Learning-based Multimedia Processing

2022/2023

Lab classes (4)

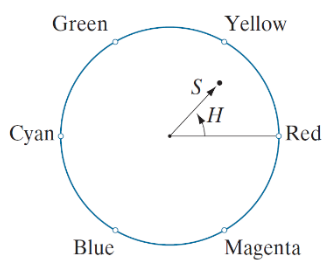
1 Colour image processing

Colour masking

Creating a mask based on the image colour is an example of a colour manipulation.

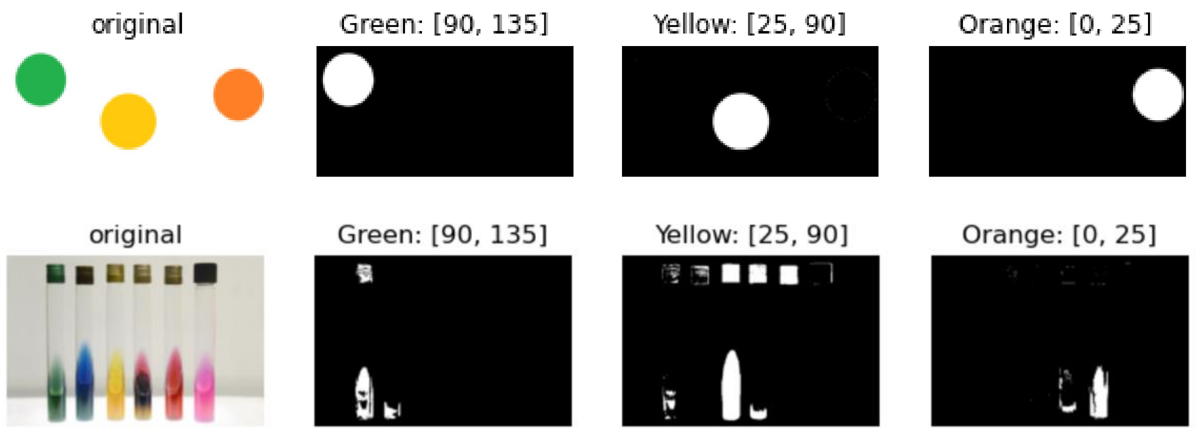
Function `cv2.inRange()` checks if array elements lie between the elements of two parameter arrays:

```
cv2.inRange(src, lowerb, upperb[, dst])
```



Example to detect green areas in image:

```
img = cv2.imread("images/t06.jpg")
hsv_img = cv2.cvtColor(img, cv2.COLOR_BGR2HSV)

# Define colour range of interest: Green
min_angle = 90
max_angle = 135

lower_value = np.array([int(min_angle/360*255), 50, 50])
upper_value = np.array([int(max_angle/360*255), 255, 255])

mask = cv2.inRange(hsv_img, lower_value, upper_value)
```

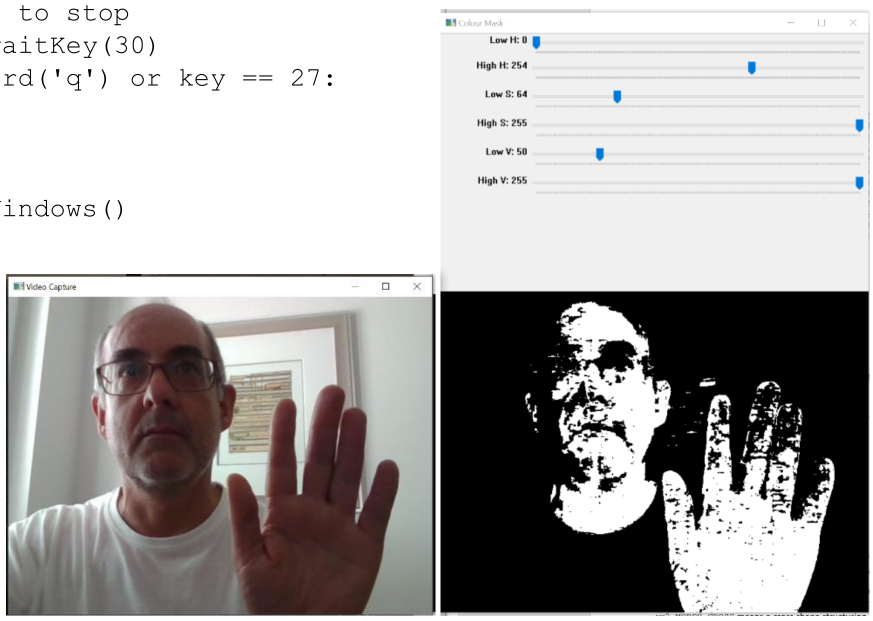


Capture video and apply H, S, V thresholds for masking

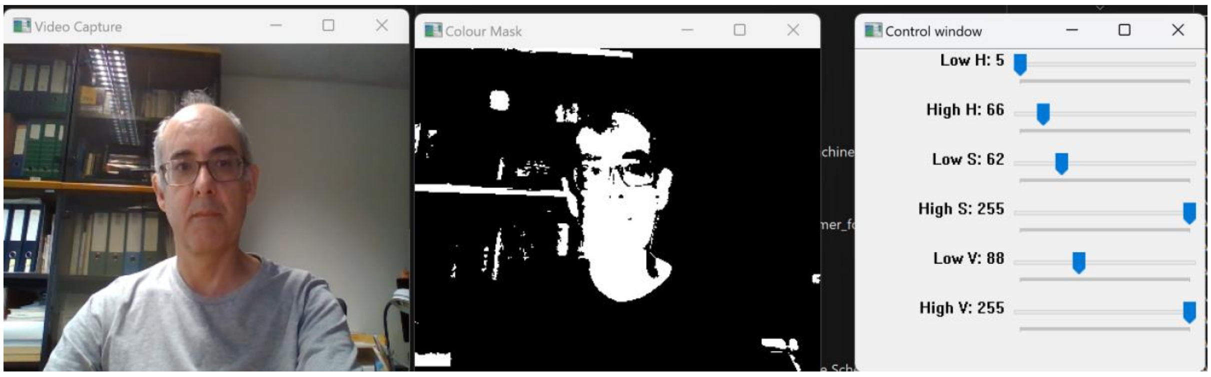
```
import cv2

min_value_H = 0
max_value_H = 360
low_H = int(min_value_H/360*255)
high_H = int(max_value_H/360*255)

min_value = 50
max_value = 255
low_V = min_value
high_V = max_value

low_S = min_value
high_S = max_value

window_capture_name = 'Video Capture'
window_detection_name = 'Colour Mask'
window_control_name = 'Control window'

low_H_name = 'Low H'
low_S_name = 'Low S'
low_V_name = 'Low V'
high_H_name = 'High H'
high_S_name = 'High S'
high_V_name = 'High V'

def empty(i):
    pass

def on_low_H_thresh_trackbar(val):
    global low_H
    global high_H
    low_H = int(val/360*255)
    low_H = min(high_H-1, low_H)
    cv2.setTrackbarPos(low_H_name, window_control_name,
int(low_H*360/255))


def on_high_H_thresh_trackbar(val):
    global low_H
    global high_H
    high_H = int(val/360*255)
    high_H = max(high_H, low_H+1)
    cv2.setTrackbarPos(high_H_name, window_control_name,
int(high_H*360/255))


def on_low_S_thresh_trackbar(val):
    global low_S
    global high_S
    low_S = val
    low_S = min(high_S-1, low_S)
    cv2.setTrackbarPos(low_S_name, window_control_name, low_S)
```

```
def on_high_S_thresh_trackbar(val):
    global low_S
    global high_S
    high_S = val
    high_S = max(high_S, low_S+1)
    cv2.setTrackbarPos(high_S_name, window_control_name, high_S)


def on_low_V_thresh_trackbar(val):
    global low_V
    global high_V
    low_V = val
    low_V = min(high_V-1, low_V)
    cv2.setTrackbarPos(low_V_name, window_control_name, low_V)


def on_high_V_thresh_trackbar(val):
    global low_V
    global high_V
    high_V = val
    high_V = max(high_V, low_V+1)
    cv2.setTrackbarPos(high_V_name, window_control_name, high_V)



# Start video capture
cap = cv2.VideoCapture(0)


# create windows to display capture video and computed colour mask
cv2.namedWindow(window_capture_name)
cv2.namedWindow(window_detection_name)
cv2.namedWindow(window_control_name)


# create trackbars
cv2.createTrackbar(low_H_name, window_control_name , low_H,
max_value_H, on_low_H_thresh_trackbar)
cv2.createTrackbar(high_H_name, window_control_name , high_H,
max_value_H, on_high_H_thresh_trackbar)
cv2.createTrackbar(low_S_name, window_control_name , low_S,
max_value, on_low_S_thresh_trackbar)
cv2.createTrackbar(high_S_name, window_control_name , high_S,
max_value, on_high_S_thresh_trackbar)
cv2.createTrackbar(low_V_name, window_control_name , low_V,
max_value, on_low_V_thresh_trackbar)
cv2.createTrackbar(high_V_name, window_control_name , high_V,
max_value, on_high_V_thresh_trackbar)
```

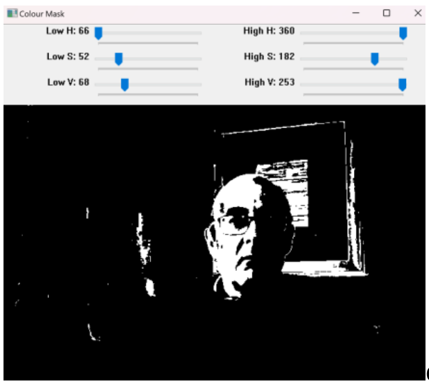
```
while True:
    ret, frame = cap.read()
    if frame is None:
        break

    frame_HSV = cv2.cvtColor(frame, cv2.COLOR_BGR2HSV)
    mask = cv2.inRange(frame_HSV, (low_H, low_S, low_V), (high_H,
high_S, high_V))

    # display images
    cv2.imshow(window_capture_name, frame)
    cv2.imshow(window_detection_name, mask)

    # press "q" to stop
    key = cv2.waitKey(30)
    if key == ord('q') or key == 27:
        break

cap.release()
cv2.destroyAllWindows()
```



Low H: 0
High H: 254
Low S: 64
High S: 255
Low V: 50
High V: 255



Low H: 5
High H: 66
Low S: 62
High S: 255
Low V: 88
High V: 255



Low H: 66
High H: 360
Low S: 52
High S: 182
Low V: 68
High V: 253

q