

프랙탈 영상압축에서 다양한 분할 방식 적용가능 하드웨어 구현

김동현, 송문빈, 정연모
경희대학교 전자공학과
e-mail:me-and-me@daum.net

Hardware Design for Various Partitioning Method of Fractal Image Compression

Donghyun Kim, Moonvin Song, Yunmo Chung
Dept of Electronic Engineering, Kyung Hee University

요 약

프랙탈 압축 알고리즘에서는 필요에 따라 다양한 분할 방식을 사용하고 있다. 기존에는 분할방식에 따라 다른 isometry 변환계수 값만을 사용하고 있다. 본 논문에서는 이러한 문제를 해결하기 위하여 모든 분할 방식에서 공통적으로 사용할 수 있도록 변환 가능한 isometry 변환계수 방식을 제안하였으며 이를 하드웨어로 구현하였다.

I. 서론

인터넷과 모바일 기술의 발전과 더불어 이를 이용한 멀티미디어 산업이 더욱 활발해 지고 있다. 인터넷 및 모바일 상에서 고화질의 이미지 및 동영상을 원활히 전송하고 처리하기 위해서는 효율적인 압축 및 복원 기법이 필요하다. 일반적으로 영상 압축 기법은 Shannon의 정보 이론을 근간으로 하고 있으며, 이는 예측 부호화(predictive coding) 기법과 변환 부호화(transform coding) 기법으로 나뉜다. 그러나 프랙탈 압축 알고리즘은 영상이 자기 유사성(self-similarity)을 갖는 특징을 이용한 것이다[1]. 이 압축 방법은 높은 압축률을 가지면서도 JPEG등과 비교해 복원 영상의 화질을 일정 수준 이상으로 유지할 수 있는 장점을 가진다. 프랙탈 영상 압축 알고리즘은 압축하는 과정에서 많은 계산량으로 시간이 오래 걸린다. 그러나 복원과정은 상대적으로 매우 빠르고 간단하다. 따라서 미리 압축하여 저장한 후 복원하는 비디오나 CD-ROM과 같은 응용분야에 적합하다[8].

본 논문에서는 프랙탈 영상 압축과정에서 다양한

분할 방식에 따른 여러 가지 isometry 변환계수가 사용되는 것을 해결하기 위해 공통적으로 사용할 수 있는 변환 가능한 isometry 변환계수 방식을 하드웨어로 설계하였다.

II. 이론

프랙탈 영상 압축을 수행하기 위해서는 필수적으로 기본 도형을 확대, 축소, 반사, 휨, 회전, 및 이동 과정을 거쳐야 한다. 이때 확대, 축소, 및 회전을 담당하는 2 차원상의 아핀 변환은 (식 1, 2)와 같다 [1,7].

$$w \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} e \\ f \end{pmatrix} = Ax + T \quad (1)$$

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix} = \begin{pmatrix} r_1 \cos \theta_1 & -r_2 \sin \theta_2 \\ r_1 \sin \theta_1 & r_2 \cos \theta_2 \end{pmatrix} \quad (2)$$

$A = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$ 는 2x2의 실수 행렬이며 T 또한 열벡터로 실수 행렬이다. [그림 1]은 (식 1)을 이용한 아

핀 변환을 나타낸 그림이다. 순서대로 확대, 축소, 휨, 및 회전 변환을 나타내고 있다.

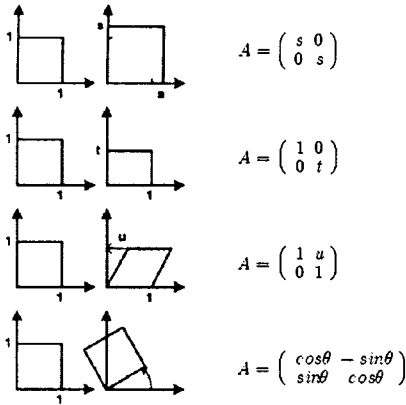


그림 1. 여러 가지 종류의 아핀 변환

3차원에서의 아핀 변환은 (식 3)과 같이 행렬 형태로 나타낼 수 있다. $z=f(x, y)$ 는 x, y 위치의 픽셀의 밝기 값을 나타낸다. s 는 대조도 변환계수이고 o 는 밝기 변환계수이다. 3×3 의 행렬은 확대, 축소, 회전, 대칭 변환을 수행한 뒤 선형 이동을 수행한다. 프랙탈 부호화과정을 수행할 때 매개변수 a, b, c, d 는 회전과 대칭을 나타내는 isometry 변환계수를 사용하여 부호화한다. e, f 값은 domain의 위치를 나타내는 선형 이동 변환을 나타낸다[1,6,8].

$$w \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} a & b & 0 \\ c & d & 0 \\ 0 & 0 & s \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} + \begin{pmatrix} e \\ f \\ o \end{pmatrix} \quad (3)$$

수축 변환들로만 이루어진 집합을 IFS(Iterative Function System)라고 한다. (식 4)의 변환식들을 여러 번 반복 수행하면 IFS이론대로 수렴하는 영상이 [그림 2]와 같이 만들어 진다[9].

$$\begin{aligned} w_1 \begin{pmatrix} x \\ y \end{pmatrix} &= \begin{pmatrix} 0.5 & 0 \\ 0 & 0.5 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} 0 \\ 0.5 \end{pmatrix} \\ w_2 \begin{pmatrix} x \\ y \end{pmatrix} &= \begin{pmatrix} 0.5 & 0 \\ 0 & 0.5 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} 0.5 \\ 0.5 \end{pmatrix} \\ w_3 \begin{pmatrix} x \\ y \end{pmatrix} &= \begin{pmatrix} 0.5 & 0 \\ 0 & 0.5 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \end{pmatrix} \end{aligned} \quad (4)$$

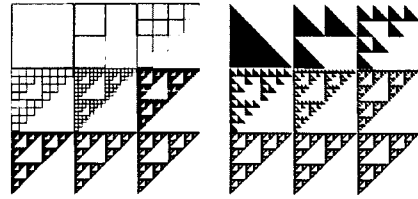


그림 2. IFS를 반복 수행한 결과

[그림 3]은 전체적인 프랙탈 영상압축의 순서도이다[3,8].

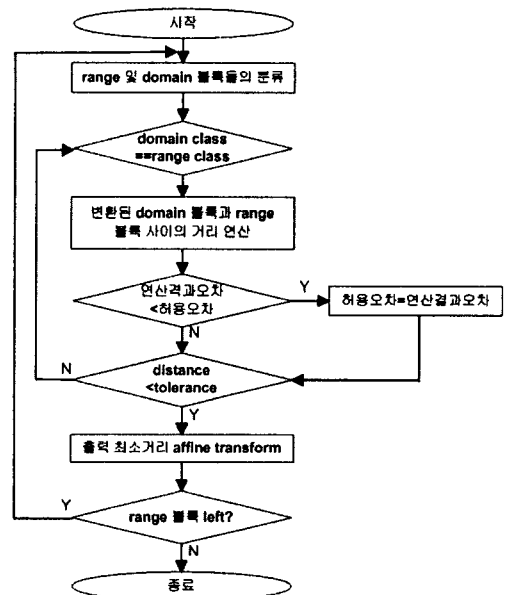


그림 3. 프랙탈 이미지 압축과정

그림에서 range 및 domain블록들을 분류한 후 각 range블록마다 (식 5)의 RMSE(Root Mean Square Error)가 최소가 되는 domain블록을 찾아서 그 블록을 변환하는 과정에서 아핀 변환식을 사용한다. RMSE는 (식 5)의 range 및 domain블록들인 $A=(a_1, a_2, a_3, \dots, a_m)$ 와 $B=(b_1, b_2, b_3, \dots, b_m)$ 간의 유사성을 나타낸다.

$$RMSE = d_c(A, B) = \sqrt{\frac{1}{m} \sum_{i=1}^m (a_i - b_i)^2} \quad (5)$$

RMSE가 적을수록 두 블록은 서로 유사하다.

표 1. isometry 변환계수

symmetry	행렬	설명
0	$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$	identification
1	$\begin{pmatrix} -1 & 0 \\ 0 & 1 \end{pmatrix}$	reflection in y-axis
2	$\begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$	reflection in x-axis
3	$\begin{pmatrix} -1 & 0 \\ 0 & -1 \end{pmatrix}$	180° rotation
4	$\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$	reflection in line y=x
5	$\begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}$	90° rotation
6	$\begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}$	270° rotation
7	$\begin{pmatrix} 0 & -1 \\ -1 & 0 \end{pmatrix}$	reflection in line y=-x

[표 1]은 Jacquin의 균등분할이나 Fisher의 quadtree 분할방식을 사용한 프랙탈 영상압축에서 주로 사용하는 8가지의 기하학적인 변화에 대한 고려이다[2,4,5]. 이 변환계수는 이미지를 분할하는 형태에서 삼각분할이나 다각형분할 등 대각선모서리 부분을 이용한 분할방식에는 적용하지 못한다.

이를 보완하기 위하여 (식 3)의 값을 상황에 따라 변환 가능하게 설계하였다.

III. 변환 가능한 아핀 변환의 하드웨어 설계

자연 세계에 존재하는 영상들은 자기 닮음성의 성질이 크지 않고 찾아내기가 어렵다. 또한, 색 정보를 가지고 있으므로 단순한 형태의 IFS로는 표현하기 어렵고 (식 3)과 같은 확장된 형태의 아핀 변환을 사용해야 한다. 이를 이용한 아핀 변환에 대한 하드웨어 블록 다이어그램은 [그림 4]와 같다.

Isometry 변환계수 부분인 ANG_BLOCK은 균등분할, quadtree분할, 삼각분할, 다각형분할 등 다양한 분할방식에 모두 적용 가능하도록 설계함하여 그 범위를 다양하게 적용할 수 있다.

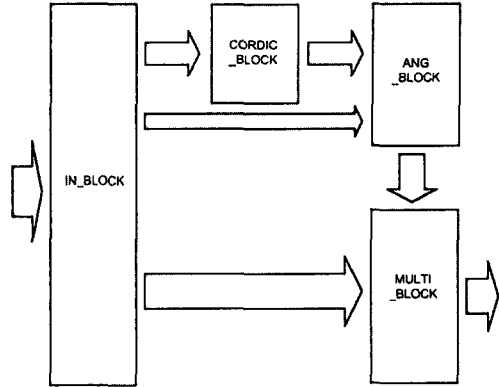


그림 4. TOP_BLOCK 내부

전체적인 동작은 유사블록 탐색에 따른 11개의 매개변수 값을 입력으로 받아서 (식 3)에 의해서 변환된 $w(x)$, $w(y)$, $w(z)$ 값을 MULTI_BLOCK에서 출력한다. 이 과정을 모든 압축 대상이 되는 range블록들에 대해 수행하면 각 range블록마다 하나의 아핀 함수를 갖는다. 아핀 함수의 계수들을 부호화하면 전체 부호화 과정이 끝난다.

각 블록을 자세히 살펴보면 IN_BLOCK은 range블록과 domain블록 사이의 RMSE가 최소가 되는 블록의 데이터를 입력받는다. 입력받은 11개의 매개변수는 CLK과 enable신호에 의해서 각각 다음 블록으로 전달한다.

CORDIC_BLOCK은 두 개의 16비트 데이터를 입력받아서 4개의 삼각함수 값을 출력하는 블록이다.

ANG_BLOCK은 16비트인 4개의 삼각함수 값들과 4비트의 데이터를 입력받아서 (식 2)의 a , b , c , d 값을 계산하여 출력하는 블록이다.

MULTI_BLOCK은 (식 2)의 4비트 a , b , c , d 값과 32비트의 contrast값, 9비트의 brightness값, 8비트의 X, Y, Z값을 입력으로 사용하여 최종 결과인 $w(x)$, $w(y)$, $w(z)$ 값을 출력하는 블록이다.

[그림 5]는 CORDIC_BLOCK의 VHDL코드를 나타낸다. 삼각함수 sin과 cos값을 출력하는 코드이다. 이때 삼각함수는 $0^\circ - 90^\circ$ 의 범위를 갖는다. 이 CORDIC_BLOCK은 기존의 quadtree와 균등분할에 사용되는 3비트 Isometry 변환계수 및 삼각형분할과 다각형분할 등에 사용되는 Isometry 변환계수들을 포함한다.

```

function CATAN(n : natural) return integer is
variable result : integer;
begin
  case n is
    when 0 => result := 16#02000#;
    when 1 => result := 16#01240#;
    when 2 => result := 16#09734#;
    when 3 => result := 16#05111#;
    when 4 => result := 16#02881#;
    when 5 => result := 16#01450#;
    when 6 => result := 16#0A27#;
    when 7 => result := 16#0518#;
    when 8 => result := 16#020C#;
    when 9 => result := 16#0146#;
    when 10 => result := 16#0A3#;
    when 11 => result := 16#051#;
    when 12 => result := 16#029#;
    when 13 => result := 16#014#;
    when 14 => result := 16#0A#;
    when 15 => result := 16#05#;
    when 16 => result := 16#03#;
    when 17 => result := 16#01#;
    when others => result := 16#0#;
  end case;
  return result;
end CATAN;
    
```

그림 5. arc-tangent값을 계산하는 코드

IV. 결론

프랙탈 이미지 압축 방법이 기존의 여러 압축방법과 비교하여 좋은 압축률을 가지고 있음에도 불구하고 압축하는 과정에서 많은 연산을 수행하기 때문에 오랜 시간이 걸린다. 이러한 시간을 줄이기 위하여 압축과정에서 다양한 이미지 분할방식을 사용되고 있다. 그러나 다양한 분할방식마다 필요에 따라 여러 가지 isometry 변환계수를 계산하여 사용하고 있다. 본 논문에서는 이러한 문제를 해결하기 위하여 모든 분할 방식에서 공통적으로 사용할 수 있는 변환 가능한 isometry 변환계수 방식을 제안하였으며 이를 하드웨어로 구현하였다. 또한 소프트웨어로 압축하기 힘든 임베디드 환경 및 모바일 환경에서의 영상 압축을 위한 하드웨어 설계를 고려하여 하드웨어 디자인을 하였다.

참고문헌

[1] M. F. Barnsley, *Fractals Everywhere*, Academic Press Inc. , San Diego, 1988.
 [2] A. E. Jacquin, "Image Coding Based on a Fractal Theory of Iterated Contractive Image Transformations," *IEEE Transactions on Image Processing.*, Vol. 1. No. 1. Jan, pp18-30, 1992.
 [3] Y. Fisher, Ed., *Fractal Image Compression, Theory and Application*, Springer-Verlag, New York, NY, USA, 1995.
 [4] A. E. Jacquin, "A fractal theory of iterated

Markov operators with applications to digital image coding," *Ph.D. dissertation*, Georgia Inst. Technol., Atlanta, 1989.

[5] J. M. Beaumont, "Image data compression using fractal techniques," *British Telecommunications Technical Journal*, vol. 9(4), Oct. pp93-108, 1991.
 [6] D. M. Monro, F.Dudbridge, "Fractal approximation of image blocks," *ICASSP 92*, Vol. 3, pp.485-488, 1992.
 [7] D. M. Monro, "A Hybrid Fractal Transform," *ICASSP 93*, pp169-172, 1993.
 [8] 김종구, "A Fractal Image Compression using The Normalized Variance", *Ph.D. dissertation*, Aju Inst. Aug. 2001.
 [9] 서기범, "A Study on the Image Compression and Hardware Design", *Ph.D. dissertation*, Hanyang Inst. June. 2000.