DSP Lab. Week 3
My Image

Kyuheon Kim
Media Lab. Rm567
kyuheonkim@khu.ac.kr
Last update : August 24, 2019

❖ Popular file formats

TABLE 3.1: Macromedia Director file formats.

File import					File export		Native
Image	Palette	Sound	Video	Animation	Image	Video	
BMP, DIB,	PAL	AIFF	AVI	DIR	BMP	AVI	DIR
GIF, JPG,	ACT	AU	MOV	FLA		MOV	DXR
PICT, PNG,		MP3		FLC			EXE
PNT, PSD,		WAV		FLI			
TGA, TIFF,				GIF			
WMF				PPT			

❖ 1 Bit images

- Which is consisted of on and off bits only (0 or 1).
- Which is also referred to as a binary image
- A 640 x 480 monochrome image requires 38.4 kilobytes of storage (= 640 x 480/8).

• 8 bit gray-level images

- Where Each pixel is represented by a single byte.
- Thus, each pixel has a gray value between 0 and 255.
- A frame buffer is required to store this image array.
- A 640 x 480 grayscale image requires 300 kilobytes of storage (640 x 480 = 307,200).

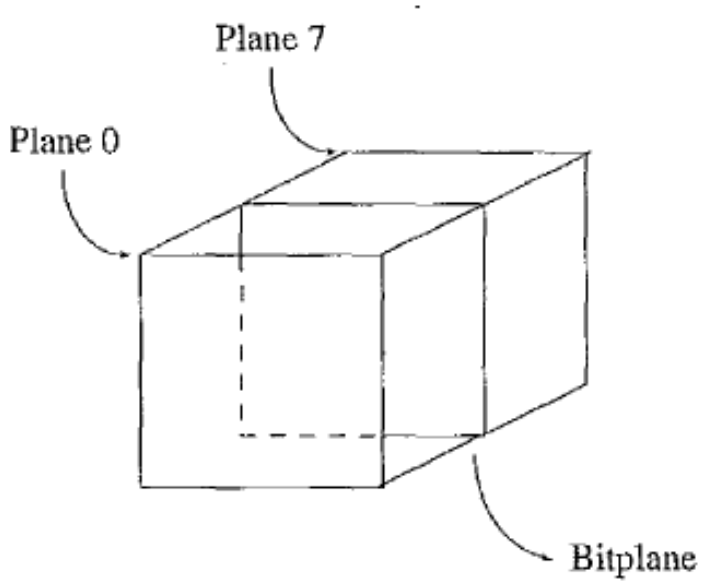
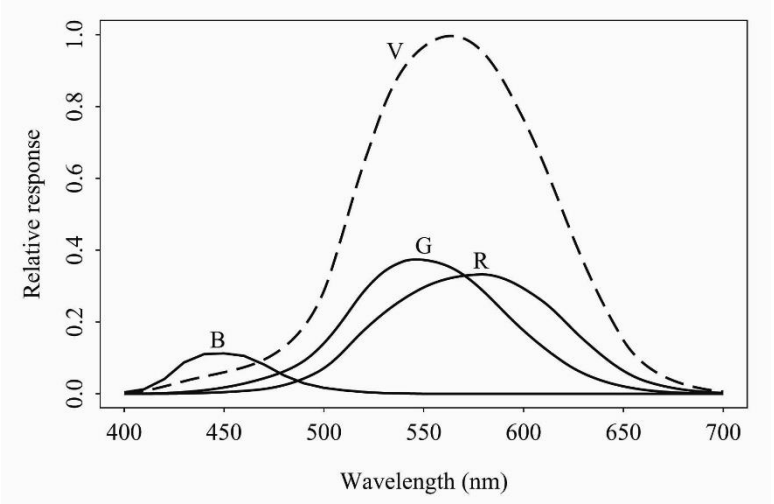


FIGURE 3.2: Bitplanes for 8-bit grayscale image.

- In human vision, the three kinds of cones are most sensi tive to red **(R),** green (G), and blue **(B)** light.
- it seems likely that the brain makes use of dijferences R-G, G~B, and B-R, as well as combining all of R, G, and B into a high-light-level achromatic channel.
- The proportions of R, G, and B cones are different, which are present in the ratios 40:20: 1 like 2R + G + B/20.



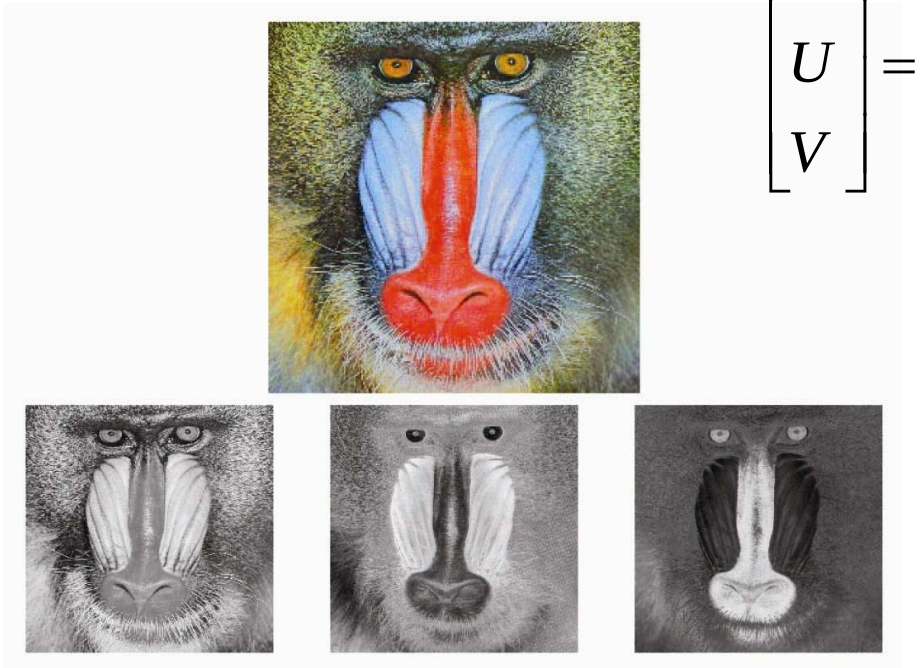
Cone sensitivities: R, G, and B co nes, and luminous-efficiency curv e YeA)

❖ YUV color model

- Y: Luminance (Brightness)
- U & V: colorfulness scale (chrominance)

$$U = B' - Y' \qquad V = R' - Y'$$

$$\begin{bmatrix} Y' \\ U \\ V \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.144 \\ -0.299 & -0.587 & 0.866 \\ 0.701 & -0.587 & -0.144 \end{bmatrix} \begin{bmatrix} R' \\ G' \\ B' \end{bmatrix}$$



(a) original color image; (b) Y; (c) U; (d) V.

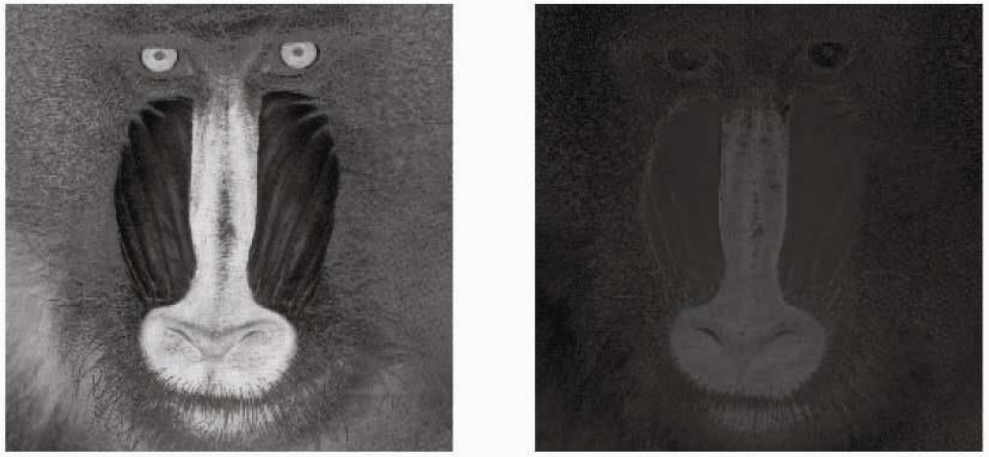
❖ YIQ

- YIQ is used in NTSC color TV broadcasting

$$I = 0.492111(R' - Y')\cos 33° - 0.877283(B' - Y')\sin 33°$$

$$Q = 0.492111(R' - Y')\sin 33° + 0.877283(B' - Y')\cos 33°$$

$$\begin{bmatrix} Y' \\ I \\ Q \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.144 \\ 0.595879 & -0.274133 & -0.321746 \\ 0.211205 & -0.523083 & 0.311878 \end{bmatrix} \begin{bmatrix} R' \\ G' \\ B' \end{bmatrix}$$



(a) I
(b) Q components of color image.

Image

❖ 영상은 unsigned char *에 저장한다.

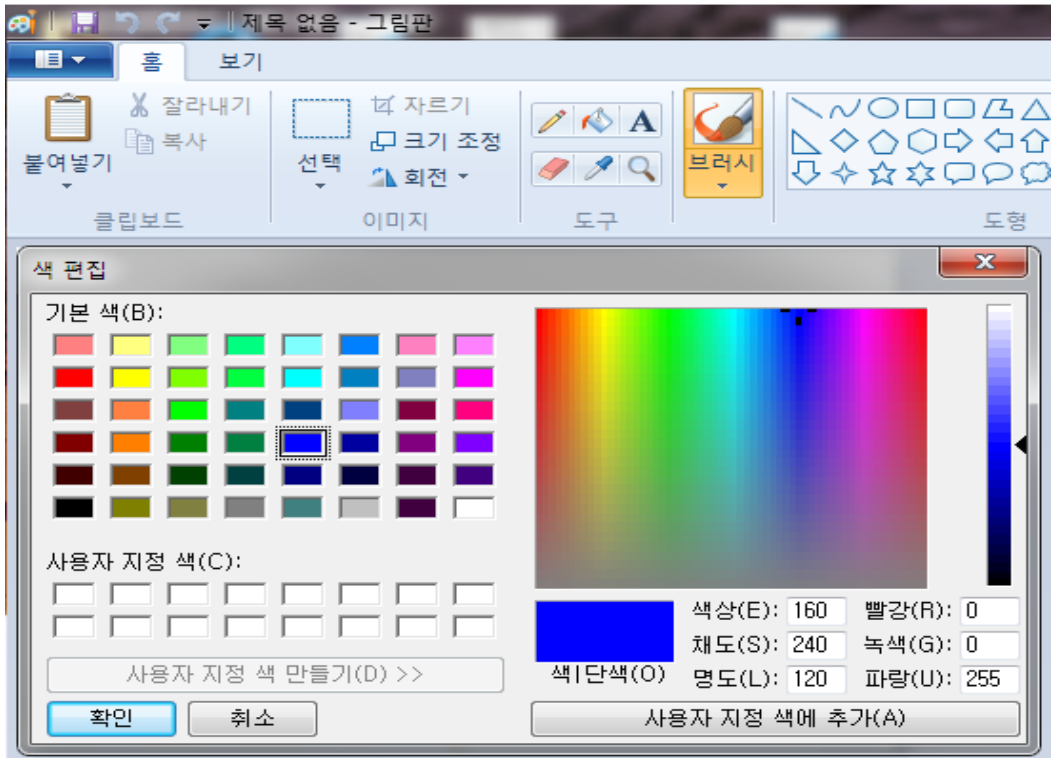
1 byte: 0~255(black~white)
W: width, H: height
// Width W Height H
#define W 300
#define H 200

❖ Raw ➔ rgb file

unsigned char *R, *G, *B;
R = new unsigned char[W*H];
G = new unsigned char[W*H];
B = new unsigned char[W*H];

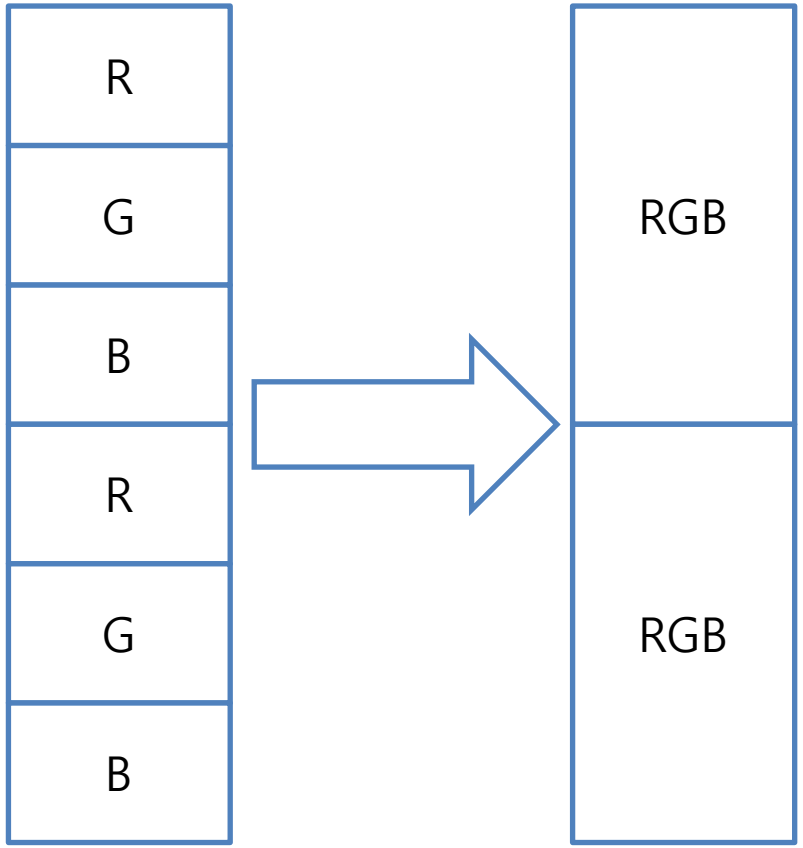
❖ Raw 동영상

연속적으로 저장하면 된다.



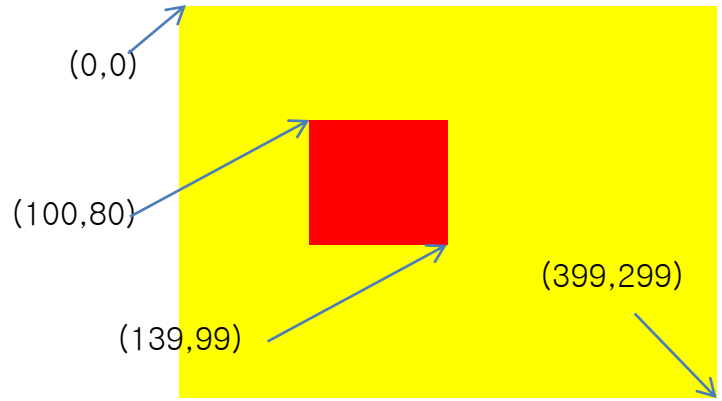
Image

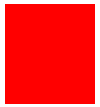
```
unsigned char *R, *G, *B, *RGB;
R = new unsigned char[W*H]; //H는 영상 세로 사이즈
G = new unsigned char[W*H]; //W는 영상 가로 사이즈
B = new unsigned char[W*H]; //W*H는 영상 사이즈
RGB = new unsigned char[3*W*H];
```




```
for(i=0; i<H; i++) //H는 영상 세로 사이즈
  for(j=0; j<W; j++){ //W는 영상 가로 사이즈
    individual_idx = W*i+j;
    compositive_idx = (W*i+j)*3;
    RGB[compositive_idx]         = R[individual_idx];
    RGB[compositive_idx + 1]     = G[individual_idx];
    RGB[compositive_idx + 2]     = B[individual_idx]; }
```

Image





(R,G,B)=(255,0,0)

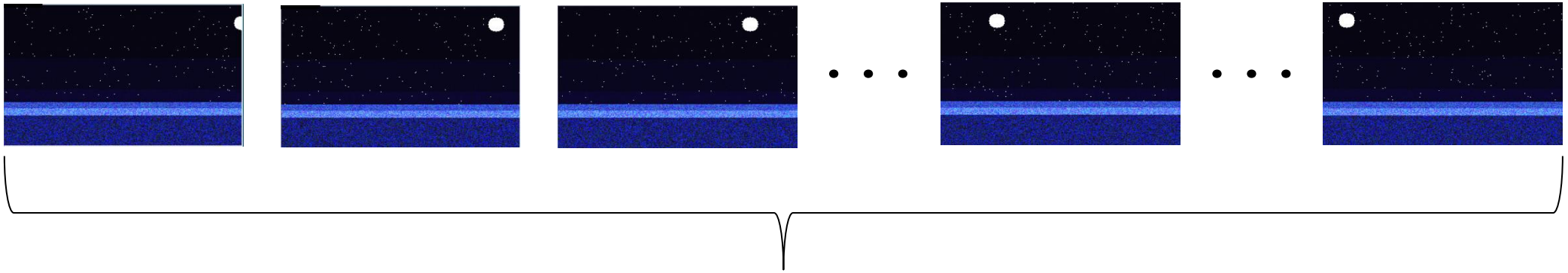
(R,G,B)=(255,255,0)

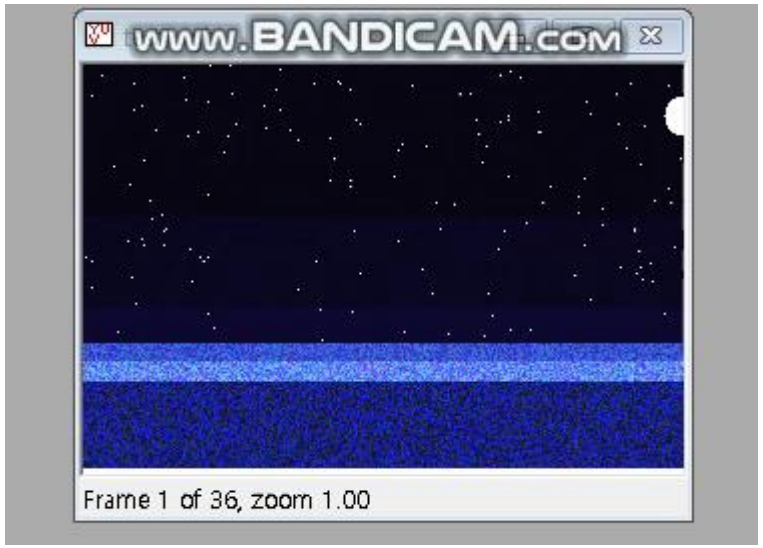
```
unsigned char R[120000], G[120000], B[120000], RGB[360000];
ofstream fff("flag.rgb", ios::out | ios::binary); // binary 파일로 생성

//전체 노랗게
for(i=0;i<120000;i++){ R[i] = G[i] = 255; *(B+i) = 0;}
// 가운데 빨간 사각형
for(i=80; i<100; i++) for(j=0; j<40; j++)
{ individual_idx = i*400+j+100;    R[individual_idx] = 255; *(G+individual_idx) = *(B+individual_idx) = 0; }
// 파일에 RGB 묶어서 넣기 (interlaced)
for (i = compositive_idx = 0; i < 120000; i++, compositive_idx += 3) {
        RGB[compositive_idx] = R[i]; RGB[compositive_idx+1] = G[i]; RGB[compositive_idx+2] = B[i];  }
fff.write((const char*)RGB, 36000);
```

Video

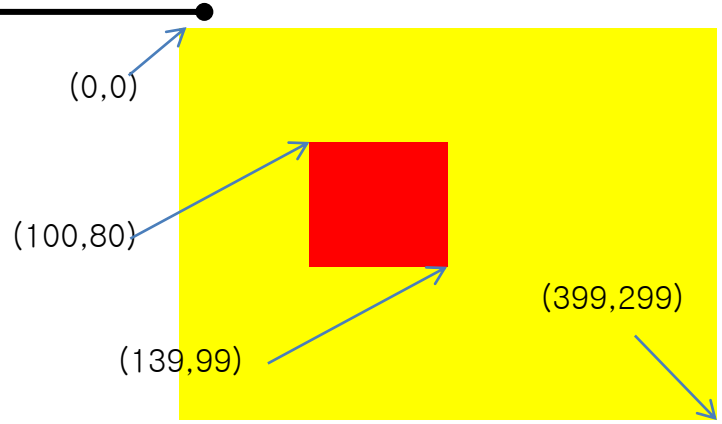
❖ 동영상

이미지를 연속적으로 저장하면 된다.





Video



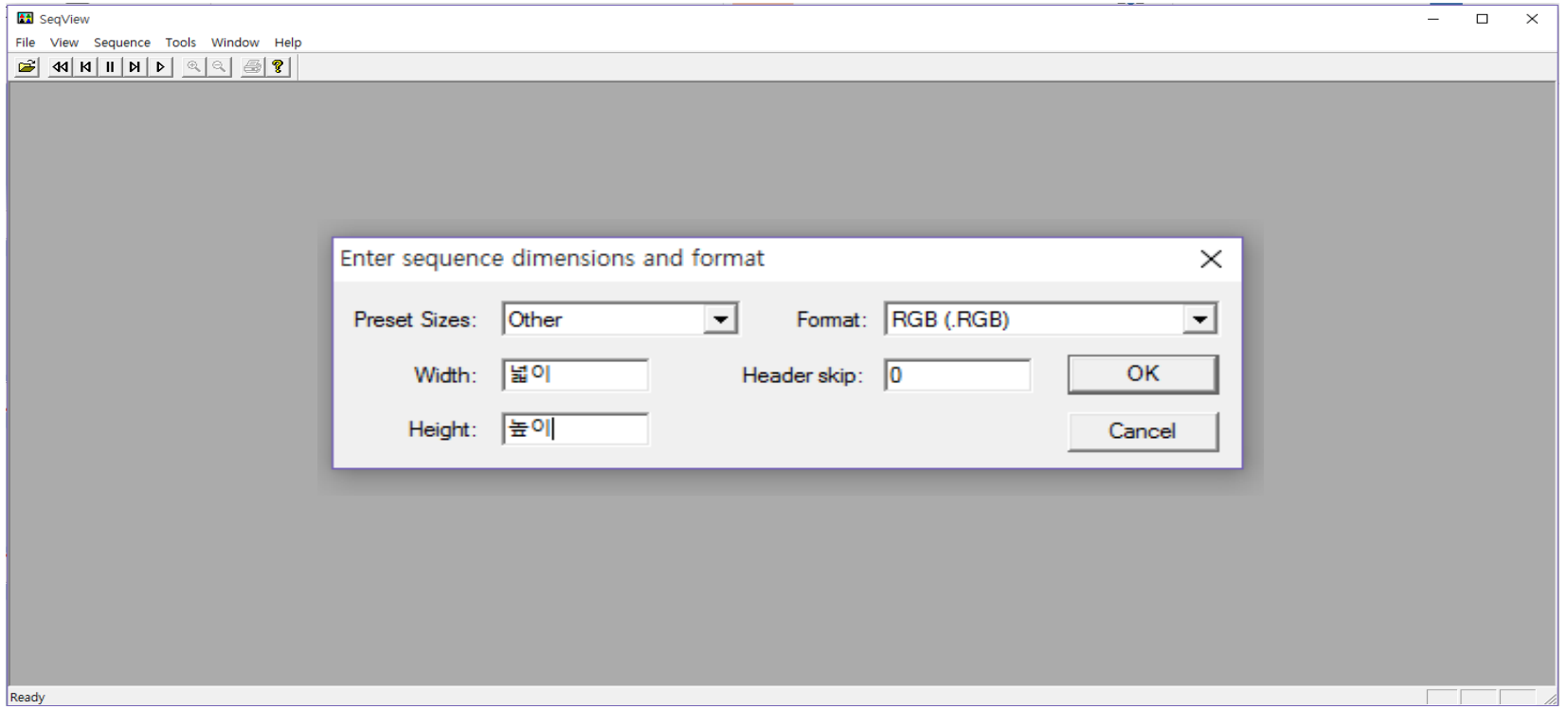
(R,G,B)=(255,0,0)

(R,G,B)=(255,255,0)

```
unsigned char R[120000], G[120000], B[120000], RGB[360000];
ofstream fff("flag.rgb", ios::binary); // binary 파일로 생성
For(m=100; m<200; m += 5){// 동영상 만들 때
    //전체 노랗게
    for(i=0;i<120000;i++){ R[i] = G[i] = 255; *(B+i) = 0;}
    // 가운데 빨간 사각형
    for(i=80; i<100; i++) for(j=0; j<40; j++)
    { individual_idx = i*400+j+m;   R[individual_idx] = 255; *(G+ individual_idx) = *(B+ individual_idx) = 0; }
    // 파일에 RGB 묶어서 넣기 (interlaced)
    for (i = compositive_idx = 0; i < 120000; i++, compositive_idx += 3) {
        RGB[compositive_idx] = R[i]; RGB[compositive_idx +1] = G[i]; RGB[compositive_idx +2] = B[i];  }
    fff.write((const char*)RGB, 360000);
}
```

Video

- YUV viewer (RGB 형식의 raw file을 볼 때)
  - 프로그램 내에서 파일을 저장할 때 " *.rgb"로 확장자를 ". rgb"로 한다.
    - YUVSequenceViewer.exe 실행
    - File→ Open→ 모든파일 → 저장된 .rgb 선택.
    - image의 넓이, 높이 입력 **format은 반드시 "RGB"**



C-program (Sinusoidal wave)

```
1.  #include <iostream> // cout, cin
2.  #include <fstream>  // ifstream, ofstream 파일 라이브러리
3.  using namespace std;
4.  #define Width 400
5.  #define Height 300
6.  #define WH 120000

7.  int main(){
8.          int i, j, m, individual_idx, compositive_idx;
9.          ofstream fff( " video.rgb " , ios::out | ios::binary);    // 출력 파일 선언.바이너리
10.         unsigned char R[WH], G[WH], B[WH], RGB[3* WH]; // 이미지 저장 공간

11.         for(m=100; m<200; m += 5){// 동영상 만들 때
12.           //전체 노랗게
13.              for(i=0;i< WH;i++){ R[i] = G[i] = 255; *(B+i) = 0;}
14.           // 가운데 빨간 사각형
15.              for(i=80; i<100; i++) for(j=0; j<40; j++)
16.                {individual_idx = i*400+j+m;    R[individual_idx] = 255; *(G+ individual_idx) = *(B+ individual_idx) = 0; }
17.           // 파일에 RGB 묶어서 넣기 (interlaced)
18.         for (i = compositive_idx = 0; i < WH; i++, compositive_idx += 3) {
19.              RGB[compositive_idx] = R[i]; RGB[compositive_idx +1] = G[i]; RGB[compositive_idx +2] = B[i];  }
20.         fff.write((const char*)RGB, 3* WH); // 동영상의 한 프레임 저장
21.         }

22.         outFile.close();
23.         return 0;
24. }
```

main Function

Week 3 assignment

❖ **움직이는 RGB 영상을 만들어 보라.** (크기: H = 200, W = 300, 확장자는 *.rgb)

- 자유 주제
- YUV Viewer 프로그램을 통해 5~8장의 프레임을 캡처한 후, 보고서에 첨부

Week 3 assignment

"KLAS에 제출할 때 다음 사항을 꼭 지켜주세요"

1. 파일명 : "Lab00_요일_대표자이름.zip"
Ex) Lab01_목_홍길동.zip **(압축 툴은 자유롭게 사용)**

2. 제출 파일 **(보고서와 프로그램을 압축해서 제출)**
   - 보고서 파일 (hwp, word): 이름, 학번, 목적, 변수, 알고리즘(순서), 결과 분석, 느낀 점
   - 프로그램

DSP 실험 보고서

과제 번호	Lab01	제출일	2019.09.02
학번/이름	20xxxxxxxx 홍길동 20xxxxxxxx 푸리에		

1. 목적	
2. 변수	
3. 알고리즘	
4. 결과분석	
5. 느낀 점