DISSERTAÇÃO DE MESTRADO Nº 643

NAVEGAÇÃO SEGURA DE UM CARRO AUTÔNOMO UTILIZANDO CAMPOS VETORIAIS E O MÉTODO DA JANELA DINÂMICA

Danilo Alves de Lima

DATA DA DEFESA: 13/12/2010

Universidade Federal de Minas Gerais

Escola de Engenharia

Programa de Pós-Graduação em Engenharia Elétrica

NAVEGAÇÃO SEGURA DE UM CARRO AUTÔNOMO UTILIZANDO CAMPOS VETORIAIS E O MÉTODO DA JANELA DINÂMICA

Danilo Alves de Lima

Dissertação de Mestrado submetida à Banca Examinadora designada pelo Colegiado do Programa de Pós-Graduação em Engenharia Elétrica da Escola de Engenharia da Universidade Federal de Minas Gerais, como requisito para obtenção do Título de Mestre em Engenharia Elétrica.

Orientador: Prof. Guilherme Augusto Silva Pereira

Belo Horizonte - MG

Dezembro de 2010

"Navegação Segura de Um Carro Autônomo Utilizando Campos Vetoriais e O Método da Janela Dinâmica"

Danilo Alves de Lima

Dissertação de Mestrado submetida à Banca Examinadora designada pelo Colegiado do Programa de Pós-Graduação em Engenharia Elétrica da Escola de Engenharia da Universidade Federal de Minas Gerais, como requisito para obtenção do grau de Mestre em Engenharia Elétrica.

Aprovada em 13 de dezembro de 2010.

Por:

Prof. Dr. Guilherme Augusto Silva Pereira
DEE (UFMG) - Orientador

Prof. Dr. Luciano Cunha de Araújo Pimenta
DELT (UFMG)

Prof. Dr. Mario Fernando Montenegro Campos
DCC (UFMG)

Dedico esta dissertação à minha mãe,
eterna em sua sabedoria.

Agradecimentos

Agradecer é a forma mais humilde que tenho para reconhecer que sozinhos não somos ninguém. Desta forma, deixo aqui o meu breve agradecimento a todos aqueles que direta e indiretamente contribuíram com este trabalho.

Primeiramente, gostaria expressar o meu muito obrigado ao Professor Guilherme, não somente pela orientação neste trabalho, mas pela amizade e companheirismo em quase 5 anos de vida acadêmica. Nesta jornada, eu pude conhecer uma pessoa realmente ética e verdadeira, cujas idéias foram fundamentais para este sucesso.

Agradeço aos meus pais, Neusa e Aderaldo, pela minha primeira formação e a mais importante delas, a formação para a vida. Aos meus irmãos, companheiros de alegrias, tristezas e, até mesmo, desentendimentos, necessários para o meu amadurecimento como pessoa. A minha Flor, pelo carinho e compreensão em cada momento difícil pelo qual passei. A minha avó, sinônimo de força e perseverança. Aos demais parentes, pela compreensão de minha ausência em diversos momentos.

Agradeço também aos amigos conquistados durante toda a minha caminhada e fundamentais neste trabalho: Marco, Elias, Michelle, Vitor, Diego, Fred, Matheus, João, Tiago Mendonça, Tiago Arruda, Bruno e demais companheiros do laboratório CORO. Deixo minha consideração aos demais professores, amigos e companheiros de mestrado pelas trocas de experiências e convívio. Meus velhos amigos também não poderiam ser esquecidos, alias, é impossível esquecê-los, pois a presença de todos é eterna em minha vida. Prefiro não citar nomes para não cometer injustiças, mas todos sabem o quanto foram importantes para mim.

Agradeço de forma única e especial a Deus, mestre maior e guia espiritual de todos os meus desafios.

Por fim, deixo o meu agradecimento a todos os brasileiros que indiretamente contribuíram para este trabalho, possibilitando que minha bolsa fosse fornecida pela CAPES e o financiamento do projeto pela FAPEMIG.

"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 Klink

Resumo

A navegação segura é uma tarefa fundamental para os veículos autônomos, os quais necessitam de uma interação completa com o meio em que estão inseridos. Saber se localizar no mundo, planejar seu movimento, perceber o ambiente e desviar de possíveis obstáculos, são apenas algumas das etapas que devem ser realizadas pelo veículo. Este trabalho aborda o problema de navegação segura de um carro autônomo. Para tanto, é utilizado um planejamento de movimento por meio de Campos Vetoriais de Velocidade aliado ao Método da Janela Dinâmica para o desvio de obstáculos não modelados. Basicamente, o campo vetorial é associado a um controlador cujas saídas são validadas pelo Método da Janela Dinâmica e aplicadas como entradas de controle do carro. Para auxiliar na navegação, técnicas de localização por fusão sensorial e percepção do ambiente por uma grade de ocupação local foram incorporadas à solução. A validação da metodologia apresentada foi realizada em um ambiente de simulação, onde sensores a laser e visuais foram avaliados, e posteriormente implementada no CADU (Carro Autônomo Desenvolvido na Universidade Federal de Minas Gerias) que utiliza visão estéreo para a detecção dos obstáculos. Os resultados, tanto em simulação quanto no CADU, controlaram adequadamente o veículo em um ambiente não estruturado. Neles, o veículo foi capaz de se guiar pelo campo vetorial e desviar de obstáculos em seu caminho, navegando de forma segura ao longo do ambiente. Espera-se que a incorporação de mais sensores e um sistema localização mais preciso permita que o carro navegue por ambientes mais complexos utilizando a metodologia proposta neste trabalho.

Palavras chave: Navegação segura; Carro autônomo; Campo Vetorial; Método da Janela Dinâmica.

Abstract

Safe navigation is fundamental for autonomous vehicles, which requires a complete interaction with its surroundings. Self localization, motion planning, environment perception and obstacle avoidance, are important steps that must be realized by the vehicle. This work presents a safe navigation approach for a car-like robot. It uses a path planning algorithm based on Velocity Vector Fields along with a Dynamic Window Approach for avoiding unmodeled obstacles. The vector field is associated to a controller whose outputs are validated by the Dynamic Window Approach and applied as control inputs for the car. To assist navigation, some known techniques have been incorporated to the final solution, such as a sensor fusion system for localization and a local occupancy grid for environment perception. The methodology of this work was validated in simulation, where lasers and visual sensors were evaluated, and later applied to CADU (a car-like robot developed in the Federal University of Minas Gerais) that uses a stereo vision system for obstacle detection. The results, for both cases, controlled the vehicle in an unstructured environment. The vehicle was able to track the vector field and avoid obstacles in its way, navigating safely through the environment. It is expected that more sensors and a better localization system would allow the car to navigate about more complex places using the methodology presented in this work.

Keywords: Safe navigation; Car-like robot; Vector Field; Dynamic Window Approach.

Lista de Figuras

Lista de Abreviaturas e Siglas

CADU	Carro Autônomo Desenvolvido na Univesidade Federal de Minas Gerais (UFMG).
CDT	Triangulação de Delaunay com Restrições ou do inglês *Constrained Delaunay Triangulation*.
CVM	Método das Curvas de Velocidade ou do inglês *Curvature-Velocity Method*.
DARPA	Do inglês *Defence Advanced Research Projects Agency*.
DWA	Método da Janela Dinâmica ou do inglês *Dynamic Window Approach*.
FBL	Linearização por realimentação estática ou do inglês *static Feedback Linearization*.
FOV	Campo de visão ou do inglês *Field of View*.
ICC	Centro de Curvatura Instantâneo ou do inglês *Instantaneous Center of Curvature*.
ICS	Estado de Colisão Inevitável ou do inglês *Inevitable Collision State*.
LCM	Método das Curvas de Caminho ou do inglês *Lane-Curvature Method*.
MDF	Arquivo de Dados de Missão ou do inglês *Mission Data File*.
ND	Diagrama de Proximidade ou do inglês *Nearness Diagram*.
PDVA	Grupo de Pesquisa e Desenvolvimento de Veículos Autônomos.
RDDF	Formato de Dados que Definem a Rota ou do inglês *Route Definition Data Format*.
RNDF	Arquivo de Define a Rede de Ruas ou do inglês *Road Network Definition File*.
VFH	Histograma de Campos Vetoriais ou do inglês *Vector Field Histogram*.
VO	Método dos Obstáculos com Velocidade ou do inglês *Velocity Obstacles*.

Lista de Algoritmos

Sumário

Capítulo 1

Introdução

1.1 Motivação

O mundo é um sistema dinâmico e em constante evolução. As tecnologias atuais fazem parte desta evolução, fornecendo aos seres humanos meios que facilitem suas tarefas diárias, melhorem sua qualidade de vida e os auxiliem na redução de erros que podem gerar acidentes. A robótica e suas aplicações são alguns destes desdobramentos tecnológicos atuais. No geral, a robótica é utilizada para realizar tarefas repetitivas e perigosas aos seres humanos. Por possuir várias dessas tarefas, são nas indústrias que os robôs estão mais inseridos e com uma infinidade de modelos. São robôs manipuladores de pequenos objetos, transportadores de carga, soldadores, entre outros (ver exemplos na Figura 1.1).

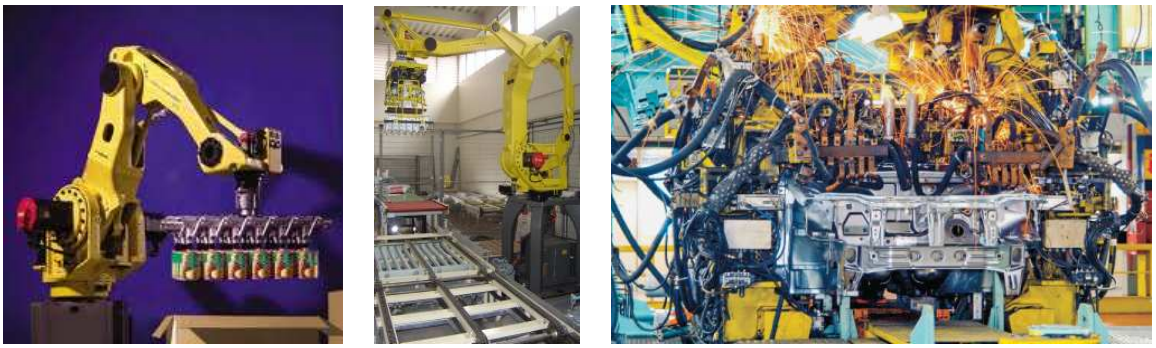


Figura 1.1: Exemplos de robôs industriais e suas aplicações, onde à esquerda tem-se um para a manipulação de pequenos objetos, ao centro outro para o transporte de grandes cargas e à direita um para a soldagem de carros.

Apesar de muitos terem finalidades industriais, o desenvolvimento da robótica, em muitos casos, não se inicia com este fim. O surgimento de carros autônomos, por exem-

plo, ocorreu em laboratórios e centro de pesquisas, com as principais aplicações focadas em veículos de passeio convencionais. Os objetivos destas aplicações eram os mais diversos como explorar ao máximo as redes de ruas, ter um baixo consumo de combustível/energia e aumentar as condições de segurança para um motorista que o utilizasse [Broggi et al., 1999].

Dentre os trabalhos desenvolvidos para carros autônomos, os mais significativos surgiram com os desafios propostos pela agência americana DARPA[1] ([DARPA, 2005] e [DARPA, 2008]). Neles, várias inovações e soluções para diversos problemas comuns aos carros autônomos foram compartilhadas com a comunidade científica. Os resultados apresentados provaram que, em alguns anos, seria possível conceber carros completamente autônomos ou com certa inteligência embarcada que auxiliasse o motorista. Os veículos campeões destes desafios estão na Figura 1.2.

Figura 1.2: Carros autônomos campeões dos desafios DARPA ([DARPA, 2005] e [DARPA, 2008]). O veículo da esquerda é o Stanley [Thrun et al., 2006], campeão em 2005, e o da direita é o Boss [Urmson et al., 2008], campeão em 2007.

Atualmente, muitas das tecnologias utilizadas nos veículos participantes dos desafios DARPA já podem ser vistas em circulação em alguns carros de passeio. Entre elas, destacam-se os sistemas para auxílio de estacionamento (*park assistance*), controle ativo de velocidade (*active cruise control*) e alerta de colisão (*collision alert*). Os desenvolvimentos são tamanhos que em atividades de mineração já é possível encontrar veículos autônomos que auxiliam seres humanos em tarefas de alto risco [Jamasmie, 2009], de acordo com a Figura 1.3.

Em uma análise geral, os veículos autônomos têm sido amplamente estudados e muitos já possuem aplicações práticas. Porém, esta é uma realidade pouco comum no

[1] Do inglês *Defence Advanced Research Projects Agency*.

Figura 1.3: Caminhão fora de estrada autônomo produzido pela Komatsu para ambientes de mineração [Jamasmie, 2009].

Brasil, pois grande parte da tecnologia disponível foi desenvolvida em outros países. Foi com o intuito de desenvolver tecnologias nacionais, na Universidade Federal de Minas Gerais (UFMG), que pesquisadores fundaram o Grupo de Pesquisa e Desenvolvimento de Veículos Autônomos (PDVA). Desde de 2007 este grupo trabalha no desenvolvimento do CADU[2], um automóvel que serve como plataforma de estudo e experimentação de aplicações voltadas para carros autônomos.

Muitos trabalhos foram desenvolvidos no CADU, com a incorporação de recursos que permitem torná-lo autônomo. Ele possui, por exemplo, sistemas de atuação e instrumentação que permitem movê-lo e localizá-lo no mundo ([Freitas et al., 2009], [Freitas and Pereira, 2010] e [Santos, 2009]). No entanto, antes do desenvolvimento deste trabalho ele ainda não possuía um sistema de navegação segura, que integrasse seus recursos e realize o seu movimento entre dois pontos quaisquer no espaço.

Assim, com o intuito de se agregar funcionalidades de navegação segura ao CADU, foi realizado este trabalho. Os objetivos para sua concepção serão apresentados a seguir.

1.2 Objetivos

O resultado do presente trabalho tem por objetivo realizar a navegação segura de um carro autônomo por meio de um planejamento de movimento baseado em Campos Vetoriais de Velocidade aliado ao Método da Janela Dinâmica para o desvio de obstáculos não modelados. Sua concepção compreenderá:

[2] Sigla para Carro Autônomo Desenvolvido na UFMG.

- A modelagem cinemática do carro, com suas respectivas entradas de controle;
- A adaptação das técnicas de planejamento de movimento e desvio de obstáculos, para que sejam fornecidas saídas correspondentes às entradas de controle do carro;
- O desenvolvimento de um sistema de percepção de ambientes por meio de sensores a laser e de visão estéreo;
- A integração de um sistema de localização e controle de velocidade já existentes à solução de navegação proposta;
- A agregação e validação das novas funcionalidades no CADU.

Com estes elementos, pretende-se fornecer subsídios para que se possa prosseguir com o projeto e utilizá-lo em pesquisas futuras.

1.3 Contribuições e relevância

Para permitir que um carro autônomo se mova com segurança é necessário que este possua um completo sistema de navegação. A navegação é responsável por realizar um planejamento de movimento que guie o carro de uma origem a um destino, e garanta segurança com a detecção e o desvio de obstáculos. Durante todo este processo, o carro, também deve conhecer sua localização no mundo e controlar seu movimento.

Existem diversas composições de técnicas para realizar a navegação de um carro autônomo. Nos trabalhos estudados, por exemplo, o mais comum é o uso de técnicas que criam caminhos que posteriormente são seguidos por um controlador de trajetória específico. No entanto, como as entradas de controle de veículos autônomos são normalmente velocidades, planejar o movimento e desviar de obstáculos por meio de comandos de velocidade, eliminam a necessidade de controladores específicos e simplificam a solução, integrando planejamento de movimento e controle na mesma técnica.

A solução proposta neste trabalho foi concebida tendo como princípio utilizar técnicas que fornecessem comandos de velocidade para o controle baixo nível do carro. Neste caso, optou-se pela união de duas técnicas em sequência, o Campo Vetorial de

Velocidades e o Método da Janela Dinâmica. A primeira técnica é responsável pelo planejamento prévio do movimento do veículo, imutável no decorrer de seu movimento. Já a segunda realiza a validação destas velocidades segundo os obstáculos detectados ao redor do veículo em cada instante e fornece novas velocidades para permitir o desvio dos obstáculos, caso estas sejam inválidas. Deve ser observado que, neste caso, não existe replanejamento do movimento.

Uma importante contribuição deste trabalho é a incorporação da solução de navegação estudada ao CADU. Esta contará com a integração dos recursos de atuação, controle de velocidade e localização já existentes no carro. Com ela será fornecida ao CADU a capacidade de perceber o ambiente e realizar um movimento seguro de um ponto de início a um de destino.

O desenvolvimento desta dissertação resultou na publicação de um artigo apresentado no XVIII Congresso Brasileiro de Automática. O artigo, que pode ser encontrado em [Lima and Pereira, 2010], detalha a implementação da detecção de obstáculos por um sistema de visão estéreo incorporada à solução de navegação segura proposta nesta dissertação. No artigo, o resultado obtido com a visão estéreo foi avaliado no desvio de obstáculos de um carro autônomo por meio de um método similar ao Histograma de Campos Vetoriais.

1.4 Organização da dissertação

O Capítulo 1 apresentou a introdução deste trabalho com a motivação que levou à sua elaboração, os objetivos pretendidos e algumas de suas contribuições. No Capítulo 2, é realizado um breve histórico sobre o desenvolvimento de carros autônomos, o qual é a base deste trabalho. Serão apresentadas, também, algumas técnicas isoladas de planejamento de movimento, detecção e desvio de obstáculos, que em conjunto compõem os trabalhos relacionados sobre a navegação autônoma. Pretende-se, assim, contextualizar o leitor sobre os elementos abordados nesta dissertação e esclarecer como eles foram utilizados em outros trabalhos. Em seguida, no Capítulo 3, estão os principais conceitos e técnicas de navegação que foram incorporados direta ou indiretamente neste trabalho. O quarto capítulo é o responsável por apresentar a metodologia adotada para resolver o problema de navegação segura. Nele está a estrutura da solução, com a percepção do

ambiente e o controle de navegação. Apresentada a metodologia, os resultados experimentais estão descritos no Capítulo 5. Eles foram divididos em duas etapas, sendo a primeira para os testes em um ambiente de simulação e a segunda para os resultados reais no CADU. As conclusões deste trabalho, com seus benefícios e limitações metodológicas estão discutidas no Capítulo 6, o qual apresenta também idéias para trabalhos futuros. Por fim, o Apêndice A traz uma análise da complexidade computacional da solução proposta, para que o leitor compreenda melhor quais etapas que interferem no seu desempenho.

Capítulo 2

Revisão de Literatura

Nas últimas décadas, muito foi pesquisado e desenvolvido no âmbito dos carros autônomos, também conhecidos por *Driverless Cars* ou carros sem motorista. O passo inicial foi dado pelo robô criado no laboratório de engenharia mecânica japonês de Tsukuba no final da década de 70. Este robô foi concebido em uma época onde os computadores ainda eram muito lentos e pesados, sendo criados hardwares específicos para ele. No entanto, os primeiros carros robôs do mundo foram concebidos na Universidade Bundeswehr de Munique (UniBW) pelo professor Ernst Dickmanns [Dickmanns and Zapp, 1987] na década de 80. Seus veículos utilizavam desde visão com movimentos sacádicos a métodos probabilísticos, tais como o Filtro de Kalman, e computação paralela. Algumas fontes norte-americanas comparam Dickmanns ao irmãos Wright[1], tamanha sua importância no desenvolvimento dos carros autônomos.

Com o avanço dos computadores portáteis e sensores, vários veículos mais complexos surgiram na década de 90. Nos Estados Unidos, a Universidade de Carnegie Mellon (CMU) apresentou o robô Navlab 5 com o intuito de realizar o projeto intitulado "*No Hands Across America*", ou Sem as Mãos Percorrendo a America[2]. A viagem foi realizada em 1990, na qual o veículo percorreu 96% dos 490 km autonomamente, com uma velocidade média de 91 km/h. Outro exemplo, também deste período, é o projeto italiano ARGO [Broggi et al., 1999], que igualmente ao Navlab 5, foi capaz de realizar viagens em alta velocidade (média acima de 90 km/h), por longas distâncias (cerca de 2000 km) e a maior parte dela de forma autônoma (94% do tempo).

[1] Os irmãos Orville Wright e Wilbur Wright são considerados os pais da aviação pelos norte-americanos.
[2] `http://www.cs.cmu.edu/afs/cs/usr/tjochem/www/nhaa/nhaa_home_page.html`.

No entanto, os maiores avanços e contribuições ocorreram a partir do ano 2000, principalmente com os desafios propostos pela agência americana DARPA ([DARPA, 2005] e [DARPA, 2008]) entre carros autônomos. Para superar estes desafios, diferentes técnicas foram desenvolvidas e aperfeiçoadas por pesquisadores de todo o mundo. As técnicas eram capazes de enfrentar as adversidades comuns aos veículos autônomos terrestres e realizar a sua navegação autônoma em ambientes desérticos e urbanos. Os desenvolvimentos foram tamanhos que veículos autônomos já podem ser comprados e aplicados em tarefas que apresentam risco aos seres humanos [Jamasmie, 2009].

Na presente dissertação, serão focadas as técnicas necessárias para se realizar a navegação autônoma de robôs, com suas aplicações aos carros autônomos, sem a consideração de regras de trânsito. Assim, este capítulo está dividido nas áreas de planejamento de movimento, detecção e desvio de obstáculos, com uma posterior análise dessas técnicas nos trabalhos relacionados com carros autônomos. O problema de localização, que é de grande importância para a navegação autônoma, não será abordado neste trabalho. O sistema de localização utilizado na parte prática do trabalho, bem como os conceitos por ele abordados, podem ser vistos em [Santos, 2009].

2.1 Planejamento de movimento

O problema de planejamento de movimento, também conhecido por planejamento de caminhos, pode ser definido como o problema de se encontrar um caminho livre de obstáculos entre uma configuração inicial e uma configuração final do veículo. O planejamento deve sempre encontrar um caminho se ele existir, ou retornar que não existe uma solução. A definição apresentada anteriormente é muito ampla para o que de fato é uma junção de vários conceitos e técnicas. Na prática, deve-se conhecer o espaço de configurações do robô, escolher a técnica de planejamento a ser usada e, com a sua localização, realizar o controle da trajetória planejada para a posição atual (para maiores detalhes ver [Choset et al., 2005]). Nesta seção, para o estudo sobre o planejamento de movimento, foi considerado que os ambientes de trabalho do robô são conhecidos e estáticos.

Para criar o espaço de configurações de um robô é necessário conhecer seu espaço de trabalho, ou seja, o ambiente no qual o robô está inserido. O espaço de configurações

descreve todas as configurações do veículo que são livres ou não de obstáculos. Porém, a complexidade computacional para a sua análise e obtenção varia exponencialmente com sua dimensão [Frazzoli et al., 2000], o que pode torná-lo inviável. Para um robô puntual, o espaço de configurações será equivalente ao de trabalho. No entanto, se o robô possuir dimensões, a obtenção do espaço de configurações é, no geral, complexa e não trivial.

Outro fator importante para o planejamento de movimento é considerar ou não as restrições de movimento dos veículos autônomos. Robôs com restrições onde o número de velocidades independentes é menor que o número de variáveis a serem controladas são classificados como não-holonômicos, os quais necessitam de um planejamento e controle mais complexos. Na sua construção, o planejamento deve considerar que o robô não é capaz de realizar qualquer movimento independente. Esta dificuldade pode ser simplificada considerando-o holonômico (sem restrições em seu movimento), pois o planejamento admite que ele conseguirá realizar qualquer movimento proposto.

Por isso, em muitos trabalhos sobre veículos autônomos, classificados como robôs não-holonômicos e cuja pose pode ser expressa por (x, y, θ), o planejamento global é realizado de forma a considerá-lo puntual e holonômico. Assim, para seguir o caminho planejado, deve ser utilizada alguma técnica de controle por realimentação de estados [Luca et al., 1998]. Neste caso, as possíveis colisões devem ser tratadas com técnicas de detecção e desvio de obstáculos, como as que serão apresentadas nas Seções 2.2 e 2.3. Nos trabalhos apresentados no *DARPA Grand Challenge*[3] [DARPA, 2005], por exemplo, o planejamento foi realizado localmente, desviando dos obstáculos detectados, e garantindo apenas que o robô se mantivesse no caminho global. Neste caso, o planejamento global (sem considerar obstáculos) foi fornecido pela própria organização do evento. Alguns exemplos de planejamentos globais para veículos autônomos podem ser encontrados em [Svestka et al., 1995], [Frazzoli et al., 2000] e [Lamiraux and Laumond, 2001]. Outras técnicas aplicadas a robôs não-holonômicos podem ser encontradas em [Laumond et al., 1998].

As técnicas mais comuns de planejamento de movimento baseiam-se na decomposição do espaço de configurações em células marcadas com o estado "livre" ou "ocupado".

[3] Neste desafio realizado em 2005, equipes deveriam cruzar um deserto com veículos autônomos seguindo trilhas e pontos definidos por GPS.

As discretizações mais comuns são obtidas na forma de grades, triangulações e em grafos de visibilidade. As grades (*grids*) são normalmente quadrados com tamanhos iguais ou variados (obtidos por *quadtree*[4] [Berg et al., 2000], por exemplo). O tamanho mínimo de cada quadrado varia com a resolução desejada para descrever os obstáculos, interferindo diretamente no tempo de processamento dos dados e nos caminhos planejados. As discretizações por triangulações e por grafos de visibilidade são geralmente realizadas quando o mapa é poligonal, com a utilização das quinas dos obstáculos para realizar as divisões [Berg et al., 2000].

Com a discretização do mapa, o planejamento, de acordo com [Choset et al., 2005], é geralmente realizado com algoritmos de busca em grades/grafos, navegação em *Roadmaps*, ou na geração de campos potenciais. Os grafos são construídos por meio da conexão de cada célula da grade à sua vizinha com a atribuição de pesos representando o custo para percorrê-la. Com algoritmos tais como o *Dijkstra* e o *A**, o menor caminho a ser percorrido pode ser encontrado. Este, por exemplo, é o mesmo fundamento utilizado na geração de rotas de um GPS veicular. No segundo conjunto de algoritmos são criados os *Roadmaps*, que são caminhos onde o robô pode navegar sem nenhum obstáculo. Nesse caso, a tarefa do robô seria encontrar um ponto de entrada da configuração inicial para o *Roadmap*, e outro de saída, do *Roadmap* para o ponto de destino, para realizar o seu movimento. Um algoritmo bastante comum para a geração de *Roadmaps* é o *BrushFire* [Choset et al., 2005]. Outra técnica para a busca de rotas aplicada em veículos autônomos pode ser observada em [Gonçalves et al., 2010]. Nele, cada posição representa um estado que o carro pode se encontrar no mundo, os quais são modelados como nós de uma árvore de busca. O problema de planejamento é então remodelado com um algoritmo de busca.

O último conjunto de algoritmos é dos que utilizam o princípio atrativo/repulsivo das funções potenciais. Nessa abordagem são geradas funções de potenciais que atraem o robô ao destino e o repele de seus obstáculos. A vantagem em relação às demais técnicas apresentadas anteriormente é que além de guiarem o robô, também realizam o seu controle. O controle é possível porque para cada ponto no espaço pode ser definido um vetor (campo vetorial) que conduz o robô para um local de menor potencial. Utilizando-se técnicas de linearização por realimentação de estados [Luca et al., 1998],

[4] É uma árvore onde cada nó que não seja folha possui interligação com mais 4 nós.

por exemplo, este vetor pode ser convertido em entrada de controle para um robô não-holonômico. Porém, alguns métodos de funções de potenciais podem gerar mínimos locais que não garantem a completude do caminho para o robô. Como forma de solucionar este problema, as funções potenciais foram combinadas com outros elementos gerando duas classes de soluções [Choset et al., 2005]: o primeiro amplia as funções potenciais com algum algoritmo de busca, como os apresentados anteriormente; e o segundo utiliza funções de navegação.

Funções de navegação definem apenas um mínimo, no destino do robô, e garantem a completude até este destino [Rimon and Koditschek, 1992]. Como exemplo mais restrito desta técnica tem-se o *Wavefront*, ou frente de onda. Nele o campo atrativo para o destino é criado a partir de uma "onda" gerada no ponto destino, atribuindo valores crescentes até o robô. Percorrendo-se essas células, tem-se sempre o menor caminho entre a origem e o destino. O porquê de ser restrita está no fato de ser uma técnica de custo elevado para armazenamento e processamento, principalmente para ambientes externos. Existem também técnicas que utilizam funções harmônicas, que são soluções para as equações de Laplace, como funções de navegação [Pimenta et al., 2006]. No entanto, assim como o *Wavefront*, possui um custo elevado de processamento e armazenamento em ambientes externos.

Uma técnica para ambientes externos foi apresentada inicialmente em [Pimenta et al., 2007] e aperfeiçoada em [Pereira et al., 2009], com a aplicação de campos vetoriais que guiam e controlam o veículo autônomo. Nesses trabalhos, os campos vetoriais foram gerados de forma eficiente e contínua dentro de uma sequência de triângulos que delimitam o caminho pelo qual o veículo deve passar. Cada triângulo possui um custo para ser percorrido, o que é importante para um carro autônomo em ambientes externos, pois permite diferenciar, por exemplo, grama de rua pavimentada e não pavimentada e utilizar velocidades distintas em cada situação. Aos triângulos são atribuídos vetores em seus vértices que ponderam o cálculo do vetor em determinado ponto interno a este. Assim, tem-se um método de simples armazenamento e processamento que, apesar de discretizar o espaço de configurações do robô, fornece dados contínuos de velocidade. A Figura 2.1 apresenta um exemplo de trajetória simulada de um robô imerso em um campo gerado por esta técnica de navegação.

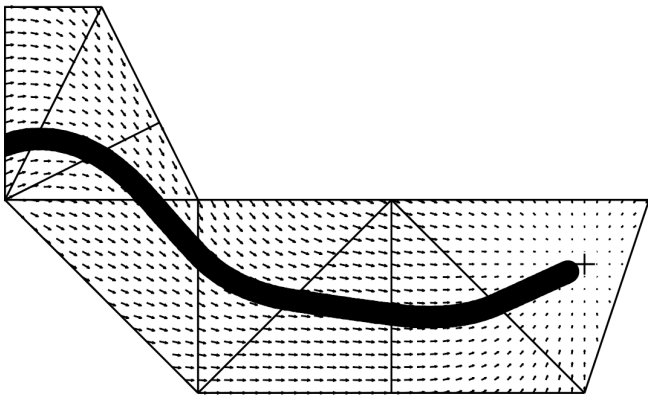


Figura 2.1: Exemplo de planejamento de movimento de um robô por campos vetoriais até um ponto de destino.

2.2 Detecção de obstáculos

Perceber o ambiente é uma tarefa de grande importância para uma navegação segura de veículos autônomos. Assim como os seres humanos percebem o ambiente com os seus diversos sensores, tais como olhos e tato, por exemplo, os robôs também necessitam de sensores para se locomoverem em segurança. Dos sensores existentes e mais utilizados para esse fim têm-se os sonares, os lasers e aqueles baseados em visão computacional [Pereira, 2006]. O uso de cada um deles depende de alguns fatores, entre eles a aplicação final no robô, seu custo/benefício e o poder de processamento dos dados.

Os sonares são um dos sensores de distância mais antigos e amplamente usados na robótica móvel. Funcionam a partir do efeito reflexivo do som nos obstáculos e, por isso, suas leituras podem sofrer com interferências sonoras externas e com as formas dos obstáculos. Dentre os sensores estudados, os sonares são os mais limitados por serem unidirecionais, sendo necessários vários sensores e alguns algoritmos de processamento do seu sinal para se ter uma percepção razoável do ambiente. No entanto, atualmente são muito utilizados em veículos comerciais como sensores de estacionamento e em aplicações mais simples na robótica onde a precisão e a quantidade de leituras não são relevantes [Park et al., 2008].

Para se obter dados com menos erros de medição, os sensores a laser são os mais comuns, possuindo uma varredura em um plano com uma grande precisão e grande alcance de leitura. Das equipes participantes dos desafios *DARPA Grand Challenge* [DARPA, 2005], praticamente todas utilizaram sensores a laser em seus veículos. A

limitação destes sensores está no fato de, em geral, lerem em apenas um plano, o que dificulta o reconhecimento de obstáculos abaixo e acima deste plano. Para superar este problema e realizar uma navegação segura faz-se necessário o uso de mais sensores ou técnicas (como a apresentada em [Patz et al., 2008]) para se ampliar a dimensão de leitura e, assim, perceber o ambiente em sua plenitude. Outro problema destes sensores é o seu custo elevado quando comparado aos demais sensores de distância. Somente recentemente, com as necessidades do *DARPA Urban Challenge* [DARPA, 2008], que foram desenvolvidos sensores a laser 3D como o Velodyne [Halterman and Bruch, 2010], capazes de realizar varreduras de 360° com leituras simultâneas em diferentes planos. Porém, assim como os lasers de varredura planar, eles são de alto custo.

Por fim, os sistemas de câmeras estéreo fornecem uma perspectiva do ambiente em três dimensões, simulando a visão humana. São frequentemente utilizados em problemas de reconstrução geométrica [Skiena, 2004], quando se quer ter uma percepção geral do ambiente. Os problemas com relação a este sistema estão na análise das imagens, processamento dos dados e na robustez quanto à variação de iluminação do ambiente. No entanto, o seu custo é menor que o dos lasers.

O uso de apenas uma câmera como sensor também é comum em muitas aplicações de navegação segura de robôs. Com algoritmos específicos é possível, por exemplo, encontrar linhas para serem seguidas [Leonard et al., 2008], estimar regiões trafegáveis por similaridade [Rauskolb et al., 2008] e detectar as sinalizações ([Bahlmann et al., 2005] e [Moreira et al., 2007]) para respeitar as leis de trânsito.

Neste trabalho, os estudos foram direcionados para aplicações com laser e visão estéreo e suas técnicas para detecção de obstáculos. Para os lasers, a detecção de obstáculos é feita por meio das descontinuidades de suas leituras, estando ele paralelo ou inclinado em direção ao solo [Borges and Aldon, 2004]. Nas aplicações desenvolvidas para carros autônomos, os lasers são preferencialmente inclinados em direção ao solo para explorar ao máximo a alta resolução de suas leituras e para detectar elementos não perceptíveis aos demais sensores em ambiente externo. Em [Wijesoma et al., 2001], por exemplo, esta resolução é explorada para encontrar os limites da rua e do passeio, cujas elevações são relativamente pequenas.

Enquanto a detecção de obstáculos com lasers pode ser realizada de forma mais simples, com visão estéreo as técnicas demandam bastante processamento, o que está

diretamente relacionado com o tamanho das imagens. Para carros autônomos, as técnicas devem ser capazes de distinguir entre planos trafegáveis e possíveis obstáculos. Uma técnica apresentada em [Lee and Lee, 2004], por exemplo, realiza esta distinção e os obstáculos são detectados. Porém, o resultado obtido não se apresentou robusto nas ruas que não são horizontais, com subidas e descidas.

Uma técnica mais robusta e aplicável em superficies não planas foi desenvolvida e aperfeiçoada em [Broggi et al., 2005], [Soquet et al., 2007] e [Labayrade et al., 2002]. A técnica utiliza, como princípio, o chamado mapa de disparidade "V", que será apresentado com maiores detalhes no Capítulo 4. Esta técnica foi aplicada em 2005 no *DARPA Grand Challenge*, com as equipes *Oshkosh Truck Corporation* [Braid et al., 2006] e a da universidade de Princeton [Atreya et al., 2006]. Essa mesma técnica foi adaptada e aplicada pela equipe *Oshkosh Truck Corporation* no *DARPA Urban Challenge* de 2008 [Chen et al., 2008].

Os diferentes sensores e técnicas apresentados anteriormente são capazes de detectar diferentes tipos de obstáculos, mas em uma região restrita do ambiente. Assim, um obstáculo perceptível por um sensor pode não ser por outro. A combinação e a adição de sensores, no entanto, permite descrever o espaço de trabalho do robô com melhor qualidade, utilizando o que cada um tem de melhor. No caso dos veículos autônomos, dificilmente é encontrado na literatura algum veículo que somente use um sensor. O mais comum é uso de vários lasers espalhados pelo veículo, com diferentes inclinações, como também a combinação de câmeras e lasers que permitam uma percepção ampla do ambiente [Newman et al., 2009, Perrollaz et al., 2006, Wijesoma et al., 2001].

No entanto, o uso de vários sensores necessita que sejam utilizadas técnicas que combinem seus dados, principalmente quando estes realizam leituras em uma mesma região do espaço, e, assim, resultem em apenas uma única leitura por região. Este é o princípio da fusão sensorial, combinar os dados e eliminar os pontos de dúvida, onde um obstáculo é percebido por um sensor mas não por outro. Várias são as técnicas para este fim, sendo comum o uso de técnicas probabilísticas [Thrun et al., 2005] na análise dos dados. Quando não é possível o uso de vários sensores, ou existem regiões do ambiente sem o monitoramento, suas leituras podem ser armazenadas em uma grade de ocupação e atualizadas a partir da posição do robô (ver [Thrun et al., 2005] para maiores detalhes).

2.3 Desvio de obstáculos

Evitar obstáculos durante o movimento de um robô é fundamental para sua integridade física e para este poder ser considerado realmente autônomo. Ao longo da história de desenvolvimento da robótica móvel, várias foram as técnicas criadas para realizar o desvio de obstáculos. Segundo [Minguez et al., 2008], as técnicas de desvio de obstáculos podem ser divididas em dois grupos: o dos métodos que calculam o movimento em uma etapa e dos que o calculam em várias etapas.

Os métodos do primeiro grupo transformam a informação dos sensores diretamente em controle de movimento, englobando os métodos heurísticos e os baseados em analogias físicas. Seus princípios baseiam-se nos métodos apresentados na Seção 2.1 para o planejamento de movimento. Desta forma, o robô realiza o desvio de obstáculos ao longo de um planejamento de movimento local. Os métodos heurísticos utilizam algoritmos derivados da pesquisa em grades e grafos, enquanto os baseados em analogias físicas referem-se ao princípio de geração de campos potenciais atrativos e repulsivos. Existem várias aplicações destas técnicas em veículos autônomos, mas utilizá-las depende algumas vezes de um controle de trajetória adequado. Como exemplo de método heurístico tem-se o trabalho de [Kolski and Macek, 2007], onde é apresentada uma solução baseada no algoritmo de planejamento por busca em grades chamado *E** e em [Krogh and Thorpe, 1986] tem-se uma aplicação baseada na geração de campos potenciais locais. Outros exemplos deste grupo podem ser encontrados em [Ferreira et al., 2004] e [Pereira, 2006].

No segundo grupo estão as técnicas que computam alguma informação intermediária com os dados do sensor, sendo dividido nos métodos que geram subconjuntos de controle e nos métodos que calculam uma informação de mais alto nível. Os métodos que geram subconjuntos de controle são bastante aplicados atualmente na robótica móvel em geral. Dentre as principais técnicas tem-se: o Histograma de Campos Vetoriais, o Método da Janela Dinâmica, o Método da Velocidade de Curvatura e o Método dos Obstáculos com Velocidade. O Histograma de Campos Vetoriais (VFH)[5] é uma técnica desenvolvida em [Borenstein and Koren, 1991] e aperfeiçoada em [Ulrich and Borenstein, 1998] e [Ulrich and Borenstein, 2000] que gera comandos para direção de

[5] Do inglês *Vector Field Histogram*.

movimento. Nela é analisada a distância entre o robô e os obstáculos próximos por meio de um histograma, retornando um vetor direcionado para a região mais livre para locomoção e próximo a um ponto de destino. Este princípio foi utilizado em [Lima and Pereira, 2010] para realizar o desvio de obstáculos por um carro autônomo.

No entanto, o uso de comandos de direção não é a única forma de atuar em um robô para realizar o desvio de obstáculos. Os dois próximos métodos, por exemplo, consideram a dinâmica do robô para gerar comandos de velocidade para o mesmo. Os métodos da Janela Dinâmica (DWA)[6] [Fox et al., 1997] e das Curvas de Velocidade (CVM)[7] [Simmons, 1996] possuem o mesmo princípio teórico e as formas de análise são muito parecidas. Ambos realizam cálculos para encontrar o melhor conjunto de controle que respeite, conjuntamente, a melhor velocidade linear do robô, a maior distância segura para o obstáculo e o melhor direcionamento para o destino final. Essas informações são compostas e, ao final, é realizada uma otimização local para obter o melhor resultado de trajetória segura para o conjunto de controle retornado. Apesar de serem semelhantes, o desenvolvimento de ambos foi em separado e sem o conhecimento da existência de um ou outro. A grande vantagem entre esses dois métodos e os demais apresentados até o momento está no fato de mapearem as possíveis velocidades de atuação no robô para realizar a sua análise. Isso permite que os comandos possam ser aplicados diretamente ao robô e garantam o seu total controle.

Como o DWA e o CVM realizam seus cálculos otimizando uma série de regras, é possível alterá-las para aprimorar o funcionamento de ambos e torná-los mais flexíveis. Alterá-las, por exemplo, permite aplicá-las em carros autônomos, como em [Rebai et al., 2007]. Nesse trabalho, o DWA foi aplicado como uma alternativa para o controle de um carro em altas velocidades. Mas os trabalhos mais significativos se basearam no CVM, colocando em prova sua eficiência nos desafios do DARPA. A equipe vencedora do desafio de 2005, por exemplo, fez uso de uma técnica baseada no CVM em seu veículo Stanley [Thrun et al., 2006] (Figura 1.2), o método das Curvas de Caminho (LCM)[8] [Ko and Simmons, 1998]. Enquanto o CVM gera comandos de velocidade para desviar dos obstáculos, o LCM gera trajetórias em curva. Apesar da diferença, o princípio é semelhante e comprova a eficiência dos métodos para o desvio de obstáculos.

[6] Do inglês *Dynamic Window Approach*.
[7] Do inglês *Curvature-Velocity Method*.
[8] Do inglês *Lane-Curvature Method*.

Porém, o DWA e o CVM não consideram que os obstáculos tenham velocidades. Na prática, quando obstáculos possuem velocidades menores que o robô, a atualização da posição dos obstáculos pelo DWA e CVM é suficiente para os cálculos do método. No entanto, para velocidades maiores, é necessário considerá-las nos cálculos de trajetórias possíveis. Uma solução foi apresentada por [Fiorini and Shillert, 1998], com o Método dos Obstáculos com Velocidade (VO)[9]. Com essa técnica, utilizando o mesmo quadro de velocidades do DWA e CVM, é possível considerar a velocidade dos obstáculos para realizar os cálculos das trajetórias seguras, o que é de grande importância para ambientes dinâmicos como os das vias urbanas.

Com relação aos métodos que calculam uma informação de mais alto nível, a técnica que descreve bem este conjunto é o Diagrama de Proximidade[10] [Minguez and Montano, 2000] e [Minguez et al., 2004]. Esta técnica utiliza a estratégia do algoritmo divisão e conquista (*divide-and-conquer*) baseado em situações que simplifiquem o problema de desvio de obstáculos. A cada situação é atribuída uma lei de controle para que, durante a execução, uma situação seja identificada e a lei correspondente possa ser utilizada. Por mapear todas as situações possíveis, este é um método cuja saída é altamente preditiva. Assim como uma adaptação do CVM foi bem utilizada pelo veículo Stanley, o ND foi aplicado pela equipe Prospect Eleven [Atreya et al., 2006] neste mesmo desafio. Dentro dos métodos de mais alto nível, tem-se também técnicas de desvio baseadas em lógica Fuzzy, que não serão abordadas neste trabalho. Um exemplo pode ser visto em [CARVALHO JUNIOR, 2007].

No entanto, mesmo com essa diversidade de técnicas para o desvio de obstáculos, o robô ainda pode encontrar um estado de colisão inevitável (ICS)[11] e ficar sem reação. Atualmente, estão sendo estudas muitas técnicas para evitar que o robô se encontre em um ICS. Em [Bekris, 2010] e [Shiller et al., 2010], por exemplo, tem-se exemplos de estudos recentes que incorporam o trato com os ICS nos métodos de desvio que calculam o movimento em uma etapa e nos que calculam em mais etapas.

[9] Do inglês *Velocity Obstacles*.
[10] Do inglês *Nearness Diagram*.
[11] Do inglês *Inevitable Collision State*.

2.4 Trabalhos relacionados

As técnicas apresentadas nas seções 2.1, 2.2 e 2.3 são comuns em diversas aplicações da robótica móvel, nas quais incluem-se os carros autônomos. Porém, planejar um movimento não é suficiente para que o robô se mova em segurança, pois um novo obstáculo pode aparecer e impedir a sua realização. Igualmente, detectar um obstáculo não terá significado se o robô não souber como evitá-lo. Portanto, para realizar a navegação autônoma de robôs é comum usar em conjunto técnicas de planejamento de movimento, detecção e desvio de obstáculos, além das técnicas de localização, entre outras. Apesar de importante no processo de navegação, o uso de diferentes técnicas de detecção de obstáculos, geralmente, não interfere diretamente no resultado da navegação, desde que os obstáculos sejam reconhecidos corretamente. O que não acontece com as técnicas de planejamento de movimento e desvio de obstáculos, que podem alterar o controle do robô. Há vários trabalhos que aplicam apenas uma delas, o que resulta em alguma limitação no movimento global desse. Os exemplos desta seção serão direcionados para aplicações em carros autônomos.

Em [Bemporad et al., 1996] foi apresentada uma solução utilizando campos de forças locais para alterar o sentido de movimento de um carro autônomo. O método consistia em criar, a partir dos obstáculos detectados, forças repulsivas e atrativas a um ponto de destino para gerar comandos de atuação no veículo por uma técnica de controle por realimentação de estados. Porém, como o método não possuía um planejamento de movimento, não era possível garantir a sua completude.

Uma outra solução local para carros autônomos pode ser observada em [Rebai et al., 2007]. Nesse trabalho, depois de uma detecção prévia dos obstáculos do ambiente, o método da janela dinâmica (DWA) foi aplicado para o desvio de obstáculos. Com esse recurso, o veículo podia movimentar-se com segurança mesmo com velocidades mais elevadas, pois a técnica prioriza as altas velocidades. Essa técnica possui uma vantagem em relação à apresentada por [Bemporad et al., 1996] que permite torná-la global com mais facilidade. Como o DWA é composto por funções calculadas em relação às velocidades linear e angular do carro, é possível alterar e criar funções que respeitem um planejamento de movimento realizado previamente e que garanta, além do desvio, a completude do método. Em [Brock and Khatib, 1999], por exemplo, foi adicionada

uma função de navegação ao DWA baseada no algoritmo de frente de ondas (do inglês *wavefront*), que permitiu a navegação autônoma global de robôs não-holonômicos. Por ser um método dinâmico, para obstáculos com velocidades significativamente inferiores à do carro, ele consegue garantir a segurança do movimento com a atualização da posição dos obstáculos para a nova janela. Para os casos onde é necessário considerar a velocidade dos obstáculos tem-se o trabalho de [Seder and Petrovic, 2007].

No geral, as aplicações que realizam a navegação autônoma o fazem de duas maneiras, baseadas em um planejamento de movimento inicial e em uma técnica de desvio de obstáculos. A primeira utiliza técnicas de desvio de obstáculos que alteram o planejamento de movimento inicial, enquanto a segunda aplica alguma técnica de desvio de obstáculos que não altere o planejamento inicial, na tentativa de segui-lo sempre que possível. As aplicações do primeiro conjunto calculam o movimento em uma etapa para realizar o desvio de obstáculo e realizam um planejamento de movimento local, como visto na Seção 2.3. Em [Kolski and Macek, 2007], por exemplo, a navegação autônoma é realizada com um planejamento de movimento de busca em grades com o algoritmo *E**, o mesmo algoritmo utilizado para o replanejamento local e desvio de obstáculos. Esta é uma solução cujo custo computacional depende do tamanho das células da grade e necessita de um controlador específico para seguir a trajetória gerada. É possível, também, "deformar" o planejamento inicial com a ação de campos potenciais para que trajetória final contorne os obstáculos [Lefebvre et al., 2004].

Apesar dos dois exemplos anteriores serem aplicados a carros autônomos, a navegação autônoma realizada da segunda maneira é a mais encontrada na prática. Um provável motivo é que as aplicações práticas que envolvem esses veículos definem rotas a serem realizadas que já são completas. Apesar de não considerarem a presença de obstáculos, alterar essas rotas, além de custoso, pode não garantir a completude da nova rota. Nos desafios *DARPA Grand Challenge* [DARPA, 2005] e *Urban Challenge* [DARPA, 2008] para veículos autônomos terrestres, por exemplo, estão os trabalhos mais significativos que aplicam a segunda forma de navegação.

Os desafios *DARPA* foram altamente produtivos para a divulgação de novas pesquisas e desenvolvimento dos carros autônomos. Várias técnicas discutidas no presente trabalho foram utilizadas em equipes desses desafios. O *Grand Challenge* foi realizado em um

deserto com um caminho definido pelo RDDF[12], onde o vencedor deveria completá-lo no menor tempo. Resumidamente, os veículos deveriam trafegar por pontos via (*waypoints*), que marcavam a trilha no deserto, cuja largura variava entre 3 e 30 metros. Além disso, eles teriam que perceber e desviar dos obstáculos (rochas, plantas e os demais robôs participantes), como garantia de sua integridade. O *Urban Challenge* por sua vez, realizou-se em um ambiente urbano cinematográfico. O RNDF[13] definia a distribuição das ruas, marcando pontos de caminho gerais, linhas de pare e o ponto de entrada ou saída que conectavam as pistas, já o MDF[14] definia os pontos de caminho de cada equipe. Por se tratar de um ambiente urbano, as tarefas a serem cumpridas eram mais complexas que as do deserto. O ambiente urbano possui mais regras a serem respeitadas, obstáculos que se movem com diferentes velocidades, necessitando de técnicas mais arrojadas e específicas para superar tais problemas. Pelas funcionalidades abordadas, as soluções apresentadas no *DARPA Grand Challenge* se adequam bem à proposta deste trabalho.

A equipe da universidade de Princeton, por exemplo, participou do *Grand Challenge* com o carro autônomo *Prospect Eleven* [Atreya et al., 2006]. Com apenas sensores de visão estéreo para detectar obstáculos, este veículo apresentou uma solução simples e funcional para ambientes desérticos. Inicialmente, o robô utilizou os dados do RDDF para seguir o caminho desejado, sem um planejamento específico e com apenas um controle de direção. Para detectar os obstáculos com a visão estéreo, utilizou-se a técnica apresentada em [Broggi et al., 2005], baseada no mapa de disparidade "V", que é uma solução amplamente difundida para este tipo de aplicação. Com a informação de obstáculos, a segurança do caminho realizado a partir do RDDF foi garantida com um método de desvio de obstáculos, o Diagrama de Proximidade (ND) [Minguez and Montano, 2000].

Também participante do *Grand Challenge*, a equipe *Oshkosh Truck Corporation*, com o seu veículo TerraMax [Braid et al., 2006], utilizou um sistema de detecção de obstáculos baseado em sensores a laser e sistemas estéreo trinoculares. Os lasers foram posicionados para identificar obstáculos positivos perto do veículo e bordas negativas (buracos) nas trilhas, pela descontinuidade dos dados recebidos. O sistema de visão

[12] Do inglês *route definition data format*.
[13] Do inglês *road network definition file*.
[14] Do inglês *mission data file*.

estéreo trinocular foi utilizado para detectar tanto obstáculos quanto o espaço livre e o caminho a ser seguido pelo robô, obtidos pela técnica do mapa de disparidade "V", também utilizada em [Atreya et al., 2006]. O uso de três câmeras fez-se necessário por permitir que o veículo trabalha-se bem em pequenas distâncias (até 20 metros) e em grandes distâncias (entre 20 e 60 metros). Os obstáculos foram então armazenados em um banco de dados específico e, juntamente com o espaço livre, foram utilizados por um algoritmo de planejamento de caminho em tempo real. O algoritmo de planejamento baseou-se no trabalho de [KUFFNER JR. and Lavalle, 2000] e teve como princípio a criação de árvores aleatórias de exploração rápida que geravam caminhos aleatórios, mas que respeitavam os dados do RDDF. Por fim, a navegação segura foi garantida por meio de uma arquitetura distribuída baseada em lógicas de decisão.

O principal exemplo dentre os trabalhos do *DARPA Grand Challenge* é o veículo Stanley [Thrun et al., 2006] (Figura 1.2) desenvolvido pela equipe *Stanford Racing Team*, vencedor da edição. Com um complexo sistema de cinco sensores a laser e uma câmera de video, o veículo foi capaz de completar todo o percurso em segurança e com altas velocidades. O mapeamento realizado com os lasers foi a principal fonte de detecção de obstáculos, associados a descontinuidades nos dados. Porém, a equipe verificou que os lasers sozinhos garantiam um mapeamento de no máximo 22 m, limitando a velocidade do carro em 25 mph ($\simeq$ 45 km/h). Para ampliar esse limite, foi analisado o terreno à frente do veículo com uma câmera de video, de onde extraiu-se a região trafegável nele presente. O resultado ampliou o mapeamento para até 70 m e permitiu velocidades superiores a 35 mph ($\simeq$ 60 km/h). Como foram utilizados vários sensores, a combinação e armazenamento dos dados adquiridos fez-se em uma grade de ocupação [Thrun et al., 2005], técnica probabilística que permite estimar a presença ou não de um obstáculo em certa região do espaço.

A navegação autônoma do Stanley teve como fundamento a criação de trajetórias a serem seguidas. Primeiramente, o planejamento de movimento global, fornecido pelo RDDF, foi suavizado por aproximações polinomiais locais, com a imposição de que o caminho não poderia sair do túnel definido pelos pontos de caminho e seus raios. O próximo passo foi então validar a trajetória com alguma técnica de desvio de obstáculos. No caso, o método das Curvas de Caminho (LCM) [Ko and Simmons, 1998], método derivado das Curvas de Velocidade (CVM) [Simmons, 1996], foi utilizado. Por trabalhar

com trajetórias, a equipe precisou criar um controle específico para poder garantir a sua realização pelo robô. Os avanços apresentados pela equipe *Stanford Racing Team* foram tão importantes para o evento, que praticamente todas as equipes do *Urban Challenge* referenciaram seus trabalhos.

Nos trabalhos do *DARPA Urban Challenge*, todos realizaram a navegação autônoma de seus veículos com técnicas de planejamento e desvio de obstáculos. Porém, as técnicas utilizadas são complexas e vão além do escopo deste trabalho. Para compreender mais sobre essas técnicas, recomenda-se ver os trabalhos das equipes melhores colocadas na competição, a *Tartan Racing* [Urmson et al., 2008], a *Stanford Racing Team* [Montemerlo et al., 2008] e a *VictorTango* [Bacha et al., 2008]. Para desenvolvimentos futuros seus conceitos deverão ser melhor explorados.

O presente trabalho propõe o desenvolvimento de um sistema de navegação segura para um veículo autônomo e o seu controle por comandos de velocidade. Este tem como base a utilização de um planejamento de movimento por campos vetoriais de velocidade aliado a uma técnica de desvio de obstáculos que valide este campo para cada posição do veículo. Para permitir a validação, em paralelo será criado um mapa local dos obstáculos ao redor do veículo, que não se limite ao campo de visão dos sensores utilizados, baseada na técnica da grade de ocupação [Thrun et al., 2005]. Caso as velocidades geradas pelo campo resultem em colisão no mapa, novas velocidades serão calculadas pelo algoritmo de desvio de obstáculos. Para simplificar o problema, foram considerados apenas ambientes em que os obstáculos não se movam ou possuam velocidade muito menor à do carro.

A escolha das técnicas baseou-se nos bons resultados obtidos pelas equipes participantes do *DARPA Grand Challenge* [DARPA, 2005], como também de alguns trabalhos específicos apresentados neste capítulo. Foram priorizadas técnicas que fornecessem comandos de velocidade para realizar o planejamento de movimento e o desvio de obstáculos e, consequentemente, controlar o robô sem a necessidade de um controlador de trajetórias específico. Na detecção de obstáculos procurou-se respeitar as limitações sensoriais, mas sem deixar de garantir segurança ao sistema final. O próximo capítulo apresentará alguns conceitos preliminares e recursos necessários para a elaboração deste trabalho.

Capítulo 3

Conceitos Preliminares

Para realizar a navegação de um veículo autônomo como proposto, diversos conceitos e recursos são necessários. Conhecer o modelo que rege a sua cinemática, realizar leituras constantes de sua pose e ter um controle de velocidade adequado são apenas alguns dos elementos importantes para controlar o seu movimento e, assim, permitir a navegação autônoma. Em etapas, o controle de movimento pode ser expresso de acordo com a Figura 3.1. Baseado na revisão de literatura, este capítulo abordará os conceitos necessários para o controle do carro juntamente com os princípios por trás das técnicas de navegação utilizadas.

Figura 3.1: Etapas necessárias para se realizar o controle de movimento de um carro autônomo.

3.1 Modelagem do carro

Apesar de não estar explicitamente descrito na Figura 3.1, para se controlar um carro é importante conhecer o modelo que o descreve. Com o modelo é possível prever como o sistema irá se comportar com a aplicação de determinadas entradas, o que é importante para todas as etapas listadas na figura. Para um carro, o modelo deve associar as possíveis atuações com as variáveis que descrevem a sua pose.

Existem diversas formas de se levantar o modelo de um robô (sistema), que dependem das informações que se tem sobre o mesmo. É importante lembrar que todo modelo

é uma aproximação da realidade, onde sua complexidade está diretamente associada com o quão próximo do real é este modelo. Segundo [Aguirre, 2007], a modelagem pode ser classificada como caixa branca ou caixa preta. A modelagem caixa branca considera os elementos físicos do sistema para levantar o modelo. Já a modelagem caixa preta é realizada por meio de uma série de experimentos, sem as características físicas envolvidas. No caso de veículos autônomos, cuja física é bem conhecida, o modelo aproximado é normalmente criado a partir da modelagem caixa branca de sua cinemática.

Na literatura estudada, existem diferentes modelos aproximados para um veículo ([Bemporad et al., 1996], [Egerstedt et al., 1998], [Rebai et al., 2007] e [Luca et al., 1998]). Estes modelos dependem, principalmente, de sua tração e do quão real deve ser a descrição da direção de movimento do veículo. A tração, que pode ser dianteira ou traseira, interfere no sentido da velocidade fornecida ao carro pelo torque das rodas. A descrição da direção interfere em como será o movimento do carro em curvas. Como cada uma das rodas frontais, quando direcionadas para uma curva, possuem ângulos diferentes, a direção final precisa ser aproximada de alguma forma. O modelo mais comum para aproximar a direção é o modelo de Ackerman [Choset et al., 2005], o qual aproxima o carro a uma bicicleta, com a direção final sendo a média dos ângulos das duas rodas frontais.

Neste trabalho, foi considerado um modelo para veículos com tração nas rodas dianteiras, ou seja, a velocidade é a das rodas frontais e a aproximação da direção é de acordo com o modelo de Ackerman. O modelo cinemático pode ser encontrado em [Luca et al., 1998] e pode ser expresso como:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \\ \dot{\phi} \end{bmatrix} = \begin{bmatrix} \cos\theta\cos\phi \\ \sin\theta\cos\phi \\ \sin\phi/l \\ 0 \end{bmatrix} v_1 + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} v_2, \tag{3.1}$$

onde a configuração do veículo pode ser expressa por (x, y, θ, ϕ), com (x, y) igual à posição do seu referencial $\{\mathcal{R}\}$, θ igual ao ângulo entre esse referencial e o do mundo $\{\mathcal{O}\}$ e ϕ igual ao ângulo de esterçamento das rodas dianteiras aproximado. As entradas

deste modelo são as velocidades linear das rodas dianteiras v_1 e da variação angular do esterçamento destas rodas v_2. A constante l é a distância entre os eixos traseiro e dianteiro. A Figura 3.2 ilustra estas variáveis. Nesta figura o referencial $\{\mathcal{R}\}$ está orientado segundo ($x_\mathcal{R}$, $y_\mathcal{R}$). Ao centro do eixo dianteiro pode ser visualizada a roda virtual gerada pelo modelo de Ackerman, com a média do ângulos das rodas frontais. É importante frisar que v_1 é diferente da velocidade linear do veículo, dada por $v_1 \cos\phi$.

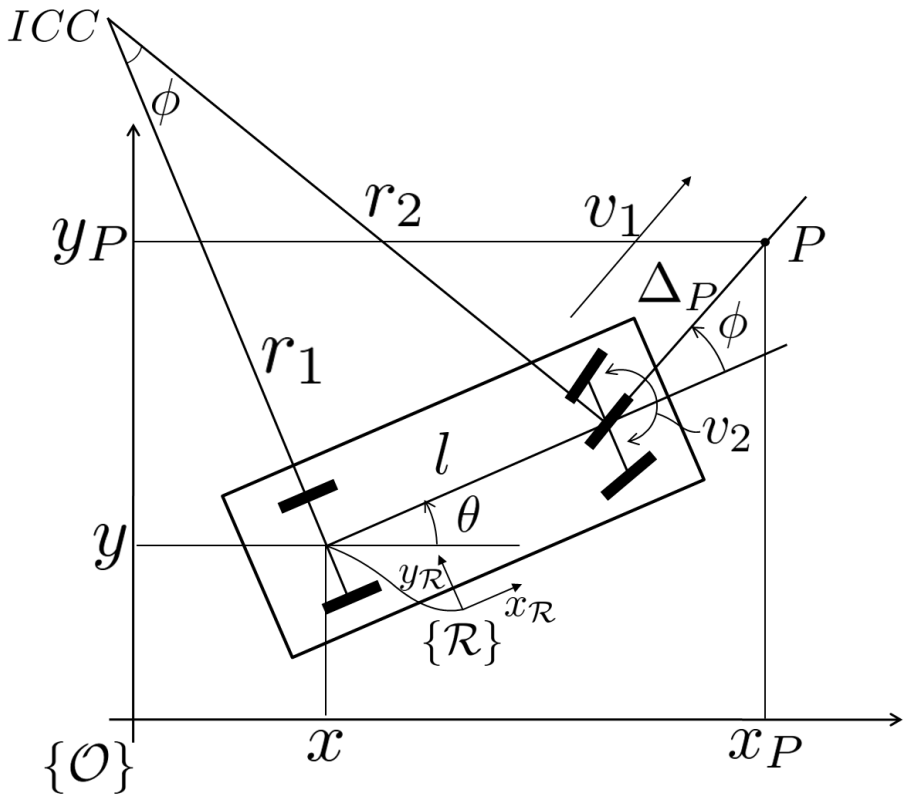


Figura 3.2: Diagrama descritivo do modelo cinemático do veículo (3.1), onde estão representadas sua configuração (x, y, θ, ϕ) e suas entradas v_1 e v_2. Neste modelo, o veículo se move por trajetórias circulares definidas pelo centro de curvatura instantâneo (ICC). O ponto $P = (x_P, y_P)$ é a projeção necessária para realizar a linearização por realimentação estática, apresentada na Seção 3.4. O tamanho desta projeção é definido por Δ_P.

O modelo apresentado considera que o veículo move-se em trajetórias circulares definidas pelo centro de curvatura instantâneo (ICC), com o raio definido por r_1 (ver Figura 3.2). Neste caso o veículo possuirá uma velocidade angular definida por:

$$\dot{\theta} = \frac{v_1 \cos\phi}{r_1} = \omega. \tag{3.2}$$

Para o caso de $r_1 = \infty$, a velocidade angular será igual a zero e o veículo se moverá em linha reta. A partir do modelo cinemático apresentado é possível descrever o movimento do veículo em situações ideais. Isto é feito por meio das equações de movimento (3.3). Com elas é possível determinar como serão as mudanças na configuração do veículo para um pequeno intervalo de tempo $\triangle t$.

$$\begin{cases} x(t+\triangle t) = x(t) + \triangle t\dot{x}(t), \\ y(t+\triangle t) = y(t) + \triangle t\dot{y}(t), \\ \theta(t+\triangle t) = \theta(t) + \triangle t\dot{\theta}(t), \\ \phi(t+\triangle t) = \phi(t) + \triangle t\dot{\phi}(t). \end{cases} \tag{3.3}$$

3.2 Localização

A localização é a primeira etapa apresentada na Figura 3.1 para realizar o controle de movimento de um carro autônomo. Sua importância está no fato de ser a responsável por fornecer a pose (x, y, θ) do veículo para o planejamento de movimento. Um estudo importante nesta área foi proposto por [Santos, 2009], onde se desenvolveu um sistema de localização capaz de utilizar dados de alguns sensores em conjunto para estimar a trajetória de veículos autônomos terrestres. Esse é o sistema de localização presente neste trabalho e seu princípio de funcionamento será brevemente apresentado nesta seção.

O método se inicia com a leitura de dados proveniente de 4 sensores distribuídos no veículo, dentre os quais estão um Sistema de Posicionamento Global (GPS), uma Unidade de Medição Inercial (IMU), um sensor de posição angular do volante e sensores de velocidade das rodas dianteiras[1]. Com base no referencial definido por ($x_{\mathcal{R}}$, $y_{\mathcal{R}}$) na Figura 3.2, as medidas fornecidas pelos sensores em questão para o veículo são: a posição dos eixos $x_{\mathcal{R}}$ e $y_{\mathcal{R}}$, a aceleração no eixo $x_{\mathcal{R}}$, a posição angular em torno do eixo $z_{\mathcal{R}}$, a posição angular do volante e a velocidade das rodas dianteiras.

Em primeira análise, os dados recebidos são suficientes para retornar a pose do veículo. No entanto, ao analisar os dados do GPS por exemplo, verifica-se que ele é capaz fornecer dados de posição (x, y) a uma taxa limitada em 1 Hz. Na prática, esta taxa é muito baixa para fornecer dados de posição a um veículo movendo-se apenas com

[1] Para maiores detalhes sobre os sensores utilizados ver [Santos, 2009].

este sensor, isto sem considerar o erro de suas medidas, que não é pequeno. Esta falta de informação gerada pela leitura do GPS é comum também aos demais sensores em questão, pois apesar de possuírem uma taxa de leitura maior, os dados são assíncronos e constantemente algum deles não estará presente para compor a pose final do carro.

A solução proposta por [Santos, 2009] foi, então, utilizar um modelo cinemático semelhante ao apresentado na Seção 3.1 em conjunto com um algoritmo de filtragem de Kalman [Kalman, 1960]. O Filtro de Kalman (KF) realiza a estimação dos próximos estados de interesse do veículo. Para tanto, o KF considera os erros provenientes dos sensores para realizar seus cálculos e retornar dados mais confiáveis. Esta solução permite que, mesmo com a falta de algum dos sensores listados, o modelo seja capaz de retornar a pose estimada do veículo, a uma frequência superior à do GPS.

3.3 Campos vetoriais de velocidade

O uso de técnicas baseadas em campos vetoriais é bastante comum na literatura para se realizar o planejamento de movimento de robôs, como pôde ser observado em alguns dos trabalhos apresentados na Seção 2.1. Para compor o controle de movimento do veículo autônomo foi escolhida uma destas técnicas. Esta técnica foi apresentada em [Pimenta et al., 2007] e [Pereira et al., 2009], e será discutida nesta seção como parte da etapa de planejamento apresentada na Figura 3.1.

O problema é definido para um robô $\mathcal{R}$ em um mundo $\mathcal{W}$, com a configuração $\mathbf{q}$ representada no seu espaço de configurações $\mathcal{C}$ e o seu espaço de configurações livre representado por $\mathcal{F} \subseteq \mathcal{C}$. Para simplificar a abordagem, o robô é considerado puntual e holonômico, com sua pose definida por (x, y). Isto gera um espaço de configurações bidimensional igual ao seu espaço de trabalho (mundo). O problema em questão é então guiar o robô de sua configuração inicial $\mathbf{q}_0 \in \mathcal{F}$ para uma configuração de destino $\mathbf{q}_g \in \mathcal{F}$, em um tempo finito.

A representação do mundo é uma das principais diferenças entre as técnicas estudadas. Em [Pimenta et al., 2007], o mundo é representado apenas como espaço livre e espaço ocupado (por exemplo passeios e ruas), expresso por $\mathcal{W} = \{(x, y)|x_{min} \leq x \leq x_{max}; y_{min} \leq y \leq y_{max}\}$. Desta forma um veículo teria seu movimento igual para todas as regiões classificadas como espaço livre. Já em [Pereira et al., 2009], o mundo é re-

presentado como um mapa temático que considera as variações espaciais do ambiente. O mapa então passa a ser definido por $\mathcal{W} = \{(x, y, g(x, y))|x_{min} \leq x \leq x_{max}; y_{min} \leq y \leq y_{max}\}$, onde $0 \leq g(x, y) \leq +\infty$ é a função de custo para atravessar uma dada região (x, y) por unidade de distância. Assim, a região inválida é definida como $g(x, y) = +\infty$, o que permite que um veículo planeje seu movimento por regiões de menor custo e considere as variações das vias (tais como terra, asfalto ou calçamento) em seu deslocamento.

Os campos vetoriais propostos por [Pimenta et al., 2007] e [Pereira et al., 2009] têm como objetivo fornecer pares vetoriais $\mathbf{u}(\mathbf{q}) = (\mathbf{v}_{\mathbf{q}_x}, \mathbf{v}_{\mathbf{q}_y})$ para todo ponto no espaço onde tenha um campo definido, de maneira que este seja contínuo em toda a sua extensão e guie o robô. Neste trabalho, os pares vetoriais serão utilizados para representar velocidades nos respectivos eixos de coordenadas do mundo a serem aplicadas a um robô com dinâmica $\dot{\mathbf{q}} = \mathbf{u}(\mathbf{q})$.

O cálculo do campo se inicia com a discretização do espaço $\mathcal{F}$ onde o robô deverá se mover. Para tanto é adotada a Triangulação de Delaunay com Restrições (CDT) [Berg et al., 2000], onde os triângulos que a compõe possuem os ângulos internos maximizados e respeitam as bordas de $\mathcal{C}$. Os triângulos gerados são então numerados pelas faces de f_0 até f_n, com o ponto de configuração inicial $\mathbf{q}_0 \in f_0$ e o de destino $\mathbf{q}_g \in f_n$. Estes triângulos são então armazenados em grafos, conectados aos seus vizinhos e com os respectivos pesos para serem percorridos. Neste caso, a busca pelo melhor caminho entre $\mathbf{q}_0$ e $\mathbf{q}_g$ pode ser realizada com algoritmos tais como o *Dijkstra* ou *A**, retornando a sequência dos triângulos a serem percorridos. Na Figura 3.3 está representada um exemplo deste planejamento em etapas, com a triangulação do espaço e busca pelo melhor caminho.

Com o caminho definido pela sequência de triângulos, o próximo passo é definir o campo vetorial contínuo $\mathbf{u}(\mathbf{q})$ que guie o robô de $\mathbf{q}_0 \in f_0$ para $\mathbf{q}_g \in f_n$. Isto é feito a partir vetores base definidos para cada vértice dos triângulos da sequência planejada e que respeitem a uma série de restrições listadas em [Pimenta et al., 2007] e [Pereira et al., 2009], onde sua obtenção também é abordada. Para uma dada face f_l, com vetores base $\mathbf{v}_{\mathbf{q}_i}$, $\mathbf{v}_{\mathbf{q}_j}$ e $\mathbf{v}_{\mathbf{q}_k}$ e seus respectivos vértices (x_i, y_i), (x_j, y_j) e (x_k, y_k), ordenados

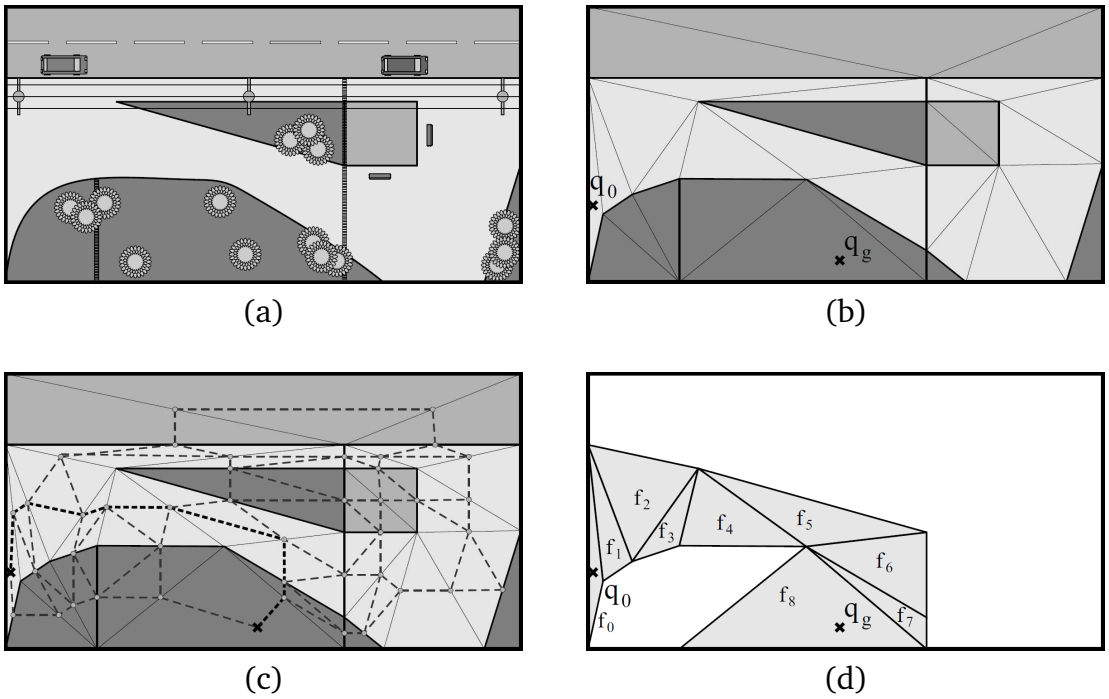


Figura 3.3: Exemplo da metodologia de planejamento por campos vetoriais [Pereira et al., 2009]. Nesta composição, um ambiente externo típico de um robô (a) é discretizado segundo a CDT (b). Com a discretização e os pontos de início $\mathbf{q}_0$ e destino $\mathbf{q}_g$ definidos, é criado um grafo de visibilidade (c). Por fim, tem-se a sequência de triângulos que representa o melhor caminho (d).

em sentido anti-horário, o campo vetorial $\mathbf{u}(\mathbf{q})$ pode ser então obtido pela combinação convexa destes vetores, de acordo com a Equação (3.4).

$$\mathbf{u}(\mathbf{q}) = \frac{A_i \mathbf{v}_{\mathbf{q}_i} + A_j \mathbf{v}_{\mathbf{q}_j} + A_k \mathbf{v}_{\mathbf{q}_k}}{A_i + A_j + A_k}. \tag{3.4}$$

Nesta equação A_i, A_j e A_k são as áreas dos triângulos gerados pela conexão dos vértices $\mathbf{q}_i$, $\mathbf{q}_j$ e $\mathbf{q}_k$ de um dado triângulo com o ponto $\mathbf{q}$. Estes parâmetros podem ser melhor visualizados na Figura 3.4. Possíveis descontinuidades em $\mathbf{u}(\mathbf{q})$ são tratadas durante a obtenção dos vetores base.

É importante observar que o campo vetorial foi elaborado para um robô holonômico puntual. Por consequência, os comandos de velocidade retornados ($\mathbf{v}_{\mathbf{q}_x}$, $\mathbf{v}_{\mathbf{q}_y}$) são diferentes das entradas de controle de um carro (v_1, v_2), de acordo com o modelo apresentado na Seção 3.1, e o movimento deste não levará em consideração a presença de obstáculos. Na próxima seção será discutida uma forma de guiar um veículo autônomo inserido em um campo vetorial de velocidades.

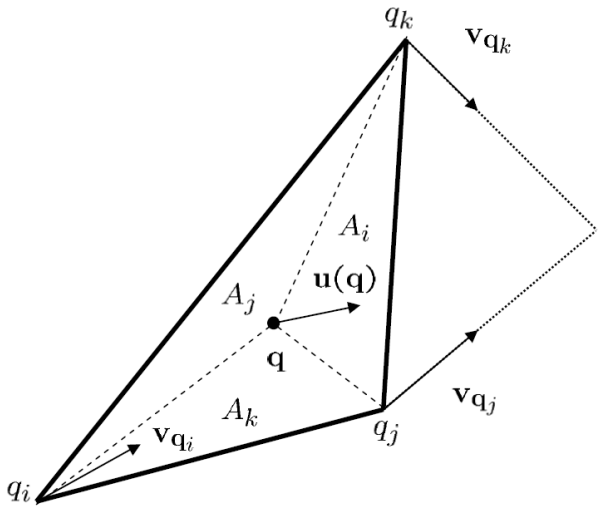


Figura 3.4: Combinação convexa dos vetores base $\mathbf{v}_{\mathbf{q}_i}$, $\mathbf{v}_{\mathbf{q}_j}$ e $\mathbf{v}_{\mathbf{q}_k}$ para gerar o campo vetorial $\mathbf{u}(\mathbf{q})$ segundo a Equação (3.4) [Pereira et al., 2009].

3.4 Linearização por realimentação de estados

Para se atuar em um sistema e poder controlá-lo é fundamental a aplicação de entradas adequadas às definidas por seu modelo. Em um carro autônomo, por exemplo, as entradas de controle, para o modelo da Equação (3.1), são as velocidades linear das rodas dianteiras v_1 e da variação angular do esterçamento destas rodas v_2. Neste caso, técnicas de planejamento de movimento ou desvio de obstáculos que forneçam esses comandos de velocidade, facilitam a atuação e o controle de movimento do veículo (ver Figura 3.1). No entanto, as técnicas que fornecem comandos de velocidade geralmente o fazem para robôs holonômicos e precisam ser adaptadas em entradas válidas para serem aplicadas em robôs não-holonômicos (como o carro). Este é o caso dos campos vetoriais de velocidade apresentados na Seção 3.3, criados para um robô de pose $\mathbf{q} = (x, y)$ e velocidades $\dot{\mathbf{q}} = (v_x, v_y)$. Nesta seção será discutido um controlador baseado na linearização por realimentação estática (FBL) que permite gerar entradas de controle que fazem o veículo seguir o campo vetorial[2].

[2] Maiores detalhes podem ser encontrados em [Luca et al., 1998].

O primeiro passo é definir um ponto z, onde serão avaliadas as velocidades do campo vetorial, tal que $\dot{z} = \dot{\mathbf{q}}$. Seja $v = (v_1, v_2)$, o objetivo da FBL é encontrar um sistema (representado pela matriz A) que permita linearizar a relação:

$$\dot{z} = A \cdot v. \tag{3.5}$$

Se A tiver inversa, então o sistema pode ser linearizado pela realimentação estática de seus estados, segundo:

$$v = A^{-1}\dot{z}. \tag{3.6}$$

Por simplicidade, é natural considerar o ponto z inicialmente sobre o referencial do veículo, como segue:

$$z = \begin{bmatrix} x \\ y \end{bmatrix}. \tag{3.7}$$

A relação entre $\dot{z}$ e v é encontrada com o cálculo da derivada temporal de z, cujo o resultado está expresso por:

$$\dot{z} = \begin{bmatrix} \cos(\theta)\cos(\phi) & 0 \\ \sin(\theta)\cos(\phi) & 0 \end{bmatrix} v = A(\theta, \phi)v. \tag{3.8}$$

Neste resultado, o sistema $A(\theta, \phi)$ é claramente não invertível e não permite a linearização da entrada com a saída para desacoplar o problema. Uma forma de se contornar este problema é redefinir as entradas (ponto z) para uma projeção frontal ao veículo, segundo a Equação (3.9). Esta projeção é representada pela letra P na Figura 3.2.

$$z = \begin{bmatrix} x + l\cos(\theta) + \triangle_P \cos(\theta + \phi) \\ y + l\sin(\theta) + \triangle_P \sin(\theta + \phi) \end{bmatrix}. \tag{3.9}$$

Com o cálculo da derivada temporal da Equação (3.9), tem-se que:

$$\begin{aligned} \dot{z} &= \begin{bmatrix} \cos(\theta)\cos(\phi) - \sin(\theta)\sin(\phi) - \dfrac{\triangle_P \sin(\theta + \phi)\sin(\phi)}{l} & -\triangle_P \sin(\theta + \phi) \\ \sin(\theta)\cos(\phi) + \cos(\theta)\sin(\phi) + \dfrac{\triangle_P \cos(\theta + \phi)\sin(\phi)}{l} & \triangle_P \cos(\theta + \phi) \end{bmatrix} v \\ &= A(\theta, \phi)v. \end{aligned} \tag{3.10}$$

Assim, para que $A(\theta, \phi)$ tenha inversa, seu determinante precisa ser diferente de zero. Neste caso, o $\det(A(\theta, \phi)) = \triangle_P \neq 0$, e o sistema pode ser linearizado pela Equação (3.6).

Para compreender este resultado, considere z como sendo um ponto de um caminho a ser seguido e $\dot{z}$ as velocidades desejadas neste caminho. Se $\triangle_P$ for grande, pequenas variações na direção do carro serão suficientes para seguir este caminho. Caso contrário, o sistema se torna mais sensível a essas variações, ou seja, mais oscilatório. Porém, utilizar $\triangle_P$ grande também implica em estar mais distante do caminho real, pois o ponto de análise estaria muito à frente do robô. No caso da linearização por realimentação estática, isto implica em fornecer entradas de controle mais suaves ou não e mais próximas das reais desejadas para o carro. Portanto, deve-se encontrar um equilíbrio na magnitude de $\triangle_P$ para que o movimento do veículo seja ao mesmo tempo suave e o mais próximo do real desejado. O equilíbrio pode ser encontrado variando-se gradativamente o valor de $\triangle_P$ quando este é iniciado com um valor próximo de zero[3].

3.5 Método da janela dinâmica

Realizar o controle de um carro autônomo apenas com as informações vindas do planejamento de movimento não garante sua segurança. Isto ocorre principalmente em decorrência das mudanças no ambiente do robô e simplificações em seu espaço de configurações. Para contornar estes problemas, é comum associar técnicas de desvio de obstáculos às de planejamento de movimento para que o robô navegue em segurança e completude, como apresentado no Capítulo 2. Nesta seção serão discutidos os princípios de uma destas técnicas, o Método da Janela Dinâmica (DWA), para compor o planejamento da Figura 3.1 em conjunto com o campo vetorial de velocidades (Seção 3.3).

O DWA foi inicialmente desenvolvido por [Fox et al., 1997] para realizar o controle de um robô não-holonômico do tipo *Synchro-Drive* em um ambiente com obstáculos, sem garantia de completude de seu caminho. Seu princípio está na criação de um mapa de velocidades aplicáveis como entradas de controle do *Synchro-Drive*[4]. O mapa é obtido por técnicas reativas locais de desvio de obstáculos, que levam em consideração as

[3] Para o veículo autônomo em questão o valor encontrado foi igual 0,5 metros.

[4] Os conceitos preliminares do DWA serão apresentados para o robô *Synchro-Drive*. No entanto, na literatura existem várias adaptações deste método para diferentes robôs e podem ser verificadas em [Brock

restrições cinemáticas e dinâmicas deste robô. No caso do *Synchro-Drive*, a entrada de controle é o par (v, ω), que representam suas velocidades linear e angular respectivamente. As restrições cinemáticas são relacionadas à estrutura do robô e limitam o mapa de velocidades em:

$$V_s = \{(v, \omega) | v \in [v_{min}, v_{max}], \omega \in [\omega_{min}, \omega_{max}]\}. \tag{3.11}$$

Para exemplificar o conjunto V_s, será considerado o robô *Synchro-Drive* no instante apresentado na Figura 3.5. Nesse caso, as velocidades que compõem V_s são todos os pares internos ao limite indicado na Figura 3.6. As restrições dinâmicas (máximas acelerações possíveis) limitam localmente as velocidades atuais (v_a, ω_a) do robô, de acordo com a Equação (3.12). Desta forma, são fornecidas as velocidades máximas e mínimas que o robô pode atingir em um intervalo de tempo $\triangle t$ e o mapa de velocidades é reduzido para uma janela dinâmica V_d, que se altera a cada $\triangle t$. Na Figura 3.6 as velocidades que compõem V_d estão limitadas em vermelho.

$$V_d = \{(v, \omega) | v \in [v_a - \dot{v} \cdot \triangle t, v_a + \dot{v} \cdot \triangle t], \omega \in [\omega_a - \dot{\omega} \cdot \triangle t, \omega_a + \dot{\omega} \cdot \triangle t]\}. \tag{3.12}$$

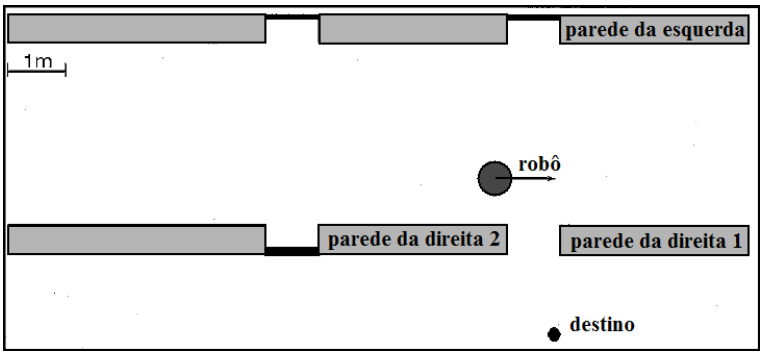


Figura 3.5: Instante de movimento de um robô do tipo *Synchro-Drive*.

Nesta janela V_d são apenas consideradas velocidades que sejam seguras em relação aos obstáculos, tais que permitam ao *Synchro-Drive* parar sem se colidir. Isto é feito calculando-se as distâncias para os obstáculos vizinhos do robô quando este trafega com cada um dos pares de velocidades de V_d. O cálculo foi implementado na função $dist(v, \omega)$, a qual estima a trajetória do robô com suas equações de movimento (ver [Fox

and Khatib, 1999], [Arras et al., 2002], [Rebai et al., 2007], [Seder and Petrovic, 2007] e [Rebai and Azouaoui, 2009].

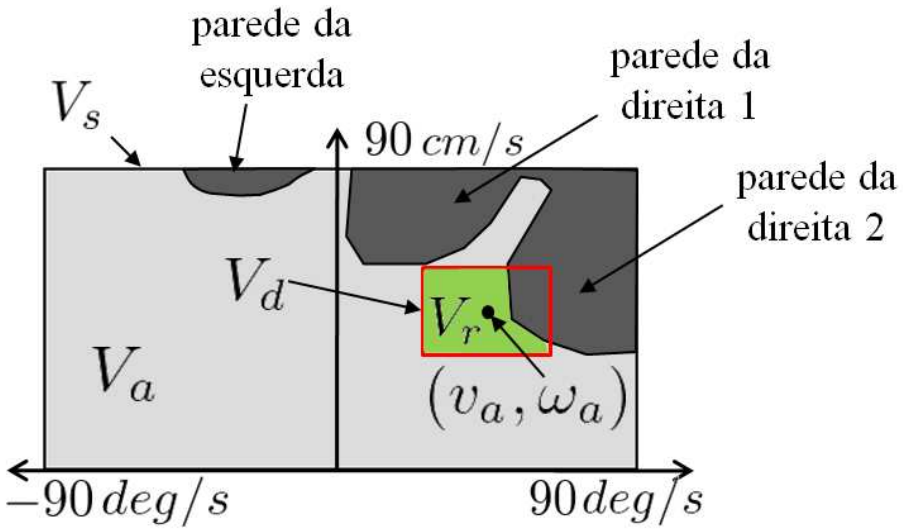


Figura 3.6: Exemplo dos conjuntos V_s, V_a, V_d e V_r, necessários para a aplicação do Método da Janela Dinâmica em um robô do tipo *Synchro-Drive*. Estes conjuntos foram obtidos por meio da leitura sensorial no ambiente da Figura 3.5, com os valores atuais das velocidades representados por (v_a, ω_a). Em cinza escuro estão todos os pares de (v, ω) que implicam em um movimento inválido no robô, com possíveis colisões. A região destacada em verde são as velocidades válidas para a janela dinâmica.

et al., 1997]) e verifica a presença de obstáculos nesta trajetória. A menor distância entre o robô e os obstáculos é então retornada. Nos princípios de cinemática, é possível encontrar uma relação entre velocidade final v_f e deslocamento $\triangle d$, com a chamada Equação de Torricelli:

$$v_f^2 = v_0^2 + 2a\triangle d, \tag{3.13}$$

na qual v_0 é a velocidade inicial e a é a aceleração do veículo. Esta relação, se aplicada com a distância fornecida pela função $dist(v, \omega)$ e as velocidades linear (movimento em linha reta) e angulares do robô (movimento circular), permite encontrar qual a velocidade limite que o robô pode ter para conseguir parar em segurança. Neste caso, v_f é igual a zero e v_0 é igual a velocidade atual do robô, seja ela linear ou angular. Assim, as velocidades de V_d serão válidas se:

$$V_a = \left\{(v, \omega) | v \leq \sqrt{2 \cdot dist(v, \omega) \cdot \dot{v}_b}, \omega \leq \sqrt{2 \cdot dist(v, \omega) \cdot \dot{\omega}_b}\right\}, \tag{3.14}$$

onde $\dot{v}_b$ e $\dot{\omega}_b$ são as acelerações de frenagem do robô. Estas velocidades estão representadas pela região em cinza claro e verde da Figura 3.6. O espaço resultante de busca V_r é então definido como a interseção dos demais espaços apresentados, como segue:

$$V_r = V_s \bigcap V_a \bigcap V_d. \tag{3.15}$$

Ainda na Figura 3.6, estas velocidades foram marcadas com a cor verde. O par de velocidades final é escolhido segundo uma função objetivo $G(v, \omega)$. Esta função representa a soma ponderada de outras subfunções que medem o direcionamento (*heading*) do robô, sua velocidade (*velocity*) e a clareza (*dist*) do caminho em relação ao obstáculos, segundo:

$$G(v, \omega) = \alpha \cdot heading(v, \omega) + \beta \cdot dist(v, \omega) + \gamma \cdot velocity(v, \omega), \tag{3.16}$$

onde α, β e γ são os pesos destas subfunções. Na função objetivo $G(v, \omega)$, a subfunção *heading* favorece velocidades que impliquem em uma orientação do robô mais próximas do seu destino. O segundo termo *dist* (mesmo utilizado para calcular as velocidades válidas V_a) é uma tentativa de maximizar o espaço entre o robô e seus obstáculos. Já a subfunção *velocity* prioriza as maiores velocidades de translação para realizar o movimento do robô quando estiver longe do destino e menores, caso contrário. O objetivo do DWA, portanto, é otimizar $G(v, \omega)$ a cada $\triangle t$ de maneira que o robô trafegue em segurança até um ponto de destino e respeite as subfunções apresentadas.

Como o DWA é construído em torno de subfunções das velocidades de controle do robô, novas funções podem ser criadas para que o mesmo se adeque a tarefas mais específicas. Em [Brock and Khatib, 1999] por exemplo, duas funções foram adicionadas para permitir que o DWA também seguisse um planejamento de movimento global previamente realizado. No Capítulo 4, serão apresentadas as adaptações feitas ao DWA apresentado nesta seção, para que este funcione segundo o modelo de um carro autônomo que é a proposta deste trabalho.

3.6 Controle de velocidade

O controle de movimento de veículos autônomos apresentado neste capítulo priorizou técnicas que utilizassem comandos de velocidade e que pudessem ser adaptadas ao modelo da Equação (3.1). No entanto, as atuações reais de um veículo não são velocidades e sim a aceleração, o freio e a posição angular do volante. Assim, para impor a velocidade linear desejada ao veículo, deve-se atuar em sua aceleração e em seu sistema de frenagem. Igualmente, a velocidade angular de esterçamento deverá ser fornecida pela variação da posição angular do volante. Controlar estas entradas, portanto, é permitir imposição destas velocidades de controle no carro. Nesta seção o controle destas velocidades será abordado, como composição da última etapa da Figura 3.1.

Para realizar o controle da velocidade linear v_1 de veículos autônomos, também chamado de controle longitudinal, existem diversas técnicas disponíveis na literatura aplicadas em veículos reais, tais como [Thrun et al., 2006], [Braid et al., 2006], [Urmson et al., 2008]. Dentre elas está a solução proposta por [Freitas and Pereira, 2010], adotada neste trabalho. Seu funcionamento baseia-se no uso da lógica *Fuzzy* para um controle em mais alto nível. Segundo [Freitas and Pereira, 2010], existem alguns benefícios em se realizar o controle da velocidade linear por lógica *Fuzzy*, pois esta fornece:

1. Uma maneira formal de se representar, manipular e implementar as ações que um ser humano utilizaria para controlar o sistema;

2. Um tratamento para as não linearidades do sistema, presentes nas dinâmicas do veículo e nas de seus atuadores, incorporando-as na solução.

O projeto de um controlador *Fuzzy* normalmente segue algumas etapas importantes, tais como ([Freitas and Pereira, 2010]): seleção das entradas e das saídas de controle, definição das funções de pertinência das entradas e das saídas, especificação das regras de controle, seleção do método de inferência associado às regras de controle, seleção dos métodos de fuzzyficação e de defuzzificação e, por fim, a avaliação do controlador. A entrada do sistema *Fuzzy* é o erro normalizado da velocidade, obtido pela diferença entre a velocidade desejada e a velocidade atual do veículo. Já as saídas são os comandos para a aceleração e para o freio. Com as entradas e saídas, definem-se as funções de pertinência para indicar com que grau um elemento pertence a um conjunto nebuloso.

A definição foi nominal aos possíveis estados da entrada e das saídas, de acordo com a Figura 3.7.

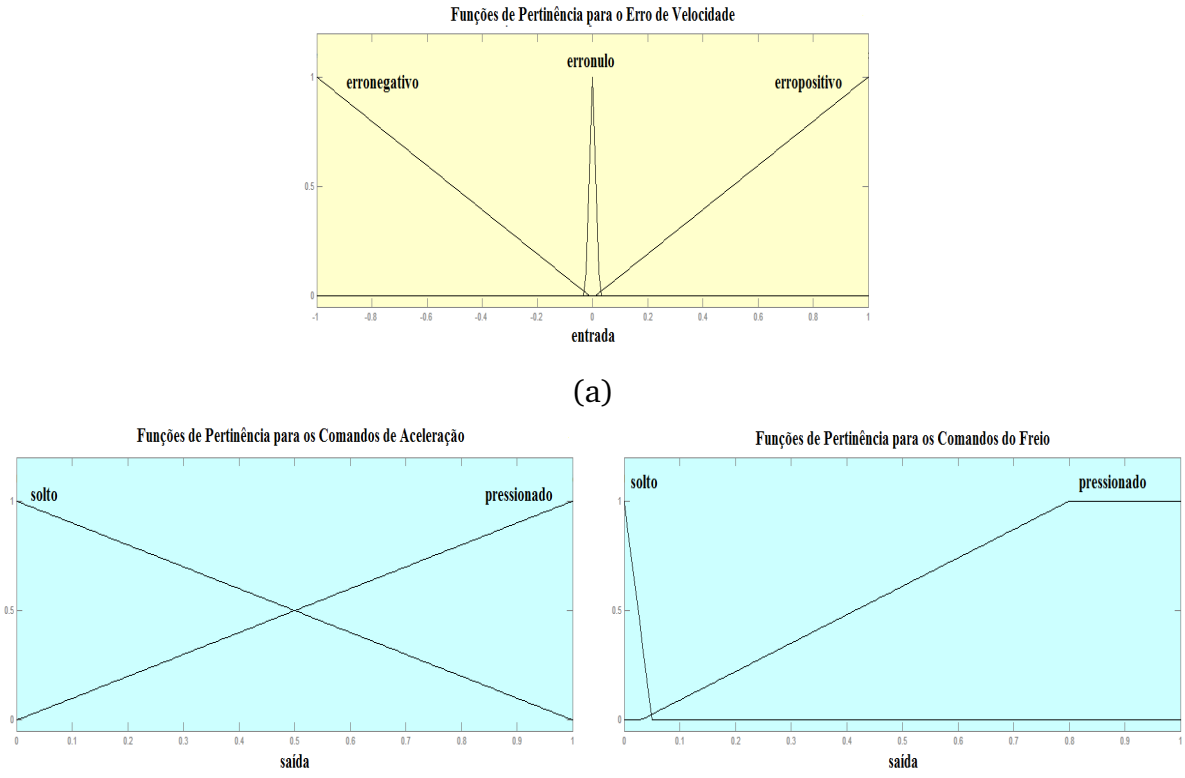


Figura 3.7: Funções de Pertinência para o controle de velocidade por lógica *Fuzzy*, definidas para a entrada de erro de velocidade (a) e para as saídas de aceleração (b) e freio (c).

As regras de controle expressam as ações que serão tomadas para cada entrada do controlador. Suas definições basearam-se nas mesmas ações que um motorista utilizaria para realizar o controle. Se a velocidade do veículo estiver acima da desejada, por exemplo, o erro será positivo e as saídas serão o acelerador “solto” e o freio “pressionado”. Este princípio foi adotado para cada uma das demais possíveis entradas. Na prática, estas regras são inferidas por algum modelo nebuloso, que permita converter para valores numéricos os estados da entrada e das saídas. Em [Freitas and Pereira, 2010] esta operação foi realizada pelo modelo de Mandani e operações de interseção e união para a fuzzificação e defuzzificação.

No próximo capítulo será apresentada a estrutura metodológica utilizada neste trabalho. Os conceitos preliminares para o controle de movimento aqui apresentados serão

integrados com técnicas de percepção do ambiente para realizarem a navegação autônoma de um carro em ambientes não estruturados.

Capítulo 4

Metodologia

Como discutido no Capítulo 2, a navegação segura de veículos autônomos pode ser realizada de diversas formas. Neste trabalho, este problema foi segmentado em percepção do ambiente e controle de navegação, de acordo com o diagrama da Figura 4.1. A percepção do ambiente é a etapa responsável por garantir, ao controle de navegação, o conhecimento constante do ambiente explorado, etapa realizada com a detecção de obstáculos e a grade de ocupação. No controle de navegação estão os passos necessários para gerar os comandos de atuação do carro autônomo, que respeitem os obstáculos detectados pela percepção do ambiente.

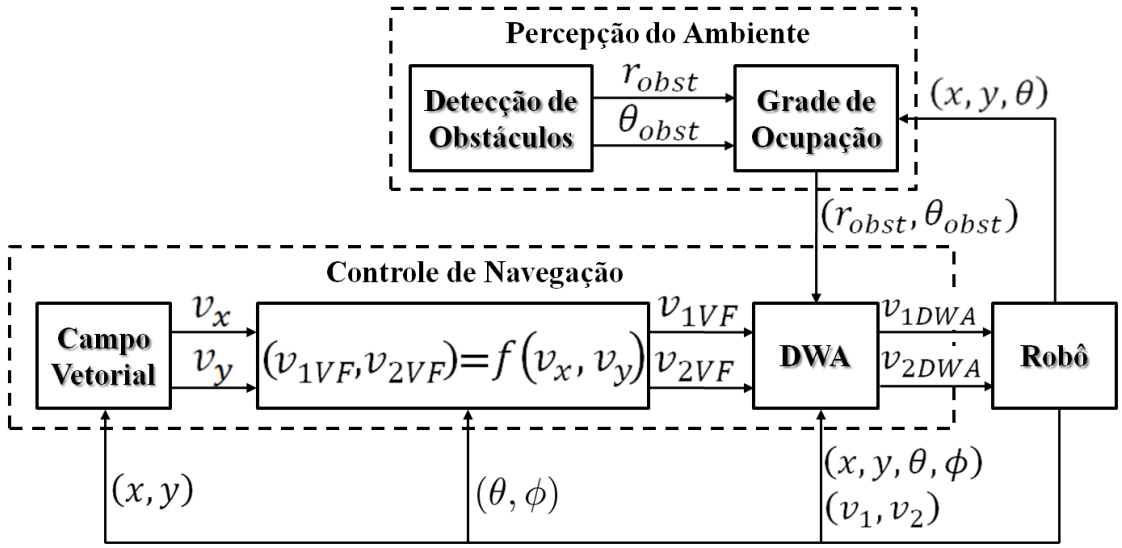


Figura 4.1: Diagrama da solução adotada para realizar a navegação segura de um veículo autônomo.

Por ser uma solução complexa, ela depende de uma série de recursos e formulações relativos ao robô, como descritos no Capítulo 3. O presente capítulo utilizará estes

conceitos para esclarecer a estrutura da solução apresentada na Figura 4.1. Algumas das técnicas apresentadas no capítulo anterior sofreram alterações para se adequar à esta estrutura, e também serão abordadas a seguir.

4.1 A percepção do ambiente

Assim como os olhos são fundamentais para um motorista conhecer o ambiente em sua volta e dirigir um automóvel em segurança, a percepção do ambiente por meio de sensores é uma etapa crucial para a navegação autônoma. De acordo com os sensores utilizados, existem diversas técnicas para esse fim, como apresentadas na Seção 2.2. Para este trabalho assume-se que os recursos disponíveis são uma câmera de visão estéreo e um sensor a laser, os quais foram o foco para a percepção do ambiente. Esta seção apresentará uma solução para a detecção de obstáculos por visão estéreo e uma técnica para fundir suas informações com os dados do laser e manter um mapa local ao redor do veículo.

4.1.1 Detecção de obstáculos utilizando visão estéreo

A detecção de obstáculos por visão estéreo foi segmentada nas etapas apresentadas na Figura 4.2. Os conceitos aqui utilizados são aplicáveis a qualquer sistema de visão estéreo e podem ser melhor compreendidos em [Faugeras, 1993]. Com este sistema torna-se possível obter informações tridimensionais a partir de duas ou mais imagens de diferentes pontos de vista.

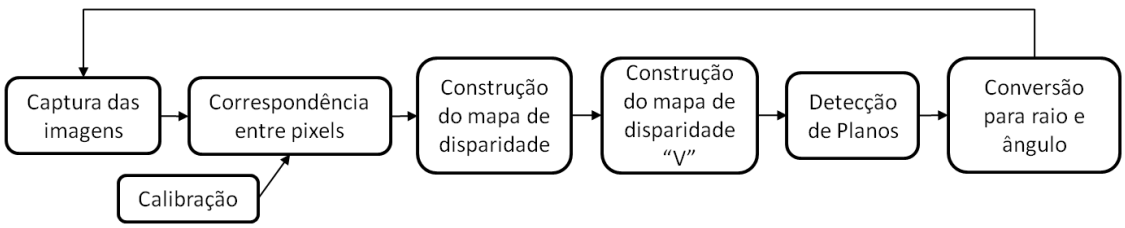


Figura 4.2: Diagrama da solução adotada para se detectar obstáculos com a visão estéreo.

Captura das imagens

Como início do processo de detecção, tem-se a captura das imagens do ambiente. Isso deve acontecer de forma síncrona pelas duas ou mais câmeras que compõem o sistema estéreo. O fato destas imagens serem síncronas garante que a informação contida nas duas imagens sejam temporalmente equivalentes.

Calibração

A calibração é uma etapa de extrema importância para se trabalhar com sistemas de visão estéreo. Nela são obtidos os parâmetros intrínsecos e extrínsecos de cada uma das câmeras do conjunto. Estes parâmetros fornecem dados de distância focal, centro da imagem e a pose relativa entre as câmeras do conjunto. Quanto melhor esta calibração, melhor os resultados dos próximos passos da Figura 4.2. Considerando um sistema de câmeras rígido, a calibração é um processo que pode ser realizado uma única vez.

Correspondência entre pixels

Com as câmeras calibradas, deve ser realizado um processo de busca por correspondências entre pixels nas imagens. Para duas imagens Π e Π', o problema consiste em encontrar o pixel p' em Π' e o pixel correspondente p em Π que representem a projeção do mesmo ponto no espaço P. Esta etapa pode ter um custo computacional elevado dependendo da técnica utilizada e do tamanho do espaço de busca.

Para diminuir o tempo de processamento, é geralmente realizada a retificação das imagens, que consiste em se aplicar uma série de rotações em cada uma das imagens de forma que pixels correspondentes nas duas imagens estejam situados na mesma linha. Assim, restringe-se a busca por correspondência a uma linha das imagens (ou a algumas linhas devido a erros de calibração). Encontrar os pixels correspondentes p e p' permite construir o mapa de disparidade, próximo passo para a detecção de obstáculos.

Construção do mapa de disparidade

Disparidade é um valor de distância (medida em número de pixels) entre dois pixels correspondentes p e p' quando as imagens Π e Π' retificadas são sobrepostas. Na prá-

tica, como p e p' estão na mesma linha das duas imagens, a disparidade corresponde à diferença dos índices que indicam a coluna em que estão cada pixel. O mapa de disparidade, $\triangle$, é o conjunto de disparidades para cada par de pixels das imagens, sendo, portanto, da mesma dimensão da imagem retificada. Este mapa é apresentado como uma imagem em tons de cinza. É fácil mostrar que o valor da disparidade varia com o inverso da distância do ponto no espaço ([Faugeras, 1993]) e, portanto, esta informação pode ser utilizada para detectar a distância entre um ponto no espaço e as câmeras. A Figura 4.3 apresenta um exemplo de mapa de disparidade, onde os pontos claros estão mais próximos da câmera e os escuros mais distantes. Como o processo de correlação não é perfeito e algumas regiões da cena são percebidas por apenas uma das câmeras, existem pixels em $\triangle$ que não possuem disparidade. Estes são então marcados como preto (zero) na imagem.

Figura 4.3: Par estéreo para criar o mapa de disparidade (acima) e mapa de disparidade resultante (abaixo).

Se p e p' são projeções do ponto espacial $P = (X, Y, Z)$ em Π e Π', respectivamente, as coordenadas de P podem ser encontradas a partir da disparidade d entre eles de acordo com a Equação (4.1).

$$\begin{aligned} X &= \frac{p_x Z}{f}, \\ Y &= \frac{p_y Z}{f}, \\ Z &= \frac{fB}{d}. \end{aligned} \tag{4.1}$$

Onde f é a distância focal da câmera, p_x e p_y são as coordenadas do ponto p na imagem Π, B é a distância entre as câmeras.

Construção do mapa de disparidade "V"

A criação do mapa de disparidade "V" ($I_{v\triangle}$) é um processo simples se comparado ao necessário para se obter o mapa de disparidade ($\triangle$). Basicamente, $I_{v\triangle}$ é uma transformação do $\triangle$ que conta o número de pixels presentes em cada linha[1] que possuem o mesmo valor de disparidade. Ao final é montada uma figura onde as linhas são as mesmas da imagem real e as colunas representam a intensidade do mapa de disparidade original (normalmente de 0 a 255). Nesta figura, o valor de cada pixel representa quantos pixels possuem o mesmo valor de disparidade na linha correspondente do mapa original, sendo assim chamado mapa de disparidade "V". A Figura 4.4 apresenta o resultado desta transformação de $\triangle$ em $I_{v\triangle}$, a partir das imagens mostradas na Figura 4.3.

Figura 4.4: Exemplo de mapa de disparidade "V".

[1] Em uma imagem as coordenadas das linhas são geralmente chamadas de v e das colunas de u.

Detecção de planos

De maneira simplificada, o mundo visualizado por um carro pode ser classificado em regiões trafegáveis e obstáculos. Como forma de facilitar a percepção de ambos, eles foram aproximados a planos horizontais e verticais. Em uma situação ideal, isto significaria uma rua plana e obstáculos iguais a objetos achatados e sem forma. De acordo com [Labayrade et al., 2002], [Soquet et al., 2007] e [Caraffi et al., 2007], ao analisar essas situações em um mapa de disparidade ($\triangle$), as regiões trafegáveis apresentariam variações graduais na disparidade até o horizonte, com o mesmo valor em uma linha. Já os obstáculos possuiriam a mesma disparidade em uma mesma coluna. Esta mesma análise em um mapa de disparidade "V" ($I_{v\triangle}$) implicaria em regiões trafegáveis iguais a uma reta com inclinação superior a 90^{o}, devido a variação gradual da disparidade. Enquanto os obstáculos seriam iguais a retas com inclinação de 90^{o}, ou seja, elementos com a mesma disparidade. Aplicando este conceito, a detecção de planos resumiu-se a encontrar retas em $I_{v\triangle}$.

Por se tratar de uma imagem, para encontrar retas no $I_{v\triangle}$, técnicas de análise de imagem foram utilizadas. A imagem do $I_{v\triangle}$ foi binarizada e retas foram encontradas com o uso de um método baseado na Transformada de Hough [Duda and Hart, 1972]. As retas encontradas foram selecionadas quanto aos parâmetros de tamanho, número de pixels, largura e inclinação. Para facilitar o processamento, os planos trafegáveis foram processados primeiro, por serem normalmente maioria em uma imagem. O resultado foi então removido do $I_{v\triangle}$, para realizar o processamento dos planos não trafegáveis. A Figura 4.5 apresenta um exemplo de detecção de planos no $I_{v\triangle}$.

O passo final desta análise é o mapeamento das retas encontradas nas regiões de interesse da imagem original. Este mapeamento foi realizado verificando no $I_{v\triangle}$ os pontos marcados como planos trafegáveis (ou não trafegáveis) e seu valor de disparidade correspondente no $\triangle$. O processamento adotado gerou imagens binárias com a mesma dimensão da imagem original, onde pixels com o valor 0 (preto) representam a ausência do plano trafegável (ou não trafegável) e 255 (branco) representam a presença do plano em questão. No caso dos obstáculos, esta marcação é para todos os obstáculos. Porém, como é necessário considerar a distância do obstáculo à câmera neste trabalho,

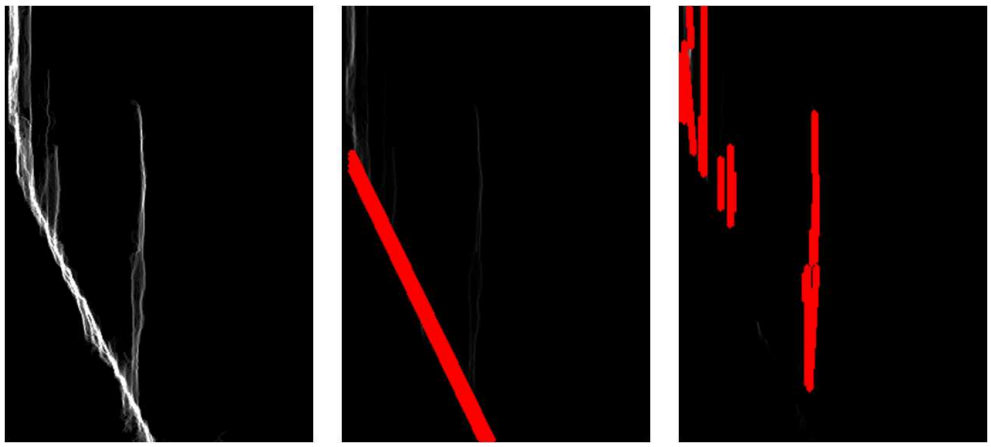


Figura 4.5: Mapa de disparidade "V" (esquerda), retas que definem os planos trafegáveis (centro) e retas que definem obstáculos (direita).

ao se verificar no $\triangle$ os pixels que foram marcados no $I_{v\triangle}$, basta utilizar a Equação (4.1) para se obter a informação desejada.

A partir das retas encontradas em $I_{v\triangle}$ e das distâncias obtidas para os obstáculos, quatro imagens binárias foram criadas com as seguintes regiões em destaque:

1. Planos trafegáveis (retas no $I_{v\triangle}$ com inclinação maior que 90^o);
2. Regiões inválidas (disparidade igual a 0 ou 255);
3. Obstáculos distantes (pixels correspondentes a distâncias maiores que 15 metros);
4. Obstáculos próximos (pixels correspondentes a distâncias menores que 15 metros)[2].

Como forma de visualização, essas imagens foram aplicadas sobre a imagem retificada da esquerda com tonalidades de cores distintas, de acordo com a Figura 4.6. Nesta imagem, com as cores reais estão os planos trafegáveis, em amarelo estão os obstáculos distantes e regiões inválidas e em vermelho estão os obstáculos próximos. Uma utilização prática destas imagens pode ser encontrada em [Lima and Pereira, 2010].

Conversão para coordenadas polares

A etapa final de detecção de obstáculos por visão estéreo é a conversão do resultado encontrado em informação de distância para o obstáculo (raio) e ângulo em relação

[2] Este limite foi determinado pela câmera utilizada, onde valores maiores de distância possuem um erro muito grande e por segurança foram descartados.

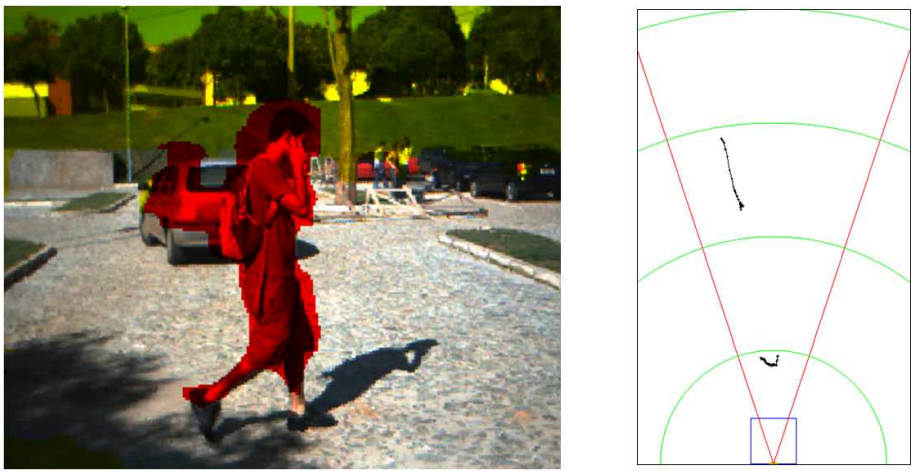


Figura 4.6: Exibição dos planos detectados a partir do mapa de disparidade “V” (à esquerda) e sua conversão em coordenadas polares (à direita). Nos planos detectados, nas cores reais da imagem estão os planos trafegáveis, em vermelho os obstáculos próximos e em amarelo os obstáculos distantes e as regiões sem disparidade. Na conversão, a câmera esta representada em amarelo, a frente do carro em azul e o FOV da câmera em vermelho. Os semicírculos em verde estão espaçados por 5 metros.

ao centro da câmera, ou seja, coordenadas polares. Esse é o formato dos dados de entrada para a grade de ocupação criada. Para concebê-los, foi considerada a imagem de obstáculos próximos obtida na detecção de planos.

Inicialmente, as colunas que limitam a imagem foram associadas aos limites do campo de visão (FOV) do sistema estéreo utilizado, com a distância angular entre elas correspondendo ao ângulo do FOV. Os ângulos das demais colunas foram então obtidos por relações diretas com os ângulos das extremidades e armazenados em um vetor de ângulos. Para cada ângulo (coluna da imagem), procurou-se pela linha de maior disparidade (menor distância) para as câmeras. Esta distância foi então armazenada em um vetor de raios para ser fornecida à grade de ocupação. Na Figura 4.6 esta conversão foi representada em preto, limitada pelo FOV em vermelho. Para facilitar a compreensão, semicírculos espaçados de 5 metros foram desenhados, juntamente com a posição da câmera em amarelo e a dimensão da frente do carro em azul.

4.1.2 Detecção de obstáculos por sensor a laser

A detecção de obstáculos utilizando sensores a laser depende de como este foi instalado no veículo, o que implicará em uma solução direta ou não. A maioria dos lasers comerciais, como discutido na Seção 2.2, realizam suas leituras em apenas um plano

no espaço tridimensional e retornam, em coordenadas polares, a distância lida e o ângulo ao qual ela corresponde. Desta forma, se o laser estiver orientado paralelamente à superfície, os obstáculos estarão na distância real retornada pelo laser. Caso contrário, algoritmos específicos deverão ser utilizados para extrair, por exemplo, as descontinuidades na leitura para assim definir o que é obstáculo e qual é a distância real para o veículo [Newman et al., 2009, Perrollaz et al., 2006, Wijesoma et al., 2001].

Como este trabalho não foca a detecção de obstáculos propriamente dita, mas sim uma solução que permita aplicar a navegação autônoma, o laser foi considerado como se estivesse em paralelo ao solo. Isso implica que seus dados sejam diretamente fornecidos à grade de ocupação, tratada com maiores detalhes a seguir.

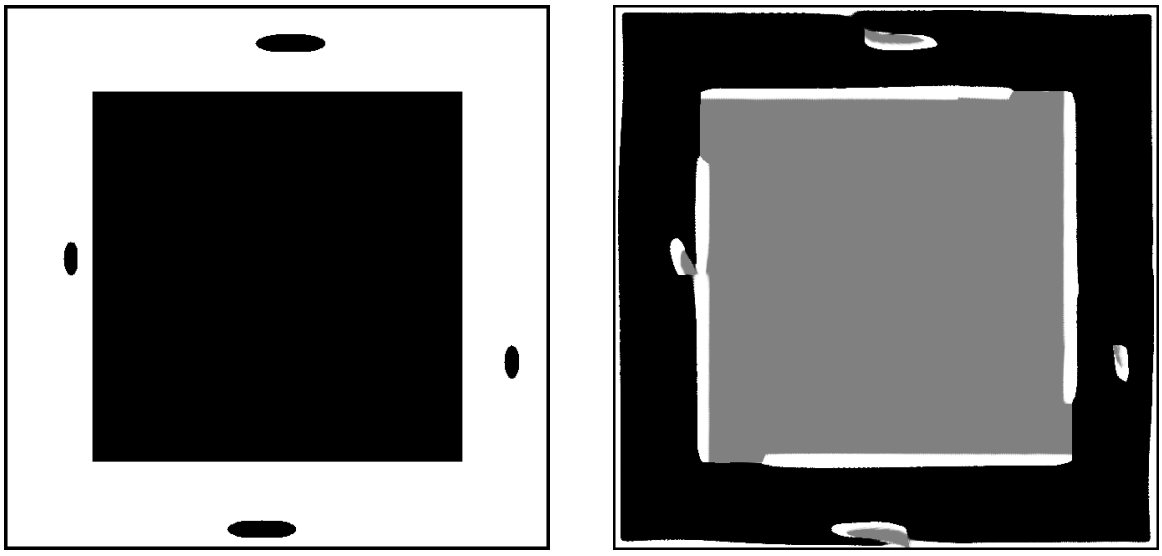
4.1.3 A grade de ocupação local

A grade de ocupação [Thrun et al., 2005] é uma técnica probabilística para descrever o ambiente de um robô a partir de seus dados sensoriais e sua pose. Nela, o ambiente é discretizado em uma matriz multidimensional (geralmente 2D ou 3D) com espaçamentos iguais, onde cada célula armazena uma variável aleatória com valor probabilístico entre 0 e 1, onde 0 representa a total ausência de obstáculos e 1 a total presença. Como, a princípio, não se tem informação sobre o ambiente explorado, estas células são normalmente iniciadas com valor $0,5$ de probabilidade. Por questões computacionais, é comum a planificação dos espaços tridimensionais em bidimensionais, o que resulta em um problema de mapeamento onde deseja-se encontrar a probabilidade *a posteriori* de uma célula do mapa a partir de uma série de dados, de acordo com a Equação (4.2)

$$p(m_i|z_{1:t}, x_{1:t}), \tag{4.2}$$

onde m_i é a célula da grade com índice i, $z_{1:t}$ é o conjunto de medidas sensoriais até o tempo t e $x_{1:t}$ é o caminho realizado pelo robô definido por suas poses. A Figura 4.7 apresenta um exemplo de grade de ocupação gerada em um ambiente de simulação, onde cada pixel da imagem representa uma célula da grade.

Por utilizar uma discretização do ambiente, a grade de ocupação é pouco eficiente quando necessita armazenar ambientes maiores, tais como os externos. Porém,



(a) (b)

Figura 4.7: Exemplo de uma grade de ocupação (b) gerada a partir do ambiente de simulação (a). No ambiente, obstáculos estão em preto e regiões livres em branco. Na grade de ocupação, tonalidades próximas ao branco equivalem a obstáculos. Em preto estão as regiões livres de obstáculos e em cinza são as regiões inexploradas. A grade de ocupação foi criada pela simulação de um veículo trafegando neste ambiente com um sensor a laser, com FOV de 180°, acoplado na sua dianteira.

tratando-se de pequenas porções do ambiente, seu uso se torna viável e agrega alguns benefícios inerentes à sua forma de obtenção. Neste trabalho, a grade de ocupação foi utilizada para representar ambientes estáticos em uma pequena janela (local) de esquecimento centralizada no referencial do veículo, para ampliar a percepção dos obstáculos ao seu redor. Esta janela foi chamada de grade de ocupação local e sua construção será discutida nesta seção.

O primeiro passo é contornar os problemas de instabilidade numérica gerados para valores de probabilidade próximos a 0 e 1. Uma solução comum é utilizar o logaritmo da probabilidade (ou *log-odds*) para representar os valores da grade de ocupação, definido por (4.3).

$$l_{t,i} = \log \frac{p(m_i|z_{1:t}, x_{1:t})}{1 - p(m_i|z_{1:t}, x_{1:t})}. \tag{4.3}$$

Neste caso, a probabilidade pode ser facilmente recuperada pela Equação (4.4).

$$p(m_i|z_{1:t}, x_{1:t}) = 1 - \frac{1}{1 + \exp l_{t,i}}. \tag{4.4}$$

O próximo passo é estimar o valor de cada célula em um dado tempo t, etapa realizada com o Filtro de Bayes, que se baseia no Teorema de Bayes, como descrito em (4.5):

$$p(m_i|z_{1:t}, x_{1:t}) = \frac{p(z_t|z_{1:t-1}, m_i, x_t)p(m_i|z_{1:t-1}, x_{1:t-1})}{p(z_t|z_{1:t-1})}. \tag{4.5}$$

onde o termo $p(z_t|z_{1:t-1}, m_i, x_t)$ descreve o modelo probabilístico do sensor de profundidade, $p(m_i|z_{1:t-1}, x_{1:t-1})$ é a probabilidade da célula m_i no instante $t-1$ e $p(z_t|z_{1:t-1})$ é a leitura atual do sensor dadas as leituras passadas. Como neste trabalho os ambientes são considerados estáticos, as leituras sensoriais não dependem da medição anterior. Desta forma, é possível simplificar a Equação (4.5) e obter:

$$p(m_i|z_{1:t}, x_{1:t}) = \frac{p(z_t|m_i, x_t)p(m_i|z_{1:t-1}, x_{1:t-1})}{p(z_t)}. \tag{4.6}$$

Em [Thrun et al., 2005] pode ser encontrada a transformação do filtro de Bayes em *log-odds*, cuja a adaptação gerou o Algoritmo 4.1 de mapeamento sensorial na grade de ocupação. De acordo com a linha 3 deste algoritmo, cada célula da grade, dentro do campo de visão do sensor, é atualizada por meio da leitura atual do sensor de profundidade (z_t, modelada pela função *inverse_range_sensor_model*) e dos dados passados da célula ($l_{t-1,i}$). Esta dependência resulta em uma filtragem de possíveis ruídos nas leituras sensoriais. As células que se encontram fora do campo de visão mantém o seu valor anterior (linha 5 do algoritmo). A constante l_0 representa a ocupação *a priori* em *log-odds* (4.7).

Algoritmo 4.1 Mapeamento sensorial na grade de ocupação local baseado no filtro de Bayes.

occupancy_grid_mapping($l_{t-1,i}, z_t$**)**

1: **para** todas as células m_i **faça**
2: **se** m_i está no campo de percepção de z_t **então**
3: $l_{t,i} = l_{t-1,i} +$ **inverse_range_sensor_model**$(m_i, x_{sens}, z_t) - l_0$
4: **senão**
5: $l_{t,i} = l_{t-1,i}$
6: **fim se**
7: **fim para**
8: **retorne** $l_{t,i}$

$$l_0 = \log \frac{p(m_i) = 1}{p(m_i) = 0} = \log \frac{p(m_i)}{1 - p(m_i)}. \tag{4.7}$$

Como descrito na Equação (4.6) e no Algoritmo 4.1, cada leitura do sensor pode ser descrita por um modelo probabilístico, o qual pode incorporar diversas incertezas. Existem formas mais básicas de se descrever um modelo, como a apresentada em [Thrun et al., 2005], a qual considera um sensor ideal para representar as leituras na grade, sem incertezas. Um modelo mais abrangente foi apresentado em [Elfes, 1989], para considerar incertezas tanto na distância lida para um objeto quanto no ângulo de cobertura de sonares. O modelo foi descrito por uma distribuição Gaussiana bidimensional reproduzida como:

$$p(z_t|m_i, x_t) = p(r_i|z_t, \phi_i) = \frac{1}{2\pi\sigma_r\sigma_\phi} e^{-\frac{1}{2}\left(\frac{(r_i - z_t)^2}{\sigma_r^2} + \frac{\phi_i^2}{\sigma_\phi^2}\right)}, \tag{4.8}$$

onde o modelo probabilístico $p(z_t|m_i, x_t)$ da leitura atual do sensor é associado à distância r_i do sensor à célula m_i e ao ângulo ϕ_i entre o centro da célula e o eixo principal do sensor. Os demais parâmetros podem ser melhor compreendidos na Figura 4.8. Para a leitura sensorial de um sonar, esta Gaussiana possui a forma apresentada na Figura 4.9. Em analogia ao modelo apresentado por [Elfes, 1989], cada leitura de distância (raio) fornecida pelos procedimentos de detecção de obstáculos discutidos neste capítulo foram equiparadas a sonares. Estes sonares estariam centralizados na mesma origem do sensor original e rotacionados de acordo com o ângulo de leitura da distância. Assim, o modelo probabilístico para cada leitura dos sensores deste trabalho pode ser considerado igual ao da Equação (4.8). No entanto, esta é apenas uma aproximação a qual não utilizou nenhum teste prático com os sensores utilizados para determinar se este é o melhor modelo a se usar. As variâncias σ_r^2 e σ_ϕ^2 computam as incertezas na distância medida r_i e no ângulo ϕ_i respectivamente, estimadas segundo o *datasheet* destes sensores. Com essas premissas, a função *inverse_range_sensor_model* pode ser construída de acordo com o Algoritmo 4.2.

É importante observar que a função *inverse_range_sensor_model* somente pode ser realizada caso a célula m_i esteja dentro do FOV do sensor em questão, de acordo com o Algoritmo 4.1. Com esta condição atendida, deve-se encontrar o menor ângulo entre a leitura do sensor e o centro de massa da célula m_i ($\phi_{k,sens}$), o que equivale às linhas de 2 a 5 no Algoritmo 4.2. A Equação (4.8) é então aplicada na linha 6 para este ângulo

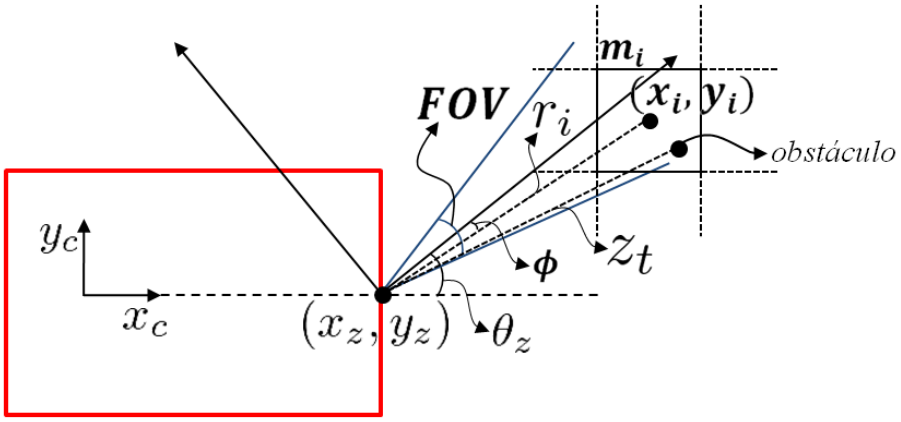


Figura 4.8: Mapeamento de uma leitura sensorial baseada em distribuição Gaussiana bidimensional.

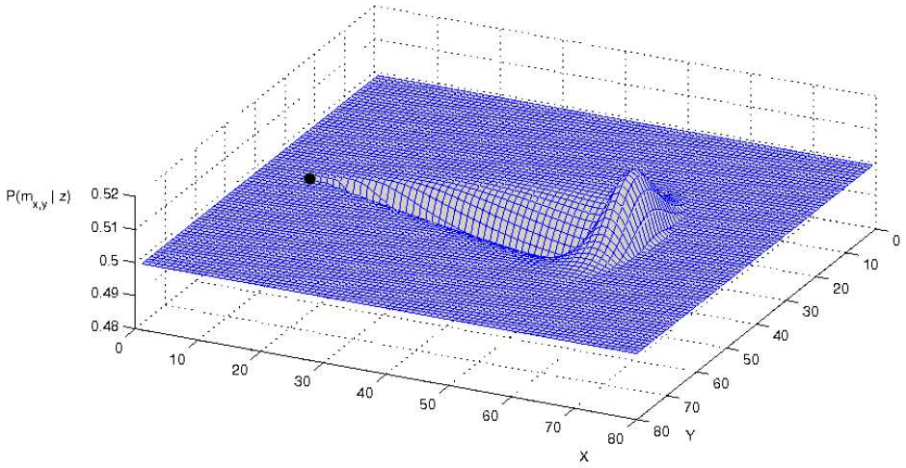


Figura 4.9: Exemplo de uma leitura sensorial baseada em uma distribuição Gaussiana bidimensional [Souza, 2008].

e sua leitura de distância (z_t^k) correspondente. O retorno da função será a constante l_0 (linha 8), caso a célula m_i esteja mais distante que o obstáculo encontrado, ou a probabilidade encontrada em *log-odds* (linha 10).

A última etapa para a construção da grade de ocupação local é a atualização de seus dados com o deslocamento do veículo. A cada movimento do carro no mundo real, os dados da grade anterior devem ser mapeados na nova grade. Este mapeamento pode ser facilmente calculado por transformações entre os referenciais do veículo e o mundo, como descreve a Figura 4.10. Ao final é calculada a transformação direta do referencial anterior (1) para o atual (2), aplicável a cada um dos pontos da grade anterior. Os pontos que forem mapeados fora dos limites da nova grade de ocupação local não são armazenados. Os demais pontos são atribuídos à suas novas posições (região colorida

Algoritmo 4.2 Modelo de medição inverso de um sensor de profundidade com leituras baseadas na distribuição Gaussiana bidimensional.

inverse_range_sensor_model(m_i,x_{sens},z_t**)**

1: Seja x_i, y_i o centro de massa de m_i e $x_{sens} = (x, y, \theta)$ a pose do sensor
2: $r_i = \sqrt{(x_i - x)^2 + (y_i - y)^2}$
3: $\phi_i = atan2((y_i - y), (x_i - x)) - \theta$
4: $k = argmin_j \, \|\phi_i - \theta_{j,sens}\|$
5: $\phi_{k,sens} = \phi_i - \theta_{k,sens}$
6: $prob = p(r_i | z_t^k, \phi_{k,sens})$
7: **se** $r_i > z_t^k$ e $prob < 0.5$ **então**
8: **retorne** l_0
9: **senão**
10: **retorne** $\log prob/(1 - prob)$
11: **fim se**

da figura). É importante observar que não é considerado o erro de localização do carro na transformação entre os seus referenciais. Consequentemente, se as posições foram muito variantes, a informação contida na grade de ocupação local não corresponderá à real. Uma forma de se incorporar o erro de localização do veículo na grade seria incorporar a covariância da matriz de transformação entre os referenciais (1) e (2), obtida por meio do sistema de localização, na grade de ocupação.

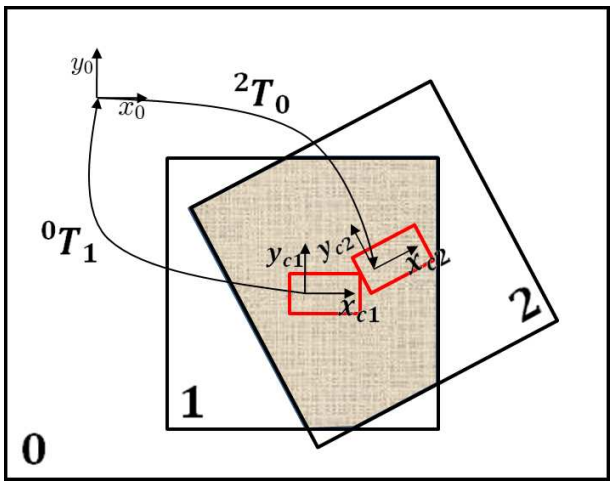


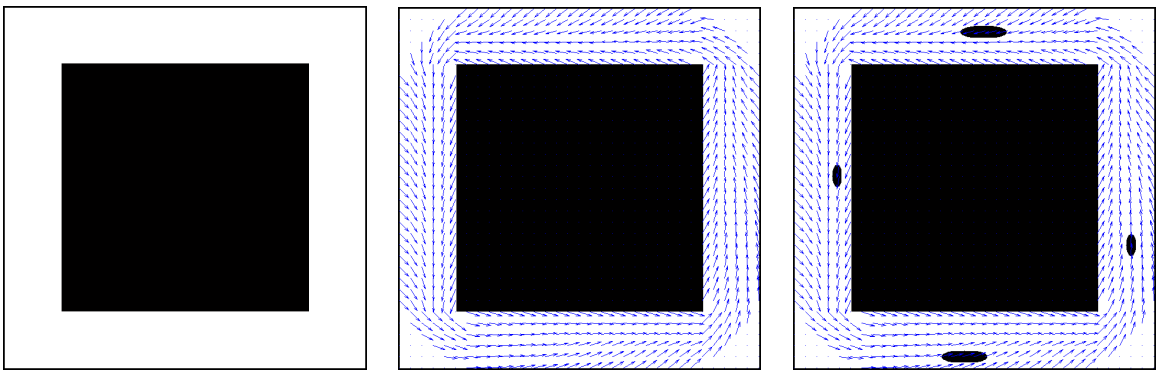
Figura 4.10: Transformações aplicadas à grade de ocupação local anterior (referencial 1) para mapear seus dados na nova grade (referencial 2).

Os dados sensoriais utilizados para construir a grade de ocupação local normalmente cobrem uma região limitada do mapa, definida pelo seu FOV. Apesar deste limitante, é possível ter uma boa estimativa dos elementos presentes ao redor do veículo, devido à atualização da grade de ocupação local com o movimento deste. Se a leitura senso-

rial for então realizada na grade e seus dados fornecidos para algoritmos de desvio de obstáculos, isso permitirá uma movimentação mais segura ao veículo. Para garantir o melhor funcionamento deste método, duas considerações devem ser feitas:

- Todo obstáculo, para ser reconhecido na grade de ocupação local, precisa ter sido reconhecido em algum momento por algum dos sensores instalados no robô; e
- O movimento dos obstáculos que estejam na região monitorada pela grade de ocupação deve ser considerado nulo, pois não serão consideradas técnicas para detectar o movimento dos mesmos.

Para exemplificar o uso da grade de ocupação local para a leitura sensorial, a Figura 4.12 apresenta um automóvel trafegando em um ambiente de simulação (obtido segundo a Figura 4.11) com um sensor para obstáculos com FOV de 43° (similar ao de um sistema de visão estéreo). Nesta mesma figura está a correspondente grade de ocupação local obtida pelo movimento deste veículo, juntamente com a representação de uma leitura sensorial expandida para toda a dimensão do carro. É interessante observar que o obstáculo próximo ao carro continua a ser percebido pela grade, mesmo quando está fora do FOV do sensor real.



(a) (b) (c)

Figura 4.11: Passos realizados para a obtenção do ambiente de simulação. Em (a) está o ambiente inicial próximo a um quarteirão de uma cidade hipotética, onde as ruas estão em branco e os seus limites em preto. Em seguida, um planejamento de movimento inicial por campo vetorial de velocidades (b) foi criado (sem um início e fim específicos) no sentido anti-horário, ao longo de suas ruas. Por fim, em (c) foram adicionados alguns obstáculos, simbolizando carros, pessoas, entre outros, que interferem na segurança do planejamento.

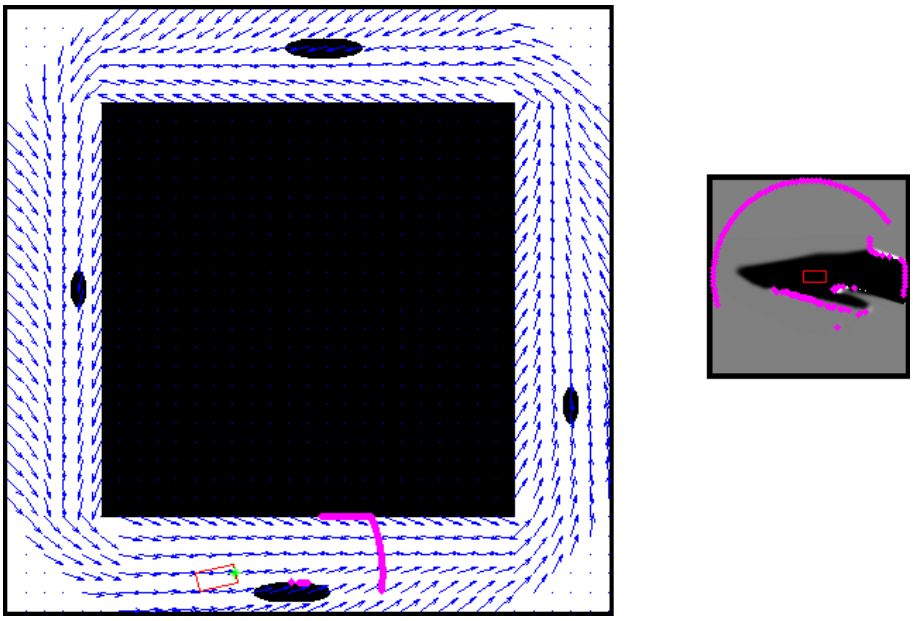


Figura 4.12: Exemplo de um automóvel trafegando no ambiente simulação da Figura 4.11(c) com um sensor de FOV limitado (à esquerda) e sua correspondente grade de ocupação local (à direita). Os pontos em rosa representam as leituras sensoriais em ambas situações.

4.2 Controle de navegação

Controlar a navegação de um carro autônomo é permitir que este realize seu movimento previamente planejado de forma segura. No Capítulo 3, diversas técnicas importantes foram apresentadas para este fim. Algumas prontas para serem incorporadas ao veículo, tais como o sistema de localização e o controle de velocidades, e outras que necessitam adaptações, caso do campo vetorial de velocidades e do Método da Janela Dinâmica (DWA). Esta seção tem por objetivo, apresentar o controle de navegação da Figura 4.1, com a incorporação das técnicas apresentadas no Capítulo 3, mais a percepção do ambiente e os dados de localização do robô. O princípio adotado é o de um controlador híbrido, com uma etapa inicial deliberativa, constantemente validada por uma etapa final reativa.

4.2.1 A etapa de controle deliberativo

O controle deliberativo é uma estratégia de controle clássica na robótica, onde as ações do robô são realizadas por tomadas de decisões. Para realizá-lo, o robô inicialmente deve: (i) conhecer o ambiente no qual se está inserido, o que pode ser feito por meio de

sensores, (ii) planejar um movimento que garanta a sua segurança e, finalmente, (iii) executar este movimento. No entanto, esta é uma estratégia que depende da quantidade de informação envolvida para realizar todo o processamento. Em ambientes externos, como os dos carros autônomos, este processamento é custoso e, por isso, geralmente realizado uma única vez no início do movimento.

A técnica deliberativa escolhida para o controle de navegação do carro autônomo deste trabalho é a de Campos Vetoriais, apresentada na Seção 3.3. Seu planejamento ocorre antes de qualquer movimento, para definir um campo vetorial contínuo que forneça comandos de velocidade (v_x, v_y) em qualquer ponto onde o campo esteja definido. Para tanto, é necessário que o sistema de localização (Seção 3.2) forneça a posição (x, y) do carro a cada instante de interesse. Ao final, o carro deve ser guiado pela região do campo por meio das velocidades retornadas. A Figura 4.12 apresenta um exemplo de campo vetorial de velocidades aplicado em um ambiente de simulação.

Porém, os comandos de velocidade (v_x, v_y) não correspondem às entradas de controle do carro, segundo a Equação (3.1). A solução adotada neste trabalho e apresentada na Figura 4.1 foi aplicar as velocidades em uma função $f(v_x, v_y)$, que fornecesse as velocidades (v_{1VF}, v_{2VF}), para serem atribuídas ao carro. Esta função implementa a Equação (3.6) discutida na Seção 3.4, responsável por guiar o veículo no campo vetorial pela linearização por realimentação estática de estados (FBL). Além das velocidades, ela recebe como entrada os dados de orientação do veículo (θ), fornecido pelo sistema de localização, e ângulo de esterçamento do volante (ϕ).

4.2.2 A etapa de controle reativo

A proposta de controle deliberativo apresentada anteriormente, não garante segurança ao movimento do veículo, pois o planejamento de movimento não considera a dinâmica do ambiente e as dimensões do robô para ser construído. Por isso, ao se acompanhar o diagrama da solução (Figura 4.1), percebe-se que as velocidades (v_{1VF}, v_{2VF}) não são diretamente aplicadas ao robô. Elas passam por um processo de "validação das velocidades". Este processo foi composto por uma solução de controle reativo para o desvio de obstáculos que utiliza as informações da percepção do ambiente, pose e velocidades atuais do robô.

Os controladores reativos são, provavelmente, os mais utilizados na robótica. Eles são capazes de reagirem, em um curto espaço de tempo, às mudanças do ambiente, ações que dão significado ao seu nome. Para conseguir realizar estas ações rápidas, os códigos que os implementa são geralmente simples e criados para uma pequena região do mundo, ou seja, é uma solução local para o desvio de obstáculos. Assim, se forem conciliados os controles reativo e deliberativo em um controlador híbrido, as vantagens de cada um poderão ser incorporadas à solução final. Neste trabalho, o controle reativo foi combinado com o deliberativo para garantir a completude do caminho por meio do Método da Janela Dinâmica (DWA), apresentado na Seção 3.5, porém adaptado para um carro autônomo[3].

Neste sentido, uma contribuição importante foi apresentada em [Rebai et al., 2007], onde o DWA foi aplicado a um veículo autônomo com tração nas rodas traseiras. Devido à esta semelhança, muitos princípios utilizados aqui poderão ser encontrados no trabalho [Rebai et al., 2007].

Como o modelo apresentado na Seção 3.1 é a para um veículo com tração nas rodas dianteiras, estruturalmente diferente do robô *Synchro-Drive* [Fox et al., 1997] e do carro de tração nas rodas traseiras [Rebai et al., 2007], as restrições cinemáticas e dinâmicas que definem a janela dinâmica também serão diferentes. No entanto, os passos a serem seguidos para obter a janela se assemelham aos discutidos na Seção 3.5. Em [Rebai et al., 2007], a janela dinâmica foi definida para a velocidade linear v ($v_1 \cos(\phi)$ para veículos com tração na dianteira) e angular ω ($\dot{\theta}$) do carro, semelhante ao *Synchro-Drive*. Porém, ω não é uma velocidade diretamente aplicável no carro, pois ela depende não linearmente do ângulo de esterçamento do volante ϕ e da velocidade v_1, segundo a Equação (4.9), extraída do modelo cinemático do carro (Equação (3.1)).

$$\dot{\theta} = \omega = v_1 \frac{\sin \phi}{l}. \tag{4.9}$$

[3] Para garantir a completude foi utilizado o mesmo princípio abordado por [Brock and Khatib, 1999], onde combinou-se um planejamento de movimento global (no caso *Wavefront*) com o DWA e foi provado que a solução final era global.

Esta relação, aplicada aos limites definidos pela Equação (3.11), retorna um mapa de velocidades V_s triangular[4], de análise e visualização computacionais mais complexas. Para evitar um mapa triangular e permitir uma relação mais próxima com as reais entradas de controle do carro (v_1, v_2), os limites de V_s foram definidos para v_1 e para ϕ, como segue:

$$V_s = \{(v_1, \phi) | v_1 \in [v_{1min}, v_{1max}], \phi \in [\phi_{min}, \phi_{max}]\}, \tag{4.10}$$

onde v_{1max} e v_{1min} são as velocidades das rodas dianteiras máxima e mínima; e ϕ_{max} e ϕ_{min} são os ângulos de esterçamento máximo e mínimo respectivamente, limitados fisicamente no veículo[5]. Apesar de ϕ não ser uma velocidade, ele possui uma relação com ω (Equação (4.9)), o que permite classificar V_s como um mapa de velocidades. A janela dinâmica V_d será então criada localmente para os valores atuais da velocidade e esterçamento (v_{1a}, ϕ_a), como:

$$\begin{aligned} V_d = \{(v_1, \phi) | v_1 \in [v_{1a} - \dot{v}_{1max} \cdot \triangle t, v_{1a} + \dot{v}_{1max} \cdot \triangle t], \\ \phi \in [\phi_a - v_{2max} \cdot \triangle t, \phi_a + v_{2max} \cdot \triangle t]\}. \end{aligned} \tag{4.11}$$

A janela V_d é dinâmica, porque ela é alterada a cada intervalo de tempo $\triangle t$. A validação dos pares de V_d é feita com o auxílio da função $dist(v_1, \phi)$. Esta função calcula a menor distância entre o carro e os obstáculos à sua volta, quando este trafega com a velocidade v_1 e ângulo de esterçamento ϕ. Aplicando-se esta função em conjunto com as acelerações linear (movimento em linha reta) e angular do veículo (movimento circular) na Equação (3.14), é possível verificar se este conseguirá parar em segurança antes de alcançar o obstáculo. A velocidade angular ω pode ser calculada pela Equação (4.9), cuja derivada temporal fornece a aceleração angular $\dot{\omega}$. Então, o conjunto de velocidades válidas V_a é definido por:

$$V_a = \left\{ (v_1, \phi) | v_1 \cos(\phi) \leq \sqrt{2 \cdot dist(v_1, \phi) \cdot \dot{v}_{1b}}, v_1 \frac{\sin\phi}{l} \leq \sqrt{2 \cdot dist(v, \phi) \cdot \dot{v}_{1b} \frac{\sin\phi}{l}} \right\}, \tag{4.12}$$

[4] O mapa V_s é triangular pelo fato de ω ser linearmente dependente de v_1. Substituindo-se ϕ pelo ângulo máximo e mínimo de esterçamento do volante (ϕ_{max}, ϕ_{min}) na Equação (4.9), o resultado são as duas retas que limitam o V_s e se cruzam em $v_1 = 0$.

[5] Em veículos de passeio convencionais, estes ângulos são iguais a $\pm 29°$.

onde $\dot{v}_{1b}$ é a aceleração de frenagem do veículo. É importante observar que se os valores possíveis de ϕ forem definidos pelo valor atual de v_2, e não por v_{2max} na Equação (4.11), a Equação (4.12) pode ser utilizada para validar o par de velocidades de entrada (v_{1VF}, v_{2VF}). Caso sejam válidas, as velocidades são aplicadas ao robô, senão todo o processamento do DWA é realizado. O espaço resultante de busca V_r pode ser obtido pela Equação (3.15).

Para ilustrar como seriam na prática os conjuntos V_s, V_a, V_d e V_r, a Figura 4.13 foi construída com os dados sensoriais da grade de ocupação local da Figura 4.12, para os valores atuais da velocidade e esterçamento (v_{1a}, ϕ_a). Nesta figura tem-se:

- V_s - Conjunto de todos os pares (v_1, ϕ) alcançáveis pelo veículo. Para determinação deste conjunto somente são considerados os limites físicos do veículo (Equação (4.10)). Os valores de V_s compreendem todos os pares internos ao limite indicado na Figura 4.13;

- V_a - Conjunto similar a V_s excluídas as regiões onde o par (v_1, ϕ) leva à colisão do veículo (regiões escuras na Figura 4.13 - Equação (4.12));

- V_d - Janela dinâmica que não considera regiões de colisão com obstáculos. Subconjunto de V_s calculado em torno da velocidade e do ângulo de esterçamento atuais, dado pelo par (v_{1a}, ϕ_a). Para determinação deste conjunto somente são consideradas a aceleração linear máxima do veículo e a velocidade angular de esterçamento máxima (Equação (4.11)). Na Figura 4.13, é a região limitada pelo retângulo vermelho;

- V_r - Janela dinâmica final. É um conjunto similar a V_d onde são excluídas as regiões onde o par (v_1, ϕ) leva à colisão do veículo. Observe que V_r é um subconjunto de V_a, calculado em torno de (v_{1a}, ϕ_a). A região em verde na Figura 4.13 representa V_r no instante considerado.

Assim como o espaço V_r para o robô *Synchro-Drive* foi calculado por uma função objetivo G, este espaço para o carro também deverá ser calculado. A função foi definida como:

$$G(v_1, \phi) = \alpha \cdot vf1(v_1, \phi) + \beta \cdot dist(v_1, \phi) + \gamma \cdot velocity(v_1, \phi). \quad (4.13)$$

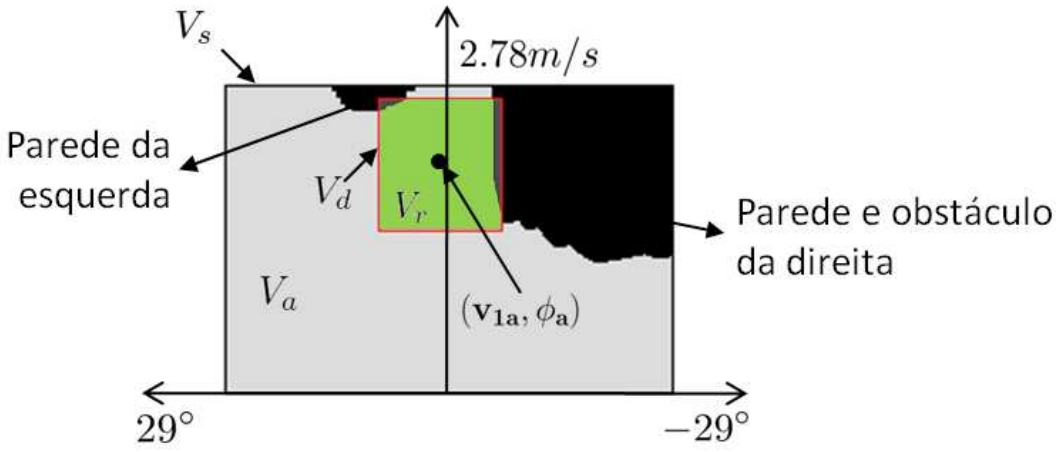


Figura 4.13: Exemplo dos conjuntos V_s, V_a, V_d e V_r, necessários para a aplicação do Método da Janela Dinâmica em um carro autônomo. Estes conjuntos foram obtidos por meio da leitura sensorial na grade de ocupação local da Figura 4.12, com os valores atuais da velocidade e esterçamento do carro representados por (v_{1a}, ϕ_a). Em preto estão todos os pares de (v_1, ϕ) que implicam em um movimento inválido no veículo, com possíveis colisões. Em verde estão as velocidades válidas para a janela dinâmica.

As subfunções $vf1$, $dist$ e $velocity$, que compõe a função objetivo serão detalhadas com maiores detalhes a seguir.

Subfunção $vf1(v_1, \phi)$

O DWA é uma solução local para desvio de obstáculos que, a principio, não garante que o robô convirja para um destino global. Para resolver este problema, Brock e Khatib [Brock and Khatib, 1999] incorporaram ao DWA um método de planejamento de movimento global baseado no *Wavefront*. Esta incorporação foi realizada por uma subfunção adicionada à função objetivo chamada *vf1*. Ela era responsável por calcular, para cada elemento da janela dinâmica válida V_r, uma relação entre a pose do robô e a direção de movimento fornecida pelo *Wavefront*. Seu valor seria máximo quando o robô estivesse orientado segundo o *Wavefront* e mínimo caso contrário. Se comparada à função *heading* do DWA original [Fox et al., 1997], observa-se que ambas tendem a levar o robô para um destino, porém somente a *vf1* garante convergência para o mesmo. Com o princípio de [Brock and Khatib, 1999] em mente, neste trabalho a função *vf1* foi implementada para tentar seguir a orientação do campo vetorial fornecido pelo controle deliberativo.

Primeiramente, as novas poses do carro devem ser preditas a partir da pose inicial e de cada par (v_1, ϕ) de V_r. Isto é feito por meio da Equação (3.3). Para cada nova pose, o vetor de velocidade resultante do controle deliberativo é calculado, pois a orientação do campo varia com o movimento do carro. A diferença entre o ângulo deste vetor e o da orientação predita do carro é chamada de θ_{diff}. Com θ_{diff}, o valor de vf1 será dado pela Equação (4.14). Para aumentar a variação do resultado de vf1 e facilitar sua análise, seus dados foram normalizados entre 0 e 1. Na Figura 4.14 encontra-se um exemplo de como se obter o ângulo θ_{diff}.

$$vf1 = 1 - \frac{|\theta_{diff}|}{\pi}. \tag{4.14}$$

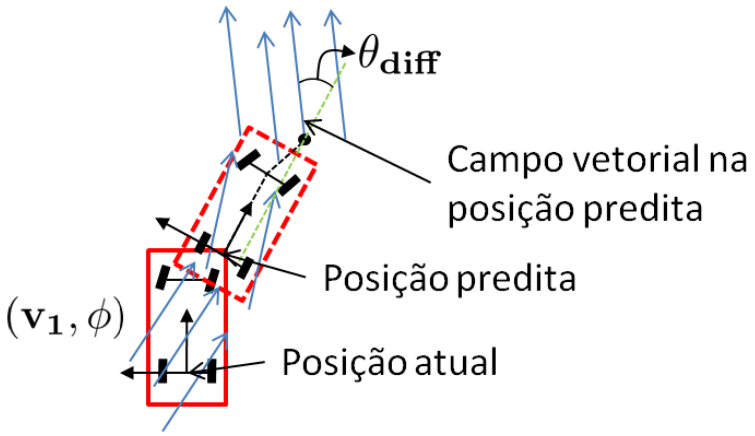


Figura 4.14: Ângulo θ_{diff} obtido para uma pose predita do carro.

Na Figura 4.15(a) tem-se uma avaliação real da função *vf1*. Apesar dos cálculos serem realizados na prática apenas para a janela dinâmica V_r, para visualizar melhor os seus dados, todo o espaço de velocidades V_a foi considerado. Este espaço está representado na Figura 4.13, cujo instante de simulação para construí-lo e o campo vetorial de velocidades considerado são os mesmos da Figura 4.12. Assim como os pares inválidos de v_1 e ϕ (que não pertencem a V_a) foram marcados como pretos na Figura 4.13, na avaliação da Figura 4.15(a) eles receberam peso nulo (zero). O demais valores estão normalizados entre 0 e 1.

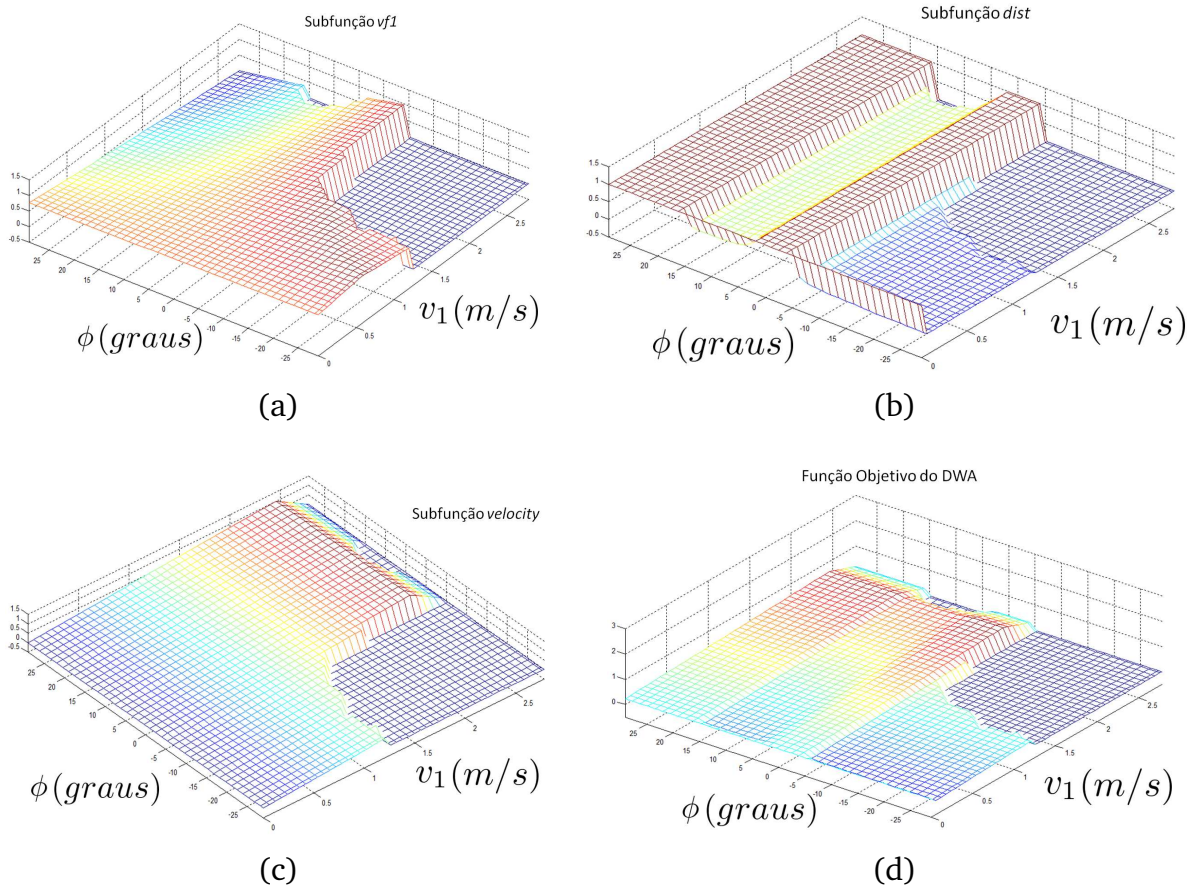


Figura 4.15: Exemplo de avaliação das subfunções do DWA com sua função objetivo resultante. Nesta composição, as subfunções *vf1* (a), *dist* (b) e *velocity* (c), obtidas para o espaço de velocidades V_a da Figura 4.13, foram somadas para gerar a função objetivo $G(v_1, \phi)$ (d), segundo a Equação (4.13). Os valores das constantes foram definidos como $\alpha = 0,04$, $\beta = 0,2$ e $\gamma = 0,4$.

Subfunção $dist(v_1, \phi)$

Calcular a distância para a colisão é uma etapa fundamental para o DWA. Com ela, é possível verificar se um par de controle (v_1, ϕ) gera um movimento seguro e válido (4.12). Esta função também pode ser aplicada na função objetivo $G(v_1, \phi)$ (4.13) a fim de serem priorizadas as maiores distâncias para os obstáculos. Para compor esta função, foi adotada a técnica apresentada em [Arras et al., 2002], onde colisões são detectas em robôs poligonais que seguem trajetórias circulares, como os carros autônomos.

O referencial de análise é o do robô $\{\mathcal{R}\}$, para tornar as equações de análise menos complexas. Definido este referencial, é necessário descrever a trajetória que um ponto O de um obstáculo realiza em $\{\mathcal{R}\}$. Então, o ponto P, onde o obstáculo supostamente

colide com o veículo, pode ser estimado pela interseção dos contornos do robô com esta trajetória. Pontos sem interseção significam que não geram colisão. Para encontrar esta trajetória, primeiramente são definidos os vetores e variáveis (Figura 4.16):

$$\begin{aligned}
\overrightarrow{P_C} &= \overrightarrow{P} - \overrightarrow{C}, \\
\overrightarrow{O_C} &= \overrightarrow{O} - \overrightarrow{C}, \\
\overrightarrow{R_C} &= \overrightarrow{R} - \overrightarrow{C}, \\
|r| &= \left|\overrightarrow{C}\right|, \\
r &= v/\omega, \\
v &= v_1 \cos\phi, \\
r_O &= \left|\overrightarrow{O_C}\right|.
\end{aligned} \tag{4.15}$$

onde a velocidade angular ω é definida segundo a Equação (4.9). Todos os vetores estão representados no sistema de coordenadas do robô $\{\mathcal{R}\}$, com configuração inicial $(x, y, \theta, \phi) = (0, 0, 0, \phi)$. Os dados de entrada que interferem nos cálculos dessa função são a velocidade das rodas frontais v_1, o ângulo das rodas frontais ϕ e os dados sensoriais de obstáculos que definem a posição do ponto O no espaço.

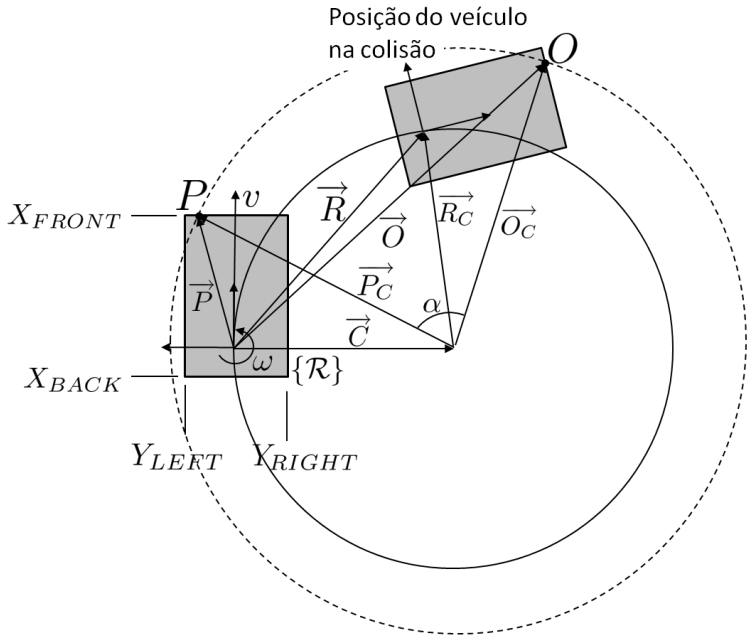


Figura 4.16: Movimento de um ponto O de um obstáculo no referencial do veículo (linha tracejada). A linha contínua representa o movimento do veículo no referencial do mundo.

De acordo com a Figura 4.16, o ponto O descreve um movimento circular em $\{\mathcal{R}\}$ (linha pontilhada), com centro igual ao ao movimento que o carro descreve (linha contínua). O raio desta circunferência é dado por r_O, que é o comprimento do vetor $\overrightarrow{O_C}$. A equação deste círculo é então definida por:

$$r_O^2 = x_{coll}^2 + (y_{coll} - r)^2. \tag{4.16}$$

Para encontrar o ponto de colisão $P = (x_{coll}, y_{coll})$, para a parte frontal, lateral esquerda, lateral direita e traseira do veículo, basta resolver o seguinte sistema de equações:

- Dianteira do carro, onde $y_{coll} \in [Y_{RIGHT}, Y_{LEFT}]$

$$\left.\begin{array}{c} x_{coll}^2 + (y_{coll} - r)^2 = r_O^2 \\ x_{coll} = X_{FRONT} \end{array}\right\} \Rightarrow \begin{array}{c} y_{coll} = r \pm \sqrt{r_O^2 - X_{FRONT}^2} \\ x_{coll} = X_{FRONT} \end{array} \tag{4.17}$$

- Lateral esquerda do carro, onde $x_{coll} \in [X_{BACK}, X_{FRONT}]$

$$\left.\begin{array}{c} x_{coll}^2 + (y_{coll} - r)^2 = r_O^2 \\ y_{coll} = Y_{LEFT} \end{array}\right\} \Rightarrow \begin{array}{c} x_{coll} = \pm\sqrt{r_O^2 - (Y_{LEFT} - r)^2} \\ y_{coll} = Y_{LEFT} \end{array} \tag{4.18}$$

- Lateral direita do carro, onde $x_{coll} \in [X_{BACK}, X_{FRONT}]$

$$\left.\begin{array}{c} x_{coll}^2 + (y_{coll} - r)^2 = r_O^2 \\ y_c = Y_{RIGHT} \end{array}\right\} \Rightarrow \begin{array}{c} x_{coll} = \pm\sqrt{r_O^2 - (Y_{RIGHT} - r)^2} \\ y_{coll} = Y_{RIGHT} \end{array} \tag{4.19}$$

- Traseira do carro, onde $y_{coll} \in [Y_{RIGHT}, Y_{LEFT}]$

$$\left.\begin{array}{c} x_{coll}^2 + (y_{coll} - r)^2 = r_O^2 \\ x_{coll} = X_{BACK} \end{array}\right\} \Rightarrow \begin{array}{c} y_{coll} = r \pm \sqrt{r_O^2 - X_{BACK}^2} \\ x_{coll} = X_{BACK} \end{array} \tag{4.20}$$

Caso algum dos sistemas anteriores retorne uma solução real, o veículo colidirá no ponto P. A distância percorrida d pode ser encontrada a partir do ângulo α, definido entre os vetores $\overrightarrow{C_C}$ e $\overrightarrow{O_C}$, e o raio r de movimentação do carro, segundo a Equação (4.21).

Na Figura 4.16 é possível verificar que o valor de α depende do sentido de movimento do veículo (definido por v_1) e se a curva é para a esquerda ou para a direita.

$$d = \alpha \cdot r. \tag{4.21}$$

Como o ponto de obstáculo O pode colidir em mais de um lugar no veículo, a distância de colisão d_{coll} será a menor distância dentre as demais possíveis, calculada por:

$$d_{coll} = \min\left(d_{FRONT}, d_{LEFT}, d_{RIGHT}, d_{BACK}\right). \tag{4.22}$$

O valor final d_{coll} foi então normalizado por um valor limite de leitura d_{max}, para tornar a subfunção $dist(v_1, \phi)$ em igual extensão às demais subfunções. Na Figura 4.15(b), está um exemplo da avaliação desta subfunção para todas as entradas do espaço de velocidade V_a da Figura 4.13, criado no instante de simulação da Figura 4.12. As regiões inválidas foram definidas como zero.

Subfunção $velocity(v_1, \phi)$

Uma característica importante do campo vetorial proposto na Seção 3.3 é o fornecimento de velocidades de movimento que respeitem as limitações do ambiente, por meio dos pesos atribuídos às regiões do campo. Portanto, a velocidade de movimento proposta pelo campo vetorial precisa ser priorizada pelo DWA no cálculo da função objetivo $G(v_1, \phi)$ (4.13). Este é o papel da subfunção $velocity(v_1, \phi)$: fornecer valores numéricos que favoreçam a escolha de pares (v_1, ϕ) cuja velocidade linear das rodas dianteiras seja igual à do campo vetorial.

Esta subfunção foi definida como:

$$velocity(v_1, \phi) = \begin{cases} \dfrac{v_1}{(v_{1max} - v_{1min})} & \text{se } v_1 \leq v_{1VF}, \\ \\ \dfrac{(v_1 - v_{1max})}{(v_{1VF} - v_{1max})} & \text{se } v_1 > v_{1VF}. \end{cases} \tag{4.23}$$

Para os dados de V_a no instante de simulação apresentado na Figura 4.12, a análise da subfunção $velocity(v_1, \phi)$ é mostrada na Figura 4.15(c).

Maximização da função objetivo

O procedimento final ao se realizar o DWA em um dado instante é calcular o máximo valor da função objetivo $G(v_1, \phi)$ (4.13) na janela definida por V_r. Por se tratar de uma região menor, não é necessário utilizar programas de otimização para encontrar o ponto de interesse. Este ponto, na prática, pode ser obtido por varredura direta da janela resultante.

Para o espaço válido V_a, a Figura 4.15 apresenta o resultado da função objetivo para a soma das demais subfunções para o instante definido pela Figura 4.12. Para exemplificar o resultado real, a janela dinâmica definida por V_r está apresentada no gráfico da Figura 4.17 para o mesmo instante anterior. Nele, a seta indica o valor máximo da função, obtido pela varredura direta.

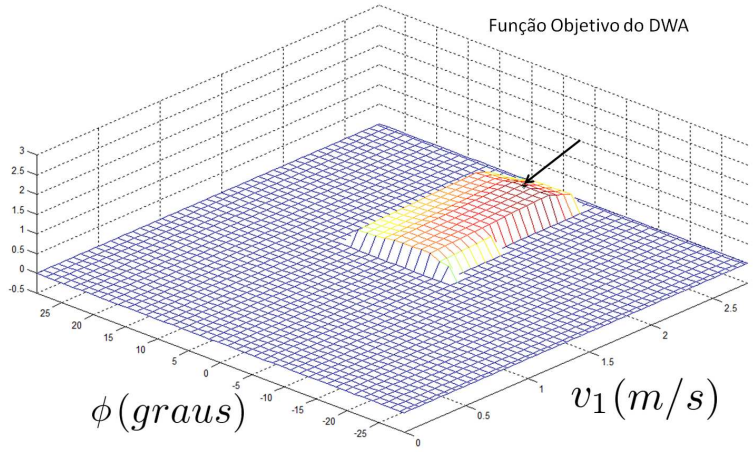


Figura 4.17: Exemplo de janela dinâmica resultante da análise dos pares definidos por V_r nas subfunções $vf1$, $dist$ e $velocity$ da Figura 4.15. A janela foi calculada para $(v_{1a}, \phi_a) = (2.46m/s, 1°)$ e o seu valor máximo está indicado pela seta.

Sejam os valores ótimos obtidos pela maximização de $G(v_1, \phi)$ iguais a (v_{1otm}, ϕ_{otm}), eles devem ser convertidos em entradas de controle para o carro iguais a (v_{1DWA}, v_{2DWA}). A conversão desses valores pode ser realizada por:

$$\begin{cases} v_{1DWA} = v_{1otm}, \\ v_{2DWA} = \dfrac{\phi_{otm} - \phi_a}{\triangle t}. \end{cases} \tag{4.24}$$

No próximo capítulo serão apresentados os resultados experimentais realizados em simulação e em um veículo autônomo real. Para tanto, a solução aqui apresentada (Figura 4.1) foi implementada a fim de verificar a sua funcionalidade.

Capítulo 5

Resultados Experimentais

Para avaliar a solução apresentada anteriormente foram utilizadas duas plataformas de teste. A primeira foi realizada em simulação, com o movimento do veículo expresso pela Equação (3.3). A segunda foi realizada em um carro autônomo que está em desenvolvimento na Universidade Federal de Minas Gerais, o CADU. Neste capítulo, estes resultados serão apresentados e discutidos a fim de validar a solução proposta.

5.1 Resultados em ambiente de simulação

Pelos recursos que possui, o Matlab[1] foi o software de desenvolvimento utilizado para simular a solução de navegação deste trabalho. Neste software, existem diversas funções que facilitam tanto a implementação da solução quanto a visualização dos resultados gráficos. Estes benefícios foram importantes na hora de avaliar os resultados, corrigir possíveis erros de código e definir as constantes do Método da Janela Dinâmica (DWA), por exemplo. Porém, o uso destes recursos não permitiu focar no desempenho computacional.

A aplicação foi construída linearmente, onde cada ciclo do programa executa cada umas das tarefas apresentadas na Figura 4.1. Antes de entrar no ciclo de simulação, foi definido o campo vetorial para um mundo planar, sem considerar a presença de obstáculos para a sua concepção, como foi discutido na Seção 3.3. Em palavras, cada ciclo da simulação realiza as seguintes etapas:

1. Leitura sensorial no ambiente de simulação;

[1] http://www.mathworks.com/products/matlab/.

2. Inclusão da leitura na grade de ocupação local;
3. Leitura na grade de ocupação local;
4. Cálculo do vetor de velocidades (v_x, v_y) para a posição atual do veículo;
5. Cálculo das velocidades (v_{1VF}, v_{2VF}) pela linearização por realimentação estática (FBL);
6. Validação das velocidades pelo Método da Janela Dinâmica (DWA);
7. Simulação do movimento do carro com as velocidades válidas (v_{1DWA}, v_{2DWA});
8. Atualização da grade de ocupação local com o movimento do carro.

Para se assemelhar a um ambiente real, obstáculos estáticos foram acrescentados ao longo do campo para representar carros e pessoas no caminho do robô. As dimensões do veículo na simulação também foram definidas iguais as reais, o mesmo acontecendo com o campo de visão (FOV) dos sensores e o seu máximo alcance. Apesar da solução proposta ter sido criada para ser independente dos sensores utilizados, a quantidade, o FOV e o máximo alcance destes sensores podem influenciar na segurança de movimento e nas máximas e mínimas velocidades permitidas para o carro. Por isso, fez-se necessário considerar estas informações na simulação para se ter uma boa estimativa do comportamento do carro no ambiente real. No entanto, não foram consideradas dinâmicas complexas entre o carro e o ambiente na simulação, tais como as derrapagens por exemplo. Isto significa que qualquer velocidade de controle será inteiramente convertida em velocidade no carro.

Na Figura 4.12 está representado um instante de simulação para um veículo imerso num campo vetorial contínuo que não considera os obstáculos presentes. Nela podem ser observadas as dimensões do veículo (em vermelho), a posição do sensor no carro (ponto verde) e as leituras sensoriais (pontos em rosa), com os seus limites de alcance de 17 metros e o FOV de $43°$, similares ao de uma câmera de visão estéreo. A seguir serão apresentados alguns dos experimentos realizados neste ambiente de simulação, com a finalidade de configurar o DWA[2] e validar a solução de navegação proposta, para

[2] A configuração do DWA compreende desde a definição dos valores das constantes α, β e γ ao tamanho da janela dinâmica com sua discretização, sendo necessário compreender como cada um deles interfere no movimento final do veículo.

posteriormente aplicá-la no veículo real. Com a simulação pretende-se, também, expor algumas possíveis limitações da metodologia quanto à percepção do ambiente.

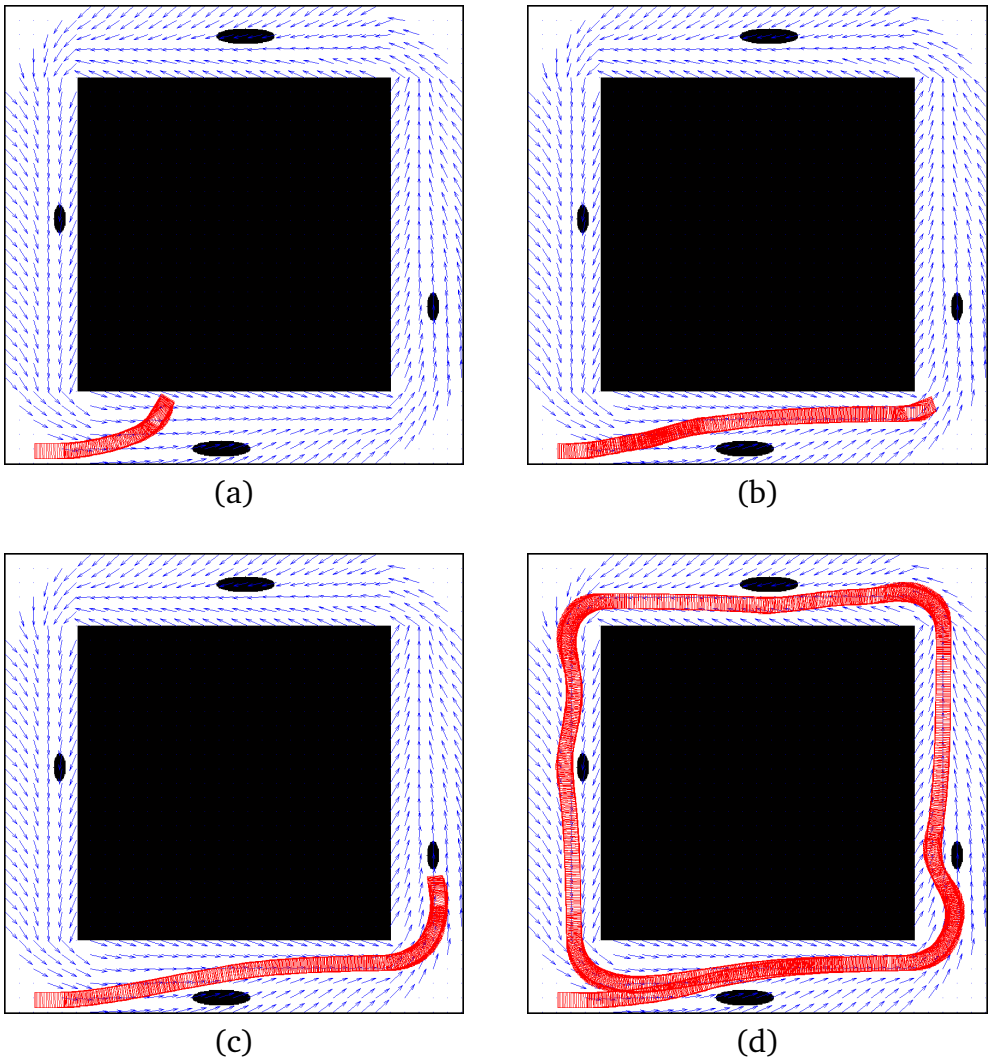
5.1.1 Simulação 1: Configuração do DWA

O DWA, como discutido nos capítulos anteriores, utiliza a composição das subfunções *vf1*, *dist* e *velocity* em uma função objetivo G (4.13), ponderadas pelas respectivas constantes α, β e γ, obtidas via simulação. Para verificar a influência das constantes na solução do DWA, várias simulações foram realizadas.

A escolha dos valores de α, β e γ, que atendam a navegação segura, deve ser baseada na compreensão de como cada um deles interferem na solução final. Em separado, as subfunções podem até mover o veículo, mas não garantem que o mesmo desvie de obstáculo ou alcance um objetivo. O uso somente da subfunção $velocity$, por exemplo, gera um movimento sem rumo, definido com o foco apenas em manter o módulo da velocidade v_{1VF} fornecida pelo campo vetorial. Este movimento pode ser visualizado na Figura 5.1(a), com o trajeto feito pelo veículo expresso pelos retângulos em vermelho. É importante observar que mesmo sem rumo, o veículo ainda é capaz de parar antes de bater nos obstáculos, pois apenas velocidades seguras são fornecidas pelo DWA. O uso de outra subfunção, a *dist*, tenta manter o veículo o mais distante o possível dos obstáculos ao seu redor. No entanto, o movimento se cessa ao encontrar um ponto onde os obstáculos estão equidistantes ao carro, como visto na Figura 5.1(b).

A limitação no movimento do veículo pode ser contornada com a união de duas ou mais subfunções à função objetivo. Com a união das subfunções *dist* e *velocity*, por exemplo, o carro passa a mover-se sem rumo e longe dos obstáculos. Porém, como o veículo continua sem rumo, este pode se deparar com situações que bloqueiam mais facilmente o seu movimento. A Figura 5.1(c), mostra a simulação para esta condição.

Para contornar a falta de rumo no movimento do veículo e tentar evitar situações bloqueantes, a subfunção *vf1* pode ser utilizada. Seu uso combinado com a subfunção *velocity* é uma forma possível de se realizar a navegação segura e orientada ao campo vetorial. Apesar de completo, seu movimento pode ser mais próximo dos contornos dos obstáculos e mais fechado nas curvas, de acordo com a Figura 5.1(d).



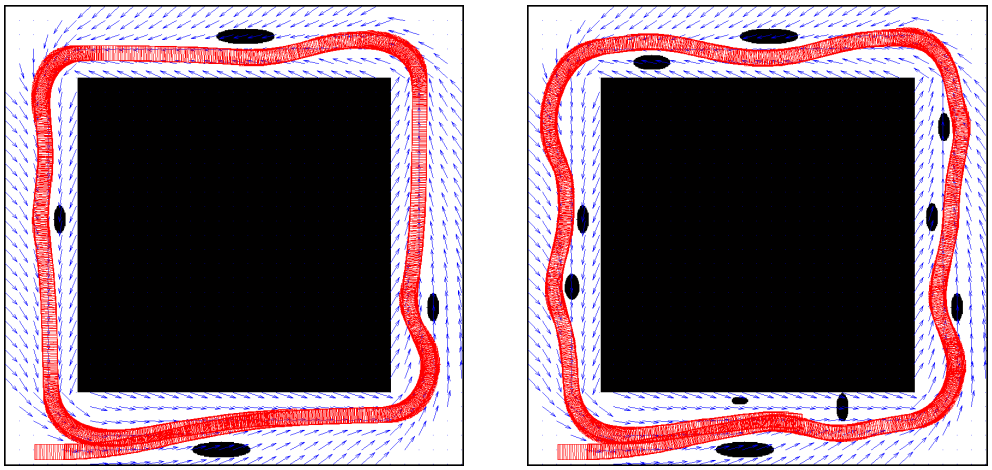
(a) (b)

(c) (d)

Figura 5.1: Resultados de simulação para a avaliar como as subfunções do DWA influenciam no movimento do veículo. Nesta composição, em (a) utilizou-se apenas a subfunção *velocity* e em (b) apenas a *dist* para realizar o movimento do veículo. Já em (c), fez-se a composição de duas subfunções, *dist* e *velocity*.Em (d) combinou-se somente as subfunções *vf1* e *velocity*.

Ao adicionar a função $dist$ na solução anterior, o movimento resultante será suavizado e ficará mais distante dos obstáculos. Esta é a importância de se ajustar as constantes da função objetivo, para conciliar as vantagens de cada subfunção na solução final pelo aumento ou não de cada constante. A escolha destes valores foi empírica, apenas com a informação do que cada constante altera na solução final. Inicialmente, foram atribuídos valores iguais para cada constante e verificado o movimento resultante do veículo na simulação. Pelo movimento resultante, as constantes foram ajustadas uma a uma com pequenas variações levando sempre em consideração como elas afetam nesse movimento. Os valores finais escolhidos foram $\alpha = 0,04$, $\beta = 0,2$ e $\gamma = 0,4$, o que não

quer dizer que seja a solução ótima. O efeito deles em simulação podem ser visualizados na Figura 5.2, onde na letra (a) o resultado pode ser comparado ao apresentado na letra (d) da Figura 5.1.



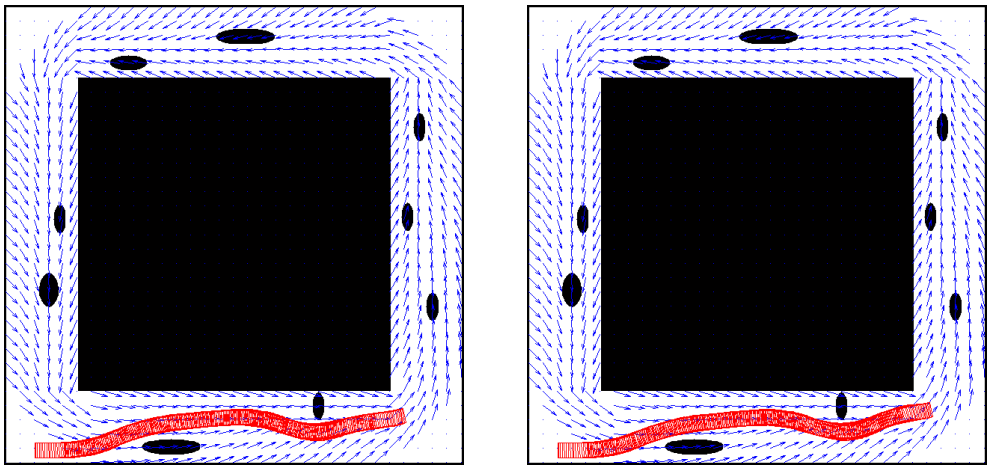
(a) (b)

Figura 5.2: Resultados de simulação para as constantes definidas para o DWA iguais a $\alpha = 0,04$, $\beta = 0,2$ e $\gamma = 0,4$. Em (a) está o resultado para o mesmo ambiente da Figura 5.1 e em (b) o resultado é para um ambiente com mais obstáculos.

5.1.2 Simulação 2: Análise dos comandos de controle

Analisar os comandos de controle de velocidade gerados pela solução é outro item importante que pode ser explorado na simulação. Saber se os comandos são suaves ou se o controlador consegue estabilizar o robô em uma trajetória são fundamentais para viabilizar sua aplicação em um veículo real. Para verificar estes comandos, foram consideradas duas simulações realizadas em um mesmo ambiente, alterando-se apenas a discretização da janela dinâmica utilizada. Os trajetos gerados por essas simulações estão apresentados nas letras (a) e (b) da Figura 5.3. O ambiente utilizado possui trechos que necessitam da intervenção do controlador para desviar ou não de obstáculos.

A análise da janela dinâmica parte do princípio da discretização da mesma, pois é essa discretização que torna o método viável ou não. No Apêndice A é discutida a análise de complexidade da solução proposta para a navegação de veículos autônomos, no qual é possível verificar como a discretização na janela interfere no tempo de execução do algoritmo. As duas discretizações utilizadas foram definidas a partir do número de



(a) (b)

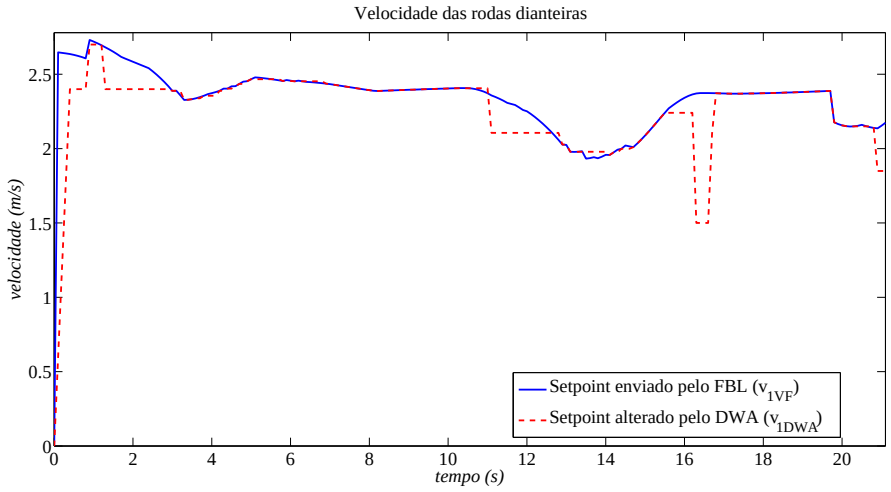
Figura 5.3: Trajetos simulados para diferentes janelas dinâmicas, sendo uma pouco discretizada (a) e outra muito discretizada (b).

medidas desejadas entre os valores máximo e mínimo dos pares (v_1, ϕ). A janela menor foi definida em 5×5 e a maior em 15×15.

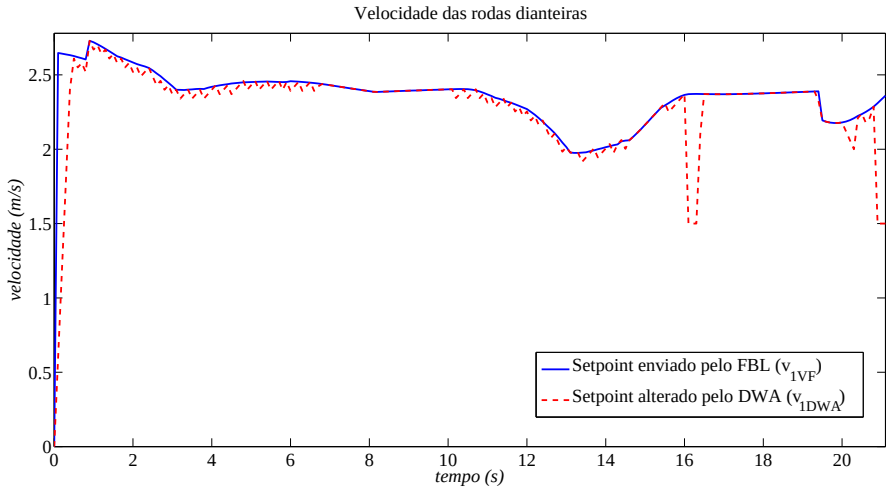
Nas simulações apresentadas na Figura 5.3, os comandos de velocidade fornecidos ao robô foram calculadas no campo vetorial, processadas pela etapa de linearização por realimentação estática (FBL) e validadas pelo Método da Janela Dinâmica (DWA). Os comandos de velocidade gerados estão dispostos nos gráficos das Figuras 5.4 e 5.5, para as velocidades linear e de esterçamento das rodas dianteiras, respectivamente.

Ao se comparar os caminhos realizados pelo veículo na Figura 5.3, percebe-se que houve pouca influência da quantidade de dados nas janelas dinâmicas. No entanto, os comandos de saída gerados são significativamente diferentes, de acordo com as Figuras 5.4 e 5.5. Nas duas figuras, as velocidades tendem sempre a seguir as definidas pelo campo vetorial e FBL na ausência de obstáculos, como proposto pela metodologia do DWA (Seção 4.2). Porém, as diferenças nas janelas dinâmicas implicam em comandos com mais valores intermediários ou não.

Na prática, saltos maiores nos comandos de controle de velocidade das rodas dianteiras são suavizados pela própria dinâmica do veículo e de seus atuadores (acelerador e freio), como pode ser observado nos resultados apresentados em [Freitas and Pereira, 2010]. Esta dinâmica funciona como um filtro passa baixas para estes comandos. No caso da velocidade de esterçamento das rodas dianteiras, sua aplicação é realizada pelo atuador no volante, mostrado no trabalho de [Freitas et al., 2009] e apresentado nos re-



(a)

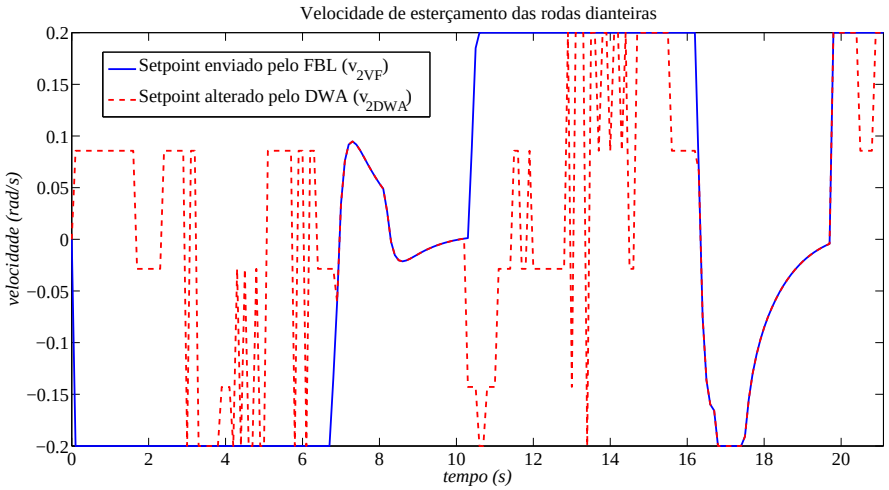


(b)

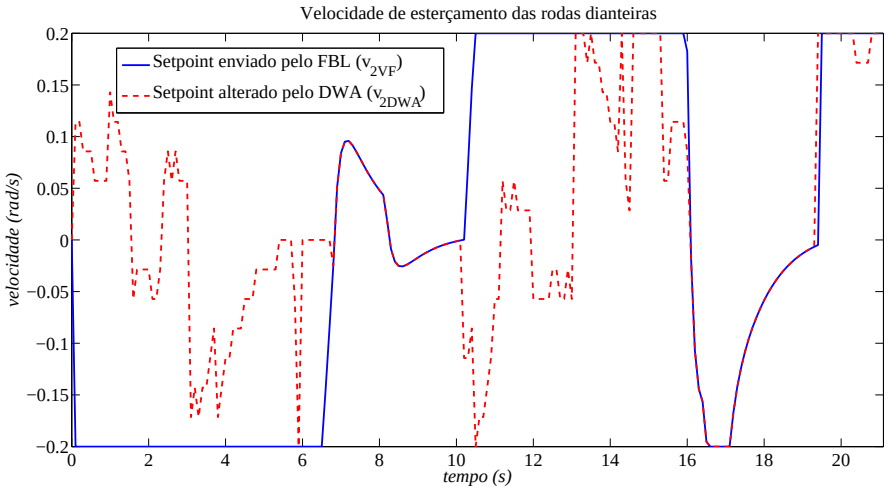
Figura 5.4: Comparação entre os comandos de controle da velocidade linear das rodas dianteiras (v_1), gerados pelo campo vetorial e FBL, com a validação feita por uma janela dinâmica pouco discretizada (a) e outra muito discretizada (b).

sultados em um veículo autônomo real (Seção 5.2). Assim como o acelerador e o freio possuem dinâmicas que suavizam os comandos de velocidade das rodas dianteiras, o atuador do volante também possui dinâmicas próprias que suavizam os comandos de esterçamento.

Uma análise também pode ser feita para o ângulo de esterçamento do veículo predito para cada uma das velocidades da Figura 5.5, ou seja, o ângulo de esterçamento que o carro teria depois de um intervalo de tempo $\triangle t$ com a aplicação de cada uma das velocidades. Os gráficos resultantes estão dispostos na Figura 5.6. Nestes gráficos,



(a)

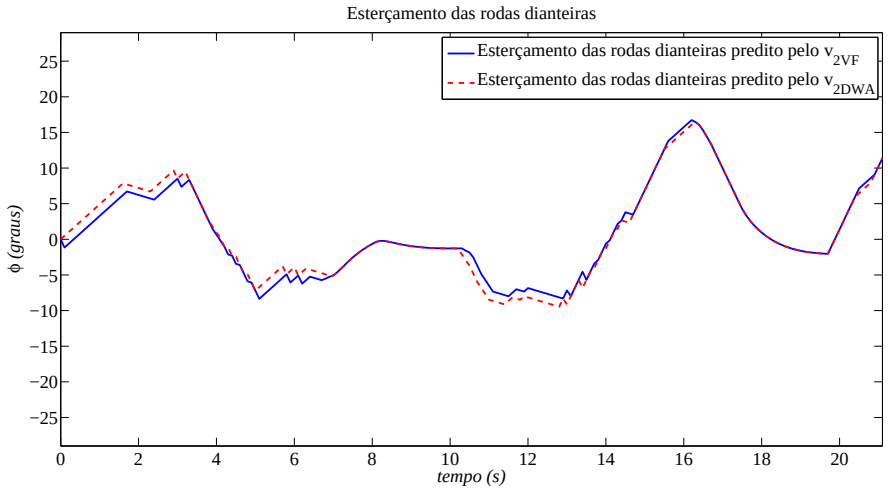


(b)

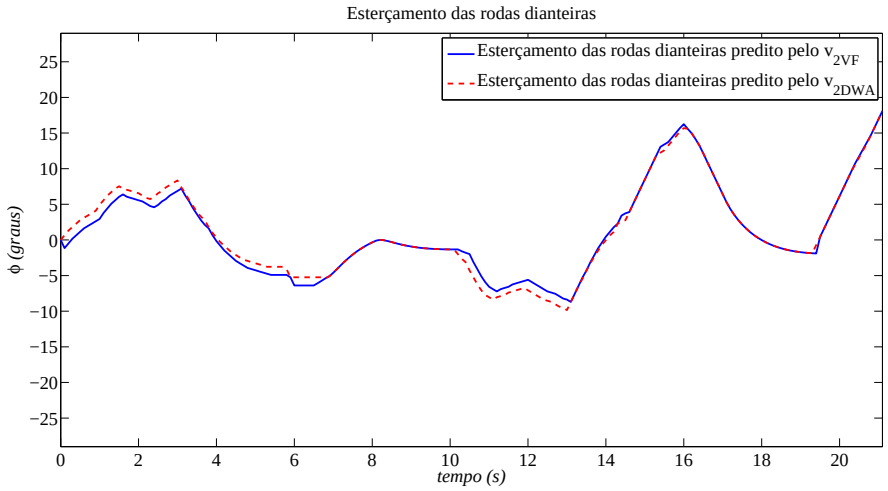
Figura 5.5: Comparação entre os comandos de controle da velocidade de esterçamento das rodas dianteiras (v_2), gerados pelo campo vetorial e FBL, com a validação feita por uma janela dinâmica pouco discretizada (a) e outra muito discretizada (b).

mesmo com uma variação brusca da velocidade, o ângulo de esterçamento para todos os casos varia suavemente, o que implica em uma trajetória sem oscilação.

A partir dos gráficos anteriores, verifica-se que os comandos de velocidade são capazes de controlar um veículo suavemente. Outro fator que pode ser observado foi que o aumento da discretização da janela dinâmica interferiu pouco nos resultados finais. Concluiu-se, portanto, que como o uso de uma janela pouco discretizada é mais eficiente computacionalmente, esta deve ser utilizada para realizar a navegação segura de veículos autônomos.



(a)



(b)

Figura 5.6: Comparação entre os os ângulos de esterçamento preditos pelos correspondentes comandos de controle de velocidade de esterçamento das rodas dianteiras (v_2). Estes comandos foram gerados para uma janela dinâmica pouco discretizada (a) e outra muito discretizada (b).

5.1.3 Simulação 3: Limitações sensoriais

Apesar da solução proposta neste trabalho utilizar uma grade de ocupação local, não é garantido que o seu uso permita que o veículo se mova em total segurança em um ambiente. De acordo com as considerações realizadas na Subseção 4.1.3, a grade apenas auxilia na percepção do ambiente estático ao redor do veículo autônomo, mas os sensores utilizados são fundamentais para garantir a segurança. A detecção dos obstáculos presentes depende de dois elementos importantes que compõem os sensores, o campo de visão (FOV) e a máxima distância de alcance. Se um obstáculo próximo

ao veículo não tiver passado em nenhum instante pelo FOV do sensor e dentro da sua faixa de distâncias perceptíveis, o veículo não poderá tentar evitar uma colisão com este obstáculo.

Para exemplificar como o FOV dos sensores interferem na criação da grade de ocupação local, foram criadas duas grades de ocupação para o ambiente de simulação da Figura 4.7(a). A primeira, apresentada também na Figura 4.7(b), foi concebida por um FOV de $180°$, semelhante ao de um sensor a laser convencional. Na segunda, o FOV foi reduzido para $43°$, próximo do valor de uma câmera de visão estéreo, o que resultou na grade da Figura 5.7.

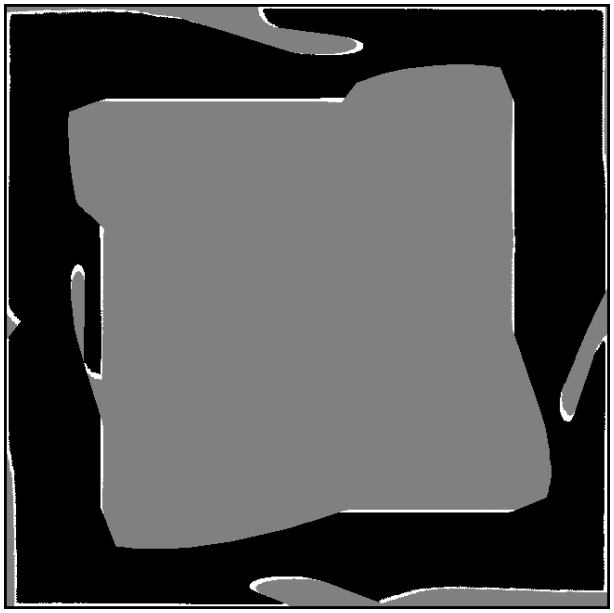


Figura 5.7: Exemplo de uma grade de ocupação gerada por um carro com um sensor com FOV de $43°$, semelhante ao de uma câmera estéreo, a partir do ambiente de simulação da Figura 4.7.

A principal diferença entre as grades de ocupação apresentadas está na quantidade de região inexplorada existente nelas. Na grade gerada pelo FOV menor, as bordas finais dos obstáculos e as regiões próximas de curvas mais fechadas não foram bem reconhecidas. Como a grade de ocupação local é apenas uma redução da grade de ocupação global, ambas terão as mesmas limitações decorrentes do FOV menor. Por isso, ao se utilizar apenas uma câmera de visão estéreo como sensor de percepção do ambiente, não é possível garantir total segurança ao movimento de um carro autônomo. Para exemplificar uma falta de segurança causada por limitações no FOV de sensores, tem-se a Figura 5.8. Ela foi composta com instantes de navegação de um veículo autônomo,

quando este navegava com um sensor de FOV $180°$ (esquerda) e $43°$ (direita). Note que em nenhum instante o obstáculo à esquerda do veículo foi detectado pelo sensor de FOV menor, o que no caso gerou uma colisão inevitável, fato que não ocorreu com o sensor de FOV maior.

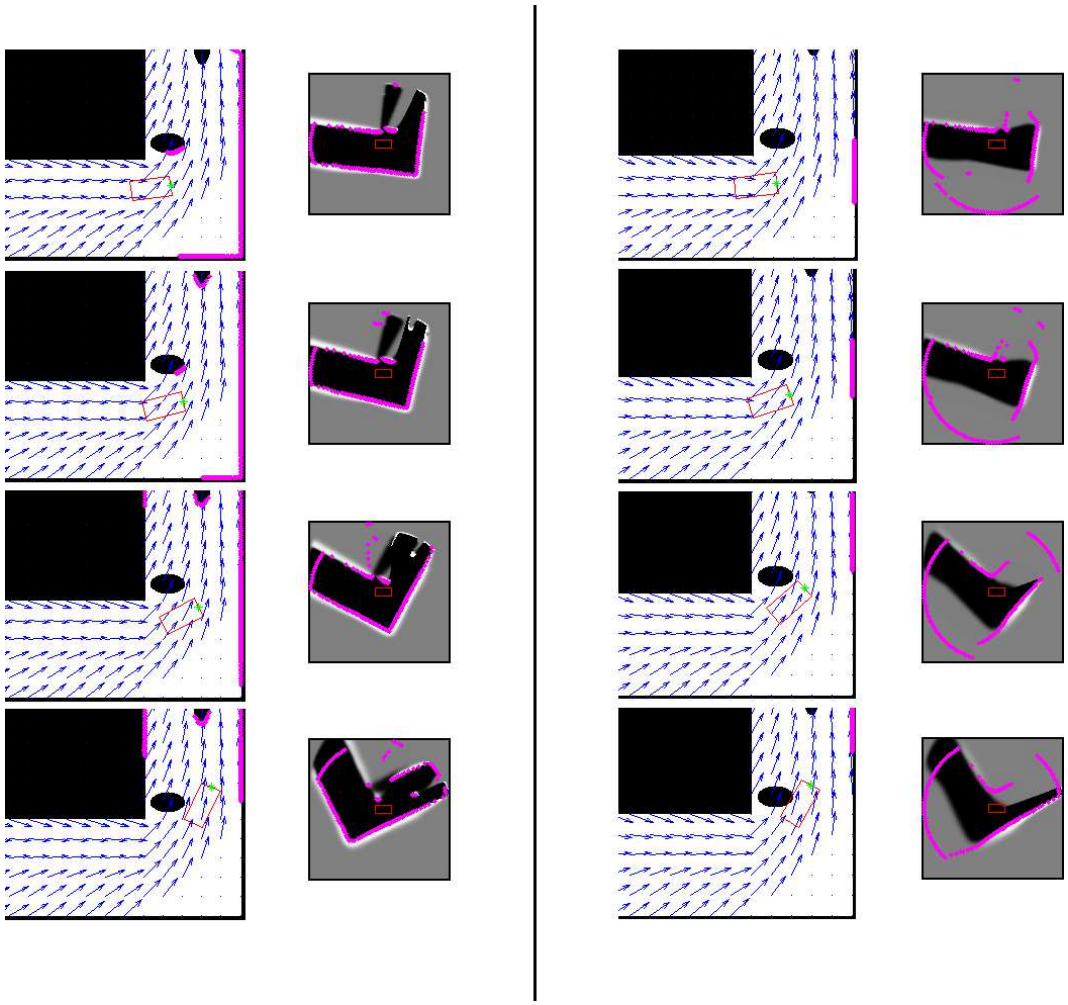


Figura 5.8: Movimentos simulados de um veículo trafegando com sensores distintos e sua corresponde grade de ocupação local. Nas etapas da esquerda, o ambiente foi percebido com o mesmo FOV de um sensor a laser ($180°$). Enquanto que, nas da direita, foi utilizado um sensor com FOV semelhante ao de um sistema de visão estéreo ($43°$). Pela limitação sensorial, o movimento da direita ocasionou uma colisão, mesmo com o auxílio de uma grade de ocupação local.

Outra limitação sensorial analisada foi a causada pelo seu alcance máximo, ou seja, a maior distância que um obstáculo consegue ser reconhecido. Esta distância máxima, se aplicada na Equação (4.12), resulta em uma máxima velocidade válida (v_1) para as

rodas dianteiras[3]. Então, mesmo que o campo vetorial forneça velocidades superiores à máxima permitida, o DWA terá que limitá-la para a segurança do veículo. Esta ação limitadora está representada no gráfico da Figura 5.9, onde um veículo foi imerso em um campo vetorial com velocidades superiores às permitidas pelos seus sensores de obstáculos. O caminho resultante realizado se assemelha ao apresentado na Figura 5.3(a).

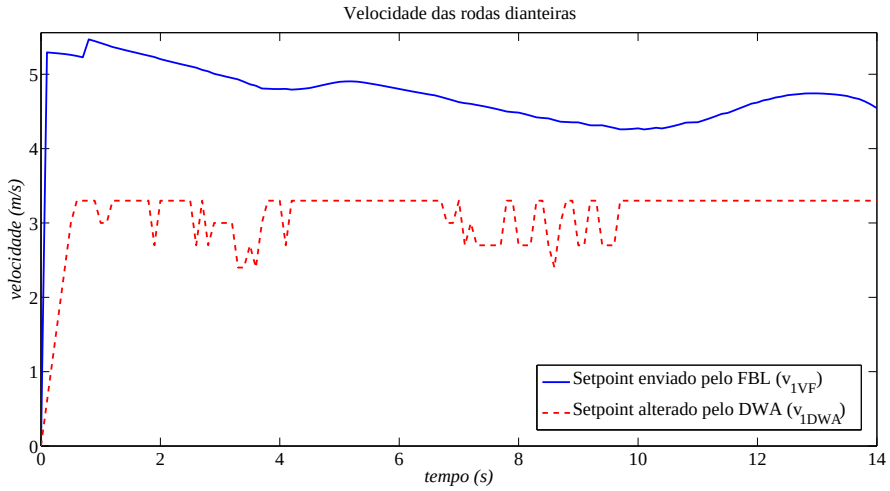


Figura 5.9: Gráfico da velocidade calculada pelo campo vetorial e FBL (v_{1VF}), e seu valor real limitado pelo DWA (v_{1DWA}). O limite está em função do máximo alcance do sensor de obstáculos. O caminho realizado pelo veículo é semelhante ao apresentado na Figura 5.3(a) e o alcance máximo do sensor de obstáculos é de 17 metros.

5.2 Resultados em um veículo autônomo real

Os testes realizados em simulação mostraram-se totalmente aplicáveis em veículos autônomos reais. O CADU é um destes veículos, o qual foi escolhido para testar na prática a metodologia deste trabalho. Nesta seção, a aplicação desenvolvida para a simulação foi adaptada para funcionar neste veículo. Para tanto, serão apresentados o veículo CADU, as aplicações desenvolvidas e alguns experimentos realizados para validar a solução.

5.2.1 O veículo autônomo CADU

O Carro Autônomo Desenvolvido na Universidade Federal de Minas Gerais (UFMG), também conhecido como CADU, é um veículo que está em desenvolvimento pelo Grupo

[3] Esta função pode ser limitada também pela máxima desaceleração possível para a velocidade v_1, mas esta é uma limitação física do carro.

de Pesquisa e Desenvolvimento de Veículos Autônomos (PDVA)[4] desde 2007. O CADU utiliza a plataforma de um Chevrolet Astra 2003 para receber toda a automação necessária para torná-lo autônomo. Na Figura 5.10 está uma imagem atual deste veículo.

Figura 5.10: Carro autônomo desenvolvido na UFMG (CADU).

A automação embarcada do CADU possui atuação em todos os dispositivos necessários para fazer um carro convencional se mover. Além dessas atuações, alguns sensores estão disponíveis para auxiliar em tarefas de localização e locomoção, dentre os quais estão um Sistema de Posicionamento Global (GPS), uma Unidade de Medição Inercial (IMU), um sensor de posição angular do volante e sensores de velocidade das rodas dianteiras. A comunicação com o carro e seus dispositivos é realizada via USB/Serial, com auxílio em alguns casos de microcontroladores PIC18F2550, fabricado pela Microchip. Maiores detalhes sobre estes recursos e sua implantação no CADU estão disponíveis em [Freitas et al., 2009].

Com o trabalho de [Freitas et al., 2009], por exemplo, uma das entradas de controle do veículo foi controlada, a de esterçamento das rodas dianteiras v_2. Nele, foi apresentada a implantação de um atuador comercial no volante do CADU capaz de atuá-lo com velocidade constante. A solução consiste de um motor MAXON RE40 acoplado ao

[4] O PDVA é um grupo multidisciplinar registrado no CNPq em 2007. Atualmente realiza pesquisas na UFMG no desenvolvimento de instrumentação aeronáutica, na identificação e controle de pequenos helicópteros, no desenvolvimento de veículos aéreos não tripulados (UAVs) e na automação de um automóvel de passeio.

eixo do volante com um controlador EPOS 24/5, que implementa o controle de posição, velocidade e corrente do motor.

Para este trabalho um novo sensor foi adicionado ao carro: uma câmera de visão estéreo chamada *Bumblebee2*, fabricada pela *Point Grey Research* (PGR)[5], e apresentada na Figura 5.11. Ela é constituída por duas câmeras coloridas de 640×480 com lentes de 6mm e FOV de $43°$, que são afixadas a uma estrutura de alumínio e ligadas a um barramento IEEE1394. A câmera possui características importantes, tanto de *hardware* quanto de *software*, que facilitam a obtenção e o tratamento de imagens estéreo. Nas características de *hardware* destacam-se o fato de o par estéreo ter sido calibrado na sua fabricação e ser sincronizado a partir do barramento IEEE1394. Em *software* são fornecidas funções otimizadas para se criar o mapa de disparidade[6], encontrar os pontos em 3D por meio de triangulação e filtrar erros no mapa de disparidade, por exemplo. Assim, todos os passos descritos na Figura 4.2, anteriores à construção do mapa de disparidade "V", já estão prontos com um bom desempenho computacional e qualidade nos resultados.

Figura 5.11: Câmera de visão estéreo *Bumblebee2*.

Alguns sensores a laser também foram adquiridos para serem incorporados ao CADU, mas a instalação deles no veículo não ocorreu em tempo hábil para os testes deste trabalho. No entanto, um deles foi considerado nas aplicações finais desenvolvidas para o CADU, o laser de modelo LMS 291 - S05, fabricado pela SICK[7]. Ele é capaz de realizar medidas a distâncias de até 80 metros, em um FOV de $180°$ com leituras a cada $0,5°$ com frequências de aquisição que chegam a até 70 Hz.

Com a automação do CADU, muitos trabalhos puderam ser realizados. Alguns exemplos, já apresentados em capítulos anteriores, são o sistema de localização por fusão

[5] http://www.ptgrey.com/.

[6] O processo de criação do mapa de disparidade utiliza o método de correlação da Soma das Diferenças Absolutas [Hiroshi et al., 1995].

[7] http://www.sick.com/.

sensorial [Santos, 2009] (Seção 3.2), o controle de velocidade das rodas dianteiras por lógica Fuzzy apresentado em [Freitas and Pereira, 2010] (Seção 3.6) e a navegação do veículo com o auxílio da visão estéreo [Lima and Pereira, 2010] (Subseção 4.1.1). Outro trabalho importante criado para o carro foi um sistema de controle que o permitia seguir trajetórias pré-definidas no espaço [Sabbagh et al., 2010].

Existem, também, muitos trabalhos em andamento. São sistemas para estacionar autonomamente, controlar à distância o veículo via controle remoto, perceber pequenos obstáculos, simular o CADU e todos os seu sensores em um ambiente virtual, redes internas de dados mais eficientes, entre outras aplicações. Este leque de sistemas em desenvolvimento mostra que o CADU é uma plataforma em constante evolução.

5.2.2 Software de navegação

A solução proposta na Figura 4.1 é complexa e depende de muitos recursos simultaneamente, sejam eles computacionais ou do próprio hardware. Para melhorar o desempenho e conseguir um tempo entre cada ciclo próximo a 100 milisegundos, tempo este utilizado como base na simulação, a aplicação final foi dividida em quatro subprogramas: *CarWaypoints*, *CarNavigationControl*, *CarCameraSensor* e *CarLaserSensor*. As etapas descritas para o programa de simulação na Seção 5.1, foram divididas nestes subprogramas. Por serem programas distintos, alguns deles precisam se comunicar para trocar dados processados. Nestes casos, a comunicação utilizada foi a *ethernet*, com o protocolo UDP, que permite uma transferência rápida dos dados e programas funcionando em diferentes máquinas com mais recursos computacionais disponíveis. Todos os programas foram criados em C++, com auxílio de algumas bibliotecas específicas, tais como a *OpenCV* e a *Boost*.

O primeiro programa, *CarWaypoints*, é responsável pela etapa de planejamento de movimento com a criação do campo vetorial que será seguido pelo carro. Este programa foi adaptado daquele criado em [Pereira et al., 2008] para a navegação de um UAV. Nesta adaptação, foi adicionada uma interface gráfica em .Net que permite a interação do usuário para definir e visualizar o campo gerado. A implementação computacional do campo foi realizada pelo Engenheiro Diego Rocha Rebelo. Ele também permite

executar os demais programas listados anteriormente. Na Figura 5.12 está apresentada a sua interface gráfica.

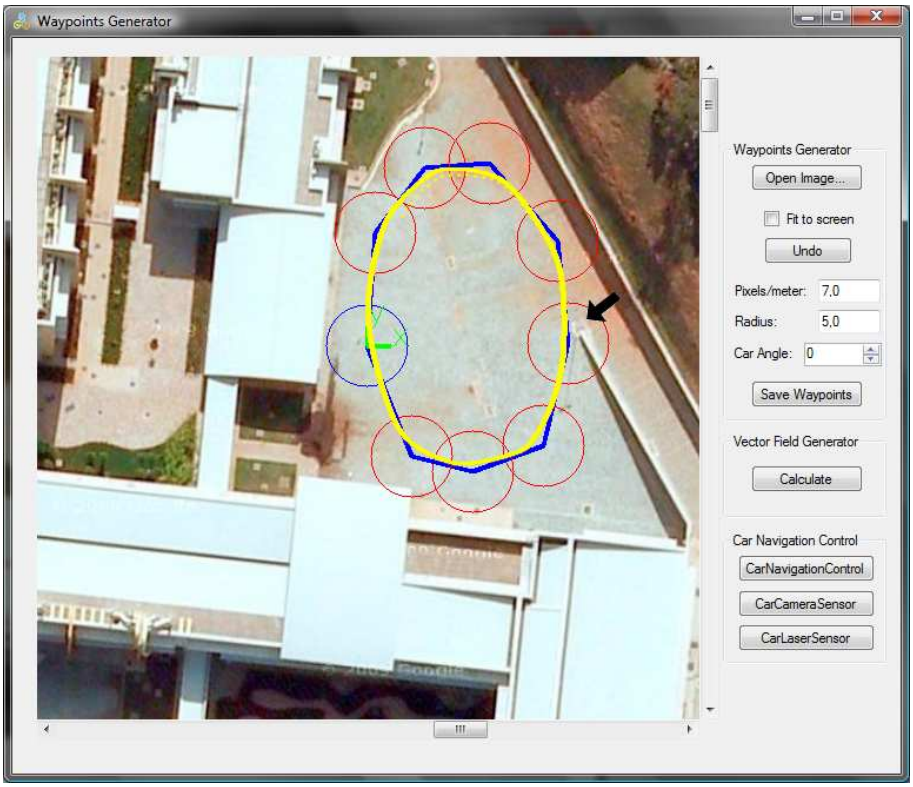


Figura 5.12: Interface gráfica do programa *CarWaypoints*. Na imagem central estão exibidos o referencial (x, y) do carro em verde, os círculos que circunscrevem os *waypoints* em azul e vermelho e a integral do campo em amarelo. A imagem central é uma projeção em escala do mundo e sua relação com os pixels pode ser definida pelo usuário juntamente com o raio dos círculos dos *waypoints*. A seta indica um obstáculo no mundo real não considerado pelo campo vetorial.

Nos desafios do DARPA ([DARPA, 2005] e [DARPA, 2008]), veículos participantes deveriam seguir uma sequência de *waypoints* definidos pela organização do evento. Para se assemelhar à forma de navegação utilizada pelo DARPA, o campo vetorial definido pelo *CarWaypoints* também utilizou o princípio dos *waypoints* fornecidos pelo usuário. Neste caso, cada *waypoint* possui um círculo, de raio também atribuído pelo usuário, que o circunscreve. Ao se conectar estes círculos por suas tangentes, obtém-se um túnel de ligação entre eles. A este túnel é aplicado o algoritmo de campo vetorial discutido na Seção 3.3, o qual realiza a sua triangulação e define os vetores bases que garantam que o campo seja contínuo e mantenha o veículo em seu interior. Para permitir que o veículo tente sempre se mover no túnel, nas regiões externas foi definido um campo vetorial direcionado para ele. Porém, nas bordas do túnel não é garantida

a continuidade do campo. Mais detalhes sobre os passos anteriores podem ser obtidos em [Pereira et al., 2008]. Na interface do programa, apresentada na Figura 5.12, pode ser observado um exemplo de *waypoints* definidos pelo usuário, com seus círculos circunscritos (em azul e vermelho), e a integral do campo entre eles (em amarelo). O campo vetorial criado a partir destes *waypoints* e todos os elementos intermediários necessários para gerá-lo podem ser observados na Figura 5.13.

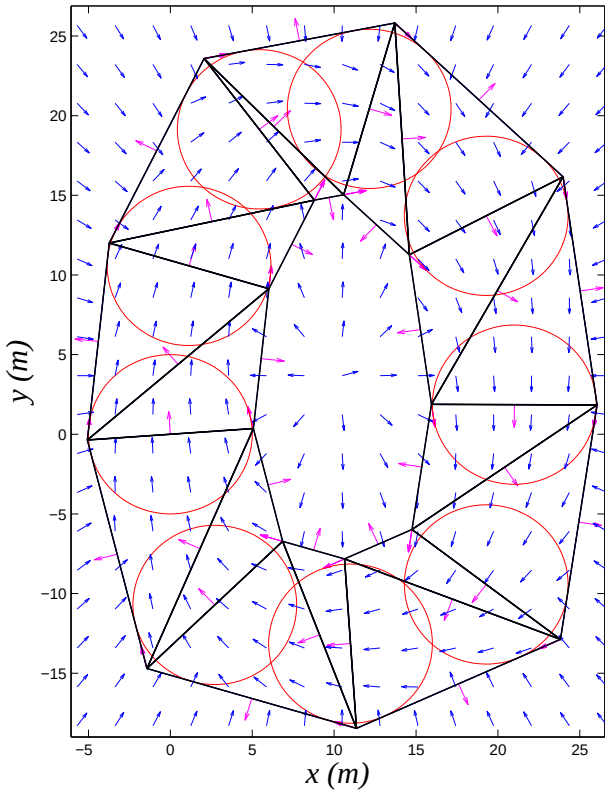


Figura 5.13: Campo vetorial circular resultante dos *waypoints* definidos na interface do programa *CarWaypoints*. Além do campo vetorial, estão representados, também, os elementos utilizados para criar este campo, tais como os círculos dos *waypoints*, as tangentes que definem os túneis, os triângulos e os vetores bases.

O campo vetorial não é fixo no espaço, ou seja, ele é criado para uma posição inicial relativa do CADU, definida como origem $(0, 0)$. Desta forma, o mesmo campo vetorial pode ser utilizado em diferentes localizações do veículo e não está sujeito a erros de exatidão do GPS. Seus dados são armazenados em disco para serem utilizados pelo programa *CarNavigationControl*. Este programa é o responsável por receber os dados sensoriais e aplicá-los à grade de ocupação local, calcular os comandos de velocidade vindos do campo vetorial, validá-los em relação aos obstáculos detectados e aplicá-los

no carro. Na Figura 4.1, a única etapa não realizada neste aplicativo é a leitura sensorial do mundo com a detecção de obstáculos.

Para receber os dados sensoriais, no *CarNavigationControl* existe um servidor sempre à espera por novas conexões de programas que forneçam dados sensoriais de obstáculos. A cada novo dado recebido, a grade de ocupação local é atualizada, igualmente quando acontece uma mudança na pose do veículo.

A comunicação com o CADU é uma das funções mais importante realizada pelo programa. Nesta aplicação é onde estão implementados os procedimentos de atuação (câmbio, direção, acelerador e freio), e aquisição de dados sensoriais (GPS, IMU, velocidade das rodas dianteiras e posição angular do volante). Com estes dados, o programa realiza as etapas de localização por fusão sensorial (descrita na Seção 3.2) e controle de velocidade (apresentada na Seção 3.6).

No entanto, a incorporação da solução para a localização do CADU neste trabalho não foi trivial e passou por diversas correções tanto conceituais quanto práticas. Já a metodologia nela adotada manteve-se inalterada. Conceitualmente, o modelo utilizado não correspondia ao de um carro com tração nas rodas dianteiras, necessário para este trabalho. Desta forma, o modelo foi substituído pelo apresentado na Seção 3.1, o que implicou, também, na alteração das matrizes que definem o Filtro de Kalman (KF).

As correções práticas foram realizadas nos aplicativos finais da solução. Antes era uma aplicação para cada sensor, responsável pela aquisição de seus dados, e uma central para receber os dados dos demais programas e realizar a fusão sensorial. Nesta configuração, ocorriam muitos problemas de comunicação com os sensores e, devido ao grande número de programas, era difícil de manuseá-los. Agora, as aplicações foram condensadas em classes, referenciadas por um único programa, no caso o *CarNavigationControl*. As comunicações com os sensores também foram melhoradas para tornar a classe mais robusta.

O sistema final foi capaz de fornecer as novas poses do veículo sempre relativas à primeira. Porém, pela quantidade de erros nas medidas dos sensores, muitas variações ainda ocorreram com o movimento do veículo.

Para auxiliar o usuário, no *CarNavigationControl* são exibidas duas janelas em tempo de execução, uma com a grade de ocupação local atual e outra com informações gerais de controle do veículo. Dentre estes dados tem-se a localização, os comandos de veloci-

dade gerados por cada etapa de controle de navegação e algumas mensagens de alerta para o usuário. Uma dessas mensagens, por exemplo, é gerada na ausência de sensores de obstáculos por um tempo maior que 3 segundos. Este alerta cessa o movimento do veículo até a chegada de um novo dado sensorial. Para compreender melhor a estrutura da janela de dados e as informações que ela fornece, uma cópia desta janela pode ser visualizada na Figura 5.14.

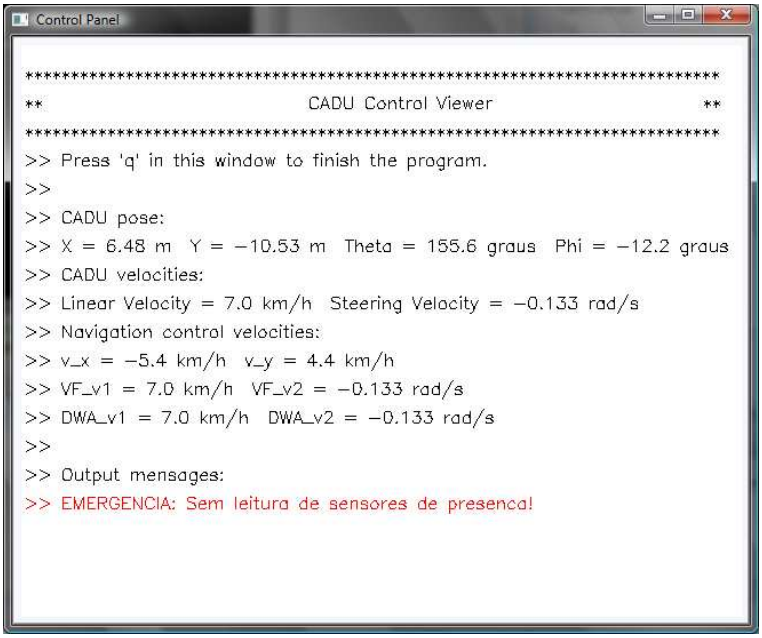


Figura 5.14: Janela de dados do programa *CarNavigationControl* com informações gerais do CADU e possíveis alertas.

As duas aplicações restantes, *CarCameraSensor* e *CarLaserSensor*, são responsáveis por adquirir os dados dos sensores de obstáculos, processá-los e fornecê-los ao *CarNavigationControl* como coordenadas polares (raio r_{obst} e ângulo θ_{obst}), conforme definido na Figura 4.1. O primeiro programa realiza a aquisição dos dados da câmera de visão estéreo *Bumblebee2* e implementa a técnica de detecção de obstáculos discutida na Subseção 4.1.1, para fornecer o resultado ao *CarNavigationControl*. O segundo programa recebe os dados do laser SICK, considera que suas leituras são paralelas ao solo (Subseção 4.1.2) e os envia como foram adquiridos para o *CarNavigationControl*. Para fornecer uma visualização amigável dos dados sensoriais lidos ao usuário, os dois programas apresentam uma janela de exibição com o formato equivalente ao da imagem da direita da Figura 4.6. No Apêndice A é apresentada uma breve análise sobre a complexidade dos programas desenvolvidos nesta solução.

5.2.3 Experimentos

Os experimentos no CADU foram realizados com os mesmos parâmetros configurados na etapa de simulação. Durante os testes, foi adotada a configuração computacional apresentada na Figura 5.15, a qual utiliza:

- Uma câmera estéreo Bumblebee2 conectada a um *hub Firewire* pelo barramento IEEE1394 para a alimentação da câmera;
- Um computador portátil com processador Intel Pentium Core II Duo de 1.83GHz e 4G de memória RAM rodando Windows Vista, para executar o programa *CarCameraSensor* com os dados da câmera, recebidos do *hub* pelo barramento IEEE1394;
- Um segundo computador portátil equipado com Intel Pentium Core II Duo de 1.66GHz e 2G de memória RAM rodando Windows Vista, para receber os dados de obstáculos via *Ethernet* e aplicá-los ao programa *CarNavigationControl* para atuar nos dispositivos do CADU por USB/Serial.

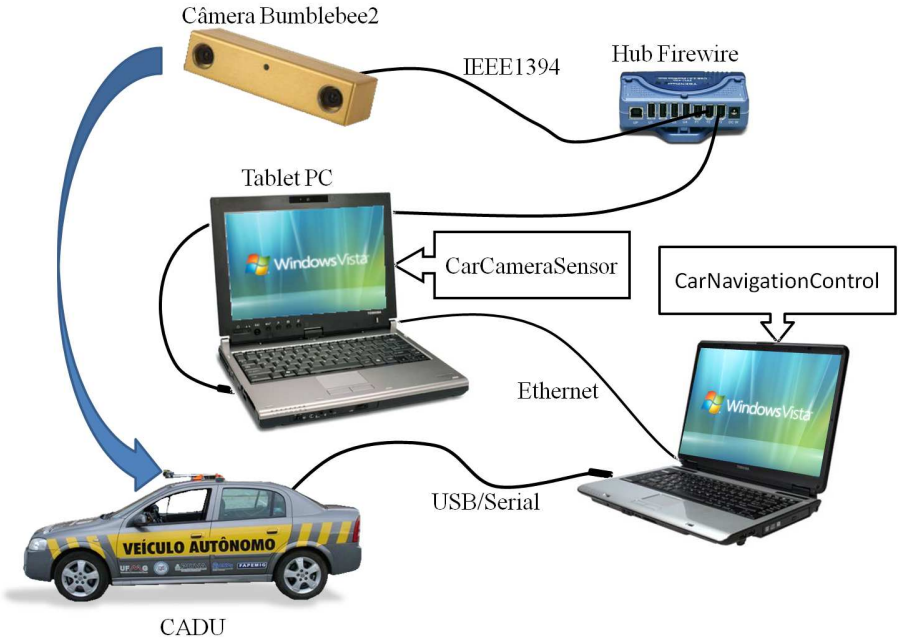


Figura 5.15: Configuração adotada para realizar os experimentos no CADU com uma câmera estéreo e dois computadores portáteis.

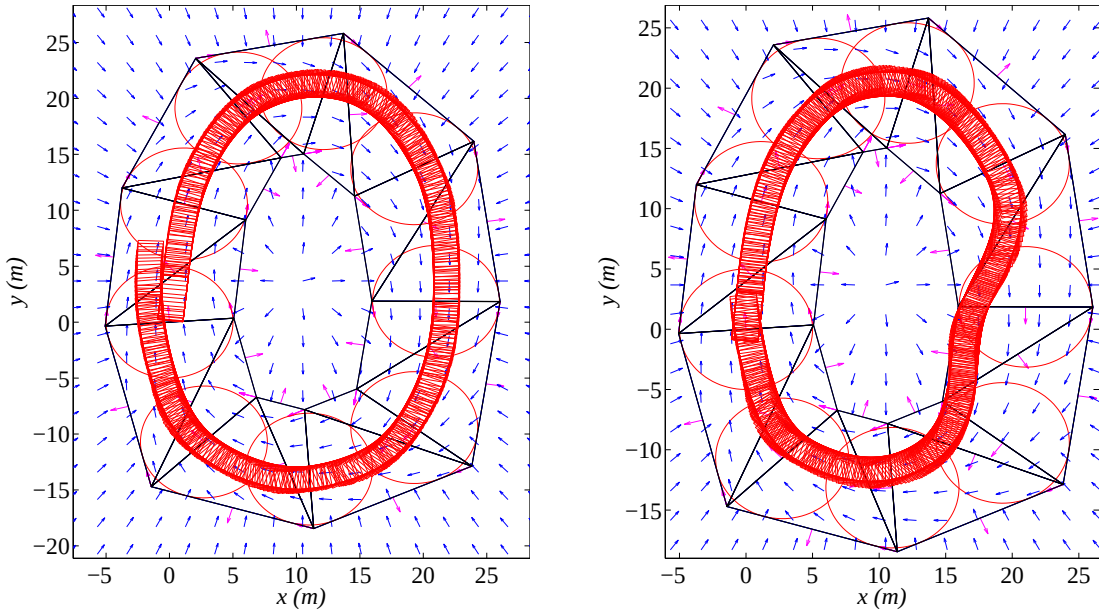
Dos vários experimentos feitos no CADU, dois foram escolhidos para representar a solução final. Para ambos foi criado um campo vetorial elipsoidal, com a diferença de no

segundo existir um obstáculo que necessitava da intervenção do programa para realizar o desvio. O campo final é o mesmo criado na Figura 5.12 e representado em detalhes na Figura 5.13. Na imagem da Figura 5.12 é possível verificar um obstáculo (indicado por uma seta) ao longo da trajetória proposta pelo campo vetorial. Porém, como a posição inicial do campo vetorial é relativa à do veículo, se este estiver mais deslocado para a esquerda no ambiente real (esquerda da imagem), o obstáculo não estará ao longo da integral do campo.

Foi com o princípio do campo ser relativo que os experimentos foram realizados neste trabalho. O primeiro experimento, com a pose inicial do veículo mais a esquerda que a definida na imagem da Figura 5.12, gerou o caminho que pode ser observado na Figura 5.16(a). Este caminho, como era de se esperar, ocorreu na ausência de obstáculos. Resultado diferente do apresentado na letra (b) desta mesma figura, realizado para o CADU na posição inicial mostrada na imagem da Figura 5.12. No segundo experimento, apesar de se utilizar o mesmo campo vetorial para guiar o CADU, foi necessário utilizar os recursos de detecção e desvio de obstáculos para garantir uma trajetória segura ao veículo. Estes resultados mostram que o veículo tende sempre a seguir o campo vetorial com uma trajetória suave, mesmo quando está desviando de um obstáculo.

Os comandos de velocidade gerados para cada umas das situações podem ser observados nos gráficos das Figuras 5.17 e 5.18. Nos gráficos do experimento com obstáculos (letra b em ambas as figuras), os comandos de desvio de obstáculos foram gerados entre os instantes de 15 e 20 segundos.

Ao analisar estes gráficos percebe-se claramente as diferenças na ação do DWA na ausência e na presença de obstáculos. Os comandos de velocidade do DWA somente sofreram grandes variações quando havia algum obstáculo próximo. Nos demais casos eles seguiram o campo vetorial. Na velocidade linear das rodas dianteiras (gráficos da Figura 5.17), foi possível verificar que, apesar de gerar ações de controle suaves, a dinâmica do veículo demora para responder às variações e não percebe pequenas variações. Outro problema com esta velocidade é a alta oscilação na sua leitura sensorial, o que dificulta a ação do controlador fuzzy da Seção 3.6. Há, também, a presença de ruídos (falsos obstáculos), que geram comandos bruscos de desaceleração. No entanto, estes comandos são filtrados pela dinâmica do veículo e pela grade de ocupação local e não interferem no movimento do veículo.

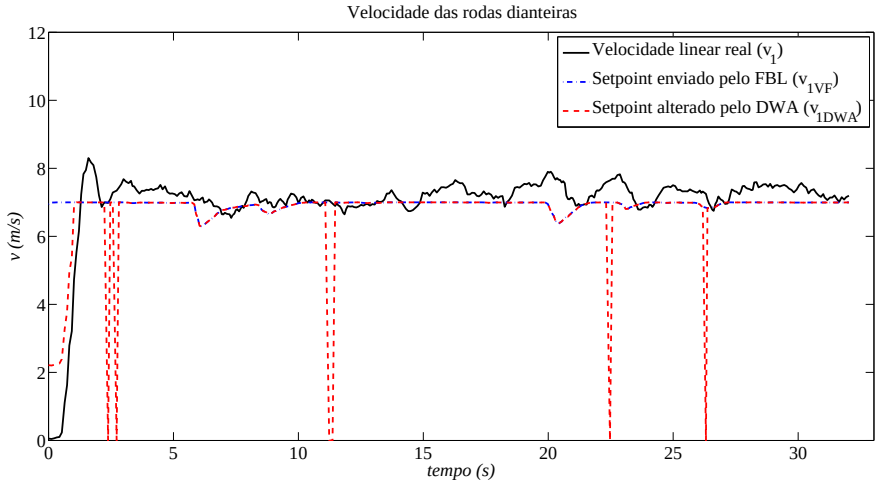


(a) (b)

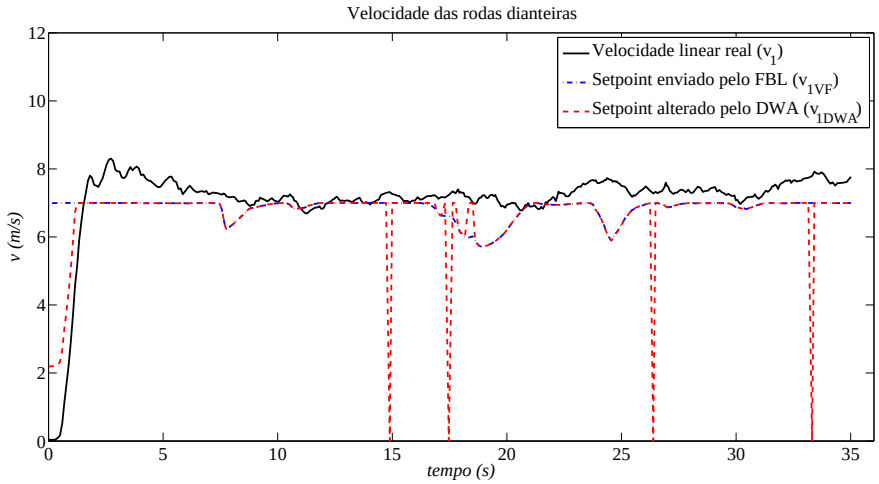
Figura 5.16: Caminho realizado pelo CADU para o mesmo campo vetorial, mas com duas poses iniciais distintas. Em (a), a pose inicial foi deslocada a esquerda em relação à mostrada na Figura 5.12. Em (b), a pose inicial foi mantida igual à definida na Figura 5.12. A pose do veículo durante a simulação está representada por retângulos vermelhos em ambos os casos.

Os comandos de velocidade de esterçamento das rodas dianteiras, assim como na simulação (Seção 5.1), foram bastante oscilatórios. Como não há realimentação da velocidade deste atuador, não é possível saber diretamente se ele está realmente aplicando a velocidade desejada pelo DWA (v_{2DWA}). A solução foi comparar o esterçamento real das rodas dianteiras com os esterçamentos estimados pelos comandos do campo vetorial e da janela dinâmica, ao longo caminho realizado. Os gráficos gerados estão na Figura 5.19, onde verifica-se que o esterçamento variou suavemente durante todo o trajeto sem e com obstáculos. Em quase todo o trajeto, o ângulo predito pelo comando de velocidade do DWA foi igual ao real. Nestes casos, o atuador foi capaz de aplicar a velocidade desejada. Apenas nos trechos onde a velocidade sofreu grandes variações ocorreram pequenos desvios entre o valor desejado e o real, mas o resultado continuou suave pela própria dinâmica do atuador.

Há, também, um outro elemento que pode ser analisado como resultado dos experimentos no CADU e que não é facilmente observado nos gráficos anteriores. Este elemento é o conforto que um passageiro sente no interior do veículo durante o movimento. Apesar de ser algo subjetivo, a ausência do conforto é um forte indicativo de que



(a)

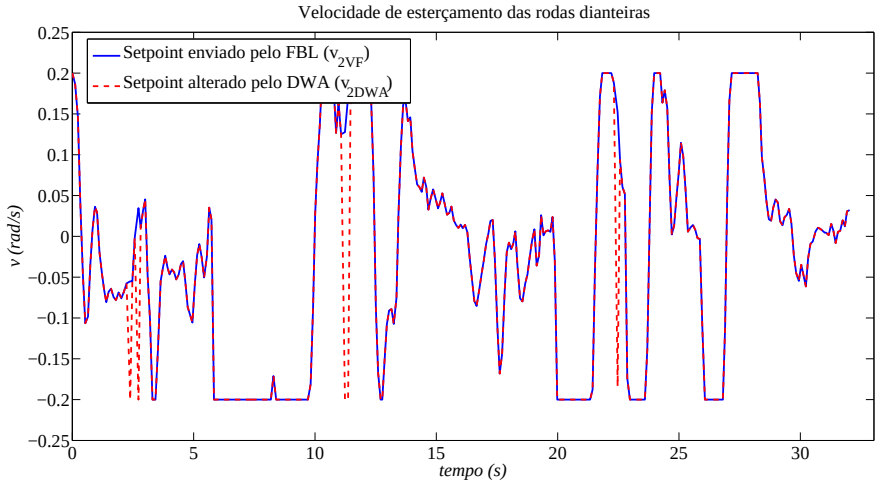


(b)

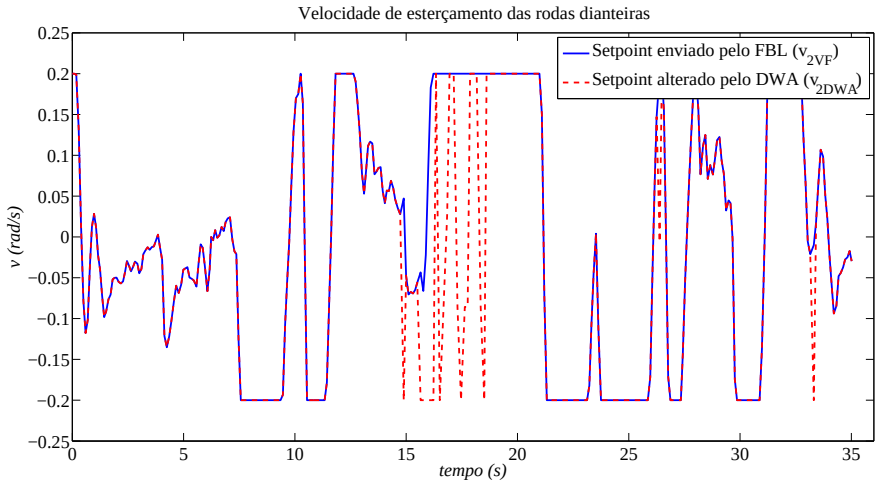
Figura 5.17: Comandos de controle da velocidade linear das rodas dianteiras (v_1), gerados pelo campo vetorial e FBL, com a validação feita pelo DWA em uma situação sem obstáculos (a) e outra com obstáculos (b). O valor real desta velocidade no CADU também pode ser observado nesta figura.

os comandos de controle não estão adequados. Movimentos oscilatórios e grandes variações na velocidade linear são alguns dos elementos que podem causar o desconforto. No entanto, apesar dos comandos de controle dos gráficos possuírem grandes variações em certos momentos, eles não foram suficientes para gerar sensações de desconforto para um passageiro dentro do veículo. Portanto, os comandos controlam o CADU com suavidade ao longo do campo vetorial e durante o desvio de obstáculos.

Para compreender melhor os resultados aqui apresentados, um video com a navegação segura no CADU está disponível no endereço `http://coro.cpdee.ufmg.br/`. Na



(a)



(b)

Figura 5.18: Comandos de controle da velocidade de esterçamento das rodas dianteiras (v_2), gerados pelo campo vetorial e FBL, com a validação feita pelo DWA em uma situação sem obstáculos (a) e outra com obstáculos (b).

próxima seção será realizada uma breve análise dos resultados obtidos em simulação e no CADU.

5.3 Discussão dos resultados

Os experimentos apresentados, tanto em simulação quanto no CADU, foram suficientes para validar a solução proposta neste trabalho e expor as suas limitações. Apesar da simulação não considerar a dinâmica do veículo, para as velocidades utilizadas os resultados mostraram-se semelhantes aos reais. Os gráficos disponibilizados ao longo

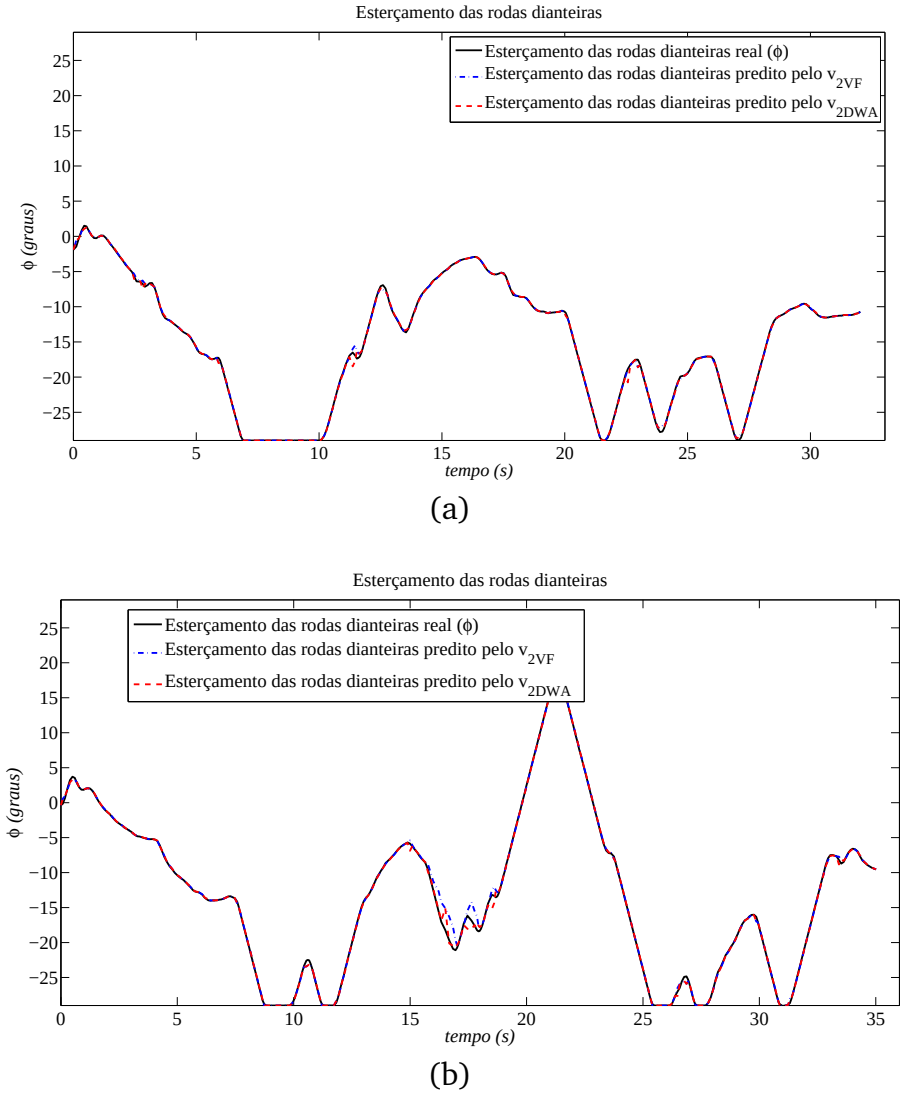


Figura 5.19: Comparação entre os os ângulos de esterçamento preditos pelos correspondentes comandos de controle de velocidade de esterçamento das rodas dianteiras (v_2) e o valor real decorrente da aplicação destas velocidades no CADU. Estes comandos foram gerados para um ambiente sem obstáculos (a) e com obstáculos (b).

deste capítulo ilustram essa afirmação. Esta condição da simulação permitiu que todos os parâmetros necessários pela metodologia deste trabalho fossem escolhidos e testados em segurança, para finalmente serem verificados no veículo real.

As limitações sensoriais da câmera estéreo, observadas na simulação, fizeram com que os obstáculos no mundo real fossem cuidadosamente colocados para permitir que o CADU pudesse enxergá-los a tempo de desviá-los. As limitações quanto à detecção de obstáculos apresentadas nos resultados de [Lima and Pereira, 2010] também são problemas que estiveram presentes nos resultados deste trabalho. Dentre eles, por exemplo, tem-se a dificuldade em se reconhecer obstáculos pequenos, como os meios-fios. Ou

seja, com apenas este sensor não foi possível garantir uma navegação segura para um carro autônomo em condições normais de operação. Outra questão sensorial foi a incorporação do laser à solução final, que somente foi possível em laboratório, pois sua instalação no CADU não foi concluída em tempo hábil para a finalização deste trabalho. No laboratório verificou-se que o mecanismo de fusão sensorial funciona adequadamente e a simulação mostrou o ganho em segurança que um sensor com um FOV maior como este pode adicionar ao veículo.

Ainda sobre a percepção dos obstáculos, o seu movimento não é considerado no DWA. Esta é uma observação importante, pois dependendo do movimento do obstáculo, uma colisão com o veículo torna-se inevitável. Porém, por ser um método dinâmico, os obstáculos que se movem com velocidades consideravelmente inferiores à do carro podem ser corretamente detectados e evitados pelo DWA.

Capítulo 6

Conclusões e Trabalhos Futuros

Neste trabalho, desenvolveu-se uma solução para a navegação segura de carros autônomos. A revisão de literatura mostrou como a navegação autônoma pode ser dividida na robótica móvel nas etapas de planejamento de movimento, detecção e desvio de obstáculos, além das etapas de localização e controle de movimento. Nela, também, foram apresentadas algumas das aplicações de navegação desenvolvidas para carros autônomos, com técnicas que criavam caminhos a serem seguidos por um controlador específico.

Diferentemente desses trabalhos, optou-se por soluções que fornecessem comandos de velocidades para realizar o planejamento de movimento e o desvio de obstáculos pelo veículo. Com esta opção, o movimento do carro pôde ser controlado apenas com os comandos fornecidos por elas, sem a necessidade de um controlador específico. No entanto, as técnicas não foram a princípio criadas para veículos autônomos e precisaram ser adaptadas para ele. A idéia principal foi realizar o planejamento de movimento por Campos Vetoriais de Velocidade e validá-lo com o Método da Janela Dinâmica (DWA) para o desvio de obstáculos não modelados. A validação fez-se necessária porque o planejamento foi realizado uma única vez para todo o movimento de um robô puntual/holonômico e, por consequência, sua aplicação direta no veículo poderia gerar colisões com os obstáculos que surgissem no ambiente. O DWA foi escolhido pela possibilidade de se trabalhar com as velocidades do veículo na criação da janela dinâmica. Como saída, ele calcula uma melhor alternativa caso a opção fornecida pelo campo vetorial não seja válida. Uma vantagem deste arranjo foi justamente o fato de campo vetorial não sofrer alterações, mesmo depois da etapa de desvio de obstáculos, pois a

completude do planejamento pôde ser sempre mantida e o resultado de navegação final considerado global.

A solução proposta, apesar de parecer simples, necessitou de vários conceitos preliminares, adquiridos na literatura, para poder garantir o controle de movimento e navegação do veículo. Estes conceitos foram então combinados na metodologia deste trabalho e resultaram em uma estrutura final, dividida nas etapas de Percepção do Ambiente e Controle de Navegação. Integrar os trabalhos relacionados à presente estrutura e corrigir suas deficiências foi o primeiro grande desafio encontrado.

O Controle de Navegação foi composto pelas etapas de localização, planejamento e controle. A localização, realizada com técnicas de fusão sensorial, foi responsável por fornecer a pose estimada do veículo no mundo para todas as etapas de planejamento. Porém, o sistema fornecia apenas poses relativas à inicial e com variações algumas vezes grandes o suficiente para prejudicar no movimento do carro. No planejamento ficaram as etapas relativas ao campo vetorial e ao DWA. Como o campo vetorial fornecia vetores de velocidades para guiar um robô holonômico e puntual no mundo, uma adaptação precisou ser aplicada aos vetores. A adaptação, composta pelo método de Linearização por Realimentação Estática (FBL), foi uma solução simples e eficaz. A saída do FBL, já no formato das entradas permitidas para o carro, foram então validadas e, eventualmente, alteradas pelo DWA.

O DWA pode ser considerada a técnica que mais sofreu alterações para ser incorporada neste trabalho. O método inicialmente desenvolvido para um robô do tipo *Synchro-Drive*, passou por duas mudanças principais:

- Adequação ao modelo cinemático do carro, com as variáveis que definem a janela dinâmica diretamente relacionadas com as entradas do modelo; e
- Criação de novas subfunções para tentar seguir os dados provenientes do campo vetorial.

Com estas mudanças, o DWA foi capaz de analisar os comandos de velocidades vindas do campo vetorial e fornecer, quando necessário, novos comandos. Os novos comandos tentariam fazer o carro seguir o sentido do campo vetorial, com velocidades lineares iguais aos módulos dos vetores do campo, e mantê-lo o mais distante possível de obstáculos ao seu redor.

Em paralelo ao controle de navegação, a Percepção do Ambiente foi desenvolvida para atualizar o mapa de obstáculos ao redor do veículo e disponibilizá-lo constantemente ao DWA. Pela quantidade de sensores utilizados nesse processo de percepção, ocorreu uma limitação sensorial que precisava ser contornada de alguma forma. Considerando apenas um laser e um sistema de visão estéreo, ambos direcionados para a região frontal do veículo, diversos segmentos ao redor do veículo ficariam sem cobertura sensorial. Além disso, a fusão das informações destes sensores seria crucial para poder aumentar a quantidade de obstáculos reconhecidos e a região coberta.

A solução adotada foi a criação de uma grade de ocupação local para armazenar um mapa probabilístico do entorno do veículo. Esta grade foi atualizada a cada nova leitura sensorial e com o movimento do veículo. A grade de ocupação local expandiu as leituras sensoriais do veículo e permitiu o reconhecimento de obstáculos em regiões antes sem informação sensorial. No entanto, duas considerações foram necessárias para garantir o seu funcionamento:

- Os obstáculos deveriam ser reconhecidos pelos sensores do carro em algum instante para estarem presentes na grade de ocupação local; e
- O movimento dos obstáculos enquanto estivessem na região coberta pela grade deveria ser considerado nulo.

A importância destas considerações para a navegação segura, como também todo o funcionamento da metodologia adotada, puderam ser observados nos resultados deste trabalho. Estes foram divididos em duas etapas, com a primeira responsável pelos testes em simulação e a segunda pelos práticos. Em simulação foram trabalhados os aspectos gerais da solução em experimentos que realizaram a configuração dos parâmetros do DWA, a análise dos comandos de controle e a verificação das limitações sensoriais. Com os resultados de configuração do DWA foi possível observar como este método funciona na presença de cada uma das suas subfunções que compõem a função objetivo. Por não ter sido priorizada uma solução ótima, os ajustes das constantes que ponderam essas subfunções aconteceu empiricamente. Mesmo assim, o resultado apresentado foi satisfatório e se mostrou capaz conduzir o carro por todo o ambiente de simulação em segurança. Os experimentos seguintes, para analisar os comandos de controle enviados

ao carro, permitiram verificar a tendência que o DWA tem de seguir sempre os dados fornecidos pelo campo vetorial. Observou-se, também, que uma janela dinâmica pequena é suficiente para validar os comandos do campo vetorial e desviar de obstáculos. Por fim, os últimos experimentos mostraram as limitações da solução com o uso dos sensores propostos, principalmente no caso da visão estéreo, a qual possui um campo de visão (FOV) reduzido. Esta limitação não permitiu a cobertura de muitas regiões do espaço pelo sensor e, consequentemente, entrou em desacordo com a primeira consideração para o funcionamento da grade de ocupação local para garantir segurança ao veículo. As simulações também permitiram que muitos dos problemas de implementação que poderiam ser encontrados na prática fossem corrigidos em segurança.

Os testes realizados em um sistema real utilizaram o Carro Autônomo Desenvolvido na Universidade Federal de Minas Gerais (CADU), com apenas um sensor de visão estéreo para detectar obstáculos. Graças aos resultados de simulação, problemas de segurança decorrentes das limitações sensoriais puderam ser evitados neste testes. Desta forma, os experimentos foram realizados respeitando estas limitações. Para estes experimentos, considerou-se também uma outra abordagem para o campo vetorial, definido pelos pontos de caminho (*waypoints*), onde o campo contínuo foi restringido para as regiões internas ao túnel que conecta dois *waypoints*. Esta adaptação aproximou a solução ao contexto dos desafios propostos pela agência americana DARPA, onde estão os principais trabalhos sobre carros autônomos da última década. Observa-se, no entanto, que a metodologia proposta é genérica o suficiente para utilizar qualquer campo vetorial.

Os resultados práticos foram similares aos obtidos em simulação e comprovaram tanto a validade da metodologia utilizada para a navegação segura quanto a validade das simulações realizadas. A grande variação dos comandos de controle, observada durante a simulação, também foi encontrada na prática. Assim, apesar de parecerem bruscos em certos momentos, os comandos acabaram suavizados pela própria dinâmica dos atuadores do veículo.

Com base nos resultados obtidos e nos trabalhos sobre navegação existentes, foi possível concluir que a navegação de um carro autônomo utilizando Campos Vetoriais e o Método da Janela Dinâmica é uma solução viável. No entanto, como o DWA utilizado não considerou obstáculos que se movem, a navegação somente pode ser garantida para

um ambiente onde os obstáculos são estáticos. Igualmente, os obstáculos ao redor do veículo precisam ser completamente percebidos por seus sensores para serem evitados. Atendendo a estas duas restrições, a navegação de um carro autônomo pelo método apresentado pode ser considerada segura.

O uso do DWA como técnica base para realizar o desvio de obstáculos mostrou a versatilidade das subfunções que o compõe e abriu a possibilidade para a criação de novas funções. A disposição do DWA permite, por exemplo, a incorporação de subfunções que considerem o movimento dos obstáculos, as leis de trânsito e outros elementos que podem ser importantes na navegação de um carro. A consideração dos *waypoints* como forma de se definir o caminho a ser realizado na parte experimental, possibilitou a ele seguir os princípios dos desafios proposto pelo DARPA, principalmente os relativos ao deserto. Estes resultados indicam que o CADU, com os recursos que dispõe mais a adição de alguns sensores de obstáculos, poderia participar de um desafio no deserto similar ao do DARPA.

No geral, a falta de mais sensores de percepção foi uma das maiores limitações encontradas neste trabalho. A navegação segura do CADU com apenas um sensor de visão estéreo não foi possível de ser garantida. Isto não somente pelo FOV do sensor, mas também pelo seu alcance limitado e dificuldade para se reconhecer obstáculos pequenos, devido à baixa resolução do mapa de disparidade obtido. Soluções para vários destes problemas sensoriais puderam ser observadas nos próprios resultados de simulação. Para o caso de se usar mais sensores, por exemplo, o efeito foi o mesmo de se aumentar o FOV, o qual permitiu reconhecer mais regiões do espaço. Para sensores com maior alcance, o veículo seria capaz de trafegar com velocidades superiores.

Outro problema encontrado, foi a descontinuidade do campo quando o veículo sai da região onde esse está definido, situação que pode ocorrer, por exemplo, quando este realiza o desvio de um obstáculo. Se o campo não existir, o veículo simplesmente para, mas se o campo for descontínuo, ele pode ficar desorientado em relação ao campo. Isto pode ser corrigido com a criação de barreiras virtuais para limitarem o veículo no campo vetorial. O resultado seria algo semelhante ao de se criar um “corredor” marcado como obstáculo na grade de ocupação local que impossibilitaria que o veículo saísse da região de onde o campo está definido. Uma outra solução seria a construção de um

campo contínuo para todo o espaço, e não somente a uma região específica, como o apresentado em [Goncalves et al., 2010].

O sistema de localização utilizado também foi uma grande fonte de erros, com medidas muitas vezes oscilantes e, devido ao fato de fornecer valores relativos, o erro inicial de posicionamento acumulava em erros maiores no decorrer dos experimentos. Estes erros, consequentemente, interferiram na qualidade da grade de ocupação local em alguns casos.

Para o futuro, portanto, são propostos os seguintes trabalhos:

- Agregação de um sensor a laser, a fim de ampliar o FOV sensorial do carro e possibilitar a detecção de pequenos obstáculos, tais como meios-fios, buracos e lombadas;
- Ampliação do número de sensores de obstáculos ao redor do veículo e seus alcances, para permitir manobras mais seguras e aumentar a velocidade máxima do CADU;
- Criação de novas subfunções para o DWA que agreguem funcionalidades que permitam ao veículo mover-se em segurança dentre obstáculos que se movem e respeitar as as leis de transito vigentes, entre outros;
- Incorporação do sistema de barreiras virtuais para garantir que o veículo não saia da região de campo vetorial e se mova com suavidade;
- Substituição do atual sistema de localização do CADU por um mais preciso, se possível baseado em um Sistema de Posicionamento Global (GPS) diferencial;

Por fim, este trabalho contribuiu para a publicação do artigo abaixo listado em congresso:

- LIMA, D. A., and PEREIRA, G. A. S. Um sistema de visão estéreo para navegação de um carro autônomo em ambientes com obstáculos. In *Anais do XVIII Congresso Brasileiro de Automática* (2010), pp. 224-231.

Referências

[Aguirre, 2007] Aguirre, L. A. (2007). *Introdução à identificação de sistemas*. Campus, 3rd edition.

[Arras et al., 2002] Arras, K., Persson, J., Tomatis, N., and Siegwart, R. (2002). Real-time obstacle avoidance for polygonal robots with a reduced dynamic window. In *Proceedings of the IEEE International Conference on Robotics and Automation*, volume 3, pages 3050–3055.

[Atreya et al., 2006] Atreya, A. R., Cattle, B. C., Collins, B. M., Essenburg, B., Franken, G. H., Saxe, A. M., Schiffres, S. N., and Kornhauser, A. L. (2006). Prospect eleven: Princeton university's entry in the 2005 DARPA Grand Challenge. *Journal of Field Robotics*, 23(9):745–753.

[Bacha et al., 2008] Bacha, A., Bauman, C., Faruque, R., Fleming, M., Terwelp, C., Reinholtz, C., Hong, D., Wicks, A., Alberi, T., Anderson, D., Cacciola, S., Currier, P., Dalton, A., Farmer, J., Hurdus, J., Kimmel, S., King, P., Taylor, A., Covern, D. V., and Webster, M. (2008). Odin: Team Victortango's entry in the DARPA Urban Challenge. *Journal of Field Robotics*, 25(8):467–492.

[Bahlmann et al., 2005] Bahlmann, C., Zhu, Y., Ramesh, V., Pellkofer, M., and Koehler, T. (2005). A system for traffic sign detection, tracking, and recognition using color, shape, and motion information. In *Proceedings of the IEEE Symposium on Intelligent Vehicles*, pages 255–260.

[Bekris, 2010] Bekris, K. E. (2010). Avoiding inevitable collision states: Safety and computational efficiency in replanning with sampling-based algorithms. In *Proceedings of the Workshop on Guaranteeing Safe Navigation in Dynamic Environments, International Conference on Robotics and Automation*, pages 1–8.

[Bemporad et al., 1996] Bemporad, A., Luca, A. D., Oriolo, G., and De, A. (1996). Local incremental planning for a car-like robot navigating among obstacles. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1205–1211.

[Berg et al., 2000] Berg, M., van Krefeld, M., Overmars, M., and Schwarzkopf, O. (2000). *Computational Geometry: Algorithms and Applications*. Springer.

[Borenstein and Koren, 1991] Borenstein, J. and Koren, Y. (1991). The vector field histogram - fast obstacle avoidance for mobile robots. *IEEE Journal of Robotics and Automation*, 7(3):278–288.

[Borges and Aldon, 2004] Borges, G. A. and Aldon, M.-J. (2004). Line extraction in 2D range images for mobile robotics. *Journal of Intelligent & Robotic Systems*, 40:267–297.

[Braid et al., 2006] Braid, D., Broggi, A., and Schmiedel, G. (2006). The TerraMax autonomous vehicle. *Journal of Field Robotics*, 23(9):693–708.

[Brock and Khatib, 1999] Brock, O. and Khatib, O. (1999). High-speed navigation using the global dynamic window approach. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 341–346.

[Broggi et al., 1999] Broggi, A., Bertozzi, A., Fascioli, A., and Guarino, C. (1999). The argo autonomous vehicles vision and control systems. *International Journal on Intelligent Control Systems*, 3(4):409–441.

[Broggi et al., 2005] Broggi, A., Caraffi, C., Fedriga, R. I., and Grisleri, P. (2005). Obstacle detection with stereo vision for off-road vehicle navigation. In *Proceedings of the International IEEE Workshop on Machine Vision for Intelligent Vehicles*, pages 1–8.

[Caraffi et al., 2007] Caraffi, C., Cattani, S., and Grisleri, P. (2007). Off-road path and obstacle detection using decision networks and stereo vision. *IEEE Transactions on Intelligent Transportation Systems*, 8(4):607–618.

[Chen et al., 2008] Chen, Y.-L., Sundareswaran, V., Anderson, C., Broggi, A., Grisleri, P., Porta, P. P., Zani, P., and Beck, J. (2008). TerraMax™: Team Oshkosh urban robot. *Journal of Field Robotics*, 25(10):841–860.

[Choset et al., 2005] Choset, H., Lynch, K. M., Hutchinson, S., Kantor, G. A., Burgard, W., Kavraki, L. E., and Thrun, S. (2005). *Principles of Robot Motion: Theory, Algorithms, and Implementations*. MIT Press, Cambridge, MA.

[Dickmanns and Zapp, 1987] Dickmanns, E. D. and Zapp, A. (1987). Autonomous high speed road vehicle guidance by computer vision. In *10th* IFAC *World Congress Munich*, pages 232–237.

[Duda and Hart, 1972] Duda, R. O. and Hart, P. E. (1972). Use of the hough transformation to detect lines and curves in pictures. *Communications of the* ACM, 15(1):11–15.

[Egerstedt et al., 1998] Egerstedt, M., Hu, X., and Stotsky, A. (1998). Control of a car-like robot using a dynamic model. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3273–3278.

[Elfes, 1989] Elfes, A. (1989). Using occupancy grids for mobile robot perception and navigation. *Computer*, 22(6):46–57.

[Faugeras, 1993] Faugeras, O. (1993). *Three-dimensional computer vision: A geometric view point*. MIT Press, Cambridge.

[Ferreira et al., 2004] Ferreira, A., Filho, M. S., and Filho, T. F. B. (2004). Desvio tangencial de obstáculos para um robô móvel navegando em ambientes semi-estruturados. In *Anais do XV Congresso Brasileiro de Automática*, pages 1–6.

[Fiorini and Shillert, 1998] Fiorini, P. and Shillert, Z. (1998). Motion planning in dynamic environments using velocity obstacles. *International Journal of Robotics Research*, 17:760–772.

[Fox et al., 1997] Fox, D., Burgard, W., and Thrun, S. (1997). The dynamic window approach to collision avoidance. *IEEE Robotics and Automation Magazine*, 4:23–33.

[Frazzoli et al., 2000] Frazzoli, E., Dahleh, M. A., and Feron, E. (2000). Real-time motion planning for agile autonomous vehicles. *Journal of Guidance, Control, and Dynamics*, 25(1):116–129.

[Freitas and Pereira, 2010] Freitas, E. J. R. and Pereira, G. A. S. (2010). *Controle longitudinal de um veículo autônomo*. Trabalho de Conclusão de Curso, Escola de Engenharia da Universidade Federal de Minas Gerais.

[Freitas et al., 2009] Freitas, E. J. R., Vinti, M. N. W., Santos, M. M., Iscold, P., Tôrres, L. A. B., and Pereira, G. A. S. (2009). Desenvolvimento de automação embarcada para um robô móvel baseado em um carro de passeio. *Simpósio Brasileiro de Automação Inteligente*, 9:1–6.

[Gonçalves et al., 2010] Gonçalves, L. M., Almeida, R. M. A., and Honório, L. M. (2010). Modelagem de busca de rotas para a navegação local de um veículo autônomo inteligente. In *Anais do XVIII Congresso Brasileiro de Automática*, pages 2073–2079.

[Goncalves et al., 2010] Goncalves, V. M., Pimenta, L. C. A., Maia, C. A., Dutra, B. C. O., and Pereira, G. A. S. (2010). Vector fields for robot navigation along time-varying curves in n-dimensions. *IEEE Transactions on Robotics*, 26(4):647–659.

[Halterman and Bruch, 2010] Halterman, R. and Bruch, M. (2010). Velodyne HDL-64E lidar for unmanned surface vehicle obstacle detection. In *Proceedings of the SPIE: Unmanned Systems Technology XII*, pages 76920D–76920D–8.

[Hiroshi et al., 1995] Hiroshi, T. K., Kano, H., Kimura, S., Yoshida, A., and Oda, K. (1995). Development of a video-rate stereo machine. In *IROS '95: Proceedings of the International Conference on Intelligent Robots and Systems-Volume 3*, page 3095, Washington, DC, USA. IEEE Computer Society.

[Jamasmie, 2009] Jamasmie, C. (2009). Lonely trucks in a lonely place: Autonomous trucks debut in Chile's desert. MINING.COM *a mine of information*, 2:23–24.

[Kalman, 1960] Kalman, R. E. (1960). A new approach to linear filtering and prediction problems. *Transactions of the ASME, Journal of Basic Engineering*, 82:35–45.

[Ko and Simmons, 1998] Ko, N. Y. and Simmons, R. (1998). The lane-curvature method for local obstacle avoidance. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1615–1621.

[Kolski and Macek, 2007] Kolski, S. and Macek, K. (2007). Path planning, replanning, and execution for autonomous driving in urban and offroad environments. In *In Proceedings of the Workshop on Planning, Perception and Navigation for Intelligent Vehicles, International Conference on Robotics and Automation*, pages 1–6.

[Krogh and Thorpe, 1986] Krogh, B. and Thorpe, C. (1986). Integrated path planning and dynamic steering control for autonomous vehicles. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1664–1669.

[Labayrade et al., 2002] Labayrade, R., Aubert, D., and Tarel, J. P. (2002). Real time obstacle detection in stereovision on non flat road geometry through "v-disparity" representation. In *Proceedings of the IEEE Symposium on Intelligent Vehicles*, volume 2, pages 646–651.

[Lamiraux and Laumond, 2001] Lamiraux, F. and Laumond, J. P. (2001). Smooth motion planning for car-like vehicles. *IEEE Transactions on Robotics and Automation*, 17:498–502.

[Laumond et al., 1998] Laumond, J. P., Sekhavat, S., and Lamiraux, F. (1998). *Robot Motion Planning and Control*, volume 229, chapter Guidelines in Nonholonomic Motion Planning for Mobile Robots, pages 1–44. Springer Berlin / Heidelberg.

[Lee and Lee, 2004] Lee, K. and Lee, J. (2004). Generic obstacle detection on roads by dynamic programming for remapped stereo images to an overhead view. In *Proceedings of the IEEE International Conference on Networking, Sensing and Control*, volume 2, pages 897–902.

[Lefebvre et al., 2004] Lefebvre, O., Lamiraux, F., and Pradalier, C. (2004). Obstacles avoidance for car-like robots. integration and experimentation on two robots. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 4277–4282.

[Leonard et al., 2008] Leonard, J., How, J., Teller, S., Berger, M., Campbell, S., Fiore, G., Fletcher, L., Frazzoli, E., Huang, A., Karaman, S., Koch, O., Kuwata, Y., Moore, D., Olson, E., Peters, S., Teo, J., Truax, R., Walter, M., Barrett, D., Epstein, A., Maheloni, K., Moyer, K., Jones, T., Buckley, R., Antone, M., Galejs, R., Krishnamurthy, S., and Williams, J. (2008). A perception-driven autonomous urban vehicle. *Journal of Field Robotics*, 25(10):727–774.

[Lima and Pereira, 2010] Lima, D. A. and Pereira, G. A. S. (2010). Um sistema de visão estéreo para navegação de um carro autônomo em ambientes com obstáculos. In *Anais do XVIII Congresso Brasileiro de Automática*, pages 224–231.

[Luca et al., 1998] Luca, A. D., Oriolo, G., De, A., and Samson, C. (1998). *Robot Motion Planning and Control*, volume 229, chapter Feedback Control Of A Nonholonomic Car-Like Robot, pages 171–253. Springer Berlin / Heidelberg.

[Minguez et al., 2008] Minguez, J., Lamiraux, F., and Laumond, J.-P. (2008). Motion planning and obstacle avoidance. In an Oussama Khatib, B. S., editor, *Handbook of Robotics*. Springer, first edition.

[Minguez et al., 2004] Minguez, J., Member, A., and Montano, L. (2004). Nearness diagram (ND) navigation: Collision avoidance in troublesome scenarios. *IEEE Transactions on Robotics and Automation*, 20:2004.

[Minguez and Montano, 2000] Minguez, J. and Montano, L. (2000). Nearness diagram navigation (ND): A new real time collision avoidance approach. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2094–2100.

[Montemerlo et al., 2008] Montemerlo, M., Becker, J., Bhat, S., Dahlkamp, H., Dolgov, D., Ettinger, S., Haehnel, D., Hilden, T., Hoffmann, G., Huhnke, B., Johnston, D., Klumpp, S., Langer, D., Levandowski, A., Levinson, J., Marcil, J., Orenstein, D., Paefgen, J., Penny, I., Petrovskaya, A., Pflueger, M., Stanek, G., Stavens, D., Vogt, A., and Thrun, S. (2008). Junior: The stanford entry in the urban challenge. *Journal of Field Robotics*, 25(9):569–597.

[Moreira et al., 2007] Moreira, M., Machado, H., Mendonca, C., and Pereira, G. (2007). Mobile robot outdoor localization using planar beacons and visual improved odometry. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2468–2473.

[Newman et al., 2009] Newman, P., Sibley, G., Smith, M., Cummins, M., Harrison, A., Mei, C., Posner, I., Shade, R., Schroeter, D., Murphy, L., Churchill, W., Cole, D., and Reid, I. (2009). Navigating, recognizing and describing urban spaces with vision and lasers. *The International Journal of Robotics Research*, 28(11-12):1406–1433.

[Park et al., 2008] Park, W.-J., Kim, B.-S., Seo, D.-E., Kim, D.-S., and Lee, K.-H. (2008). Parking space detection using ultrasonic sensor in parking assistance system. In *Proceedings of the IEEE Symposium on Intelligent Vehicles*, pages 1039–1044.

[Patz et al., 2008] Patz, B. J., Papelis, Y., Pillat, R., Stein, G., and Harper, D. (2008). A practical approach to robotic design for the DARPA Urban Challenge. *Journal of Field Robotics*, 25(8):528–566.

[Pereira, 2006] Pereira, F. G. (2006). Navegação e desvio de obstáculos usando um robô móvel dotado de sensor de varredura laser. Master's thesis, Centro Tecnológico da Universidade Federal do Espírito Santo.

[Pereira et al., 2009] Pereira, G. A. S., Pimenta, L. C. A., Chaimowicz, L., Fonseca, A. R., de Almeida, D. S. C., de Q. Corrêa, L., Mesquita, R. C., and Campos, M. F. M. (2009). Robot navigation in multi-terrain outdoor environments. *The International Journal of Robotics Research*, 28(6):685–700.

[Pereira et al., 2008] Pereira, G. A. S., R.Rebelo, D., Iscold, P., and Torres, L. A. B. (2008). A vector field approach to guide small UAVs through a sequence of waypoints. In *Anais do XVII Congresso Brasileiro de Automática*, pages 1–6.

[Perrollaz et al., 2006] Perrollaz, M., Labayrade, R., Royère, C., Hautière, N., and Aubert, D. (2006). Long range obstacle detection using laser scanner and stereovision. In *Proceedings of the IEEE Symposium Intelligent Vehicles*, pages 182–187.

[Pimenta et al., 2006] Pimenta, L., Fonseca, A., Pereira, G., Mesquita, R., Silva, E., Caminhas, W., and Campos, M. (2006). Robot navigation based on electrostatic field computation. *IEEE Transactions on Magnetics*, 42(4):1459–1462.

[Pimenta et al., 2007] Pimenta, L., Pereira, G., and Mesquita, R. (2007). Fully continuous vector fields for mobile robot navigation on sequences of discrete triangular regions. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1992–1997.

[Rauskolb et al., 2008] Rauskolb, F. W., Berger, K., Lipski, C., Magnor, M., Cornelsen, K., Effertz, J., Form, T., Graefe, F., Ohl, S., Schumacher, W., Wille, J.-M., Hecker, P., Nothdurft, T., Doering, M., Homeier, K., Morgenroth, J., Wolf, L., Basarke, C., Berger, C., Gülke, T., Klose, F., and Rumpe, B. (2008). Caroline: An autonomously driving vehicle for urban environments. *Journal Field Robotics*, 25(9):674–724.

[Rebai and Azouaoui, 2009] Rebai, K. and Azouaoui, O. (2009). Bi-steerable robot navigation using a modified dynamic window approach. In *Proceedings of the 6th International Symposium on Mechatronics and its Applications*, pages 1–6.

[Rebai et al., 2007] Rebai, K., Azouaoui, O., Benmami, M., and Larabi, A. (2007). Car-like robot navigation at high speed. In *Proceedings of the IEEE International Conference on Robotics and Biomimetics*, pages 2053–2057.

[Rimon and Koditschek, 1992] Rimon, E. and Koditschek, D. (1992). Exact robot navigation using artificial potential functions. *IEEE Transactions on Robotics and Automation*, 8(5):501–518.

[Sabbagh et al., 2010] Sabbagh, V. B., Freitas, E. J. R., Castro, G. M. M., Santos, M. M., Baleeiro, M. F., Silva, T. M., Iscold, P., Tôrres, L. A. B., and Pereira, G. A. S. (2010). Desenvolvimento de um sistema de controle para um carro de passeio autônomo. In *Anais do XVIII Congresso Brasileiro de Automática*, pages 928–933.

[Santos, 2009] Santos, M. M. (2009). Desenvolvimento de um sistema de localização e reconstrução de trajetórias para um veículo terrestre. Master's thesis, Escola de Engenharia da Universidade Federal de Minas Gerais.

[Seder and Petrovic, 2007] Seder, M. and Petrovic, I. (2007). Dynamic window based approach to mobile robot motion control in the presence of moving obstacles. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1986–1991.

[Shiller et al., 2010] Shiller, Z., Gal, O., and Fraichard, T. (2010). The nonlinearvelocity obstacle revisited: The optimal time horizon. In *Proceedings of the Workshop on Guaranteeing Safe Navigation in Dynamic Environments, International Conference on Robotics and Automation*, pages 1–5.

[Simmons, 1996] Simmons, R. (1996). The curvature-velocity method for local obstacle avoidance. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3375–3382.

[Skiena, 2004] Skiena, S. S. (2004). Geometric reconstruction problems. In Goodman, J. E. and O'Rourke, J., editors, *Handbook of Discrete and Computational Geometry*. Chapman & HALL/CRC, second edition.

[Soquet et al., 2007] Soquet, N., Aubert, D., and Hautiere, N. (2007). Road segmentation supervised by an extended v-disparity algorithm for autonomous navigation. In *Proceedings of the IEEE Symposium on Intelligent Vehicles*, pages 160–165.

[Souza, 2008] Souza, A. A. S. (2008). Mapeamento com sonar usando grade de ocupação baseado em modelagem probabilística. Master's thesis, Centro de Tecnologia da Universidade Federal do Rio Grande do Norte.

[Svestka et al., 1995] Svestka, P., Overmars, M. H., Overmars, M. H., and Overmars, M. H. (1995). Motion planning for car-like robots using a probabilistic learning approach. *International Journal of Robotics Research*, 16.

[CARVALHO JUNIOR, 2007] CARVALHO JUNIOR, H. H. (2007). Métodos inteligentes de navegação e desvio de obstáculos. Master's thesis, Universidade Federal de Itajubá.

[DARPA, 2005] DARPA (2005). DARPA Grand Challenge. Disponível em: `http://www.darpa.mil/grandchallenge05/`. Acesso em: 16 Jan. 2010.

[DARPA, 2008] DARPA (2008). DARPA Urban Challenge. Disponível em: `http://www.darpa.mil/grandchallenge/index.asp`. Acesso em: 16 Jan. 2010.

[KUFFNER JR. and Lavalle, 2000] KUFFNER JR., J. J. and Lavalle, S. M. (2000). RRT-connect: An efficient approach to single-query path planning. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 995–1001.

[Thrun et al., 2005] Thrun, S., Burgard, W., and Fox, D. (2005). *Probabilistic Robotics*. The MIT Press.

[Thrun et al., 2006] Thrun, S., Montemerlo, M., Dahlkamp, H., Stavens, D., Aron, A., Diebel, J., Fong, P., Gale, J., Halpenny, M., Hoffmann, G., Lau, K., Oakley, C., Palatucci, M., Pratt, V., Stang, P., Strohband, S., Dupont, C., Jendrossek, L.-E., Koelen, C., Markey, C., Rummel, C., van Niekerk, J., Jensen, E., Alessandrini, P., Bradski, G., Davies, B., Ettinger, S., Kaehler, A., Nefian, A., and Mahoney, P. (2006). Stanley: The robot that won the DARPA Grand Challenge. *Journal of Field Robotics*, 23(9):661–692.

[Ulrich and Borenstein, 1998] Ulrich, I. and Borenstein, J. (1998). VFH+: Reliable obstacle avoidance for fast mobile robots. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1572–1577.

[Ulrich and Borenstein, 2000] Ulrich, I. and Borenstein, J. (2000). VFH*: Local obstacle avoidance with look-ahead verification. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 2505–2511.

[Urmson et al., 2008] Urmson, C., Anhalt, J., Bagnell, D., Baker, C., Bittner, R., Clark, M. N., Dolan, J., Duggins, D., Galatali, T., Geyer, C., Gittleman, M., Harbaugh, S., Hebert, M., Howard, T. M., Kolski, S., Kelly, A., Likhachev, M., McNaughton, M., Miller, N., Peterson, K., Pilnick, B., Rajkumar, R., Rybski, P., Salesky, B., Seo, Y.-W., Singh, S., Snider, J., Stentz, A., Whittaker, W. R., Wolkowicki, Z., Ziglar, J., Bae, H., Brown, T., Demitrish, D., Litkouhi, B., Nickolaou, J., Sadekar, V., Zhang, W., Struble, J., Taylor, M., Darms, M., and Ferguson, D. (2008). Autonomous driving in urban

environments: Boss and the urban challenge. *Journal of Field Robotics*, 25(8):425–466.

[Wijesoma et al., 2001] Wijesoma, W. S., Kodagoda, K. R. S., Balasuriya, A. P., and Teoh, E. K. (2001). Road edge and lane boundary detection using laser and vision. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 3, pages 1440–1445.

Apêndice A

Complexidade Computacional

A implementação computacional da solução proposta para a navegação segura de um carro autônomo, tanto para os experimentos de simulação quanto para os práticos, seguiram passos semelhantes. Exceto pelas partes de simulação, todas as demais etapas descritas na Seção 5.1 são comuns entre os experimentos. No entanto, para melhorar o desempenho do programa final no CADU, algumas destas etapas foram separadas entre 4 programas para serem executados em máquinas distintas. A análise aqui apresentada será apresentada para cada um destes programas.

A primeira etapa da solução foi criar o campo vetorial de velocidades, implementada no programa *CarWaypoints*. Sabendo-se que no túnel definido entre dois 2 *waypoints* são gerados no máximo 2 triângulos, então para n_w *waypoints* o número de triângulos gerados será da ordem de $O(n_w)$. A complexidade para a criação do campo vetorial é dada por $O(n_w \log n_w)$ [Pereira et al., 2009]. Esta etapa é realizada uma única vez, razão pela qual foi separada dos demais passos. O próximos passos são realizados em sequência durante cada ciclo do programa *CarNavigationControl* e, por isso, a complexidade final dele estará relacionada a cada função deste ciclo, sempre considerando o pior caso no qual todas as funções são chamadas.

O ciclo se inicia com o recebimento de um novo dado sensorial de obstáculos e sua inclusão na grade de ocupação local. Seja n_l o número de leituras do sensor e n_{occ} o número de linhas e colunas da grade de ocupação local. Para se incluir novos dados na grade de ocupação local, por meio do Algoritmo 4.1 *occupancy_grid_mapping*, é necessário percorrer toda a grade. O número de operações necessárias neste processo é da ordem de $O(n_{occ}^2)$, o qual independe de n_l, já que a subfunção *inverse_range_sensor_model* pode ser executada em tempo constante ($O(1)$).

A próxima etapa é responsável pela leitura sensorial na grade de ocupação local. Sua implementação, por meio de coordenadas polares, fez uma varredura em toda a extensão da grade. Por percorrer novamente toda a extensão da grade de ocupação local, esta etapa é da mesma ordem da anterior, ou seja, $O(n_{occ}^2)$.

Em seguida é calculado o vetor de velocidade para a posição atual do veículo. Processo realizado com uma busca em todos os triângulos do campo vetorial, no pior caso. Como a quantidade de triângulos gerados é da ordem de $O(n_w)$, a busca e o cálculo do vetor também terá a mesma ordem [Pereira et al., 2009]. No entanto, no melhor caso, que também é o caso mais frequente, a busca pelo triângulo atual é apenas de ordem $O(1)$. Isto ocorre devido a dinâmica do veículo que garante que a busca pelo triângulo atual será realizada somente no triângulo do instante anterior e em seus vizinhos. A aplicação da linearização por realimentação estática a este vetor por ser realizada em tempo constante.

O Método da Janela Dinâmica (DWA), dependendo da dimensão da janela dinâmica utilizada, pode resultar em uma etapa de alto custo computacional. Sejam $n_{v_1} \times n_\phi$ a dimensão da janela dinâmica. Para compor as subfunções do DWA é necessário percorrer toda a extensão da janela. A cada chamada da subfunção *vf1*, um novo vetor no campo vetorial é calculado na ordem de $O(n_w)$. Para todas as posições da janela dinâmica o tempo é da ordem de $O(n_{v_1} n_\phi n_w)$. A mesma análise é feita para a subfunção *dist*, onde, para cada posição da janela, todo o vetor de dados de obstáculos lido da grade de ocupação local é percorrido. A quantidade de dados lidos da grade é dada por $n_{l_{occ}}$ e implica em uma complexidade final para a subfunção *dist* de $O(n_{v_1} n_\phi n_{l_{occ}})$. A última subfunção possui tempo constante para executar, portanto, seu tempo é da ordem de $O(n_{v_1} n_\phi)$. Se a janela for pouco discretizada, tal que o produto $n_{v_1} \cdot n_\phi$ seja muito menor que n_w e $n_{l_{occ}}$, a complexidade do DWA pode ser aproximada por $O(n_w + n_{l_{occ}})$ que é de primeira ordem.

A última etapa é a atualização da grade de ocupação local com o movimento do veículo. Ela é feita a partir de operações matriciais de transformação para toda a dimensão da grade. Assim como as demais operações da grade, o tempo desta operação é limitado por $O(n_{occ}^2)$. Portanto, o tempo final, dominante para o funcionamento do programa *CarNavigationControl*, é da ordem de $O(n_{occ}^2)$.

Os dois programas restantes, *CarCameraSensor* e *CarLaserSensor*, são responsáveis pela aquisição dos dados da câmera de visão estéreo e do sensor a laser, detecção dos possíveis obstáculos e fornecimento dos resultados para o programa *CarNavigationControl*. Como o foco deste trabalho não é a detecção de obstáculos, será fornecida apenas o valor encontrado para a complexidade em ambos os casos. Para *CarCameraSensor*, a complexidade é $O(n_i^3)$, onde n_i é o maior lado da imagem. O tempo de execução é ponderado pela etapa de criação do mapa de disparidade. Neste caso a imagem adquirida pela câmera da esquerda é inteiramente percorrida e, para cada posição, toda a linha da imagem da direita é analisada a procura de pontos correspondentes. Já o *CarLaserSensor* é mais simples, pois não realiza nenhum processamento adicional nos dados sensoriais adquiridos. Sua complexidade, portanto, é linear, diretamente relacionada com a quantidade de dados adquiridos pelo laser.