DENSIDADE DE OCUPAÇÃO ESPACIAL COMO REPRESENTAÇÃO PARA COMPARAÇÃO DE NUVENS DE PONTOS E APLICAÇÕES

ANTÔNIO WILSON VIEIRA

DENSIDADE DE OCUPAÇÃO ESPACIAL COMO REPRESENTAÇÃO PARA COMPARAÇÃO DE NUVENS DE PONTOS E APLICAÇÕES

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

Orientador: Mario Fernando Montenegro Campos

Belo Horizonte

Dezembro de 2012

Vieira, Antônio Wilson.

V658d Densidade de ocupação espacial como representação para comparação de nuvens de pontos e aplicações / Antônio Wilson Vieira — Belo Horizonte, 2012.
xxiv, 94 f. : il. ; 29cm.

Tese (doutorado) — Universidade Federal de Minas Gerais - Departamento de Ciência da Computação.

Orientador: Mario Fernando Montenegro Campos.

1. Computação - Teses. 2. Visão computacional - Teses. 3. Reconhecimento de padrões - Teses.
I. Orientador. II. Título.

519.6*85(043)

UNIVERSIDADE FEDERAL DE MINAS GERAIS
INSTITUTO DE CIÊNCIAS EXATAS
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

FOLHA DE APROVAÇÃO

Densidade de ocupação espacial como representação para comparação de nuvens de pontos e aplicações

ANTONIO WILSON VIEIRA

Tese defendida e aprovada pela banca examinadora constituída pelos Senhores:

PROF. MARIO FERNANDO MONTENEGRO CAMPOS - Orientador
Departamento de Ciência da Computação - UFMG

PROF. FLÁVIO LUIS CARDEAL PÁDUA
Departamento de Computação - CEFET/MG

PROF. MARCELO FERREIRA SIQUEIRA
Departamento de Informática e Matemática Aplicada - UFRN

PROF. THOMAS MAURICE LEWINER
Departamento de Matemática – PUC/RJ

PROF. WILLIAM ROBSON SCHWARTZ
Departamento de Ciência da Computação - UFMG

Belo Horizonte, 04 de dezembro de 2012.

Dedico este trabalho a todos os meus professores, do ensino fundamental à pós-graduação, especialmente Dona Eva e Neusa Faria.

Agradecimentos

A Deus, pelo dom da vida que se renova a cada dia e pelo milagre da esperança que justifica o amanhã;

Aos meus pais, pelo apoio e referência, e aos meus irmãos e cunhados, pela acolhida de sempre, especialmente nesse período em Belo Horizonte;

À minha noiva Fernanda, que, em detrimento da distância, esteve sempre ao meu lado com seu amor e carinho imprescindíveis;

Ao professor Mario, pela orientação, incentivo e, sobretudo, por acreditar no meu trabalho;

Ao amigo Zicheng Liu, da Microsoft Research, por compartilhar dados e ideias importantes ao desenvolvimento do trabalho.

Aos membros da banca, pela leitura cuidadosa do texto da tese, pelas críticas e sugestões;

Aos colegas do laboratório Verlab e do DCC, pelo ambiente de amizade e colaboração;

À secretaria do PPGCC, especialmente Renata e Sheila, pelo suporte eficiente em todas as demandas;

À Fundação de Amparo à Pesquisa do Estado de Minas Gerais (FAPEMIG), pelo apoio financeiro durante o curso;

Finalmente, à Universidade Estadual de Montes Claros (Unimontes), por ceder meu afastamento para o desenvolvimento deste trabalho.

"Há duas formas para viver a sua vida: Uma é acreditar que não existe milagre. A outra é acreditar que todas as coisas são um milagre."

(Fernando Pessoa)

Resumo

Este trabalho propõe padrões de densidade de ocupação espacial como representação de alto nível para objetos dados por nuvens de pontos em aplicações que requerem comparação entre diferentes objetos. Abordagens tradicionais utilizam métricas complexas que incluem informação estatística associada à distribuição subjacente aos pontos e são fortemente dependentes do modelo e do número de distribuições associadas ao conjunto. A representação por densidade de ocupação espacial, no entanto, é obtida em tempo linear e permite que a consulta seja feita em tempo constante. Além disso, permite que a comparação entre conjuntos seja feita eficientemente, em tempo linear. Neste trabalho, a densidade de ocupação espacial é construída de forma a obter uma representação em volumes implícitos para os dados ou histograma saturado de contagem de pontos por células espaciais. As vantagens dessa representação são exploradas em problemas de detecção de mudanças em ambientes $3D$ e no reconhecimento de ações humanas em sequências de mapas de profundidade. Em ambos os casos, a representação por densidade de ocupação espacial apresenta resultados superiores aos resultados do estado da arte.

Palavras-chave: Nuvens de pontos, operação booleana, detecção de mudança, reconhecimento de padrões.

Abstract

In this work, spatial density maps are proposed as a high-level representation for objects described by point clouds in applications that require comparison between different objects. Traditional approaches present complex metrics, which include statistical information associated with the underlying point distribution. They are highly dependent on the model and on the number of distributions of the set. Our method obtains a representation by spatial density maps in linear time, allowing for constant time query and efficient comparison between different sets. The spatial density pattern is constructed in order to obtain an implicit volume for the data or saturated histogram of points per voxel. The advantages of this representation are exploited in the problems of detecting changes in 3D environments and recognition of human actions in depth map sequences. In both cases, the spatial density maps representation achieved superior results compared to state-of-the-art methods.

Keywords: point clouds, boolean operation, change detection, pattern recognition.

Lista de Figuras

Lista de Tabelas

Sumário

Capítulo 1

Introdução

O desenvolvimento e o uso crescente de ferramentas de captura de dados tridimensionais (3D) tem possibilitado o acesso a uma grande variedade de modelos geométricos na forma de nuvens de pontos. Mais recentemente, dispositivos comerciais de baixo custo já fornecem dados na forma de mapas de profundidade em tempo real. Um mapa de profundidade pode ser visto como uma imagem onde cada elemento, ou *pixel*, está associado a um valor de profundidade com relação ao plano da câmera, sendo representado numa estrutura matricial onde a comparação entre diferentes conjuntos se resume a operação entre matrizes. Por outro lado, uma nuvem de pontos é geralmente um conjunto discreto $\mathcal{P} \subset \mathbb{R}^3$ amostrado de um objeto sólido sem estrutura de vizinhança ou conectividade, onde simples operações de comparação entre diferentes conjuntos se tornam uma tarefa muito complexa.

Enquanto é sempre possível representar um mapa de profundidade por uma nuvem de pontos, nem sempre é possível representar uma nuvem de pontos por um mapa de profundidade. Um mapa de profundidade I pode ser representado por uma nuvem de pontos mapeando cada pixel (i, j) ao ponto $(i, j, I(i, j)) \in \mathbb{R}^3$. Por outro lado, uma nuvem de pontos não tem, necessariamente, um único plano suporte ou distribuição homogênea de pontos que permita obter uma representação matricial como mapa de profundidade. Além disso, várias nuvens de pontos podem gerar o mesmo mapa de profundidade, dependendo da resolução deste. Desta forma, uma grande variedade de algoritmos de processamento de imagens, que podem ser estendidos para mapas de profundidade, não são aplicáveis a nuvens de pontos em geral.

Para um grande número de aplicações com nuvens de pontos $3D$, como renderização, detecção de mudança ou reconhecimento de padrões, uma representação de mais alto nível para a nuvem de pontos precisa ser construída. Para renderização, algoritmos de construção de malhas poligonais ou ajuste de superfícies implícitas aos

pontos são geralmente utilizados. Para comparação entre diferentes conjuntos, um agrupamento dos pontos em primitivas básicas é frequentemente aplicado para permitir operações em mais alto nível. Métodos de ajuste de superfícies, como reconstrução de superfícies, produzem uma estrutura de superfície combinatória ou malha geralmente triangular altamente eficiente para renderização. No entanto, tais métodos requerem que a distribuição de dados defina uma superfície e que, geralmente, a normal em cada ponto seja conhecida. Os métodos de *clustering* como modelagem em mistura de Gaussianas (GMM) geram uma abstração dos dados mesmo em presença de ruído e *outliers*. Entretanto, além do alto custo computacional dos métodos de *clustering*, as formas são limitadas a algumas primitivas implícitas, como quádricas e super-quádricas, que pouco se ajustam aos dados.

A tarefa de comparação entre diferentes conjuntos de pontos está relacionada à representação utilizada para os conjuntos e à métrica de similaridade utilizada. Neste trabalho, abordamos algumas aplicações que utilizam nuvens de pontos, onde a escolha da representação e da métrica de similaridade determinam a eficiência dos métodos de comparação e a robustez dos resultados. Como contribuição, propomos a representação de nuvens de pontos de densidade de ocupação espacial e duas estratégias para comparação nesta representação.

1.1 Definição do problema

O problema investigado neste trabalho pode ser assim formulado: Dadas duas nuvens de pontos $\mathcal{P}$ e $\mathcal{Q}$, possivelmente corrompidas por ruído e *outliers*, como representar os objetos definidos por $\mathcal{P}$ e $\mathcal{Q}$ de forma a permitir operações eficientes de comparação?

Diversas aplicações requerem a comparação entre objetos descritos por nuvens de pontos capturadas por algum dispositivo. Entretanto, a comparação direta entre nuvens de pontos é uma tarefa ineficiente e, frequentemente, ineficaz porque duas amostragens consecutivas do mesmo objeto podem gerar nuvens de pontos sem nenhum ponto em comum. Portanto, a comparação entre nuvens de pontos requer alguma estrutura que permita estabelecer uma relação de vizinhança. Quando um objeto é dado por uma nuvem de pontos $3D$, sua representação é um conjunto

$$\mathcal{P} = \{(x_i, y_i, z_i) \in \mathbb{R}^3, i = 1, \ldots, n\}, \tag{1.1}$$

onde os pontos são apresentados sem estrutura e a consulta pela ocupação de uma posição espacial $\mathbf{p}_i = (x_i, y_i, z_i)$ requer tempo linear de acesso a todos os pontos do conjunto ou $O(\lg n)$ em estruturas especializadas como *kd-trees*.

Em diversas aplicações, como detecção de mudança e reconhecimento de padrões em nuvens de pontos, uma etapa importante dos algoritmos propostos trata de encontrar uma representação para os objetos definidos pelas nuvens de forma a permitir comparação eficiente entre os objetos. Entre as abordagens adotadas para fornecer uma representação de alto nível para comparação, diversos trabalhos usam particionamento espacial [Brunet & Navazo, 1990], modelos de mistura de Gaussianas [Núñez et al., 2009], reconstrução de superfícies [Xujia Qin & Li, 2006] ou segmentação de primitivas básicas [Pauling et al., 2009]. Quando os dados são representados como uma função de densidade espacial, os pontos podem ser agrupados por níveis de densidade espacial para definir objetos sólidos de formas arbitrárias, não estando limitado a nenhum conjunto de primitivas básicas.

Na representação por densidade, uma função escalar

$$g : \Omega \subset \mathbb{R}^3 \rightarrow \mathbb{R}, \tag{1.2}$$

definida para uma caixa envolvente Ω do conjunto de pontos, pode ser usada para retornar o valor de densidade de ocupação espacial $g(\mathbf{p})$, para uma posição espacial $\mathbf{p}$, em tempo constante. Para efeito de armazenamento, o domínio de g pode ser uma amostragem discreta em uma grade definida para Ω, de forma que o armazenamento se torna dependente do nível de detalhes da representação pretendida. Neste caso, a orientação da grade pode ser alinhada com a distribuição dos pontos usando análise de componentes principais de forma a reduzir esparsidade.

Cumpre observar que a construção da amostragem de g em uma grade pode ser feita em tempo linear em função do número de pontos da nuvem. Uma vez construída g, a densidade em um ponto arbitrário $\mathbf{p} \in \Omega$ pode ser aproximada por interpolação trilinear ou simples consulta ao vértice mais próximo na grade. A resolução da grade é definida de acordo com a distância média entre pontos vizinhos e a orientação escolhida de acordo com direções principais da distribuição dos pontos.

1.2 Contribuições

Apresentamos, como contribuição deste trabalho, a exploração de representações eficientes para comparação e operações booleanas em nuvens de pontos baseadas em densidade de ocupação espacial. Em relação aos trabalhos mais recentes da literatura, mostramos que a representação por densidade de ocupação espacial proposta aqui traz as seguintes vantagens:

- Construção da representação em tempo linear: uma função densidade global é construída como somatório da contribuição de densidade local por cada ponto do conjunto, sendo que cada ponto é visitado apenas uma vez na construção da representação.

- Armazenamento dependente do nível de detalhes: a densidade global pode ser amostrada e armazenada em uma grade que pode ser recuperada por interpolação. O refinamento da grade depende do nível de detalhes requerido pela aplicação.

- Consulta em tempo constante: Uma vez construída a representação por densidade de ocupação espacial, a consulta pela ocupação de uma posição independe da quantidade de pontos do conjunto original, mas é feita em tempo constante pelo acesso ao elemento da grade.

- Comparação em tempo linear: A consulta de tempo constante pela ocupação espacial permite que a comparação entre diferentes conjuntos seja feita em tempo linear no número de pontos.

- Representação independente de primitivas básicas: O agrupamento por nível de densidade de ocupação permite que formas arbitrárias, sem restrições a geometria e topologia, sejam representadas.

Este trabalho apresenta ainda aplicações para a representação por densidade de ocupação espacial em problemas da literatura onde a representação por modelos de mistura de Gaussianas é substituída pela representação por densidade de ocupação espacial com ganhos quanto à acurácia nos resultados e eficiência em termos de tempo de processamento.

A primeira aplicação trata da detecção de mudanças em nuvens de pontos que é um tópico também de interesse em robótica móvel para tarefas de monitoramento de ambientes e reconstrução. Tipicamente, tais tarefas utilizam sensores laser onde é proposta a construção da representação por densidade contínua e um limiar de densidade é usado para agrupar os pontos em volumes implícitos e operações booleanas são aplicadas no campo escalar obtido para obter detecção de mudanças entre duas nuvens de pontos. Experimentos são apresentados para avaliar a robustez e eficiência da detecção de mudança. Nossos resultados, além de robustos, demonstram grande ganho em eficiência, de até 10 vezes, em comparação com resultados anteriores.

A segunda aplicação trata do reconhecimento de ações humanas em sequências de mapas de profundidade que é um tópico que tem ganhado relevância com a popularização dos sensores $3D$ de tempo real como o *Kinect*. Nessa aplicação, um

histograma saturado de contagem de pontos por células é usado como representação da densidade espacial e dois métodos de classificação são propostos para comparação de resultados usando uma base de dados pública. Nessa aplicação, o nosso método, além de melhorar a acurácia no reconhecimento e maior eficiência da representação, permitiu que resultados em tempo real pudessem ser obtidos.

1.3 Organização

Este trabalho está assim organizado: o Capítulo 2 apresenta uma revisão de alguns trabalhos da literatura relacionados à representação espacial e comparação de nuvens de pontos; o Capítulo 3 apresenta, como alternativa para representar a nuvem de pontos como função de densidade de ocupação espacial, a construção de um campo escalar global, com resultados em detecção de mudanças em nuvens de pontos. No Capítulo 4 apresentamos o histograma saturado de contagem de pontos como representação de densidade de ocupação espaço temporal. A seção 4.3 apresenta resultados em reconhecimento de ações humanas em mapas de profundidade e a Seção 4.3.4 apresenta uma combinação das nuvens de pontos com modelos gráficos (esqueletos) para recuperação do alinhamento espacial e invariância ao ponto de vista. No Capítulo 5, apresentamos as conclusões e direções para continuação do trabalho.

Capítulo 2

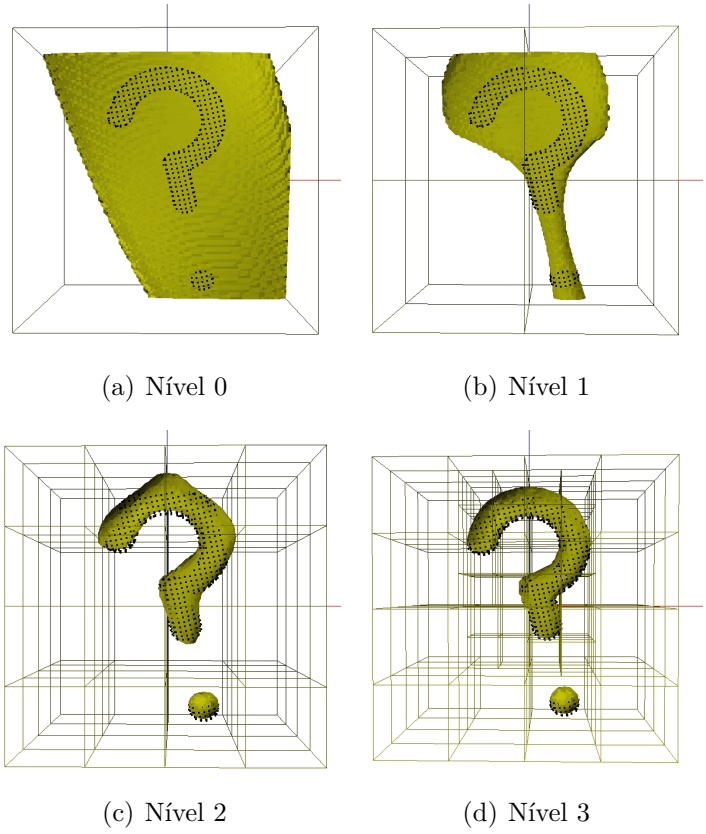
Trabalhos Relacionados

Apesar do crescente desempenho de *hardware* permitir processamento direto de densas nuvens de pontos para diversos propósitos, a maioria das aplicações requer um pré-processamento para construir uma estrutura de conectividade, ou *cluster*, para representar os dados na forma de superfície, modelos de mistura, combinação de primitivas geométricas ou "esqueletonização" de forma a possibilitar a comparação de similaridade ou mesmo operações entre diferentes conjuntos. Neste capítulo, apresentamos uma revisão de algumas das principais abordagens utilizadas para construir uma representação de mais alto nível para as nuvens de pontos e de técnicas usadas para comparação de similaridade.

2.1 Modelos implícitos

A reconstrução de modelos implícitos de superfícies a partir da nuvem de pontos obtém a conectividade na forma de uma malha triangular quando os dados estão bem organizados de forma a definir uma superfície, usualmente com normal conhecida em todos os pontos. O trabalho de Gabriel Taubin [Taubin, 1991] estima superfícies definidas por equações implícitas com aplicações na segmentação de mapas de profundidade. Diversos trabalhos recentes em reconstrução de superfícies constroem um campo escalar onde o nível zero define uma superfície implícita que se ajusta aos pontos: V-RIP [Curless & Levoy, 1996], RBF [Carr et al., 2001], MLS [Levin, 2003], MPU [Ohtake et al., 2003], Poisson [Kazhdan et al., 2006] e outros [Zheng & Zhang, 2011; Macêdo et al., 2011]. Uma abordagem intuitiva para reconstrução implícita é apresentada no trabalho de Ohtake et al. [Ohtake et al., 2003], onde é proposto um método baseado em partição da unidade em multiníveis (MPU, do inglês *Multilevel Partition of Unity*), um algoritmo de particionamento espacial onde uma *octree* é

construída e a cada nó é associada uma quádrica Q de melhor ajuste aos pontos. Um nó se torna folha se seus pontos se ajustam à função implícita gerada pelas quádricas na partição da unidade com um erro máximo pré-definido. Caso contrário, o nó se divide até encontrar um ajuste satisfatório. A Figura 2.1 ilustra esse processo.



(a) Nível 0 (b) Nível 1

(c) Nível 2 (d) Nível 3

Figura 2.1. Exemplo do processo de ajuste de superfície à nuvem de pontos usando *MPU Implicits*. A cada nó é associada uma quádrica de melhor ajuste aos pontos. Se o ajuste não for satisfatório, o nó se divide e uma nova quádrica é computada até que os pontos do nó se ajustem localmente com um erro aceitável.

Mais precisamente, a reconstrução considera um domínio fechado Ω como uma caixa envolvente do conjunto de pontos $\mathcal{P} \subset \Omega \subset \mathbb{R}^3$ e um conjunto de funções não-negativas de suporte compacto $\{\phi_i\}$ tais que formam uma partição da unidade em Ω, ou seja:

$$\sum_i \phi_i(p) = 1, \quad \forall p = (x, y, z) \in \Omega. \tag{2.1}$$

A cada subdomínio, restrito ao suporte compacto de ϕ_i, é associada uma aproximação local, Q_i, de forma a obter uma aproximação global

$$f(x,y,z) \approx \sum_i \phi_i(x,y,z) Q_i(x,y,z), \tag{2.2}$$

onde Q_i é uma superfície quádrica. A função $f(x,y,z)$, assim definida em Ω, é uma função implícita cujo nível zero é uma aproximação para a superfície definida pelos pontos de $\mathcal{P}$. Cada ϕ_i funciona como "peso" e pode ser definida como

$$\phi_i(x,y,z) = \frac{w_i(x,y,z)}{\sum_{j=1}^{n} w_j(x,y,z)}, \tag{2.3}$$

onde cada w_i pode ser uma *B-spline* definida para um suporte esférico ou outra função de distância suave de suporte compacto.

As aproximações locais Q_i, para um subconjunto de pontos $\mathcal{L} \subset \mathcal{P}$, são computadas como quádricas implícitas que se ajustam localmente a $\mathcal{L}$. Uma quádrica é uma superfície algébrica de grau dois, dada por uma superfície implícita $Q^{-1}(0)$. Usando uma notação vetorial, como proposto por Mederos et al. [Mederos et al., 2007], Q pode ser expressa pelo polinômio:

$$Q(x,y,z) = a_0x^2 + a_1y^2 + a_2z^2 + a_3xy + a_4xz + a_5yz + a_6x + a_7y + a_8z + a_9. \tag{2.4}$$

Escrevendo:

$$A = [a_0, a_1, a_2, a_3, a_4, a_5, a_6, a_7, a_8, a_9]^t \tag{2.5}$$

$$M = \left[x^2, y^2, z^2, xy, xz, yz, x, y, z, 1\right]^t \tag{2.6}$$

temos, em notação vetorial, a quádrica Q como um produto escalar $Q(x,y,z) = M^tA$. No caso ideal, em que cada ponto $\mathbf{p}_i \in \mathcal{L}$ pertence à mesma quádrica Q, temos que $Q(p_i) = M_i^tA = 0$ para todo $\mathbf{p}_i \in \mathcal{L}$. Na prática, o ajuste local busca encontrar o conjunto de coeficientes A de forma a minimizar as coordenadas de

$$\sum_i Q(\mathbf{p}_i) = M_i^tA. \tag{2.7}$$

Usando mínimos quadrados temos, por associatividade, que

$$\sum (M_i^tA)^2 = \sum (M_i^tAM_i^tA) = A^t \left(\sum M_iM_i^t\right) A. \tag{2.8}$$

Como a matriz obtida $T = \sum M_i M_i^t$ é um operador linear simétrico, existe uma base ortonormal dada pelos autovetores de T. Então, a quádrica de melhor ajuste local, Q, tem coeficientes A dados pelo autovetor associado ao menor dos autovalores da matriz T [Taubin, 1991].

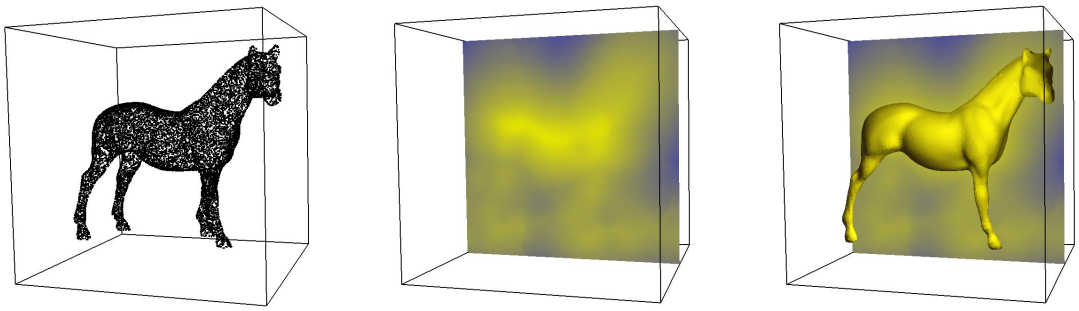
Usando esse processo, uma quádrica Q é computada como ajuste aos pontos de cada nó da *octree*.

A reconstrução implícita assim obtida constrói um campo escalar, do qual se extrai uma superfície de nível usando algoritmos de extração de superfícies como o *Marching Cubes* [Lorensen & Cline, 1987] para um campo escalar amostrado em uma grade uniforme. Como a uniformidade da grade pode gerar redundâncias e irregularidades na malha, pós-processamento para simplificação [Garland & Heckbert, 1997] e remoção de ruído [Fan et al., 2010; Vieira et al., 2010] são geralmente necessários. Uma malha com menos redundâncias e irregularidades é obtida com o *Dual Marching Cubes* [Schaefer & Warren, 2004] que faz amostragem adaptativa usando uma *octree* e extrai a superfície na grade dual (*octree dual*).

Esse método, simples e intuitivo de reconstrução de superfícies implícitas, tem a desvantagem de gerar alguns artefatos especialmente em regiões de geometria mais complexa. Diversos trabalhos, como o proposto por Gois et al. [Gois et al., 2007], foram propostos no sentido de evitar tais artefatos. A reconstrução de superfícies implícitas, como em [Ohtake et al., 2003; Gois et al., 2007], faz um ajuste local da superfície aos pontos e respectivas normais. Um ajuste global das normais pode ser obtido, como proposto por Kazhdan et al. [Kazhdan et al., 2006], onde a construção do campo escalar é modelado como uma instância do problema de Poisson. A Figura 2.2 ilustra uma nuvem de pontos (a), o campo escalar global obtido pela reconstrução implícita (b) e o conjunto de nível zero extraído como superfície de ajuste à nuvem de pontos (c).

A representação da nuvem de pontos em superfícies implícitas permite que uma série de operações complexas como *morphing* e operações booleanas sejam efetuadas. Em [Ohtake et al., 2003], exemplos de aplicações em modelagem são apresentados. Em [Xujia Qin & Li, 2006], propõe-se um algoritmo que usa a representação em superfícies implícitas para realizar operações booleanas em nuvens de pontos. Nesse algoritmo, as nuvens de pontos são previamente convertidas para superfícies implícitas usando interpolação variacional em Funções de Base Radial (RBF) [Carr et al., 2001] para, posteriormente, aplicar formas simples de operações booleanas em funções implícitas.

Em outra aplicação de superfícies implícitas, proposto por [Sprenger et al., 2000], é apresentado um algoritmo de *clustering* e visualização de grandes volumes de dados em uma estrutura hierárquica usando superfícies implícitas onde a hierarquia



(a) Nuvem de Pontos (b) Campo Escalar (c) Superfície de Nível

Figura 2.2. Ilustração do processo de extração de superfície de nível em um campo escalar global. Em (a) uma caixa envolvente é definida para a nuvem de pontos, em (b) uma secção do campo escalar obtido e, em (c), a superfície de nível zero extraída do campo escalar.

é construída pela extração de superfícies em diferentes níveis de densidade. A Figura 2.3 ilustra os níveis de hierarquia obtidos pelo *clustering* em superfícies implícitas.

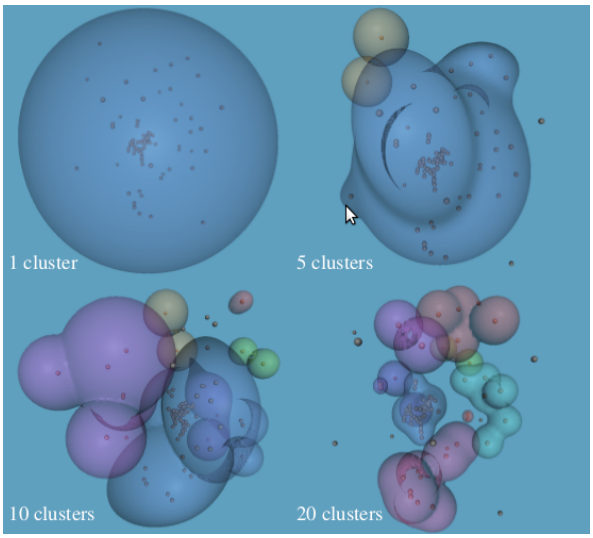


Figura 2.3. Ilustração do *clustering* hierárquico da nuvem de pontos por superfícies implícitas conforme [Sprenger et al., 2000].

A reconstrução de superfícies é mais adequada quando a distribuição dos dados é homogênea, sem muito ruído e *outliers*, definindo bem uma superfície, ou variedade topológica e, geralmente, com normal conhecida em cada ponto.

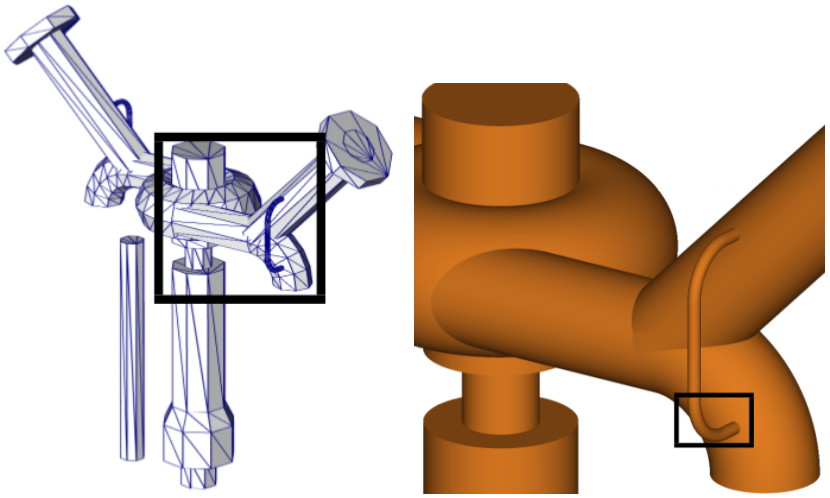
No Capítulo 3 exploramos o mesmo princípio de construção de um campo escalar global para a construção de uma representação para nuvens de pontos por densidade de ocupação espacial. Diferentemente da reconstrução de superfícies implícitas, onde o campo escalar é construído de forma que o nível zero defina uma superfície que se

ajuste aos pontos, a função escalar que construímos tem nível zero definindo o bordo de um volume espacial que agrupa os pontos dados. Esse agrupamento implícito é utilizado como representação de alto nível para operações de comparação entre nuvens de pontos.

2.2 Segmentação de primitivas

Uma forma muito utilizada de se obter uma representação em alto nível para nuvens de pontos é a segmentação da nuvem de pontos em primitivas geométricas básicas, como planos, cilindros, elipsoides, etc.

No trabalho de Toledo & Levy [Toledo & Levy, 2008], propõe-se uma estratégia de representação de dados tridimensionais para efeito de visualização de modelos industriais onde a maioria dos triângulos é substituída por primitivas implícitas como cones, cilindros e toros. Esta representação implícita, além de permitir mais rápido processamento com recursos de GPU, reduz memória para armazenamento, pois objetos implícitos são descritos por seus parâmetros em vez de armazenar vértices, normais e topologia dos modelos tesselados. Além disso, resulta em melhoria da qualidade da visualização (silhuetas suaves, sombreamento e profundidade por pixel e continuidade entre primitivas) conforme ilustrado na Figura 2.4.



(a) Modelo tesselado (b) Primitivas implícitas

Figura 2.4. Ilustração da representação por primitivas implícitas e comparação com modelo tesselado conforme [Toledo & Levy, 2008]. Em (a), um modelo tesselado mostra a aproximação de uma superfície suave por faces triangulares. Em (b), a representação por primitivas implícitas obtém suavidade e maior qualidade da visualização.

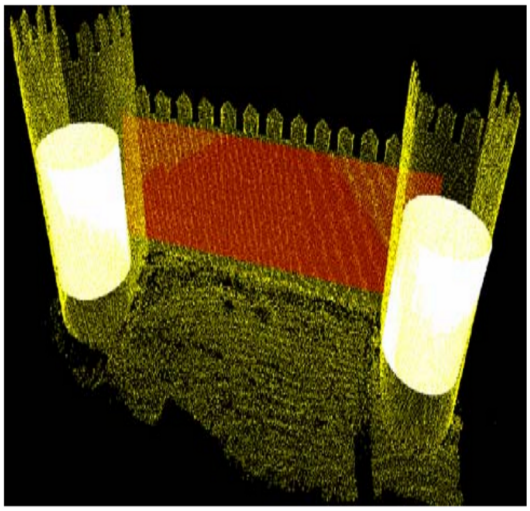


Figura 2.5. Ilustração da segmentação da nuvem de pontos por primitivas básicas conforme [Gonzálvez et al., 2007]. Dois cilindros e um plano são detectados para prover uma representação de alto nível para a nuvem de pontos.

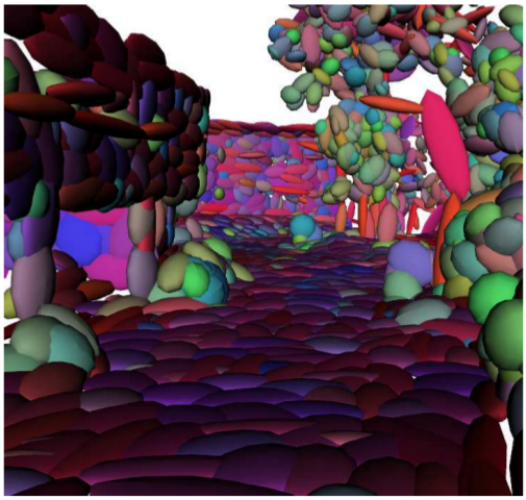


Figura 2.6. Ilustração da segmentação da nuvem de pontos por primitivas quádricas conforme [Pauling et al., 2009]. Um conjunto de elipsoides é usado para prover uma representação de alto nível para a nuvem de pontos.

Trabalhos como [Gonzálvez et al., 2007] propõem a detecção e segmentação da nuvem de pontos pela extração de primitivas como planos, cones e cilindros. Em [Pauling et al., 2009] é proposta a segmentação da nuvem de pontos por um conjunto de elipsoides, onde a segmentação é gerada através da manipulação de modelos intermediários e uma medida de dissimilaridade de forma entre pares de elipsoides no espaço euclidiano é apresentada. A Figura 2.5 ilustra a detecção de um plano e dois cilindros em uma nuvem de pontos e a Figura 2.6 ilustra a segmentação da nuvem de pontos por elipsoides.

Os trabalhos sobre segmentação em primitivas geralmente usam alguma técnica de *clustering*, extração de superfícies ou modelos de mistura que limitam consideravelmente o espectro de formas na composição da representação final. Nossa abordagem, no Capítulo 3, permite uma segmentação independente de qualquer conjunto de primitivas básicas, sendo bastante flexível para descrever uma variedade ampla de objetos com respeito à geometria e topologia.

2.3 Modelos de mistura

Em diversas aplicações, em particular em robótica, onde o sistema de aquisição geralmente é embarcado em algum veículo móvel com localização imprecisa, as nuvens de pontos obtidas são ruidosas e imprecisas. Além disso, a maioria das aplicações em mapeamento não requer a reconstrução de uma superfície em detalhes, mas uma representação em alto nível, suficiente para tarefas como localização e navegação. Neste contexto, modelos de mistura permitem construir primitivas básicas para subconjuntos de pontos em um conjunto maior pela identificação de uma mistura de distribuições. Os modelos de mistura de Gaussianas (GMM) são amplamente utilizados para prover uma estrutura de mais alto nível para nuvens de pontos não estruturados. Uma GMM é um modelo de mistura dado pela combinação afim de funções de densidade Gaussianas da forma:

$$f(\mathbf{p}, \Theta) = \sum_{k=1}^{K} w_k \Phi(\mathbf{p}; \mu_k, \Sigma_k), \tag{2.9}$$

onde:

$$\Phi(\mathbf{p}; \mu_k, \Sigma_k) = \frac{1}{\sqrt{(2\pi)^N |\Sigma_k|}} e^{-\frac{1}{2}((\mathbf{p}-\mu_k)\Sigma_k^{-1}(\mathbf{p}-\mu_k))}, \tag{2.10}$$

com μ_k e Σ_k representando a média e a matriz de covariância, respectivamente, e $|\Sigma_k|$ indica o determinante de Σ_k. Neste modelo, cada Gaussiana é associada a um coeficiente w_k, satisfazendo

$$w_k \geq 0 \qquad e \quad \sum w_k = 1. \tag{2.11}$$

Uma GMM modela uma nuvem de pontos em agrupamentos, ou *clusters*. Cada agrupamento corresponde a uma função de densidade Gaussiana com média no centroide do agrupamento e com uma matriz de covariância descrevendo a distribuição

dos pontos no agrupamento em forma de elipsoide, no caso $3D$. Finalmente, uma GMM pode ser denotada pelo vetor

$$\Theta = ((\theta_1, w_1), \ldots, (\theta_k, w_k)), \tag{2.12}$$

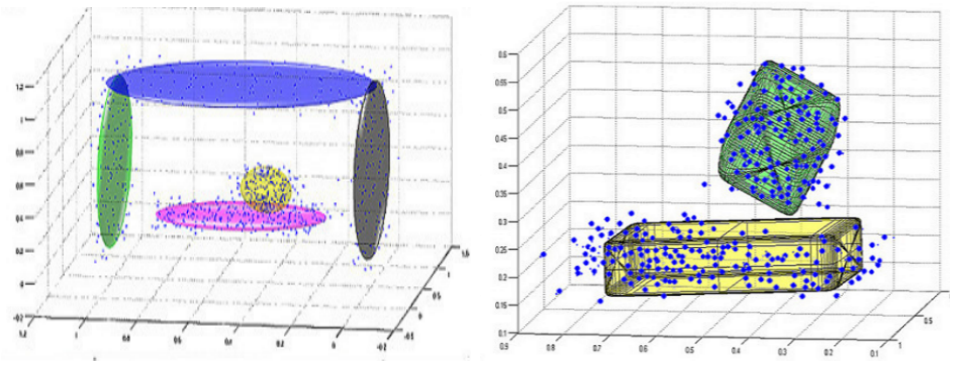
onde $\theta_k = (\mu_k, \Sigma_k)$ é um vetor contendo todas as coordenadas da média μ_k e todas as entradas da matriz de covariância Σ_k. Desta forma, Θ é um vetor com todos os parâmetros da GMM. A dimensão do vetor Θ é dado por $K(1 + N + N^2)$, onde K é o número de componentes da GMM e N a dimensão do espaço.

Uma questão vital na representação de dados como GMM é a computação dos parâmetros de cada θ_k. O algoritmo de Maximização da Expectativa (EM, do inglês *Expectation Maximization*) é geralmente utilizado para computar os parâmetros da GMM. A partir de um conjunto inicial de parâmetros, o algoritmo EM realiza um processo iterativo de maximização da verosimilhança até a convergência. Entretanto, a qualidade dos resultados, ou do ajuste da GMM à distribuição dos pontos, é fortemente dependente do conjunto inicial de parâmetros [Paalanen et al., 2006].

Uma representação similar às GMMs é a transformada de distribuições normais (NDT, do inglês *Normal Distribution Transform*). Na NDT, um conjunto de distribuições normais é ajustado à nuvem de pontos sem o processo iterativo de otimização usando EM. Uma grade é construída para agrupar os pontos por células espaciais e, para cada célula com uma quantidade mínima de pontos, é construída uma distribuição normal local. Este conjunto de distribuições é usado como representação para a nuvem de pontos como no trabalho de Stoyanov et al. [Stoyanov et al., 2011]. Originalmente, a NDT foi proposta por Biber & Strasser [Biber & Strasser, 2003] para alinhamento global de nuvens de pontos em $2D$ sem estabelecer, explicitamente, pares de pontos correspondentes e posteriormente estendido para 3D-NDT [Magnusson et al., 2007].

As representações baseadas em GMMs geram uma abstração da nuvem de pontos mesmo em presença de ruído e *outliers*. Vários trabalhos, como os relatados em [I. Amorim & Dias, 2008; Núñez et al., 2009; Pauling et al., 2009], propõem a utilização de GMM para modelar a nuvem de pontos com o propósito de detectar mudanças. Entretanto, a técnica se torna computacionalmente cara devido à utilização do algoritmo EM na construção da GMM, cujos parâmetros precisam ser estimados a cada nova nuvem de pontos. Além disso, as formas são limitadas a algumas primitivas básicas como elipsoides. Em [Drews Jr et al., 2010], os autores estendem o mesmo método para considerar super-quádricas como primitivas básicas. A Figura 2.7 ilustra

a segmentação da nuvem de pontos usando GMM. Em (a), a nuvem é ajustada por formas quádricas e, em (b), a nuvem é ajustada por formas super-quádricas.



(a) quádricas (b) super-quádricas

Figura 2.7. Ilustração da recuperação de formas quádricas e super-quádricas da nuvem de pontos, a partir de Misturas de Gaussianas conforme [Drews Jr et al., 2010]. Em (a), formas quádricas e, em (b), por formas super-quádricas.

O mesmo método foi posteriormente estendido por [Núñez et al., 2010], usando um algoritmo de casamento estrutural para determinar mudanças no espaço das GMM. Apesar do limitado número de testes realizados, resultados mostram vantagens em termos de custo computacional e sensibilidade ao número de Gaussianas necessárias para representar os dados. Na Seção 3.6, propomos um algoritmo para detecção de mudanças em nuvens de pontos usando volumes implícitos e mostramos que uma representação por densidade de ocupação espacial substitui a representação por GMM com vantagens em termos de complexidade de tempo e acurácia nos resultados obtidos.

Sequências de nuvens de pontos são utilizadas por Li et al. [Li et al., 2010] para reconhecimento de ações humanas usando GMM para caracterizar a forma das poses que são compartilhadas por diversos indivíduos na realização de diversas ações. Para tanto, assume-se que a distribuição dos pontos de uma certa pose pode ser aproximada por uma mistura de K componentes gaussianas. Na Seção 4.3, propomos um algoritmo para reconhecimento de ação em nuvens de pontos e mostramos que uma representação por densidade de ocupação espacial substitui a representação por GMM com vantagens em termos de complexidade de tempo e precisão nos resultados obtidos.

2.4 Esqueletos

Uma nuvem de pontos pode ser representada por uma abstração geométrica e topológica para fornecer uma estrutura de grafo que gera um esqueleto do objeto,

especialmente para objetos articulados como o corpo humano ou de animal, fornecendo uma abstração intuitiva que facilita o entendimento e comparação. Essa abstração representa a forma 3D de um objeto identificando suas partes significativas e mapeando sua relação geométrica em uma estrutura de grafo. Tais grafos são geralmente obtidos a partir de uma representação volumétrica usando a topologia de uma grade $3D$ [Bucksch & Appel van Wageningen, 2006] ou uma representação implícita [Sharf et al., 2007]. Como em [Sharf et al., 2007], Tagliasacchi et al. [Tagliasacchi et al., 2009] apresentam um método para extração de esqueletos de nuvens de pontos que se propõe a tratar mesmo nuvens de pontos incompletas. A Figura 2.8 mostra alguns exemplos de esqueletos obtidos a partir de nuvens de pontos.

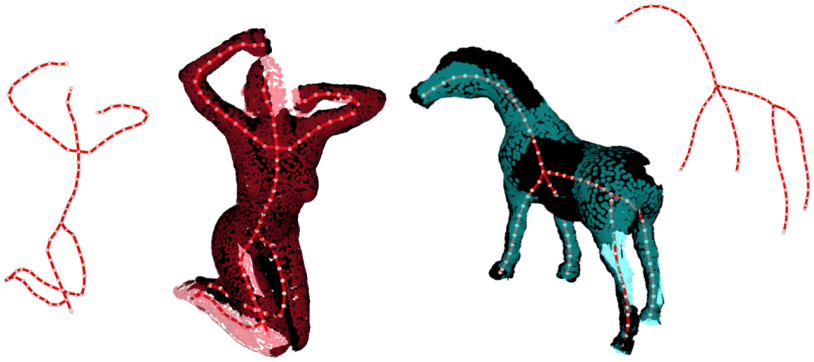
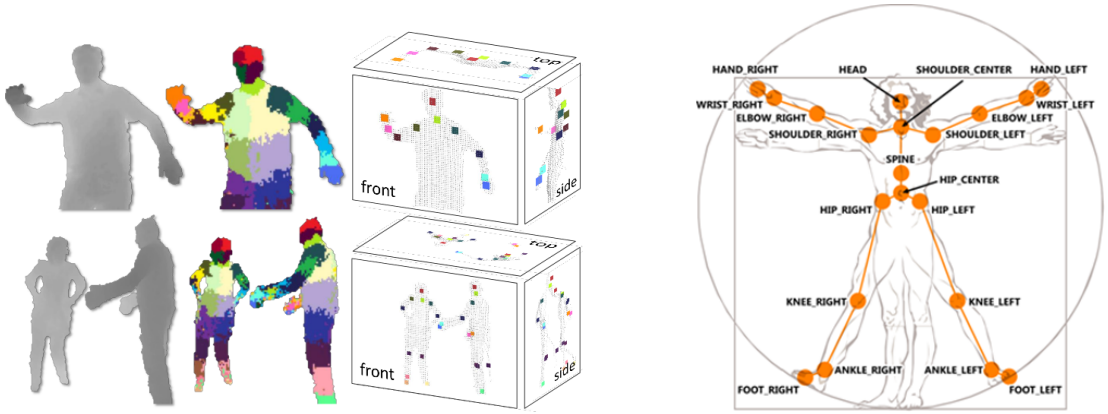


Figura 2.8. Exemplos de esqueletos extraídos de nuvens de pontos conforme em [Tagliasacchi et al., 2009]. Um conjunto de pontos interconectados geram uma estrutura de grafo como representação para a nuvem de pontos.

Os esqueletos obtidos com métodos desenvolvidos para tratar nuvens de pontos arbitrárias obtêm um grafo onde os nós e arestas não tem significado semântico e apenas fornecem uma abstração geométrica e topológica para o objeto. Em situações específicas, as nuvens de pontos descrevem objetos ou formas previamente conhecidas, de maneira que um significado semântico pode ser atribuído a cada nó do grafo como no caso de juntas do corpo humano. O trabalho proposto por Shotton et al. [Shotton et al., 2011] estima um esqueleto do corpo humano a partir da identificação de juntas do corpo em mapas de profundidade com invariância ao ponto de vista. A Figura 2.9 ilustra este processo. Em (a), uma única imagem de profundidade é usada para inferir uma distribuição de partes do corpo por *pixel*. Em (b), cada parte do corpo é associado a um nó de um grafo que define um esqueleto.

Como as juntas são identificadas, dado um par de nuvens de pontos não alinhadas, as coordenadas das juntas correspondentes definem um conjunto de pares de pontos correspondentes que pode ser usado para alinhar as nuvens de pontos. Na Seção 4.3.4,



(a) (b)

Figura 2.9. Ilustração do processo de estimação das juntas a partir do mapa de profundidade conforme [Shotton et al., 2011]. Com uma única imagem de profundidade de entrada, é inferida uma distribuição de partes do corpo por *pixel* (a). Cada parte do corpo é associado a um nó de um grafo que define um esqueleto como representação do corpo humano (b).

nossa representação por densidade de ocupação espacial, usada para reconhecimento de ações em nuvens de pontos, usa as juntas do esqueleto para obter um sistema de coordenadas local para cada nuvem de pontos de forma a obter invariância ao ponto de vista antes de computar a representação por densidade.

2.5 Medidas de similaridade

A comparação entre diferentes nuvens de pontos requer métricas adequadas de similaridade e, geralmente, essas métricas apresentam alta complexidade tornando ineficiente o seu uso em grandes nuvens de pontos, além de depender do tipo de representação usada para os pontos. A distância de Hausdorff é uma medida de similaridade para nuvens de pontos não organizados e a distância *Earth Mover's Distance* (EMD) é uma medida de similaridade aplicável quando a nuvem de pontos está organizada em distribuições multi-dimensionais como no caso das GMMs. A eficiência na comparação entre diferentes conjuntos de pontos está, então, diretamente relacionada à representação utilizada para os conjuntos de pontos, bem como à medida de similaridade. Nesta Seção descrevemos duas medidas de similaridade presentes na literatura, EMD e distância de Hausdorff.

2.5.1 *Earth mover's distance*

A medida *Earth Mover's Distance* (EMD), apresentada por [Rubner et al., 1998], é uma medida de dissimilaridade entre duas distribuições multi-dimensionais. Em diversos trabalhos de comparação entre nuvens de pontos, as nuvens são estruturadas em GMMs e a comparação é feita usando EMD [I. Amorim & Dias, 2008; Núñez et al., 2009; Drews Jr et al., 2010].

A distância EMD é definida como um tipo de distância entre duas distribuições de pontos $\mathcal{P}$ e $\mathcal{Q}$ no espaço, para os quais a distância par a par é dada, isto é, uma distância entre pontos. Essa distância entre conjuntos de pontos é baseada na solução de um tipo particular de problema de transporte em otimização linear. A ideia básica é que, dadas duas distribuições, pode-se imaginar uma delas como uma massa de terra espalhada no espaço, enquanto a outra pode ser vista como uma coleção de buracos no mesmo espaço. Pode-se sempre assumir que há pelo menos tanta terra quanto for necessário para encher todos os buracos (por comutação dos papéis de terra e buracos, se necessário). Em seguida, a $EMD(\mathcal{P}, \mathcal{Q})$ mede a menor quantidade de movimento de terra necessário para encher os buracos com terra.

Formalmente, sejam dois conjuntos $\mathcal{P}, \mathcal{Q} \subset \mathbb{R}^N \times \mathbb{R}^+$ dados por

$$\mathcal{P} = \{(\mathbf{p}_1, w_2), (\mathbf{p}_2, w_2), \ldots, (\mathbf{p}_m, w_m)\} \tag{2.13}$$

e

$$\mathcal{Q} = \{(\mathbf{q}_1, u_2), (\mathbf{q}_2, u_2), \ldots, (\mathbf{q}_n, u_n)\}, \tag{2.14}$$

onde $m \leq n$, $\mathbf{p}_i, \mathbf{q}_i \in \mathbb{R}^N$ são os pontos em cada conjunto e $w_i, u_i \in \mathbb{R}^+$ são pesos associados a cada ponto. Denote por W e U o somatório dos pesos em $\mathcal{P}$ e $\mathcal{Q}$, respectivamente, ou seja

$$W = \sum_{i=1}^{m} w_i \quad e \quad U = \sum_{i=1}^{n} u_i. \tag{2.15}$$

Então, a distância $EMD(\mathcal{P}, \mathcal{Q})$ entre os conjuntos $\mathcal{P}$ e $\mathcal{Q}$ é definida por

$$EMD(\mathcal{P}, \mathcal{Q}) = \min_{F \in \mathcal{F}(\mathcal{P}, \mathcal{Q})} \frac{\sum_{i=1}^{m} \sum_{j=1}^{n} f_{ij} d_{ij}}{\min\{W, U\}}, \tag{2.16}$$

onde d_{ij} denota a distância entre os pontos $\mathbf{p}_i \in \mathcal{P}$ e $\mathbf{q}_j \in \mathcal{Q}$ e $F = \{f_{ij}\} \in \mathcal{F}(\mathcal{P}, \mathcal{Q})$ com $\mathcal{F}(\mathcal{P}, \mathcal{Q})$ denotando o conjunto de todos os possíveis fluxos entre $\mathcal{P}$ e $\mathcal{Q}$ e sujeito às seguintes restrições:

$$f_{ij} \geq 0 \quad i = 1, \ldots, m; \quad j = 1, \ldots, n; \tag{2.17}$$

$$\sum_{j=1}^{n} f_{ij} \leq w_i, \quad i = 1, \ldots, m; \tag{2.18}$$

$$\sum_{i=1}^{m} f_{ij} \leq u_j, \quad j = 1, \ldots, n; \tag{2.19}$$

$$\sum_{i=1}^{m} \sum_{j=1}^{n} f_{ij} = \min\{W, U\}. \tag{2.20}$$

Para melhor entendimento do conjunto de restrições acima, considere o peso associado a cada ponto em um dos conjuntos de dados como uma porção de "terra" que precisa ser movida para o outro conjunto de dados, os buracos, sendo que o peso destes últimos pontos podem ser pensados como sendo a sua capacidade. O conjunto de pontos tomado como contendo terra é o que tem maior peso total. Nesse contexto, as quantidades f_{ij} representam a quantidade de terra que é movida entre a posição i de terra e o buraco j. Então, a primeira restrição (Equação 2.17) afirma que não há quantidade negativa de terra; a segunda (Equação 2.18) e terceira (Equação 2.19) restrições significam que a partir de cada ponto, um não pode prover mais terra do que a quantidade que ele contém, e não se pode colocar mais terra em um buraco do que a sua capacidade. Finalmente, a última restrição (Equação 2.20) significa que, no final, todos os buracos devem ser preenchidos. Em [I. Amorim & Dias, 2008], os autores observam que a EMD é uma distância, de fato, apenas quando o peso total de cada um dos conjuntos é o mesmo.

O problema de programação linear associado é então resolvido como o problema do fluxo ótimo em um grafo.

Alguns trabalhos propõem aproximações para EMD que podem ser computadas em tempo linear [Shirdhonkar & Jacobs, 2008; Jang et al., 2011]. Entretanto, para comparação de nuvens de pontos, a etapa de construção da representação em modelos de mistura consome tanto tempo que o tempo de comparação se torna insignificante na prática, de forma que tais aproximações não oferecem ganhos significativos no compto total do tempo de processamento. Os trabalhos apresentados em [I. Amorim & Dias, 2008; Núñez et al., 2009; Drews Jr et al., 2010] tratam da detecção de mudança em nuvens de pontos pela construção de GMMs e a computação da diferença usando EMD no espaço das Gaussianas, onde $\mathbf{p}_i$ e $\mathbf{q}_i$ representam parâmetros das distribuições e w_i e u_i os pesos associados a cada distribuição na mistura. Na Seção 3.6, mostramos

que a construção de uma representação por densidade de ocupação espacial permite que a diferença seja computada mais eficientemente do que a complexa combinação de GMMs e EMD, usando simples operações booleanas em campo escalar.

2.5.2 Distância de Hausdorff

A distância de Hausdorff é comumente usada como medida de similaridade entre duas nuvens de pontos [Cignoni et al., 1998b; Knauer et al., 2011]. Usando esta distância, um conjunto de pontos $\mathcal{P} = \{\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_m\}$ é considerado similar a outro conjunto $\mathcal{Q} = \{\mathbf{q}_1, \mathbf{q}_2, \ldots, \mathbf{q}_n\}$ se, e somente se, todo ponto em $\mathcal{P}$ está próximo de, pelo menos, um ponto em $\mathcal{Q}$ e vice-versa.

Formalmente, a distância de Hausdorff, $d_H(\mathcal{P}, \mathcal{Q})$, pode ser computada como

$$d_H(\mathcal{P}, \mathcal{Q}) = max\{\max_{\mathbf{p}\in\mathcal{P}}\{\min_{\mathbf{q}\in\mathcal{Q}}\{d(\mathbf{p}, \mathbf{q})\}\}, \max_{\mathbf{q}\in\mathcal{Q}}\{\min_{\mathbf{p}\in\mathcal{P}}\{d(\mathbf{p}, \mathbf{q})\}\} \tag{2.21}$$

onde $d(\mathbf{p}, \mathbf{q})$ indica distância Euclidiana. Em outras palavras, a distância de Hausdorff computa o máximo entre um elemento $\mathbf{p} \in \mathcal{P}$ e seu vizinho mais próximo $\mathbf{q} \in \mathcal{Q}$. Apesar de similar aos problemas de vizinho mais próximo e vizinho mais distante, computar a distância de Hausdorff é um problema mais complexo, já que envole tanto a maximização como a minimização em contraste com vizinho mais próximo e vizinho mais distante que usam apenas uma operação ou a outra. Uma forma tradicional de computar $d_H(\mathcal{P}, \mathcal{Q})$ usa uma busca linear em $\mathcal{P}$ e um índice para computar o vizinho mais próximo em $\mathcal{Q}$ para cada $\mathbf{p} \in \mathcal{P}$. A Figura 2.10 ilustra componentes do cálculo da distância de Hausdorff entre dois conjuntos.

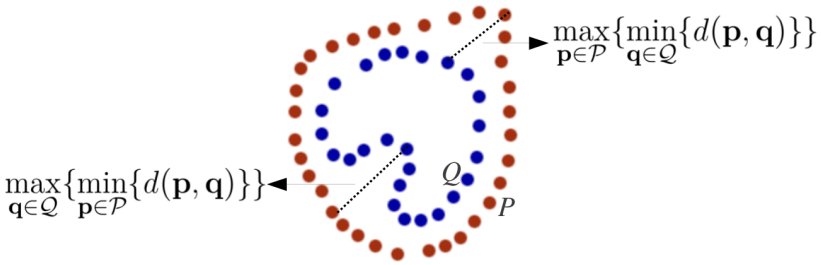


Figura 2.10. Ilustração das componentes do cálculo da distância de Hausdorff entre dois conjuntos

O problema de se determinar *max-min* entre dois conjuntos ocorre em problemas espaciais que requerem medida de similaridade entre dois conjuntos de pontos. Uma aplicação comum é a medida de similaridade entre duas formas como a comparação entre duas malhas triangulares para determinação de erro de simplificação ou

refinamento [Cignoni et al., 1998a]. Um algoritmo de força bruta calcula a distância de Hausdorff entre duas nuvens de pontos $\mathcal{P}$ e $\mathcal{Q}$ consistindo de m e n pontos, respectivamente, em tempo $O(mn)$.

Em [Li et al., 2010], a distância de Hausdorff é usada para computar a dissimilaridade entre duas nuvens de pontos em uma sequência de nuvens obtidas de ações humanas para efeito de comparação entre poses para reconhecimento de ações. Na Seção 4.3, mostramos que a construção de um histograma de ocupação espacial e comparação direta das densidades por células leva a uma comparação mais eficiente e robusta para este tipo de dados.

Capítulo 3

Densidade espacial como função escalar global

Neste capítulo apresentamos a construção da representação por densidade de ocupação espacial por função escalar contínua. Neste caso, as operações de comparação entre diferentes conjuntos são implementadas como operações booleanas entre volumes implícitos.

Dada uma nuvem de pontos $\mathcal{P} = \{(x_i, y_i, z_i) \in \mathbb{R}^3, 1 \leq i \leq n\}$ como representação para uma cena tridimensional, obtemos uma representação por densidade de ocupação espacial usando uma função escalar global. A principal ideia dessa representação para nuvens de pontos é a estimação de uma função densidade global desconhecida

$$G : \Omega \subset \mathbb{R}^3 \rightarrow \mathbb{R} \tag{3.1}$$

para uma caixa envolvente Ω, tal que $\mathcal{P} \subset \Omega$, de forma que G tenha maiores valores quando o ponto estiver mais próximo do objeto definido pelo conjunto de amostras em $\mathcal{P}$. Essa função densidade global deve definir um campo escalar suave a partir do qual um limiar de densidade h pode ser usado para agrupar pontos no volume implícito definido pelo conjunto $\mathcal{S} = \{\mathbf{p} \in \mathbb{R}^3; G(\mathbf{p}) \geq h\}$.

Dada uma nuvem de pontos $\mathcal{P}$ com n pontos, definimos, para cada ponto $\mathbf{p}_i \in \mathcal{P}$, uma função simples de densidade local $g_i : \mathbb{R}^3 \rightarrow \mathbb{R}$ que é uma função de base radial (RBF, do inglês *Radial Basis Function*) escolhida como sendo a Gaussiana:

$$g_i(\mathbf{p}) = exp\left(-\frac{\|\mathbf{p}_i - \mathbf{p}\|^2}{2\sigma^2}\right). \tag{3.2}$$

O parâmetro σ suaviza a influência de $\mathbf{p}_i$ sobre sua vizinhança. Escolhemos σ como sendo a maior distância Euclidiana entre dois pontos $\mathbf{p}_i, \mathbf{p}_j \in \mathcal{P}$ para que eles sejam considerados vizinhos. O valor de σ pode ser empiricamente definido de acordo com a escala da aplicação. Em nossos experimentos estimamos automaticamente σ como sendo o tamanho da aresta média da malha dual da octree definida para $\mathcal{P}$. Então, se $\mathbf{p}_i$ e $\mathbf{p}_j$ são vizinhos, o que significa $\|\mathbf{p}_i - \mathbf{p}_j\| \leq \sigma$, a contribuição de valor de densidade δ entre $\mathbf{p}_i$ e $\mathbf{p}_j$ é dada por

$$\delta = g_i\left(\mathbf{p}_j\right) = e^{-\frac{\|\mathbf{p}_i - \mathbf{p}_j\|^2}{2\sigma^2}} \geq e^{-\frac{1}{2}}. \tag{3.3}$$

Para definir a função de densidade global G, acumulamos as funções de densidade local g_i com as contribuições de todos os pontos $\mathbf{p}_i$, $1 \leq i \leq n$,

$$G\left(\mathbf{p}\right) = \sum_{i=1}^{n} g_i\left(\mathbf{p}\right). \tag{3.4}$$

Uma vez que a função de densidade local tem valor $g_i\left(\mathbf{p}_i\right) = 1$ para todo $\mathbf{p}_i \in \mathcal{P}$, se $\mathbf{p}_i$ tem k vizinhos, então $G\left(\mathbf{p}_i\right) \geq 1 + k\delta$. Para agrupar pontos com k ou mais vizinhos, ajustamos um limiar de densidade

$$h = 1 + k\delta. \tag{3.5}$$

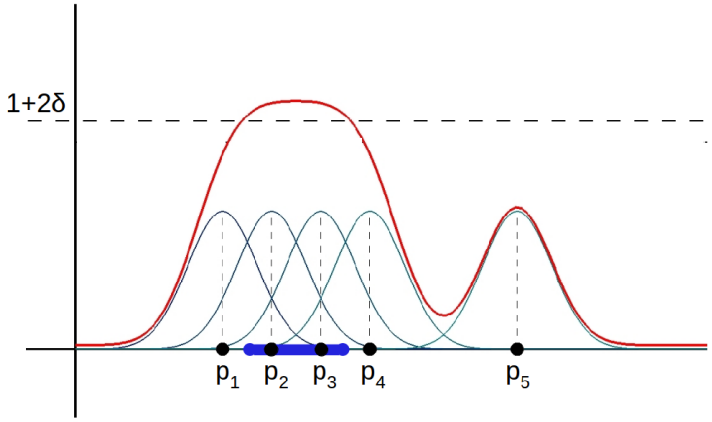
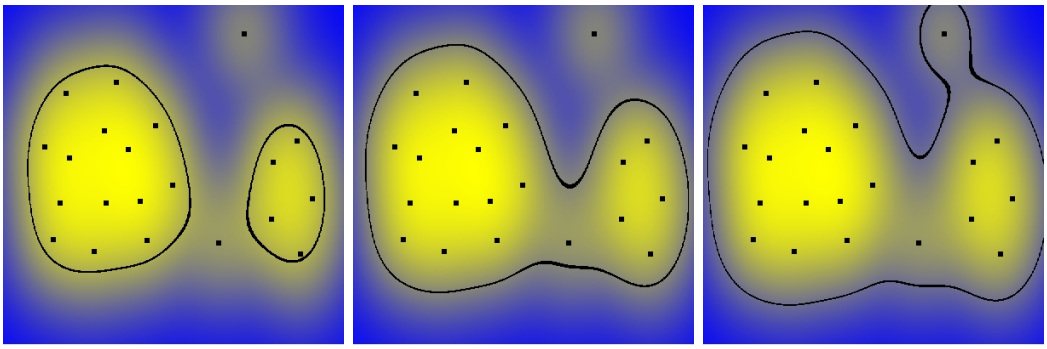


Figura 3.1. Exemplo de *cluster* em 1D. Funções de densidade local são acumuladas para definir uma função de densidade global. A barra azul, no eixo horizontal, ilustra o *cluster* $1D$ de pontos para os quais a densidade acumulada é superior a $1 + 2\delta$. Note que a densidade acumulada para p_1, p_4 e p_5 não é suficiente para incluí-los no *cluster*, dado o limiar de densidade.

A Figura 3.1 ilustra nossa estratégia de agrupamento para uma nuvem de pontos $1D$. A barra azul, no eixo horizontal, delimita o *cluster* de pontos com pelo menos

dois vizinhos, incluindo p_2 e p_3. Note que os pontos p_1, p_4 e p_5 não têm dois vizinhos de acordo com o limiar de densidade escolhido $h = 1 + 2\delta$ que agrupa pontos com, pelo menos, dois vizinhos. Note ainda que, se a densidade local usa um suporte ilimitado, um ponto com menos de k vizinhos pode acumular densidade superior a h. Esse efeito é minimizado com o uso de um *kernel* discreto de suporte compacto mas, independentemente do tamanho do suporte, todos os pontos com k ou mais vizinhos acumularão densidade superior a h e suficiente para serem incluídos no *cluster*.

Aumentando o parâmetro de suavização σ, leva a um aumento do raio da vizinhança e, consequentemente, mais pontos são incluídos no *cluster*. A Figura 3.2 ilustra esse efeito para uma nuvem de pontos $2D$. Note que, aumentar σ tem efeito equivalente a diminuir o limiar h. Como variar σ implica recalcular G em todos os pontos, em nossa implementação σ é mantido constante e, para variar o raio das vizinhanças, variamos apenas o limiar h, com a vantagem de que h é definido em função do número de vizinhos que define uma vizinhança.



(a) (b) (c)

Figura 3.2. Exemplo de agrupamento de pontos em $2D$. Valores crescentes do parâmetro de suavização σ, leva a volume crescente do *cluster* e mais pontos são incluído no *cluster*. Em (a), dois pontos são considerados *outliers*, em (b) apenas um ponto é considerado *outliers* e, em (c), nenhum *outlier* é considerado.

3.1 Modelos de densidade espacial

A forma construtiva da função densidade global G é um modelo não-paramétrico de estimação de densidade. Modelos paramétricos de densidade são geralmente ajustados aos dados para estimar uma função de densidade, assumindo que os dados são gerados por um modelo de distribuição conhecido cujos parâmetros são escolhidos de forma a obter o melhor ajuste aos dados. Entretanto, na maioria dos casos,

modelos paramétricos não se ajustam adequadamente a dados reais levando a uma falsa representação para os dados.

As GMMs são alternativas semi-paramétricas de modelos de densidade, onde um conjunto finito de modelos paramétricos são estimados para melhor se ajustar aos dados [Archambeau & Verleysen, 2003]. O conjunto de parâmetros da mistura, e respectivos pesos, são estimados via algoritmo EM para obter uma função de densidade global que maximize a probabilidade nos pontos do conjunto dado. Tais modelos são efetivos quando se tem uma boa estimativa do número de componentes necessárias, o que geralmente não é previamente conhecido, levando a falsa representação para os dados.

Em modelos não-paramétricos, como a densidade global G definida na Equação 3.4, uma função de densidade local conhecida é associada a cada ponto e a densidade global em cada ponto é estimada como soma, ou média, das funções de densidade locais definidas em cada ponto. Em contraste com modelos paramétricos, não é assumida uma forma para função de densidade, permitindo que assuma forma inteiramente arbitrária e determinada pelos dados. Um modelo especial não-paramétrico que estima a densidade subjacente sendo bastante geral é a estimação de densidade por *kernel* (KDE, do inglês *Kernel Density Estimation*) [Scott, 1992]. Em uma KDE, a função de densidade subjacente é estimada como a soma de funções *kernels*, tipicamente Gaussianas, com centro em cada ponto do conjunto. Note que isto é diferente de computar uma mistura de Gaussianas, já que uma Gaussiana é computada para cada ponto e nenhum processo de otimização para estimação de parâmetros ou pesos é necessário.

A flexibilidade do ajuste a formas arbitrárias com modelos não-paramétricos, como KDE, tem o alto custo computacional $O(mn)$ para estimar densidade global em m pontos a partir das densidades locais dadas por um conjunto de n pontos. De fato, a forma construtiva da função de densidade global G implica em complexidade linear para computar o valor de densidade em um simples ponto $\mathbf{p} \in \mathbb{R}^3$. A transformação rápida de Gauss (FGT, do inglês *Fast Gauss Transform*) [Greengard & Strain, 1991] permite que uma aproximação para a densidade em uma soma de m Gaussianas seja computada em n pontos em tempo $O(s^N(m+n))$, onde N indica a dimensão do espaço e s o tamanho do suporte do *kernel*. Considerando constantes N e s, essa complexidade se torna $O(m+n)$, em oposição ao tempo $O(mn)$ para abordagem simples, e pode ser utilizada para acelerar consideravelmente a estimativa por KDE. Naturalmente a complexidade cresce exponencialmente se variarmos a dimensão N do espaço.

3.2 Amostragem e interpolação

Em nosso trabalho, a complexidade para estimar a densidade global $G(\mathbf{c})$ em um ponto $\mathbf{c} \in \Omega$ é reduzida para tempo constante usando uma estratégia mais simples, porém eficiente, de amostragem discreta de G em uma grade e de interpolação triafim. Obtemos uma amostragem discreta para a função de densidade global computando seus valores para uma grade $3D$ definida para uma caixa envolvente da nuvem de pontos $\mathcal{P}$. Essa estratégia, eficiente para dados $3D$, torna-se proibitiva em altas dimensões, $N > 4$, pois a complexidade da amostragem discreta é exponencial na dimensão N. Nossos experimentos discutem a influência do tamanho da grade nos resultados. Para cada ponto $\mathbf{p}_i \in \mathcal{P}$, sua contribuição para a densidade local é calculada e somada apenas para um número constante de localizações da grade na sua vizinhança de acordo com o tamanho s de um *kernel* compacto discreto escolhido. Fazendo isso para uma nuvem de pontos com n pontos, a densidade global G é amostrada para todos os pontos da grade, em tempo $O\left(s^N n\right)$ que, considerando tamanho fixo s do suporte do *kernel* e dimensão $N = 3$, resulta complexidade linear $O\left(n\right)$ no número de pontos n. Os valores calculados para pontos da grade são usados como amostragem discreta para G na caixa envolvente Ω.

A fim de obter uma aproximação contínua para G em todos os pontos da caixa envolvente da nuvem $\mathcal{P}$, usamos um método de interpolação multivariada na grade regular [Bai & Wang, 2010]. Usando essa estratégia, a função de densidade global G é construída em tempo linear e um valor particular $G(\mathbf{p})$ pode ser calculado em tempo constante para cada ponto $\mathbf{p} \subset \Omega$. Mudanças na orientação da grade provocam pequenas oscilações no valor interpolado, mas sem impacto nas aplicações práticas. Essa oscilação é tão menor quanto maior o nível de detalhe, ou refinamento, da grade. Além disso, a complexidade do armazenamento não depende do número de pontos da nuvem, mas do tamanho da grade, o qual está relacionado com o nível de detalhe desejado para amostragem e definido pelos números n_x, n_y e n_z de segmentos em que a caixa envolvente é dividida ao longo dos eixos x, y e z, respectivamente. Denotamos por $n_x \times n_y \times n_z$ o tamanho da grade e por ρ o comprimento de cada segmento na grade.

Dado um ponto $\mathbf{c} \in \Omega$, o valor de densidade global $G(\mathbf{c})$ será então estimado como interpolação triafim dos vértices do cubo da grade que contém $\mathbf{c}$. A interpolação triafim estima o valor $G(\mathbf{c})$ de um ponto $\mathbf{c}$, interior ao cubo, pela interpolação afim nas arestas e biafim nas faces do cubo.

Apenas cinco arestas e duas faces precisam ser interpoladas para se obter a interpolação triafim e o resultado independe da escolha inicial das arestas e faces.

Em nossa grade de cubos, o tamanho do segmento ρ define o tamanho das aresta dos cubos. Dado o ponto $\mathbf{c} = (x, y, z)$, obtemos os vértices da grade com coordenadas $\mathbf{c}_{000} = (x_0, y_0, z_0)$ e $\mathbf{c}_{111} = (x_1, y_1, z_1)$ com $\mathbf{c}_{111} - \mathbf{c}_{000} = (\rho, \rho, \rho)$, de forma que $x_0 \leq x \leq x_1$, $y_0 \leq y \leq y_1$ e $z_0 \leq z \leq z_1$ e calculamos

$$(x_d, y_d, z_d) = (x - x_0, y - y_0, z - z_0)/\rho. \tag{3.6}$$

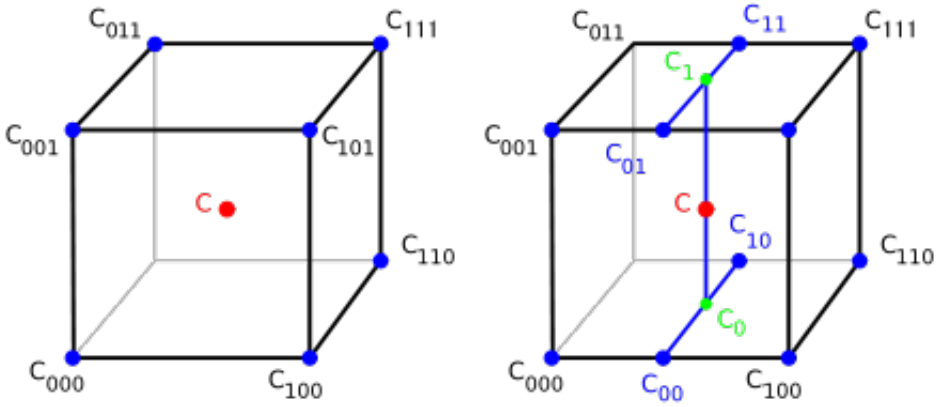


Figura 3.3. Ilustração do processo de interpolação $3D$. Em (a) um cubo onde os valores $G(\mathbf{c}_{ijk})$ são conhecidos para os vértices $\mathbf{c}_{ijk}$ do cubo e um ponto interior ao cubo onde o valor $G(\mathbf{c})$ é desconhecido. Em (b) o processo de interpolação nas arestas e nas faces para, finalmente, obter $G(\mathbf{c})$.

A Figura 3.3 ilustra o processo de interpolação triafim. Inicialmente, realizamos interpolação afim ao longo do eixo x, obtendo valores para quatro arestas:

$$G(\mathbf{c}_{00}) = G(\mathbf{c}_{000})(1 - x_d) + G(\mathbf{c}_{100})x_d; \tag{3.7}$$

$$G(\mathbf{c}_{10}) = G(\mathbf{c}_{010})(1 - x_d) + G(\mathbf{c}_{110})x_d; \tag{3.8}$$

$$G(\mathbf{c}_{01}) = G(\mathbf{c}_{001})(1 - x_d) + G(\mathbf{c}_{101})x_d; \tag{3.9}$$

$$G(\mathbf{c}_{11}) = G(\mathbf{c}_{011})(1 - x_d) + G(\mathbf{c}_{111})x_d. \tag{3.10}$$

Então, os valores obtidos nas quatro arestas são interpolados ao longo do eixo y para obter a interpolação biafim em duas faces:

$$G(\mathbf{c}_0) = G(\mathbf{c}_{00})(1 - y_d) + G(\mathbf{c}_{10})y_d; \tag{3.11}$$

$$G(\mathbf{c}_1) = G(\mathbf{c}_{01})(1 - y_d) + G(\mathbf{c}_{11})y_d. \tag{3.12}$$

Finalmente, obtemos a interpolação triafim em c interpolando, ao longo de z, os valores obtidos nas duas faces:

$$G(\mathbf{c}) = G(\mathbf{c}_0)(1 - z_d) + G(\mathbf{c}_1)z_d. \tag{3.13}$$

O procedimento acima estima $G(\mathbf{c})$ em tempo constante com acesso ao valor de G em oito vértices da grade, 14 operações de multiplicação e 14 operações de adição ou subtração. Dessa forma, a amostragem discreta de G é construída em tempo linear para uma grade e o valor de $G(\mathbf{c})$, para um ponto $\mathbf{c} \in \Omega \subset \mathbb{R}^3$, é estimado em tempo constante.

3.3 Agrupamento por volumes implícitos

Usaremos o limiar h e a função de densidade global G para particionar a caixa envolvente Ω da nuvem de pontos $\mathcal{P}$ em três subconjuntos:

$$S_1 = \{p \in \Omega; G(\mathbf{p}) < h\}, \tag{3.14}$$

$$S_2 = \{p \in \Omega; G(\mathbf{p}) = h\}, \tag{3.15}$$

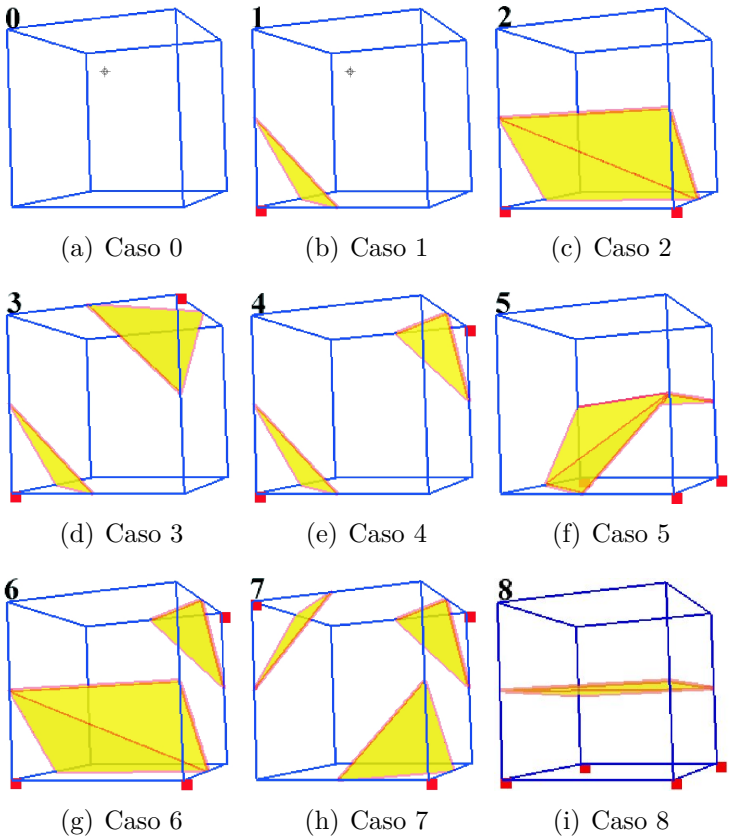
$$S_3 = \{p \in \Omega; G(\mathbf{p}) > h\}. \tag{3.16}$$

Ajustando um limiar de densidade $h > 1$, definimos o conjunto $S_2 \subset \Omega$ como sendo a superfície de contorno de um volume implícito que agrupa um subconjunto de pontos da nuvem $\mathcal{P} \subset \Omega$. Ajustando $h = 1 + k\delta$, definimos um volume que agrupa o subconjunto dos pontos que tenham pelo menos k vizinhos cuja distância é menor ou igual a σ. Esse volume é implicitamente definido pelo conjunto S_3.

Se h não é um valor regular para G, não necessariamente temos S_2 como uma superfície no sentido formal da topologia, devido a singularidades. Entretanto, um agrupamento de pontos por densidade estará bem definido.

Para representar e visualizar o conjunto S_2, extraímos do campo escalar uma superfície contínua, linear por partes, como superfície de contorno do agrupamento

de pontos usando o algoritmo *Marching Cubes* [Lorensen & Cline, 1987]. O método *Marching Cubes* produz uma malha triangular aproximando a imagem inversa $G^{-1}(h)$ a partir de uma grade de cubos onde os valores da função de densidade global G são conhecidos nos seus vértices. Cada vértice em um cubo da grade é classificado como positivo ou negativo, de acordo com o sua posição relativa no conjunto S_3. A Figura 3.4 ilustra alguns casos do *Marching Cubes*.



(a) Caso 0 (b) Caso 1 (c) Caso 2

(d) Caso 3 (e) Caso 4 (f) Caso 5

(g) Caso 6 (h) Caso 7 (i) Caso 8

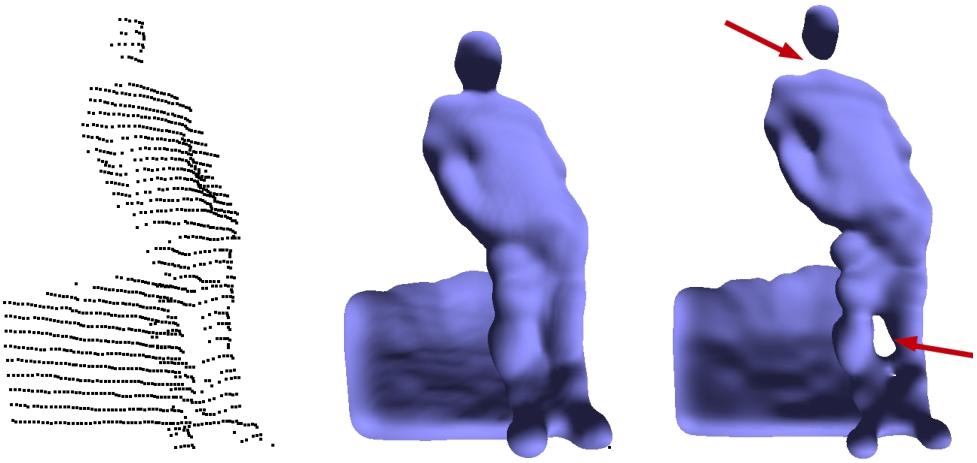
Figura 3.4. Configuração de alguns casos do algoritmo Marching Cubes conforme [Lorensen & Cline, 1987].

Em nossa implementação, usamos uma versão do *Marching Cubes* que produz uma malha com garantias topológicas de forma que obtemos uma aproximação para S_2 como uma superfície combinatória [Lewiner et al., 2003]. A malha resultante representa a superfície de contorno do volume implícito que agrupa os pontos em um certo intervalo de densidade. Diferentes conjuntos de nível podem ser extraídos para a mesma nuvem de pontos escolhendo um valor diferente para o limiar h.

Diferentemente da reconstrução de superfícies implícitas, onde o nível zero define uma superfície que se ajusta aos pontos dados, a função escalar que construímos define o bordo de um volume espacial que agrupa os pontos por um limiar de densidade. Em

nossos experimentos esse agrupamento implícito tem se mostrado mais eficiente como representação de alto nível para operações com nuvens de pontos quando comparado com representações baseadas nas distribuições estatísticas subjacentes aos conjuntos de pontos, como é o caso de GMMs. Além disso, essa representação proposta aqui permite que operações de comparação e operações booleanas em geral sejam efetuadas eficientemente.

A Figura 3.5-(a) mostra um exemplo de uma nuvem de pontos com alguns *outliers*; a superfície de nível extraída para um limiar de densidade global $h = 1 + 2\delta$ é mostrada na Figura 3.5-(b), e uma outra superfície de nível extraída para um limiar de densidade $h = 1 + 3\delta$ é mostrada na Figura 3.5-(c). Note que múltiplas componentes são diferenciadas. Note, ainda, que na Figura 3.5-(c) o valor maior de densidade leva ao aparecimento de componentes desconexas para a cabeça da pessoa e que nos membros inferiores, buracos aparecem separando as pernas.



(a) (b) (c)

Figura 3.5. Exemplo do agrupamento de pontos em $3D$. Em (a) a nuvem de pontos, em (b) o volume implícito definido para o limiar $h = 1 + 2\delta$ e, em (c) o volume implícito definido para o isovalor $h = 1 + 3\delta$ (c).

Em nossos experimentos, o volume implícito é definido para agrupar pontos com, pelo menos, $k = 3$ vizinhos. Então, usamos um limiar $h = 1 + 3\delta$ onde $\delta = e^{-\frac{1}{2}}$. O parâmetro de suavização σ é definido automaticamente como o tamanho da aresta média da malha dual da octree. O tamanho de um segmento ρ na grade regular depende do número de segmentos em que a caixa envolvente é dividida ao longo dos eixos principais para definir o tamanho da grade. O *kernel* discreto é computado para uma grade $s \times s \times s$ com $s = \frac{6\sigma}{\rho}$. Finalmente, o tamanho da grade, ou número de segmentos com que a amostragem de G é feita ao longo dos eixos, e que afeta

diretamente o desempenho do algoritmo, será tomada com valores variados para avaliar sua influência na acurácia da detecção e tempo de processamento.

3.4 Operações booleanas entre nuvens de pontos

Dado um limiar h, consideraremos uma função densidade global normalizada $\mathcal{G}(\mathbf{p}) = h - G(\mathbf{p})$, onde o valor zero define a superfície de contorno, valores negativos e positivos definem interior e exterior, respectivamente. O uso de $\mathcal{G}$ permite, sem perda de generalidade, o uso eficiente de operações booleanas nos volumes implícitos.

Sejam duas nuvens de pontos $\mathcal{P}_1$ e $\mathcal{P}_2$ representando dados atuais e de referência, respectivamente. Operações booleanas entre nuvens de pontos, como $\mathcal{P}_1 \cap \mathcal{P}_2$ ou $\mathcal{P}_1 - \mathcal{P}_2$, são geralmente complexas. Entretanto, usando suas funções de densidade global normalizada dadas por $\mathcal{G}_1, \mathcal{G}_2 : \mathbb{R}^3 \rightarrow \mathbb{R}$, como apresentado anteriormente, cada operação booleana constrói uma nova função densidade global $\mathcal{G}_3$ em um passo simples usando as seguintes operações entre funções:

$$\mathcal{P}_1 \cup \mathcal{P}_2 \Rightarrow \mathcal{G}_3(\mathbf{p}) = min(\mathcal{G}_1(\mathbf{p}), \mathcal{G}_2(\mathbf{p})); \tag{3.17}$$

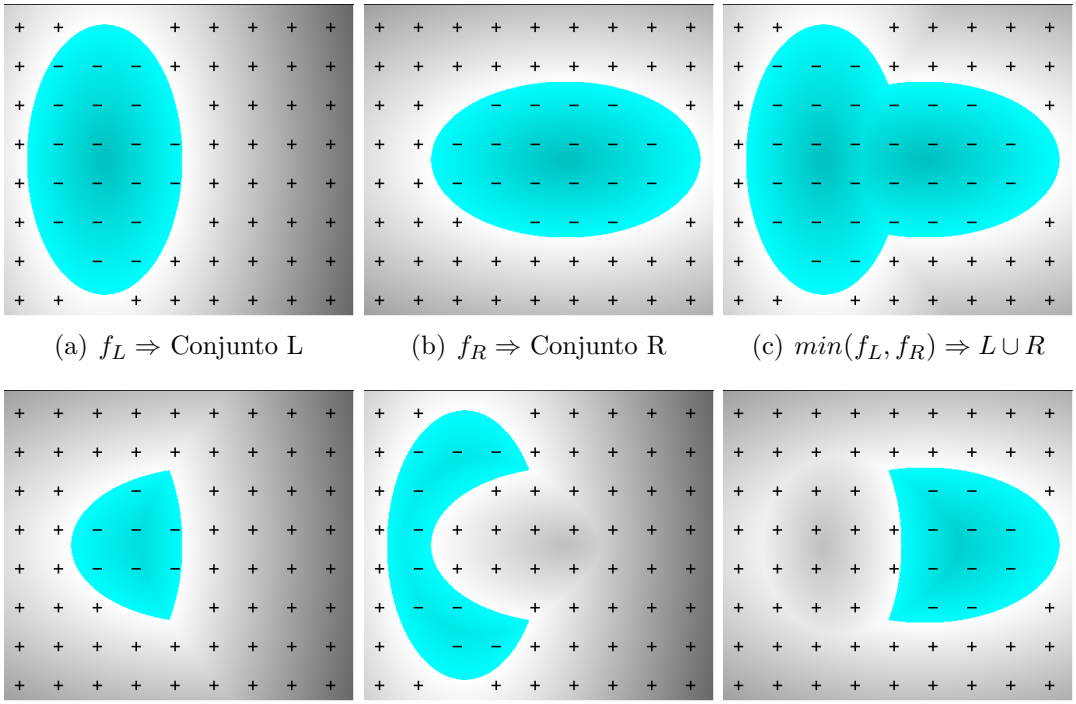
$$\mathcal{P}_1 \cap \mathcal{P}_2 \Rightarrow \mathcal{G}_3(\mathbf{p}) = max(\mathcal{G}_1(\mathbf{p}), \mathcal{G}_2(\mathbf{p})); \tag{3.18}$$

$$\mathcal{P}_1 \diagdown \mathcal{P}_2 \Rightarrow \mathcal{G}_3(\mathbf{p}) = max(\mathcal{G}_1(\mathbf{p}), -\mathcal{G}_2(\mathbf{p})); \tag{3.19}$$

$$\mathcal{P}_2 \diagdown \mathcal{P}_1 \Rightarrow \mathcal{G}_3(\mathbf{p}) = max(-\mathcal{G}_1(\mathbf{p}), \mathcal{G}_2(\mathbf{p})). \tag{3.20}$$

A Figura 3.6 ilustra as operações booleanas em dois campos escalares. Em (a) e (b), os conjuntos $\mathcal{L}$ e $\mathcal{R}$ são definidos pelos campos escalares f_L e f_R. Operações booleanas definem, em (c), união, em (d), intersecção e, em (e) e (f), diferenças. Na união, por exemplo, o mínimo entre as duas funções faz prevalecer valores negativos de ambas, resultando na união dos interiores.

As operações booleanas, como apresentado acima, são definidas para funções implícitas apenas [Bloomenthal, 1997; Mäntylä, 1988; Ricci, 1973]. Para aplicar as vantagens desse procedimento na comparação de nuvens de pontos, obtemos $\mathcal{G}_3(\mathbf{p}) = max(\mathcal{G}_1(\mathbf{p}), -\mathcal{G}_2(\mathbf{p}))$, que será uma nova função escalar cujos valores negativos definem um volume onde os pontos associados à diferença entre as duas nuvens devem estar. Finalmente, os pontos $\mathbf{p} \in \mathcal{P}_1$ são detectados como mudança se $\mathcal{G}_3(\mathbf{p}) < 0$, o que



(a) $f_L \Rightarrow$ Conjunto L (b) $f_R \Rightarrow$ Conjunto R (c) $min(f_L, f_R) \Rightarrow L \cup R$

(d) $max(f_L, f_R) \Rightarrow L \cap R$ (e) $max(f_L, -f_R) \Rightarrow L - R$ (f) $max(-f_L, f_R) \Rightarrow L - R$

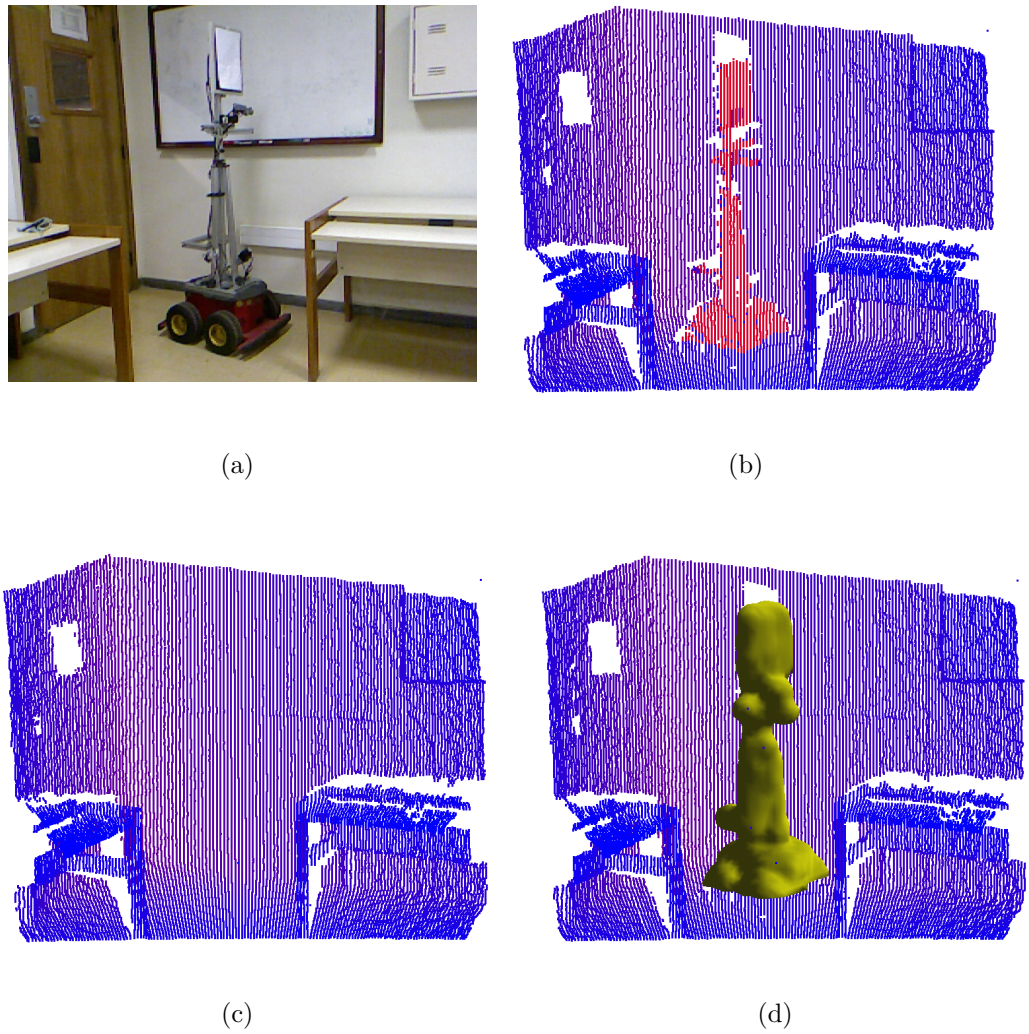
Figura 3.6. Exemplo de operações booleanas em campos escalares. Valores negativos definem o interior dos diversos conjuntos e operações booleanas são obtidas pela combinação do sinal das funções escalares.

é computado em tempo linear. O Algoritmo 1 detalha o nosso método para detecção de mudança.

Algorithm 1 ChangeDetect($\mathcal{P}_1, \mathcal{P}_2, h$)

1: Compute $\mathcal{G}_1$ from $\mathcal{P}_1$
2: Compute $\mathcal{G}_2$ from $\mathcal{P}_2$
3: $\mathcal{D} = \emptyset$
4: **for** each $\mathbf{p} \in \mathcal{P}_1$ **do**
5: **if** $(max(\mathcal{G}_1(\mathbf{p}), -\mathcal{G}_2(\mathbf{p})) < 0)$ **then**
6: $\mathcal{D} \leftarrow \mathcal{D} \cup \{\mathbf{p}\}$
7: **end if**
8: **end for**
9: return $\mathcal{D}$

Uma aplicação para esta representação é discutida na Seção 3.6 onde uma representação por GMMs é substituída pela representação por densidade de ocupação espacial e a detecção de mudança, usando EMD, é substituída por simples operações booleanas.



(a) (b)

(c) (d)

Figura 3.7. Exemplo de representação por densidade de ocupação espacial e operação booleana. Em (a), a imagem de um ambiente com a presença de um robô, em (b) uma nuvem de pontos do ambiente com o robô, em (c) uma nuvem de pontos de referência do ambiente sem o robô e, em (d), o volume obtido pela operação booleana de diferença entre as densidades de ocupação dadas pelas nuvens de pontos.

A Figura 3.7 mostra um exemplo de representação por densidade de ocupação e segmentação usando operação booleana. Note que, como apresentado, o Algoritmo 1 computa mudanças como objetos que aparecem na cena. Diferenças associadas ao desaparecimento de objetos podem ser computadas usando o mesmo algoritmo permutando $\mathcal{P}_1$ e $\mathcal{P}_2$ na entrada. Da mesma forma, a união e intersecção podem ser computadas com o mesmo algoritmo pela substituição da operação booleana adequada na Linha 3 pela operação correspondente.

3.5 Alinhamento espacial

Uma condição necessária para que a comparação proposta seja efetiva é que as diferentes nuvens de pontos sejam capturadas de um mesmo ponto de vista, sendo descritas em um mesmo sistema de coordenadas de referência. Entretanto, nem sempre esta condição é satisfeita. Nesta Seção, apresentamos alternativas para recuperar o alinhamento de forma a permitir que a comparação seja efetiva. Em casos onde o desalinhamento é pequeno, um algoritmo iterativo chamado *Iterative Closest Point* (ICP) [Besl & McKay, 1992] encontra o ajuste local ótimo. Para casos em que esse ajuste ótimo local não fornece um alinhamento ótimo global, um ajuste prévio é necessário, geralmente usando algum descritor local invariante a transformações rígidas para obter um conjunto de pares de pontos correspondentes nas duas nuvens de pontos.

O algoritmo ICP é apresentado em diversas variantes [Rusinkiewicz & Levoy, 2001]. Em nossa implementação, selecionamos, iterativamente, pares de pontos mais próximos $\{(\mathbf{p}_j, \mathbf{q}_j) \in \mathcal{P} \times \mathcal{Q}; \|\mathbf{p}_j - \mathbf{q}_j\| \leq \tau\}$ para um valor de limiar τ e usamos mínimos quadrados para encontrar a transformação rígida $\mathbf{R}$ que minimiza a função custo

$$\sum_j \|\mathbf{p}_j - \mathbf{R}(\mathbf{q}_j)\|^2. \tag{3.21}$$

A nuvem $\mathcal{Q}$ é então alinhada à nuvem $\mathcal{P}$ usando a transformação $\mathbf{R}$ e este processo é repetido até que $\mathbf{R}$ se aproxime suficientemente de uma matriz identidade, significando que os dois modelos estão suficientemente alinhados. Mesmo havendo diferença entre as nuvens, desde que regiões comuns possam ser identificadas, o ICP ainda produz o ajuste local porque o limiar τ ignora regiões com muito afastamento.

Em situações em que o ajuste local não é satisfatório, a convergência pode levar para um mínimo local que não fornece um ajuste ótimo global. Nesse caso, descritores especializados para representar de forma invariante pontos chave escolhidos nas nuvens de pontos podem ser usados para estabelecer pares de pontos correspondentes a partir dos quais uma transformação rígida inicial é obtida [Nascimento et al., 2012; Planitz et al., 2005]. Geralmente, os descritores locais obtêm pares de correspondências com algumas incorreções devido a ambiguidades. Dessa forma, a transformação rígida obtida é tratada como uma aproximação global para, posteriormente, ser refinada localmente como o ICP. O descritor *Spin Image* [Johnson, 1997] é um exemplo de descritor aplicável a nuvens de pontos gerais, sem estrutura matricial ou vizinhança como as nuvens que tratamos.

Nos dados de experimentos, oriundos de capturas em sequências de mapas de profundidade usados neste trabalho, as nuvens de pontos não apresentaram desalinhamento significativo e obtiveram alinhamento adequado apenas com ajuste local usando ICP. Esta estratégia foi aplicada para ajuste de todos os conjuntos de pontos usados em nossos experimentos. Note que as abordagens de comparação de nuvens de pontos baseadas em GMM também consideram as nuvens previamente alinhadas de forma que essa etapa do processamento, comum em ambos os algoritmos, não foi considerada no compto do tempo para comparação dos algoritmos.

3.6 Aplicação em detecção de mudanças em nuvens de pontos

Nesta seção, apresentamos uma aplicação que trata da detecção de mudanças em nuvens de pontos, que é um tópico de interesse em robótica móvel para tarefas de monitoramento de ambientes e reconstrução usando sensores laser. Nessa aplicação, propomos a construção da representação por volumes implícitos e operações booleanas para detecção de mudanças.

Várias atividades em vigilância automática requerem a capacidade de se detectar, eficientemente, mudanças nos ambientes por muitas razões. Por exemplo, uma mudança pode revelar uma bagagem abandonada em um aeroporto, um deslizamento de terra em uma mina abandonada ou o andamento de uma construção em um canteiro de obras. Além disso, os robôs autônomos, que são capazes de explorar e navegar em um ambiente dinâmico e desconhecido precisam determinar sua pose e, simultaneamente, construir mapas com base em suas habilidades de percepção (SLAM) [Thrun et al., 2005]. Depois de um reconhecimento inicial do ambiente, o robô deve ser capaz de detectar e responder a alterações em sua volta. A capacidade de detectar mudanças é fundamental para a vigilância robótica e sistemas de segurança [Drews Jr et al., 2010], onde mudanças podem implicar ameaças potenciais. Em ambientes perigosos, um robô deve ser capaz de identificar situações de risco com base nas alterações que são detectadas ao longo de sua trajetória em relação a um mapa previamente conhecido. Portanto, a detecção de mudanças surge como um recurso importante para permitir que um robô possa se adaptar a situações novas e manter a operação correta a despeito de mudanças no ambiente.

A ideia básica por trás da maioria das abordagens em detecção de mudanças é combinar a leitura atual dos sensores com informações previamente coletadas sobre o meio ambiente, que pode ser um mapa obtido anteriormente ou algum outro tipo

de representação abstrata como, por exemplo, modelos obtidos por *Computer Aided Design* (CAD). O sucesso do processo, como descrito em [Núñez et al., 2010], é geralmente condicionado a:

- Existência de sensores precisos capazes de obtenção de informação a partir do ambiente;
- Disponibilidade de algoritmos rápidos e capazes de extrair uma representação de alto nível para um grande conjunto de dados ruidosos e incertos;
- Existência de um método preciso capaz de detectar alterações possíveis de acordo com as representações utilizadas.

A fim de cumprir a primeira condição, isto é, perceber o ambiente, sensores de profundidade ou baseados em sistemas de visão $3D$ são normalmente usados. Por um lado, métodos baseados na extração de características visuais sofrem com mudanças de iluminação e muitas vezes exigem o uso de CPU de maior capacidade, devido à complexidade dos algoritmos de visão computacional [Núñez et al., 2010]. Por outro lado, existem sensores a laser, capazes de adquirir diretamente as nuvens de pontos em $3D$, e são cada vez mais precisos e menos dispendiosos. Atualmente, é possível adquirir nuvens de pontos $3D$, tanto em ambientes internos, por exemplo, usando sensor *Kinect*, e em ambientes abertos usando *scanners* $3D$ a laser. Esses sensores fornecem nuvens de pontos de boa qualidade, mesmo em tempo real, com uma pequena quantidade de ruído.

Para tratar a segunda questão, a nuvem de pontos precisa ser representada em modelos de mais alto nível semântico, onde os métodos de agrupamento de pontos são largamente utilizados. Métodos de agrupamento, como Mistura de Gaussianas (GMM) [Núñez et al., 2009], fornecem uma abstração para os dados, mesmo na presença de ruído e *outliers*. No entanto, além do alto custo computacional de métodos de agrupamento para nuvens de pontos, as formas possíveis estão limitadas a algumas primitivas implícitas que mal se ajustam aos dados. Quando os dados são representados como uma função de densidade de ocupação espacial, um volume implícito definido por um intervalo de densidade pode ser usado para agrupar os pontos, como mostramos na Seção 3.3. A vantagem de tal agrupamento com base em densidade espacial é que as formas dos objetos não estão limitadas a nenhum conjunto de primitivas específicas, mas são bastante flexíveis para descrever uma variedade ampla de objetos com respeito à sua geometria e topologia. Em [Sprenger et al., 2000], esse agrupamento por densidade é utilizado para propor um método de agrupamento hierárquico baseado

em superfícies implícitas que são adequados para a visualização de grandes conjuntos de dados.

Quanto à terceira questão, várias métricas têm sido propostas para detectar alterações usando dados adquiridos pelos sensores $3D$ [Brunet & Navazo, 1990; Paalanen et al., 2006; I. Amorim & Dias, 2008]. Diferentemente das imagens de intensidade ou profundidade, onde os dados tem uma estrutura matricial de forma que que as comparações são simples operações de diferença entre matrizes, a comparação entre duas nuvens de pontos continua a ser um desafio considerável. Uma das razões é que a informação de vizinhança não está disponível e, devido ao ruído e *outliers* na aquisição de dados, mesmo que duas nuvens de pontos sejam amostras consecutivas e bem alinhadas de uma mesma cena, elas podem não compartilhar nenhum ponto em comum. Além disso, nuvens de pontos não têm uma estrutura matricial que permita a operação de diferença direta. Assim, uma comparação quadrática, a princípio, deve ser empregada para detectar aqueles pares de pontos cuja distância é menor que um dado limiar.

A fim de reduzir o custo computacional desse processo, métricas mais complexas que incluem a informação estatística associada à distribuição subjacente dos pontos têm sido utilizadas. A principal desvantagem desse tipo de abordagem é sua forte dependência do número de distribuições associadas com o mapa além das restrições de forma dos modelos de distribuição.

Kaestner et al. [Kaestner et al., 2005] propõem um método onde uma abordagem probabilística é usada para alinhamento e detecção de mudança usando dados de sensores laser. O alinhamento proposto fornece um registro não rígido da nuvem de pontos e os erros de estimativa e mudanças entre conjuntos de dados são detectadas usando um limite probabilístico para reconhecer as mudanças. No entanto, a robustez e o desempenho não são discutidos.

Uma abordagem baseada em GMM foi proposta por [Drews Jr et al., 2010], onde são recuperadas formas super-quádricas para os dados a fim de classificar as alterações encontradas. Apesar de ter resultados satisfatórios em robustez, a complexidade de tempo de processamento não era adequada para grandes conjuntos de dados. Isto devido, principalmente, à utilização do algoritmo EM para recuperar as GMM. Esse trabalho foi estendido em [Núñez et al., 2010] com uma melhoria em termos de custo computacional e sensibilidade ao número de gaussianas necessário para a representação espacial.

Em nossa abordagem, propomos construir uma função densidade global e efetuar a comparação e detecção de mudanças na cena como simples operação booleana entre duas funções implícitas, como mostramos na Seção 3.4. Esta abordagem permite a

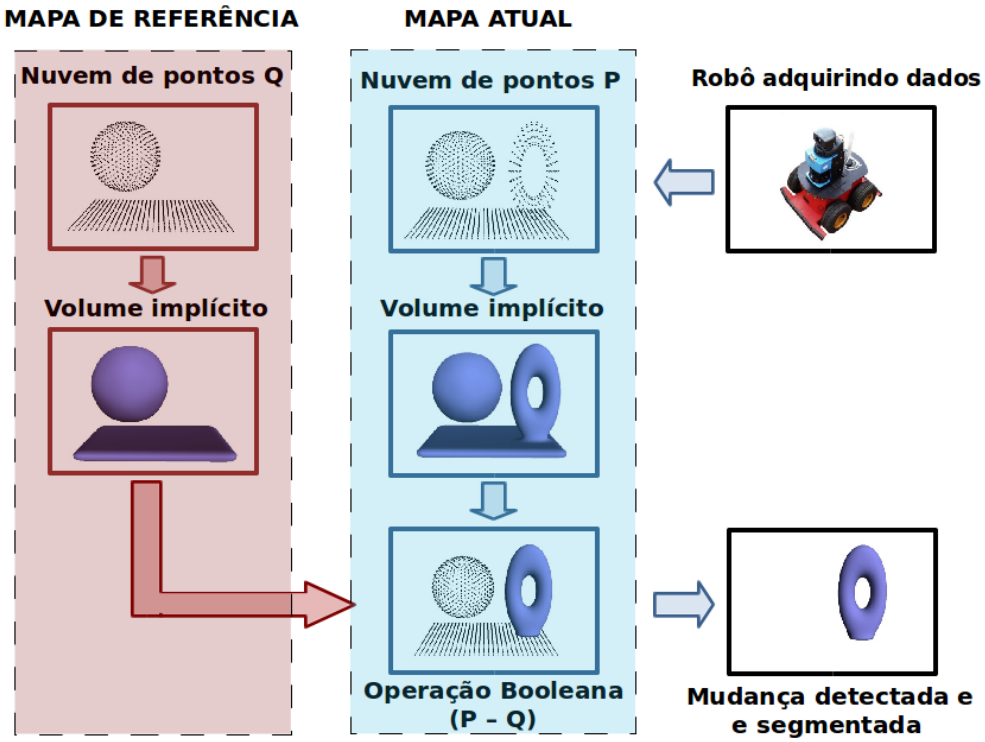


Figura 3.8. Definição do Problema: Dados um mapa $3D$ adquirido por um robô e um mapa previamente conhecido sobre o ambiente, o robô detecta e segmenta as alterações na cena usando agrupamentos implícitos e operações booleanas.

rápida comparação entre duas nuvens de pontos e é eficiente em termos de custo computacional como mostraremos em nossos experimentos. A Figura 3.8 ilustra a abordagem proposta, onde as caixas vermelhas, à esquerda, representam os dados de referência, os dados adquiridos são mostrados nas caixas em azul e os dados de entrada e saída são mostrados nas caixas pretas, à direita.

3.7 Experimentos em detecção de mudanças

Para validação da nossa metodologia para detecção de mudanças em nuvens de pontos, realizamos experimentos em duas bases de dados e apresentamos comparação com outros métodos propostos na literatura. Inicialmente, descrevemos a metodologia de avaliação dos resultados adotada de forma a permitir comparação com a literatura. Em seguida analisamos a robustez e sensibilidade aos parâmetros do nosso método. Finalmente, uma comparação com abordagens no estado-da-arte é apresentada.

3.7.1 Avaliação dos resultados

A fim de obter uma avaliação estatística quantitativa do método proposto, usamos a metodologia proposta por Vieira Neto e Nehmzow [Vieira Neto & Nehmzow, 2008] para avaliar a detecção de mudanças. Eles propõem o uso de tabelas de contingência para relacionar a resposta do sistema com os dados obtendo três diferentes indicadores estatísticos baseados na análise χ^2. Em nosso trabalho, os pontos da base de dados foram segmentados manualmente como sendo mudança, ou não, para efeito de avaliação dos resultados.

Vieira Neto e Nehmzow propõem a utilização de três métricas: Cramer's V, coeficiente de incerteza $U(0 \leq V, U \leq 1)$ e índice κ de concordância. Essas métricas foram utilizadas para quantificar a força de associação de dados, onde os valores menores indicam associações fracas e valores mais próximos de um representam associações mais fortes.

Devido à semelhança entre aplicações dessas métricas, nos nossos resultados mostramos apenas o índice κ a fim de mostrar a precisão do nosso método. O índice κ pode ser calculado utilizando a equação

$$\kappa = \frac{2(TP \times TN - FP \times FN)}{(TP + FN)(FN + TN) + (TP + TN)(TN + FN)}, \tag{3.22}$$

onde:

- TP - Quantidade de verdadeiro positivo
- FP - Quantidade de falso positivo
- TN - Quantidade de verdadeiro negativo
- FN - Quantidade de falso negativo

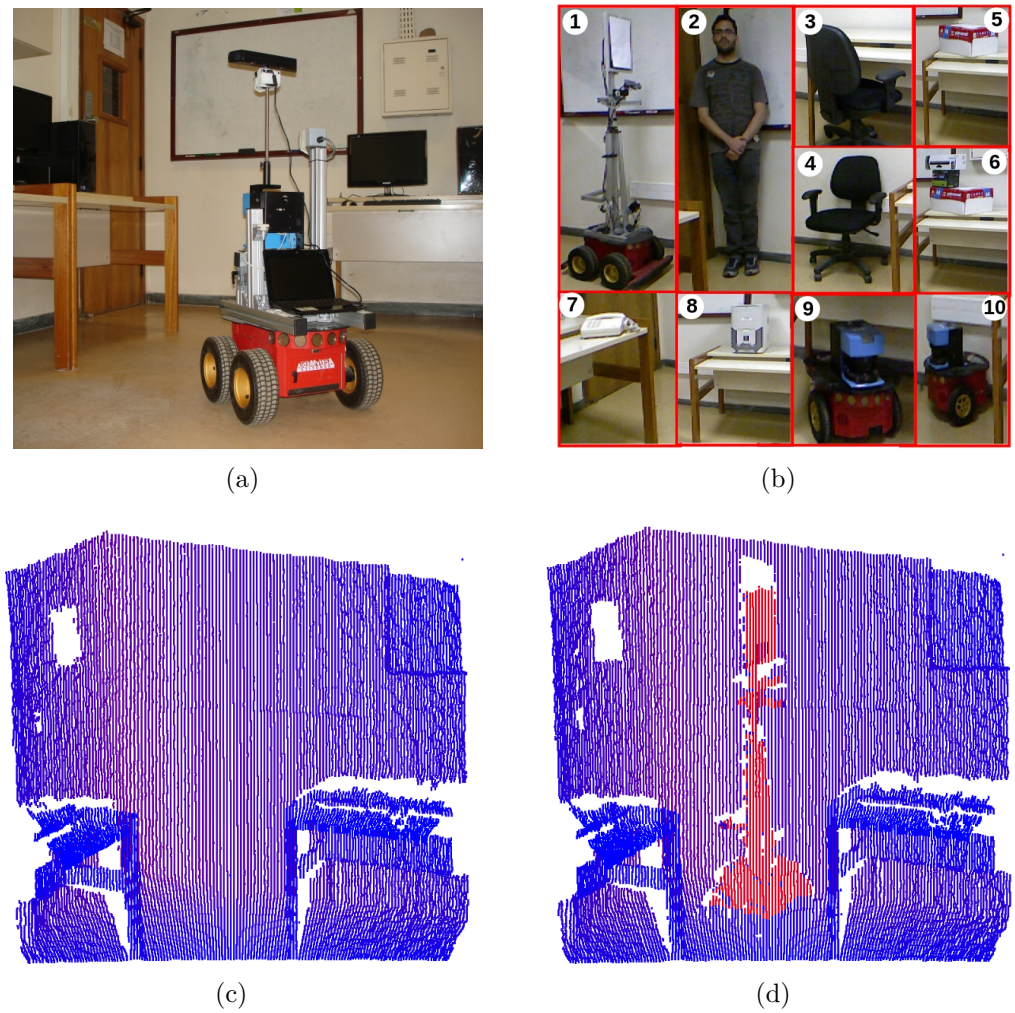
O índice κ varia entre $[-1; 1]$, com valores menores indicando falta de acordo e valores maiores indicando concordância. Mais precisamente:

- $-1 \leq \kappa \leq 0.1 \Rightarrow$ não há acordo
- $0.1 < \kappa \leq 0.4 \Rightarrow$ fraca concordância
- $0.4 < \kappa \leq 0.6 \Rightarrow$ clara concordância
- $0.6 < \kappa \leq 1.0 \Rightarrow$ forte concordância

Mais detalhes sobre estas métricas podem ser encontrados em [Vieira Neto & Nehmzow, 2008].

3.7.2 Robustez e sensibilidade aos parâmetros

Os experimentos para avaliar a robustez do nosso método usam um conjunto de dados $3D$ adquirido em um ambiente de escritório como mapa de referência e 10 conjuntos de dados diferentes do mesmo ambiente modificado pela inserção de objetos de diferentes tamanhos e formas. Um robô Pioneer 2-AT equipado com um sensor *Kinect* foi utilizado para obter os dados e cada conjunto de dados, com cerca de $76k$ pontos, foi amostrado cinco vezes para apresentar resultados estatísticos. O ambiente e as 10 diferentes mudanças são mostrados na Figura 3.9. Os testes foram executados em um PC com um CPU Core 2 Duo rodando a 2.0 GHz e com 2 GB de RAM.

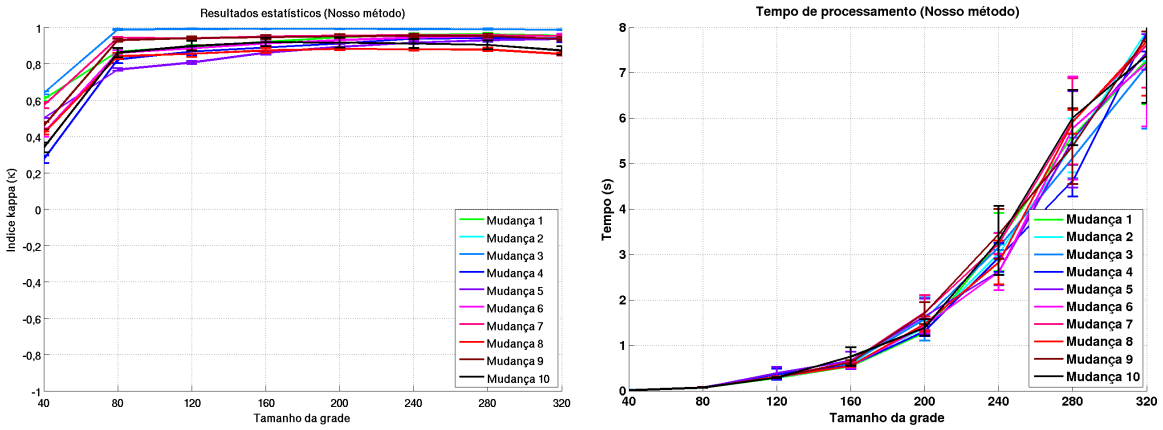


(a) (b)

(c) (d)

Figura 3.9. Configuração experimental 1: (a) Ambiente composto de um robô Pionner equipado com um sensor *Kinect* usado para aquisição de dados $3D$. (b) Dez diferentes mudanças inseridas no ambiente de escritório, com tamanho, pose e forma variados. Cada uma das 10 mudanças foi amostrada 5 vezes. (c) Nuvem de pontos $3D$ de referência do ambiente. (d) Exemplo de nuvem de pontos $3D$ do ambiente com a mudança 1 inserida.

Nosso algoritmo obtém uma amostragem discreta da função de densidade global computando seus valores para uma grade $3D$ definida para a caixa envolvente da nuvem de pontos. Uma grade mais refinada leva a amostragem densa, o que melhora a robustez, mas diminui o desempenho, como mostrado na Figura 3.10. Na Figura 3.10-(a) mostramos os resultados estatísticos usando o índice κ com média e desvio padrão. Esses resultados foram obtidos para a mesma caixa envolvente fixa com uma grade variando o refinamento da amostragem discreta na caixa envolvente, de $40 \times 40 \times 40$ até $320 \times 320 \times 320$, com espaçamento de 40. Os resultados mostram que obtemos alta acurácia a partir da grade de tamanho $120 \times 120 \times 120$ e que esta acurácia é estável para valores maiores. Considerando valores maiores que 0.6 do índice κ como boa detecção, os resultados mostram que podemos detectar mudanças mesmo para pequenos objetos (Figura 3.9-b). Os mesmos experimentos foram realizados supondo remoção de objetos com os mesmos resultados de detecção. Note que, conforme exposto no Algoritmo 1, diferenças associadas à remoção de objetos são computadas pela simples permutação dos conjuntos como parâmetros de entrada do algoritmo.



(a) (b)

Figura 3.10. Resultados variando o tamanho da grade para detecção de mudança com objetos de diversas formas e tamanhos, adquiridos com o dispositivo *Kinect*. (a) Avaliação do índice κ de concordância para diferentes tamanhos da grade. (b) Tempo de processamento para cada conjunto de dados para diferentes tamanhos de grade.

A Figura 3.10-(b) mostra o tempo de processamento do nosso algoritmo. O tempo de processamento é obtido considerando o tempo total de três passos: (1) construir a representação por volumes implícitos para o mapa de referência, (2) construir a representação por volumes implícitos para o mapa atual e (3) efetuar a operação booleana (detecção de mudança). Os resultados são mostrados com tempo médio e desvio padrão. Esses gráficos mostram que aumentando o tamanho da grade aumenta o

tempo de processamento, mas sem ganho significativo em acurácia para grades maiores que $80 \times 80 \times 80$, para o qual alta acurácia é obtida em menos de meio segundo.

A fim de mostrar, experimentalmente, a complexidade linear do nosso método em relação ao número de pontos, computamos mudanças para nuvens com diferentes tamanhos. Para este experimento, nove conjuntos foram usados, para os quais o número de pontos varia de $17k$ a $144k$. Para cada conjunto de pontos, cinco diferentes aquisições foram efetuadas a fim de apresentar média e desvio padrão do tempo de processamento como mostrado na Figura 3.11.

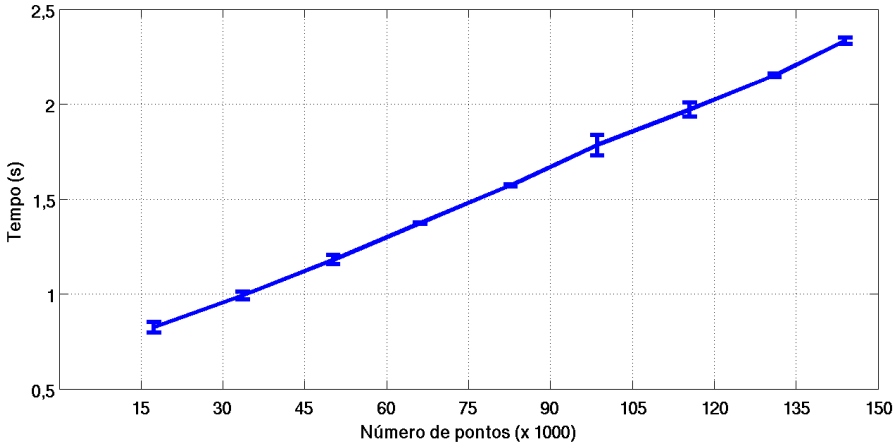


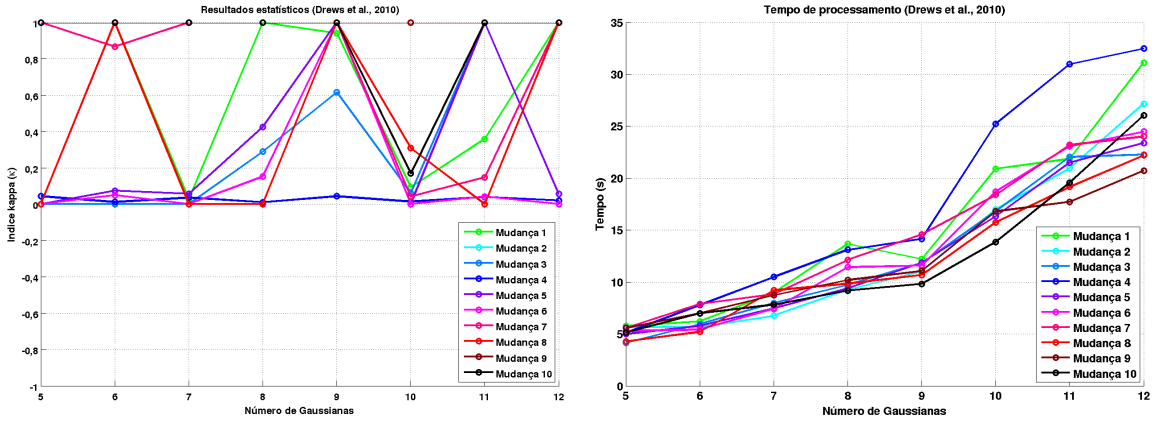
Figura 3.11. Tempo de processamento da nuvem para diferentes quantidades de pontos. Nove conjuntos com diferentes tamanhos variando de $17k$ até $144k$ foram usados para mostrar a complexidade linear em relação ao número de pontos.

Nesse experimento, uma grade fixa particionada em $200 \times 200 \times 200$ foi usada na representação de todos os conjuntos. Também os parâmetros de suavização σ, limiar h e tamanho do *kernel* foram mantidos constantes para todos os conjuntos. Dessa forma, apenas a variação do número de pontos é usada para comparar a variação do tempo de execução do algoritmo.

3.7.3 Comparação com abordagens no estado-da-arte

Usamos o mesmo conjunto de dados para comparar o nosso método com o método apresentado por Drews Jr et al. [Drews Jr et al., 2010] que usa uma combinação de GMM e EMD para detectar as mudanças recuperadas como formas super-quádricas. Enquanto o nosso método tem no tamanho da grade um importante parâmetro que afeta o desempenho e a acurácia, o método proposto em [Drews Jr et al., 2010] é altamente sensível ao número de Gaussianas utilizadas para representar a nuvem de pontos. Assim, apresentamos o desempenho e resultados de acurácia para diferentes

números de gaussianas para efeito de comparação com nosso método. As Figuras 3.12-(a) e 3.12-(b) mostram a acurácia e tempo de processamento para oito diferentes números de Gaussianas usando [Drews Jr et al., 2010]. Notamos que, apesar de o tempo de processamento aumentar com o número de Gaussianas (Figura 3.12-(b)), a acurácia não melhora de forma consistente com o aumento desse parâmetro (Figura 3.12-(a)), alcançando o melhor resultado com nove distribuições Gaussianas. Além disso, enquanto o nosso método atinge alta precisão em menos de meio segundo, o outro método gasta mais de 10 segundos para obter sua melhor precisão com nove Gaussianas.



(a) (b)

Figura 3.12. Resultados variando o número de Gaussianas para detecção de mudança com objetos de diversas formas e tamanhos, adquiridos com o dispositivo *Kinect*. (a) Avaliação do índice κ de concordância para diferentes números de Gaussianas. (b) Tempo de processamento de cada conjunto de dados para diferentes números de Gaussianas.

Comparamos, ainda, o nosso método com o método apresentado em [Núñez et al., 2010] usando o conjunto de dados proposto por eles. Esse conjunto de dados foi adquirido usando um robô Pioneer 2-AT equipado com dois *scanners* SICK LMS-200 montados ortogonalmente, a fim de localizar e adquirir mapas 3D. Um PC rodando a 1.66GHz e com 1GB de RAM foi usado para a detecção de mudança. Para os experimentos, três diferentes mudanças foram inseridas no espaço de trabalho do robô: um cilindro, uma pessoa e uma caixa de impressora. O ambiente e os objetos utilizados são mostrados na Figura 3.13-(a) e 3.13-(b).

Assim como em [Núñez et al., 2010], podemos detectar mudanças em todos os três cenários, porém mais rápido. A Tabela 3.1 mostra os melhores tempos (em segundos) obtidos em [Núñez et al., 2010]. Ambos os métodos são divididos em dois passos, onde o primeiro passo constrói uma representação "especial" e o segundo passo, chamado

(a) (b)

Figura 3.13. Configuração experimental 2: (a) Ambiente experimental composto de um robô Pioneer 2-AT equipado com dois *scanners* a laser montados ortogonalmente para aquisição de dados $3D$. (b) Três diferentes mudanças inseridas no ambiente do robô.

detecção, compara essa representação. No caso de [Núñez et al., 2010], a representação é construída usando um método de simplificação e estimativas de GMM. Em nosso método, a representação é adaptada para a construção da representação por densidade de ocupação espacial. A detecção de alterações é feita usando o casamento estrutural descrito em [Núñez et al., 2010] e operações booleanas em nossa abordagem. Percebe-se que a construção da representação para as nuvens de pontos é a etapa de maior consumo de tempo de processamento. Entretanto, o cálculo de GMMs em [Núñez et al., 2010] torna essa etapa altamente ineficiente consumindo até 100 vezes mais tempo do que a nossa representação.

	N^o de pontos	Construção da representação		Detecção de mudança	
Conjunto	Mapa ref. & atual	Nuñez et al.	Nosso	Nuñez et al.	Nosso
Mudança 1	158804	$341,77$	$\mathbf{2,19}$	$\mathbf{0,014}$	$0,146$
Mudança 2	160365	$288,22$	$\mathbf{2,21}$	$\mathbf{0,014}$	$0,149$
Mudança 3	159283	$277,10$	$\mathbf{2,20}$	$\mathbf{0,028}$	$0,166$

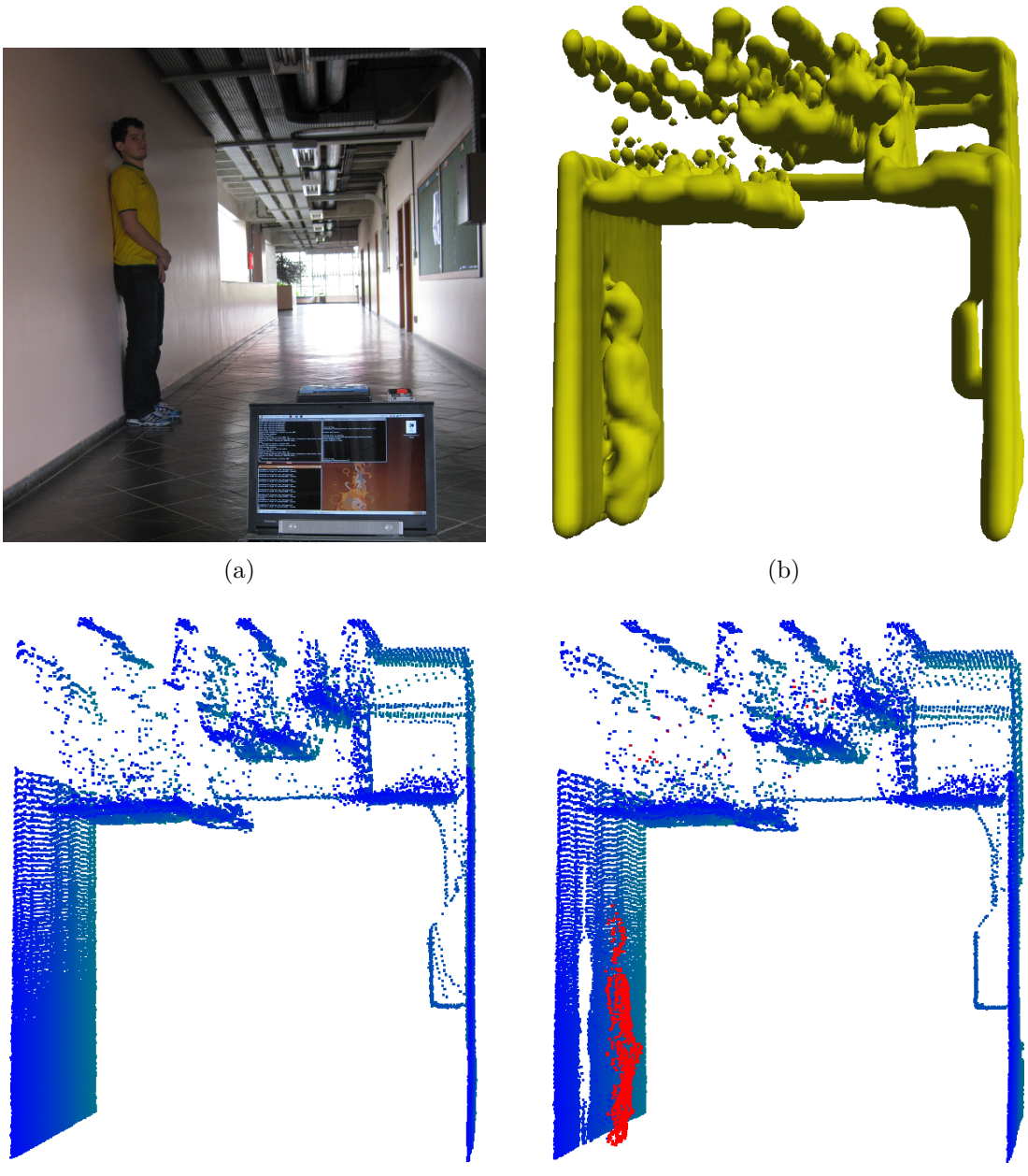
Tabela 3.1. Estudo comparativo do desempenho entre diferentes algoritmos de detecção de mudança. Os tempos para construção da representação e detecção de mudança são mostrados em segundos.

Note que, para todos os conjuntos de dados, o nosso método é substancialmente mais rápido para construir a representação requerida. Enquanto a construção da representação por GMM, em [Núñez et al., 2010], leva mais de 200 segundos, a nossa

representação por volumes implícitos é construída em cerca de dois segundos apenas. A principal limitação do método proposto em [Núñez et al., 2010] está relacionada com os modelos de misturas gaussianas que utiliza o algoritmo EM, com menor desempenho computacional e é bastante sensível ao número de gaussianas, que é um dos parâmetros do algoritmo EM. Na detecção da mudança (última coluna da tabela), vemos que a comparação de um número limitado de GMMs como em [Núñez et al., 2010] é mais rápida que a nossa comparação por operações booleanas. Este ganho na etapa de comparação se deve à utilização do algoritmo de casamento estrutural que permite comparação eficiente entre GMMs. Entretanto, essa etapa tem peso irrelevante frente ao alto custo computacional da etapa de construção da representação, onde reduzimos consideravelmente o tempo.

Além da melhora no tempo de processamento computacional, observamos, empiricamente, a partir de outros resultados como mostrado na Figura 3.14, que o nosso método atua de forma robusta mesmo em dados com geometria muito complexa. No trabalho de [Núñez et al., 2010], os conjuntos de dados obtidos pelo robô mostram as paredes e alguns objetos, mas o teto também foi adquirido, onde os tubos e outros objetos estão presentes no teto, como mostrado na Figura 3.13-(a). No entanto, esses dados não são considerados no seu experimento devido à alta complexidade dos dados e à ausência de alterações significativas. Em outro experimento, utilizamos o *dataset* completo a partir desse conjunto para descrever os dados, incluindo todos os pontos.

A Figura 3.14 mostra um exemplo de um conjunto de dados como usado em [Núñez et al., 2010]. Na Figura 3.14-(a), vê-se uma pessoa inserida no ambiente. Na Figura 3.14-(b) o volume implícito obtido, na Figura 3.14-(c) a nuvem de pontos do ambiente de referência sem alteração e, na Figura 3.14-(d), o mapa atual com a mudança segmentada. Os pontos azuis representam os pontos detectados como não mudança, e os pontos vermelhos como pontos de mudança detectada pelo nosso algoritmo. Alguns pontos em vermelho estão no teto, o que se deve ao efeito de oclusão na aquisição e ruído.



(a) (b)

(c) (d)

Figura 3.14. Detalhes da Configuração Experimental 2: (a) Uma pessoa é a mudança no ambiente. (b) Volume implícito obtido para a nuvem de pontos. (c) Nuvem de pontos $3D$ do ambiente de referência. (d) Nuvem de pontos $3D$ do ambiente com a mudança.

Capítulo 4

Densidade espacial como histograma saturado

Neste capítulo, um histograma saturado de contagem de pontos por células é usado como representação da densidade espacial e dois métodos de classificação são propostos para comparação de resultados usando uma base de dados pública. Esta aplicação trata do reconhecimento de ações humanas em sequências de mapas de profundidade que é um tópico que tem ganhado relevância com a popularização dos sensores $3D$ de tempo real como o Kinect.

A representação por histograma saturado é, em parte, inspirada na conhecida grade de ocupação que é comumente utilizada para navegação de robôs [Elfes, 1989]. Um volume $2D$ ou $3D$ do espaço é particionado em uma grade onde a cada célula está associada uma probabilidade indicando a certeza da sua ocupação. Da mesma forma, em nossa abordagem, dada uma sequência de mapas profundidade, consideramos a sequência como um conjunto

$$\mathcal{A} = \{(x_i, y_i, z_i, t_i), i = 1 \dots N\} \tag{4.1}$$

em um volume espaço temporal em que a quarta coordenada t_i indica o índice do quadro na sequência de mapas de profundidade que especifica o tempo. A Figura 4.1 ilustra uma nuvem de pontos $4D$ em suas secções tridimensionais em diversos quadros ao longo do eixo tempo.

Esse volume espaço temporal é então dividido em uma grade 4-dimensional. Considere x, y, z e t denotando os quatro eixos, $\Omega \subset \mathbb{R}^4$ denotando o volume espaço temporal e denote por n_x, n_y, n_z e n_t o número de segmentos divididos uniformemente ao longo dos eixos x, y, z e t, respectivamente. Então Ω é dividido em

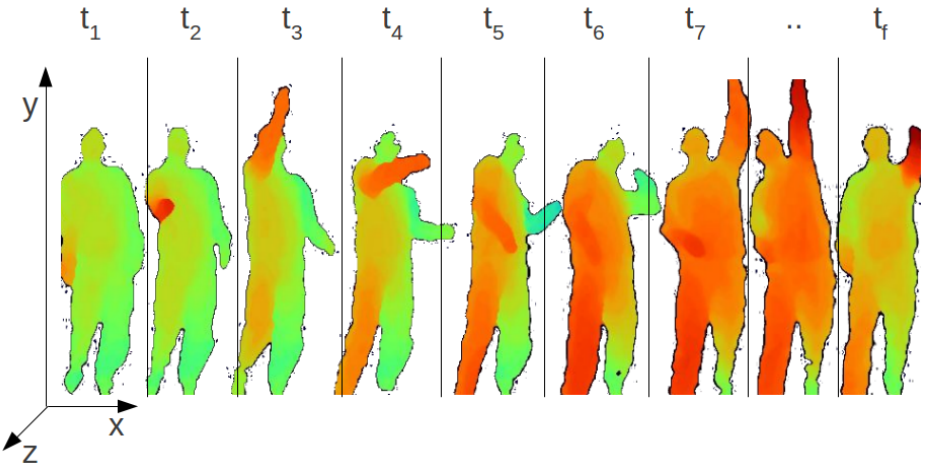


Figura 4.1. Ilusração de nuvem de pontos $4D$ em secções tridimensionais ao longo do tempo. A cada ponto é associado suas coordenadas tridimensionais (x, y, z) e sua coordenada temporal t, que indica índice do quadro na sequência.

$m = n_x \times n_y \times n_z \times n_t$ células $4D$. Cada célula pode interceptar diversos quadros na sequência. Usamos c_i para designar a i-ésima célula $i = 1, ..., m$. O conjunto de células é chamado de *Partição*, denotada como $C = \{c_1, ..., c_m\}$.

Para cada célula c_i, denotamos por $\mathcal{A}_i$ sua intersecção com o conjunto $\mathcal{A}$ de pontos 4-dimensionais, isto é, $\mathcal{A}_i = \mathcal{A} \cap c_i$. O valor de ocupação de c_i é definido como

$$P(c_i) = \begin{cases} 1, & \text{se} \quad |\mathcal{A}_i| \geq q \\ \frac{|\mathcal{A}_i|}{q} & \text{senão} \end{cases} \tag{4.2}$$

onde q é um parâmetro de saturação pré-definido e $|\mathcal{A}_i|$ indica a cardinalidade de $\mathcal{A}_i$. Para cada célula que contém q ou mais pontos, o seu valor de ocupação é definido como o valor máximo 1. A razão para este esquema de saturação é a seguinte. O número de pontos em uma célula não vazia pode ser tão pequeno quanto um, e tão grande quanto milhares. Para uma sequência típica, a maioria das células não-vazias contém apenas algumas centenas de pontos. Por um lado, se o histograma é usado diretamente, sem saturação, as células que contêm um pequeno número de pontos não teriam qualquer papel significativo na representação porque os seus valores relativos são demasiado pequenos. Por outro lado, as células que contêm um pequeno número de pontos são realmente muito importantes para o reconhecimento de padrões. Para definir automaticamente o parâmetro q, obtemos um histograma normalizado de densidade de ocupação das células, onde 1 é a densidade máxima, e tomamos q como o número de pontos associado à densidade $0,01$.

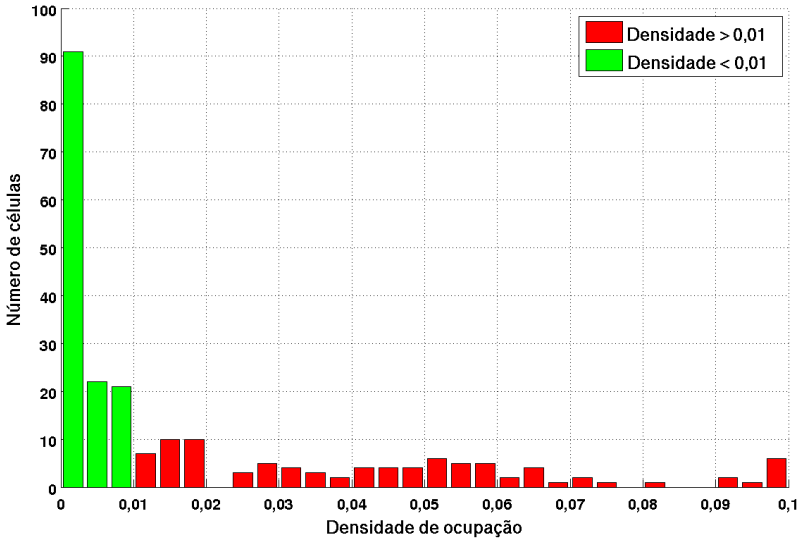


Figura 4.2. Ilusração da distribuição de densidade por células espaço-temporais. Um grande número de células (em verde) acumula pouca densidade, enquanto que um pequeno número de células (em vermelho) acumula alta densidade.

A Figura 4.2 ilustra a distribuição de densidade por células espaço-temporais. Neste experimento, usamos sequências de mapas de profundidade descrevendo ações humanas [Li et al., 2010] para construir o histograma de contagem de pontos por células. Note que um grande número de células (em verde) acumula pouca densidade, enquanto que um pequeno número de células (em vermelho) acumula alta densidade. Células com baixa densidade estão associadas a bordas das células e partes móveis na sequência, enquanto células com alta densidade estão associadas a partes estáticas. Na Figura 4.3, pontos em verde nas células que interceptam os braços indicam movimento e nas células que interceptam os pés indicam bordas.

Na Seção 4.3 mostramos, experimentalmente, a influência do valor de saturação q no reconhecimento de ações em sequências de mapas de profundidade. Para construir o histograma para uma determinada sequência de mapas de profundidade $\mathcal{A}$ e partição C, denotamos

$$H\left(\mathcal{A}, C\right) = \left(P\left(c_1\right), P\left(c_2\right), \ldots, P\left(c_m\right)\right)^T. \tag{4.3}$$

Então, $H(\mathcal{A}, C)$ é um vetor m-dimensional, o qual denominamos Padrão de Ocupação Espaço Temporal (STOP) de $\mathcal{A}$ com respeito à partição C. $H(\mathcal{A}, C)$ é a descrição visual de $\mathcal{A}$. A Figura 4.3 ilustra o particionamento de um volume obtido pela acumulação de 20 quadros de uma sequência de mapas de profundidade. A caixa

maior representa a caixa envolvente e as caixas menores são as células espaciais. Apenas células ocupadas são mostradas.

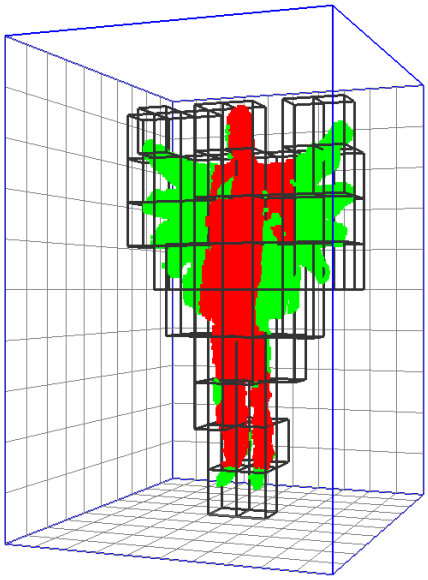


Figura 4.3. Células espaciais de um volume obtido de uma sequência de 20 mapas de profundidade. A caixa grande é a caixa envolvente onde são destacadas apenas as células não-vazias

4.1 Padrões de ocupação espaço temporal

O histograma saturado de contagem de pontos é usado como descritor para reconhecimento de padrões conforme propomos na Seção 4.3, onde usamos diferentes valores de saturação q de acordo com o número de quadros por segmento de tempo na classificação de ações humanas em sequências de mapas de profundidade. Percebemos que a precisão da classificação é estável para valores pequenos de q, mas muito sensível para grandes valores de q, para o qual a precisão diminui. Nossos experimentos mostram esse efeito, apresentando a precisão de reconhecimento para diferentes valores do parâmetro de saturação q.

Em geral, um vetor de características STOP é bastante esparso, ou seja, a maioria dos seus elementos são nulos. Isso nos motivou a realizar uma redução de dimensionalidade, como descrito na próxima seção.

4.2 Redução de dimensionalidade

Usamos um método de redução de dimensionalidade baseado em SVD, que tira proveito da alta dimensionalidade dos vetores descritores quando comparado ao número de exemplos disponíveis, ou seja, temos um conjunto com n descritores m-dimensionais onde $m \gg n$. Neste caso, uma base com apenas n componentes é usada para representar os novos descritores. Além disso obtemos uma nova base tal que, nesta base, classes distintas definem subespaços ortogonais.

Considere, inicialmente, que temos r classes distintas, cada uma representada por k vetores m-dimensionais de forma que temos $n = r \times k$ vetores descritores. Sejam $f_1, f_2, \ldots, f_n$ os vetores descritores de todos os dados de treinamento e seja $\mathbf{F}_{m \times n}$ a matriz onde as colunas são formadas pelo vetores descritores, ou seja,

$$\mathbf{F} = (f_1, f_2, ..., f_n). \tag{4.4}$$

Usando a decomposição SVD, temos $\mathbf{F} = \mathbf{USV}^T$, onde $\mathbf{U}_{m \times m}$ e $\mathbf{V}_{n \times n}$ são matrizes quadradas ortonormais e $\mathbf{S}$ é diagonal. Com a ordenação dos autovalores pode-se optar pela representação usando apenas as primeiras componentes principais para redução de dimensão. Em vez de formar a nova base a partir de colunas de $\mathbf{U}$ como no PCA usual, usamos o fato de que

$$\mathbf{V}^T = (\mathbf{S}^T\mathbf{S})^{-1}\mathbf{S}^T\mathbf{U}^T\mathbf{F}, \tag{4.5}$$

e formamos a nova base $\mathbf{B} = (\mathbf{S}^T\mathbf{S})^{-1}\mathbf{S}^T\mathbf{U}^T$ tal que, nesta nova base, todos os novos vetores descritores são colunas da matriz $\mathbf{V}$ (portanto de dimensão $n \ll m$) sendo vetores unitários e dois a dois ortogonais. Note que a operação $(\mathbf{S}^T\mathbf{S})^{-1}\mathbf{S}^T$ é elementar pois envolve apenas a matriz diagonal $\mathbf{S}$. A vantagem desta nova representação é que, dadas as classes distintas i e j e seus vetores de descritores como matrizes $\mathbf{F}_i$ e $\mathbf{F}_j$ formados por colunas de $\mathbf{F}$, obtemos novas matrizes $\mathbf{C}_i = \mathbf{B}.\mathbf{F}_i$ e $\mathbf{C}_j = \mathbf{B}.\mathbf{F}_j$, de modo que as colunas de $\mathbf{C}_i$ e $\mathbf{C}_j$ são vetores dois a dois ortogonais, permitindo a classificação por projeção ortogonal. Finalmente, nós usamos os fatos acima para construir o nosso classificador.

Dada a matriz $\mathbf{F} = (f_1, f_2, ..., f_n)$ cujas colunas são descritores STOP dos dados de treinamento, independentemente dos rótulos de sua classe, efetuamos a decomposição $\mathbf{F} = \mathbf{USV}^T$, denotamos a matriz $\mathbf{V}$ por $\mathbf{V} = (e_1, e_2, ..., e_n)$ e tomamos as colunas de $\mathbf{V}$ como novos descritores, denotados por e_i.

Cada descritor e_i é um vetor de menor dimensão para uma sequência de mapas de profundidade. Chamamos os vetores e_i de descritores PCA-STOP. Esses são os

vetores de características utilizados na etapa de classificação conforme apresentaremos na Seção 4.3. Nesta representação, cada classe define subespaços independentes e complementares do espaço $\mathbb{R}^n$, de forma que a norma da projeção de um vetor de testes e em cada subespaço pode ser usada como indicativo da classe a que e pertence. Esta classificação por aprendizagem de classes ortogonais (OCL, do inglês *Orthogonal Class Learning*) foi aplicada com sucesso para reconhecimento *offline* de objetos em imagens [Oliveira et al., 2012].

Uma aplicação para esta representação é discutida na Seção 4.3, sobre reconhecimento de ações humanas em sequências de mapas de profundidade, onde uma representação por GMMs é substituída por densidade de ocupação espacial nos descritores PCA-STOP.

4.3 Aplicação em reconhecimento de ações

Os principais algoritmos de reconhecimento de ações humanas são baseados em sequências de vídeo e poucos trabalhos em reconhecimento de ação exploram nuvens de pontos. Uma revisão sobre o reconhecimento de ações é apresentada em [Weinland et al., 2010], onde é proposta uma classificação da representação em local, global e paramétrica, onde os métodos, em sua maioria, são baseados em vídeos e poucos trabalhos se baseiam em mapas de profundidade ou nuvens de pontos. Em Robótica, por exemplo, mapas de profundidade são utilizados principalmente para localização, mapeamento e reconstrução de ambiente e apenas um número limitado de trabalhos explora o reconhecimento de ações humanas utilizando mapas de profundidade.

Com a popularização dos sensores de profundidade que utilizam padrões de luz estruturada [Geng, 2011], trabalhos para reconhecimento de ações em sequências de mapas de profundidade têm sido propostos como em [Reyes et al., 2011]. Recentemente, a Microsoft lançou uma câmera de profundidade, chamada *Kinect*, que usa luzes estruturadas invisíveis para produzir um mapa de profundidade da cena. Dispositivos como o *Kinect* disponibilizam, em tempo real, dados RGB-D e tem despertado o interesse da comunidade de visão computacional para a utilização de mapas de profundidade em pesquisas de reconhecimento de objetos, reconstrução tridimensional, bem como reconhecimento de poses e ações humanas.

Um grande avanço na utilização de mapas de profundidade para o reconhecimento de ações é a recuperação de poses $3D$ em tempo real. Shotton et al. [Shotton et al., 2011] apresentam um método que recupera rapidamente a posição 3D de diversas articulações do corpo usando uma única imagem de profundidade e nenhuma

informação temporal. Esses resultados permitem projetar esqueletos articulados nos mapas de profundidade de forma que um conjunto de coordenadas associadas a articulações do corpo são disponibilizadas por diversos *drivers* como o KinectSDK e o OpenNI.

Um trabalho que usa mapas de profundidade em tempo real para o reconhecimento de ações é proposto por Li et al. [Li et al., 2010]. Eles desenvolveram uma técnica de reconhecimento de ações a partir de mapas de profundidade capturada por uma câmera de profundidade semelhante ao *Kinect*. Eles capturaram um conjunto de dados com várias pessoas realizando ações diferentes. A representação das nuvens de pontos, para caracterizar as poses, é baseada na definição de *poses salientes* que são modeladas como GMMs para capturar a distribuição estatística dos pontos. Para cada mapa de profundidade em uma sequência, o método proposto por eles amostra um pequeno conjunto de pontos $3D$ que são os pontos de interesse. A amostragem é realizada usando as silhuetas projetadas sobre os três planos coordenados conforme ilustrado na Figura 4.4 extraída de [Li et al., 2010].

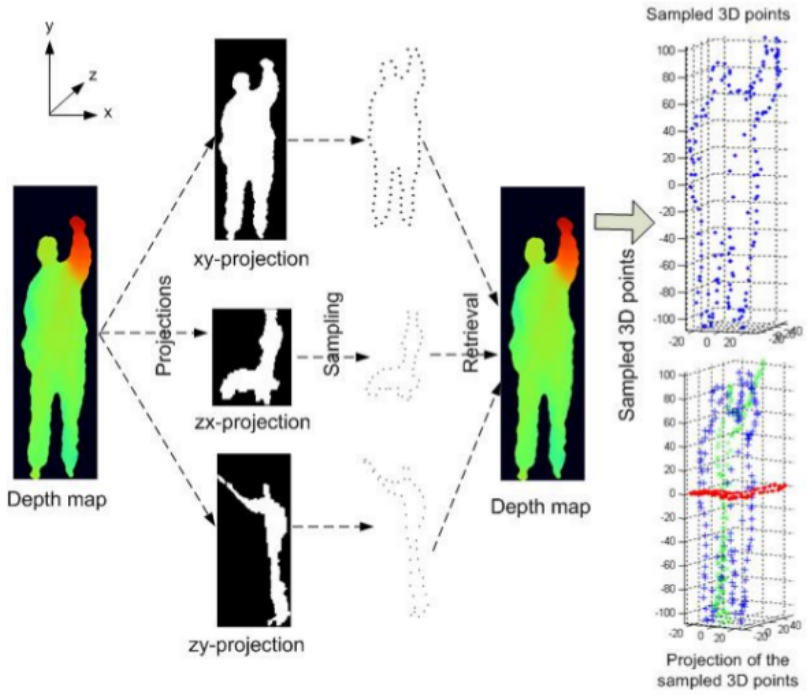


Figura 4.4. Ilustração da amostragem da nuvem de pontos por silhueta projetadas sobre os planos coordenados para construção de *bag of points* conforme [Li et al., 2010]

Para cada ponto de interesse, eles usam as coordenadas $3D$ como descritor e um *Bag of Features* desses pontos é a representação visual do quadro. A dissimilaridade

entre dois mapas de profundidade é calculada pela distância de Hausdorff entre os dois conjuntos de pontos de interesse. Uma limitação desta abordagem é que o contexto espacial entre os pontos de interesse é perdido. Além disso, devido a ruído e oclusão nos mapas de profundidade, as silhuetas vistas de lado e de cima para baixo não são muito confiáveis. Isto torna muito difícil uma amostragem robusta de pontos de interesse, dada a complexa geometria e as variações entre as diferentes pessoas. Provavelmente por isso, a precisão de reconhecimento para os testes *cross-subject* é muito inferior à obtida nos demais testes.

O uso de GMMs para caracterizar as poses em nuvens de pontos e distância de Hausdorff como medida de dissimilaridade entre poses impõe alta complexidade ao método, o que inviabiliza a classificação de grandes bases de dados, como relatado nos experimentos em [Li et al., 2010].

Nesta aplicação, abordamos o reconhecimento de ações humanas a partir de mapas de profundidade e propomos a utilização da representação por densidade de ocupação espacial para construir descritores de ação em substituição às GMMs e a classificação usando OCL conforme apresentado na seção refsec:stop. Para o reconhecimento de ações, em nosso trabalho, propomos uma classificação *offline*, onde as sequências são previamente segmentadas, e uma classificação *online*, onde a segmentação de ações é computada automaticamente. Em ambos os casos, utilizamos sequências de informações tridimensionais disponibilizadas na forma de mapas de profundidade como em [Li et al., 2010].

4.3.1 Classificação *offline*

Na classificação *offline*, experimentamos diferentes particionamentos para construção do descritor STOP, e apresentamos os melhores resultados para essa base de dados, com $n_x = n_y = n_z = 10$ e $n_t = 3$, onde os quadros de uma sequência são acumulados em três janelas de tempo. A escolha de três janelas de tempo se deve ao fato de que podemos considerar que cada sequência de ação consiste em três fases: fase inicial (não muito movimento), fase intermediária (realizando a ação) e fase final (não muito movimento). O segmento médio (fase intermediária) é o mais importante na classificação. Dessa forma, o vetor descritor STOP H é dado pela concatenação dos três descritores associados a cada segmento temporal de modo que a dimensão de $H(A, C)$ é 3000. A Figura 4.5 ilustra as células espaço temporais de uma sequência de mapas de profundidade da ação *chutar*. A sequência é dividida em três segmentos de tempo e cada segmento é composto por cerca de 20 quadros. Somente as células não-vazias são mostradas. Os pontos verdes são aqueles em que as células contêm até q

pontos. Podemos ver que essas células contêm pontos sobre as silhuetas ou interceptam partes móveis do corpo. Quando uma célula intercepta uma parte móvel do corpo, essa intersecção normalmente dura apenas por um pequeno número de quadros, assim, o número de pontos da célula tende a ser pequeno. Naturalmente, a velocidade do movimento é determinante para a quantidade de pontos que interceptam a célula.

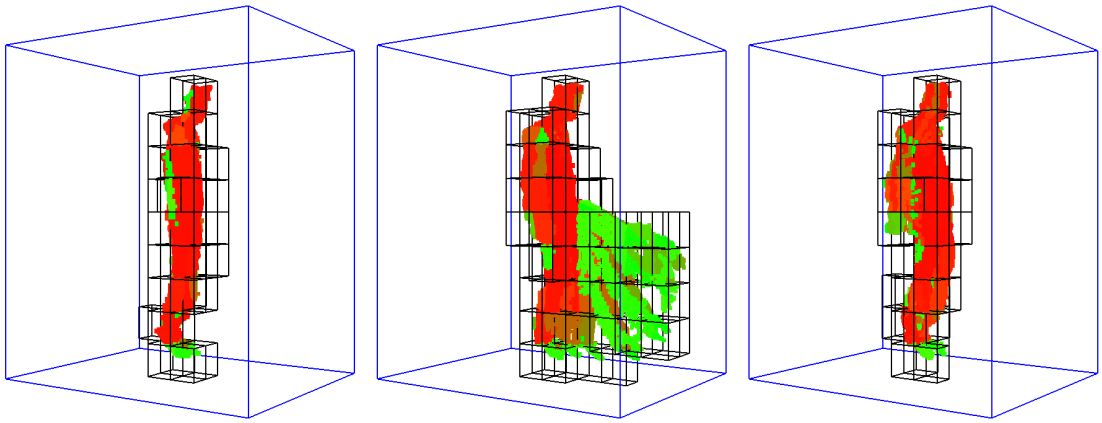


Figura 4.5. Exemplo de sequência de mapas de profundidade dividida em três segmentos de tempo. A caixa grande é a caixa envolvente e as caixas pequenas ilustram as células não vazias.

Uma vez que a ênfase principal desta parte do trabalho é a representação visual, usamos nosso classificador com base na distância do cosseno entre vetores PCA-STOP obtidos pelo método OCL na seção 4.2. Nossos experimentos mostram que, mesmo com um classificador simples, a eficácia do descritor permite uma alta taxa de reconhecimento.

Seja L o número de classes de ação. Seja ainda E_i o conjunto de todos os vetores PCA-STOP dos dados de treinamento da classe de ação i, $i = 1, ..., L$. Dado qualquer descritor PCA-STOP e a ser classificado, a distância de e para a classe de ação i é dada por

$$D_i(e) = 1 - \frac{1}{|E_i|} \sum_{\hat{e} \in E_i} \frac{< e, \hat{e} >}{\|e\|\|\hat{e}\|}, \tag{4.6}$$

onde $|E_i|$ é o número de exemplos de ações de treinamento da classe i.

Note que os descritores obtidos com OCL para os dados de treinamento definem bases ortonormais E_i de subespaços definidos pelas diferentes classes de ação. Dessa forma, a distância do cosseno como proposto permite escolher a ação i, em cuja base E_i, um vetor de teste e tem maior projeção. Note que uma característica dos descritores obtidos com OCL é a normalização dos vetores para norma 1, o que inviabiliza a comparação pela distância Euclidiana. Então, a escolha da distância do cosseno

em detrimento da distância Euclidiana se deve à forma construtiva dos descritores. Em nossos experimentos, comparamos a classificação obtida com o classificador OCL usando a distância do cosseno em relação à classificação obtida com classificadores clássicos. Experimentos usando distância Euclidiana e descritores obtidos com PCA usual mostram que, para bases onde a dimensão do espaço é superior ao número de exemplos, a distância do cosseno em descritores OCL leva a melhor classificação.

Para validar nossa classificação *offline*, realizamos experimentos utilizando a base de dados *MSR Action3D Dataset* [Li et al., 2010] conforme ilustrado na Figura 4.6.

Figura 4.6. Ilustração, conforme [Li et al., 2010], de sequências de mapas de profundidade usadas em nossos experimentos.

O conjunto de dados está disponível em [Liu, 2011]. Esses dados foram gravados usando uma câmera de profundidade semelhante ao dispositivo *Kinect*. Estão disponíveis sequências de mapas de profundidade de 20 tipos de ação, realizadas por 10 pessoas diferentes. As ações são todas curtas e não repetitivas. Cada indivíduo executa cada ação duas a três vezes. No total, existem 567 sequências de mapas de profundidade. Cada sequência é de cerca de três a quatro segundos de duração. A taxa de quadros é de 15 fps. A resolução dos mapas de profundidade é 320×240.

Para efeito de experimentos, conforme em [Li et al., 2010], os 20 tipos de ação são divididos em três subconjuntos AS1, AS2 e AS3, cada um com oito tipos de ação, conforme listado na Tabela 4.1 (os três subconjuntos se sobrepõem).

Para cada subconjunto de tipos de ação, três testes foram realizados. No Teste I, para cada tipo de ação e de cada indivíduo, $\frac{1}{3}$ das sequências foi usado na fase de treinamento, enquanto que o restante foi utilizado em testes. No Teste II, para cada tipo de ação e de cada indivíduo, $\frac{2}{3}$ das sequências foram usados para treinamento, enquanto as restantes foram usadas para testes. O Teste III é um teste cruzado (*cross-subject*) para o qual as sequências executadas pela metade dos indivíduos foram utilizadas para treinamento, enquanto as sequências realizadas pela outra metade dos indivíduos foram utilizadas para teste. Em outras palavras, os indivíduos nos dados de teste não aparecem nos dados de treinamento.

AS1	AS2	AS3
Horizontal arm wave	High arm wave	High throw
Hammer	Hand catch	Forward kick
Forward punch	Draw x	Side kick
High throw	Draw tick	Jogging
Hand clap	Draw circle	Tennis swing
Bend	Two hand wave	Tennis serve
Tennis serve	Forward kick	Golf swing
Pickup & throw	Side boxing	Pickup & throw

Tabela 4.1. Subconjuntos de ações utilizados nos experimentos.

Os resultados de taxa de reconhecimento obtidos por nosso método e por Li et al. sobre os testes I, II e III são mostrados nas Tabelas 4.2, 4.3 e 4.4, respectivamente, onde comparamos também nossos resultados com um método de classificação de ações baseado em esqueletos.

Conjunto	Li et al.	Esqueletos	STOP
AS1	89,50	68,00	98,23
AS2	89,00	73,86	94,82
AS3	96,30	78,67	97,35
Avg	91,36	73,51	96,80

Tabela 4.2. Comparação das taxas de reconhecimento no Teste I

Conjunto	Li et al.	Esqueletos	STOP
AS1	93,30	72,97	99,12
AS2	92,90	70,67	96,95
AS3	96,30	83,78	98,67
Avg	94,20	75,81	98,25

Tabela 4.3. Comparação das taxas de reconhecimento no Teste II

Conjunto	Li et al.	Esqueletos	STOP
AS1	72,90	40,28	84,70
AS2	71,90	50,00	81,30
AS3	79,20	73,91	88,40
Avg	74,70	54,73	84,80

Tabela 4.4. Comparação das taxas de reconhecimento no Teste III

Para mostrar que os descritores PCA-STOP são mais discriminativos para o reconhecimento de ações que os esqueletos obtidos a partir de mapas de profundidade,

usamos esqueletos recuperados da mesma base de dados para apresentar uma comparação de acurácia no reconhecimento. Esses esqueletos são obtidos a partir de mapas de profundidade usando o trabalho de Shottton et al. [Shotton et al., 2011]. Os melhores resultados usando esqueletos foram obtidos usando coeficientes da Transformada Rápida de Fourier (FFT) como descritores. Para cada junta j, $1 \leq j \leq 20$, do esqueleto, tomamos suas coordenadas (x, y, z) e construímos curvas $x(t)$, $y(t)$ and $z(t)$, com t indicando o tempo. Então, computamos a FFT em cada curva para construir descritores usando a magnitude das 10 primeiras frequências, de forma que obtemos um descritor com $20 \times 3 \times 10 = 600$ dimensões. Nas tabelas 4.2, 4.3 e 4.4, nas segundas colunas, mostramos os resultados para cada teste usando o classificador baseado em esqueleto para comparação com nossos resultados. O trabalho recente apresentado em [Raptis et al., 2011] sobre reconhecimento de passos de dança em sequências de esqueletos relata alta acurácia. Entretanto, fortes suposições sobre alinhamento temporal pelo ritmo musical e participação de dançarinos profissionais limitam consideravelmente a generalização do método proposto.

Podemos ver que o nosso método superou o método anterior [Li et al., 2010], que usa *Bag of 3D Points* a partir de silhuetas, em todos os casos. Uma limitação do uso de *Bag of 3D Points* e silhuetas é o fato de que a informação de contexto espacial entre os pontos de interesse é perdida. Além disso, devido a ruído e oclusões dos mapas de profundidade, as silhuetas vistas de lado e de cima para baixo são imprecisas. Os descritores STOP, por outro lado, preservam a informação de contexto espacial e são mais robustos a ruído. No Teste I, a taxa de reconhecimento médio aumentou de 91.36% para 96.80%. No Teste II, aumentamos a taxa de reconhecimento médio de 94.20% para 98.25%. No caso mais difícil, *cross-subject*, melhoramos a taxa de reconhecimento de $74,70\%$ para $84,80\%$.

Para demonstrar o efeito do parâmetro de saturação q, a Figura 4.7 mostra as taxas de reconhecimento do descritor PCA-STOP para o teste *cross-subject* com diferentes valores do parâmetro de saturação. O eixo horizontal mostra os parâmetros de saturação (valores em potência de 2), e o eixo vertical mostra a taxa de reconhecimento.

Neste experimento, usamos a mesma base de dados [Li et al., 2010], sendo que a metade dos indivíduos foi usada como treinamento e o restante como teste. Podemos ver que a taxa de reconhecimento é bastante estável quando o parâmetro de saturação é menor do que 300. O desempenho diminui à medida que o parâmetro de saturação fica maior. A razão para essa perda de acurácia com o aumento do parâmetro de saturação q é o seguinte: o número de pontos em uma célula não-vazia pode ser tão pequeno quanto um e tão grande quanto muitos milhares. Em uma sequência típica a maioria

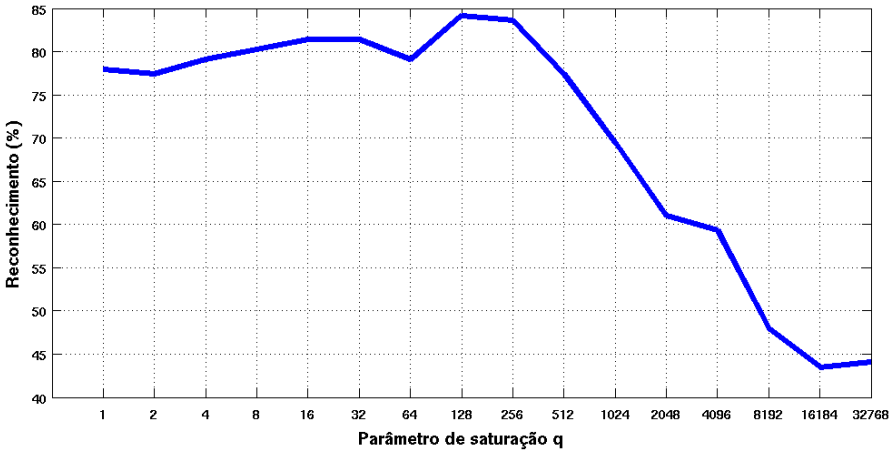


Figura 4.7. Taxa de reconhecimento para diferentes valores do parâmetro de saturação q. Todos os 20 tipos de ação são incluídos em um teste *Cross-Subject*, onde metade dos sujeitos são usados para treinamento e a outra metade como teste.

das células contém menos de q pontos. Se usássemos diretamente o histograma sem saturação, ou com valores de saturação muito altos, células que contém poucos pontos teriam papel insignificante na classificação. Por outro lado, células com poucos pontos estão associadas a silhuetas e partes móveis do corpo e, portanto, são muito importantes para a classificação e, então, a saturação por valores pequenos permite que essas células desempenhem um papel significativo na classificação. Esse comportamento se repete nos testes I e II.

Em segundo lugar, apresentamos nossos resultados de reconhecimento sobre todos os 20 tipos de ação sem dividir em subconjuntos. Não podemos comparar nossos resultados com os de Li et al. porque eles não realizaram esse experimento. A representação usada em [Li et al., 2010] requer o processamento de grande volume de dados tanto no processo de modelagem das poses por GMMs como no processo *clustering* para computação das poses salientes. A Tabela 4.5 mostra a taxa de reconhecimento e as matrizes de confusão para os três testes são mostradas nas Figura 4.8, 4.9 e 4.10. Pode-se ver que, apesar de o número de classes de ações ter aumentado de oito para 20, a taxa do reconhecimento só foi afetada por uma pequena porcentagem de cerca de 3%.

Conforme pode-se observar, na Figura 4.10, a classificação no Teste III (*cross-subject*) apresenta maiores erros para determinados tipos de ação. Esses error são mais acentuados para ações como *Hammer*, *draw X*, *Draw circle* e *Tennis serve*, compostos por poses não muito distintas e com grande variabilidade entre diferentes sujeitos. Conforme observado em [Li et al., 2010], durante a coleta de dados, os sujeitos

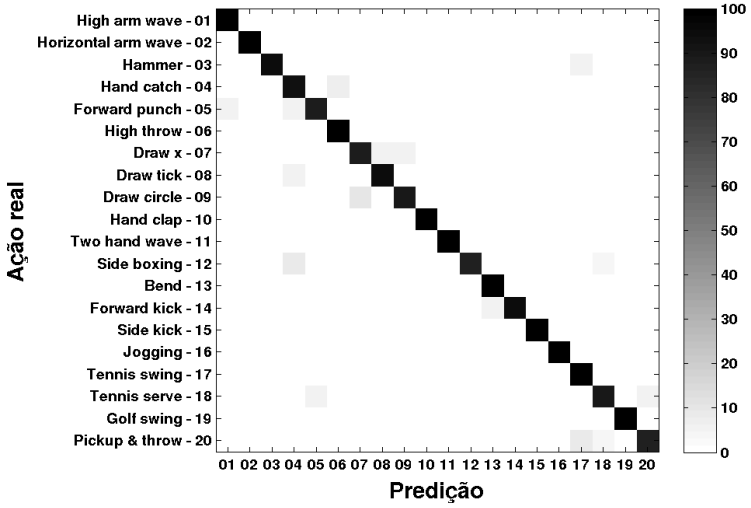


Figura 4.8. Matriz de confusão dos resultados para o Teste I realizado para todos os 20 tipos de ação. Neste experimento, 1/3 das amostras foram usadas para treinamento e os outros como testes com $95,21\%$ de reconhecimento.

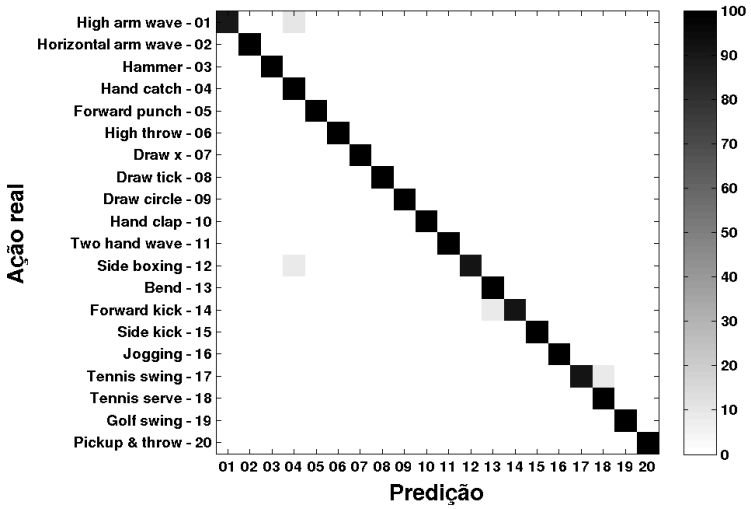


Figura 4.9. Matriz de confusão dos resultados para o Teste II realizado para todos os 20 tipos de ação. Neste experimento, 2/3 das amostras foram usadas para treinamento e os outros como testes com $97,83\%$ de reconhecimento.

Teste	Reconhecimento (%)
Teste I	95,21
Teste II	97,83
Teste III	81,55

Tabela 4.5. Resultados do reconhecimento usando PCA-STOP em todos os 20 tipos de ação.

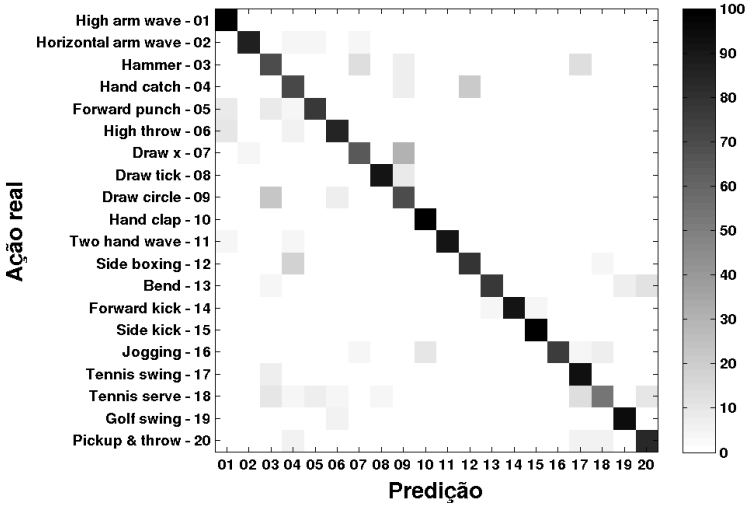


Figura 4.10. Matriz de confusão dos resultados para o Teste III realizado para todos os 20 tipos de ação. Neste experimento, metade das amostras foram usadas para treinamento e os outros como testes com $81,55\%$ de reconhecimento.

escolheram livremente o estilo para execução das ações, levando a significantes variações na execução da mesma ação executada por sujeitos diferentes. Na subseção 4.3.3, onde cada ação será modelada como um grafo sobre um mesmo conjunto de poses salientes, comparamos tais grafos para apresentar uma medida de similaridade entre as diferentes ações.

Finalmente, comparamos a classificação *offline* usando nosso classificador OCL com distância do cosseno em relação a dois métodos clássicos da literatura: KNN com distância euclidiana [Dasarathy, 1991] e SVM [Chang & Lin, 2011]. Nesse experimento, realizamos os testes I, II e III considerando todas as 20 classes de ação. Na Figura 4.11 os resultados nos três testes são mostrados para os classificadores OCL, KNN e SVM. Percebemos que uma característica da classificação com OCL é a normalização dos vetores para norma 1, o que inviabiliza a comparação pela distância Euclidiana usando KNN. Enquanto que a classificação usando OCL obtém média de 91.53% de reconhecimento, a classificação usando KNN obtém apenas 41.08%. Já a classificação

usando SVM obtém taxa de reconhecimento médio de 72.04%, ainda muito inferior ao obtido com o uso do classificador OCL.

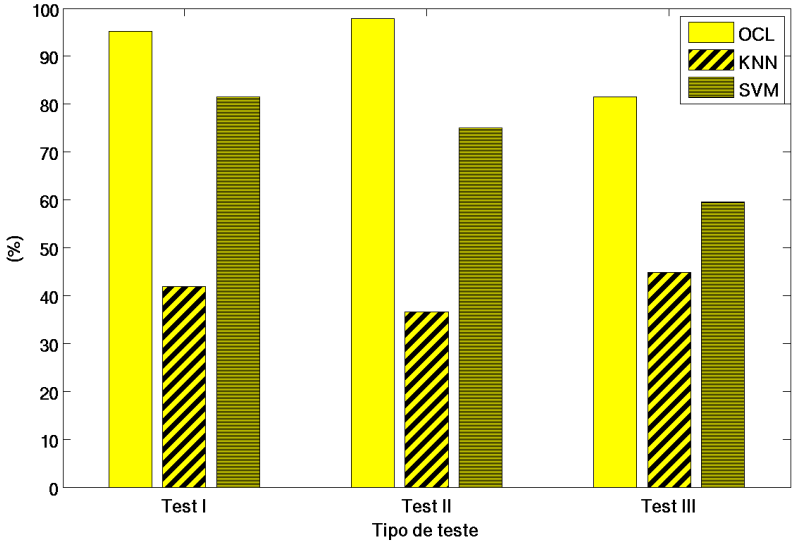


Figura 4.11. Comparação da classificação *offline* usando OCL e outros métodos de classificação. Nos três testes, verifica-se que nossa classificação usando OCL obtém as melhores taxas de reconhecimento.

4.3.2 Classificação *online*

Para permitir a classificação *online* de sequências de mapas de profundidade, duas condições principais precisam ser tratadas: o alinhamento temporal e a segmentação de ações na sequência. Para tratar o alinhamento temporal, usamos um classificador baseado num grafo de ação para aprender um conjunto de posturas salientes e um modelo de transição para cada classe de ação, de forma que cada ação é modelada como um caminho no grafo. Para tratar a segmentação de ações no tempo, adaptamos nosso grafo de ações com um detector de poses neutras para identificar os quadros de início e fim das ações na sequências de mapas de profundidade.

Para uma pequena janela de quadros na sequência de mapas de profundidade, computamos um descritor STOP curto. Nossa implementação usa uma janela de cinco quadros. Como o número de quadros é pequeno, definimos um único segmento ao longo do eixo do tempo quando particionamos o volume espaço temporal dos cinco quadros. Obtemos então um descritor PCA-STOP para cada STOP obtido. O conjunto de descritores obtidos a cada cinco quadros será então utilizado em nossa estratégia de reconhecimento *online*.

Para executar a etapa de segmentação no reconhecimento *online*, treinamos um classificador de poses neutras utilizando uma máquina de suporte vetorial (SVM, do inglês *Support Vector Machine*), conforme implementação disponibilizada na biblioteca *LibSVM* [Chang & Lin, 2011]. Neste reconhecimento de poses neutras, usamos os mesmos descritores STOP que usamos para o reconhecimento das diversas ações. Um conjunto de quadros onde pessoas estão em posição de repouso, sem executar qualquer das ações de treinamento, foi coletado para ser utilizado como dados positivos no classificador. Um outro conjunto, onde pessoas executam diversas ações da base de treinamento, foi coletado para ser utilizado como dados negativos no classificador. Estas sequências foram segmentadas em pequenas janelas de cinco quadros para computar os descritores PCA-STOP usados para treinar o classificador SVM de poses neutras.

O sistema de classificação *online* com segmentação automática é implementado em uma máquina de estados finitos que mantém dois estados: estado *inativo* e estado de *ação*. Note que esses dois estados são diferentes dos estados no grafo de ação. Para cada cinco quadros, o sistema calcula o descritor PCA-STOP e aplica o classificador SVM de poses neutras. Se o estado atual é *inativo*, enquanto recebe uma pose neutra, ele permanece no estado *inativo*. Se ele recebe uma pose não neutra, ocorre uma transição de estado de *inativo* para *ação*. Enquanto em estado de *ação*, o grafo de ação é acionado para realizar a decodificação sempre que um novo descritor PCA-STOP é adicionado. Quando o sistema detecta uma pose neutra, ele retorna para o estado *inativo*. Nesse meio tempo, ele reinicia o grafo de ação para se preparar para a próxima ação.

A Figura 4.12 mostra uma visão geral do sistema de classificação online com segmentação.

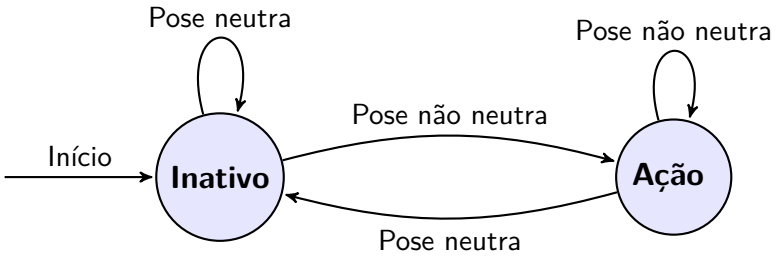


Figura 4.12. Visão geral do sistema *online* de reconhecimento de ações. Estados *inativo* e *ação* são usados para manter o sistema inativo, enquanto recebe poses neutras e uma transição para o estado de *ação* é feita quando uma pose não-neutra é recebida. Enquanto em estado de *ação*, o grafo de ação realiza a decodificação.

Para o alinhamento temporal, usamos uma abordagem chamada *action graph*, proposta em [Li et al., 2008], para o reconhecimento de ação em vídeo. Em comparação com a classificação *offline*, o grafo de ação tem a vantagem de permitir se realizar a classificação (decodificação) em tempo real, não sendo necessário esperar até que a ação termine para iniciar a classificação. Similar à cadeia de Markov escondida (HMM, do inglês *Hiden Markov Model*), o gráfico de ação é flexível para tratar variações de velocidade na execução da ação e leva em consideração a dependência temporal [Rabiner, 1990]. Comparado à HMM, o grafo de ação tem a vantagem de que ele exige menos dados de treinamento e permite que diferentes ações possam compartilhar os estados.

Formalmente, um grafo de ação é um sistema composto de um conjunto $\Psi = \{a_1, a_2, \ldots, a_L\}$ com L classes de ações treinadas, um conjunto $\Pi = \{v_1, v_2, \ldots, v_R\}$ com R poses salientes, um conjunto $\Sigma = \{p(e|v_1), p(e|v_2), \ldots, p(e|v_R)\}$ com modelos de observação de um vetor descritor e com relação à pose saliente $v_i \in \Pi$ e, finalmente, um conjunto $\Gamma = \{P_1, P_2, \ldots, P_L\}$ com L matrizes $R \times R$ para modelar as probabilidades de transição entre as poses salientes, dada uma classe de ação.

O procedimento de treinamento para o grafo de ação consiste em aprender o conjunto de poses salientes Π e o conjunto de matrizes de transição Γ entre poses salientes. Para aprender as poses salientes, agrupamos todos os vetores descritores PCA-STOP, de todos os tipos de ação dos dados de treinamento usando o procedimento de *clustering* K-means. Essa etapa é diferente de [Li et al., 2010] porque, na nossa representação por densidade de ocupação, a pose é representada por um único vetor descritor, enquanto [Li et al., 2010] utiliza um conjunto de pontos não organizados na sua descrição baseada em *bag of features*. Uma outra distinção é que eles usam distância de Hausdorff no processo de *clustering* de poses dadas por pontos não organizados (*bag of points*) e nós usamos a distância do cosseno para as poses dadas por um único vetor descritor. As poses salientes resultantes Π serão os nós (ou estados) do grafo de ação. Para cada *cluster* $v_j \in \Pi$, ajustamos uma distribuição Gaussiana e estimamos o modelo de probabilidade $p(e|v_j) \in \Sigma$ de observação da pose saliente v_j, dada uma pose de entrada e.

Cada matriz de transição $P_k \in \Gamma$ é computada como $p(j|i) = \frac{N_{(i \to j)}}{N_i}$, onde $p(j|i)$ é a probabilidade de transição da pose v_i para a pose v_j, $N_{(i \to j)}$ é o número de transições da pose saliente v_i para v_j nos dados de treinamento que pertencem à ação a_k e N_i é o número de vezes que a pose v_i é observada nos dados de treinamento que pertencem à ação a_k.

A Figura 4.13 ilustra um grafo de ação onde duas ações distintas, representadas pelas transições em vermelho e azul, são modeladas como grafos sobre o mesmo

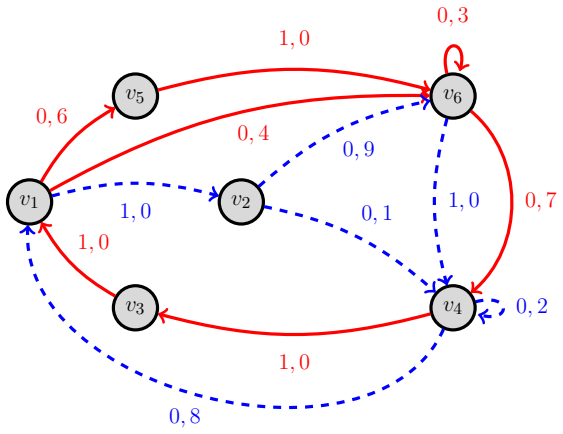


Figura 4.13. Exemplo de grafo de ação onde duas ações distintas são modeladas para o mesmo conjunto de poses salientes. Os grafos distintos sobre o mesmo conjunto de poses são identificados nas transições em vermelho (contínuo) e azul (pontilhado) de suas transições. Note que os estados v_1, v_4 e v_6 são compartilhados pelas duas ações, cada ação é definida por mais de um caminho e o laço em alguns estados indica a probabilidade de uma pose ter duração variada entre diferentes execuções da mesma ação.

conjunto de poses salientes. Note que os estados v_1, v_4 e v_6 são compartilhados pelas duas ações, cada ação é definida por mais de um caminho, e o laço em alguns estados indica a probabilidade de uma pose ter duração variada entre diferentes execuções da mesma ação.

Dada uma sequência de mapas de profundidade de teste, obtemos uma sequência de vetores PCA-STOP de teste $E = \{e_1, e_2, \ldots, e_n\}$, onde cada vetor será classificado como pose *neutra* ou pose *não-neutra*. A máquina de estados acionará a decodificação de ações para as subsequências de poses *não-neutras*.

Para computar a probabilidade de ocorrência de uma sequência S de poses salientes com relação a uma determinada classe de ação a_k, precisamos computar a pose saliente $v_j \in \Pi$ associada a cada elemento $e_i \in E$. Isso leva a considerar $p(e_i|v_j)$ para todo $e_i \in E$ e todo $v_j \in \Pi$, resultando em um grande número de hipóteses para sequências de poses salientes que gerou E. Tomando, por hipótese, a sequência $S = \{v_1, v_2, \ldots, v_n\}$ de poses salientes, a probabilidade de ocorrência de E com relação a sequência S e ação a_k é dada por

$$p(E|S, a_k) = p(v_2|v_1, a_k)p(v_3|v_2, a_k) \ldots p(v_n|v_{n-1}, a_k)p(e_1|v_1)p(e_2|v_2) \ldots p(e_n|v_n). \tag{4.7}$$

Note que $p(v_j|v_i, a_k)$ é dado pelo elemento (i, j) da matriz P_k que modela a probabilidade de transição entre poses v_i e v_j dada a ação a_k; e $p(e_i|v_i)$ é calculada considerando a distribuição $\mathcal{N}(v_i, \sigma_i)$ que modela a probabilidade de observação de e_i dada a pose saliente v_i. De forma compacta, escrevemos

$$p(E|S, a_k) = \prod p(v_{j+1}|v_j, a_k)_{j=1,\ldots,n-1} \prod p(e_i|v_i)_{i=1\ldots n}. \tag{4.8}$$

Considerando todas as possíveis sequências S_q de poses salientes, a probabilidade de ocorrência de E com relação a ação a_k é dada por

$$p(E|a_k) = \max_q \{p(E|S_q, a_k)\}. \tag{4.9}$$

Finalmente, a ação $\bar{a}$ mais provável com relação à sequência de vetores PCA-STOP E será dada por

$$\bar{a} = arg \max_{a_k} \{p(E|a_k)\}. \tag{4.10}$$

Para não ter que enumerar todas as possíveis sequências de poses salientes e, consequentemente, pagar o alto custo computacional de calcular todas as probabilidades envolvidas, usamos uma técnica de programação dinâmica para a decodificação de ações no grafo. O algoritmo de programação dinâmica usado é o *Viterbi* [Viterbi, 2006] que considera apenas a subsequência de maior probabilidade a cada passo em vez de calcular as probabilidades para todos os caminhos possíveis. A ideia principal é computar, para cada classe de ação a_k, uma matriz D_k de dimensão $R \times n$. D_k tem n colunas correspondendo aos n vetores PCA-STOP da sequência a ser classificada. Cada coluna possui R linhas, correspondendo às R possíveis poses salientes.

A primeira coluna de D_k guardará, em cada linha j, as probabilidades de observação do vetor e_1 em relação a cada uma das poses salientes, ou seja,

$$D_k(j, 1) = p(e_1|v_j), j = 1, \ldots, R. \tag{4.11}$$

A segunda coluna de D_k guardará, em cada linha j, a maior das probabilidades associadas a subsequências de 2 poses salientes que terminam em v_j, ou seja

$$D_k(j, 2) = \max_t \{D_k(t, 1)p(v_j|v_t, a_k)\}p(e_2|v_j), j = 1, \ldots, R. \tag{4.12}$$

Generalizando, a i-ésima coluna de D_k guardará, em cada linha j, a maior das probabilidades associadas a subsequências de i poses salientes que terminam em v_j, ou seja

$$D_k(j,i) = \max_t \{D_k(t,i-1)p(v_j|v_t,a_k)\}p(e_i|v_j), j = 1,\ldots,R. \tag{4.13}$$

Concluída a computação da matriz D_k, o maior valor na última coluna será a probabilidade de observação da sequência E em relação à classe de ação a_k.

Um subproduto que se obtém com a computação da matriz D_k é que obtemos a sequência mais provável de estados, ou poses salientes, associada à sequência de vetores PCA-STOP dada. Neste caso, basta guardar a sequência de melhores índices t ao computar $D_k(j,i)$.

Desta forma, para cada classe de ação a_k, construímos D_k com o procedimento de programação dinâmica para obter $p(E|a_k)$. Finalmente, a ação $\bar{a}$ mais provável com relação a sequência de vetores PCA-STOP E será dada por

$$\bar{a} = arg\max_{a_k} \{p(E|a_k)\}. \tag{4.14}$$

Este procedimento de decodificação de ações é muito eficiente porque na Equação 4.13, $D_k(t,i-1)$ e $p(v_j|v_t,a_k)$ são consultas à matriz dinâmica D_k e à matriz de transição P_k. Observando que $p(e_i|v_j)$ será calculado retornando o mesmo valor para todas as classes de ação, esses valores podem também ser previamente calculados e armazenados em uma matriz de forma que o procedimento de decodificação dinâmica faz apenas consultas a matrizes previamente calculadas. O Algoritmo 2 ilustra o procedimento de decodificação dinâmica de ações.

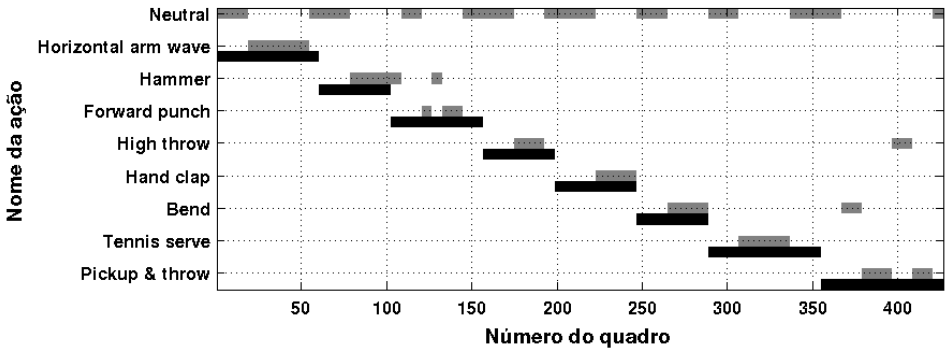
Para nossos experimentos em classificação *online*, usamos sequências longas não segmentadas de mapas de profundidade da mesma base dados [Li et al., 2010], onde diversas ações são realizadas ao longo do tempo. Nosso classificador de poses neutras, para segmentação, e grafo de ação, para alinhamento temporal, são usados para o reconhecimento de ações com a segmentação e alinhamento automáticos. Para este experimento foram utilizadas 511 sequências de mapas de profundidade de nove sujeitos como conjunto de treinamento e 56 sequências de mapas de profundidade de um sujeito diferente como conjunto de teste. O conjunto de treinamento é usado para aprender o grafo de ação. As sequências no conjunto de teste são concatenadas para formar uma sequência longa e não segmentada de teste. Fazemos isso 10 vezes, cada uma com uma partição diferente da base de dados, onde os sujeitos no conjunto de teste não são vistos no conjunto de treinamento. Nosso grafo de ação foi modelado com 50 poses salientes. A precisão global para esta classificação cruzada *10-fold* foi de $98,41\%$ em um teste

Algorithm 2 DynamicDecode(E,Π)

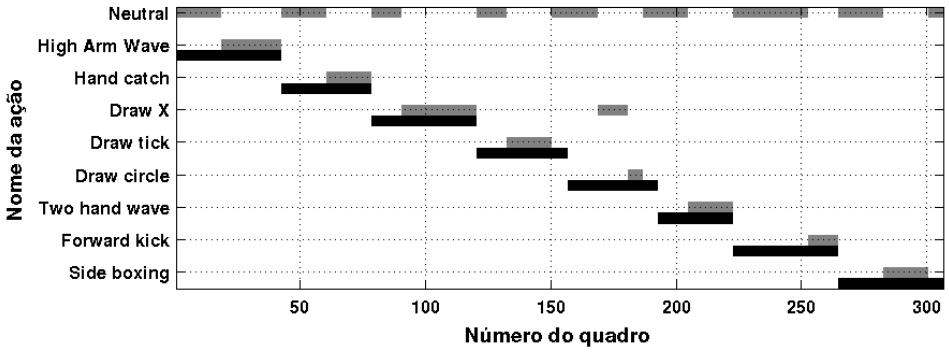
for each $e_i \in E$ **do**
  for each $v_j \in \Pi$ **do**
    compute $p(e_i|v_j)$
  end for
end for
Z = Zeros(L);
for k from 1 to L **do**
  D_k = Zeros(R,n)
  for j from 1 to R **do**
    $D_k(j,1) = p(e_1|v_j)$
  end for
  for i from 2 to n **do**
    for j from 1 to R **do**
      $D_k(j,i) = \max_t\{D_k(t,i-1)p(v_j|v_t,a_k)\}p(e_i|v_j)$
    end for
  end for
  $Z(k) = max(D_k(:,n))$;
end for
return $argmax_k(Z(k))$

usando todos os 20 tipos de ação, o que enfatiza que o tratamento do alinhamento temporal melhora a classificação.

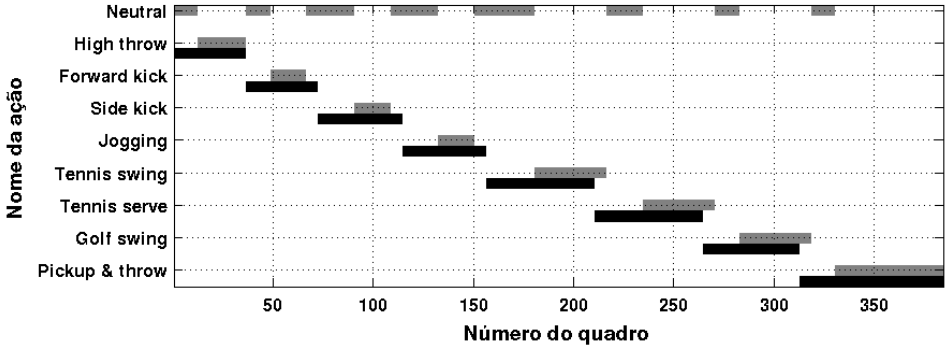
Para ilustrar a segmentação e classificação, a Figura 4.14, mostra alguns exemplos de classificação de sequências não segmentadas. A figura mostra, em cada quadro, matrizes onde colunas representam o número de quadros ao longo do tempo e as linhas representam as classes de ação. Em preto, a segmentação manual das ações na sequência ao longo do tempo e, em cinza, nossa segmentação e classificação automática. Note que a pose neutra deve ser classificada antes e depois da execução de cada classe de ação. Isto é coerente com a base de dados onde cada sujeito assume a posição neutra entre a execução de duas diferentes ações. Em (a) temos um exemplo de classificação para o conjunto de ações $AS1$ onde a ação *Pickup & throw* é confundida com *Bend* e *High throw* em alguns quadros, em (b) temos um exemplo de classificação para o conjunto de ações $AS2$ onde a ação *Draw circle* é confundida com *Draw X* em alguns quadros e, em (c), temos um exemplo de classificação para o conjunto de ações $AS3$. Na maioria dos casos, os erros de classificação se devem a características da própria base de dados, por exemplo, a ação *Pickup & throw* é uma composição das ações *Bend* e *High throw*.



(a)



(b)



(c)

Figura 4.14. Exemplos de classificação de sequências longas e não segmentadas. Em preto, a segmentação manual das ações na sequência ao longo do tempo e, em cinza, a segmentação e classificação automática. (a) Exemplo de classificação para ações do $AS1$ onde ações *Pickup & throw, Bend e High throw* são confundidas em alguns quadros. (b) Exemplo de classificação para ações do $AS2$ onde a ação *Draw circle e Draw X* são confundidas em alguns quadros. (c) Exemplo de classificação para ações do $AS3$.

4.3.3 Similaridade entre ações

Os resultados, tanto na classificação *offline* como *online*, mostram que os erros de classificação são mais acentuados para um certo grupo de ações. Então, desenvolvemos uma estratégia para estimar uma medida de similaridade entre ações que permita investigar e estabelecer numericamente a proximidade entre ações a partir do seu grafo de ação.

Diversos trabalhos na literatura propõem métricas para comparação de grafos gerais. Entretanto o grafo que construímos, como representação para as classes de ação, compartilham nós e tem características peculiares que permitem o estabelecimento de estratégia simples de comparação.

Como cada ação é modelada como um grafo sobre o mesmo conjunto de nós, eles diferem apenas nas transições e suas probabilidades, ou arestas orientadas e seus pesos. Dessa forma, propomos uma métrica de similaridade baseada nos pesos das arestas. Sejam dois grafos de ação, dados por suas matrizes de transição P_a e P_b, identificamos o maior subgrafo comum P_{ab}, dado pela matriz de transição

$$P_{ab}(i,j) = \min\{P_a(i,j), P_b(i,j)\}, \tag{4.15}$$

e computamos a razão entre a soma dos pesos nas arestas de P_{ab} em relação à soma dos pesos em P_a e P_b. Mais formalmente, a similaridade $\Delta(P_a, P_b)$ entre os grafos de ação P_a e P_b é computada como

$$\Delta(P_a, P_b) = \frac{2\sum_{ij} P_{ab}(i,j)}{\sum_{ij} P_a(i,j) + \sum_{ij} P_b(i,j)}. \tag{4.16}$$

A medida de similaridade Δ varia no intervalo $[0,1]$ com 0 significando grafos de ação completamente distintos e 1 grafos de ação idênticos. Comparando todas as ações, verificamos que a similaridade média é de $0,1455$ com desvio de $0,0809$. Os maiores valores de similaridade foram verificados para as ações *forward punch* e *high throw* com valor $0,4248$ e para ações *draw tick* e *draw circle* com valor $0,4064$. Para ilustrar graficamente o nível de similaridade entre as ações apresentamos, na Figura 4.15, uma matriz de similaridade para os 20 tipos de ação com base na comparação dos seus grafos de ação. Usamos o maior subgrafo comum, não necessariamente conexo, devido ao fato de que ações similares podem ser caracterizadas por semelhanças em diversos movimentos, caracterizando mais de um bloco conexo na matriz de transição.

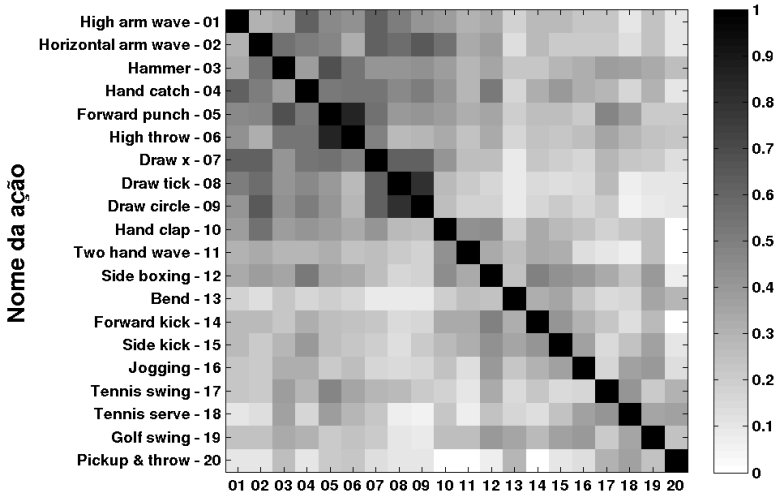


Figura 4.15. Matriz de similaridade entre os 20 tipos de ação com base na comparação dos grafos de ação. A similaridade média é de $0,1455$ e as ações com maiores valores de similaridade são *forward punch* e *high throw* com similaridade $0,4248$ e ações *draw tick* e *draw circle* com $0,4064$.

4.3.4 Invariância ao ponto de vista

No reconhecimento de ações humanas, a classificação está diretamente relacionada ao problema de correspondência entre poses. Independentemente da representação espacial adotada, o reconhecimento depende da correspondência entre características de uma sequência de teste e as sequências de treinamento. Um problema comum que se apresenta ao lidar com correspondências está no fato de que os objetos não são, geralmente, capturados de um mesmo ponto de vista e, portanto, não são representados no mesmo sistema de coordenadas. Dessa forma, o problema de invariância ao ponto de vista ou, mais formalmente, a transformações de translação, rotação e escala é um importante aspecto a ser tratado no desenvolvimento de um sistema robusto para reconhecimento de ações.

Geralmente as sequências de nuvens de pontos obtidas para descrever a execução de uma ação têm coordenadas em um sistema de coordenadas global de forma que, sequências relacionadas à mesma ação, mesmo que executadas por uma mesma pessoa, se capturadas de pontos de vista distintos ou com diversa orientação do corpo, podem descrever nuvens de pontos muito diferentes tornando difícil o processo de classificação. Idealmente, as nuvens de pontos deveriam ser capturadas de um mesmo ponto de vista e com os corpos numa mesma orientação em relação à câmera para que o nosso

processo de classificação por comparação de densidade de ocupação espacial tenha êxito, especialmente devido a diferenças na orientação da grade. Como nem sempre isso é possível, descrevemos nesta seção uma metodologia para estimar a orientação do corpo de forma a permitir uma classificação com invariância ao ponto de vista.

Em diversos trabalhos sobre alinhamento de nuvens de pontos, um conjunto de pares de pontos correspondentes é obtido usando algum descritor de características que permita estabelecer a correspondência independente do ponto de vista [Nascimento et al., 2012; Tombari et al., 2011]. No caso particular do reconhecimento de ações humanas, a identificação de juntas e extremidades de forma independente em cada nuvem de pontos permite estabelecer tais pares pela identificação de juntas correspondentes. A localização destas juntas do corpo humano pode ser estimada, como proposto por Shotton et al. [Shotton et al., 2011], com um único mapa de profundidade de entrada, inferindo uma distribuição de partes do corpo por *pixel* com invariância ao ponto de vista. A Figura 4.16 ilustra esse processo.

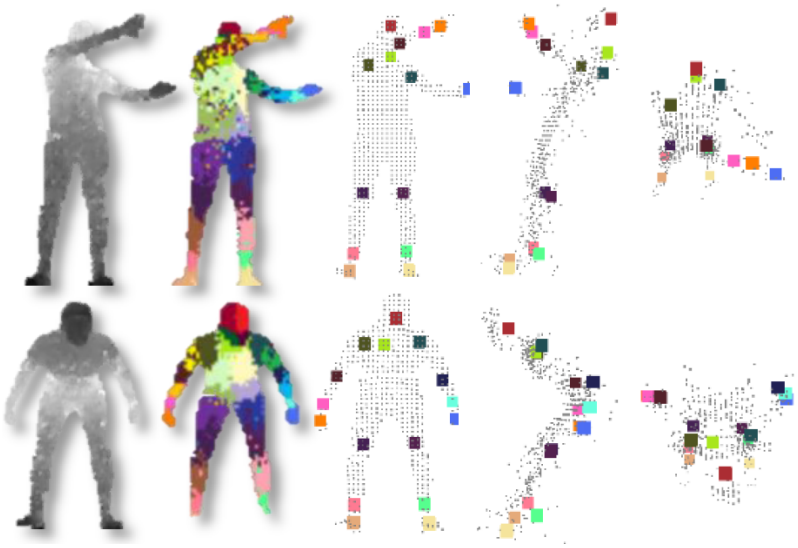
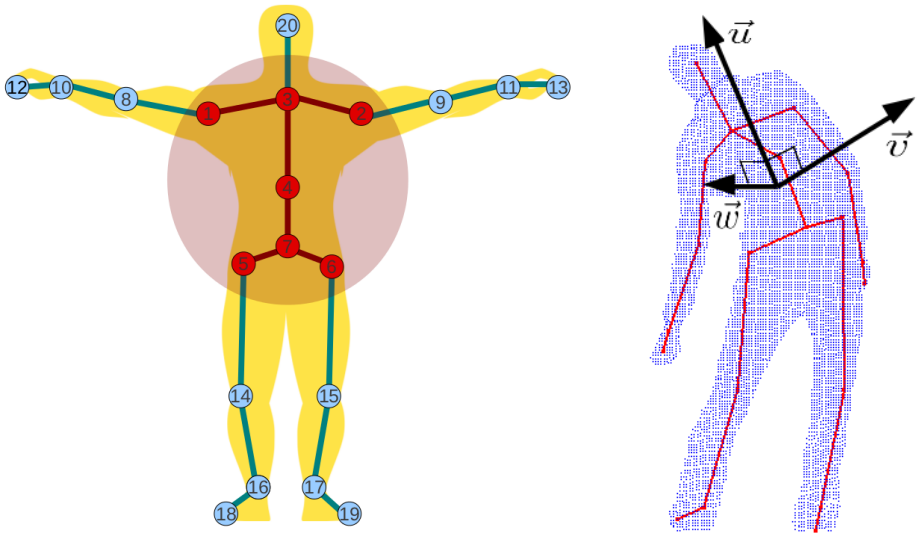


Figura 4.16. Ilustração do processo de estimação das juntas a partir do mapa de profundidade conforme [Shotton et al., 2011]. Com uma única imagem de profundidade de entrada, é inferida uma distribuição de partes do corpo por *pixel*. Da esquerda para direita, o mapa de profundidade, a classificação dos *pixels* em cores indicando a junta mais provável a que pertence, localização estimada das juntas vistas de frente, juntas vistas de lado, e juntas vistas de cima.

Diversos trabalhos em reconhecimento de ações humanas tem sido baseados somente nos esqueletos [Raptis et al., 2011; Miranda et al., 2012]. Usando esqueletos é possível obter invariância ao ponto de vista construindo descritores baseados em ângulos entre juntas ou usando uma matriz de distâncias, como mostramos em [Vieira et al., 2012]. Uma estratégia que tem sido muito utilizada é a estimação de um sistema

de coordenadas local, a partir do qual as juntas são descritas de forma invariante. No trabalho de Raptis et al. [Raptis et al., 2011], esse sistema de coordenadas local é estimado a partir de sete pontos associados ao tronco do corpo humano. Eles calculam as componentes principais para os sete pontos do tronco, ou seja, uma base ortonormal $3D$ como resultado da aplicação de PCA à matriz 7×3 de pontos do torso. A primeira componente principal $\vec{u}$ estará sempre alinhada com a dimensão maior do torso. Em seguida, a segunda componente principal $\vec{v}$ é alinhada com a linha que liga os ombros. Finalmente, o último eixo da base ortonormal $\vec{w}$ é calculado como o produto vetorial das duas primeiras componentes. Esse sistema de coordenadas local é denominado *torso PCA frame*. A Figura 4.17 detalha as juntas do esqueleto e sistema de coordenadas local obtido. São identificadas 20 juntas das quais sete, relacionadas ao torso, tem posição relativa rígida com pouca variabilidade servindo para definir um sistema de coordenadas local, a partir do qual a nuvem de pontos será descrita.



(a) (b)

Figura 4.17. Detalhes das juntas do esqueleto e sistema de coordenadas local obtido conforme em [Raptis et al., 2011]. Em (a) são identificadas 20 juntas das quais sete, relacionadas ao torso, tem posição relativa rígida com pouca variabilidade servindo para definir um sistema de coordenadas local (b), a partir do qual a nuvem de pontos será descrita.

Geralmente, esse sistema de coordenadas local é usado para criar descritores invariantes ao ponto de vista para classificação usando as próprias juntas do esqueleto [Raptis et al., 2011; Miranda et al., 2012]. Como nos propomos a comparar as nuvens de pontos representadas por densidade de ocupação, este sistema de coordenadas local será utilizado para alinhar as respectivas nuvens de pontos que serão, posteriormente,

usadas para construir os descritores STOP com invariância ao ponto de vista. Para tanto, usamos os esqueletos associados à cada nuvem de pontos para obter o sistema de coordenadas local, projetamos os pontos nesta base local e, então, executamos o nosso algoritmo de classificação para as sequências de nuvens de pontos em suas coordenadas locais. A Figura 4.18 ilustra o processo de classificação de ações usando alinhamento espacial da nuvem de pontos pela extração dos esqueletos.

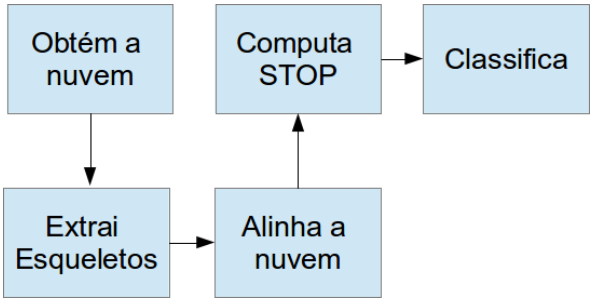


Figura 4.18. Ilustração do processo de classificação de ações usando alinhamento espacial da nuvem de pontos. A partir da nuvem de pontos são extraídos os esqueletos, os quais são usados para obter um sistema de coordenadas local, no qual os pontos são projetados para obter alinhamento espacial. Os descritores STOP são, então, computados para a nuvem alinhada e executamos o nosso algoritmo para classificação.

Em nossos experimentos com invariância ao ponto de vista, os testes I e II não apresentaram ganhos em termos de acurácia no reconhecimento. Nesses testes, os indivíduos na base de treino e teste são os mesmos, então há pouca variação na orientação do corpo e alta taxa de reconhecimento já é obtida mesmo sem alinhamento prévio das nuvens de pontos. Entretanto, para o Teste III, em que os indivíduos na base de treino não são usados na base de testes, observamos um ganho significativo na taxa de reconhecimento. Este ganho se deve ao fato de que, ao descrever a nuvem de pontos num sistema de coordenadas local, diferenças de orientação do corpo entre indivíduos é corrigida pelo alinhamento obtido com as juntas correspondentes do torso. A Tabela 4.6 mostra as taxas de classificação para o Teste III (*cross subject*) usando descritores STOP com e sem alinhamento da nuvem de pontos.

Note que, para os conjuntos de ação $AS1$ e $AS3$, as taxas de reconhecimento aumentaram com o alinhamento da nuvem de pontos usando o sistema de coordenadas local obtido com as juntas do torso. Para o conjunto de ações $AS2$, entretanto, houve uma redução na taxa de reconhecimento. Atribuímos esse resultado ao fato de que, para este conjunto de ações, onde as mãos se projetam sobre o torso, os pontos do torso sofrem oclusão e a estimativa de suas coordenadas fica prejudicada inserindo muito

Conjunto	Sem alinhamento	Com alinhamento
AS1	84,70	**91,67**
AS2	81,30	72,22
AS3	88,40	**98,61**
Avg	84,80	**87,50**

Tabela 4.6. Comparação das taxas de reconhecimento no Teste III com e sem alinhamento da nuvem de pontos.

ruido à estimativa do sistema de coordenadas local. Consequentemente, o alinhamento incorreto, em vez de contribuir, prejudica a classificação.

4.3.5 Reconhecimento em tempo real

Em [Li et al., 2010], onde *Bag of Features* e GMM são usadas como representação e a métrica de similaridade é a distância de Hausdorff, não foi apresentado tempo de processamento para treinamento e classificação. Entretanto os autores relatam o grande volume de processamento necessário devido à representação ineficiente e métrica de similaridade usadas, que torna inviável o reconhecimento de tempo real. Na representação por densidade de ocupação, a construção da representação e comparação eficientes permitem que o reconhecimento seja feito em tempo real. A fim de testar o desempenho do sistema para classificação em tempo real, realizamos um teste *Cross-Subject* com todos os 20 tipos de ações e medimos o tempo para treinamento e testes usando um PC padrão rodando a 2.9 GHz com 2 GB RAM. O treinamento para 270 sequências de mapas de profundidade foi feito em $67s$, enquanto que os testes para 297 sequências foram feitos em $14s$. Desta forma, nosso método é capaz de classificar ações com entrada a taxa de até 20 quadros por segundo em um PC padrão.

Para realizar experimentos em tempo real, foi utilizado um robô móvel Pioneer P3-AT equipado com um sensor do tipo *Kinect* e um notebook para executar o programa de classificação e interface com o robô. A Figura 4.19 ilustra o *setup* experimental de nossos testes. O sistema consta dos seguintes passos:

- Leitura: Um sensor de profundidade é adaptado ao robô para captura de nuvens de pontos a taxa de 15 quadros por segundo;
- Janela deslizante: Uma segmentação temporal por janela deslizante é adaptada para agrupar quadros dos últimos 5 segundos com intervalos de 1 segundo;
- Normalização: Os quadros são normalizados com relação à localização e escala antes de construir os descritores;

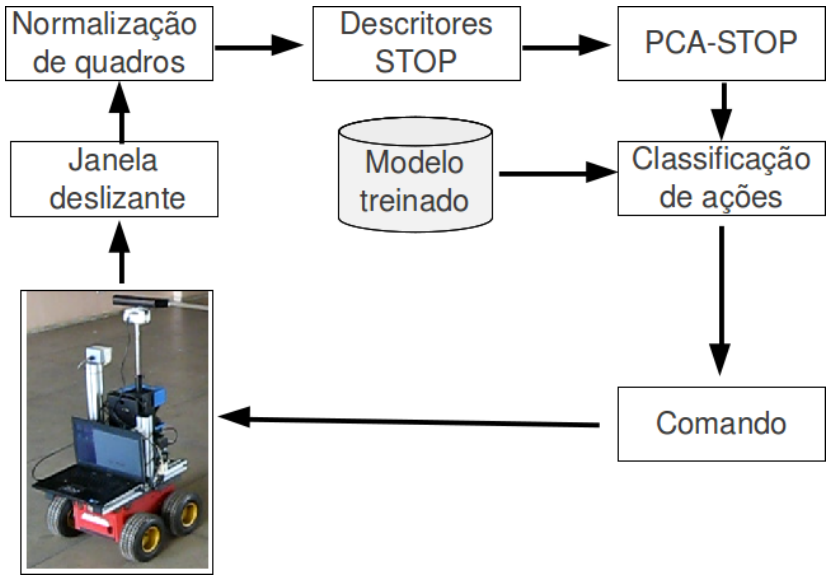


Figura 4.19. Um robô adaptado com um sensor de profundidade captura sequências de mapas de profundidade e usa um conjunto de ações treinadas para reconhecer comandos a partir de ações executadas por um pessoa.

- Descritores: A sequência de quadros acumulados na janela de 5 segundos é usada para construir os descritores STOP e PCA-STOP;
- Classificação: Um modelo previamente treinado com um conjunto de comandos é usado para classificar a ação executada;
- Comando: A ação é então convertida em comando para controlar o robô.

Treinamos seis ações diferentes a serem usadas para comandar o robô: Parar, avançar, voltar, virar à direita, virar à esquerda e retornar. O sistema de classificação de ações deve reconhecer os gestos executados por uma pessoa e enviar comandos correspondentes para que o robô execute uma das ações. Três indivíduos com tamanhos diferentes foram utilizados nos experimentos. Na fase de treinamento, cada ação foi realizada três vezes por cada indivíduo. Na fase de classificação em tempo real, cada ação foi executada 10 vezes por cada um dos três indivíduos em diferentes posições e distâncias em relação ao robô. A taxa média de reconhecimento foi $96,11\%$. A Figura 4.20 apresenta uma matriz de confusão para as taxas de reconhecimento de cada ação em tempo real.

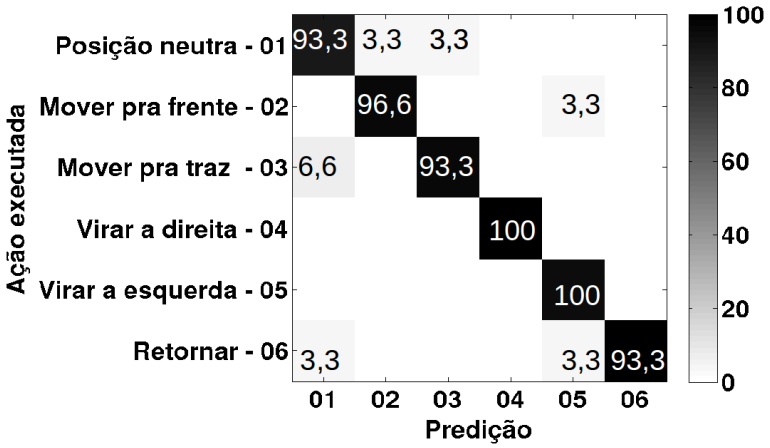


Figura 4.20. Resultados para testes em tempo real em %. Cada ação foi realizada 10 vezes por diferentes sujeitos de diferentes posições e distância do robô. A taxa média do reconhecimento foi de $96,11\%$.

Capítulo 5

Conclusões e Trabalhos Futuros

5.1 Conclusões

Este trabalho explorou a solução de problemas relacionados à representação e comparação entre nuvens de pontos. Ao longo da nossa revisão bibliográfica, verificamos que a eficiência na comparação está diretamente relacionada à representação usada para o conjunto de pontos. Uma grande variedade de alternativas para representação das nuvens de pontos foi estudada, cada uma apresentando características que a torna eficiente dentro de determinado contexto. Entretanto, verificamos que, para comparação, os modelos paramétricos de densidade espacial são muito usados, especialmente os modelos de mistura, em que uma ou mais distribuições de melhor ajuste são calculadas para representar a densidade espacial definida pela nuvem de pontos.

Tais modelos, geralmente normalizados para definir uma função de densidade de probabilidade, definem, para uma amostragem discreta, um campo escalar contínuo e suave onde a cada ponto em um volume espacial é associada uma densidade que torna simples a consulta pela densidade local e, consequentemente, a comparação entre diversos conjuntos. A eficiência destes métodos, entretanto, depende de inferências acerca da forma das distribuições e número de instâncias a serem usadas, tornando limitado o seu poder de generalização e exigindo alto custo computacional para estimação de parâmetros.

Este trabalho explorou formas mais eficientes de obter a representação por densidade de ocupação espacial, bem como alternativas para comparação nesta representação.

Primeiramente, abordamos a detecção de mudança em nuvens de pontos substituindo a representação em mistura de Gaussianas por uma representação

construída por interpolação multi-variada de uma amostragem discreta de densidade estimada eficientemente em pontos de uma grade de um volume espacial. A estimação da densidade nos pontos da grade é obtida eficientemente pelo somatório de densidades locais obtidas pela computação de funções de base radial com centro em cada ponto da nuvem. Diferentemente de modelos paramétricos de densidade, onde primitivas básicas são ajustadas à nuvem de pontos, esta representação permite o ajuste de formas arbitrárias para melhor explicar a distribuição de pontos na nuvem. Nesta representação, ao definir um campo escalar contínuo, a detecção de mudança pode ser obtida como resultado de simples operação booleana de diferença em campos escalares que, também, é uma operação muito eficiente. Os experimentos e comparação com resultados da literatura mostraram que nossa abordagem tem vantagens em termos de eficiência de custo computacional e qualidade dos resultados.

Em outra aplicação, abordamos a comparação de sequências de nuvens de pontos para reconhecimento de ações humanas. Nessa aplicação, também substituímos uma representação em mistura de Gaussianas por uma representação construída como amostragem discreta de densidade estimada em pontos de um volume espacial. Nesta representação, um histograma saturado de contagem de pontos por células espaço-temporais é construído eficientemente e usado como descritor para comparação de sequências de ações humana. Usando uma base de dados pública e comparação com resultados da literatura, mostramos que nossa abordagem é eficiente em termos de custo computacional e obtém ganhos significativos em termos de taxa de classificação.

Entre as contribuições deste trabalho, mostramos que uma representação por densidade de ocupação pode ser obtida em tempo linear para nuvens de pontos. Abordagens tradicionais de representação por densidade para comparação de nuvens de pontos constroem a representação por processos iterativos onde todos os pontos são considerados em cada iteração, levando a um custo computacional que depende do número de pontos, bem como do número de iterações do processo. Além disso, a representação que obtemos não depende da composição de formas básicas, sendo bastante flexível para se ajustar às formas definidas pelas nuvens de pontos.

Outra contribuição da representação proposta é que o armazenamento independe do número de pontos, mas sim do nível de detalhes requerido por cada aplicação. Em diversos cenários, nuvens densas apresentam uma super-amostragem implicando em alto custo de armazenamento e processamento. A nossa representação por amostra de densidade em uma grade permite que a construção e armazenamento sejam obtidos de acordo com o nível de detalhes desejado pelo simples refinamento do particionamento da grade. Este refinamento depende da escala da aplicação e, em nossos experimentos,

exploramos a eficiência da comparação entre nuvens de pontos considerando diversos níveis de refinamento.

Também mostramos que a consulta pela ocupação de uma posição, em nossa representação, é feita em tempo constante independentemente da quantidade de pontos do conjunto original. Uma vez construída a representação por densidade de ocupação espacial, amostrada em uma grade, a densidade local em um ponto é computada pela simples consulta aos vértices da grade. Isto permite que a comparação entre diferentes conjuntos seja feita em tempo linear no número de pontos. Mostramos que, usando operações booleanas no campo escalar definido pela representação por densidade, obtemos operações eficientes para comparação de nuvens de pontos e detecção de mudanças.

Uma limitação da nossa abordagem para representação, bem como das demais abordagens aqui discutidas, é quanto a invariância ao ponto de vista. Para tarefas de comparação, como em detecção de mudança em nuvens de pontos, um pré-processamento usando algoritmos de alinhamento e registro como ICP é necessário para garantir que uma localização espacial, em ambas as nuvens de pontos, esteja associada ao mesmo ponto. No caso de comparação de sequências de nuvens de pontos para reconhecimento de ações humanas, apresentamos uma alternativa de obter o alinhamento das nuvens pela estimação de uma base local a partir das juntas de um esqueleto extraído.

Outra limitação importante está na escalabilidade da representação para altas dimensões. Todas as nossas aplicações consideram nuvens de pontos de, no máximo, dimensão quatro. A restrição para dimensões maiores está no fato de que a representação é construída sobre uma grade que particiona uma caixa envolvente de dimensão N, pela partição ao longo de cada um dos N eixos coordenados. Desta forma, o número de células da grade cresce exponencialmente com a dimensão N do espaço. Entretanto, nuvens em alta dimensão, geralmente obtidas como descritores *Bag of words* em reconhecimento de padrões ou na classificação de textos, costumam não apresentar densidade, tendo às vezes menos pontos que a dimensão do espaço. Por outro lado, o crescente uso de sensores para aquisição de nuvens de pontos do mundo real gera grande quantidade de nuvens e sequências de nuvens tridimensionais onde nossa representação é altamente eficiente para comparação e detecção de mudanças.

5.2 Artigos publicados

O desenvolvimento deste trabalho gerou investigação em diversos campos da visão computacional, robótica e reconhecimento de padrões, gerando resultados que foram publicados em conferências nacionais e internacionais conforme abaixo:

- Vieira, Antônio W. & Neto, Armando A. & Macharet, Douglas G. & Campos, Mario F.M. *Mesh Denoising Using Quadric Error Metric*. In proc. SIBGRAPI 2010. Gramado/Brazil, August 2010. (Qualis B1).

- Vieira, Antônio W. & Drews, Paulo J. & Campos, Mario F.M. *Efficient Change Detection in 3D Environment for Autonomous Surveillance Robots based on Implicit Volume*. In Proc. ICRA 2012. Minnesota/USA, May 2012. (Qualis A1).

- Vieira, Antonio W. & Nascimento, Erickson R. & Oliveira, Grabriel L. & Liu, Zicheng & Campos, Mario F.M. *STOP: Space-Time Occupancy Patterns for 3D Action Recognition from Depth Map Sequences*. In Proc. CIARP 2012, Buenos Aires, Set 2012, (Qualis B2).

- Vieira, Antonio W. & Lewiner, Thomas & Schwartz, William R & Campos, Mario F.M. *Distance Matrices as Invariant Features for Classifying MoCap Data*. In Proc. ICPR 2012, Tsukuba, Nov 2012. (Qualis A1).

- Miranda, Leandro & Vieira, Thales & Martinez, Dimas & Lewiner, Thomas & Vieira, Antônio W. & Campos Mario F. M. *Real-time gesture recognition from depth data through key poses learning and decision forests*. In proc. SIBGRAPI 2012. Ouro Preto/Brazil, August 2012. (Qualis B1).

Os seguintes trabalhos foram submetidos como versão extendida para revistas indexadas:

- Vieira, Antônio W. & Drews, Paulo J. & Campos, Mario F.M. *Spatial Density Patterns for Efficient Change Detection in 3D Environment for Autonomous Surveillance Robots*. Submitted as invited paper to IEEE Transactions on Automation Science and Engineering (T-ASE). (Qualis A1).

- Vieira, Antonio W. & Nascimento, Erickson R. & Oliveira, Grabriel L. & Liu, Zicheng & Campos, Mario F.M. *On The Improvement of Human Action Recognition from Depth Map Sequences using Spatial Density Patterns*. Submited

as invited paper to Journal of Pattern Recognition Letters, Special Issue on Robust Recognition Methods for Multimodal Interaction. (Qualis A1).

5.3 Trabalhos futuros

As direções para continuação deste trabalho incluem a investigação de formas mais eficientes de construção e armazenamento da representação por densidade de ocupação espacial, bem como o desenvolvimento de novas aplicações para nuvens de pontos usando esta representação. Assim, enumeramos abaixo algumas direções para continuação deste trabalho:

- A implementação atual da nossa representação em volumes implícitos constrói o campo escalar por interpolação de uma amostragem discreta em uma grade $3D$. Embora a continuidade é garantida pela interpolação, esta estratégia depende do armazenamento prévio de valores nos vértices da grade. Ela será eficiente enquanto a questão de memória não for um problema. Entretanto, a fim de evitar o armazenamento em uma grade, pretendemos descrever a densidade global por aprendizagem de um subconjunto de pontos com pesos associados em um processo de otimização para melhor aproximar o campo de densidade desejado. Apesar de ser computacionalmente mais caro, essa abordagem leva a um modelo matemático do campo de densidade que evita o uso de armazenamento na grade.

- A representação por densidade espacial, amostrada em uma grade $3D$, gera uma imagem tridimensional de forma que, adaptando técnicas de construção de descritores locais para imagens $2D$, pode-se investigar a construção de descritores locais para serem usados no reconhecimento de ações usando uma estratégia *bag of features* ou para obter correspondências para alinhamento de nuvens de pontos a partir da sua representação por densidade de ocupação espacial.

- A medida de similaridade que desenvolvemos a partir dos grafos de ação foi usada apenas para comparar as classes e identificar ações similares. Entretanto, a matriz de transição normalizada pode ser usada como representação para as ações e a medida de similaridade usada como um classificador. Uma extensão do nosso trabalho será a modelagem de um classificador automático usando as matrizes de transição e a medida de similaridade. Este classificador dependerá apenas das sequências de índices de poses salientes, que podem ser obtidas dos esqueletos ou diretamente dos mapas de profundidade.

- O reconhecimento de ações em nuvens de pontos, neste trabalho, considera ações descritas por sequências de posturas do corpo sem interação com objetos. Por outro lado, o reconhecimento de objetos em nuvens de pontos é tratado em diversos trabalhos em reconhecimento de padrões. Uma direção futura inclui a utilização de densidade de ocupação espacial para reconhecimento de objetos e reconhecimento de ações descritas pela interação de pessoas com objetos.

- A construção das funções de densidade local usam um *kernel* Gaussiano cujo suporte é ilimitado. Para remover esta restrição, pode-se adaptar um *kernel* de suporte compacto como Wyvill [Wyvill et al., 1986] ou Wendland [Wendland, 1995]. Este tipo de *kernel* tem valor zero em todos os pontos, a menos de um suporte compacto, com a vantagem de que a transição de valores não nulos para valores nulos é feita de forma contínua e suave, diferentemente da abordagem atual onde o *kernel* gaussiano é truncado para um suporte compacto com perda de continuidade e suavidade na transição para valores nulos.

Referências Bibliográficas

Archambeau, C. & Verleysen, M. (2003). Fully nonparametric probability density function estimation with finite gaussian mixture models. Em *Proceedings of the 5th International Conference on Advances in Pattern Recognition (ICAPR 2003)*.

Bai, Y. & Wang, D. (2010). On the comparison of trilinear, cubic spline, and fuzzy interpolation methods in the high-accuracy measurements. *Trans. Fuz Sys.*, 18(5):1016--1022. ISSN 1063-6706.

Besl, P. J. & McKay, N. D. (1992). A method for registration of 3-d shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 14(2):239--256. ISSN 0162-8828.

Biber, P. & Strasser, W. (2003). The normal distributions transform: A new approach to laser scan matching. Em *Proceedings of the IEEE International Conference on Intelligent Robots and Systems (IROS)*, pp. 2743 –2748.

Bloomenthal, J. (1997). *Introduction to Implicit Surfaces*. Morgan Kaufmann Publishers, Inc, San Francisco, California. ISBN 1-558-60233-X.

Brunet, P. & Navazo, I. (1990). Solid representation and operation using extended octrees. *ACM Transactions on Graphics*, 9(2):170--197.

Bucksch, A. K. & Appel van Wageningen, H. (2006). Skeletonization and segmentation of point clouds using octrees and graph theory. Em Maas, H.-G. & Schneider, D., editores, *ISPRS Symposium: Image Engineering and Vision Metrology (IEVM)*, volume XXXVI, pp. 1--6. Dresden University of Technology.

Carr, J. C.; Beatson, R. K.; Cherrie, J. B.; Mitchell, T. J.; Fright, W. R.; McCallum, B. C. & Evans, T. R. (2001). Reconstruction and representation of 3d objects with radial basis functions. Em *Proceedings of conference on Computer graphics and interactive techniques (SIGGRAPH)*, pp. 67--76. ACM.

Chang, C.-C. & Lin, C.-J. (2011). LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1--27:27. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

Cignoni, P.; Montani, C. & Scopigno, R. (1998a). A comparison of mesh simplification algorithms. *Computer & Graphics*, 22(1):37--54.

Cignoni, P.; Rocchini, C. & Scopigno, R. (1998b). Metro: measuring error on simplified surfaces. *Computer Graphics Forum*, 17:167–174.

Curless, B. & Levoy, M. (1996). A volumetric method for building complex models from range images. Em *Proceedings of conference on Computer graphics and interactive techniques (SIGGRAPH)*, pp. 303--312, New York, NY, USA. ACM.

Dasarathy, B. (1991). *Nearest neighbor (NN) norms: nn pattern classification techniques*. IEEE Computer Society Press tutorial. IEEE Computer Society Press. ISBN 9780818659300.

Drews Jr, P.; Núñez, P.; Rocha, R.; Campos, M. & Dias, J. (2010). Novelty detection and 3d shape retrieval using superquadrics and multi-scale sampling for autonomous mobile robots. Em *Proceedings of the International Conference on Robotics and Automation (ICRA)*, pp. 3635--3640.

Elfes, A. (1989). Using occupancy grids for mobile robot perception and navigation. *Computer*, 22(6):46--57. ISSN 0018-9162.

Fan, H.; Yu, Y. & Peng, Q. (2010). Robust feature-preserving mesh denoising based on consistent subneighborhoods. *IEEE Transactions on Visualization and Computer Graphics*, 16(2):312--324. ISSN 1077-2626.

Garland, M. & Heckbert, P. S. (1997). Surface simplification using quadric error metrics. Em *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pp. 209--216, New York, NY, USA. ACM Press/Addison-Wesley Publishing Co.

Geng, J. (2011). Structured-light 3d surface imaging: a tutorial. *Advances in Optics and Photonics*, 3.

Gois, J. P.; Polizelli-Junior, V.; Etiene, T.; Tejada, E.; Castelo, A.; Ertl, T. & Nonato, L. G. (2007). Robust and adaptive surface reconstruction using partition of unity implicits. Em *Proceedings of the XX Brazilian Symposium on Computer Graphics*

and Image Processing, SIBGRAPI '07, pp. 95--104, Washington, DC, USA. IEEE Computer Society.

Gonzálvez, P. R.; Aguilera, D. G. & Lahoz, G. G. (2007). From point cloud to surface: Modeling structures in laser scanner point clouds. Em *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences (ISPRS)*, volume 3, pp. 338--343.

Greengard, L. & Strain, J. (1991). The fast gauss transform. *SIAM J. Sci. Stat. Comput.*, 12(1):79--94.

I. Amorim, R. R. & Dias, J. (2008). Mobile robotic surveillance systems: Detecting and evaluating changes in 3d mapped environments. *ICR*.

Jang, M.-H.; Kim, S.-W.; Faloutsos, C. & Park, S. (2011). A linear-time approximation of the earth mover's distance. Em *Proceedings of the 20th ACM international conference on Information and knowledge management*, CIKM '11, pp. 505--514, New York, NY, USA. ACM.

Johnson, A. (1997). *Spin-Images: A Representation for 3-D Surface Matching*. Tese de doutorado, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA.

Kaestner, R.; Thrun, S.; Montemerlo, M. & Whalley, M. (2005). A non-rigid approach to scan alignment and change detection using range sensor data. Em *FSR*, pp. 179–194.

Kazhdan, M.; Bolitho, M. & Hoppe, H. (2006). Poisson surface reconstruction. Em *Symposium on Geometry processing (SGP)*, pp. 61--70. Eurographics.

Knauer, C.; Löffler, M.; Scherfenberg, M. & Wolle, T. (2011). The directed Hausdorff distance between imprecise point sets. *Theoretical Computer Science*, 412(32):4173--4186. ISSN 0304-3975.

Levin, D. (2003). Mesh-independent surface interpolation. *Geometric Modeling for Scientific Visualization*, pp. 37--49.

Lewiner, T.; Lopes, H.; Vieira, A. W. & Tavares, G. (2003). Efficient implementation of marching cubes cases with topological guarantees. *Journal of Graphics Tools*, 8(2):1--15.

Li, W.; Zhang, Z. & Liu, Z. (2008). Expandable data-driven graphical modeling of human actions based on salient postures. *IEEE Trans. on Circuits and Systems for Video Technology*, 18(11).

Li, W.; Zhang, Z. & Liu, Z. (2010). Action recognition based on a bag of 3d points. Em *CVPR Workshop for Human Communicative Behavior Analysis*.

Liu, Z. (2011). MSR action recognition datasets and codes. url: http://research.microsoft.com/en-us/um/people/zliu/actionrecorsrc.

Lorensen, W. E. & Cline, H. E. (1987). Marching cubes: A high resolution 3d surface construction algorithm. Em *Proceedings of the 14th annual conference on Computer graphics and interactive techniques*, pp. 163--169. ACM Press.

Macêdo, I.; Gois, J. P. & Velho, L. (2011). Hermite radial basis functions implicits. *Computer Graphics Forum*, 30(1):27–42.

Magnusson, M.; Lilienthal, A. & Duckett, T. (2007). Scan registration for autonomous mining vehicles using 3D-NDT: Research articles. *J. Field Robot.*, 24(10):803--827. ISSN 1556-4959.

Mäntylä, M. (1988). *An Introduction to Solid Modeling*. Computer Science Press, Inc. ISBN 0-88175-108-1.

Mederos, B.; Lage, M.; Arouca, S.; Petronetto, F.; Velho, L.; Lewiner, T. & Lopes, H. (2007). Regularized implicit surface reconstruction from points and normals. *Journal of the Brazilian Computer Society*, 13:7 – 16. ISSN 0104-6500.

Miranda, L.; Vieira, T.; Martinez, D.; Lewiner, T.; Vieira, A. W. & Campos, M. F. M. (2012). Real-time gesture recognition from depth data through key poses learning and decision forests. Em *Sibgrapi 2012 (XXV Conference on Graphics, Patterns and Images)*, Ouro Preto, MG. IEEE.

Nascimento, E. R.; Oliveira, G. L.; Campos, M. F. M.; Vieira, A. W. & Schwartz, W. R. (2012). Brand: A robust appearance and depth descriptor for RGB-D images. Em *Proceedings of the IEEE International Conference on Intelligent Robots and Systems (IROS)*.

Núñez, P.; Drews, P.; Bandera, A.; Rocha, R.; Campos, M. & Dias, J. (2010). Change detection in 3d environments based on gaussian mixture model and robust structural matching for autonomous robotic applications. Em *Proceedings of the IEEE International Conference on Intelligent Robots and Systems (IROS)*, pp. 2633 –2638. ISSN 2153-0858.

Núñez, P.; Drews, P.; Rocha, R.; Campos, M. & Dias, J. (2009). Novelty detection and 3d shape retrieval based on gaussian mixture models for autonomous surveillance

robotics. Em *Proceedings of the IEEE International Conference on Intelligent Robots and Systems (IROS)*, pp. 4724--4730.

Ohtake, Y.; Belyaev, A.; Alexa, M.; Turk, G. & Seidel, H.-P. (2003). Multi-level partition of unity implicits. Em *Proceedings of conference on Computer graphics and interactive techniques (SIGGRAPH)*, pp. 463--470. ACM.

Oliveira, G. L.; Nascimento, E. R.; Vieira, A. W. & Campos, M. F. M. (2012). Sparse spatial coding: A novel approach for efficient and accurate object recognition. Em *Proceedings of the International Conference on Robotics and Automation (ICRA)*, pp. 2592--2598, St. Paul, MN.

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 Recogn.*, 39(7):1346--1358. ISSN 0031-3203.

Pauling, F.; Bosse, M. & Zlot, R. (2009). Automatic segmentation of 3d laser point clouds by ellipsoidal region growing. Em *Proceedings of Australasian Conference on Robotics and Automation (ACRA)*.

Planitz, B. M.; Maeder, A. J. & Williams, J. A. (2005). The correspondence framework for 3d surface matching algorithms. *Computer Vision and Image Understanding (CVIU)*, 97(3):347--383. ISSN 1077-3142.

Rabiner, L. R. (1990). Readings in speech recognition. capítulo A tutorial on hidden Markov models and selected applications in speech recognition, pp. 267--296. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

Raptis, M.; Kirovski, D. & Hoppe, H. (2011). Real-time classification of dance gestures from skeleton animation. Em *Symposium on Computer Animation*, pp. 147--156. ACM.

Reyes, M.; Domínguez, G. & Escalera, S. (2011). Feature weighting in dynamic time warping for gesture recognition in depth data. Em *ICCV Workshop on Consomer Depth Cameras for Computer Vision*.

Ricci, A. (1973). A constructive geometry for computer graphics. *The Computer Jornal*, 16(2):157--160.

Rubner, Y.; Tomasi, C. & Guibas, L. J. (1998). A metric for distributions with applications to image databases. Em *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pp. 59--66.

Rusinkiewicz, S. & Levoy, M. (2001). Efficient variants of the ICP algorithm. Em *Third International Conference on 3D Digital Imaging and Modeling (3DIM)*.

Schaefer, S. & Warren, J. (2004). Dual marching cubes: Primal contouring of dual grids. Em *Proceedings of the Computer Graphics and Applications, 12th Pacific Conference*, pp. 70--76, Washington, DC, USA. IEEE Computer Society.

Scott, D. (1992). *Multivariate Density Estimation: Theory, Practice, and Visualization*. Wiley series in probability and mathematical statistics: Applied probability and statistics. John Wiley & Sons. ISBN 9780471547709.

Sharf, A.; Lewiner, T.; Shamir, A. & Kobbelt, L. (2007). On-the-fly curve-skeleton computation for 3d shapes. Em *Eurographics 2007 (Computer Graphics Forum)*, volume 26, pp. 323--328, Prague. Eurographics.

Shirdhonkar, S. & Jacobs, D. W. (2008). Approximate earth mover's distance in linear time. Em *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE Computer Society.

Shotton, J.; Fitzgibbon, A.; Cook, M.; Sharp, T. & Finocchio, M. (2011). Real-time human pose recognition in parts from single depth images. Em *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 224--231.

Sprenger, T. C.; Brunella, R. & Gross, M. (2000). H-Blob: a hierarchical visual clustering method using implicit surfaces. Em *Proceedings of Visualization*, pp. 61--68. IEEE.

Stoyanov, T.; Magnusson, M.; Almqvist, H. & Lilienthal, A. J. (2011). On the Accuracy of the 3D Normal Distributions Transform as a Tool for Spatial Representation. Em *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*.

Tagliasacchi, A.; Zhang, H. & Cohen-Or, D. (2009). Curve skeleton extraction from incomplete point cloud. *ACM Trans. Graph.*, 28(3):71:1--71:9.

Taubin, G. (1991). Estimation of planar curves, surfaces and nonplanar space curves defined by implicit equations, with applications to edge and range image segmentation. *In IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Thrun, S.; Burgard, W. & Fox, D. (2005). *Probabilistic Robotics*. MIT Press. ISBN 0262201623.

Toledo, R. & Levy, B. (2008). Visualization of industrial structures with implicit gpu primitives. Em *Proceedings of the 4th International Symposium on Advances in Visual Computing (ISVC)*, pp. 139--150, Berlin, Heidelberg. Springer-Verlag.

Tombari, F.; Salti, S. & di Stefano, L. (2011). A combined texture-shape descriptor for enhanced 3d feature matching. Em *Proceedings of Internation Conference on Image Processing (ICIP)*, pp. 809–812.

Vieira, A. W.; Alves Neto, A.; Guimarães Macharet, D. & Campos, M. F. M. (2010). Mesh denoising using quadric error metric. Em *XXIII Brazilian Symposium on Computer Graphics and Image Processing (SIBGRAPI'10)*, Gramado, RS, Brazil.

Vieira, A. W.; Lewiner, T.; Schwartz, W. & Campos, M. F. M. (2012). Distance matrices as invariant features for classifying mocap data. Em *21st International Conference on Pattern Recognition (ICPR)*, Tsukuba Science City, Japan. IEEE.

Vieira Neto, H. & Nehmzow, U. (2008). Visual novelty detection for autonomous inspection robots. Em *Service Robot Applications*, pp. 309–330. I-Tech.

Viterbi, A. (2006). Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Trans. Inf. Theor.*, 13(2):260--269. ISSN 0018-9448.

Weinland, D.; Ronfard, R. & Boyer, E. (2010). A survey of vision-based methods for action representation, segmentation and recognition. *Computer Vision and Image Understanding (CVIU)*.

Wendland, H. (1995). Piecewise polynomial, positive definite and compactly supported radial functions of minimal degree. *Advances in Computational Mathematics*, pp. 389–396.

Wyvill, G.; McPheeters, C. & Wyvill, B. (1986). Data structure for soft objects. *The Visual Computer*, 2(4):227--234.

Xujia Qin, W. W. & Li, Q. (2006). Practical boolean operations on point-sampled models. *Computational Science and Its Applications - ICCSA 2006*, 3980:393–401.

Zheng, C. & Zhang, H. (2011). Implicit surface reconstruction based on adaptive clustering. Em *Proceedings of International Conference on Computer-Aided Design and Computer Graphics (CADGRAPHICS)*, pp. 509--515. IEEE.