

Daubechies 웨이블릿 필터를 사용한 볼륨 데이터 인코딩

허 영 주[†] · 박 상 훈^{††}

요 약

데이터 압축 기술은 대용량의 데이터를 효율적으로 저장하고 전송할 수 있게 해주는 기술로, 요구되는 데이터의 용량이 커지고 네트워크의 트래픽이 증가함에 따라 그 중요도가 점점 더 높아지고 있다. 특히 다양한 응용과학과 공학 분야에서 산출되는 볼륨 데이터는 컴퓨팅 기술의 발전에 힘입어 그 용량이 점점 더 증가하는 추세에 있다. 본 논문에서는 Daubechies 웨이블릿 변환을 적용해서 볼륨 데이터를 압축하는 기법을 제안한다. 구현된 D4 웨이블릿 필터 기반 압축 기법은 3차원 볼륨 데이터에 대한 손실 압축과 블록 단위의 무작위 추출 복원을 지원한다. 본 기법은 기존의 Haar 필터를 이용한 압축 방식에 비해 복원 데이터의 손실율이 낮기 때문에, 정밀한 복원 영상이 중요시되는 대용량 데이터의 압축 및 인터랙티브 가시화 응용에 유용하게 사용될 수 있다.

키워드 : Daubechies 웨이블릿, 과학적 가시화, 3D 볼륨 데이터 압축, 무작위 추출

Volumetric Data Encoding Using Daubechies Wavelet Filter

Youngju Hur[†] · Sanghun Park^{††}

ABSTRACT

Data compression technologies enable us to store and transfer large amount of data efficiently, and become more and more important due to increasing data size and the network traffic. Moreover, as a result of the increase of computing power, volumetric data produced from various applied science and engineering fields has been getting much larger. In this paper, we present a volume compression scheme which exploits Daubechies wavelet transform. The proposed scheme basically supports lossy compression for 3D volume data, and provides unit-wise random accessibility. Since our scheme shows far lower error rates than the previous compression methods based on Haar filter, it could be used well for interactive visualization applications as well as large volume data compression requiring image fidelity.

Key Words : Daubechies Wavelet, Scientific Visualization, 3D Volume Data Compression, Random Access

1. 서 론

사용자가 필요로 하는 데이터의 용량이 커짐에 따라, 그리고 네트워크 사용량이 증가함에 따라 데이터 압축 기술에 대한 필요성 역시 증가해 왔다. 데이터 압축 기술은 적용 분야와 데이터의 특성에 따라 그 기법이 매우 다양한데, 본 논문에서는 과학적 가시화(scientific visualization) 분야에서 사용되는 방대한 볼륨 데이터를 효과적으로 압축하기 위한 인코딩 기법을 제안한다.

일반적으로 볼륨 데이터는 자연과학과 공학의 다양한 분야에서 시뮬레이션의 결과로 산출되거나 정밀한 측정기기(의학에서 CT, MRI PET 장비 등)를 사용하여 얻어지며, 이러한 3D 수치 데이터를 효율적으로 분석하기 위한 다양한 가시화 기법에 관한 연구 결과가 계속 발표되고 있다.

최근 센서기술과 시뮬레이션 기술의 발전으로 각 분야에서 산출되는 볼륨 데이터의 크기가 점점 대형화되는 추세에 있으며, 이런 데이터들의 대부분은 전체 데이터를 컴퓨터 주기억장치(RAM)에 로딩하는 것이 불가능할 정도로 큰 크기를 갖는다. 따라서 효과적인 볼륨 데이터 압축 기법과 압축 방식의 표준화에 대한 연구의 중요성은 점점 더 부각되고 있지만, 가시화를 위한 볼륨 데이터 인코딩에 관한 연구는 실제로 그리 활발하게 이뤄지고 있지는 않다. 압축 기법은 크게 무손실 압축(lossless compression)과 손실 압축(lossy compression)으로 분류할 수 있다. 가시화를 목적으로 하는 볼륨 데이터의 경우, 원본 데이터의 자체 크기가 방대할 뿐만 아니라 원본 데이터와 복원 데이터 사이에 약간의 오차가 있다하더라도 렌더링 결과로 생성된 이미지에 영향을 그리 크지 않기 때문에 압축률이 높은 손실 압축 기법을 많이 적용한다.

본 논문에서는 3차원 볼륨 데이터에 대한 새로운 손실 압축 기법을 제안한다. 이 기법은 Daubechies 웨이블릿(wavelet) 필

[†] 정 회 원 : 한국과학기술정보연구원(KISTI) 연구원
^{††} 정 회 원 : 동국대학교 영상대학원 멀티미디어학과 조교수
 논문접수 : 2006년 8월 4일, 심사완료 : 2006년 10월 23일

터를 이용하기 때문에 Haar 필터를 사용한 기존 볼륨 데이터 압축 방식에 비해 동일한 압축률에서 우수한 복원화질을 나타낸다. 또한, 주로 정수 데이터 압축에 초점을 맞춘 기존 방식과는 달리 이 기법은 부동 소수점(single precision floating-point) 데이터에도 적용 가능하도록 설계됐으며, 렌더링 코드와의 연계를 고려하여 블록 단위의 무작위 추출(random access)이 가능한 인코딩 스킴(encoding scheme)을 구현했다.

본 논문은 다음과 같이 구성된다. 2장에서는 볼륨 데이터 압축과 관련된 기존 연구 결과들을 정리하고, 3장에서는 본 논문에서 제안한 Daubechies D4 필터 기반 인코딩 기법에 대해 설명한다. 4장에서는 본 기법의 압축 및 복원 결과를 나열한 뒤, 5장의 결론으로 끝을 맺는다.

2. 관련 연구

데이터 압축에서는 데이터의 중복도가 높거나 분포하는 범위가 좁을수록 더 좋은 압축률을 얻을 수 있다. 따라서 인코딩에 앞서 다른 영역(domain)으로 데이터를 변환, 데이터의 분포 범위를 줄이거나 압축에 직접 영향을 미치는 유효 데이터의 개수를 줄임으로써 데이터 압축률을 높일 수 있다. 이런 용도로 사용되는 데이터 변환 기법은 그 종류가 매우 다양하며 DCT(Discrete Cosine Transform) 변환이나 웨이블릿 변환 기법이 많이 사용된다. 특히 웨이블릿은 계산이 용이하고 기저(basis)가 다양하다는 특징이 있기 때문에 많이 사용되며, 압축분야 뿐만 아니라 데이터 필터링, LOD(Level Of Details) 등 다양한 분야에 사용되고 있다. 웨이블릿 변환에 기반을 둔 압축 기법에서는 기저(basis)에 따라 압축률, 복원 화질, 복원 속도 등에 차이가 생긴다. 압축 기법에 많이 사용되는 웨이블릿 기저로는 Haar 웨이블릿을 들 수 있는데, Haar 웨이블릿은 변환 연산이 간단하고 연산 비용이 적다는 장점을 갖기 때문에 복원 속도를 중시하는 실시간 응용 분야에서 활용되고 있다.

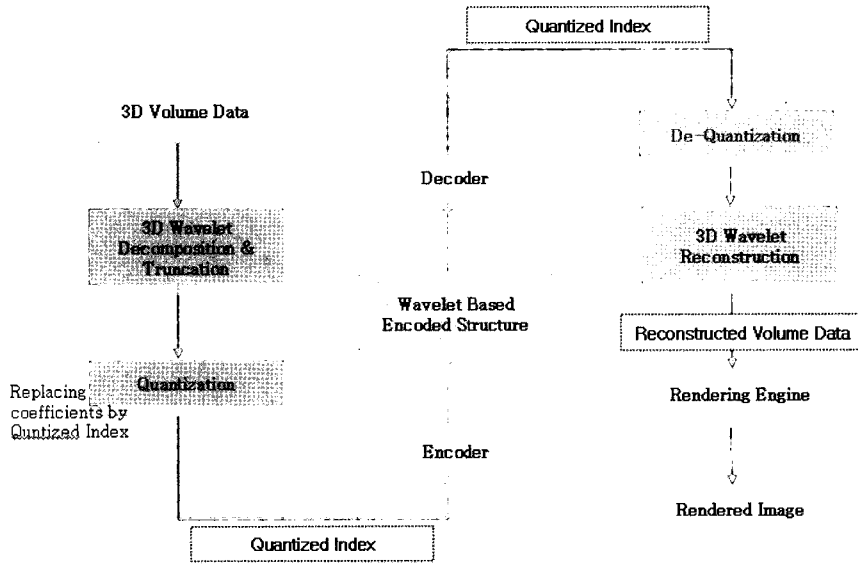
웨이블릿 변환에는 Haar 웨이블릿 기저함수 외에도 여러 다양한 기저 함수가 존재한다. 그 중, Daubechies[4][5] 웨이블릿을 사용하면 손실 압축에서 Haar 웨이블릿보다 복원 데이터의 손실을 줄일 수 있다. 이 기저함수군은 2개에서 20개의 계수를 사용한 데이터 변환방식을 제공하는데, 그 중, 4개의 계수를 사용하는 D4 방식은 계산이 비교적 간단하면서도 복원 화질의 손상도가 낮다. D4 웨이블릿 역시 Haar 웨이블릿과 마찬가지로 다양한 데이터에 적용 가능하며, 데이터 변환시에 이를 적용함으로써 압축률을 높이고 복원 화질을 향상시키는 것이 가능하다.

2D 이미지 데이터 압축에서는 웨이블릿을 이용한 데이터 변환 기법이 많이 사용되고 있다. [15]와 [18]에서는 웨이블릿을 적용하여 생성된 계수들의 데이터 계층 구조를 이용한 인코딩 방식을 소개했으며, [17]에서는 비트 스트림에서의 비트의 중요도에 따라 인코딩의 우선순위를 달리하는 EBCOT이라는 방식을 소개했다. EBCOT은 JPEG2000[9]에서 표준으로 채택한 인코딩 방식이기도 하다.

웨이블릿은 3차원 볼륨 데이터의 압축에도 다양하게 활용되었다. 볼륨 데이터 압축기법은 중에서는 웨이블릿 변환을 활용한 기법을 몇 가지를 찾아볼 수 있는데, 대부분의 압축 방식은 본격적으로 인코딩 기법을 제안하는 대신, 변환된 웨이블릿 계수에 런-LENGTH(run-length)나 허프만(Huffman) 코딩같은 기존 인코딩 기법을 적용했다. 특히 [6]은 볼륨 데이터를 애니메이트해서 여러 개의 프레임을 생성한 뒤, Haar, Daubechies D4, CDF 등, 다양한 웨이블릿 필터를 적용하여 변환하고 LZH나 산술(arithmetic) 코딩과 같은 기존의 인코딩 방식을 적용했다. [7]에서는 Daubechies D4 필터를 사용, 데이터를 변환한 뒤, 오차 범위를 설정함으로써 광선추적법(ray-casting)의 속도를 높이는 기법을 제안했으며, [12]에서는 Haar, D4, D6로 볼륨 데이터를 변환한 뒤, 복셀(voxel)의 중요도에 따라 데이터를 절삭(truncation)함으로써 볼륨 데이터 전송시 소요되는 네트워크 대역폭을 낮추는 방법을 제안했다. 여기에서도 별도의 독창적인 인코딩 기법은 제안되지 않았다.

Haar 웨이블릿을 사용한 몇몇 3차원 볼륨 데이터 압축 기법에서는 볼륨 데이터에 특화된 인코딩 기법을 제안했다. [10]과 [11]에서는 Haar 웨이블릿을 3차원 볼륨 데이터에 적용하는 변환식을 소개했다, [8]에서는 [10]의 변환식을 응용, 최적화된 Haar 웨이블릿 변환식을 소개하면서 웨이블릿 계수의 성질을 활용한 제로비트(zerobit) 인코딩 기법을 제안했다. 이 방법은 임의의 위치에 존재하는 복셀 데이터를 빠른 속도로 복원할 수 있으며, 여타의 다른 방식에 비해 압축률이 높다. 그러나 이 방식은 12비트의 정수로 표현되는 복셀들로 구성된 볼륨 데이터에만 적용 가능하며, 높은 압축률의 데이터를 사용하는 경우 Haar 기저의 사용으로 인한 화질 저하 현상이 나타난다. [14]에서는 각 슬라이스간의 데이터 연관성과 Haar 웨이블릿을 이용한 인코딩 기법을 제안했지만, 이 역시 최대 2마이트로 저장 가능한 정수로 구성된 볼륨 데이터에만 적용 가능하다.

정수 데이터에 대한 인코딩 기법은 [1]과 [2]에서 제안한 바 있으며, 이 기법을 통해 D4 필터와 Haar 필터의 성능을 비교, D4 필터의 우수성을 입증한 바 있다. 본 논문에서는 [1]과 [2]에서 제안한 압축 방식을 확장하여 부동 소수점 데이터에 적용할 수 있는 새로운 인코딩 기법을 제안한다. Daubechies D4 필터를 사용한 대부분의 기존 압축 기법들이 새롭고 효율적인 인코딩 스킴의 제안보다는 이미 개발된 데이터 인코딩 기법을 그대로 또는 혼합해서 사용하고 있는 것과는 달리, 본 논문에서는 볼륨 데이터 렌더링에 특화된 새로운 인코딩 기법을 제시한다. 또한, 기존에 제안된 대부분의 볼륨 데이터 인코딩 기법이 정수 데이터에만 적용 가능하다는 점을 개선, 볼륨 데이터에서 많이 나타나는 부동 소수점 데이터에도 적용 가능한 기법을 제시한다. 본 기법에서는 D4 필터를 사용함으로써 Haar 필터를 사용한 기존의 기법에 비해 화질 개선 효과를 얻을 수 있으며, 블록 단위의 무작위 추출을 지원하는 디코딩 스킴을 구현함으로써 볼륨 렌더링 코드에 직접 적용이 가능하다.



(그림 1) 웨이블릿 변환을 이용한 데이터의 압축 및 복원

3. Daubechies D4 기반 인코딩

웨이블릿 변환을 이용한 3차원 볼륨 데이터의 압축 및 복원의 전체 과정은 (그림 1)과 같다. 데이터 압축 과정에서 가장 먼저 수행해야 하는 작업은 볼륨 데이터를 적절한 크기의 단위 블록으로 나누는 것이다. 단위 블록의 최소 크기는 8 x 8 x 8 크기의 정육면체 복셀 그리드로, 필요에 따라 16 x 16 x 16, 또는 32 x 32 x 32 등의 크기로 확장하는 것이 가능하다. 본 논문에서는 [1]과 [2]에서 제안한 압축 방식을 확장함으로써 볼륨 데이터에서 많이 사용되는 부동 소수점 데이터에도 적용 가능한 방식에 대해 설명한다.

3.1 웨이블릿 변환

D4 기반 인코딩에서는 단위 블록 데이터를 변환하는데 Daubechies의 D4 웨이블릿을 사용하며, 이 과정은 (그림 2)에서 볼 수 있다. Daubechies D4 웨이블릿 변환에 대한 변환식은 다음과 같다.

$$lowpass = \frac{1+\sqrt{3}}{4\sqrt{2}}s_0 + \frac{3+\sqrt{3}}{4\sqrt{2}}s_1 + \frac{3-\sqrt{3}}{4\sqrt{2}}s_2 + \frac{1-\sqrt{3}}{4\sqrt{2}}s_3$$

$$highpass = \frac{1-\sqrt{3}}{4\sqrt{2}}s_0 + \frac{-3+\sqrt{3}}{4\sqrt{2}}s_1 + \frac{3+\sqrt{3}}{4\sqrt{2}}s_2 + \frac{-1-\sqrt{3}}{4\sqrt{2}}s_3$$

또, Daubechies D4 웨이블릿으로 변환된 계수를 본래 상태로 되돌리는 역변환식은 다음과 같다.

$$s_0 = \frac{3-\sqrt{3}}{4\sqrt{2}}a_0 + \frac{3+\sqrt{3}}{4\sqrt{2}}c_0 + \frac{1+\sqrt{3}}{4\sqrt{2}}a_1 + \frac{1-\sqrt{3}}{4\sqrt{2}}c_1$$

$$s_1 = \frac{1-\sqrt{3}}{4\sqrt{2}}a_0 + \frac{-1-\sqrt{3}}{4\sqrt{2}}c_0 + \frac{3+\sqrt{3}}{4\sqrt{2}}a_1 + \frac{-3+\sqrt{3}}{4\sqrt{2}}c_1$$

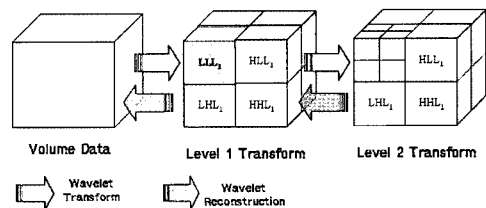
3차원 웨이블릿 변환은 x, y, 그리고 z 방향으로 한번씩 D4 필터를 적용해서 계산하는 과정으로 이뤄지며, 변환된 데이터 중에서 로우패스(lowpass) 필터를 통과하여 생성된 계수들에 대해 웨이블릿을 다시 적용하여 다음 단계의 계수

들을 반복적으로 얻는다. 웨이블릿 변환을 반복적으로 수행하는 이유는 데이터 복원에 영향을 미치지 않는 계수의 비율을 높임으로써 압축 효율을 높이기 위함이다.

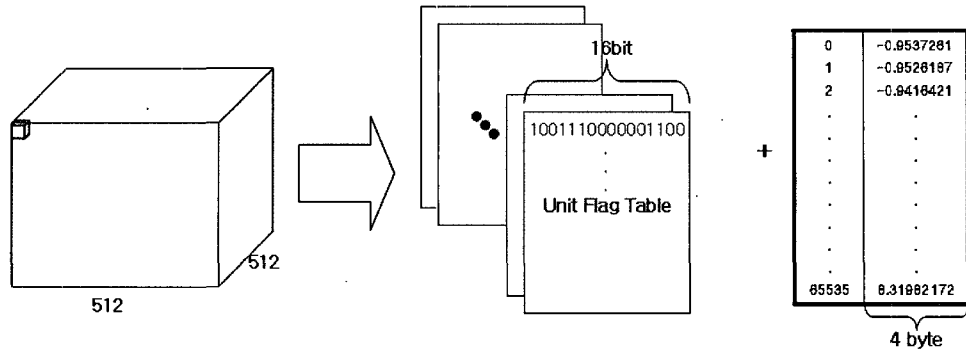
절삭 과정은 변환된 데이터에서 복원에 영향을 거의 미치지 않는 데이터를 잘라내는 과정이다. 이 때, 데이터가 복원에 미치는 영향을 판단하는 기준인 임계값(threshold)을 결정해야 한다. 이 임계값보다 작은 웨이블릿 계수는 복원에 미치는 영향이 매우 작다고 가정되며 이후 인코딩 과정에 포함되지 않는다. 따라서 임계값은 데이터 압축률에 직접적인 영향을 미치며, 임계값이 클수록 원본 데이터와 복원된 데이터의 오차는 커진다. 임계값을 결정하고 데이터를 절삭하는 과정은 다음과 같다.

- (1) 웨이블릿 계수를 크기 순서대로 정렬한다.
- (2) 전체 계수 중에서 사용할 계수의 개수를 결정한다. 일반적으로는 복원에 사용할 계수의 비율로 계수의 개수를 결정한다.
- (3) 크기 순서로 정렬한 웨이블릿 계수 중에서 사용할 계수 비율에 따라 임계값을 결정한다.
- (4) 변환된 웨이블릿 계수 중에서 임계값보다 절대값이 작은 계수를 모두 0으로 전환한다.

절삭 과정을 통과한 웨이블릿 계수는 양자화(quantization) 과정을 거치게 되며, 이 과정을 통해 데이터가 차지하는 저장 공간의 크기는 더욱 줄어든다.



(그림 2) 단위 블록에 대한 2단계 웨이블릿 변환



(그림 3) 블록 인코딩 데이터 헤더

3.2 양자화

본 논문에서 제안하는 D4 기반 인코딩 기법은 부동 소수점 데이터를 대상으로 하며, 부동 소수점(single precision floating point) 데이터는 데이터 저장에 4바이트가 사용된다. 이런 4바이트 데이터를 이보다 작은 데이터 공간을 차지하는 데이터로 변환할 수 있다면 압축률은 훨씬 더 높아질 것이다. 여기에 사용되는 기법이 양자화이다. 양자화는 데이터를 일정한 단계로 나눠서 분류하기 때문에 어떤 종류의 데이터 유형이라도 주어진 데이터 공간에 맞게 저장할 수 있다.

본 인코딩 기법에서는 일양 양자화(uniform quantization)를 사용해서 4바이트 크기의 데이터를 2바이트 인덱스로 저장한다. 이 때, 인덱스로 변환되는 데이터는 복원에 영향을 미치는 데이터만을 대상으로 한다. 실제 양자화 작업은 3.3에서 설명한 데이터 인코딩 과정 중 UBT와 BSDT를 저장하는 과정에서 수행되며, UBT에 복원에 영향을 미치는 데이터의 위치를 표기한 후, 그 값에 대해 양자화를 적용하여 인덱스로 변환, 저장한다.

양자화 테이블(quantization table)은 압축 데이터의 헤더 부분에 저장되며, 2바이트 크기의 인덱스와 4바이트 크기의 실제 부동 소수점 데이터로 구성된다. 일양 양자화에서 테이블 작성과정에서 데이터 값들 가운데 가장 큰 값에서 가장 작은 값을 빼고, 그 사이의 간격을 원하는 데이터 단계의 개수로 나눈 뒤, 각 단계 범위에 속한 값 중 대푯값을 선정한다. 일반적으로는 각 범위에 대한 중간값이 대푯값으로 사용된다.

데이터를 양자화 인덱스로 변환할 때는 데이터 값이 포함되는 범위를 대표하는 인덱스를 사용하면 된다. 이 과정을 거쳐 4바이트 크기의 부동 소수점 데이터는 2바이트 크기(65536 단계)의 정수 인덱스로 변환된다.

3.3 데이터 인코딩

데이터 인코딩 과정은 최종적으로 압축 데이터를 부호화하는 과정으로, 사용자는 인코딩된 데이터를 사용함으로써 데이터 처리에 필요한 메모리를 절약할 수 있다. 인코딩 과정에서는 데이터 복원에 영향을 미치는 데이터의 '값'과 '위치'를 함께 저장해야 한다. 본 논문의 D4 기반 인코딩 방식

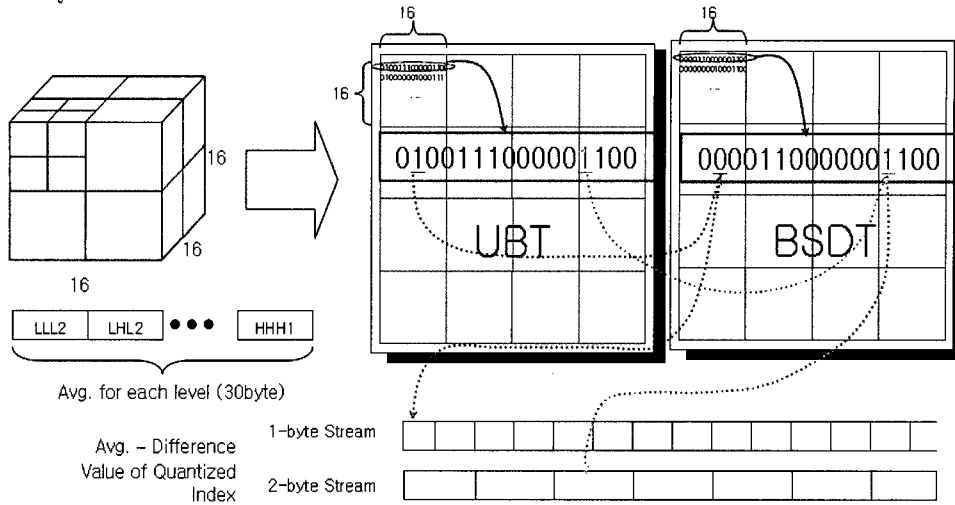
에서는 유닛 플래그 테이블(Unit Flag Table)을 사용해서 블록 데이터에서 유효한 블록, 즉 유닛의 위치를 저장한다. 인코딩 데이터의 헤더 부분에는 유닛 플래그 테이블과 양자화 테이블을 함께 저장한다(그림 3).

유닛 플래그 테이블은 특정 위치에 있는 유닛이 의미 있는(즉, 모두 0이 아닌) 데이터를 포함하고 있는지, 그렇지 않은지를 한 비트로 표현한다. 다시 말해, 이 테이블에서 0으로 설정된 비트에 위치한 유닛 내의 웨이블릿 계수는 모두 0으로, 이 유닛에서는 복원을 위한 계산 작업을 따로 수행할 필요가 없다. 빈 공간이 많은 블록 데이터나 웨이블릿 변환의 결과로 생성된 계수 중 복원에 영향을 미치지 않는 계수가 많이 존재하는 경우에는 유닛 플래그 테이블의 사용만으로도 데이터 압축률을 획기적으로 높일 수 있다.

데이터 복원에 직접적으로 영향을 미치는 유닛, 즉 유닛 플래그 테이블에서 1로 표기된 유닛은 UBT(Unit Bit Table)와 BSDT(Byte Stream Designation Table)를 사용해서 복원에 영향을 미치는 계수들의 위치를 저장한다(그림 4). UBT는 유닛 내에서 복원에 영향을 미치는 계수의 위치를 표기한다. UBT에서 표기해야 할 위치에 해당하는 데이터 값이 0인 경우에는 해당 비트에 0을 표기하고 그렇지 않으면 1을 표기한다. 이 때, 0이 아닌 데이터 값은 앞에서 설명한 바와 같이 양자화 과정을 통해 인덱스 값으로 변환되며, 이 과정에서 4바이트 데이터가 2바이트 인덱스로 변환되기 때문에 압축률은 약 2배 정도 증가하게 된다.

인덱스 값을 저장하는 과정은 다음과 같다.

- (1) 유닛 하나를 웨이블릿 변환의 각 레벨, 즉 LLL2, LLH2, LHL2, LHH2, HLL2, HLH2, HHL2, HHH2, LLH1, LHL1, LHH1, HLL1, HLH1, HHL1, HHH1의 15개 영역으로 나누고, 각 영역에 저장되어 있는 인덱스의 평균을 구한다.
- (2) 인덱스 위치에 따라 웨이블릿 변환 영역을 파악하고, 해당 영역의 인덱스 값의 평균과 인덱스의 차이를 계산한다.
- (3) 인덱스의 차이 값의 크기에 따라 1바이트짜리 데이터 영역에 저장할 것인지, 2바이트짜리 데이터 영역에 저장할 것인지를 결정한 뒤, BSDT에 해당 스트림을 지정해서 저장한다.



(그림 4) 데이터 복원에 영향을 미치는 유닛의 인코딩

인덱스와 인덱스 영역 평균의 차이가 [-128, 128)의 영역에 속하는 경우에는 1바이트 크기의 스트림에 저장되고, 그 영역을 벗어나는 경우에는 2바이트 크기의 스트림에 저장된다. BSDT는 유닛 내에서 값이 0이 아닌 계수의 위치를 나타내는 비트로 BSDT의 비트 값이 0이면 해당 데이터 인덱스가 1바이트 스트림에 위치하고 비트 값이 1이면 데이터 인덱스가 2바이트 스트림에 위치하게 된다. 이렇게 1바이트 스트림과 2바이트 스트림을 분리해서 저장하면 크기가 큰 데이터 공간에 작은 데이터가 저장됨으로써 낭비되는 저장 공간을 최소화할 수 있다[6].

본 논문에서 제안한 D4 기반 데이터 인코딩 방식에서 인덱스 외의 부가정보 저장에 필요한 공간은 다음과 같이 계산할 수 있다.

$$\begin{aligned} \text{Sizeof (Header)} &= \lceil (\# \text{ of Units}) / 16 \rceil * 2 + \text{Quantization Level} * 6 \text{ (byte)} \\ \text{Sizeof (UBT + BSDT + Avg.Index for each wavelet level)} &= \\ &= (\lceil \text{Unit-Size} / 16 \rceil * 2 + \lceil \text{Unit-Size} / 16 \rceil * 2 + 30) * (\# \text{ of nonzero units}) \end{aligned}$$

이 부가정보는 최종적으로 데이터 복원에 영향을 미치는 계수 값에 대한 양자화 인덱스를 찾아가는데 필요하다. 수식에서 알 수 있듯이 부가정보 저장에 필요한 데이터 공간은 데이터 복원에 영향을 미치는 유닛의 개수에 비례한다.

3.4 디코딩

인코딩된 복셀 값을 본래 데이터로 복원하는 과정은 다음과 같다.

- (1) 복원하고자 하는 블록(유닛)의 위치를 유닛 플래그 테이블에서 확인한다. 해당 비트가 0이라면 더 이상의 복원 작업은 수행할 필요가 없고, 유닛에 존재하는 모든 데이터 값은 0이다. 그렇지 않은 경우

에는 다음 과정을 계속 실행한다.

- (2) 해당 유닛에 대한 UBT를 살펴서 비트 값이 0인 곳의 계수는 0으로 표기한다.
- (3) UBT의 비트 값이 1인 경우, 계수 복원을 위해 BSDT를 참조한다. BSDT의 해당 비트가 0이면 1바이트 스트림에서, 1이면 2바이트 스트림에서 데이터를 검색한다. 디코딩 과정은 유닛 단위로 발생하므로, UBT와 BSDT를 참조해서 순차적으로 바이트 스트림에서 값을 가져오면 된다. 예를 들어, (그림 4)에서 UBT의 2번째 비트에 표기된 데이터의 경우, 해당 BSDT 비트가 0이므로 1바이트 스트림에서 값을 가져오면 된다.
- (4) 값을 가져온 데이터의 위치에 대한 웨이블릿 영역을 파악한다. 웨이블릿 영역이 파악되면, 해당 영역의 인덱스 평균을 더해서 최종 양자화 인덱스 값을 구한다.
- (5) 양자화 테이블에서 인덱스를 검색, 대푯값을 찾아서 유닛의 해당 위치에 채워 넣는다.
- (6) 유닛 내의 모든 계수에 대해 (2)~(4)의 과정을 되풀이한다. 유닛 내의 모든 계수 값이 채워지면 역 웨이블릿 변환 필터를 적용, 본래 데이터를 복원한다.

데이터의 복원은 유닛 단위로 수행되며, 이렇게 복원된 값은 바로 대화식(interactive) 응용 프로그램에서 이용될 수 있다.

4. 실험결과 및 성능분석

<표 1>에서 부동 소수점 데이터 압축에 앞서 [0, 4096] 범위의 정수로 구성된 볼륨 데이터에 인코딩 방식을 적용하

<표 1> zerobit 인코딩 방식[7]과 D4 인코딩 방식의 비교

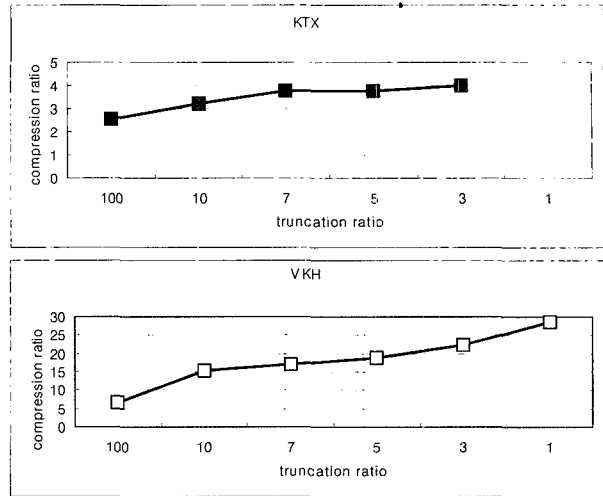
인코딩 방식		10%	7%	5%	3%
제로비트 인코딩 (Haar)	Data Size(MB)	63.84	49.13	38.42	26.46
	PSNR(dB)	53.84	50.41	47.79	44.49
	Comp-Ratio	11.27	14.65	18.74	27.21
D4 인코딩 (Daubechies D4)	Data Size(MB)	63	51	44	35
	PSNR(dB)	60.15	57.67	55.24	51.76
	Comp-Ratio	10	12.35	14.32	18

여 정수 데이터를 압축한 성능을 [8]의 결과와 비교하여 확인할 수 있다. 이와 같이 한정된 범위에 속한 정수 데이터의 경우에는 2바이트만으로 모든 데이터를 표현할 수 있으므로, 별도의 양자화 과정을 거치지 않고 UBT와 BSDT를 사용하여 블록 내의 유효 데이터의 위치와 값을 인코딩한다 [1][2]. 기존 기법과의 비교를 위해 실험에는 [8]에서 사용한 VH(Visible Human) 데이터를 사용했으며, 512 x 512 x 1248 해상도의 데이터(624MB)를 사용해서 실험을 수행했다.

<표 1>에서 볼 수 있듯이 동일한 데이터로 인코딩한 결과를 비교했을 때, 비슷한 크기로 압축된 데이터에 대해 D4를 이용한 인코딩 방식의 PSNR(Peak Signal-to-Noise Ratio)이 훨씬 높은 것을 볼 수 있으며, 따라서 같은 비율의 계수를 절삭했을 때 D4를 이용한 방식의 데이터 오류율이 훨씬 낮다는 것을 확인할 수 있다.

부동소수점 데이터 실험에는 VKH(Visible Korea Human)와 KTX(Korean Trainer eXperimental, 한국형 고등 훈련기) 데이터를 사용했다. VKH 데이터는 512 x 512 x 512의 해상도를 갖춘 인체 영상 데이터로써 실험을 위해 각각의 수치 데이터를 4바이트짜리 부동소수점으로 저장했다. KTX 데이터는 비행 훈련기의 주변에 나타나는 공기의 흐름을 표현하는 데이터로써, 실험을 위해 여러 개로 나뉘어 있는 데이터를 통합해서 256 x 256 x 256 해상도의 격자로 만들었다. KTX 데이터는 값의 범위가 [0.404195, 1.346161]인 4바이트짜리 부동소수점 데이터이며, 전체 데이터 크기는 64MB이다.

본 논문의 D4 기반 인코딩 방식으로 압축한 데이터의 압축률과 화질은 <표 2>에서 볼 수 있다. 인코딩 결과로 산출된 압축 데이터의 용량은 원본 데이터의 40% 이하로 줄어들었지만, RMSE(Root Mean Square Error)로 표시된 오차는 그다지 크지 않은 것을 볼 수 있다. 또, 원본 데이터와



(그림 5) 절삭 비율과 압축률

복원 데이터의 화질차를 수치화한 PSNR과 SNR(Signal-to-Noise Ratio)을 비교해 봐도 데이터 손상은 그다지 크지 않다. 즉 압축률에 비해 복원 화질은 큰 차이가 없는 것을 볼 수 있다. 압축에 사용된 양자화 단계는 65536 단계이며, 물론 양자화 단계를 낮추면 압축률을 높일 수 있다.

(그림 5)는 절삭 비율에 대한 압축률의 변화를 그래프로 보여준다. KTX 데이터는 VKH 데이터에 비해 블록 데이터의 복셀 값에 0이 나오는 비율이 낮다. 따라서 유닛 전체가 1비트로 표현되는 비율도 VKH 데이터에 비해 낮아질 수밖에 없으며, 이런 이유로 VKH 데이터에 비해 압축률이 떨어진다. 또한, 실제로 사용되는 계수 비율이 줄어드는데 따른 압축률의 향상도 그다지 크지 않은 것을 확인할 수 있다. 이는 실제로 저장될 계수의 개수가 줄어들어 따라 데이터 저장에 필요한 공간은 줄어들지만, 블록 정보 복원을 위해 기본적으로 필요한 부가 데이터의 크기는 그대로이기 때문에 나타나는 현상이다. 따라서 향후에는 이런 기본 정보 저장에 필요한 공간에 대한 개선이 요구된다.

VKH의 경우 512 x 512 x 512 해상도 블록 데이터(복셀 134,217,728개) 전체를 디코딩하는데 걸리는 시간은 14~20초 정도로 디코딩 시간은 절삭 비율에 반비례한다. 즉, 10% 계수만 압축한 데이터의 디코딩 시간이 가장 크고, 1% 계수만으로 압축한 데이터의 디코딩 시간이 가장 작다.

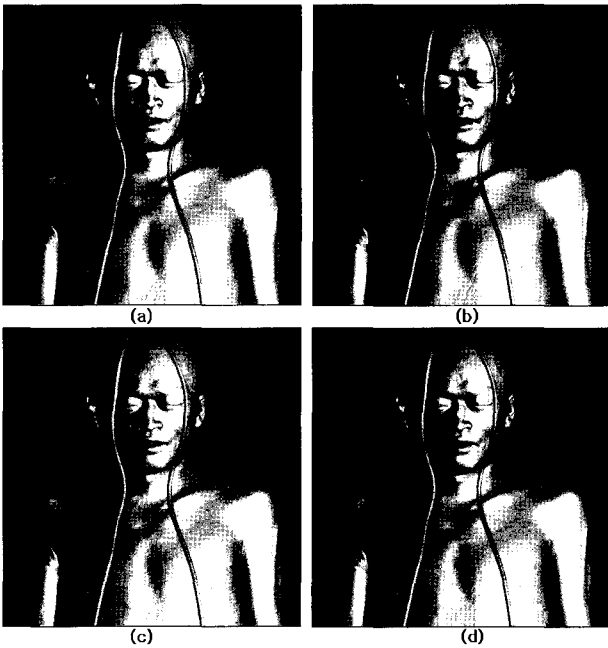
정수 데이터 인코딩 기법과의 비교를 위해 <표 3>에서는 VH 데이터를 4바이트 부동 소수점 데이터로 저장한 뒤, D4

<표 2> D4 기반 인코딩 방식의 부동 소수점 데이터 압축 결과

Data	KTX: 256 x 512 x 512 (64MB) Max = 1.346161, Min = 0.404195						VKH: 512 x 512 x 512 (512MB) Max: 4096.0, Min: = 0.0					
	100%	10%	7%	5%	3%	1%	100%	10%	7%	5%	3%	1%
Truncation Ratio(%)	100%	10%	7%	5%	3%	1%	100%	10%	7%	5%	3%	1%
Data Size(MB)	25M	20M	17M	17M	16M	-	77M	33M	30M	27M	23M	18M
RMSE	0.0021	0.0022	0.0023	0.0024	0.0027	-	2.70	6.29	8.17	10.37	14.17	32.14
SNR (dB)	53.26	53.22	52.62	52.47	51.36	-	44.29	36.95	34.67	32.61	29.89	22.78
PSNR (dB)	55.94	55.91	55.40	55.26	54.14	-	63.62	56.98	54.58	52.51	49.77	42.68
Compression Ratio	2.56	3.2	3.76	3.76	4	-	6.65	15.52	17.07	18.96	22.26	28.44

〈표 3〉 VH 데이터의 부동 소수점 압축 결과

Data	VH: 512 x 512 x 1248 (1,248MB) Max: 4096.0, Min: = 0.0				
	100%	7%	5%	3%	1%
Truncation Ratio(%)	100%	7%	5%	3%	1%
Data Size(MB)	81M	69M	64M	57M	41M
RMSE	4.19	5.94	7.24	11.44	32.29
SNR (dB)	42.39	39.37	37.65	33.68	24.66
PSNR (dB)	62.34	59.03	57.26	52.90	43.53
Comp- Ratio	15.40	18.09	19.5	21.89	30.44



(그림 6) 렌더링된 결과 이미지 (a) 10%, (b) 5%, (c) 3%, (d) 1% 계수 사용

인코딩 기법을 적용한 결과를 정리하였다. 인코딩에서 16 x 16 x 16 크기의 블록을 사용하였으며, 부동 소수점 데이터의 압축률에 비해 화질의 저하는 거의 없는 것을 확인할 수 있다. VH 데이터도 VKH 데이터와 같이 데이터 내에 빈 공간이 상대적으로 많이 나타나기 때문에 KTX 데이터에 비해 높은 압축률을 나타낸다.

(그림 6)에서 각각 10%, 5%, 3%, 1%의 웨이블릿 계수로 복원한 데이터를 볼륨 광선추적법으로 렌더링한 결과영상을 비교할 수 있다. 이 영상들은 볼륨 데이터를 인코딩된 상태로 메모리에 로드한 후, 무작위 추출에 의해 현재 연산을 위해 필요한 블록만을 선택적으로 디코딩하면서 볼륨 렌더링을 수행하여 얻은 것이다. 결과 영상들의 비교를 통해 확인할 수 있듯이 압축률에 대한 이미지 화질의 저하는 그리 크지 않다.

5. 결론

본 논문에서는 가시화를 위한 3차원 볼륨 데이터 압축 기

법인 Daubechies D4 기반 인코딩 기법을 소개했다. 이 압축 기법은 4바이트 부동소수점으로 구성된 볼륨 데이터를 대상으로 하며, 볼륨 데이터를 단위 블록으로 나눈 뒤, Daubechies의 D4 웨이블릿을 사용하여 데이터를 변환한다. 변환된 계수들 중에서도 중요도가 떨어지는 계수들을 제거한 후 인코딩 과정을 수행하는 데 이 때 데이터에 손실이 발생하게 된다. 인코딩은 블록 단위의 인코딩을 기본으로 하며, 이렇게 구현된 압축 기법은 사용된 필터의 특성과 인코딩 기법에 의해 높은 압축률에 비해 화질의 손상이 적다는 장점을 가진다. 따라서 이 기법은 정밀한 영상 복원을 필요로 하는 응용이나 대용량 볼륨 데이터 압축이 필수적인 응용 소프트웨어 개발에 유용하게 활용될 수 있다. 또한, 블록단위로 압축 및 복원 작업을 수행할 수 있기 때문에 필요한 데이터를 선별적으로 활용할 수 있으며, 이 특성으로 인해 본 압축 기법은 볼륨 렌더링에의 적용이 용이하다.

향후에는 실시간 렌더링 애플리케이션에 적용하는 방향으로 인코딩 방식을 좀 더 발전시키고 부가적인 데이터의 인코딩을 최소화함으로써 압축률을 높이는 방안을 고려할 뿐만 아니라, 양자화 방식을 개선하여 압축률을 높이고 화질 저하를 줄이는 방안을 모색할 예정이다. 또, 현재의 인코딩 스킴을 확장해서 각 시간에 따른 변화를 볼륨 데이터로 표현하는 4차원 데이터인 시간가변(time-varying) 볼륨 데이터에 적용할 계획이다.

참고 문헌

- [1] 허 영주, 박 상훈, "Daubechies 웨이블릿 변환을 이용한 볼륨 데이터 압축", 제 24회 한국정보처리학회 추계 학술발표대회, 제 12권 2호, 2005년 11월.
- [2] 허 영주, 박 상훈, "Daubechies D4 웨이블릿 필터를 이용한 유닛 기반 볼륨 데이터 압축 기법", HCI 학술발표대회, 2006년 2월.
- [3] C. Bajaj, I. Ihm, S. Park, "3D RGB compression for interactive applications," ACM Transactions on Graphics, Vol.20, pp.10-38, 2001.
- [4] I. Daubechies, D4 Wavelet Transform, http://www.bearcave.com/misl/misl_tech/wavelets/daubechies/.
- [5] I. Daubechies, Ten Lectures on Wavelets, SIAM, 1992.
- [6] S. Guthe, W. Straber, "Real-time decompression and visualization of animated volume data," Conference on Visualization, pp.349-256, October, 2001.
- [7] T. He, "Wavelet-assisted volume ray casting," Pacific Symposium on Biocomputing, Vol.3, pp.153-164, 1998.
- [8] I. Ihm, S. Park, "Wavelet-based 3D compression scheme of interactive visualization of very large volume data," Computer Graphics Forum, Vol.18, 1999.
- [9] M. Marcellin, M. Gormish, A. Bilgin, M. Boliek, "An Overview of JPEG-2000," Proceedings of Data Compression

Conference 2000, pp.523-544, Mar., 2000.

[10] S. Muraki, "Approximation and rendering of volume data using wavelet transform," Proceedings of Visualization '92, pp.21-28, October, 1992.

[11] S. Muraki, "Volume data and wavelet transforms," IEEE Computer Graphics and Applications, pp.50-56, 1996.

[12] A.Norton, A.Rookwood, "Enabling view-dependent progressive volume visualization on the grid," IEEE Computer Graphics and Applications, pp.22-31, March/April, 2003.

[13] S. Park, "Compression-based visualization of large volume data," Proceedings of Korea Supercomputing Workshop, April, 2005.

[14] F. Rodler, "Wavelet based 3D compression with fast random access for very large volume data," Proceedings of Pacific Graphics'99, pp.108-117, IEEE Press, 1999.

[15] J. Shapiro, "Embedded image coding using zerotrees of wavelet coefficients," IEEE Transactions on Signal Processing, Vol.41, pp.3445-3463, Dec., 1993.

[16] E. Stollnitz, T. DeRose, D. Salesin, Wavelets for Computer Graphics: Theory and Applications, Morgan Kaufmann Publishers, 1996.

[17] D. Taubman, "High Performances Scalable Image Compression with EBCOT," IEEE Transactions on Image Processing, Vol.9, No.7, July, 2000.

[18] C. Valens, EZW encoding, <http://perso.wanadoo.fr/polyvalens/clemens/ezw/ezw.html>.



허 영 주

e-mail : popea@kisti.re.kr
 1997년 2월 성균관대학교 정보공학과 (공학사)
 2002년 8월 성균관대학교 컴퓨터공학과 (공학석사)
 2002년 9월~현재 한국과학기술정보연구원(KISTI) 연구원

관심분야 : 컴퓨터 그래픽스, 과학적 가시화, 데이터 압축 등



박 상 훈

e-mail : mshpark@dongguk.edu
 1993년 서강대학교 수학과(이학사)
 1995년 서강대학교 컴퓨터학과(공학석사)
 2000년 서강대학교 컴퓨터학과(공학박사)
 2000년~2000년 서강대학교 컴퓨터학과 박사후연구원

2000년~2002년 University of Texas at Austin의 TICAM (Texas Institute of Computational and Applied Mathematics) 박사후연구원

2002년~2005년 대구가톨릭대학교 컴퓨터정보통신공학부 조교수

2005년~현재 동국대학교 영상대학원 멀티미디어학과 조교수
 관심분야 : 컴퓨터 그래픽스, 과학적 가시화, 고성능 컴퓨팅, 모바일 그래픽스 등