COMPUTER VISION LAB

(Course Code: 22UPCSC1E20)

A programming laboratory record submitted to Periyar University, Salem

In partial fulfillment of the requirements for the degree of

MASTER OF COMPUTER APPLICATIONS

By

ELANCHEZHIAN M

[Reg. No.: U22PG507CAP006]

DEPARTMENT OF COMPUTER SCIENCE

PERIYAR UNIVERSITY

(NAAC `A++` Grade with CGPA 3.61) – NIRF RANK 59 – ARIIA RANK 10

PERIYAR PALKALAI NAGAR,

SALEM – 636 011.

(APRIL - 2023)

CERTIFICATE

This is to certify that the Programming Laboratory entitled "**COMPUTER VISION LAB (22UPCSC1E20)"** is a bonafide record work done by Mr. /Ms. _____

Register No: _____ as partial fulfillment of the requirements for the degree of Master of Computer Applications, in the Department of Computer Science, Periyar University, Salem, during the Academic Year 2023-2024.

Staff In-charge Head of the Department

Submitted for the practical examination held on……..……

Internal Examiner External Examiner

CONTENTS

SOURCE CODE:

```
import cv2

img = cv2.imread("C:/Users/Dell/Downloads/modi.jpeg")
imgGray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
cv2.imshow("Gray Image", imgGray)
cv2.waitKey(0)
cv2.imwrite("E:/College Info/sem 2/lab/cv/modi.jpeg", img)
```

OUTPUT:

SOURCE CODE:

```
import cv2

img = cv2.imread('C:/Users/Dell/Downloads/1.jpg', cv2.IMREAD_COLOR)
value = img[10, 10, :]
print("ACCESSING PIXEL VALUES :", value)

img[10, 10, 0] = 255
value = img[10, 10, :]
print("MODIFYING PIXEL VALUES :", value)

cv2.imshow('Image', img)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

OUTPUT:

```
MODIFYING PIXEL VALUES FOR GRAYSCALE IMAGES
[ 28  98 246]
ACCESSING PIXEL VALUES FOR COLOR IMAGES
[255  98 246]
```

:

SOURCE CODE:

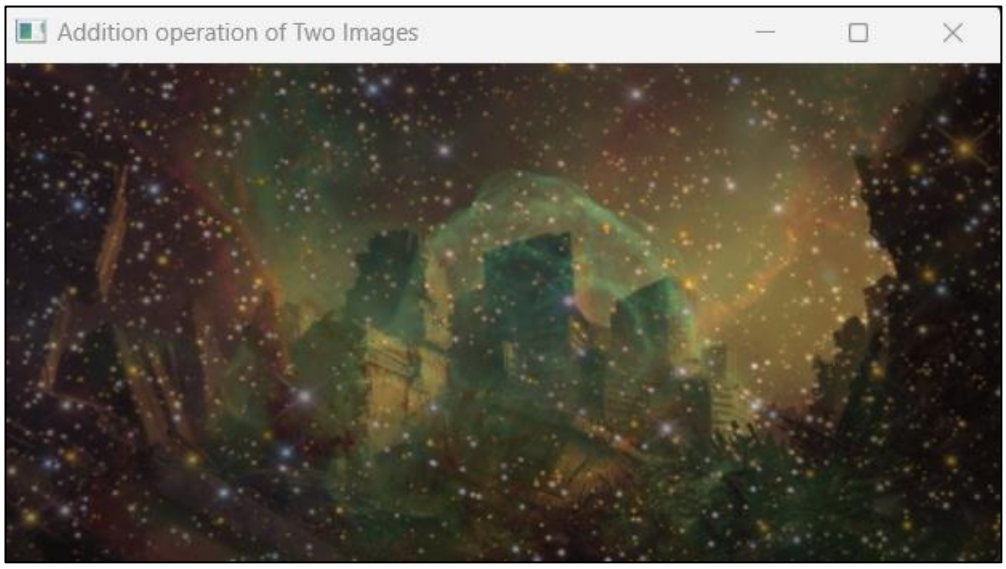
```
import cv2

img = cv2.imread("C:/Users/Dell/Downloads/modi.jpeg")

resized_img = cv2.resize(img, (600, 300))

rotation_matrix = cv2.getRotationMatrix2D((img.shape[1]/2,
img.shape[0]/2), 30, 1)

rotated_img = cv2.warpAffine(img, rotation_matrix, (img.shape[1],
img.shape[0]))


cv2.imshow("Resized Image", resized_img)

cv2.imshow("Rotated Image", rotated_img)

cv2.waitKey(0)
```

OUTPUT:

SOURCE CODE:

```
import cv2

image_one = cv2.imread("C:/Users/Dell/Downloads/img1.jpg")
image_two = cv2.imread("C:/Users/Dell/Downloads/img2.jpg")
result_image = cv2.addWeighted(image_one, 0.5, image_two, 0.5, 0)
cv2.imshow('Addition operation of Two Images', result_image)
cv2.waitKey(0)
```

OUTPUT:



SOURCE CODE:

```
import cv2

import numpy as np

from matplotlib import pyplot as plt


# Mean Filtering

image = cv2.imread('C:/Users/Dell/Downloads/71.jpg')

new_image = cv2.blur(image,(9, 9))

plt.subplot(121), plt.imshow(cv2.cvtColor(image,
cv2.COLOR_BGR2RGB)),plt.title('Original')

plt.subplot(122), plt.imshow(cv2.cvtColor(new_image,
cv2.COLOR_BGR2RGB)),plt.title('Mean Filter')

plt.show()


# Gaussian Filtering

img = cv2.imread("C:/Users/Dell/Downloads/nature.jpeg")

dst = cv2.GaussianBlur(img,(9,9),cv2.BORDER_REFLECT_101)

cv2.imshow('Gaussian Blur Image', np.hstack((img, dst)))

cv2.waitKey(0)

cv2.destroyAllWindows()
```

OUTPUT:

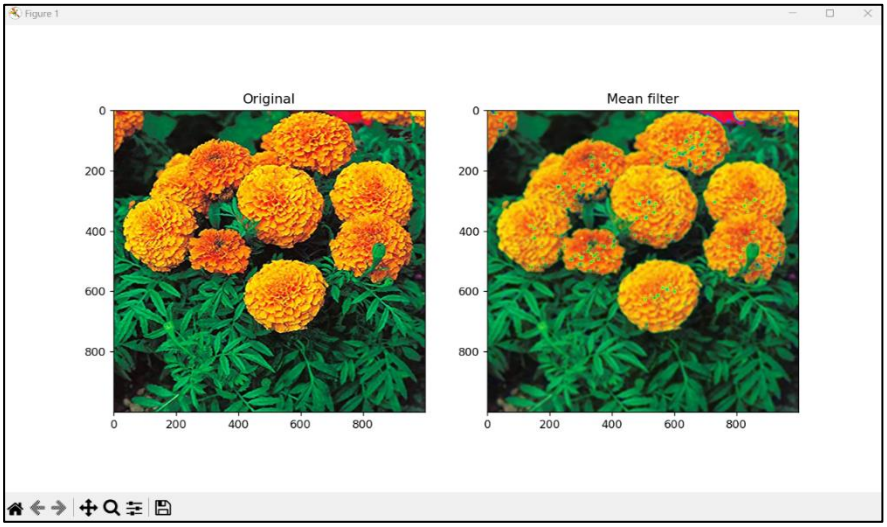


Figure 1
Original
Mean filter
0
200
400
600
800


Gaussion Blur Image

SOURCE CODE:

```
import cv2


im = cv2.imread("C:/Users/Dell/Downloads/im2.jpeg")
img = cv2.cvtColor(im, cv2.COLOR_BGR2GRAY)
ret, thresh1 = cv2.threshold(img, 120, 255, cv2.THRESH_BINARY)
cv2.imshow('Binary Threshold', thresh1)
cv2.waitKey(0)
```

OUTPUT:

SOURCE CODE:

```
import cv2
import matplotlib.pyplot as plt

imageObj = cv2.imread('C:/Users/Dell/Downloads/im2.jpeg')
plt.axis("off")
plt.title("Original Image")
plt.imshow(cv2.cvtColor(imageObj, cv2.COLOR_BGR2RGB))
plt.show()

blue_color = cv2.calcHist([imageObj], [0], None, [256], [0, 256])
red_color = cv2.calcHist([imageObj], [1], None, [256], [0, 256])
green_color = cv2.calcHist([imageObj], [2], None, [256], [0, 256])
plt.title("Histogram of RGB Colors")
plt.hist(blue_color, color="blue")
plt.hist(green_color, color="green")
plt.hist(red_color, color="red")
plt.show()
```

OUTPUT:

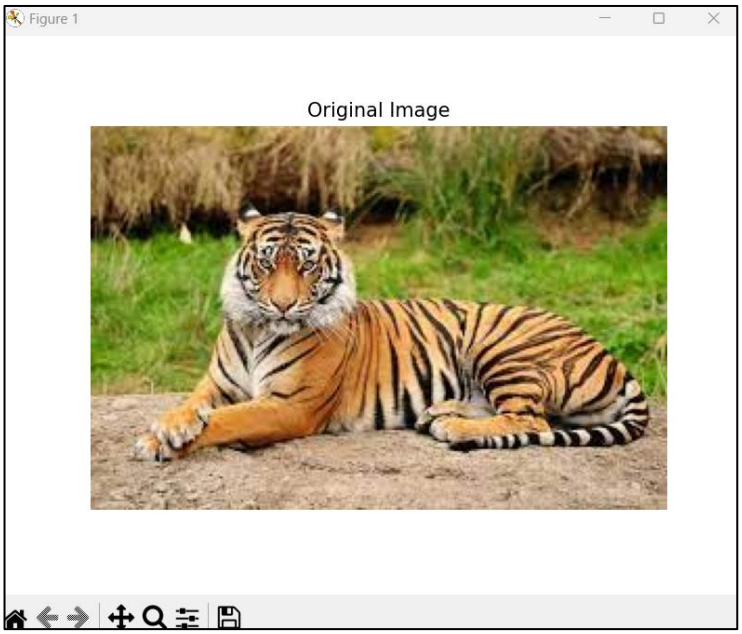


Figure 1
Original Image

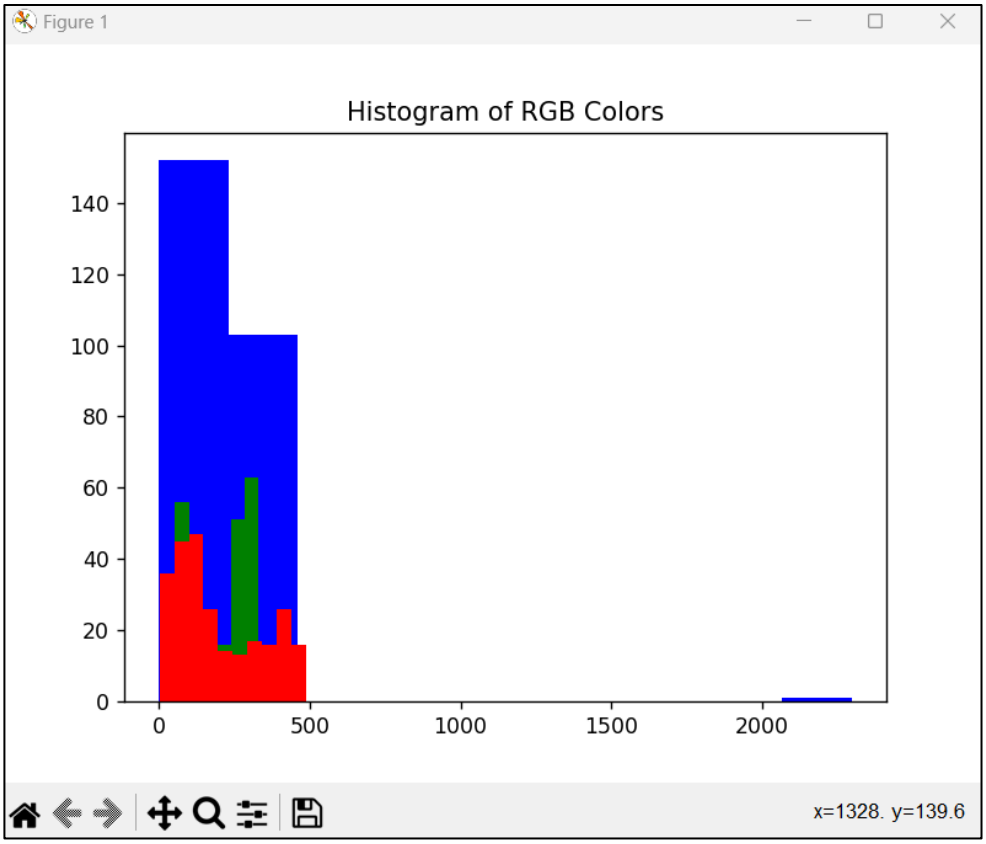


Figure 1
Histogram of RGB Colors
140
120
100
80
60
40
20
0
0
500
1000
1500
2000
x=1328. y=139.6

SOURCE CODE:

```
import numpy as np
import cv2 as cv

img = cv.imread('C:/Users/Dell/Downloads/ig2.jpg')
Z = img.reshape((-1,3))
Z = np.float32(Z)
criteria = (cv.TERM_CRITERIA_EPS +
cv.TERM_CRITERIA_MAX_ITER, 10, 1.0)
K = 8
ret,label,center=cv.kmeans(Z,K,None,criteria,10,cv.KMEANS_RANDOM_CENTERS)
center = np.uint8(center)
res = center[label.flatten()]
res2 = res.reshape((img.shape))
cv.imshow('Image using K-means Cluster',res2)
cv.waitKey(0)
cv.destroyAllWindows()
```

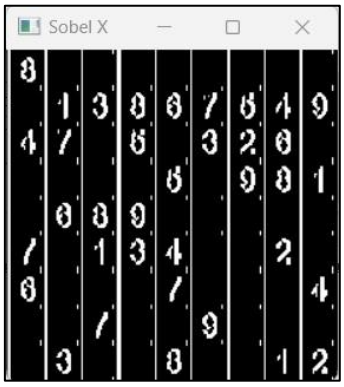
OUTPUT:

SOURCE CODE:

```
import cv2

img = cv2.imread('C:/Users/Dell/Downloads/sudo.png',
cv2.IMREAD_GRAYSCALE)

sobelx = cv2.Sobel(img, cv2.CV_64F, 1, 0, ksize=3)
sobely = cv2.Sobel(img, cv2.CV_64F, 0, 1, ksize=3)

scharrx = cv2.Scharr(img, cv2.CV_64F, 1, 0)
scharry = cv2.Scharr(img, cv2.CV_64F, 0, 1)

cv2.imshow('Original', img)
cv2.imshow('Sobel X', sobelx)
cv2.imshow('Sobel Y', sobely)
cv2.imshow('Scharr X', scharrx)
cv2.imshow('Scharr Y', scharry)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

OUTPUT:

Sobel



Scharr

SOURCE CODE:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import time
from sklearn.datasets import load_digits

digits = load_digits()
print(digits.keys())
print('Label Data Shape', digits.target.shape)

X = digits.images

from sklearn.metrics import accuracy_score, confusion_matrix
from sklearn.model_selection import train_test_split
from sklearn.multiclass import OneVsRestClassifier
from sklearn.neighbors import KNeighborsClassifier
import seaborn as sns

X = digits.data
y = digits.target

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25,
random_state=0)
knn = OneVsRestClassifier(KNeighborsClassifier())
knn.fit(X_train, y_train)
predictions = knn.predict(X_test)
print('KNN Accuracy: %.3f' % accuracy_score(y_test, predictions))

cm = confusion_matrix(y_test, predictions)
plt.figure(figsize=(9,9))
sns.heatmap(cm, annot=True, fmt='.3f', linewidths=.5, square=True,
cmap='Blues_r')
plt.ylabel('Actual label')
plt.xlabel('Predicted label')
all_sample_title='Accuracy_Score:{0}'.format(accuracy_score(y_test,pre
dictions))
plt.title(all_sample_title, size=15)
```

OUTPUT:

```
dict_keys(['data', 'target', 'frame', 'feature_names', 'target_names', 'images', 'DESCR'])
Label Data Shape (1797,)
KNN Accuracy: 0.980
```