## **เมาส์เป็น Paint-Brush**

เป้าหมาย

* เรียนรู้เพื่อจัดการเหตุการณ์เมาส์ใน OpenCV
* คุณจะได้เรียนรู้ฟังก์ชันเหล่านี้: cv2.setMouseCallback ()

สาธิตง่ายๆ

## ที่นี่เราจะสร้างแอพพลิเคชั่นง่ายๆที่จะใช้เป็นวงกลมในรูปภาพเมื่อใดก็ตามที่เราคลิกสองครั้ง ขั้นแรกเราจะสร้างฟังก์ชันเรียกกลับเม้าส์ซึ่งจะทำงานเมื่อมีเหตุการณ์เมาส์เกิดขึ้น เหตุการณ์เมาส์อาจเป็นอะไรก็ได้ที่เกี่ยวข้องกับเมาส์เช่น left-button down, left-button up, left-button double-click ฯลฯ มันทำให้เรามีพิกัด (x, y) สำหรับทุกๆเมาส์ ด้วยเหตุการณ์และที่ตั้งนี้เราสามารถทำสิ่งที่เราต้องการได้ เมื่อต้องการแสดงรายการเหตุการณ์ทั้งหมดที่พร้อมใช้งานให้เรียกใช้โค้ดต่อไปนี้ใน Python terminal:

**>>> import** **cv2**

**>>>** events = [i **for** i **in** dir(cv2) **if** 'EVENT' **in** i]

**>>> print** events

['EVENT\_FLAG\_ALTKEY', 'EVENT\_FLAG\_CTRLKEY', 'EVENT\_FLAG\_LBUTTON', 'EVENT\_FLAG\_MBUTTON', 'EVENT\_FLAG\_RBUTTON', 'EVENT\_FLAG\_SHIFTKEY', 'EVENT\_LBUTTONDBLCLK', 'EVENT\_LBUTTONDOWN', 'EVENT\_LBUTTONUP', 'EVENT\_MBUTTONDBLCLK', 'EVENT\_MBUTTONDOWN', 'EVENT\_MBUTTONUP', 'EVENT\_MOUSEHWHEEL', 'EVENT\_MOUSEMOVE', 'EVENT\_MOUSEWHEEL', 'EVENT\_RBUTTONDBLCLK', 'EVENT\_RBUTTONDOWN', 'EVENT\_RBUTTONUP']

การสร้างฟังก์ชันเรียกกลับของเมาส์มีรูปแบบเฉพาะซึ่งเหมือนกันทุกที่ มันแตกต่างเฉพาะในสิ่งที่ฟังก์ชั่นไม่ ฟังก์ชันการเรียกกลับเม้าส์ของเราทำสิ่งหนึ่งอย่างใดโดยจะดึงวงกลมที่เราคลิกสองครั้ง ดู CODE ด้านล่างนี้ CODE เป็นตัวอธิบายจากความเห็น:

**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()

การสาธิตขั้นสูงเพิ่มเติม

ตอนนี้เราไปสำหรับโปรแกรมที่ดีมาก ในนี้เราวาดสี่เหลี่ยมผืนผ้าหรือแวดวง (ขึ้นอยู่กับโหมดที่เราเลือก) โดยการลากเมาส์เหมือนที่เราทำในแอ็พพลิเคชัน Paint ฟังก์ชันการเรียกกลับเม้าส์ของเรามีสองส่วนให้วาดสี่เหลี่ยมผืนผ้าและอื่น ๆ เพื่อวาดวงกลม ตัวอย่างเฉพาะนี้จะเป็นประโยชน์ในการสร้างและทำความเข้าใจแอ็พพลิเคชันเชิงโต้ตอบบางอย่างเช่นการติดตามวัตถุการแบ่งส่วนภาพเป็นต้น

**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)

ต่อไปเราจะต้องผูกฟังก์ชันการเรียกกลับของเมาส์นี้กับหน้าต่าง OpenCV ในลูปหลักเราควรตั้งค่าแป้นพิมพ์สำหรับคีย์ 'm' เพื่อสลับระหว่างสี่เหลี่ยมผืนผ้าและวงกลม

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()