**텀프로젝트 결과보고서**

2018036029 김민정

목차

1. 사용한 함수 설명
2. 알고리즘 설명
3. 16X16 DB 이미지를 사용해 모자이크하기 : Mosaic16X16( )
4. 8X8 DB 이미지를 사용해 모자이크하기 : Mosaic8X8( )
5. 4X4 DB 이미지를 사용해 모자이크하기 : Mosaic4X4( )
6. 16X16 DB 이미지를 사용해 랜덤 모자이크하기

: RANDOM\_Mosaic16X16( )

1. 입력 / 출력 모자이크 영상 보기 : main( )
2. 결과 영상
3. 입력영상
4. 출력영상1) 모자이크16\*16
5. 출력영상2) 모자이크8\*8
6. 출력영상3) 모자이크4\*4
7. 출력영상4) 랜덤 모자이크16\*16

Ⅰ. 사용한 함수 설명

① 이미지 사이즈를 반으로 줄이는 함수

**MakeHalf**(int\_rgb\*\*\* db\_imgs32, int num, int db\_height, int db\_width, int\_rgb\*\*\* db\_imgs16)

원본 db이미지를 입력으로 받고 결과로 원본 db이미지의 사이즈를 반으로 줄인 것을 받음.

② 배경 영상의 모자이크 범위 내의 r,g,b 각 값의 평균 구하기

**R\_Average**(int a, int b, int size, int\_rgb\*\* original\_img)

**G\_Average**(int a, int b,int size, int\_rgb\*\* original\_img)

**B\_Average**(int a, int b, int size, int\_rgb\*\* original\_img)

③ db이미지들에 대해 r,g,b 각 값의 평균 구하기

**average\_rgb**(int\_rgb\*\*\* db\_imgs, int db\_height, int db\_width, int\_rgb db\_rgb\_average[DB\_NUM])

④ 배경 영상의 모자이크 내 범위의 r,g,b 값과 비슷한 db이미지 영상찾아 해다하는 index return함.

**Find\_Match\_Index**(int\_rgb db\_rgb\_average[DB\_NUM], int r\_average, int g\_average, int b\_average)

: 2번과 3번 함수를 통해 구한 db영상 i 번째 영상의 r,g,b 평균과 배경 영상의 모자이크 범위에 해당하는 r,g,b 평균의 차를 구하고 절댓값 취해 그 차가 가장 작은 db이미지 인덱스 구하기

⑤ 출력영상 만들기

**OUT\_IMAGE**(int MosaicSize, int height, int width, int\_rgb db\_rgb\_average[DB\_NUM],

int\_rgb\*\* db\_imgs[DB\_NUM], int\_rgb\*\* original\_img, int\_rgb\*\* img\_out)

: 배경 영상을 모자이크 크기 간격을 반복하는 for문 작성

각 변수들을 선언&초기화

여기서 db\_img\_y는 db이미지의 행을 반복할 변수이고, db\_img\_x는 db이미지의 열을 반복할 변수이다.

⑤-1 랜덤하게 출력영상 만들기

**OUT\_IMAGE\_RANDOM**(int MosaicSize, int height, int width, int\_rgb db\_rgb\_average[DB\_NUM], int\_rgb\*\* db\_imgs[DB\_NUM], int\_rgb\*\* original\_img, int\_rgb\*\* img\_out)

⑥ 16\*16의 db이미지로 모자이크하기

**Mosaic16X16()**

: MakeHalf()를 한번 사용해 32\*32 db이미지를 16\*16으로 줄인다.

OUT\_IMAGE() 이 함수로 모자이크 한 img\_out 받아오기

ColorImageShow("모자이크16\*16", img\_out, height, width)를 사용해 img\_out 이미지 출력하기

⑦ 8\*8의 db이미지로 모자이크하기

**Mosaic8X8()**

: MakeHalf()를 두번 사용해 32\*32 db이미지를 16\*16으로 줄인후 다시 16\*16을 8\*8로 줄인다.

OUT\_IMAGE() 이 함수로 모자이크 한 img\_out 받아오기

ColorImageShow("모자이크8\*8", img\_out, height, width)를 사용해 img\_out 이미지 출력하기

⑧ 깔끔한 코드를 위해 ③rgb 색상평균, ⑤img\_out 만드는 과정을 함수화.

**set\_img**(int\_rgb\*\* db\_imgs16[DB\_NUM], int db\_height16, int db\_width16,

int\_rgb db\_rgb\_average[DB\_NUM], int height, int width, int\_rgb\*\* original\_img, int\_rgb\*\* img\_out)

: db\_height가 10보다 크면 16\*16 모자이크 이미지임을 알고 출력화면 상단에 [모자이크16\*16] 띄움.

5보다 크고 10 이하이면 , 8\*8 모자이크 이미지임을 알고 출력화면 상단에 [모자이크8\*8] 띄움.

5이하이면 4\*4 모자이크 이미지임을 알고 출력화면 상단에 [모자이크4\*4] 띄움.

⑧-1 랜덤버전으로 만들기

**set\_img\_RANDOM**(int\_rgb\*\* db\_imgs16[DB\_NUM], int db\_height16, int db\_width16, int\_rgb db\_rgb\_average[DB\_NUM], int height, int width, int\_rgb\*\* original\_img, int\_rgb\*\* img\_out)

⑨ 메인문

**main( )**

: 먼저 원본 사진을 출력하고 Mosaic16X16( ), Mosaic8X8( ), Mosaic8X8( ) 함수를 호출해 16\*16 db를 사용한 모자이크와 8\*8 db를 사용한 모자이크와 4\*4 db를 사용한 모자이크를 출력함. 또한 랜덤한 위치에 모자이크한 결과 영상 출력.

Ⅱ. 알고리즘 설명

ⅰ. 16X16 DB 이미지를 사용해 모자이크하기 : Mosaic16X16( )

① Mosaic16X16()안에서 사용할 db이미지 위치, db이미지 접두사, db이미지 높이, db이미지 너비, db16\*16 이미지 높이, db16\*16이미지 너비, 배경영상의 높이와 너비 등 필요한 변수들을 선언한다.

char db\_dir[300] = "db 파일경로 작성";

char db\_fname\_prefix[100] = "dbs";

char db\_fname[300];

int height, width;

int db\_height, db\_width, db\_height16, db\_width16;

② 32\*32크기의 db영상을 저장할 구조체형 배열, 16\*16크기의 db영상을 저장할 구조체형 배열,

각 데이터 베이스 이미지의 평균 RGB를 담을 구조체형 배열, 배경이미지를 읽을 구조체, 결과영상을 구조체 등 필요한 구조체 배열 선언 및 구조체 메모리 할당한다.

int\_rgb\*\* db\_imgs32[DB\_NUM];

int\_rgb\*\* db\_imgs16[DB\_NUM];

int\_rgb db\_rgb\_average[DB\_NUM];

char original\_img\_name[100] = "스파이더맨.jpg";

int\_rgb\*\* original\_img = ReadColorImage(original\_img\_name, &height, &width);

int\_rgb\*\* img\_out = (int\_rgb\*\*)IntColorAlloc2(height, width);

③ 32\*32사이즈의 db이미지를 MakeHalf함수를사용해 16\*16으로 줄이고 16\*16db의 높이와 너비를

32\*32 db의 높이와 너비의 반으로 줄인다.

MakeHalf(db\_imgs32, DB\_NUM, db\_height, db\_width, db\_imgs16);

: db\_imgs32 의 영상크기를 4픽셀을 하나의 픽셀로 합쳐 줄인후 db\_imgs16 배열에 저장

db\_height16 = db\_height/2;

db\_width16 = db\_width/ 2;

④ 이미지를 매칭 시키고 가장 적절한 이미지를 출력하기 위한 set\_img( ) 함수 호출한다.

set\_img( db\_imgs16, db\_height16, db\_width16, db\_rgb\_average, height, width, original\_img, img\_out);

: db\_height가 10보다 크기 때문에 16\*16 모자이크 이미지임을 알고 출력화면 상단에

[모자이크16\*16] 띄운다.

⑤ rgv 색상평균 구하는 average\_rgb()함수 호출한다.

average\_rgb(db\_imgs16, db\_height16, db\_width16, db\_rgb\_average);

⑤-1) 각 db영상의 r,g,b 의 평균값 구한다. db\_rgb\_average[]배열의 .r/.g/.b에 평균을 저장.

void average\_rgb(int\_rgb\*\*\* db\_imgs, int db\_height, int db\_width, int\_rgb

db\_rgb\_average[DB\_NUM])

{

for (int i = 0; i < DB\_NUM; i++) {

int db\_r\_sum = 0;

int db\_g\_sum = 0;

int db\_b\_sum = 0;

for (int y = 0; y < db\_height; y++) {

for (int x = 0; x < db\_width; x++) {

db\_r\_sum += db\_imgs[i][y][x].r;

db\_g\_sum += db\_imgs[i][y][x].g;

db\_b\_sum += db\_imgs[i][y][x].b;

}

}

db\_rgb\_average[i].r = (db\_r\_sum / (db\_height \* db\_width));

db\_rgb\_average[i].g = (db\_g\_sum / (db\_height \* db\_width));

db\_rgb\_average[i].b = (db\_b\_sum / (db\_height \* db\_width));

}

}

⑥ img\_out 구하기 위한 OUT\_IMAGE() 함수 호출한다.

OUT\_IMAGE(db\_height16, height, width, db\_rgb\_average, db\_imgs16, original\_img, img\_out);

⑥-1) 배경영상의 모자이크 크기내 범위의 r,g,b 의 평균값 구하기

R\_Average(int i, int j, int size, int\_rgb\*\* original\_img)

G\_Average(int i, int j,int size, int\_rgb\*\* original\_img)

B\_Average(int i, int j, int size, int\_rgb\*\* original\_img)

각 함수에서 구한 각 값을 r\_average, g\_average, b\_average 값을 리턴하여 OUT\_IMAGE 함수 내에서 받아온다.

int r\_average = R\_Average(i, j, MosaicSize, original\_img);

int g\_average = G\_Average(i, j, MosaicSize, original\_img);

int b\_average = B\_Average(i, j, MosaicSize, original\_img);

⑥-2) 배경영상의 모자이크 크기내 범위의 r.g.b 값과 가장 비슷한 db이미지 찾기

: ⑤-1)db영상의 i 번째 영상 r/g/b 값의 평균 - ⑥-1)배경영상의 모자이크 범위에 해당하는 평균 r/g/b 값을 계산 🡪 절대값 취하기 🡪 가장 작은 값의 인덱스 구하기 🡪 return 하기

int Find\_Match\_Index(int\_rgb db\_rgb\_average[DB\_NUM], int r\_average, int g\_average, int b\_average) {

int match\_index = -1; // index를 비교하기위해 -1 값으로 초기화

int difference = -1; // 차이값을 비교하기위해 -1 값으로 초기화

for (int a = 0; a < DB\_NUM; a++) {

int r\_difference, g\_difference, b\_difference ;

r\_difference = abs(db\_rgb\_average[a].r - r\_average);

g\_difference = abs(db\_rgb\_average[a].g - g\_average);

b\_difference = abs(db\_rgb\_average[a].b - b\_average);

if (difference == -1) {

match\_index = a;

difference = r\_difference+ g\_difference+ b\_difference;

match\_index = a;

}

else {

if (difference > (r\_difference + g\_difference + b\_difference)) {

difference = r\_difference + g\_difference + b\_difference;

match\_index = a;

}

}

}

return match\_index; //가장 비슷한 영상의 INDEX값을 return 하기.

}

⑥-3) 구한 match\_index 값을 OUT\_IMAGE( ) 함수 내 match\_index에 넣기.

int match\_index = Find\_Match\_Index(db\_rgb\_average, r\_average, g\_average, b\_average);

⑥-4) 구한 match\_index을 토대로 img\_out[y][x]만들기

int db\_img\_y = 0; for (int y = i - MosaicSize; y < i; y++) {

int db\_img\_x = 0;

for (int x = j - MosaicSize; x < j; x++) {

mg\_out[y][x] = db\_imgs[match\_index][ db\_img\_y][ db\_img\_x];

// img\_out[y][x]픽셀에 db\_imgs[match\_index]의 db\_y열 db\_x행 픽셀의 값을 넣어주기

db\_img\_x++;

}

db\_img\_y++;

}

ⅱ. 8X8 DB 이미지를 사용해 모자이크하기 : Mosaic8X8( )

① 앞의 Mosaic16X16( )와 같게 진행하되, 8\*8에 해당하는 변수,구조체배열을 추가 선언.

int db\_height8, db\_width8;

int\_rgb\*\* db\_imgs8[DB\_NUM];

② 32\*32사이즈의 db이미지를 MakeHalf함수를 두번 이용해 16\*16으로 한번 줄이고 다시 8\*8로

줄인다.

for (int i = 0; i < DB\_NUM; i++)

db\_imgs8[i] = (int\_rgb\*\*)IntColorAlloc2(db\_height / 4, db\_width / 4);

db\_height8 = db\_height/4;

db\_width8= db\_width/4;

MakeHalf(db\_imgs16, DB\_NUM, db\_height16, db\_width16, db\_imgs8);

③ 이미지를 매칭 시키고 가장 적절한 이미지를 출력하기 위한 set\_img() 함수 호출

set\_img( db\_imgs8, db\_height8, db\_width8, db\_rgb\_average, height, width, original\_img,

img\_out);

: db\_height가 10보다 작고 5보다 크기때문에 8\*8 모자이크 이미지임을 알고 출력화면 상단에

[모자이크8\*8] 띄운다.

④ 이후 과정들에 대해선 Mosaic16X16()와 동일함.

ⅲ. 4X4 DB 이미지를 사용해 모자이크하기 : Mosaic4X4( )

① 앞의 Mosaic8X8( )와 같게 진행하되, 4\*4에 해당하는 변수,구조체배열을 추가 선언.

int db\_height4, db\_width4;

int\_rgb\*\* db\_imgs4[DB\_NUM];

② 32\*32사이즈의 db이미지를 MakeHalf함수를 세번 이용해 16\*16으로 한번 줄이고 다시 8\*8로

줄이고 또다시 4\*4로 인다.

for (int i = 0; i < DB\_NUM; i++)

db\_imgs4[i] = (int\_rgb\*\*)IntColorAlloc2(db\_height / 8, db\_width / 8);

db\_height4 = db\_height/8;

db\_width4= db\_width/8;

MakeHalf(db\_imgs8, DB\_NUM, db\_height8, db\_width8, db\_imgs4);

③ 이미지를 매칭 시키고 가장 적절한 이미지를 출력하기 위한 set\_img() 함수 호출

set\_img( db\_imgs4, db\_height4, db\_width4, db\_rgb\_average, height, width, original\_img,

img\_out);

: db\_height가 5보다 작기 때문에 8\*8 모자이크 이미지임을 알고 출력화면 상단에

[모자이크8\*8] 띄운다.

④ 이후 과정들에 대해선 Mosaic16X16( ), Mosaic8X8( )와 동일함.

ⅳ. 16X16 DB 이미지를 사용해 랜덤 모자이크하기 : RANDOM\_Mosaic16X16( )

① “ⅰ. 16X16 DB 이미지를 사용해 모자이크하기 : Mosaic16X16( )” 와 동일하게 진행하되,

기존의 OUT\_IMAGE( )대신 OUT\_IMAGE\_RANDOM( ) 함수를 사용함.

void OUT\_IMAGE\_RANDOM(int MosaicSize, int height, int width, int\_rgb db\_rgb\_average[DB\_NUM], int\_rgb\*\* db\_imgs[DB\_NUM], int\_rgb\*\* original\_img, int\_rgb\*\* img\_out)

이 함수는 기존의 OUT\_IMAGE( )에 아래 for문을 추가함으로써 랜덤한 위치에 db이미지가 출력되도록 하였다.

for (int t = 0; t < 300; t++) {

int pos\_y = rand() % (height - MosaicSize);

int pos\_x = rand() % (width - MosaicSize);

}

ⅴ. 입력이미지, 16\*16 db / 8\*8 db / 4\*4 db모자이크 / 랜덤모자이크 보기 : main( )

① main( ) 함수에 Mosaic16X16( ), Mosaic8X8( ) 호출하기.

void main()

{

int height = 665;

int width=351;

int\_rgb\*\* img = ReadColorImage("스파이더맨.jpg", &height, &width);

int\*\* img\_out = (int\*\*)IntAlloc2(height, width);

ColorImageShow("입력영상", img, height, width); //원본영상 출력

Mosaic16X16(); // db가 16\*16인 모자이크 영상 출력

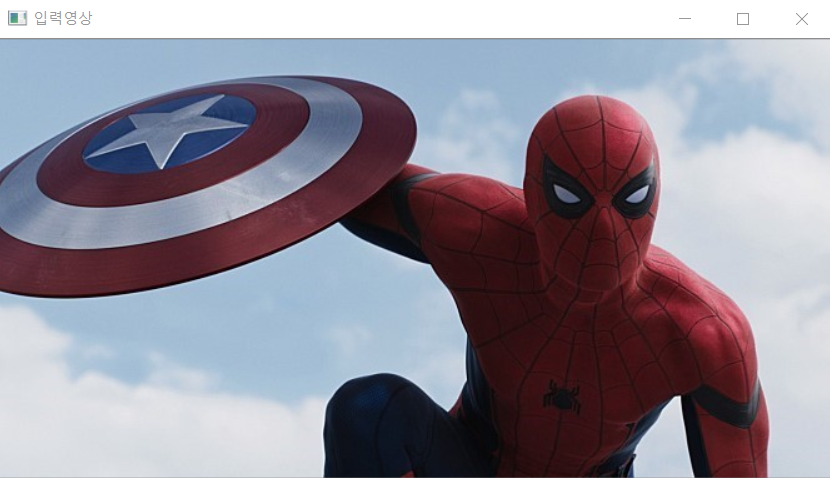
Mosaic8X8(); //db가 8\*8인 모자이크 영상 출력

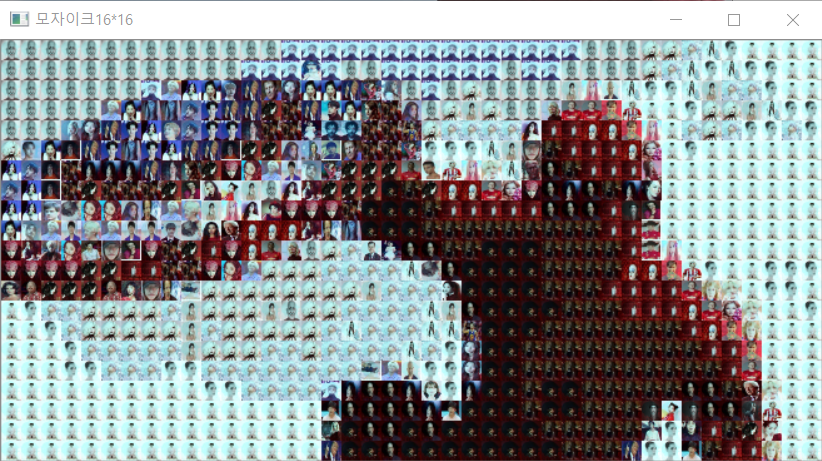
Mosaic4X4(); //db가 4\*4인 모자이크 영상 출력

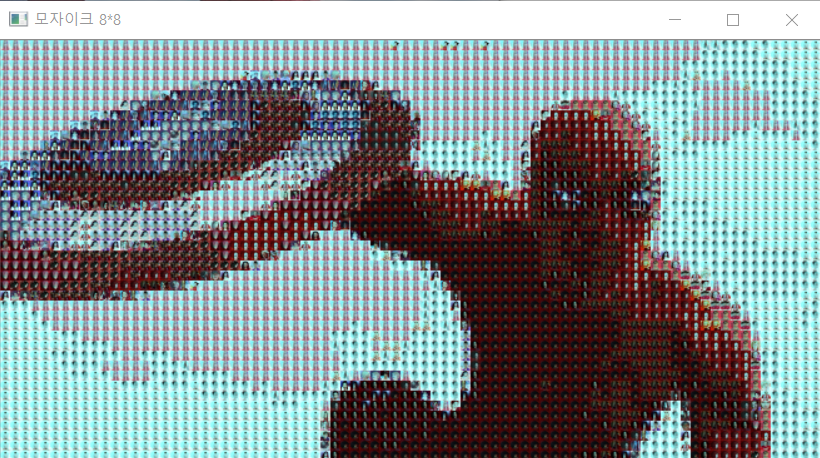
RANDOM\_Mosaic16X16(); // db가 16\*16인 모자이크 영상 출력

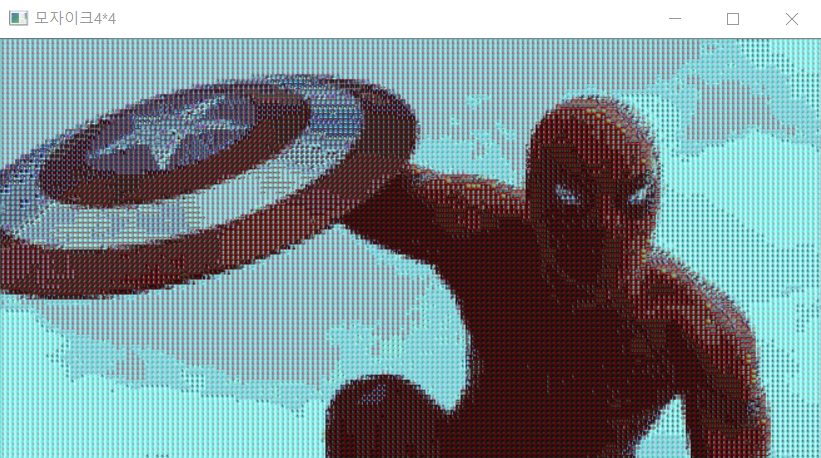
}

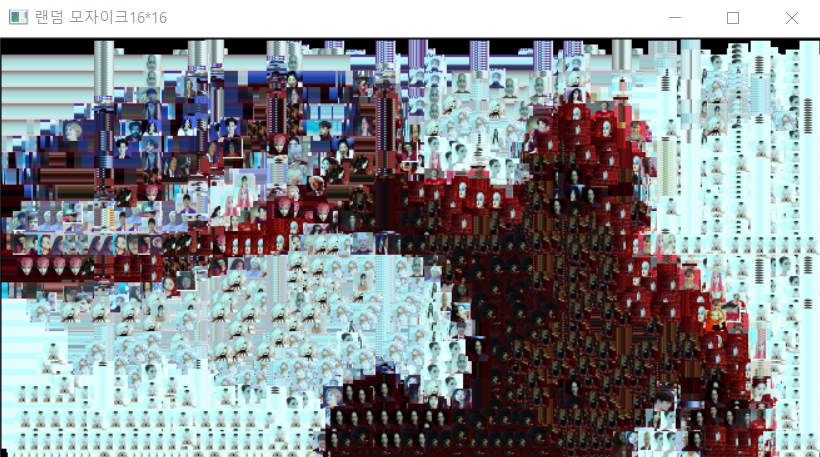
Ⅲ. 결과영상

ⅰ. 입력영상

ⅱ. 출력영상1) 모자이크16\*16

ⅲ. 출력영상2) 모자이크8\*8

ⅳ. 출력영상3) 모자이크4\*4

ⅴ. 출력영상4) 랜덤 모자이크16\* 16