ORMEDIAN RESEARCH INSTITUTE

TOPIC: : OpenCV Tutorials (part three)

By:

ADEKOLA OLUWASEUN O.

February 17, 2023

TABLE OF CONTENT

IMAGE TRANSFORMATION IN OPENCV

Image Transformation

In OpenCV, we regard image transformation as the process of transforming image data for further reprocessing. It works on the principle of the Euclidean transformation in which the image is subjected to some changes in angles and dimensions without a change in its structure-like area. Specifically, the transformation of this form involves the mapping of some points in one coordinate to some other points in another coordinate.

Some common basic image transformation techniques are:Image Translation

- Image Rotation
- Image Cropping
- Image Resizing
- Image Flipping

IMAGE TRANSFORMATION IN OPENCV Cont'd

Image Translation

Image Translation simply refers to the shifting of an image along the x- and y-axis direction i.e. the shifting of an image in any direction be it left, right, up, or down or possibly by combining these options. While translating an image along the x and y-axis direction, it is important to understand that the values of x or y matters. The effects of the possible values of x or y and their implications on the image translation are listed below:

Negative value of x implies shifting of an image to the left

Negative value of y implies shifting of an image up

Positive value of x implies shifting of an image to the right

Positive value of y implies shifting of the image down

IMAGE TRANSFORMATION IN OPENCV Cont'd

Algorithm for image translation

```
import cv2 as cv
import numpy as np
img = cv.imread("image path")
def translate(img, x, y):
    TransMat = np.float32 ([[1, 0, x], [0, 1, y]])
    dimensions = [img.shape[0], img.shape[1]]
    return cv.warpAffine(img, TransMat, dimensions)
Translated_img = translate (img, x-axis value, y-axis value )
cv.imshow("Original Image", img)
cv.imshow("Translated Image", Translated_img)
cv.waitKey(0)
```

IMAGE TRANSFORMATION IN OPENCV Cont'd

Image Transformation Code Interpretation

Line 1: Import OpenCV module i.e. cv

Line 2: Import the NumPy module

Line 3: Specify the image path and read the image

Line 4: Create a function for the translation which will take the input image, x-axis, and y-axis values as parameters.

Line 5: Specify the translational matrix

Line 6: To get the image dimensions

Line 7: Apply cv.warpAffine() to perform the remapping routine for the translation process

Line 8: Finally call the function [previously defined and pass in the necessary arguments

Lines 9 & 10: Display the old image (Optional) and display the translated image respectively to see the results.

NB: Although the **cv.warpAffine()** method takes in a few parameters such as the src**:** input image, **dst:** output image that has the same size and type as the input image, M: Transformation matrix, **dsize:** the size of the output image, **flags: the** combination of interpolation method. Others are borderMode and borderValue. But in this example, we have only used three parameters and made the rest optional.

IMAGE TRANSFORMATION IN OPENCV Cont'd

Example 1: Translate the image named "seun.jpg" by shifting the image to right by 70 pixels and down by 50 pixels.

```
import cv2 as cv
import numpy as np
img = cv.imread("Images/Seun.jpg")
def translate(img, x, y):
    TransMat = np.float32 ([[1, 0, x], [0, 1, y]])
    dimensions = [img.shape[0], img.shape[1]]
    return cv.warpAffine(img, TransMat, dimensions)
Translated_img = translate (img, 170, 120)
cv.imshow("Original Image", img)
cv.imshow("Translated Translated", Translated_img)
cv.waitKey(0)
```

OUTPUT

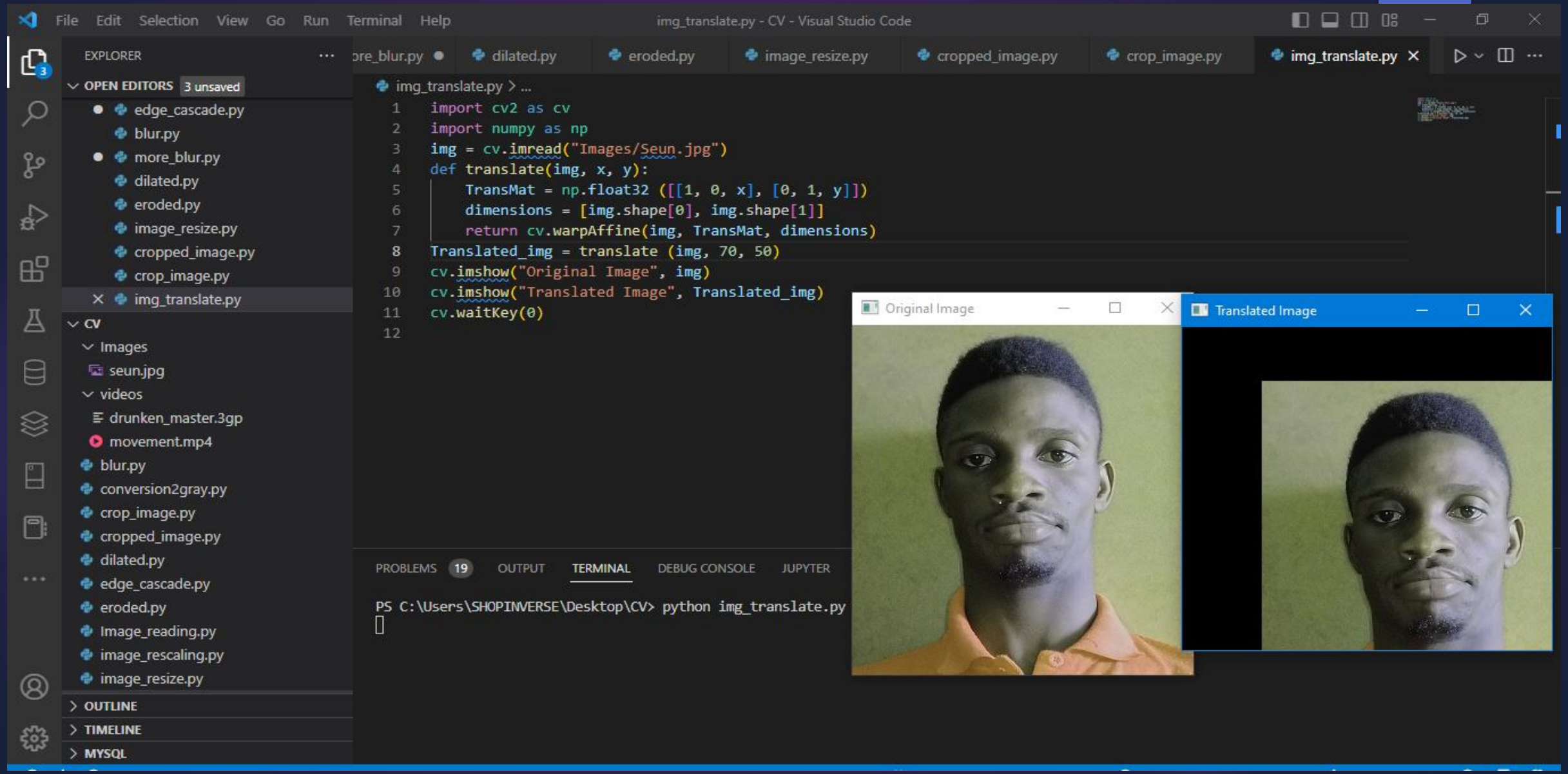

File Edit Selection View Go Run Terminal Help
img_translate.py - CV - Visual Studio Code
EXPLORER
OPEN EDITORS 3 unsaved
edge_cascade.py
blur.py
more_blur.py
dilated.py
eroded.py
image_resize.py
cropped_image.py
crop_image.py
img_translate.py
CV
Images
seun.jpg
videos
drunken_master.3gp
movement.mp4
blur.py
conversion2gray.py
crop_image.py
cropped_image.py
dilated.py
edge_cascade.py
eroded.py
Image_reading.py
image_rescaling.py
image_resize.py
OUTLINE
TIMELINE
MYSQL
img_translate.py > ...
import cv2 as cv
import numpy as np
img = cv.imread("Images/Seun.jpg")
def translate(img, x, y):
TransMat = np.float32 ([[1, 0, x], [0, 1, y]])
dimensions = [img.shape[0], img.shape[1]]
return cv.warpAffine(img, TransMat, dimensions)
Translated_img = translate (img, 70, 50)
cv.imshow("Original Image", img)
cv.imshow("Translated Image", Translated_img)
cv.waitKey(0)
PROBLEMS 19 OUTPUT TERMINAL DEBUG CONSOLE JUPYTER
PS C:\Users\SHOPINVERSE\Desktop\CV> python img_translate.py
Original Image
Translated Image

IMAGE TRANSFORMATION IN OPENCV Cont'd

Example 2: Translate the image named "seun.jpg" by shifting the image to left by 70 pixels and up by 50 pixels.

OUTPUT

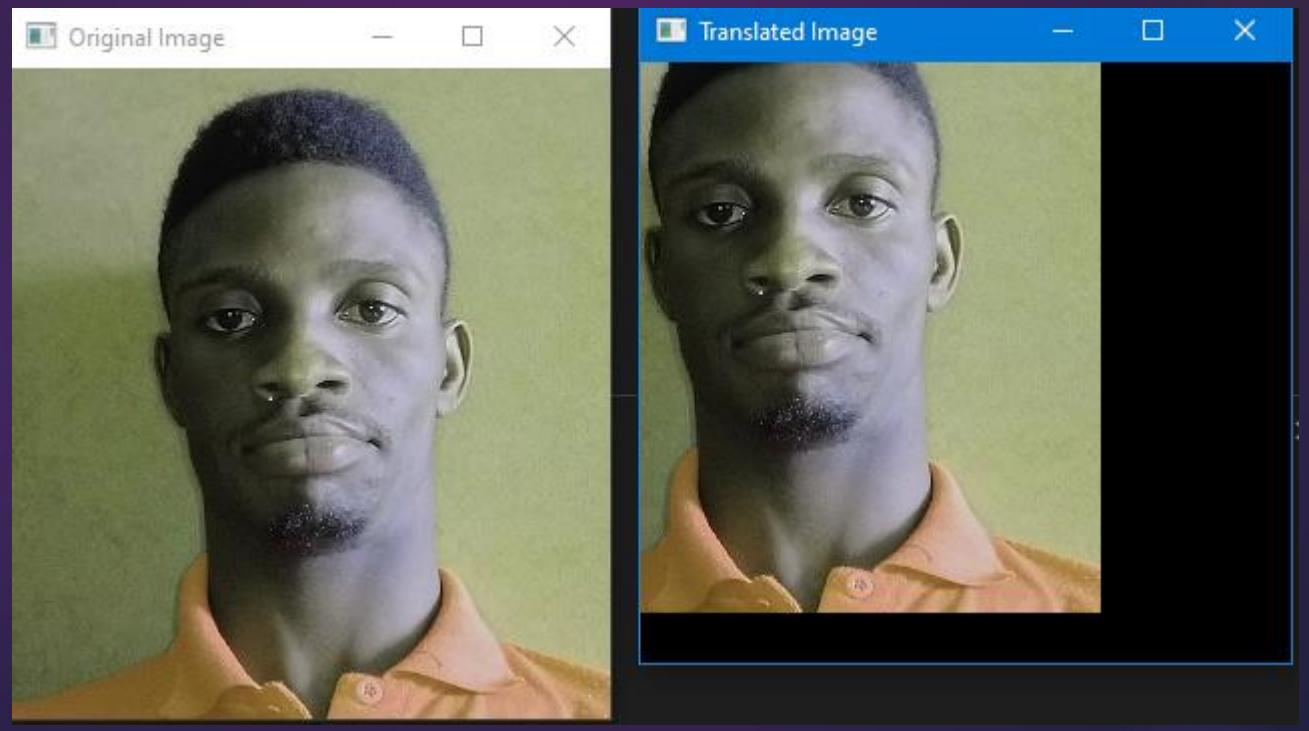


IMAGE TRANSFORMATION IN OPENCV Cont'd

Image Rotation in OpenCV

Image Rotation in openCV refers to the image processing technique in which the image is correctly displayed from different angles. Image rotation involves the computation of the inverse transformation for every destination pixel. Simply put, it is the process of rotating an image. OpenCV has a method to simply rotate an image. But if there is a need for some arbitrary angles for rotation then there is a need to use a more complex one. This means that the cv2.rotate() method is used for the rotation of a 2D array of an image in multiple of 90° whereas the cv2.getRotationMatrix2D() is used in the case of arbitrary angles when transformation matrix will be required for the rotation of an image.

For the cv2.rotate() method, the simplest syntax will be:

```
rotated_image = cv.rotate(src, rotationCode)
```

Where src implies the input image while rotationCode could be cv2.ROTATE_90_CLOCKWISE, cv2.ROTATE_90_COUNTERCLOCKWISE or cv2.ROTATE_180. These are the three basic rotation codes available in OpenCV.

IMAGE TRANSFORMATION IN OPENCV Cont'd

Using the cv2.rotate() method

SYNTAX

```
import cv2 as cv
src = cv.imread(“path to the image file")
rotated_image = cv.rotate(src, rotationCode)
cv.imshow("Original Image", src)
cv.imshow("Rotated Image", rotated_image)
cv.waitKey(0)
```

IMAGE TRANSFORMATION IN OPENCV Cont'd

Using the cv2.rotate() method

Example 1: Rotate the original image "seun.jpg" by (i) 90° clockwise (ii) 90° counterclockwise and, (iii) 180° clockwise.

SOLUTION

(i)

```
import cv2 as cv
img = cv.imread("Image/seun.jpg")
rotated_image = cv.rotate(img,
cv.ROTATE_90_CLOCKWISE)
cv.imshow("Original Image", img)
cv.imshow("Rotated Image",
rotated_image)
cv.waitKey(0)
```

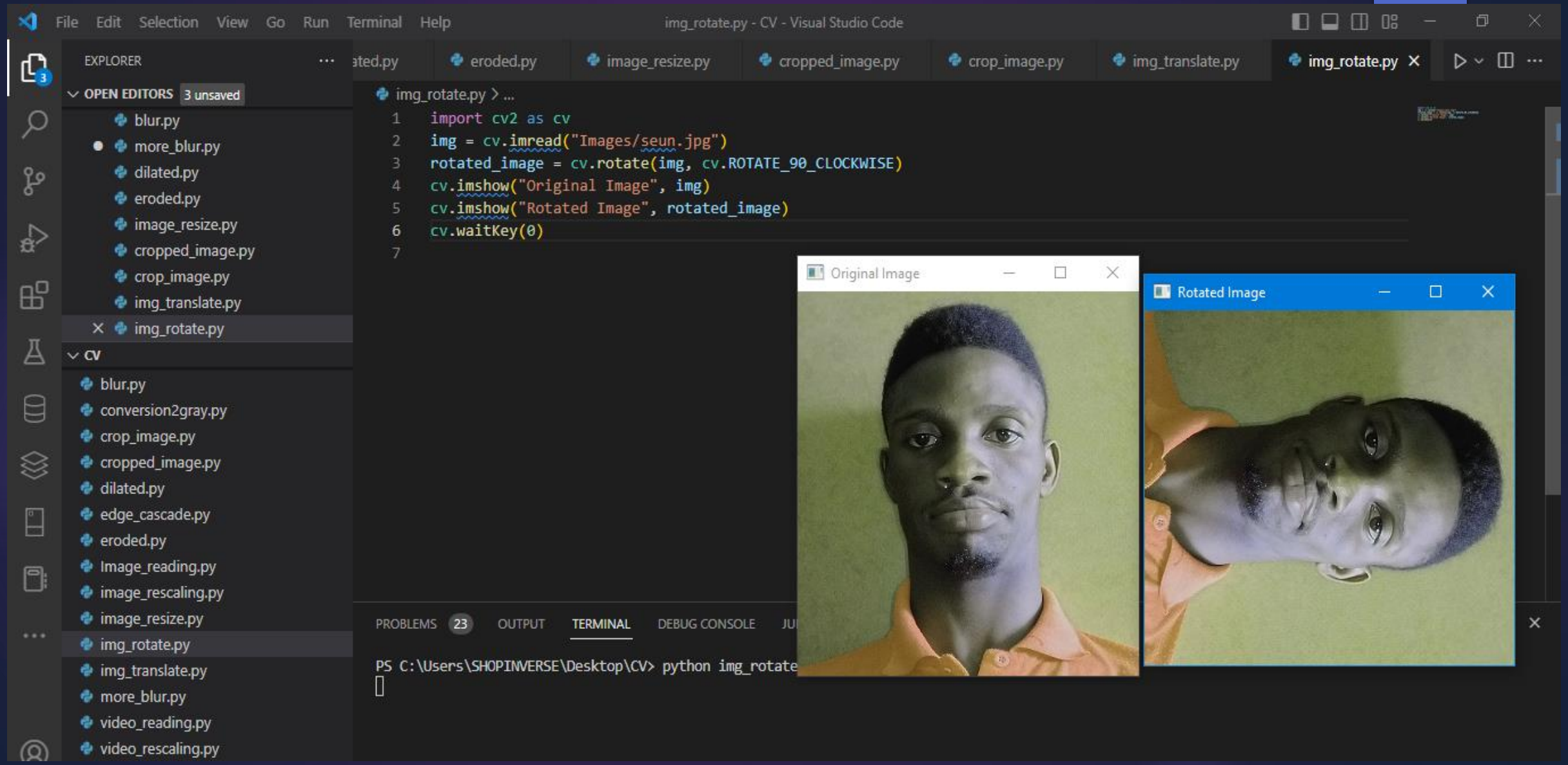
(ii)

```
import cv2 as cv
img = cv.imread("Image/seun.jpg")
rotated_image = cv.rotate(img,
cv.ROTATE_90_COUNTERCLOCK
WISE)
cv.imshow("Original Image", img)
cv.imshow("Rotated Image",
rotated_image)
cv.waitKey(0)
```

(iii)

```
import cv2 as cv
img = cv.imread("Image/seun.jpg")
rotated_image = cv.rotate(img,
cv.ROTATE_180)
cv.imshow("Original Image", img)
cv.imshow("Rotated Image",
rotated_image)
cv.waitKey(0)
```

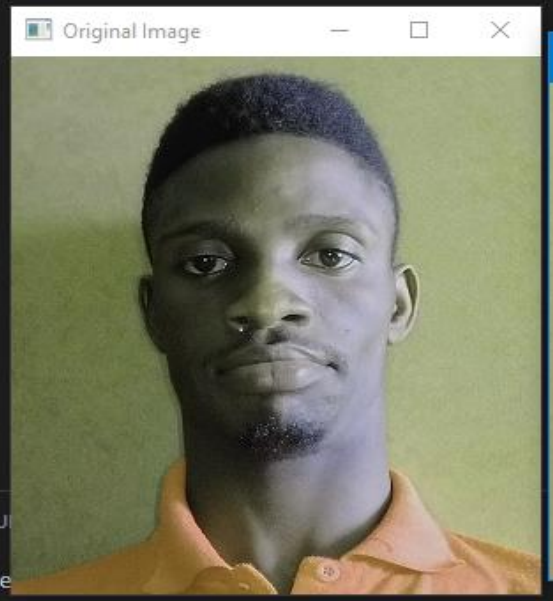
OUTPUT

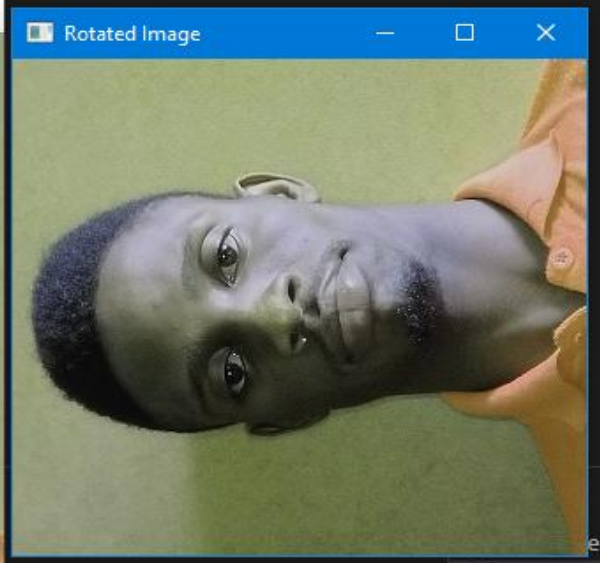

File Edit Selection View Go Run Terminal Help
img_rotate.py - CV - Visual Studio Code
EXPLORER
OPEN EDITORS 3 unsaved
blur.py
more_blur.py
dilated.py
eroded.py
image_resize.py
cropped_image.py
crop_image.py
img_translate.py
img_rotate.py
CV
blur.py
conversion2gray.py
crop_image.py
cropped_image.py
dilated.py
edge_cascade.py
eroded.py
Image_reading.py
image_rescaling.py
image_resize.py
img_rotate.py
img_translate.py
more_blur.py
video_reading.py
video_rescaling.py
eroded.py
image_resize.py
cropped_image.py
crop_image.py
img_translate.py
img_rotate.py
img_rotate.py > ...
import cv2 as cv
img = cv.imread("Images/seun.jpg")
rotated_image = cv.rotate(img, cv.ROTATE_90_CLOCKWISE)
cv.imshow("Original Image", img)
cv.imshow("Rotated Image", rotated_image)
cv.waitKey(0)
PROBLEMS 23 OUTPUT TERMINAL DEBUG CONSOLE
PS C:\Users\SHOPINVERSE\Desktop\CV> python img_rotate
Original Image
Rotated Image

OUTPUT

```python
import cv2 as cv
img = cv.imread("Images/seun.jpg")
rotated_image = cv.rotate(img, cv.ROTATE_90_COUNTERCLOCKWISE)
cv.imshow("Original Image", img)
cv.imshow("Rotated Image", rotated_image)
cv.waitKey(0)
```

```
PS C:\Users\SHOPINVERSE\Desktop\CV> python img_rotate
```





OUTPUT

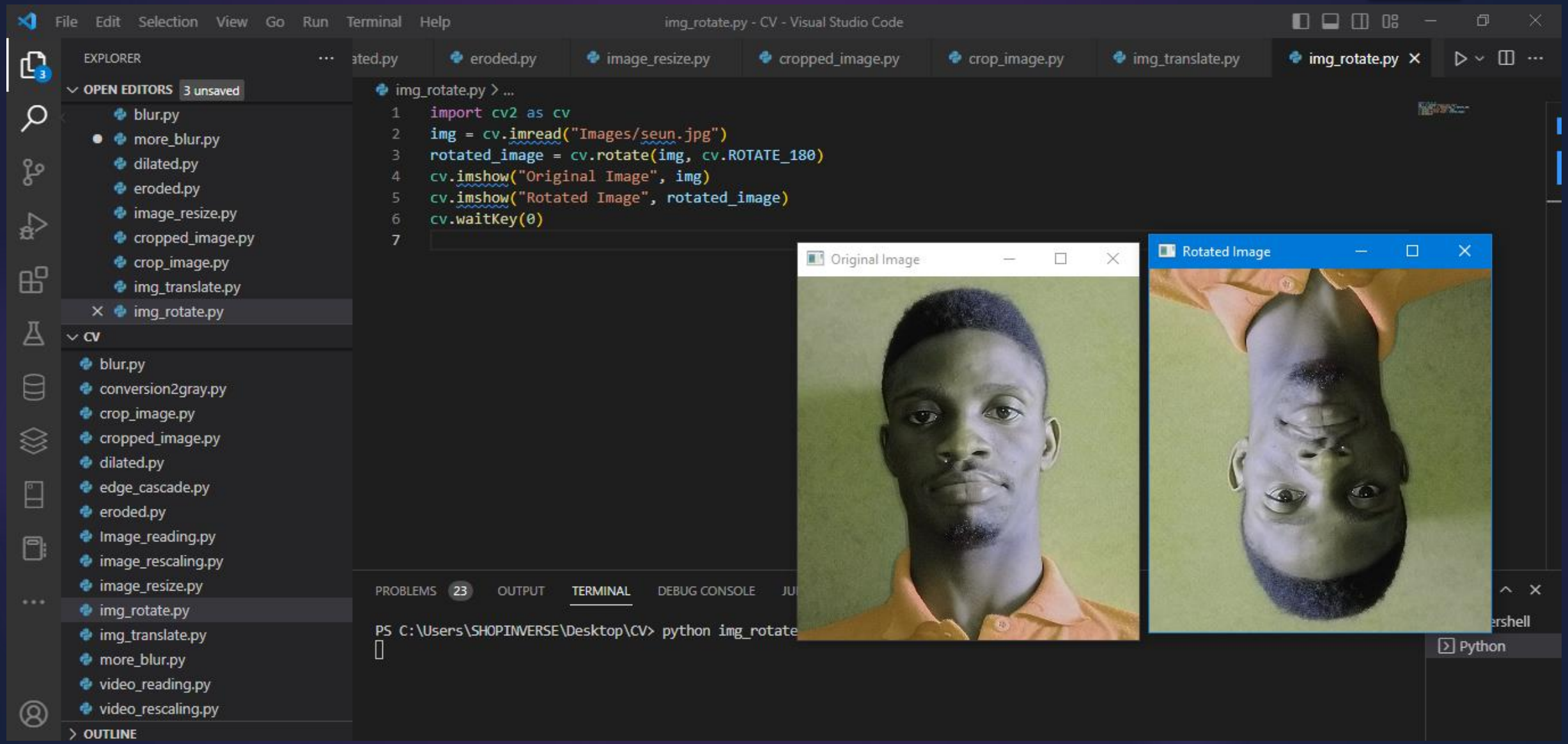

File Edit Selection View Go Run Terminal Help
img_rotate.py - CV - Visual Studio Code
EXPLORER
OPEN EDITORS 3 unsaved
blur.py
more_blur.py
dilated.py
eroded.py
image_resize.py
cropped_image.py
crop_image.py
img_translate.py
img_rotate.py
CV
blur.py
conversion2gray.py
crop_image.py
cropped_image.py
dilated.py
edge_cascade.py
eroded.py
Image_reading.py
image_rescaling.py
image_resize.py
img_rotate.py
img_translate.py
more_blur.py
video_reading.py
video_rescaling.py
OUTLINE
eroded.py
image_resize.py
cropped_image.py
crop_image.py
img_translate.py
img_rotate.py
img_rotate.py > ...
import cv2 as cv
img = cv.imread("Images/seun.jpg")
rotated_image = cv.rotate(img, cv.ROTATE_180)
cv.imshow("Original Image", img)
cv.imshow("Rotated Image", rotated_image)
cv.waitKey(0)
PROBLEMS 23 OUTPUT TERMINAL DEBUG CONSOLE
PS C:\Users\SHOPINVERSE\Desktop\CV> python img_rotate
Original Image
Rotated Image
Python

IMAGE TRANSFORMATION IN OPENCV Cont'd

Image Rotation using the cv2.getRotationMatrix2D() Method

```
import cv2 as cv
img = cv.imread("Images/seun.jpg")
def rotate(img, angle, RotPt=None):
    (width, height) = img.shape[:2]
    if RotPt is None:
        rotPt = (width//2, height//2)
        rotMat = cv.getRotationMatrix2D(rotPt, angle, 1.0)
        dimensions = (width, height)
        return cv.warpAffine(img, rotMat, dimensions)
rotated_image = rotate(img, angle_value)
cv.imshow("Original Image", img)
cv.imshow("Rotated Image", rotated_image)
cv.waitKey(0)
```

IMAGE TRANSFORMATION IN OPENCV Cont'd

Image Transformation Code Interpretation

Line 1: Import OpenCV module i.e. cv

Line 2: Specify the image path and read the image

Line 3: Create a function for the rotation which will take the input image, angle, and rotation point as parameters.

Line 4: Grab the image dimension and calculate the image center point.

Line 5: To get the image dimensions

Line 6: Grab the rotation matrix by specifying the angle and scaling.

Line 7: Set the dimension variable

Line 8: Perform the remapping routing using the cv.warpAffine() method.

Lines 9 & 10: Display the old image (Optional) and display the translated image respectively to see the results.

IMAGE TRANSFORMATION IN OPENCV Cont'd

Example: Rotate the original image "seun.jpg" by 150°, then by 320°.

SOLUTION

Rotation by 150°

```
import cv2 as cv
img = cv.imread("Images/seun.jpg")
def rotate(img, angle, RotPt=None):
    (width, height) = img.shape[:2]
    if RotPt is None:
        rotPt = (width//2, height//2)
        rotMat = cv.getRotationMatrix2D(rotPt, angle, 1.0)
        dimensions = (width, height)
        return cv.warpAffine(img, rotMat, dimensions)
rotated_image = rotate(img, 150)
cv.imshow("Original Image", img)
cv.imshow("Rotated Image", rotated_image)
cv.waitKey(0)
```

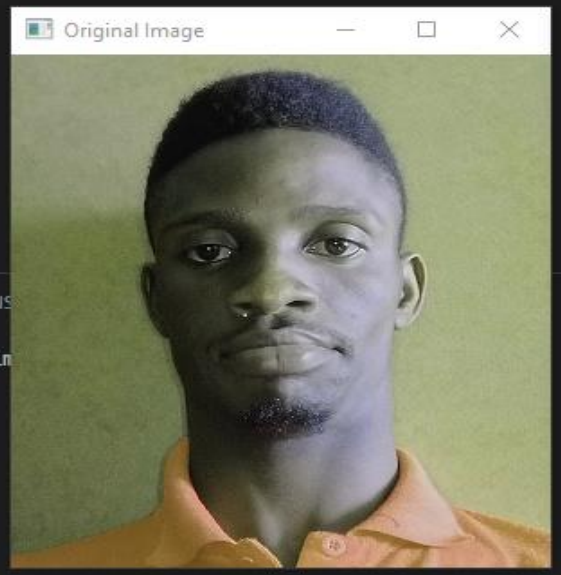
Rotation by 320°

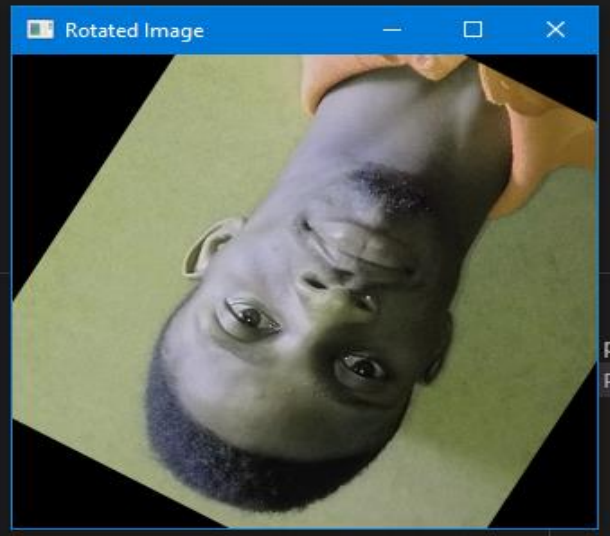
```
import cv2 as cv
img = cv.imread("Images/seun.jpg")
def rotate(img, angle, RotPt=None):
    (width, height) = img.shape[:2]
    if RotPt is None:
        rotPt = (width//2, height//2)
        rotMat = cv.getRotationMatrix2D(rotPt, angle, 1.0)
        dimensions = (width, height)
        return cv.warpAffine(img, rotMat, dimensions)
rotated_image = rotate(img, 320)
cv.imshow("Original Image", img)
cv.imshow("Rotated Image", rotated_image)
cv.waitKey(0)
```

OUTPUT

```python
import cv2 as cv
img = cv.imread("Images/seun.jpg")
def rotate(img, angle, RotPt=None):
    (width, height) = img.shape[:2]
    if RotPt is None:
        rotPt = (width//2, height//2)
        rotMat = cv.getRotationMatrix2D(rotPt, angle, 1.0)
        dimensions = (width, height)
        return cv.warpAffine(img, rotMat, dimensions)
rotated_image = rotate(img, 150)
cv.imshow("Original Image", img)
cv.imshow("Rotated Image", rotated_image)
cv.waitKey(0)
```

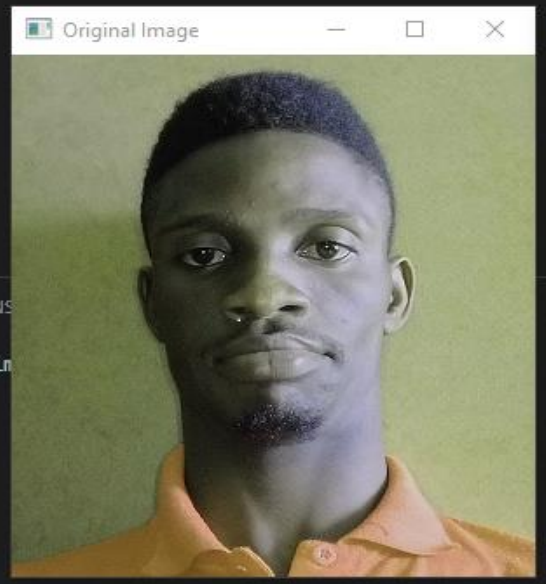
```
PS C:\Users\SHOPINVERSE\Desktop\CV> python im
```





OUTPUT

```python
import cv2 as cv
img = cv.imread("Images/seun.jpg")
def rotate(img, angle, RotPt=None):
    (width, height) = img.shape[:2]
    if RotPt is None:
        rotPt = (width//2, height//2)
        rotMat = cv.getRotationMatrix2D(rotPt, angle, 1.0)
        dimensions = (width, height)
        return cv.warpAffine(img, rotMat, dimensions)
rotated_image = rotate(img, 320)
cv.imshow("Original Image", img)
cv.imshow("Rotated Image", rotated_image)
cv.waitKey(0)
```



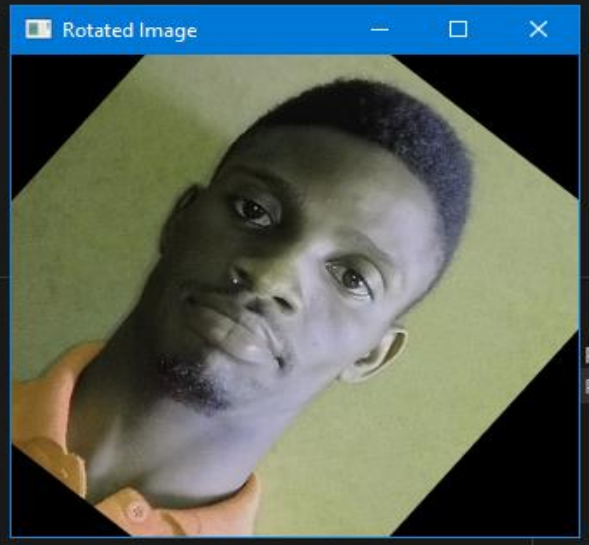


IMAGE TRANSFORMATION IN OPENCV Cont'd

Image Flipping in OpenCV

This could be regarded as the process of turning an image horizontally or vertically. In the process of flipping the right side of an image is made to become the left, and the top becomes the bottom. In a nutshell, flipping is simply the mirror image of the original image.

In the process of flipping an image, there is no need to define a function. cv2.flip() method is applied to such an image. The cv2.flip() method takes in two arguments i.e. the input image and the flipCode. The flipCode takes in value 0 or 1 or -1. 0 specifies vertical flipping, 1 specifies horizontal flipping while -1 specifies both horizontal and vertical flipping.

SYNTAX

```
Flipped_image = cv.flip(img, flipCode)
```

IMAGE TRANSFORMATION IN OPENCV Cont'd

Example: Flip the original image "seun.jpg" vertically, horizontally, and both horizontally and vertically.

Vertical Flipping

```
import cv2 as cv
img = cv.imread("Images/seun.jpg")
flipped_image = cv.flip(img, 0)
cv.imshow("Original Image", img)
cv.imshow("Flipped Image", flipped_image)
cv.waitKey(0)
```

Horizontal Flipping

```
import cv2 as cv
img = cv.imread("Images/seun.jpg")
flipped_image = cv.flip(img, 1)
cv.imshow("Original Image", img)
cv.imshow("Flipped Image", flipped_image)
cv.waitKey(0)
```

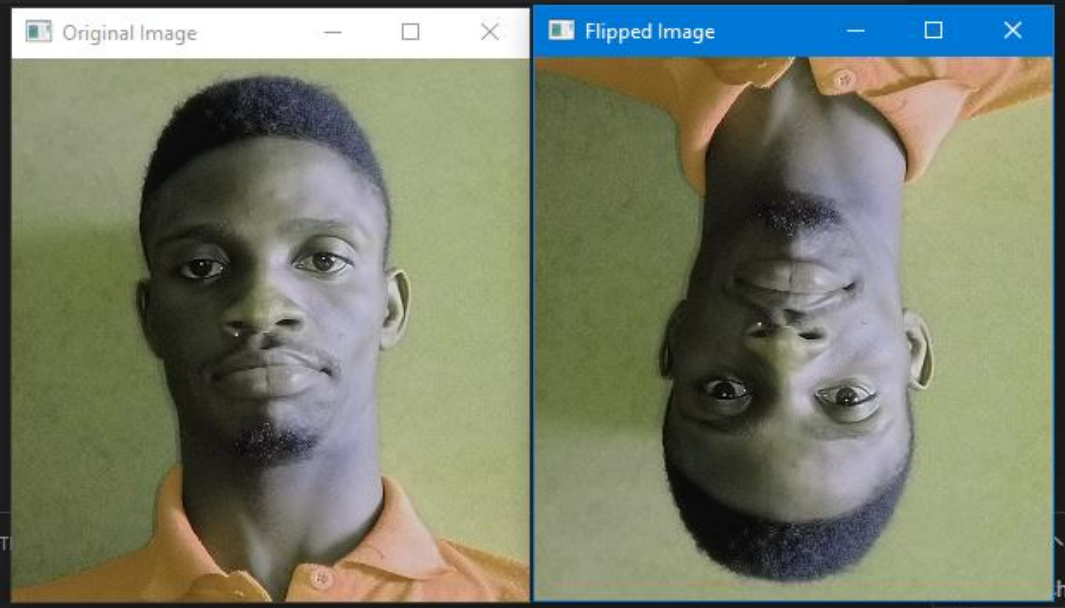
Horizontal and Vertical Clipping

```
import cv2 as cv
img = cv.imread("Images/seun.jpg")
flipped_image = cv.flip(img, -1)
cv.imshow("Original Image", img)
cv.imshow("Flipped Image", flipped_image)
cv.waitKey(0)
```

OUTPUT

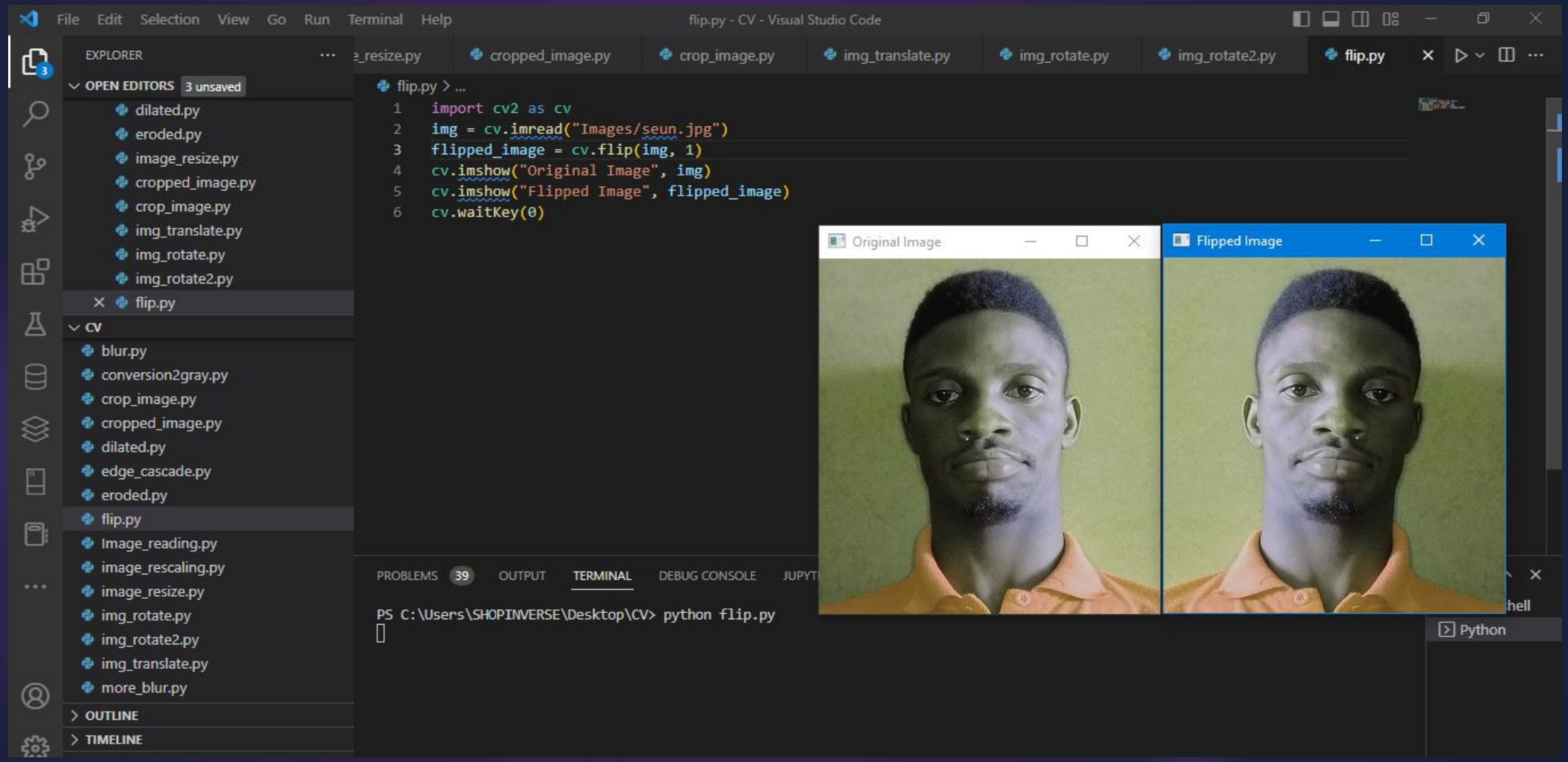
flip.py - CV - Visual Studio Code

```python
import cv2 as cv
img = cv.imread("Images/seun.jpg")
flipped_image = cv.flip(img, 0)
cv.imshow("Original Image", img)
cv.imshow("Flipped Image", flipped_image)
cv.waitKey(0)
```




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

OUTPUT


File Edit Selection View Go Run Terminal Help
flip.py - CV - Visual Studio Code
EXPLORER
OPEN EDITORS 3 unsaved
dilated.py
eroded.py
image_resize.py
cropped_image.py
crop_image.py
img_translate.py
img_rotate.py
img_rotate2.py
flip.py
CV
blur.py
conversion2gray.py
crop_image.py
cropped_image.py
dilated.py
edge_cascade.py
eroded.py
flip.py
Image_reading.py
image_rescaling.py
image_resize.py
img_rotate.py
img_rotate2.py
img_translate.py
more_blur.py
OUTLINE
TIMELINE
cropped_image.py
crop_image.py
img_translate.py
img_rotate.py
img_rotate2.py
flip.py
import cv2 as cv
img = cv.imread("Images/seun.jpg")
flipped_image = cv.flip(img, 1)
cv.imshow("Original Image", img)
cv.imshow("Flipped Image", flipped_image)
cv.waitKey(0)
Original Image
Flipped Image
PROBLEMS 39
OUTPUT
TERMINAL
DEBUG CONSOLE
PS C:\Users\SHOPINVERSE\Desktop\CV> python flip.py
Python

OUTPUT

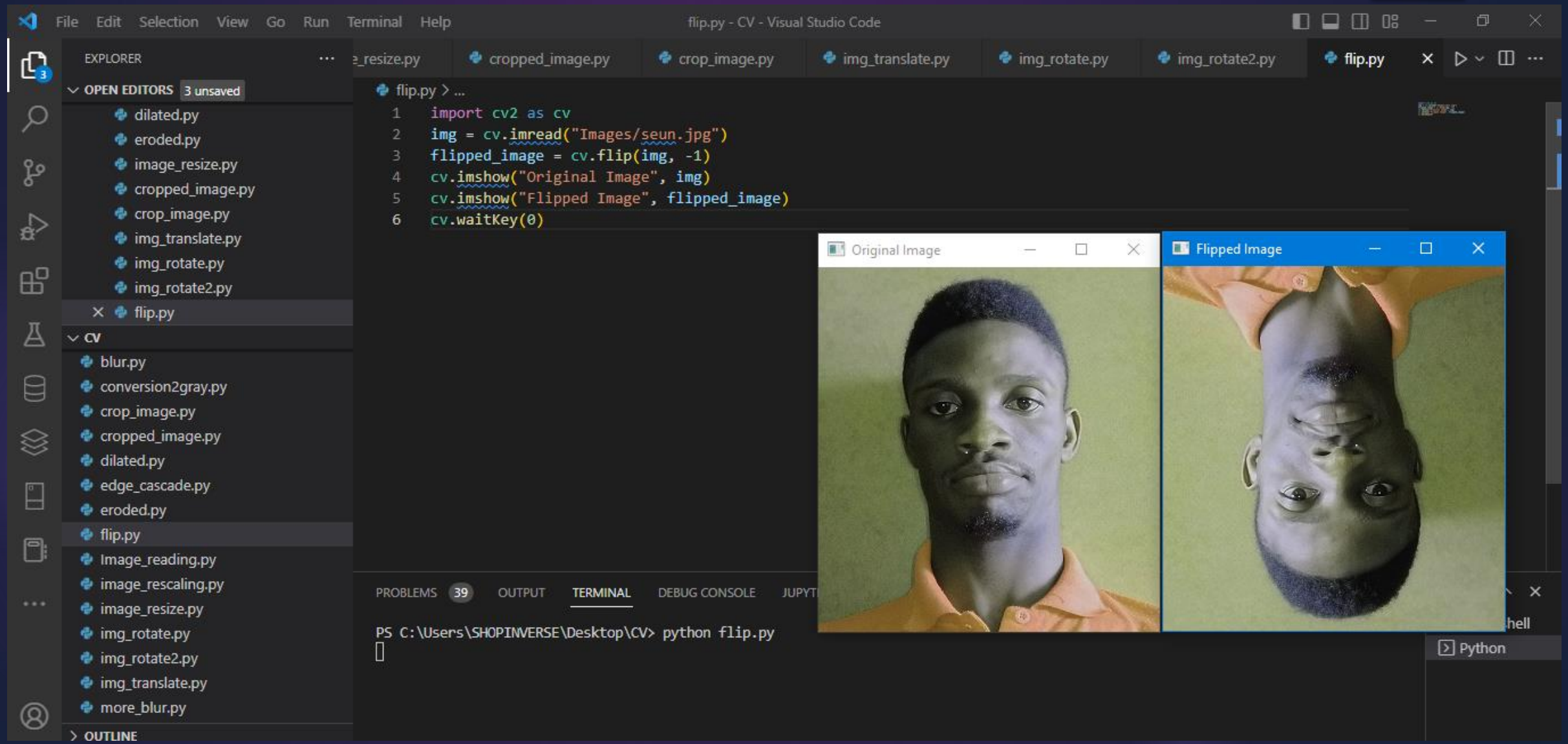

File Edit Selection View Go Run Terminal Help
flip.py - CV - Visual Studio Code
EXPLORER
OPEN EDITORS 3 unsaved
dilated.py
eroded.py
image_resize.py
cropped_image.py
crop_image.py
img_translate.py
img_rotate.py
img_rotate2.py
flip.py
CV
blur.py
conversion2gray.py
crop_image.py
cropped_image.py
dilated.py
edge_cascade.py
eroded.py
flip.py
Image_reading.py
image_rescaling.py
image_resize.py
img_rotate.py
img_rotate2.py
img_translate.py
more_blur.py
OUTLINE
cropped_image.py
crop_image.py
img_translate.py
img_rotate.py
img_rotate2.py
flip.py
import cv2 as cv
img = cv.imread("Images/seun.jpg")
flipped_image = cv.flip(img, -1)
cv.imshow("Original Image", img)
cv.imshow("Flipped Image", flipped_image)
cv.waitKey(0)
Original Image
Flipped Image
PROBLEMS 39
OUTPUT
TERMINAL
DEBUG CONSOLE
PS C:\Users\SHOPINVERSE\Desktop\CV> python flip.py
Python

IMAGE TRANSFORMATION IN OPENCV Cont'd

NOTE: Image cropping and Image resizing were explained in the previous lecture (part two)

READING OF IMAGE/VIDEO FROM WEBCAM

Syntax for reading images/videos from webcam

```
import cv2 as cv
captured_image = cv.VideoCapture(0)
while True:
    isTrue, frame = captured_image.read()
    cv.imshow("frame", frame)
    if cv.waitKey(1) == ord('r'):
        break
captured_image.release()
cv.destroyAllWindows()
```

CODE INTERPRETATION

Line 1: Import the OpenCV module

Line 2: Create an instance of the VideoCapture to read the camera event

Line 3: Check if the frame can be read

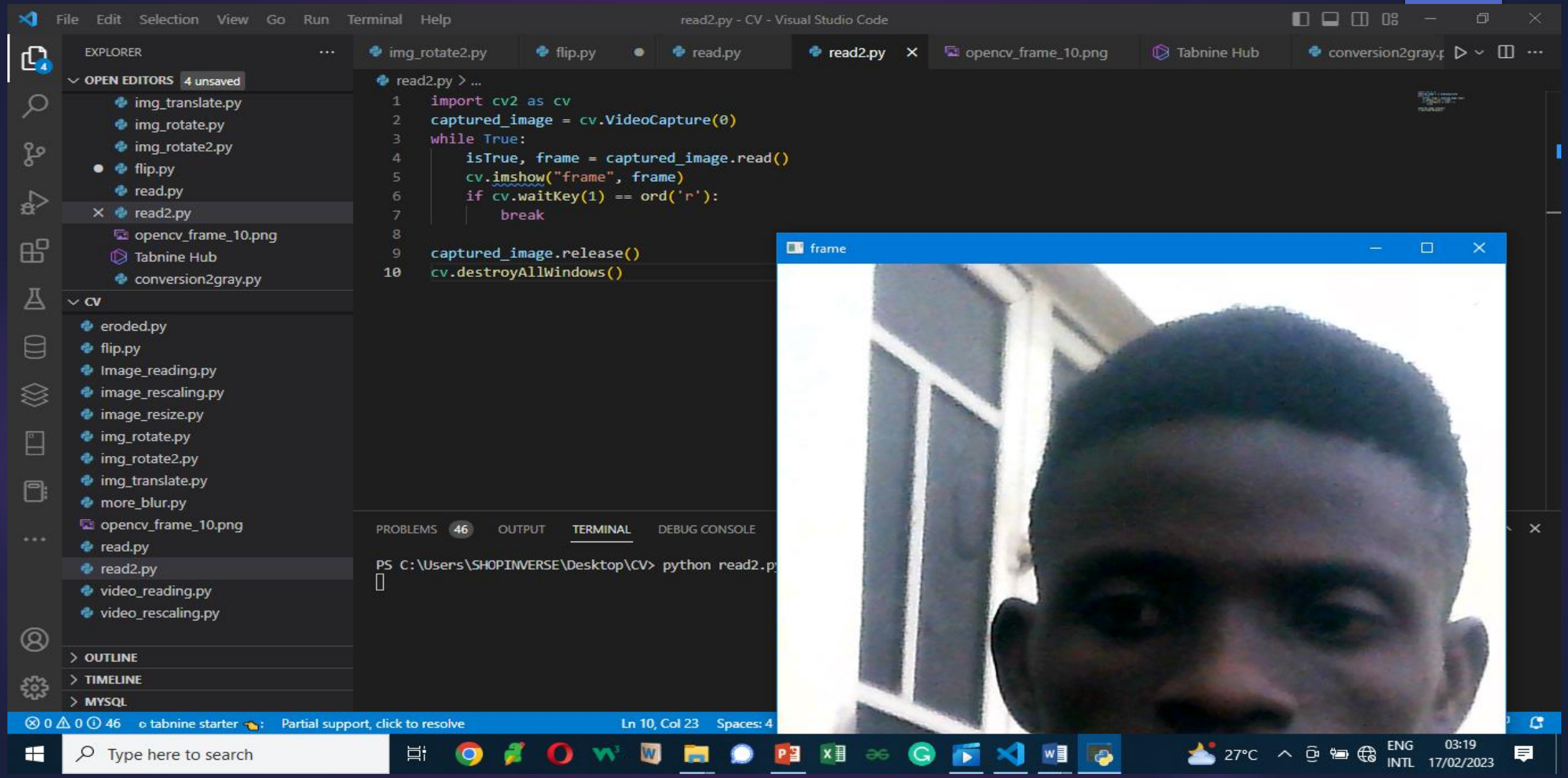
Line 4: Displays the image

Line 5: To specify the wait key and key to close the frame.

Line 6: Used to release the camera so that other resources can use it.

Line 7: Used terminate all windows

OUTPUT


read2.py - CV - Visual Studio Code
EXPLORER
OPEN EDITORS 4 unsaved
img_translate.py
img_rotate.py
img_rotate2.py
flip.py
read.py
read2.py
opencv_frame_10.png
Tabnine Hub
conversion2gray.py
CV
eroded.py
flip.py
Image_reading.py
image_rescaling.py
image_resize.py
img_rotate.py
img_rotate2.py
img_translate.py
more_blur.py
opencv_frame_10.png
read.py
read2.py
video_reading.py
video_rescaling.py
OUTLINE
TIMELINE
MYSQL
read2.py > ...
import cv2 as cv
captured_image = cv.VideoCapture(0)
while True:
isTrue, frame = captured_image.read()
cv.imshow("frame", frame)
if cv.waitKey(1) == ord('r'):
break
captured_image.release()
cv.destroyAllWindows()
PROBLEMS 46
OUTPUT
TERMINAL
DEBUG CONSOLE
PS C:\Users\SHOPINVERSE\Desktop\CV> python read2.py
frame
Ln 10, Col 23
Spaces: 4
Partial support, click to resolve
Type here to search
27°C
ENG INTL
03:19
17/02/2023

THANKS FOR VIEWING

More tutorials will be covered in part four