

# Assignment-14

- **Installing open-cv:-**

The screenshot shows the Visual Studio Code interface. The Explorer sidebar on the left displays a folder named 'OPEN\_CV\_PKG' containing files 'car.jpg', 'cv.jpg', 'cv1.py', and 'main.py'. The main editor area shows two files: 'cv1.py' and 'main.py'. Below the editor is a terminal window with the following content:

```
PS D:\Open_cv_pkg> pip install numpy
Requirement already satisfied: numpy in c:\users\admin\appdata\local\programs\python\python313\lib\site-packages (2.2.6)
PS D:\Open_cv_pkg> pip install opencv-python
Requirement already satisfied: opencv-python in c:\users\admin\appdata\local\programs\python\python313\lib\site-packages (4.1.2.0.88)
Requirement already satisfied: numpy<2.3.0,>=2 in c:\users\admin\appdata\local\programs\python\python313\lib\site-packages
  (from opencv-python) (2.2.6)
PS D:\Open_cv_pkg>
```

The status bar at the bottom indicates Python 3.13 (64-bit) and the current time as 4:47 PM.

## Reading and writing image:-

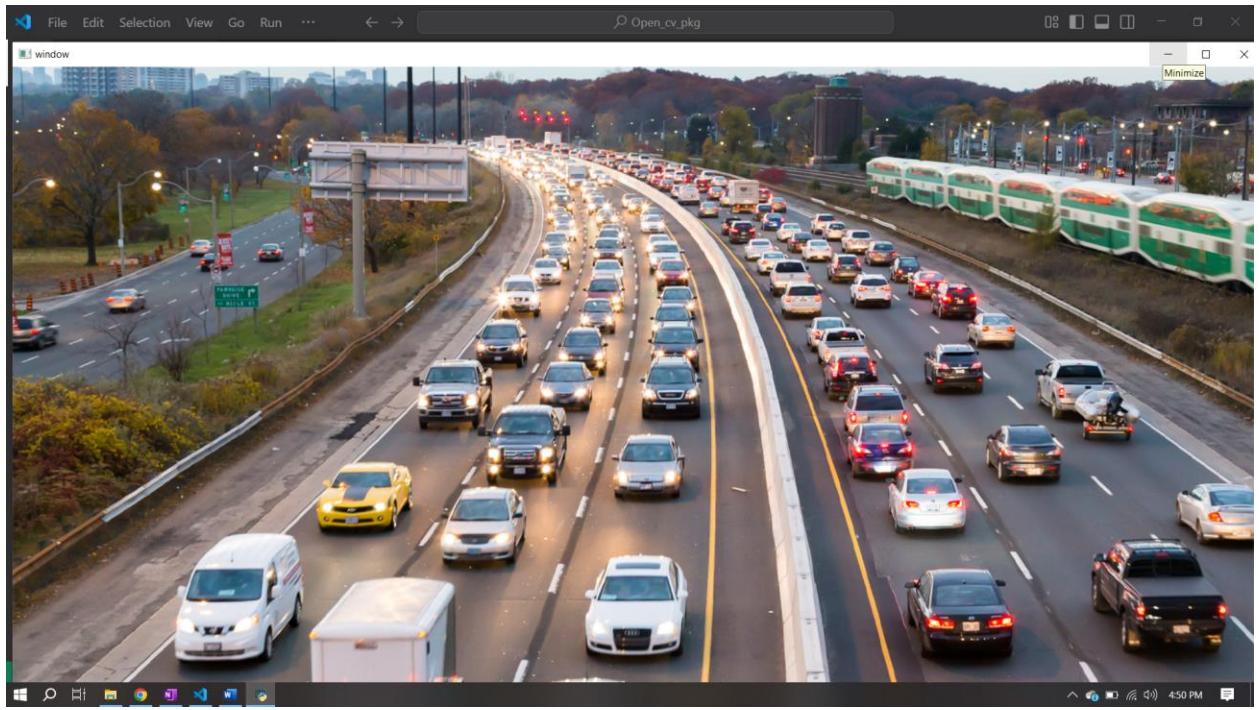
```
import cv2

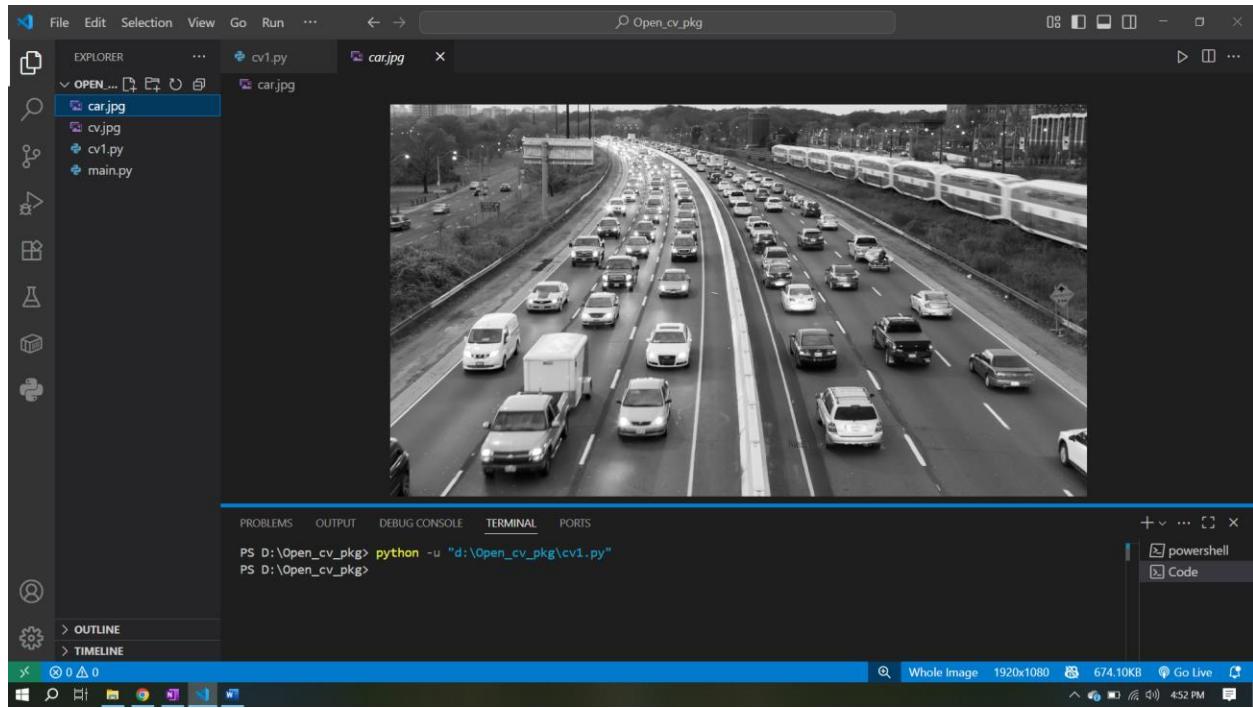
img = cv2.imread("D:/Open_cv_pkg/cv.jpg",0)    #for black n white img add = 0 zero
for original img = 1.-1. its default

cv2.imshow('window', img)
cv2.imwrite("car.jpg", img) # it saved in given path but wrtting img,can mention
filename / path where we saved file

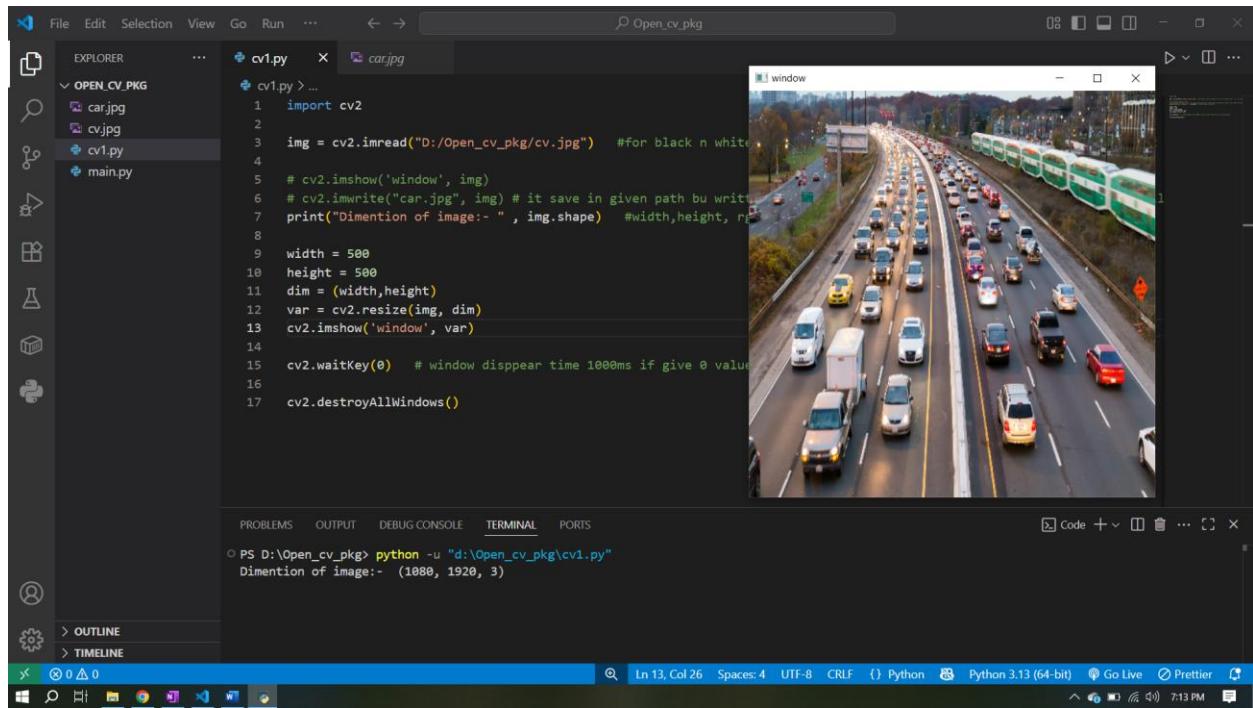
cv2.waitKey(0)    # window disappear time 1000ms if give 0 value window not close
automatically

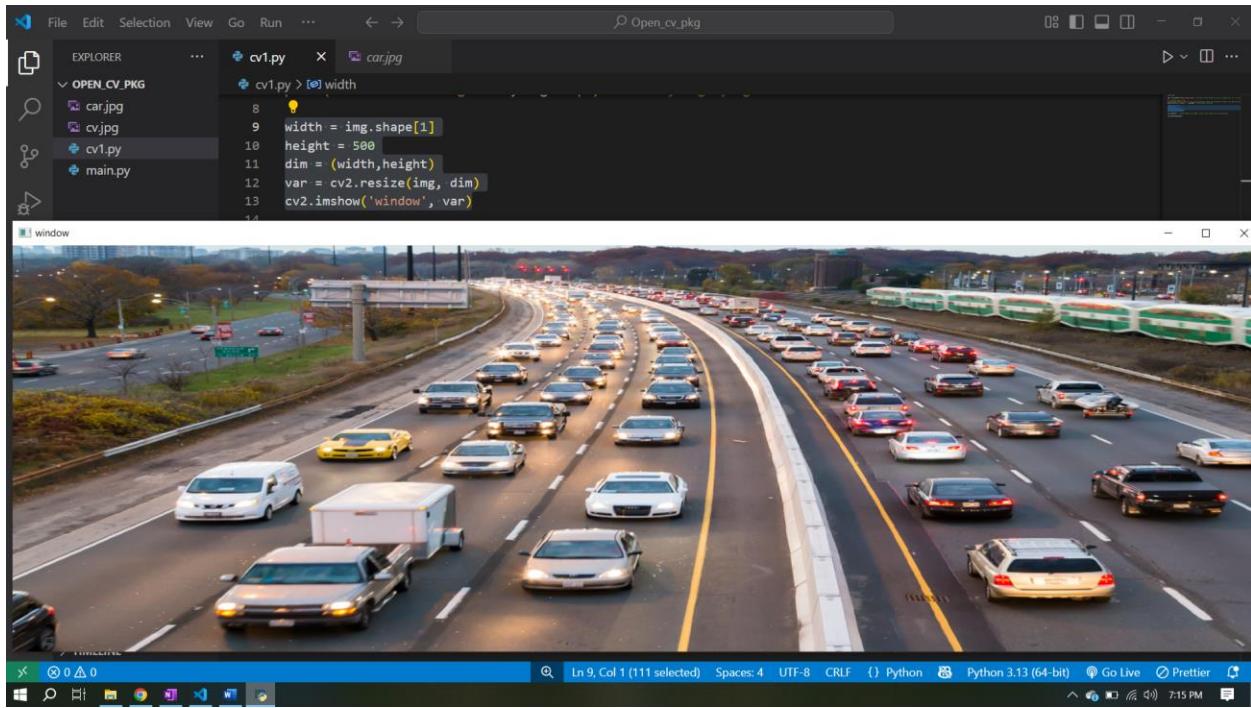
cv2.destroyAllWindows()
```





## Resizing an image:





### Morphological Operations:

```
import cv2
import numpy as np

img = cv2.imread("C:/Users/Admin/OneDrive/Desktop/cv.jpg")
width = 600
height = 850
dim = (width,height)
var = cv2.resize(img, dim)

kernel = np.ones((5,5), dtype='uint8')
erosion = cv2.erode(var, kernel, iterations=1)
dilation = cv2.dilate(var, kernel, iterations=1)
opening = cv2.morphologyEx(var, cv2.MORPH_OPEN, kernel)
closing = cv2.morphologyEx(var, cv2.MORPH_CLOSE, kernel)
gradient = cv2.morphologyEx(var, cv2.MORPH_GRADIENT, kernel)
tophat = cv2.morphologyEx(var, cv2.MORPH_TOPHAT, kernel)
blackhat = cv2.morphologyEx(var, cv2.MORPH_BLACKHAT, kernel)

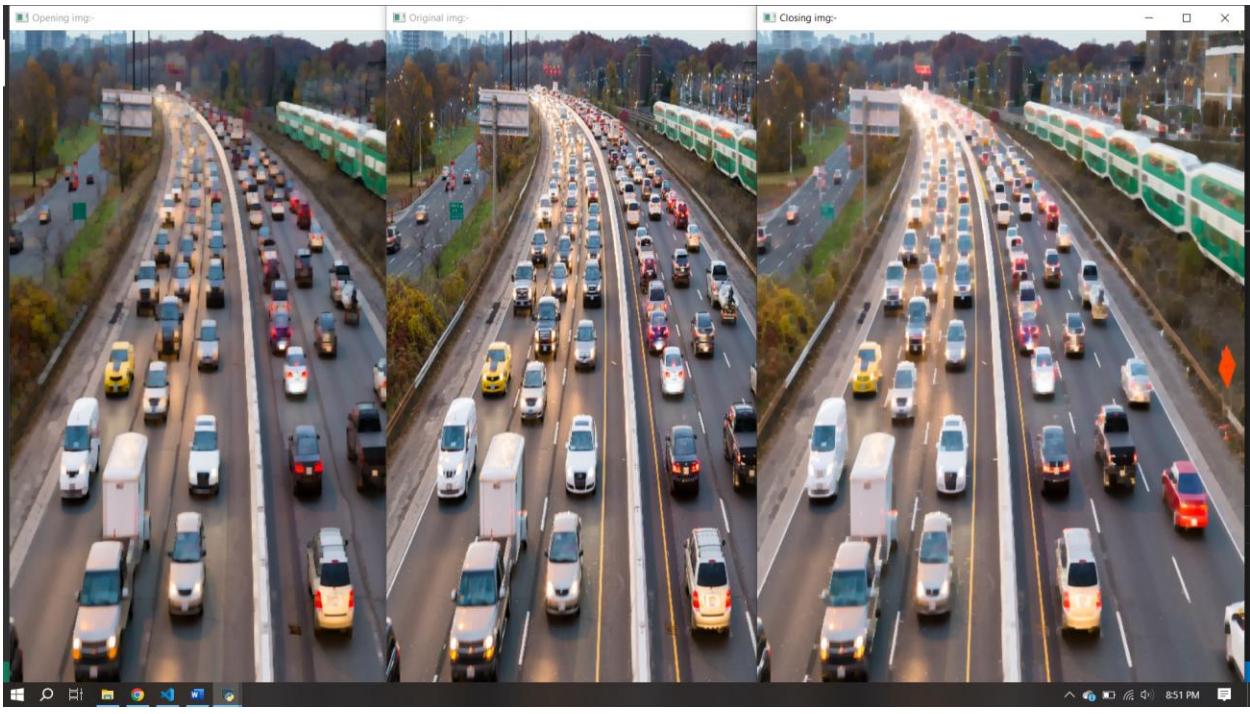
cv2.imshow('Original img:- ', var)
cv2.imshow('Erosion img:- ', erosion)
cv2.imshow('Dilation img:- ', dilation)
cv2.imshow('Opening img:- ', opening)
cv2.imshow('Closing img:- ', closing)
cv2.imshow('Gradient img:- ', gradient)
```

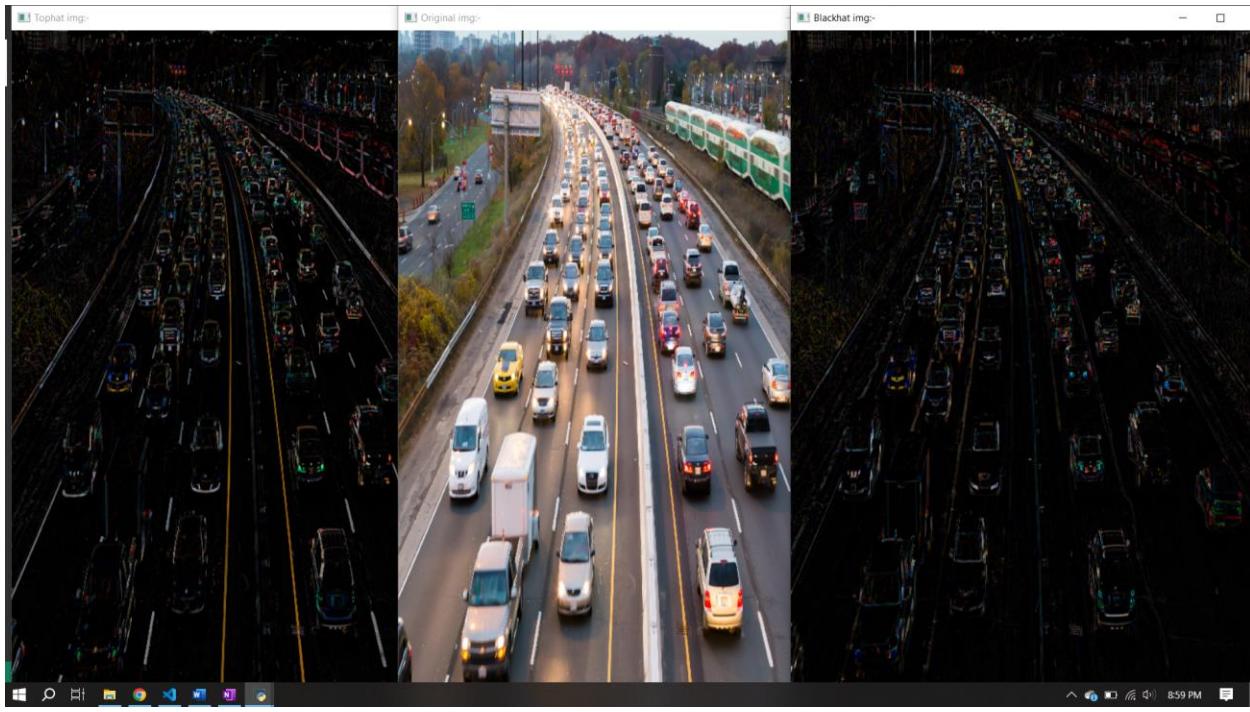
```
cv2.imshow('Tophat img:- ', tophat)
cv2.imshow('Blackhat img:- ', blackhat)

cv2.waitKey(0)

cv2.destroyAllWindows()
```







### Flipping an image:-

```
import cv2
import numpy as np

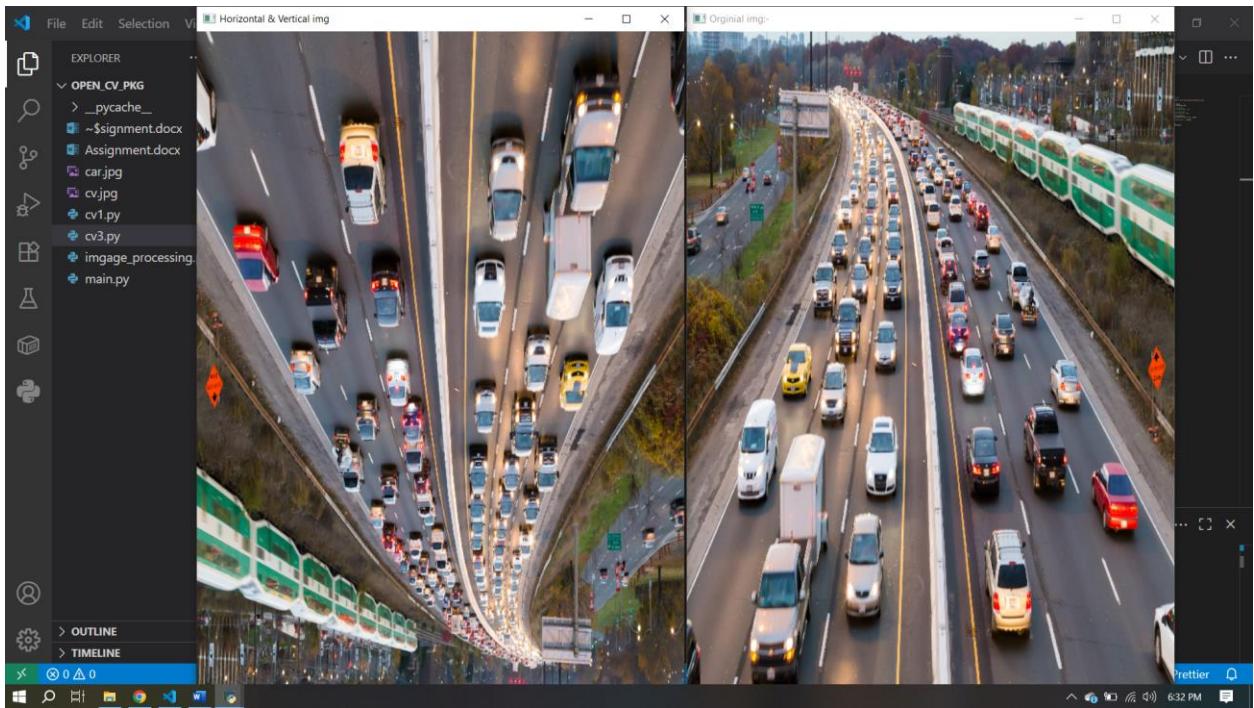
img = cv2.imread("C:/Users/Admin/OneDrive/Desktop/cv.jpg")
width = 600
height = 850
dim = (width, height)
resized = cv2.resize(img, dim)
print("Size of img:- ", img.size)
# flip = cv2.flip(resized,1)
# cv2.imshow("Horizontal img ",flip)

# flip1 = cv2.flip(resized,0)
# cv2.imshow("Vertical img ",flip1)

flip2 = cv2.flip(resized,-1)
cv2.imshow("Horizontal & Vertical img ",flip2)

cv2.imshow("Orginial img:- ", resized)

cv2.waitKey(0)
cv2.destroyAllWindows()
```



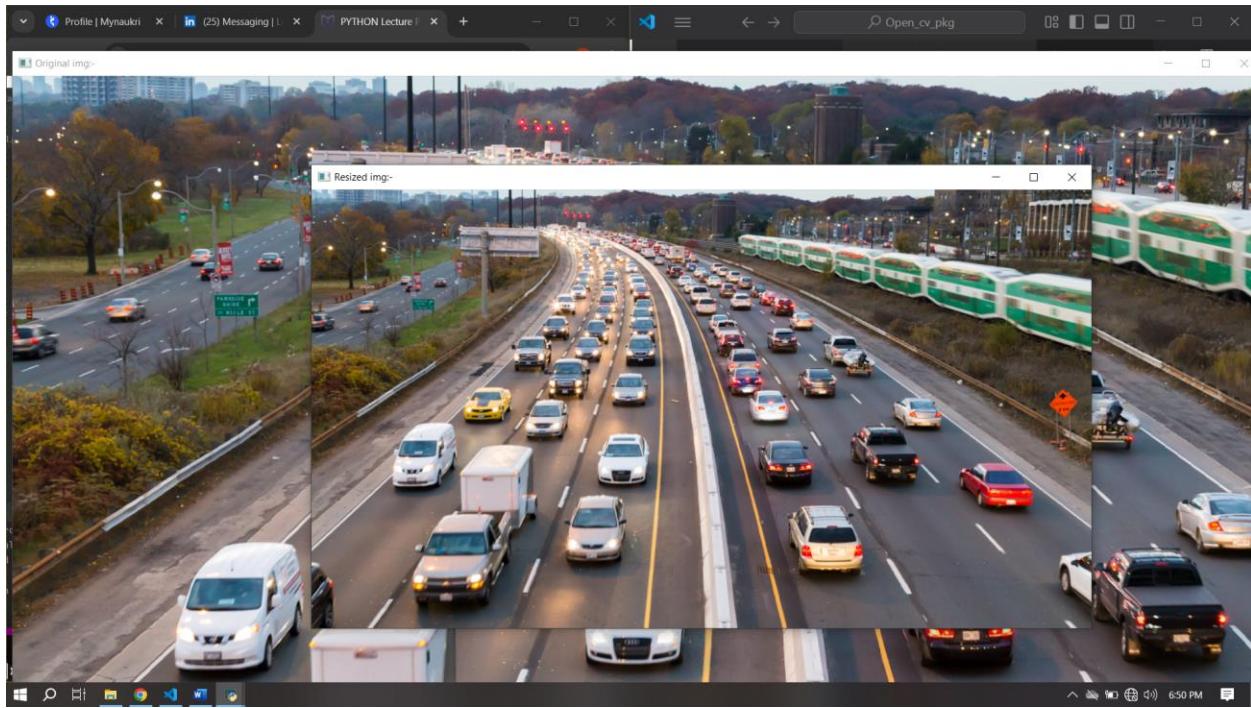
The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** Shows a folder named "OPEN\_CV\_PKG" containing files: \_pycache\_, ~\$ignment.docx, Assignment.docx, car.jpg, cv.jpg, cv1.py, cv3.py, image\_processing.py, and main.py.
- Code Editor:** The active file is "cv3.py". The code reads an image, resizes it, and shows both the original and resized images.
- Terminal:** Shows the command "python -u "d:\Open\_cv\_pkg\cv3.py"" was run, and the output includes the size of the original image (6220800).
- Status Bar:** Displays the current file is "cv3.py", line 19, column 39, and the Python version is Python 3.13 (64-bit).

## Resizing and img based on aspect ratio:-

The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** Shows a folder named "OPEN\_CV\_PKG" containing files: \_pycache\_, ~\$ignment.docx, Assignment.docx, car.jpg, cv.jpg, cv1.py, cv3.py, image\_processing.py, and main.py.
- Code Editor:** The active file is "cv4.py". The code reads an image, prints its original shape, scales it by 50%, and then prints the shape of the resized image.
- Terminal:** Shows the command "python -u "d:\Open\_cv\_pkg\cv4.py"" was run, and the output shows the original image shape (1080, 1920, 3) and the resized image shape (540, 960, 3).
- Status Bar:** Displays the current file is "cv4.py", line 5, column 1, and the Python version is Python 3.13 (64-bit).



File Edit Selection View Go Run ... ← → ⌘ Open\_cv\_pkg

EXPLORER OPEN\_CV\_PKG cv1.py image\_processing.py cv3.py cv4.py

cv4.py > ...

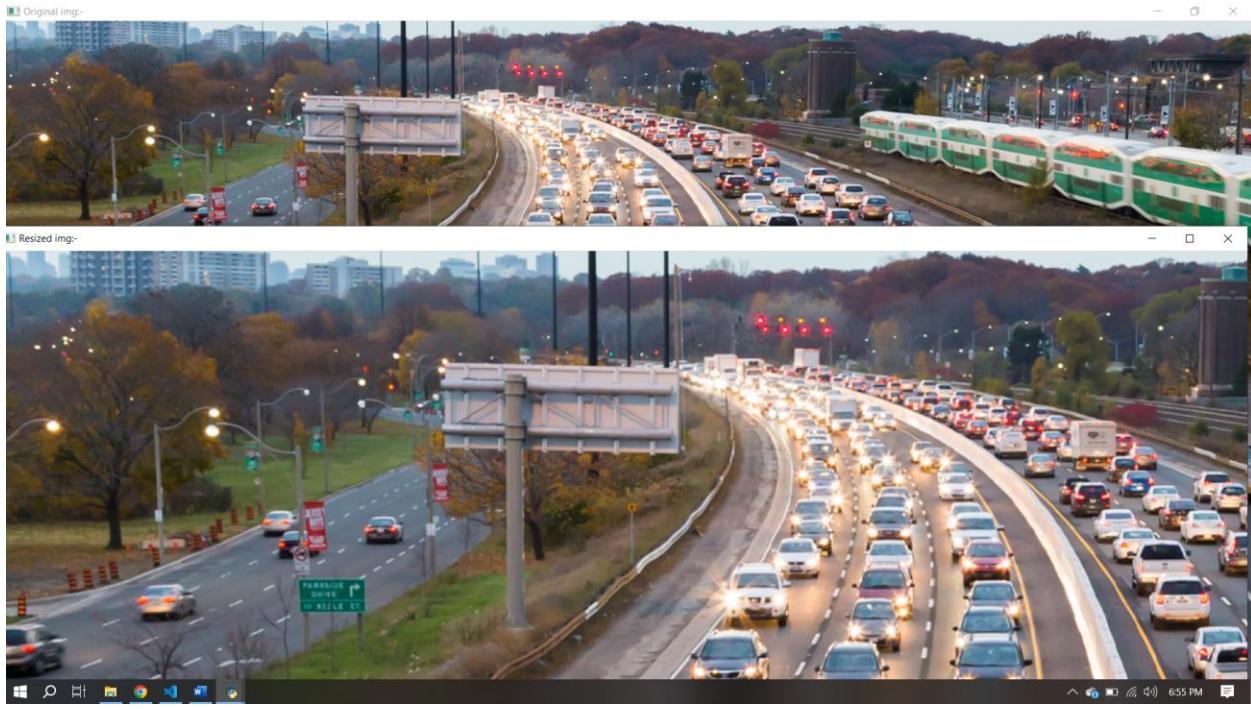
```
1 import cv2
2
3 img = cv2.imread("C:/Users/Admin/OneDrive/Desktop/cv.jpg")
4 print("Shape of Original img:- ", img.shape)
5
6 scale = 150
7
8 width = int(img.shape[1]* scale /100)
9 height= int(img.shape[0]*scale /100)
10 dim= (width,height)
11 resized = cv2.resize(img, dim, interpolation=cv2.INTER_AREA)
12
13 print("Shape of Resized img:- ", resized.shape)
14
15 cv2.imshow("Original img:- ", img)
16 cv2.imshow("Resized img:- ", resized)
17
18 cv2.waitKey(0)
19 cv2.destroyAllWindows()
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS D:\Open_cv_pkg> python -u "d:\Open_cv_pkg\cv4.py"
Shape of Original img:- (1080, 1920, 3)
Shape of Resized img:- (540, 960, 3)
PS D:\Open_cv_pkg> python -u "d:\Open_cv_pkg\cv4.py"
Shape of Original img:- (1080, 1920, 3)
Shape of Resized img:- (1620, 2880, 3)
```

OUTLINE TIMELINE

Ln 6, Col 10 Spaces: 4 UTF-8 CRLF {} Python Python 3.13 (64-bit) Go Live Prettier



### Adding Shape and Text:-

```

import cv2
import numpy as np

# img = cv2.imread("C:/Users/Admin/OneDrive/Desktop/cv.jpg", cv2.IMREAD_COLOR)
img = np.zeros(shape=[600,800,3],dtype='uint8') #zeros= balck background screen
# img = np.zeros(shape=[600,800])
img.fill(255) #white bg
cv2.line(img,(0,0),(150,150),(255,0,0),2) #img, point1, point2, color, thickness

cv2.rectangle(img, (200,150),(250,300),(0,255,0),3) #img,point1, point2, color, thickness

cv2.circle(img,(300,75),70,(255,0,255),3) #img,center,radius,color,thickness

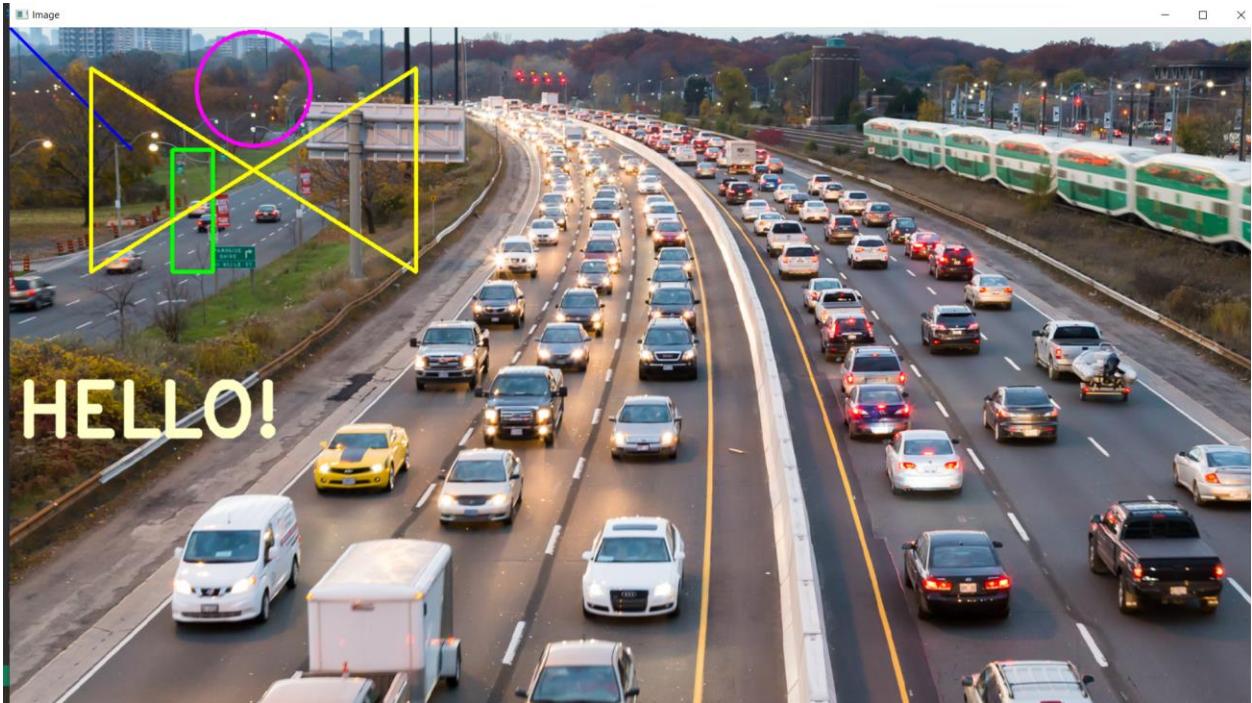
pts_polygon = np.array([[100,50],[100,300],[500,50],[500,300]], np.int32)
cv2.polyline(img,[pts_polygon],True,(0,255,255),3) #img,pts,isclosed=true,color, thickness

font = cv2.FONT_HERSHEY_DUPLEX
cv2.putText(img,
'HELLO!',(10,500),font,3,(200,255,255),8,cv2.LINE_AA) #img,text,org=bottom of corner,fontface,fontscale,color,thickness,line_type

cv2.imshow('Image', img)

```

```
cv2.waitKey(0)  
cv2.destroyAllWindows()
```

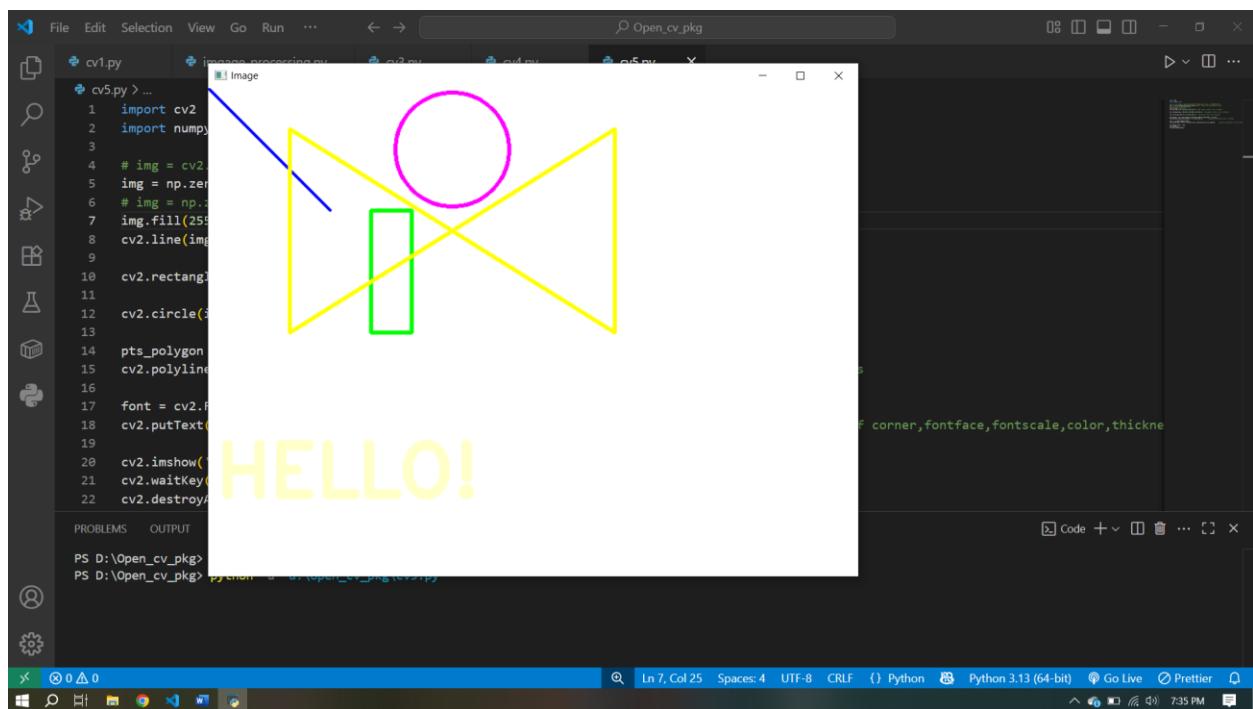
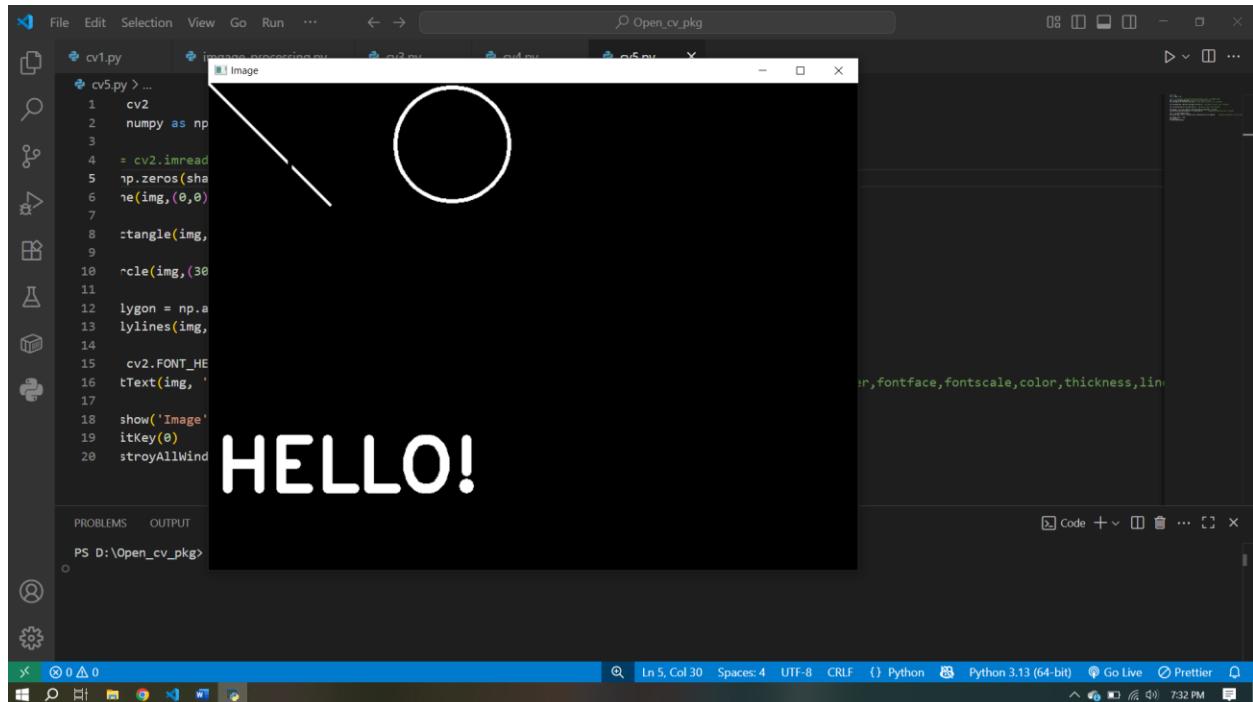


The screenshot shows a Python development environment with the following details:

- File Explorer:** Shows files `cv1.py`, `cv2.py`, `cv3.py`, `cv4.py`, and `cv5.py`.
- Code Editor:** The file `cv5.py` contains the following code:

```
cv1.py
cv2.py
cv3.py
cv4.py
cv5.py

cv1.py > ...
1 cv2
2 numpy as np
3
4 = cv2.imread
5 np.zeros(sh
6 ne(img,(0,0)
7
8 ctangle(img,
9
10 ~cle(img,(30
11
12 lygon = np.a
13 lylines(img,
14
15 cv2.FONT_HE
16 tText(img, '
17
18 show('Image'
19 itKey(0)
20 stroyAllWind
```
- Output Window:** Displays the output of the code execution, showing the image generated by OpenCV.
- Bottom Status Bar:** Shows the current line (Ln 5, Col 67), spaces (Spaces: 4), and encoding (UTF-8). It also indicates the file is Python 3.13 (64-bit) and includes links for Go Live and Prettier.



### Shifting an img:-

```

import cv2
import numpy as np

img = cv2.imread("C:/Users/Admin/OneDrive/Desktop/cv.jpg")

```

```

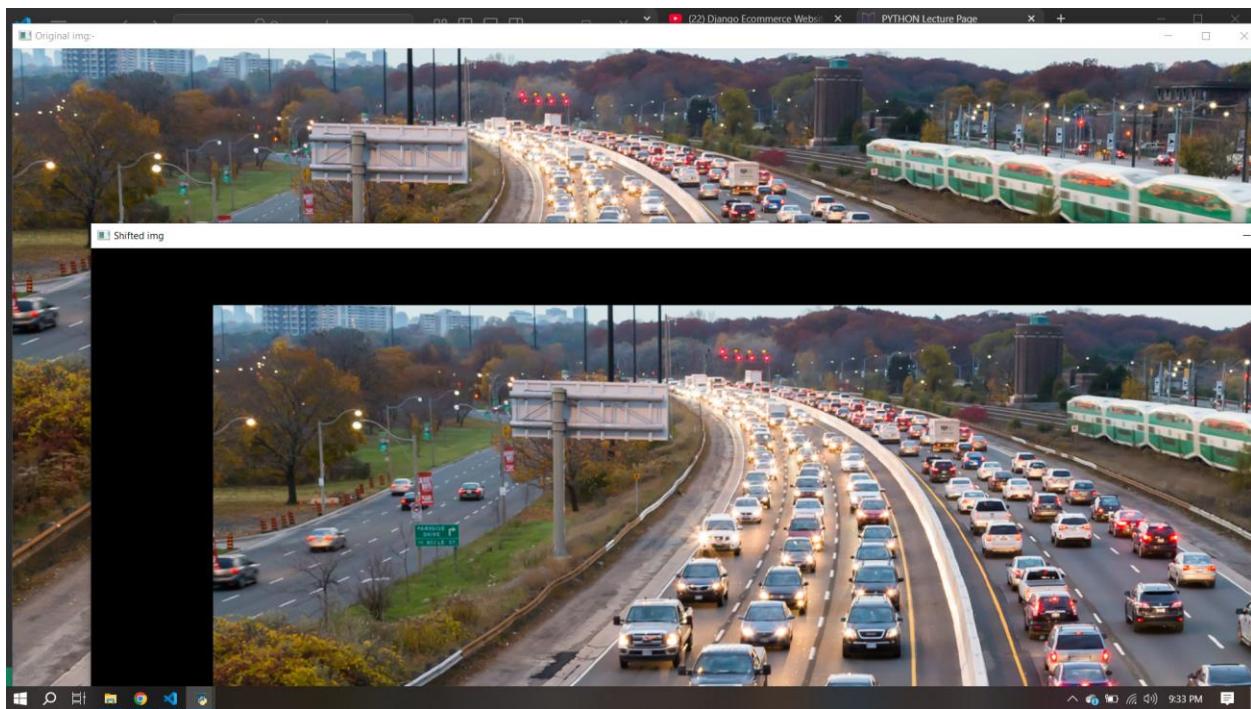
column = img.shape[1]
row = img.shape[0]

s = np.float32([(1,0,150),(0,1,70)])      #(right shiffting,height,150 ), (down
shifted = cv2.warpAffine(img,s,(column,row))

cv2.imshow("Original img:- ", img)
cv2.imshow("Shifted img", shifted)

cv2.waitKey(0)
cv2.destroyAllWindows()

```



### How to rotate img:-

```

import cv2
import numpy as np

img = cv2.imread("C:/Users/Admin/OneDrive/Desktop/cv.jpg")

column = img.shape[0]
row = img.shape[1]

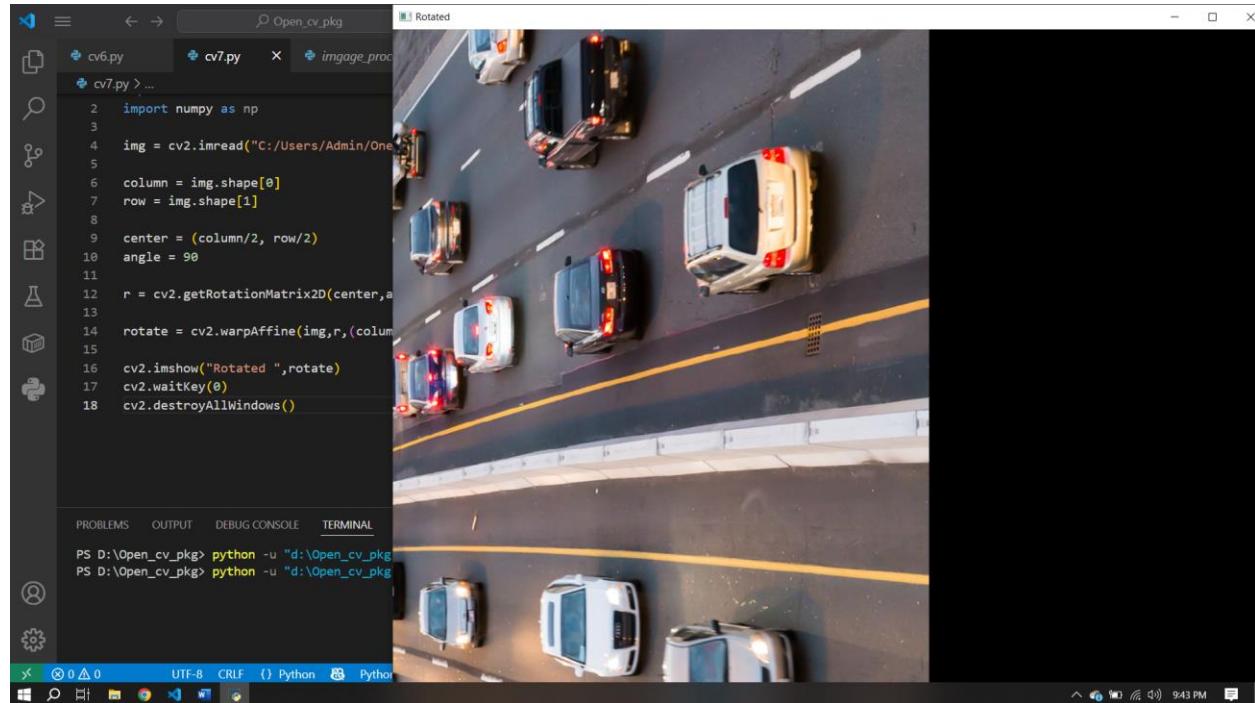
center = (column/2, row/2)
# angle = 90

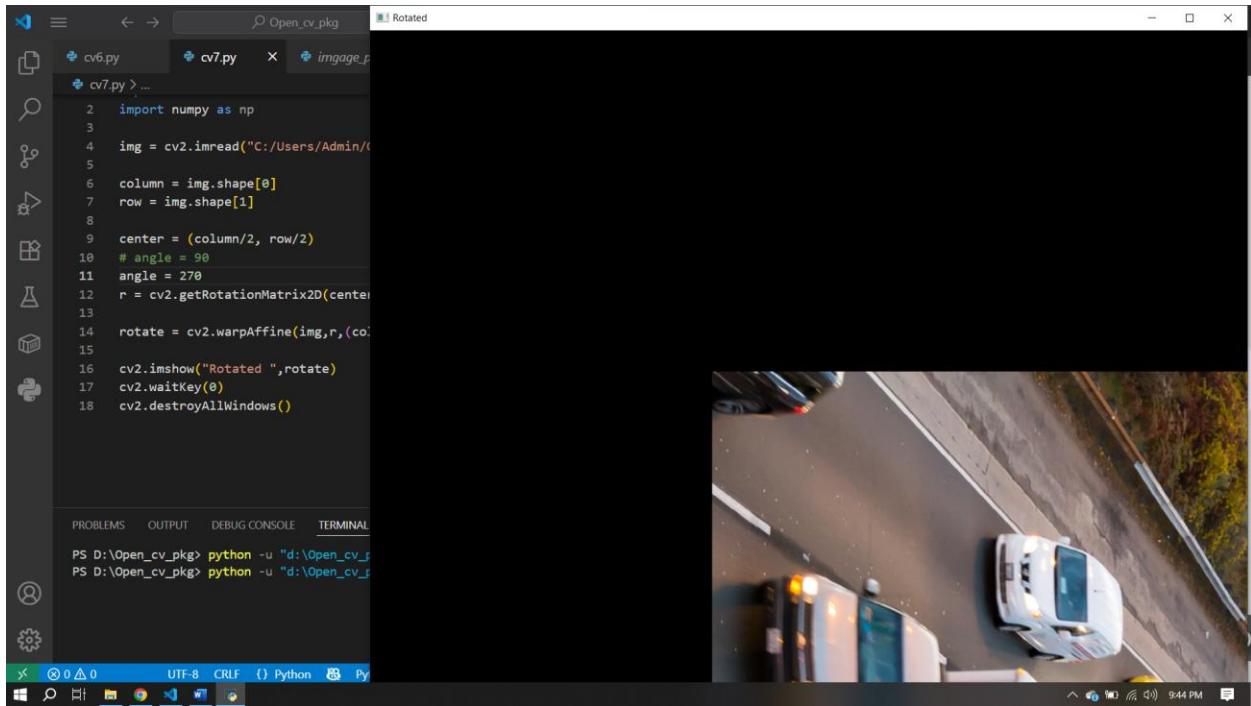
```

```
angle = 270
r = cv2.getRotationMatrix2D(center,angle,1)

rotate = cv2.warpAffine(img,r,(column,row))

cv2.imshow("Rotated ",rotate)
cv2.waitKey(0)
cv2.destroyAllWindows()
```





### Image thresholding: -

```
# CHANNEL -->0 TO 255

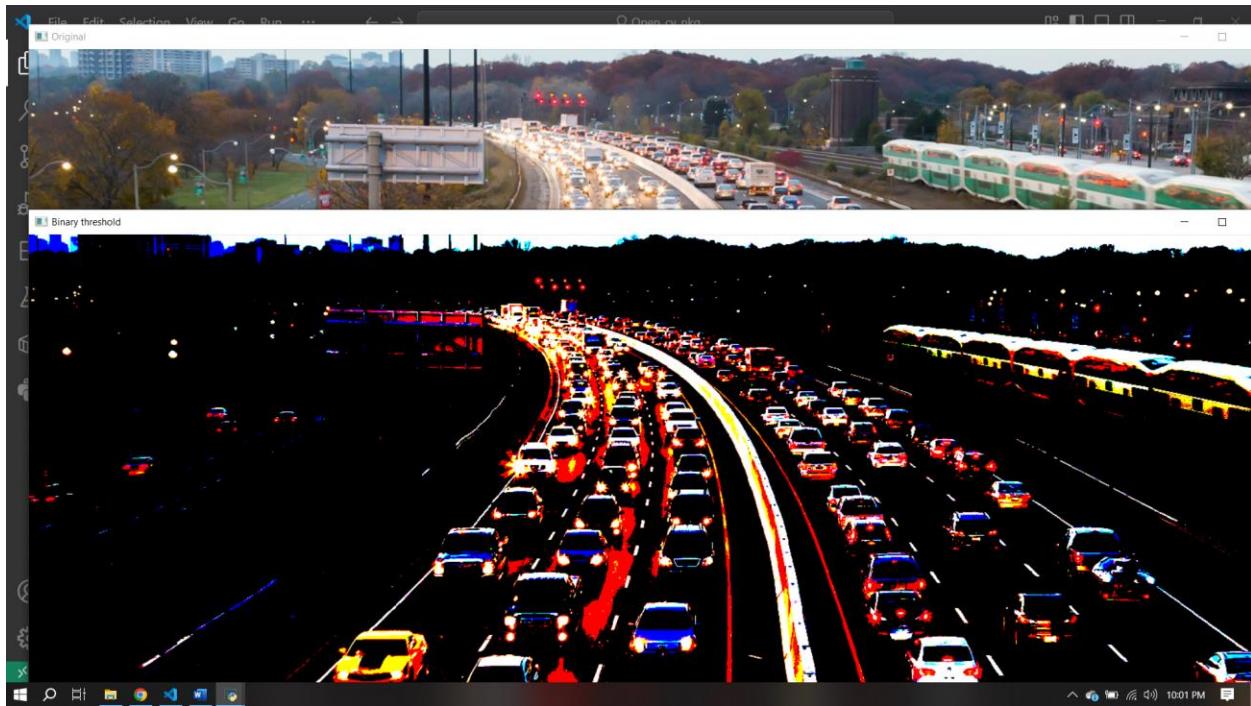
import cv2
import numpy as np

img = cv2.imread("C:/Users/Admin/OneDrive/Desktop/cv.jpg")

threshold_value= 200

_, binary_threshold = cv2.threshold(img,threshold_value,255,cv2.THRESH_BINARY)

cv2.imshow("Original ", img)
cv2.imshow("Binary threshold", binary_threshold)
cv2.waitKey(0)
cv2.destroyAllWindows()
```



### Gaussian blur:-

```
import cv2
img = cv2.imread("C:/Users/Admin/OneDrive/Desktop/cv.jpg")

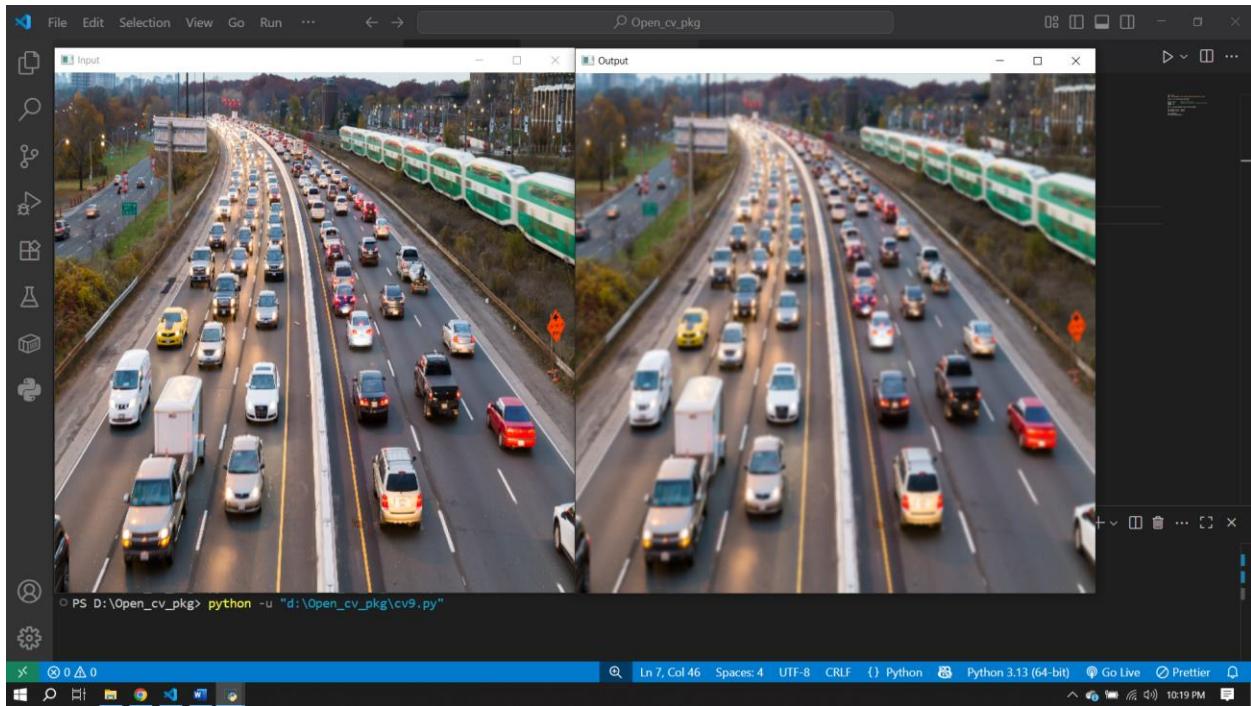
resize = cv2.resize(img,(640,640))

ksize = (7,7)      #ksize= kernel size
sigmaX = 0          #sigmax,y its 0 then its taken from ksize
sigmaY = 0

blur = cv2.GaussianBlur(resize,ksize,sigmaX)

cv2.imshow("Input", resize)
cv2.imshow("Output", blur)

cv2.waitKey(0)
cv2.destroyAllWindows()
```



### Median blur:-

```
import cv2
img = cv2.imread("C:/Users/Admin/OneDrive/Desktop/cv.jpg")

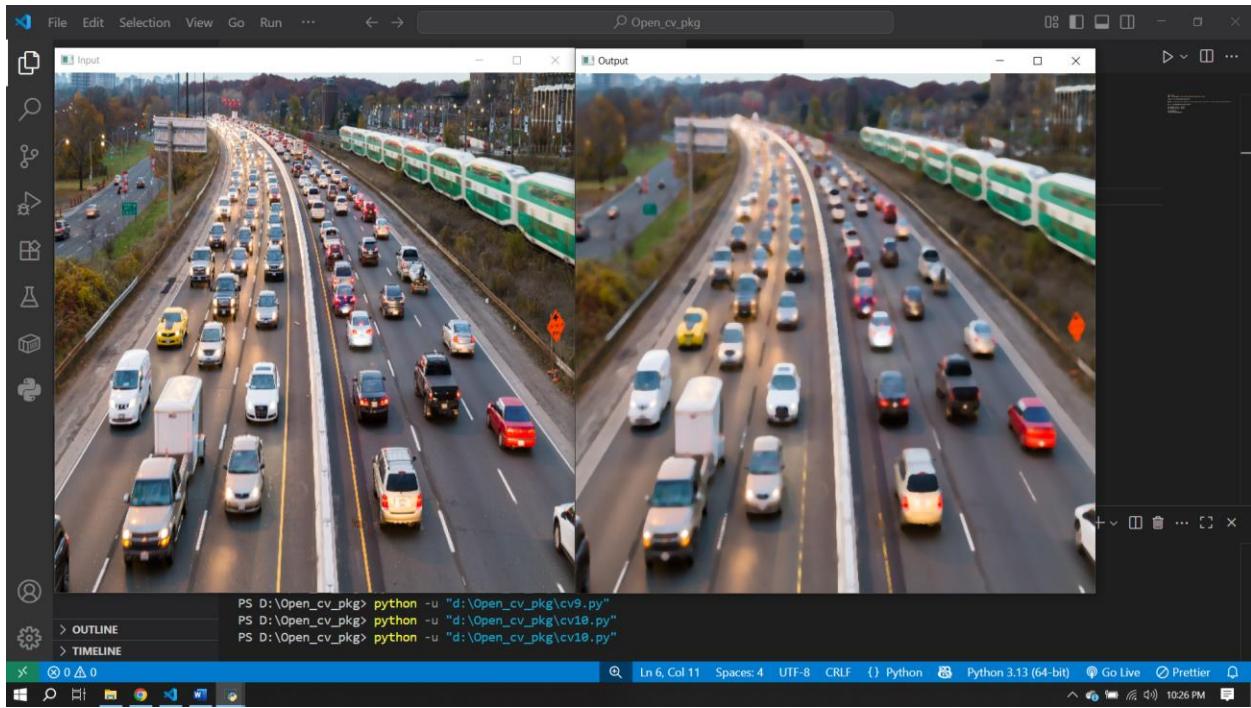
resize = cv2.resize(img,(640,640))

kernel = 7 #gaussian and median blur give same result only diff no need to give
           standard deviation

blur = cv2.medianBlur(resize,kernel)

cv2.imshow("Input", resize)
cv2.imshow("Output", blur)

cv2.waitKey(0)
cv2.destroyAllWindows()
```



### Bilateral Filter-

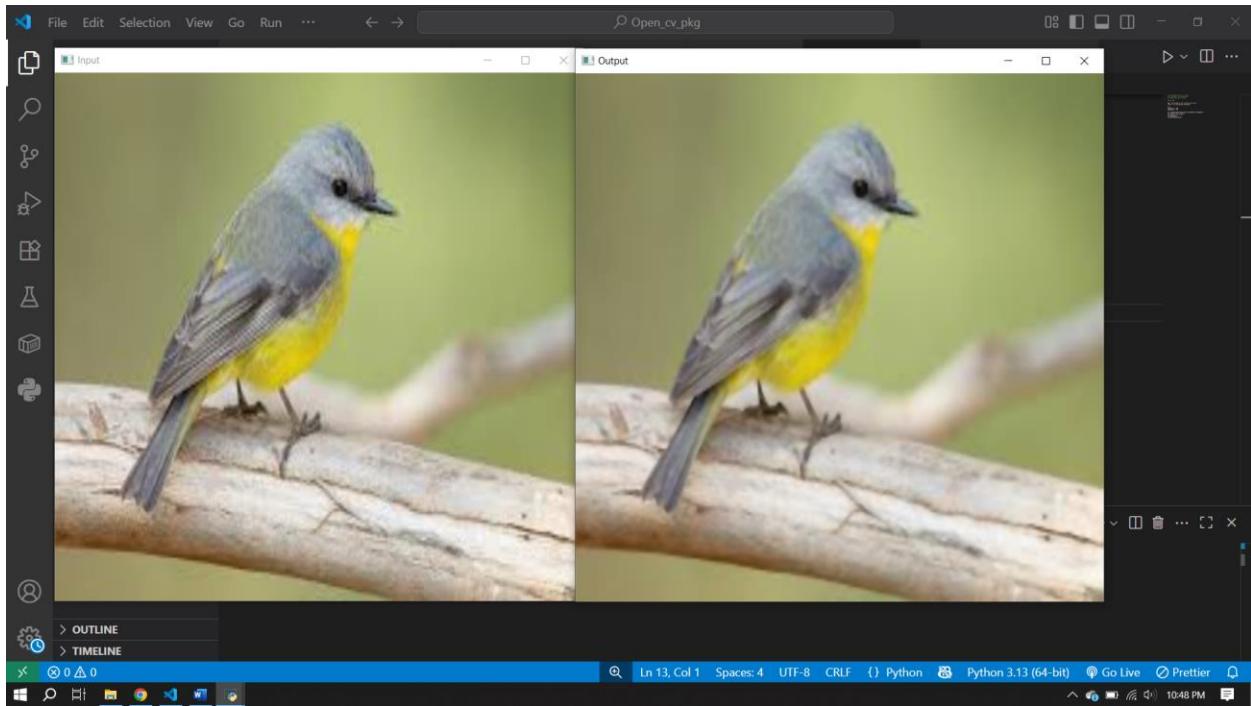
```
# In bilateral filter:- 1.src=ip
# 2.d= diameter of neighborhood
# 3.sigma color and 4. space

import cv2

img = cv2.imread("D:/web_scraping/bird.jpeg")
resize = cv2.resize(img, (650,650))

d= 7
sigmaColor = 100
sigmaSpace = 100

b = cv2.bilateralFilter(resize,d,sigmaColor,sigmaSpace)
cv2.imshow("Input",resize)
cv2.imshow("Output", b)
cv2.waitKey(0)
cv2.destroyAllWindows()
```



### Edge detection:-

```
# Canny edge detection
# Noise Reduction
# Intensity of the gradient of the img
# non-max suppression
# thresholding
import cv2

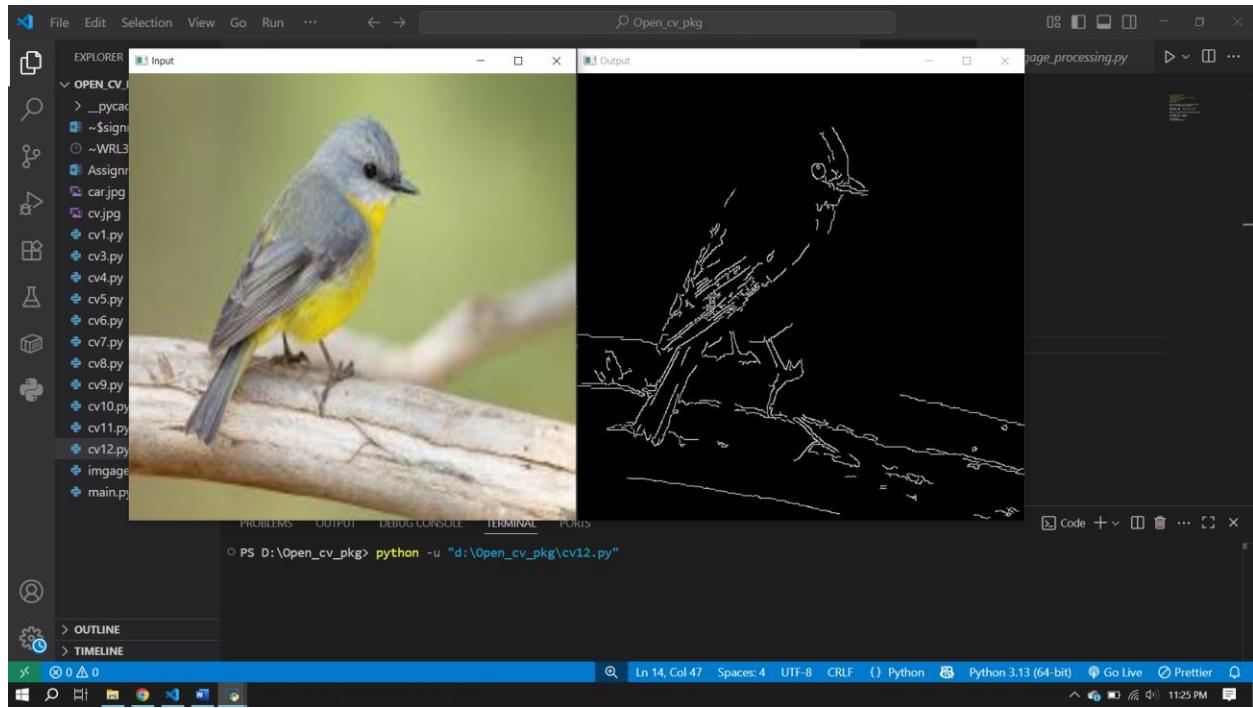
img = cv2.imread("D:/web_scraping/bird.jpeg")
resize = cv2.resize(img, (550,550))

min_thresh = 100      #below 100 its black
max_thresh = 200      #above 200 its white

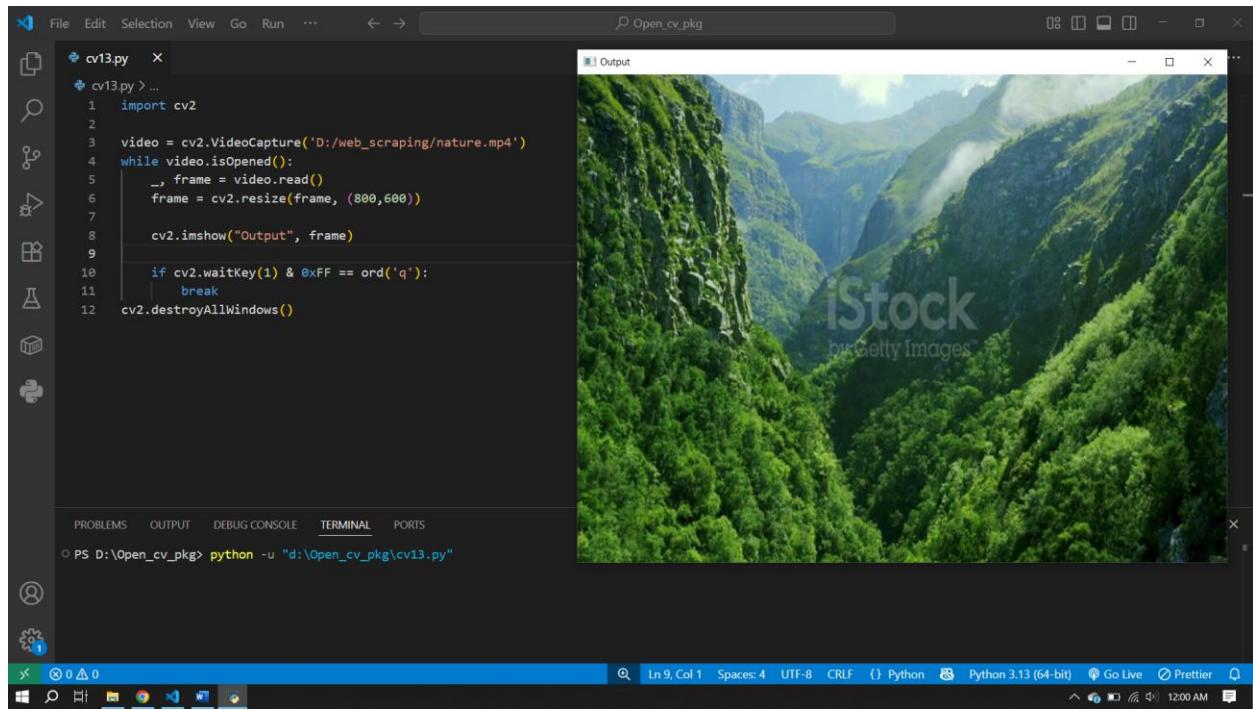
edges = cv2.Canny(resize,min_thresh,max_thresh)

cv2.imshow("Input", resize)
cv2.imshow("Output",edges)

cv2.waitKey(0)
cv2.destroyAllWindows()
```



## Reading and writing a video:-



```
import cv2

video = cv2.VideoCapture('D:/web_scraping/nature.mp4')
```

```

fourcc = cv2.VideoWriter_fourcc(*'mp4v')           #fourcc= compress file
output = cv2.VideoWriter('output.mp4', fourcc,23.98,(1767,1769) )   #frame,
parameter in properties.details

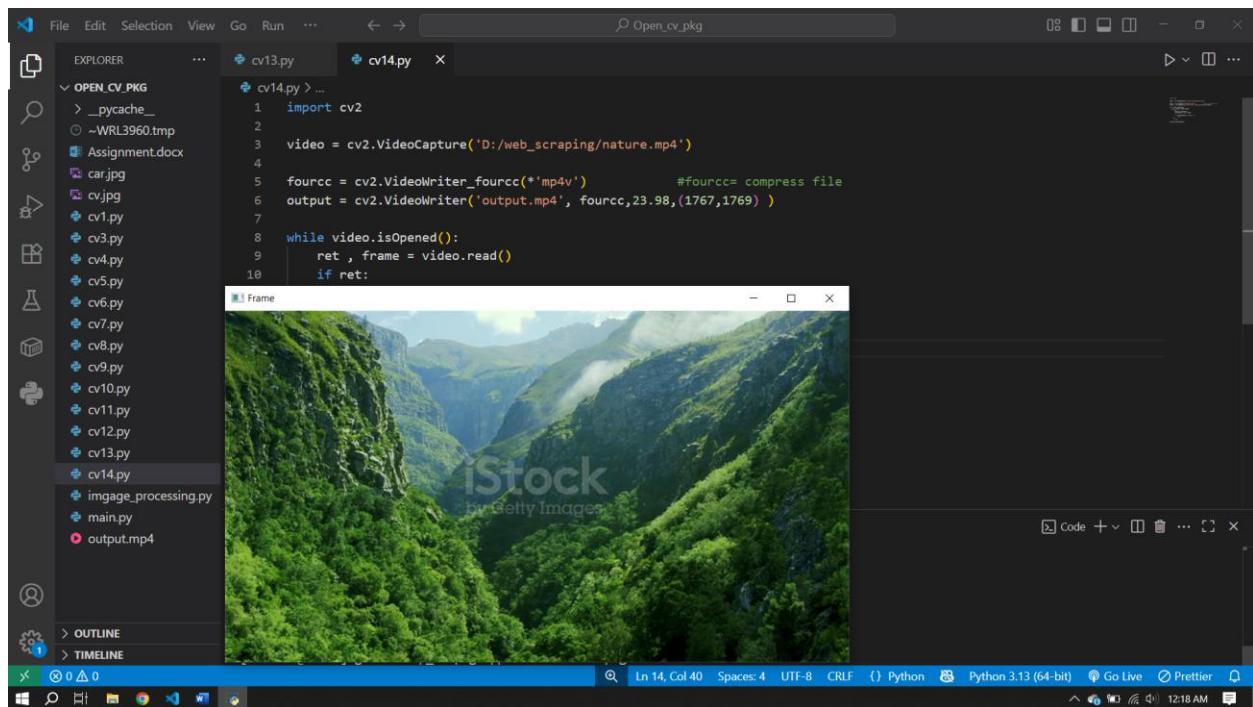
while video.isOpened():
    ret , frame = video.read()
    if ret:
        output.write(frame)
        cv2.imshow('Frame',frame)

    if cv2.waitKey(10) == ord('s'):
        break

else:
    break

cv2.destroyAllWindows()

```



**Accessing the webcam:-**

The screenshot shows a Windows desktop environment with a Visual Studio Code (VS Code) window open. The code editor displays a Python script named `cv15.py` which reads frames from a webcam and shows them in a window. The script is as follows:

```
import cv2
cap = cv2.VideoCapture(0) #passing 0 to access webcam
while cap.isOpened():
    _, frame = cap.read()
    cv2.imshow("Live", frame)
    if cv2.waitKey(10) == ord('z'):
        break
cv2.destroyAllWindows()
```

The terminal tab in VS Code shows the command `python -u "d:\Open_cv_pkg\cv15.py"` being run, with some FFMPEG warning messages displayed.

A live video feed from a webcam is overlaid on the VS Code interface, showing a person's face. The video feed has a watermark for "TuteDude" and text advertising "Enjoy 100% Refund On Course Completion" and "Level Up Your Career With Expert Mentorship & Internships For FREE".