Visão Computacional: O Guia Completo - Detecção de Faces

OpenCV

Carregamento da imagem

```
import cv2 # OpenCV
```

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

```
imagem = cv2.imread('/content/drive/MyDrive/Cursos - recursos/Visão Computacional Guia Completo/Images/people1.jpg')
```

```
imagem.shape
```

```
(1280, 1920, 3)
```

```
#cv2.imshow(imagem)
from google.colab.patches import cv2_imshow
cv2_imshow(imagem)
```

```
imagem = cv2.resize(imagem, (800, 600))
imagem.shape
```

```
(600, 800, 3)
```

```
cv2_imshow(imagem)
```

```
600 * 800 * 3, 600 * 800
```

```
(1440000, 480000)
```

```
imagem_cinza = cv2.cvtColor(imagem, cv2.COLOR_BGR2GRAY)
cv2_imshow(imagem_cinza)
```

```
imagem_cinza.shape
```

```
(600, 800)
```

Detecção de faces

```
detector_facial = cv2.CascadeClassifier('/content/drive/MyDrive/Cursos - recursos/Visão Computacional Guia Completo/Cascades/haarcascade_
```

```
deteccoes = detector_facial.detectMultiScale(imagem_cinza)
```

```
deteccoes
```

```
array([[677,  72,  68,  68],
       [115, 124,  53,  53],
       [475, 123,  59,  59],
       [387, 233,  73,  73],
       [ 92, 239,  66,  66],
       [390, 323,  56,  56]], dtype=int32)
```

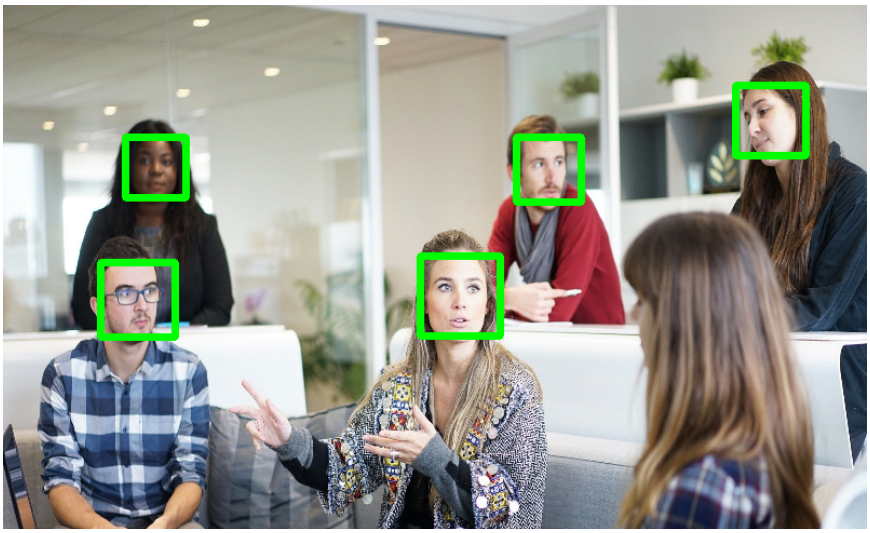
```
len(deteccoes)
```

```
6
```

```
for x, y, w, h in deteccoes:
  #print(x, y, w, h)
  cv2.rectangle(imagem, (x, y), (x + w, y + h), (0,255,255), 5)
cv2_imshow(imagem)
```

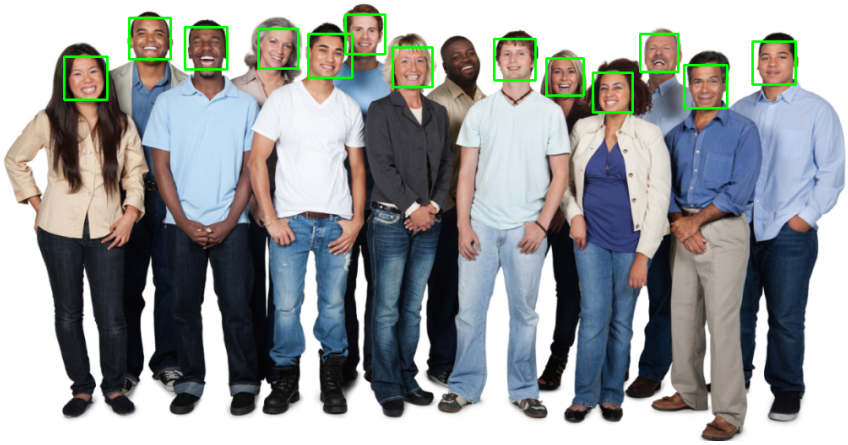
Parâmetros haarcascades

```
imagem = cv2.imread('/content/drive/MyDrive/Cursos - recursos/Visão Computacional Guia Completo/Images/people1.jpg')
imagem = cv2.resize(imagem, (800, 600))
imagem_cinza = cv2.cvtColor(imagem, cv2.COLOR_BGR2GRAY)
deteccoes = detector_facial.detectMultiScale(imagem_cinza, scaleFactor=1.09)
for (x, y, w, h) in deteccoes:
  cv2.rectangle(imagem, (x, y), (x + w, y + h), (0,255,0), 5)
cv2_imshow(imagem)
```



```
imagem = cv2.imread('/content/drive/MyDrive/Cursos - recursos/Visão Computacional Guia Completo/Images/people2.jpg')
imagem_cinza = cv2.cvtColor(imagem, cv2.COLOR_BGR2GRAY)
deteccoes = detector_facial.detectMultiScale(imagem_cinza, scaleFactor=1.2, minNeighbors=3,
                                             minSize=(32,32), maxSize=(100,100))
for (x, y, w, h) in deteccoes:
  print(w, h)
  cv2.rectangle(imagem, (x, y), (x + w, y + h), (0,255,0), 2)
cv2_imshow(imagem)
```

```
47 47
47 47
45 45
49 49
49 49
44 44
51 51
52 52
50 50
51 51
47 47
51 51
47 47
```



Detecção de olhos

```
detector_olhos = cv2.CascadeClassifier('/content/drive/MyDrive/Cursos - recursos/Visão Computacional Guia Completo/Cascades/haarcascade_e
```

```
imagem = cv2.imread('/content/drive/MyDrive/Cursos - recursos/Visão Computacional Guia Completo/Images/people1.jpg')
#imagem = cv2.resize(imagem, (800, 600))
imagem_cinza = cv2.cvtColor(imagem, cv2.COLOR_BGR2GRAY)
```

```
deteccoes = detector_facial.detectMultiScale(imagem_cinza, scaleFactor = 1.3, minSize = (30,30))
for (x, y, w, h) in deteccoes:
  cv2.rectangle(imagem, (x, y), (x + w, y + h), (0,255,0), 2)

deteccoes_olhos = detector_olhos.detectMultiScale(imagem_cinza, scaleFactor=1.09, minNeighbors=10, maxSize=(70,70))
for (x, y, w, h) in deteccoes_olhos:
  #print(w,h)
  cv2.rectangle(imagem, (x, y), (x + w, y + h), (0,0,255), 2)

cv2_imshow(imagem)
```

Outros objetos

Carros

```
detector_carros = cv2.CascadeClassifier('/content/drive/MyDrive/Cursos - recursos/Visão Computacional Guia Completo/Cascades/cars.xml')
imagem = cv2.imread('/content/drive/MyDrive/Cursos - recursos/Visão Computacional Guia Completo/Images/car.jpg')
imagem_cinza = cv2.cvtColor(imagem, cv2.COLOR_BGR2GRAY)
deteccoes = detector_carros.detectMultiScale(imagem_cinza, scaleFactor = 1.03, minNeighbors=5)
for (x, y, w, h) in deteccoes:
  cv2.rectangle(imagem, (x, y), (x + w, y + h), (0,255,0), 2)
cv2_imshow(imagem)
```

Relógios

```
detector_relogios = cv2.CascadeClassifier('/content/drive/MyDrive/Cursos - recursos/Visão Computacional Guia Completo/Cascades/clocks.xml
imagem = cv2.imread('/content/drive/MyDrive/Cursos - recursos/Visão Computacional Guia Completo/Images/clock.jpg')
imagem_cinza = cv2.cvtColor(imagem, cv2.COLOR_BGR2GRAY)
deteccoes = detector_relogios.detectMultiScale(imagem_cinza, scaleFactor = 1.03, minNeighbors=1)
for (x, y, w, h) in deteccoes:
  cv2.rectangle(imagem, (x, y), (x + w, y + h), (0,255,0), 2)
cv2_imshow(imagem)
```

Corpo inteiro

```
detector_corpo = cv2.CascadeClassifier('/content/drive/MyDrive/Cursos - recursos/Visão Computacional Guia Completo/Cascades/fullbody.xml'
imagem = cv2.imread('/content/drive/MyDrive/Cursos - recursos/Visão Computacional Guia Completo/Images/people3.jpg')
imagem_cinza = cv2.cvtColor(imagem, cv2.COLOR_BGR2GRAY)
deteccoes = detector_corpo.detectMultiScale(imagem_cinza, scaleFactor = 1.05, minNeighbors=5,
                                            minSize = (50,50))
for (x, y, w, h) in deteccoes:
  cv2.rectangle(imagem, (x, y), (x + w, y + h), (0,255,0), 2)
cv2_imshow(imagem)
```

Dlib

Obs: execute a linha abaixo somente caso ocorra o erro "code: 9, reason: CUDNN_STATUS_NOT_SUPPORTED" (lembre-se de reiniciar o ambiente de execução caso já tenha importado o Dlib). Isso fará com que seja instalada uma versão anterior àquela que começou a dar o problema de conflito com a GPU do Colab.

```
!pip install dlib==19.22.1
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Collecting dlib==19.22.1
  Downloading dlib-19.22.1.tar.gz (7.4 MB)
     |████████████████████████████████| 7.4 MB 4.8 MB/s
Building wheels for collected packages: dlib
  Building wheel for dlib (setup.py) ... done
  Created wheel for dlib: filename=dlib-19.22.1-cp37-cp37m-linux_x86_64.whl size=4202369 sha256=b5442c23d4db250998dab6e6567016e390b
  Stored in directory: /root/.cache/pip/wheels/10/6f/db/ef0380a66d955f9caa81cff6027fb77c0b556568e0daace292
Successfully built dlib
Installing collected packages: dlib
  Attempting uninstall: dlib
    Found existing installation: dlib 19.24.0
    Uninstalling dlib-19.24.0:
      Successfully uninstalled dlib-19.24.0
Successfully installed dlib-19.22.1
```

```
import dlib
```

```
dlib.__version__
```

```
'19.22.1'
```

Detecção de faces com HOG

```
imagem = cv2.imread('/content/drive/MyDrive/Cursos - recursos/Visão Computacional Guia Completo/Images/people2.jpg')
cv2_imshow(imagem)
```

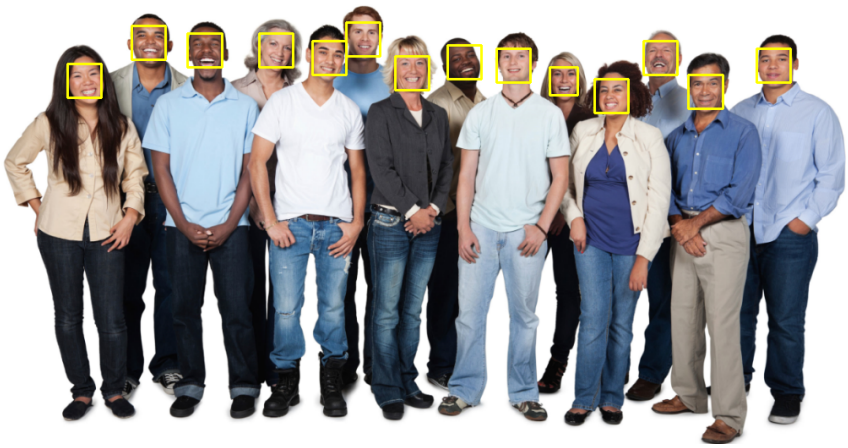
```
detector_face_hog = dlib.get_frontal_face_detector()
```

```
deteccoes = detector_face_hog(imagem, 4) # escala
```

```
deteccoes, len(deteccoes)
```

```
(rectangles[[(425, 38) (465, 78)], [(385, 60) (425, 100)], [(719, 105) (759, 145)], [(911, 69) (951, 109)], [(830, 100) (871,
140)], [(666, 91) (699, 124)], [(483, 78) (523, 118)], [(603, 69) (643, 109)], [(777, 60) (817, 100)], [(95, 87) (135, 127)],
[(171, 42) (211, 82)], [(237, 51) (278, 91)], [(322, 51) (362, 91)], [(545, 65) (585, 105)]],
 14)
```

```
for face in deteccoes:
  #print(face)
  #print(face.left())
  #print(face.top())
  #print(face.right())
  #print(face.bottom())
  l, t, r, b = face.left(), face.top(), face.right(), face.bottom()
  cv2.rectangle(imagem, (l, t), (r, b), (0,255,255), 2)
cv2_imshow(imagem)
```



Detecção de faces com CNN (redes neurais convolucionais)

```
imagem = cv2.imread('/content/drive/MyDrive/Cursos - recursos/Visão Computacional Guia Completo/Images/people2.jpg')
detector_face_cnn = dlib.cnn_face_detection_model_v1('/content/drive/MyDrive/Cursos - recursos/Visão Computacional Guia Completo/Weights/
```

```
deteccoes = detector_face_cnn(imagem, 1)
for face in deteccoes:
  l, t, r, b, c = face.rect.left(), face.rect.top(), face.rect.right(), face.rect.bottom(), face.confidence
  print(c)
  cv2.rectangle(imagem, (l,t), (r,b), (255,255,0), 2)
cv2_imshow(imagem)
```

```
1.1440614461898804
1.137049913406372
1.1278995275497437
1.1200283765792847
1.1149375438690186
1.1131553649902344
1.0975688695907593
1.0942121744155884
1.085315227508545
1.0801889896392822
1.0800762176513672
1.0784764289855957
1.066403865814209
1.0641791820526123
```

Haarcascade x HOG x CNN

Haarcascade


```
imagem = cv2.imread('/content/drive/MyDrive/Cursos - recursos/Visão Computacional Guia Completo/Images/people3.jpg')
imagem_cinza = cv2.cvtColor(imagem, cv2.COLOR_BGR2GRAY)
detector_haarcascade = cv2.CascadeClassifier('/content/drive/MyDrive/Cursos - recursos/Visão Computacional Guia Completo/Cascades/haarcas
deteccoes = detector_haarcascade.detectMultiScale(imagem_cinza, scaleFactor = 1.001, minNeighbors=5, minSize = (5,5))
for (x, y, w, h) in deteccoes:
  cv2.rectangle(imagem, (x, y), (x + w, y + h), (0,255,0), 2)
cv2_imshow(imagem)
```

HOG

```
imagem = cv2.imread('/content/drive/MyDrive/Cursos - recursos/Visão Computacional Guia Completo/Images/people3.jpg')
detector_hog = dlib.get_frontal_face_detector()
deteccoes = detector_hog(imagem, 4)
for face in deteccoes:
    l, t, r, b = (face.left(), face.top(), face.right(), face.bottom())
    cv2.rectangle(imagem, (l, t), (r, b), (0, 255, 255), 2)
cv2_imshow(imagem)
```

CNN

```
imagem = cv2.imread('/content/drive/MyDrive/Cursos - recursos/Visão Computacional Guia Completo/Images/people3.jpg')
detector_cnn = dlib.cnn_face_detection_model_v1('/content/drive/MyDrive/Cursos - recursos/Visão Computacional Guia Completo/Weights/mmod_
deteccoes = detector_cnn(imagem, 4)
```

```
for face in deteccoes:
  l, t, r, b, c = face.rect.left(), face.rect.top(), face.rect.right(), face.rect.bottom(), face.confidence
  print(c)
  cv2.rectangle(imagem, (l, t), (r, b), (255, 255, 0), 2)
cv2_imshow(imagem)
```

```
0.4463871121406555
0.08115974068641663
```