

Date: 20221022 รหัสนักศึกษา B6321451 ชื่อ-สกุล นางสาวขวัญจิรา พันธฤต

Week1010-Haar-Cascade Train, Face Recognition

Week1020-Tesseract and Sudoku to Text

Week1030-Count and Classification

- ทำเอกสารให้สมบูรณ์
- กำหนดชื่อไฟล์ตามรูปแบบนี้ "B3601234-Week10-นายวิชัย ศรีสุรักษ์.pdf"
- MC ส่งงาน Class Check ก่อน 21:00น วันพุธที่ 12 ตค 65 ที่ <https://forms.gle/3doeZyNXqJZ1qeSL7>
- MC ส่งงาน Homework ก่อน 06:00น วันพุธที่ 19 ตค 65 ที่ <https://forms.gle/zxWaMeCkaqBDWHa57>
- PC ส่งงาน Class Check ก่อน 17:00น วันเสาร์ที่ 15 ตค 65 ที่ <https://forms.gle/dC6s6GKSj8bq5H4M9>
- PC ส่งงาน Homework ก่อน 06:00น วันเสาร์ที่ 22 ตค 65 ที่ <https://forms.gle/B4xjuKq3J83vsbrk6>

Week1010-Haar-Cascade Train, Face Recognition

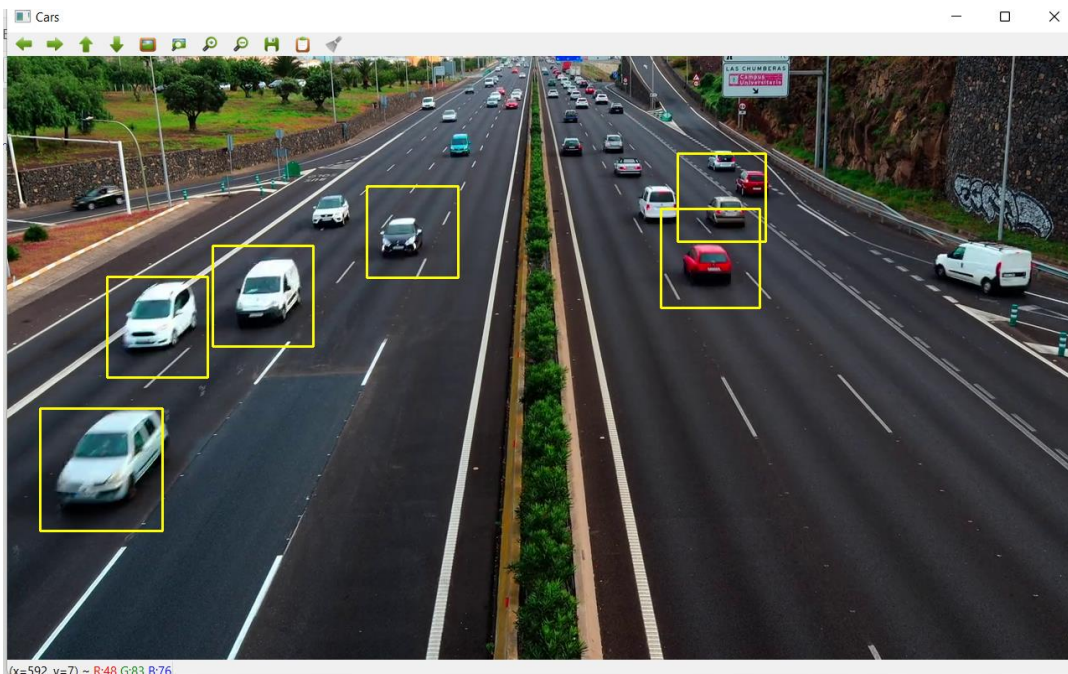
4.3 - การนำโมเดลไปใช้ในการตรวจหาวัตถุ

4.7 Test My Model = cascade_PkC17 เทียบกับ cascade_car (Download Model)

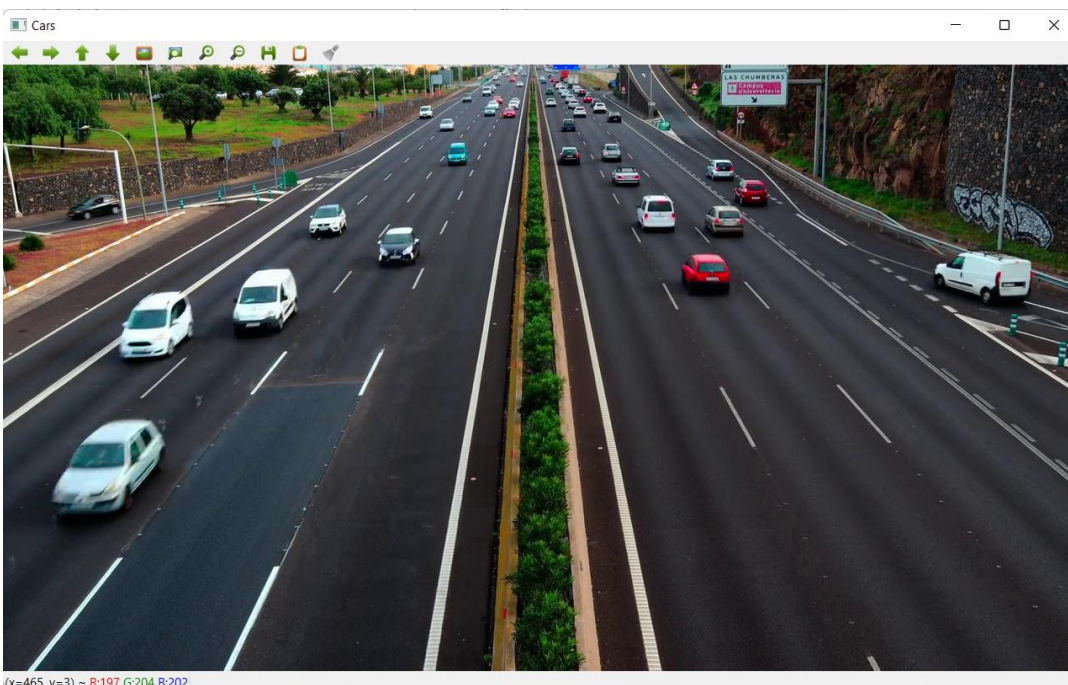
- โมเดลจากข้อมูลและวิธีการฝึกสอนที่ต่างกัน

Video

cascade_PkC17

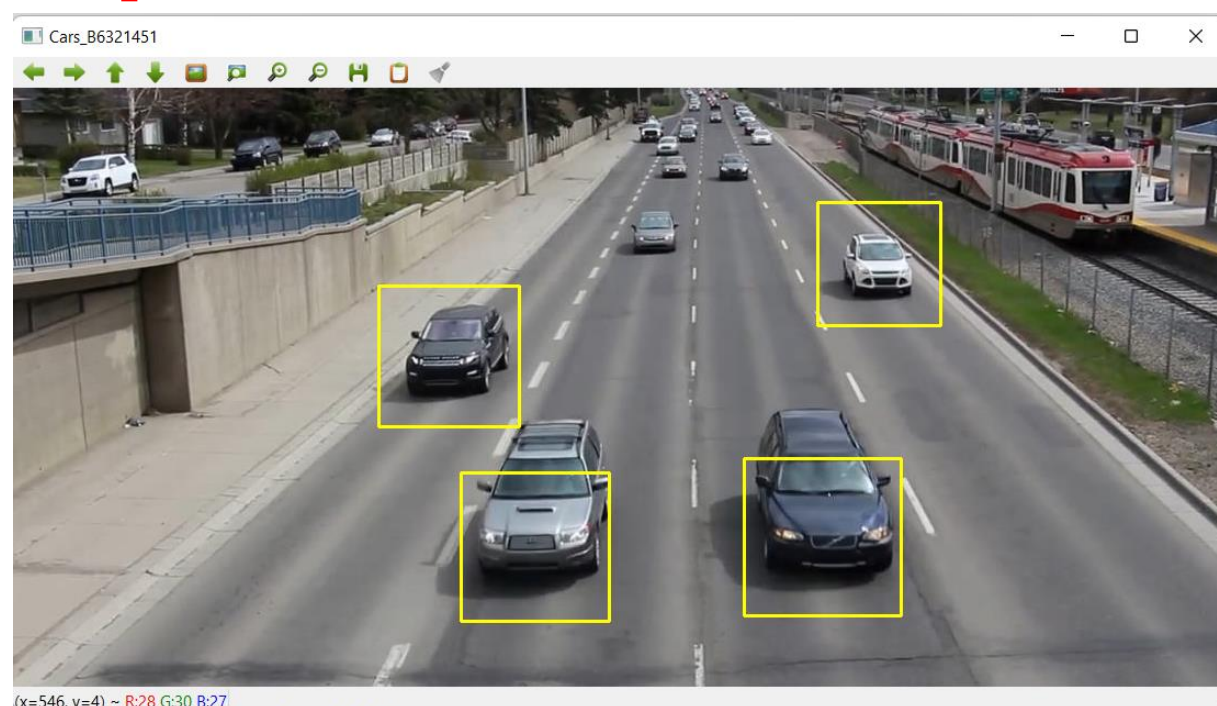


cascade_car = haarcascade_car

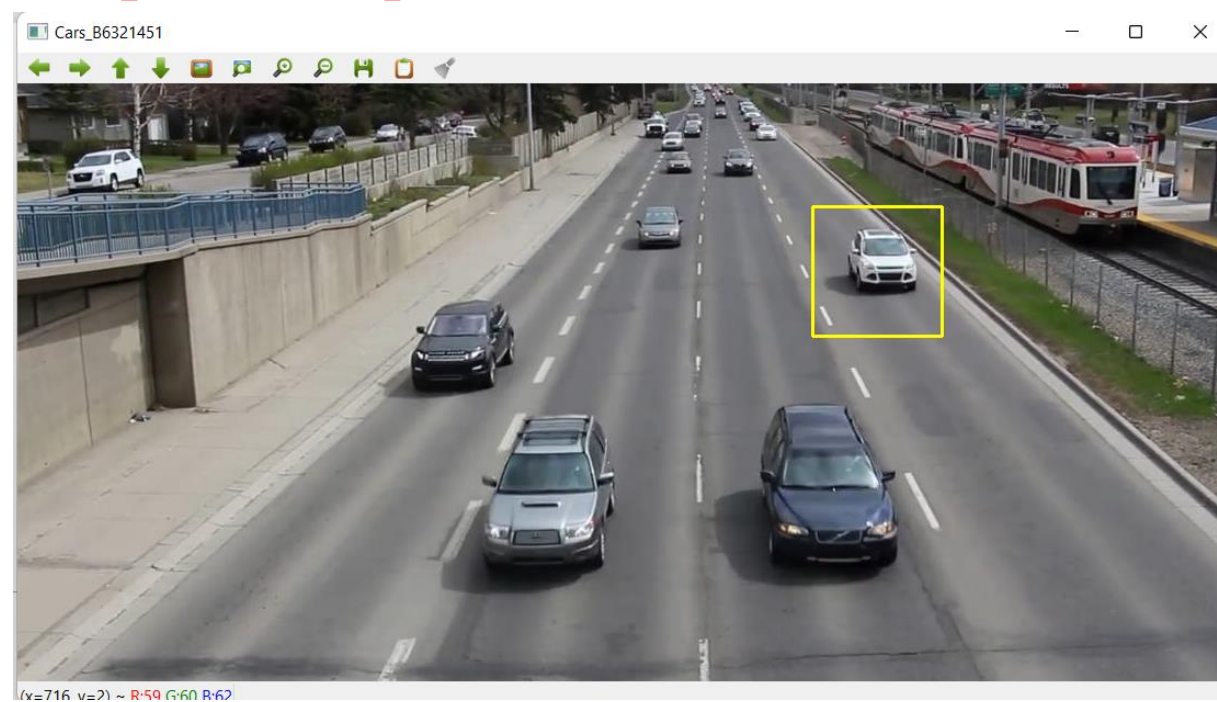


Picture

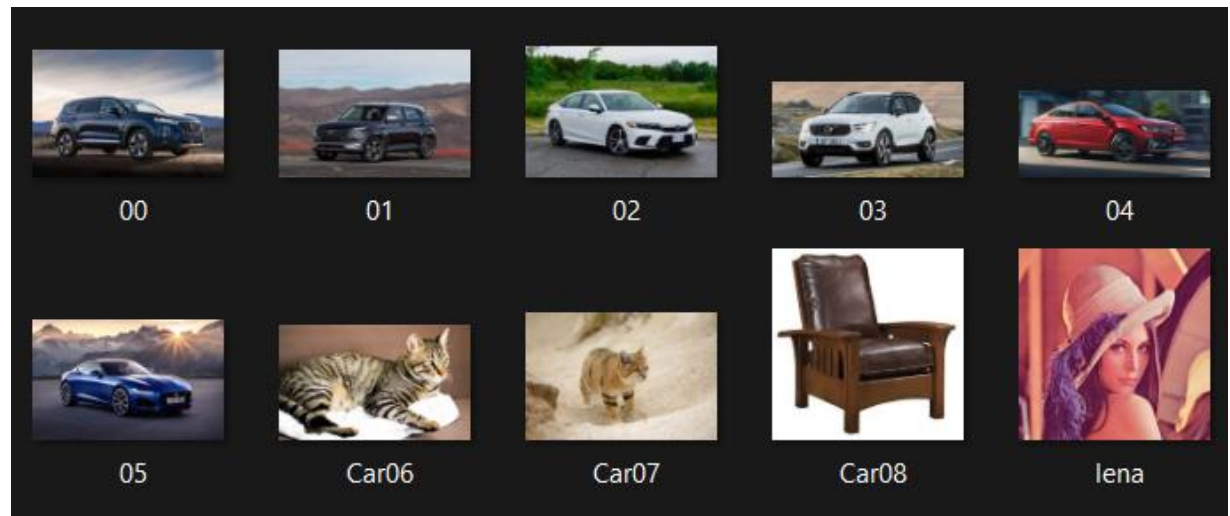
cascade_PkC17



cascade_car = haarcascade_car

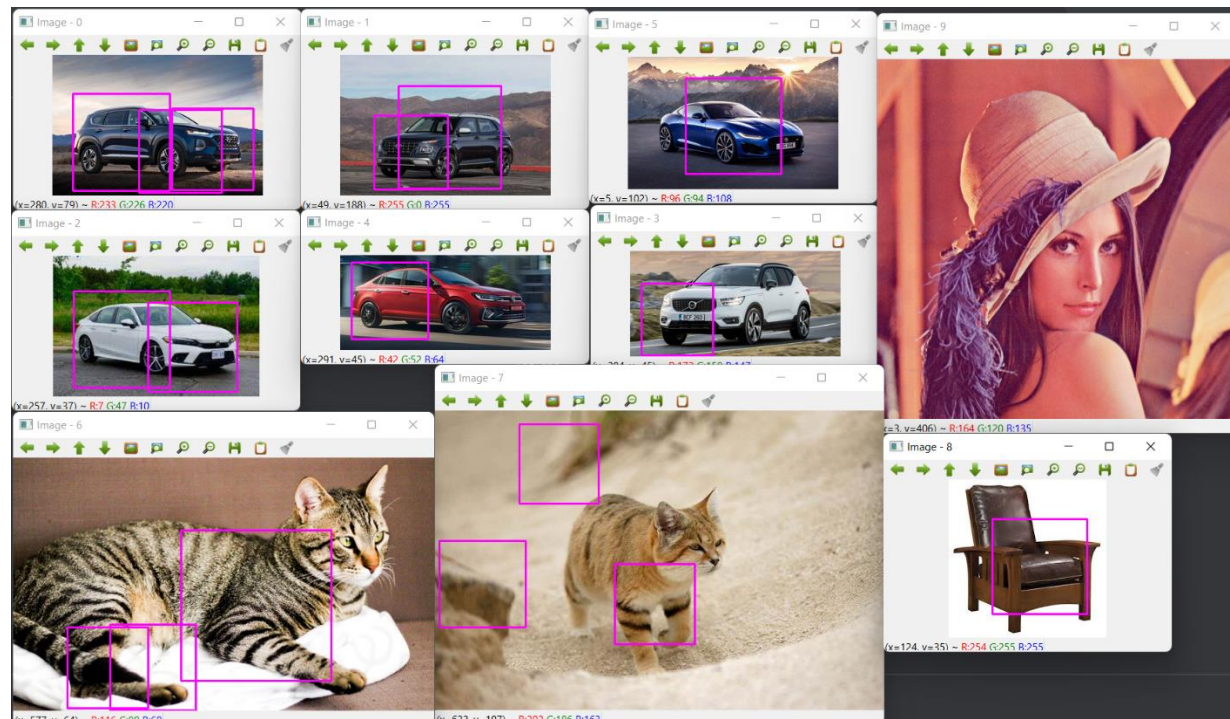


- หาภาพรถใหม่แทนภาพที่ให้มาทั้ง 6 ภาพ นำมารวมกับ Cat, Cat, Chair, lena

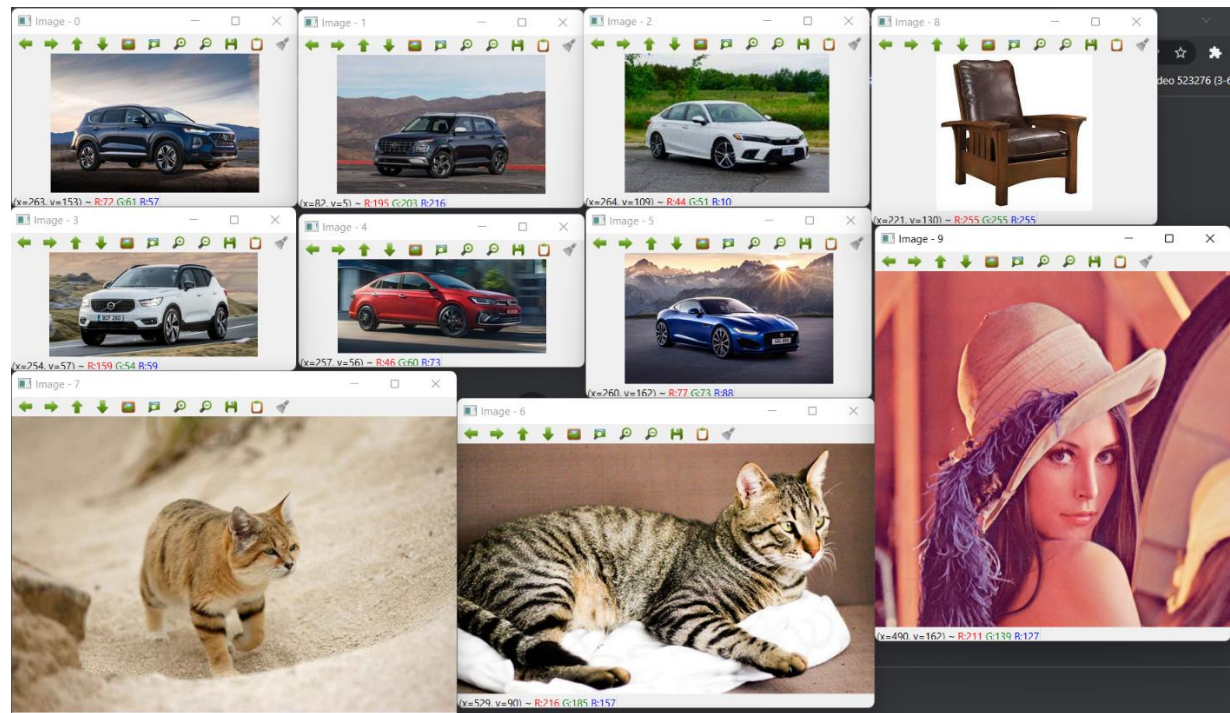


- ทดสอบและอภิปรายผล

cascade_PkC17



`cascade_car = haarcascade_car`



อภิปรายผล

จากการทดสอบหาภาพรถ cascade_PkC17 เทียบกับ cascade_car พบว่า cascade_PkC17 สามารถตรวจจับภาพรถได้ แต่ยังไม่แม่นยำมากพอ เพราะบางรูปเช่นรูปแมวยังตีกรอบว่าเป็นรถ จึงทำให้เห็นว่ายังไม่แม่นยำ

cascade_car จากภาพที่ทดสอบไม่สามารถสร้างกรอบสี่เหลี่ยมได้เลย

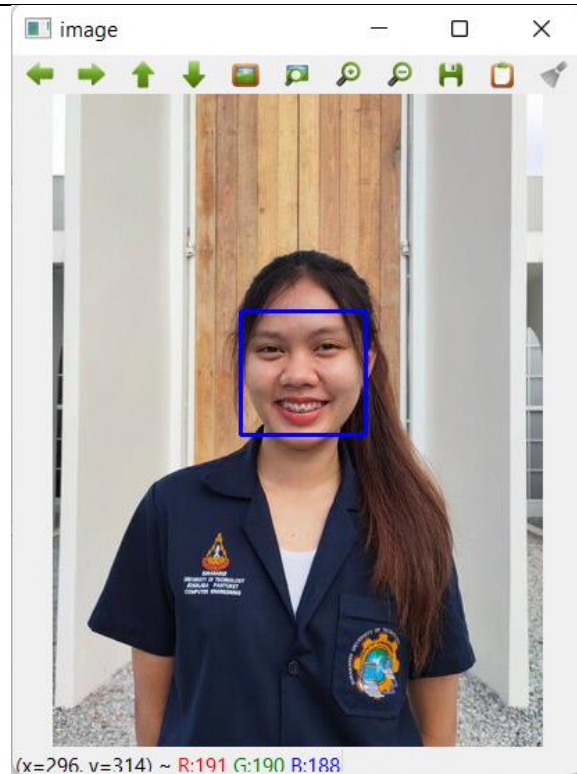
สรุป cascade_PkC17 ผลลัพธ์ดีกว่า cascade_car

5.1 - Face Detection

Face Detection (1)

```
In [*]: 1 import face_recognition
        2 import numpy as np
        3 import cv2
        4 image = cv2.imread("D:\\Machine\\Week10\\me\\1.jpg")
        5 face_locations = face_recognition.face_locations(image)
        6 (top, right, bottom, left) = face_locations[0]
        7 image = cv2.rectangle(image, (left, top), (right, bottom), (255,0,0), 2)
        8 cv2.imshow('image', image)
        9 cv2.waitKey(0)
       10 cv2.destroyAllWindows()
```

```
import face_recognition
import numpy as np
import cv2
image = cv2.imread("D:\\Machine\\Week10\\me\\1.jpg")
face_locations = face_recognition.face_locations(image)
(top, right, bottom, left) = face_locations[0]
image = cv2.rectangle(image, (left, top), (right, bottom), (255,0,0), 2)
cv2.imshow('image', image)
cv2.waitKey(0)
cv2.destroyAllWindows()
```



Face Detection (2) วาด Landmark

```

In [1]: 1 import face_recognition
        2 import numpy as np
        3 import cv2
        4
        5 image = cv2.imread("D:\\Machine\\Week10\\me\\1.jpg")
        6 h, w, c = image.shape
        7 face_locations = face_recognition.face_locations(image)
        8 face_landmarks_list = face_recognition.face_landmarks(image)
        9 left_eye_poly = np.array(face_landmarks_list[0]['left_eye']).reshape((-1,1,2))
       10 right_eye_poly = np.array(face_landmarks_list[0]['right_eye']).reshape((-1,1,2))
       11 top_lip_poly = np.array(face_landmarks_list[0]['top_lip']).reshape((-1,1,2))
       12 bottom_lip_poly = np.array(face_landmarks_list[0]['bottom_lip']).reshape((-1,1,2))
       13
       14 (top, right, bottom, left) = face_locations[0]
       15 cv2.rectangle(image, (left, top), (right, bottom), (255,0,0),2)
       16 cv2.polylines(image, left_eye_poly, True, (0,255,255))
       17 cv2.polylines(image, right_eye_poly, True, (0,255,255))
       18 cv2.polylines(image, top_lip_poly, True, (0,255,255))
       19 cv2.polylines(image, bottom_lip_poly, True, (0,255,255))

```

```

Out[1]: array([[255, 255, 243],
               [255, 255, 243],
               [255, 255, 243],
               ...,
               [254, 253, 255],
               [254, 253, 255],
               [254, 254, 254]],

              [[255, 255, 243],
               [255, 255, 243],
               [255, 255, 243],
               ...,
               [254, 253, 255],
               [254, 253, 255],
               [254, 254, 254]]])

```

```

import face_recognition
import numpy as np
import cv2

image = cv2.imread("D:\\Machine\\Week10\\me\\1.jpg")
h, w, c = image.shape
face_locations = face_recognition.face_locations(image)
face_landmarks_list = face_recognition.face_landmarks(image)
left_eye_poly = np.array(face_landmarks_list[0]['left_eye']).reshape((-1,1,2))
right_eye_poly = np.array(face_landmarks_list[0]['right_eye']).reshape((-1,1,2))
top_lip_poly = np.array(face_landmarks_list[0]['top_lip']).reshape((-1,1,2))
bottom_lip_poly = np.array(face_landmarks_list[0]['bottom_lip']).reshape((-1,1,2))

(top, right, bottom, left) = face_locations[0]
cv2.rectangle(image, (left, top), (right, bottom), (255,0,0),2)

```



```

cv2.polylines(image, left_eye_poly, True, (0,255,255))
cv2.polylines(image, right_eye_poly, True, (0,255,255))
cv2.polylines(image, top_lip_poly, True, (0,255,255))
cv2.polylines(image, bottom_lip_poly, True, (0,255,255))

```

Face Detection (3) Landmark

```

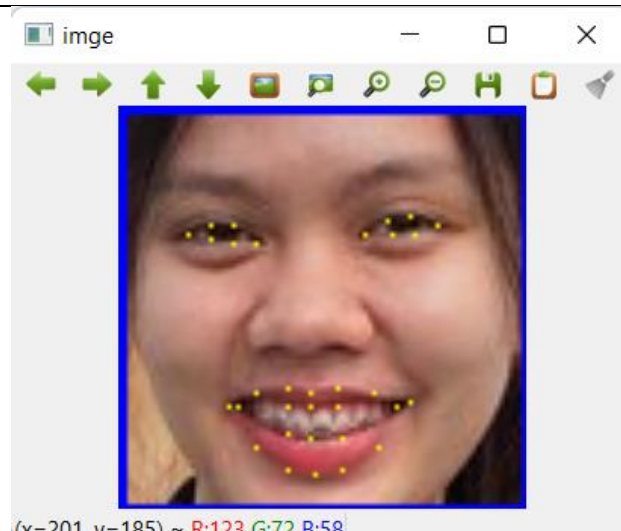
In [*]: 1 crop_image = image[top:bottom, left:right]
        2 crop_image = cv2.resize(crop_image, None, fx=3, fy=3)
        3 cv2.imshow('image', crop_image)
        4 cv2.waitKey(0)
        5 cv2.destroyAllWindows()

```

```

crop_image = image[top:bottom, left:right]
crop_image = cv2.resize(crop_image, None, fx=3, fy=3)
cv2.imshow('image', crop_image)
cv2.waitKey(0)
cv2.destroyAllWindows()

```



Face Detection (4)

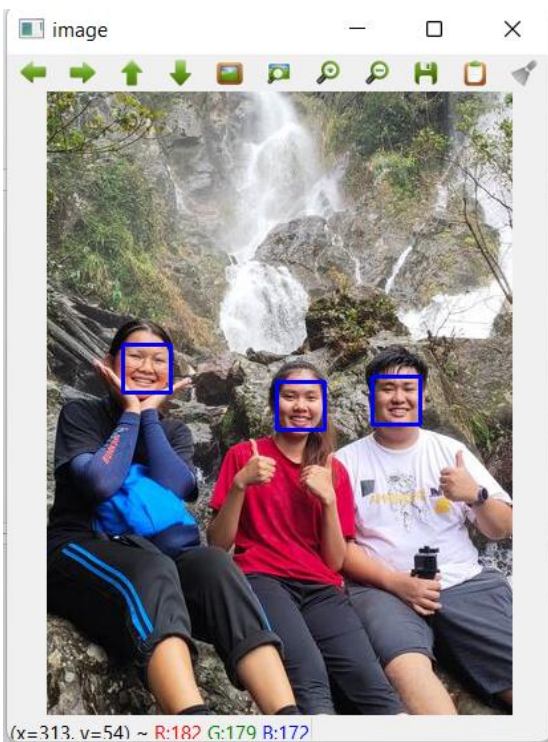
```

In [ ]: 1 import face_recognition
        2 import numpy as np
        3 import cv2
        4 image = cv2.imread("D:\\Machine\\Week10\\me\\3.jpg")
        5 face_locations = face_recognition.face_locations(image)
        6 for face_location in face_locations:
        7     (top, right, bottom, left) = face_location
        8     image = cv2.rectangle(image, (left, top), (right, bottom), (255,0,0), 2)
        9 cv2.imshow('image', image)
       10 cv2.waitKey(0)
       11 cv2.destroyAllWindows()

```



```
import face_recognition
import numpy as np
import cv2
image = cv2.imread("D:\\Machine\\Week10\\me\\3.jpg")
face_locations = face_recognition.face_locations(image)
for face_location in face_locations:
    (top, right, bottom, left) = face_location
    image = cv2.rectangle(image, (left, top), (right, bottom), (255,0,0), 2)
cv2.imshow('image', image)
cv2.waitKey(0)
cv2.destroyAllWindows()
```



Face Detection (5) จากสตรีมวิดีโอ

```

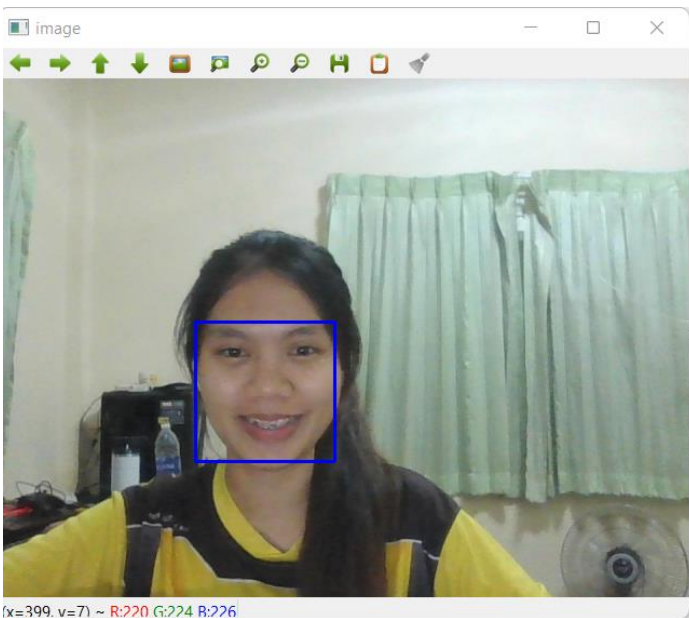
In [ ]: 1 import face_recognition
        2 import numpy as np
        3 import cv2
        4 video_capture = cv2.VideoCapture(0)
        5 while True:
        6     ret, frame = video_capture.read()
        7     face_locations = face_recognition.face_locations(frame)
        8     for face_location in face_locations:
        9         (top, right, bottom, left) = face_location
        10         frame = cv2.rectangle(frame, (left, top), (right, bottom), (255, 0, 0), 2)
        11     cv2.imshow('image', frame)
        12     if cv2.waitKey(1) & 0xFF == ord('q'):
        13         break
        14 cv2.destroyAllWindows()

```

```

import face_recognition
import numpy as np
import cv2
video_capture = cv2.VideoCapture(0)
while True:
    ret, frame = video_capture.read()
    face_locations = face_recognition.face_locations(frame)
    for face_location in face_locations:
        (top, right, bottom, left) = face_location
        frame = cv2.rectangle(frame, (left, top), (right, bottom), (255, 0, 0), 2)
    cv2.imshow('image', frame)
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break
cv2.destroyAllWindows()

```



5.2 - Face Recognitions

```

In [1]: 1 import face_recognition
        2 import numpy as np
        3 import cv2
        4 known_faces = [
        5     ( 'Gim', "D:\\Machine\\Week10\\me\\gim.jpg"),
        6     ( 'Non', "D:\\Machine\\Week10\\me\\non.jpg"),
        7     ( 'Au', "D:\\Machine\\Week10\\me\\au.jpg"),
        8 ]
        9 known_face_names = []
       10 known_face_encodings = []
       11 for face in known_faces:
       12     known_face_names.append(face[0])
       13     face_image = face_recognition.load_image_file(face[1])
       14     face_encoding = face_recognition.face_encodings(face_image)[0]
       15     known_face_encodings.append(face_encoding)

```

```

import face_recognition
import numpy as np
import cv2
known_faces = [
    ( 'Gim', "D:\\Machine\\Week10\\me\\gim.jpg"),
    ( 'Non', "D:\\Machine\\Week10\\me\\non.jpg"),
    ( 'Au', "D:\\Machine\\Week10\\me\\au.jpg"),
]
known_face_names = []
known_face_encodings = []
for face in known_faces:
    known_face_names.append(face[0])
    face_image = face_recognition.load_image_file(face[1])
    face_encoding = face_recognition.face_encodings(face_image)[0]
    known_face_encodings.append(face_encoding)

```

```

In [2]: 1 image = cv2.imread("D:\\Machine\\Week10\\me\\3.jpg")
        2 face_locations = face_recognition.face_locations(image)
        3 face_encodings = face_recognition.face_encodings(image, face_locations)
        4 face_names = []
        5 for face_encoding in face_encodings:
        6     matches = face_recognition.compare_faces(known_face_encodings, face_encoding)
        7     name = "Unknown"
        8     face_distances = face_recognition.face_distance(known_face_encodings, face_encoding)
        9     best_match_index = np.argmin(face_distances)
       10     if matches[best_match_index]:
       11         name = known_face_names[best_match_index]
       12     face_names.append(name)
       13     print(known_face_names, face_distances)

['Gim', 'Non', 'Au'] [0.2299222  0.4684679  0.45339485]
['Gim', 'Non', 'Au'] [0.46438315 0.26019209 0.4393081 ]
['Gim', 'Non', 'Au'] [0.48614609 0.4616963  0.22365476]

```

```

image = cv2.imread("D:\\Machine\\Week10\\me\\3.jpg")
face_locations = face_recognition.face_locations(image)
face_encodings = face_recognition.face_encodings(image, face_locations)
face_names = []
for face_encoding in face_encodings:
    matches = face_recognition.compare_faces(known_face_encodings, face_encoding)
    name = "Unknown"
    face_distances = face_recognition.face_distance(known_face_encodings, face_encoding)
    best_match_index = np.argmin(face_distances)
    if matches[best_match_index]:
        name = known_face_names[best_match_index]
    face_names.append(name)
    print(known_face_names, face_distances)

```



```
In [*]: 1 for face_location, name in zip(face_locations, face_names):
2         (top, right, bottom, left) = face_location
3         cv2.rectangle(image, (left, top), (right, bottom), (255,0,0), 2)
4         font = cv2.FONT_HERSHEY_DUPLEX
5
6         cv2.putText(image, name, (left + 6, bottom - 6), font, 1.0, (255, 255, 255),1)
7     cv2.imshow('image', image)
8     cv2.waitKey(0)
9     cv2.destroyAllWindows()
```

```
for face_location, name in zip(face_locations, face_names):
```

```
    (top, right, bottom, left) = face_location
```

```
    cv2.rectangle(image, (left, top), (right, bottom), (255,0,0), 2)
```

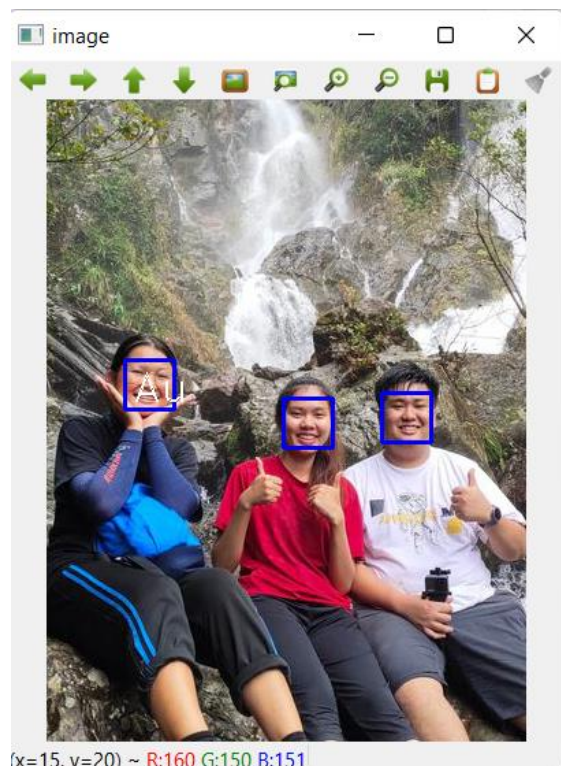
```
    font = cv2.FONT_HERSHEY_DUPLEX
```

```
cv2.putText(image, name, (left + 6, bottom - 6), font, 1.0, (255, 255, 255),1)
```

```
cv2.imshow('image', image)
```

```
cv2.waitKey(0)
```

```
cv2.destroyAllWindows()
```



```

In [*]: 1 video_capture = cv2.VideoCapture(0)
        2 while True:
        3     ret, frame = video_capture.read()
        4     face_locations = face_recognition.face_locations(frame)
        5     face_encodings = face_recognition.face_encodings(frame, face_locations)
        6     face_names = []
        7     for face_encoding in face_encodings:
        8         matches = face_recognition.compare_faces(known_face_encodings, face_encoding)
        9         name = "Unknown"
       10         face_distances = face_recognition.face_distance(known_face_encodings, face_encoding)
       11         best_match_index = np.argmin(face_distances)
       12         if matches[best_match_index]:
       13             name = known_face_names[best_match_index]
       14         face_names.append(name)
       15
       16     for face_location, name in zip(face_locations, face_names):
       17         (top, right, bottom, left) = face_location
       18         cv2.rectangle(frame, (left, top), (right, bottom), (255,0,0), 2)
       19         font = cv2.FONT_HERSHEY_DUPLEX
       20         cv2.putText(frame, name, (left+6, bottom-6), font, 0.75, (255,255,255), 1)
       21     cv2.imshow('image', frame)
       22     if cv2.waitKey(1) & 0xFF == ord('q'):
       23         break
       24     video_capture.release()
       25     cv2.destroyAllWindows()

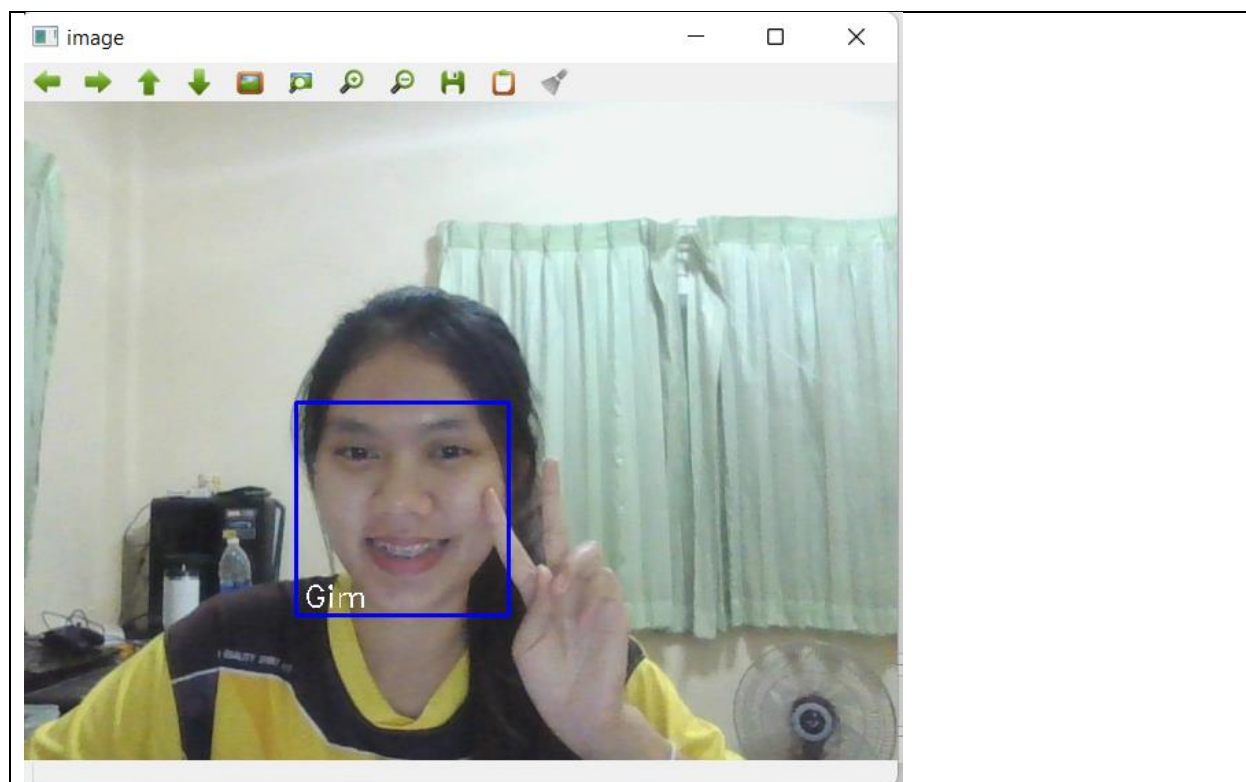
```

```

video_capture = cv2.VideoCapture(0)
while True:
    ret, frame = video_capture.read()
    face_locations = face_recognition.face_locations(frame)
    face_encodings = face_recognition.face_encodings(frame, face_locations)
    face_names = []
    for face_encoding in face_encodings:
        matches = face_recognition.compare_faces(known_face_encodings, face_encoding)
        name = "Unknown"
        face_distances = face_recognition.face_distance(known_face_encodings, face_encoding)
        best_match_index = np.argmin(face_distances)
        if matches[best_match_index]:
            name = known_face_names[best_match_index]
        face_names.append(name)

    for face_location, name in zip(face_locations, face_names):
        (top, right, bottom, left) = face_location
        cv2.rectangle(frame, (left, top), (right, bottom), (255,0,0), 2)
        font = cv2.FONT_HERSHEY_DUPLEX
        cv2.putText(frame, name, (left+6, bottom-6), font, 0.75, (255,255,255), 1)
    cv2.imshow('image', frame)
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break
    video_capture.release()
cv2.destroyAllWindows()

```



Week1020-Tesseract and Sudoku to Text

Q1. ทดสอบกับภาพข้อความภาษาอังกฤษแบบอื่นๆ จำนวนอย่างน้อย 2 ภาพ

ข้อความภาษาอังกฤษ ภาพที่ 1

As the name implies, Birds Rotisserie specializes in chicken recipes and they make them in the bona fide French style. Their farm-to-table menu created by chef Jeremy—a co-owner with experiences from a Michelin-star restaurant in France and a few big names in Thailand—and his crew, is uniquely simple yet full of flavors. The guys have selected only the free-range chicken from an organic farm in Khao Yai and soaked them in their marinade made from secret ingredients and herbs for

```
In [1]: import cv2
from PIL import Image
import pytesseract
pytesseract.pytesseract.tesseract_cmd = r'C:\Program Files\Tesseract-OCR\tesseract.exe'

imageC = Image.open("D:\Machine\Week10\me\E1.jpg")
imageC.show()

text_from_image = pytesseract.image_to_string(imageC)
print(text_from_image)
```

As the name implies, Birds Rotisserie specializes in chicken recipes and they make them in the bona fide French style. Their farm-to-table menu created by chef Jeremy—a co-owner with experiences from a Michelin-star restaurant in France and a few big names in Thailand—and his crew, is uniquely simple yet full of flavors. The guys have selected only the free-range chicken from an organic farm in Khao Yai and soaked them in their marinade made from secret ingredients and herbs for

ข้อความภาษาอังกฤษ ภาพที่ 2

Abstract

This thesis aims to design and create demonstration set of electromagnetic levitation system using PLC and Pneumatics with transient electromagnetics from inductor coil which appropriate for work pieces holding and releasing.

The proposers designed the coil inductor set to generate the electromagnetics through no.20 copper coil with 2.188 amperes resistance. The copper coil was bended around E transformer iron bar for a hundred times.

```
In [2]: import cv2
from PIL import Image
import pytesseract
pytesseract.pytesseract.tesseract_cmd = r'C:\\Program Files\\Tesseract-OCR\\tesseract.exe'

imageC = Image.open("D:\\Machine\\Week10\\me\\E2.jpg")
imageC.show()

text_from_image = pytesseract.image_to_string(imageC)
print(text_from_image)
```

Abstract

This thesis aims to design and _ create demonstration set of electromagnetic levitation system using PLC and Pneumatics – with transient electromagnetics from inductor coil which appropriate for work pieces holding and releasing.

The proposers designed the coil inductor set to generate the electromagnetics through no.20 copper coil with 2.188 amperes resistance. The copper coil was

bended around E transformer iron bar for a hundred times.

Q2. ทดสอบกับภาพข้อความภาษาไทยแบบอื่นๆ จำนวนอย่างน้อย 2 ภาพ

ข้อความภาษาไทยภาพที่ 1

ภาษาไทย เป็นภาษาทางการของประเทศไทย และภาษาแม่ของชาวไทย และชนเชื้อสายอื่นในประเทศไทย ภาษาไทยเป็นภาษาในกลุ่มภาษาไต ซึ่งเป็นกลุ่มย่อยของตระกูลภาษาไท-กะได สันนิษฐานว่า ภาษาในตระกูลนี้มีถิ่นกำเนิดจากทางตอนใต้ของประเทศจีน และนักภาษาศาสตร์บางท่านเสนอว่า ภาษาไทยน่าจะมีความเชื่อมโยงกับตระกูลภาษาออสโตร-เอเชียติก ตระกูลภาษาออสโตรนีเซียน ตระกูลภาษาจีน-ทิเบต

ภาษาไทยเป็นภาษาที่มีระดับเสียงของคำแน่นอนหรือวรรณยุกต์เช่นเดียวกับภาษาจีน และออกเสียงแยกคำต่อคำ เป็นที่ลำบากของชาวต่างชาติเนื่องจาก การออกเสียงวรรณยุกต์ที่เป็นเอกลักษณ์ของแต่ละคำ และการสะกดคำที่ซับซ้อน นอกจากภาษากลางแล้ว ในประเทศไทยมีการใช้ ภาษาไทยถิ่นอื่นด้วย

```
In [2]: 1 # Test-I - Thai invoid
2 import cv2
3 import pytesseract
4
5 pytesseract.pytesseract.tesseract_cmd = r'C:\\Program Files\\Tesseract-OCR\\tesseract.exe'
6 custom_config = '-l tha+eng --psm 6'
7
8 imageC = cv2.imread("D:\\Machine\\Week10\\me\\T1.jpg")
9 text_from_image = pytesseract.image_to_string(imageC, config=custom_config)
10 print(text_from_image)
11
12 cv2.imshow('img', imageC)
13 cv2.waitKey(0)
14 cv2.destroyAllWindows()
15
```

ภาษาไทย เป็นภาษาทางการของประเทศไทย และภาษาแม่ของชาวไทย และชนเชื้อสายอื่นในประเทศไทย ภาษาไทยเป็นภาษาในกลุ่มภาษาไต ซึ่งเป็นกลุ่มย่อยของตระกูลภาษาไท-กะได สันนิษฐานว่า ภาษาในตระกูลนี้มีถิ่นกำเนิดจากทางตอนใต้ของประเทศจีน และนักภาษาศาสตร์บางท่านเสนอว่า ภาษาไทยน่าจะมีความเชื่อมโยงกับตระกูลภาษาออสโตร-เอเชียติก

ติก ตระกูลภาษาออสโตรนีเซียน ตระกูลภาษาจีน-ทิเบต

ภาษาไทยเป็นภาษาที่มีระดับเสียงของคำแน่นอนหรือวรรณยุกต์เช่นเดียวกับภาษาจีน และออกเสียงแยกคำต่อคำ เป็นที่

ลำบากของชาวต่างชาติเนื่องจาก การออกเสียงวรรณยุกต์ที่เป็นเอกลักษณ์ของแต่ละคำ และการสะกดคำที่ซับซ้อน นอกจากภาษากลางแล้ว ในประเทศไทยมีการใช้ ภาษาไทยถิ่นอื่นด้วย

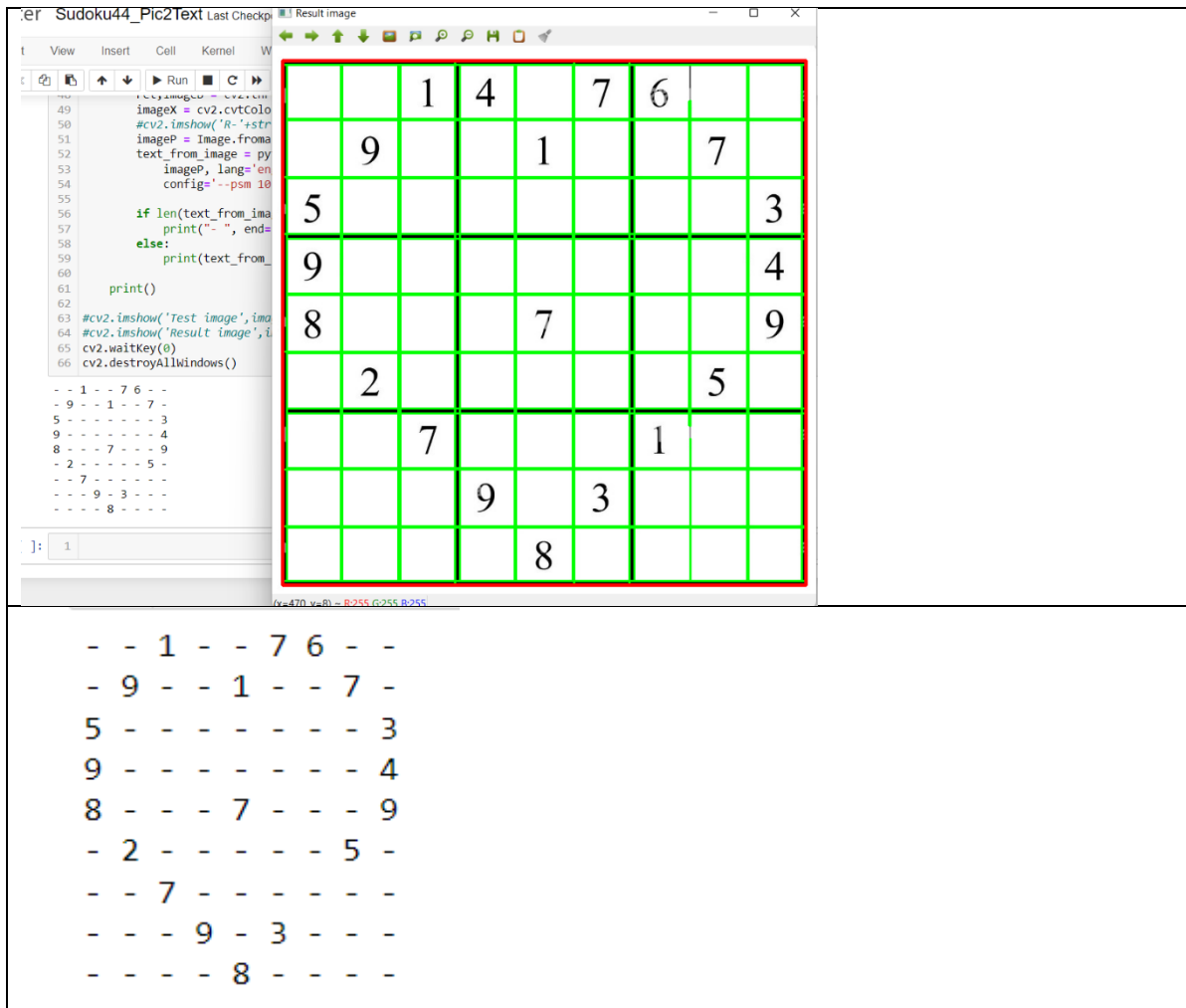
ข้อความภาษาไทยภาพที่ 2

คำว่า **ไทย** หมายความว่า อีสราภาพ เสรีภาพ หรืออีกความหมายหนึ่งคือ ใหญ่ ยิ่งใหญ่ เพราะการจะเป็นอิสระได้จะต้องมีกำลังที่มากกว่า แข็งแกร่งกว่า เพื่อป้องกันการรุกรานจากข้าศึก แม้คำนี้จะมีความหมายเหมือนคำยืมจากภาษาบาลีสันสกฤต แต่แท้ที่จริงแล้ว คำนี้เป็นคำไทยแท้ที่เกิดจากกระบวนการสร้างคำที่เรียกว่า 'การลากคำเข้าวัด' ซึ่งเป็นการลากความวิธีหนึ่ง ตามหลักคติชนวิทยา คนไทยเป็นชนชาติที่นับถือกันว่า ภาษาบาลีซึ่งเป็นภาษาที่บันทึกพระธรรมคำสอนของพระพุทธเจ้าเป็นภาษาอันศักดิ์สิทธิ์และเป็นมงคล เมื่อคนไทยต้องการตั้งชื่อประเทศว่า **ไท** ซึ่งเป็นคำไทยแท้ จึงเติมตัว **ย** เข้าไปข้างท้าย เพื่อให้มีลักษณะคล้ายคำในภาษาบาลีสันสกฤตเพื่อความเป็นมงคลตามความเชื่อของตน ภาษาไทยจึงหมายถึงภาษาของชนชาติไทยผู้เป็นไทนั่นเอง

```
In [3]: 1 # Test-I - Thai invoid
2 import cv2
3 import pytesseract
4
5 pytesseract.pytesseract.tesseract_cmd = r'C:\\Program Files\\Tesseract-OCR\\tesseract.exe'
6 custom_config = '-l tha+eng --psm 6'
7
8 imageC = cv2.imread("D:\\Machine\\Week10\\me\\T2.jpg")
9 text_from_image = pytesseract.image_to_string(imageC, config=custom_config)
10 print(text_from_image)
11
12 cv2.imshow('img', imageC)
13 cv2.waitKey(0)
14 cv2.destroyAllWindows()
15
```

คำว่า **ไทย** หมายความว่า อีสราภาพ เสรีภาพ หรืออีกความหมายหนึ่งคือ ใหญ่ ยิ่งใหญ่ เพราะการจะเป็นอิสระได้จะต้องมีกำลังที่มากกว่า แข็งแกร่งกว่า เพื่อป้องกันการรุกรานจากข้าศึก แม้คำนี้จะมีความหมายเหมือนคำยืมจากภาษาบาลีสันสกฤต แต่แท้ที่จริงแล้ว คำนี้เป็นคำไทยแท้ที่เกิดจากกระบวนการสร้างคำที่เรียกว่า 'การลากคำเข้าวัด' ซึ่งเป็นการลากความวิธีหนึ่ง ตามหลักคติชนวิทยา คนไทยเป็นชนชาติที่นับถือกันว่า ภาษาบาลีซึ่งเป็นภาษาที่บันทึกพระธรรมคำสอนของพระพุทธเจ้าเป็นภาษาอันศักดิ์สิทธิ์และเป็นมงคล เมื่อคนไทยต้องการตั้งชื่อประเทศว่า **ไท** ซึ่งเป็นคำไทยแท้ จึงเติมตัว **ย** เข้าไปข้างท้าย เพื่อให้มีลักษณะคล้ายคำในภาษาบาลีสันสกฤตเพื่อความเป็นมงคลตามความเชื่อของตน ภาษาไทยจึงหมายถึงภาษาของชนชาติไทยผู้เป็นไทนั่นเอง

Q3. ทดสอบกับภาพ Sudoku อื่นๆ จำนวนอย่างน้อย 2 ภาพ



```

Sudoku44_Pic2Text Last Checkpoint
t View Insert Cell Kernel W
c
49 imageX = cv2.cvtColor
50 #cv2.imshow('R'+str
51 imageP = Image.froma
52 text_from_image = py
53 imageP, lang='eng',
54 config='--psm 10
55
56 if len(text_from_ima
57 print("-", end=
58 else:
59 print(text_from_
60
61 print()
62
63 #cv2.imshow('Test image', ima
64 #cv2.imshow('Result image', i
65 cv2.waitKey(0)
66 cv2.destroyAllWindows()
-- 1 - - 7 6 - -
- 9 - - 1 - - 7 -
5 - - - - - - 3
9 - - - - - - 4
8 - - - 7 - - - 9
- 2 - - - - - 5 -
- - 7 - - - - -
- - - 9 - 3 - - -
- - - 8 - - - -
] : 1
(x=470, y=8) - R:255 G:255 B:255

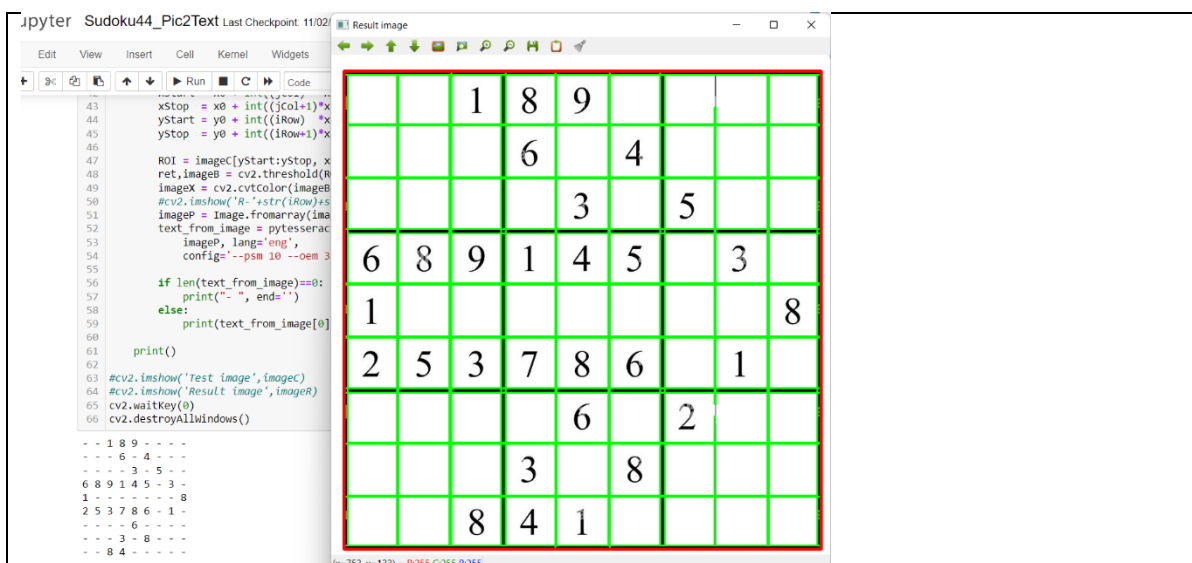
```

		1	4		7	6		
	9			1			7	
5								3
9								4
8				7				9
	2						5	
		7				1		
			9		3			
				8				

```

- - 1 - - 7 6 - -
- 9 - - 1 - - 7 -
5 - - - - - - 3
9 - - - - - - 4
8 - - - 7 - - - 9
- 2 - - - - - 5 -
- - 7 - - - - -
- - - 9 - 3 - - -
- - - 8 - - - -

```



```

Jupyter Sudoku44_Pic2Text Last Checkpoint: 11/02/
Edit View Insert Cell Kernel Widgets
43 xstop = x0 + int((iCol+1)*x
44 ystart = y0 + int((iRow)*x
45 ystop = y0 + int((iRow+1)*x
46
47 ROI = imageC[ystart:ystop, x
48 ret, imageB = cv2.threshold(R
49 imageX = cv2.cvtColor(imageB
50 #cv2.imshow('R'+str(iRow)+s
51 imageP = Image.fromarray(ima
52 text_from_image = pytesseract
53 imageP, lang='eng',
54 config='--psm 10 --oem 3
55
56 if len(text_from_image)==0:
57 print("-", end='')
58 else:
59 print(text_from_image[0])
60
61 print()
62
63 #cv2.imshow('Test image', imageC)
64 #cv2.imshow('Result image', imageR)
65 cv2.waitKey(0)
66 cv2.destroyAllWindows()
-- 1 8 9 - - -
- - 6 - 4 - -
- - - 3 - 5 -
6 8 9 1 4 5 - 3 -
1 - - - - - 8
2 5 3 7 8 6 - 1 -
- - - 6 - - -
- - - 3 - 8 - -
- 8 4 - - - -
] : 1
(x=753, y=133) - R:255 G:255 B:255

```

		1	8	9				
			6		4			
				3		5		
6	8	9	1	4	5		3	
1								8
2	5	3	7	8	6		1	
				6		2		
			3		8			
		8	4	1				

```

-- 1 8 9 - - -
- - 6 - 4 - -
- - - 3 - 5 -
6 8 9 1 4 5 - 3 -
1 - - - - - 8
2 5 3 7 8 6 - 1 -
- - - 6 - - -
- - - 3 - 8 - -
- 8 4 - - - -

```



```

- - 1 8 9 - - - -
- - - 6 - 4 - - -
- - - - 3 - 5 - -
6 8 9 1 4 5 - 3 -
1 - - - - - - 8
2 5 3 7 8 6 - 1 -
- - - - 6 - - - -
- - - 3 - 8 - - -
- - 8 4 - - - - -

```

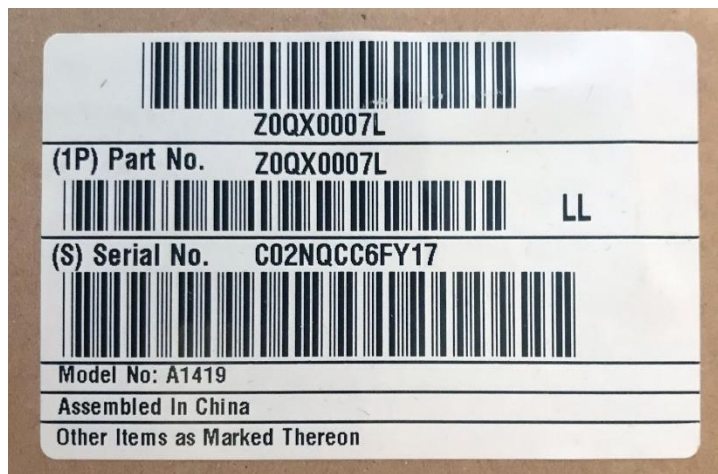
Q4. ให้ออกแบบระบบเพื่ออ่าน Serial Number ของอุปกรณ์ที่เคลื่อนที่บนสายพานลำเลียง เพื่อเก็บว่ามีอุปกรณ์ S/N อะไรผ่านไปบ้าง รวมทั้งหมดกี่ตัว

Your Text Here

Optional 2nd line here



123456



Week1030-Count and Classification

Q31: ทดสอบ Key Point Matching – Video Project นับของเพียง 1 ตัวอย่าง (ให้เน้นที่ Video_B ส่วน Video_A เพียงแค่นำโค้ดจาก VideoB มารันทดสอบ)

บันทึกผลในตาราง

VideoB_Type1 YouTube Video Link = <https://youtu.be/AAAAAAAAA>

VideoA_Type1 YouTube Video Link = <https://youtu.beBBBBBBBBBB>

ลำดับ	รายการ	Count	จำนวนชิ้นงาน จริง	ผลการนับด้วย โปรแกรม	ผลต่าง	ถูกต้อง(%)
1	Video_A1	นับเฉพาะชนิดที่ 1	28	1	1/28	0.035
2	Video_B1	นับเฉพาะชนิดที่ 1	10	2	2/10	0.2

ข้อจำกัด ปัญหา ข้อเสนอแนะ

วิดีโอ VideoA_Type1 ไม่สามารถนับได้ เนื่องจาก

พื้นหลังที่เป็นสายพาน ทำให้ค่าที่อ่านจากการจับวัตถุ

คงที่ จึงไม่มีการนับจำนวนชิ้นงาน

วิดีโอ VideoB_Type1 ยังไม่สามารถนับจำนวนของ

วัตถุได้ถูกต้องและแม่นยำ

อีกข้อจำกัดคือ โค้ดที่ได้ปรับแก้ของหนูไม่สามารถ export

วิดีโอ VideoB_Type1 ได้ค่ะ

Code Text

```
# Step 30 - Keypoint Mactching on Video
```

```
# If Match Point > refGoodPoint(8) --> this frame is Match
```

```
# if Sum(20 Match frame) == 0 --> Count Index = +1
```

```
# if Sum(20 Match frame) > refFrameCount(10) --> Total = Total+Count Index, Count Index = 0
```

```
import cv2
```

```
import numpy as np
```

```
nDimGoodArray = 20
```

```
refGoodPoint = 8
```

```
refFrameCount = 10
```

```
sift = cv2.xfeatures2d.SIFT_create()
```

```
#image1C = cv2.imread("./image/Skittles.jpg")
```

```
image1C = cv2.imread("D:\\Machine\\Week11\\im_me\\B.jpg")
```

```
image1G = cv2.cvtColor(image1C, cv2.IMREAD_GRAYSCALE)
```

```
kp1, des1 = sift.detectAndCompute(image1G, None)
```

```

#cap = cv2.VideoCapture("./image/Candy.mp4")
#cap = cv2.VideoCapture("./image/video_BU.avi")
cap = cv2.VideoCapture("D:\\Machine\\Week9\\video_B_new7.avi")

ret, frame = cap.read()
height, width = frame.shape[:2]
positionText = (30, int(0.20*height)) # 25%
runGoodArray1 = np.zeros(nDimGoodArray)
Obj1_Total, Obj1_Adder = 0, 0

while(cap.isOpened()):
    ret, image2C = cap.read()
    if ret == True:
        image2G = cv2.cvtColor(image2C, cv2.IMREAD_GRAYSCALE)
        kp2, des2 = sift.detectAndCompute(image2G, None)
        good = []
        if len(kp2) != 0 :
            match = cv2.BFMatcher()
            matches = match.knnMatch(des1, des2, k=2)
            for i_matche in range(len(matches)):
                try:
                    m, n = matches[i_matche]
                except (ValueError):
                    pass
                else:
                    if m.distance < 0.5 * n.distance :
                        good.append(m)

        for iShift in range(nDimGoodArray-1):
            runGoodArray1[iShift] = runGoodArray1[iShift+1]
        if len(good) > refGoodPoint:
            runGoodArray1[nDimGoodArray-1] = 1
        else:
            runGoodArray1[nDimGoodArray-1] = 0

        summFrame = runGoodArray1.sum(dtype=np.int32)
        if summFrame == 0:
            Obj1_Adder = 1

```

```
if summFrame > 10:
```

```
    Obj1_Total = Obj1_Total + Obj1_Adder
```

```
    Obj1_Adder = 0
```

```
textShow = str(Obj1_Total) + "<" + str(len(good)) + "," + str(summFrame) + ">"
```

```
cv2.putText(image2C, textShow, positionText, cv2.FONT_HERSHEY_PLAIN, 4, (0,0,255), 2)
```

```
match_result = cv2.drawMatches(image1C, kp1, image2C, kp2, good[:50], None, flags=2)
```

```
cv2.imshow('Frame', match_result)
```

```
if cv2.waitKey(1) & 0xFF == ord('q'):
```

```
    break
```

```
else:
```

```
    break
```

```
cap.release()
```

```
cv2.destroyAllWindows()
```

```
print("Object Type-1 = ", Obj1_Total)
```

Code Capture

```
In [*]: 1 import cv2
2 import numpy as np
3 nDimGoodArray = 20
4 refGoodPoint = 8
5 refFrameCount = 10
6
7 sift = cv2.xfeatures2d.SIFT_create()
8 #image1C = cv2.imread("./image/Skittles.jpg")
9 image1C = cv2.imread("D:\\Machine\\Week11\\im_me\\B.jpg")
10 image1G = cv2.cvtColor(image1C, cv2.IMREAD_GRAYSCALE)
11 kp1, des1 = sift.detectAndCompute(image1G, None)
12
13 #cap = cv2.VideoCapture("./image/Candy.mp4")
14 #cap = cv2.VideoCapture("./image/video_BU.avi")
15 cap = cv2.VideoCapture("D:\\Machine\\Week9\\video_B_new7.avi")
16
17 ret, frame = cap.read()
18 height, width = frame.shape[:2]
19 positionText = (30, int(0.20*height)) # 25%
20 runGoodArray1 = np.zeros(nDimGoodArray)
21 Obj1_Total, Obj1_Adder = 0, 0
22
23 while(cap.isOpened()):
24     ret, image2C = cap.read()
25     if ret == True:
26         image2G = cv2.cvtColor(image2C, cv2.IMREAD_GRAYSCALE)
27         kp2, des2 = sift.detectAndCompute(image2G, None)
28         good = []
29         if len(kp2) != 0 :
30             match = cv2.BFMatcher()
31             matches = match.knnMatch(des1, des2, k=2)
32             for i_matche in range(len(matches)):
33                 try:
34                     m, n = matches[i_matche]
35                     except (ValueError):
```

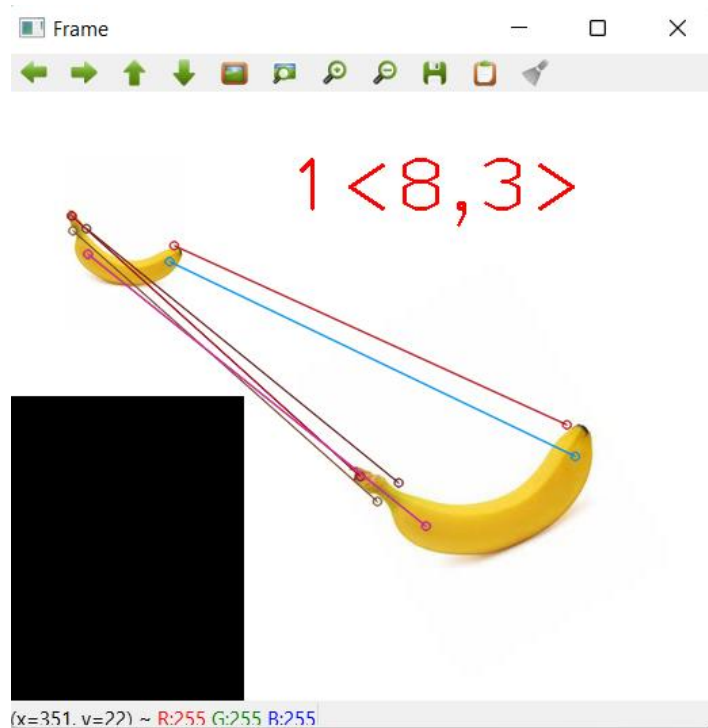


```

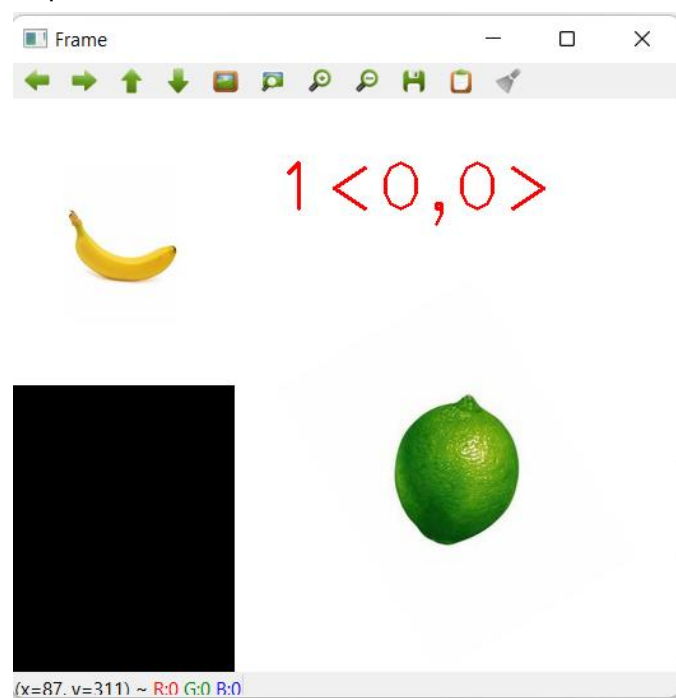
36         pass
37     else:
38         if m.distance < 0.5 * n.distance :
39             good.append(m)
40
41
42     for iShift in range(nDimGoodArray-1):
43         runGoodArray1[iShift] = runGoodArray1[iShift+1]
44     if len(good) > refGoodPoint:
45         runGoodArray1[nDimGoodArray-1] = 1
46     else:
47         runGoodArray1[nDimGoodArray-1] = 0
48
49     summFrame = runGoodArray1.sum(dtype=np.int32)
50     if summFrame == 0:
51         Obj1_Adder = 1
52     if summFrame > 10:
53         Obj1_Total = Obj1_Total + Obj1_Adder
54         Obj1_Adder = 0
55
56
57     textShow = str(Obj1_Total) + "<" + str(len(good)) + "," + str(summFrame) + ">"
58     cv2.putText(image2C, textShow, positionText, cv2.FONT_HERSHEY_PLAIN, 4, (0,0,255), 2)
59
60     match_result = cv2.drawMatches(image1C, kp1, image2C, kp2, good[:50], None, flags=2)
61
62     cv2.imshow('Frame', match_result)
63     if cv2.waitKey(1) & 0xFF == ord('q'):
64         break
65     else:
66         break
67
68 cap.release()
69 cv2.destroyAllWindows()
70 print("Object Type-1 = ", Obj1_Total)

```

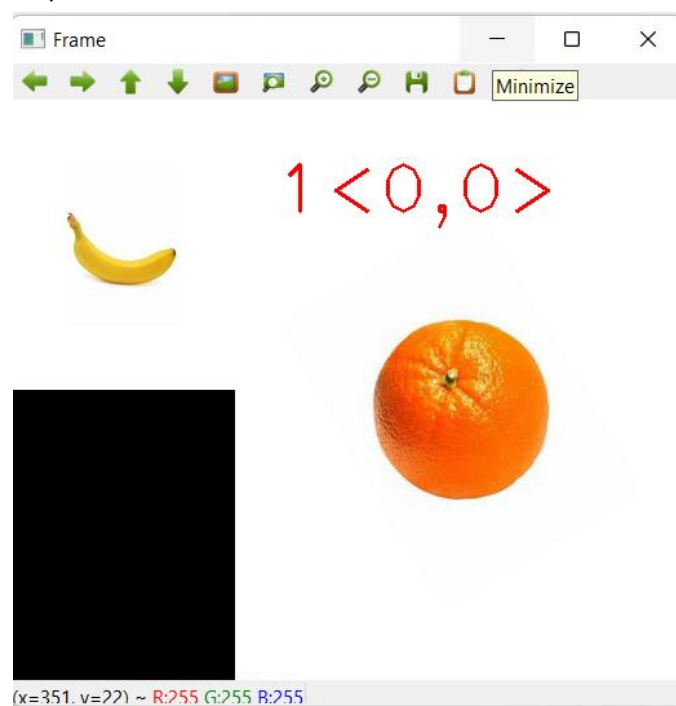
Capture Result-1



Capture Result-2



Capture Result-3



Q32: ทดสอบ Key Point Matching – Video Project นับของเพียง 1 ตัวอย่าง และนับจำนวนรวม (ให้เน้นที่ Video_B ส่วน Video_A เพียงแค่นำเข้าโค้ดจาก VideoB มารันทดสอบ)

บันทึกผลในตาราง

VideoB_Type1_CountAll YouTube Video Link = <https://youtu.be/CCCCCCCCCCCC>

VideoA_Type1_CountAll YouTube Video Link = <https://youtu.be/DDDDDDDDDDDD>

ลำดับ	รายการ	ผลการนับ	จำนวนชิ้นงานจริง	ผลการนับด้วยโปรแกรม	ผลต่าง	ถูกต้อง(%)
1	Video_A1	นับเฉพาะชนิดที่ 1	28	1	1/28	0.035
		นับรวมทุกชนิด	77	1	1/77	0.013
2	Video_B1	นับเฉพาะชนิดที่ 1	10	10	10/10	1
		นับรวมทุกชนิด	80	80	80/80	1

ข้อจำกัด ปัญหา ข้อเสนอแนะ

วิดีโอ VideoA_Type1_CountAll ไม่สามารถนับได้

เนื่องจากพื้นหลังที่เป็นสายพาน ทำให้คำที่อ่านจากการ

จับวัตถุคงที่ จึงไม่มีการนับจำนวนชิ้นงาน

วิดีโอ VideoB_Type1_CountAll สามารถนับจำนวนของ

วัตถุได้ถูกต้องและแม่นยำ

Code Text

```
# Step 30 - Keypoint Mactching on Video
import cv2
import numpy as np
dimensionFrameRecord = 20
minGoodPoint_A, maxFrame_AAdd, minFrame_ASet = 10, 14, 13
minGoodPoint_1, maxFrame_1Add, minFrame_1Set = 4, 10, 2

sift = cv2.xfeatures2d.SIFT_create()
#image1C = cv2.imread("./image/Skittles.jpg")
image1C = cv2.imread("D:\\Machine\\Week11\\im_me\\B.jpg")
image1G = cv2.cvtColor(image1C, cv2.IMREAD_GRAYSCALE)
kp1, des1 = sift.detectAndCompute(image1G, None)

#cap = cv2.VideoCapture("./image/Candy.mp4")
#cap = cv2.VideoCapture("./image/video_BU2.avi")
#cap = cv2.VideoCapture("./image/video_BU2S.avi")
#cap = cv2.VideoCapture("./image/video_BU2F.avi")
#cap = cv2.VideoCapture("D:\\Machine\\Week9\\video_B_new7.mp4")
cap = cv2.VideoCapture("D:\\Machine\\Week9\\video_BU1_gg_13.avi")
```

```

fourcc = cv2.VideoWriter_fourcc(*'MP4V')
frame_width = int(cap.get(cv2.CAP_PROP_FRAME_WIDTH))
frame_height = int(cap.get(cv2.CAP_PROP_FRAME_HEIGHT))
out = cv2.VideoWriter('./im_m1/B1_Weeek10.avi', fourcc, 30, (frame_width, frame_height))

ret, frame = cap.read()
height, width = frame.shape[:2]
posTextObj1 = (30, int(0.10*height)) # 10%
posTextObjA = (30, int(0.20*height)) # 20%
runGoodArray1 = np.zeros(dimensionFrameRecord)
runGoodArrayA = np.zeros(dimensionFrameRecord)
Obj1_Total, Obj1_Adder = 0, 0
ObjA_Total, ObjA_Adder = 0, 0

while(cap.isOpened()):
    ret, image2C = cap.read()
    if ret == True:
        for iShift in range(dimensionFrameRecord-1):
            runGoodArray1[iShift] = runGoodArray1[iShift+1]
            runGoodArrayA[iShift] = runGoodArrayA[iShift+1]

        image2G = cv2.cvtColor(image2C, cv2.IMREAD_GRAYSCALE)
        kp2, des2 = sift.detectAndCompute(image2G, None)
        #-----
        good = []
        if len(kp2) != 0 :
            match = cv2.BFMatcher()
            matches = match.knnMatch(des1, des2, k=2)
            for i_matche in range(len(matches)):
                try:
                    m, n = matches[i_matche]
                except (ValueError):
                    pass
                else:
                    if m.distance < 0.5 * n.distance :
                        good.append(m)

            if len(good) >= minGoodPoint_1:
                runGoodArray1[dimensionFrameRecord-1] = 1
            else:
                runGoodArray1[dimensionFrameRecord-1] = 0

        summFrame = runGoodArray1.sum(dtype=np.int32)
        if summFrame <= minFrame_1Set:

```

```

    Obj1_Adder = 1
    if summFrame >= maxFrame_1Add:
        Obj1_Total = Obj1_Total + Obj1_Adder
        Obj1_Adder = 0

    textShow = str(Obj1_Total) + "<" + str(summFrame) + "," + str(len(good)) + ">"
    cv2.putText(image2C, textShow, posTextObj1, cv2.FONT_HERSHEY_PLAIN, 4, (0,0,255), 3)

    #-----
    if len(kp2) >= minGoodPoint_A:
        runGoodArrayA[dimensionFrameRecord-1] = 1
    else:
        runGoodArrayA[dimensionFrameRecord-1] = 0

    summFrame = runGoodArrayA.sum(dtype=np.int32)
    if summFrame <= minFrame_ASet:
        ObjA_Adder = 1
    if summFrame >= maxFrame_AAdd:
        ObjA_Total = ObjA_Total + ObjA_Adder
        ObjA_Adder = 0

    textShow = str(ObjA_Total) + "<" + str(summFrame) + "," + str(len(kp2)) + ">"
    cv2.putText(image2C, textShow, posTextObjA, cv2.FONT_HERSHEY_PLAIN, 4, (255,0,0), 3)

    #-----
    match_result = cv2.drawMatches(image1C, kp1, image2C, kp2, good[:50], None, flags=2)
    cv2.imshow('Frame',match_result)
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break
    else:
        break

cap.release()
cv2.destroyAllWindows()
print("Total = ",ObjA_Total ,", Object Type_1 = ", Obj1_Total)

```


Code Capture

```

In [ ]: 1 # Step 30 - Keypoint Mactching on Video
2 import cv2
3 import numpy as np
4 dimensionFrameRecord = 20
5 minGoodPoint_A, maxFrame_AAdd, minFrame_ASet = 10, 14, 13
6 minGoodPoint_1, maxFrame_1Add, minFrame_1Set = 4, 10, 2
7
8 sift = cv2.xfeatures2d.SIFT_create()
9 #image1C = cv2.imread("./image/Skittles.jpg")
10 image1C = cv2.imread("D:\\Machine\\Week11\\im_me\\B.jpg")
11 image1G = cv2.cvtColor(image1C, cv2.IMREAD_GRAYSCALE)
12 kp1, des1 = sift.detectAndCompute(image1G, None)
13
14 #cap = cv2.VideoCapture("./image/Candy.mp4")
15 #cap = cv2.VideoCapture("./image/video_BU2.avi")
16 #cap = cv2.VideoCapture("./image/video_BU2S.avi")
17 #cap = cv2.VideoCapture("./image/video_BU2F.avi")
18 #cap = cv2.VideoCapture("D:\\Machine\\Week9\\video_B_new7.mp4")
19 cap = cv2.VideoCapture("D:\\Machine\\Week9\\video_BU1_gg_13.avi")
20
21 fourcc = cv2.VideoWriter_fourcc(*'MP4V')
22 frame_width = int(cap.get(cv2.CAP_PROP_FRAME_WIDTH))
23 frame_height = int(cap.get(cv2.CAP_PROP_FRAME_HEIGHT))
24 out = cv2.VideoWriter("./im_m1/B1_Weeek10.avi", fourcc, 30, (frame_width, frame_height))
25
26 ret, frame = cap.read()
27 height, width = frame.shape[:2]
28 posTextObj1 = (30, int(0.10*height)) # 10%
29 posTextObjA = (30, int(0.20*height)) # 20%
30 runGoodArray1 = np.zeros(dimensionFrameRecord)
31 runGoodArrayA = np.zeros(dimensionFrameRecord)
32 Obj1_Total, Obj1_Adder = 0, 0
33 ObjA_Total, ObjA_Adder = 0, 0
34
35 while(cap.isOpened()):
36     ret, image2C = cap.read()
37     if ret == True:
38         for ishift in range(dimensionFrameRecord-1):
39             runGoodArray1[ishift] = runGoodArray1[ishift+1]
40             runGoodArrayA[ishift] = runGoodArrayA[ishift+1]
41
42         image2G = cv2.cvtColor(image2C, cv2.IMREAD_GRAYSCALE)
43         kp2, des2 = sift.detectAndCompute(image2G, None)
44         #-----
45         good = []
46         if len(kp2) != 0 :
47             match = cv2.BFMatcher()
48             matches = match.knnMatch(des1, des2, k=2)
49             for i_matche in range(len(matches)):
50                 try:
51                     m, n = matches[i_matche]
52                 except (ValueError):
53                     pass
54                 else:
55                     if m.distance < 0.5 * n.distance :
56                         good.append(m)
57
58             if len(good) >= minGoodPoint_1:
59                 runGoodArray1[dimensionFrameRecord-1] = 1
60             else:
61                 runGoodArray1[dimensionFrameRecord-1] = 0
62
63             summFrame = runGoodArray1.sum(dtype=np.int32)
64             if summFrame <= minFrame_1Set:
65                 Obj1_Adder = 1
66             if summFrame >= maxFrame_1Add:
67                 Obj1_Total = Obj1_Total + Obj1_Adder
68                 Obj1_Adder = 0
69
70             textShow = str(Obj1_Total) + "<" + str(summFrame) + "," + str(len(good)) + ">"
71             cv2.putText(image2C, textShow, posTextObj1, cv2.FONT_HERSHEY_PLAIN, 4, (0,0,255), 3)
72
73             #-----
74             if len(kp2) >= minGoodPoint_A:
75                 runGoodArrayA[dimensionFrameRecord-1] = 1
76             else:
77                 runGoodArrayA[dimensionFrameRecord-1] = 0
78
79             summFrame = runGoodArrayA.sum(dtype=np.int32)
80             if summFrame <= minFrame_ASet:
81                 ObjA_Adder = 1
82             if summFrame >= maxFrame_AAdd:
83                 ObjA_Total = ObjA_Total + ObjA_Adder
84                 ObjA_Adder = 0

```

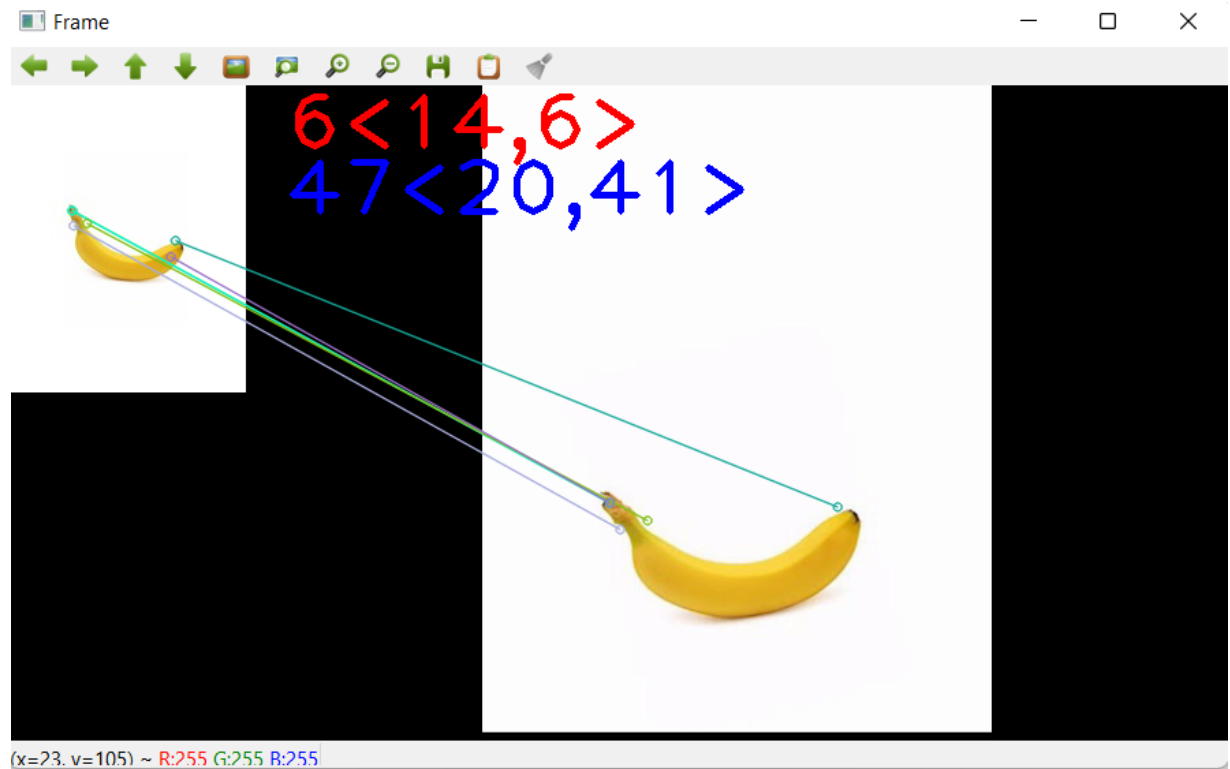
```

85
86     textShow = str(ObjA_Total) + "<" + str(summFrame) + "," + str(len(kp2)) + ">"
87     cv2.putText(image2C, textShow, posTextObjA, cv2.FONT_HERSHEY_PLAIN, 4, (255,0,0), 3)
88
89     #-----
90     match_result = cv2.drawMatches(image1C, kp1, image2C, kp2, good[:50], None, flags=2)
91     cv2.imshow('Frame', match_result)
92     if cv2.waitKey(1) & 0xFF == ord('q'):
93         break
94     else:
95         break
96
97 cap.release()
98 cv2.destroyAllWindows()
99 print("Total = ", ObjA_Total, ", Object Type_1 = ", Obj1_Total)

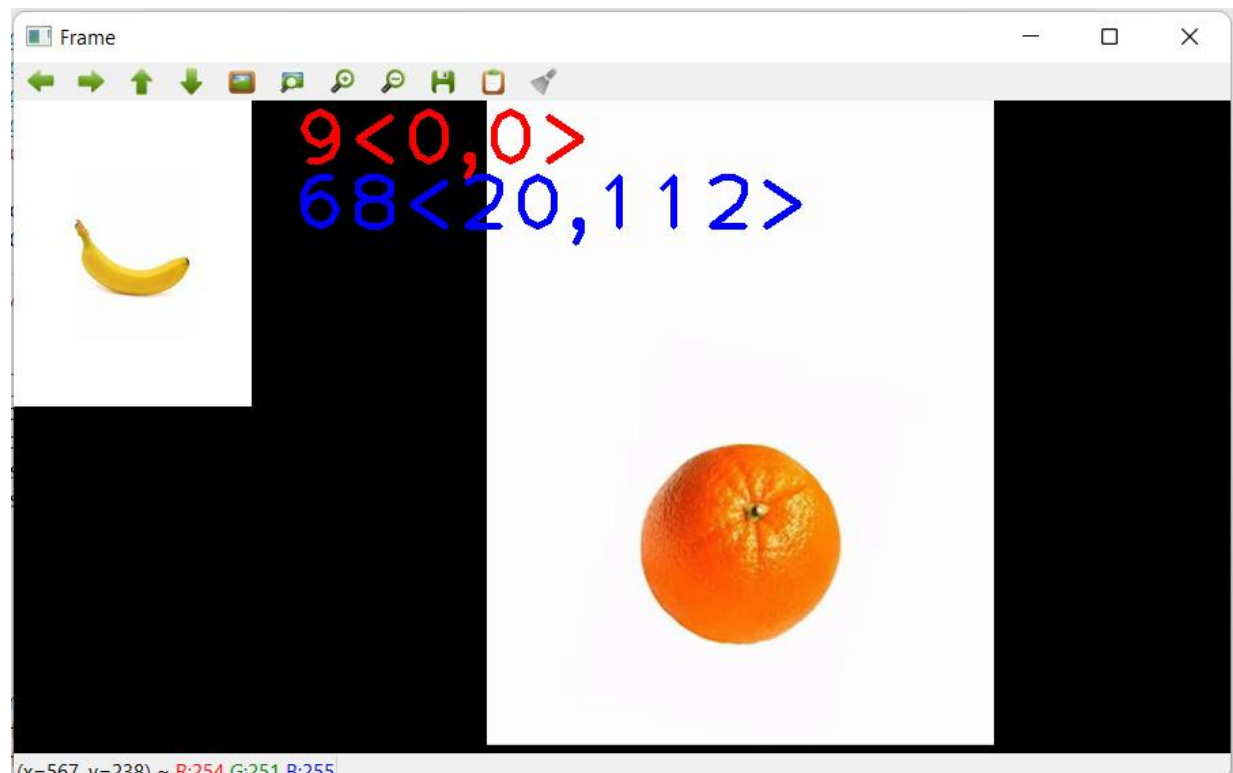
```

Total = 80 , Object Type_1 = 10

Capture Result-1



Capture Result-2



Capture Result-3

