電腦視覺與深度學習
(Computer Vision and Deep Learning)
Homework 1

TA:

Lydia: lydia2200284@gmail.com

Office Hour: 17:00~19:00, Mon.

10:00~12:00, Wed.

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 2020/11/04 (Wed.)
  - No delay. If you submit homework after deadline, you will get 0.
- Upload to => 140.116.154.1 -> Upload/Homework/Hw1
  - User ID: cvdl2020 Password: cvdl2020
- Format
  - Filename: Hw1_StudentID_Name_Version.rar
    - Ex: Hw1_F71234567_ 林小明 _V1.rar
    - If you want to update your file, you should update your version to be V2, ex: Hw1_F71234567_ 林小明 _V2.rar
  - Content: project folder*(including the pictures)
    *note: remove your "Debug" folder to reduce file size

Notices (2/2)

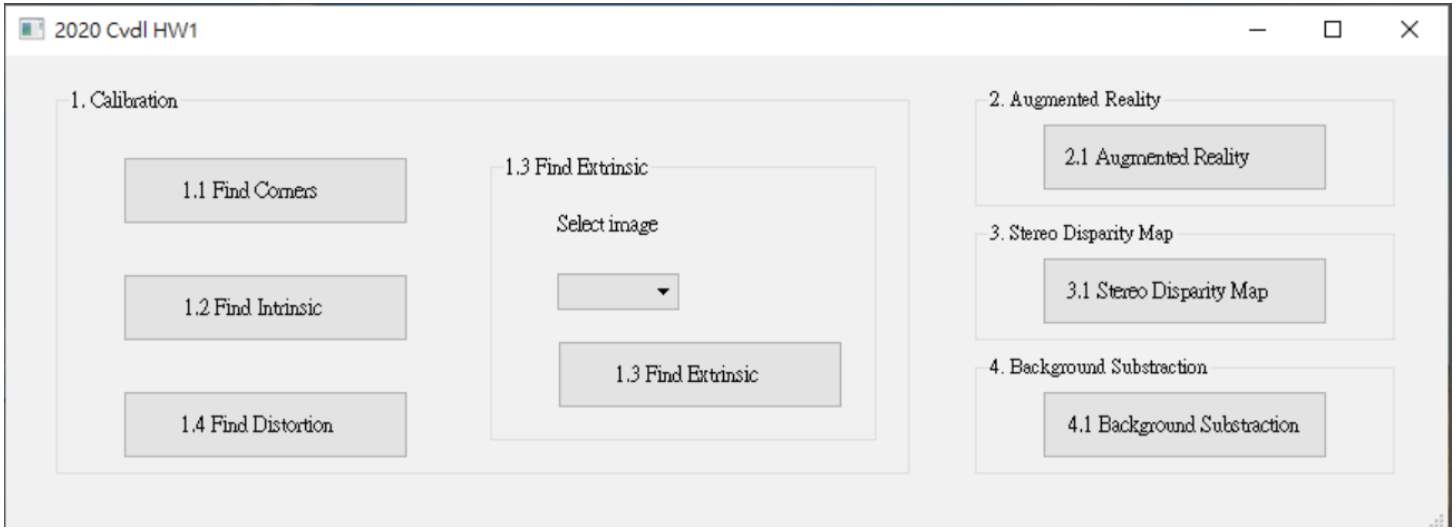
- Python
  - Python 3.7 (https://www.python.org/downloads/)
  - opencv-contrib-python (3.4.2.17)
  - Matplotlib 3.1.1
  - UI framework: pyqt5 (5.15.1)

- 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%, reference) Camera Calibration (出題： Max)
   - 1.1 Corner detection (5%)
   - 1.2 Find the intrinsic matrix (5%)
   - 1.3 Find the extrinsic matrix (5%)
   - 1.4 Find the distortion matrix (5%)
2. (20%) Augmented Reality (出題： Oran)
3. (20%) Stereo Disparity Map (出題： Mark)
4. (20%) SIFT (出題： Brian)
   - 4.1 Show keypoints (10%)
   - 4.2 Show matching keypoints (10%)
5. (20%) Training Cifar10 Classifier Using VGG16

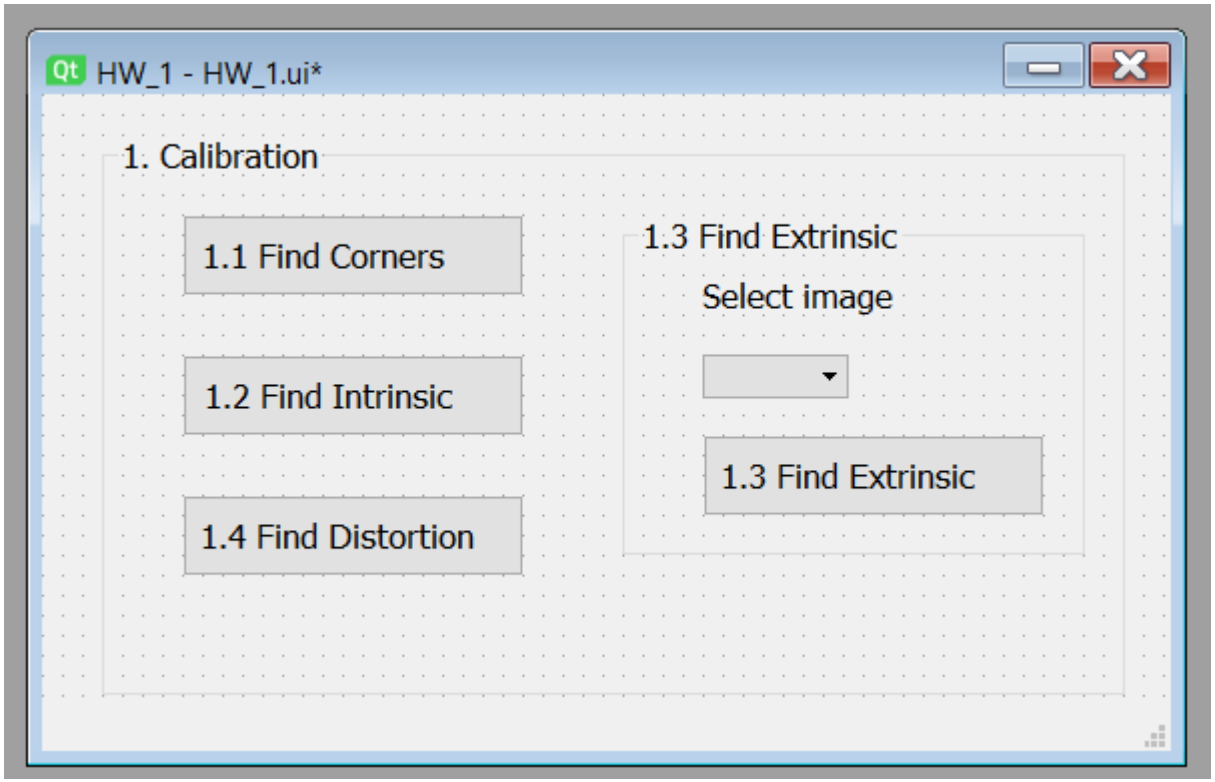
UI example



1. (20%) Camera Calibration

(出題： Max)

1.1 (5%) Corner detection
1.2 (5%) Find the intrinsic matrix
1.3 (5%) Find the extrinsic matrix
1.4 (5%) Find the distortion matrix

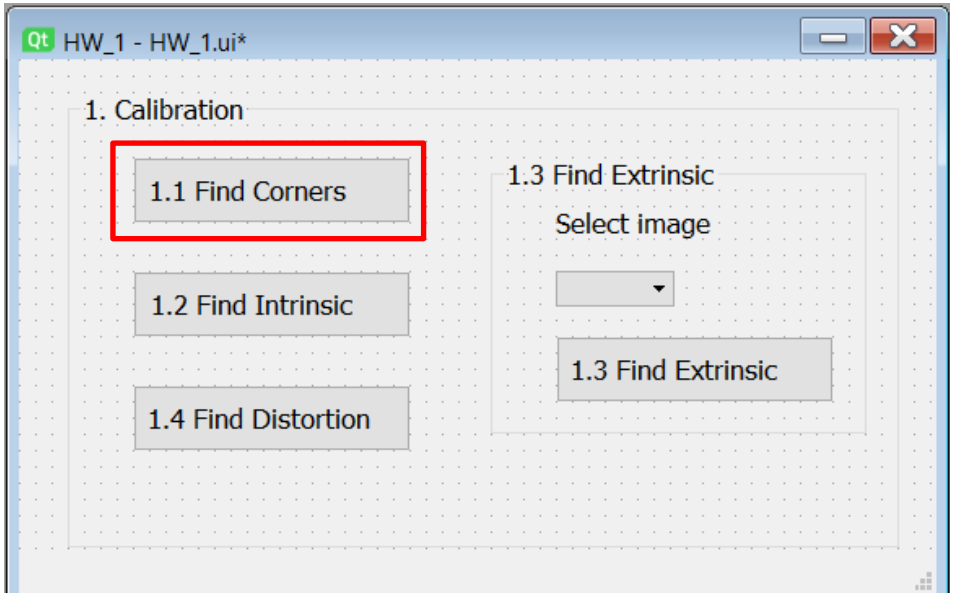


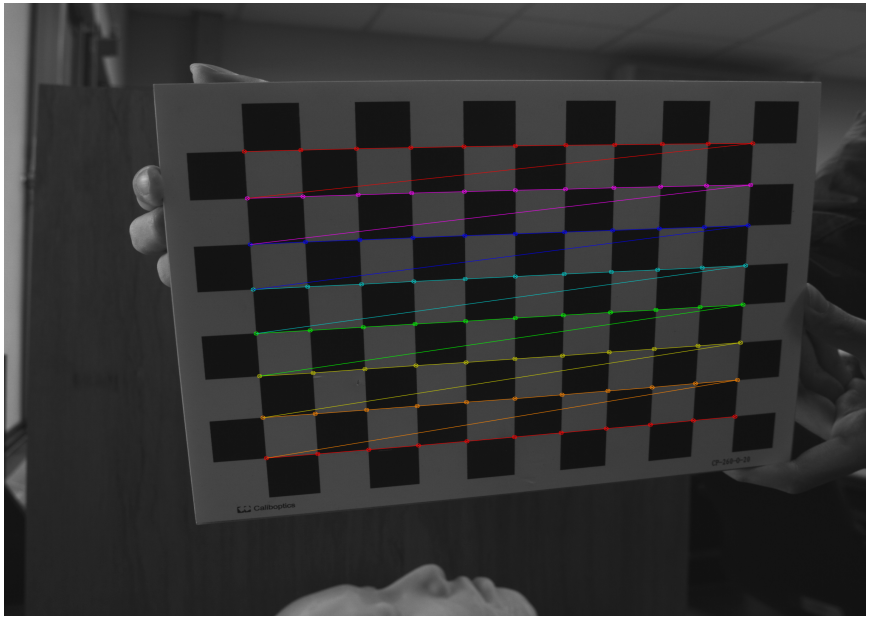
1.1 Corner Detection

- Given: 15 images, 1.bmp ~ 15.bmp
- Q: 1) Find and draw the corners on the chessboard for each image.
  2) Click button “1.1” to show the result.
- Hint :

  OpenCV Textbook Chapter 11 (p. 398 ~ p. 399)

  cvShowImage(…);
- Ex:





1.2 Find the Intrinsic Matrix

- Given: 15 images, 1.bmp ~ 15.bmp
- Q: 1) Find the intrinsic matrix ():

$$\begin{bmatrix} \alpha & \gamma & u_0 \\ 0 & \beta & v_0 \\ 0 & 0 & 1 \end{bmatrix}$$

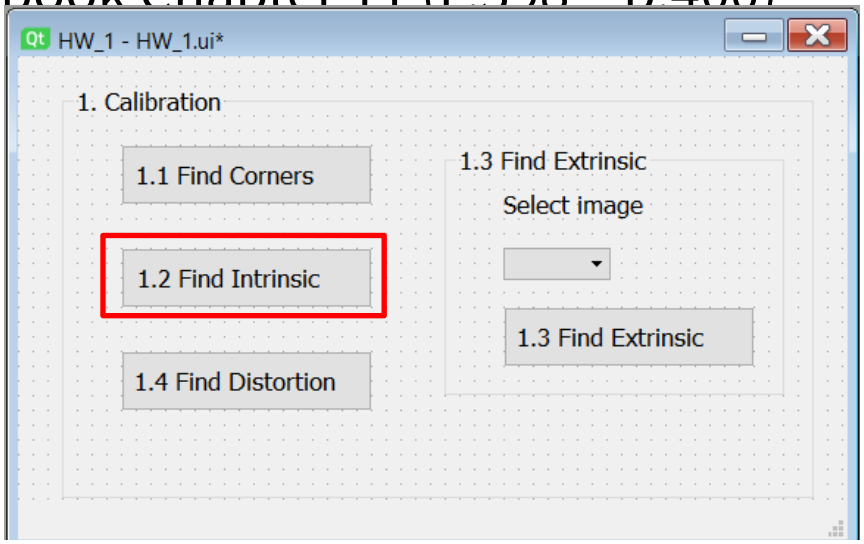
2) Click button “1.2” and then show the result on the console window.

- Output format:

```
[2227.333008, 0.000000, 384.186066;
0.000000, 2226.654541, 299.351746;
0.000000, 0.000000, 1.000000]
```

(Just an example)

- Hint: OpenCV Textbook Chapter 11 (P.398 ~ p.400)



1.3 Find the Extrinsic Matrix

- Given: intrinsic parameters, distortion coefficients, and the list of 15 images
- Q: 1) Find the extrinsic matrix of the chessboard for each of the 15 images, respectively:

$$\begin{bmatrix} R_{11} & R_{12} & R_{13} & T_1 \\ R_{21} & R_{22} & R_{23} & T_2 \\ R_{31} & R_{32} & R_{33} & T_3 \end{bmatrix}$$

2) Click button “1.3” and then show the result on the console window.

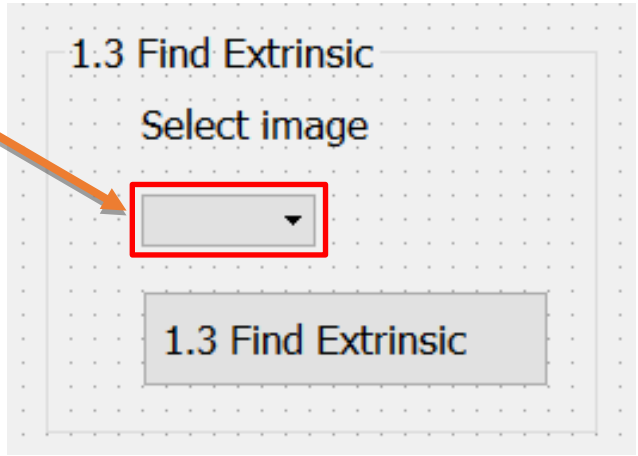
- Output format:

```
[-0.128827  ,0.991169  ,-0.031426  ,-1.969988 ;
0.983549  ,0.131755  ,0.123583  ,-1.105037 ;
0.126632  ,-0.014988  ,-0.991836  ,49.121323 ;  ]
```

(Just an example)

- Hint: OpenCV Textbook Chapter 11, p.370~402

(1) List of numbers: 1~15
(2) Select 1, then 1.bmp will be applied, and so on



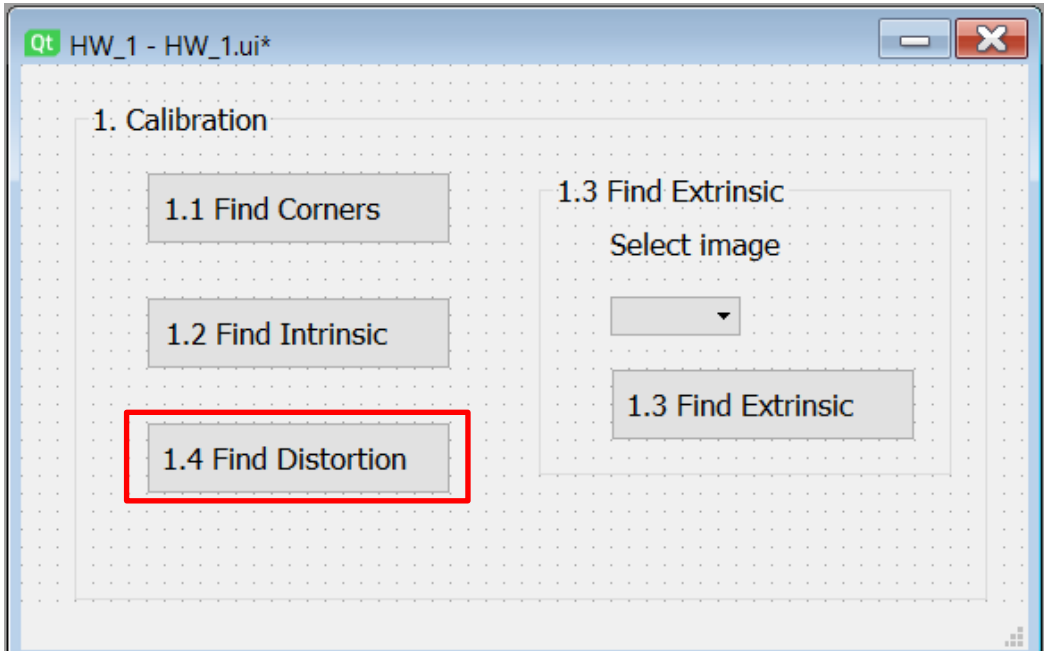
1.4 Find the Distortion Matrix

- Given: 15 images
- Q: 1) Find the distortion matrix: $[k_1, k_2, p_1, p_2, k_3]$
  2) Click button “1.4” to show the result on the console window.
- Output format:

```
[-0.072230, -0.261944, -0.000024, -0.003354, 4.228090]
```

(Just an example)

- Hint:
  - Distortion coefficients can be obtained simultaneously with intrinsic parameters
  - OpenCV Textbook Chapter 11 (P.398 ~ p.400)



2. (20%) Augmented Reality

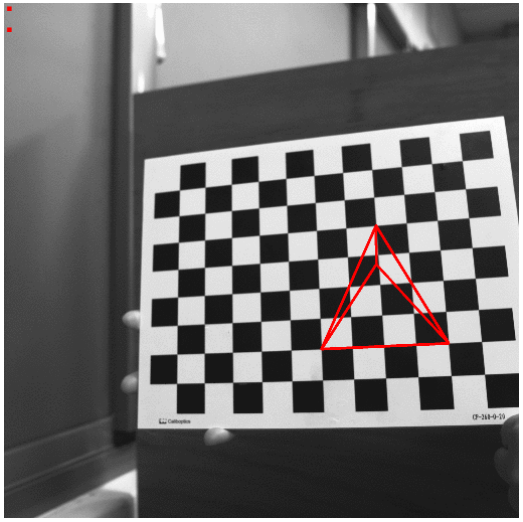
(出題： Oran)

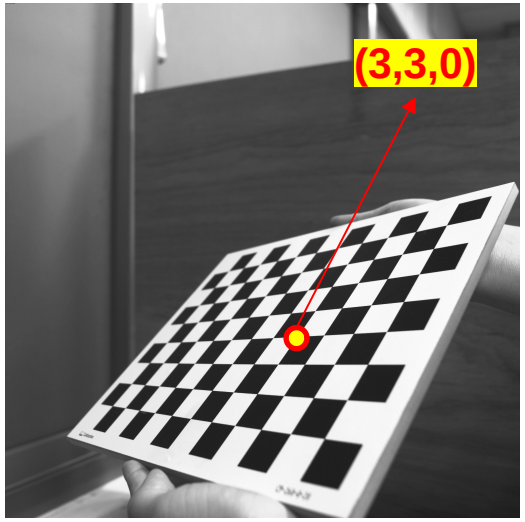
- Given: 5 images: 1~5.bmp
- Q:
  - 1) Calibrate 5 images to get intrinsic, distortion and extrinsic parameters
  - 2) Draw a “tetrahedron” on the chessboards images(1.bmp to 5.bmp)
  - 3) Click the button to show the tetrahedron on the picture. Show each picture 0.5 seconds (total 5 images)

- Hint : Textbook Chapter 11, p.387~395 Calibration
  p.405~412 Projection

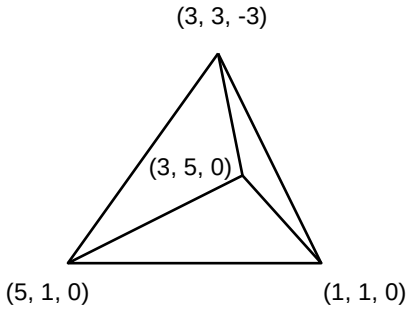
```
cv::calibrateCamera()
cv::projectPoints()
```

Demo





- 3D Object coordinates:
  Vertex (3, 3, -3)
  Corners(1, 1, 0)(3, 5, 0)(5, 1, 0)



3. (20%) Stereo Disparity Map

(出題： Mark)

- Given: a pair of images, imL.png and imR.png (have been rectified)
- Q: Find the disparity map/image based on Left and Right stereo images.

imL.png

Left Image (Reference Image)

imR.png

Right Image

3.1 (20%) Disparity Map

❑ Guides:

(1) Window Size: Must be odd and within the range [5, 255]

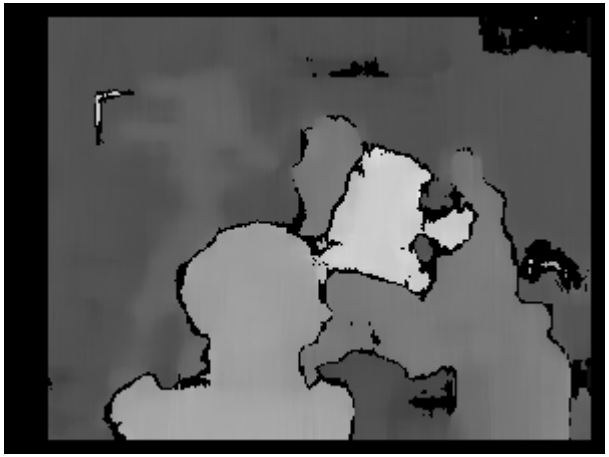
(2) Search range and direction:

- Disparity range:
  - Must be positive and divisible by 16.
  - Map disparity range to gray value range 0~255 for the purpose of visualization.

- If the left image is the reference image (the one used to cal. depth info for each pixel of that img), then the search direction at right image will go from the right to left direction.

✈ Hint: OpenCV Textbook Chapter 12 (P.451)

StereoBM::create(64, 9);

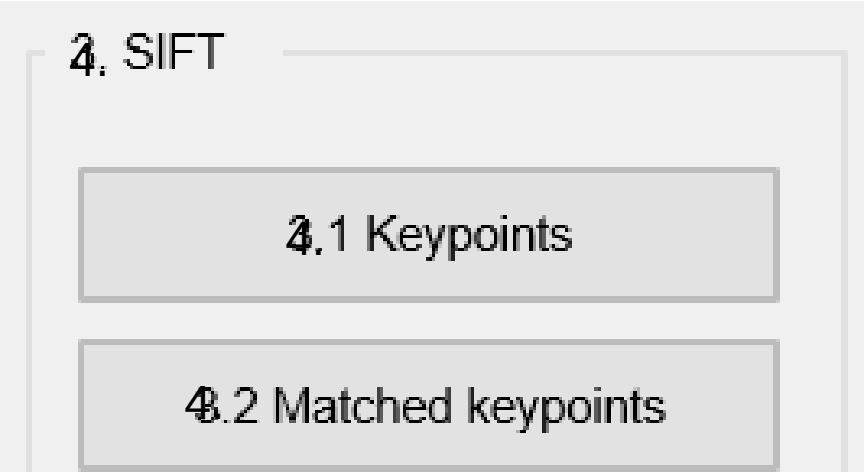


Result

4. (20%) SIFT

(出題：)

- ❑ User interface for the third question:

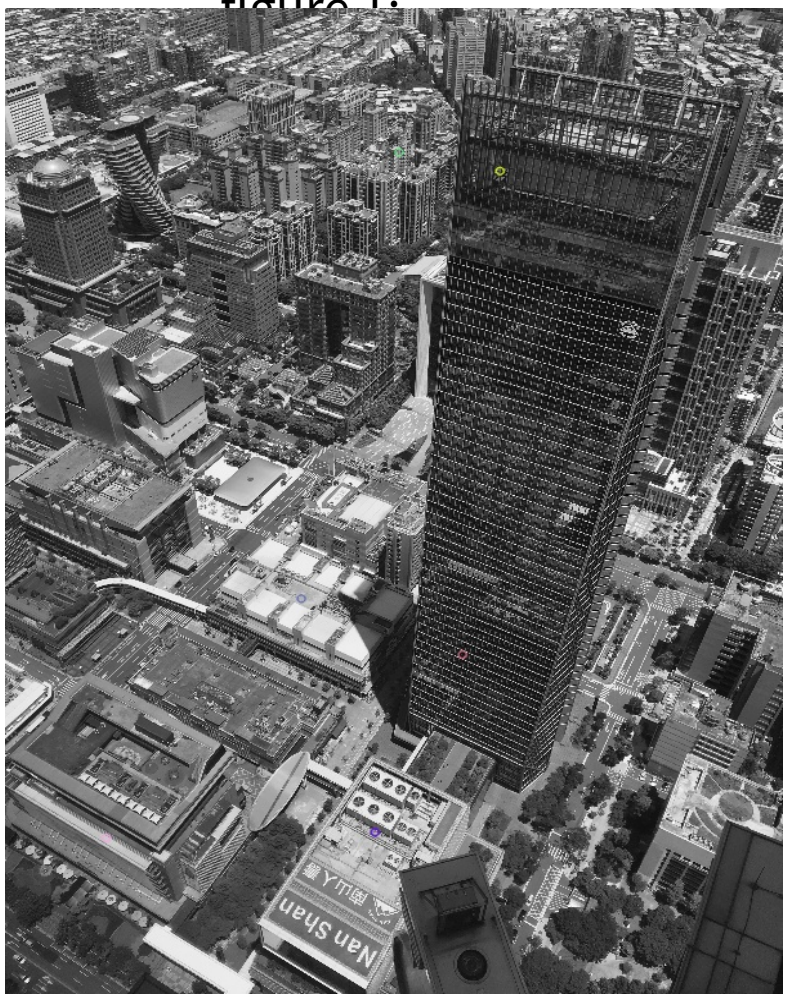


4. (20%) SIFT

(出題：)

❑ Q: 4.1) (10%) Click button “4.1 Keypoints” to show:

- 6 feature points on each Aerial1.jpg and Aerial2.jpg
- then save results as FeatureAerial1.jpg and FeatureAerial2.jpg as figure 1;



FeatureAerial1.jpg

FeatureAerial2.jpg

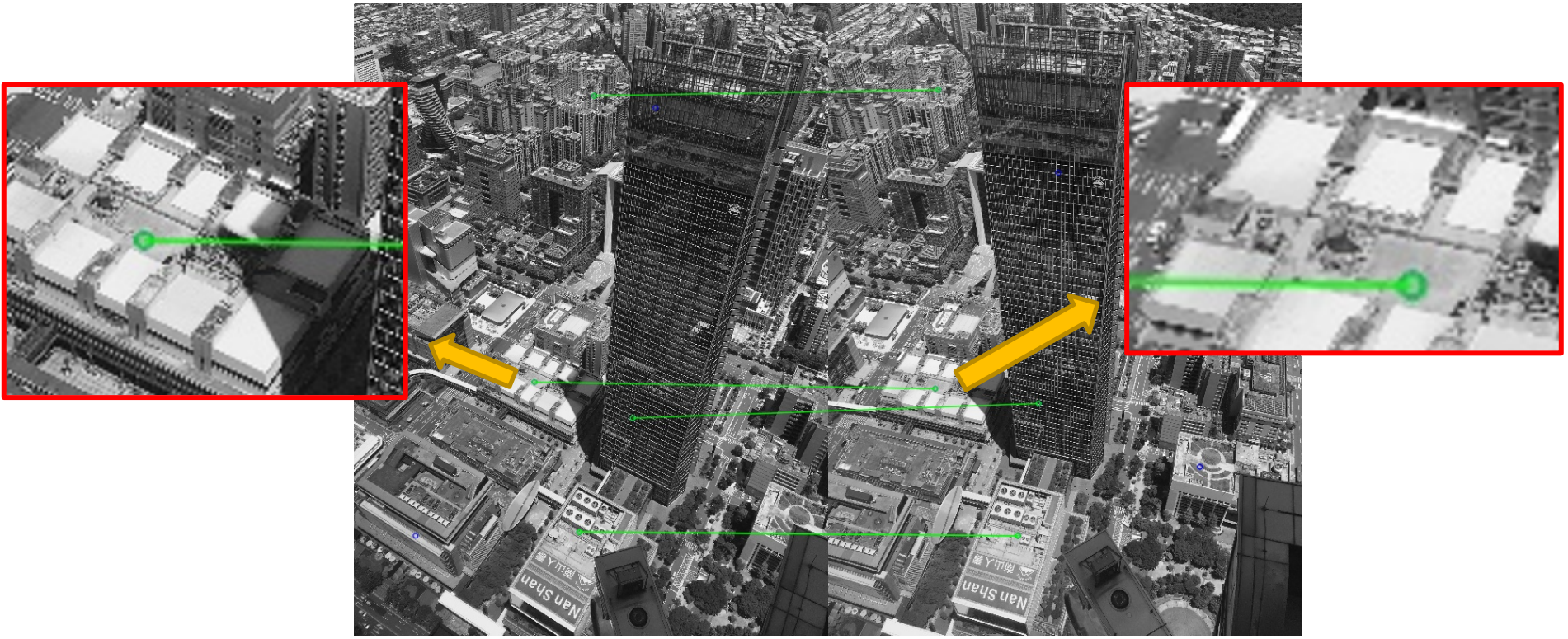
Figure 1. Feature points on two images.

❑ Hint : (ref. : opencv2refman_2.4.7.pdf) ref. p663 ~ p670

4. (20%) SIFT

(出題：)

- Q: 4.2) (10%) Click button “4.2 Matched Keypoints”,
  - draw the matched feature points between two images from 6 keypoints pairs obtained in Q: 4.1) and show the results as Figure 2:



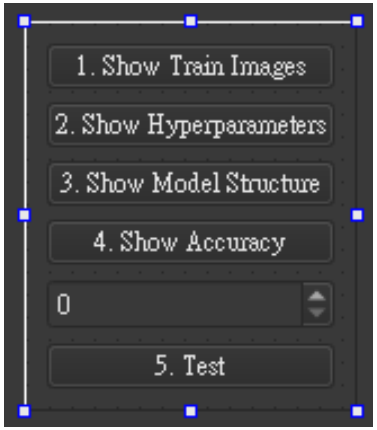
Aerial1.jpg　　Aerial2.jpg

Figure 2. Feature points and their corresponding points.

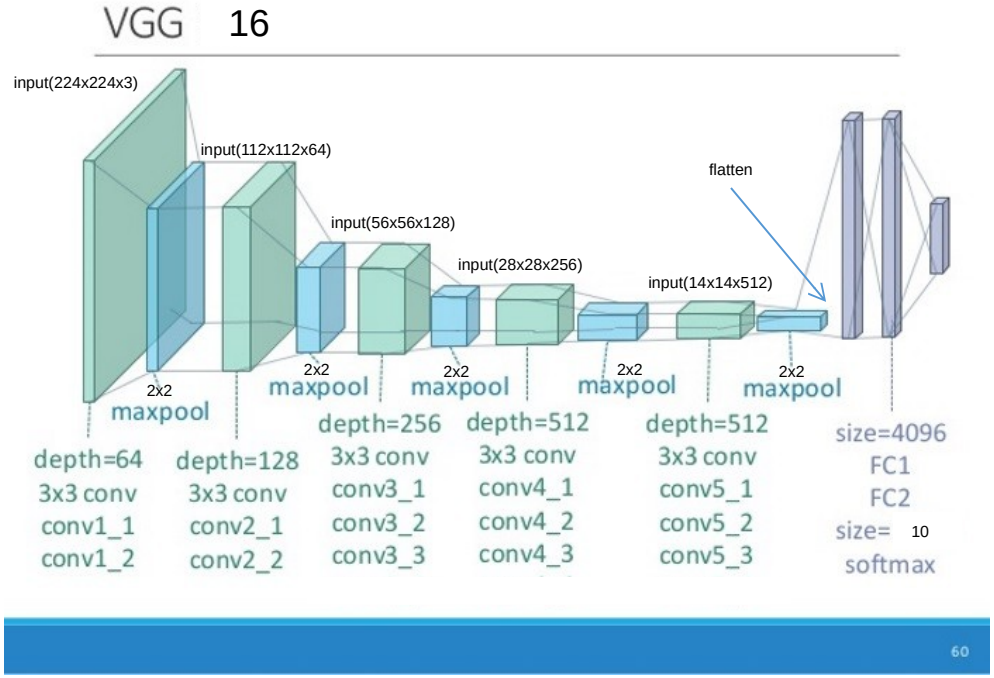
- Hint : (ref. : opencv2refman_2.4.7.pdf) ref. p663 ~ p670

5.0 Training Cifar10 Classifier Using VGG16 （出題： Kevin)

1. Learn how to construct VGG16 and train it on Cifar10.
2. Environment Requirement
   1) Python 3.6
   2) Tensorflow 2 / PyTorch 1.3.0
   3) opencv-contrib-python 3.4.2.17 (for image show and write)
   4) Matplotlib 3.1.1 (for chart drawing)



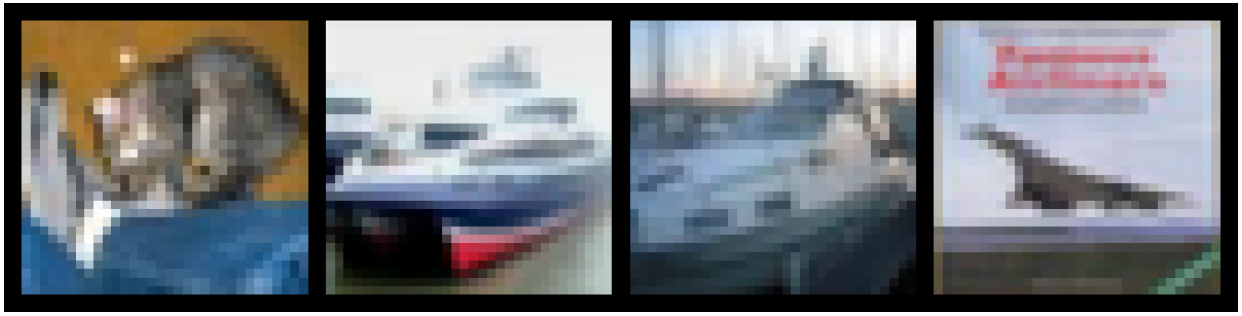
GUI example



VGG16 framework

3. Reference
   1) Very Deep Convolutional Networks for Large-Scale Image Recognition(VGG16)
   2) https://www.cs.toronto.edu/~kriz/cifar.html(Cifar10)

5.1 Load Cifar10 **training** dataset and randomly show 10 images and labels respectively. (4%)



Label: cat ship ship airplane …

5.2 Print out training hyperparameters (batch size, learning rate, optimizer). (4%)

```
hyperparameters:
batch size: 32
learning rate: 0.001
optimizer: SGD
```

5.3 Construct and show your model structure.
(Build it your self, do not use architecture provided by ML framework) (4%)

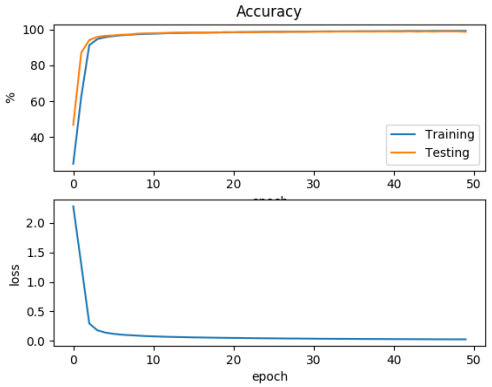
```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 32, 32, 64)        1792
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 32, 32, 64)        36928
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 16, 16, 64)        0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 16, 16, 128)       73856
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 16, 16, 128)       147584
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 8, 8, 128)         0
```

…

```
conv2d_15 (Conv2D)           (None, 2, 2, 512)         2359808
_________________________________________________________________
max_pooling2d_4 (MaxPooling2 (None, 1, 1, 512)         0
_________________________________________________________________
flatten (Flatten)            (None, 512)               0
_________________________________________________________________
dense (Dense)                (None, 4096)              2101248
_________________________________________________________________
dense_1 (Dense)              (None, 4096)              16781312
_________________________________________________________________
dense_2 (Dense)              (None, 10)                40970
=================================================================
Total params: 38,947,914
Trainable params: 38,947,914
Non-trainable params: 0
_________________________________________________________________
```

0 airplane
1 automobile
2 bird
3 cat
4 deer
5 dog
6 frog
7 horse
8 ship
9 truck

5.4 Training your model at least **20** epochs by your own computer, than save your model and take a screenshot of your training loss and accuracy. **No saved images no points** (4%) (you only need to show the screenshot image , DO NOT train in homework UI)



(record accuracy/loss per epoch)

5.5 Load your model trained at 5.4, let us choose one image from test images, inference the image, show image and estimate the image as following. **No saved model no points** (4%)