社団法人 情報処理学会 研究報告
IPSJ SIG Technical Report

2006－CG－123 (6)
2006／7／11

適応的四面体格子表現のDual Grid生成法

高間康文 田中弘美

CT(Computed Tomography) や MRI(Magnetic Resonance Imaging) に代表される医用断層画像取得装置の高速・高精度化により，再構成されるボリュームデータのサイズが増大している．このようなデータに対してボクセル数を削減し，レンダリングなどの計算負荷を減らすために，3 次元空間を構成する最も単純でロバストなセルである四面体を用いたボリュームデータの多重解像度表現が広く認識されている．しかし，不規則な格子構造を持つこのような表現のボリュームデータを，レイキャスティング法によりレンダリングするためには，膨大な量の四面体セルの中から面を共有して隣接している四面体セルを探し出す必要があるので困難である．そこで我々は，並列に生成された四面体格子表現に隣接関係を示す Dual Grid を新しく付加する方法を提案し，これを用いたボリュームレイキャスティングを試みることでその有効性を示す．

Dual Grid Generation for Tetrahedral Adaptive Grid

YASUFUMI TAKAMA and HIROMI T.TANAKA

We propose dual grids generation for adaptive represented tetrahedrons. Hierarchical representation is very useful to perform several tasks efficiently since a volume data is very large. There are some famous hierarchical representations. In particular, tetrahedron-based approach has been used because a tetrahedron cell is the most robust cell in 3D.But rendering a hierarchical volume model using volume ray-casting is difficult since we can not understand a neighbour cell. In this paper, we try to generate dual grids that show us links to neighbour tetrahedrons sharing a face of tetrahedrons.

1. はじめに

CT や MRI により撮像された複数の断面画像を再構成して得られるボリュームデータは膨大であるので，効率的にタスクを行うには適応的に表現された多重解像度のボリュームモデルが必要とされる[1]．

ボリュームデータの代表的な多重解像度表現として，立方体を再帰的に 8 分割する方法を用いたオクツリー表現[2] や，3 次元空間を構成する最も単純でロバストなセルである四面体を再帰的に分割することで得られる四面体格子表現[3][4][5][6] などがある．

これらの方法により生成される多重解像度表現を持つボリュームデータのレンダリングには，従来から知られているセルプロジェクション法[7][8] や，レイキャスティング法[9][10] が有効であるが，四面体セルをスクリーン平面に投影するセルプロジェクション法[7] は，視点やセルが動く度に投影される四面体の形状を再計算する必要がある．さらにこの方法を GPU(Graphics Processing Unit) を用いてレンダリングする方法[8] は，視点やセルが動く度に再計算した四面体形状を GPU に送る必要がある．

一方，レイキャスティング法を用いてオクツリー表現されたボリュームモデルをレンダリングする方法[9] は，不透明度が 0 であるリサンプリング点はピクセルのカラー合成に貢献しないので，8 つのボクセルの不透明度がすべて 0 であるときには，その内部のリサンプリングを行わないようにすることによって時間効率の向上が期待できるが，オクツリー表現によるボリュームモデルは異なる階層間に 8 倍の近似精度差が生じてしまう．また，階層間の近似精度差が 2 倍ですむ四面体格子表現を用いた場合，四面体セル集合の中から面を共有して隣接している四面体セルを探し出す必要があり，Weiler らはこのような隣接四面体セルの情報をすべて記憶し，GPU を用いて四面体格子表現されたボリュームデータのレイキャスティングを実現している[10]．しかし，使われているデータは GPU に読み込める程度の比較的小さなデータであり，膨大な量の四面体セルにおいて面のデータ構造を定義することはメモリコストの増大化をもたらすと考えられる．

† 立命館大学 コンピュータビジョン研究室
Computer Vision Lab., Ritsumeikan University

そこでこの問題の解決方法として，隣り合う四面体同士の隣接関係を示す双対グラフを生成することが考えられる．従来では，4分木や8分木で表されたデータ構造において，隣接する正方形セル，立方体セルを探索する方法が提案されている[11][12]．また，四面体を2分割することで生成される2分木において隣接する四面体セルを探索する方法[13]も提案されているが，この方法で使われている四面体分割法ではアルゴリズムを並列化していないので，並列に処理を実行した場合，分割の整合性を保った四面体格子表現を生成することは不可能であり，隣接セルを探索する方法も並列処理によって生成された四面体格子表現を用いていない．さらに，多重解像度表現では隣接セルのレベル差が生じることも考えられるがこの方法では考慮されていない．

そこで本論文では，Takama らによって提案された適応的四面体格子表現並列生成法[6]を用いることで，膨大なボリュームデータの適応的四面体格子表現を並列に生成し，並列処理によって得られた四面体格子表現の隣接関係を示す双対グラフ（Dual Grid と呼ぶ）を四面体2分割法の特徴である鏡面性と再帰性を用いて効率よく生成するアルゴリズムを新しく提案する．さらに，Dual Grid を付加された四面体格子表現を用いてボリュームレイキャスティングを行うことにより，本アルゴリズムが有効であることを示す．

2. 適応的四面体格子表現

3次元空間を構成する最小のプリミティブである四面体セルを用いて，膨大なボリュームデータを局所特徴量の変化に基づき再帰的に分割し，入力ボリュームデータの適応的四面体格子表現を並列に生成する[6]．

適応的四面体格子表現並列生成法は，最初にボリューム空間を1辺が n ボクセルの初期立方体格子にわけ，それぞれの初期立方体格子を6つの鏡面対称な四面体に分割し，各四面体を最も長い辺の中点に新しく格子点を生成しながら再帰的に2分割することにより適応的四面体格子表現を生成できる．

生成された表現は図1のような2分木構造により表され，初期立方体格子を分割した6つの四面体が，それぞれ2分木のルートノードになる．2分木の各ノードは四面体セルを持ち，四面体セルは4つの格子点の情報や，親四面体，左右子四面体を示すアドレスの他に，四面体の形状 (TYPE1～TYPE3) と四面体の向き（左右）も格納している．

本論文で提案する Dual Grid 生成法は，この2分木のデータ構造に，各四面体セルが面を共有して隣接している四面体セルへのリンクを生成するものである．

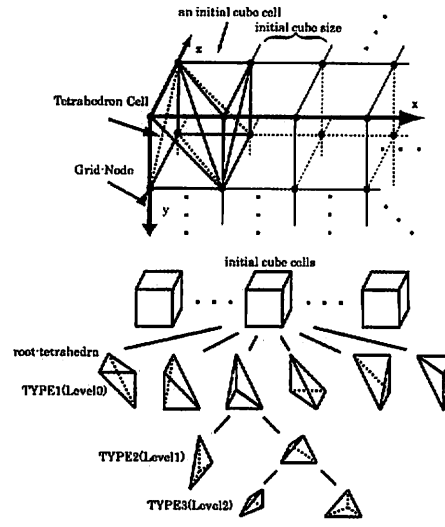


図1 適応的四面体格子表現のデータ構造

3. Dual Grid 生成法

生成された適応的四面体格子表現の隣接関係を示す Dual Grid を生成する方法について述べる．

本研究で用いた2分木構造を持つ適応的四面体格子表現は，「再帰性」と「鏡面性」という2つの性質を持っている（図2）．再帰性とは，四面体を2分割する時に TYPE1 → TYPE2 → TYPE3 → TYPE1（図2）という3回の分割を1周期として繰り返されることである．この1周期の間にできる2分木を小2分木と呼ぶことにし，TYPE1 の四面体を小2分木のルートノードとする．

また，鏡面性とは，四面体 TYPE1 の場合，面を共有して隣接している四面体同士は鏡面対称になることである．すなわち，四面体 TYPE1 の小2分木と，四面体 TYPE1 に隣接している四面体の小2分木は，左右対称な構造になる（図2）．

Dual Grid は，最初に隣り合うルートノード間の隣接関係を生成し，小2分木内と小2分木外の隣接関係を1周期毎に書き換えることを全ての隣接するセル同士がリーフノードのセルになるまで繰り返すことで生成できる．

3.1 ルートノード間の隣接関係の生成

立方体を6つの四面体（鏡面対称含む）に分割する時，一般にその切り方は図3のように立方体の対角線に従って4つ存在する．Tetrahedral Adaptive Grid では，これらの切り方が最終的な結果に影響を及ぼさ

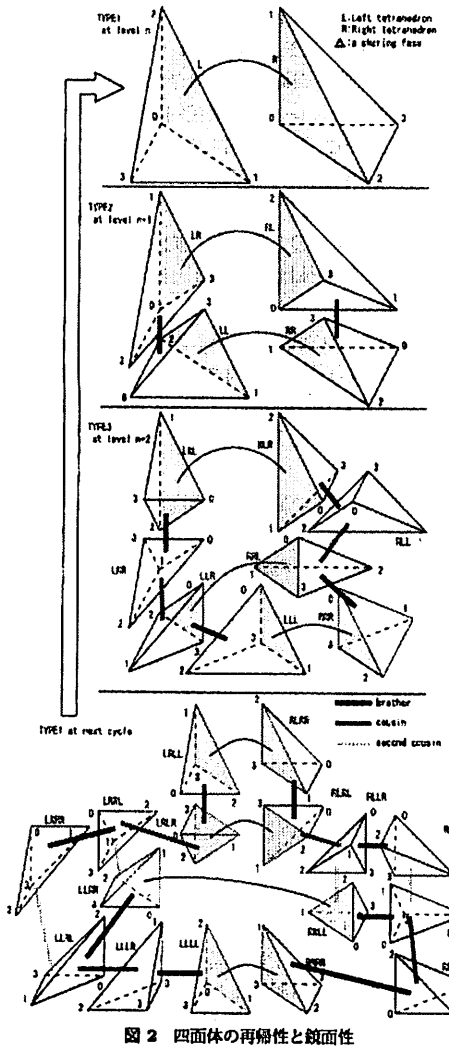


図 2　四面体の再帰性と鏡面性

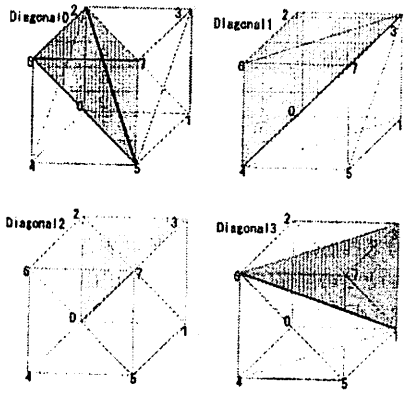


図 3　初期立方体格子の切り方

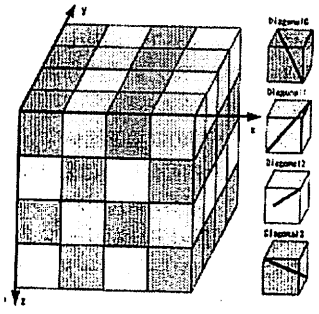


図 4　初期立方体格子

ないようにするために，初期立方体格子を図 4 のように交互に分割している．

これらの切り方で初期立方体格子を分割すると，各ルート四面体は切り方によって表 1 のように立方体の頂点 ID 用いて表すことができる．例えば，Diagonal0 の切り方で初期立方体格子を分割した場合，0 番目のルート四面体の頂点は，立方体の頂点 ID6,5,2,7 で表される．この時，偶数番目のルート四面体を Left，奇数番目のものを Right とする．

次にルート四面体間の隣接関係を表 2 を用いて生成する．四面体の 4 つの頂点を V_0，V_1，V_2，V_3，4 つの面を Face0:V_1,V_2,V_3，Face1:V_0,V_2,V_3，Face2:V_0,V_1,V_3，Face3:V_0,V_1,V_2 とすると，0 番目のルート四面体の Face0 を共有して隣接する四面体は，同じ初期立方体格子の 5 番目のルート四面体になる．同様に Face3 を共有して隣接するものは 0 番目のルート四面体になる．また，Face1，Face2 を共有して隣接するものは，いずれのルート四面体でも隣の初期立方体格子のルート四面体になる．このようにしていずれの切り方でも表 2 を用いて隣接するルート四面体を特定することができる．

表 1　ルート四面体

Root ID	Diagonal0	Diagonal1	Diagonal2	Diagonal3
0-Left	6,5,2,7	2,4,3,6	3,0,7,2	7,1,6,3
1-Right	6,2,5,4	2,3,4,0	3,7,0,1	7,6,1,5
2-Left	0,5,2,4	1,4,3,0	5,0,7,1	4,1,6,5
3-Right	0,2,5,1	1,3,4,5	5,7,0,4	4,6,1,0
4-Left	3,5,2,1	7,4,3,5	6,0,7,4	2,1,6,0
5-Right	3,2,5,7	7,3,4,6	6,7,0,2	2,6,1,3

3.2　小 2 分木内の隣接関係の生成

小 2 分木内の各四面体の隣接関係は四面体を 2 分割する過程で「兄弟」「いとこ」「はとこ」の関係にあたり，面を共有している四面体同士を結ぶことにより生

表 2 隣接ルート四面体

Root ID	Face0	Face1	Face2	Face3
0-Left	5	3	5	1
1-Right	2	4	4	0
2-Left	1	3	5	3
3-Right	4	0	2	2
4-Left	3	1	1	5
5-Right	0	0	2	4

成できる．図 2 の四面体 LLR において小 2 分木内で隣接する四面体とは，兄弟の関係にあたる四面体 LLL と，いとこの関係にあたる四面体 LRR である．

3.3 小 2 分木外の隣接関係の生成

小 2 分木外の隣接四面体同士の隣接関係は，四面体 TYPE1 の面を共有して隣接する四面体をルートノードとする小 2 分木を逆に探索することで生成できる．例えば，図 2 において TYPE3 の探索履歴 LRL に位置する四面体に隣接する四面体は，小 2 分木内では LRR（兄弟），小 2 分木外では隣接している小 2 分木で LRL を逆にした RLR という探索履歴を利用することで特定することができる．

3.4 リーフノード間の隣接四面体

適応的に表現された四面体セルは隣接するセル同士で分割レベルが異なる場合があるので，小 2 分木内と小 2 分木外の隣接関係の生成をリーフノードまで続けても探索先の四面体がリーフノードに至らないことがある．この問題を解決するために，本手法で用いた適応的四面体格子表現がクラックを生成しないことを保証していることに注目する．図 5(a) に示すように，適応的四面体格子表現にクラックが存在する場合，図 5 中の太線の四面体セルと隣接する四面体間の分割レベルの差は最大 2 である．しかし，図 5(b) のようにクラックが存在しない場合，隣接する四面体間の分割レベルの差は±1 に収まることがわかる．このことを利用すると，分割レベル 0 の時は隣接四面体同士は同レベルであり，また分割レベル差が-1 の時は隣接する四面体の探索を一つ浅いレベルまで行えばお互いにリーフノードに至るので隣接四面体が特定できたと言える．しかし，分割レベル差が+1，つまり現在の四面体がリーフノードであるが探索先の四面体がリーフノードでない場合は，さらに一つ深いレベルまで探索を行う必要がある．このような場合は探索履歴を使うことができないが，現在の四面体の TYPE，共有面の ID(FaceNo)，四面体の向き（左右）を用いて特定できる．

現在の四面体が TYPE1 で次のレベルの TYPE2 を探索したい時，共有面 ID が 1 ならば，隣接四面体は TYPE1 と同じレベルの隣接四面体を左に辿った先の四面体であり，他の共有面 ID ならば，右に辿った先の四面体である（図 6(a)）．

現在の四面体が TYPE2 で次のレベルの TYPE3 を探索したい時，親の四面体が左向きならば，隣接四面体は TYPE2 と同じレベルの隣接四面体を右に辿った先の四面体であり，親の四面体が右向きならば，逆の左に辿った先の四面体である（図 6(b)）．

現在の四面体が TYPE3 で次のレベルの TYPE1 を探索したい時，現在の四面体が左向きで共有面 ID が 1 ならば，隣接四面体は TYPE3 と同じレベルの隣接四面体を左に辿った先の四面体に，他の共有面 ID ならば右に辿った先の四面体である．また，現在の四面体が右向きで共有面 ID が 2 ならば，隣接四面体は TYPE3 と同じレベルの隣接四面体を右に辿った先の四面体に，他の共有面 ID ならば左に辿った先の四面体である（図 6(c)）．

リーフノードにおいて隣接する四面体間の分割レベルに差があるときの手順を **Procedure_Find_Tetrahedron_AtNextLv** に示す．

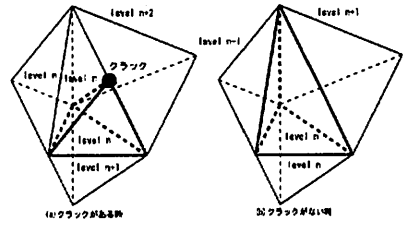


図 5 分割レベルの差

3.5 アルゴリズム

以上の操作を隣接四面体同士が 2 分木のリーフノードになるまで続けることにより Dual Grid を生成することができる．本アルゴリズムは，**Procedure_Dual_Grid_Generation** 内で，**Procedure_Binary_Search_For_Dual_Grid** を再帰的に呼び出すことにより実装している．

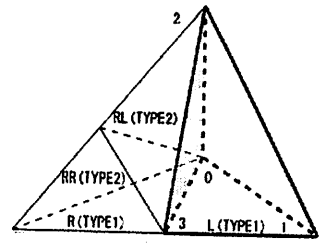
4. 実験結果

実験環境として用意した分散メモリ方式並列計算機の PC クラスタは，OS が Redhat Linux，計算ノードとして Intel Pentium4 2.8GHz の CPU と 2GB のメインメモリを持つ計算機を 32 台，ファイルサーバーとして Dual Intel Xeon 2.8 GHz の CPU と 1GB のメインメモリを持つ計算機 1 台より構成される．開発言語に C++，表示用のグラフィックライブラリに VTK4.0，並列処理プログラムに MPI(Message Pass-

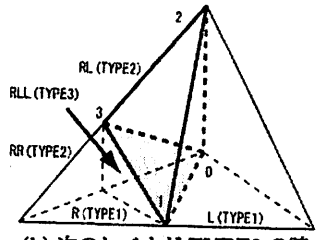
```
Procedure_Find_Tetrahedron_AtNextLv(T_c, i)
T_c:a tetrahedron cell of a node of binary tree
T_n(i):a neighbor tetrahedron cell of T_c sharing i_th
face
i:face number
begin
  if T_n(i) is NOT LEAF then
    set temporal T_n(i) to neighbor cell of T_c
                                    at same level.
   (* Estimage cell type of T_c *)
    (* When a cell type of T_c is TYPE1 *)
     if face number is 1 then
       get LEFT CHILD of temporal T_n(i);
     else
       get RIGHT CHILD of temporal T_n(i).
    (* When a cell type of T_c is TYPE2*)
     if a PARENT tetrahedron is LEFT then
       get RIGHT CHILD of temporal T_n(i);
     else
       get LEFT CHILD of temporal T_n(i).
    (* When a cell type of T_c is TYPE3*)
     if a CURRENT tetrahedron is LEFT and
      if face number is 1 then
       get LEFT CHILD of temporal T_n(i);
      else
       get RIGHT CHILD of temporal T_n(i).
     else
      if face number is 2 then
       get RIGHT CHILD of temporal T_n(i);
      else
       get LEFT CHILD of temporal T_n(i).
end
```

ing Interface) を使用した．入力データには大きさが 161 × 321 × 129 で 8bit の諧調値を持つ足のデータを用意した．

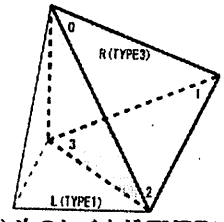
適応的四面体格子表現生成法により原データの全ボクセル数（6,666,849 個）を約 1/10（684,940 個）に減らし，この時の四面体セル集合（7,786,677 セル）の Dual Grid を約 4 秒で生成した．さらに，Dual Grid を付加した適応的四面体格子表現を用いてレイキャスティングを行い，最大値投影法（Maximum Intensity Projection）によってレンダリングした結果を図 8 に示す．原データを最大値投影法によってレンダリングした結果である図 7 と比べて，約 1/10 のボクセル数でレンダリングしても十分に足の形状を残した結果を



(a) 次のレベルが TYPE2 の時



(b) 次のレベルが TYPE3 の時



(c) 次のレベルが TYPE1 の時

図 6 リーフノードの処理

```
Procedure_Dual_Grid_Generation()
begin
  (* Initialize *)
    Calcurate vertices of dual grids
    Geneate dual grids between ROOT tetrahedra
  (* Dual Grids Generation *)
    Binary_Search_For_Dual_Grid(all binary trees)
end
```

生成することができると言える．

5. おわりに

本論文では，適応的四面体格子表現が持つ 2 つの性質，「再帰性」と「鏡面性」，に注目した Dual Grid 生成法を提案した．ボリューム空間を適応的に四面体分割することで得られる四面体セル間の隣接関係を新しく生成することにより，直線と交差する四面体セルを抽出することが容易となり，適応的な四面体格子表現を持つボリュームデータにおいてもレイキャスティングが可能である．今後は，レイキャスティングと同時

Procedure_Binary_Search_For_Dual_Grid
(Tree, histories)

```
Tree:a pointer of binary tree
histories:3 bits for recording a way in binary tree
begin
  (* generate dual grids outer subtree *)
   if Tree is NOT ROOT
  (* generate dual grids inner subtree *)
   if a type of tetrahedron cell is TYPE0
  (* Recursive call *)
   if Tree is NOT LEAF
      histories[cell type -1] ← LEFT
      Binary_Search_For_Dual_Grid
                        (LEFT of Tree, histo-
ries);
      histories[cell type -1] ← RIGHT
      Binary_Search_For_Dual_Grid
                        (RIGHT of Tree, histo-
ries).
   else
      while (face number ≦ 4)
       Find_Tetrahedron_AtNextLv
                        (Tree, face number)
    Binary_Search_For_Dual_Grid
end
```

に不透明度も計算することで，適応的な四面体格子構造を持つボリュームデータのレンダリングを行い，原データに対する画質の評価を行う予定である．

参 考 文 献

1) G.M. Nielson: Volume Modelling, *Volume Graphics*, Springer, pp.29–48(2000).
2) C.H. Chien and J.K.Aggarwal: VolumeSurface Octrees for the Representation of Three-Dimensional Objects, *Computer Vision Graphics and Image Processing*, Vol.36, pp.100-113(1986).
3) Y Zhou, B Chen and A Kaufman: Multiresolution Tetrahedral Framework for Visualising Regular Volume Data, *Proc.IEEE Visualization '97*, pp.135-142(1997).
4) J. Bey: Tetrahedral mesh refinement, *Computing*, Vol.55, No.13, pp.355-378(1995).
5) JM.Maubach: Local bisection refinement for N-simplicial grids generated by reflection, *SIAM Journal of Scientific Computing*, Vol.16, No.1, pp.210-227(1995).

図 7 従来法

図 8 提案手法

6) Y. Takama, A. Kimura and H.T. Tanaka: " Tetrahedral Adaptive Grid for Parallel Hierarchical Tetrahedrization ", *7th Eurographics Workshop on Multimedia 04*, Nanjing(2004).
7) P. Shirley and A. Tuchman: " A polygonal approximation to direct scalar volume rendering", *In 1990 Workshop on Volume Visualization"*, pp.63-70, San Diego, CA, USA(1990).
8) Davis King, Craig M. Wittenbrink and Hans J. Wolters: "An Architecture for Interactive Tetrahedral Volume Rendering", *Volume Graphics 2001*, June 21-22, Stony Brook, New York, USA(2001).
9) M. Levoy:"Efficient Ray Tracing of Volume Data", *ACM Transactions on Graphics*, Vol.9(3), pp.245-261, July(1990).
10) M. Weiler, M. Kraus, M.Merz and T. Ertl:"Hardware-Based Ray Casting for Tetrahedral Meshes", *In Proceedings of IEEE Visualization '03*, pp.333-340, Seattle, Washington, USA(2003).
11) H. Samet:"Neighbor Finding Techniques for Image Represented by Quadtrees", *Int. Journal Computer Graphics and Image Processing*, Vol.18(1), pp.37-57, Jan(1982).
12) H. Samet:"Neighbor Finding in Images Represented by Octree", *Int. Journal Computer Vision, Graphics and Image Processing*, Vol.46(6), pp.367-386, June(1989).
13) M. Lee, L. De Floriani and H. Samet:"Constant-Time Neighbor Finding in Hierarchical Tetrahedral Meshes", *IEEE Int. Conf. on Shape Modelling 2001*, pp.286-295, Genova, Italy(2001).