컬러 영상의 부호화를 위한 쿼드트리 코덱의 구현

*방민석, *김병연, *강상구, *서재선, *박재성, *강동욱

* 국민대학교 전자공학부

* bmons85@kookmin.ac.kr

Realization of Quadtree Codec for Color Image Coding

*Min-Suk Bang, *Byung-Yeon Kim, *Sang-Goo Kang, *Jae-Son Seo, *Jae-Sung Park, *Dong-Uk Kang

* Dept. of Electronics Engineering, Kookmin University

요약

본 논문은 최근에 휘도 영상 부호화를 위해서 제안된 에지 선 기반의 쿼드트리 영상 압축 기법을 기초로 해서 이를 컬러 영상을 압축하도록 확장하고, 확장된 알고리즘을 C-언어를 사용해서 실제로 구현한 쿼드트리 영상 코덱을 소개한다. 개발된 코덱의 부호율-왜곡 성능과 복원영상의 주관적 화질을 JPEG과 비교할 때, 컬러 영상을 고화질로 부호화하는 경우에 개발된 쿼드트리 코덱의 성능이 특히 우수함을 확인하였다.

1. 서론

최근 에지 선(edge line) 기반의 쿼드트리 영상 압축 기법이 제안되었다[1]. 에지 선이란 입력 영상에서 서로 유사한 방향의 휘도 변화가 서로 연결되어 나타나는 것을 의미하며, 이는 주로 물체의 윤곽선을 따라서 발생한다. 그 이유로 영상 분해 과정에서 에지 선을 보존하게 되면 분할된 블록의 경계선이 물체의 윤곽선과 일치하게 된다. 이처럼 윤곽선들로 분리된 객체들을 부호화하면 데이터 압축 과정에서 발생하는 고주파수 성분의 손실에 따른 윤곽선의 끊김 현상이나 윤곽선의 물결 현상(ringing effect)을 원천적으로 배제시킬 수가 있다[2]. 참고문헌 [1]에서는 에지 선 기반 영상 부호기가 고정 블록 부호화 방식인 JPEG에 비해서 더 효율적으로 휘도 영상을 부호화할 수 있음을 보이고 있다. 특히 강한 윤곽선 부근에서의 물결 현상은 고정블록 부호화 기법에서 상당히 높은 비트율로 부호화하는 경우에도 완전히 없앨 수가 없다. 하지만 제안하는 방식에서는 물결 현상이 나타나지 않기 때문에, 고화질 부호화의 경우 복원영상의 주관적 화질이 JPEG보다 뚜렷하게 우수함을 보이고 있다.

참고문헌 [1]은 매트랩상에서의 모의실험 결과를 비교하였다. 표준화된 JPEG 등과의 공정한 부호율-왜곡 성능 비교를 위해서는 실제 코덱을 통해서 비트스트림을 생성하고 이 비트스트림으로부터 영상을 복원해냄으로써 복호에 필수적인 오버헤드 비트까지 고려된 부호율-왜곡 성능을 비교할 필요가 있다. 그리고 참고문헌 [1]에서는 휘도 영상에 대한 실험 결과만을 제시하고 있다. 디지털 영상들이 대부분 컬러 영상이므로 제안하는 압축 기법의 실용성을 높이기 위해서는 컬러 영상을 압축하도록 알고리즘을 확장할 필요가 있다.

본 논문에서는 참고문헌 [1]의 쿼드트리 영상압축 프레임워크를 기초로 해서 이를 컬러 영상을 압축하도록 확장하고, 확장된 알고리즘을 C-언어를 사용해서 실제로 구현함으로써 쿼드트리 영상 코덱을 개발하였다. 그리고 개발된 코덱의 부호율-왜곡 성능과 복원영상의 주관적 화질을 JPEG과 비교하였다.

2. 알고리즘

코덱 구현을 위한 알고리즘은 다음 그림1 과 같이 최근 제안된 에지 선 기반의 쿼드트리 영상 압축 기법[1]의 알고리즘을 바탕으로 한다. 참고문헌[1]과 달라진 점은 컬러영상을 압축하기 위한 전처리 과정과 잎 노드의 부호화를 위해서 예측부호화 기법과 양자화 계수에 대한 가변길이부호화를 적용한 점이다. 알고리즘에 대한 자세한 설명은 참고문헌 [1]을 참조하면 된다. 이 논문에서는 참고문헌 [1]에 없거나 달라진 부분에 대해서 소개한다.

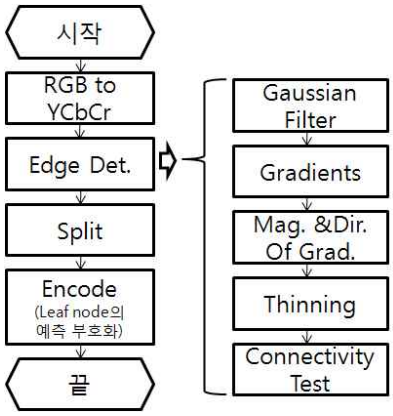


그림 1 쿼드트리 인코더

가. 컬러 처리

식(1)을 통해 RGB 영상을 YCbCr 영상으로 변환하고, Cb, Cr 영상은 수직, 수평 방향으로 각각 2:1로 서브샘플링을 수행해서 크기를 1/4로 줄여서 부호화하도록 하였다.

$$Y=0.299R + 0.587G + 0.114B$$
$$Cb=128 - 0.168636R - 0.331264G + 0.5B \quad (1)$$
$$Cr=128 + 0.5R - 0.418688G - 0.081312B$$

Y와 Cb, Cr은 독립적으로 쿼드트리 분해하고 부호화한다. 즉 Y와 Cb, Cr 영상마다 에지 선 맵을 각각 구하고, 이를 활용해서 탑-다운(top-down) 방식으로 영상을 분해한다. Cb와 Cr에 대한 쿼드트리 분해 구조는 매우 유사하다. 그렇지만 완전히 일치하지는 않는다. 따라서 다음 절에서 소개할 잎 노드에 대한 예측부호화를 결합해서 부호화하는 경우, 두 영상을 독립적으로 분해 부호화하는 것이 하나의 분해 구조를 보내고 그 구조에서의 잎 노드마다 Cb와 Cr을 모두 보내는 방식보다 더 효율적이다.

나. 잎 노드에 대한 예측 부호화

잎 노드 블록에 대해서 블록의 대푯값을 부호화하는 대신에 인접 블록을 이용해서 대푯값을 예측한 다음 그 예측 오차를 부호화하였다. 에지 선이 존재하는 경우, 강제적으로 1x1 블록까지 분해하게 되는 경우가 자주 발생하는데, 이처럼 작은 블록들로 분해하게 되면 이 작은 블록들은 인접하는 블록들과의 상관성이 매우 높다. 따라서 이 블록들을 예측부호화하면 부호화 효율이 향상된다.

예측부호화는 가장 큰 블록부터 작은 블록의 순으로 잎 노드를 부호화하는 과정에서, 다음과 같은 순서로 수행한다.

① 현재 블록과 경계를 이루는 블록 중에서 이미 부호화된 블록이 존재하는지를 확인한다. 탑-다운 방식으로 부호화하고 있으므로 부호화된 이웃 블록은 현재 블록의 크기보다 크거나 같다. 따라서 확인해야 할 블록은 최대 4개로 한정된다(그림 2 참조).

② 부호화된 이웃 블록의 수에 따라서 다음과 같이 현재 블록의 대푯값을 예측한다.

a. 부호화된 이웃 블록이 없으면(그림 2(a)의 경우), 현재 블록의 대푯값을 8-비트 영상의 중간값인 128로 예측한다.

b. 부호화된 이웃 블록이 하나만 있으면(예를 들어 그림 2(b)의 경우), 그 이웃 블록의 대푯값으로 현재 블록을 예측한다.

c. 부호화된 이웃 블록이 두 개이면(예를 들어 그림 2(c)의 경우), 두 이웃 블록의 대푯값들의 평균으로 현재 블록을 예측한다.

d. 부호화된 이웃 블록이 세 개이면(그림 2(d)의 경우) 세 이웃 블록 대푯값들의 중간값(median)으로 현재 블록을 예측한다.

e. 네 이웃 블록이 모두 부호화되어 있으면(그림 2(e)의 경우), 네 이웃 블록의 대푯값들 중에서 중간 크기의 두 점의 평균값으로 현재 블록을 예측한다.

③ 현재 블록의 평균값을 위 단계를 통해 예측하고 남은 예측 오차를 지수 골롬 부호(exponential Golomb code[3])를 사용해서 가변길이 부호화한다.

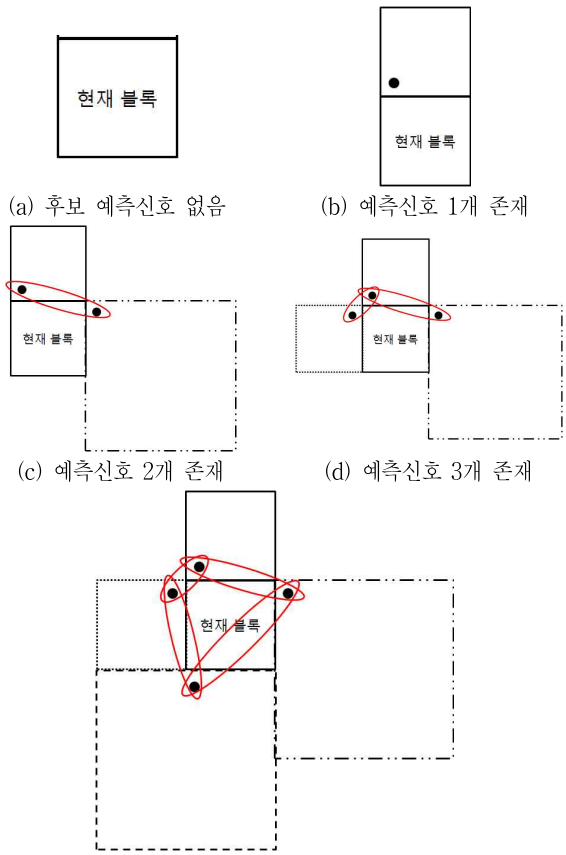
다. 소프트웨어 코덱 구현

Microsoft의 Visual Studio 2008을 사용해서 앞에서 설명한 알고리즘을 적용한 쿼드트리 코덱 소프트웨어를 구현하였다. 쿼드트리 비트스트림의 복호에 필요한 오버헤드는 총 28바이트로서 다음과 같은 구조체로서 정의하였다.

```
typedef struct tagQUADTREEHEADER {
  WORD   bqType;          //'QT'라고 쓴다
  LONG   bqWidth;         // 영상의 폭(화소수)
  LONG   bqHeight;        // 영상의 높이(화소수)
  BYTE   bqBitCount;      // 8=회색영상, 24=컬러영상
  BYTE   bqPSNR;          // 양자화기 설정에 사용
  BYTE   bqTopLayer[3];   // 최상위 레이어(매크로블록)
  DWORD bqSizeStructure;  // 트리 구조를 표현하는 비트수
  DWORD bqSizeImage;      // 잎 노드 값을 표현하는 비트수
} QUADTREEHEADER;
```

bqPSNR은 부호기에서 사용한 각 레이어별 양자화 스텝 크기를 복호기가 계산해내는데 사용하는 파라미터이다. 부호기는 bqPSNR 값을 사용해서 블록 분할의 기준이 되는 목표 MSE(mean squared errors)를 계산하기도 한다.

bqTopLayer[]는 Y,Cb,Cr 영상의 분해를 시작하는 최상위 레이어의 크기를 나타낸다. 즉. 예를 들어 bqTopLayer[0]이 5라고 하면, Y 영상 분해를 위한 최상위 레이어 블록의 크기는 2^5x2^5임을 의미한다.



(a) 후보 예측신호 없음 (b) 예측신호 1개 존재

(c) 예측신호 2개 존재 (d) 예측신호 3개 존재

(e) 예측신호 4개 존재

그림 2 현재 블록의 예측 부호화에 사용되는 예측신호 후보들

3. 성능평가

TIF 영상과 BMP 영상을 인터넷에서 내려 받아 실험에 사용하였다. 그림 3은 실험에 사용된 25개의 영상이다. 영상 밑의 숫자는 영상의 가로x세로 크기를 나타낸다.

가. R-D 성능

실험 영상은 크게 세 분류로 나뉠 수가 있다. 하나는 애니메이션을

그림 3. 실험영상 (영상 밑의 숫자는 영상의 크기)

포함한 그림이다. 캐릭터나 만화 컷도 이 범주에 들어간다. 다른 하나는 디지털 카메라로 찍은 일반적인 디지털 사진이다. 특히 최근에는 고해상도 디지털 카메라가 보편화되면서 디지털 사진은 크기가 매우 크고, 물체를 정밀하게 표현하는 경우가 많이 있다. 마지막으로는 포스터 영상이 있다. 배경으로 그림을 사용하거나 다른 디지털 영상을 사용하면서 많은 글자나 문양들이 겹쳐져 있는 것이 특징이다.

표 1은 제안하는 쿼드트리 부호화기가 같은 부호율로 JPEG 부호화하는 경우에 비해서 복호영상의 PSNR이 얼마나 좋은지를 영상별로 보여준다. 이 차이를 DPSNR이라고 부른다. DPSNR은 평균적으로 2.25dB (최대 14.87dB, 최소 -3.24dB) 개선됨을 알 수 있다.

표 1. 영상별 DPSNR 성능. (해당 영상은 그림 3과 위치 대응)

1.15	14.87	12.57	0.48	1.08
0.59	1.64	-3.24	0.76	1.05
7.50	5.29	-0.34	0.19	-0.45
1.32	1.68	1.39	-1.83	1.99
1.73	1.20	1.75	0.41	3.58

영상 범주별로 R-D 성능을 비교하면 다음과 같다.

1) 애니메이션 영상

그림 4는 애니메이션 영상에 대한 R-D 성능을 비교해서 보여준다. 애니메이션 영상은 색상의 변화율이 극히 적어서 JPEG의 경우에도 비교적 낮은 비트율로도 매우 높은 PSNR을 얻을 수 있다. 그러나 쿼드트리를 사용하면 압축율을 더욱 높일 수 있음을 확인할 수 있다.

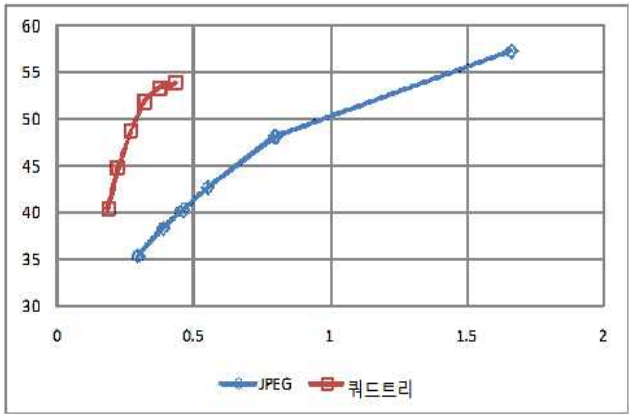


그림 4. 애니메이션 영상에 대한 RD(bpp-dB) 성능 비교

2) 일반적인 디지털 사진

그림 5는 일반적인 디지털 사진에 대한 R-D 성능을 비교한 한 가지 예이다. 낮은 비트율에서는 JPEG과 쿼드트리의 성능이 유사하고, 비트율을 높임에 따라서 쿼드트리가 JPEG보다 우수한 R-D 특성을 보인다. 비트율을 높아갈수록 PSNR의 차이가 점점 더 벌어진다.

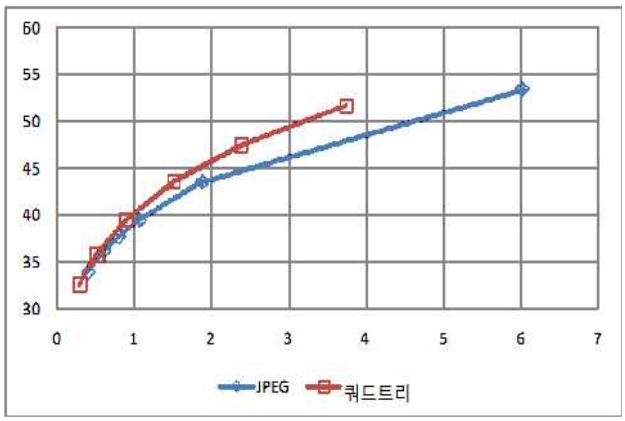


그림 5. 디지털 사진 영상에 대한 RD(bpp-dB) 성능 비교

3) 포스터

그림 6은 포스터 영상에 대한 R-D 성능을 비교한 한 예이다. 기본적으로는 디지털 사진에서와 같은 경향을 보인다. 즉 비트율을 높임에 따라서 쿼드트리가 JPEG보다 더 우수한 R-D 성능을 보인다. 그런데 상대적으로 낮은 부호율에서는 쿼드트리가 JPEG에 비해서 오히려 R-D 성능이 떨어진다.

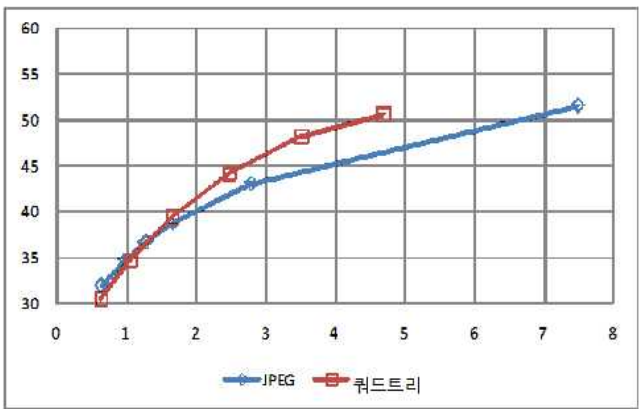


그림 6. 포스터 영상에 관한 RD(bpp-dB) 성능 비교

이상의 결과로부터 제안하는 쿼드트리 부호는 비교적 높은 부호율의 고화질 부호화에 유리함을 알 수 있다. 디지털 사진은 4bpp, 포스터 영상은 5bpp 이하로 50dB 이상의 PSNR을 갖도록 모든 영상을 부호화할 수 있다. 원영상이 24-비트 컬러영상이라고 가정할 때, 대략적으로 6:1 내지 5:1로 압축하여도 매우 뛰어난 화질의 영상을 복원해낼 수 있음을 의미한다. 따라서 초고해상도 디지털 카메라에서 고화질 영상 저장을 위해서는 쿼드트리를 사용하는 것이 기존의 JPEG을 사용하는 것보다 유리하다고 말할 수 있다.

나. 주관적 화질평가

제안하는 쿼드트리 코덱은 R-D 성능이 우수할 뿐만 아니라, 복원영상의 주관적 화질도 우수하다. 그 이유는 시각적으로 중요한 물체의 윤곽을 잘 보존하고 있기 때문이다. 그림 7-9는 JPEG과 쿼드트리로 부호화된 복원영상을 서로 비교해서 보여준다.

그림 7은 비교적 낮은 비트율로 부호화할 때 JPEG에서 나타나는 거짓 윤곽선(false contour) 현상이 쿼드트리에서는 발생하지 않음을 보여준다. 배경 하늘이 채도가 점차적으로 변함에 따라서 JPEG 복원영상에서는 파도 모양의 거짓 윤곽선이 뚜렷하게 발생하고 있지만, 쿼드트리에서는 그러한 현상이 나타나지 않는다.

(a)

(b)

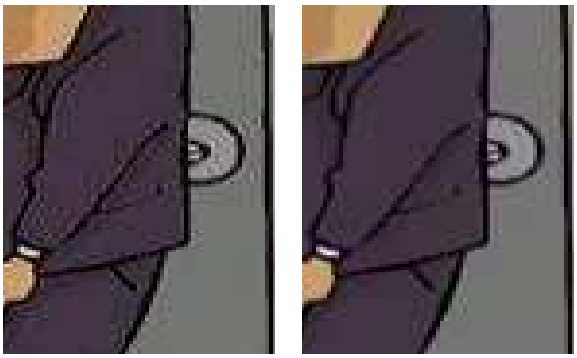
그림 7. 복원화질 비교, (a) JPEG 0.35bpp, (b) QT 0.38bpp.

그림 8과 9는 JPEG에서 발생하는 윤곽선에서의 물결 현상(ringing effect)이 쿼드트리에서는 발생하지 않음을 보여준다. JPEG에서는 물결 현상이 비교적 높은 비트율로 부호화하는 경우에도 여전히 남아있다. 40dB 이상으로 고화질로 부호화한 경우에도 영상을 확대해보면 물결 현상이 분명하게 드러난다. 그러나 쿼드트리의 경우는 물결 현상으로부터는 원천적으로 자유롭다. 즉, 물결 현상의 원인은 매우 강한 에지인데, 강한 에지가 있는 경우에는 에지를 경계로 두 영역을 나누어서 부호화하기 때문에 쿼드트리에서는 원천적으로 물결 현상이 발생하지 않는다.

디지털 사진에서는 JPEG으로 부호화한 경우에도 물결 현상이 뚜렷하게 드러나지는 않는다. 그러나 이 경우에도 영상을 확대하는 경우 숨겨져 있던 물결 현상이 드러나게 된다.

(a) (b)

그림 8. 복원화질 비교, (a) JPEG 0.81bpp, (b) QT 0.81bpp.



(a) (b)

그림 9. 복원화질 비교, (a) JPEG 1.01bpp, (b) QT 1.08bpp.

4. 결 론

컬러영상을 효율적으로 쿼드트리 압축하기 위한 소프트웨어 코덱을 구현하였다. 코덱의 압축 효율은 JPEG보다 약간 우수하고, 에지 주변에서의 물결 현상을 제거함에 따라 주관적 화질은 확연하게 우수하다. 본 논문에서는 Y, Cb, Cr 신호를 독립적으로 쿼드트리 분해하고 부호화하였는데, 이 신호들 사이의 상관성을 이용해서 부호화의 효율을 높일 수 있는 추가적인 연구가 필요하다.

참 고 문 헌

[1] 장호석, 정경훈, 김기두, 강동욱, "에지 선을 보존 하는 쿼드트리 영상 압축 기법" (방송공학회 논문지 심사중)

[2] 채규열, 전병문, 정창성 "에지 검출과 split and merge의 통합에 의한 영상 분할" 한국정보과학회 1996년도 가을 학술발표논문집 제23권 제2호(A), 1996. 10

[3] " http://en.wikipedia.org/wiki/Golomb_coding"