UNIVERSIDADE FEDERAL DO PAMPA

Sherlon Almeida da Silva

Recuperação de Objetos 3D Baseada em Relações Geométricas entre *Surflets*

Alegrete
2018

Sherlon Almeida da Silva

Recuperação de Objetos 3D Baseada em Relações Geométricas entre *Surflets*

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Ciência da Computação da Universidade Federal do Pampa como requisito parcial para a obtenção do título de Bacharel em Ciência da Computação.

Orientador: Prof. Dr. Marcelo Resende Thielo

Alegrete
2018

Sherlon Almeida da Silva

Recuperação de Objetos 3D Baseada em Relações Geométricas entre *Surflets*

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Ciência da Computação da Universidade Federal do Pampa como requisito parcial para a obtenção do título de Bacharel em Ciência da Computação.

Trabalho de Conclusão de Curso defendido e aprovado em 06 de dezembro de 2018
Banca examinadora:

Prof. Dr. Marcelo Resende Thielo
Orientador
UNIPAMPA

Prof. Me. Diego Luis Kreutz
UNIPAMPA

Prof. Me. Eliezer Soares Flores
UNIPAMPA

Dedico este trabalho àqueles que, assim como eu, abriram mão de certas coisas para correr atrás de seus sonhos. E aos ainda indecisos dedico estas palavras, façam escolhas que os fazem bem, e a felicidade irá alcançá-los.

AGRADECIMENTOS

Agradeço e dedico este trabalho especialmente aos meus pais e irmãos pelo apoio, sem eles nunca teria conseguido chegar até aqui. Principalmente, pela compreensão por eu não poder estar frequentemente presente por estar muito envolvido com a faculdade.

Agradeço também à namorada/colega/amiga Isadora Ferrão pelo companheirismo durante esta trajetória, somente nós podemos saber quantas noites sem dormir fazendo trabalhos ou estudando para provas, sempre um ajudando o outro nos momentos mais difíceis. Muito obrigado!

É importante valorizar os colegas que se mostraram dispostos a ajudar quando encontrei dificuldades, com isto agradeço a todos que contribuíram de alguma forma. Em especial, gostaria muito de agradecer ao colega/amigo Rodrigo Bisso, pois sempre esteve presente e disposto a ajudar.

Além disso, agradeço ao professor Claudio Schepke pelas oportunidades e portas abertas, foi muito importante para minha formação acadêmica e com certeza utilizarei tudo o que foi aprendido da melhor forma, sempre sendo grato pelas oportunidades.

Agradeço também ao professor Marcelo Thielo pela orientação e pelo incentivo que tive do início ao fim deste trabalho, meu encanto por esta área de pesquisa começou lá em Computação Gráfica graças ao senhor, muito obrigado mesmo!

Para finalizar, gostaria de deixar um sincero e abrangente obrigado a todos que me apoiaram durante este caminho, e também, a todos os professores que compartilharam seus conhecimentos.

"Nunca se esqueça de quem é, porque é certo que o mundo não se lembrará. Faça disso sua força. Assim, não poderá ser nunca a sua fraqueza. Arme-se com esta lembrança, e ela nunca poderá ser usada para magoá-lo."— Tyrion Lannister

RESUMO

A queda do custo de câmeras digitais com o tempo tornou seu acesso mais fácil, o que popularizou sua utilização. Isto fez crescer o número de imagens na *internet*, demandando o estudo de técnicas eficientes para recuperação de informação em imagens. Analogamente, o mesmo está acontecendo com as câmeras RGB-D, uma vez que outros recursos passaram a ser disponibilizados por estas câmeras, como, por exemplo, a percepção tridimensional (3D) do ambiente, e até mesmo a possibilidade de realizar capturas no escuro. Isto traz novas aplicações, sendo estas a partir de uma percepção 3D do mundo, como mapeamento de ambientes, escaneamento de objetos, impressão 3D, entre outros. Neste momento, semelhante ao ocorrido com as imagens, o número de objetos 3D está crescendo, tornando necessários algoritmos que encontrem padrões em tais objetos para fácil recuperação por similaridade. A partir disso, este trabalho investiga técnicas de extração de características de objetos 3D baseadas em relações geométricas entre seus polígonos, e também avalia métricas de comparação entre seus histogramas. Além disso, este trabalho apresenta uma proposta para extração de características e comparação dos objetos 3D. A validação das métricas investigadas e propostas deu-se a partir de quatro bases de objetos selecionados e classificados em grupos. Os resultados apontam que a relação geométrica intrínsica a cada objeto, proposta neste trabalho, oferece características únicas que podem ser utilizadas para recuperação de objetos 3D por similaridade.

Palavras-chave: Recuperação de objetos 3D. Câmeras RGB-D. Similaridade entre objetos 3D. Recuperação 3D baseada em formas.

ABSTRACT

The fall in the cost of digital cameras over time made their access easier, which made their use more popular. This increased the number of images on the Internet, demanding the study of efficient techniques for information retrieval in images. Analogously, the same is happening with RGB-D cameras, since other features are now made available by these cameras, such as three-dimensional (3D) perception of the environment, and even the possibility of capturing in the dark. This brings new applications, these being from a 3D perception of the world, such as environment mapping, object scanning, 3D printing, among others. At this time, similar to what happened with the images, the number of 3D objects is growing, making algorithms that find patterns in such objects necessary for easy retrieval by similarity. From this, this work investigates techniques for extracting features of 3D objects based on geometric relations between their polygons, and also evaluates comparison metrics among their histograms. In addition, this work presents a proposal for extraction of features and comparison of 3D objects. The validation of the metrics investigated and proposed was based on four bases of objects selected and classified into groups. The results indicate that the intrinsic geometric relation to each object, proposed in this work, offers unique features that can be used to retrieve 3D objects by similarity.

Key-words: *3D object retrieval. RGB-D cameras. 3D object similarity. 3D shape-based retrieval.*

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS

2D 2 dimensões

3D 3 dimensões

4D 4 dimensões

BRISK *Binary Robust Invariant Scalable Keypoints*

CNN *Convolutional Neural Networks*

CoSPAIR *Colored SPAIR*

DPD Distância de Programação Dinâmica

GPCNN *Group-Pair Convolutional Neural Networks*

HOG *Histograms of Oriented Gradients*

LDA *Latent Dirichlet Allocation*

ORB *Oriented FAST, rotated BRIEF*

RGB *Red-Green-Blue*

RGB-D *Red-Green-Blue-Depth*

SIFT *Scale Invariant Feature Transform*

SPAIR *Histograms of Spatial Concentric Surflet-Pairs*

Surflets Pontos 3D com coordenadas espaciais + Vetor Normal ao plano

LISTA DE SÍMBOLOS

α	Subcaracterística Alfa
β	Subcaracterística Beta
δ	Subcaracterística Delta
ϵ	Subcaracterística Epsilon
γ	Subcaracterística Gama
C	Centro da Esfera
$d1, d2$	Distância a Partir do Ponto O
$H_{o'}$	Histograma do Objeto de Entrada Para Consulta
H_o	Histograma do Objeto da Base de Objetos
$hits$	Número de Acertos Dentro da Classe
l	Vetor Que Dá a Direção da Reta (Vetor Normal ao Triângulo)
Nt	Número Total de Objetos da Classe
O	Ponto de Partida da Reta
r	Raio da Esfera
$S_{o'}$	Conjunto de Características do Objeto O
Tx	Taxa de Acerto do Objeto
X	Pontos na Esfera e na Reta (Ponto Médio do Triângulo)
S	Conjunto de Todas as Características de Um Objeto
card	Cardinalidade de Um Conjunto
d	Número Total de Caixas do Histograma
H	Histograma
h	Mapeamento das Características nas Caixas do Histograma
i	Índice da Caixa do Histograma do Objeto
O	Objeto
S	Característica Extraída

SUMÁRIO

1 INTRODUÇÃO

Os princípios de sobrevivência dos seres vivos são associados às necessidades de encontrar alimento, água e segurança. Estes são instintos naturais, os quais até mesmo o ser no topo da cadeia alimentar busca. Para isso, é importante aos seres vivos compreenderem o ambiente onde estão inseridos e, desta forma, manipulá-lo à sua maneira para garantir seu conforto e segurança.

A matemática e a física são áreas do conhecimento que possibilitam ao ser humano compreender e simular o comportamento da natureza, sendo a natureza o ambiente no qual o ser humano está inserido. A utilização destas e outras áreas do conhecimento, juntamente com o avanço tecnológico, possibilitam alcançar resultados em pesquisas científicas mais rapidamente, podendo assim validar teorias através de métodos numéricos. Desta forma, os computadores facilitam a vida de pesquisadores ao redor do mundo, realizando cálculos complexos para a representação computacional da natureza.

Como mencionado, a computação é aplicada em outras áreas para gerar conhecimento sobre a natureza. Modelos biológicos são exemplos chave para pesquisadores, uma vez que a compreensão destes modelos pode ser utilizada em diversas aplicações do cotidiano da sociedade. O frio e o calor, a proximidade entre objetos, a visão, a audição, o olfato, o tato, as sinapses neurais, a locomoção e o relacionamento entre indivíduos de uma espécie, são fatores relevantes em aplicações que demandam a interação com o ambiente, uma vez que são "entradas" a serem processadas pelo cérebro.

Para este fim, é necessário definir um conjunto de sensores para as máquinas serem capazes de detectar esses fatores biológicos como "entradas", assim como existem na natureza. Por exemplo, o cérebro humano faz a amostragem do ambiente à sua volta através de sensores periféricos, onde cada um deles capta formas diferentes de energia e substâncias químicas. A partir disso, os sinais são vistos como impulsos neurais que são transmitidos para o sistema nervoso central, onde são processados simultaneamente para criar uma representação sensorial do mundo externo (THESEN et al., 2004).

Com isto, existem diversos sensores para estabelecer a conexão máquina-ambiente e prover esta interação. Sensores térmicos, câmeras, microfones, sensores de proximidade, entre outros, são utilizados para tal propósito. A partir da evolução tecnológica, foi possível a automatização de tarefas e o aumento da produtividade na indústria com a utilização de máquinas que utilizam estes sensores.

Com este avanço, os computadores tornaram-se capazes de realizar tarefas repetitivas e exaustivas com precisão. Sendo assim, a programação de um sistema computacional inteligente para cumprir determinado objetivo se tornou cada vez mais comum e mais conveniente, incluindo por exemplo, a utilização de carros autônomos, robôs anti-bomba, drones de mapeamento, *chat bots* e mecanismos de busca eficientes em grandes volumes de dados, dentre várias outras aplicações.

Porém, máquinas só fazem aquilo que são programadas para fazerem e muitas apli-

cações do mundo real demandam que elas "enxerguem" o mundo. Para isto, são utilizadas câmeras digitais e sensores *Red-Green-Blue-Depth* (RGB-D), que podem ser encontrados no dia-a-dia das pessoas, nos celulares, nos tablets e nos sensores de movimento de jogos. Estes sensores são utilizados como coletores de informação para que a máquina a processe e identifique padrões (MONICA; ALEOTTI; CASELLI, 2016).

Existem vários mecanismos de busca de informações em imagens atualmente. Porém, muitas vezes, imagens não são capazes de descrever o ambiente em um aspecto tridimensional, como os seres humanos estão habituados (NAPOLÉON; SAHBI, 2010). Isto é relevante para a compreensão do ambiente, como por exemplo, para o deslocamento de robôs e aplicações como reconstrução de ambientes em 3 dimensões (3D), bem como para a simulação física da natureza.

Com isto, e com a popularização de câmeras RGB-D nos últimos anos, houve um aumento do uso de modelos tridimensionais e a computação gráfica ganhou maior visibilidade na indústria cinematográfica, de jogos digitais, de modelagem de cenas para arquitetura e engenharia, impressão 3D, dentre outras áreas (LOGOGLU; KALKAN; TEMIZEL, 2016). No entanto, com o aumento da quantidade de objetos 3D disponíveis e com a popularização de câmeras RGB-D e impressoras 3D, tornam-se necessários algoritmos de recuperação de objetos 3D capazes de prover consultas rápidas e precisas.

Levando em consideração a necessidade de novos algoritmos para este fim, este trabalho objetiva aprofundar os estudos na área de visão computacional. Especificamente, são investigadas técnicas de extração de características de objetos 3D baseadas em relações geométricas entre seus polígonos e avaliadas métricas de comparação entre seus histogramas. Além disso, este trabalho também apresenta uma proposta para extração de características e comparação de objetos 3D para recuperação por similaridade.

Este capítulo apresenta, na seção 1.1, a motivação que levou ao desenvolvimento deste trabalho. Na seção 1.2, são apresentados os objetivos gerais e específicos. Na seção 1.3, é descrito brevemente como o trabalho foi desenvolvido. Por fim, a seção 1.4 apresenta a organização deste trabalho e visão geral dos demais capítulos.

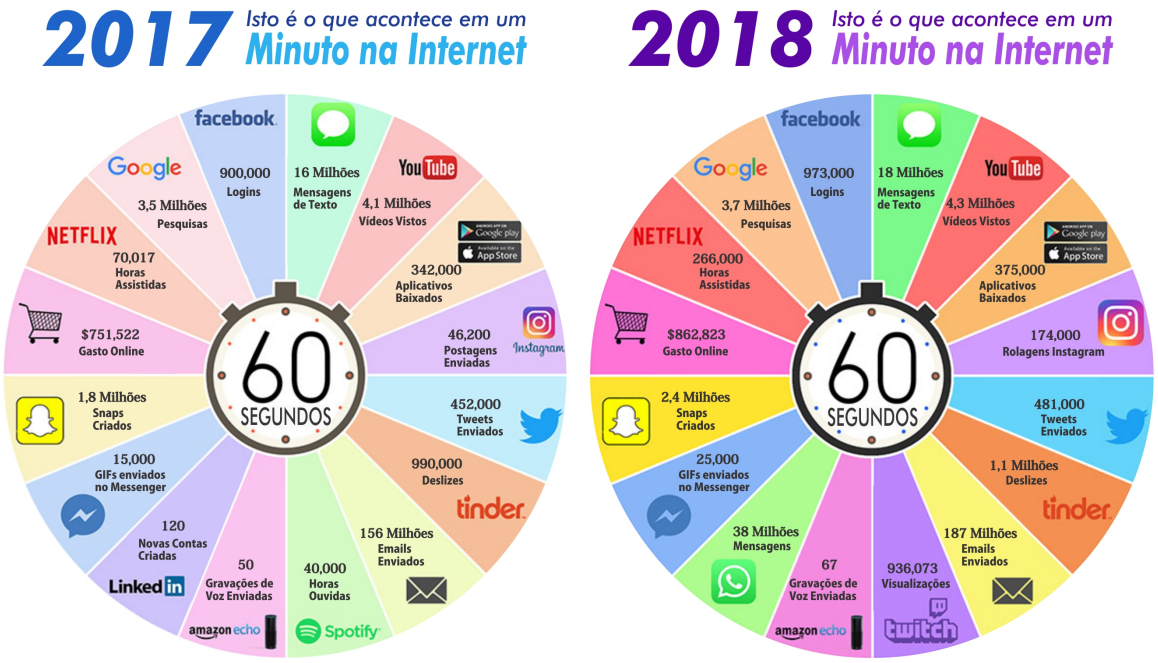
1.1 Motivação

Atualmente, existe maior facilidade em obter recursos tecnológicos do que havia alguns anos atrás, quando estes eram exclusividade de grandes centros de pesquisa. Hoje, grande parte da população mundial possui dispositivos eletrônicos com câmeras *Red-Green-Blue* (RGB), sendo estas um recurso muito popular para compartilhamento de fotos em redes sociais, entre outras aplicações. À medida em que esta tecnologia se popularizou, um aumento do volume de imagens na rede levou à necessidade do desenvolvimento de algoritmos capazes de realizarem buscas eficientes em grandes volumes de dados.

Foram realizados levantamentos de estatísticas do fluxo de dados e consultas na *internet* dos últimos três anos em (LEBOEUF, 2016; DESJARDINS, 2017; DESJAR-

DINS, 2018). Com as informações providas por estes autores, é possível observar o que acontece na *internet* em apenas um minuto. A Figura 1 apresenta os dados de 2017 e 2018 lado a lado para gerar uma comparação.

Figura 1 – O Que Acontece Em Um Minuto Na *Internet*?



Fonte: Adaptado de (DESJARDINS, 2018)

É possível observar que o uso de sistemas que utilizam imagens e vídeos, como redes sociais (i.e, Instagram, Facebook e Snapchat), Netflix e Youtube, tem crescido. Consequentemente, o volume de dados na rede tem aumentado, o que está diretamente relacionado à popularização de câmeras RGB. De forma análoga, as câmeras RGB-D estão se popularizando, sendo utilizadas para mapeamento de ambientes, escaneamento de objetos 3D, reconhecimento facial, entre outras aplicações, o que está impactando no aumento da quantidade de objetos 3D e possibilitando um horizonte amplo de aplicações.

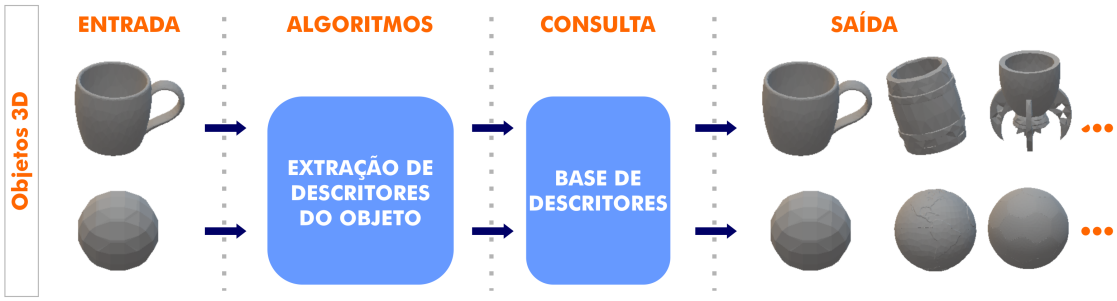
1.2 Objetivos

Este trabalho tem por objetivo geral propor um novo algoritmo de extração de descritores e recuperação de objetos 3D. A visão geral desse algoritmo é descrita na Figura 2, a qual ilustra as etapas do algoritmo proposto.

Basicamente, são inseridos objetos de entrada para consulta, e o algoritmo extrai as características destes objetos para comparação em uma base de descritores. Os objetos são representados por estes descritores, que são relações representativas/descritivas entre

seus polígonos[1]. Por fim, é realizada uma busca na base de descritores à procura de objetos com características parecidas, para, então, retornar os objetos similares à entrada fornecida.

Figura 2 – Exemplo de Consulta.



Fonte: Elaborado pelo autor.

São definidos como objetivos específicos os tópicos a seguir:

- Investigar algoritmos para extração de descritores e recuperação de objetos 3D;
- Implementar um algoritmo existente no estado da arte para compreendê-lo e realizar um *benchmarking*[2] com o algoritmo proposto neste trabalho;
- Obter e criar bases de objetos 3D variados para validação do algoritmo do estado da arte e do algoritmo proposto;
- Validar o algoritmo do estado da arte e o algoritmo proposto para as bases de objetos 3D criadas.

1.3 Metodologia

A seguir são enumerados os tópicos destacados como importantes para atingir os objetivos deste trabalho:

1. **Definir tema do trabalho:** avaliar tópicos na área de visão computacional e, a partir disso, definir a linha de pesquisa;
2. **Explorar o estado da arte:** pesquisar trabalhos que se enquadram na linha de pesquisa deste trabalho;
3. **Avaliar trabalhos relacionados:** qualificar e classificar os trabalhos relacionados em grau de relevância, para utilizá-los como referência neste trabalho;

[1] Polígonos são relações entre vértices e arestas de um objeto, as quais formam planos (i.e, Triângulos).
[2] *Benchmarking* entre os algoritmos significa prover uma comparação equivalente entre eles.

4. **Estudar conceitos:** compreender os principais conceitos que são a base para os trabalhos relacionados e identificar quais técnicas podem contribuir para o desenvolvimento deste trabalho;

5. **Desenvolver o protótipo:** a partir das lições aprendidas, desenvolver um protótipo baseado em um algoritmo do estado da arte e validá-lo;

6. **Obter base de objetos:** investigar objetos disponíveis na rede, para montar uma base de objetos robusta. Além disso, criar um conjunto de objetos regulares básicos (i.e, cubos, esferas, etc) e também montar um conjunto de objetos variados (i.e, pessoas, animais, objetos, etc);

7. **Desenvolver o algoritmo:** dar continuidade à implementação do protótipo, desenvolvendo uma versão completa do algoritmo do estado da arte e do proposto;

8. **Realizar testes:** validar ambos algoritmos com as bases de objetos criadas/obtidas;

9. **Registrar trabalho:** avaliar as lições estudadas e aprendidas neste trabalho, registrando os conceitos, técnicas utilizadas, validação da proposta final e os resultados obtidos.

1.4 Organização do Documento

1. Capítulo 2 - Fundamentação Teórica: são apresentados conceitos essenciais para o desenvolvimento deste trabalho, como tendências consolidadas e atuais na área de visão computacional, e a base teórica necessária para seu entendimento e utilização em aplicações científicas.

2. Capítulo 3 - Trabalhos Relacionados: são apresentados os métodos de procura por trabalhos relacionados e apresentados os trabalhos mais relevantes para o contexto deste trabalho.

3. Capítulo 4 - Desenvolvimento: é apresentado o que foi feito para atingir os resultados deste trabalho.

4. Capítulo 5 - Resultados Obtidos: são apresentados os resultados dos testes realizados com os algoritmos implementados.

5. Capítulo 6 - Considerações Finais: apresenta as considerações finais da pesquisa realizada, apontando os principais tópicos observados.

2 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo é apresentada a fundamentação teórica para a compreensão deste trabalho, e sua organização é a seguinte: na seção 2.1 é apresentada a área de visão computacional. Na seção 2.2 são apresentados conceitos e utilização de câmeras com 2 dimensões (2D) e 3D atuais. Em seguida, conceitos de extração de características 2D e 3D são apresentados na seção 2.3. E, por fim, na seção 2.4 são apresentados alguns critérios de recuperação de objetos através da similaridade.

2.1 Visão Computacional

A visão computacional é uma área do conhecimento que estuda a visão dos seres vivos e tenta replicar tais conceitos para utilização em máquinas. Está associada à diversos segmentos de pesquisa dentro da inteligência artificial e da biologia, onde pesquisadores tentam descobrir como o cérebro detecta padrões em imagens e objetos.

Existem diversas linhas de pesquisa dentro da ciência da computação que geram esforços para processar imagens e objetos, como exemplos principais existem as áreas de processamento de imagens, computação gráfica e aprendizado de máquina. Estas áreas de pesquisa são aplicadas em conjunto em aplicações de robótica.

Ainda dentro de visão computacional é importante distinguir o significado de alguns termos[3], como os a seguir:

- **Classificação:** Trata da associação de imagens ou objetos 3D semelhantes à uma classe previamente rotulada, como por exemplo: gato, cachorro, pessoa.

- **Detecção:** Trata da análise de características em uma imagem ou cena 3D a fim de encontrar/detectar objetos ali existentes e onde eles estão localizados.

- **Reconhecimento ou Identificação:** Trata da busca pelo domínio do objeto detectado em uma imagem ou cena 3D, com o intuito de nomeá-lo, saber o que ele é, por exemplo: cachorro 97%, gato 83%, pessoa 89%.

- **Segmentação:** Trata da divisão da imagem ou cena 3D em regiões pertencentes a um mesmo objeto.

- **Recuperação:** Trata da coleta de informação de imagens ou objetos 3D para então usar métricas de comparação por semelhança/similaridade. Exemplo: fotos distintas da torre Eiffel possuem características parecidas.

[3] Algumas dessas definições podem se sobrepor total ou parcialmente com as demais, uma vez que estão correlacionadas.

2.2 Câmeras RGB e Câmeras RGB-D

Câmeras são sensores que obtém imagens ou objetos de cenários reais. Uma imagem é uma projeção do cenário em um plano bidimensional, e um objeto é a projeção do cenário no espaço tridimensional, provendo também a profundidade na cena.

Uma vez que imagens não possuem profundidade, câmeras RGB podem simular uma percepção 3D a partir da fotografia de diversos pontos de vista de um mesmo objeto. Esta técnica é chamada de fotogrametria, e identifica os padrões a partir da iluminação da cena e remonta o objeto em 3D utilizando um *software* de reconstrução. Porém, a utilização de fotogrametria fica limitada em alguns quesitos, pois, é imprecisa em objetos com reflexo e em materiais como vidro e metal por questões de ótica e dispersão da luminosidade. Outra limitação é a necessidade de iluminação, uma vez que utiliza fotografias no espectro visível, inviabilizando escaneamento à noite.

Há também o escaneamento 3D com câmeras RGB-D, sendo alguns modelos baseados na projeção de uma nuvem de pontos a partir de um sensor infravermelho, o qual possibilita obter as coordenadas tridimensionais da cena. Assim, é possível coletar informações da cena e escanear objetos mesmo no escuro, diferentemente da fotogrametria que utiliza a luminosidade para destacar as características dos objetos.

A partir destas vantagens em sensores 3D, sua utilização tem aumentado (ALDOMA et al., 2012). A obtenção de amostras 3D de um objeto/ambiente é muito utilizada atualmente em diversos campos. Por exemplo, pode ser usado na robótica para o mapeamento de ambientes por robôs (FILLIAT et al., 2012), no reconhecimento de objetos em uma cena (SONG; LICHTENBERG; XIAO, 2015), na extração de informações do ambiente por carros autônomos (HIMMELSBACH; LUETTEL; WUENSCHE, 2009), assim como na modelagem tridimensional para cinema, jogos, realidade virtual e aumentada e impressão 3D (LÜ; HE; XUE, 2009). Sua vantagem de utilização é que com a terceira dimensão é possível obter mais detalhes da cena/objeto do que exclusivamente em imagens 2D. Por exemplo, com a utilização de câmeras RGB-D, obtém-se a noção de profundidade da cena, escala e posição dos objetos, além de possibilitar visão noturna.

2.2.1 Microsoft Kinect e Impressão 3D

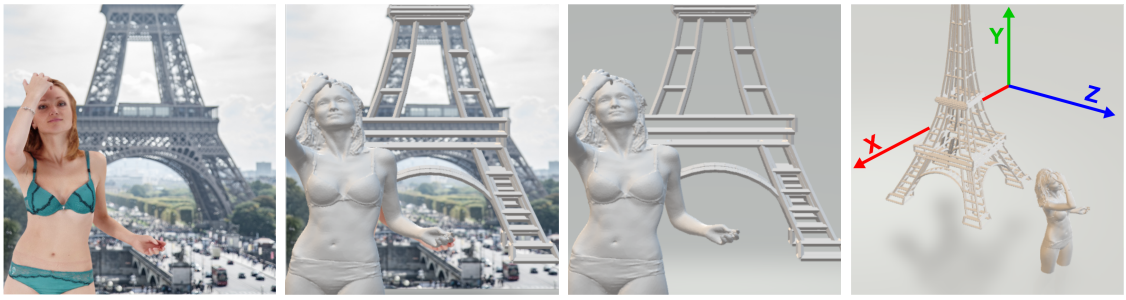
Um exemplo de sensor RGB-D popular é o Kinect da Microsoft (MICROSOFT, 2014). Uma das suas aplicações, por exemplo, é na impressão 3D, onde servem para escanear modelos de objetos reais em 3D para serem gerados a partir de impressoras 3D (i.e, peças industriais). Neste contexto, as impressoras 3D funcionam de forma análoga às impressoras convencionais. Por exemplo, as impressoras convencionais imprimem em folhas, que são planos 2D. Já as impressoras 3D imprimem objetos em 3D.

Nos últimos anos vem crescendo o uso desse tipo de tecnologia, a qual tem dominado muitos mercados, uma vez que está ganhando mais espaço a cada dia. Diversas

aplicações estão fazendo uso de impressão 3D, como a medicina, a área militar, a área industrial e até a área gastronômica.

Como um exemplo ilustrativo observemos a Figura 3. Primeiramente um objeto é escaneado na cena a partir da nuvem de pontos gerados pelo infravermelho e renderizado pelo computador como um conjunto de polígonos. Após isso, pode ser editado com *softwares* de manipulação de objetos 3D e, por fim, ser impresso.

Figura 3 – Exemplo de Utilização de Kinect + Impressão 3D.



(a) Cena. (b) Nuvem de Pontos. (c) Objeto Obtido. (d) Impressão 3D.

Fonte: Elaborado pelo autor.

2.2.2 3D Builder e 3D Scan

3D Builder é um *software* desenvolvido pela Microsoft para edição de modelos de objetos 3D para impressão. Este *software* conta com o 3D Scan para gerar objetos escaneados com o Kinect. Desta forma, a utilização do Kinect e edição de modelos 3D, assim como a impressão, tornam-se mais próximas das pessoas, uma vez que todas pessoas com acesso a um computador podem usufruir destes sistemas gratuitamente. Além disso existe o Paint, ferramenta bastante conhecida por acompanhar o Windows, que ganhou uma versão 3D na qual é possível importar objetos tridimensionais para edição.

A utilização do 3D Scan junto ao 3D Builder contempla grande parte do trabalho na etapa de obtenção de modelos tridimensionais escaneados do mundo real. Eles são o passo intermediário, uma vez que o passo inicial (entrada) é feito pelo Kinect, e o passo final (saída) é feito pela impressora 3D. Ressaltando que apenas o escaneamento com o Kinect já apresenta diversas aplicações via *software*. Por exemplo, na robótica, a saída é a interação do robô com o ambiente a partir do que ele aprendeu com o escaneamento.

2.3 Extração de Características 2D Vs 3D

Imagens e objetos apenas são modelos representados por pixels e polígonos, respectivamente. Somente possuem algum significado para os seres humanos, sendo que para o computador são apenas dados brutos. Para que aplicações sejam desenvolvidas no

campo de visão computacional, imagens e objetos devem ter características particulares que possam ser utilizadas como informação.

Existem diversas técnicas que extraem descritores/características de imagens e objetos 3D, tais como: *Oriented FAST, rotated BRIEF* (ORB), *Scale Invariant Feature Transform* (SIFT), *Binary Robust Invariant Scalable Keypoints* (BRISK), *Surflet-Pair Relation*, entre outras.

2.3.1 Representação de Imagens 2D e Objetos 3D

As câmeras RGB projetam uma imagem em duas dimensões (2D) que discretizam o ambiente contínuo através de pixels. Estes pixels são pontos no plano que possuem valores de intensidade de cores primárias RGB que variam de 0 a 255. Estes valores são armazenados na memória do computador, o qual apresenta tais dados no monitor para gerar uma visualização ao ser humano.

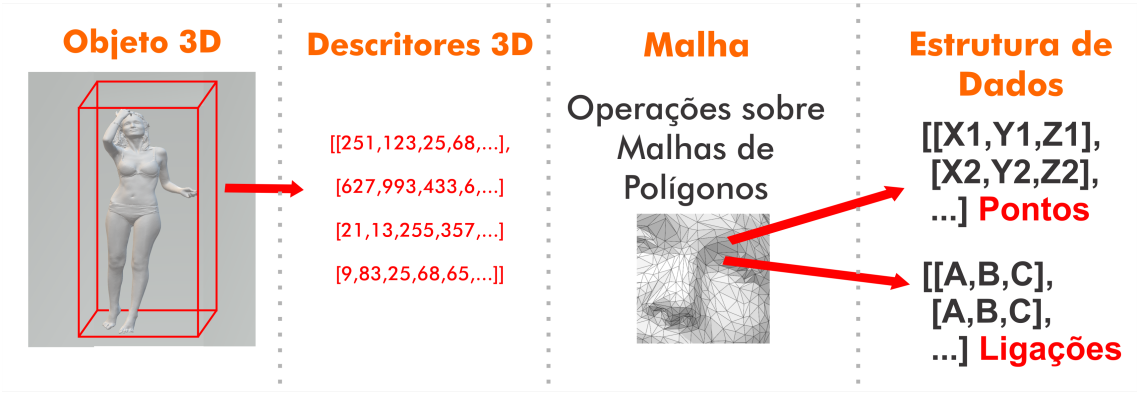
Porém, uma máquina necessita extrair padrões e utilizá-los como informação desta imagem. Para isto, são realizadas operações sobre pixels, as quais apresentam características das relações entre eles. Conforme é possível ver na Figura 4, as características da imagem são extraídas e armazenadas como vetores de descritores para então serem utilizadas como informação sobre aquela imagem.

Figura 4 – Representação Computacional de Imagens 2D.

Fonte: Elaborado pelo autor.

Os objetos 3D são armazenados de forma distinta e são representados pela utilização de pontos no espaço 3D chamados de vértices, e por ligações entre estes pontos, chamadas de arestas. Com isso, o objeto pode ser representado virtualmente no computador. Porém, não há mais a extração de características de relações de pixels, mas sim das relações entre os polígonos deste objeto. Os triângulos são a forma geométrica mais utilizada, pois constituem o menor polígono que forma um plano. Na Figura 5, é possível observar um objeto 3D representado por descritores, sendo que a representação é sobre uma malha de polígonos que são estruturados como pontos e ligações.

Figura 5 – Representação Computacional de Objetos 3D.



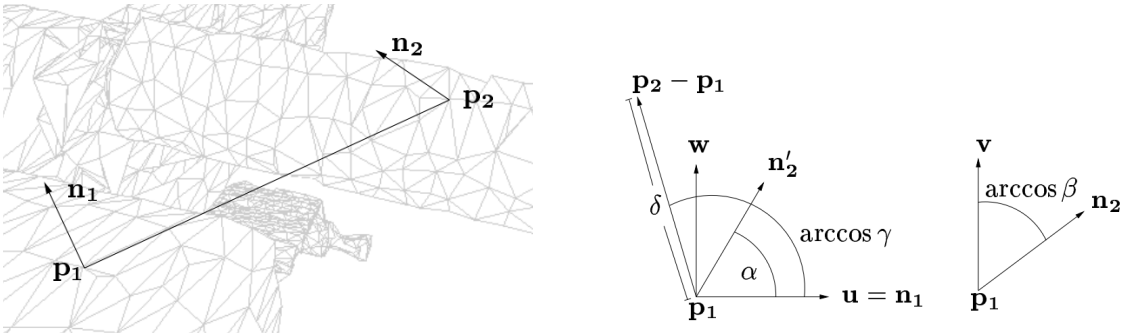
Fonte: Elaborado pelo autor.

2.3.2 Histograma de Relação Par-*Surflets*

Objetos 3D são representados por polígonos, os quais são planos com características próprias. Tais características podem ser extraídas a partir de técnicas de geometria analítica e álgebra linear. Para isto, estabelece-se uma relação entre os polígonos de forma a efetuar cálculos que descrevam tal relação de um polígono com o outro.

A relação entre pares de *Surflets* - Pontos 3D com coordenadas espaciais + Vetor Normal ao plano (*Surflets*) - é uma técnica proposta por (WAHL; HILLENBRAND; HIRZINGER, 2003), onde o resultado da relação entre os polígonos é um conjunto de características de 4 dimensões que são definidas pela interação entre todos os pares de *Surflets*. Esta técnica é invariante à translação e rotação. A relação entre os pontos p_1,p_2 e suas normais são ilustradas na Figura 6, bem como os parâmetros das características com 4 dimensões (4D) propostas pelo autor.

Figura 6 – Relação Entre Pares de *Surflets*.



(a) Pontos p_1,p_2 e sua Orientação n_1,n_2 (b) Parâmetros$(\alpha, \beta, \gamma, \delta)$

Fonte: (WAHL; HILLENBRAND; HIRZINGER, 2003)

O algoritmo de extração de características utiliza os pontos 3D dados por (X,Y,Z)

de polígonos representados por, no mínimo, três vértices (triângulos). Com isto, o vetor normal a cada plano é calculado para utilização nos cálculos de extração de características.

Os seguintes símbolos são utilizados nas equações a seguir: $(\cdot)$ denota o produto escalar entre dois vetores, $(\times)$ denota o produto vetorial entre dois vetores, $||.||$ denota a norma de um vetor, e $|.|$ denota o módulo de um número real. Para cada par de *Surflets* (p_1,n_1) e (p_2,n_2), o sistema de coordenadas foi definido como a seguir.

A origem é definida como p_1 se a Equação 2.1 for satisfeita; senão p_2 é definido como origem. Assumindo que p_1 seja a origem temos as equações a seguir. Mas caso p_2 seja definido como origem, as próximas equações devem substituir p_1 por p_2, e vice-versa.

$$|n_1 \cdot (p_2 - p_1)| \leq |n_2 \cdot (p_2 - p_1)| \tag{2.1}$$

Os vetores base do sistema de coordenadas são definidos como:

$$u = n_1, \tag{2.2}$$

$$v = \frac{(p_2 - p_1) \times u}{||(p_2 - p_1) \times u||}, \tag{2.3}$$

$$w = u \times v. \tag{2.4}$$

E a relação entre os *Surflets* (p_1,n_1) e (p_2,n_2) é descrita da seguinte maneira:

$$\alpha = \arctan(w \cdot n_2, u \cdot n_2), \tag{2.5}$$

$$\beta = v \cdot n_2, \tag{2.6}$$

$$\gamma = u \cdot \frac{p_2 - p_1}{||p_2 - p_1||}, \tag{2.7}$$

$$\delta = ||p_2 - p_1||. \tag{2.8}$$

As Equações 2.5, 2.6, 2.7 e 2.8 representam uma característica S = $(\alpha, \beta, \gamma, \delta)$, sendo que:

$$\arctan(x, y) = \begin{cases} \arctan(y/x) & \text{se} \quad x > 0 \wedge y > 0 \\ \arctan(y/x) + \pi & \text{se} \quad x < 0 \\ \arctan(y/x) + 2\pi & \text{se} \quad x > 0 \wedge y < 0 \end{cases} \tag{2.9}$$

Os atributos definidos como α e β representam a normal n_2 como um ângulo azimutal e o cosseno de um ângulo polar, respectivamente. E os outros dois, γ e δ representam a direção e o comprimento da translação de p_1 para p_2, respectivamente.

A fase de treinamento do algoritmo se dá a partir da extração dos descritores dados pelas características 4D e o mapeamento destas características em um histograma para posterior comparação.

A Equação 2.10 apresenta o mapeamento das características obtidas para um histograma com d caixas, sendo que d possui bom reconhecimento com apenas $5^4 = (625)$ caixas, segundo os autores (WAHL; HILLENBRAND; HIRZINGER, 2003).

$$h = S \mapsto i \in \{1, 2, ..., d\}; \tag{2.10}$$

A Equação 2.11 denota o processo de mapeamento das características no histograma. Para isso, os dados são normalizados entre 0 e 1, de forma que a soma de todas as caixas do histograma seja 1.

$$H(i) = \frac{card\{S \in S \mid h(S) = i\}}{cardS}; \tag{2.11}$$

Com este processo finalizado, o modelo do objeto é criado a partir de suas características. Desta forma, o modelo de outros objetos de entrada poderão ser comparados usando critérios de recuperação por similaridade.

2.4 Critérios de Similaridade Entre Descritores

Após a fase de extração de descritores e persistência do modelo em uma base de dados, poderá ser realizada uma consulta a partir de uma entrada. Comumente chamada de fase de reconhecimento, esta fase objetiva comparar o modelo do objeto da consulta com os modelos da base de dados para encontrar similaridades.

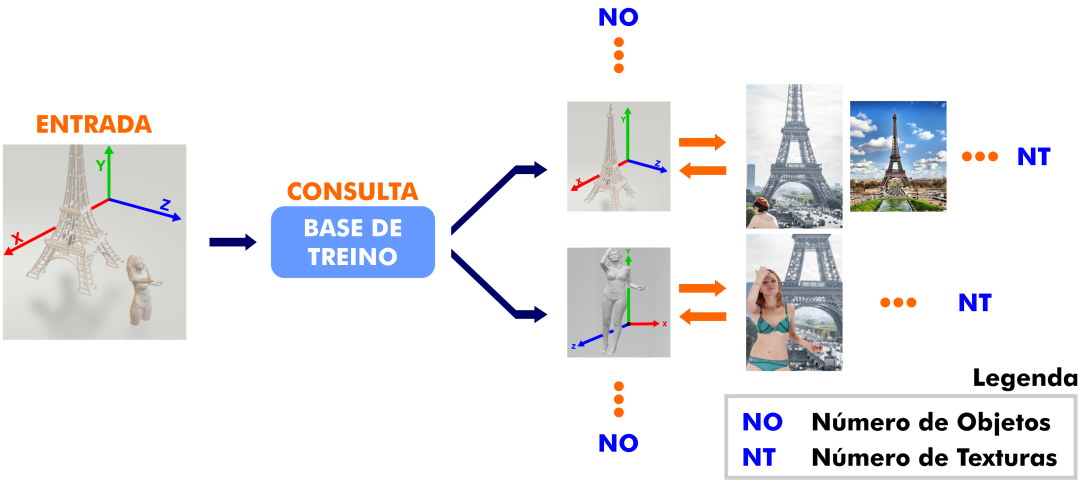
O funcionamento, de modo geral, de algoritmos de recuperação de informação é ilustrada na Figura 7. Há uma entrada, de onde são obtidas características, a verificação de similaridade desta entrada com dados da base de treino já existente, e o ranqueamento de dados por similaridade. Vale ressaltar que para o escopo deste trabalho as entradas são objetos 3D.

A fim de efetuar uma recuperação de objetos 3D por similaridade se fazem necessárias técnicas que explorem os descritores obtidos do objeto. Para isso, tais técnicas de recuperação devem ser capazes de mensurar a similaridade entre objetos a partir de um valor. A técnica de comparação entre histogramas deverá obter um número dentro de um intervalo conhecido para definir um grau de semelhança entre um objeto e outro.

Os autores (WAHL; HILLENBRAND; HIRZINGER, 2003) apresentam seis técnicas para recuperação de objetos 3D, as quais serão introduzidas a seguir. Para a compreensão das subseções a seguir é importande definir o significado das variáveis, veja:

- H_o: Histograma do objeto da base de objetos.
- $H_{o'}$: Histograma do objeto de entrada para consulta.

Figura 7 – Exemplo Ilustrativo de Consultas.



Fonte: Elaborado pelo autor.

- i: Índice da caixa do histograma do objeto.
- d: Número total de caixas do histograma.
- S: Característica extraída.
- S: Conjunto de todas as características de um objeto.
- h: Mapeamento das características nas caixas do histograma.
- H: Histograma.
- card: Cardinalidade de um conjunto.
- O: Objeto.
- $S_{o'}$: Conjunto de características do objeto O.

O método de intersecção entre o histograma dos objetos é definido pela Equação 2.12. Este método percorre todos os histogramas (H_o) da base de objetos comparando-os com o objeto de entrada ($H_{o'}$), fazendo o somatório do valor mínimo entre $H_o(i)$ e $H_{o'}(i)$. Desta forma, é obtido um valor que descreve o quanto ambos os objetos tem em comum.

$$\bigcap(H_o, H_{o'}) = \sum_{i=1}^{d} min(H_o(i), H_{o'}(i)), \tag{2.12}$$

A Equação 2.13 apresenta o método do quadrado da distância euclidiana. Este método basicamente percorre o objeto de entrada com os objetos da base de objetos calculando a diferença entre as alturas das caixas de seus histogramas. Assim como este, existem mais duas variações, chamadas de Teste χ_1^2 na Equação 2.14 e Teste χ_2^2

na Equação 2.15. Estes métodos diferem do quadrado da distância euclidiana por fatores de divisão, sendo o primeiro divisível pela caixa do histograma do objeto da base de objetos, e o segundo divisível pela caixa do histograma da base de objetos somado à caixa do histograma do objeto de entrada.

$$\varepsilon(H_o, H_{o'}) = \sum_{i=1}^{d}(H_o(i) - H_{o'}(i))^2, \tag{2.13}$$

$$\chi_1^2(H_o, H_{o'}) = \sum_{i=1}^{d}\frac{(H_o(i) - H_{o'}(i))^2}{H_o(i)}, \tag{2.14}$$

$$\chi_2^2(H_o, H_{o'}) = \sum_{i=1}^{d}\frac{(H_o(i) - H_{o'}(i))^2}{H_o(i) + H_{o'}(i)}, \tag{2.15}$$

O método *Kullback-Leibler*, apresentado na Equação 2.16, utiliza a comparação dos histogramas dos objetos baseado na diferença entre o objeto de entrada e os objetos da base de objetos, multiplicado pelo logaritmo de sua divisão.

$$\kappa(H_o, H_{o'}) = \sum_{i=1}^{d}(H_{o'}(i) - H_o(i)) \ln\frac{H_{o'}(i)}{H_o(i)}, \tag{2.16}$$

Por fim, o método *Likelihood* apresentado na Equação 2.17 calcula uma posição dentro do histograma a partir das características S=[α, β, γ, δ] atribuindo o valor da caixa do histograma com seu índice igual a esta posição.

$$L(O, S_{o'}) = \sum_{S \in S_{o'}} \ln(H_o(h(S)), \tag{2.17}$$

3 TRABALHOS RELACIONADOS

Este capítulo apresenta a abordagem de procura por trabalhos relacionados. Para isto, a seção 3.1 apresenta o caminho trilhado para levantar trabalhos relacionados e os critérios de filtragem, sendo por fim apresentados na seção 3.2 os trabalhos selecionados.

3.1 Método de Pesquisa

Foi realizada uma pesquisa ao estado da arte objetivando encontrar trabalhos que apresentassem propostas de extração de descritores 3D de objetos, assim como os validassem a partir de critérios de recuperação. Para alcançar este fim, foi utilizado o mecanismo de busca de trabalhos científicos Google Acadêmico, no qual foram inseridas palavras chave referentes ao escopo do trabalho.

Palavras chave como ("*3D Object Retrieval", "3D Shape Retrieval", "3D Object Classification", "3D Object Detection", "3D Object Recognition", "RGB-D Sensors", "Kinect Sensors"*) foram utilizadas para encontrar trabalhos relacionados ao tema. A partir desta busca foram baixados 118 trabalhos, onde uma leitura superficial foi realizada para identificar a sua contribuição ao tema deste trabalho.

Dois critérios de filtragem foram estabelecidos, o primeiro baseado em uma escala de prioridade quanto à sua relevância, como a seguir:

- **Prioridade baixa:** O trabalho, embora apresente aplicações de visão computacional, não se enquadra ao tema deste trabalho.
- **Prioridade média:** O trabalho apresenta certa contribuição como referencial teórico e aplicações práticas de visão computacional, mas se enquadra parcialmente ao tema desse trabalho.
- **Prioridade alta:** O trabalho apresenta alta contribuição de conhecimento a respeito do tema deste trabalho, como sensores 3D e algoritmos relevantes para recuperação de objetos 3D, merecendo maior prioridade em relação aos demais trabalhos.

E o segundo foi aplicado sobre os artigos de alta prioridade visando ordená-los em questão de contribuição e importância para o desenvolvimento deste trabalho.

3.2 Resultados da Pesquisa

Após a primeira filtragem, 48 trabalhos permaneceram com prioridade alta, sendo a maioria deles utilizados para fundamentar teoricamente este trabalho. Apenas 12 trabalhos foram selecionados para serem avaliados mais profundamente em virtude de suas propostas para extração de descritores e recuperação de objetos 3D.

Este trabalho é inspirado em (WAHL; HILLENBRAND; HIRZINGER, 2003), no qual utilizam uma relação entre pares de *Surflets* para gerar conjuntos de características

4D. A partir disso, as características de cada objeto da base de dados são mapeados em seu respectivo histograma, o qual representa as informações sobre sua forma 3D. São aplicados 6 critérios para validar a recuperação de objetos 3D, sendo que 2 deles (*Likelihood* e *Kullback-Leibler*) apresentam aproximadamente 99% de acerto na recuperação de objetos modelados em computador, e apresentam boas taxas de recuperação para objetos com ruído e oclusão. Para objetos reais extraídos a partir de sensores e scanners RGB-D não há teste, porém foi realizada uma simulação a partir da modificação dos objetos da base de dados em relação a oclusão, ruído, visibilidade e resolução, obtendo resultados promissores.

Os autores (CHAOUCH; VERROUST-BLONDET, 2007) apresentam um descritor para imagens de profundidade adaptadas para a correspondência e recuperação de modelos 2D/3D. Eles propõem uma representação de um modelo 3D em 20 imagens de profundidade obtidas a partir dos vértices de um dodecaedro regular. Uma imagem de profundidade de um modelo 3D é associada a um conjunto de linhas de profundidade que são transformadas em sequências. A informação da sequência de profundidade fornece uma descrição mais precisa dos limites da forma 3D do que usando outros descritores de forma 2D. A computação de similaridade é executada quando a Distância de Programação Dinâmica (DPD) é usada para comparar os descritores de linha de profundidade. O DPD leva a uma correspondência precisa de seqüências, mesmo na presença de deslocamento local na forma.

O trabalho de (LÜ; HE; XUE, 2009) apresenta um algoritmo de recuperação e classificação de modelos 3D. Um histograma de distância e invariantes de momento é proposto para melhorar o desempenho de recuperação. A principal vantagem de usar um histograma de distância é sua invariância às transformações de escala, translação e rotação. Com base na premissa de que dois objetos semelhantes devem ter alta informação mútua, a consulta de dados 3D deve transmitir uma grande quantidade de informações sobre a forma dos dois objetos e, assim, é proposta uma medição mútua de informações para realizar a comparação de similaridade de objetos 3D.

Os autores (SCHERER; WALTER; SCHRECK, 2010) adaptam o descritor de imagem 2D *Histograms of Oriented Gradients* (HOG) para o domínio 3D. Apresentam um conceito para transferir o algoritmo de extração do descritor HOG de 2D para 3D. Além disso, fornecem uma estrutura de implementação para extrair recursos do 3D HOG de modelos de malha 3D e apresentam uma avaliação experimental sistemática da eficácia de recuperação desse novo descritor 3D.

Os autores (LI; JOHAN, 2010) apresentam um extrator de descritores híbrido que combina os recursos de distância radial global e local, utilizando abordagens baseadas em histograma e visualização. Seus resultados apontam que o descritor de formas híbridas apresentado supera várias abordagens típicas baseadas em histograma e baseadas em visualizações.

O trabalho de autoria de (BO; REN; FOX, 2011) desenvolve um conjunto de recursos de kernel em imagens de profundidade que modelam tamanho, forma 3D e bordas de profundidade em uma única estrutura. Através de experimentos, sua proposta de características locais captura diferentes aspectos de sugestões de uma visão de profundidade que se complementam, superando significativamente os recursos 3D tradicionais.

Os autores (MAHMOUDI; BENJELLOUN; ANSARY, 2011) propõem indexação 3D e uma abordagem de pesquisa baseada na similaridade entre as visualizações. A comparação de similaridade de imagens 2D correspondem ao contorno visual da imagem, obtidas através de segmentação de formas com o descritor baseado em contornos, a fim de resolver problemas de escala e invariância. Os autores propõem combinar a descrição de pesquisa parcial com uma segunda descrição baseada na região global por momentos de Zernike.

É apresentada uma melhoria dos métodos de recuperação de formas 3D com base na abordagem *bag-of-feature*, no trabalho de (WARDANI; MOSTAFA; TADONKI, 2012). Esses métodos usam essa abordagem para integrar um conjunto de recursos extraídos de visões 2D de objetos 3D usando o algoritmo SIFT em histogramas usando quantização vetorial que é baseada em um livro de códigos visual global. A fim de melhorar os desempenhos de recuperação, eles propõem associar a cada objeto 3D seu livro de códigos visual local em vez de um livro de códigos global único.

No trabalho de (LOGOGLU; KALKAN; TEMIZEL, 2016) são apresentados dois descritores locais 3D para reconhecimento de objetos, a primeira técnica baseada em histogramas espaciais de *Surflets* (*Histograms of Spatial Concentric Surflet-Pairs* (SPAIR)) e a segunda técnica baseada em cores (*Colored SPAIR* (CoSPAIR)). Os descritores propostos são comparados com o estado da arte através de bases de dados disponíveis de forma pública, e foi constatado que o desempenho em taxas de reconhecimento foi melhorado.

Os autores (HONG; KIM, 2017) apresentam um método de identificação de modelo 3D baseado nas *Convolutional Neural Networks* (CNN) de imagem em profundidade. Para identificar um modelo 3D, primeiro obtém um número adequado de versões modificadas que poderiam ser feitas por infratores de direitos autorais. Então, eles são representados por um número de imagens de profundidade de visão 2D que são capturadas de vértices distribuídos uniformemente em um poliedro convexo regular. Finalmente, uma CNN é treinada por essas imagens de profundidade para adquirir a capacidade de identificar o modelo 3D.

O modelo *Latent Dirichlet Allocation* (LDA) para a extração de tópicos visuais é aplicado no trabalho de (NIE et al., 2017), onde utilizam o recurso visual de distribuição de tópicos da imagem para lidar com o problema de recuperação de objetos 3D. Eles adicionam informações espaciais do recurso visual para geração de documentos. Primeiramente extraem informações com o algoritmo SIFT de cada imagem 2D extraída do objeto 3D. Para posteriormente estruturar os documentos visuais de acordo com as informações

espaciais do modelo 3D. Finalmente, o modelo LDA é usado para extrair o modelo de tópico para a recuperação. Os autores também propõem um modelo multitarefa para melhorar o desempenho de recuperação.

Os autores (GAO et al., 2018) exploram as limitações ainda existentes na área de recuperação de objetos 3D desenvolvendo uma nova solução chamada *Group-Pair Convolutional Neural Networks* (GPCNN). Esta rede pode aprender em conjunto os recursos visuais de várias vistas de um modelo 3D e otimizar a recuperação de objetos. Empregam um esquema de aprendizado por pares, que aprende os parâmetros do modelo a partir da similaridade de cada par de amostras, em vez da maneira tradicional de aprender com a correspondência esparsa de amostra.

Na Tabela 1 os trabalhos relacionados são apresentados por título, autores e técnicas utilizadas, respectivamente. A partir deste levantamento bibliográfico é possível identificar diferentes técnicas para melhorar a recuperação de objetos 3D.

Tabela 1 – Trabalhos Relacionados.

Título do trabalho	**Autores**	**Técnica utilizada**
Surflet-Pair-Relation Histograms: A Statistical 3D-Shape Representation for Rapid Classification	(WAHL; HILLENBRAND; HIRZINGER, 2003)	Surflet-Pair-Relation Histograms
A New Descriptor for 2D Depth Image Indexing and 3D Model Retrieval	(CHAOUCH; VERROUST-BLONDET, 2007)	Depth Line
Content-Based Similarity for 3D Model Retrieval and Classification	(LÜ; HE; XUE, 2009)	Distance Histogram
Histograms of Oriented Gradients for 3D Object Retrieval	(SCHERER; WALTER; SCHRECK, 2010)	Histogram of Oriented Gradients
3D Model Retrieval Using Global Radial Distances	(LI; JOHAN, 2010)	Global + Local Radial Distances
Depth Kernel Descriptors for Object Recognition	(BO; REN; FOX, 2011)	Kernel Descriptor
3D Objects Retrieval Using Curvature Scale Space and Zernike Moments	(MAHMOUDI; BENJELLOUN; ANSARY, 2011)	Curvature Scale Space + Zernike Moments
Improving 3D Shape Retrieval Methods Based On Bag-Of-Feature Approach by Using Local Codebooks	(WARDANI; MOSTAFA; TADONKI, 2012)	BF-SIFT + CM-BOF
CoSPAIR: Colored Histograms of Spatial Concentric Surflet-Pairs for 3D object recognition	(LOGOGLU; KALKAN; TEMIZEL, 2016)	Histograms of Spatial Concentric Surflet-Pairs + Colored SPAIR.
A 2D-View Depth Image and CNN-Based 3D Model Identification Method	(HONG; KIM, 2017)	Convolutional Neural Network
3D Object Retrieval Based on Spatial+LDA Model	(NIE et al., 2017)	Spatial + Latent Dirichlet Allocation Model
Group-Pair Convolutional Neural Networks for Mult-View Based 3D Object Retrieval	(GAO et al., 2018)	Group Pair Convolutional Neural Network

Fonte: Elaborado pelo autor.

4 DESENVOLVIMENTO

Neste capítulo são definidas as etapas a serem adotadas para atingir o objetivo deste trabalho. Na seção 4.1, é definida a tecnologia utilizada. Na seção 4.2, são apresentadas as bases de objetos utilizadas para validação do algoritmo proposto. A seguir, na seção 4.3, é apresentado o protótipo até então desenvolvido. Por fim, na seção 4.7, é apresentada a visão geral das tarefas a serem cumpridas para a conclusão deste trabalho.

4.1 Ferramentas e Tecnologia

Para validar os algoritmos, primeiramente é necessário obter uma base de objetos 3D de entrada. Para isto, foram buscadas bases de objetos utilizadas pela comunidade de visão computacional para pesquisa científica. Com isto, foi encontrado um benchmark de Princeton (SHILANE et al., 2004) que oferece cerca de 1800 objetos e arquivos de configuração para treinamento e reconhecimento.

Os objetos 3D, modelados e escaneados, são formados por malhas de polígonos, existindo diversos formatos atualmente para representá-los. Pela facilidade de compreensão, inicialmente foi utilizado o formato OBJ, uma vez que a versão simplificada deste formato armazena em texto as coordenadas (X,Y,Z) de todos os pontos do objeto, e os índices dos pontos que serão conectados para formar as arestas. Também foram utilizados objetos no formato OFF, os quais são o formato padrão utilizado no Princeton Shape Benchmark. Este formato assemelha-se ao OBJ, pois nele também são apresentados em texto os vértices e arestas.

Embora o Princeton Shape Benchmark ofereça grande quantidade de objetos 3D, sua base de dados é de 2004 e não oferece objetos simples, como por exemplo poliedros regulares. Tais objetos simples e regulares são necessários para a primeira validação dos algoritmos uma vez que todos objetos reais possuem certa simetria. Com isto, foi utilizado o 3D Builder para a geração de 90 objetos pertencentes a 9 classes: dodecaedro, hexaedro, icosaedro, octaedro, tetraedro, cilindro, cone, esfera e toróide. Mais especificações destes objetos são apresentados na seção 4.2.

Para a implementação dos algoritmos, a linguagem de programação Python foi utilizada. Esta escolha deve-se à capacidade de abstrair técnicas de programação e focar diretamente no problema, o que facilitou na implementação inicial do protótipo.

4.2 Bases de Objetos 3D

Para validar os algoritmos, este trabalho reúne quatro bases de dados com diferentes perfis. A seguir estas bases serão apresentadas:

4.2.1 Bases de Objetos 1 e 2

As bases de objetos 1 e 2 reunem 304 objetos e 70 objetos, respectivamente. Os objetos selecionados correspondem a uma amostra do total de aproximadamente 1800 objetos do Princeton Shape Benchmark.

Estas duas bases foram organizadas da seguinte forma. Inicialmente foi criada a base de objetos 1 com os 304 objetos que atendem as seguintes restrições: 1) ocupam menos de 50 KB de memória; 2) pertencem a uma classe que possui pelo menos cinco objetos.

A primeira restrição deve-se ao problema de gerenciamento de memória do Python, uma vez que a memória não é desalocada durante a execução do algoritmo. Isto acarreta em um alto consumo de memória, inviabilizando a execução em computadores "tradicionais"com até 4GB de memória RAM. Foram retornados 881 objetos com tamanho inferior a 50KB.

A partir disso, a segunda restrição está relacionada aos arquivos de descrição contidos no benchmark, os quais classificam e agrupam os objetos em classes destacando o seu tipo (i.e, pessoa, pássaro, avião) e os demais objetos similares. Portanto, baseado nesta pré-classificação oferecida pelo benchmark, foram agrupados aqueles objetos que atendem às restrições 1 e 2 mencionadas.

Com isso, a base de objetos 1 possui 20 classes e um total de 304 objetos 3D, enquanto a base de objetos 2 é uma versão reduzida desta mesma base, possuindo um total de 14 classes com 5 objeto cada, totalizando 70 objetos 3D. O menor número de objetos da base de objetos 2, deve-se à validação da escalabilidade da acurácia dos algoritmos, visando verificar o comportamento dos resultados de acordo com a variação de objetos e classes da base de objetos.

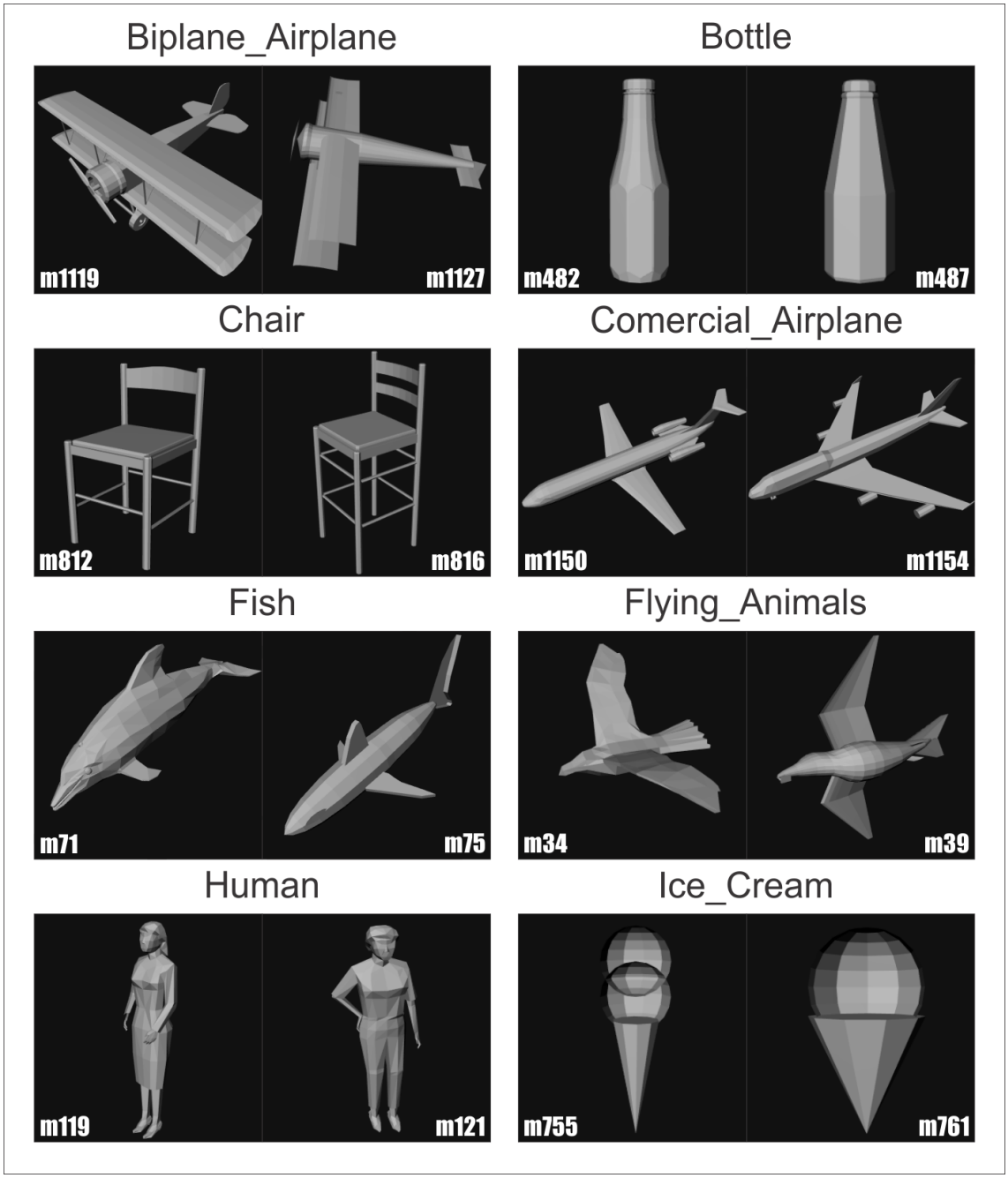
Nas Figuras 8 e 9 são apresentados alguns dos objetos pertencentes às bases de objetos 1 e 2, nelas são mostrados dois objetos de 14 classes das bases de objetos 1 e 2. Vale ressaltar que todos os objetos não são apresentados visualmente neste trabalho por questão de espaço, uma vez que são utilizados mais de 500 objetos, mas a lista completa dos objetos utilizados está presente nos Apêndices A, B, C e D.

4.2.2 Base de Objetos 3

Esta base de objetos foi criada para suprir a necessidade de validar os algoritmos com objetos mais simples, e também devido à similaridade destes objetos com formas simétricas em objetos reais (e.g, bola/esfera, cilindro/copo, etc).

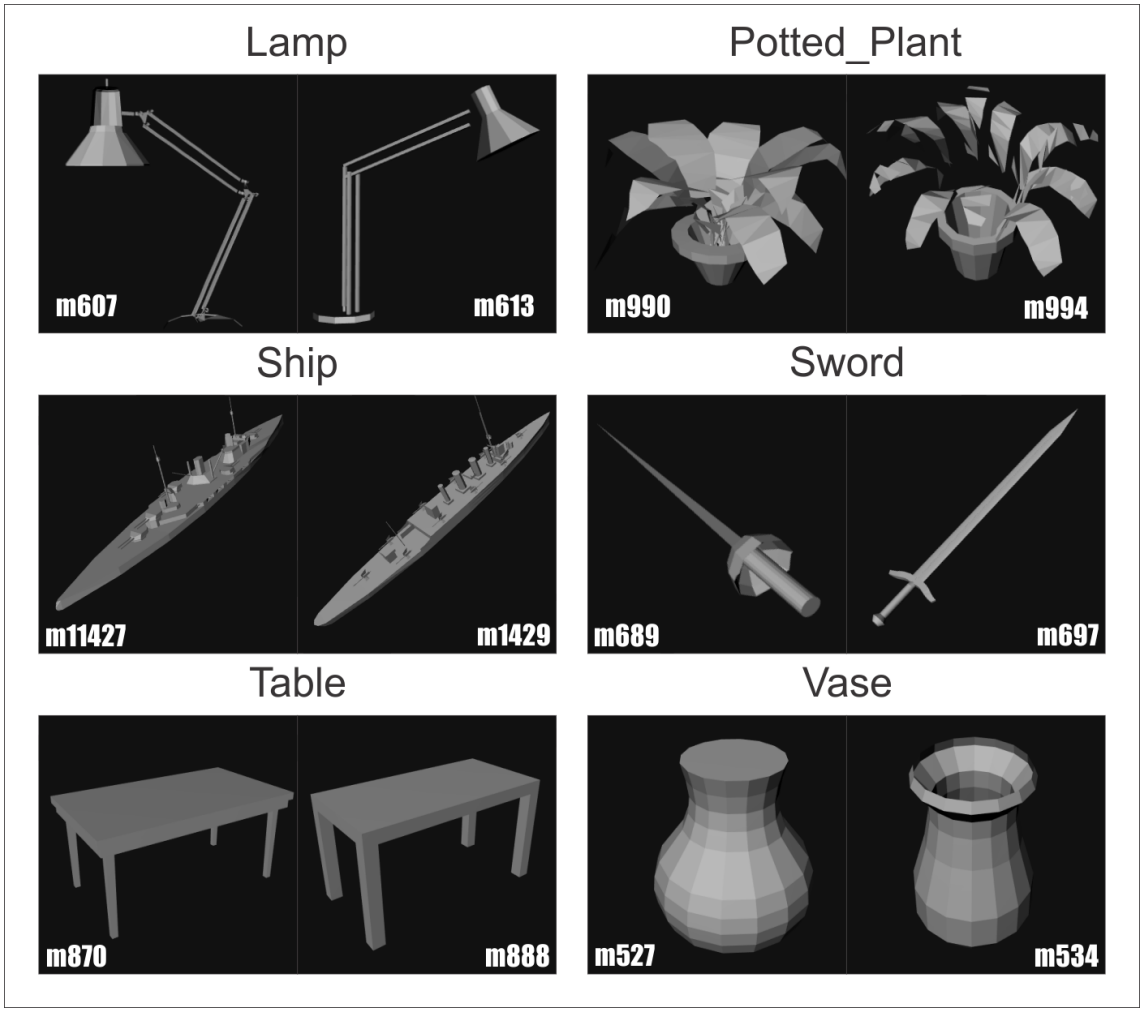
Além disso, esta base de objetos foi criada pensando-se em testar características fundamentais de algoritmos de recuperação de objetos 3D, como invariância à escala, rotação e translação. Para isto, foram criados 90 objetos pertencentes às 9 classes já mencionadas (dodecaedro, hexaedro, icosaedro, octaedro, tetraedro, cilindro, cone, esfera

Figura 8 – Bases de Objetos 1 e 2 - Amostra de 28 Objetos



Fonte: Elaborado pelo autor.

Figura 9 – Bases de Objetos 1 e 2 - Amostra de 28 Objetos



Fonte: Elaborado pelo autor.

e toróide). Para cada classe foram criados 10 objetos, nominados de 0 à 9 seguindo o aumento da escala do número de polígonos, bem como com posição e rotação distinta, como é possível observar na Figura 10.

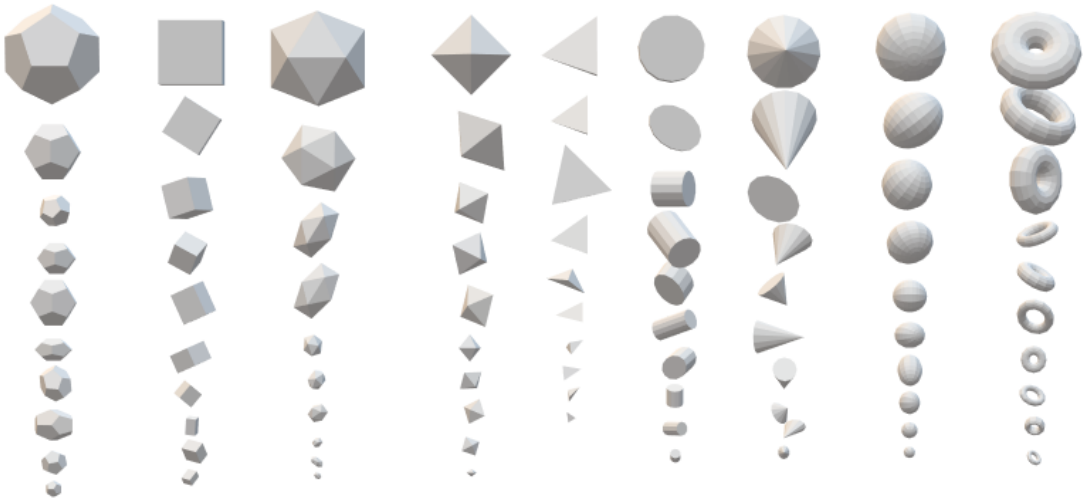
4.2.3 Base de Objetos 4

Foi identificada a necessidade da criação de uma base de objetos com irregularidades, pois, na prática, objetos 3D escaneados por sensores são capturados com ruído.

Para isto, foram selecionadas 15 classes entre as apresentadas nas bases 1, 2 e 3, e foi selecionado 1 objeto de cada uma. Então foi desenvolvido um *script* que carrega estes objetos e gera 5 novos objetos com ruído distribuído aleatoriamente no objeto.

Gerar ruído significa manipular o objeto original de forma a distorcer o posicionamento inicial dos seus vértices. A variação de ruído foi definida em 2%, variando 1% para mais ou para menos a posição espacial dos vértices do objeto. Este valor foi escolhido,

Figura 10 – Base de Objetos 3 - 90 Objetos



Fonte: Elaborado pelo autor.

pois ainda apresenta características similares ao objeto original, uma vez que foram realizados variações progressivas 0% à 20% de ruído e em todas a partir de 2% o objeto ficou irreconhecível.

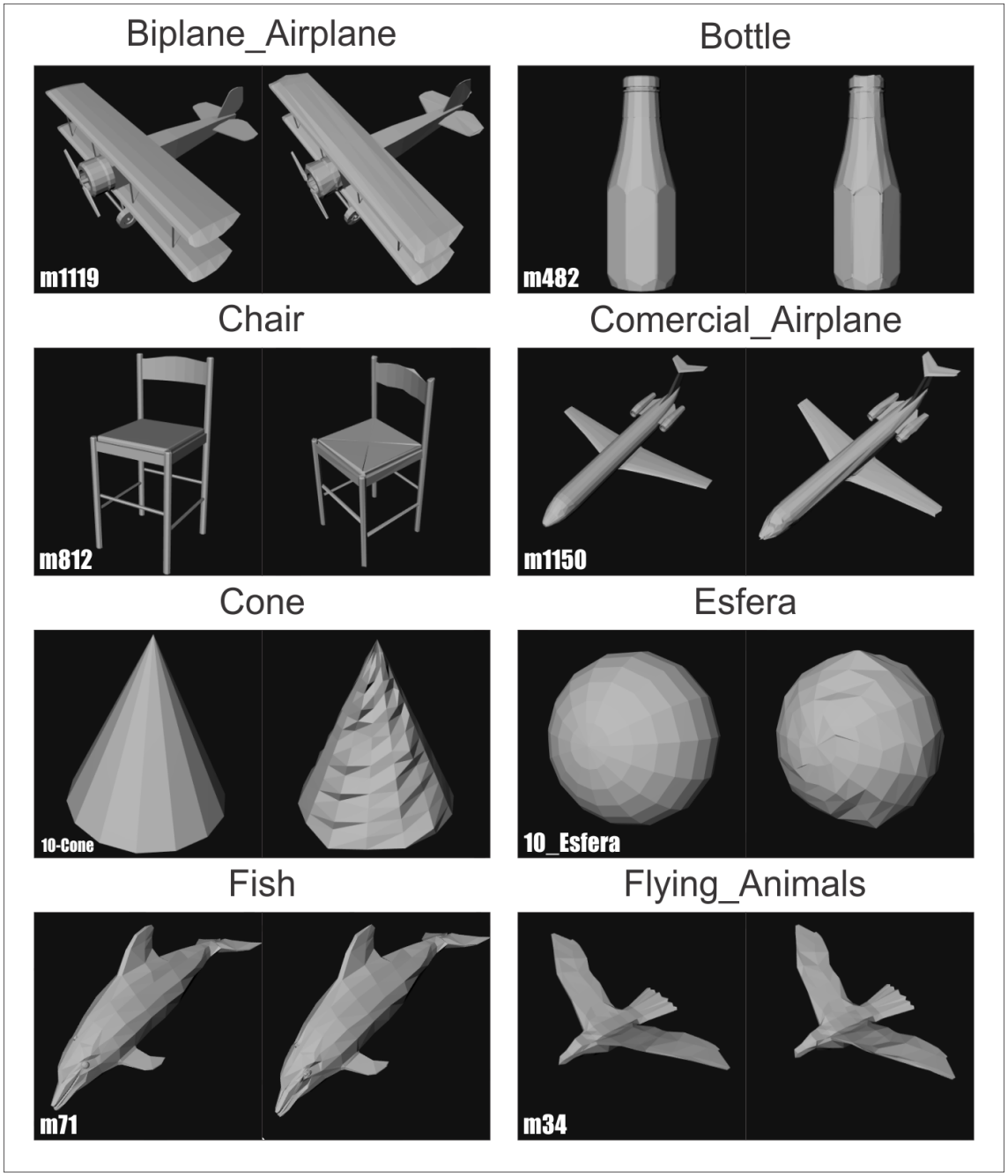
Nas Figuras 11 e 12 é apresentado um exemplo de cada uma das 15 classes com ruído, onde cada classe é descrita acima dos objetos. Em cada classe, o objeto à esquerda é o objeto original, e o objeto à direita é um dos que foram criados com ruído.

4.3 Funcionamento do Algoritmo

Este trabalho foi baseado em (WAHL; HILLENBRAND; HIRZINGER, 2003). O algoritmo consiste em duas fases principais. A fase 1, como ilustra a Figura 13, consiste no treinamento do algoritmo. Uma base de objetos é passada como entrada, de onde são extraídas características de cada objeto. Posteriormente o histograma de cada objeto é montado com estas informações, e por fim o modelo é persistido em arquivo de texto.

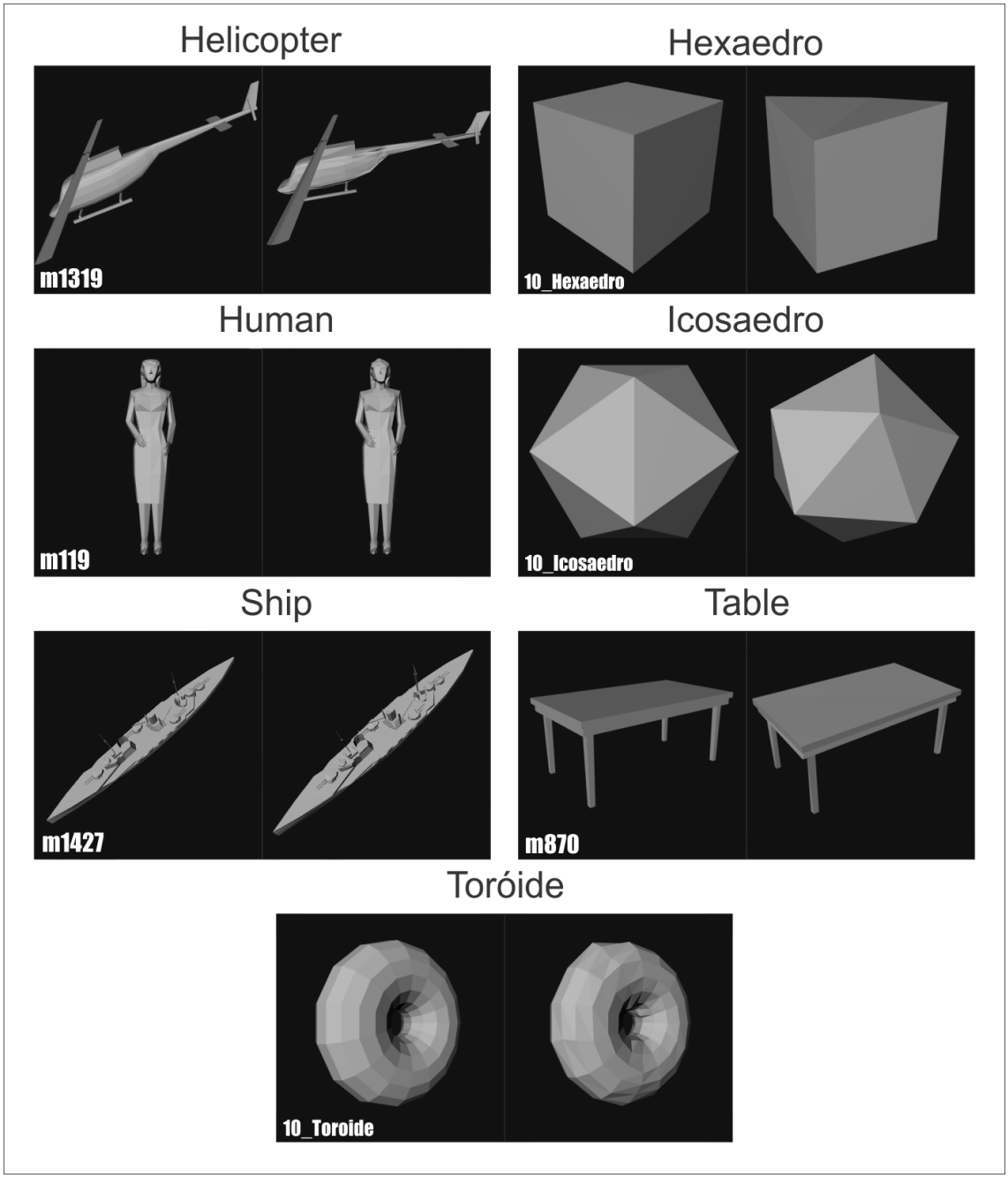
A fase 2 apresentada na Figura 14 é a parte da consulta, onde há um objeto de entrada para encontrar similares entre os modelos já treinados. A entrada passa pela extração de características e geração de seu modelo, o qual será comparado com os já existentes no banco de descritores. Por fim, os objetos são ranqueados por similaridade.

Figura 11 – Base de Objetos 4 - Amostra de 30 Objetos



Fonte: Elaborado pelo autor.

Figura 12 – Base de Objetos 4 - Amostra de 30 Objetos



Fonte: Elaborado pelo autor.

Figura 13 – Fase 1 do Algoritmo.

Fonte: Elaborado pelo autor.

Figura 14 – Fase 2 do Algoritmo.

Fonte: Elaborado pelo autor.

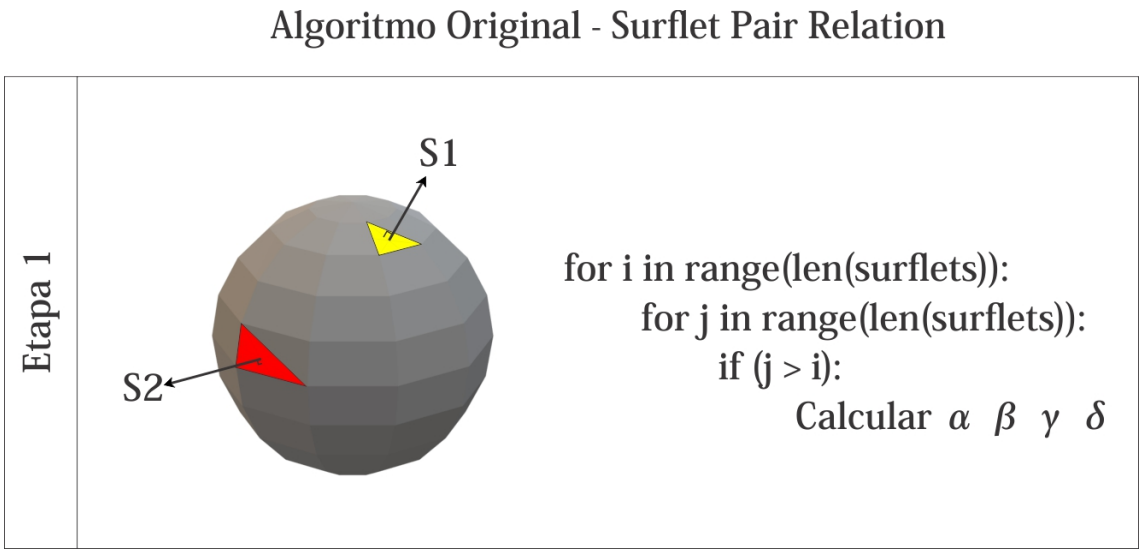
4.4 Etapas de Execução do Algoritmo

O algoritmo de (WAHL; HILLENBRAND; HIRZINGER, 2003) passa por 5 etapas de execução, que são: Extração de Características, Mapeamento no Histograma, Critérios de Similaridade Entre Histogramas, Ordenação e Recuperação dos objetos 3D. A etapa de extração de características é ilustrada na Figura 15, onde a relação de todos os *Surflets* do objeto é avaliada entre si para calcular as subcaracterísticas $[\alpha, \beta, \gamma, \delta]$, que foram descritas anteriormente na subseção 2.3.2, e que podem ser observadas na Figura 6.

Após a obtenção das subcaracterísticas $[\alpha, \beta, \gamma, \delta]$ do objeto, elas são utilizadas como índices para o mapeamento de um histograma de 4 dimensões. A Figura 16 mostra como o processo é realizado. A partir da indexação das subcaracterísticas, é gerado um histograma normalizado de 625 caixas.

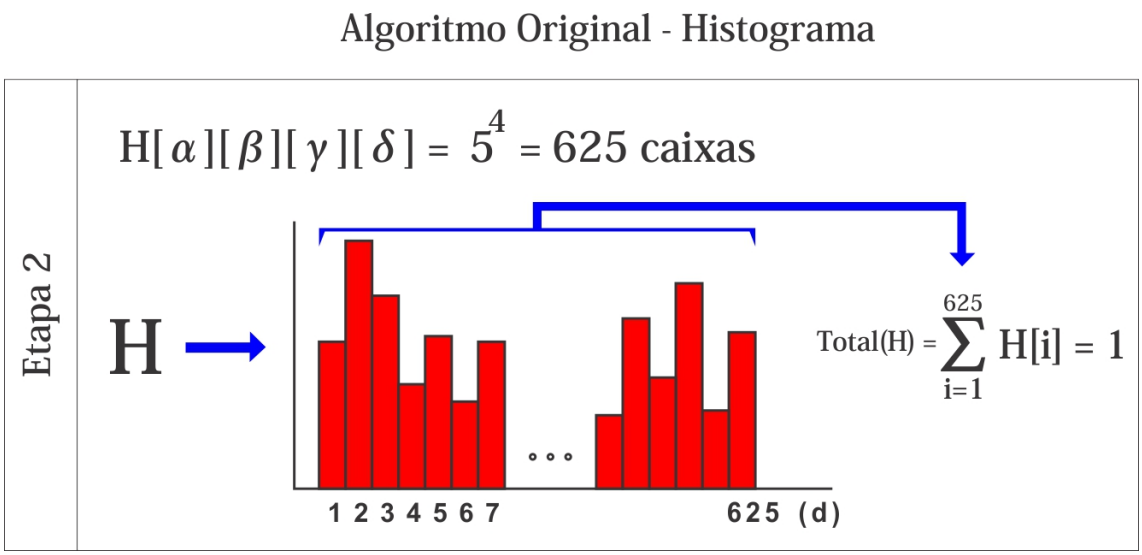
Todos os objetos treinados possuem seu histograma. Assim, quando a fase de consulta for realizada, todos estes histogramas serão analisados utilizando critérios de similaridade. Foram então implementados os 6 critérios de similaridade utilizados pelos autores (WAHL; HILLENBRAND; HIRZINGER, 2003), que são definidos a partir da Equação 2.12 à Equação 2.17. Além disso, foi proposto um novo critério de similaridade chamado de *Variation Rate*. Cada um dos critérios de similaridade implementados

Figura 15 – Etapa 1 - Extração de Subcaracterísticas.



Fonte: Elaborado pelo autor.

Figura 16 – Etapa 2 - Mapeamento no Histograma.

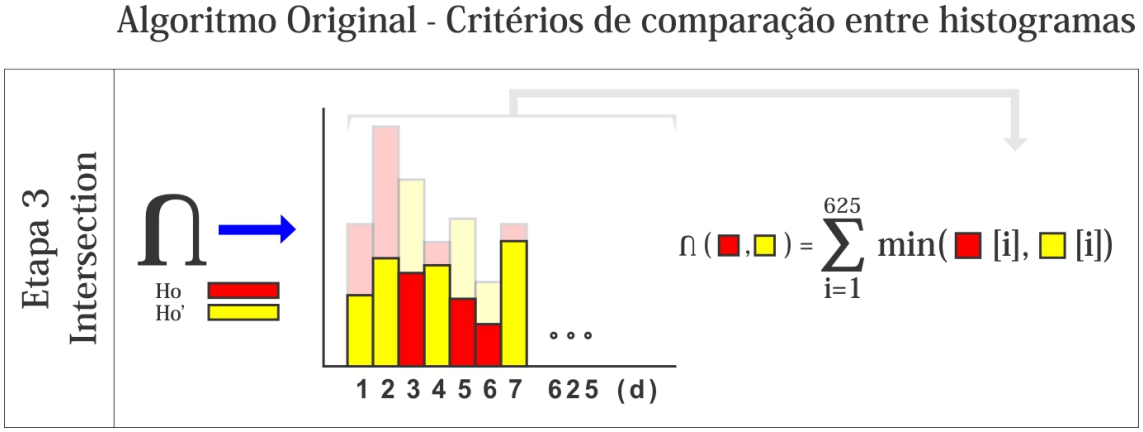


Fonte: Elaborado pelo autor.

são apresentados de forma ilustrada a seguir.

O critério *Intesection*, ilustrado na Figura 17 compara todas as caixas do histograma da base de dados (em vermelho) com o as caixas do histograma do objeto de consulta (em amarelo). Por fim é resultado o menor valor entre as duas caixas (i.e, min(Amarelo[i],Vermelho[i])), que é somada à variável de dissimilaridade. Ou seja, o resultado desta métrica é um valor que representa quanto o histograma de um objeto se aproxima do outro.

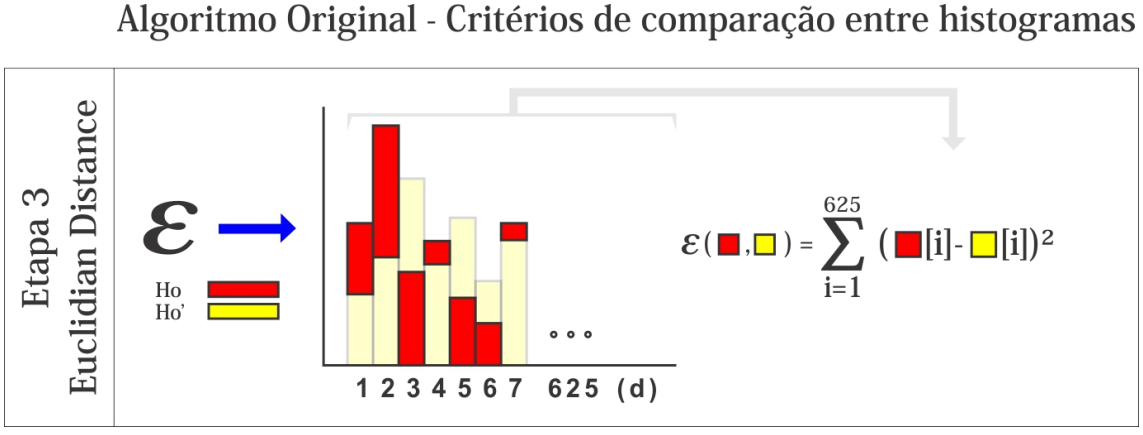
Figura 17 – Etapa 3 - Comparação de Histogramas: *Intersection*.



Fonte: Elaborado pelo autor.

Como é possível observar no critério *Euclidian Distance*, ilustrado na Figura 18, o valor atribuído ao somatório da dissimilaridade é o quadrado da diferença entre as caixas do histograma da base de dados (em vermelho) com o as caixas do histograma do objeto de consulta (em amarelo). Assim, o valor apresentado em vermelho na Figura 18 é elevado ao quadrado, sendo uma alternativa para tornar todos os valores positivos no somatório.

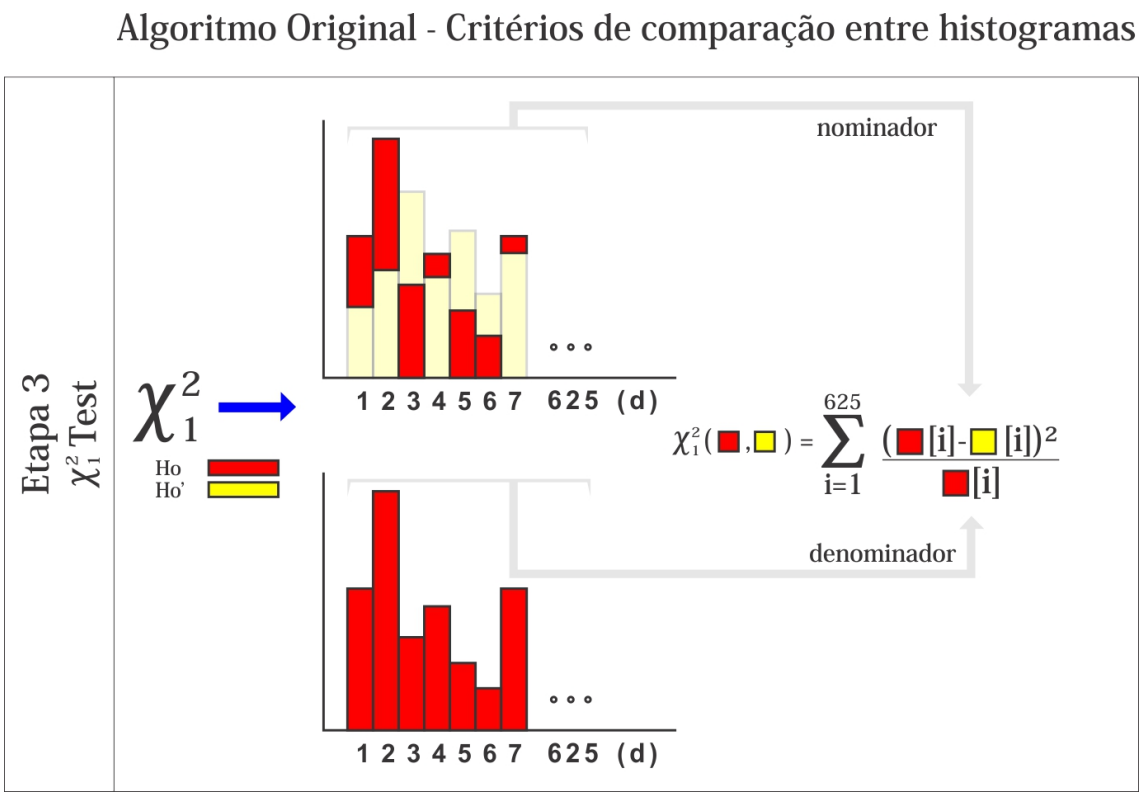
Figura 18 – Etapa 3 - Comparação de Histogramas: *Euclidian Distance*.



Fonte: Elaborado pelo autor.

Os critérios χ_1^2 *Test* e χ_2^2 *Test* são variações do critério *Euclidian Distance*, partindo do mesmo princípio. Porém, no critério χ_1^2 *Test*, ilustrado na Figura 19, a diferença entre as caixas do histograma é dividida pela caixa original proveniente do objeto da base de objetos (em vermelho).

Figura 19 – Etapa 3 - Comparação de Histogramas: χ_1^2 *Test*.



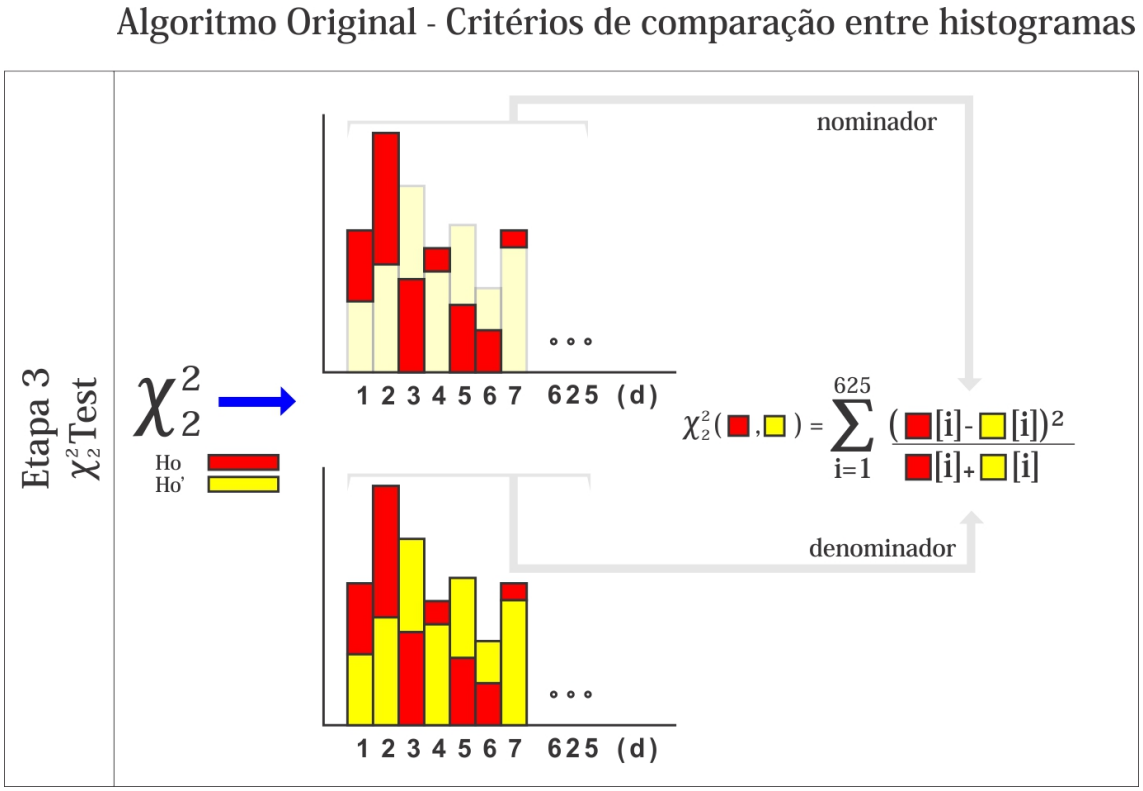
Fonte: Elaborado pelo autor.

Em seguida, o critério χ_2^2 *Test* utiliza como denominador da divisão a soma das caixas de ambos histogramas. Estas variações do método *Euclidian Distance* permitem distribuir o resultado da comparação dos histogramas em faixas de valores distintas, aonde se torna possível enxergar o resultado sob várias perspectivas.

O critério *Kullback-Leibler* utiliza a diferença entre os histogramas dos objetos multiplicada pelo logaritmo de sua divisão. Os valores dos histogramas aplicados no cálculo são ilustrados na Figura 21. Como já mencionado, o resultado destas operações sobre a diferença entre as caixas do histograma oferecem uma distribuição dos valores em uma nova faixa, servindo também como recompensa ou penalidade às caixas aproximadas ou distintas, respectivamente.

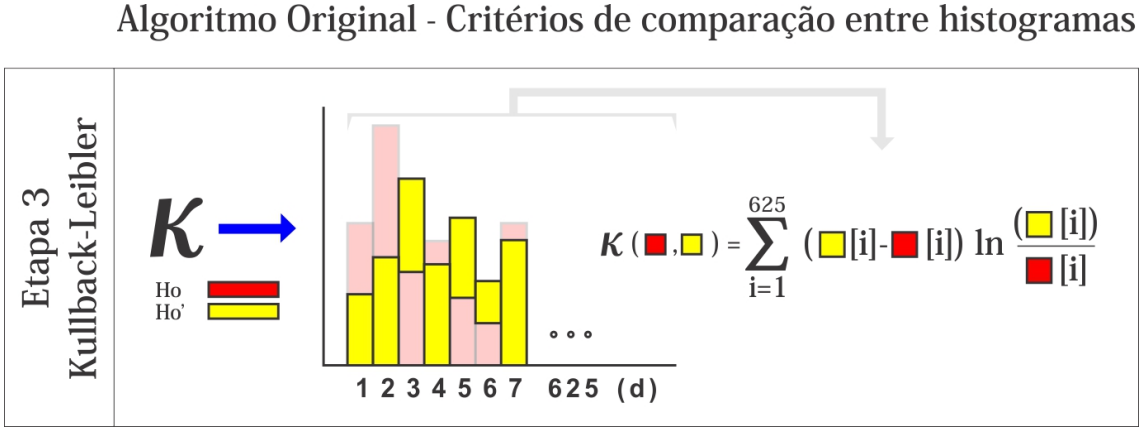
De forma distinta às apresentadas até então, o critério *Likelihood* compara os objetos em tempo de execução. Como é possível observar na Figura 22, inicialmente os *Surflets* do objeto são percorridos para obtenção das subcaracterísticas. Em seguida as subcaracterísticas 4D são convertidas para um índice contíguo que será utilizado poste-

Figura 20 – Etapa 3 - Comparação de Histogramas: χ^2_2 *Test*.



Fonte: Elaborado pelo autor.

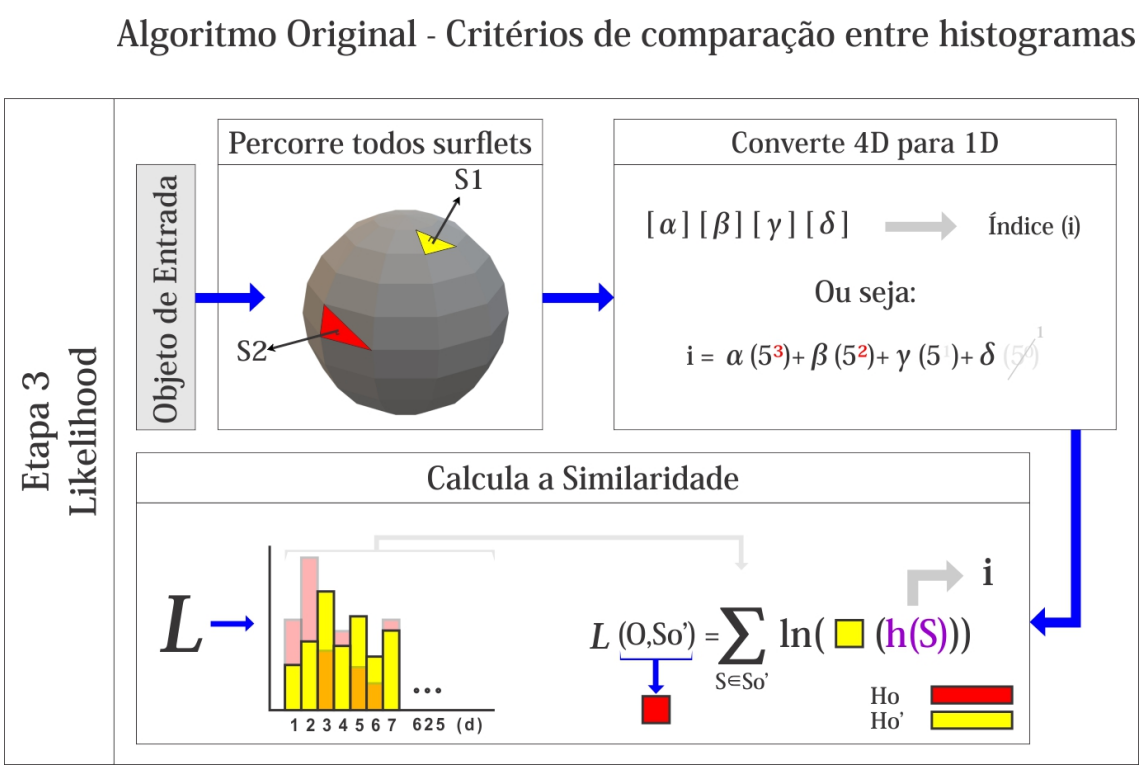
Figura 21 – Etapa 3 - Comparação de Histogramas: *Kullback-Leibler*.



Fonte: Elaborado pelo autor.

riormente no mapeamento do histograma. Com isto, os histogramas são comparados a partir do logaritmo deste mapeamento.

Figura 22 – Etapa 3 - Comparação de Histogramas: *Likelihood*.



Fonte: Elaborado pelo autor.

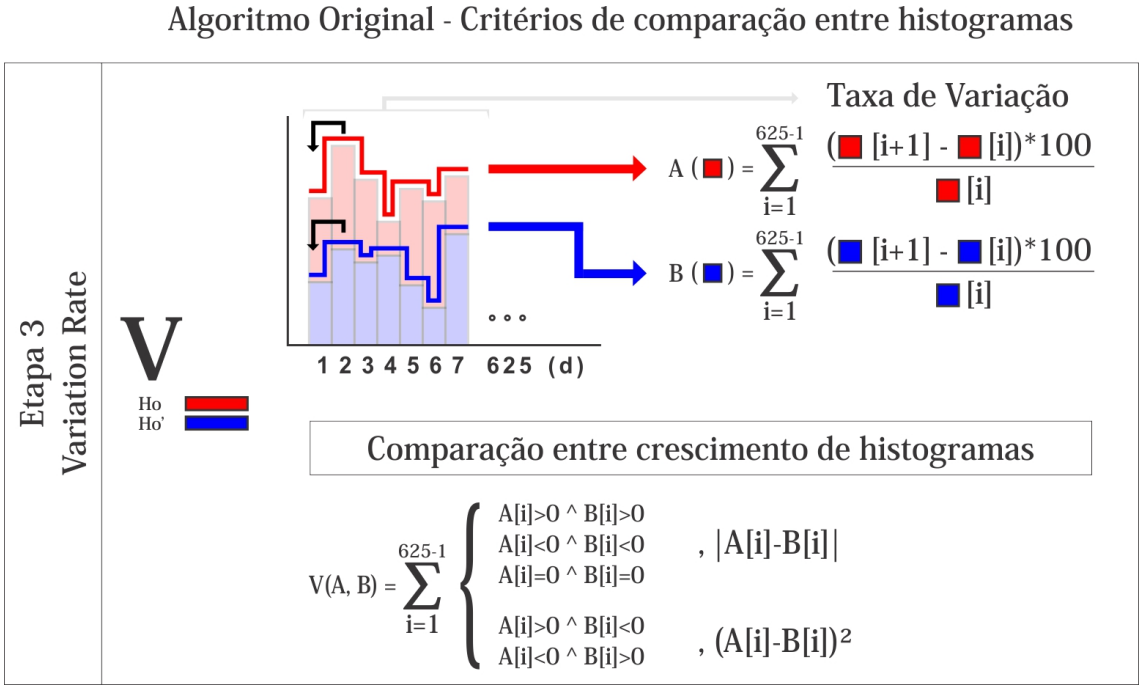
Por fim, a ilustração do critério proposto, *Variation Rate*, pode ser observada na Figura 23. Até então as técnicas utilizadas comparam o histograma de um objeto de consulta com os histogramas dos objetos já treinados, na base de objetos. Com isso, a técnica proposta busca uma alternativa diferenciada, primeiramente avaliando os histogramas internamente para posteriormente comparar suas taxas de variação entre si.

Nesta técnica, cada caixa do histograma é comparada com a caixa anterior do próprio histograma, calculando uma variação em porcentagem de uma para outra e armazenando em uma lista de variação. Em seguida, a lista de variação do objeto de consulta é comparada com a lista de variação dos objetos da base de objetos.

Com isso, o resultado atribuído à dissimilaridade entre os objetos é recompensado caso as taxas de variação entre os objetos sejam semelhantes, e penalizada caso contrário. Ou seja, caso a variação de ambos seja positiva, negativa, ou igual, há a atribuição de sua diferença. E caso a variação seja oposta, há penalização utilizando o quadrado da sua diferença. Esta técnica de recompensa e penalização foi definida, pois ao elevar ao quadrado a diferença entre as taxas de variação obter-se-á um valor maior, ou seja, uma dissimilaridade maior significa que os objetos são menos parecidos.

Na sequência, após as etapas 4 e 5, a execução do algoritmo é finalizada. Como

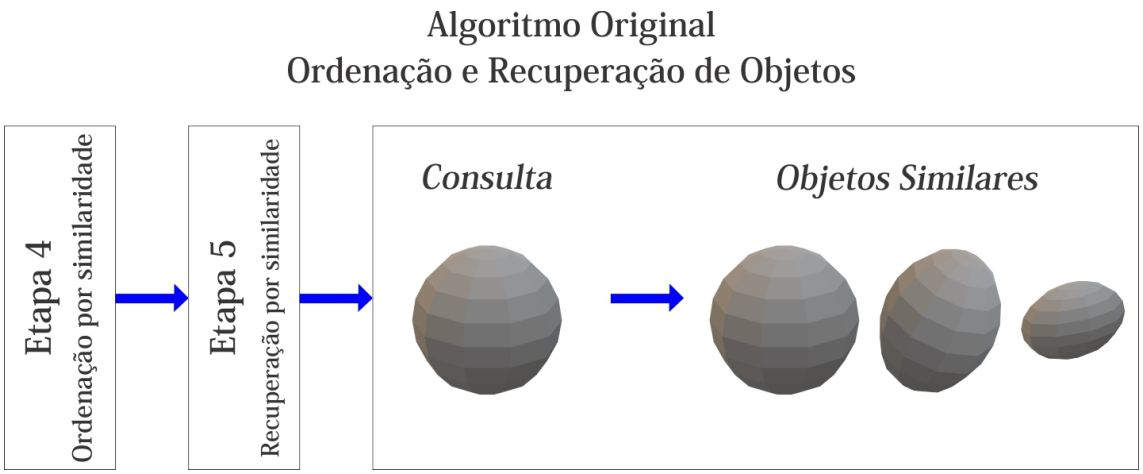
Figura 23 – Etapa 3 - Comparação de Histogramas: *Variation Rate*.



Fonte: Elaborado pelo autor.

pode ser observado na Figura 24, há então a ordenação dado os valores de similaridade obtidos, e então o ranqueamento dos objetos menos dissimilares - ou seja, mais similares.

Figura 24 – Etapas 4 e 5 - Ordenação e Recuperação de Objetos



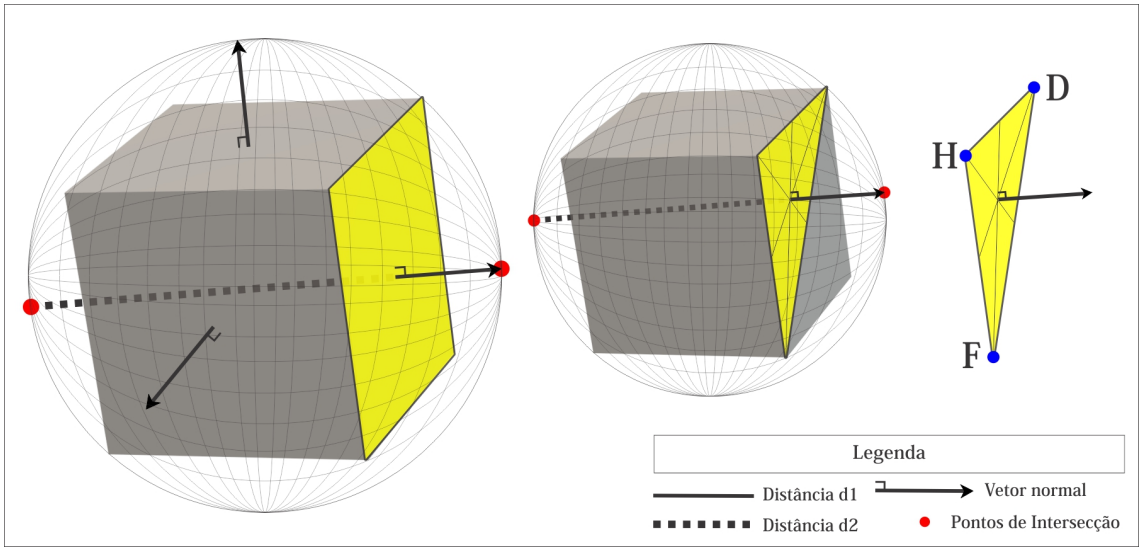
Fonte: Elaborado pelo autor.

4.5 Subcaracterística Proposta - Relação Esfera-*Surflet*

Como maneira de ampliar os estudos do trabalho original, foi desenvolvida uma nova subcaracterística ϵ, que determina a relação dos *Surflets* do objeto com uma esfera

que o envolve. O objetivo desta técnica é obter a distância de cada *Surflet* até a esfera, dado pela intersecção da reta direcionada pelo vetor normal ao triângulo. Isto pode ser observado na Figura 25, a qual apresenta um cubo 3D. O cubo à esquerda é definido por 8 vértices e 12 arestas, por isto suas faces são quadrados. Já o cubo à direita é definido por 8 vértices e 18 arestas, uma vez que suas faces são definidas por triângulos.

Figura 25 – Subcaracterística Baseada na Relação Esfera-*Surflet*.



Fonte: Elaborado pelo autor.

A interpretação da Figura 25 para quaisquer objetos é dada da seguinte forma. Para cada polígono do objeto é calculado o vetor normal, que dá a orientação para obter-se a equação de uma reta. Como todos os objetos estão envolvidos completamente pela esfera, esta reta sempre irá intersectar 2 pontos na esfera. Com isto, são obtidas as distâncias $(d1, d2)$, que são as distâncias do plano até os pontos de intersecção com a esfera. O objetivo a partir desta nova métrica é aperfeiçoar a invariância rotacional, incluindo uma medida da relação entre as distâncias das faces com um volume envolvente (neste caso, uma esfera), a qual será menor em objetos similares. Além disso, reforçar a invariância à escala dos objetos, uma vez que os valores normalizados da subcaracterística serão proporcionais à objetos com diferentes tamanhos e orientações, pois independentemente do ângulo que o objeto estiver rotacionado, as distâncias até a esfera serão as mesmas.

4.5.1 Equação da Intersecção Reta-Esfera

Os cálculos para obtenção da equação a ser utilizada no algoritmo são apresentados a seguir. Para tanto, são definidas as variáveis utilizadas e seus respectivos significados.

Variáveis da equação da Esfera:

- C : Centro da Esfera.

- r : Raio da Esfera.
- X : Pontos na Esfera.

A Equação 4.1 denota a equação da esfera.

$$||X - C||^2 = r^2 \tag{4.1}$$

Variáveis da equação da Reta:

- O : Ponto de partida da reta.
- d : Distância a partir do ponto O.
- l : Vetor que dá a direção da reta (Vetor normal ao triângulo).
- X : Pontos na Reta (Ponto médio do triângulo).

A Equação 4.2 denota a equação da reta.

$$X = O + dl \tag{4.2}$$

Para achar os pontos de intersecção da reta com a esfera deve-se igualar as Equações 4.1 e 4.2 e encontrar d, que é a distância entre o ponto médio do *Surflet* e os pontos de intersecção da reta com a esfera.

Substituindo a Equação 4.2 na Equação 4.1, tem-se:

$$||X - C||^2 = r^2 \rightarrow ||(O + dl) - C||^2 = r^2 \tag{4.3}$$

Expandindo a Equação 4.3, obtém-se:

$$\begin{aligned}(O + dl - C)(O + dl - C) &= r^2 \\ &\rightarrow d^2(ll) + 2d(l(O - C)) + (O - C)(O - C) - r^2 = 0\end{aligned}$$

Ou seja, a junção das equações 4.1 e 4.2 resume-se a uma equação de segundo grau organizada da seguinte forma:

$$(l^2)d^2 + (2(l(O - C)))d + (||O - C||^2 - r^2) = 0 \rightarrow ad^2 + bd + c = 0$$

Onde:

$$a = ||l||^2, \quad b = (2(l(O - C))), \quad c = (||O - C||^2 - r^2)$$

Então, os valores (a, b, c) são substituídos na fórmula de Bhaskara (Equação 4.4), obtendo-se a Equação 4.5.

$$d = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a} \tag{4.4}$$

$$d = \frac{-(2(l(O-C))) \pm \sqrt{(2(l(O-C)))^2 - 4(||l||^2)(||O-C||^2 - r^2)}}{2(||l||^2)} \tag{4.5}$$

Como l é um vetor unitário, $||l||^2 = 1$, assim:

$$\begin{aligned} d &= -(l(O-C)) \pm \sqrt{\Delta} \\ \Delta &= (l(O-C))^2 - ||O-C||^2 + r^2 \\ &\begin{cases} \Delta > 0, & \text{se } 2\ Intersecções \\ \Delta = 0, & \text{se } 1\ Intersecção \\ \Delta < 0, & \text{se } 0\ Intersecções \end{cases} \end{aligned} \tag{4.6}$$

Transformando a Equação 4.6, tem-se:

$$\begin{aligned} d &= -((lx,ly,lz).((Ox,Oy,Oz) - (Cx,Cy,Cz))) \pm \sqrt{\Delta} \\ \Delta &= ((lx,ly,lz).((Ox,Oy,Oz) - (Cx,Cy,Cz)))^2 - ||(Ox,Oy,Oz) - (Cx,Cy,Cz)||^2 + r^2 \end{aligned} \tag{4.7}$$

4.5.2 Exemplo: Relação Esfera-*Surflet*

Nesta subseção é apresentado um exemplo da técnica proposta. Para a demonstração é utilizado um cubo 3D de dimensões $1x1x1$ unidades, o qual possui 8 vértices e 18 arestas, sendo formado por 12 triângulos.

Para fim de demonstração, é utilizado um dos triângulos do cubo, apresentado na Figura 25, sendo este definido pelos pontos F=(1,0,1), H=(1,1,1) e D=(1,1,0). Então, é demonstrada matematicamente a obtenção dos valores que se desejam encontrar a partir da relação deste triângulo com a esfera.

Inicialmente é necessário obter os valores das variáveis que serão substituídas na Equação 4.7, que são: O (Ponto médio do *Surflet*); C (Centro do Objeto); l (Vetor normal ao triângulo); e r (Raio da esfera).

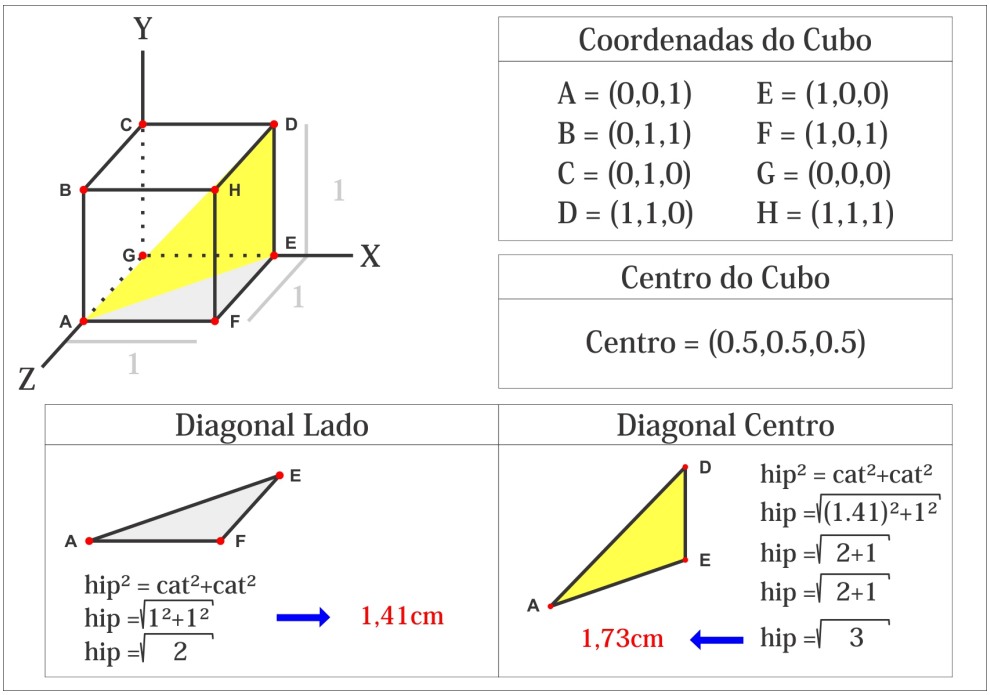
Para obter uma posição no espaço para centralizar a esfera, é necessário calcular o centro do objeto o qual será envolvido por ela. Para isto, deve-se fazer a média de todos os pontos do objeto, conforme apresentado na Equação 4.8.

$$\begin{aligned} Centro &= \frac{Todos\ os\ Pontos}{Número\ de\ Pontos} \\ &= \frac{(0,0,0)+(0,0,1)+(0,1,0)+(0,1,1)+(1,0,0)+(1,0,1)+(1,1,0)+(1,1,1)}{8} \\ &= (\frac{4}{8},\frac{4}{8},\frac{4}{8}) \rightarrow C = (0.5,0.5,0.5) \end{aligned} \tag{4.8}$$

Agora que a esfera está centralizada com o objeto, é necessário saber o diâmetro deste objeto para que ele caiba totalmente dentro da esfera. Para isto, é calculada a maior distância entre 2 pontos do objeto, que é utilizada como o diâmetro da esfera.

Computacionalmente, obter este valor requer testar a distância euclidiana entre todos os pontos e armazenar aqueles pontos que possuem a maior distância entre si. Porém, para um cubo $1x1x1$ é possível identificar visualmente que a maior distância entre 2 pontos é a diagonal interior do cubo. Como é possível ver na Figura 26, para se obter a maior distância deve-se aplicar o Teorema de Pitágoras inicialmente em um dos lados do cubo (i.e, Triângulo Retângulo AEF), visando obter o comprimento da hipotenusa. Após isto, deve-se utilizar esta medida encontrada como cateto adjacente ao triângulo ADE, encontrando, então, o valor que é atribuído ao diâmetro da esfera.

Figura 26 – Representação de Um Cubo $1x1x1$.



Fonte: Elaborado pelo autor.

Neste momento, o cubo está centralizado no interior da esfera, então já é possível traçar a distância da intersecção de cada triângulo com esta esfera. Para isto, primeiramente é necessário calcular o ponto médio e o vetor normal ao triângulo.

O cálculo do ponto médio segue o mesmo princípio do cálculo do centro do objeto, na Equação 4.8, porém, é realizada a média dos pontos do polígono. Sendo assim, o ponto médio do triângulo definido pelos pontos FHD é:

$$\begin{aligned} PontoMédio &= \frac{F+H+D}{3} = \frac{(1,0,1)+(1,1,1)+(1,1,0)}{3} \\ &= (\frac{3}{3},\frac{2}{3},\frac{2}{3}) \rightarrow O = (1,0.6,0.6) \end{aligned} \tag{4.9}$$

Para obter o vetor normal, deve-se calcular o produto vetorial entre dois vetores pertencentes ao plano (triângulo). O resultado obtido do produto vetorial é um vetor

que é ortogonal ao plano, e dará a direção da reta que irá intersectar a esfera. O cálculo realizado pode ser acompanhado na equação 4.10, onde são obtidos os vetores pertencentes ao triângulo e submetidos ao produto vetorial.

$$\begin{aligned}
\overrightarrow{FH} &= H - F = (1,1,1) - (1,0,1) = (0,1,0) \\
\overrightarrow{HD} &= D - H = (1,1,0) - (1,1,1) = (0,0,-1) \\
\overrightarrow{FH} \times \overrightarrow{HD} &= \begin{vmatrix} i & j & k \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{vmatrix} = \begin{vmatrix} 1 & 0 \\ 0 & -1 \end{vmatrix} i - \begin{vmatrix} 0 & 0 \\ 0 & -1 \end{vmatrix} j + \begin{vmatrix} 0 & 1 \\ 0 & 0 \end{vmatrix} k \rightarrow l = (-1,0,0)
\end{aligned} \tag{4.10}$$

Finalmente, os valores das variáveis foram obtidos e já podem ser aplicados à Equação 4.7, aonde serão calculadas as distâncias $d1$ e $d2$, que são dadas pela reta descrita pelo vetor normal até a intersecção com a esfera.

Substituindo na Equação 4.7 os valores obtidos, obtem-se:

$$\begin{aligned}
d &= -((-1,0,0).((1,0.6,0.6) - (0.5,0.5,0.5))) \pm \sqrt{\Delta} \\
d &= 0.5 \pm \sqrt{\Delta} \\
\Delta &= ((-1,0,0).((1,0.6,0.6) - (0.5,0.5,0.5)))^2 - ||(1,0.6,0.6) - (0.5,0.5,0.5)||^2 + (0.86)^2 \\
\Delta &= 0.25 - \sqrt{(-0.5)^2 + (-0.16)^2 + (-0.16)^2}^2 + 0.74 \\
\Delta &= 0.6944 \\
&\textit{Como } \Delta > 0, \textit{ admite 2 raízes} \\
d1 &= 0.5 + \sqrt{0.6944} = 0.5 + 0.83 = 1.33 \\
d2 &= 0.5 - \sqrt{0.6944} = 0.5 - 0.83 = -0.33 \\
&\textit{Sendo que } |d1|+|d2| \textit{ é o comprimento da reta.}
\end{aligned}$$

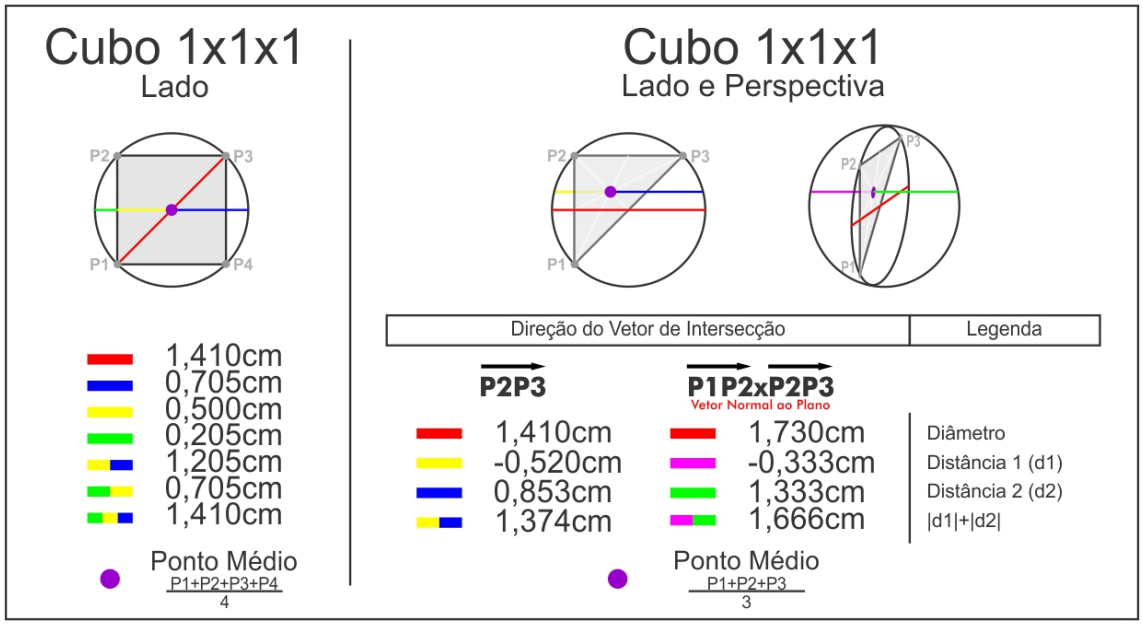
Como apresentado acima, a distância $d1$ é 1.33 e a distância $d2$ é -0.33. Contudo, utiliza-se o valor absoluto das distâncias, sendo então a soma $|d1|+|d2|= 1.66$, que é o comprimento da reta que intersecta a esfera, passando pelo ponto médio do *Surflet* e direcionada pelo vetor normal ao triângulo.

Caso fosse utilizado o ponto médio de um dos lados do cubo que formam um quadrado (Figura 25), a respectiva reta passaria pelo centro da esfera. Desta forma, a soma das distâncias $|d1|+|d2|$ possuiria o diâmetro da esfera, que neste caso é 1.73. A fim de ilustrar e facilitar a representação das distâncias de um cubo $1x1x1$ e sua relação com a esfera envolvente, algumas medidas são apresentadas em 2D na Figura 27. À esquerda da figura são apresentadas medidas de um cubo cujos lados são representados por quadrados, já à direita os lados do cubo são representados por triângulos.

4.6 Métrica para Obtenção da Taxa de Acertos

Para mensurar a qualidade dos algoritmos e dos critérios de comparação de histogramas é necessário a utilização de uma métrica. Para tanto, a porcentagem de acertos na recuperação dos objetos 3D é medida da seguinte forma: inicialmente, todos os objetos

Figura 27 – Dados a Serem Obtidos do Cubo.



Fonte: Elaborado pelo autor.

são comparados com todos, gerando um arquivo para cada objeto com dados de todos objetos da base de objetos de forma ordenada por similaridade. Caso os N primeiros objetos forem os que estão na mesma classe, há um acerto de 100% para aquele objeto. Ao final, o percentual de acerto do algoritmo é a média do percentual de todos os objetos em todas as classes.

Por exemplo, digamos que a classe 1 possua 13 objetos. Então, é contabilizado o número de vezes que os objetos pertencentes à classe 1 aparecem nas 13 primeiras posições do *ranking* de similaridade. Por exemplo, caso apenas 9 dos 13 objetos estejam nas 13 primeiras posições, será calculado o percentual de acerto daquele objeto da forma denotada pela Equação 4.11:

- Tx = Taxa de acerto do objeto
- hits = Número de acertos dentro da classe
- Nt = Número total de objetos da classe

$$Tx = \frac{hits \times 100\%}{Nt} \tag{4.11}$$

Ou seja:

$$Tx = \frac{9 \times 100}{13} = \frac{900}{13} = 69,23\%$$

Assim, diz-se que a taxa de acerto para o objeto atual é de 69,23%. Por fim, este mesmo cálculo será realizado para todos os objetos da base de objetos. Ao final da execução, será apresentada a média de acertos do algoritmo para a base de objetos.

4.7 Tarefas Realizadas

Na Tabela 2 estão listadas as atividades definidas para a validação dos algoritmos, bem como a dependência entre as atividades. As atividades foram nomeadas como ATN, onde N é uma numeração iniciando em zero, sendo que estas doze atividades contemplam desde a obtenção dos objetos 3D até o ranqueamento por similaridade.

Tabela 2 – Definição das Atividades.

ID	**Atividade**	**Dependência**
AT0	Obter objetos 3D para banco de objetos	–
AT1	Carregar objeto 3D do banco de objetos	AT0
AT2	Extrair características	AT1
AT3	Construir descritores	AT2
AT4	Persistir descritores	AT3
AT8	Receber objeto 3D de consulta	–
AT9	Obter descritores do objeto 3D de consulta	AT8
AT10	Calcular similaridade do objeto 3D recebido com os do banco de objetos	AT9
AT11	Ranquear e apresentar objetos 3D mais similares	AT10

Fonte: Elaborado pelo autor.

5 RESULTADOS OBTIDOS

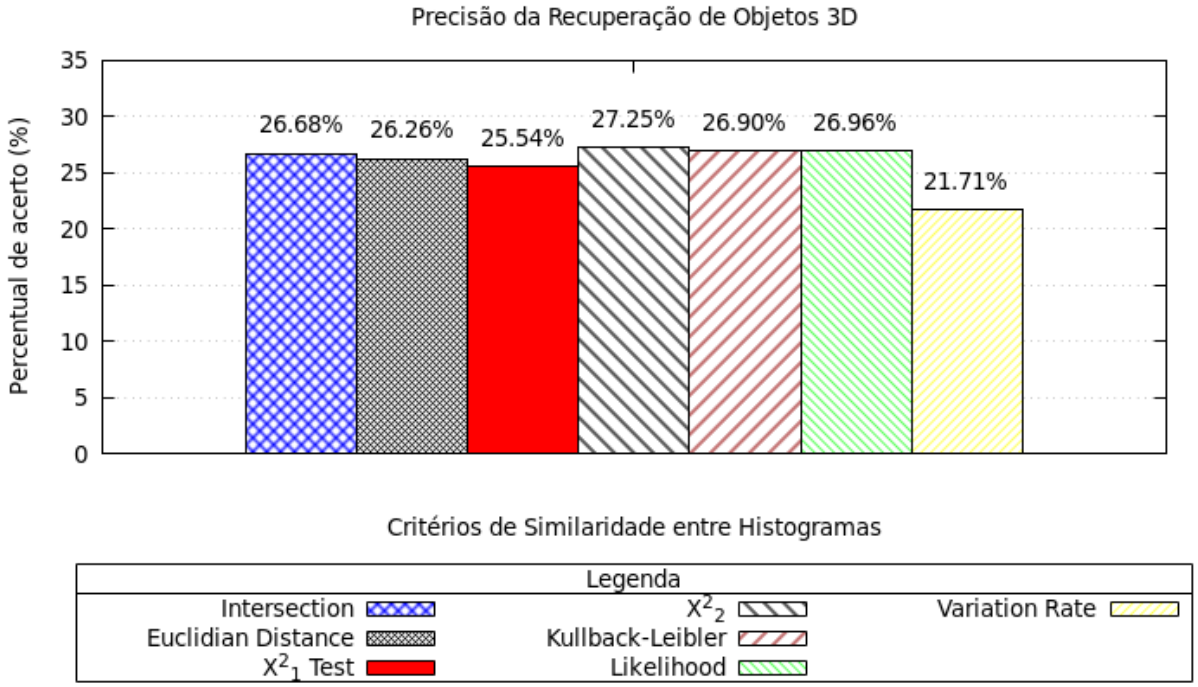
Neste capítulo são apresentados e discutidos os resultados obtidos para o treinamento e recuperação dos objetos 3D das 4 bases de objetos utilizadas. A seguir serão discutidos os resultados para cada base, de forma comparativa.

5.1 Análise dos Resultados da Base de Objetos 1

Como já foi mencionado, a base de objetos 1 possui 304 objetos distribuídos entre 20 classes. O agrupamento dos objetos em suas respectivas classes foi seguida a partir de um arquivo definido pelo Princeton Shape Benchmark.

Para esta base de dados, os algoritmos original (Figura 28) e proposto (Figura 29) apresentaram médias de acerto de 25.9% e 23.6%, respectivamente. Além disso, de maneira geral para os critérios de comparação entre histogramas, o algoritmo proposto apresentou menor desempenho.

Figura 28 – Algoritmo Original - Base de Objetos 1

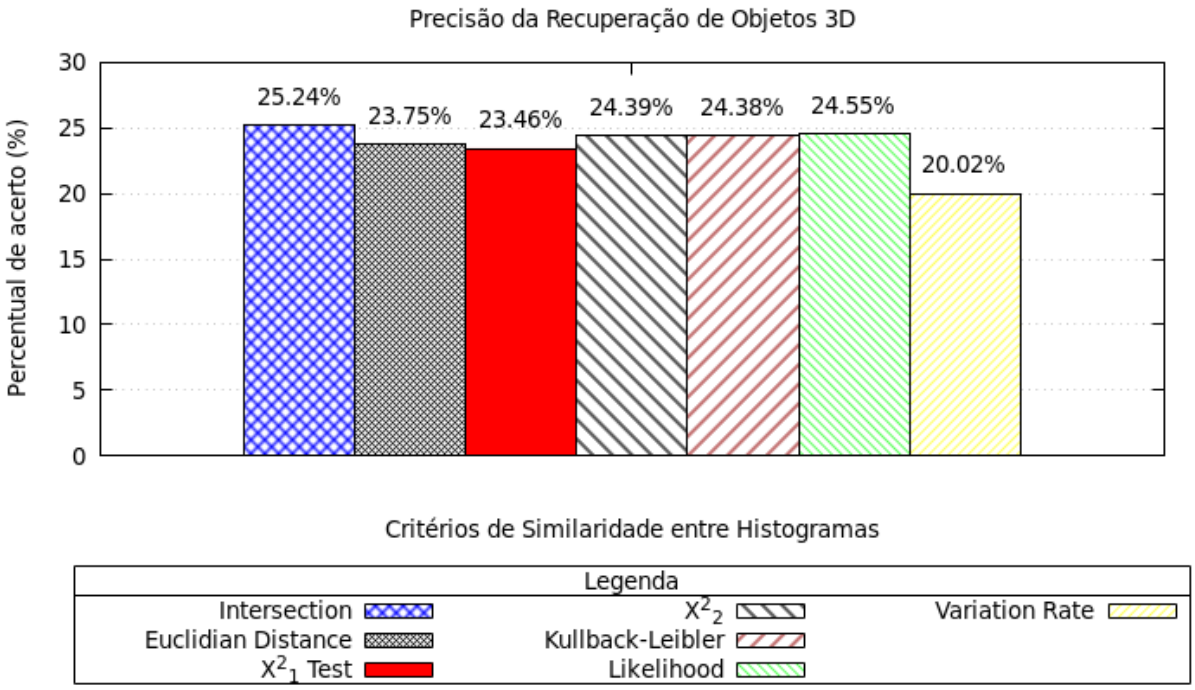


Fonte: Elaborado pelo autor.

O desempenho inferior a 30% de ambos algoritmos pode ser explicado pela quantidade de classes na base de objetos, pois quanto maior a variabilidade entre as formas dos objetos entre as classes, maior a chance do algoritmo gerar classificação incorreta. Por exemplo, podem haver objetos que em certas partes se assemelhem a objetos de classes distintas.

Já o desempenho do algoritmo proposto, para esta base de dados, pode ter apresentado menor desempenho devido à uma falta de descritividade da técnica. É possível que, ao aplicar a técnica de uma maneira distinta, haja uma melhora da taxa de acerto.

Figura 29 – Algoritmo Intersecção Esfera-*Surflet* - Base de Objetos 1



Fonte: Elaborado pelo autor.

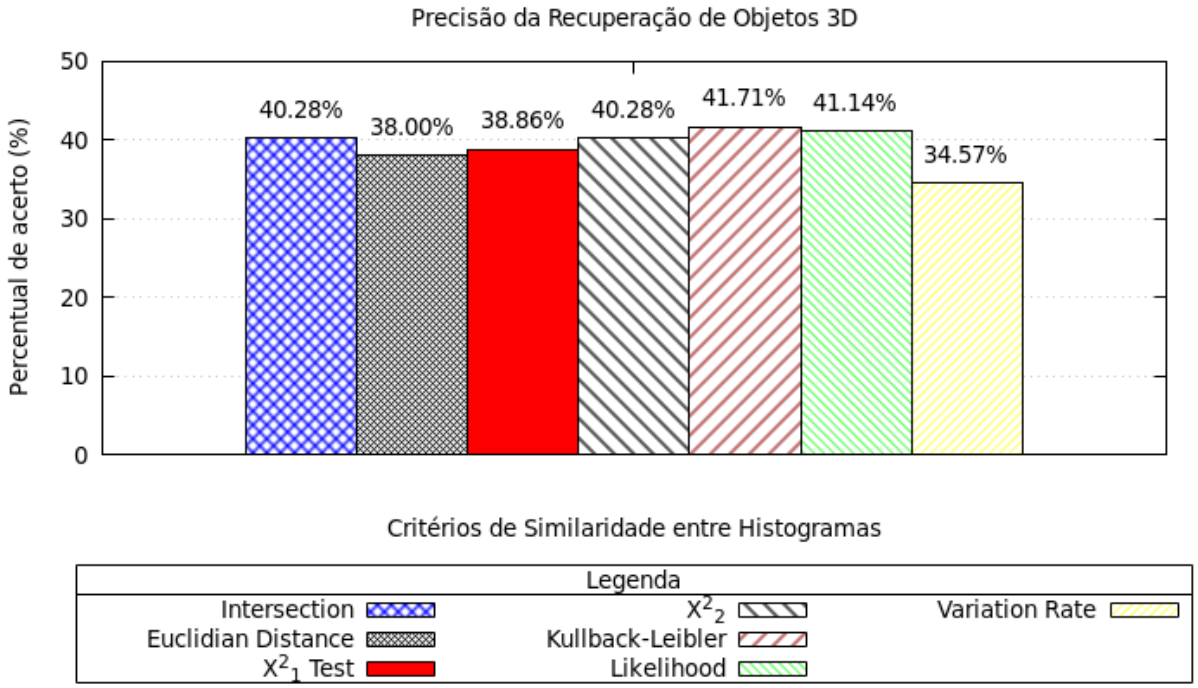
5.2 Análise dos Resultados da Base de Objetos 2

Esta base de objetos é uma amostra de 70 objetos da base de objetos 1, os quais são classificados em 14 classes. Para esta base de objetos, os algoritmos original (Figura 30) e proposto (Figura 31) apresentaram médias de acerto de 39.2% e 36.3%, respectivamente. Assim como nos resultados obtidos na base de objetos 1, de maneira geral os resultados apresentados pelo algoritmo proposto obtiveram menor desempenho.

Nesta base de objetos o desempenho supera a anterior que era inferior a 30%, aproximando-se de 40% de acerto. Isto afirma que o número de objetos e classes influencia diretamente na taxa de acerto de ambos algoritmos, uma vez que esta redução de objetos e classes diminui a probabilidade de uma recuperação incorreta.

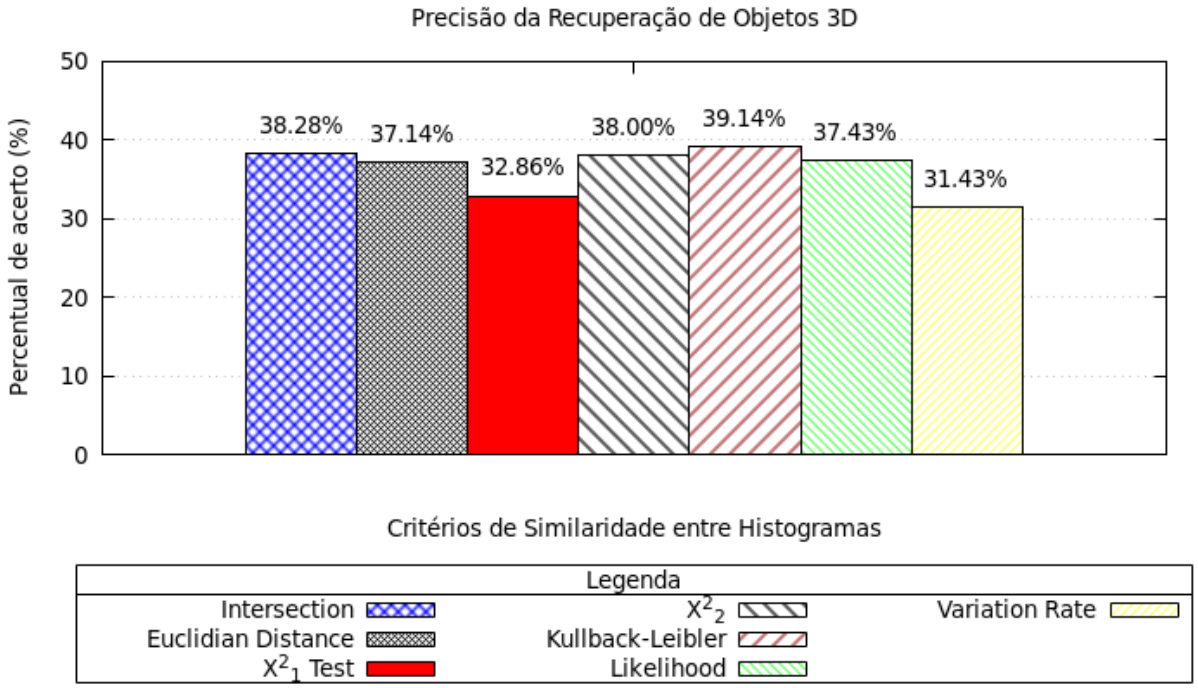
Como esta base é uma amostra da anterior, pode-se dizer que o desempenho do algoritmo proposto se comporta de maneira semelhante à anterior, pois pode possuir uma falta de descritividade em sua técnica. Além disso, não há como garantir que estes 70 objetos possuam realmente similaridade alta para pertencerem às suas respectivas classes, o que pode causar a baixa taxa de acerto.

Figura 30 – Algoritmo Original - Base de Objetos 2



Fonte: Elaborado pelo autor.

Figura 31 – Algoritmo Intersecção Esfera-*Surflet* - Base de Objetos 2



Fonte: Elaborado pelo autor.

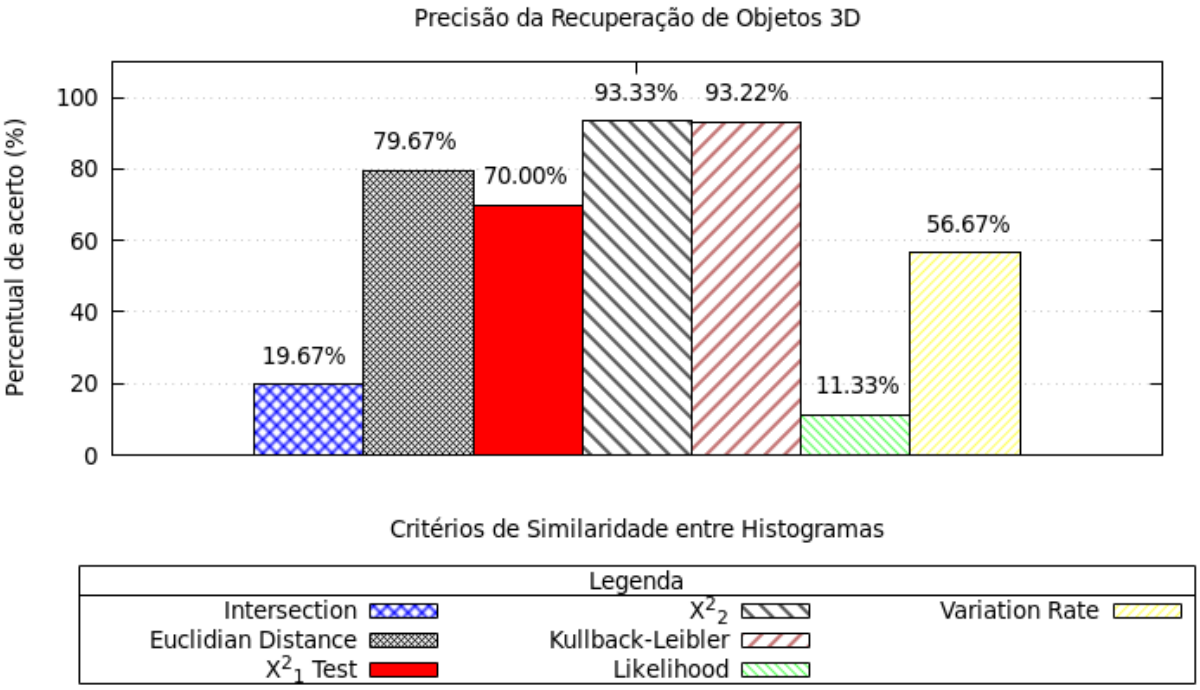
5.3 Análise dos Resultados da Base de Objetos 3

Para contornar as dúvidas deixadas pela classificação dos objetos imposta pelo benchmark de Princeton nas bases de objetos 1 e 2, esta e a base de objetos 4 foram criadas. Nesta base de objetos são apresentados 90 objetos distribuídos em 10 classes. Esta base de objetos é constituída por objetos com formas básicas, como: esferas, hexaedros, toróides, entre outros. Estes objetos são essenciais para a validação do algoritmo, uma vez sua forma e simetria se assemelha à partes de objetos reais mais complexos.

Este perfil de objetos não foi avaliado no trabalho original. Com isso, agora a técnica dos autores é avaliada de maneira mais abrangente, pois o algoritmo é testado para objetos com transformações de escala, rotação e translação.

Para esta base de objetos, os algoritmos original (Figura 32) e proposto (Figura 33) apresentaram médias de acerto de 60.5% e 60.2%, respectivamente. Embora a média obtida pelo algoritmo proposto tenha sido pior, o comportamento da taxa de acerto individualmente entre os critérios de similaridade obtiveram melhores resultados em 3 dos 7 critérios. Positivamente, os 3 critérios melhorados foram justamente os 3 melhores do algoritmo original, o que aumentou a taxa de acerto do algoritmo original de 93,3% para 95,8% do algoritmo proposto.

Figura 32 – Algoritmo Original - Base de Objetos 3

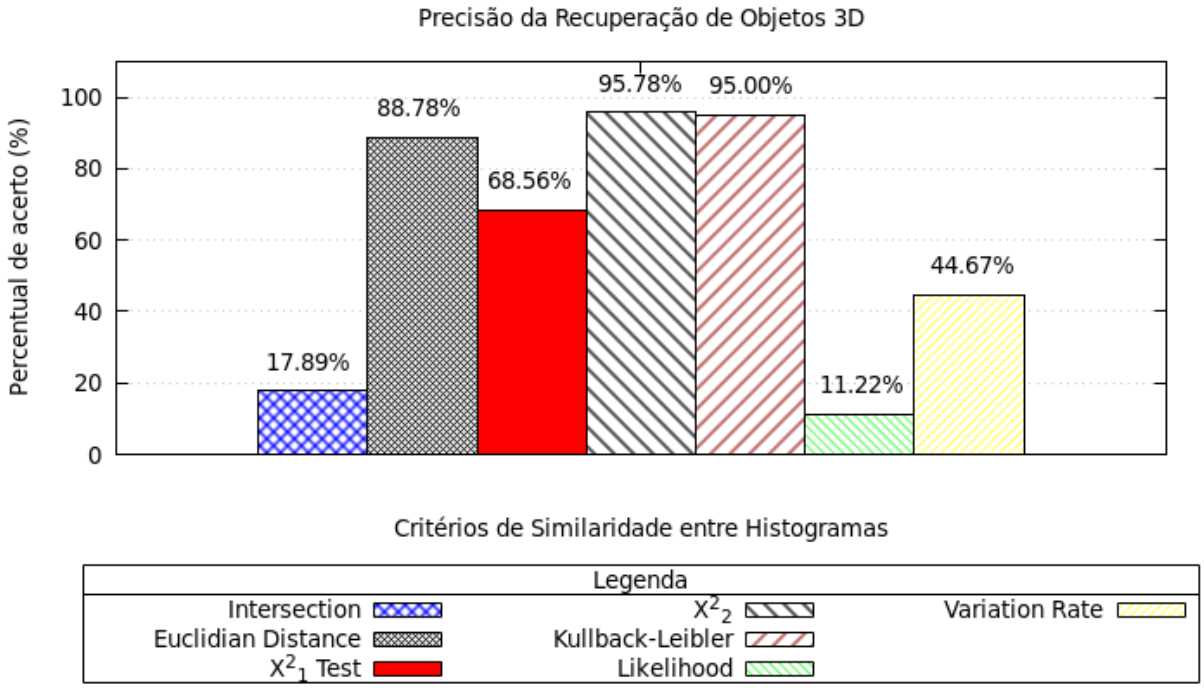


Fonte: Elaborado pelo autor.

A melhoria obtida no algoritmo proposto para esta base de objetos deve-se à forma simétrica dos objetos, pois ao envolvê-los na esfera as novas subcaracterísticas obtidas são

recompensadas pela sua simetria, a qual consegue mensurar a similaridade dos objetos em todos os seus lados.

Figura 33 – Algoritmo Intersecção Esfera-*Surflet* - Base de Objetos 3



Fonte: Elaborado pelo autor.

5.4 Análise dos Resultados da Base de Objetos 4

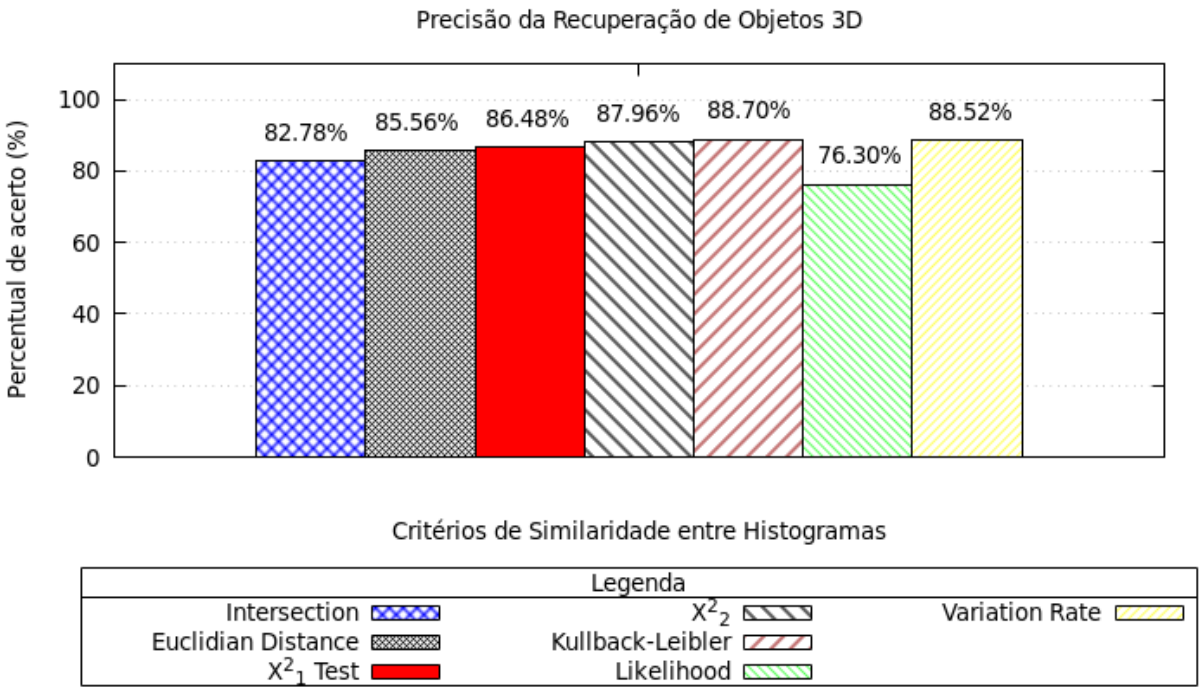
Assim como a base de objetos 3, esta base foi criada para viabilizar uma validação mais confiável dos algoritmos. Para isto, foram selecionadas 15 classes mistas das bases de objetos 2 e 3, aonde, para cada classe, 1 objeto foi obtido para geração de 5 versões suas modificadas. Desta forma, garante-se que um objeto possua características similares aos objetos da mesma classe. Além disso, esta base ilustra cenários mais realistas em aplicações práticas, pois o escaneamento 3D realizado por câmeras RGB-D inclui algum tipo de ruído.

Portanto, nesta base de objetos são apresentados 90 objetos distribuídos em 15 classes. Esta base é constituída por objetos com formas básicas e, também, por objetos complexos, como pessoas. Como já foi mencionado, estes objetos são essenciais para a validação do algoritmo, pois mescla objetos de forma simétrica e objetos mais complexos, do Princeton Shape Benchmark.

Para esta base de objetos, os algoritmos original (Figura 34) e proposto (Figura 35) apresentaram médias de acerto de 85.2% e 81.3%, respectivamente. Assim como na base

de objetos 3, nesta base os resultados do algoritmo original foram melhores que os resultados do algoritmo proposto, a partir de uma taxa de acertos média. Porém, ao analisar individualmente os critérios de similaridade, nota-se que 3 dos 7 critérios foram melhorados. Inicialmente o algoritmo original apresentava 88.7% de acerto no melhor caso, e o algoritmo proposto melhorou a taxa de acertos para 89.07%, no melhor caso.

Figura 34 – Algoritmo Original - Base de Objetos 4



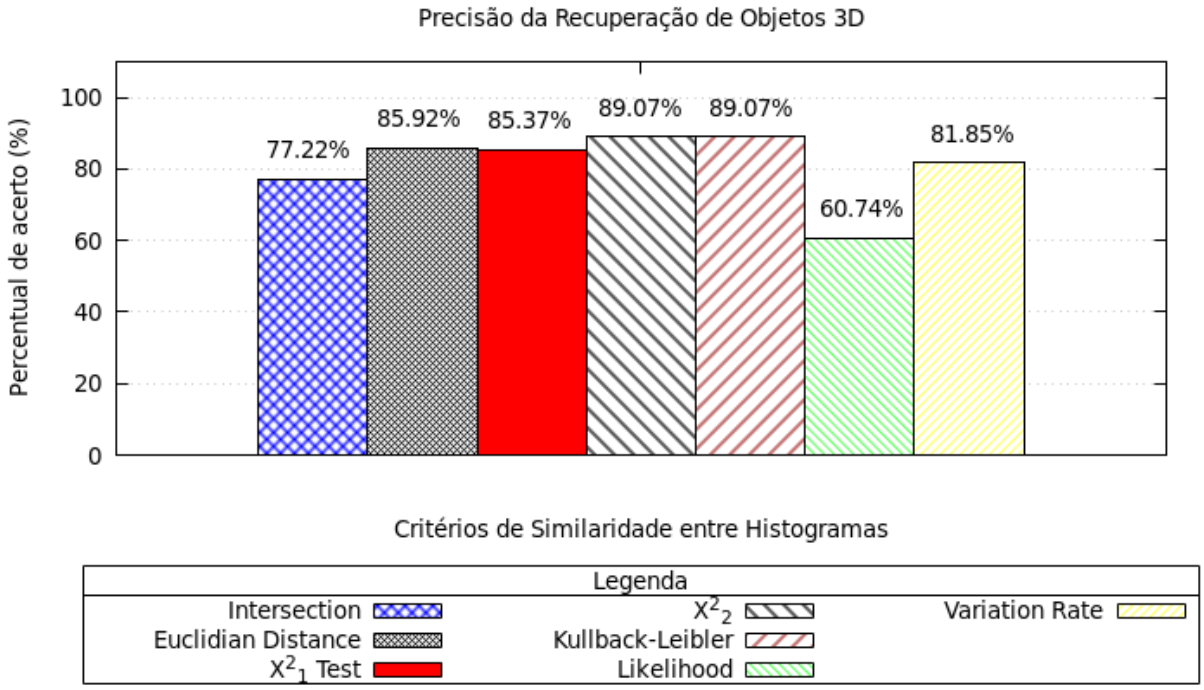
Fonte: Elaborado pelo autor.

5.5 Discussão Geral dos Resultados Alcançados

As métricas de comparação entre histogramas apresentam menor percentual de acerto nas bases de objetos de Princeton, o que pode representar o agrupamento equivocado dos objetos em suas respectivas classes, uma vez que a similaridade geométrica - baseada em forma - entre objetos é diferente da similaridade semântica - baseada em significado. Por exemplo, a classe pessoa pode conter pessoas com diferenças físicas (e.g, altas, baixas, gordas, magras, etc), que, embora sejam pessoas, possuem formas geométricas distintas. Além disso, os objetos do Princeton Shape Benchmark são mais complexos, apresentando diferentes pontos característicos. Por exemplo, as cadeiras da classe *Chair* apresentam similaridade com as mesas da classe *Table*, uma vez que estes dois objetos são constituídos, basicamente, por um conjunto de paralelepípedos.

Embora haja tais variações entre os critérios, principalmente nas bases 3 e 4, ficou claro a partir dos experimentos realizados que o critério *Kullback-Leibler* obteve resultados

Figura 35 – Algoritmo Intersecção Esfera-*Surflet* - Base de Objetos 4



Fonte: Elaborado pelo autor.

mais promissores para as bases de objetos utilizadas. Porém, os resultados variam com os implementados originalmente por (WAHL; HILLENBRAND; HIRZINGER, 2003). Por exemplo, o critério *Likelihood* possui baixo percentual de acerto de acordo com os experimentos realizados, sendo que para os autores originais o *Likelihood* é o melhor critério. A explicação mais plausível é que a base de teste utilizada pelos autores não tenha sido muito descritiva, uma vez que possui apenas 20 objetos.

Ao analisar os resultados do critério *Variation Rate*, o qual foi proposto neste trabalho, observa-se que seu melhor resultado é na base de objetos 4, pois as características obtidas desta base favorecem a comparação entre as taxas de variação entre as caixas do histograma nesta métrica, uma vez que é garantido que os objetos são similares. O mesmo não acontece nas bases de dados 1 e 2, pois os objetos possuem características mais dissimilares entre si do que as bases 3 e 4.

6 CONSIDERAÇÕES FINAIS

Uma vez que o número de objetos 3D está crescendo devido a popularização do uso de câmeras RGB-D, é uma tendência natural que a importância de algoritmos de recuperação de objetos 3D cresça.

Com isto, neste trabalho foram organizadas 2 bases de objetos com objetos do Princeton Shape Benchmark, bem como foram criadas 2 bases de objetos para validar os algoritmos sob cenários de escala, rotação, translação e ruído.

Devido à análise dos resultados, conclui-se que a métrica *Kullback-Leibler* é mais robusta que as demais, uma vez que apresentou os melhores resultados nos experimentos realizados em bases de dados distintas. Porém o critério *Likelihood* apresentou resultado insatisfatório para os testes realizados, ao contrário do que os autores originais apresentam.

A partir do desenvolvimento deste trabalho foi possível identificar os principais fatores que influenciam na extração de características dos objetos 3D, e também em como funciona a recuperação de objetos baseados no mapeamento dos descritores, uma vez que foram implementados e validados os 6 critérios de similaridade entre histogramas, além disso, um novo critério de similaridade foi proposto.

Embora tenha sido possível validar o algoritmo original, bem como propor uma nova subcaracterística, nota-se que há a viabilidade do aperfeiçoamento das técnicas implementadas, visando torná-las mais representativas em um cenário de grande volume de objetos e de grande variabilidade de formas.

Como trabalhos futuros vislumbra-se a criação de uma base de objetos real, escaneada por um sensor Kinect V2.0, visando também a investigação da taxa de acertos dos algoritmos original e proposto nesta base de objetos. Além do mais, outras técnicas de extração de características baseadas em relações geométricas entre os polígonos podem sem investigadas, implementadas e comparadas com os resultados deste trabalho.

Outra etapa importante a ser alcançada é implementar a versão atual dos algoritmos em C++ ou outra linguagem compilada, utilizando técnicas de otimização e paralelização para viabilizar maior velocidade na obtenção dos resultados. A utilização de C++ poderá contribuir significativamente para a pesquisa no tema, uma vez que permitirá mitigar o problema no uso da memória principal, o qual o Python traz consigo.

REFERÊNCIAS

ALDOMA, A. et al. Tutorial: Point cloud library: Three-dimensional object recognition and 6 dof pose estimation. **IEEE Robotics Automation Magazine**, v. 19, n. 3, p. 80–91, Sept 2012. ISSN 1070-9932. Citado na página 30.

BO, L.; REN, X.; FOX, D. Depth kernel descriptors for object recognition. In: IEEE. **Intelligent Robots and Systems (IROS), 2011. IEEE/RSJ International Conference**. San Francisco, 2011. p. 821–826. Citado 2 vezes nas páginas 41 e 42.

CHAOUCH, M.; VERROUST-BLONDET, A. A new descriptor for 2d depth image indexing and 3d model retrieval. In: IEEE. **Image Processing (ICIP), 2007. IEEE International Conference**. San Antonio, 2007. v. 6, p. VI–373. Citado 2 vezes nas páginas 40 e 42.

DESJARDINS, J. **What Happens in an Internet Minute in 2017?** 2017. Disponível em: <http://www.visualcapitalist.com/happens-internet-minute-2017/>. Acesso em: 25 mai. 2018. Citado 2 vezes nas páginas 24 e 25.

DESJARDINS, J. **What Happens in an Internet Minute in 2018?** 2018. Disponível em: <http://www.visualcapitalist.com/internet-minute-2018/>. Acesso em: 25 mai. 2018. Citado 2 vezes nas páginas 24 e 25.

FILLIAT, D. et al. Rgbd object recognition and visual texture classification for indoor semantic mapping. In: IEEE. **Technologies for Practical Robot Applications (TePRA), 2012. IEEE International Conference**. Woburn, 2012. p. 127–132. Citado na página 30.

GAO, Z. et al. Group-pair convolutional neural networks for multi-view based 3d object retrieval. **XXXII AAAI Conference on Artificial Intelligence**, 2018. Citado na página 42.

HIMMELSBACH, M.; LUETTEL, T.; WUENSCHE, H.-J. Real-time object classification in 3d point clouds using point feature histograms. In: IEEE. **Intelligent Robots and Systems (IROS), 2009. IEEE/RSJ International Conference**. Saint Louis, 2009. p. 994–1000. Citado na página 30.

HONG, Y.; KIM, J. A 2d-view depth image-and cnn-based 3d model identification method. **Applied Sciences**, Multidisciplinary Digital Publishing Institute, v. 7, n. 10, p. 988, 2017. Citado 2 vezes nas páginas 41 e 42.

LEBOEUF, K. **update: what happens in one Internet minute**. 2016. Disponível em: <http://www.excelacom.com/resources/blog/2016-update-what-happens-in-one-internet-minute>. Acesso em: 25 mai. 2018. Citado 2 vezes nas páginas 24 e 25.

LI, B.; JOHAN, H. 3d model retrieval using global and local radial distances. **Proceedings of the International Workshop On Advanced Image Technology (IWAIT)**, 2010. Citado 2 vezes nas páginas 40 e 42.

LOGOGLU, K. B.; KALKAN, S.; TEMIZEL, A. Cospair: colored histograms of spatial concentric surflet-pairs for 3d object recognition. **Robotics and Autonomous Systems**, Elsevier, v. 75, p. 558–570, 2016. Citado 3 vezes nas páginas 24, 41 e 42.

LÜ, K.; HE, N.; XUE, J. Content-based similarity for 3d model retrieval and classification. **Progress in Natural Science**, Elsevier, v. 19, n. 4, p. 495–499, 2009. Citado 3 vezes nas páginas 30, 40 e 42.

MAHMOUDI, S.; BENJELLOUN, M.; ANSARY, T. F. 3d objects retrieval using curvature scale space and zernike moments. **Journal of Pattern Recognition Research**, v. 6, n. 1, 2011. Citado 2 vezes nas páginas 41 e 42.

MICROSOFT, C. **Kinect for Windows Human Interface Guidelines v2.0**. 2014. Citado na página 30.

MONICA, R.; ALEOTTI, J.; CASELLI, S. A kinfu based approach for robot spatial attention and view planning. **Robotics and Autonomous Systems**, Elsevier, v. 75, p. 627–640, 2016. Citado na página 24.

NAPOLÉON, T.; SAHBI, H. From 2d silhouettes to 3d object retrieval: contributions and benchmarking. **EURASIP Journal on Image and Video Processing**, Springer, v. 2010, n. 1, p. 367181, 2010. Citado na página 24.

NIE, W. et al. 3d object retrieval based on spatial+lda model. **Multimedia Tools and Applications**, v. 76, p. 4091–4104, 2017. Citado 2 vezes nas páginas 41 e 42.

SCHERER, M.; WALTER, M.; SCHRECK, T. Histograms of oriented gradients for 3d object retrieval. Václav Skala-UNION Agency, 2010. Citado 2 vezes nas páginas 40 e 42.

SHILANE, P. et al. **The Princeton Shape Benchmark**. [S.l.]: Shape Modeling International, Genova, Italy, 2004. Disponível em: <http://shape.cs.princeton.edu/benchmark/>. Acesso em: 25 mai. 2018. Citado na página 43.

SONG, S.; LICHTENBERG, S. P.; XIAO, J. Sun rgb-d: A rgb-d scene understanding benchmark suite. In: IEEE. **Computer Vision and Pattern Recognition (CVPR), 2015. IEEE Computer Society Conference**. Boston, 2015. p. 567–576. Citado na página 30.

THESEN, T. et al. Neuroimaging of multisensory processing in vision, audition, touch, and olfaction. **Cognitive Processing**, v. 5, n. 2, p. 84–93, 2004. Citado na página 23.

WAHL, E.; HILLENBRAND, U.; HIRZINGER, G. Surflet-pair-relation histograms: a statistical 3d-shape representation for rapid classification. In: IEEE. **3-D Digital Imaging and Modeling (3DIM), 2003. Proceedings. Fourth International Conference**. Banff, 2003. p. 474–481. Citado 7 vezes nas páginas 33, 35, 39, 42, 47, 50 e 71.

WARDANI, D. E.; MOSTAFA, D. E.; TADONKI, C. Improving 3d shape retrieval methods based on bag-of-feature approach by using local codebooks. **International Journal of future Generation Communication and Networking**, v. 5, n. 4, p. 29–38, 2012. Citado 2 vezes nas páginas 41 e 42.

Apêndices

APÊNDICE A – BASE DE OBJETOS 1

Tabela 3 – Base de Objetos 1 - 304 Objetos Selecionados do *Princeton Shape Benchmark* e Classificados em 20 Classes.

Classe	**Total**	**Objetos**
Ship	10	m1437 m1427 m1432 m1428 m1431 m1430 m1433 m1439 m1429 m1434
Sword	9	m703 m690 m693 m714 m699 m697 m700 m689 m704
Couch Seat	10	m834 m830 m841 m831 m840 m836 m833 m829 m835 m839
Table	48	m920 m876 m889 m897 m903 m901 m874 m902 m877 m871 m898 m870 m884 m891 m882 m883 m911 m915 m881 m900 m887 m875 m890 m919 m879 m894 m905 m873 m910 m908 m893 m913 m895 m907 m912 m896 m917 m878 m916 m904 m918 m892 m899 m880 m906 m885 m914 m888
Head	11	m354 m352 m362 m364 m356 m361 m359 m351 m346 m357 m360
Vase	14	m527 m526 m540 m530 m538 m543 m535 m532 m534 m533 m541 m545 m537 m542
Helicopter	13	m1322 m1330 m1312 m1332 m1329 m1315 m1303 m1334 m1335 m1333 m1316 m1319 m1331
Shelves Furniture	22	m857 m862 m851 m855 m856 m848 m861 m849 m847 m866 m865 m858 m864 m860 m852 m853 m868 m845 m854 m850 m867 m863
Ice Cream	10	m755 m761 m752 m757 m758 m750 m760 m756 m759 m754
Flying Animals	10	m39 m38 m43 m44 m37 m34 m47 m46 m35 m42
Fish	14	m68 m61 m74 m69 m64 m70 m71 m72 m79 m58 m81 m73 m75 m78
Chair	20	m826 m810 m816 m824 m818 m815 m823 m827 m808 m813 m819 m809 m811 m821 m825 m814 m812 m807 m820 m817
Human	41	m213 m208 m158 m209 m171 m119 m177 m192 m127 m122 m189 m121 m163 m167 m136 m160 m154 m202 m159 m120 m124 m210 m176 m205 m166 m186 m165 m191 m175 m135 m203 m184 m215 m126 m172 m144 m185 m164 m123 m157 m161
Biplane Airplane	13	m1133 m1126 m1124 m1137 m1131 m1135 m1125 m1127 m1128 m1136 m1119 m1132 m1134
Lamp	9	m612 m613 m611 m607 m614 m605 m603 m608 m606
Boat	9	m1455 m1449 m1450 m1452 m1459 m1454 m1451 m1456 m1448
TV	9	m568 m566 m570 m563 m571 m561 m565 m569 m562
Potted Plant	14	m990 m993 m996 m994 m1034 m1030 m1036 m998 m1033 m992 m1031 m1029 m997 m1032
Comercial Airplane	16	m1158 m1154 m1153 m1158 m1164 m1165 m1164 m1166 m1150 m1151 m1153 m1154 m1150 m1166 m1165 m1151
Bottle	10	m488 m485 m489 m491 m483 m490 m492 m482 m484 m487

Fonte: Elaborado pelo autor.

APÊNDICE B – BASE DE OBJETOS 2

Tabela 4 – Base de Objetos 2 - 70 Objetos Selecionados do *Princeton Shape Benchmark* e Classificados em 14 Classes.

Classe	**Total**	**Objetos**
Ship	5	m1427 m1428 m1431 m1430 m1429
Sword	5	m690 m693 m699 m697 m689
Table	5	m871 m870 m891 m873 m888
Vase	5	m527 m526 m532 m534 m537
Ice_Cream	5	m755 m761 m752 m758 m750
Flying_Animals	5	m39 m43 m34 m46 m42
Fish	5	m71 m72 m79 m75 m78
Chair	5	m816 m815 m813 m814 m812
Human	5	m119 m122 m121 m120 m123
Biplane_Airplane	5	m1126 m1124 m1125 m1127 m1119
Lamp	5	m612 m613 m611 m607 m606
Potted_Plant	5	m990 m996 m994 m992 m997
Comercial_Airplane	5	m1154 m1164 m1150 m1151 m1153
Bottle	5	m485 m483 m482 m484 m487

Fonte: Elaborado pelo autor.

APÊNDICE C – BASE DE OBJETOS 3

Tabela 5 – Base de Objetos 3 - 90 Objetos Regulares Criados no *3D Builder* e Classificados em 10 Classes.

Classe	**Total**	**Objetos**
1_Dodecaedro	10	1_Dodecaedro 2_Dodecaedro 3_Dodecaedro 4_Dodecaedro 5_Dodecaedro 6_Dodecaedro 7_Dodecaedro 8_Dodecaedro 9_Dodecaedro 10_Dodecaedro
2_Hexaedro	10	1_Hexaedro 2_Hexaedro 3_Hexaedro 4_Hexaedro 5_Hexaedro 5_Hexaedro 7_Hexaedro 8_Hexaedro 9_Hexaedro 10_Hexaedro
3_Icosaedro	10	1_Icosaedro 2_Icosaedro 3_Icosaedro 4_Icosaedro 5_Icosaedro 6_Icosaedro 7_Icosaedro 8_Icosaedro 9_Icosaedro 10_Icosaedro
4_Octaedro	10	1_Octaedro 2_Octaedro 3_Octaedro 4_Octaedro 5_Octaedro 6_Octaedro 7_Octaedro 8_Octaedro 9_Octaedro 10_Octaedro
5_Tetraedro	10	1_Tetraedro 2_Tetraedro 3_Tetraedro 4_Tetraedro 5_Tetraedro 6_Tetraedro 7_Tetraedro 8_Tetraedro 9_Tetraedro 10_Tetraedro
6_Cilindro	10	1_Cilindro 2_Cilindro 3_Cilindro 4_Cilindro 5_Cilindro 6_Cilindro 7_Cilindro 8_Cilindro 9_Cilindro 10_Cilindro
7_Cone	10	1_Cone 2_Cone 3_Cone 4_Cone 5_Cone 6_Cone 7_Cone 8_Cone 9_Cone 10_Cone
8_Esfera	10	1_Esfera 2_Esfera 3_Esfera 4_Esfera 5_Esfera 6_Esfera 7_Esfera 8_Esfera 9_Esfera 10_Esfera
9_Toroide	10	1_Toroide 2_Toroide 3_Toroide 4_Toroide 5_Toroide 6_Toroide 7_Toroide 8_Toroide 9_Toroide 10_Toroide

Fonte: Elaborado pelo autor.

APÊNDICE D – BASE DE OBJETOS 4

Tabela 6 – Base de Objetos 4 - 90 Objetos, Sendo 15 Originais Obtidos das Bases 2 e 3, e, Para Cada, 5 Objetos Distorcidos Com Ruído.

Classe	**Total**	**Objetos**
Icosaedro	6	4-10_Icosaedro 1-10_Icosaedro 3-10_Icosaedro 10_Icosaedro 2-10_Icosaedro 0-10_Icosaedro
Hexaedro	6	1-10_Hexaedro 10_Hexaedro 4-10_Hexaedro 3-10_Hexaedro 2-10_Hexaedro 0-10_Hexaedro
Esfera	6	4-10_Esfera 1-10_Esfera 10_Esfera 2-10_Esfera 3-10_Esfera 0-10_Esfera
Ship	6	m1427 0-m1427 4-m1427 1-m1427 2-m1427 3-m1427
Toroide	6	4-10_Toroide 0-10_Toroide 2-10_Toroide 3-10_Toroide 10_Toroide 1-10_Toroide
Table	6	4-m870 1-m870 m870 3-m870 0-m870 2-m870
Cone	6	4-10_Cone 3-10_Cone 2-10_Cone 10_Cone 0-10_Cone 1-10_Cone
Helicopter	6	3-m1319 4-m1319 0-m1319 m1319 1-m1319 2-m1319
Flying Animals	6	4-m34 3-m34 2-m34 m34 1-m34 0-m34
Fish	6	4-m71 0-m71 1-m71 m71 3-m71 2-m71
Chair	6	1-m812 2-m812 0-m812 4-m812 3-m812 m812
Human	6	2-m119 m119 0-m119 1-m119 3-m119 4-m119
Biplane Airplane	6	0-m1119 2-m1119 1-m1119 3-m1119 4-m1119 m1119
Comercial Airplane	6	4-m1150 2-m1150 m1150 0-m1150 3-m1150 1-m1150
Bottle	6	0-m482 1-m482 4-m482 3-m482 m482 2-m482

Fonte: Elaborado pelo autor.

Anexos

ANEXO A – IMPLEMENTAÇÃO SIMPLIFICADA DOS CRITÉRIOS DE SIMILARIDADE ENTRE HISTOGRAMAS

```
1 def intersectionCriteria():
2     for obj in range(len(allHistograms)):
3         Ho = allHistograms[obj]
4         x = 0
5         for i in range(len(Ho')):
6             x += min(Ho[i], Ho'[i])
```

Algoritmo 1: Código Simplificado - Critério *Intersection*

```
1 def squaredEuclidianDistanceCriteria():
2     for obj in range(len(allHistograms)):
3         Ho = allHistograms[obj]
4         x = 0
5         for i in range(len(Ho')):
6             x += (Ho[i] - Ho'[i]) ** 2
```

Algoritmo 2: Código Simplificado - Critério *Euclidian Distance*

```
1 def x21Criteria():
2     for obj in range(len(allHistograms)):
3         Ho = allHistograms[obj]
4         x = 0
5         for i in range(len(Ho')):
6             x += ((Ho[i] - Ho'[i]) ** 2)/(Ho[i])
```

Algoritmo 3: Código Simplificado - Critério X_1^2 *Test*

```
1 def x22Criteria():
2     for obj in range(len(allHistograms)):
3         Ho = allHistograms[obj]
4         x = 0
5         for i in range(len(Ho')):
6             x += ((Ho[i] - Ho'[i]) ** 2)/(Ho[i] + Ho'[i])
```

Algoritmo 4: Código Simplificado - Critério X_2^2 *Test*

```
1 def kullbackLeiblerDivergenceCriteria():
2     for obj in range(len(allHistograms)):
3         Ho = allHistograms[obj]
4         x = 0
5         for i in range(len(Ho')):
6             x += (Ho[i] - Ho'[i]) * math.log(Ho'[i]/Ho[i])
```

Algoritmo 5: Código Simplificado - Critério *Kullback-Leibler*

```
def likelihoodCriteria():
    for obj in range(len(allHistograms)):
        Ho = allHistograms[obj]
        x = 0
        for s in range(len(caracteristicas)):
            a = int((caracteristicas[s][0]*10)/2.0) % 5
            b = int((caracteristicas[s][1]*10)/2.0) % 5
            c = int((caracteristicas[s][2]*10)/2.0) % 5
            d = int((caracteristicas[s][3]*10)/2.0) % 5
            ind = (a*(5**3))+(b*(5**2))+(c*(5))+(d)
            x += math.log(Ho[ind])
```

Algoritmo 6: Código Simplificado - Critério *Likelihood*

```
def variationRateCriteria(self, criterio):
    txVarHo' = self.calcula_taxa_de_variacao(Ho')
    for obj in range(len(self.todosHist)):
        Ho = self.todosHist[obj]
        txVarHo = self.calcula_taxa_de_variacao(Ho)
        x = 0
        for i in range(len(txVarHo')):
            if (((txVarHo'[i] > 0) and (txVarHo[i] > 0)) or ((txVarHo'[i]
                < 0) and (txVarHo[i] < 0)) or (txVarHo'[i] == txVarHo[i])):
                x += abs(txVarHo'[i] - txVarHo[i])
            else:
                x += pow(txVarHo'[i] - txVarHo[i], 2)
        dissimilaridade[Ho[0]] = x

def calcula_taxa_de_variacao(self, Ho):
    for i in range(len(Ho)-1):
        var = (Ho[i+1] - Ho[i])
        tx = (var * 100)/Ho[i]
        txVarHo.append(tx)
    return txVarHo
```

Algoritmo 7: Código Simplificado - Critério *Variation Rate*