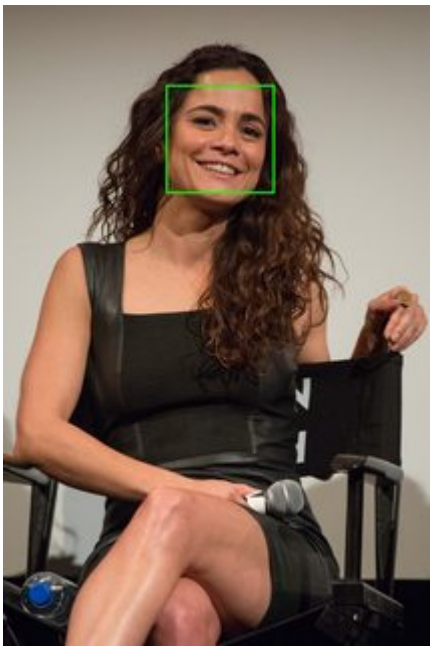
Face and hair detection for Python

Asked 4 years, 1 month ago Active 2 years, 2 months ago Viewed 3k times

2

1

I am using `OpenCV` or `dlib` to detect face from images. The result is very good. Here is an example:



However, I also want to take the hair and the neck from the image, like that:

I have tried to look for a library or framework to help me achieve that but I can't find one.

Are there any way to do that?

python opencv deep-learning face-detection dlib

Share Edit Follow

asked Jun 1 '17 at 17:56

pexea12
911 1 14 30

2 You need to train your own model for this. As this is out of scope for StackOverflow I'd recommend to post this on either CrossValidated or Data Science page of SO. – Martin Krämer Jun 1 '17 at 19:13

2 You just need to train a model that instead of considering the face, also cconsiders hair and neck. You can probably use the same face detection library, but you just change the training data. – Dr. Snoopy Jun 1 '17 at 20:16

try to increase `for (x,y,w,h) in faces:` `cv.rectangle(img,(x,y),(x+w,y+h),(255,0,0),2)` **X,Y,W,H** – Vinu Vish Jun 11 '18 at 5:49

2 Answers

Active | Oldest | Votes

0

In case you want to extract exactly region of hair and neck, you need to train your own model because the current dlib model does not include them.

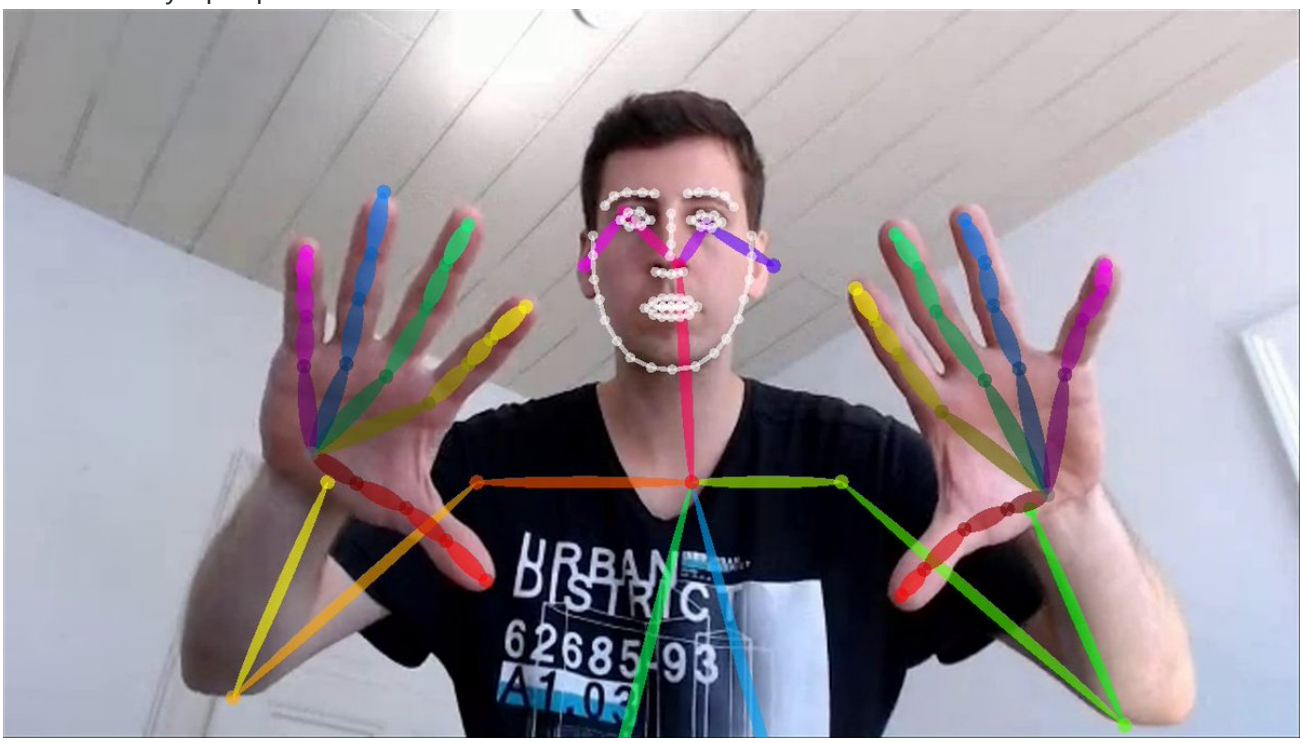
Otherwise, you just want to capture relatively, you can use Openpose which gives you the landmarks of faces + ears + shoulders (even body and hand fingers). From those landmarks you can draw your interested area.

Example:

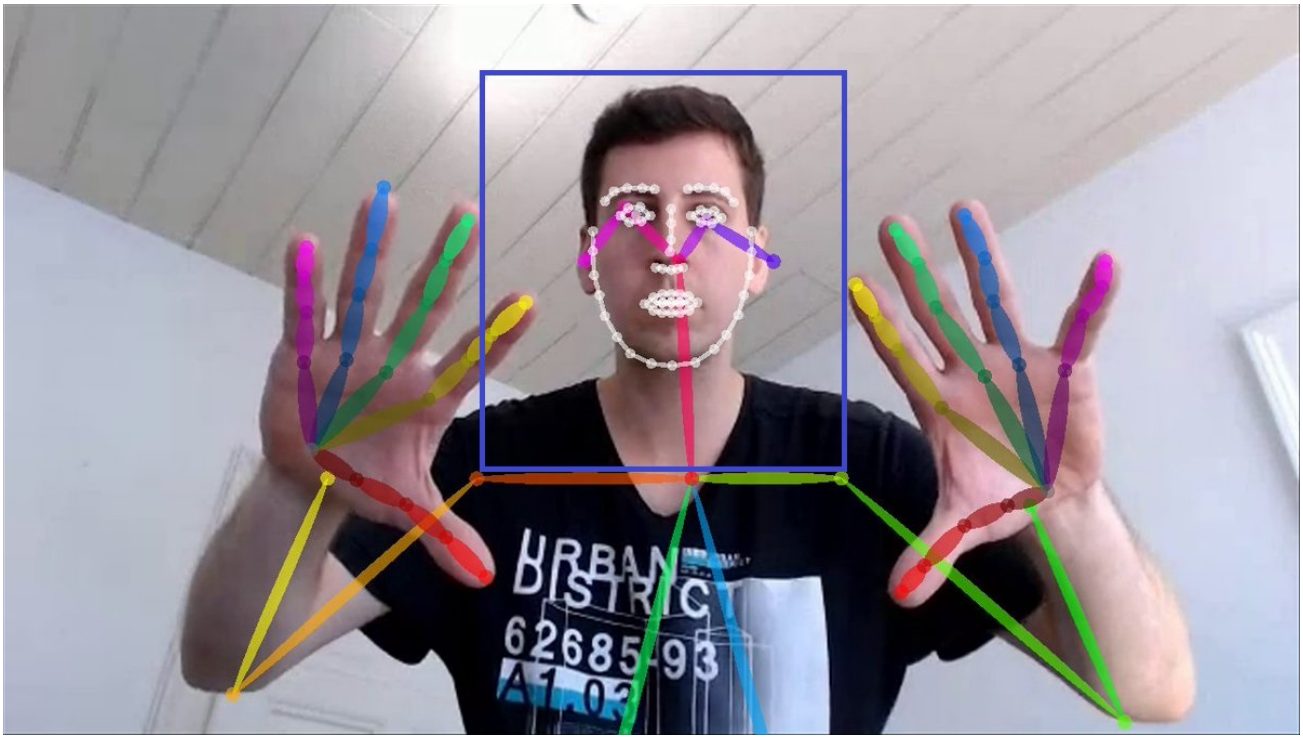
the width of rectangle = the length of shoulder (point 2 -> point 5)

the height = the length from the neck to (point 1) to the nose (point 0) x 2. (point 1 - point 0)*2

landmarks by openpose



face + hair + neck



Share Edit Follow

answered May 16 '19 at 7:28

Peter Lee

351 2 7

use this code to increase the bounding box by percentage.

0

```
rects = detector(original_image, 1)
for rect in rects:
    (x, y, w, h) = rect_to_bb(rect)
    x_inc = int(w*0.3)
    y_inc = int(h*0.3)
    sub_face = original_image[y-y_inc:y+h+y_inc, x-x_inc:x+w+x_inc]
    newimg = cv2.resize(sub_face,(int(224),int(224)))
```

Share Edit Follow

answered May 15 '19 at 9:05

Manikandan
265 2 7