OPENCV GİRİŞ

(INTRODUCTION TO OPENCV)

KONU BAŞLIKLARI

1. **OpenCV Nedir?**
2. **OpenCV Bileşenleri Nelerdir?**
3. **OpenCV – PYTHON Uygulamaları**

1. OpenCV Nedir?

- 1999 yılında C programlama dili ile geliştirildi.
- Açık kaynak kodlu bir kütüphanedir.
- 500'den fazla algoritma ve bu algoritmalar ile birlikte çalışan binlerce fonksiyona sahiptir.
- BSD (Berkeley Software Distribution) lisansı adı altında dağıtılmaktadır.

OpenCV

- OpenCV (Open Source Computer Vision) açık kaynak kodlu görüntü işleme kütüphanesidir.
- C++, C, Python, Java, Matlab, EmguCV kütüphanesi aracılığıyla kolaylıkla OpenCV uygulamaları geliştirilebilir.
- OpenCV kütüphanesi içerisinde görüntü işlemeye (image processing) ve makine öğrenmesine (machine learning) yönelik 2500'den fazla algoritma bulunmaktadır.
- Kolaylıkla matris işlemleri yapabileceğimiz Numpy kütüphanesi, OpenCV kütüphanesi yanında sıklıkla tercih edilir.
- Numpy kütüphanesi bilimsel hesaplama işlemleri kolaylaştırmak için yazılmış olan bir python kütüphanesidir.

2. OpenCV Bileşenleri Nelerdir?

- **CORE:** Temel fonksiyonları barındırır.
- **HighGUI:** Resim görüntüleme ve GUI için gerekli metotları barındırır.
- **IMGPROC:** Filtreleme operatörleri, nesne belirleme gibi fonksiyonları barındırır.
- **IMGCODECS:** Resim/video okuma-yazma işlemlerini barındırır.
- **VIDEOIO:** Kamera ve video cihazlarına erişim için gerekli fonksiyonları barındırır.

3. OpenCV – PYTHON Uygulamaları

3.1. Read/Display an Image (Resim Okuma)

- Load an image (using **cv::imread**)
- Display an image in an OpenCV window (using **cv::imshow**)

cv.destroyAllWindows() simply destroys all the windows we created.

cv.waitKey() is a keyboard binding function. Its argument is the time in milliseconds. The function waits for specified milliseconds for any keyboard event.

```
import numpy as np
import cv2

# Load an color image in grayscale
img = cv2.imread('resim1.jpg',0)
```

```
cv2.imshow('image',img)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

3.1. Read/Display an Image (Resim Okuma)

3.2. Write an Image (Resim Kaydetme)

```python
import numpy as np
import cv2

img = cv2.imread('resim1.jpg',0)
cv2.imshow('image',img)

k = cv2.waitKey(0)
if k == 27:         # wait for ESC key to exit
    cv2.destroyAllWindows()
elif k == ord('s'): # wait for 's' key to save and exit
    cv2.imwrite('resim2.png',img)
    cv2.destroyAllWindows()
```

3.3. Capture Video from Camera (Kameradan Görüntü Yakalama)

```python
import numpy as np
import cv2

cap = cv2.VideoCapture(0)

while(True):
    # Capture frame-by-frame
    ret, frame = cap.read()

    # Our operations on the frame come here
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)

    # Display the resulting frame
    cv2.imshow('frame',gray)
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break

# When everything done, release the capture
cap.release()
cv2.destroyAllWindows()
```

3.4. Playing Video from File
(Var olan Videoyu Oynatma)

```
import numpy as np
import cv2

cap = cv2.VideoCapture('Sergen.mp4')

while(cap.isOpened()):
    ret, frame = cap.read()

    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)

    cv2.imshow('frame',gray)
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break

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

3.5. Saving a Video (Video Kaydetme)

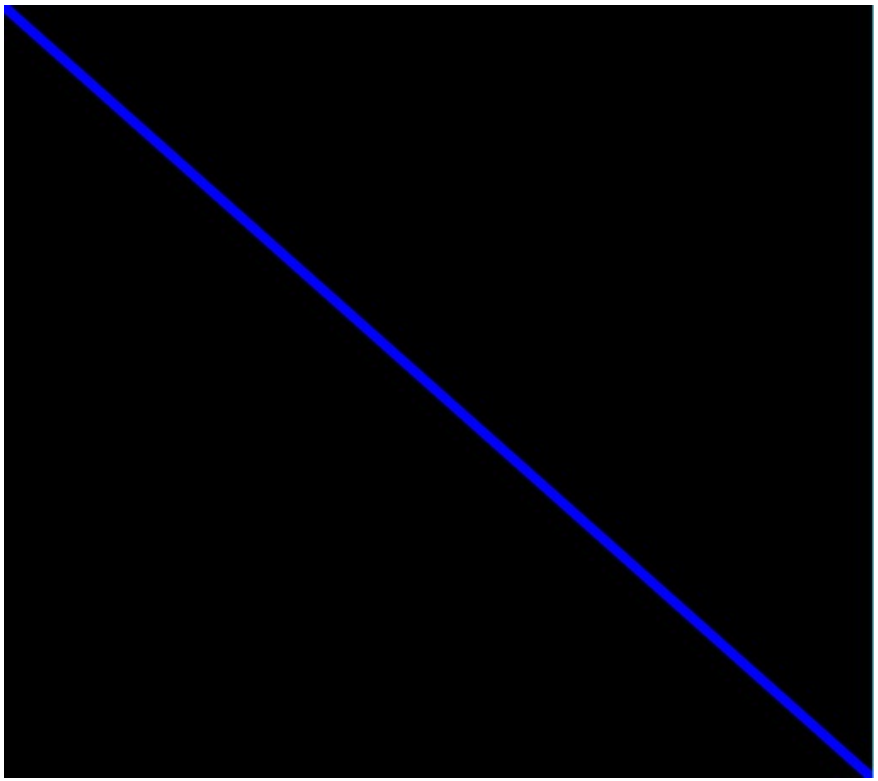
```python
import numpy as np
import cv2

cap = cv2.VideoCapture(0)

# Define the codec and create VideoWriter object
fourcc = cv2.VideoWriter_fourcc(*'XVID')
out = cv2.VideoWriter('output.avi',fourcc, 20.0, (640,480))

while(cap.isOpened()):
    ret, frame = cap.read()
    if ret==True:
        frame = cv2.flip(frame,0)

        # write the flipped frame
        out.write(frame)

        cv2.imshow('frame',frame)
        if cv2.waitKey(1) & 0xFF == ord('q'):
            break
    else:
        break

# Release everything if job is finished
cap.release()
out.release()
cv2.destroyAllWindows()
```

3.6. Drawing Functions - Line
(Bir Görüntü Üzerine Çizgi Çizmek)

```
import numpy as np
import cv2

# Create a black image
img = np.zeros((512,512,3), np.uint8)

cv2.line(img,(0,0),(511,511),(255,0,0),5)

cv2.imshow('image',img)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

https://docs.opencv.org/2.4/modules/core/doc/drawing_functions.html#cv2.line

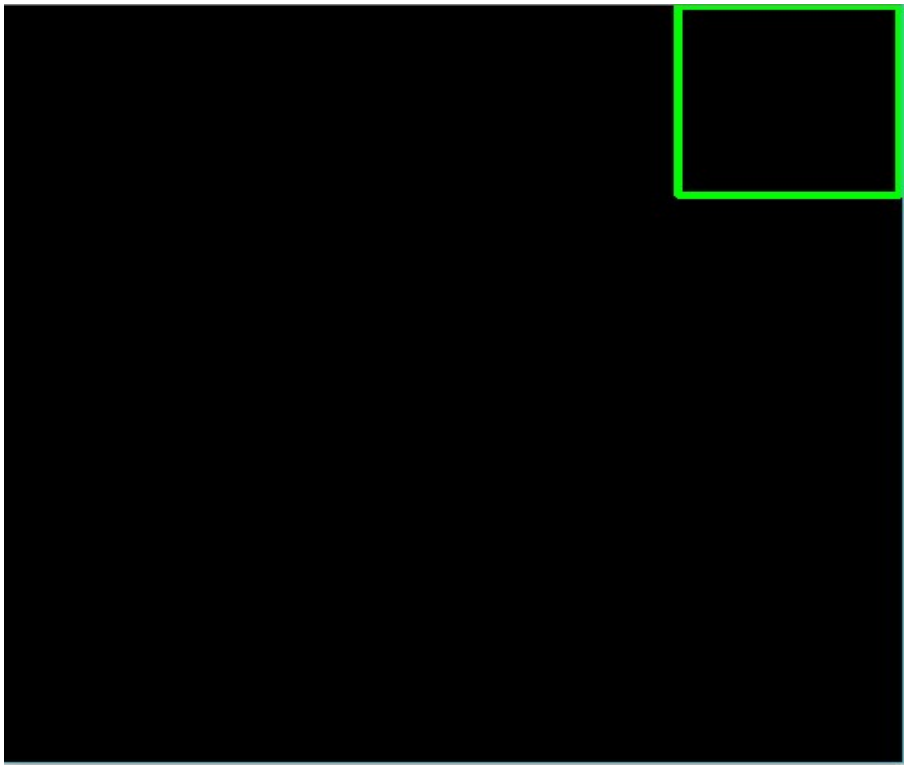
3.6. Drawing Functions - Line
(Bir Görüntü Üzerine Çizgi Çizmek)



3.7. Drawing Functions - Rectangle (Bir Görüntü Üzerine Kare Çizmek)

```
import numpy as np
import cv2

# Create a black image
img = np.zeros((512,512,3), np.uint8)

cv2.rectangle(img,(384,0),(510,128),(0,255,0),3)

cv2.imshow('image',img)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

3.7. Drawing Functions - Rectangle
(Bir Görüntü Üzerine Kare Çizmek)



3.8. Drawing Functions - Circle
(Bir Görüntü Üzerine Daire Çizmek)

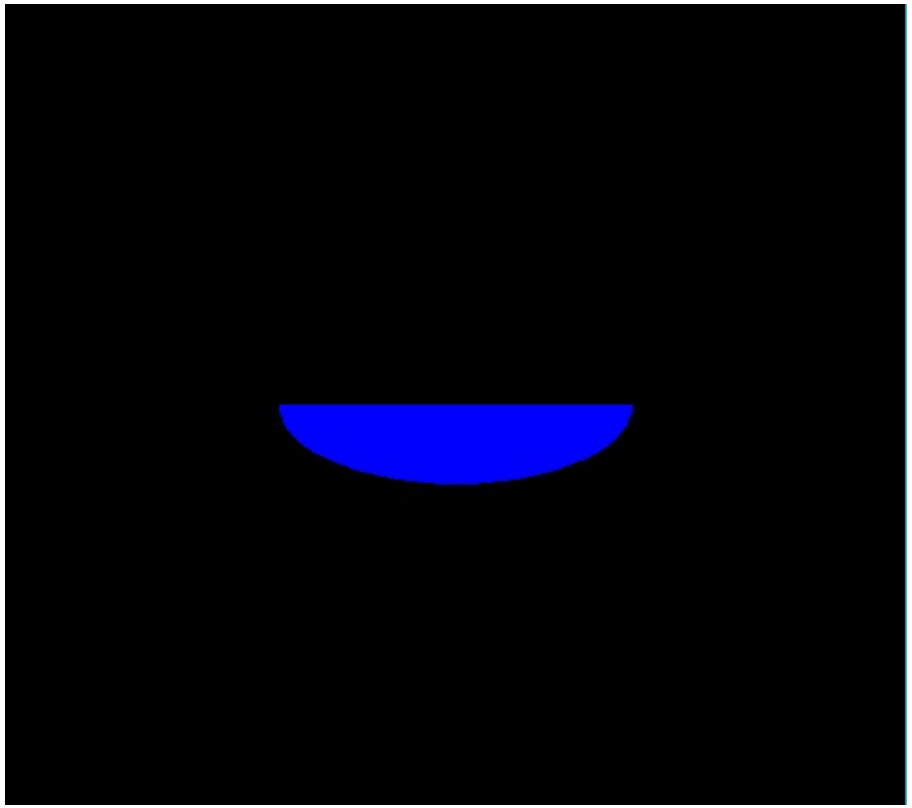
```
import numpy as np
import cv2

# Create a black image
img = np.zeros((512,512,3), np.uint8)

cv2.rectangle(img,(384,0),(510,128),(0,255,0),3)

cv2.imshow('image',img)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

3.8. Drawing Functions - Circle
(Bir Görüntü Üzerine Daire Çizmek)

3.9. Drawing Functions - Ellipse
(Bir Görüntü Üzerine Elips Çizmek)

```
import numpy as np
import cv2

# Create a black image
img = np.zeros((512,512,3), np.uint8)

cv2.ellipse(img,(256,256),(100,50),0,0,180,255, -1)

cv2.imshow('image',img)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

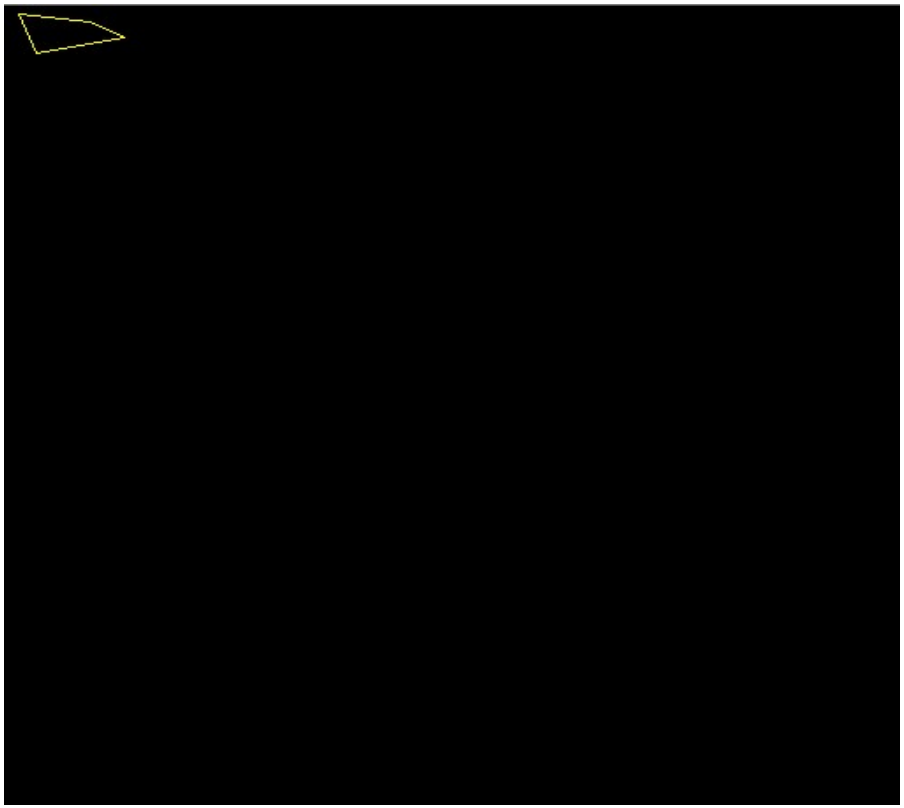
3.9. Drawing Functions - Ellipse
(Bir Görüntü Üzerine Elips Çizmek)



3.10. Drawing Functions - Polygon
(Bir Görüntü Üzerine Polygon Çizmek)

```
import numpy as np
import cv2

# Create a black image
img = np.zeros((512,512,3), np.uint8)

pts = np.array([[10,5],[20,30],[70,20],[50,10]], np.int32)
pts = pts.reshape((-1,1,2))
cv2.polylines(img,[pts],True,(0,255,255))

cv2.imshow('image',img)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

3.10. Drawing Functions - Polygon
(Bir Görüntü Üzerine Polygon Çizmek)

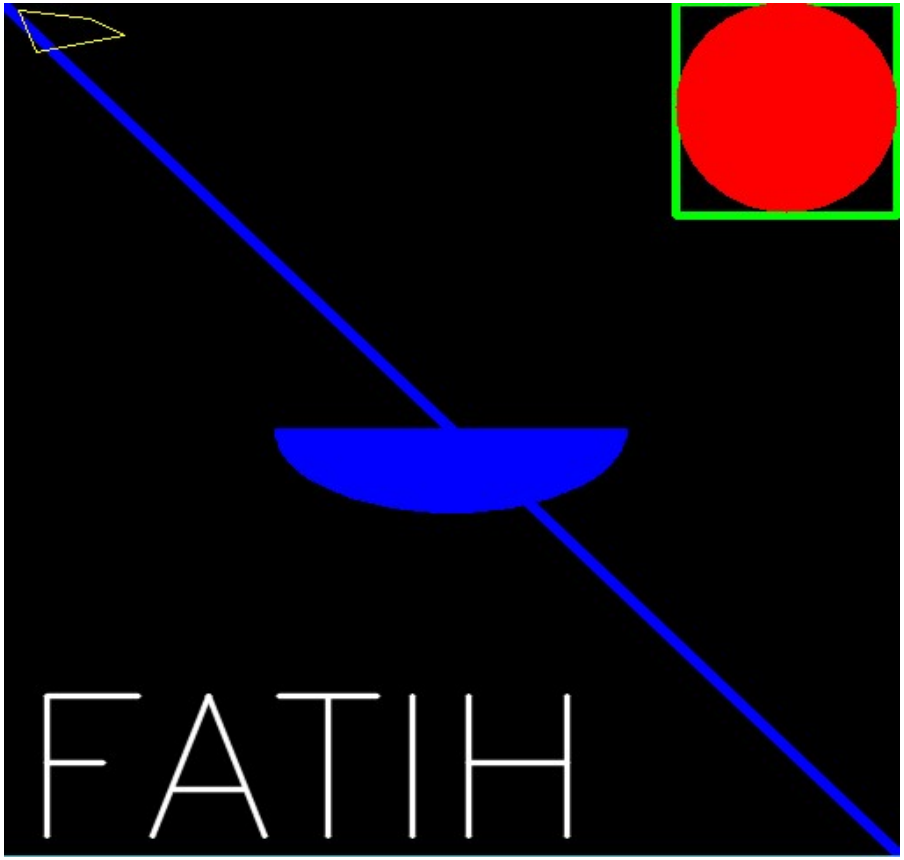


3.11. Drawing Functions – Adding Text
(Bir Görüntü Üzerine Yazı Ekleme)

```
import numpy as np
import cv2

# Create a black image
img = np.zeros((512,512,3), np.uint8)

font = cv2.FONT_HERSHEY_SIMPLEX
cv2.putText(img,'FATIH B',(10,500), font, 4,(255,255,255),2,cv2.LINE_AA)

cv2.imshow('image',img)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

3.11. Drawing Functions – Adding Text (Bir Görüntü Üzerine Yazı Ekleme)

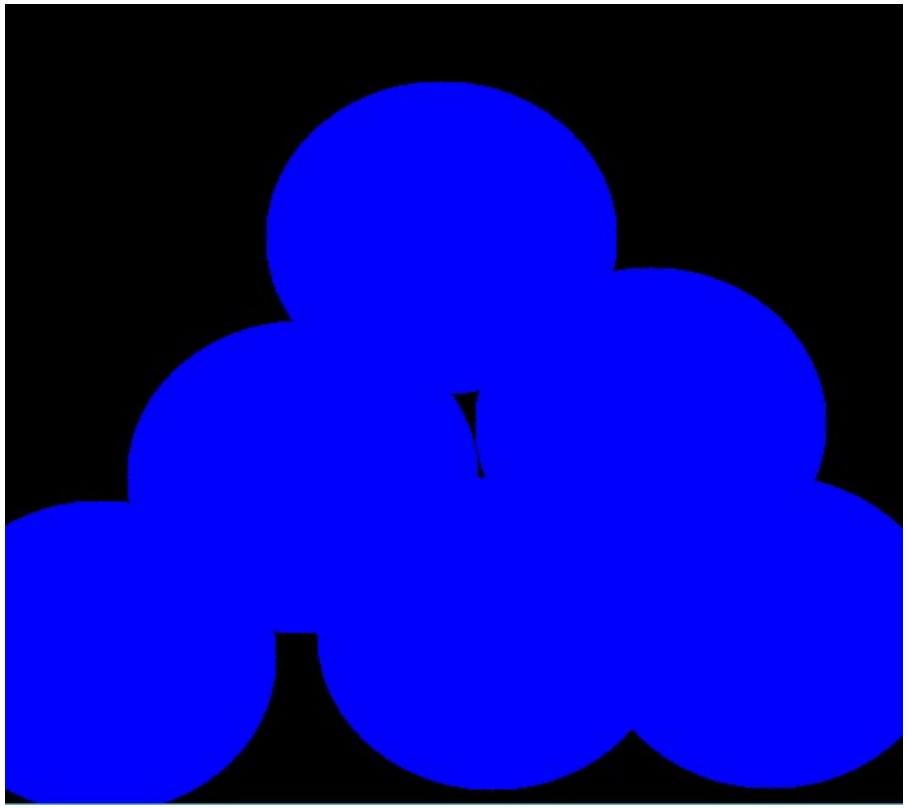
3.12. Drawing Functions – All of them



3.13. Mouse as a Paint-Brush – Simple Demo

```python
import cv2
import numpy as np

# mouse callback function
def draw_circle(event,x,y,flags,param):
    if event == cv2.EVENT_LBUTTONDBLCLK:
        cv2.circle(img,(x,y),100,(255,0,0),-1)

# Create a black image, a window and bind the function to window
img = np.zeros((512,512,3), np.uint8)

cv2.namedWindow('image')
cv2.setMouseCallback('image',draw_circle)

while(1):
    cv2.imshow('image',img)
    if cv2.waitKey(20) & 0xFF == 27:
        break
cv2.destroyAllWindows()
```

3.13. Mouse as a Paint-Brush – Simple Demo

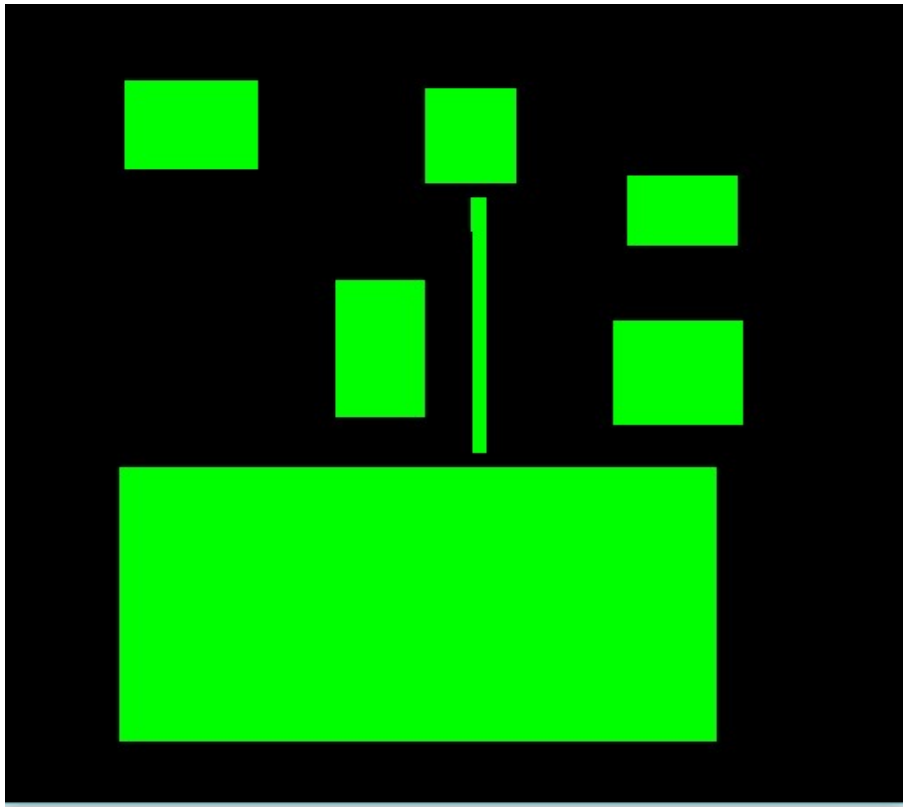


3.14. Mouse as a Paint-Brush – Advanced Demo

```python
import cv2
import numpy as np

drawing = False # true if mouse is pressed
mode = True # if True, draw rectangle. Press 'm' to toggle to curve
ix,iy = -1,-1

# mouse callback function
def draw_circle(event,x,y,flags,param):
    global ix,iy,drawing,mode

    if event == cv2.EVENT_LBUTTONDOWN:
        drawing = True
        ix,iy = x,y

    elif event == cv2.EVENT_MOUSEMOVE:
        if drawing == True:
            if mode == True:
                cv2.rectangle(img,(ix,iy),(x,y),(0,255,0),-1)
            else:
                cv2.circle(img,(x,y),5,(0,0,255),-1)

    elif event == cv2.EVENT_LBUTTONUP:
        drawing = False
        if mode == True:
            cv2.rectangle(img,(ix,iy),(x,y),(0,255,0),-1)
        else:
            cv2.circle(img,(x,y),5,(0,0,255),-1)
```

```python
img = np.zeros((512,512,3), np.uint8)
cv2.namedWindow('image')
cv2.setMouseCallback('image',draw_circle)

while(1):
    cv2.imshow('image',img)
    k = cv2.waitKey(1) & 0xFF
    if k == ord('m'):
        mode = not mode
    elif k == 27:
        break

cv2.destroyAllWindows()
```

3.14. Mouse as a Paint-Brush – Advanced Demo



3.15. Trackbar as the Color Palette

```python
import cv2
import numpy as np

def nothing(x):
    pass

# Create a black image, a window
img = np.zeros((300,512,3), np.uint8)
cv2.namedWindow('image')

# create trackbars for color change
cv2.createTrackbar('R','image',0,255,nothing)
cv2.createTrackbar('G','image',0,255,nothing)
cv2.createTrackbar('B','image',0,255,nothing)

# create switch for ON/OFF functionality
switch = '0 : OFF \n1 : ON'
cv2.createTrackbar(switch, 'image',0,1,nothing)
```

```python
while(1):
    cv2.imshow('image',img)
    k = cv2.waitKey(1) & 0xFF
    if k == 27:
        break

    # get current positions of four trackbars
    r = cv2.getTrackbarPos('R','image')
    g = cv2.getTrackbarPos('G','image')
    b = cv2.getTrackbarPos('B','image')
    s = cv2.getTrackbarPos(switch,'image')

    if s == 0:
        img[:] = 0
    else:
        img[:] = [b,g,r]

cv2.destroyAllWindows()
```

cv.createTrackbar()

trackbarname	Name of the created trackbar.
winname	Name of the window that will be used as a parent of the created trackbar.
value	Optional pointer to an integer variable whose value reflects the position of the slider. Upon creation, the slider position is defined by this variable.
count	Maximal position of the slider. The minimal position is always 0.
onChange	Pointer to the function to be called every time the slider changes position. This function should be prototyped as void Foo(int,void*); , where the first parameter is the trackbar position and the second parameter is the user data (see the next parameter). If the callback is the NULL pointer, no callbacks are called, but only value is updated.