RODRIGO DE SOUZA VIEIRA

PROTÓTIPO DE UM SISTEMA DE MONITORAMENTO REMOTO INTELIGENTE

Dissertação apresentada ao Curso de Pós-graduação em Engenharia de Produção da Universidade Federal de Santa Catarina, para obtenção do título de Mestre em Engenharia

Universidade Federal de Santa Catarina

Florianópolis
1999

RODRIGO DE SOUZA VIEIRA

PROTÓTIPO DE UM SISTEMA DE MONITORAMENTO REMOTO INTELIGENTE

Dissertação apresentada ao Curso de Pós-graduação em Engenharia de Produção da Universidade Federal de Santa Catarina, para obtenção do título de Mestre em Engenharia

Área de concentração:
Inteligência Aplicada

Orientador:
Prof. Neri dos Santos, Dr. Ing.

Universidade Federal de Santa Catarina

Florianópolis
1999

PROTÓTIPO DE UM SISTEMA DE MONITORAMENTO REMOTO INTELIGENTE

RODRIGO DE SOUZA VIEIRA

Esta dissertação foi julgada adequada para obtenção do título Mestre em Engenharia, Especialidade Engenharia de Produção, e aprovada em sua forma final pela Banca Examinadora formada pelos professores:

Prof. Neri dos Santos, Dr. Ing. – Presidente

Prof. Alejandro Martins Rodriguez, Dr. – Membro

Prof. Aran B. Tcholakian Morales, Dr. – Membro

Florianópolis, junho de 1999

Agradecimentos

Ao professor Neri dos Santos por aceitar o desafio de orientar este trabalho.

Aos meus familiares que sempre me motivaram e torceram pelo sucesso do resultado final.

A Elza Galdino pelas sugestões e observações sobre o vernáculo, que tanto enriqueceram o texto final.

Ao amigo Salvador Tirloni, incansável no auxílio no desenvolvimento e teste dos circuitos eletrônicos.

A Rosiani Soares pelo apoio, verificação e adequação às normas de todas as citações e referências bibliográficas.

Ao professor Nelci Barros, sem o qual as fotos do robô não seriam possíveis.

Resumo

O presente trabalho tem como objetivo o projeto e construção de um protótipo de sistema de telepresença com determinado nível de inteligência que permita interagir com o usuário remoto.

A interação é caracterizada pela capacidade que o robô tem de optar pelo comando do usuário ou pela leitura que faz do ambiente onde está inserido, utilizando para tal uma importante ferramenta de inteligência artificial: a lógica difusa.

Esta escolha deve-se em grande parte a sua própria conceituação teórica, que a torna apta a manipular dados imprecisos, vagos ou aleatórios, presentes neste caso.

A lógica difusa foi utilizada sob a forma de um controlador difuso, cuja programação foi inserida em um microcontrolador da família Intel 8051, habilitando o robô a evitar choques com obstáculos – fixos ou móveis – dispersos ao seu redor, sem prévio conhecimento de suas posições.

Para o desenvolvimento do controlador foi adotado o conceito de controle de robô em camadas, ou níveis de competência de Rodney A. Brooks. Este modelo força o emprego de um processamento paralelo das informações, uma vez que os dados de entrada são divididos entre os diversos níveis – baseados na sua relevância – resultando em uma ação única.

Para permitir a telepresença, um canal de comunicação foi estabelecido entre operador e robô. A comunicação é sem fios, possibilitando por um lado o envio de comandos do usuário para a máquina e da imagem coletada no ambiente remoto no sentido inverso.

A utilização de processamento a baixo nível no robô, executado pelo microcontrolador, simplificou a comunicação entre as duas unidades envolvidas no sistema, pois o trafego de dados entre elas ficou reduzido ao estritamente necessário.

O robô – batizado com o nome RAU – *Remote Action Unit*, foi desenvolvido especificamente para a tarefa de telepresença apresentada no trabalho, utilizando um projeto modular, que permite a expansão das suas habilidades, possibilitando que sirva de base para outras pesquisas.

Abstract

The objective of the present work is the project and construction of a telepresence system prototype with a certain intelligence level that allows interacting with the remote user.

The interaction is characterized by the capacity that the robot has to opt by the user command or by the reading that makes the environment where it is inserted, using for this an important tool of artificial intelligence: the fuzzy logic.

This choice is mostly because of its own theoretical conception, that becomes it apt to manipulate imprecise, vague or random data, present in this case.

The fuzzy logic was used on the form of a fuzzy controller, which the software was inserted in a microcontroller of Intel 8051 family, helping the robot to avoid crashing obstacles – fixed or mobile – spread around, without previous acknowledge of their positions.

For the development of the controller was adopted the robot control concept in layer, or competence levels of Rodney A. Brooks. This model forces the employment of an information parallel processing, since the input data are divided among several levels, based in its relevance, resulting in a single action.

To allow the telepresence, a communication channel was established between the operator and the robot. The communication is without wires, making possible on one side to send the commands of the user for the machine and the image collected in the remote environment in an inverted sense. The use of the processing under the robot level, executed by the microcontroller, simplified the communication between the two units involved in the system, because the data that are changed between them was reduced to strictly necessary.

The robot, named as RAU – Remote Action Unit, was developed specifically for this telepresence task, using a modular project, being able, for this reason to have their skills expanded, serving as the base for other researches.

Sumário

Lista de Figuras

Lista de Tabelas

Capítulo 1 – Introdução

1.1. Apresentação do problema de pesquisa

Vivemos a era da automação onde produtividade, rapidez e eficiência são cada vez mais presentes. O avanço da tecnologia, em especial a da microinformática, tem facilitado em muito nossas vidas, além de nos familiarizarmos com sistemas que há pouco tempo habitavam apenas os filmes de ficção científica.

A fiabilidade que os sistemas automatizados adquiriram durante o período de seu desenvolvimento nos faz acreditar cada vez mais nas máquinas, fato impensável até muito recentemente. Isto pode ser observado quando utilizamos um caixa automático de banco. Nele confiamos primeiro quando lhe informamos uma senha, que é um código sigiloso, acreditando que a máquina não a fornecerá a mais ninguém e, em segundo, quando operamos as transações financeiras desejadas certos de que elas realmente se efetuarão, por intermédio de uma série de computadores conectados ao terminal que utilizamos. Se não acreditássemos que a máquina cumprisse o seu papel, talvez os caixas automáticos não seriam tão populares como hoje.

Além disto, as máquinas estão se tornando cada vez mais inteligentes, capazes de um certo grau de raciocínio, na maioria das vezes específico para determinada aplicação. Como sua capacidade de realizar cálculo é infinitamente maior que a nossa, em algumas situações elas são até mais eficientes, como ocorreu no derrota do enxadrista Garry Kasparov para o *Deep Blue*, um computador da IBM desenvolvido especialmente para jogar xadrez.

Não foi a primeira vez que o campeão mundial de xadrez jogou com uma máquina; Kasparov já havia vencido o precursor do *Deep Blue* (ALCÂNTARA, 1997) mas o duelo de 1997 mostrou que "*nunca o potencial humano havia sido medido contra a máquina de maneira tão brutalmente crua*" (CHABRIS, 1997). Deve-se lembrar que por trás da programação da máquina trabalhou uma equipe inteira, que provavelmente seria batida por Kasparov se todos sentassem ao tabuleiro, pois ela ensinou ao computador como jogar. Logo, no momento de realizar cada lance, a máquina partia do seu próprio conhecimento, aprendendo, inclusive, o modo de jogar do oponente.

Mas a utilização das máquinas está além de nos fornecer o dinheiro de nossas contas bancárias quando precisamos ou de testar nossas habilidades em jogos de tabuleiros. Elas são, acima de tudo, nossas escravas onde quer que as coloquemos, e isto deve ser sempre o ponto de maior relevância na relação entre homem e máquina. Como escravas, elas efetuam as operações rotineiras, entediosas e insalubres que outrora eram executadas por seres humanos, o que fica patente quando se fala de robôs. Os robôs surgiram como máquinas na década de 20 e eram cópias morfológicas dos seres humanos. O *Televox*, criado em 1928 por J. Wensley, se assemelhava a nós e apresentava movimentos básicos, de acordo com os comandos de seu operador (POLONSKII, 1996).

Durante as duas décadas seguintes os robôs desenvolvidos apresentavam sempre estas características básicas: a semelhança morfológica com os humanos e a capacidade de repetir comandos dados por um operador. Foi somente em 1954, com o advento da eletrônica, que surgiu o primeiro robô automático, da idéia de J. K. Divol. Este projeto serviu de base para o *Unimate* e o *Wersatran*, que foram os primeiros robôs digitais comerciais de uso industrial (POLONSKII, 1996).

De lá para cá os robôs passaram a ser comuns nas fábricas do mundo todo e, apesar da fama inicial de tomadores de postos de trabalho, hoje convivem pacificamente com os humanos. Afora a discussão do impacto social causado pelo robotização das plantas fabris do mundo todo, há de se perceber que eles vêm tendo aplicações cada vez maior também fora das indústrias, chegando inclusive aos nossos lares, novamente voltados à ações repetitivas e entediosas, como o robô aspirador de pó desenvolvido por Firby (1993).

Na área de pesquisa científica, os robôs sempre figuram como elementos importantes e sua presença é mais marcante na conquista do espaço. Desde o lançamento da primeira sonda robotizada *Surveyor* – que tinha o objetivo de recolher informações sobre a Lua – em 1966, veículos robotizados percorrem o cosmos, tornando as missões espaciais sinônimo de robótica. Este campo de utilização dos robôs é fortalecido, principalmente, pela capacidade que eles têm de estender a percepção humana a locais remotos da nossa galáxia, através da telepresença.

A telepresença é outra área de pesquisa que vem ganhando grande destaque, seja como entretenimento, aliada a realidade virtual; como ferramenta de gestão empresarial, através da vídeo conferência; e como instrumento de exploração e pesquisa de locais distantes ou impróprios para os seres humanos. Inserida em um contexto maior que é a teleoperação, pode tornar ilimitada a presença do homem em nosso mundo, levando-a desde o mais profundo abissal até os mais distantes planetas do universo.

Aliando estes três fatores mencionados anteriormente, a robótica, a telepresença e a inteligência artificial, homens e máquinas poderão interagir de forma ainda mais íntima, trocando informações, experiências e conhecimento, nos fazendo entender mais sobre nosso meio e sobre nós mesmos.

Sob outro aspecto, podemos fazer uma breve analogia entre homem e máquina, onde a robótica provê os músculos e articulações necessários as ações pretendidas; os sentidos como tato e visão são obtidos pela telepresença e a inteligência artificial, por sua vez, consegue representar o sistema nervoso central e suas terminações.

É dentro deste cenário – da interligação entre estas três áreas de conhecimento – que apresentamos este trabalho, o qual utiliza um robô autônomo em operações básicas de telepresença, quebrando limitações de sobrevivência na exploração de outros locais.

1.2. Justificativa

Os homens, como todos os seres vivos, têm limitações para se adaptarem a determinados locais, principalmente onde não são encontrados elementos básicos para suas funções vitais, como o oxigênio. Além disto, resistimos apenas a uma estreita faixa de variação de temperatura e pressão, não sobrevivemos na presença de gases tóxicos e somos incapazes de nos manter por muito tempo em ambientes sem a presença de luz e água.

Vale observar que temos um grande poder de adaptação ao meio, o que justifica o domínio que temos sobre o planeta, pois o homem sobrevive tanto no mais quente deserto africano como na mais fria geleira antártica. Naturalmente, existem meios que tornam esta sobrevivência possível sem, no entanto, excluir as limitações descritas anteriormente.

Graças a sua capacidade intelectual o ser humano conquistou os pólos, os mais profundos abissais dos oceanos, as mais altas montanhas e distantes pontos da nossa galáxia.

A veia exploradora no ser humano sempre teve um forte apelo, o que pode ser percebido desde a infância, quando desejamos sempre conhecer coisas novas, tocando, vendo, ou mesmo escutando. Os sentidos são os nossos principais aliados nestas descobertas, traduzindo para nosso cérebro o que observamos.

No princípio, ao segurar uma bola, a criança tem que associar o nome "bola" ao objeto, porém, após a aprendizagem do que é uma bola, ao ver algo de formato esférico ou tocá-lo mesmo com os olhos fechados, a conhecimento é resgatado do nosso cérebro "– é uma bola!".

No entanto, nossos sentidos, como nossa vida, são limitados; a partir de uma certa distância não podemos tocar ou mesmo enxergar. O desejo de ultrapassar esses limites certamente foi um fator preponderante na criação de sistemas de telepresença, abrindo espaço para a extensão dos sentidos humanos, seja a audição, a visão ou o tato.

O sistema de telepresença tratado neste trabalho tem o objetivo de fazer com que o usuário possa monitorar um ambiente remoto através da visão, podendo ser futuramente adaptado a outros sentidos como a audição, olfato e tato, por exemplo. Um robô deste tipo – sendo compacto e podendo trabalhar em situações de atmosfera hostil – tem emprego em um sem número de operações, como inspeção de dutos de ar-condicionado e tubulações de alta pressão, na busca de sobreviventes em desabamentos ou terremotos, em sistemas de segurança patrimonial, no desarme de bombas e na exploração espacial, entre outros.

A utilização destes robôs em postos que anteriormente eram tomados por humanos vem no sentido de preservar o patrimônio mais valioso que temos: a vida.

1.3. Objetivos

Para delinear o projeto apresentado definimos dois níveis de objetivos a serem alcançados na execução deste trabalho, um geral e os específicos, que podem ser entendidos, respectivamente, como metas e submetas, divididos e descritos conforme segue.

1.3.1. Objetivo geral

O objetivo geral desta dissertação é conceber um protótipo de um sistema de telepresença alicerçado em três pontos básicos: a robótica, a telepresença e a inteligência artificial.
Para tal foi desenvolvido um robô, não comercial, conectado a um computador pessoal, capaz de interagir com o operador buscando o melhor caminho em um ambiente desconhecido, e de fornecer informações do meio remoto onde se encontra.
Como o robô aqui desenvolvido não partiu de qualquer modelo existente no mercado, o seu projeto foi voltado à aplicação para telepresença, servindo de base para pesquisas futuras neste campo.

1.3.2 Objetivos específicos

Podemos elencar 4 objetivos específicos a serem alcançados neste trabalho:

- Propiciar a utilização de um robô controlado por lógica difusa: onde o que se deseja é analisar como um controlador difuso opera em um sistema real, numa aplicação com uma forte presença de incerteza e aleatoriedade, inexistindo um modelo analítico para comparação dos resultados.
- Produzir um sistema de teleoperação a baixo custo: deve-se observar sempre a componente custo no projeto, objetivando verificar a possibilidade da construção de tal sistema a um preço acessível.

- Testar a capacidade de programação de um sistema inteligente em um microcontrolador conhecido no mercado: consiste em averiguar a possibilidade de utilizar microcontroladores em operações deste tipo, em robôs autônomos.
- Verificar a funcionalidade do controle em camadas de Brooks: como base do desenvolvimento do controlador do robô, deve-se constatar a funcionalidade e operacionalidade do conceito de Brooks em uma aplicação prática.

1.4. Questão a investigar

Diante do exposto anteriormente, podemos determinar como questão a ser investigada a capacidade de se produzir um sistema de telepresença compacto, simples e de baixo custo, para aplicações de monitoramento remoto.

Procura-se aqui definir um protótipo, de operação simplificada, capaz de realizar operações a distância, ajudando o seu usuário na tomada de decisões quando este não tiver condições de fazê-lo por insuficiência de informações.

O aspecto modular deve também ser levado em conta, de forma a facilitar a eliminação de erros durante a montagem, além de propiciar o incremento da capacidade do sistema, possibilitando sua adaptação a diversos usos.

1.5. Delimitação do estudo

Este trabalho de pesquisa está definido como o projeto e construção de um protótipo de um sistema de telepresença baseado em um microcomputador compatível IBM-PC, definido de acordo com os requisitos descritos em seu capítulo 3.

De forma a delimitar a pesquisa realizada, optou-se por um sistema reativo (SAFFIOTTI, RUSPINI & KONOLIGE,1999) no qual o robô responde a estímulos – captados por seus sensores – provenientes do meio onde está inserido. As reações do robô são determinadas por um sistema inteligente, baseado em lógica difusa, capaz de traçar o melhor caminho, evitando os obstáculos – móveis ou fixos – ao seu redor.

Outro ponto relevante é a teleoperação do sistema, garantida pelo canal de comunicação entre o robô e o operador, projetado para funcionar sem fios, oferecendo total mobilidade ao protótipo.

A teleoperação resulta na necessidade de o projeto poder receber comandos de um operador remoto, o que nos leva, conseqüentemente, à adoção do conceito de robô em camadas de Brooks (1986) em seu controle, utilizando diferentes níveis de competência para o operador e para o controlador embarcado. Assim, o robô poderá verificar a ação determinada pelo usuário e a direção determinada pelo seu controlador difuso, optando por uma ou outra, em prol da segurança do sistema.

Por fim, visando a simplificação deste sistema, optou-se por limitar a aplicação a uma superfície completamente plana, livre de declives e degraus, com um conjunto de obstáculos de cor clara aleatoriamente distribuídos. Este último requisito deve ser considerado no uso de sensores infravermelhos, que acabam gerando grandes variações de sensibilidade de acordo com a cor do obstáculo.

1.6 Estrutura do trabalho

O presente trabalho está dividido em oito diferentes capítulos e três anexos, abordando os conteúdos descritos a seguir.

O capítulo 1 introduz o trabalho, falando do problema abordado, justificando-o e limitando-o. São descritos também os objetivos geral e específicos, além da questão a ser investigada.

No capítulo 2 é dado um breve histórico sobre os robôs móveis, a definição do que é um robô móvel segundo diversos autores, concluindo com a conceituação de robôs inteligentes, prevendo aplicações e tendências para estes equipamentos.

O capítulo 3 define o problema a ser tratado. Nele são versados os conceitos de telepresença e teleoperação, os requisitos a serem cumpridos pelo projeto, além de abordar as etapas envolvidas na produção do protótipo.

No capítulo 4 são mostrados os procedimentos metodológicos utilizados na busca da solução para o problema. São descritos os aspectos cinemáticos do robô, juntamente com a lógica difusa aplicada a controladores.

O capítulo 5 trata do hardware do robô, descrevendo como foi definido o seu corpo, a utilização dos sensores, os equipamentos de saída e de força capazes de permitir a comunicação do robô com o seu operador e sua movimentação.

No capítulo 6 é descrito o cérebro do sistema, dando atenção ao microcontrolador utilizado, o software de controle e de comando desenvolvidos. São abordadas também a utilização da linguagem C e a definição do robô em camadas de Brooks.

Já o capítulo 7 apresenta os testes, as modificações ocorridas no projeto original e a montagem do protótipo, discutindo os resultados obtidos nos ensaios realizados.

Por fim, o capítulo 8 apresenta a conclusão do trabalho relatando os objetivos alcançados e sugerindo trabalhos que podem vir a ser realizados tomando por base o ora apresentado.

Os anexos, em número de três foram designados A, B e C. O primeiro apresenta a programação utilizada no controlador do robô. É mostrada a listagem do programa escrito em C desenvolvido para o microcontrolador compatível Intel 8051.

O anexo B apresenta algumas fotos dos componentes, da montagem e dos testes do protótipo do sistema.

O anexo C, apresenta, finalmente, o modelo computacional simulado no software MATLAB, mostrando também a seqüência de procedimentos adotados na sua implementação.

Capítulo 2 – Robôs móveis: uma breve revisão de literatura

2.1. Introdução

O sonho de criar um semelhante – como o próprio Deus o fez na narrativa bíblica – acompanha o homem desde remotos tempos, como relatado na história da arte, quando ao acabar a escultura *Moisés*, Michelangelo ordenou: "- Fala!". O fascínio que esta idéia provoca nas pessoas pode ser facilmente demonstrado ao ser lembrada uma novela escrita em 1817 por Mary Shelley: "Frankenstein or Modern Prometheus". Nesta obra, o Doutor Frankenstein, um cientista de idéias não muito sensatas, cria um ser humanóide, de aparência assustadora, mas de temperamento dócil, ao contrário do que o cinema preconizou nas diversas filmagens. O monstro não tinha outra função que não tarefas rotineiras e exaustivas (GROOVER, WEISS, NAGEL et al, 1988).

Na história também são fartos os exemplos da utilização de imagens humanizadas de criaturas, como descrito na obra de Homero, onde Hofesto, deus grego das forjas, possuía duas mulheres feitas em ouro, capazes de realizar tarefas tipicamente humanas (ASIMOV, 1994). No século 18, Jacques de Vaucanson, um mecânico francês, se tornou célebre por criar autônomos que tocavam música (GROOVER, WEISS, NAGEL et al, 1988), simulando inclusive os movimentos labiais ao se tocar uma flauta, utilizando um complexo engenho com látex (MCKERROW, 1991). O movimento destes autônomos era totalmente planejado, estruturado em cames e eixos – exclusivamente mecânico – encadeados de tal forma que fornecessem o movimento final desejado. (MCKERROW, 1991). Nesta época também começaram a surgir as primeiras máquinas programáveis, demonstrando que estes novos mecanismos poderiam executar outras tarefas além daquelas, destinadas ao entretenimento das pessoas (GROOVER, WEISS, NAGEL et al, 1988).

A história da robótica se confunde com a história da literatura de ficção científica, de onde até tirou o seu próprio nome. Em 1921, apesar de já existirem aplicações de sistemas que reproduziam ações típicas de humanos, de certa forma menos sofisticadas

e extremamente especializadas se comparadas aos dias de hoje, o teatrólogo tcheco Karel Capek, na sua peça *Os Robôs Universais de Rossum* utilizou pela primeira vez o nome robô (que em eslovaco significa trabalhador) para identificar homens artificialmente fabricados pela personagem principal para executar todas as tarefas árduas do mundo (ASIMOV, 1994; GROOVER, WEISS, NAGEL et al, 1988; KOREN, 1985; MCKERROW, 1991). A literatura tem muitas definições para robô, desde um mero escravo a serviço do homem (KOREN, 1985) até máquinas que, ainda que não na forma, se assemelham aos humanos nas suas funções (MCCOMB, 1987). Para outros, no entanto, os robôs sempre serão apenas conexões inteligentes entre percepção e ação (JONES & FLYNN, 1993; RICH & KNIGHT, 1994), ou ainda, agentes responsáveis por ações inteligentes em face de situações reais**.**

Uma definição que foi utilizada neste trabalho como conceituação sobre o significado de robô foi extraída da obra de David Nitzan: "*um robô é uma máquina de uso geral que, como um humano, pode executar uma variedade de diferentes tarefas sob condições que não necessitam ser conhecidas a priori*" (NITZAN, 1985). Na mesma linha, complementando Nitzan, Gevarter apresenta a definição de Marsh (1981) para robôs: *"...são máquinas flexíveis capazes de controlar suas próprias ações em uma variedade de tarefas utilizando uma programação armazenada.*" (GEVARTER, 1984).

Foram eleitos três conceitos relativos aos robôs para servirem de alicerce para esta dissertação:

- realizar tarefas que não necessitam ser conhecidas a priori;
- máquinas flexíveis, e
- ser capaz de controlar suas próprias ações baseado na programação armazenada.

A visão que a maioria das pessoas tem quando se fala a respeito de robôs é de mecanismos fixos, capazes de efetuar tarefas como a manipulação de objetos em sua área de trabalho, sendo reconhecidos pela forma de um braço mecânico com um certo número de articulações (MCCOMB, 1987). Isto é ainda mais marcante quando se pensa em robótica na indústria, onde até os dias atuais os robôs fixos estão em grande parte das aplicações. Muitos dos conceitos típicos dos robôs fixos, como graus de liberdade, tipos de junções e análise cinemática – certamente muito mais relacionados

aos manipuladores (ANDEEN, 1988) ou braços mecânicos – são empregados de forma generalizada no campo da robótica, pelo fato deste tipo de robô estar no mercado e ser objeto de pesquisa a mais tempo.

Como o próprio nome indica, os robôs móveis são muito mais versáteis, pois não precisam estar fixados a uma célula de trabalho, podendo ser utilizados em tarefas onde não existam limites geográficos, movimentando-se por meio de pernas ou rodas (MCKERROW, 1991).

Mesmo sendo referenciados como atores principais nos filmes de ficção científica, a lembrar o C3-P0 e o R2-D2 da trilogia *Star Wars* (1978), os robôs móveis encontram hoje um vasto campo nas aplicações industriais, como na limpeza do piso de prédios e fábricas, nos sistemas móveis de vigilância, no transporte de componentes em fábricas, na colheita e seleção de frutas (MONDADA, FRANZI & IENNE, 1993).

Da mesma forma, os robôs móveis têm aplicação no meio acadêmico e científico, onde biólogos, psicólogos e etnólogos podem, através deles, simular conhecimentos retirados do mundo real (ARKIN, 1990; MONDADA, FRANZI & IENNE, 1993, HALLAM & HAYES, 19__), como a interação entre indivíduos de uma mesma comunidade, em prol de um objetivo geral.

2.2. Breve histórico dos robôs móveis

O primeiro robô móvel construído e reconhecido na bibliografia é o *Shakey*, desenvolvido pelo *Stanford Research Institute*, em 1968. Ele tinha uma variedade enorme de sensores, incluindo uma câmara de vídeo e sensores de toque, binários, (GROOVER, WEISS, NAGEL et al, 1988) e navegava entre as salas do laboratório, enviando sinais de rádio ao seu cérebro, um computador DEC PDP-10, que permitia efetuar algumas tarefas como empurrar caixas e evitar obstáculos (NITZAN, 1985). Deve-se atentar para o fato que a unidade de processamento embarcada no robô apenas coletava os sinais sensoriais e os enviava para um computador remoto que executava o processamento, transmitindo ao robô o comando que geraria a ação desejada.

Outra experiência relatada data de 1977 com o veículo *StanfordCart* do *Stanford Artificial Intelligence Laboratory* (BOTELHO, 1996). O *StanfordCart* - trabalhando

em um local plano com obstáculos esparçamente colocados - utilizava um sistema de navegação baseado no "*parar e seguir*", parando e fazendo leitura de seus sensores a cada metro percorrido, realizando o planejamento da rota a seguir, (BRUMITT, COULTER & STENTZ, 1992).

Chamados de móveis autônomos, os robôs descritos acima, são capazes de encontrar solução para os seus próprios problemas em um ambiente sem supervisão externa (BOTELHO, 1996). Entre os vários trabalhos nesta emergente área do conhecimento, impulsionada em grande parte pelo aumento da capacidade de manipulação de dados dos computadores, vale ressaltar o Hilare, um robô multisensorial desenvolvido no LAAS – *Laboratoire d'Architecture et d'Analyse de Systèmes* em 1983 (BOTELHO, 1996).

Durante os anos 80 vários trabalhos foram desenvolvidos em todo o mundo como o *NAVLAB*, *PSEIKI* do *Robot Vision Laboratory* da *Purdue University*, baseado em reconhecimento de imagens através da visão (1989). Outro importante estudo sobre robôs móveis foi o desenvolvimento do robô *Khepera*, do *K-Team*, no *Microcomputing Laboratory* do *Swiss Federal Institute of Technology*, com o apoio de outras entidades de pesquisa da Europa, como *University of Zürich*, *University of Karlsruhe*, *University of Sussex*, *Laboratoire d'Etudes et Recherche en Informatique* e *Ecole Nationale Supérieure (Paris)* (MONDADA, FRANZI & IENNE, 1993; MONDADA & FRANZI, 1993). Este robô, com apenas 55 mm de diâmetro por 30 mm de altura tem a capacidade de desviar de obstáculos e seguir ou evitar fontes luminosas, de forma similar ao descrito na obra de Braitenberg (1984), além de permitir a utilização de algumas extensões, como sistema de visão e um pequeno manipulador. Seu tamanho reduzido e seu custo relativamente baixo, comparado a outros robôs no mercado, tem feito o *Khepera* um sucesso dentro das universidades, sendo objeto de pesquisa inclusive no Brasil (BOTELHO, 1996).

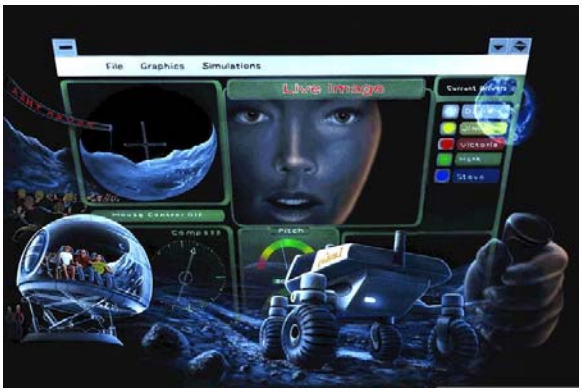
No entanto, a mais espetacular aplicação de robôs móveis deste século é a família *Rocky*, resultado das pesquisas do *Jet Propulsion Laboratory* do *California Institute of Technology* (HAYATI, 1996; HAYATI, VOLPE, BACKES et al, 1997; MATTHIES, 1995). Um modelo *Rocky 4*, com dimensões comparadas às de um brinquedo de radiocontrole e batizado carinhosamente como *Sojourner* (hóspede temporário em

inglês) (fig.2.1), foi capaz de realizar uma das maiores façanhas da pesquisa do espaço pelo homem: a exploração, a distância, do planeta Marte. Graças ao *Sojourner* – comandado da Califórnia na coleta de material e na sua própria locomoção - os 193 milhões de quilômetros que nos separam do Planeta Vermelho foram reduzidos a apenas 10 minutos (desprezada a diferença entre as grandezas envolvidas), tempo que cada imagem levou para chegar à Terra (KAC, 1998), ficando provado que a teleoperação em ambientes completamente desconhecidos é perfeitamente viável, fato mostrado ao vivo para todo mundo pela rede de televisão CNN. Ainda no campo espacial uma nova apoteose está guardada para o uso de robôs móveis. A empresa *LunaCorp*[1], juntamente com o *Robotics Institute*, da *Carnegie Mellon University* pretendem lançar, até a virada do século, dois veículos móveis autônomos à Lua.

Fig 2.1 – O robô *Sojourner* do *Jet Propulsion Laboratory* (foto JPL).

O objetivo da empresa é criar um parque temático na Terra, de onde as pessoas poderão comandar os veículos e sentir-se virtualmente explorando o solo lunar (fig. 2.2a). Os veículos contarão com uma câmera a bordo que transmitirá as imagens da Lua para a Terra (fig. 2.2b). Esta imagem será então projetada em uma cápsula de 360º, com seis graus de liberdade, simulando os movimentos coletados por sensores colocados na suspensão dos veículos, propiciando um elevado grau de imersão em um mundo virtual. No tocante a teleoperação, o maior problema que os cientistas terão que contornar, sem dúvida será a demora de 3 a 5 segundos no envio de sinais da Lua até o nosso planeta.

[1] Informações obtidas no endereço http://www.lunacorp.com, na Internet



(a)

(b)

Fig. 2.2 a) o projeto *LunaCorp*; b) o veículo de exploração (fotos LunaCorp)

O primeiro protótipo da *LunaCorp*, batizado de *Nomad*, já vem sendo testado no deserto de Atakama, no Chile, sob supervisão do *Robotics Institute*, com financiamento da NASA.(fig. 2.3)

Outro projeto de robô móvel autônomo que merece destaque é o *Honda Humanoid Robot*. Topologicamente semelhante aos seres humanos – com cabeça tronco e membros – o *Honda Humanoid* conta com noções de equilíbrio, é capaz de subir escadas, rampas e de se manter em pé, apesar dos seus 130 kg (MAIOR, 1998). A atual versão P3, difere do seu antecessor, o modelo P2, nas proporções, mais próximas das humanas, como pode ser visto, comparativamente na figura 2.4[2].

Fig. 2.3 – O Veículo Nomad (foto Robotics Institute)

[2] Alguns dados do Honda P3: 1,6 metros de altura, 0,6 metros de largura, pernas com 12 graus de liberdade, braços com 14 graus de liberdade e construído em liga de Magnésio.

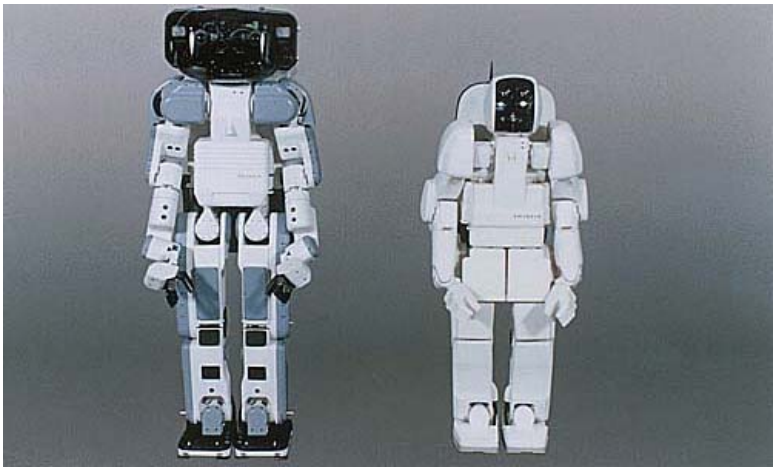


Fig. 2.4 – Os robôs antropométricos da Honda, P2 (esq.) e P3

2.3 Definições de robôs móveis

2.3.1 O que é um robô móvel

Segundo McComb, existem duas vertentes na definição de robôs móveis: uma que diz ser o robô completo, auto-contido, autônomo, que necessita instruções de seu mestre apenas ocasionalmente. Já a segunda é aquela que define robô como sendo qualquer equipamento que se mova por seus próprios meios, com o objetivo de executar tarefas próximas as humanas. (MCCOMB, 1987). As duas definições de McComb levam a aceitar que um robô móvel é capaz de manobrar livremente em seu ambiente, alcançando metas enquanto desvia de outros obstáculos (SPENCE & HUTCHINSON, 1995).

Uma vez que necessitam ser livres, ou seja, sem conexões e fios, a aplicação de robôs móveis sofre limitações, basicamente à armazenagem e geração de energia para o próprio robô (MCKERROW, 1991).

2.3.2 Topologia de robôs móveis

Nitzan (1985) define o robô como um conjunto de quatro componentes:

- atuadores,
- sensores,
- computadores e
- equipamentos auxiliares.

O autor entende como atuadores: os braços, mãos, pernas e pés, ao passo que os sensores ele classifica em com e sem contato. Os computadores são responsáveis pelo controle, a alto e baixo nível, ou seja, pelo planejamento do caminho e pelo levantamento sensorial do ambiente, bem como os canais de comunicação entre eles. Por fim, os equipamentos auxiliares são as ferramentas, guias, fixadores, mesas, entre outros (NITZAN,1985). Esta topologia é compartilhada por (JONES & FLYNN, 1993), onde dizem que um robô é um sistema com uma coleção de sensores, atuadores e elementos computacionais.

Ampliando a visão topológica dos autores anteriormente citados (MCCOMB, 1987) vê os robôs móveis como um conjunto composto por seis elementos:

- corpo;
- sistema de força;
- sistema de locomoção;
- braços e atuadores;
- sensores e
- equipamentos de saída.

Para o autor, "*o suporte eletrônico*" deve ser colocado separadamente (MCCOMB, 1987), ou seja, o sistema de controle computacional do robô é considerado externo a ele, o que justifica a inclusão, na lista acima, de equipamentos de saída responsáveis pela sua comunicação.

McKerrow (1991) prefere definir o robô como um conjunto de subsistemas que de forma bem abrangente são classificados como:

- de processo;
- de planejamento;
- de controle;
- de sensores;
- elétrico e
- mecânico.

O subsistema de processo é o responsável pelas tarefas a serem realizadas pelo robô e sua interrelação com o ambiente onde está contido. Já o subsistema de planejamento é o responsável pelo planejamento das tarefas definidas no primeiro, enquanto no subsistema de controle os comandos de alto-nível são transformados em ações dos atuadores, que fazem parte do subsistema elétrico juntamente com interfaces e computadores. Para tornar possível a ação dos atuadores são utilizados sistemas

hidráulicos ou pneumáticos, elementos do subsistema mecânico. Por fim o subsistema de sensores serve como monitoramento das ações do robô, fornecendo um retorno ao subsistema de controle (MCKERROW, 1991). A interrelação entre os subsistemas pode ser verificada na figura 2.5.

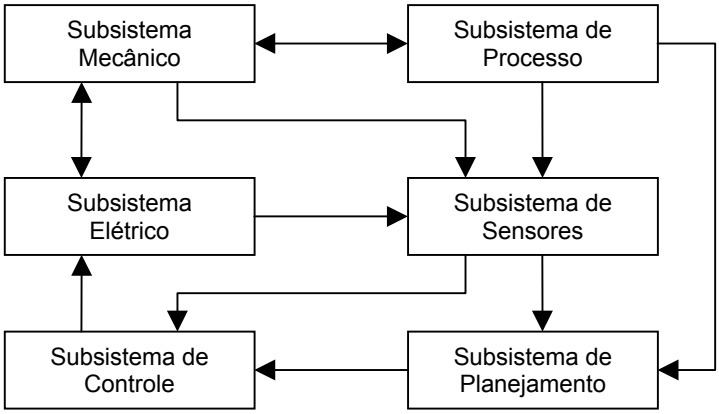


Fig. 2.5 Interrelação entre os subsistemas de McKerrow

Por acreditarmos ser a de mais clara aplicação no problema proposto neste trabalho, A topologia apresentada por McComb será aqui utilizada, conforme será descrito no Capítulo 5.

2.3.3 Robôs inteligentes

Como foi relatado anteriormente, para alguns autores os robôs sempre serão apenas conexões inteligentes entre percepção e ação (JONES & FLYNN, 1993; RICH & KNIGHT, 1994) e, para que estas conexões possam ser definidas e tenham um comportamento realmente inteligente, técnicas de inteligência artificial podem ser utilizadas.

Ainda que o estudo semântico do que seja inteligência artificial não seja estudo deste trabalho, adotamos uma conceituação como balizamento.

A inteligência artificial, surgida no Dartmounth College, no verão de 1956 (GEVARTER, 1984), pode ser definida, por sua vez, como um campo de estudo que busca explicar e emular comportamentos inteligentes em termos de processos computacionais (SCHALKOFF, 1990). Se restringirmos o foco da discussão, encontraremos também conceitos mais simplistas, onde a inteligência artificial é uma

tecnologia nova que tem sido utilizada de tal forma que os programas computacionais tornem os computadores mais espertos (GEVARTER, 1984). Fazendo um paralelo, finalmente encontra-se a interpretação, do ponto de vista da engenharia, dado por Shalkoff. Para ele a inteligência artificial é "*...gerar representações e procedimentos que automaticamente (autonomamente) resolvam problemas até agora resolvidos por humanos*" (SCHALKOFF, 1990).

Algumas aplicações da inteligência artificial no campo da robótica estão relacionadas ao planejamento do caminho e movimento (navegação), visão de computador, controle, sistemas de diagnóstico e planejamento da manufatura (SCHALKOFF, 1990).

Alguns autores classificam robôs inteligentes como aqueles que realizam decisões em tempo real, suportadas por algoritmos de inteligência artificial, baseados em sensores (KOREN, 1985), o que lhes possibilita trabalharem em ambientes completamente imprevisíveis.

Comparando as técnicas de controle tradicionais com aquelas provenientes da aplicação da inteligência artificial, pode-se classificar estas últimas como hierarquicamente superiores às primeiras (GEVARTER, 1984). No caso de aplicação em máquinas autônomas, especificamente no caso ora estudado - ou seja, no campo da robótica - o emprego da inteligência nos robôs não os tornará apenas mais flexíveis nas linhas de montagens, mas também os tornarão cada vez mais aptos a executarem atividades fora do ambiente industrial (GEVARTER, 1984) mais próximas das humanas, no conceito de Shalkoff.

2.3.4 Métodos de navegação

Conforme descrito anteriormente, a navegação permite aos robôs móveis uma livre desenvoltura pelo seu ambiente de trabalho, ora alcançando metas, ora desviando de obstáculos. Isto é obtido através de sistemas de navegação, presentes em todos os robôs deste tipo, algumas vezes diferentes quanto à forma de executar o seu trabalho, mas chegando objetivamente a um dos dois pontos: na geração de uma trajetória, ou no rastreamento de uma trajetória. Spence & Hutchinson (1995) ainda admitem que

esta maneira de dividir o problema é acarretado pela diferença entre os pesquisadores em planejamento de movimento e em teoria de controle. Segundo os autores existem aqui duas situações que podem ser devidamente separadas. A primeira é o planejamento da trajetória a ser seguida pelo robô, baseado principalmente na leitura dos obstáculos presentes no local. Já a segunda é o controle desta trajetória evitando o choque e preservando a integridade do robô.

Com isto, a geração de uma trajetória está mais voltada ao mundo externo, não considerando características cinemáticas ou dinâmicas do robô, o que não ocorre com os modelos apresentados para o rastreamento do caminho adotado (SPENCE & HUTCHINSON, 1995).

Sob outro aspecto, a navegação também é dividida por alguns autores em duas tarefas básicas: a de se localizar e a de evitar obstáculos (DEV, KRÖSE & GROEN, 1997). Pode-se perceber que a semântica da localização é referenciada como um controle de retorno para o operador, além de servir como um medidor de desempenho no alcance de um meta preestabelecida, assim, o robô saberá o quão próximo ou distante está do seu objetivo.

É inegável que muitas das técnicas adotadas no planejamento de um sistema de navegação têm elevado embasamento em sistemas biológicos, sendo estes últimos responsáveis por várias heurísticas empregadas neste campo. Isto é plenamente aceitável pois os sistemas biológicos têm demonstrado que através dos anos de evolução das espécies, tem encontrado modos extremamente eficientes de resolver problemas (MONDADA & FRANZI, 1993). Basear as ações dos robôs em ações típicas de seres vivos é uma excelente técnica no sentido de implementar comportamentos inteligentes (MCFARLAND & BÖSSER, 1993).

No caso de robôs que operam em locais abertos, como o proposto neste trabalho, alguns pontos na determinação do modelo de navegação devem ser levados em consideração, como (BRUMITT, COULTER & STENTZ, 1992):

a) a incerteza do meio ambiente – o sensoriamento deve ser simultâneo ao translado do robô, uma vez que é impossível ter um caminho pré-gravado;
b) a segurança do robô – se acontece uma falha do sistema de controle, o robô deve parar imediatamente, evitando qualquer choque;

c) o tempo computacional na resposta à situação – deve ser rápido o suficiente para que, com o aumento da velocidade do robô, seja possível ainda ter um sistema de navegação confiável;
d) a complexidade do terreno –a capacidade de trabalhar com as imperfeições do terreno sem acarretar em elevação do custo computacional é de primordial importância, e
e) a dinamicidade – se a velocidade é razoável, variáveis dinâmicas e cinemáticas devem ser levadas em consideração.

Dentre os modelos de navegação existentes para robôs móveis, pode-se definir três grandes grupos, baseados, cada um: em cálculo de posição, em balizas e por último, em mapas e modelos do ambiente (MCKERROW, 1991; BORENSTEIN, EVERETT & FENG, 1996).

A navegação por cálculo de posição é aquela em que o robô recebe uma trajetória para ser executada e, por meio de odometria (sensores que demonstram a rotação das rodas) ou acelerômetros, calcula a sua posição instantânea, avaliando o erro existente (MCKERROW, 1991). Como em um sistema de otimização, a minimização deste erro aproximará o robô do caminho desejado.

A baseada em balizas é aquela que, por meio de sensores, possibilita dizer ao robô se ele está no caminho certo (BORENSTEIN, EVERETT & FENG, 1996). No caso do uso de balizas utilizando emissores de infravermelho (MCKERROW, 1991) ou ondas de rádio (BORENSTEIN, EVERETT & FENG, 1996), a triangulação entre dois sensores e o robô permite o cálculo da sua posição, conforme a figura 2.6. Também existem aplicações hoje empregando o Sistema de Posicionamemto Global (G*lobal Positioning System)* – GPS que, por meio do cálculo da distância entre o robô em uma série de satélites geo-estacionários, é capaz de localizar o robô na Terra com precisão de metros (BORENSTEIN, EVERETT & FENG, 1996).

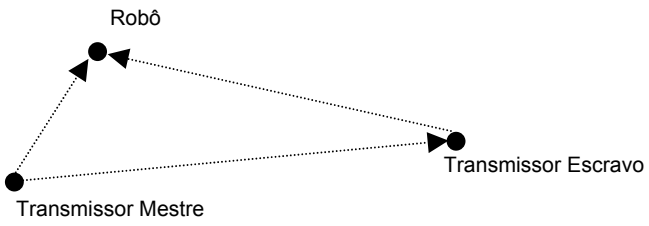


Fig. 2.6 – Triangulação entre balizas e robô

Estes dois primeiros modelos de navegação, os quais não impedem do robô se chocar com algum obstáculo, não serão abordados nesta dissertação, sendo uma excelente referência a obra *Navigating mobile robots: systems and techniques,* de Borenstein, Everett e Feng (1996).

Os métodos baseados em mapas e modelos do ambiente hoje estão divididos basicamente em dois sub-grupos, os de planejamento do caminho e os de uso de sensores (BOTELHO, 1996), conforme descrito a seguir.

2.3.4.1 Navegação baseada em planejamento do caminho

O problema de planejamento do caminho pode ser definido como a busca de um percurso a ser seguido, ou a seqüência de ações a serem tomadas para que o robô possa, saindo de um ponto de partida, chegar a um ponto meta, evitando a colisão com um conjunto de obstáculos conhecidos (JAHANBIN & FALLSIDE, 1988; SCHALKOFF, 1990).

O método de planejamento do caminho tem duas vertentes principais: o global e o local, que devem ser utilizados simultaneamente para que se possa ter um melhor desempenho do robô em um ambiente real.

O planejamento global é responsável pelo mapeamento do ambiente onde está inserido o robô em um modelo simplificado, estático, pré-gravado (CROWLEY, 1985), que lhe permite traçar um caminho mesmo por um local que não consiga perceber com seus sensores, seja por estar fora do alcance ou por estar obstruído por algum obstáculo (FIRBY, 1994). Já o local é responsável pela navegação curta (FIRBY, 1994), baseada nos valores coletados pelos seus sensores. Uma vez modelado o ambiente, o planejamento local será o responsável pela localização do robô neste mundo simplificado e dinâmico, devido à constante atualização dos dados não fornecidos pelo modelo global. Uma maneira de diminuir o custo computacional requerido no tratamento destes modelos é utilizar polígonos na representação dos objetos. Esta representação é aceitável pois, como se deseja que o robô não se aproxime muito do objeto, os seus pequenos detalhes são irrelevantes (JAHANBIN & FALLSIDE, 1988), sendo as aproximações geométricas, por meio de linhas retas, uma excelente solução.

Existem diferentes técnicas matemáticas para se alcançar o planejamento do caminho, sendo os *Roadmaps*, as Decomposições em Células e o Campo Potencial as principais. Entre os desenvolvimentos mais recentes na área de Campos Potenciais estão a Navegação por Campo de Força Virtual (VFF) e a Navegação por Histograma de Campo Vetor (VFH), ambos desenvolvidos na Universidade de Michigan (BORENSTEIN & KOREN, 1991).

A Navegação por Campo de Força Virtual é baseada em duas etapas. A primeira é definir um histograma bidimensional, onde cada ponto - chamado de célula - contém uma certeza de existir um obstáculo c_{ij}. Este histograma passa então a ser atualizado a partir das informações fornecidas pelos sensores, que verificam a certeza ou não da existência de obstáculo em cada célula. Assim, as que representarem os obstáculos, terão valores maiores de certeza do que as que não representarem.

O próximo passo é aplicar o conceito de campo potencial, onde as células que recebem valores de certeza de um obstáculo exercem forças repulsivas sobre o robô, ao mesmo tempo em que forças atrativas são exercidas pelas células que representam a meta a ser alcançada. A resultante destas forças fornece a direção a ser tomada pelo robô (BORENSTEIN & KOREN, 1991).

O método de Navegação por Histograma de Campo Vetor tenta resgatar uma falha do modelo anteriormente exposto: a perda de informação quando se passa do histograma para o campo de forças. O método consegue evitar este problema por meio de uma redução de dados em dois estágios. Assim, utiliza-se três níveis para se determinar a posição a ser seguida, ao invés de apenas dois como no modelo de Campo de Força Virtual. O primeiro nível é o histograma utilizado e atualizado tal como no método anterior; o segundo é a transformação do histograma cartesiano em polar, centrado no próprio robô, dividido em n setores angulares, onde cada qual tem uma densidade polar de obstáculos, calculada a partir da disposição dos dados no histograma cartesiano. O terceiro nível é a definição da direção a ser tomada pelo robô, que passa a ser calculada através do mapeamento das densidades polares de obstáculos (BORENSTEIN & KOREN, 1991).

Ambos os métodos têm uma vasta aplicação mesmo que alguns autores não o recomendem isoladamente, em grandes ambientes, uma vez que sem um certo raciocínio embutido o robô poderá chegar a uma rota ineficiente (FROOM, 1995).
Vale lembrar que os processos baseados em construção de mapas são ineficientes quando trabalham em ambientes onde os obstáculos mudam freqüentemente de lugar (FIRBY, 1994), ou seja, são menos flexíveis a situações tipicamente dinâmicas.

2.3.4.2 Navegação baseada em sensores

A navegação baseada em sensores é executada através da conexão de dois sistemas: o de percepção, responsável pelo tratamento e envio dos dados adquiridos pelos sensores e o sistema de controle, capaz de tomar a decisão de qual caminho seguir (BOTELHO, 1996). O emprego de modernas técnicas de inteligência artificial tem obtido sucesso no controle de tais robôs, como o projeto ALVINN – Autonomous Land Vehicle In a Neural Network, desenvolvido na *Carnegie Mellon University* (POMERLEAU, 1993). Neste projeto é utilizado um controlador baseado em redes neuronais, capaz de identificar o trajeto a ser seguido por um veículo através da imagem produzida por duas câmeras de vídeo. A rede consegue distinguir o que é obstáculo e o que é estrada, permitindo o livre trânsito do veículo de maneira autônoma.
Da mesma forma, a Lógica Difusa *(Fuzzy Logic)* vem sendo bastante utilizada no desenvolvimento de controladores de robôs móveis em várias universidades, como: *Escola Federal de Engenharia de Itajubá* (LAPRADE & LAMBERT-TORRES, 1993), *Purdue University* (PAN, PACK, KOSAKA et al, 1995), *Texas A&M University* (YEN & PFLUGER, 1995), *Lausanne Université* (GODJEVAC, 1995), *University of Salford* (GHANEA-HERCOCK & BARNES, 1996), *Universidade Federal do Rio Grande do Sul* (BOTELHO, 1996) e, *Université Libre de Bruxelles* (SAFFIOTTI, RUSPINI & KONOLIGE, 1994; SAFFIOTTI,1997, SAFFIOTTI, RUSPINI & KONOLIGE, 1999) apenas para citar alguns exemplos.
A lógica aplicada ao emprego de controladores difusos em robôs é baseada em sistemas difusos, compostos de regras e conjuntos, selecionados conforme o entendimento dos autores na resolução de diferentes tarefas.

Os sistemas difusos, comparativamente aos sistemas de planejamento de caminho, gozam da capacidade de trabalhar com imprecisões e sistemas não-lineares (KLIR & YUAN, 1995), conforme Zadeh, o pai da lógica difusa, demonstrou através do princípio da incompatibilidade, que diz:

> *"A medida em que a complexidade de um sistema aumenta, nossa habilidade em fazer precisas e ainda significantes afirmações sobre seu comportamento diminuem até alcançar um limite, através do qual precisão e significância (ou relevância) começam a ser características quase mutualmente exclusivas."* (COX, 1994)

Desta forma Zadeh começou a representar o mundo na forma de variáveis lingüísticas, capazes de embutir toda a imprecisão e incerteza do mundo real, surgindo assim a lógica difusa. A sua descoberta abriu uma nova frente de pesquisa, onde matematicamente passou a ser possível trabalhar com valores qualitativos, uma vez que a matemática clássica nunca será capaz de diferenciar se uma pessoa é magra ou gorda sem a imposição de limites rígidos.

A lógica difusa, seus conceitos e a sua aplicação na navegação de robôs, especificamente em robôs reativos, são objeto do capítulo 4 desta dissertação.

2.4 Conclusão: aplicações e tendências

Conforme descrito anteriormente, quando se falou sobre o *Sojourner* da NASA/JPL, pôde-se imaginar a utilização dos robôs móveis em aplicações típicas de telepresença, ou seja, aquelas que estendem os sentidos humanos a locais remotos, dando aos cientistas ou usuários uma compreensão muito mais intuitiva do ambiente ou sistema em estudo (STOKER, BARCH, HINE III et al, 1995).

A telepresença vem alcançando, a cada dia, novos usos nos diversos campos do conhecimento humano, o que poderá favorecer enormemente o emprego de robôs móveis. Exemplos são os trabalhos desenvolvidos no *Georgia Institute of Technology* voltados à aplicação militar, como o veículo teleoperado de (BENTIVEGNA, ALI, ARKIN et al, 19__). Na mesma linha se tem o trabalho desenvolvido por (STOKER, BARCH, HINE III et al, 1995), que explorou a fauna subaquática Antártica através do

robô TROV, operando submerso no mar gelado, explorando locais onde não há a mínima chance de sobrevivência do ser humano, pelo menos por longos períodos.

Paralelamente à telepresença existem diferentes usos destinados aos robôs móveis, como em tarefas de reconhecimento, logística e operações militares, comboio e escolta (KHALED & ARKIN, 1994; BENTIVEGNA, ALI, ARKIN et al, 19__), em operações de resgate, em minerações, em processos de escavação automáticos (BRUMITT, COULTER & STENTZ, 1992) e em estações de pilhagem (KHALED & ARKIN, 1994). Alguns autores ainda reforçam a larga utilização industrial, como no transporte de peças e limpeza de prédios. (DEV, KRÖSE & GROEN, 1997; MONDADA & FRANZI, 1993; FIRBY, 1993). Outras áreas de possível aplicação também apontadas são na saúde, como enfermeiros (BORENSTEIN & KOREN, 1987), na colheita e seleção de frutas (MONDADA & FRANZI, 1993) em veículos espaciais (PELL, BERNARD, CHIEN et al, 1996), na vigilância de prédios como museus e bancos, e como guia em lojas ou exposições (PIAGGIO, SGORBISSA, VERCELLI et al, 1997). O combate ao fogo, a exploração dos mares também são áreas factíveis de utilização de robôs autônomos (GEVARTER, 1984).

Lavery (1996), levantou que "*logo, será possível comprar robôs que operarão com pouco ou nenhum conhecimento a priori para uso em plantações, minas de cobre e carregamento em docas*" o que, segundo ele, possibilitará aos robôs móveis superarem em número os seus irmãos mais velhos - os robôs fixos de chão de fábrica - na razão de 4 para 1.

Um bom exemplo de pesquisas nesta área é o projeto *Demeter*, com participação da *NASA*, *Carnegie Mellon University* e *New Holland*, com o objetivo de produzir um robô autônomo capaz de colher alfafa em uma fazenda, sem intervenção do homem (LAVERY, 1996). Por outro lado, alguns psicólogos, biólogos e etnólogos tem utilizado robôs móveis para simular e validar estruturas observadas no mundo real (MONDADA, FRANZI & IENNE, 1993). Eles embutem em vários robôs, que compartilham um mesmo ambiente, algumas características de relações entre membros de uma mesma comunidade e observam as suas ações.

Braitenberg ficou famoso com sua obra *Vehicles* (BRAITENBERG, 1984) por simular, em veículos autônomos, características psicológicas dos humanos como amar, repudiar e agredir, baseado apenas nas suas ações e reações.

É inegável que também a nova corrida espacial, decorrente da conquista de Marte, será um elemento catalisador do processo de pesquisa e desenvolvimento de robôs móveis, uma vez que estão programados mais seis lançamentos de pequenos robôs, em 2001, 2003 e 2005 (HAYATI,1996).

Apesar de alguns pesquisadores discordarem (GATES, 1995; KOREN, 1985) a utilização de robôs autônomos em lares também deverá se tornar uma realidade, seja na realização de tarefas domésticas, seja como meio de entretenimento (GATES, 1995), ainda que não consigamos implementar sistemas inteligentes de elevada complexidade nos seus controladores (MCFARLAND & BÖSSER, 1993).

Por fim, pode-se perceber que, comparativamente às primeiras gerações, cujas aplicações se restringiam à soldagem de chapas, pintura de peças e preenchimento de pallets (MONDADA & FLOREANO, 1996), os robôs móveis terão uma aplicação cada vez maior na sociedade atual.

Capítulo 3 – Definição do problema a ser tratado

3.1 Introdução

Neste capítulo será abordado a definição do problema tratado neste trabalho, passando primeiramente pela conceituação de telepresença e teleoperação, levantando, inclusive, exemplos de aplicações destas tecnologias.
A seguir, é determinado o problema do desenvolvimento de um protótipo de sistema de telepresença, listando os requisitos que devam ser cumpridos na busca da solução, bem como as limitações impostas ao desenvolvimento do projeto. Por fim, com base na metodologia apresentada por Back (1983), são elencadas as etapas envolvidas no desenvolvimento deste trabalho.

3.2 Telepresença e teleoperação

A telepresença é o objeto principal deste trabalho. Ela consiste em um refinamento da teleoperação: enquanto esta é a ação de utilizar uma máquina –teleoperador - com capacidade de agir a distância, sob o controle de um humano (GEVARTER, 1984), a primeira torna possível, além da operação, a sensação de estar em um ambiente distante do operador (GIBILISCO, 1994).
O conceito de Gibilisco é também compartilhado por (STOKER, BARCH, HINE III et al**,**1995) quando diz que "*...a telepresença garante a extensão dos sentidos dos seres humanos a locais remotos*", podendo estes estar a metros ou milhares de quilômetros, como ocorreu com o robô *Sojourner*, já comentado no Capítulo 2.
Foi Robert Heinlein, em seu romance *Waldo*, da década de 40, quem primeiro utilizou o conceito de telepresença. Na obra, Waldo F. Jones é um cientista que, para superar os efeitos de uma séria doença degenerativa, constrói um laboratório que orbita em volta da Terra, de onde comanda os "*waldoes*", instrumentos na forma de braços mecânicos, os quais lhe fornecem a capacidade de executar operações a distância.(KAC, 1993).

Marvin Minsky com o artigo “Telepresence” de 1980, é outra referência a ser considerada. Nele, o autor defendia esta nova área de pesquisa afirmando “*...o quanto poderíamos ter aprendido com um veículo permanentemente na Lua*”, (KAC, 1993) destacando que a telepresença era capaz de servir muito mais à ciência do que se imaginara até então.

Mesmo existindo registros de patentes norte-americanas já na década de 50 (GROOVER, WEISS, NAGEL et al, 1989), a primeira grande aplicação de teleoperação no mundo se deu na ex-União Soviética, em 1970, com o lançamento do Lunokhod 1 (figura 3.1), um veículo sem tripulantes para exploração do solo lunar, com pouco mais de 2 metros de comprimento. Na Terra, na base de Baikonur, uma equipe composta de comandante, navegador, engenheiro, rádio operador e motorista comandavam a sonda através de imagens fornecidas por suas câmeras. O Lunokhod 1 operava a uma velocidade de 22 m/h, possuía tração nas suas oito rodas e dispunha de um controle de baixo nível instalado no veículo que era inibido por comandos dados pela equipe em Terra (MCKERROW, 1991).

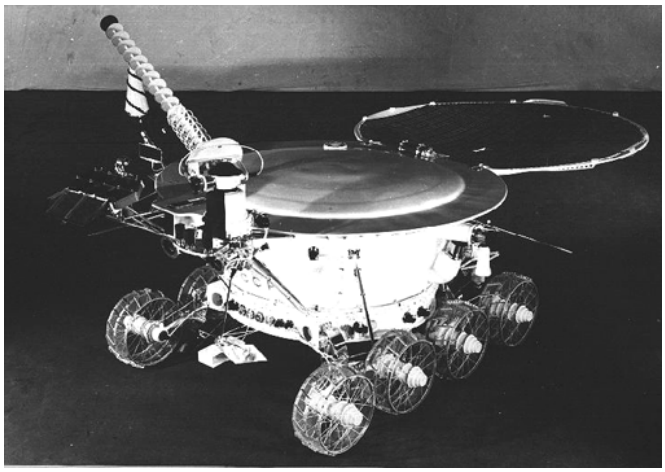


Fig. 3.1 O veículo lunar Lunokhod 1

Outro evento importante nesta área ocorreu em 1976, quando a sonda norte-americana Viking I pousou em Marte para realizar análises do solo, tirar fotos e monitorar as condições atmosféricas do planeta, tudo remotamente executado a partir de comandos enviados da Terra (MCKERROW, 1991).

Em 1979, com o acidente na usina nuclear *Three Mile Island*, na Pensilvânia (EUA), a telepresença encontrou uma nova frente de aplicação. Um veículo de reconhecimento

remoto, chamado de RRV, desenvolvido na *Carnegie-Mellon University* foi utilizado para inspecionar a estrutura dos prédios e retirar a água radioativa que vazou do reator. Câmeras instaladas no robô transmitiam aos operadores a sensação de estarem dentro da usina (MCKERROW, 1991).

Os robôs teleoperados vêm, desde então, ganhando grande aplicação no mundo todo, como o *TROV*, da NASA, para pesquisa subaquática (STOKER, BARCH, HINE III et al, 1995), o *ROSA*, utilizado submerso na inspeção de linhas de vapor de diversas companhias (MCKERROW, 1991) e o *Sojourner* do JPL, responsável por "*levar a telepresença às massas*" (KAC, 1998) quando da sua exploração do planeta Marte, em julho de 1997.

3.3 Proposta do problema

Diante do que foi explanado até aqui fica patente que, em muitos casos onde se torna impossível a presença humana - seja por risco de sua integridade física, ou por elevado custo ou falta de conhecimento das reais necessidades para a manutenção da vida, ou ainda por meras restrições de espaço - os sistemas de telepresença são a solução hoje encontrada.

Além do fato de tornar real a interação com novos mundos a telepresença, ao possibilitar a quebra de todas as amarras que limitam naturalmente a vida humana, e ao mesmo tempo, eliminar as ameaças inerentes a esta ação, torna-se um vigoroso estimulante a nossa já fértil imaginação. Isto se traduz na possibilidade de se ter missões espaciais inéditas, de explorar os abissais mais profundos dos oceanos, de percorrer tubulações que transportam fluidos a pressão e temperatura elevadas e, até, viajar pelo interior do corpo humano. Aliada à Realidade Virtual, a telepresença pode levar o homem a mundos hoje apenas imaginários, criando situações novas, completamente diferentes das que estamos acostumados a experimentar.

O problema a ser abordado neste trabalho é o de definir e implementar um protótipo para desenvolvimento de sistema de telepresença, utilizando conceitos e técnicas de inteligência artificial, para aplicação em locais insalubres ou inacessíveis ao homem.

O protótipo deve ser composto por duas unidades básicas. A primeira, chamada de unidade de ação remota (*RAU - Remote Action Unit*), transmitirá os dados coletados para o usuário, enquanto a segunda, denominada unidade de processamento de informações (*IPU – Information Processing Unit*), será responsável pela visualização destes dados e pelo comando da unidade remota, por meio de instrumentos de interfaceamento homem-máquina, como *joysticks*.

As duas unidades devem ser interfaceadas com um software que possibilite o envio e recebimento de sinais entre elas, por meio de ligação física – fios – ou por ondas de rádio, com o objetivo principal de melhoria da manobrabilidade da unidade de ação remota.

A unidade de ação remota deve conter um sistema interno de controle, baseado em técnicas de inteligência artificial, de tal sorte que auxilie o usuário na tomada de decisão. Este controle, que atuará apenas na manobra da unidade, não tomará decisões por si só, cotejando a sua visão do ambiente onde está inserido com os comandos do usuário, de forma a definir o caminho a ser seguido. Assim, no caso de haver um obstáculo ao lado direito da unidade, por exemplo, o sistema de controle deverá evitar tomar esta direção, mesmo que desejada pelo operador.

Os dados a serem transmitidos entre as duas unidades será definido como imagem, da unidade remota para a de processamento e sinal de comando, quando no sentido oposto.

São também pré-requisitos de projeto, a serem cumpridos por este protótipo:

a) utilizar materiais que não sofram grandes alterações físico-químicas em ambientes hostis, uma vez que a aplicação é definida para lugares insalubres ou de difícil acesso;
b) adotar sistema eletro-eletrônico protegido para trabalhar em condições adversas, não permitindo que ocorram avarias durante a sua operação;
c) priorizar dimensões reduzidas, a fim de que possa ser utilizado em locais onde haja acesso fisicamente restrito ou limitações de manobrabilidade;
d) optar por um modelo leve, de tal forma que se possa utilizar comandos e equipamentos com baixa potência, resultando em menor consumo de energia;

e) dotar a unidade remota de equipamentos e comandos de fácil aquisição no mercado, visando uma diminuição dos custos envolvidos no projeto;
f) empregar, na unidade remota, o sistema de direção que garanta elevado grau de manobrabilidade;
g) prover a unidade remota de processamento paralelo, possibilitando o levantamento do ambiente e a leitura simultânea de todos os sensores envolvidos;
h) priorizar, na definição dos comandos a serem enviados pela unidade de processamento, a elevada interação entre operador e unidade remota;
i) projetar a unidade remota para resistir a impactos durante a sua operação; e
j) capacitar o sistema de gerenciamento do protótipo, baseado em software para a plataforma PC, para que possa enviar os sinais de comando e receber os sinais de vídeo em tempo real, ou seja, simultaneamente.

O protótipo utilizará um processamento local, chamado de baixo nível, como utilizado em diversos trabalhos (ALVES, RESENDE, LEITE et al, 1993; MONDADA, FRANZI & IENNE, 1993; SAFFIOTTI, RUSPINI & KONOLIGE, 1994; FIRBY, 1994; STOKER, BARCH, HINE III et al, 1995; HAYATI, VOLPE, BACKES et al, 1997; CHENG & ZELINSKY, 1997; PIAGGIO & ZACCARIA, 1997) devendo ser baseado na utilização de sensores, recorrendo a técnicas de inteligência artificial.

A interação entre os módulos do protótipo pode ser representada de forma esquemática, como na figura 3.2. Deve existir um canal de comunicação *full duplex*, entre computador e robô mantendo o requisito de projeto de um sistema em tempo real. Assim, todo comando dado pelo usuário representará, naquele exato momento, uma ação do computador sobre o robô, com a conseqüente reação, através da imagem transmitida.

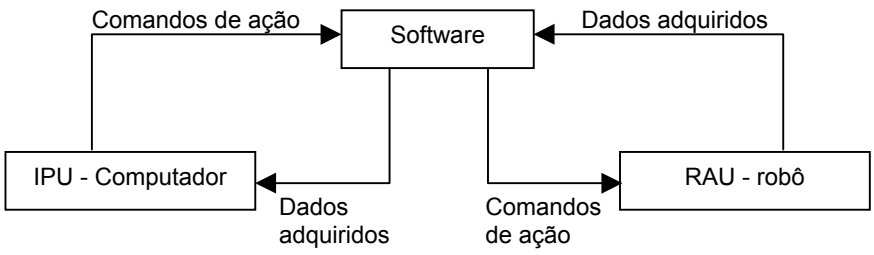


Fig. 3.2. Interação entre os módulos do sistema proposto

3.4 Limitações impostas ao desenvolvimento do protótipo

Como na realização do projeto de um produto, juntamente com os requisitos a serem cumpridos pelo modelo preliminar, deve-se listar as limitações impostas. Por tratar-se de um protótipo que visa, inicialmente, servir de base para estudos futuros na área acadêmica, e pelo tempo limitado de que se dispõe para a elaboração desta dissertação, utilizou-se o conjunto de limitadores de projeto definidos como seguem:

a) a unidade remota operará em uma superfície plana, livre de depressões ou elevações;
b) não será levado em consideração o movimento de marcha a ré da unidade remota, devendo ser encontrado outro mecanismo para retornar a uma posição previamente alcançada;
c) deve-se limitar o número de direções a serem exploradas pelos sensores. Em (ERKAMP, 1996), a pesquisadora utiliza apenas 3 direções para construção de mapas, porém deve-se lembrar o que é colocado por (TRIVED, ABIDI, EASON et al, 1990), onde diz que "*...o acréscimo no número de sensores possibilita um aumento da precisão na estimação de um local e distância de um objetivo*". Um bom valor é o utilizado por (BOTELHO, 1996), que chegou a resultados satisfatórios empregando 5 direções.

3.5 Conclusão: etapas de desenvolvimento da solução

Partindo-se do problema ora definido, a busca da solução se dará da seguinte forma: primeiramente se passará a etapa do projeto preliminar, onde são criadas várias soluções possíveis para a definição do robô, elegendo-se a melhor após análise detalhada de cada uma.

A segunda etapa concerne no detalhamento da solução eleita na primeira fase, estabelecendo-se um planejamento do projeto adotado e especificando-se componentes.

Por fim a etapa seguinte, provavelmente a mais entediosa, é caracterizada pela revisão e testes do produto. Uma vez definido o protótipo no projeto detalhado, este segue para uma bateria de testes, utilizando-se basicamente situações que possam levar o robô a valores ambíguos, encontrados na literatura.

No próximo capítulo, serão descritas as principais técnicas utilizadas na resolução do problema, apresentando na seqüência as soluções para o hardware e software empregados.

Capítulo 4 – Procedimentos metodológicos utilizados para a resolução do problema

4.1 Introdução

Apresentada, no capítulo anterior, a definição do problema, passa-se a descrição das principais ferramentas e técnicas utilizadas na busca de sua solução, cumpridos os requisitos e limitações de projeto.

Primeiramente, serão analisados aspectos relacionados a dinâmica do robô RAU, enfatizada a formulação matemática de seu movimento, ou seja sua cinemática. Em um segundo momento se discutirá sobre as técnicas de inteligência artificial visando a definição do controlador da unidade, a fim de tornar o sistema capaz de encontrar o melhor caminho para o deslocamento do robô em um local desconhecido.

Exposta a base teórica utilizada na confecção do sistema, os capítulos seguintes conterão a explicação detalhada da solução encontrada.

4.2 Cinemática dos Veículos

A cinemática é conhecida como a parte da física que estuda os movimentos, relacionando posições, velocidades e acelerações, desprezadas as forças de ação ou reação envolvidas (MCKERROW, 1991).

Está presente no campo da robótica, sendo imprescindível na determinação de trajetórias e posições de braços mecânicos ou manipuladores em uma célula de trabalho. O grande número de graus de liberdade envolvidos requer a utilização de duas formas diferentes de solução para estes problemas: a cinemática direta e a reversa (ANDEEN, 1988). Todavia, como se verifica nesta dissertação, esses mesmos conceitos tem grande aplicação em robôs móveis quando da necessidade de seu estudo cinemático, com o objetivo de determinar matematicamente o seu funcionamento.

Na definição preliminar da solução, levou-se em conta os pré-requisitos elencados no capítulo anterior, especificamente os que determinam um projeto simples e com

sistema de direção que garanta elevado grau de manobrabilidade, presentes, em pról da simplificação, e as limitações relativas a sua utilização em superfície plana, livre de depressões e a não permissão do movimento em marcha à ré.

Na literatura são encontradas diversas topologias de sistemas de direção aplicados a robôs, divididas em duas grandes áreas: a dos que se locomovem com uso de pernas e a daqueles que utilizam rodas.

McKerrow (1991) apresenta diversas configurações possíveis, questionando-as quanto a sua cinemática e controle. O autor analisa os veículos que se locomovem por meio de esteiras, 4, 3 e 6 rodas, e seus comentários subsidiaram a decisão da topologia a ser adotada.

Para os veículos de 4 rodas, McKerrow comenta que a forma mais comum é aquela que encontramos em alguns automóveis, ou seja, com direção nas rodas da frente e tração nas rodas de trás, ressaltando que durante uma curva deve existir, pela própria natureza do movimento circular, alguma forma de controle para as diferentes velocidades das rodas durante o movimento, como os diferenciais dos veículos, conforme pode ser demonstrado pela figura 4.1.

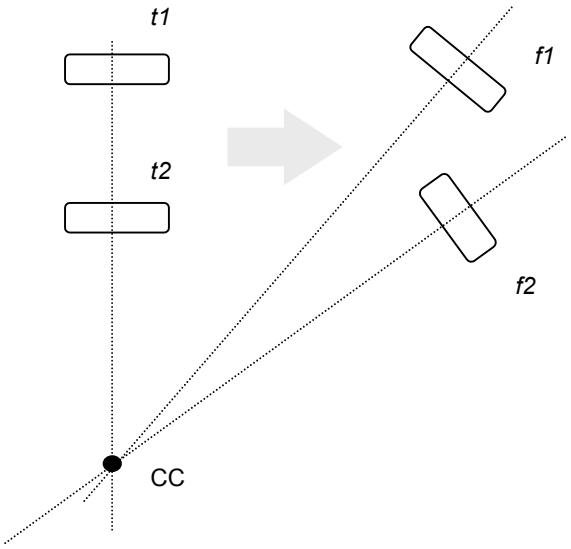


Fig. 4.1 – Movimento circular de um veículo com 4 rodas

O centro da curva (ponto *CC*) é único, garantido pela teoria do movimento de veículos, que define o rolamento puro para cada roda, sem escorregamento (BUNKER, 1968) e é dado pela intersecção do eixo perpendicular que parte de cada

uma das rodas do veículo, o que nos leva a perceber que cada uma tem um raio de curvatura próprio, diferente entre si. Com isto, levando-se em consideração que o veículo perfaz o movimento circular a uma velocidade constante, chegamos a conclusão de que a velocidade angular de cada roda, por estar relacionada com o raio de curvatura, deve ser diferente. Assim, a velocidade da roda *t1* é diferente da roda *t2*, que por sua vez não é igual a *f1* ou a *f2*. Este problema apontado por McKerrow é agravado quando o veículo conta com tração nas quatro rodas, necessitando de complexos sistemas de transmissão que permitam o deslizamento de uma roda em relação a outra.

Uma outra solução para esta situação, porém não menos complexa, é a adoção de um motor para cada roda, como no desenvolvido no *Sojourner* (HAYATI, 1996; HAYATI, VOLPE, BACKES et al, 1997), no entanto, isto provoca a elevação dos graus de liberdade do robô, o que implica na sofisticação do sistema de controle.

Usando-se 3 rodas no veículo estes problemas são reduzidos, mas torna-se necessário, entretanto, um controle de velocidade das rodas motrizes bem mais preciso (MCKERROW, 1991). Neste caso tem-se a vantagem de ser desnecessário a utilização de um sistema de suspensão, pois as rodas sempre estarão em contato com o solo. A figura 4.2 representa um veículo de três rodas fazendo uma curva. Enquanto as rodas motrizes, *f1* e *f2*, rodam em direções opostas, a roda traseira *t1* movimenta-se livremente em relação ao ponto *CE* possibilitando o giro completo sobre o eixo dianteiro.

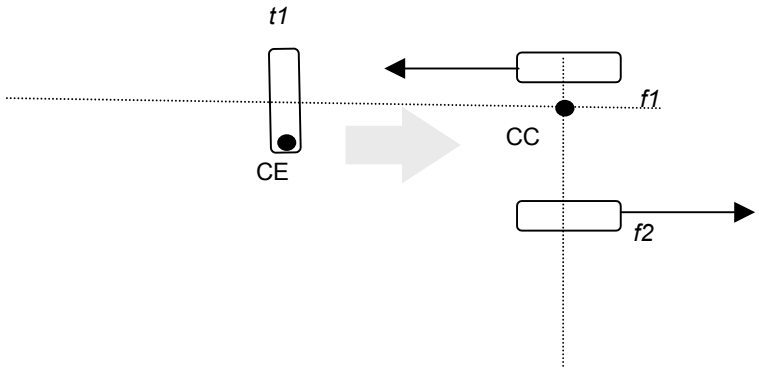


Fig. 4.2 – Representação de um veículo com 3 rodas

Como no caso do veículo de 4 rodas, o centro da curva, indicado pelo ponto *CC,* não é fixo, mas sim diretamente relacionado ao ângulo entre a roda traseira e o eixo

longitudinal do veículo, que é definido pela diferença entre as velocidades das duas rodas dianteiras.

Existem variantes desta configuração onde a roda isolada é também a motriz, utilizando um mecanismo que controla a sua direção, elevando-se, mais uma vez, a complexidade do sistema.

Os veículos de 6 rodas e os com esteiras têm o seu sistema de direção definido como esterçamento com escorregamento, uma vez que para realizar uma curva existe deslizamento entre as áreas de contato pneu-piso.

Os veículos de esteira são mais indicados para uso em terrenos irregulares, possibilitando uma maior superfície de contato entre o veículo e o solo (GIBILISCO, 1994), conforme pode ser visto na figura 4.3

Fig. 4.3 Veículo de esteira transpondo um obstáculo (foto: Engesa)

Igualmente os veículos com 6 rodas, desde que tenham um sistema de suspensão independente, capaz de assegurar o aumento e manutenção da área de contato, são indicados para locais onde existem irregularidades no solo. A vantagem destes dois modelos reside na capacidade de executarem um giro completo sobre o próprio eixo, conforme mostrado na figura 4.4, sendo possível, portanto, cumprir o requisito de projeto que determina o emprego de sistema de direção que garanta elevado grau de manobrabilidade.

Outro ponto positivo desta topologia com relação as anteriormente mostradas é que o seu centro da curva (*CC*) é fixo, não dependendo de ângulos entre os eixos do veículo.

Isto garante maior estabilidade no controle direcional do veículo e simplifica a sua formulação matemática, vindo ao encontro das premissas formuladas para este projeto.

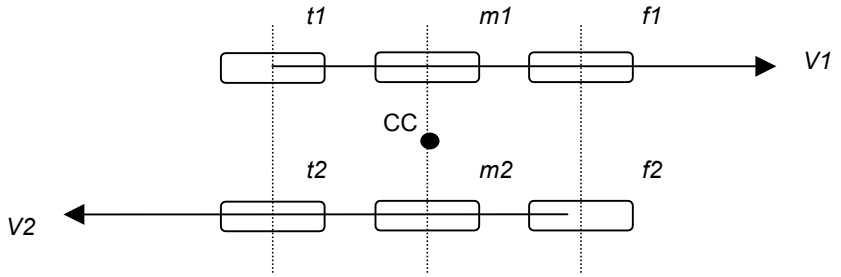


Fig. 4.4 – Veículo com esterçamento com escorregamento,
com 6 rodas, fazendo uma curva

Uma vez que o esterçamento com escorregamento se adapta perfeitamente à operação em superfícies planas e - pelo fato de não existir a marcha a ré - permite que o veículo realize um giro sobre o seu próprio eixo, retornando a uma determinada posição, sem o risco de se chocar com obstáculos, entendemos ser esta a topologia mais adequada ao projeto.

Definido o modelo, passou-se a estudar o comportamento cinemático dos veículos com esterçamento por escorregamento, como segue. Vale notar que pelo fato das forças envolvidas serem de baixa magnitude, não se levou em conta o estudo dinâmico do veículo.

4.3 Cinemática de Veículos com Esterçamento por Escorregamento

Para definir matematicamente a cinemática dos veículos com esterçamento por escorregamento, utilizaremos algumas condições:

1. a superfície de contato é plana;
2. a velocidade do veículo é baixa e constante; e
3. não há deformação na interface veículo-piso.

Assim, pode-se eliminar forças de inércia e garantir que a pressão na superfície de contato é perfeitamente uniforme.

De acordo com a figura 4.5, podemos traçar algumas definições sobre o movimento curvilíneo de um veículo com esterçamento por escorregamento.

Verifica-se que, para realizar uma curva de raio *R*, a uma velocidade *v*, em relação ao centro da curva *CC*, o veículo deve ter velocidades diferentes para cada esteira, chamadas de *v1* para a esteira externa e *v2* para a esteira interna à curva.

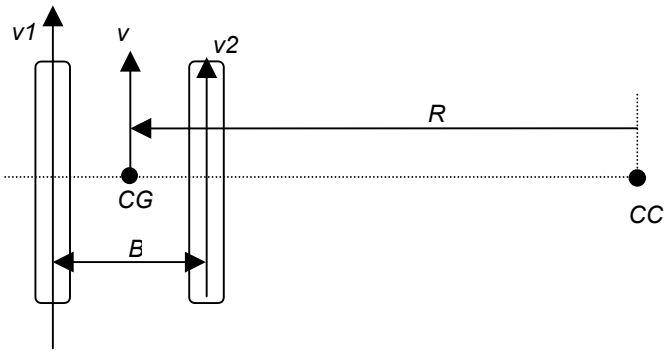


Fig. 4.5 - Cinemática do movimento curvilíneo para veículo com esterçamento por escorregamento

As velocidades podem ser definidas matematicamente como sendo:

$$v_1 = \omega_P(R + 0{,}5B) \qquad \text{e} \qquad v_2 = \omega_P(R - 0{,}5B) \qquad \text{(eq. 4.1)}$$

onde: ω_P é a velocidade angular do veículo com relação ao centro de curvatura *CC*.

Relacionando as duas equações 4.1 entre si, através do raio de curvatura *R*, tem-se a seguinte relação:

$$R = B\frac{v_1 + v_2}{2(v_1 - v_2)} = B\frac{n_1 + n_2}{2(n_1 - n_2)} \qquad \text{(eq. 4.2)}$$

de onde se chega a:

$$\rho = \frac{R}{B} = \frac{n_1 + n_2}{2(n_1 - n_2)} \qquad \text{(eq. 4.3)}$$

onde n_1 e n_2 são as rotações das respectivas rodas motrizes e ρ é o chamado raio relativo de giro.

Por fim, para que se possa obter o giro sobre o próprio eixo, é necessário que o raio relativo seja igual a zero, o que nos leva a seguinte conclusão matemática:

$$R = 0 \qquad \rightarrow \qquad \rho = 0$$

$$\frac{\mathrm{n}_1 + n_2}{2(n_1 - n_2)} = 0 \qquad \forall\, B$$

O que fornece:

$$n_1 = -n_2$$

Da mesma forma, podemos relacionar a velocidade angular das rodas motrizes com a velocidade angular de rotação do veículo, permitindo a definição do ângulo desejado na realização da curva.

Podemos definir a velocidade do roda motriz em relação ao solo como sendo:

$$V_r = \omega_r r_r \qquad \text{(eq. 4.4)}$$

onde V_r é a velocidade linear, ω_r a velocidade angular e r_r o raio, todos relativos a roda motriz, como pode ser observado na figura 4.6

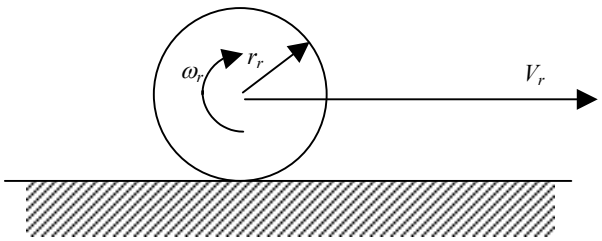


Fig 4.6 – Movimento da roda motriz

Assim, como V_r e V_1 fornecidas pela equação 4.1 são iguais, obtemos:

$$V_r = V_1$$
$$\omega_r r_r = \omega_p (R + 0{,}5B)$$

Como desejamos o giro sobre o próprio eixo, $R = 0$, temos:

$$\omega_r r_r = 0{,}5\omega_p B$$
$$\frac{2\omega_r r_r}{B} = \omega_p \qquad \text{(eq. 4.5)}$$

Logo, como ω_p é a variação do ângulo da direção do veículo na unidade do tempo, escrevendo-a em função da largura do veículo, velocidade angular e do raio das rodas motrizes, podemos definir a direção a ser obtida após realizar uma curva, dado um tempo "t".

4.4 Lógica Difusa

A teoria clássica dos conjuntos, relaciona os seus elementos através de funções pertence/não pertence. Lotfi Zadeh, através de seu trabalho (ZADEH, 1965), questionou estes preceitos afirmando que, no mundo real esta relação não se dava de forma exata, com valores limites discretos. Deste questionamento surgiu a lógica difusa, que abriu um fértil e extenso campo de discussão e pesquisa.

Zadeh definiu os conjuntos difusos como uma classe com graus contínuos de pertinência, ou seja, dado um elemento $\boldsymbol{a}$, do espaço $\boldsymbol{A}$, pertencente a um conjunto difuso $\boldsymbol{X}$ – percebe-se aqui a relação com a teoria clássica dos conjuntos – existe uma função característica, ou de pertinência, dada por $\boldsymbol{f_X(a)}$, que associa a esta pertinência um valor real no intervalo [0,1] (ZADEH, 1965).

A partir de então percebeu-se que esta nova área de pesquisa se adequava satisfatoriamente às situações onde a imprecisão, a ambigüidade ou a incerteza estavam presentes (VIEIRA, MODRO, MARTINS et al, 1998), abrindo frentes de aplicações em áreas como: reconhecimento de padrões, sistemas especialistas, bancos de dados e controladores, entre outros (KLIR & YUAN, 1995).

Os conjuntos difusos são centrados no conceito de variáveis lingüísticas que permitem encapsular, de maneira exeqüível para um computador, as propriedades de imprecisão e aproximação (COX, 1994). Assim, é possível o perfeito manuseio de valores como "*alto*", "*gordo*", "*um pouco*" e "*muito*", por exemplo. Computacionalmente estar-se-á tratando de uma função transformação que, associada ao conceito lingüístico, fornece um valor numérico, ou seja, é a "ponte" entre a representação qualitativa e a quantitativa.

Para esclarecer os conceitos ora apresentados, utilizaremos como exemplo o conjunto difuso "*homem alto*". Admita-se que este conjunto difuso possa ser definido – graficamente – de acordo com a figura 4.7.

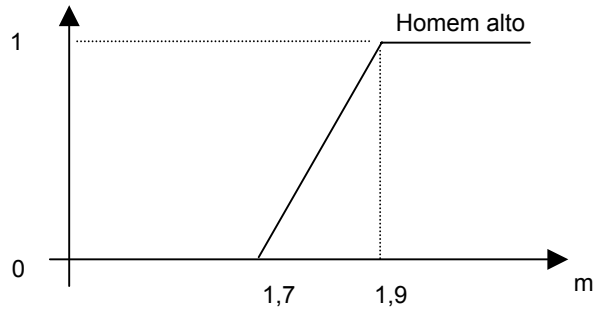


Fig 4.7 – Representação do conjunto difuso *homem alto*

Pode-se perceber que o conjunto partiu da seguinte premissa: "*homem alto é todo aquele que tem mais de 1,9 metro"*. Assim, todos os elementos do espaço *homem* que possuem altura superior a 1,9 metro, pertencerão ao conjunto *homem alto* com grau **1**.

Optou-se arbitrariamente por tratar homem alto a partir do valor de 1,7 metro de altura, dando a este valor 0 para o grau de pertinência, chegando-se linearmente ao valor 1 para 1,9 metro. Cabe ressaltar que o formato adotado na função de pertinência, ou seja, na função que relaciona os valores de entrada (abscissa) e os valores de saída (coordenada), é livre, sendo as curvas parametrizadas: trapezoidais, triangulares e as em forma de sino (KLIR & YUAN, 1995; DRIANKOV, HELLENDOORN & REINFRANK, 1996) as mais utilizadas por contarem, principalmente, com baixo custo computacional (DRIANKOV, HELLENDOORN & REINFRANK, 1996).

Para o mesmo exemplo – se utilizada a teoria clássica dos conjuntos – todo homem com altura inferior a 1,9 metro teria grau de pertinência **0** ao conjunto *homem alto* e sua forma gráfica seria como mostrado na figura 4.8. Na teoria dos conjuntos difusos, entretanto, esta mudança brusca não ocorre, existindo uma suavização entre este limiar fixo e um outro arbitrado, surgindo uma região de transição. Esta característica possibilita aos conjuntos difusos trabalhar com problemas não lineares.

Assim, os conjuntos difusos fornecem uma visão mais flexível de um dado elemento pertencer ou não a um conjunto, principalmente na presença de limites não muito bem definidos, como demonstrado no exemplo ilustrativo acima. Todavia, como comenta Ross (1995), em problemas onde não ocorre imprecisão, a lógica difusa é menos eficiente do que as soluções baseadas em equações matemáticas, capazes de representar o problema em plenitude.

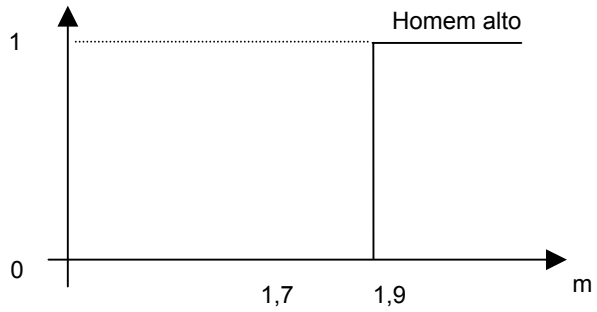


Fig 4.8 – Representação do conjunto clássico *homem alto*

Um estudo mais completo da teoria de conjuntos difusos – o que foge do escopo deste trabalho – pode ser encontrado nas obras de Klir & Yuan (1995) e de Ross (1995).

4.5 Sistemas Difusos

Sucintamente, os sistemas difusos são aqueles que têm, em seu sistema de inferência, o emprego de conjuntos difusos.

Earl Cox, em sua obra (COX, 1994), lista 5 vantagens do uso de sistemas difusos, comparativamente a sistemas clássicos:

a) Habilidade para modelar problemas extremamente complexos: segundo Cox, os sistemas convencionais, ao tratarem de problemas reais, geralmente não lineares, aumentam consideravelmente o custo computacional, enquanto a lógica difusa torna possível resolvê-los utilizando o raciocínio, requerendo menos regras.
b) Aumento da modelagem cognitiva dos sistemas especialistas: a grande vantagem de se trabalhar com sistemas difusos está na forma como é armazenado o conhecimento, possibilitando a associação das regras à forma de pensar dos especialistas.
c) Habilidade para modelar sistemas envolvendo vários especialistas: os sistemas difusos possibilitam relacionar posições e decisões conflitantes, usuais quando são vários os participantes de um projeto de sistema especialista. A forma como as regras interagem em um sistema difuso garantem esta característica.

d) Complexidade do modelo reduzida: os modelos difusos requerem menos regras que os convencionais, traduzindo-se em menor complexidade do modelo. O conhecimento ser explicitado nas próprias regras do sistema é outra característica que beneficia os sistemas difusos.
e) Melhora na manipulação de incerteza e possibilidades: enquanto boa parte dos sistemas especialistas presentes no mercado são baseados na probabilidade bayesiana, os modelos difusos tem como característica intrínseca o tratamento de imprecisão e incerteza, o que se traduz em uma manipulação mais consistente, comparativamente aos sistemas tradicionais.

Importante aplicação de um sistema especialista, baseado em lógica difusa é o encontrado em controladores difusos (KLIR & YUAN, 1995). Para melhor entendimento, este sistema pode ser dividido em 4 módulos básicos, conforme demonstrado na figura 4.9.

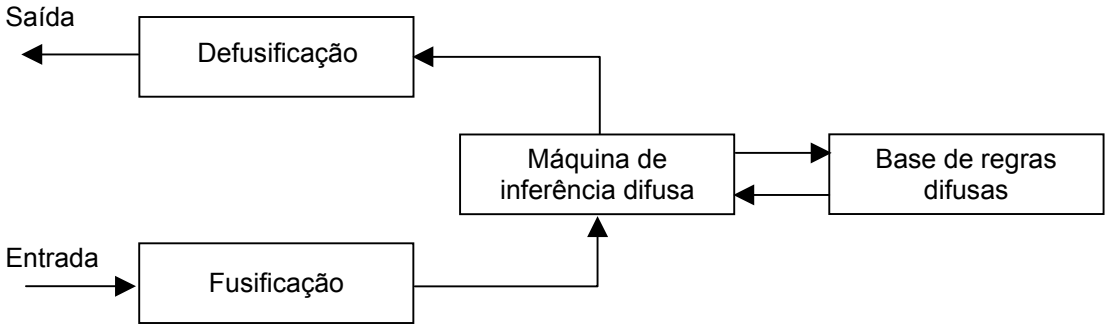


Fig. 4.9: Representação de um sistema difuso (Fonte Klir & Yuan)

Estes módulos operam a cada ciclo com a seguinte seqüência: primeiro a medição das variáveis envolvidas no controle do processo é feita, servindo como entrada ao processo de fusificação. Os dados, já na forma difusa, são enviados à máquina de inferência difusa, que avalia a ação de controle a ser tomada, partindo da base de regras difusas. O resultado passa para a fase de defusificação remetendo um valor numérico ao processo. Por ser esta a base do controlador desenvolvido neste trabalho, cada um destes módulos será descrito com maior profundidade a seguir.

4.5.1 Fusificação

A fusificação é o processo que torna qualquer quantidade numérica – também chamada *crisp* na literatura – em quantidade difusa (ROSS, 1995). É,portanto, uma função que garante certo grau de imprecisão a um valor numérico, mapeando o valor físico de uma variável de um processo em um universo normalizado de discussão (DRIANKOV, HELLENDOORN & REINFRANK, 1996). Isto é necessário para que a entrada do processo se torne compatível com a representação difusa adotada na base de regras.

Matematicamente, a fusificação pode ser descrita como:

$$f_e : [-a, a] \rightarrow \Re$$

onde $\Re$ denota o conjunto de todos os números difusos, no intervalo [– *a, a*] (KLIR & YUAN, 1995), e pode ser interpretada como uma função transformação que leva uma variável de um espaço numérico para um espaço difuso.

Klir e Yuan (1995) destacam também que alguns controladores difusos não fusificam as variáveis de entrada, sendo estas colocadas diretamente na máquina de inferência, como fatos.

4.5.2 Base de regras

A base de regras tem, como objetivo, representar de forma sistemática a maneira como o controlador gerenciará o sistema sob sua supervisão (DRIANKOV, HELLENDOORN & REINFRANK, 1996). Adotando valores lingüísticos iguais aos utilizados por nós quando efetuamos um controle sobre determinado processo, as regras envolvidas apresentam a forma sintática : *if-then*, onde a parcela *if* é relacionada a um estado de entrada, ou antecedente, enquanto que a parcela *then* indica uma ação de controle, ou conseqüente (DRIANKOV, HELLENDOORN & REINFRANK, 1996; ROSS, 1995)

Relacionadas as variáveis lingüísticas do processo, pode-se claramente estabelecer os antecedentes – ou estados do processo – (DRIANKOV, HELLENDOORN &

REINFRANK, 1996) e associá-los com ações de controle. Para exemplificar a utilização da base de regras, toma-se o controle de uma caldeira, onde se estabelece uma relação entre a temperatura interna da unidade com a vazão necessária de água para o seu resfriamento, o que produz as seguintes regras:

if temperatura is alta then vazão de água is alta
if temperatura is baixa then vazão de água is baixa
if temperatura is estável then vazão de água is normal

Aparentemente simplista, o exemplo dado é capaz de manter estável a temperatura em uma caldeira, desde que os conjuntos difusos nele envolvidos tenham uma definição condizente com a realidade. O que vale observar é que para cada estado do processo é relacionada uma ação de controle.

Este conjunto demonstra que a base de regras difusa é bastante intuitiva, se a classificarmos sob o ponto de vista humano. É como o clássico exemplo de estacionar um veículo em uma vaga, virando-o um pouco mais para a esquerda ou para direita. Não existe, neste caso, uma formulação matemática explícita envolvida na solução do problema, mas sim a representação de um conhecimento adquirido pelo operador, o que justifica a dificuldade encontrada pelos iniciantes e a facilidade com que os motoristas com mais prática executam as manobras necessárias.

Portanto é este conhecimento – o da experiência – que as regras difusas representam, sendo perfeitamente inteligível o seu significado.

4.5.3 Máquina de inferência

A máquina de inferência é a responsável pela combinação do dado de entrada – já no formato de número difuso – com as regras difusas existentes, as quais, trabalhando em cima de regras de produção, descrevem o processo de tal forma que se obtenha, através de inferência, o desejado valor de saída (KLIR & YUAN, 1995).

Existem algumas diferentes classificações com relação as máquinas de inferência de um sistema difuso. Driankov, Hellendoorn & Reinfrank (1996) propõem a

classificação em dois grandes grupos: as de inferência baseada em composição e as em regras individuais.

Para o primeiro caso, combinam-se todas as regras da base e faz-se uma única inferência, enquanto para o segundo é feita a inferência regra a regra, aplicando-se *t-normas* ao final do processo para que se obtenha um valor único de saída.

Já Klir e Yuan (1995) classificam as máquinas de inferência de acordo com a avaliação que elas perfazem sobre a base de regras. Existem as dirigidas aos dados, onde são valores de entrada os antecedentes e se busca o conseqüente de uma regra, e as dirigidas às metas, que executa uma busca em sentido inverso. Estas formas de operar são chamadas de *modus ponens* e *modus tollens*, respectivamente.

4.5.4 Defusificação

A defusificação é "*... a fase final do raciocínio difuso.*" (COX,1994), definida como a operação inversa da fusificação, ou seja, tem como objetivo converter cada conclusão difusa do controlador em uma variável numérica (KLIR & YUAN, 1995).

A literatura apresenta diversos métodos de defusificação, tendo por base o centróide, o centro de máxima, a média das máximas, a média do suporte, a soma dos centros, o centro da maior área, o primeiro dos máximos (KLIR & YUAN, 1995; COX, 1994; ROSS, 1995), cada qual com suas vantagens e desvantagens, num vasto campo de aplicações. Neste trabalho iremos nos ater apenas ao método baseado no centróide de área, principalmente pela sua facilidade computacional, entendendo necessário, entretanto, salientar a excelência da obra *Fuzzy Logic with Engineering Applications*, de Ross (1995), para pesquisa dos outros métodos.

O método baseado no centróide – também conhecido como método do centro de gravidade ou do centro de área – é aquele que para um dado suporte *s*, encontra-se o ponto *P*, que divide a área sob a curva *C* em duas sub-áreas de igual tamanho, conforme a figura 4.10.

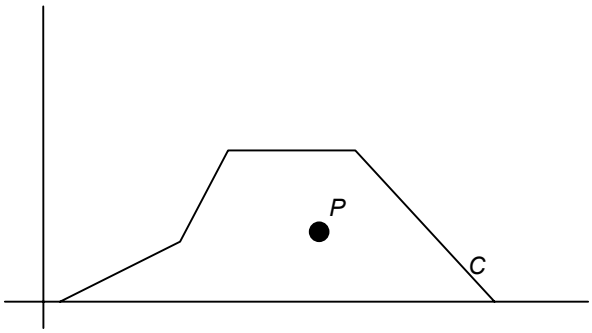


Fig. 4.10 – Ilustração do centróide da área

Matematicamente, as coordenadas do ponto *P* são fornecidas por:

$$x_P = \frac{\sum_i x_i A_i}{\sum_i A_i} \qquad y_P = \frac{\sum_i y_i A_i}{\sum_i A_i}$$

4.6 Aplicações em Controladores

Indiscutivelmente um marco na história da pesquisa em controladores é o trabalho de Mamdani e Assilian (1975), referenciado inclusive por Lotfi Zadeh (1996) quando diz que com este trabalho "*...a exeqüibilidade do controle difuso foi estabelecida por aplicá-lo no controle de uma máquina a vapor.*"

Mamdani e Assilian utilizaram a lógica difusa em um controlador por verificarem que necessitavam de um modelo de cálculo semi-quantitativo (MAMDANI & ASSILIAN, 1975), encontrado nos estudos de Zadeh, sob a forma da lógica difusa. Todavia, o destaque a que fazem jus perante a comunidade científica resulta não somente daquela utilização, mas sim por terem aplicado, com sucesso, o controlador difuso em um sistema completamente não-linear, onde em diferentes pontos de operação são encontradas diferentes características. A utilização de um simples controlador linear levaria, no caso, ao exaustivo método de tentativa e erro (MAMDANI & ASSILIAN, 1975).

Harris & Moore (1994) reforçam a utilização de controladores difusos ao comentar que a teoria de controle moderna propicia o projeto de controladores que: produzem ótima performance com relação a uma situação particular; garantem robustez para

determinados casos, dentro de certos limites e possibilitam fazer estimativas de estabilidade e performance para um dado estado de operação. Segundo os autores, o controlador difuso é capaz de executar qualquer um dos requisitos acima expostos, com a vantagem de ser mais robusto e bem mais simples, não requerendo um complexo modelamento do processo em estudo.

Descrevendo por partes, o controlador difuso é mais robusto por causa da capacidade de trabalhar com imprecisões. Os conjuntos difusos, como foi demonstrado no item 4.4 são baseados em conceitos lingüisticos, que são, "congenitamente" imprecisos.

A simplicidade é garantida porque ele armazena ações de controle do processo baseadas nas ações humanas (DRIANKOV, HELLENDOORN & REINFRANK, 1996) e não necessita, para isto, de complexas formulações matemáticas, bastando para tal o inter-relacionamento entre estados e ações de controle, como foi descrito no item 4.5.2.

Nascida nos anos 60, a lógica difusa só teve o seu grande "boom" a partir de 1990, quando se mostrou aplicável a controladores na indústria, donde se conclui que seu emprego é, ainda, muito recente.

De lá para cá surgiram duas grandes correntes com relação aos controladores: a de que os baseados em lógica difusa promoveriam uma revolução na área de controle, possibilitando resolver complexos problemas com muito pouco esforço; e aquela, antagônica, que defendia a posição que nos casos onde se aplica um controlador difuso a utilização de técnicas de controle convencionais fornece excelentes resultados (DRIANKOV, HELLENDOORN & REINFRANK, 1996).

Dizer com certeza qual vertente tem razão é um árduo trabalho, no entanto o que se percebe é que os controladores difusos devem aumentar cada vez mais sua aplicação em sistemas de elevada complexidade matemática, cabendo aos modelos clássicos as áreas restantes.

Outro motivo que leva à escolha de controladores difusos é a dificuldade encontrada na formulação matemática da solução, como apresentado aqui. Neste trabalho, onde o objetivo é fazer com que o robô possa desviar de obstáculos aleatoriamente dispersos no ambiente, a discretização das ações e reações através da matemática se torna extremamente complexa, por não dizer praticamente inviável.

Neste projeto utilizamos um controlador difuso no controle do robô, por entendermos ser a melhor ferramenta para possibilitar o seu deslocamento em um ambiente totalmente aleatório, carregado de imprecisões e incertezas.

4.7 Conclusão

Uma vez definidas as ferramentas teóricas utilizadas na concepção do protótipo, o próximo passo será a busca de soluções e a determinação do produto final, ou seja, do robô RAU.

O capítulo seguinte retrata os aspectos puramente mecânicos envolvidos no projeto, ou seja, seu corpo, seu sistema de locomoção e direção, além de outros elementos eletrônicos que darão suporte a telepresença como o sistema de força, o de geração e posicionamento de imagem, além do módulo de comunicação com o microcomputador.

Capítulo 5 – O Hardware desenvolvido

5.1 Introdução

Quando se fala em construir um robô, a primeira impressão que fica é de um sistema cheio de mecanismos pneumáticos, com uma enorme quantidade de controles eletro-eletrônicos. Todavia, no projeto aqui apresentado, esta suposta complexidade fica bastante reduzida, principalmente pelo propósito definido para o protótipo.

Deve-se primeiramente lembrar que a função básica do sistema é retornar informações do meio onde se encontra para um usuário remoto, evitando que este coloque em risco o sistema como um todo.

De forma esquemática o hardware do sistema é composto de uma unidade de ação remota e uma unidade de processamento de informações, conforme mostrado na figura 5.1.

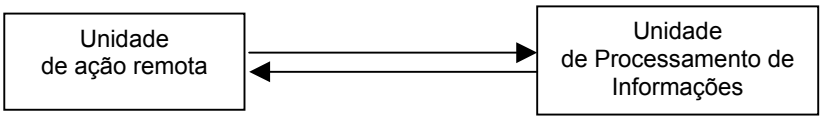


Fig. 5.1. – Esquema do hardware empregado

A troca de informações entre estas duas unidades deve priorizar a capacidade de mobilidade desejada ao sistema, optando-se preferencialmente por ondas de rádio como meio de comunicação de dados, evitando assim o uso de fios.

Baseando-se na definição topológica de robôs móveis dada por McComb (1987), dividiu-se o hardware em cinco elementos, conferindo uma identidade modular ao projeto. São eles:

i. corpo;
ii. sistema de locomoção;
iii. sensores;
iv. equipamentos de saída e
v. sistema de força.

Cada elemento do sistema é responsável por uma tarefa específica, podendo ser independente e exclusivamente alterado, substituído ou mesmo retirado, sem

interferência nos demais, capacitando o protótipo para trabalhos futuros. Este aspecto modular confere ao protótipo uma vantagem a mais, pois facilita sobremaneira o seu desenvolvimento.
Passemos agora à definição de cada um dos elementos constituintes do projeto, baseado na topologia de McComb.

5.2 O corpo

Para McComb (1987), o corpo *"..é uma superestrutura que previne que suas partes eletrônicas ou eletro-mecânicas caiam"*, ou seja, é um componente estritamente mecânico que dá forma, tamanho e estabilidade mecânica a todo o robô.
Na sua definição, pensou-se nas premissas básicas do projeto relacionadas no capítulo 3 deste trabalho, como: utilizar materiais que não sofram grandes alterações físico-químicas; priorizar dimensões reduzidas; optar por um modelo leve; dotar a unidade remota de equipamentos e comandos de fácil aquisição no mercado e projetar a unidade remota para resistir a impactos durante a sua operação.
Estes requisitos, somados à limitação de trabalhar em superfície plana, livre de desníveis, balizaram a investigação das possíveis variáveis envolvidas na confecção do corpo do robô.

5.2.1 O material empregado

A definição do material empregado passou por três opções existentes e utilizadas em robôs voltados à área acadêmica no mundo todo. O alumínio, a madeira e os plásticos.
Vale ressaltar que McKerrow (1991), também lista as ligas de aço como material estrutural utilizado em robôs, no entanto, devido ao seu elevado peso específico, estas ligas não se enquadram no rol de materiais avaliados para este projeto.
O alumínio é descrito por McComb (1987) como "*...o melhor material para construir robôs, porque..."* comparativamente ao plástico e à madeira *"...oferece uma resistência extra*", além de contar com fatores importantes como a leveza e facilidade

de manuseio. O autor, no entanto, ainda o classifica de um pouco mais caro que os outros materiais anteriormente listados.

A madeira é indicada para *hobbystas* por se tratar de um material barato e fácil de trabalhar, pois pode assumir qualquer forma desejada (MCCOMB, 1987). Todavia, conta com um comportamento estrutural que não permite muita elasticidade, além do elevado peso específico – comparativamente aos outros dois materiais – que resulta aumento do peso final do robô. Por fim, a madeira é um material anisotrópico, característica esta dada pela sua estrutura fibrilar (VLACK, 1988), e sensível à ação do tempo.

Os plásticos, por sua vez, são mais baratos do que a madeira ou qualquer metal, são fortes, têm boa relação peso/resistência e em alguns casos garantem uma elevada deformação elástica (MCCOMB, 1987). Necessitam, porém, ferramentas especiais e uma certa prática para o seu manuseio, principalmente se comparados ao alumínio, além de não aceitarem muitas vezes reparos ou emendas e, ainda, não serem encontrados facilmente no mercado.

O autor ainda relaciona como materiais elegíveis para a fabricação de um robô o estanho e o ferro. Suas vantagens residem no preço e na facilidade de aquisição no mercado, aliados a maior resistência mecânica que o alumínio. No entanto, comparado a este último, são por volta de duas vezes mais pesados (MCCOMB, 1987).

Analisadas cada uma das opções, elegeu-se o alumínio, não só pelas características já mencionadas, mas também pela facilidade de sua aquisição no mercado de Florianópolis.

5.2.2 A definição da geometria

Para a definição da geometria do robô foi adotado, como ponto de partida, o sistema utilizado em sua locomoção – adiante descrito – seguindo os dois principais modelos apresentadas por McKerrow (1991), a saber: os robôs com rodas e os com pernas. A opção pelo primeiro, conforme descrito no capítulo 4, deve-se à sua simplicidade cinemática e dinâmica, principalmente ao analisar-se a manobrabilidade do robô, uma

vez que as pernas não fornecem a correta estabilidade a par de exigirem um complexo controle ao executar um curva (MCCOMB, 1987).

Pelo que mostra McComb (1987), em função da geometria utilizada para alguns casos, a alteração de um sistema de locomoção para outro não requer muitas modificações, entretanto, para determinados formatos, a adoção de um modelo inviabiliza a conversão para outro.

Pensando nisto, utilizou-se uma configuração de chassi com longarinas, aproveitando-as como parte de uma estrutura autoportante, onde é montada a base que serve de suporte aos componentes eletro-eletrônicos da unidade.

Desta forma, foram utilizadas duas longarinas principais em perfil "L", apoiadas de forma longitudinal, obtendo-se um chassi com perfil em "U", conforme a vista frontal apresentada na figura 5.2.

Fig. 5.2 – Vista frontal do chassi do robô

Como neste ponto o sistema de locomoção ainda não havia sido definido em seus pormenores, a geometria apresentada tem caráter meramente teórico, pois carece de definições do tamanho das rodas e motores envolvidos.

Por sobre as longarinas será apoiada uma chapa de alumínio que servirá de base para a fixação dos componentes do robô, além de ser o elo de ligação entre os dois perfis em "L".

5.3 O sistema de locomoção

Por se tratar de um modelo com esterçamento por escorregamento, a definição dos sistemas de locomoção e de direção se confundem, uma vez que ambos contém os mesmos componentes.

A primeira opção a ser estudada foi a da utilização de esteiras, similar aos projetos SCARAB II e IIA da *R.O.V. Technologies, Inc.*[1], ao PEBBLES do *Artificial Intelligence Laboratory – MIT*[2] e ao ANDROS V, da *University of Michigan* (BORENSTEIN, EVERETT & FENG, 1996). Conforme McComb (1987), este tipo de sistema de locomoção é mais adequado quando se deseja utilizar o robô em locais onde há presença de irregularidades, obtendo-se elevada tração com a utilização do correto par de material esteira/piso.

Todavia, esta configuração foi abandonada pela dificuldade de se conseguir uma esteira que se adaptasse perfeitamente ao protótipo. Após contatar várias lojas de ferragens, autopeças e equipamentos, não se encontrou nenhuma correia que pudesse ser utilizada como esteira, de tal sorte que o sistema de locomoção teve que ser repensado.

A solução mais próxima, mantendo a configuração de esterçamento por escorregamento, seria a utilização de 6 rodas – como utilizado no robô *Terragator* da *Carnegie-Mellon University* e o *Prowler* da *RDS* (MCKERROW, 1991) – com os motores atuando ou em todas as rodas simultaneamente, ou apenas nas rodas centrais. O fato de existir 6 rodas tracionando eleva a complexidade do protótipo, pois é necessário um sistema de transmissão que distribua o movimento para todas as rodas ao mesmo tempo. Pode-se no entanto adotar mais motores, solução esta completamente descartada por implicar no perfeito sincronismo de rotação de todo o conjunto, o que acarretaria não só maior complexidade, como também maior consumo de energia pela unidade. Assim a opção de ter apenas as rodas centrais acionadas foi a escolhida.

---

[1]Maiores informações no site: http://www.rovtech.com/index.html
[2]Maiores informações no site: http://www.ai.mit.edu/projects/mobile-robots

Como elementos de propulsão foram utilizados dois motores de passo de 1,8 graus/passo, conectados diretamente as rodas centrais através de buchas, conforme figura 5.3, restando as demais com movimento livre, fixadas por parafusos ao chassi.

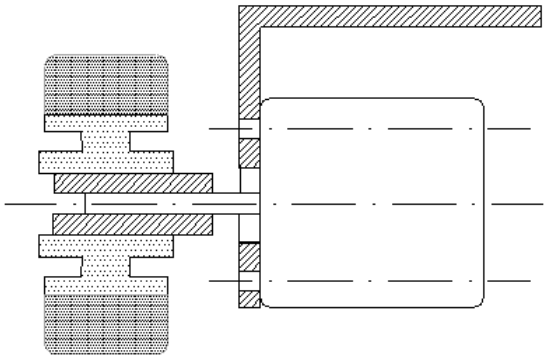


Fig. 5.3 – Montagem da roda motriz

Definida a geometria do corpo do RAU, foi criada uma maqueta eletrônica em AutoCAD14 para se visualizar o modelo final, conforme mostrado pela figura 5.4.

Fig. 5.4 – Desenho final do corpo do RAU

5.4 Os sensores

A utilização de sensores tem elevada importância neste projeto, por serem eles os responsáveis pela validação dos comandos do usuário. Através dos sinais gerados o ambiente onde está inserido o robô é avaliado, passando estas informações para a central de controle. O sensoramento escolhido para o RAU foi o baseado em infravermelho que, comparado aos ultra-sônicos – também amplamente utilizados em

robótica –, apresenta algumas vantagens, como ser invisível ao olho humano e ter um custo praticamente irrisório (JONES & FLYNN, 1993). No entanto, sua precisão não é garantida (GIBILISCO, 1994), pois não fornece uma medida precisa de distância (ZIMMER, 1996). Também deve-se tomar cuidado com aplicações onde possa existir forte radiação infravermelha de baixo comprimento de onda, como no caso de lâmpadas fluorescentes (JONES & FLYNN, 1993).

Todavia, como desejamos trabalhar com um controlador difuso no comando do robô, não há necessidade de se preocupar com a precisão quantitativa dos sensores, mas com a qualitativa, ou seja, a presença ou não de um obstáculo a sua frente. Isto fica reforçado por (BOTELHO,1996; MONDADA, FRANZI & IENNE, 1993) que utilizaram com sucesso sensores infravermelhos em seus projetos.

Através da utilização de sensores os robôs móveis podem realizar três tarefas básicas: evitar obstáculos durante o seu deslocamento; manter e atualizar dados referentes ao modelo global do ambiente onde está inserido e determinar a sua posição neste modelo (DAM, KRÖSE & GROEN, 1996).

Como neste projeto o robô será comandado por um operador, não se viu necessidade de utilizar os sensores como ferramentas de localização e atualização de uma representação do ambiente, por isso optou-se pela utilização dos sensores apenas no controle da trajetória, evitando o choque com obstáculos.

Definida a utilização, passou-se à determinação da posição dos sensores no robô. Erkamp (1996) mostra, de forma clara e objetiva, a aplicação de sensores na exploração de um ambiente desconhecido. A pesquisadora prova que são necessários no máximo três sensores para, sem conhecimento prévio, mapear um local em áreas adjacentes, chamadas de aspectos. Desta forma o robô consegue detectar em que aspecto está, sendo possível encadear uma série de comandos que o movimente a outro aspecto desejado. No entanto, a utilização de apenas três sensores se restringe a aplicações onde é possível uma exploração detalhada do local, guardando na memória do robô todos os dados coletados.

A partir do momento que se deseja um sistema que evite obstáculos fixos ou móveis, é necessário que este responda a informações provenientes de diversas direções, exigindo o emprego de maior número de sensores. O acréscimo no número de sensores

possibilita um aumento da precisão da estimação do local e distância de um objeto (TRIVEDI, ABIDI, EASON et al, 1990).

Desta forma, o robô foi então dividido em 8 direções principais, conforme figura 5.5, garantindo que pelo menos dois sensores estejam varrendo a mesma região, uma vez que seu foco de ação é de aproximadamente 45 graus.

Lembrando, no entanto, que o fator simplicidade no desenvolvimento do projeto tem um peso elevado na definição do protótipo, reduziu-se o número de direções envolvidas para 5, conforme limitações impostas no capítulo 3.

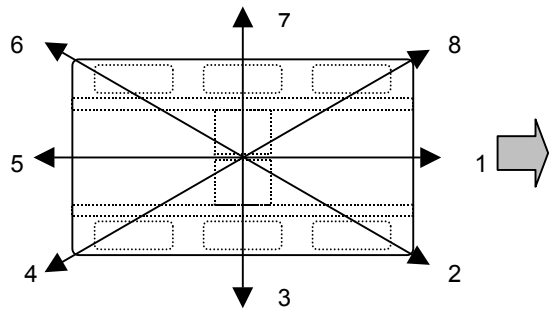


Fig. 5.5 – Direções principais dos sensores

Para não haver confusão com relação as direções manteve-se a mesma numeração dada no primeiro instante, permanecendo portanto as direções 1, 2, 3, 7 e 8.

Vale ressaltar que os sensores utilizados em cada uma das cinco direções serão responsáveis pela medição de proximidade de um obstáculo, enviando os dados para um controlador central – amplamente descrito no capítulo 6 – que avaliará o ambiente, tomando a decisão de que caminho seguir.

5.4.1 Definição do sensor infravermelho

Visando utilizar elementos já existentes no mercado, devidamente testados e aprovados, com o intuito de facilitar a montagem do robô, procurou-se por *kits* disponíveis, sendo o primeiro pesquisado o *Infrared Proximity Detector Kit – IRPD*, da *Linxmotion*[3] (figura 5.6).

[3] Informações podem ser obtidas no endereço: http://www.lynxmotion.com/

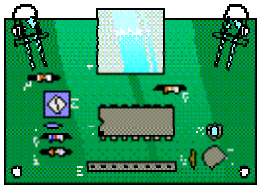


Fig. 5.6 – Ilustração do *kit* IRPD

Foi feita a importação de um *kit* IRPD para os devidos testes, uma vez que o catálogo do fornecedor não mostrava de forma clara seu funcionamento. Havia apenas a garantia de que o sinal era modulado, assegurando elevada robustez, tornando-o praticamente imune a ruídos externos.

A montagem do *kit* provou que, apesar de contar com dois emissores capazes de varrer um cone de mais de 90 graus, o sinal de saída fornecido era no padrão TTL, ou seja, assumia dois valores fixos – 0 ou 5 volts – inviabilizando o projeto por impossibilitar a medição desta proximidade de forma gradativa, necessária para servir de valor de entrada para o controlador difuso, cérebro do robô. Diante deste fato, o *kit* teve que ser abandonado e partiu-se para a busca de uma nova solução. Dando continuidade à busca, não se encontrou no mercado qualquer *kit* que relacionasse a distância do objeto com o sinal de saída conforme ilustrado pelas figuras 5.7 (GIBILISCO, 1994), não restando outra saída que não construir um *kit* próprio para o robô RAU.

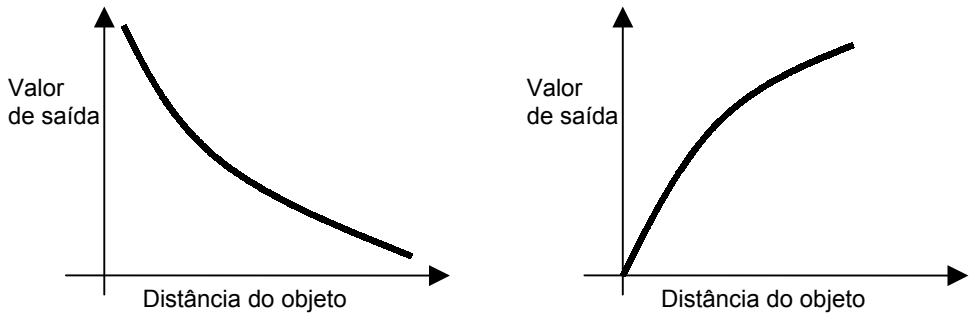


Fig. 5.7 – Funções de saída típicas de sensores de proximidade (fonte: GIBILISCO, 1996)

Foi então procurada a empresa Tirloni & Cia. Ltda. que desenvolveu os sensores de forma que o sinal de saída variasse de 0 a 9 volts, de forma gradativa e proporcional à distância entre robô e obstáculo.

Na confecção de cada sensor foram utilizados transistores, resistores, capacitores eletrolíticos e cerâmicos, diodos e o led infravermelho, não tendo sido fornecidos pela empresa os diagramas do circuito desenvolvido.

Todos os testes e calibrações necessários para o perfeito funcionamento dos *kits* serão abordados no capítulo 7.

5.5 Equipamentos de saída

Na definição de McComb (1987), o robô necessita utilizar equipamentos de saída para se comunicar com o mundo a sua volta. No caso do robô RAU aqui apresentado, o objetivo principal é permitir que uma pessoa, sentada à frente de um computador pessoal, com informações suficientes sobre onde e como o robô se encontra, possa comandá-lo a distância, sem fios.

Para melhor adaptar a definição de McComb ao projeto foi necessário realizar uma pequena alteração, permitindo que este módulo realize o envio de sinais do robô para o computador e vice-versa, passando então a equipamentos de saída/entrada como foi mostrado na figura 5.1.

Assim tem-se esquematicamente uma transmissão de dados conforme mostrado na figura 5.8. O computador envia dois tipos de sinal: de posicionamento da câmera e de direção que o robô deve seguir. O RAU por sua vez envia o sinal de vídeo para o computador de forma a possibilitar a teleoperação.

O transmissor do sinal composto de vídeo foi o primeiro a ser desenvolvido, utilizando um sistema de geração de imagem – definido no decorrer deste capítulo – e um emissor de sinal desenvolvido pela Tirloni & Cia. Ltda. com alcance de aproximadamente 50 metros. O sinal de vídeo composto é enviado por radiofreqüência, modulado para o canal 10 de qualquer televisor.

Fig. 5.8 – Esquema de transmissão de dados entre o Computador e o RAU

No computador foi instalada uma placa de captura de vídeo CIMAACT, modelo LGV – 5446TV para permitir a recepção e a projeção do canal de imagem na tela do micro. A escolha desta placa deve-se em grande parte ao software que a acompanha, responsável pelo ajuste fino da imagem, além, é claro, do custo inferior ao de outras placas pesquisadas no mercado.

Já para o envio de dados do computador para o RAU, utilizou-se a porta serial do micro. A comunicação estabelecida é de 8 bits, com 1 stop bit, sem paridade e com *baud rate* de 110 bps. A baixa velocidade é suficiente para o perfeito sincronismo entre robô e microcomputador. Além disto, o decodificador instalado no RAU passa a trabalhar de forma mais segura.

Foi conectada à porta serial do micro uma interface – também desenvolvida pela Tirloni & Cia. Ltda. – que envia, através de radiofreqüência o sinal para um decodificador, colocado na placa do circuito eletrônico do RAU. No RAU, o microcontrolador adotado – conforme será mostrado no capítulo 6 – recebe os dados através de três pinos de entrada de sinal digital, o 26, 27 e 28, formatado de acordo com o mostrado na tabela 5.1.

Posição do Joystick	Sinal Binário (port. 26,27 e 28)
Neutro	000
Para frente	111
Para direita	100
Para esquerda	001
Para trás	001
Para frente com o botão pressionado	110
Para trás com o botão pressionado	011
Para direita com o botão pressionado	101
Para esquerda com o botão pressionado	010

Tabela 5.1 – Valores de entrada do joystick no microcontrolador

Uma vez que a Tirloni & Cia. Ltda. já tinha o projeto das interfaces que se adaptavam ao protótipo, optou-se pela utilização de dois canais de comunicação *simplex*, ao invés de um *full duplex* ou *half duplex*.
Outro fator que levou à solução através da comunicação *simplex* foi o da validação dos comandos por parte do RAU, através de sua central de controle, pois o robô sempre testará qualquer ação solicitada pelo usuário.

5.6 Sistema de força

Para alimentar os equipamentos de saída, sensores e o sistema de locomoção, além da própria central de controle – descrita no próximo capítulo – fez-se necessária a adoção de baterias de Níquel-Cádmio, recarregáveis, com autonomia de 2 horas.
As baterias são montadas em um compartimento na parte inferior do RAU e contam com sistema de recarga, através de plug conectado à rede pública de energia elétrica. O conjunto de baterias fornece um diferencial de tensão de 12 volts com uma corrente máxima de 1.200 mili-ampéres.
Como existem diferentes tensões no robô, por exemplo, 5 volts para alimentação do microcontrolador, 12 volts para os motores e 9 volts para os sensores, vários reguladores de tensão foram utilizados.

5.7 Outros elementos do hardware

Além da topologia apresentada por McComb, o RAU tem algumas particularidades que tornam necessária a inclusão de mais alguns elementos em sua constituição. São eles: o sistema de geração de imagem; o sistema de posicionamento da imagem e a central de controle. Este último será abordado em um capítulo exclusivo por ser o mais complexo, o mais importante de todo o trabalho e o mais intimamente ligado ao software de controle do protótipo, baseado em técnicas de inteligência artificial.

5.7.1. Sistema de geração de imagem

Para gerar a imagem transmitida pelo robô RAU, adquiriu-se no mercado uma mini-câmera filmadora Kodo, modelo REV(S) que gera imagem em preto e branco, com resolução de 380 linhas e opera com iluminação de até 0.4 lux. Não se utilizou uma câmera que fornecesse imagem colorida principalmente devido ao preço, praticamente o dobro da que foi utilizada.

O sinal captado pela câmera é passado diretamente para os equipamentos de saída, responsáveis por enviá-lo até o microcomputador. A definição da imagem é relativamente boa, além de permitir a adoção futura de um módulo para reconhecimento de imagens.

5.7.2 Sistema de posicionamento da imagem

O sistema de posicionamento da imagem permite ao usuário controlar o azimute e a inclinação da câmera, através do *joystick* instalado no computador. Para isto o sistema tem uma geometria um pouco complexa, conforme a figura 5.9, e conta com a adoção de dois motores de passo, monitorizados - como no sistema de locomoção – pela central de controle.

Os motores utilizados são da marca Matsushita, modelo MSDA020L de 12 volts. Estes motores foram aproveitados do sistema de posicionamento do cabeçote de leitura de dois drivers de 5 ¼", visando, mais uma vez, reduzir custos. O valor de deslocamento dos motores, medido em graus por pulso, não era fornecido em sua etiqueta, sendo necessário um ensaio, anterior a montagem, para que se pudesse avaliá-lo.

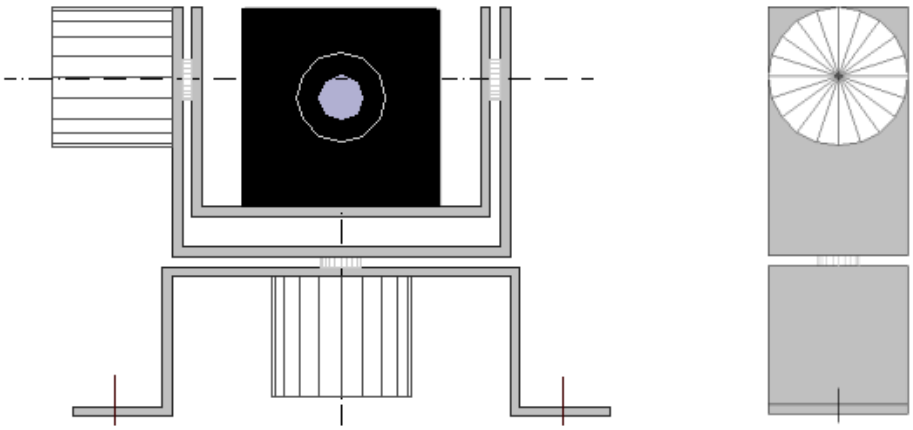


Fig. 5.9 - Vistas do sistema de posicionamento da imagem

5.8 Conclusão

Com este capítulo fica concluído o desenvolvimento da parte ordinária do robô. Agora, o que resta é a definição do cérebro do sistema, baseado em lógica difusa, capaz de interagir com o usuário.

Isto será objeto do capítulo 6, que tratará do desenvolvimento do hardware de controle, ou seja, do microcontrolador e de todos os componentes necessários para o seu interfaceamento com o resto do robô, e do software desenvolvido especialmente para esta aplicação e plataforma.

Toda a parte de ensaios e testes realizados, visando o perfeito e harmonioso trabalho entre os diversos componentes, pode ser acompanhado no capítulo 7 desta dissertação.

Capítulo 6 – O cérebro do sistema

6.1 Introdução

A central de controle – que compreende o software e o hardware de controle do robô – e o software de comando, por se tratarem da mais importante parte de todo projeto são aqui tratados mais detalhadamente.

Primeiramente será abordada a definição de Brooks adotada neste projeto e a seguir o hardware utilizado no controle embarcado no robô, justificando a aplicação de microcontroladores como unidade de processamento de informações. Após, se define e demonstra o software desenvolvido especialmente para esta plataforma, o qual dá ao robô condições de reagir de forma inteligente a modificações no ambiente onde está inserido.

Por fim, o software de comando, responsável pela comunicação entre a Unidade de Processamento de Informações (o microcomputador) e a Unidade de Ação Remota (o robô) será mostrado.

6.2 O Robô em camadas de Brooks

Tradicionalmente os controladores aplicados a robôs móveis trabalhavam em uma configuração chamada de módulos funcionais, na qual existia um encadeamento serial de ações, como percepção, modelagem, planejamento, execução de tarefa e controle dos motores, no caminho que ligava os sensores aos atuadores (BROOKS, 1986).

Brooks alterou a arquitetura reinante de maneira que os módulos fossem dispostos paralelamente, passando a ser chamados de camadas, trabalhando isoladamente (SIMMONS, GOODWIN, ZITA et al, 1997). Neste modelo, as camadas operam de forma concorrente, de tal sorte que mais de uma ação seja executada simultaneamente. Esta disposição, permite que cada camada seja uma produtora de atividade, não interferindo nas ações das demais (LEITE, ALVES & RESENDE, 1993). Surgia então

o controlador de robô baseado em comportamento, a também chamada de arquitetura de assunção (BROOKS, 1991; LEITE, ALVES & RESENDE, 1993).

Para desenvolver a sua teoria Brooks utilizou o conjunto de argumentos a seguir descrito:

1) comportamentos complexos não precisam necessariamente ser produtos de um sistema de controle extremamente complexo;
2) as coisas devem ser simples;
3) nós queremos construir robôs baratos que possam vagar através de espaços não habitados, sem intervenção do homem;
4) o mundo humano é tridimensional;
5) sistema de coordenadas absolutas para um robô são fontes de grandes erros cumulativos;
6) os mundos onde os robôs farão seu trabalho não são construídos de exatos e simples poliedros;
7) dados de sonares, apesar de fáceis de coletar, não levam a descrições ricas do mundo;
8) em consideração a robustez, o robô deverá ser capaz de trabalhar quando um ou mais de seus sensores falharem ou começarem a dar informações erradas;
9) nós estamos interessados em construir seres artificiais.

Pode-se claramente perceber a preocupação que o pesquisador teve em direcionar seus trabalhos para uma solução simples, capaz de representar de forma completa uma pequena parte de todo um comportamento, dividindo o problema em subproblemas.

Desta argumentação, Brooks chegou ao conceito de níveis de competência e camadas de controle. Aos níveis o pesquisador atribuiu o conceito de *"...especificação informal de uma classe de comportamento desejado por um robô sobre todos os ambientes que ele encontrará."* (BROOKS, 1986)

Um exemplo de nível de competência seria o fato de alcançar uma meta, ou evitar obstáculos, ou até mesmo buscar uma fonte de recarga de energia, o que nos leva a concluir sobre o elevado grau cognitivo esperado do robô.

Para as camadas, Brooks define que:

> *"...a idéia chave de níveis de competência, é que podemos construir camadas de controle correspondentes a cada nível de competência e simplesmente adicionar uma nova camada a um conjunto existente para mover para o nível mais alto de competência total."* (BROOKS, 1986)

Uma vez definidas as camadas, Brooks utilizou dois aspectos básicos em seu projeto: as máquinas de estado finito e a comunicação, o primeiro voltado à parte interna da estrutura e o segundo responsável pela ligação da estrutura com o resto do controlador. As máquinas de estado finito são definidas como modelos matemáticos que respondem a entradas e saídas discretas, assumindo um número finito de estados, onde são conservadas apenas as informações do passado relevantes a ações a serem tomadas na próxima entrada (MENEZES, 1997). Cada camada de controle tem uma série de módulos, conectados entre si como na decomposição tradicional, ou seja, encadeados serialmente. Cada módulo destes é considerado uma máquina de estado finito.

Com relação a comunicação Brooks apresenta um módulo com as possíveis conexões, que são os ramos de comunicação de cada máquina de estado finito com outras, seja no seu nível de competência ou não (figura 6.1).

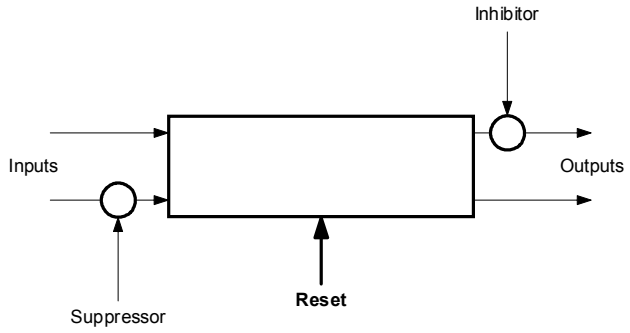


Fig. 6.1 – Representação da comunicação de um módulo (Brooks, 1986)

Existem sinais de entrada e de saída, sendo estes últimos, responsáveis por supressões – quando atuam nas entradas de outros módulos –, ou por inibições –quando estão conectados às ramificações de saída. Estas supressões e inibições podem atuar em uma mesma camada ou em camadas diferentes, dependendo do comportamento desejado.

As conexões de supressão e inibição só agem quando existe sinal trafegando por elas. Se existe sinal, o valor original de saída ou entrada é trocado pelo valor do ramo extra, proporcionando um comportamento diferente daquele fornecido aos sinais originais, como se fossem conectados módulos de prioridades diferentes (LEITE, ALVES & RESENDE, 1993).

Partindo-se para uma visão macroscópica do controlador em camadas de Brooks, podemos, através da figura 6.2, observar que dado um conjunto de sensores, que fornecem sinais para processamento do controlador, existe uma série de camadas de controle, que capturam os valores relevantes ao seu nível de competência e operam estes valores através de seus módulos, fornecendo dados de saída que serão enviados aos atuadores. Assim, de forma simples, como era buscado pelo pesquisador, chegou-se a um modelo de controle adotado por uma série de outros pesquisadores. (SIMMONS, GOODWIN, ZITA et al, 1997; ALVES, RESENDE, LEITE et al, 1993; SAFFIOTTI, 1997; PAN, PACK, KOSAKA et al, 1995; BOTELHO, 1996; SEQUEIRA & RIBEIRO, 1997).

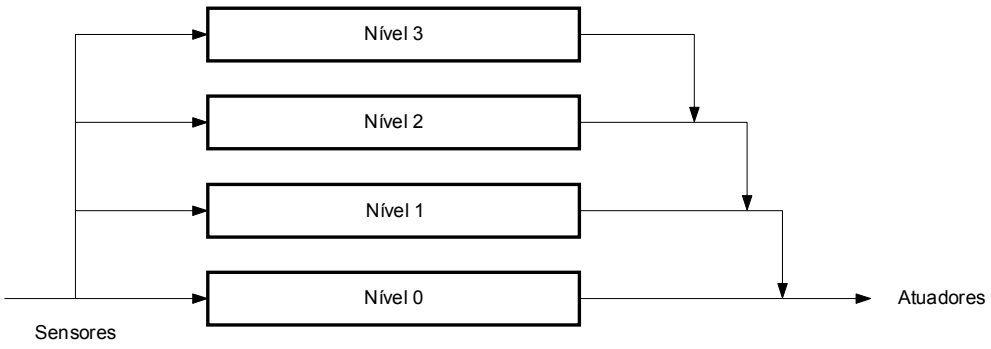


Fig. 6.2 – As camadas do controlador de Brooks

Como principal vantagem da adoção deste conceito de controle podemos citar a capacidade de tornar as camadas mais específicas possível (BROOKS, 1991), se traduzindo em níveis de competência com elevado grau de cognição, possibilitando ao robô a execução de tarefas mais complexas, de forma simultânea.

Outra grande vantagem é a capacidade incremental oferecida pelo modelo, que permite a adição de mais níveis e camadas à medida em que elas se tornarem necessárias, sem

interferência nas camadas anteriormente utilizadas (ALVES, RESENDE, LEITE et al, 1993).

Além disto, fica patente a capacidade de processamento paralelo, tornando o controlador apto a trabalhar com um maior número de informações simultaneamente.

6.2.1 A utilização das Camadas de Brooks no robô RAU

Por não se tratar de um robô totalmente autônomo, mas sim de um sistema de telepresença onde existe uma forte relação de dependência entre robô e usuário, viu-se a necessidade de uma adaptação do conceito de Brooks ao RAU.

Primeiramente, faz-se necessário lembrar o comportamento esperado do nosso robô, que pode ser definido como:

"O robô RAU deve alcançar objetivos traçados pelo usuário – responsável pelo seu comando –, evitando que as ordens recebidas resultem em movimentos que coloquem em risco a integridade do sistema, possuindo então certa inteligência, capaz de desviar de obstáculos que possam aparecer no seu caminho."

Da definição do problema dado, pode-se chegar a dois comportamentos esperados. O primeiro é *seguir um comando dado pelo usuário*, sendo o robô neste caso, mero sistema reativo, similar aos robôs da primeira geração, chamados *Pick and Place* (ALVES, RESENDE, LEITE et al, 1993).

O segundo comportamento do robô RAU é a *capacidade de desviar de obstáculos* que possam estar em seu caminho enquanto este estiver realizando uma ação relativa ao primeiro comportamento. Podemos observar também que este comportamento deve inibir o primeiro no sentido de salvaguardar a integridade do sistema.

Assim, o sistema de telepresença aqui apresentado funciona baseado em duas camadas, a de comando do usuário e a de controle próprio do RAU (fig. 6.3), possibilitando que todos os comandos fornecidos pelo usuário sejam cotejados com os provenientes do controle do robô.

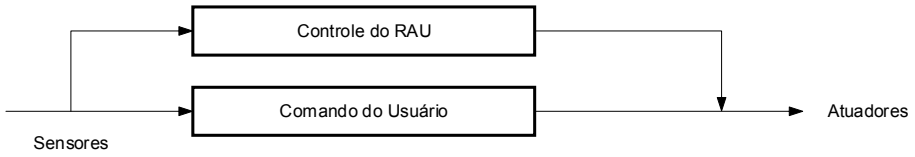


Fig. 6.3 – Modelo em camadas do sistema de telepresença

Percebe-se, de acordo com a figura 6.3, que a camada de comando do usuário pode ter sua saída inibida pela camada de controle do robô. Isto se deve em grande parte à característica de ação remota, onde a falta de informações pode levar a uma tomada de decisão errônea. Para minimizar esta deficiência o controle do robô deve ser capaz de – através de sensores – retirar informações do ambiente em que se encontra de tal forma que, aliadas aos dados enviados pelo usuário, possa estabelecer uma ação adequada.

Passa-se à analise dos tipos de sensores e atuadores exigidos pelo projeto, iniciando pelos utilizados no nível 0 de competência, o comando do usuário. Para a interface entre usuário e robô é necessário que o primeiro indique como deseja que o segundo se comporte, assim o *joystick* utilizado no microcomputador onde está instalado o software de comando é o nosso primeiro sensor de entrada do sistema. Também para o nível 1 – responsável pelo controle do robô – são necessários sensores que traduzam o ambiente para uma representação digital ou eletrônica. Como já foi descrito no capítulo 4, os sensores infravermelhos fornecerão estes dados ao controlador.

Como atuadores do sistema temos apenas os motores de passo que – conectados às rodas centrais do robô – são capazes de receber e interpretar comandos de ação de ambos os níveis (v. capítulo 4).

Borenstein (1993) apresenta um modelo bastante similar a este, no qual o operador fornece dados ao interpolador de trajetória, que traça informações com o controlador embarcado no veículo, que atuará na velocidade de cada um de seus motores.

Outro modelo que apresenta semelhança ao ora apresentado é o desenvolvido por Pan, Pack, Kosaka *et al* (1995), baseado em um controlador difuso que está conectado com outros seis módulos – entre os quais a supervisão humana - responsáveis pela navegação do robô.

6.3 Microcontroladores

Os microcontroladores são componentes eletrônicos que permitem o desenvolvimento de projetos compactos e sofisticados, semelhantes em desempenho aos dos microprocessadores (SILVA JÚNIOR, 1988), mas com custos infinitamente menores.

Existe na literatura alguma divergência sobre a diferença entre microcontroladores e microprocessadores. Alguns autores classificam o primeiro como um caso especial do segundo (JONES & FLYNN, 1993), enquanto outros classificam como dois componentes eletrônicos distintos, para usos específicos (SILVA JÚNIOR, 1988; SCHULTZ, 1993).

Mesmo com o advento no mercado de microprocessadores de 32 ou 64 bits, existem aplicações mais simples, específicas e que necessitam de pequenos códigos para executá-las. Para estas aplicações, microcontroladores de 16 ou 8 bits representam a melhor relação custo/benefício de desenvolvimento (SCHULTZ, 1993). Duarte (1998) salienta que o *"custo é um importante fator a considerar quando se adiciona inteligência a um aparelho."* Portanto, o microcontrolador é – comparativamente a outros circuitos integrados, como os de aplicação específica (ASIC) e os microprocessadores listados pelo autor – a melhor solução.

Mas o que torna um microcontrolador um componente capacitado para estas aplicações altamente especializadas? O primeiro ponto significante é a incorporação de uma série de blocos dentro de um mesmo encapsulamento, dentre eles:

- linha de comunicação serial, que capacita o microcontrolador a trocar informações com outros microcontroladores ou até mesmo microprocessadores;
- portas de entrada/saída digitais, que permitem a comunicação do microcontrolador com outros componentes digitais;
- temporizador/contador, responsável por ações cíclicas, baseadas no tempo;
- memória de acesso aleatório (RAM), onde são executados os programas e armazenadas as variáveis em tempo de execução e

- memória *flash* programável, que simplifica a programação e reprogramação do *chip*, tornando desnecessário o uso de memórias programáveis apagáveis de somente-leitura (EPROM).

Jones & Flynn (1993) classificam o surgimento dos microcontroladores como um marco no desenvolvimento de robôs, e comentam:

> *"Antes do advento do microcontrolador, para alcançar os requisitos de sensoramento e de ação de atuadores de um robô, era necessário construir um sistema que consistia de um grande número de placas de circuitos impressos conectados entre si..."*
>
> *"Hoje, o tamanho, complexidade, consumo de energia, bem como o custo de um sistema, pode ser reduzido pelo uso de um microcontrolador para executar todas as tarefas de processamento em um só chip."*

Assim, o uso de um microcontrolador se apresenta ideal, pela redução de tamanho e custos aliada à menor complexidade, fatores imperativos no desenvolvimento deste projeto.

Além disto a constante evolução dos semicondutores em geral vem fazendo com que os microcontroladores e os microprocessadores diminuam seu consumo de energia, aspecto relevante neste projeto, principalmente pelo caráter autônomo do robô, isto é, sua total desvinculação, através de fios, do ambiente onde se situa.

6.3.1 O 8051 e suas aplicações

O subsistema de comando direcional é baseado em um microcontrolador ATMEL 89C52, da família de microcontroladores Intel 8051. Este componente tem 4 portas de comunicação de 8 bits que permitem variadas conexões. A família 8051 foi escolhida pela facilidade de aquisição, pela capacidade e simplicidade de programação conferida a memória flash, pelo seu tamanho reduzido e pela possibilidade de se utilizar comandos em linguagens de alto nível, como a linguagem C.

Outro fator que também fez do 8051 o componente ideal para este projeto foi a facilidade com que se encontrou livros sobre sua programação e funcionamento (SILVA JÚNIOR**,** 1988; SHULTZ, 1993; SILVA JÚNIOR, 1994; PREDKO, 1999).

Os *chips* da família 8051 têm, em geral, uma CPU com processador booleano, 5 a 6 interrupções, 2 a 3 *timers* de 16 bits, uma porta serial *full-duplex* programável, 32 linhas digitais de entrada/saída, memória RAM e ROM, além de memória EPROM e flash, em alguns casos (SILVA JÚNIOR, 1994).

Colocados no mercado pela Intel em 1980, os microcontroladores 8051 são hoje utilizados em diversas aplicações de controles inteligentes, de sofisticadas cafeteiras ao mais moderno sistema de freios veicular ABS. Atualmente são 98 as versões disponíveis, produzidas por 8 diferentes fabricantes (DUARTE, 1998).

A enorme produção de conhecimento sobre o funcionamento da família 8051, um dos mais antigos controladores comercializados, faz crer que ainda há aplicações a serem desenvolvidas.

6.4 O controlador difuso do RAU

O problema a ser solucionado é típico de navegação em robôs autônomos onde se deseja, por meio de um sensoramento adequado, percorrer um espaço desconhecido, evitando o choque com algum obstáculo. Este controlador deve ter três características fundamentais: ser baseado em sensores, que alimentarão o sistema de controle com os dados para a escolha da melhor trajetória; ser capaz de desviar de obstáculos, de tal sorte que o robô nunca se exponha a riscos gerados por erros do operador e por fim, não necessitar de conhecimento prévio do meio, uma vez que o controlador receberá toda informação necessária dos sensores.

O controlador estudado neste capítulo é do tipo reativo – conforme definido por Saffiotti, Ruspini e Konolige (1999) –, ou seja, um processo de inferência difusa – que resultará em uma reação do robô – é disparado em função de informações provenientes do meio onde o ele estiver inserido.

Basicamente, o software de controle faz o levantamento do nível de proximidade dos obstáculos ao RAU e escolhe a melhor trajetória possível, de forma a manter a integridade do sistema.

Antes de definirmos os procedimentos e a metodologia utilizados no desenvolvimento do controlador difuso do robô RAU, é interessante observar primeiramente, algumas observações de Timothy Ross (1995) sobre o projeto de um controlador difuso:

1. o processo deve ser de fácil observação e controle, apresentando estados e variáveis de entrada e saída mensuráveis e que permitam sua avaliação;
2. deve existir conhecimento armazenado sob a forma de regras de produção, ou de um conjunto de dados de entrada e saída ou, ainda, de um modelo analítico que possa ser fusificado;
3. deve existir uma solução;
4. deve-se buscar uma boa solução, não importando em ser a ótima;
5. o controlador deve ser projetado baseando-se no conhecimento disponível a respeito do problema;
6. trabalhar tendo presente que os problemas de estabilidade e otimização em controladores difusos ainda não estão completamente explorados.

Todos os seis pontos levantados por Ross devem ser observados quando do início do projeto do controlador e podemos concatená-los em apenas três princípios que utilizaremos em nosso projeto:

1. definir o controlador sobre uma base de conhecimento segura, confiável, disponível e a mais concisa possível;
2. buscar uma boa solução existente;
3. definir os conjuntos de variáveis do problema de forma que se possa facilmente avaliar seu desempenho, ao mesmo tempo em que se evite problemas de estabilidade e otimização.

Cabe uma análise mais profunda – a partir do problema abordado – dos citados princípios de Ross.

Quando se deseja que o robô vá de um ponto para outro em um dado espaço existente, delimitado pelo ambiente onde está inserido, observa-se que existem estados bem

definidos entre esses pontos e uma função que recebe como entrada as variáveis lidas do meio, produzindo como saída um valor de direção.

Assim, a função transformação que leva o robô de um ponto ***A*** para um ponto ***B*** tem uma formulação específica, baseada nas regras difusas, para que possa produzir efeitos partindo-se de valores de entrada. Este espaço de busca deve obrigatoriamente ter uma solução, que não precisa ser a ótima.

A solução encontrada deve ser mensurável e comparável a um dado padrão, de forma que possamos medir a precisão do controlador, o que é possível e muito útil para ajudar a definir a melhor configuração.

Aplicando estes conceitos aos robôs móveis encontramos três premissas que, segundo Saffiotti, Ruspini e Konolige (1999), devem ser observadas para um controlador de um robô móvel operando no mundo real:

1. trabalhar sob condições de imprecisão e incerteza;
2. alcançar metas explícitas, de forma automática ou por meio de comando do usuário; e por fim,
3. gerenciar múltiplas informações simultaneamente.

Os três pontos indicam que o controlador difuso é a melhor opção para esta aplicação, principalmente por trabalhar com imprecisões e incertezas e por ter um processamento paralelo, já que as regras do sistema são avaliadas em conjunto.

A metodologia de projeto do sistema de controle difuso adotada neste trabalho é a apresentada por Earl Cox (1994). Nela o autor apresenta 5 pontos a serem seguidos no projeto de um controlador.

1. selecionar as variáveis de performance, ou seja, as de controle e solução;
2. definir as superfícies de controle através dos conjuntos difusos;
3. construir relações entre os espaços de entrada e saída, adotando regras de produção;
4. rodar um modelo simulado do controlador;
5. interpretar as saídas, defusificando os conjuntos para obter os valores esperados.

Tal metodologia é muito parecida com a apresentada por Klir e Yuan (1995), que se baseia também em cinco passos:

1. identificar as variáveis de entrada e saída do problema;

2. definir os conjuntos difusos relevantes para a busca da solução;
3. criar regras de inferência difusas;
4. utilizar a máquina de inferência para relacionar os conjuntos de entrada e saída do problema;
5. defusificar as conclusões obtidas, obtendo um valor numérico.

Por parecer mais completa, uma vez que se preocupa com a integridade do controlador, sugerindo rodar um modelo simulado deste, optou-se pela metodologia de Earl Cox. Entendendo que é preciso escolher, após sucessivos testes comparativos um método de defusificação que mais se aproxime do resultado esperado, foram utilizados os pontos 4 e 5 simultaneamente.

Uma vez definida a metodologia empregada, serão descritas, a seguir, cada uma das etapas vencidas no projeto.

6.4.1 Definição das variáveis

O primeiro ponto a ser abordado no projeto do controlador difuso empregado no robô RAU, conforme a metodologia de Cox, será a definição das variáveis relevantes à solução de parte do problema apresentado na seção 6.2.1, ou seja: **a capacidade de desviar de obstáculos**. Vale lembrar que ao controlador difuso cabe apenas o nível de competência 1 do conceito de Brooks.

Para melhor entender como o controlador realmente funcionará é necessário voltar às definições de hardware apresentadas no capítulo anterior. A seção 4.3, que trata dos sensores, primeiramente definiu 8 direções principais de varredura que, em favor da simplificação, foram reduzidas para 5.

A cada direção foi associado um sensor que envia o valor de leitura como dado de entrada para o controlador do robô. Assim, o controlador receberá uma variável para cada sensor utilizado –aqui chamada de proximidade –, e que está relacionada a distância entre o obstáculo e o robô.

Vale ressaltar que as variáveis para cada direção são completamente independentes umas das outras.

Diante dos valores de entrada, ou seja, dos sinais de proximidade de cada uma das cinco direções fornecidos pelos respectivos sensores, o controlador deve realizar operações que possam definir qual direção seguir, que será, por sua vez, a única variável de saída do controlador.

Às variáveis lingüísticas envolvidas no sistema serão associados valores lingüísticos capazes de representar as condições e ações nas regras de produção, o que será visto na próxima seção.

Cox (1994) comenta que *"variáveis similares podem ser consolidadas..."* enquanto que macro-variáveis *"...podem ser decompostas em variáveis individuais."* (COX, 1994). Assim, é necessário observar quais variáveis realmente afetam o funcionamento do controlador, eliminando todas as demais.

Atendendo recomendações de Cox segundo as quais as variáveis similares devem ter a mesma unidade de medida, o conjunto de variáveis de entrada, num total de cinco é:

- Proximidade na direção 1
- Proximidade na direção 2
- Proximidade na direção 3
- Proximidade na direção 7
- Proximidade na direção 8

Enquanto isso, existe apenas uma variável lingüística de saída, que é a direção a ser tomada.

Cabe observar aqui a semântica de variável lingüística utilizada por cada autor. Cox (1994), diferentemente do que é definido por Zadeh (1975) apud Driankov, Hellendoorn e Reinfrank (1996), Mamdani e Assilian (1975) e Klir e Yuan (1995), atribui o significado de variável lingüística a todo e qualquer conjunto difuso utilizado no sistema, o que pode causar dificuldade ao leitor, uma vez que não é possível ter uma variável com diferentes valores lingüísticos.

Para evitar prolongar a discussão sobre o sentido das variáveis lingüísticas, se adotará o conceito de Zadeh, que diz: *"por variável lingüística nós entendemos ser uma variável cujos valores são palavras ou sentenças em linguagem natural ou artificial"*, e exemplifica: *"...idade é uma variável lingüística se seus valores são lingüísticos ao invés de numéricos."* (ZADEH, 1975 apud DRIANKOV, HELLENDOORN E REINFRANK ,1996).

Desta forma, na próxima seção, serão discutidos os valores lingüísticos, também chamados de estados (KLIR & YUAN, 1995) de cada variável definida anteriormente.

6.4.2 Definição dos conjuntos difusos

Quando se trata com conjuntos difusos a questão mais debatida é o formato do conjunto, ou seja, qual tipo de função de pertinência utilizar (COX, 1994; ROSS, 1995). Klir e Yuan (1995) comentam em sua obra que muitas das aplicações não são excessivamente sensíveis a variações de forma, o que é reforçado por Viot (1993) que diz *"Quanto mais simples o formato da função de pertinência, mais fácil a construção, manutenção manipulação e execução desta função"*. O autor ainda completa indicando ser os de formato triangular e trapezoidal os mais populares, sem perda de eficiência.

Driankov, Hellendoorn e Reinfrank (1996) estabelecem que a escolha da função de pertinência deve se basear na eficiência computacional, no uso otimizado de memória e na análise da performance, sugerindo uma representação uniforme, ou seja, baseada em funções parametrizadas. Os autores referem-se a representação triangular, trapezoidal e na forma de sino como as mais indicadas.

Cox (1994) comenta também sobre o *overlap* entre os conjuntos difusos. Segundo ele, *"a experiência dita que o overlap para regiões difusas entre triângulos ou entre triângulos e trapezóides em média fica entre 25% a 50% da base do conjunto difuso."* Ele esclarece que o *overlap* vai depender também do *"...conceito modelado e do grau intrínseco de imprecisão..."*(COX, 1994) associado aos conjuntos difusos em questão.

Por ser problema em estudo de complexa formulação matemática, impossibilitando obter expressões que possam fornecer os comportamentos esperados pelo robô, partiu-se de um dos métodos possíveis para definir as funções de pertinência dos conjuntos difusos: a intuição (ROSS, 1995).

A intuição nada mais é do que a capacidade que nós humanos temos de discernir, ou seja, de criar distinção entre determinados conjuntos de informação elegendo padrões. Cotejados os vários exemplos ilustrativos encontrados nas obras citadas, passou-se à

definição dos valores lingüísticos associados às variáveis lingüísticas previamente determinadas para o controlador do robô RAU.

O controlador proposto tem uma particularidade que acaba simplificando a sua implementação. Conforme mostrado temos cinco variáveis lingüísticas de entrada para o controlador, iguais em sua origem mas diferentes em sua semântica, ou seja, variáveis do mesmo tipo, o que nos leva a admitir que possam ter o mesmo conjunto de estados ou valores lingüísticos. Assim sendo, às variáveis lingüísticas "proximidade", para cada uma das cinco direções, podem ser fornecidos os valores **longe** e **perto**.

Da mesma forma, optou-se por definir cinco valores lingüísticos diferentes para a variável direção a ser tomada: virar à esquerda muito, virar à esquerda pouco, em frente, virar à direita pouco e virar à direita muito.

Definidos os valores lingüísticos, voltou-se a atenção para a determinação do suporte de cada um dos conjuntos difusos. Para o caso das variáveis proximidade, que estão intimamente ligadas ao sinal oriundo dos sensores, o mais racional é ter como suporte os valores limite de saída dos sensores, ou seja, de 0 a 9 volts.

Já para a direção a ser tomada o raciocínio é um pouco diferente. Primeiramente se adotou a direção 1 como valor zero, convencionando-se valores positivos para direções à direita deste e negativos à esquerda, como pode ser visto na figura 6.4.

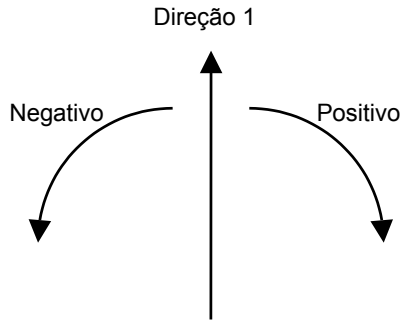


Fig. 6.4 – Convenção das direções de saída do controlador

Em seguida, limitou-se o valor máximo da direção em 90º para cada um dos lados, centrando-se o valor lingüístico em frente na direção zero, com um limite de 45º para a esquerda e direita, simetricamente, fornecendo um suporte igual a 90º. O *overlap* adotado para cada conjunto ficou em 50%.

Estabelecida a estrutura básica dos conjuntos envolvidos, o próximo passo é a definição de sua forma. Como já foi comentado, alguns autores relatam que muitos sistemas difusos são invariantes à forma dos conjuntos que os constituem (COX, 1994; KLIR & YUAN, 1995), ou seja, a forma pouco ou nada influencia nos valores de saída.

Pensando na suavidade de operação do sistema proposto, adotou-se, em um primeiro instante, uma função de pertinência sob a forma de sino para cada um dos conjuntos difusos anteriormente definidos, conforme mostrado na figura 6.5.

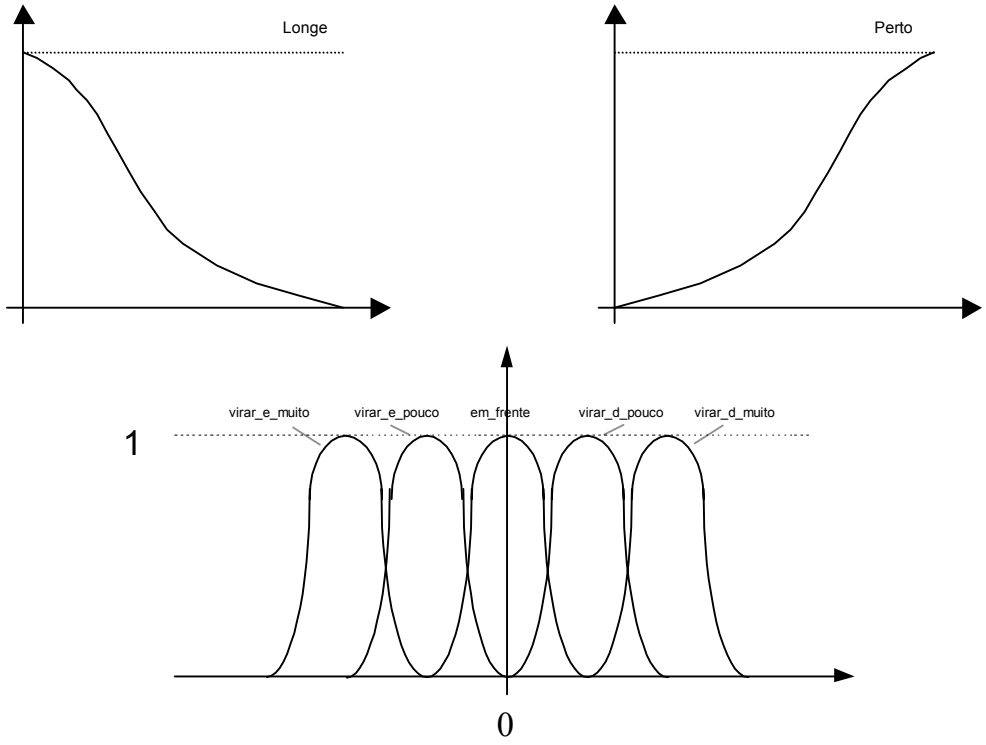


Fig. 6.5 – Conjuntos difusos utilizados

Na seqüência do trabalho, quando for tratado o refino do sistema (seção 6.4.4), será retomada a questão sobre a forma dos conjuntos difusos utilizados.

6.4.3 Definição do conjunto de regras

Uma vez determinados os conjuntos envolvidos no controlador, passou-se à definição das regras, remetendo-se o problema à forma como os humanos agem em tal situação, ou seja, evitando obstáculos quando estes se colocam em situação próxima e ignorando-os quando estão distantes.

As regras foram elaboradas levando em conta a direção por onde o obstáculo se aproxima, fazendo que o controlador defina a ação de contorno. Por exemplo, tomando a **direção 1** como 0º, se existe um obstáculo localizado a 25º à direita, ele será sentido por dois sensores, o 1 e o 2. Estes gerarão um grau de pertinência nos conjuntos **longe** e **perto** para as respectivas variáveis de proximidade, o que acarretará na ação de virar à esquerda. Deve-se observar que para o exemplo ora ilustrado, a meta a ser alcançada é dada em função do comando do usuário e, neste caso, há necessidade de que ele ordene que o robô siga em frente.

Assim, se algum objeto for detectado por mais de um sensor, a ação de virar para um lado ou para o outro deverá ser mais rápida do que no caso em que apenas um forneça um grau de pertinência significante para o controlador.

Começou-se então a definir regras possíveis de serem executadas, levando em conta o comportamento humano diante deste tipo de problema, chegando-se ao conjunto de 40 regras, conforme a tabela 6.1. Para tornar o conjunto de regras mais claro e inteligível, as variáveis lingüísticas proximidade serão chamadas proximidade_***X***, onde ***X*** representa a direção.

01	**Se** proximidade_1 é perto **e** proximidade_2 é perto, **então** direção a tomar é virar à esquerda muito
02	**Se** proximidade_1 é perto **e** proximidade_3 é perto, **então** direção a tomar é virar à esquerda muito
03	**Se** proximidade_1 é perto **e** proximidade_7 é longe, **então** direção a tomar é virar à esquerda muito
04	**Se** proximidade_1 é perto **e** proximidade_8 é longe, **então** direção a tomar é virar à esquerda muito
05	**Se** proximidade_2 é perto **e** proximidade_3 é perto, **então** direção a tomar é virar à esquerda muito
06	**Se** proximidade_2 é perto **e** proximidade_7 é longe, **então** direção a tomar é virar à esquerda muito
07	**Se** proximidade_1 é longe **e** proximidade_2 é perto, **então** direção a tomar é virar à esquerda pouco
08	**Se** proximidade_3 é perto **e** proximidade_7 é longe, **então** direção a tomar é virar à esquerda pouco
09	**Se** proximidade_2 é perto **e** proximidade_3 é longe, **então** direção a tomar é virar à esquerda pouco
10	**Se** proximidade_7 é longe **e** proximidade_8 é longe, **então** direção a tomar é virar à esquerda pouco

11	**Se** proximidade_7 é perto **e** proximidade_8 é longe, **então** direção a tomar é virar à esquerda pouco
12	**Se** proximidade_2 é perto **e** proximidade_8 é longe, **então** direção a tomar é virar à esquerda pouco
13	**Se** proximidade_2 é perto **e** proximidade_7 é perto, **então** direção a tomar é virar à esquerda pouco
14	**Se** proximidade_3 é perto **e** proximidade_8 é longe, **então** direção a tomar é virar à esquerda pouco
15	**Se** proximidade_1 é longe **e** proximidade_2 é longe, **então** direção a tomar é em frente
16	**Se** proximidade_1 é longe **e** proximidade_3 é longe, **então** direção a tomar é em frente
17	**Se** proximidade_1 é longe **e** proximidade_3 é perto, **então** direção a tomar é em frente
18	**Se** proximidade_1 é longe **e** proximidade_8 é longe, **então** direção a tomar é em frente
19	**Se** proximidade_1 é longe **e** proximidade_7 é longe, **então** direção a tomar é em frente
20	**Se** proximidade_1 é longe **e** proximidade_7 é perto, **então** direção a tomar é em frente
21	**Se** proximidade_3 é longe **e** proximidade_7 é longe, **então** direção a tomar é em frente
22	**Se** proximidade_3 é perto **e** proximidade_7 é perto, **então** direção a tomar é em frente
23	**Se** proximidade_2 é longe **e** proximidade_8 é longe, **então** direção a tomar é em frente
24	**Se** proximidade_2 é perto **e** proximidade_8 é perto, **então** direção a tomar é em frente
25	**Se** proximidade_2 é longe **e** proximidade_7 é longe, **então** direção a tomar é em frente
26	**Se** proximidade_3 é longe **e** proximidade_8 é longe, **então** direção a tomar é em frente
27	**Se** proximidade_1 é longe **e** proximidade_8 é perto, **então** direção a tomar é virar à direita pouco
28	**Se** proximidade_3 é longe **e** proximidade_7 é perto, **então** direção a tomar é virar à direita pouco
29	**Se** proximidade_2 é longe **e** proximidade_3 é longe, **então** direção a tomar é virar à direita pouco
30	**Se** proximidade_2 é longe **e** proximidade_3 é perto, **então** direção a tomar é virar à direita pouco
31	**Se** proximidade_7 é longe **e** proximidade_8 é perto, **então** direção a tomar é virar à direita pouco
32	**Se** proximidade_2 é longe **e** proximidade_8 é perto, **então** direção a tomar é virar à direita pouco
33	**Se** proximidade_2 é longe **e** proximidade_7 é perto, **então** direção a tomar é virar à direita pouco
34	**Se** proximidade_3 é perto **e** proximidade_8 é perto, **então** direção a tomar é virar à direita pouco
35	**Se** proximidade_1 é perto **e** proximidade_2 é longe, **então** direção a tomar é virar à direita muito
36	**Se** proximidade_1 é perto **e** proximidade_3 é longe, **então** direção a tomar é virar à direita muito
37	**Se** proximidade_1 é perto **e** proximidade_8 é perto, **então** direção a tomar é virar à direita muito
38	**Se** proximidade_1 é perto **e** proximidade_7 é perto, **então** direção a tomar é virar à direita muito
39	**Se** proximidade_7 é perto **e** proximidade_8 é perto, **então** direção a tomar é virar à direita muito
40	**Se** proximidade_3 é longe **e** proximidade_8 é perto, **então** direção a tomar é virar à direita muito

Tabela 6.1 – Primeiro conjunto de regras do controlador

Ao elencar critérios para se escrever o conjunto de regras para um sistema difuso, Cox (1994) sugere que as de mesma variável de saída sejam agrupadas. Assim, é possível representar todo o conjunto anterior reduzindo-o a apenas 5 regras, conforme a tabela 6.2.

01	**Se** proximidade_1 é perto **e** proximidade_2 é perto **e** proximidade_3 é perto **e** proximidade_7 é longe **e** proximidade_8 é longe, **então** direção a tomar é virar à esquerda muito
02	**Se** proximidade_1 é perto **e** proximidade_2 é perto **e** proximidade_3 é longe **e** proximidade_7 é longe **e** proximidade_8 é longe, **então** direção a tomar é virar à esquerda pouco
03	**Se** proximidade_1 é perto **e** proximidade_2 é longe **e** proximidade_3 é longe **e** proximidade_7 é longe **e** proximidade_8 é perto, **então** direção a tomar é virar à direita pouco
04	**Se** proximidade_1 é perto **e** proximidade_2 é longe **e** proximidade_3 é longe **e** proximidade_7 é perto **e** proximidade_8 é perto, **então** direção a tomar é virar à direita muito
05	**Se** proximidade_1 é longe **e** proximidade_2 é longe **e** proximidade_3 é perto **e** proximidade_7 é perto **e** proximidade_8 é longe, **então** direção a tomar é em frente

Tabela 6.2 – Conjunto final de regras difusas

O autor comenta sobre a necessidade de se fazer um conjunto de regras condicionais, adicionadas de pelo menos uma outra, chamada de incondicional. Esta última estabelece uma restrição de contorno no espaço difuso, permitindo que o controlador tenha sempre uma resposta de saída. Isto é reforçado por Saffiotti (1997) que alerta sobre o principal problema encontrado em controladores baseados em comportamento devido a coordenação destes comportamentos:

> *"... considere que um robô que encontra um obstáculo inesperado enquanto segue um caminho, e suponha que existe a opção de rodear o obstáculo pela esquerda ou pela direita. Esta escolha é indiferente para o comportamento de desvio de obstáculos"*.

No entanto, o autor comenta que para o comportamento de seguir uma trajetória, *"...uma escolha pode ser dramaticamente melhor que a outra."* (SAFFIOTTI, 1997)
Utilizou-se este conceito neste trabalho, o que será comentado na próxima seção, sobre a validação e teste de todo o sistema.

6.4.4 Definição do método de defusificação e refino do sistema

Conforme exposto na seção 6.4 optou-se por escolher um método de defusificação inicial, possibilitando a confecção de um modelo de simulação para o seu conseqüente refino.

Por razões de simplicidade de construção, princípio este que acompanha todos os passos dados neste trabalho, utilizou-se como ponto de partida o método do centro de gravidade como modelo de defusificação.

Para validar a capacidade do controlador testado, foi criado um modelo computacional para ser rodado no MATLAB 4.2.C, utilizando as especificações até agora descritas, conforme Anexo C deste trabalho.

Ante a dificuldade de simular matematicamente o modelo, utilizou-se um conjunto de 2000 *n-uplas* de dados aleatórios, referentes ao nível de tensão nos sensores, possibilitando comparar os valores de entrada e de saída.

Foi rodado um conjunto de dados para os métodos de defusificação por centróide de área e pela média dos máximos que, respectivamente, forneceram 5,4% e 23,6% de erro, calculado sobre o estado de tensões do sensor da direção 1.

Adotou-se que para todo valor fornecido por este sensor com grau de pertinência superior a 0,5 o robô deveria virar à esquerda ou à direita. Somadas as oportunidades em que isto não aconteceu e dividindo o valor pelo total de vezes que a direção 1 foi tomada, encontrou-se o erro percentual para o controlador.

Cotejados, o método dos centróides revelou uma variação do valore de saída mais tênue a mudanças bruscas dos valores de entrada, o que é reforçado por Cox, que diz que para este método *"... os valores defusificados tendem a se mover suavemente dentro da região de saída difusa"*(COX, 1994). Isto é facilmente comprovado quando se visualiza o gráfico mostrado na figura 6.6. Ai são apresentados os valores de saída para cada uma das 2000 *n-uplas* utilizadas, ordenados em forma crescente. Em cor verde estão os valores da média do máximo e em cor vermelha os do centróide da área. Os platôs que se observam em verde demonstram que o método do máximo é de baixa flexibilidade para este tipo de controlador.

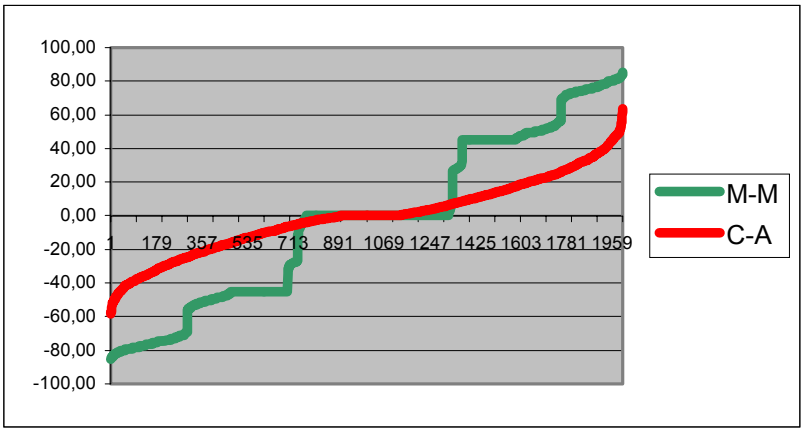


Fig. 6.6 – Valores de saída para os métodos de defusificação

Vale observar que o conjunto de regras utilizado é o da tabela 6.2 e que a forma dos conjuntos difusos é a de sino.

A opção pelo método de defusificação por centróide de área tornou-se a escolha natural, acrescida, ainda, à facilidade computacional a ele associada.

Outro ponto importante a ser considerado no tratamento do controlador apresentado é o tocante ao formato das funções de pertinência de cada conjunto difuso. Para validar o modelo desenvolvido, optou-se pelo teste utilizando o formato de sino – anteriormente descrito – e o formato triangular.

Validando as observações de Cox (1994) e Klir & Yuan (1995) sobre a existência de controladores invariáveis no resultado com relação à forma dos conjuntos difusos envolvidos, percebeu-se, no gráfico da figura 6.7, que os valores fornecidos pelos diferentes formatos da função são bem próximos, havendo ligeiras modificações em situações adjacentes ao ângulo de saída 0 e maior dispersão para os valores que vão de 20 a 40 graus à esquerda ou à direita.

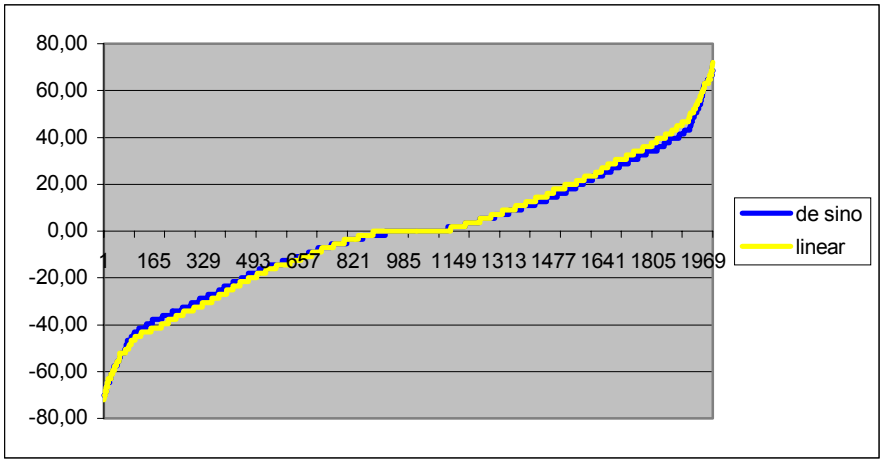


Fig. 6.7 – Valores de saída para os formatos dos conjuntos

Matematicamente, no entanto, atendo-se aos valores que orientam o robô a seguir na direção 1, ou seja, 0ºcomo foi descrito, os percentuais de erro são 4,8% para os conjuntos triangulares e 5,4% para os conjuntos em forma de sino. Além deste fator importante, ou seja, uma melhor precisão no tocante a quando seguir em frente – que na verdade é o ponto chave do controlador –, a escolha dos conjuntos triangulares mostrou-se mais vantajosa também por serem facilmente armazenados e contarem com baixo custo computacional em comparação aos em forma de sino.

Definido o método de defusificação e o formato dos conjuntos, passou-se ao refino das regras do sistema. Deve-se lembrar que os modelos anteriormente simulados trabalharam com o conjunto de regras da tabela 6.2, ou seja, 5 regras, envolvendo as diversas variáveis.

Existe, entretanto, um valor elevado de erro que pode provocar o choque do robô com algum obstáculo colocado a sua frente. Para evitar isto, pensou-se na utilização de uma regra não condicional conforme definido por Cox (1994) e Ross (1995).

Ross (1995) diz que as regras não condicionais podem seguir o formato apresentado pelas regras condicionais, na típica topologia *if-then*. Assim, para verificar a validade da inclusão de mais uma regra no sistema, adotou-se no modelo simulado no MATLAB a regra 06 dada por: ***Se*** *proximidade_1 é perto,* ***então*** *direção a tomar é virar à esquerda muito*. O resultado encontrado foi significamente melhor, baixando o erro calculado para 2,9%. Diante desta situação, tentou-se adotar mais uma nova regra, na forma: ***Se*** *proximidade_1 é perto,* ***então*** *direção a tomar é virar à esquerda pouco.*

Os resultados encontrados foram surpreendentes, obtendo-se um erro igual a 0%. No entanto, percebeu-se que o robô apresentou tendência de apenas virar à esquerda, não ocorrendo em nenhum instante a ação de virar à direita, independentemente do nível de tensão em seus sensores. Com a utilização do conjunto de seis regras, não houve tendência do robô virar mais em uma direção do que em outra, o que ratificou a escolha deste conjunto.

Esta seção conclui o embasamento teórico adotado no controlador do robô, cabendo agora explanar sobre a aplicação prática. Para tal, nas seções seguintes serão abordados a linguagem de programação utilizada e as implementações nas duas plataformas envolvidas no projeto, a para IBM-PC e a para 8051.

6.5 A linguagem C

Para implementação do projeto pensou-se em utilizar um software de comando remoto, baseado em PC, que possibilitasse ao usuário as informações básicas de retorno do robô que contaria com um software baseado em microcontrolador, responsável pela navegação e controle.

Optou-se por esta configuração – utilizando um sistema microcontrolado no robô – para evitar o processamento, mesmo a baixo nível, no microcomputador, simplificando dois pontos importantes: o processamento fora do robô e a linha de comunicação entre ele e o microcomputador. O primeiro é garantido em razão de o processamento dos dados ser dividido entre robô e microcomputador; o segundo evita o desenvolvimento de um complexo sistema de comunicação *full-duplex* entre a unidade remota e o microcomputador, uma vez que dados dos sensores trafegariam no sentido RAU – PC no mesmo instante que dados referentes a que ação tomar, ou a que direção seguir, passariam do PC para o RAU.

Para um sistema baseado em tempo real, como no presente caso, é possível imaginar o grau de velocidade e estabilidade que seria necessário assegurar num sistema *full-duplex* e a simplificação decorrente da solução adotada.

É inegável que um sistema baseado em um microcomputador, como o Pentium 233 MHz que foi utilizado no experimento, tem resultados mais rápidos, podendo ainda realizar tarefas mais complexas. No entanto, visto a simplicidade computacional que se obteve no controlador do robô, o modelo microprocessado passou a ser uma excelente opção.

Neste trabalho adotou-se a nomenclatura **comando** para o software que roda no microcomputador e permite ao usuário *comandar* as ações do robô, e **controle** para o software instalado no robô, responsável pelo controle a baixo nível do RAU.

O software de controle – para plataforma Intel 8051 – foi desenvolvido inteiramente em ANSI C, utilizando-se o MS-DOS C51 Compiler, versão 3.07 da Franklin Software Inc. Este programa, que roda em modo DOS, possibilita a compilação e *link-edição* do software para rodar em plataforma 8051, permitindo que toda a programação seja definida e testada em um PC.

Já o software de comando foi desenvolvido utilizando-se modernas ferramentas, baseadas em orientação ao objeto. O compilador utilizado na confecção da interface foi o Borland C++ Builder, e na elaboração dos comandos a baixo nível, como leitura de dados do joystick e comunicação serial o Borland C++ 4.52. Em ambos, a linguagem de programação utilizada foi a C, com ligeiras adaptações em face do compilador adotado.

A linguagem C surgiu no início da década de 70, no *Bell Laboratories*, da idéia de Brian W. Kernighan e Dennis M. Ritchie de utilizarem uma poderosa ferramenta de programação para desenvolver um sistema operacional já em estudos naquele laboratório, o UNIX. (MÓDULO, 1989).

Esta linguagem de programação é considerada a ideal para se trabalhar com os microcomputadores do tipo IBM PC por oferecer um excelente controle de estruturas e dados, funções de comunicação, gráficas e matemáticas além de possibilitar implementações a baixo nível, acessando qualquer parte do sistema (BARKAKATI, 1990).

A título e comparação, as outras linguagem disponíveis para os microcontroladores 8051 hoje são o BASIC e o Assembler. Enquanto a primeira não permite um bom

controle do fluxo de programação, a segunda traz dificuldades ao programador que não conhecer profundamente o ambiente onde rodará o programa.
Se olharmos agora para a plataforma IBM PC, as opções são muito maiores, passando do COBOL até o Delphi, no entanto, comparadas, a linguagem C mostrou ser a melhor opção, tendo ainda as vantagens de ser uma ferramenta portável e devidamente testada pelo longo tempo de comercialização.

6.6 A Estrutura do Software de Controle

Basicamente, o software de controle foi montado sobre o fluxograma apresentado na figura 6.8. Primeiramente é recebida a entrada do joystick e avaliada para verificar se há algum comando, ou seja, se o usuário inclinou a haste para algum dos lados. Se há comando, o próximo passo é verificar se o botão foi pressionado. Em caso afirmativo, é chamado o procedimento para movimentação da câmera, senão, é averiguado se o comando do usuário é para frente. Confirmado, verifica-se o ângulo fornecido pelos sensores colocados no robô. Esta verificação é baseada no controlador difuso discutido na seção 6.4, ou seja, em face do campo de sinais apresentado pelos sensores na entrada do microcontrolador o programa calcula um valor, inferindo a base de conhecimento.

Se o ângulo calculado for igual a zero, ele permite que o robô se locomova à frente, no entanto, se esse valor for diferente de zero é provocado um giro na unidade, incrementando no caso do ângulo ser à direita, ou decrementando no caso do ângulo ser à esquerda, utilizando a equação 4.5 (ver capítulo 4). Uma vez alcançada a direção definida pelo controlador, o programa faz com que a unidade volte a se locomover de acordo com a indicação do usuário. No caso de o comando fornecido ser virar à direita ou à esquerda, o programa permite o giro da unidade no seu próprio eixo, enviando o sinal de giro às rodas enquanto o joystick estiver inclinado.

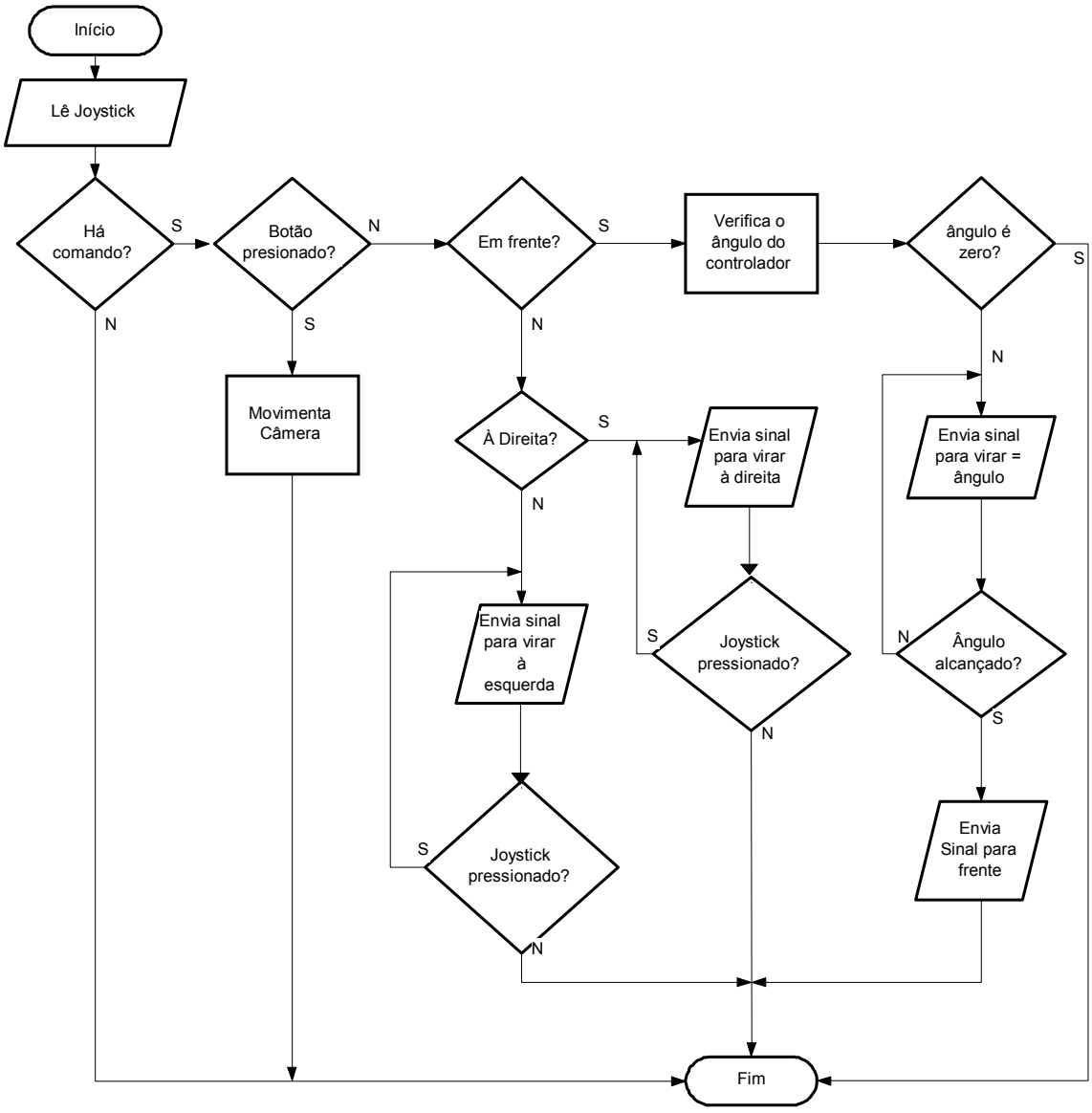


Fig. 6.8 – Fluxograma do programa de controle do RAU

Tendo por origem o eixo que aponta do centro para a frente do robô, conforme figura 6.4, ângulos à direita são tratados pelo programa como valores positivos, ao passo que valores de ângulo à esquerda recebem sinal negativo.

Para movimentar a câmera utilizou-se um fluxograma similar, conforme mostrado na figura 6.9.

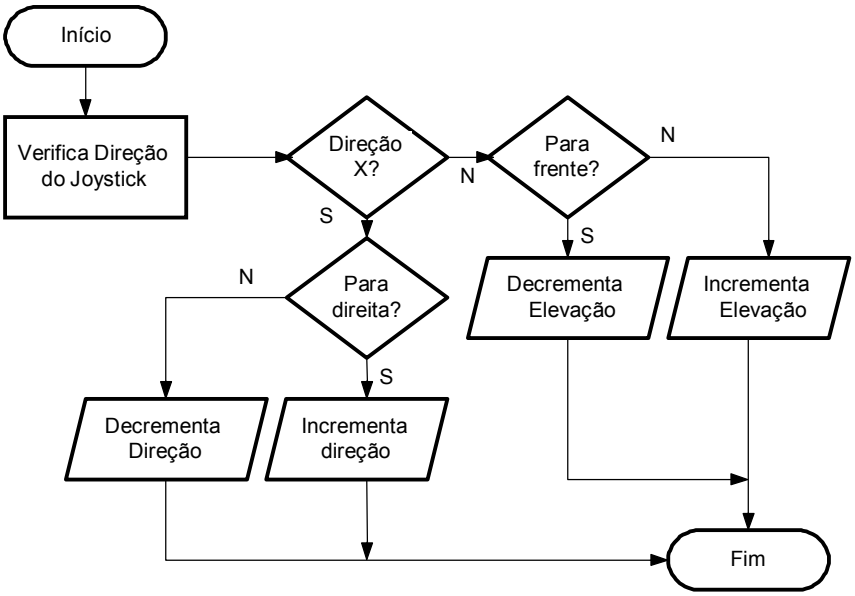


Fig. 6.9 – Fluxograma do programa de controle da câmera

Primeiramente é verificada a direção do *joystick*, dividida em X e Y, conforme mostrado na figura 6.10. Se a haste estiver inclinada sobre a Y, o software constata se para a frente ou para trás. Em função desta posição ele decrementa ou incrementa a elevação da câmera, respectivamente. O mesmo ocorre com relação a direção X. Mover o joystick para a direita significa incrementar a direção, enquanto movê-lo à esquerda provoca reação oposta pelo controlador da câmera.

Fig. 6.10 – Definição dos eixos do joystick

Desta maneira, em um único programa e com um único controlador é possível controlar o robô e a câmera acoplada a ele, mais uma vez priorizando a diminuição de custos e o incremento da simplificação do projeto. Observa-se que pela inexistência de obstáculos na movimentação da câmera, o software apenas repassa informação aos motores, não necessitando cotejo com valores calculados pelo controlador.

6.6.1 Detalhes da implementação

O microcontrolador foi dividido em portas de entrada e saída para comunicação com os sensores e motores envolvidos em todo o projeto. Foram conectados ao 8051 dois motores de passo com 1,8 graus/passo, responsáveis pela movimentação do robô; cinco conjuntos de sensores infravermelhos com conversores analógico-digitais capazes de indicar ao controlador o nível de proximidade dos obstáculos e dois motores de passo de 3,6 graus/passo para movimentação da câmera, além das portas de comunicação serial e de alimentação.

A comunicação com os sensores ficou conforme o mostrado na tabela 6.3

Pino	*Sensor relacionado*
Pinos 01, 02 e 03	Valores de entrada do sensor na direção 1
Pinos 04, 05 e 06	Valores de entrada do sensor na direção 2
Pinos 07, 08 e 10	Valores de entrada do sensor na direção 3
Pinos 11, 12 e 13	Valores de entrada do sensor na direção 7
Pinos 14, 15 e 16	Valores de entrada do sensor na direção 8

Tabela 6.3 – Configuração dos pinos utilizados pelos sensores

Os conversores analógico-digitais trabalham com uma saída de 3 bits. Assim, o valor contínuo fornecido pela variação de tensão dos sensores infravermelhos é discretizado em 8 valores diferentes de tensão, conforme mostrado na tabela 6.4. Vale ressaltar que o bit mais significativo é mostrado como o primeiro pino na tabela 6.3.

Entrada em volts	*Saída Binária*
0,000	000
1,286	001
2,572	010
3,858	011
5,144	100
6,430	101
7,716	110
9,000	111

Tabela 6.4 – Valores analógicos e digitais de saída dos sensores

O valor binário é processado pelo microprocessador, utilizando o programa inserido em sua memória flash, cuja listagem pode ser vista no Anexo A deste trabalho. Da mesma forma, fez-se necessário a comunicação entre o 8051 e os motores de passo. Assim, foram utilizados os seguintes pinos mostrados na tabela 6.5

Pino	*Descrição*
Pinos 36 a 39	Aciona motor de passo esquerdo
Pinos 32 a 35	Aciona motor de passo direito
Pinos 25 a 28	Aciona motor de passo de azimute da câmera
Pinos 21 a 24	Aciona motor de passo de elevação da câmera

Tabela 6.5 – Configuração dos pinos utilizados pelos motores de passo

Além destes pinos foram utilizados os de número 40 para alimentação do microcontrolador e o 20 para o terra, enquanto a entrada serial do joystick, enviada por ondas de rádio, se fez pelo pino 10. Os sinais do joystick serão definidos mais adiante, na próxima seção deste trabalho, juntamente com o comentário relativo ao programa de comando do RAU.

6.7 O Software de comando

O software de comando do sistema, responsável pelo controle remoto do robô, tem basicamente quatro funções:

1. iniciar o robô RAU;
2. verificar o joystick;
3. iniciar a transmissão de dados e
4. controlar o joystick.

A primeira função é testar se a comunicação entre unidade remota e computador estão em harmonia. Esta função estabelece a comunicação com o robô, enviando-lhe um simples comando de virar à esquerda. Como o robô será sempre iniciado próximo ao computador, o controle da resposta a esse primeiro comando pode ser visual, tornando desnecessária uma confirmação por parte dele.

A segunda função básica diz respeito a interface entre usuário e programa. Ela é responsável pela calibração do joystick de tal forma que o ponto morto, ou seja, a

situação em que o programa fornecerá ao robô a informação de permanecer parado, possa ser corretamente definido.

A terceira e quarta funções praticamente se misturam durante a operação do robô. A transmissão de imagem por parte do robô RAU ocorre a partir do momento em que ele é ligado, enquanto o envio de sinais do joystick para esse inicia quando se clica o botão *Run program*[1]. Para evitar problemas durante a operação do sistema, este botão só aparece ativo após a calibragem do joystick.

A utilização de uma ferramenta visual como o C++ Builder permitiu maior atenção à interface com o usuário, levando-se em consideração aspectos ergonômicos. O resultado pode ser visto na aparência final do programa, mostrada na figura 6.11.

Fig. 6.11 – Interface do software de comando

No desenho da interface com o usuário também foram levadas em conta as funções do programa. Assim, dividiu-se a tela em três grupos principais: o primeiro de comandos de iniciação do sistema[2], o segundo de operação do RAU e o terceiro de controle de execução do software.

---

[1] Toda a interface do programa foi desenvolvida em língua inglesa, visando a participação deste trabalho em eventos internacionais.

[2] Esta divisão se deve em parte também ao sistema a ser adotado no robô. Quando começou a ser desenvolvido o software de comando, existia a possibilidade de se utilizar o Microcontrolador Motorola 68HC11, que necessitaria do envio do programa por parte do computador, uma vez que não existe a memória flash neste tipo de *chip*. Com a opção pelo 8051, este problema foi eliminado, mas por se achar conveniente a forma como a interface foi elaborada, manteve-se esta divisão.

Ao primeiro grupo foi atribuído o comando de “ligar o robô RAU”. Na verdade, isto corresponde à ação de enviar ao robô o sinal de virar à esquerda, possibilitando ao usuário verificar se o canal de transmissão de dados está operando normalmente. Os elementos de interface com o usuário são um botão para ligar o robô e um termômetro indicativo da progressão do comando de iniciação.

O grupo de operação do robô é dividido em comandos de execução e elementos de monitoramento. Nos comandos de execução o usuário pode calibrar o joystick – seguindo as recomendações fornecidas pelo software de forma a calcular a região de ponto morto – e executar o programa, que se resume em manter a comunicação entre as duas unidades do protótipo.

Ao clicar no botão *Run program* o usuário informa ao software para passar ao robô todas as informações de posicionamento lidas do joystick.

Os elementos de monitoramento são compostos por duas *trackbars* que indicam a posição do joystick no eixo *X* e *Y*, de tal forma que o usuário possa verificar como o comando do joystick está sendo interpretado pelo programa.

Por fim o terceiro grupo, responsável pelo controle de execução do programa, tem como elementos de interface três botões. O primeiro, intitulado *Reset*, zera todos os valores de calibragem do joystick. O segundo permite o fechamento do programa, enquanto o terceiro aciona as operações de ajuda do sistema. Todos podem ser acessados a qualquer instante pelo usuário.

No topo da tela o usuário tem a imagem gerada pela câmera *on-board* e capturada por uma placa modelo LGV-5444TV, da LG International Corp. baseada no *chipset* Cirus Logic de 64 bits, padrão PCI. Esta tela é gerada pelo software *TV Program* que acompanha a placa e que deve ser inicializado antes de se rodar o programa de comando do RAU.

Como a imagem não necessita controle, uma vez que o sistema proposto apenas transmite informações do meio onde o RAU se encontra, utilizou-se este software e a referida placa, ganhando-se em tempo de desenvolvimento do trabalho, pois a criação de funções que pudessem ler dados de vídeo inviabilizaria a proposta.

Lembrando a configuração proposta, fez-se necessário criar a interface entre o joystick e o RAU, desenvolvido no C++ 4.52, de forma que o software permita ao usuário calibrar o joystick e enviar a sua posição para o controlador instalado no RAU.

Para efetuar a leitura dos valores do joystick calculou-se uma região chamada de região de ponto morto. Seu cálculo é baseado no seguinte procedimento: dada a posição central do joystick, calcula-se o valor médio entre esta coordenada e as extremidades máximas, definindo o valor em cada eixo. Assim, se o joystick estiver na região de ponto morto, o valor 0 é enviado pela porta serial para avaliação do controlador, no entanto se o valor for exterior ao da região de ponto morto, será enviado ao RAU um valor de referência conforme listado na tabela 6.6.

Pode-se perceber que não foram levadas em consideração as posições diagonais, como para trás e à esquerda por exemplo, pois nestes casos, o *eixo X* do joystick, responsável pelo movimento à esquerda e à direita terá sempre preferência, fazendo com que o RAU receba o comando de virar para um lado ou para o outro. Pensou-se desta forma pois o RAU gira no próprio eixo, não havendo portanto o risco de se chocar com algum obstáculo enquanto estiver mudando de direção.

Posição do joystick	***Valor de referência***
Ponto morto	NADA
À frente	EM_FRENTE
À direita	DIREITA
À esquerda	ESQUERDA
Para trás	NADA
À frente (botão pressionado)	CAM_BAIXO
À direita (botão pressionado)	CAM_CIMA
À esquerda (botão pressionado)	CAM_ESQUERDA
Para trás (botão pressionado)	CAM_DIREITA

Tabela 6.6. – Valores dados pelo programa em relação ao joystick

Vale ressaltar que os valores da tabela foram definidos conforme a utilização de praxe na programação em C, ou seja, foram atribuídos valores numéricos a cada uma destas variáveis. A representação literal, conforme mostrado é apenas para melhor entendimento da ação solicitada pelo usuário. Assim, se o valor lido do joystick é "CAM_ESQUERDA", significa que o usuário deseja que a câmera vire à esquerda.

Este sinal é enviado pela porta serial do micro, padrão RS 232, até o microcontrolador, a uma velocidade de 110 bps. Na saída da porta do micro foi adaptada uma interface

que envia o sinal da tabela 6.6 ao receptor instalado na RAU, através de rádio frequência.

Como foi feito com relação às posições diagonais, o mesmo cuidado se teve com a posição para trás, tomando valor nulo. Neste caso, para simplificar o protótipo reduzindo o número de sensores envolvidos, optou-se por não permitir o movimento reverso simultâneo dos motores, conforme foi justificado no capítulo 5.

6.8 Conclusão

Com este capítulo conclúi-se a fundamentação teórica do trabalho. Neste ponto, a definição de todos os componentes do sistema de telepresença já está concluída, restando então a parte prática.

Com relação a parte prática, serão considerados a montagem do robô e os testes considerados relevantes para o perfeito funcionamento do sistema, tópicos estes que serão descritos no próximo capítulo.

Capítulo 7 - Montagem e teste do robô RAU

7. 1 Introdução

Após a definição detalhada dos componentes que constituem este projeto, iremos nos ater neste capítulo à montagem e teste dos componentes do sistema. Serão também apresentadas, em decorrência desta própria montagem, algumas modificações no projeto proposto no capítulo 5, que tornaram-no mais confiáveis e exeqüível.

7.2 A comunicação com o microcomputador

Na definição apresentada no capítulo 6 a unidade de processamento de informações, representada no projeto pelo microcomputador, comunicava-se com a unidade de ação remota (RAU) através da porta serial, padrão RS 232. Este sinal seria enviado diretamente ao pino 10 do microcontrolador, que o interpretaria.

Durante a montagem do robô, principalmente pelo fato de existirem kits já prontos, verificou-se que a comunicação serial poderia ser facilmente substituída por um comando capaz de enviar sinais digitais a três portas do microcontrolador, de tal forma que a combinação destes 3 bits pudesse fornecer todas as ações definidas pelo usuário. Assim, toda a parte de software prevista para habilitar esta comunicação, como ajuste de velocidade e paridade, mostrou-se desnecessária.

O novo modelo de comunicação entre a unidade de processamento de informações e a unidade de ação remota seria baseado no que mostra a figura 7.1.

Dos blocos mostrados na figura 7.1 o *encoder*, o *decoder* e a interface já existiam como kits, bastando apenas pequenos ajustes para que pudessem funcionar no presente projeto.

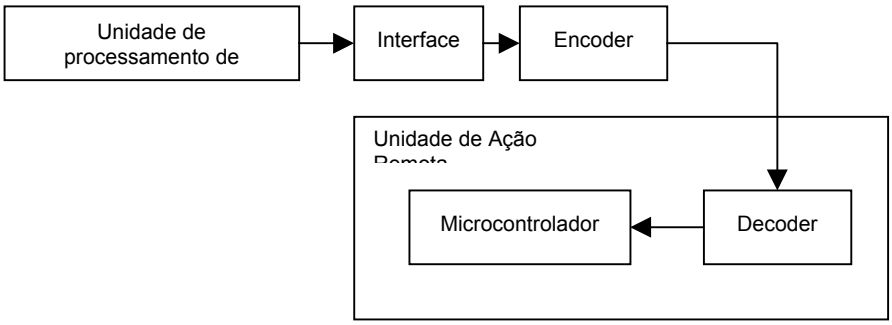


Fig. 7.1 – Nova configuração da comunicação entre as unidades do protótipo

A comunicação entre as duas unidades tem inicio quando o sinal do joystick – conforme mostrado na tabela 6.7 – é enviado pelo software. Este sinal é recebido pelo circuito de acionamento do *encoder* a uma velocidade de 2400 bps, com 8 bits, 1 stop bit e sem paridade. Acionado, o encoder emite um conjunto de bits com sincronismo para modular um transmissor de radiofreqüência. Desta forma, o sinal é enviado por onda de rádio ao receptor – no robô – que o demodula, quadra e o remete ao decoder.

O decoder recebe e decodifica o sinal e – através do chaveamento dos chips 4016 – transforma-o em uma nova seqüência de bits – agora no número de 3 – conforme apresentado na tabela 7.1. Deve-se observar que o sinal para trás é nulo, conforme descrito na seção 6.6

Posição do Joystick	Sinal binário
Neutro	000
Para frente	111
Para direita	100
Para esquerda	001
Para trás	000
Para frente com botão pressionado	010
Para trás com botão pressionado	110
Para direita com botão pressionado	011
Para esquerda com botão pressionado	101

Tabela 7.1 – Valores produzidos pelo *decoder*

Por fim, os valore binários fornecidos pelo decoder alimentam por fim os pinos 26, 27 e 28 do microcontrolador que, através de seu software, perfaz a ação desejada pelo usuário.

Com esta nova configuração a comunicação entre as duas unidades passou a ser codificada, portanto menos susceptível a interferências do meio ambiente, mais

robusta e capaz de trabalhar em ambientes com fontes emissoras de ruídos e ondas de rádio.

7.3 Teste dos sensores infravermelhos

O primeiro teste a ser realizado com os sensores infravermelhos foi o de sensibilidade à detecção de obstáculos. Para tanto se montou em uma bancada um par de sensores com uma régua milimetrada, um anteparo branco com as dimensões de 65 x 75 x 40 mm e um multiteste para medir a voltagem de saída. O anteparo foi colocado no final da escala da régua – 30 centímetros – e foi sendo aproximado a cada 5 milímetros, ao passo em que era medida a tensão de saída.

Na primeira configuração foi utilizado o sensor padrão com o oscilador originalmente montado. Os dados obtidos revelaram uma variação muito grande de tensão entre 19,5 a 18 cm de distância de um obstáculo o que não é desejável para o projeto. Diante desta situação pensou-se em alterar o capacitor do oscilador do infravermelho, primeiramente diminuindo sua freqüência. Assim, foram substituídos os dois capacitores originais de 10 nF por outros de 100nF.

Observou-se que a freqüência foi abaixada na razão de 3 e a região de maior variação de tensão foi suavizada, passando para a distância entre 21 a 18,5 cm.

Para verificar se estes dados expressavam uma melhoria no projeto foi utilizado o raciocínio inverso, ou seja, aumentar a freqüência do oscilador do infravermelho. Para isto, os capacitores foram novamente substituídos por dois de 6,8 nF. Os resultados comprovaram que o aumento da freqüência resultou em uma piora da resposta dos sensores. A região de variação de tensão passou a ser entre 14 e 12 cm, porém a grande falta de sensibilidade dos sensores à região compreendida entre 30 a 14 cm de distância levou à escolha da segunda opção, ou seja, a dos capacitores de 100 nF.

Os resultados podem ser observados no gráfico da figura 7.2, de acordo com os capacitores utilizados.

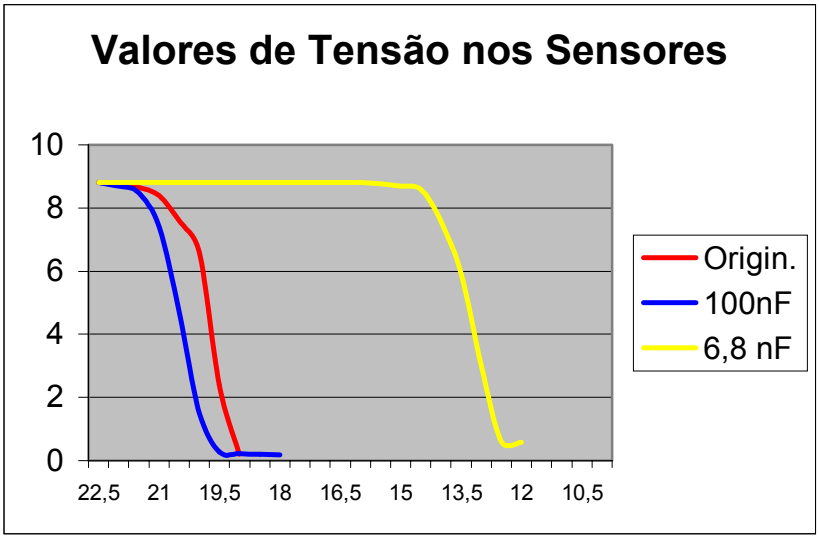


Fig. 7.2 – Valores de tensão de saída dos sensores

Um novo teste teve que ser executado para se verificar qual posição relativa entre o emissor e o receptor de infravermelho forneceria os resultados mais confiáveis. Isto se deve ao fato dos sensores infravermelhos trabalharem com reflexão, ou seja, seu funcionamento é como um raio de luz normal: enviado, parte sofre refração e parte é refletida. Esta última é então recebida no sensor e avaliada, através da tensão, o quanto foi refletido. Quanto mais próximo maior é a reflexão e, conseqüentemente menor é a tensão de resposta do receptor.

Para este segundo teste, montou-se uma bancada de testes conforme a figura 7.3

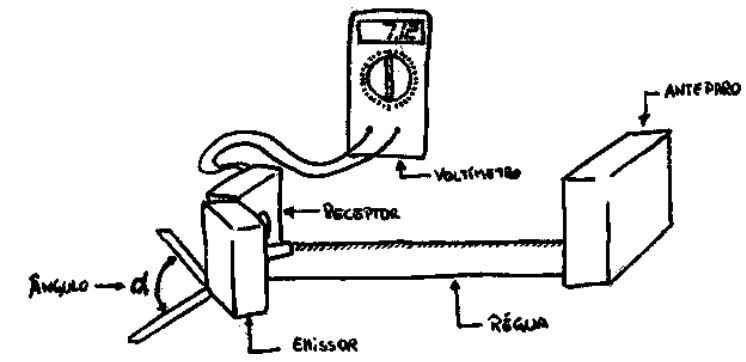


Fig. 7.3 – Bancada de testes para verificar o ângulo relativo entre os sensores

Variou-se o ângulo formado pelas duas unidades do infravermelho, o emissor e o receptor. Após fixá-los na bancada, colocava-se um anteparo branco com as dimensões

de 65 x 75 x 40 mm no final da escala da régua – 30 centímetros. O anteparo foi sendo deslocado a cada 5 milímetros e medida a tensão de saída do receptor, conforme pode ser observado no gráfico da fig. 7.4.

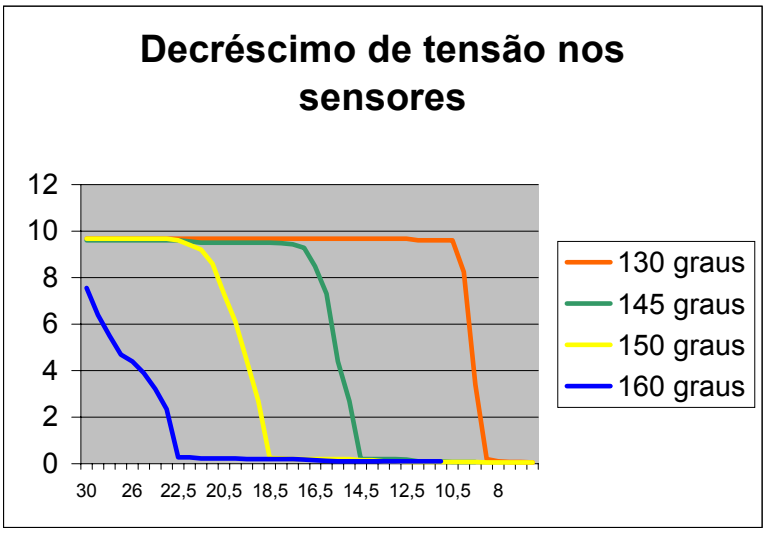


Fig. 7.4 – Valores de tensão de saída do receptor
versus ângulo entre os sensores

A primeira conclusão que se pode tirar do testes realizados é que a sensibilidade à presença ou não do obstáculo não é variável com relação à posição relativa dos sensores, pois para todos os ângulos utilizados obteve-se praticamente a mesma tangente para as curvas, com exceção da posição relativa de 160 º.

Outra conclusão é que a distância em que o obstáculo é sentido é diretamente proporcional ao ângulo entre os sensores, como pode ser obtido da leitura do gráfico da figura 6.4. Este fato nos permite escolher o ângulo entre os sensores em função da atribuição dada a ele, ou seja, se em determinada direção precisamos detectar obstáculos a uma grande distância podemos utilizar ângulos maiores, enquanto que para detecção de obstáculos mais próximos menor será o ângulo aplicado. Como contamos com 5 direções principais, adota-se este artifício para possibilitar que o RAU não tenha problemas ao passar por uma porta ou percorrer um corredor. Assim, utilizaremos o ângulo de 130º para as direções 3 e 7 e o de 160º para as demais direções. Este último foi escolhido não só pela distância mas também porque a região de transição entre as tensões máximas e mínimas ser maior do que nos outros casos.

Isto não interfere no controlador difuso porque os valores lingüísticos **perto** e **longe** continuam a ser os mesmos; o que difere é a definição de perto e longe para cada direção, o que também é aceitável.

O último teste realizado com os sensores infravermelhos foi baseado no trabalho de Botelho (1996) e determina a área de varredura do sensor. Foi montada uma bancada de testes com o emissor e o receptor infravermelho fixados no centro. Ao seu redor foi marcada uma escala angular com intervalo de 10º e com raio de 5 centímetros, por onde se deslocaria o anteparo, como pode ser observado na figura 7.5.

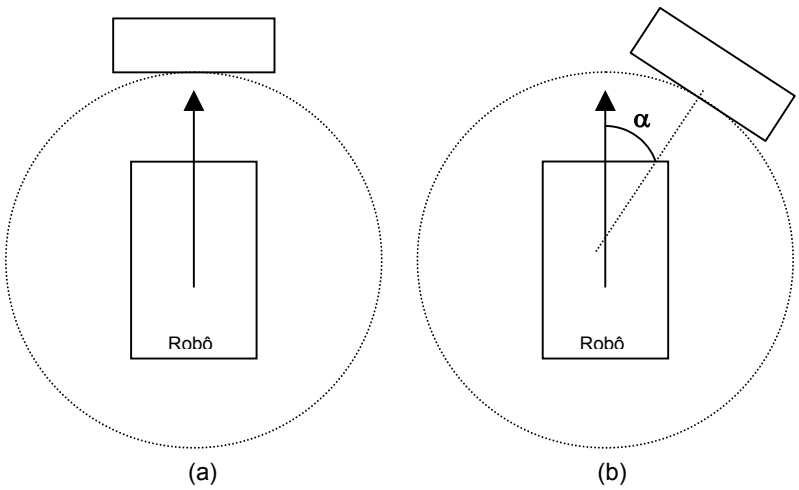


Fig. 7.5 – Montagem do teste de varredura com o anteparo na posição inicial (a) e com deslocamento de α graus (b)

Para cada deslocamento na escala angular foi medido o nível de tensão nos sensores, determinando assim a área em que ele apresenta "sensibilidade" à presença de obstáculos. De posse dos dados coletados, utilizou-se o MS Excel para gerar o gráfico da área de varredura, conforme apresentado na figura 7.6.

Através deste teste pode-se observar que o robô RAU é capaz de perceber a presença de obstáculos na área compreendida entre 250º a – 30º com a origem na direção do sensor 7, o que perfaz um ângulo de varredura de 260º. Constatou-se também uma excelente característica do sistema: a sobreposição da área de varredura de cada sensor isoladamente. Pode-se perceber pelo gráfico que existe entre eles uma área de 20º que é comum, o que significa dizer que desta forma o controlador consegue obter mais dados para a sua tomada de decisão.

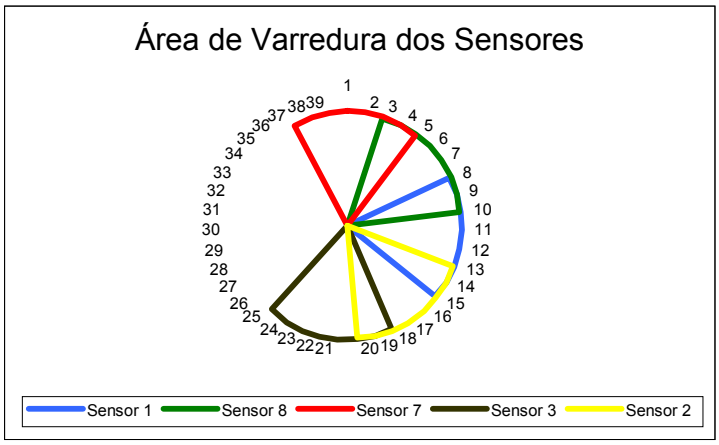


Fig. 7.6 – Gráfico obtido com o teste de varredura

Depois desta bateria de testes os sensores foram montados no chassi do robô com um parafuso de fixação, mantendo os ângulos anteriormente determinados.

7.4 Teste de resposta do microcontrolador

Outro teste realizado foi no controlador do RAU. Após a programação do microcontrolador ATMEL 89C52 utilizado, ele foi montado no soquete da placa de circuito impresso projetada para o protótipo e verificou-se a programação.

De acordo com o que foi descrito até aqui os pinos de entrada de dados do usuário, ou seja, seus comandos, são os de número 26, 27 e 28. Através de um injetor de sinais gerou-se a seqüência de dados mostrada na tabela 7.1 e verificou-se, com o uso de um osciloscópio, a saída dos pinos 1 a 8, responsáveis pelos motores que impulsionam a unidade e dos pinos 21 a 24, responsáveis pela comunicação com o motor de acionamento da câmera.

Aqui cabe uma explicação: os movimentos da câmera foram também modificados. Por não ser possível a adaptação dos motores Matsushita modelo MSDA020L ao suporte da câmera, o projeto foi alterado, utilizando apenas um motor MINEBA modelo 15 PM, similar aos utilizados nas rodas motrizes, responsável pelo movimento de elevação da câmera. O movimento de azimute pôde ser substituído pelo movimento do próprio robô, à esquerda e à direita. Em conseqüência as duas últimas linhas de dados da tabela 7.1 deixaram de ser utilizadas.

Foram feitas simulações de seguir em frente, virar para um lado e para o outro e movimentar a câmera para cima e para baixo, averiguando-se o funcionamento dos motores. É interessante também observar que a ordem das bobinas dos motores foi também testada e verificada, uma vez que não havia documentação, por tratar-se de material usado.

Os resultados foram muito satisfatórios: o sinal que era enviado pelas portas de comunicação do microcontrolador e passava em um chip 2004 – utilizado como driver dos motores – foi recebido pelos motores sem problemas.

7.5 Construção e teste dos conversores analógico-digitais

Os sensores infravermelhos funcionam de forma analógica, ou seja, com grandezas reais que variam infinitamente entre valores de um dado intervalo, gerando sinaistambém chamados de analógicos.

Estes valores não são detectados por um sistema digital, que trabalha somente com dois estados físicos possíveis, ou **zero** ou **um**. Para superar este problema, foi instalado na interface entre microcontrolador e cada sensor infravermelho um conversor analógico-digital, que "*...muda a variável contínua, ou analógica em uma série de pulsos digitais*" (GIBILISCO,1994), transformando a tensão fornecida pelos sensores em um conjunto de bits.

A construção dos conversores foi um dos pontos de maior complexidade do projeto, excluindo-se a programação do microcontrolador, uma vez que não existe no mercado um componente que perfaça a operação desejada.

Pela limitação do número de portas de comunicação disponíveis em nosso microcontrolador, optou-se pela codificação binária em 3 bits, resultando portanto em 2^3 estados, ou seja: 8. Estes estados são conseguidos através de uma rede resistiva, que permite uma codificação de 8 sinais binários, que vão do valor "00000000" ao "11111111".

Todavia, esta informação não pode ser passada diretamente ao microcontrolador, devido principalmente a falta de disponibilidade de suas portas de comunicação, além da escala ser do tipo de variação digital binária proporcional.

A solução encontrada foi projetar um circuito lógico utilizando 2 *chips* diferentes – 4070 e 4071 – que contam com as portas *XOR* e *OR* necessárias para a codificação dos 8 valores na seqüência de 3 bits mostrada na tabela 5.5, sendo estes valores os sinais de alimentação do comando do usuário no microcontrolador.

Comparativamente ao projeto, seu teste foi relativamente simples. Enquanto se aproximava um anteparo do sensor, media-se a tensão de saída com um multiteste, ao mesmo tempo em que, com o osciloscópio, fazia-se a varredura do sinal digital de saída do conversor. Vale lembrar que o circuito lógico foi testado e depurado no software Eletronics Workbench EDA, versão 5.0 da Interactive Img. Tec. Ltd. antes de ser construído.

7. 6 Montagem do protótipo.

A montagem do robô RAU seguiu um esquema definido de acordo com o acesso aos elementos de fixação. Assim, a seqüência de montagem – da qual algumas fotos integram o Anexo B deste trabalho – deu-se como segue:

1. fixação das longarinas à placa do chassi;
2. fixação do suporte de nylon da antena de envio de sinal de vídeo;
3. montagem da placa do controlador do robô e fixação no chassi;
4. fixação da unidade de envio de sinal de vídeo e antena;
5. fixação do decoder e unidade de recepção de sinal de comando;
6. fixação da câmera no motor de passo;
7. fixação do suporte do motor da câmera no chassi;
8. fixação do motor da câmera no suporte;
9. fixação dos sensores infravermelhos;
10. fixação e ajuste das 4 rodas livres do robô;
11. fixação dos motores de passo principais e das rodas motrizes;

12. fixação das baterias na parte inferior do chassi, uma anterior e outra posterior; e
13. conexão de todos os cabos, conforme marcado nas respectivas anilhas.

7.7 Testes finais do protótipo

Os testes finais do protótipo foram realizado utilizando dois grupos básicos de obstáculos – fixos e móveis – para avaliar a resposta do controlador à configuração do ambiente onde está inserido. Primeiramente utilizou-se uma área de 5 m^2, onde foram colocados obstáculos dispersos de acordo com a figura 7.7.

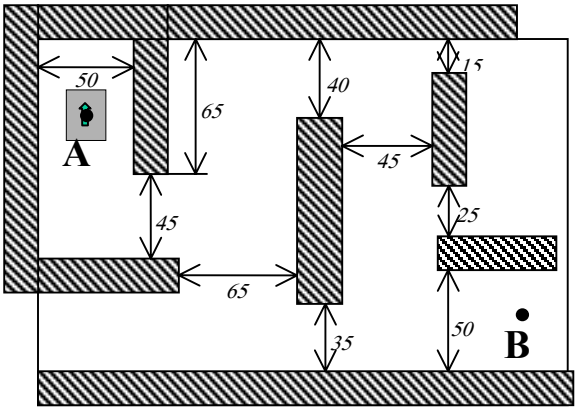


Fig. 7.7 Definição dos obstáculos estáticos

O robô foi posicionado no ponto ***A*** mostrado na figura 7.7, voltado para a parede, tendo como destino o ponto ***B***. Para avaliar a resposta do sistema, o operador permaneceu o tempo inteiro do teste com o joystick para frente, deixando para o controlador do robô a responsabilidade de encontrar o melhor caminho.

Percebeu-se que ao fazer a primeira curva, o robô se aproximou muito da parede interna, evitando o choque com a posicionada externamente. Isto ocorreu devido a falta de reflexão do feixe emitido pelo sonsor 8, que conjuntamente com o 7 poderiam disparar a regra que forçaria o robô virar a direita. Outro fator importante que contribuíu com esta situação foi a detecção da parede externa por 3 sensores.

Durante este teste, foi notado que o robô apresentava falta de sensibilidade nos sensores 2 e 8, pois ignorava qualquer obstáculo que existisse apenas nesta direção. O modelo para o MATLAB foi retomado, simulando-se a utilização apenas os dois, de

forma a testar o controlador sob estas condições. Neste caso, o erro acontecido na prática também mostrou-se presente na simulação.

Assim, para eliminar esta situação, adotou-se mais duas regras no controlador difuso: ***se*** *proximidade_2 é perto,* ***então*** *direção a tomar é virar à esquerda pouco* e, ***se*** *proximidade_8 é perto,* ***então*** *direção a tomar é virar à direita pouco*, o que melhou o desempenho dinâmico do robô.

Para os obstáculos móveis, utilizou-se um voluntário que cruzou a frente do robô através das linhas tracejadas da figura 7.8

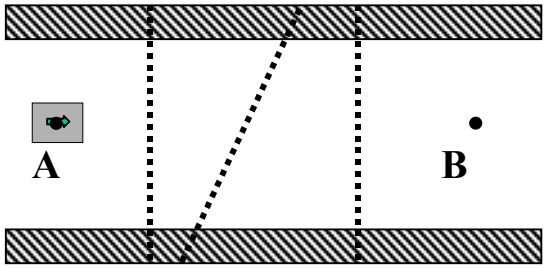


Fig. 7.8 – Localização e trajeto dos obstáculos móveis

Como no teste anterior, o joystick manteve-se inclinado para a frente, para avaliar a capacidade de decisão do controlador da máquina.

O robô alcançou o ponto B sem problemas, com pequenas variações em sua rota, evitando o choque com os obstáculos.

Um último teste foi realizado para avaliar o seu desempenho em um corredor, de forma a verificar sua aptidão para trabalhar em tubulações. Para isto, utilizou-se uma bancada de teste similar à mostrada na figura 7.8 com uma largura inicial de 90 cm. No seu meio, colocou-se um estreitamento que reduzia para 40 cm a largura útil, conforme pode ser observado na figura 7.9.

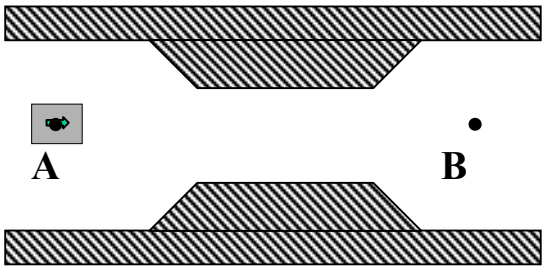


Fig. 7.9 – Teste do corredor com afunilamento

Na realização deste teste, como nos anteriores, o joystick foi colocado na posição que gerava o comando de seguir em frente.

Este teste comprovou a simetria do controlador difuso desenvolvido. O resultado esperado, que era passagem pelo estreitamento sem alteração da trajetória foi obtido, demonstrando que as regras que disparadas pelos sensores do lado direito do robô eram simultâneamente inibidas pelas associadas aos sensores do lado esquerdo.

Referências Bibliográficas

ALCÂNTARA, Eurípedes. Onde está o rei? **Veja**, São Paulo, n.18, p.114-120, maio/1997

ALVES, João Bosco da Mota, RESENDE, Marcos Ferreira, LEITE, Alzira et al. Ficcão, realidade e expectativa de robôs inteligentes baseados em comportamento. In: SIMPÓSIO BRASILEIRO DE AUTOMAÇÃO INTELIGENTE, 1, 1993, Rio Claro. **Anais...** p.145-154.

ANDEEN, Gerry B. **Robot design handbook**. New York: McGraw-Hill, 1988.

ARKIN, Ronald C. The impact of cybernetics on the design of a mobile robot system: a case study. **IEEE Transactions on systems, man, and cybernetics**, v.20, n.6, p.1245-1257, nov./dez., 1990.

ASIMOV, Isaac. **Visões de robô**. São Paulo: Círculo do Livro, 1994. 424p.

BARKAKATI, Nabajyoti. **The Waite Group's Turbo C++ bible**.Carmel: Sams, 1990. 1084p.

BENTIVEGNA, Darrin C., ALI, Khaled S., ARKIN, Ronald C. et al. **Design and implementation of a teleautonomous hummer**. Disponível na Internet. http://www.cc.gatech.edu/aimosaic/robot-lab/research/MissionLab/. set./1997.

BORENSTEIN, Johann. Multi-layered control of a four-degree-of-freedom mobile robot with compliant linkage. In: IEEE INTERNATIONAL CONFERENCE ON ROBOTICS AND AUTOMATION, 1993, Atlanta. **Proceedings...** p.3.7-3.12.

_____, EVERETT, H. R., FENG, Liqiang. **Navigating mobile robots: systems and techniques**. Wellesley: A K Peters, 1996. 225p.

_____, KOREN, Yoram. Motion control analysis of a mobile robot. **Transaction of asme, journal of dynamics, measurement and control**, v.109, n.2, p.73-79, 1987.

_____, _____. The vector field histogram – fast obstacle avoidance for mobile robots. **IEEE Journal of Robotics and Automation**, v.7, n.3, p.278-288, jun.,1991.

BOTELHO, Silvia Silva da Costa. **Desenvolvimento de sistemas inteligentes para controle de robôs móveis**. Porto Alegre, 1996. 124p. Dissertação (Mestrado em Ciências da Computação) – Instituto de Informática, Universidade Federal do Rio Grande do Sul.

BRAITENBERG, Valentino. **Vehicles:** experiments in synthetic psychology. Cambridge: MIT Press, 1984. 152p.

BROOKS, Rodney A. A robust layered control system for a mobile robot. **IEEE journal of robotics and automation**, v.RA-2, n.1, p.14-23, mar./1986.

_____. New approaches to robotics. **Science**, v.253, p.1227-1232, sept., 1991.

BRUMITT, Barry L., COULTER, R. Craig , STENTZ, Anthony. Dynamic trajectory planning for a cross-country navigator. In: SPIE MOBILE ROBOTS, 7, 1992, Boston. **Proceedings...** [s. l.]

BUNKER, K. J. Theoretical and practical approaches to motor vehicle steering mechanisms. In: GILES, J. G. (edit.). **Automotive technology series:** steering, suspension and tyres. London: Iliffe Books, 1968, 164p., v.1.

CHABRIS, Christopher. Lição histórica. **Veja**, São Paulo, n.20, p.105-106, maio/1997.

CHENG, Gordon , ZELINSKY, Alexander. Supervised autonomy: a paradigm for teleoperating mobile robots. In: IEEE/RSJ INTERNATIONAL CONFERENCE ON INTELLIGENT ROBOTS AND SYSTEMS, 1997, Grenoble. **Proceedings...** [s. l.].

CHUDAKOV, D. A. **Fundamentos de la teoría y el cálculo de tractores y automóviles**. Moscou: Editorial Mir, 1977. 439p.

COX, Earl. **The fuzzy systems handbook:** a practitioner's guide to building, using, and maintainig fuzzy systems. Boston: Academic Press 1994. 623p.

CROWLEY, James L. Navigation for na intelligent mobile robot. **IEEE Journal of Robotics and Automation**, v.RA-1, n.1, p.31-41, mar./1985.

DAM, Joris W. M. van, KRÖSE, Bem J.A., GROEN, Franciscus C.A. Adaptive sensor models. In: IEEE/SICE/RSJ INT. CONF. ON MULTISENSOR FUSION AND INTEGRATION FOR INTELLIGENT SYSTEMS, 1996, Washington. **Proceedings...** [s. l.]. p.705-712.

DEV, Anuj, KRÖSE, Bem, GROEN, Franciscus. Navigation of a mobile robot on the temporal development of the optic flow. In: IEEE/RSJ/GI INT. CONF. ON INTELLIGENT ROBOTS AND SYSTEMS IROS'97, 1997, Grenoble. **Proceedings...** [s. l.]. p.558-563.

DRIANKOV, Dimiter, HELLENDOORN, Hans, REINFRANK, Michael. **An Intoduction to Fuzzy Control**. 2. ed. Berlin: Springer Verlag, 1996. 316 p.

ERKAMP, Heleen. **A representation in sensor values of na unknown environment trough exploration of a mobile robot**. Amsterdam, 1996. 69p. Dissertação (Mestrado) – Faculty of Mathematics & Computer Science, University of Amsterdam.

FIRBY, R. James. An architecture for a synthetic vacuum cleaner. In: AAAI FALL SYMPOSIUM SERIES WORKSHOP ON INSTANTIATING REAL-WORLD AGENTS, 1993, Raleigh. **Proceedings...** [s. l.].

_____. Architecture, representation and integration: na example from robot navigation. In: AAAI FALL SYMPOSIUM SERIES WORKSHOP ON THE CONTROL OF THE PHYSICAL WORLD BY INTELLIGENT AGENTS, 1994, New Orleans. **Proceedings...** [s. l.].

FROMM, Richard Allan. **High-speed navigation with approximate maps**. Austin, 1995. 113p. Dissertação (Mestrado) – Department of Computer Sciences, University of Texas at Austin.

GATES, Bill. **A estrada do futuro.** São Paulo: Companhia das Letras, 1995. 347p.

GEVARTER, William B. **Artificial Intelligence, Expert Systems, Computer Vision and Natural Language Processing**. New Jersey: Noyes Publications, 1984. 226p.

GHANEA-HERCOCK, Robert, BARNES, David P. **An envolved fuzzy reactive control system for co-operating autonomous robots**. Disponível na Internet. http://www.bioele.nuce.nagoya-w.ac.jp/wsc1/papers/FL_Robots.html. Ago./ 1997.

GIBILISCO, Stan (Edit.). **The McGraw-Hill Illustrated Encyclopedia of Robotics & Artificial Intelligence.** New York: TAB Books, 1994. 420p.

GODJEVAC, Jelena. A learning procedure for a fuzzy system: application to obstacle avoidance. In: THE INTERNATIONAL SYMPOSIUM ON FUZZY LOGIC,1995, Zurich. **Proceedings...** [s. l.].

GROOVER, Mikell P., WEISS, Mitchell, NAGEL, Roger N. et al. **Robótica:** tecnologia e programação. São Paulo: McGraw-Hill, 1988. 401p.

HALLAM, Bridget , HAYES, Gillian. **Comparing robot and animal bahaviour**. Disponível na Internet. file://ftp.dai.ed.ac.uk/pub/papers/dairp598.tar.Z. Ago./1997.

HAYATI, Samad. Microrover research for exploration of mars. In: AIAA – FORUM ON ADVANCED DEVELOPMENTS IN SPACE ROBOTICS,1996, Madison. **Proceedings...** [s. l.].

_____, VOLPE, Richard, BACKES, Paul et. all. The rocky 7 rover: a Mars sciencecraft prototype. In: IEEE INTERNATIONAL CONFERENCE ON ROBOTICS AND AUTOMATION, 1997,Albuquerque. **Proceedings...** [s. l.], v.3, p.2458-2464.

JAHANBIN, M.R., FALLSIDE, F. Path planning using a wave simulation technique in the configuration space. In: GERO, J.S. **Artificial intelligence in engineering:** robotics and process. Amsterdan: Elsiver, 1988.

JONES, Joseph L., FLYNN, Anita M. **Mobile robots:** inspiration to implementation**.** Wellesley: A K Peters, 1993. 349p.

KAC, Eduardo. Telepresence art. In: KRIESCHE, Richard (Edit.). **Teleskulptur.** Graz, Austria: Kulturdata, 1993. p.48-72.

_____. Live from Mars. **Leonardo.** [s. l.], v.31, n.1, p.1-2, 1998.

KHALED, S. Ali., ARKIN, Ronald C. Multiagent teleautonomous behavioral control. In: INTERNATIONAL CONFERENCE ON SIMULATION OF ADAPTATIVE BAHAVIOR, 3, 1994, Brighton. **Proceedings...** [s. l.], p.473-478.

KLIR, George J., YUAN, Bo. **Fuzzy sets and fuzzy logic: theory and applications**. New Jersey: Prentice Hall, 1995. 573p.

KOREN, Yoram. **Robotics for engineers**. New York: McGraw- Hill, 1985. 347p.

LAPRADE, Alain, LAMBERT-TORRES, Germano. Controle de um veículo utilizando a teoria dos conjuntos nebulosos. In: SIMPÓSIO BRASILEIRO DE AUTOMAÇÃO INTELIGENTE, 1, 1993, RioClaro. **Anais...** [s. l.], p.165-174.

LAVERY, Dave. The future of telerobotics. **Robotics world**. summer/1996

LEITE, Alzira, ALVES, João Bosco da Mota, RESENDE, Marcos Ferreira. Sistema para depuração de um robô móvel autônomo inteligente. In: SIMPÓSIO BRASILEIRO DE AUTOMAÇÃO INTELIGENTE, 1, 1993, Rio Claro. **Anais...** p.175-184.

MAIOR, Dagoberto Souto. Os robôs estão chegando, **CiênciaHoje - suplemento technologia**, v.23, n.136, p.3-2, mar.,1998.

MAMDANI, E.H., ASSILIAN, S. An experiment in linguistic synthesis with a fuzzy logic controller. **International Journal of Man-Machine Studies**, v.7, n.1 p.11-13, 1975.

MATTHIES, Larry, GAT, Erann, HARRISON, Reid et al. Mars microrover navigation: performance evaluation and enhancement. **Auton. robots**, v..2, n.4, p.291-311, 1995.

MCCOMB, Gordon. **The robot builder's bonanza:** 99 inexpensive robotics projects. New York: TAB Books, 1987. 326p.

MCFARLAND, David, BÖSSER, Thomas. **Intelligent behavior in animals and robots.** Cambridge: Bradford Book, 1993. 308 p.

MCKERROW, Phillip John. **Introduction to robotics**. Sydney: Addison-Wesley, 1991. 811 p.

MENEZES, Paulo Fernando Blauth. **Linguagens formais e autômatos**. v.3. Porto Alegre: Sagra Luzzatto, 1997. 168p.

MÓDULO Consultoria e Informática. **Linguagem C:** programação e aplicações. 3. ed. Rio de Janeiro: Livros Técnicos e Científicos, 1989. 266p.

MONDADA, Francesco, FLOREANO, Dario. Evolution and mobile robotics. In: SANCHEZ, E., TOMMASINI, M. (Edit.). **Towards evolvable hardware**. Berlin: Springer Verlag, 1996. p.221-249.

_____, FRANZI, Edoardo. Biologically inspired mobile robot control algorithms. In: NFP-PNR SYMPOSIUM, 23, 1993, Zuerich. **Proceedings...** [s. l.].

_____, _____, IENNE, Paolo. Mobile robot miniaturisation: a tool for investigation in control algorithms. In: INTERNATIONAL SYMPOSIUM ON EXPERIMENTAL ROBOTICS, 3, 1993, Kyoto. **Proceedings...** [s. l.], p.501-513.

MORE, C. G., HARRIS, C. J. Aspects of fuzzy control and estimation, in HARRIS, C. J. (Edit). **Advances in Intteligent Control**. London: Taylor & Francis, 1994. p.201-242.

NITZAN, David. Development of intelligent robots: achievements and issues. **IEEE Journal of Robotics and Automation.** v.RA-1, n.1, p.3-13, mar.1985.

O'MALLEY, Chris. Biologgy Computes, **Popular Science**, v.254, n.3, p.60-64, mar.,1999.

PAN, J., PACK, D. J., KOSAKA, A. et al. Fuzzy-nav: a vision-based robot navigation architecture using fuzzy inference for uncertainty-reasoning. In: THE WORLD CONGRESS ON NEURAL NETWORKS, 1995, Washington. **Proceedings...** [s. l.], v.2, p.602-607.

PELL, Barney, BERNARD, Douglas E., CHIEN, Steve A. et al. An implemented architecture integrating onboard planning, scheduling, execution, diagnosis, monitoring and control for autonomous spacecraft. In: SPIE, 1996, **Proceedings...** v.2810, p.74-90

PIAGGIO, Maurizio, SGORBISSA, Antonoio, VERCELLI, Gianni et al. Autonomous robot navigation using a reactive agent. **Lecture notes in artificial intelligence**, Vol. 1321, pp.96-105, 1997

_____, ZACCARIA, Renato. An autonomous system for a vehicle navigating in a partially or totally unknown environment. In: INT. WORKSHOP ON MECHATRONICAL COMPUTER SYSTEMS FOR PERCEPTION AND ACTION, 1997, Pisa. **Proceedings...** [s. l.]

POLONSKII, Mikhail M. **Introdução à robótica e mecatrônica**. Caxias do Sul: EDUCS, 1996. 146p.

POMERLEAU, Dean A. **Neural network perception for mobile robot guidance**. Boston: Kluwer, 1993. 191p.

RICH, Elaine, KNIGHT, Kevin. **Inteligência artificial**. 2. ed. São Paulo: Makron, 1994. 722p.

ROSS, Timothy J. **Fuzzy Logic with Engineering Applications.** New York: McGraw-Hill, 1995. 600p.

SAFFIOTTI, Alessandro. Using fuzzy logic for autonomous robotics: an on-line workshop. **The knowledge engineering review**, Cambridge, v.12, n.1, p.91-94, 1997.

_____. Fuzzy logic in autonomous robotics: behavior coordination. In: IEEE INT. CONF. ON FUZZY SYSTEMS, 6, 1997, Barcelona. **Proceedings...** [s. l.], p.573-578

_____, RUSPINI, Enrique H., KONOLIGE, Kurt. Robust execution of robot plans using fuzzy logic. In: RALESCU, Anca L. (Ed), **Fuzzy logic in artificial intelligence – IJCAI '93 Workshop.** Berlin: Springer-Verlag, 1994. p.24-37

_____, _____, _____. Using Fuzzy Logic for Mobile Robot Control. In: DUBOIS, D., PRADE, H. , ZIMMERMANN, H.J. (Edit.) **Handbook of fuzzy sets and possibility theory**. Boston: Kluwer Academic, 1999.

SCHALKOFF, Robert J. **Artifitial Intelligence:** an engineering aproach. New York: McGraw Hill, 1990. 646p.

SEQUEIRA, João , RIBEIRO, M. Isabel. A behaviour-based kernel architecture for robot control. In: SYMPOSIUM ON ROBOT CONTROL – SYROCO'97, 5, 1997, Nantes. **Proceedings...** [s.l], p.833-838.

SHULTZ, Thomas W. **C and 8051:** programming for multitasking. New Jersey: Prentice Hall, 1993. 366p.

SILVA JÚNIOR, Vidal Pereira da. **Microcontroladores**. São Paulo: Editora Érica, 1988. 187p.

_____. **Aplicações práticas do microcontrolador 8051**. 5. ed. São Paulo: Érica, 1994. 270p.

SIMMONS, Reid, GOODWIN, Richard, ZITA, Karen et al. A layered architecture for office delivery robots. **Autonomous agents**, p.245-252, fev./1997.

SPENCE, Rob, HUTCHINSON, Seth. An integrated architecture for robot motion planning and control in the presence of obstacles with unknown trajectories. **IEEE trans. on systems, man, and cybernetics**, v.25, n.1, p.100-110, jan., 1995.

STOKER, Carol R., BARCH, Donald R., HINE III, Butler P. et. al. Antartic undersea exploration using a robotic submarine with a telepresence user interface. **IEEE Expert**, v.10, n.6, p.14-23, dez., 1995.

TRIVEDI, Mohan Manubhai, ABIDI, Mongi A., EASON, Richard O. et al. Developing robotics systems with multiple sensors. **IEEE Trans. On Systems, Man, and Cybernetic**. v.20, n.6, p.1.285-1.299, nov./dec.,1990.

VIEIRA, Rodrigo de Souza, MODRO, Nilson Ribeiro, MARTINS, Alejandro et al. An approach to multiple inheritance ina fuzzy object modeling. In: EUFIT'98, 1998, Aachen . **Proceedings...** [s. l.], p.765-770.

VIOT, Greg. Fuzzy logic: concepts to constructs. **AI expert**, p.26-33, nov. 1993.

VLACK, Lawrence Hall Van. **Princípio de ciências e tecnologias dos materiais**. 2. ed. Rio de Janeiro: Campus, 1988. 567p.

YEN, John, PFLUGER, Nathan. A fuzzy logic basedextension to Payton and Rosenblatt's Command Fusion Method for mobile robot navigation. **IEEE trans. on systems, man, and cybernetics**, v.25, n.6, p.971-978, jun./1995.

ZADEH, Lotfi A. Fuzzy Sets. **Information and control**, n.8, p.338-353, 1965

_____. Fuzzy Control: A personal perspective by Lotfi A. Zadeh. **Control engineering**, p.51-52, jul./1996.

ZIMMER, Uwe R. Robust world-modelling and navigation in a real world. **Neurocomputing**, v.13, n.2-4,pp.247-260, 1996

Anexo A

Listagem do código em C do programa de controle do RAU

```
/*-----------------------------------------------
□
 Programa de controle do robo RAU, utilizando
 8 regras de inferencia difusa.
□
-----------------------------------------------
   Desenvolvido por Rodrigo de Souza Vieira
        Fpolis, marco 1998/maio de 1999
-----------------------------------------------*/
#include <reg51.h>              /* definicoes para o 8051 */

#define NADA            0       /* modos de operacao do joystick*/
#define ESQUERDA        1
#define DIREITA         2
#define EM_FRENTE       3
#define CAM_BAIXO       4
#define CAM_CIMA        5

char ultimoComando =0;

/* dados das funcoes de pertinencia de saida */
/* A ordem estabelecida eh: Virar_e_muito
                            Virar_e_pouco
                            Virar_d_pouco
                            Virar_d_muito
                            Em_frente        */
static float xbarra[5];    /* Definicao de xbarra (centroide da area)
                           para cada funcao acima */
                        □
static float  alfacut[5];  /* Definicao de alfacut para cada funcao acima
static int Passo_Esquerdo = 0,  /* contador para o passo do motor esq. */
           Passo_Direito = 0,  /* contador para o passo do motor dir. */
           Passo_Camera = 0;   /* contador para o passo do motor da cam.



/* Funcao que envia o sinal para a porta P2 para comandar o passo do
motor de controle da camera */
void MovimentaCamera(void)
{
    char comando = 0;

    if(Passo_Camera>3)
      Passo_Camera = 0;
    if(Passo_Camera<0)
      Passo_Camera = 3;
    switch(Passo_Camera)
      {
      case 0 : comando = 0xA;
               break;
      case 1 : comando = 0x9;
               break;
      case 2 : comando = 0x5;
               break;
      case 3 : comando = 0x6;
               break;
      default: comando=0x0;
```

```
      }
    P2 = comando;
}
/* Funcao que envia o sinal para a porta P1 para
comandar os passos dos motores */
void MovimentarMotores(void)
{
    unsigned int i=0;
    char  comando = 0;
    /* verifica o motor direito*/
    switch(Passo_Direito)
      {
      case 0 : comando = 0xA0;
               break;
      case 1 : comando = 0x90;
               break;
      case 2 : comando = 0x50;
               break;
      case 3 : comando = 0x60;
               break;
      default: comando=0x00;
      }
    /* verifica o motor esquerdo */
    switch(Passo_Esquerdo)
      {
      case 0 : comando = comando | 0x0A;
               break;
      case 1 : comando = comando | 0x09;
               break;
      case 2 : comando = comando | 0x05;
               break;
      case 3 : comando = comando | 0x06;
               break;
      default: comando = comando | 0x00;
      }
    /* envia o sinal para a porta P1*/
    P1 = comando;
    ultimoComando=comando;
    for (i=0;i<1500;i++);   /* equivalente a 0.5 ms */
}



/* Funcao para virar a direita */
void VirarADireita(void)
{
    Passo_Esquerdo ++;
    if (Passo_Esquerdo >3)
      Passo_Esquerdo = 0;
    Passo_Direito --;
    if (Passo_Direito < 0)
      Passo_Direito = 3;
    MovimentarMotores();
    P1 = 0x00;           /* zera a porta*/
}
```

```
/* Funcao para virar a esquerda */
void VirarAEsquerda(void)
{
    Passo_Esquerdo --;
    if (Passo_Esquerdo <0)
      Passo_Esquerdo = 3;
    Passo_Direito ++;
    if (Passo_Direito > 3)
      Passo_Direito = 0;
    MovimentarMotores();
    P1=0x00;           /* zera a porta*/
}



/* Funcao para seguir em frente */
void EmFrente(void)
{
    Passo_Esquerdo ++;
    if (Passo_Esquerdo >3)
      Passo_Esquerdo = 0;
    Passo_Direito ++;
    if (Passo_Direito > 3)
      Passo_Direito = 0;
    MovimentarMotores();
    P1=0x00;           /* zera a porta*/
}



/* funcao que converte entrada digital em analogica */
float binToFloat(int result)
{
    float valor= 0;

    switch(result)
      {
      case 0  : break;
      case 1  : valor = 0.714;
                break;
      case 10 : valor = 1.428;
                break;
      case 11 : valor = 2.142;
                break;
      case 100: valor = 2.856;
                break;
      case 101: valor = 3.570;
                break;
      case 110: valor = 4.284;
                break;
      case 111: valor = 5.;
                break;
      }
    return valor;
}
```

```
/* Funcao de leitura do Joystick */
char LeJoystick(void)
{
    char c;
    unsigned char valor= 0;

    if(P27)
      valor++;                        /* soma o primeiro valor ##x */
    if(P26)
      valor+=10;                      /* soma o segundo valor #1# */
    if(P25)
      valor +=100;                    /* soma o terceiro valor 1## */
    switch (valor)
      {
      case  0 : c = NADA;
                break;
      case  1 : c = ESQUERDA;
                break;
      case  11: c = CAM_BAIXO;
                break;
      case 100: c = DIREITA;
                break;
      case 110: c = CAM_CIMA;
                break;
      case 111: c = EM_FRENTE;
                break;
      default: c= NADA;
      }
    return (c);            /* retorna valor */
}



/* funcao viraRobo - permite virar em uma direcao ou outra */
void viraRobo(float angulo,char Joystick)
{
    float contador =0;

    switch (Joystick)
      {
      case NADA    : P1 = ultimoComando;
                    /* se o joystick nao mandou nenhum comando */
                     break; /* nao manda sinal para porta
                               de controle do motor */
      case ESQUERDA: VirarAEsquerda();   /* se manda para a esquerda */
                     P1=ultimoComando;  /* freia */
                     break;
      case DIREITA : VirarADireita();   /* se manda para a direita */
                     P1=ultimoComando;  /* freia */
                     break;
      case EM_FRENTE:if((angulo>-1) && (angulo<1))
                        EmFrente();
                     else if (angulo <0 ) /* sensor manda para a esquerda
                       {
                       contador = 0;        /* zera contador */
                       while(contador > angulo)
                         {
```

```
                        VirarAEsquerda();
                        contador -=0.63;       /* decrementa os graus */
                        }
                     }
                   else if (angulo >0 )  /* sensor manda para a direita
                     {
                     contador = 0;               /* zera contador */
                     while(contador < angulo)
                        {
                        VirarADireita();
                        contador +=0.63;       /* incrementa graus */
                        }
                     }
                   P1=ultimoComando;  /* teste */
                   break;
      case CAM_CIMA : Passo_Camera++; /* se o joystick mandou virar a
                                        camera para cima */
                     MovimentaCamera();
                     break;
      case CAM_BAIXO: Passo_Camera--; /* se o joystick mandou virar a
                                         camera para baixo */
                     MovimentaCamera();
                     break;
      default: P1 = ultimoComando;
      }
}



/* define a funcao minimo */
float min(float x1, float x2)
{
    if(x1 <x2)
      return x1;
    return x2;
}



/* definicao da funcao Perto Obs.: Longe = 1 - Perto;
   ou seja, nao necessita implementacao */
float perto(float x)
{
    return(x);       /* equacao da reta (utilizar x/5 no final)*/
}



/* funcao de leitura dos sensores */
float LeSensor(int Sensor)
{
    char auxiliar = 0;      /* auxiliar para a mascara de leitura da P0*/
    unsigned char valor = 0;

    switch(Sensor)
      {
      case 1: auxiliar = P0 & 0x07;             /* usa mascara AND */
```

```
        if(auxiliar == 0x01)              /* 001 */
           valor=1;
        else if(auxiliar == 0x02)         /* 010 */
           valor=10;
        else if(auxiliar == 0x03)         /* 011 */
           valor=11;
        else if(auxiliar == 0x04)         /* 100 */
           valor=100;
        else if(auxiliar == 0x05)         /* 101 */
           valor=101;
        else if(auxiliar == 0x06)         /* 110 */
           valor=110;
        else if(auxiliar == 0x07)         /* 111 */
           valor=111;
        else                              /* 000 */
           valor =0;
        return(binToFloat(valor));
case 2: auxiliar = P0 & 0x38;
        if(auxiliar == 0x08)              /* 001 */
           valor=1;
        else if(auxiliar == 0x10)         /* 010 */
           valor=10;
        else if(auxiliar == 0x18)         /* 011 */
           valor=11;
        else if(auxiliar == 0x20)         /* 100 */
           valor=100;
        else if(auxiliar == 0x28)         /* 101 */
           valor=101;
        else if(auxiliar == 0x30)         /* 110 */
           valor=110;
        else if(auxiliar == 0x38)         /* 111 */
           valor=111;
        else                              /* 000 */
           valor =0;
        return(binToFloat(valor));
case 3: auxiliar = P0 & 0xC0;
        if(auxiliar == 0x40)              /* 001 */
           valor=1;
        else if(auxiliar == 0x80)         /* 010 */
           valor=10;
        else if(auxiliar == 0xC0)         /* 011 */
           valor=11;
        else                              /* 000 */
           valor=0;
        if(P30)
           valor+=100;                     /* 1## */
        return(binToFloat(valor));
case 7: if(P31)
           valor++;
        if(P32)
           valor+=10;
        if(P33)
           valor+=100;
        return(binToFloat(valor));
case 8: if(P34)
           valor++;
        if(P35)
```

```
                valor+=10;
             if(P36)
                valor+=100;
             return(binToFloat(valor));
      }
   return(0);
}



void main(void)
{
   /* declara variaveis de defusificacao */
   float AreaTotal,
   Centro;
   char i;   /* contador geral */
   /*declara variaveis: Variaveis linguisticas */
   float  dir_1_perto,
          dir_2_perto,
          dir_3_perto,
          dir_7_perto,
          dir_8_perto;

   IE = 0;                        /* desabilita todas as interrupcoes */
   /* cria dados: seleciona os centroides */
   xbarra[0] = -90;
   xbarra[1] = -45;
   xbarra[2] = 45;
   xbarra[3] = 90;
   xbarra[4] = 0;
   /* Zera as portas paralelas de comunica‡Æo */
   P0 = 0x00;
   P1 = 0x00;
   P2 = 0x00;
   P3 = 0x00;
   while(1)
     {
     AreaTotal =0;      /* Zera area total */
     Centro = 0;
     /* le dados de entrada */
     /*  le vetor na forma x1,x2,x3,x7,x8 */
     /* calcula os valores de Longe e Perto; */
     dir_1_perto = perto(LeSensor(1));
     dir_2_perto = perto(LeSensor(2));
     dir_3_perto = perto(LeSensor(3));
     dir_7_perto = perto(LeSensor(7));
     dir_8_perto = perto(LeSensor(8));
     /* avalia regras */
     /* atencao, por motivo de memoria, os valores para longe sao dados
     por 1 - perto*/
     alfacut[0] =
       min(dir_1_perto,min(dir_2_perto,min(dir_3_perto,min(1-
       dir_7_perto,1-dir_8_perto))));  /* virar_e_muito*/
     if(alfacut[0] <dir_1_perto)
        alfacut[0] = dir_1_perto;        /* sexta regra */
     alfacut[1] =
       min(dir_1_perto,min(dir_2_perto,min(1-dir_3_perto,min(1-
```

```
        dir_7_perto,1-dir_8_perto)))); /* virar_e_pouco*/
      if(alfacut[1] <dir_2_perto)
        alfacut[1] = dir_2_perto;        /* setima regra */
      alfacut[2] =
        min(dir_1_perto,min(1-dir_2_perto,min(1-dir_3_perto,min(1-
        dir_7_perto,dir_8_perto))));  /* virar_d_pouco*/
      if(alfacut[2] <dir_8_perto)
        alfacut[2] = dir_8_perto;        /* oitava regra */
      alfacut[3] =
        min(dir_1_perto,min(1-dir_2_perto,min(1-
        dir_3_perto,min(dir_7_perto,dir_8_perto))));  /* virar_d_muito */
      alfacut[4] =
        min(1-dir_1_perto,min(1-dir_2_perto,min(
        dir_3_perto,min(dir_7_perto,1-dir_8_perto))));  /* em_frente */
      /* defusifica : calcula pela formula (alfa*base*(3*h-alfa))/4h */
      for(i=0;i<5;i++)
        AreaTotal += alfacut[i]*90*(3-alfacut[i])/4;
        /* calculo do centroide */
      if (AreaTotal==0)
        AreaTotal =1;
      for(i=0;i<5;i++)
        Centro += xbarra[i]*alfacut[i]*90*(3-alfacut[i])/4;
      Centro = Centro/AreaTotal;
      viraRobo(Centro,LeJoystick());
      }   /* fim do while */
}
```

Anexo B

Fotos da montagem do robô RAU

A seguir, algumas fotos da montagem e dos componentes utilizados na confecção do robô RAU.

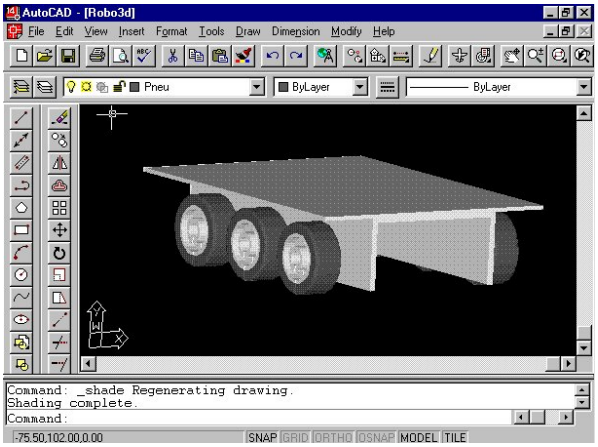


Fig. B.1 – Projeto do chassi no AutoCAD R.14

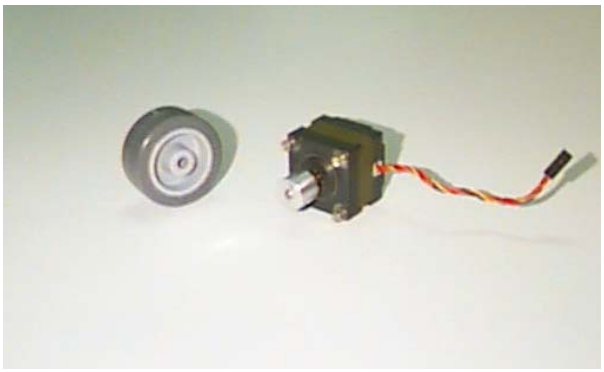


Fig B.2 – Roda e motor de passo do RAU

Fig B.3 – Roda, motor de passo e bucha de montagem

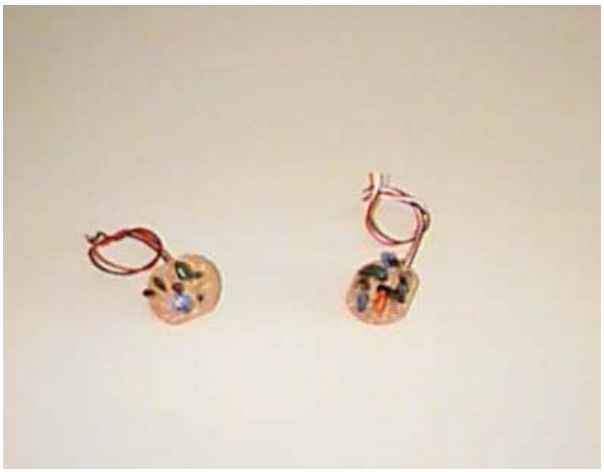


Fig B.4 – Transmissor e receptor infravermelho sem encapsulamento

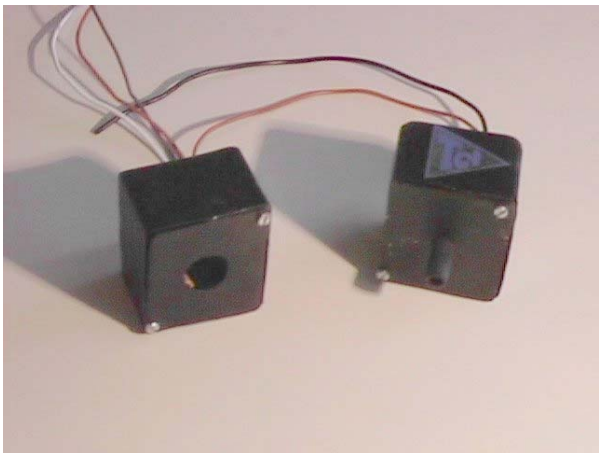


Fig. B.5 – Transmissor e receptor infravermelho com encapsulamento

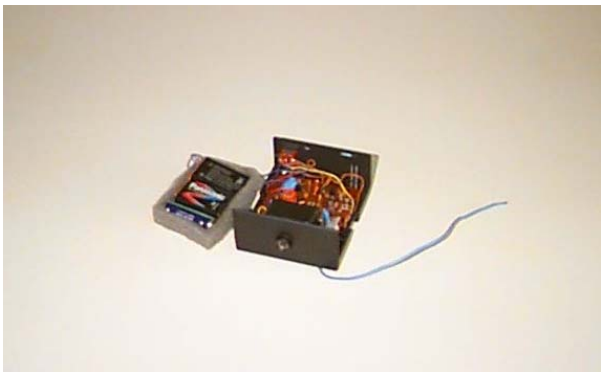


Fig. B.6 – *Kit* original de envio de sinal de vídeo

Fig. B.7 – *Kit* modificado de envio de sinal de vídeo

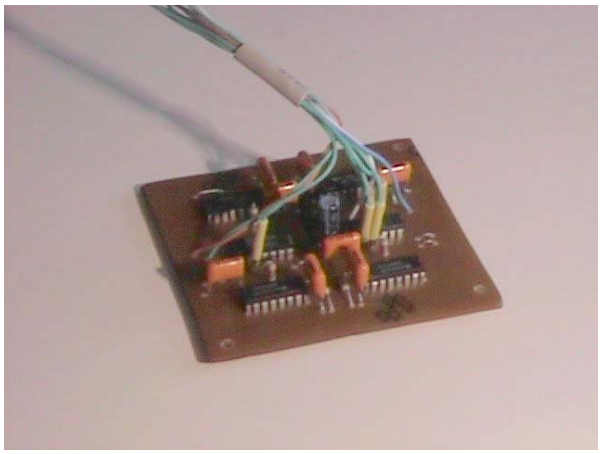


Fig. B.8 – Placa original do *kit* do decoder

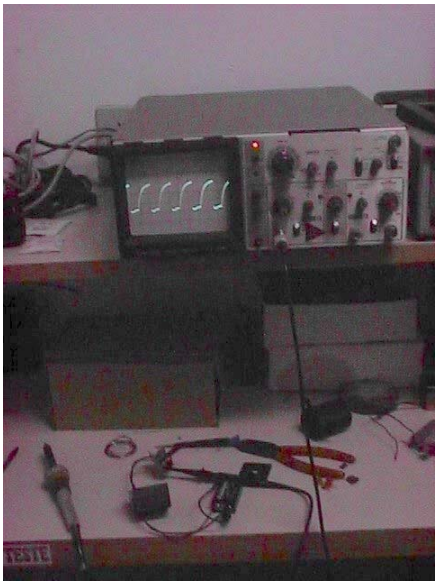


Fig B.9 –Verificação do sinal do emissor de infravermelho

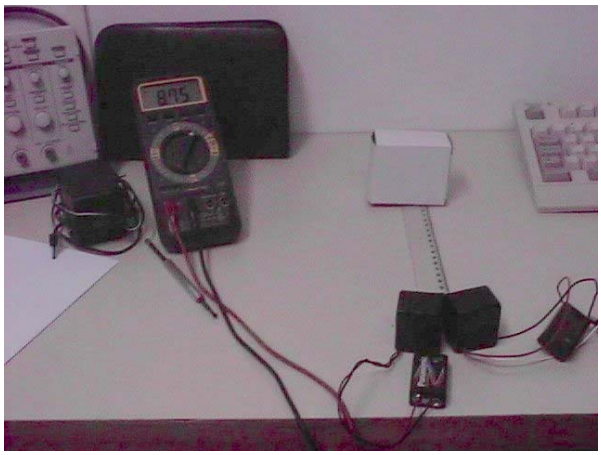


Fig. B.10 – Teste de sensibilidade dos sensores infravermelhos

Fig. B.11 – Levantamento da sensibilidade dos sensores infravermelhos

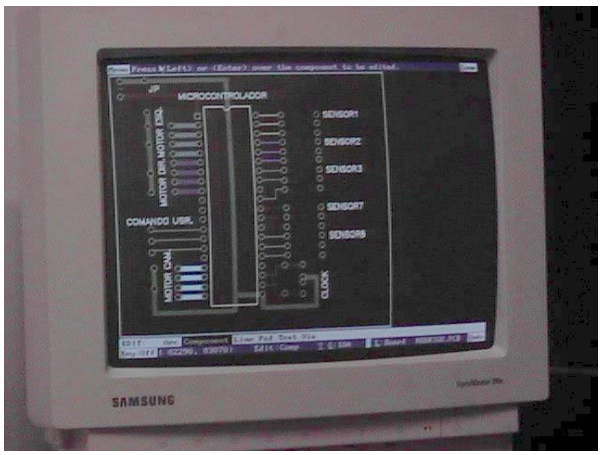


Fig. B.12 – Projeto da placa controladora do RAU

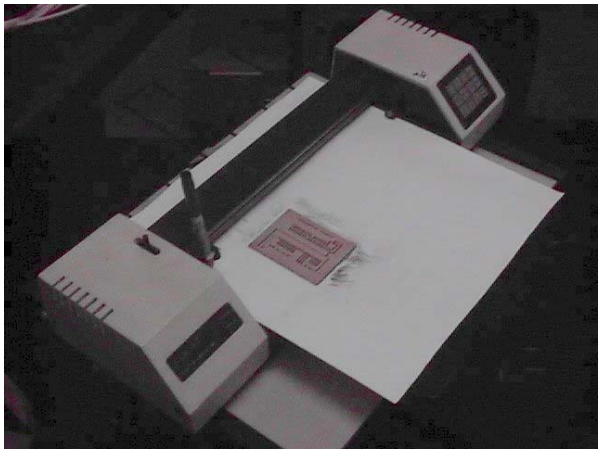


Fig. B.13 – Confecção da placa controladora do RAU

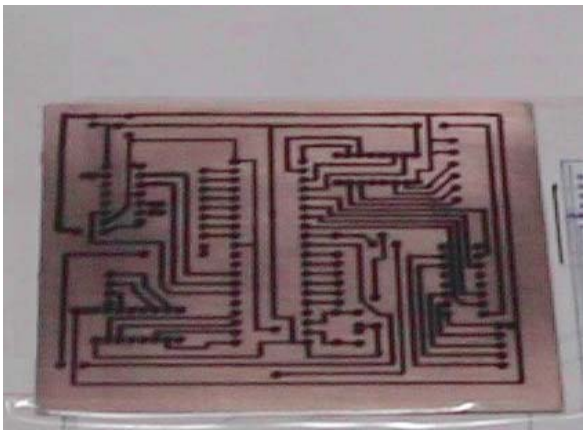


Fig. B.14 – Placa controladora do RAU traçada

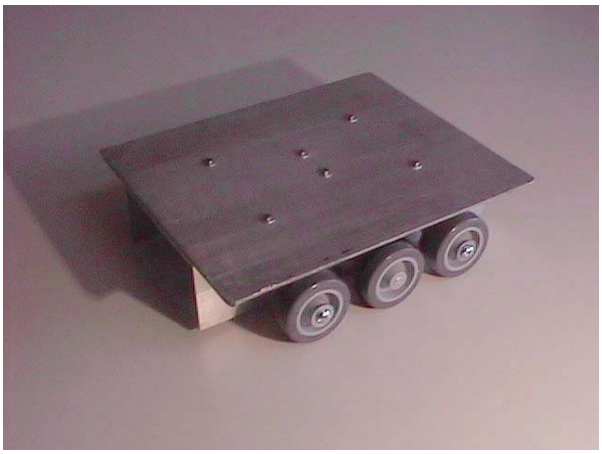


Fig. B.15 – O chassi do RAU montado

Fig. B.16 – Programação do microcontrolador

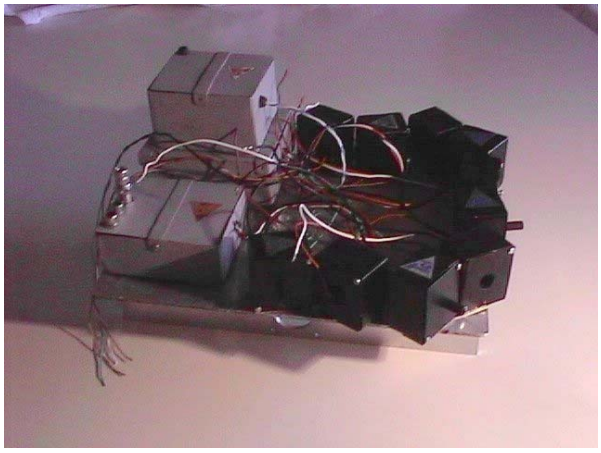


Fig. B.17 – Chassi do RAU com sensores, decoder e kit de envio de sinal de vídeo

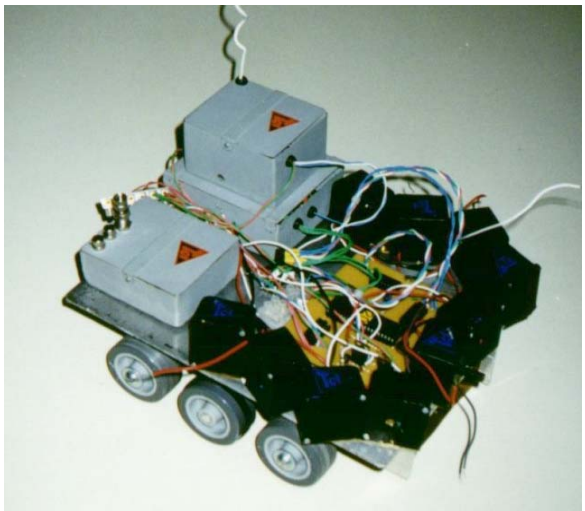


Fig. B.18 – O RAU montado, sem a câmera e conversor A/D

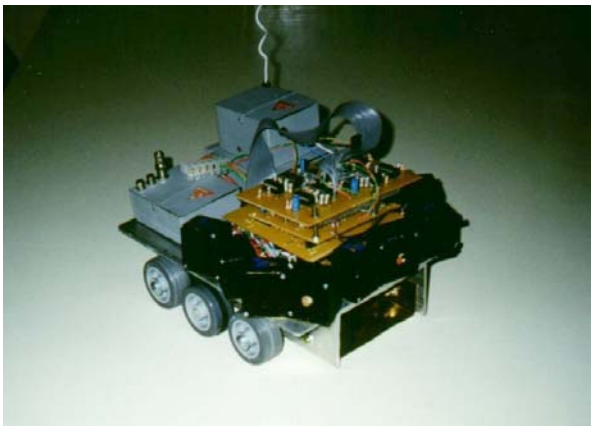


Fig. B.19 – O RAU com o conversor A/D

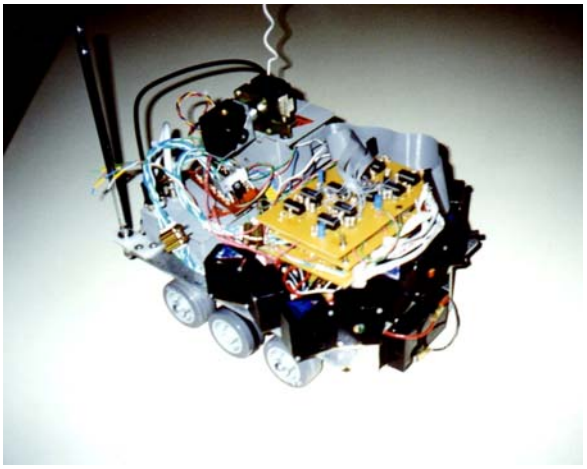


Fig. B.20 – O RAU montado

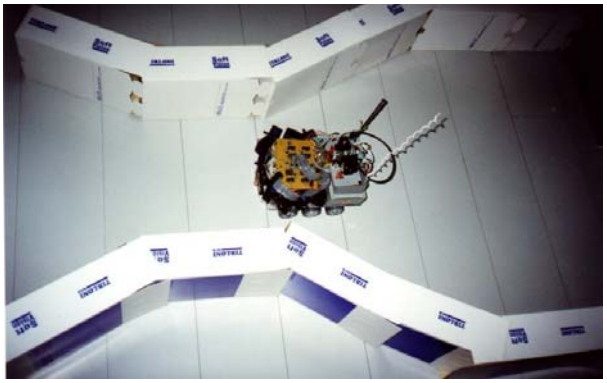


Fig. B.21 – O RAU no teste do corredor com afunilamento

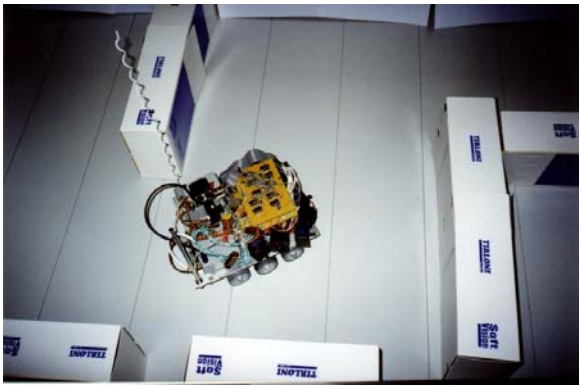


Fig. B.22 – O RAU no teste de obstáculos estáticos

Anexo C

Versão do controlador baseada no MATLAB 4.2.C

A utilização do MATLAB como ferramenta para desenvolvimento de protótipos para o controlador difuso utilizado no robô RAU se deu em três etapas básicas. Primeiramente foi desenvolvido um controlador difuso, utilizando o FIS – *Fuzzy Inference System* da *Toolbox Fuzzy* do MATLAB 4.2.C. Este sistema permite escrever controladores difusos com elevada rapidez e confiabilidade.

São os parâmetros iniciais do FIS o método de inferência e de defusificação, as variáveis de entrada e saída e seus valores. De acordo com o sistema desenvolvido, através da figura C.1 pode-se perceber as variáveis de entrada dadas por **dir_1**, **dir_2**, **dir_3**, **dir_7** e **dir_8** e a variável de saída **direção**. Outros valores podem também ser mudados como as normas e t-conormas.

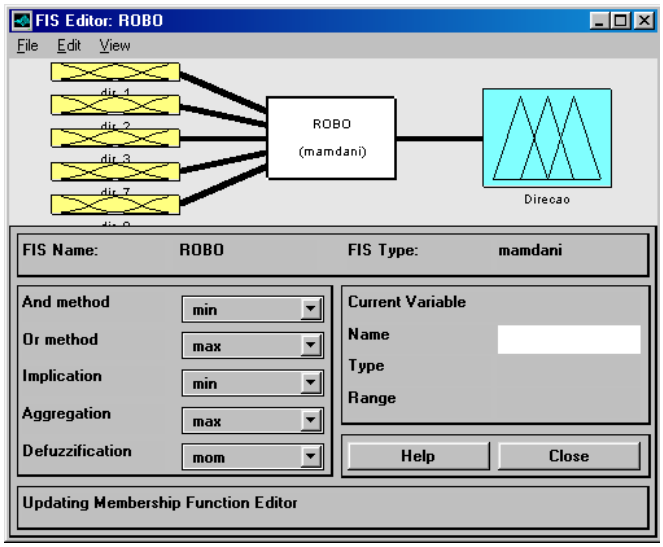


Fig. C.1 – Controlador difuso do RAU escrito no MATLAB

Definidos os nomes das variáveis envolvidas, o passo seguinte é a definição dos valores lingüísticos que podem ser atribuídos às variáveis. Na figura C.2 pode-se ver os valores **longe** e **perto**, referentes as variáveis de entrada do controlador. Deve-se lembrar que, conforme descrito no capítulo 5 as 5 variáveis de entrada têm os mesmos valores lingüísticos.

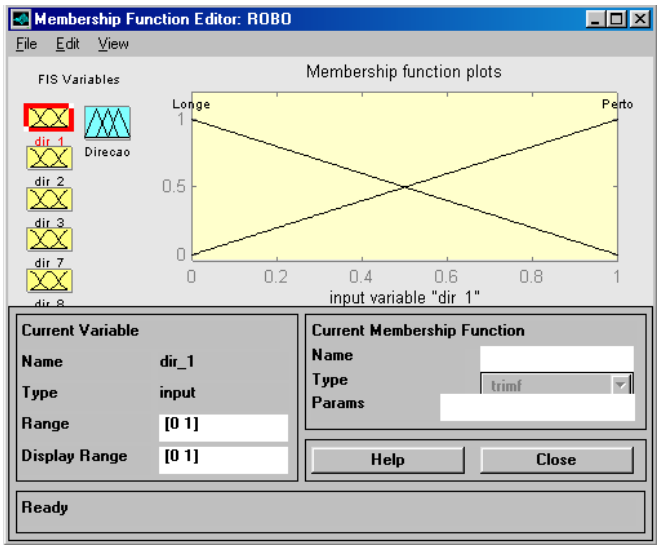


Fig. C.2 – Valores lingüísticos das variáveis de entrada

A figura C.3 apresenta os valores lingüísticos da variável direção, dados por **virar_e_muito**, **virar_e_pouco**, **em_frente**, **virar_d_pouco** e **virar_d_muito**. Pode-se observar o overlap entre os conjuntos e a definição de seus suportes, conforme descrito no capítulo 5.

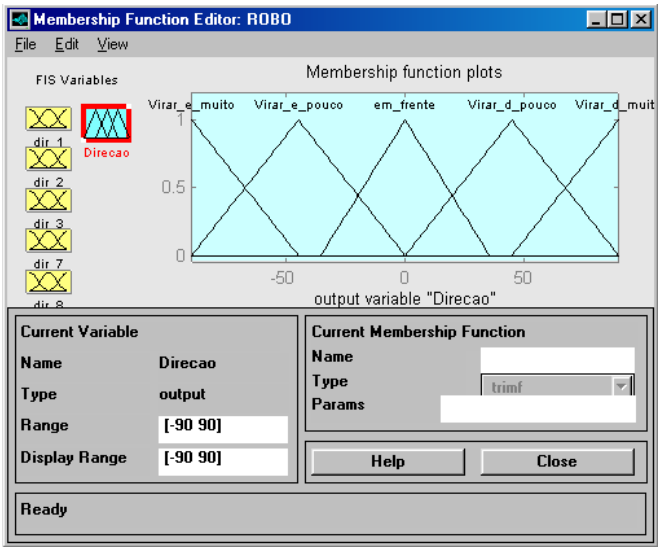


Fig. C.3 – Valores lingüísticos da variável de saída

Um ponto importante com relação às variáveis é que o FIS não permite mais do que 5 variáveis de entrada, sendo esta uma importante limitação a ser observada quando se pensar em utilizar esta ferramenta.

Na seqüência do desenvolvimento do controlador difuso no MATLAB, deve-se definir o conjunto de regras que relacionam entradas e saídas.

O conjunto de regras final utilizado no controlador foi o mostrado na tabela C.1.

N.º da regra	Regra
01	If (dir_1 is Perto) and (dir_2 is Perto) and (dir_3 is Perto) and (dir_7 is Longe) and (dir_8 is Longe) then (Direcao is Virar_e_muito)
02	If (dir_1 is Perto) and (dir_2 is Perto) and (dir_3 is Longe) and (dir_7 is Longe) and (dir_8 is Longe) then (Direcao is Virar_e_pouco)
03	If (dir_1 is Perto) and (dir_2 is Longe) and (dir_3 is Longe) and (dir_7 is Longe) and (dir_8 is Perto) then (Direcao is Virar_d_pouco)
04	If (dir_1 is Perto) and (dir_2 is Longe) and (dir_3 is Longe) and (dir_7 is Perto) and (dir_8 is Perto) then (Direcao is Virar_d_muito)
05	If (dir_1 is Longe) and (dir_2 is Longe) and (dir_3 is Perto) and (dir_7 is Perto) and (dir_8 is Longe) then (Direcao is em_frente)
06	If (dir_1 is Perto) then (Direcao is Virar_e_muito)

Tab. C.1 – Conjunto de regras utilizadas no controlador

Estas regras são inseridas no programa através de texto, pela janela mostrada na figura C.4

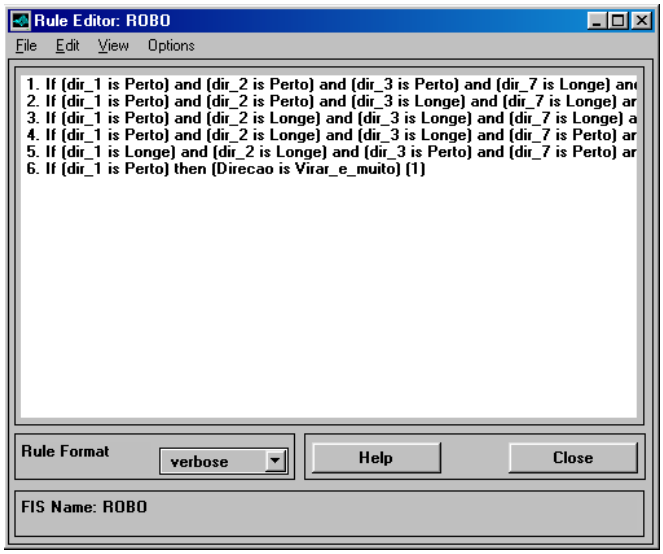


Fig. C.4 – Conjunto de regras escritas no MATLAB

Com o conjunto de regras, as variáveis e valores lingüísticos de entrada e saída e o modelo de inferência determinados, já é possível testar o controlador. Uma das

formas é fornecer valores de entrada através do visualizador de regras (Rule Viewer), mostrado na figura C.5.

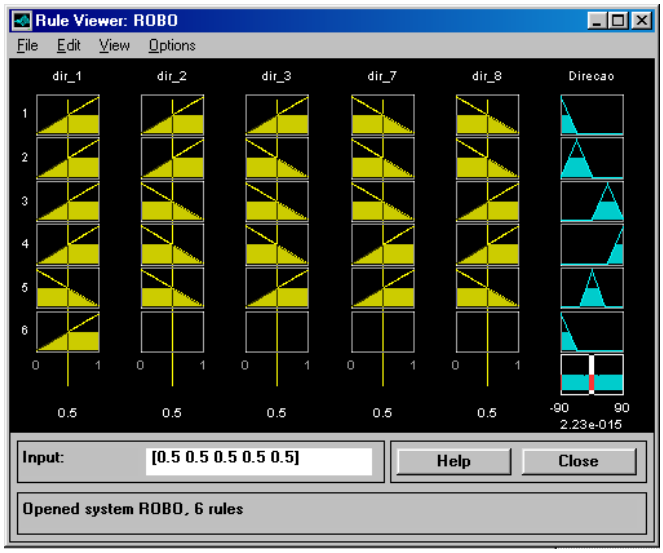


Fig. C.5 – Visualizador de regras

Através do mouse – arrastando as linhas verticais que representam os valores de entrada – ou mesmo pelo teclado - indicando a *n-upla* de entrada - é possível avaliar o comportamento do controlador. O visualizador apresenta, em forma gráfica, linhas que representam as regras e colunas que representam as variáveis. As primeiras cinco colunas à esquerda são referentes as variáveis de entrada, enquanto a última representa a de saída.

Na parte inferior direita da tela se tem o conjunto final da inferência de Mamdani e o seu valor numérico, fornecido pelo modelo de defusificação selecionado na primeira tela.

Toda a informação referente ao controlador difuso é armazenada em forma de matriz, chamada de *matriz fis*.

A segunda etapa do uso do MATLAB foi utilizando o SIMULINK. Este módulo permite construir sistemas dinâmicos dentro do programa, de forma a testar, por exemplo, um controlador difuso.

A construção do modelo se dá através da inclusão de blocos, que compreendem desde fontes geradoras de sinal até controladores PID e redes neuronais.

O SIMULINK foi utilizado neste trabalho com o objetivo de verificar a característica de invariável do sistema com relação a forma apresentada pelos conjuntos difusos. Assim, montou-se um diagrama de blocos como mostrado na figura C.6.

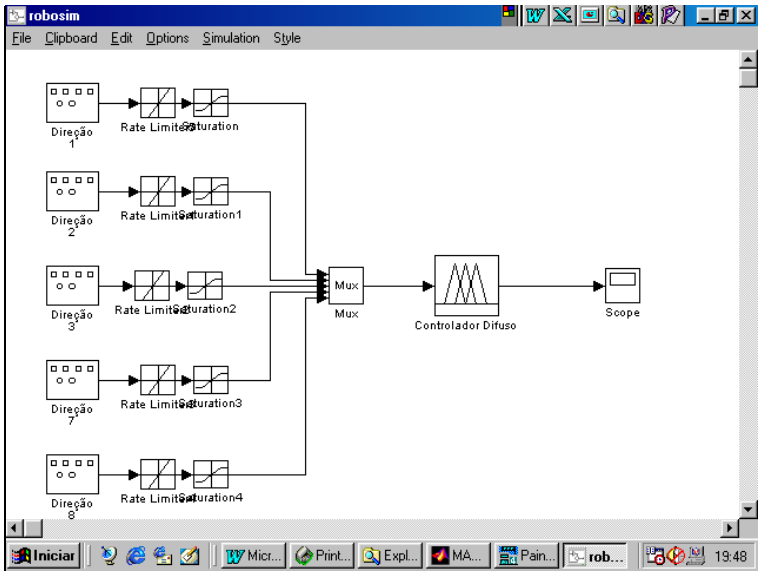
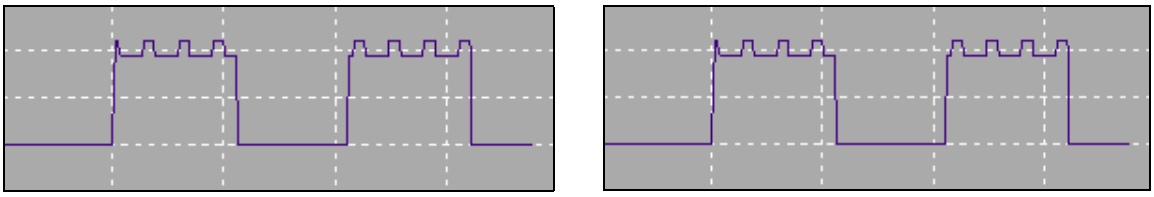


Fig. C.6 – Diagrama de blocos do modelo no SIMULINK

Os blocos à esquerda da figura são responsáveis pela geração de um sinal simulando a variação do valor de cada variável de entrada do sistema. Estes cinco valores são colocados em um vetor que é passado ao controlador difuso escrito no FIS. Para verificar o valor de saída do controlador foi conectado um *scope*.

Através deste modelo pôde-se verificar que com a mudança da forma dos conjuntos difusos de forma de sino para triangular o sinal de saída, mostrado na figura C.7 não se alterou.



(a) *(b)*

Fig. C.7 – Valores de saída para (a) forma de sino e para (b) forma triangular

A terceira e última etapa do uso do MATLAB foi referente ao teste com as 2000 n-uplas descrito na seção 5.3.4.

Foi criado o script mostrado abaixo:

```
% testa o conjunto de regras
% desenvolvido por Rodrigo de S. Vieira
% carrega o conjunto de dados
load entrada.mat
% rearranja os dados de entrada
sensores = sensores(:,1:5)
%modifica o formato de saída
format short
% carrega a matriz fis e calcula a saída
a = readfis('robo')
grau = evalfis(sensores,a)
% grava saída
save saida.dat sensores grau /ascii /tabs
```

O script perfaz as seguintes operações: primeiramente carrega o arquivo **entrada.mat** com valores aleatórioste tensão para os cinco sensores, gerados pelo Excel. Estes valores têm o mesmo significado dos sinais gerados no SIMULINK, representando valores das variáveis de entrada. Na seqüência, eles são avaliados pela *matriz fis* gerada pelo *Fuzzy Inference System*, listado anteriormente, que resulta em valores de saída para cada *n-upla* de entrada. Estes são, por fim, gravados em um arquivo chamado **saida.dat,** que foi posteriormente analisado no Excel, onde se chegou aos valores de erro e aos gráficos mostrados na seção 5.3.4