ORMEDIAN RESEARCH INSTITUTE

TOPIC: : OpenCV Tutorials (part six)

By:

ADEKOLA OLUWASEUN O.

February 23, 2023

TABLE OF CONTENT

COLOUR SPACES

Colour space in computer vision generally refers to the space of colours. It is a system of representing the array of pixels colours.

There are several colour spaces commonly recognized in computer vision. Some of which are:

- Grayscale
- HSV(Hue Saturation Value)
- LAB
- RGB

COLOUR SPACES Cont'd

Coversion of BGR to Gray

This conversion is usually carried out when there is a need to show the pixel intensity distribution at a particular locations in the image.

SYNTAX

```
import cv2 as cv
img = cv.imread('Images/friends.jpg')
gray = cv.cvtColor(img,
cv.COLOR_BGR2GRAY)
cv.imshow('ORIGINAL IMAGE', img)
cv.imshow('GRAY OUTPUT', gray)
cv.waitKey(0)
```

OUTPUT

File Edit Selection View Go Run Terminal Help — gray_spaces.py - CV - Visual Studio Code

EXPLORER

OPEN EDITORS 5 unsaved

CV

- eroded.py
- flip.py
- gray_spaces.py
- Image_reading.py
- image_rescaling.py
- image_resize.py
- img_rotate.py
- img_rotate2.py
- img_translate.py
- line.py
- more_blur.py
- opencv_frame_10.png
- paint_blank.py
- read.py
- read2.py
- rectangle.py
- rectangle1.py
- Threshcont.py
- triangle.py
- triangle1.py
- video_reading.py
- video_rescaling.py
- writing.py

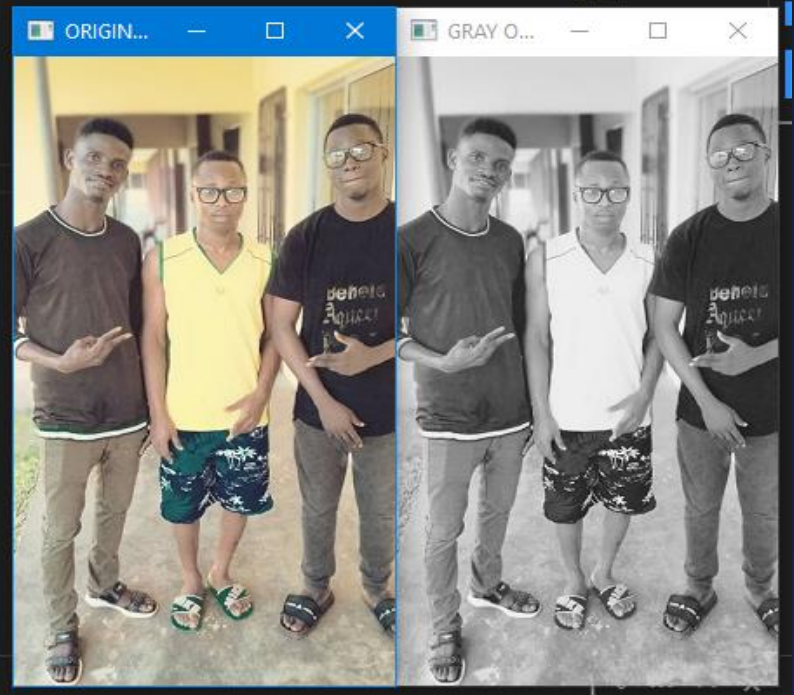
OUTLINE

TIMELINE

contreduced.py | Threshcont.py | drawContours.py | gray_spaces.py | rectangle1.py | opencv_frame_10.png | convers

gray_spaces.py > ...

```
1  import cv2 as cv
2  img = cv.imread('Images/friends.jpg')
3  gray = cv.cvtColor(img, cv.COLOR_BGR2GRAY)
4  cv.imshow('ORIGINAL IMAGE', img)
5  cv.imshow('GRAY OUTPUT', gray)
6  cv.waitKey(0)
7
```



PROBLEMS 22 OUTPUT TERMINAL DEBUG CONSOLE JUPYTER COMMENTS

```
PS C:\Users\SHOPINVERSE\Desktop\CV> python gray_spaces.py
```

powershell

Python

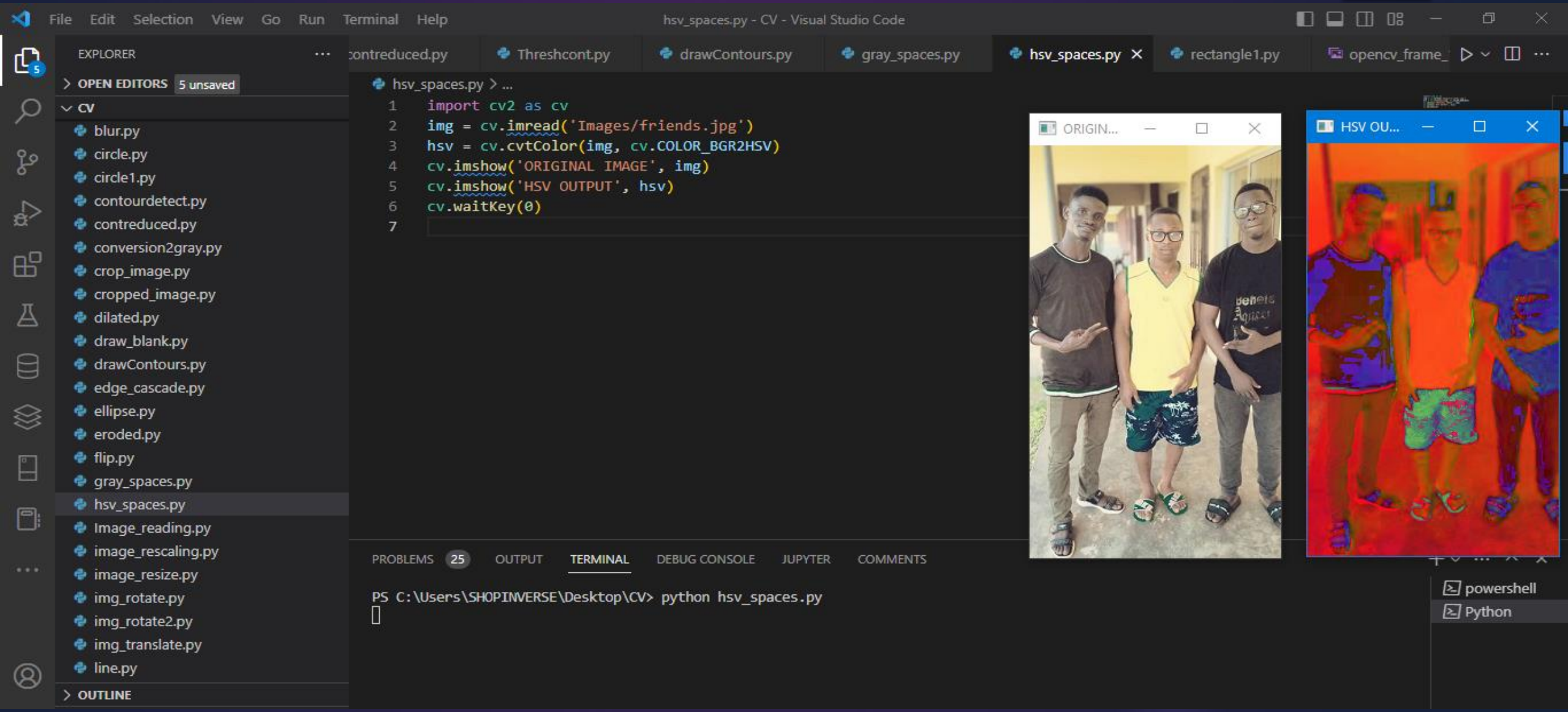
COLOUR SPACES Cont'd

Coversion of BGR Image to HSV Format

SYNTAX

```
import cv2 as cv
img = cv.imread('Images/friends.jpg')
gray = cv.cvtColor(img,
cv.COLOR_BGR2HSV)
cv.imshow('ORIGINAL IMAGE', img)
cv.imshow('HSV', hsv)
cv.waitKey(0)
```

OUTPUT


File Edit Selection View Go Run Terminal Help
hsv_spaces.py - CV - Visual Studio Code
EXPLORER
OPEN EDITORS 5 unsaved
CV
blur.py
circle.py
circle1.py
contourdetect.py
contreduced.py
conversion2gray.py
crop_image.py
cropped_image.py
dilated.py
draw_blank.py
drawContours.py
edge_cascade.py
ellipse.py
eroded.py
flip.py
gray_spaces.py
hsv_spaces.py
Image_reading.py
image_rescaling.py
image_resize.py
img_rotate.py
img_rotate2.py
img_translate.py
line.py
OUTLINE
contreduced.py
Threshcont.py
drawContours.py
gray_spaces.py
hsv_spaces.py
rectangle1.py
opencv_frame_
hsv_spaces.py > ...
1 import cv2 as cv
2 img = cv.imread('Images/friends.jpg')
3 hsv = cv.cvtColor(img, cv.COLOR_BGR2HSV)
4 cv.imshow('ORIGINAL IMAGE', img)
5 cv.imshow('HSV OUTPUT', hsv)
6 cv.waitKey(0)
7
PROBLEMS 25 OUTPUT TERMINAL DEBUG CONSOLE JUPYTER COMMENTS
PS C:\Users\SHOPINVERSE\Desktop\CV> python hsv_spaces.py
ORIGIN...
HSV OU...
powershell
Python

COLOUR SPACES Cont'd

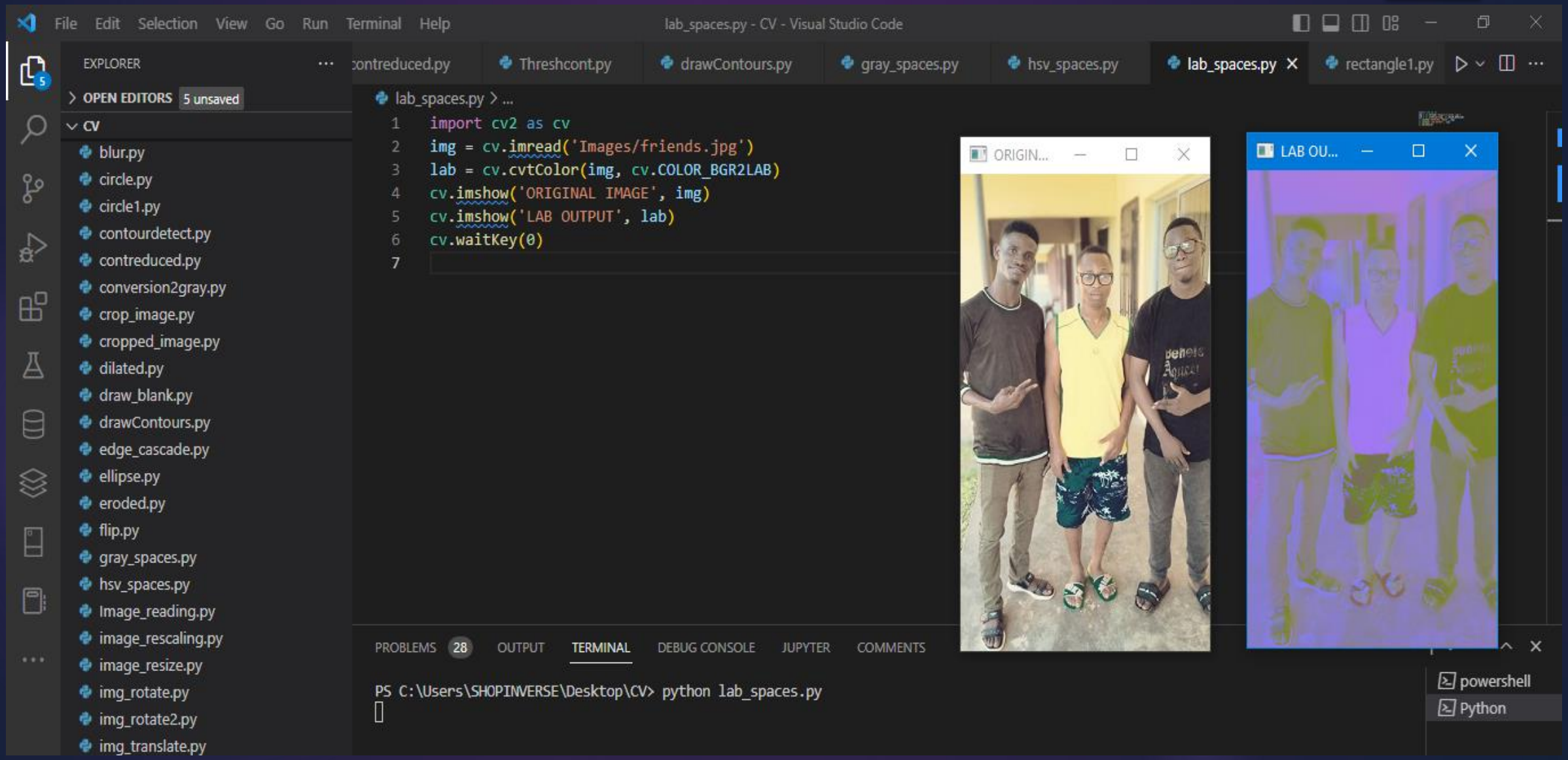
Coversion of BGR image to LAB format

LAB version of the BGR image looks like a the washed-down version of the BGR

SYNTAX

```
import cv2 as cv
img = cv.imread('Images/friends.jpg')
lab = cv.cvtColor(img,
cv.COLOR_BGR2LAB)
cv.imshow('ORIGINAL IMAGE', img)
cv.imshow('LAB OUTPUT', lab)
cv.waitKey(0)
```

OUTPUT


File Edit Selection View Go Run Terminal Help
lab_spaces.py - CV - Visual Studio Code
EXPLORER
OPEN EDITORS 5 unsaved
CV
blur.py
circle.py
circle1.py
contourdetect.py
contreduced.py
conversion2gray.py
crop_image.py
cropped_image.py
dilated.py
draw_blank.py
drawContours.py
edge_cascade.py
ellipse.py
eroded.py
flip.py
gray_spaces.py
hsv_spaces.py
Image_reading.py
image_rescaling.py
image_resize.py
img_rotate.py
img_rotate2.py
img_translate.py
contreduced.py
Threshcont.py
drawContours.py
gray_spaces.py
hsv_spaces.py
lab_spaces.py
rectangle1.py
lab_spaces.py > ...
1 import cv2 as cv
2 img = cv.imread('Images/friends.jpg')
3 lab = cv.cvtColor(img, cv.COLOR_BGR2LAB)
4 cv.imshow('ORIGINAL IMAGE', img)
5 cv.imshow('LAB OUTPUT', lab)
6 cv.waitKey(0)
7
ORIGIN...
LAB OU...
PROBLEMS 28 OUTPUT TERMINAL DEBUG CONSOLE JUPYTER COMMENTS
PS C:\Users\SHOPINVERSE\Desktop\CV> python lab_spaces.py
powershell
Python

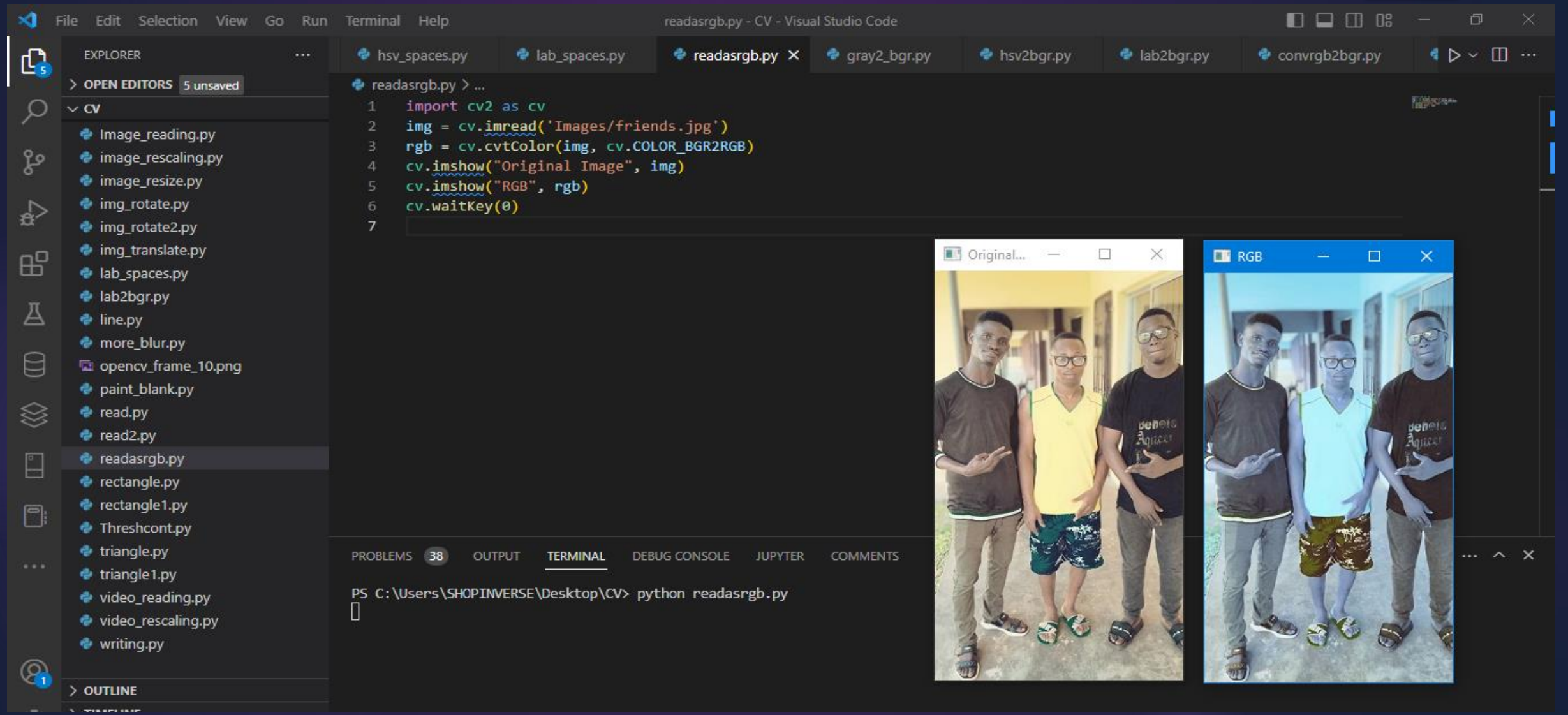
COLOUR SPACES Cont'd

Example: Convert an BGR Image to RGB in OpenCV

SOLUTION

```
import cv2 as cv
img = cv.imread('Images/friends.jpg')
rgb = cv.cvtColor(img, cv.COLOR_BGR2RGB)
cv.imshow("Original Image", img)
cv.imshow("RGB", rgb)
cv.waitKey(0)
```

OUTPUT


File Edit Selection View Go Run Terminal Help
readasrgb.py - CV - Visual Studio Code
EXPLORER
OPEN EDITORS 5 unsaved
CV
Image_reading.py
image_rescaling.py
image_resize.py
img_rotate.py
img_rotate2.py
img_translate.py
lab_spaces.py
lab2bgr.py
line.py
more_blur.py
opencv_frame_10.png
paint_blank.py
read.py
read2.py
readasrgb.py
rectangle.py
rectangle1.py
Threshcont.py
triangle.py
triangle1.py
video_reading.py
video_rescaling.py
writing.py
OUTLINE
hsv_spaces.py
lab_spaces.py
readasrgb.py
gray2_bgr.py
hsv2bgr.py
lab2bgr.py
convrgb2bgr.py
readasrgb.py > ...
1 import cv2 as cv
2 img = cv.imread('Images/friends.jpg')
3 rgb = cv.cvtColor(img, cv.COLOR_BGR2RGB)
4 cv.imshow("Original Image", img)
5 cv.imshow("RGB", rgb)
6 cv.waitKey(0)
7
PROBLEMS 38
OUTPUT
TERMINAL
DEBUG CONSOLE
JUPYTER
COMMENTS
PS C:\Users\SHOPINVERSE\Desktop\CV> python readasrgb.py
Original...
RGB

COLOUR SPACES Cont'd

UNDERSTANDING BGR AND RGB IMAGES

In OpenCV, images are read in BGR format. But outside of OpenCV, RGB format is usually used. For example, other libraries like the matplotlib reads images in RGB format, which is the inversion of the BGR format. Therefore, it is very important to keep in mind that, by default, different libraries read images differently.

COLOUR SPACES Cont'd

Conversion of Images from other Formats to BGR

- Conversion from RGB format to BGR format
- Conversion from GRAY format to BGR format
- Conversion from HSV format to BGR format
- Conversion from LAB format to BGR format

The downside of these conversions is that, to convert from the grayscale format to HSV format directly, such an image must first be converted to BGR before it can be converted to HSV.

COLOUR SPACES Cont'd

Example: Convert a RGB version of friends.jpg to BGR format.

SOLUTION

```
import cv2 as cv
img = cv.imread('Images/friends.jpg')
rgb = cv.cvtColor(img, cv.COLOR_BGR2RGB)
bgr = cv.cvtColor(rgb, cv.COLOR_RGB2BGR)
cv.imshow("RGB", rgb)
cv.imshow("BGR", img)
cv.waitKey(0)
```

OUTPUT

File Edit Selection View Go Run Terminal Help — convrgb2bgr.py - CV - Visual Studio Code

EXPLORER

OPEN EDITORS 5 unsaved

CV

- blur.py
- circle.py
- circle1.py
- contourdetect.py
- contreduced.py
- conversion2gray.py
- convrgb2bgr.py
- crop_image.py
- cropped_image.py
- dilated.py
- draw_blank.py
- drawContours.py
- edge_cascade.py
- ellipse.py
- eroded.py
- flip.py
- gray_spaces.py
- gray2_bgr.py
- hsv_spaces.py
- hsv2bgr.py
- Image_reading.py
- image_rescaling.py
- image_resize.py
- img_rotate.py

OUTLINE

hsv_spaces.py | lab_spaces.py | readasrgb.py 1 | gray2_bgr.py | hsv2bgr.py | lab2bgr.py | convrgb2bgr.py

convrgb2bgr.py > ...

```
1  import cv2 as cv
2  img = cv.imread('Images/friends.jpg')
3  rgb = cv.cvtColor(img, cv.COLOR_BGR2RGB)
4  bgr = cv.cvtColor(rgb, cv.COLOR_RGB2BGR)
5  cv.imshow("RGB", rgb)
6  cv.imshow("BGR", img)
7  cv.waitKey(0)
```

PROBLEMS 40 | OUTPUT | TERMINAL | DEBUG CONSOLE | JUPYTER | COMMENTS

```
PS C:\Users\SHOPINVERSE\Desktop\CV> python convrgb2bgr.py
```

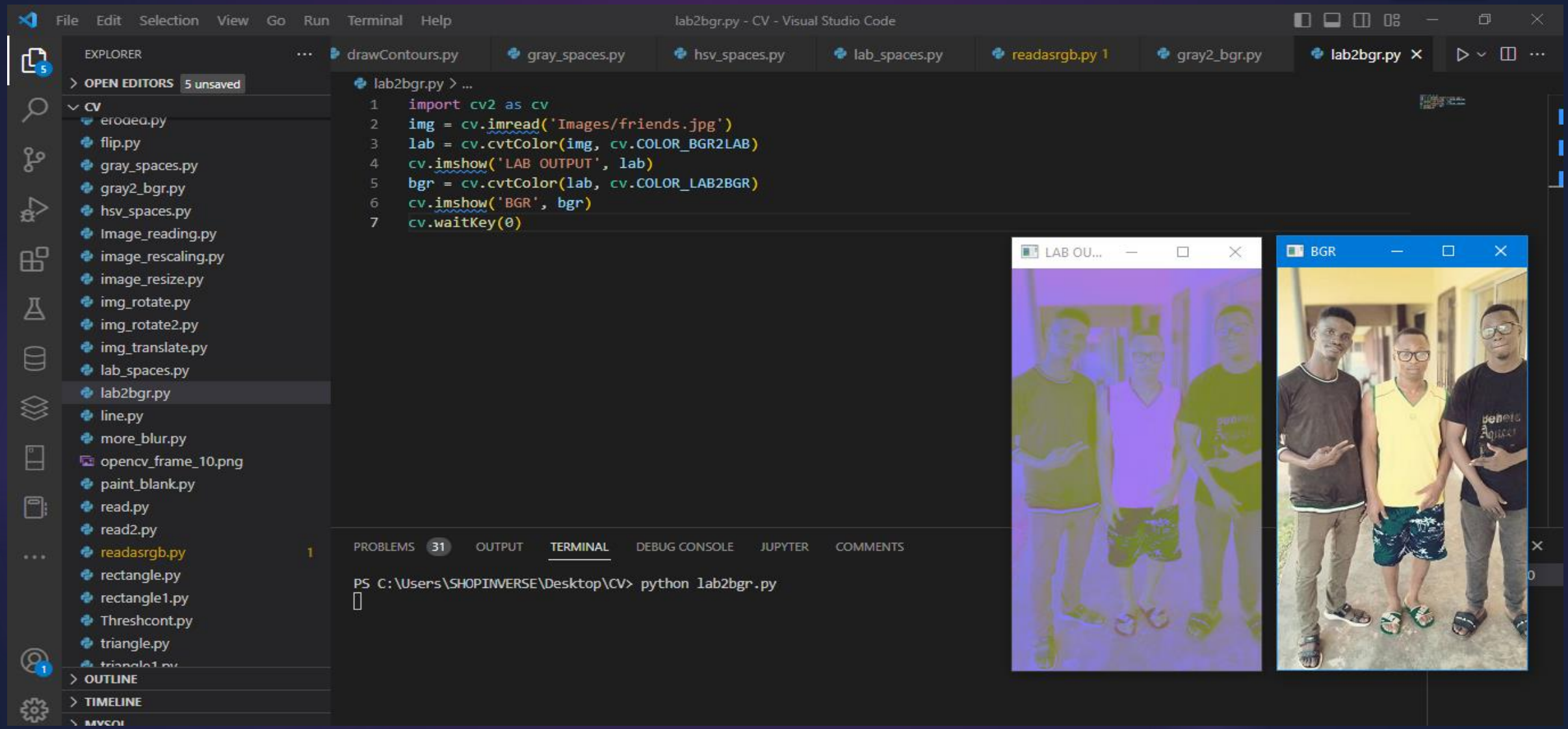
COLOUR SPACES Cont'd

Example: Convert a LAB version of friends.jpg to BGR format.

SOLUTION

```
import cv2 as cv
img = cv.imread('Images/friends.jpg')
lab = cv.cvtColor(img, cv.COLOR_BGR2LAB)
cv.imshow('LAB OUTPUT', lab)
bgr = cv.cvtColor(lab, cv.COLOR_LAB2BGR)
cv.imshow('BGR', bgr)
cv.waitKey(0)
```

OUTPUT


File Edit Selection View Go Run Terminal Help
lab2bgr.py - CV - Visual Studio Code
EXPLORER
drawContours.py
gray_spaces.py
hsv_spaces.py
lab_spaces.py
readasrgb.py 1
gray2_bgr.py
lab2bgr.py
OPEN EDITORS 5 unsaved
CV
flip.py
gray_spaces.py
gray2_bgr.py
hsv_spaces.py
Image_reading.py
image_rescaling.py
image_resize.py
img_rotate.py
img_rotate2.py
img_translate.py
lab_spaces.py
lab2bgr.py
line.py
more_blur.py
opencv_frame_10.png
paint_blank.py
read.py
read2.py
readasrgb.py 1
rectangle.py
rectangle1.py
Threshcont.py
triangle.py
OUTLINE
TIMELINE
lab2bgr.py > ...
1 import cv2 as cv
2 img = cv.imread('Images/friends.jpg')
3 lab = cv.cvtColor(img, cv.COLOR_BGR2LAB)
4 cv.imshow('LAB OUTPUT', lab)
5 bgr = cv.cvtColor(lab, cv.COLOR_LAB2BGR)
6 cv.imshow('BGR', bgr)
7 cv.waitKey(0)
PROBLEMS 31 OUTPUT TERMINAL DEBUG CONSOLE JUPYTER COMMENTS
PS C:\Users\SHOPINVERSE\Desktop\CV> python lab2bgr.py
LAB OU...
BGR

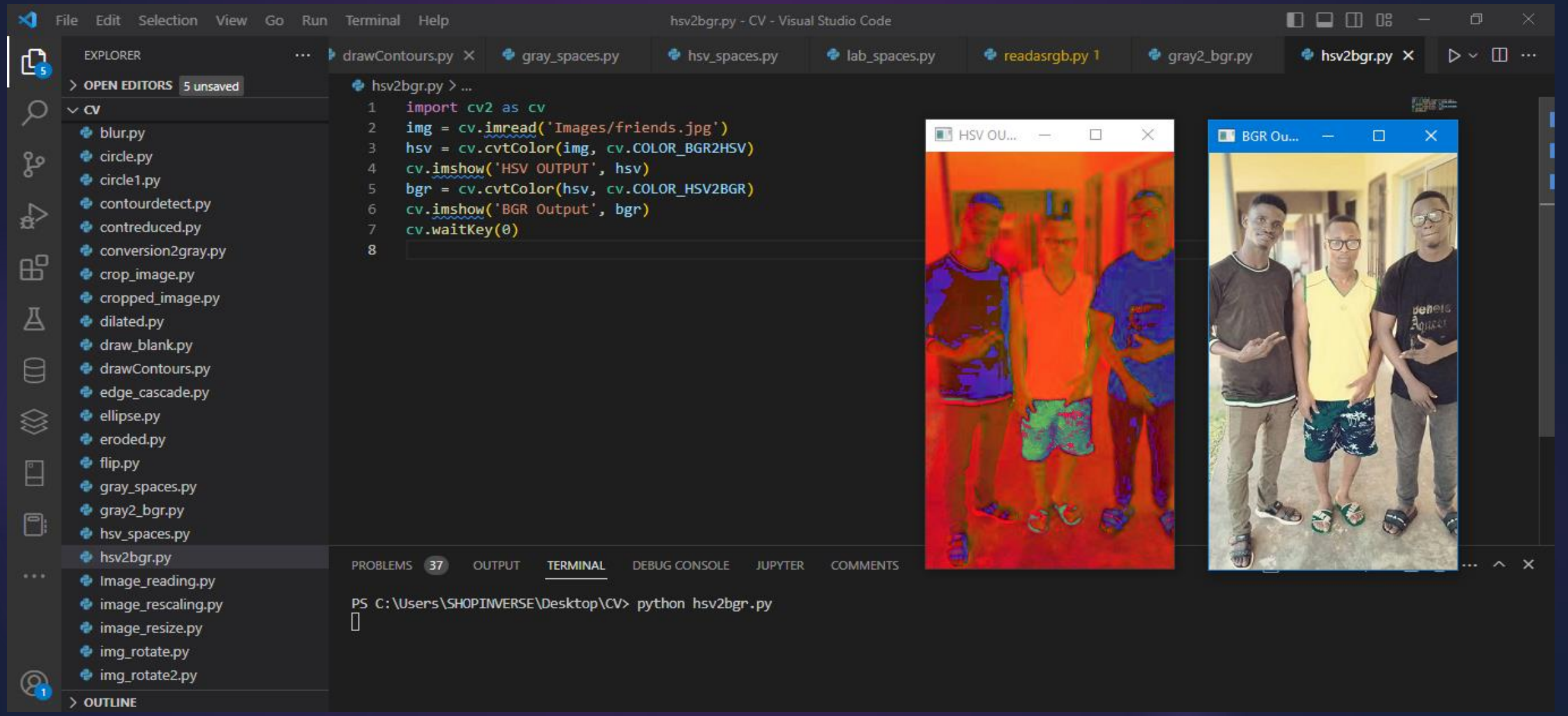
COLOUR SPACES Cont'd

Example: Convert a HSV version of "friends.jpg" to BGR format.

SOLUTION

```
import cv2 as cv
img = cv.imread('Images/friends.jpg')
hsv = cv.cvtColor(img, cv.COLOR_BGR2HSV)
cv.imshow('HSV OUTPUT', hsv)
bgr = cv.cvtColor(hsv, cv.COLOR_HSV2BGR)
cv.imshow('BGR Output', bgr)
cv.waitKey(0)
```

OUTPUT


File Edit Selection View Go Run Terminal Help
hsv2bgr.py - CV - Visual Studio Code
EXPLORER
drawContours.py
gray_spaces.py
hsv_spaces.py
lab_spaces.py
readasrgb.py 1
gray2_bgr.py
hsv2bgr.py
OPEN EDITORS 5 unsaved
CV
blur.py
circle.py
circle1.py
contourdetect.py
contreduced.py
conversion2gray.py
crop_image.py
cropped_image.py
dilated.py
draw_blank.py
drawContours.py
edge_cascade.py
ellipse.py
eroded.py
flip.py
gray_spaces.py
gray2_bgr.py
hsv_spaces.py
hsv2bgr.py
Image_reading.py
image_rescaling.py
image_resize.py
img_rotate.py
img_rotate2.py
OUTLINE
hsv2bgr.py > ...
1 import cv2 as cv
2 img = cv.imread('Images/friends.jpg')
3 hsv = cv.cvtColor(img, cv.COLOR_BGR2HSV)
4 cv.imshow('HSV OUTPUT', hsv)
5 bgr = cv.cvtColor(hsv, cv.COLOR_HSV2BGR)
6 cv.imshow('BGR Output', bgr)
7 cv.waitKey(0)
8
HSV OU...
BGR Ou...
PROBLEMS 37
OUTPUT
TERMINAL
DEBUG CONSOLE
JUPYTER
COMMENTS
PS C:\Users\SHOPINVERSE\Desktop\CV> python hsv2bgr.py

SPLITING AND MERGING COLOUR CHANNELS

SPLITTING OF IMAGE COLOUR CHANNELS

Colour Image consists of 3 colour channels be it RGB or BGR image. OpenCV has the functionality that allows us to split a colour image into its respective components of red, green, and blue colour. To achieve the splitting operation in OpenCV, the cv2.split() method is used on such an image.

When colour is being split, they are depicted and displayed as grayscale images. This shows the distribution of pixel intensities, whereby the lighter region depicts a higher concentration of pixels, whereas the darker regions depicts that there is little or no pixel in such a region.

The grayscale copy of each component is one channel, which renders the grayscale a 2-D form of image unlike the RGB or BGR images that are 3-D. By verifying the shape of each component, we will see that the shape is displayed as [width, height], unlike BGR, which is displayed as [width, height, channels].

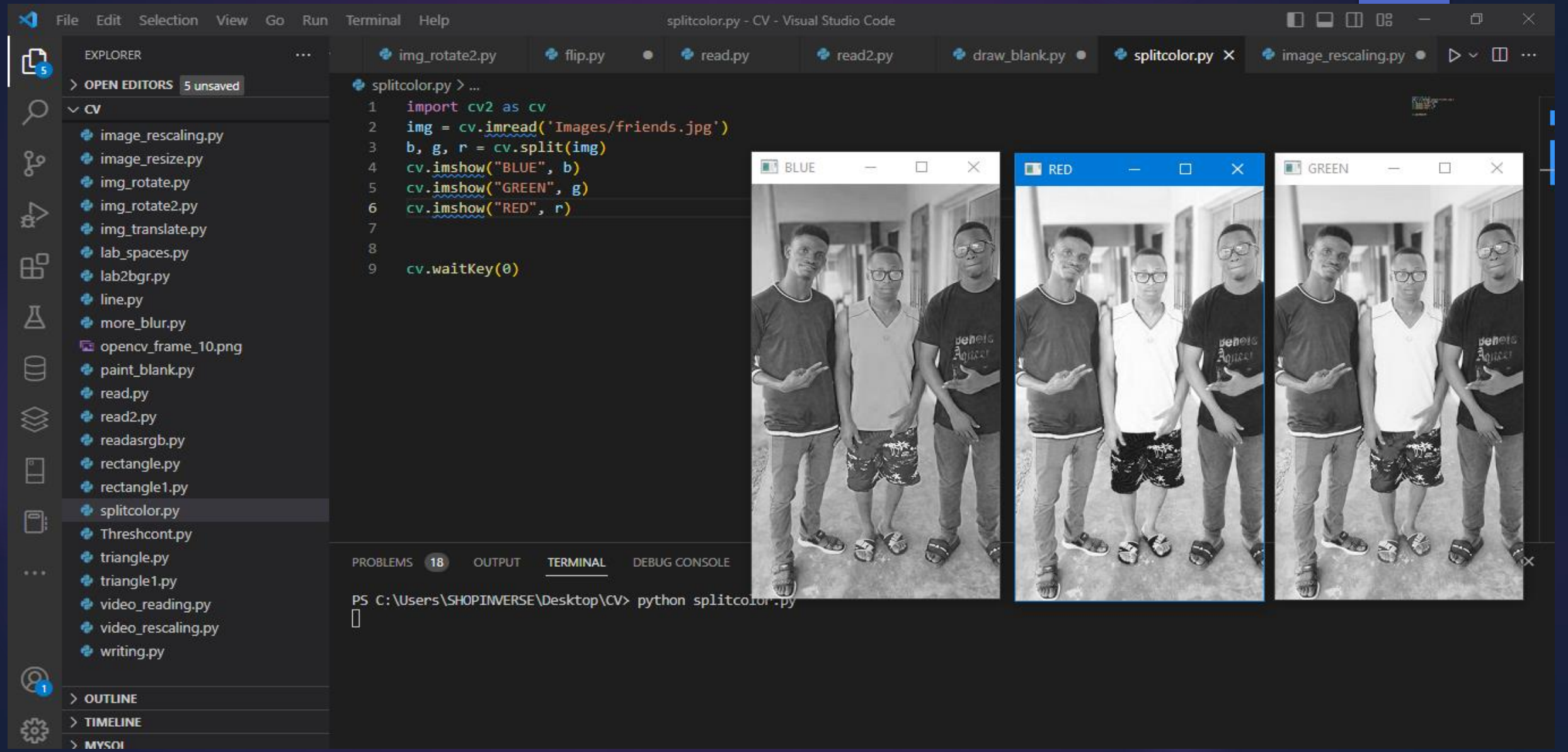
SPLITING AND MERGING COLOUR CHANNELS Cont'd

Example 1: Split the image "friends.jpg" to its colour channel components

SOLUTION

```
import cv2 as cv
img = cv.imread('Images/friends.jpg')
b, g, r = cv.split(img)
cv.imshow("BLUE", b)
cv.imshow("GREEN", g)
cv.imshow("RED", r)
cv.waitKey(0)
```

OUTPUT


File Edit Selection View Go Run Terminal Help
splitcolor.py - CV - Visual Studio Code
EXPLORER
OPEN EDITORS 5 unsaved
CV
image_rescaling.py
image_resize.py
img_rotate.py
img_rotate2.py
img_translate.py
lab_spaces.py
lab2bgr.py
line.py
more_blur.py
opencv_frame_10.png
paint_blank.py
read.py
read2.py
readasrgb.py
rectangle.py
rectangle1.py
splitcolor.py
Threshcont.py
triangle.py
triangle1.py
video_reading.py
video_rescaling.py
writing.py
OUTLINE
TIMELINE
MYSQL
img_rotate2.py
flip.py
read.py
read2.py
draw_blank.py
splitcolor.py
image_rescaling.py
splitcolor.py > ...
1 import cv2 as cv
2 img = cv.imread('Images/friends.jpg')
3 b, g, r = cv.split(img)
4 cv.imshow("BLUE", b)
5 cv.imshow("GREEN", g)
6 cv.imshow("RED", r)
7
8
9 cv.waitKey(0)
BLUE
RED
GREEN
PROBLEMS 18
OUTPUT
TERMINAL
DEBUG CONSOLE
PS C:\Users\SHOPINVERSE\Desktop\CV> python splitcolor.py

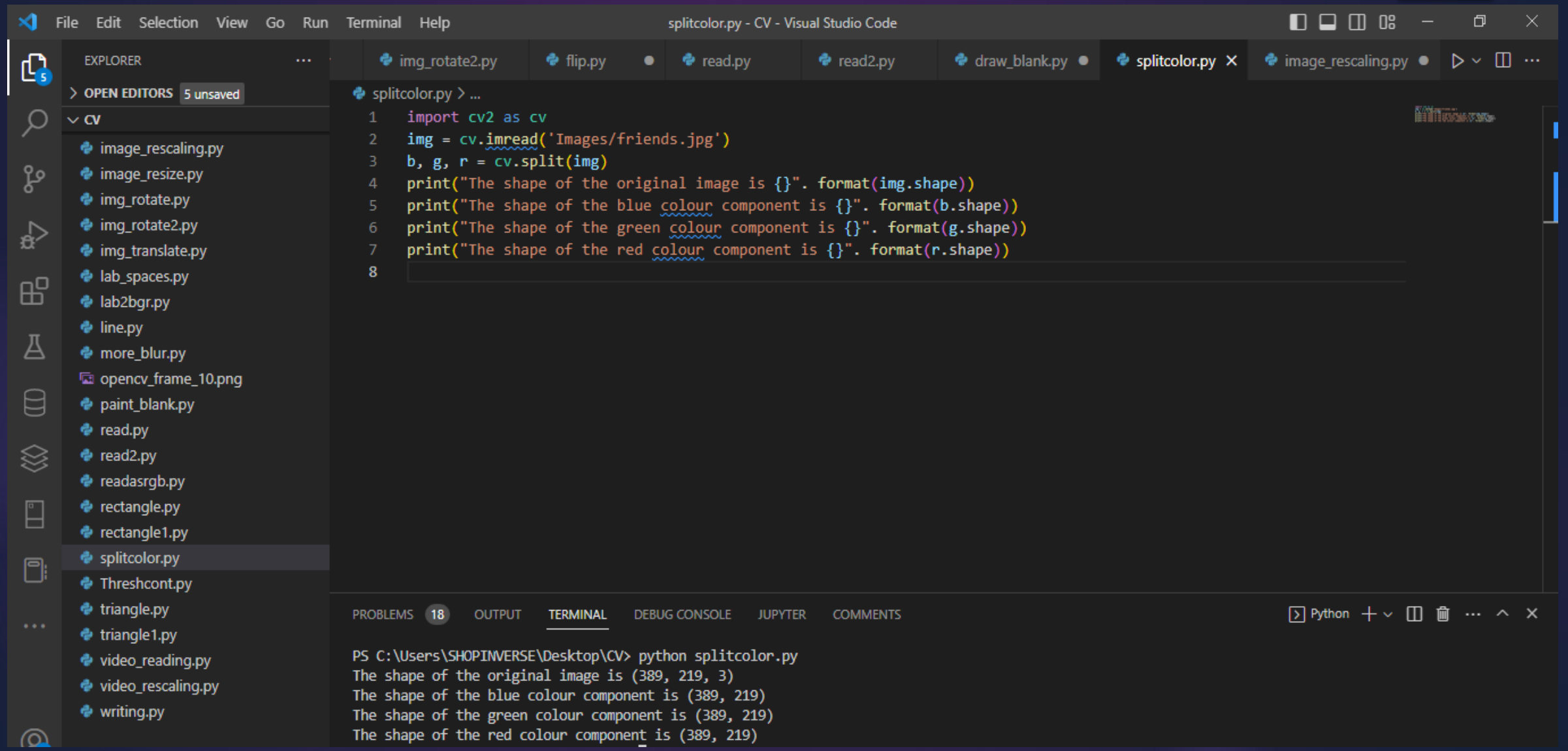
SPLITING AND MERGING COLOUR CHANNELS Cont'd

Displaying the shapes of colour components of the BGR image

Example 2: Consider the "friends.jpg" image, find the shape of the BGR image and the shape of its colour components.

```
import cv2 as cv
img = cv.imread('Images/friends.jpg')
b, g, r = cv.split(img)
print("The shape of the original image is {}". format(img.shape))
print("The shape of the blue colour component is {}". format(b.shape))
print("The shape of the green colour component is {}". format(g.shape))
print("The shape of the red colour component is {}". format(r.shape))
```

OUTPUT


File Edit Selection View Go Run Terminal Help
splitcolor.py - CV - Visual Studio Code
EXPLORER
OPEN EDITORS 5 unsaved
CV
image_rescaling.py
image_resize.py
img_rotate.py
img_rotate2.py
img_translate.py
lab_spaces.py
lab2bgr.py
line.py
more_blur.py
opencv_frame_10.png
paint_blank.py
read.py
read2.py
readasrgb.py
rectangle.py
rectangle1.py
splitcolor.py
Threshcont.py
triangle.py
triangle1.py
video_reading.py
video_rescaling.py
writing.py
img_rotate2.py
flip.py
read.py
read2.py
draw_blank.py
splitcolor.py
image_rescaling.py
splitcolor.py > ...
1 import cv2 as cv
2 img = cv.imread('Images/friends.jpg')
3 b, g, r = cv.split(img)
4 print("The shape of the original image is {}". format(img.shape))
5 print("The shape of the blue colour component is {}". format(b.shape))
6 print("The shape of the green colour component is {}". format(g.shape))
7 print("The shape of the red colour component is {}". format(r.shape))
8
PROBLEMS 18 OUTPUT TERMINAL DEBUG CONSOLE JUPYTER COMMENTS
Python
PS C:\Users\SHOPINVERSE\Desktop\CV> python splitcolor.py
The shape of the original image is (389, 219, 3)
The shape of the blue colour component is (389, 219)
The shape of the green colour component is (389, 219)
The shape of the red colour component is (389, 219)

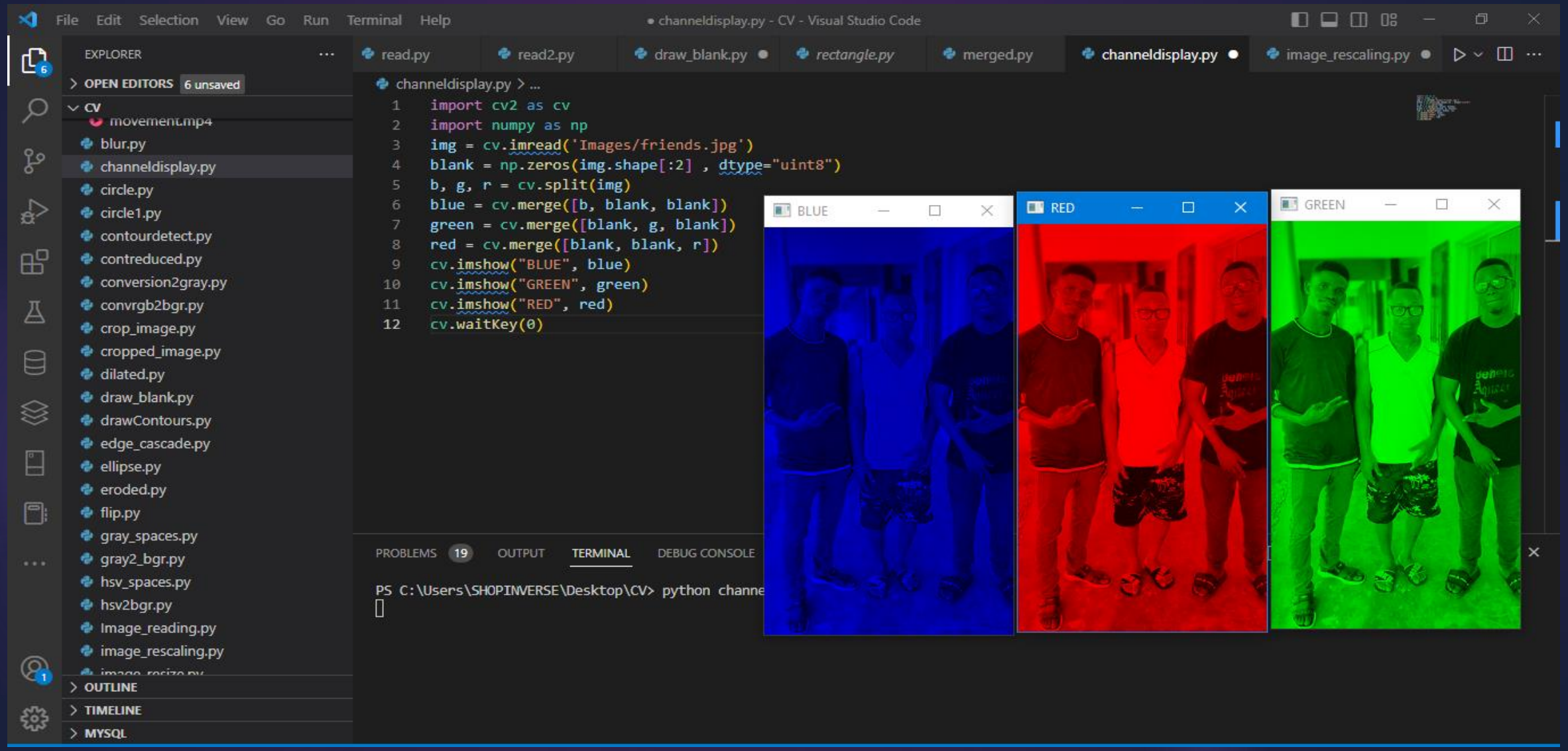
SPLITING AND MERGING COLOUR CHANNELS Cont'd

Individual colour channel output as a grayscale image can also be merged with others in a more strategic format to get the actual colour that the channel represents. This merging technique can be realized by setting other channels blank while the actual colour component to be outputted is not made blank.

SYNTAX

```
import cv2 as cv
import numpy as np
img = cv.imread('Images/friends.jpg')
blank = np.zeros(img.shape[:2] , dtype="uint8")
b, g, r = cv.split(img)
blue = cv.merge([b, blank, blank])
green = cv.merge([blank, g, blank])
red = cv.merge([blank, blank, r])
cv.imshow("BLUE", blue)
cv.imshow("GREEN", green)
cv.imshow("RED", red)
cv.waitKey(0)
```

OUTPUT


File Edit Selection View Go Run Terminal Help
• channeldisplay.py - CV - Visual Studio Code
EXPLORER
OPEN EDITORS 6 unsaved
CV
movement.mp4
blur.py
channeldisplay.py
circle.py
circle1.py
contourdetect.py
contreduced.py
conversion2gray.py
convrgb2bgr.py
crop_image.py
cropped_image.py
dilated.py
draw_blank.py
drawContours.py
edge_cascade.py
ellipse.py
eroded.py
flip.py
gray_spaces.py
gray2_bgr.py
hsv_spaces.py
hsv2bgr.py
Image_reading.py
image_rescaling.py
OUTLINE
TIMELINE
MYSQL
read.py
read2.py
draw_blank.py
rectangle.py
merged.py
channeldisplay.py
image_rescaling.py
channeldisplay.py > ...
1 import cv2 as cv
2 import numpy as np
3 img = cv.imread('Images/friends.jpg')
4 blank = np.zeros(img.shape[:2] , dtype="uint8")
5 b, g, r = cv.split(img)
6 blue = cv.merge([b, blank, blank])
7 green = cv.merge([blank, g, blank])
8 red = cv.merge([blank, blank, r])
9 cv.imshow("BLUE", blue)
10 cv.imshow("GREEN", green)
11 cv.imshow("RED", red)
12 cv.waitKey(0)
PROBLEMS 19 OUTPUT TERMINAL DEBUG CONSOLE
PS C:\Users\SHOPINVERSE\Desktop\CV> python channe
BLUE
RED
GREEN

SPLITING AND MERGING COLOUR CHANNELS

MERGING OF IMAGE COLOUR COMPONENTS

The merging of individual colour channels automatically gives us the colour combination of the original image. This means that the blue, green, and red colours are combined together to get the original image, from which they were initially separated.

Example 3: Consider merging colour components obtained in the previous example 2 together.

```
import cv2 as cv
img = cv.imread('Images/friends.jpg')
b, g, r = cv.split(img)
merged = cv.merge([b, g, r])
cv.imshow("MERGED IMAGE", merged)
cv.waitKey(0)
```

OUTPUT

File Edit Selection View Go Run Terminal Help — merged.py - CV - Visual Studio Code

EXPLORER

OPEN EDITORS 5 unsaved

CV

- gray2_bgr.py
- hsv_spaces.py
- hsv2bgr.py
- Image_reading.py
- image_rescaling.py
- image_resize.py
- img_rotate.py
- img_rotate2.py
- img_translate.py
- lab_spaces.py
- lab2bgr.py
- line.py
- merged.py
- more_blur.py
- opencv_frame_10.png
- paint_blank.py
- read.py
- read2.py
- readasrgb.py
- rectangle.py
- rectangle1.py
- splitcolor.py
- Threshcont.py
- triangle.py
- triangle1.py

img_rotate2.py | flip.py | read.py | read2.py | draw_blank.py | splitcolor.py | merged.py | im

merged.py > ...

```
1  import cv2 as cv
2  img = cv.imread('Images/friends.jpg')
3  b, g, r = cv.split(img)
4  merged = cv.merge([b, g, r])
5  cv.imshow("MERGED IMAGE", merged)
6  cv.waitKey(0)
```

PROBLEMS 19 | OUTPUT | TERMINAL | DEBUG CONSOLE | JUPYTER | COMMENTS

```
PS C:\Users\SHOPINVERSE\Desktop\CV> python merged.py
```

SMOOTHING TECHNIQUES IN OpenCV

Images that are taken by cameras could be corrupted by the camera in one or more ways, maybe due to affected camera sensor or as a result of lighting from the camera. To get rid of this noise that could result from such scenarios, there is a need for smoothing of the image, which could be achieved by simply applying some smoothing techniques to the image to remove some noise from it.

In OpenCV, there are some common blurring techniques that can be applied:

- Averaging Blurring
- Median Blurring
- Bilateral Filtering
- Gaussian Blurring.

SMOOTHING TECHNIQUES IN OpenCV Cont'd

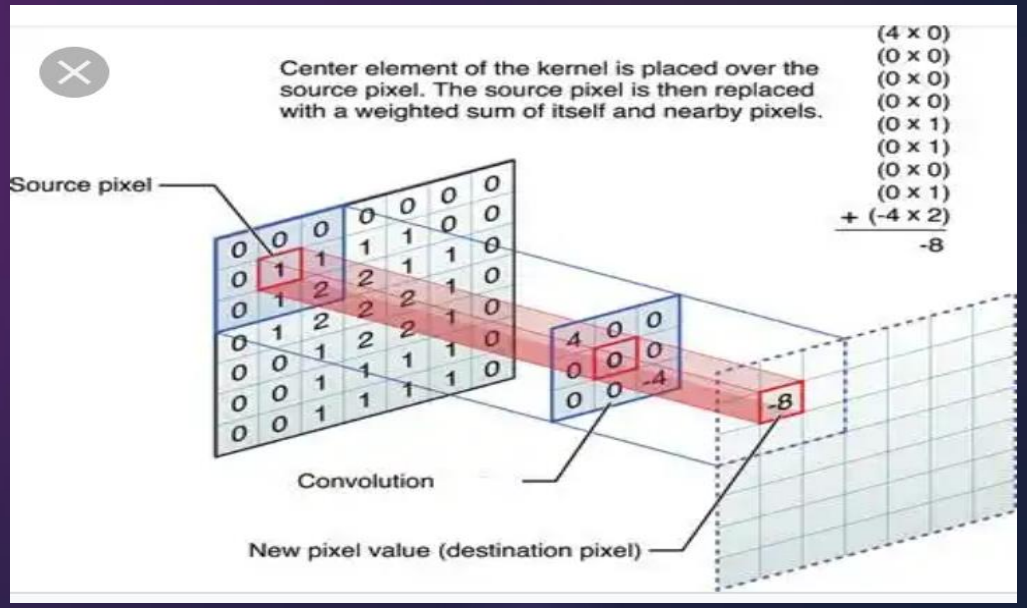
AVERAGING BLUR

This is a smoothing technique that utilizes a normalized box filter to convolve an image in such a way that the average of all pixels that are under the kernel area replaces the central element in the blurry image to be formed.

SYNTAX

cv2.blur() or cv2.boxFilter()

Averaging blur is shown pictorially.



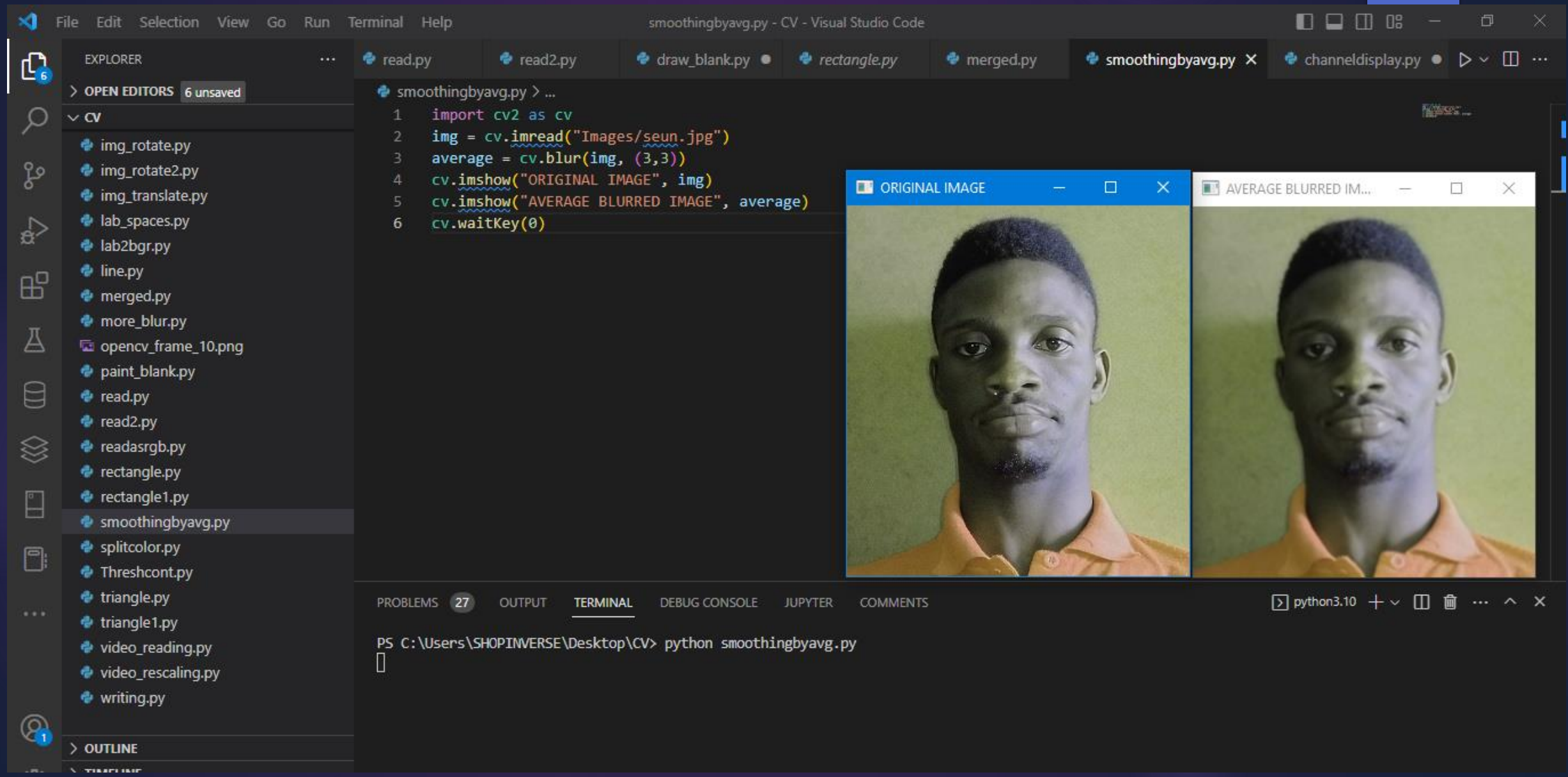
SMOOTHING TECHNIQUES IN OpenCV Cont'd

Example 1: Apply the averaging blur function to "seun.jpg" to smooth the image by using a 3 x 3 kernel size.

SOLUTION

```
import cv2 as cv
img = cv.imread("Images/seun.jpg")
average = cv.blur(img, (3,3))
cv.imshow("ORIGINAL IMAGE", img)
cv.imshow("AVERAGE BLURRED IMAGE", average)
cv.waitKey(0)
```

OUTPUT


File Edit Selection View Go Run Terminal Help
smoothingbyavg.py - CV - Visual Studio Code
EXPLORER
OPEN EDITORS 6 unsaved
CV
img_rotate.py
img_rotate2.py
img_translate.py
lab_spaces.py
lab2bgr.py
line.py
merged.py
more_blur.py
opencv_frame_10.png
paint_blank.py
read.py
read2.py
readasrgb.py
rectangle.py
rectangle1.py
smoothingbyavg.py
splitcolor.py
Threshcont.py
triangle.py
triangle1.py
video_reading.py
video_rescaling.py
writing.py
OUTLINE
read.py
read2.py
draw_blank.py
rectangle.py
merged.py
smoothingbyavg.py
channeldisplay.py
smoothingbyavg.py > ...
1 import cv2 as cv
2 img = cv.imread("Images/seun.jpg")
3 average = cv.blur(img, (3,3))
4 cv.imshow("ORIGINAL IMAGE", img)
5 cv.imshow("AVERAGE BLURRED IMAGE", average)
6 cv.waitKey(0)
ORIGINAL IMAGE
AVERAGE BLURRED IM...
PROBLEMS 27 OUTPUT TERMINAL DEBUG CONSOLE JUPYTER COMMENTS
python3.10
PS C:\Users\SHOPINVERSE\Desktop\CV> python smoothingbyavg.py

SMOOTHING TECHNIQUES IN OpenCV Cont'd

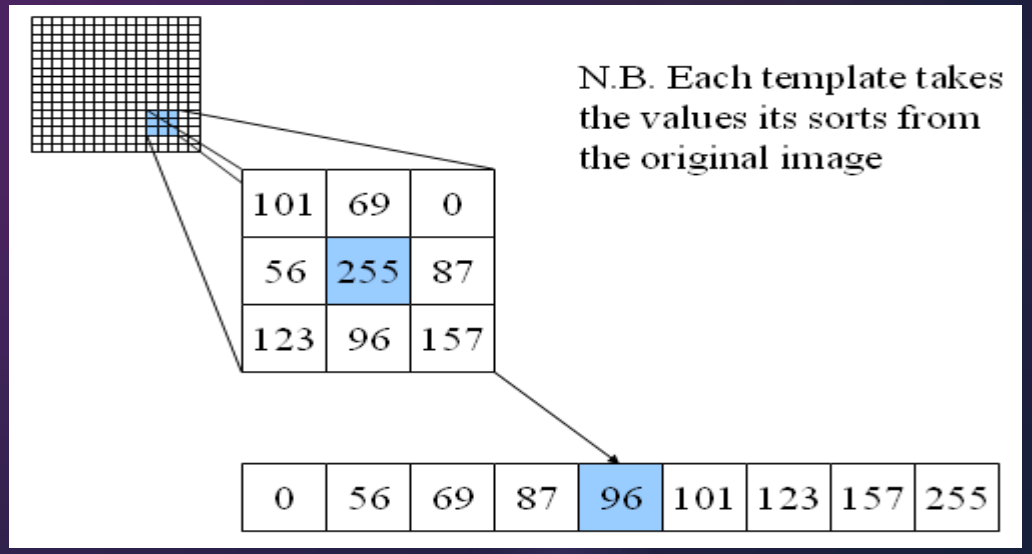
MEDIAN BLUR

In the median blur smoothing technique, the median of all the pixels under the kernel area is selected after sorting in ascending order, and this median value is used to replace the central element. The median blur is a more effective way of smoothing the image because it helps reduce the salt-and-pepper noise.

SYNTAX

cv2.medianBlur(img, kernel as single integer).

Pictorially, Median Blur technique is shown.



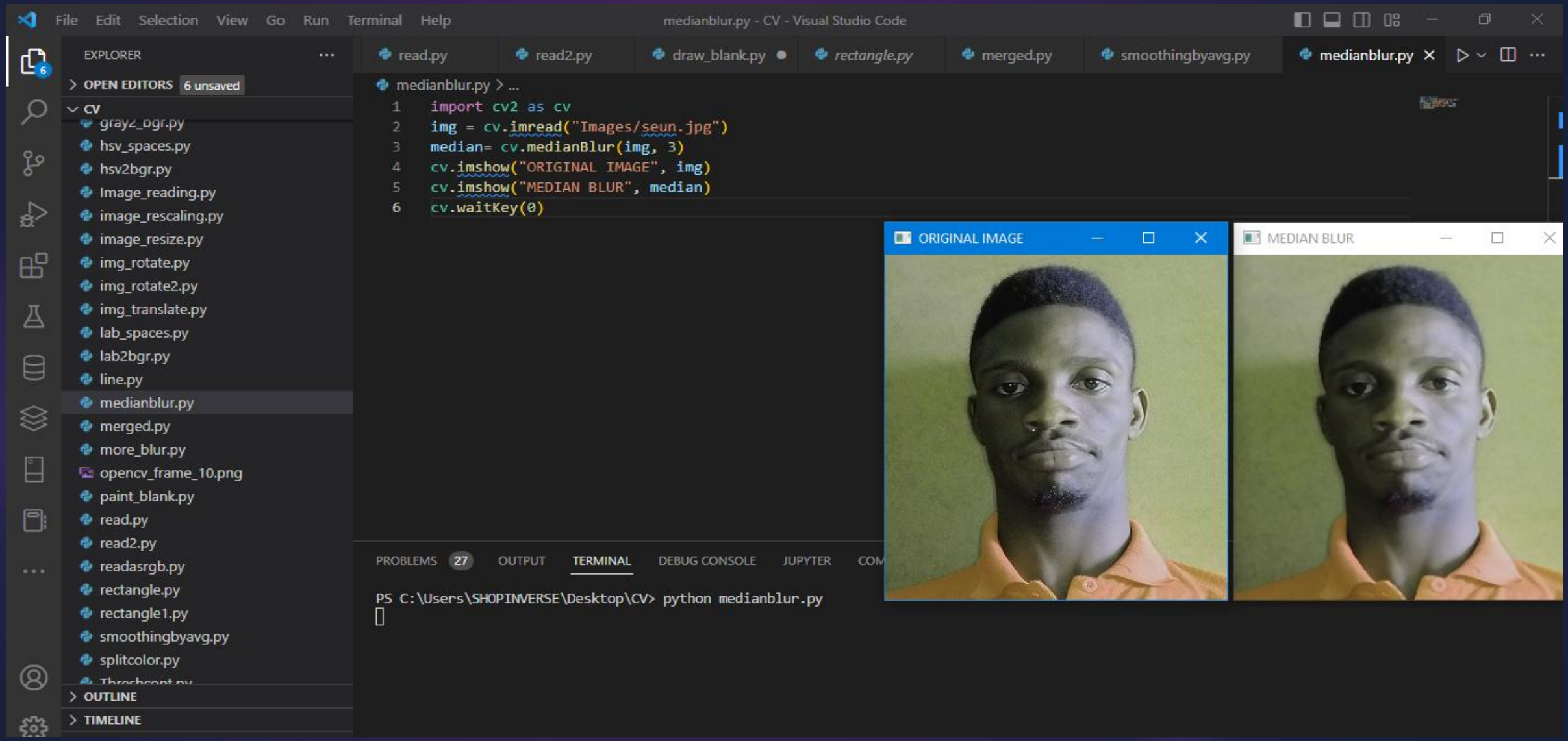
SMOOTHING TECHNIQUES IN OpenCV Cont'd

Example 1: Apply the median blur function to "seun.jpg" to smooth the image using a 3 x 3 kernel size.

SOLUTION

```
import cv2 as cv
img = cv.imread("Images/seun.jpg")
median= cv.medianBlur(img, 3)
cv.imshow("ORIGINAL IMAGE", img)
cv.imshow("MEDIAN BLUR", median)
cv.waitKey(0)
```

OUTPUT


File Edit Selection View Go Run Terminal Help
medianblur.py - CV - Visual Studio Code
EXPLORER
OPEN EDITORS 6 unsaved
CV
gray2_bgr.py
hsv_spaces.py
hsv2bgr.py
Image_reading.py
image_rescaling.py
image_resize.py
img_rotate.py
img_rotate2.py
img_translate.py
lab_spaces.py
lab2bgr.py
line.py
medianblur.py
merged.py
more_blur.py
opencv_frame_10.png
paint_blank.py
read.py
read2.py
readasrgb.py
rectangle.py
rectangle1.py
smoothingbyavg.py
splitcolor.py
OUTLINE
TIMELINE
read.py
read2.py
draw_blank.py
rectangle.py
merged.py
smoothingbyavg.py
medianblur.py
medianblur.py > ...
1 import cv2 as cv
2 img = cv.imread("Images/seun.jpg")
3 median= cv.medianBlur(img, 3)
4 cv.imshow("ORIGINAL IMAGE", img)
5 cv.imshow("MEDIAN BLUR", median)
6 cv.waitKey(0)
PROBLEMS 27 OUTPUT TERMINAL DEBUG CONSOLE JUPYTER
PS C:\Users\SHOPINVERSE\Desktop\CV> python medianblur.py
ORIGINAL IMAGE
MEDIAN BLUR

SMOOTHING TECHNIQUES IN OpenCV Cont'd

BILATERAL FILTERING

This is the type of filtering that is considered most effective because of the way it keeps the sharp edges of an image even after the removal of noise. Although its operation is slower compared to other blurring methods. The bilateral filtering approach works in such a way that each pixel is replaced by the weighted average of its neighbours. Each neighbor is usually weighted by spartial component that penalizes distance pixels. Simply put, bilateral filtering aims to preserve the sharp edges without blurring, but just removing the most texture, fine details, and noise from an image. It is closely related to Gaussian blur, but it is an advanced form of Gaussian blur. It is important to know that after filtering an image bilaterally, the resulting image is always as clear as the original image. This is because edges are well preserved. But if we try to fine tune the parameters, for example, by increasing the sigmaSpace or the sigmaColor, there are chances that the background information of the image becomes blurry.

SYNTAX

```
cv2.bilateralFilter(src, d, sigmaColor, sigmaSpace)
```

SMOOTHING TECHNIQUES IN OpenCV Cont'd

Where,

src: Input image

d: Diameter of each pixel neighbourhood

sigmaColor: A variable of type integer that represents filter sigma in color space

sigmaSpace: A variable of type Integer that represents filter sigma in the coordinate space.

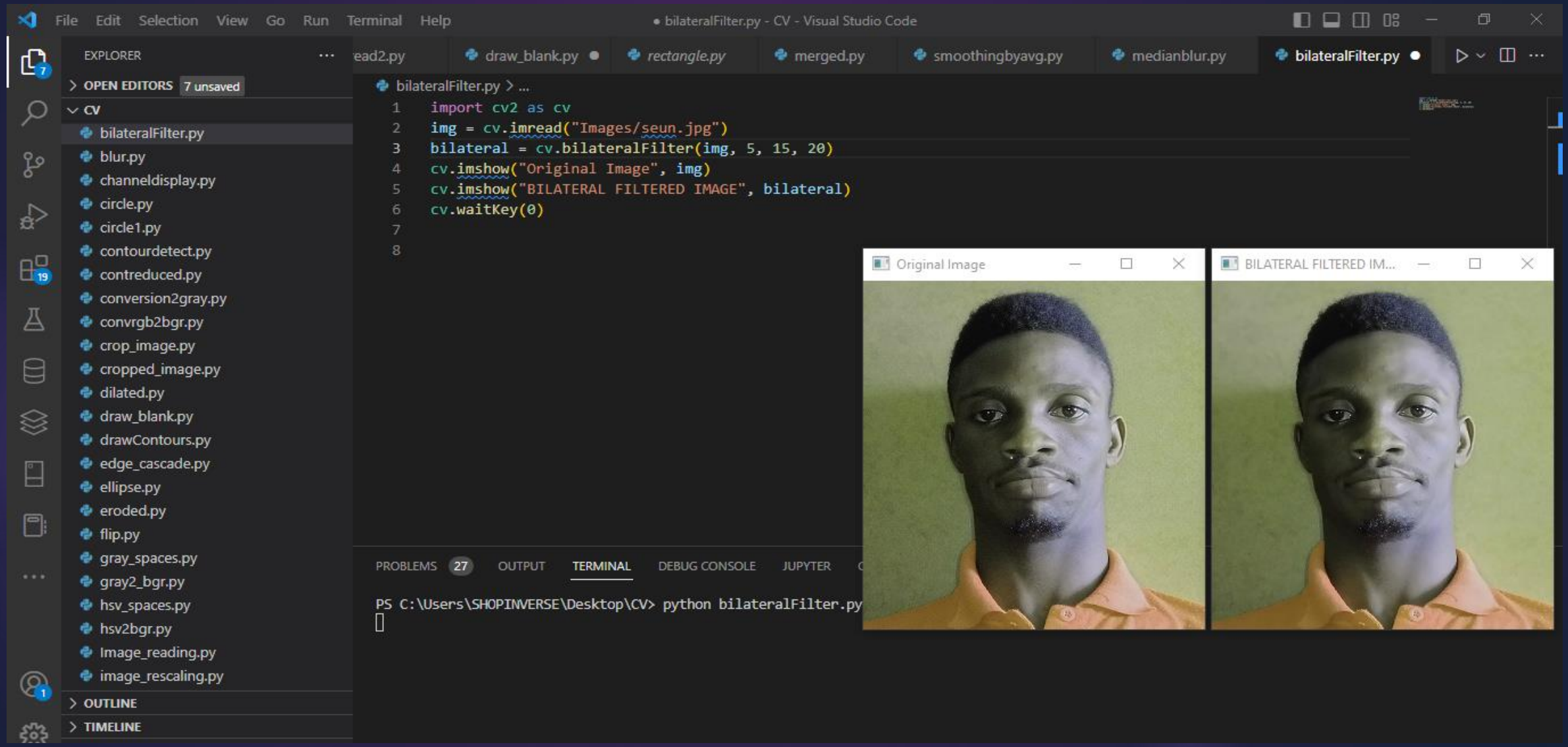
SMOOTHING TECHNIQUES IN OpenCV Cont'd

Example: Let us consider the image "seun.jpg" and apply the bilateral filter method to it to remove noise from the image. By applying the following arguments: diameter = 5 , sigmaColor = 15, sigmaSpace = 20

SOLUTION

```
import cv2 as cv
img = cv.imread("Images/seun.jpg")
bilateral = cv.bilateralFilter(img, 5, 15, 20)
cv.imshow("Original Image", img)
cv.imshow("BILATERAL FILTERED IMAGE", bilateral)
cv.waitKey(0)
```

OUTPUT


File Edit Selection View Go Run Terminal Help
• bilateralFilter.py - CV - Visual Studio Code
EXPLORER
OPEN EDITORS 7 unsaved
CV
bilateralFilter.py
blur.py
channeldisplay.py
circle.py
circle1.py
contourdetect.py
contreduced.py
conversion2gray.py
convrgb2bgr.py
crop_image.py
cropped_image.py
dilated.py
draw_blank.py
drawContours.py
edge_cascade.py
ellipse.py
eroded.py
flip.py
gray_spaces.py
gray2_bgr.py
hsv_spaces.py
hsv2bgr.py
Image_reading.py
image_rescaling.py
OUTLINE
TIMELINE
draw_blank.py
rectangle.py
merged.py
smoothingbyavg.py
medianblur.py
bilateralFilter.py
bilateralFilter.py > ...
1 import cv2 as cv
2 img = cv.imread("Images/seun.jpg")
3 bilateral = cv.bilateralFilter(img, 5, 15, 20)
4 cv.imshow("Original Image", img)
5 cv.imshow("BILATERAL FILTERED IMAGE", bilateral)
6 cv.waitKey(0)
7
8
PROBLEMS 27 OUTPUT TERMINAL DEBUG CONSOLE JUPYTER
PS C:\Users\SHOPINVERSE\Desktop\CV> python bilateralFilter.py
Original Image
BILATERAL FILTERED IM...

NB: The Gaussian Blur Smoothing technique was discussed in one of the previous tutorials. That is why it is not explained further in this presentation.

THANKS FOR VIEWING

More tutorials will be covered in part seven