Explorando o algoritmo de Viola-Jones na detecção e reconhecimento facial

Julio Batista Silva

Orientador: Prof. Dr. Alexandre Luis Magalhães Levada
Departamento de Computação
Universidade Federal de São Carlos

12 de julho de 2018

Sumário

Introdução
- Motivação
- Objetivos

Detecção Facial
- Viola-Jones
- OpenCV

Reconhecimento Facial
- PCA
- Eigenfaces
- OpenCV

Implementação do Projeto
- Módulo 1
- Módulo 2

Conclusões

Referências

Sumário

Introdução

Detecção Facial

Reconhecimento Facial

Implementação do Projeto

Conclusões

Referências

Motivação

Uso de reconhecimento facial em lojas de varejo

Motivação

Uso de reconhecimento facial em lojas de varejo para:

- Segurança

Motivação

Uso de reconhecimento facial em lojas de varejo para:

- Segurança
- Personalização de serviço

Motivação

Uso de reconhecimento facial em lojas de varejo para:

- Segurança
- Personalização de serviço
- Marketing

Motivação

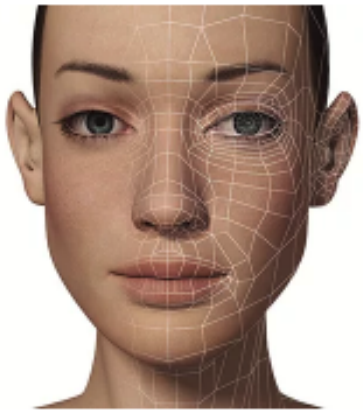
Uso de reconhecimento facial em lojas de varejo para:

- Segurança
- Personalização de serviço
- Marketing
- Análise de sentimento

FORTUNE SUBSCRIBE

FACIAL RECOGNITION

Walmart's Use of Sci-fi Tech To Spot Shoplifters Raises Privacy Questions



The biometric facial map helps read emotions on a human face. Photograph by Alfred Pasieka — Getty Images/Science Photo Libra

By JEFF JOHN ROBERTS November 9, 2015

VAREJISTAS USAM CADA VEZ MAIS O RECONHECIMENTO FACIAL PARA ENCONTRAR GRANDES COMPRADORES E... LADRÕES

25 DE JANEIRO DE 2017 WANDER F. S. NETO

Usar câmeras de segurança para identificar e deter os ladrões têm sido uma prática comum entre os varejistas, mas agora parece que estas lojas estão aprimorando seus sistemas de segurança e utilizando a tecnologia de reconhecimento facial para outros fins.

De acordo com um relatório da BBC, um número crescente de varejistas no Reino Unido estão investindo pesado em câmeras com tecnologia de reconhecimento facial para alem identificar ladrões conhecidos, aplicar a mesma tecnologia para identificar também grandes consumidores e clientes em potencial.

EXAME Assine

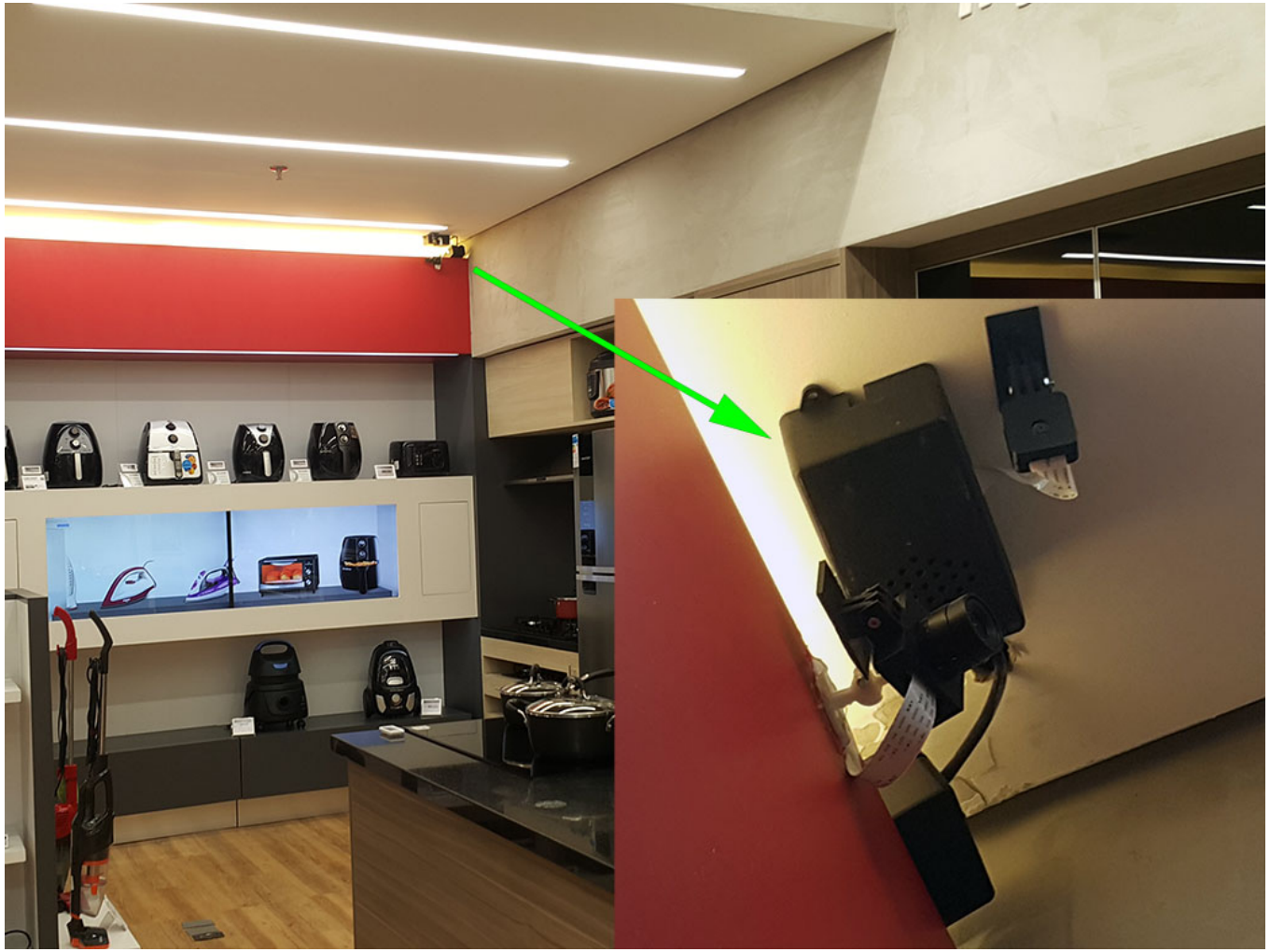
NEGÓCIOS

Loja Pontofrio tem de vitrine virtual a reconhecimento facial

Na nova unidade, vendedores terão acesso a histórico de compras online, câmeras analisam a satisfação do consumidor e há estantes virtuais em tamanho real

Por **Karin Salomão**

29 jan 2018, 14h59 - Publicado em 29 jan 2018, 11h14



Objetivos

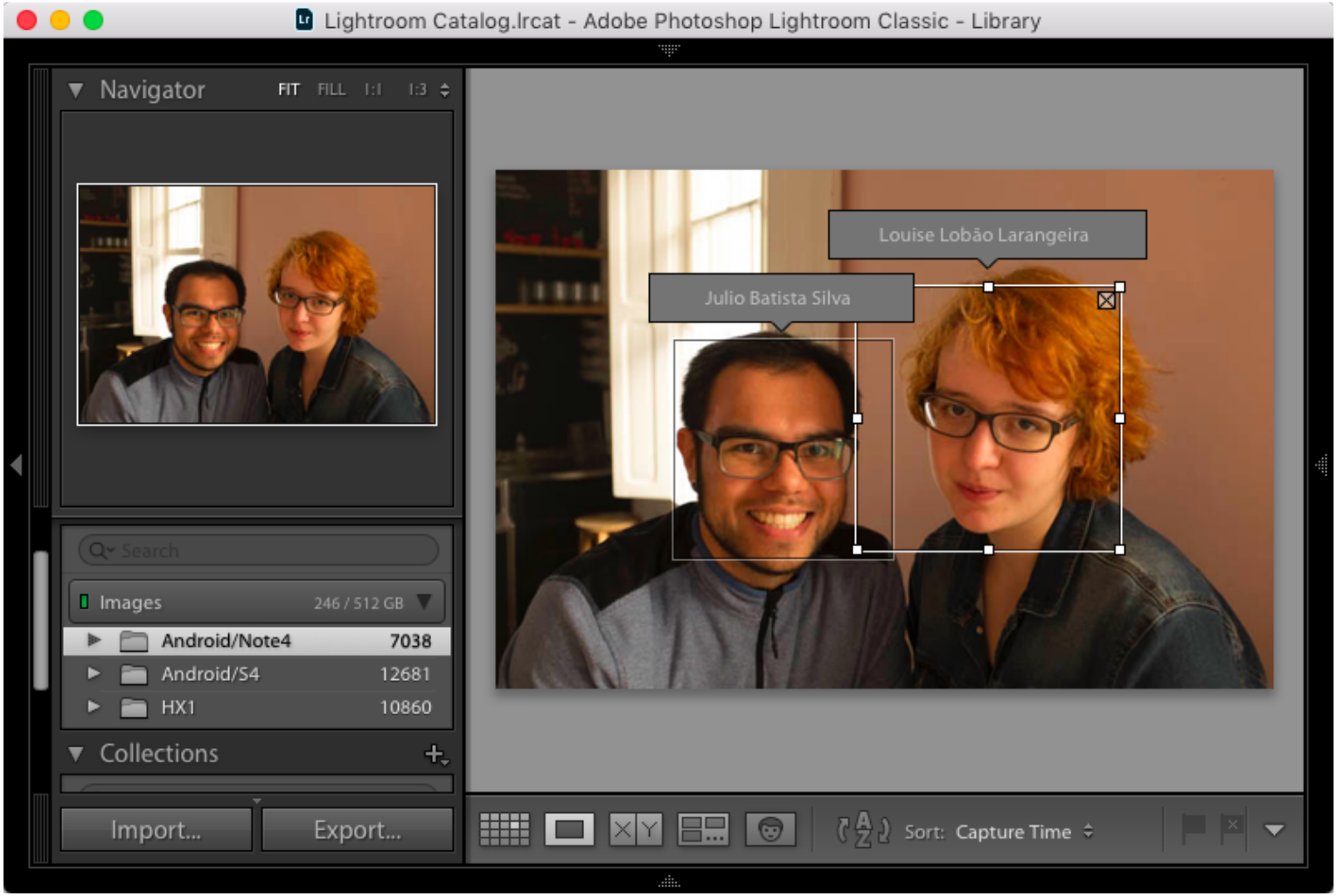
- Fornecer uma base teórica e prática para a construção de um sistema de detecção e reconhecimento facial, que será utilizado em uma loja de varejo
- Encontrar bibliotecas, APIs e ferramentas que auxiliem no desenvolvimento do projeto

Aplicações

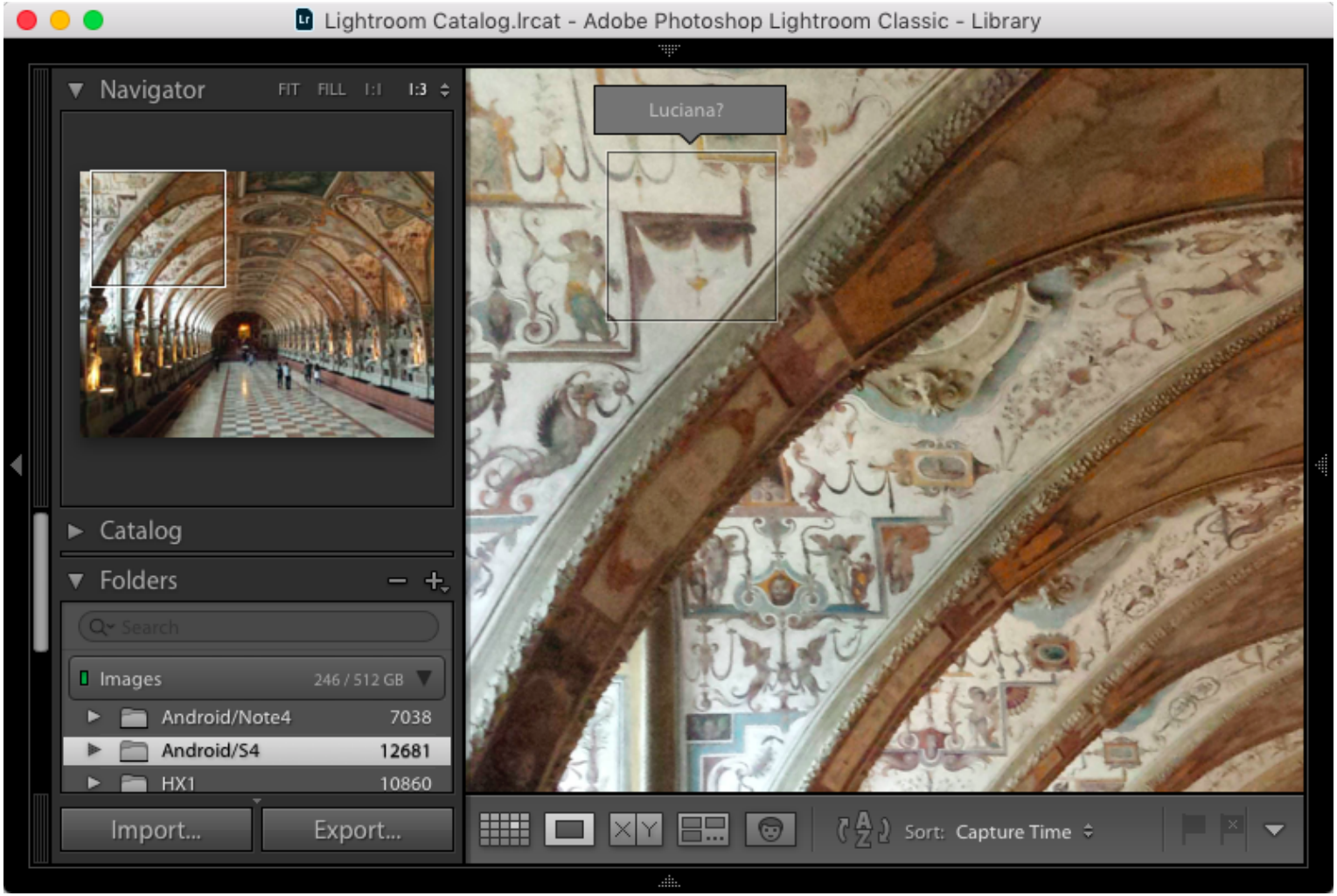
Aplicações de detecção e reconhecimento facial:

- Identificação criminal
- Sistemas de segurança
- Biometria
- Processamento de imagens
- Interação humano-computador

Exemplo

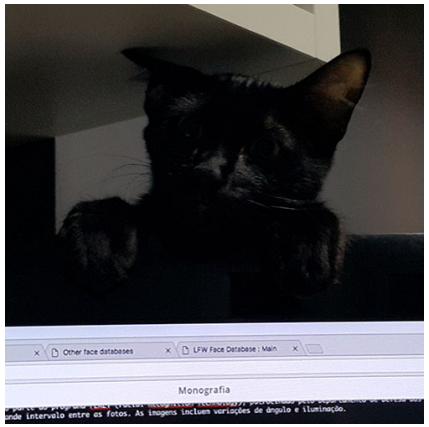

Lightroom Catalog.lrcat - Adobe Photoshop Lightroom Classic - Library
Navigator
FIT FILL 1:1 1:3
Louise Lobão Larangeira
Julio Batista Silva
Search
Images 246 / 512 GB
Android/Note4 7038
Android/S4 12681
HX1 10860
Collections
Import...
Export...
Sort: Capture Time

Nem sempre funciona como esperado



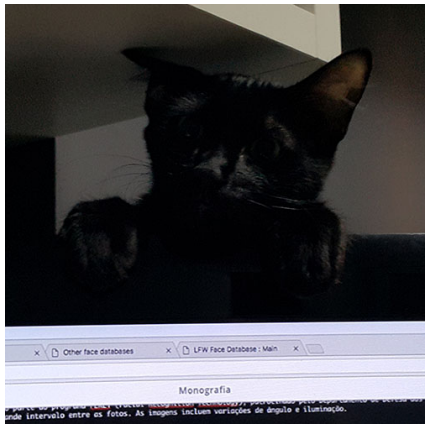
Dificuldades

Iluminação



Dificuldades

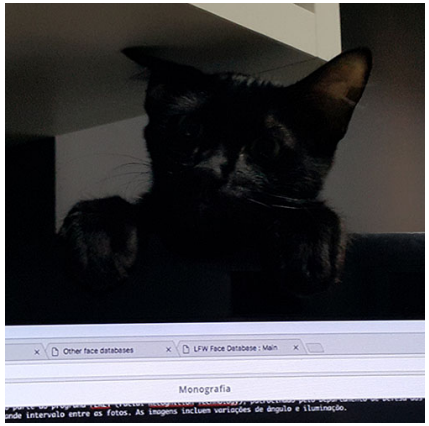
Iluminação



Pose

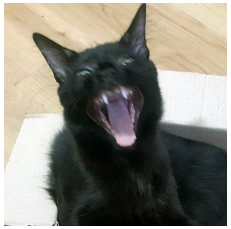
Dificuldades

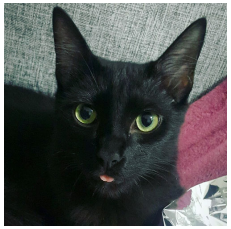
Iluminação



Pose

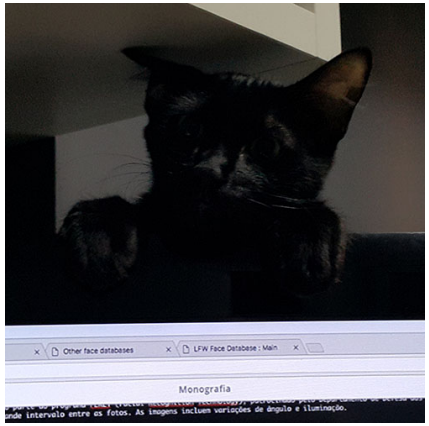
Expressão Facial





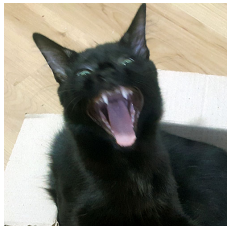
Dificuldades

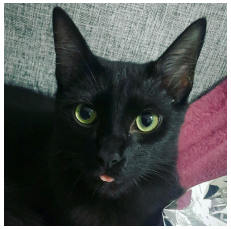
Iluminação



Pose

Expressão Facial

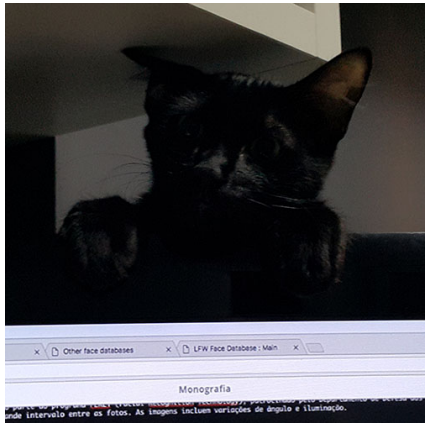




Oclusão

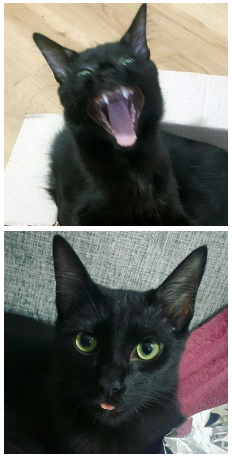
Dificuldades

Iluminação



Pose

Expressão Facial



Oclusão

Ponto de Vista

Sumário

Introdução

Detecção Facial

Reconhecimento Facial

Implementação do Projeto

Conclusões

Referências

Definição

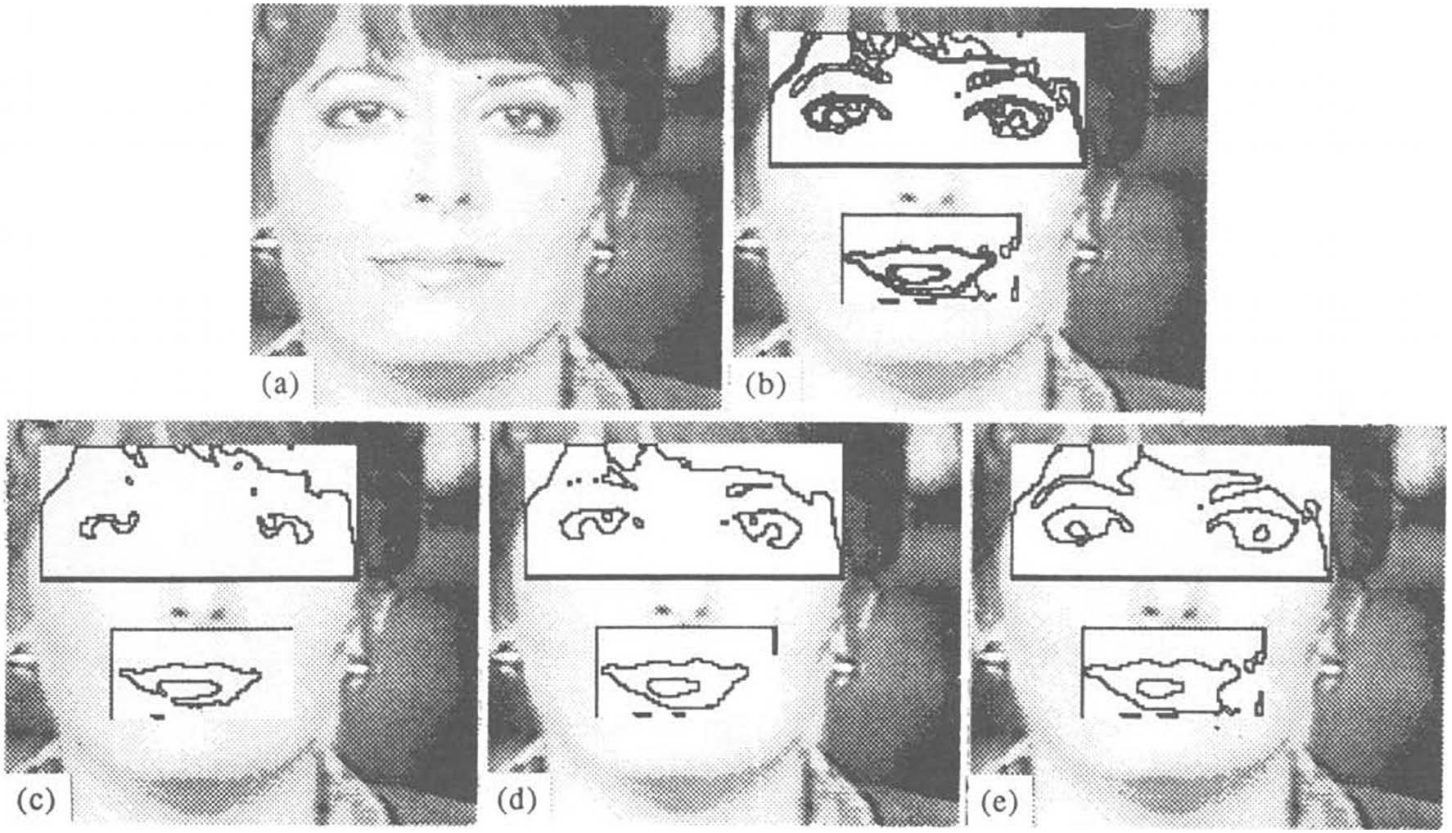
O papel de um detector de faces é, dada uma imagem arbitrária, determinar se ela contém faces ou não e, caso positivo, retornar a localização e dimensões de cada face (CEN, 2016)

Fonte: RSCG (2008)

MÉTODOS

- Baseado em conhecimento

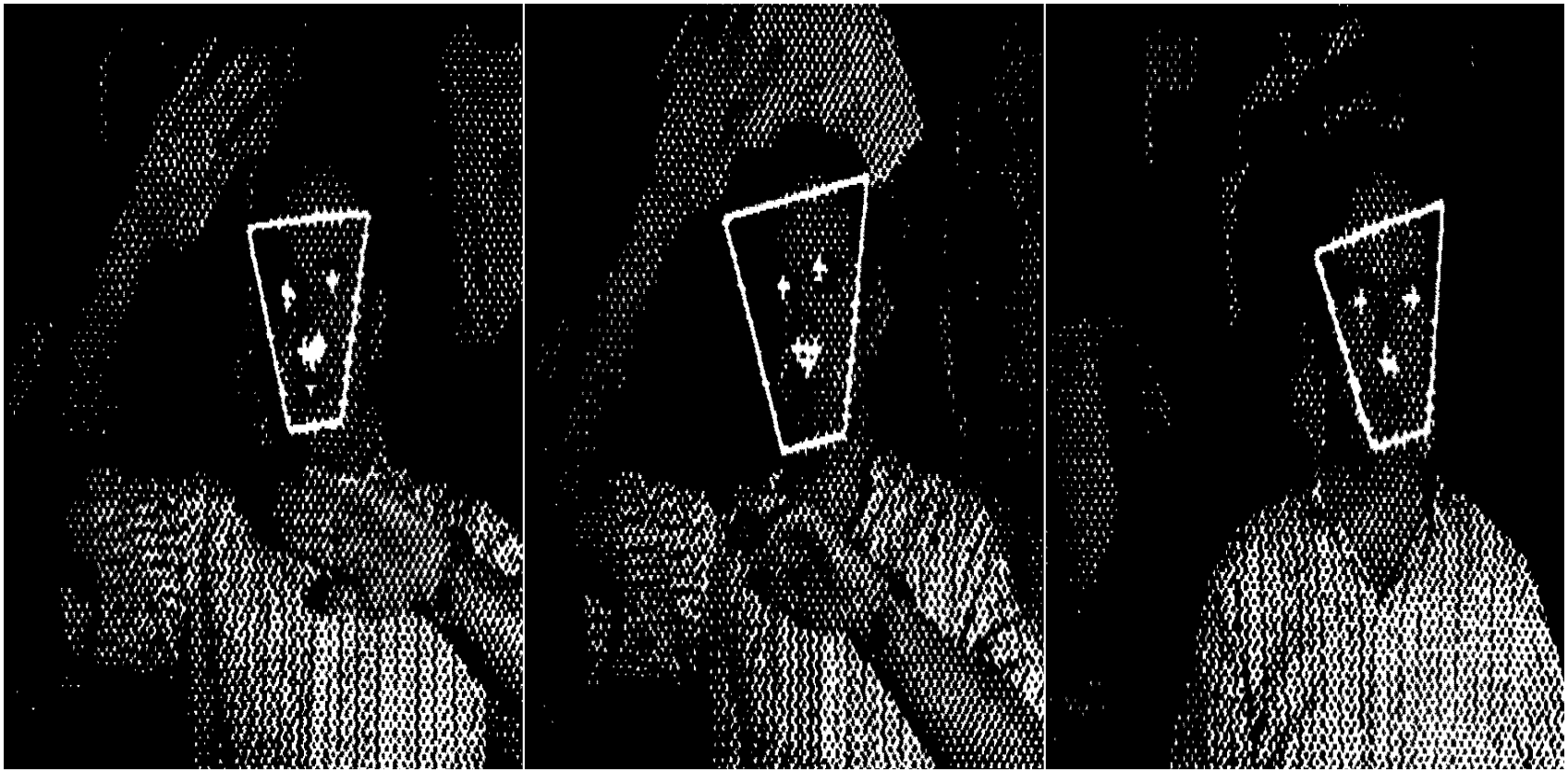
Fonte: Yang e Huang (1994)



MÉTODOS

- Baseado em características invariantes

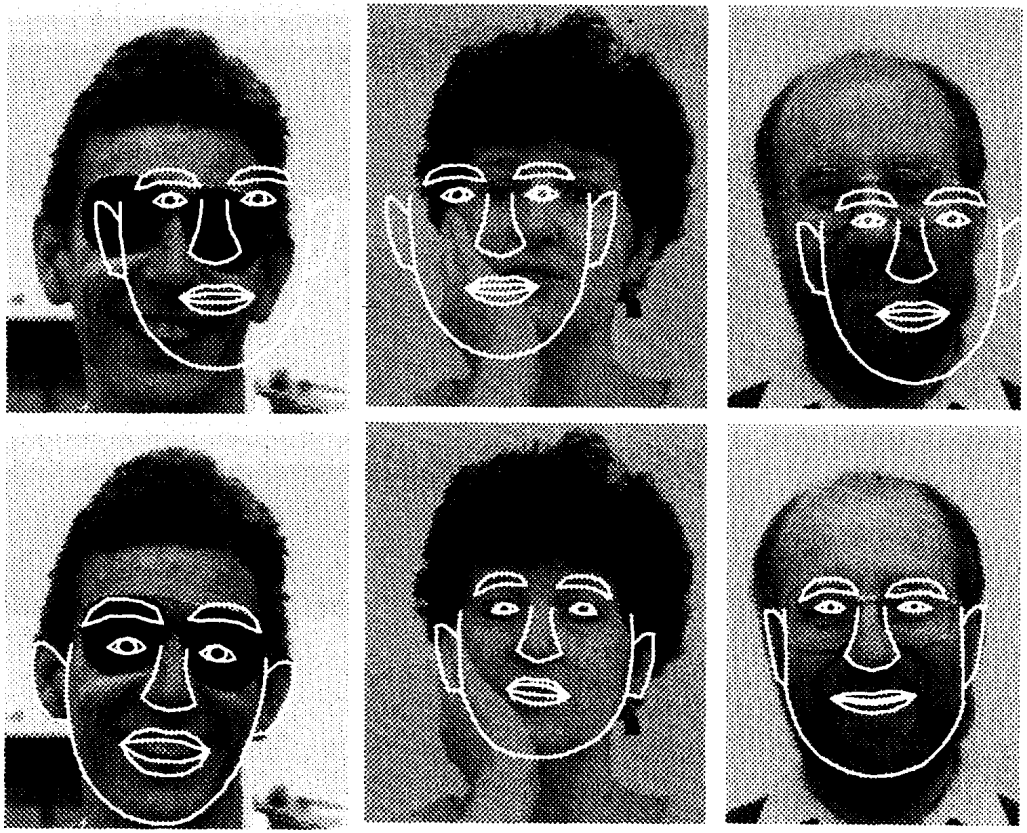
Fonte: Leung, Burl e Perona (1995)



Métodos

- Casamento de padrões

Fonte: Lanitis, Taylor e Cootes (1995)



MÉTODOS

- Baseado na aparência

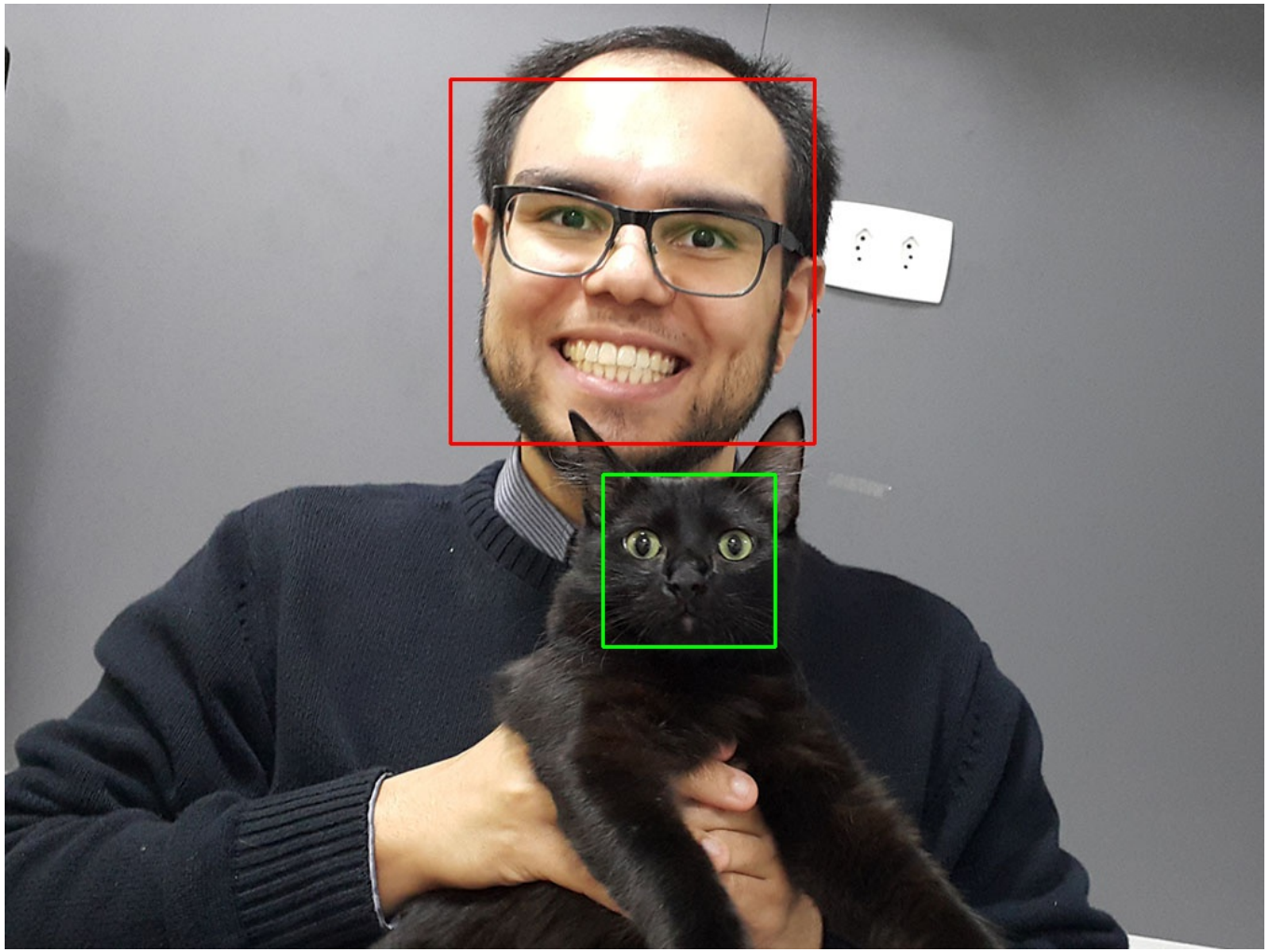
Fonte: Turk e Pentland (1991)

VIOLA-JONES

- Algoritmo apresentado por Paul Viola e Michael Jones em 2001 (VIOLA; JONES, 2001)
- Foi o primeiro método de detecção de faces em tempo real Omaia (2009)

VIOLA-JONES

- Não é apenas para faces humanas



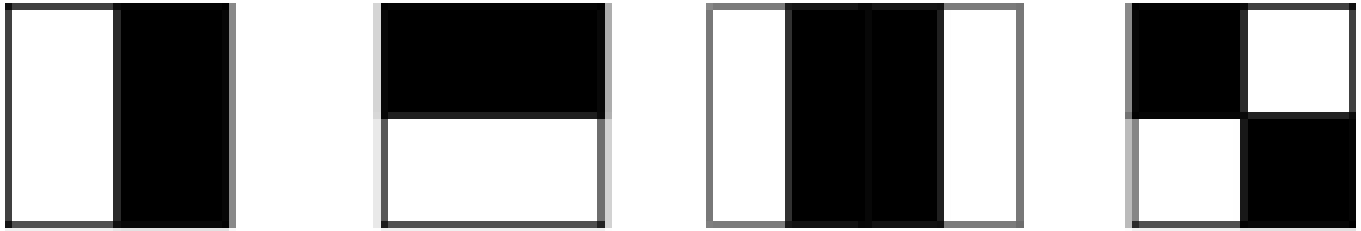
VIOLA-JONES

Quatro estágios:

- Seleção de características Haar-like retangulares
- Criação da imagem integral
- Treino de classificadores por AdaBoost
- Uso de classificadores em cascata

Características Haar

- Formadas por dois, três ou quatro retângulos
- O cálculo de uma única característica é bem simples: soma das intensidades dos pixels sob os retângulos pretos menos a soma dos valores sob os retângulos brancos



1 2 3 4

Características Haar

- Compara quão próximo o caso real está do caso ideal

0	0	1	1
0	0	1	1
0	0	1	1
0	0	1	1

Intensidades ideais

0.2	0.2	0.8	0.6
0.1	0.3	0.6	0.8
0.2	0.1	0.8	0.8
0.2	0.1	0.6	0.9

Valores reais

$$\Delta = \frac{1}{n}\sum_{preto}^{n} I(x) - \frac{1}{n}\sum_{branco}^{n} I(x)$$

$$= 0{,}74 - 0{,}18 = 0{,}56$$

Características Haar

- Características usadas para detectar olhos e nariz

Características Haar

Para cada subjanela de 24×24 pixels, seria necessário calcular 45396 características

Inviável!

Imagem Integral

- Permite calcular as características em tempo constante
- É uma tabela onde cada elemento equivale à soma de todos os pixels à esquerda e acima do pixel atual

$$ii(x,y) = \sum_{x' \leq x, y' \leq y} i(x', y')$$

Imagem Integral

Imagem original

0.1	0.1	0.2	0.1	0.7	0.1
0.2	0.3	0.2	0.7	0.8	0.2
0.1	0.4	0.3	0.3	0.1	0.3
0.1	0.5	0.1	0.1	0.2	0.8
0.1	0.4	0.8	0.5	0.6	0.5

Imagem integral

0.1	0.2	0.4	0.5	1.2	1.3
0.3	0.7	1.1	1.9	3.4	3.7
0.4	1.2	1.9	3.0	4.6	5.2
0.5	1.7	2.5	3.7	5.3	6.7
0.6	2.3	3.9	5.6	8.0	9.9

Imagem Integral

Imagem original

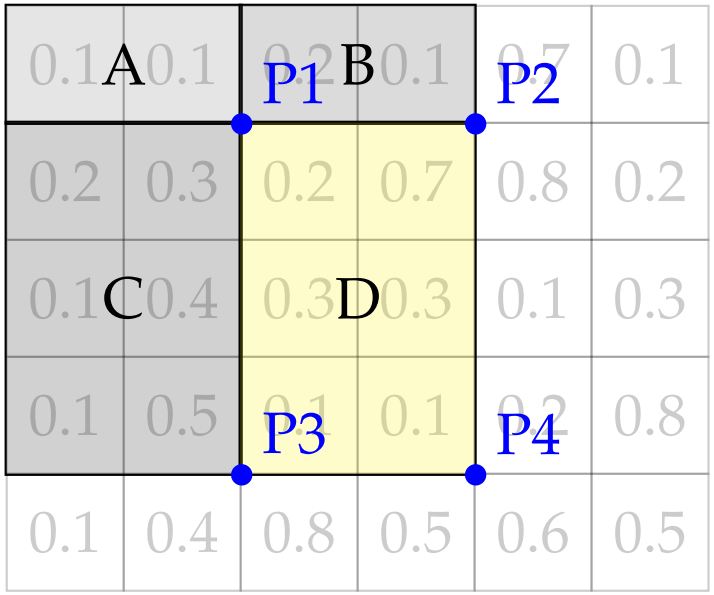


Imagem integral

$ii(P1)$ $ii(P2)$

0.1	0.2	0.4	0.5	1.2	1.3
0.3	0.7	1.1	1.9	3.4	3.7
0.4	1.2	1.9	3.0	4.6	5.2
0.5	1.7	2.5	3.7	5.3	6.7
0.6	2.3	3.9	5.6	8.0	9.9

$ii(P3)$ $ii(P4)$

$$D = (ii(P4) + ii(P1)) - (ii(P2) + ii(P3))$$
$$= (3{,}7 + 02) - (0{,}5 + 1{,}7) = 1{,}7$$

Imagem Integral

Imagem original

0.1	0.1	0.2	0.1	0.7	0.1
0.2	0.3	0.2	0.7	0.8	0.2
0.1	0.4	0.3	0.3	0.1	0.3
0.1	0.5	0.1	0.1	0.2	0.8
0.1	0.4	0.8	0.5	0.6	0.5

Imagem integral

0.1	0.2	0.4	0.5	1.2	1.3
0.3	0.7	1.1	1.9	3.4	3.7
0.4	1.2	1.9	3.0	4.6	5.2
0.5	1.7	2.5	3.7	5.3	6.7
0.6	2.3	3.9	5.6	8.0	9.9

Imagem Integral

Imagem original

0.1	0.1	0.2	0.1	0.7	0.1
0.2	0.3	0.2	0.7	0.8	0.2
0.1	0.4	0.3	0.3	0.1	0.3
0.1	0.5	0.1	0.1	0.2	0.8
0.1	0.4	0.8	0.5	0.6	0.5

Imagem integral

0.1	0.2	0.4	0.5	1.2	1.3
0.3	0.7	1.1	1.9	3.4	3.7
0.4	1.2	1.9	3.0	4.6	5.2
0.5	1.7	2.5	3.7	5.3	6.7
0.6	2.3	3.9	5.6	8.0	9.9

Imagem Integral

Imagem original

0.1	0.1	0.2	0.1	0.7	0.1
0.2	0.3	0.2	0.7	0.8	0.2
0.1	0.4	0.3	0.3	0.1	0.3
0.1	0.5	0.1	0.1	0.2	0.8
0.1	0.4	0.8	0.5	0.6	0.5

Imagem integral

0.1	0.2	0.4	0.5	1.2	1.3
0.3	0.7	1.1	1.9	3.4	3.7
0.4	1.2	1.9	3.0	4.6	5.2
0.5	1.7	2.5	3.7	5.3	6.7
0.6	2.3	3.9	5.6	8.0	9.9

Imagem Integral

Imagem original

0.1	0.1	0.2	0.1	0.7	0.1
0.2	0.3	0.2	0.7	0.8	0.2
0.1	0.4	0.3	0.3	0.1	0.3
0.1	0.5	0.1	0.1	0.2	0.8
0.1	0.4	0.8	0.5	0.6	0.5

Imagem integral

0.1	0.2	0.4	0.5	1.2	1.3
0.3	0.7	1.1	1.9	3.4	3.7
0.4	1.2	1.9	3.0	4.6	5.2
0.5	1.7	2.5	3.7	5.3	6.7
0.6	2.3	3.9	5.6	8.0	9.9

AdaBoost

- Mesmo usando a imagem integral, calcular todas as características continua inviável
- É possível formar um classificador eficiente usando apenas as **características mais relevantes**. Elas podem ser selecionadas pelo algoritmo de aprendizado de máquina AdaBoost

AdaBoost

- **Ada**ptive **Boost**
- Combina, de forma ponderada, classificadores fracos para obter um classificador forte

$$F(x) = \alpha_1 f_1(x) + \alpha_2 f_2(x) + \alpha_3 f_3(x) \ldots$$

- f_1 → classificador fraco
- α_1 → peso
- x → imagem
- F → classificador forte

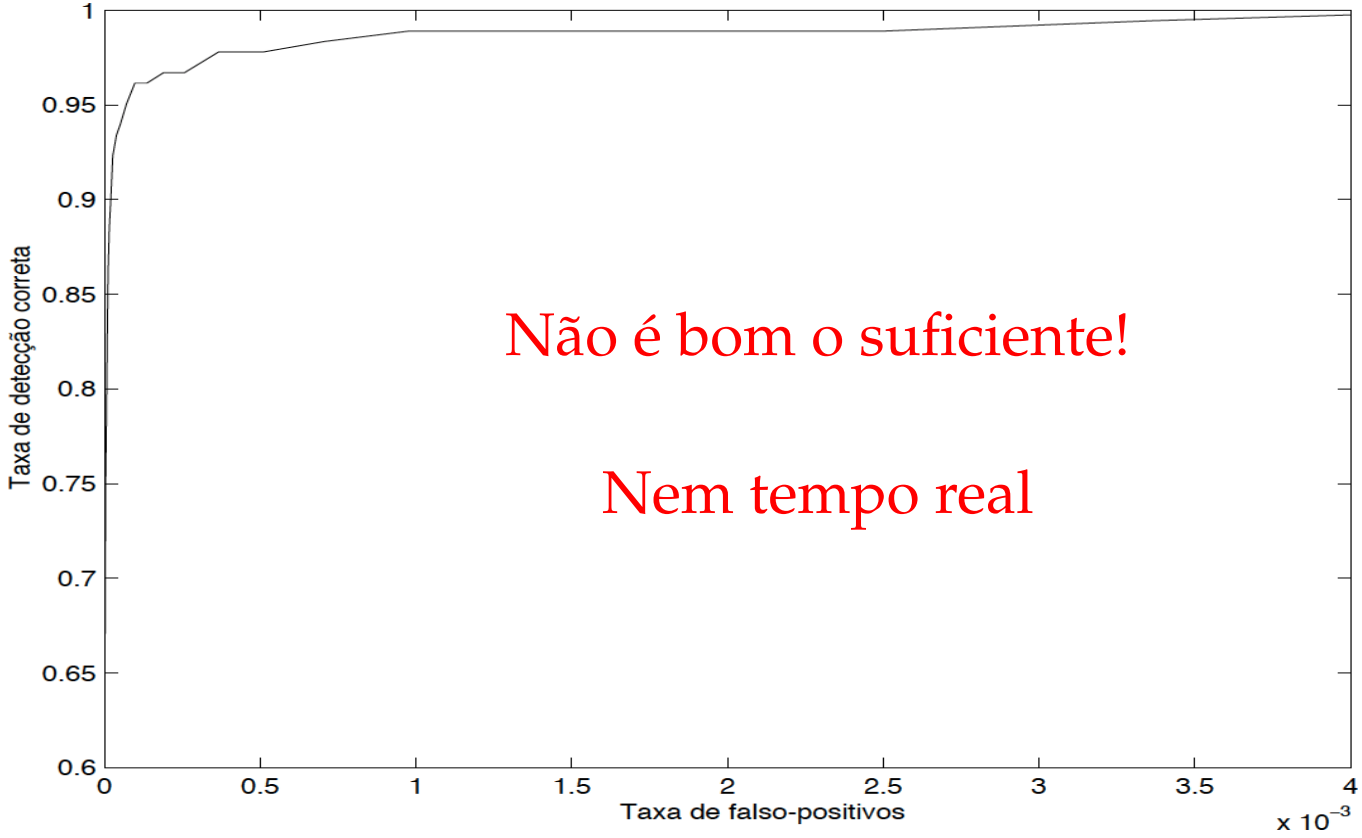
AdaBoost

Algoritmo:

- Inicia com pesos iguais para todas as imagens de treino
- Em cada iteração, encontra o classificador fraco com menor erro e aumenta o peso dos exemplos classificados erroneamente
- Calcula o classificador forte como uma combinação linear dos classificadores fracos

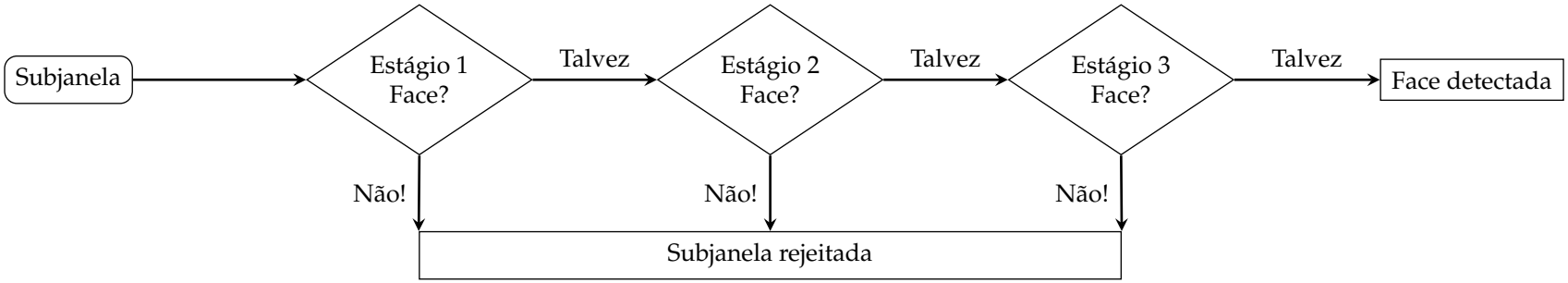
AdaBoost

- Classificador com 200 características:
  - 95% de detecção
  - 1 falso-positivo em 14084 ($1{,}4 \times 10^{-4}$ FPR)
  - 1,43 FPS (0,7 segundos para 384×288 pixels)



Classificadores em Cascata

- Gasta tempo apenas com faces em potencial
- Classificador de duas características: quase 100% de detecção com 50% de falso-positivos. Pode ser o primeiro estágio da cascata
- Uma subjanela que falha em qualquer estágio é imediatamente rejeitada

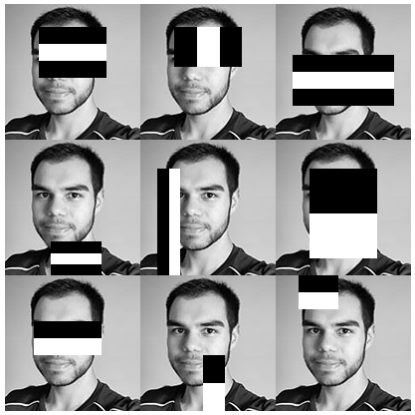


Classificadores em Cascata

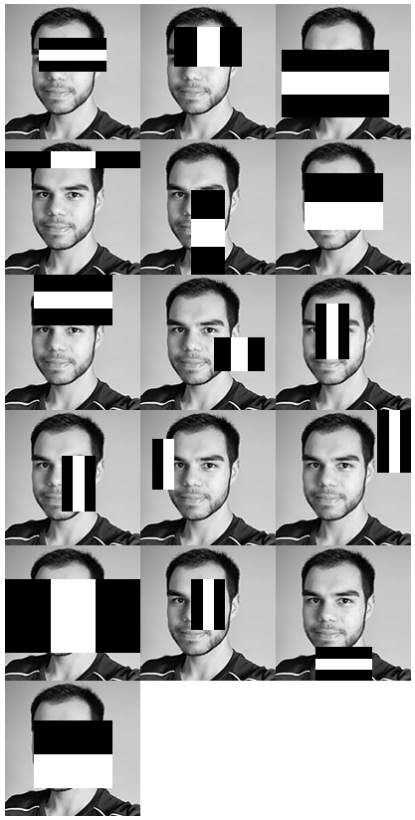
- Os estágios são progressivamente mais complexos e com menor taxa de falso-positivos
- A taxa de detecção da cascata é obtida multiplicando as taxas de cada estágio individual
- Uma cascata de 10 estágios, cada um com 99% de detecção e um pouco menos de 30% de falso-positivos resulta em uma taxa de detecção de 90% e uma taxa de falso-positivos na ordem de 10^{-6}

Classificadores em Cascata

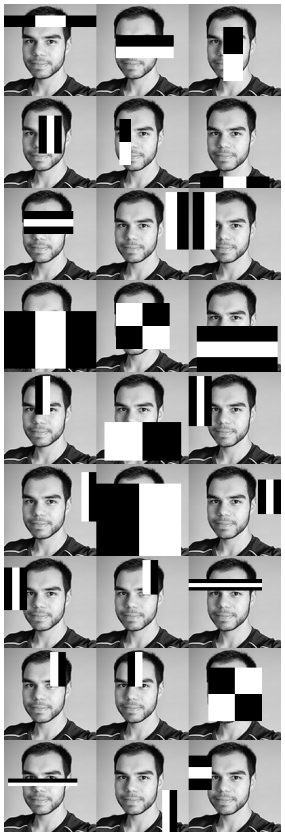
Estágio 1



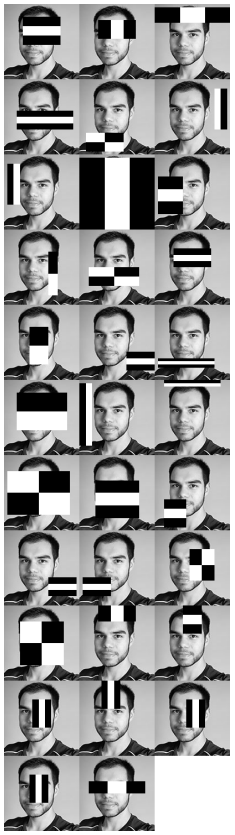
Estágio 2



Estágio 3

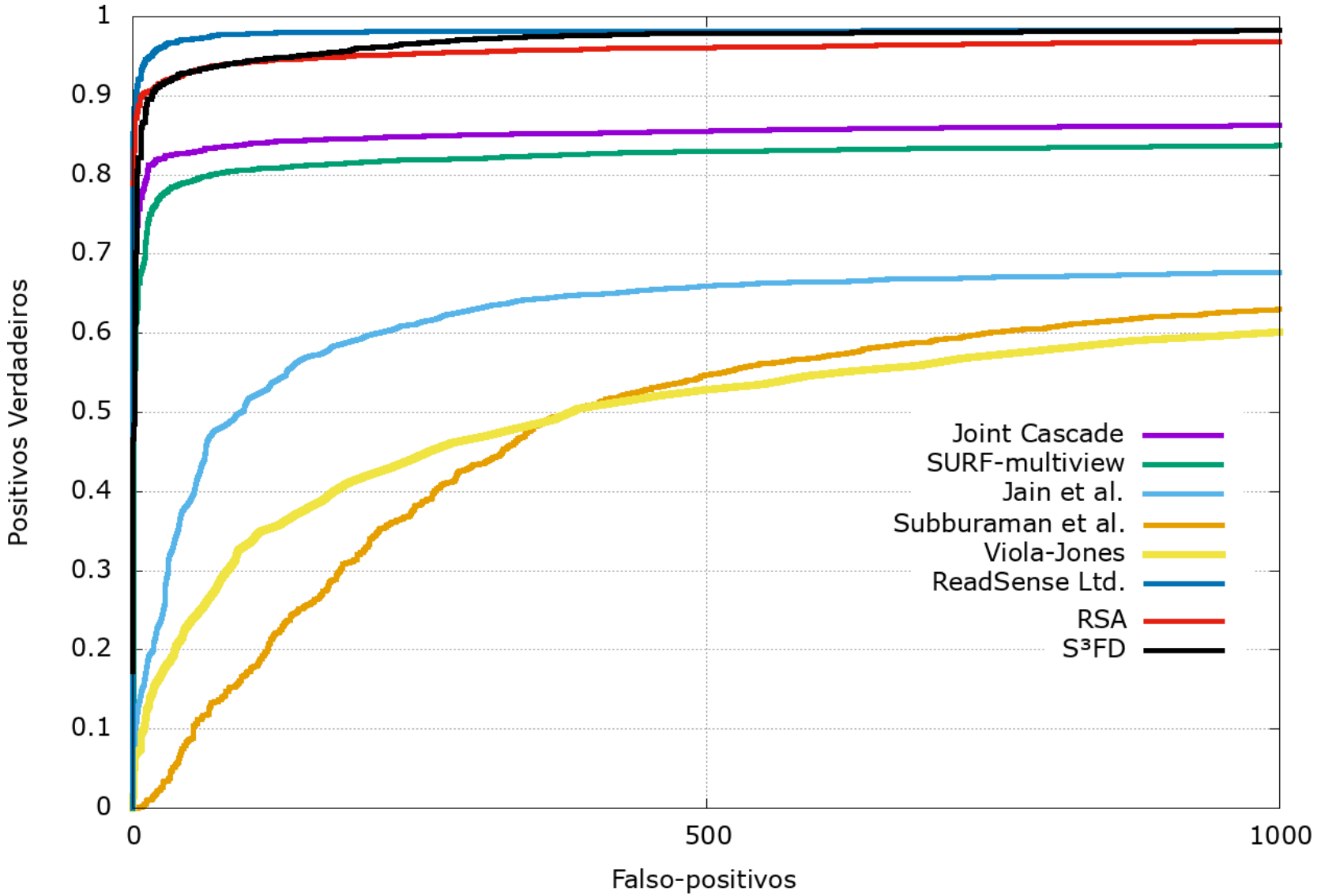


Estágio 4



Classificadores em Cascata

FDDB Benchmark



OpenCV

Treinamento do classificador em cascata usando `opencv_traincascade`:

- 10 estágios
- 5000 imagens positivas
- 3000 imagens negativas
- 36×36 pixels
- 99,5% de detecção
- No máximo 50% de falso-positivos por estágio
- Concluído em 1 dia 7 horas e 42 minutos

OpenCV

```
import cv2

classificador = cv2.CascadeClassifier('haarcascade_frontalface.xml')
camera = cv2.VideoCapture(0)

while camera.isOpened():
    _, imagem = camera.read()

    cinza = cv2.cvtColor(imagem, cv2.COLOR_BGR2GRAY)

    faces = classificador.detectMultiScale(
        cinza, scaleFactor=1.2, minNeighbors=5, minSize=(30, 30))

    for (x, y, w, h) in faces:
        cv2.rectangle(imagem, (x, y), (x + w, y + h), (0, 0, 255), 2)

    cv2.imshow('Detector de Faces', imagem)
```

OpenCV

Teste com imagens da Labeled Faces in the Wild

Sumário

Introdução

Detecção Facial

Reconhecimento Facial

Implementação do Projeto

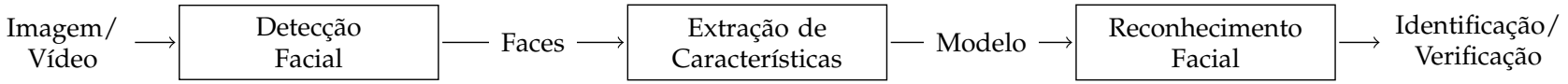
Conclusões

Referências

Definição

É uma tecnologia capaz de realizar:

- **Verificação**: dada uma face e uma alegação de identidade, informa se a face realmente pertence ao indivíduo.
- **Identificação**: compara a imagem de uma face desconhecida com todas as imagens em um banco de dados para determinar sua identidade.



Técnicas

- Eigenfaces (PCA)
- Fisherfaces (LDA)
- Local Binary Patterns (LBP)
- Filtros de Gabor
- GaussianFace (DGP-LVM)
- Redes Neurais Profundas (DNN)

Espaço das imagens de faces

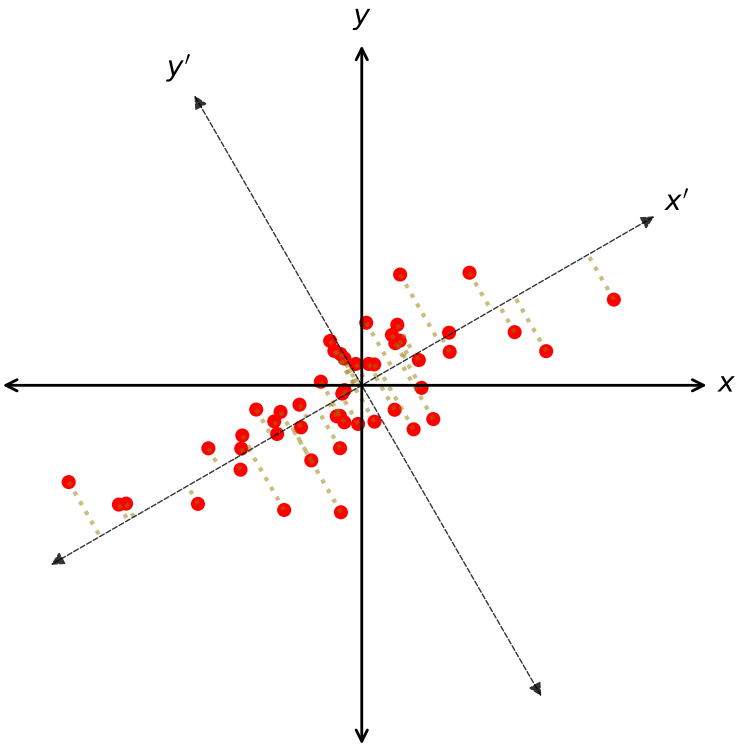
- Uma imagem de 100×100 pixels pode ser representada como um vetor de 10000 dimensões
- Apenas um pequeno grupo desses vetores correspondem a faces
- É possível representar essas faces em um subespaço de dimensão reduzida
- Os principais métodos de redução de dimensionalidade são o PCA e o LDA

Análise de Componentes Principais (PCA)

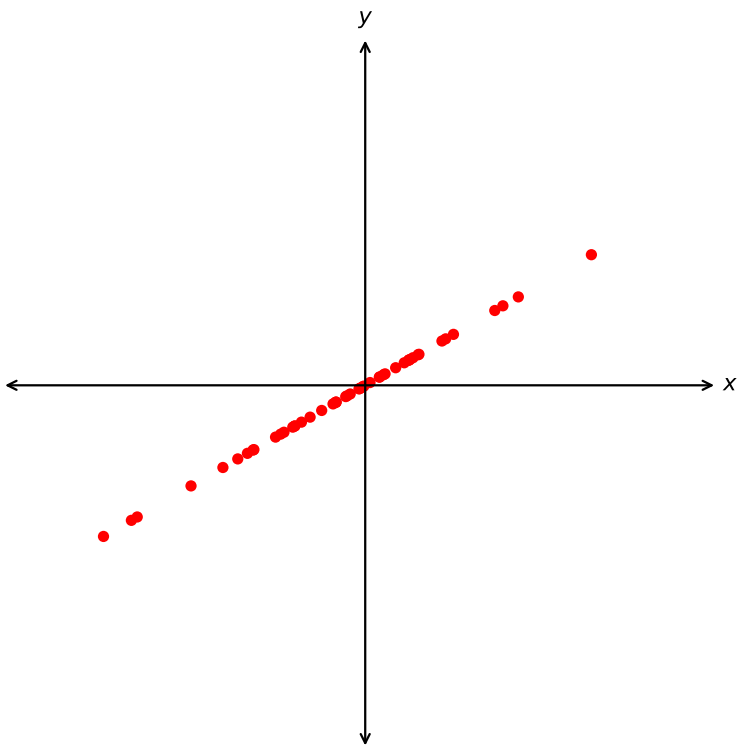
- É uma técnica que converte um conjunto de observações de variáveis possivelmente correlacionadas em um conjunto de valores linearmente não correlacionados, chamados de componentes principais
- O primeiro componente principal é o que possui a maior variância
- Cada componente seguinte possui a máxima variância e é ortogonal aos compoentes anteriores

Análise de Componentes Principais (PCA)

Distribuição original

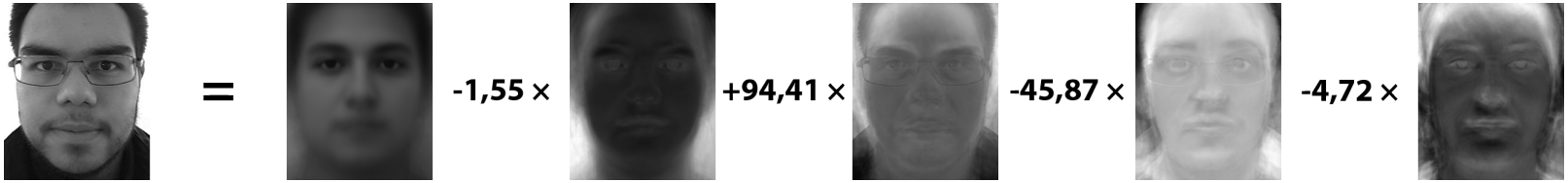


Apenas 1ºcomponente principal



EIGENFACES

- Método desenvolvido por Turk e Pentland (1991)
- Eigenfaces são os componentes principais de uma distribuição de faces
- Cada face pode ser representada como uma combinação linear de eigenfaces



EIGENFACES

Algoritmo:

1. Obter M imagens $I_1, I_2, I_3 \ldots I_M$ com dimensão $N \times N$. As imagens devem estar centralizadas.
2. Representar cada imagem I_i como um vetor Γ_i de dimensão $N^2 \times 1$.

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1N} \\ a_{21} & a_{22} & \cdots & a_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ a_{N1} & a_{N1} & \cdots & a_{NN} \end{bmatrix}_{N \times N} \rightarrow \begin{bmatrix} a_{11} \\ \vdots \\ a_{1N} \\ \vdots \\ a_{2N} \\ \vdots \\ a_{NN} \end{bmatrix}_{N^2 \times 1}$$

EIGENFACES

3. Calcular o vetor Ψ correspondente à face média.

$$\Psi = \frac{1}{M} \sum_{i=1}^{M} \Gamma_i$$

4. Subtrair a face média de cada vetor Γ_i para obter o conjunto de vetores Φ_i.

$$\Phi_i = \Gamma_i - \Psi$$

5. Encontrar a matriz C de covariância.

$$C = AA^T, \text{ onde } A = \begin{bmatrix} \Phi_1 & \Phi_2 & \dots & \Phi_3 \end{bmatrix}$$

C é uma matriz $N^2 \times N^2$ e A é uma matriz $N^2 \times M$.

EIGENFACES

6. Calcular os autovetores u_i de $C = AA^T$.
   Em vez desse cálculo, que resultaria em N^2 autovetores, calcular os autovetores v_i da matriz A^TA de dimensão $M \times M$.

$$\begin{aligned} A^TAv_i &= \mu_i v_i \\ \Rightarrow \quad AA^TAv_i &= \mu_i Av_i \\ \Rightarrow \quad CAv_i &= \mu_i Av_i \\ \Rightarrow \quad u_i &= Av_i \end{aligned}$$

   ou seja, Av_i são autovetores de $C = AA^T$. Os M autovalores de A^TA correspondem aos M maiores autovalores de AA^T.

7. Manter apenas os K autovetores, correspondentes aos K maiores autovalores.

OpenCV

```
# Obtém imagens de treino com ids e nomes
imgs, ids, nomes = le_imagens_de_treino(dir_treino)

# Treina o reconhecedor de faces
reconhecedor = cv2.face.EigenFaceRecognizer_create(300)
reconhecedor.train(np.asarray(imgs),np.asarray(ids))

# Realiza o reconhecimento
cinza = cv2.imread(foto, cv2.IMREAD_GRAYSCALE)
num, confianca = reconhecedor.predict(cinza)
nome = nomes[num]
```

EIGENFACES

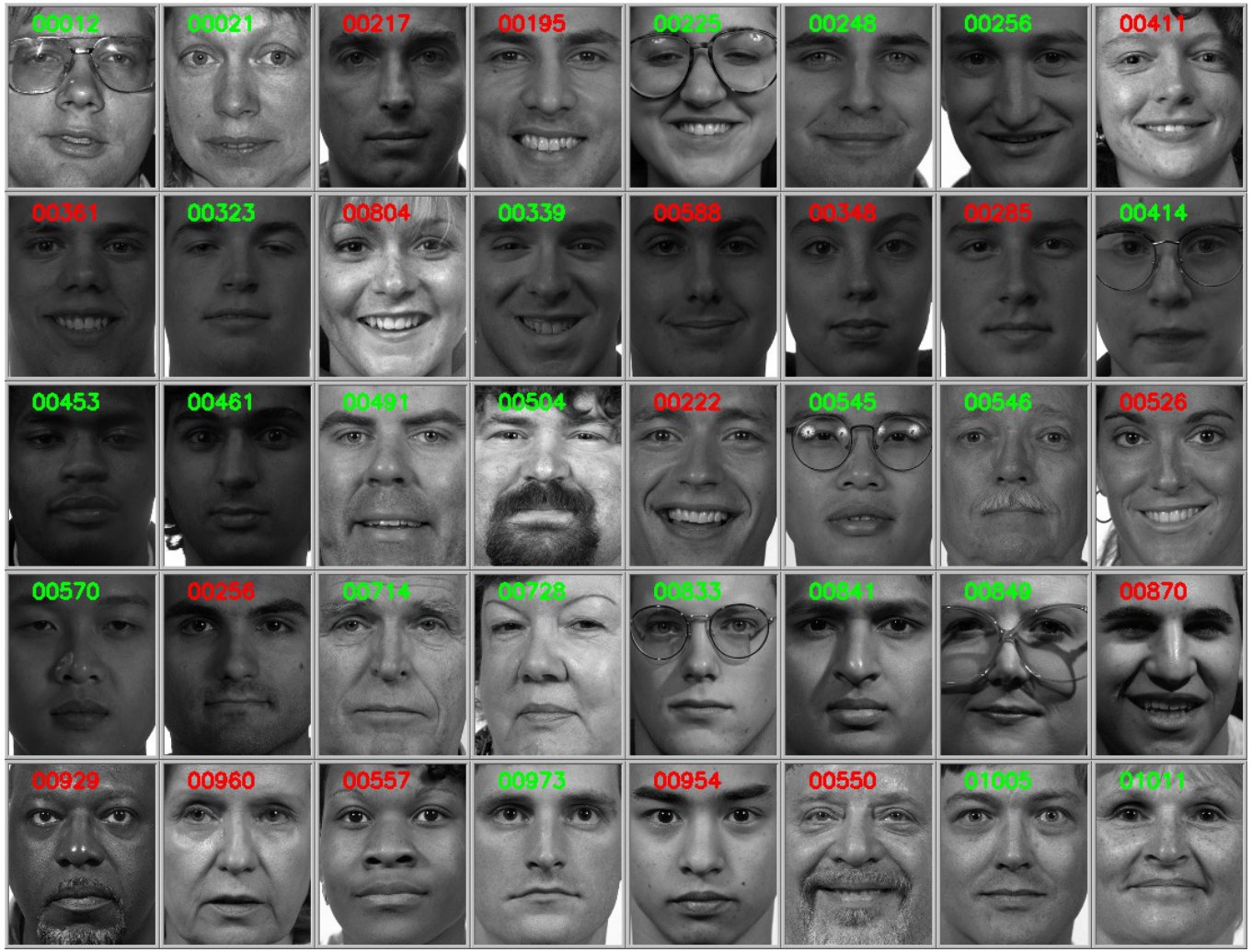
FEI Face Database

Face média

Eigenfaces

EIGENFACES

Teste com o banco de faces FERET



EIGENFACES

Tabela: Bancos de faces utilizados para testes

Banco de faces	**Indivíduos**	**Fotos de treino**
AT&T	40	360
FERET	865	1542
Extended Yale	38	1862

EIGENFACES

Tabela: Comparação dos algoritmos de reconhecimento facial

Método	Banco de faces	Taxa de acertos
Eigenfaces	AT&T	95,0%
	FERET	63,4%
	Extended Yale	81,6%
Fisherfaces	AT&T	97,5%
	FERET	-
	Extended Yale	100%
LBPH	AT&T	100%
	FERET	90,7%
	Extended Yale	94,7%

Sumário

Introdução

Detecção Facial

Reconhecimento Facial

Implementação do Projeto

Conclusões

Referências

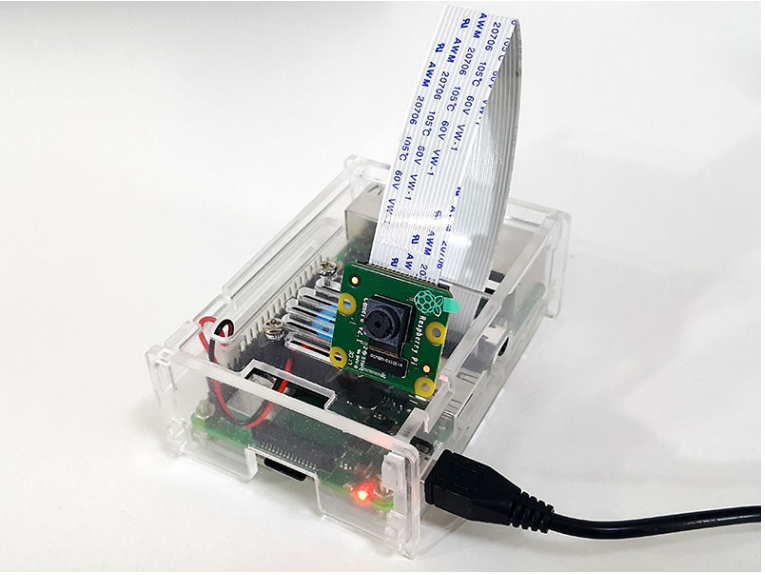
Implementação do Projeto

O projeto é dividido em dois módulos:

- **Módulo 1**: Captura fotos, detecta faces e envia as faces para o segundo módulo
- **Módulo 2**: Verifica se a mesma face já foi analisada no passado e extrai características como idade, emoções e gênero

Módulo 1: Raspberry Pi

- É um computador de placa única de baixo custo
- Processador quad-core 64-bit de 1,4 GHz
- 1 GB de memória SDRAM
- Interface CSI para câmera
- Wifi e Ethernet
- Roda Linux ARM



Módulo 1: Raspberry Pi

- Será instalado em pontos estratégicos da loja
- Realiza detecção utilizando o algoritmo de Viola-Jones implementado na OpenCV
- Envia apenas as imagens das faces para o segundo módulo para reduzir o tráfego de dados e as chamadas para a API

Módulo 2: Microsoft Face API

- Parte do Microsoft Azure Cognitive Services
- Realiza detecção, reconhecimento, agrupamento facial e extrai diversas outras características como emoção, idade, sexo, cor do cabelo, barba, maquiagem e uso de acessórios
- Grátis até 30000 transações por mês
- Aproximadamente R$ 3 a cada 1000 transações

Módulo 2: Microsoft Face API

1. IPython: Users/julio (python3.6)

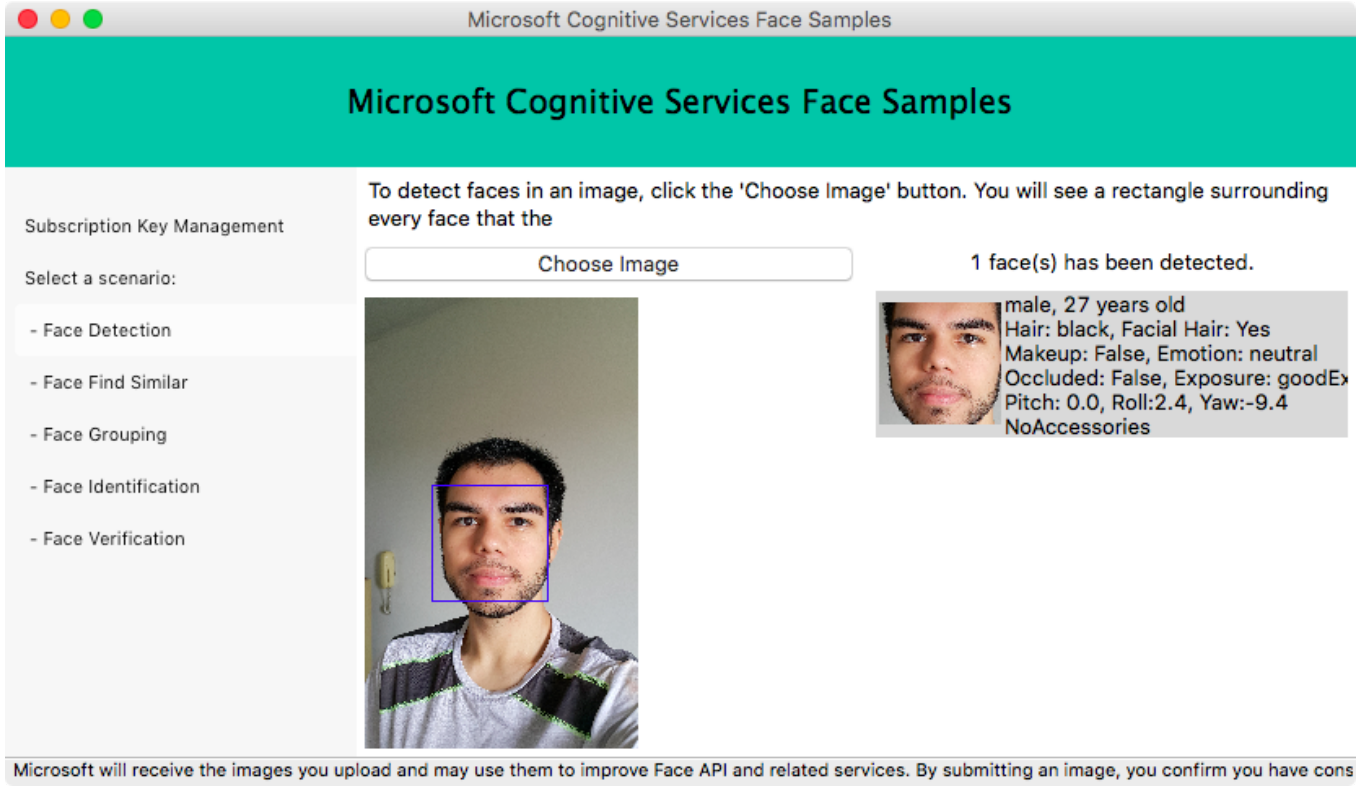
```
~ ❯❯❯ ipython
Python 3.6.5 |Anaconda, Inc.| (default, Apr 26 2018, 08:42:37)
Type 'copyright', 'credits' or 'license' for more information
IPython 6.4.0 -- An enhanced Interactive Python. Type '?' for help.

In [1]: import urllib3
   ...: import json
   ...: import cognitive_face as CF
   ...: urllib3.disable_warnings()
   ...: CF.Key.set('[illegible]')
   ...: CF.BaseUrl.set('https://brazilsouth.api.cognitive.microsoft.com/face/v1.0')
   ...:
   ...: imagem = 'https://juliobs.com/img/profile.jpg'
   ...: atributos = 'age,gender,emotion'
   ...: resultado = CF.face.detect(imagem, False, False, atributos)
   ...: print(json.dumps(resultado, indent=4))
   ...:
   ...:
```

Microsoft Face API

```
[
    {
        "faceRectangle": {
            "top": 110,
            "left": 68,
            "width": 237,
            "height": 237
        },
        "faceAttributes": {
            "gender": "male",
            "age": 28.0,
            "emotion": {
                "anger": 0.0,
                "contempt": 0.0,
                "disgust": 0.0,
                "fear": 0.0,
                "happiness": 0.449,
                "neutral": 0.55,
                "sadness": 0.0,
                "surprise": 0.0
            }
        }
    }
]
```

Microsoft Face API

Python SDK para a Microsoft Face API



Sumário

Introdução

Detecção Facial

Reconhecimento Facial

Implementação do Projeto

Conclusões

Referências

Conclusões

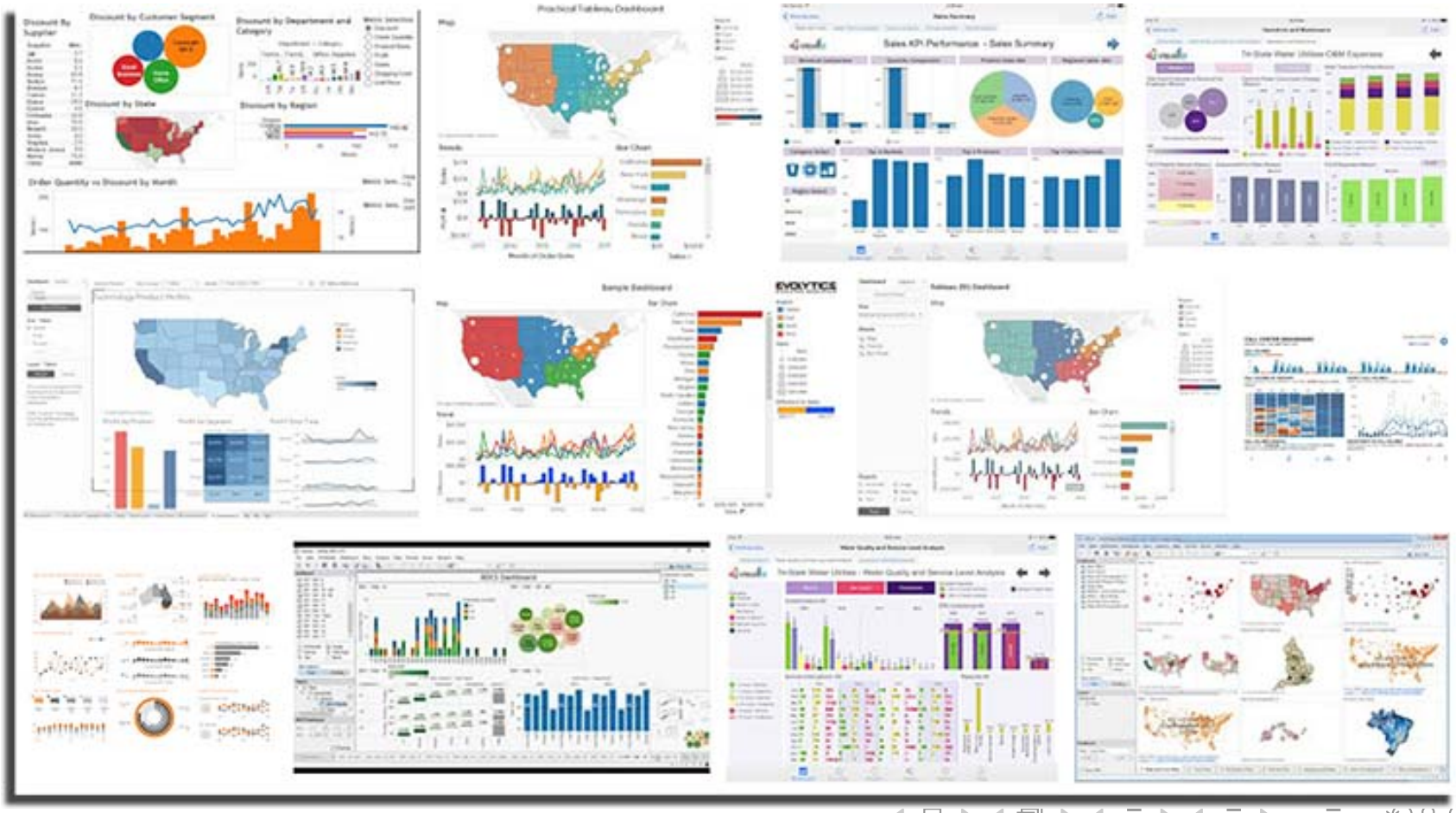
- O algoritmo de Viola-Jones é eficiente para detectar faces
- A biblioteca OpenCV possui diversos algoritmos implementados e facilita o desenvolvimento de aplicações
- A taxa de acertos do Eigenfaces é baixa quando comparada a outros métodos
- Reconhecimento facial irrestrito ainda é um desafio por fatores como iluminação, baixa resolução, variação de pose, oclusão e expressão facial
- Soluções comerciais, como a Face API da Microsoft, oferecem o estado da arte em diversas tarefas como detecção facial, reconhecimento facial, detecção de emoções e identificação de gênero
- É possível utilizar o Raspberry Pi como uma solução barata de câmera inteligente

Trabalhos Futuros

- Treinar um classificador com menos falso-positivos
- Estudar novos métodos de reconhecimento facial e extração de características para substituir a Face API da Microsoft

Trabalhos Futuros

- Criar um dashboard com gráficos e tabelas que facilitem a tomada de decisões utilizando as informações coletadas



Sumário

Introdução

Detecção Facial

Reconhecimento Facial

Implementação do Projeto

Conclusões

Referências

Referências I

CEN, K. Study of viola-jones real time face detector. *Stanford University*, 2016.

LANITIS, A.; TAYLOR, C. J.; COOTES, T. F. Automatic face identification system using flexible appearance models. *Image and vision computing*, Elsevier, v. 13, n. 5, p. 393–401, 1995.

LEUNG, T. K.; BURL, M. C.; PERONA, P. Finding faces in cluttered scenes using random labeled graph matching. In: IEEE. *Computer Vision, 1995. Proceedings., Fifth International Conference on*. [S.l.], 1995. p. 637–644.

OMAIA, D. e. a. *Um sistema para detecção e reconhecimento de face em vídeo utilizando a transformada cosseno discreta*. Dissertação (Mestrado) — Universidade Federal da Paraíba, 2009.

Referências II

RSCG, S. E. *The Nikon S60. Detects up to 12 faces.* 2008. Disponível em: <http://www.adsavvy.org/great-new-nikon-ads-facial-detection/>. Acesso em: 27 jun. 2018.

TURK, M.; PENTLAND, A. Eigenfaces for recognition. *Journal of cognitive neuroscience*, MIT Press, v. 3, n. 1, p. 71–86, 1991.

VIOLA, P.; JONES, M. Rapid object detection using a boosted cascade of simple features. *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, v. 1, p. I–I, 2001.

YANG, G.; HUANG, T. S. Human face detection in a complex background. *Pattern recognition*, Elsevier, v. 27, n. 1, p. 53–63, 1994.