



SUMMER SESSION I 2014
EECS 10 WEEK5 DISCUSSION

Che-Wei Chang

OUTLINE

- Something about the discussion session
- Assignment 5
 - Digital Image Processing [80 pts + 20 bonus pts]



SOMETHING ABOUT THE DISCUSSION SESSION

- Last discussion for this course
- On this Thursday, please go to computer lab at 1:00 PM directly (2 hours lab hours)



ASSIGNMENT DISCUSSION

- Assignment 5
 - For this assignment, you will need to use **selection structure**, **repetition structure**, **function** and **array data structure**.
 - Read the assignment handout carefully
 - View the results at <http://newport.eecs.uci.edu/~youruserid>
- Digital Image Processing
 - What is the input? What is the output?
 - What algorithm to solve this problem?
 - What is the control flow for this program?
 - How to implement this program?



ASSIGNMENT 5

- A manual driven digital image processing program
- Using function calls for image input/output, processing, and testing.
 - Function declaration, function definition, function call
 - Function parameter, argument.
 - Scope of the variables
- One-week assignment. Plan the schedule of your work.
 - Lab 1: setup the working environment
 design the menu
 building up the frame of the operation function
 try 1~2 operations on the image
 - Lab 2: Complete the operation, and test your program
- Use the web browser to view your image



MENU DRIVEN DIGITAL IMAGE PROCESSING

```
eecs10@zuma.eecs.uci.edu:106 > ./PhotoLab
```

```
-----
```

- 1: Load a PPM image
- 2: Save an image in PPM and JPEG format
- 3: Change a color image to black and white
- 4: Make a negative of an image
- 5: Flip an image horizontally
- 6: Flip an image vertically
- 7: Sharpen an image
- 8: Mirror an image horizontally (Bonus)
- 9: Mirror an image vertically (Bonus)
- 10: Test all functions
- 11: Exit



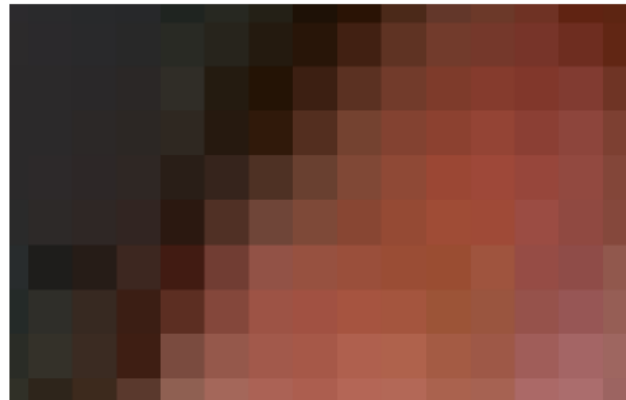
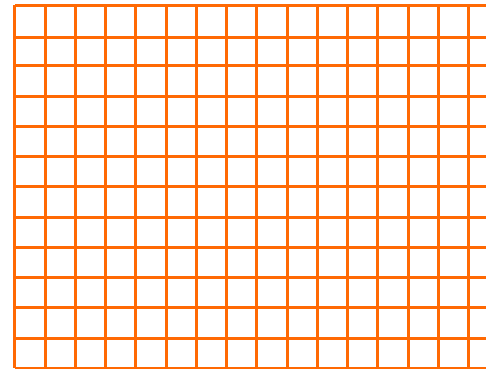
WHAT IS THE INPUT / OUTPUT

- Input: original image (in ppm format)
- Output: processed image (in ppm and jpeg format)



WHAT IS THE INPUT / OUTPUT

- How to represent a picture in computer?
 - A picture is composed of pixels
 - One color for each pixel
 - Example: 16x12



WHAT IS THE INPUT / OUTPUT

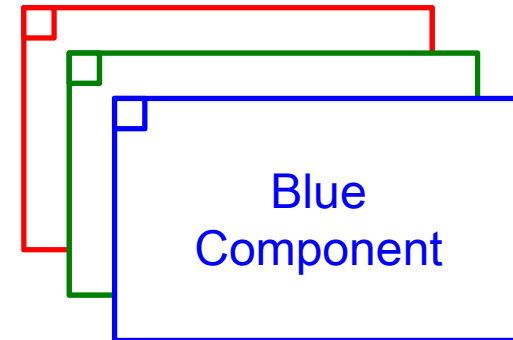
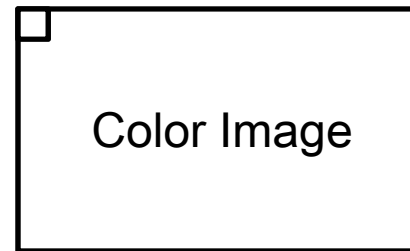
- RGB color model
 - Three component for one color

- 3-tuple (R, G, B)

- R: intensity of Red
- G: intensity of Green
- B: intensity of Blue
- For image in ppm format, the range of the intensity is [0,255], using unsigned char for each intensity

- Color examples:

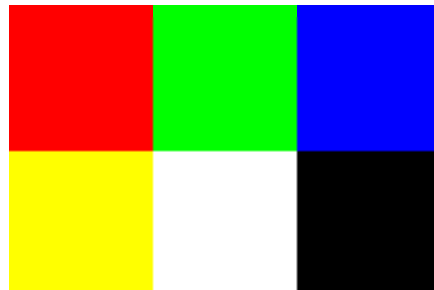
- **Red** (255, 0, 0), **Green** (0, 255, 0), **Blue** (0, 0, 255)
- **Yellow** (255, 255, 0), **Cyan** (0, 255, 255), **Magenta**(255, 0, 255)
- **White** (255, 255, 255), **black**(0, 0, 0)



WHAT IS THE INPUT / OUTPUT

- Input: original image (in ppm format)
- Output: processed image (in ppm and jpeg format)
- Any color = combination of 3 primary colors
- PPM example

- P3 (colors)
3 2 (3 columns, 2 rows)
255 (255 for max color)
255 0 0 0 255 0 0 0 255
255 255 0 255 255 255 0 0 0



WHAT IS THE INPUT / OUTPUT

- The data structure to represent a picture in computer

- Two-dimensional arrays for the intensities of each pixel

- an image of size 16x12

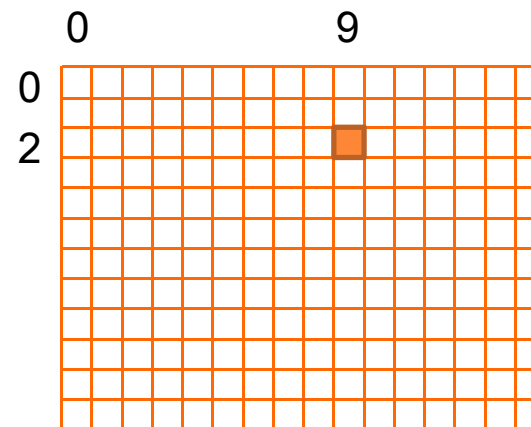
```
unsigned char R[16][12];
```

```
unsigned char G[16][12];
```

```
unsigned char B[16][12];
```

- How to access a pixel ?

- Coordinate of a pixel (x, y)
- x = number of the column
- y = number of the row
- The color of the pixel (x, y) = (R[x][y], G[x][y], B[x][y])

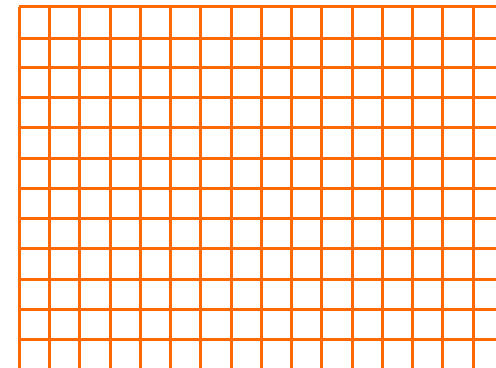


WHAT IS THE INPUT / OUTPUT

- The data structure to represent a picture in computer
 - Two-dimensional arrays for the intensities of each pixel
 - How to access every pixel ?
 - List all possible coordinates of the pixel
 - Two for-loops to scan all the pixels
 - Inner loop : fix the number of the column, iterate the pixel in the same column with different row numbers

- Outer loop : iterate all the columns

```
int x, y ;
for (x=0; x < 16; x++){
    for (y=0; y < 12; y++){
        processing on pixel(x, y);
    }
}
```



WHAT IS THE INPUT / OUTPUT

- Input: original image (in ppm format)
- Output: processed image (in ppm and jpeg format)
- Use `scanf("%s", fname)` to input file name
 - Lecture 7 slides 33 for a complete example
- Provided Functions
 - `int ReadImage (char fname[SLEN],
 unsigned char R[WIDTH][HEIGHT],
 unsigned char G[WIDTH][HEIGHT],
 unsigned char B[WIDTH][HEIGHT]) ;`
 - `int SaveImage (char fname[SLEN],
 unsigned char R[WIDTH][HEIGHT],
 unsigned char G[WIDTH][HEIGHT],
 unsigned char B[WIDTH][HEIGHT]) ;`
 - Arguments are passed to the function by reference.
 - Please refer to lecture slide 28-29 in lecture 7 for “pass by reference”



WHAT ALGORITHM

```
eecs10@zuma.eecs.uci.edu:106 > ./PhotoLab
```

```
-----
```

- 1: Load a PPM image
- 2: Save an image in PPM and JPEG format
- 3: Change a color image to black and white
- 4: Make a negative of an image
- 5: Flip an image horizontally
- 6: Flip an image vertically
- 7: Sharpen an image
- 8: Mirror an image horizontally (Bonus)
- 9: Mirror an image vertically (Bonus)
- 10: Test all functions
- 11: Exit



BLACK & WHITE



- Get the average value of the three color channels for each pixel (x,y) .
- Set $R[x][y]$, $B[x][y]$ and $G[x][y]$ to be the average value.



NEGATIVE



- Subtract $R[x][y]$, $B[x][y]$ and $G[x][y]$ from 255 and set the new value back.



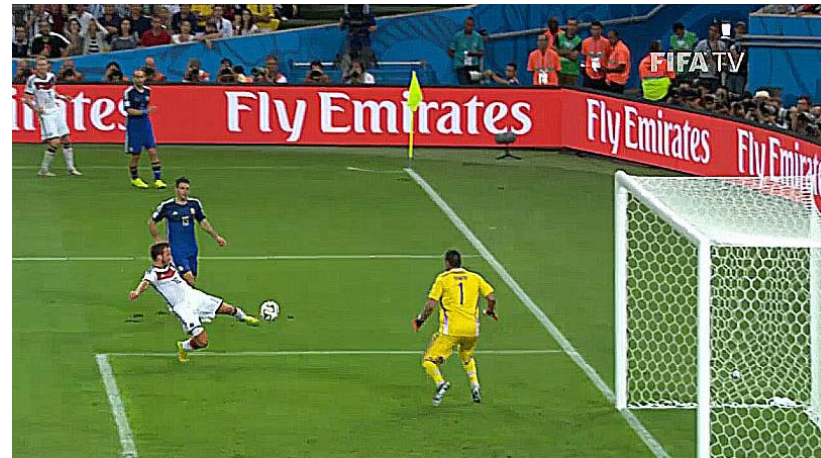
FLIP THE IMAGE



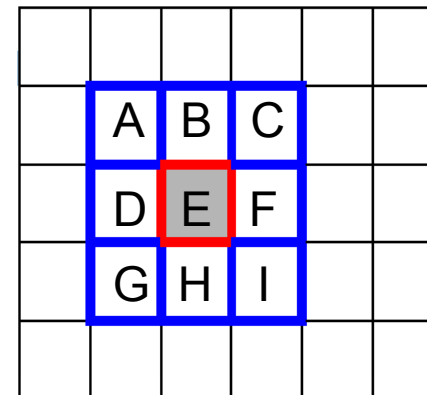
- Swap pixel (x,y) with pixel $(\text{width}-1-x, y)$
- Scan half of the picture



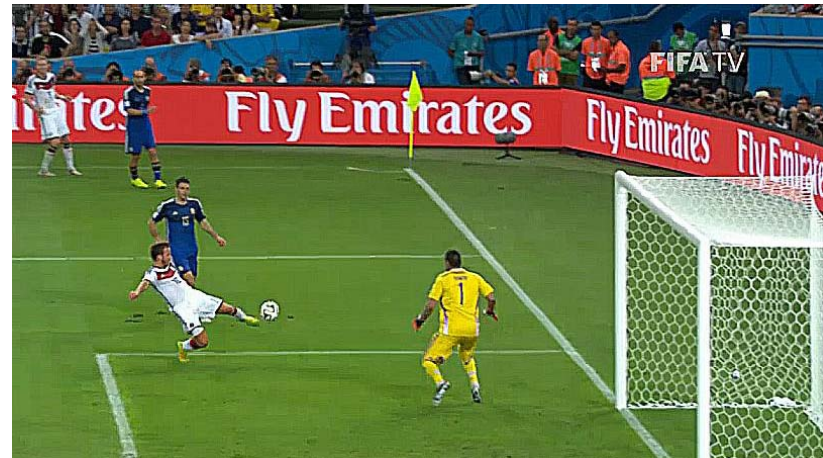
SHARPEN



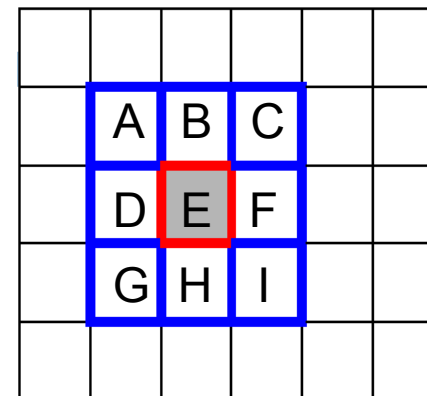
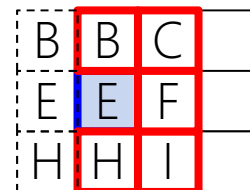
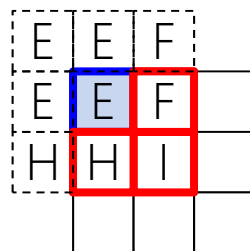
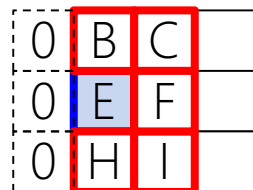
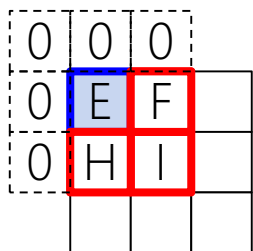
- Set the pixel's color at E with equation:
$$\text{new_E} = 9 * E - A - B - C - D - F - G - H - I$$
- Use temporary array to avoid computing with contaminated color intensities.
- Pixels on the corners and the edges have fewer neighbors.
- new_E should be in the range [0, 255]



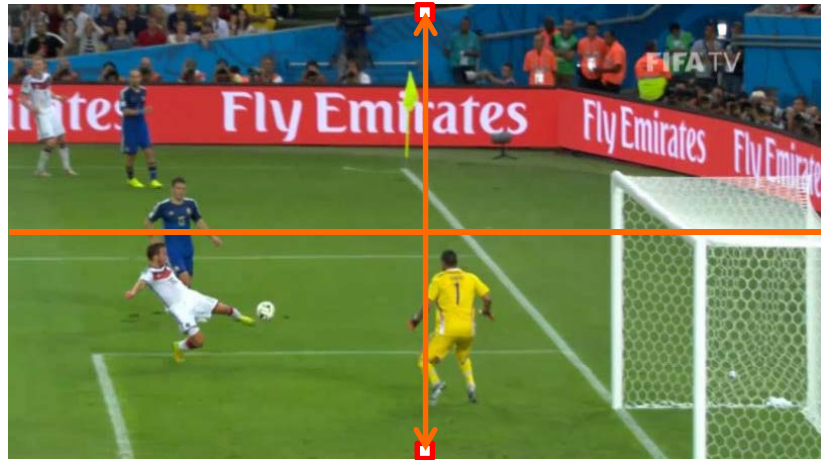
SHARPEN (CONT.)



- Pixels on the edges



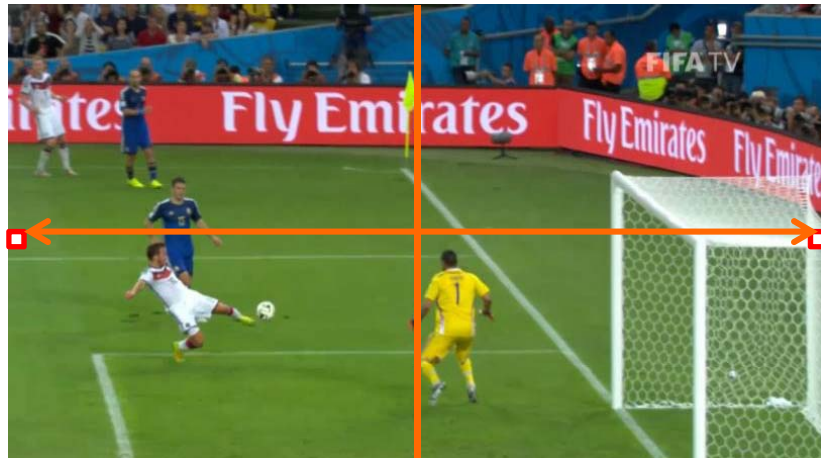
MIRROR VERTICALLY



- Replace pixel $(x, \text{height} - 1 - y)$ with pixel (x, y)
- Scan half of the picture



MIRROR HORIZONTALLY



- Replace pixel (width -1 -x, y) with pixel (x, y)
- Scan half of the picture



PROVIDED FUNCTION

```
○ #define WIDTH 810      /* Image width */
○ #define HEIGHT 450    /* image height */
○ #define SLEN 80       /* maximum length of file names */

○ int main()
○ {
○     /*
○      * Two dimensional arrays to hold the current image data
○      * One array for each color component
○      */
○     unsigned char  R[WIDTH][HEIGHT];
○     unsigned char  G[WIDTH][HEIGHT];
○     unsigned char  B[WIDTH][HEIGHT];
○     /* Please replace the following code with proper menu */
○     /* with function calls for DIP operations */
○     AutoTest(R, G, B);
○     /* end of replacing*/

○     return 0;
○ }
```



- `void Aging(unsigned char R[WIDTH][HEIGHT],
 unsigned char G[WIDTH][HEIGHT],
 unsigned char B[WIDTH][HEIGHT])`
- `{`
- `int x, y;`
- `for(y = 0; y < HEIGHT; y++)`
- `for(x = 0; x < WIDTH; x++) {`
- `B[x][y] = (R[x][y]+G[x][y]+B[x][y])/5;`
- `R[x][y] = (unsigned char) (B[x][y]*1.6);`
- `G[x][y] = (unsigned char) (B[x][y]*1.6);`
- `}`
- `}`
- `Void AutoTest(unsigned char R[WIDTH][HEIGHT],
 unsigned char G[WIDTH][HEIGHT],
 unsigned char B[WIDTH][HEIGHT])`
- `{`
- `char fname[SLEN] = "UCI_Firetrucks";`
- `char sname[SLEN];`
- `ReadImage(fname, R, G, B);`
- `Aging(R, G, B);`
- `strcpy(sname, "aging");`
- `SaveImage(sname, R, G, B);`
- `printf("Aging tested!\n\n");`
- `}`



ASSIGNMENT 5

- Test All
 - Call all the image processing in the program
 - Default input name WorldCup2014
 - Read the WorldCup2014.ppm
 - Save the processed image (ppm and jpg)
- View your result
 - eog
 - <http://newport.eecs.uci.edu/~youruserid>
- Submission
 - Run 'test all' option in the script
 - Name your files PhotoLab.c, PhotoLab.txt, PhotoLab.script

