ข้อมูลเข้า และออก ใน OpenCV

ในบทนี้เราจะมาเรียนรู้การนำเข้าข้อมูล ซึ่งมีสามทางคือ อ่านจากภาพ อ่านจากภาพเคลื่อนไหว อ่านจากกล้อง หลังจากอ่านแล้วก็บันทึกเป็นไฟล์ใหม่ และสุดท้ายแสดงผลใน GUI และเพื่อให้ง่ายต่อทำความเข้าใจ รูปแบบนำเสนอของบทนี้จะใช้เป็นในลักษณะ อ่าน – บันทึก – แสดงผลจากบันทึก ของทั้งสามชนิดข้อมูลเข้า

กายภาพของภาพโทนเทา

ภาพประกอบด้วยข้อมูลตัวเลขหรือจุดสี (pixel) ที่ขึ้นอยู่กับชนิดของภาพ เช่น ภาพโทนเทา มีข้อมูลตัวเลขตั้งแต่ 0-255 (uint8) ข้อมูลตัวเลขทั้งหลายนี้จะเรียงกันขึ้นอยู่กับ ขนาดของภาพ (วัดด้วยใช้คำสั่ง shape ของ NumPy) เช่นภาพขนาด 3x3 เท่ากับมีจำนวนสี 9 จุดสี จะเขียนได้ว่า:

Code 1.

```
import cv2
import numpy as np
img = np.zeros((3,3), dtype=np.uint8)
print(img.shape)
print(img.dtype)
print(img)
cv2.imshow('Black', img)
cv2.waitKey()
cv2.destroyAllWindows()
'''
 (3, 3)
unit8
[[0 0 0]
 [0 0 0]
 [0 0 0]]
'''
```

จุดสีจะเรียงตามแนวแกน x, y โดยเริ่มต้นที่บนสุด-ซ้ายสุด หมายความว่าถ้าต้องการใช้ จุดบนสุด-ซ้ายสุด มีสีขาว (255) จะต้องกำหนดจุด (0,0) ดูจากตัวอย่างต่อไปนี้

Code 2.

```
import cv2
import numpy as np
img = np.zeros((3,3), dtype=np.uint8)
img[0,0]=255
print(img)
cv2.imshow('(0,0) is White', img)
cv2.waitKey()
cv2.destroyAllWindows()
'''
[[255   0   0]
 [  0   0   0]
 [  0   0   0]]
'''
```

รูป 1 ภาพขนาด 3x3 ทุกจุดสีเป็น 0 (ซ้าย) และจุดสีแรกเป็น 255*

**ภาพที่แสดงนี้ ทำงานบนโปรแกรม Anaconda หากใช้ CLI บน Wondows จะได้ภาพขนาดเล็กจนมองไม่เห็น*

กายภาพของภาพสี

กรณีเป็นภาพสี ใช้สี 3 แยกกัน คือ Blue-Green-Red(BGR) เมื่อเทียบกับสีโทนเทา มีเพียงสีเดียว ซึ่งคือค่าใดค่าหนึ่งใน 0-255 แต่ของภาพสีใช้สามค่า ที่แสดงถึงสีโทนน้ำเงิน โทนเขียว และโทนแดง จากการดูขนาดมิติ จะมี 3 มิติ แต่ละมิติแทน B, G, และ R นั่นเอง

Code 3.

```
import cv2
img = np.zeros((3,3), dtype=np.uint8)
img = cv2.cvtColor(img, cv2.COLOR_GRAY2BGR)
print(img.shape)
print(img)

'''
(3, 3, 3)
[[[0 0 0]
  [0 0 0]
  [0 0 0]]

 [[0 0 0]
  [0 0 0]
  [0 0 0]]

 [[0 0 0]
  [0 0 0]
  [0 0 0]]]
'''
```

เช่นเดียวกับภาพโทนเท่า ภาพสีกำหนดจุดสีแต่ละจุดได้ เช่น ถ้าต้องการใช้จุด (0,0,0) มีสีน้ำเงิน (Blue) ทำได้โดยกำหนดให้มีค่าเป็น 255 ซึ่งขนาดสูงสุด (ชนิดข้อมูลเป็น unit8 เมื่อรวมทั้ง 3 สี (B-G-R) จะมีค่า 24 บิท)

Code 4.

```
import cv2
import numpy as np
img = np.zeros((3,3), dtype=np.uint8)
img = cv2.cvtColor(img, cv2.COLOR_GRAY2BGR)

img[0,0,0] = 255
print(img)

cv2.imshow('Blue', img)
cv2.waitKey()
cv2.destroyAllWindows()
'''
[[[255   0   0]
  [  0   0   0]
  [  0   0   0]]

 [[  0   0   0]
  [  0   0   0]
  [  0   0   0]]

 [[  0   0   0]
  [  0   0   0]
```

```
  [  0   0   0]]]
'''
```

จากตัวอย่างนี้จะเห็นว่า แถวแรก (255, 0, 0) เป็นส่วนผสมของสี B-G-R การให้ค่าอื่นๆ 0 หมดยกเว้นตัวแรกเป็น 255 เท่าให้สีน้ำเงินทำงานอย่างเดียว และถ้ากำหนดส่วนผสมสี ณ จุดเดิมนี้ เป็นสีผสมระหว่าง B กับ G เช่น กำหนด (255, 255, 0) ด้วยคำสั่ง

Code 5.

```
import cv2
import numpy as np

img = np.zeros((3,3), dtype=np.uint8)
img = cv2.cvtColor(img, cv2.COLOR_GRAY2BGR)

img[0,0,0] = 255
img[0,0,1] = 255

print(img)
cv2.imshow('Blue', img)
cv2.waitKey()
cv2.destroyAllWindows()
```

ข้อมูลในเมตริกซ์ จะเป็นดังนี้

```
[[[255 255   0]
  [  0   0   0]
  [  0   0   0]]

 [[  0   0   0]
  [  0   0   0]
  [  0   0   0]]

 [[  0   0   0]
  [  0   0   0]
  [  0   0   0]]]
```

จากข้อมูลเมตริกซ์ ทำให้ทราบแน่ชัดว่า แถวแรกของเมตริก คือ หนึ่งจุดสีแรกที่ตำแหน่ง (0,0) ของแกน (x,y) ของระนาบสองมิติ ด้วยข้อมูลนี้ ถ้าให้ คอลัมน์แรกของเมตริกซ์ เป็น 255 เท่ากับการทำให้ภาพ แถวแรกเป็นสีน้ำเงิน ด้วยคำสั่ง

```
img[0,:,0] = 255
```

ด้วยคำสั่งข้างบนนี้เป็นการเลือก เฉพาะมิติ 0 ของมิติแรก คือเลือกแถวแรก ต่อมา : แทนเลือกทุกแถวของมิติ 2 และ 0 แทนเลือก คอลัมน์แรก ของมิติ 3 ดังนั้นผลของเมตริกคือ และผลของภาพดูรูป 2 ประกอบ

```
[[[255   0   0]
  [255   0   0]
  [255   0   0]]      0,:, 0 => แถวแรก  ทุกแถว คอลัมน์แรก

 [[  0   0   0]
  [  0   0   0]
  [  0   0   0]]      แถวที่สอง

 [[  0   0   0]
```

```
[  0   0   0]
[  0   0   0]]]      แถวที่สาม
```

รูป 2 ภาพสีขนาด 3x3 จุดสีแรก(0,0,0) เป็น 255 (Blue) (ซ้ายสุด)
จุด (0, 0, 0) และ จุด (0, 0, 1) เป็น 255 (กลาง)
และ แถวแรกของภาพเป็น 255 (ขวาสุด)

เพื่อให้เข้าใจโครงสร้างกายภาพที่ดีขึ้น อธิบายได้ด้วยโครงสร้างอาร์เรย์ 3x3x3 มิติ ของภาพ 3 แถว 3 คอลัมน์ และแต่ละคอลัมน์แทนด้วยส่วนผสมของสี 3 ส่วน (24 บิท) ดังเขียนเป็นโครงสร้างอาร์เรย์ได้คือ

```
[
    Row0[
       Col0[B G R]
       Col1[B G R]
       Col2[B G R]
    ]

    Row1[
       Col0[B G R]
       Col1[B G R]
       Col2[B G R]
    ]

    Row2[
       Col0[B G R]
       Col1[B G R]
       Col2[B G R]
    ]
]
```

รูป 3 โครงสร้างอาร์เรย์ของภาพ ขนาด 3x3x3

รูปภาพที่เราเห็นมีภาพใน 2 มิติ (เป็นระนาบ) และข้อมูลของภาพกรณีเป็นสี แทนด้วย อาร์เรย์ขนาด 3 รวม 24 บิท (8x3) ซึ่งแทนด้วย B-G-R เป็นสิ่งที่มนุษย์รับรูปได้ หากว่าเป็นอาร์เรย์ 3x3x4 ในมิติสุดท้าย ก็อาจแทน B-G-R-A ซึ่ง A เป็นสี Alpha ที่แทนความโปร่งแสง (transparency) ซึ่งก็เป็นภาพที่มนุษย์รับรู้ได้เช่นกัน แต่ถ้า เป็นอาร์เรย์ 3x3x3x3 อันนี้ก็คือภาพ 3 มิติ (เป็นลูกบาศก์) ที่มีความทิศทางความลึก และข้อมูลสีคือ มิติที่ 4 ที่แทนด้วยอาร์เรย์ขนาด 3

ในตัวอย่าง เป็นภาพขนาดเล็กมาก บางทีการแสดงผลอาจดูไม่เห็น จึงควรให้ตัวอย่างมีขนาดใหญ่ขึ้น เช่น ขนาด 200x200x3 ให้ทดลองทำภาพกับแนวตั้ง เช่น [0:10,:, 0] = 255 (แนวนอนหนา 10 เป็นสีน้ำเงิน) [0:10,:,1] = 255 เป็นสีเขียว

ข้อมูลภาพ

การอ่านภาพนิ่ง ด้วย OpenCV ด้วย imread( ) ซึ่งได้พบมาบ้างแล้ว ในเรื่องการใช้ NumPy ซึ่งต่อไปจะเขียนไฟล์นั้นอย่างไร การเขียนเป็นไฟล์ หรือเขียนข้อมูลไบท์ เพื่อการส่งข้อมูลในเครือข่าย

Code 6.

```
import cv2
import numpy as np

img = cv2.imread('d:/chess.jpg')

cv2.imwrite('new_img.jpg', img)

new_img = cv2.imread('new_img.jpg')
cv2.imshow('new_img', new_img)

cv2.waitKey()
cv2.destroyAllWindows()
```

จากตัวอย่างนี้ จะเห็นว่า เราสามารถเขียน ข้อมูลภาพได้ด้วยคำสั่ง cv2.imwrite('new_img.jpg') ได้ และสามารถเรียกอ่านภาพไฟล์นี้ ได้มีผลเหมือนเดิม

กรณีต้องการส่งไฟล์ทางเครือข่ายให้อยู่ในรูปของ ไบท์ ซึ่งมีขนาด 8 bit สำหรับภาพโทนเทา เราใช้ คำสั่งของ Python ที่มีอยู่แล้ว คือ **bytearray(img)** ก็จะได้ลำดับอาร์เรย์ของไบท์ และเมื่อส่งไฟล์ในรูปอาร์เรย์ไปแล้ว ผู้รับก็สามารถแปลงไฟล์กลับเป็นรูปข้อมูลภาพได้เหมือนเดิมแต่ต้องจัดการกับมิติใหม่ ให้เท่ากับมิติเดิมของภาพ ดังเห็นได้จากตัวอย่างต่อไปนี้

Code 7.

```
import cv2
import numpy as np

img = cv2.imread('d:/icon.png',0)
print(img.shape)
img_byte = bytearray(img)
print(img_byte)

img_gray = np.array(img_byte).reshape(10,10)

cv2.imshow('new', img_gray)
cv2.waitKey()
cv2.destroyAllWindows()

'''
(10, 10)
bytearray(b"\xff\xff\xff\xff\xf7\xf7\xff\xff\xff\xff\xff\xff\xb5\x14\x00\x00\x14\xb6\xff\xf
f\xff\xb4\x00\x00\x05\x05\x00\x00\xb6\xff\xff\x12\x00\x00jj\x00\x00\x15\xff\xf7\x00\x00\x00
\'\'\x00\x00\x00\xf7\xf7\x00\x00\x00~~\x00\x00\x00\xf7\xff\x13\x00\x00\x80\x80\x00\x00\x14\
xff\xff\xb5\x00\x00\x02\x02\x00\x00\xb5\xff\xff\xff\xb5\x13\x00\x00\x14\xb6\xff\xff\xff\xff
\xff\xff\xf7\xf7\xff\xff\xff\xff")
'''
```

อ่านภาพบางส่วน

การอ่านภาพบางส่วนของภาพ คือการเลือกพื้นที่บางส่วน ซึ่งระบุได้ด้วย จำนวนข้อมูลภาพในลักษณะอาร์เรย์ เช่น ภาพขนาด img[0:10] จากภาพขนาด (100,100) ซึ่งดูได้ด้วยคุณสมบัติ shape

เมื่อเลือกบางส่วนได้ เราก็สามารถกำหนดให้บางส่วนที่เลือกนี้ ไปทำอะไรก็ได้ เช่น เอาเฉพาะส่วนที่เลือกให้เท่ากับสีหรือภาพอื่น

ตัวอย่างต่อไปนี้ ภาพขนาด (333, 486, 3) เลือกเฉพาะบางส่วนคือ (42, 36) ซึ่งมีขนาดเท่ากับภาพเล็ก อีกภาพหนึ่งพอดี โดยตำแหน่งที่เลือกนี้ อยู่ ที่ 0 ถึง 42 ในแนวแกนนอน และ ตำแหน่ง 0 ถึง 36 อยู่ในแนวแกนตั้ง

Code 8.

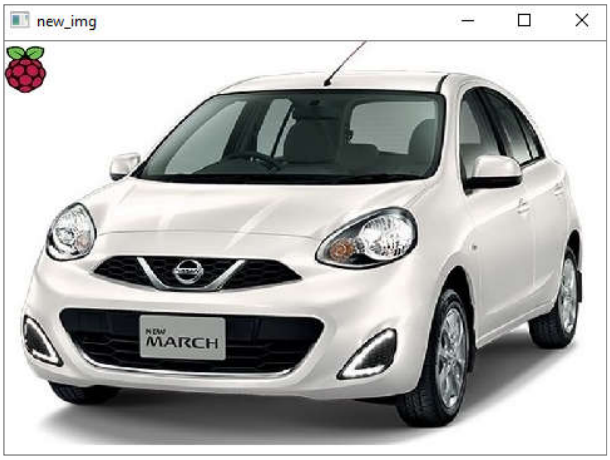
```
import cv2
import numpy as np

img = cv2.imread('d:/car.jpg')
print(img.shape) # (333, 486, 3)
logo = cv2.imread('d:/raspberry_small.jpg')
print(logo.shape)#(42, 36, 3)

img[0:42,0:36] = logo
cv2.imwrite('new_img.jpg', img)

new_img = cv2.imread('new_img.jpg')
cv2.imshow('new_img', new_img)

cv2.waitKey()
cv2.destroyAllWindows()
```



รูป 3 การทำสำเนาภาพเล็กไปวางบนภาพใหญ่

ข้อมูลภาพเคลื่อนไหว

การอ่านภาพจากวีดีโอ หรือ ภาพเคลื่อนไหว OpenCV มีสองคลาสสำคัญคือ VideoCapture และ VideoWriter ทั้งสองคลาสนี้สนับสนุนการทำงานภาพวิดีโอในหลายรูปแบบ เช่น avi, ogv, flv

เมธอด read( ) ของ VideoCapture ใช้อ่านเฟรม หรือภาพแต่ละภาพของไฟล์วีดีโอ ซึ่งอยู่ในรูปแบบ BGR การอ่านจะอ่านจริงกระทั่ง หมดไฟล์ ดังนั้นเมธอดนี้จะส่งค่าออกมาสองค่า คือ เฟรม และค่าการอ่านสำเร็จ กรณีอ่านไม่สำเร็จซึ่งอนุมานได้ว่าอ่านจบแล้ว

เมธอด write( ) ของคลาส VideoWriter จะเป็นการแทรกเฟรมไปยังไฟล์วิดีโอ คลาส VideoWriter มีคอนสตรักเตอร์ ที่มีตัวแปรเข้า คือ ไฟล์ที่จะบันทึก รูปแบบบันทึก อัตราเฟรมต่อวินาที และขนาดกว้างลและสูงของวิดีโอ

Code 9.

```
import cv2
import numpy as np
```

```
#https://videos.pexels.com/search/animals
vdoCap = cv2.VideoCapture('d:/hawk.mp4')
fps = vdoCap.get(cv2.CAP_PROP_FPS)
size = ( int(vdoCap.get(cv2.CAP_PROP_FRAME_WIDTH)),
         int(vdoCap.get(cv2.CAP_PROP_FRAME_HEIGHT)) )

vdoWrite = cv2.VideoWriter(
               'd:/myCopyHawk.avi',
               cv2.VideoWriter_fourcc('I','4','2','0'),
               fps, size)
success, frame = vdoCap.read()

while success:
    vdoWrite.write(frame)
    success, frame = vdoCap.read()

vdoWrite.release()
vdoCap.release()
cv2.destroyWindow('MyWindow')
```

จากตัวอย่างนี้ ตัวแปรเข้าหลายตัวของคอนสตรักเตอร์ VideoWriter หาได้จากการอ่านไฟล์วิดีโอ เช่น อัตราเฟรมต่อวินาที หาจากการอ่านเมธอด get(cv2.CAP_PROP_FPS) และการขนาดความกว้างและสูงก็เช่นกัน

สำหรับรูปแบบบันทึกของ VideoWriter_fourcc( ) มีอยู่ด้วยกันหลายรูปแบบ ดังคือ:

- `('I', '4', '2', '1')` เป็นรูปแบบวิดีโอบีดอัดข้อมูลไม่ได้ เป็นรูปแบบที่ใช้ทั่วไป ควรให้เป็นไฟล์นามสกุล .avi
- `('P', 'I', 'M', '1')` เป็นรูปแบบวิดีโอบีดอัดข้อมูลได้ เป็นรูปแบบ MPEG-1 ควรให้เป็นไฟล์นามสกุล .avi
- `('X', 'V', 'I', 'D')` เป็นรูปแบบวิดีโอบีดอัดข้อมูลได้ เป็นรูปแบบ MPEG-4 ควรให้เป็นไฟล์นามสกุล .mp4
- `('T', 'H', 'E', 'O')` เป็นรูปแบบวิดีโอบีดอัดข้อมูลได้ เป็นรูปแบบ MPEG-4 ควรให้เป็นไฟล์นามสกุล .ogv
- `('F', 'L', 'V', '1')` เป็นรูปแบบวิดีโอบีดอัดข้อมูลได้ เป็นรูปแบบ Flash ควรให้เป็นไฟล์นามสกุล .flv

จากที่เขียนข้อมูลได้แล้ว เราทดสอบเปิดข้อมูลดูผลการเขียนไฟล์วิดีโอได้ หรือจะใช้ Python เปิดอ่าน กรณีที่ใช้ Python เปิดอ่าน เราอาจสร้างเป็น GUI เพื่ออ่าน และมีเหตุการณ์รับการคลิก เพื่อปิดหน้าต่างแสดงวิดีโอ

ตัวอย่างต่อไปนี้ แสดงการใช้ อีเว้นท์ (event) ของการคลิกเม้าท์ซ้าน (EVENT_LBUTTONUP) เพื่อการควบคุมการอ่านภาพวิดีโอ ในเงื่อนไข while จะมีควบเงื่อนไข clicked เป็นจริงหรือไม่ หากเงื่อนไขนี้เป็นจริง จะทำให้หยุดการอ่าน

นอกจาก while มีเงื่อนไขควบกับ clicked แล้ว ยังมีเงื่อนไขการกดคีย์บอร์ดร่วมด้วย อีกตัวหนึ่งผ่านฟังก์ชัน waitKey(1) การให้เท่ากับ -1 หมายความว่า ยังไม่มีการกดคีย์บอร์ดใดๆ หากกดอะไรบนคีย์บอร์ด จะคืน่า รหัส ASCII ซึ่งไม่ใช่ -1 แน่นอน ก็จะทำให้เงื่อนไขเป็นเท็จ และออกจาก while ซึ่งหมายถึงการหยุดอ่านภาพวิดีโอ

Code 10.

```
#read vdo on Windows GUI
import cv2

clicked = False

def onMouse(event, x, y, flags, param):
    global clicked
    if event == cv2.EVENT_LBUTTONUP:
        clicked = True
```

```
cv2.namedWindow('MyWindow')
cv2.setMouseCallback('MyWindow', onMouse)

vdoCap = cv2.VideoCapture('d:/myCopyHawk.avi')
success, frame = vdoCap.read()
print(success)

while success and cv2.waitKey(1) == -1 and not clicked:
    cv2.imshow('MyWindow', frame)
    success, frame = vdoCap.read()

vdoCap.release()
cv2.destroyWindow('MyWindow')
```

สำหรับตัวแปรของ onMouse มีอีเว้นท์มากมาย ดังแสดงเป็นรายการต่อไปนี้

- cv2.EVENT_MOUSEMOVE
- cv2.EVENT_LBUTTONDOWN
- cv2.EVENT_RBUTTONDOWN
- cv2.EVENT_MBUTTONDOWN
- cv2.EVENT_LBUTTONUP
- cv2.EVENT_RBUTTONUP
- cv2.EVENT_MBUTTONUP
- cv2.EVENT_LBUTTONDBLCLK
- cv2.EVENT_RBUTTONDBLCLK
- cv2.EVENT_MBUTTONDBLCLK

อีเว้นท์เหล่านี้เราเดาความหมายได้ เช่น L หมายถึง left, R หมายถึง right และ M หมายถึง middle เมื่อรวมกับการคลิกเม้าท์หมายถึง การคลิกปุ่มเม้าท์ ซ้าย ขวา และกลาง ตามลำดับ

ตัวแปร flags ก็เป็นตัวแปรหนึ่งที่ใช้ร่วมกับ อีเว้นท์ได้ รายการตัวแปร flags ต่อไปนี้ แสดงอาการการกำลังทำ เช่น กำลังเริ่มคลิกเม้าท์ซ้าย (cv2.EVENT_FLOAG_LBUTTON)

- cv2.EVENT_FLAG_LBUTTON
- cv2.EVENT_FLAG_RBUTTON
- cv2.EVENT_FLAG_MBUTTON
- cv2.EVENT_FLAG_CTRLKEY
- cv2.EVENT_FLAG_SHIFTKEY
- cv2.EVENT_FLAG_ALTKEY

สำหรับตัวแปร param เป็นค่าปรับแต่งเสริม ที่ส่งมาจาก setMouseCallback ซึ่งส่งค่าปริยายมา เป็น 0

OpenCV ไม่มีเว้นท์ควบคุมหน้าต่าง เช่น คลิกปิดหน้าต่าง อันนี้ไม่มี ซึ่งจะเห็นได้ว่า เมื่อเราคลิกปิดแล้ว โปรแกรมก็ยังทำงานอยู่ หรือเกิดอะไรที่คาดเดาไม่ได้ เช่นโปรแกรมทำงานผิดพลาด

ข้อมูลจากล้อง

การอ่านข้อมูลจากกล้อง เช่น กล้องเว็บ (Web Cam) ที่มาพร้อมกับคอมพิวเตอร์โน้ตบุค หรือการใช้กล้องต่อผ่าน USB กล้องเหล่านี้ OpenCV จะอ่านผ่านฟังก์ชัน VideoCapture( ) ที่ต้องระบุตัวเลข เช่น ระบุเป็นเลขศูนย์ จะหมายถึง ใช้อุปกรณ์กล้องตัวแรกที่ระบบพบ ซึ่งมักเป็น กล้อมที่สร้างมากับเครื่องคอมพิวเตอร์โน้ตบุค

การอ่านภาพจากกล้องมีปัญหาอย่างหนึ่งคืออัตราเฟรมต่อวินาที ที่หาไม่ได้ แม้จะใช้ฟังก์ชัน get( ) อย่างที่เคยใช้ได้กับการอ่านภาพวิดีโอจากไฟล์ แต่กับกล้องแล้ว การอ่าน แบบ get( ) นี้ อาจไม่ได้ผล ซึ่งจะคืนค่า 0 ดังนั้น จึงต้องใส่ค่าอัตรา

เฟรมต่อวินาทีเอง ในตัวอย่างกำหนดไว้ที 30 ซึ่งเป็นค่าที่ทั่วไปของกล้อม การจะรู้ว่ากล้อมควรมีค่านี้เท่าใด ให้อ่านค่าที่กำหนดมากับเครื่อง หรือให้ทดลองอ่านจากโปรแกรมอื่น แล้วนำไฟล์นั้นมาจับอัตราเฟรมต่อวินาที

ในการอ่านหรือบันทึกผ่านกล้อง เราจะหยุดการอ่านหรือบันทึกเมื่อใด เราอาจใช้อีเว้นท์ของเม้าท์ หรือคีย์บอร์ด ได้ แต่ในตัวอย่างต่อไปนี้ใช้วิธีที่ต่างออกไป กล่าวคือ นำจำนวนเฟรม เช่น เรากำหนดให้ fps เป็น 30 เราจะบันทึก ไป 300 เฟรม ก็ให้คูณ 10 และเมื่อทราบจำนวนเฟรมทั้งหมด ก็ทำให้ทราบจำนวนวินาทีทั้งหมดด้วย เราใช้เป็นเงื่อนไขในการอ่าน หรือบันทึกภาพได้ ด้วยการนับเฟรมนี้

Code 11.

```
#capture and write
import cv2
cameraCap = cv2.VideoCapture(0)
fps = 30
#fps = vdoCap.get(cv2.CAP_PROP_FPS) #ruturn 0 then it is an ERROR

size = (int(vdoCap.get(cv2.CAP_PROP_FRAME_WIDTH)),
        int(vdoCap.get(cv2.CAP_PROP_FRAME_HEIGHT))
       )
vdoWrite = cv2.VideoWriter(
           'd:/myCameraCap1.avi',
           cv2.VideoWriter_fourcc('I','4','2','0'),
           fps, size)
success, frame = cameraCap.read()
print(success)
numframesRemaining = 6 * fps - 1
while success and numframesRemaining > 0:
    vdoWrite.write(frame)
    success, frame = cameraCap.read()
    numframesRemaining -= 1

cameraCap.release()
vdoWrite.release()
```

หลังจากที่เราได้บันทึกภาพวิดีโอ ที่ทำไว้ตัวอย่างก่อนหน้านี้แล้ว เราจะมาทดลองอ่านภาพนั้น เราทำได้ดังที่เคยอ่านภาพวิดีโอทั่วไป

Code 12.

```
#read cameraCap and show on Windows GUI
import cv2

clicked = False
def onMouse(event, x, y, flags, param):
    global clicked
    if event == cv2.EVENT_LBUTTONUP:
        clicked = True

cv2.namedWindow('MyWindow')
cv2.setMouseCallback('MyWindow', onMouse)

cameraCap = cv2.VideoCapture(0)

success, frame = cameraCap.read()
print(success)
while success and cv2.waitKey(1) == -1 and not clicked:
    cv2.imshow('MyWindow', frame)
```

```
    success, frame = cameraCap.read()

cameraCap.release()
cv2.destroyWindow('MyWindow')
```

ถึงแม้ว่า OpenCV ไม่สามารถอ่าน จำนวนเฟรมต่อวินาทีได้ เทคนิคประมาณค่าแบบหนึ่งที่เผยแพร่บนเว็บ www.learnopencv.com ได้นำเสนอการประมาณค่าจาก อ่านเฟรม ผ่านเวลาตั้งต้น และเวลาสิ้นสุด โดยให้อ่านไป 120 เฟรม แล้วใช้เวลาไปเท่าใด ด้วยการทำอย่างนี้ จะได้ อัตราเฟรม โดยใช้ จำนวนเฟรมที่อ่าน หารกับเวลาที่ใช้ไปในการอ่าน ซึ่งผลอัตราเฟรมที่ได้ประมาณ 30 pfs ซึ่งใกล้เคียงกับที่ประมาณไว้ก่อนหน้านี้

Code 13.

```
#https://www.learnopencv.com/
#how-to-find-frame-rate-or-frames-per-second-fps-in-opencv-python-cpp/
import cv2
import time

print(cv2.__version__) #3.4.1

num_frames = 120
vdo = cv2.VideoCapture(0);

start = time.time()
print("Start:{}".format(start))

for i in range(0, num_frames):
     ret, frame = vdo.read()

end = time.time()
print("End:{}".format(end))

seconds = end - start
fps = num_frames/seconds

print("FPS:{}".format(fps))

vdo.release()
```

แบบฝึกหัด

1. ให้เลือกรูปมาหนึ่งรูป และให้ตีกรอบขนาด 5 จุดสี ให้มีสีพื้นเป็นสีแดง แล้วแสดงผล
2. ให้เลือกไฟล์วีดีโอ แล้ว ให้เขียนไฟล์วีดีโอ_ใหม่ โดยที่มุมขวาสุด-ซ้ายสุด มีโลโก้ ขนาดเล็ก (ที่หาเอง) ในวีดีโอใหม่