UNIVERSIDADE FEDERAL DO PARANÁ

MARIA LUIZA SAMPAIO LOGRADO

SPLINES PARA REPRESENTAÇÃO DE CARACTERÍSTICAS FACIAIS COM APLICAÇÃO EM BIOMETRIA

CURITIBA

2025

MARIA LUIZA SAMPAIO LOGRADO

SPLINES PARA REPRESENTAÇÃO DE CARACTERÍSTICAS FACIAIS COM APLICAÇÃO EM BIOMETRIA

Trabalho apresentado como requisito parcial à coclusão do curso de graduação em Matemática Industrial no Setor de Exatas da Universidade Federal do Paraná.

Orientador: Prof Thiago de Oliveira Quinelato, DSc.

CURITIBA

2025

RESUMO

O resumo deve ressaltar o objetivo, o método, os resultados e as conclusões do documento. A ordem e a extensão destes itens dependem do tipo de resumo (informativo ou indicativo) e do tratamento que cada item recebe no documento original. O resumo deve ser precedido da referência do documento, com exceção do resumo inserido no próprio documento. (. . .) As palavras-chave devem figurar logo abaixo do resumo, antecedidas da expressão Palavras-chave:, separadas entre si por ponto e finalizadas também por ponto.Ter no máximo 500 palavras!!! As palavras chave são separadas por ponto e vírgula.

Palavras-chaves: latex; abntex; editoração de texto.

ABSTRACT

This is the english abstract.

Key-words: latex. abntex. text editoration.

SUMÁRIO

1 INTRODUÇÃO

O reconhecimento facial é uma das áreas mais promissoras e desafiadoras dentro do campo da biometria. Sua aplicação abrange desde sistemas de segurança até autenticação em dispositivos móveis, tornando-se cada vez mais presente no cotidiano. Nos últimos anos, métodos baseados em aprendizado profundo, especialmente redes neurais convolucionais (CNNs), têm dominado esse campo devido à sua alta acurácia. No entanto, tais métodos apresentam algumas limitações, especialmente no que diz respeito à interpretabilidade dos modelos e à necessidade de grandes volumes de dados para treinamento.

Nesse contexto, a busca por alternativas mais eficientes e interpretáveis tem ganhado relevância. Uma dessas alternativas é o uso de *splines*, funções matemáticas capazes de gerar curvas suaves a partir de um conjunto reduzido de pontos. Ao representar as características faciais por meio de curvas, é possível reduzir significativamente a redundância dos dados, além de tornar o processo de extração e análise mais estruturado e compreensível.

Este trabalho propõe uma abordagem híbrida que combina a expressividade das *splines* com técnicas de aprendizado de máquina para o reconhecimento facial. A metodologia baseia-se na extração de contornos das principais características do rosto, sua modelagem com *splines* e posterior classificação utilizando algoritmos de comparação de sequências, como o *Dynamic Time Warping* (DTW) (Sakoe; Chiba, 1978). Espera-se, com isso, alcançar uma representação mais compacta e interpretável das feições faciais, sem comprometer o desempenho do sistema de identificação.

A introdução dessa nova abordagem se justifica não apenas pela busca por eficiência computacional, mas também pela necessidade de modelos que ofereçam maior transparência quanto às decisões tomadas. Dessa forma, o presente trabalho contribui para a ampliação das possibilidades de representação biométrica e propõe caminhos para o desenvolvimento de sistemas mais acessíveis, leves e explicáveis.

1.1 OBJETIVO

O objetivo deste trabalho é investigar a viabilidade do uso de *splines* como método de representação de características faciais em sistemas biométricos, avaliando sua eficácia na melhoria do desempenho da identificação de indivíduos. A abordagem proposta visa combinar a expressividade das *splines* com a capacidade do *Dynamic Time Warping* (DTW) para reconhecimento facial, buscando uma solução que seja ao mesmo tempo eficiente e interpretável.

1.2 JUSTIFICATIVA

Métodos convencionais baseados em redes neurais convolucionais (CNNs) e aprendizado profundo apresentam altos níveis de acurácia, mas muitos desses modelos operam como “caixas-pretas”, dificultando a interpretação das características extraídas e utilizadas para a tomada de decisão.

Diante desse cenário, a abordagem baseada no uso de *splines* surge como uma alternativa para representação de características faciais de forma mais estruturada e interpretável. As *splines* são funções matemáticas que permitem a modelagem de curvas suaves a partir de um conjunto reduzido de pontos de controle, garantindo uma representação eficiente das características faciais sem a necessidade de armazenar grandes quantidades de dados redundantes.

Além da compactação da informação, o uso de *splines* facilita a extração de características relevantes, permitindo que apenas as estruturas mais significativas do rosto sejam utilizadas como entrada para o modelo de aprendizado. Isso pode melhorar a eficiência do DTW aumentando a precisão na identificação.

Dessa forma, este trabalho se justifica pela necessidade de explorar métodos alternativos para reconhecimento facial que combinem eficiência computacional e interpretabilidade.

1.3 METODOLOGIA

A abordagem proposta para o reconhecimento facial baseada em *splines* segue um fluxo estruturado, dividido em cinco etapas principais: detecção de características faciais, extração de contornos, redução de pontos, modelagem com *splines* e classificação por DTW.

É importante esclarecer que a primeria versão das 4 primeiras etapas do método proposto neste trabalho foi desenvolvida durante o Programa de Voluntariado Acadêmico (PVA) em 2024 e neste TCC será desenvolvida a última etapa, que é a classificação por *Dynamic Time Warping*, além de algumas melhorias nas etapas anteriores.

1.3.1 Detecção de Características Faciais

Inicialmente, são identificadas as principais regiões do rosto, como olhos, boca e nariz. Para isso, é utilizado o modelo Haar Cascades (Viola; Jones, 2001; OPENCV. . . , s.d.[a]) como a técnica de detecção. O modelo é ajustado para reconhecer padrões de características faciais em imagens, permitindo a localização das regiões de interesse.

1.3.2 Extração de Contornos

Após a identificação das regiões faciais, são extraídos os contornos dessas características por meio da técnica de processamento de imagens de detecção de bordas utilizando o operador de Canny (segmentação baseada em gradientes) (Canny, 1986; OpenCV, s.d.[b]). O objetivo é obter um conjunto inicial de pontos que descrevem as características faciais.

1.3.3 Redução de Pontos

Para evitar redundância e reduzir a complexidade computacional, os pontos extraídos nos contornos são submetidos a um processo de simplificação. Estruturas de dados como grafos e árvore geradora mínima são utilizadas para manter apenas os pontos mais significativos. Nesta etapa utilizamos o algoritimo que encontra a árvore geradora mínima através da biblioteca *Scipy* (Virtanen *et al.*, 2020).

1.3.4 Modelagem com *splines*

Com os pontos reduzidos, são traçadas curvas suaves utilizando *splines* Catmull-Rom (Catmull; Rom, 1974; Twigg, 2003; Maggini *et al.*, 2007). Essa modelagem permite representar as feições faciais de forma contínua e diferenciável, garantindo uma estrutura mais compacta e interpretável.

1.3.5 Classificação por Dynamic Time Warping

Por fim, os pontos gerados a partir dessas curvas são utilizados como entrada para um algoritmo de *Dynamic Time Warping* (DTW) (Sakoe; Chiba, 1978; Tavenard, 2021). O DTW é um algoritmo que mede a similaridade entre duas sequências temporais, permitindo o alinhamento não linear entre elas. Essa abordagem é especialmente útil para lidar com variações na velocidade e, neste trabalho, na forma das características faciais.

1.4 REFERENCIAL TEÓRICO

A ideia de representar características faciais por meio de *splines* é discutida em (Maggini *et al.*, 2007), que apresenta uma abordagem baseada em *splines* Catmull-Rom para modelagem de curvas suaves.

O uso do algoritmo de Dynamic Time Warping (DTW) para reconhecimento facial é abordado em (Levada *et al.*, 2008), em que os autores discutem a eficácia do DTW na comparação de sequências temporais, destacando sua aplicabilidade em tarefas de reconhecimento facial. No entanto, a abordagem proposta no artigo difere da adotada neste trabalho, uma vez que os autores concentram-se na análise das

variações das cores dos pixels da imagem, em vez de explorar a forma ou a geometria das características faciais.

1.5 CRONOGRAMA

A Tabela 1 apresenta o cronograma de atividades para o desenvolvimento do trabalho, com as datas previstas para cada etapa.

TABELA 1 – Cronograma de Atividades

Atividade	**Mês 1**	**Mês 2**	**Mês 3**	**Mês 4**
Refatoração do código	X			
Revisão Bibliográfica	X	X	X	X
Estudo sobre *Dynamic Time Warping*		X	X	
Implementação do DTW		X	X	
Escrita da Monografia	X	X	X	X
Defesa do Trabalho				X

2 EXTRAÇÃO DE PONTOS DE FORMA AUTOMÁTICA

Neste capítulo, será apresentado o método de extração automática de pontos de forma a facilitar a modelagem de *splines* para reconhecimento facial. O método proposto consiste em um algoritmo que utiliza técnicas de processamento de imagem e otimização para extrair pontos relevantes em imagens faciais. A abordagem é baseada na detecção de características faciais, seguida pela extração de contornos e filtragem dos pontos obtidos.

Inicialmente, realizamos uma pesquisa para identificar as principais características faciais a serem extraídas, decidindo focar na detecção dos olhos, nariz e boca. Para isso, utilizamos o algoritmo Haar Cascade para detectar o rosto da pessoa na imagem, concentrando a análise nessa área específica e facilitando a detecção das características menores (Viola; Jones, 2001).

Em seguida, utilizamos o método `Canny` do OpenCV (OpenCV, s.d.[b]) para extrair os contornos das características faciais. O `Canny` é um algoritmo de detecção de contornos eficiente que nos ajuda a identificar os limites das características faciais com maior precisão (Canny, 1986).

Após a extração dos contornos, suprimimos alguns pontos fazendo o uso de grafos e árvores geradoras mínimas. Essa etapa é crucial para reduzir a quantidade de pontos a serem utilizados na modelagem das *splines*, mantendo apenas os pontos mais significativos que representam as características faciais. A simplificação dos pontos é realizada utilizando o algoritmo construído de forma autoral e implementado em Python, que utiliza a biblioteca *Scipy* (Virtanen *et al.*, 2020) para encontrar a árvore geradora mínima. Essa abordagem garante que os pontos extraídos sejam representativos e relevantes para a modelagem das *splines*.

Por fim, ainda é possível aplicar um filtro de suavização para diminuir a quantidade de pontos, fazendo isso de forma completamente aleatória, mantendo apenas os pontos iniciais e finais de cada segmento, o que pode ser útil para melhorar a qualidade da modelagem das *splines*.

2.1 MODELOS PARA DETECÇÃO DE CARACTERÍSTICAS

Foi utilizada uma abordagem popular de detecção chamada Haar Cascade, implementada com OpenCV e Python. Este método, introduzido e estudado no artigo (Viola; Jones, 2001), permite a detecção eficaz das características faciais.

O classificador Haar Cascade já foi calibrado e validado com um vasto conjunto

de dados de rostos humanos, eliminando a necessidade de calibração adicional. Basta carregar o classificador da biblioteca e utilizá-lo para detectar rostos em uma imagem de entrada.

Para distinguir com precisão entre amostras que contêm ou não um rosto humano, utilizamos um classificador forte, resultante da combinação de diversos classificadores. Esta técnica envolve a utilização de uma cascata de classificadores para identificar diferentes características em uma imagem.

Os seguintes classificadores foram utilizados:

- `haarcascade_frontalface_default.xml`
- `haarcascade_eye.xml`
- `haarcascade_mcs_nose.xml`
- `haarcascade_mcs_mouth.xml`

Todos os classificadores mencionados podem ser encontrados no repositório do OpenCV no GitHub (OPENCV. . . , s.d.[a]).

2.1.1 Parâmetros

Após carregar os classificadores, podemos utilizá-los aplicando quatro parâmetros específicos para cada um.

O método `detectMultiScale()` é utilizado para identificar faces de diferentes tamanhos na imagem de entrada. Os quatro parâmetros principais deste método são detalhados a seguir:

- `image`:
  - O primeiro parâmetro é a imagem em tons de cinza. Essa imagem serve como entrada para o método e com ela em tons de cinza facilita a detecção de características faciais.
- `scaleFactor`:
  - Este parâmetro é usado para reduzir o tamanho da imagem de entrada, facilitando a detecção de faces maiores pelo algoritmo. Especificamos um fator de escala de 1.1, indicando que queremos reduzir o tamanho da imagem em 10%.
- `minNeighbors`:

  - O classificador em cascata aplica uma janela deslizante através da imagem para detectar faces. Essas janelas são representadas como retângulos. Inicialmente, o classificador captura um grande número de falsos positivos. O parâmetro `minNeighbors` especifica o número de retângulos vizinhos que precisam ser identificados para que um objeto seja considerado uma detecção válida, ou seja, valores pequenos como 0 ou 1 resultam em muitos falsos positivos, enquanto valores grandes podem levar à perda de verdadeiros positivos. É necessário encontrar um equilíbrio que elimine falsos positivos e identifique com precisão os verdadeiros positivos.

- `minSize`:

  - Este parâmetro define o tamanho mínimo do objeto a ser detectado. O modelo ignorará faces menores do que o tamanho mínimo especificado.

  Foi colocado como default para todos os classificadores os seguintes valores:

  ```
  detectMultiScale(image, scaleFactor=1.1, minNeighbors=5, minSize=(40, 50))
  ```

2.1.2 Resultado da Detecção das Características

Para realizar a detecção das características faciais, inicialmente identificamos o rosto na imagem. Este processo retorna um *array* com quatro valores: as coordenadas x e y do ponto onde o rosto foi detectado, além de sua largura e altura. Em seguida, recortamos a imagem nessa região, de modo a isolar a face da pessoa.

Com o rosto isolado, aplicamos classificadores específicos para detectar o nariz, a boca e os olhos.

Essa abordagem, que realiza a detecção passo a passo, garante maior precisão na identificação das características menores, pois concentra a análise na área previamente delimitada pelo rosto.

Como exemplo, considere a **??** e a **??** , onde a sequência de detecção é ilustrada:

1. **Detecção do Rosto:** Para identificar e isolar o rosto na imagem usamos o classificador `haarcascade_frontalface_default.xml`.
2. **Detecção dos Olhos:** Aplicamos o classificador `haarcascade_eye.xml` para localizar os olhos dentro da área do rosto previamente detectada.
3. **Detecção do Nariz:** Utilizamos o classificador `haarcascade_mcs_nose.xml` para identificar o nariz na mesma área delimitada.

4. **Detecção da Boca:** Finalmente, aplicamos o classificador `haarcascade_mcs_mouth.xml` para localizar a boca.

2.2 EXTRAÇÃO DE CONTORNOS

A detecção de bordas é essencial para a análise de imagens, pois permite a identificação de contornos e a segmentação de objetos em uma cena. O algoritmo (Canny, 1986) é um dos métodos mais populares devido à sua eficiência e precisão. Esta parte irá explorar o funcionamento do algoritmo de Canny e demonstrará sua aplicação prática com a biblioteca OpenCV (OpenCV, s.d.[b]).

2.2.1 Algoritmo de Canny

O algoritmo de detecção de bordas de Canny é composto por várias etapas, conforme descrito a seguir:

1. **Redução de Ruído**: A imagem é suavizada com um filtro Gaussiano para minimizar a interferência do ruído na detecção de bordas.
2. **Cálculo do Gradiente de Intensidade**: Os gradientes de intensidade são calculados nas direções horizontal G_x e vertical G_y usando operadores Sobel. A mag nitude do gradiente é dada por:

$$G_x = \frac{\partial I}{\partial x} \quad G_y = \frac{\partial I}{\partial y}$$

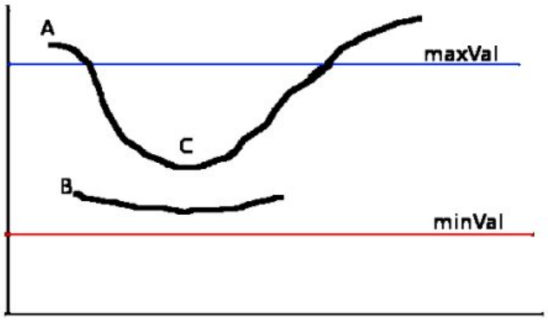
$$Borda\_Gradiente(G) = \sqrt{G_x^2 + G_y^2}$$

3. **Supressão Não Máxima**: Elimina *pixels* que não são máximos locais no gradiente, preservando apenas os que representam bordas fortes.
4. **Rastreamento de Bordas por Histerese**: Classifica as bordas detectadas usando dois limiares, *minVal* e *maxVal*:
   - Bordas com gradiente maior que *maxVal* são fortes.
   - Bordas com gradiente menor que *minVal* são descartadas.
   - Bordas entre *minVal* e *maxVal* são fracas e mantidas apenas se conectadas a bordas fortes.

Por exemplo, na Figura FIGURA 1, a borda A está acima do *maxVal*, sendo assim considerada uma *sure-edge*. Embora a borda C esteja abaixo do *maxVal*, ela está conectada à borda A, de modo que também é considerada como borda válida,

resultando em uma curva completa. Já a borda B, embora esteja acima do *minVal*, não está conectada a nenhuma *sure-edge* e, portanto, é descartada.

FIGURA 1 – RASTREAMENTO DE CONTORNO POR HISTERESE



FONTE: (OpenCV, s.d.[b])
LEGENDA: Calculada a magnitude do gradiente ao longo de uma curva parametrizada.

É crucial selecionar adequadamente os valores de *minVal* e *maxVal* para obter resultados precisos. Esse estágio também ajuda a remover pequenos ruídos de pixels, assumindo que as bordas são linhas longas e contínuas.

2.3 RESULTADO DA DETECÇÃO DE BORDAS

Ao aplicar o algoritmo de Canny para a detecção de bordas, obtemos um array binário onde os valores ’0’ e ’255’ representam diferentes tipos de pixels. Os pixels com valor ’0’ são considerados descartáveis, ou seja, não fazem parte das bordas detectadas. Em contraste, os pixels com valor ’255’ são os que definem as bordas, indicando regiões de mudança de intensidade significativa na imagem original.

A seguir são apresentados os resultados da aplicação do algoritmo de Canny em imagens de exemplo. As Figuras , e mostram as imagens originais e as imagens com as bordas detectadas.

REFERÊNCIAS

CANNY, J. A Computational Approach to Edge Detection. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, PAMI-8, n. 6, p. 679–698, 1986. DOI: 10.1109/TPAMI.1986.4767851. Citado 3 vezes nas páginas 7, 9, 12.

CATMULL, E.; ROM, R. A CLASS OF LOCAL INTERPOLATING SPLINES. *In:* BARNHILL, R. E.; RIESENFELD, R. F. (ed.). **Computer Aided Geometric Design**. [*S. l.*]: Academic Press, 1974. p. 317–326. ISBN 978-0-12-079050-0. Disponível em: https://doi.org/10.1016/B978-0-12-079050-0.50020-5. Citado 1 vez na página 7.

LEVADA, A. L. M.; CORREA, D. C.; SALVADEO, D. H. P.; SAITO, J. H.; MASCARENHAS, N. D. A. Novel approaches for face recognition: Template-matching using Dynamic Time Warping and LSTM neural network supervised classification. *In:* 2008 15th International Conference on Systems, Signals and Image Processing. [*S. l.: s. n.*], 2008. p. 241–244. DOI: 10.1109/IWSSIP.2008.4604412. Citado 1 vez na página 7.

MAGGINI, M.; MELACCI, S.; SARTI, L. Representation of Facial Features by Catmull-Rom Splines. *In:* p. 408–415. ISBN 978-3-540-74271-5. DOI: 10.1007/978-3-540-74272-2_51. Citado 2 vez na página 7.

OPENCV. [*S. l.: s. n.*]. https://github.com/opencv/opencv/tree/master/data. Acessado em: 24-05-2024. Citado 2 vezes nas páginas 6, 10.

OPENCV. **Canny Edge Detection**. Disponível em: https://docs.opencv.org/3.4/da/d22/tutorial_py_canny.html. Citado 3 vezes nas páginas 7, 9, 12, 13.

SAKOE, H.; CHIBA, S. Dynamic programming algorithm optimization for spoken word recognition. **IEEE Transactions on Acoustics, Speech, and Signal Processing**, v. 26, n. 1, p. 43–49, 1978. DOI: 10.1109/TASSP.1978.1163055. Citado 2 vezes nas páginas 5, 7.

TAVENARD, R. **An introduction to Dynamic Time Warping**. [*S. l.: s. n.*], 2021. https://rtavenar.github.io/blog/dtw.html. Citado 1 vez na página 7.

TWIGG, C. **Catmull-Rom splines**. Pittsburg: CMU school of computer science, 2003. Disponível em: http://graphics.cs.cmu.edu/nsp/course/15-462/Fall04/assts/catmullRom.pdf. Citado 1 vez na página 7.

VIOLA, P.; JONES, M. Rapid object detection using a boosted cascade of simple features. *In:* PROCEEDINGS of the 2001 IEEE Computer Society Conference on

Computer Vision and Pattern Recognition. CVPR 2001. [*S. l.: s. n.*], 2001. v. 1. DOI: 10.1109/CVPR.2001.990517. Citado 3 vezes nas páginas 6, 9.

VIRTANEN, P.; GOMMERS, R.; OLIPHANT, T. E.; HABERLAND, M.; REDDY, T.; COURNAPEAU, D.; BUROVSKI, E.; PETERSON, P.; WECKESSER, W.; BRIGHT, J.; VAN DER WALT, S. J.; BRETT, M.; WILSON, J.; MILLMAN, K. J.; MAYOROV, N.; NELSON, A. R. J.; JONES, E.; KERN, R.; LARSON, E.; CAREY, C. J.; POLAT, İ.; FENG, Y.; MOORE, E. W.; VANDERPLAS, J.; LAXALDE, D.; PERKTOLD, J.; CIMRMAN, R.; HENRIKSEN, I.; QUINTERO, E. A.; HARRIS, C. R.; ARCHIBALD, A. M.; RIBEIRO, A. H.; PEDREGOSA, F.; VAN MULBREGT, P.; SCIPY 1.0 CONTRIBUTORS. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. **Nature Methods**, v. 17, p. 261–272, 2020. DOI: 10.1038/s41592-019-0686-2. Citado 2 vezes nas páginas 7, 9.