극소원자블록 특성을 이용한 고속 프랙탈 영상압축

위 영 철
정보통신대학
아주대학교
ycwee@ajou.ac.kr

A High-Speed Fractal Coding Using the Characteristics of Ultra-Smal l Atomic Block

Young Cheul Wee
College of Information Technology
Ajou University

요 약

본 논문에서는 프랙탈 영상압축에서 화질/압축률을 향상 시키면서 압축시간을 획기적으로 향상시키는 방법을 제안한다. 본 방법은 레인지블록의 크기가 아주 작으면 유사변환 계수들의 값을 극히 제한하더라도 유사한 도메인블록을 아주 가까운 주변에서 쉽게 찾을 수 있음을 활용하여 압축시간을 단축시키고 화질/압축률을 향상 시킨다. 또한, 본 방법은 인접한 레인지블록 들의 압축에 사용되는 계수들이 좋은 상호관계를 가지도록 유도하여 화질/압축률을 더욱 향상 시킨다. 본 방법은 대부분의 프랙탈 영상압축 방법에 쉽게 적용되어 그 성능을 향상시킬 수 있다. 특히 압축시간이 전역탐색 방법에 비해서 512×512 영상에서 약 60000 배 256×256 영상에서 약 15000 배 빠르며 실시간 동영상의 I-frame에도 사용가능 하다.

1. 서론

프랙탈(fractal) 영상압축의 가장 큰 문제점은 압축시간이다. 본 논문에서는 레인지블록(range block)의 크기가 아주 작으면 유사변환 계수들의 값을 극히 제한하더라도 유사한 도메인블록(domain block)을 아주 가까운 주변에서 쉽게 찾을 수 있다. 본 논문에서는 이를 활용하여 압축시간을 획기적으로 단축시키는 방법을 제안한다. 또한, 본 방법은 인접한 레인지블록 들의 압축에 사용되는 계수들이 좋은 상호관계를 가지도록 유도하여 화질/압축률을 더욱 향상 시킨다.

프랙탈 영상압축은 영상을 서로 겹치지 않는 레인지블록들로 분할한 다음 레인지블록과 같은 크기로 축소한 도메인블록을 유사변환(affine transformation)으로 밝기(grey value)를 수정하여 레인지블록의 근사치를 구한다. 프랙탈 코드(code)는 각 레인지블록의 분할에 사용되는 정보, 도메인블록의 위치, 유사변환의 계수들로 구성된다.

프랙탈 영상압축 방법에서의 화질, 압축시간, 압축률은 각각 유사블록 오차, 유사블록 탐색시간, 영상분할 방법과 서로 밀접한 관계를 가지고 있다. 한 레인지블록 R과 변환된 도메인블록 $\hat{R} = s \cdot D + g$, s: 대비(contrast), g: 밝기(brightness)의 오차 $d(R,\hat{R})$는 식(1)의 평균오차(Mean Square Error)로 측정한다.

$$d(R,\hat{R}) = (1/n)\sum_{i=1}^{n}(r_i - \hat{r}_i)^2 \qquad (1)$$

화질은 유사블록과의 오차에 비례하는데 화질을 높이기 위해서는 일반적으로 1) 탐색범위 확대 2) 유사변환 확장 3) 레인지블록 크기 축소를 하는데 이에 따라 압축시간이 길어지고 압축률이 저하되게 된다. 탐색범위를 크게 하면 유사블록과의 오차가 줄어서 화질은 높아지나 유사블록 탐색시간이 늘어나게 되고 도메인블록의 위치를 저장하는데 더 많은 비트를 할당하게 되므로 압축률이 저하된다. 도메인블록의 위치에는 탐색범위가 $k \times l$이면 $O(\log_2 kl)$의 비트(bit)를 할당된다. 유사변환에 사용되는 계수(parameter)의 범위를 크게 하거나 다양한 변환을 추가하면 탐색시간이 늘어나고 압축률이 저하된다. 레인지블록의 크기를 적게 하면 화질은 높아지나 압축률이 저하되게 된다.

레인지블록의 픽셀(pixel) 수를 $n(R)$ 이라하고 압축코드에서 할당된 비트수를 $bit(R)$이라 하자. 한 레인지블록 R의 압축률은 $8 \cdot n(R)/bit(R)$ 이므로 레인지블록의 크기가 작아지면 압축률이 저하된다. 그런데 $n(R)$이 아주 작으면 탐색영역 및 유사변환을 극히 제한하여도 오차가 적은 유사블록 변환의 가능성이 높아지므로 $bit(R)$이 감소한다. 예를 들면, 512×512 크기의 영상에서 레인지블록의 크기가 4×4인 경우 고화질을 위해서는 보통 $bit(R) = 32 \pm \alpha$ (k,l: 16~18 bit, s : 3~5 bits, g: 7~8 bits, rotation and flip: 3bits)[2]이 되어 약 4배의 압축이 된다. 레인지블록의 크기가 2×2인 경우 4×4와 같이 비트를 할당하면 압축이 거의 되지 않는다. 그러나 $bit(R)$을 레인지블록의 크기가 4×4인 경우보다 1/4 이하로 사용하고 유사변환 오차를 줄일 수 있으면 더 높은 화질/압축률을 얻을 수 있게 된다. 특히 탐색 점의 수가 현격히 적으므로 압축시간이 현격히 줄어든다.

전역탐색(full search)과 같은 화질을 얻는 알고리즘들은[6][7][8] 탐색 점의 수를 최대 수십 배 줄이는데 그치고 있고 화질을 일부 저하시키고 압축시간을 줄이는 알고리즘[3][4][5]들은 고화질에서 압축시간이 현격히 늘어나고 압축률이 떨어지는 문제점을 가지고 있다. 본 논문에서는 레인지블록의 크기를 아주 작게 하고 탐색범위와 유사변환에 필요한 계수들의 범위를 극히 제한하여 화질/압축률을 향상시키고 압축시간을 현격히 단축시키는 방법을 소개한다. 또한, 인접한 레인지블록 간의 압축에 사용되는 계수들이 서로 좋은 상호관계(correlation)를 가지도록 유도하여 화질/압축률을 더욱 향상시키는 방법을 소개한다. 실험결과를 보면, 본 방법은 압축속도에서 전역탐색 방법에 비하여 약 15000 배 이상의 속도 향상을 보이고, CIF(352x288) 크기의 영상 압축시간이 0.03초 이하가 되어서 30 fps의 실시간 동영상 압축에서도 프랙탈 영상압축 방법을 사용 할 수 있게 된다. 화질은 최소 레인지블록의 크기를 4×4 한 다른 방법에 비하여 같은 압축률에서 평균 0.26 dB가 향상된

다.

본 논문은 다음과 같이 구성된다. 2 절에서는 극소원자블록의 특성과 코딩방법을 설명한다. 3절에서는 원자블록의 계수예측과 병합방법에 대해서 논의한다. 4절에서는 PC(Intel Pentium IV 2.0G Hz, 512M DRAM)에서의 실험결과를 기존의 방법들과 비교분석 한다. 5절에는 결론이 주어진다.

2. 극소원자블록의 특성 및 코딩

크기가 $b \times b$인 레인지블록이 한계오차 δ 이내에서 한 도메인 블록과 유사 할 확률은 $(\delta/256)^{b^2}$ 에 비례하므로 레인지블록의 크기가 작을수록 유사블록과의 오차가 급속히 줄어든다. 그러나 레인지블록의 크기가 작을수록 압축률이 저하되므로 일반적인 프랙탈 압축방법은 압축률을 높이기 위해서 블록의 크기를 가능한 크게 하고 많은 탐색 점을 두거나 복잡한 변환을 사용하여 유사변환 오차를 줄인다. 이 경우 압축시간이 크게 늘어나고 화질이 저하되는 문제점을 가진다.

레인지블록의 크기가 극히 작으면 지역성이 매우 강하게 되며 극히 단순한 변환을 하여도 유사블록의 오차를 작게 유지 할 수 있게 된다. 여기서의 문제점은 압축률이 저하되는 것이다. 그러나 극소원자블록의 강한 지역성을 잘 활용하면 할당되는 비트를 크게 줄여서 화질/압축률을 오히려 향상시킬 수 있다. 원자블록의 크기가 2×2일 때의 코딩 방법을 생각하여보자. 원자블록의 크기가 4×4인 경우보다 유사하거나 향상된 압축률을 유지하기 위해서는 4×4 원자블록 코딩보다 블록 당 1/4 이하의 비트를 사용해야 한다.

본 방법은 2×2원자블록의 강한 지역성을 잘 활용하여 4×4 원자블록 코딩보다 적은 bpp(bit per pixel)로 더 높은 화질/압축률을 가지도록 한다. 코딩 방법의 핵심은 유사변환에서 오차가 적을 확률이 매우 높은 탐색 점 및 변환계수들만 사용하는 것이다: 4개의 인접한 탐색 점에 대하여 두개의 고정된 s 값과 g값은 4 비트로 양자화여 유사변환을 함으로써 4개의 2×2블록이 하나의 4×4 원자블록 전역탐색 보다 비트를 적게 사용하게 한다. 압축률을 더욱 높이기 위해서는 3절에서 소개되는 변환계수 예측에 의한 블록병합 방법을 사용한다.

강한 지역성을 활용하기 위하여 탐색 점은 가능한 도메인블록과 근접한 위치를 선택한다. 도메인블록의 크기는 4×4로 하고 4개의 탐색 점은 레인지 블록을 도메인블록의 내부에 포함하는 $(0,0)$, $(-2,0)$, $(0,-2)$, $(-2,-2)$으로 구성한다. $(-1,-1)$은 도메인블록의 중앙에 레인지블록이 위치한 경우인데 다른 탐색 점에 비하여 평균 유사변환 오차는 적으나 복원(decode)시 수렴에 문제가 있으므로 탐색 점에서 제외한다. 실험결과 대부분의 영상에서 원자블록의 크기가 2×2 이면 탐색 점의 수를 4개로 제한하여도 원자블록의 크기가 4×4 이고 전역탐색을 할 때 보다 더 나은 PSNR(peak signal to noise ratio)을 얻을 수 있었다. 크기가 $m \times n$인 영상에서 탐색시간은 레인지블록 당 탐색 점의 수와 비례하게 되는데 전역탐색에 비해서 약 $4/mn$이 된다.

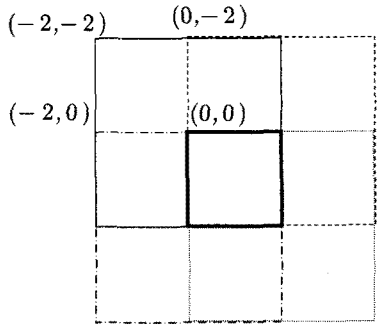


그림 1: 4개의 도메인블록 위치.

유사변환의 대비 값 s 는 밝기 g의 상호관계, 복원의 수렴성(convergence)과 연산속도를 고려하여 0.75와 0.5의 두 상수를 사용 한다. 0.75 또는 0.5와의 곱셈은 shift 연산으로 계산이 된다. 여기서 s 에 할당되는 bpp는 1/4 이므로 원자블록의 크기를 4×4로 할 때 (보통 s 값을 4~32개로 하고 2~5 bit을 할당)와 거의 같게 된다. 대부분의 영상에서 $s = 0.5$는 에지(edge) 부근에 적

용되고 $s=0.75$ 는 영상이 비교적 복잡하지 않은 부분에 적용된다. 또한 s 값은 주변블록과 좋은 상호관계(correlation)를 가지므로 무 손실(entropy) 압축을 할 경우 약 2배의 압축이 되어서 bpp가 약 1/8이 된다. 또한, 압축시간은 대비 값의 수에도 비례하므로 압축시간이 2배 이상 단축된다.

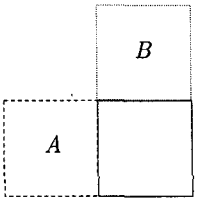
밝기 값 g 는 uniform quantization을 적용하여 4~5 bit을 할당 한다. 밝기 값이 가장 많은 비트를 사용하게 되는데 주변블록간의 상호관계를 잘 활용하면 비트수를 대폭 줄일 수 있다.

본 방법은 하나의 4×4 레인지블록 코딩을 4개의 2×2 서브블록으로 분할하여 각 서브블록이 독립적으로 극히 제한된 탐색범위와 유사변환으로 코딩 하는 것으로도 볼 수 있다. 한 레인지블록 당 할당하는 총 비트 수는 6~8로 평균 7/4 bpp가 되므로 원자블록의 크기를 4×4로 할 때 (평균 2 bpp) 보다 더 높은 압축률을 가진다. 화질은 평균 0.3 dB가 향상된다. 유사블록 탐색시간은 전역탐색 시간의 약 $mn/4$ 이 되므로 512×512 영상에서 60000배 이상 256×256 영상에서 15000배 이상의 속도 향상이 된다.

3. 휘도예측 블록병합에 의한 프렉탈 코딩 알고리즘

화질/압축률을 더욱 높이기 위해서 주변블록 간의 상호관계를 활용하여 극소원자블록들을 병합하는 방법을 소개한다. 일반적인 원자블록 병합방법들[3][4][5]과 같이 병합된 블록들이 같은 위치벡터 및 변환계수들을 공유한다. 원자블록 병합방법들은 고화질을 유지하기 위해서 탐색영역을 확장하고 변환계수들의 범위를 늘리게 되므로 압축시간이 늘어나고 극소원자블록(2×2)의 특성이 없어진다. 따라서 본 논문에서 소개하는 휘도예측 블록병합 프렉탈 코딩방법 Quick Fractal은 참조블록을 2개로 제한하고 매우 간단한 블록병합 방법을 사용한다.

병합 후에 각 원자블록의 유사성은 유사변환에 사용되는 밝기 g 값에 영향을 가장 많이 받으므로 각 원자블록이 독립적으로 g 값을 가지게 하면 탐색영역을 확장하지 않아도 유사블록과의 오차가 크게 증가하지 않게 할 수 있다. 또한 인접한 블록들은 g 값이 유사한 경우가 많으므로 각 원자블록이 참조(reference)하는 인접한 블록과의 밝기차이 값을 각각 가지도록 한다. 압축시간을 증가 시키지 않기 위해서 인접한 블록을 참조 할 때 각 레인지블록을 코딩하는 순서를 고려하여 왼쪽 블록 또는 위쪽 블록을 참조한다. 두개의 참조블록 중 도메인블록의 위치 및 대비 값을 같게 두고 g + 유사변환 오차의 cost가 적은 블록을 선택한다(그림 2 참조). 여기서 더 많은 인접한 블록들을 참조하더라도 압축률이 크게 향상되지 않고 압축시간이 늘어나므로 다음과 같이 두개의 블록만을 참조 하게한다.



$T=$ A 와 B 각각의 위치벡터와 대비 값으로 유사 변환한 오차
T + 밝기 값의 차이 cost가 적은 블록을 선택

그림 2: 참조블록과 정보공유 방법.

Quick Fractal

1. 영상을 2×2 크기의 레인지블록 R들로 분할한다.

2.1. 각 레인지블록에 대해서 왼쪽 블록과 같은 위치벡터 $(\triangle x, \triangle y)$와 s 값으로 도메인블록 D를 변환한 오차 $d(R,\hat{R})$가 한계오차(Tolerance) 보다 작고 g의 차이 값이 $\pm\alpha$ 이내 이면 inherit flag를 1로 하고 g 의 차이 값 α를 저장한다.

2.2. $d(R,\hat{R})$가 한계오차 보다 크면 두개의 s 값 0.75와 0.5에서 가장 유사한 도메인블록을 찾은 다음 inherit flag를 0으로 하고 도메인블록의 위치 벡터, 대비 값, 밝기 값을 저장한다.

3. non-inherit 블록 중에서 inherit로 하였을 때 보다 사용되는 비트의 cost가 줄어든 오차의 cost보다 크면 inherit 블록으로 전환한다.

End Quick Fractal

한 영상의 inherit 블록의 수를 p, non-inherit 블록의 수를 q라 하자. Inherit 블록은 약 2 비트를 사용하고 non-adoptive 블록은 약 9 비트를 사용하므로 bpp는 $(2p+9q)/4$가 된다. 따라서 Quick Fractal의 압축률은 inherit 블록의 수에 좌우된다. Inherit 블록은 주로 영상이 복잡하지 않은 부분에 위치하고 Non-inherit 블록은 에지가 있는 부분에 위치한다(그림 3 참조). 실험결과를 보면 아주 복잡한 영상을 제외하고는 adoptive 블록이 90% 이상이 되었다.

그림 3: Lenna 영상의 Inherit 블록 분포.

4. 실험결과

512×512 크기의 표준영상 Lenna, Baboon, Boat, Pepper에 대해서 수행된 본 연구의 실험결과는 다음과 같다. 먼저 2×2 원자블록 코딩을 4×4 원자블록 전역탐색과 비교하여 보면 표 1과 같이 평균 압축시간은 약 60,000 배, PSNR은 약 0.26 dB, 압축률은 약 10% 향상된다. 기존의 프랙탈 압축방법에서 4×4블록으로 코딩하는 부분을 4개의 2×2블록 코딩으로 대체하면 실험결과와 같은 성능 향상이 된다. 고화질에서는 원자블록인 4×4블록 코딩이 많아지므로 고화질 일수록 더 많은 성능향상이 된다.

	4×4 Full Search			2×2 코딩		
영상	시간 (sec)	PSNR	CR	시간 (sec)	PSNR	CR
Lenna	2805	38.337	4.00	0.041	39.100	4.47
Baboon	2727	27.284	4.00	0.063	27.639	4.29
Boat	2775	34.251	4.00	0.060	34.329	4.49
Pepper	2797	35.207	4.00	0.044	35.327	4.33

표 1: 2×2 코딩과 4×4 Full Search의 성능 비교.

Quick Fractal를 Saupe[2]의 Horizontal-Vertical 분할방법과 Tong와 Wong[1]의 Adoptive Approximate Nearest Search 방법과 비교하여 보면 비슷한 압축률에서 표 2와 같이 Adaptive Approximate Nearest Search 방법 보다 평균 압축시간이 약 700 배, PSNR은 약 0.71dB 향상되었다. Quick Fractal II의 압축률은 블록병합 방법에 따라 더욱 향상 될 수 있으나 동영상에서도 활용가능 하게 하기위하여 압축시간을 제한하고 매우 간단한 블록병합 방법을 사용하였다. 압축시간은 inherit 블록의 수가 적을수록 길어지므로 영상이 복잡하면 다소 길어지나 큰 차이는 없다. 화질/압축률은 영상이 단순할수록 다른 방법에 비하여 더 많은 향상을 보인다.

	HV			Adaptive			Qucik Fractal		
영상	시간	PSNR	CR	시간	PSNR	CR	시간	PSNR	CR
Lenna	60	35.8	8.3	17	35.8	9.0	0.024	37.0	9.45
Baboo	168	26.6	4.1	25	26.7	5.2	0.057	26.9	5.52
Boat	90	32.1	5.9	22	32.4	6.6	0.025	32.7	7.27
Pepper	60	33.1	8.1	16	33.4	9.5	0.019	34.0	9.72

표 2: Quick Fractal과 HV 분할방법과 Adoptive Approximate Nearest Search 방법과의 성능 비교.

5. 결론

일반적인 프랙탈 영상압축 방법은 화질을 높이기 위해서 탐색영역을 크게 하고 유사변환을 다양하게 함으로써 압축시간이 매우 길어지게 된다. 제안하는 방법은 레인지블록의 크기를 극히 작게 하고 탐색영역을 아주 작게 하고 유사변환을 극히 단순하게 함으로써 압축률을 유지하면서 압축시간을 수백에서 수만 배 향상시킨다. 본 방법은 화질/압축률 또한 크게 개선한다. 본 방법은 기존의 프랙탈 영상압축 방법에 쉽게 적용되어 성능을 개선 할 수 있으므로 활용도가 높을 것으로 기대된다.

본 방법의 가장 큰 장점은 고화질을 유지하면서 압축시간을 획기적으로 향상시키는 데 있다. 실험결과 CIF (352×288) 크기의 영상들의 압축시간은 0.02초 이하가 된다. 따라서 CIF 크기의 실시간 동영상코딩 I-frame에 사용가능하게 된다.

참고문헌

[1] C.S. Tong, M. Wong, Adaptive Approximate Nearest Neighbor Search for Fractal Image Compression, IEEE Trans. on Image Processing, Vol. 11, No. 6, pp. 605-615, June 2002.

[2] D. Saupe, Fractal Image Compression via Nearest Neighbor Search, in Conf. Proc. NATO ASI Fractal Image Encoding and Analysis, Trondheim, Norway, July 1995.

[3] M. Ruhl, H. Hartenstein, D. Saupe, Adaptive Partitionings for Fractal Image Compression, IEEE International Conference of Image Processing, Oct. 1997.

[4] H. Hartenstein, M. Ruhl, D. Saupe, Region-Based Fractal Image Compression, IEEE Trans. on Image Processing, Vol. 9, No.7, pp. 1171-1184, July 2000.

[5] K. Belloulata, J. Konard, Fractal Image Compression With Region-Based Functionality, IEEE Trans. on Image Processing, Vol. 11, No. 4, pp. 351-362, April 2002.

[6] C. Lai, K. Lam, W. Siu, A Fast Fractal Image Coding Based on Kick-Out and Zero Contrast Conditions, IEEE Trans. on Image Processing, Vol. 12, No.11, pp. 1398-1403, November 2003.

[7] J. Cardinal, Fast Fractal Compression of Greyscale Images, IEEE Trans. on Image Processing, Vol. 10, No.1, pp. 159-163, January 2001.

[8] S. Lee, A Fast Variance-Ordered Domain Block Search Algorithm for Fractal Encoding, IEEE Transactions on Consumer Electronics, Vol. 45, No. 2, pp. 275-277, May 1999.