影像處理、電腦視覺及深度學習概論
(Introduction to Image Processing, Computer Vision and Deep Learning)

Homework 1

TA:

West: player210421@gmail.com

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

10:00~12:00, Wed.

At CSIE 9F Robotics Lab.

Notice (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 =>23:59:59, 2020/11/O5 (Thu.)

  No delay. If you submit homework after deadline, you will get 0.
- Upload to => 140.116.154.1 -> Upload/Homework/OpenCv_Hw1
  - User ID: opencvdl2020 Password: opencvdl2020
- 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

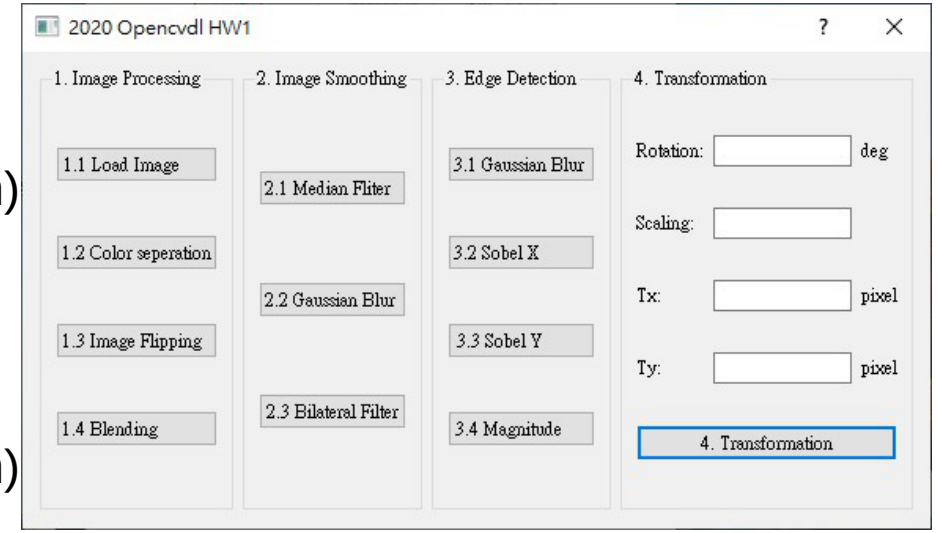
Notice (2/2)

- Python (recommended)
  - 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

Assignment scoring (Total: 100%)

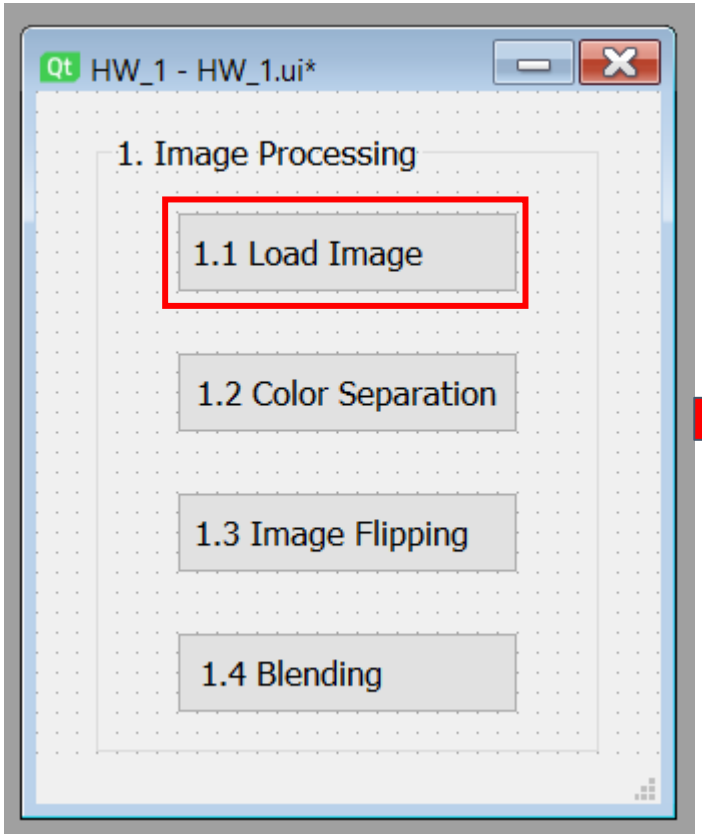
1. (20%) Image Processing (出題： Jimmy)
   - 1.1 (5%) Load Image File
   - 1.2 (5%) Color Separation
   - 1.3 (5%) Image Flipping
   - 1.4 (5%) Blending
2. (20%) Image Smoothing (出題： Brian)
   - 2.1 (7%) Median filter
   - 2.2 (7%) Gaussian blur
   - 2.3 (6%) Bilateral filter
3. (20%) Edge Detection (出題： Shan)
   - 3.1 (5%) Gaussian Blur
   - 3.2 (5%) Sobel X
   - 3.3 (5%) Sobel Y
   - 3.4 (5%) Magnitude
4. (20%) Transforms: (出題： Jimmy)
   - 4.1 (7%) Rotation, (出題： Kevin)
   - 4.2 (7%) Scaling,
   - 4.3 (6%) Translation
5. (20%) Training Cifar-10 Classifier Using VGG16



1.1 Load Image File

(出題： Jimmy)

- Given: “Uncle_Roger.jpg” image
- Q: 1) Open a new window to show the image (Uncle_Roger.jpg)
  2) Show the height and width of the image in **console mode**
- Hint : Textbook Chapter 2, p. 22~23

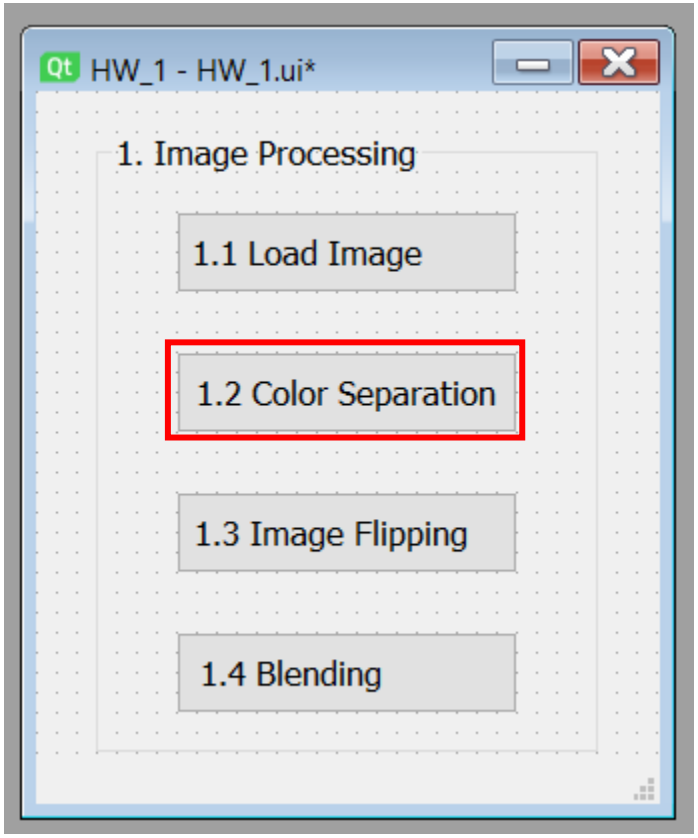


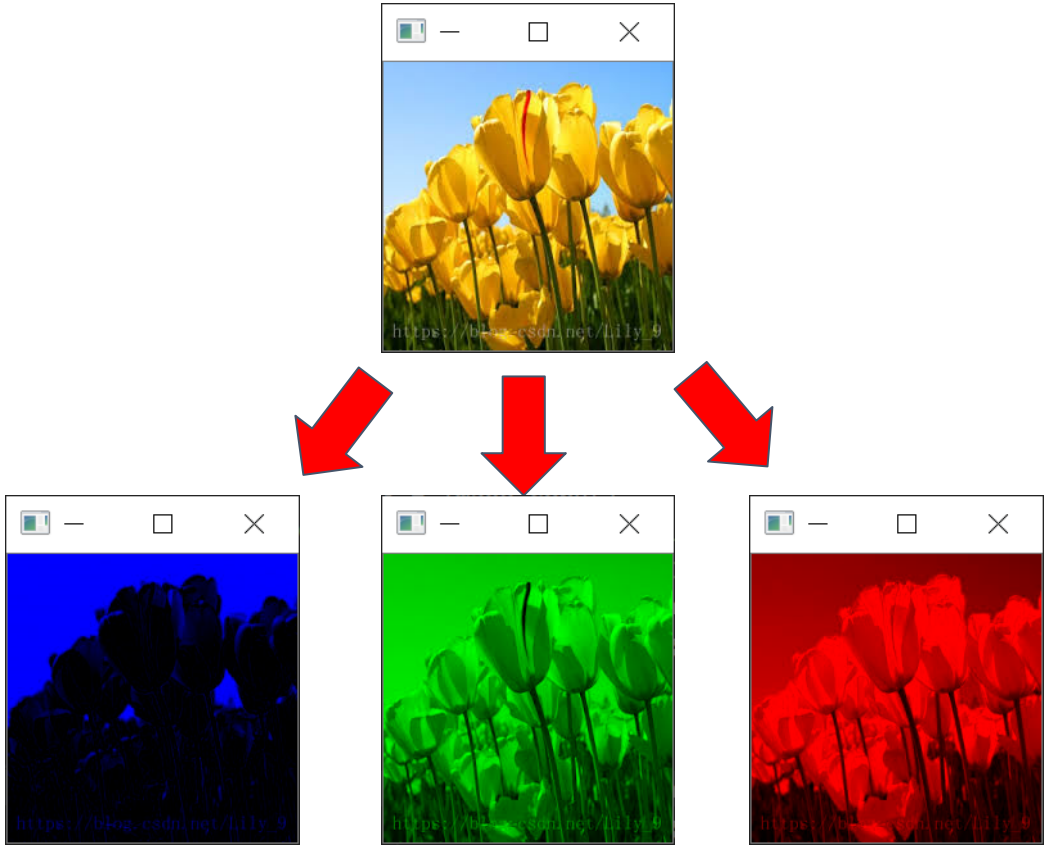
圖片出處 : Youtuber Uncle Roger (https://www.youtube.com/channel/UCVjlpEjEY9GpksqbEesJnNA)

1.2 Color Separation

(出題： Jimmy)

- Given: a color image, “Flower.jpg”
- Q: 1) Extract 3 channels of the image BGR to 3 separated channels.
  2) Open 3 new windows to show result images.
- Hint: Textbook Chapter 3, p.31 ~ p.44

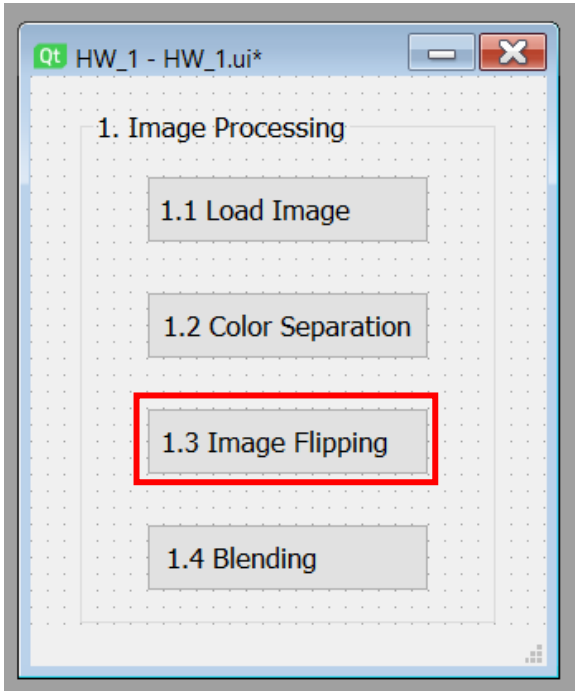


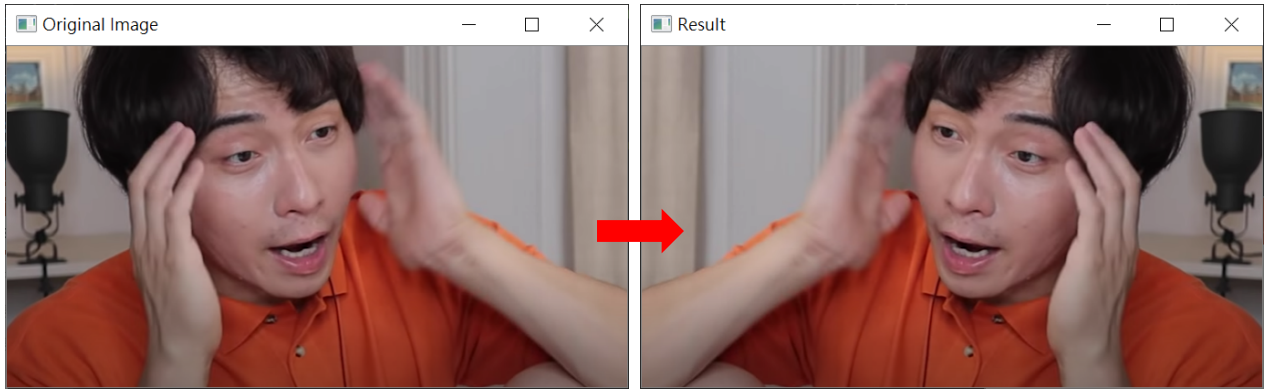


1.3 Image Flipping

(出題： Jimmy)

- Given: an image, “Uncle_Roger.jpg”
- Q: 1) Flip the image (Uncle_Roger.jpg) and open a new window to show the result.
- Hint: Textbook Chapter 3, p.31 ~ p.44

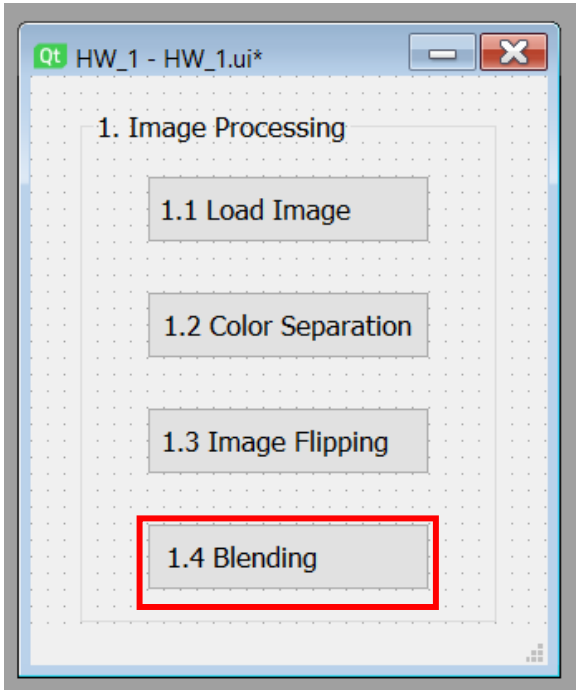


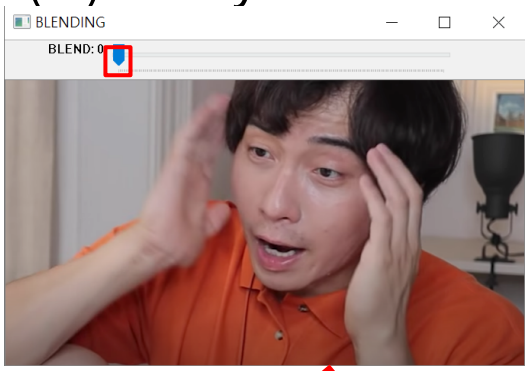


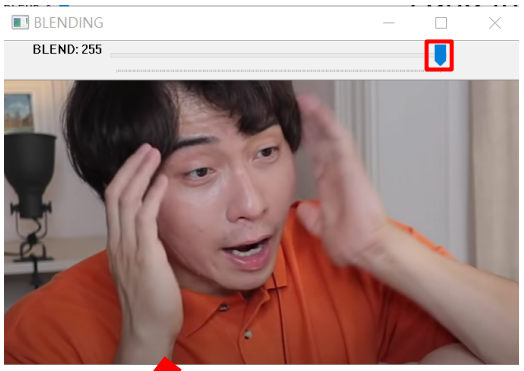
1.4 Blending

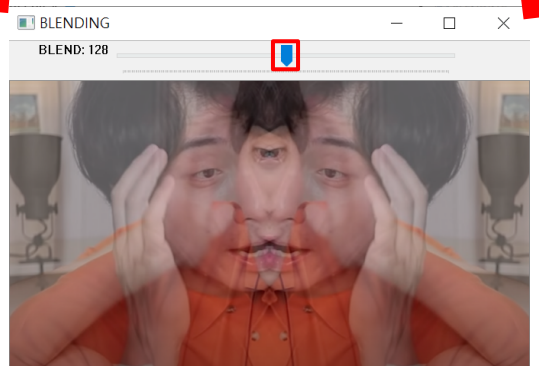
(出題： Jimmy)

- Given: 2 images, “Uncle_Roger.jpg” and the result of 1.3
- Q: 1) Combine two images (Uncle_Roger.jpg and the result of 1.3).
  2) Use Trackbar to change the weights and show the result in the new window.
- Hint:
  - Textbook Chapter 3, p. 51 ~ 51
  - cv2.createTrackbar(...) for Python









2.1 Median filter

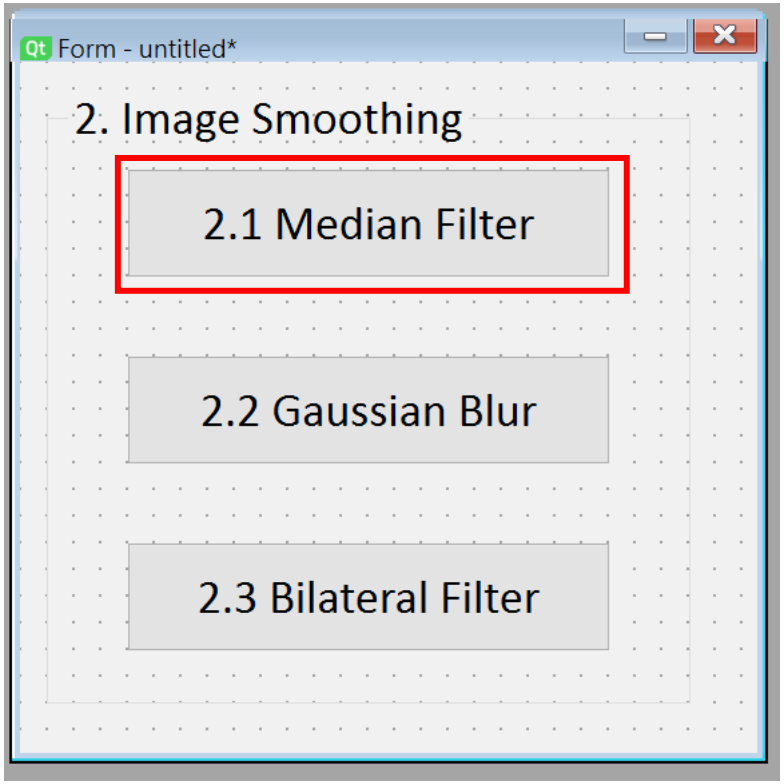
(出題： Brian)

- Given: a color image, “Cat.png”
- Q: 1) Apply 7x7 median filter to “cat.png”
- Hint: Textbook Chapter 5, p.109 ~ p.115

223	186	114
204	161	106
219	194	138

106 114 138 161 **186** 194 204 219 223

Median filter example



2.2 Gaussian Blur

(出題： Brian)

- Given: a color image, “Cat.png”
- Q: 1) Apply 3x3 Gaussian blur to “Cat.png”
- Hint: Textbook Chapter 5, p.109 ~ p.1

1/16

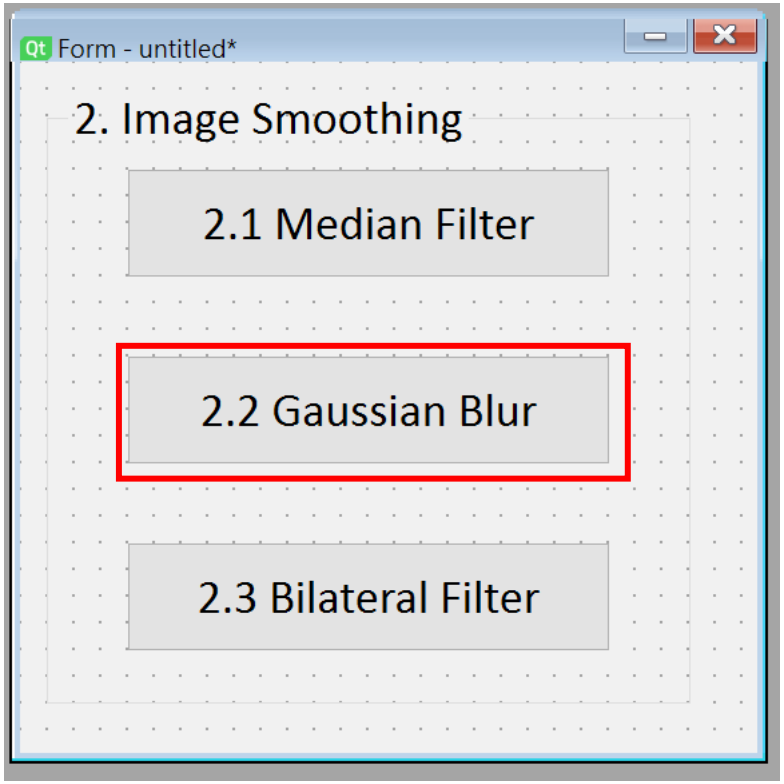
1	2	1
2	4	2
1	2	1

1/273

1	4	7	4	1
4	16	26	16	4
7	26	41	26	7
4	16	26	16	4
1	4	7	4	1

1/1003

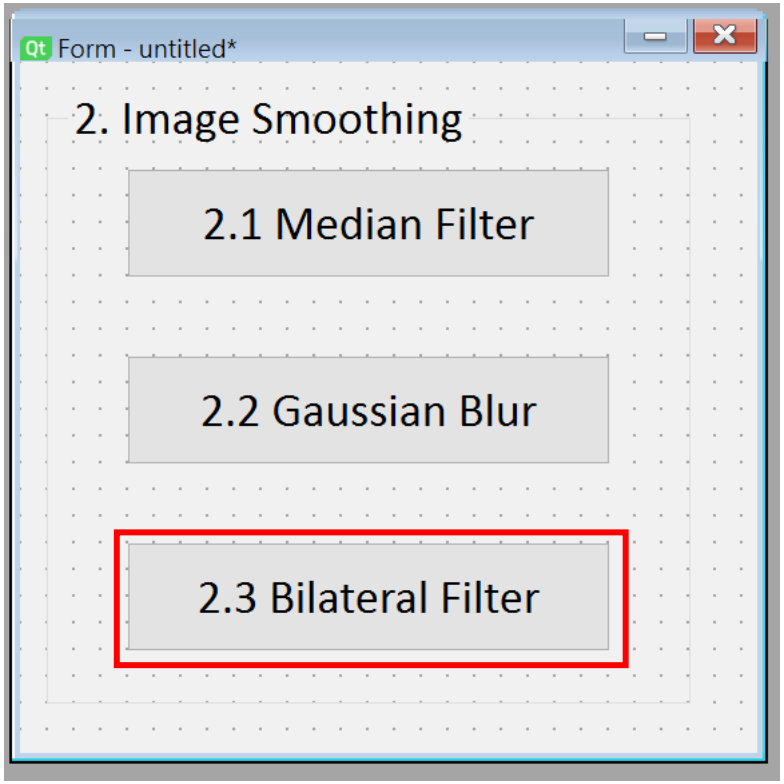
0	0	1	2	1	0	0
0	3	13	22	13	3	0
1	13	59	97	59	13	1
2	22	97	159	97	22	2
1	13	59	97	59	13	1
0	3	13	22	13	3	0
0	0	1	2	1	0	0

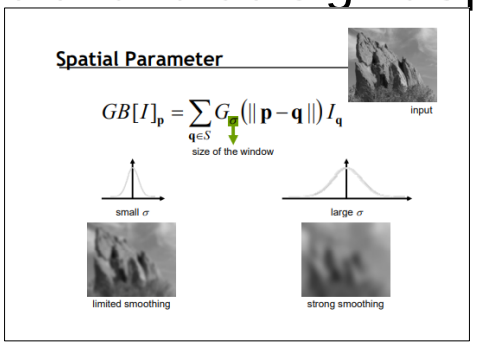


2.3 Bilateral filter

(出題： Brian)

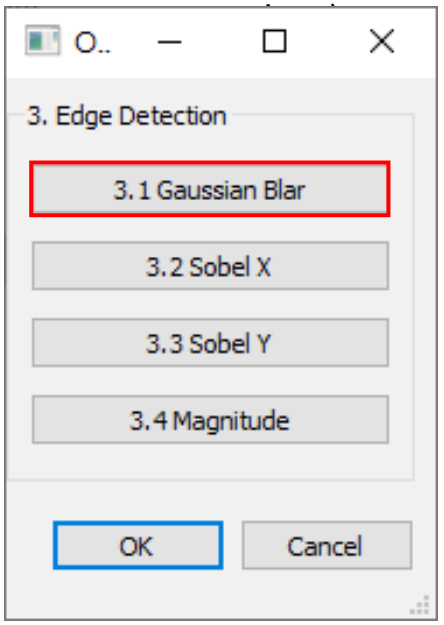
- Given: a color image, “Cat.png”
- Q: 1) Apply 9x9 Bilateral filter with 90 sigmaColor and 90 sigmaSpace to “Cat.png”
- Hint: Textbook Chapter 5, p.109 ~ p.115





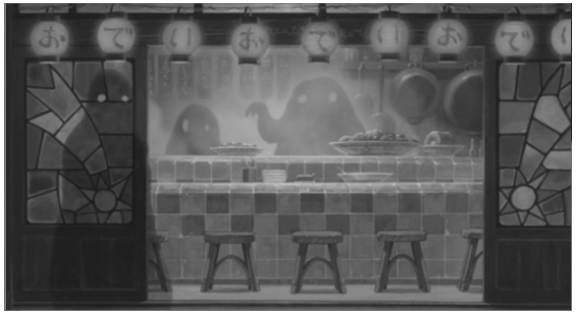
3.1 Gaussian Blur

(出題：

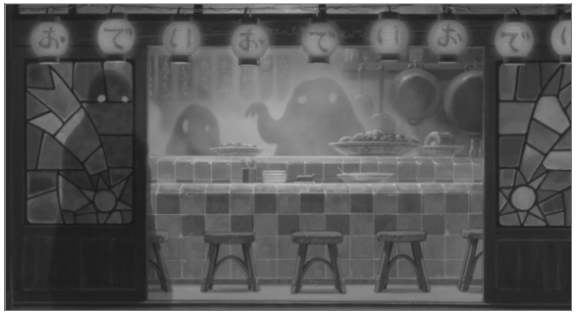


- Given: a color image, “Chihiro.jpg”
- Q: 1) **Gaussian Blur**: Convert the color image into a grayscale image, then smooth it by your own 3x3 Gaussian smoothing filter (Can not use OpenCV Function). Please show the result.
- Hint: Textbook Chapter 5, p.109 ~ p.115
  How to generate Gaussian Filter:
  ①Make
  ②Sub into
  ③Normalize

Chihiro.jpg



Grayscale

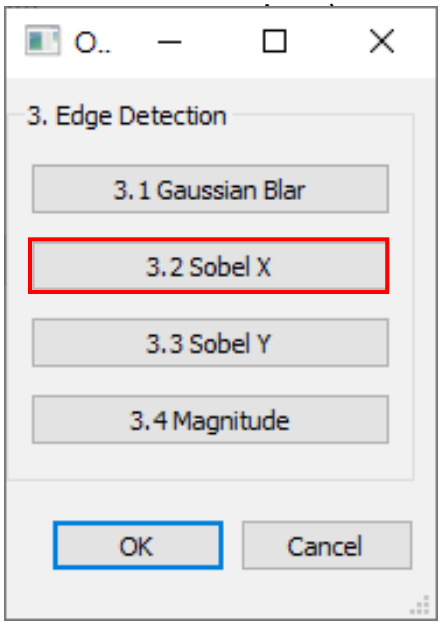


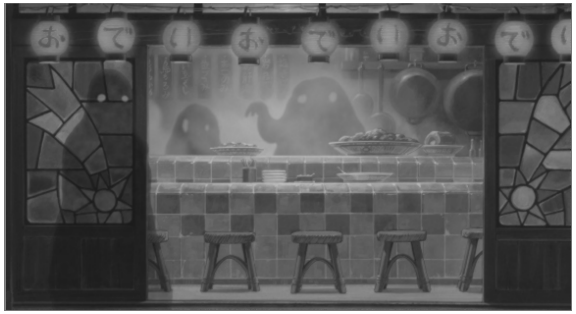
Gaussian Blur

3.2 Sobel X

(出題：

- Given: the result of 3.1) Gaussian Blur
- Q: 2) **Sobel X**: Use Sobel edge detection to detect vertical edge by your own 3x3 Sobel X operator (Can not use OpenCV Function). Please show the result.
- Hint: Textbook Chapter 6, p.148 ~ 149
  If the image can not be show, please normalize the result to 0~255.

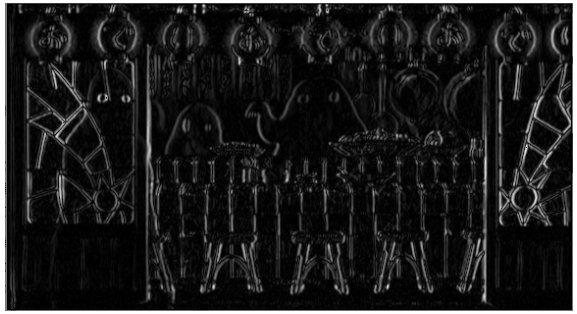




Gaussian Blur

-1	0	1
-2	0	2
-1	0	1

Sobel X

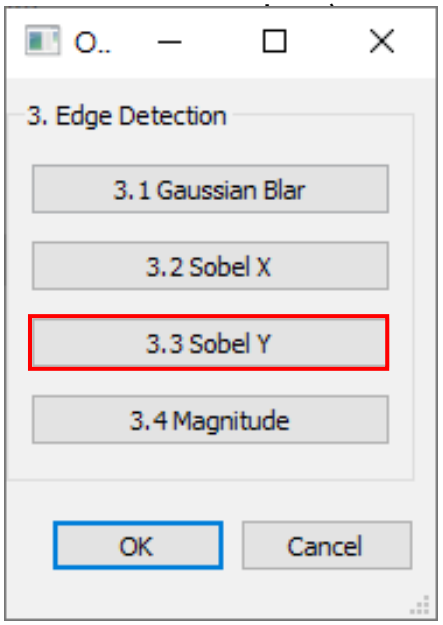


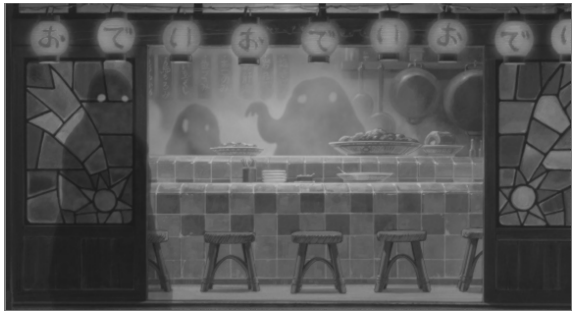
Sobel X

3.3 Sobel Y

(出題：

- Given: the result of 3.1) Gaussian Blur
- Q: 3) **Sobel Y**: Use Sobel edge detection to detect horizontal edge by your own 3x3 Sobel Y operator (Can not use OpenCV Function). Please show the result.
- Hint: Textbook Chapter 6, p.148 ~ 149
  If the image can not be show, please normalize the result to 0~255.





Gaussian Blur

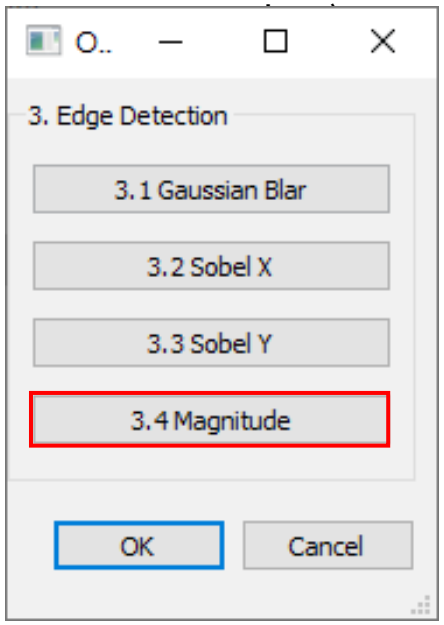
1	2	1
0	0	0
-1	-2	-1

Sobel Y

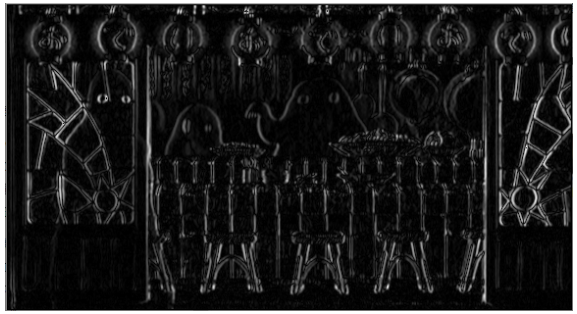
Sobel Y

3.4 Magnitude

(出題：

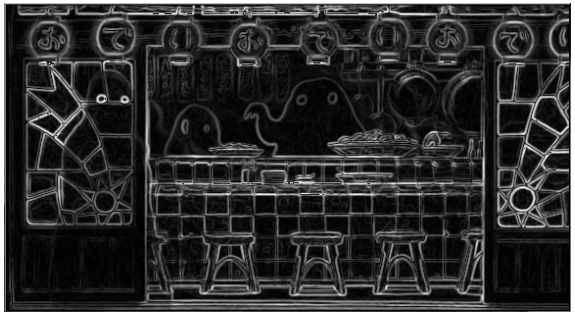


- Given: The result of 3.2) Sobel X and 3.3) Sobel Y
- Q: 4) **Magnitude**: Use the results of 3.2) Sobel X and 3.3) Sobel Y to calculate the magnitude. Please show the result.
- Hint: Textbook Chapter 6, p.148 ~ 149
  If the image can not be show, please normalize the result to 0~255.
  Magnitude =



Sobel X

Sobel Y



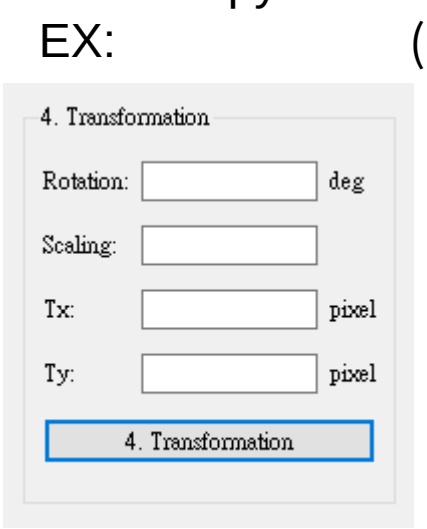
Magnitude

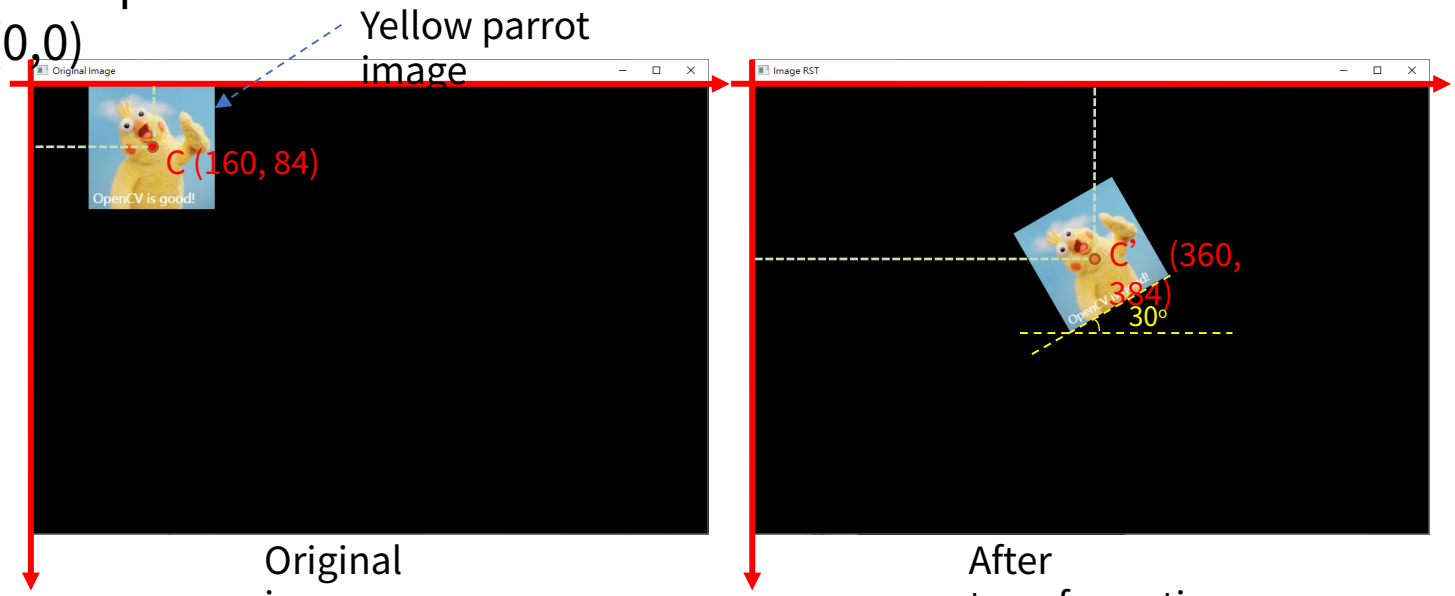
4.1 Transforms: Rotation, Scaling, Translation

(出題 : Jimmy)

- Given: "*Parrot.png*"
- Q: 1) Click button "4. Transformation", *Parrot.png* should be showed.
  2) Please rotate, scale and translate the yellow parrot (as image below) with following parameters (should be entered in the GUI)
  (1) Angle = 30° (counter-clockwise)
  (2) Scale = 0.9,
  (3) Translation with:
  - Xnew = Xold + 200 pixels = 160 + 200 = 360
  - Ynew = Yold + 300 pixels = 84 + 300 = 384
    Point C (160, 84) is center of yellow parrot image

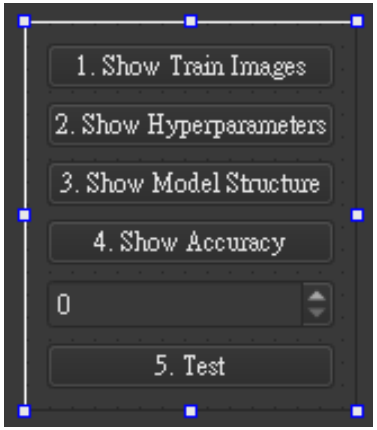
- Hint: Textbook Chapter 12, (p.407 ~ 412)
  python: `cv.warpAffine`
- EX:



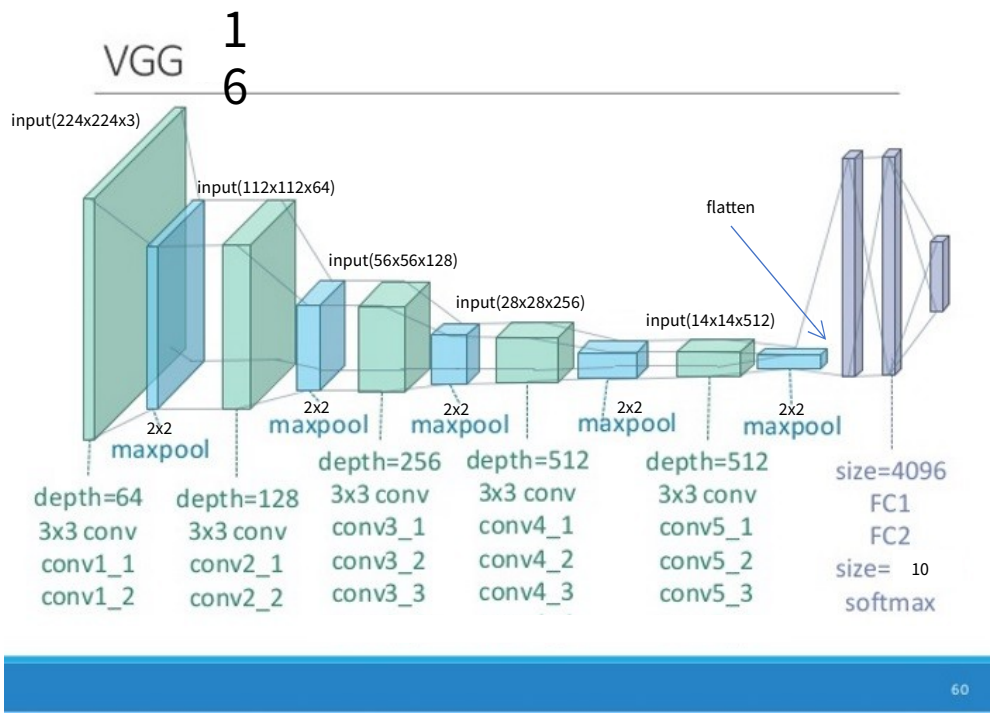


5.0 Training Cifar10 Classifier Using VGG16 （出題：Kevin)

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)



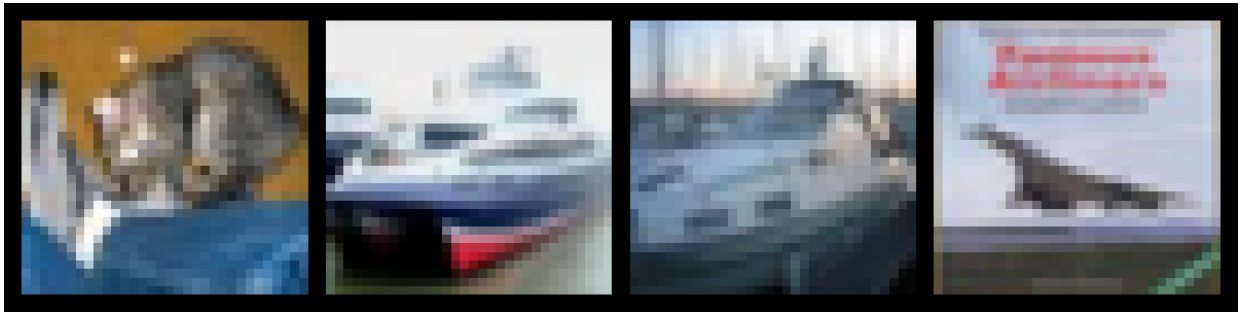
GUI example



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).

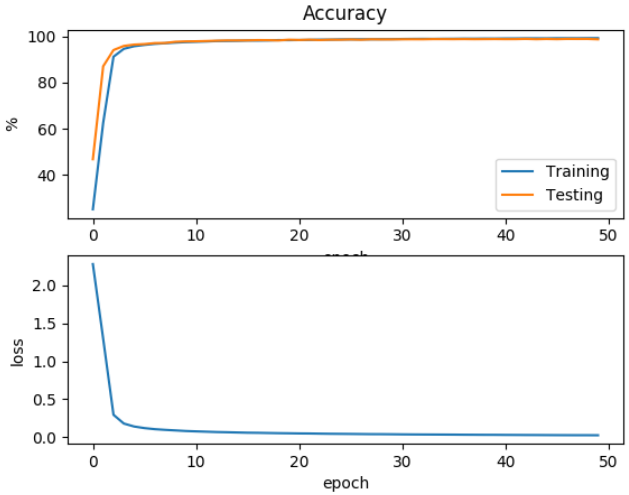
```
hyperparameters:
batch size: 32
learning rate: 0.001
optimizer: SGD
```

5.3 Construct and show your model structure. (You can use available network to build your model)

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
vgg16 (Model)                (None, 1, 1, 512)         14714688
_________________________________________________________________
flatten (Flatten)            (None, 512)               0
_________________________________________________________________
dense (Dense)                (None, 4096)              2101248
_________________________________________________________________
dense_1 (Dense)              (None, 4096)              16781312
_________________________________________________________________
dense_2 (Dense)              (None, 10)                40970
=================================================================
Total params: 33,638,218
Trainable params: 33,638,218
Non-trainable params: 0
```

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

5.4 Training your model at least **20** epochs by your own computer, save your model and take a screenshot of your training loss and accuracy. **No saved images no points** (4%)



(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 poin**