大韓機械學會論文集 第 18 卷 第 12 號, pp. 3159~3174, 1994

〈論 文〉

삼차원 유한요소의 자동생성(1)
—사면체 옥트리의 구성—

정융호* · 이건우**

(1994년 5월 6일 접수)

Automatic Generation of 3-D Finite Element Meshes : Part(I)
— Tetrahedron- Based Octree Encoding —

Y. H. Jung and K. Lee

Key Words : Tetrahedron(사면체), Octree(옥트리), Automatic Generation(자동생성), Finite Element Mesh(유한요소), Geometric Modeling(기하학적 모델링), Computer Aided Design(CAD)

Abstract

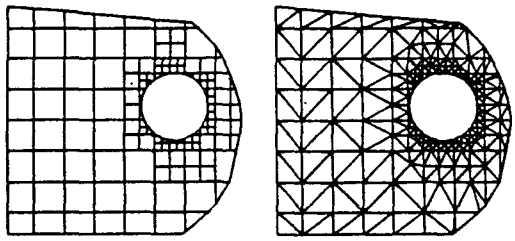
A simple octree encoding algorithm based on a tetrahedron root has been developed to be used for fully automatic generation of three dimensional finite element meshes. This algorithm starts octree decomposition from a tetrahedron root node instead of a hexahedron root node so that the terminal node has the same topology as the final tetrahedral mesh. As a result, the terminal octant can be used as a tetrahedral finite element without transforming its topology. In this part(I) of the thesis, an efficient algorithm for the tetrahedron-based octree is proposed. For this development, the following problems have been solved, : (1) an efficient data structure for storing the octree and finite elements, (2) an encoding scheme of a tetrahedral octree, (3) a neighbor finding technique for the tetrahedron-based octree.

I. 서 론

공학 전반에 걸쳐 역학적인 해석을 위해 유한요소법의 사용이 보편화됨에 따라 이의 입력 자료인 유한요소를 쉽게 생성하는 방법에 대한 관심이 높아지고 있다. 그 중 하나로 현재 CAD분야에서 많이 사용되고 있는 솔리드모델링 시스템(solid modeling system)으로부터 입체 형상(solid)이 결정되어 있을때, 이로부터 삼차원 유한요소를 자동으로 생성하는 기법들[1,2]이 많은 기대를 받고 있다. 이러한 기법들 중 삼차원 문제의 경우 Fig. 1에 소개한 회귀적 공간분할법(Recursive spatial decomposition approach[2])이 다음과 같은 이유로 각광을 받고 있다.

첫째 공학해석 문제에 있어서는 유한요소를 생성하는 과정에서 해석자의 직관이나 경험이 중요한 요소인데, 유한요소해법에 대한 경험이 부족한 설계자의 경우 가능한 한 유한요소를 생성하는 전과정이 자동으로 이루어지길 원한다. 이러한 경우 회귀적 공간분할법은 유한요소를 생성하는 전과정을 자동으로 할 수 있는 방법중의 하나이다. 따라서 문제의 경계형상이 매번 변화하는 형상 최적화(shape optimization) 문제, 또는 성형과정 시뮬레이션(forming simulation) 등을 자동화하는 문제에 있어서는 유한요소의 생성과정중에 사용자로부터의

*삼성항공(주) 항공우주연구소
**정회원, 서울대학교 기계설계학과



(a) Tree building (b) Element generation

Fig. 1 Recursive spatial decomposition approach

입력을 요구하지 않는 이와같은 방법이 유용한 특징을 가지고 있다.

둘째, 입체형상의 경계근처에서는 다소 조악한 형상의 유한요소가 생성될 수 있으나, 그 내부에서는 우수한 형상의 요소가 생성되는 장점을 가지고 있다. 경계근처에서 생성될 수도 있는 좋지 못한 형상의 요소들은 평활화(smoothing) 과정을 수행함으로써 개선될 수 있다.

세째, 생성되는 나무구조(tree structure)의 계층번호(level)를 국부적으로 증가시킴으로써 원하는 곳의 유한요소의 밀도를 조정하기가 용이하다. 따라서 오차지표(error indicator)에 근거한 요소 세분화과정(refinement)으로 문제의 특성에 대응하는 적응(adaptive)기능을 발휘할 수 있다.

네째, 요소생성에 소요되는 시간이 생성되는 요소의 개수에 근사적으로 비례[1]하므로 소요시간이 적은 편이다.

그런데 기존의 회귀적 공간분할법은 일반적으로 다음의 두 단계로 이루어져 있다. 먼저 3차원의 경우 주어진 입체형상을 포함하는 최소한의 정육면체를 생성하여 여덟개의 조각으로 분할한다. 이러한 조각을 팔분체(octant)라고 한다. 만약 하나의 팔분체가 입체형상의 내부 혹은 외부에 완전히 포함되면 분할을 중지하고, 그렇지 않으면 분할하는 과정을 회귀적으로 반복한다. 이러한 과정을 사용자가 지정하는 해상도까지 계속 수행하여 나무구조(tree structure)를 형성하게 된다. 두번째 단계에서는, 입체형상을 근사적으로 나타내는 육면체의 요소들을 사면체 유한요소로 변환시키는데, 이 과정은 여러가지 특수한 경우들을 고려한 복잡한 방법들에 의해 수행되고 있다. 또한 이 과정에서, Fig. 2에 보인 바와 같이 서로 모순되는 이웃하는 유한요소가 생성될 수 있는 문제도 있다.

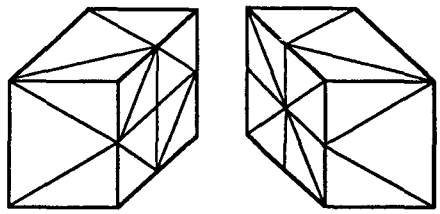


Fig. 2 Incompatibly mating mesh elements

회귀적 공간분할법을 사용하여 주어진 입체형상으로부터 전자동으로 유한요소를 생성하기 위해서는 앞에서 설명한 두번째 단계가 필수적인데, 본 연구에서는 이와같은 두번째 단계가 필요없는 효율적인 방법을 개발하였다. 즉, Shephard-Yerry (S-Y)의 방법[2~5]이나 Perucchio-Saxena-Kela (P-S-K)의 방법[2,6,7]에서는 나무구조의 형성과정을 정육면체에서 시작하는 반면, 본 연구에서는 사면체에서 회귀적 분할을 시작한다. 결과적으로 생성되는 나무구조의 단말요소(leaf element)는 사면체 유한요소와 같은 위상(topology)을 가지므로 입체형상의 경계를 포함하는 단말요소의 몇몇 꼭지점들만 입체형상의 경계로 이동하면 입체형상의 경계를 정확히 나타내는 적합한 유한요소가 생성된다. 본 논문의 (Ⅰ)편에서는, 앞에서 기술한 바와 같이 주어진 입체형상으로부터 사면체 옥트리의 구성방법에 대해 기술하고, (Ⅱ)편에서는 사면체 옥트리의 단말 요소들이 유한요소가 되는 과정을 기술하고자 한다.

2. 자료구조

삼차원 모델링 시스템으로 생성한 입체형상으로부터 회귀적 공간분할법을 이용하여 자동적으로 유한요소를 생성하기 위해서는, 생성된 입체형상의 경계를 저장하는 자료구조(data structure) 외에 옥트리와 생성되는 유한요소를 저장하기 위한 자료구조가 필요하다. 본 연구에서 물체의 경계는 일반적으로 솔리드모델링 시스템에서 많이 사용하는 NURBS(non-uniform-rational b-spline)[8]와 Winged Edge 자료구조[9]로 저장하였으며, 옥트리와 유한요소를 저장하기 위해서는 Fig. 3과 같이 'tree part'와 'queue part'로 구성된 자료구조를 설계하였다. 이 두 부분은 지시인자(pointer)에 의해 서로 상대부분을 쉽게 참조할 수 있는 긴밀한 관계로 되어 있다.

Fig. 3의 'queue part'는 옥트리의 각 팔분체가

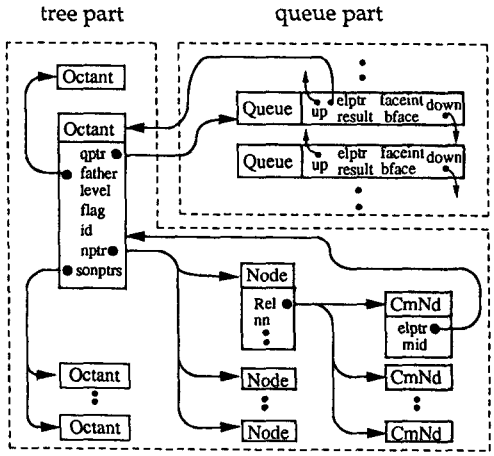


Fig. 3 Data structure for octree generation

입체형상에 대해 갖는 포함관계를 판별해 주기 위한 기하학적 정보를 저장하는 곳이다. 이것은 옥트리를 형성하는 과정에서 생성되는 모든 팔분체를 저장하기 위해 선입 선출(first in first out)의 구조를 가지는 연결리스트(linked list)로 되어 있다. 즉, 연결리스트의 한쪽 끝(top_Q)에서 팔분체를 하나 추출하여 입체형상에 대한 포함관계를 결정한다. 만약 입체형상에 부분적으로 겹쳐져 있으면 그 팔분체를 분할하여 연결리스트의 반대쪽 끝(bottom_Q)에 순차적으로 삽입하고, 그렇지 않으면 그 팔분체는 tree 부분에만 등록된다. 각 항목들에 대한 상세한 설명은 다음과 같다.

elptr : 나무구조의 팔분체로의 지시인자이다.

faceint[4] : 팔분체의 각 면이 원래 입체형상과 교차하는지의 여부에 관한 정보를 제공한다. 이 정보를 이용하여 팔분체의 입체형상에 대한 포함관계를 판정할 수 있다.

result[4] : 팔분체의 4개의 꼭지점이 원래 입체형상에 대해 갖는 포함관계를 나타낸다. 이것은 팔분체의 모든 면이 입체형상과 교차하지 않을때 '내포'인지 '외연'인지를 판단하는데 사용될 뿐만 아니라, 옥트리 구성 후 '부분'인 팔분체를 입체형상의 경계로 이동시킬 때 어느 꼭지점을 이동시킬 것인가에 대한 정보도 제공해 준다.

bface[4] : 팔분체의 각 면과 교차된 원래 입체형상의 면에 대한 지시인자를 저장함으로써, 그 팔분체의 자식들의 포함관계를 판정할 때 원래 입체형상의 면과의 교차여부에 대한 계산 범위를 축소시켜 계산효율을 높이는 데 기여한다.

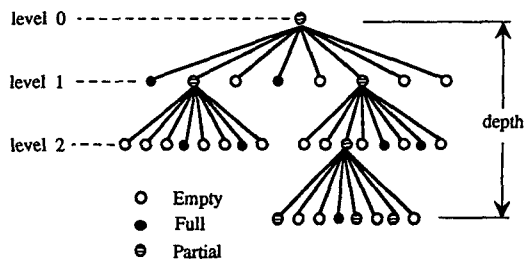


Fig. 4 The octree

up/down : 연결리스트로 만들기 위한 지시인자이다.

Fig. 3에서 'tree part'은 Fig. 4와 같은 옥트리를 저장하는 곳이다. 옥트리에서 하나의 요소는 그 상위 계층에 있는 '부모'로의 지시 인자와 그 하위 계층에 있는 '자식'들에 대한 지시 인자를 가짐으로써 계층적인 나무 구조를 이루게 된다. 나무 구조에서 뿌리의 요소를 뿌리절점(root node)이라고 하며 이는 주어진 입체형상을 완전히 포함하는 사면체에 해당한다. 'tree part'의 세부항목에 대한 설명은 다음과 같다.

qptr : 팔분체가 주어진 입체형상에 대해 갖는 기하학적인 포함관계를 나타내기 위한 정보를 저장하는 'queue part'에 대한 지시인자이다.

father : 팔분체의 그 부모에 대한 지시인자이다. 뿌리의 경우는 부모로의 지시인자로 가상의 위치에 해당하는 NIL을 가르키게 된다.

level : 나무 구조에 있어서 뿌리로 부터의 계층번호를 나타낸다. 이 수치는 팔분체의 실제 크기와 관련된다.

flag : 입체형상에 대한 그 팔분체의 내포(FULL), 외연(EMPTY), 부분(PARTIAL)등의 포함관계를 나타낸다.

id : 나무 구조에서의 팔분체의 고유번호로서 유한요소로 변환되었을 때 유한요소 번호가 된다.

nptr[10] : 팔분체의 10개의 절점(4개의 꼭지점과 6개의 모서리 중간점)에 대한 관련정보, 즉 Node들에 대한 지시인자이다. 유한요소해석을 수행할 때, node가 4개인 요소를 사용할 경우 4개의 지시인자만 사용할 수도 있다. 각 node에 대한 보다 상세한 내용은 곧이어 설명된다.

sonptrs[8] : 8개의 자식들에 대한 지시인자이다. 옥트리의 단말요소의 경우에는 자식들에 대한 지시인자로 NIL을 가르킨다.

한편 자동 유한요소 생성의 궁극적인 결과는 유한요소와 요소의 절점 즉, node와의 관계이다. 따라서 node에 대한 자료구조는 단순히 절점의 위치에 대한 정보뿐만 아니라 이웃하는 요소와의 유기적인 관계를 유지하여야 한다. 또한 본 논문의 (Ⅱ)편에서 설명될 경계이동 과정에서, 절점위치의 제한조건에 대한 정보도 필요하므로 아래와 같은 구조를 갖는다. 이의 각 세부항목들에 대한 설명은 다음과 같다.

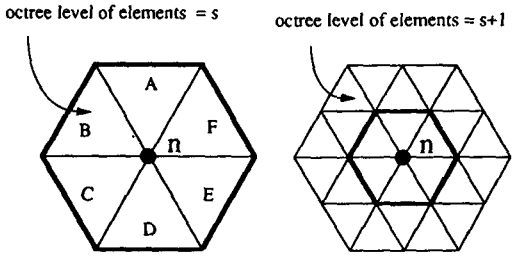
```
struct  node
{       int             nn;
        CmNd            *Rel;
        int             level;
        Point           *mvpt;
        struct node     *son;
        struct node     *father;
        int             flag;
        Vertex          *onVt;
        Edge            *onEd;
        Face            *onface;
}
```

nn : 그 절점의 고유번호를 나타낸다. 이는 팔분체를 유한요소로 변환시켰을 때 유한요소의 고유한 절점번호가 된다.

Rel : 그 점을 공유하는 모든 팔분체를 elptr로서 참조할 수 있는 CmNd에 대한 지시인자이다. 그리고 CmNd의 mid는 그 절점이 모서리 중간점의 여부를 나타낸다.

mvpt 사면체 옥트리의 생성과정에서 생성되는 각 팔분체의 꼭지점들의 좌표값은 저장하지 않고 생성과정으로부터 계산에 의해 구할 수 있었다. 그러나 물체의 경계주위의 사면체의 꼭지점을 실제 물체의 경계로 이동하는 '경계이동' 과정에서 유한요소의 꼭지점의 위치를 이동시켜야 할 필요가 있는데, 이때의 새로운 좌표값을 저장하는 곳에 대한 지시인자이다. '경계이동' 과정에 대한 설명은 본 논문의 (Ⅱ)편을 참조하기 바란다.

level : 하나의 절점을 공유하는 팔분체들의 계층번호를 나타낸다. 즉 각 절점에 대해 계층번호별로 그 절점을 공유하는 요소들과의 관계를 구성하기 위함이다. 2차원의 예를 들어, Fig. 5(a)에서 절점 n의 level값은 이를 공유하고 있는 요소 A, B, C, D, E가 생성된 옥트리의 계층번호 값 s를 가지며,



(a) Level of node n=s (b) Level of node n=s+1

Fig. 5 Level of node

동일한 점이지만 이를 공유하는 요소들이 한번 더 세분화된 (b)의 경우, 절점 n의 level값은 공유하는 요소들의 옥트리의 계층번호 값($s+1$)을 가진다.

father : 상위계층의 절점에 대한 지시인자를 저장하는 곳이다.

son : 하위계층의 절점에 대한 지시인자를 저장하는 곳이다.

flag : 만약 꼭지점이 경계이동된 점일 때, 그 점이 이동된 곳의 위치를 나타낸다. 프로그램 내부에서는 Node의 위치가 입체형상의 꼭지점으로 이동되었으면 3, 모서리로 이동되었으면 2, 면으로 이동되었으면 1, 내부의 점이면 0의 값을 갖는다.

onVt : 경계이동 단계에서 입체형상의 꼭지점으로 이동되었을 때, 입체 형상의 실제 꼭지점에 관한 정보를 저장하는 곳에 대한 지시인자이다.

onEd : 경계이동 단계에서 입체형상의 모서리로 이동되었을 때, 입체형상의 실제 모서리에 관한 정보를 저장하는 곳에 대한 지시인자이다. 모서리 상의 점일 경우 요소 평활화 과정에서 입체형상의 모서리 상에서만 절점의 위치가 변화하도록 하는 기하학적 구속조건을 제공한다.

onFc : onEd의 경우와 마찬가지로 경계이동 단계에서 입체형상의 면으로 이동되었을 때, 입체형상의 실제 면에 관한 정보를 저장하는 곳에 대한 지시인자이다. 역시 요소 평활화 과정에서 입체형상의 면상에서 절점의 위치가 변화하도록 하는 기하학적 구속조건을 제공한다.

3. 사면체의 옥트리

3.1 사면체의 분할

본 절에서는 사면체 옥트리를 형성하기 위해 사

면체가 8개로 분할되는 과정을 설명하고자 한다. 사면체 옥트리를 구성하기 위해서는 사면체를 분할하는 방법이 회귀적으로 적용될 수 있어야 하므로, 분할된 팔분체도 모체의 위상(topology)과 같은 사면체이어야 한다. 또한 생성되는 사면체들은 가능한 한 같은 크기를 유지하는 것이 좋다. 따라서 사면체의 4개의 꼭지점과 각 모서리의 이등분 점들로써 분할한다. 분할하는 방법은 Fig. 6에서와 같이, 각 꼭지점을 포함하는 4개의 사면체를 절단해 내고 남은 팔면체를 다시 4개의 등분으로 분할한다.

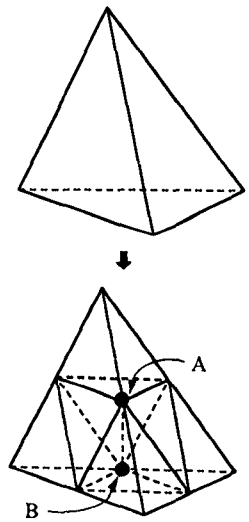


Fig. 6 Split of tetrahedron

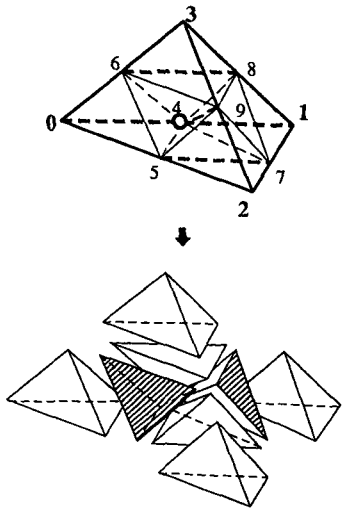


Fig. 7 Splitting of an irregular tetrahedron

그런데 사면체를 분할함에 있어서 생성된 사면체들은 두가지로 대별된다. 본 논문에서는 사면체의 6개의 변이 모두 같으면 '정사면체'로, 그렇지 않으면 '비정사면체'로 정의한다. 하나의 정사면체를 팔등분하면 Fig. 6에서와 같이 그 정사면체의 꼭지점을 포함하는 분할된 사면체들은 정사면체인 반면, 나머지 4개의 사면체들은 비정사면체이다. 왜냐하면, 각 비정사면체의 한 변(Fig. 6에서 꼭지점 A와 B를 연결한 선분)이 나머지 5개 변보다 $2\frac{1}{2}$배 길기 때문이다. 그러나 비정사면체를 같은 방법으로 분할하였을 때, 2개는 정사면체이고 나머지 6개는 비정사면체로 된다. 즉, Fig. 7에서와 같이 꼭지점 0번과 1번을 잇는 선분이 꼭지점 1과 2, 2와 0, 3과 0, 3과 1, 그리고 3과 2를 잇는 선분들보다 $2\frac{1}{2}$배 긴 비정사면체를 고려하자. 이 비정사면체가 분할되었을 때, 꼭지점 0과 4, 4와 1, 5와 7, 6과 8을 잇는 4개의 선분은 분할된 각 사면체의 변들보다 $2\frac{1}{2}$배 길다. 따라서 빗금을 치지 않은 분할된 사면체들은 비정사면체가 되고, 나머지 빗금친 2개는 정사면체가 된다.

결국 정육면체로부터 옥트리를 형성하면 옥트리 전체를 통하여 정육면체의 팔분체들만 생성되는데 반해, 정사면체로부터 옥트리를 형성하면 옥트리 전체를 통하여 두 종류의 사면체가 생성된다. 그러나 생성되는 팔분체는 모두 사면체의 위상을 가지고 있으므로 사면체를 분할하는 방법이 회귀성을 가지고 있음이 입증된다.

3.2 사면체인 팔분체의 번호체계

본 연구에서 개발할 사면체 옥트리는 육면체의 옥트리보다 여러가지 장점들을 가진다. 그러나 사면체의 옥트리가 육면체의 경우보다 다음과 같은 이유에서 복잡한 것으로 여겨진다.

첫째, 육면체의 경우처럼 각 팔분체의 꼭지점들의 좌표를 간단하게 구하기가 곤란하다. 즉, 지금까지 대부분의 사람들은 x-y-z의 직교좌표계에 익숙해 있으므로, 직교좌표계에 평행한 방향으로 정육면체를 생성하였을 때 분할된 팔분체들도 직교좌표계에 평행한 정육면체가 된다. 반면에 사면체인 경우, 그 형상이 직교좌표계에 평행하지 않기 때문에, 꼭지점들의 좌표계산이 육면체의 경우처럼 용이하지 않다.

둘째, 육면체인 팔분체의 경우 모체에 대한 자신의 위치가 쉽게 정의될 수 있다. 따라서 이웃하는

팔분체들과의 위치관계도 쉽게 정의될 수 있다. 반면 사면체의 경우 생성되는 팔분체는 모체에 대한 자신의 위치를 육면체의 경우처럼 직교좌표계의 방향으로 정의하기가 곤란하고 따라서 이웃하는 팔분체와의 공간적인 상관관계를 찾기가 곤란하다.

세째, 육면체의 경우 생성되는 팔분체는 모체의 방향성을 그대로 가지고 있을 뿐만 아니라 모체의 형상도 그대로 유지하고 있다. 따라서 직교좌표계에 평행하고 서로 수직한 평면들로 분할하는 방법을 회귀적으로 적용하기가 쉽다. 반면, 사면체의 경우 Figs. 6, 7에서 보인 것과 같이 모체로부터 생성되는 팔분체 중에는 모체의 방향과 형상을 유지하지 않는 팔분체도 발생하게 된다. 따라서 이를 회귀적으로 분할하는 방법이 육면체의 경우처럼 용이해 보이지 않는다.

이상의 문제점들은 팔분체의 각 꼭지점, 모서리 면에 다음과 같은 조건들을 만족하도록 고유한 번호들을 체계적으로 부여함으로써 해결될 수 있다.

첫째, 하나의 팔분체의 부모에 대한 관계를 그 팔분체가 부모로부터 생성되는 순서(이하 '자식번호')로 정의하였을 때, 자식번호만 알면 그 팔분체의 꼭지점들의 번호, 즉 위치는 유일하게 규정되어야 한다. 일단 팔분체의 꼭지점들의 번호가 결정되면, 팔분체의 면과 모서리의 번호, 그리고 꼭지점의 좌표가 결정될 수 있다. 따라서 이 조건에 의해 팔분체를 분할하는 하나의 알고리즘을 회귀적으로 사용하여 옥트리를 구성할 수 있게 된다.

둘째, 한 면을 공유하고 동일한 계층수의 이웃하는 팔분체는, 각각의 선조가 달라도 대응하는 상대쪽의 자식번호는 유일하게 정해져야 한다. 트리구조에서 특정 팔분체의 이웃 팔분체를 찾는 알고리즘은 본 연구의 효율성에 중요한 역할을 수행하는데, 앞의 조건에 의해 특정 팔분체의 면에 이웃하는 팔분체를 찾을 수 있다. 이웃 팔분체를 찾는 방법은 다음 절에 그 예와 함께 설명하겠다.

이상의 조건을 만족하는 사면체의 팔분체를 정의하는 방법을 본 과제에서는 Fig. 8과 같이 정의하였다. Fig. 8에서 일반 숫자 0, 1, 2, 3은 팔분체의 4개의 꼭지점의 고유번호를 정의하고 4부터 9까지의 숫자는 6개의 모서리 중간점의 고유번호를 정의한다. 그리고 사각형으로 둘러싸인 숫자는 분할된 각 팔분체의 공통 부모에 대한 자식번호를 부여하는데, 이는 Table 1에 정의된 부모 팔분체의 꼭지점 번호에 의해 결정된다. 각 팔분체는 '기준점(reference vertex),' '기준면(base plane),' 그리고 '천정점(apex)'에 의해 부모에 대한 위치관계가 결정된다. Table 1에서 '기준점'은 *로 표시하였다.

Fig. 8의 부모 팔분체의 경우, 꼭지점 번호 0, 1, 2로 정의되는 면이 그 팔분체의 '기준면'이 되며 '기준면'의 꼭지점 번호의 순서는 그 면을 바깥쪽에서 보았을 때 반시계방향 순서가 되게 한다. 그리고 꼭지점 번호 0번 점이 그 팔분체의 '기준점'이 된다. 이 부모 팔분체로부터 생성되는 여덟개의 자식들도 Table 1에 정의된 '기준점,' '기준면,' 그리고 '천정점'에 의해 일의적으로 정의된다. 즉 Table 1의 행번호는 자식번호를 나타내고, '기준면'의 첫째 열의 숫자는 부모 팔분체의 꼭지점 번호인데, 그 점이 각 행에 자식의 0번 꼭지점이 됨

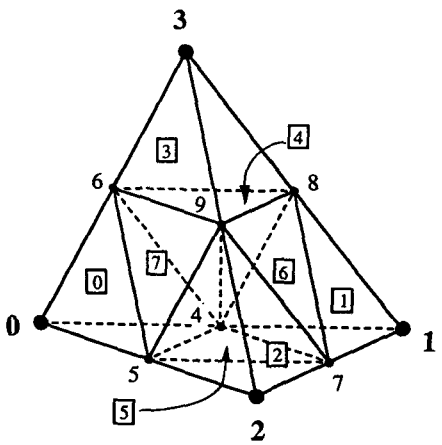


Fig. 8 Encoding scheme of the tetrahedral octant

Table 1 Definition of son octants

Vert. no. / Son type	Base plane			Apex
	0	1	2	3
[0]	6*	5	4	0
[1]	7*	8	4	1
[2]	5*	7	2	9
[3]	8*	6	3	9
[4]	8*	6	9	4
[5]	5*	7	9	4
[6]	7*	8	9	4
[7]	6*	5	9	4

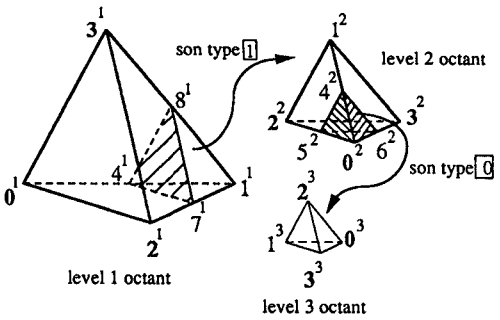


Fig. 9 Recursive numbering of octants

을 보여준다. '기준면'의 두번째 열의 각 숫자는 각 행에 해당하는 자식의 1번 꼭지점이 된다. 이러한 규칙에 의해 팔분체가 생성될 때마다 그 4개의 꼭지점의 번호는 0, 1, 2, 3으로 설정된다.

본 연구에서 고안된 사면체의 팔분체를 정의하는 방법이 앞의 첫번째 조건을 만족함을 보이기 위하여, 꼭지점 번호 0^1-1^1-2^1-3^1(이하 꼭지점번호의 윗첨자는 그 팔분체의 계층수를 나타냄)으로 정의되는 계층수 1의 팔분체를 Fig. 9에 나타내었다. 계층수 1의 팔분체로부터 생성되는 계층수가 2이고 자식번호가 1인 팔분체는 꼭지점 번호 7^1-8^1-4^1-1^1로 정의된다. 그다음 계층수 1인 팔분체의 7^1번 꼭지점은 계층수 2이고 자식번호가 1인 팔분체의 0^2번 꼭지점이 된다. 같은 방법으로 하면 Table 1에 정의된 바와 같이 8^1은 1^2, 4^1은 2^2, 1^1은 3^2가 된다.

앞에서 생긴 계층수 2의 팔분체가 계층수 3의 팔분체로 다시 분할되었을 때, 계층수 3이고 자식번호 0인 팔분체는 꼭지점 번호 6^2-5^2-4^2-0^2로 정의된다. 마찬가지로 계층수 2인 팔분체의 6^2번 꼭지점은 계층수 3이고 자식번호 0인 팔분체의 0^3번 꼭지점이되고, 5^2은 1^3으로, 4^2는 2^3으로, 0^2는 3^3으로 된다.

하나의 팔분체는 그 부모로부터 생성될 때마다 4개의 꼭지점의 번호가 0, 1, 2, 3으로 설정되므로 이러한 분할은 회귀적으로 제한없이 사용될 수 있다. 이와같은 방법으로 하나의 팔분체가 뿌리 팔분체로부터 생성되는 과정을 추적하면 그 꼭지점의 좌표도 뿌리 사면체의 꼭지점의 좌표로부터 계산할 수 있다.

팔분체의 기준 평면 이외의 면에 대한 정의는 Table 2에 정의하였고 6개의 모서리에 대한 고유번호는 Table 3에 규정되어 있다. 여기서 면 번호와 모서리 번호를 정의하는 꼭지점 번호는 팔분체 자신의 것이지만, 기준점, 기준면, 천정점 등을 정의하는 꼭지점 번호는 그 부모의 것이다.

Table 2 Definition of face number

Face id.	Vert. no.		
0	0	1	2
1	0	3	1
2	1	3	2
3	0	2	3

Table 3 Definition of edge number

Edge id.	Vert. no.	
0	0	1
1	1	2
2	2	0
3	0	3
4	1	3
5	2	3

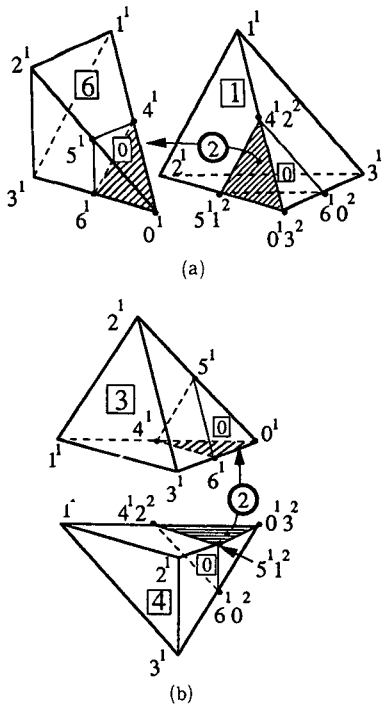


Fig. 10 Consistent relationship between neighbors

이제 앞에서 기술한 사면체에 대한 번호 체계가 두번째 조건을 만족함을 Fig. 10의 예로 보이겠다. Fig. 10에서는 자식번호가 1, 6, 3, 4인 4개의 팔분체를 고려하겠다. Fig. 9에서와 같이 꼭지점 번호의 첨자는 그 팔분체의 계층수를 나타낸다. Fig. 10(a)에서, 계층수가 1이고 자식번호가 1인 팔분체에서 생성된 계층수가 2이고 자식번호가 0인 팔분체의 2번 면을 공유하는 팔분체는 계층수가 1이고 자식번호가 6인 부모로부터 생성된 계층수 2이고 자식번호가 0인 팔분체임을 보여준다. 마찬가지로 Fig. 10(b)에서는, 계층수가 1이고 자식번호가 4인 팔분체에서 생성된 계층수가 2이고 자식번호가 0인 팔분체의 2번 면을 공유하는 팔분체의 자식번호는, 앞의 Fig. 10(a)의 예와 동일한 숫자인 "0번"의 계층수 2인 팔분체임을 보여준다. 따라서 어떤 팔분체의 한 면을 공유하는 동일 계층수의 팔분체의 자식번호는 각 팔분체의 부모의 자식번호를 고려하지 않더라도 유일하게 결정될 수 있음을 보여준다.

3.3 사면체의 이웃찾기

앞에서 언급한 바와 같이, 팔분체의 입체형상에 대한 포함관계의 판정은 이웃하는 팔분체의 관련정보를 이용할 수 있으면 아주 효율적으로 수행될 수 있다. 예로서, Fig. 11에서 계층수가 1인 팔분체 C의 1번 면의 이웃은 같은 계층수의 팔분체 D임이 알려졌다고 가정하자. 팔분체 C의 입체형상에 대한 포함관계를 판정할 때, 팔분체의 각 면이 입체형상의 경계면과 교차하는지를 계산하여야 한다.

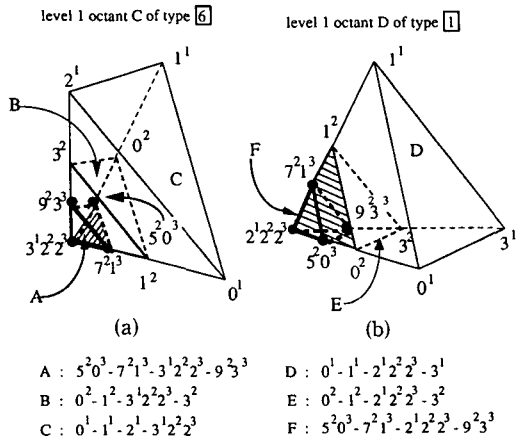


Fig. 11 Example of neighbor finding

만약 팔분체 D의 0번 면이 입체형상의 경계면과 교차하는지에 대한 계산이 먼저 되어있으면, 팔분체 C의 1번 면이 입체형상의 경계면과 교차하는지에 대한 계산을 수행할 필요없이 팔분체 D의 0번 면의 정보를 그대로 사용할 수 있다. 또한, 팔분체 D의 0^1, 1^1, $2^1 2^2 2^3$번 꼭지점의 입체형상에 대한 내·외 판정이 먼저 계산되어 있으면, 팔분체 C의 0^1, 1^1, $3^1 2^2 2^3$번 꼭지점의 내·외 판정은 계산할 필요없이 팔분체 D의 0^1, 1^1, $2^1 2^2 2^3$번 꼭지점의 정보를 그대로 사용할 수 있다.

따라서 주어진 팔분체의 이웃하는 팔분체를 찾는 방법은 옥트리의 효율적 형성을 위해 절대적으로 필요하다. 이를 위해 본 연구에서는, 사각형의 quadtree에 대해 제안된 Hanan Samet[10]의 방법을 이용하여 사면체의 옥트리에 적합한 새로운 이웃찾기 알고리즘을 개발하였다. 본 알고리즘으로 주어진 팔분체의 한 면을 공유하는 이웃 팔분체를 찾을 수 있다. 즉, Table 4와 5를 이용하면 특정한 팔분체의 자식번호(Table의 윗 행)에 대하여 그 팔분체의 면(Table의 왼쪽 열)을 공유하는 팔분체의 자식번호를 찾을 수 있다. Tables 4, 5의 사용법을 다음의 예를 이용하여 설명하겠다.

Fig. 11(b)의 자식번호가 2이고 계층수가 2인 팔분체 E의 0번 면(0^1-1^1-$2^1 2^2 2^3$)을 공유하는 동일 계

Table 4 Table of REFLECT2

Face id. \ Son type	0	1	2	3	4	5	6	7
0	7	6	3	2	3	2	7	6
1	1	0	5	4	5	4	1	0
2	0	1	2	3	7	6	4	5
3	0	1	2	3	6	7	5	4

Table 5 Table of REFLECT3

Face id. \ Son type	0	1	2	3	4	5	6	7
0	7	6	2	3	3	2	7	6
1	1	0	5	4	4	5	1	0
2	1	0	2	3	7	6	4	5
3	1	0	2	3	6	7	5	4

층수의 이웃 팔분체를 찾아보자. Fig. 11(b)에서 볼 수 있듯이 팔분체 E의 0번 면은 그 부모 팔분체 D의 0번 면(0^1-1^1-2^2-2^3)이기도 하다. 그런데 팔분체 D의 0번 면은 더이상 D의 부모 팔분체의 바깥면이 아니다. 따라서 Fig. 12에서 알 수 있듯이, 찾고자 하는 이웃 팔분체는 팔분체 E와 동일한 조부를 갖고 있다. 이경우 Table 4의 REFLECT2를 이용하여 이웃 팔분체를 구할 수 있다. 즉, 팔분체는 E의 자식번호 2와 그 면번호 0으로써 Table 4에서 3이라는 숫자를 얻는다. 이 숫자는 찾고자 하는 동일 계층수의 이웃 팔분체의 자식번호가 3번임을 말한다. 그 다음, 팔분체 E의 부모 D(자식번호 1)의 0번 면을 공유하는 동일 계층수의 이웃은 다시 Table 4를 이용하여 자식번호가 6번임을 알 수 있다. 따라서 뿌리 팔분체로부터 각 계층수의 찾아진 이웃을 따라가면, 즉 Fig. 12에서 그 경로가 ROOT → 자식번호 6 → 자식번호 3이므로 찾고자 하는 동일 계층수의 이웃 팔분체는 B임을 알 수 있다.

두번째 예는, Fig. 11(a)의 계층수가 3이고 자식번호가 2인 팔분체 A의 0면($5^2$$0^3$-$7^2$$1^3$-$3^1$$2^2$$2^3$)을 공유하는 이웃을 찾는 문제이다. 이 예에서도 마찬가지 방법으로 이웃 팔분체가 찾아질 수 있다. 단, 차이점은 찾고자 하는 이웃이 주어진 팔분체와 동일한 증조부를 갖는다는 것이다. 이 경우, Table 5의 REFLECT3를 이용하여 자식번호가 2인 팔분체 A의 0번 면을 공유하는 팔분체는 자식번호가 2번임을 알 수 있다. 그 다음 REFLECT2를 이용하여, A의 부모인 팔분체 B의 이웃이 자식번호가 2번(팔분체 E)임을, 다시 B의 부모인 팔분체 C의 이웃이 자식번호가 1번(팔분체 D)임을 알아낸다. 따라서 마찬가지로 Fig. 12에서 그 경로는 ROOT → 자식번호 1 → 자식번호 2 → 자식번호 2이므로,

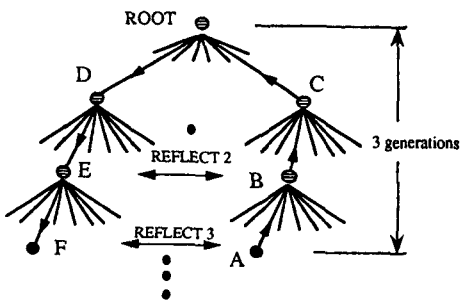


Fig. 12 Octree of the neighbor finding example

찾고자 하는 동일 계층의 이웃 팔분체가 F임을 알 수 있다. 이상의 예에서 알 수 있는 바와 같이, 이웃하는 팔분체와 동일한 조상이 3세대 이상 차이나면 REFLECT3를 사용한다.

4. 사면체 옥트리의 형성

4.1 옥트리의 구성

사면체 옥트리를 형성하는 기본적인 개념은 회귀적 공간분할법을 따른다. 그러나 기존의 방법이 육면체의 뿌리 절점에서 시작하는 것과 달리, 본 연구에서는 "사면체"의 뿌리 절점에서 시작한다. 사면체의 분할은 앞에서 정의된 좌표 시스템을 사용하여 회귀적으로 반복 적용하면 옥트리를 구성할 수 있다.

사면체로써 옥트리를 형성하는 과정을 간략히 기술하면 다음과 같다. 제일 먼저 나무구조의 뿌리 절점을 초기화한다. 즉, 뿌리 절점의 'father'를 NIL로 두고, 그 'flag'를 무조건 '부분'(실제 프로그램에서는 '1'로 정의됨)으로 설정한다. 이는 뿌리 사면체에 대해 분할 작업이 시작될 수 있도록 하기 위함이다. 그 다음, 'PUSH_Q'라는 함수로써 뿌리 절점을 앞에서 기술한 자료구조의 'queue'의 윗 끝에 넣는다. 이 단계에서 'result', 'faceint', 'bface' 등 그 팔분체의 입체형상에 대한 포함관계에 관련한 정보가 채워지게 된다. 그 다음 'PULL_Q'함수에 의해 'queue'의 윗끝으로 부터 팔분체를 하나 꺼집어 내어 주어진 입체형상에 대한 포함 관계를 결정한다. 만약 그 팔분체가 '부분'이면서 그 팔분체의 계층수가 사용자가 미리 지정한 숫자(실제 알고리즘에서는 'depth'로 정의됨)보다 작으면, 그 팔분체는 8개의 자식들로 분할되고 생성된 자식들은 'tree'에 등록됨과 아울러 기하학적 계산이 수행되면서 PUSH_Q함수에 의해 'queue'의 아래 끝에 삽입된다. 만약 해당되는 팔분체의 계층수가 미리 지정한 숫자와 같으면 그 팔분체는 더이상 분할되지 않는다. 이러한 과정을 'queue'에 팔분체가 하나도 남지 않을 때까지 계속한다. 이 과정을 C언어로 구현한 것을 Fig. 13에 보였다.

각 팔분체가 입체형상에 대해 갖는 포함 관계의 결정은 다음과 같이 이루어진다. 우선 'queue'의 faceint를 조사하여 팔분체의 4개의 면 중 하나라도 입체형상과 교차되어 있으면 그 팔분체는 '부분'으로 분류된다. 나머지 팔분체는 꼭지점의 입체형

```
MAKE_TREE ( depth )
int     depth ;  /* the tree level */
{
        Tree        *p ;
        Queue       *q ;

        initialize ROOT ;
        PUSH_Q ( ROOT ) ;
        q = PULL_Q ( ) ;
        p = q->elptr ;
        while ( p != NIL ) {
              p->flag = classify ( q ) ;
              if ( p->flag==1 && p->level < depth ) {
                    for ( i=0 ; i<8 ; i++ ) {
                          p->sonptrs[i]->father = p ;
                          p->sonptrs[i]->level = p->level + 1 ;
                          p->sonptrs[i]->id = maxtree++ ;
                          p->sonptrs[i]->qptr = NIL ;
                          for ( j=0 ; j<10 ; j++ )
                              p->sonptrs[i]->nptr[j] = NIL ;
                          for ( j=0 ; j<8 ; j++ )
                              p->sonptrs[i]->sonptrs[j] = NIL ;
                    }
                    for ( i=0 ; i<8 ; i++ )
                          PUSH_Q ( p->sonptrs[i] ) ;
              }
              q = PULL_Q ( ) ;
              p = q->elptr ;
        }
}
```

Fig. 13 Octree generation algorithm

상에 대한 위치관계에 따라 '내포' 또는 '외연'으로 분류되게 된다. 즉, 팔분체의 4개의 꼭지점들이 모두 입체형상의 내부에 있거나, 일부는 내부에 있고 나머지는 입체형상의 표면에 있는 경우는 '내포'로

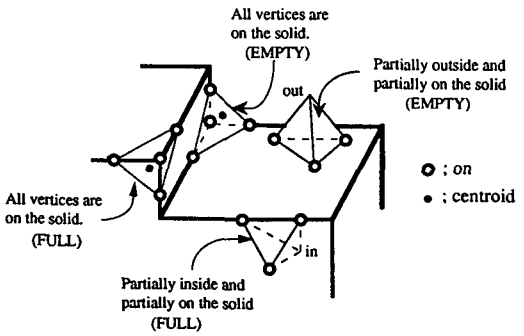


Fig. 14 FULL and EMPTY octants not intersecting the solid

판정된다. 팔분체의 4개의 꼭지점들이 모두 입체형상의 외부에 있거나, 일부는 외부에 있고 나머지는 입체형상의 표면에 있는 경우는 '외연'으로 판정된다. 만약 팔분체의 4개의 꼭지점들이 모두 입체형상의 표면에 있을 경우 그 팔분체의 무게중심점의 위치를 조사하여, 내부에 있으면 '내포'로, 외부에 있으면 '외연'으로 분류한다. 각 경우에 대한 예를 Fig. 14에 보였다. 이상의 과정 중 몇몇 중요한 단계들은 다음의 각 소절에서 설명된다.

4.2 사면체 꼭지점의 좌표계산

팔분체의 꼭지점의 좌표값은 입체형상에 대한 포함 관계를 판별하기 위한 기하학적 계산 등 사면체 유한요소의 생성과정중 빈번히 필요하므로 그 값을 간단히 구할 수 있어야 한다.

팔분체가 육면체인 경우 그 꼭지점의 좌표값은

나무구조의 뿌리 육면체의 좌표로부터 체계적으로 구할 수 있으며, 따라서 그 값을 저장할 필요없이 필요할 때마다 계산할 수 있다. 나무구조를 이용하여 유한요소를 생성하는 기법이 가지고 있는 장점 중의 하나는, 나무구조에서 특정 절점에 해당하는 요소의 꼭지점을 저장하지 않고 계산에 의해 구할 수 있다는 것이다.

사면체인 팔분체의 경우도 마찬가지로 그 꼭지점의 좌표값을 뿌리 사면체의 좌표로부터 체계적으로 구할 수 있다. 이것은 앞 절에서 결정한 일관성있는 번호 체계에 의해 가능해 진다. 즉 옥트리는 정사면체인 뿌리 사면체를 회귀적으로 분할하므로 부모 팔분체에 대한 자식의 고유번호가 정해지면 부모 팔분체의 꼭지점들의 좌표로부터 그 자식의 꼭지점의 좌표를 간단히 구할 수 있다. 이러한 개념을 확장하면, 임의의 팔분체의 꼭지점의 좌표도 그 팔분체에 해당하는 옥트리구조상의 절점으로 부터 뿌리 절점까지의 생성 경로를 따라가면서 앞의 계산 과정을 회귀적으로 반복하여 구할 수 있다. 여기서 뿌리 사면체의 꼭지점의 좌표는 나무구조를 형성하기 시작할 때 입체형상을 포함하도록 미리 계산되어진다.

예로써, Fig. 15의 자식번호가 7이고 계층수가 2인 사면체의 꼭지점의 좌표를 계산하는 과정을 설명하겠다. 자식번호가 7인 팔분체의 0^2번 꼭지점의 경우 제 3장의 Table 1에서 보는 바와 같이 계층수가 1인 부모의 6^1번 꼭지점이다. 따라서 0^2번 꼭지점의 좌표값은 그 부모의 꼭지점 0^1과 3^1의 중간점의 좌표이다. 나머지 3개의 꼭지점의 좌표값도 마찬가지로 다음과 같이 구할 수 있다.

$$V_{70}=\frac{1}{2}(V_0+V_3) \quad V_{72}=\frac{1}{2}(V_2+V_3)$$

$$V_{71}=\frac{1}{2}(V_0+V_2) \quad V_{73}=\frac{1}{2}(V_0+V_1)$$

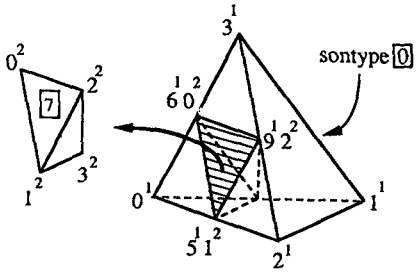


Fig. 15 Calculation of vertex coordinates

여기서,

V_i : 부모 팔분체의 i번째 꼭지점 좌표값

V_{jk} : 자식번호가 j인 팔분체의 k번째 꼭지점의 좌표값

4.3 면/면 교차 판정

경계표현방식(B-rep[11])으로 표현된 입체형상에 대해 특정 팔분체의 정확한 포함관계를 판정하기 위해서는 앞에서 설명한 바와 같이 팔분체의 면과 주어진 입체형상의 경계면과의 교차 여부를 알아야 한다. 그런데 이러한 계산은 그 양이 방대하고 많은 시간을 요하므로 이를 개선하기 위해 다음과 같은 사항들이 고려되었다.

첫째, 팔분체의 각 면이 입체형상과 교차하는지의 여부에 대한 계산이 필요할때, 그 이웃이나 부모로부터 관련 정보를 얻을 수 있으면 이를 이용함으로써 중복 계산을 피한다. 이를 위해 사면체의 옥트리에 대한 효율적인 이웃찾기 알고리즘(neighbor finding algorithm)이 개발되었다.

둘째, 사면체의 특정 면이 입체형상의 경계면과 교차하는지의 여부에 대한 계산을 줄이기 위해, 입체형상의 경계면에 대해 삼각형의 집합인 '절면모델(faceted model)'을 사용함으로써 사면체의 삼각형 평면과 절면모델의 삼각형 평면의 교차문제로 단순화 하였다. 이를 위해 본 연구에서는 Choi[12]의 알고리즘을 이용하여 삼각형의 절면 모델을 자동으로 생성하였다. 따라서 임의의 곡면으로 된 입체형상도 다룰 수 있게 되었다. 절면 모델의 생성은, 입체형상의 표면을 표현하는 NURBS 곡면의 조정점(control point)이 실제 곡면으로 투사된 점을 입력으로 사용하였다. 입체형상의 표면 근처의 꼭지점들은 본 논문의 (Ⅱ)편에서 설명될 '경계이동'단계에서 절면 상이 아닌 입체형상의 실제 표면으로 이동되게 하였으므로, 절면모델을 이용하는

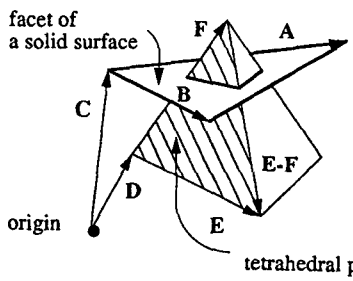


Fig. 16 Intersection between facet and octant face

것이 생성되는 유한요소의 정확도에 영향을 미치지는 않는다.

Fig. 16에 절면의 삼각형과 팔분체의 삼각형이 교차하는 경우를 나타내었다. 팔분체의 각 면에 대하여, 입체형상의 절면과 교차하는 경우는 다음의 다섯가지 중의 하나이다.

(i) 벡터 E선분이 절면과 교차하는 경우
(ii) 벡터 F선분이 절면과 교차하는 경우
(iii) 벡터 (E-F)선분이 절면과 교차하는 경우
(iv) 벡터 A선분이 팔분체의 면과 교차하는 경우
(v) 벡터 B선분이 팔분체의 면과 교차하는 경우

(i)의 경우는 다음과 같이 수식화할 수 있으며

$$C+Au+B\nu=D+Et \tag{1}$$

이를 행렬식으로 나타내면 다음 식(4.2)와 같다.

$$\begin{bmatrix} u \\ \nu \\ t \end{bmatrix} = \begin{bmatrix} A_x & B_x & -E_x \\ A_y & B_y & -E_y \\ A_z & B_z & -E_z \end{bmatrix}^{-1} \begin{bmatrix} D_x & -C_x \\ D_y & -C_y \\ D_z & -C_z \end{bmatrix} \tag{2}$$

여기서, u, ν는 절면의 매개변수이고 t는 벡터 E의 매개변수이다.

만약($0 \le u \le 1$ and $0 \le \nu \le 1$ and $0 \le u+\nu \le 1$ and $0 \le t \le 1$)이면, (i)의 경우가 발생하였다고 판단한다. 나머지의 경우도 마찬가지로 판정할 수 있다.

4.4 꼭지점의 내/외 판정

팔분체의 꼭지점이 입체형상의 내부 혹은 외부에 존재하는가를 판정하는 것은 나무구조의 형성과정 중 그 팔분체의 입체형상에 대한 포함 관계를 결정하는 데 필요하다. 이를 위해 본 연구에서는 Fig. 17에서와 같이 주어진 점에서 시작하여 임의의 방향으로 직선을 생성한다. 그 직선의 길이는, 팔분체의 꼭지점으로부터 주어진 입체형상을 충분히 통과할 수 있도록 하기 위해, 뿌리 사면체의 한변의 길이로 택하였다. 그 다음, 그 직선과 입체형상의 모든 면과의 교점의 개수를 구한다. 만약 그 개수가 짝수이면 '입체형상의 외부에', 홀수이면 '입체형상의 내부에 존재한다'라고 판단한다. 직선과 입체형상의 면과의 교차여부에 대한 계산도 앞의 소절에서 설명한 바와 같이 절면의 집합과 직선의 교차문제로 생각한다. 만약 그 직선이 절면에 접하거나 포함되는 경우, 동일한 점으로부터 다른 방향의 직선을 생성하여 위의 계산과정을 수행한다.

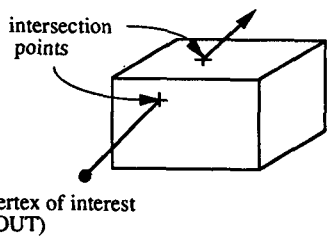


Fig. 17 IN/OUT test of a vertex

5. 적용 예

본 논문에서 목적하는 3차원 입체형상의 물체를 대상으로 유한요소를 자동으로 생성하기 위해, 먼

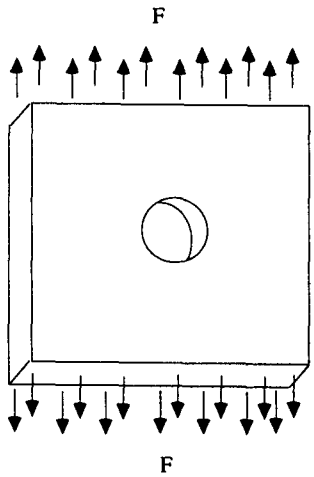


Fig. 18 Plate with hole in tension

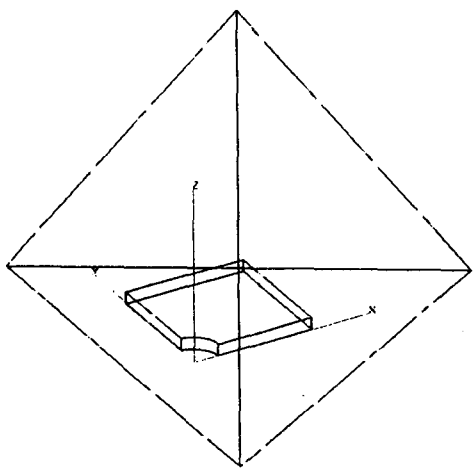


Fig. 19 Root tetrahedron of analysis model

저 논문의 (I)편에서는 실제로 3차원 입체 형상으로 부터 사면체 옥트리가 구성되는 예를 보이겠다. 입체 형상은 서울대학교에서 개발 중인 삼차원 형상 모델링 시스템을 사용하여 생성하였는데, 생성된 입체형상의 기하학적인 정보는 NURBS로 표현되며 위상학적인 정보는 winged edge data structure로 저장된다. 개발된 알고리즘의 실행은 10 Mips(Personal Iris : Silicon Graphics 사)의 시스템에서 수행되었다.

먼저 Fig. 18과 같이 인장하중을 받는 얇은 평판에 구멍이 있는 경우를 해석하고자 할 때, 이에 대한 유한요소를 자동으로 생성하는 경우를 고려하자. 이 문제는 대칭이므로 그 4분의 1만을 취하여 해석 모델을 만들고, 사면체의 옥트리로 회귀적 분할을 시작하기 위해 이를 포함하는 정사면체를 나타낸 것이 Fig. 20이다. Fig. 21은, 주어진 해석 모델에 대해 옥트리의 depth를 4로 입력하였을 때 생성된 단말 요소들 중 '부분' 요소와 '내포' 요소만을 은선제거한 결과이다.

그 다음의 예는 Fig. 21과 같이 육면체의 블럭위에 실린더 형상이 비대칭의 위치에 놓인 입체형상의 경우이다. Fig. 22은 사면체의 옥트리로 회귀적

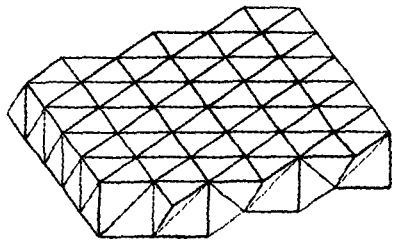


Fig. 20 Tetrahedron-based octree representation

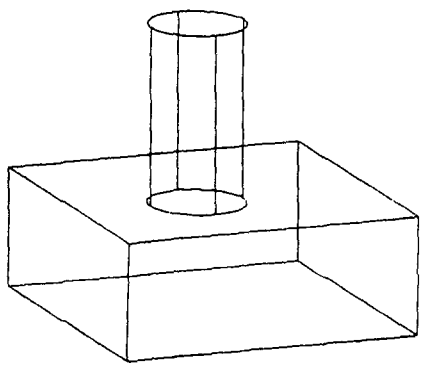


Fig. 21 Test model of cylinder on block

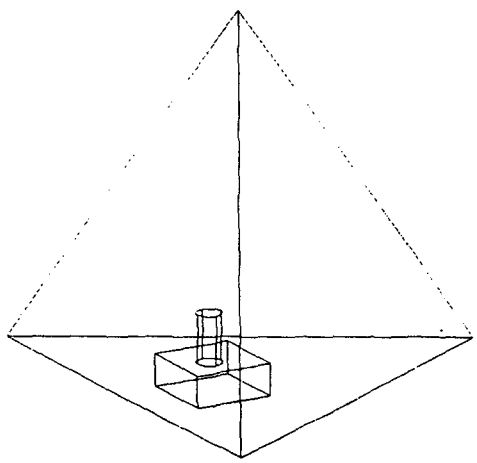


Fig. 22 Root tetrahedron enclosing model

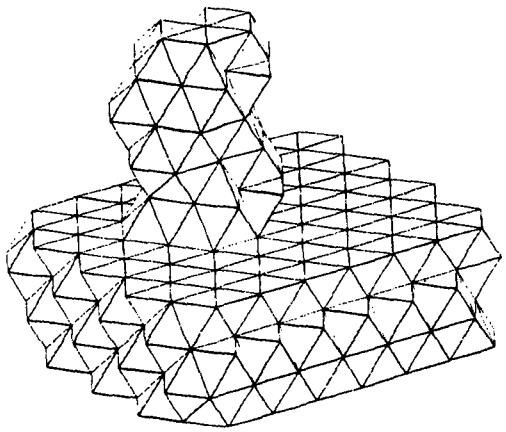


Fig. 23 Octree representaion of depth 5

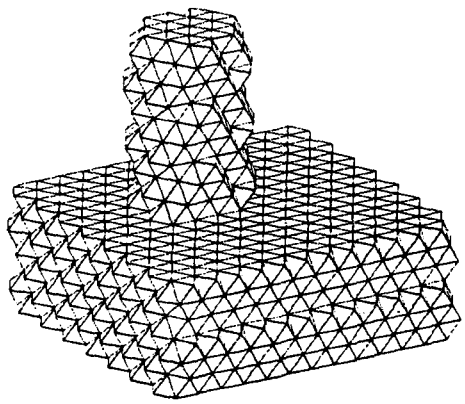


Fig. 24 Octree representaion of depth 6

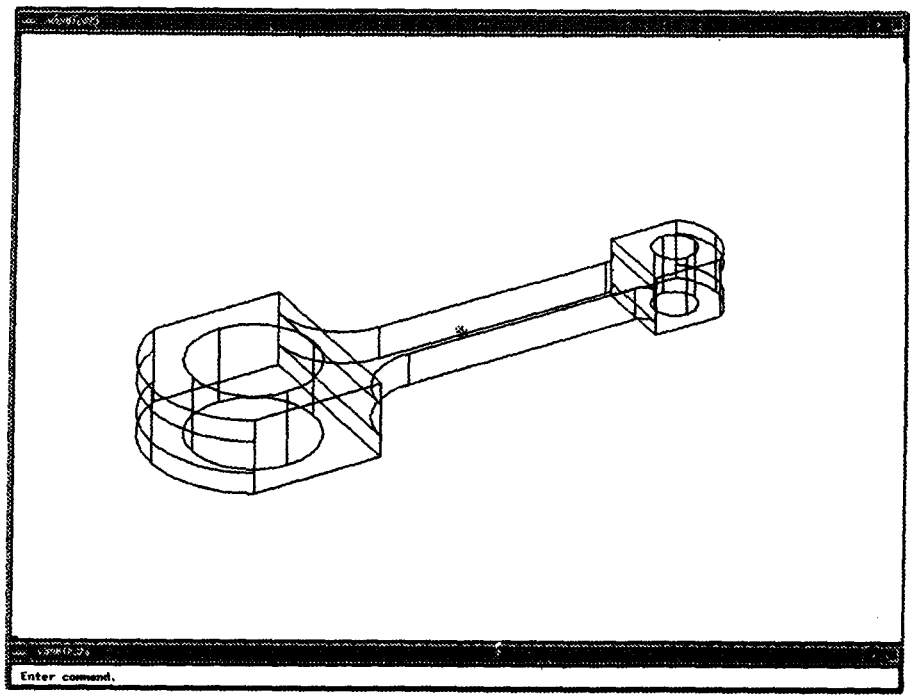


Fig. 25 3-D model of connecting rod

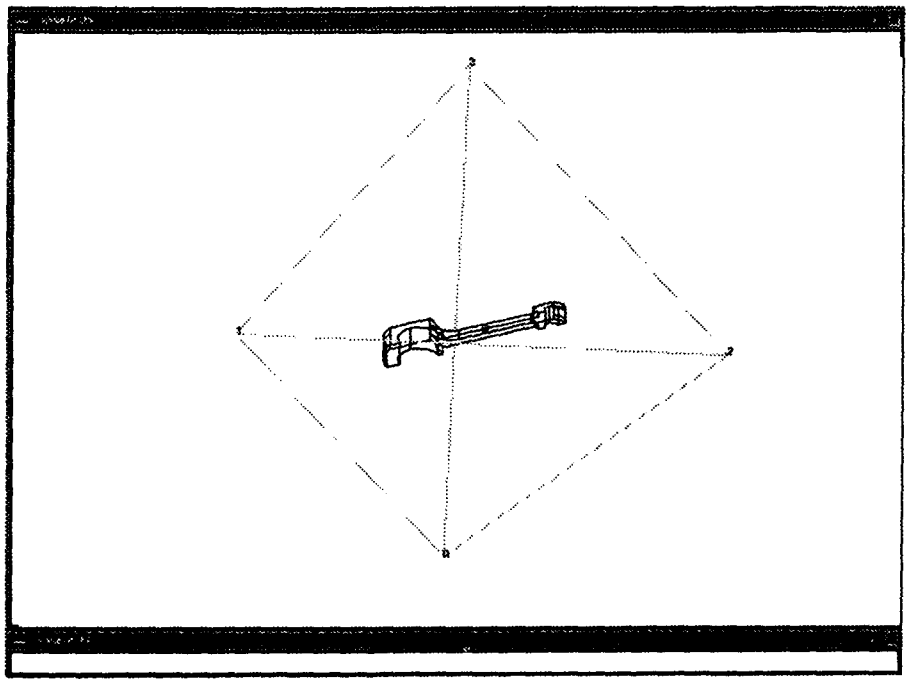


Fig. 26 Root tetrahedron of the half connecting rod

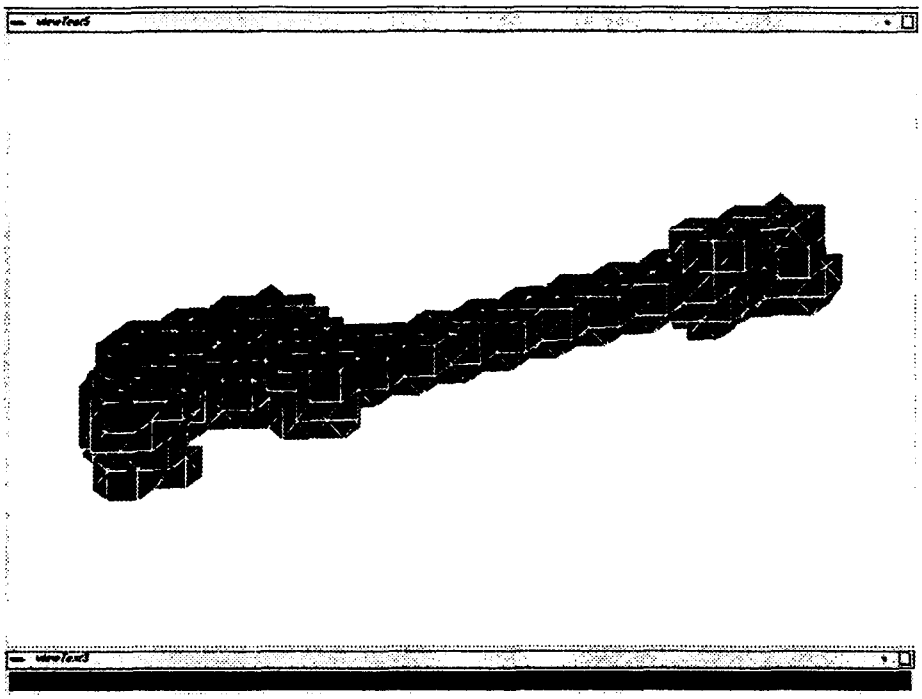


Fig. 27 Partial and full elements of octree with hidden lines removed

분할을 시작하기 위해 주어진 입체를 포함하는 정사면체를 나타낸 것이고, Figs. 23, 24는 옥트리의 depth를 각각 5와 6으로 하였을 때의 결과를 은선제거한 것이다.

그 다음의 예는 3차원 기계요소의 대표적인 형상이라고 할 수 있는 connecthing rod를 예제로 선정하고 이에 대한 입체형상을 Fig. 25와 같이 생성하였다. 생성된 입체형상은 좌우 대칭인 특징을 갖고 있으므로 그 2분의 1만을 취하여 생성한 형상에 대해, 이를 포함하는 뿌리 사면체를 생성한 것을 Fig. 26에 나타내었다. Fig. 27은, 본 연구에서 제안된 알고리즘의 유일한 입력인 옥트리의 depth를 6으로 입력하였을 때 생성된 단말 요소들 중 '내포' 및 '부분' 요소만을 은선제거하여 나타낸 결과이다.

6. 결 론

3차원 문제를 해석하기 위한 유한요소를 자동으로 생성하기 위해, 본 연구의 (Ⅰ)편에서는 사면체에 근거한 옥트리의 생성 알고리즘을 새롭게 개발하였다. 그 기본적인 개념은 주어진 입체형상을 근사화하기 위해 사면체로부터 분할을 시작한다는 것이 육면체로부터 분할을 시작하는 기존의 방법들과 근본적으로 다른 점이다. 또한 사면체 옥트리를 개발함에 있어서 트리(tree)기법의 효율에 필수적인 이웃찾기 알고리즘도 새로이 제안하였다.

이러한 사면체 옥트리기법으로 물체를 근사화하였을 때, 그 단말요소들은 유한요소해법에서 허용되는 사면체 유한요소와 같은 위상을 갖는다. 따라서 입체형상의 경계를 표현하는 단말요소의 몇몇 꼭지점들만 입체형상의 표면으로 이동하면 이웃하는 요소와의 적합성이 자동으로 만족되는 유한요소가 생성되게 할 수 있다. 옥트리의 단말요소로부터 사면체 유한요소를 생성하는 과정은 본 논문의 (Ⅱ)편에서 소개된다.

참고문헌

(1) Ho-Le, K., 1988, "Finite Element Mesh Generation Methods: a Review and Classification," *Computer-Aided Design*, Vol. 20, No. 1, pp. 27~38.

(2) Sapidis, N. and Perucchio, R., 1989, "Advanced Techniques for Automatic Finite Element Mesh-

ing from Solid Models," *Computer-Aided Design*, Vol. 21, No. 4, pp. 248~253.

(3) Yerry, M. A. and Shephard, M. S., 1984, "Automatic Three-Dimensional Mesh Generation by the Modified-Octree Technique," *Int, J. Num. Meth. Eng.*, Vol. 20, pp. 1965~1990.

(4) Yerry, M. A. and Shephard, M. S., 1985, "Automatic Mesh Generation for Three-Fimensional Solids," *Comput. Struct.*, Vol. 41, Np. 4, pp. 169~185.

(5) Shephard, M. S., 1988, "Approaches to the Automatic Generation and Control of Finite Element Meshes," *Applied Mechanics Review*, Vol. 41, No. 4, pp. 169~185.

(6) Perucchio, R., Saxena, M. and Kela, A., 1989, Automatic Mesh Generation from Solid Models Based on Recursive Spatial Decompositions," *Int. J. Num. Meth. Eng.*, Vol. 28, pp. 2469~2501

(7) Saxena, M. and Perucchio, R., 1987, "Geometrical and Topological Issues in Octree Based Automatic Meshing," *Proc. NAFEMS Int. Conf.* on Quality Assurance and Standards in Finite Element Analysis Brighton, UK, Vol. 1, Paper No 2.4.

(8) Farin, G., 1990, *Curves and Durfaces for Computer Aided Geometric Design*, Academic press, Inc., San Diego.

(9) Baumgart, B. G., 1974, "Geometric Modeling for Computer Vision," PhD thesis, Stanford University.

(10) Samet, H., 1982, "Neighbor Finding Technique for Images Represented by Quadtrees," *Comput. Graph. & Image Proc.*, Vol. 18, pp. 37~57.

(11) Requicha, A. A. G., 1982, "Solid Modeling : an Historical Summary and Contemporary Assessment," *IEEE Comput. Graph. Appl.*, Vol. 2, No. 2, pp. 9~24.

(12) Chio, B. K., Shin, H. Y., Yoon, Y. I. and Lee, J. W., 1988, "Triangulation of Scattered Data in 3D Space," *Computer-Aided Design*, Vol. 20, No. 5, pp. 239~248.