Drawing Lab

Read an image
img = cv2.imread(<image_filename>)

Create a 500x500 BLACK pixel image
img = np.zeros((500, 500, 3), np.uint8)

Create a 500x500 WHITE pixel image
img = np.full((500, 500, 3), (255,255,255), np.uint8)

Show Image in Popup Window
cv2.imshow(<window_name>, <image>)
close_windows()

Show Image Inline (instructor-written function!)
show_inline(<image>)

Tool to Check OpenCV Coordinates (instructor-written function!)
coordinates()

Draw a Line
cv2.line(<image>, <start_coordinate>, <end_coordinate>, <color>, <thickness>)

Draw a Circle
cv2.circle(<image>, <center_coordinate>, <radius>, <color>, <thickness>)

Draw a Rectangle
cv2.rectangle(<image>, <topleft_coordinate>, <bottomright_coordinate>, <color>, <thickness>)

Draw Text
cv2.putText(<image>, <text>, <bottomleft_coord>, <font>, <scale>, <color>, <thickness>)

Color Codes
Black: (0,0,0)
White: (255,255,255)
Blue: (255,0,0)
Green: (0,255,0)
Red: (0,0,255)
Yellow: (0,255,255)

Color Spaces

Change Colorspace
img = cv2.cvtColor(<image>, <flag>)

Flags:
BRG -> Gray: cv2.COLOR_BGR2GRAY
BGR -> HSV: cv2.COLOR_BGR2HSV

Color Trackbar

Create a Trackbar
cv2.createTrackbar(<trackbar_name>, <window_name>, <min_value>, <max_value>, <callback_function>)

Reading a Trackbar
trackbar_value = cv2.getTrackbarPos(<trackbar_name>, <window_name>)

Detect Mouse
cv2.setMouseCallback(<window_name>, <callback_function>)

Birdie Mask

Mask an image based on HSV values (instructor-written function!)
hsv_select(filename)

Mask an image between specific HSV values
mask = cv2.inRange(<image>, <hsv_lower>, <hsv_upper>)

Show the real colors in a mask
color_mask = cv2.bitwise_and(<image1>, <image2>, mask=<input_mask>)

Reverse masks
inv_mask = cv2.bitwise_not(<mask_to_invert>)

Green Screen

Overlay front image on background (instructor-written function!)

res = screenProcessing(<front_img>, <background_img>, <hsv_lower>,< hsv_upper>)

Open a live video (instructor-written function!)
video(<function>)

Mask a live video based on HSV values (instructor-written function!)
hsv_select_live()

Painter

Find the center coordinates of the contour (instructor-written function!)
center = find_center(<single_contour>)

Find the radius of the contour (instructor-written function!)
radius = find_radius(<single_contour>)

Contours

Threshold an Image
thresh = cv2.threshold(<grayscale_image>, <threshold_value>, <maxVal>, <minVal>)[1]

Find Contours
contours = cv2.findContours(<mask>, 3, 2)[1]
Flag Notes
3 - cv2.RETR_TREE
2 - cv2.CHAIN_APPROX_SIMPLE

Draw Contours
cv2.drawContours(<image>, <contours>, <contour_index>, <color>, <thickness>)
Flag Notes
To draw all contours: <contour_index>=-1
To draw specific contour: <contour_index> = 0, <contours> = [contour[i]]

Get contour area
area = cv2.contourArea(<single_contour>)

Find Straight Bounding Rectangle
x, y, w, h = cv2.boundingRect(<single_contour>)

Draw Straight Bounding Rectangle
cv2.rectangle(img, (x,y), (x+w,y+h), <color>, <thickness>)

Find Minimum Area Rectangle
rect = cv2.minAreaRect(<single_contour>)

Draw Minimum Area Rectangle
box = np.int0(cv2.boxPoints(rect))
res = cv2.drawContours(img, [box], 0, (0,0,255), 2)

Find Minimum Enclosing Circle
(x,y), radius = cv2.minEnclosingCircle(<single_contour>)

Draw Minimum Enclosing Circle
cv2.circle(img, (x,y), radius, <color>, <thickness>)

Feature Detection Lab

Draw keypoints
cv2.drawKeypoints(<image>, <keypoints>, <image>, flags=5)

Compute Keypoints
keypoints, descriptors = sift.detectAndCompute(<grayscale_image>, None)
keypoints , descriptors = surf.detectAndCompute(<grayscale_image>, None)
keypoints , descriptors = orb.detectAndCompute(<grayscale_image>, None

SURF Hessian Thresholds
Bigger hessianThresholds -> Less keypoints. Usually between 300 to 500.

ORB nFeatures
nFeatures sets the maximum number of features found in the image. The default value is 500.

Find matches between keypoints
matches = flann.knnMatch(<description1>, <description2>, k=<num_best_matches>)

Get Keypoints Between SIFT, SURF, and ORB
img = cv2.imread(filepath)
find_keypoints(img, <feature_detection_algorithm>)
(note: <feature_detection_algorithm> accepts “sift”, “surf”, or “orb”)

Edge Detection Lab

Sobel Detection
res = cv2.Sobel(<image>, cv2.CV_64F, <show_vertical>, <show_horizontal>, ksize=5)
(note for show_vertical/show_horizontal: use 0 for False, 1 for True)

Canny Edge Detection
res = cv2.Canny(<image>, <min_threshold>, <max_threshold>)

Get Hough Lines
lines = get_hough_lines(<image>)

Draw Hough Lines
draw_hough_lines(<image>, lines)

More OpenCV Resources

Online Python 3 → Want to practice coding at home? Do it here!
**** Make sure to select "Python 3" in "Language" ****
https://www.onlinegdb.com/

OpenCV Tutorials
https://docs.opencv.org/3.4.2/d6/d00/tutorial_py_root.html

Stack Overflow → Got coding questions? Other people do too! Ask them here :)
**** Type your question in the "Search" box above! ****
https://stackoverflow.com/