電腦視覺與深度學習
(Computer Vision and Deep Learning)
Homework 2

TA:

家銘: nckubot65904@gmail.com

Office Hour: 17:00~19:00, Mon.
10:00~12:00, Fri.

At CSIE 9F Robotics Lab.

Notices (1/2)

- ❑ Copying homework is strictly prohibited!! Penalty: Grade will be zero for both persons!!
- ❑ If the code can’t run, you can come to our Lab within one week and show that your programming can work. Otherwise you will get zero!!
- ❑ Due date => Midnight 23:59:59 on 2021/12/22 (Wed.)
  - No delay. If you submit homework after deadline, you will get 0.
- ❑ Upload to => 140.116.154.1 -> Upload/Homework/Hw2
  - User ID: cvdl2021   Password: cvdl2021
- ❑ Format
  - Filename: Hw2_StudentID_Name_Version.rar
    - Ex: Hw2_F71234567_林小明_V1.rar
    - If you want to update your file, you should update your version to be V2, ex: Hw2_F71234567_林小明_V2.rar
  - Content: project folder*(including the pictures)
    *note: remove your “Debug” folder to reduce file size

Notices (2/2)

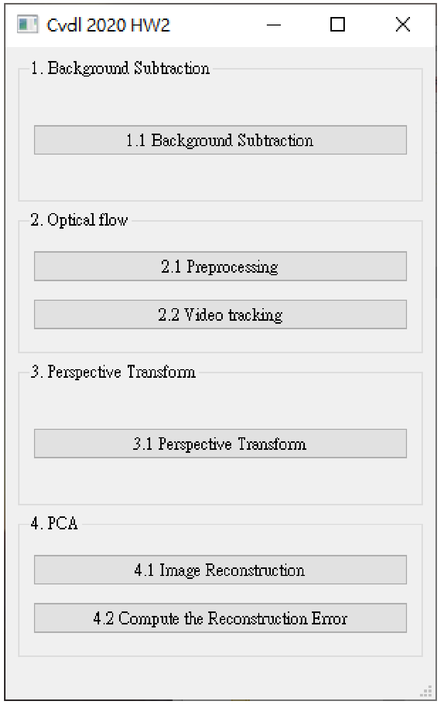
- Python
  - Python 3.7 (https://www.python.org/downloads/)
  - opencv-contrib-python (3.4.2.17)
  - Matplotlib 3.1.1
  - UI framework: pyqt5 (5.15.1)

- C++ (check MFC guide in ftp)
  - OpenCV 3.3.1 (https://opencv.org/release.html)
  - Visual Studio 2015 (download from http://www.cc.ncku.edu.tw/download/)
  - UI framework: MFC

Grading

1. (20%) Background Subtraction (出題：Tony)
2. (20%) Optical Flow (出題：David)
   - 2.1 Preprocessing (10%)
   - 2.2 Video tracking (10%)
3. (20%) Perspective Transform (出題：West)
4. (20%) PCA (出題：Willy)
   - 4.1 Image Reconstruction (10%)
   - 4.2 Compute the reconstruction error (10%)
5. (20%) Dogs and Cats classification Using ResNet50 (出題：育成)

UI example



1. (20%) Background Subtraction: Simple Gaussian Model (1/2)

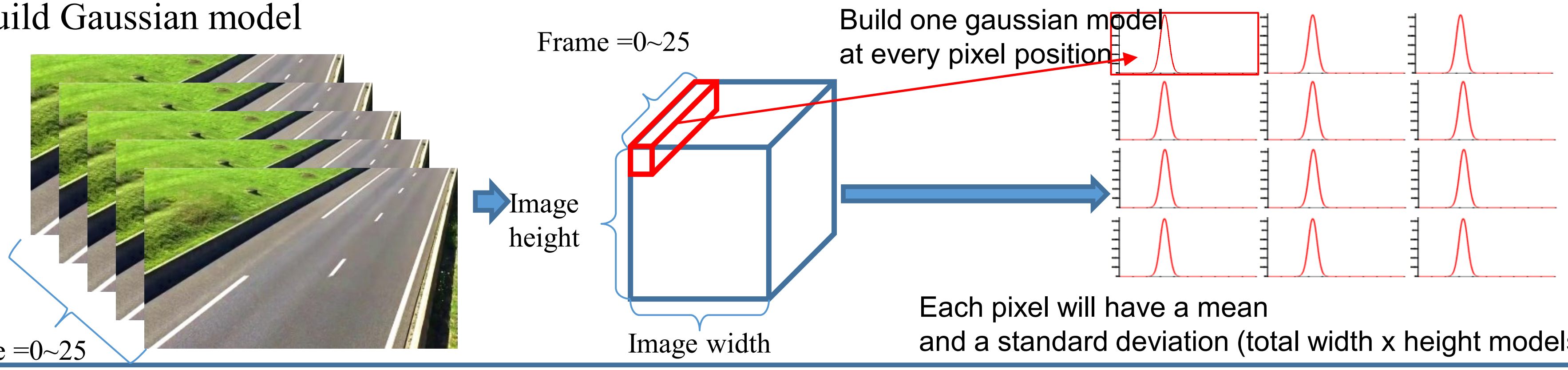
(出題：Tony)

- ❑ Given: a video in Q1_Image/Traffic.mp4
- ❑ Requires: 1. Click the button “1.1 Background Subtraction” to show 3 windows:
  - Origin video: Traffic.mp4
  - Mask video: (background: black, foreground: white)
  - Foreground video

  2. Use first 25 frames to build the background model

  3. DO NOT use OpenCV function: cv2.createbackgroundSubtractor()

1. Build Gaussian model



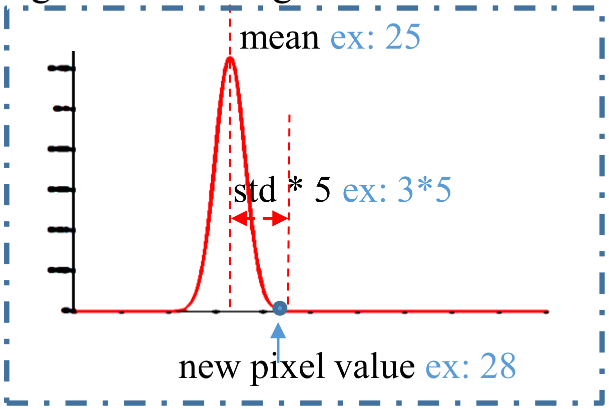
Build one gaussian model at every pixel position

Each pixel will have a mean and a standard deviation (total width x height models)

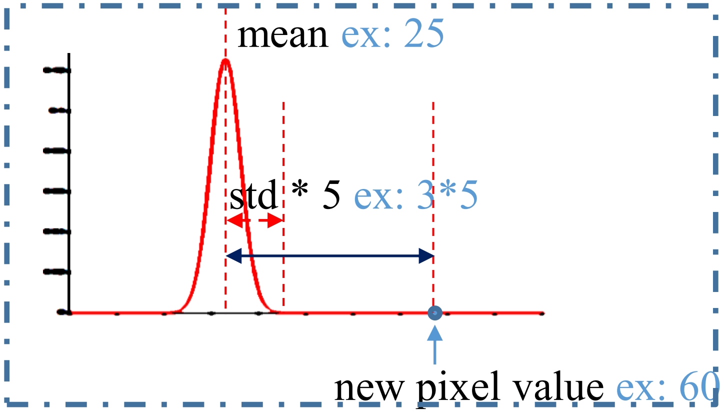
Output: w x h gaussian models (mean and std)

2. Test new pixel belongs to foreground or background:

Ex:
Mean = 25
Std (standard deviation) = 3



Belongs to Background



Belongs to Foreground

(出題：Tony)

1. (20%) Background Subtraction: Simple Gaussian Model (2/2)

- Hint:
  1. Convert video frames to Gray (cv2.cvtColor)
  2. For every pixels in video from 0~25 frames, build a gaussian model with mean and standard deviation (if standard deviation is less then 5, set to 5)
  3. For frame > 25, test every frame pixels with respective gaussian model. If gray value difference between testing pixel and gaussian mean is larger than 5 times standard deviation, set testing pixel to 255 (foreground, white), 0 (background, black) otherwise.
  4. Remove background (cv2.bitwise_and)
- Demo video:

2. (20%) Optical Flow

(出題：David)

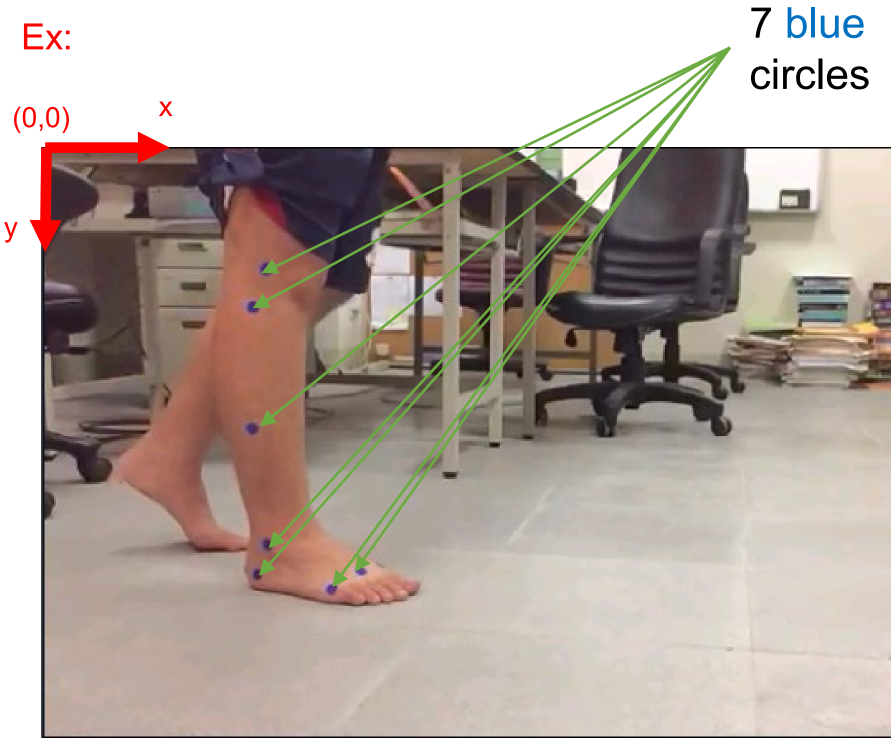
2.1 Preprocessing (10%)

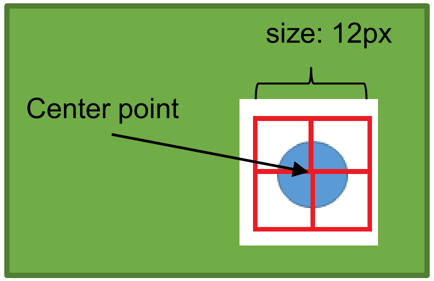
2.2 Video tracking (10%)

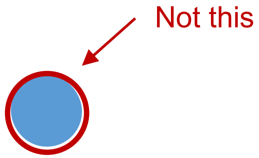
2.1 Preprocessing (10%)

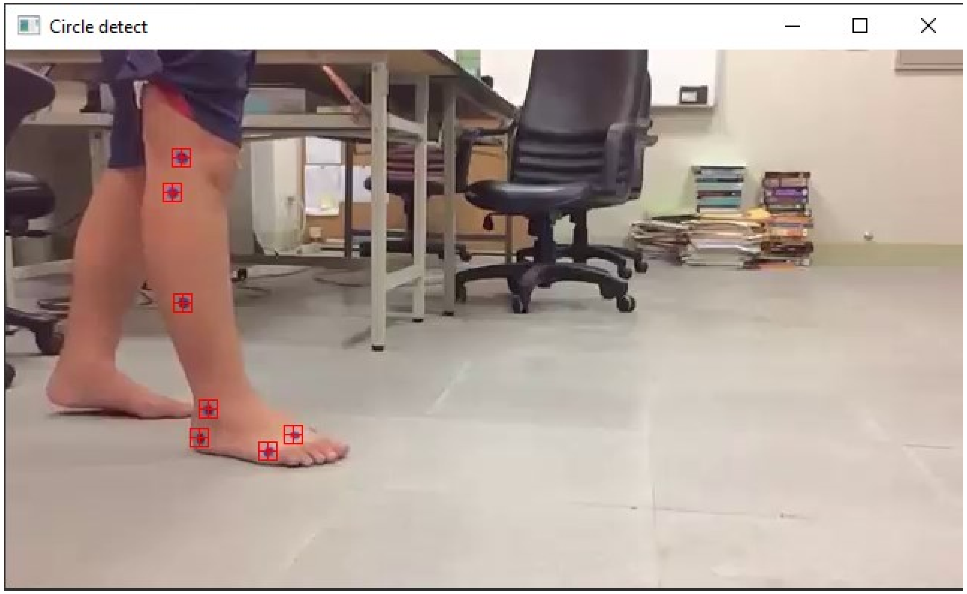
(出題：David)

- ❑ Given a video: optical_flow.mp4
- ❑ Q: Detect the 7 blue circles and display image with them marked with a red square bounding box and cross mark.









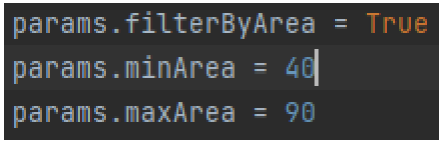
※ Tip: Draw your own marker with opencv functions cv.rectangle() and cv.line()

❑ How to detect center points of 7 blue circles (出題：David)

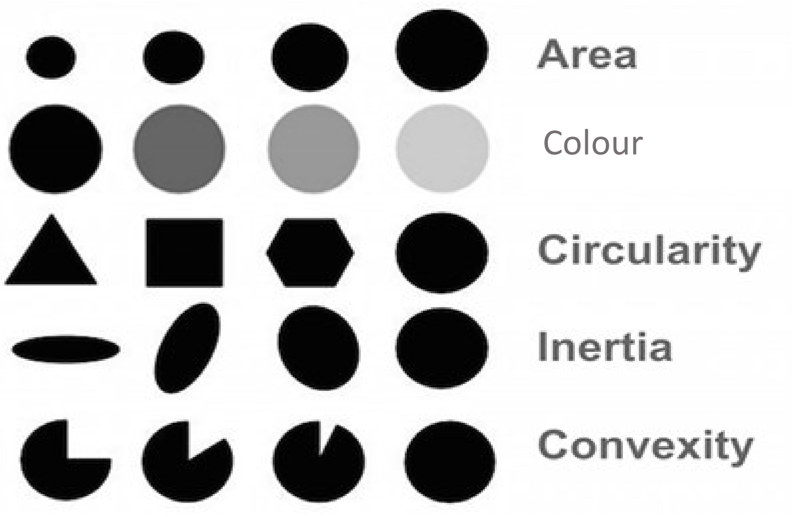
❑ You can use `cv2.SimpleBlobDetector()` to detect blobs (binary large object), it can filter out blobs by different properties:

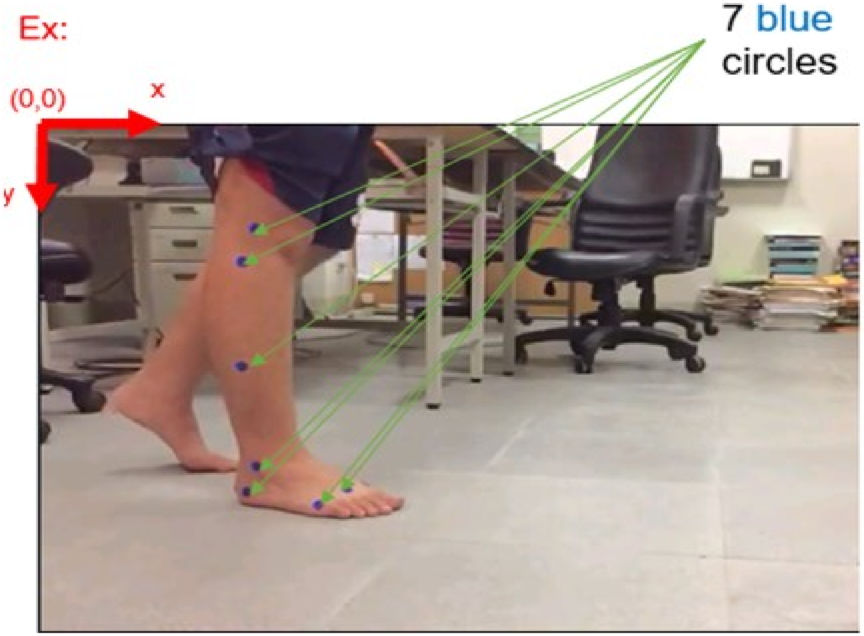
1) Area (of blob in pixels, e.g. 100 px)
2) Color (gray level, 0 for black, 255 for white)
3) Circularity (if perfect circle, then it is 1.0)
4) Inertia (ratio of the minor and major axes from 0.0 to 1.0)
5) Convexity (if no gap, then it is 1.0)

❑ Minimum and maximum bounds can be chosen. E.g. ->



```
params.filterByArea = True
params.minArea = 40
params.maxArea = 90
```





2.2 Video tracking (10%)

(出題：David)

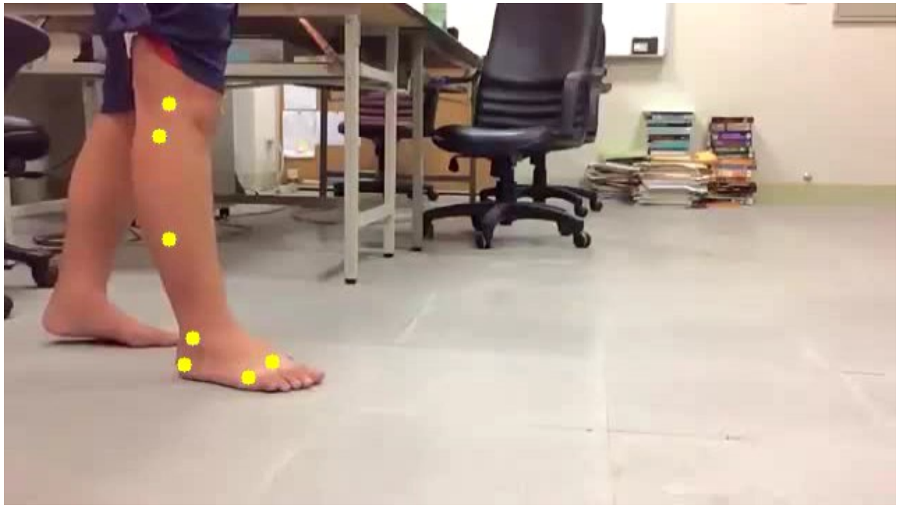
□ Q: Click button to:

1) (5%) Track the 7 center points on the whole video using OpenCv function `cv2.calcOpticalFlowPyrLK`.
   Ex: https://github.com/opencv/opencv/blob/master/samples/cpp/lkdemo.cpp
2) (5%) Display the trajectory of each of the 7 tracking points throughout the video. Pick a color that is highly visible. Ex: the demo videos bellow.

□ Tool site:

1. Load video
2. Circle detect
3. Optical flow

Demo videos:



※ Tracking process may have some errors. At least 4-5 points should be tracked with success.

3. (20%) Perspective Transform:

(出題：West)

- ❑ Given a video: test4perspective.mp4
- ❑ Q: 1) Click the button “ 3.1 Perspective Transform” to show result video:
- ❑ Hint : (1) Read the test video and process frame by frame

  (2) Detect the ArUco maker inside the frames

  (3) Get the pixel position (x, y) of four makers in the frame

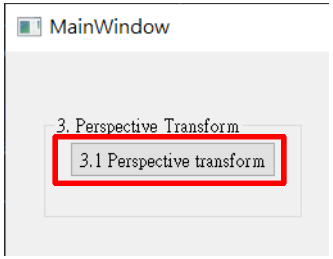
  (4) Do perspective transform

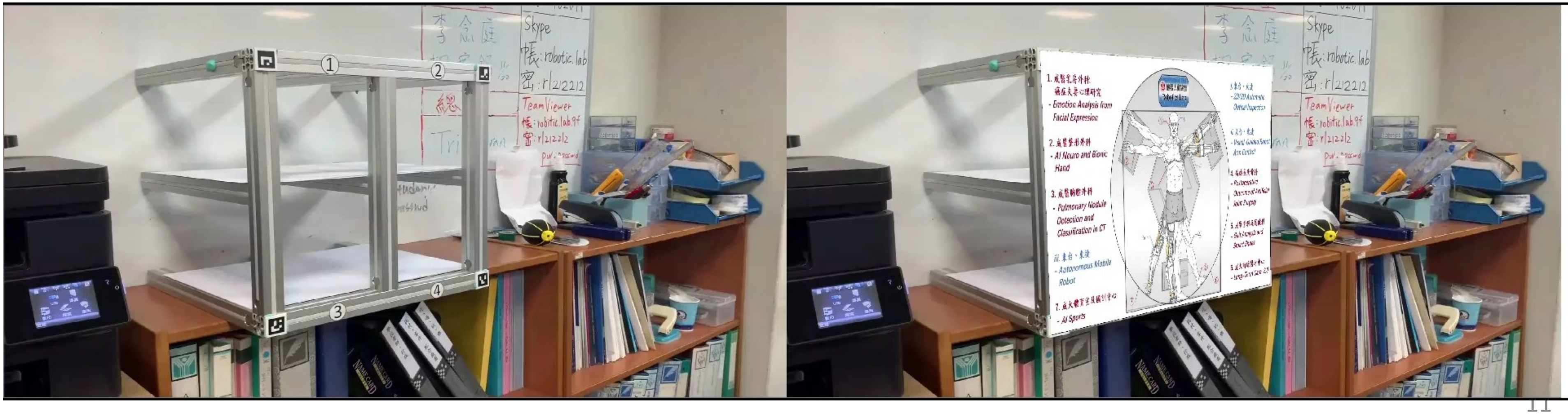
  (5) Show result video

Next page will show how to detect ArUco Maker

Please do perspective transform using OpenCV method

- ❑ Ex:

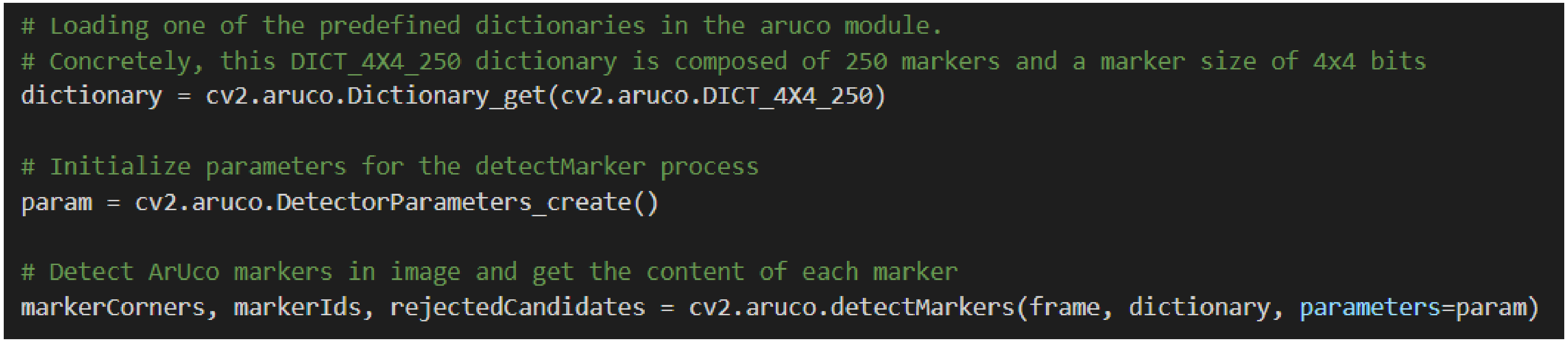




❑ Hints: How to use ArUco Makers (1/2)

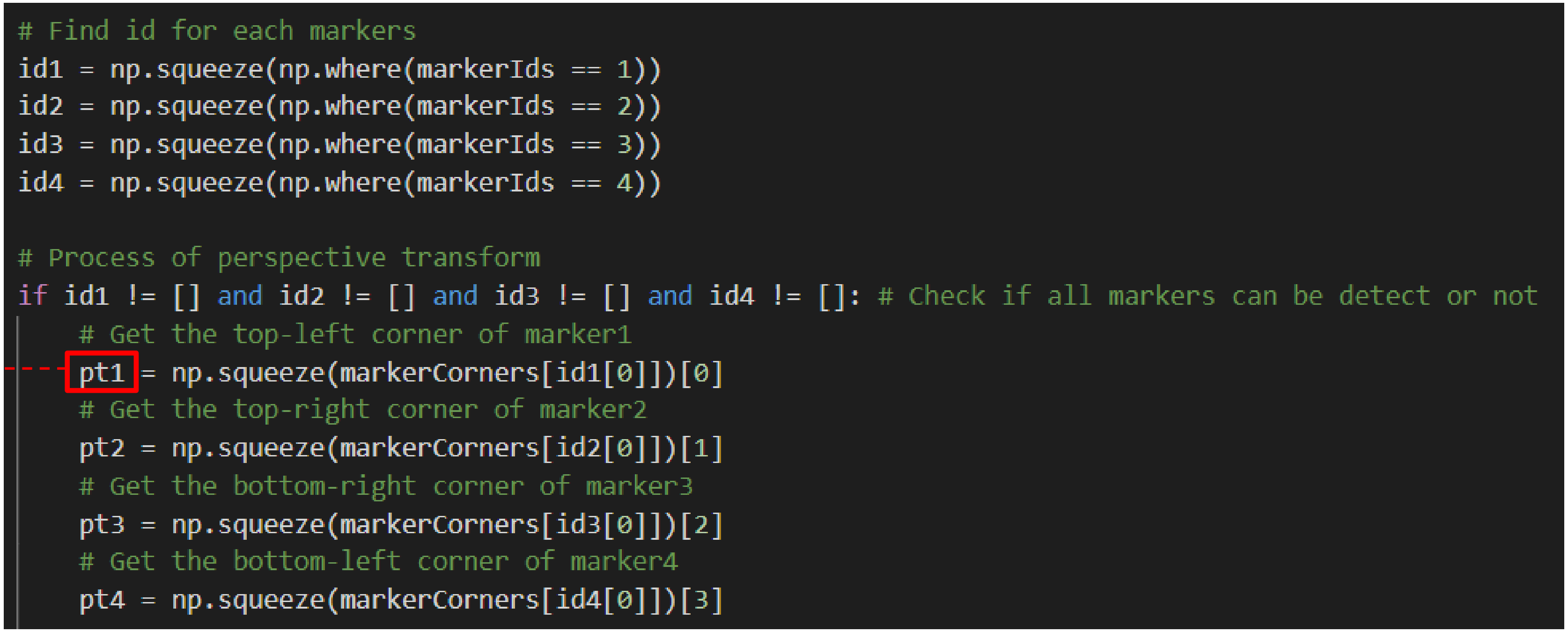
(出題：West)

Step 1: Detect ArUco markers in image.



```
# Loading one of the predefined dictionaries in the aruco module.
# Concretely, this DICT_4X4_250 dictionary is composed of 250 markers and a marker size of 4x4 bits
dictionary = cv2.aruco.Dictionary_get(cv2.aruco.DICT_4X4_250)

# Initialize parameters for the detectMarker process
param = cv2.aruco.DetectorParameters_create()

# Detect ArUco markers in image and get the content of each marker
markerCorners, markerIds, rejectedCandidates = cv2.aruco.detectMarkers(frame, dictionary, parameters=param)
```

Step 2: Get corner position of each marker.



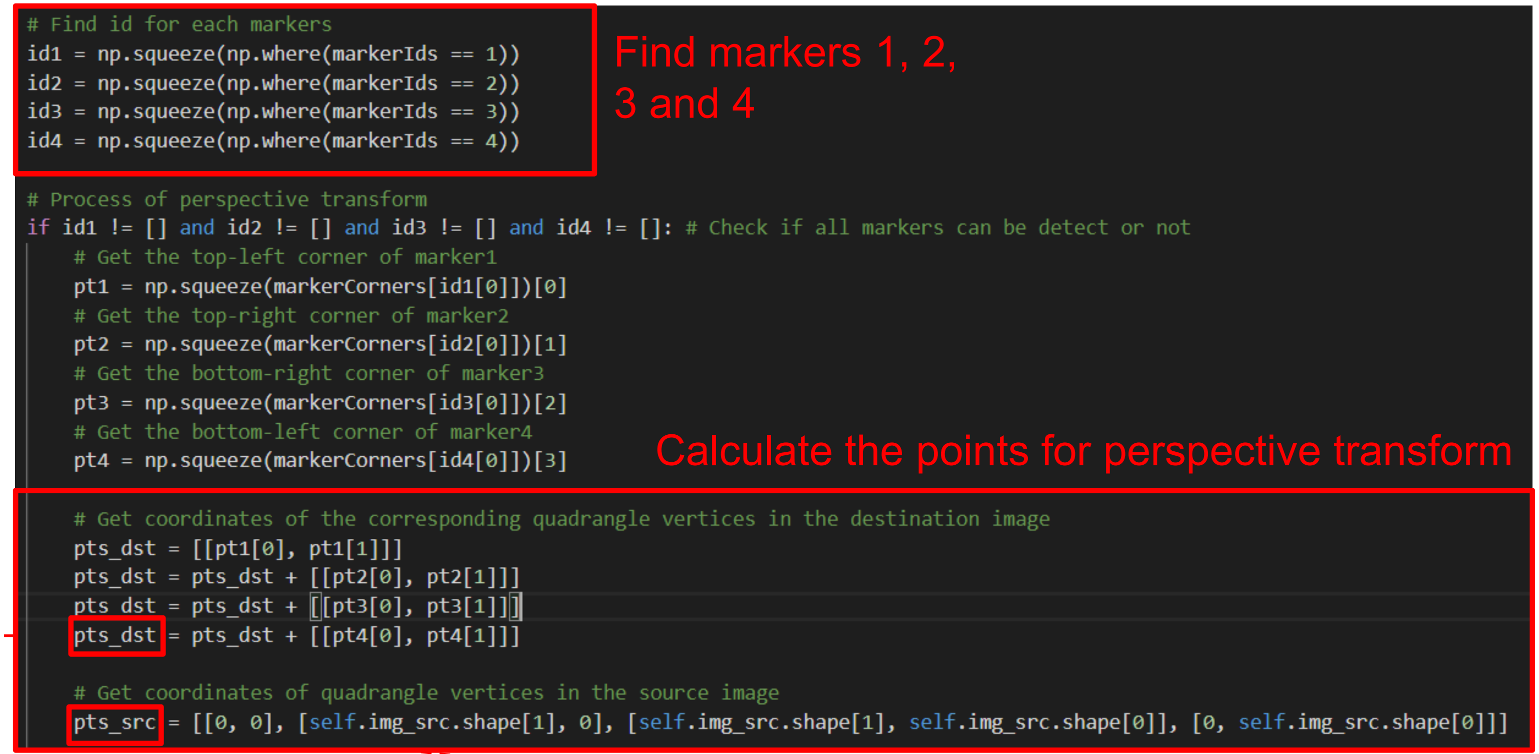
```
# Find id for each markers
id1 = np.squeeze(np.where(markerIds == 1))
id2 = np.squeeze(np.where(markerIds == 2))
id3 = np.squeeze(np.where(markerIds == 3))
id4 = np.squeeze(np.where(markerIds == 4))

# Process of perspective transform
if id1 != [] and id2 != [] and id3 != [] and id4 != []: # Check if all markers can be detect or not
    # Get the top-left corner of marker1
    pt1 = np.squeeze(markerCorners[id1[0]])[0]
    # Get the top-right corner of marker2
    pt2 = np.squeeze(markerCorners[id2[0]])[1]
    # Get the bottom-right corner of marker3
    pt3 = np.squeeze(markerCorners[id3[0]])[2]
    # Get the bottom-left corner of marker4
    pt4 = np.squeeze(markerCorners[id4[0]])[3]
```

Get corner position (x, y) of makers which's ID is 1

❑ Hints: How to use ArUco Makers (2/2)

(出題：West)

Step 3: Then calculate the four edges of the region you want to perspective

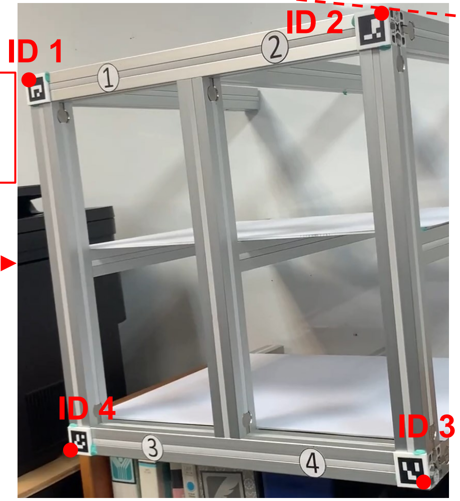


```
# Find id for each markers
id1 = np.squeeze(np.where(markerIds == 1))
id2 = np.squeeze(np.where(markerIds == 2))
id3 = np.squeeze(np.where(markerIds == 3))
id4 = np.squeeze(np.where(markerIds == 4))

# Process of perspective transform
if id1 != [] and id2 != [] and id3 != [] and id4 != []: # Check if all markers can be detect or not
    # Get the top-left corner of marker1
    pt1 = np.squeeze(markerCorners[id1[0]])[0]
    # Get the top-right corner of marker2
    pt2 = np.squeeze(markerCorners[id2[0]])[1]
    # Get the bottom-right corner of marker3
    pt3 = np.squeeze(markerCorners[id3[0]])[2]
    # Get the bottom-left corner of marker4
    pt4 = np.squeeze(markerCorners[id4[0]])[3]

    # Get coordinates of the corresponding quadrangle vertices in the destination image
    pts_dst = [[pt1[0], pt1[1]]]
    pts_dst = pts_dst + [[pt2[0], pt2[1]]]
    pts_dst = pts_dst + [[pt3[0], pt3[1]]]
    pts_dst = pts_dst + [[pt4[0], pt4[1]]]

    # Get coordinates of quadrangle vertices in the source image
    pts_src = [[0, 0], [self.img_src.shape[1], 0], [self.img_src.shape[1], self.img_src.shape[0]], [0, self.img_src.shape[0]]]
```

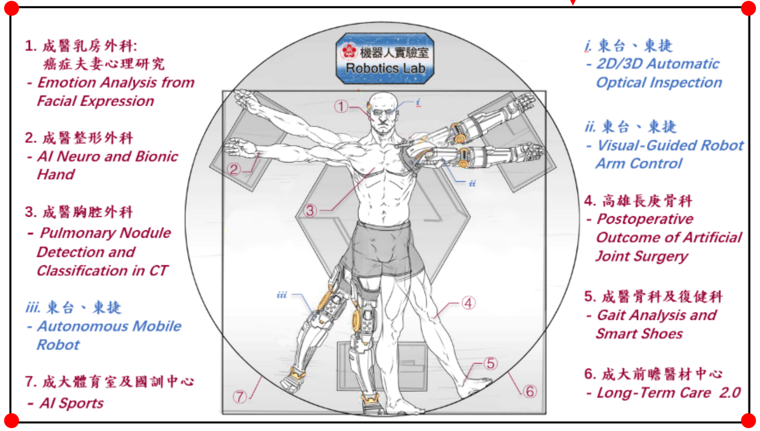
Find markers 1, 2, 3 and 4

Calculate the points for perspective transform

pts_dst are 4 marker points



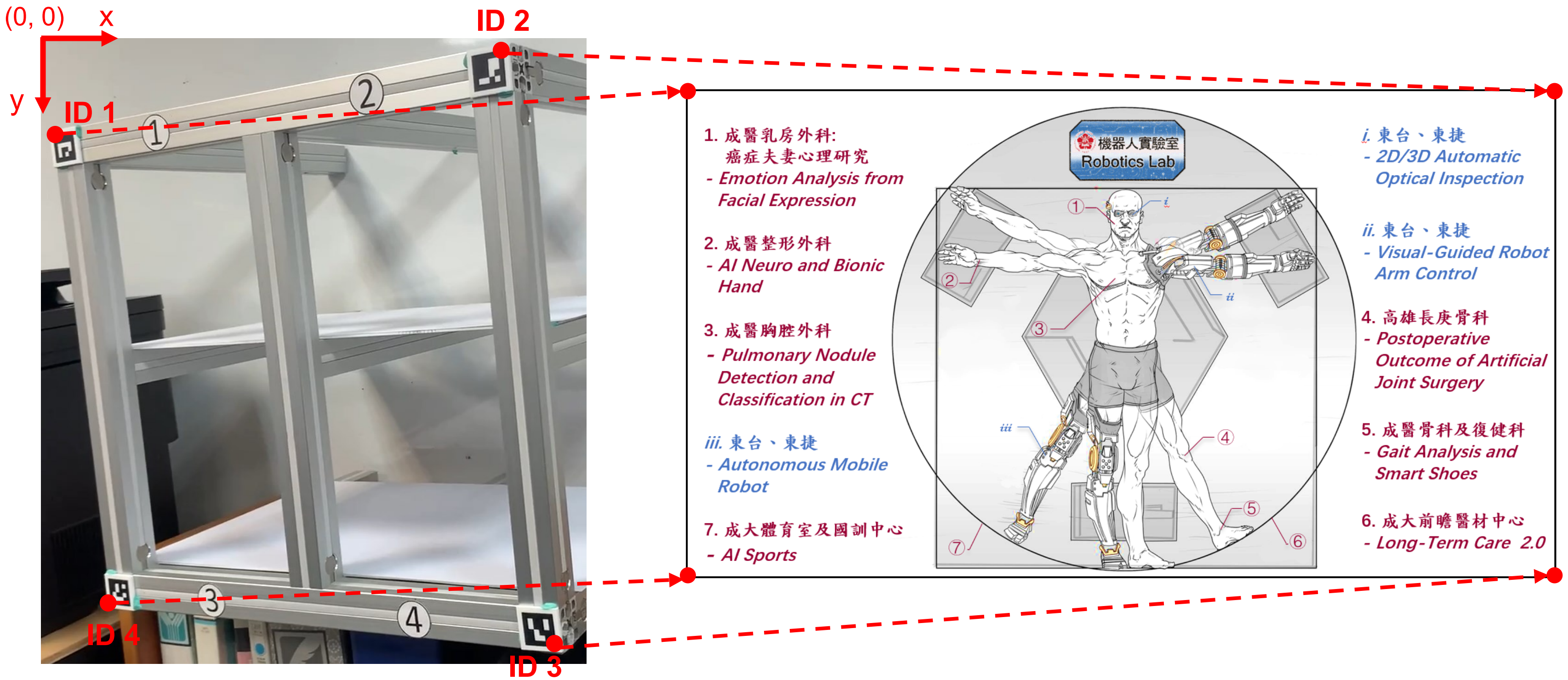
pts_dst are 4 image points


1. 成醫乳房外科:
癌症夫妻心理研究
- Emotion Analysis from Facial Expression
2. 成醫整形外科
- AI Neuro and Bionic Hand
3. 成醫胸腔外科
- Pulmonary Nodule Detection and Classification in CT
iii. 東台、東捷
- Autonomous Mobile Robot
7. 成大體育室及國訓中心
- AI Sports
機器人實驗室
Robotics Lab
i. 東台、東捷
- 2D/3D Automatic Optical Inspection
ii. 東台、東捷
- Visual-Guided Robot Arm Control
4. 高雄長庚骨科
- Postoperative Outcome of Artificial Joint Surgery
5. 成醫骨科及復健科
- Gait Analysis and Smart Shoes
6. 成大前瞻醫材中心
- Long-Term Care 2.0

3. (20%) Perspective Transform:

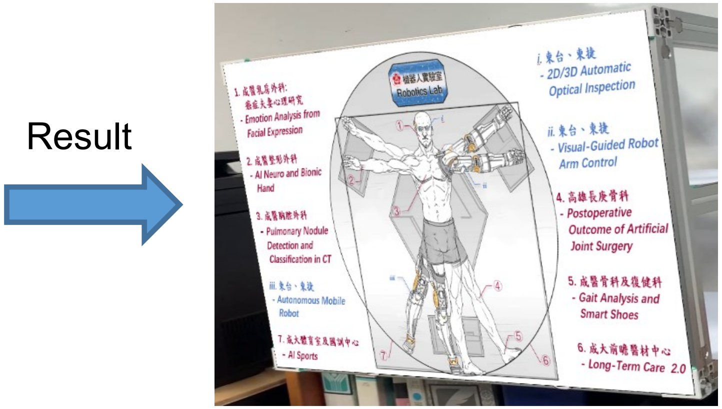
(出題：West)

- ❑ Do the perspective transform
- ❑ Hint:
  - ▪ cv.findHomography() and cv.warpPerspective()



(0, 0)
x
y
ID 1
ID 2
ID 3
ID 4
1
2
3
4
1. 成醫乳房外科:
癌症夫妻心理研究
- Emotion Analysis from Facial Expression
2. 成醫整形外科
- AI Neuro and Bionic Hand
3. 成醫胸腔外科
- Pulmonary Nodule Detection and Classification in CT
iii. 東台、東捷
- Autonomous Mobile Robot
7. 成大體育室及國訓中心
- AI Sports
機器人實驗室
Robotics Lab
i. 東台、東捷
- 2D/3D Automatic Optical Inspection
ii. 東台、東捷
- Visual-Guided Robot Arm Control
4. 高雄長庚骨科
- Postoperative Outcome of Artificial Joint Surgery
5. 成醫骨科及復健科
- Gait Analysis and Smart Shoes
6. 成大前瞻醫材中心
- Long-Term Care 2.0

Result



4. (20%) PCA

(出題：Willy)

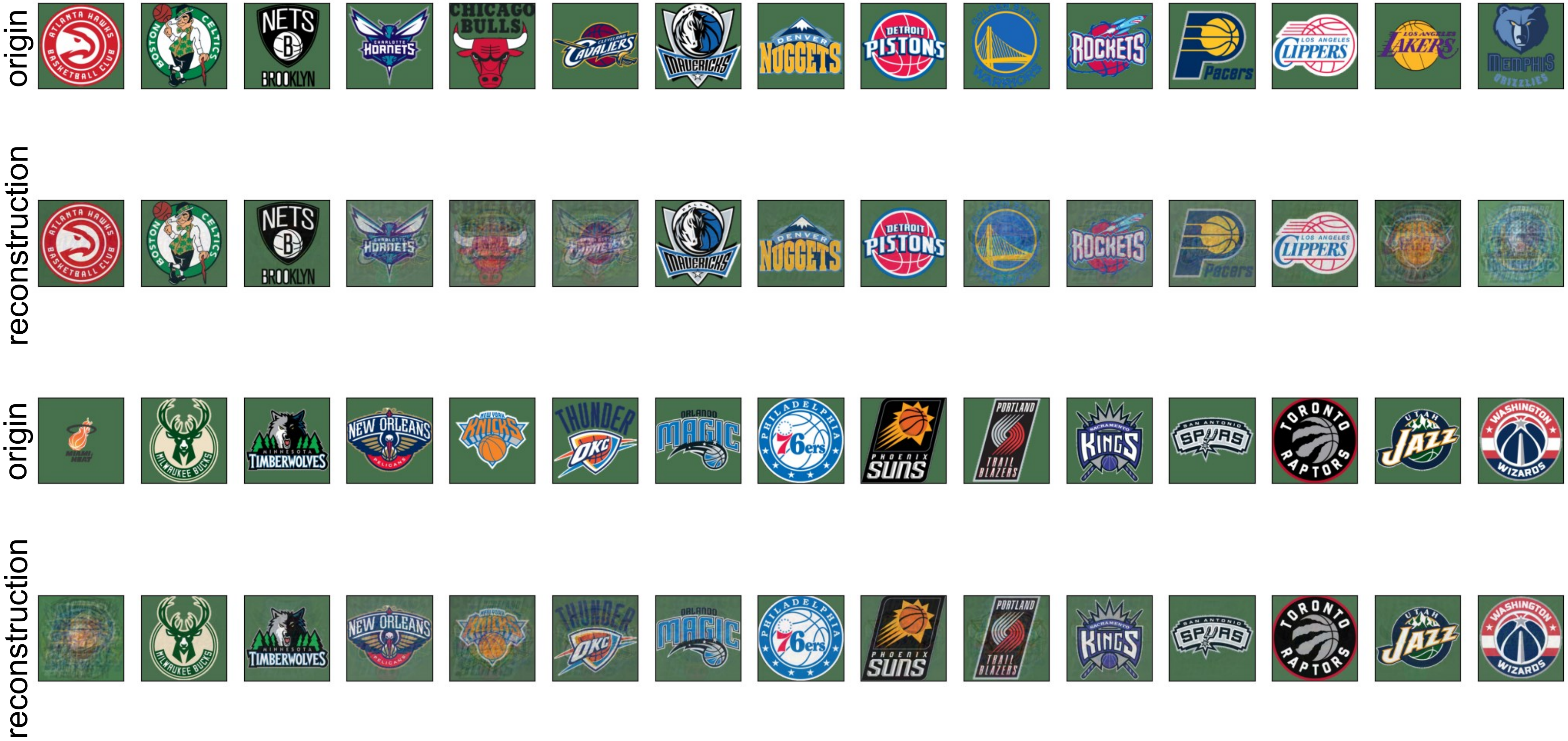
4.1 Image Reconstruction (10%)

4.2 Compute the reconstruction error (10%)

4.1 (10%) PCA - Image Reconstruction (1/2)

(出題：Willy)

- □ Given: Images of NBA team logo
- □ Q1: Using PCA(Principal components analysis) to do dimension reduction and then reconstruct, click button “4.1 Image Reconstruction” and show original and reconstructed images.
- □ Hint: Use PCA from python library `sklearn.decomposition`
  `Use matplotlib to show all images at once`



4.2 (10%) PCA - Compute the Reconstruction Error (2/2)

(出題：Willy)

- ❑ Given: Images of NBA team logo
- ❑ Q2: Computing the reconstruction error (RE) for each logo, click button "4.2 Compute the reconstruction error" and then print out on the console.
- ❑ EX: [4214708, 27793154, 26582084, 30843153, 32780303, 30726368, 27606505, 19924229, 30379090, 31211554, 31149559, 30267885, 22194939, 13194457, 30980966, 26039009, 26227989, 32877982, 31474702, 15196576, 18912022, 31835902, 24525323, 29398470, 24614317, 31329428, 28970970, 33122038, 29224809, 27085982]

- ❑ Using the reference formula as shown below:

1. Using PCA to reconstruct image
2. Normalize reconstruct image to [0, 255]
3. Convert original image and reconstruction image to gray scale
4. Calculate reconstruction error using below formula

$$RE = \sum_{i=1}^{n_{pixels}} ||\vec{x_i} - \vec{y_i}||\, , where$$

$\vec{x_i}$ *is the gray value of pixel in original image*

$\vec{y_i}$ *is the gray value of pixel in reconstruction image*

5.0 (20%)Dogs and Cats classification Using ResNet50 (出題：育成)

1) Dataset introduction:

(1) ASIRRA (Animal Species Image Recognition for Restricting Access) is a HIP(Human Interactive Proof) that works by asking users to identify photographs of cats and dogs, that's supposed to be easy for people to solve, but difficult for computers. Now we can use artificial intelligence technology to achieve this goal.

(2) The dataset includes 12501 photos of cats and 12501 photos of dogs. You need to download them in Reference below(R2), and split the training set, validation set and test set using 8:1:1 ratio in both dogs and cats directory.

2) Objective:

- You need to use python to write the ResNet network and complete the questions on the next few pages.

3) Environment Requirement

(1) Python

(2) Tensorflow

(3) Opencv-contrib-python

(4) Matplotlib

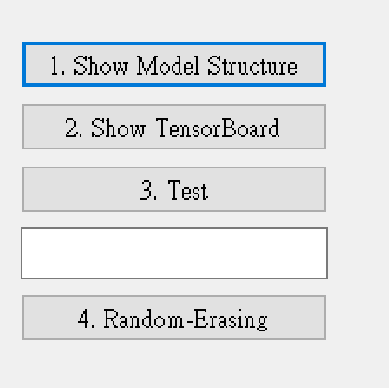
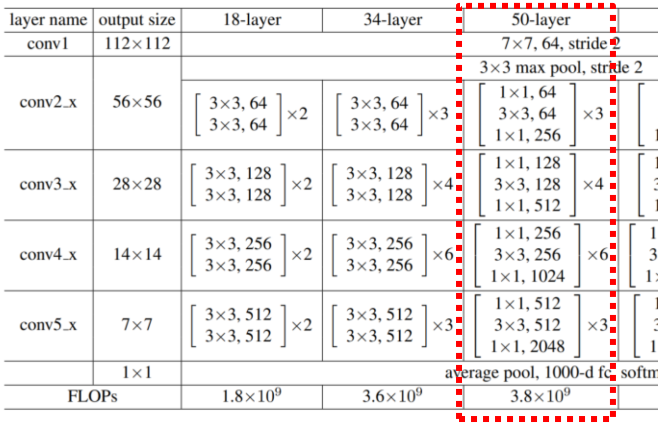
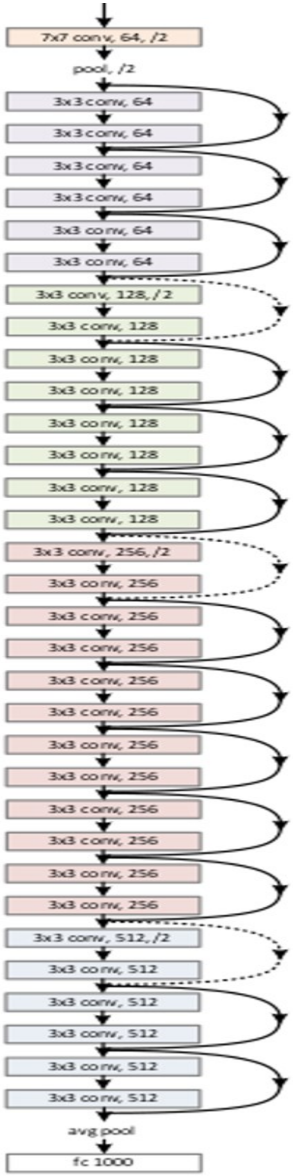


Fig. *Example UI*



layer name	output size	18-layer	34-layer	50-layer
conv1	112×112	7×7, 64, stride 2		
conv2_x	56×56	3×3 max pool, stride 2		
		$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$
conv3_x	28×28	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$
conv4_x	14×14	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times6$
conv5_x	7×7	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$
	1×1	average pool, 1000-d fc, softm		
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9

Fig. *ResNet's Network Architecture*



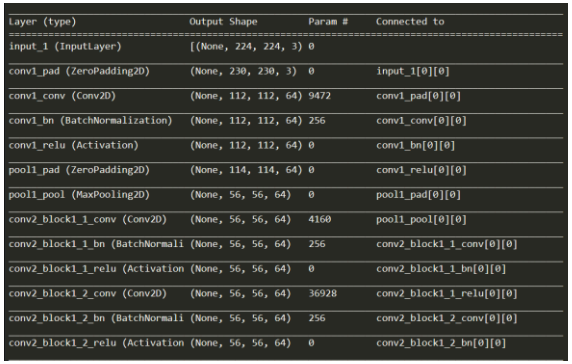
R. Reference

R1) Deep Residual Learning for Image Recognition

R2) https://www.microsoft.com/en-us/download/details.aspx?id=54765 (ASIRRA)

Fig. *ResNet50's Schematic Diagram*

5.1 Construct and show summary of your model structure by printing out on the terminal. **(5%)**



```
________________________________________________________________________________________________
Layer (type)                    Output Shape         Param #     Connected to
================================================================================================
input_1 (InputLayer)            [(None, 224, 224, 3) 0
________________________________________________________________________________________________
conv1_pad (ZeroPadding2D)       (None, 230, 230, 3)  0           input_1[0][0]
________________________________________________________________________________________________
conv1_conv (Conv2D)             (None, 112, 112, 64) 9472        conv1_pad[0][0]
________________________________________________________________________________________________
conv1_bn (BatchNormalization)   (None, 112, 112, 64) 256         conv1_conv[0][0]
________________________________________________________________________________________________
conv1_relu (Activation)         (None, 112, 112, 64) 0           conv1_bn[0][0]
________________________________________________________________________________________________
pool1_pad (ZeroPadding2D)       (None, 114, 114, 64) 0           conv1_relu[0][0]
________________________________________________________________________________________________
pool1_pool (MaxPooling2D)       (None, 56, 56, 64)   0           pool1_pad[0][0]
________________________________________________________________________________________________
conv2_block1_1_conv (Conv2D)    (None, 56, 56, 64)   4160        pool1_pool[0][0]
________________________________________________________________________________________________
conv2_block1_1_bn (BatchNormali (None, 56, 56, 64)   256         conv2_block1_1_conv[0][0]
________________________________________________________________________________________________
conv2_block1_1_relu (Activation (None, 56, 56, 64)   0           conv2_block1_1_bn[0][0]
________________________________________________________________________________________________
conv2_block1_2_conv (Conv2D)    (None, 56, 56, 64)   36928       conv2_block1_1_relu[0][0]
________________________________________________________________________________________________
conv2_block1_2_bn (BatchNormali (None, 56, 56, 64)   256         conv2_block1_2_conv[0][0]
________________________________________________________________________________________________
conv2_block1_2_relu (Activation (None, 56, 56, 64)   0           conv2_block1_2_bn[0][0]
________________________________________________________________________________________________
```

Fig. *Summary of ResNet50*

5.2 Training by your computer at home at least 5 epochs and use TensorBoard to monitor, Save the final **screenshot of TensorBoard** (**5%**, Use other tools can get 3%).

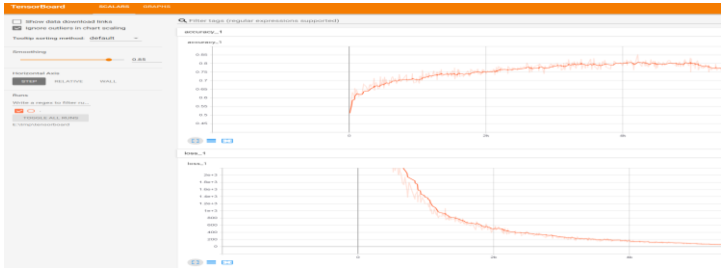


Fig. *Example of training with TensorBoard*

5.3 **Randomly select** a picture from the test set and mark its predicted category.(**5%**)

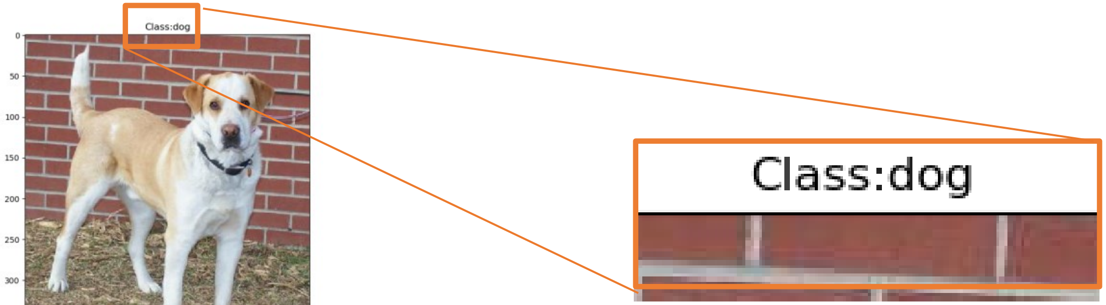
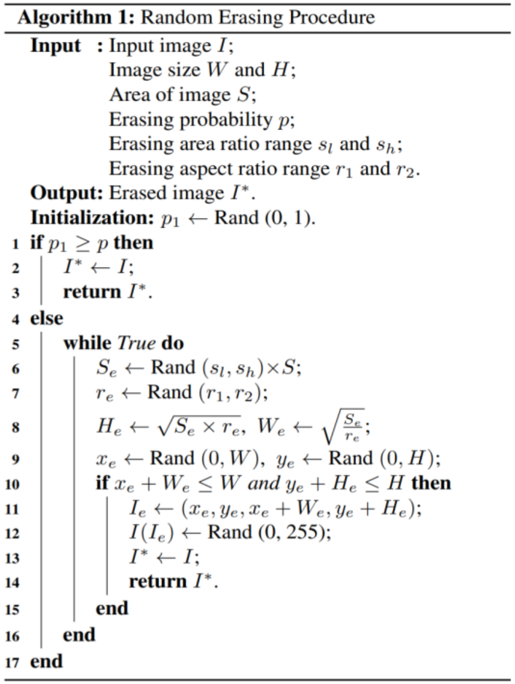


Fig. *Classification display example*

5.4 Improve ResNet50 with Random-Erasing:

Train another model with Random-Erasing data augmentation. **Write the code of Random-Erasing (3%)** and **draw the comparison table of accuracy**, save it as a picture.**(2%)**



```
Algorithm 1: Random Erasing Procedure
Input  : Input image $I$;
         Image size $W$ and $H$;
         Area of image $S$;
         Erasing probability $p$;
         Erasing area ratio range $s_l$ and $s_h$;
         Erasing aspect ratio range $r_1$ and $r_2$.
Output: Erased image $I^*$.
Initialization: $p_1 \leftarrow$ Rand (0, 1).
if $p_1 \geq p$ then
    $I^* \leftarrow I$;
    return $I^*$.
else
    while True do
        $S_e \leftarrow$ Rand $(s_l, s_h) \times S$;
        $r_e \leftarrow$ Rand $(r_1, r_2)$;
        $H_e \leftarrow \sqrt{S_e \times r_e}$,  $W_e \leftarrow \sqrt{\frac{S_e}{r_e}}$;
        $x_e \leftarrow$ Rand $(0, W)$,  $y_e \leftarrow$ Rand $(0, H)$;
        if $x_e + W_e \leq W$ and $y_e + H_e \leq H$ then
            $I_e \leftarrow (x_e, y_e, x_e + W_e, y_e + H_e)$;
            $I(I_e) \leftarrow$ Rand (0, 255);
            $I^* \leftarrow I$;
            return $I^*$.
        end
    end
end
```

Fig. *Random-Erasing algorithm*

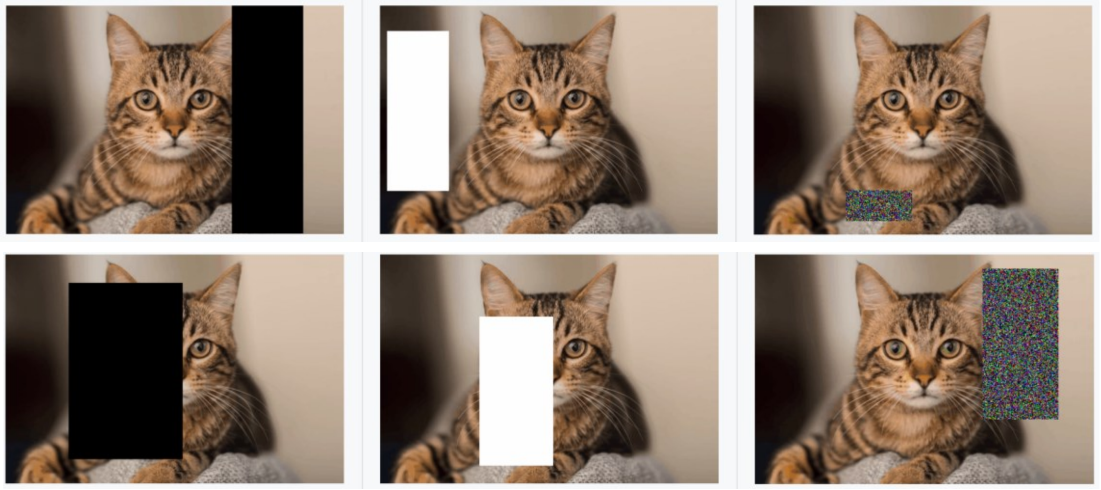


Fig. *Examples of the use of Random-Erasing*

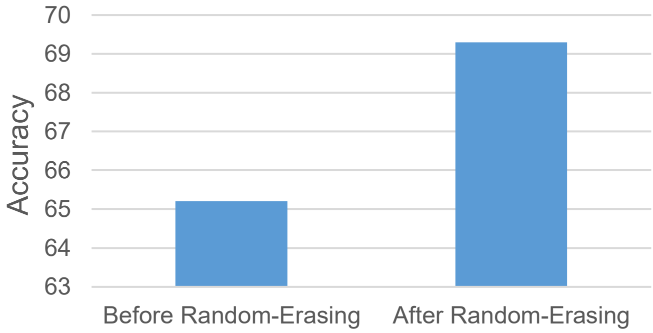


Fig. *Random-Erasing effect comparison example*

R. Reference

Random Erasing Data Augmentation