​PYRAMID

import cv2

image = cv2.imread('c:\\users\\egek\\images\\ataturk.jpg')

smaller = cv2.pyrDown(image)

larger = cv2.pyrUp(smaller)

cv2.imshow('Original', image )

cv2.imshow('Smaller ', smaller )

cv2.imshow('Larger ', larger )

cv2.waitKey(0)

cv2.destroyAllWindows()

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

CROP

import cv2

import numpy as np

image = cv2.imread('c:\\users\\egek\\images\\ataturk.jpg')

height, width = image.shape[:2]

start\_row, start\_col = int(height \* .05), int(width \* .30)

end\_row, end\_col = int(height \* .75), int(width \* .75)

cropped = image[start\_row:end\_row , start\_col:end\_col]

cv2.imshow("Original Image", image)

cv2.waitKey(0)

cv2.imshow("Cropped Image", cropped)

cv2.waitKey(0)

cv2.destroyAllWindows()

INCREASE / DECREASE BRIGHTNESS

import cv2

import numpy as np

image = cv2.imread('c:\\users\\egek\\images\\ataturk.jpg')

M = np.ones(image.shape, dtype = "uint8") \* 75

added = cv2.add(image, M)

cv2.imshow("Added", added)

subtracted = cv2.subtract(image, M)

cv2.imshow("Subtracted", subtracted)

cv2.waitKey(0)

cv2.destroyAllWindows()

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

BITWISE

import cv2

import numpy as np

square = np.zeros((300, 300), np.uint8)

cv2.rectangle(square, (50, 50), (250, 250), 255, -2)

cv2.imshow("Square", square)

cv2.waitKey(0)

ellipse = np.zeros((300, 300), np.uint8)

cv2.ellipse(ellipse, (150, 150), (150, 150), 30, 0, 180, 255, -1)

cv2.imshow("Ellipse", ellipse)

cv2.waitKey(0)

# Shows only where they intersect

And = cv2.bitwise\_and(square, ellipse)

cv2.imshow("AND", And)

cv2.waitKey(0)

# Shows where either square or ellipse is

bitwiseOr = cv2.bitwise\_or(square, ellipse)

cv2.imshow("OR", bitwiseOr)

cv2.waitKey(0)

# Shows where either exist by itself

bitwiseXor = cv2.bitwise\_xor(square, ellipse)

cv2.imshow("XOR", bitwiseXor)

cv2.waitKey(0)

# Shows everything that isn't part of the square

bitwiseNot\_sq = cv2.bitwise\_not(square)

cv2.imshow("NOT - square", bitwiseNot\_sq)

cv2.waitKey(0)

cv2.destroyAllWindows()