

## ▼ Reconhecimento de pontos do rosto

```
1 from google.colab import drive  
2 drive.mount("/content/drive")
```

Mounted at /content/drive

```
1 import cv2  
2 import numpy as np  
3 import dlib  
4 import matplotlib.pyplot as plt
```

```
1 %cd '/content/drive/MyDrive/tcc/'
```

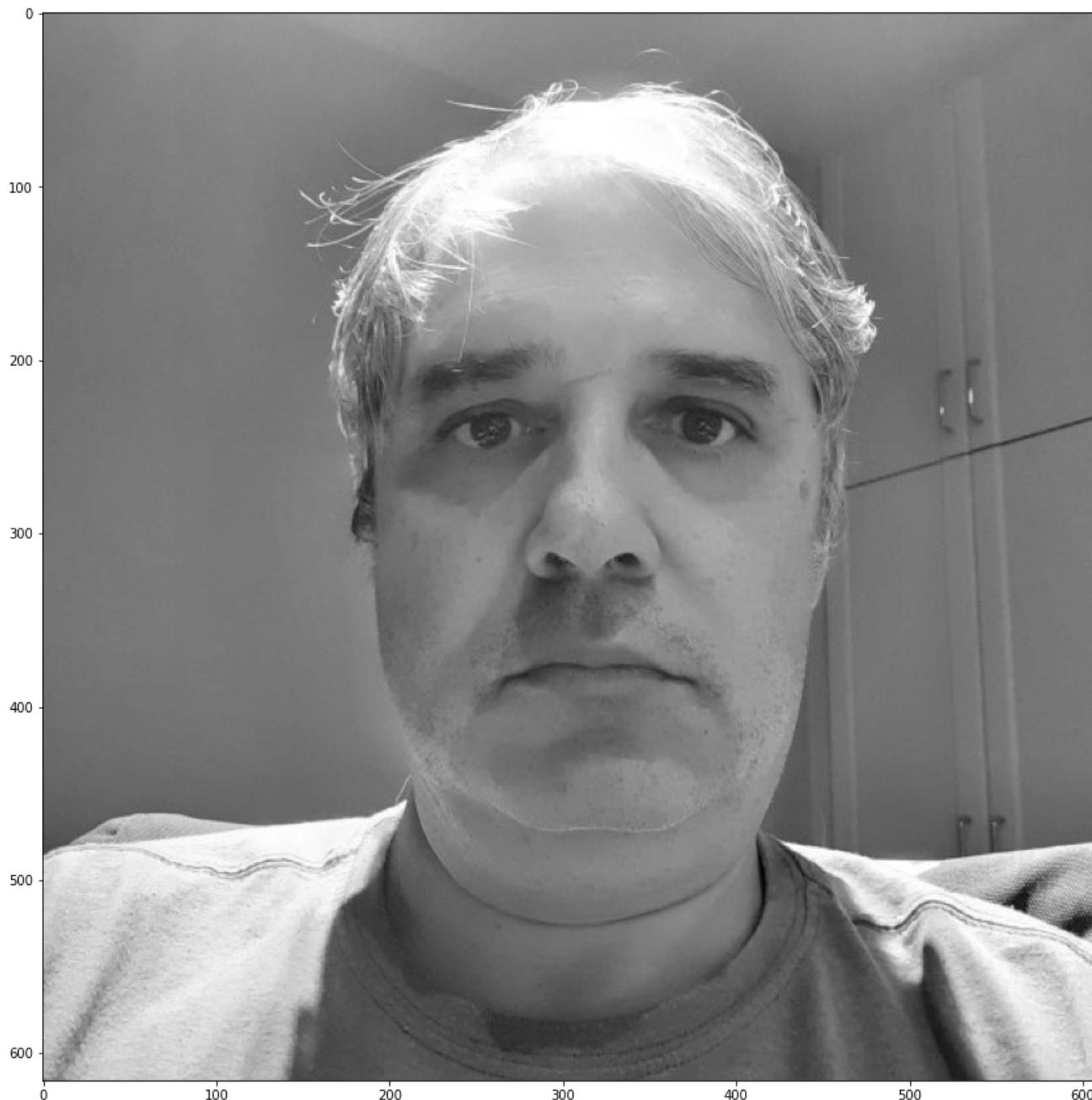
/content/drive/MyDrive/tcc

```
1 imagem = cv2.imread("FotosPauloAlmeida/rostoAtual.jpg", cv2.IMREAD_GRAYSCALE)
```

+ Código + Texto

```
1 plt.figure(figsize=(15,15))  
2 plt.imshow(imagem, cmap="gray")
```

```
<matplotlib.image.AxesImage at 0x7f5fa5a311d0>
```



```
1 !wget http://dlib.net/files/shape_predictor_68_face_landmarks.dat.bz2
```

<https://colab.research.google.com/drive/18YyEZtQGpqFY-mpieC4ZaJ9GAPGS3sBw#scrollTo=uX9Y1UCi8mXc&printMode=true>

```
2 !bunzip2 /content/drive/MyDrive/tcc/shape_predictor_68_face_landmarks.dat.bz2
```

```
--2021-09-22 13:29:27-- http://dlib.net/files/shape\_predictor\_68\_face\_landmarks.dat.bz2
```

```
Resolving dlib.net (dlib.net)... 107.180.26.78
```

```
Connecting to dlib.net (dlib.net)|107.180.26.78|:80... connected.
```

```
HTTP request sent, awaiting response... 200 OK
```

```
Length: 64040097 (61M)
```

```
Saving to: 'shape_predictor_68_face_landmarks.dat.bz2.7'
```

```
shape_predictor_68_ 100%[=====] 61.07M 15.0MB/s in 5.4s
```

```
2021-09-22 13:29:34 (11.4 MB/s) - 'shape_predictor_68_face_landmarks.dat.bz2.7' saved [64040097/64040097]
```

```
bunzip2: Output file /content/drive/MyDrive/tcc/shape_predictor_68_face_landmarks.dat already exists.
```

```
1 # Classificador 68 pontos
2
3 classificador_68_path = "shape_predictor_68_face_landmarks.dat"
4 classificador_dlib = dlib.shape_predictor(classificador_68_path)
5 detector_face = dlib.get_frontal_face_detector()
```

```
1 def anotar_rosto(imagem):
2     retangulos = detector_face(imagem, 1)
3     if len(retangulos) == 0:
4         return None
5     for k, d in enumerate(retangulos):
6         # k - numero rostos encontrados / d - dimensões
7         print(f"Rostos identificados: {str(k)}")
8         cv2.rectangle(imagem, (d.left(), d.top()), (d.right(), d.bottom()), (255,255,0), 2)
9     return imagem
```

```
1 def plota_imagem(imagem):
2     plt.figure(figsize=(15,15))
3     plt.imshow(imagem, cmap="gray")
```

```
1 imagem_anotada = imagem.copy()
```

```
2     imagem_anotada = anotar_rosto(imagem_anotada)
3     plota_imagem(imagem_anotada)
```

Rostos identificados: 0



```
1 def pontos_marcos_faciais(imagem):
2     retangulos = detector_face(imagem, 1)
3     if len(retangulos) == 0:
4         return None
5     marcos = []
6     for ret in retangulos:
7         marcos.append(np.matrix([[p.x, p.y] for p in classificador_dlib(imagem,ret).parts()]))
8     return marcos
```

```
1 marcos_faciais = pontos_marcos_faciais(imagem_anotada)
2 len(marcos_faciais[0])
```

68

```
1 def anotar_marcos_faciais(imagem, marcos):
2     for marco in marcos:
3         for idx, ponto in enumerate(marco):
4             centro = (ponto[0,0], ponto[0,1])
5             cv2.circle(imagem, centro, 3, (0,0,0), -1)
6             cv2.putText(imagem, str(idx), centro, cv2.FONT_HERSHEY_SIMPLEX, 0.8, (255,255,255), 2)
7     return imagem
```

```
1 imagem_anotada = imagem.copy()
```

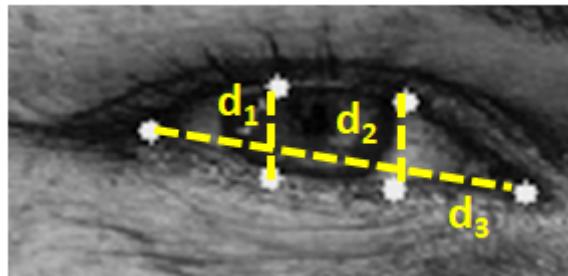
```
2     imagem_anotada = anotar_marcos_taciais(imagem_anotada, marcos_taciais)
3     plota_imagem(imagem_anotada)
```



```
1 # PONTOS DO ROSTO - ASPECTO RAZÃO
2
3 ROSTO = list(range(17,68))
4 ROSTO_COMPLETO = list(range(0,68))
5 LABIOS = list(range(48,61))
6 SOBRANCELHA_DIREITA = list(range(17,22))
7 SOBRANCELHA_ESQUERDA = list(range(22,27))
8 OLHO_ESQUERDO = list(range(36,42))
9 OLHO_DIREITO = list(range(42,48))
10 NARIZ = list(range(27,35))
11 MANDIBULA = list(range(0,17))
```

```
1 from scipy.spatial import distance as dist
```

## ▼ Aspecto Razão



```
1 def aspecto_razao_olhos(pontos_olhos):
2     a = dist.euclidean(pontos_olhos[1],pontos_olhos[5])
3     b = dist.euclidean(pontos_olhos[2],pontos_olhos[4])
4     c = dist.euclidean(pontos_olhos[0],pontos_olhos[3])
5
6     aspecto_razao = (a+b) / (c * 2.0)
7
8     return aspecto_razao
```

```
1 def anota_marcos_casca_convexa(imagem, marcos):
2     retangulos = detector_face(imagem, 1)
3     if len(retangulos) == 0:
4         return None
5
6     for idx, ret in enumerate(retangulos):
7         marco = marcos[idx]
8
9         pontos = cv2.convexHull(marco[OLHO_ESQUERDO])
10        cv2.drawContours(imagem, [pontos],0,(0,255,0),2)
11
12        pontos = cv2.convexHull(marco[OLHO_DIREITO])
13        cv2.drawContours(imagem, [pontos],0,(0,255,0),2)
14
15    return imagem
```

```
1 imagem_anotada = imagem.copy()
2 imagem_anotada = anota_marcos_casca_convexa(imagem, marcos_faciais)
3 print("Contorno dos olhos")
4 plota_imagem(imagem_anotada)
```

Contorno dos olhos





```
1 # distância entre olhos
2 valor_olho_direito = aspecto_razao_olhos(marcos_faciais[0][OLHO_DIREITO])
3 valor_olho_esquerdo = aspecto_razao_olhos(marcos_faciais[0][OLHO_ESQUERDO])
4 print("Aspecto razão dos olhos: \n" \
5      f"Olho esquerdo {valor_olho_esquerdo} \n" \
6      f"Olho direito {valor_olho_direito}")
```

Aspecto razão dos olhos:  
Olho esquerdo 0.3823040321939709  
Olho direito 0.3599280215928025

```
1 def analisa_olhos(imagem):
2     # plota imagem natural
3     print("Imagen")
4     plota_imagem(imagem)
5
6     # anota rosto
7     imagem_anotada = imagem.copy()
8     imagem_anotada = anotar_rosto(imagem_anotada)
9     plota_imagem(imagem_anotada)
10
11    # anota marcos faciais
12    marcos_faciais = pontos_marcos_faciais(imagem_anotada)
13    imagem_anotada = imagem.copy()
14    imagem_anotada = anotar_marcos_faciais(imagem_anotada, marcos_faciais)
15    plota_imagem(imagem_anotada)
16
17    # plota imagem contorno
18    imagem_anotada = imagem.copy()
19    imagem_anotada = anota_marcos_casca_convexa(imagem, marcos_faciais)
20    print("Contorno dos olhos")
21    plota_imagem(imagem_anotada)
22
```

```
23 # informa aspecto razão dos olhos
24 valor_olho_direito = aspecto_razao_olhos(marcos_faciais[0][OLHO_DIREITO])
25 valor_olho_esquerdo = aspecto_razao_olhos(marcos_faciais[0][OLHO_ESQUERDO])
26 print("Aspecto razão dos olhos: \n" \
27       f"Olho esquerdo {valor_olho_esquerdo} \n" \
28       f"Olho direito {valor_olho_direito}")
```

## ▼ Olhos fechados

```
1 olhos_fechados = cv2.imread("FotosPauloAlmeida/olhosFechados.jpg", cv2.IMREAD_GRAYSCALE)
2 analisa_olhos(olhos_fechados)
```

Imagen

Rostos identificados: 0

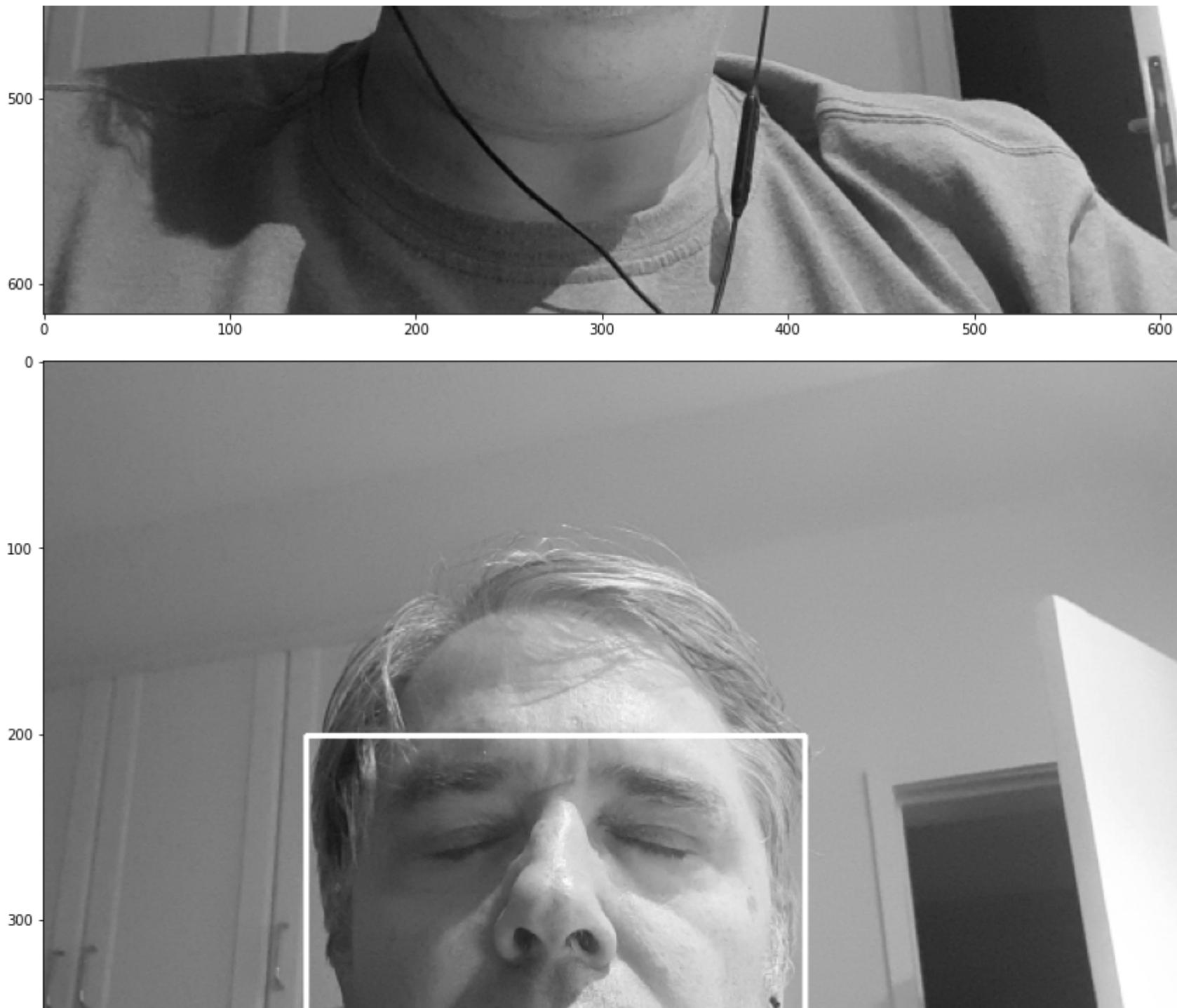
Contorno dos olhos

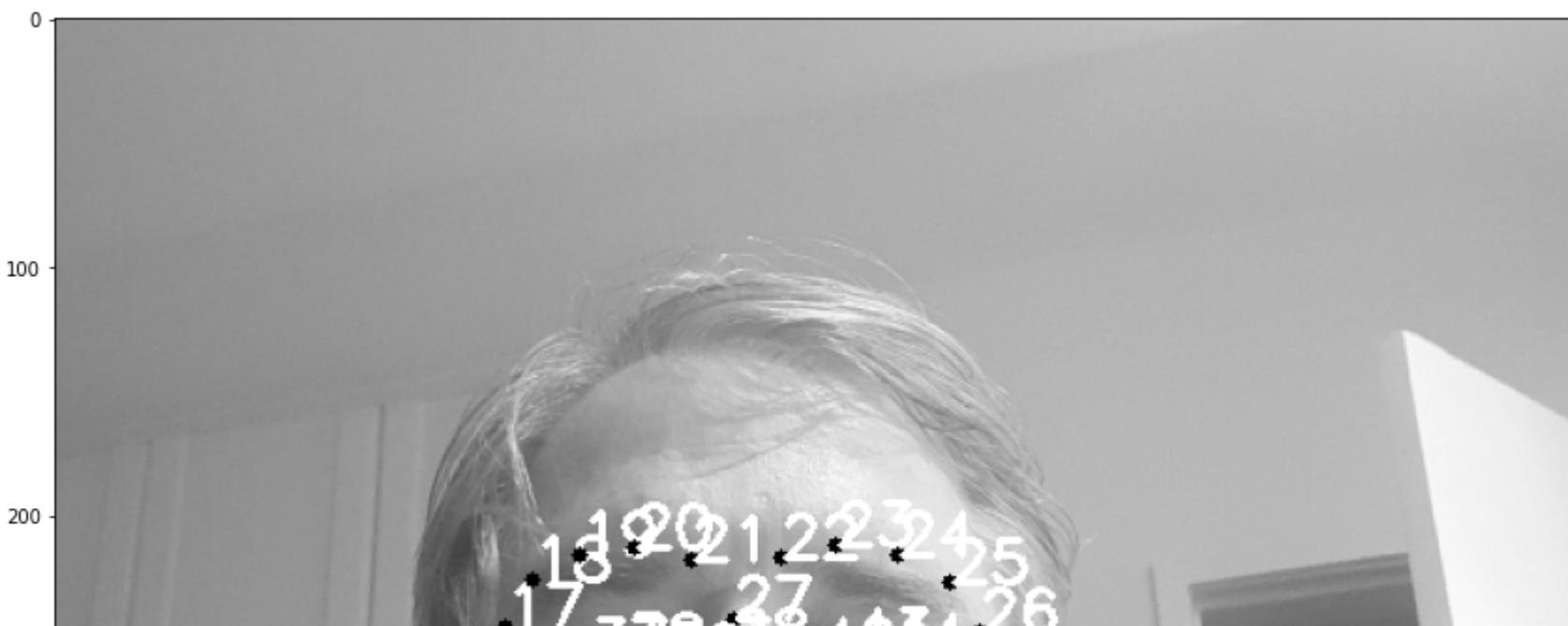
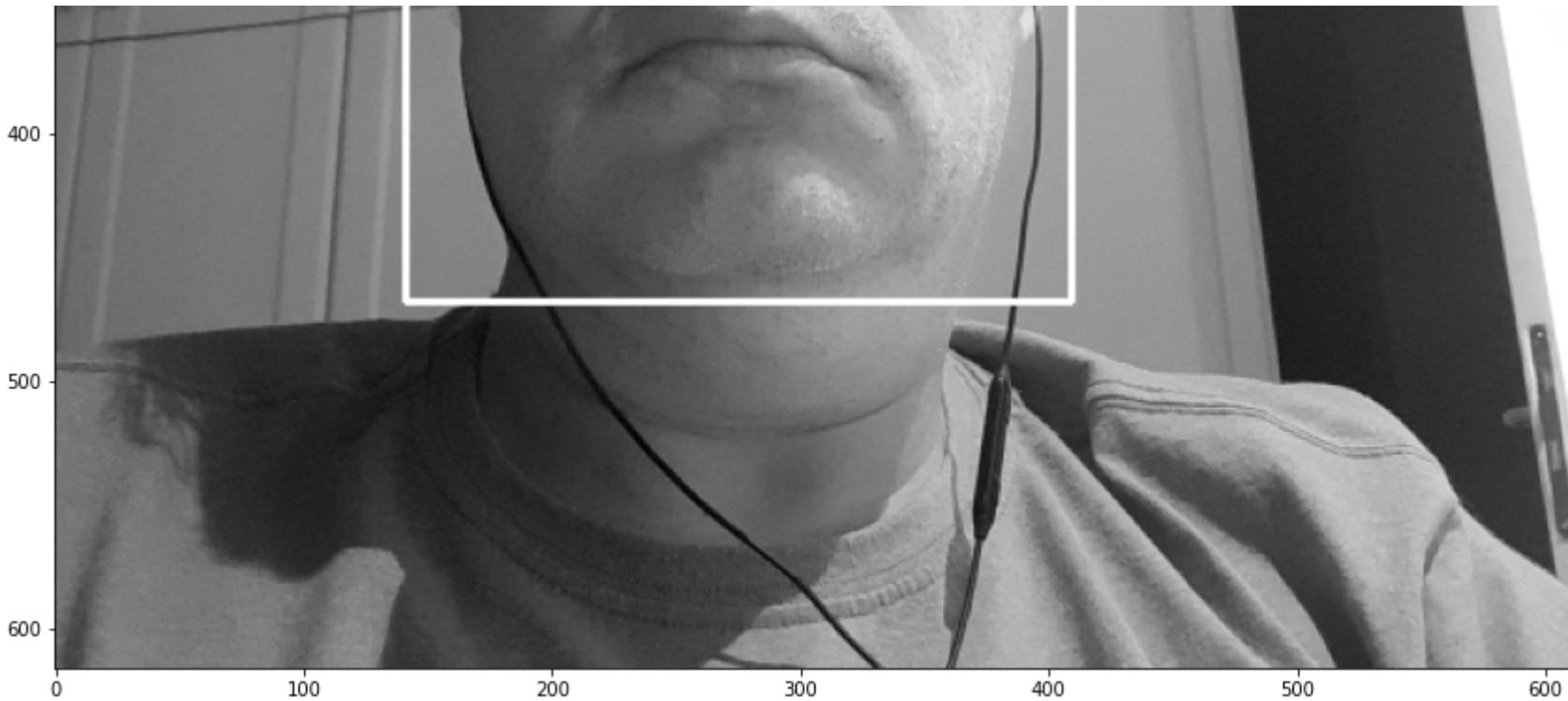
Aspecto razão dos olhos:

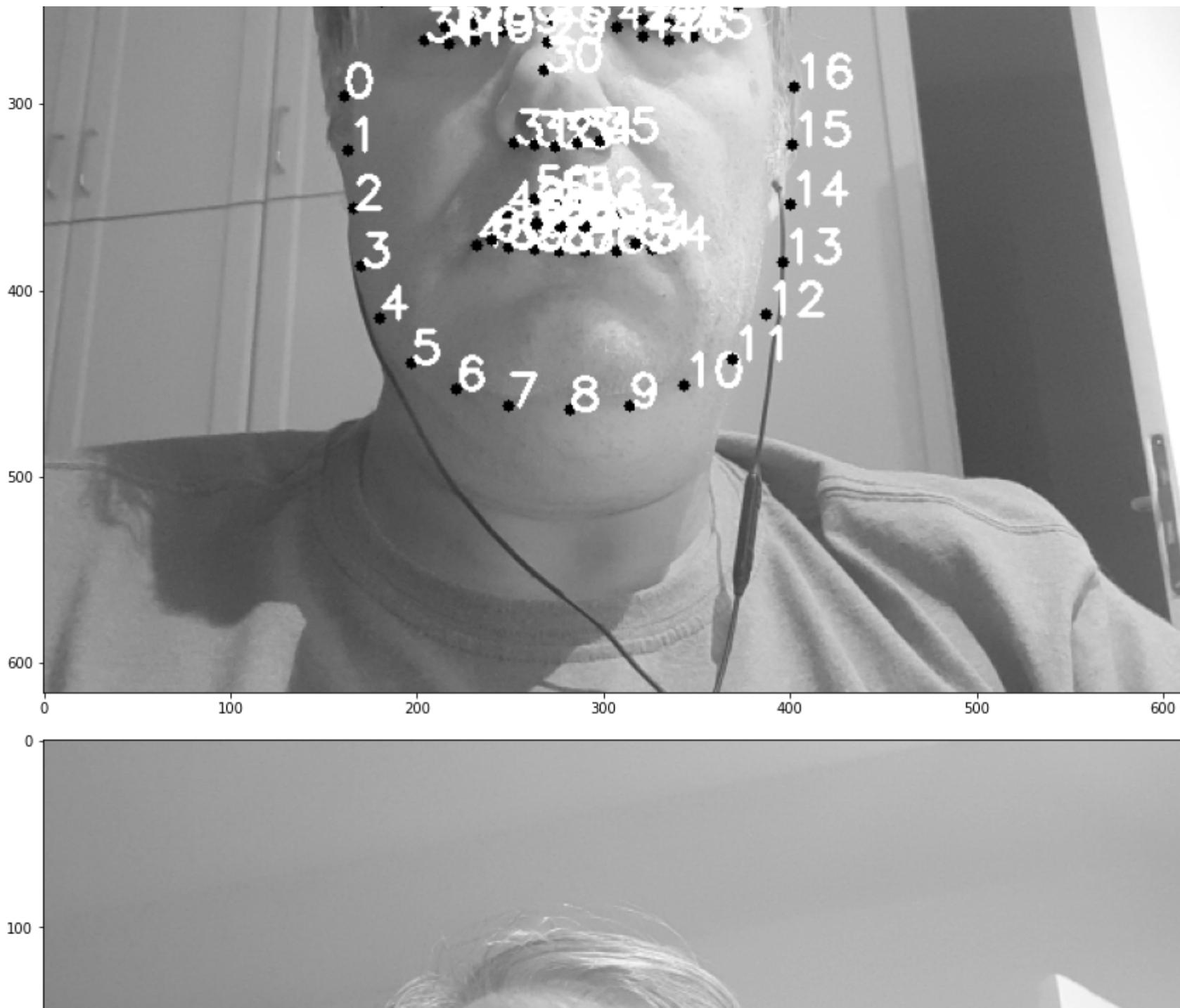
Olho esquerdo 0.21603322380952905

Olho direito 0.21789787555987158











## ▼ Olhos Semicerrados

```
1 olhos_semicerrados = cv2.imread("FotosPauloAlmeida/semicerrados.jpg", cv2.IMREAD_GRAYSCALE)
2 analisa_olhos(olhos_semicerrados)
```

Imagen

Rostos identificados: 0

Contorno dos olhos

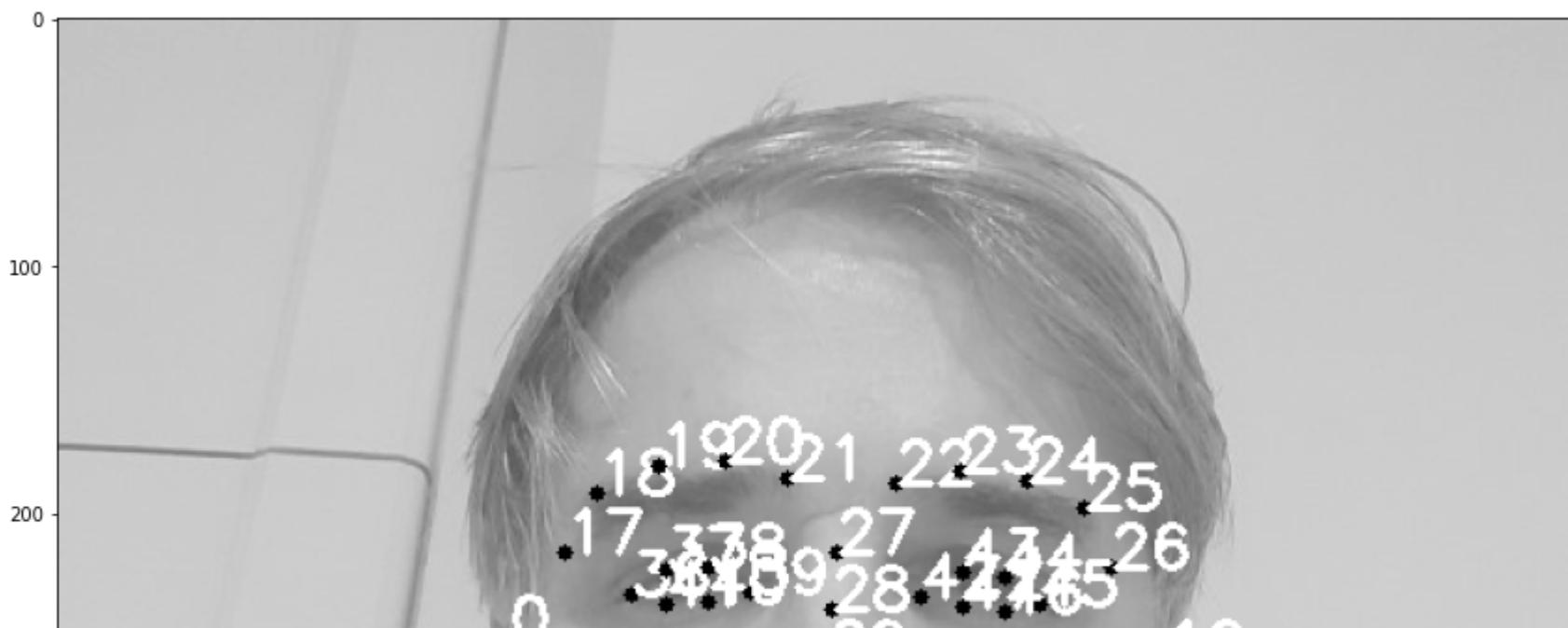
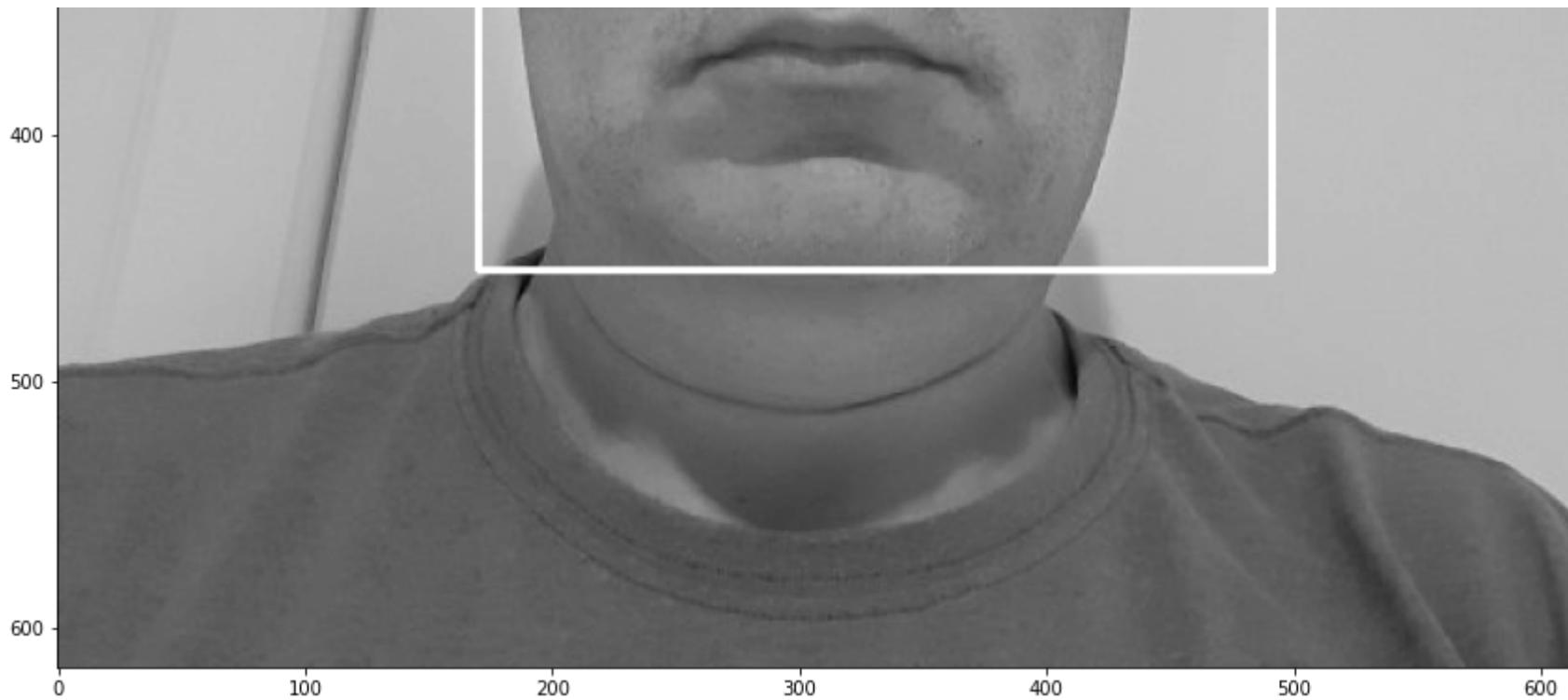
Aspecto razão dos olhos:

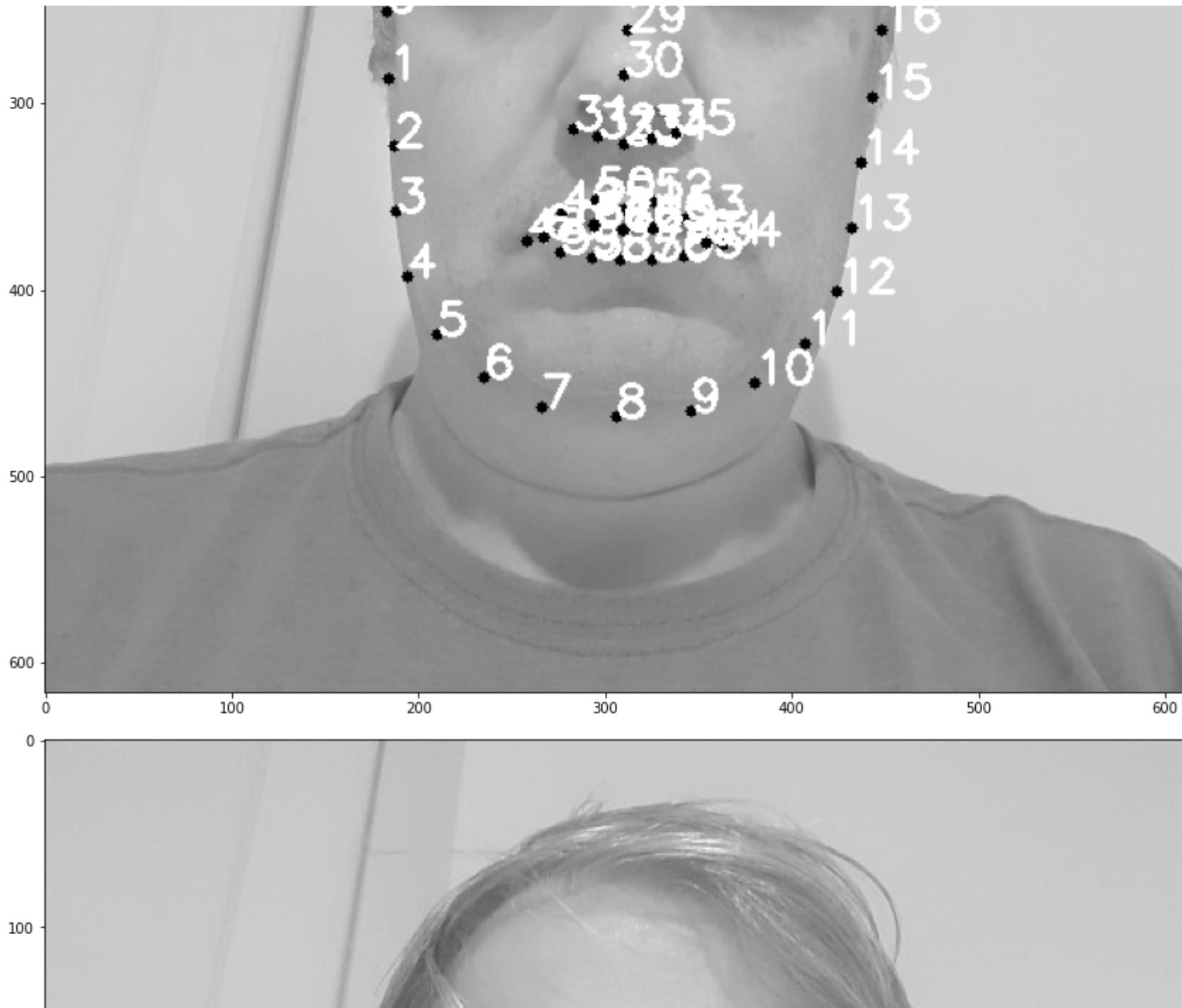
Olho esquerdo 0.29160339154569936

Olho direito 0.2910986687241758











## ▼ Olhos Arregalados

```
1 olhos_arregalados = cv2.imread("FotosPauloAlmeida/arregalados.jpg", cv2.IMREAD_GRAYSCALE)
2 analisa_olhos(olhos_arregalados)
```

Imagen

Rostos identificados: 0

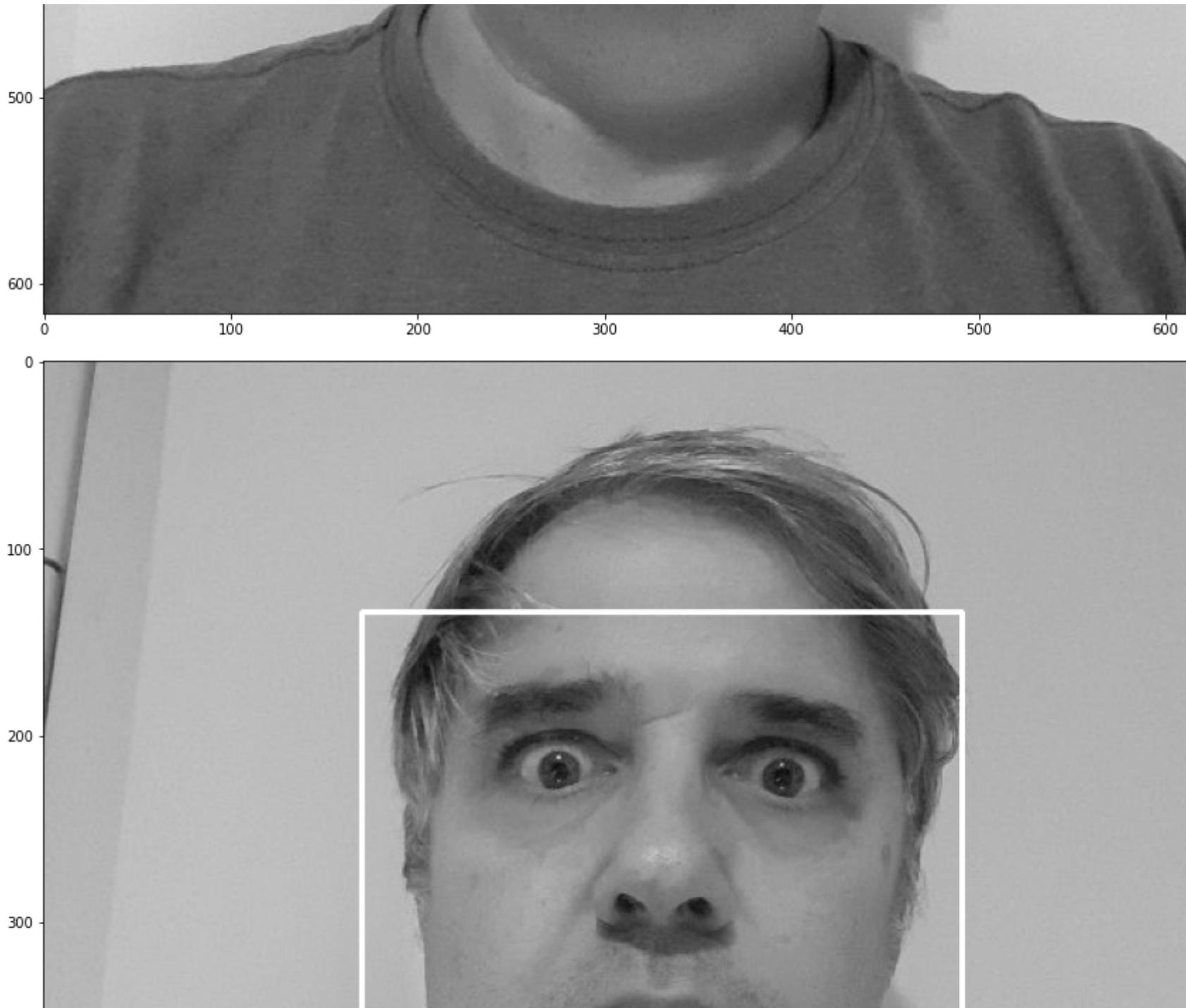
Contorno dos olhos

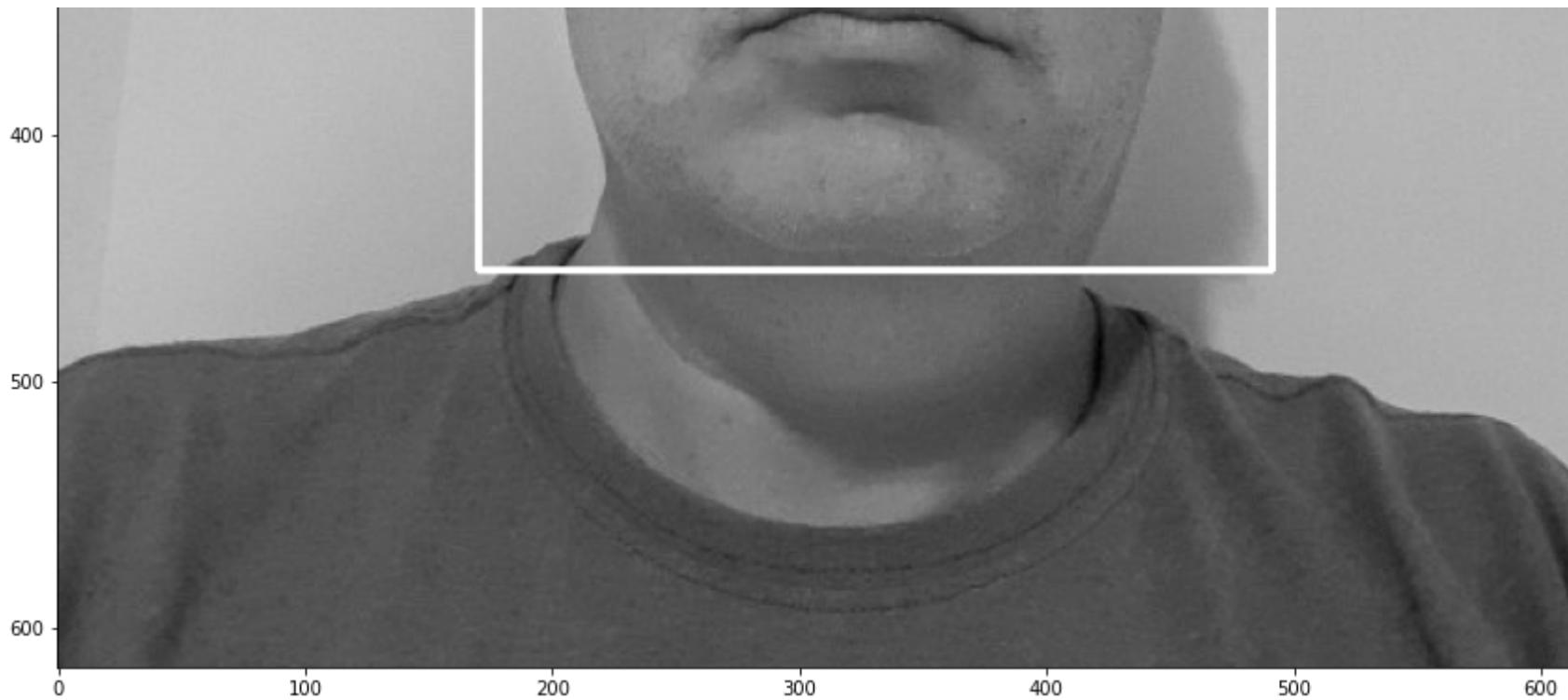
Aspecto razão dos olhos:

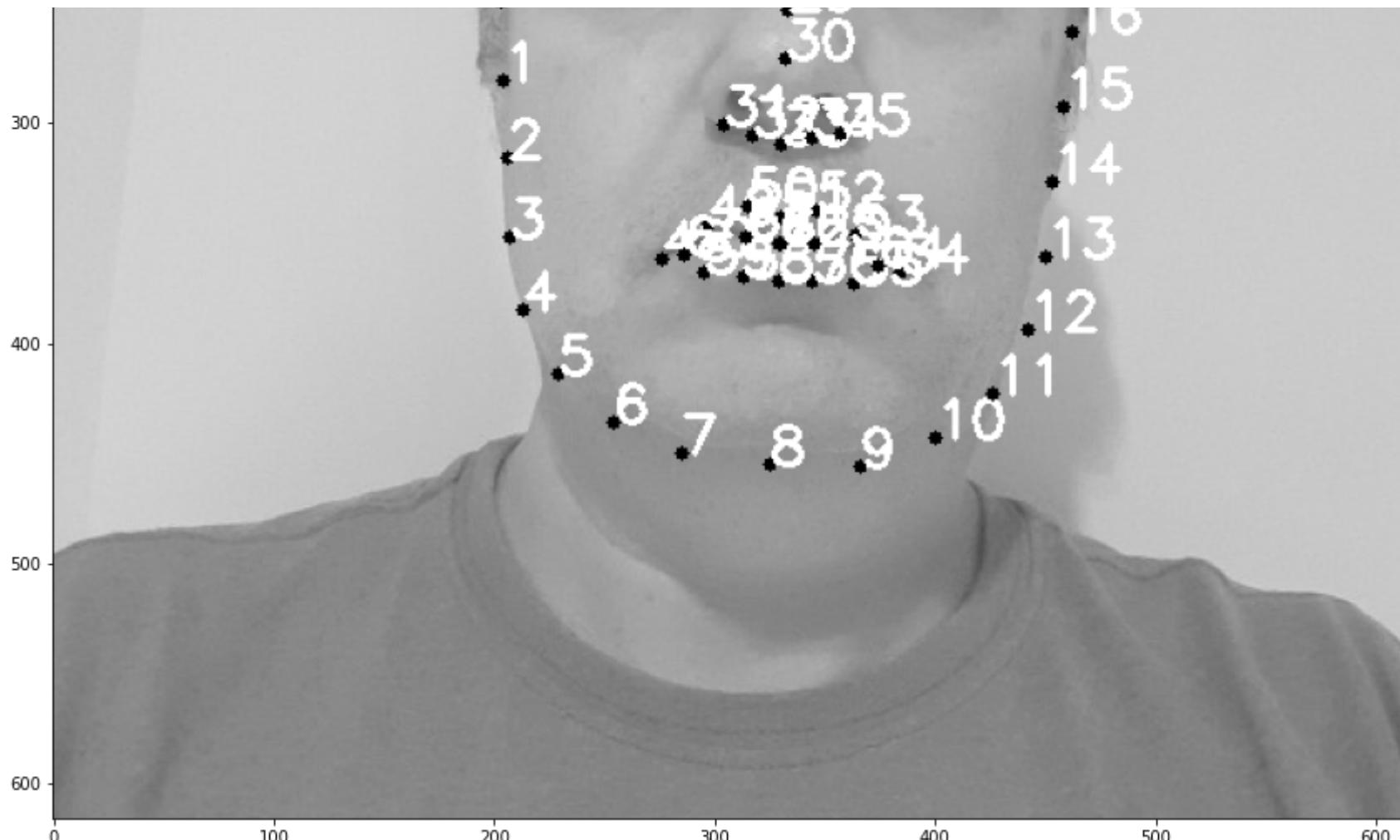
Olho esquerdo 0.46089988151779815

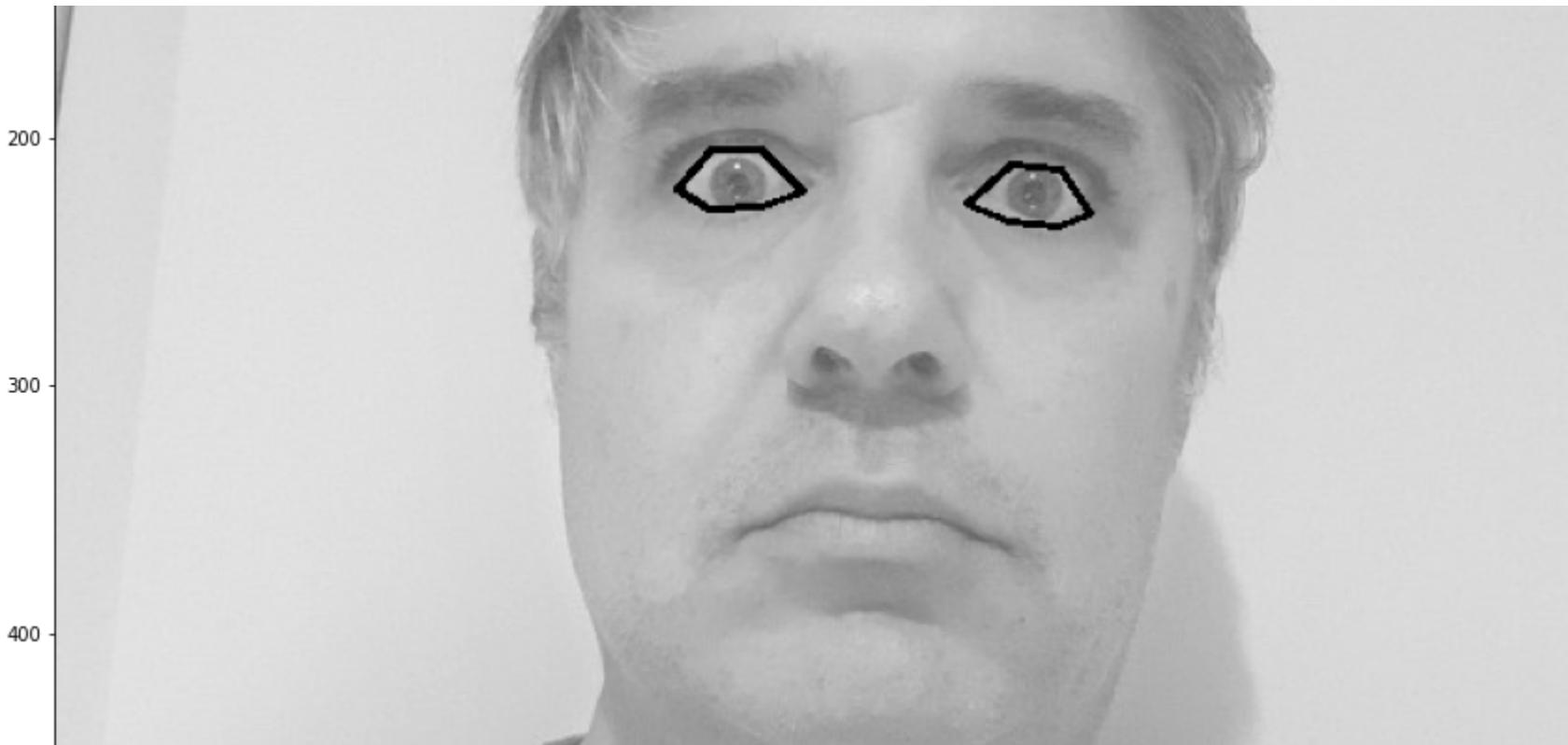
Olho direito 0.4589682214477241











```
1 olhos_atual = cv2.imread("FotosPauloAlmeida/rostoAtual.jpg", cv2.IMREAD_GRAYSCALE)
2 analisa_olhos(olhos_atual)
```

Imagen

Rostos identificados: 0

Contorno dos olhos

Aspecto razão dos olhos:

Olho esquerdo 0.3823040321939709

Olho direito 0.3599280215928025

