DETECÇÃO DE MUDANÇAS E RECUPERAÇÃO DE FORMA EM MAPAS 3D BASEADOS EM NUVENS DE PONTOS

PAULO LILLES JORGE DREWS JR

DETECÇÃO DE MUDANÇAS E RECUPERAÇÃO DE FORMA EM MAPAS 3D BASEADOS EM NUVENS DE PONTOS

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação do Instituto de Ciências Exatas da Universidade Federal de Minas Gerais como requisito parcial para a obtenção do grau de Mestre em Ciência da Computação.

ORIENTADOR: PROF. MARIO FERNANDO MONTENEGRO CAMPOS

Belo Horizonte

Dezembro de 2009

Drews Jr, Paulo Lilles Jorge
D776d Detecção de Mudanças e Recuperação de Forma em Mapas 3D Baseados em Nuvens de Pontos / Paulo Lilles Jorge Drews Jr. — Belo Horizonte, 2009
xxx, 130 f. : il. ; 29cm

Dissertação (mestrado) — Universidade Federal de Minas Gerais
Orientador: Prof. Mario Fernando Montenegro Campos

1. Robótica - Teses. 2. Imagem tridimensional - Teses. I. Título.

CDU 519.6*82.10(043)

[Folha de Aprovação]

Quando a secretaria do Curso fornecer esta folha,
ela deve ser digitalizada e armazenada no disco em formato gráfico.

Se você estiver usando o `pdflatex`,
armazene o arquivo preferencialmente em formato PNG
(o formato JPEG é pior neste caso).

Se você estiver usando o `latex` (não o `pdflatex`),
terá que converter o arquivo gráfico para o formato EPS.

Em seguida, acrescente a opção `approval={`*nome do arquivo*`}`
ao comando `\ppgccufmg`.

Agradecimentos

Minha trajetória até chegar a este ponto, finalizando o mestrado na Universidade Federal de Minas Gerais, passou por diversos passos. Sei que não consigo agradecer a todos, mas espero não esquecer de ninguém.

Gostaria de agradecer, primeiramente, à minha mãe, Maria da Graça, pelo apoio incondicional dado a todas as loucuras que já fiz, e ainda vou fazer, principalmente quando decidi largar o sul do Brasil e correr atrás de um sonho em Minas Gerais. Ao meu pai, Paulo, e meu irmão, Mauro, pelas palavras de incentivo. À Profa. Silvia Botelho, por ter sempre me apoiado muito e me mostrado as minhas capacidades, das quais sempre duvidei. Ao meu "segundo" irmão Eng. Fábio Pedrotti, por ter sempre acreditado em mim, além de ter me dado a infra-estrutura quando cheguei a Belo Horizonte. Também aos "irmãos de BH", Eng. Rodrigo Morasco e Eng. Marcelo Motta, pelas várias horas de alegria que me proporcionaram nesse lugar tão distante. Também, aos amigos João e Alexandre, por terem dividido mais que uma casa comigo. Pelas conversas, pelas aulas de violão, canto, italiano e, em especial, de vida e vivência. À Profa. Priscila Martins por ter me ajudado muito na proposta de dissertação. Não posso esquecer, também, de meus ex-colegas de Eng. de Computação da FURG, por terem sempre me dado aquela recepção em Porto Alegre nas minhas poucas idas ao Sul e compartilhado comigo minhas dúvidas e frustrações em relação ao futuro sempre incerto: Tomás, Daniel, Ulisses, Juliano e Igor.

Queria agradecer, especialmente, ao Prof. Mário Campos, pois se não fosse aquele convite no SBAI de 2007, com certeza não teria vindo para Belo Horizonte. Por todo incansável apoio e ensinamentos passados durante essa longa jornada e, claro, pelo apoio na minha idéia maluca de sair do país durante o mestrado. Às suas sempre sábias palavras, muito obrigado!

Agradeço a todos os professores que tive durante a vida, pois cada um teve uma parcela importante na minha formação, como engenheiro, pesquisador e como cidadão. Recentemente, aos professores dos quais fui aluno na UFMG, Prof. Luiz Chaimowicz, Prof. Antônio Loureiro e Prof. Geraldo Robson, pelos conhecimentos importantes que

me passaram. Aos professores que participaram da banca de defesa desta dissertação, os quais agregaram valiosas sugestões a este trabalho. Ao Prof. Hugo Vieira Neto, especialmente, pela revisão criteriosa do texto.

Agradeço, também, a todos os colegas do VeRLab que dividiram comigo algumas horas de lazer e trabalho, com discussões técnicas ou não, mas que foram muito importante para que eu chegasse ao fim dessa etapa. Da Iniciação Científica: Soriano, Costa, Balbino, Renato, Dimas, Bruno e Allyson. Do mestrado (muitos hoje em dia já no doutorado): Armando, Douglas, Erickson, Renato, Rafael, Thiago e Cantoni. Também aos doutorandos: Vilar, Pedro, Marcelo, Sandro, Wagner e Celso. E, claro, ao nosso colaborador Wolmar.

Gostaria de agradecer, também, a oportunidade que me foi dada pelo Prof. Jorge Dias de pesquisar em Portugal, por quase 8 meses, em um laboratório de pesquisa exemplar com pessoas fantásticas. Entre essas pessoas incluo, especialmente, o Prof. Rui Rocha, por ter acreditado em mim, mesmo quando parecia que não ia dar conta, pelos ensinamentos e empenho constante para me ajudar. Aos colegas de MRL e ISR, de todas as partes do mundo, são tantas coisas a agradecer, mas cada um sabe a importância que teve nesse período importante da minha vida. Aos colegas brasileiros: Diego, José, Luis Mirisola, Oswaldo, Cristiano. Aos colegas africanos: Abed e Omar. Aos colegas iranianos: Hadi e Kamrad. E aos diversos colegas portugueses: Nuno, João Machado, João Monteiro, Luís Santos, Carlos, Alexandre, Ricardo, Paula e Rita.

Me gustaria de agradecer en español, aunque nosotros siempre hablamos en inglés, a una persona fenomenal que conoci en Portugal, el Prof. Pedro Nuñez. Por todo que me hay ayudado, trabajando conjuntamente, siempre con nuevas ideas y palabras de apoyo. Por creer en mi y me ayudar asta por el Skype. Gracias "cabrón".

Agradeço, também, às pessoas que me ajudaram a transformar meus dias em Portugal, em dias bem melhores, dividindo comigo um lar: Fernando, Gabriela, José e, especialmente, Isabela, por ser uma namorada fantástica, além uma excelente corretora ortográfica. Por ter me ajudado e apoiado incansavelmente, quando eu pensava em desistir de tudo, além da paciência - e que paciência! - acho que nem ela sabia que tinha tanta.

Agradeço, ainda, a CAPES pelo apoio financeiro nesses quase 2 anos de trabalho. Também ao Departamento de Ciência da Computação da UFMG e todos os seus funcionários e colaboradores.

A todos amigos e colegas, citados ou não, muito obrigado!

"Para obter algo que você nunca teve, é preciso fazer algo que nunca fez"
(Autor Desconhecido)

Resumo

O presente trabalho propõe um arcabouço para determinação e segmentação de mudanças em nuvens de pontos 3D de forma eficiente. Inicialmente, é efetuada uma simplificação nesses pontos, gerando uma representação em árvore que busca preservar as características relevantes da nuvem por meio da estimação da variação da superfície. Além disso, é utilizada uma combinação de Modelos de Misturas Gaussianas (*GMM*) e da métrica de distância *Earth Mover's Distance* (*EMD*), por meio de uma técnica gulosa. Ela permite detectar o que é mudança em um dado local, a partir de leituras prévias do sensor, e segmentar as mudanças detectadas.

Para recuperação de forma 3D são propostos dois métodos. Primeiramente, um novo método para recuperação de formas básicas é desenvolvido, no espaço das Gaussianas. Esse método é limitado a um conjunto de primitivas geométricas que incluem planos, esferas e cilindros. Esse método é comparado, ao longo do trabalho, com uma abordagem no espaço Euclidiano utilizando *Random Sampling Consensus* (*RANSAC*).

Uma segunda abordagem utiliza superquádricas, visando dar uma maior expressividade à representação de forma 3D. Este trabalho utiliza uma abordagem multi-escala com o paradigma *split-and-merge*, que busca ultrapassar mínimos locais da segmentação fornecida pela *GMM*, bem como dar maior eficiência ao método como um todo.

Resultados experimentais utilizando dados simulados e reais foram obtidos para avaliar a eficácia do método de simplificação utilizado, assim como da abordagem de detecção de mudança. O método para recuperação de formas básicas mostrou-se melhor que o método utilizando o espaço Euclidiano, em termos de custo computacional e exatidão, porém, apresenta algumas restrições que são contornadas pelo uso de superquádricas.

Palavras-chave: Detecção de Mudanças, Mapas 3D, Recuperação de Forma, Robótica Móvel.

Abstract

This work proposes a framework to efficiently detect and segment changes in 3D point clouds. Initially, we simplify the point cloud which generates a tree representation in order to preserve the relevant characteristics using surface variation as the key constraint. Then, a combined approach using *Gaussian Mixture Models* (GMM) and the *Earth Mover's Distance* (EMD) is proposed, together with a greedy technique. It detects what has changed in a given locale from a previous sensor reading and segments the detected novelty.

The 3D shape retrieval is achieved using two different approaches. The first approach is a new method that we developed which directly retrieves the detected shape in the Gaussian space. It uses a limited set or geometric primitives including planes, spheres and cylinders. This method is compared with another one that operates in the Euclidean space using *Random Sampling Consensus* (RANSAC).

The second approach uses superquadric models due to their good expressiveness in representing more generalized three dimensional shapes. In order to achieve a good level of efficiency and to overcome the limitations of *GMM* segmentation, a multi-scale approach using the *split-and-merge* paradigm was devised and successfully implemented.

Experimental results in both real and simulated scenarios were obtained to evaluate the efficacy of the simplification and change detection methods. The shape retrieval method in the Gaussian space showed better results than the Euclidean space method, both in accuracy and computational cost. The limitations on shape representation were adequately overcome by the use of superquadrics.

Keywords: Change Detection, 3D Maps, Shape Retrieval, Mobile Robotics.

Lista de Figuras

Lista de Tabelas

Sumário

Lista de Algoritmos

Lista de Abreviações

3D-NDT	*3D Normal Distribution Transform*
CAD	*Computer-Aided Design*
Carmen	*Carnegie Mellon Robot Navigation Toolkit*
DFT	*Discrete Fourier Transform*
DSO	*Discrete Shape Operator*
EKF	*Extended Kalman Filter*
EM	*Expectation-Maximization*
EMD	*Earth Mover's Distance*
FastFPS	*Fast Marching Farthest Point Sampling*
GMM	*Gaussian Mixture Model*
GPS	*Global Positioning System*
GPU	*Graphics Processing Unit*
GWR	*Grow When Required*
HSOM	*Habituating Self-Organizing Maps*
ICP	*Iterative Closest Point*
IITD	*Integration of Intensity and Texture Differences*
IPC	*Inter-Process-Communication*
IRPS	*Intelligent Robotic Porter System*
LAD	*Local Average of Differences*
LRF	*Laser Ranger Finder*
LSAD	*Local Sum of Absolute Differences*
LSSD	*Local Sum of Squared Differences*
LVQ	*Learning Vector Quantization*
MEMS	*Micro-Electro-Mechanical Systems*
ML	*Maximum Likelihood*
MDL	*Minimum Description Length*
MLS	*Moving Least Squares*
MST	*Minimum Spanning Tree*

OBB	*Oriented Boundary Box*
PCA	*Principal Component Analysis*
PDF	*Probability Density Function*
RANSAC	*Random Sample Consensus*
RMS	*Root Mean Square*
SIFT	*Scale-Invariant Feature Transform*
SLAM	*Simultaneous Localization and Mapping*
SOM	*Self-Organizing Map*
SQ	Superquádricas
TKM	*Temporal Kohonen Map*
UFMG	Universidade Federal de Minas Gerais
VeRLab	Laboratório de Visão Computacional e Robótica

Lista de Símbolos

α	Constante utilizada no processo de remoção de *outliers*
γ	Mistura Gaussiana, representando o conjunto de pesos e Gaussianas
Δ	Conjunto de matrizes de transformação
ϵ	Fator de forma da superquádrica
λ	Conjunto de autovalores de uma matriz
Λ	Parâmetros de uma superquádrica
θ	Outra forma de representar misturas Gaussiana (como γ)
θ_k	Gaussiana, representada pela média e desvio padrão
θ^i	Mistura Gaussiana na iteração i
θ^*	Mistura Gaussiana ótima
$\theta^*_{(l,m)}$	Mistura Gaussiana ótima com restrição de uma união entre Gaussianas
ϑ	Ângulo de guinada (*yaw*)
μ	Média de uma distribuição Gaussiana $\mathcal{N}$
$\overline{\mu}_k$	Média estimado no *M-Step* para a Gaussiana k
π	Peso de uma distribuição Gaussiana associada a uma mistura Gaussiana
$\overline{\pi}_k$	Peso estimado no *M-Step* para a Gaussiana k
Π	Permutação do ponto p com todos outros pontos da nuvem
σ	Desvio padrão relativo a um dado agrupamento de covariância C
$\sigma_k(p_j)$	Variação da superfície da vizinhança do ponto p_j
σ_{max}	Máxima variação da superfície tolerada
$\mathbf{\Sigma}$	Matriz de covariância da uma distribuição Gaussiana $\mathcal{N}$
Σ_k	Covariância estimada no *M-Step* para a Gaussiana k
φ	Ângulo de arfagem (*pitch*)
ψ	Ângulo de rolamento (*roll*)
Ψ	Conjunto de primitivas de forma
a	Fator de escala da superquádrica
c	Função cosseno simplificada

C	Conjuntos de pontos que representam as mudanças segmentadas
$\mathbf{C}$	Matriz de covariância da k-vizinhança de p
d_{ij}	Métrica de distância entre dois pontos i e j
d_{GMM}	Distância entre duas misturas Gaussianas
$\mathbf{E}$	Matriz de fatores de escala
$\mathbf{I}$	Matriz de momentos centrais de uma superquádrica
K	Tamanho de uma Mistura Gaussiana
L	Constante real utilizada para estimação dos parâmetros do modelo
M	Dimensionalidade dos dados
n_{p_j}	Vetor normal ao ponto p_j e à vizinhança $N_{p_j}^k$
N	Número de pontos de uma nuvem de pontos ou cardinalidade dos dados independentes
$N_{p_j}^k$	Conjunto de índices dos k-vizinhos do ponto p_j na nuvem de pontos P
$\overline{N}_k$	Parâmetro de normalização estimado no *M-Step* para a Gaussiana k
$\mathcal{N}$	Distribuição normal multivariável ou Gaussiana
p_i	Ponto com índice i da nuvem de pontos P
p_j	Ponto com índice j da nuvem de pontos P
p_x	Posição do centro da superquádrica no eixo x
p_y	Posição do centro da superquádrica no eixo y
p_z	Posição do centro da superquádrica no eixo z
$\overline{p_j}$	Centróide da vizinhança de p_j
P	Nuvem de pontos com tamanho N em $\mathbb{R}^3$
$\|P\|_{max}$	Tamanho máximo da nuvem de pontos P
$r_{p_j}^k$	Raio máximo formado pelos k-vizinhos de p_j
$\mathbf{R}$	Matriz de rotação
s	Função seno simplificada
S	Outra forma de representar uma nuvem de pontos de tamanho N (como P)
$Size_p$	Número de pontos desejados na amostra
$\mathbf{T}$	Matriz de transformação
$T(x)$	Plano Tangente a Superfície formada pelos autovetor v_1 e v_2
$Tr(\Sigma)$	Traço da matriz de covariância
U_{th}	Limiar para determinação de mudança
$\mathbf{v}$	Conjunto de autovetores de uma matriz
X	Variável aleatória
Y	Dados independentes, ou dados de entrada
Z_{max}	Constante utilizada no processo de eliminação do plano do chão

Capítulo 1

Introdução

Neste primeiro capítulo será apresentada uma definição do problema tratado no trabalho, bem como as motivações e contribuições. Por fim, a organização geral do trabalho é mostrada visando introduzir as partes que o compõem.

1.1 Definição do Problema

O problema discutido neste trabalho refere-se à detecção de mudanças em mapas 3D (em inglês, *change detection* ou *novelty detection*), juntamente com o reconhecimento da forma dessas mudanças. Esses mapas podem ser obtidos por meio de diversas técnicas e conter diferentes tipos de informação. Neste trabalho, considera-se que um mapa do ambiente já foi obtido *a priori* e que é possível obter um novo mapa utilizando, por exemplo, um robô dotado de sensores. Esse novo mapa deve conter informações tridimensionais na forma de uma nuvem de pontos 3D, mas sem restrição na topologia desta informação, ou seja, os pontos podem ser esparsos e não-uniformemente distribuídos.

Observa-se que um mesmo conjunto de pontos 3D pode representar uma infinidade de objetos do mundo real. Sendo assim, apenas um conjunto de pontos é pouco expressivo para tais representações. Podem ser utilizados modelos não-paramétricos (como triangulação entre pontos) e paramétricos para essas representações. Entre outras vantagens, estes últimos são úteis pela facilidade de representação e indexação e, também, pela compactação dos dados.

Assim, o problema pode ser descrito como: *Dado um par de mapas 3D de um mesmo ambiente (um deles conhecido a priori e outro recém obtido) em um mesmo sistema de coordenadas, determinar pontos inseridos ou excluídos (mudanças) no mapa original com robustez a ruído, e segmentar e reconhecer a forma correspondente ao conjunto de pontos excluídos ou inseridos, com base em um modelo.* Impõem-se restri-

ções temporais ao sistema, restrições essas relacionadas ao tempo para aquisição dos mapas.

1.2 Motivação

O problema de detecção de mudança e recuperação de forma tem diversas aplicações importantes em diversas áreas da ciência. Uma área muito importante é a segurança e vigilância, na qual o uso de sistemas remotos são cada vez mais utilizados, principalmente com uso de câmeras de vídeo. Porém, a identificação e classificação ainda são realizadas, essencialmente, de forma manual. Devido à grande quantidade de informação fornecida por esses sistemas, a eficiência fica comprometida. Assim, a identificação automática de mudanças pode dar uma autonomia maior a tais sistemas.

Além disso, a detecção de mudança é extremamente importante no contexto de busca e salvamento, principalmente em locais de difícil acesso ou perigosos aos seres humanos. Robôs móveis estão sendo cada vez mais utilizados para tais tarefas, porém, dar maior autonomia a tais robôs ainda é um objetivo a ser alcançado. Considerando um conhecimento prévio do ambiente, pode-se utilizar a detecção de mudança para buscar objetos ou pessoas de interesse no local, como por exemplo sobreviventes ou objetos em incêndios, bem como determinar a gravidade de um desastre. A Figura 1.1 ilustra dois exemplos de robôs móveis utilizados em busca e salvamento, sendo o primeiro exemplo um robô terrestre dotado de diversos sensores, inclusive *laser scanner* [Andriluka et al., 2009]. O segundo exemplo trata de um robô aéreo utilizado para monitorar os destroços de um terremoto na Itália [Nardi et al., 2007].

Figura 1.1. Exemplos de robôs móveis utilizados em busca e salvamento, um robô móvel terrestre, à esquerda, e um robô móvel aéreo, em destaque à direita.

Atualmente, uma fonte sensora utilizada em larga escala nos robôs móveis são os

laser scanners, devido à boa relação custo × benefício, principalmente pela alta exatidão. As melhoras tecnológicas permitiram o desenvolvimento de equipamentos de alta qualidade, com custos cada vez menores, mais rápidos e com capacidade de obtenção de nuvem de pontos de alta densidade. Porém, para lidar com grandes quantidades de informação, são necessários métodos cada vez mais eficiente e automáticos, a fim de que essa informação seja efetivamente utilizada.

A Figura 1.2 ilustra um *laser scanner* 3D montado sobre um veículo móvel, juntamente com uma câmera de vídeo. Além disso, a figura mostra um mapa 3D de alta densidade em cores gerado a partir fusão de ambos os sensores.

Figura 1.2. *Laser scanner* 3D montado sobre veículo móvel, além de um exemplo de nuvem de pontos 3D gerada pelo sistema [3D Laser Mapping Co., 2009].

Considerando o uso de robôs móveis, a detecção de mudança ainda pode ser útil para alcançar o objetivo de se ter um robô completamente autônomo, para alcançar tal objetivo é necessário resolver três problemas:

- Usar as informações percebidas pelos sensores de forma a construir uma representação do ambiente;
- Localizar-se nessa representação;
- Decidir qual será o próximo movimento;

Existe uma gama muito ampla de técnicas que tentam resolver os dois primeiros problemas. Elas são conhecidas como técnicas de *Simultaneous Localization and Mapping* (*SLAM*). Essas técnicas estimam o mapa do ambiente e localizam o robô nesse mapa, sendo conhecidas suas percepções e movimentos.

Nos últimos anos, bastante progresso foi alcançado no problema de *SLAM*, buscando permitir aplicações em tempo real da técnica [Thrun et al., 2005]. Porém, para

permitir que o robô tome decisões sobre seus movimentos, ou seja, resolva o terceiro problema citado acima, é necessário dotá-lo da capacidade de detectar mudanças importantes no ambiente. Assim, quando o robô está se locomovendo por um lugar pela segunda vez, ele pode determinar se aconteceram mudanças e atualizar sua representação interna.

Um exemplo interessante de aplicação desse sistema é em um veículo autônomo que está trafegando por uma via obstruída por algum objeto não previsto anteriormente. A Figura 1.3 ilustra uma possível cenário, no qual após uma tempestade uma árvore obstruí a via, fazendo com que seja necessário detectar a mudança, bem como reconhecer a forma para calcular seu volume e posição. Com isso permitindo um replanejamento da trajetória, desviando do obstáculo.

Figura 1.3. Exemplo de cenário para detecção de mudança e reconhecimento de forma.

Uma motivação importante foi a participação do autor no projeto *Intelligent Robotic Porter System* (*IRPS*) . Esse projeto tem como parceiro o Instituto de Sistemas e Robótica da Universidade de Coimbra (ISR-UC), no qual foi desenvolvido parte do trabalho. O *IRPS* tem por objetivo desenvolver um sistema capaz de mapear grandes ambientes e detectar mudanças, além do desenvolvimento uma nova tecnologia de *laser scanner* tridimensional. A capacidade de detectar mudança é, portanto, uma parte muito importante do projeto. Visto o conhecimento prévio do mapa em *CAD* (Computer-Aided Design) do ambiente, é necessário recuperar as formas das mudanças a fim de atualizar tal mapa. Além disso, o *IRPS* tem por objetivo gerar mapas 3D de ambientes, fornecendo, assim, os dados de "entrada" ao sistema desenvolvido no presente trabalho. Esses mapas são gerados utilizando uma abordagem *SLAM* híbrida, que faz divisões topológicas no ambiente, sendo cada região topológica composta de um mapa métrico. Portanto, a principal motivação é detectar mudanças em uma região topológica, considerando duas leituras distintas da mesma região, bem como permitir a atualização dos mapas. No Anexo A pode ser encontrados mais detalhes sobre o

projeto.

1.3 Contribuições do Trabalho

Diversas contribuições podem ser destacadas no presente trabalho. A primeira é o uso de simplificação de pontos em mapas 3D. Esse tipo de técnica é extensivamente utilizada para nuvem de pontos 3D na área de computação gráfica, de forma a permitir uma visualização rápida. Porém, poucos trabalhos na área de mapas 3D com nuvem de pontos não-estruturadas em robótica fazem uso este tipo de técnica e, quando fazem, efetuam basicamente uma discretização da nuvem de pontos, sem conservar as características geométricas relevantes.

Outra contribuição importante foi dada na área de detecção de mudança em nuvem de pontos. A proposta de utilizar misturas Gaussianas (*GMM*) e a métrica *EMD*, conjuntamente com um algoritmo guloso, para permitir a detecção e segmentação das mudanças, mostra-se uma abordagem inovadora e promissora. Embora com um custo computacional ainda elevado, essa abordagem mostrou-se eficiente quando utilizada em conjunto com algoritmos de simplificação de pontos que preservam as características geométricas.

Na área de recuperação de formas foi proposto um novo método utilizando o espaço das Gaussianas, buscando utilizar as informações fornecidas pelo processo de detecção de mudanças, de modo a obter uma recuperação rápida e robusta. Uma análise comparativa com um método clássico foi realizada, mostrando as vantagens do método proposto.

Outra contribuição foi dada com a recuperação de formas utilizando superquádricas. Utilizou-se uma abordagem *split-and-merge*, que busca contornar restrições devido ao uso do algoritmo *EM* para cálculo da *GMM*. Adaptações foram feitas buscando sempre maximizar a eficiência considerando o uso do algoritmo *GMM* e do processo de simplificação. Foi proposta uma nova metodologia que utiliza como inicialização a segmentação de mudança, além de recuperar a informação topológica utilizando os dados fornecidos pela *GMM*. Um refinamento utilizando múltiplas escalas foi proposto, com objetivo de tornar a recuperação mais eficiente.

1.4 Organização do Texto

Neste capítulo foi apresentada uma introdução sobre o problema de detecção de mudanças e de recuperação de forma. No capítulo seguinte é realizada uma revisão dos

trabalhos encontrados na literatura sobre os assuntos tratados ao longo do texto. Trabalhos que abordam a simplificação de nuvem de pontos 3D, bem como detecção de mudança e recuperação de forma são discutidos tendo em vista sua aplicabilidade ao problema deste trabalho.

No Capítulo 3 é descrita, em detalhes, a metodologia proposta, começando por uma breve revisão da idéia e dos passos necessários para alcançá-la. Posteriormente, são descritos os métodos de simplificação de pontos utilizando agrupamento hierárquico, bem como o método de detecção e segmentação de mudança utilizando as técnicas *GMM* e *EMD*. Por fim, são apresentadas as técnicas de recuperação de formas, inicialmente utilizando formas básicas, e posteriormente, superquádricas.

No Capítulo 4 são descritas as instanciações experimentais utilizadas. Inicialmente, com o uso de dados simulados, supondo um *laser scanner* ideal. Por fim, a plataforma experimental para aquisição de mapas 3D é descrita em detalhes.

No Capítulo 5 são discutidos os resultados alcançados, onde cada parte da metodologia é validada. Utilizando dados reais e simulados, os resultados mostram a capacidade da metodologia em resolver o problema proposto.

O Capítulo 6 apresenta as conclusões finais do trabalho. Algumas possibilidades futuras também são apresentadas, que permitirão diversas extensões e melhorias ao presente trabalho. Além disso, as publicação obtidas como fruto deste trabalho são mostradas.

Capítulo 2

Trabalhos Relacionados

Neste capítulo serão apresentados alguns dos principais trabalhos encontrados na literatura pertinente à presente dissertação, tanto na detecção de mudanças quanto na recuperação de formas tridimensionais. Uma primeira parte sobre trabalhos relacionados à simplificação de nuvem de pontos também é pertinente, visto que as nuvens de pontos geradas por *laser scanners* apresentam redundância que prejudicam a eficiência dos algoritmos.

Nos trabalhos de detecção de mudança, embora a problemática apresentada pelo presente trabalho ainda esteja em aberto, são mostradas algumas abordagens com destaque aos trabalhos na área de robótica e trabalhos que utilizam sensores laser, principalmente da área de fotogrametria e sensoriamento remoto.

Por fim, são apresentados trabalhos relacionados com a recuperação de formas, basicamente, oriundos da área de computação gráfica e visão computacional, com destaque para os trabalhos que utilizaram formas básicas e superquádricas.

2.1 Simplificação de Nuvem de Pontos

Considera-se nuvem de pontos P, no escopo deste trabalho, um conjunto de pontos tridimensionais p_i tal que $i = 1..N$ e $p_i \in \mathbb{R}^3$, onde N é o tamanho da nuvem e no qual se tem apenas a informação geométrica sobre esses pontos, sem conhecimento de qualquer relação topológica (relação entre os pontos). Assim, um passo importante, antes de detectar a mudança e definir uma forma, é simplificar tal nuvem. Esta simplificação deve ser feita de forma que pontos importantes para representação geométrica permaneçam na nuvem, e que a restrição temporal seja respeitada. Além disso, é interessante para esta abordagem que seja possível recuperar a nuvem completa posteriormente, além de gerar uma representação multi-escala.

As abordagens encontradas na literatura são basicamente da área de computação gráfica com objetivo principal de visualização, normalmente buscando a qualidade da nuvem gerada e não o desempenho temporal. Dentre elas, destacam-se as técnicas de simplificação baseadas em malhas. Um interessante comparativo entre tais métodos foi feito por Cignoni et al. [1997]. Uma análise da área também foi feita por Heckbert & Garland [1997], com uma revisão posterior sob o ponto de vista de desenvolvimento dos algoritmos por Luebke [2001].

Embora os métodos de simplificação de malhas sejam muito eficientes, esses precisam de um conhecimento prévio da malha, ou seja, a relação topológica entre os pontos. Porém, essa relação não está disponível *a priori* em nuvens de pontos 3D fornecida por *laser scanners*. Logo, não está disponível neste trabalho, como mencionado anteriormente. Técnicas que modelam tais nuvens de pontos são largamente discutidas na literatura [Hoppe, 1994; Amenta et al., 1998; Gopi et al., 2000; Liu & Yuen, 2003; Schroeder et al., 2004], e a grande maioria delas constrói malhas de polígonos que melhor aproximam, ou interpolam, os dados de entrada. No entanto, esses métodos apresentam um elevado custo computacional, o que os tornam menos atrativos ao presente trabalho. Assim, o uso de técnicas de simplificação de nuvem de pontos, que dependem do conhecimento prévio da malha é inviável a este trabalho.

Na literatura, diversas abordagens foram propostas ao longo dos anos para simplificação de nuvem de pontos que não consideram um conhecimento prévio da malha. O trabalho de Kobbelt & Botsch [2004] faz análise dos métodos que utilizam nuvem de pontos com intuito de simplificação, determinação de vizinhança, além de determinação de malhas. Entre os primeiros trabalhos a lidar com o problema de simplificação diretamente sobre a nuvem de pontos destacam-se: o trabalho de Dey et al. [2001] que explora a estrutura celular dos diagramas de Voronoi 3D para determinar regiões superamostradas, e assim os pontos candidatos a remoção. Uma eliminação de pontos é efetuada levando-se em consideração um valor de densidade de pontos definido pelo usuário. Tal método exige a computação e manutenção de diagramas de Voronoi 3D resultando assim em um alto consumo de memória e de processamento.

O trabalho de Boissonnat & Cazals [2001] também constrói e mantém uma triangulação de Voronoi 3D. Nesse trabalho é realizada uma simplificação por meio de uma abordagem *coarse-to-fine*. Partindo de um subconjunto de pontos aleatoriamente selecionados é realizada a triangulação. Utilizando os diagramas se obtém uma métrica de distância entre pontos. Então, usando uma tolerância máxima, o subconjunto é aumentado iterativamente, considerando esta distância. Posteriormente, o diagrama é recalculado e a superfície é reconstruída. Apesar dos bons resultados, considerando o problema deste trabalho, tal abordagem se mostra inadequada, pois além do elevado

custo computacional associado ao diagrama de Voronoi, ainda obtém-se como resultado a superfície da nuvem de pontos, cujo custo associado para computar não traz vantagens ao problema deste trabalho.

O artigo, também pioneiro, de Alexa et al. [2001] propôs o uso de *Moving Least Squares* (*MLS*) para modelar as superfícies definidas pela nuvem de pontos. O *MLS* é uma técnica que aproxima localmente a superfície por um polinômio de duas variáveis usando a técnica de minimização de quadrados. Assim, os pontos são projetados na superfície determinada pelo *MLS*, e os pontos que pior representam a superfície são excluídos iterativamente. A Figura 2.1a ilustra uma nuvem de pontos com 37.000 pontos, sendo que a cor, que varia de azul a vermelho, representa a relevância de cada ponto. Caso seja azul, é um forte candidato a ser eliminado. A Figura 2.1a mostra a nuvem após a simplificação com, aproximadamente, 20.000 pontos.

(a) (b)

Figura 2.1. Simplificação de nuvem de pontos proposto por Alexa et al. [2001]. Em (a) é mostrada a nuvem composta por 37.000 pontos, com a cor variando do azul ao vermelho, representando a importância de cada ponto. Em (b), o resultado do algoritmo de simplificação é mostrado com uma redução para, aproximadamente, 20.000 pontos.

Alexa et al. [2001] também propõem um método para "aumentar" o número de pontos, em caso de nuvem de pontos subamostrada, usando MLS e diagramas de Voronoi. Apesar desse método obter bons resultados, o custo computacional associado ao passo iterativo de reprojetar pontos e recalcular a superfície usando MLS é muito elevado.

Linsen [2001] também foi um dos trabalhos pioneiros a tratar do problema de simplificação direta de nuvem de pontos com informação de textura. A abordagem baseia-se no uso de entropia sob a informação disponível de cada ponto, composta da curvatura local e da variação de cor em relação aos k-vizinhos mais próximos. O método

é simples e eficiente, porém no caso da informação de cor não estar disponível, ele se torna pouco efetivo visto que o uso de entropia ficaria limitado apenas à informação de curvatura.

No trabalho recente de Lee & Jong [2008] é proposto um método de simplificação baseado no uso de *Discrete Shape Operator* (*DSO*). Tal operador estima a variação local da superfície, possibilitando determinar os melhores pontos a serem eliminados, ou seja, pontos com curvatura. Um processo iterativo de eliminação dos pontos é realizado até que o modelo tenha o número de pontos desejado. A necessidade de computar as torções e curvaturas das superfícies para cálculo do *DSO*, embora discretas em relação aos k-vizinhos mais próximos, torna o método muito custoso.

O trabalho de Pauly & Gross [2001] tratou o problema de simplificação utilizando a transformada de Fourier. Nesse trabalho, a nuvem de pontos é aproximada localmente em uma grade regular a partir da qual é aplicada a *Discrete Fourier Transform* (*DFT*). Em seguida, são aplicados filtros no domínio da frequência. Uma reamostragem é realizada gerando o modelo simplificado por meio da Transformada Inversa de Fourier. Os problemas dessa abordagem que inviabilizam seu uso estão associados ao elevado custo computacional da Transformada de Fourier, tanto direta quanto inversa.

Um importante trabalho foi desenvolvido por Pauly et al. [2002]. Nesse trabalho são adaptados e comparados métodos largamente utilizados para simplificação de malhas, no contexto de nuvem de pontos. É proposto o uso do método dos k-vizinhos mais próximos e de análise de covariância sobre a vizinhança para suprir a falta da informação topológica. Quatro algoritmos foram desenvolvidos no referido trabalho: dois baseados em métodos de agrupamento (*clustering*), um utilizando abordagem *top-down* (hierárquico) e outro com abordagem *bottom-up* (incremental); outro método em simplificação iterativa usando erro quadrático e um quarto método usando simulação de partículas. Os métodos de agrupamento se mostraram muito eficientes no tempo de processamento e no uso de memória, com vantagem para o método de agrupamento hierárquico em relação ao método incremental, na qualidade dos resultados obtidos comparados à nuvem original. Além disso, o agrupamento hierárquico, devido à criação de cima para baixo (*top-down*) constrói uma árvore que permite uma representação multi-escala da nuvem de pontos. O método de simplificação iterativa mostrou-se muito sensível ao tamanho da nuvem de pontos em relação ao custo computacional, apesar de apresentar uma qualidade de simplificação superior aos métodos de agrupamento. O mesmo aconteceu com o método de simulação de partículas que obteve resultados de excelente qualidade, mas ineficiente computacionalmente.

A Figura 2.2 mostra um exemplo de simplificação utilizando os algoritmos propostos por Pauly et al. [2002]. As imagens na primeira coluna foram produzidas utilizando

agrupamento incremental, e as imagens na segunda resultaram do agrupamento hierárquico. A Figura 2.2a, mostra o resultado do método utilizando todos os pontos, com as cores representando os agrupamentos. A Figura 2.2b ilustra a simplificação com 1.000 pontos, com o centróide representando o ponto simplificado do agrupamento. O tamanho do agrupamento é mostrado pelo "tamanho" do ponto preto.

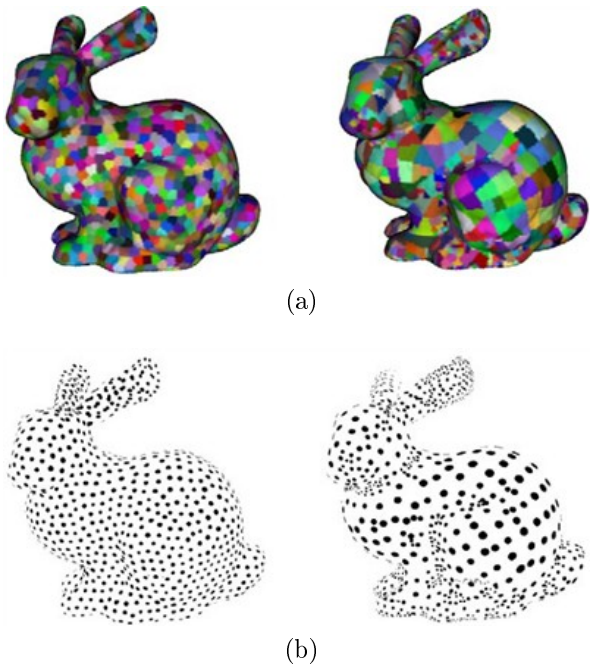


Figura 2.2. Simplificação de nuvem de pontos utilizando agrupamento [Pauly et al., 2002]. Em (a) são ilustradas as superfícies geradas, com as cores representando os pontos agrupados. Em (b) é mostrado o resultado da simplificação, por meio de centróide e do raio de cada agrupamento.

Os trabalhos de Moenning & Dodgson [2003] e, posteriormente, Moenning & Dodgson [2004], propõem um método para simplificação de nuvem de pontos, com a finalidade de visualização. Tal método utiliza-se de uma abordagem *coarse-to-fine* por meio do *Fast Marching Farthest Point Sampling* (*FastFPS*), que possibilita a geração de uma nuvem de pontos simplificada, uniforme e com representação em múltiplas escalas. Tal método, porém, tem dificuldade para tratar de nuvens com pontos não uniformemente distribuídos e sem garantia de densidade de pontos, embora melhorias tenham sido propostas em Moenning & Dodgson [2004]. Mesmo que a nuvem de pontos gerada tenha garantias de densidade, o custo computacional torna seu uso inviável para o caso do presente trabalho, que privilegia a manutenção da informação de forma e não a qualidade de visualização, que requer a distribuição dos pontos mais uniforme.

Um trabalho de simplificação e representação multi-escala utilizando *Graphics Processing Unit* (*GPU*) foi desenvolvido por Gobbetti & Marton [2004]. Nesse ar-

tigo é proposta uma abordagem para representar uma nuvem de pontos em camadas utilizando o paralelismo computacional fornecido pelas placas gráficas atuais. Nessa abordagem, a arquitetura da *GPU* é explorada visando maximizar o uso de processamento, bem como a simplificação com objetivo de visualização. Trata-se de uma abordagem *coarse-to-fine*, que permite uma visualização grosseira com uma posterior melhoria na qualidade, de forma a permitir aplicações tempo real. A principal restrição do método que impede seu uso direto neste trabalho é a necessidade de se iniciar com uma nuvem de pontos uniformemente espaçados, tendo em vista a maneira como método explora a distribuição dos dados.

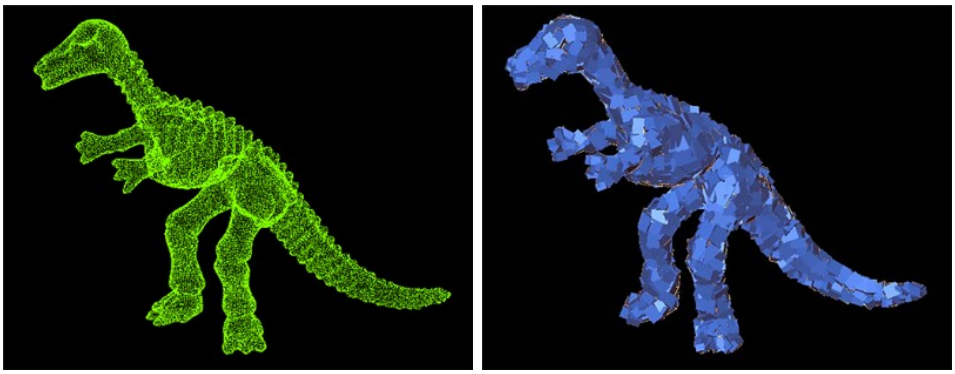
A Figura 2.3 ilustra os diversos níveis de simplificação propostos no trabalho de Gobbetti & Marton [2004]. De um modelo original com 28.000.000 pontos, foi efetuada a simplificação do nível mais grosseiro (à esquerda) até o mais fino (à direita), cada nível com 4.000, 12.000 e 3.450.000 pontos, respectivamente.

Figura 2.3. Simplificação em níveis proposta por Gobbetti & Marton [2004], com variação do refinamento da nuvem de pontos, da esquerda para a direita.

Um trabalho interessante foi desenvolvido por Yamazaki et al. [2006], onde é realizada uma simplificação com a finalidade posterior de se obter a segmentação. O método determina os k-vizinhos mais próximos, e então constrói um grafo a partir do qual determina a função de centralidade [Freeman, 1979]. Esta medida define uma aproximação da distância entre todos os pontos de forma a determinar os pontos mais representativos que serão definidos como características relevantes da nuvem (*features*). Tais pontos são uma simplificação da nuvem de pontos utilizados no processo de segmentação usando a técnica de corte em grafos. A complexidade do método foi apresentada no trabalho, que tem como principal restrição a computação da função de centralidade com desempenho $O(n^2 logn)$, sendo n o número de pontos. Tal complexidade semelhante ao cálculo de uma malha a partir de nuvem de pontos é impeditiva ao

presente trabalho, embora no mesmo trabalho tenha sido proposta uma simplificação para a função de centralidade com complexidade final de $O(n\sqrt{n}logn)$.

O trabalho proposto por Zou & Ye [2007] realiza segmentação utilizando uma abordagem multi-escala baseada em *Oriented Boundary Box* (*OBB*) hierárquica [Gottschalk et al., 1996], que basicamente constrói uma árvore de regiões definidas por caixas orientadas pela normal da região local e localizadas no centro de massa. Para segmentação é utilizado um método de agrupamento nebuloso baseado no algoritmo *K-means*. Embora a abordagem tenha alcançado bons resultados de segmentação, o uso da *OBB* para simplificação não se mostra adequado devido à sua representação retangular do espaço, mesmo que considerando as curvaturas locais, o que pode dificultar o posterior reconhecimento de forma. A Figura 2.4 ilustra a simplificação, com a nuvem original na Figura 2.4a e a simplificação utilizando *OBB*, na Figura 2.4b.



(a) (b)

Figura 2.4. Simplificação de uma nuvem de pontos utilizando o algoritmo *OBB* [Zou & Ye, 2007], em (a) é mostrada a nuvem de pontos original, enquanto em (b) é ilustrada a representação da *OBB*.

2.1.1 Resumo

No que tange a simplificação de nuvem de pontos, a Tabela 2.1 busca comparar os métodos considerando características como eficácia, eficiência, facilidade para representação em escalas e a necessidade de informações adicionais, além da informação da distribuição geométrica dos pontos. Cada trabalho foi classificado em três níveis: muito bom(+), bom ($\pm$) e regular ($-$). Assim, os métodos de agrupamento propostos no trabalho de Pauly et al. [2002] se destacam devido, principalmente, a eficiência. A possibilidade de uma representação em escala obtida diretamente pelo método também faz dele um método adequado para a presente proposta.

Tabela 2.1. Trabalhos em simplificação de nuvem de pontos.

Método	Eficiência	Eficácia	Escalas	Inf. Adicional
[Dey et al., 2001]	−	±	−	Não
[Boissonnat & Cazals, 2001]	−	±	+	Não
[Alexa et al., 2001]	−	+	−	Não
[Linsen, 2001]	+	±	−	Sim (Cor)
[Pauly & Gross, 2001]	−	+	−	Não
[Pauly et al., 2002]-*Cluster*	+	±	+	Não
[Pauly et al., 2002]-Outros	±	+	±	Não
[Moenning & Dodgson, 2004]	−	+	±	Não
[Gobbetti & Marton, 2004]	+	+	+	Sim (Densidade)
[Yamazaki et al., 2006]	±	+	−	Não
[Zou & Ye, 2007]	+	±	±	Não
[Lee & Jong, 2008]	−	+	−	Não

2.2 Detecção de Mudanças

Os trabalhos de detecção de mudanças podem ser encontrados nas mais diferentes áreas da ciência, sob nomenclatura de detecção de mudanças, novidades, diferenças ou anormalidades. Nos trabalhos de Markou & Singh [2003a,b], são feitas revisões dos trabalhos de detecção de mudanças com aplicações em detecção de alvos em radares [Carpenter et al., 1997], controle de processos estatísticos [Guh et al., 1999], reconhecimento de escrita [Tax & Duin, 2000], detecção de tumores [Tarassenko et al., 1995; Cheng et al., 2006], detecção de falhas [King et al., 2002], Internet [Jin et al., 2007], além de robótica e outros. A taxonomia empregada em Markou & Singh [2003a,b] divide os trabalhos de detecção de mudança pelos tipos de técnicas empregadas para solucionar o problema. Basicamente, as técnicas são divididas como baseadas em redes neurais [Markou & Singh, 2003b] e técnicas baseadas em abordagens estatísticas [Markou & Singh, 2003a].

Neste trabalho, os trabalhos de interesse serão pela área de aplicação, devido a dificuldade de encontrar trabalhos que tratam de problemas semelhantes. Uma primeira seção mostram alguns trabalhos em visão computacional, com foco em detecção de mudanças em imagens. Posteriormente, os trabalhos na área de fotogrametria e sensoriamento remoto são apresentados, com foco nos trabalhos utilizando informação tridimensional. Os trabalhos de detecção de mudança aplicados à robótica também são introduzidos. Por fim, um resumo da revisão de detecção de mudança é mostrado.

2.2.1 Técnicas de Detecção de Mudanças em Visão Computacional

Diversos trabalhos sobre detecção de mudanças em visão computacional foram encontrados na literatura. Uma recente análise da área foi desenvolvida por Radke et al. [2005]. Os trabalhos podem ser divididos em detecção baseada em pixels e em regiões. Basicamente, os trabalhos tentam encontrar o limiar que determina se há ou não uma mudança, sendo, em alguns casos, capaz de identificá-las. Quanto aos trabalhos baseados em pixels, o trabalho de Rosin [1998] descreve alguns métodos, que são desenvolvidos e comparados. Wren et al. [1997] propôs um método robusto para seleção adaptativa de limiares de cada pixel, baseado na distribuição dos níveis de cinza da imagem.

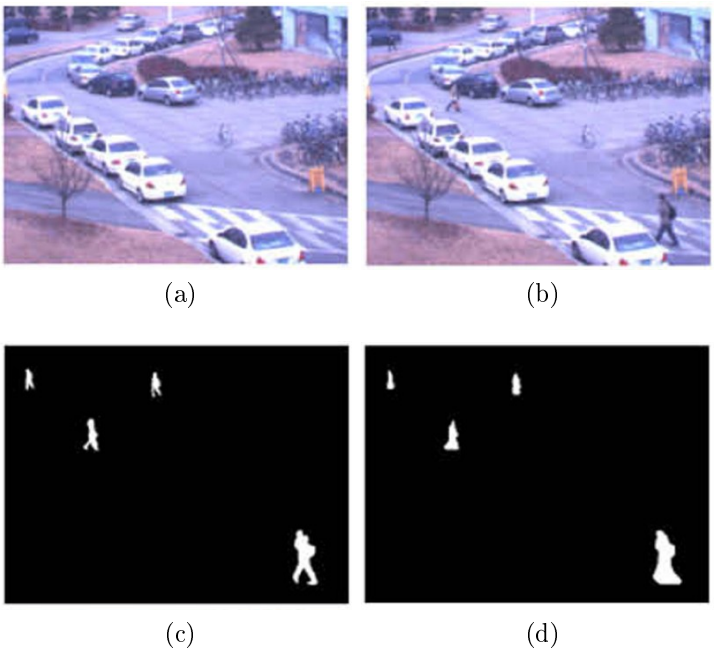
A abordagens baseadas em pixels têm baixo custo computacional em relação às abordagens baseadas em regiões, porém são mais suscetíveis a ruído, mudanças de iluminação e diferenças no ponto de vista, assim como ocorre nas técnicas de detecção de mudanças em nuvem de pontos baseadas na relação individual entre pontos, que tendem a ser sensíveis a ruído e erros de localização.

Dos trabalhos que utilizam técnicas baseadas em regiões para detecção de mudanças em imagens, destaca-se o trabalho pioneiro proposto por Hsu et al. [1984]. Nesse trabalho, são desenvolvidos três modelos de aproximação de regiões, a partir disto, são detectadas mudanças por meio do cálculo de um limiar. Usando pares de imagens de vigilância, concluiu-se que o modelo quadrático era o mais adequado devido ao alto grau de confiança, embora com resultados similares na detecção. Aach et al. [1993] apresentou três métodos estatísticos para detecção de mudanças: *Local Sum of Squared Differences* (*LSSD*), *Local Sum of Absolute Differences* (*LSAD*) e *Local Average of Differences* (*LAD*). Tais métodos são comparados em desempenho e individualmente.

No trabalho de Hotter et al. [1996], utilizou-se a diferença de níveis de cinza, de textura e de movimento, todos em cascata para redução de alarmes falsos em detecção de pessoas. Essa técnica apresentou boa tolerância a ruído, embora não obtendo bons resultados quando submetida a variações na condição de iluminação. Por outro lado, anteriormente, Skifstad & Jain [1989] propuseram um método baseado em modelo de sombra, que utiliza a variância das intensidades e um limiar determinado empiricamente para realizar a detecção.

Recentemente, Hwang et al. [2004] propuseram um método para detecção estatística de mudanças em imagens, cujo método foi melhor explorado em Hwang et al. [2008]. Por meio de uma modelagem do ruído em cada canal de cor e da distância euclidiana, é feita a seleção do limiar para detecção de mudanças em imagens colori-

das. Os resultados comparados com o algoritmo *Integration of Intensity and Texture Differences* (*IITD*) proposto por Li & Leung [2002], um dos algoritmos mais utilizados até então, mostrou a robustez do método a ambientes internos e externos, para realização da detecção. Porém, o problema de diferença de ponto de vista entre a imagem de referência e a imagem mudada ficou em aberto. A Figura 2.5 ilustra os resultados obtidos pelo método *IITD*. A Figura 2.5a ilustra a imagem original e a Figura 2.5b após a mudança. A Figura 2.5c mostra o *ground truth*, e na Figura 2.5d é mostrado o resultado obtido pelo método proposto por Hwang et al. [2008] utilizando o espaço de cor *CIE Lab*.



(a) (b)

(c) (d)

Figura 2.5. Exemplo de detecção de mudança em imagens utilizando método proposto por [Hwang et al., 2008]. (a) mostra a imagem sem a mudança, enquanto em (b) é mostrada a mesma cena após a inserção da mudança; (c) é mostrado o *ground truth* da diferença e em (d) é mostrado o resultado obtido pelo método de Hwang et al. [2008].

Ainda, no trabalho de Tanjung & Lu [2007], é apresentada uma revisão de vários métodos para detecção automática de mudanças em ambientes internos. Além disso, é proposto um sistema que utiliza a *Scale-Invariant Feature Transform* (*SIFT*) [Lowe, 1999] para registro das imagens, bem como uma diferença e limiarização entre a imagem a ser detectada a mudança e a imagem de referência. Por fim, utiliza-se algoritmos de morfologia matemática para remoção de ruído e dessa forma facilitar a localização. Os resultados mostram a capacidade do método em detectar mudanças, apesar da sensibilidade a variações de iluminação, bem como a mudanças de ponto de vista.

2.2.2 Técnicas de Detecção de Mudanças em Fotogrametria e Sensoriamento Remoto

Na área de detecção de mudanças em fotogrametria e sensoriamento remoto, os trabalhos utilizando dados tridimensionais foram focados, principalmente adquiridos por *laser scanners* 3D.

O trabalho pioneiro de Steinle et al. [1999] mostrou uma abordagem simples. Para isso, um *laser scanner* foi preso a um veículo aéreo, equipado com *Global Positioning System* (*GPS*) diferencial para localização. A partir da nuvem de pontos 3D coletada é gerado manualmente um mapa do ambiente utilizando *Computer-Aided Design* (*CAD*). Posteriormente, novos dados são comparadas com o arquivo *CAD*, permitindo visualizar as mudanças. As restrições do método são, basicamente, limitações do *laser scanner*, como a não visualização de regiões transparentes ou pequenas, dificuldade na determinação correta das bordas, etc. Além disso, não se trata de um método automático. A Figura 2.6 mostra um exemplo de diferença entre o *CAD*, em tons mais claros, e os dados adquiridos pelo do *laser scanner*, tons mais escuros.

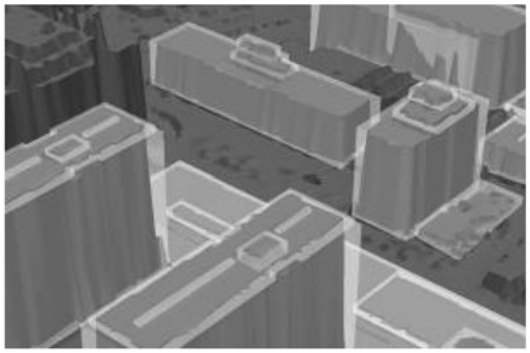


Figura 2.6. Detecção de Mudança utilizando dados adquiridos por *laser scanner* aerotransportado comparados com dados de *CAD* [Steinle et al., 1999]. Em tons mais escuros são mostradas as informações do laser scanner, enquanto os tons mais claros representam a informação do *CAD*.

Além desse trabalho, diversos outros trabalhos utilizando *laser scanner* são reportados na literatura dessas duas áreas. Na maioria dos trabalhos reportados são utilizados *laser scanners* aerotransportados. Esses sensores obtém imagens de profundidade da superfície, para uma posterior comparação manual com dados adquiridos em um instante de tempo anterior. Porém, houve pouco avanço quanto ao uso de métodos totalmente automáticos para realizar a detecção de mudanças.

No trabalho de Vosselman et al. [2004a] foi proposto um método para detecção de mudanças com capacidade de atualizar mapas, usando *laser scanner* aerotransportado. Esse trabalho utiliza o método de segmentação proposto por Sithole & Vosselman

[2003], no qual se divide o mapa adquirido em planos de mesma altura. Para cada plano é construído um grafo e determinada a árvore geradora mínima (*Minimum Spanning Tree* (*MST*)), com algumas arestas sendo removidas por meio de limiares de inclinação e comprimento dos planos. A *MST* é dividida em segmentos de reta que definem regiões a serem segmentadas. Tais regiões são classificadas em vegetação, construções ou áreas livres, usando para sua determinação características como cor, tamanho e altura. A classificação é realizada por meio da comparação de um mapa conhecido anteriormente com os dados adquiridos pelo *laser scanner*. Basicamente, o trabalho efetua uma classificação sobre a existência ou não de mudança, com pouca capacidade de lidar com falsos positivos (*outliers*).

Outro trabalho similar foi publicado por Vogtle & Steinle [2004], onde são comparadas duas aquisições de *laser* usando o método Modelo de Superfícies Digitais (*DSM*) diferencial, onde a finalidade é classificar a mudança no contexto de terremotos. Assim, determina-se o tipo de construção: nova, aumentada, reduzida ou demolida. Para tal, utiliza-se a informação de altura e limiares pré-determinados.

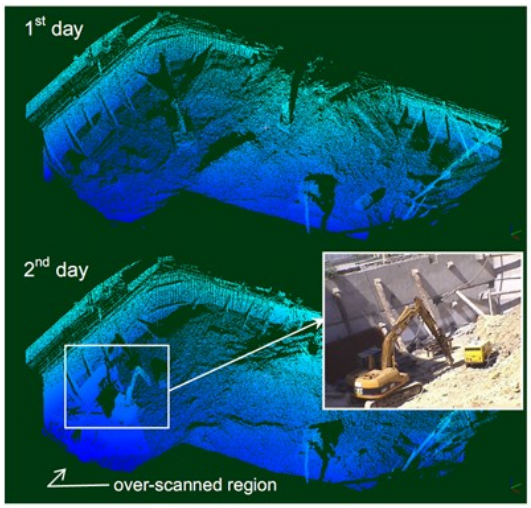
O trabalho de Vu et al. [2004] utiliza dados de sensor *laser* aerotransportado para detectar mudanças. Uma abordagem muito simples com uso de histogramas de altura e morfologia matemática permite a determinação de mudanças, além de classificar coma região como demolida ou nova. Os resultados mostram que, embora as novas construções sejam classificadas, dificilmente consegue classificar se a construção foi demolida.

O trabalho de Brito et al. [2008]faz uso de *DSM* por meio de dados adquiridos por um par estéreo de câmeras aerotransportado. Nesse trabalho, o conhecimento *a priori* de pontos de controle selecionados manualmente para permitir o alinhamento de dados temporalmente distintos é considerado. A detecção de mudança é feita por meio de uma comparação direta entre dados com mesma informação de altura.

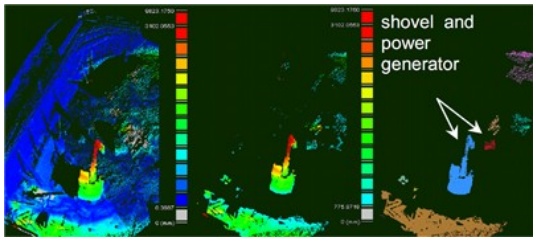
Um trabalho interessante utilizando *laser scanner* foi proposto por Girardeau-Montaut et al. [2005]. Nesse trabalho, utilizou-se um sensor laser 3D montado em solo para detectar mudanças, gerando mapas 3D do ambiente, de modo semelhante ao presente trabalho. É proposto um método automático para determinar a existência de mudanças, bem como segmentá-las. Considerando conhecida a posição e orientação do laser durante a aquisição dos dados, é utilizado o algoritmo *Iterative Closest Point* (*ICP*) [Besl & McKay, 1992] para alinhamento da nuvem de pontos, e uma posterior criação do mapa usando árvores *octree*, com profundidade máxima conhecida. Diferentes métodos para comparação das árvores foram testados, usando distância média entre pontos, determinação da melhor orientação entre planos e a distância de Hausdorff Rote [1991]. A distância de Hausdorff apresentou o melhor desempenho, embora

tenha apresentado um custo computacional maior e dependente da altura da árvore *octree*. A determinação da altura da árvore é feita por meio de um limiar que determina o número máximo de pontos em uma célula, sendo assim possível determinar a menor altura para a árvore.

A Figura 2.7 ilustra a detecção de mudança do trabalho de Girardeau-Montaut et al. [2005]. A Figura 2.7a mostra os dois mapas 3D utilizados nos experimentos, com a mudança mostrada na imagem em destaque. A Figura 2.7b mostra as mudanças detectadas inicialmente, após aplicação de limiares e após ajustar os componentes conexos, respectivamente. A figura ainda mostra a detecção da mudança, ou seja, a retroescavadeira (*shovel*) e o gerador de energia (*power generator*), bem como as regiões que foram escaneadas apenas no mapa mais "novo".



(a)



(b)

Figura 2.7. Detecção de mudança utilizando *laser scanner* 3D [Girardeau-Montaut et al., 2005], em (a) são mostrados os dois mapas 3D utilizados nos experimentos, bem como uma imagem real da mudanças do segundo mapa e as regiões *over-scanned*. Na Figura (b) é mostrado o resultado da distância dos pontos do novo mapa (à esquerda). Ao centro, o resultado após o uso de limiares e, à direita, após colorir com a mesma cor os componentes fortemente conexos.

Ainda nesse trabalho são tratados outros problemas como diferenças de amostra-

gem entre aquisições do *laser* e a comparação com o vizinho mais próximo. O primeiro problema é tratado comparando o ponto a uma superfície formada pelos vizinhos mais próximos. Porém, isso implica num aumento expressivo do tempo de processamento necessário para o método. O segundo problema é tratado por meio de mapas de visibilidade do mapa de referência gerados a partir de imagens de profundidade. Assim, pontos escondidos ou fora do campo de visão, que não têm correspondência, podem ser tratados. É proposto um tratamento para os pontos escondidos, com o uso de um terceiro mapa. O primeiro problema não é apresentado no neste trabalho devido ao uso de métodos de agrupamento. Para o segundo problema tal abordagem não se adapta, devido à inviabilidade de se efetuar projeções 2D do mapa tridimensional, que permitiriam criar imagens de profundidade.

2.2.3 Técnicas de Detecção de Mudanças em Robótica

A literatura de detecção de mudanças em robótica apresenta alguns trabalhos com diferentes abordagens, tanto usando métodos probabilísticos, quanto métodos baseados em redes neurais. A maioria dos trabalhos utiliza câmeras de vídeo como fonte de informação. A principal vantagem deste tipo de sensor é possibilidade de detecção de cores e texturas; que facilita a tarefa de detectar mudanças. Poucos trabalhos utilizam sensores de distância (*range*), embora esses apresentem vantagens significativas na obtenção de dados tridimensionais como maior exatidão e maior robustez a variações de condições de iluminação. Sonares foram utilizados em alguns trabalhos apresentados nesta seção, bem como *laser scanners*.

Nos últimos anos, os trabalhos do grupo de robótica cognitiva sob coordenação do Prof. Ulster Nehmzow tem se destacado na área de detecção de mudanças. Inicialmente, com o trabalho de Marsland et al. [1999] que propõem uma rede neural utilizando o princípio da habituação de Wang [1993], que tenta explicar como o cérebro aprende a ignorar estímulos repetidos, fundamentado no principio básico da plasticidade (*plasticity*), ou seja, adaptação a mudanças. Essa rede neural é composta de um grande número de colunas organizadas verticalmente, com cinco camadas de células em cada coluna, onde cada camada consiste de um número n de neurônios. Tal técnica mostra-se inviável em aplicações práticas de robótica devido ao elevado custo computacional. Outras formas de desenvolver habituação foram propostas, de forma a permitir a sua implementação em robôs reais.

Em Marsland et al. [2000c], é utilizada uma adaptação dos mapas de Kohonen (*SOM*) [Kohonen, 1982, 2001], utilizando o principio da habituação com o modelo proposto por Stanley [1976]. Basicamente, é utilizado um mapa auto-organizável (*Self-*

Organizing Map (*SOM*)) para classificar o vetor de entrada, baseado em informações de sonares. Além disso, é usada a habituação para implementar o filtro de novidade. Tais métodos são descritos com maiores detalhes nos trabalhos de Marsland et al. [2000a] e Marsland et al. [2001]. Em Marsland et al. [2000b] são propostas duas alternativas ao método anterior, que utilizava *SOM* como entrada do método. Nesse trabalho, propõe-se utilizar *K-means* [Macqueen, 1967] e *Temporal Kohonen Map* (*TKM*) [Chappell & Taylor, 1993] como métodos de agrupamento (*clustering*). O *TKM* utiliza o *SOM* clássico, porém com a atividade dos neurônios decaindo exponencialmente com o tempo (*leaky integrator*), fazendo assim com que o mapa de Kohonen tenha um comportamento semelhante a uma memória de curta duração. É realizada uma comparação dos dois algoritmos, além da adaptação dos *SOMs* [Marsland et al., 2000c], com o principio de habituação. Os resultados foram obtidos em um robô muito simples com sensores de iluminação. Os três métodos apresentaram resultados muito semelhantes, mesmo com algoritmo *SOM* apresentando resultado consistente quando um novo padrão é mostrado e o *TKM* sendo mais rápido. Quando dois novos padrões são apresentados, o *TKM* e o *K-means* foram melhores que o *SOM*, com vantagem para o *TKM* na velocidade de resposta.

No trabalho de Marsland et al. [2000a], o algoritmo *Habituating Self-Organizing Maps* (*HSOM*) foi descrito com maiores detalhes. Esse mostrou-se capaz de detectar novos objetos. O trabalho basicamente descreve a utilização de um SOM implementando *Learning Vector Quantization* (*LVQ*) [Kohonen, 1982] com a estratégia do vencedor ganha tudo (*"winner-take-all"*). As principais desvantagens da abordagem se devem principalmente ao uso do *SOM*, com sua estrutura estática, o que muitas vezes não é útil em robótica, devido ao conhecimento dinâmico e crescente que se tem do ambiente de trabalho do robô.

Marsland et al. [2001] propõem uma alternativa ao *SOM*, que foi denominada *Grow When Required* (*GWR*). Essa técnica permite a inserção de novos neurônios, quando necessário. Por meio de um contador implementado em cada neurônio, é possível saber se um neurônio está "saturado". Assim, com o uso de dois limiares, definidos empiricamente, é possível determinar se a rede precisa crescer. Basicamente, um limiar define que o neurônio está recebendo muitas informações e outro que define se a resposta dele, observando o principio da habituação, está de acordo com o esperado. O algoritmo foi comparado com dois *SOM* de tamanhos diferentes, em relação ao problema de detecção de mudanças. Os testes mostram que o *SOM*, com menor dimensão saturou rapidamente, não conseguindo "aprender" a partir de então. Enquanto o *SOM* de maior dimensão teve dificuldade em detectar a mudança. Por outro lado, a *GWR* mostrou resultados promissores, conseguindo reconhecer o novo objeto ao final

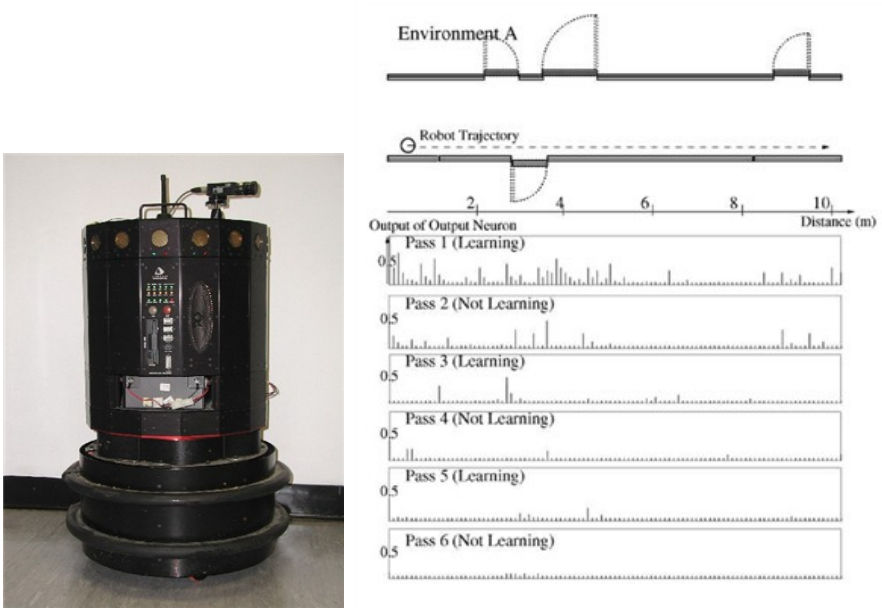
do terceiro ciclo de aprendizado.

Crook et al. [2002] compararam o algoritmo *GWR* com outro algoritmo baseado em redes de Hopfield [Hopfield, 1982]. Os experimentos foram realizados em duas plataformas robóticas: a primeira utilizando um robô dotado de sonar e a segunda de um robô obtendo imagens de uma câmera binária de 48×48 pixels. Embora os dois algoritmos tenham mostrado resultados semelhantes, a *GWR* apresentou uma pequena vantagem, além de mostrar-se mais tolerante a ruído.

No trabalho de Marsland et al. [2002] foi proposta uma extensão para a *GWR*, na qual se utilizam vários filtros de novidade, com seleção dinâmica entre eles. Nesse sistema, vários filtros são treinados em diferentes ambientes. Um vetor de familiaridade é utilizado para manter a resposta dos diversos filtros de novidade e selecionar qual ambiente está sendo explorado no momento (auto-localização). Os experimentos foram realizados de maneira similar aos outros trabalhos dos autores, utilizando dados de sonar captados por um robô usando algoritmo seguidor de paredes em um ambiente de corredores, demonstrando que o algoritmo determinou corretamente o ambiente em todas as três situações testadas.

Uma revisão da área de detecção de mudanças usando redes neurais foi feita por Marsland [2003]. No trabalho de Marsland et al. [2005] é apresentada uma aplicação dos conceitos de *GWR* com seleção de filtro de novidade e uso de habituação aplicados à tarefa de inspeção. Nesse trabalho, foram testados tanto o uso de sonar quanto de uma câmera monocromática validando assim a metodologia proposta. Um ponto em aberto nesse trabalho foi como utilizar a informação de mais de um sensor para detectar mudanças. A Figura 2.8a ilustra o robô utilizado. A Figura 2.8b, mostra o ambiente de corredor e portas utilizado nos experimentos, bem com a resposta da rede neural à passagem (com e sem aprendizado) em cada lugar do ambiente. Pode-se notar que após três passagens com aprendizado pelo ambiente de corredores, o robô "aprendeu" o local.

Em Vieira Neto & Nehmzow [2004] é proposta uma aplicação para detecção de mudanças, baseada no trabalho de Marsland et al. [2002], utilizando *GWR*. Porém, utilizando informação visual colorida, em vez de dados de dados de sonares ou de imagens monocromáticas. Para redução dimensional do problema é utilizado o modelo de saliência proposto por Itti et al. [1998], juntamente com um histograma de cores usando o espaço de cores *HSI*. Os resultados foram obtidos em duas fases em um ambiente restrito utilizando um robô móvel dotado de uma câmera. Na primeira fase foi feita a exploração do ambiente e, em uma segunda, a inspeção, após adicionar-se uma bola laranja ao ambiente. Os resultados mostraram a capacidade do sistema de identificar a mudança.



(a) (b)

Figura 2.8. Detecção de mudança usando *GWR* [Marsland et al., 2005]. Em (a) mostra o robô utilizado nos experimentos; em (b), o ambiente de testes e a resposta da rede neural à passagem do robô por tal ambiente são mostrados.

No trabalho de Vieira Neto & Nehmzow [2005b] é proposto outro método de classificação do dados de entrada baseado no *Incremental PCA* [Artac et al., 2002] e no uso do *Root Mean Square* (*RMS*) como métrica de distância (erro). A abordagem proposta utilizou a mesma entrada do trabalho anterior do autor, usando modelo de saliência sobre imagens coloridas. Os experimentos mostraram que ambos os métodos apresentam resultados similares, demonstrando a capacidade de detectar mudanças, com a vantagem do *PCA* incremental fazer uma redução dimensional significativa, apesar de demandar um número bem maior de amostras para o "aprendizado". Por outro lado, o *GWR* precisou de um número menor de amostras, porém com dados de dimensões bem maiores. No trabalho de Vieira Neto & Nehmzow [2005a] mostraram-se basicamente os mesmos resultados com mais experimentos para comparação entre os métodos, mantendo as mesmas considerações anteriores.

Em Vieira Neto & Nehmzow [2007] é testada uma nova abordagem para as técnicas de processamento de imagens com finalidade de detecção de mudanças. É utilizado o detector de Harris com seleção automática de escala para determinação da área de interesse em imagens [Lindeberg, 1998], em comparação com o modelo de saliência de Itti et al. [1998]. O uso do sistema de detecção automática do tamanho de janelas de interesse na cena demonstrou uma redução no número de características necessárias

para a detecção de mudança, porém a utilização de janelas de tamanho fixo obteve melhores resultados.

Em Vieira Neto & Nehmzow [2008] é feita uma explanação de todos os resultados obtidos com maior quantidade de detalhes, além de um detalhamento da estatística utilizada na validação dos resultados. Um aspecto importante dos trabalhos dos autores é que, devido ao fato de usar visão, as mudanças utilizadas nos experimentos são objetos com contraste de cores, como bolas laranjas e caixas verdes, o que facilita a tarefa do detecção. Embora tenham sido realizados experimentos com mudanças de cores semelhantes ao ambiente, com resultados satisfatórios.

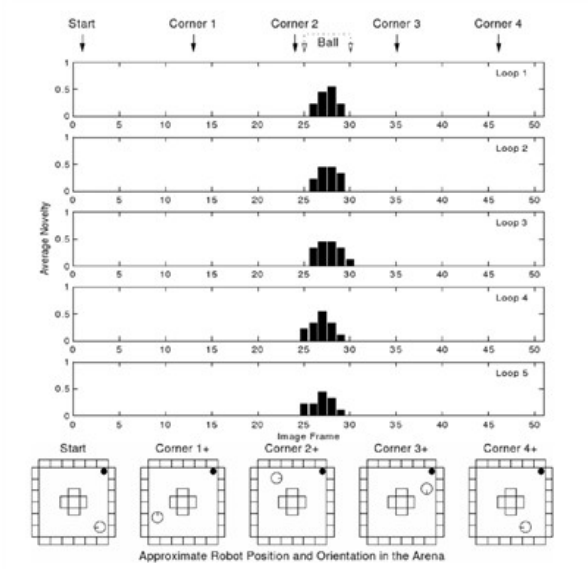
A Figura 2.9a mostra o robô utilizado, e na Figura 2.9b é exemplificada uma imagem do ambiente de teste utilizados nos experimentos do trabalho de Vieira Neto & Nehmzow [2008]. Após, um aprendizado inicial do ambiente retangular, mostrado na parte inferior da Figura 2.9c, sem a presença da "bola laranja", é mostrada a resposta da rede neural à passagem do robô no ambiente, agora com a presença da mudança (inserção da bola laranja).

Trabalhos de outros grupos de pesquisa também foram desenvolvidos na área, como o trabalho de Primdahl et al. [2005] onde é proposto um método para detectar objetos estacionários em um local bem definido e conhecido previamente, por meio de uma câmera presa em um veículo em movimento. A localização do veículo é realizada utilizando um *Extended Kalman Filter* (*EKF*) que combina dados de *GPS* diferencial com os fornecidos por uma central inercial e pela odometria. Os resultados mostraram a capacidade do sistema resolver o problema de paralaxe e de variação de iluminação, embora ainda seja bem restrito ao problema específico de um ambiente com restrições planares, além de necessitar uma seleção manual de regiões de interesse.

No trabalho de Ishikawa et al. [2005] a detecção de mudança foi estudada no contexto de reconhecimento de veículos em um estacionamento. Para tal, foi utilizada uma câmera omnidirecional, um receptor de *GPS* e um sensor inercial, juntamente com a técnica de *stereo from motion (SFM)* para obter um mapa 3D. Esse foi comparado com um modelo pré-construído do local. Embora tal método tenha se mostrado eficaz em ambiente real, com uma exatidão de centímetros, ele necessita de um ambiente com objetos texturados para que funcione corretamente.

No trabalho recente de Andreasson et al. [2007] é proposto um sistema para vigilância baseado no uso de *laser scanner* e câmera. Inicialmente, é criado um modelo de referência, a partir do qual são detectadas as mudanças. Os mapas são criados utilizando *3D Normal Distribution Transform* (*3D-NDT*), onde o espaço é discretizado em células, cada uma representada por uma distribuição Gaussiana. A mudança é determinada por meio da informação espacial e de variação de cor das células. O

(a) (b)



(c)

Figura 2.9. Detecção de mudança utilizando imagens coloridas [Vieira Neto & Nehmzow, 2008]. Em (a) é mostrado o robô utilizado, em (b) é mostrada uma imagem do ambiente de teste utilizados nos experimentos. Um resultado de detecção de mudança é mostrado em (c), com a inserção da bola laranja mostrada em (b).

sistema foi validado utilizando um robô Pioneer dotado de câmera, SICK Laser 2D e uma unidade de *pan-tilt*. Os resultados mostraram a capacidade de detecção de mudanças em ambientes pequenos, e a dificuldade do método de lidar com erros de localização. Embora o sistema tenha capacidade de detectar pequenas mudanças, estas devem ter uma significativa diferença de cor em relação ao ambiente.

A Figura 2.10 ilustra um resultado obtido pelo algoritmo proposto por Andreasson et al. [2007]. Da esquerda para direita, tem-se o mapa 3D, o mapa 3D com mudança e o mapa da probabilidade de mudança, ilustrada por tons de cinza que variam de escuras (menor probabilidade) a claras (maior probabilidade).

Um trabalho importante, que influenciou bastante o presente trabalho foi desen-

Figura 2.10. Detecção de mudança utilizando uma câmera e um *laser scanner*, com o mapa, à esquerda, representando o mapa 3D sem mudança, enquanto, ao centro, mostra o mapa com adição de uma cadeira. O mapa, à direita, representando a probabilidade de cada ponto do mapa, ao centro, ser efetivamente um mudança [Andreasson et al., 2007].

volvido por Amorim et al. [2008]. Tal trabalho faz um estudo sobre métodos para realizar detecção de mudanças em nuvem de pontos usando como métrica de distância o método *Earth Mover's Distance* (*EMD*) [Rubner et al., 1998]. São comparados os métodos *Principal Component Analysis* (*PCA*) [Jolliffe, 2002], *Gaussian Mixture Model* (*GMM*) [Paalanen et al., 2006] e de extração de planos. Os resultados foram todos obtidos por meio de simulações de nuvem de pontos 3D. Porém, foi adicionado aos dados ruído Gaussiano com diferentes covariâncias. Foram realizados 30 experimentos e analisou-se a resposta temporal, além da capacidade de detecção. A combinação *GMM-EMD* apresentou os melhores resultados em todos os quesitos, sendo o único método capaz de detectar mudanças em presença de ruído, embora não tenha sido realizada uma validação com dados reais nesse trabalho.

2.2.4 Resumo

Os trabalhos de detecção de mudanças são de difícil comparação entre si, pelo fato de usarem dados de entrada diferentes e pela maioria dos trabalhos apresentarem apenas resultados qualitativos. A Tabela 2.2 mostra os trabalhos mais importantes, bem como os dados utilizados para realizar a detecção, além de uma terceira coluna que mostra as restrições do método para uso no presente problema. A característica dos dados é a maior restrição ao uso no presente trabalho, pois demandariam adaptações dos métodos apresentados de forma a se adequarem ao uso de nuvem de pontos 3D.

2.3 Recuperação de Forma

Considerando o problema abordado neste trabalho, além da detecção de mudança, o conhecimento sobre a forma geométrica da mudança também é importante para

Tabela 2.2. Trabalhos em detecção de mudanças.

Método	Dados Utilizados	Principais Restrições
[Hotter et al., 1996], [Wren et al., 1997], [Tanjung & Lu, 2007], [Hwang et al., 2008]	Imagens	Características dos Dados
[Marsland et al., 2002], [Marsland et al., 2005]	Distância(Sonar), Imagens Monocromáticas	Características dos Dados
[Vieira Neto & Nehmzow, 2008]	Imagens	Características dos Dados
[Ishikawa et al., 2005]	Imagens Omnidirecional	Características dos Dados
[Primdahl et al., 2005]	Imagens	Características dos Dados
[Andreasson et al., 2007]	Mapas 3D Texturados	Informação de Textura
[Amorim et al., 2008]	Mapas 3D (Laser)	-
[Steinle et al., 1999]	Imagens de Profundidade	Detecção Manual
[Heller et al., 2001]	Mapas 3D (Par Estéreo)	Características dos Dados
[Vosselman et al., 2004b]	Mapas 3D Texturados	Informação de Textura
[Vogtle & Steinle, 2004]	Imagens de Profundidade	Características dos Dados
[Vu et al., 2004]	Imagens de Profundidade	Características dos Dados
[Girardeau-Montaut et al., 2005]	Mapas 3D (Laser)	Densidade dos Pontos, Robustez à Ruído

se obter uma representação da mudança. Tal representação fornece uma abstração entre a nuvem de pontos e o mundo real, além de permitir uma compressão de dados. Utilizando formas básicas pode-se facilmente obter modelos de alto nível semântico, representando uma gama enorme de elementos do ambiente. Modelos complexos são combinações destas formas mais simples [Schnabel et al., 2007], também conhecido como modelagem *part-level* [Jaklič et al., 2000]. Assim, primeiramente, será feita uma revisão sobre trabalhos que utilizam formas básicas (esferas, cilindros, planos, cubos, etc.) para representar conjuntos de pontos e, uma segunda, sobre trabalhos que utilizam superquádricas como formas primitivas.

2.3.1 Formas Básicas

A detecção de formas é uma tarefa comum em muitas áreas da geometria aplicada à Ciência da Computação, dentre elas destacam-se a computação gráfica e a visão computacional. Uma revisão da área foi realizada por Tangelder & Veltkamp [2004].

Ao longo das últimas décadas, diversos algoritmos e metodologias foram propostas para solucionar o problema de recuperação de formas. Ballard [1987] propôs uso de transformada de Hough para obter a forma 2D. Tal método apresenta alto custo computacional para a computação de formas mais complexas, devido ao modelo apresentar

muitos parâmetros. Khoshelham [2007] propôs uma extensão desse trabalho em nuvem de pontos 3D usando transformada de Hough, apresentando as mesmas características de elevado custo computacional da solução de Ballard [1987].

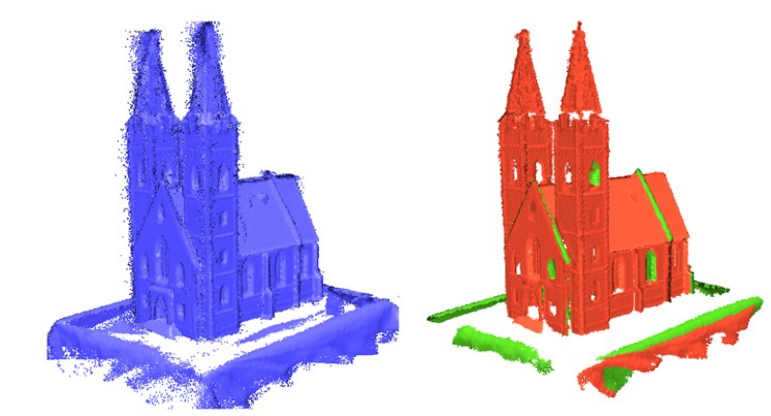
Outro trabalho utilizando a transformada de Hough em dados 3D foi desenvolvido por Vosselman et al. [2004b]. Ali é proposto o uso da transformada de Hough para determinação de planos, cilindros e esferas em nuvem de pontos. Em conjuntos de dados muito grandes, um pré-processamento para segmentar as formas foi utilizado por meio de crescimento de regiões. Os resultados mostrados foram satisfatórios. Porém, não há menção sobre o custo computacional do método.

Rabbani [2006] desenvolveu um trabalho que utiliza um processo de segmentação com posterior determinação de forma, usando como primitivas planos e cilindros, por meio de uma estimação da orientação e, posteriormente, da posição e tamanho. A limitação desse método se deve ao fato de que nem todos os pontos podem ser descritos por essas duas primitivas básicas, o que restringe bastante o método.

Nos últimos anos, alguns autores [Wahl et al., 2005; Schnabel et al., 2007] propuseram o uso de *Random Sample Consensus* (*RANSAC*) [Fischler & Bolles, 1981] para determinação de forma. O trabalho de Schnabel et al. [2007] representa um marco importante, pois consegue representar uma gama relativamente grande de formas básicas. As formas são: planos, esferas, cilindros, cones e toróides. O algoritmo *RANSAC* permite uma elevada robustez a ruído nos dados, além de ser muito eficiente. Os resultados mostram a capacidade do método de determinar formas eficientemente em nuvem de pontos 3D, mas assumindo dados completos (rotação de 360 graus em torno do objeto a ser modelado) e com boa densidade de pontos. A Figura 2.11 ilustra um resultado do algoritmo proposto por Schnabel et al. [2007]. A Figura 2.11a mostra a nuvem de pontos, enquanto a Figura 2.11b mostra o resultado obtido pela aplicação do algoritmo, com formas semelhantes representadas pelas mesmas cores.

Alguns trabalhos exploraram outras metodologias para extração da forma básicas, além de outros tipos de dados, como imagens de profundidade. No trabalho de Fitzgibbon et al. [1997] é explorado o conceito de crescimento de regiões para determinar superfícies em imagens de profundidade, regular e igualmente distribuída, nas quais se consegue determinar facilmente a topologia. Diversas outras metodologias foram aplicadas a imagens de profundidade. Porém, como neste trabalho é tratado o problema de nuvens de pontos não organizados, estas abordagens não são diretamente aplicáveis.

O trabalho Schnabel et al. [2008] propõe o reconhecimento de forma em nuvem de pontos 3D com uma metodologia diferente, no qual determina-se um grafo topológico a partir da nuvem de pontos e, posteriormente, divide esse grafo considerando



(a) (b)

Figura 2.11. Recuperação de forma utilizando formas básicas [Schnabel et al., 2007], em (a) mostra a nuvem de pontos 3D junto com a superfície gerada por meio de malhas. Em (b) é mostrado o resultado da aplicação do algoritmo com as formas semelhantes sendo representadas pelas mesmas cores.

algumas restrições (processo de segmentação). Para cada grafo gerado é determinada sua forma, considerando algumas primitivas conhecidas. Como principal característica desse método está a capacidade de se definirem as primitivas a serem buscadas, que podem ser úteis caso exista algum conhecimento prévio das formas disponíveis na cena. Contudo, no caso deste trabalho, tal abordagem não se mostra adequada por não se ter nenhum conhecimento prévio.

Na literatura, outras primitivas de formas foram propostas para determinação de formas complexas. Destacam-se os *cilindros generalizados* [Binford, 1971], obtidos por meio da varredura de um conjunto bidimensional ao longo de uma curva no espaço; os *geons* [Biederman, 1985] com inspiração na percepção humana, que são compostos de blocos sólidos inicialmente derivados de cilindros generalizados e suas relações. Outra forma que se destaca na literatura são as *superquádricas* [Barr, 1981]. Elas são uma representação com parametrização suficientemente simples e expressiva, permitindo uma ampla descrição de sólidos. Outras formas também foram propostas ao longo dos anos, mas com menor adoção pela comunidade do que as anteriormente citadas. Dentre elas se destacam: Hiperquádricas [Hanson, 1988], modelos de bolhas (*Blobs*) [Blinn, 1982], polinômios de quarto grau [Keren et al., 1994].

2.3.2 Superquádricas

Diversos trabalhos foram propostos ao longo dos anos objetivando descrever formas usando superquádricas, estas com alto grau de representatividade e com custo com-

putacional aceitável para sua determinação, apresentando uma ótima relação custo × beneficio.

O livro de Jaklič et al. [2000] faz uma revisão da área de superquádricas com foco nos trabalhos que usam como informação sensorial as imagens de profundidade, embora não se restrinja a esses tipos de dados.

Existem diversos métodos para obtenção de uma superquádrica a partir de um conjunto de pontos já segmentados, como o método analítico [Pentland, 1986], modelagem de distribuição de pontos (*PDM*) [Pilu & Fisher, 1999] e alguns métodos de minimização como *simulated annealing* [Yokoya et al., 1992] e algoritmos genéticos [Chevalier, 2004]. O método de minimização proposto por Solina & Bajcsy [1990] é o mais utilizado. Ele utiliza a minimização de quadrados da função implícita das superquádricas, com algumas restrições. O trabalho de Whaite & Ferrie [1991] propôs uma melhoria, definindo uma função de minimização um pouco diferente, utilizando a distância euclidiana radial. Essa, porém, é um pouco mais lenta para a computação que a original de Solina [Chevalier et al., 2003].

O processo de obtenção de uma superquádrica pode ser dividido em segmentação e recuperação da forma, embora em alguns trabalhos esses métodos não estejam claramente divididos em duas fases.

O trabalho de Pirrone & Chella [2003] propõe uma abordagem utilizando redes neurais para segmentação e ajuste de superquádricas. Primeiro, é realizada uma simplificação da nuvem de pontos utilizando *SOM* e posteriormente com o uso do método *K-means* é efetuada uma segmentação sobre o mapa gerado pelo *SOM*. Após essa etapa, os dois parâmetros de forma de superquádricas são determinados utilizando redes neurais, os outros parâmetros são determinados utilizando a proposta de Solina & Bajcsy [1990]. Essa abordagem se mostra muito promissora, embora o treinamento das redes neurais não possa ser efetuado *offline*, o que tornaria a abordagem extremamente rápida.

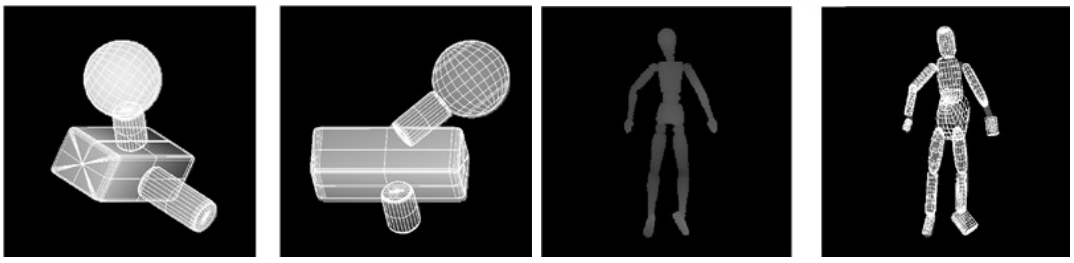
O trabalho de Katsoulas & Kosmopoulos [2006] propõe a detecção de objetos buscando por caixas modeladas como superquádricas parabolicamente deformáveis. Usando detector de quinas 3D para refinar e acelerar a busca, o método faz segmentação e ajuste da superquádricas. Porém, esse método é limitado ao reconhecimento de objetos retangulares.

Outro trabalho relevante foi proposto por Bruni et al. [2004], que utiliza a transformada Wavelet multi-escala [Simoncelli et al., 1992]. Utilizando-se de imagens de profundidade, é possível realizar a transformada com facilidade, o que não seria possível se o algoritmo utilizasse como entrada uma nuvem de pontos. Além disso, o método se baseia na informação do momento de inércia da região segmentada para determinar

a pose do objeto a ser ajustado pela superquádrica. Os resultados mostraram que o método, dada as restrições, recupera a forma, mas não é facilmente extensível para nuvem de pontos.

Leonardis et al. [1997] introduziram o paradigma *recover-and-segment* para segmentar a nuvem de pontos e recuperar forma. Esse método parte de um conjunto de sementes iniciais, efetuando iterativamente o crescimento de regiões e seleção das melhores superquádricas. Isto faz com que o método efetue a obtenção da forma de uma maneira eficaz. Contudo, devido à redundância causada pela inicialização aleatória das sementes, o método acaba sendo muito custoso. Além disso, ele se vale da informação topológica para determinar facilmente as regiões vizinhas de crescimento, pois considera como entrada as imagens de profundidade.

A Figura 2.12 ilustra dois exemplos de aplicação do algoritmo de Leonardis et al. [1997], para recuperção de superquádrica em imagens de profundidade. A Figura 2.12 mostra uma aplicação em duas imagens com objetos facilmente detectáveis com superquádricas, enquanto na Figura 2.12b apresenta um objeto complexo (forma humana) e as formas recuperadas, com partes de difícil representação.



(a) (b)

Figura 2.12. Recuperação de forma utilizando superquádricas com paradigma *recover-and-segment* [Leonardis et al., 1997]. Em (a) é mostrado um exemplo de recuperação usando o paradigma, em duas imagens de profundidade da mesma cena. Em é (b) mostrada a imagem de profundidade de um objeto complexo (forma humana), à esquerda, e das superquádricas obtidas pelo algoritmo, à direita.

Tal método foi desenvolvido e testado em detalhes no trabalho de Jaklič et al. [2000]. Em um trabalho subsequente são feitas algumas melhorias [Krivic & Solina, 2004], com a finalidade de reconhecimento de objetos. Considera-se no processo o conhecimento *a priori* dos objetos buscados na cena. Para o reconhecimento é utilizada a informação de conectividade entre as formas determinadas pelo algoritmo original de Leonardis et al. [1997]. Tal relação, devido à redundância, permite lidar com oclusão em nível moderado. A utilização de conhecimento *a priori* facilita muito o processo, embora nem sempre esteja disponível, como no caso deste trabalho.

O trabalho de Tao et al. [2004] apresentou uma melhoria em relação ao método de Leonardis et al. [1997], no qual utiliza-se seleção aleatória de pontos para acelerar o processo de ajuste das superquádricas. Essa abordagem é direcionada para a robótica, concentrando-se no problema de detecção de tubos em imagens de profundidade capturadas por sonares.

Outro trabalho interessante, ainda tratando com imagens de profundidade, foi proposto por Biegelbauer & Vincze [2007]. Nesse trabalho é proposto o uso do RANSAC para buscar os objetos na imagem modelados por superquádricas, ou seja, busca-se determinar a pose, visto que o modelo é conhecido *a priori*. Para tal, é utilizada uma busca hierárquica, na qual os pontos são subamostrados. Caso o objeto seja encontrado utiliza-se então, a escala mais fina para verificar a hipótese. O uso do *RANSAC* se mostra muito interessante pela eficiência e robustez. Além disso, a idéia de se usar duas escalas se mostra relevante, principalmente quando se lida com imagens muito grandes, embora o uso de conhecimento *a priori* não seja muito adequado ao presente trabalho. A Figura 2.13 mostra o resultado da aplicação do algoritmo em uma imagem de profundidade.

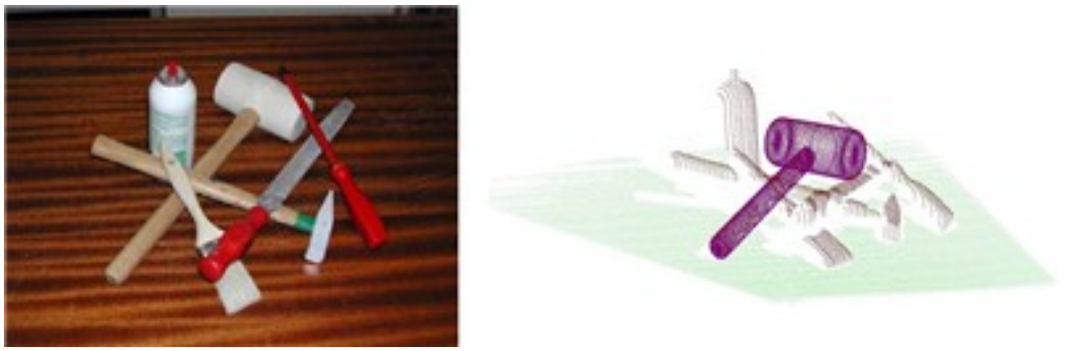


Figura 2.13. Recuperação de forma utilizando superquádrica com modelos conhecidos [Biegelbauer & Vincze, 2007], com uma imagem da cena, à esquerda, e a recuperação do martelo usando modelos de superquádricas, à direita.

Os trabalhos anteriormente citados tratam do problema de recuperação de formas em imagens de profundidade. Para o caso de nuvem de pontos não organizados, um tratamento diferente tem que ser dado. Assim, o trabalho de Chevalier et al. [2001] aparece como uma alternativa para esse caso. Nesse trabalho foi proposto o método *split-and-merge* para recuperar a forma de superquádricas em nuvem de pontos não estruturadas.

Em Chevalier et al. [2003], tal método foi apresentado em mais detalhes e comparado ao método *recover-and-segment* de Leonardis et al. [1997], sendo este último estendido para nuvem de pontos não estruturados. Porém, devido as características do

método, esse apresentou dificuldade de escolher boas sementes e selecionar as melhores superquádricas que deviam continuar crescendo. A Figura 2.14 mostra o resultado da aplicação dos algoritmos à nuvem de pontos do canto superior esquerdo. Na parte superior, o resultado do método da aplicação do *split-and-merge*, enquanto na parte inferior os resultados parciais e final da aplicação do algoritmo *recover-and-segment* adaptado de Leonardis et al. [1997].

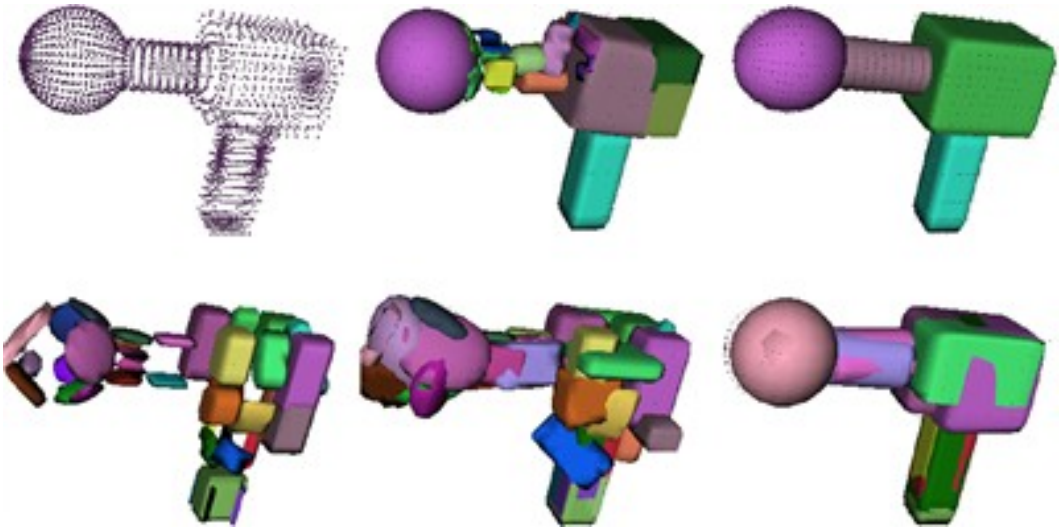


Figura 2.14. Recuperação de forma utilizando superquádrica em nuvem de pontos não estruturadas [Chevalier et al., 2003]. Na parte superior, à esquerda, a nuvem de pontos original. Ainda na parte superior o resultado da aplicação do *split-and-merge*, enquanto na parte inferior os resultados parciais e final da aplicação do método *recover-and-segment*.

No trabalho de Chevalier [2004] são mostrados maiores detalhes e resultados dos métodos, bem como um algoritmo para classificação de objetos baseados em grafos gerados a partir do algoritmo *split-and-merge*. Tal trabalho apresenta como principais vantagens a facilidade de implementação e ótima adaptabilidade à nuvem de pontos 3D. Além disso, é possível, com algum conhecimento *a priori*, tornar o algoritmo eficiente.

Zhang [2003] propõe um trabalho interessante no qual considera a recuperação de formas baseado em múltiplas nuvens de pontos 3D de diferentes vistas. Por meio dessas vistas, é construída uma representação única, a partir da qual é gerada a malha. Posteriormente, utilizando a informação de curvaturas e bordas com um algoritmo de crescimento de regiões é feita a segmentação. O ajuste das superquádricas é realizado utilizando o método de minimização da distância euclidiana radial. O trabalho embora apresente diversas contribuições importantes, não prioriza o desempenho, mas sim a qualidade da representação, o que difere do foco do presente trabalho.

O trabalho de Bierbaum et al. [2008] apresentou uma abordagem diferente, que

considera como entrada pontos 3D adquiridos por uma interface háptica. Basicamente a contribuição do trabalho foi utilizar, no processo de ajuste da superquádrica, algoritmos genéticos para minimizar as informações da normal e da localização do ponto adquiridas diretamente da interface háptica. Tal abordagem não se aplica ao problema apresentado neste trabalho, pois em nuvem de pontos adquirida por laser a informação do vetor normal a cada ponto não está disponível e o cálculo é apenas uma aproximação, além do que os algoritmos genéticos não apresentam um bom desempenho temporal.

2.3.3 Resumo

Os trabalhos de recuperação de forma são mostrados na Tabela 2.3. Foram comparados os modelos utilizados pelos trabalhos, bem como os dados de entrada para a obtenção da forma e completude desses dados, ou seja, se o método necessita de dados com uma representação dos dados de 360^0 em torno das formas a serem modeladas. Por fim, a eficiência e eficácia dos métodos também foram levadas em consideração. A eficácia leva em consideração a capacidade de recuperar a forma e a expressividade do modelo utilizado.

Tabela 2.3. Trabalhos em recuperação de formas.

Método	Modelo	Dados Utilizados	Completude	Eficiência	Eficácia
[Binford, 1971]	Cilindros Generalizados	Imagens de Profundidade	Não	±	±
[Fitzgibbon et al., 1997]	Superfícies	Imagens de Profundidade	Não	±	+
[Khoshelham, 2007]	Modelos 3D Abstratos	Imagens de Profundidade	Não	−	±
[Vosselman et al., 2004a]	Planos, Cilindros, Esferas	Nuvem de Pontos 3D	Não	−	+
[Rabbani, 2006]	Planos, Cilindros	Nuvem de Pontos 3D Texturadas	Sim	±	±
[Schnabel et al., 2007]	Planos, Cilindros, Esferas, Cones, Torus	Nuvem de Pontos 3D	Sim	+	+
[Schnabel et al., 2008]	Modelos de grafos *a priori*	Nuvem de Pontos 3D	Sim	+	+
[Leonardis et al., 1997]	Superquádricas	Imagens de Profundidade	Não	+	+
[Pirrone & Chella, 2003]	Superquádricas	Imagens de Profundidade	Não	−	±
[Zhang, 2003]	Superquádricas	Nuvem de Pontos 3D	Sim	−	+
[Bruni et al., 2004]	Superquádricas	Imagens de Profundidade	Não	±	±
[Chevalier, 2004]	Superquádricas	Nuvem de Pontos 3D	Sim	+	+
[Katsoulas & Kosmopoulos, 2006]	Superquádricas	Imagens de Profundidade	Não	+	−
[Biegelbauer & Vincze, 2007]	Superquádricas *a priori*	Imagens de Profundidade	Não	+	+
[Bierbaum et al., 2008]	Superquádricas	Nuvem de Pontos 3D (Interface Háptica)	Sim	−	+

Capítulo 3

Metodologia

Este capítulo descreve a metodologia concebida para tratar dos problemas da detecção, segmentação e representação da mudanças em nuvens de pontos tridimensionais não estruturadas, conhecidas *a priori*. Inicialmente, apresenta-se uma visão geral das partes que compõem a metodologia, visando introduzir o sistema e a importância de cada parte para o resultado final. Posteriormente, uma descrição detalhada é feita. A detecção e segmentação da mudança são realizadas por meio de misturas Gaussianas, enquanto que para a representação tridimensional da forma da mudança são propostos dois métodos: o primeiro, utiliza o espaço Gaussiano para determinar diretamente a forma, e o segundo, mais expressivo porém menos eficiente, utiliza o modelo de superquádricas, para permitir uma melhor representação da mudança.

3.1 Visão Geral

Considerando o problema a ser tratado, pode-se dividi-lo em duas partes principais: uma primeira, na qual se detecta e se segmenta a mudança em um mapa 3D fornecido como entrada, e uma etapa posterior cujo propósito é obter uma representação para a mudança por meio de formas geométricas.

A metodologia é composta pelas duas partes, como mostra a Figura 3.1. A **entrada do sistema** é definida como dois mapas 3D de um mesmo ambiente, em um mesmo sistema de coordenadas, porém, em instantes distintos. A **Parte I** é composta por dois módulos: um primeiro chamado de **simplificação da nuvem 3D**, no qual o objetivo é efetuar um pré processamento nos mapas 3D de forma a permitir a eficiência e a eficácia do sistema. Nesse passo é feita a simplificação da nuvem de pontos em uma abordagem que permite múltiplas escalas. Além disso, é efetuada a remoção de *outliers* e de pontos que representam o solo (*ground*) no mapa.

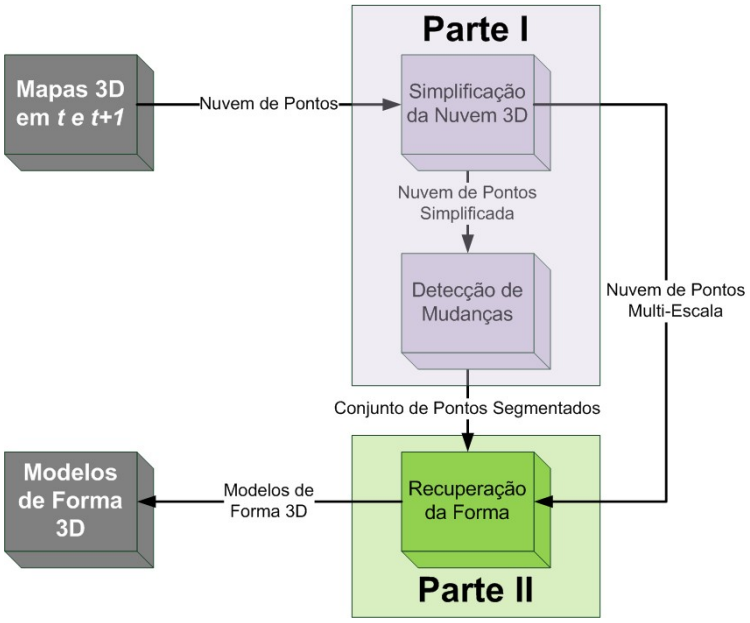


Figura 3.1. Visão geral da metodologia proposta.

Esse módulo tem por objetivo simplificar a nuvem de pontos de uma maneira "inteligente", ou seja, facilitando o desenvolvimento dos módulos posteriores, além de não aumentar o custo computacional global da metodologia. Uma abordagem que permite uma simplificação eficiente é proposta e detalhada na seção 3.2. Essa abordagem permite uma representação em árvore, na qual se consegue obter tanto uma representação mais refinada, quanto uma representação mais grosseira. Essa representação mais grosseira, deve ter um número mínimo de pontos que permita identificar tanto as mudanças, quanto reconhecer formas. Além disso, essa representação deve permitir um acesso rápido aos dados, nos diversos níveis da árvore. Assim, uma representação eficiente para esta árvore também é necessária.

Considerando os requisitos citados, o método de agrupamento hierárquico aplicado à nuvem de pontos (*hierarchical clustering*) foi utilizado neste trabalho [Pauly et al., 2002], com algumas mudanças para permitir a abordagem multi-escala eficiente. Tal método se baseia na partição binária do espaço utilizando a análise de covariância dos conjuntos de pontos. Utiliza-se abordagem *top-down*, considerando, inicialmente, a árvore com apenas um agrupamento englobando todos os pontos do mapa a ser simplificado, e iterativamente realizam-se divisões binárias desse conjunto até chegar à folha da árvore com apenas um ponto. Embora seja construída até sua raiz, a altura da árvore definirá a escala utilizada.

O segundo módulo da **Parte I** da metodologia é a **detecção de mudanças**, no qual os dois mapas 3D após simplificação serão utilizados. Tal módulo segmentará

ambos os mapas e, por fim, definirá quais segmentos representam mudança.

Assim, seguindo o estudo feito por Amorim et al. [2008], decidiu-se utilizar o Modelo de Misturas Gaussianas (*GMM*) como método de agrupamento dos pontos 3D, associado ao método *Earth Mover's Distance* (*EMD*) como métrica de distância, de forma a definir a existência ou não de mudança. Além disso, uma contribuição deste trabalho é permitir a definição de quais agrupamentos são mudanças, efetuando a segmentação.

Para isso, foi proposto o uso de um algoritmo usando técnica gulosa de forma a permitir a rápida segmentação das mudanças, de forma a permitir uma abordagem mais eficiente. Utilizou-se da informação probabilística dada pela mistura Gaussiana sobre os pontos 3D associados a cada função, de forma a determinar a relação topológica entre elas. Tal metodologia é descrita em maiores detalhes na seção 3.3.

O último módulo da metodologia que compõe a **Parte II** trata do problema de **recuperação da forma**. Embora a forma possa ser representada por meio de triangulação, ou descrição polinomial dos pontos, este não é o objetivo deste trabalho, mas sim, determinar uma descrição matemática da forma, por meio de **modelos de forma 3D** conhecidos, gerando uma redução significativa dos dados.

São propostos dois métodos neste trabalho: o primeiro utilizando formas básicas com o reconhecimento sendo efetuado diretamente no espaço N-dimensional definido pela *GMM* utilizado no processo de detecção de mudança, sendo esta uma das contribuições importantes deste trabalho. As formas básicas utilizadas foram o *cilindro*, a *esfera* e o *plano*, sendo que esse último não é uma forma 3D fechada, embora se possam definir essas como combinação de planos.

O segundo modelo proposto para reconhecimento de forma foi o modelo de superquádricas. Esse modelo permite uma expressividade maior, com uma complexidade computacional aceitável. Isso é possível pois se pode, eficientemente, ajustar uma superquádrica a um conjunto de pontos 3D. Além disso, foi proposto o uso de um método para ultrapassar as restrições impostas pela segmentação feita pela *GMM*, no qual se garante a segmentação apenas de "mínimos locais", o que não garante a melhor representação final para a mudança. Portanto, o método *split-and-merge*, juntamente com superquádricas foi utilizado baseado na proposta de Chevalier et al. [2003], buscando alcançar os "mínimos globais", gerando assim, uma representação mais adequada ao modelo de superquádricas. Como contribuição, tal abordagem foi adaptada a fim de aproveitar as características do processo de segmentação com *GMM*, que fornece uma boa inicialização e as relações topológicas entre os conjuntos de pontos que foram segmentados. Deste modo, o método é mais eficiente e robusto.

3.2 Simplificação da Nuvem de Pontos

A simplificação de pontos proposta neste trabalho baseou-se no trabalho de Pauly et al. [2002]. Além do processo de simplificação propriamente dito, este módulo é responsável por outras duas funções importantes. A primeira é tratar duas dificuldades encontradas no processo de mapeamento 3D utilizando *laser scanners*: a presença de *outliers* e a presença de pontos do solo nos dados. A segunda, define uma representação multi-escala, para melhorar a eficiência dos módulos posteriores.

Dentre os diversos algoritmos disponíveis na literatura, o mais adequado à presente metodologia foi o algoritmo de agrupamento hierárquico adaptado para nuvem de pontos por Pauly et al. [2002]. Isto se deve ao fato de apresentar como características: execução rápida, baixo consumo de memória e facilidade de representação em escala, devido à abordagem *top-down* utilizada.

Nas próximas seções, alguns conceitos pertinentes ao método serão apresentados, como o variação da superfície, baseada na análise de covariância. Além disso, o método propriamente dito será apresentado na seção 3.2.2 e posteriormente serão mostrados os métodos de remoção dos pontos do solo e dos *outliers*.

3.2.1 Conceitos Básicos para Simplificação

Alguns conceitos básicos baseados no trabalho de Pauly [2003] serão tratados a seguir, como o conceito de vizinhança e de análise de covariância, que estão diretamente associados, neste trabalho, com a estimação da normal e variação da superfície.

Os conceitos serão apresentados considerando que informação apresentada como entrada do sistema é uma *nuvem de pontos*, como descrito na Seção 2.1, $P = \{p_i | i \in [1; N] \wedge p_i \in \mathbb{R}^3\}$ onde $N = |P|$ e p_i é um ponto tridimensional, adquiridos por algum dispositivo, como *laser scanners*.

3.2.1.1 Vizinhança Local

Como a informação topológica não está disponível diretamente nos dados 3D adquiridos por *laser scanner*, é necessário se conhecer os pontos vizinhos, ou seja, a relação espacial entre os pontos. Considerando um ponto $p_i \in P$ e outro $p_j \in P$ com $i, j \in [1; N]$, a vizinhança local é definida como um conjunto de índices N_{p_j} para um ponto p_j, tal que $i \in N_{p_j}$, se p_i satisfaz certas condições de vizinhança. Essas condições podem ser definidas de forma que os pontos vizinhos representem adequadamente uma pequena superfície local em torno do ponto p_j.

O método dos ***k*-vizinhos mais próximos** é baseado na ordenação de todos os pontos da nuvem P de acordo com suas distâncias Euclidianas. Sendo Π uma lista dos índices de P ordenados pela distância Euclidiana a p_j, tal que $||p_{\Pi(1)} - p_j|| > 0$ e $||p_{\Pi(i)} - p_j|| \leq ||p_{\Pi(i+1)} - p_j||$, considerando $i \in [1; n-1]$. Assim, pode-se definir o conjunto de índices $N_{p_j}^k$ dos k-vizinhos mais próximos do ponto p_j como:

$$N_{p_j}^k = \{\Pi(1), \Pi(2), ..., \Pi(k)\}. \tag{3.1}$$

Tal conjunto $N_{p_j}^k$ define uma esfera centrada em p_j com raio $r_{p_j}^k = ||p_{\Pi(k)} - p_j||$, assim o ponto p_i está dentro da esfera se e, somente se, $i \in N_{p_j}^k$.

A Figura 3.2 mostra um exemplo de vizinhança local ao ponto p_j para pontos 2D, com a representação da vizinhança $N_{p_j}^k$ e do raio $r_{p_j}^k$, no caso $k = 8$.

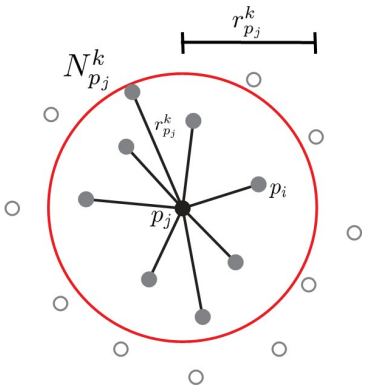


Figura 3.2. Exemplo de vizinhança local 2D utilizando algoritmo dos k-vizinhos mais próximos em um ponto p_j, com $k = 8$ [Pauly, 2003].

3.2.1.2 Análise de Covariância

Considerando as relações de vizinhança, algumas propriedades da superfície local no ponto p_j podem ser estimadas usando análise estatística da vizinhança. Uma ferramenta muito poderosa é a análise de componentes principais da matriz de covariância do ponto e sua vizinhança. Ela permite estimar, eficientemente, o vetor normal à superfície, além das variações da superfície e da normal.

Assim, considerando $\overline{p_j}$ o centróide da vizinhança de p_j definido por:

$$\overline{p_j} = \frac{1}{|N_{p_j}|} \sum_{i \in N_{p_j}} p_i, \tag{3.2}$$

pode-se definir a matriz de covariância $\mathbf{C}_{3\times3}$ para o ponto p_j como:

$$\mathbf{C} = \begin{bmatrix} p_{i_1} - \overline{p_j} \\ ... \\ p_{i_k} - \overline{p_j} \end{bmatrix}^T \cdot \begin{bmatrix} p_{i_1} - \overline{p_j} \\ ... \\ p_{i_k} - \overline{p_j} \end{bmatrix}, i \in N_{p_j}. \tag{3.3}$$

A matriz de covariância descreve as propriedades estatísticas de uma distribuição de pontos. Além disso, a matriz de covariância pode ser representada por meio de autovalores e autovetores [Steinbruch & Winterle, 1987], como mostra a Equação 3.4. Uma vez que, $\mathbf{C}$ é simétrica e semi-definida positiva, todos os autovalores λ são valores reais. Também, os autovetores $\mathbf{v}$ são ortogonais, correspondendo aos componentes principais do conjunto de pontos definido por N_{p_j} [Jolliffe, 2002].

Os valores de λ representam a variação da vizinhança do ponto p_j ao longo da direção dos autovetores, ou seja, representa o módulo do autovetor, um vetor unitário $\mathbf{v}$. Assim, como os dados são tridimensionais, tem-se três autovetores associados aos três autovalores, assumindo $\lambda_0 \leq \lambda_1 \leq \lambda_2$.

$$\mathbf{C} \cdot \mathbf{v} = \lambda \cdot \mathbf{v}. \tag{3.4}$$

A Figura 3.3 mostra uma ilustração 2D dos conceitos de análise de covariância. Uma elipse é gerada a partir da matriz de covariância $\mathbf{C}$, no caso 2×2, bem como os autovalores (λ_0 e λ_1) e autovetores ($\mathbf{v}_0$ e $\mathbf{v}_1$), ilustrando assim, a análise de covariância do ponto p_j e seus k-vizinhos mais próximos ($k = 10$). O centróide $\overline{p_j}$ também é ilustrado na figura.

3.2.1.3 Estimação da Normal

O plano tangente à superfície $T(x)$ pode ser determinando, utilizando-se os autovetores da matriz $\mathbf{C}$: $\mathbf{v}_1$ e $\mathbf{v}_2$, os autovetores associados aos maiores autovalores [Jolliffe, 2002]. Logo, como $\mathbf{v}_0$ é ortogonal a ambos, ele representa a normal ao ponto p_j que tem plano tangente $T(x)$. A Equação 3.5 mostra o plano tangente, assim, $\mathbf{v}_0$ aproxima a normal a superfície $\mathbf{n}_{p_j}$ em p_j. A Figura 3.3 ilustra o processo de determinação da normal em 2D e mostra o plano $T(x)$ (no caso uma reta), formada pelo autovetor $\mathbf{v}_1$.

$$T(x) \in \mathbb{R}^3 | (x - \overline{p_j}) \cdot \mathbf{v}_0 = 0. \tag{3.5}$$

Para estimar o sentido da normal, Pauly [2003] propõem a aplicação do método *Minimum Spanning Tree* (MST) na nuvem de pontos 3D como descrito em Hoppe et al. [1992]. O algoritmo inicia com o ponto extremo, ou seja, o ponto com maior coordenada

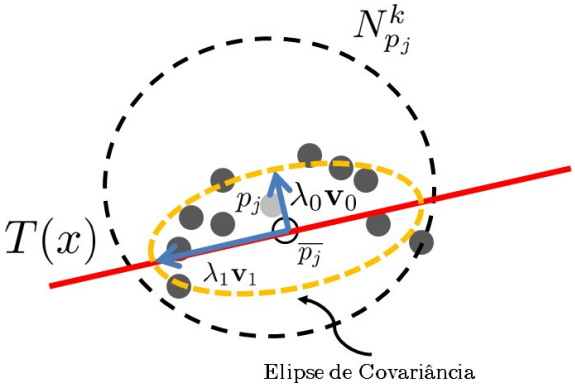


Figura 3.3. Exemplo de estimação 2D da tangente $T(x)$ (no caso, uma reta) utilizando a análise da covariância em uma k-vizinhança em torno de p_j, com $k = 10$.

Z (altura), e orienta a normal com relação ao centróide da nuvem de pontos. Assim, pontos adjacentes se valem da informação que o ângulo deve ser menor que $\frac{\pi}{2}$. Essa consideração nem sempre é valida, pois a nuvem de pontos pode não ser suficientemente densa, então um processo iterativo pode ser necessário para revalidar a orientação da normal.

3.2.1.4 Variação da Superfície

O autovalor λ_0 quantifica a variação ao longo da direção normal, estimando o quanto os pontos estão afastados do plano tangente, a **variação da superfície** ($\sigma_k(p_j)$) no ponto p_j com uma vizinhança de tamanho k pode ser definida como mostra a Equação 3.6 [Pauly et al., 2002].

$$\sigma_k(p_j) = \frac{\lambda_0}{\lambda_0 + \lambda_1 + \lambda_2}. \tag{3.6}$$

Quando a variação é zero, isto significa que os pontos são coplanares. Já quando $\sigma_k(p_j)$ tem valor máximo, ou seja, $\sigma_k(p_j) = \frac{1}{3}$, significa que os pontos estão igualmente distribuídos. Estes valores se devem ao fato de $\lambda_0 \leq \lambda_1 \leq \lambda_2$.

3.2.2 Algoritmo de Agrupamento Hierárquico com Representação Multi-Escala

O principal objetivo deste algoritmo é minimizar o número de pontos, ou seja, reduzir a complexidade da nuvem de pontos. Além disso, busca-se também preservar as características geométricas pertinentes à forma original da nuvem de pontos. Além disso, esse passo é de extrema importância para tornar o método eficiente, levando em consi-

deração que os próximos passos são dependentes diretamente do número de pontos da nuvem.

Um método eficiente para fazer tal operação foi proposto por Pauly et al. [2002], como já mencionado. Uma estratégia padrão para reduzir a complexidade de nuvem de pontos é utilizar o modelo do paralelepípedo envoltório (*bounding box*). Nesse modelo, os pontos são divididos em células e, então, todos os pontos dentro de uma mesma célula são reamostrados por um único ponto que representa esse conjunto, como o centróide, por exemplo. Entretanto, tal abordagem volumétrica tem algumas desvantagens, devido à utilização de células de tamanho fixo, não se adaptando às deformidades na nuvem de pontos. Além disso, é difícil a determinação ótima do tamanho das células utilizadas, bem como do número de células necessárias [Zou & Ye, 2007].

Uma abordagem que tenta ultrapassar esses problemas utiliza o conceito de *octrees*, que crescem conforme a necessidade [Girardeau-Montaut et al., 2005]. Porém, uma metodologia ainda mais interessante utiliza a informação de vizinhança para realizar agrupamentos evitando, assim, uma discretização da nuvem de pontos. Duas abordagens gerais para agrupamento foram descritas por Pauly et al. [2002]: o agrupamento incremental e agrupamento hierárquico.

A primeira, menos eficiente como mostraram os resultados de Pauly et al. [2002], utiliza uma abordagem *bottom-up* ou crescimento de regiões, ou seja, o agrupamento parte de um ponto aleatório e cresce em torno da vizinhança desse ponto, levando em consideração as restrições de número máximo de pontos por agrupamento e máxima variação de superfície (σ_{max}). Essa abordagem, além de não permitir a criação de uma árvore multi-escala diretamente, ainda apresenta o problema de *fragmentação*. Alguns pontos ficam sem agrupamento definido demandando um processo posterior para atribuir esses pontos a algum agrupamento. No caso, o agrupamento escolhido é o que possui centróide mais próximo. Essa limitação faz com que os parâmetros de máximo número de pontos e máxima variação da superfície possam ser ultrapassados, degradando a qualidade do método. A Figura 3.4 ilustra o problema de fragmentação em três agrupamentos distintos.

Por outro lado, um segundo método utilizando abordagem *top-down*, busca um modo alternativo de computar o conjunto de agrupamentos. O Algoritmo 1 mostra como o método foi desenvolvido neste trabalho. Basicamente, partindo-se de um único agrupamento, são efetuadas partições binárias recursivamente, construindo uma árvore.

As partições são feitas utilizando o plano definido pelo centróide do agrupamento $\overline{p}$ e pelo maior autovetor $\mathbf{v}_2$ associado a matriz de covariância $\mathbf{C}$ do agrupamento, calculada pela função $Calcula\_PlanoCorte(\overline{p}, \mathbf{C})$. Assim, a nuvem de pontos é sempre

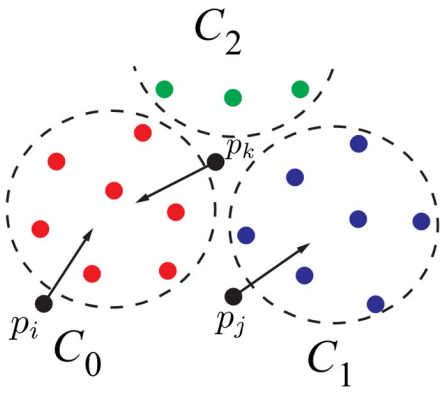


Figura 3.4. Exemplo de fragmentação causada pelo método de agrupamento incremental [Pauly et al., 2002], onde alguns pontos p_i, p_j e p_k precisam ser atribuídos, no caso, aos agrupamentos C_1,C_2 ou C_3.

Algoritmo 1 Calcula Recursivamente o Algoritmo de Agrupamento Hierárquico - $Calcula\_Arvore(Agrupamento\ A)$

```
1: [p̄, A] ← Calcula_Centroide_Elimina_Outliers(A)
2: C ← Calcula_Covariancia(A, p̄)
3: Plano ← Calcula_PlanoCorte(p̄, C)
4: FilhoDir ← ∅
5: FilhoEsq ← ∅
6: for all ponto ∈ A do
7:    if Esta_Direita(ponto, Plano) then
8:       FilhoDir ← FilhoDir ∪ ponto
9:    else
10:      FilhoEsq ← FilhoEsq ∪ ponto
11:   end if
12: end for
13: A ← Atualiza_Filhos(A, FilhoDir, FilhoEsq)
14: Calcula_Arvore(FilhoDir)
15: Calcula_Arvore(FilhoEsq)
```

dividida ao longo da direção de maior variação. Na abordagem original, a divisão acaba quando os objetivos são alcançados, ou seja, o valor do número de pontos máximo ou a variação da superfície é maior que limiares pré-definidos ($|P|_{max}$ e σ_{max}, respectivamente). Na abordagem proposta, a árvore é construída até que o agrupamento seja mínimo. Então, utilizando uma estrutura de dados que armazena os agrupamentos que respeitam alguns valores de $|P|_{max}$ e σ_{max}, pode-se definir as várias escalas desejadas.

Como mostra a Figura 3.5, com ilustração em 2D, o método constrói uma árvore binária, onde as folhas da árvore correspondem ao agrupamento na escala desejada. A Figura 3.5 mostra um exemplo 2D com apenas duas escalas. Além disso, são mostrados os planos de corte em vermelho, baseado no centróide e no elipsóide de covariância.

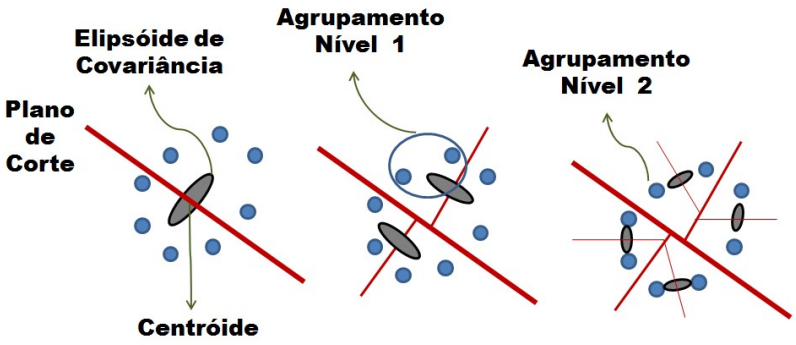


Figura 3.5. Esquema 2D do algoritmo de agrupamento hierárquico com apenas duas escalas.

Outra abordagem para realizar agrupamento hierárquico é possível utilizando filas de prioridades para ordenar as operações de divisão, conforme proposto no trabalho de Pauly [2003]. Embora essa abordagem permita um melhor controle sobre o tamanho da nuvem de pontos reamostrada, ela apresenta um significativo aumento no custo computacional.

A Figura 3.6 mostra o resultado do trabalho de Pauly [2003], efetuando divisão em partições binárias utilizando o algoritmo de agrupamento hierárquico em uma nuvem de pontos que representa uma face. Pode-se notar que a abordagem *top-down* utilizada gera agrupamentos maiores, que são diminuídos ao longo do processo. A figura mostra as superfícies geradas pelos agrupamentos. Por outro lado, na Figura 3.7, o resultado da simplificação efetuada na face, é representado por elipsóides gerados a partir da covariância dos agrupamentos e dos centróides. Pode-se notar que regiões com maior variação na superfície, e que necessitam um número maior de pontos para representar como olhos e boca, foram corretamente amostradas, assim como partes mais planas como a testa e pescoço foram representadas por poucos agrupamentos. A representação mostra os elipsóides geradas a partir do centróide e da covariância do agrupamento.

3.2.3 Remoção do Solo e de *Outliers*

A primeira dificuldade encontrada em dados adquiridos por *laser scanners* é a presença de *outliers*, devido a características do sensor. Uma das possíveis soluções é utilizar uma abordagem semelhante a proposta por Rusu et al. [2008], onde é proposto o uso da média e do desvio padrão sobre a vizinhança de um ponto de forma a eliminar os *outliers*.

No presente trabalho é efetuado algo semelhante, porém, aplicado diretamente ao processo de simplificação hierárquica, ou seja, na função

Figura 3.6. Ilustração do processo de simplificação hierárquica representada pela superfície gerada a partir da nuvem de pontos [Pauly, 2003], com agrupamentos maiores sendo diminuídos iterativamente, da esquerda para a direita.

Figura 3.7. Agrupamentos gerados após processo de simplificação em uma nuvem de pontos de alta densidade, com representação utilizando elipsóides [Pauly, 2003].

$Calcula\_Centroide\_Elimina\_Outliers(A)$ do Algoritmo 1. Nesta função, além de se computar o centróide do agrupamento, também é computado seu o desvio padrão (σ). Assim, pontos que estão fora do intervalo definido por $\overline{p} \pm \alpha \cdot \sigma$ são excluídos da nuvem de pontos. O valor de α utilizado neste trabalho e proposto por Rusu et al. [2008] foi $\alpha = 1$.

Para lidar com a presença nos dados de informação do solo (*ground*), que não é de interesse, é utilizada uma adaptação do algoritmo de remoção proposto por Lai & Fox [2009]. Nessa abordagem é feita uma reamostragem da nuvem de pontos em grades de tamanho fixo. Para cada célula da grade é ajustado um plano utilizando o algoritmo RANSAC [Fischler & Bolles, 1981]. Então, são escolhidas as células que ajustam planos com até 30^0 de inclinação em relação ao plano horizontal. Após este passo, o algoritmo RANSAC é utilizado, novamente, com a informação dessas células escolhidas, de forma a definir o melhor plano. Por fim, os pontos que melhor representavam esse

plano horizontal são excluídos.

No presente trabalho faz-se um processo parecido, porém, usa-se o conhecimento do menor autovetor ($\mathbf{v}_0$) de cada agrupamento, no nível mais grosseiro da árvore. Assim, buscam-se vetores aproximadamente verticais, ou seja, com variação de $\pm 15^0$. Além disso, é efetuada uma restrição de altura (coordenada Z), por meio de um valor Z_{max}. Assim, caso os dois requisitos sejam satisfeitos, esses agrupamentos são candidatos à exclusão. Sobre estes agrupamentos é efetuado o ajuste de um plano utilizando RANSAC, excluindo da árvore os agrupamentos que representam o solo.

A Figura 3.8 mostra um exemplo de uma imagem 3D obtida com um *laser scanner*, com a segmentação de solo. Os pontos segmentados do solo estão representado pela cor ciano, enquanto as outras partes dos dados são representados pela cor vermelha.

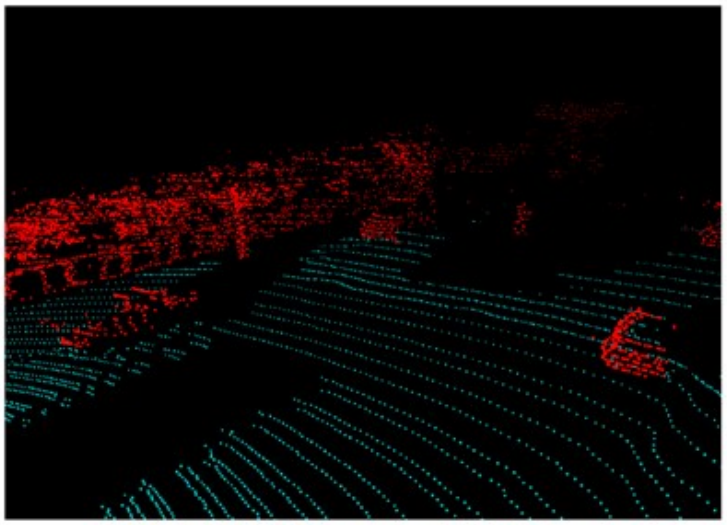


Figura 3.8. Segmentação de solo efetuada pelo Algoritmo RANSAC [Lai & Fox, 2009], os pontos na cor ciano representam a segmentação do solo, enquanto os pontos na cor vermelha representam as outras partes dos dados.

3.3 Detecção de Mudanças

A detecção de mudança é um problema importante em diversas áreas. No contexto da detecção de mudança em nuvem de pontos, a abordagem ingênua de efetuar a subtração ponto-a-ponto entre duas nuvem de pontos somente em condições muito controladas geraria uma resposta aceitável.

O trabalho de Amorim et al. [2008] propõe um estudo comparativo de algumas técnicas para compactar os dados da nuvem de pontos, que permitam obter uma representação simplificada e eficiente, para uma posterior comparação entre as nuvens

de pontos usando o algoritmo *Earth Mover's Distance* (*EMD*) [Rubner et al., 1998]. O método basicamente estabelece uma métrica de comparação entre distribuições. Tal método será descrito com maiores detalhes na seção 3.3.2.

Nesse trabalho, a técnica que se mostrou mais adequada para a modelagem das mudanças foi a *GMM*, que utiliza modelos de misturas finitas, no qual funções Gaussianas representam a distribuição dos pontos no espaço. Modelos de misturas Gaussianas são excelentes classificadores [Paalanen et al., 2006], além disso, atuam como método de agrupamento (*clusterização*).

No trabalho de Amorim et al. [2008], as misturas de Gaussianas foram comparadas com método de modelagem e compactação dos dados com outros dois métodos. O primeiro baseado em *Principal Component Analysis* (*PCA*), adaptado para utilizar *EMD* como métrica de comparação. O *PCA* é um método estatístico bem conhecido para identificar padrões em dados [Jolliffe, 2002]. Ele é muito útil quando se necessita analisar dados com alta dimensionalidade e grande número de variáveis. Ao identificar o padrão, é realizada uma compressão dos dados, sem excessiva perda de informação, reduzindo o número de variáveis. O método baseia-se no uso dos autovetores e autovalores associados aos dados, além do centróide, para realizar a redução dimensional. Contudo, devido à característica totalmente não linear dos dados, esse método apresentou alta flutuação na resposta e dificuldade de lidar com ruído [Amorim et al., 2008].

O outro método testado no trabalho de Amorim et al. [2008] foi a extração de planos, baseada em uma abordagem chamada *fuzzy planes* [Vandorpe et al., 1996]. Nesse método, os planos são extraídos utilizando uma abordagem semelhante ao RANSAC. Definindo assim, os pontos que "pertencem" ao plano, bem como os quatro parâmetros que definem o plano no espaço. Tal método, além da baixa eficácia, apresenta elevado custo computacional [Amorim et al., 2008].

Neste trabalho é proposto um algoritmo de forma a determinar a existência de mudança em nuvem de pontos utilizando a abordagem *GMM-EMD*, bem como segmentar essa mudança. Na Seção 3.3.3, esse método é detalhado.

3.3.1 Modelos de Misturas

Em reconhecimento de padrões são utilizados, basicamente, métodos estruturais, estatísticos e neurais. Os métodos estatísticos têm ganho importância por fornecerem informações sobre a confiabilidade da decisão tomada, além de permitirem incorporar ao método os modelos de custo e risco. As decisões tomadas por tais métodos têm bases

interpretáveis permitindo que diferentes resultados possam ser comparados [Paalanen et al., 2006].

Considerando os métodos estatísticos, os modelos de misturas podem ser uma ótima opção. Eles aproximam uma variedade grande de funções densidade de probabilidade (*Probability Density Function* (*PDF*)), e são muito atrativos quando os dados não podem ser representados por uma única função. Uma única distribuição básica normalmente é utilizada em modelos de mistura para permitir uma implementação prática, embora se possam utilizar diferentes tipos de funções de distribuição.

Uma *PDF* típica, devido à sua simplicidade e expressividade é a distribuição normal multivariável, também conhecida como *distribuição Gaussiana*. Outras *PDFs* também podem ser utilizadas, principalmente quando se tem algum conhecimento *a priori* sobre o fenômeno a ser modelado. Contudo, no caso da ausência de conhecimento, a função Gaussiana é uma opção bastante utilizada [Tong, 1990].

3.3.1.1 Distribuição Gaussiana

Uma distribuição Gaussiana ou distribuição normal, de uma variável aleatória $X \mapsto x$ pode ser definida pela Equação 3.7, considerando M a dimensionalidade dos dados (*e.g.* $M = 3$ neste trabalho) e $x \in \mathbb{R}^M$. Além disso, μ é o vetor de médias e $\mathbf{\Sigma}$ a matriz de covariância da variável aleatória X normalmente distribuída.

$$X \sim \mathcal{N}(x|\mu, \mathbf{\Sigma}) = \frac{1}{(2\pi)^{\frac{M}{2}}\sqrt{|\mathbf{\Sigma}|}} \exp\left[-\frac{1}{2}(x-\mu)^T\mathbf{\Sigma}^{-1}(x-\mu)\right]. \tag{3.7}$$

A Figura 3.9 ilustra uma distribuição Gaussiana bidimensional ($M = 2$). Uma superfície Gaussiana centrada em μ é mostrada em um plano superior, com várias elipses concêntricas representando a distribuição, em um plano inferior. Para o caso tridimensional, poderia ser representada por elipsóides concêntricos [Tong, 1990].

3.3.1.2 Modelos de Misturas Finitas

Embora uma única *PDF* pode modelar corretamente uma dada distribuição de dados, algumas restrições precisam ser feitas para que essa afirmativa seja verdadeira. Uma delas é que os dados devem ser compostos de uma única classe que varia suavemente ao redor da média. A outra é a suposição de unimodalidade que é uma restrição bastante forte, podendo causar, além de erros por limitação de representação, interpretações errôneas dos dados. Assim, a multimodalidade dada por modelos de mistura busca superar essas restrições.

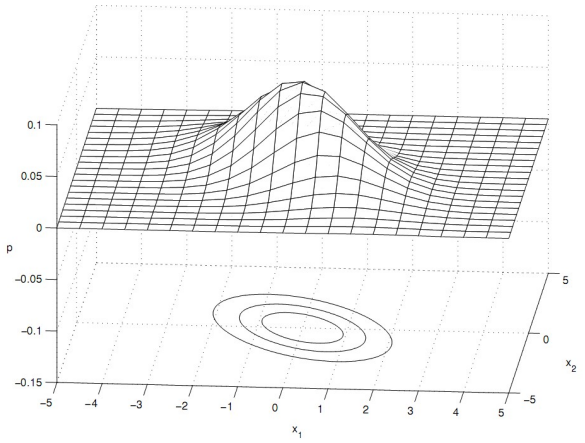


Figura 3.9. Distribuição Gaussiana bidimensional com, no plano inferior, elipses concêntricas de iguais probabilidades [Paalanen et al., 2006].

Para uma variável aleatória multimodal, cujos valores são gerados por diversas fontes independentes, em vez de uma única origem, o modelo de misturas finita pode ser utilizado para aproximar a real PDF. Se a forma Gaussiana é suficiente para uma única origem, então o modelo de misturas Gaussianas (GMM) pode ser uma boa aproximação [Paalanen et al., 2006].

A Figura 3.10 ilustra uma distribuição Gaussiana bidimensional (*i.e.* $M = 2$). Semelhante à Figura 3.9, porém apresentando uma modelagem dos dados por três Gaussianas, formando uma GMM.

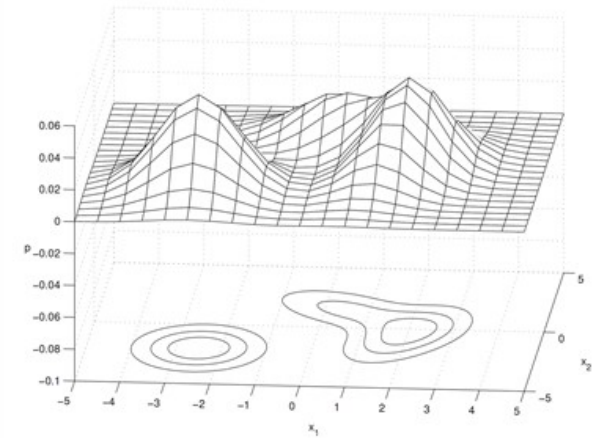


Figura 3.10. Mistura Gaussiana bidimensional formada por três Gaussianas com, no plano inferior, elipses concêntricas de iguais probabilidades [Paalanen et al., 2006].

3.3.1.3 Modelos de Misturas Gaussianas (GMM)

O modelo de misturas Gaussianas (*GMM*) é uma função densidade de probabilidade (*PDF*) dada por uma combinação linear de Gaussianas. Mais precisamente, a função é uma mistura de funções Gaussianas se ela tem a seguinte forma:

$$p(x|K,\theta) = \sum_{k=1}^{K} \pi_k \mathcal{N}(x|\mu_k, \mathbf{\Sigma}_k), \tag{3.8}$$

onde K é o número de *PDF* Gaussianas e π_k é o peso de cada uma delas na mistura. Esse peso pode ser interpretado como a probabilidade *a priori* do valor da variável aleatória ter sido gerado por uma origem k. Considerando, $0 \leq \pi_k \leq 1$ e $\sum_{k=1}^{K} \pi_k = 1$, a *GMM* pode ser definida por uma lista de parâmetros θ, que representa os parâmetros de cada Gaussiana juntamente com o peso, ou seja, $\theta = \{\pi_1, \mu_1, \mathbf{\Sigma}_1, ..., \pi_K, \mu_K, \mathbf{\Sigma}_K\}$.

A dificuldade na estimação das misturas de funções Gaussianas está em determinar θ, considerando que não se conhece os parâmetros das Gaussianas (π_k e $\theta_k = (\mu_k, \mathbf{\Sigma}_k)$), nem K (o número de funções Gaussianas da mistura).

Considerando que M é a dimensionalidade dos dados de entrada, é necessário estimar os parâmetros do modelo de mistura. Sendo K o número de componentes do modelo de mistura, o número total de parâmetros livres são: $K(0,5M^2+1,5M)+K-1$ [Paalanen et al., 2006]. Assim, tem-se um número grande de parâmetros a serem estimados, considerando ainda K é conhecido, o que nem sempre é verdadeiro.

Na literatura existem duas abordagens principais para estimação dos parâmetros do modelo de misturas: estimação baseada em máxima verossimilhança (*Maximum Likelihood* (*ML*)) e estimação Bayesiana. Embora existam diversos argumentos teóricos e metodológicos apoiando o uso da estimação Bayesiana, o uso de *ML* é mais simples e prático, sendo por isso mais utilizado [Paalanen et al., 2006].

Considerando $Y = \{y_1, ..., y_n, ..., y_N\}$ e $y_n \in \mathbb{R}^M$, o conjunto de amostras independentes. Neste trabalho, tais valores y_n representam, na prática, cada ponto da nuvem de pontos, *i.e.* Y é equivalente a P, e N o número de pontos da nuvem. Pode-se estimar diretamente a probabilidade $p(y_n|K,\theta)$ para cada K . Porém, para estimação do *ML* utiliza-se normalmente a função logarítmica da probabilidade pela facilidade de manipulação numérica. Assim temos:

$$\log p(y|K,\theta) = \sum_{n=1}^{N} \log p(y_n|K,\theta), \tag{3.9}$$

com o cálculo de $p(y_n|K,\theta)$ definido pela Equação 3.8. A estimação da máxima veros-

similhança (ML) busca determinar os parâmetros K e θ. Assim, tem-se:

$$\hat{\theta}_{ML} = \underset{\theta,K}{\operatorname{argmax}} \log p(y|K,\theta). \tag{3.10}$$

A estimação do ML, considerando a Equação 3.10, não determina corretamente o valor de K, pois a probabilidade sempre melhora quando se escolhe um número maior de classes. Esse problema vem sendo estudado ao longo dos anos, sendo conhecido como problema de identificação de ordem (*order identification*). Os métodos geralmente precisam da adição de termos de penalização para estimar o valor correto de K [Figueiredo & Jain, 2002]. Esse termo de penalização é utilizando na probabilidade de forma a evitar o uso de modelos de alta ordem, ou seja, com valores grandes de K.

Um método largamente utilizado para identificação de ordem de modelos é o *Minimum Description Length* (MDL), proposto por Rissanen [1983]. Tal método busca encontrar a ordem do modelo que minimiza o número de bits que podem ser necessários para codificar tanto os dados amostrados (Y), quanto os parâmetros do modelo (θ). Uma expressão aproximada desenvolvida por Rissanen com objetivo de buscar a minimização é mostrada na Equação 3.11, onde $L = \frac{K}{2}(M^2 + 3M + 2) - 1$ [Bouman, 1997].

$$MDL(K,\theta) = -\log p(y|K,\theta) + \frac{1}{2} L \log(N \cdot M). \tag{3.11}$$

Embora MDL seja um estimador consistente para um grande número de problemas conhecidos, o MDL não é um estimador consistente para o caso da estimação da GMM. Porém, ele é largamente utilizado devido a qualidade dos resultados gerados [Bouman, 1997]. Um estimador consistente foi proposto por Redner & Walker [1984], mas devido ao elevado custo computacional é pouco utilizado.

A determinação do MDL é difícil por diversas razões: o termo logarítmico faz com que a otimização direta de π e θ não seja trivial, além disso, a minimização de K também é complexa, visto que para sua determinação é necessário o conhecimento de π e θ. Assim, caso sejam conhecidas as classes de cada dado amostrado, y_n, a determinação dos valores de cada componente da GMM, média e covariância, é trivial.

Para a determinação da classe de cada dado amostrado é utilizado o algoritmo *Expectation-Maximization* (EM) [Dempster et al., 1977]. O algoritmo tem por objetivo solucionar problemas nos quais não se conhece todas as informações necessárias para a solução. A presente abordagem se baseou no trabalho de Bouman [1997].

O algoritmo é composto de duas fases:

- *E-step* - Nesse passo, os dados que faltam são estimados utilizando os dados observados e o estado atual dos parâmetros do modelo.

- *M-step* - A função de máxima verossimilhança (*ML*) é maximizada, considerando que os dados que faltam são conhecidos.

O algoritmo *EM*, para o caso da estimação de *GMM*, busca classificar os dados y_n em classes, ou Gaussianas, e, posteriormente, re-estimar os valores de cada classe. Por meio da regra de Bayes é computada a probabilidade de um ponto y_n pertencer à classe k. Considerando $\theta^{(i)}$ o valor de θ no passo iterativo i do algoritmo, e conhecido nesse momento da execução do algoritmo. A probabilidade calculada no *E-step* é dada por:

$$p(k|y_n, \theta^{(i)}) = \frac{\pi_k \cdot \mathcal{N}(y_n|\theta_k^{(i)})}{\sum_{l=1}^{K} \pi_l \cdot \mathcal{N}(y_n|\theta_l^{(i)})}. \tag{3.12}$$

Conhecidas essas probabilidades, pode-se estimar os parâmetros θ e π. As equações abaixo mostram como cada valor é estimado na etapa de maximização (*M-step*). Inicialmente um parâmetro de normalização $\overline{N}_k$ é estimado, com a posterior estimação de $\overline{\pi}_k, \overline{\mu}_k, \overline{\mathbf{\Sigma}_k}$.

$$\overline{N}_k = \sum_{n=1}^{N} p(k|y_n, \theta^{(i)}), \tag{3.13}$$

$$\overline{\pi}_k = \frac{\overline{N}_k}{N}, \tag{3.14}$$

$$\overline{\mu}_k = \frac{1}{\overline{N}_k} \sum_{n=1}^{N} y_n p(k|y_n, \theta^{(i)}), \tag{3.15}$$

$$\overline{\mathbf{\Sigma}}_k = \frac{1}{\overline{N}_k} \sum_{n=1}^{N} (y_n - \overline{\mu}_k) \cdot (y_n - \overline{\mu}_k)^T \, p(k|y_n, \theta^{(i)}). \tag{3.16}$$

A partir disso, podemos definir a equação de atualização do algoritmo *EM*. A Equação 3.17 define a função $Q(\theta; \theta^{(i)})$, utilizando o critério *MDL*, onde Y e X representam as variáveis aleatórias conhecidas e desconhecidas, respectivamente.

$$Q(\theta; \theta^{(i)}) = E[\log p(Y, X|\theta)|Y, \theta^{(i)}] - \frac{1}{2} L \log(N \cdot M). \tag{3.17}$$

Uma relação fundamental para o algoritmo *EM* é que $\{\forall\theta \rightarrow MDL(K, \theta) - MDL(K, \theta^{(i)}) < Q(\theta^{(i)}; \theta^{(i)}) - Q(\theta; \theta^{(i)})\}$. Assim, qualquer valor de θ que aumenta o valor de $Q(\theta; \theta^{(i)})$ reduz o critério *MDL*. O algoritmo *EM* busca, iterativamente, otimizar o valor de θ para alcançar o mínimo local do *MDL*.

O *M-step* é o passo que visa otimizar o valor de θ por meio da maximização de $Q(\theta; \theta^{(i)})$. Considerando conhecida as probabilidades do passo *E-step*, calculada pela

Equação 3.12, pode-se estimar o *M-step*. Deste modo, a otimização de θ é da por:

$$(\pi^{(i+1)}, \mu^{(i+1)}, \mathbf{\Sigma}^{(i+1)}) = \underset{(\pi,\mu,\mathbf{\Sigma})}{\operatorname{argmax}} Q(\theta; \theta^{(i)}). \tag{3.18}$$

Considerando o *M-step* para o caso do presente trabalho, estima-se os valores de $\overline{\pi}_k, \overline{\mu}_k, \overline{\mathbf{\Sigma}_k}$ utilizando as Equações 3.14, 3.15 e 3.16, respectivamente. Assim, iterativamente estimando os passos *M-step* e o *E-step*, é possível chegar a um ótimo local do critério *MDL*. Porém, não é possível determinar o valor ótimo de K, ou seja, a ordem do modelo.

Para diminuir o número de Gaussianas K, sem a necessidade de recalcular novamente todos os passos do algoritmo *EM*, uma abordagem que realiza a união de Gaussianas de maneira mais eficiente é utilizada e descrita a seguir.

Considerando duas Gaussianas l e m pertencentes ao conjunto θ de tamanho k, o objetivo é encontrar as duas que minimizem o critério *MDL*, ou seja:

$$(l^*, m^*) = \underset{(l,m)}{\operatorname{argmax}}\, d(l, m) \rightarrow \{\forall\, l, m | l, m \in \theta \wedge l \neq m\}. \tag{3.19}$$

O problema restante é a definição da função de distância $d(l, m)$, além de como realizar a união das duas classes l e m. Para a união dos conjuntos são utilizadas as seguintes equações, que definem os três parâmetros de cada classe:

$$\pi_{(l,m)} = \overline{\pi}_l + \overline{\pi}_m, \tag{3.20}$$

$$\mu_{(l,m)} = \frac{\overline{\pi}_l \overline{\mu}_l + \overline{\pi}_m \overline{\mu}_m}{\overline{\pi}_l + \overline{\pi}_m}, \tag{3.21}$$

$$\mathbf{\Sigma}_{(l,m)} = \frac{\overline{\pi}_l(\overline{\mathbf{\Sigma}}_l + (\overline{\mu}_l - \mu_{(l,m)})(\overline{\mu}_l - \mu_{(l,m)})^T) + \overline{\pi}_m(\overline{\mathbf{\Sigma}}_m + (\overline{\mu}_m - \mu_{(l,m)})(\overline{\mu}_m - \mu_{(l,m)})^T)}{\overline{\pi}_l + \overline{\pi}_m}. \tag{3.22}$$

A função de distância também pode ser definida pela função de atualização, considerando θ^* como o conjunto ótimo de parâmetros sem restrições, e $\theta^*_{(l,m)}$ o conjunto com a restrição de uma união. Assim, tem-se a Equação 3.23 que define a distância a ser otimizada para determinar qual o melhor par de Gaussianas a ser unido, onde N é o número de amostras (pontos). Tal equação é baseada na premissa de que $d(l, m) = Q(\theta^*; \theta^{(i)}) - Q(\theta^*_{(l,m)}; \theta^{(i)})$ (Mais detalhes sobre a dedução e sobre as propriedades da

função distância podem ser encontradas em Bouman [1997]) :

$$d(l,m) = \frac{N\overline{\pi}_l}{2}\log\left(\frac{|\mathbf{\Sigma}_{(l,m)}|}{|\overline{R}_l|}\right) + \frac{N\overline{\pi}_m}{2}\log\left(\frac{|\mathbf{\Sigma}_{(l,m)}|}{|\overline{R}_m|}\right). \tag{3.23}$$

Um último aspecto importante e crítico para a determinação do modelo de *GMM* deve ser tratado: considerando a nuvem de pontos com número máximo de Gaussianas na *GMM* igual a K_0, é necessária a determinação do $\theta^{(0)}$, ou seja, do conjunto inicial de parâmetros da *GMM*. Embora a inicialização seja crítica, a presente abordagem difere de trabalhos que utilizam algoritmos de agrupamento para a inicialização como o *K-means* [Figueiredo & Jain, 2000]. Neste trabalho, foi utilizado um modelo simplificado: A inicialização é baseada em geração aleatória com K_0 conhecido. As equações abaixo definem-se os valores de $\theta^{(0)}$ de uma forma simples e eficiente:

$$\pi_k^{(1)} = \frac{1}{K_0}, \tag{3.24}$$

$$\mu_k^{(1)} = y_n \rightarrow n = \lfloor (k-1)(N-1)/(K_0-1) \rfloor + 1, \tag{3.25}$$

$$\mathbf{\Sigma}_k^{(1)} = \frac{1}{N}\sum_{n=1}^{N} y_n y_n^T. \tag{3.26}$$

A descrição completa do algoritmo utilizado para determinação da *GMM* da nuvem de pontos de cada mapa é feita no Algoritmo 2. Uma consideração importante sobre as entradas do algoritmo: o valor de K_0 deve ser conhecido, ou seja, o número máximo de classes que a *GMM* pode possuir, além da nuvem de pontos simplificada Y. Essa última é definida para o caso do presente trabalho, pelo nível mais grosseiro da árvore determinada pelo Algoritmo 3.2.2.

A função $Inicializa\_\theta$, utiliza-se das Equações 3.24, 3.25 e 3.26. A função $Calcula\_EM(k, Y, \theta^{(i)})$ efetua a estimação do modelo, considerando a nuvem de pontos Y e o número de Gaussianas k. Conforme descrito, essa última função utiliza, basicamente, a Equação 3.12 no *E-step*, e a Equação 3.18 no *M-step*. A última função, $Reduz\_Ordem(\theta^{(i)})$, implementa a Equação 3.19, que define quais classes serão unidas, gerando uma representação mais compacta.

A Figura 3.11 mostra a nuvem de pontos e o resultado da aplicação do método de estimação de *GMM* para o caso de três mapas 3D simulados, sem a presença de ruído. A Figura 3.11a mostra apenas um corredor, bem como a *GMM* resultante, à direita. A Figura 3.11b ilustra um plano paralelo à parede do corredor e, na Figura 3.11c é mostrada uma esfera e um plano ao centro, além do corredor. As Gaussianas que compõem as *GMMs* foram representadas por elipsóide com centro na média μ e pelos

Algoritmo 2 Calcula Mistura Gaussiana da Nuvem de Pontos Simplificada - $Calcula\_GMM(Nuvem\,Simplificada\,Y, Inteiro\,K_0)$

1: $\theta^{(0)} \leftarrow Inicializa\_\theta(K_0)$ {Equações 3.24, 3.25 e 3.26}
2: $k = K_0$
3: $[MDL^{(1)}, \theta^{(1)}] \leftarrow Calcula\_EM(k, Y, \theta^{(0)})$ {Equações 3.12 (*E-Step*), 3.14, 3.15 e 3.16 (*M-Step*)}
4: $MDL^* \leftarrow MDL^{(1)}$
5: $\theta^* \leftarrow \theta^{(1)}$
6: $K^* \leftarrow k$
7: $i = 1$
8: **while** $(k > 0)$ **do**
9: $\quad \theta^{(i)}_{(l,m)} \leftarrow Reduz\_Ordem(\theta^{(i)})$ {Equação 3.19}
10: $\quad k \leftarrow k - 1$
11: $\quad [MDL^{(i+1)}, \theta^{(i+1)}] \leftarrow Calcula\_EM(k, Y, \theta^{(i)}_{(l,m)})$ {Equações 3.12 (*E-Step*), 3.14, 3.15 e 3.16 (*M-Step*)}
12: $\quad i \leftarrow i + 1$
13: $\quad$ **if** $(MDL^* > MDL^{(i)})$ **then**
14: $\quad\quad MDL^* \leftarrow MDL^{(i)}$
15: $\quad\quad \theta^* \leftarrow \theta^{(i)}$
16: $\quad\quad K^* \leftarrow k$
17: $\quad$ **end if**
18: **end while**

autovalores e autovetores da matriz de covariância $\mathbf{\Sigma}$. Pode-se notar que o número de classes determinado iterativamente permitiu uma representação adequada da nuvem de pontos.

3.3.2 *Earth Mover's Distance (EMD)*

A *EMD* foi proposta por Rubner et al. [1998], como uma "distância" [1] entre duas distribuições. A *EMD* é baseada na solução de um tipo particular de problema de transporte, muito conhecido na área de otimização linear. A idéia básica é que dadas duas distribuições, pode-se imaginar que uma corresponde a um volume de terra, enquanto a outra a um conjunto de buracos. Assume-se que a quantidade de terra é suficiente para tapar todos buracos, bem como é possível levar a terra de um lugar para outro. A *EMD* representa o montante de trabalho necessário para tapar todos os "buracos" utilizando a "terra" disponível [Rubner et al., 1998].

Para definir a *EMD* em $\mathbb{R}^n$, é preciso compreender o conceito de pontos com pesos. Trata-se de um par (w, x) com $x \in \mathbb{R}^n$ e $w \in \mathbb{R}^1_+$, sendo o peso do ponto x. Assim,

[1] A *EMD* somente é uma distância real quando a soma total dos pesos de cada uma das duas distribuições forem iguais [Rubner et al., 1998].

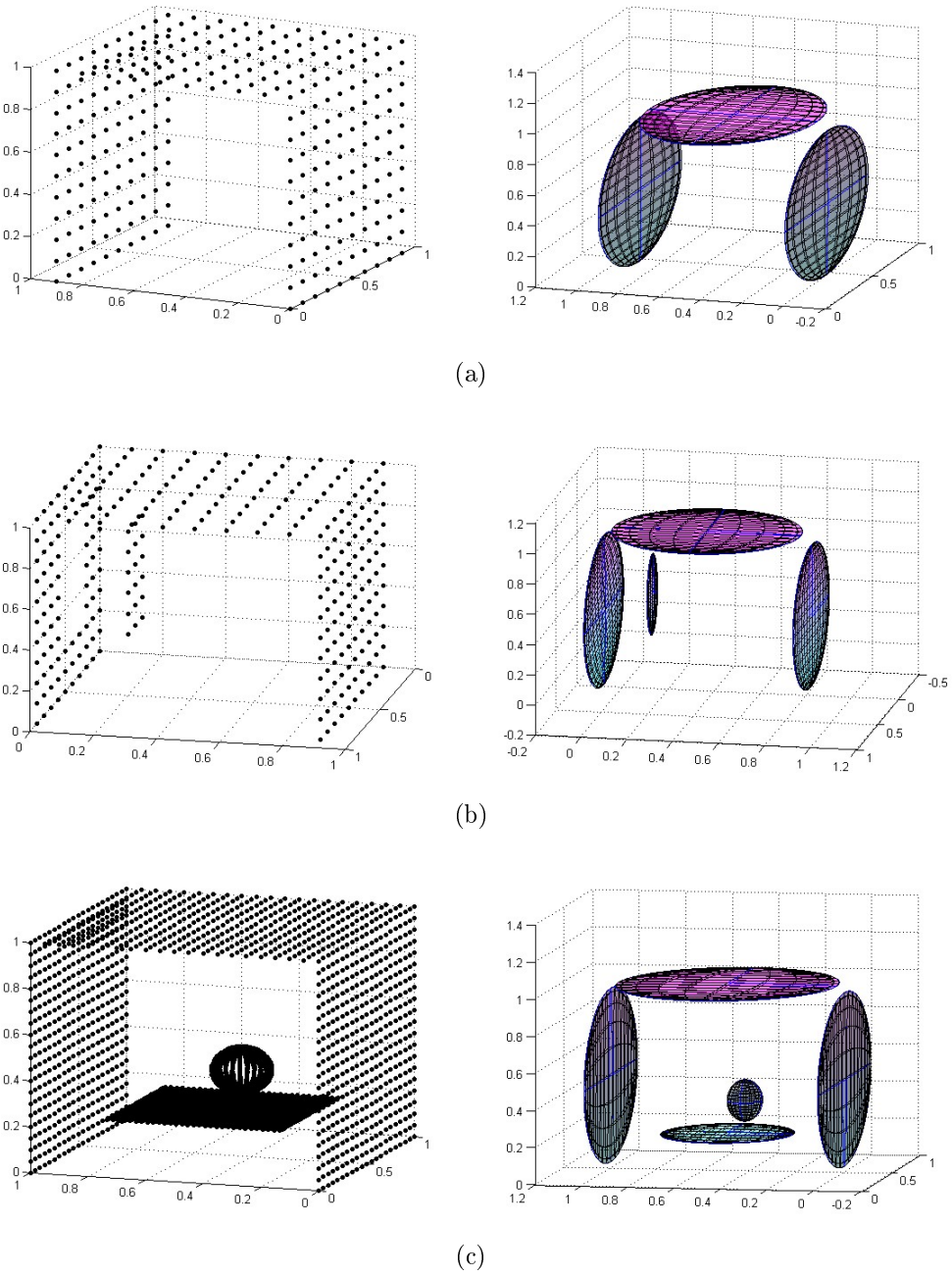


Figura 3.11. Nuvem de pontos 3D simulados, à esquerda, com suas respectivas representações por *GMM*, à direita. Em (a) é mostrado apenas um corredor, em (b) é ilustrado um plano paralelo ao corredor. Em (c), uma esfera e um plano são adicionados ao centro do corredor. As Gaussianas das *GMMs* foram representadas por elipsóides, por meio da média e matriz de covariância.

considerando duas distribuições de pontos com pesos $A = \{(w_1, x_1), ...(w_m, x_m)\}$ e $B = \{(u_1, y_1), ...(u_n, y_n)\}$, com $m \leq n$, tem-se o total dos pesos de A e B, respectivamente,

representados na Equação 3.27.

$$W = \sum_{i=1}^{m} w_i \;\; e \;\; U = \sum_{j=1}^{n} u_j. \tag{3.27}$$

Assim, pode-se definir a *EMD* como:

$$EMD(A, B) = \min_{F \in \mathcal{F}(\mathcal{A},\mathcal{B})} \frac{\sum_{i=1}^{m} \sum_{j=1}^{n} f_{ij} d_{ij}}{\min\{W, U\}}, \tag{3.28}$$

onde d_{ij} é alguma distância entre x_i e y_j, por exemplo, a distância Euclidiana em $\mathbb{R}^n$. Além disso, $F = \{f_{ij}\} \in \mathcal{F}(A, B)$ é o conjunto de todas as possibilidades de fluxo entre A e B definida pelas restrições:

1. $f_{ij} \geq 0 \rightarrow 1 \leq i \leq m \; e \; 1 \leq j \leq n$;
2. $\sum_{j=1}^{n} f_{ij} \leq w_i \rightarrow 1 \leq i \leq m$;
3. $\sum_{i=1}^{m} f_{ij} \leq u_j \rightarrow 1 \leq j \leq n$;
4. $\sum_{i=1}^{m} \sum_{j=1}^{n} f_{ij} = \min\{W, U\}$.

O peso w de um ponto no conjunto A está relacionado com a quantidade de terra, numa posição x do espaço, e o peso de um ponto $y \in B$ está relacionado com o tamanho do buraco numa posição y do espaço. A quantidade f_{ij} representa a quantidade de terra que será movida entre o i-ésimo ponto (terra) para o j-ésimo ponto (buraco). Então, as condições acima representam: (1) A inexistência de quantidades negativas de terra; as condições (2) e (3) representam que cada buraco não pode ser preenchido por uma quantidade de terra maior que sua capacidade, nem que se pode retirar mais terra do que existente em um dado local; a condição (4) significa que, ao final, todos os buracos terão que ser preenchidos ou toda terra utilizada.

Considerando o problema de se estimar a distância entre duas misturas Gaussianas (*GMM*), a *EMD* aparece como uma alternativa para identificar a presença de mudanças. Como definido por Amorim et al. [2008], a aplicação da *EMD* entre duas *GMM* é trivial, pois considera-se diretamente π_k da distribuição como peso w, o ponto x é representado pelos dados μ_k e $\mathbf{\Sigma}_k$, ou seja, θ_k. Deste modo, define-se d_{GMM} entre duas *GMM*: $\gamma = \{\pi_1^1, \mu_1^1, \mathbf{\Sigma}_1^1, ..., \pi_{K_\gamma}^1, \mu_{K_\gamma}^1, \mathbf{\Sigma}_{K_1}^1\}$ de tamanho K_γ e $\theta = \{\pi_1^2, \mu_1^2, \mathbf{\Sigma}_1^2, ..., \pi_{K_\theta}^2, \mu_{K_\theta}^2, \mathbf{\Sigma}_{K_\theta}^2\}$ de tamanho K_θ como:

$$d_{GMM} = EMD(\gamma, \theta). \tag{3.29}$$

3.3.3 Algoritmo para Detecção e Segmentação de Mudanças

Considerando como entrada do método dois mapas 3D em períodos temporais distintos, representando o mesmo ambiente e no mesmo sistema de coordenadas: a idéia por trás do algoritmo de detecção de mudanças proposto neste trabalho pode ser divida em dois passos: um primeiro, no qual se determina a existência de mudança e um outro, no qual é feita a segmentação do conjunto de pontos que representa a mudança. A Figura 3.12 ilustra essa idéia.

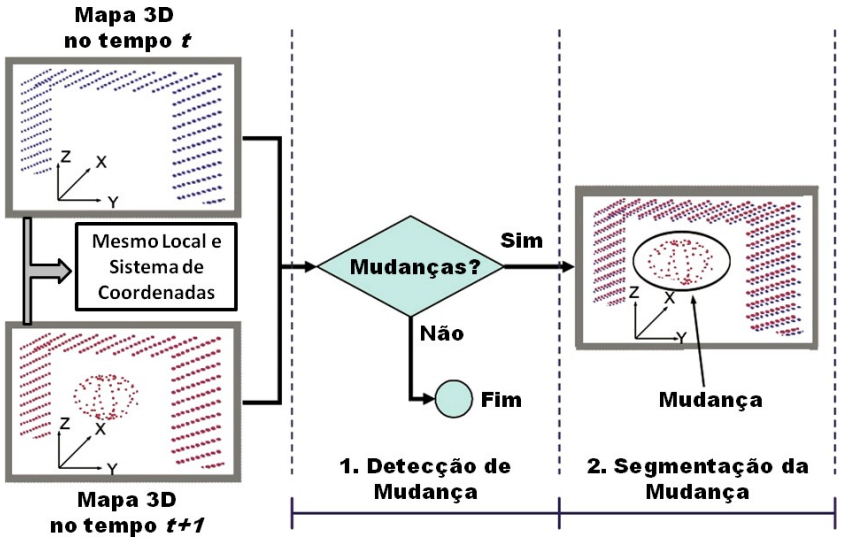


Figura 3.12. Esquema que ilustra a detecção e segmentação de mudanças, com dois mapas 3D em tempos distintos como dados entrada para o sistema.

A Equação 3.29 pode ser usada para quantificar a existência de mudança entre os dois mapas 3D. Para efetivamente conseguir realizar isso, uma abordagem possível é utilizar um limiar U_{th}, que representa o valor acima do qual se considera que houve uma mudança. Assim, valores de $d_{GMM} \in [0; U_{th}]$ representam apenas a presença de ruído nos mapas. Logo, o algoritmo identifica mudança se e, somente se:

$$d_{GMM} > U_{th} \tag{3.30}$$

Porém, tal abordagem, além de não possibilitar a segmentação das mudanças, ainda é de difícil determinação, pois esse limiar é específico para cada ambiente e respectiva nuvem de pontos, sendo de difícil aplicação prática.

Uma abordagem alternativa para ultrapassar essas limitações é proposta neste trabalho, utilizando uma técnica gulosa. A técnica gulosa se baseia de que na idéia de que escolhas locais tendem a gerar melhores resultados globais. Assim, sempre é feita a melhor escolha no momento. A técnica gulosa nem sempre produz o resultado ótimo,

mas são muito simples e eficientes [Cormen et al., 2001]. O algoritmo proposto detecta mudança baseado no principio de que, depois de tomada uma decisão, ela nunca é alterada. Assim, a cada passo do algoritmo é determinado se há uma Gaussiana que representa mudança.

A estrutura geral da abordagem proposta é apresentada em pseudo-código no Algoritmo 3. Tal algoritmo permite, além da detecção de mudanças, a segmentação e recuperação dos pontos associados a cada mudança, utilizando a probabilidade *a posteriori* de cada ponto pertencer a uma dada *GMM* utilizando a Equação 3.12.

Inicialmente, o algoritmo recebe como entrada dois conjuntos de *GMMs* representados por γ, do mapa 3D no tempo t, e θ, do mapa 3D no tempo $t+1$. A cada iteração do algoritmo é selecionada uma Gaussiana $\theta_i = (\mu_i, \mathbf{\Sigma}_i)$ da mistura Gaussiana θ, que apresenta maior redução na distância d_{GMM}, computada pela função *Selecao_Gulosa*. Essa função computa a *EMD* entre as *GMMs* γ e $\theta - \theta_{new}$, removendo uma Gaussiana por vez e escolhendo a que minimiza a distância d_{GMM}. Assim, a Gaussiana que minimiza a distância é adicionada ao conjunto θ_{new}.

Um limiar U_{th} é utilizando apenas para limitar a exatidão e o número de iterações. O resultado do algoritmo é uma lista de conjuntos de pontos C. Cada conjunto de C representa uma região da nuvem de pontos segmentada por uma única Gaussiana, determinada utilizando a Equação 3.12 e computada pela função *Pontos_Associados_Gaussiana*. Essa função tem como argumento a nuvem de pontos P utilizada para gerar a GMM e uma única Gaussiana $\theta_{new_i} = (\mu, \mathbf{\Sigma})$. Se $C = \{\emptyset\}$, então não existe mudanças entre os mapas 3D.

Além disso, uma importante contribuição deste trabalho é utilizar as relações entre Gaussianas determinadas diretamente por meio da nuvem de pontos e a probabilidade de cada ponto da nuvem. Se a probabilidade de que um ponto pertença a duas Gaussianas ao mesmo tempo for maior que 1%, então essas duas Gaussianas são *vizinhas*. Isso permite que o sistema identifique as relações topológicas entre as regiões segmentadas, sendo útil para o método proposto mais adiante para a recuperação de forma utilizando superquádricas.

3.4 Recuperação de Formas 3D

Nesta seção serão introduzidos os algoritmos de recuperação de formas 3D, usados para modelar as mudanças detectadas. Assim, além de detectar mudanças, o presente trabalho determina as formas dessas mudanças utilizando modelos conhecidos. Duas abordagens foram propostas e testadas. A primeira utiliza o espaço das Gaus-

Algoritmo 3 Algoritmo para Seleção de Mudanças - $Calcula\_Mudanca(Conjuntos\ \gamma, \theta)$

1: $d_{GMM} \leftarrow EMD(\gamma, \theta)$ {Equação 3.29}
2: $\theta_{new} \leftarrow \{\emptyset\}$
3: $size \leftarrow 0$
4: **repeat**
5: $\quad d_{GMM_{old}} \leftarrow d_{GMM}$
6: $\quad [\theta_{new}, d_{GMM}] \leftarrow Selecao\_Gulosa(\gamma, \theta - \theta_{new})$
7: $\quad size \leftarrow size + 1$
8: **until** $(d_{GMM_{old}} < d_{GMM}) \vee (d_{GMM} < U_{th})$
9: $\theta_{new} \leftarrow \theta_{new} - \theta_{new_{size}}$
10: $C \leftarrow \{\emptyset\}$
11: **for all** $\theta_{new_i}(\mu, \mathbf{\Sigma}) \in \theta_{new}$ **do**
12: $\quad C \leftarrow C \cup Pontos\_Associados\_Gaussiana(P, \theta_{new_i}(\mu, \mathbf{\Sigma}))$ {Equação 3.12}
13: **end for**

sianas para determinação de algumas formas básicas. A segunda utiliza o modelo de superquádricas visando uma representação mais expressiva, porém menos eficiente computacionalmente. Além disso, nessa última abordagem, outro problema é superado: o mínimo local da segmentação por *GMM* ao se adotar o paradigma *split-and-merge*.

3.4.1 Modelagem Utilizando Formas Básicas

Em estágios anteriores, além da simplificação, foi realizada a detecção de mudanças utilizando o modelo de misturas Gaussianas. Assim, obtém-se o conjunto $\theta_{new} = \{\pi_1, \mu_1, \mathbf{\Sigma}_1, ..., \pi_n, \mu_n, \mathbf{\Sigma}_n\}$ associado às mudanças entre as nuvens de pontos. Sendo $\theta_k = (\mu_k, \mathbf{\Sigma}_k)$ o vetor média e a matriz de covariância associada com a uma Gaussiana, e n o número de formas que representam as mudanças detectadas.

Para validação do método proposto de recuperação de forma usando a matriz de covariância, esse será comparado no Capítulo 5 com outra metodologia clássica da área, utilizando *RANSAC* [Schnabel et al., 2007]. Ambas utilizam a mesma idéia, segundo a qual, para determinar a forma da nuvem de pontos é possível usar forma básicas, no caso, planos, esferas e cilindros.

Uma pequena revisão sobre o *RANSAC* será apresentada na Seção 3.4.1.1, com base no trabalho de Schnabel et al. [2007]. Em seguida será feito um detalhamento do método utilizando o espaço de Gaussianas para determinação de forma, na Seção 3.4.1.2.

A Figura 3.13 ilustra o objetivo dos métodos descritos. Considerando que as mudanças foram determinadas no espaço de Gaussianas, assim uma representação no espaço Euclidiano pode ser efetuada de duas maneiras: a primeira é converter essa

mudanças para o espaço Euclidiano e efetuar o reconhecimento da forma. A segunda é converter modelos conhecidos do espaço Euclidiano para o espaço de Gaussianas e definir a forma nesse espaço. Na Figura 3.13, as duas *GMM* são representadas por elipses, com a elipse em destaque representando a mudança.

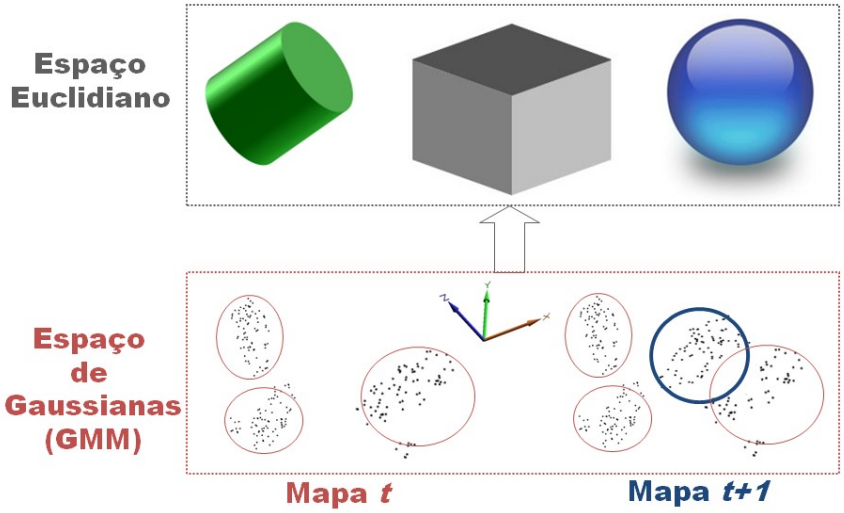


Figura 3.13. Modelagem das mudanças do espaço de Gaussianas com formas básicas, sendo a mudança representada pela elipse em destaque deve ser modelada pelas formas no espaço Euclidiano.

3.4.1.1 Recuperação de Forma no Espaço Euclidiano

Recentemente, um método utilizando o *RANSAC* foi proposto por Schnabel et al. [2007]. Ele foi adaptado para ter como entrada uma nuvem de pontos segmentada pela *GMM*, mas diferentemente do processo convencional de segmentação baseada em probabilidade de pontos, esses pontos foram reamostrados. Como os modelos de formas são restritos, informação excessiva pode prejudicar a detecção. Assim, são amostrados pontos na superfície do elipsóide que representa a Gaussiana.

Deste modo, conhecidas as Gaussianas representadas pelo vetor média e pela matriz de covariância, é possível determinar os autovalores e autovetores associados à matriz de covariância. Os autovetores representam as três direções principais do elipsóide e os autovalores representam o tamanho desses eixos. Utilizando a equação do elipsóide (Equação 3.31), transladada e rotacionada, onde a,b e c representam o tamanho dos eixos e x, y e z são as coordenadas de um ponto, são amostrados pontos na superfície, embora com uma perda de informação da forma. Essa reamostragem facilita o reconhecimento das três formas básicas propostas neste trabalho, devido à densidade que proporciona e à robustez a ruído. A função $Gera\_Amostras\_GMM(\mu_i, \mathbf{\Sigma}_i)$ do

Algoritmo 4 é responsável por efetuar esse processo de reamostragem.

$$\frac{x}{a} + \frac{y}{b} + \frac{z}{c} = 1. \tag{3.31}$$

A Figura 3.14c mostra dois exemplos de nuvem de pontos 3D geradas a partir do processo de reamostragem obtido a partir das Gaussianas mostradas na Figura 3.14b. Na Figura 3.14a são mostradas as nuvens de pontos utilizadas para gerar as Gaussianas.

Sendo $S = \{s1, \ldots, s_n\}$ o conjunto de pontos 3D sintéticos gerados pelo processo de reamostragem, o algoritmo *RANSAC* é aplicado a esses dados. Tal algoritmo permite detectar a forma geométrica das mudanças do mapa.

O *RANSAC* é uma estratégia bem conhecida que pode ser utilizada para reconhecimento de formas a partir de nuvem de pontos 3D, selecionando aleatoriamente subconjuntos mínimos desses dados que são usados para obtenção dos parâmetros do modelo. Os parâmetros são testadas com todos os pontos do conjunto, para determinar quantos pontos aproximam bem a forma. Após um número de tentativas, a forma que melhor se aproxima do conjunto de pontos é a escolhida.

O algoritmo utilizado neste trabalho para determinação de formas básicas no espaço Euclidiano utiliza a abordagem proposta por Schnabel et al. [2007]. Ela implementa eficiente o reconhecimento de forma a partir de nuvem de pontos. O resultado do algoritmo é um conjunto de primitivas de forma $\Psi = \{\Psi_1, \ldots, \Psi_n\}$, com o correspondente conjunto de pontos $S_{\Psi_1} \subset S$, ..., $S_{\Psi_n} \subset S$, além do conjunto de *outliers* $R = S \setminus \{S_{\Psi_1}, \ldots S_{\Psi_n}\}$. O Algoritmo 4 mostra a abordagem utilizada, onde K é o número de Gaussianas detectadas como mudanças e $Size_p$ é o número de pontos desejados na reamostragem da nuvem de pontos da mudança. Maiores detalhes sobre a função de recuperação de forma usando $RANSAC$, $Recupera\_Forma\_RANSAC(P)$, podem ser encontrados em Schnabel et al. [2007]. No presente trabalho, o conjunto de formas possíveis foi limitado a um conjunto $\Psi_{forma} = \{esfera, cilindro, plano\}$.

A Figura 3.14d ilustra o resultado, em azul, do algoritmo para os conjuntos de pontos mostrados na Figura 3.14c. Na imagem à direita, (cilindro) da Figura 3.14d, é mostrado, em azul, a forma gerada a partir de um cilindro ideal. Pode-se notar um erro de orientação na forma recuperada pela abordagem utilizando $RANSAC$. Isto se deve, principalmente, ao fato da perda de informação gerada pela reamostragem da GMM.

3.4.1.2 Recuperação de Forma no Espaço das Gaussianas

Este método busca encontrar a forma que melhor se aproxima de uma forma básica ideal, considerando Ψ_{forma}. Para tal, este trabalho contribui com um método que usa

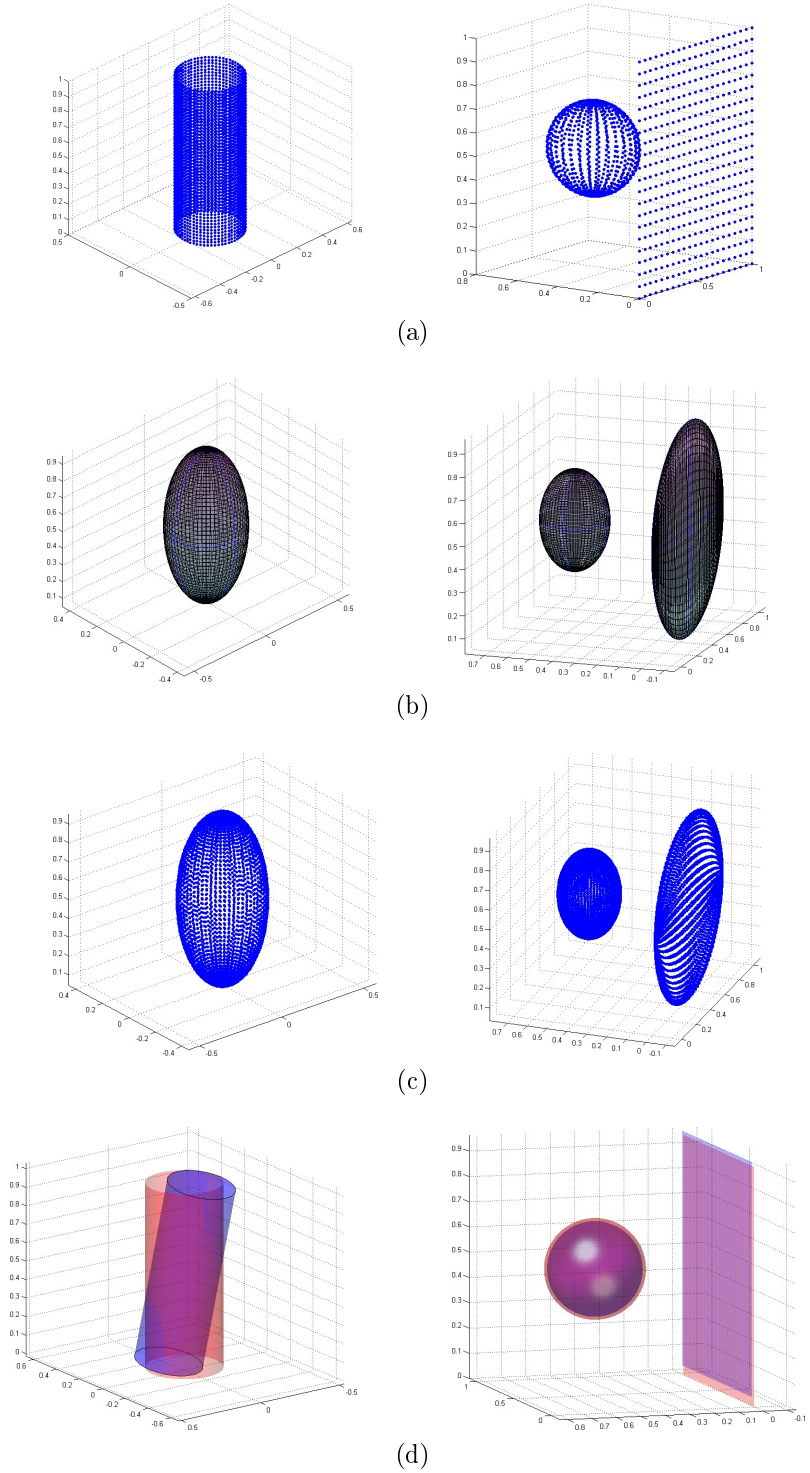


Figura 3.14. Ilustração do resultado do algoritmos de recuperação de formas usando formas básicas: (a) Nuvem de pontos original da mudança; (b) Gaussianas que descrevem as mudanças; (c) Nuvem de pontos geradas por reamostragem; (d) forma recuperada de acordo com método utilizado: espaço Euclidiano usando *RANSAC* (azul) e espaço de Gaussianas (vermelho).

Algoritmo 4 Algoritmo para Recuperação de Formas Básicas no Espaço Euclidiano

$\Psi \leftarrow \emptyset$
$S \leftarrow \emptyset$
for $i = 0$ to K **do**
  while $n <= Size_p$ **do**
    $P \leftarrow P \cup Gera\_Amostras\_GMM(\mu_i, \mathbf{\Sigma}_i)$
    $n \leftarrow n + 1$
  end while
  $(\Psi_i, S_{\Psi_i}) \Leftarrow Recupera\_Forma\_RANSAC(P)$
  $\Psi \leftarrow \Psi \cup \Psi_i$
  $S \leftarrow S \cup S_{\Psi_i}$
end for

Algoritmo 5 Algoritmo para Recuperação de Formas Básicas no Espaço de Gaussianas

1: $\Psi \leftarrow \emptyset$
2: $\Delta \leftarrow \emptyset$
3: $N_{formas} \leftarrow 3$
4: **for** $i = 1$ to N_{formas} **do**
5: **for** $j = 0$ to K **do**
6: $[d^j_\Psi, \Psi^j, \mathbf{T}^j] \leftarrow Distancia\_Covariancia(\mathbf{\Sigma}_j, \mathbf{\Sigma}_i)$
7: **end for**
8: $(\Psi_i, \mathbf{T}_i) \Leftarrow Seleciona\_Melhor\_Forma(d_\Psi)$
9: $\Psi \leftarrow \Psi \cup \Psi_i$
10: $\Delta \leftarrow \Delta \cup \mathbf{T}_i$
11: **end for**
12: **return** (Ψ, Δ)

o espaço das Gaussianas (GMM), descrito pela matriz de covariância e pela média das funções Gaussianas.

Sendo θ a mistura Gaussiana associada às mudanças identificadas no mapa, cada mudança k é caracterizada por uma função Gaussiana $\theta_k = (\mu_k, \mathbf{\Sigma}_k)$. Desse modo, um método para recuperação de forma baseado na correspondência de matrizes de covariância é apresentado no Algoritmo 5. A obtenção do melhor modelo de forma e a sua transformação em relação à forma ideal, $\mathbf{T}$, são os objetivos desse algoritmo. É efetuada a correspondência entre funções Gaussianas e cada forma básica, medindo-se a similaridade entre essas matrizes de covariância denominada $d_\Psi = \{d_{esfera}, d_{cilindro}, d_{plano}\}$. O menor valor de d_Ψ determina a forma, juntamente com a transformação que melhor aproximam a nuvem de pontos.

A comparação entre matrizes de covariância é uma tarefa básica no desenvolvimento de medidas. O objetivo é determinar uma medida de distância entre duas matrizes. O espaço dessas matrizes não é vetorial, assim a aritmética padrão não

permite determinar a diferença entre as medidas. Por outro lado, uma matriz de covariância é simétrica e semi-definida positiva. Assim, a distância pode ser formulada baseada na métrica de Riemann [Forstner & Moonen, 1999].

Neste trabalho, a métrica de distância descrita por Forstner & Moonen [1999] foi utilizada e é definida como:

$$d(\mathbf{\Sigma}_1, \mathbf{\Sigma}_2) = \sqrt{\sum_{i=1}^{N} ln^2 \lambda_i(\mathbf{\Sigma}_1, \mathbf{\Sigma}_2)}, \tag{3.32}$$

onde $\mathbf{\Sigma}_1$ e $\mathbf{\Sigma}_2$ são as duas matrizes de covariância a serem comparadas, λ representa os autovalores generalizados de $\mathbf{\Sigma}_1$ e $\mathbf{\Sigma}_2$, e N é a dimensionalidade das matrizes. O problema dos autovalores generalizados pode ser expresso como $Ax = \lambda Bx$, onde A e B são matrizes conhecidas e λ (escalar) e x (vetor) são as variáveis a serem encontradas. No caso do problema clássico dos autovalores, a matriz B é a matriz identidade [Zakrajšek, 1976].

A matriz $\mathbf{\Sigma}_1$ é a covariância da função Gaussiana que representa uma mudança a ser recuperada a forma e $\mathbf{\Sigma}_2$ é a covariância de escala unitária conhecida *a priori* de cada forma básica, ou seja, da esfera, do cilindro ou do plano. Para considerar as possíveis rotações e variações de escala dos eixos, deve-se notar que:

$$\mathbf{\Sigma}_i = \mathbf{T}\mathbf{\Sigma}_j\mathbf{T}^T = (\mathbf{R}\cdot\mathbf{E})\mathbf{\Sigma}_j(\mathbf{R}\cdot\mathbf{E})^T, \tag{3.33}$$

onde $\mathbf{T}$ representa a transformação aplicada à forma geométrica ideal, que é composta pelas matrizes de rotação e escala, $\mathbf{R}$ e $\mathbf{E}$, mostradas abaixo nas equações 3.34 e 3.35. Nessa abordagem, a translação é conhecida diretamente por meio da informação do vetor média μ de cada Gaussiana.

$$\mathbf{R} = \begin{bmatrix} c\psi c\varphi - c\vartheta s\varphi s\psi & c\psi s\varphi + c\vartheta s\varphi s\psi & s\psi s\vartheta \\ -s\psi c\varphi - c\vartheta s\varphi c\psi & -s\psi s\varphi + c\vartheta c\varphi c\psi & c\vartheta s\psi \\ -s\vartheta s\varphi & -s\vartheta c\varphi & c\vartheta \end{bmatrix}, \tag{3.34}$$

$$\mathbf{E} = \begin{bmatrix} E_x & 0 & 0 \\ 0 & E_y & 0 \\ 0 & 0 & E_z \end{bmatrix}, \tag{3.35}$$

onde ψ, ϑ, φ, são os ângulos de rolamento, guinada e arfagem, respectivamente, c e s são abreviações das funções cosseno e seno, E_x, E_y e E_z são os fatores de escala em cada eixo.

Desse modo, é possível minimizar a Equação 3.32 usando um método de mínimos quadrados como o método de *Levenberg-Marquardt* [Marquardt, 1963], com objetivo de modificar os parâmetros de rotação e escala em cada iteração. Uma boa inicialização é necessária para reduzir o número de iterações, bem como determinar corretamente os parâmetros a serem estimados, evitando mínimos locais.

Para realizar a inicialização, o algoritmo usa uma aproximação da transformação T de acordo com os autovetores e os autovalores da matriz de covariância, determinando, assim, a rotação e a escala. A Figura 3.14d mostra, em vermelho, as formas recuperadas utilizando o método para o espaço de Gaussianas.

3.4.2 Modelagem Utilizando Superquádricas

Esta seção introduz o segundo método proposto para recuperação de formas 3D, que visa encontrar o modelo de Superquádricas (SQ) para as mudanças detectadas. Esse modelo tem como principal vantagem, em relação ao anterior, a expressividade, permitindo a representação de uma variedade grande de formas, utilizando apenas dois parâmetros, como mostra a Figura 3.15. Além disso, o algoritmo proposto para recuperação de formas usando superquádricas ultrapassa a restrição imposta pela segmentação de mudanças utilizando EM para estimar a GMM, que somente consegue garantir mínimos locais.

Durante a detecção de mudanças, são obtidos os conjuntos de pontos relacionados com as mudanças identificadas no ambiente. Diferentemente do método utilizando formas básicas no espaço Euclidiano que efetua reamostragem para migrar do espaço de Gaussianas para o Euclidiano, o presente método utiliza a probabilidade *a posteriori* para segmentar os pontos no espaço Euclidiano. A abordagem de reamostragem não é atraente neste caso, devido à perda de informação, o que inviabilizaria a recuperação de um modelo de superquádricas.

A Figura 3.15 busca ilustrar o fluxo de dados do método proposto para recuperação de formas usando superquádricas. As entradas do sistema, definidas pelas setas pretas, são as mudanças segmentadas com as respectivas representações em múltiplas escalas. O resultado gerado são os modelos das superquádricas (SQ). Os três primeiros passos, assim como a detecção de mudança, são efetuados com a nuvem simplificada. Posteriormente, são efetuados os refinamentos utilizando múltiplas escalas, de forma a permitir uma abordagem mais eficiente e exata. Os passos mostrados na Figura 3.15 serão detalhados nas próximas seções.

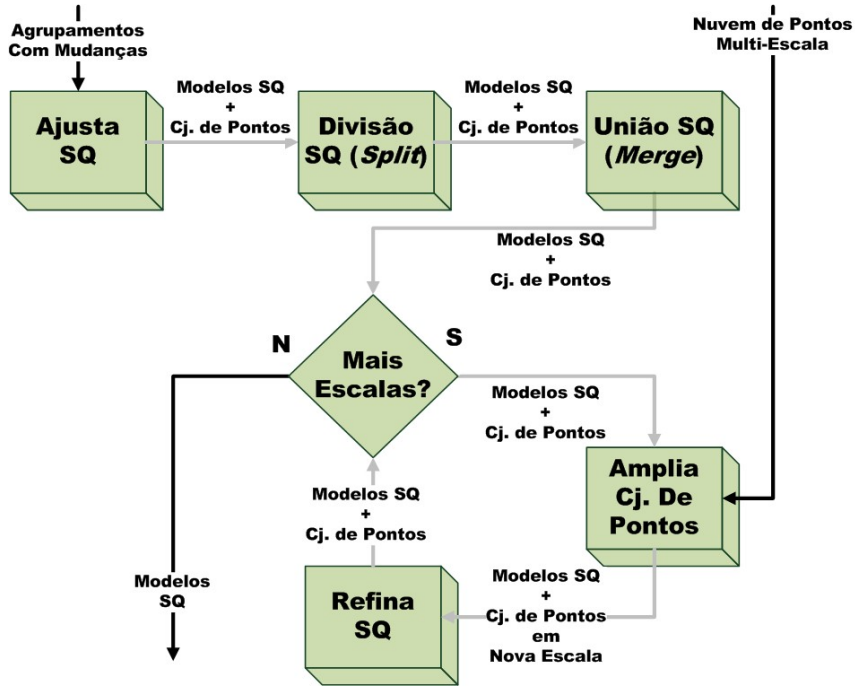


Figura 3.15. Diagrama de fluxo de dados para recuperação de formas 3D usando superquádricas.

3.4.2.1 Modelo de Superquádricas

As superquádricas [2] são uma familia de formas geométricas que podem ser divididas em quatro modelos: supertoróides, superhiperbolóides com uma folha, superhiperbolóides com duas folhas e superelipsóides. Este trabalho é focado nos superelipsóides devido à sua utilidade para determinação de forma, por serem compactos e terem forma fechada.

As superquádricas são definidas usando dois parâmetros de forma, ϵ_1 e ϵ_2, e três parâmetros de escala, a_1, a_2 e a_3. A Figura 3.16 mostra as formas geradas, variando-se apenas os parâmetros de forma e mantendo-se constante os parâmetros de escala. Para geração de superquádricas, geralmente, é utilizado o modelo paramétrico da Equação 3.36:

$$S(\eta,\omega) = \begin{bmatrix} a_1 \cos^{\epsilon_1}(\eta)\cos^{\epsilon_2}(\omega) \\ a_2 \cos^{\epsilon_1}(\eta)\sin^{\epsilon_2}(\omega) \\ a_3 \sin^{\epsilon_1}(\eta) \end{bmatrix}, -\frac{\pi}{2} \leq \eta \leq \frac{\pi}{2}\, e\, \pi \leq \omega \leq \pi. \tag{3.36}$$

Por outro lado, as superquádricas também podem ser definidas pela equação implícita, como mostra a Equação 3.37. A possibilidade de utilizar as duas representações diretamente, implícita e paramétrica, é um dos pontos principais para sua larga uti-

[2] Ao longo do texto, a palavra superquádrica é usada para representar o modelo de superelipsóide.

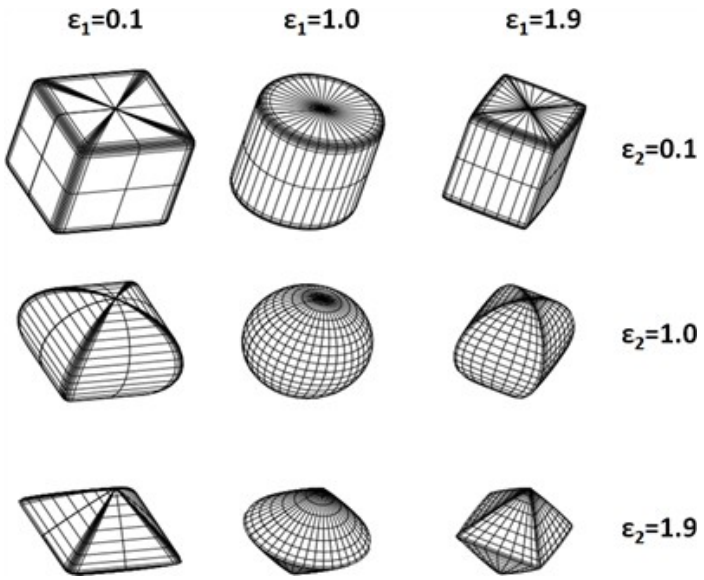


Figura 3.16. Superquádricas geradas com diferentes valores de ϵ_1 e ϵ_2.

lização. Esta vantagem permite renderizar e amostrar a forma, utilizando a equação paramétrica, bem como recuperar a forma utilizando a equação implícita.

$$F(x,y,z) = \left(\left(\frac{x}{a_1} \right)^{\frac{2}{\epsilon_2}} + \left(\frac{y}{a_2} \right)^{\frac{2}{\epsilon_2}} \right)^{\frac{\epsilon_2}{\epsilon_1}} + \left(\frac{z}{a_3} \right)^{\frac{2}{\epsilon_1}}. \tag{3.37}$$

A equação implícita fornece a informação da posição de um ponto 3D em relação à superfície da superquádrica. Basicamente, o valor dessa função é:

1. $F(x,y,z) = 1$, se o ponto está situado na superfície;

2. $F(x,y,z) > 1$, se o ponto está fora da superfície;

3. $F(x,y,z) < 1$, se o ponto está dentro da superfície.

Para representar o modelo em um sistema de coordenadas global, a fim de recuperar a forma a partir de nuvem de pontos, são necessários mais seis parâmetros. Eles permitem expressar a translação e a rotação da superquádrica. Neste trabalho são utilizados os ângulos de Euler (ϕ,θ,ψ) para representar a rotação, além da translação p_x, p_y e p_z. Assim, a função implícita tem a seguinte forma:

$$F(x,y,z;\Lambda) = F(x,y,z;\epsilon_1,\epsilon_2,a_1,a_2,a_3,\phi,\theta,\psi,p_x,p_y,p_z). \tag{3.38}$$

3.4.2.2 Ajuste de Superquádricas Usando Multi-Escalas

Considerando um conjunto de pontos 3D, o primeiro objetivo é estimar os parâmetros do modelo da superquádrica. O método de minimização de gradiente utilizando mínimos quadrados, baseado no algoritmo de Levenberg-Marquardt, é o mais utilizado para resolver esse problema [Jaklič et al., 2000]. O método busca minimizar a seguinte expressão:

$$\min_{\Lambda} \sum_{i=1}^{n} (\sqrt{a_1 a_2 a_3}(F^{\epsilon_1}(x_i, y_i, z_i; \Lambda) - 1))^2. \tag{3.39}$$

Essa equação representa uma métrica de distância que permite comparar superquádricas. A restrição $\sqrt{a_1 a_2 a_3}$ tem por objetivo forçar a recuperação da menor superquádrica possível. Em situações onde os dados são incompletos, pode-se obter o modelo de superquádricas com escalas infinitas caso a superfície formada pelos pontos não seja fechada. O expoente ϵ_1 faz com que o erro gerado seja independente da forma da superquádrica [Jaklič et al., 2000].

Embora existam outros métodos para computar essa distância entre um ponto e uma superquádrica, como o caso da distância radial [Whaite & Ferrie, 1991], esses métodos são mais lentos [Chevalier et al., 2003].

Outro aspecto muito importante no ajuste de superquádricas é a determinação de um modelo inicial para os dados. Esse modelo determina para qual mínimo local o método vai convergir, bem como o número de iterações necessárias para que isso ocorra. Desse modo, uma boa inicialização é crucial para o sucesso do método de ajuste de superquádricas. Considerando um conjunto de pontos em $\mathbb{R}^3$ com tamanho n, é utilizado para estimação da pose inicial, o centro de gravidade ($\{\overline{x}, \overline{y}, \overline{z}\}$), do qual se obtém o centro da superquádrica, e os momentos centrais ($\mathbf{I}$) desses pontos, como mostra Equação 3.40, dos quais se obtém os ângulos de Euler que representam as rotações [Jaklič et al., 2000]. Além disso, considera-se como forma inicial um elipsóide, ou seja, $\epsilon_1, \epsilon_2 = 1$. Para estimação dos fatores de escala é possível utilizar os valores máximos e mínimos nos eixos cartesianos x,y e z, como no trabalho de Leonardis et al. [1997]. Porém, neste trabalho, utiliza-se os autovalores (λ) da matriz de inércia I, como mostra a Equação 3.41. Esse método tem como principal vantagem uma maior tolerância aos *outliers*.

$$\mathbf{I} = \frac{1}{n} \sum_{i=1}^{n} \begin{bmatrix} (y_i - \overline{y})^2 + (z_i - \overline{z})^2 & -(y_i - \overline{y})(x_i - \overline{x}) & -(z_i - \overline{z})(x_i - \overline{x}) \\ -(x_i - \overline{x})(y_i - \overline{y}) & (x_i - \overline{x})^2 + (z_i - \overline{z})^2 & -(z_i - \overline{z})(y_i - \overline{y}) \\ -(x_i - \overline{x})(z_i - \overline{z}) & -(y_i - \overline{y})(z_i - \overline{z}) & (x_i - \overline{x})^2 + (y_i - \overline{y})^2 \end{bmatrix}, \tag{3.40}$$

$$
\begin{aligned}
a_1 &= \sqrt{\tfrac{3}{2}(\lambda_2 + \lambda_3 - \lambda_1)}, \\
a_2 &= \sqrt{\tfrac{3}{2}(\lambda_1 + \lambda_3 - \lambda_2)}, \\
a_3 &= \sqrt{\tfrac{3}{2}(\lambda_1 + \lambda_2 - \lambda_3)}.
\end{aligned} \tag{3.41}
$$

O presente trabalho contribui com uma abordagem multi-escala para ajuste de superquádricas. A idéia do método, como mostra a Figura 3.15, é computar um modelo inicial, utilizando os pontos segmentados pela *GMM*, ou seja, simplificados. Depois disso, o modelo é refinado e para tal é utilizado como modelo inicial do algoritmo iterativo o modelo ajustado com os pontos simplificados, além de um novo conjunto de pontos. Esse conjunto é formado pelos pontos em uma escala mais fina, ou seja, por um número maior de pontos.

Uma abordagem para superar problemas de estimação inicial é utilizada neste trabalho. Como a estimação inicial é feita com pontos simplificados, assim podem ocorrer erros devido aos mínimos locais. Assim, são estimadas duas superquádricas em cada processo de refinamento. A primeira utiliza o modelo já determinado pelos pontos simplificados como inicialização, e a segunda utiliza a estimação usando a matriz de inércia. Por fim, é escolhido o melhor modelo, ou seja, aquele que minimiza a distância global dos pontos para a superquádrica, também conhecida como distorção.

3.4.2.3 Paradigma *Split-and-Merge*

As mudanças determinadas pelo método *GMM-EMD* têm uma séria limitação: elas podem falhar devido a mínimos locais fazendo com que a forma gerada pudesse ter sido dividida em menos partes do que o esperado. Uma abordagem baseada em *split-and-merge* foi proposta por Chevalier et al. [2003] para superar esse problema. Neste trabalho, é proposta uma extensão ao método, constituindo-se uma contribuição importante: para inicialização do método são utilizadas as regiões segmentadas; além disso se determina as relações topológicas por meio do método de detecção de mudanças utilizando *GMM*. Isso permite simplificar o método, bem como reduzir o custo computacional.

O primeiro passo é o ***split***, ou divisão dos dados, que busca garantir que todos os pontos em um subconjunto pertençam a um mesmo objeto a ser modelado por uma superquádrica. Desse modo, ao fim desse passo, é gerado um novo conjunto de superquádricas, de maior ou igual tamanho ao inicialmente estimado. Esse conjunto é gerado computando-se a matriz de inércia de cada superquádrica candidata a ser dividida (*split*), assim é possível determinar os autovalores e autovetores. Por meio desses é encontrado o plano de corte. Um plano pode ser definido por meio de um

vetor normal ao plano e um ponto. Assim, o ponto utilizado é o centro de massa do conjunto e o vetor normal utilizado é o autovetor associado ao maior autovalor da matriz de inércia do conjunto.

Após efetuada a divisão do conjunto em duas partes, é então ajustada uma superquádrica para cada uma das partes e verificado se a distorção (estimada pela Equação 3.39) dos dois conjuntos é menor que a distância do conjunto original. Esse método cria uma árvore binária que representa a relação topológica entre os segmentos gerados, assumindo como conhecida a relação topológica inicial dada por meio da GMM, ela é estendida para os filhos gerados pela divisão. Essa relação topológica ajuda a evitar que o método de união (*merge*) precise recalcular as relações, pois seria extremamente custoso.

O outro passo é o processo de união, ou ***merge***. Ele considera resolvido o problema de que cada subconjunto de pontos representa um único objeto a ser modelado, não permitindo em um mesmo subconjunto que existam pontos de dois objetos distintos. Assim, é efetuada a união dos conjuntos com objetivo de reduzir a número de formas, sem aumentar a distorção. Este método pode ser dividido em duas partes:

1. **Atualização da Matriz de Custos**: Para cada subconjunto de pontos é computada a matriz custo, que estima o custo de unir os vizinhos, presumindo que a relação topológica é conhecida. Esse passo é efetuado a cada vez que ocorre uma união, por meio de um método de memoização (*memoization*).

2. **União das Superquádricas**: São escolhidos dois subconjuntos que formam uma superquádrica de menor distorção. Assim, os conjuntos são unidos se a nova distorção é menor que a média das distorções dos dois conjuntos, efetuando assim, uma redução global da distorção. Além disso, é considerado o volume da nova forma, que deve ser menor que a soma dos volumes da duas outras superquádricas.

Além da contribuição no uso de conhecimento da topologia, utilizando o detector de mudanças, que reduz bastante o custo computacional do método de *merge*, uma importante contribuição é o uso do método de memoização [Cormen et al., 2001]. Essa técnica de otimização é utilizada para uma redução no custo de processamento, evitando recalcular valores. Neste trabalho, é utilizada uma matriz onde são guardadas as distorções da superquádrica gerada pela união. Quando os conjuntos são unidos, a linha e a coluna referente aos conjuntos unidos são zeradas e recalculadas, porém, os outros valores permanecem inalterados. Além disso, durante o processo de união, as relações topológicas são atualizadas. Assim, a abordagem deste trabalho resulta em

um método mais eficiente do que o proposto no trabalho de Chevalier et al. [2003]. O tempo de processamento para computar a forma 3D da nuvem de pontos é aproximadamente 5% da detecção de mudanças, como será mostrado no Capítulo 5. Esse custo é dependente do número de mudanças detectadas, além das características topológicas e do número de pontos.

Capítulo 4

Arcabouço Experimental

Este capítulo descreve o aparato experimental utilizado neste trabalho. Primeiramente, para validação e testes preliminares, foram utilizados dados de mapas 3D simulados, com e sem adição de ruído Gaussiano branco. Em uma segunda etapa, foram utilizados mapas 3D reais. Para a obtenção de tal foi feita a integração de uma unidade de pan-tilt, um *laser scanner* 2D e uma central inercial. Em um segundo momento foram obtidos mapas 3D reais a partir de um robô móvel equipado com um laser 2D montado ortogonalmente ao movimento do mesmo, gerando assim mapas 3D.

4.1 Experimentos Usando Simulação

Os primeiros resultados experimentais foram realizados utilizando dados de simulação. Assim como no trabalho de Amorim et al. [2008], foram gerados conjuntos de dados de teste, os quais simulam a leitura de um *laser scanner* ideal. Esses dados foram gerados em Matlab [The Mathworks Inc., 2009].

Foram efetuados diversos experimentos, nos quais foram adicionados ruídos Gaussianos aleatórios, com média zero e variância pré-definida, de forma a gerar dados mais próximos do real. Porém, a presença de *outliers* não foi simulada.

A Figura 4.1 mostra o resultado da aplicação de ruído nos dados simulados. Em preto, tem-se um corredor simulado, com a presença de um plano à direita, sem ruído. Pode-se notar que o ruído de variância $0,1m^2$ torna os dados inutilizáveis. Para os resultados deste trabalho utilizou-se variância $0,001m^2$.

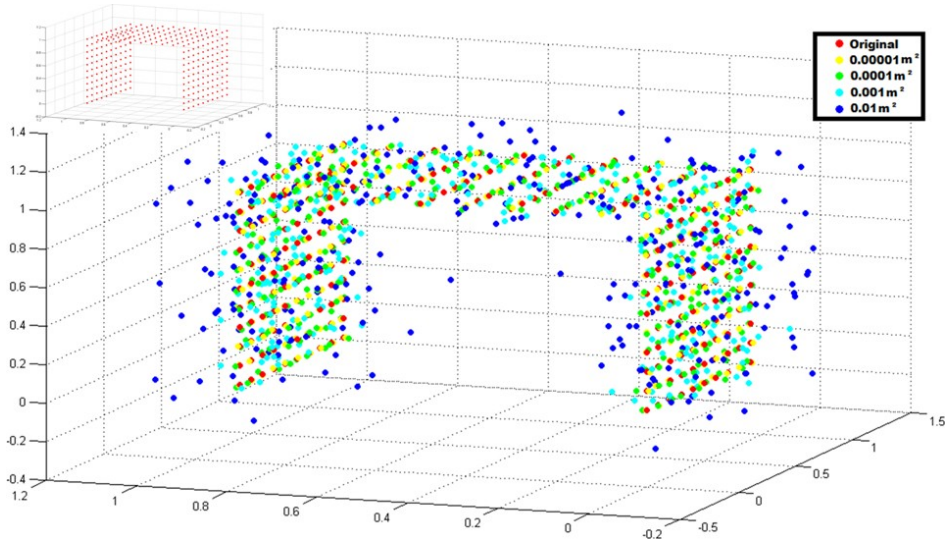


Figura 4.1. Exemplo de dados simulados com diferentes níveis de ruído, incluíndo a ausência de ruído em vermelho (em destaque no canto superior esquerdo). As diversas cores ilustram os resultados dos valores de variância no ruído sobre os dados.

4.2 Experimentos Usando Dados Reais

A idéia inicial deste trabalho era utilizar como entrada os dados 3D adquiridos pelo projeto europeu *IRPS*, o qual se propõe a modelar o espaço 3D. Estes mapas seriam gerados a partir de um conjunto de *laser scanners* 2D ou de um *laser scanner* 3D projetado durante o projeto. Porém, esses mapas não estavam disponíveis a tempo de serem utilizados neste trabalho. Maiores detalhes sobre o projeto *IRPS* podem ser encontrados no Anexo A. Então, definiu-se por utilizar inicialmente uma solução estática utilizando um único *laser scanner* montado sobre uma unidade de *pan-tilt*, como mostra a Figura 4.2.

Para a aplicação em vigilância, uma montagem estática pode até ser realística. Porém, para uma aplicação em robótica, foi realizado experimentos utilizando um veículo móvel dotado de *laser scanner* 2D e de odometria nas rodas. Com tal veículo foi adquirido diversos mapas 3D. Tal abordagem é detalhada na Seção 4.2.2.

4.2.1 Montagem Estática

Laser scanners 3D são, geralmente, construídos movimentando-se *scanners* 2D em torno de um eixo. Rotacionando um *laser scanner* 2D no seu eixo radial é possível obter as coordenadas tridimensionais dos pontos medidos. Porém, essa construção cria alguns problemas como a dificuldade de ajustar o centro de rotação dos espelhos do *laser*

com o centro de rotação do próprio *laser* [Scaramuzza et al., 2007], definido por d_x e d_z na Figura 4.2a. O *laser scanner* 3D utilizado nos experimentos foi construído usando um *laser* 2D Hokuyo e uma unidade de *pan-tilt* fornecida pela Directed Perception, com correção de angulação dada por uma central inercial. Este tipo de configuração pode ser modelada pela Equação 4.1.

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} c_i c_j & -c_i s_j & s_i & c_i d_x + s_i dz \\ s_j & c_j & 0 & 0 \\ -s_i c_j & s_i s_j & c_i & -s_i d_x + c_i dz \end{bmatrix} \begin{bmatrix} \rho_{ij} \\ 0 \\ 0 \\ 1 \end{bmatrix}, \tag{4.1}$$

$$c_i = \cos(\varphi_i), c_j = \cos(\theta_j), s_i = \sin(\varphi_i), s_j = \sin(\theta_j),$$

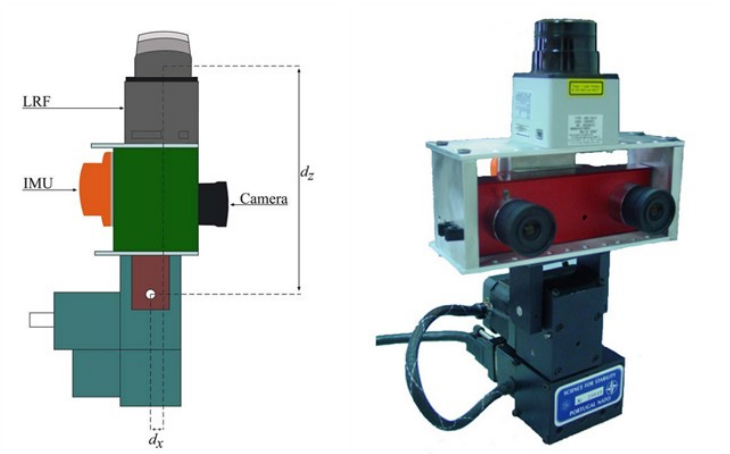
onde ρ_{i_j} é a j-ésima medida de distância com orientação θ_j no i-ésimo plano de leitura, que faz o ângulo φ_j com o plano horizontal. Os deslocamentos do eixo de rotação com o centro do espelho tem componente d_x e d_z, como mostra a Figura 4.2a. Considera-se ainda que $[x\, y\, z]^T$ são as coordenadas de cada ponto medido em relação ao centro de rotação do *laser*, com eixo x apontando para frente e o eixo z apontando para cima.

Utilizou-se um sistema composto por uma unidade de *pan-tilt* fornecida pela Directed Perception do tipo PTU-D46-17 [Directed Perception, 2009]. Essa unidade tem uma resolução de $0,051^0$, além de permitir velocidades de $300^0/s$. Foi utilizada uma amplitude de rotação de 180^0 no ângulo de arfagem (*pitch*), em passos de 1^0. Essa unidade foi controlada por meio de uma porta serial RS-232.

Como *Laser Ranger Finder* (*LRF*) foi utilizado 2D Hokuyo URG-04LX [Hokuyo Automatic Co., 2009]. Esse sensor tem acurácia de 1% sobre a distância medida e, além disso, tem uma amplitude de 240^0, com resolução de $0,36^0$. Para a leitura deste sensor utilizou-se uma interface USB.

Foi utilizado um sistema inercial para determinação da pose exata do *laser scanner* em cada leitura. Trata-se de um sensor MTi fabricado pela XSens Technologies [XSens Technologies B.V., 2009]. Esse sensor é dotado de uma central inercial do tipo *Micro-Electro-Mechanical Systems* (*MEMS*) com comunicação USB e tem uma resolução angular de $0,5^0$ com taxa de atualização de 120 Hz.

A Figura 4.2 mostra a montagem utilizada, onde um diagrama esquemático da montagem é mostrado na 4.2a e a imagem real do sistema na Figura 4.2b. Observa-se nessa figura a existência de um par de câmeras estéreo. Esse par não foi utilizado nos experimentos, mas há a previsão de utilização em trabalhos futuros.



(a) (b)

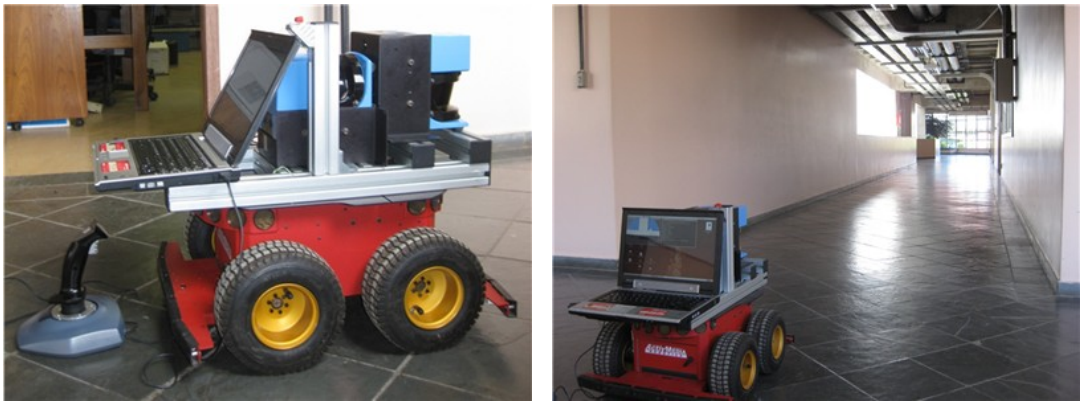
Figura 4.2. Montagem de um *laser scanner*, junto com uma unidade de *pan-tilt* e uma central inercial, em (a) ilustra o diagrama da montagem, enquanto em (b) uma foto da montagem é mostrada.

4.2.2 Montagem Dinâmica

Esta montagem baseou-se no uso de um *laser scanner* 2D montado ortogonalmente ao plano de movimento, de forma a obter uma nuvem de pontos 3D. A Figura 4.3 ilustra a montagem.

Os experimentos utilizam um robô Pioneer 3 AT fabricado pela Mobile Robotics [MobileRobots Inc., 2009], dotado de odometria com resolução angular de $0,72^0$. O *laser scanner* utilizando foi um SICK LMS 200 [Sick AG., 2009]. Tal sensor é dotado de resolução angular de até $0,25^0$ com amplitude de 180^0 e alcance de 80 metros. Como mostra a Figura 4.3a foram montados ortogonalmente dois sensores SICK, porém para a aquisição dos mapas foi utilizado apenas o *laser scanner* montado ao centro do robô. O *joystick* mostrado na figura foi utilizado apenas para movimentação do robô até o local dos experimentos, não sendo utilizando durante a aquisição dos mapas. Esta aquisição foi feita de maneira automática.

Os experimentos foram realizados em um ambiente de corredores, como mostra a Figura 4.3b. Nesse ambiente foram adicionados objetos próximos a parede, a fim de tornar o cenário mais realístico. O ponto de partida do robô em todos experimentos foi aproximadamente o mesmo para se obter o mesmo sistema de coordenadas, porém isso não seria necessário caso se conhecesse a transformação entre as posições de partida do robô. A trajetória executado foi pré-definida com 8 metros em um linha reta, próxima ao centro do corredor.



(a) (b)

Figura 4.3. Montagem de um *laser scanner 2D* sobre um robô móvel do tipo Pioneer 3 - AT utilizada para aquisição de mapas 3D. Em (a) é mostrado o robô em destaque, em (b) é mostrado o ambiente de corredores utilizados nos experimentos.

4.3 Módulo de Software

A necessidade de obter-se uma implementação eficiente que permitisse a integração com o projeto *IRPS* e o *software Carnegie Mellon Robot Navigation Toolkit* (*Carmen*), fez com que a metodologia proposta fosse desenvolvida em linguagem C/C++, com exceção do módulo de reconhecimento de forma utilizando o espaço Gaussiano, desenvolvido em Matlab [The Mathworks Inc., 2009].

Os experimentos foram executados em um computador pessoal com processador AMD Turion X2 ZM-82 com 2Mb de memória cache e 3 Gb de memória RAM DDR2-800. Além disso, utilizou-se sistema operacional GNU-Linux com distribuição Ubuntu 9.04.

Considerando o esboço da metodologia mostrado na Figura 3.1 observa-se a divisão em três módulos de *software*. O primeiro módulo, que efetua a simplificação dos mapas 3D, foi desenvolvido utilizando como base as funções do *software* PointShop 3D [Zwicker et al., 2002].

O módulo de detecção de mudanças utiliza como base o cálculo das misturas Gaussianas (*GMM*), bem como a métrica de distância *EMD*. Para a determinação das *GMMs* utilizou-se, como base, o software livre fornecido por Bouman [1997], com diversas modificações descritas na Seção 3.3.1.3. O *EMD* também foi adaptado da implementação original fornecida por Rubner et al. [1998].

O módulo de reconhecimento de forma usando superquádricas foi desenvolvido em C/C++, enquanto o módulo de reconhecimento de formas básicas no espaço de

Gaussianas foi implementado em Matlab. Embora ele tenha sido desenvolvido também em C/C++, não se conseguiu obter a mesma qualidade de resultados, devido à implementação ineficiente do método de minimização de Levenberg-Marquardt e de estimação de autovalores generalizados, comparada à do software Matlab.

A implementação do módulo de reconhecimento de formas básicas no espaço Euclidiano foi originalmente disponibilizada por Schnabel et al. [2007] em C/C++. A partir desta, foram realizadas alterações a fim de se adaptar ao presente trabalho, como descrito na Seção 3.4.1.1.

Um sistema de software utilizado no projeto *IRPS* e, também, no presente trabalho para comunicação e controle dos dispositivos sensores, foi o *Carmen*. No Anexo A são fornecidos maiores detalhes sobre essa ferramenta.

Capítulo 5

Resultados

Neste capítulo são apresentados e discutidos os resultados obtidos utilizando a metodologia proposta, com dados simulados e com dados reais. Primeiramente, é validada a utilização do método de simplificação como um passo fundamental desta metodologia. O método de detecção de mudanças também é validado e, posteriormente, é realizada uma comparação entre o método de reconhecimento de formas básicas no espaço de Gaussianas e o método de determinação no espaço Euclidiano. Por fim, são mostrados os resultados de reconhecimento de forma usando superquádricas.

A fim de testar os algoritmos, objetos simulados de diferentes tamanhos foram introduzidos em diferentes poses dentro do corredor padrão. Assim, trinta diferentes conjuntos de mapas 3D foram gerados com adição de ruído. Os mapas reais, tanto os obtidos por meio de um *laser scanner* sobre o robô em movimento quanto os obtidos pelo movimento do *laser scanner* na unidade de *pan-tilt*, são de seis diferentes cenários sendo efetuada uma captura com e outra sem mudanças.

5.1 Detecção de Mudanças com Simplificação da Nuvem de Pontos

Um exemplo do resultado de detecção de mudança utilizando dados simulados é ilustrado na Figura 5.1. As figuras 5.1a e 5.1b ilustram os mapas que simulam um *laser scanner* ideal, sem e com a mudança, respectivamente, ou seja, os mapas nos tempos t e $t+1$ anterior a adição de ruído. A Figura 5.1c mostra o mapa em $t+1$: em azul, o mapa após a adição de ruído, e em vermelho, o resultado da simplificação de pontos no nível mais grosseiro da nuvem de pontos em azul. O mesmo procedimento de adição de ruído foi efetuado no mapa em t. Já as figuras 5.1d e 5.1e ilustram os resultados

do processo de agrupamento utilizando *GMM* para os mapas nos tempos t e $t+1$. Os pontos azuis mostrados são os dados de entrada do processo de agrupamento, ou seja, os pontos com ruído simplificados, no tempo t e $t+1$, respectivamente. O retângulo preto na Figura 5.1e foi colocado para melhor visualização da mudança no mapa no tempo $t+1$. A Figura 5.1f mostra o resultado do processo de segmentação de mudança (retângulo preto da Figura 5.1e) usando o algoritmo guloso proposto neste trabalho, com um destaque na mudança. Pode-se notar a segmentação da mudança de uma forma adequada, junto com a segmentação dos pontos, utilizando a probabilidade *a posteriori*.

Com estes mesmos dados, foram obtidos alguns resultados interessantes, mostrando a vantagem da utilização da simplificação no contexto de nuvem de pontos corrompida por ruído. A Figura 5.2 mostra duas *GMM*s obtidas pelos dados da Figura 5.1c. A Figura 5.2a é a representação em forma de Gaussianas dos dados em azul da Figura 5.1c, enquanto os dados em vermelho são representados pela Figura 5.2b, ou seja, a representação da nuvem de pontos simplificada. Devido à presença forte de ruído, o algoritmo de determinação do número ótimo de Gaussianas apresenta dificuldade, encontrando assim, mais Gaussianas que o necessário. Por outro lado, a nuvem de pontos simplificada, devido à simplificação e redução quantitativa do ruído, torna possível a determinação do número exato de Gaussianas.

Outro resultado interessante, para o mesmo caso de teste, mostra a probabilidade *a posteriori* da nuvem de pontos dada uma Gaussiana. A Figura 5.3 ilustra esse caso por meio de diferentes cores para os pontos, segundo escala de cores mostrada na própria figura. Um aspecto a ser destacado é que essa probabilidade permite determinar a relação de vizinhança entre Gaussianas. Essa relação é importante para o método *split-and-merge* para determinar agrupamentos candidatos a união, como mencionado na Seção 3.3.3. Na região de intersecção entre a esfera (ao centro) e o plano (abaixo da esfera), existem diversos pontos com probabilidades intermediárias. Essas probabilidades permitem que, conhecendo a Gaussiana que melhor representa cada ponto, ou seja, com maior probabilidade, pode-se inferir tal relação. Resumindo, após a determinação inicial a qual Gaussiana um dado ponto pertence (maior probabilidade), é possível determinar a relação de vizinhança, caso este ponto tenha probabilidades menores de pertencer a outras Gaussianas, então essas Gaussianas são vizinhas.

Neste trabalho foram obtidos inicialmente três conjuntos de dados reais adquiridos utilizando a montagem estática (*laser scanner 2D* e unidade de *pan-tilt*) para validar a presente metodologia. O primeiro, mostrado na Figura 5.4, é apresentando seguindo a mesma ordem dos resultados simulados, porém, mostrando uma imagem do local de teste, visto a dificuldade de visualização da nuvem de pontos 3D. A Figura

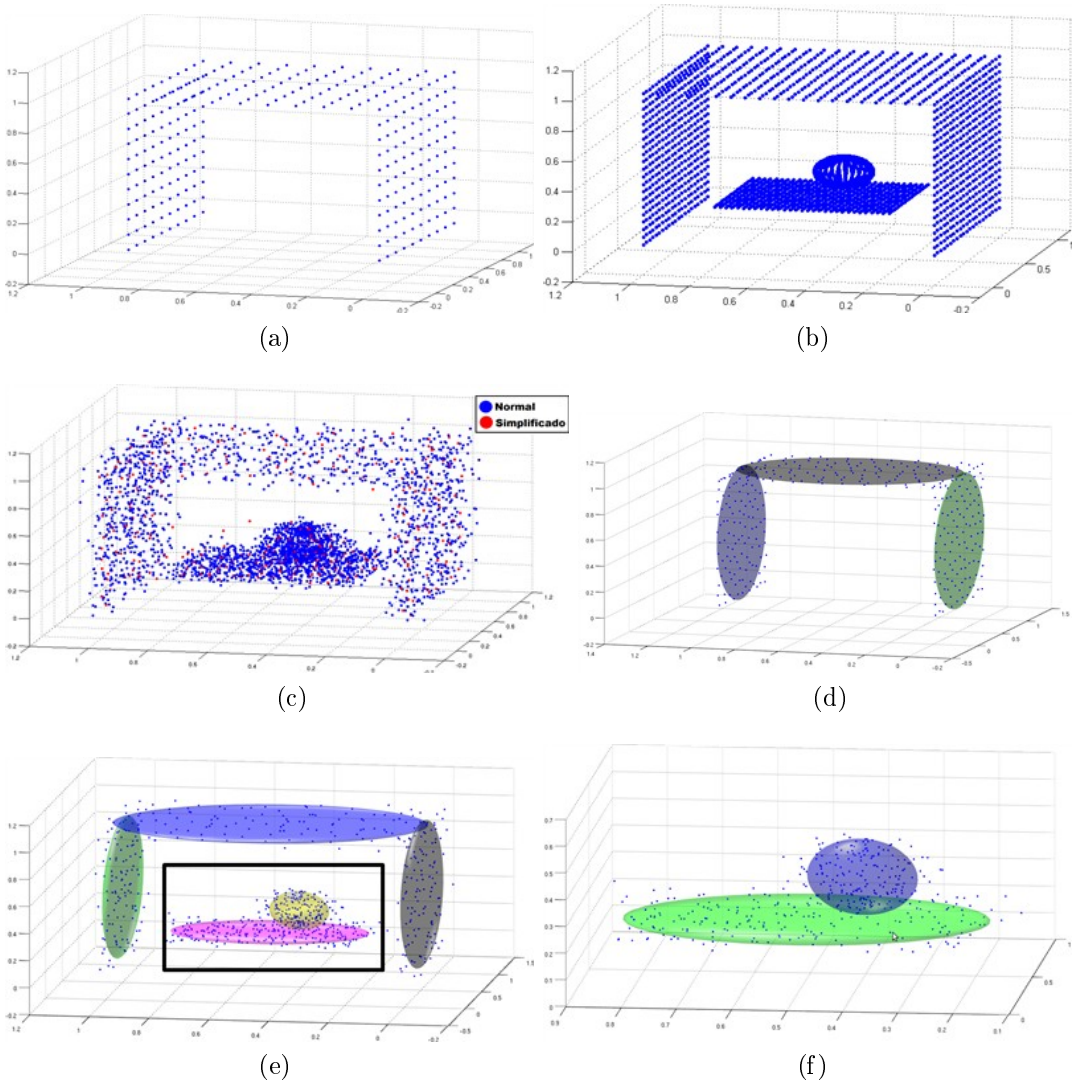
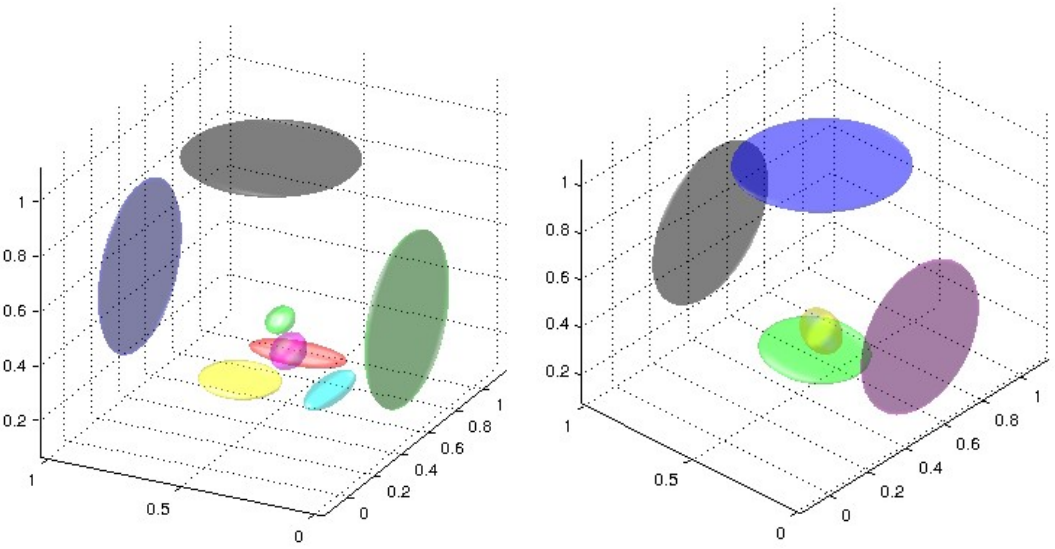


Figura 5.1. Exemplo de detecção de mudanças em dados simulados. As figuras mostram: (a) e (b) os mapas simulados sem e com mudança, respectivamente; (c) o mapa com mudança após adição de ruído (em azul) e simplificado (vermelho); (d) e (e) os resultados do processo de agrupamento utilizando *GMM* para os mapas sem e com mudança, respectivamente; (f) destaque na mudança detectada pelo algoritmo proposto (retângulo preto em (e)).

5.4a ilustra uma imagem do ambiente após a inserção da mudança (uma pessoa ao centro do corredor). As figuras 5.4b e 5.4c, mostram a representação dos pontos simplificados, em azul, e das Gaussianas representadas por elipsóides coloridos. Na Figura 5.4c, assim como na Figura 5.1e, foi adicionado um retângulo preto para destacar a mudança existente na nuvem de pontos. Por fim, a Figura 5.4d mostra o resultado do algoritmo guloso de segmentação de mudanças, no qual a mudança (nuvem de pontos representando a pessoa) foi corretamente segmentada por uma Gaussiana. Embora a



(a) (b)

Figura 5.2. Diferença na determinação do número de Gaussianas para o agrupamento por *GMM*, em (a) sem simplificação nos dados (nuvem completa) e em (b) com dados simplificados como entrada.

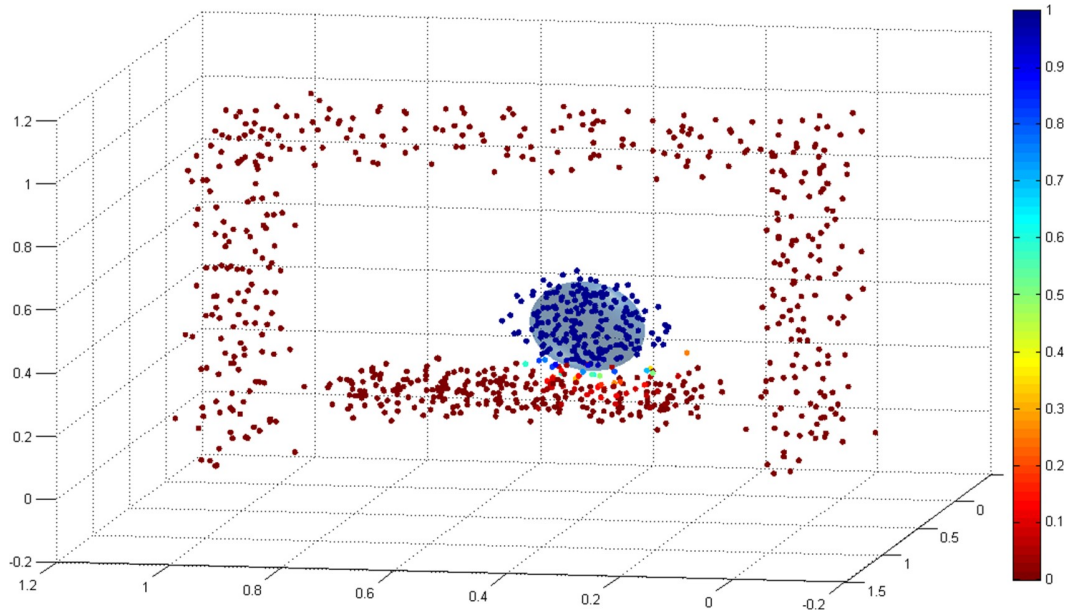
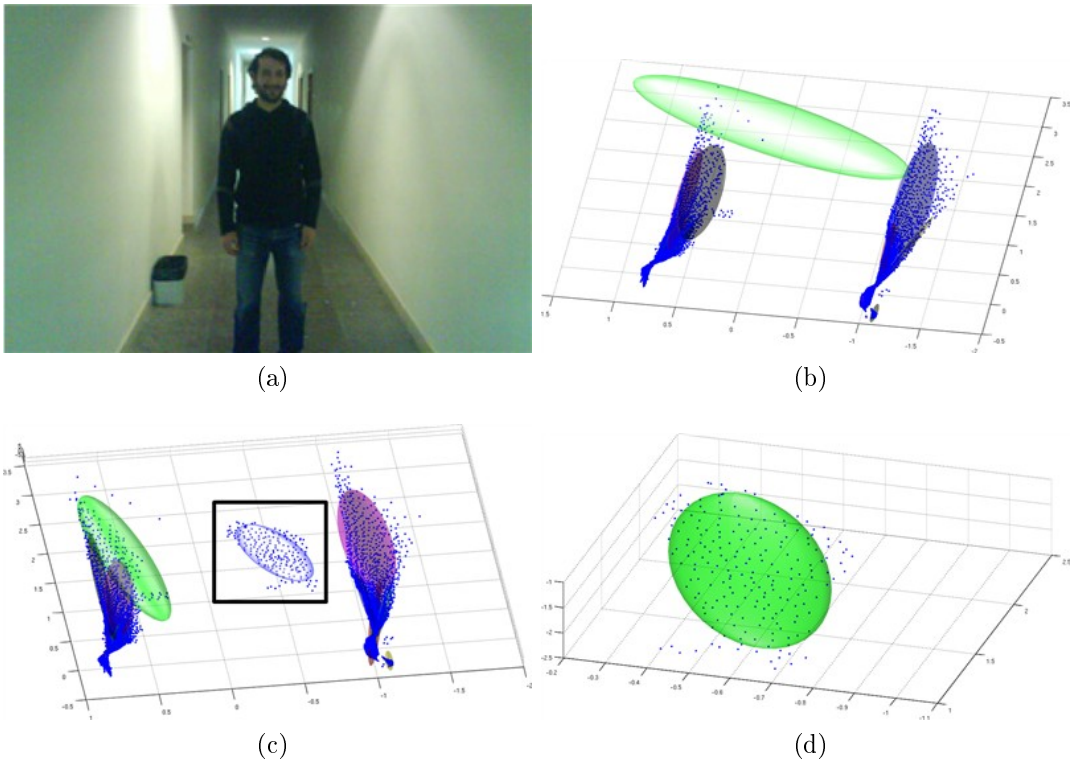


Figura 5.3. Probabilidade de cada ponto pertencer à Gaussiana mostrada ao centro, considerando a escala de cor mostrada à direita, que representa, em tons mais azulados, as altas probabilidades e, em tons avermelhados, as baixas probabilidades.

segmentação tenha sido correta, foi determinada apenas uma Gaussiana para a mudança, que poderia ser representada por mais de uma Gaussiana, caso se considerasse que o ideal seria dividir uma forma humana em cabeça, tronco e os membros. Esse pro-

cesso, no entanto, será realizado pelo algoritmo *split-and-merge*, nos passos posteriores da metodologia.



(a) (b) (c) (d)

Figura 5.4. Detecção de mudanças em dados reais - Conjunto de dados 1. As figuras mostram: (a) imagem representando o ambiente utilizado nos experimentos com a presença da mudança (pessoa ao centro do corredor); (b) e (c) os resultados do processo de agrupamento utilizando *GMM* juntamente com a nuvem de pontos (em azul) para os mapas sem e com mudança, respectivamente; (d) destaque na mudança detectada pelo algoritmo (retângulo preto em (c)).

O segundo conjunto de dados é ilustrado na Figura 5.5. A Figura 5.5a mostra a imagem do ambiente após a inserção da mudança, no caso a mudança é a porta "semi" aberta. No mapa no tempo t, a porta está completamente aberta, como ilustra a Figura 5.5b, na qual se pode ver a abertura da porta, além das Gaussianas que representam o conjunto de pontos. Na Figura 5.5c é mostrado o mapa no tempo $t+1$ (com mudança), no qual a porta aparece, como na Figura 5.5a, "semi' aberta. Esse fato facilita a detecção, que foi destacada pelo retângulo preto na Figura 5.5c, visto que a abertura não era representada e, então, passa a ser representada por uma Gaussiana. A Figura 5.5d ilustra o resultado obtido na detecção de mudança, junto com o conjunto de pontos segmentado pelo algoritmo, em azul. Devido à presença de ruído e diferenças existentes entre os dois mapas, pode-se notar os diferentes agrupamentos gerados para representar o mapa no tempo t e no tempo $t+1$. Apesar disso, a detecção de mudança

foi realizada com sucesso.

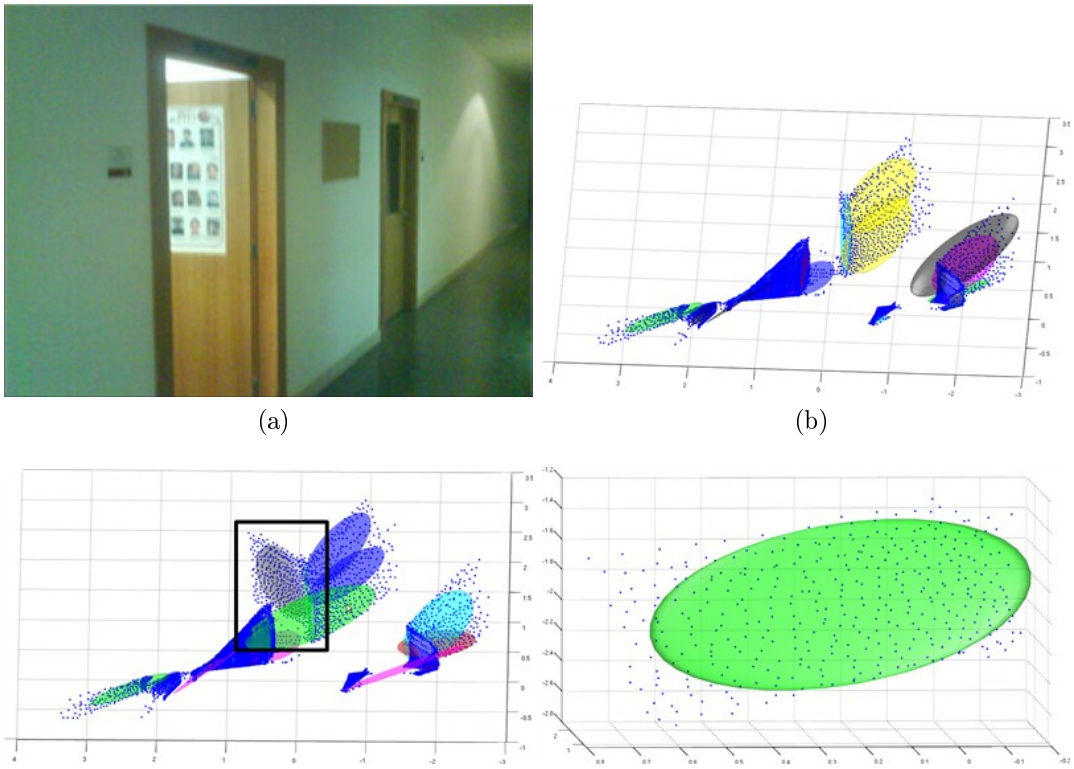
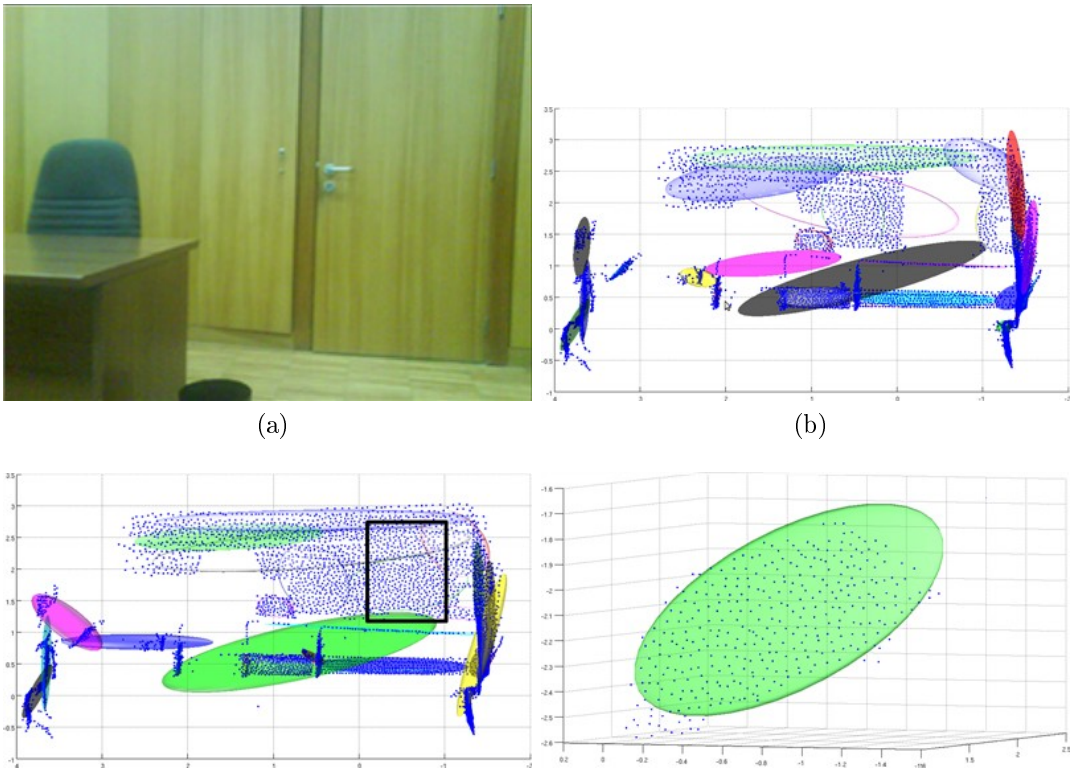


Figura 5.5. Detecção de mudanças em dados reais - Conjunto de dados 2. As figuras mostram: (a) imagem representando o ambiente utilizado nos experimentos com a presença da mudança (porta "semi' aberta); (b) e (c) os resultados do processo de agrupamento utilizando *GMM* juntamente com a nuvem de pontos (em azul) para os mapas sem e com mudança, respectivamente; (d) destaque na mudança detectada pelo algoritmo (retângulo preto em (c)).

O terceiro conjunto de dados reais utilizados nos testes é mostrado na Figura 5.6. Primeiramente, na Figura 5.6a, uma imagem do ambiente de teste é mostrada. Esse ambiente é uma sala de escritório no qual a mudança é o fechamento da porta que anteriormente estava aberta. As figuras 5.6b e 5.6c, mostram os mapas gerados, inicialmente com a porta aberta, e posteriormente, com a porta fechada. Essa mudança é de difícil detecção, visto que obviamente há ausência de pontos no vão da porta, com a presença deles próximos à parede, sendo classificados, como um único objeto (parede e porta). Posteriormente, a leitura captura apenas uma superfície uniforme, porém, devido às saliências detectadas pelo *laser scanner*, a porta no mapa em $t+1$ será representada por uma Gaussiana, mostrada em cor verde claro. Na Figura 5.6b, é mostrada a detecção da mudança, juntamente com a segmentação dos pontos que representam essa mudança.



(a) (b)

(c) (d)

Figura 5.6. Detecção de mudanças em dados reais - Conjunto de dados 3. As figuras mostram: (a) imagem representando o ambiente utilizado nos experimentos com a presença da mudança (porta fechada); (b) e (c) os resultados do processo de agrupamento utilizando *GMM* juntamente com a nuvem de pontos (em azul) para os mapas sem e com mudança, respectivamente; (d) destaque na mudança detectada representada pela Gaussiana verde (retângulo preto em (c)), bem como pelos pontos (em azul).

Outro resultado importante é mostrado na Tabela 5.1. Ela apresenta dados de desempenho e de qualidade de detecção de mudanças, tanto para dados simulados, quanto para dados reais. A melhora alcançada pelo processo de simplificação foi significativa. Como mencionado anteriormente, os resultados são de três conjuntos de dados reais e a média dos trinta cenários simulados, com repetição de dez vezes cada para aquisição dos tempos de execução. As colunas mostram o número de pontos no mapa sem mudança (tempo t) e com mudança (tempo $t+1$), tempo médio de execução, que inclui o tempo de execução do algoritmo de simplificação e do algoritmo de obtenção da *GMM* para os dois mapas, além do algoritmo guloso para segmentação da mudança usando *EMD*. Por fim, a última coluna a porcentagem de pontos corretamente segmentados. O *ground truth* para o processo de detecção de mudança foi gerado manualmente. A simplificação mostrou melhores resultados em dados reais do que em dados simulados, porque como nos dados simulados o ruído inserido é maior que o ruído típico do sensor

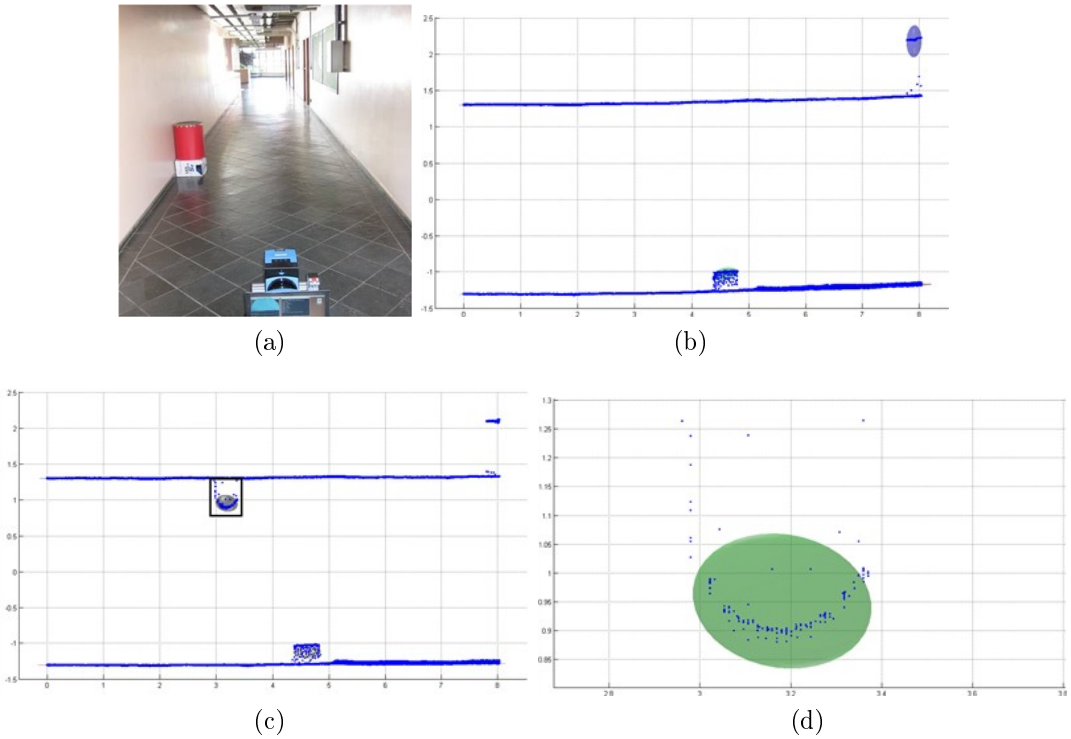
que capturou os dados reais. Além disso, nos dados reais os objetos estão mais "distantes" entre si. Nos dados simulados, foram colocadas mudanças próximas às paredes do corredor, o que gerou uma grande dificuldade de detecção. Nos dados reais, o ganho em tempo de execução foi de $4,84$ vezes em média, enquanto que o ganho com dados simulados foi de $9,62$ vezes em média. Essa diferença é devido a limitação imposta ao método de determinação de agrupamentos *GMM* que computa um número máximo de Gaussianas, limitando, assim, o custo computacional do método. Caso esse número fosse um valor muito grande, possivelmente o ganho em tempo de execução com o uso do método de simplificação seria parecido entre dados reais e simulados.

Tabela 5.1. Análise comparativa de detecção de mudança com simplificação da nuvem de pontos

		Num. Pontos t	Num. Pontos $t+1$	Tempo Exec.(s)	Taxa de Acerto(%)
Dados Simulados	Simplificado	362	805	0,21	92,3%
	Completo	1322	2960	2,02	93,4%
Dados Reais 1	Simplificado	8575	9251	91,83	92,6%
	Completo	31786	34003	445,67	91,5%
Dados Reais 2	Simplificado	9354	9142	119,68	93,6%
	Completo	34453	33696	576,32	91,4%
Dados Reais 3	Simplificado	7884	8286	100,75	88,3%
	Completo	29302	30722	490,85	86,2%

Mais três conjuntos de dados reais foram adquiridos utilizando a montagem dinâmica (*laser scanner 2D* e robô móvel) com a finalidade de validar a presente metodologia. Esses três conjuntos foram denominados durante este trabalho de: **Conjunto de dados 4**, tendo como mudança um cilindro; **Conjunto de dados 5**, a presença de uma caixa e **Conjunto de dados 6**, no qual foi inserida uma pessoa como mudança. O Conjunto de dados 4, mostrado na Figura 5.7, é apresentado seguindo a mesma ordem dos resultados anteriores. A Figura 5.7a ilustra uma imagem do ambiente após a inserção da mudança (um cilindro vermelho no corredor, à esquerda). Nessa figura pode-se notar que abaixo do cilindro existe uma caixa, essa foi inserida devido a diferença de altura entre o chão e o *laser scanner 2D* preso ao robô, de forma a permitir a leitura completa do cilindro pelo *laser scanner*. As figuras 5.7b e 5.7a, mostram uma vista superior de cada um dos mapas 3D obtidos (com e sem mudança), onde são ilustradas a nuvem de pontos simplificada, por pontos azuis, e as Gaussianas, por elipsóides coloridos. Para a obtenção dos resultados foi efetuada uma restrição nos dados, na qual somente pontos com altura menor que um $1,5$ metros foram utilizados (em relação a leitura do *laser scanner*. Na Figura 5.7c foi adicionado um retângulo preto para destacar a mudança existente na nuvem de pontos. Por fim, a Figura 5.7d mostra um "zoom" no resultado da segmentação da mudança, no qual a mudança foi

corretamente segmentada por uma Gaussiana, na cor verde. Na Figura 5.7c, a mesma Gaussiana foi mostrada em cinza (retângulo preto).



(a) (b)

(c) (d)

Figura 5.7. Detecção de mudanças em dados reais - Conjunto de dados 4. As figuras mostram: (a) imagem representando o ambiente utilizado nos experimentos com a presença da mudança (cilindro à esquerda); (b) e (c) os resultados do processo de agrupamento utilizando *GMM* juntamente com a nuvem de pontos (em azul) para os mapas sem e com mudança, respectivamente (vista superior); (d) destaque na mudança detectada pelo algoritmo (retângulo preto em (c)). A unidade de todos os eixos mostrado nos gráficos é o metro.

A restrição de utilizar apenas os pontos com alturas menores que um dado valor pode ser muito restritiva, devido ao fato que normalmente não se conhecer o valor ótimo(limiar). Foi testado o desempenho do sistema de detecção de mudança considerando a nuvem de pontos completa para o Conjunto de dados 4 (todos os pontos). A Figura 5.8 ilustra os resultados obtidos. As figuras 5.8a e 5.8b mostram os mapas tridimensionais obtidos (com em sem mudança), considerando uma vista frontal. Nesses mapas, pode-se notar as características do teto do corredor, no qual foram detectadas muitas Gaussianas, mostrada pelos elipsóides coloridos. Na Figura 5.8b foi adicionado um retângulo preto para destacar a mudança existente na nuvem de pontos. Por fim, as figuras 5.8c e 5.8d ilustram a segmentação da mudança efetuada pelo algoritmo proposto. A Figura 5.8d mostra um "zoom" na mudança para permitir uma melhor vi-

sualização. Pode-se notar que mesmo com a presença do teto nos mapas 3D, o método conseguiu segmentar corretamente a mudança.

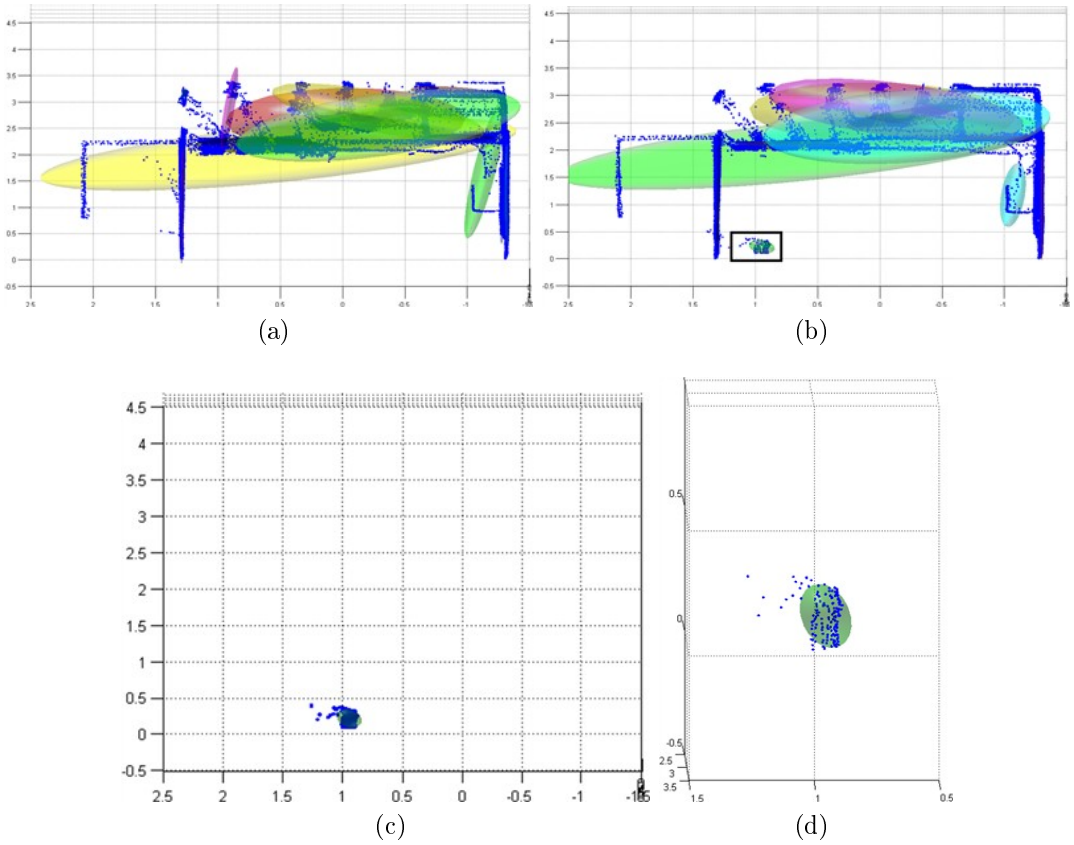


Figura 5.8. Detecção de mudanças em dados completos - Conjunto de dados 4. As figuras mostram: (a) e (b) os resultados do processo de agrupamento utilizando *GMM* juntamente com a nuvem de pontos (em azul) para os mapas sem e com mudança, respectivamente (vista frontal); (c) e (d) a mudança detectada pelo algoritmo (retângulo preto em (b)), com "zoom" na mudança em (d). A unidade de todos os eixos mostrados nos gráficos é o metro.

Os resultados obtidos com o Conjunto de dados 5 são mostrados na Figura 5.9, esses são apresentados seguindo a mesma ordem dos resultados anteriores. A Figura 5.9a ilustra uma imagem do ambiente após a inserção da mudança (uma caixa no corredor em destaque na cor vermelha, à esquerda). Nessa figura pode-se notar a existência uma outra caixa abaixo, assim como no caso anterior, essa foi inserida devido à diferença de altura entre o chão e o *laser scanner*, de forma a permitir a leitura completa da caixa de interesse pelo sensor. As figuras 5.9b e 5.9a, mostram uma vista superior de cada um dos mapas 3D obtidos (com e sem mudança), onde são ilustradas a nuvem de pontos simplificada, por pontos azuis, e as Gaussianas, por elipsóides coloridos. Para a obtenção dos resultados foi efetuada uma restrição nos

dados, assim como no exemplo anterior, na qual somente pontos com altura menor que um $1,5$ metros foram utilizados. Na Figura 5.9c foi adicionado um retângulo preto para destacar a mudança existente na nuvem de pontos. Por fim, a Figura 5.9d mostra um "zoom" no resultado da segmentação da mudança, no qual a mudança foi corretamente segmentada por uma Gaussiana, na cor azul. Na Figura 5.9c, a mesma Gaussiana foi mostrada na cor amarela (retângulo preto).

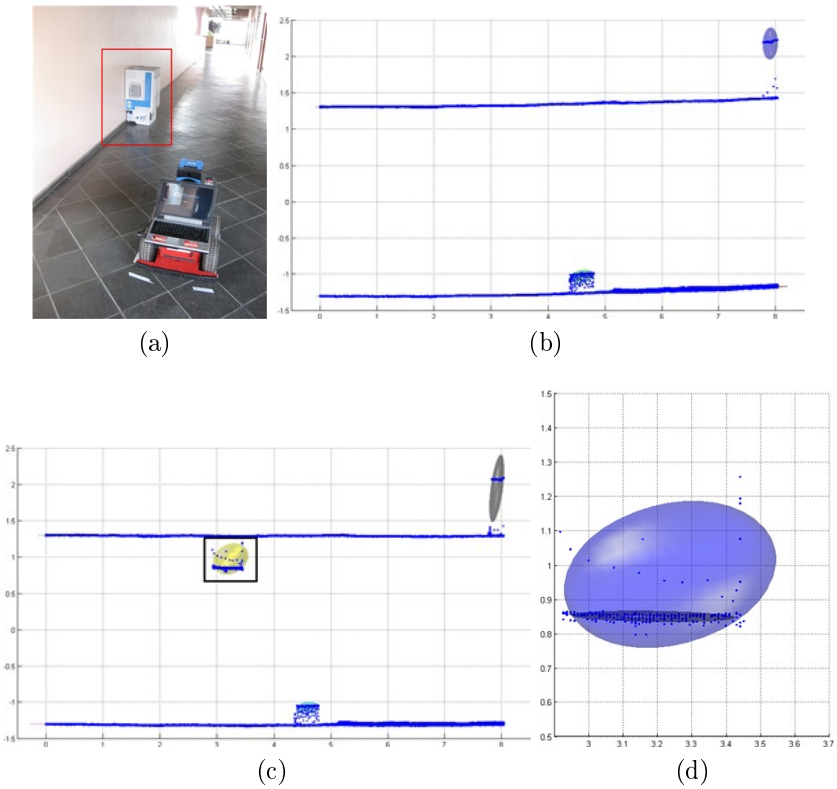
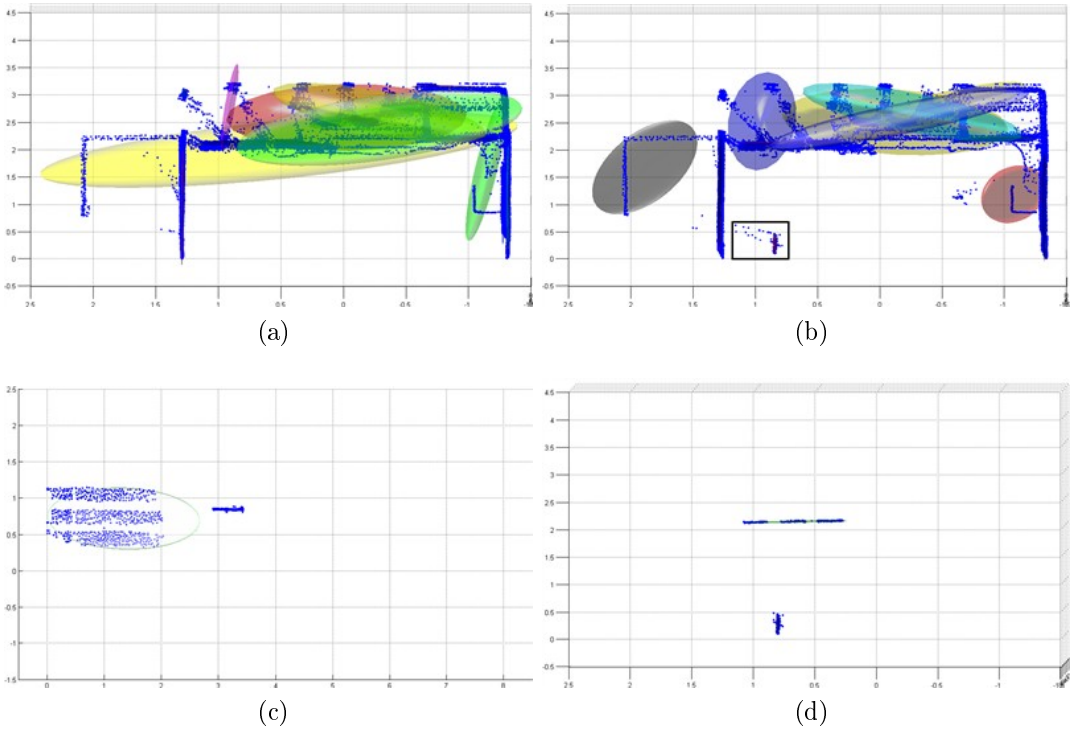


Figura 5.9. Detecção de mudanças em dados reais - Conjunto de dados 5. As figuras mostram: (a) imagem representando o ambiente utilizado nos experimentos com a presença da mudança (caixa à esquerda); (b) e (c) os resultados do processo de agrupamento utilizando *GMM* juntamente com a nuvem de pontos (em azul) para os mapas sem e com mudança, respectivamente (vista superior); (d) destaque na mudança detectada pelo algoritmo (retângulo preto em (c)). A unidade de todos os eixos mostrados nos gráficos é o metro.

A restrição de utilizar apenas os pontos com alturas menores que um dado valor foi retirada, assim como no exemplo anterior, para o Conjunto de dados 5. Foi testado o desempenho do sistema de detecção de mudança considerando a nuvem de pontos completa (todos os pontos). Os resultados obtidos são mostrados na Figura 5.10. As figuras 5.10a e 5.10b mostram os mapas 3D obtidos (com e sem mudança), considerando a vista frontal. Nesses mapas, pode-se notar as características do teto do corredor, que são minimamente diferentes devido à trajetória realizada pelo robô móvel durante a

aquisição dos dados. Na Figura 5.10b foi adicionado um retângulo preto para destacar a mudança real existente na nuvem de pontos. Por fim, as figuras 5.10c e 5.10d ilustram a segmentação da mudança efetuada pelo algoritmo proposto. A Figura 5.10c mostra a vista superior, ilustrando a detecção da caixa e, também, de parte do teto. Isso se deve a diferença na posição que foi iniciada a aquisição de dados. Visto que não se tem um método de registro, esta parte do teto foi detectada como uma mudança. A Figura 5.10d mostra o mesmo resultado utilizando uma vista frontal.



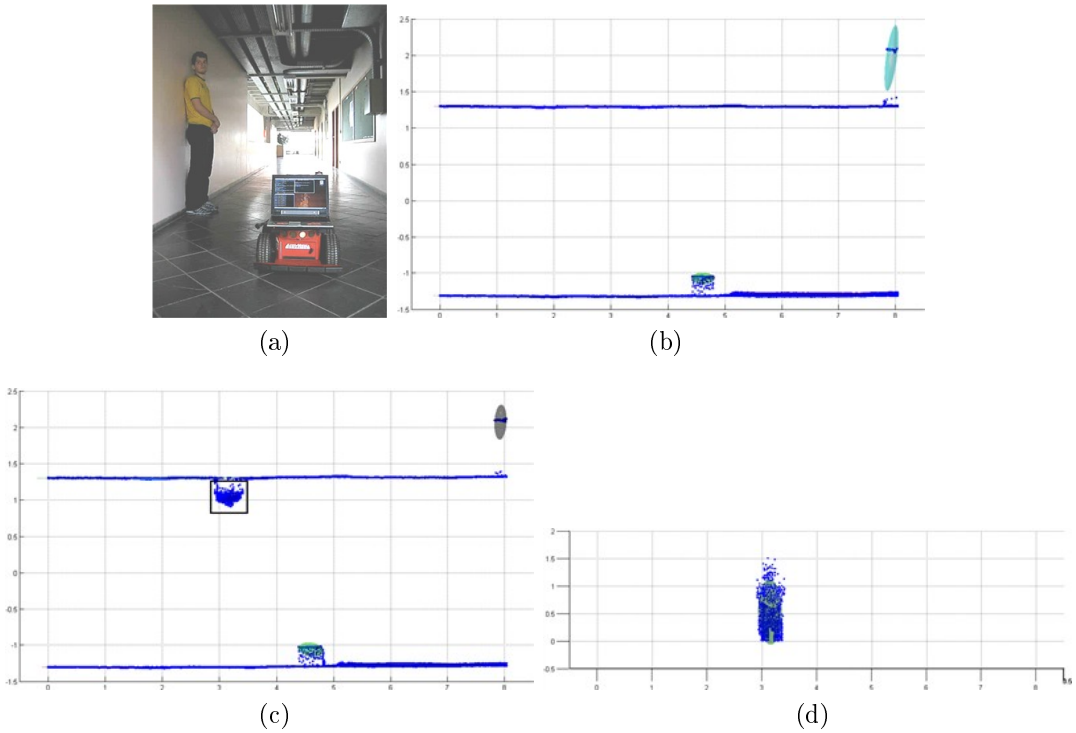
(a) (b)

(c) (d)

Figura 5.10. Detecção de mudanças em dados completos - Conjunto de dados 5. As figuras mostram: (a) e (b) os resultados do processo de agrupamento utilizando *GMM* juntamente com a nuvem de pontos (em azul) para os mapas sem e com mudança, respectivamente (vista frontal); (c) e (d) a mudança detectada pelo algoritmo (retângulo preto em (b) e parte do teto), com vista superior em (c) e frontal em (d). O algoritmo detectou uma região do teto como mudança devido à diferença entre as posições iniciais de aquisição de dados para o robô. A unidade de todos os eixos mostrados nos gráficos é o metro.

Para o Conjunto de dados 6, os resultados são mostrados na Figura 5.11, esses são apresentados seguindo a mesma ordem dos resultados anteriores. A Figura 5.11a ilustra uma imagem do ambiente após a inserção da mudança (uma pessoa, à esquerda). As figuras 5.11b e 5.11a, mostram uma vista superior de cada um dos mapas 3D obtidos (com e sem mudança), onde são ilustradas a nuvem de pontos simplificada, por pontos azuis, e as Gaussianas, por elipsóides coloridos. Para a obtenção dos resultados foi

efetuada uma restrição aos dados, assim como nos exemplos anteriores, na qual somente pontos com altura menor que um $1,5$ metros foram utilizados (em relação a altura do *laser scanner*). Na Figura 5.11c foi adicionado um retângulo preto para destacar a mudança existente na nuvem de pontos. Por fim, a Figura 5.11d mostra a vista lateral do resultado da segmentação da mudança, no qual a mudança foi corretamente segmentada por uma Gaussiana, na cor verde. Na Figura 5.11c, a mesma Gaussiana foi mostrada na cor azul (retângulo preto).



(a) (b) (c) (d)

Figura 5.11. Detecção de mudanças em dados reais - Conjunto de dados 6. As figuras mostram: (a) imagem representando o ambiente utilizado nos experimentos com a presença da mudança (pessoa à esquerda); (b) e (c) os resultados do processo de agrupamento utilizando *GMM* juntamente com a nuvem de pontos (em azul) para os mapas sem e com mudança, respectivamente (vista superior); (d) destaque na mudança detectada pelo algoritmo (retângulo preto em (c)) em vista lateral. A unidade de todos os eixos mostrados nos gráficos é o metro.

A restrição de utilizar apenas os pontos com altura menor que um dado valor também foi desconsiderada para o Conjunto de dados 6. Foi testado o desempenho do sistema de detecção de mudança considerando a nuvem de pontos completa (todos os pontos). Os resultados obtidos são mostrados na Figura 5.12. As figuras 5.12a e 5.12b mostram os mapas 3D obtidos (com e sem mudança, considerando uma vista frontal. Nesses mapas, pode-se notar as características do teto do corredor, que são diferentes devido à trajetória realizada pelo robô móvel durante a aquisição dos dados. Na Figura

5.12b foi adicionado um retângulo preto para destacar a mudança real existente na nuvem de pontos. Por fim, as figuras 5.12c e 5.12d ilustram a segmentação da mudança efetuada pelo algoritmo proposto. A Figura 5.12c mostra a vista superior, ilustrando a detecção da pessoa e, também, de parte do teto. Isso se deve a diferença na posição que foi iniciada a aquisição de dados, bem como uma variação na trajetória do robô móvel entre os dois mapas, devido à imprecisão da odometria, que obtêm um mapa do teto diferente devido aos vários desníveis existentes (ver Figura 5.11a). Assim, como não se tem um método para efetuar o registro entre os mapas, parte do teto foi detectada como uma mudança. A Figura 5.12d mostra o mesmo resultado sob um ponto de vista frontal.

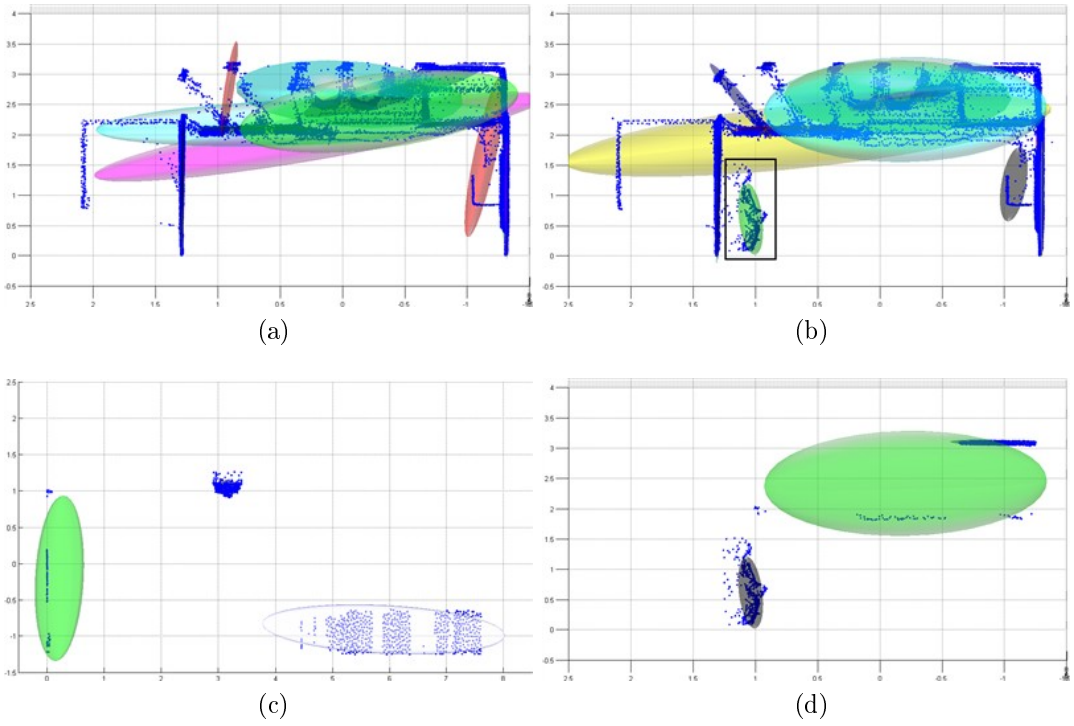


Figura 5.12. Detecção de mudanças em dados completos - Conjunto de dados 6. As figuras mostram: (a) e (b) os resultados do processo de agrupamento utilizando *GMM* juntamente com a nuvem de pontos (em azul) para os mapas sem e com mudança, respectivamente (vista frontal); (c) e (d) as mudanças detectadas pelo algoritmo (retângulo preto em (b)), com vista superior em (c) e frontal em (d). O algoritmo detectou duas regiões do teto como mudança devido à diferença entre as posições iniciais de aquisição dos mapas pelo robô, bem como uma variação de trajetória entre as aquisições dos dois mapas 3D. A unidade de todos os eixos mostrados nos gráficos é o metro.

5.2 Recuperação de Forma

Como apresentado nesse texto, foram propostos dois métodos para recuperação de forma; o primeiro utilizando formas básicas e o segundo, utilizando superquádricas.

Os resultados do método de recuperação de formas básicas no espaço de Gaussianas, visam validar o método frente a um algoritmo muito difundido na literatura da área usando *RANSAC* [Schnabel et al., 2007]. Um estudo comparativo quanto à exatidão e o custo computacional será realizado entre os dois métodos, na Seção 5.2.1.

Os resultados do método de recuperação de formas utilizando superquádricas visa validar a metodologia, mostrando a expressividade das superquádricas, bem como a eficiência do *split-and-merge* para determinação do conjunto de formas básicas, a fim de representar objetos complexos. Além disso, os resultados mostram as vantagens das adaptações feitas ao método, para adequação com o uso de *GMM*.

5.2.1 Formas Básicas

A fim de avaliar o algoritmo proposto foram obtidos resultados experimentais com dados reais e simulados. Foram obtidos mais dados simulados para a validação do algoritmo de recuperação de formas básicas do que para a validação do algoritmo de detecção de mudança. Para a obtenção de resultados estatísticos foram gerados cem casos de teste (ao invés de trinta), com cada forma básica de diferentes tamanhos e em poses distintas. Para cada um dos três conjunto de dados reais utilizados nos experimentos (Conjunto de dados 1, 2 e 3), foram realizadas dez leituras a fim de se obter resultados estatísticos. Assim, se obteve trinta conjunto de dados. Os dados reais usando a montagem dinâmica (robô móvel) não foram testados com formas básicas.

Os resultados experimentais da recuperação de formas básicas, como já mencionado, tiveram foco na exatidão do método e no tempo de processamento de ambos os métodos. Assim, a exatidão foi definida, como:

$$\begin{aligned} Verdadeiros &= \frac{Num\_de\_Formas\_Corretas}{Num\_Total} \\ Falsos\_Verdadeiros &= \frac{Num\_de\_Formas\_Erradas}{Num\_Total}, \end{aligned} \tag{5.1}$$

onde $Num\_de\_Formas\_Corretas$ é o número de formas recuperadas corretamente, Num_Total é o número total de formas, e $Num\_de\_Formas\_Erradas$ é o número de recuperações incorretas. Existem, também, alguns casos em que não foi possível realizar a recuperação da forma devido, principalmente, a problemas de segmentação. Para determinar a precisão do algoritmo, as formas obtidas foram comparadas com a pose e escala do objeto original. Os resultados foram obtidos automaticamente e mos-

Tabela 5.2. Análise comparativa entre algoritmos de recuperação de formas básicas em dados simulados.

Algoritmo	Espaço Euclidiano	Espaço das Gaussianas
Tempo de Execução (s)	0,3491	0,07425
$Verdadeiros$	0,8330	0,9367
$Falsos\_Verdadeiros$	0,1467	0,0400
$\sigma_{\Delta E(E_x,E_y,E_z)}$ [cm]	(12,13; 11,01; 8,09)	(9,11; 7,998; 7,11)
$\sigma_{\Delta R(\psi,\vartheta,\varphi)}$ [graus]	(2,01; 1,12; 3,22)	(1,32; 1,99; 2,88)

trados nas tabelas 5.2 e 5.3 (dados simulados e reais, respectivamente). $\sigma_{\Delta E(E_x,E_y,E_z)}$ e $\sigma_{\Delta R(\psi,\vartheta,\varphi)}$ representam, respectivamente, a variância nos dados de escala e de rotação recuperados pelo algoritmo em relação à referência, respectivamente.

A Tabela 5.2 resume os resultados para recuperação de formas básicas utilizando dados simulados. Os dados mostram a diferença no tempo de processamento entre os métodos. Essa diferença se deve ao fato de que o tempo gasto pelo *RANSAC* é dependente do número de pontos, enquanto o tempo para o método proposto no espaço de Gaussianas depende da diferença entre a estimação inicial do método de minimização e a estimação da forma a ser recuperada. Como o tempo de computação do passo inicial da detecção de mudança é o mesmo para os dois métodos, ele não altera o resultado e, portanto, não foi incluído no somatório dos tempos. Como já mencionado, a implementação do algoritmo de recuperação de forma básica no espaço de Gaussianas foi feita utilizando uma versão compilada do software desenvolvido em Matlab [The Mathworks Inc., 2009], enquanto o restante foi implementado em C/C++.

Algumas razões para falhas do algoritmo foram: a) estimação errada fornecida pelo algoritmo de determinação de *GMM* (*EM*), devido à proximidade entre agrupamentos, como por exemplo a mudança próxima à parede do corredor simulado; b) representação da matriz de covariância próxima da singularidade e c) falha no processo de convergência. Além disso, o erro médio associado à forma recuperada é maior para o algoritmo no espaço Euclidiano do que para o algoritmo proposto no para o espaço de Gaussianas. Uma consideração importante é que os piores resultados para o método *RANSAC* foram obtidos com o cilindro simulado, enquanto para o espaço de Gaussianas foram obtidos com o plano simulado.

Inicialmente, dois exemplos de dados simulados com apenas um objeto são apresentados (Figura 5.13). A Figura 5.13a mostra o mapa no tempo $t+1$, com destaque para a mudança, em vermelho, que mostra um cilindro ideal, com a adição de ruído, e sem o uso de simplificação. Já a Figura 5.13b mostra o mesmo tipo de exemplo,

porém, sendo que a mudança foi um plano bem próximo a parede do corredor, também em vermelho. Nas figuras 5.13c e 5.13d são mostrados os resultados da recuperação de forma utilizando o espaço de Gaussianas (em vermelho) e no espaço Euclidiano com algoritmo *RANSAC* (em azul). Na Figura 5.13c, pode-se notar a existência de uma pequena inclinação na determinação da forma no espaço Euclidiano, isso se deve ao fato do processo de reamostragem para converter os dados do espaço de Gaussianas para o espaço Euclidiano apresentar uma pequena perda de informação devido a reamostragem, resultado este já discutido na Seção 3.4.1.1. Outro resultado foi obtido na recuperação de forma de um plano, como mostra a Figura 5.13d. Embora o resultado seja adequado visualmente, o plano não foi recuperado devido ao ruído adicionado, que faz com que os pontos não estejam mais em um plano e sim distribuídos, como mostra a Figura 5.13d. Nesse exemplo as formas recuperadas pelos dois tipos de algoritmos foram de cilindros.

A Figura 5.14 mostra, em detalhes, os resultados do algoritmo iterativo proposto para recuperação de forma no espaço de Gaussianas, e descrito em detalhes na Seção 3.4.1.2, bem como o resultado do processo de reamostragem para o algoritmo no espaço Euclidiano, para as mudanças apresentadas nas figuras 5.13a e 5.13b. Nas figuras 5.14a e 5.14b são ilustrados os pontos gerados pelo processo de reamostragem, utilizados como entrada do algoritmo *RANSAC*. As outras figuras, basicamente, mostram a Gaussiana que representa a mudança (em azul) e a Gaussiana obtida a partir de uma inicialização com uma forma padrão (plano, cilindro ou esfera) e minimizada para "alcançar" a forma da mudança. Assim, as figuras 5.14c e 5.14d, o resultado da Gaussiana que representa a mudança, em azul, e, em vermelho, após o processo de minimização com um plano ideal, no espaço de Gaussianas, variando-se a escala e a rotação. Pode-se notar que, embora a mudança da Figura 5.14b represente um plano, devido ao ruído, a Gaussiana tem valores de escalas maiores que zero em todos os eixos, ou seja, não aparenta visualmente ser um plano, o que torna o ajuste de um plano com os dados uma tarefa muito difícil. Consequentemente, na Figura 5.14d não se obtém um bom ajuste.

Nas figuras 5.14e e 5.14f são mostrados, também, o resultado da Gaussiana que representa a mudança, em azul, e, em vermelho, o resultado após o processo de minimização com uma esfera ideal, variando-se a escala e a rotação. Devido a problemas de escala na imagem, a forma em vermelho parece uma esfera achatada, embora não seja. Pode-se notar que para os dois casos a esfera se aproxima da forma ideal, mas não consegue recuperar a forma de maneira ótima. Por fim, nas figuras 5.14g e 5.14h, o resultado da mudança, em azul, e o resultado após a minimização, em vermelho, para um **cilindro ideal**, convertido em Gaussiana. Pode-se notar que ele se ajusta muito

bem, para os dois casos de teste, tanto para o caso que a entrada é um cilindro (coluna da esquerda), quanto para o plano (coluna da direita). Devido à forte presença de ruído no plano que representava a mudança, o resultado obtido foi um cilindro.

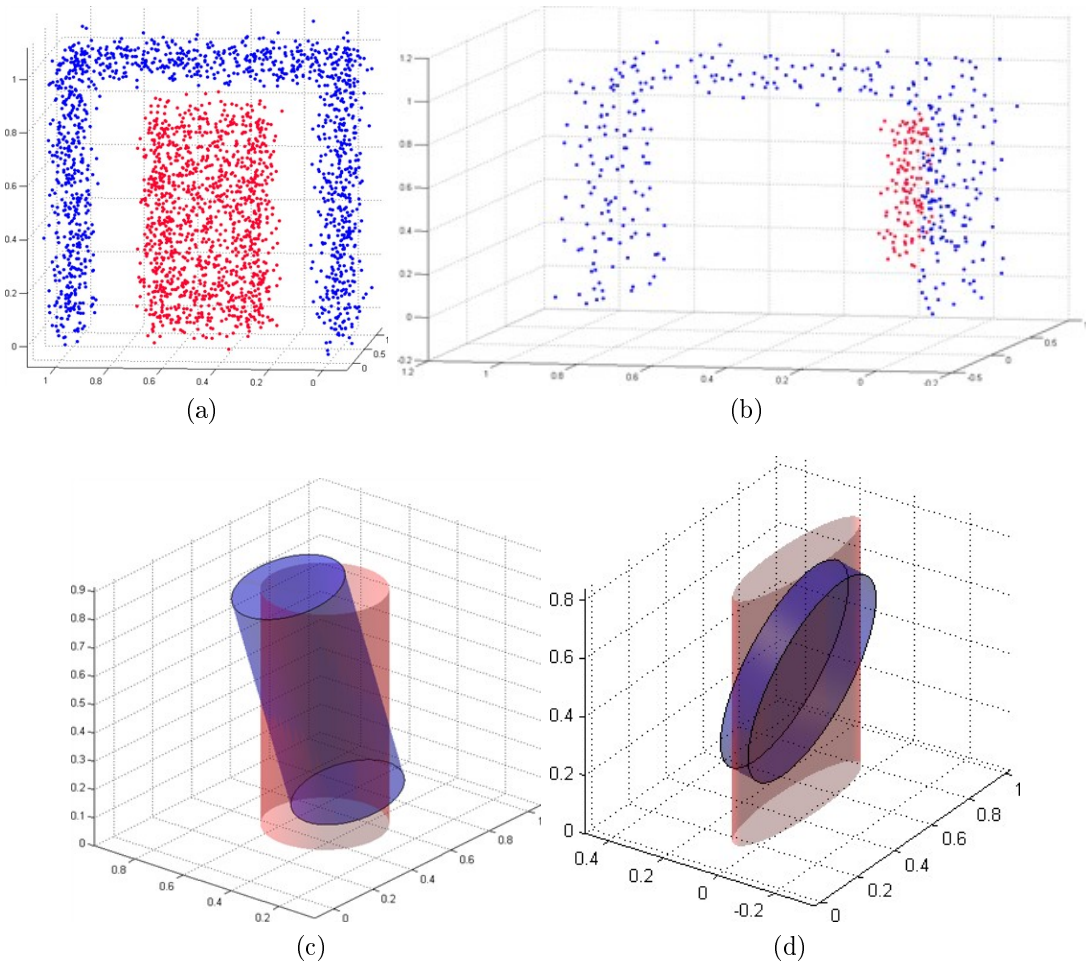


Figura 5.13. Análise comparativa de recuperação de formas básicas para dados simulados. Nas figuras: (a) e (b) mostram as mudanças segmentadas em vermelho; (c) e (d) mostram os resultados da recuperação de forma utilizando os dois algoritmos propostos.

A Figura 5.15 mostra um teste com mudança sendo dois objetos, um plano e uma esfera. A Figura 5.15a apresenta o mapa em $t+1$, com a mudança marcada em vermelho. Os pontos gerados pelo processo de reamostragem são mostrados na Figura 5.15b. Por fim, o resultado mostra muitas semelhanças entre os dois métodos, com pequenas diferenças de escala (Figura 5.15c). Assim, para mundanças compostas, nas quais o processo de *EM* consegue obter a segmentação satisfatória, o resultado obtido é muito bom.

Semelhantemente ao que foi analisado com os dados simulados, foi realizado um

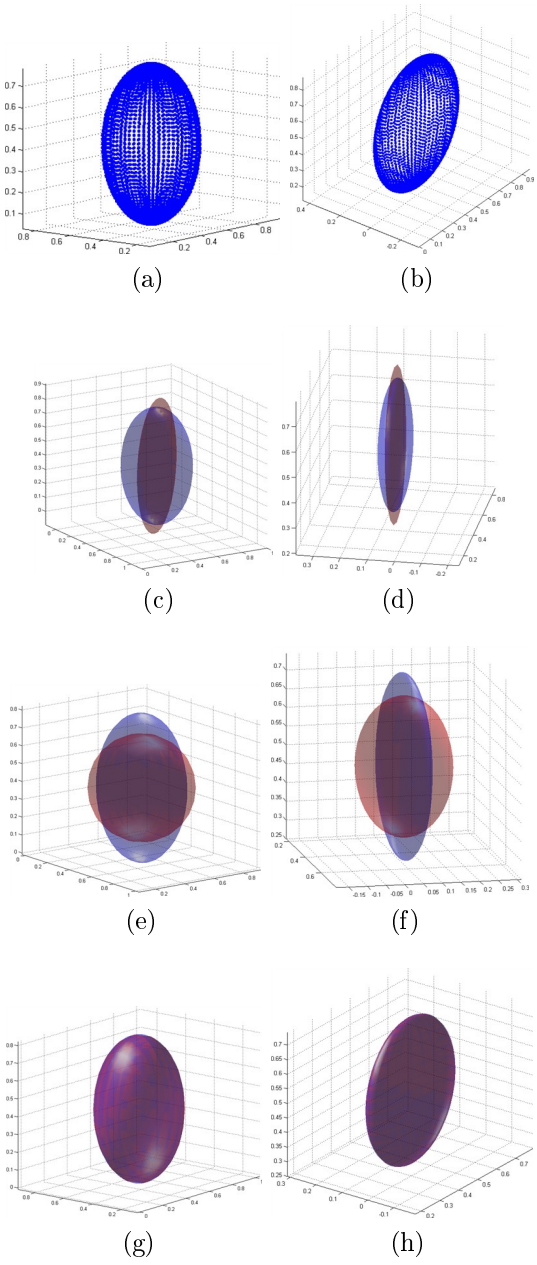


Figura 5.14. Recuperação de formas básicas utilizando o espaço de Gaussianas e o espaço Euclidiano, ilustrando a reamostragem e método iterativo em dados simulados. As figuras mostram: (a) pontos reamostrados de um cilindro no espaço de Gaussiana (mostrado em vermelho na Figura 5.13a); (b) pontos reamostrados de um plano no espaço de Gaussiana (mostrado em vermelho na Figura 5.13b); (c) e (d) representação em Gaussiana da mudança (em azul) e da Gaussiana inicializada como um plano ideal após processo de minimização para obtenção a forma da mudança (em vermelho); (e) e (f) resultado do processo de minimização para uma esfera ideal; (g) e (h) resultado do processo de minimização para um cilindro ideal.

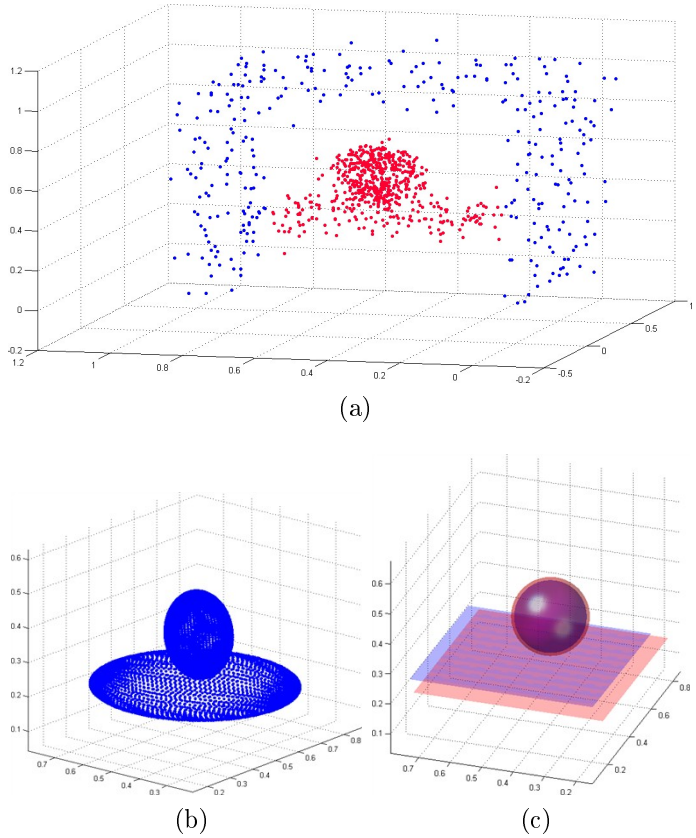
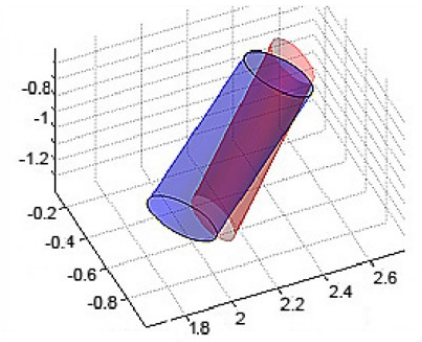


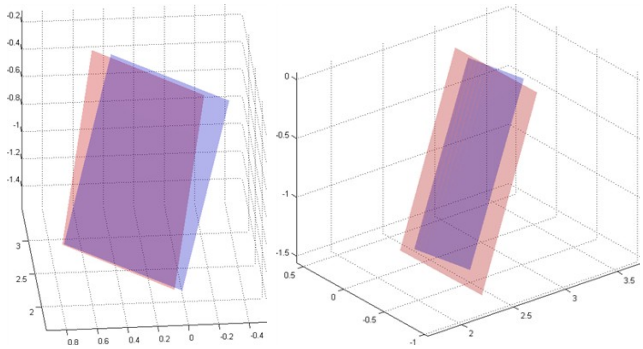
Figura 5.15. Análise comparativa de recuperação de uma mudança composta de duas formas básicas usando dados simulados. Em (a) é mostrada a mudança segmentada em vermelho, em (b) são mostrados os pontos reamostrados para as Gaussianas que representam a mudança e, em (c), o resultado da obtenção de forma pelos dois algoritmos propostos para recuperação de formas básicas.

estudo comparativo com os dados reais apresentado em uma tabela. A Tabela 5.3 ilustra os resultados dos algoritmos propostos. Devido ao número reduzido de experimentos, os resultados obtidos são um pouco diferentes dos dados obtidos com nuvem de pontos simuladas (cinco amostras para cada um dos 3 conjuntos). O tempo de processamento para o algoritmo baseado em matriz de covariância é maior que para os dados simulados. Em dados reais, devido à maior variação de orientação da forma e a dificuldade de obter uma boa inicialização da orientação e escala, o processo de minimização tem seu custo computacional aumentado. Contudo, se o número de pontos disponíveis for muito maior, o processo teria um aumento para ambos os métodos, devido ao cálculo da GMM, mas, com um aumento muito maior para o método $RANSAC$, que é fortemente dependente do número de pontos. Além disso, a exatidão de ambos os métodos degradou-se para dados reais. A principal razão é que as formas básicas não são perfeitamente iguais às reais, mas são aproximações. Mesmo assim, a factibilidade das abordagens foi comprovada.

Os três conjuntos de dados mostrados nas figuras 5.4, 5.5 e 5.6, foram utilizados para validar o algoritmo de recuperação de formas básicas, mostrados nas figuras 5.16a, 5.16b e 5.16c, respectivamente. Elas mostram os resultados da recuperação da forma das mudanças utilizando o espaço de Gaussianas (em vermelho) e no espaço Euclidiano com algoritmo *RANSAC* (em azul). Na Figura 5.16a pode-se notar que a "pessoa" foi modelada por um cilindro, enquanto as portas, nas figuras 5.16b e 5.16c, foram modeladas corretamente por planos. Embora com pequenos erros de orientação, os resultados mostraram a capacidade do método de determinar a forma em dados reais. Vale notar que os dados utilizados já apresentam um erro de orientação, como mostra a Figura 5.4.



(a)



(b) (c)

Figura 5.16. Análise comparativa de recuperação de formas básicas em dados reais. As figuras mostram a recuperação pelo método do espaço de Gaussianas (em vermelho) e pelo algoritmo *RANSAC* (em azul), para as mudanças das Figuras 5.4, 5.5 e 5.6, respectivamente.

5.2.2 Superquádricas

Tendo em vista a restrições dos algoritmos de recuperação de formas básicas, foi proposto o uso de superquádricas. Essas buscam superar a dependência da segmentação de

Tabela 5.3. Análise comparativa entre algoritmos de recuperação de formas básicas em dados reais.

Algoritmo	Espaço Euclidiano	Espaço das Gaussianas
Tempo de Execução (s)	0,3231	0,3156
$Verdadeiros$	0,8667	0,9000
$Falsos\_Verdadeiros$	0,0667	0,0667
$\sigma_{\Delta E(E_x,E_y,E_z)}$ [cm]	(16,21; 13,24; 10,55)	(12,12; 6,22; 8,19)
$\sigma_{\Delta R(\psi,\vartheta,\varphi)}$ [graus]	(4,21; 3,98: 5,12)	(3,12, 2,01, 3,02)

mudanças pelo algoritmo *EM*, que garante apenas mínimos locais, bem como a pouca expressividade das formas básicas. Além disso, um refinamento em escalas é uma alternativa eficaz de evitar a degradação do desempenho do algoritmo de recuperação de forma. Os resultados mostraram que o tempo para cálculo da superquádrica não representou um gargalo para o sistema, sendo próxima de 5% do tempo para detecção e segmentação (esse tempo varia com a quantidade de mudança nos mapas). Isto se deve, principalmente, à abordagem proposta de utilizar as informações já computadas na detecção de mudança, como a relação topológica entre as partes, a inicialização do método e a técnica de programação dinâmica utilizada na fase *split-and-merge*.

Um exemplo de resultado simulado é ilustrado na Figura 5.17, que utiliza recuperação de forma de superquádricas, da mudança apresentada na Figura 5.1f. A Figura 5.17a mostra o ajuste inicial da superquádrica obtido sobre os dados das mudanças (simplificados). Podendo-se notar que, devido ao ruído, a esfera não foi determinada corretamente, apesar da segmentação produzida pela *GMM* ter sido satisfatória. A partir das duas superquádricas determinadas no ajuste inicial, foi utilizado o método *split*. É obtida apenas uma divisão, no plano abaixo da esfera, devido ao bom ajuste já efetuado pelo passo anterior, como mostra a Figura 5.17b. A partir disso é utilizado o método *merge*, mostrado na Figura 5.17c, que obtém um resultado semelhante ao passo inicial (Figura 5.17a). Após, é efetuado o processo de refinamento, utilizando apenas uma escala, ou seja, todos os pontos originalmente presentes na nuvem de pontos segmentada são utilizados para refinar o modelo. A Figura 5.17d apresenta o resultado do refinamento, no qual o modelo de superquádrica chega mais próximo do ideal (em relação a distorção do modelo), embora a esfera não tenha sido completamente recuperada, devido ao ruído inserido nos dados.

Resultados também foram obtidos utilizando dados reais. Um primeiro experimento foi realizado utilizando os dados da segmentação de mudança, mostrados na Figura 5.4d, onde uma forma humana é a mudança. Utilizando uma forma básica de

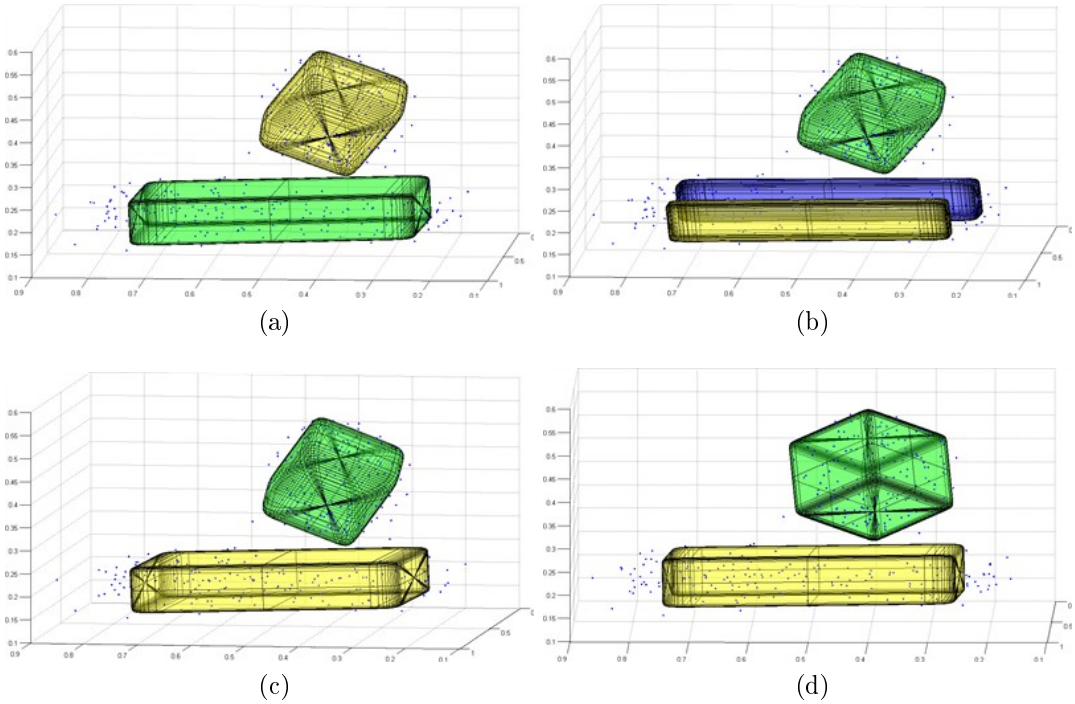


Figura 5.17. Recuperação de formas utilizando superquádricas em dados simulados. As figuras mostram: (a) resultado do ajuste inicial das superquádricas usando dados simplificados; (b) resultado obtido após método *split*; (c) resultado obtido após o método *merge*; (d) resultado final obtido após o refinamento com uso de uma única escala.

um cilindro, mostrada na Figura 5.16a. Porém, essa é pouco significativa para uma pessoa, visto que ela poderia ter sido dividida em mais partes como: cabeça, tronco e membros. Na abordagem proposta foi aplicado o método *split-and-merge*, que visa buscar uma recuperação de forma mais adequada. A Figura 5.18 mostra os passos do algoritmo de recuperação de formas usando superquádricas. Na Figura 5.18a é mostrada a forma que foi melhor ajustada por uma única superquádrica, utilizando dados simplificados. Após o processo de *split*, que gera um grande número de possíveis candidatas a superquádrica, mostradas na Figura 5.18b. Por fim, o resultado final da superquádrica é gerado e mostrado nas figuras 5.18c e 5.18d em duas vistas distintas a fim de facilitar a visualização. Pode-se notar que o corpo foi segmentado de uma maneira bem mais detalhada do que com algoritmo de formas básicas. Os pés e pernas foram segmentados com superquádricas distintas, entretanto o corpo e um dos braços foram segmentados como uma única superquádrica, devido à proximidade entre essas partes. Por outro lado, um dos braços foi segmentado separadamente por uma superquádrica, assim como a cabeça. Devido à dificuldade de ajustar uma forma exata para o tronco, alguns pontos ficaram fora do volume da superquádrica, principal-

mente devido às restrições impostas ao algoritmo de ajuste de forma para selecionar a superquádrica de menor volume possível. Outro detalhe foi a segmentação da cabeça, efetuada por uma forma semelhante a um paralelepípedo, devido a baixa qualidade dos dados amostrados. Tal abordagem é explicada em detalhes na Seção 3.4.2.2.

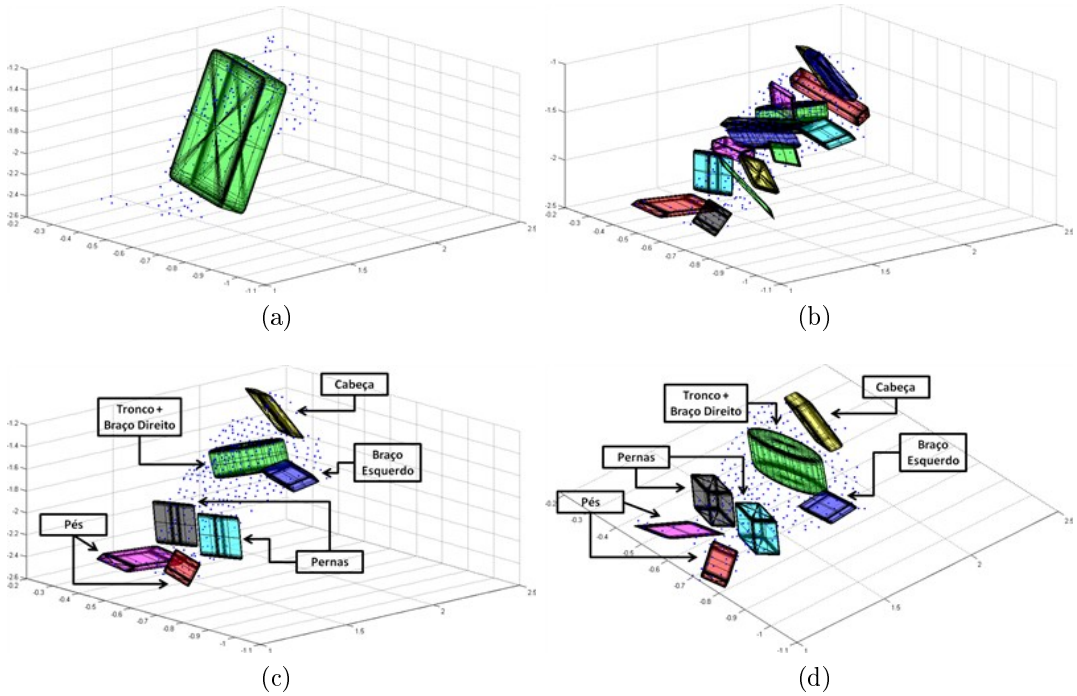
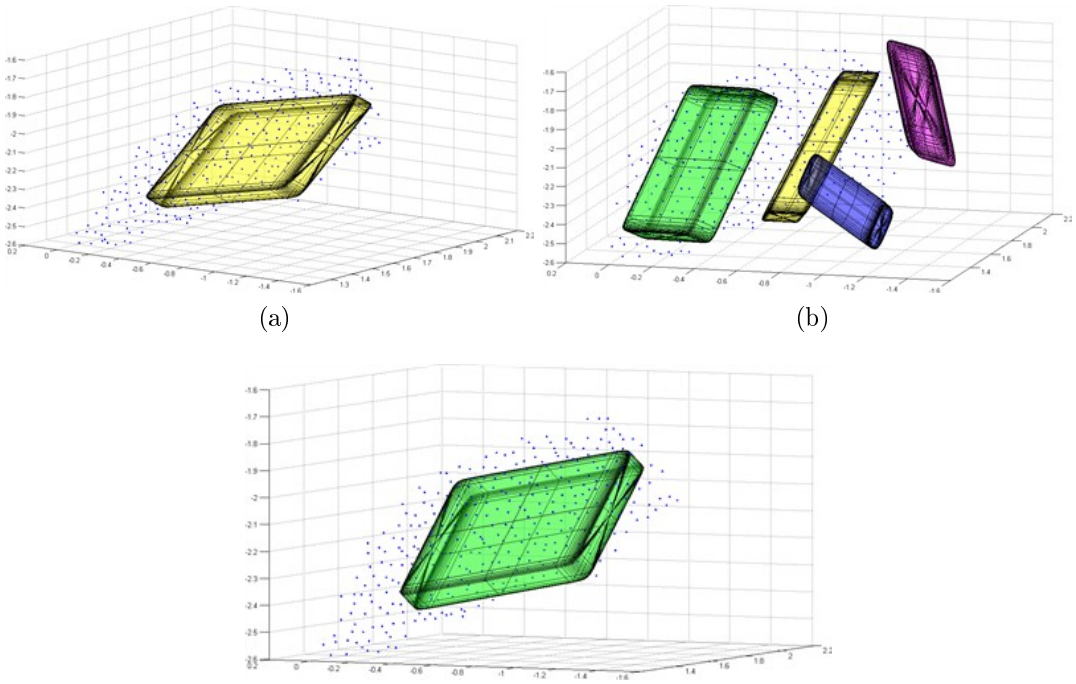


Figura 5.18. Recuperação de formas utilizando superquádricas em dados reais - Conjunto de dados 1. As figuras mostram: (a) resultado do ajuste inicial da uma única superquádrica aos dados simplificados da mudança; (b) resultado obtido após método *split*; (c) e (d) resultado obtido após o método *merge* e ao processo de refinamento com uma única escala, com duas vistas distintas. A unidade de todos os eixos mostrados nas figuras é o metro.

Outro experimento foi feito utilizando a mudança do Conjunto de dados 3, mostrada na Figura 5.6d. Basicamente, é feito um ajuste inicial de uma superquádrica para a nuvem de pontos de forma a representar a porta. Posteriormente, foram aplicados os algoritmos de *split* e *merge*, esse último juntamente com o refinamento. Os resultados são mostrados nas figuras 5.19b e 5.19c, respectivamente. Pode-se notar que o resultado final foi muito semelhante ao inicial, devido à boa segmentação efetuada pelo algoritmo de detecção de mudanças. As tentativas de divisão pelo *split* foram unidas, posteriormente, pelo algoritmo *merge*, mostrando a robustez do método. A etapa de refinamento piorou a qualidade do resultado, considerando a métrica utilizada, ou seja, a distorção obtida pelo processo de minimização de superquádricas. Assim, o resultado final gerado foi exatamente igual ao ajuste inicial, pois o *split-and-merge* se anularam

e o refinamento não melhorou a qualidade. Nesse resultado pode-se notar que alguns pontos não foram completamente representados pela superquádrica. Isso se deve em parte ao ruído, bem como à tentativa de fazer um ajuste de superquádrica de um plano (porta) ser limitada a um valor mínimo de escala, ou seja, é impossível determinar um plano perfeito utilizando a presente abordagem para determinação superquádricas, pois não é permitido utilizar escala zero para quaisquer das coordenadas. O valor mínimo utilizando neste trabalho foi de $0,1$ cm.



(a) (b)

(c)

Figura 5.19. Recuperação de formas utilizando superquádricas em dados reais - Conjunto de dados 3. As figuras mostram: (a) resultado do ajuste inicial da uma única superquádrica aos dados simplificados da mudança (porta); (b) resultado obtido após método *split*; (c) resultado obtido após o método *merge* e o processo de refinamento com uma única escala. A unidade de todos os eixos mostrados nas figuras é o metro.

Foi recuperada também a forma com modelo de superquádrica para a mudança do Conjunto de dados 4, mostrada na Figura 5.7d. Basicamente, é feito um ajuste inicial de uma superquádrica para a nuvem de pontos de forma a representar o cilindro que representa a mudança. Posteriormente, foram aplicados os algoritmos de *split* e *merge*, esse último juntamente com o refinamento. Como o resultado dos passos foram semelhantes, assim como no exemplo anterior, é mostrado apenas o resultado final, na Figura 5.20. A representação em superquádrica apresenta as dimensões de aproximadamente 20cm de raio por 30cm de altura, considerando que o cilindro real tem

dimensão de 20cm de raio por 40cm de altura. Desse modo, a abordagem apresentou um bom resultado, visto que parte do cilindro não foi escaneado por completo, mesmo com a inserção de uma baixa abaixo do cilindro (mostrada na Figura 5.7a). Os pontos da figura, em azul, ilustram os pontos segmentados para a mudança, que mostra que apenas parte do cilindro foi escaneado.

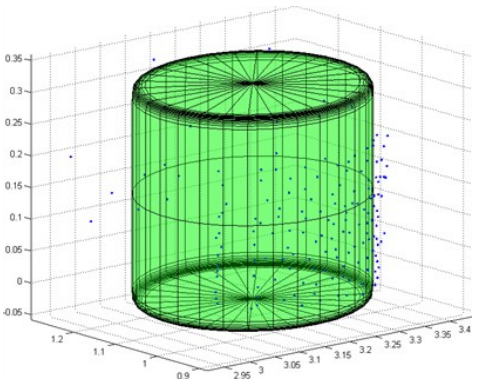


Figura 5.20. Recuperação de formas utilizando superquádricas em dados reais - Conjunto de dados 4. A figura mostra o resultado do ajuste de uma única superquádrica aos dados da mudança após refinamento (cilindro). A unidade de todos os eixos mostrado na figura é o metro.

Resultados foram obtidos utilizando o Conjunto de dados 5, no qual a mudança é uma caixa na esquerda de um corredor. A Figura 5.9d mostra essa mudança em destaque. A Figura 5.21 mostra os passos do algoritmo de recuperação de formas usando superquádricas. Na Figura 5.21a é mostrada a forma que foi melhor ajustada pelos dados segmentados pela GMM (apenas uma GMM), utilizando dados simplificados. Após o processo de *split*, a superquádrica recuperada inicialmente é divida em duas, como mostra a Figura 5.21b. Após o processo de *merge*, obtém-se a superquádrica idêntica a inicial (Figura 5.21c), ou seja, os processos de *split* e *merge* se anulam para este caso. Por fim, o resultado final da superquádrica é gerado, com o uso do processo de refinamento. A Figura 5.21d ilustra o resultado final. A caixa utilizada no experimento apresenta as dimensões de 59cm de altura por 49 cm de largura e 42,5 cm de profundidade, considerando que a superquádrica após o refinamento apresenta dimensões de aproximadamente 47cm de altura por 48 cm de largura e 40 cm de profundidade, os resultados foram bem próximos do real, visto a pouca amostragem da caixa, como mostra os pontos azuis da Figura 5.21d. A diferença na altura do objeto real e do recuperado se deve ao fato, como no exemplo anterior, de não se ter escaneado a completude do objeto devido ao *laser scanner* não estar na mesma altura do objeto.

O último experimento experimental de recuperação de superquádricas foi realizado utilizando os dados da segmentação de mudanças do Conjunto de dados 6, mos-

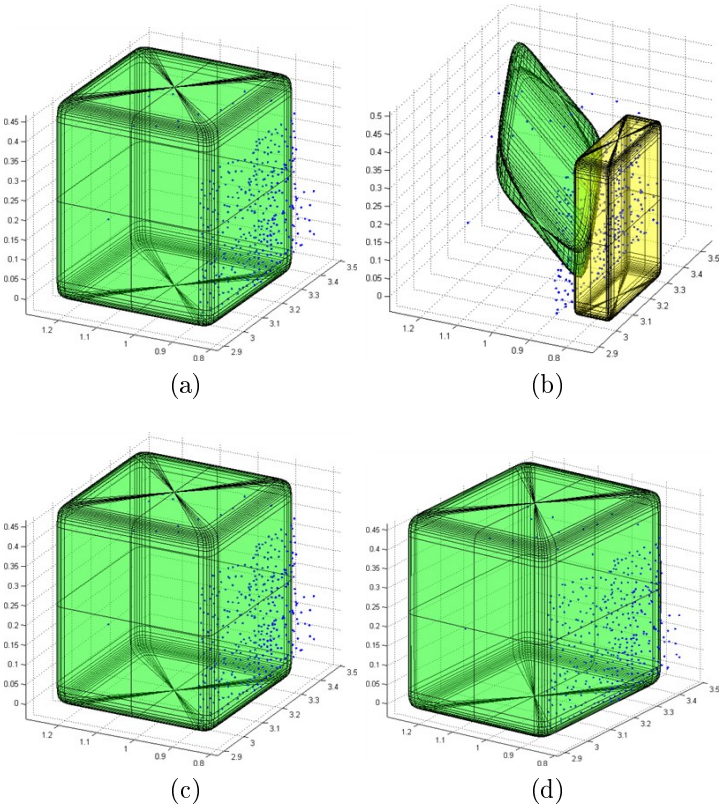


Figura 5.21. Recuperação de formas utilizando superquádricas em dados reais - Conjunto de dados 5. As figuras mostram: (a) resultado do ajuste inicial de uma superquádrica usando dados simplificados; (b) resultado obtido após método *split*; (c) resultado obtido após o método *merge*; (d) resultado final obtido após o refinamento. A unidade de todos os eixos mostrados nas figuras é o metro.

trados na Figura 5.11d, onde uma forma humana é a mudança. A Figura 5.22 mostra os passos do algoritmo de recuperação de formas usando superquádricas. Na Figura 5.22a é mostrada a forma que foi melhor ajustada por uma única superquádrica, utilizando dados simplificados. Após o processo de *split*, que gera um número maior de possíveis candidatas a superquádrica, mostradas na Figura 5.18b. O método *merge* não uni nenhuma das superquádricas geradas pelo *split* como mostra a Figura 5.22c em uma vista distinta. Por fim, o resultado após o refinamento das superquádricas são gerados e mostrados na Figura 5.22d. Pode-se notar que o corpo foi segmentado de uma maneira detalhada, onde as pernas foram segmentados com superquádricas distintas, entretanto o tronco e os braços foram segmentados como duas superquádricas (uma representando o tórax e outra o abdômen e os braços), devido à proximidade entre essas partes. Outro detalhe foi a representação da cabeça, que mesmo após o refinamento obteve uma forma semelhante a um paralelepípedo, devido a baixa qualidade e quantidade dos dados amostrados.

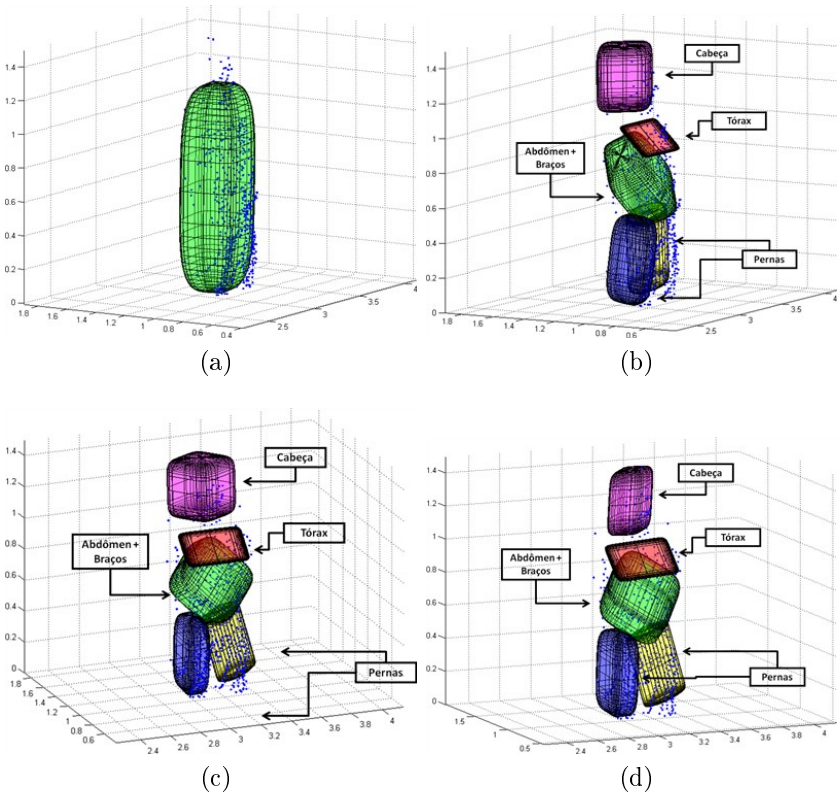


Figura 5.22. Recuperação de formas utilizando superquádricas em dados reais - Conjunto de dados 6. As figuras mostram: (a) resultado do ajuste inicial de uma superquádrica usando dados simplificados; (b) resultado obtido após método *split*; (c) resultado obtido após o método *merge*; (d) resultado final obtido após o refinamento. A unidade de todos os eixos mostrados nas figuras é o metro.

Capítulo 6

Conclusões

São apresentadas, neste capítulo, as conclusões sobre a adequação da metodologia proposta na abordagem do problema de detecção de mudanças e recuperação de forma em mapas 3D baseados em nuvens de pontos. As principais características dos métodos tratados na metodologia são revistas com suas vantagens e desvantagens. As direções futuras também são discutidas como forma de continuação deste trabalho. Por fim, as publicações obtidas fruto deste trabalho são apresentada.

6.1 Discussão dos Resultados

Neste trabalho foi apresentado um estudo sobre detecção de mudanças e recuperação de formas em nuvens de pontos tridimensionais.

O trabalho inicia com uma consideração de que estão disponíveis os dois mapas tridimensionais, ou nuvem de pontos, em tempos distintos e do mesmo local em um mesmo sistema de coordenadas, ou com transformação de sistema de coordenadas conhecida. Com base nesse mapa é realizada a simplificação de pontos, de forma a ultrapassar um dos problemas encontrados inicialmente na metodologia. Esse problema se deve ao fato do custo do cálculo da *GMM* ser diretamente proporcional ao número de pontos de entrada, ou seja, se os mapas forem grandes, o tempo para computar a *GMM* será grande. Assim, a simplificação visa reduzir o número de pontos, buscando preservar a informação geométrica da nuvem de pontos para os processos de detecção de mudanças e de recuperação de forma. Deste modo, o método utilizado mostrou-se muito eficiente, seja pela simplicidade, seja pela representação em escalas ou pelo baixo custo computacional.

Os resultados mostraram que a simplificação não degradou significativamente o desempenho da detecção de mudança, sendo que, em alguns casos, até melhorou

(Tabela 5.1). Além disso, essa simplificação possibilitou a redução de aproximadamente dez vezes no tempo de execução para dados simulados e de mais de quatro vezes para dados reais. Isso mostrou a relevância da inclusão dessa etapa para o processo de detecção de mudança em nuvem de pontos para aceleração do processo.

O algoritmo de detecção e segmentação de mudanças mostrou-se promissor, mesmo na presença de ruído, mostrando-se eficiente mesmo em ambientes de difícil determinação, como o exemplo da porta fechada do terceiro conjunto de dados reais (mostrado na Figura 5.6 na Seção 5.1).

A *EMD* permite, com baixo custo computacional, determinar a distância entre *GMMs*, embora a segmentação de mudança por esse método possa ser muito custosa utilizando uma abordagem exaustiva. A abordagem gulosa proposta, por não possuir um custo computacional expressivo devido ao pequeno número de Gaussianas na *GMM*, possibilitou a determinação e segmentação de mudanças nos mais diferentes contextos, de uma forma eficiente. Alguns problemas na detecção de mudança ainda existem, como regiões conhecidas em apenas um dos mapas, que não necessariamente representam mudanças, e sim, partes do mapa não adquiridas, mostrados nas figuras 5.10 e 5.12. Outros problemas, como "sombra" ou oclusão no mapa causados por uma mudança também interferem, fazendo com que a mudança pareça maior do que realmente é, como mostrado na Figura 5.12.

Um aspecto importante na detecção de mudança proposta neste trabalho é a inexistência de limiares restritivos, como mostra o Algoritmo 3 da Seção 3.3.3. Esses limiares seriam de difícil determinação, sendo necessário estipulá-los manualmente. Além disso, cada limiar seria específico para uma aplicação ou situação, fazendo com que um determinado limiar seja adequado para um dado conjunto de dados e não para outro.

Os resultados de recuperação de forma básicas mostraram a eficiência e a eficácia do método no espaço de Gaussianas, em relação ao método genérico de recuperação no espaço Euclidiano utilizando *RANSAC*. Esse método busca utilizar a informação já disponível da *GMM* para recuperar a forma. Porém, a expressividade das formas básicas é muito restrita. Foi proposto então, o uso de superquádricas, potencialmente mais expressivo, pois com apenas dois parâmetros, obtém-se uma quantidade grande de formas, ilustradas na Figura 3.16 da Seção 3.4.2.1.

Outro problema encontrado, a princípio, foi a segmentação realizada pela *GMM* por meio do algoritmo *EM*, que frequentemente resulta em mínimos locais. Assim, a abordagem *split-and-merge* aplicada a superquádricas se mostrou muito eficaz na busca de um mínimo global, ou mais próximo disto quando possível, gerando, assim, um conjunto de formas que se adequa à mudança de uma maneira bem mais realista,

como mostram as figuras 5.18 e 5.22.

Por fim, devido à simplificação da nuvem de pontos ter se mostrado eficiente, além de preservar a qualidade geométrica da nuvem de pontos, o uso de uma nuvem de pontos simplificados para determinação da forma se mostrou adequado, embora alguns testes tenham sido feitos, nos quais se observarou uma redução da distorção com o uso de uma escala mais fina. Apesar do sistema ter sido desenvolvido para lidar com múltiplas escalas, foram utilizadas apenas duas escalas, sem perda de eficácia.

6.2 Direções Futuras

A linha de pesquisa de detecção de mudanças e recuperação de forma abre um grande número de possibilidades para direções futuras. Uma primeira a ser seguida é efetuar mais testes utilizando a infra-estrutura disponível no Laboratório de Visão Computacional e Robótica (VeRLab) da Universidade Federal de Minas Gerais (UFMG), com uso dos robôs Pioneer, em ambiente *outdoor*. Esses testes poderiam evidenciar outras questões não cobertas pela presente abordagem.

Embora o uso do algoritmo *EM* para estimar a *GMM* seja adequado quando se conhece o mapa completo, pode-se pensar em uma abordagem para obtenção de *GMMs* à medida que os dados são adquiridos pelo *laser scanner*. Uma possibilidade é o uso da abordagem de Kristan et al. [2008] que utiliza *Iterative GMM*. Outra direção possível é utilizar o método *split-and-merge* durante a estimação da *GMM*, ultrapassando o problema de mínimos locais do algoritmo *EM*. O método de Xiang & Wang [2004] implementa tal proposta, utilizando *EM* e *split-and-merge* conjuntamente.

O uso do *split-and-merge* foi feito apenas na estimação do modelo de superquádricas. Porém, testes utilizando um pré-processamento com *split-and-merge* em formas básicas, poderiam gerar resultados melhores, ultrapassando as limitações da estimação usando o algoritmo *EM*.

Outra direção possível, visando um melhor alinhamento do mapa, é utilizar o algoritmo *ICP* [Besl & McKay, 1992] antes do processo de simplificação, ou após, de forma a remover pontos que não tenham sido lidos em mapas distintos, como na abordagem proposta por Girardeau-Montaut et al. [2005]. A fim de evitar mudanças detectadas por problemas de oclusão, pode-se utilizar o método de projeção do raio *laser* a fim de evitar detecção indevida de mudança Girardeau-Montaut et al. [2005].

Uma outra possibilidade é o estudo de outras técnicas de simplificação de nuvem de pontos de maior qualidade, porém com custo de processamento bem maior. Contudo, como o uso de processamento paralelo tem sido cada vez maior, pode-se pen-

sar em outro método de simplificação que visa preservar as características geométricas de maneira mais otimizada, e sem degradar o desempenho em termos de tempo de execução.

Outras linhas de pesquisa futuras que já começaram a ser investigadas pelo autor, como o uso de mapas com textura, obtidos a partir de informação de sensores *lasers* e câmeras de vídeo, como proposto em Núñez et al. [2009]. Essa informação adicional pode facilitar o processo de determinação de mudança, além de texturizar a forma recuperada e ajudar em um posterior processo de classificação.

A utilização de mapas 3D obtidos a partir de sensores táteis (hápticas) [Faria et al., 2009] é outra direção interessante, que também já começou a ser investigada pelo autor. Nesse trabalho foi utilizado o método proposto nesta metodologia para recuperação de formas básicas no espaço de Gaussianas, bem como classificadores Bayesianos para determinar mais rapidamente a forma.

Outra direção é o uso de abordagens probabilísticas que permitiriam obter resultados mais robustos. Um exemplo poderia ser o uso de teoria da informação e entropia, como no trabalho de Rocha [2005], para melhorar o ajuste da forma, bem como para auxiliar no processo de detecção de mudança por meio do conhecimento de níveis de confiança de cada ponto realmente existir no mapa.

A classificação de formas pode ser outra linha de pesquisa interessante que permitiria utilizar os resultados deste trabalho em um produto comercial. Uma direção poderia ser a utilização de mapas CAD como uma das entradas da metodologia, bem como utilizar a forma obtida da mudança para atualizar tais mapas CAD.

A classificação de formas, em formas estáticas ou dinâmicas, poderia permitir, a atualização do mapa, caso a mudança fosse estática. Além disso, pode-se considerar conhecido um banco de dados de formas e fazer a busca apenas por formas já conhecidas, facilitando o processo de detecção de mudança e o de recuperação de forma, como no trabalho de [Biegelbauer & Vincze, 2007]. O uso de banco de dados de formas 3D poderia também ser uma outra linha promissora [Lai & Fox, 2009].

Por fim, outra linha de pesquisa que poderia ser utilizada de modo a reconhecer forma com maior precisão é o uso de superquádricas deformáveis, que permitem, com algum custo computacional extra, a determinação de formas bem mais próximas das reais [Jaklič et al., 2000].

6.3 Publicações Relevantes

Até o momento da finalização deste trabalho, havia sido geradas três publicações em conferência fruto deste trabalho e uma publicação para um periódico estava em elaboração.

P. Drews Jr, P. Núñez, R. Rocha, M. Campos and J. Dias. "Novelty Detection and 3D Shape Retrieval using Superquadrics and Multi-Scale Sampling for Autonomous Mobile Robots". In Proc. of 2010 IEEE Int. Conf. on Robotics and Automation (ICRA'2010), Anchorage, Alaska, USA, May 3-8, 2010.

P. Núñez, P. Drews Jr, R. Rocha, M. Campos and J. Dias. "Novelty Detection and 3D Shape Retrieval based on Gaussian Mixture Models for Autonomous Surveillance Robotics". In Proc. of 2009 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS'2009), St. Louis, MO, USA, Oct. 11-15, 2009

Diego R. Faria, José Prado, Paulo Drews Jr. and Jorge Dias. "Object Shape Retrieval through Grasping Exploration". In Proc.of 4th European Conference on Mobile Robots (ECMR'09) , Mlini/Dubrovnik, Croatia, September 2009.

Referências Bibliográficas

3D Laser Mapping Co. (2009). 3d Scanning - coastal applications. Disponível em http://www.3dlasermapping.com/uk/3d/applications/coastal.htm.

Aach, T.; Kaup, A. & Mester, R. (1993). Statistical model-based change detection in moving video. *Signal Processing*, 31(2):165–180.

Alexa, M.; Behr, J.; Cohen-Or, D.; Fleishman, S.; Levin, D. & Silva, C. T. (2001). Point set surfaces. In *IEEE Visualization (VIS)*, pp. 21–28, San Diego, EUA.

Amenta, N.; Bern, M. & Kamvysselis, M. (1998). A new Voronoi-based surface reconstruction algorithm. In *International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*, pp. 415–421, Orlando, EUA. ACM.

Amorim, I.; Rocha, R. & Dias, J. (2008). Mobile robotic surveillance systems: Detecting and evaluating changes in 3D mapped environments. In *Israeli Conference on Robotics (ICR)*, pp. 1–7, Herzliya, Israel.

Andreasson, H.; Magnusson, M. & Lilienthal, A. J. (2007). Has something changed here? Autonomous difference detection for security patrol robots. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 3429–3435, San Diego, EUA.

Andriluka, M.; Friedmann, M.; Kohlbrecher, S.; Meyer, J.; Petersen, K.; Reinl, C.; Schauss, P.; Schnitzspan, P.; Strobel, A.; Thomas, D. & von Stryk, O. (2009). RoboCupRescue 2009 - Robot League Team: Darmstadt Rescue Robot Team. Technical report, Technische Universität Darmstadt.

Artac, M.; Jogan, M. & Leonardis, A. (2002). Incremental PCA for on-line visual learning and recognition. In *International Conference on Pattern Recognition (ICPR)*, volume 3, pp. 781–784, Quebec, Canada. IEEE Computer Society.

Ballard, D. H. (1987). *Generalizing the Hough transform to detect arbitrary shapes*, pp. 714–725. Morgan Kaufmann Publishers Inc., San Francisco, EUA.

Barr, A. H. (1981). Superquadrics and angle-preserving transformations. *IEEE Computer Graphics and Applications*, 1(1):11–23.

Besl, P. J. & McKay, N. D. (1992). A method for registration of 3D shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):239–256.

Biederman, I. (1985). Human image understanding: Recent research and a theory. *Computer Vision Graphics and Image Processing*, 32(1):29–73.

Biegelbauer, G. & Vincze, M. (2007). Efficient 3D object detection by fitting superquadrics to range image data for robot's object manipulation. In *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1086–1091, Roma, Itália.

Bierbaum, A.; Gubarev, I. & Dillmann, R. (2008). Robust shape recovery for sparse contact location and normal data from haptic exploration. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 3200–3205, Nice, França.

Binford, T. (1971). Visual perception by computer. In *IEEE Conference on Systems and Control*, Miami, EUA.

Blanco, J.-L.; Fernandez-Madrigal, J.-A. & Gonzalez, J. (2008). Toward a unified bayesian approach to hybrid metric-topological SLAM. *IEEE Transactions on Robotics*, 24(2):259–270.

Blinn, J. F. (1982). A generalization of algebraic surface drawing. *ACM Transactions on Graphics*, 1(3):235–256.

Boissonnat, J.-D. & Cazals, F. (2001). Coarse-to-fine surface simplification with geometric guarantees. *Computer Graphics Forum*, 20(3):490–499.

Bouman, C. A. (1997). Cluster: An unsupervised algorithm for modeling Gaussian mixtures. Disponível em http://www.ece.purdue.edu/~bouman.

Brito, J.; Silveira, M.; Jacobsen, K.; Amorim, S.; Mota, G.; Feitosa, R. & Heipke, C. (2008). Monitoring of height changes in urban areas from multi-temporal, multi-scale and multi-platform remotely sensed data. In *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences (IAPRS)*, volume 37 - B1, pp. 835–840, Pequim, China.

Bruni, V.; Vitulano, D. & Maniscalco, U. (2004). Fast segmentation and modeling of range data via steerable pyramid and superquadrics. *Journal of Winter School of Computer Graphics*, 12(1):73–80.

Carmen Development Team (2009). *Carmen Manual*. Carnegie Melon University, Pittsburgh, EUA. Disponível em http://carmen.sourceforge.net/.

Carpenter, G.; Rubin, M. & Streilein, W. (1997). ARTMAP-FD: Familiarity discrimination applied to radar target recognition. In *IEEE International Conference on Neural Networks (ICNN)*, volume 3, pp. 1459–1464, Houston, EUA.

Chappell, G. J. & Taylor, J. G. (1993). The temporal Kohonen map. *Neural Networks*, 6(3):441–445.

Cheng, H. D.; Shi, X. J.; Min, R.; Hu, L. M.; Cai, X. P. & Du, H. N. (2006). Approaches for automated detection and classification of masses in mammograms. *Pattern Recognition*, 39(4):646–668.

Chevalier, L. (2004). *Modélisation et indexation d'objets 3D à l'aide de superellipsöides*. PhD thesis, Universitat Claude Bernard, Lyon, França.

Chevalier, L.; Jaillet, F. & Baskurt, A. (2001). 3D shape coding with superquadrics. In *International Conference on Image Processing (ICIP)*, volume 2, pp. 93–96, Thessaloniki, Grécia.

Chevalier, L.; Jaillet, F. & Baskurt, A. (2003). Segmentation and superquadric modeling of 3D objects. *Journal of Winter School of Computer Graphics*, 11(1):9–16.

Cignoni, P.; Montani, C. & Scopigno, R. (1997). A comparison of mesh simplification algorithms. *Computers & Graphics*, 22(1):37–54.

Cormen, T. H.; Leiserson, C. E.; Rivest, R. L. & Stein, C. (2001). *Introduction to Algorithms*. The MIT Press, Cambridge, EUA, segunda edição. ISBN 0-2620-3293-7.

Crook, P. A.; Marsland, S.; Hayes, G. & Nehmzow, U. (2002). A tale of two filters - online novelty detection. In *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3894–3900, Washington, EUA.

Dempster, A. P.; Laird, N. M. & Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society*, 39(1):1–38.

Dey, T. K.; Giesen, J. & Hudson, J. (2001). Decimating samples for mesh simplification. In *Canadian Conference on Computational Geometry (CCCG)*, pp. 85–88, Waterloo, Canada.

Directed Perception (2009). *Computer Controlled Pan-Tilt Unit Model PTU-D46 User Manual*. Burlingame, EUA. Disponível em http://www.dperception.com/pdf/PTU-manual-d46.pdf.

Faria, D.; Prado, J.; Drews Jr., P. & Dias, J. (2009). Object shape retrieval through grasp exploration. In *European Conference on Mobile Robots (ECMR)*, pp. 43–48, Zagreb, Croácia.

Figueiredo, M. A. T. & Jain, A. K. (2000). Unsupervised learning of finite mixture models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(3):381–396.

Figueiredo, M. A. T. & Jain, A. K. (2002). Unsupervised learning of finite mixture models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(3):381--396.

Fischler, M. A. & Bolles, R. C. (1981). Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395.

Fitzgibbon, A. W.; Eggert, D. W. & Fisher, R. B. (1997). High-level CAD model acquisition from range images. *Computer-Aided Design*, 29(4):321–330.

Forstner, W. & Moonen, B. (1999). A metric for covariance matrices. Technical report, Departament of Geodesy and Geoinformatics - Stuttgart University, Stuttgart, Alemanha.

Freeman, L. (1979). Centrality in social networks conceptual clarification. *Social Networks*, 1(3):215–239.

Girardeau-Montaut, D.; Roux, M.; Marc, R. & Thibault, G. (2005). Change detection on point cloud data acquired with a ground laser scanner. In Vosselman, G. & Brenner, C., editores, *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences (IAPRS)*, volume 36 - 3/W19, pp. 1–6, Enschede, Holanda.

Gobbetti, E. & Marton, F. (2004). Layered point clouds. *Computers & Graphics*, 28(6):815 – 826.

Gopi, M.; Krishnan, S. & Silva, C. T. (2000). Surface reconstruction based on lower dimensional localized Delaunay triangulation. *Computer Graphics Forum*, 19(3):467–478.

Gottschalk, S.; Lin, M. C. & Manocha, D. (1996). OBBTree: a hierarchical structure for rapid interference detection. In *International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*, pp. 171–180, New Orleans, EUA. ACM.

Guh, R. S.; Zorriassatine, F.; Tannock, J. D. T. & O'Brien, C. (1999). On-line control chart pattern detection and discrimination - a neural network approach. *Artificial Intelligence in Engineering*, 13(4):413 – 425.

Hanson, A. J. (1988). Hyperquadrics: smoothly deformable shapes with convex polyhedral bounds. *Computer Vision, Graphics, and Image Processing*, 44(2):191–210.

Heckbert, P. S. & Garland, M. (1997). Survey of polygonal surface simplification algorithms. Technical report, Carnegie Mellon University, Pittsburgh, EUA.

Heller, A. J.; Leclerc, Y. G. & Luong, Q.-T. (2001). A framework for robust 3D change detection. In *The International Society for Optical Engineering (SPIE)*, volume 4540, pp. 639–649, San Jose, EUA.

Hokuyo Automatic Co. (2009). Hokuyo URG-04LX. Disponível em http://www.hokuyo-aut.jp/02sensor/07scanner/urg_04lx.html.

Hopfield, J. J. (1982). Neural networks and physical systems with emergent collective computational abilities. *National Academy of Sciences of the United States of America*, 79(8):2554–2558.

Hoppe, H. (1994). *Surface reconstruction from unorganized points*. PhD thesis, Department of Computer Science and Engineering, University of Washington, Seattle, EUA.

Hoppe, H.; DeRose, T.; Duchamp, T.; McDonald, J. & Stuetzle, W. (1992). Surface reconstruction from unorganized points. In *International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*, pp. 71–78, Chicago, EUA. ACM.

Hotter, M.; Mester, R. & Muller, F. (1996). Detection and description of moving objects by stochastic modelling and analysis of complex scenes. *Signal Processing: Image Communication*, 8(4):281–293.

Hsu, Y. Z.; Nagel, H.-H. & Rekers, G. (1984). New likelihood test methods for change detection in image sequences. *Computer Vision, Graphics and Image Processing*, 26(1):73–106.

Hwang, Y.; Kim, J.-S. & Kweon, I.-S. (2008). Change detection using a statistical model in an optimally selected color space. *Computer Vision and Image Understanding*, 112(3):231–242.

Hwang, Y.-B.; Kim, J.-S. & Kweon, I. S. (2004). Change detection using a statistical model of the noise in color images. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 2713–2718, Sendai, Japão.

Ishikawa, K.; Meguro, J.; Amano, Y.; Hashizume, T.; Takiguchi, J.; Kurosaki, R. & Hatayama, M. (2005). Parking-vehicles recognition using spatial temporal data (a study of mobile robot surveillance system using spatial temporal GIS part 2). In *IEEE International Workshop on Safety, Security and Rescue Robotics (SSRR)*, pp. 151–157, Kobe, Japão.

Itti, L.; Koch, C. & Niebur, E. (1998). A model of saliency-based visual attention for rapid scene analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(11):1254–1259.

Jaklič, A.; Leonardis, A. & Solina, F. (2000). *Segmentation and Recovery of Superquadrics*, volume 20 of *Computational imaging and vision*. Kluwer Academic Publishers, Norwell, EUA. ISBN 0-7923-6601-8.

Jin, S.; Yeung, D. S. & Wang, X. (2007). Network intrusion detection in covariance feature space. *Pattern Recognition*, 40(8):2185–2197.

Jolliffe, I. T. (2002). *Principal Component Analysis*. Springer Series in Statistics. Springer-Verlag Inc., New York, EUA, segunda edição. ISBN 0-3879-5442-2.

Katsoulas, D. & Kosmopoulos, D. I. (2006). Box-like superquadric recovery in range images by fusing region and boundary information. In *International Conference on Pattern Recognition (ICPR)*, volume 1, pp. 719–722, Hong Kong, China. IEEE Computer Society.

Keren, D.; Cooper, D. & Subrahmonia, J. (1994). Describing complicated objects by implicit polynomials. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(1):38–53.

Khoshelham, K. (2007). Extending generalized Hough transform to detect 3D objects in laser range data. In *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences (IAPRS)*, volume 36 - 3/W52, pp. 206–210, Espoo, Finlândia.

King, S.; King, D.; Astley, K.; Tarassenko, L.; Hayton, P. & Utete, S. (2002). The use of novelty detection techniques for monitoring high-integrity plant. In *International Conference on Control Applications.*, volume 1, pp. 221–226, Glasgow, Escócia.

Kobbelt, L. & Botsch, M. (2004). A survey of point-based techniques in computer graphics. *Computers & Graphics*, 28(6):801–814.

Kohonen, T. (1982). Self-organized formation of topologically correct feature maps. *Biological Cybernetics*, 43(1):59–69.

Kohonen, T. (2001). *Self-Organizing Maps*. Springer Series in Information Sciences. Springer-Verlag Inc., New York, EUA, terceira edição. ISBN 3-540-67921-9.

Kristan, M.; Skocaj, D. & Leonardis, A. (2008). Incremental learning with Gaussian mixture models. In *Computer Vision Winter Workshop (CVWW)*, pp. 25–32, Moravske Toplice, Eslovênia.

Krivic, J. & Solina, F. (2004). Part-level object recognition using superquadrics. *Computer Vision and Image Understanding*, 95(1):105–126.

Lai, K. & Fox, D. (2009). 3D laser scan classification using web data and domain adaptation. In *Robotics: Science and Systems (RSS)*, pp. 1–8, Seattle, EUA.

Lee, P.-F. & Jong, B.-S. (2008). Point-based simplification algorithm. *WSEAS Transactions on Computer Research*, 3(1):61–66.

Leonardis, A.; Jaklič, A. & Solina, F. (1997). Superquadrics for segmenting and modeling range data. *IEEE Transactions on Pattern Analisys and Machine Intelligence*, 19(11):1289–1295.

Li, L. & Leung, M. (2002). Integrating intensity and texture differences for robust change detection. *IEEE Transactions on Image Processing*, 11(2):105–112.

Lindeberg, T. (1998). Feature detection with automatic scale selection. *International Journal of Computer Vision*, 30(2):79–116.

Linsen, L. (2001). Point cloud representation. Technical report, Universität Karlsruhe, Karlsruhe, Alemanha.

Liu, Y. & Yuen, M. (2003). Optimized triangle mesh reconstruction from unstructured points. *The Visual Computer*, 19(1):23–37.

Lowe, D. G. (1999). Object recognition from local scale-invariant features. In *International Conference on Computer Vision (ICCV)*, volume 2, pp. 1150–1157, Kerkyra, Grécia. IEEE Computer Society.

Luebke, D. P. (2001). A developer's survey of polygonal simplification algorithms. *IEEE Computer Graphics and Application*, 21(3):24–35.

Macqueen, J. B. (1967). Some methods of classification and analysis of multivariate observations. In Cam, L. M. L. & Neyman, J., editores, *Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, pp. 281–297, Berkeley, EUA. University of California Press.

Markou, M. & Singh, S. (2003a). Novelty detection: A review - part 1: Statistical approaches. *Signal Processing*, 83(12):2003.

Markou, M. & Singh, S. (2003b). Novelty detection: A review - part 2: Neural network based approaches. *Signal Processing*, 83(12):2499–2521.

Marquardt, D. W. (1963). An algorithm for least-squares estimation of nonlinear parameters. *SIAM Journal on Applied Mathematics*, 11(2):431–441.

Marsland, S. (2003). Novelty detection in learning systems. *Neural Computing Surveys*, 3(1):157–195.

Marsland, S.; Nehmzow, U. & Shapiro, J. (1999). A model of habituation applied to mobile robots. In *Towards Intelligent Mobile Robots Conference (TIMR)*, pp. 1–10, Bristol, UK.

Marsland, S.; Nehmzow, U. & Shapiro, J. (2000a). Detecting novel features of an environment using habituation. In *International Conference on Simulation of Adaptive Behaviour (SAB)*, pp. 1–10, Paris, França.

Marsland, S.; Nehmzow, U. & Shapiro, J. (2000b). Novelty detection for robot neotaxis. In *ICSC/IFAC International Symposium on Neural Computation*, pp. 554 – 559, Berlin, Alemanha.

Marsland, S.; Nehmzow, U. & Shapiro, J. (2000c). A real-time novelty detector for a mobile robot. In *EUREL European Advanced Robotics Systems Masterclass and Conference*, pp. 1–8, Salford, UK.

Marsland, S.; Nehmzow, U. & Shapiro, J. (2001). Novelty detection in large environments. In *Towards Intelligent Mobile Robots Conference (TIMR)*, pp. 1–10, Manchester, UK.

Marsland, S.; Nehmzow, U. & Shapiro, J. (2002). Environment-specific novelty detection. In *International Conference on Simulation of Adaptive Behaviour (SAB)*, pp. 36–45, Edinburgh, UK.

Marsland, S.; Nehmzow, U. & Shapiro, J. (2005). On-line novelty detection for autonomous mobile robots. *Robotics and Autonomous Systems*, 51(2-3):191–206.

Melex A&D Tyszkiewicz Sp. (2009). Melex Model 745. Disponível em http://www.melex.com.pl/pliki/745.pdf.

MobileRobots Inc. (2009). P3 - AT - the high performance all-terrain robot. Disponível em http://www.activrobots.com/ROBOTS/p2at.html.

Moenning, C. & Dodgson, N. A. (2003). A new point cloud simplification algorithm. In *The IASTED International Conference on Visualization, Imaging and Image Processing (VIIP)*, pp. 1027–1033, Benalmádena, Espanha.

Moenning, C. & Dodgson, N. A. (2004). Intrinsic point cloud simplification. In *International Conference on Computer Graphics & Vision (GraphiCon)*, pp. 1–8, Moscow, Russia.

Nardi, D.; Matteis, G. D.; Capobianco, C. & Coletti, A. (2007). A quad-rotor for information gathering in rescue scenarios. Fourth Symposium for Advanced Technology.

Núñez, P.; Drews Jr, P.; Rocha, R. & Dias, J. (2009). Data fusion calibration for a 3D laser range finder and a camera using inertial data. In *European Conference on Mobile Robots (ECMR)*, pp. 31–36, Zagreb, Croácia.

Paalanen, P.; Kamarainen, J.-K.; Ilonen, J. & Kälviäinen, H. (2006). Feature representation and discrimination based on gaussian mixture model probability densities-practices and algorithms. *Pattern Recognition*, 39(7):1346–1358.

Pauly, M. (2003). *Point Primitives for Interactive Modeling and Processing of 3D Geometry*. PhD thesis, ETH Zurich, Zurich, Suíça.

Pauly, M. & Gross, M. (2001). Spectral processing of point-sampled geometry. In *International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*, pp. 379–386, Los Angeles, EUA. ACM.

Pauly, M.; Gross, M. & Kobbelt, L. P. (2002). Efficient simplification of point-sampled surfaces. In *IEEE Visualization (VIS)*, pp. 163–170, Boston, EUA. IEEE Computer Society.

Pentland, A. P. (1986). Perceptual organization and the representation of natural form. *Artificial Intelligence*, 28(3):293–331.

Pilu, M. & Fisher, R. B. (1999). Training PDMs on models: the case of deformable superellipses. *Pattern Recognition Letters*, 20(5):463–474.

Pirrone, R. & Chella, A. (2003). A neural architecture for segmentation and modelling of range data. *Lecture Notes in Computer Science*, 2829(1):130–141.

Primdahl, K.; Katz, I.; Feinstein, O.; Mok, Y. L.; Dahlkamp, H.; Stavens, D.; Montemerlo, M. & Thrun, S. (2005). Change detection from multiple camera images extended to non-stationary cameras. In *International Conference on Field and Service Robotics (FSR)*, pp. 1–10, Port Douglas, Australia.

Rabbani, T. (2006). *Automatic reconstruction of industrial installations - Using point clouds and images*. PhD thesis, Delft University of Technology, Delft, Holanda.

Radke, R. J.; Andra, S.; Al-kofahi, O. & Roysam, B. (2005). Image change detection algorithms: A systematic survey. *IEEE Transactions on Image Processing*, 14(3):294–307.

Redner, R. A. & Walker, H. F. (1984). Mixture densities, maximum likelihood and the EM algorithm. *SIAM Review*, 26(2):195–239.

Rissanen, J. (1983). A universal prior for integers and estimation by minimum description length. *The Annals of Statistics*, 11(2):416–431.

Rocha, R. (2005). *Building Volumetric Maps with Cooperative Mobile Robots and Useful Information Sharing: a Distributed Control Approach based on Entropy*. PhD thesis, Faculty of Engineering - University of Porto, Porto, Portugal.

Rosin, P. L. (1998). Thresholding for change detection. In *International Conference on Computer Vision (ICCV)*, pp. 274–279, Bombay, India. IEEE Computer Society.

Rote, G. (1991). Computing the minimum hausdorff distance between two point sets on a line under translation. *Information Processing Letters*, 38:123–127.

Rubner, Y.; Tomasi, C. & Guibas, L. J. (1998). A metric for distributions with applications to image databases. In *International Conference on Computer Vision (ICCV)*, pp. 59–66, Bombay, India. IEEE Computer Society.

Rusu, R. B.; Marton, Z. C.; Blodow, N.; Dolha, M. & Beetz, M. (2008). Towards 3D point cloud based object maps for household environments. *Robotics and Autonomous Systems*, 56(11):927–941.

Scaramuzza, D.; Harati, A. & Siegwart, R. (2007). Extrinsic self calibration of a camera and a 3D laser range finder from natural scenes. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 4164–4169, San Diego, EUA.

Schnabel, R.; Wahl, R. & Klein, R. (2007). Efficient RANSAC for point-cloud shape detection. *Computer Graphics Forum*, 26(2):214–226.

Schnabel, R.; Wessel, R.; Wahl, R. & Klein, R. (2008). Shape recognition in 3D point-clouds. In *International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision*, pp. 65–72, Plzen - Bory, República Checa.

Schroeder, W.; Martin, K. & Lorensen, B. (2004). *The Visualization Toolkit*. Kitware Inc., Clifton Park, EUA, quarta edição. ISBN 1-9309-3419-X.

Segway Inc. (2009). Segway Robotic Mobile Plataform (RMP) Specification. Disponível em http://www.segway.com/downloads/pdfs/brochures/RMP_SpecSheet_09LQ.pdf.

Sick AG. (2009). Laser Measurement Technology LMS2xx / LMS200 / Indoor / Short-Range. Disponível em http://www.hizook.com/files/publications/SICK_LMS200-291_Tech_Info.pdf.

Simoncelli, E. P.; Freeman, W. T.; Adelson, E. H. & Heeger, D. J. (1992). Shiftable multiscale transforms. *IEEE Transactions on Information Theory*, 38(2):587–607.

Sithole, G. & Vosselman, G. (2003). Automatic structure detection in a point-cloud of an urban landscape. In *IEEE/ISPRS Joint Workshop on Remote Sensing and Data Fusion over Urban Areas (URBAN)*, pp. 67–71, Berlin, Alemanha.

Skifstad, K. & Jain, R. (1989). Illumination independent change detection for real world image sequences. *Computer Vision, Graphics, and Image Processing*, 46(3):387–399.

Solina, F. & Bajcsy, R. (1990). Recovery of parametric models from range images: The case for superquadrics with global deformations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(2):131–147.

Stanley, J. C. (1976). Computer simulation of a model of habituation. *Nature*, 261:146–148.

Steinbruch, A. & Winterle, P. (1987). *Geometria Analítica*. McGraw-Hill, São Paulo, Brasil, segunda edição. ISBN 0-0745-0409-6.

Steinle, E.; Oliveira, F.; Bohr, H.-P. & Loch, C. (1999). Assessment of laser scanning technology for change detection in buildings. In *The CIPA International Symposium*, volume 17, pp. 1–7, Olinda, Brasil.

Tangelder, J. W. & Veltkamp, R. C. (2004). A survey of content based 3D shape retrieval methods. In *Shape Modeling International (SMI)*, pp. 145–156, Genova, Itália. IEEE Computer Society.

Tanjung, G. & Lu, T.-F. (2007). A study on indoor automatic change detection for a mobile-camera. In *Australasian Conference on Robotics and Automation (ARAA)*, pp. 1–6, Brisbane, Austrália.

Tao, L.; Castellani, U. & Murino, V. (2004). Robust 3D segmentation for underwater acoustic images. In *International Symposium on 3D Data Processing, Visualization, and Transmission (3DPVT)*, pp. 813–819, Thessaloniki, Grécia. IEEE Computer Society.

Tarassenko, L.; Hayton, P.; Cerneaz, N. & Brady, M. (1995). Novelty detection for the identification of masses in mammograms. In *International Conference on Artificial Neural Networks*, pp. 442–447, Cambridge, UK.

Tax, D. & Duin, R. (2000). Data description in subspaces. In *International Conference on Pattern Recognition (ICPR)*, volume 2, pp. 672–675, Barcelona, Espanha.

The Mathworks Inc. (2009). MATLAB - the language of technical computing. Disponível em http://www.mathworks.com/.

Thrun, S.; Burgard, W. & Fox, D. (2005). *Probabilistic Robotics*. The MIT Press, Cambridge, EUA, primeira edição. ISBN 0-2622-0162-3.

Tong, Y. L. (1990). *The Multivariate Normal Distribution*. Springer Series in Statistics. Springer-Verlag Inc., New York, EUA. ISBN 0-3879-7062-2.

Vandorpe, J.; Brussel", H. V. & Xu, H. (1996). Exact dynamic map building for a mobile robot using geometrical primitives produced by a 2d range finder. In *IEEE International Conference on Robotics and Automation*, pp. 901–908, Minneapolis, EUA.

Vieira Neto, H. & Nehmzow, U. (2004). Visual novelty detection for inspection tasks using mobile robots. In *Brazilian Symposium on Neural Networks (SBRN)*, pp. 1–6, São Luís, Brasil.

Vieira Neto, H. & Nehmzow, U. (2005a). Automated exploration and inspection: Comparing two visual novelty detectors. *International Journal of Advanced Robotic Systems*, 2(4):355–362.

Vieira Neto, H. & Nehmzow, U. (2005b). Incremental PCA: An alternative approach for novelty detection. In *Towards Autonomous Robotic Systems (TAROS)*, pp. 227–233, London, UK.

Vieira Neto, H. & Nehmzow, U. (2007). Visual novelty detection with automatic scale selection. *Robotics and Autonomous Systems*, 55(9):693–701.

Vieira Neto, H. & Nehmzow, U. (2008). Visual novelty detection for autonomous inspection robots. In Takahashi, Y., editor, *Service Robot Applications*, pp. 309–330. I-Tech Education and Publishing, Vienna, Austria.

Vogtle, T. & Steinle, E. (2004). Detection and recognition of changes in building geometry derived from multitemporal laser scanning data. In *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences (IAPRS)*, volume 35 - B2, pp. 428–433, Istanbul, Turquia.

Vosselman, G.; Gorte, B. & Sithole, G. (2004a). Change detection for updating medium scale maps using laser altimetry. In *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences (IAPRS)*, volume 34 - B3, pp. 207–212, Istanbul, Turqueia.

Vosselman, G.; Gorte, B.; Sithole, G. & Rabbani, T. (2004b). Recognising structure in laser scanner point clouds. In *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences (IAPRS)*, volume 46 - 8/W2, pp. 33–38, Freiburg, Alemanha.

Vu, T. T.; Matsuoka, M. & Yamazaki, F. (2004). LIDAR-based change detection of buildings in dense urban areas. In *IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, volume 5, pp. 3413–3416, Anchorage, EUA.

Wahl, R.; Guthe, M. & Klein, R. (2005). Identifying planes in point-clouds for efficient hybrid rendering. In *The Pacific Conference on Computer Graphics and Applications*, pp. 1–8, Macao, China.

Wang, D. (1993). A neural model of synaptic plasticity underlying short-term and long-term habituation. *Adaptive Behavior*, 2(2):111–129.

Whaite, P. & Ferrie, F. P. (1991). From uncertainty to visual exploration. *IEEE Transaction on Pattern Analisys and Machine Intelligence*, 13(10):1038–1049.

Wren, C.; Azarbayejani, A.; Darrell, T. & Pentland, A. (1997). Pfinder: Real-time tracking of the human body. *IEEE Transaction on Pattern Analisys and Machine Intelligence*, 19(7):780–785.

Xiang, R. & Wang, R. (2004). Range image segmentation based on split-merge clustering. In *International Conference on Pattern Recognition (ICPR)*, volume 3, pp. 614–617, Cambridge, UK. IEEE Computer Society.

XSens Technologies B.V. (2009). *MTI - Miniature Attitude and Heading Reference System.* Enschede, Holanda. Disponível em http://www.xsens.com/images/stories/products/PDF_Brochures/mti009.pdf.

Yamazaki, I.; Natarajan, V.; Bai, Z. & Hamann, B. (2006). Segmenting point sets. In *IEEE International Conference on Shape Modeling and Applications (SMI)*, pp. 1–10, Matsushima, Japão.

Yokoya, N.; Kaneta, M. & Yamamoto, K. (1992). Recovery of superquadric primitives from a range image using simulated annealing. In *International Conference on Pattern Recognition (ICPR)*, volume 1, pp. 168–172, The Hague, Holanda.

Zakrajšek, E. (1976). On the generalized eigenvalue problem. *Journal of the Institute of Mathematics and its Applications*, 18(2):155–157.

Zhang, Y. (2003). *Superquadric Representation of Scenes from Multi-View Range Data.* PhD thesis, University of Tennessee, Knoxville, EUA.

Zou, W. & Ye, X. (2007). Multi-resolution hierarchical point cloud segmenting. In *International Multi-Symposiums of Computer and Computational Sciences Conference (IMSCCS)*, pp. 137–143, Iowa City, EUA.

Zwicker, M.; Pauly, M.; Knoll, O. & Gross, M. (2002). Pointshop 3D: an interactive system for point-based surface editing. *ACM Transaction on Graphics*, 21(3):322–329.

Anexo A

Projeto IRPS

Considerando o escopo e atividades do projeto mostrado na Figura A.1, a detecção de mudanças faz parte do cronograma de atividades e por isso uma das motivações do presente trabalho. A caixa vermelha na figura mostra as atividades relativas ao Instituto de Sistemas e Robótica (ISR), como já mencionado na Seção 1.2, por meio da detecção de mudanças 3D.

Inicialmente, a idéia era utilizar neste trabalho a abordagem do projeto *IRPS* para criação de mapas e a localização de um robô, por meio de um algoritmo de *SLAM* proposto por Blanco et al. [2008] e estendida para aquele projeto. O algoritmo de *SLAM* utiliza uma metodologia de mapas híbridos: métricos e topológicos. Assim, cada mapa topológico é composto por um mapa métrico, tornando o sistema muito mais eficiente para realizar o *SLAM* em grandes ambientes. A Figura A.2 ilustra a idéia, onde dois mapas topológicos em instantes T_i e T_{i+1} são mostrados com seus respectivos mapas métricos, filtros de partículas e sistemas de coordenadas independentes. A seta, que relaciona os mapas topológicos, representa que a transformação entre os sistemas de coordenadas dos mapas é conhecida.

Cada mapa topológico produzido para esse projeto poderia servir como entrada para o presente trabalho. Porém, existe ainda a necessidade de um segundo mapa, em um tempo posterior. Para tal, seria necessário um sistema de *Loop Closure*, capaz de identificar quando o mapa topológico gerado representa uma mesma região de um mapa anteriormente conhecido, permitindo, além do sistema de detecção de mudanças, a correção do mapa. No entanto, até o momento da finalização deste trabalho, tal sistema ainda não se encontra disponível.

A Figura A.3 ilustra dois mapas 3D gerados a partir do sistema de *SLAM* produzidos no contexto do projeto *IRPS*. O primeiro foi gerado em um ambiente de corredores e salas da zona de escritórios do Aeroporto de Faro, Portugal. Pode-se notar a exis-

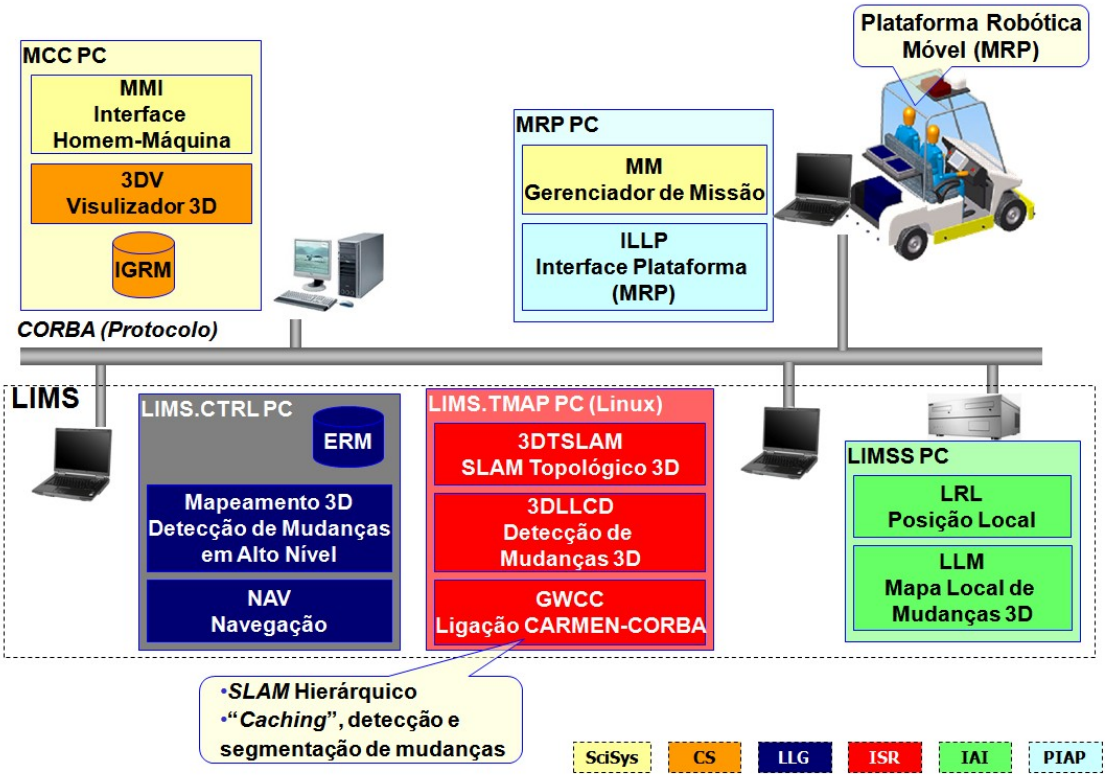


Figura A.1. Diagrama de módulos do projeto *IRPS*, com destaque para a caixa vermelha que representa as atividades desenvolvidas no ISR, relativas ao presente trabalho.

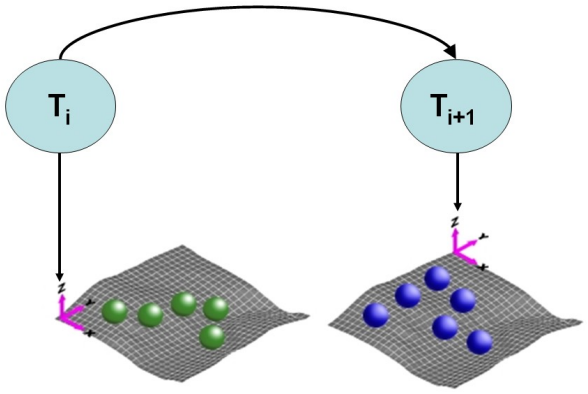


Figura A.2. Abordagem de *SLAM* híbrida utilizada no projeto *IRPS*, na qual dois mapas topológicos em instantes T_i e T_{i+1} são mostrados, com seus respectivos mapas métricos, filtros de partículas e sistemas de coordenadas independentes.

tência de bolas azuis, que representam o centro de regiões topológicas. Um segundo mapa é de uma área externa do mesmo Aeroporto. A Figura A.4 mostra imagens do ambiente dos quais os mapas 3D da Figura A.3 foram obtidos.

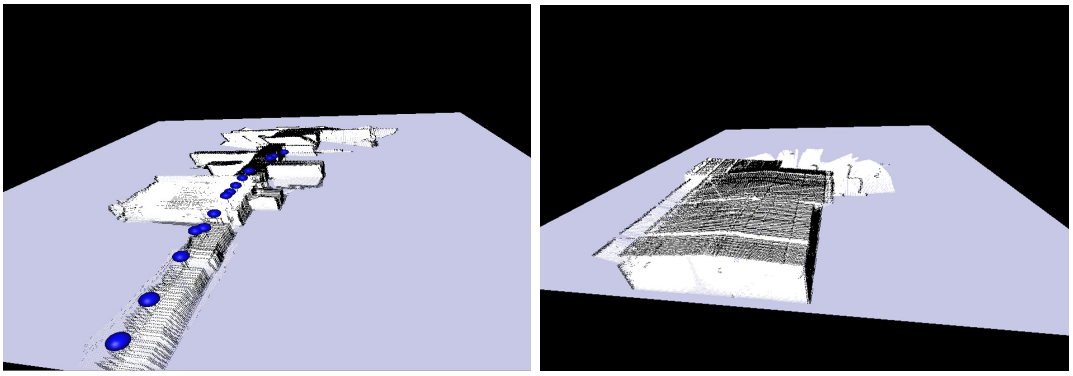


Figura A.3. Mapas 3D obtidos a partir da abordagem do projeto *IRPS*.

Figura A.4. Imagens reais dos ambientes dos mapas 3D da Figura A.3.

Para a geração dos mapas foram utilizadas duas plataformas experimentais, mostradas na Figura A.5. O primeiro, um robô *Segway Mobile Robotic Plataform (MRP)* [Segway Inc., 2009], e um segundo, um veículo elétrico fabricado pela empresa Melex [Melex A&D Tyszkiewicz Sp., 2009] e dotado de odometria pela empresa PIAP, ambas parceiras do projeto IRPS. Além disso, os dois veículos foram dotados de um sistema de dois *laser scanners* 2D SICK LMS 200 [Sick AG., 2009], montados ortogonalmente, como mostra a Figura A.5. As câmeras apesar de estarem na montagem são apenas para uso em trabalhos futuros.

Um arcabouço computacional utilizado no projeto *IRPS* para comunicação e controle com os dispositivos sensores, e com o próprio robô, foi o *software Carmen*. Trata-se de um conjunto de ferramentas livres, desenvolvidas pela Universidade de Carnegie Mellon, nos Estados Unidos. A idéia básica do *Carmen* é o desenvolvimento de módulos de *software* para comunicação entre dispositivos e aplicações. A parte central da ferramenta é o *Inter-Process-Communication* (*IPC*), que permite a troca de

Figura A.5. Arcabouço experimental do projeto IRPS, com duas plataformas móveis: veículo elétrico Melex, à esquerda, e Segway RMP, ao centro, ambos com sistema de dois *laser scanner* 2D montados ortogonalmente, mostrado em detalhes à direita.

mensagens entre processos. O *Carmen* possui diversos módulos já desenvolvidos para comunicação com a maioria dos robôs comerciais e sensores, tipicamente utilizados em robótica. A Figura A.6 ilustra a estrutura básica do *Carmen*. Os processos se comunicam com o *IPC* por meio de mensagens. As mensagens podem ser publicadas por meio do comando *publish* ou recebidas, por meio do comando *subscribe*. Maiores detalhes sobre o uso do *Carmen* podem ser encontrados no manual da ferramenta [Carmen Development Team, 2009].

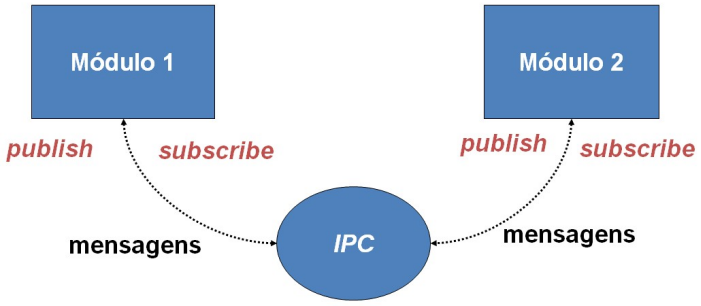


Figura A.6. Estrutura básica do *Carmen*, no qual os comandos *publish*, para publicação de mensagens, e *subscribe*, para solicitação, ao módulo *IPC* implementam a comunicação entre processo.