16비트 칼라 LCD를 장착한 휴대단말기의 GUI용 비손실 영상 압축 효율을 위한 전처리

오황석°
한국산업기술대학교 게임공학과
hsoh@kpu.ac.kr

Preprocessing of lossless 16-bit color image compressions for mobile terminal GUI

Hwang-Seok Oh°
Dept. of Game & Multimedia Engineering, Korea Polytechnic University

요 약

본 논문은 16비트 칼라 LCD 디스플레이 장치를 갖는 휴대용 단말기에 영상을 이용한 그래픽 사용자 인터페이스(Graphic User Interface : GUI)를 제공함에 있어 단말기의 저장 공간을 효율적으로 사용하기 위하여 대부분 비손실 영상 압축 기법을 사용하는데, 이들 압축 기법의 압축 효과를 높이기 위한 전처리 과정 방법에 대하여 기술한다. 전처리 과정은 일반적인 칼라 영상의 각 8비트로 구성된 R, G, B 컴포넌트를 16비트로 구성된 RGB 비트 수에 맞도록 원 영상의 칼라를 조정, 16비트로 축소된 칼라 영상에서 사용된 칼라 수를 계산하는 단계 칼라 수가 사용자가 입력한 칼라수의 임계값 보다 작은 경우 인덱스 칼라 이미지를 만드는 단계, 인덱스화된 영상과 칼라 수가 임계값 보다 큰 경우 16비트 칼라 영상을 대상으로 영상을 겹치지 않게 분할하는 단계 분할된 영상의 각 블록 단위로 Run의 평균 개수가 가장 많은 스캐닝 방법을 결정하는 단계 그리고 스케닝 순서에 따른 기존의 비손실 압축 기법을 적용의 순으로 이루어진다. 본 논문에서 기술한 전처리 단계를 거친 영상을 JPEG-LS에 적용하였을 경우 동일한 영상에서 약 10~20%의 압축 효율을 얻을 수 있었다.

1. 서 론

이동통신 단말기, PMP 등 대부분의 휴대 단말기들이 채용하고 있는 16비트 칼라 표시 장치에서 사용자와 상호 작용을 하는 UI(User Interface)를 영상 등으로 구성하여 동적인 효과와 시각적인 편리함을 제공함에 있어서 영상 리소스를 단말기에 저장할 경우 매우 많은 저장 장치를 필요로 한다. 또한 사용자 인터페이스가 활성화되어 사용자와 상호 작용을 수행함에 따라 저장 장치에 저장되어 있는 리소스들을 메모리에 적재하여 빠르게 수행하기 위해 많은 메모리를 필요로 한다.

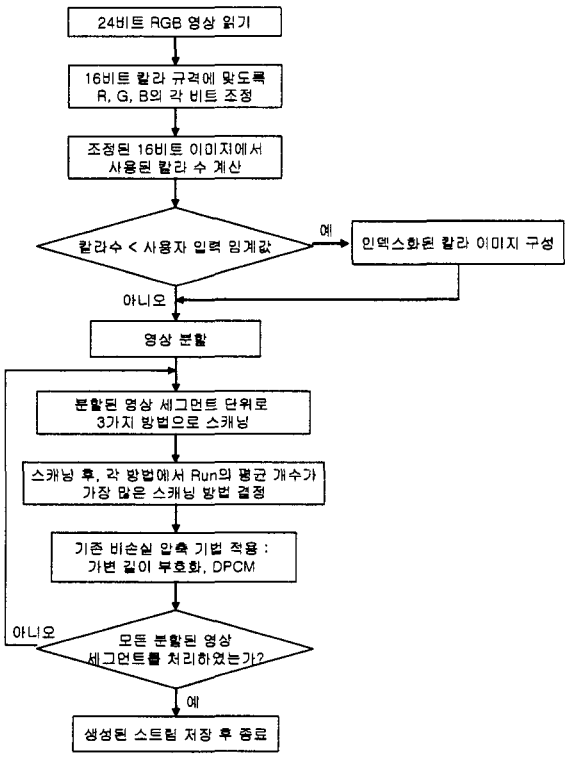
기존의 휴대 단말기나 휴대형 임베디드 시스템에서 영상을 이용하여 사용자 인터페이스를 제공하기 위해 Run length 기반 영상 압축 기법, 디렉토리 개념을 이용한 영상 압축 기법 등을 사용하였다. 사용자 인터페이스에 사용되는 영상은 주로 그레이디언트가 많거나 그래픽 툴들을 이용해서 생성한 영상들을 많이 사용하기 때문에 사진과 같은 실사 영상들의 특성과는 차이가 있다. 특히, 사용자 인터페이스로 사용되는 영상들은 특정 칼라를 투생색으로 할당하여 활용하는 경우가 있어 손실 압축을 할 경우 사용자에게 눈에 띄게 드러나는 결점이 발생한다. 이러한 문제 때문에 사용자 인터페이스로 활용되는 영상 리소스들은 리소스의 양을 줄여 휴대 단말기의 저장 공간을 최소화하고자 비손실 압축 영상 기법을 활용하고 있다. 대표적인 압축 기법은 간단하면서도 인위적으로 합성한 영상에 효과가 큰 run-length 기반 압축 기법으로 동일한 칼라의 경우에 run과 run의 길이 쌍으로 표현하면 영상의 각 화소를 표현하는 것보다 데이터 양을 줄일 수 있다. 또 다른 방법은 대부분의 단말기들이 16비트 칼라를 사용한다는 것을 착안하여 디스플레이될 때, 24비트 칼라 중에 절사되는 부분을 미리 절사 한 후, 기존의 run-length 기법을 적용하거나 LZW, LZO, ZIP, PNG, JPEG-LS 기법을 적용하는 기법을 사용하였다. 휴대 단말기의 사용자 인터페이스 용도로 사용된 영상 압축 기법들은 휴대 단말기의 성능을 고려해서 상대적으로 간단한 방법을 적용하고 있다. 이러한 제약 사항으로 인해 압축 효율이 높지 못하다는 단점이 있다. 또한 기존의 압축 기법을 가져와 속도 측면에서 최적화하는 기법들이 동일한 기법들이 휴대 단말기에 적용되어 사용되어 왔다.

압축 효율을 높이기 위한 전처리 과정은 주로 실사 영상에서 고주파 영역을 제거하거나 노이즈를 제거하여 압축 효율을 높이는 방법들이 사용되어 왔다. 이러한 기법은 상기한 바와 같이 사용자 인터페이스로 사용되는 영상의 특성에서는 발생하지 않는다. 또한 사용자 인터페이스 영상에서 강조를 하기 위해 영상의 비연속성 특징을 이용하고 있는데 이러한 전처리 과정은 사용자 인터페이스에서 사용될 영상의 주요 특성들을 제거하는 문제점을 야기시킨다.

본 논문에서는 휴대형 단말기에서 GUI(graphics user interface)를 사용되는 영상의 특성을 반영하여 영상 리소스를 압축하는데 있어 효과적인 전처리를 함으로써 압축 효율을 높이 방법에 대하여 기술한다.

2. 비손실 압축 기법의 효율을 위한 전처리 과정

휴대 단말기의 사용자 인터페이스로 사용되는 영상의 효율적인 압축을 위한 전처리 과정으로 일반적인 영상이 가지는 칼라 비트 수를 타겟 단말기의 칼라 비트 수에 맞추는 방법, 영상의 특성을 이용하여 인덱스화된 이미지 구성, 압축의 효율성을 높이기 위해 중복성이 많은 블록 단위로 분할 및 각 블록 단위로 스캐닝하여 최적의 효율을 낼 수 있는 스캐닝 방향 결정 및 기존의 알고리즘을 적용하여 최종 압축 스트림을 만들어 내는 과정으로 구성된다. [그림 1]은 전처리 과정을 흐름도 형식으로 기술하였다. 각 단계의 세부적인 방법은 다음과 같다.



[그림 1] 비손실 압축 기법에 적용하는 전처리 과정

● **입력 영상의 비트 수 조정.**

사용자 인터페이스로 활용될 영상은 기본적으로 트루 칼라(true color)로 대부분 24비트(R 8비트, G 8비트, B 8비트) 칼라로 구성되어 있다. 경우에 따라 RGB 각 요소가 10비트로 구성된 영상도 있으나 일반성을 잃지 않기 위해 24비트로 만들어진 영상을 전처리 과정의 입력으로 본다. 24비트 칼라 영상에서도 사용되는 칼라 수가 256가지 이하인 경우 인덱스화된 칼라 이미지를 양식을 사용한다. 이 경우도 각 화소를 24비트 RGB로 표현할 수 있기 때문에 24비트 RGB 영상을 입력으로 가정한다. 입력된 영상은 16비트 칼라 규격에 맞도록 조정이 된다. 16비트 칼라 규격은 휴대 단말기에서 채용한 LCD의 각 칼라 성분이 몇 비트로 구성되었느냐에 따라 달라질 수 있다. 예를 들어 R 성분이 5비트 , G 성분이 6비트, B 성분이 5비트인 경우 통상 RGB565로 명명한다. 8비트 R, G, B의 각 성분을 MSB(Most significant bit)부터 타겟 비트 수(R 5비트, G 6비트, 5비트) 만큼 잘라 16비트 칼라 영상을 만든다. 즉, 8비트 R 성분 중에서 상위 5비트만을 타겟 R 성분으로 사용하고, G 성분은 상위 6비트를, B 성분은 상위 5비트를 이용해서 16비트 RGB565 영상을 만든다. 타겟 LCD에서 수용하는 칼라 성분의 비트 수가 RGB565가 아닐 경우에는 동일한 방법으로 16비트 칼라 영상을 만든다. 타겟 LCD의 칼라 비트 수가 16비트가 아닌 경우에도 동일한 방법으로 적용이 가능하다.

● **칼라 수 계산 및 인덱스 이미지 만들기**

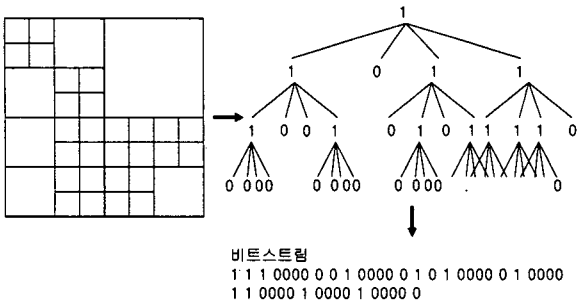
16비트 칼라 영상을 왼쪽 위에서 오른쪽 아래 방향으로 가로 스캐닝 방향대로 각 화소의 2바이트 값을 검사하여 이전에 동일한 값이 검색되었는지 확인하여 이전에 나오지 않은 칼라의 경우 새로운 칼라 값을 하나 추가하고, 그 칼라의 빈도수를 1로 설정한다. 이전에 나온 칼라의 경우 해당 칼라의 빈도수를 1 증가 시킨다. 영상의 모든 화소에 대하여 검사를 완료하면 사용된 칼라의 개수와 각 칼라의 출현 빈도를 얻을 수 있다. 이 결과를 이용해서 사용자가 설정한 인덱스 이미지 구성할 칼라 수의 임계값과 비교하여 임계값보다 작을 경우 인덱스 칼라 영상을 구성하여 영상 분할이 인덱스화된 영상에 적용되고, 그렇지 않을 경우 축소된 16비트 칼라 영상이 영상 분할의 대상이 된다. 사용자가 설정한 인덱스화할 때 사용되는 칼라 수 임계값은 일반적으로 256 값을 많이 사용하고 있으나 사용되는 칼라의 수에 따라 압축 효율이 날 수 있으므로 사용자가 설정할 수 있도록 한다.

인덱스화된 칼라 이미지 구성은 사용된 칼라 수만큼 인덱스 테이블을 구성하고, 16비트 축소된 영상의 각 화소는 역인덱스되어 화소값 대신 인덱스 값을 가진다.

● **영상 분할 및 스캐닝 방향 결정**

인덱스화된 영상과 칼라 수가 임계값보다 클 경우 16비트 축소된 영상을 대상으로 영상 분할이 적용된다. 영상 분할은 대상 영역 내의 모든 화소 값들이 동질성을 지니는지를 판단하고, 만약 서로 상이한 성질을 가진 화소들로 영역이 구성되어 있다면 대상 블록을 더 세부적인 블록으로 나누고, 동질성을 가지는 경우 더 이상 블록을 분할하지 않는다. 블록이 같은 값 또는 유사한 값으로만 이루어져 있을 경우, 인접한 화소간의 차이가 크지 않기 때문에 압축 효율이 좋아진다. 이러한 특성을 이용하여 입력 영상을 대상으로 먼저 최대 블록의 크기를 32x32, 최소 블록의 크기를 4x4로 설정한 후, 입력 영상을 겹치지 않게 32x32 블록으로 나눈다. 각 블록에 대하여 동질성 테스트를 수행한 후, 유사한 화소들로 구성되었다고 판별되는 경우는 더 이상 블록을 나누지 않고, 동질성이 낮을 경우는 해당 블록을 가로/세로 각각 1/2의 크기를 가지는 4개의 부블럭으로 나눈다. 4개의 각 부블럭에 대하여 동일한 유사성 검사를 통해서 더욱 세부적으로 나눌 수 있다. 본 논문에서는 블록의 최대 크기와 최소 크기를 각각 32x32, 4x4로 나누었지만 그 값은 사용자가 설정할 수 있으며 동일한 방법을 적용할 수 있다. 영상 분할에서 동질성 검사는 다음과 같이 수행된다. 동질성 테스트는 각 블록의 표준 편차와 임계값의 비교를 통해서 이루어진다. 16비트 축소된 영상 RGB 565에 서 동질성 검사를 위해 사용되는 임계값은 ST_{16}(Segmentation Threshold for 16 bits color image), 인덱스화된 영상의 동질성 검사를 위한 임계값은 ST_I(Segmentation Threshold for Indexed color image)로 사용자가 값을 결정하여 사용한다. 16비트 축소된 영상에서는 RGB565 각 성분의 표준 편차를 구하여 이 편차가 사용자가 설정

한 분할 임계값ST_{16}보다 작을 경우 더 이상 분할하지 않으며, 분할 임계값 ST_{16}보다 해당 블록의 표준 편차가 클 경우, 4개의 부블록으로 나눈다. 인덱스화된 영상에서는 동일한 방법으로 영상을 분할하지만 분할 임계값을 ST_i을 사용한다. 나주어진 4개의 부블럭에 대해서도 같은 절차를 계속적으로 적용하여 블록을 분할한다. 최종적으로 분할된 블록들은 영상 압축의 단위로 이용된다. [그림2]에서는 최종적으로 분할된 블록 형태와 이를 표현한 예를 보여준다. 분할된 블록은 계층 구조로 표현할 수 있다. 여기서 '1'은 해당 블록이 분할되는 경우이고, '0'은 더 이상 분할되지 않음을 의미한다. 계층 구조에서 하나의 블록은 4개의 부블럭으로 분할되므로, 부모 노드의 값이 1인 경우는 4개의 자식 노드를 가지게 된다. 분할 정보는 계층 구조를 전위 우선 탐색(Preorder traversal)으로 탐색하였을 경우의 비트 값의 순서이다. 이 경로 탐색 형태로 비트 스트림을 만들었을 경우, 반대로 비트스트림으로부터 분할된 블록 형태를 복원할 수 있다. 전위 우선 탐색 기법은 트리에서 루트 노드를 먼저 방문하고, 자식 노드에서는 왼쪽에서 오른쪽으로 방문한다. 자식 노드가 또 서브 트리로 구성되어 있다면 자식 노드의 루트를 먼저 방문하고 그 자식 노드의 서브트리를 왼쪽에서 오른쪽으로 방문한다. 비트스트림에서 첫번째 비트 1은 전체 영상이 분할되어 4개의 서브 블록으로 나누어지고, 두번째 비트 1은 4 서브 블록 중 왼쪽위 블록이 다시 4개로 분할된다는 의미이며, 3번째 비트 1은 분할된 4개의 블록 중에 왼쪽위쪽 블록이 4번 분할된다는 의미이다. 다음의 4 비트 0000은 방금 분할된 4블록이 더 이상 분해되지 않음을 표시한다. 다음의 두 비트 0, 0는 각각 오른쪽위, 왼쪽 아래 블록이 두번째 단계에서 더 이상 분해 되지 않음을 의미한다. 이와 같와 분해된 블록으로부터 비트 스트림을, 반대로 비트 스트림으로부터 분해된 블록의 형태를 복원할 수 있다.



[그림2] 영상 분할 및 분할 정보 표현

분할된 블록은 블록 분할 정보 비트 스트림에 기술된 0이 나오는 순서대로 압축 알고리즘이 적용된다. 본 논문에서 제안한 방법은 기존의 DPCM 압축 기법이나 run-length 압축 기법의 효율을 높이기 위한 전처리 과정에 관한 것으로 기존의 압축 기법을 그대로 적용할 수 있다. 블록 분할 비트 스트림의 0에 해당되는 블록을 입력 받아 3가지 스캐닝 방향, 즉 가로 우선 스캐닝, 세로 우선 스캐닝, 지그재그 스캐닝의 순서대로 화소를 방문하여 각 방향으로 스캐닝하였을 때 평균 run의 길이를 구한다. 스캐닝 결과의 run의 평균값이 가장 큰 방향으로 스캐닝하면서 기존 압축 알고리즘이 적용되어 영상이 압축된다. 최종 압축 스트림은 인덱스화된 영상의 경우, 칼라 수, 인덱스 테이블, 블록 분할 비트 스트림, 각 블록의 스캐닝 방향 정보, 각 블록의 압축된 스트림으로 구성된다. 16비트로 축소된 칼라 영상의 경우 블록 분할 비트 스트림, 각 블록의 스캐닝 방향 정보, 각 블록의 압축된 스트림으로 구성된다.

3. 실험 결과

본 논문에서 제안한 방법의 성능을 평가하기 위해서 휴대 단말기에서 UI로 사용하는 영상을 대상으로 전처리를 수행하지 않은 경우와 전처리를 수행한 후의 JPEG-LS의 압축 효율을 비교하였다. 전처리를 수행하지 않은 경우도 16비트 칼라로 만든 후에 적용하였다.

- 실험 영상의 크기 : 176x220 24비트 칼라 영상 3개
- 실험 결과 : [표1]에 실험결과를 기술하였다. 전처리 과정 전에 비하여 약 10-20% 정도의 압축 효율을 얻을 수 있었다.

[표1] 제안한 방법의 압축 효율(바이트수)

실험영상	전처리 과정 전	전처리 과정 후
영상1	16450	14501
영상2	18277	15986
영상3	13539	10302

4. 결론

본 논문에서는 휴대용 단말 장치, 휴대용 게임기, 휴대용 미디어 재생기와 같이 제한된 칼라 수를 사용하는 LCD를 부착하여 영상을 출력하는 표시장치를 가지며, 제한된 영상 리소스 메모리를 가지는 단말기에서 비손실 고압축 방법으로 영상을 압축하여 많은 양의 리소스 영상 데이터를 저장하고, 이를 이용해서 보다 다이나믹한 영상을 이용한 그래픽 기반 UI(GUI) 표현이 가능하도록 하기 위한 전처리 과정을 다루었다. 동일한 영상 리소스를 작은 양의 저장장치에 저장하여 활용할 수 있으며, 동일한 저장 장치를 가지는 단말기에서 보다 많은 영상 데이터를 저장할 수 있어 저장장치의 비용을 감소시키거나 보다 다양한 UI를 구성할 수 있는 리소스를 저장할 수 있다. 또한 제안한 방법은 단말기의 영상 리소스 뿐만 아니라 비손실 영상 압축 방법을 사용하는 다른 응용 프로그램에 광범위하게 활용될 수 있다.

[참고문헌]

1. SO/IES 14495-1 International Standard, "Information technology - Lossless and near-l oss less eo m press i on of continuous-tone still images: Baseline", ISO/IEC, 1999.
2. Marcelo Weinberger, Gadiel Seroussi, Guillermo Sapiro, "The LOCO-I Lossless Image Compression Algorithm: Principles and Standardization into JPEG-LS", HPL-98-193, Nov., 1998.
3. Mark Nelson, "The Data Compression Book", 2nd ed., M&T Books, 1995.