OPENCV CHEATSHEET

python YoungWonks

INSTALLATION

```
pip3 install opencv-python
```

MINIMUM CODE

Minimum Code

```
#import the module
import cv2
#waits for a key to be pressed
cv2.waitKey()
#closes the OpenCV windows
cv2.destroyAllWindows()
```

READ AN IMAGE

Read

```
img=cv2.imread("path-of-the-image")
```

SHOW AN IMAGE

SHOW IMAGE

```
cv2.imshow("OpenCV Window", img)
```

RESIZE AN IMAGE

Resize

```
img=cv2.imread("path-of-the-image")
img1 = cv2.resize(img, (new_width, new_height))
cv2.imshow("OpenCV Window", img)
cv2.imshow("OpenCV Resize", img1)
cv2.waitKey()
cv2.destroyAllWindows()
```

READ AND SHOW A VIDEO

READ VIDEO FEED

```
videoObject = cv2.VideoCapture('path-of-the-video')
#argument in videocapture can be the portnumber of the webcam
while True:
    #read the video feed frame by frmae
    var, frame = videoObject.read()
    #show the video feed
    cv2.imshow('OpenCV Video', frame)
    #wait for the key to be pressed
    if cv2.waitKey(1) and 0xFF == ord('q'):
       break
#stop the capture
videoObject.release()
#close all windows
cv2.destroyAllWindows()
```

OPENCV CHEATSHEET

python YoungWonks

CONVERT INTO GRAYSCALE IMAGE

GRAYSCALE

```
img=cv2.imread("path-of-the-image")
gray_img = cv2.cvtColor (img, cv2.COLOR_BGR2GRAY)
```

INVERT AN IMAGE

INVERT

```
img=cv2.imread("path-of-the-image")
invert_img=cv2.bitwise_not(img)
```

FLIP AN IMAGE

FLIP

```
img=cv2.imread("path-of-the-image")
#0 for horizontal and 1 for vertical flip
flip_img=cv2.flip(img, 0)
```

BLUR AN IMAGE

BLUR

```
img=cv2.imread("path-of-the-image")
# kernel is a matrix of ones. Row/Column numbers = ODD
kernel = np.ones((rows, columns))
img=cv2.imread("path-of-the-image")
blur_img=cv2.blur(img, (kernel))
blur_img=cv2.blur(img, (kernel_width, kernel_height))
```

DRAW SHAPES ON AN IMAGE

Draw Shapes

```
img=cv2.imread("path-of-the-image")
img = cv2.rectangle(img, (x1, y1), (x2, y2), color, thickness)
img = cv2.circle(img, circle_center, radius, color, thickness)
img = cv2.line(img, start, end, color, thickness)
```

ADD TEXT TO AN IMAGE

Add text

```
img=cv2.imread("path-of-the-image")
cv2.putText(img, text, (x, y), font style, fontscale, color, thickness)
```

OPENCV CHEATSHEET

python YoungWonks

DILATE AN IMAGE

DILATE

```
img=cv2.imread("path-of-the-image")
# kernel is a matrix of ones. Row/Column numbers = ODD
kernel = np.ones((rows, columns))
img=cv2.imread("path-of-the-image")
dilated_img= cv2.dilate(img, (kernel))
```

ERODE AN IMAGE

ERODE

```
img=cv2.imread("path-of-the-image")
# kernel is a matrix of ones. Row/Column numbers = ODD
kernel = np.ones((rows, columns))
img=cv2.imread("path-of-the-image")
dilated_img= cv2.erode(img, (kernel))
```

THRESHOLDING AN IMAGE

THRESHOLDING

```
img=cv2.imread("path-of-the-image")
#first convert image into greyscale
gray_img = cv2.cvtColor (img, cv2.COLOR_BGR2GRAY)
var, threshold_img = cv2.threshold(gray_img, threshold_value, maximum_pixel_value, cv2.THRESH_BINARY)
#all pixels with values lesser than the lower threshold value will convert to black and the other pixels to white.
cv2.imshow("OpenCV THRESHOLD Image", threshold_img)
```

CANNY EDGE DETECTION TECHNIQUE

EDGE DETECTION

```
img=cv2.imread("path-of-the-image")
img_edge = cv2.Canny(img, threshold_value1, threshold_value2)
#smallest value between threshold value1 and threshold value2 is used for edge linking
#largest value is used to find initial segments of strong edges
```

WRITE AN IMAGE

WRITE

```
img=cv2.imread("path-of-the-image")
cv2.imwrite('new image path', img_edge)
```

WRITE VIDEO FRAMES AS IMAGES

WRITE VIDEO FRAMES AS IMAGES

```
videoObject = cv2.VideoCapture('path-of-the-video')
# count each frame of the video
counter=0
while True:
    var, frame = videoObject.read()
    cv2.imshow('OpenCV Video', frame)
    cv2.imwrite("NewFrame"+str(counter)+".jpg", frame)
    # write each frame of the video as an image
    counter=counter+1
    if counter==10:
        break
    if cv2.waitKey(1) and 0xFF == ord('q'):
        break
```