

# EECS10 Discussion Week8

TA: Emad Arasteh  
[emalekza@uci.edu](mailto:emalekza@uci.edu)  
[eeecs10@eeecs.uci.edu](mailto:eeecs10@eeecs.uci.edu)

Office Hours: Fri, 8:00-9:00am EH 3404

University of California, Irvine

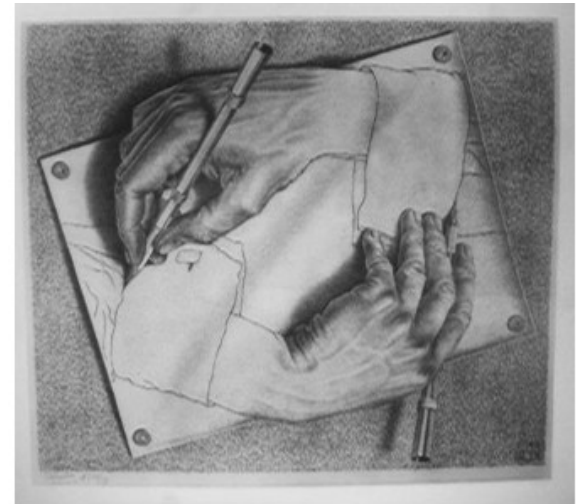


# Recursion

- “A method of solving a problem where the solution depends on solution to smaller instances of the same problem.”
- Recursion step + base case
- Use gdb to track the control flow of the program
- “He who wonders discovers that this in itself is wonder.” M.C. Escher
- Let's code sum of the first  $n$  natural numbers recursively



Russian dolls illustrate recursion [cambridgemaths.org]



M. C. Escher drawing

# Data structures

- Arrays
- Structures: **struct**
  - user-defined, composite data type
- Unions: **union**
  - user-defined, composite data type but only one member may be used at a time!
- Enumerators: **enum**
  - user-defined data type that members are an enumeration of integral constants
- Understand the meaning of *declaration*, *definition*, *instantiation* and *initialization* and use these in the context of programming properly

# Data structures

- By using **typedef**, you can write cleaner code and save extra keystrokes typing **struct** all over the code :

```
struct point_t {  
    int x;  
    int y;  
}
```

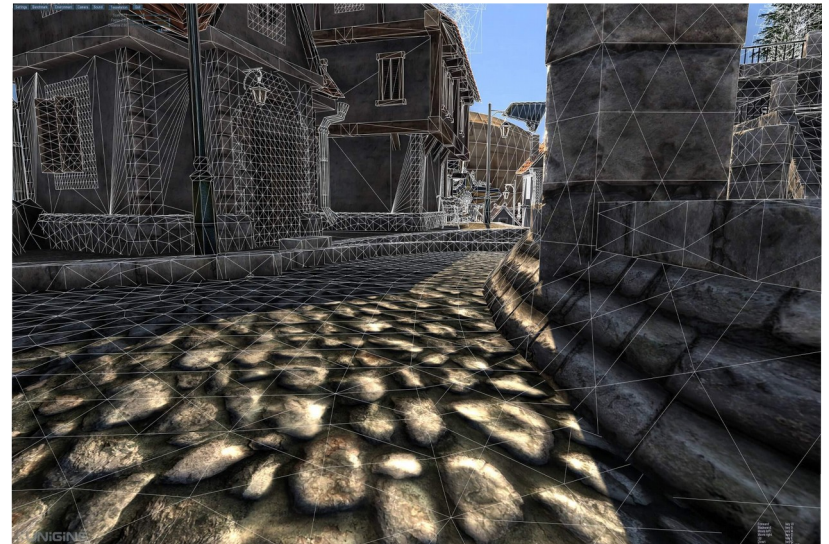
```
struct point_t p1;
```

```
typedef struct {  
    int x;  
    int y;  
} point_t
```

```
point_t p1;
```

# 3D game rendering

- Modern games can use millions of triangles to create their worlds, and every single one of those vertices will have been transformed and lit in some way.
- Let's code a simple triangle in 2D plane and find its perimeter
- Optional exercise: find the area of the triangle in 2D plane



<https://www.techspot.com/article/1857-how-to-3d-rendering-vertex-processing/>

# Assignment 7

- A menu driven digital image processing program
- Using function calls for image file handling, image processing, and testing
  - Function declaration, function definition, function call
  - Function parameters
  - Scope of the variables
- Two-week assignment: **Start early, finish early!**
  - Week1: Setup the working environment, design the user menu, try 1~2 operations on the image.
  - Week2: Complete the operations, test your program.
- Use the web browser to view your image.

# Pixels

- How to represent an image in digital computers:
  - An image is composed of picture elements aka pixels

pixel



# RGB color components

- Three components (R, G, B) are used to represent one pixel:
  - R: intensity for red color
  - G: intensity for green color
  - B: intensity for blue color
- The range of intensity for each color component in the 'library' image is values between [0 to 255] (8-bit). Therefore, we use **char** type to store these values.
- 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)
  - (255, 255, 255), **Black** (0, 0, 0)



# Image size

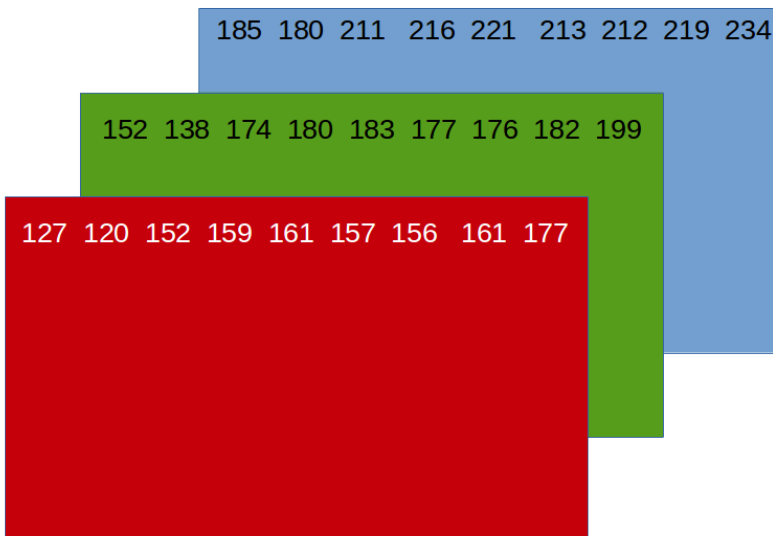
- Size of image is (640 x 480) as (width x height)



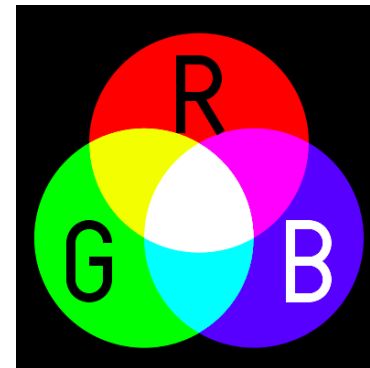
# RGB colors

- A 2-dimensional array defines intensity of each color component

```
unsigned char R[WIDTH][HEIGHT];  
unsigned char G[WIDTH][HEIGHT];  
unsigned char B[WIDTH][HEIGHT];
```



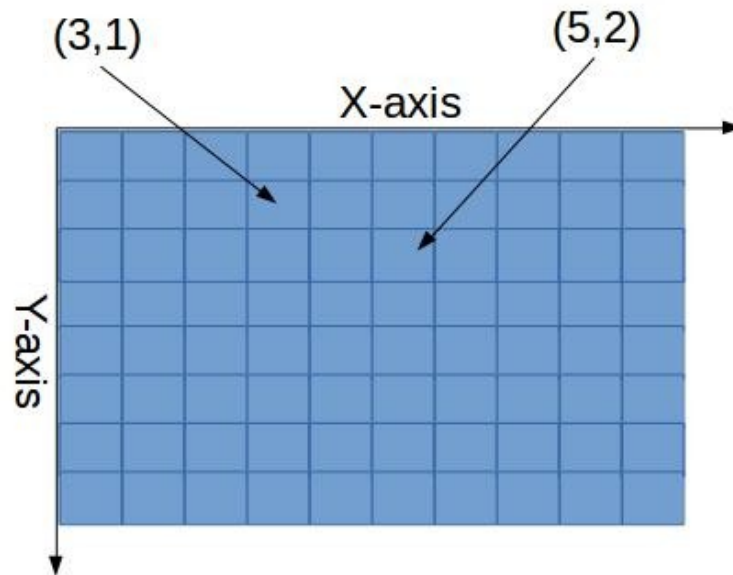
Colors intensities for red, green, blue colors taken from 'library.ppm' image by E. Arasteh



Additive color mixing [Wikipedia]

# How to manipulate an image

- First, how to access every pixel in an image?
  - By coordinate of a pixel  $(x, y)$ ,  $x$  is coordinate on the X-axis and  $y$  is the coordinate on Y-axis
  - The color tuple of the pixel at coordinate  $(x, y)$  is :  
 $(R[x][y], B[x][y], G[x][y])$



# How to manipulate an image

- You can use nested for loops to manipulate pixels of an image:

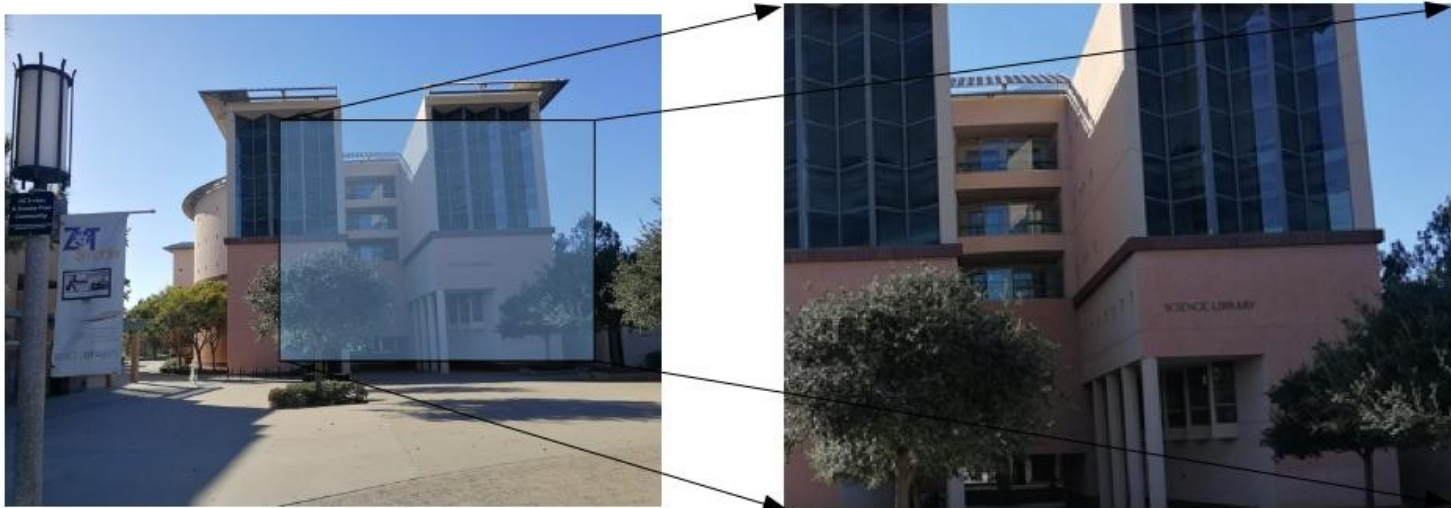
```
for (int y = 0; y < HEIGHT; y++) {  
    for (int x = 0; x < WIDTH; x++) {  
        operate on pixel(x,y)  
    }  
}
```

# Image processing functions

- Black and white
  - For each pixel at coordinate  $(x,y)$ , compute the average of three color channels
  - Set the new value for all three color channels equal to the average
- Negative
  - Subtract  $R[x][y]$ ,  $G[x][y]$  and  $B[x][y]$  from the max intensity value (255) and update the pixel value
- Flip horizontally
  - Hint: scan only half of the image
- Mirror horizontally
  - Hint: scan only half of the image

# Image processing functions

- Zoom-in
  - Hint: arrows are pointing to coordinates in the new image



# Image processing functions

- Sharpen
  - Slide the filter on the image and compute the weighted sum for each pixel
  - Watch out for pixel values greater than max intensity (255) or less than min intensity (0)
  - Watch out for pixel coordinates at the border of the image
- Exchange RG
  - Swap intensity of red and green channels
- Add noise
  - Randomly generate coordinates (2 random number for x and y)
  - Set the intensity values to maximum (255, 255, 255) or minimum (0, 0, 0) alternatively for those noisy pixels

# Image processing functions (bonus)

- Overlay
  - Pick either a pixel from the original image or a pixel from the overlay image depending on the background pixel intensity
- Add borders
  - Turn the pixels on the border into a specific color (defined by the user)



# AutoTest()

- Test your program
  - AutoTest() function
    - Call all functions together in the program
    - Be careful with the arguments for each functions
    - Sample function calls are listed in the assignment
- Global constants
- Scope of the variables
- Pass by reference when using array parameters
- Function prototypes mentioned in the assignment are very helpful hints