**CAM SCANNER IN PYTHON**

**Project Problem Statement**

*The Aim of our Project is:*

*To create our own Cam Scanner that takes an image as input and then scans the document from the image by applying few image processing techniques and gives the output image with scanned effect. Since the Cam Scanner application being launched in Play Store by China in 2011 that allowed iOS and Android devices to be used as image scanners was banned by the Indian Government for stealing the data of its users. So, we felt the need to target on this project where we can create our own Cam Scanner which does not harm the privacy of its users.*

**Key features/Benefits**

1. *Cam Scanner allows users to 'scan' documents (by taking a photo with the device's camera) and share the photo as either a JPEG or PDF.*
2. *It converts images you take with your phone into PDF documents instantly without the need of another application for converting image to PDF.*
3. *It helps users save time by taking photos of desired documents and converting it into PDF format when exporting. Comparing the difference between pictures and PDF documents, CamScanner provides users with high clarity and high quality work when users want to print their documents out.*
4. *Most of us are used to taking a picture with the phone when we see any important information we want to note down in daily life. Then we find that the phone album is overwhelmed by pictures of selfies, sceneries, posters, whiteboards, slide shows, receipts and more. And most of the time, we find it’s extremely difficult to dig out the picture when we need some information. That’s why you need CamScanner instead of the phone camera.*

**List of software used**

* Python 3.7.0 IDLE.
* Virtual Studio Code for fixing the bugs in Python 3.7.0 IDLE and running the python code more efficiently.
* Discord for screen sharing and meetings for discussing about the project.
* Microsoft PowerPoint for doing the PPTs so as to conduct our presentation.
* Github- Here we have uploaded our code, presentation and working of the Cam Scanner.

Why am I using Virtual Studio Code instead of Python IDLE?

I am using Virtual Studio Code 64-Bit 2020 version in my system. VS Code is lightweight and should easily run on 32-bit or 64-bit system.

System requirements for running Virtual Studio Code are:

1. 1.6 GHz or faster processor
2. 1 GB of RAM
3. Microsoft .NET Framework 4.5.2 is required for VS Code. If the user is using Windows 7, we have to make sure .NET Framework 4.5.2 is installed.

According to me, the main reason I am using Virtual Studio Code because it has everything which any programmer expects from any code editor with some additional and useful features. It’s lightweight, fast, open source and cross-platform nature along with other features gives it an extra edge over any other editor.

1. It comes with an in-built debugger which is also one of its key features. It helps in accelerating any programmer’s edit, compile and debug loop.
2. Visual Studio Code also provides features for code management like Go to Definition, Peek definition, Find all references and rename Symbol.

**Deliverables**

1. ***Link for the modules used in our project:***

[***https://github.com/sahuljr7/CAM-Scanner-in-Python***](https://github.com/sahuljr7/CAM-Scanner-in-Python)

1. ***Link for the presentation:***

[*https://github.com/sahuljr7/CAM-Scanner-in-Python/blob/main/Presentation%20Final.pptx*](https://github.com/sahuljr7/CAM-Scanner-in-Python/blob/main/Presentation%20Final.pptx)

1. ***Link for the code Scanner.py and mapper.py:***

[*https://github.com/sahuljr7/CAM-Scanner-in-Python/blob/main/Scanner.py*](https://github.com/sahuljr7/CAM-Scanner-in-Python/blob/main/Scanner.py)

[*https://github.com/sahuljr7/CAM-Scanner-in-Python/blob/main/mapper.py*](https://github.com/sahuljr7/CAM-Scanner-in-Python/blob/main/mapper.py)

1. ***Link for the image we used in our project:***

[*https://github.com/sahuljr7/CAM-Scanner-in-Python/blob/main/test\_img.jpg*](https://github.com/sahuljr7/CAM-Scanner-in-Python/blob/main/test_img.jpg)

Summary of the Cam Scanner Project

Initially we need to resize the images so OpenCV can handle it and then the following functions are applied: -

1. Gaussian Blur to smoothen image.
2. Canny Edges to detect the edges.
3. Find contours and boundary of the page.
4. Map the end points of contours to 800 \* 800 window.
5. Apply perspective transform to get scanned or bird eye view of the image.

Source code  
mapper.py

import numpy as np

def mapp(h):

    h = h.reshape((4,2))

    hnew = np.zeros((4,2),dtype = np.float32)

    add = h.sum(1)

    hnew[0] = h[np.argmin(add)]

    hnew[2] = h[np.argmax(add)]

    diff = np.diff(h,axis = 1)

    hnew[1] = h[np.argmin(diff)]

    hnew[3] = h[np.argmax(diff)]

    return hnew

Scanner.py

import cv2

import numpy as np

import mapper

image=cv2.imread("test\_img.jpg")   #read in the image

image=cv2.resize(image,(1300,800)) #resizing because opencv does not work well with bigger images

orig=image.copy()

gray=cv2.cvtColor(image,cv2.COLOR\_BGR2GRAY)  #RGB To Gray Scale

cv2.imshow("Gray Scale",gray)

blurred=cv2.GaussianBlur(gray,(5,5),0)  #(5,5) is the kernel size and 0 is sigma that determines the amount of blur

cv2.imshow("Blur",blurred)

edged=cv2.Canny(blurred,30,50)  #30 MinThreshold and 50 is the MaxThreshold

cv2.imshow("Canny",edged)

contours,hierarchy=cv2.findContours(edged,cv2.RETR\_LIST,cv2.CHAIN\_APPROX\_SIMPLE)  #retrieve the contours as a list, with simple apprximation model

contours=sorted(contours,key=cv2.contourArea,reverse=True)

#the loop extracts the boundary contours of the page

for c in contours:

    p=cv2.arcLength(c,True)

    approx=cv2.approxPolyDP(c,0.02\*p,True)

    if len(approx)==4:

        target=approx

        break

approx=mapper.mapp(target) #find endpoints of the sheet. Passing the target image

to mapper.py

pts=np.float32([[0,0],[800,0],[800,800],[0,800]])  #map to 800\*800 target window

op=cv2.getPerspectiveTransform(approx,pts)  #get the top or bird eye view effect

dst=cv2.warpPerspective(orig,op,(800,800))

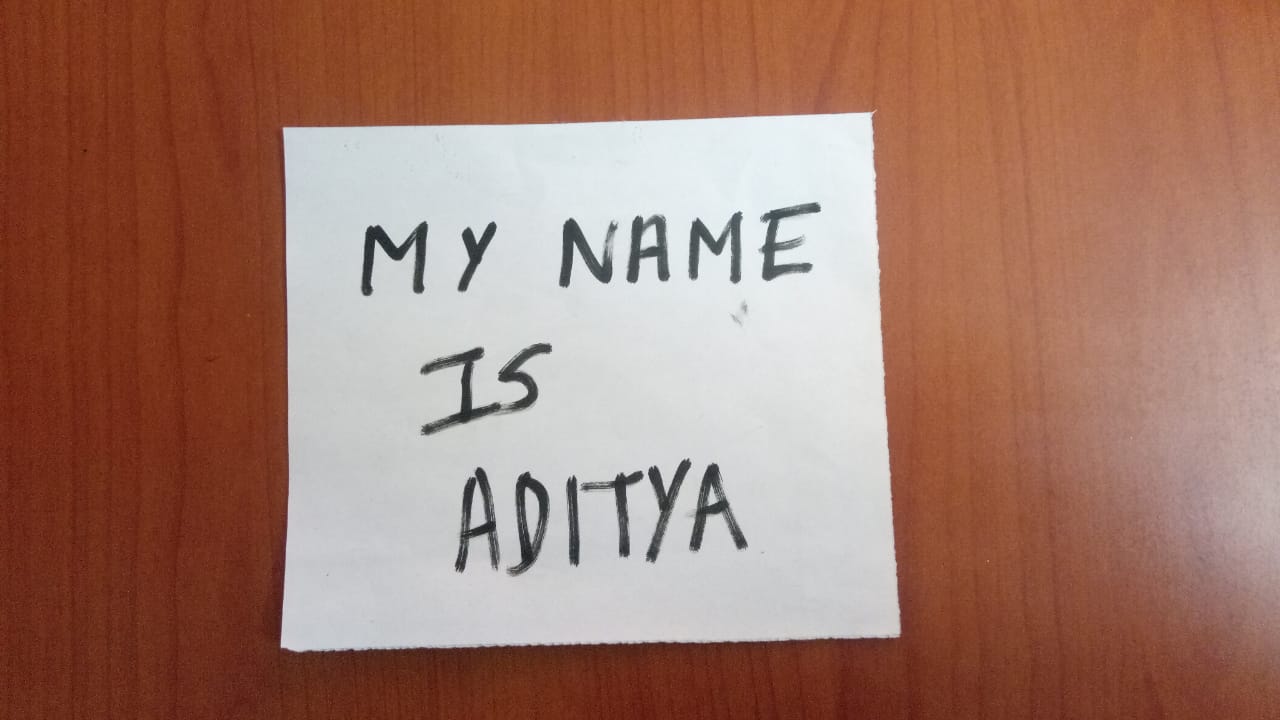
cv2.imshow("Scanned",dst)

# press q or Esc to close

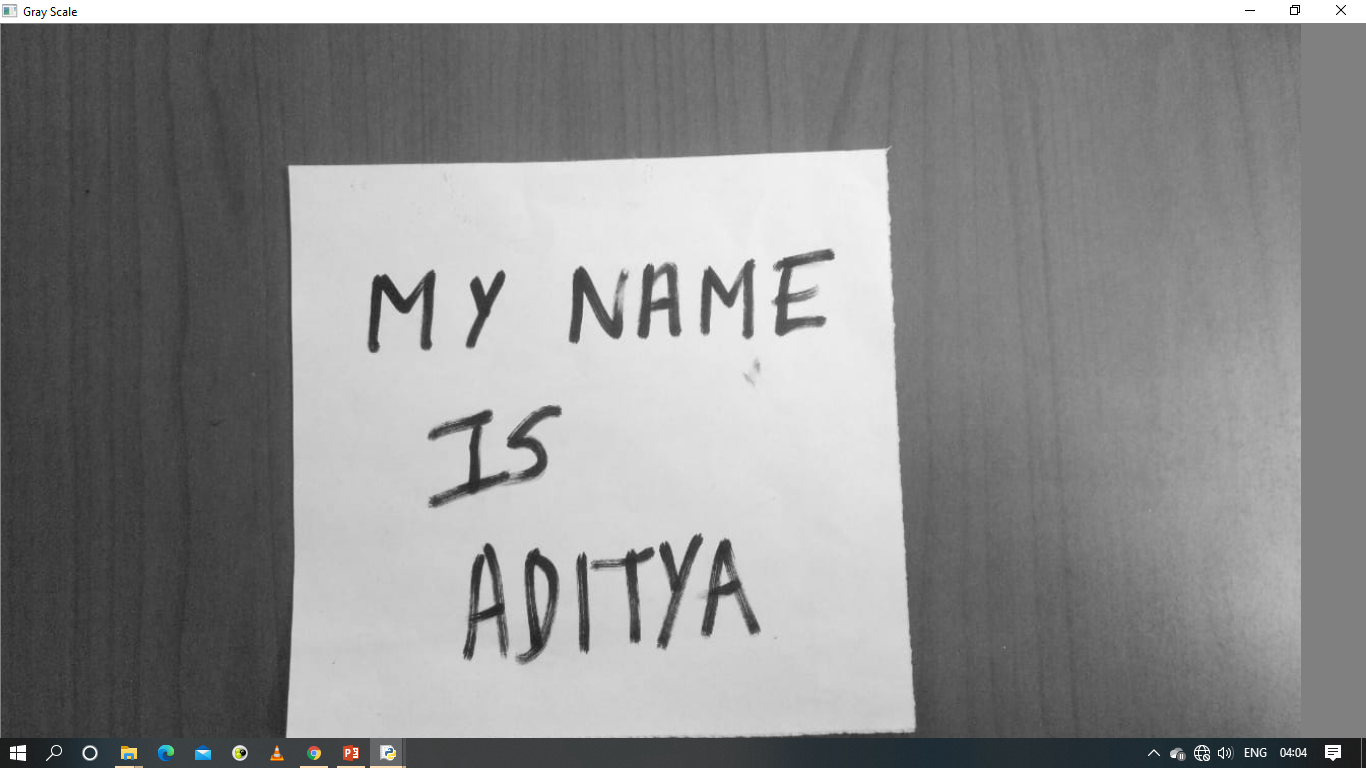
cv2.waitKey(0)

cv2.destroyAllWindows()

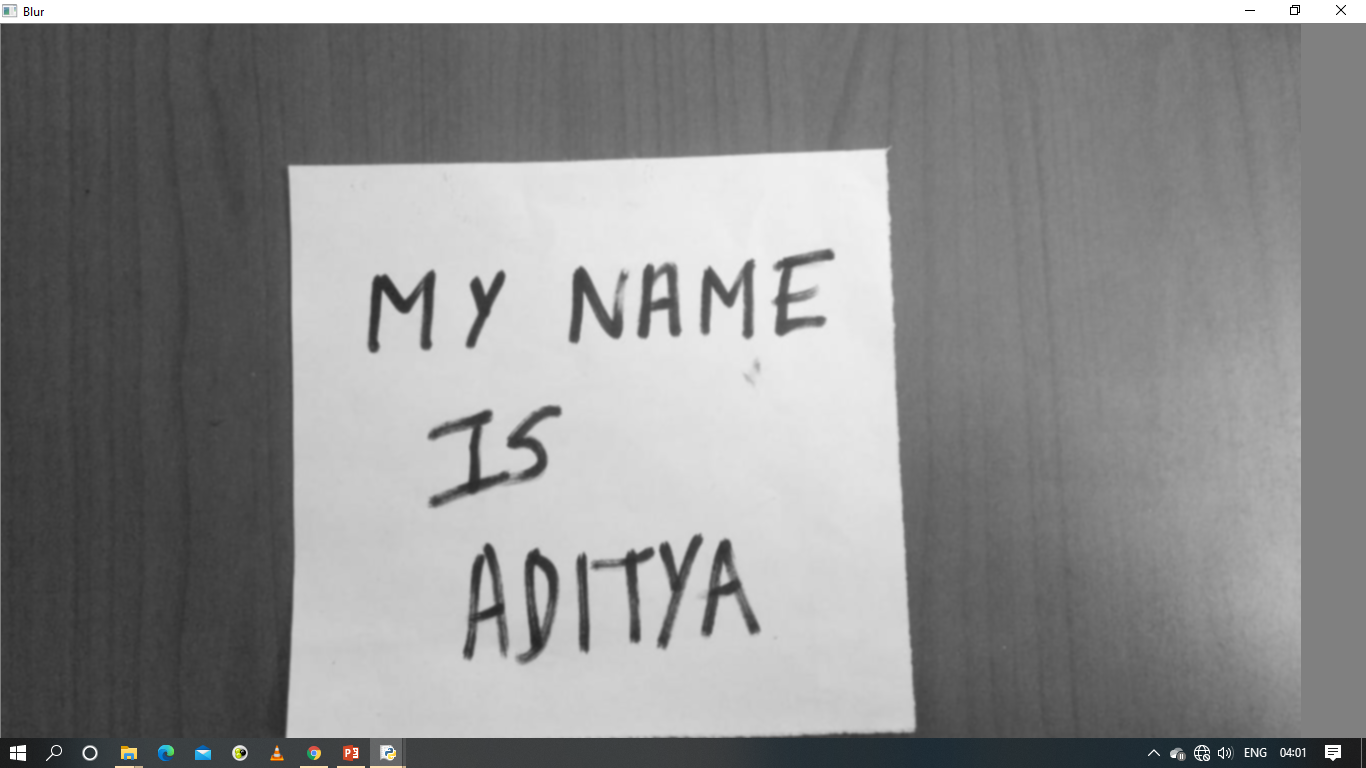
Original Image



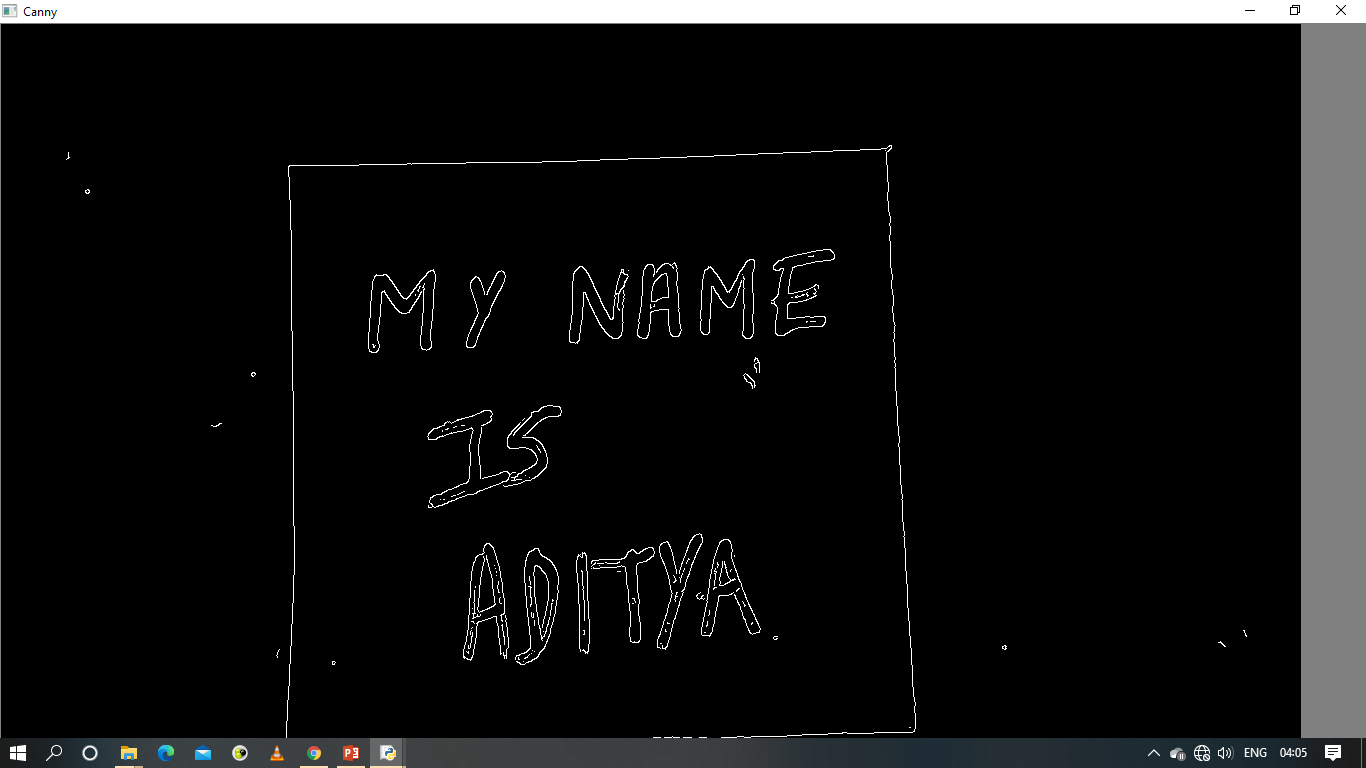
Output after running the Python Code for Scanner.py  
First we get the gray scale image entitled “Gray Scale” as output.



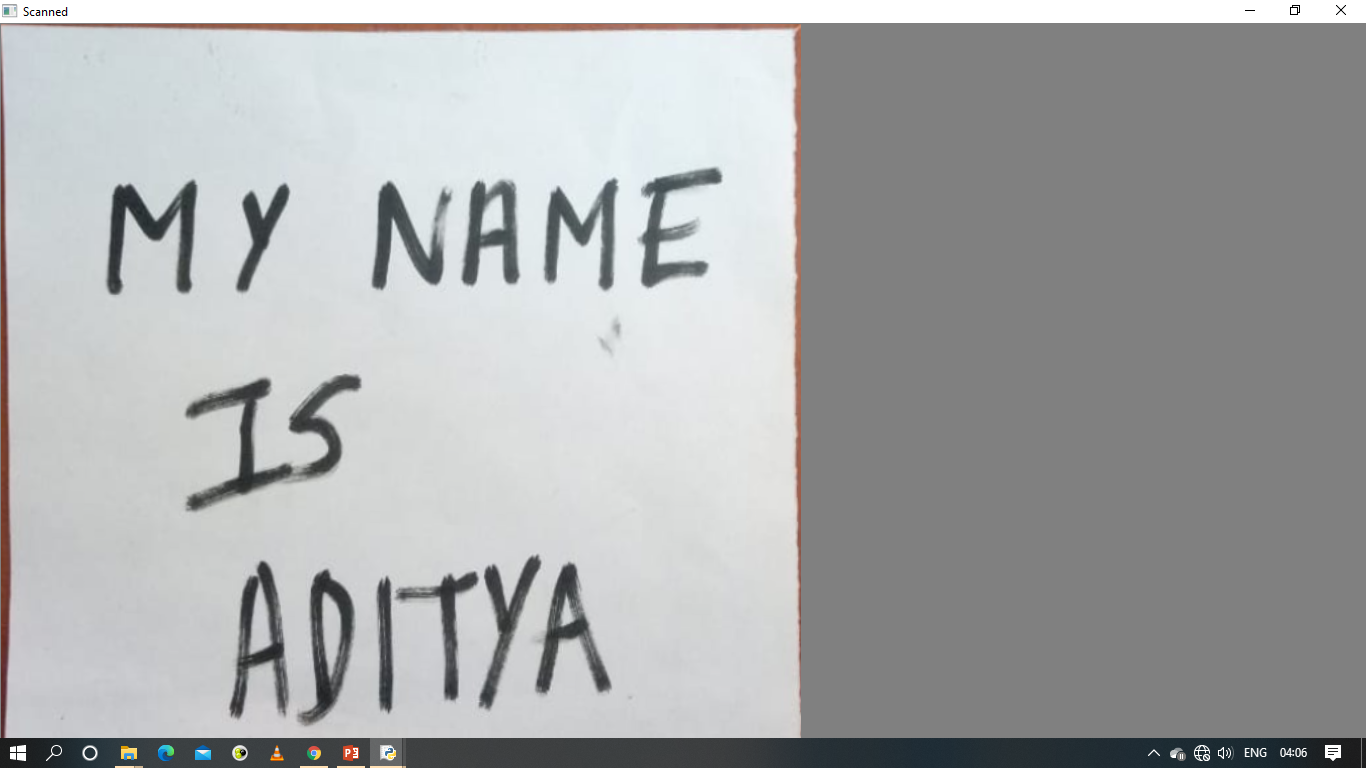
Secondly we get the Blurred Image entitled “Blur”.



Thirdly we get the Canny Detected Image entitled “Canny”.



Fourthly we get the Scanned Image entitled “Scanned” which is the final output.



**Conclusion**

This python script takes an image as input and then scans the document from the image by applying few image processing techniques and gives the output image with scanned effect.