%matplotlib inline

img = cv2.imread('aa.jpg')

cv2\_imshow(img)

#Convert to grayscale

img\_gray = cv2.cvtColor(img, cv2.COLOR\_BGR2GRAY)

#Blur the image for better edge detection

img\_blur = cv2.GaussianBlur(img\_gray, (3,3), 0)

#Sobel edge detection

sobelx = cv2.Sobel(src = img\_blur, ddepth = cv2.CV\_64F, dx = 1, dy = 0, ksize = 5) #Sobel for x

sobely = cv2.Sobel(src = img\_blur, ddepth = cv2.CV\_64F, dx = 0, dy = 1, ksize = 5) #Sobel for y

sobelxy = cv2.Sobel(src = img\_blur, ddepth = cv2.CV\_64F, dx = 1, dy = 1, ksize = 5) #Combine for xy

#Display sobel edge detection images

plt.subplot(1,3,1),plt.imshow(sobelx, cmap = 'gray')

plt.title('Sobel X axis'), plt.xticks([]), plt.yticks([])

plt.subplot(1,3,2), plt.imshow(sobely, cmap= 'gray')

plt.title('Sobel Y axis'), plt.xticks([]), plt.yticks([])

plt.subplot(1,3,3), plt.imshow(sobely, cmap= 'gray')

plt.title('Sobel XY'), plt.xticks([]), plt.yticks([])

plt.show()

Graphical user interface, text

Description automatically generated

#Canny Edge detection

edges = cv2.Canny(image = img\_blur, threshold1 = 100, threshold2 = 200) #Canny edge detection

#Display canny edge detectuib image

cv2\_imshow(edges)

Text

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

A group of people smiling

Description automatically generated

Text

Description automatically generated

%matplotlib inline

image1 = np.zeros((400,400), dtype = "uint8")

img1 = cv2.imread('bb.jpg')

img2 = cv2.imread('cc.jpg')

cv2\_imshow(img1)

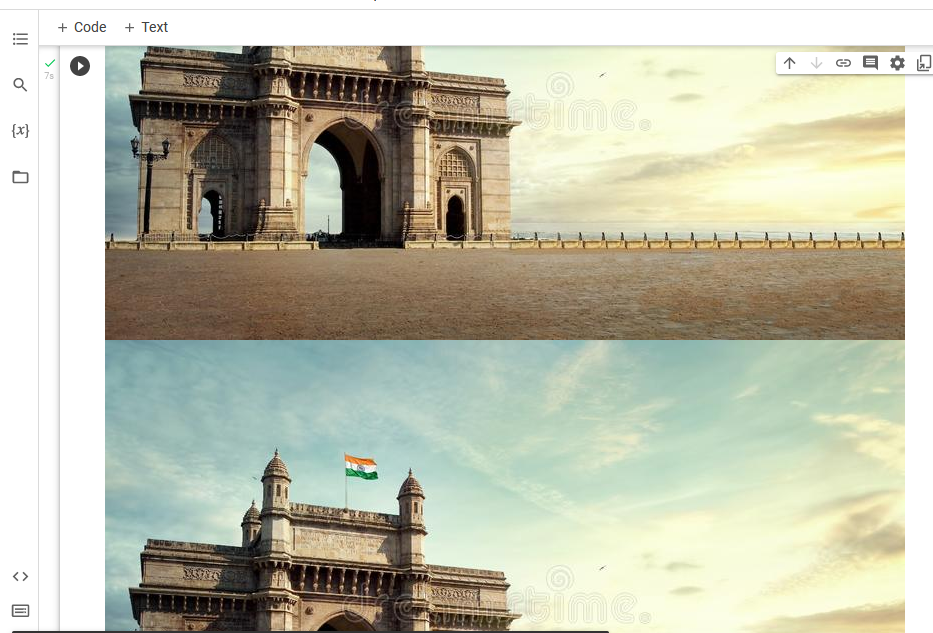
cv2\_imshow(img2)

weightedsum = cv2.addWeighted(img1, 0.5, img2, 0.4, 0)

cv2\_imshow(weightedsum)

Graphical user interface

Description automatically generated

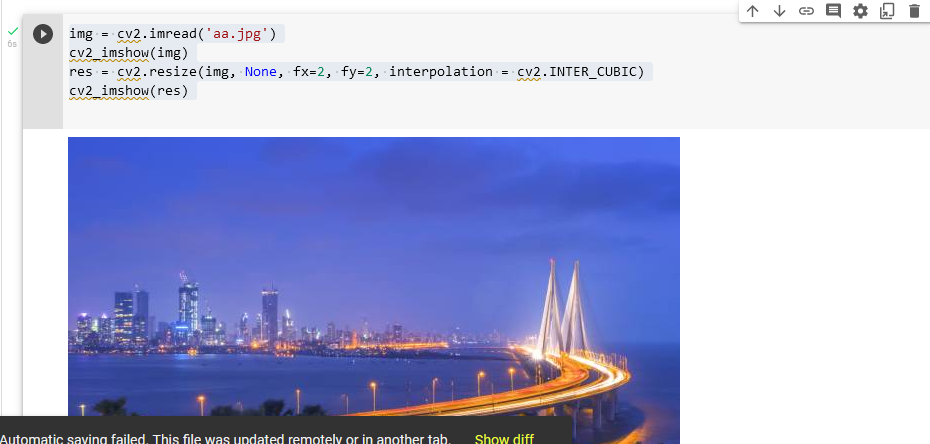


img = cv2.imread('aa.jpg')

cv2\_imshow(img)

res = cv2.resize(img, None, fx=2, fy=2, interpolation = cv2.INTER\_CUBIC)

cv2\_imshow(res)



A bridge over a body of water with a city in the background

Description automatically generated with medium confidence

height, width = img.shape[:2]

res = cv2.resize(img,(2\*width, 2\*height), interpolation = cv2.INTER\_CUBIC)

cv2\_imshow(res)

Graphical user interface

Description automatically generated with medium confidence

A picture containing text, city, screenshot

Description automatically generated

img\_shrinked = cv2.resize(img,(100,100), interpolation = cv2.INTER\_AREA)

cv2\_imshow(img\_shrinked)

Graphical user interface, text, application, email

Description automatically generated

img = cv2.imread('gaurav.jpg',0)

cv2\_imshow(img)

rows,cols = img.shape

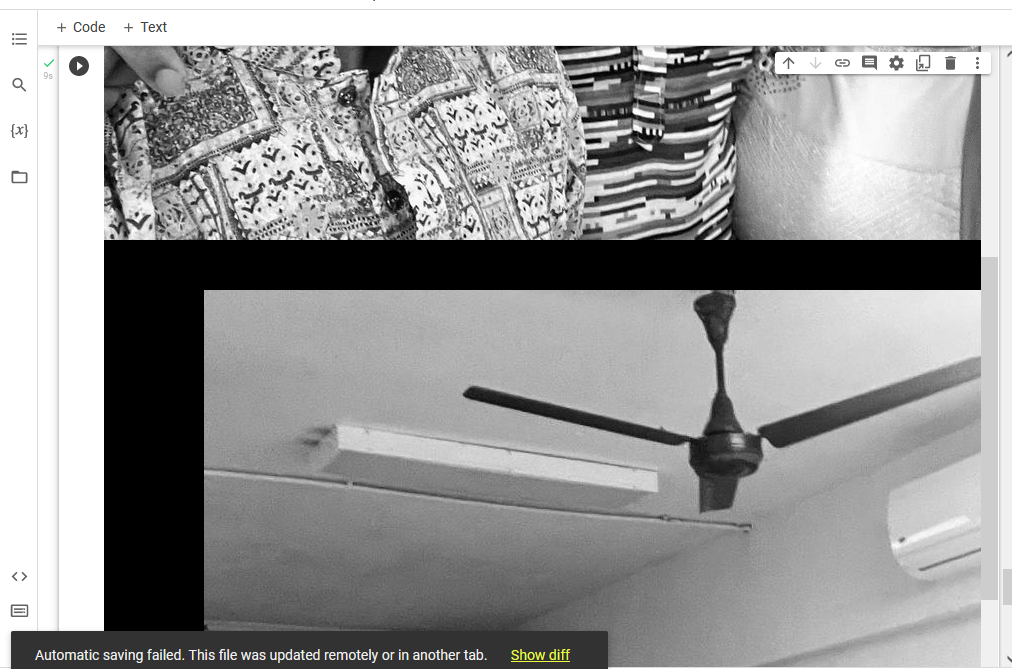
M = np.float32([[1,0,100], [0,1,50]])

dst = cv2.warpAffine(img, M, (cols, rows))

cv2\_imshow(dst)

A picture containing text, person, indoor, screenshot

Description automatically generated



img = cv2.imread('dd.jpg')

cv2\_imshow(img)

rows,cols = img.shape[:2]

pts1 = np.float32([[50,50], [200,50], [50,200]])

pts2 = np.float32([[10,100], [200,50], [100,250]])

M = cv2.getAffineTransform(pts1, pts2)

dst = cv2.warpAffine(img, M, (cols, rows))

plt.subplot(121), plt.xticks([]), plt.yticks([]), plt.imshow(img), plt.title('Input')

plt.subplot(122), plt.xticks([]), plt.yticks([]), plt.imshow(img), plt.title('Output')

plt.show()

