Homework 1-a

1. BMP 파일 포맷

비트맵 디지털 그림을 저장, 윈도우, OS/2 운영체제에 널리 쓰인다.

1~24비트의 색 표현가능, 데이터를 압축하지 않고 사용되지만 압축 방식도 지원한다.

파일 1개의 크기가 4GiB를 넘게 저장할 수 없다.

그림을 가장 아래쪽부터 저장하는 특징이 있다.

헤더가 총 54바이트 (파일헤더 14바이트 비트맵 정보 헤더 40바이트)로 이루어져 있으며 그 외 색상 테이블, 픽셀 데이터로 이루어져 있다. 픽셀 하나를 몇 비트로 저장하느냐에 따라 구조가 달라진다.

색상 테이블의 경우 픽셀 당 색상 수가 16비트 미만일 때 색상 정보를 따로 저장하는 공간이다. 일반적인 24색 비트맵 파일을 사용하는 경우 색상 테이블은 처리하지 않다. 또한 이 경우 red, blue, green정보를 각 1바이트씩 활용해 저장할 수 있다.

1. PNG 파일 포맷

특허 문제가 있는 GIF의 대안으로 개발되었으며 GIF보다 압축률이 높고 8비트와 24비트 트루 컬러를 지원한다. 또한 1600만 색을 지원한다. 비손실 압축으로 인해 이미지 편집에서 많이 사용된다. 하지만 그에 따라 용량이 큰 편이다.

투명한 배경을 가진 것이 특징이다.

고정된 8바이트의 헤더가 있다. Magic number라고도 한다.

헤더의 뒤에는 chunk들로 이루어지는데 chunk는 PNG 파일을 구성하는 기본 단위로 여기에는 데이터 혹은 메타 데이터가 들어간다. Chunk의 구조는 다음과 같다.

Length 4바이트, Chunk type 4바이트, Chunk Data는 Length에서 지정한 바이트만큼, CRC 4바이트

Chuck type은 크게 critical, ancillary가 존재, 4 개의 ASCII 문자로 표현된다. Chuck data는 Chunk type에 맞는 데이터가 들어가며 이미지 데이터가 들어갈 수도, 메타 데이터가 들어갈 수도 있는 필드이다. CRC는 Length를 제외한 type과 data를 checksum 낸 것 (검사합)

Critical Chuck는 필수 정보를 담는 컨테이너이다. Decoder는 반드시 이를 읽을 수 있어야 한다. 이미지 기본 정보, 색상 팔레트, 이미지 데이터, 이미지의 끝에 관한 정보 등을 담는다.

Ancillary Chunk는 critical chunk에서 담지 않는 다른 이미지 속성들을 담을 수 있는 컨테이너이다. 배경 색 지정, Exif metadata, Gamma value, 투명도 정보 등 다양한 추가 정보들을 저장한다.

1. GIF 파일 포맷

최대 256 색까지 저장 가능한 비손실 압축 형식이다.

해당 포맷에 쓰인 LZW 알고리즘에 대한 특허를 유니시스 사가 가지고 있어 색 표현에 대한 단점과 더불어 이에 대한 대체로 PNG가 개발되었다.

Pixel 당 8비트 지원, 움직이는 이미지 지원

GIF의 시작은 Header로 버전 및 포맷을 알려주는 매직 넘버로 시작한다. 그 이후 logical screen이 가지는 기본적인 속성들을 명시한 descriptor가 이어진다.

GIF 에서는 기본적으로 사용되는 색상을 table에 놓고 그 인덱스 값을 pixel에 대응시키는 식으로 이미지를 표현한다. 따라서 각 프레임은 파일 내 모든 프레임이 공유하는 global color table과 프레임 개별로 가질 수 있는 local color table 중에 선택하여 색상을 인덱싱한다.

Header가 끝나면 일련의 block 집합으로 표현되며 각 block은 1 바이트의 고유 sentinel로 구분된다. (이미지, Extension block, Trailer)

이미지 block 에서는 이미지를 정의, image descriptor로 시작한다. 이미지 위치, 크기 등이 여기에 들어가고 local color table이 필요한 경우 함께 명시된다. 이후 LZW 알고리즘으로 압축된 sub block 들이 연결 리스트로 이어져서 명시된다.

Extension block 은 이미지를 더 풍부하게 만들어진다. Plain text, graphic control, comment 등의 종류가 있으며 이 또한 sub block으로 이루어진다. Extension block이 이미지를 수정하는 경우 해당 이미지 앞에 위치해야 한다. 해당 블록은 필수가 아니고 해석 불가능해도 이미지를 그릴 수 있다.

GIF는 RGB값으로 색상이 table에 기록되며 각 픽셀 별 8비트로 256 색상 표현이 가능하다. 이후 dithering이라는 것을 통해 작은 color palette로도 여러 색상을 비슷하게 표현 가능하게 되었다고 한다.

처음부터 움직이는 이미지를 저장하기 위한 포맷은 아니었지만 한 파일에 여러 장의 이미지를 저장할 수 있다는 속성이 프레임의 시간적 순서를 저장하기까지 확장된 것이다.

1. JPEG 확장자 포맷

이미지를 압축 저장하는 방식 중 하나이며 기본적인 압축 방식은 lossy compression이다. 이는 원본 데이터를 보존하지 못하고 이를 희생해서 압축률을 높인다. (용량과 화질 간의 trade off 존재) 보통 10:1의 압축률을 보인다.

JPEG는 여러 segment로 구성되는데 각 segment 처음에는 marker가 존재한다. 모든 marker는 0xFF로 시작하며 그 뒤에 1byte의 marker 정보를 담는다.

그 뒤에 playload data의 length, payload data가 나오며 payload 없이 2byte만 가지고 있는 marker도 있다.

Marker에 따라 어떤 정보를 지니는지, 어떤 기법 기반의 JPEG인지 표시할 수 있으며 이미지 정보 역시 marker롤 통한다.

1. RAW 파일 포맷

확장자가 업체나 제조사에 따라 다르다.

이미지가 언제나 원본 그대로이다.

카메라의 raw 파일들은 12비트나 14비트의 명암 정보를 가진다.

색 공간을 사용자가 원하는 대로 설정 가능하다.

RAW 이미지에서는 픽셀 값이 빨강 녹색 파랑 중 한 가지의 값만 가지고, EV 스톱 단위의 상대치로 저장된다. 색 공간에 맞춰진 상태가 아니므로 절대적인 흰색 레벨이나 검은색 레벨도 정해져 있지 않다.

RAW에서 비트 수는 단순히 색상의 단계가 아니라 저장되는 노출 단계 수이다.

헤더 정보가 없이 완전한 데이터만으로 구성되어 영상처리가 복잡하지 않다. 다만 그러한 이유로 사용자가 마리 영상의 색상 정보나 해상도 정보를 알고 있어야 한다.

Homework1-b

텍스트이(가) 표시된 사진

자동 생성된 설명

GUI는 결국 만들지 못해서 콘솔 창으로라도 사용자의 입력을 받아 이미지를 저장할 수 있게 했습니다. 아래 사진은 각 회전 이미지/좌우/상하 반전 이미지입니다.

다른, 색, 여러개이(가) 표시된 사진

자동 생성된 설명

윗줄은 왼쪽부터 원본, 좌우반전, 상하반전 이미지이고,

아랫줄은 왼쪽부터 오른쪽을 90도 회전, 180도 회전, 왼쪽으로 90도 회전 이미지입니다.

Discussion chapter

컴퓨터 앞에서 오래 고민하기 보다는 생활하는 틈틈히 어떻게 이미지를 회전하면 좋을 지에 대해 생각했고, 덕분에 컴퓨터 앞에서 코딩하는 시간은 많지 않았습니다. 생각하는 즐거움이 있는 과제였다고 생각합니다.

인덱스 표기법이 아닌 포인터를 사용함으로써 조금 더 생각을 해야 해서 재밌는 퀴즈를 푸는 기분이었습니다. 그런데 왜 인덱스 표기법을 사용하지 말라고 하신 지에 대해서는 궁금했습니다. 하지만 조교님과 피드백 과정에서 배열이 아닌 포인터를 사용하면 메모리 사용을 줄일 수 있다고 하셔서 해당 궁금증이 해결되었습니다. 또한 포인터로 2차원 배열에서 원하는 위치를 찾아가는 것이 인덱스 표기법의 원리와 크게 다르지 않다는 사실을 배웠고, 생각보다 금방 적응되었습니다.

또한 에러로 큰 고생을 하진 않았는데, 오른쪽으로 90도를 생각하고 코드를 짰더니 왼쪽으로 90도 회전했고, 왼쪽으로 90도를 생각하고 코드를 짰더니 오른쪽으로 회전해서 약간의 당황스러움이 있었고 사실 아직도 헷갈리는 부분입니다. 본 수업이 점수가 아닌 순수 목적 공부이기에 이 또한 틈틈히 생각해서 이해하겠습니다.

각 회전 각도나 좌우반전/상하반전을 모두 함수 분리를 해서 프로그래밍을 하면 변수 선언 부분이 너무 중복되어 프로그래밍을 함수 한 개에서 정수 입력을 받아 그 정수 값에 따라 조건문을 사용, 90도 회전 180도 회전 좌우 반전 등을 선택할 수 있게 하였는데 조교님과 피드백 과정에서 프로그래밍 구조가 더 복잡해지면 변수가 중복되어도 괜찮으니 함수 분리를 하는 것이 더 낫다고 해서 수정해서 작성했습니다.

저번 수업이 끝나고 GUI로 과제를 프로그래밍 하는 것을 교수님과 약속했지만 결국에는 성공하지 못해 아쉬움이 남습니다. 대체물로 콘솔창을 통해 프로그래밍했습니다.

개인적으로 과제 기간이 주어지면 최대한 빠르게 하기 보다는 넉넉하게 시간을 잡고 하는 편인데 여러모로 최대한 빠르게 과제를 하는 것이 과제에 대한 추가적으로 해보고 싶은 것들도 해보고 생각도 더 해보기 좋을 것 같아 다음 과제부터는 최대한 빠르게 과제를 해야 겠다는 생각을 했습니다.