Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

DETECÇÃO DE ROSTOS EM NUVENS DE PONTOS

Victor Fabre Figueiredo

Monografia apresentada como requisito parcial
para conclusão do Curso de Engenharia da Computação

Orientador
Prof. Dr. Ricardo Lopes de Queiroz

Brasília
2019

Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

DETECÇÃO DE ROSTOS EM NUVENS DE PONTOS

Victor Fabre Figueiredo

Monografia apresentada como requisito parcial
para conclusão do Curso de Engenharia da Computação

Prof. Dr. Ricardo Lopes de Queiroz (Orientador)
CIC/UnB

Prof. Dr. Camilo Chang Dórea
Universidade de Brasília

Prof. Dr. Eduardo Peixoto Fernandes da Silva
Universidade de Brasília

Prof. Dr. José Edil
Coordenador do Curso de Engenharia da Computação

Brasília, 05 de julho de 2019

Dedicatória

Dedico esse trabalho aos meus pais por sempre me apoiarem e por tornarem possível a realização de um projeto de vida e para que eu chegasse neste patamar acadêmico. Aos meus irmãos, pois sei que com eles posso sempre contar. E à minha namorada, que é meu porto seguro.

Agradecimentos

Um forte agradecimento a todos os meus professores do ensino básico ao ensino médio por terem me ensinado e formado a minha base para que eu pudesse chegar até aqui.

Um especial agradecimento aos meus professores da Universidade que me apoiaram e acreditaram em mim, além de me passar o conhecimento necessário para que eu pudesse realizar esse trabalho e despertar em mim o desejo de seguir a carreira acadêmica.

Agradeço a toda a minha família, àqueles que estão comigo e àqueles que já partiram. Só consegui chegar aqui com o apoio e incentivo de vocês.

Ao meu pai, Ricardo, por sempre me apoiar nas minhas decisões e me incentivar a seguir o caminho que eu gosto.

À minha mãe, Alice, que sempre me ajudou nos estudos e foi sempre um exemplo a ser alcançado.

Aos meus irmãos, Felipe e Henrique, por serem meus melhores amigos e sempre me acompanharem nas aventuras da vida.

Quero agradecer também à Universidade de Brasília, a qual me proporcionou o ambiente necessário para obter a minha formação e aos meus colegas de curso que foram essenciais na construção do meu aprendizado.

Ao meu colega, M.Sc. Gustavo Sandri, pela paciência em me ensinar e me ajudar no desenvolvimento deste projeto.

Por fim, gostaria de agradecer ao meu orientador, professor Dr. Ricardo Lopes de Queiroz, por me permitir realizar este trabalho e me acompanhar na minha pesquisa.

Resumo

Este projeto apresenta a detecção de rostos em nuvens de pontos, utilizando projeções bidimensionais. Uma nuvem de pontos é um conjunto de pontos representados em um sistema de coordenadas tridimensionais no espaço e comumente tem o objetivo de representar a superfície externa de um objeto ou de uma cena. Devido à grande revolução que ocorreu nos últimos anos na maneira como conseguimos representar o mundo a nossa volta, a qual inclui a criação das nuvens de pontos, vê-se necessário métodos para a detecção de rostos e outras regiões de interesse nessa nova tecnologia. O principal objetivo do projeto é desenvolver uma técnica que permita a aplicação de métodos de processamento de imagens, tais como a identificação de rostos, em nuvens de pontos. O *framework* proposto busca atingir os objetivos, utilizando métodos já conhecidos de processamento de imagens bidimensionais em projeções e, a partir daí, extrapolar esses resultados para as nuvens de pontos tridimensionais. Também é proposto um método que utiliza a consistência temporal entre uma sequência de quadros, para diminuir o custo computacional, analisando sequências, e para tentar melhorar os resultados. O que foi buscado no projeto foi encontrar regiões de interesse nas nuvens de pontos mais especificamente rostos. Isso pode ser desejado para, por exemplo, realizar a compressão da nuvem de pontos utilizando a codificação da região de interesse. Os testes foram realizados em diversas sequências de nuvens de pontos *voxelizadas*. Os resultados experimentais mostraram que foi possível extrair a região de interesse desejada das nuvens de pontos utilizando as projeções bidimensionais de maneira bem sucedida.

Palavras-chave: Nuvem de pontos, Viola-Jones, região de interesse, consistência temporal

Abstract

This project proposes a face detection method in point clouds using two dimensional projections. Point cloud is a set of points represented in a three-dimensional space that commonly has the purpose of representing the external surface of an object or a scene. Due to the revolution in the way that we can represent the world that surround us, this includes the advent of point clouds, that took place in the last few years, it is clear the necessity to detect faces and other regions of interest in this new technology. The main objective of the project is to develop a technique that allows to use bi-dimensional image processing methods in point clouds such as face identification. The proposed framework seeks to achieve these goals using the already well-known methods of image processing in two dimensional projections and then extrapolate these results to three dimensional point clouds. A method that uses the temporal consistency between a sequence of frames is also proposed in the intent to reduce the computational cost of analyzing those sequences and to improve the results. The specific goal of this work was to identify regions of interest in the point cloud. In this work, the region of interest is defined to be the subject's face in the point cloud. The detection of faces can be useful to compress the point cloud using the region of interest coding. The tests were made with different voxelized point cloud sequences. The experimental results show that the proposed objectives were reached with success.

Keywords: Point cloud, Viola-Jones, region of interest, temporal consistency

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas e Siglas

AR Realidade Aumentada.

MPEG Moving Picture Experts Group.

MR Realidade Mista.

RGB Red Green Blue.

ROI Região de Interesse.

VR Realidade Virtual.

Capítulo 1

INTRODUÇÃO

Ao longo dos últimos anos foi feita uma grande revolução na maneira como conseguimos representar o mundo a nossa volta devido a grandes avanços na computação gráfica, criando tecnologias como a realidade virtual (VR), realidade aumentada (AR) e realidade mista (MR) [1]. Esses sistemas de interação utilizam modelos tridimensionais para representar cenas ou objetos. Uma das maneiras como esses modelos podem ser representados é por meio das nuvens de pontos (do inglês *point clouds*).

As nuvens de pontos, que representam o mundo 3D por amostragem, tem se tornado cada vez mais importantes nos últimos anos, em virtude da proliferação de imagens computacionais voltadas para a representação 3D.

Uma nuvem de pontos pode ser interpretada como o equivalente a uma imagem 3D e uma sequência de quadros de uma nuvem de pontos pode ser interpretada como o equivalente a um vídeo 3D. Assim como em imagens e vídeos, diferentes informações da cena podem ser adquiridas a depender da aplicação. Algumas informações que podem ser desejadas são:

- Segmentação por cores;
- Definição de regiões de interesse;
- Identificação de objetos;
- Reconhecimento facial.

Já existem algoritmos [2] [3] [4] bem conhecidos e que produzem bons resultados para extrair algumas dessas informações de imagens e vídeos 2D. No entanto, extrair tais informações de nuvens de pontos ainda se mostra um desafio.

Portanto, o principal objetivo do projeto é desenvolver uma técnica que permita a aplicação de métodos de processamento de imagens, tais como a identificação de rostos, em nuvens de pontos.

Os resultados obtidos pelo presente trabalho possibilitaram o desenvolvimento do trabalho *"Point Cloud Compression Incorporating Region of Interest Coding"* [5] publicado no *IEEE Int'l Conf. Image Processing (ICIP)* de 2019.

1.1 Motivação

Como relatado, a área de pesquisa em nuvem de pontos é uma área com muito espaço para crescimento e desenvolvimento de novas técnicas ou para aplicação de técnicas existentes para imagens e vídeos 2D em nuvens de pontos.

Assim como imagens e vídeos, as nuvens de pontos e as nuvens de pontos dinâmicas contêm grande quantidade de dados. Portanto, a compressão de nuvens de pontos é necessária para qualquer aplicação prática. Atualmente, o *Moving Picture Experts Group* (MPEG) está padronizando um formato de compressão de nuvem de pontos (PCC) com essa finalidade [6].

Como as imagens e vídeos, as nuvens de pontos geralmente têm regiões de interesse (ROI) que têm significado ou relevância especiais - por exemplo, rostos - para os quais a preservação da alta fidelidade durante a compressão pode ser importante. Para imagens e vídeos, a compressão por região de interesse ou *codificação ROI* é bem estudada. (Veja, por exemplo, [7]). No entanto, para nuvens de pontos, não há vasta literatura disponível sobre compressão por região de interesse.

Dessa maneira, a motivação deste trabalho surgiu da necessidade de desenvolver um algoritmo para identificar as regiões de interesse de forma que fosse possível desenvolver um projeto de compressão por região de interesse para nuvens de pontos, como foi feito em [5].

1.2 Objetivos

1.2.1 Objetivo geral

O presente projeto tem como finalidade desenvolver uma técnica de identificação de regiões de interesse em nuvens de pontos, utilizando os métodos já conhecidos e bem estabelecidos para a detecção em duas dimensões e extrapolando tais métodos para três dimensões.

Propõe-se atingir o objetivo ao utilizar projeções bidimensionais da nuvem de pontos a partir de múltiplas vistas e aplicar os métodos já existentes nessas projeções. Dessa forma, os resultados obtidos nas projeções 2D devem ser reprojetados na nuvem de pontos.

1.2.2 Objetivos específicos

- Implementar um método para que resultados obtidos em uma projeção possam ser transportados para a nuvem de pontos.
- Aplicar um método de detecção de regiões de interesse nas projeções.
- Combinar os resultados de múltiplas projeções na nuvem de pontos.
- Implementar um método para que a consistência temporal de uma sequência de quadros de uma nuvem de pontos possa ser explorada na identificação da região de interesse.
- Analisar os resultados obtidos.

1.3 Apresentação do projeto

No Capítulo 2 é feita uma revisão bibliográfica sobre as ferramentas utilizadas para o desenvolvimento do projeto.

O Capítulo 3 apresenta a metodologia utilizada e uma descrição detalhada da implementação do algoritmo desenvolvido.

Os resultados experimentais são apresentados e discutidos no Capítulo 4 e as conclusões são expostas no Capítulo 5.

Capítulo 2

REVISÃO BIBLIOGRÁFICA

Assim como imagens e vídeos, as nuvens de pontos geralmente possuem Regiões de Interesse (ROI) que têm significado ou relevância especiais - por exemplo, rostos - para diversas aplicações nas áreas de compressão, visão computacional, médica, navegação autônoma de veículos, entre outras.

Para imagens e vídeos, o uso de regiões de interesse já é bem difundido em diversas áreas [7][8]. No entanto, para nuvens de pontos ainda existem áreas, como a compressão, onde não há literatura disponível sobre o uso de regiões de interesse. Para a detecção de rostos ou classificação de objetos em nuvens de pontos existe uma vasta literatura disponível com diferentes métodos de detecção [9][10][11][12][13].

Neste projeto, optou-se por uma nova abordagem para a detecção de regiões de interesse nas nuvens de ponto a partir de projeções 2D e a detecção nessas projeções utilizando o bem conhecido algoritmo de Viola-Jones [2].

Nas próximas seções, será apresentada uma revisão bibliográfica sobre nuvens de pontos, região de interesse e as demais técnicas utilizadas para o desenvolvimento deste projeto.

2.1 Nuvem de Pontos

Com o surgimento de conteúdo digital tridimensional, foi necessário o desenvolvimento de maneiras para representá-lo que possibilitassem seu consumo direto por seres humanos, uma das maneiras desenvolvida para realizar tal representação é a nuvem de pontos.

Nuvem de pontos (do inglês *point cloud* ou PC) é um conjunto de pontos no espaço representados em um sistema de coordenadas tridimensional (X, Y, Z) e comumente têm o objetivo de representar a superfície externa de um objeto ou uma cena. A Figura (2.1) traz um exemplo de nuvem de pontos.

Figura 2.1: Nuvem de pontos RomanOilLight (Fonte: [14]).

As nuvens de pontos consistem em geometria e todos os seus atributos [15]. A parte geométrica de uma nuvem de pontos com N pontos, pode ser descrita por um conjunto V que contém as coordenadas de todos os pontos, tal que:

$$V = \{v_1, v_2, \ldots, v_n\} = \left\{ \begin{array}{c} (x_1, y_1, z_1) \\ (x_2, y_2, z_2) \\ \vdots \\ (x_n, y_n, z_n) \end{array} \right\} \tag{2.1}$$

Onde $n = 1, \ldots, N$ e $v_i = (x_i, y_i, z_i)$ define a posição do ponto p_i.

Os atributos de uma nuvem de pontos podem ser representados de maneira similar por um conjunto C em que cada entrada deste conjunto possui D atributos. Chega-se então ao conjunto:

$$C = \{c_1, c_2, \ldots, c_n\} = \left\{ \begin{array}{c} (a_{11}, \ldots, a_{1D}) \\ (a_{21}, \ldots, a_{2D}) \\ \vdots \\ (a_{n1}, \ldots, a_{nD}) \end{array} \right\} \tag{2.2}$$

Comumente, os atributos incluem componentes de cor, mas podem também incluir transparência, normais, vetores de movimento e mais. Uma vez que a geometria é dada, os atributos podem ser vistos como um sinal definido por um conjunto de pontos.

Baseando-se nas Equações (2.1) a (2.2), os conjuntos V e C podem ser representados por matrizes. Dessa forma, é obtida uma lista de pontos (coordenadas espaciais) e uma lista de atributos cujos itens são pareados um a um com os pontos da lista de posições. O resultado é uma representação esparsa do conteúdo visual do objeto representado.

O ponto na nuvem de pontos é a noção primitiva na qual a geometria é construída. Ele não possui comprimento, área ou volume, ele é usado como uma localização única no espaço Euclidiano. No entanto, para renderização, os pontos são comumente representados como esferas. Empregar nuvens de pontos *voxelizadas* (VPC) é mais conveniente para a representação de tais dados.

Um *voxel* é a extensão 3D de um pixel, enquanto o pixel é um quadrado e o elemento fundamental de uma imagem 2D, o *voxel* é um cubo e é o elemento fundamental de um objeto 3D.

Assumimos que o objeto representado está contido em um cubo de tamanho $L \times L \times L$, onde L é um inteiro positivo. Podemos dividir esse cubo nas três dimensões L vezes, obtendo-se L^3 cubos de dimensão $1 \times 1 \times 1$, que são por definição *voxels*. A vantagem de tal abordagem é que a posição de cada *voxel* é discreta e não contínua e o elemento fundamental não é mais adimensional.

2.2 Região de Interesse

Uma Região de Interesse (ROI) é definida por uma amostra de um conjunto de dados identificado para uma finalidade específica [8]. O conceito de ROI é comumente usado em muitas áreas de aplicação, por exemplo, em visão computacional e reconhecimento óptico de caracteres (OCR). A ROI define as bordas de um objeto em consideração.

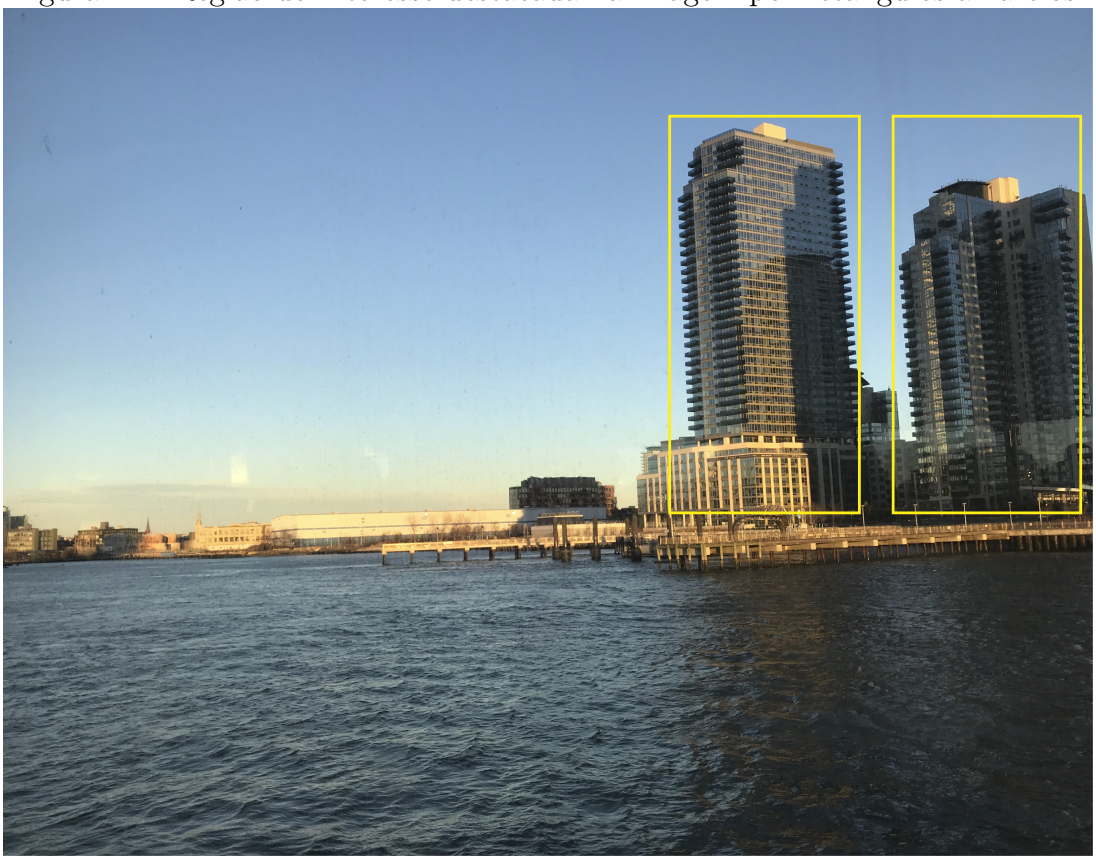
A região de interesse pode ser definida em um conjunto de dados unidimensional, bidimensional (ver exemplo na Figura (2.2)), tridimensional ou quadrimensional. Exemplos de regiões de interesse de cada um desses tipos são dados abaixo.

- Conjunto de dados 1D: intervalo de frequência de um formato de onda.
- Conjunto de dados 2D: contorno de um objeto em uma imagem.
- Conjunto de dados 3D: contorno de uma superfície de um objeto em um volume.
- Conjunto de dados 4D: contorno de um objeto durante um intervalo de tempo em um volume de tempo.

As regiões de interesse podem ser criadas manualmente, de maneira semiautomática ou automaticamente via software. Diferencia-se:

- ROI manual: o usuário define manualmente uma área usando teclado, mouse ou outro acessório que possibilite a delimitação.
- ROI semiautomática: o software ajuda o usuário no desenho ou delimita a região com base em parâmetros fornecidos pelo usuário.
- ROI automática: o software determina os critérios da delimitação sem intervenção do usuário.

Figura 2.2: Região de Interesse destacada na imagem por retângulos amarelos.



Regiões de interesse são muito utilizadas na compressão de imagens, como no padrão *JPEG2000* e vídeos [7]. Com a detecção de região de interesse em nuvem de pontos pode-se utilizar isso para a compressão com uma melhor qualidade destas.

2.3 Viola-Jones

Esta seção contém uma descrição do funcionamento detalhado do algoritmo utilizado para a detecção de rostos no projeto. O algoritmo de Viola-Jones foi escolhido por duas prin-

cipais razões, seu baixo custo computacional, necessário por causa da grande quantidade de dados a ser processada, e sua alta taxa de acertos com baixa taxa de falsos positivos.

O *framework* de detecção de objetos Viola-Jones foi o primeiro *framework* de detecção de objetos a fornecer taxas competitivas de detecção de objetos em tempo real. Foi proposto por Paul Viola e Michael Jones em 2001 [2].

O problema que deve ser solucionado é a detecção de rostos em uma imagem, uma tarefa facilmente realizada por um ser humano, mas que para um computador são necessárias instruções e restrições precisas. Para diminuir a complexidade, o *framework* exige rostos frontais com visualização completa, ou seja, o rosto deve apontar para a câmera e não pode estar inclinado.

2.3.1 Funcionamento do algoritmo

O algoritmo de Viola-Jones possui características que o classificam como um bom algoritmo de detecção, elas são:

- Robustez: taxa de detecção alta com poucos falsos positivos e negativos.
- Tempo real: consegue detectar faces a 15 quadros por segundo.
- Apenas detecção de rosto: o objetivo é separar rosto de não-rosto.

O algoritmo pode ser dividido em três etapas:

1. Criação de uma imagem integral.
2. Treinamento AdaBoost.
3. Classificadores em cascata.

O procedimento de detecção de objetos do algoritmo classifica imagens com base no valor de atributos simples. A principal motivação para o uso de atributos ao invés de pixels diretamente é que os atributos codificam o conhecimento de domínio ad-hoc o que é difícil de aprender usando uma quantidade finita de dados de treinamento. Outra justificativa dos autores para o uso de atributos foi que um sistema baseado em atributos é muito mais rápido do que um sistema baseado em pixels.

Os atributos utilizados possuem semelhanças com as funções base de Haar que foram usadas em outros artigos para detecção de objetos [16].

Figura 2.3: Exemplo de retângulos de atributos relativo a janela de detecção. A soma dos pixels no retângulo branco é subtraída da soma dos pixels no retângulo cinza (Fonte: [17]).

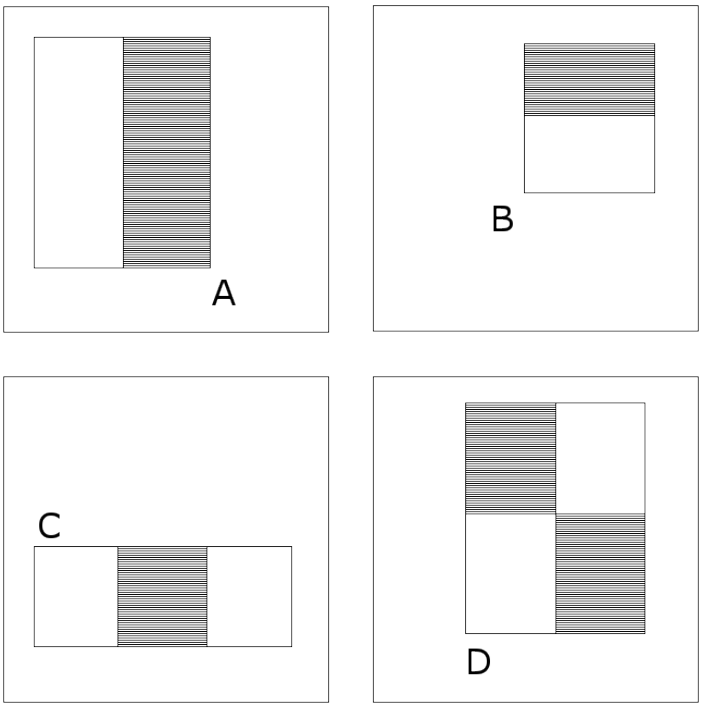


Imagem Integral

Os autores utilizaram o que foi chamado de imagem integral para computar rapidamente os atributos retangulares. Um ponto (x, y) de uma imagem integral contém a soma dos pixels acima e à esquerda dele, como visto na Equação (2.3), onde $ii(x, y)$ é a imagem integral e $i(x, y)$ é a imagem original.

$$ii(x, y) = \sum_{x' \leq x, y' \leq y} i\left(x', y'\right) \tag{2.3}$$

Utilizando as Equações (2.4) a (2.5) a imagem integral pode ser calculada com apenas uma passada na imagem original.

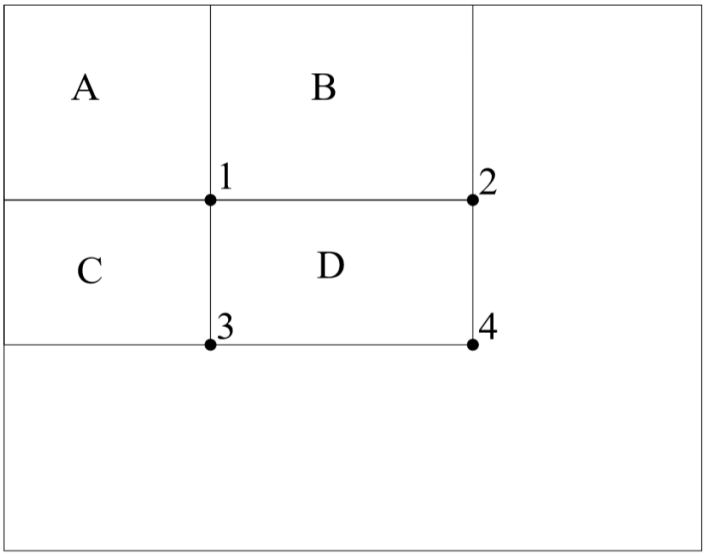
$$s(x, y) = s(x, y - 1) + i(x, y) \tag{2.4}$$

$$ii(x, y) = ii(x - 1, y) + s(x, y) \tag{2.5}$$

O uso de imagens integrais é importante pois ele permite que qualquer soma retangular possa ser calculada com apenas quatro referências à matriz, como mostrado na

Figura (2.4).

Figura 2.4: A soma dos pixels no retângulo D pode ser calculada com quatro referências à matriz. O valor da imagem integral em 1 é a soma dos pixels no retângulo A. O valor em 2 é $A + B$, em 3 é $A + C$, e em 4 é $A + B + C + D$. A soma de D pode ser calculada como $4 + 2 - (2 + 3)$ (Fonte: [17]).



Treinamento AdaBoost

O sistema implementado por Viola e Jones utiliza uma variante do AdaBoost para selecionar os atributos e treinar os classificadores [18]. Na sua forma original, o algoritmo de aprendizado AdaBoost é usado para melhorar a performance de um algoritmo de aprendizado simples. Ele faz isso ao combinar uma coleção de funções de classificação fracas para construir um classificador forte.

Como existem mais de 180 mil retângulos de atributo associados a cada imagem, é proposta a hipótese de que um pequeno número desses atributos pode ser combinado para formar um classificador efetivo. O maior desafio é achar esses atributos.

Para isso, os autores projetaram o algoritmo de aprendizado fraco para selecionar um único retângulo de atributo que melhor separa os exemplos positivos e negativos. Para cada atributo, é determinado uma função de classificação de *threshold* ótima, de forma que o menor número de exemplos é classificado erroneamente. Um classificador fraco $h_j(x)$ consiste de um atributo f_j, um *threshold* θ_j e uma paridade p_j indicando a direção do sinal de inequalidade, como visto na Equação (2.6).

$$h_j(x) = \begin{cases} 1 & \text{se } p_j f_j(x) < p_j \theta_j \\ 0 & \text{caso contrário} \end{cases} \tag{2.6}$$

Abaixo uma versão simplificada do algoritmo de aprendizado é apresentada:

- Dado um conjunto de imagens exemplo $(x_1, y_1), \ldots, (x_n, y_n)$, onde $y_i = 0, 1$ para exemplos negativos e positivos, respectivamente.
- Inicialize os pesos $w_{1,i} = \frac{1}{2m}, \frac{1}{2l}$ para $y_i = 0, 1$, respectivamente, onde m e l são o número de negativos e positivos, respectivamente.
- Para $t = 1, \ldots, T$:
  1. Normalize os pesos,
  $$w_{t,i} \leftarrow \frac{w_{t,i}}{\sum_{j=1}^{n} w_{t,j}} \tag{2.7}$$
  assim w_t é uma distribuição de probabilidade.
  2. Para cada atributo, j, treine um classificador h_j que é restrito a usar um único atributo. O erro é avaliado em relação a w_t,
  $$\epsilon_j = \sum_i w_i \left| h_j\left(x_i\right) - y_i \right| \tag{2.8}$$
  .
  3. Escolha o classificador, h_t, com o menor erro ϵ_t.
  4. Atualize os pesos:
  $$w_{t+1,i} = w_{t,i} \beta_t^{1-e_i} \tag{2.9}$$
  onde $e_i = 0$ se o exemplo x_i foi classificado corretamente, $e_i = 1$ caso contrário e $\beta_t = \frac{\epsilon_t}{1-\epsilon_t}$.
- O classificador forte final é:
$$h(x) = \begin{cases} 1 & \sum_{t=1}^{T} \alpha_t h_t(x) \geq \frac{1}{2} \sum_{t=1}^{T} \alpha_t \\ 0 & \text{caso contrário} \end{cases} \tag{2.10}$$
onde $\alpha_t = \log \frac{1}{\beta_t}$.

Classificadores em Cascata

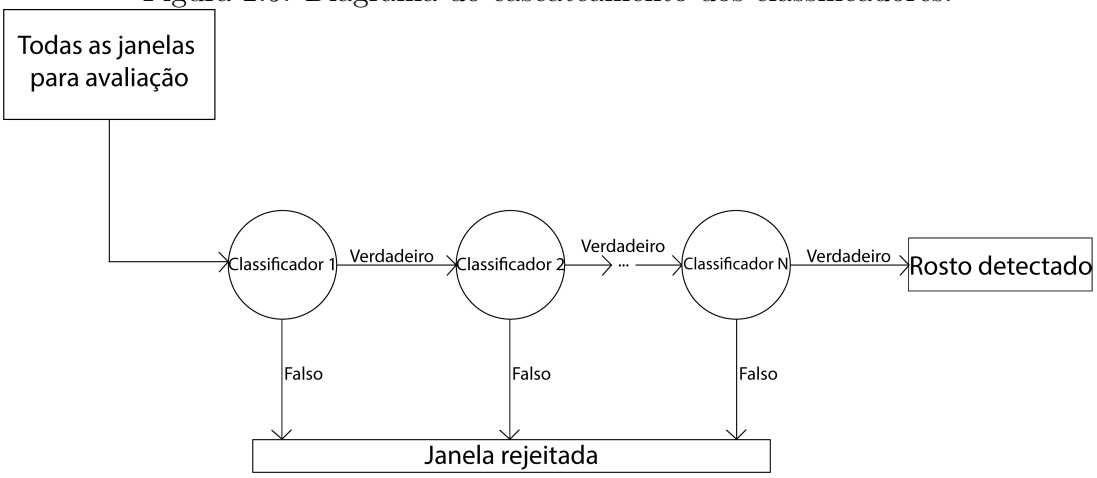
Para conseguir melhorar as taxas de detecção de rostos e diminuir as taxas de falsos positivos, os autores desenvolveram um sistema de cascateamento dos classificadores, de modo que classificadores mais simples são usados para rejeitar a maioria das imagens negativas antes que classificadores mais complexos sejam utilizados para atingir baixas taxas de falsos positivos.

A forma geral do processo de detecção é a de uma árvore de decisão degenerada, como pode ser visto na Figura (2.5). Um resultado positivo de um primeiro classificador desencadeia a avaliação por um segundo classificador, que também foi ajustado para atingir altas taxas de detecção. Um resultado positivo do segundo classificador desencadeia a avaliação por um próximo classificador e assim por diante. Um resultado negativo em qualquer ponto da cadeia leva à rejeição imediata.

Os classificadores foram construídos, utilizando AdaBoost, de maneira a minimizar os falsos negativos.

O cascateamento dos classificadores permite um menor custo computacional uma vez que a maioria das janelas avaliadas serão de resultados negativos, o que faz com que elas sejam eliminadas no começo do processamento evitando ao máximo que uma janela seja avaliada por todos os classificadores.

Figura 2.5: Diagrama do cascateamento dos classificadores.



2.4 Código de Morton

Esta seção traz uma descrição do que é o código de Morton utilizado no desenvolvimento deste projeto, também um exemplo de como ele pode ser calculado. A maneira que ele foi utilizado no projeto será melhor detalhada em uma próxima seção.

Na ciência da computação, funções que são código Morton mapeiam dados multidimensionais para uma dimensão e conseguem preservar a localidade especial dos dados, isto é, após o mapeamento unidimensional dos dados é possível retorna-los para o mapeamento multidimensional. Esse tipo de função foi, primeiramente, aplicada por Guy Macdonald Morton em 1966 para sequenciamento de arquivos. O código Morton de um ponto é calculado intercalando o valor binário de suas coordenadas.

Tabela 2.1: Coordenadas e seus respectivos códigos Morton.

x: 0 1 2 3 y: 00 01 10 11				
0 00	0000	0001	0100	0101
1 01	0010	0011	0110	0111
2 10	1000	1001	1100	1101
3 11	1010	1011	1110	1111

Uma vez que os dados são ordenados utilizando o código de Morton, qualquer método de busca em estruturas unidimensionais pode ser utilizado.

A Tabela (2.1) mostra os códigos Morton para coordenadas bidimensionais com $0 \leq x \leq 3$ e $0 \leq y \leq 3$, os valores são apresentados tanto em decimal quanto em binário na tabela. Intercalando as coordenadas binárias o código Morton é obtido, como mostrado.

O exemplo dado na Tabela (2.1) é para coordenadas bidimensionais, mas o mesmo princípio é facilmente expandido para um espaço tridimensional como o utilizado nas nuvens de pontos, sendo suficiente intercalar o valor binário da terceira coordenada para realizar o mapeamento de cada *voxel* por seu código Morton.

```
1  vx = *(VZ++);
2  vy = *(VY++);
3  vz = *(VX++);
4
5  val[i] =
6  ((0x000001 & vx))+((0x000001 & vy)<< 1)+((0x000001 & vz)<< 2)+
7  ((0x000002 & vx)<< 2)+((0x000002 & vy)<< 3)+((0x000002 & vz)<< 4)
      +
8  ((0x000004 & vx)<< 4)+((0x000004 & vy)<< 5)+((0x000004 & vz)<< 6)
      +
9  ((0x000008 & vx)<< 6)+((0x000008 & vy)<< 7)+((0x000008 & vz)<< 8)
      +
10 ((0x000010 & vx)<< 8)+((0x000010 & vy)<< 9)+((0x000010 & vz)<<10)
      +
11 ((0x000020 & vx)<<10)+((0x000020 & vy)<<11)+((0x000020 & vz)<<12)
      +
12 ((0x000040 & vx)<<12)+((0x000040 & vy)<<13)+((0x000040 & vz)<<14)
      +
```

```
13 ((0x000080 & vx)<<14)+((0x000080 & vy)<<15)+((0x000080 & vz)<<16)
     +
14 ((0x000100 & vx)<<16)+((0x000100 & vy)<<17)+((0x000100 & vz)<<18)
     +
15 ((0x000200 & vx)<<18)+((0x000200 & vy)<<19)+((0x000200 & vz)<<20)
     +
16 ((0x000400 & vx)<<20)+((0x000400 & vy)<<21)+((0x000400 & vz)<<22)
     +
17 ((0x000800 & vx)<<22)+((0x000800 & vy)<<23)+((0x000800 & vz)<<24)
     +
18 ((0x001000 & vx)<<24)+((0x001000 & vy)<<25)+((0x001000 & vz)<<26)
     +
19 ((0x002000 & vx)<<26)+((0x002000 & vy)<<27)+((0x002000 & vz)<<28)
     +
20 ((0x004000 & vx)<<28)+((0x004000 & vy)<<29)+((0x004000 & vz)<<30)
     +
21 ((0x008000 & vx)<<30)+((0x008000 & vy)<<31)+((0x008000 & vz)<<32)
     +
22 ((0x010000 & vx)<<32)+((0x010000 & vy)<<33)+((0x010000 & vz)<<34)
     +
23 ((0x020000 & vx)<<34)+((0x020000 & vy)<<35)+((0x020000 & vz)<<36)
     +
24 ((0x040000 & vx)<<36)+((0x040000 & vy)<<37)+((0x040000 & vz)<<38)
     +
25 ((0x080000 & vx)<<38)+((0x080000 & vy)<<39)+((0x080000 & vz)<<40)
     +
26 ((0x100000 & vx)<<40)+((0x100000 & vy)<<41)+((0x100000 & vz)<<42)
```

Listing 2.1: Algoritmo utilizado para a obtenção dos códigos Morton de cada *voxel*

2.5 Trabalhos Relacionados

Os trabalhos desenvolvidos por P. Nair *et al.* [9] e A. Colombo *et al.* [10] tratam também da detecção de rostos em objetos tridimensionais.

O método desenvolvido em [9] é baseado em um modelo de distribuição de pontos 3D (PDM), que é ajustado com informações sobre a estrutura e posição, e em pontos candidatos. A detecção de faces é realizada classificando as transformações entre os pontos do modelo e os vértices candidatos com base no limite superior do desvio dos parâmetros do modelo. No trabalho é relatada uma taxa de $99,6\%$ de detecção facial.

O método descrito em [10] combina uma abordagem baseada em características com uma abordagem holística para a detecção de rostos tridimensionais. Características como os olhos e o nariz são detectadas através de uma análise da curvatura da superfície. Cada trio candidato de par de olhos e nariz é processado por um classificador treinado para distinguir entre rostos e não rostos. Os resultados apresentados pelo autor indicam uma taxa de acerto de $96,85\%$.

O método proposto usa uma abordagem diferente ao não realizar nenhum processamento diretamente nos objetos tridimensionais. São utilizadas projeções bidimensionais para poder realizar o processamento e a detecção de rostos e depois os resultados obtidos são extrapolados para o objeto tridimensional.

Capítulo 3

METODOLOGIA E DESENVOLVIMENTO

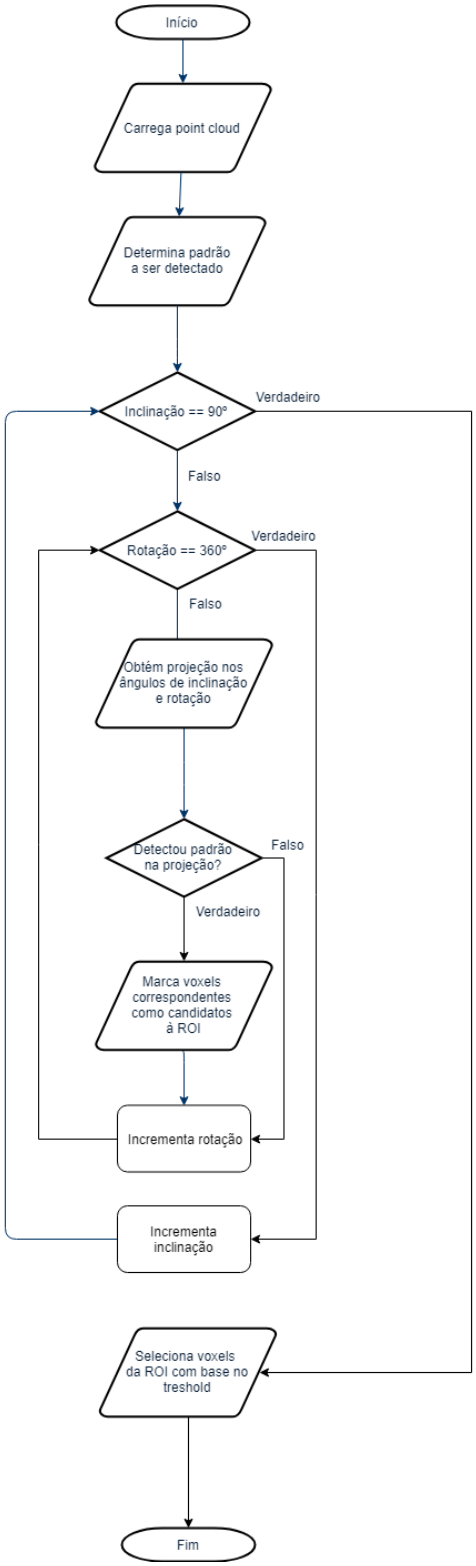
Este capítulo apresenta como foi desenvolvido o algoritmo implementado neste projeto assim como uma descrição do seu funcionamento e de todas as funções que precisaram ser implementadas para o desenvolvimento do algoritmo.

É possível dividir o projeto em dois principais métodos de atuação: a detecção da região de interesse para apenas uma nuvem de pontos e a detecção da região de interesse para uma sequência de nuvens de pontos.

- **Detecção da região de interesse para apenas uma nuvem de pontos:** Este método de atuação busca a região de interesse em apenas uma única nuvem de pontos. A entrada desse método é uma nuvem de pontos *voxelizada* da qual não se tem nenhuma informação sobre a localidade da região de interesse ou sua existência. Sua saída é uma nuvem de pontos contendo apenas os *voxels* da região de interesse obtida.
- **Detecção da região de interesse para uma sequência de nuvens de pontos:** Este método de atuação busca a região de interesse em uma sequência de nuvens de pontos, fazendo uso de consistência temporal para otimizar a busca e diminuir o custo computacional. A entrada desse método é uma sequência de nuvens de pontos *voxelizadas* na qual a primeira nuvem de pontos não se tem nenhuma informação sobre a localidade da região de interesse ou sua existência. Sua saída é uma sequência de nuvens de pontos contendo apenas os *voxels* das regiões de interesse obtidas.

O diagrama da Figura (3.1) apresenta uma explicação geral gráfica do método para a detecção e extração da região de interesse em uma nuvem de pontos.

Figura 3.1: Diagrama do algoritmo para detecção da região de interesse em uma nuvem de pontos.



3.1 Projeção de uma nuvem de pontos

Para poder realizar a projeção da nuvem de pontos, é necessário saber quais *voxels* serão projetados na imagem e quais serão descartados. Isso é feito analisando quais *voxels* possuem um menor valor de Z para determinada coordenada (X, Y) do ponto de vista que se deseja fazer a projeção. Neste projeto, foram utilizadas apenas projeções para o plano $X - Y$.

É calculado o *voxel* que está mais próximo, ou seja, que está na frente dos outros *voxels* na mesma coordenada (X, Y) de maneira que ele oclua os outros impedindo sua visualização. Após esse cálculo, os atributos desse *voxel* são projetados na imagem na coordenada (X, Y). A projeção é criada utilizando a equação abaixo

$$C(n_x, n_y) = \arg\min_{C_i} \{\mathrm{z_i} | \mathrm{x_i} = \mathrm{n_x}, \mathrm{y_i} = \mathrm{n_y}\}, \tag{3.1}$$

onde C é a projeção obtida, x_i, y_i, z_i representam as coordenadas (X, Y, Z) do i-ésimo *voxel* e C_i representa o atributo de cor do i-ésimo *voxel*.

```
for i=1:Nvox
    nx = Vr(i,2);
    ny = Vr(i,1);
    dep = Vr(i,3);
    if (dep < D(nx,ny))
        R(nx,ny) = Crgb(i,1);
        G(nx,ny) = Crgb(i,2);
        B(nx,ny) = Crgb(i,3);
        D(nx,ny) = dep;
    end
end
```

Listing 3.1: Algoritmo utilizado para realizar a projeção de uma nuvem de pontos

No algoritmo descrito em (3.1), $Nvox$ representa o número de *voxels* da nuvem de pontos, Vr são as coordenadas (X, Y, Z) de cada *voxel* e $Crgb$ representa os atributos referentes a cada *voxel*.

3.2 Rotação de uma nuvem de pontos

Para que seja possível ter projeções de todas as possíveis vistas da nuvem de pontos, é necessário que ela seja rotacionada de acordo com o ponto de vista e que, portanto, os valores das coordenadas (X, Y, Z) sejam alterados sem que a geometria seja alterada e

que os atributos de cada *voxel* se mantenham constantes também. O cálculo da rotação é feito como o exposto a seguir

$$\mathrm{c}_1 = \cos\frac{\theta_{XY}\pi}{180^\circ},\ \mathrm{s}_1 = \sin\frac{\theta_{XY}\pi}{180^\circ}\ , \tag{3.2}$$

$$\mathrm{c}_2 = \cos\frac{\theta_{XZ}\pi}{180^\circ},\ \mathrm{s}_2 = \sin\frac{\theta_{XZ}\pi}{180^\circ}\ , \tag{3.3}$$

$$\mathrm{c}_3 = \cos\frac{\theta_{YZ}\pi}{180^\circ},\ \mathrm{s}_3 = \sin\frac{\theta_{YZ}\pi}{180^\circ}\ , \tag{3.4}$$

$$M_{rot} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & c_3 & s_3 \\ 0 & -s_3 & c_3 \end{bmatrix} * \begin{bmatrix} c_2 & 0 & s_2 \\ 0 & 1 & 0 \\ -s_2 & 0 & c_2 \end{bmatrix} * \begin{bmatrix} c_1 & s_1 & 0 \\ -s_1 & c_1 & 0 \\ 0 & 0 & 1 \end{bmatrix}\ , \tag{3.5}$$

$$V_r = V * M_{rot}\ , \tag{3.6}$$

onde θ_{XY} é o ângulo de rotação nos eixos X e Y, θ_{XZ} é o ângulo de rotação nos eixos X e Z, θ_{YZ} é o ângulo de rotação nos eixos Y e Z, M_{rot} é a matriz de rotação, V é o conjunto de coordenadas da nuvem de pontos de entrada e V_r é o conjunto de coordenadas da nuvem de pontos de saída.

Para isso é utilizada uma matriz de rotação obtida a partir dos ângulos de rotação indicados. Os ângulos de rotação são dados em graus e correspondem à rotação da nuvem de pontos nos três eixos, assim Oxy é o ângulo de rotação nos eixos X e Y, Oxz nos eixos X e Z e Oyz nos eixos Y e Z.

```
th = [Oxy Oxz Oyz]
c1 = cos(th(1)*2*pi/360);
s1 = sin(th(1)*2*pi/360);
c2 = cos(th(2)*2*pi/360);
s2 = sin(th(2)*2*pi/360);
c3 = cos(th(3)*2*pi/360);
s3 = sin(th(3)*2*pi/360);
rotM = [1 0 0;0 c3 s3;0 -s3 c3] *
       [c2 0 s2;0 1 0; -s2 0 c2] *
       [c1 s1 0; -s1 c1 0; 0 0 1];
CenterVal = round(Width*0.5);
Vr = ((V-CenterVal) * rotM) + CenterVal;
```

Listing 3.2: Algoritmo utilizado para realizar a rotação de uma nuvem de pontos

No algoritmo descrito em (3.2), $rotM$ é a matriz de rotação, Vr são as coordenadas (X, Y, Z) de cada *voxel* e $Width$ é o tamanho da nuvem de pontos.

3.3 Consistência Temporal

Esta seção apresenta uma breve explicação sobre o que é a consistência temporal e como ela foi explorada para a detecção ao longo de uma sequência de quadros.

Em vídeos bidimensionais, a consistência temporal é definida como a semelhança existente entre um quadro e o quadro seguinte. Tal consistência é fortemente explorada de diferentes maneiras como para compressão mais eficiente de vídeos [19], rastreamento de objetos [20], estimação de disparidade [21], etc.

A consistência temporal parte do pressuposto que objetos não simplesmente somem de cena ou realizam um movimento com uma grande amplitude entre um quadro e outro quando as sequências são capturadas a 30 quadros por segundo.

Partindo desse mesmo pressuposto, é possível estimar a posição do rosto na nuvem de pontos em um quadro a partir da sua posição no quadro anterior. Como obtemos o rosto a partir de uma determinada vista da nuvem de pontos, a estimação da posição também foi implementada baseando-se nas vistas. Para isso, foi calculado o centroide da posição que daria uma melhor visão do rosto, o método de cálculo do centroide será melhor explicado a seguir.

3.3.1 Cálculo dos centroides

Em matemática e física, o centroide é a média aritmética da posição de todos os pontos de um objeto, assim o centroide de um conjunto de pontos seria a média de cada uma das coordenadas.

Para descobrir qual dos ângulos de visualização daria uma melhor visão da região de interesse na nuvem de pontos, foi feito o cálculo do centroide a partir do número de vezes que uma região de interesse foi encontrada em um determinado ângulo, isso foi feito para os dois ângulos de rotação utilizados. Dessa forma, foi possível obter a posição que a nuvem de pontos deveria estar para que a região de interesse pudesse ser melhor visualizada. O cálculo dos centroides dos ângulos foi feito como descrito nas equações Equações (3.7) a (3.8)

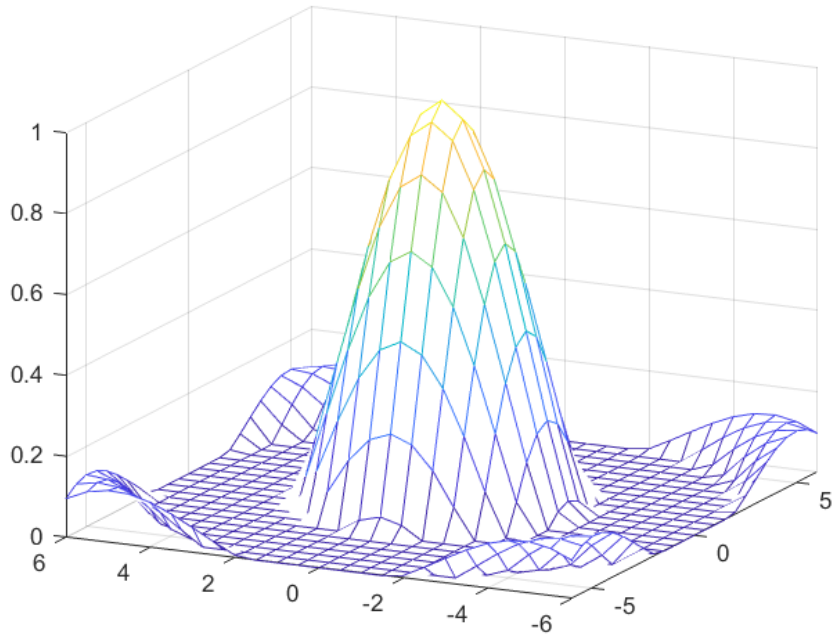
$$\Omega_r = \angle \sum_{\omega=0^\circ}^{360^\circ} n_\omega * e^{\frac{j\omega\pi}{180}}, \tag{3.7}$$

$$\Omega_i = \angle \sum_{\omega=-70^\circ}^{90^\circ} n_\omega * e^{\frac{j\omega\pi}{180}}, \tag{3.8}$$

onde $\angle$ retorna o ângulo de um número complexo, Ω_r representa o centroide do ângulo de rotação, Ω_i representa o centroide do ângulo de inclinação e n_ω representa o número de vezes que foi encontrada uma região de interesse no ângulo ω.

A Figura (3.2) mostra uma representação visual do mapa de densidade da localização das regiões de interesse. O ponto mais elevado da curva é considerado o centroide da região.

Figura 3.2: Mapa de densidade da localidade das regiões de interesse em relação às diferentes vistas de uma nuvem de pontos.



3.4 Detecção da região de interesse

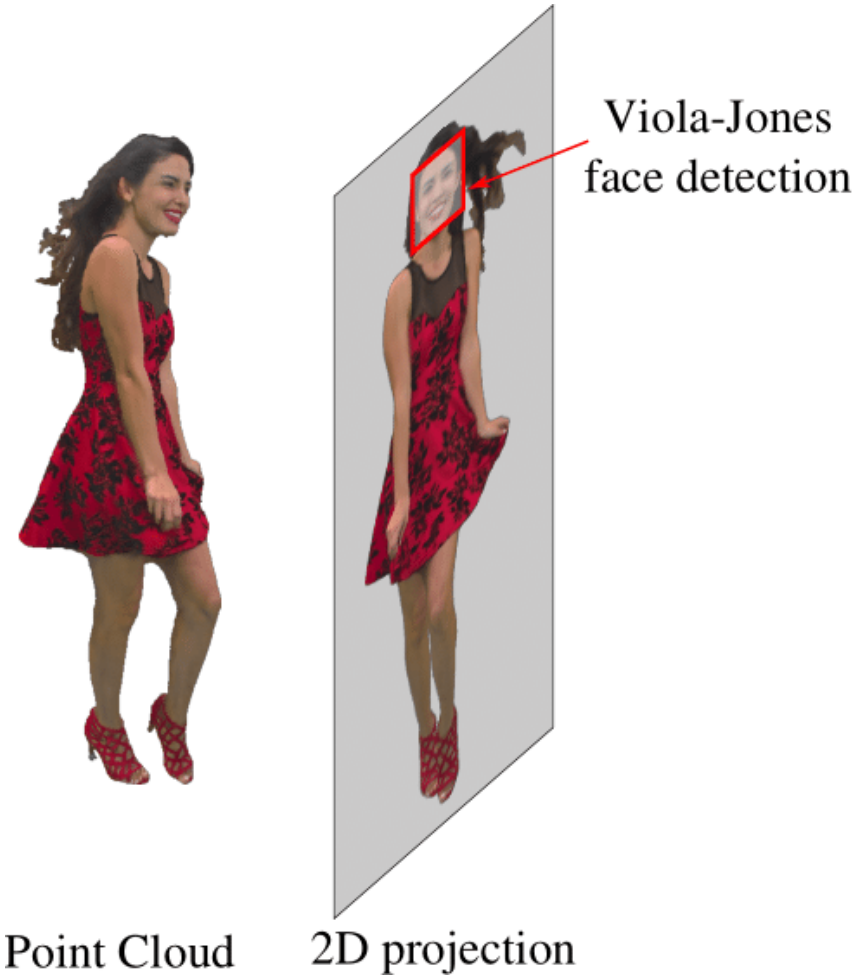
A detecção da região de interesse é muito semelhante nos métodos para apenas um quadro e para uma sequência de quadros, ambos usam do algoritmo de Viola-Jones [2] associado a projeções 2D de múltiplas vistas para encontrar a região de interesse, assim como a expansão pelo código de Morton para diminuir o número de buracos na face.

A seguir cada um dos métodos será apresentado com um maior detalhamento.

3.4.1 Detecção para apenas uma nuvem de pontos

A entrada desse método é uma nuvem de pontos *voxelizada* da qual não se tem nenhuma informação sobre a localidade da região de interesse ou sua existência. A principal ideia do algoritmo é conseguir determinar e extrair a região de interesse (ROI) da nuvem de pontos de entrada utilizando apenas projeções 2D, veja o exemplo demonstrativo na Figura (3.3).

Figura 3.3: Projeção 2D da nuvem de pontos e respectiva detecção de rosto por Viola-Jones.

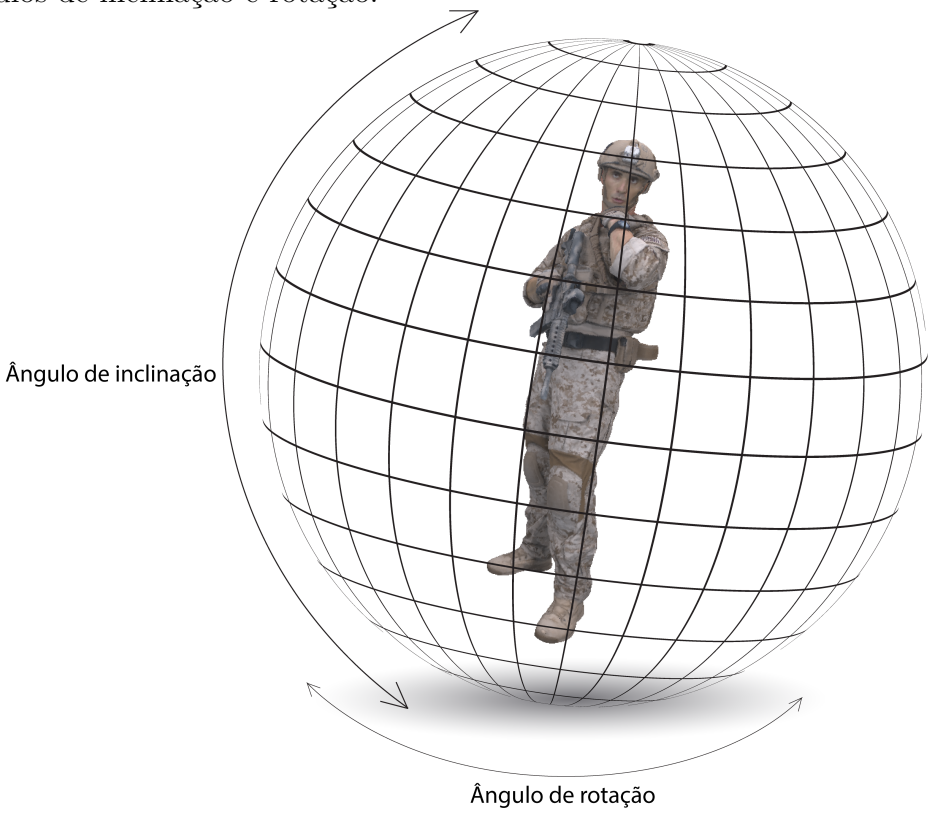


Neste projeto, a região de interesse é escolhida como sendo o rosto do sujeito na nuvem de pontos. Para entender o algoritmo deve-se considerar que as projeções são feitas a partir de vistas externas da nuvem de pontos e que o ângulo de visão da projeção cobre uma área limitada da casca esférica na qual a nuvem de pontos está contida. Uma vez que isso é estabelecido, o algoritmo divide a movimentação da visão da projeção em inclinação e rotação, a inclinação corresponde a uma movimentação latitudinal na casca esférica e a rotação a uma movimentação longitudinal, como mostrado na Figura (3.4).

Um exemplo de projeções com rotações por diferentes ângulos pode ser visto na Figura (3.5). O passo de variação foi de 5° no ângulo de rotação enquanto o ângulo de elevação foi mantido constante.

O rosto é identificado como ilustrado na Figura (3.3). A nuvem de pontos é girada para um determinado ângulo de visão definido por um par de ângulos de rotação e inclinação e, em seguida, projetada para uma imagem 2D, utilizando o método descrito em

Figura 3.4: Representação de nuvem de pontos contida na casca esférica e respectivos ângulos de inclinação e rotação.



(3.1). Usando o algoritmo de Viola-Jones [2] o rosto é detectado, como mostrado na Figura (3.6), e os *voxels* correspondentes são marcados como *face*. Esse processo é repetido para diferentes ângulos de visão. Foi escolhido variar o ângulo de rotação de 0° até 360° em passos de 10°, e variar a inclinação de $-70°$ até 90° em passos de 10°, dessa forma são obtidas 629 projeções. O passo de 10° foi escolhido por apresentar uma variação não muito grande na projeção, como pode ser visto na Figura (3.5), e para diminuir o número de projeções necessárias. Os *voxels* marcados como *face* em pelo menos 20% dos ângulos de visão são marcados como pertencentes à ROI.

Esse processo gera alguns buracos na região de interesse, uma vez que alguns *voxels* do rosto são ocluídos dependendo do ângulo de vista. Isso é mais visível nas bochechas, já que a maioria das projeções que o Viola-Jones é capaz de detectar um rosto são projeções frontais ou semi-frontais. A Figura (3.7) (a) mostra um exemplo destes buracos. Para superar esse problema, nós expandimos a ROI para seus *voxels* vizinhos.

A nuvem de pontos é regularmente dividida em cubos de largura fixa, referidos como blocos. Se ao menos um *voxel* dentro de cada cubo for marcado como ROI, todos os outros *voxels* dentro do mesmo cubo também são marcados como ROI. Na Figura (3.7), é mostrado o resultado de expandir a ROI usando cubos de diferentes larguras. Larguras

Figura 3.5: Aproximação na região do rosto de projeções de nuvens de pontos com passo variação de 5° no ângulo de rotação enquanto o ângulo de elevação foi mantido constante.

maiores resultam em menos buracos e foi decidido usar cubos de largura 8 até o fim deste projeto. Esse método de expansão da região de interesse foi escolhido por sua simplicidade, uma vez que pode ser facilmente implementado utilizando o código Morton associado a cada *voxel*, veja a Figura (3.8). O código de Morton permite determinar quais *voxels* pertencem a um mesmo bloco utilizando apenas o prefixo do código, dependendo do tamanho do bloco desejado o número de bits no prefixo pode aumentar para blocos menores ou diminuir para blocos maiores.

3.4.2 Detecção para uma sequência de quadros de uma nuvem de pontos

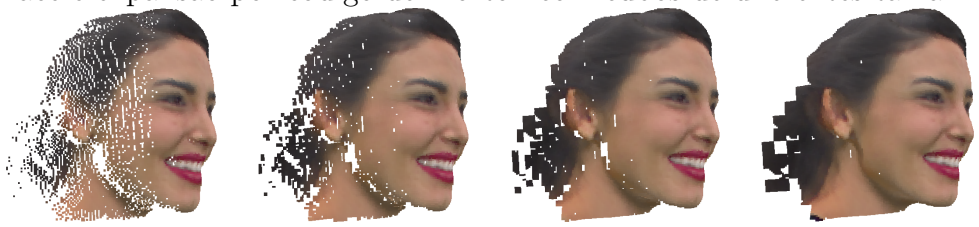
Na seção (3.4.1) a região de interesse é procurada quadro a quadro, sem que o resultado de um quadro interfira na busca do quadro seguinte. Para melhorar a consistência temporal

Figura 3.6: Projeção 2D com rosto detectado.

das regiões de interesse, foi observado que elas não se movem de maneira abrupta de um quadro para o outro. Isso acontece porque os sujeitos não simplesmente desaparecem de um quadro para o outro, em geral. Assim, pode-se considerar que a movimentação é suave no tempo.

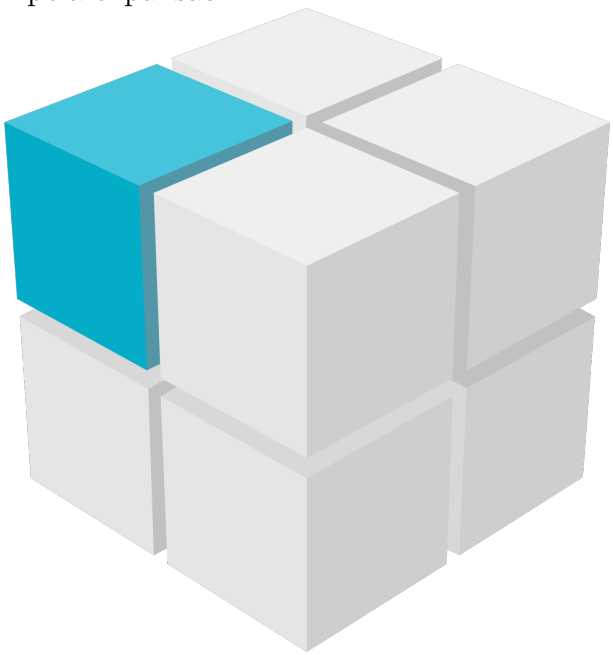
Levando em consideração a suavidade do movimento no tempo, foi possível extrapolar o método de detecção da região de interesse em um quadro para uma sequência de quadros

Figura 3.7: Artefatos na identificação da ROI devido a obliquidade das projeções dos lados da face e expansão por código de Morton com cubos de diferentes tamanhos.



(a) *largura = 1* (b) *largura = 2* (c) *largura = 4* (d) *largura = 8*

Figura 3.8: Cubo de largura 2 utilizado para a expansão por código de Morton. No caso, apenas o *voxel* em azul foi marcado como ROI, mas todos os *voxels* do cubo são selecionados como ROI pela expansão.



de uma nuvem de pontos.

A entrada desse método é uma sequência de nuvens de pontos *voxelizadas* na qual a primeira nuvem de pontos não se tem nenhuma informação sobre a localidade da região de interesse ou sua existência. Os mesmos princípios de projeções 2D e busca da região de interesse utilizando o algoritmo de Viola-Jones são utilizados neste método.

No primeiro quadro, onde não se tem nenhuma informação sobre a localidade da região de interesse ou sua existência, o método de detecção não difere em nada do método utilizado para apenas um quadro, no entanto, a partir do segundo quadro novas informações são obtidas e os parâmetros para a busca são refinados. Em adição de detectar a região de interesse e extraí-la, também é criado um mapa de densidade indicando quais vistas

houve um rosto encontrado, dessa forma a partir da segunda iteração do algoritmo não é mais necessário varrer toda a casca esférica novamente, apenas é necessário analisar uma proximidade do centroide que é calculado a partir das vistas em que foram detectadas regiões de interesse. A partir de cada novo quadro o centroide é recalculado e passado para o quadro seguinte.

Para garantir que o centroide está de fato seguindo a região de interesse, é aferida a taxa de quantas projeções próximas ao centroide foram encontradas uma face, se essa taxa for menor do que 80% toda a casca esférica é novamente varrida utilizando o método descrito anteriormente.

A região de proximidade do centroide é dada por uma variação de 20° a mais e a menos dos ângulos de elevação e de rotação e então são feitas projeções com uma variação de 4° em cada ângulo entre uma e outra. Dessa forma, dado que o centroide é correto, são feitas apenas 121 projeções por quadro ao invés das originais 629. Como o processo ainda gera alguns buracos, a expansão por código de Morton mantém-se necessária.

Capítulo 4

RESULTADOS

Este capítulo traz os resultados obtidos e a análise destes resultados. A análise dos resultados foi dividida em dois tipos: análise quantitativa e análise qualitativa. Os testes foram realizados em nuvens de pontos *voxelizadas* disponibilizadas pela 8i [22].

Para os testes foram utilizadas quatro sequências de nuvens de pontos diferentes, cada uma das sequências é composta por 300 quadros contendo uma pessoa de corpo inteiro realizando algum movimento. Cada quadro é composto por uma nuvem de pontos *voxelizada* com dimensões $1024 \times 1024 \times 1024$ com atributos de cor no espaço Red Green Blue (RGB).

4.1 Modelos de Análise

Foram adotados dois modelos de análise para os resultados obtidos: o modelo de análise qualitativo e o modelo de análise quantitativo.

- **Modelo de análise qualitativo:** Este modelo de análise consiste em uma avaliação subjetiva se o rosto foi ou não detectado com base na análise visual do resultado.
- **Modelo de análise quantitativo:** Este modelo de análise consiste em uma avaliação estatística dos *voxels* classificados como rosto pelo algoritmo em comparação com um *ground truth*.

Os *ground truths* foram gerados manualmente utilizando o software *MeshLab* [23]. Para escolher o que deveria ser incluso no *ground truth*, foi utilizada a definição de rosto na língua portuguesa [24].

4.1.1 Modelo de Análise Qualitativo

Este modelo consiste na avaliação subjetiva dos resultados obtidos. Esta avaliação foi feita baseada na análise visual dos resultados e no julgamento se o rosto foi ou não identificado corretamente na nuvem de pontos.

Foi decidido usar um modelo qualitativo para que a qualidade visual do rosto extraído pudesse ser avaliada, isto é, para que se pudesse analisar se o rosto não possui muitos buracos que causam incômodo visual ou se o rosto foi detectado com algumas falhas mas que não causam grande incômodo visual.

Dessa forma essa avaliação traz três categorias diferentes de resultados:

- **Rosto não encontrado:** Caso em que o algoritmo é incapaz de extrair o rosto da nuvem de pontos.
- **Rosto encontrado com sucesso:** Caso em que o algoritmo é capaz de extrair o rosto da nuvem de pontos sem a existência de grandes buracos no rosto que causem incômodo visual.
- **Rosto encontrado com ressalvas:** Caso em que o algoritmo é capaz de extrair o rosto da nuvem de pontos porém são notados grandes buracos no rosto que causam incômodo visual.

4.1.2 Modelo de Análise Quantitativo

Este modelo de análise consiste em uma avaliação estatística dos resultados. Os *voxels* classificados como rosto pelo algoritmo são comparados com um *ground truth* e então é possível montar uma matriz de confusão [25].

A matriz de confusão utilizada possui algumas adaptações para melhor se adequar a realidade deste projeto. Na matriz utilizada o campo de falso positivo foi separado em dois campos: falso positivo na região da cabeça e falso positivo na região do corpo. Essa separação foi feita pois é plausível considerar que um falso positivo na região do corpo indica um erro mais grave do que um falso positivo na região da cabeça, que está bem próxima do rosto que é a região de interesse procurada.

Dessa forma a matriz de confusão utilizada contém quatro campos, eles são: verdadeiro positivo, falso negativo, falso positivo na região da cabeça e falso positivo na região do corpo.

- **Verdadeiro positivo:** *Voxel* existente tanto no resultado obtido quanto no *ground truth*.

- **Falso negativo:** *Voxel* existente no *ground truth* porém inexistente no resultado obtido.
- **Falso positivo na região da cabeça:** *Voxel* existente no resultado obtido porém inexistente no *ground truth* que se encontra acima da linha da base do pescoço.
- **Falso positivo na região do corpo:** *Voxel* existente no resultado obtido porém inexistente no *ground truth* que se encontra abaixo da linha da base do pescoço.

4.2 Resultados para Detecção da Região de Interesse em um Quadro

Esta seção irá apresentar os resultados obtidos para a detecção da Região de Interesse (ROI), no caso rostos frontais, em apenas um quadro da nuvem de pontos. As Figuras (4.1) a (4.3) mostram as nuvens de pontos utilizadas nos testes. Para as nuvens de pontos em que mais de um quadro foi utilizado, apenas um quadro foi mostrado.

Figura 4.1: Nuvens de pontos utilizadas para os testes.

(a) thaidancer_vox10 (b) boxer_vox10

4.2.1 Resultados Quantitativos

Para o método de avaliação quantitativo foram utilizadas quatorze nuvens de pontos distintas. Os resultados obtidos são apresentados na Tabela (4.1).

Figura 4.2: Nuvens de pontos utilizadas para os testes.

(a) longdress_vox10_1231 (b) redandblack_vox10_1450

Figura 4.3: Nuvens de pontos utilizadas para os testes.

(a) soldier_vox10_0536 (b) loot_vox10_1000

4.2.2 Resultados Qualitativos

Para o método de avaliação qualitativo foram utilizadas as mesmas quatorze nuvens de pontos distintas. Os resultados obtidos são apresentados na Tabela (4.2).

4.2.3 Discussão dos Resultados

As Tabelas (4.1) a (4.2) apresentam os resultados quantitativo e qualitativo, respectivamente, para os testes realizados neste projeto. É possível notar que os resultados dos dois métodos de avaliação possuem uma forte correlação, uma vez que uma alta taxa de verdadeiros positivos implicou que o rosto foi encontrado com sucesso em todos os casos.

Tabela 4.1: Tabela de confusão dos resultados para detecção do rosto em um quadro. Os valores representam o número de *voxels*.

Arquivo	Verdadeiro postivo	Falso negativo	Falso positivo na região da cabeça	Falso positivo na região do corpo
longdress_vox10_1051	15918 (94,97%)	843	5798	0
longdress_vox10_1231	10531 (98,37%)	175	7090	0
longdress_vox10_1300	14141 (87,84%)	1958	14856	0
longdress_vox10_1350	11981 (90,97%)	1189	3661	0
loot_vox10_1000	12180 (98,50%)	185	8914	0
loot_vox10_1200	11908 (93,26%)	860	6863	0
redandblack_vox10_1450	11712 (96,87%)	379	15663	0
redandblack_vox10_1496	11119 (87,74%)	1554	15098	0
redandblack_vox10_1550	12226 (87,17%)	1799	13621	0
soldier_vox10_0536	8645 (57,94%)	6276	5000	0
soldier_vox10_0651	4524 (36,85%)	7752	17518	0
soldier_vox10_0690	0 (0,00%)	11298	0	12611
thaidancer_vox10	9819 (98,25%)	175	11229	0
boxer_vox10	11363 (91,62%)	1039	10607	8195
Total	**146067 (80,46%)**	**35482**	**135918**	**20806**

Tabela 4.2: Resultados do método de avaliação qualitativo para detecção do rosto em um quadro.

Arquivo	Rosto encontrado com sucesso	Rosto encontrado com ressalvas	Rosto não encontrado
longdress_vox10_1051	1	0	0
longdress_vox10_1231	1	0	0
longdress_vox10_1300	1	0	0
longdress_vox10_1350	1	0	0
loot_vox10_1000	1	0	0
loot_vox10_1200	1	0	0
redandblack_vox10_1450	1	0	0
redandblack_vox10_1496	1	0	0
redandblack_vox10_1550	1	0	0
soldier_vox10_0536	0	1	0
soldier_vox10_0651	0	1	0
soldier_vox10_0690	0	0	1
thaidancer_vox10	1	0	0
boxer_vox10	1	0	0
Total	**11**	**2**	**1**

Figura 4.4: *Ground truth* comparado ao resultado obtido na nuvem de pontos longdress_vox10_1051.

(a) *ground truth* (b) região encontrada

Figura 4.5: *Ground truth* comparado ao resultado obtido na nuvem de pontos boxer_vox10.

(a) *ground truth* (b) região encontrada

Portanto, pode-se considerar que o método quantitativo serviu para corroborar o método de avaliação qualitativo que possui apenas quesitos subjetivos de avaliação.

Em relação aos resultados qualitativos obtidos, é possível ver que o algoritmo obteve uma alta taxa de sucesso, encontrando o rosto em treze das quatorze nuvens de pontos utilizadas, sendo que apenas dois dos treze rostos encontrados foram gerados buracos no rosto que causam incômodo visual. Na Figura (4.7) é mostrado um exemplo de detecção em que foi considerado que houve a presença de buracos que causam incômodo visual. A Figura (4.6) é um exemplo de detecção considerada bem sucedida.

Em relação aos resultados quantitativos obtidos, eles mostram um índice de verdadeiros positivos muito acima da taxa de falsos negativos, com taxas de acerto de até 98,5%,

Figura 4.6: Exemplo de resultado classificado como "rosto encontrado com sucesso". Resultante da busca na nuvem de pontos redandblack_vox10_1496.

e com apenas dois testes onde o número de falsos negativos superou o número de verdadeiros positivos. Vale ressaltar que um desses testes o rosto não conseguiu ser encontrado pelo algoritmo. A taxa total de acerto de verdadeiros positivos no final foi de 80,45%.

O número de falsos positivos foi elevado. No entanto, 85,3% deles foram localizados na região da cabeça, o que é menos grave do que os que foram localizados na região do corpo. A Figura (4.4) traz um exemplo da região encontrada em comparação com o *ground truth*, é possível ver que o rosto foi obtido com sucesso. Os falsos positivos apresentados ocorrem, principalmente, no cabelo, uma vez que o *ground truth* não possui cabelo.

Já a Figura (4.5) traz um exemplo de falsos positivos localizados fora da região do rosto. Neste resultado, é de fato notável o quanto de informação extra e errada foi extraída, tendo uma grande região completamente desconexa do rosto que foi extraída pelo algoritmo como sendo pertencente ao rosto.

Na Figura (4.8) é mostrada a nuvem de pontos em que o algoritmo falhou completamente em detectar o rosto. É possível argumentar que uma das causas que pode ter levado o algoritmo à falha é o fato de grande parte do rosto estar ocluso. A oclusão ocorre tanto pelo capacete, o qual atrapalha a detecção e interferiu na detecção dos outros dois quadros

Figura 4.7: Exemplo de resultado classificado como "rosto encontrado com ressalvas". Resultante da busca na nuvem de pontos soldier_vox10_0651.

testados da nuvem de pontos *soldier*, quanto pela mira do rifle que está na frente do rosto tampando quase metade dele. Tal falha pode ser esperada pelas próprias restrições do algoritmo utilizado [2] para a detecção de rostos nas projeções 2D.

4.3 Resultados para Detecção da Região de Interesse em uma Sequência de Quadros

Esta seção irá apresentar os resultados obtidos para a detecção da Região de Interesse (ROI), no caso rostos frontais, em uma sequência de quadros da nuvem de pontos. Apenas as nuvens de pontos da Figura (4.2) foram utilizadas nos testes. Todos os trezentos quadros de cada uma das duas nuvens de pontos foram utilizados nos testes.

4.3.1 Resultados Quantitativos

Para o método de avaliação quantitativo foram utilizados sete quadros distintos das nuvens de pontos avaliadas com a adição da técnica de consistência temporal. Foram utilizados quatro quadros não consecutivos da nuvem de pontos *longdress* e três não consecutivos

Figura 4.8: Nuvem de pontos em que o rosto não foi detectado.

da nuvem de pontos *redandblack*, são os mesmos quadros referentes às nuvens de pontos em questão utilizados para a avaliação do método de detecção para apenas um quadro.

Os resultados obtidos são apresentados na Tabela (4.3), que não apresenta a coluna "Falso positivo na região do corpo"porque em nenhum dos casos testados tal erro se fez presente. Portanto, para uma melhor visualização da tabela a coluna foi removida.

4.3.2 Resultados Qualitativos

Para o método de avaliação qualitativo foram utilizados todos os trezentos quadros das duas nuvens de pontos avaliadas com a adição da técnica de consistência temporal. Os resultados obtidos são apresentados na Tabela (4.4).

Tabela 4.3: Tabela de confusão dos resultados para detecção do rosto em uma sequência de quadros. Os valores representam o número de *voxels*.

Arquivo	Verdadeiro postivo	Falso negativo	Falso positivo na região da cabeça
longdress_vox10_1051	15918 (94,97%)	843	5798
longdress_vox10_1231	10483 (97,92%)	223	14041
longdress_vox10_1300	12360 (76,77%)	3739	4132
longdress_vox10_1350	11981 (90,97%)	1189	3661
redandblack_vox10_1450	11712 (96,87%)	379	15663
redandblack_vox10_1496	9688 (76,45%)	2985	7485
redandblack_vox10_1550	12059 (85,98%)	1966	12396
Total	**84201 (88,15%)**	**11324**	**63176**

Tabela 4.4: Resultados do método de avaliação qualitativo para detecção do rosto em uma sequência de quadros. Os valores representam o número de quadros.

Sequência	Rosto encontrado com sucesso	Rosto encontrado com ressalvas	Rosto não encontrado
longdress	289 (96,33%)	11	0
redandblack	287 (95,67%)	13	0
Total	**576 (96,00%)**	**24**	**0**

A Tabela (4.4) apresenta os resultados totais obtidos de cada sequência. Isso foi feito para poder simplificar a visualização da tabela, uma vez que apresentar os resultados de cada um dos quadros seria inviável.

4.3.3 Discussão dos Resultados

As Tabelas (4.3) a (4.4) apresentam os resultados quantitativo e qualitativo, respectivamente, para os testes realizados utilizando a detecção para uma sequência de quadros. Assim como os resultados obtidos na seção (4.2), é possível ver que existe uma forte correlação entre os resultados do método de avaliação quantitativo e do método de avaliação qualitativo. Dessa forma, novamente, o método quantitativo serviu para validar os resultados do método qualitativo.

Começando pelos resultados qualitativos, é notável a alta taxa de sucesso que foi atingida pelo algoritmo proposto. Dos 600 quadros analisados, o rosto foi perfeitamente encontrado em 576 deles e encontrado com buracos que causam incômodo visual em 24 dos quadros. É importante destacar que não houve nenhum quadro em que o algoritmo não foi capaz de extrair o rosto como desejado.

Os resultados quantitativos obtidos apresentam um alto índice de verdadeiros positivos em comparação com o índice de falsos negativos, com uma taxa total de acerto de 88,14% na soma de todos os *voxels* que deveriam ter sido detectados como pertencentes a região de interesse. Dos testes medidos, a maior taxa de acerto foi de 97,91% e em nenhum o número de falsos negativos superou o de verdadeiros positivos.

O número de falsos positivos apresentados novamente foi elevando, porém 100% deles ocorreram na região da cabeça, além de ter havido uma queda do número total de falsos

Tabela 4.5: Comparação entre os resultados para a detecção em apenas um quadro e a detecção em uma sequência de quadros.

	Verdadeiro postivo	**Falso positivo na região da cabeça**
Detecção em um quadro	87628	75787
Detecção em uma sequência	84201	63176

positivos. Os falsos positivos seguem o mesmo padrão de ocorrer em regiões como cabelo e pescoço do método de detecção em apenas um quadro.

Em comparação com os resultados obtidos para apenas um quadro, a performance do algoritmo foi mantida em relação aos resultados positivos, como pode ser visto na Tabela (4.5), porém o custo computacional de utilizar os centroides para calcular a região de maior probabilidade de existir um rosto em quadros sequenciais é muito menor do que o custo computacional de procurar o rosto quadro a quadro sem que nenhuma informação intra-quadros seja utilizada.

A Figura (4.9) mostra a comparação da mesma nuvem de pontos detectada utilizando a técnica em apenas um quadro e a técnica em uma sequência de quadros. É possível notar que a detecção da ROI em uma sequência (Figura (4.9) (c)) se aproxima mais visualmente ao *ground truth* (Figura (4.9) (a)) do que a detecção em um quadro (Figura (4.9) (b)). Além do custo de processamento ser bem menor, como explicado na seção (3.4.2).

Figura 4.9: Comparação entre o método de detecção para um quadro e para uma sequência de quadros. Nuvem de pontos longdress_vox10_1300.

(a) *ground truth* (b) detecção em um quadro (c) detecção em uma sequência

A comparação das Tabela (4.3) e Tabela (4.1) mostra que houve uma diminuição do número de falsos positivos detectados, porém também houve uma queda no número de verdadeiros positivos. Para melhor comparar os resultados, foi feita a Tabela (4.5), onde foram incluídos apenas os resultados obtidos referentes aos mesmos quadros das mesmas nuvens de pontos utilizando os dois diferentes métodos.

A queda na taxa de verdadeiro positivo foi de 3,9% (3427 *voxels*), enquanto a queda na taxa de falso positivo foi de 16,6% (12611 *voxels*). Isso mostra que o método utilizado

para a sequência de nuvens de pontos foi melhor em evitar falsos positivos, mas teve uma performance um pouco pior ao detectar os verdadeiros positivos do que o método utilizado para analisar cada quadro individualmente.

É interessante ressaltar que nas duas sequências analisadas sempre que um resultado foi classificado como tendo encontrado o rosto mas com buracos que causam incômodo visual, o quadro após ele era analisado utilizando o método de análise para o primeiro quadro. Ou seja, após ter um resultado não satisfatório, o algoritmo descarta os centroides deste resultado e analisa o próximo quadro como se fosse o primeiro de uma sequência para que o erro não seja propagado. A identificação do resultado não satisfatório é feita pelo próprio algoritmo de maneira automática, ao analisar as proximidades do centroide e procurar por um rosto, caso em menos de 80% das análises se detecte um rosto, o resultado é classificado como insatisfatório.

Capítulo 5

CONCLUSÃO

O desenvolvimento deste projeto possibilitou um estudo aprofundado sobre nuvens de pontos e algoritmos para detecção de regiões de interesse em imagens 2D, mais especificamente detecção de rostos. Foi proposto então um *framework* capaz de utilizar as técnicas existentes para o processamento e análise de imagens e vídeos bidimensionais para realizar a detecção de regiões de interesse em nuvens de pontos.

Existem outros métodos propostos na literatura para a identificação de regiões de interesse ou classificação de objetos em nuvens de pontos que fazem uso de diversas técnicas. No entanto, não há nenhum que faça o uso de projeções bidimensionais para atingir os objetivos propostos. Dessa forma, o método sugerido inova nesse sentido e apresentou resultados que podem ser considerados bem sucedidos.

O *framework* desenvolvido foi capaz de realizar algum processamento em projeções 2D e extrapolar os resultados obtidos para nuvens de pontos 3D. Ele também foi capaz de juntar o processamento realizado de múltiplas vistas e aplicar de maneira proporcional em cada *voxel* os resultados obtidos.

O principal objetivo proposto, encontrar e extrair rostos de uma nuvem de pontos da qual nenhuma informação é conhecida, foi atingido com sucesso e mostrou uma alta taxa de acerto, como visto nas Tabelas (4.1) a (4.2). Tem-se como exemplo, a Figura (4.4), que mostra que os resultados obtidos ficaram extremamente próximos do *ground truth*, apresentando uma alta qualidade na determinação da região de interesse.

Por fim, foi utilizada a consistência temporal, para otimizar a detecção do rosto ao longo de uma sequência de quadros da mesma nuvem de pontos. Dessa forma, o *framework* dá suporte, tanto para a detecção de rosto em apenas um quadro, como em uma sequência de quadros de uma forma otimizada em relação ao custo computacional. Os resultados atingidos pelo módulo que explora a consistência temporal também foram bem sucedidos, como pode ser visto nas Tabelas (4.3) a (4.4). A Figura (4.9) demonstra uma aparente

melhoria no resultado com uso da consistência temporal em relação ao resultado sem tal informação.

Com os resultados obtidos, foi possível desenvolver o trabalho *"Point Cloud Compression Incorporating Region of Interest Coding"* [5] publicado no *IEEE Int'l Conf. Image Processing (ICIP)* de 2019.

Futuros estudos podem ser feitos na intenção de melhorar o *framework* proposto. Uma possibilidade de melhoria seria selecionar a maior região conexa em um resultado, ou seja, aquela que contém mais *voxels* conectados, para tentar eliminar falsos positivos na região do corpo, como os que ocorrem na Figura (4.5). Com base nas observações realizadas a partir dos resultados obtidos, infere-se que a maior região conexa tende a ser a que contém a região de interesse procurada.

Outra possibilidade de melhoria, a fim de tentar diminuir o número de falsos negativos, seria utilizar as projeções 2D para encontrar a ponta do nariz e a partir dela determinar uma esfera ou um elipsoide onde todos os *voxels* contidos nesse sólido serão parte do rosto. Essa ideia é semelhante com o que é proposto em [9].

Referências

[1] Milgram, Paul e Herman Colquhoun: *A taxonomy of real and virtual world display integration.* janeiro 2001. 1

[2] Viola, P. e M. Jones: *Rapid object detection using a boosted cascade of simple features.* Em *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, volume 1, páginas I–I, Dec 2001. 1, 4, 8, 21, 23, 35

[3] Otsu, N.: *A threshold selection method from gray-level histograms.* IEEE Transactions on Systems, Man, and Cybernetics, 9(1):62–66, Jan 1979, ISSN 0018-9472. 1

[4] Girshick, R., J. Donahue, T. Darrell e J. Malik: *Rich feature hierarchies for accurate object detection and semantic segmentation.* Em *2014 IEEE Conference on Computer Vision and Pattern Recognition*, páginas 580–587, June 2014. 1

[5] Sandri, Gustavo, Victor F. Figueiredo, Philip A. Chou e Ricardo de Queiroz: *Point cloud compression incorporating region of interest coding.* Em *IEEE Int'l Conf. Image Processing (ICIP)*, 2019. aceito para publicação. 2, 41

[6] Schwarz, Sebastian, Marius Preda, Vittorio Baroncini, Madhukar Budagavi, Pablo Cesar, Philip A. Chou, Robert A. Cohen, Maja Krivokuća, Sebastien Lasserre, Zhu Li, Joan Llach, Khaled Mammou, Rufael Mekuria, Ohji Nakagami, Ernestasia Siahaan, Ali Tabatabai, Alexandros Tourapis e Vladyslav Zakharchenko: *Emerging MPEG standards for point cloud compression.* IEEE J. Emerging Topics in Circuits and Systems. aceito para publicação. 2

[7] Hadizadeh, Hadi e Ivan V. Bajić: *Saliency-aware video compression.* 23, janeiro 2014. 2, 4, 7

[8] Brinkmann, Ron: *The Art and Science of Digital Compositing.* Morgan Kaufmann, 1999. 4, 6

[9] Nair, P. e A. Cavallaro: *3-d face detection, landmark localization, and registration using a point distribution model.* IEEE Transactions on Multimedia, 11(4):611–623, June 2009, ISSN 1520-9210. 4, 14, 41

[10] Colombo, Alessandro, Claudio Cusano e Raimondo Schettini: *3d face detection using curvature analysis.* Pattern Recognition, 39:444–455, março 2006. 4, 14, 15

[11] Fabry, T., D. Vandermeulen e P. Suetens: *3d face recognition using point cloud kernel correlation*. Em *2008 IEEE Second International Conference on Biometrics: Theory, Applications and Systems*, páginas 1–6, Sep. 2008. 4

[12] Wang, Z., L. Zhang, L. Zhang, R. Li, Y. Zheng e Z. Zhu: *A deep neural network with spatial pooling (dnnsp) for 3-d point cloud classification*. IEEE Transactions on Geoscience and Remote Sensing, 56(8):4594–4604, Aug 2018, ISSN 0196-2892. 4

[13] Habermann, D., E. Dranka, Y. Caceres e J. B. R. do Val: *Drones and helicopters classification using point clouds features from radar*. Em *2018 IEEE Radar Conference (RadarConf18)*, páginas 0246–0251, April 2018. 4

[14] Zuffo, Marcelo: *University of são paulo point cloud dataset*. http://uspaulopc.di.ubi.pt/. 5

[15] Sandri, G., R. L. de Queiroz e P. A. Chou: *Comments on "Compression of 3D Point Clouds Using a Region-Adaptive Hierarchical Transform"*. ArXiv e-prints, maio 2018. `https://arxiv.org/abs/1805.09146v1`. 5

[16] Papageorgiou, C.P., Michael Oren e Tomaso Poggio: *General framework for object detection*. Volume 6:, páginas 555 – 562, fevereiro 1998, ISBN 81-7319-221-9. 8

[17] Viola, Paul e Michael Jones: *Robust real-time object detection*. Em *International Journal of Computer Vision*, 2001. 9, 10

[18] Freund, Yoav e R.E. Schapire: *A decision-theoretic generalization of on-line learning and an application to boosting*. Journal of computer and system sciences, 55:119–, agosto 1997. 10

[19] Hariharan, H. P., T. Lange e T. Herfet: *Low complexity light field compression based on pseudo-temporal circular sequencing*. Em *2017 IEEE International Symposium on Broadband Multimedia Systems and Broadcasting (BMSB)*, páginas 1–5, June 2017. 20

[20] Lipton, A. J., H. Fujiyoshi e R. S. Patil: *Moving target classification and tracking from real-time video*. Em *Proceedings Fourth IEEE Workshop on Applications of Computer Vision. WACV'98 (Cat. No.98EX201)*, páginas 8–14, Oct 1998. 20

[21] Khoshabeh, R., S. H. Chan e T. Q. Nguyen: *Spatio-temporal consistency in video disparity estimation*. Em *2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, páginas 885–888, May 2011. 20

[22] d'Eon, E., B. Harrison, T. Myers e P. A. Chou: *8i voxelized full bodies - a voxelized point cloud dataset*. ISO/IEC JTC1/SC29 Joint WG11/WG1 (MPEG/JPEG) input document WG11M40059/WG1M74006, January 2017. 28

[23] Cignoni, Paolo, Marco Callieri, Massimiliano Corsini, Matteo Dellepiane, Fabio Ganovelli e Guido Ranzuglia: *MeshLab: an Open-Source Mesh Processing Tool*. Em Scarano, Vittorio, Rosario De Chiara e Ugo Erra (editores): *Eurographics Italian Chapter Conference*. The Eurographics Association, 2008, ISBN 978-3-905673-68-5. 28

[24] *Michaelis dicionário escolar língua portuguesa.* Editora Melhoramentos Ltda., 2008, ISBN 978-85-06-05464-2. 28

[25] Stehman, Stephen: *Selecting and interpreting measures of thematic classification accuracy.* Remote Sensing of Environment, 62:77–89, outubro 1997. 29