graus de pertinência, de acordo com a propriedade que o caracteriza. Ou seja, a função de pertinência de F deve ser “construída” de forma coerente com o termo “pequeno” que caracteriza seus elementos no conjunto universo dos números naturais (BARROS, 2002).

Antes de começar a estudar a aplicação da lógica *Fuzzy*, Dias (2007) evidencia a importância de se levar em consideração três diferentes tipos de incertezas. São elas:

- A estocástica é o grau de incerteza da ocorrência de um determinado evento. O evento propriamente dito é bem definido e a incerteza estocástica não está relacionada à quando o evento ocorre. Normalmente, este tipo de incerteza, é usado na descrição de fenômenos com um grande número de possibilidades.
- A incerteza informal resulta de uma falta de informação e do conhecimento sobre uma situação.
- A incerteza linguística resulta da imprecisão do método linguístico, pois descreve categorias subjetivas com significados que dependem do contexto de sua utilização (DIAS, 2007).

Estas incertezas citadas por Dias (2007), devem ser consideradas uma vez que haverá uma tradução do conhecimento de um especialista humano para processos dentro de um controlador *Fuzzy*.

2.6.1 Controlador *Fuzzy*

Gomide; Gudwin e Tanscheit (1995) propõe como estrutura fundamental de um controlador *Fuzzy* conforme a Figura 5, sendo seus componentes básicos: a interface de fuzzyficação, a base de conhecimento, a base de dados, o procedimento de inferência e a interface de Defuzzificação.

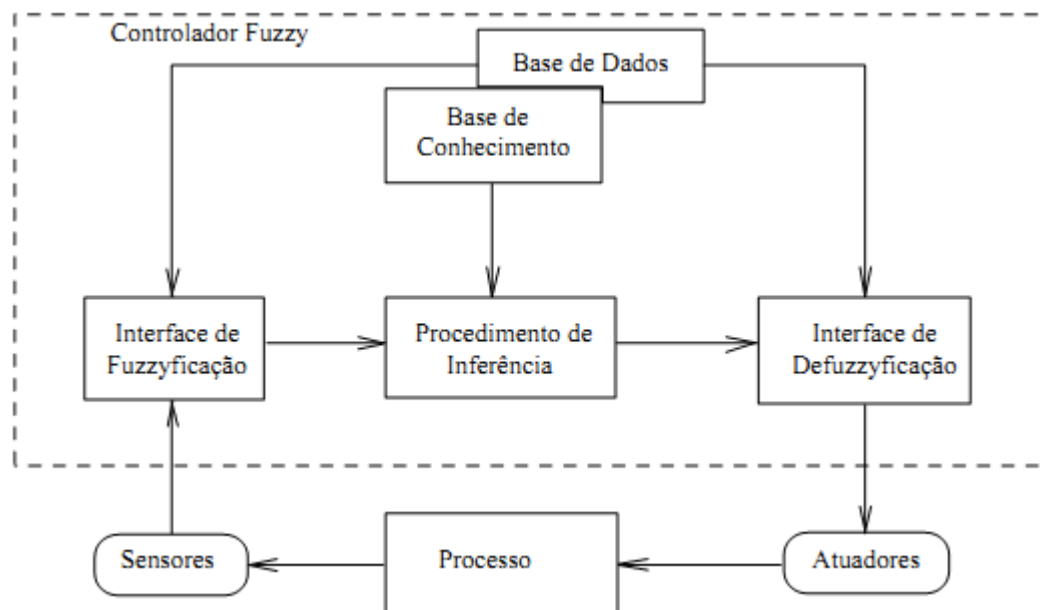


Figura 5 - Estrutura Controlador Fuzzy
 Fonte: GOMIDE; GUDWIN e TANSCHKEIT, 1995

A interface de fuzzyficação transforma os valores de entrada em valores fuzzyficados, ou seja, transforma números em conjuntos Fuzzy, de modo que transforma essas entradas em instâncias de variáveis linguísticas. A base de conhecimento é onde ficam armazenadas as regras e outras definições necessárias, como as funções para cálculo de pertinência.

A base de dados armazena as definições necessárias sobre discretizações, fórmulas matemáticas e funções de pertinências.

No procedimento de inferência é onde os dados de entrada, já fuzzyficados, são processados, valendo-se das regras definidas anteriormente na base de conhecimento e dos operadores de implicação Fuzzy.

Por fim, na interface de defuzzyficação transforma as ações de controle Fuzzy inferidas em ações de controle não-fuzzy. Ou seja, transforma os valores Fuzzy, vindos do passo anterior, com os valores dos universos de discurso reais das variáveis.

2.6.2 Processo de Fuzzyficação

Para modelar informações qualitativas, com falta de exatidão, imprecisas, incertas ou ainda com ruído, que são utilizadas pela lógica *Fuzzy*, existem as técnicas de modelagem e controle *Fuzzy*. Para Gomide; Gudwin e

Tanscheit (1995) modelagem e controle *Fuzzy* trata do relacionamento entre entradas e saídas, agregando vários parâmetros de processo e controle. O autor explica que isso permite a consideração de processos complexos, de modo que os sistemas de controle resultantes proporcionam um resultado mais acurado, além de um desempenho estável e robusto. Sendo assim, Gomide; Gudwin e Tanscheit (1995) garante que problemas anteriormente intratáveis passam agora a ser solúveis.

As variáveis de entrada de um controlador Fuzzy devem ser traduzidas ou transformadas de valores numéricos para conjuntos Fuzzy que as representem.

O Quadro 1 mostra uma coleção de itens comumente usados nas comunicações. Nela temos alguns termos e quais valores esses podem assumir. Por ela se pode verificar a *fuzzificação* de alguns valores que normalmente são numéricos.

Quadro 1 - Termos nas telecomunicações.

Termos	Valores
Alias	Nenhum, um pouco, alto
Largura da banda	Banda estreita, banda larga
Blur	Algum, um pouco, muito
Correlação	Baixa, média, alta, perfeita
Erros	Grande, médio, pequeno, muito, não muito bom, muito grande, muito pequeno, quase zero.
Frequência	Alta, baixa, ultra-alta.
Resolução	Baixa, alta
Amostragem	Baixa-taxa, média-taxa, alta-taxa, muito alta-taxa
Estabilidade	Estável (levemente distorcida, altamente distorcida, acima do normal distorcida, criticamente distorcida), não estável.

Fonte: MENDEL, 1995

Ainda no Quadro 1 é possível verificar a coluna chamada de “Termos”. Ela simboliza as variáveis linguísticas que representam os conjuntos

Fuzzy. Mendes (1995) destaca o exemplo de um dos itens do quadro, a estabilidade. Se for analisado o termo estabilidade, não é possível encontrar nada de *Fuzzy* nele, uma vez que um sistema pode estar estável ou não. Contudo, para descrever o grau de estabilidade pode usar que a estabilidade da comunicação está em um grau de 0.3. *Fuzzyficando* esse termo numérico, poderíamos ter uma estabilidade levemente distorcida.

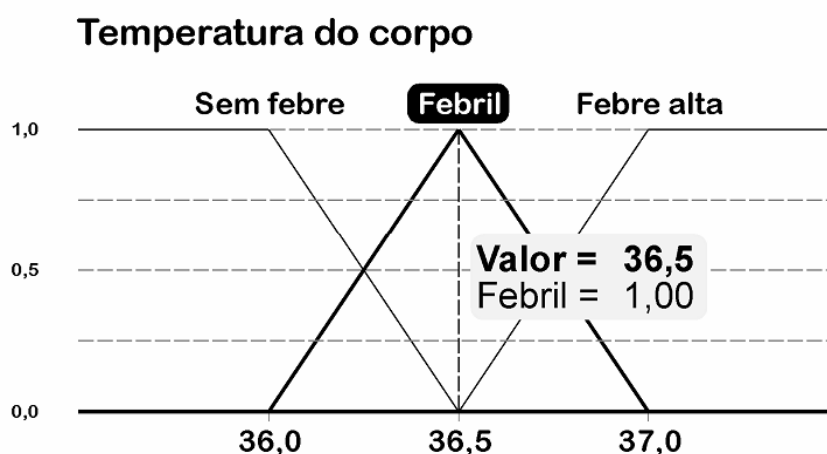


Figura 6 - Variáveis linguísticas e valores numéricos
Fonte: DIAS, 2007

A correlação entre os termos linguísticos e os valores numéricos que uma propriedade pode assumir é explicada por Dias (2007), no exemplo do corpo febril. Existe uma área *Fuzzy* entre os valores de 36 a 37 graus, onde poderia se dar o diagnóstico de febre. A Figura 6, mostra que existem regiões de sobreposição de termos linguísticos, já intervalo de 36,0 até 36,5 °C é possível considerar o paciente como “Febril” ou “Sem febre”. A conclusão sobre o diagnóstico se dará da interpretação do valor do grau de pertinência $\mu(T)$ (plano y) de cada um dos termos linguísticos.

Tomando por base ainda o exemplo do corpo febril, se o indivíduo estivesse com a temperatura do corpo em 36,6 °C pode-se concluir que esse valor estaria na área nebulosa, pois está contido na área do triângulo que representa os valores que podem ser considerados como corpo febril, com o grau de pertinência (eixo y) de 0,80 e está também dentro da área do trapézio que representa os valores que podem ser considerados como febre alta, com o grau de pertinência de 0,20. Isto é demonstrado na Figura 7.

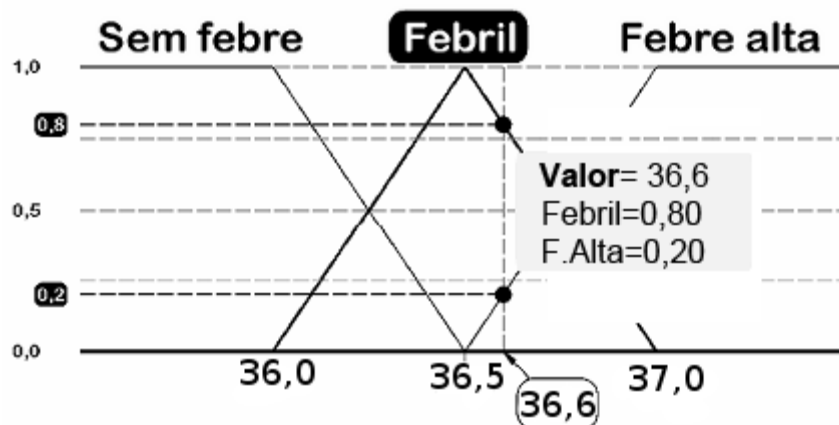


Figura 7 - Grau de pertinência para febre de 36,6 °C

Fonte: o autor, adaptado de DIAS, 2007

O grau de pertinência, para Dias (2007), pode ser considerado como o grau de verdade de uma informação, podendo variar de absolutamente verdadeira, sendo um, e absolutamente falsa, sendo zero. A lógica Fuzzy é baseada na teoria dos conjuntos, sendo que um valor pode estar contido em um conjunto, com determinado grau de verdade nessa informação, que indica a pertinência total, parcial ou a não pertinência total do valor em um grupo.

As funções de pertinência triangulares e trapezóides são as mais comuns, e têm provado serem boas em efetividade e eficiência. Os conjuntos difusos devem abranger o eixo X, cobrindo todo o intervalo, ou o universo de discurso, para uma entrada de um sistema, mapeando para o intervalo de 0 a 1 do eixo Y as pertinências de uma entrada. Sobreposição entre limites de conjuntos é desejável e a chave para a operação suave do controlador. São permitidas pertinências em múltiplos – até mesmo em contraditórios – conjuntos (CABRAL, 1994).

2.6.3 Operações entre conjuntos Fuzzy

Existem operações que podem ser feitas entre conjuntos Fuzzy, conforme explica Barros (2002). Considerando um conjunto Fuzzy A e um conjunto Fuzzy B, a união entre A e B resulta em outro conjunto Fuzzy cuja pertinência é dada por:

$$(A \cup B)(x) = \text{máximo}_{x \in U} \{A(x), B(x)\}$$

A interseção desses mesmos conjuntos resultaria então em outro conjunto Fuzzy de pertinência:

$$(A \cap B)(x) = \text{mínimo}_{x \in U} \{A(x), B(x)\}$$

Por fim, o complemento de A é o conjunto Fuzzy A', cuja função de pertinência é dada por:

$$A'(x) = 1 - A(x), x \in U$$

Estas operações são demonstradas por gráficos na Figura 8 abaixo:

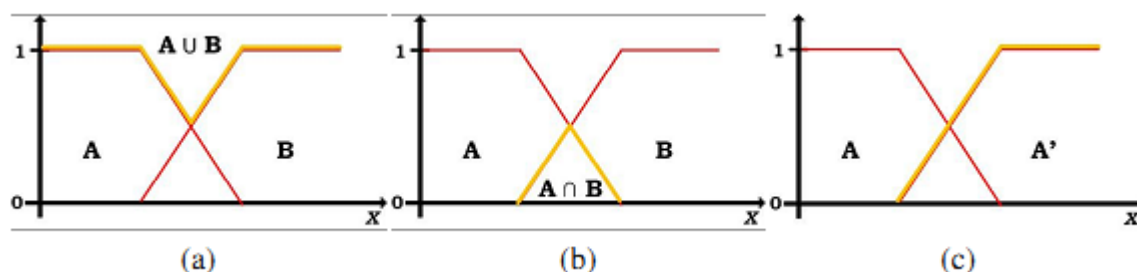


Figura 8 - (a) União entre conjuntos Fuzzy (b) Interseção entre conjuntos Fuzzy (c) Complemento de um conjunto Fuzzy

Fonte: DIAS,2007

No **Erro! Fonte de referência não encontrada.**, Dias (2007) simplifica a lógica das operações entre conjuntos Fuzzy a comparando com a lógica booleana tradicional. Sendo assim, união pode ser considerada o operador lógico "OU", interseção com o operador lógico "E" e complemento com a operação de negação.

Quadro 2 - Operadores para lógica Fuzzy

Operador	Definição
Fuzzy E = (f-E)	A (f-E) B = $\min[\mu(A), \mu(B)]$
Fuzzy OU = (f-OU)	A (f-OR) B = $\max[\mu(A), \mu(B)]$
Fuzzy NÃO = (f-NÃO)	(f-NÃO) A = $1 - \mu(A)$

Fonte: DIAS,2007

Tabela 1 - Tabela verdade Fuzzy

Operador E			Operador OU			Negação	
A	B	Min(A,B)	A	B	Max(A,B)	A	1-A
0	0	0	0	0	0	0	0
0	1	0	0	1	1	0	1
1	0	0	1	0	1		
1	1	1	1	1	1		

Fonte: DIAS, 2007

No Quadro 2 é demonstrado na coluna "Operador" o que seria o equivalente Fuzzy do operador na lógica tradicional (SE, OU e NÃO). A coluna "Definição" mostra como é a operação dentro de uma lógica Fuzzy. Por fim, fazendo uma correlação dos operadores da lógica tradicional e seus correspondentes na Fuzzy pode-se estabelecer uma tabela verdade para operações Fuzzy, conforme é demonstrado na Tabela 1.

2.6.4 Base de conhecimento

Para Sandri e Correa (1999), a base de conhecimento consiste em uma base de regras que caracterizam as estratégia e meta que o sistema Fuzzy solucionará.

Essa base de regras é formada por premissas do tipo:

Se <premissa> então <conclusão>

Como por exemplo: "Se temperatura é alta e umidade do ar é alta então ligue o ar condicionado". Estas regras, em um controlador Fuzzy, são processadas juntamente com os dados de entrada pelo procedimento de inferência. Sandri e Correa (1999) ainda reforçam que é importante que existam tantas regras quantas forem necessárias para mapear totalmente as combinações dos termos das variáveis. Isto é necessário para garantir que a base esteja completa, e garantir que sempre ao menos uma regra seja disparada a qualquer entrada. Na formulação dessas regras se faz necessário uma checagem de consistência, evitando que existam contradições ou interações entre as regras que possam gerar um ciclo.

Para ilustrar melhor um mapa de regras, Dias (2007) usa o exemplo de automatização da manobra de um caminhão de um ponto inicial arbitrário até uma rampa de carregamento. Nesse exemplo o caminhão deve se locomover em velocidade constante parar quando estiver na rampa de carregamento.

O Quadro 3 descreve essa manobra, com dois parâmetros de entrada: posição do veículo e orientação do veículo. Cada um desses parâmetros pode assumir, respectivamente, cinco e sete variáveis linguísticas, gerando assim trinta e cinco possibilidades de combinação para regras diferentes.

Quadro 3 - Exemplo de mapa de regras

E		Posição do veículo – x (m)				
		Totalmente à esquerda	Esquerda	Centro	Direita	Totalmente à direita
Orientação do veículo – β (°)	Totalmente à esquerda	Esquerda	Mediamente esquerda	Mediamente esquerda	Bem esquerda	Bem esquerda
	Bem à esquerda	Direita	Esquerda	Mediamente esquerda	Bem esquerda	Bem esquerda
	Esquerda	Mediamente direita	Direita	Esquerda	Mediamente esquerda	Bem esquerda
	Neutra	Mediamente direita	Mediamente direita	Neutro	Mediamente esquerda	Mediamente esquerda
	Direita	Bem direita	Mediamente direita	Direita	Esquerda	Mediamente esquerda
	Bem à direita	Bem direita	Bem direita	Mediamente direita	Direita	Esquerda
	Totalmente à direita	Bem direita	Bem direita	Mediamente direita	Mediamente direita	Esquerda

Fonte: DIAS,2007

Dada as variáveis de entrada para o controlador Fuzzy, cada regra terá um fator de pertinência, ou seja, qual o grau de verdade que as entradas terão se foram englobadas por esta determinada regra. Nesse momento, aplicam-se as regras dos operadores Fuzzy.

Por exemplo, um determinado valor de entrada para orientação e posição, se enquadrrou a seguinte regra:

- SE orientação é bem à esquerda E posição é esquerda ENTÃO esquerda. Sendo o grau de pertinência para orientação 0,7 e para

posição 0,5 então por se tratar de um operador E, a saída será o mínimo entre os dois valores, ou seja, 0,5.

Ainda caso houver mais de uma regra a qual a entrada se encaixe com algum dos termos linguísticos, une-se as regras com o operador OU, sendo assim, utilizará o maior valor de pertinência.

- Regra 1: SE orientação é bem à esquerda (0,7) E posição é esquerda (0,5) ENTÃO esquerda.

- Regra 2: SE orientação é bem à esquerda (0,7) E posição é bem à esquerda (1,0) ENTÃO esquerda.

Sendo assim a regra 1 resultará em Esquerda (0,5), uma vez que é uma operação de E (mínimo) e a regra 2 resultará em Esquerda (0,7). Combinando as duas regras, como a função OU (máximo), a saída final será Esquerda (0,7).

2.6.5 Defuzzificação

O valor gerado pelo sistema de inferência, baseado no mapa de regras, deve ser “defuzzificado” de modo que gere um valor preciso, ou seja, um valor mais físico, real e não um valor Fuzzy. Isto é, o resultado da inferência Fuzzy será um ou mais conjunto Fuzzy com certo grau de verdade diferente de zero. Na Defuzzificação esses valores serão combinados de uma ou mais saídas linguísticas para uma saída numérica e mais precisa.

Cabral (1994) defende a execução de um processo de Defuzzificação por duas razões: a primeira é decifrar o significado de ações vagas (difusas), utilizando funções de pertinência; a segunda é resolver os conflitos entre ações conflitantes, que podem ter sido acionadas durante certas condições na avaliação das regras.

Sandri e Correa (1999) afirmam em seu trabalho que não há nenhum procedimento sistemático para a escolha da estratégia de Defuzzificação, existem alguns métodos dos quais se destaca:

- Critério do máximo (MAX), que escolhe o ponto onde a função inferida tem seu máximo. Ou seja, encontra o valor de saída através do ponto em que o grau de pertinência da distribuição da ação de controle atinge

o primeiro valor máximo; Entretanto, Corcoll-Spina (2010) considera o critério do máximo um procedimento radical uma vez nele que só são levadas em conta as regiões de maior possibilidade entre os possíveis valores da variável que modela o conceito Fuzzy em questão. Isto é, nesse método é plausível que sejam desconsiderados valores importantes para o resultado final, interferindo diretamente na qualidade do resultado.

- A média dos máximos (MDM), que representa o valor médio dentre todos os pontos de máximo quando existe mais de um máximo;
- Método do centro de área (CDA), que retorna o centro de gravidade de área da função inferida.

O método de centro de área ou centróide, para Corcoll-Spina (2010) é o mais utilizado e se assemelha muito a uma média aritmética. O diferencial fica nos pesos da média, que no caso, indicam o grau de compatibilidade ou pertinência do valor Fuzzy com o conceito que foi modelado no conjunto Fuzzy. Resumidamente, este método oferece a média das áreas de todas as figuras que representam o grau de pertinência de um subconjunto Fuzzy.

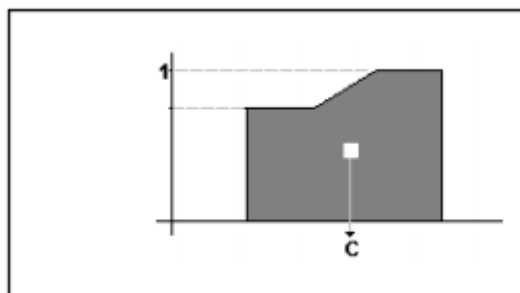
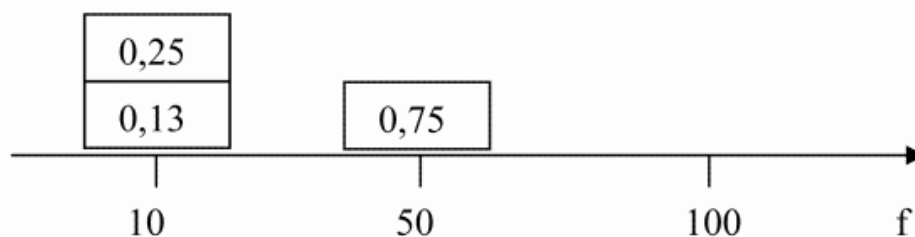


Figura 9 - Exemplo do método de centro de áreas
Fonte: CORCOLL-SPINA, 2010

A Figura 9 exemplifica justamente esse método. A área tracejada representa a área dos conjuntos Fuzzy. Já a área em cinza, representa a soma das áreas dos subconjuntos Fuzzy inferidos pelo controlador Fuzzy. O ponto C é o valor que representa o centro dessa área, ou seja, o valor desfuzzificado.

Aritmeticamente, para obter-se a saída do controlador Fuzzy, os graus de pertinência obtidos pela aplicação de cada regra serão os valores

para a média, e os pesos serão dados pelos limites da função de pertinência da variável de saída.



$$f = \frac{0,25 \times 10 + 0,13 \times 10 + 0,75 \times 50}{0,25 + 0,13 + 0,75} = \frac{42}{1,13} = 38$$

Figura 10 - Exemplo aritmético Defuzzificação
Fonte: Adaptado de SOUZA, 2007

Usando a Figura 10 de Souza (2007) como um exemplo hipotético de um processo de Fuzzy, os valores 10, 50 e 100 são os limites para a variável linguística que descreve o conjunto Fuzzy, ou seja, são os valores dos termos linguísticos das ações possíveis. Os valores 0.25, 0.13 e 0.75 são os respectivos graus de verdade. Logo abaixo, faz a fórmula da média aritmética ponderada, tendo como resultado um possível valor para resultado da defuzzificação.

2.7 GEOREFERENCIAMENTO E GPS

De maneira objetiva georreferenciamento é descrito como sendo a relação de uma informação com sua localização geográfica. Comumente representado por objetos como mapas e globos, o georreferenciamento está hoje embutido em vários sistemas e serviços online, que nos mostram onde um endereço ou local está, quão longe fica e como podemos chegar lá (HILL, 2006).

Em um sentido um pouco mais amplo, georreferenciamento consiste em relacionar informações (eventos, imagens, descobertas, amostras) com localizações geográficas, seja através de referências informais, como o

nome do lugar, ou formais geoespaciais, com base em coordenadas de latitude e longitude (VICEDO;GÓMEZ, 2007).

Hasegawa et al (2000) define os Sistemas de Posicionamento Global, ou GPS, são aparelhos de grande potencial e utilizados em aplicações que envolvam posicionamento rápido e acurado. O autor explica seu funcionamento:

O GPS funciona em um sistema que é composto por uma constelação de 24 satélites distribuídos em seis planos orbitais localizados aproximadamente a 20000 km de altitude. Os sinais enviados pelos satélites GPS são capturados na superfície terrestre por equipamentos apropriados (receptores GPS), permitindo a determinação da posição 3D; solução que só é possível se pelo menos quatro satélites forem sintonizados simultaneamente pelo receptor. (HASEGAWA et al ,2000).

Nessa citação, Hasegawa et al(2000) demonstra o funcionamento de um GPS por posicionamento absoluto, conforme ilustra a Figura 11 abaixo.

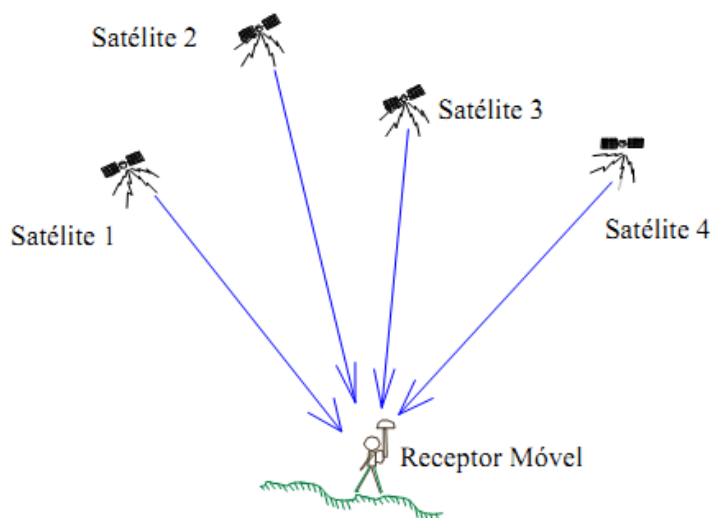


Figura 11 - Posicionamento Absoluto
Fonte: HASEGAWA et al ,2000

Ao contrário da técnica de posicionamento absoluto, a técnica de posicionamento relativo, dependente de receptores em lugares fixos na terra, os quais cruzam a informação dos satélites com a localização do usuário do aparelho GPS, conforme a Figura 12 abaixo.

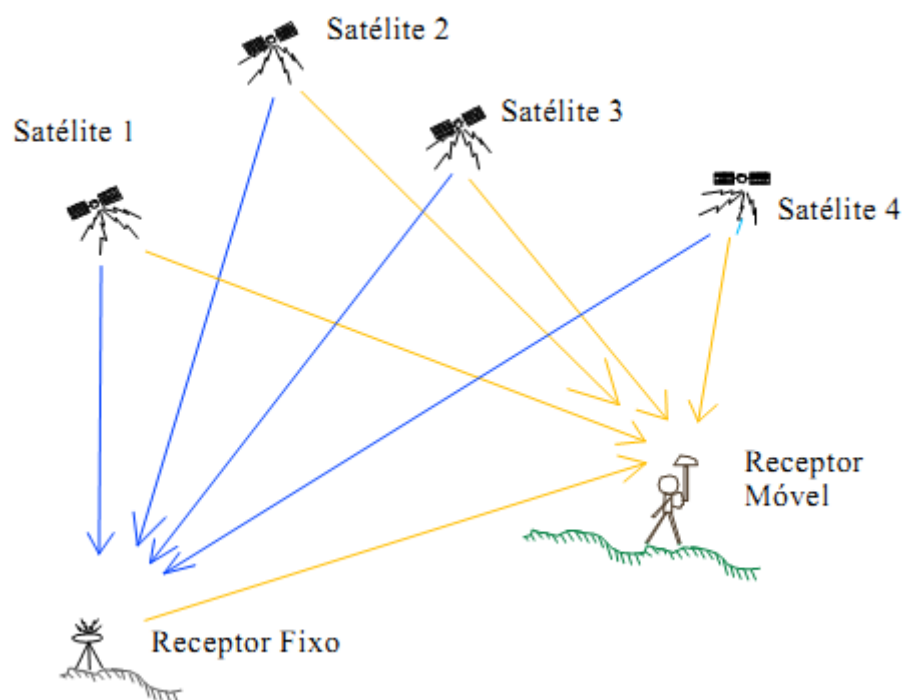


Figura 12 - Posicionamento Relativo
 Fonte: HASEGAWA et al, 2000

Fazendo uma ligação entre georreferenciamento e agricultura, é possível citar Tschiedel e Ferreira (2002) que justificam esse elo como uma maneira satisfazer a necessidade crescente de eficiência, em todos os setores da economia, para manter a competitividade. Sendo assim, Tschiedel e Ferreira (2002) vêem nos avanços nas tecnologias em georreferenciamento uma nova forma de enxergar a propriedade rural, deixando de ser uma somente e sim várias propriedades dentro da mesma, porém com características específicas. Desse modo o empresário rural, por controlar cada vez mais a sua linha de produção, pode se tornar mais competitivo no mercado.

2.8 SISTEMA DE NAVEGAÇÃO

Um sistema de navegação tem como principal função obter a posição do veículo relativa a algum sistema de referência e calcular comandos de direção (KAYTON, 1969). A posição do veículo obtida com um sistema de monitoramento via satélite e GPS, associado a mapas digitais, representados

no monitor de um computador, têm sido usados na concepção de sistemas de localização e navegação terrestre (PUGLIESI, 2002).

Para Pallett (1993), um veículo requer para sua navegação a determinação de sua posição em relação a um ponto de partida e aos pontos de sua rota a fim de alcançar um destino estabelecido. O autor ainda define como sendo básicas algumas informações, são elas: tempo, velocidade, distância entre pontos, latitude, longitude, direção do polo magnético, direção e velocidade do vento, posições relativas de pontos conhecidos na superfície da terra e pontos celestiais.

Os autores Laird Evans e Steven (1996) concluíram em suas pesquisas comparando mapas de navegação, guias de rotas e dispositivo contendo somente texto, que pessoas têm mais facilidade em assimilar informações gráficas que aquelas apresentadas somente na forma de texto. Ainda nessa linha de raciocínio, Morita (1993), cita como uma das vantagens dos sistemas automatizados de navegação o fato que eles ajudam o usuário a determinar a sua próxima ação. O autor ainda compara esses sistemas a simples atlas eletrônicos, que não podem prover tal informação porque não têm interface humana para navegação. Portanto, para Morita (1993) a grande vantagem é que os sistemas que utilizam processos automatizados ajudem o motorista na tomada de decisões.

Hasegawa et al (1999) demonstra em seu estudo uma estrutura básica para um sistema de posicionamento e locomoção, conforme mostra a Figura 13. Deve ser feita a integração das informações provenientes da base de dados e as informações de posição obtidas em tempo real. Em função das necessidades do usuário e da disponibilidade do sistema podem-se fazer consultas específicas e, finalmente, fazer a visualização de uma porção da base de dados com a posição instantânea do veículo, seja um automóvel, uma aeronave, uma embarcação, etc.

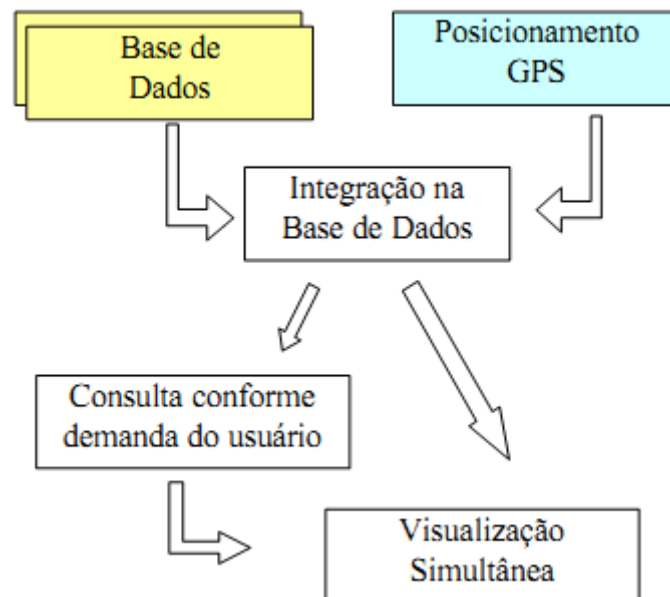


Figura 13 - Estrutura básica para sistemas de navegação
 Fonte: HASEWAGA et al, 1999

2.9 ROBÓTICA AGRÍCOLA

Para Souza (2007) o panorama mundial da agricultura tem mudado acentualmente nos últimos anos. A automação dos processos agrícolas tornou-se indispensável para que o agricultor mantenha-se competitivo e, ainda, buscando minimizar os impactos ambientais. Para que a agricultura tenha um suporte nesse novo contexto, destaca-se a agricultura de precisão, para que haja um manejo diferenciado do solo, insumos e plantas. Assim, Souza (2007) explica que uma das aplicações da agricultura de precisão, é na robótica agrícola, que visa a otimização do uso dos recursos, uso racional da terra e a precisão na execução das tarefas agrícolas.

Blackmore (2009) afirma que as novas tecnologias de automação agrícola estão sendo desenvolvidos por pesquisadores universitários preocupados com a eficiência e eficácia com que é realizado as práticas agrícolas, dando origem a novas oportunidades de serviços e requisitos agronômicos, embora de maneiras radicalmente diferentes daqueles usados, e apresentou novos conceitos, como dependência e independência de tempo, inteligência da máquina, segurança, confiabilidade, entre outros.

Para exemplos de estudos de robôs agrícolas pelo mundo, pode-se citar trabalho como o de Monta, Kondo e Shibano (1995), que propõe um robô com braços mecânicos, sensor visual e locomoção autônoma para o cultivo da uva. Os autores Astrand e Baerveldt (2002) também deram sua contribuição nesta área na criação de um robô autônomo para controle de plantas invasoras em ambiente aberto, por reconhecimento de cores.

2.10 TRABALHOS RELACIONADOS

Seguindo a linha do capítulo anterior, sobre trabalhos na robótica agrícola, mas em especial, fazendo a utilização da lógica Fuzzy, pode-se citar o trabalho de Zhang (2001), no qual apresenta o desenvolvimento de um controlador para o volante de direção de um veículo agrícola. O controlador de Zhang (2001) implementa as correções de direção com base apenas na direção do veículo, e o erro entre o ângulo desejado e os ângulos reais das rodas.

Apesar de ter como escopo veículos agrícolas fisicamente desiguais, o projeto de Zhang (2001) aparenta ter focar em tratores, ou seja, veículos de porte grande.

Se for traçado um comparativo entre o modelo de Zhang (2001) e o proposto neste projeto é possível perceber que a grande diferença está no uso de GPS e georreferenciamento. No projeto de Zhang (2001), a correção dada pela lógica Fuzzy é aplicada no direcionamento das rodas de acordo com a direção que o usuário quer que o veículo tome, enquanto no projeto Navigo-Agrobot a Fuzzy é feita para a correção da rota usando dados georreferenciados. Além disso, o projeto agrobot é focado para um veículo agrícola não tripulado, autônomo, de pequeno porte, movido a geradores elétricos.

Também pode ser visto na literatura o trabalho de Schmidt e Lages (2000) que cruzam robótica com lógica Fuzzy. O modelo de Schmidt e Lages (2000) não tem como apelo o universo agrônomo, mas sim visa a criação de um sistema baseado em visão computacional para navegação. Os

dados que orientam a movimentação do robô são obtidos através da transformada de Hough.

Segundo Andersen e Seibel (2001) e Atiquzzaman (1999), A transformada de Hough é conhecida como uma técnica matemática eficiente para descobrir padrões descontínuos inseridos em imagens ruidosas. Também é conhecido por realizar a detecção de formas geométricas em imagens digitais. No trabalho de Schmidt e Lages (2000) o controle de movimentação do robô é baseado em lógica Fuzzy.

Apesar de utilizar Fuzzy na robótica, a correção de rota diferentemente do aqui proposto, é feita por reconhecimento de padrões em imagens e não utilizando dados georreferenciados.

Guerra (2009) desenvolve um sistema de controle de navegação para um veículo autônomo, utilizando lógica Fuzzy. O sistema de Guerra (2009) controla e monitora a navegação do veículo de uma estação remota através de uma ligação de dados de telemetria.

De uma maneira aparentemente parecida com a ideia do modelo do Navigo Agrobot, o usuário indica, usando um aplicativo desenvolvido, os pontos de navegação, referenciando-os pela sua latitude e longitude em um mapa.

Quando a navegação é iniciado, o sistema recebe periodicamente os pacotes de dados que indicam a posição e direção do veículo, estes dados são processados pelo controle Fuzzy que retorna para acionar os comandos sobre o veículo.

O diferencial entre o presente trabalho e o projeto de Guerra (2009) está na telemetria. O veículo transmite as informações para uma base que calcula a correção e devolve esta instrução para o veículo. Com a barateamento do hardware, atualmente é possível deixar todo este processamento no próprio veículo, sem custo alto ou erro de transferência de pacotes, que ocorre por telemetria.

3 METODOLOGIA

Para o desenvolvimento do sistema computacional de navegação e correção de rota usando Lógica Fuzzy foi necessário um computador embarcado, equipado de um receptor GPS para aquisição dos dados georreferenciados e um computador pessoal para a programação do dispositivo. Ainda foi feito uso da IDE de desenvolvimento Eclipse, bibliotecas para desenvolvimento, como Google Maps e Fuzzy Engine.

Os testes foram realizados no próprio computador embarcado ARM, em campo aberto, no município de Ponta Grossa - Paraná.

3.1 MATERIAL

Para alcançar os objetivos definidos foram utilizados os seguintes materiais:

- Computador embarcado Friendly ARM modelo Mini6410, fabricado por FriendlyARM Co., Ltd, a Figura 14 mostra a fotografia do dispositivo, com as seguintes características técnicas⁵.
 - a) CPU: Samsung S3C6410A, ARM1176JZF-S, 533 MHz.
 - b) DDR RAM: 256 Mb onboard e 32 bit data bus.
 - c) Flash Memory: 1 Gb MLC Nand flash.
 - d) LCD: 4.3 polegadas, colorido, sensível ao toque.
 - e) Interfaces e acessórios externos.
 - Porta Ethernet, 100 Mbit RJ-45 (chipset DM9000).
 - Porta serial DB9 RS232 five-wire.
 - Porta mini USB Slave 2.0.

⁵ Disponível em: http://armdevs.Googlecode.com/files/overview_mini6410.pdf

- Saída de áudio.
- Entrada de microfone.
- Porta RCA, saída de TV.
- Porta USB Host 1.1.
- Conector padrão para leitura/gravação de cartão SD.
- Receptor de infravermelho.
- Conector de energia 5V.
- I2C-EEPROM chip (256 byte) para teste de I2C Bus.
- 4 * LEDs (verde).
- 8 * Botões de usuário.
- Resistor ajustável para conversor AD.
- PWM sirene.
- Bateria backup para EPROM.

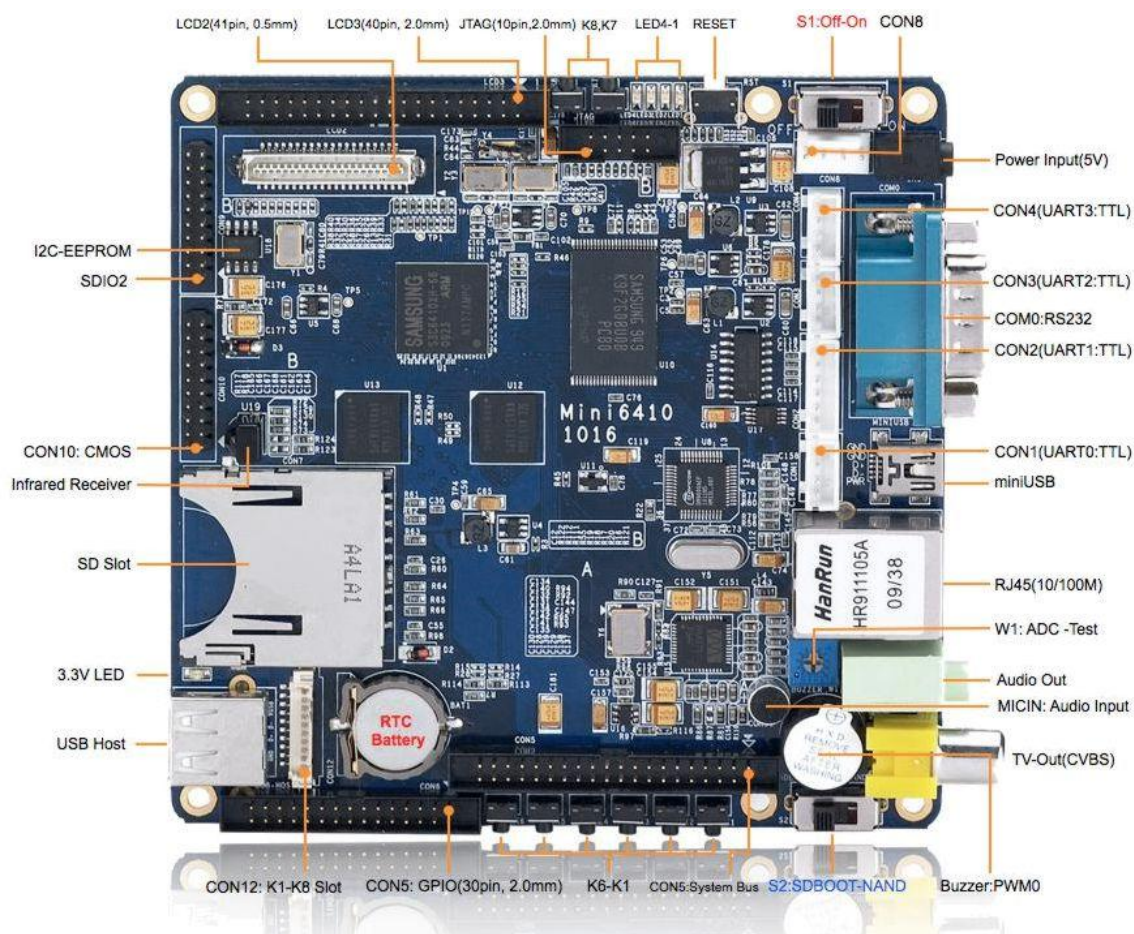


Figura 14 - Fotografia do Friendly ARM Mini 6410

Fonte: http://armdevs.googlecode.com/files/overview_mini6410.pdf

- Receptor de sinal GPS Gstar GS-87 (<http://icdshop.net/download/Gstar/GS-87.pdf>), que utiliza o microcontrolador GPS SiRFstar III, sendo este um microcontrolador GPS de alta sensibilidade, fabricado pela empresa SiRF Technology Holdings, Inc, a Figura 15 mostra a unidade de recepção de sinal GPS.



Figura 15 - Receptor GPS Gstar GS-87
Fonte: o autor

- Sistema Operacional Android 2.3.4, para o ambiente de teste no computador embarcado Friendly ARM.
- Linguagem de programação Java para Android.
- Ambiente de programação Eclipse, que é uma IDE (Integrated Development Environment) para a produção de programas de computadores aplicando técnica RAD (Rapid Application Development).
- *Plug-In* do Eclipse ADT (Android Development Tools).
- Biblioteca Google Maps.

3.1.1 Biblioteca GeoUtils

A empresa Google, responsável pela criação e desenvolvimento da plataforma Android, disponibiliza para a comunidade uma classe em Java que facilita para o desenvolvedor fazer cálculos geodésicos entre pontos no plano, ou seja, envolvendo latitude e longitude. No presente trabalho foi utilizada uma função para calcular em quilômetros a distância entre dois pontos no globo e também a função "*bearing*", que calcula o rumo em graus entre dois pontos na Terra. Essa função parte do princípio que o grau

zero é o norte, retornando a posição para um determinado destino, ou seja, o ângulo em relação ao norte.

A função para calcular a distância entre os pontos leva em consideração que a terra não é plana e sim elipsoidal. Para ter um cálculo mais preciso, a função trata o planeta Terra como uma esfera. Com isso, é possível calcular a distância entre dois pontos no globo tendo como base o raio da Terra de, aproximadamente, 6371 km.

Distância = $\arccos(\sin(\text{lat1}) * \sin(\text{lat2}) + \cos(\text{lat1}) * \cos(\text{lat2}) * \cos(\text{Lon2} - \text{Lon1}))$
 $* \text{RaioDaTerra};$

Sendo:

- lat1 e lat2 = latitude ponto 1 e 2, respectivamente, em radianos.
- lon1 e lon2 = longitude ponto 1 e 2, respectivamente, em radianos.
- RaioDaTerra = 6371 km
- Conversão entre graus e radianos é obtida multiplicando-se o ângulo por pi e dividindo-o por 180.

Para calcular a direção (*bearing*) entre os pontos, a biblioteca GeoUtils faz uso da seguinte fórmula:

$\Delta\varphi = \ln(\tan(\text{latB} / 2 + \pi / 4) / \tan(\text{latA} / 2 + \pi / 4))$
 $\Delta\text{lon} = \text{abs}(\text{lonA} - \text{lonB})$
 Bearing : $\theta = \text{atan2}(\Delta\text{lon} , \Delta\varphi)$

Sendo:

- ln = logaritmo natural
- se $\Delta\text{lon} > 180^\circ$ então $\Delta\text{lon} = \Delta\text{lon} \pmod{180}$.

3.1.2 Biblioteca Fuzzy Engine

Sazonov et al (2002) desenvolveram uma ferramenta *Open Source*, ou seja, de código livre e aberto, para realizar a implementação computacional da Lógica Fuzzy chamada Fuzzy Engine. Feita em Java, a Fuzzy Engine é um conjunto de classes e métodos que permitem a avaliação de regras Fuzzy apresentadas numa forma linguística. Os próprios autores definem as funcionalidades da ferramenta:

- A ferramenta suporta avaliação de regras Fuzzy na forma simbólica;
- Operações Fuzzy de “E” / “Ou” / “Não”;
- Aninhamento arbitrário de expressões Fuzzy na parte esquerda da regra; Operações múltiplas de atribuição na parte direita de uma regra;
- Limites ou pesos de regras pré definidos ou definidos pelo usuário;
- Operações de atribuição incondicionais;
- Regras ponderadas dinamicamente;
- Avaliação de regras individuais ou múltiplas (Sazonov et al, 2002).

Para usar a biblioteca no aplicativo é necessário colocar no projeto Java o pacote com as classes e fazer a chamada da classe FuzzyEngine e seus métodos dentro do programa.

A biblioteca utiliza para o processo de defuzzificação o método de centróide, ou centro de área, que, conforme já explicado no presente projeto, capítulo 2.6.5 , faz uma espécie de cálculo médio, onde faz um escalonamento do produto das funções membro combinadas por soma. Esse método gera uma exceção caso nenhuma regra seja disparada pelas entradas dadas para a Fuzzy.

Os seguintes passos são necessários no uso da ferramenta:

- Alocação das variáveis linguísticas.
- Definições das funções membro.
- Definições das regras Fuzzy e seus respectivos pesos.
- Registros de todas as variáveis linguísticas.
- Obtenção de um resultado defuzzificado.

3.1.3 Eclipse e Android

Para utilizar a IDE de programação Eclipse para programar para Android, devem ser instalados na máquina utilizada para o desenvolvimento alguns requisitos. Como as aplicações para Android são desenvolvidas em Java e apenas compiladas de maneira que rodem na Máquina Virtual Dalvik, deve ser feita a instalação na máquina de desenvolvimento a JVM ou Máquina Virtual Java e também o Kit de Desenvolvimento de Software Android (SDK Android). Ambos são disponíveis gratuitamente para download.

Além disso, para a IDE Eclipse é necessário instalar o *plug-in* Android Development Tools (ADT). Essa ferramenta auxilia na criação de projetos, *layout* de telas e componentes, e exportação da aplicação assinada ou não assinada em arquivos de extensão apk para distribuição do aplicativo. Além disso, esse *plug-in* faz a instalação da chamada AVD ou *Android Virtual Devices* (Figura 16), responsável pela emulação ou simulação de máquinas Android para testar os aplicativos sem precisar instalá-los em um aparelho móvel. Usando a AVD é possível escolher até mesmo a versão do Android que será emulado.

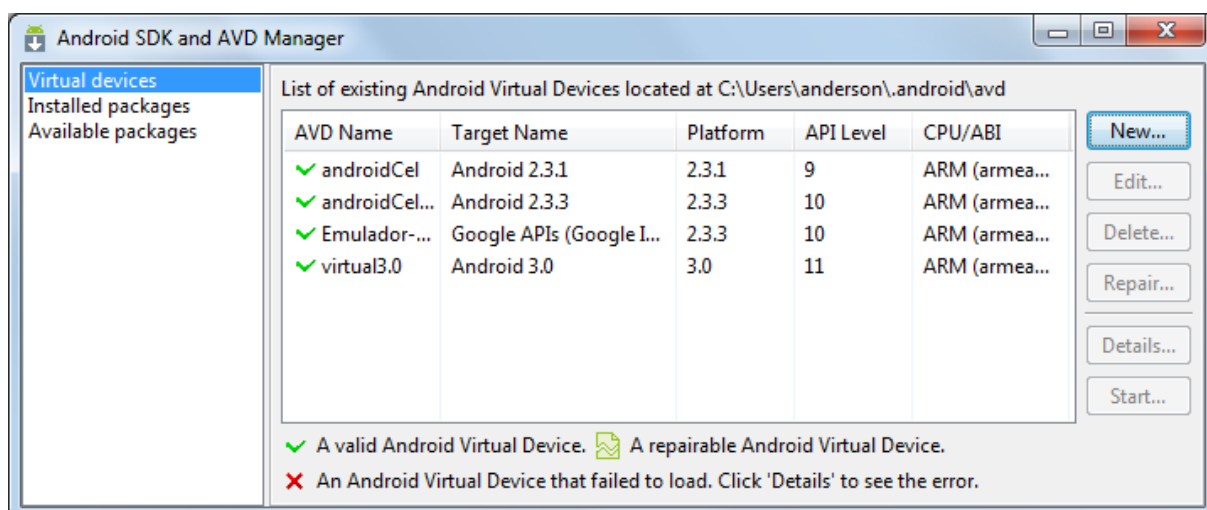


Figura 16 - Exemplo da versão Android SDK and AVD para Eclipse

Fonte: o autor

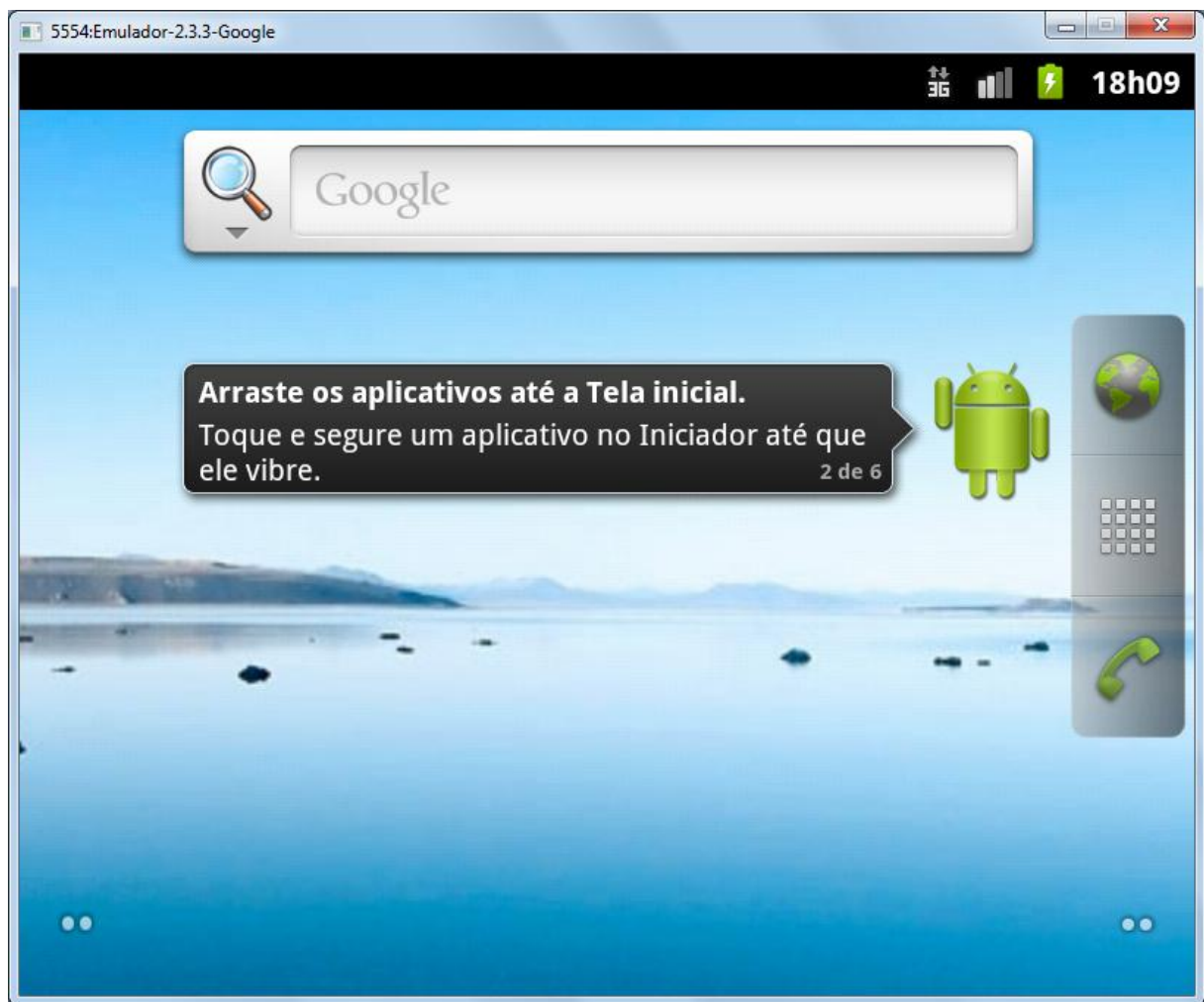


Figura 17 - Máquina Android Emulada em Ambiente Windows
Fonte: o autor

A Figura 17 ilustra uma máquina Android sendo emulada em uma máquina Windows, como uma Máquina Virtual. Quase todas as funcionalidades de um aparelho são mantidas menos as que dependem de hardware, como por exemplo, GPS e *Bluetooth*.

3.1.4 Biblioteca Google Maps

O sistema proposto tem por intenção utilizar ferramentas de fácil interação e que sejam intuitivas para o usuário. A ideia principal é desenvolver um ambiente gráfico para que o usuário assinale em um mapa para o dispositivo os locais onde deseja que o Agrobot percorra. Ou seja, um

software que exija o mínimo possível de esforço cognitivo do usuário e que com poucos cliques uma rota seja traçada.

A solução encontrada foi usar de um serviço oferecido pela empresa Google, através da biblioteca Google Maps, dentro do pacote “com.google.android.maps”. O Google Maps, não faz parte do pacote padrão Open Source da Google e é necessário que o desenvolvedor obtenha uma chave de ativação disponibilizada pela própria empresa para a utilização desse serviço.

Essa biblioteca permite que seja integrado o serviço do Google Maps com a aplicação que está sendo desenvolvida. Sendo assim, é possível fazer o *download*, renderizar, colocar marcadores e outras várias funcionalidades de exibição e controles. Isso é feito através de requisições e manipulação de dados de mapas, feitas de maneira simples por métodos de uma ou várias classes. Como são acessados os dados diretos da Google, a empresa exige que o desenvolvedor faça o registro de sua aplicação, e concordar com os termos de serviço do Google Maps.

O primeiro passo para registrar a aplicação é gerar o MD5 *Fingerprint*, ou seja, gerar um código, como uma “impressão digital”, que será utilizado para assinar a aplicação que será desenvolvida. Para gerar esse código, será utilizado o Java Key and Certificate Management Tool, chamado Keytool. Esta ferramenta fica no diretório %JAVAHOME%/bin ou seja, no diretório chamado bin, dentro da pasta de instalação do Java. Também é necessário gerar o certificado de debug do SDK instalado na máquina. Normalmente, este arquivo fica no caminho “C:\Documents and Settings\<username>\Local Settings\Application Data\Android” e se chama debug.keystore. Para gerar a chave deve ser executada a seguinte instrução:

```
keytool.exe -list -alias androiddebugkey -keystore
"C:\android\debug.keystore" -storepass android -keypass android
```

A Figura 18 demonstra que como foi gerado o MD5 Fingerprint para o desenvolvimento da aplicação proposta nesse trabalho

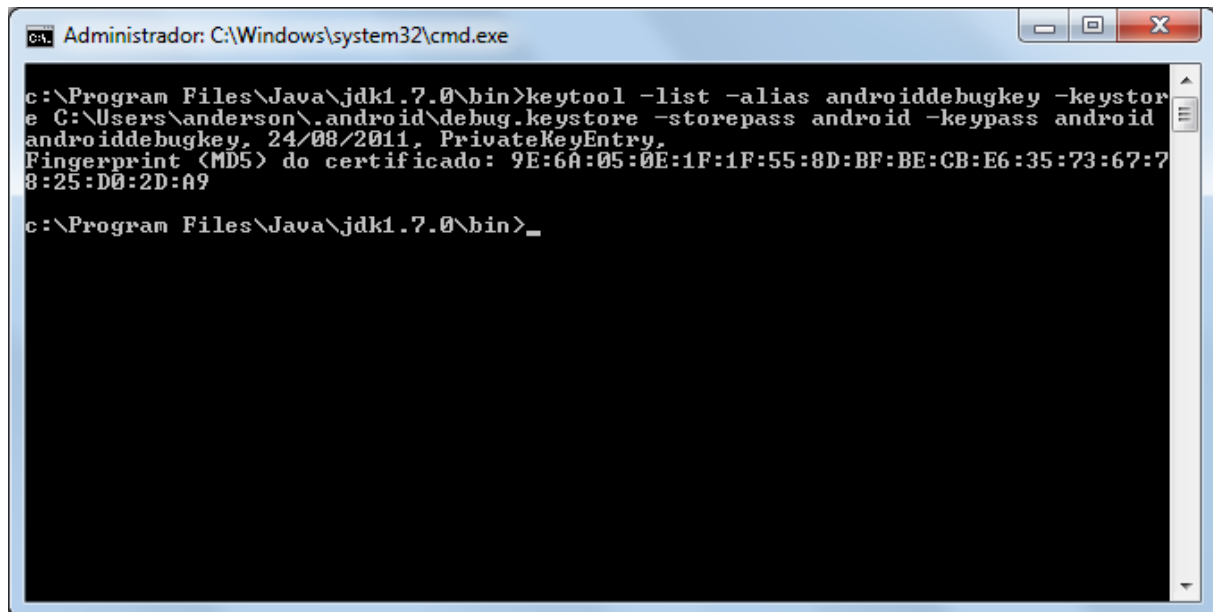


Figura 18 - Obtenção do MD5 Fingerprint para uso dos serviços do Google Maps
Fonte: o autor

Por fim, deve ser acessada uma página⁶ na internet, da empresa Google para o registro e obtenção da Google Maps Key, conforme pode ser visto na Figura 19. No caso do trabalho proposto a chave utilizada foi:

0YWGtc9BewAQU5yBetp3qvKuwQwYBXW8AmEVmA.

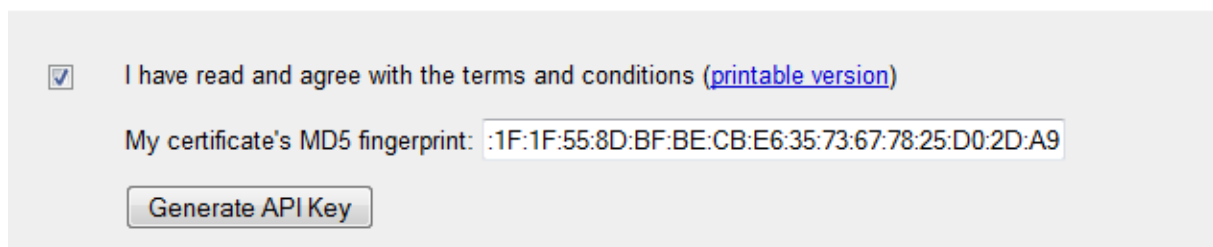


Figura 19 - Página para registro da chave para utilização do Google Maps
Fonte: o autor

Esta chave gerada pelo site, a API Key Maps, é usada no arquivo de layout dentro do projeto Android, o chamado main.xml, localizado na pasta res/layout. Nele foi adicionado a tag <com.google.android.maps.MapView>. Nesta tag foi colocado o parâmetro

⁶ <http://code.google.com/android/maps-api-signup.html>.

android:apiKey com a chave gerada. Isso resulta na exibição do mapa na tela do aparelho.

A programação do Google Maps para Andr id   dividida em tr s partes: os arquivos Java, onde est o os principais m todos e atividades, o arquivo XML de layout da tela, na pasta layout, e por fim o arquivo XML de permiss es, o AndroidManifest.xml, respons vel pela permiss es da aplica  o acessar a internet, informa  es de GPS e no caso tamb m onde fica a chave de ativa  o do Maps.



Figura 20 - O Google Maps, em um dispositivo Android com dois pontos marcados
Fonte: o autor

A Figura 20 ilustra a biblioteca do Google Maps funcionando em um dispositivo Android, com a aplica  o de marcadores no evento *onTouch* do aparelho, ou seja, ao tocar a tela do dispositivo, no mapa. Com essa ferramenta   poss vel a captura das coordenadas geogr ficas do ponto marcado, para cria  o de uma rota. Neste projeto est  sendo dada uma maior

prioridade a facilidade de interação do software e não tanto a acurácia dos pontos marcados.

3.2 MÉTODOS

Para alcançar os objetivos propostos nesse projeto, foram realizados os seguintes procedimentos:

a) Instalação e configuração da IDE de desenvolvimento Eclipse, junto com o conjunto de ferramentas para programação de aplicativos para o sistema operacional Android.

b) Utilização da biblioteca Google Maps e autenticação para utilização do serviço Google Maps da empresa google a fim de criar uma interface gráfica com o usuário, para marcação dos pontos a serem seguidos pelo robô de campo, apenas com um toque na tela do computador embarcado ou celular.

c) Modelagem do problema de correção de rota para aplicação do componente Fuzzy Engine, com a criação das variáveis linguísticas, funções membro e mapa de regras para elaboração do sistema de manobra a ser gerenciado pelo sistema de navegação.

d) Desenvolvimento do sistema computacional, chamado Navigo, ligando a interface gráfica de mapa virtual, feita com a Biblioteca Google Maps (item b acima), com uma interface para navegação, na qual seja demonstrado visualmente, a cada mudança de localização, a correção de rota, dada pelo componente criado baseado na biblioteca Fuzzy Engine (item c acima).

e) Instalação do sistema computacional Navigo (item d acima) no computador embarcado ARM.

f) Realização de saídas em campo, a pé, com o computador embarcado ARM em mãos, módulo de GPS ligado, a fim de simular com o Navigo percursos para o robô.

g) Criação de um arquivo texto para registro de cada mudança de localização (arquivo de *log*), com o ângulo entre o ponto atual com o ponto destino e o resultado de correção dado pelo componente Fuzzy.

h) Validação do resultado obtido pela Fuzzy em comparação com a correção matemática (prova real) a fim de verificar o comportamento da Fuzzy quanto a imprecisão do dispositivos GPS.

4 RESULTADOS E DISCUSSÕES

Neste capítulo serão abordados os resultados obtidos com o desenvolvimento do sistema computacional Navigo e sua utilização. Ainda, discorrerá detalhes de como foram realizados os testes para avaliação do sistema e, por fim, serão demonstrados os resultados obtidos.

4.1 SISTEMA COMPUTACIONAL NAVIGO

O sistema computacional para navegação desenvolvido neste projeto tem como objetivo que o usuário marque em um mapa virtual o ponto que deseja que o robô alcance. A Figura 21 mostra a tela onde o usuário vê onde está no mapa e marca aonde quer chegar. Logo, o sistema deve entender o ponto que o robô está e como início de sua rota e onde deve chegar como fim.



Figura 21 - Tela onde o usuário vê onde está (ponto azul) e marca onde quer chegar (estrela)
Fonte: o autor

No percurso pode haver obstáculos, tais quais pedras, arbustos, galhos, buracos, os quais poderão atrapalhar o percurso do robô e contribuir para que ele saia da rota previamente estipulada. Com isso, o software deve, dentro de um intervalo de tempo parametrizado, capturar qual a posição então atual do robô e fazer a correção da rota para que chegue ao alvo.

A menor distância entre dois pontos em um plano é uma reta. Partindo dessa máxima, o método utilizado, traçará uma reta entre o ponto inicial, ou seja, de partida do robô, até o alvo que deve alcançar, marcado pelo usuário. A lógica Fuzzy entra nesse contexto para fazer a correção dos desvios da rota, ou seja, para ajustar o ângulo de giro das rodas e tentar manter sempre o robô dentro da primeira reta traçada.

O GPS do dispositivo retorna a informação do ponto onde o robô está, em latitude e longitude, assim como o ponto de alvo dado pelo usuário ao marcar seu destino no mapa. A primeira localização e o alvo marcado como destino são armazenados, respectivamente, como ponto de partida e destino. A distância entre o ponto de partida e o ponto aonde se quer chegar geometricamente pode ser considerada a hipotenusa de um triângulo. Utilizando a biblioteca GeoUtils, é possível obter o ângulo entre os dois pontos, utilizando a função *bearing*, já explicada anteriormente nesse trabalho. Aplicando o teorema de Pitágoras é obtida a distância em X e em Y, sendo elas o cateto adjacente e oposto desse triângulo formado. A construção deste triângulo retângulo é demonstrada na Figura 22.

A regra de inferência da Fuzzy é criada dividindo o valor de X (cateto adjacente) e em Y (cateto oposto) em cinco faixas de termos linguísticos. Para X, terá: Muito a esquerda, esquerda, neutro, direita e muito para direita. Para Y, terá: Muito abaixo, abaixo, neutro, acima, muito acima. Isso é demonstrado na Figura 23 e Figura 24.

Sendo assim, com essas variáveis linguísticas já determinadas, o gráfico da Fuzzy a ser usada está na Figura 25 e Figura 26. Nessas imagens, foram gerados gráficos de valores hipotéticos, sendo de 20 metros o cateto oposto e adjacente do triângulo gerado.



Figura 22 - Construção do primeiro triângulo retângulo
Fonte: o autor



Figura 23 - Termos linguísticos para X
Fonte: o autor



Figura 24 - Termos linguísticos para Y
Fonte: o autor

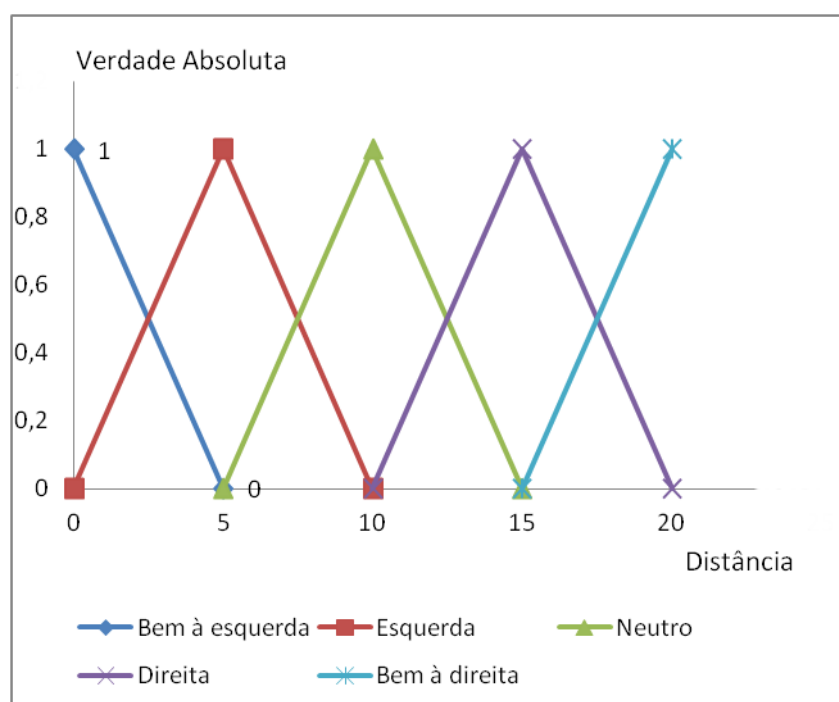


Figura 25 - Gráfico dos termos linguísticos da Fuzzy em X (cateto adjacente)
Fonte: o autor

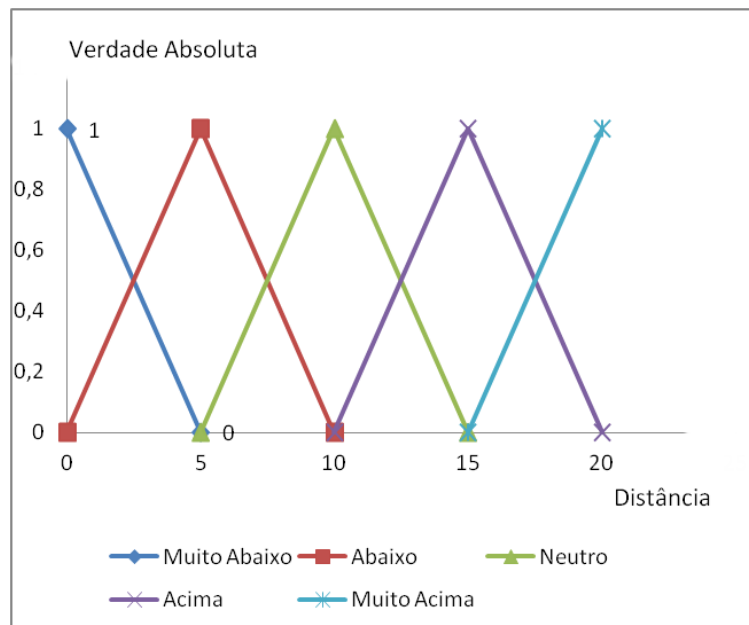


Figura 26 - Gráfico dos termos linguísticos da Fuzzy em Y (cateto oposto)

Fonte: o autor

Na Figura 27 e na Figura 28 são ilustrados dois casos de desvio de rotas, onde o dispositivo se encarrega de gerar os novos triângulos, os quais os catetos servirão de parâmetros de entrada para a utilização da lógica Fuzzy.



Figura 27 - Desvio de rota 1

Fonte: o autor



Figura 28 - Desvio de rota 2
Fonte: o autor

Fazendo uma correlação entre X e Y é possível montar o Quadro 4, com uma ação para cada possibilidade do robô estar em um cruzamento entre os valores em X e Y.

Quadro 4 - Regras usadas para Fuzzy

		Posição do veículo em X				
		Muito Esquerda	Esquerda	Neutro	Direita	Muito Direita
Posição do veículo em Y	Muito Abaixo	Neutro	Direita	Bem à direita	Bem à direita	Bem à direita
	Abaixo	Esquerda	Neutro	Direita	Bem à direita	Bem à direita
	Neutro	Bem à esquerda	Esquerda	Neutro	Direita	Bem à direita
	Acima	Bem à esquerda	Bem à esquerda	Esquerda	Neutro	Direita
	Muito Acima	Bem à esquerda	Bem à esquerda	Bem à esquerda	Esquerda	Neutro

Fonte: o autor

Ou seja, as variáveis linguísticas de saída da Fuzzy serão:

- Neutro: não faça nada
- Esquerda: vire um pouco para esquerda
- Direita: vire um pouco para direita
- Bem à esquerda: vire bem à esquerda
- Bem à direita: vire bem à direita

Para cada variável linguística de saída foi determinada uma faixa de valores para que indiquem ao robô o quanto a roda do robô de campo deverá ser girada, demonstrada na Tabela 2 e Figura 29.

Tabela 2 - Tabela de valores para cada termo linguístico

Variável Linguística	De	Até
Bem à Esquerda	-30	-15
Esquerda	-30	0
Neutro	-15	15
Direita	0	30
Bem à Direita	15	30

Fonte: o autor

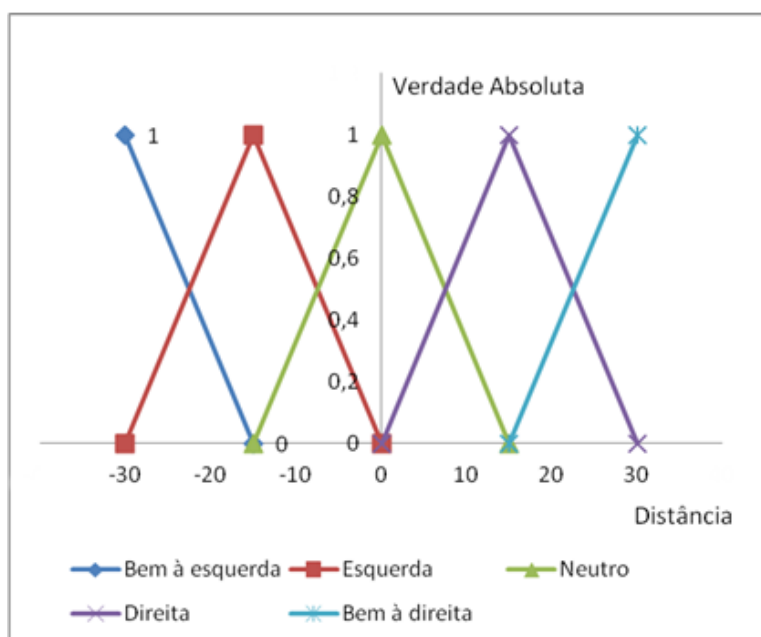


Figura 29 - Gráfico dos termos linguísticos de saída da Fuzzy
Fonte: o autor

Ao deslocar-se do ponto de partida, a cada mudança de localização apontada pelo dispositivo GPS, o programa computacional se encarregará de fazer a correção de algum possível desvio da rota traçada. Para tanto, é feito um novo triângulo, entre o ponto inicial de partida e o ponto atual do robô. Esse triângulo é construído pelo mesmo procedimento do triângulo principal que gerou a matriz base da Fuzzy, onde é obtido o cateto oposto e adjacente do novo triângulo, pelo ângulo entre o primeiro ponto, ou seja, ponto de partida, com o ponto atual. A distância entre eles será então a nova hipotenusa.

Os valores dos novos catetos obtidos, oposto e adjacente, do novo triângulo são usados como parâmetros de entrada para o cálculo na Fuzzy, que por sua vez retornará o valor numérico do ângulo de correção para voltar a rota estabelecida.

4.2 RESULTADOS E AVALIAÇÃO DO APLICATIVO

Para avaliação do sistema computacional Navigo foram feitas saídas em campo, com o computador ARM em mãos, módulo de GPS ligado, executando o Navigo e observando os resultados da correção da rota exibidos na tela do computador. Simultaneamente, o Navigo também se encarregará de fazer as coletas dos registros em um arquivo texto com a latitude, longitude, ângulo com o alvo e o resultado obtido com a aplicação da lógica Fuzzy para a correção da rota, conforme mostra a Tabela 3.

O autor deste trabalho, saiu na rua seguindo em linha reta com o dispositivo em mãos, testando fazer a pé a rota estipulada, simulando um possível comportamento do robô de campo, mantendo uma velocidade constante em direção ao ponto marcado como destino.

Tabela 3 - Tabela com o formato do arquivo texto gerado para validação dos resultados

Horário	Latitude	Longitude	Ângulo com o Alvo	Saída da Fuzzy
17:44:34	-25,1146	-50,1544	35	0,023473
17:44:35	-25,1146	-50,1544	34	0,069928
17:44:36	-25,1146	-50,1544	35	0,00192

Horário	Latitude	Longitude	Ângulo com o Alvo	Saída da Fuzzy
17:44:37	-25,1146	-50,1544	36	-0,14322
17:44:38	-25,1146	-50,1544	36	-0,15164
17:44:39	-25,1146	-50,1544	36	-0,18241
17:44:40	-25,1146	-50,1544	35	0,017843

Fonte: o autor

Alguns obstáculos, que são comuns no campo, como pedras, buracos, plantas, provocariam alguns desvios no percurso. Estes desvios foram simulados, algumas vezes de forma mais acentuada e outras de forma mais sutil, sendo registrado abaixo a realização, detalhamentos e resultado de três principais coletas.

4.2.1 Primeira coleta de resultados

Na primeira coleta de dados foi realizado um percurso de aproximadamente 253 metros, com duas saídas da rota um pouco mais acentuadas, conforme é demonstrada na Figura 30.

Na Figura 31 são mostradas as capturas de tela do aplicativo indicando a correção da rota. As telas são respectivamente marcadas pelos números que representam o lugar no mapa onde elas foram realizadas. Nela pode-se perceber que a indicação de correção que o aplicativo gera visualmente coincide com o tamanho do desvio na rota. Por exemplo, no momento “1” a saída da rota foi para a direita e o aplicativo indicou uma correção de 10 graus a esquerda. No momento “2”, simulou-se uma saída consideravelmente grande da rota ideal, fazendo com que o aplicativo sugerisse a correção de 18 graus para a direita a fim de voltar a rota inicial. Por fim, no momento “3”, os desvios de rota foram muito pequenos, e o aplicativo se comportou indicando correções pequenas, de até cinco graus, ou simplesmente, indicou para seguir o rumo em frente.

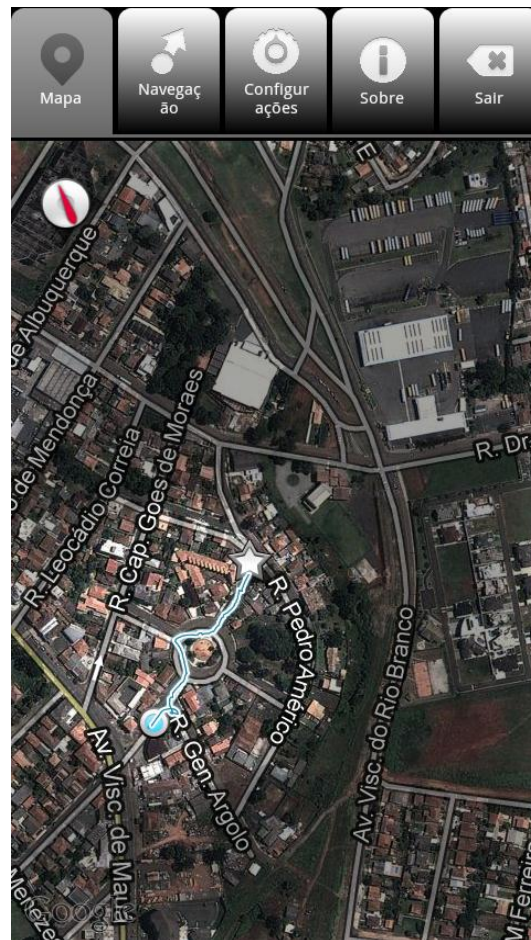


Figura 30 - Rota seguida na primeira coleta de dados
Fonte: o autor



Figura 31 - Capturas de telas: demonstração no mapa do lugar onde foi realizada a primeira coleta e os resultados obtidos pelo Navigo durante o percurso
Fonte: o autor

É possível perceber pelos dados coletados que mesmo havendo a alteração de ângulo do ponto atual com o ponto de alvo não há

grande variação no resultado da Fuzzy. Isso acarretará movimentos mais suaves, evitando assim que o protótipo ande em ziguezague. Isto é visível no gráfico demonstrado na Figura 32.

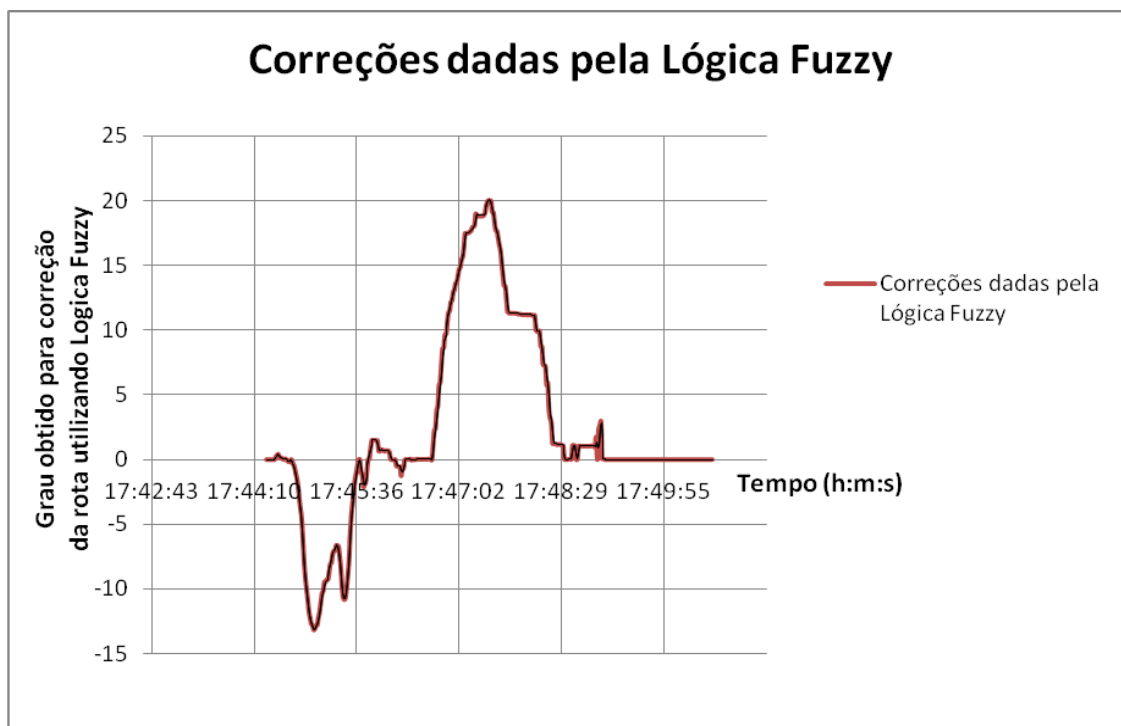


Figura 32 - Gráfico das correções dadas pela Fuzzy por tempo
Fonte: o autor

Para fazer a avaliação do Navigo, nos dados coletados foi aplicado a correção matemática, onde o grau de correção se dará apenas diminuindo o grau do primeiro ponto e o grau do ponto atual em relação ao alvo. Ou seja, a correção será igual à subtração do ângulo do ponto atual em relação ao alvo e do ponto inicial em relação ao alvo. Dessa maneira foram obtidos os resultados demonstrados na Figura 33.

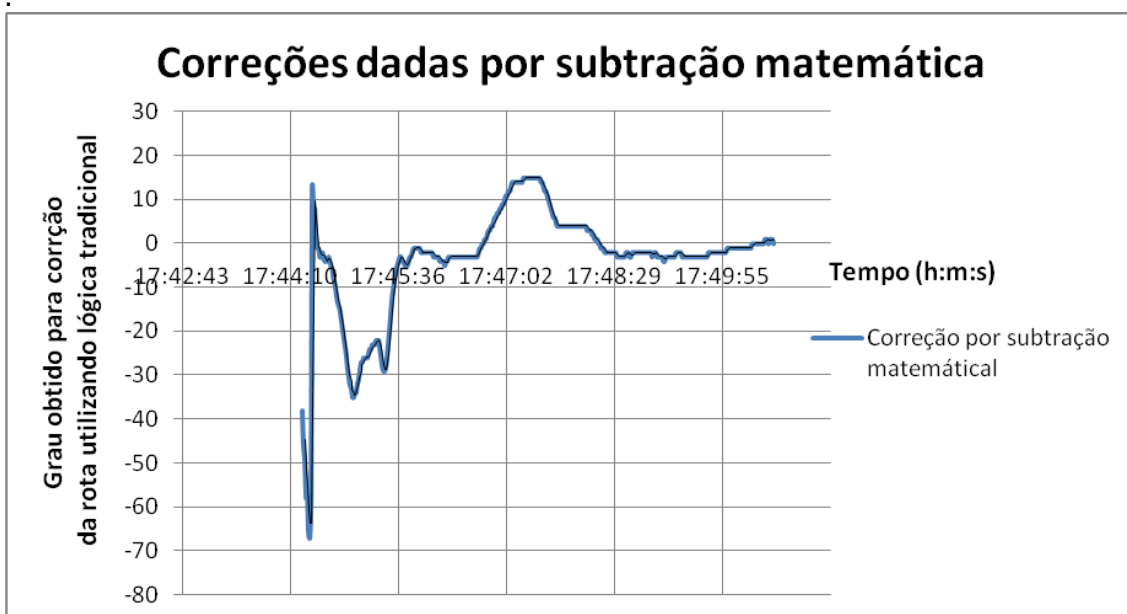


Figura 33 - Gráfico das correções dadas por subtração matemática
Fonte: o autor

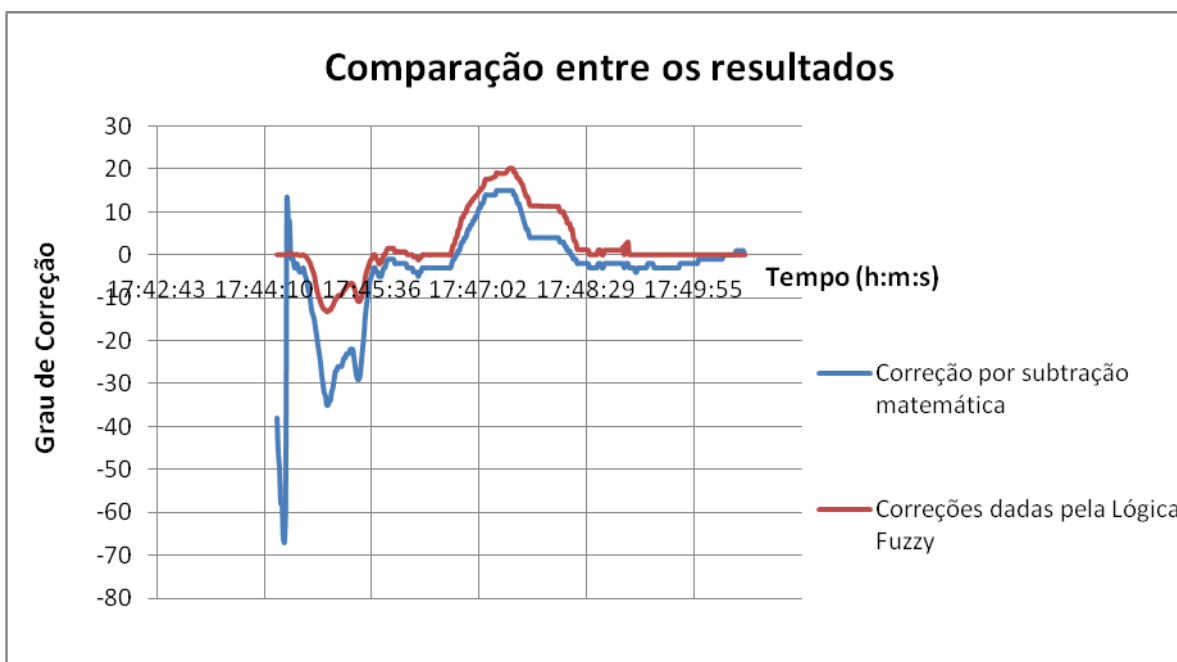


Figura 34 - Gráfico comparativo entre a correção da rota por Lógica Fuzzy e por subtração matemática
Fonte: o autor

Nesta primeira coleta é possível perceber que a correção obtida utilizando a lógica Fuzzy se comporta de maneira mais amena do que a correção por subtração apenas. Na Figura 34 é demonstrado um gráfico comparativo entre os dois resultados, por lógica Fuzzy em vermelho e por subtração dos ângulos em azul. Essa figura demonstra que os resultados por subtração dos ângulos oscila, não mantendo muito uma linearidade nos

resultados. Isso se deve a falta de precisão do dispositivo GPS, pois seu resultado, dentro de um pequeno intervalo de tempo, oscila de -70 graus para esquerda até 10 para direita.

No traço vermelho da Fuzzy também é possível notar que os graus de correção sugeridos são consideravelmente menores do que a lógica tradicional, o que mandaria menos instruções de giro para o controlador que atuará como volante do robô de campo.

4.2.2 Segunda coleta de resultados

Na segunda coleta de dados, foi simulada uma rota com desvios bastante acentuados para avaliar o comportamento do aplicativo. No total foram percorridos aproximadamente 350 metros. A rota é demonstrada na Figura 35.



Figura 35 - Rota seguida na segunda coleta de dados

Fonte: o autor

Na segunda rota a Fuzzy também se comportou como o esperado. Novamente foram feitas coletas de dados e a comparação dos resultados da Fuzzy com os resultados da lógica tradicional, conforme o gráfico da Figura 36.

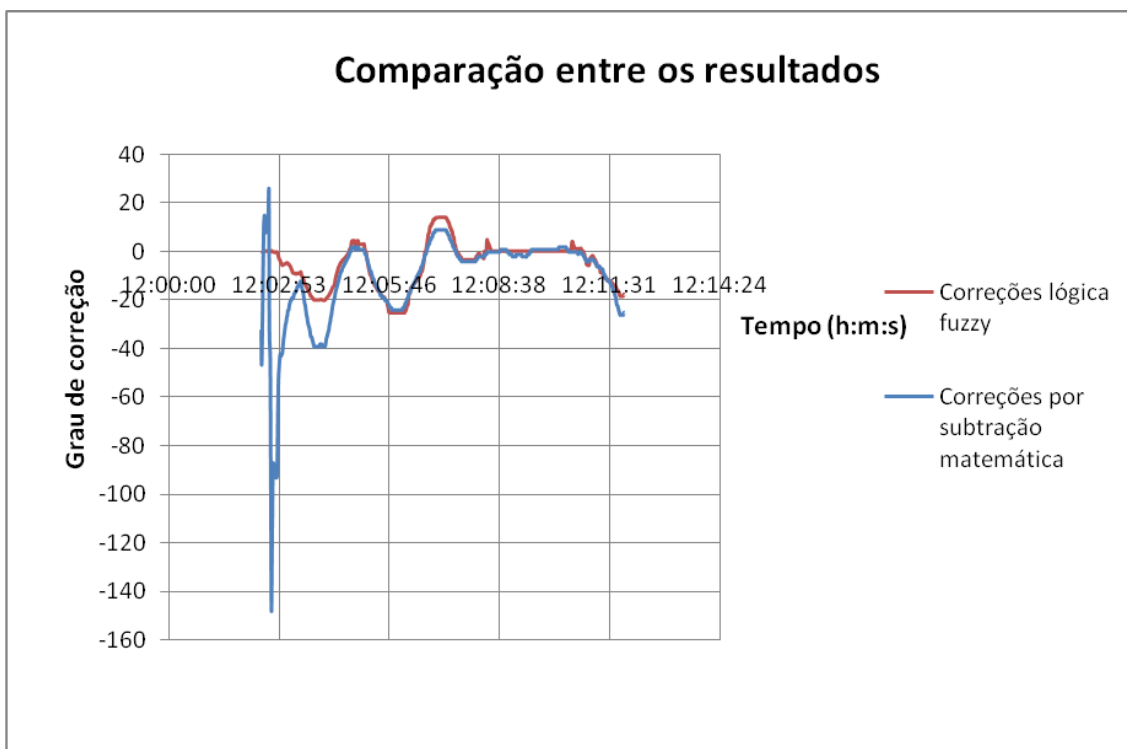


Figura 36 - Gráfico de comparação entre a correção por lógica fuzzy e por subtração matemática dos ângulos na segunda coleta de dados

Fonte: o autor

Semelhante ao comportamento da primeira coleta, novamente a correção sugerida pela lógica Fuzzy demonstrou-se mais amena que a simples subtração matemática. Os graus de correção sugeridos pela Fuzzy foram mais lineares, repetitivos e menos discrepantes entre si. Isso aponta novamente que a Fuzzy conseguiu trabalhar melhor com os erros de localização do dispositivo GPS.

No cenário de automação agrícola, a economia de bateria, obtida pelo fato de ser feito menos instruções para atuações mecânicas (de giro da roda) pode ser um diferencial, permitindo ampliação da autonomia da bateria em campo.



Figura 37 - Capturas de telas e o local onde foram realizadas (Coleta 2)
Fonte: o autor

Na Figura 37 é possível verificar que novamente as correções apontadas pelo dispositivos são proporcionais e visualmente coerentes com a quantia de desvio que foi aplicado. Analisando a imagem em termos linguísticos, a ação sugerida pelo controlador Fuzzy segue o padrão abaixo:

Figura 37 - Número 1

SE desvio da rota em X (cateto adjacente) é pequeno

E desvio da rota em Y (cateto oposto) é pequeno

ENTÃO siga em frente a rota

Figura 37 - Número 4

SE desvio da rota em X (cateto adjacente) é grande

E desvio da rota em Y (cateto oposto) é grande

ENTÃO vire em 25 graus para a esquerda

O cateto adjacente e cateto oposto, mencionados acima, são dos triângulos gerados a cada alteração da localização e, conforme explicado anteriormente neste trabalho, na modelagem para Fuzzy que foi criada serão as variáveis de entradas do sistema de correção de rota Fuzzy.

Com isso é possível explicar o funcionamento da Fuzzy, onde medidas dos catetos (variáveis reais) passam por um processo de “Fuzzyficação” (fatos) a fim de serem representadas por variáveis e termos linguísticos Fuzzy. Através de um sistema de inferência, a partir da base de regras dinamicamente é produzido um valor de saída, representado por uma variável Fuzzy de saída (conclusões), que informará ao controlador do robô qual a ação deverá ser realizada para corrigir a rota.

4.2.3 Terceira coleta de resultados

Na terceira coleta, Figura 38, foi feito um trajeto considerando todas as sugestões de acerto do desvio do aplicativo, ou seja, a cada sugestão de correção de rota, foi dado um passo a direita ou a esquerda conforme a saída vista

em tela buscando sempre manter a instrução "siga em frente", sem fazer nenhum desvio muito amplo.

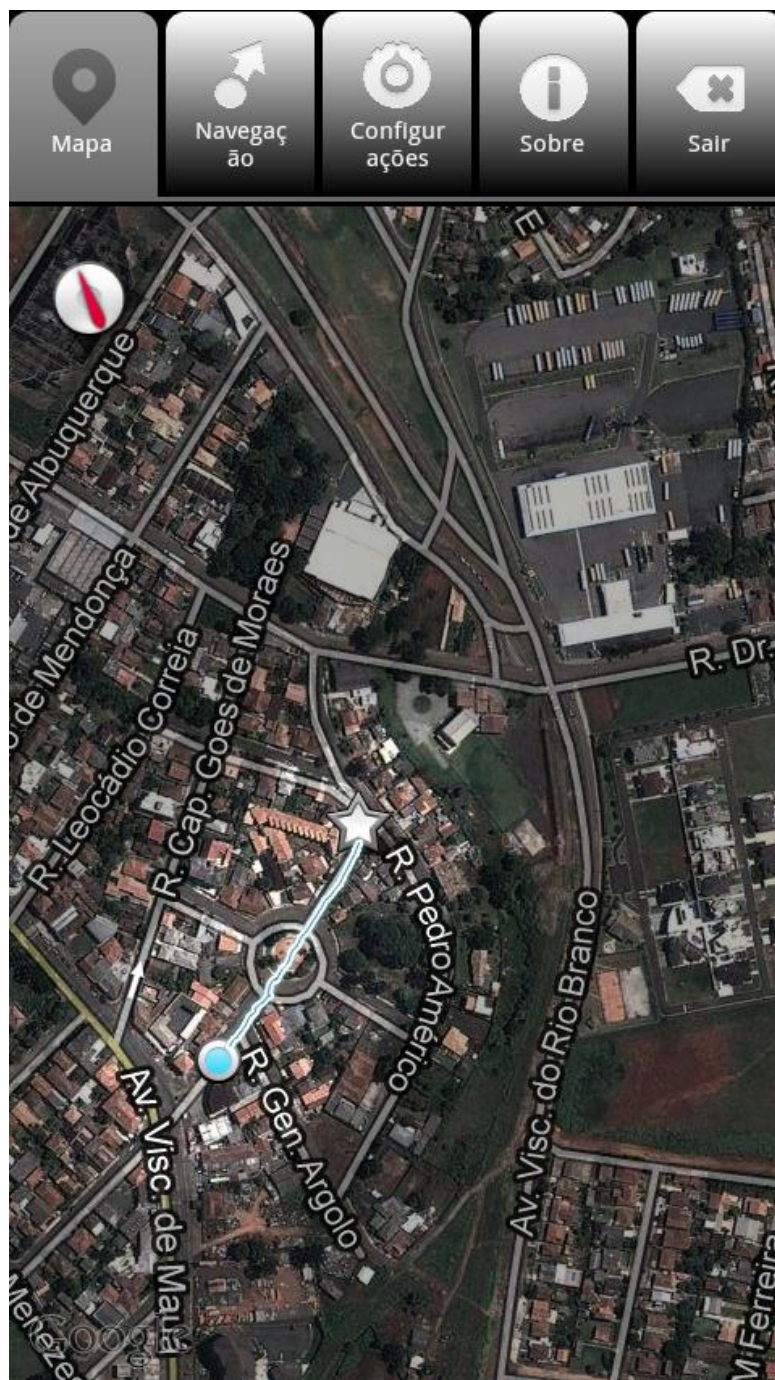


Figura 38 - Rota seguida na terceira coleta de dados
Fonte: o autor

Desta vez, foi mantido um trajeto onde correções oscilavam entre no máximo 3 graus para a direita ou esquerda, nunca ultrapassando isso e sempre buscando a correção zero. No total, foram percorridos aproximadamente 200 metros. Ao fazer a comparação entre a correção empregando lógica Fuzzy e por subtração dos ângulos é possível perceber que a diferença entre elas é sutil, sendo

maior no início do percurso o qual é mais susceptível ao erro do dispositivo GPS. Novamente o alinhamento de rota com Fuzzy mostrou-se mais eficiente ao contornar o erro que aconteceu.

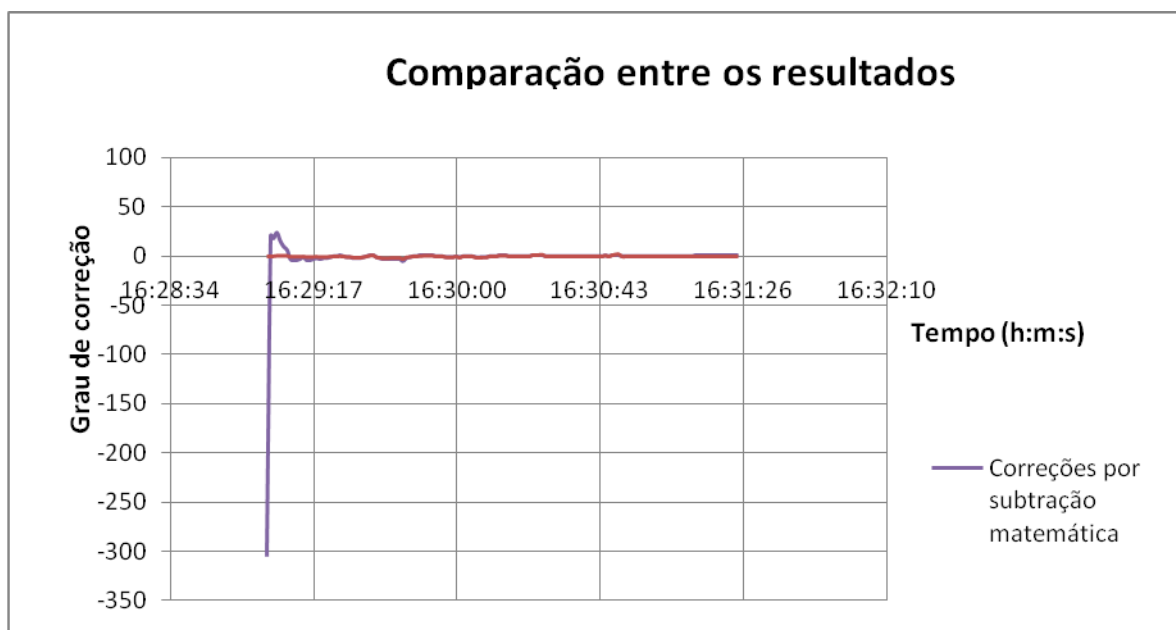


Figura 39 - Gráfico de comparação entre a lógica fuzzy e subtração matemática na terceira coleta de dados

Fonte: o autor

Na Figura 39, a linha em vermelho demonstra as sugestões de correção por subtração dos ângulos, onde fica claro que no início do trajeto houve instruções erradas as quais fariam o controlador do volante do robô corrigir um erro de rota que não existia, uma vez que sempre se buscou seguir em linha reta e o erro acusado no gráfico aparenta ser consideravelmente grande.

Estes graus errados obtidos pelo aplicativo no início do trajeto devem-se ao fato do GPS possuir um erro de precisão. Fazendo uma coleta das informações dadas pelo dispositivo é possível perceber que mesmo o veículo estando parado, a localização obtida oscila. Isto é demonstrado na Tabela 4, que possui um trecho de um arquivo no qual feita a coleta da latitude e longitude dada pelo GPS, com dispositivo em inércia.

Há uma certa diferença entre os valores obtidos que, notoriamente, deveriam ser sempre iguais. Sendo assim, a localização dada pelo GPS não é um ponto exato e sim uma dispersão.

Tabela 4 - Leituras realizadas com o dispositivo parado

Tempo	Latitude	Longitude
12:02:25	-25,11482092	-50,1547061
12:02:26	-25,11482087	-50,15470501
12:02:29	-25,11481785	-50,15470313
12:02:30	-25,11481726	-50,15470298
12:02:31	-25,11481694	-50,1547028

Fonte: o autor

Assim que o usuário do GPS locomover-se e afastar-se do ponto de início do trajeto, essa dispersão também locomoverá, diminuindo assim o erro. Isso é percebido nos gráficos gerados pelas três coletas, principalmente nas correções obtidas pela subtração matemática.

5 CONCLUSÕES

Os resultados obtidos no presente trabalho permitiram as seguintes conclusões:

- O modelo Fuzzy de controle mostrou-se eficiente no processo de correção da rota principalmente comparado ao processo de correção o qual leva em consideração apenas o ângulo atual com o ângulo do início do trajeto.
- As correções obtidas pela metodologia Fuzzy demonstraram-se mais adaptáveis aos erros do dispositivo de GPS.
- As correções obtidas pela metodologia Fuzzy também se mostram mais brandas, ou seja, suaves. Com isso o caminho realizado com a correção por Fuzzy se fará com curvas suaves e não por segmentos de reta.
- Pelo fato do Agrobot ser um projeto de um veículo que funcionará a bateria, quanto menos ações de virada de volante e menor o percurso percorrido, melhor será a otimização do uso da bateria, ou seja, a bateria durará mais. No modelo comparativo utilizado nesse trabalho, as correções se mostraram mais bruscas, instruindo a um percurso em ziguezague.

O presente trabalho tem como limitação o uso apenas da subtração matemática dos ângulos inicial e atual do dispositivo com o alvo marcado de destino, a fim de demonstrar que o uso da Fuzzy é uma solução interessante para a suavização do percurso e economia de energia. Não foram realizados comparativos com o uso da Fuzzy e outras abordagens que possam existir na literatura, fazendo ou não o uso de algum algoritmo de inteligência computacional.

Como trabalho futuro pode-se citar a integração do sistema computacional desenvolvido com o robô que está ainda sendo desenvolvido no Laboratório de Automação e Robótica da Universidade Estadual de Ponta Grossa-PR.

Pelo fato do projeto Agrobot se tratar de um modelo *open source*, o aplicativo desenvolvido e o seu código fonte encontram-se disponíveis para a comunidade fazer o *download* na página na web: <http://code.google.com/p/agrobot-uepg/>

REFERÊNCIAS BIBLIOGRÁFICAS

ALETAN, S.O. **An overview of RISC architecture**. In *Proceedings of the 1992 ACM/SIGAPP Symposium on Applied computing: technological challenges of the 1990's* (SAC '92), Hal Berghel, Ed Deaton, George Hedrick, David Roach, and Roger Wainwright (Eds.). ACM, New York, NY, USA, 1992.

ANDERSEN, J.; SEIBEL, E. **Real-time Hazard detection via machine vision for wearable low vision aids**. IEEE Fifth International Symposium on Wearable Computers (ISWC'01), Outubro, Zurich, Suíça, 2001.

ASTRAND, B.; BAERVELDT, A.. An Agricultural Mobile Robot with Vision-Based Perception for Mechanical Weed Control. **Autonomous Robots**. Halmstad, Suécia. v. 13, n. 1, p. 21-35, 2002.

ATIQUEZZAMAN, M. Coarse-to-Fine Search Technique to Detect Circles in Images. **International Journal of Advanced Manufacturing Technology**, Vol. 15, 1999

BARROS, L. C. de. **Teoria Fuzzy x Biomatemática**. Campinas: IMECC- UNICAMP, 2002.

BATES, S. **Case Study: Advanced Risc Machines, The Future Today**. The Institution of Electrical Engineers. Publicado em IEE, Savoy Place, Londres WCPR OBL, UK, 1996.

BILOBROVEC, M.;MARÇAL, R.F.M. ;KOVALESKI, J.L. **Implementação de um sistema de controle inteligente utilizando a Lógica Fuzzy**. In: *SIMPÓSIO DE ENGENHARIA DE PRODUÇÃO*, 11p , Bauru-SP, 2004.

BLACKMORE, B. S.. **New concepts in agricultural automation**. HGCA conference – Stoneleigh Park, Kenilworth, Warwickshire, UK, 28 e 29 de Outubro, 2009

CABRAL, R. B. **Adaptação Computacional de Sistemas de Inferência Difusos: um caso aplicado**. Dissertação de Mestrado. Florianópolis, 1994.

CORCOLL-SPINA, C. O. **Lógica fuzzy: reflexões que contribuem para a questão da subjetividade na construção do conhecimento matemático**. 2010. Tese de doutorado - Faculdade de Educação, Universidade de São Paulo, São Paulo, 2010.
Disponível em: <<http://www.teses.usp.br/teses/disponiveis/48/48134/tde-21012011-104236/>>. Acesso em: 23 fev 2012.

DIAS, A. H. **Sistema especialista para automação do processo de aeração de grãos de milho em pequenas propriedades**. Tese de doutorado. FCA-UNESP. Botucatu. São Paulo, 2007. 70 p.

ESPONDA, M. **The RISC Concept - A Survey of Implementations**. Technical Report , Freie Universität Berlin, Setembro 1991

FURBER, S. The Future of Computer Technology and its Implications for theComputer Industry, **The Computer Journal**, Vol. 51 No. 6, 2008.

GAJSKI, D.D.; VAHID, F. **Specification and design of embedded hardware-software systems**, Design & Test of Computers, IEEE , vol.12, no.1, pp.53-67, 1995

GOOGLE. **Android developers**. Disponível em: <<http://developer.android.com/>>, acesso em: 10 jan 2012

GOMIDE, F. A. C.; GUDWIN, R. R.; TANSCHKEIT, A. **Conceitos fundamentais da teoria de conjuntos fuzzy, Lógica fuzzy e aplicações**. In: *Sixth International Fuzzy Systems Association World Congress/ Tutorials* - IFSA95,pages 01 – 38., Julho 1995.

GUERRA, S. S. C. **Sistema de navegação para veículo autônomo utilizando lógica difusa**. Dissertação de Mestrado em Processamento de Sinais de Instrumentação - Escola de Engenharia de São Carlos, Universidade de São Paulo, São Carlos, 2009.

Disponível em: <<http://www.teses.usp.br/teses/disponiveis/18/18152/tde-09092009-082215/>>. Acesso em: 12 jul 2012 .

HASEGAWA, J. K; GALO, M; MONICO, J. F. G.; IMAI, N. N. **Planejamento Logístico de Rotas para Sistema de Navegação Apoiado por GPS**. In: COBRAC - CONGRESSO BRASILEIRO DE CADASTRO TÉCNICO MULTIFINALITÁRIO, Santa Catarina-SC. 2000.

HASEGAWA, J. K.; GALO, M.; MONICO, J. F. G. ,IMAI, N. N. **Sistema de Localização e Navegação Apoiado por GPS**. In: CONGRESSO BRASILEIRO DE CARTOGRAFIA, Rio de Janeiro, 1999.

HILL, L.L. **Georeferencing: The Geographical Associations of Information**. Cambridge, MA: MIT Press, 2006.

KAYTON, M. **Avionics Navegation Systems**. Wiley, 1969.

LAIRD EVANS, J., STEVEN, A. Measures of graphical complexity for navigation and route guidance display. **Displays Journal**, vol. 17. 1997, p. 89 –93.

LECHETA, R. **Aprenda a criar aplicações para dispositivos móveis com o Andróid SDK**. São Paulo, Novatec, 2009.

MARTINS, N. A. **Sistemas Microcontrolados**. São Paulo, editora Novatec, 2005.

MAZIERO, C. A. **Sistemas Operacionais - Conceitos Básicos**. Acervo da PUC-PR, 2006. Disponível em: <<http://www.san.uri.br/~pbetencourt/II/SO.PDF>>. Acesso em: 15/04/2012

MENDEL, J. M. **Fuzzy Logic Systems for Engineering : A Tutorial**, *IEEE Proc.*, Vol. 83, pp. 345-377, Março 1995.

MEIER, R. **Professional Android application development**. Wiley Publishing, Indianapolis, USA, 2009.

MONTA, M. N.; KONDO, Y.; SHIBANO Y. **Agricultural Robot in Grape Production System**. In Proceedings of IEEE International Conference on Robotics and Automation. Nagoya, Japan. p. 2504-2509, Maio, 1995.

MORITA, T. **Grading of the Map Functions in Navigation System**. In: 16TH INTERNACIONAL CONFERENCE., 1993. Cologne – Koln. International Cartographic Association. 1993.

MURDOCCA, M. J.; HEURING, V. P. **Introdução a Arquitetura de Computadores**, Editora Campus, Rio de Janeiro, 1999

OLIVEIRA, C. **Dicionário Cartográfico**. 4ª ed. Rio de Janeiro: IBGE, 1993

OLSSON, S. ; AKESSON, P. **“Distributed Mobile Computer Vision and Applications on the Android Platform.”** Dissertação de Mestrado. Lund University, Suécia. 2009.

PALLET, E.H.J. **Automatic Flight Control**. Blackwell Science Ltda, 1993

PATTERSON, D. Reduced Instruction Set Computers, **Communications of the ACM**, Vol. 28, No.1, Janeiro 1985.

PUGLIESI, E. A. **Desenvolvimento de mapa dinâmico para sistema de navegação terrestre**. Dissertação de Mestrado. Universidade Estadual Paulista. Faculdade de Ciências e Tecnologia. São Paulo, 2002

RITCHIE, D.M.; THOMPSON K. The UNIX time-sharing system. **Commun. ACM** 17, 7 .Julho 1974.

RYZHYK, L. **The ARM Architecture**. Chicago University, Illinois, E.U.A. 2006.
Disponível em: <<http://www.cse.unsw.edu.au/~cs9244/06/seminars/08-leonidr.pdf>>.
Acesso em: 14 jun 2011.

SANDRI, S.; CORREA, C. - **Lógica Nebulosa**. V Escola de Redes Neurais, Conselho Nacional de Redes Neurais. pp. c073-c090, 19 de julho. ITA, São José dos Campos – SP, 1999.

SAZONOV, E.; KLINKHACHORN, P.; GANGARAO, H. S; HALABE, U. B. Fuzzy logic expert system for automated damage detection from changes in strain energy mode shapes. **Nondestructive Testing and Evaluation** Vol. 18, Iss. 1, 2010

SCHMIDT, R. M., LAGES, W. F. **Identificação Visual para Navegação de Robôs Móveis Utilizando um Controlador Fuzzy Sintonizado por Otimização Numérica**. XIII Congresso Brasileiro de Automática, Florianópolis - SC, Setembro, 2000

SHARSHUNOV, S. G. **Functional Tests for RISC-microprocessors**. In *Autom. Remote Control* 65, 11 Novembro 2004

SOUSA, C. A. **Teoria de conjuntos fuzzy e regressão logística na tomada de decisão para realização de cintilografia das paratiróides**. Dissertação Mestrado em Saúde Pública — Universidade de São Paulo, São Carlos, 2007.

SOUSA, R. V. **Robô agrícola móvel (RAM): uma arquitetura baseada em comportamentos hierárquicos e difusos para sistemas autônomos de guiagem e navegação**. 2007. Tese (Doutorado em Engenharia Mecânica) - Escola de Engenharia de São Carlos, Universidade de São Paulo, São Carlos, 2007. Disponível em: <<http://www.teses.usp.br/teses/disponiveis/18/18135/tde-20102010-151246/>>. Acesso em: 19 de agosto de 2012.

TSCHIEDEL, M.; FERREIRA, M.F. **Introdução à agricultura de precisão: Conceitos e vantagens**. *Ci. Rural*, 32:159-163, 2002.

VON HIPPEL, E; VON KROGH, G. Open source software and the "private-collective" innovation model: issues for organization science. **Organization Science**, v. 14, n. 2, 2003.

VICEDO, J. L.; GÓMEZ, J. Georeferencing: The geographic associations of information. **Journal of the American Society for Information Science and Technology**, 2007.

ZADEH, L. A.. **Fuzzy sets**. Fuzzy Sets, Information and Control, 8:338 – 353, 1965.

ZHANG, Q.. **Design of a Generic Fuzzy Controller for Electrohydraulic Steering**. Proceedings of the American Control Conference. Arlington, VA, 25-27 Junho, 2001.

GLOSSÁRIO

Alguns termos são comumente utilizados em sistemas de navegação, entretanto, nem sempre seu real significado é completamente entendido pelo usuário. Alguns termos retirados do Dicionário Cartográfico (OLIVEIRA, 1993) utilizados no proposto trabalho, são descritos para melhor entendimento do leitor.

Orientação	O ângulo horizontal de um determinado ponto medido na direção dos ponteiros de um relógio, a partir de um ponto de referência, para um segundo ponto. (...) O ângulo horizontal em um determinado ponto medido no sentido dos ponteiros do relógio, a partir de um datum de referencia para outro ponto. O mesmo que ângulo de orientação. A direção horizontal de um ponto terrestre para outro, expressa como distância angular a partir de uma direção de referência. É medida, habitualmente, a partir de 0° numa direção de referência, no sentido dos ponteiros do relógio, até 360°. Os termos orientação e azimuth têm às vezes, uso recíproco, mas, em navegação, o primeiro é aplicado, quase sempre, a assuntos terrestres, e o segundo a direção de um ponto da esfera celeste, a partir de um ponto da Terra
Orientação geodésica	"Sistema de controle horizontal apoiado em um lado cujo azimuth geodésico é conhecido"
Localização	"Determinação exata de um ponto ou detalhe numa carta ou numa fotografia; traçado de acabamento depois da marcação dos pontos principais, como um diagrama; traçado e marcação (um ponto) como em um papel milimetrado, por meio de suas coordenadas; colocação de dados de levantamento num mapa"

Localização geográfica	"mesmo que posição geográfica"
Posição geográfica	"A posição de um ponto da superfície da Terra, expressa em termos de latitude e longitude, seja geodésica ou astronômica."