

OOP 期末專題報告


專題製作人: 蔡亞彤

報告製作時間: 2024/6/16

1. Step1~Step5 結果圖


i. Step1 : Play around data loader class

```
Please type in 1~4 to choose which function you want to demonstrate.
1:setup 2:step2 3:bit_field_filter 4:photo mosaic
1
For gray image, I use "Image-Folder/mnist/img_119.jpg" as input, and output as task1_present1.jpg
```



- Gray image X server display -

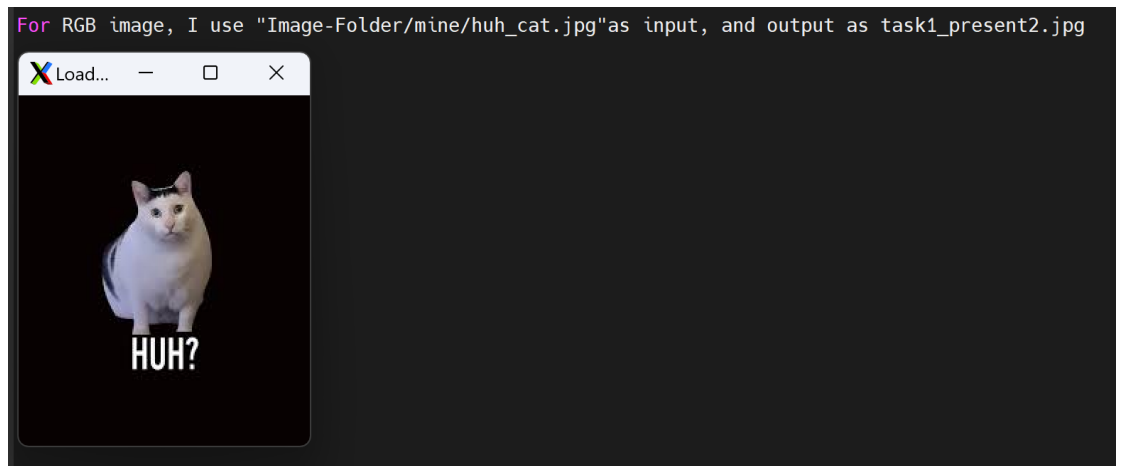
```
For gray image, I use "Image-Folder/mnist/img_119.jpg" as input, and output as task1_present1.jpg
```



- Gray image CMD display -

```
#####
@#@#####
@@@@@@@@@##
@@@@+++++--
@@+
..@@@
--@
##@@@@##+@##
@@@@@@@@@.
--+ .++@@@##
-- --@-
@@@@+
--+
--@##
..@.
..@.
--#
++##-- ..##+
..@@@@++##@.
##@@@@@@@@-
---#####@+..
```

- Gray image ASCII code display -



- RGB image X server display -



- RGB image CMD display -



- RGB image ASCII code display -

ii. Step2 :Construct image inheritance and polymorphism

```

1 //image.h
2 #ifndef _IMAGE_H_
3 #define _IMAGE_H_
4
5 #include<string>
6 #include "data_loader.h"
7 using namespace std;
8
9 class Image{
10 public:
11     Image();
12     ~Image();
13     virtual void LoadImage(string);
14     virtual void DumpImage(string);
15     virtual void Display_X_Server();
16     virtual void Display_ASCII();
17     virtual void Display_CMD();
18     virtual void Image_Box_Filter(int);
19     virtual void Image_Median_Filter(int);
20     virtual void Image_Contrast_Filter();
21     virtual void Image_Advanced_Contrast_Filter();
22     virtual void Image_Mosaic_Filter(int);
23     virtual void loadfile_name(string);
24     int read_w ();
25     int read_h ();
26 protected:
27     int w;
28     int h;
29 };
30
31 #endif

```

- Image.h -

```

1 //gray_image.h
2 #ifndef _GRAY_IMAGE_H_
3 #define _GRAY_IMAGE_H_
4
5 #include "image.h"
6 #include <string>
7
8 class GrayImage : public Image{
9 public:
10     GrayImage();
11     ~GrayImage();
12     virtual void LoadImage(string);
13     virtual void DumpImage(string);
14     virtual void Display_X_Server();
15     virtual void Display_CMD();
16     virtual void Display_ASCII();
17 private:
18     int ** pixels;
19     string class_filename;
20
21 };
22
23 #endif

```

- Gray image.h -

```

1 //rgb_image.h
2 #ifndef _RGB_IMAGE_H_
3 #define _RGB_IMAGE_H_
4
5 #include "image.h"
6 #include <string>
7
8 class RGBImage : public Image{
9 public:
10     RGBImage();
11     ~RGBImage();
12     virtual void LoadImage(string);
13     virtual void DumpImage(string);
14     virtual void Display_X_Server();
15     virtual void Display_CMD();
16     virtual void Display_ASCII();
17     virtual void Image_Box_Filter(int);
18     virtual void Image_Median_Filter(int);
19     virtual void Image_Contrast_Filter();
20     virtual void Image_Advanced_Contrast_Filter();
21     virtual void Image_Mosaic_Filter(int);
22     virtual void loadfile_name(string);
23     int readpixels(int,int,int);
24     void writepixels(int,int,int,int);
25 private:
26     int *** pixels;
27     string class_filename;
28
29 };
30
31 #endif

```

-RGB image.h -

```

1 //photo_mosaic.h
2 #ifndef _PHOTO_MOSAIC_H_
3 #define _PHOTO_MOSAIC_H_
4
5 #include "data_loader.h"
6 #include "rgb_image.h"
7 #include <vector>
8 using namespace std;
9 class PhotoMosaic : public RGBImage{
10 public:
11     RGBImage* photo_mosaic(string , string);
12     int find_the_best_tile(int*,int**,int);
13 private:
14     vector <string> filenames;
15     RGBImage* outputimg;
16 protected:
17     void writepixels_photo(int,int,int,int);
18 };
19
20 #endif

```

- Photo mosaic.h -

iii. Step3 :bit_field_filter

```
1  #include "bit_field_filter.h"
2  using namespace std;
3  void Apply_Box_Filter(Image* a,int kernal){
4      //cout<<"inbox"<<endl;
5      a->Image_Box_Filter(kernal);
6      return ;
7  }
8  void Apply_Median_Filter(Image* a,int mask){
9      a->Image_Median_Filter(mask);
10     return ;
11 }
12 void Apply_Contrast_Filter(Image* a){
13     a->Image_Contrast_Filter();
14     return ;
15 }
16 void Apply_Advanced_Contrast_Filter(Image* a){
17     a->Image_Advanced_Contrast_Filter();
18     return;
19 }
20 void Apply_Mosiac_Filter(Image* a,int kernal){
21     a->Image_Mosiac_Filter(kernal);
22     return;
23 }
24 void Apply_Alpha_Trimmed_Mean_Filter(Image* a){
25
26     return;
27 }
28 void Apply_Sobel_Gradient_Filter(Image* a){
29
30     return;
31 }
```

- Bit field filter.cpp -

```

1 //using bitfield to not to force user to passing all of the argument
2 //using bitwise or to passing the options
3 //using bitwise and to get the info of the bitfield
4
5 //1.box filter 2.median filter 3.contrast streching 4.mosiac filter
6 #ifndef BIT_FIELD_FILTER_H
7 #define BIT_FIELD_FILTER_H
8
9 #include <stdio.h>
10 #include <stdint.h>
11 #include <iostream>
12 #include "image.h"
13
14 #define Box_Filter          0b00000001
15 #define Median_Filter      0b00000010
16 #define Contrast_Strechng  0b00000100
17 #define Mosiac_Filter      0b00001000
18 #define Alpha_Trimmed_Mean_Filter 0b00010000
19 #define Sobel_Gradient_Filter 0b00100000
20
21 void Apply_Box_Filter(Image* a,int kernal);
22 void Apply_Median_Filter(Image* a,int mask);
23 void Apply_Contrast_Filter(Image* a);
24 void Apply_Advanced_Contrast_Filter(Image* a);
25 void Apply_Mosiac_Filter(Image* a,int kernal);
26 void Apply_Alpha_Trimmed_Mean_Filter(Image* a);
27 void Apply_Sobel_Gradient_Filter(Image* a);
28
29 #endif

```

- Bit field filter.h -

(Alpha Trimmed Mean Filter 以及 Sobel Gradient Filter 來不及做完)

Box Filter 結果：

(左上角有 filter 選擇跟 kernel size)



原圖



經過 Box Filter 之後的圖

(kernel size : 3)

(kernel size : 7)

可以看出 kernel size 是 7 的比 3 的更模糊

Median Filter 結果： (左上角可以看到 filter 選擇跟 mask size)



原圖



(這裡 mask 選擇 5)



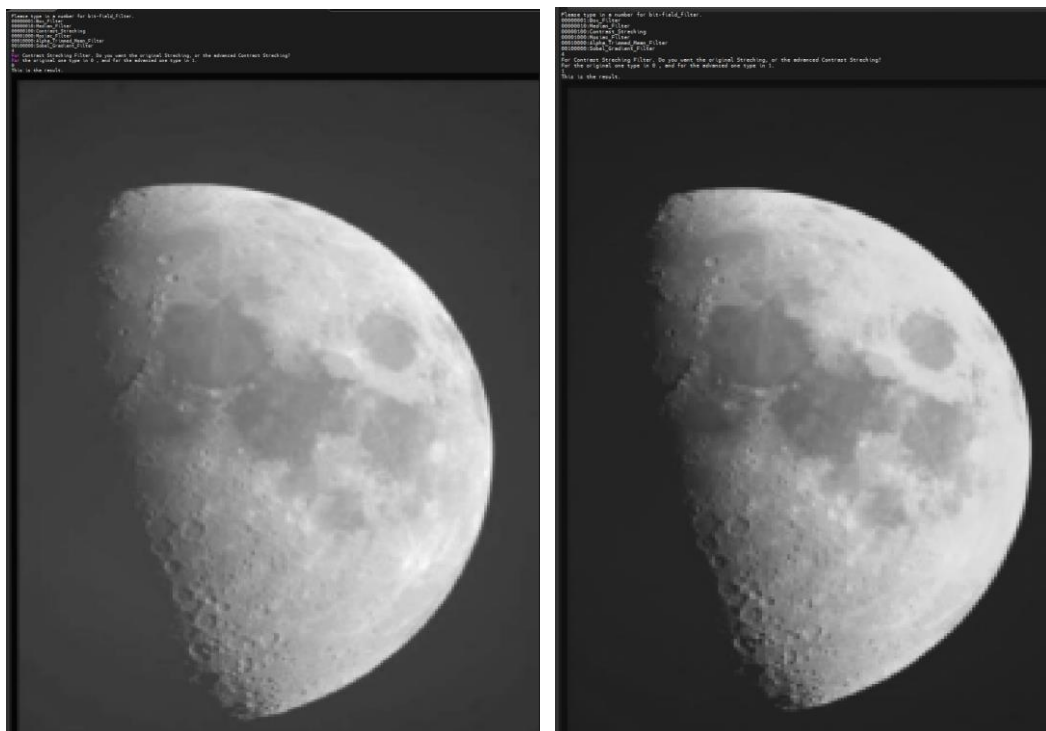
(這裡 mask 選擇 11)

可以看出 Median Filter 是 11 的比 5 得更模糊，雜訊濾除效果就更好了。

Contrast stretching 結果： (左上角可以看到



原圖



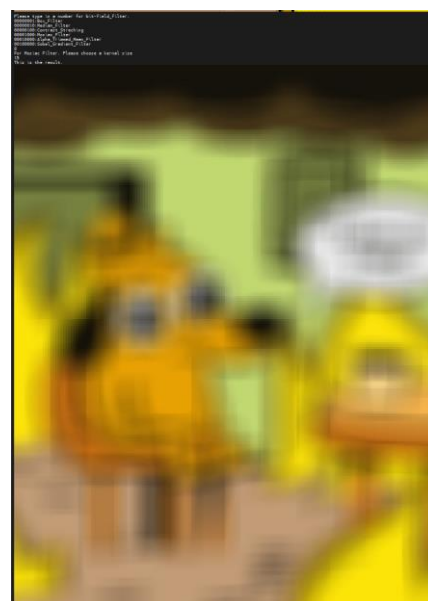
經過 Contrast Stretching 後的圖

(左圖是使用基礎的 Contrast Stretching 公式，而右圖是使用助教在 HackMD 筆記中提到由郭庭維同學所做的算法，可以由此看出右圖在對比度拉伸的部分做得比左圖更好。)

Mosaic Filter 結果: (左上角可以看到 filter 選擇跟 kernal size)



原圖



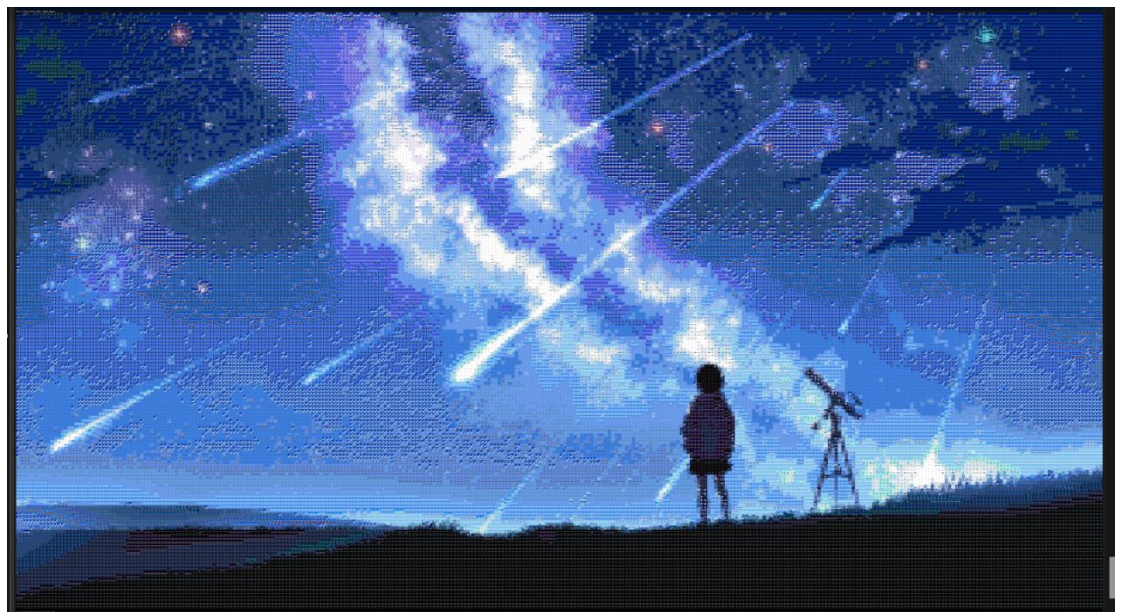
經過 Mosaic Filter 後的圖

(左圖 kernal size : 7) (右圖 kernal size : 15)

- iv. Step4 :Photo Mosaic (左上角可以看到選擇 photo mosaic)



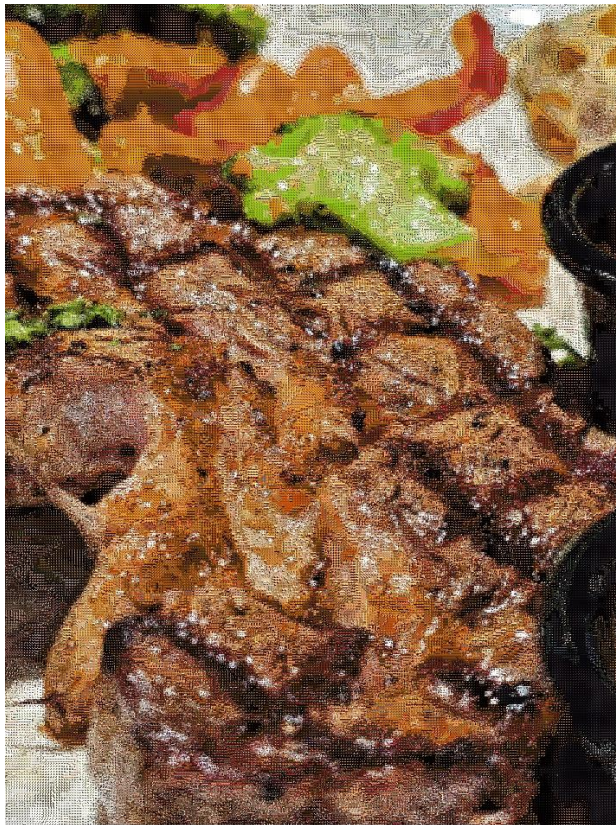
原圖



經過 Photo Mosaic 後的圖



原圖



經過 Photo_Mosaic 後的圖

2. 解釋整份 project 中哪邊使用了繼承與多型

繼承:

我在這份 project 中把 Photo_Mosaic 繼承了 RGBImage，這是為了讓 Photo_Mosaic 可以取得 RGBImage 的 loading、等等 function。除此之外，還有 RGBImage 跟 GrayImage 繼承了 Image，這是為了讓 derived classes 取的 Image 的 w、h，還有其他 function。

多型:

在這份 project 中，我把 Image 的 LoadImage、DumpImage、Display 跟 Filter 都設為了 virtual，這是因為 Image 的 load、dump、display、filter 都跟 pixels 這個矩陣有關，然而 gray image 的 pixels 是二維矩陣，RGB image 是三維矩陣，所以處理起來會不相同，所以我把 RGB 跟 Gray 區分開來，讓程式運作時再決定要使用 RGB 的 function 還是 Gray 的 function。

3. 分享你在這份 project 中遇到了甚麼困難，又是如何解決的?

在這份 project 中，我遇到了像是 bit field filter 是甚麼、要如何實現、photo mosaic 要怎麼將大圖切塊等等問題。

Bit field filter 是甚麼:他是一個可以避免傳輸很多資料的資料結構，他會有一串位元數列，讓使用者可以傳輸 1、2、4、8、16...來選擇第 1、2、3、4、5 項，或是傳輸 3 來執行 1 跟 2，6 來執行 2 跟 4 等等。

要如何實現:利用#define Box_Filter 0b00000001 來設定 case 對應到哪一個數字，再用 uint8_t options = in_;以及 if(options&Box_Filter){}來設定使用 box filter 要做甚麼。

Photo mosaic 要怎麼把大圖切塊:我起初想到可以把大圖分割成長跟寬的最大公因數，這樣可以讓這張圖以最大的正方形小圖來表示。不過之後偶然發現助教給的小圖都是 32*32 像素的，那這樣的話就可以直接把大圖切成每 32*32 一塊。

4. 跟其他組別比起來，你覺得你這組有什麼優勢?

我認為這組相較其他組別，我這組比較實用。因為影像處理在很多領域上都可以用的到，像是:醫療影像分析、自動駕駛、人機互動等方面，然而其他組別的圖書管理系統就較為狹隘，只能使用在管理線上圖書上，而迷

宮遊戲更是只限縮在遊戲領域。

5. 心得與回饋

在這次的 project 中，我重新複習了一次本學期的 OOP 內容，並且將之活用在 project 中，雖然最後來不及做完兩個 filter，希望可以在往後的這個禮拜內將她補齊。不過我認為這次的經驗還是相當寶貴的，可以以助教給的 HackMD 筆記中淺顯易懂的方式來接觸影像處理，這肯定會為我以後打下扎實的基礎，謝謝助教們讓我有這次實作的機會。

6. Documentation / QC / QA

i. Q1:請大致解釋 'make install' 做了甚麼事情。

i. Ans: 安裝第三方依賴項：

install 目標主要是通過執行 scripts/clone_env.sh 來安裝第三方依賴項。這個腳本通常會包含下載和配置所需第三方軟體或工具的步驟，可能包括複製外部的 Git 倉庫、下載已經編譯好的二進制文件、或者編譯源代碼等。

ii. Ans:準備環境：

install 目標可以確保所有必要的依賴項都正確安裝和配置。

ii. Q2:makefile 是如何協助編譯這份 project 的?(從 inc/ src/回答)

i. Ans:inc 是用來存放 header files 的，而 src 是用來存放 cpp files 的。

CXX 和 CXXFLAGS：

CXX 定義了使用的編譯器（這裡是 g++）。

CXXFLAGS 定義了編譯器標誌，包括：

-I ./inc: 指定 inc/ 目錄為 header file 的搜索路徑，這樣編譯器可以找到 inc/ 目錄中的 header file。

其他 -I 標誌：指定了第三方庫和數據加載器的 header file 路徑。-std=c++11: 指定使用 C++11 標準編譯。

SRCDIR, OBJDIR, INCDIR：

SRCDIR 是 source code 所在的目錄 (src/)。

OBJDIR 是編譯後的對象文件存放的目錄 (obj/)。

INCDIR 是 header file 所在的目錄 (inc/)。

SRCS, OBJS, DEPS :

SRCS 使用 wildcard 函數來獲取 src/ 目錄下的所有 .cpp 文件。OBJS 使用 patsubst 函數將 .cpp 文件名替換為 .o，並將路徑從 src/ 替換為 obj/。DEPS 則生成對應的依賴文件 (.d)，這些文件用於跟蹤 .cpp 文件的依賴關係。

- iii. Q3:在 Step4 中，你如何處理邊界問題?(若大圖的長寬不是小圖的倍數，你會怎麼處理?)
 - i. Ans:我在處理邊界時，把是 32 的倍數的部份去做 photo mosaic，而多出來的部分就把他設為 0，因為在大圖中剩下那塊小於 32 像素的部分幾乎小到看不到，所以設為黑色看不太出來。
- iv. Q4:在 Step4 中，如果每張小圖的大小都不一樣，你會怎麼處理?
 - i. Ans:我會利用 Mosaic Filter 的縮小作用(在經過 Mosaic Filter 之後，圖片的像素數量會變少，也就可以縮小圖片)，但是若圖片不是正方形的話，我會先把它縮小成一邊是 32 像素，另一邊把他裁切成 32 像素。
- v. Q5:使用 valgrind 及 cppcheck 來對你的程式做動態分析與靜態分析，並秀出執行結果與報告。並解釋這兩種分析有何不同?
 - i. Ans:很抱歉助教，我最後時間不太夠，沒做出 valgrind 測試的 no memory leak 以及 cppcheck，希望可以在這個禮拜將它補完。