# 等値面生成のための高速ポリゴン構築方法

伊藤貴之[†] 山口 泰[††] 小山田 耕二[†††]

等値面は，ボリュームデータ中のスカラ場で同じ値をもつ点の集合であり，一般的にはポリゴンの集合で近似表現される．ここで，等値面のポリゴン頂点の大半は，複数のポリゴンに共有されている．そのため，等値面のポリゴン構築過程では，ポリゴン頂点を検索する処理が必要である．この検索処理は，ポリゴン構築過程の中でも大きな計算量を占めている．

本論文では，ポリゴン頂点の検索処理を不要にすることで，高速に等値面のポリゴンを構築する手法を提案する．本手法では，ポリゴン頂点を生成した際に，ポリゴン頂点が接する格子辺を共有するすべての格子を同時に探索し，それらの格子の内部にあるポリゴンにポリゴン頂点を登録する．このポリゴン頂点は，ポリゴン構築の過程で再び必要になることがない．よって本手法では，ポリゴン頂点の検索処理が不要である．筆者らの実装では，本手法によって約 20 パーセントの処理速度の向上を実現することができた．

# A Fast Polygon Construction Method for Isosurface Generation

Takayuki Itoh,[†] Yasushi Yamaguchi[††] and Koji Koyamada [†††]

An isosurface is a set of points where the same value lies. That is usually approximated as a set of polygons. Here, most of polygon-vertices of an isosurfaces are shared by several polygons. Therefore, the polygon-vertex identification process which searches for the polygon-vertex generated at the same position is necessary in isosurfacing methods. The identification process generally occupies the largest part of the computational time in constructing polygons.

This paper proposes an efficient polygon construction method for isosurface generation. The method does not require the polygon identification process, since the method processes all cells adjacent to a cell-edge which a polygon-vertices lies at the same time. In the authors' implementation, the method is about 20 percent faster than the conventional isosurfacing methods.

## 1. はじめに

等値面生成は，ボリュームデータに分布するスカラ場を視覚化する最も効果的な方法のひとつであり，例えば科学技術計算結果や医療測定結果などの視覚化の分野で広く実用化されている．これらの分野では，ボリュームデータの大規模化の傾向が続いているため，等値面生成の高速化手法に関する議論も依然として活発である．

等値面とは，ボリュームデータに与えられたスカラ場の値を $S(x, y, z)$ としたときに，$S(x, y, z) = S$ ($S$ は定数) を満たす点の集合である．等値面生成の代表的な手法[1),2)] では，以下の３つの処理によって，ポリゴンの集合で近似された等値面を生成する．

**処理 1:** ボリュームデータ中の各格子について，格子点 $N_i$ のもつスカラ値 $S_i$ と，与えられた定数 $S$ の大小比較をする．$S_i > S$ である格子点と $S_i < S$ である格子点の両方が存在するとき，その格子は等値面と交差すると判定される．

**処理 2:** 等値面と交差すると判定された格子について，その交差部分をポリゴンで近似する．

**処理 3:** 各ポリゴン頂点における座標値，および必要があれば法線ベクトルを算出する．

ここで，等値面と交差する格子は，ボリュームデータ中のごく一部であることが多い．このことから近年では，等値面と交差しない格子の多くに対する処理を省略する，高速な等値面生成手法[3)~11)] が多く報告された．これらの手法によって，上記の **処理 1** に要する時間は非常に小さくなった．表 1 は，四面体格子で

† 日本アイ・ビー・エム (株) 東京基礎研究所
IBM Reserach, Tokyo Research Laboratory
†† 東京大学大学院総合文化研究科
Graduate School of Arts and Sciences, University of Tokyo
††† 岩手県立大学ソフトウェア情報学部
Faculty of Software and Information Science, Iwate Prefectual University

構成されるボリュームデータに対して，スカラ値を変えながら 20 枚の等値面を生成したときの計算時間の測定例を示している．この計算時間測定では，全ての格子に対して等値面との交差を判定する単純なアルゴリズムと，筆者らが報告しているボリューム細線化手法[11] を用いたアルゴリズムを比較している．ここで，

- $n_c$ および $n_n$ は，ボリュームデータを構成する格子数および格子点数を示している．
- $n_t$ および $n_v$ は，20 枚の等値面の三角形ポリゴン数の合計，およびポリゴン頂点の合計を示している．
- $t_1$ は，**処理 1** (等値面と交差する格子を探し出す処理) の計算時間を示している．
- $t_2$ は，**処理 2** (等値面と交差する格子の内部に生成されるポリゴンの位相を構築する処理) の計算時間を示している．
- $t_3$ は，**処理 3** (等値面のポリゴン頂点の座標値および法線方向を算出する処理) の計算時間を示している．
- $t_{total}$ は，等値面生成に要する計算時間の合計を示している．

**表 1** 等値面生成中の処理時間

| ボリューム | 1 | 1 | 2 | 2 |
|---|---|---|---|---|
| 手法 | 全格子判定 | ボリューム細線化 | 全格子判定 | ボリューム細線化 |
| $n_c$ | 61680 | | 346644 | |
| $n_n$ | 11624 | | 62107 | |
| $n_t$ | 80995 | | 135358 | |
| $n_v$ | 43158 | | 71358 | |
| $t_1$ (sec.) | 8.30 | 0.09 | 42.23 | 0.57 |
| $t_2$ (sec.) | 3.80 | 3.45 | 6.25 | 5.88 |
| $t_3$ (sec.) | 0.76 | 0.75 | 1.14 | 1.15 |
| $t_{total}$ (sec.) | 12.86 | 4.29 | 49.62 | 7.60 |

表 1 から，上記のボリューム細線化手法が **処理 1** を微々たる計算時間にまで低減することに成功し，その結果として等値面生成の計算時間を大幅に低減していることがわかる．ボリューム細線化手法以外の従来の高速化手法を用いても，おそらく表 1 に近い結果が得られるものと思われる．逆に言えば，さらに等値面生成を高速化するためには，**処理 1** 以外の処理，とりわけ **処理 2** の高速化を議論する必要がある．

一般に，等値面を構成するポリゴン頂点の大半は，複数のポリゴンに共有されている．そのため，**処理 2** の過程で生成されるポリゴンについて，そのポリゴン頂点を共有する隣接ポリゴンがすでに生成されているか確認し，すでに隣接ポリゴンが生成されていればそのポリゴン頂点を検索して共有するように実装するのが一般的である．ポリゴン頂点を共有しないように実装すること自体は可能であるが，その場合にはポリゴン頂点のメモリ使用量は約 6 倍となる．その結果として，等値面のメモリ使用総量が 3 倍程度に増加する．また，ポリゴン頂点を共有しないことにすれば，ポリゴン頂点の検索が不要になるので，$t_2$ が小さくなることが期待できるが，逆に $t_3$ が約 6 倍になってしまうので，等値面生成に要する計算時間が増加することは明らかである．また，ポリゴンデータの単純化手法や圧縮手法と等値面生成手法を併用する場合には，ポリゴン頂点の共有は不可欠である．これらのことから，等値面生成技術には，ポリゴン頂点を共有させる処理が不可欠であると言える．

筆者らの従来の実装では，ポリゴン頂点の検索にハッシュ・テーブルを用いて高速化を図っている[12]．しかしそれでも，ポリゴン頂点の検索は **処理 2** の中でも多くの計算時間を費やしている．

本論文では，ポリゴン頂点の検索処理を用いずに，隣接ポリゴン間のポリゴン頂点の共有を実現する，効率的な等値面生成手法を提案する．本手法では，等値面と交差する格子の内部におけるポリゴンを生成したときに，そのポリゴンのポリゴン頂点が接する格子辺について，格子辺を共有するすべての格子を探索する．そして，探索した格子の内部に生成されるポリゴンに対して，同じポリゴン頂点を登録する．このアルゴリズムによって，ひとつのポリゴン頂点を共有するすべてのポリゴンに，そのポリゴン頂点が同時に登録される．よって，そのポリゴン頂点を後になって検索する必要がない．

なお筆者らは，本論文の内容の一部を，すでに研究会で報告している[13),14)]．本論文では，これらの研究会原稿の内容を詳細化し，さらに既に発表しているボリューム細線化手法[11] と組み合わせたアルゴリズムを示している．

## 2. 関 連 技 術

### 2.1 等値面のポリゴン頂点の検索

等値面のポリゴン生成の典型的な手法である Marching Cube 法[2] では，すべてのポリゴンは 1 個の格子の内部に構築され，ポリゴン頂点は格子辺上に配置される．大半の格子辺は複数の隣接格子に共有されるので，その上に生成されるポリゴン頂点は複数のポリゴンに共有される．

もしボリュームデータが，格子辺のデータを保持するようなデータ構造で記述されていれば，ポリゴン頂

点を格子辺ごとに登録することができるので，ポリゴン頂点の検索は不要である．しかし，ボリュームデータのデータ構造は，メモリ使用量の観点から，格子点および格子だけを保持し，格子辺を保持しないのが一般的である．そのため，すでに登録されたポリゴン頂点を共有するポリゴンを後から生成する際には，そのポリゴン頂点を検索する処理が必要になる．

複数のポリゴンに共有されるポリゴン頂点を検索する方法については，すでにいくつかの論文[7),12),15)]で議論されている．筆者らは，ハッシュ・テーブルを用いてポリゴン頂点の検索を高速化する手法[12)]を実装している．この手法では，ポリゴン頂点が生成されるときに，ポリゴン頂点が接する格子辺を，ポリゴン頂点と一緒にハッシュ・テーブルに登録する．ここで，一般的にボリュームデータは格子の集合および格子点の集合で表現され，格子辺の集合は保持されていないことが多い．よってこの手法では，格子辺は両端の格子点を用いて表現されている．

図 1 は，ポリゴン頂点の検索処理の一例を示したものである．この例では，ポリゴン $P_1$ を生成したときに，そのポリゴン頂点 $V_a, V_b, V_c, V_d$ をハッシュ・テーブルに登録している．この時，例えば $V_b$ は格子点 $(N_1, N_2)$ を端点とする格子辺上にあるので，この格子点とポリゴン頂点を一緒にハッシュ・テーブルに登録する．続いて，$P_1$ に接するポリゴン $P_2$ を生成する時に，$P_2$ のポリゴン頂点が接する格子辺がすでにハッシュ・テーブルに登録されているかを検索する．この時，格子点の組 $(N_1, N_2)$ がすでに登録されているので，$(N_1, N_2)$ と一緒に登録されているポリゴン頂点 $V_b$ を抽出して，$P_2$ のポリゴン頂点として用いる．

上記の手法では，ハッシュ・テーブルを用いることで，すでに登録されている格子辺（＝2個の格子点）を高速に検索する．しかしそれでも，筆者らの測定では，この検索に多くの計算時間を費やしていることがわかっている．

### 2.2 自己増殖的な等値面生成手法

等値面に交差する格子は互いに隣接している．そこで，等値面に交差する格子が1つでも抽出されれば，その格子に隣接する交差格子を再帰的に探索できる．この手順により，等値面と交差しない格子との処理を省きながら，自己増殖的に等値面を生成する (Isosurface propagation) ことができる[1),16),17)]．筆者らはすでにこの手法を実装して，関連研究[10),11)]に用いている．

筆者らの実装では，幅優先探索によって隣接交差格子を処理している．まず，あらかじめ抽出された交差格子を FIFO に登録する．続いて，FIFO から格子を

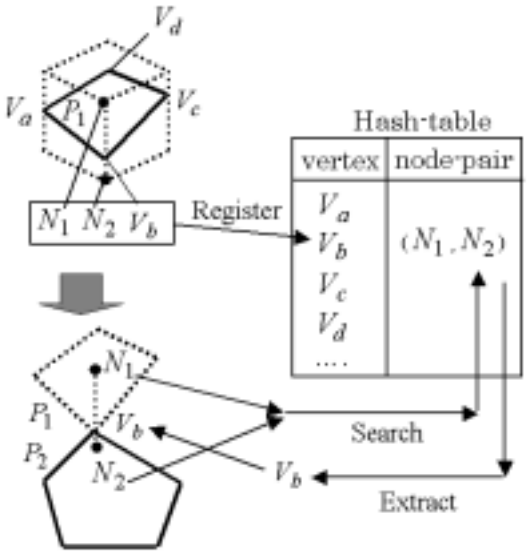



**図 1** ポリゴン頂点の検索処理
Fig. 1 Polygon-vertex search process.

1個ずつ抽出し，格子内部のポリゴンを生成するとともに，隣接交差格子の中でまだ FIFO に登録されていないものがあれば登録する．以上の処理を FIFO が空になるまで反復することで，等値面が生成される．

図 2 は，筆者らの実装を図示したものである．この実装では，ポリゴン $P_1$ を最初に生成したときに，隣接する4個の交差格子 $C_2, C_3, C_4, C_5$ を FIFO に登録する．続いて，これらの格子を FIFO から順に抽出し，ポリゴン $P_2, P_3, P_4, P_5$ を生成する．$P_2$ に隣接する交差格子 $C_1, C_6, C_7, C_8$ のうち，$C_1$ はすでに FIFO に登録されたことがあるので，$P_2$ の生成時には $C_6, C_7, C_8$ の3格子を FIFO に登録する．そして，これらの格子を FIFO から抽出したときに，ポリゴン $P_6, P_7, P_8$ を生成する．

自己増殖的な等値面生成手法では，あらかじめ最低1個の交差格子が抽出されている必要がある．特に，等値面が複数の非連結な部位で構成される場合には，すべての部位において交差格子が最低1個抽出されている必要がある．

筆者らは，複数の非連結な部位で構成される等値面に対して，すべての部位に最低1個の交差格子を自動抽出する手法を提案している[10),11)]．この手法ではまず，スカラ値が極小または極大となる格子 (Extrema) をすべて抽出する．続いて，これらの格子を連結する骨格線 (Extrema skeleton) を，格子の集合として形成する．この骨格線は，ボリュームデータから抽出できるすべての等値面と交差する，という特徴をもつ．

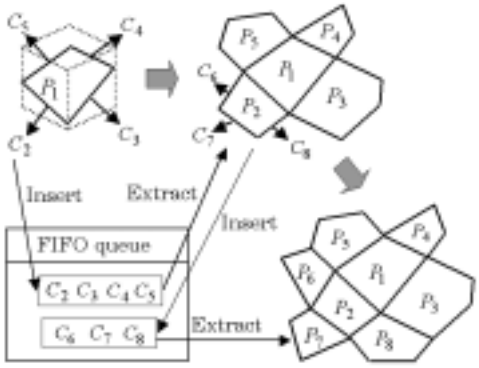



**図 2** 自己増殖的な等値面生成手法
Fig. 2 Isosurface propagation.

よって，骨格線を構成する格子を探索することにより，等値面のすべての非連結な部位に対して，等値面と交差する格子 (Isosurface cell) を最低１個抽出することができる．抽出した格子を出発点にして，それ以外の交差格子を再帰的に探索することにより，自己増殖的に等値面を生成することができる．

この手法を２次元で図示したものを，図 3 に示す．図 3(a) に示すボリュームデータ中で，黒い四角形で表現した格子が，スカラ値が極大または極小となる格子の例である．図 3(b) に示す白い四角形および黒い四角形が，骨格線を構成する格子の例である．図 3(c) に示す灰色の四角形が，骨格線を構成する格子を探索して発見された交差格子の例である．図 3(d) に示す灰色の四角形が，骨格線から発見された交差格子を出発点にして探索された交差格子の例である．この図からわかるように，この手法では，骨格線を構成する格子と，交差格子だけを処理し，図 3(d) に図示されてない格子との処理をすべて省略している．その結果として，1 章で説明した **処理 1** の計算時間を大幅に低減し，等値面生成の大幅な高速化を実現している．

等値面生成の高速化手法の大半[3)~10)] には，前処理の計算時間が $O(n_c)$ より大きくなるという問題点がある．ここで $n_c$ は，ボリュームデータの格子の総数を示す．それに対して，ボリューム細線化手法 (Volume thinning) を用いた手法[11)] では，等値面生成の前処理と位置づけられる「骨格線生成」の計算時間が $O(n_c)$ 以下である，という利点がある．

## 3. ポリゴン頂点の検索が不要なポリゴン構築手法

本論文では，等値面を構成する複数のポリゴンに共有されるポリゴン頂点を，検索処理を用いずに共有させることで，高速に等値面のポリゴンを構築する方法を提案する．また，ボリューム細線化手法と組み合わせた実装方法を示し，１章で説明した **処理１** と **処理２** の両方の高速化を実現する．

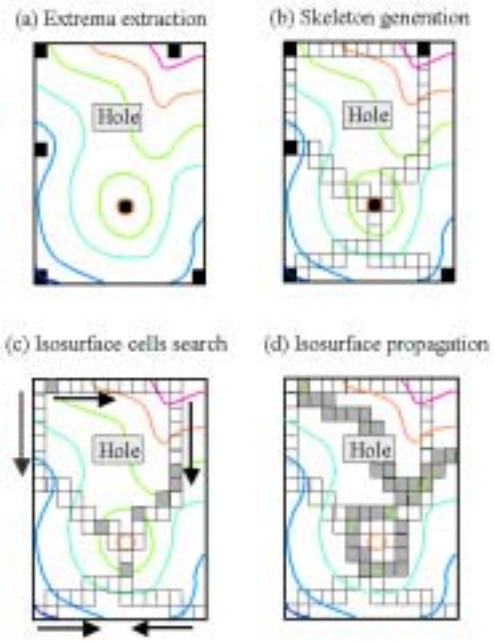



**図 3** ボリューム細線化手法を用いた等値面生成
Fig. 3 Isosurface generation with vokume thinning method.

本論文で提案する手法は，１個のポリゴン頂点に接する格子辺を共有するすべての格子を同時に処理する．これによって，ポリゴン頂点を共有するすべてのポリゴンに対して，そのポリゴン頂点を同時に登録することができるので，後になって再びそのポリゴン頂点が必要になることがない．よって，すでに登録されたポリゴン頂点を検索することなく等値面ポリゴンを構築することができる．

本論文で提案する手法は，等値面と交差する格子を隣接順に処理するという点において，前章で紹介した自己増殖的な等値面生成手法[1),16),17)] に類似している．しかし，従来の自己増殖的な等値面生成手法では，１個のポリゴン頂点を共有するすべてのポリゴンを同時に処理することができないので，依然としてポリゴン頂点の検索処理が必要である．本論文で提案する手法は，ポリゴン頂点の検索処理を不要にすることで，１章で説明した **処理２** の高速化を実現したという点に意義がある．

### 3.1 アルゴリズムの概要

本論文で提案する手法の概要を図 4 に示す．このア

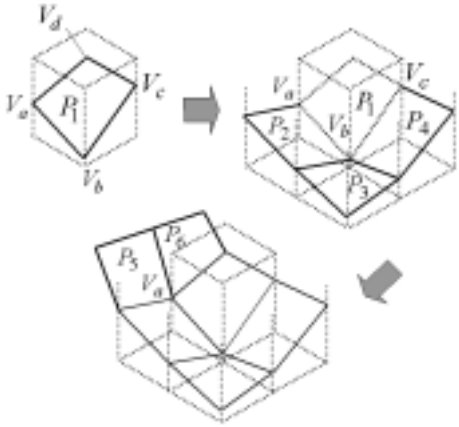



**図 4** アルゴリズムの概要
Fig. 4 Algorithm overview.

ルゴリズムでは，まず等値面と交差する格子 $C_1$ が与えられたとして，$C_1$ の内部に Marching Cube 法[2] または同等な手法を用いてポリゴン $P_1$ を生成する．図 4 では，$P_1$ を生成した際に，ポリゴン頂点 $V_a, V_b, V_c, V_d$ も同時に生成している．続いて，生成された各々のポリゴン頂点について，そのポリゴン頂点に接する格子辺を共有するすべての格子を探索する．そして，探索した格子の内部にまだポリゴンが生成されていないときには，ポリゴンを生成する．図 4 では，ポリゴン頂点 $V_b$ を共有する格子を探索して，ポリゴン $P_2, P_3, P_4$ を生成し，それらのポリゴンに $V_b$ を登録している．同様に，ポリゴン頂点 $V_a$ を共有する格子を探索してポリゴン $P_5, P_6$ を生成し，ポリゴン $P_2, P_5, P_6$ に $V_a$ を登録している．以上の処理によって，ポリゴン頂点 $V_a$ および $V_b$ を共有するすべてのポリゴンに，$V_a$ および $V_b$ が一気に登録されるので，ポリゴン構築処理の途中で再び $V_a$ や $V_b$ が必要とされることがまったくない．よって，以前に生成されたポリゴン頂点を検索する必要がないので，ポリゴン頂点の検索処理を用いずにポリゴン頂点の共有を実現することができる．

### 3.2 ボリューム細線化手法と組み合わせたアルゴリズム

前述の通り，本論文で提案するポリゴン構築手法は，従来の自己増殖的な等値面生成手法と同様に，等値面と交差する格子が最低１個与えられていることを前提としている．そのため本手法は，等値面と交差するいくつかの格子を自動抽出する手法と組み合わせて使う必要がある．筆者らは，等値面と交差する格子の自動抽出のために，ボリューム細線化手法を用いた手法[11] を用いている．この手法は，等値面のすべての非連結な部位に対して最低１個の交差格子を抽出することができるので，それらの格子を出発点にすることで，本論文で提案するポリゴン構築手法を実行することができる．

ボリューム細線化手法と組み合わせたポリゴン構築方法の擬似コードを，図 5 に示す．本手法では，ボリューム細線化手法を１回だけ実行して骨格を生成し，その骨格をプログラムが終了するまで利用して高速に等値面を生成する．スカラ値を変えながら多数の等値面を生成したいときに，本手法は特に有用であるといえる．

```
void Isosurfacing() {
  for(each cell Ci in an extrema skeleton) {
    if(Ci is an isosurface cell) {
      insert Ci into FIFO;
    }
  }

  /* for-loop (1) */
  for (each cell Ci extracted from FIFO) {
    if(polygon Pi in Ci is not constructed) {
      Construct Pi in Ci;
    }
    /* for-loop (2) */
    for(each isosurface edge En) {
      if(a polygon-vertex Vn on En
        is not registered into Pi ) {
        Allocate Vn on En;
        Register Vn into Pi;
        /* for-loop (3) */
        for(each cell Cj which shares En) {
          if(Pj in Cj is not constructed) {
            Construct Pj in Cj;
            Insert Cj into FIFO;
          }
          Register Vn into Pj;
        } /* for(each Cj) */
      } /* if(there is not Vn) */
    } /* for(each En) */
  } /* for(each Ci) */

  for(each polygon-vertex V) {
    Calculate position and normal vector;
  }
} /* end Isosurfacing() */

void main() {
  Generate an extrema skeleton by volume thinning;
  while( 1 ) {
    Specify the isovalue C;
    isosurfacing();
  }
} /* end main() */
```

**図 5** ボリューム細線化手法と組み合わせた本手法の擬似コード
Fig. 5 Pseudo code with the volume thinning method.

スカラ値が指定されたときに，本手法はまず，骨格を構成する格子の中から等値面に交差する格子を抽出し，FIFO キューに登録する．続いて，FIFO から格子 $C_i$ を抽出し，その内部にポリゴンがまだ生成されていなければポリゴン $P_i$ を生成する．続いて，$P_i$ のポリゴン頂点 $V_n$ が接する各々の格子辺 $E_n$ について，$E_n$ を共有する格子 $C_j$ を順に探索し，$C_j$ の内部にあるポリゴン $P_j$ に $V_n$ を登録する．もしこの時，$C_j$ の内部にポリゴン $P_j$ が生成されてなければ，$P_j$ を生成してから，$V_n$ を $P_j$ に登録する．このとき同時に，$C_j$ を FIFO に登録する．ここまでの処理を終えたら，FIFO から別の格子を抽出して，同様な処理を反復する．そして，FIFO が空になるまで格子を抽出して同様な処理を反復することで，等値面のポリゴン構築の過程を完了する．最後に，ポリゴン頂点の座標値および法線方向を算出して，等値面を完成する．座標値および法線方向の算出方法は，Marching Cube 法[2] を始めとする既存の等値面生成手法の通りである．

本論文で提案するポリゴン構築方法は，ポリゴン頂点の検索を必要としないが，そのかわりに等値面と交差する格子を複数回探索する．大半の格子は，まず格子辺を共有する格子の探索（図 5 の for-loop(3)）によって数回処理され，その後に FIFO からの抽出（図 5 の for-loop(1)）によって１回処理される．しかしそれでも，次章に示す計算時間測定結果からわかるように，本手法は従来のポリゴン構築手法よりも高速であることが言える．

## 4. 実 行 例

本手法を IBM PowerStation RS/6000 (Model 560) で実装し，計算時間を測定した結果を示す．本章で示す結果は，有限要素法などの数値解析によって得られた，四面体格子で構成される４種類の非構造ボリュームを用いて測定したものである．

図 6 は，ボリューム細線化手法によって生成された骨格線を示している．また図 7 は，骨格線を利用して生成された数枚の等値面を示している．いずれの画像も，ボリューム細線化手法を提案した論文[11] にカラー画像として掲載されている．

表 2 は，スカラ値を変えながら 20 枚の等値面を生成したときの計算時間を示したものである．ここで，$n_c$ および $n_n$ はボリュームの格子数および格子点数を示し，$n_t$ および $n_v$ は 20 枚の等値面のポリゴン数およびポリゴン頂点数を示している．$t_{旧}$ は従来の自己増殖的な等値面生成手法に要した計算時間，$t_{新}$ は本手法による計算時間を示している．$t_{p\,旧}$ および $t_{p\,新}$ は，両手法において，ポリゴンの生成に要した計算時間を示している．

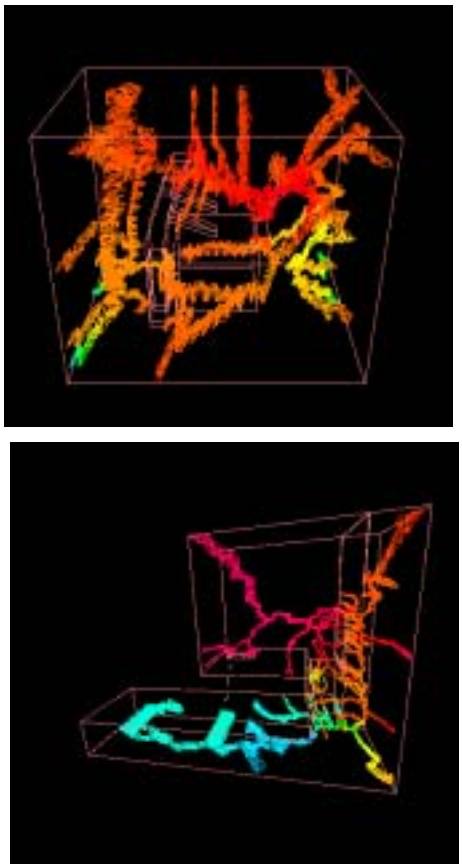


**図 6** 骨格線の例
Fig. 6 Examples of an extrema skeleton.

**表 2** 等値面生成に要する処理時間

| ボリューム | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| $n_c$ | 61680 | 346644 | 458664 | 557868 |
| $n_n$ | 11624 | 62107 | 80468 | 97473 |
| $n_t$ | 80995 | 135398 | 494480 | 1164616 |
| $n_v$ | 43158 | 71358 | 251506 | 588796 |
| $t_{p\,旧}$ (sec.) | 3.45 | 5.88 | 21.35 | 49.80 |
| $t_{p\,新}$ (sec.) | 2.53 | 4.01 | 15.72 | 36.20 |
| $t_{旧}$ (sec.) | 4.29 | 7.60 | 26.65 | 60.81 |
| $t_{新}$ (sec.) | 3.35 | 5.52 | 20.61 | 46.80 |

この結果から，本手法ではポリゴン生成処理において約 25 パーセント，等値面生成処理全体においても約 20 パーセントの処理速度向上を実現したことがわかる．

## 5. む す び

本論文では，ポリゴン頂点の検索処理を不要にする

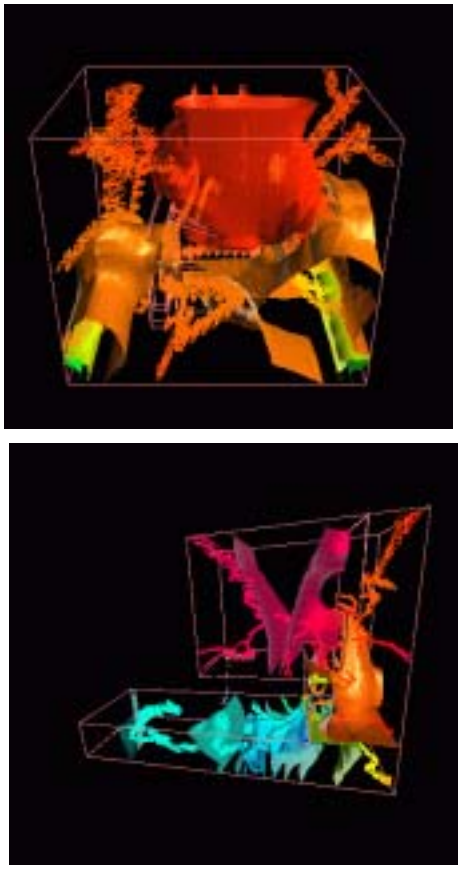


**図 7** 等値面の例
Fig. 7 Examples of isosurfaces.

ことで，高速に等値面のポリゴンを構築する手法を提案した．本手法では，ポリゴン頂点を生成した際に，ポリゴン頂点が接する格子辺を共有するすべての格子を同時に探索し，それらの格子の内部にあるポリゴンにポリゴン頂点を登録する．等値面のポリゴン構築の過程において，そのポリゴン頂点が再び必要になることがないので，本手法ではポリゴン頂点の検索処理が不要である．

また本論文では，すでに報告しているボリューム細線化手法と組み合わせたアルゴリズムを示した．実行例では，本手法がポリゴン構築処理の計算時間を 25 パーセント程度短縮し，等値面生成処理全体の計算時間も 20 パーセント程度短縮していることを示した．

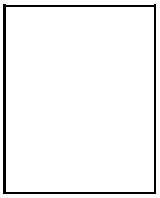
本論文の実行例では，四面体格子で構成するボリュームデータを対象にして実装および計算時間の測定を行ったが，原理的には六面体格子で構成されるボリュームデータにも同様に実装が可能である．そこで今後の課題として，六面体格子で構成されるボリュームデータに対して本手法を実装し，計算時間を測定することがあげられる．

## 参 考 文 献


1) Wyvill G., McPheeters C., and Wyvill B., Data Structure for Soft Objects, The Visual Computer, Vol. 2, No. 4, pp. 227-234, 1986.
2) Lorensen W. E., and Cline H. E., Marching Cubes: A High Resolution 3D Surface Construction Algorithm, Computer Graphics, Vol. 21, No. 4, pp. 163-169, 1987.
3) Giles M., and Haimes R., Advanced Interactive Visualization for CFD, Computer Systems in Engineering, Vol. 1, No. 1, pp. 51-62, 1990.
4) Gallagher R. S., Span Filtering: An Optimization Scheme for Volume Visualization of Large Finite Element Models, Proceedings of IEEE Visualization '91, pp. 68-74, 1991.
5) Livnat Y., Shen H., and Johnson C. R., A Near Optimal Isosurface Extraction Algorithm Using the Span Space, IEEE Transactions on Visualization and Computer Graphics, Vol. 2, No. 1, pp. 73-84, 1996.
6) Shen H., Hansen C. D., Livnat Y., and Johnson C. R., Isosurfacing in Span Space with Utmost Efficiency (ISSUE), Proceedings of IEEE Visualization '96, pp. 287-294, 1996.
7) Cignoni P., Marino P., Montani C., Puppo E., and Scopigno R., Speeding Up Isosurface Extraction Using Interval Trees, IEEE Transactions on Visualization and Computer Graphics, Vol. 3, No. 2, pp. 158-170, 1997.
8) Welhelms J., and Gelder A. Van, Octrees for Fast Isosurface Generation, ACM Transactions on Graphics, Vol. 11, No. 3, pp. 201-227, 1992.
9) Silver D., and Zabusky N. J., Quantifying Visualization for Reduced Modeling in Nonlinear Science: Extracting Structures from Data Sets, Journal of Visual Communication and Image Representation, Vol. 4, No. 1, pp. 46-61, 1993.
10) Itoh T., and Koyamada K., Automatic Isosurface Propagation by Using an Extrema Graph and Sorted Boundary Cell Lists, IEEE Transactions on Visualization and Computer Graphics, Vol. 1, No. 4, pp. 319-327, 1995.
11) Itoh T., Yamaguchi Y., and Koyamada K., Fast Isosurface Generation Using the Volume Thinning Algorithm, IEEE Transactions on Visualization and Computer Graphics, accepted.
12) Doi A., and Koide A., An Efficient Method of Triangulating Equi-valued Surfaces by Using Tetrahedral Cells, IEICE Transactions, Vol. E74, No. 1, pp. 214-224, 1991.
13) 伊藤, 山口, 小山田, ポリゴン頂点の検索処理の不要な高速等値面生成手法, 情報処理学会グラフィク

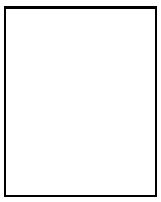
スと CAD 研究報告, 97-CG-86, pp. 23-28, 1997.
14) Itoh T., Yamaguchi Y., and Koyamada K., Fast Isosurface Generation Using the Cell-Edge Centered Propagation Algorithm, International Symposium on High Performance Computing (ISHPC), accepted.
15) Gueziec A., and Hummel R., Exploiting Triangulated Surface Extraction Using Tetrahedral Decomposition, IEEE Transactions on Visualization and Computer Graphics, Vol. 1, No. 4, pp. 328-342, 1995.
16) Bloomenthal J., Polygonization of Implicit Surfaces, Computer Aided Geometric Design, Vol. 5, No. 4, pp. 341-355, 1988.
17) Speray D., and Kennon S., Volume Probe: Interactive Data Exploration on Arbitrary Grids, Computer Graphics, Vol. 24, No. 5, pp. 5-12, 1990.

(平成？年？月？日受付)
(平成？年？月？日採録)



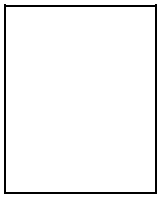
**伊藤　貴之**（正会員）

1968 年生．1992 年早稲田大学大学院理工学研究科電気工学専攻修士課程修了．同年日本アイ・ビー・エム (株) 入社．現在同社東京基礎研究所に勤務．博士 (工学)．画像生成や形状処理に関する研究に従事．IEEE, ACM，芸術科学会各会員．



**山口　　泰**（正会員）

1961 年生．1988 年東京大学大学院工学系研究科情報工学専攻博士課程修了．同年東京大学教養学部助手．1989 年東京電機大学工学部講師．1993 年より東京大学大学院総合文化研究科助教授．この間，1995〜1996 年スタンフォード大学客員研究員．形状モデリング，機械系 CAD/CAM，コンピュータグラフィクスなどの研究に従事．工学博士．精密工学会，ACM，IEEE CS などの会員．



**小山田耕二**（正会員）

1983 年京都大学工学部電気工学科卒業，1985 年京都大学大学院工学研究科修了，同年日本アイ・ビー・エム株式会社入社．現在，岩手県立大学にて設計最適化・情報可視化の研究に従事．工学博士．日本シミュレーション学会，情報処理学会，日本機械学会，IEEE Computer Society の各会員．