Journal of Digital Contents Society Vol. 15 No. 3 Jun. 2014(pp. 347-355)
http://dx.doi.org/10.9728/dcs.2014.15.3.347

Primal Tree의 공간 분할 샘플링 분석 및 구현

박태정*

요 약

컴퓨터 그래픽스나 기하 정보 분석 및 검색 등의 애플리케이션에서 일반적인 octree가 널리 사용된다. 그러나 거리장 등 공간에 분포하는 특정한 연속 정보를 샘플링하기 위한 목적으로 일반적인 octree를 적용할 경우 샘플링 데이터 중복과 샘플링 지점과 표현 단위의 불일치가 발생한다. 이 문제를 해결하기 위해 dual octree가 제안된 바 있다. 본 논문에서는 dual octree가 일반적인 octree의 단점은 해결했으나 무한하게 분할을 수행하더라도 특정한 연속 영역에 액세스하지 못한다는 사실을 증명하고 이러한 모든 문제들을 해결할 수 있는 대안으로 Lefebvre와 Hoppe가 제안한 primal tree를 응용할 수 있음을 제시한다. 또한 트리 구조의 병렬화에 널리 사용되는 Morton code를 응용한 3차원 primal tree 검색 알고리즘을 제안한다.

키워드 : 옥트리, 듀얼 트리, 프리멀 트리, 공간 분할 샘플링, 모턴 코드, 트리 병렬화

Analysis on Spatial Sampling and Implementation for Primal Trees

Taejung Park*

Abstract

The general octree structure is common for various applications including computer graphics, geometry information analysis and query. Unfortunately, the general octree approach causes duplicated sample data and discrepancy between sampling and representation positions when applied to sample continuous spatial information, for example, signed distance fields. To address these issues, some researchers introduced the dual octree. In this paper, the weakness of the dual octree approach will be illustrated by focusing on the fact that the dual octree cannot access some specific continuous zones asymptotically. This paper shows that the primal tree presented by Lefebvre and Hoppe can solve all the problems above. Also, this paper presents a three-dimensional primal tree traversal algorithm based the Morton codes which will help to parallelize the primal tree method.

Keywords : octree, dual octree, primal tree, spatial division sampling, Morton code, parallel tree structures

1. 서론

1.1 연구 배경

Octree 구조는 컴퓨터 그래픽스는 물론, 로보틱스, 공학 시뮬레이션 수치 해석, 다중 해상도 지원(LOD)[1] 등 여러 분야에서 공간 정보의 효율적인 저장과 검색을 위해 사용된다.

그러나 연속 거리장의 효율적인 이산화 및 보간[2, 3], 부호 정보를 이용한 매시 압축[4, 5], 기하 모델 합성[6] 등 연속 정보의 샘플링이 필요한 응용 분야에서는 다음과 같은 일반적인 octree의 한계로 인해 효율성이 저하된다.

a. 샘플링 데이터의 중복 저장

샘플링(sampling)이란 연속 데이터를 이산화

※ 교신저자(Corresponding Author): Taejung Park
접수일:2014년 03월 30일, 수정일:2014년 06월 05일
완료일:2014년 06월 15일
* 덕성여자대학교 디지털미디어학과
Tel: +82-2-901-8339 , Fax: +82-2-901-8646
email: tjpark@duksung.ac.kr
▣ 본 연구는 덕성여자대학교 2013년도 교내연구비 지원에 의해 수행되었음

된 지점(point)에서 측정 또는 계산하는 과정을 의미한다. 일반적인 octree에서는 이러한 샘플링 작업이 직육면체로 일반적으로 표시되는 노드의 여덟 개 모서리 지점에서 수행된다. 그러나 일부 모서리를 제외하고는 거의 모든 모서리들이 주변의 octree 노드에 공통적으로 포함되며 모서리 지점 1개 당 최대 8개의 인접 노드를 가질 수도 있다. 이러한 특성으로 인해 실제 구현 코드에서 샘플링한 데이터의 단일 복사본을 유지하기 위해서 인접한 8개의 노드들이 불필요하게 단일 데이터 복사본의 포인터를 저장해야 하고 이러한 저장을 위해서는 추가적으로 인접성 검사 루틴을 실행해야 하는 문제가 발생한다.

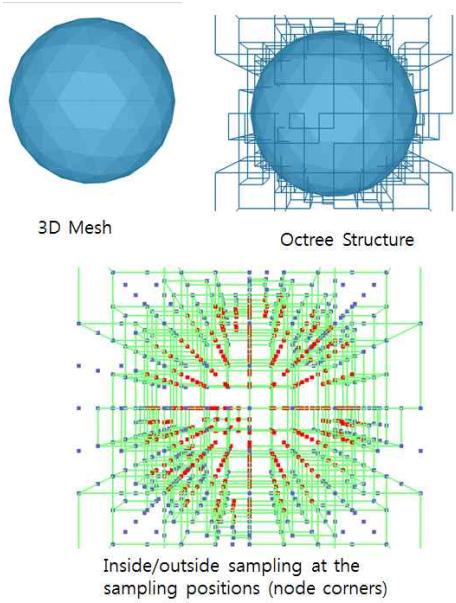
b. 샘플링 지점과 표현(representation) 영역의 불일치

앞서 설명한 대로 샘플링 지점에서는 연속 정보의 정확한 값을 측정이나 계산을 통해 얻고 저장한다. 따라서 일반적인 octree를 이용해서 샘플링을 수행한 경우 - 앞서 논의한 데이터 저장의 중복 및 비효율성 문제에도 불구하고 - 각 노드의 모서리 지점은 정확한 샘플링 값을 확보하게 된다.

그러나 일반적인 octree에서는 octree 노드의 모서리 지점이 샘플링 위치로 이용되는 것이 비해서 이러한 샘플링 데이터의 대표 또는 가시화 공간은 모서리 지점이 아니라 대부분 직육면체 부피로 표현되는 octree 노드이다. 이러한 불일치 문제를 극복하기 위해서 노드 내부 공간은 불가피하게 모서리의 정확한 정보를 보간(interpolation)한 근사값으로 표현한다. 보간은 필연적으로 보간 오차를 발생시킬 수 밖에 없다. 참고로 최근 제안된 primitive tree 기법을 이용한 정확한 거리 계산 기법[3]에서는 거리장에서 일반적인 octree에서 필연적으로 발생하는 보간 오차[2]를 줄이기 위한 방안을 제시한 바 있다.

이러한 일반적인 octree의 문제를 해결하기 위해서 dual octree[7]가 제안되었다. Dual octree는 앞서 논의한 샘플링 데이터 중복 문제와 샘플링 지점과 표현 영역의 불일치 문제를 해결한 방법이다. 이 dual octree를 이용한 여러 기법들이 제안되었으나 본 논문에서는 본 논문이 초점을 맞추고 있는 '연속 공간 정보의 샘플링'에 dual octree를 적용할 경우, 무한하게 분할을 수

(그림 1) 3차원 기하 정보에 대한 일반적인 옥트리 구조



(Figure 1) General octree structure for three-dimensional geometry

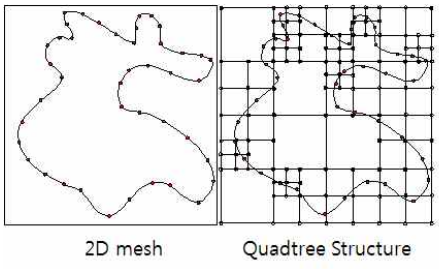
행하더라도 특정한 연속 영역(3차원의 경우 경계가 적용되는 평면, 2차원인 경우 선분)이 샘플링에서 제외된다는 점을 제시하고 그 대안으로 Lefebvre와 Hoppe가 소개한 primal tree[8]의 사용을 제안한다.

2. 공간 분할 방식으로서의 옥트리(Octree)

2.1 일반적인 Octree와 한계

Octree는 보편적으로 사용되는 공간 분할 자료 구조로 문제 공간(bounding box)을 분할 기준에 따라 각 축을 따라 1/2로 연속적으로 분할한 후 원하는 기하 정보를 검색하는 용도로 사용된다. Octree는 이진 분할(binary section) 알고리즘 또는 이진 검색 트리(binary search tree)의 3차원 공간으로의 자연스러운 확장으로 볼 수도 있다. (그림 1)에서는 일반적인 octree를 이용해서 3차원 매시의 외부와 내부를 구분하는 전형적인 애플리케이션을 볼 수 있다. 특히 이러

(그림 2) 2차원 기하 정보에 대한 일반적인 쿼드 트리



(Figure 2) General quadtree structure for two-dimensional geometry

한 애플리케이션은 부호 정보를 이용하는 매시 압축 기법[4, 5]에 응용된다. 이러한 적용 분야에서는 octree의 한 노드인 직육면체의 각 8개 꼭지점이 샘플링 지점으로 사용된다. (그림 2)에서 제시한 것처럼 octree가 2차원 평면 도형에 적용될 경우는 quadtree라고 한다.

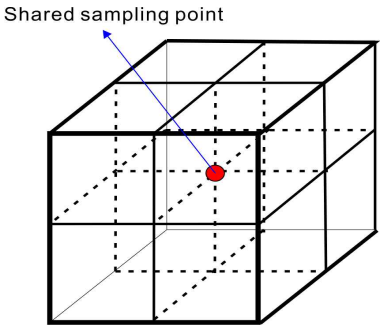
그러나 일반적인 octree에서는 노드의 꼭지점 하나가 최대 8개의 주변의 노드들과 공유되기 때문에 구현 및 성능 최적화가 어려운 단점이 있다(그림 3).

2.2 Dual Octree

2.1에서 서술된 일반적인 octree의 단점을 다른 관점에서 본다면 샘플링 지점(노드의 모서리)과 그 샘플링된 정보를 대표하는 기하 요소(직육면체로 표현되는 노드)의 공간적 불일치 때문이라고 정리할 수 있다. 이러한 문제는 - 비록 명시적인 표현으로 설명된 경우는 드물지만 - 몇몇 연구자들이 인지했고 그 대안으로 dual octree가 제안되었다[7].

이 dual octree는 Poincaré의 cell complexity에 대한 duality 개념[9]으로 정의되는데 (그림 4)에서 표현한 것처럼 일반적인 octree의 직육면체 노드가 한 점으로 표현된다. 이 dual octree는 음함수 표면의 폴리곤화 등에 널리 사용되는 dual contouring 기법[10] 등으로 응용되며 노드가 부피가 아니라 점으로 표시되기 때문에 샘플링 지점과 샘플링 값을 대표하는 위치가 일치하는 장점을 보인다. 그러나 3장에서의 논의 내용

(그림 3) 빨간색 점은 공유 지점을 표시. 이 점이 샘플링 지점을 사용될 경우, 주변의 8개 노드들이 이 정보를 모서리로 공유해야하기 때문에 중복을 피하고 샘플링된 데이터를 유지 관리하기가 어렵다.



(Figure 3) The red dot represents a shared sampling point. Since the sampling point is a common corner of all neighboring 8 nodes, it is hard to avoid duplication and maintain the sampled data.

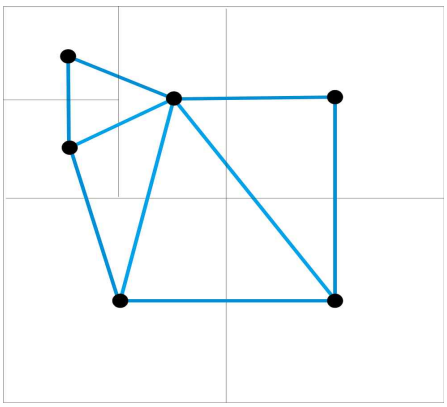
처럼, dual octree는 샘플링에 적용할 경우 치명적인 단점이 존재한다.

2.3 Primal Tree

Lefebvre와 Hoppe는 공간적으로 일관적인 데이터(예를 들어 이미지에서 한 단위로 인지하는 연속적인 영역)의 효율적인 보관과 무작위 액세스, 압축 효과를 얻을 수 있는 새로운 트리 구조를 제안하고 이 이름을 primal tree라고 명명한 바 있다[8]. 그러나 이 primal tree는 상대적인 위치에 따라서 적용되는 child 노드의 최대 개수가 서로 다르며 일부 child 노드의 위치와 데이터가 자신의 parent 노드와 항상 동일하다는 특징, 그리고 트리 구조 시각화에서의 난해함 때문에 이 데이터 구조가 제공하는 여러 장점들에도 불구하고 그 적용이 제한적이다. (그림 5)에서는 일반적인 octree와 primal tree의 차이점을 제시한다.

본 논문에서는 primal tree가 dual octree와 유사하게 샘플링 위치와 tree의 노드가 대표되는

(그림 4) 일반적인 옥트리(실제로는 설명의 단순함을 위해 쿼드트리 제시)와 듀얼 옥트리(파란색 선분과 검은색 점으로 표시)



(Figure 4) General octree (actually, quadtree is shown for simplicity) and its dual octree (light blue segments and black dots).

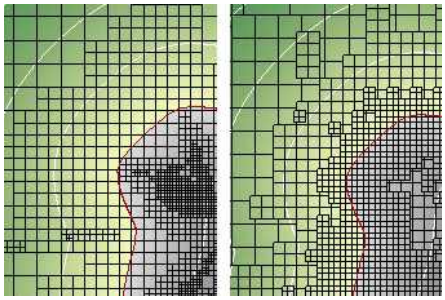
지점이 일치하는 장점은 물론, 일반적인 octree에서 볼 수 있는 샘플링 위치의 중복 문제가 없다는 장점에 주목하고자 한다. 동시에 dual octree가 내포하는 특정 영역의 샘플링 액세스 불가능 문제도 해결할 수 있다는 측면에서 공간 샘플링에서의 primal tree의 장점을 분석, 제시하고자 한다. Primal tree에 대한 보다 자세한 내용은 [8]을 참고한다.

3. Primal Tree와 Dual Octree의 비교

3.1 샘플링 공간의 접근성

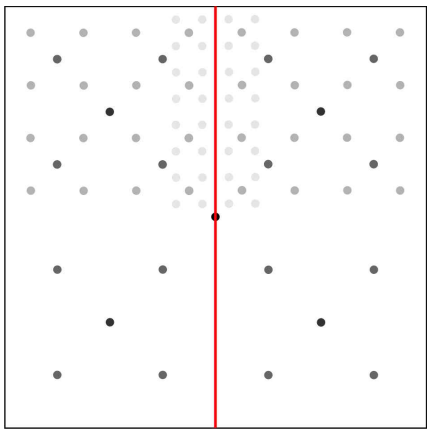
앞서 살펴 본 대로 primal tree와 dual octree는 일반적인 octree에 비해서 샘플링 지점 중복 제거를 통한 데이터 저장 및 액세스의 효율성, 구현의 용이성 등의 장점을 가진다. dual octree는 일반 octree의 노드를 sampling point로 변환하는 dual 관계로 구성되기 때문에 보다 비직관적인 분할 구조를 가지는 primal tree에 비해서 이해와 구현이 쉽다는 장점이 있다. 그러나 dual octree를 무한히 분할하더라도 특정한 연속 공간

(그림 5) 일반적인 옥트리 구조(왼쪽) 및 primal 트리 구조(오른쪽) [3].



(Figure 5) General octree structure (left) and primal tree structure (right) [3].

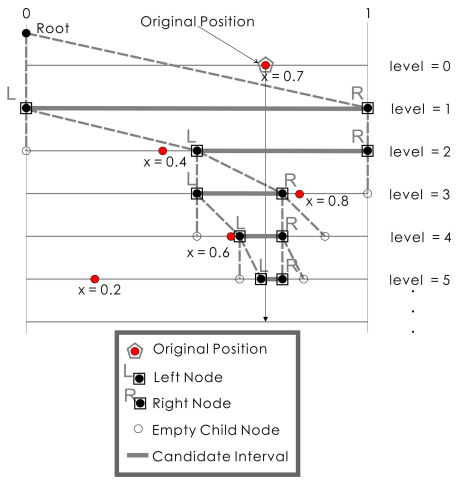
(그림 6) 듀얼 쿼드트리에서 점근적으로 접근이 불가능한 영역



(Figure 6) Asymptotically unaccessible region in dual quad-tree

을 샘플링하지 못하는 단점이 있다. 또한 dual contouring이나 dual marching cube와 같은 polygonization에 적용할 경우에는 dual octree만으로 샘플링을 실시하지 못하고 일반 octree의 한 노드에서 8개의 모서리 점을 샘플링 지점으로 이용하기 때문에 일반 octree와 동일한 샘플링 지점 공유 문제가 그대로 발생한다. 또한 dual octree의 꼭지점(vertex)을 그대로 샘플링 지점으로 활용하려고 하는 경우에는 특정한 영역이 샘플링에서 반복적인 패턴으로 제외되는 문제가 발생한다. (그림 6)에서는 dual octree의

(그림 7) 1차원 primal 트리의
1차원 순회 단계 설명



(Figure 7) Illustration of
one-dimensional traversal steps for 1D
primal trees

2차원 버전인 dual quad tree에서 이 문제를 제시한다. 이 그림에서 검은색 원으로 표시된 사각형 중심점 위의 샘플링 점(레벨 0 노드)을 제외하고는 무한하게 주변 노드들을 분할하더라도 빨간색 선으로 표시된 영역에 액세스할 수 없다. 이와 동일하게 3차원 공간에서 dual octree를 적용할 경우에도 무한하게 분할하더라도 특정한 평면들에 샘플링 점들에 액세스할 수 없는 문제가 발생한다. 특히 dual octree의 샘플링 지점들은 어떠한 부분을 확대하더라도 전체와 동일하거나 유사한 프랙털 구조를 가지기 때문에 이렇게 액세스가 불가능한 영역은 하위 레벨에서도 동일한 패턴으로 무한하게 반복된다. 위에서 살펴 본 것과 같이 공간 샘플링 문제에서 아무리 분할을 하더라도 특정한 연속 영역(2차원 dual quad tree에서 직선, 3차원 dual octree에서 평면)을 샘플링하지 못한다는 사실은 이 dual tree의 치명적인 문제점이라고 할 수 있다. 이에 비해서 (그림 5)의 오른쪽 그림에서 볼 수 있듯이 primal tree의 경우 격자 구조가 노드 분할 시에 균일한 격자 구조와 거의 동일한 간격으로 적응형으로 분할되어 특정 연속 영역이 누락되는 문제가 발생하지 않는다.

<표 1> 1차원 primal 트리 순회 코드

eval1D [2],[7]

```
class TreeNode{
    TreeNode L, R;
    Data P;
};

Set eval1D(TreeNode root, float x){
   Set C;
   TreeNode LNode = root.L, RNode = root.R;
   while(true){
      if(!LNode && !RNode)
         return C;
      if( x < 0.5 ){
         x=(x-0.0)*2.0;
         RNode = LNode ? LNode.R : NULL;
         LNode = LNode ? LNode.L : NULL;
      }else{
         x=(x-0.5)*2.0;
         RNode = RNode ? RNode.L : NULL;
         LNode = LNode ? LNode.R : NULL;
     }
     if(LNode)
          C.add(LNode.P);
     if(RNode)
          C.add(RNode.P);
    }
}
```

<Table 1> One-dimensional primal tree
traversal code

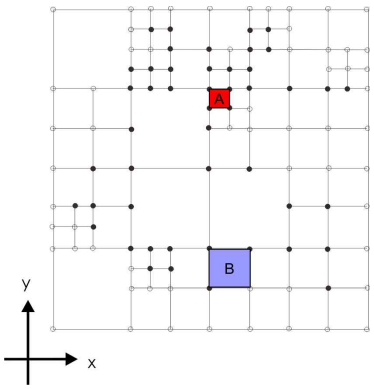
4. 3차원 Primal Tree 구현

Primal Tree의 1차원 traversal 코드는 [8]에서 처음 소개되었으며 이후 [3]에서 정확한 거리장 계산을 위한 primitive tree의 구현을 위해 적용된 바 있다. 그러나 이 논문들에서는 1차원 traversal 코드만 소개되었다. 본 논문에서는 [3] 및 [8]에서 논의된 1차원 traversal 코드를 바탕으로 효율적인 3차원 traversal 코드를 구현한다.

4.1 1차원 Primal Tree traversal 코드

<표 1>에서는 [3] 및 [8]에서 제시된 1차원 traversal 코드를 제시한다. 이 코드의 특징은 모든 단계에서 검색 영역을 [0, 1] 범위로 스케일해서 효율적인 검색을 수행한다는 점이다. (그림

(그림 8) 일반적인 쿼드 트리에서의 2차원 Morton 코드 예제.



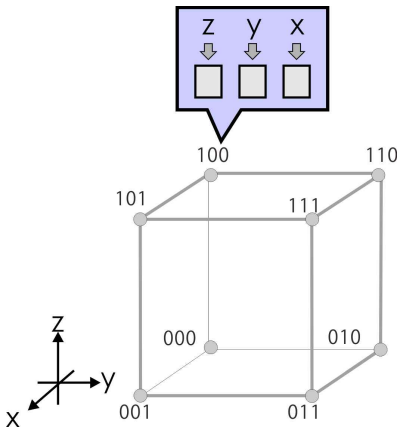
(Figure 8) 2D Morton code examples with general quadtree.

7)은 <표 1>에 따라 1차원 전체 공간 [0, 1]에서 기하 요소의 위치가 0.7일 때 <표 1>에서 제시된 검색 알고리즘으로 레벨 5까지 검색을 수행하는 과정을 나타낸 그림이다. 이 그림에서 primal tree 검색 과정이 항상 [0, 1] 범위로 문제 공간이 스케일되는 특성을 볼 수 있다. 1차원 알고리즘을 3차원으로 확장하는 문제는 검색 공간을 항상 [0, 1] 범위로 스케일하는 primal tree의 특성 때문에 3차원 공간에서의 자식 노드의 관리를 위해 좀 더 효율적인 접근이 필요하다. 따라서 본 논문에서는 octree의 GPU 병렬 구현 등에서 널리 사용되는 Morton Code 방식과 유사한 3차원 primal tree traversal 접근 방식을 제안한다.

제안하는 방식에서는 Morton code와 유사하게 각 축 단위로 child 노드의 정보를 2진수로 표현한다. 이 표현에서는 3차원 공간 좌표축의 원점과 가까운 child 노드는 0, 먼 노드는 1로 표시하고 x, y, z 축에 따라 “zyx”축 순서로 0 또는 1을 표시해서 각 child 노드를 나타낼 수 있다. 예를 들어 (그림 9)에서 제시한 것처럼 각 child 노드는 상대적인 위치에 따라서 000, 001, 010, 011, 100, 101, 110, 111로 나타낼 수 있다. 이런 표현 방법은 child 노드 한 단계에 그치지 않고 이후의 child 노드들을 표현하기 위해 연속적인 이진 코드로 표현할 수 있다. 이 코드를 Morton code라고 한다.

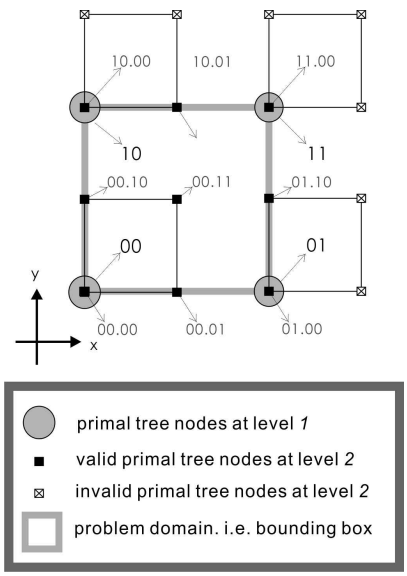
3차원 octree의 경우, 세 개의 비트를 통해 한

(그림 9) 일반적인 옥트리에 대한 Morton 코드



(Figure 9) Morton code for general octree

(그림 10) 2차원 primal 트리 Morton 코드



(Figure 10) Morton code for 2D primal tree.

레벨의 child 노드들을 표현할 수 있으며 2차원 quadtree의 경우, 두 개의 비트(yx)를 통해 동일하게 Morton code를 표현할 수 있다. 예를 들어

<표 2> Morton 코드를 사용하는 2차원 primal 트리 알고리즘

Current 노드 Indices	Positions of two-dimensional points	Assignment for next levels
00	p.y < 0.5, p.x < 0.5	00 ⇐ 00.00 01 ⇐ 00.01 10 ⇐ 00.10 11 ⇐ 00.11
01	p.y < 0.5, p.x > 0.5	00 ⇐ 00.01 01 ⇐ 01.00 10 ⇐ 00.11 11 ⇐ 01.10
10	p.y > 0.5, p.x < 0.5	00 ⇐ 00.10 01 ⇐ 00.11 10 ⇐ 10.00 11 ⇐ 10.01
11	p.y > 0.5, p.x > 0.5	00 ⇐ 00.11 01 ⇐ 01.10 10 ⇐ 10.01 11 ⇐ 11.00

<Table 2> Two-dimensional primal tree algorithm using Morton code

서 (그림 8)에서 제시한 quadtree 노드 두 개를 "yx" 순서로 표시할 경우 A는 11 00 10 10, B는 01 00 10으로 표시할 수 있다(왼쪽에서 오른쪽으로 세밀한 레벨로 진행).

본 논문에서는 이러한 Morton code와 유사한 2진 색인 적용을 통해서 <표 1>에서 제시한 기존의 1D traversal 알고리즘을 3D로 확장하는 방식을 제안한다. (그림 10)에서는 상위 레벨에서의 primal 노드 4개(회색 사각형 모서리 원 4개)가 각각 00, 01, 10, 11로 설정되고 그 하위 child 노드들이 검은색 사각형 점으로 표시되는 상황을 제시하고 있다. 이 그림에서 일반적인 octree와는 다른 primal tree의 특징들을 몇 가지 살펴 볼 수 있다.

먼저 일반적인 octree와는 달리 이 primal tree에서는 노드가 직사각형(2D)이나 직육면체(3D)로 표시되지 않고 한 점으로 표시된다. 또한 primal tree에서는 무효 노드가 발생하는 특징 때문에 한 점으로 표시되는 각 노드의 상대적인 위치에 따라서 가질 수 있는 유효 child 노드의 개수가 다르다. 예를 들어 (그림 10)에서 왼쪽 아래에 위치하는 레벨 1에 해당하는 노드 00은 총 4개의 child 노드(특히 00.00은 부모 노드와

<표 3> 3차원 primal 트리 순회 코드 (일부 내용 생략)

eval3D

```
#define AXIS_X 0
#define AXIS_Y 1
#define AXIS_Z 2

Set eval3D(float3 v)
{
   Set result;
   unsigned short int xb_idx = 0u
   unsigned short int yb_idx = 0u
   unsigned short int zb_idx = 0u;
   unsigned short int grandChildIdx = 0u;
   unsigned short int childIdx;
   unsigned short int i;
   bool bSet[3];
   PrimalTreeNode * pNode = m_pRoot;

   while(true){
      if(!pNode->doesHaveAnyChild())
         return result;
      for(i=0; i< 3; i++){
         bSet[i] =  (v[i] >= 0.5f);
         if(bSet[i])
            v[i] = (v[i] - 0.5f)*2.0f;
         else
            v[i] = (v[i] - 0.0f)*2.0f;
      }
      xb_idx = bSet[AXIS_X];
      yb_idx = bSet[AXIS_Y];
      zb_idx = bSet[AXIS_Z];
      grandChildIdx
         = xb_idx + yb_idx << 1 + zb_idx << 2;

      for(i = 0 ; i < 8; i++){
         xb_idx += i & 1u;
         yb_idx += i & 2u;
         zb_idx += i & 4u;
         childIdx
            = (zb_idx & 2) << 2
            + (yb_idx & 2) << 1
            + (xb_idx & 2) ;
     pNode->setChild(i,
 getChild(childIdx)->getChild(grandChildIdx));
     if(pNode->getChild(i))
       result.add(pNode->getChild(i)
->getChildPrimIDs());
      }
   }
};
```

<Table 3> Three-dimensional primal tree traversal code (omitted some details)

중복)를 가질 수 있는데 비해서 같은 레벨 2에 있는 왼쪽 아래 모서리 노드 01의 경우 child 노드를 2개 (마찬가지로 01.11은 부모 노드와 중복)만 가지고 나머지 두 개 노드(그림에서 ⊠로 표시되는 노드)는 문제 영역을 벗어난 무효 노드가 된다. 그리고 레벨 1에 속하는 노드로 오른쪽 위의 11 노드는 단 하나의 유효 child 노드만을 가지며 이 child 노드 11.00은 부모 노드 11과 중복된다. 이제 이 색인 코드를 이용해서 <표 1>에서 제시한 1D traversal 코드와 같이 [0, 1] 범위를 2차원, 3차원 공간으로 확장하는 방법을 <표 2>처럼 정리할 수 있다. 특히 <표 2>에서 p.x에 대한 내용만 추출하면 <표 1>에서 제시한 1D traversal 코드와 동일해 진다는 사실을 알 수 있다. <표 3>에서는 3차원 primal tree의 traversal 코드를 정리한다.

5. 결론

효율적인 연속 공간 정보 샘플링을 위해 기존 방식들이 지닌 문제점을 고찰하고 그 대안으로 주로 mipmap, 공간 데이터 압축 등의 특정한 용도로 제안되었던 primal tree를 적용할 경우 기존 방식의 여러 문제점들을 해결할 수 있음을 제시했다. 또한 primal tree의 효율적인 3차원 탐색을 위해 Morton 코드를 응용하는 방안을 설명하고 구현 코드를 제시했다. 특히 제안한 primal tree의 Morton 코드는 GPU 병렬화의 기초가 되는 방식으로 앞으로 병렬화에도 활용할 수 있을 것으로 기대한다.

References

[1] H. Kim and H Park, “A Hybrid Rendering Model to support LOD,” Journal of Digital Contents Society vol. 9 no. 3, pp. 509-516, Sep. 2008.

[2] S. F. Frisken, R. N. Perry, A. P. Rockwood, and T. R. Jones, “Adaptively sampled distance fields: a general representation of shape for computer graphics,” Proc. SIGGRAPH 2000, pp.249-254, July 2000.

[3] S. Lee, T. Park and C. Kim, "Primitive Trees for Precomputed Distance Queries," Computer Graphics Forum (Proc. EUROGRAPHICS 2013), vol. 32, no. 2, pp. 419-428, May 2013.

[4] T. Park, H. Lee and C. Kim, "Progressive Compression of Geometry Information with Smooth Intermediate Meshes," Iberian Conference on Pattern Recognition and Image Analysis 2007, vol. 4478, pp. 89-96, June 2007.

[5] H. Lee, M. Desbrun, and P. Schröder, “Progressive Encoding of Complex Isosurfaces,” ACM Transactions on Graphics vol. 22, no. 3, pp. 471-476, July 2003.

[6] S. Lee, T. Park, J. Kim and C. Kim, "Adaptive Synthesis of Distance Fields," IEEE Transactions on Visualization and Computer Graphics, vol. 17, no. 7, pp. 1202-1206, July 2012.

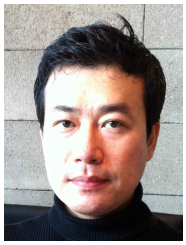
[7] T. Lewiner, V. Mello, A. Peixoto, S. Pesco and H. Lopes, “Fast Generation of Pointerless Octree Duals,” Computer Graphics Forum vol. 29, pp. 1661 - 1669 Sep. 2010.

[8] S. Lefebvre and H. Hoppe, “Compressed Random-access Trees for Spatially Coherent Data,” Proc. the 18th Eurographics Conference on Rendering Techniques (EGSR'07), pp. 339-349, 2007.

[9] A. Hatcher, “Algebraic Topology,” Cambridge University Press, 2002.

[10] T. Ju, F. Losasso, S. Schaefer, and J. Warren, “Dual Contouring of Hermite Data,” ACM Transactions on Graphics vol. 21, no. 3, pp.339-346, July 2002.

박 태 정



1997년 : 서울대 전기공학부(학사)
1999년 : 서울대 전기공학부 대학원 (공학 석사, 반도체 전공)
2006년 : 서울대 전기컴퓨터공학부 대학원 (공학박사, 컴퓨터 그래픽스 전공)
2006년～2013년: 고려대학교 연구교수
2013년～현 재: 덕성여자대학교 디지털미디어학과 조교수
관심분야 : 컴퓨터그래픽스, 병렬처리, 게임 물리, 수치해석, 3차원 모델링