Tutorial - Colored object tracking using HSV

e-Yantra Team

June 30, 2016

Contents

1 Objective

The objective of this tutorial is to track a colored object using a Camera/Webcam.

2 Prerequisites

User should have handy knowledge of the following for understanding this tutorial.

- Basics of Python.
- Basics of OpenCV.
- Basics of Image processing.
- Knowledge about BGR and HSV color spaces and conversions.

3 Hardware Requirement

- A Computer with an internal or external webcam.

4 Software Requirement

- Python 2.7.5
- OpenCV 2.4.9
- numpy 1.7.1
- **Note :** These are the versions we were working on while creating this tutorial.

5 Theory and Description

- Object detection and segmentation is the most important and challenging fundamental task of computer vision. It is a critical part in many applications such as image search, scene understanding, etc.The easiest way to detect and segment an object from an image is the color based methods . The object and the background should have a significant color difference in order to successfully segment objects using color based methods.

- Here we are using HSV Colorspace for Object detection instead of RGB Colorspace because unlike RGB, HSV separates luma, or the image intensity, from chroma or the color information.By using HSV/HSL we can also remove intensity,lightness which is not possible in RGB colorspace.Moreover HSV can also detect skin color ,fire color etc.
- As every pixel in the frame has different HSV values,we threshold(mask) the object by extracting it using the HSV values of it.We do this by masking all pixels by setting value 1(white) that have the HSV value as of the object and other pixels in the frame to 0 .
- After getting the binary image,we find contours bounding the white blobs in the frame and find their respective areas.Among all the bounded contours,the contour with the largest area would be our colored object.
- Now we generate a boundary bounding that contour.In this way we can identify a colored object in a frame.By repeating this process in every frame we can track the object.
- **Note :** This code works only, when your object (which you want to track) is the largest object of its color in the frame.

6 Experiment

The Python code using OpenCV and Numpy is given below.

```
##############################################
## Import numpy for numerical calculations
import numpy as np

## Import OpenCV for image processing
import cv2

##############################################
## Initialize webcam
cap = cv2.VideoCapture(0) ##use 1 in parameter instead of 0 for external
                          ##camera
## You can also use a video by giving location of the video in the
## parameter,If video is in the same directory of the python file
## u can use the video directly as
##cap = cv2.VideoCapture('sample.mov')
##If you are using video then uncomment previous statement
```

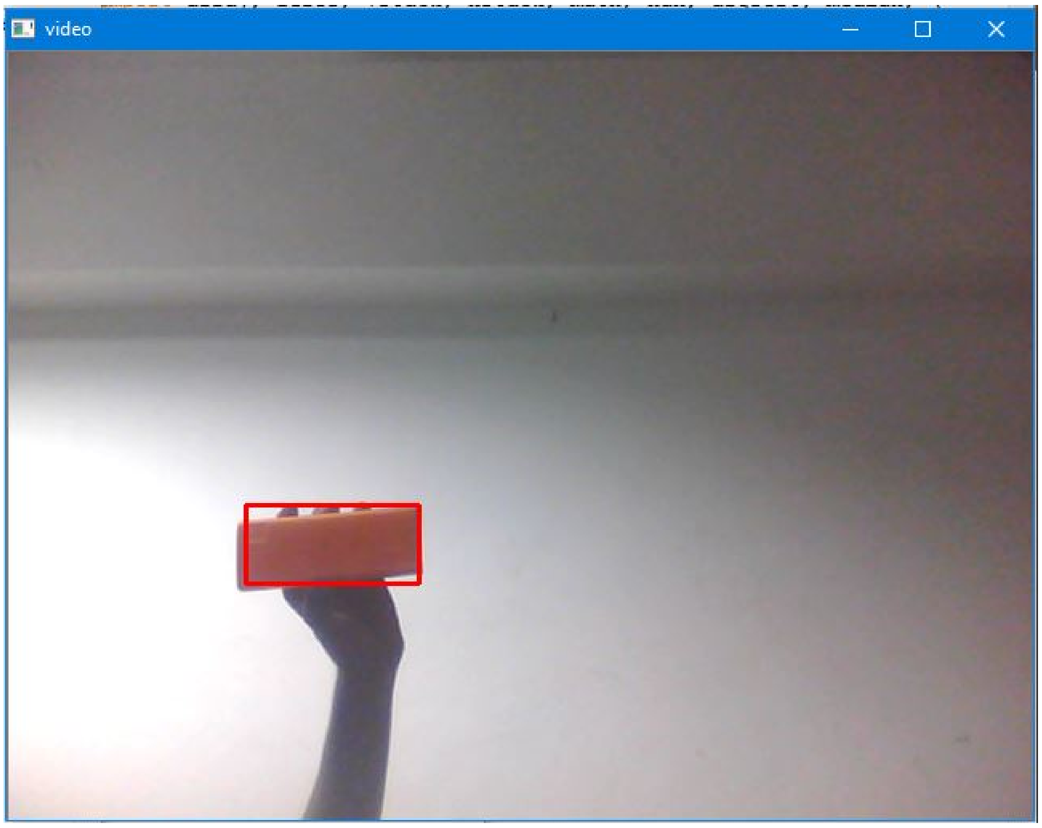
```
###############################################
## param1 and param2 are minimum and maximum range of hsv values for
## the following hsv values are of the Orange colored object
## which we are tracking
param1 = [0,100,50]        ## [H_min, S_min, V_min]
param2 = [30,255,255]      ## [H_max, S_max, V_max]
## You can put give range of the HSV values of any color and track
## that respective colored object

###############################################
## np.array will change param1 and param2 into numpy array which
## OpenCV can understand
lower = np.array(param1)
upper = np.array(param2)

###############################################
## Video Loop

while(1):

    ## Read one frame of the video at a time
    ## ret contains True if frame is successfully read otherwise it
    ## contains False
    ret, frame = cap.read()

    ## If frame is successfully read
    if(ret):

        ##Change color space of frame from BGR to HSV
        # and stores the converted frame in hsv
        hsv = cv2.cvtColor(frame,cv2.COLOR_BGR2HSV)

## Thresholding hsv frame and extracting the pixels of the desired color
        mask  = cv2.inRange(hsv, lower, upper)

        ## Removing noise from the Masked Frame (mask)
        mask = cv2.GaussianBlur(mask,(5,5),0)

        kernel = np.ones((5,5),np.uint8)
        mask = cv2.morphologyEx(mask, cv2.MORPH_OPEN, kernel)
        cv2.imshow('mask',mask)

        ##finding contours in mask
        ## and storing all the contours in contours array
```

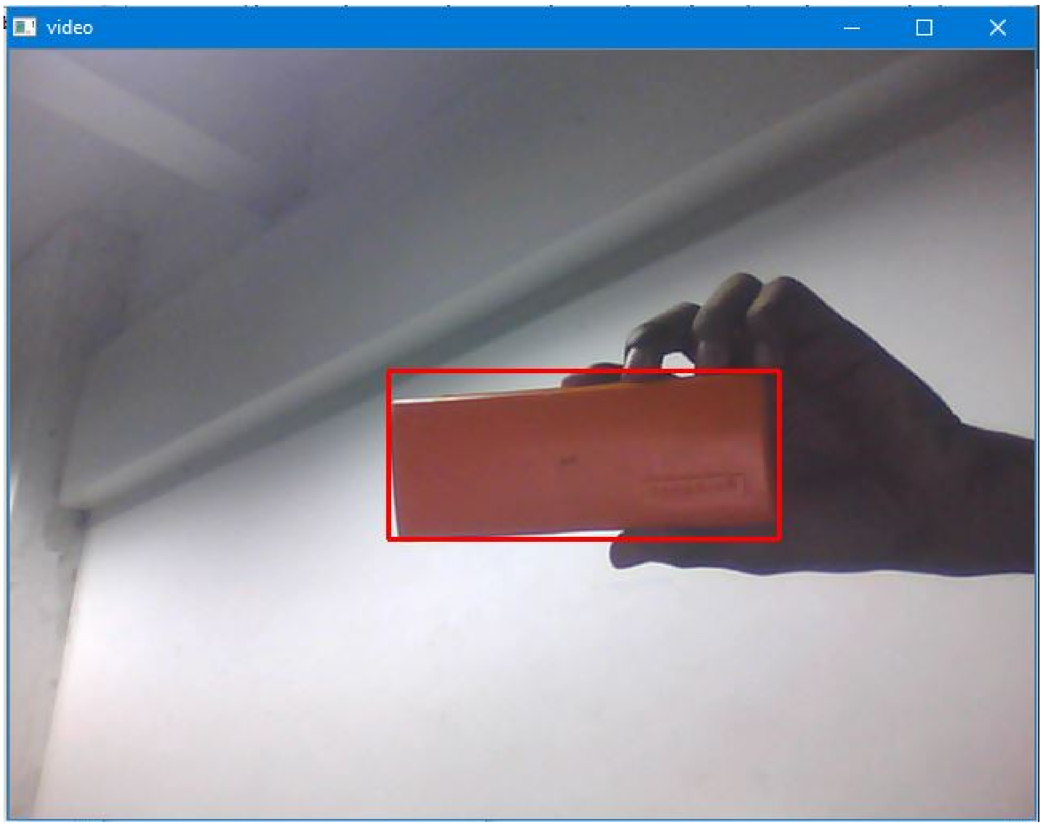
```
contours , hierarchy = cv2.findContours(mask,cv2.RETR_TREE,
                                        cv2.CHAIN_APPROX_SIMPLE)

## Let area of largest contour is zero
max_contour_area=0

##############################################
## Colored object tracking

## If the specified colored object is in the frame then there
#will be atleast one contour

## If length of contours is atleast 1  only then we need to find
#the index of largest contour


if(len(contours) >= 1):
    ## Finding index of largest contour among all the contours
    #for colored object tracking
    for i in range(0,len(contours)):
        if(cv2.contourArea(contours[i]) > max_contour_area):
            max_contour_area = cv2.contourArea(contours[i])
            max_contour_area_index = i

    ## This statement gives co-ordinates of North-West corner
    #in x and y
    ## And Width and Height in w and h of bounding rectangle
    #of Colored object
    x,y,w,h=cv2.boundingRect(contours[max_contour_area_index])

    ##create rectangle around the object which you want to track
    ##for avoiding the small contours formed
    ##when the object is not in the frame
    if w*h > 100:
        cv2.rectangle(frame, (x, y), (x+w, y+h), (0,0,255), 2)
        ##cv2.rectangle parameters are as follows:
        ##(image,co-ordinates,Color of the rectangle,thickness)

cv2.imshow('video',frame)

## Break using Escape (esc) key
## and 60 ms for frame waits(useful for video processing)
if cv2.waitKey(60) == 27:  ## 27 - ASCII for escape key
    break
```

```
        ## If frame is not successfully read or there is no frame to read
        #(in case of recorded video) this acts as stop video
        else:
            break

## Releasing camera
cap.release()

## Destroy all open windows
cv2.destroyAllWindows()
```

7 Exercise

Real time colored object tracking using webcam is shown below.





8 References

1. https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_gui/py_drawing_functions/py_drawing_functions.html#drawing-functions

2. https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_imgproc/py_colorspaces/py_colorspaces.html#converting-colorspaces

3. https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_imgproc/py_filtering/py_filtering.html#filtering

4. https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_imgproc/py_contours/py_contours_begin/py_contours_begin.html#contours-getting-started

5. https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_imgproc/py_contours/py_contour_features/py_contour_features.html#contour-features