Color table 과 context buffer 를 이용한 color conversion 과 downsampling 기법

채희중° • 이호석
호서대학교 컴퓨터 공학과

Color conversion and downsampling scheme using color table and context buffer

Hui-Joong Chae° • Ho-Suk Lee
Department of Computer Engineering, Hoseo University

요 약

본 논문은 IJG(Independent JPEG Group) JPEG 부호기의 처리 과정중 color table 과 context 버퍼를 이용한 color conversion 과 downsampling 방법에 대해 소개한다. IJG JPEG 은 전처리 과정에서 context buffer 를 사용함으로써 각 컴포넌트(RGB)에 대한 color conversion 과 downsampling 을 효과적으로 수행한다. 또한 각 컴포넌트(RGB) 에 대한 부동 소수점 연산의 처리 결과를 미리 계산하여 color table 에 저장함으로써 color converter 에서 이를 참조, 색차 변환 계산에 적용하도록 하여 처리 속도를 향상 시키고 있다. 이에 본 논문에서는 IJG JPEG 의 부호화 과정에서 사용되는 context 버퍼의 구조와 필요성 그리고 color table 의 구조와 효과에 대하여 소개한다.

1. 서론

IJG(Independent JPEG Group)는 ISO/CCITT 와는 별개의 독립적인 기관으로서 표준안에 근거를 하여 사실상 산업계의 표준으로서 자리잡은 독자적인 JPEG 시스템을 개발하였다[1][4][5].

IJG JPEG 에서 사용하고 있는 특징적인 개념으로는 MCU(Minimum Coded Unit)의 개념을 확장한 iMCU(interleaved MCU)와 rowgroup, context 가 있다.

인간의 시각은 색차(Cb, Cr)값에 대하여 매우 둔감하기 때문에 이러한 특성을 이용하여 이미지 압축시 휘도(Y) 값은 손실하지 않으며 두개의 색차(Cb,Cr)값만을 손실시키는데, 이것을 downsampling 이라고 한다. 이러한 downsampling 을 수행하기 위해서는 입력 이미지의 RGB 값을 YCbCr 값으로 색차를 변환하는 과정이 필요하다. JPEG 부호기에서의 색차 변환은 입력 이미지의 RGB 값을 변환 공식에 의해 계산하여 휘도값(Y)과 색차값(Cb,Cr)으로의 변환을 의미한다[1][4][5].

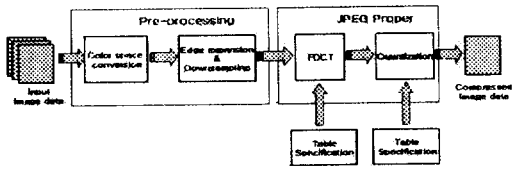
IJG JPEG 부호기에서는 color conversion 과 downsampling 의 처리 속도와 메모리 관리를 개선하기 위한 방법으로서 여러가지 방법이 사용되고 있는데 그 대표적인 방법이 context 버퍼와 color table 을 사용하는 것이다.

IJG JPEG 부호기에서는 부호화 과정시 context 를 사용할 수 있는데 이것은 입력 이미지에 대한 smoothing 팩터를 부여하여 부호화할 때 사용되어진다. smoothing 팩터가 입력 이미지에 적용되어 부호화 할 때는 downsampling 과정에서 한 픽셀 값 출력시 그 픽셀의 주변 8 개의 픽셀 값 들까지 고려하여 downsample 함으로 출력 이미지의 컬러를 부드럽게 출력할수 있도록 해 준다.

2. 본론

2.1 Color conversion

IJG JPEG 의 전체적인 구조는 아래의 [그림 1] 과 같다. color conversion 은 그림에서 보는 바와 같이 IJG JPEG 의 전처리 과정에 포함되며 입력 이미지를 압축하기 위한 첫 번째 과정에 해당된다[1][5].



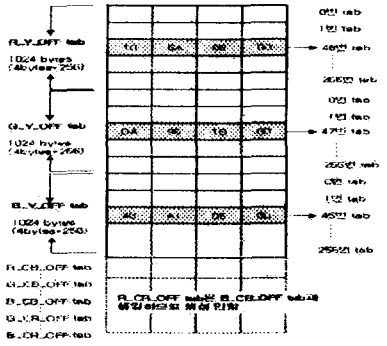
[그림 1] IJG JPEG 의 전체적인 시스템 구조

Color conversion 은 입력 이미지의 폭에 해당하는 scanline 단위로 처리하는데, 이러한 처리는 입력 이미지의 높이 만큼 반복하여 수행한다. Color converter 는 입력 버퍼에 있는 각 scanline 에 대한 RGB 값을 이용하여 [식 1]과 같은 변환식에 의해 RGB 값을 YCbCr 값으로 변환을 한다. CENTERJSAMPLE 은 변환된 Cb, Cr 값이 음수일 경우에 양수로 보정하기 위한 팩터로서 사용되었다.

$$Y = 0.29900 \times R + 0.58700 \times G + 0.11400 \times B$$
$$Cb = -0.16874 \times R - 0.33126 \times G + 0.50000 \times B + CENTERJSAMPLE$$
$$Cr = 0.50000 \times R - 0.41869 \times G - 0.08131 \times B + CENTERJSAMPLE$$

[식 1] RGB→YCbCr 변환식

Color converter 는 위의 공식에 의해 입력 이미지의 RGB 컴포넌트에 대한 YCbCr 값을 바로 계산하지 않고 위의 공식중의 일부(부동 소수점 곱셈부)를 미리 계산하여 color table 을 만들어 놓는다. 그리고 이러한 color table 을 참조하여 각 입력 RGB 컴포넌트에 대한 계산(덧셈부)을 수행한다. [그림 2] 는 위의 변환 공식중 부동 소수점 곱셈을 미리 계산하여 저장하는 color table 의 구조를 설명한다.



[그림 2] Color table 의 구조

[식 1]에서 계산 속도의 결정적인 요인이 되는 부동 소수점 곱셈 부분에 대해 [그림 2] 와 같이 color conversion 계산에 적용할 8 개의 color table 을 생성되었다. 하나의 color table 은 각 scanline 으로부터의 입력 RGB 값에 대해 각 컴포넌트마다 4 bytes 로 구성된 256 개의 color tab 을 만들어 낸다. 각 컴포넌트 당 적용할 오프셋은 모두 9 개의 테이블이 필요하지만 B_CB_OFF 테이블과 R_CR_OFF 테이블의 값은 서로 동등하기 때문에 R_CR_OFF 테이블은 생성하지 않는다.

Color table 에 저장될 계산 결과는 FIX(x) 매크로 연산을 사용하여 처리를 하는데, 이는 부동 소수점 연산을 피하여 처리 속도면에서 빠른 결과를 얻기 위한 방법으로서 bit 이동 연산을 하고 있다. [식 2]는 FIX(x) 매크로 연산에 대한 식을 나타낸다.

FIX(x) = x * (1L<<SCALEBITS) + 0.5
단, SCALEBITS=16 이며, 0.5 는 반올림계수.

[식 2] FIX(x) 매크로 연산식

이 방법은 부동 소수점으로 나타나는 color conversion 계수를 좌측으로 16bit 이동하여 그 계수를 상당히 큰 정수형으로 변환(FIX(x) operation 사용)한 후 색차 변환 공식에 그 계수를 적용하여 계산을 한다. 그리고 모든 계산이 끝난 후에는 그 결과값을 다시 우측으로 16bit 이동하여 부동 소수점으로 표현된 원래의 계수를 적용하여 계산한 결과값과 동일한 결과값을 얻을 수 있다. 이런 방법을 사용하여 얻어진 color table 의 값들은 실제 scanline 단위로 color conversion 처리를 할 때 입력 RGB 값에 따라 color tab 을 참조하며, 선택된 tab 들에 대한 덧셈 연산만을 수행함으로써 보다 빠른 계산 결과를 얻을 수 있다.

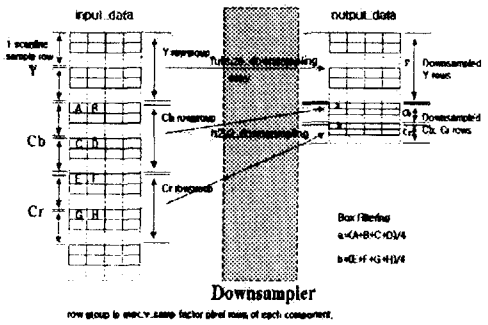
2.2 Downsampling

Downsampling 의 원리는 인간의 시각 시스템이 휘도(Y)신호의 변화에는 민감한 반면에 상대적으로 색차(Cb, Cr)신호의 변화에는 그 보다 덜 민감하게 반응하는 성질을 이용한 압축 알고리즘이다. 시각에 민감한 반응을 나타내는 휘도(Y)신호는 손실하지 않고 그대로 사용하며, 민감하지 않은 색차(Cb, Cr)신호는 원래의 이미지 데이터 양의 1/4 로 줄이는 방법이다(Y:Cb:Cr=4:1:1)[2][4][5].

IJG JPEG downsampling 처리 과정은 YCbCr 로 변환된 rowgroup 단위로 처리하는데, rowgroup 은 각 컴포넌트의 max_v_samp_factor 픽셀의 열들을 가리킨다. Downsampler 는 rowgroup 단위로 입력을 받아서 downsampling 하여 v_samp_factor 의 열들을 형성하게 된다. 그리고 down sampler 는 DCT 의 갯수를 맞추기 위해 이미지에 대한 수평 패딩 작업(expand_right_edge())처리도 수행한다. 반면에 downsampler 를 호출하는 전처리 콘트롤러에서는 수직 패딩 작업(expand_bottom_edge())처리를 수행하고 있다. 각각의 rowgroup 들은 전처리 콘트롤러에서 downsampler 모듈이 호출됨으로서 수행된다. 각 콤포넌트마다 YCbCr 에 대해 각각 두 개씩의 scanline 쌍으로 구성된 rowgroup 을 이루어 처리되어야 하기 때문에 color converter 가 두 번 호출되어 수행될 때 마다 downsampling 루틴은 한 번만 호출되어 수행된다. Downsampler 의 Y 값에 대한 downsampling 은 fullsize_downsample()을 한다. 이것은 단순히 Y(휘도)값을 손실없이 그대로 저장한다. 그런 다음 expand_right_edge()처리를 한다. 반면에 Cb 값과 Cr 값은 데이터 양을 손실시켜야 하기 때문에 h2v2_downsampling ()을 한다. 이것은 Cb 혹은 Cr 값들을 각각 4 개씩 읽어서 그 평균값을 구하여 그 대표값 하나만을 Cb 혹은 Cr 값으로서 저장한다. 그러므로 Y 값에 대한 Cb, Cr 값들은 원래 가지고 있던 데이터 양의 1/4 로 줄어들게 된다. 그리고 2v2_downsampling()은 sampling 을 하기 이전에 expand_right_edge() 처리를 한다.

Downsampler 의 입력 데이터인 YCbCr rowgroup 들의 downsampling 처리후에는 휘도(Y)값은 rowgroup 과 같은 크기를 갖는 rows 가 생성되지만 색차(Cb, Cr)신호 값은 원래의 rowgroup 에서 1/4 로 줄어든 열들이 생성된다. [그림 3] 은 downsampler 의 처리 과정을 나타낸다.

이와 관련된 모듈로는 색차 변환된 YCbCr 값이 scanline 단위로 conversion 버퍼에 각 컴포넌트당 row group 이 형성되었을때 downsampler 를 호출함으로서 스위칭 기능을 제공하는 전처리 콘트롤러가 있다.



[그림 3] Downsampler 의 동작 및 구조

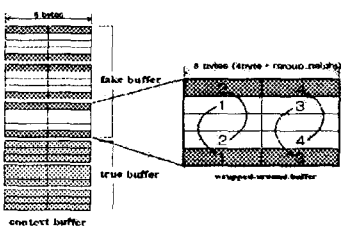
2.3 Context buffer

IJG JPEG 에서 사용하고 있는 주요 특징적인 개념중의 하나인 context 버퍼는 일반적인 경우에는 제공하지 않으며, 컴파일 타임시의 옵션으로서 프로그램 컴파일시 context_row_need()를 선택하여야 사용할 수 있다. 이러한 context 버퍼는 smoothing 팩터를 사용할 때 사용하고 있는데 smoothing 팩터의 사용은 입력된 color-dithered GIF 이미지를 JPEG 이미지로 보다 선명하게 부호화할 수 있는 기능을 제공하고 있다. 이는 입력 이미지의

컬러를 보다 부드럽게, 즉 주변 픽셀과의 색차를 최소화 함으로서 자연스럽게 보이도록 부호화하여 이미지에 대한 출력 성능을 조절할 수 있는 기능이다. IJG JPEG 에서는 기본적으로 지원하지 않으나 컴파일 타임시 제공되는 옵션으로서 smoothing 팩터를 제공하고 있다. Smoothing 팩터 값으로는 0~100 까지 지원 가능한데 0 는 smoothing 기능을 제공하지 않음을 표시한다. 적절한 크기의 smoothing 팩터의 부여는 입력된 이미지에 대해 매우 부드럽고 깨끗한 출력 이미지의 부호화가 가능하지만 너무 큰 팩터를 부여했을 경우에는 마치 입력 이미지에 blur 효과를 준 것처럼 희미하고 뿌연 출력 이미지를 생성한다.

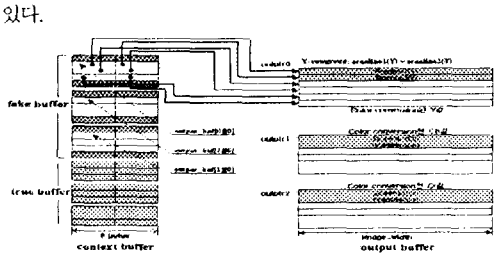
Context row 를 지원한다면 기본적으로 context 버퍼를 생성해야만 하는데 context 버퍼는 fake 버퍼와 true 버퍼로서 구성되어 있다. True 버퍼는 실제적인 입력 데이터, 즉 color conversion 에서는 RGB 값의 픽셀 scanline 의 데이터, 그리고 downsampling 모듈에서는 rowgroup 단위로 color conversion 된 YCbCr 데이터가 있는 버퍼들에 대한 포인터들의 배열이다.

[그림 4] 는 create_context()모듈을 통해 생성된 context 버퍼의 구조를 나타낸다. [그림 4] 에서 보는 바와 같이 context 버퍼는 각 컴포넌트마다 8byte*5rows 로 구성된 fake 버퍼와 실제 입력 데이터의 버퍼포인터 배열인 8byte* 컴포넌트수(3)의 true 버퍼로 구성되어 있다. fake 버퍼 가운데 3 개의 열은 true 버퍼를 그대로 복사한 열들이며 위와 아래의 열에는 true 버퍼의 마지막과 첫번째의 열을 가리키는 포인터를 둠으로서 fake 버퍼는 둥글게 감싼 것과 같은 구조로 되어있다. Downsampler 는 이러한 fake 버퍼에 포함되어 있는 각 컴포넌트 당 할당되어 있는 각각 6 개의 true 버퍼 포인터 배열을 통하여 처리후 변환된 값들을 저장하기 위한 출력 버퍼를 지정할 수 있다.



[그림 4] context 버퍼의 구조

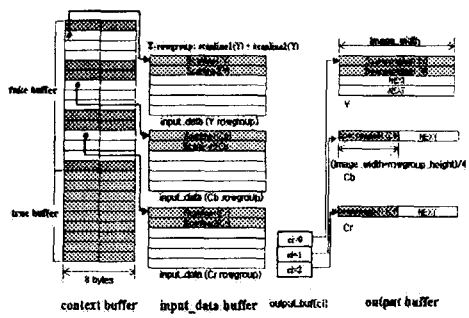
[그림 5] 는 context 버퍼를 사용하여 color conversion 하는 동작과 버퍼의 구조를 나타낸다. fake 버퍼의 2,3,4 번째 3 개의 열들(8byte*3rows)은 color conversion 된 Y 값을 저장하는 출력 버퍼를 포인터하고 있다.



[그림 5] context 버퍼를 이용한 color conversion 동작

각 픽셀들의 scanline 으로부터 읽혀진 RGB 값은 적절한 계산을 거친후에 그림에서 보는 바와 같이 입력 이미지의 폭에 해당하는 길이를 갖는열들을 입력 이미지의 높이에 해당하는 수 만큼 만들어 출력 버퍼에 저장한다.

Downsampler 는 rowgroup 단위로 처리를 하기 때문에 color conversion 처리 과정을 통해 계산되어 출력 버퍼에 저장되어 있는 YCbCr 각 컴포넌트 의 row group(2-rows)을 출력 버퍼로 하여 출력 버퍼에 저장한다. [그림 6] 은 context 버퍼를 사용한 downsampling 처리 과정과 버퍼의 구조를 나타낸다.



[그림 6] context buffer 를 이용한 downsampling 동작

3. 결론

본 논문에서는 IJG JPEG 의 context 버퍼의 구조와 context 버퍼를 사용한 color conversion, downsampling 을 살펴 보았다.

Color table 은 변환 공식에서 계산 시간에 결정적인 요인이 되는 부동 소수점 연산에 대한 계산 결과를 미리 color table 로 만들어 저장함으로서 color conversion 시 이를 참조하여 결과를 빠르게 산출할 수 있는 기능을 제공한다.

Context 버퍼의 장점은 입력 이미지의 컬러를 부드럽게 조절할 수 있는 smoothing 팩터의 부여를 가능하게 한다. 또한 메모리를 효율적으로 관리하고 사용할 수 있는 기능을 제공한다.

이와 같이 color table 과 context 버퍼는 IJG JPEG 의 시스템 구조상 큰 특징이자 장점으로서 MPEG-2, MPEG-4 에의 적용이 기대된다.

4. 참고문헌

[1] Thomas G. Lane, 'IJG(Independent JPEG Group)'s JPEG Library: Version 6a', *Independent JPEG Group, 1996.*

[2] CCITT Recommendation T.81: PART 1 "Information Technology: Digital Compression and Coding Of Continuous-Tone Still Images: Requirements and Guidelines", *ITU(International Telecommunication Union), 1993.*

[3] Gregory K. Wallace, 'The JPEG Still Picture Compression Standard', *Communication Of the ACM , Vol. 34 No. 4, pp. 30-44, April 1991.*

[4] William B. Pennebaker and Joan L. Mitchell, "JPEG Still Image Data Compression Standard', *Van Nostrand Reinhold, 1993.*

[5] 채희중, 이호석, "IJG JPEG 부호기의 구조와 동작", *한국정보과학회 99 가을학술발표논문집 제 26 권 2 호 p262-264, 한국정보과학회 , 1999.10.*