Introduction and Pre-requisites

This python program is used to crop checkboxes from raw images containing checkboxes and some noise. We primarily use OpenCV library. Other libraries needed are numpy. I am using pyCharm IDE (Community Edition 2022.1.2), python 3.10.4, opencv-python version – 4.5.5.64 and numpy version 1.22.4.

Basic Logic Used

We are going to use contours in OpenCV and manipulate them to find and crop our checkbox. Firstly, all the curves in the image are contoured with external retrieval mode and approximation method as none. Then we need to identify the checkbox from the rest of the curves. Then once we know which contour belongs to the checkbox, we can find its end points from the list of points in the contour. Using these end points, we finally crop the image.

Implementation

Contouring the curves

- The imported image is first sharpened so that no problem is caused if the image is blurry.
- Convert sharpened image to grayscale
- From the grayscaled image we remove irregularities and noise my applying gaussian blur
- Next we apply canny edge detection and finally find contours of the image

```
import cv2
import numpy as np

for j in [0,1,2,3,4,5,6,7,8,9]:       #to open and crop each image one by one
    nm = r'{}'.format(j)
    path = r"C:\Users\Vinayak Shrivastava\PycharmProjects\pythonProject\checkbox\Raw images\img-" + nm +".jpg"
    img = cv2.imread(path)
    sharpen_kernel = np. array([[-1,-1,-1], [-1,9,-1], [-1,-1,-1]])      #sharpens the image if it is blurry
    imgs = cv2. filter2D(img, -1, sharpen_kernel)
    imgg = cv2.cvtColor(imgs,cv2.COLOR_BGR2GRAY)                        #processing the image
    imgb = cv2.GaussianBlur(imgg,(5,5), 1)
    imgf = cv2.Canny(imgb,10,50)

    contours, hierarchy = cv2.findContours(imgf,cv2.RETR_EXTERNAL,cv2.CHAIN_APPROX_NONE)          # making contours
```

Finding the checkbox

- Variable z is set 0 to find the contour with maximum number of points. Variable i is used to run the loop and is also used to store index of the desired contour in variable a.
- After the loop ends z has the maximum number of points present in a contour and a has the index of that contour thus giving us the index of contour of the checkbox.

```
z=0
i = len(contours) - 1
while(i>0):
    if(z < len(contours[i])):
        z = len(contours[i])
        a=i
    i=i-1
```

Finding the endpoint of the checkbox

- Variable x_max and y_max to store the bottom right coordinate and x_min, y_min to store the top left.
- Variable y_min is assigned the number of total rows of the image or the maximum possible value of y coordinate for the image. Similarly x_min is assigned number of cols of image.
- Next we find the minimum x coordinate, minimum y coordinate, maximum x coordinate, maximum y coordinate from all the available points.
- Thus we have the top left and bottom right points of the checkbox which we can now use to crop the image.

```
x_max=y_max=0
y_min, x_min, _ = img.shape

for [[x, y]] in contours[a]:
    if(x > x_max): x_max = x
    if(y > y_max): y_max = y
    if(x < x_min): x_min = x
    if(y < y_min): y_min = y
```

Cropping and Saving the image

- New image is formed starting from the row = y_min and col= x_min and ending at row = y_max and col=x_max.
- This is basically the rectangle enclosing the checkbox.
- Next we save the image using imwrite().

```
img = img[y_min:y_max,x_min:x_max]

nm = r"image-"+nm+" crop.jpg"
cv2.imwrite(nm,img)
```

Other features used

- The whole program is enclosed under a for loop which basically opens every image one by one and crop then and then save them. That is for cropping all the present images one by one.

- Since the raw images are named number wise we run a simple for loop from 0 to 9 and name our images accordingly

```
for j in [0,1,2,3,4,5,6,7,8,9]:       #to open and crop each image one by one
    nm = r'{}'.format(j)
    path = r"C:\Users\Vinayak Shrivastava\PycharmProjects\pythonProject\checkbox\Raw images\img-" + nm +".jpg"
    img = cv2.imread(path)
```

```
nm = r"image-"+nm+" crop.jpg"
cv2.imwrite(nm,img)
```

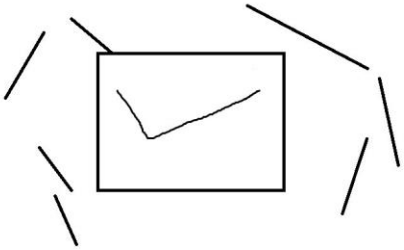
Conclusion and Problems

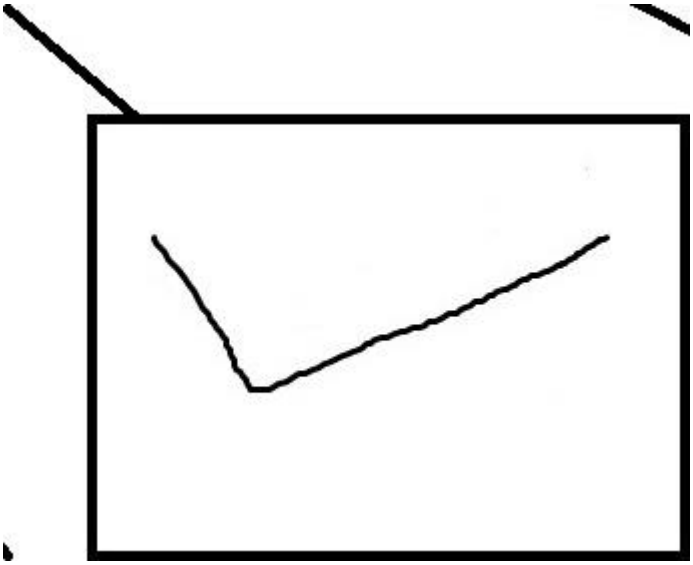
This program can be very effective in cropping out a image from a raw image containing unwanted extra parts. The program basically works by manipulating the contours of OpenCV library.

A major problem in this program is that it will not omit parts which are attached to the checkbox's rectangle. This happens because the points for that attached part are stored with all the other points for the checkbox. As a result cropping will include that attached part as well. Due to this noise omission will not be perfect.

Example

Raw image -





Cropped image -

-Vinayak Shrivastava