Name:- Krishna Mundada

Roll No:- 45

Batch:- E3

Practical 3

Topic:- Resconstructing image using PCA

`PCA:- Is a technique to reduce the dimensionality of the input matrix while retaining as much as information as possible.`

In [ ]:
```
import cv2
import numpy as np
import os
```

In [ ]:
```
def get_image_size(filename):
    size=os.stat(filename).st_size
    return size/1024.0
```

In [ ]:
```
imgpath = "krishna1.jpg"
# imgpath = "4.2.07.tiff"
img = cv2.imread(imgpath, 0)

# Reshaping the image to square matrix
if img.shape[0] != img.shape[1]:
    min_side = min(img.shape[:2])
    img = img[
        (img.shape[0]-min_side)//2:(img.shape[0]+min_side)//2,
        (img.shape[1]-min_side)//2:(img.shape[1]+min_side)//2
    ]

M = np.mean(img.T, axis=1)
C = img - M
V = np.cov(C.T)
values, vectors = np.linalg.eig(V)

p = np.size(vectors, axis =1)

idx = np.argsort(values)
idx = idx[::-1]
vectors = vectors[:,idx]
values = values[idx]

num_PC = 200

if num_PC <p or num_PC >0:
    vectors = vectors[:, range(num_PC)]
```

```
score = np.dot(vectors.T, C)
constructed_img = np.dot(vectors, score) + M
constructed_img = np.uint8(np.absolute(constructed_img))

cv2.imshow('Original Image', img)

cv2.imshow("Reconstructed Image", constructed_img)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

In [ ]:
```
filename="krishna1.jpg"
size_kb=get_image_size(filename)
print(f"The size of original image is {size_kb:.2f}")
filename="constructed_img.jpg"
size_kb=get_image_size(filename)
print(f"The size of reconstructed image is {size_kb:.2f}")
```

```
The size of original image is 445.25
The size of original image is 68.67
```

Explanation:- First, we read an input image using OpenCV and convert it to grayscale. Then, we reshape the image to a square matrix If it is not in the square shape already.

Next, we calculate the mean column-wise from the reshaped input image and subtract the resulting mean matrix from the original matrix. This results in the centered matrix C.

After that, we calculate the covariance matrix V of the centered matrix C with its transpose matrix.

Then we compute the eigenvalues and eigenvectors of the covariance matrix V. The eigenvectors represent the principal components (PCs) while the eigenvalues represent the amount of variance each PC holds.

We sort the eigenvalues and eigenvectors in descending order of values and select specific number of PCs which can be used for reconstruction.

Then we multiply the centered matrix with selected PC's to obtain score, which can be used for reconstruction. And add the mean matrix M to the resultant reconstruction to obtain final reconstructed image.

Explanation:- First, we read an input image using OpenCV and convert it to grayscale. Then, we reshape the image to a square matrix If it is not in the square shape already.

Next, we calculate the mean column-wise from the reshaped input image and subtract the resulting mean matrix from the original matrix. This results in the centered matrix C.

After that, we calculate the covariance matrix V of the centered matrix C with its transpose matrix.

Then we compute the eigenvalues and eigenvectors of the covariance matrix V. The eigenvectors represent the principal components (PCs) while the eigenvalues represent the