초고화질 해상도용 4K/8K JPEG 2000 고속 압축 처리를 위한 연구

*백봉진 **류준호 ***김정길 ****이상운

남서울대학교

***cgkim@nsu.ac.kr

A Study of Fast 4K/8K JPEG 2000 Compression for Ultra High Definition Resolution

*Baek, Bongjin **Ryu, Joonho ***Kim, Cheong Ghil ****Lee, SangWoon

Namseoul University

요약

멀티미디어 기술의 보급 확산 및 급속한 발전으로 새로운 영상 압축 기술인 HEVC(High Efficiency Video Coding) 고화질 영상 압축 표준을 탄생시켰으며, 사용자 또한 대형 화면에 대한 선호도가 높아지고 있다. 그 결과 기존의 HD급 영상보다 4배 이상, 16배까지 선명한 초고화질 UHD(Ultra High Definition) 영상 서비스가 차세대 방송기술로 새롭게 주목받고 있다. 또한 JPEG 2000 압축도 기존 처리된 4096 x 4096 픽셀 이미지를 넘어 초고화질 해상도 이미지(8K : 7680 x 4320 혹은 8192 x4320 픽셀)를 처리 지원을 하고 있다. 따라서 초고화질 이미지의 획득 및 저장을 위해서는 고속의 처리 기술이 필요하다. 이에 본 논문은 초고화질 해상도 이미지의 고속 처리를 위한 병렬처리 기술에 대해 단계적 연구를 실행하며, 이를 위하여 1차적으로. JPEG 2000의 처리 과정을 살펴보고 전처리 단계인 색공간 변환 알고리즘 적용을 위하여 사용자 정의의 쓰레드 기반 고속 처리를 수행하였다. 실험 결과 기존의 처리보다 사용자 정의 기반 쓰레드 고속처리가 초고화질 해상도 이미지(UHD 8K : 7680 x 4320)를 기준으로 최대 15배의 성능 향상의 결과를 보여주었다.

1. 서론

최근 HDTV(Full-HD급) 영상 콘텐츠가 대중화되고, 시청자들이 고화질, 고해상도의 영상에 익숙해지면서 HDTV 이후의 차세대 영상 서비스에 대한 관심이 증가하고 있는 가운데, UHD 콘텐츠는 Post-HD 미디어 시대의 주류로 부상하고 있으며, 점차로 콘텐츠 제작 환경도 UHD급으로 진화하고 있다. 그 결과 그림 1의 영상 데이터량 비교에서 확인하듯이 현재의 HDTV 서비스에서 콘텐츠 획득과 제작 분야에서 주로 사용되는 영상 데이터 량은 약 2.5Gbps 이상 필요하지만, 초고품질 4K UHDTV 서비스에 대응하기 위해서는 기존 Full-HD 데이터에 비해 최소 4배 정도인 10Gbps 데이터량 이상을 처리해야 하고, 8K UHDTV 서비스에 대응하기 위해서는 기존 Full-HD 데이터에 비해 약 29배 또는 58배 정도의 데이터 처리 기술이 필요하다[1,2].

이러한 상황의 극복은 컴퓨터의 성능을 향상시키려는 연구와 병행되어 꾸준히 계속되고 있으며, CPU의 처리 속도를 높이거나, CPU의 개수를 늘려 병렬로 작업을 처리하여 컴퓨터의 성능을 향상시키려는 방법들이 대표적인 예이다. 그 결과 최근 멀티코어 프로세서들이 범용 PC 뿐만 아니라 임베디드 시스템에서도 탑재될 만큼 그 사용이 보편화되고 있는 상황에서, 많은 멀티미디어 응용 프로그램이 이들을 활용하여 병렬화 되고 있다[3].

그러나 하드웨어적인 향상만으로 컴퓨터의 병렬 처리가 불가능하며, 이를 지원하기 위한 병렬처리 소프트웨어도 반드시 필요하다, 기존의 일번적인 프로그램을 개발하는 방법은 주로 단일 코더 CPU를 기준으로 하는 구현 방법이다, 따러서 멀티 코어를 지원하는 CPU의 성능을 충분히 활용하기 위해서는 새로운 프로그램 모델이 필요하다. 가장 일반적인 병렬처리 방법은 여러 개의 쓰레드(thread)를 생성하여, 각자의 프로세스에 역할을 할당하는 방법이다.

	해상도(WxH)	프레임율(fps)	컬러포맷(YUV)	비트심도(bits)	데이터량(Mbps)
SD	720x480	30	4:2:0	8	124Mbps
HD	1920x1080	30	4:2:2	8	996Mbps
	1920x1080	30	4:2:2	10	1.25Gbps
	1920x1080	60	4:2:2	10	2.5Gbps
4K UHD	3480x2160	30	4:2:2	8	4Gbps
	3480x2160	60	4:2:2	10	10Gbps
	3480x2160	60	4:4:4	12	18Gbps
8K UHD	7680x4320	60	4:2:2	10	40Gbps
	7680x4320	60	4:4:4	10	60Gbps
	7680x4320	60	4:4:4	12	72Gbps
	7680x4320	120	4:4:4	12	144Gbps

그림 1. UHD콘텐츠용 영상 데이터

이에 본 논문은 초고화질 해상도 이미지의 고속 처리를 위한 병렬처리의 단계적 연구를 위하여 1차적으로. JPEG 2000의 처리 과정을 살펴보고 전처리 단계인 색공간 변환 알고리즘 적용을 위하여 사용자 정의의 쓰레드 기반 고속 처리를 수행하여 성능 향상의 결과를 살펴보고자 한다, 본 논문의 구성은 2장에서는 JPEG 2000 및 쓰레드에 대해

소개하고 3장에서는 병렬처리 적용를 기술한다. 4장에서는 제안하는 실험 환경 및 결과를 기술하고, 마지막으로 5장에서 결론을 맺는다.

2. JPEG 2000과 쓰레드

1992년 JPEG(Joint Photographic Expert Group)이 국제표준으로 채택된 이후 다양한 멀티 미디어응용분야에 사용되고 있다. ISO/IEC 산하의 JTC1/SC29/WG1 그룹에서는 2000년 JPEG2000이라 하는 새로운 표준을 발표하였다[4]. JPEG2000 인코더/디코더의 블록 그림은 그림 2와 같다. 이산 웨이블렛 변환을 소스 영상 데이터에 적용되고 변환된 계수는 양자화를 거쳐 코드열을 생성하기 전에 엔트로피 부호화 과정을 거친다. 전처리 단계는 색변환 단계로서 RFB에서 YUV 변환을 실행한다[5],

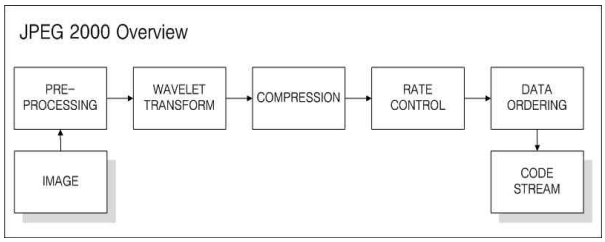

그림 2. JPEG 2000 블럭도

스레드(thread)는 한 개의 프로그램 또는 프로세스 내에서 실행되는 흐름의 단위로서, 일반적으로 한 프로그램은 하나의 스레드를 가지고 있지만, 프로그램 환경에 따라 둘 이상의 스레드를 동시에 실행 가능한 멀티스레드(multithread) 방식이 가능하다. 멀티스레드는 프로세스 내의 메모리를 공유하며 따라서 고속의 프로세스 간의 전환이 가능하다. 그림 3은 싱글쓰레드와 멀티쓰레드 모델을 보여준다.

특히, 멀티 코아 환경에서의 멀티스레드는 각각의 CPU가 스레드 하나씩을 담당하는 방법으로 속도를 높일 수 있다. 이러한 시스템에서는 여러 스레드가 실제 시간상으로 동시에 수행될 수 있기 때문이다. 사용자 스레드는 커널 영역의 상위에서 지원되며 일반적으로 사용자 레벨의 라이브러리를 통해 구현되며, 라이브러리는 스레드의 생성 및 스케줄링 등에 관한 관리 기능을 제공한다.

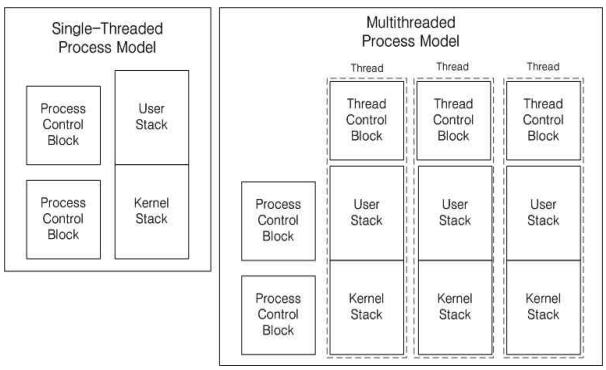

그림 3. 쓰레드 실행 모델

3. 병렬처리

RGB는 일반적으로 사용하는 색상 구조로서 3색상의 조합으로 밝기는 없고 모두 색상으로 구성되 된다, YUV는 Video Data 형식으로 사용하며 TV에서 컬러페이스로 전송하는 컬러 형식이다. YUV는 색과 빛이 별도로 구성되어 있는 형식으로 RGB방식에 비하여 작은 대역폭으로 전송이 가능한 장점이 있다. 그림 4는 색변환의 계산 과정을 보여준다.

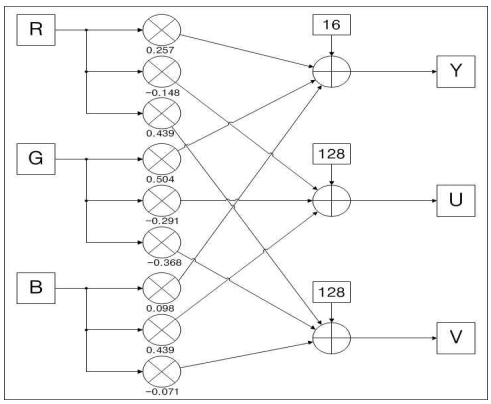

그림 4. 색변환 블럭도

쓰레드란 그림 5에서 보듯이 프로그램 내에서 실행되는 흐름의 단위이다. 모든 프로그램은 최소 하나의 쓰레드를 가지며, 이 쓰레드를 Main Thread(주 쓰레드)라고 한다. 이 Main Thread에서 Main Routine이 호출된다. 이렇게 프로그램은 여러 개의 쓰레드를 동시에 실행할 수도 있고, 이것으로 인해 흐름이 동시에 진행될 수 있다.

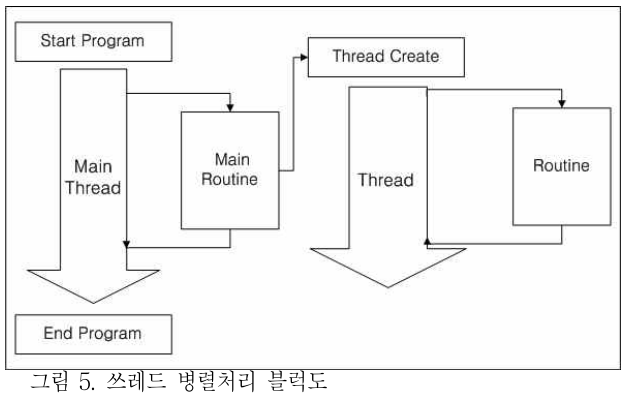

그림 5. 쓰레드 병렬처리 블럭도

그림 6는 DibColorSplitYUV() 함수로서 읽어들인 bmp 이미지를 RGB색공간에서 YUV색공간으로 변환하는 함수이다. 그림의 1번 박스는 읽어들인 bmp 이미지의 크기를 변수w,h에 저장하고, 그레이스케일하여 색공간 정보를 각가의 ptr변수에 저장한다. 마지막으로 구조체 struct argment에 각종 정보를 저장 한다. 2번 박스는 쓰레드 생성 및 실행 그리고 3번 박스는 쓰레드 대기 및 종료를 수행하는 블록이다.

```
void DibColorSplitYUV(CDib& dib, CDib& dibY, CDib& dibU, CDib& dibV)
{
    double start = 0;
    double end = 0;
    double seconds = 0;
    double sum_seconds = 0;
    double average_seconds = 0;

    start = omp_get_wtime( );

    file = fopen("Thread_DataResult_YUV.csv","a+");

1   int w = dib.GetWidth( );
    int h = dib.GetHeight( );

    dibY.CreateGrayImage(w, h);
    dibU.CreateGrayImage(w, h);
    dibV.CreateGrayImage(w, h);

    RGBBYTE** ptr = dib.GetRGBPtr( );
    BYTE** ptrY  = dibY.GetPtr( );
    BYTE** ptrU = dibU.GetPtr( );
    BYTE** ptrV = dibV.GetPtr( );

    struct argment arg;
    arg.w = w;
    arg.h = h;
    arg.ptr = ptr;
    arg.ptrY = ptrY;
    arg.ptrU = ptrU;
    arg.ptrV = ptrV;

2   HANDLE hThread[NUM_THREAD];
    DWORD dwThreadID = NULL;
    hMutex = CreateMutex(NULL, FALSE, NULL);

    for(int w=0; w<NUM_THREAD; w++){
        hThread[w] = (HANDLE)_beginthreadex(NULL,0,&ThreadFunction,&arg,0,NULL);
    }

    end = omp_get_wtime( );
    seconds = end - start;

3   fprintf(file, "Thread_RGB_to_YUV : %f \n\n", seconds);

    fclose(file);
    WaitForMultipleObjects(NUM_THREAD,hThread,TRUE,INFINITE);
    CloseHandle(hMutex);
}
```

그림 6. 쓰레드 처리 코드

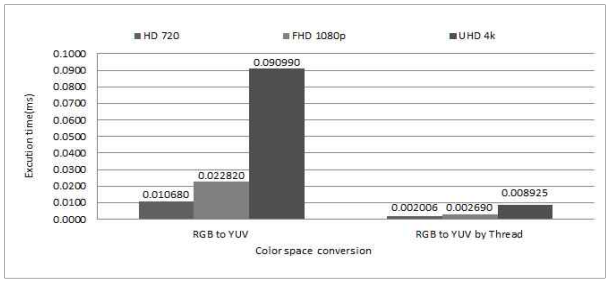

그림 7. 쓰레드처리에 따른 HD, FHD, UHD의 색공간 변환

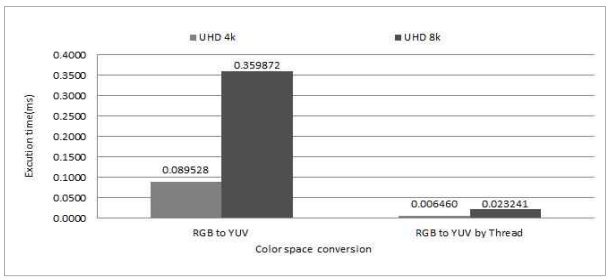

그림 8. 쓰레드처리에 따른 UHD 4k, 8k의 색공간 변환

4. 실험 및 결과

본 장에서는 실험 환경 및 방법을 소개하고 결과를 기술한다. 표 1은 실험 환경을 요약하고 있으며 실험 이미지는 BMP 파일 포맷을 사용하였다. 실험은 해당 이미지에 대하여 RGB에서 YUV로 컬러 공간 변환 연산에 소요되는 시간을 측정하였으며 최종 결과 값은 300회 반복 하여 측정하여 평균값을 구하였다. 연산 속도 측정 구간은 이미지의 모든 픽셀에 대한 컬러 변환 연산이 완료될 때까지의 계산 시간을 측정하였다. 초기 ImageLoad 과정은 동일한 프로시저를 사용함으로 연산 속도 측정 부분에서 제외하였다. 속도 측정 방법은 시스템 타임을 사용하였으며 초(sec) 단위로 표시하였다. 쓰레드를 사용하지 않은 경우 4K, 8K의 RGB to YUV 변환의 연산 시간은 각각 평균 약 0.089528sec와 0.359872sec 소요되었으며, 쓰레드를 사용한 RGB to YUV 변환의 연산시간은 4K, 8K에서 각각 평균 0.006460sec와 0.02341sec가 소요되어 10배 이상의 속도 향상을 가져왔다. 그림 7과 그림 8은 각각의 컬러 공관 변환의 속도 성능 비교를 보여준다.

표 1. 실험환경

시스템 사양	구성	
CPU	Intel Core i5-3570@3.40Ghz	
RAM	16.00GB	
System	Window7 Ultimate SP1(64bit)	
Compiler	Microsoft Visual Studio 2012	
Image Sizes	HD 720	1280x720
	FHD 1080p	1920x1080p
	UHD 4k	3840x2160
	UHD 8k	7680x4320
Image Type	bmp	

5. 결론

본 연구에서는 4096 x 4096 픽셀 이미지를 넘어 초고화질 해상도 이미지(8K : 7680 x 4320 혹은 8192 x4320 픽셀)를 처리 지원을 위한 고속 JPEG 2000 압축의 연구를 수행하였다, 이를 위하여 1차적으로, JPEG 2000의 전처리 단계인 색공간 변환 알고리즘 적용을 위하여 사용자 정의의 쓰레드 기반 고속 처리를 수행하였다. 실험 결과 기존의 처리보다 사용자 정의 기반 쓰레드 고속처리가 초고화질 해상도 이미지(UHD 8K : 7680 x 4320)를 기준으로 최대 15배의 성능 향상의 결과를 보여주었다. 본 연구는 전체 JPEG 2000 과정으로의 적용과 사용자 정의 기반 쓰레드 정의 기법과 기존 라이브러리 형태의 쓰레드 병렬 기법과의 성능 비교를 수행 계획이다.

감사의 글

본 연구는 미래창조과학부 및 정보통신기술진흥센터의 정보통신·방송 연구개발사업의 일환으로 수행하였음. [2014(I5501-14-1007), 3D 스마트미디어/증강현실 기술 한중일러 공조 국제표준화]

참고문헌

[1] 김제우, 김동순, 신화선, 최병호, "UHD(Ultra High Definition) 콘텐츠용 실시간 획득, 저장 및 편집 시스템 기술, 방송공학회지 제17권 제4호, pp. 69-80, 2012년 10월.

[2] 배성호, 하광성, 김문철, 조숙희, 최진수, HEVC 기반 초고선명(4K-UHD) 비디오 시청품질 특성 분석, 2012년도 한국방송공학회 하계 학술대회, pp. 446-449, 2012년 7월.

[3] Cheong Ghil Kim, Do Hyun Lee, JeomGu Kim, "Optimizing

Image Processing on Multi-core CPUs with Intel Parallel Programming Technologies," Multimedia Tools and Applications January 2014, Vol. 68, Issue 2, pp. 237-251, Jan. 2014.

[4] A. Skodras, C. Christopoulos, and T. Ebrahimi, "The JPEG2000 Still Image Compression Standard," IEEE Signal Processing Magazine, Vol. 18, Issue. 5, pp. 36-58, Sep.2001.

[5] 김재준, 홍진우, JPEG2000 정지영상 부호화 기술 개요, 전자통신동향분석 제17권 제4호, pp. 65-74, 2002년 8월.