8진트리로 표현된 3차원 영상의 빠른 기하학적 변환

허 영 남† 박 승 진†† 김 응 곤†††

요 약

움직이는 3차원 물체를 화면상에 디스플레이하기 위해서는 많은 기하학적 연산을 필요로 하는데 CAD나 애니메이션 응용에서는 가능한한 빠른 속도로 변환을 수행하는 것이 중요한 문제이다. 8진트리로 표현된 3차원 영상의 기하학적 변환을 수행하기 위한 기존의 방법은 8진트리의 모든 노드에 변환 행렬연산을 적용함으로써 가능하였다. 본 논문에서는 8진트리로 표현된 3차원 영상의 기하학적 변환을 효율적으로 수행하기 위하여 8진트리의 각 노드에 기본벡터를 이용하여 직각좌표로 변환시키는 효율적인 방법을 제안한다. 본 논문의 공식을 이용하면 기하학적변환을 위한 행렬 연산을 기본벡터에만 적용하면 되고, 덧셈과 2의 지수에 의한 곱셈 연산만이 소요된다.

Fast Geometric Transformations of 3D Images Represented by an Octree

Yeong-Nam Heo† Seung-Jin Park†† Eung-Kon Kim†††

ABSTRACT

Geometric transformations require many operations in displaying moving 3D objects on the screen and a fast computation is a important problem in CAD or animation applications. The general method to compute the transformation coordinates of an object represented by an octree must perform the operations on every node. This paper proposes an efficient method that computes the rectangular coordinates ofthe vertices of the octree nodes into the coordinates of the universe space using the basicvectors in order to compute quickly geometric transformations of 3D images represented by an octree. The coordinates of the vertices of each octant are computed by using the formula presented here, which requies additions and multiplications by powers of 2. This method has a very fast execution time and is compared with the general computation method.

1. 서 론

컴퓨터 그래픽스의 응용분야 특히, CAD, 애니메이션, 컴퓨터 비젼, 영상 처리 시스템 등에 있어서 3차원 물체의 효율적인 표현은 매우 중요한 문제이다. 표현하고자 하는 물체가 복잡해짐에 따라 필요한 변환이나 연산을 수행한후 디스플레이하는데 많은 시간이 소요되어 비효율적이기 때문이다.

8진트리(octree) 표현 기법[1]-[4]는 2차원 평면에서의 4진트리(quadtree)를 3차원 표현 방식으로 확장시킨 데이타 구조로서 3차원 공간상의 연관성이 포함되어 있으며, 데이타가 압축된 형식을 갖는 특성이 있다. 이러한 표현 방식은 트리 탐색을 활용함으로써 계산적인 면에서 효율적으로 탐색할 수 있고, 평행 이동(translation), 신축(scaling), 회전 이동(rotation) 및 은면 제거(hidden-surface removal)와 같은 그래픽의 기본적인 알고리즘을 공간적 연관성을 이용함으로써 효율적으로 처리할 수 있는 표현 방식이다.

*이 논문은 1994년도 교육부지원 한국학술진흥재단의 지방대육성과제 학술조성비에 의하여 연구되었음.
† 정 회 원 : 순천대학교 전자계산학과 교수
†† 정 회 원 : 경상대학교 부속병원
††† 정 회 원 : 순천대학교 전자계산학과 조교수
논문접수 : 1995년 8월17일, 심사완료 : 1995년11월13일

따라서 8진트리는 컴퓨터 그래픽스의 응용분야, 애니메이션, CAD, 로보틱스에서 움직이는 물체의 표현, 의학 분야에서 단층 촬영 결과를 합성하여 3차원 형상을 구성하는데 이용하는 등 매우 다양한 분야에서 3차원 물체의 효율적인 표현 기법으로 활용되고 있다.[2]-[8]

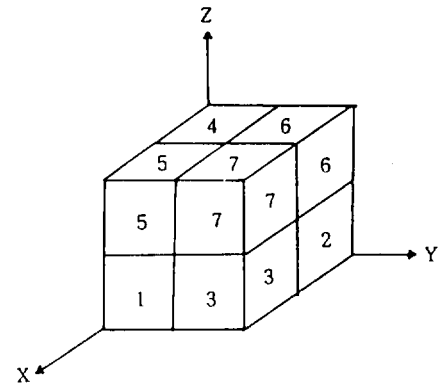
컴퓨터 그래픽스의 응용 분야에서는 물체를 표현하는 영상의 위치와 형태를 변화시켜 화면상에 물체의 연속적인 동작을 표현하거나 물체의 크기를 확대 또는 줄일 필요가 있다. 이러한 여러 가지 조작은 8진트리로 표현된 물체의 경우, 8진트리의 각 노드에 적당한 기하학적 변환을 적용시킴으로써 이루어 진다.

3차원 영상을 화면상에 디스플레이하기 위해서는 많은 기하학적 변환이 필요하며, 하나의 물체는 보통 많은 8진트리의 노드들로 구성되므로 가능한한 빠른 속도로 변환을 수행하는 것이 중요한 문제이다. 8진트리로 표현된 3차원 영상의 기하학적 변환을 수행하기 위한 기존의 방법[2, 9]은 8진트리의 모든 노드에 변환행렬 연산을 적용함으로써 가능하였다. 그 결과, 움직이는 물체와 같이 많은 변환을 수행해야 하는 경우나 임의의 축에 관한 회전 이동과 같은 복잡한 조합 변환의 경우 매우 많은 연산과 시간을 요한다.

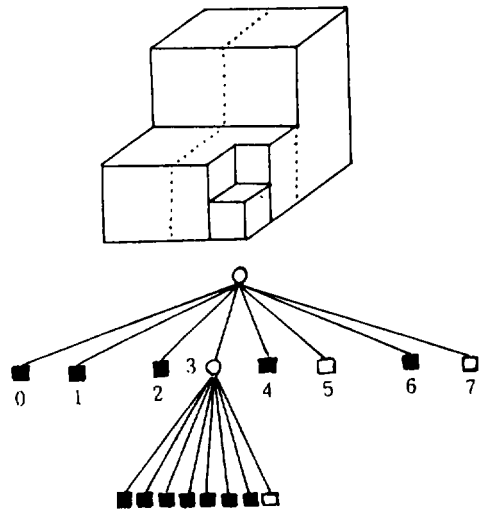
따라서 본 연구에서는 8진트리로 표현된 3차원 영상의 기하학적 변환을 수행하는데 소요되는 시간을 크게 줄이기 위하여 8진트리의 각 노드를 직각 좌표로 변환시키는 효율적인 방법을 제안하고, 이를 PC상에서 구현하여 그 효용성을 입증한다.

2. 8진트리의 구성

8진트리의 공간은 $2^n \times 2^n \times 2^n$개의 단위 입방체로 구성된 x, y, z축을 갖는 3차원 공간으로 구성할 수 있다. 여기서 n은 공간 분할 변수이며, 2^n은 8진트리 공간상의 길이를 나타낸다. 각 단위 입방체는 물체의 내부 또는 외부에 있는 가에 따라 1 또는 0의 값을 갖는다. 입방체 공간을 8개의 작은 입방체로 순환적으로 재분할해 나감으로써 물체를 8진트리로 표현할 수 있다. 분할된 각각의 입방체를 8분체(octant)라고 한다. $2^d x 2^d x 2^d (1 \le d \le n)$의 8분체는 8분체에 포함된 단위 입방체가 완전히 1이 아니거나 0이 아니면 8개의 더 작은 $2^{d-1} x 2^{d-1} x 2^{d-1}$의 8분체로 분할된다. 8분체가 하나의 값으로 된 단위 입방체를 완전히 포함할 때까지 순환적으로 분할해 나간다. 이런 과정으로 생성된 각 8분체는 (그림 1)과 같이 0에서 7까지의 값으로 레이블된다. 이러한 순환적 재분할 과정을 각 노드들이 8개의 잎(leaf)이나 8개의 자식을 가지는 차수(degree) 8의 트리로 표현할 수 있는데 이러한 트리를 8진트리라고 한다[2]. 8진트리의 루트노드는 가



(그림 1) 8분체의 숫자화 및 방향
(Fig. 1) Labels of the octants



(그림 2) 물체의 예와 8진트리의 표현
(Fig. 2) An example object and its octree representation

장 큰 입방체 전공간을 의미하며, 잎노드는 물체가 완전히 포함된 8분체노드를 의미한다. 각 노드는 8개의 분할된 8분체로 구성된 자식노드를 가지며 그 노드를 자식노드에 비하여 부모노드라 정의한다. 노드를 구성하는 공간이 모든 물체를 포함하고 있으면 흑색노드라 하며 ■로 표시하고, 모든 배경을 포함하고 있으면 백색노드라 하며 □로 표시하고, 배경과 물체가 혼합되어 있으면 회색노드라 하며 ○로 표시하고, 흑색노드와 백색노드는 잎노드, 회색노드를 내부노드이다.

(그림 2)는 간단한 예의 물체와 그에 대한 8진트리 표현을 나타내고 있다.

$2^d \times 2^d \times 2^d$의 8분체의 길이는 2^d로 정의되며, 그 8분체에 해당하는 노드의 레벨은 d, 깊이는 (n−d)가 된다[2]. 따라서 단위 입방체에 해당하는 노드는 레벨 0에 있고, 8진트리의 루트는 레벨 n(깊이 0)에 있다.

메모리 사용의 효율성을 높이기 위하여 흑색노드만을 부호화한 8진트리를 선형 8진트리라고 한다. 선형 8진트리는 배경을 제외한 오직 물체만을 표시한 잎노드만으로 구성된 압축된 형태를 갖는다. 본 논문에서는 선형 8진트리로 구성된 물체의 기하학적 변환을 다룬다.

3. 8진트리 노드의 좌표변환

평행 이동, 신축, 회전 이동 및 화면으로의 투영 등과 같은 기하학적 변환을 수행할 때는 8진트리의 모든 노드들에 대하여 행렬 변환 연산을 수행해야 하는데, 본 연구에서는 기하학적 변환 연산을 간단하게 수행하기 위하여, 기존의 방법[2 9]과 같이 모든 노드들에 대하여 표준 행렬 변환을 적용하지 않고, 위치코드로 표현된 8진트리의 노드와 그 노드에 해당하는 8분체의 꼭지점을 직각 좌표계의 좌표로 변환하는 공식을 본 절에서 유도한다.

$2^n x 2^n x 2^n$ 공간으로 둘러싸인 물체는 최대 레벨 n의 선형 8진트리로 표현할 수 있다. 8진 숫자열 $c_{n-1}\ c_{n-2} \cdots\cdots c_0$로 표현된 각 8분체의 위치코드 C는 다음과 같이 직각좌표 (x,y,z)로 변환된다.[2 9]

각 8진 숫자 cd는 각각 노드좌표 x,y,z의 2진 표현 xd,yd,zd비트들을 연결시켜 표현된다. xd, yd,zd로 표현된 8진 숫자는 공간을 (n−d)번째 재분할한 노드의 위치를 나타낸다.

물체의 3차원 좌표를 8진 위치코드로 표현하기 위해서는 각 좌표를 2진수로 표현해야 한다. x, y, z가 식 (1)과 같다면 8진 숫자 cd는 식 (2)와 같이 된다.[2 9]

$$x=\sum_{i=0}^{n-1} x_i 2^i, \quad y=\sum_{i=0}^{n-1} y_i 2^i, \quad z=\sum_{i=0}^{n-1} z_i 2^i \tag{1}$$

$$c_i=x_i 2^2+y_i 2^1+z_i 2^0 \tag{2}$$

이러한 부호화 기법을 간단히 표현하면 식 (3)과 같다.

$$C=\sum_{i=0}^{n-1} c_i 8^i=\sum_{i=0}^{n-1}(4x_i+2y_i+z_i)8^i \tag{3}$$

여기서 $c_i \in \{0,1,2,3,4,5,6,7\}$, $x_i,y_i,z_i \in \{0,1\}$, $i \in \{0,1,2,\cdots,n-1\}$이다.

C는 8진수 정수이며, 숫자 c_i는 좌표공간을 8분체로 순환재분할할 때 트리의 루트에서 특정 8분체까지의 통로를 나타내는 방향코드를 의미한다. 예를 들어, 위치코드가 763인 노드의 x,y,z 좌표는 식 (1)~(3)에 의해 (3,7,5)가 된다.

식 (1)과 같은 기존의 방법[2 9]을 이용하여 각 노드의 위치코드를 8분체 꼭지점의 좌표(x,y,z)로 변환할 경우 물체를 평행이동, 신축, 회전이동 등의 기하학적 변환시 모든 노드의 8분체에 대하여 변환행렬 연산을 수행해야만 한다. 본 논문에서는 모든 노드에 변환행렬 연산을 적용하지 않고 효율적으로 기하학적 변환을 수행하기 위하여 전체 공간을 8개의 꼭지점으로 나타내기 위한 기본벡터 U(0), U(1),···, U(7)를 식 (4)~(9)와 같이 정의한다.

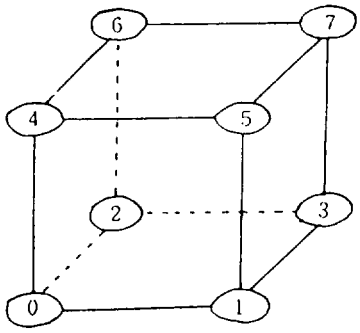
4개의 기본벡터 U(0), U(1), U(2), U(4)는 식(4)와 같다.

$$U(0)=[0\ 0\ 0],\ U(1)=[0\ 0\ 2n],$$
$$U(2)=[0\ 2^n\ 0],\ U(4)=[2^n\ 0\ 0] \tag{4}$$

벡터 U(3), U(5), U(6), U(7)은 식(5), (6), (7), (8)과 같이 벡터 U(1), U(2), U(4)의 선형조합으로 표현한다.

$$U(3)=U(1)+U(2)=[0\ 2^n\ 2^n] \quad (5)$$
$$U(5)=U(1)+U(4)=[2^n\ 0\ 2^n] \quad (6)$$
$$U(6)=U(2)+U(4)=[2^n\ 2^n\ 0] \quad (7)$$
$$U(7)=U(3)+U(4)=[2^n\ 2^n\ 2^n] \quad (8)$$

선형 8진트리의 어떤 레벨에 위치한 8분체의 꼭지점 좌표를 기본벡터를 이용하여 구하는 공식을 유도해 보자. 공간상의 8개의 꼭지점에 대한 좌표 벡터 U(0), U(1), ···, U(7)과 직각좌표로 변환하고자 하는 8분체의 위치코드 C를 알고 있다고 하고, (그림 3)과 같이 8분체의 각 꼭지점이 레이블되어 있다고 하자.



(그림 3) 8분체의 꼭지점 레이블
(Fig. 3) Vertices labels of an Octant

F를 꼭지점 k와 선형 8진트리 노드의 위치코드 C를 그 성분이 꼭지점 k의 직각좌표 (x,y,x)인 벡터 F(C,k)로 사상시키는 함수라고 하자. 유클리드기하학의 중점공식을 이용하면 공간상의 최초 재분할 8분체 c_{n-1}의 꼭지점 k의 좌표는 식(9)와 같이 된다.

$$F(c_{n-1},\ k)=\frac{U(c_{n-1})+U(k)}{2} \quad (9)$$

공간상의 두번째 재분할 8분체 $c_{n-1}\ c_{n-2}$의 꼭지점 k의 좌표는 식 (10)과 같이 된다.

$$F(c_{n-1}c_{n-2},k)$$
$$=\frac{F(c_{n-1},c_{n-2})+F(c_{n-1},k)}{2}$$
$$=\frac{1}{2}\left(\frac{U(c_{n-1})+U(c_{n-2})}{2}+\frac{U(c_{n-1})+U(k)}{2}\right)$$
$$=\frac{1}{4}(2U(c_{n-1})+U(c_{n-2})+U(k)) \quad (10)$$

일반적으로, 공간상의 j번째 재분할 8분체의 $c_{n-1}\ c_{n-2}\ c_{n-j}$의 꼭지점 k의 좌표는 식(11)와 같다.

$$F(c_{n-1}c_{n-2}\cdots c_{n-j+1}c_{n-j},\ k)$$
$$=\frac{F(c_{n-1}c_{n-2}\cdots c_{n-j+1},\ c_{n-j})+F(c_{n-1}c_{n-2}\cdots c_{n-j+1},\ k)}{2}$$
$$=\frac{1}{2^j}(U(k)+U(c_{n-1})2^{j-1}+U(c_{n-2})2^{j-2}+\cdots+U(c_{n-j})2^0)$$
$$=\frac{1}{2^j}\left(U(k)+\sum_{i=0}^{j-1}U(c_{n-j+i})2^i\right) \quad (11)$$

공간상의 (j+1)번째 재분할 8분체에 대해서는 다음 식(12)를 적용할 수 있다.

$$F(c_{n-1}c_{n-2}\cdots c_{n-j}c_{n-(j+1)},\ k)$$
$$=\frac{1}{2^{j+1}}\left(U(k)+\sum_{i=0}^{j}U(c_{n-(j+1)+i})2^i\right) \quad (12)$$

식 (9)과 (10)을 이용하면 식(13)과 같이 된다.

$$F(c_{n-1}c_{n-2}\cdots c_{n-(j+1)},\ k)$$
$$=\frac{1}{2}(F(c_{n-1}c_{n-2}\cdots c_{n-j}c_{n-(j+1)}+F(c_{n-1}c_{n-2}\cdots c_{n-j},k))$$
$$=\frac{1}{2^{j+1}}(U(c_{n-(j+1)})+\sum_{i=0}^{j-1}U(c_{n-j+i})2^i)+\frac{1}{2^{j+1}}(U(k)+\sum_{i=0}^{j-1}U(c_{n-j+i})2^i)$$
$$=\frac{1}{2^{j+1}}(U(c_{n-(j+1)})+U(k)+2\sum_{i=0}^{j-1}U(c_{n-j+i})2^i)$$
$$=\frac{1}{2^{j+1}}(U(c_{n-(j+1)})+U(k)+\sum_{i=0}^{j-1}U(c_{n-j+i})2^{i+1})$$
$$=\frac{1}{2^{j+1}}(U(k)+\sum_{i=0}^{j}U(c_{n-(j+1)+i})2^i) \quad (13)$$

따라서 트리의 잎노드에 위치한 8분체 $c_{n-1}\ c_{n-2}\cdots c_0$의 경우, 꼭지점 k의 좌표는 식(14)와 같이 된다.

$$F(c_{n-1}c_{n-2}\cdots c_0,\ k)$$
$$=\frac{1}{2^n}(U(k)+U(c_{n-1})2^{n-1}+U(c_{n-2})2^{n-2}+\cdots+U(0)2^0)$$
$$=\frac{1}{2^n}(U(k)+\sum_{i=0}^{n-1}U(c_i)2^i) \quad (14)$$

예를 들어 $2^3\times2^3\times2^3$ 공간에서 위치코드 512인 8분체에 대한 꼭지점 5의 좌표는 다음 식(15)와 같이 된다.

$$F(512,\ 5)=\frac{1}{2^3}(U(5)+2^2U(5)+2^1U(1)+2^0U(2))$$
$$=\frac{1}{8}(|8\ 0\ 8|+4|8\ 0\ 8|+2|0\ 0\ 8|+|0\ 8\ 0|)$$
$$=|5\ 1\ 6| \quad (15)$$

4. 3차원 영상의 기하학적 변환

3차원 영상의 기본적인 기하학적 변환에는 평행이동(translation), 신축(scaling), 회전이동(rotation) 등이 있다. 이들 연산 중에서 임의의 축에 관한 회전이동은 좌표축에 대한 평행이동과 회전이동을 수행하는 일련의 행렬들의 곱인 조합행렬을 구함으로써 가능한데 다른 변환에 비하여 매우 많은 연산과 시간을 요한다.

8진트리로 표현된 물체에 대하여 기하학적 연산을 수행하기 위한 식(1)을 이용한 기존의 방법[2, 9]에서는 다음과 같은 절차에 의하여 수행된다.

1) 8진트리의 모든 노드의 위치코드 C를 기존의 변환방법에 따라 직각좌표로 변환한다.
2) 기하학적 변환행렬 연산을 모든 노드의 보이는 면에 속한 꼭지점에 대하여 적용한다.

본 논문에서는 8진트리로 표현된 3차원 영상의 기하학적 변환을 효율적으로 수행하기 위하여 3장에서 구한 8진 트리 노드의 좌표변환공식 (11)이나 (14)를 이용하여 다음과 같은 절차에 의하여 기하학적 변환을 수행한다.

1) 기본벡터 U(0), U(1), U(2), U(4)에 기하학적 변환행렬연산을 수행한다.
2) 기본벡터 U(3), U(5), U(6), U(7)을 식(6)~(9)와 같이 구한다.
3) 공식 (11)이나 (14)를 이용하여 8진트리의 모든 보이는 면에 속한 노드에 대한 기하학적 변환좌표를 구한다.

임의의 축에 관한 회전이동 변환에 대하여 이를 적용하기 위하여 회전축이 지나는 점의 좌표를 (x,y,z)라 하고, 회전축에 따른 단위벡터를 (a,b,c)라 하자. 이 축에 대하여 만큼 물체를 회전이동시키기 위한 일반적인 절차는 다음과 같다[10].

(1) 평행이동행렬 T1을 이용하여 점(x,y,z)를 원점(0,0,0)으로 사상시키는 새로운 좌표계로 물체를 평행이동시킨다.
(2) 새로운 좌표계의 X,Y축에 대하여 단위벡터 (α,β,γ)가 Z축에 따른 단위벡터로 사상되도록 각각 회전이동연산을 수행한다. 이 두회전이동행렬을 각각 Rx, Ry라 하자.
(3) 새로운 좌표계의 Z축에 대하여 만큼 회전이동한다. 이 때의 변환행렬을 $Rz(\theta)$라 하자.
(4) 절차 (2)의 역연산을 수행한다. 이 때의 변환행렬을 Ry^{-1}, Rx^{-1}라 하자.
(5) 절차 (1)의 역연산을 수행한다. 이 때의 변환행렬을 $T1^{-1}$라 하자.

조합변환행렬T는 식 (16)과 같다[10].

$$T=T1\ Rx\ Ry\ Rz(\theta)\cdot Ry^{-1}\cdot Rx^{-1}\cdot T^{-1}=$$
$$\begin{vmatrix} (a^2-1)(1-\cos\theta)+1 & ab(1-\cos\theta)-c\sin\theta & ac(1-\cos\theta)+b\sin\theta & 0 \\ ab(1-\cos\theta)+c\sin\theta & (b^2-1)(1-\cos\theta)+1 & bc(1-\cos\theta)-a\sin\theta & 0 \\ ac(1-\cos\theta)-b\sin\theta & bc(1-\cos\theta)+a\sin\theta & (c^2-1)(1-\cos\theta)+1 & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix} \quad (16)$$

여기서 $\sqrt{a^2+b^2+c^2}=1$이다.

기존의 방법[2, 9]에서는 위의 다섯가지 단계를 8진트리의 모든 보이는 면에 속한 꼭지점에 대하여 적용하여야 한다. 그러나 본 논문에서는 다음 절차 1)~3)과 같이 식(11)이나 (14)를 이용함으로써 위의 다섯가지 단계를 8진트리의 루트노드에 해당하는 4개의 기본벡터에만 적용하면 되므로 효율적이다.

1) 기본벡터 U(0), U(1), U(2), U(4)를 다음과 같이 구한다.
   $U(0) = [0\ 0\ 0\ 1]\cdot T$
   $U(1) = [0\ 0\ 2^n\ 1]\cdot T$
   $U(2) = [0\ 2^n\ 0\ 1]\cdot T$
   $U(4) = [2^n\ 0\ 0\ 1]\cdot T$
2) 기본벡터 U(3), U(5), U(6), U(7)을 다음과 같이 구한다.
   U(3) = U(1)+U(2)
   U(5) = U(1)+U(4)
   U(6) = U(2)+U(4)
   U(7) = U(3)+U(4)
3) 공식 (11)이나 (14)를 이용하여 8진트리의 모든 보이는 면에 속한 노드에 대한 기하학적 변환좌표를 기본벡터를 이용하여 구한다.

다음 알고리즘은 본 논문의 방법에 의하여 기하학적 변환을 수행한 후 8진트리의 노드들을 디스플레이하는 알고리즘을 나타낸 것이다.

```
[Algorithm]
Octree-display( )
{
  /*determine octant's visible surfaces*/
  VIS-SURFACES(surface-count,surface[ surface
```

```
-count])
determine the basic vectors, U(0), U(1), U(2), and
U(4) according to the
geometric transform ;
/*determine the other vectors, U(3), U(5), U(6), and
  U(7)*/
  U(3)=U(1)+U(2) ;
  U(5)=U(1)+U(4) ;
  U(6)=U(2)+U(4) ;
  U(7)=U(3)+U(4)) ;
  /*for each node of the octree*/
  for(i=0;i<n;i++)
  {
    /*for visible surfaces of the current node*/
    for(j=0 ; j<surface-count ; j++)
    {
      /*for each vertex of the current visible
       face*/
      for(k=0 ; k<4;k++)
      {
        /*determine a vertex*/
        v=vertex-id(surface[j], k) ;
        /*detremine coordinates of the vertex
          using basic vectors*/
        F(Ci, v)=1/2^n (U(v) + Σ_{l=0}^{n-1} cl2^l) ;
      }k/*k*/
      /*draw the current visible face*/
      DRAW(surface[j]);
    }/*j*/
  }/*i*/
}/*end of algorithm*/
```

5. 실증 및 분석

8진트리로 표현된 물체에 대하여 기하학적 연산을 수행하기 위한 식(1)을 이용한 지금까지의 방법[2, 9]에서는 먼저 8진트리 노드의 위치코드를 직각좌표로 변환한 다음, 기하학적 변환행렬 연산을 모든 노드의 보이는 면에 속한 꼭지점에 대하여 적용해야 한다. 따라서 노드당 보이는 면이 3개일 경우 8진트리의 노드의 수가 N일 때 변환행렬 연산 수행횟수는 7N이 된다.

본 논문에서는 식(11)이나 (14)를 이용함으로써 기하학적변환을 위한 행렬 연산을 8진트리의 루트노드에 해당하는 4개의 기본벡터에만 적용하면 되고, 공간상의 다른 꼭지점 3, 5, 6, 7에 대한 기본벡터 U(3), U(5), U(6), U(7)은 식 (6)~(9)와 같이 기본벡터 U(1), U(2), U(4)의 선형조합으로 구할 수 있다. 즉 식(11)나 (14)를 이용하면 기하학적 변환을 위하여 필요한 행렬연산을 수행할 8진트리의 노드는 루트노드만 해당되므로 매우 효율적임을 알 수 있다. 다른 모든 노드의 꼭지점의 좌표는 식(11)이나 (14)를 이용하여 기본벡터에 의하여 간단히 구할 수 있다. 또한 기존의 변환공식[2, 9]을 이용하면 변환행렬연산을 수행하기 이전에 8진트리 노드의 위치코드를 직각좌표로 변환하여야 하나 여기서는 그럴 필요가 없다.

〈표 1〉은 노드의 수가 32, 64, 128개일 때, 노드당 보이는 면의 수가 각각 1, 2, 3개일 경우 수행해야 할 변환행렬 연산의 수를 기존의 방법과 비교하여 나타낸 것이다. 기존의 변환공식을 이용할 경우 노드의 수가 증가함에 따라 변환행렬연산의 수가 비례하여 증가하지만, 본 논문에서는 노드의 수에 관계없이 4회의 연산만이 소요된다.

〈표 1〉 변환행렬연산의 수 비교
〈Table 1〉 Comparison of the number of transformation matrix operations

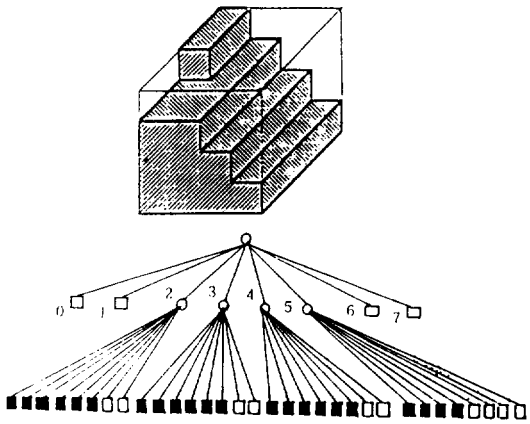
노드의 수	노드당 보이는 면의 수	식 (1)에 의한 연산의 수	본 논문의 방법에의한 연산 수
32	1	128	4
	2	192	4
	3	224	4
64	1	256	4
	2	384	4
	3	448	4
128	1	512	4
	2	768	4
	3	896	4

〈표 2〉는 기하학적 변환시 8진트리의 모든 노드의 보이는 면에 대한 꼭지점의 변환좌표를 구하는데 소요되는 산술연산의 수를 비교한 것이다. 식(1)을 이용한 기존의 방법은 노드의 위치코드를 직각좌표로 변환하기 위한 연산과 기하학적 변환행렬을 적용하기 위한 연산이 필요하나, 식(11)을 이용한 논문의 경우는 기하학적 변환행렬을 적용하기 위한 연산만이 필요하다. 노드의 수가 32개이고 노드당 보이는 면의 수가 1개일 때 기하학적 변환시 변환해야 할 8

분체의 꼭지점의 수는 128개가 되고 변환할 노드가 모두 깊이 2에 있다고 가정할 때, 소요되는 연산의 수는 기존의 방법을 이용하면 위치코드를 직각좌표로 변환시 식(1)에 의하여 꼭지점당 3회의 덧셈과 6회의 곱셈이 필요하고 변환행렬 연산시 꼭지점당 12회의 덧셈과 16회의 곱셈이 각각 필요하고, 본 논문의 방법을 이용하면 기본벡터를 구하는데 모두 48회의 덧셈과 48회의 곱셈이 각각 필요하고 식(11)에 의하여 꼭지점당 6회의 덧셈, 3회의 곱셈과 3회의 나눗셈이 각각 필요하다.

〈표 2〉 산술연산의 수 비교
〈Table 2〉 Comparison of the number of arithmetic operations

노드의 수	노드당 보이는 면수의 수	식(1)에 의한 연산의 수				본논문의 방법에 의한 연산의 수		
		노드 위치 코드의 직각좌표화		변환행렬 연산				
		덧셈	곱셈	덧셈	곱셈	덧셈	곱셈	나눗셈
32	1	384	768	1536	2048	996	432	384
	2	576	1152	2304	3072	1280	624	576
	3	672	1344	2688	3584	1472	720	672
64	1	768	1536	3072	4096	1664	816	768
	2	1152	2304	4608	6144	2432	1200	1152
	3	1344	2688	5376	7168	2816	1392	1344



(그림 4) 물체의 예와 8진트리 표현
(Fig. 4) An example object and its otree representation

〈표 3〉은 (그림 2)와 (그림 4)의 물체에 대한 기하학적 변환시 8진트리의 모든 노드의 보이는 면에 대한 꼭지점의 변환좌표를 구하는데 소요되는 산술연산의 수를 비교한 것이다.

〈표 3〉 그림 2와 그림 4의 물체에 대한 산술연산 수의 비교
〈Table 3〉 Comparison of the number of arithmetic operations of example objects

그림 구분	노드의 수		8분체의 보이는 면의 수		변환 꼭지점의 수		식(1)에 의한 연산의 수				본 논문의 방법에 의한 연산의 수		
							노드 위치 코드의 직각좌표화		변환행렬 연산				
	깊이 1	깊이 2	깊이 1	깊이 2	깊이 1	깊이 2	덧셈	곱셈	덧셈	곱셈	덧셈	곱셈	나눗셈
그림2	5	7	8	12	23	33	99	198	673	896	405	147	165
그림4	2	22	1	33	4	94	282	564	1176	1568	704	330	294

〈표 4〉는 (그림 2)와 (그림 4)의 물체에 대한 기하학적 변환시 소요되는 산술연산의 수행시간을 비교한 것이다. 계산시간을 비교하기 위하여 PC-486DX2-50상에서 Borland C++로 구현하여 실험한 결과 본 논문의 계산방법이 효율적임을 알 수 있다.

〈표 4〉 기하학적 변환시 산술연산 수행시간 비교
〈Table 4〉 Comparison of the execution time

그림 구분	식 (1)의한 수행시간	본 논문의 방법에 의한 수행시간	비 고
그림 2	0.68초	0.33초	1,000회
그림 4	1.36초	0.59초	반복 계산

6. 결 론

본 논문에서는 8진트리로 표현된 3차원 영상의 기하학적 변환을 수행하는데 소요되는 시간을 크게 줄이기 위하여 8진트리의 각 노드를 기본벡터를 이용하여 직각 좌표로 변환시키는 효율적인 방법을 제안하고 이를 PC상에서 구현하여 그 효용성을 입증하였다.

8진트리로 표현된 물체에 대하여 기하학적 연산을 수행하기 위한 지금까지의 방법에서는 먼저 8진트리 노드의 위치코드를 직각좌표로 변환한 다음, 기하학적 변환행렬 연산을 모든 노드의 보이는 면에 속한 꼭지점에 대하여 적용해야 한다.

본 논문의 기본벡터를 이용한 공식을 이용하면 기하학적변환을 위한 행렬 연산을 8진트리의 루트노드에 해당하는 4개의 기본벡터에만 적용하면 되고, 공간상의 다른 꼭지점 3,5,6,7에 대한 기본벡터 U(3), U(5), U(6), U(7)은 기본벡터 U(1), U(2), U(4)의 선형조합으로 구하고

다른 모든 노드의 꼭지점의 좌표는 기본벡터를 이용하여 간단히 구할 수 있으므로 매우 효율적이다.

본 연구결과는 컴퓨터 그래픽스의 응용분야 즉, 애니메이션, CAD, 로보틱스에서 움직이는 물체의 표현, 의학 분야에서 단층 촬영 결과를 합성하여 3차원 형상을 구성하는데 이용하는 등 매우 다양한 분야에서 3차원 물체의 효율적인 표현 기법으로 활용할 수 있다.

참 고 문 헌

[1] C. Jackins and S. L. Tanimoto, "Octree and their use in presenting three-dimensional objects",Computer Graphics Image Processing 14, 1980, pp. 249-270.

[2] Homer H. Chen and Thomas S. Huang, "A survey of construction and manipulation of octrees", Computer Vision, Graphics, and Image Processing 43, 1988, pp. 409-431.

[3] K. Yamaguchi, T. L. Kunii, and K. Fujimura, "Octree-related data structure and algorithms", IEEE Computer Graphics and Applications, Jan. 1984, pp. 53-59.

[4] Hanan Samet and Robert E. Webber, "Hierarchical data structures and algorithms for computer graphics", IEEE Computer Graphics and Applications, May 1988, pp. 48-68.

[5] N. Ahuja and C. Nash, "Octree representation of moving objects", Computer Vision, Graphics, and Image Processing, vol. 26, 1984, pp. 207-216.

[6] M. E. C. Hull, J. H. Frazer, and R. J. Millar, "Octree-based modelling of computed-tomography images", June 1990, IEE Proceedings, vol. 137, No. 3, pp. 118-122.

[7] C. H. Chien and J. K. Aggarwal,"A volume/surface octree representation, in proceedings", Intl Conf. Pattern Recog., Montreal, Canada, July 1984, pp. 199-202.

[8] C. H. Chien and J. K. Aggarwal, "Identification of 3D objects from multiple silhouettes using quadtrees/octrees", Computer Vision, Graphics, and Image Processing 36, 1986, pp. 256-273.

[9] I. Gargantini and H. H. Atkinson, "Ray tracing an octree:Numerical Evaluation of the first intersection", vol. 12, No. 4, 1993, pp. 199-210.

[10] Donald Hearn and M. Pauline Baker, "Computer Graphics", Prentice-Hall, 1986.

허 영 남

1967년 공주사범대학 졸업(학사)
1982년 조선대학교 대학원 전자공학과(공학석사)
1992년~현재 조선대학교 대학원 전산통계학과 박사과정수료
1983년~86년 순천대학교 전자계산소장
1983년~현재 순천대학교 전자계산학과 교수
관심분야:영상처리, 병렬처리

박 승 진

1984년 조선대학교 전자공학과(학사)
1986년 조선대학교 대학원 전자공학과(공학석사)
1991년 조선대학교 대학원 전자공학과(공학박사)
1973년~91년 전남대학교 부속병원
1991년~현재 경상대학교 부속병원
관심분야:의공학, 영상처리

김 응 곤

1980년 조선대학교 공학사
1987년 한양대학교 공학석사
1992년 조선대학교 공학박사
1985년~87년 금성반도체(주)연구소 연구원
1987년~91년 국방과학연구소 선임연구원
1991년~93년 여수수산대학교 전임강사
1993년~현재 순천대학교 조교수
관심분야:컴퓨터 그래픽스, CAD