Universidade Federal do Espírito Santo
Centro Tecnológico
Programa de Pós-Graduação em Engenharia Elétrica

Navegação e Desvio de Obstáculos Usando um Robô Móvel Dotado de Sensor de Varredura Laser

Flávio Garcia Pereira

Vitória

Junho de 2006

Flávio Garcia Pereira

Navegação e Desvio de Obstáculos Usando um Robô Móvel Dotado de Sensor de Varredura Laser

Universidade Federal do Espírito Santo
Centro Tecnológico
Programa de Pós-Graduação em Engenharia Elétrica

Vitória

Junho de 2006

.

Dados Internacionais de Catalogação-na-publicação (CIP)
(Biblioteca Central da Universidade Federal do Espírito Santo, ES, Brasil)

Pereira, Flávio Garcia, 1981-

P436n Navegação e desvio de obstáculos usando um robô móvel dotado de sensor de varredura laser / Flávio Garcia Pereira. - 2006.

95 f. : il.

Orientador: Mário Sarcinelli Filho.

Dissertação (mestrado) - Universidade Federal do Espírito Santo, Centro Tecnológico.

1. Lasers. 2. Distâncias - medição. 3. Navegação de robôs móveis. 4. Desvio de obstáculos. I Sarcinelli Filho, Mário. II. Universidade Federal do Espírito Santo. Centro Tecnológico. III. Título.

CDU: 621.3

.

Flávio Garcia Pereira

Navegação e Desvio de Obstáculos Usando um Robô Móvel Dotado de Sensor de Varredura Laser

Dissertação submetida ao Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal do Espírito Santo como requisito parcial para a obtenção do grau de Mestre em Engenharia Elétrica – Automação.

Aprovada em 23 de junho de 2006.

Comissão Examinadora:

Prof. Dr. Mário Sarcinelli Filho
Universidade Federal do Espírito Santo, Orientador

Prof. Dr. Teodiano Freire Bastos Filho
Universidade Federal do Espírito Santo

Prof. Dr. Eduardo Oliveira Freire
Universidade Federal de Sergipe

Vitória, junho de 2006.

Dedicatória

Aos meus pais e à minha irmã, aos quais jamais deixarei de agradecer por todo carinho e apoio.

Agradecimentos

Aos meus pais, Jorge e Rosangela, os pilares da minha vida e maiores incentivadores, pelo amor, pelo carinho e pelo apoio dados durante essa longa caminhada. Obrigado também pela força para que eu pudesse subir mais esse degrau.

À minha irmã, Flávia pela compreensão, em todas as vezes que, praticamente a expulsei do computador para dar continuidade a esta Dissertação.

À minha namorada Mariana, companheira e amiga, por toda atenção, amor, carinho e apoio que a mim foram dados e que foram muito importantes para a realização deste trabalho.

Ao Professor Dr. Mário Sarcinelli Filho ("*My Profé*ssor"), orientador desta dissertação, pela "força", e pelo incentivo para continuar trabalhando com robótica móvel.

À Professora Dr^{a} e amiga Raquel Frizera Vassallo. Pela disponibilidade e atenção prestadas. Obrigado também pela ajuda nos momentos difíceis. Também pelos conselhos e inúmeras sugestões que foram extremamente importantes para a realização deste trabalho.

A todos os outros professores do Programa de Pós-Graduação, responsáveis pela minha formação.

Ao professor Dr. Ing. Ricardo Carelli, que me recebeu de braços abertos no INAUT durante minha estada em San Juan - Argentina. Obrigado, também, pelos sábios conselhos e conhecimentos transmitidos.

Aos amigos da UFES e do INAUT, em especial ao Christiano, ao André, ao Rodolfo (Pitito) e à Sandra.

Aos meus familiares (são muitos), que, além de me apoiarem, entenderam os motivos das ausências.

Ao CNPq pela bolsa de estudo concedida, sem a qual a realização deste trabalho teria sido mais difícil.

À fundação CAPES que possibilitou a minha estada em San Juan - Argentina.

A todos aqueles que se fossem citados, ocupariam um capítulo, ou mais, desta dissertação.

A todos vocês, MUITO OBRIGADO!

Flávio Garcia Pereira

"O valor das coisas não está no tempo que elas duram, mas na intensidade com que acontecem. Por isso, existem momentos inesquecíveis, coisas inexplicáveis e pessoas incomparáveis..."

Fernando Pessoa

Lista de Tabelas

Lista de Figuras

Sumário

Resumo

Este trabalho trata da navegação de robôs móveis em ambientes semi-estruturados. Para isso, foram desenvolvidos controladores baseados na abordagem reativa. Os controladores implementados permitem que o robô navegue de forma segura entre dois pontos e por corredores, evitando os obstáculos que surgirem no seu caminho. São apresentados dois métodos de desvio de obstáculos, um tangencial e outro não tangencial. As informações sobre o ambiente de trabalho do robô são proporcionadas por um sensor de varredura *laser* instalado a bordo do robô móvel utilizado, o PIONEER 2–DX, da ActivMedia. Os controladores desenvolvidos são validados através de simulações e experimentos reais, sempre utilizando o robô móvel equipado com o sensor *laser* de varredura. Boa parte dos controladores aqui apresentados já foram apresentados em outros trabalhos, porém utilizando outros sensores para propiciar ao sistema de controle as informações sobre o ambiente. Neste sentido, a contribuição deste trabalho é explorar as vantagens de sua utilização com o sensor *laser* de varredura.

Abstract

This work deals with robot navigation in semi-structured environments. Some controllers were designed and implemented, based on reactive behaviors, which assure a safe navigation for a mobile robot when seeking for a goal or following a corridor. While navigating, the robot is also able to avoid obstacles that appear in its path. Two approaches for obstacle avoidance are addressed: a tangential and a non-tangential method. The information about the environment surrounding the robot is collected through a *laser* scanner installed onboard a PIONEER 2-DX mobile platform. The proposed controllers are validated through simulation and experimentations, always using the mobile robot equipped with the *laser* scanner. Some of the addressed controllers have been previously proposed elsewhere, but using other sensorial apparatus. The main contribution of this work is to explore the advantages of using these controllers in connection with the *laser* scanner besides some changes on the controllers designs.

1 Introdução

"O segredo de progredir é começar. O segredo de começar é dividir as tarefas árduas e complicadas em tarefas pequenas e fáceis de executar, e depois começar pela primeira".

Mark Twain

A robótica é uma área do saber que conjuga conhecimentos de engenharia mecânica, com o estudo da estática e a dinâmica, de matemática, com a descrição dos modelos dos robôs e do movimento no espaço, de engenharia eletrônica, com o projeto de sensores e interfaces para os robôs, de teoria de controle, com o projeto de algoritmos para que o robô realize os movimentos desejados, e, ainda, da informática, com a programação de todos os algoritmos desenvolvidos para que a tarefa designada para o robô possa ser realizada.

A maioria das pessoas acha que os robôs são somente máquinas semelhantes aos seres humanos (andróides). Essa idéia está altamente relacionada com os filmes de ficção científica ("*Star Wars*", por exemplo), onde os humanóides são vistos realizando tarefas bastante complexas e, até mesmo, interagindo com o homem de forma inteligente. Atualmente, já existem humanóides, entretanto, o seu nível de desempenho e as suas capacidades ainda são muito limitadas. A Figura 1 mostra o Asimo, um robô humanóide fabricado pela Honda.

Figura 1: Robô Humanóide.

Apesar dos robôs atuais ainda não terem atingido tal nível de autonomia e interatividade, estas idéias representam uma motivação para os trabalhos de robótica, que cada vez mais procuram fazer dos robôs equipamentos mais autônomos, inteligentes, interativos e capazes de se adaptar a mudanças.

A robótica pode ser dividida em duas grandes áreas: robótica industrial, ou de manipulação, e robótica móvel.

Dentro da primeira área pode-se incluir a robótica a nível industrial, onde se destacam os robôs manipuladores, que foram os primeiros robôs a serem fabricados, ou alguns robôs usados na medicina para auxílio em cirurgias. A Figura 2 mostra um robô manipulador do tipo SCARA, fabricado pela Toshiba.

Figura 2: Robô Manipulador.

Já na segunda área, a robótica móvel, incluem-se todos os robôs com capacidade de locomoção como, por exemplo, os robôs usados no futebol de robôs, os humanóides, os robôs usados em operações de inspeção, usados, por exemplo, para inspecionar zonas de reduzida dimensão, zonas perigosas, ou mesmo para exploração de outros planetas. Um exemplo de robô móvel, produzido pela ActivMedia, pode ser visto na Figura 3.

Um dos objetivos da robótica móvel é a criação de robôs autônomos. A robótica móvel enfatiza os problemas relacionados à locomoção dos robôs em ambientes complexos, que se modificam dinamicamente. Um robô autônomo deve ser capaz de navegar nestes ambientes sem que haja a intervenção humana, ou seja, deve adquirir e utilizar o conhecimento sobre o ambiente, ter a habilidade de detectar obstáculos e agir de tal forma que possa concluir a tarefa que lhe foi designada [1]. O usuário apenas determina a tarefa a ser realizada e não a maneira como o robô deve agir para que a mesma seja concluída. Por apresentarem estas características, esses robôs são considerados dotados

Figura 3: Robô Móvel.

de certa "inteligência" e dependentes de técnicas de controle e de sensores. Essa interação que o robô deve ter com o ambiente de trabalho é conseguida através dos vários sensores instalados a bordo do veículo.

Sempre que um robô navega em um ambiente não-estruturado, é necessário assegurar que ele não irá colidir com obstáculos. Logo, na maioria das aplicações em robótica móvel é necessário um método eficiente de desvio de obstáculos quando o robô é posto para navegar. Esses métodos, de acordo com a abordagem adotada, podem ser classificados como *deliberativos* e *reativos*. Dentre eles podem ser citados o Método de Detecção de Bordas [2], Grade de Certeza (*Certainty Grid*) [3], Campos Potenciais [4], Campo de Forças Virtual (VFF) [5], Histograma do Campo Vetorial (VFH) [6], Diagrama de Proximidade (ND) [7], e Desvio Tangencial. [8].

Os métodos deliberativos são aqueles que contêm um modelo simbólico do mundo, ou seja, utilizam os dados provenientes dos sensores presentes no robô para construir um mapa do ambiente onde o robô será posto para navegar. O paradigma deliberativo é seqüencial e ordenado. Primeiro o robô faz um sensoriamento do mundo e constrói um mapa global, então o robô planeja todas as ações necessárias para alcançar o alvo e, finalmente, ele age em busca deste alvo [9]. Esta decomposição é conhecida como *SMPA* (*Sense, Model, Plan and Act*) [10]. Enquadram-se nesta abordagem os métodos da Grade de Certeza, do Histograma de Campo Vetorial (VFH) e o do Campo de Forças Vetoriais (VFF). A Figura 4 apresenta uma idéia geral da abordagem deliberativa.

Como na abordagem deliberativa é feito um mapeamento do ambiente onde o robô vai navegar, é possível que o mesmo percorra uma trajetória livre de obstáculos até o ponto de destino, desde que esse caminho exista. Entretanto, os métodos que utilizam

esta abordagem têm um elevado custo operacional e não são eficientes para o uso em ambientes dinâmicos.

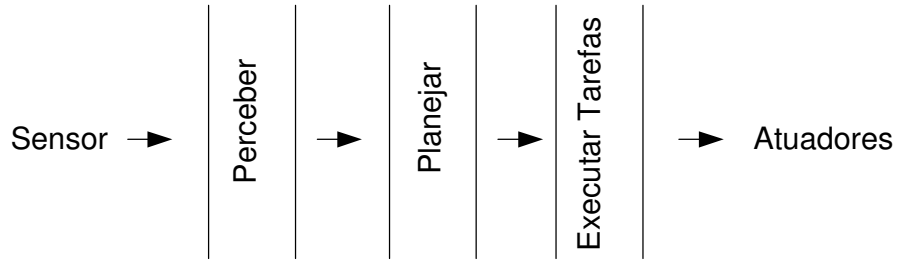


Figura 4: Arquitetura da abordagem Deliberativa.

Já a abordagem reativa não utiliza nenhum conhecimento prévio do meio, a não ser alguma informação muito geral como o conhecimento de que o ambiente é plano e fechado. Por este motivo não é possível planejar uma trajetória para o robô, pois não existe um mapa do ambiente. A navegação baseada na abordagem reativa foca o fato de que as *percepções* estão intimamente relacionadas com as *ações*, isto é, o robô reage ao que percebe. Isto resulta em simplicidade e em um baixo custo computacional, duas qualidades importantes em robótica móvel.

Assim, cada ação tomada por um robô que navega com algoritmos reativos é produzida com base nas informações sensoriais. Essa característica permite que um robô navegue em ambientes onde os obstáculos podem mudar de posição, já que o robô pode percebê-los e reagir. O método de Campos Potenciais e o da Detecção de Bordas são classificados como reativos. O esquemático da idéia geral da abordagem reativa está representado na Figura 5.

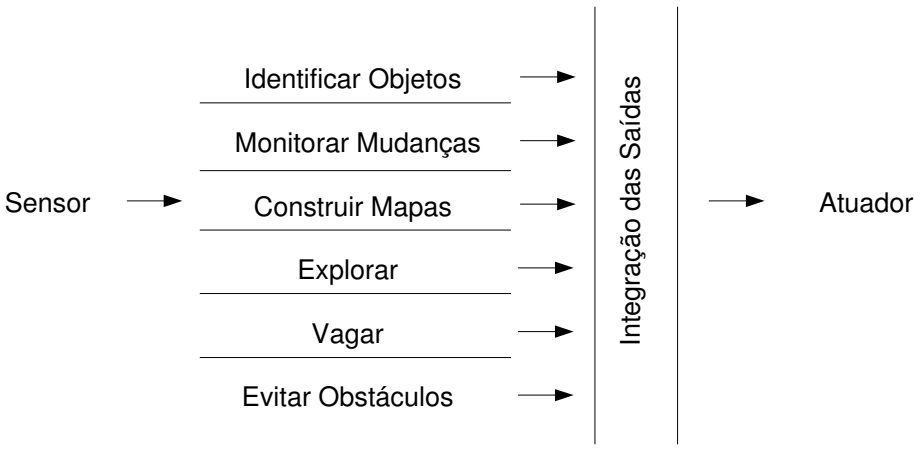


Figura 5: Arquitetura da abordagem Reativa.

Portanto, a navegação deliberativa é adequada para ambientes que não sofrem alterações, ou seja, ambientes estruturados. Já para os ambientes semi-estruturados, que são aqueles cujas informações não são totalmente conhecidas, a navegação reativa é mais apropriada.

Existe ainda o paradigma híbrido de navegação. Neste tipo, as duas técnicas de navegação citadas anteriormente são usadas em conjunto para se obter a capacidade de planejar as tarefas a serem realizadas e desviar de obstáculos dinâmicos presentes no ambiente.

É importante ressaltar que os algoritmos de desvio de obstáculos que foram implementados nesta Dissertação são baseados na arquitetura de controle reativa, ou seja, o robô deve "sentir" o ambiente e reagir de forma que a tarefa que lhe foi designada possa ser realizada.

1.1 Sensores Disponíveis

Os robôs móveis autônomos se caracterizam por sua capacidade de se movimentar em ambientes estruturados ou semi-estruturados sem a intervenção de um operador humano. Para isso, o robô deve possuir sensores que adquiram informações sobre o ambiente onde o robô está navegando, para que o mesmo possa tomar as decisões necessárias para obter sucesso na realização da tarefa que lhe foi designada.

É muito freqüente a utilização de sensores *laser* e de ultra-som (sonares) e câmaras de vídeo cuja informação permite, entre outras funções, obter informação para a criação de mapas do ambiente, localizar objetos e detectar obstáculos. Estes sensores estão presentes no robô móvel utilizado para a realização dos experimentos desta Dissertação, o PIONEER 2-DX da ActivMedia. Estes sensores podem ser utilizados em aplicações diversas como mapeamento de ambientes, detecção e desvio de obstáculos, etc.

1.1.1 Ultra-Som

Os sensores ultra-sônicos são bastante utilizados para a detecção de obstáculos em robótica móvel. Isso se dá devido ao seu baixo custo e grande disponibilidade no mercado, à capacidade de obter a distância a obstáculos de forma relativamente simples e precisa, e à imunidade a alguns agentes externos, como iluminação e ruídos audíveis [11].

Com um sensor ultra-sônico pode-se obter a informação de distância a partir da

técnica de tempo de vôo (TOF, do inglês *Time of Flight*). Mede-se o tempo entre a emissão do pulso e o retorno do eco. Este valor de tempo é multiplicado pela velocidade do som no ar para se obter um valor de distância. Entretanto, esta distância equivale ao dobro da distância real ao obstáculo. Logo, basta dividir o resultado obtido por dois para determinar a distância ao obstáculo.

Vários trabalhos têm utilizado sensores ultra-sônicos para mapear o ambiente de trabalho [3], desvio de obstáculos [12, 11, 13] e até em soldagem robotizada [14].

Há, porém, algumas situações em que os sensores ultra-sônicos encontram dificuldades em detectar certos tipos de obstáculos, como é o caso de uma quina ou de uma superfície com inclinação muito grande em relação ao eixo de emissão do sensor ultra-sônico.

1.1.2 Visão

A visão é, sem dúvida, o sentido mais complexo do ser humano. Ela é capaz de fornecer, de uma única vez, uma grande quantidade de informação sobre o meio. Utilizando informações visuais, o homem é capaz de identificar a posição de objetos, perceber movimentos, identificar formas e determinar dimensões. Tudo isso é feito sem que haja contato físico com o os objetos [15]. Devido a esta vantagem, os sistemas baseados em visão são utilizados em muitas aplicações, inclusive em robótica móvel.

Em [16] utiliza-se o fluxo óptico calculado numa imagem omnidirecional para detectar os obstáculos que existem no meio. Em [17, 18] utiliza-se, também, o fluxo óptico para fazer o desvio de obstáculos, porém utilizando-se visão estéreo. Pode-se, também, utilizar o processamento de imagens coloridas para se fazer a detecção de objetos numa cena. Em [19] é usada uma *webcam* posicionada a 3 m do solo. As imagens capturadas são processadas e os objetos detectados. Porém, aqui o objetivo não é desviar dos obstáculos encontrados e, sim, levá-los a uma determinada posição. Pode-se, também, utilizar imagens para determinar a posição e orientação de um robô móvel. Em [20] utilizam-se imagens obtidas por um sistema catadióptrico omnidirecional para determinar a pose de um robô móvel.

A visão é bastante sensível aos efeitos de iluminação. Além disso, o uso da visão requer um elevado custo computacional. Isto pode favorecer o uso da visão computacional para sistemas com maior capacidade de processamento, porém não impede que sistemas com menor capacidade de processamento usem este tipo de sistema sensorial.

1.1.3 *Laser*

O *laser* é um outro tipo de sensor que pode ser utilizado para determinação de distância. Este tipo de sensor possui uma precisão maior que a visão e, conseqüentemente, maior que o ultra-som. Não é sensível às variações da iluminação e sofre menos que o ultra-som quando se depara com quinas. Sensores *laser* emitem um feixe de *laser* de baixa potência sobre uma superfície para determinar sua distância à mesma.

Sensores de distância a *laser* podem fazer medidas de distância como 30 m, com precisão de milímetros [9]. Porém, sensores deste tipo possuem um custo mais elevado que sistemas de visão e sensores ultra-sônicos.

Uma aplicação para navegação de robôs móveis utilizando sensores *laser* é apresentada em [21]. Neste trabalho utiliza-se um sensor *laser* sobre um robô móvel para que este possa navegar desviando, de forma tangencial, dos obstáculos que surgirem em seu caminho.

Uma descrição mais detalhada sobre sensores *laser* e, especificamente, sobre o sensor de varredura *laser* que foi utilizado para a realização deste trabalho é apresentada no Capítulo 2.

1.2 Definição do Problema

O principal problema a ser tratado nesta Dissertação é a navegação segura de um robô móvel em ambientes semi-estruturados, utilizando a arquitetura de controle reativa e um sensor *laser* de varredura. A idéia principal é fazer com que o robô navegue entre dois pontos do ambiente, evitando obstáculos existentes no seu caminho. Estes obstáculos podem, inclusive, ser dinâmicos, isto é, podem aparecer de forma inesperada no caminho do robô. Uma outra abordagem é fazer com que o robô navegue pela linha média de um corredor.

1.3 Objetivo

O principal objetivo deste trabalho é, baseado na arquitetura de controle reativa, desenvolver algoritmos de controle confiáveis e adequados para garantir que o robô móvel seja capaz de realizar manobras suaves quando está diante de obstáculos, quando está navegando num corredor em busca do centro do mesmo e, também, quando navega num ambiente livre de obstáculos para atingir um ponto objetivo. Em todos estes casos, vale

destacar, a informação sensorial fica a cargo de um sensor *laser* de varredura instalado sobre o robô.

1.4 Estrutura da Dissertação

Esta Dissertação de Mestrado está organizada da seguinte forma:

- **Capítulo 1: Introdução**

  Este Capítulo descreve, em linhas gerais, o tema desta Dissertação. São apresentados os problemas que devem ser resolvidos, assim como os objetivos a serem alcançados com este trabalho.

- **Capítulo 2: Sensores *Laser***

  Este Capítulo apresenta algumas características de sensores *laser* em geral e, especificamente, do sensor de varredura *laser* SICK que foi utilizado para a realização deste trabalho.

- **Capítulo 3: Navegação em Corredores**

  É apresentado, neste Capítulo, o modelo cinemático do robô e um algoritmo para navegação de robôs móveis em corredores. Além disso, são apresentados o simulador utilizado e a ferramenta para a criação dos mapas que foram usados na simulação que também integra este Capítulo.

- **Capítulo 4: Navegação com Desvio de Obstáculos**

  Este Capítulo apresenta os dois controladores de desvio de obstáculos que foram implementados com base na arquitetura reativa. Também são mostradas diversas simulações que foram realizadas para comprovar a funcionalidade dos algoritmos de controle.

- **Capítulo 5: Resultados Experimentais**

  Neste Capítulo, além dos resultados obtidos com a realização de experimentos com o robô móvel PIONEER 2-DX, é feita, também, uma descrição do *hardware* e do *software* deste robô. Por fim, é feita uma análise dos resultados obtidos.

- **Capítulo 6: Conclusão e Trabalhos Futuros**

  Uma avaliação conclusiva desta Dissertação e as propostas para o futuro são apresentadas neste Capítulo.

2 Sensores *Laser*

> *"Se conheces o inimigo e conheces a ti mesmo, não deves temer pelo resultado de cem batalhas; se te conheces, mas não conheces o inimigo, por cada batalha ganha sofrerás também uma derrota; se não conheces o inimigo nem a ti mesmo, sucumbirás em todas as batalhas".*
>
> Sun Tzu

O *laser* (*Light Amplification by Stimulated Emission of Radiation*) é utilizado nos mais diversos campos tecnológicos e com diferentes finalidades (cirurgias, leitor de código de barras, depilação, impressoras, reprodutores de CD, etc.).

Nesta Dissertação, o *laser* foi utilizado em um sensor de distância. Este sensor é capaz de fornecer medidas de distância e foi utilizado para possibilitar a navegação segura de um robô móvel. Com este sensor de distância é possível realizar um sensoriamento do ambiente e fornecer as informações necessárias para a navegação do robô móvel.

Os sensores podem ser classificados, de acordo com a fonte de radiação (fonte emissora de energia), em em passivos e ativos.

Os sensores passivos dependem de uma fonte de radiação externa ou natural, como o Sol. Nesse caso estão, por exemplo, as câmaras fotográficas, que captam a radiação solar refletida.

São do tipo ativo quando não dependem de fonte de energia externa, isto é, são sensores que possuem sua própria fonte de energia, que é, então, refletida pelos objetos e registrada pelo sensor.

De acordo com esta classificação, os sensores *laser*, que emitem um feixe de luz infra vermelha e aguardam o seu retorno para a determinação da distância ao objeto, são classificados como ativos. Nesta classe também se enquadram os sensores de ultra-som, que emitem uma onda de som e esperam o eco para determinar a distância até o obstáculo.

Sistemas de Medição *Laser* (LMS – *Laser Measurement Systems*) são utilizados normalmente quando se deseja monitorar áreas, determinar medidas de distância, detectar e definir a posição de objetos.

Os *scanners* do LMS são sistemas de medidas sem contato, que mapeiam a região que os rodeiam em duas dimensões. Os dispositivos deste sistema de mapeamento não requerem refletores nem marcações no espaço de trabalho.

2.1 Métodos de Determinação de Distância

Os sensores *laser* são capazes de determinar a medida de distância de duas formas distintas. São elas a técnica da Diferença de Fase e a técnica do Tempo de Vôo (TOF, do inglês *Time Of Flight*).

2.1.1 Método da Diferença de Fase

Com esta técnica é possível determinar a distância até um objeto através da defasagem (deslocamento de fase) entre dois feixes de *laser*, um emitido e outro refletido. A Figura 6 apresenta um esquemático do arranjo utilizado para se medir a diferença de fase entre o feixe emitido e o feixe recebido enquanto a Figura 7 mostra duas ondas e a respectiva diferença de fase entre elas.

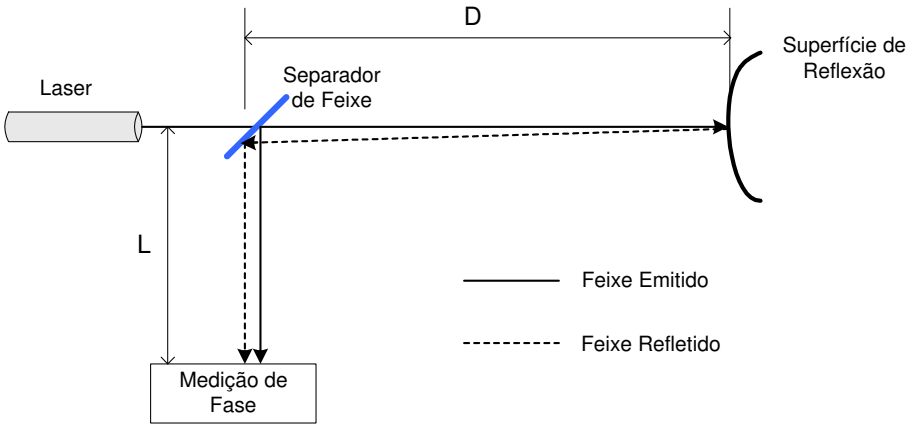


Figura 6: Arranjo para medidas de distância através da diferença de fase.

Para D=0, o feixe de referência (emitido) estará em fase com o feixe refletido. Para D>0, a distância total percorrida pelo feixe de *laser*, D', é dada por

$$D' = L + 2D \tag{2.1}$$

ou

$$\mathrm{D'} = \mathrm{L} + \frac{\alpha}{360} \cdot \lambda \tag{2.2}$$

onde λ é o comprimento de onda do *laser* e α é a diferença de fase entre o feixe emitido e o feixe recebido.

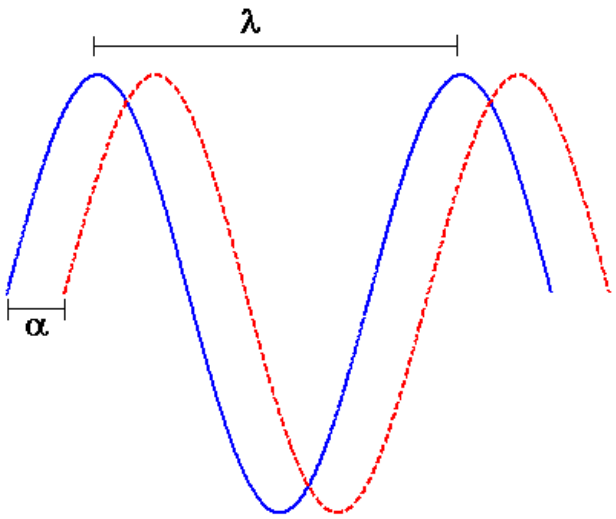


Figura 7: Diferença de fase entre duas ondas.

Mas, se $\alpha = 360°$, os dois sinais estão alinhados e não é possível diferenciar $\mathrm{D'} = \mathrm{L}$ de $\mathrm{D'} = \mathrm{L} + \mathrm{n}\lambda$, para n=1, 2, 3 ... Assim, uma solução única somente pode ser obtida se $\alpha < 360°$, o que implica que D$< 2\lambda$.

Igualando as Equações 2.1 e 2.2, a distância entre o sensor *laser* e um objeto é

$$\mathrm{D} = \frac{\alpha}{360} \cdot \frac{\lambda}{2}. \tag{2.3}$$

Entretanto, este método é impraticável para medir distâncias maiores que a metade do comprimento de onda do *laser*. Para um *laser* de hélio-neon, cujo comprimento de onda é 632,8 nm, a maior distância que pode ser medida é 316,4 nm. Uma solução simples para este problema, é modular a amplitude do *laser* com uma onda senoidal de alta freqüência. Por exemplo, para uma onda moduladora de 10 MHz, o comprimento de onda é de 30 m, o que possibilita uma medição de distância de até 15m. Uma onda senoidal modulada em amplitude pode ser vista na Figura 8.

Assim, o feixe modulado é emitido e, ao retornar, ele é demodulado e comparado com feixe de referência para determinar o deslocamento de fase.

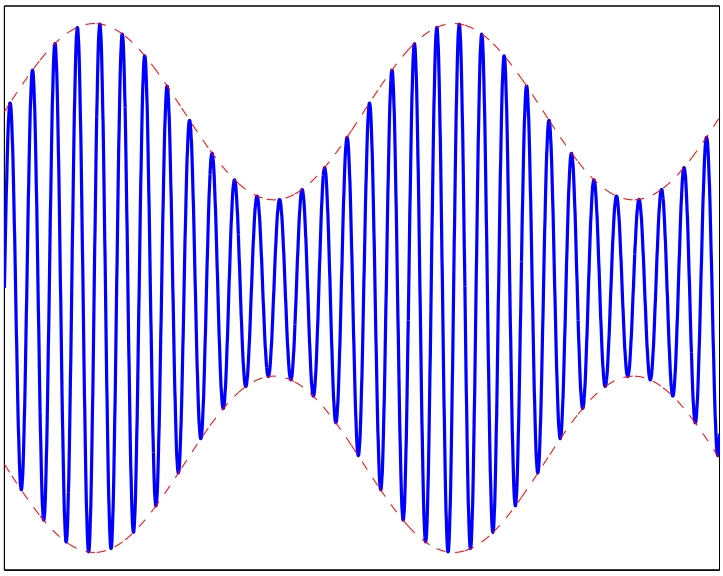


Figura 8: Onda modulada em amplitude.

2.1.2 Método do Tempo de Vôo

Neste método, um pulso de radiação é emitido pelo transmissor. Este pulso, ao encontrar uma superfície, retorna até o receptor. A medida de distância entre o sensor e a superfície refletora é determinada pela multiplicação da velocidade do pulso (no caso do *laser* esta velocidade é a velocidade da luz), pelo tempo que este levou para sair do transmissor e chegar até o receptor (ver Figura 9).

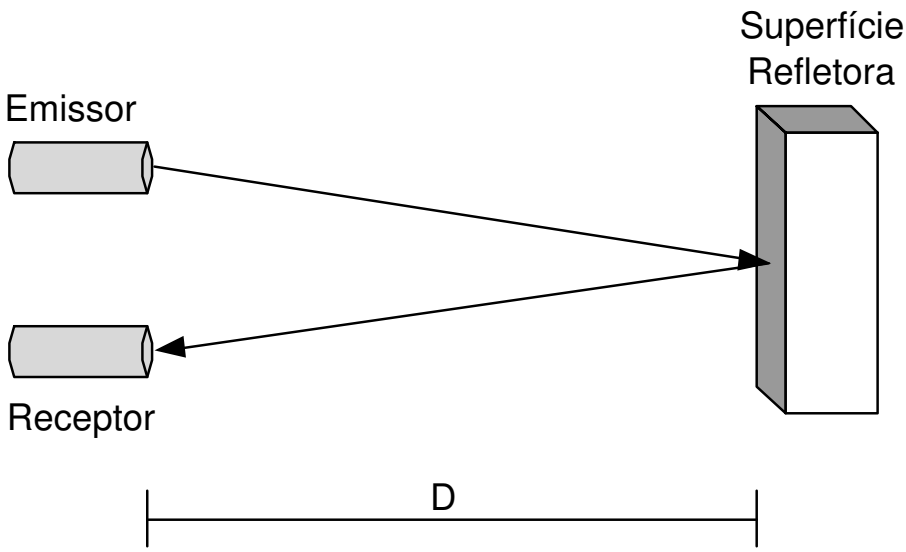


Figura 9: Princípio da técnica do tempo de vôo.

Assim sendo, a distância entre o sensor e o objeto, D, é dada por

$$2D = c \cdot \Delta t \tag{2.4}$$

logo,

$$D = \frac{c \cdot \Delta t}{2} \tag{2.5}$$

onde Δt é o tempo de propagação do pulso entre o transmissor e o receptor, c é a velocidade da luz no meio em que se propaga e D é a distância entre o instrumento de medição e a superfície refletora.

Considerando que a velocidade da luz no ar, c, é igual a $3 \times 10^8\ m/s$, o tempo entre o pulso emitido e o recebido é muito pequeno. Portanto, para se efetuar medidas de distância a partir deste método, os equipamentos utilizados devem ser altamente precisos e, conseqüentemente, caros.

2.2 O Sensor de Varredura *Laser* Utilizado

O sensor *laser* utilizado nesta Dissertação de Mestrado é o LMS 200, fabricado pela *SICK* (Figura 10). Este sensor fornece a medida de distância através da medição do tempo de vôo dos pulsos dos feixes de *laser*, os qual são emitidos e, se um objeto for encontrado, refletidos.

Figura 10: O sensor *laser*.

O feixe de *laser* é deflectado por um espelho rotativo interno, realizando uma varre-

dura de 180°, em forma de leque, da área os arredores do sensor, conforme a Figura 11. As medidas de distância podem ser tomadas em intervalos de 1°, 0.5° e 0.25°. Esta escolha é feita através de *software*. O contorno de um alvo pode ser obtido por meio de uma seqüência de impulsos recebidos cujas medidas são disponibilizadas via uma interface serial.

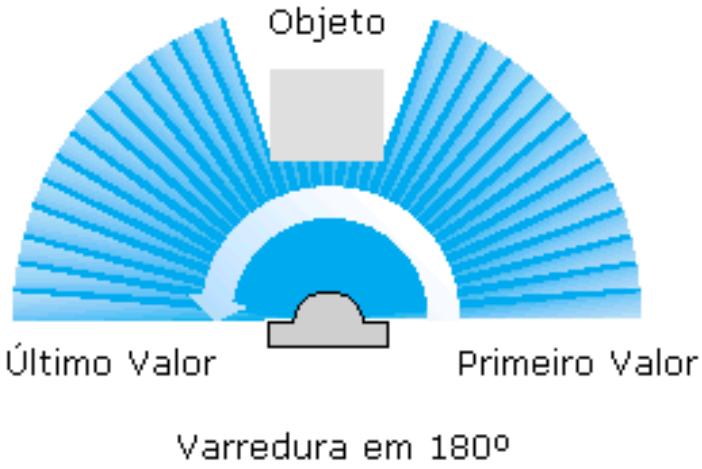


Figura 11: Varredura realizada pelo sensor *laser*.

A Figura 12 ilustra o feixe de *laser* emitido e o respectivo feixe refletido por um objeto. A distância que o sensor se encontra deste objeto é determinada a partir deste dois feixes de *laser*.

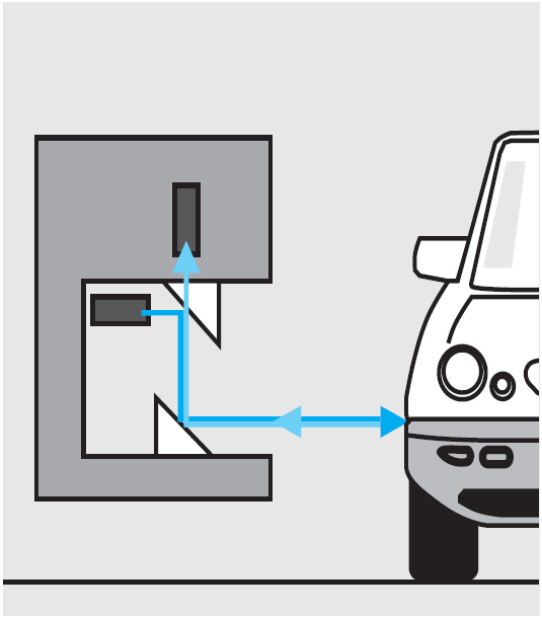


Figura 12: Princípio de operação do LMS.

Outras características do LMS 200 podem ser vistas na Tabela 1.

Tabela 1: Características do LMS 200.

Alcance	150m
Resolução	10mm
Resolução Angular	0.25°, 0.5°, 1.0° via software
Tempo de Resposta	52, 26, 13 ms dependendo da resolução angular
Tensão de Alimentação	24V
Temperatura de Operação	0° C a 50° C
Dimensões	155 x 185 x 156 mm (L x A x P)
Peso	4.5kg

3 Navegação em Corredores

"... Descobri como é bom chegar quando se tem paciência. E para se chegar, onde quer que seja, aprendi que não é preciso dominar a força, mas a razão. É preciso antes de mais nada querer".

Amyr Klynk

Quando robôs móveis autônomos são postos para navegar em ambientes fechados, eles devem ter a capacidade de seguir corredores, paredes, contornar esquinas, entrar e sair de salas, etc [22]. Alguns algoritmos baseados em visão e ultra-som já foram propostos para a navegação de robôs móveis em corredores.

Em [23, 24] utiliza-se um sistema monocular, ou seja, uma única câmara apontada para frente. As imagens capturadas são processadas e as retas paralelas referentes às paredes do corredor são analisadas. O algoritmo de controle faz com que o ponto de fuga (ponto onde tais retas se interceptam) seja posicionado no centro da imagem. Desta forma, o robô navega pelo meio do corredor. Em [25] utiliza-se fluxo ótico em duas regiões simétricas de uma única imagem e busca-se igualar tais valores a fim de que o robô navegue pela linha central do corredor. Já em [17, 18] são usadas duas câmaras montadas no robô, cada uma apontada a partir de um dos lados do robô. Dessa maneira, calcula-se o fluxo ótico em cada imagem e, a partir destes fluxos, o robô navega seguindo paredes e desviando de obstáculos.

Um algoritmo estável para a navegação em corredores com base nas medidas fornecidas pela leitura dos sensores ultra-sônicos é apresentado em [13]. São obtidas informações de distância fornecidas pelos sensores laterais presentes no robô e estas são usadas para fazer o controle do mesmo.

Neste Capítulo, é apresentado um controlador não linear projetado para fazer um robô móvel seguir corredores, baseado em um sensor *laser*. Inicialmente, apresenta-se o modelo

cinemático do robô móvel que será utilizado. Em seguida, apresenta-se o controlador para navegação em corredores. E, logo após, mostra-se a maneira como as variáveis de estado são calculadas.

Apesar de nesta Dissertação se utilizar as informações fornecidas por um sensor *laser*, fez-se uma análise de como as variáveis de estado são calculadas utilizando sensores de ultra-som, como em [22, 13]. Esta análise foi realizada pois o controlador para navegação em corredores implementado neste trabalho é o mesmo apresentado em [22, 13].

Após realizar o estudo da maneira como foi realizada o cálculo das variáveis de estado utilizando os sensores de ultra-som, apresentam-se dois métodos para o cálculo destas mesmas variáveis de estado, desta vez utilizando as informações fornecidas pelo sensor *laser*.

Este controlador permite que um robô móvel navegue pelo eixo central de um corredor, e foi desenvolvido com base nas equações cinemáticas do robô móvel utilizado.

3.1 Modelo Cinemático do Robô

O modelo cinemático de um robô fornece as equações que descrevem seu movimento em seu espaço de trabalho. Considerando que o robô seja do tipo monociclo com acionamento diferencial, seu movimento é composto pela ação conjunta das velocidades linear v, sempre sobre um dos eixos do sistema de coordenadas do robô <a>, e angular ω, em torno do eixo perpendicular ao plano de movimento. A Figura 13 mostra o robô e o seu sistema de coordenadas <a>.

Seja $[x, y, \varphi]^T$ o vetor que representa a postura do robô, onde x e y são as coordenadas do seu centro de rotação e φ é o seu ângulo de orientação (ver Figura 13). O conjunto de equações cinemáticas que descrevem o movimento do veículo é dado por

$$\begin{cases} \dot{x} = v\cos(\varphi) \\ \dot{y} = v\,\mathrm{sen}(\varphi) \\ \dot{\varphi} = \omega \end{cases}, \tag{3.1}$$

representado em coordenadas cartesianas, onde v, φ e ω são como indicados na Figura 13.

Seja agora o caso em que o robô móvel navega em um corredor. Nesta situação, as variáveis de estado são determinadas em relação ao corredor. São elas $\tilde{x}$, a distância do centro do robô à linha média do corredor e φ, que é a orientação do robô em relação ao

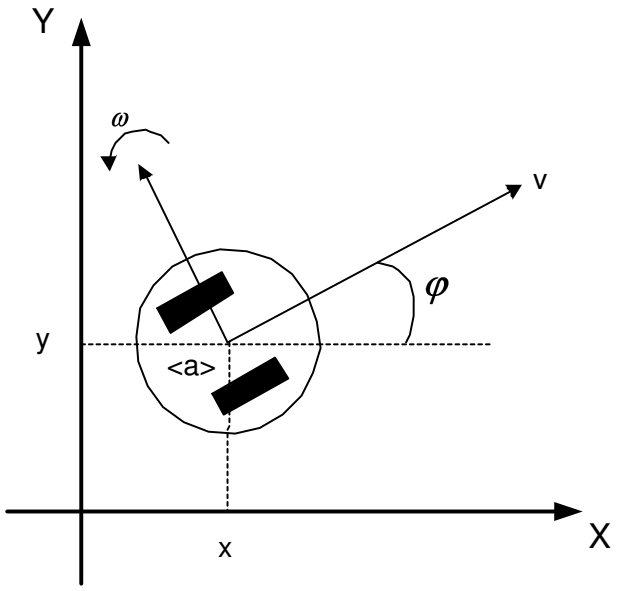


Figura 13: O robô e o sistema de coordenadas inercial.

eixo do corredor, conforme ilustrado na Figura 14.

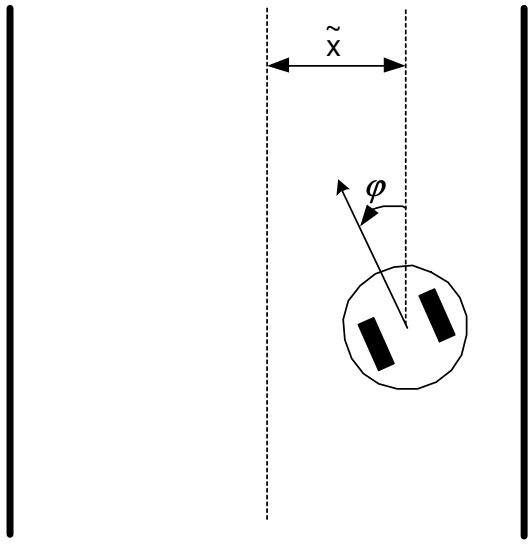


Figura 14: O robô no corredor.

Para navegação em corredores, pode-se considerar apenas duas varáveis de estado [13], o erro de posição do robô, $\tilde{x}$, e o erro de orientação do mesmo com relação à linha média do corredor. Para a situação de navegação em corredores, o erro de orientação do robô é igual à orientação do mesmo, φ. Assim pode-se escrever as equações dos erros de posição e orientação do robô como

$$\begin{cases} \dot{\tilde{x}} = v \operatorname{sen}(\varphi) \\ \dot{\varphi} = \omega \end{cases}. \tag{3.2}$$

onde v e ω são, respectivamente, as velocidades linear e angular do robô.

3.2 Projeto do Controlador

Considerando que o movimento do robô no corredor é descrito pelas Equações 3.2, o objetivo do controlador é fazer com que as variáveis de estado $\tilde{x}$ e φ tendam para zero de maneira assintótica.

Para este controlador, apresentado em [22], uma velocidade linear v constante é adotada, e a ação de controle ω, representando a velocidade angular do robô, é dada por

$$\omega = -k_1(\varphi)\,\varphi - k_2(\tilde{x})\,\tilde{x} v \frac{\operatorname{sen}(\varphi)}{\varphi}, \tag{3.3}$$

onde $k_1(\varphi)$ e $k_2(\tilde{x})$ são funções positivas escolhidas para não saturar o sinal de controle ω enviado ao robô móvel utilizado. Assim,

$$k_1(\varphi) = \frac{k_1}{a_1 + |\varphi|} \tag{3.4}$$

e

$$k_2(\tilde{x}) = \frac{k_2}{a_2 + |\tilde{x}|}. \tag{3.5}$$

com k_1, k_2, a_1 e a_2 sendo constantes positivas.

Substituindo a Equação 3.3 em 3.2 obtêm-se as equações de malha fechada do sistema para navegação em corredores, que são

$$\dot{\tilde{x}} = v \operatorname{sen}(\varphi) \tag{3.6}$$

e

$$\dot{\varphi} = -k_1(\varphi)\,\varphi - k_2(\tilde{x})\,\tilde{x} v \frac{\operatorname{sen}(\varphi)}{\varphi}. \tag{3.7}$$

Para verificar a estabilidade do sistema de controle em malha fechada quando o controlador proposto é utilizado, foi escolhida uma função que representa o somatório das energias potenciais do robô móvel, referentes aos erros de posição e de orientação do mesmo. Esta função, escolhida como função candidata de Lyapunov, é dada por

$$V(\tilde{x}, \varphi) = \frac{\varphi^2}{2} + \int_0^{\tilde{x}} k_2(\eta)\, \eta d\eta. \tag{3.8}$$

O objetivo deste controlador, como já foi mencionado, é levar o robô, de maneira suave, para o centro do corredor e fazer com que ele continue navegando paralelamente às paredes do mesmo, ou seja, fazer com que as variáveis de estado tendam assintoticamente para zero. Para isso, a função candidata de Lyapunov deve ser definida positiva, e a sua derivada temporal deve ser definida negativa. Caso isso aconteça, a soma das energias potenciais do sistema tem sempre um valor positivo, e tende para zero com o passar do tempo. Isso implica na convergência assintótica das variáveis de estado do sistema de controle, garantindo a estabilidade do mesmo.

Calculando a derivada temporal de $V(\tilde{x}, \varphi)$ obtém-se

$$\dot{V}(\tilde{x}, \varphi) = \varphi\dot{\varphi} + k_2(\tilde{x})\, \tilde{x}\dot{\tilde{x}}. \tag{3.9}$$

Substituindo os valores de $\dot{\tilde{x}}$ e $\dot{\varphi}$ da Equação 3.2 em 3.9, esta última pode ser reescrita como

$$\dot{V}(\tilde{x}, \varphi) = \varphi\left[-k_1(\varphi)\, \varphi - k_2(\tilde{x})\, \tilde{x}v\frac{\operatorname{sen}(\varphi)}{\varphi}\right] + k_2(\tilde{x})\, \tilde{x}v \operatorname{sen}(\varphi). \tag{3.10}$$

Pode-se observar em 3.10 que o segundo termo dentro dos colchetes pode ser cancelado com o termo que está fora deles. Assim, chega-se à forma final da derivada da função candidata de Lyapunov, que é

$$\dot{V}(\tilde{x}, \varphi) = -k_1(\varphi)\, \varphi^2 \leq 0. \tag{3.11}$$

Nota-se que $\dot{V}(\tilde{x}, \varphi)$ é definida negativa, já que $k_1(\varphi)$ é positiva. Com isso, conclui-se que o sistema é assintoticamente estável, como se desejava. A prova completa da estabilidade deste sistema de controle pode ser vista em [22].

3.3 Cálculo das Variáveis de Estado

Visto que para este controlador a velocidade linear foi considerada constante, a única variável de controle é a velocidade angular. Esta, como pode ser vista na Equação 3.3, é

função do erro de posição do robô ($\tilde{x}$) e do erro de orientação do mesmo em relação ao corredor (φ), que são as variáveis de estado do sistema. Para que o controlador apresentado funcione de maneira correta, é necessário que ele receba os valores das variáveis de estado a cada ciclo de controle. Neste trabalho, são apresentadas três maneiras distintas de se calcular estas variáveis de estado. A primeira delas, apresentada em [22] e [13], utiliza os dados provenientes de sensores ultra-sônicos. Nas outras duas, as medidas são obtidas a partir de um sensor *laser*. Como já dito anteriormente, nesta Dissertação, apenas serão implementados os algoritmos baseados nas informações provenientes do sensor *laser*.

3.3.1 Usando sensores ultra-sônicos

A situação ilustrada na Figura 15 mostra um robô móvel num corredor, assim como a disposição dos sensores ultra-sônicos existentes no mesmo. Os sensores que foram utilizados para se obter as medidas necessárias para o cálculo das variáveis de estado, para esta situação, também estão indicados.

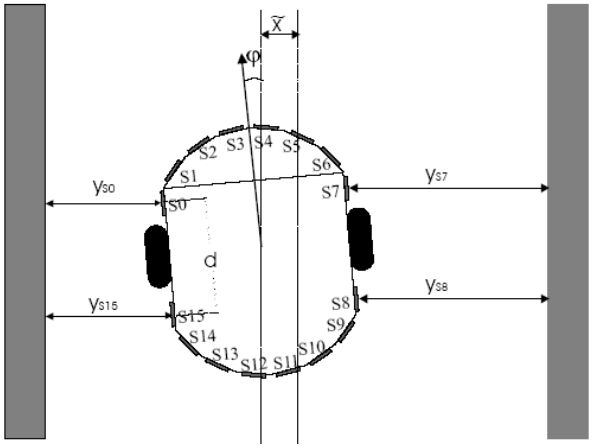


Figura 15: Disposição dos sensores ultra-sônicos e as variáveis de estado $\tilde{x}$ e φ.

As distâncias do robô às paredes à sua direita e à sua esquerda são calculadas como

$$\begin{cases} d_{esq} = \dfrac{y_{S0} + y_{S15}}{2} \\ \\ d_{dir} = \dfrac{y_{S7} + y_{S8}}{2} \end{cases}, \tag{3.12}$$

onde y_{S0}, y_{S7}, y_{S8} e y_{S15} são as distâncias assinaladas na Figura 15.

A distância do centro do robô à linha média do corredor ($\tilde{x}$) é função de d_{dir} e de d_{esq}, e seu valor pode ser obtido através da equação

$$\tilde{x} = \frac{d_{dir} - d_{esq}}{2}. \tag{3.13}$$

Para determinar a outra variável de estado, o erro de orientação em relação ao corredor, é preciso conhecer a diferença entre as medidas dos sensores laterais adjacentes, ou seja,

$$dif = (y_{S15} - y_{S0}) = (y_{S7} - y_{S8})\,. \tag{3.14}$$

Assim, o ângulo φ mostrado na figura é dado por

$$\varphi = \text{sen}^{-1}\left(\frac{dif}{d}\right). \tag{3.15}$$

onde d é a distância entre os sensores ultra-sônicos laterais do robô (ver Figura 15). Nota-se, porém, que a Equação 3.15 somente é válida se $\varphi \approx 0$, ou seja, ela dá um valor aproximado de φ quando $\varphi \approx 0$.

É válido mencionar que as informações fornecidas pelos sensores ultra-sônicos podem corresponder a medidas ruins, dependendo de circunstâncias, tais como portas abertas, quinas, ou mesmo quando o robô possui uma inclinação relativamente grande em relação ao corredor. Este último é um dos grandes problemas dos sonares, pois se a superfície encontrada pelo sinal ultra-sônico tiver um posicionamento maior que o semi-ângulo do lóbulo de emissão do sensor ultra-sônico [14], o sinal será refletido para longe do sonar e o objeto não será detectado. Isto se deve ao fato de que quanto mais inclinado estiver o sensor com relação ao corredor menor é a amplitude do eco recebido pelo transdutor [11], ocasionando, assim, um maior erro na medida efetuada. No robô PIONEER 2DX, utiliza-se o sensor ultra-sônico eletrostático da Polaroid®. Para este sensor, o ângulo de inclinação máximo para que as medidas fornecidas por ele sejam confiáveis é de aproximadamente 17° [26, 11], o qual é chamado de semi-ângulo de abertura do lóbulo de energia do sensor.

3.3.2 Usando o sensor *Laser*

- **Primeiro Método**

É apresentada, agora, uma outra maneira de se calcular as variáveis de estado do controlador proposto para a navegação em corredores. Desta vez, as variáveis serão cal-

culadas com base em um sensor *laser*. Essa nova estratégia de cálculo das variáveis de estado foi necessária, pois diferentemente dos sensores ultra-sônicos, que estão presentes tanto na frente quanto na parte de trás do robô, o sensor *laser* utilizado proporciona apenas medidas frontais. Dessa forma, não é possível, com o sensor *laser*, obter as quatro medidas laterais, como foi feito com os sensores de ultra-som. A Figura 16 mostra o sensor *laser* utilizado para adquirir as informações necessárias para o referido cálculo, enquanto a Figura 17 ilustra a forma como ele faz a leitura das medidas.

Figura 16: O sensor *laser*.

Tal sensor fornece, de uma única vez, 181 medidas de distância (de 0° a 180° em intervalos de 1°).

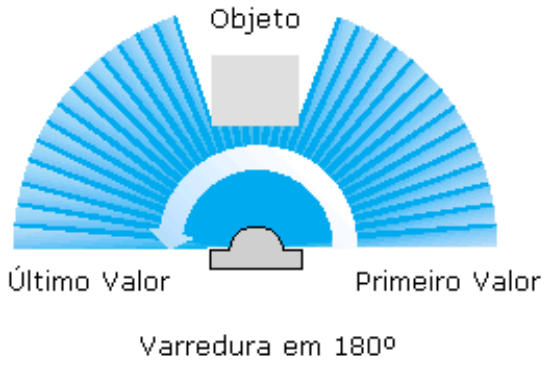


Figura 17: Varredura realizada pelo sensor *laser*.

Para determinar os valores de $\tilde{x}$ e φ, foram utilizadas as medidas dos sensores posicionados a 0° e 20° (medidas do lado direito) e a 160° e 180° (medidas do lado esquerdo).

A Figura 18 traz uma situação em que o robô encontra-se num corredor e ilustra as medidas usadas para calcular as variáveis de estado.

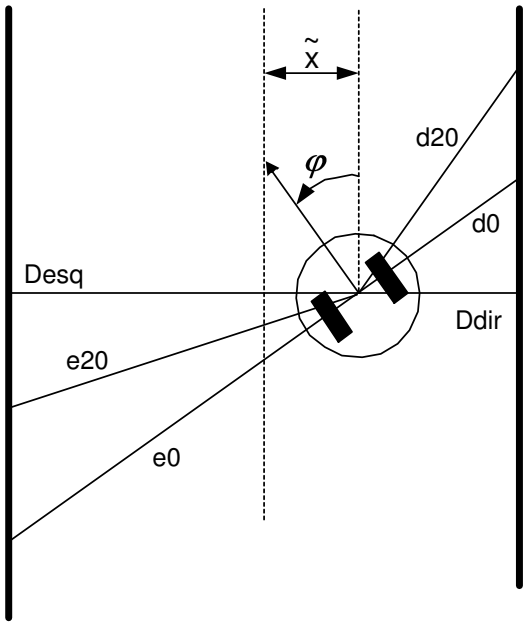


Figura 18: O robô no corredor ilustrando as medidas tomadas pelo sensor *laser* para se efetuar o cálculo de $\tilde{x}$ e φ.

Na Figura 18, D_{dir} e D_{esq} são as distâncias do robô até as paredes direita e esquerda, respectivamente, enquanto d_0, d_{20}, e_0 e e_{20} são as medidas fornecidas pelo sensor *laser* nas direções 0°, 20°, 180° e 160°, respectivamente. As variáveis de estado do sistema, $\tilde{x}$ e φ são calculadas baseadas nestas medidas. A Figura 19 mostra a representação geométrica que foi utilizada para se calcular as variáveis de estado para o lado direito do robô.

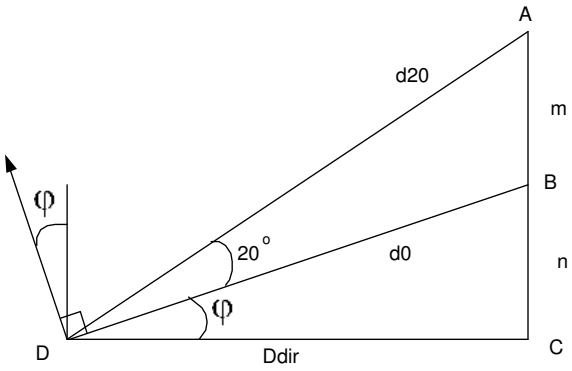


Figura 19: Representação geométrica para definir $\tilde{x}$ e φ

Os valores de d_0 e d_{20} são conhecidos, pois representam as medidas fornecidas pelo sensor *laser* nas direções 0° e 20°. Desta forma, o valor de m pode ser calculado aplicando-se a lei dos cossenos no triângulo ABD, já que se conhece a medida de dois lados do triângulo e o valor do ângulo entre eles, que nesta situação caso é igual a 20°. Daí resulta que

$$m = \sqrt{d_0^2 + d_{20}^2 - 2d_0 d_{20} \cos(20^\circ)}. \tag{3.16}$$

A seguir, aplica-se o Teorema de Pitágoras nos triângulos ACD e BCD, obtendo-se, respectivamente,

$$d_{20}^2 = (m+n)^2 + D_{dir}^2, \tag{3.17}$$

e

$$d_0^2 = n^2 + D_{dir}^2. \tag{3.18}$$

Subtraindo-se 3.17 e 3.18, o termo D_{dir}^2 desaparece, já que é comum às duas equações, resultando em

$$d_{20}^2 - d_0^2 = (m+n)^2 - n^2, \tag{3.19}$$

e, como o valor de m já foi determinado na Equação 3.16, basta isolar n para obtê-lo, através de

$$n = \frac{d_{20}^2 - d_0^2 - m^2}{2m}. \tag{3.20}$$

Agora que já se conhecem as duas variáveis auxiliares (m e n), basta substituir os seus valores em 3.17 para determinar o valor da distância do robô à parede direita (D_{dir}). Assim,

$$D_{dir} = \sqrt{d_{20}^2 - (m+n)^2}. \tag{3.21}$$

O cálculo da distância que o robô se encontra da parede esquerda (D_{esq}) é feito de maneira análoga.

Conhecidas as distâncias do robô às paredes do corredor e o valor das variáveis auxiliares (m e n), é possível determinar as variáveis de estado do sistema de controle, $\tilde{x}$ e φ, que são dadas por

$$\tilde{x} = \frac{D_{dir} - D_{esq}}{2} \tag{3.22}$$

e

$$\varphi = \tan^{-1}\left(\frac{n}{D_{dir}}\right). \tag{3.23}$$

Observa-se, aqui, que não há restrição no sentido de que $\varphi \approx 0$, pois como a função utilizada é o arco-tangente ($\tan^{-1}$) não há restrição do valor que é passado como parâmetro.

É válido mencionar que o ângulo φ é calculado tanto para o lado direito quanto para o lado esquerdo. Sendo assim, o ângulo utilizado como variável de estado é obtido pela média aritmética dos ângulos calculados nos lados esquerdo e direito, ou seja,

$$\varphi = \frac{\varphi_{dir} + \varphi_{esq}}{2}. \tag{3.24}$$

Com as variáveis de estado determinadas, basta substituir seus valores na equação do controlador (3.3) para obter o valor da ação de controle que será enviada ao robô para que o mesmo navegue de tal forma que o seu objetivo seja alcançado, ou seja, encontrar o centro do corredor e ali permanecer.

- **Segundo Método**

Assim como no método apresentado anteriormente, utilizam-se as leituras fornecidas pelo sensor *laser* nas direções 0°, 20°, 180° e 160° representadas, respectivamente, por d_0, d_{20}, e_0 e e_{20}, para se calcular as variáveis de estado $\tilde{x}$ e φ. A situação em que um robô se encontra num corredor, e as medidas usadas para se determinar as variáveis de estado, são ilustradas na Figura 18.

Uma nova maneira de se determinar os valores de $\tilde{x}$ e φ é baseada nas representações geométricas que estão apresentadas na Figura 20, oriundas da situação ilustrada na Figura 18.

Os valores de d_0 e d_{20} são conhecidos, pois representam as medidas fornecidas pelo sensor *laser* nas direções 0° e 20°. A Figura 20 (a) mostra a representação geométrica que foi utilizada para se calcular as variáveis de estado para o lado direito do robô, de acordo com o que foi mostrado na Figura 18, enquanto a Figura 20 (b) apresenta esta mesma representação, porém rotacionada para facilitar a visualização. Baseado na Figura 20 (b), a primeira variável de estado a ser calculada é o ângulo de orientação do robô com relação ao corredor, φ.

Considerando um sistema de coordenadas centrado no robô, pode-se calcular as coordenadas dos pontos A e B, mostrados na Figura 20. Estes pontos estão localizados sobre

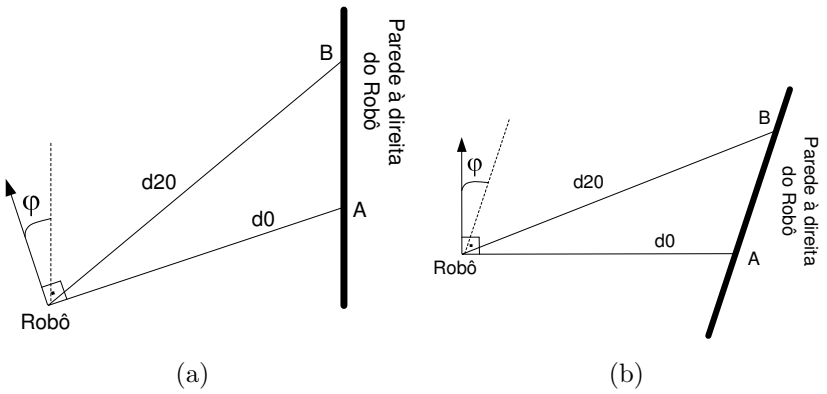


Figura 20: Representação geométrica para definir $\tilde{x}$ e φ.

a parede que se encontra à direita do robô. É possível notar que a orientação do robô é igual ao complemento do ângulo que representa a inclinação da reta que passa pelos pontos A e B (ver Figura 21). Este ângulo de inclinação da reta que passa pelos pontos A e B é chamado de β. Desta forma, se as coordenadas destes dois pontos (A e B) forem conhecidas, é possível determinar a orientação do robô com relação ao corredor.

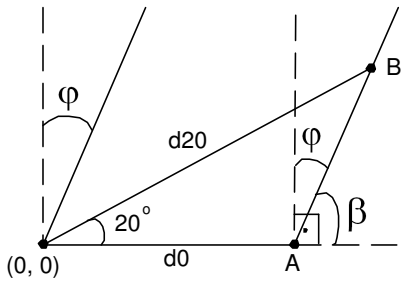


Figura 21: Determinação do ângulo φ.

Como o sistema de coordenadas está no robô e são conhecidas as distâncias d_0 e d_{20} e o ângulo entre elas, que é de 20°, pode-se, facilmente, determinar as coordenadas dos pontos A e B através de expressões trigonométricas. Assim, as coordenadas destes pontos são

$$\begin{cases} x_A = d_0 \\ y_A = 0 \end{cases} \tag{3.25}$$

e

$$\begin{cases} x_B = d_{20} \cos(20°) \\ y_B = d_{20} \operatorname{sen}(20°) \end{cases} . \tag{3.26}$$

Como já se mencionou, o ângulo φ é igual ao complemento do ângulo de inclinação da reta que une os pontos A e B, ou seja $\varphi = 90° - \beta$. A inclinação da reta que une A e

B é dada por

$$\beta = \tan^{-1}\left(\frac{y_B - y_A}{x_B - x_A}\right), \tag{3.27}$$

logo, a orientação do robô é

$$\varphi = 90^\circ - \tan^{-1}\left(\frac{y_B - y_A}{x_B - x_A}\right). \tag{3.28}$$

Substituindo os valores de x_A, x_B, y_A e y_B das Equações 3.25 e 3.26 na Equação 3.28 obtém-se o valor do ângulo φ como sendo

$$\varphi = 90^\circ - \tan^{-1}\left(\frac{d_{20}\,\mathrm{sen}\,(20^\circ)}{d_{20}\cos(20^\circ) - d_0}\right). \tag{3.29}$$

Observa-se, mais uma vez, que não há qualquer restrição no sentido de que $\varphi \approx 0$.

Este ângulo é calculado, de maneira análoga, para o lado esquerdo do robô. Sendo assim, o ângulo φ utilizado como variável de estado para este controlador é obtido pela média aritmética dos ângulos calculados nos lados esquerdo e direito, ou seja,

$$\varphi = \frac{\varphi_{dir} + \varphi_{esq}}{2}. \tag{3.30}$$

Após a determinação do ângulo φ, que é o erro de orientação do robô com relação ao corredor, deve-se encontrar o valor da outra variável de estado, ou seja, o erro de posição do robô com relação à linha média do corredor, $\tilde{x}$. Para isso, é necessário conhecer as distâncias do robô às paredes direita e esquerda, D_{dir} e D_{esq}, respectivamente.

A Figura 22 ilustra uma situação em que robô tem uma orientação positiva, ou seja, está com sua frente direcionada para a parede esquerda do corredor.

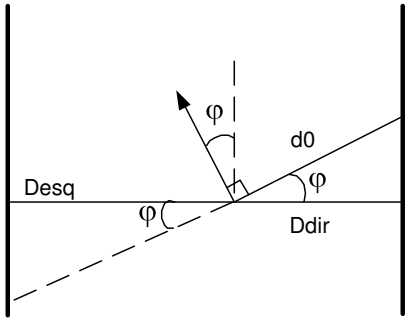


Figura 22: Cálculo de $\tilde{x}$.

Para a situação apresentada na Figura 22, a distância até a parede esquerda é igual à medida fornecida pelo sensor cuja orientação é igual ao suplemento do ângulo que representa a orientação do robô, ou seja,

$$D_{esq} = medida\left(180 - int\left(\varphi\right)\right), \tag{3.31}$$

onde $int\left(\varphi\right)$ retorna a parte inteira do ângulo φ e $medida(.)$ fornece o valor da distância capturada pelo sensor cuja orientação é passada como parâmetro. Já a distância até a parede direita é calculada como

$$D_{dir} = d_0 \cos\left(\varphi\right). \tag{3.32}$$

Agora que as distâncias entre o robô e as duas paredes são conhecidas, é possível determinar a segunda variável de estado, o erro de posição do robô, como sendo

$$\tilde{x} = \frac{D_{dir} - D_{esq}}{2}. \tag{3.33}$$

Determinadas as variáveis de estado do controlador, basta substituir seus valores na equação (3.3) para obter o valor da ação de controle que será enviada ao robô para que o mesmo navegue a fim de alcançar o seu objetivo, ou seja, encontrar o centro do corredor e ali permanecer.

Quando $\varphi < 0$, ou seja, o robô possui orientação negativa, as distâncias entre ele e as duas paredes são determinadas de maneira análoga.

Para comprovar a funcionalidade dos controladores implementados nesta Dissertação, foram realizadas simulações antes de fazer os testes com o robô propriamente dito. Para isso, duas ferramentas foram bastante úteis no que tange à simulação e à criação dos ambientes nos quais o robô será posto para navegar.

Uma delas foi o simulador *SRIsim*, que acompanha o robô PIONEER 2DX. O *SRIsim* foi usado para a obtenção dos resultados que são apresentados a seguir. Este simulador utiliza modelos reais dos sensores presentes no robô (sonar, *laser*, encoder), considerando os erros destes sensores. Entretanto, o modelo dinâmico do robô não é levado em consideração, pois os valores das velocidades angular e linear medidos diferem nas simulações e nos experimentos. Os experimentos serão apresentados no Capítulo 5. A Figura 23 mostra a interface gráfica do simulador *SRIsim*.

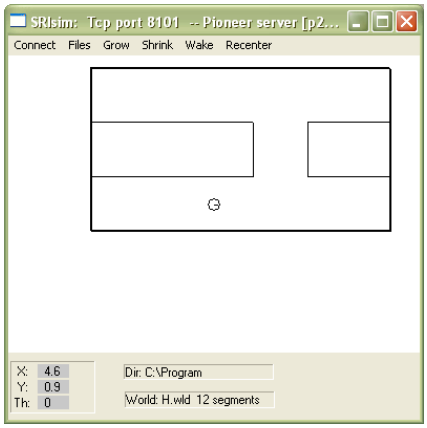


Figura 23: O simulador utilizado.

Além do simulador *SRIsim*, foi utilizado também o *software Mapper*. Este *software* permite a criação de ambientes onde o robô será posto a navegar. Pode-se ver na Figura 24 um mapa construído com tal ferramenta.

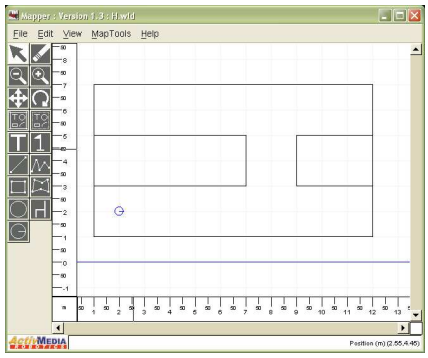


Figura 24: Ferramenta *Mapper* para a criação de mapas.

Para realização das simulações e dos experimentos que serão apresentados no Capítulo 5, os controladores foram implementados em programas escritos em C++ utilizando-se a biblioteca "Aria.h", a qual contém as funções que possibilitam capturar dados do robô e, também, enviar comandos para o mesmo. Assim, a cada instante de execução do laço de controle, são capturadas as informações provenientes da odometria do robô e do sensor *laser* instalado a bordo do mesmo. Estas informações são passadas como parâmetros para os controladores que calculam o valor das variáveis de controle e as enviam para o robô. Cabe ressaltar que tal laço de controle é executado uma vez a cada 100 ms.

São apresentados, neste ponto, dois resultados obtidos a partir de simulações nas quais o robô é posto para navegar em um corredor de 2 m de largura por um tempo de 30 s. Ele está posicionado, inicialmente, a 60 cm de distância da parede direita do corredor e orientado com relação ao eixo do corredor. Na primeira, as variáveis de estado

são calculadas de acordo com o primeiro método apresentado, enquanto na simulação seguinte estas variáveis são determinadas como no segundo método. Foram utilizadas as especificações para o controlador de navegação em corredores: $a_1 = 3$, $a_2 = 2$, $k_1 = 2$, $k_2 = 2$, $v = 300\ mm/s$. Estes valores foram escolhidos de forma empírica. As Figuras 25, 26 e 27 ilustram os resultados da simulação do controlador de navegação em corredores utilizando o primeiro método para o cálculo das variáveis de estado, enquanto as Figuras 28, 29 e 30 mostram o desempenho deste mesmo controlador, calculando-se as variáveis de estado de acordo com o segundo método.

A Figura 25 mostra a trajetória descrita pelo robô enquanto o mesmo navegava no corredor.

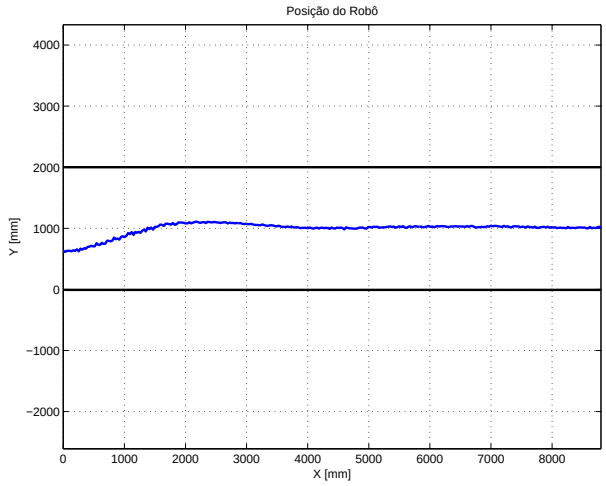


Figura 25: Trajetória descrita pelo robô.

Por observação da Figura 25, pode-se notar que o robô encontrou o meio do corredor e continuou navegando por ele em paralelo às paredes do corredor, alcançando, desta forma, o seu objetivo.

As velocidades lineares enviadas ao robô e desenvolvida pelo mesmo, obtida através das leituras dos *encoders* presentes nas rodas do robô, são apresentadas na Figura 26 (a), enquanto a Figura 26 (b) apresenta as velocidades angulares enviada e desenvolvida pelo robô móvel enquanto este navega em busca do seu objetivo, ou seja, o centro do corredor.

O erro de posição do robô em relação ao centro do corredor está apresentado na Figura 27 (a), enquanto a orientação dele em relação ao corredor pode ser vista na Figura 27 (b).

Os resultados a seguir são referentes à mesma simulação anterior, mas, desta vez, as variáveis de estado utilizadas pelo controlador são determinadas como mostrado no

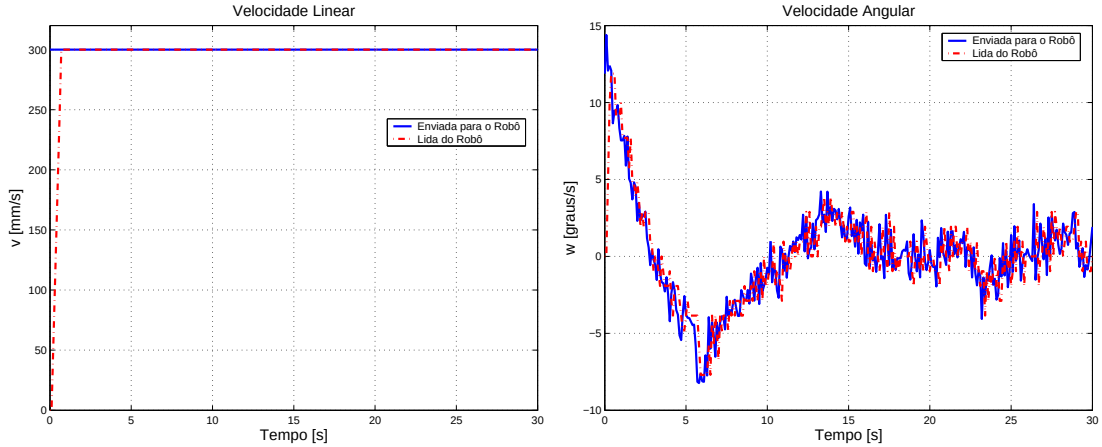
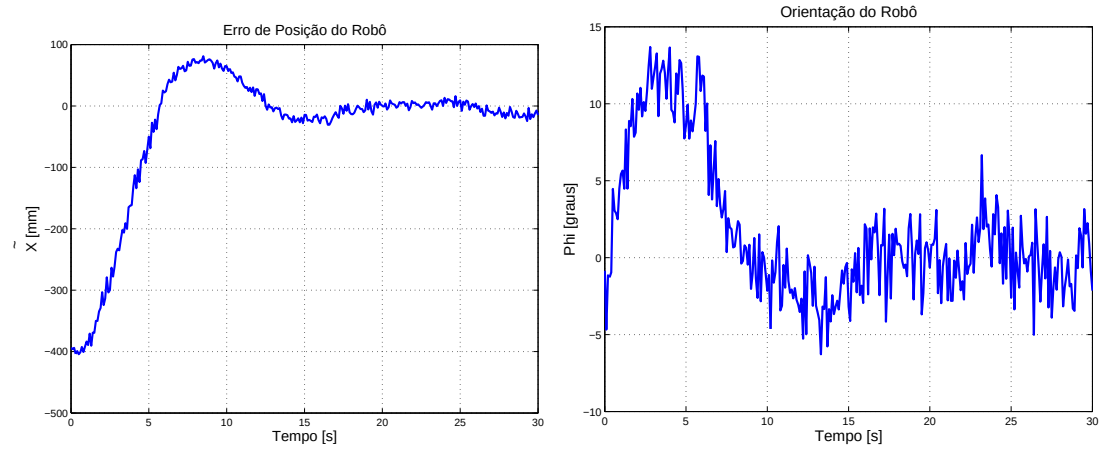


Figura 26: Velocidades linear (a) e angular (b).



(a) (b)

Figura 27: Erro de posição (a) e a orientação (b) do robô.

segundo método.

A Figura 28 mostra a trajetória descrita pelo robô enquanto o mesmo navegava no corredor.

Pode-se observar que, assim como na simulação anterior, o robô encontrou o meio do corredor e continuou navegando por ele em paralelo às paredes do corredor, que é o objetivo do controlador implementado.

Na Figura 29 (a) estão plotadas as velocidades lineares enviadas e desenvolvidas pelo robô. Já as velocidades angulares podem ser vistas na Figura 29 (b).

Apresenta-se na Figura 30 (a) o erro de posição do robô com relação ao centro do corredor, enquanto na Figura 30 (b) é apresentada a orientação do robô com relação ao corredor.

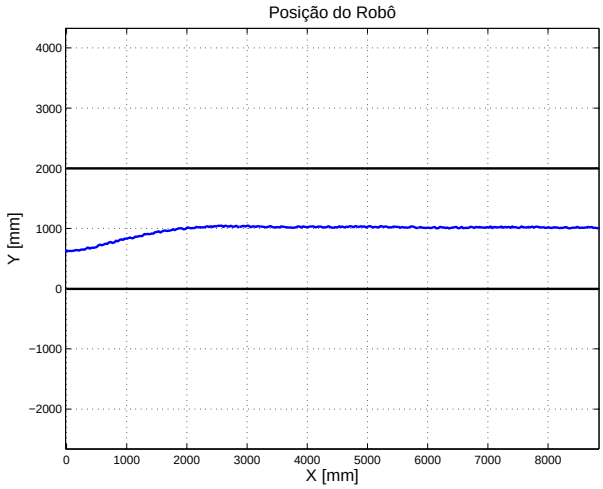


Figura 28: Trajetória descrita pelo robô.

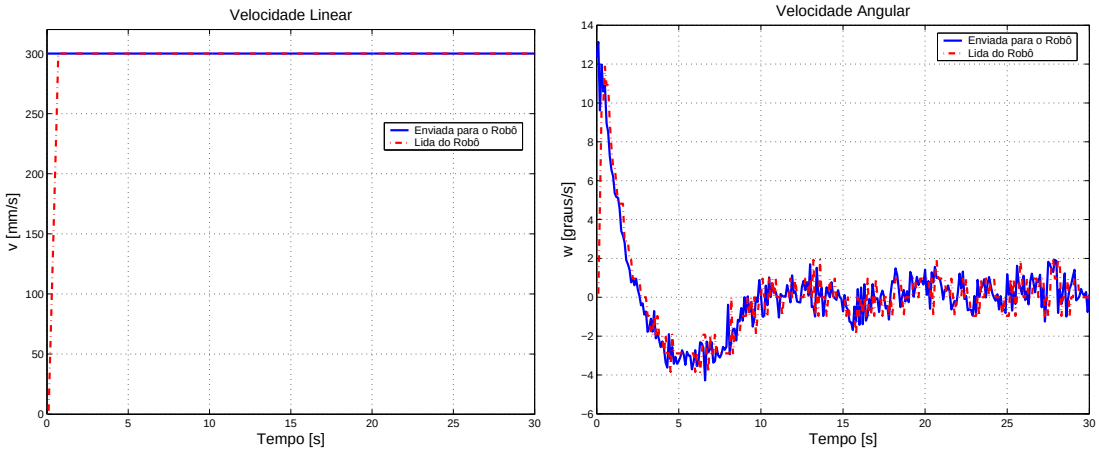
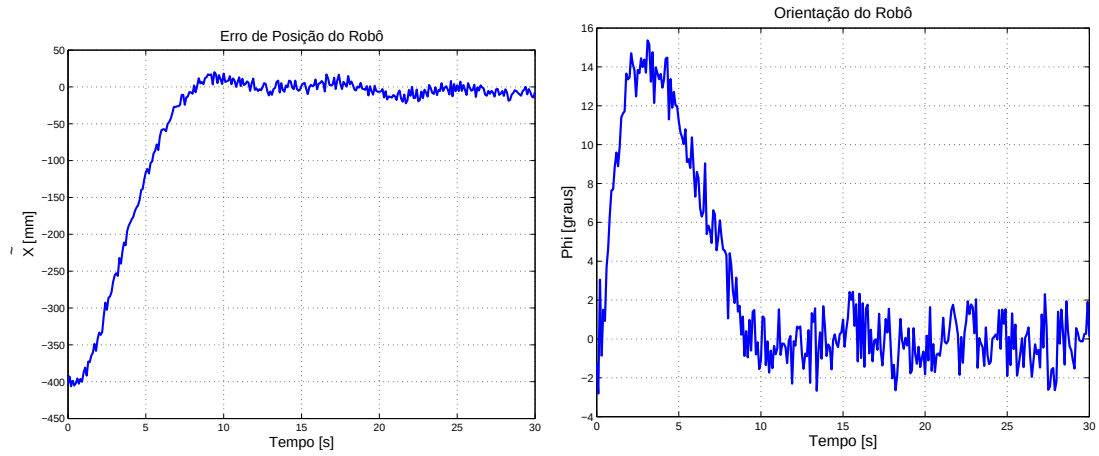


Figura 29: Velocidades linear (a) e angular (b).



(a) (b)

Figura 30: Erro de posição (a) e a orientação (b) do robô.

Pode-se notar, por observação dos gráficos apresentados nas Figuras 25 e 28, que, nestas duas simulações, o robô atingiu o seu objetivo, ou seja, encontrou a linha central do corredor e seguiu por ali, navegando em paralelo às paredes do mesmo. Porém, observando as Figuras 27 (a) e 30 (a), que representam os erros de posição do robô com relação ao centro do corredor para o primeiro e segundo métodos, respectivamente, nota-se que o robô encontra o centro do corredor mais rápido quando é utilizado o segundo método de cálculo das variáveis de estado. Observa-se, também, nas Figuras 26 (b) e 29 (b), que a velocidade angular, ω, nas duas simulações, oscila em torno de zero. Porém, na simulação realizada utilizando o segundo método, esta oscilação apresenta uma amplitude um pouco menor. Observando as Figuras 27 (b) e 30 (b), nota-se que o ângulo de orientação do robô, φ, também oscila em torno de zero e, assim como para a velocidade angular, a oscilação apresenta uma amplitude menor quando o ângulo φ é calculado a partir do segundo método. Por estes motivos, nos próximos testes realizados em corredores, e apresentados nesta Dissertação, será utilizada a segunda estratégia de determinação das variáveis de estado. Porém é importante citar que apesar do primeiro método de cálculo das variáveis de estado não estar sendo apresentado, os resultados obtidos quando o algoritmo foi implementado no robô obtiveram sucesso.

Também foi realizada uma simulação na qual, a partir de um determinado instante, as paredes do corredor deixam de ser paralelas, fazendo com que o corredor se afunile. Esta situação não foi prevista pelo método de cálculo das variáveis de estado, pois este foi desenvolvido para situações em que as paredes são sempre paralelas. A trajetória descrita pelo robô durante esta simulação está ilustrada na Figura 31.

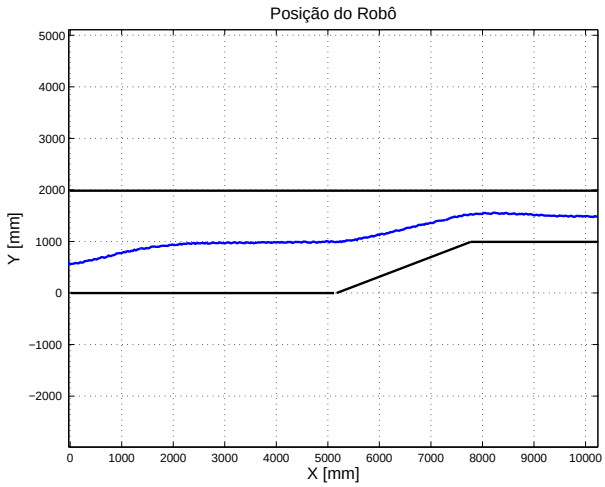
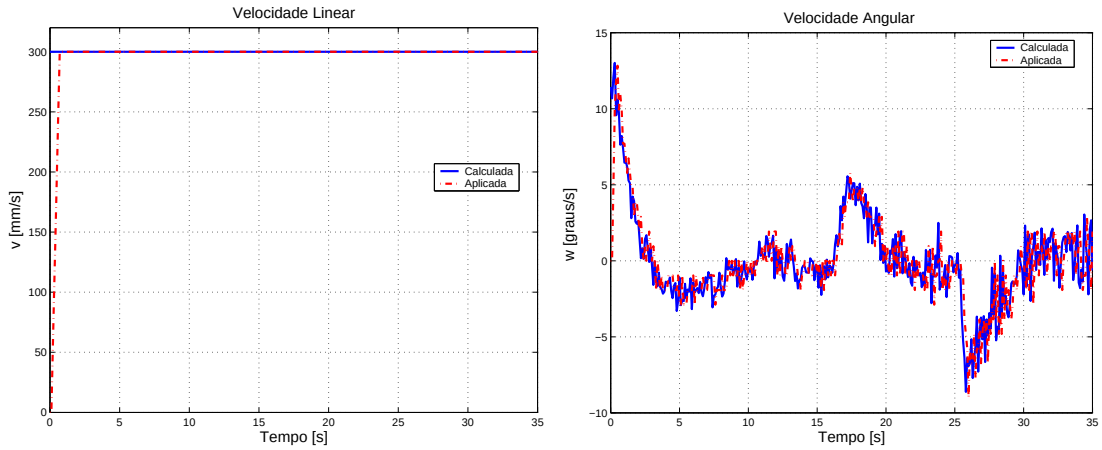


Figura 31: Trajetória descrita pelo robô.

Pode-se observar que mesmo diante de uma situação que não é prevista pelo con-

trolador de navegação em corredores, paredes não paralelas, o robô comportou-se bem e navegou pelo centro do corredor afunilado até encontrar novamente paredes paralelas e navegar pela linha média entre elas.

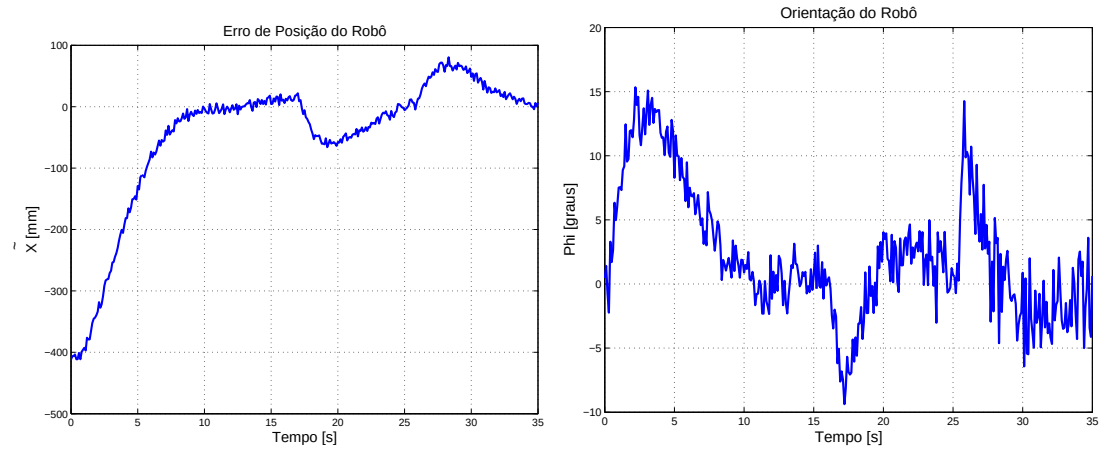
A Figura 32 (a) ilustra as velocidades lineares enviada ao robô e desenvolvida pelo mesmo, enquanto as velocidades angulares enviada e desenvolvida são apresentadas na Figura 32 (b).



(a) (b)

Figura 32: Velocidades linear (a) e angular (b).

O erro de posição em relação ao centro do corredor pode ser visualizado no gráfico apresentado na Figura 33 (a), enquanto a orientação do robô com relação ao corredor está representado na Figura 33 (b).



(a) (b)

Figura 33: Erro de posição(a) e a orientação do robô (b).

Pode-se observar na Figura 33 (a) que o erro de posição do robô tende a zero e, no instante em que o corredor começa a se afunilar, este erro aumenta em módulo. Assim, o

robô começa a navegar em busca do centro das novas paredes e, com isso, anular o novo erro de posição. Este erro estabiliza novamente em zero quando as paredes voltam a ser paralelas.

4 Navegação com Desvio de Obstáculos

"Nos momentos de crise, só a inspiração é mais importante que o conhecimento".

Albert Einstein

Neste Capítulo serão apresentados alguns controladores implementados para que o robô possa navegar em busca do seu destino e, caso sejam detectados obstáculos, desviar dos mesmos.

Será apresentado, inicialmente, um controlador não linear de posição final [27]. Em seguida, apresentam-se dois controladores para evitar obstáculos. O primeiro é baseado na idéia de desvio tangencial de obstáculos [12]. Este controlador nada mais é que o controlador de posição final com uma pequena alteração, de tal forma que o robô possa transpor os obstáculos encontrados e continuar navegando até atingir o ponto destino. O segundo procura um caminho por onde é possível a passagem do robô, e que possua a orientação mais próxima da posição final a ser alcançada pelo robô.

O modelo cinemático apresentado nas Equações 3.1 descreve, em coordenadas cartesianas, o movimento de um robô móvel, do tipo monociclo, em seu espaço de trabalho. Porém, para desenvolver o controlador de posição final, convém representar a posição do robô através de suas coordenadas polares [27].

Para isso, considera-se ρ e θ, respectivamente, o erro de posição do veículo e a orientação do mesmo em relação ao referencial inercial <g>. Isto permite escrever as equações cinemáticas em coordenadas polares. A Figura 34 mostra o robô, o sistema de coordenadas inercial e as variáveis utilizadas no modelo cinemático polar.

A Equação 4.1 mostra o modelo cinemático do robô escrito em coordenadas polares [27].

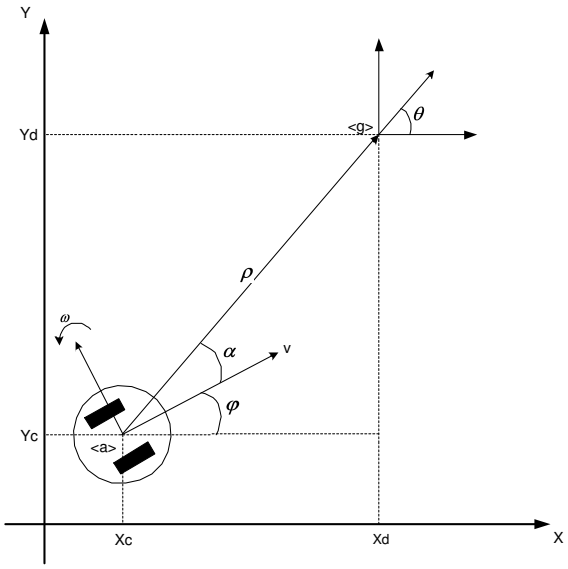


Figura 34: O robô e o sistema de coordenadas inercial.

$$\begin{cases} \dot{\rho} = -v\cos(\theta - \varphi) \\ \dot{\theta} = v\dfrac{\operatorname{sen}(\theta - \varphi)}{\rho} \\ \dot{\varphi} = \omega \end{cases} \tag{4.1}$$

Observa-se, na Figura 34, que $\theta = \alpha + \varphi$, ou seja, $\alpha = \theta - \varphi$, onde α expressa o ângulo entre o eixo de movimento do robô e o vetor de erro de posição ρ, isto é, o erro angular entre a orientação do robô e a direção do ponto que se deseja alcançar. Utilizando essa igualdade, a Equação 4.1 assume a forma

$$\begin{cases} \dot{\rho} = -v\cos(\alpha) \\ \dot{\alpha} = -\omega + v\dfrac{\operatorname{sen}(\alpha)}{\rho} \\ \dot{\theta} = v\dfrac{\operatorname{sen}(\alpha)}{\rho} \end{cases} . \tag{4.2}$$

É válido lembrar que o erro de posição ρ deve ser distinto de zero, visto que os valores de $\dot{\alpha}$ e $\dot{\theta}$ não estão definidos para erro de posição nulo.

As equações cinemáticas apresentadas em 4.2 são a base para o projeto dos controladores apresentados nesta Dissertação.

4.1 Controle Ponto a Ponto Sem Orientação Final

Este controlador tem como objetivo levar o robô de sua posição inicial até um ponto destino, previamente determinado, e fazer com que ele ali permaneça.

Para que o objetivo de controle seja alcançado, o controlador recebe os dados referentes à posição e orientação atuais do robô e, com isso, calcula os erros de posição e de orientação em relação ao ponto de destino. De posse destes erros, são calculadas as ações de controle (velocidades linear v e angular ω), as quais são enviadas ao robô para que o mesmo possa atingir o ponto alvo.

Com base na situação ilustrada na Figura 34, são calculados os erros de orientação e de distância. Nesta figura, ρ representa o erro entre a posição atual do robô e o ponto de destino. O desvio angular entre a orientação do robô e a direção do alvo é dado por α, enquanto φ representa a orientação do robô em relação ao referencial de destino.

As Equações apresentadas em 4.2 descrevem o modelo cinemático do robô em coordenadas polares. Para o controle ponto a ponto sem orientação final, assume-se $[\rho \quad \alpha]^T$ como vetor de variáveis de estado. Para análise de estabilidade foi considerada a seguinte função candidata de Lyapunov [27]

$$V(\rho, \alpha) = \frac{1}{2}\rho^2 + \frac{1}{2}\alpha^2. \tag{4.3}$$

cuja derivada temporal é dada por

$$\dot{V}(\rho, \alpha) = \rho\dot{\rho} + \alpha\dot{\alpha}. \tag{4.4}$$

Substituindo os valores de $\dot{\rho}$ e $\dot{\alpha}$ de 4.2 em 4.4, tem-se que

$$\dot{V}(\rho, \alpha) = -\rho v \cos(\alpha) + \alpha\left(-\omega + v\frac{\operatorname{sen}(\alpha)}{\rho}\right). \tag{4.5}$$

O primeiro termo da Equação 4.5 pode ser negativo se a velocidade linear v tiver a forma

$$v = v_{\max} \tanh(\rho)\cos(\alpha), \tag{4.6}$$

onde v_{max} representa o valor máximo, em módulo, da velocidade linear que será aplicada ao robô. De acordo com a escolha de v feita em 4.6, o segundo termo da Equação 4.5,

que aqui será chamado de $\dot{V}_2$, assume a forma

$$\dot{V}_2 = \alpha \left(-\omega + v_{\max} \frac{\tanh(\rho)}{\rho} \operatorname{sen}(\alpha) \cos(\alpha) \right). \tag{4.7}$$

O termo representado por $\dot{V}_2$ na Equação 4.7 pode ser negativo se a velocidade angular ω assumir a forma

$$\omega = k_\omega \alpha + v_{\max} \frac{\tanh(\rho)}{\rho} \operatorname{sen}(\alpha) \cos(\alpha) \quad \text{com} \quad \mathrm{k}_\omega > 0, \tag{4.8}$$

onde $|\omega_{\max}| = k_\omega \pi + 0.5 v_{\max}$. Com isto, $\dot{V}_2$ toma a forma

$$\dot{V}_2 = -k_\omega \alpha^2, \tag{4.9}$$

e, desta maneira, a derivada da função candidata de Lyapunov ($\dot{V}$), apresentada na Equação 4.4, pode ser reescrita como

$$\dot{V} = -v_{\max} \rho \tanh(\rho) \cos^2(\alpha) - k_\omega \alpha^2, \tag{4.10}$$

que é uma forma definida negativa. Isto significa que as variáveis de estado ρ e α convergem global e assintoticamente para zero com o decorrer do tempo, alcançando o objetivo de controle. A prova completa de estabilidade deste sistema pode ser vista em [27].

É válido mencionar que este controlador utiliza restrição das ações de controle para abordar o problema da saturação dos sinais de controle. Para a velocidade linear v, o módulo da mesma pode ser limitado, dado que $v_{\max}$ representa o maior valor da velocidade linear que será aplicada ao robô. Já para a velocidade angular ω, o valor máximo é limitado por $|\omega_{\max}| = k_\omega \pi + 0.5 v_{\max}$.

A partir das ações de controle, é possível escrever as equações de malha fechada do controlador de posição final proposto, sem considerar a orientação final, que são

$$\dot{\rho} = -v_{\max} \tanh(\rho) \cos^2(\alpha), \tag{4.11}$$

$$\dot{\alpha} = -k_\omega \alpha \tag{4.12}$$

e

$$\dot{\theta} = v_{\max} \frac{\tanh(\rho)}{\rho} \operatorname{sen}(\alpha) \cos(\alpha). \tag{4.13}$$

A Figura 35 mostra a estrutura do sistema de controle proposto. Nesta figura, x_r e x_c

são, respectivamente, as coordenadas do ponto destino e as coordenadas da posição atual do robô.

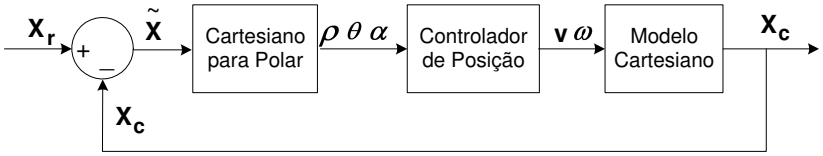


Figura 35: Diagrama de blocos do controlador de posição final [27].

Será apresentado, a seguir, o resultado de uma simulação na qual o robô parte de uma posição inicial (0, 0) e deve chegar num ponto localizado a 2000 mm à sua frente e a 2000 mm à sua esquerda, ou seja, na posição (2000 mm, 2000 mm).

Para este controlador de posição final, é necessário escolher os valores máximos das velocidades linear e angular calculadas pelo controlador. Os resultados apresentados nesta Dissertação foram obtidos com uma velocidade linear máxima de 300 mm/s e uma velocidade angular máxima de 90 $graus/s$. Para isso, fez-se $v_{max} = 0.3$ e $k_\omega = 0.45$.

É válido ressaltar que o controlador atua enquanto o erro de posição é maior que 50 mm já que o mesmo não pode ser igual a zero devido às restrições existentes sobre os valores de $\dot{\alpha}$ e $\dot{\theta}$ apresentados na Equação 4.2.

A Figura 36 mostra a trajetória descrita pelo robô da posição inicial até o mesmo atingir o ponto alvo.

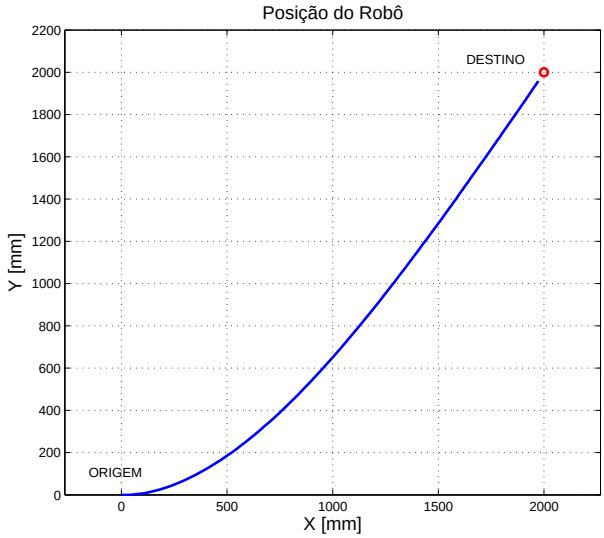
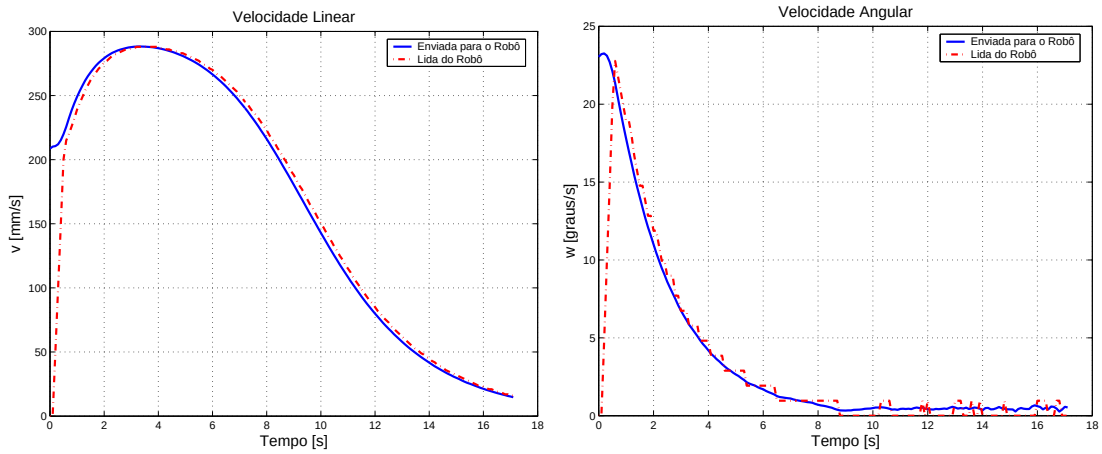


Figura 36: Trajetória descrita pelo robô.

Na Figura 37 (a) pode ser visualizada a velocidade linear, enquanto em 37 (b) está sendo mostrada a velocidade angular. Nestes dois gráficos, a velocidade, tanto angular como linear, que foi calculada pelo controlador, está plotada em linha contínua, enquanto a velocidade efetivamente desenvolvida pelo robô, recuperada via odometria, está representada como uma linha pontilhada.



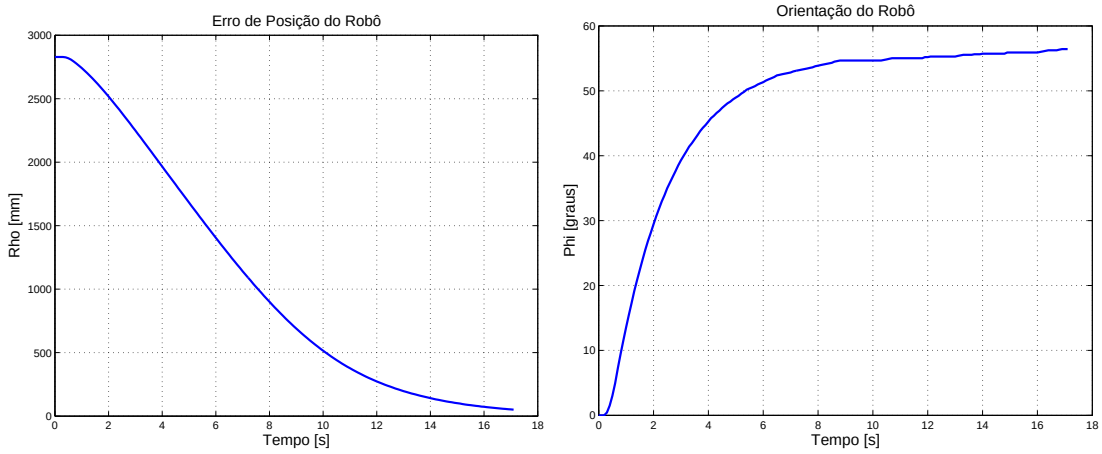
(a) (b)

Figura 37: Velocidades linear (a) e angular (b) do robô.

Pode-se observar na Figura 37 (a) que à medida que o robô se aproxima do ponto destino, sua velocidade linear decresce, da mesma forma que a velocidade angular tende a zero, quanto menor for o erro de orientação do robô com relação ao ponto alvo (Figura 37 (b)). Isso se dá devido ao fato da velocidade linear, v, depender diretamente do erro de posição do robô, assim como a velocidade angular ω ser função do erro angular.

Na Figura 38 podem ser vistos, em (a) e (b), respectivamente, o erro de posição do robô e a orientação do mesmo enquanto navega em direção ao alvo. Assim, como era esperado, o erro de posição tende a zero, ou seja, o robô se aproxima do ponto destino, apresentando um erro de posição de valor limitado.

Como o controlador de posição final implementado não leva em consideração a orientação final do robô, ou seja, é um controlador de posição final e não de orientação final, não é necessário que o erro de orientação do robô, quando este atinge o ponto destino, seja igual a zero.



(a) (b)

Figura 38: Erro de posição (a) e a orientação (b) do robô.

4.2 Controlador para Evitar Obstáculos

Nesta seção são descritos os dois algoritmos de desvio de obstáculos que foram implementados nesta Dissertação. Estes algoritmos foram implementados de acordo com a abordagem reativa, ou seja, não é feito nenhum pré-mapeamento do espaço onde o robô é posto para navegar. A proposta da primeira arquitetura de controle é apresentada em [12]. A idéia é fazer com que o robô realize trajetórias tangentes aos obstáculos detectados para, desta forma, transpô-los. A principal diferença das arquiteturas de controle propostas em [12] e aqui é a maneira como os obstáculos são detectados. Naquela, a detecção dos obstáculos é feita utilizando-se os sensores ultra-sônicos instalados a bordo do robô, enquanto nesta, os objetos são detectados por meio de um sensor *laser* acoplado ao robô. Tal sensor fornece, de uma única vez, 181 medidas de distância (de 0 a 180° em intervalos de 1°), cobrindo o semi-círculo em frente ao robô, como ilustrado na Figura 17. A segunda estratégia desenvolvida baseia-se na idéia de VFH (*Vector Field Histogram*) [6], ou seja, assim que obstáculos são detectados, faz-se a escolha do caminho que mais se aproxima da posição final que deve ser alcançada pelo robô móvel (em termos de direção), e que permita a passagem do robô com segurança.

4.2.1 Desvio Tangencial

A idéia deste controlador é fazer com que o robô desvie dos obstáculos encontrados realizando uma trajetória tangencial aos mesmos, alterando o destino temporariamente até que o caminho se torne livre novamente.

Para que o robô possa desviar dos obstáculos por ele detectados, o controlador gera um ângulo de desvio ϕ para que o alvo real seja rotacionado e, desta forma, possa ser criado um alvo virtual. Deste modo, enquanto o obstáculo não for ultrapassado, o robô navega em direção ao alvo virtual. Este ângulo é calculado em função da posição angular do obstáculo em relação ao robô. Assim, o ângulo de giro ϕ é determinado de tal forma que o alvo virtual é posicionado para que o robô possa desviar do objeto descrevendo uma trajetória tangente à sua borda.

Na Figura 39 é apresentada uma situação em que o robô encontra um obstáculo e o respectivo ângulo ϕ é gerado, para que haja um desvio tangencial.

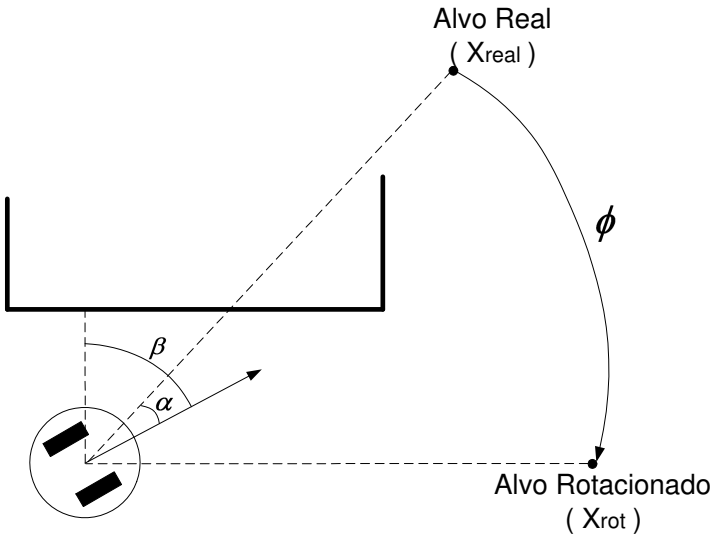


Figura 39: Determinação do ângulo ϕ [12].

Assim que um obstáculo é detectado pelo robô, é calculado o ângulo β entre a sua orientação atual e a parte do obstáculo mais próxima. Além disso, é necessário conhecer o ângulo entre a orientação do robô e a direção até o ponto destino, para que o ângulo de giro ϕ possa ser calculado. Estes ângulos podem ser vistos na Figura 39.

O ângulo β é determinado a partir das leituras do sensor *laser*. Entretanto não foram utilizadas todas as 181 medidas fornecidas pelo mesmo. As medidas foram tomadas de 0° até 180°, porém em intervalos de 10°. Foram realizados simulações e experimentos tomando todas as medidas. Porém, obtiveram-se experimentalmente melhores resultados quando as medidas foram tomadas de 10° em 10°. Já o ângulo α é obtido a partir da leitura dos *encoders* do robô. O ângulo de giro é, então, calculado conforme a equação

$$\phi = sign\,(\beta) \cdot (|\beta| - 90) - \alpha. \tag{4.14}$$

Rotações no sentido horário são consideradas negativas. Determinado o ângulo ϕ, é gerado um alvo virtual. Este alvo é obtido a partir de uma rotação de ϕ graus do alvo real, em torno do eixo-z, com relação ao referencial inercial do mundo. Isso é feito através da transformação

$$x_{rot} = \begin{pmatrix} \cos(\phi) & \text{sen}(\phi) \\ -\text{sen}(\phi) & \cos(\phi) \end{pmatrix} x_{real}, \tag{4.15}$$

onde x_{rot} é o alvo rotacionado e x_{real} é o alvo real a ser alcançado pelo robô.

Conhecida a posição do alvo rotacionado, faz-se, ainda, uma aproximação deste ponto para 1 m à frente do robô. Como se pode observar na Equação 4.6, a velocidade linear, calculada pelo controlador, é função da distância do robô ao alvo, ρ, e do erro angular entre a orientação do robô e este mesmo ponto, α. Assim, com esta aproximação, enquanto o robô se desvencilha do obstáculo e navega paralelo a ele, a velocidade linear do robô mantém um valor constante.

O controlador de desvio de obstáculo nada mais é que o controlador de posição final acrescido de um *loop* externo, responsável por gerar o alvo virtual caso um obstáculo seja detectado. Assim sendo, como o controlador de posição final é estável, o controlador de desvio tangencial de obstáculos também o é. Já que este último é um *loop* externo ao controlador de posição final e ocorre numa freqüência menor que o controlador de posição final. A Figura 40 mostra o diagrama de blocos do controlador para desvio de obstáculos, dentro dessa concepção. Nesta figura, x_d representa as coordenadas do ponto final a ser atingido pelo robô, x_c as coordenadas da posição atual do robô e x_r representa as coordenadas do alvo virtual gerado pelo controlador de desvio de obstáculos.

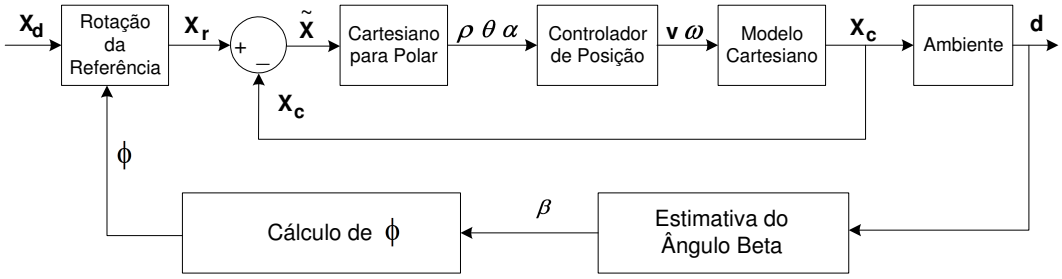


Figura 40: Diagrama de blocos do controlador de desvio de obstáculos [12].

Visto que o robô deve desviar dos obstáculos por ele encontrados sem tocá-los, é importante decidir qual a distância mínima a partir da qual os objetos serão ou não

considerados obstáculos. Para isso, deve-se levar em consideração a velocidade linear v do robô, bem como sua velocidade angular ω. Nesta Dissertação, o valor desta distância foi determinado através da realização de simulações e experimentos, e chegou-se à distância de 70 cm.

A seguir, são apresentados os resultados de uma simulação onde o robô parte da posição (0, 0) e deve chegar à posição (3000 mm, 4000 mm), tendo, agora, dois obstáculos circulares com 15 cm de raio nas posições (1500 mm, 1000 mm) e (1500 mm, 2500 mm). Esta simulação apresenta o desempenho do robô enquanto navega até o seu destino e desvia dos obstáculos com o uso do sensor *laser*. A Figura 41 mostra a trajetória descrita pelo robô durante a simulação.

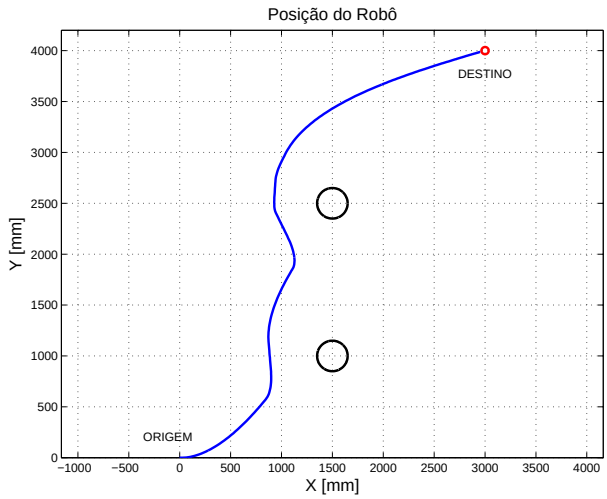


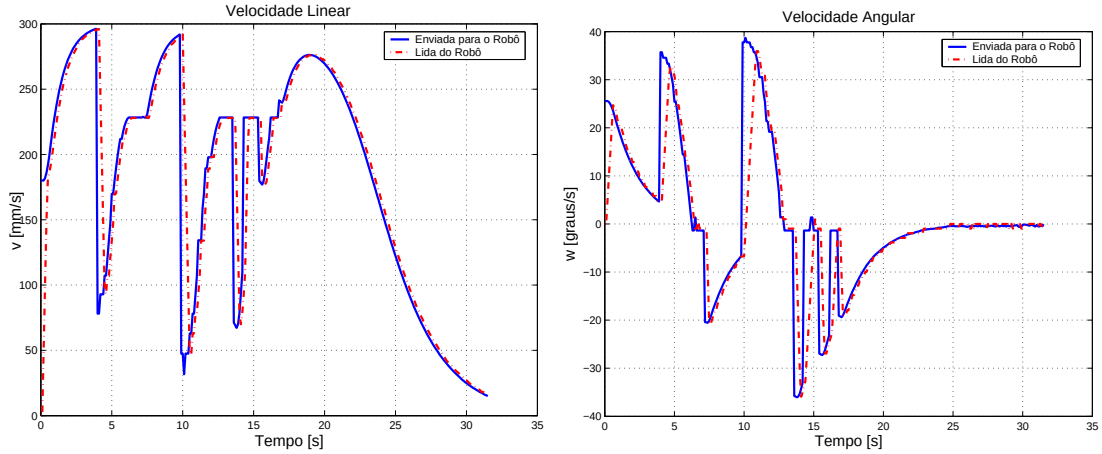
Figura 41: Trajetória descrita pelo robô.

Pode-se observar na Figura 41 que o robô navega de sua posição inicial até o ponto final desviando dos obstáculos que foram detectados pelo sensor *laser*, de acordo com o que foi proposto pelo controlador de desvio de obstáculos.

As velocidades linear e angular, calculadas e desenvolvidas pelo robô durante esta simulação são apresentadas nas Figuras 42 (a) e 42 (b), respectivamente. Pode-se observar nestas figuras que as variáveis de controle tendem para zero assim que o robô se aproxima do alvo, como foi mostrado na prova de estabilidade na Seção 4.1.

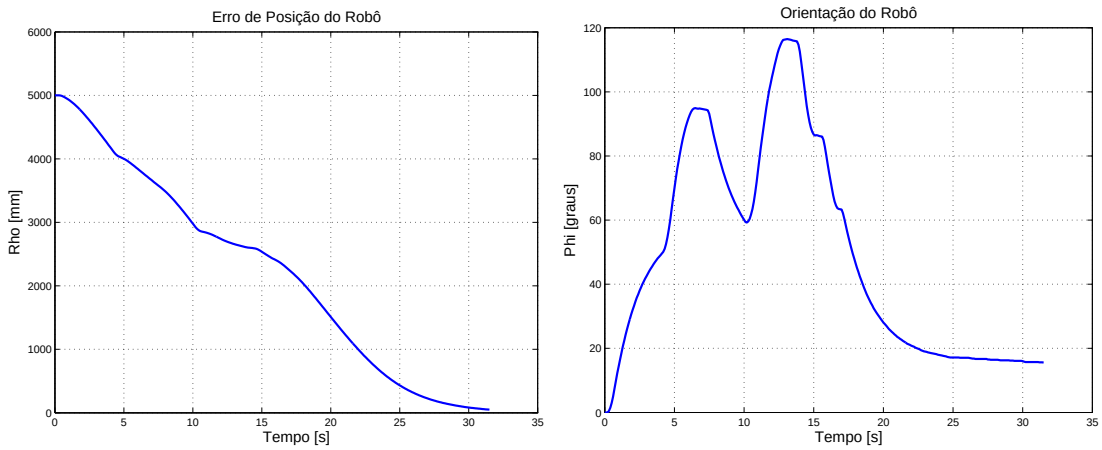
O erro de posição do robô até o ponto destino, e assim como sua orientação, podem ser vistos nas Figuras 43 (a) e 43 (b), respectivamente. Pode-se notar na Figura 43 (a) que o erro de posição ρ também tende a zero com o passar do tempo, o que indica que o robô chegou ao ponto destino.

Observa-se, através desta simulação, que o controlador de desvio tangencial de obstácu-



(a) (b)

Figura 42: Velocidades linear (a) e angular (b) do robô.



(a) (b)

Figura 43: Erro de posição (a) e a orientação (b) do robô.

los implementado possibilitou que o robô navegasse até atingir o ponto desejado mesmo quando obstáculos foram colocados no seu espaço de trabalho. Porém, como foram utilizados obstáculos circulares, não é possível perceber pela análise da trajetória descrita pelo robô, se o desvio realizado por ele foi tangencial ou não. Para que esta visualização seja possível, deve-se realizar simulações que contenham obstáculos como, por exemplo, paredes. Assim, o robô, quando as encontrar, poderá "segui-las", comprovando a funcionalidade do controlador proposto.

São apresentados, a seguir, os resultados obtidos da simulação deste mesmo controlador quando o robô navega com destino à posição (9000 mm, 5000 mm). Porém, ao invés de obstáculos circulares, existem, agora, paredes formando um ambiente em forma de "H". O objetivo desta simulação é mostrar que o algoritmo implementado realmente

proporciona um desvio tangencial do obstáculo detectado pelo sistema de sensoriamento existente a bordo do robô. A Figura 44 mostra a trajetória seguida pelo robô, neste caso.

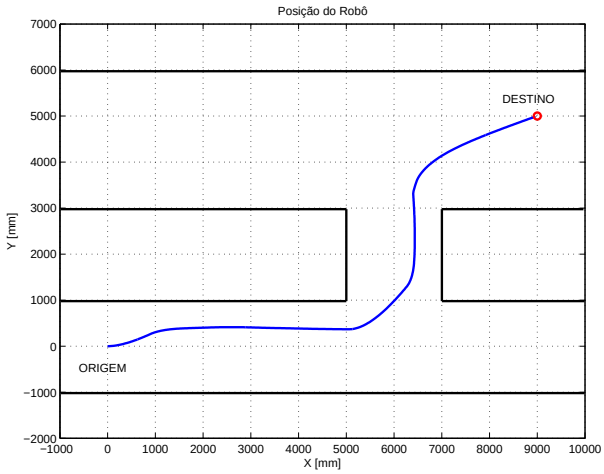
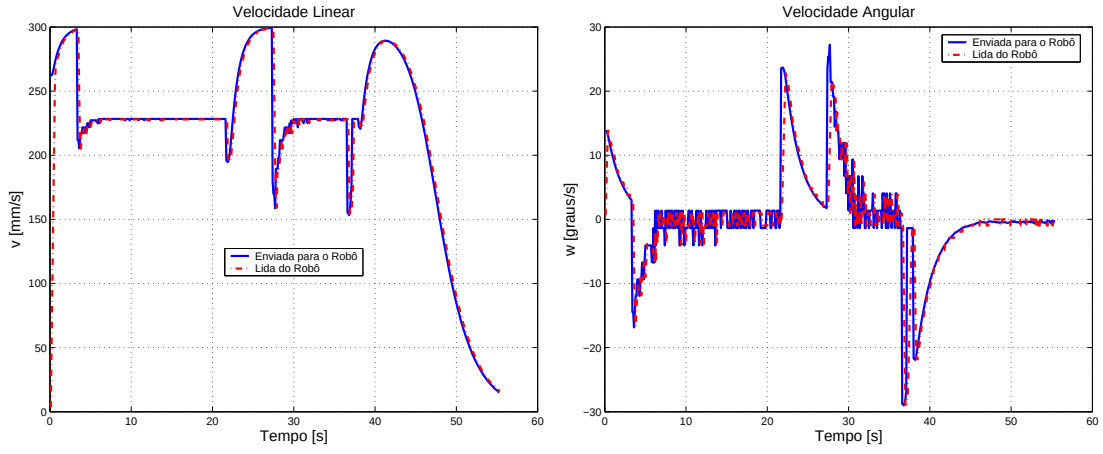


Figura 44: Trajetória descrita pelo robô.

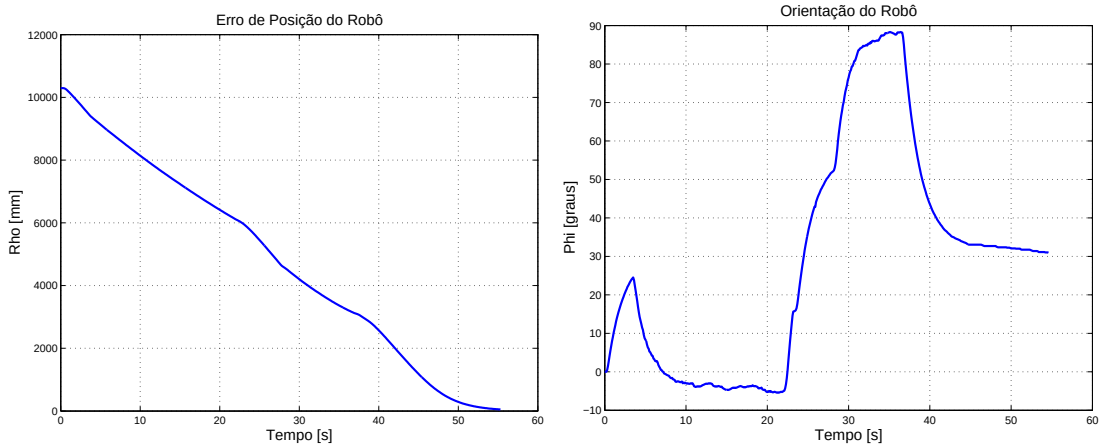
Pode-se notar, pela análise desta figura, que o robô, ao detectar o obstáculo, navega de forma paralela ao mesmo, comprovando o funcionamento do controlador de desvio tangencial implementado. A velocidade linear, calculada pelo controlador e desenvolvida pelo robô, durante esta simulação, é apresentada na Figura 45 (a), enquanto a velocidade angular pode ser vista na Figura 45 (b). O erro de posição do robô pode ser visto na Figura 46 (a), enquanto a Figura 46 (b) mostra a orientação do robô enquanto o mesmo navega em busca do ponto alvo.



(a) (b)

Figura 45: Velocidades linear (a) e angular (b) do robô.

Os resultados apresentados a seguir são de uma simulação onde o robô, mais uma vez, é posto para navegar em um ambiente com vários corredores. Porém, desta vez, o robô deverá descrever uma trajetória em forma de "U", iniciando a navegação na posição



(a) (b)

Figura 46: Erro de posição (a) e a orientação (b) do robô.

(0,0) e, tendo como alvo, a posição (1000 mm, 5000 mm). Além das paredes existem dois obstáculos circulares no ambiente por onde o robô móvel navega. Um destes obstáculos, cujo diâmetro é de 50 cm, está posicionado em (2500 mm, 0). O outro, que está na posição (6000 mm, 3000 mm), possui um diâmetro de 30 cm.

Pode-se observar na Figura 47, que representa a trajetória descrita pelo robô durante esta simulação, que o mesmo inicia o desvio tangencial da parede detectada, e, instantes depois, encontra o primeiro objeto circular. O robô, mesmo estando próximo da parede, faz uma leve curva para esquerda com o objetivo de desviar do obstáculo circular encontrado. A mesma situação acontece quando ele detecta o segundo obstáculo. Isto ocorre porque o ângulo β é calculado de acordo com a menor distância ao obstáculo detectado. Neste caso, quando os obstáculos circulares foram detectados com uma distância menor do que a parede, o desvio tangencial foi feito com relação aos mesmos.

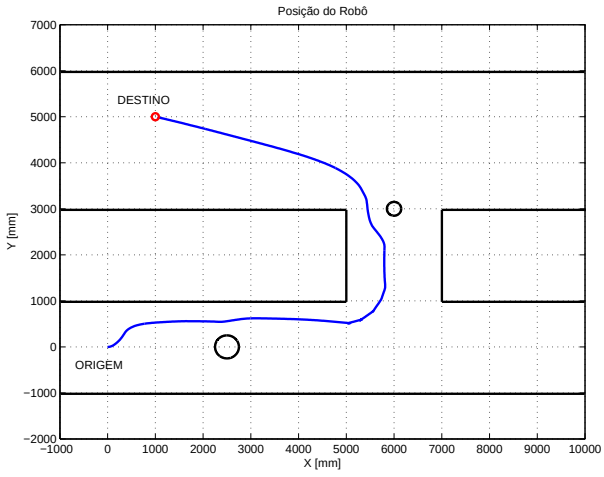
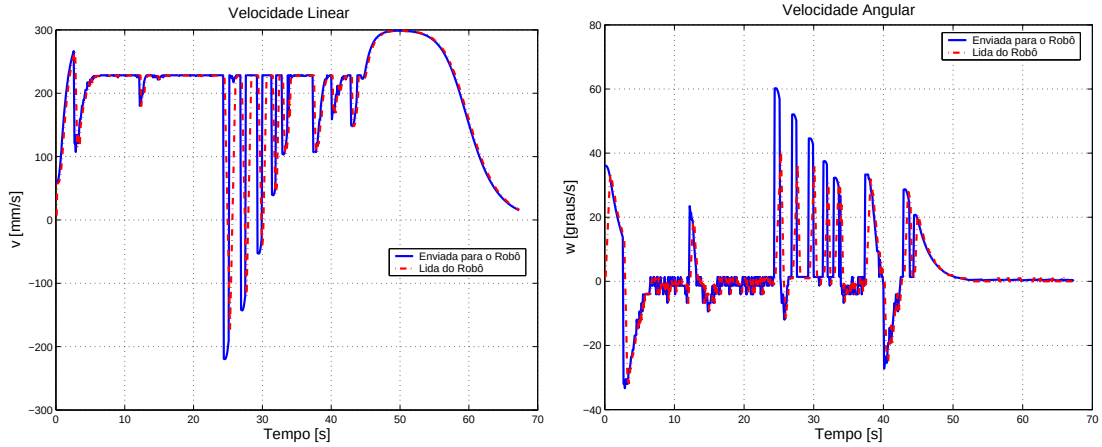


Figura 47: Trajetória descrita pelo robô.

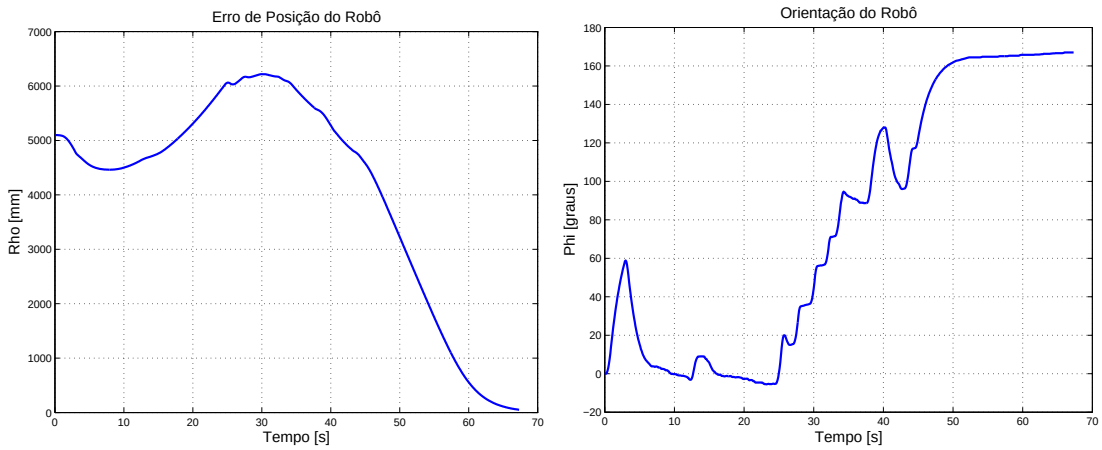
A Figura 48 (a) apresenta a velocidade linear do robô nesta simulação e a Figura 48 (b) mostra a velocidade angular do robô.



(a) (b)

Figura 48: Velocidades linear (a) e angular (b) do robô.

Nas Figuras 49 (a) e (b), respectivamente, são apresentados o erro de posição do robô e a orientação do mesmo. Nota-se, pela observação do gráfico que representa o erro de posição, que em um determinado instante este erro começa a aumentar. Isto representa o momento em que o robô começa a acompanhar a parede. Mesmo com este aumento no erro de posição, o controlador faz com que o robô continue navegando e desviando-se do obstáculo, mesmo que para isso ele tenha que se afastar do ponto destino. Ilustra-se, assim, a capacidade deste sistema de controle de evitar que o robô fique preso num mínimo local do erro de posição, conforme caracterizado em [12].



(a) (b)

Figura 49: Erro de posição (a) e a orientação (b) do robô.

4.2.2 Desvio Não Tangencial

Foi implementado, também, um outro controlador de desvio de obstáculos. Este controlador é independente do controlador de posição final, ou seja, não utiliza um laço externo para gerar uma posição virtual para o robô. O algoritmo proposto também é baseado nos dados fornecidos pelo sensor *laser*. As 181 medidas de distância que o sensor *laser* fornece são utilizadas para verificar se algum obstáculo foi encontrado. Mais uma vez, um objeto é considerado obstáculo se a sua distância até o robô for menor que 70 cm. Desta forma, é gerado um vetor de 181 posições (0° até 180°) formado por 0's e 1's, onde cada 0 representa que a medida fornecida pelo sensor numa determinada orientação é menor que 70 cm, ou seja, há um obstáculo naquela direção.

Após determinar onde se encontram os possíveis obstáculos, é determinado o caminho mais seguro por onde o robô pode passar. A região livre de obstáculos, que possuir uma abertura suficientemente grande para que o robô possa passar e que tiver o menor erro angular com relação ao ponto destino, é escolhida como o caminho a ser seguido.

A Figura 50 ilustra uma situação em que o robô encontra um obstáculo à sua frente. É possível observar que o caminho livre à esquerda do obstáculo é maior do que aquele à sua direita. Entretanto, o caminho à direita possui um menor erro angular com relação ao ponto destino. Logo, o caminho que deve ser escolhido para realizar o desvio deste obstáculo é o que se encontra à direita do robô.

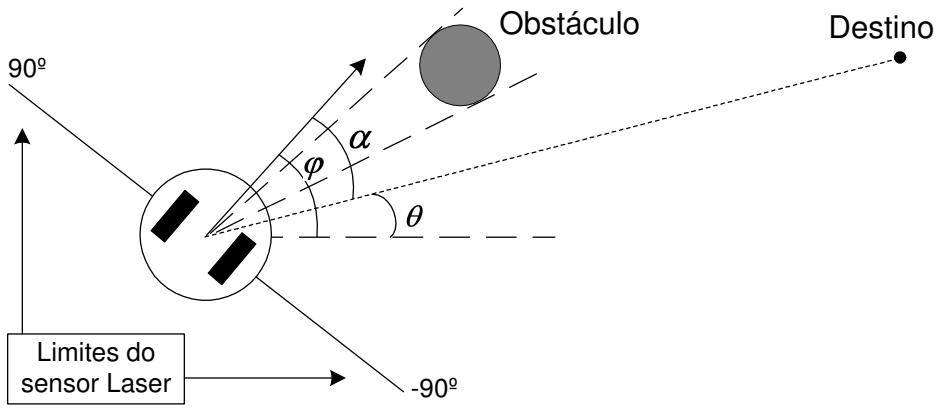


Figura 50: O robô e um obstáculo.

O ângulo de fuga do robô para evitar obstáculos é determinado como sendo o ângulo médio do caminho livre com base no referencial do robô. Toma-se, então, a região com menor erro angular com relação ao ponto final desejado. Este ângulo, ψ, e a menor distância a um obstáculo, d, serão utilizados para determinar as variáveis de controle v

e ω (velocidades linear e angular). O ângulo da região livre de obstáculos é determinado como sendo o ângulo médio do caminho livre. Isto pode ser visto na Figura 51.

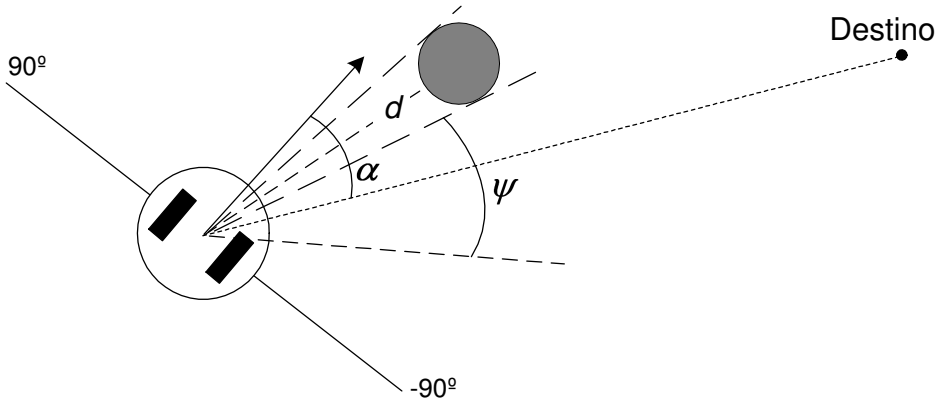


Figura 51: O ângulo ψ.

Assim, se um obstáculo for detectado, as velocidades linear e angular serão dadas por

$$v = k_v \tanh(d) \cos(\psi) \tag{4.16}$$

e

$$\omega = k_\omega \text{sen}(\psi) \tag{4.17}$$

onde k_v e k_ω são constantes positivas que determinam, respectivamente, as velocidades linear e angular máximas. É importante mencionar que as equações 4.16 e 4.17 foram escolhidas de maneira empírica e não foi realizado nenhum estudo da estabilidade das mesmas.

No controlador para desvio de obstáculos com trajetória tangencial, quando um obstáculo é encontrado, o alvo virtual é trazido para 1 m à frente do robô. Com isso, a velocidade linear, que é dada por $v = v_{max} \tanh(\rho) \cos(\alpha)$, tem seu valor limitado em $\tanh(1)\, v_{\max}$, ou seja, $0,7616 v_{\max}$. Como o $v_{\max}$ escolhido foi de 300 mm/s, a velocidade linear máxima quando um obstáculo é encontrado assume o valor de $228,48$ mm/s. Assim sendo, o valor de k_v na Equação 4.16 foi escolhido como sendo 228, para que, da mesma forma que o controlador com desvio tangencial, a velocidade máxima seja de 228 mm/s enquanto o robô executa a manobra de desvio do obstáculo detectado. Já a velocidade angular máxima foi de 90 $graus/s$, desta forma, o valor atribuído a k_ω foi 90.

A seguir, são apresentados os resultados de uma simulação, para o controlador de desvio de obstáculos com desvio pelo caminho mais próximo ao ponto alvo, na qual o robô parte da posição (0, 0) e, mais uma vez, deve chegar até a posição (3000 mm, 4000 mm)

com os obstáculos circulares de 15 cm de raio localizados nas posições (1500 mm, 1000 mm) e (1500 mm, 2500 mm).

A Figura 52 mostra a trajetória descrita pelo robô durante a simulação enquanto o mesmo navega em direção ao ponto destino e se desvencilha dos obstáculos por ele encontrados. Diferentemente do controlador de desvio tangencial de obstáculos apresentado anteriormente, neste caso o robô passa entre os dois obstáculos pois o espaço existente é suficiente para a manobra e, além disso, representa o caminho com o menor erro angular até o ponto destino. As velocidades linear e angular aplicadas ao robô durante a simulação são apresentadas nas Figuras 53 (a) e 53 (b), respectivamente.

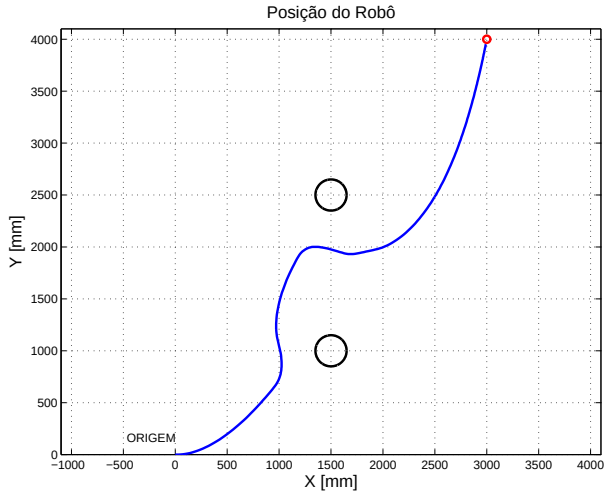
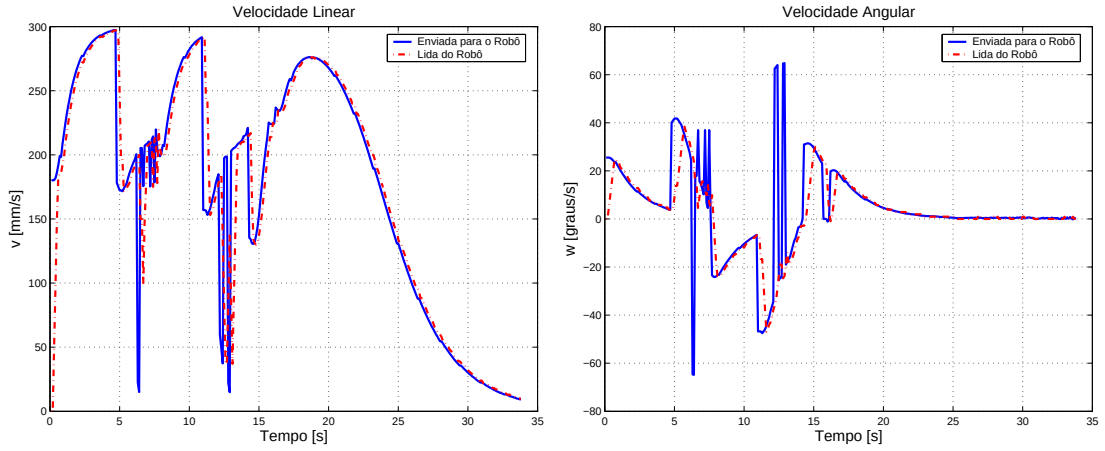


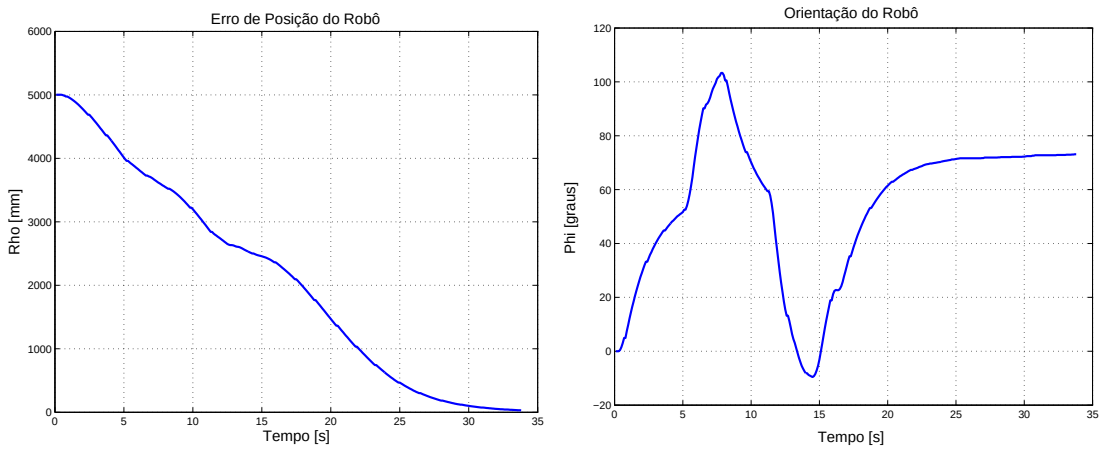
Figura 52: Trajetória descrita pelo robô.



(a) (b)

Figura 53: Velocidades linear (a) e angular (b) do robô.

A Figura 54 (a) mostra o erro de posição do robô durante esta simulação, enquanto a Figura 54 (b) mostra a orientação do mesmo.



(a) (b)

Figura 54: Erro de posição (a) e a orientação (b) do robô.

Este controlador não é indicado para o caso em que o robô deve navegar em busca de um alvo e existe uma parede em seu caminho e o mesmo deve contorná-la. Isso acontece devido à maneira como é calculado o ângulo ψ, que é o ângulo do caminho livre mais próximo ao ponto destino. Pode ocorrer de o ângulo da região livre de obstáculos, ψ, seja tal que o robô ao invés de seguir a parede contornando-a, retorne e, assim, fique vagando sem conseguir chegar ao ponto destino.

Foi feita, ainda, uma combinação do controlador para navegação em corredores com este último controlador de desvio de obstáculos, de forma que o robô, além de navegar procurando o centro do corredor, possa desviar de eventuais obstáculos que surjam no seu caminho. Neste caso, se não houver obstáculos, o controlador de navegação em corredores é ativado, caso contrário troca-se de controlador, ou seja, desabilita-se o controlador de navegação em corredores e habilita-se o controlador de desvio de obstáculos não tangencial. Porém, como o controlador de navegação em corredores não tem um ponto final desejado, fez-se uma pequena alteração no controlador de desvio de obstáculo para que fosse possível realizar esta comutação. Esta modificação foi a seguinte: no controlador de desvio de obstáculos que foi apresentado, o robô, ao detectar um ou mais obstáculos, determina o caminho livre que possui o menor erro de orientação com relação ao ponto final desejado. Ao ser implementado para desviar de obstáculos presentes em corredores, o controlador faz com que o robô busque o caminho livre com a maior abertura.

Será apresentada, agora, a trajetória descrita pelo robô, durante uma simulação, quando este foi posto para navegar em um corredor de 2 m de comprimento com um obstáculo de 30 cm de diâmetro posicionado a 3 m à frente da posição inicial do robô e 30 cm à sua esquerda (ver Figura 55).

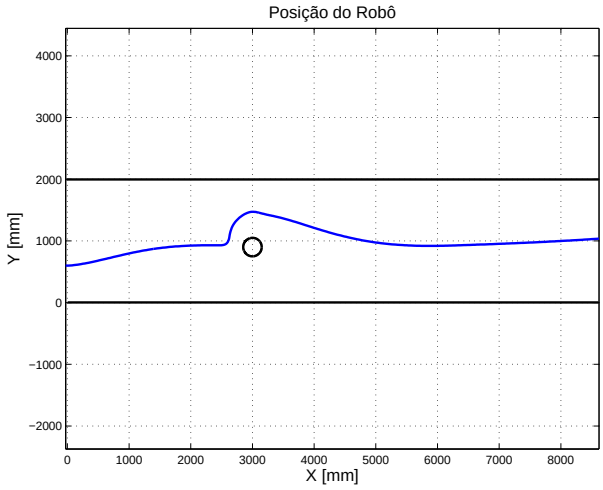


Figura 55: Trajetória descrita pelo robô.

Pode-se observar que o robô consegue encontrar o centro do corredor, e quando o obstáculo é detectado desvia-se dele e, após transpô-lo, retoma o comportamento para navegação em corredores e, novamente, consegue atingir o seu objetivo.

5 Resultados Experimentais

"A única possibilidade de descobrir os limites do possível é aventurar-se um pouco mais além deles, até o impossível".

Arthur C. Clarke

Para verificar o desempenho das arquiteturas de controle propostas, foram escritos programas em linguagem C++ para implementar os controladores apresentados. Foram executados vários testes nos quais o robô deveria chegar em um ponto de destino em ambientes de operação com quatro configurações diferentes.

Além das simulações apresentadas nos Capítulos 3 e 4, foram realizados alguns experimentos a fim de verificar o desempenho dos controladores implementados. Para isso, foi escrito um programa na linguagem C++ para cada controlador apresentado. Os testes foram realizados usando o robô móvel PIONEER 2-DX.

Em todos os casos, o foco dos experimentos mostrados nesta Dissertação é o uso de um sensor *laser* como forma de obter informação do ambiente

5.1 O Robô PIONEER 2-DX

Nesta Seção são apresentadas as características do robô móvel a rodas PIONEER 2-DX que foi utilizado para a realização dos experimentos mostrados nesta Dissertação. Tal robô faz parte de uma família de robôs móveis da *ActivMedia Robotics*, cuja arquitetura foi originalmente desenvolvida por Kurt Konolige, do *SRI International, Inc.*, e por pesquisadores da Universidade de Stanford.

5.1.1 *Hardware*

O PIONEER 2-DX possui 8 Sensores ultra-sônicos, uma câmara CCD Sony PTZ (*Pan, Tilt, Zoom*), que não foi utilizada nos experimentos e um sensor de varredura *laser*. Os dados provenientes dos sensores do robô são atualizados a cada 100 ms.

A estrutura básica deste robô, apresentado na Figura 56, é composta do microcontrolador 88C166 fabricado pela Siemens, de 20 MHz, 32 kBytes de memória FLASH ROM e 32 kBytes de RAM dinâmica [28]. Existe, ainda, um computador de bordo com um processador PENTIUM MMX, de 233 MHz bem como uma placa com conexões disponíveis para teclado, mouse, monitor e conector de rede.

Figura 56: Robô móvel a rodas PIONEER 2-DX.

5.1.2 *Software*

A programação do robô pode ser feita de duas formas: programar diretamente no microcontrolador 88C166 ou através de interfaces de alto nível como ARIA ou SAPHIRA. Os programas implementados nesta Dissertação foram desenvolvidos em C++, utilizando

a interface ARIA. Desta forma, as rotinas estabelecem uma conexão tipo cliente-servidor com um servidor, que pode ser o simulador *SRIsim* ou o P2OS (Sistema Operacional executado na ROM do microcontrolador 88C166)[29].

A interface ARIA (*ActivMedia Robotics Interface for Application*) é uma interface de programação de código aberto com orientação a objetos para aplicações com robôs móveis da ActivMedia. Ela provê o suporte necessário para controle do robô. O ARIA foi projetado para servir como uma base versátil para programas de alto nível. A interface ARIA é distribuída sob Licença Pública GNU, significando que se algum trabalho que utilize ARIA é distribuído, todo o código fonte do mesmo deve ser disponibilizado em conjunto.

Já o SAPHIRA é um ambiente de alto nível, tendo ARIA como base. Assim, as classes e funções do ARIA podem ser utilizadas quando o programa é desenvolvido em SAPHIRA. A interface SAPHIRA é desenvolvida e mantida pela SRI e não pela ActivMedia. SAPHIRA não possui código aberto.

Podem ser citadas algumas vantagens de se utilizar o ARIA. Dentre elas está a sua constante atualização (*software* e documentação), o fato de ser completamente *open-source* e a grande disponibilidade de suporte, principalmente através de listas de discussões.

5.2 Experimentos

Serão apresentados, nesta Seção, alguns dos experimentos que foram realizados. Os ambientes que foram criados para estes experimentos tentaram reproduzir, da melhor maneira possível, os mundos gerados pela ferramenta *Mapper* que foram usados para as simulações apresentadas nos Capítulos 3 e 4. Desta forma, pode-se fazer uma comparação dos resultados obtidos a partir das simulações com os obtidos experimentalmente. Assim como nas simulações, os dados são enviados ao robô a cada 100 ms. Em outras palavras, o laço de controle implementado é de 100 ms.

5.2.1 Experimento 1

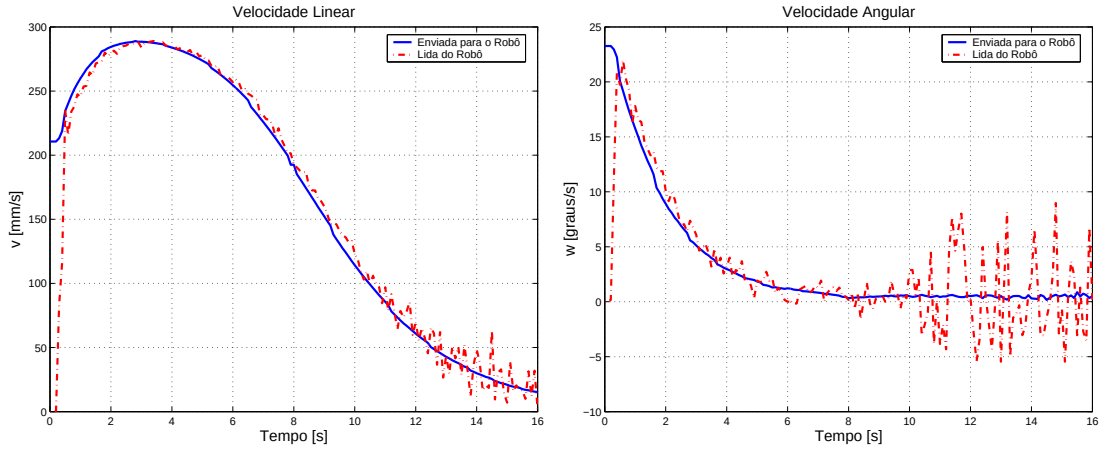
Este experimento visa mostrar os resultados do controlador de posição final implementado. O robô parte de uma posição inicial e tem que alcançar um ponto localizado na posição (2000 mm, 2000 mm). Porém, para poder observar o efeito da dinâmica, que não foi considerado pelos controladores e nem pelo simulador utilizado, este experimento

foi realizado em duas partes. Na primeira, o robô está suspenso, ou seja, ele não se move, apenas suas rodas giram e as leituras dos encoders são utilizadas para determinar sua "posição". Já na segunda parte, o experimento é efetivamente realizado, ou seja, o robô é colocado para navegar em busca do ponto final. Os resultados destas duas etapas estão apresentados a seguir.

- **Robô Suspenso:**

Nesta parte deste experimento são mostradas apenas as velocidades linear e angular calculadas pelo controlador e as que foram lidas a partir dos *encoders* presentes nas rodas do robô. Faz-se isso pois não há sentido em mostrar a posição e a orientação do robô móvel, pois, durante a realização deste experimento, o mesmo estava suspenso e não se moveu.

As velocidades lineares enviadas para o robô e as que foram lidas dos *encoders* do mesmo estão apresentadas no gráfico da Figura 57 (a). Já a Figura 57 (b) mostra as velocidades angulares enviadas ao robô e lidas do mesmo.



(a) (b)

Figura 57: Velocidades linear (a) e angular (b) do robô.

- **Robô no Chão:**

São apresentados, agora, os resultados obtidos para o experimento do controlador de posição final com o robô apoiado no chão. Como já foi mencionado, o robô parte da posição (0, 0) e deve chegar na posição (2000 mm, 2000 mm). A Figura 58 mostra a trajetória do robô durante este experimento. Nota-se, neste gráfico, que o robô não chega

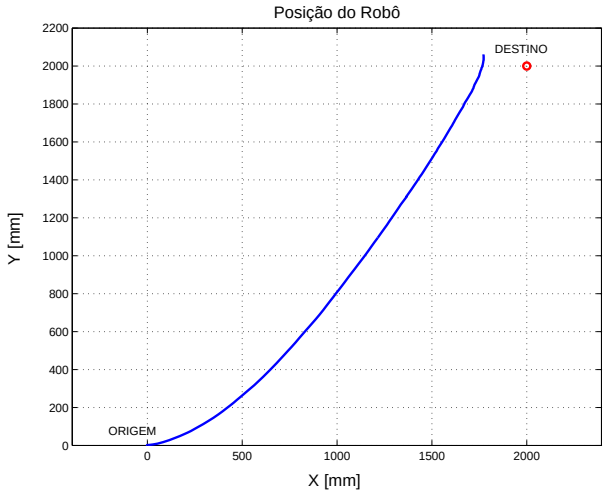
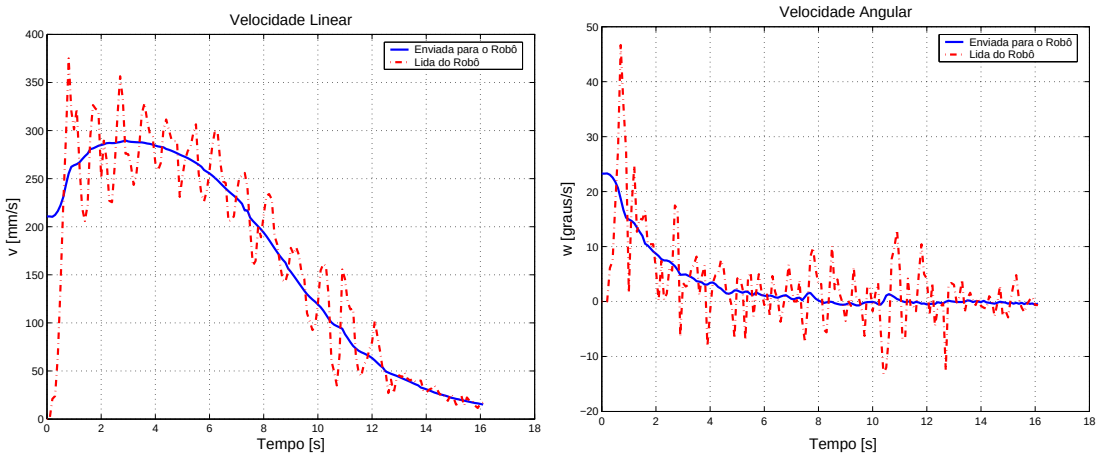


Figura 58: Trajetória descrita pelo robô.

exatamente ao ponto desejado. Este fato acontece devido aos erros de odometria do robô, que vão se acumulando com o passar do tempo.

A Figura 59 (a) mostra a velocidade linear do robô enquanto a Figura 59 (b) apresenta a velocidade angular do mesmo.

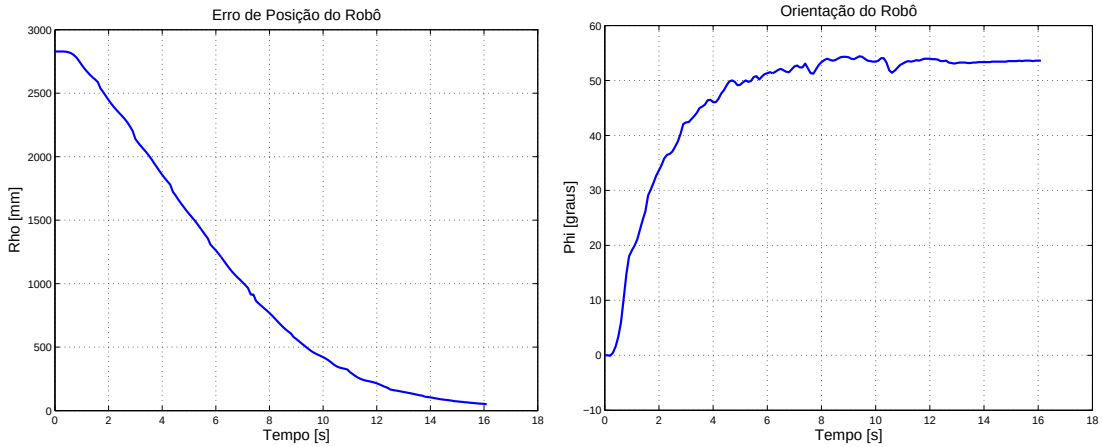


(a) (b)

Figura 59: Velocidades linear (a) e angular (b) do robô.

O erro de posição do robô e a orientação do mesmo são apresentados na Figura 60 (a) e (b) respectivamente.

Observa-se que no experimento em que o robô está suspenso, os resultados aproximam-se dos obtidos na simulação e as velocidades obtidas através das leituras dos *encoders* do robô possuem valores próximos aos calculados pelo controlador. Já no experimento em que o robô encontra-se apoiado no chão, navegando efetivamente, estas velocidades apresentam valores bastante superiores aos limites impostos pelo controlador, em alguns



(a) (b)

Figura 60: Erro de posição (a) e a orientação (b) do robô.

momentos. Entretanto, como se pode ver na Figura 60, o objetivo é alcançado, o que leva a crer que as discrepâncias nos valores medidos de velocidades devem-se aos erros de odometria e o fato do controlador implementado não considerar a dinâmica do robô móvel utilizado.

5.2.2 Experimento 2

Neste experimento é utilizado o controlador para navegação em corredores. O robô foi posto para navegar num corredor de 2 m de largura por um tempo de 30 s. A Figura 61 mostra a trajetória seguida pelo robô.

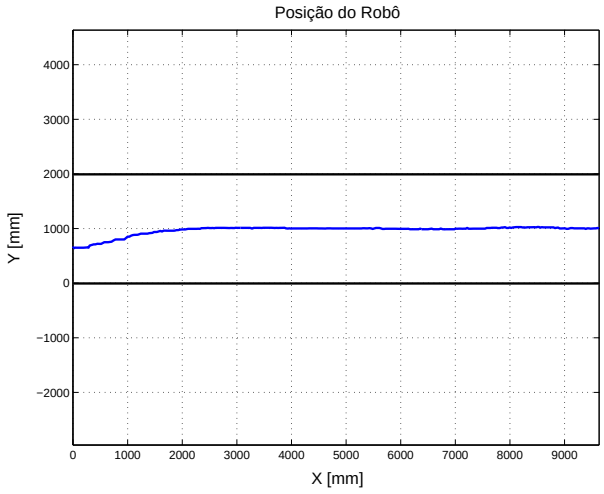


Figura 61: Trajetória descrita pelo robô.

As velocidades lineares enviada ao robô e lida dos *encoders* estão ilustradas na Figura 62 (a) enquanto a Figura 62 (b) apresenta as velocidades angulares enviada ao robô e

a que foi executada pelo mesmo.

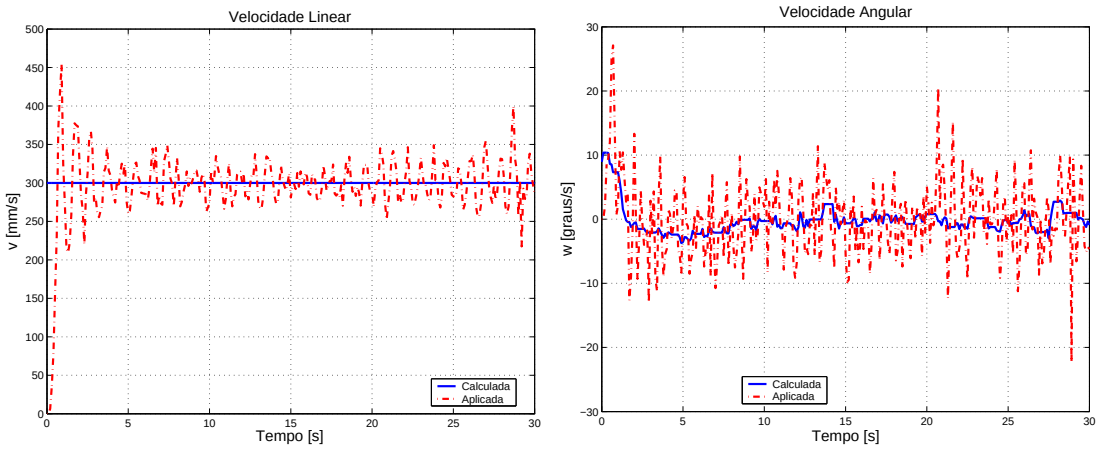
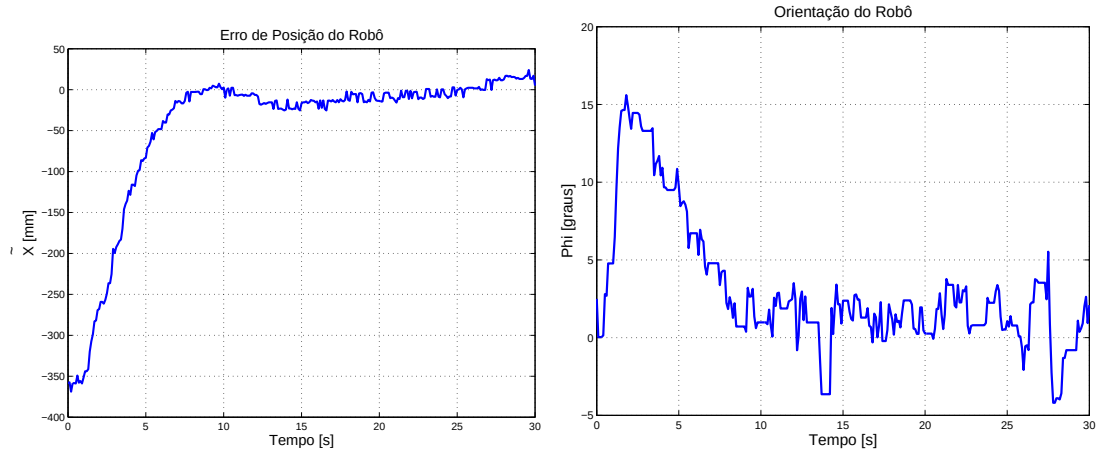


Figura 62: Velocidades linear (a) e angular (b).

O erro de posição do robô e a sua orientação em relação ao objetivo estão mostrados nas Figuras 63 (a) e 63 (b), respectivamente.



(a) (b)

Figura 63: Erro de posição (a) e a orientação (b) do robô.

Neste experimento, como era esperado, o robô atingiu o centro do corredor e seguiu navegando paralelamente às paredes do mesmo, ou seja, os erros de posição e de orientação tendem a zero com o decorrer do tempo.

5.2.3 Experimento 3

Neste experimento, utiliza-se o controlador de desvio tangencial de obstáculos. O robô é colocado para navegar de um ponto inicial até a posição (3000 mm, 4000 mm).

Porém existem 2 obstáculos de 30 cm de diâmetro posicionados em (1500 mm, 1000 mm) e (1500 mm, 2500 mm).

Pode-se observar, na Figura 64, que o robô consegue desviar dos obstáculos e aproximar-se do ponto final desejado.

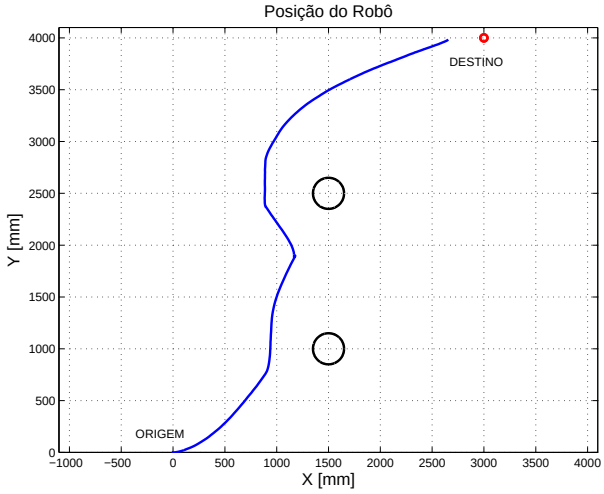
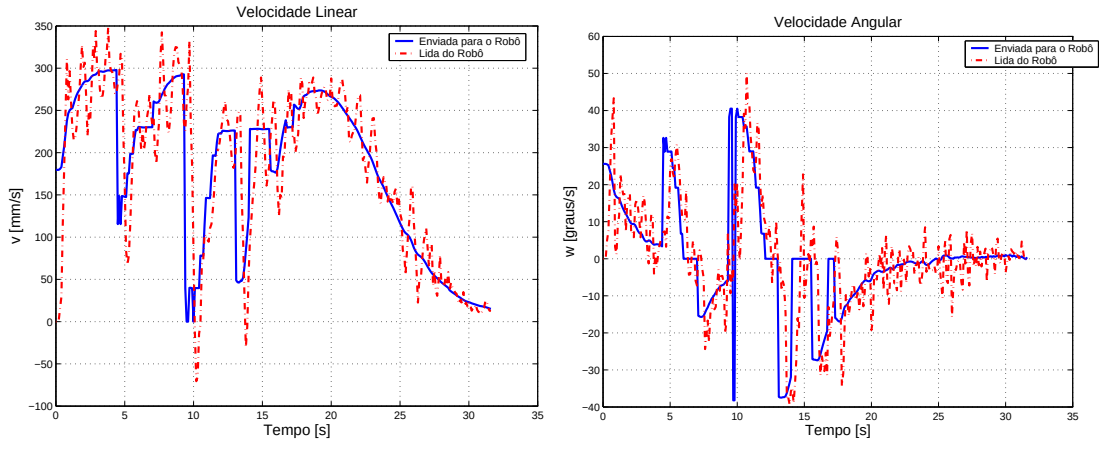


Figura 64: Trajetória descrita pelo robô.

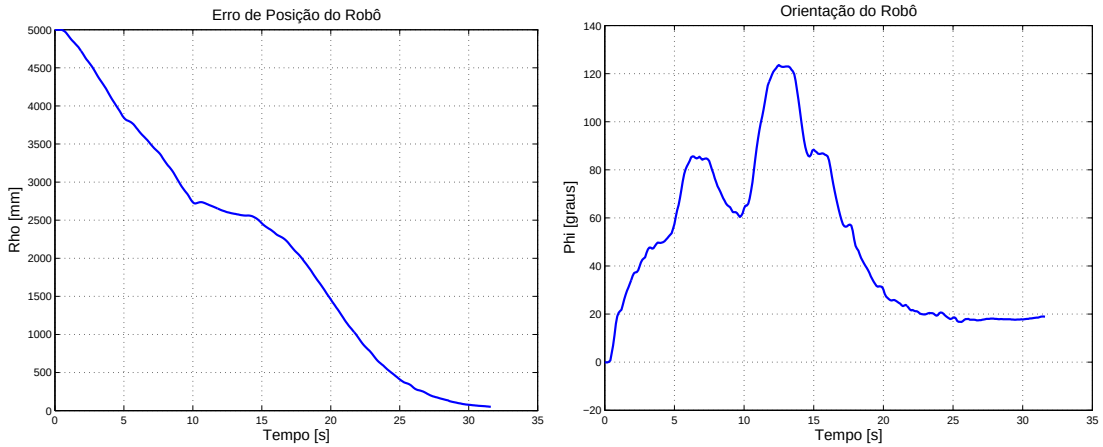
As Figuras 65 (a) e 65 (b) mostram, respectivamente, as velocidades lineares e angulares enviadas e lidas do robô. Por sua vez, o erro de posição do robô e a orientação do mesmo, enquanto navega em busca do ponto final, são mostrados nas Figuras 66 (a) e 66 (b).



(a) (b)

Figura 65: Velocidades linear (a) e angular (b) do robô.

Observa-se, com este experimento, que o robô navega em busca do seu ponto alvo, e quando depara com os obstáculos, realiza as manobras necessárias para evitá-los e alcançar o ponto final desejado.



(a) (b)

Figura 66: Erro de posição (a) a e orientação (b) do robô.

5.2.4 Experimento 4

Assim como no Experimento 3, este utiliza o controlador de desvio tangencial de obstáculos. Neste experimento, o robô foi posto para navegar num ambiente composto de vários corredores em forma de "H". O robô móvel que, inicialmente encontra-se na posição (0, 0), deve navegar em busca da posição (9000 mm, 5000 mm) desviando, de maneira tangencial, dos obstáculos (paredes) por ele detectados. A trajetória descrita pelo robô, enquanto o mesmo navegava em busca do ponto destino, está representada na Figura 67.

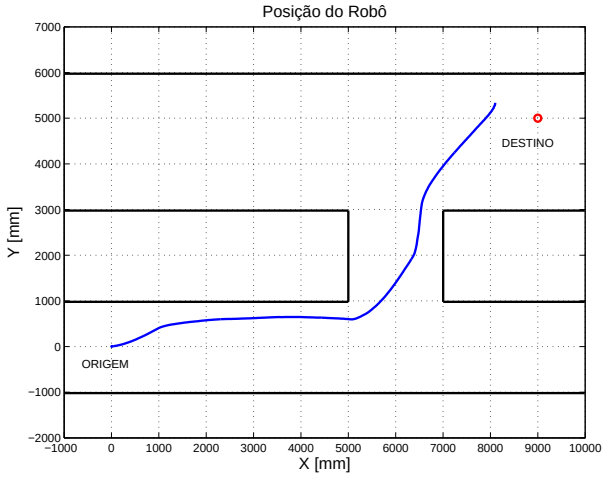
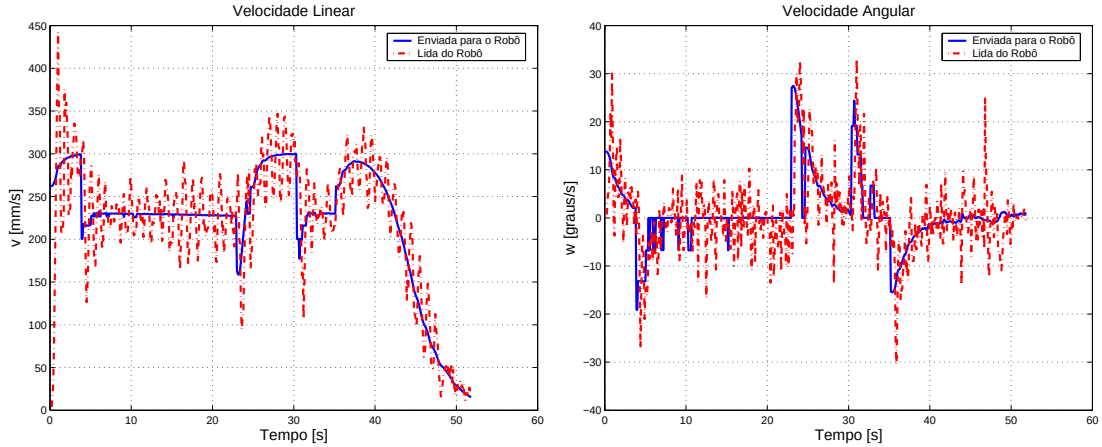


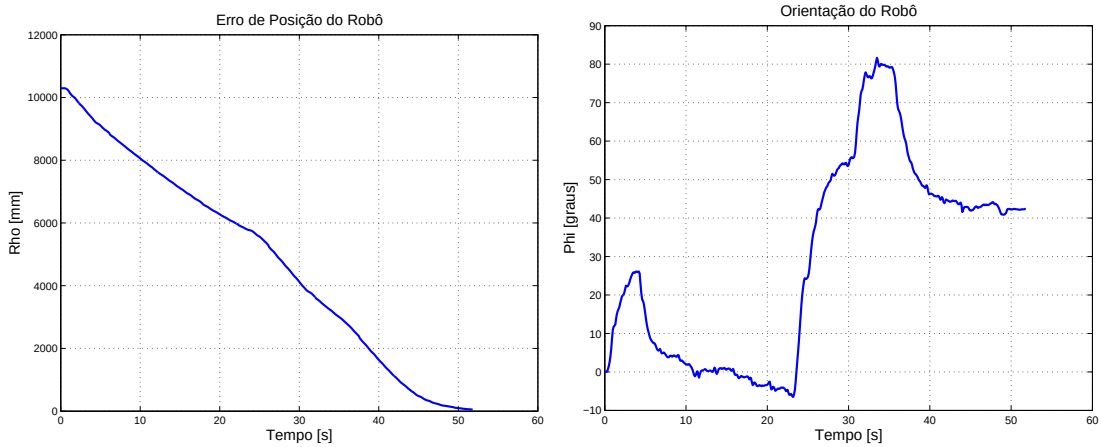
Figura 67: Trajetória descrita pelo robô.

As velocidades linear e angular enviadas e lidas do robô estão apresentadas nas Figuras 68 (a) e 68 (b), respectivamente, enquanto o erro de posição e a orientação do robô estão ilustrados, respectivamente na Figura 69 (a) e 69 (b).



(a) (b)

Figura 68: Velocidades linear (a) e angular (b) do robô.



(a) (b)

Figura 69: Erro de posição (a) e a orientação (b) do robô.

O robô navega em busca do seu ponto destino e, desta vez, os obstáculos que ele encontra são paredes, as quais são evitadas, porém de forma tangencial, comprovando a funcionalidade do controlador de desvio tangencial de obstáculos implementado.

5.2.5 Experimento 5

Novamente é utilizado o controlador de desvio tangencial de obstáculos, porém, desta vez, o robô é posto para navegar num ambiente com vários corredores em forma de "U", saindo da posição (0, 0) e chegando na posição (1000 mm, 5000 mm). Foram adicionados, ainda, dois obstáculos circulares no ambiente. Um deles, de 50 cm, de diâmetro está posicionado em (2500 mm, 0), o outro, cujo diâmetro mede 30 cm, localiza-se em (6000 mm, 3000 mm).

A Figura 70 mostra o caminho percorrido pelo robô enquanto o mesmo navega em busca do ponto destino e desvia dos obstáculos encontrados.

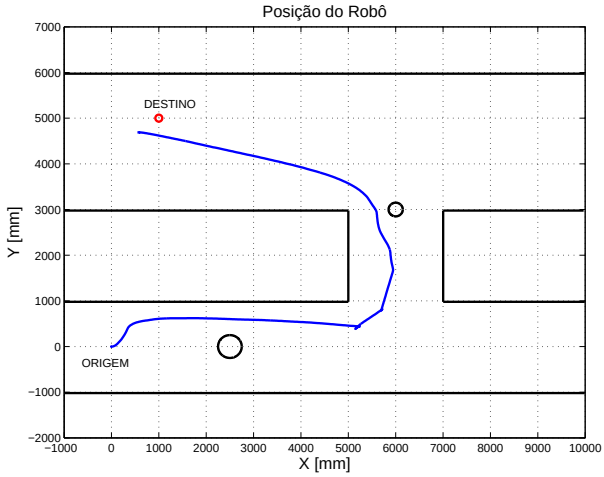
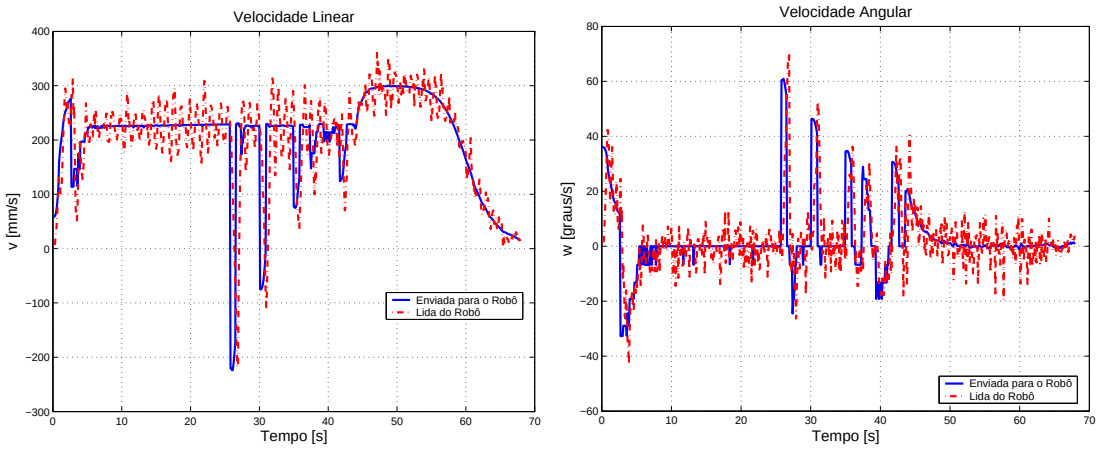


Figura 70: Trajetória descrita pelo robô.

As velocidades linear enviada ao robô e desenvolvida pelo mesmo estão apresentadas na Figura 71 (a), enquanto na Figura 71 (b) podem ser visualizadas as velocidades angulares.

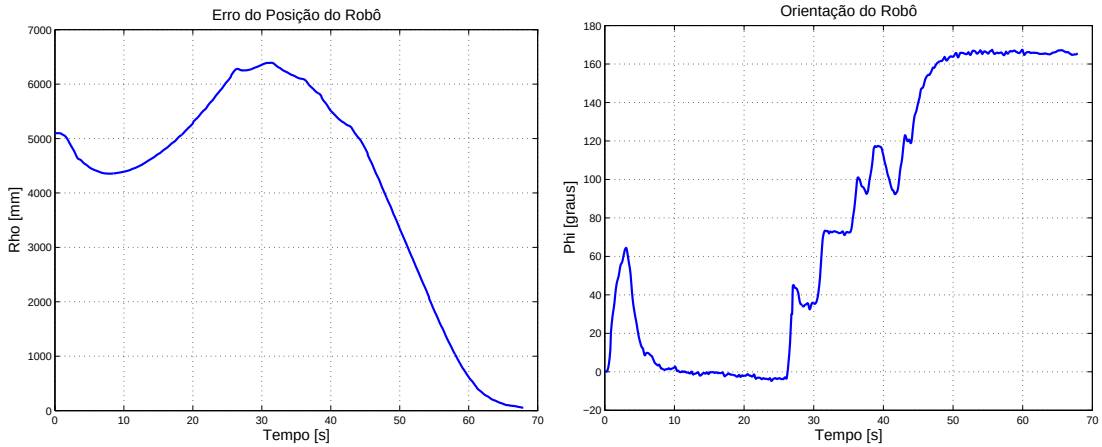


(a) (b)

Figura 71: Velocidades linear (a) e angular (b) do robô.

O erro de posição e a orientação do robô a cada instante estão ilustrados, respectivamente, na Figura 72 (a) e Figura 72 (b).

O robô encontrou, durante a realização deste experimento, paredes e objetos circulares como obstáculos. Nota-se que, mesmo quando está desviando da parede de forma tangencial e encontra um outro obstáculo, ele conseguiu desviar deste novo obstáculo.



(a) (b)

Figura 72: Erro de posição (a) e a orientação (b) do robô.

5.2.6 Experimento 6

Neste experimento o robô foi colocado para navegar da posição (0, 0) até atingir o ponto (3000 mm, 4000 mm). Da mesma forma que no Experimento 3, existem dois obstáculos circulares de 30 cm de diâmetro no ambiente por onde o robô navega. Eles estão localizados nas posições (1500 mm, 1000 mm) e (1500 mm, 2500 mm). Porém, desta vez, o algoritmo de desvio de obstáculos usado não foi o de desvio tangencial, e sim o algoritmo de desvio em que o robô busca o caminho mais próximo da posição final a ser encontrada, como mostrado na Seção 4.2.2. A trajetória descrita pelo robô pode ser vista na Figura 73.

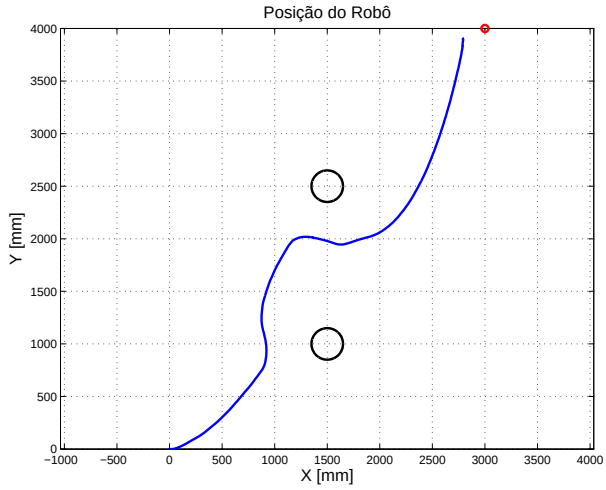
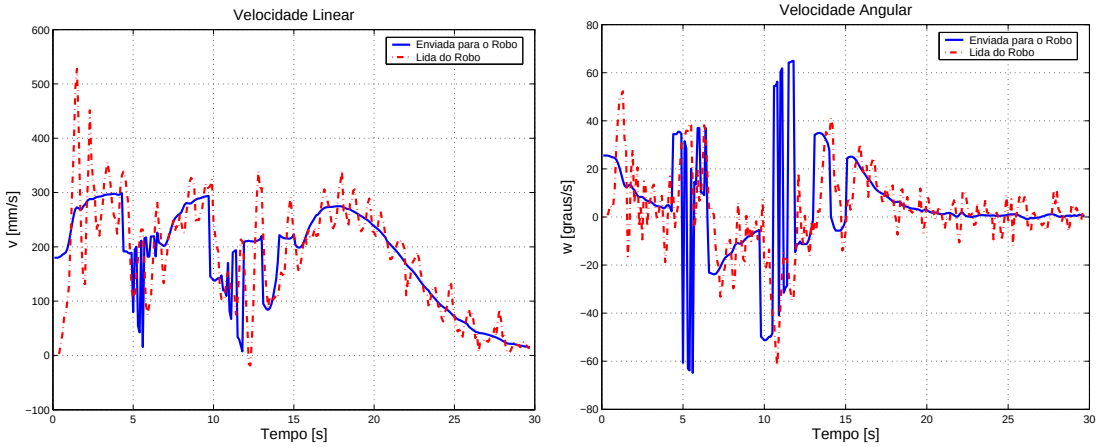


Figura 73: Trajetória descrita pelo robô.

Nota-se, observando a Figura 73 que, quando o robô encontra o segundo obstáculo, ele pode desvencilhar-se do mesmo por dois caminhos. Entretanto o caminho escolhido

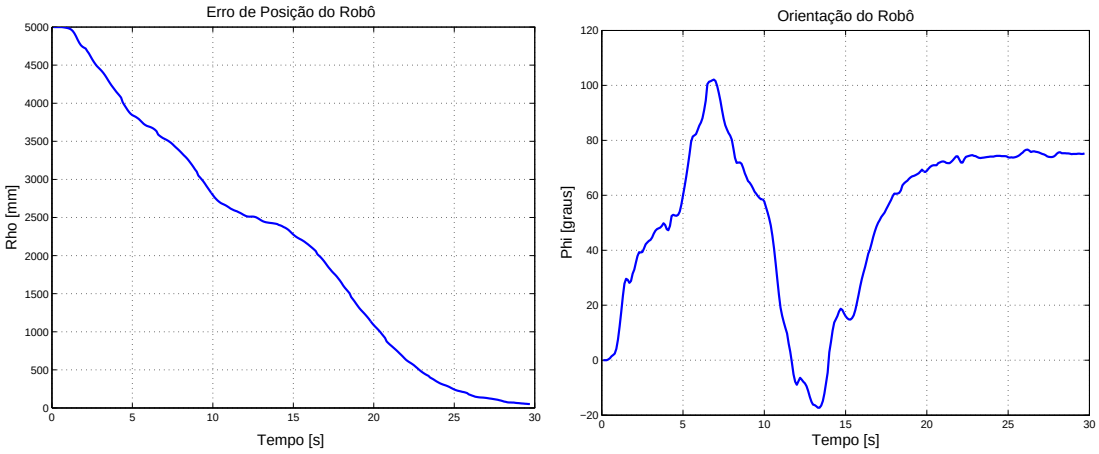
foi aquele cuja orientação possui o menor erro com relação ao ponto final desejado, como era esperado.

As velocidades linear e angular enviadas ao robô e lidas dele podem ser vistas nas Figuras 74 (a) e 74 (b), respectivamente, enquanto o erro de posição e a orientação do robô estão apresentados nas Figuras 75 (a) e 75 (b), respectivamente.



(a) (b)

Figura 74: Velocidades linear (a) e angular (b) do robô.



(a) (b)

Figura 75: Erro de posição (a) e a orientação (b) do robô.

Neste experimento, que é bastante semelhante ao Experimento 3, o robô atingiu o ponto destino desviando, de forma não-tangencial, os obstáculos encontrados. Pode-se observar que a manobra de desvio do segundo obstáculo detectado é feita de tal forma que o caminho seguido pelo robô fosse aquele com o menor erro de orientação com relação ao ponto final desejado.

5.2.7 Experimento 7

Neste experimento, é utilizado o controlador para navegação em corredores e, caso seja detectado algum obstáculo, este controlador é desativado e o controlador de desvio não tangencial é habilitado até que o obstáculo seja ultrapassado. O robô é posto para navegar em um corredor com 2 m de largura por um período de 35 s. O robô está posicionado, inicialmente, a 60 cm de distância da parede direita do corredor. É colocado, ainda, um obstáculo de 30 cm de diâmetro, 3 m à frente da posição inicial do robô e a 30 cm à esquerda do mesmo.

A Figura 76 mostra a trajetória descrita pelo robô enquanto o mesmo navegava no corredor.

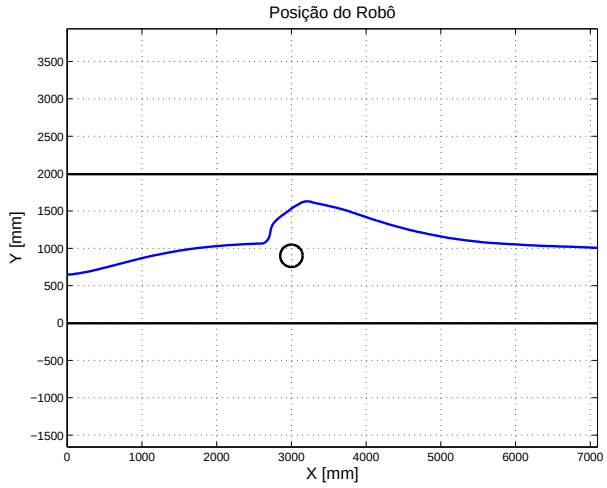


Figura 76: Trajetória descrita pelo robô.

Na Figura 77 estão plotadas as velocidades lineares e angulares enviadas ao robô e lidas da odometria do mesmo.

O erro de posição do robô em relação ao centro do corredor está apresentado na Figura 78 (a), enquanto a orientação dele em relação ao corredor pode ser vista na Figura 78 (b).

Pode-se observar que o robô encontrou o meio do corredor e continuou navegando por ele, em paralelo às paredes do mesmo, até que o sistema sensorial detectou o obstáculo e o robô reagiu para evitá-lo. Depois que o obstáculo foi vencido o robô volta a navegar com o comportamento de seguir corredores e, novamente, atinge o seu objetivo.

Nota-se que, no início do movimento o módulo do erro de posição decresce até o momento em que o robô detecta o obstáculo e inicia o movimento de manobra para evitá-

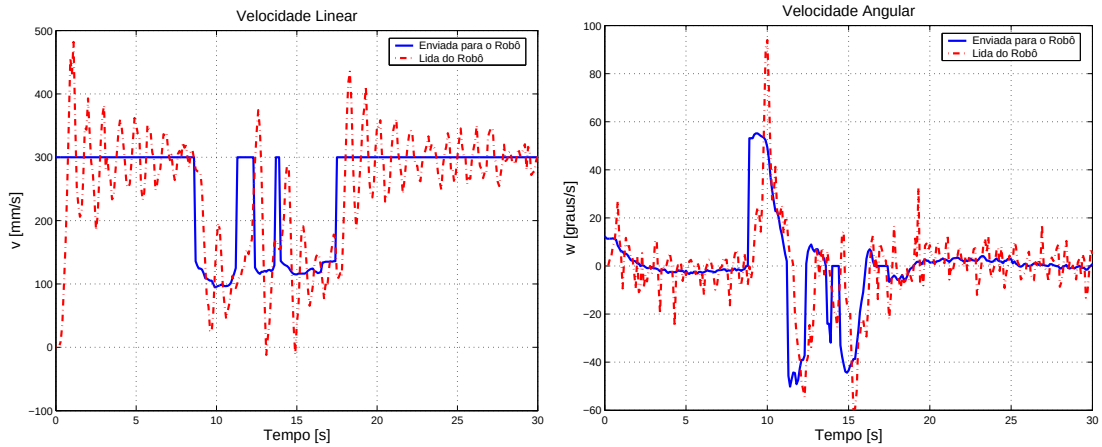
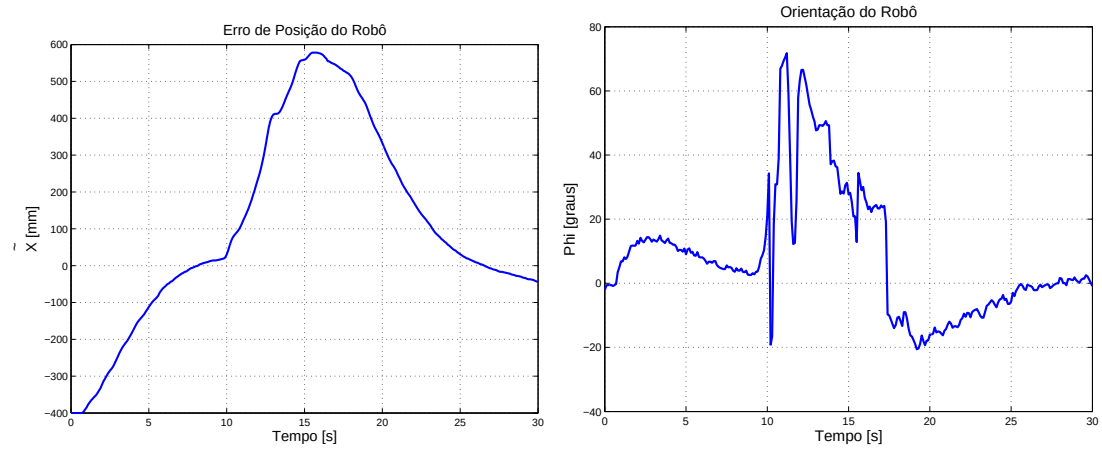


Figura 77: Velocidades linear (a) e angular (b).



(a) (b)

Figura 78: Erro de posição (a) e a orientação (b) do robô.

lo. Desta maneira, o robô se afasta do centro do corredor e o erro de posição aumenta. Após o obstáculo ter sido ultrapassado, o erro volta a decrescer e tende a zero com o decorrer do tempo.

5.2.8 Experimento 8

Neste experimento utiliza-se o controlador de navegação em corredores. O robô é posto para navegar em um corredor com 2 m de largura e, num determinado instante, uma das paredes deste corredor deixa de ser paralela à outra, gerando um afunilamento no corredor que, depois deste afunilamento, passa a ter 1 m de largura. O robô navega por um período de 35 s. Inicialmente, ele está posicionado a uma distância de 60 cm da parede direita do corredor mais largo, e orientado de acordo com o mesmo.

A Figura 79 mostra a trajetória descrita pelo robô enquanto o mesmo navegava no corredor.

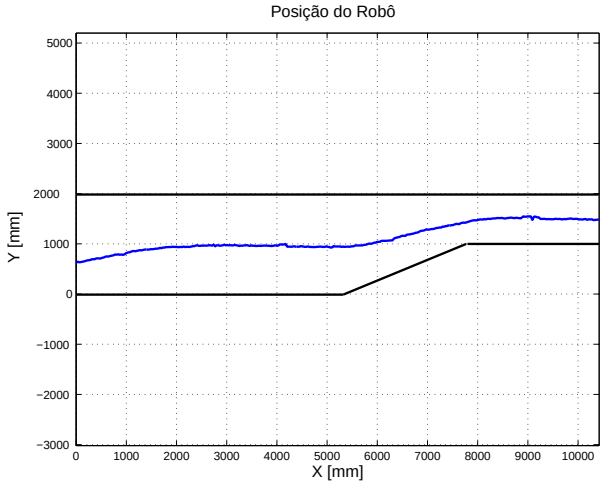


Figura 79: Trajetória descrita pelo robô.

Na Figura 80 estão plotadas as velocidades lineares e angulares enviadas ao robô e lidas da odometria do mesmo.

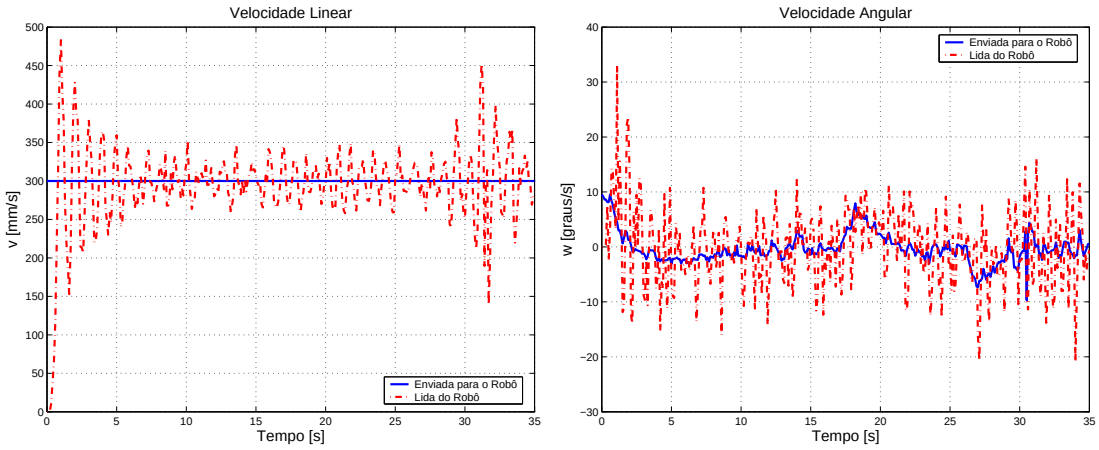
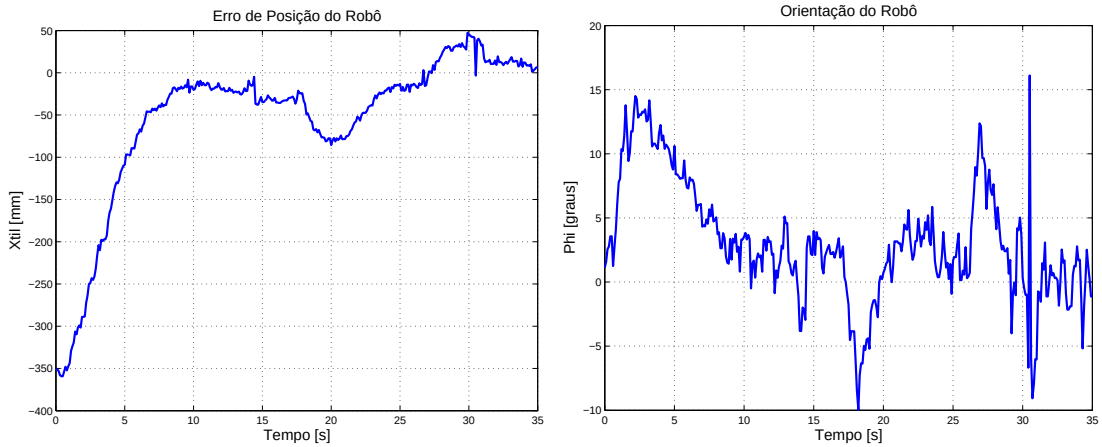


Figura 80: Velocidades linear (a) e angular (b).

O erro de posição do robô em relação ao centro do corredor está apresentado na Figura 81 (a), enquanto a orientação dele em relação ao corredor pode ser vista na Figura 81 (b).

Pode-se observar que o robô encontrou o meio do corredor e continuou navegando por ele em paralelo às paredes do corredor. No momento de afunilamento do corredor, o robô navegou, tentando manter-se equidistante às paredes direita e esquerda. Assim que as paredes voltam a ficar paralelas, ele navega, novamente, pela região central do corredor e orientado com o mesmo.



(a) (b)

Figura 81: Erro de posição (a) e a orientação (b) do robô.

5.3 Análise dos Resultados

Os experimentos apresentados neste capítulo foram feitos para reproduzir as simulações que foram apresentadas nos Capítulos 3 e 4.

Pode-se observar que os resultados experimentais foram bastante semelhantes às simulações que foram realizadas. E, ainda, o robô navega desviando dos obstáculos, quando os encontra, e consegue alcançar o ponto final desejado. Para o caso da navegação em corredores, nota-se que o robô encontra a região central do corredor e continua navegando orientado, seguindo este mesmo corredor. É importante citar que mesmo quando o robô foi colocado para navegar em corredores afunilados, situação que não foi abrangida pelos métodos de determinação das variáveis de estado, o robô conseguiu navegar por eles e conseguiu manter aproximadamente iguais as distâncias até as paredes esquerda e direita. Uma outra situação foi a navegação em corredores que continham obstáculos. Para estas circunstâncias, foi adicionado ao controlador de navegação em corredores o controlador de desvio não-tangencial de obstáculos. Observa-se que o robô navega pelo corredor até se deparar com o obstáculo e, depois que este é transposto, o robô continua navegando em busca do centro do corredor.

Observando-se os gráficos das velocidades linear e angular, medidas através da odometria do robô, durante as simulações e durante os experimentos, pode-se notar, claramente, a diferença existente entre elas. Esta diferença se dá devido ao fato do simulador *SRIsim* não levar em consideração a dinâmica do robô que está sendo simulado. Um outro fator que pode ser responsável por esta diferença é o erro de leitura apresentado pelo sistema de odometria do robô além dos algoritmos de controle não compensarem a dinâmica do

robô.

É possível observar nos experimentos que, mesmo que os controladores tenham sido implementados para garantir a não saturação dos sinais de controle, os valores destes sinais lidos do robô atingem valores superiores aos limites impostos pelo controlador. Isso ocorre porque tais controladores não levaram em consideração a dinâmica do robô, além dos erros de odometria envolvidos.

Um outro detalhe importante a mencionar é que chegou-se a utilizar obstáculos bastante estreitos como cabos de vassoura e cabos co-axiais cujos diâmetros medem, respectivamente, 2 cm e 0.5 cm. Ainda nestes casos, o robô identificou-os como obstáculos e manobrou para evitá-los, o que não acontece quando são utilizados os sensores ultrasônicos. Tais experimentos evidenciam a alta resolução angular do sensor *laser* utilizado.

6 Conclusões e Trabalhos Futuros

"Se não houver frutos,
valeu a beleza das flores...
Se não houver flores,
valeu a sombra das folhas...
Se não houver folhas,
valeu a intenção da semente".

Henfil

Neste trabalho foi abordado o problema da navegação de robôs móveis em ambientes semi-estruturados. Foram implementados controladores para navegação em corredores, busca de um ponto objetivo e desvio de obstáculos. Tais controladores foram baseados em arquiteturas reativas, ou seja, o sistema de sensoriamento utilizado deve coletar as informações referentes ao ambiente no qual o robô navega e o algoritmo de controle deve "reagir"de forma que o robô possa continuar navegando até alcançar seu objetivo. O sensoriamento do ambiente é feito através de um sensor *laser* de varredura instalado a bordo do robô móvel utilizado, o PIONEER 2-DX.

O controlador para busca do ponto objetivo é o único que não utiliza as informações de distância fornecidas pelo sensor *laser*. Este controlador, cujo objetivo é levar o robô de uma posição até outra, faz uso das leituras dos *encoders*, presentes nas rodas do robô, para determinar os valores das varáveis de controle e, desta forma, encaminhar o robô até o seu destino.

Já o controlador para navegação em corredores utiliza apenas quatro das 181 medidas de distância proporcionadas pelo sensor *laser*. Com elas foi possível determinar a orientação do robô com relação ao corredor, φ, e a distância que ele se encontra da linha média do mesmo, $\tilde{x}$. É importante citar que foram desenvolvidos dois métodos para determinar a orientação e o erro de posição do robô em relação ao centro do corredor, ambos possibilitando obter boas estimativas de $\tilde{x}$ mesmo quando φ não é pequeno.

Também foram implementados, nesta Dissertação, dois algoritmos para desvio de obstáculos. O primeiro deles acrescenta um laço externo à malha de controle de posição. Este laço externo é responsável pela criação de um alvo virtual que o robô deve alcançar enquanto está diante de um obstáculo. Assim, logo que um obstáculo é detectado, é gerado um ângulo de desvio, como no controle baseado em impedância [27], e o alvo real é rotacionado. Porém este ângulo é calculado para que o robô se desvencilhe do obstáculo, seguindo uma trajetória tangente ao mesmo [12, 8, 21]. O outro controlador para evitar obstáculos é independente do controlador de posição final. Ele, baseado nas medidas fornecidas pelo *laser*, encontra um caminho livre de obstáculos que possui o menor erro de orientação com relação ao ponto final desejado, e, dessa forma, o robô é capaz de navegar em busca do seu alvo e transpor os obstáculos que surgirem em seu caminho.

A fim de se comprovar a funcionalidade dos controladores implementados, foram escritos programa em linguagem C++ utilizando a biblioteca "Aria.h". Assim, inicialmente, foram realizadas simulações com o auxílio dos *softwares Mapper*, responsável pela edição dos ambientes onde o robô será posto para navegar, e do *SRIsim* que é o simulador propriamente dito, o qual leva em consideração os modelos reais dos erros dos sensores presentes no robô, apesar de não considerar a dinâmica do mesmo. Após isso, foram realizados experimentos semelhantes às simulações.

Os controladores implementados apresentaram sucessos tanto nas simulações quanto nos experimentos realizados. Ou seja, o robô atingiu o objetivo de controle (navegar em corredores, chegar até uma posição determinada e evitar os eventuais obstáculos) em todas as vezes que foi posto para navegar. Estes resultados satisfatórios obtidos durante as simulações e experimentos comprovam a validade dos métodos utilizados para implementação dos controladores.

Se o modelo dinâmico do robô móvel fosse considerado pelos controladores (os controladores só levam em consideração a cinemática), a diferença existente entre as velocidades calculadas pelo controlador e as obtidas dos sensores de odometria do robô poderia ser, pelo menos, amenizada, e, assim, reduzir os sobre-sinais ocorridos.

6.1 Contribuições deste Trabalho

A principal diferença entre os resultados desta Dissertação e os que são apresentados em [22, 12, 8, 13] é que nestes as informações sobre o ambiente no qual o robô navega são obtidas por sensores ultra-sônicos, enquanto aqui o sistema sensorial é formado por um

sensor de varredura *laser*.

Uma outra contribuição digna de nota são os métodos para determinação das variáveis de estado, para navegação em corredores (ver Capítulo 2) que são específicos para o sensor aqui adotado.

6.2 Trabalhos Futuros

Pode-se, futuramente, realizar diversos trabalhos como continuidade do apresentado nesta Dissertação de Mestrado. Por exemplo, como em [22], pode-se fazer a fusão dos dados provenientes de diversos controladores para decidir qual comportamento deve ser considerado.

Poderia-se, também, utilizar o sensor *laser* a para realizar um mapeamento do ambiente onde o robô será posto para navegar. O mapeamento realizado poderá ser tanto do tipo métrico quanto topológico. Em um mapeamento métrico, valores de distâncias e profundidades podem ser utilizadas para estabelecer uma localização do robô em um referencial absoluto. Já no caso do mapeamento topológico as informações coletadas podem ser utilizadas junto com, por exemplo imagens, para caracterizar topologicamente o ambiente.

Um outro contexto em que o uso do sensor *laser* de varredura seria interessante poderia ser, por exemplo, usar as medidas de distância que ele fornece para o desenvolvimento de estratégias para navegação de uma equipe de robôs.

Finalmente, vale destacar a possibilidade de uso das informações provenientes do sensor de varredura *laser* num esquema de fusão de dados sensoriais, para permitir estimar com maior grau de certeza distâncias, posições relativas, etc.

Referências

[1] HEINEN, F. J. *Sistema de Controle Híbrido para Robôs Móveis Autônomos*. Dissertação (Mestrado) — Universidade Federal do Vale do Rio dos Sinos - UNISINOS, 2002.

[2] KUC, R.; BARSHAN, B. Navigating vehicles through an unstructured enviroment with sonar. *IEEE International Conference on Robotics and Automation*, v. 3, p. 1422–1426, 1989.

[3] ELFES, A. Sonar-based real-world mapping and navigation. *IEEE Journal of Robotic and Automation*, RA-3(3), p. 249–265, 1987.

[4] KHATIB, O. Real-time obstacle avoidance for manipulators and mobile robots. *The International Journal of Robotics Research*, v. 5(1), p. 90–98, 1986.

[5] BORENSTEIN, J.; KOREN, Y. Real-time obstacle avoidance for fast mobile robots. *IEEE Transactions on Systems, Man, and Cybernetics*, v. 19, n. 5, p. 1179–1187, 1989.

[6] BORENSTEIN, J.; KOREN, Y. The vector field histogram - fast obstacle avoidance for mobile robots. *IEEE Journal of Robotics and Automation*, v. 7, n. 3, p. 278 – 288, June 1991.

[7] MINGUEZ, J.; MONTANO, L. Nearness diagram (ND) navigation: Collision avoidance in troublesome scenarios. *IEEE Transactions on Robotics and Automation*, v. 20, n. 1, p. 45–59, 2004.

[8] FERREIRA, A.; SARCINELLI-FILHO, M.; BASTOS-FILHO, T. F. A new approach to avoid obstacles in mobile robot navigation: Tangential escape. In: *International Conference on Informatics in Control, Automation and Robotics, ICINCO'05*. Barcelona, Spain: [s.n.], 2005. v. 3, p. 341–346.

[9] MURPHY, R. R. *Introduction to AI Robotics*. Massachussets: MIT, 2000.

[10] BROOKS, R. A. New approches to robotics. *in Science*, v. 253, p. 1227 – 1232, September 1991.

[11] FREIRE, E. O. *Desenvolvimento de um Sistema de Sensoreamento Ultra-Sônico para Robô Móvel Orientado a Agentes*. Dissertação (Mestrado) — UFES, 1997.

[12] FERREIRA, A. *Desvio Tangencial de Obstáculos para Um Robô Móvel Navegando em Ambientes Semi-estruturados*. Dissertação (Mestrado) — Universidade Federal do Espírito Santo, 2004.

[13] CARELLI, R.; FREIRE, E. O. Corridor navigation and wall-following stable control for sonar-based mobile robots. *Robotics and Autonomous Systems*, v. 45, p. 235–247, 2003.

[14] BASTOS-FILHO, T. F. *Seguimiento y Análisys de Entornos de Soldadura por Arco Automatizada Mediante Ultrasonidos*. Tese (Doutorado) — Universidad Complutense de Madrid, 1995.

[15] FREITAS, R. A. de C. *Sistema de Visão para Robôs Móveis: Uma Aplicação ao reconhecimento de Referências Geométricas*. Dissertação (Mestrado) — UFES, 1999.

[16] FRANCA, A. S.; VASSALLO, R. F.; SCHNEEBELI, H. J. A. Detecção de obstáculos através de um fluxo óptico padrão obtido a partir de imagens omnidirecionais. *VII - Simpósio de Automação Inteligente*, São Luís, Maranhão, Setembro 2005.

[17] SANTOS-VICTOR, J. et al. Divergent stereo for robot navigation : Learning from bees. In: *IEEE CS Conf. Computer Vision and Pattern Recognition*. New York, USA: IEEE CS Conf. Computer Vision and Pattern Recognition, 1993.

[18] SANTOS-VICTOR, J. et al. Divergent stereo in autonomous navigation: From bees to robots. *International Journal of Computer Vision*, v. 14, n. 2, p. 159–177, March 1995.

[19] BRAGANCA, J. de O. *Estratégia para Deslocamento de Cargas através de Cooperação entre Robôs Móveis a Rodas*. Dissertação (Mestrado) — UFES, 2004.

[20] PEREIRA, F. G. et al. Calibração de sistemas catadióptricos e detecção da pose de robôs móveis por segmentação de imagens omnidirecionais. *VII - Simpósio de Automação Inteligente*, São Luís, Maranhão, Setembro 2005.

[21] FERREIRA, A. et al. Avoiding obstacles in mobile robot navigation: Implementing the tangential escape approach. *ISIE*, 2006.

[22] FREIRE, E. O. *Controle de Robôs Móveis por Fusão de Sinais de Controle Usando Filtro de Informação Descentralizado*. Tese (Doutorado) — UFES, 2002.

[23] VASSALLO, R. F. *Uma Estratégia para Navegação de Robôs Móveis em Ambientes Interiores*. Dissertação (Mestrado) — UFES, 1998.

[24] VASSALLO, R. F.; SCHNEEBELI, H. J.; SANTOS-VICTOR, J. Visual servoing and appearence for navigation. *Robotics and Autonomous Systems*, v. 31, p. 87–97, 2000.

[25] GAMARRA, D. F. T. *Controle da Navegação de um Robô Móvel em um Corredor com Redundância de Controladores*. Dissertação (Mestrado) — UFES, 2004.

[26] POLAROID-CORPORATION. *Ultrasonic ranging system / technical assistence.*

[27] SECCHI, H. A. *Control de Vehículos Autoguiados con Realimentación Sensorial*. Dissertação (Mestrado) — Facultad de Inginería de la Universidad Nacional de San Juan, San Juan - Argentina, 1998.

[28] ACTIVMEDIA Robotics - Pioneer 2 Mobile Robots - Computer and Software Manual. March 2001.

[29] KONOLIGE, K. G. *Saphira and Aria Software Manual*. Menlo Park, Califórnia, September 2001.