4 分木による 3 次元地形データ表現とその段階的表示

岡本 亜紗美(1)・白井 靖人(2)

(1)静岡大学大学院情報学研究科
(2)静岡大学情報学部情報科学科

概要：3 次元地形データのように膨大な量のデータを扱う際には、データの格納と実際の処理とを別々の場所で行うことがある。この場合、大量のデータの送信が不可欠となり、送信に伴う処理の遅れが処理の有効性に影響する。本研究では、３次元地形データの処理のうち最も基本的なものである“表示”を例に、処理要求から処理結果の表示開始までにかかる時間を短縮することを目的としている。地形データを複数の 2 次元ビット配列に分割し、そのそれぞれを４分木によって表現する。表示処理は、４分木１本分のデータを単位として行う。処理側では、新たな４分木１本分のデータを受け取るたびに、それまでに得られたデータにそれを併合しその結果を表示する。処理側に届く４分木の数が増えるにつれて、表示は段階的に完成へと近付いていく。

A Quadtree-based Representation of 3D Terrain Data and its Stepwise Display

Asami Okamoto(1) and Yasuto Shirai(2)

(1)Graduate School of Information, Shizuoka University
(2)Department of Computer Science, Shizuoka University

Abstract: A large amount of data, such as 3D terrain data, is often stored and processed in separate sites. In such situation, it is necessary to transfer a lump of data between the two sites, and the delay associated with data transfer may influence the effectiveness of the processing unit. Here we consider the display of 3D terrain data, and aim to reduce the time required between the processing request invocation and the start of the display. First, the 3D terrain data is transformed into a series of 2D bit arrays, each of which is then represented using a quadtree structure. A set of quadtrees is sent from the storage unit to the processing unit, one at a time. Every time the processing unit receives a tree, it merges it with what has already been received and displays the merged result. The resultant display improves in a stepwise manner as more and more data is received.

1. はじめに

コンピュータ上で様々な作業を行うとき、処理対象となるデータがそのコンピュータ上に常に保存されているとは限らない。

本研究で扱う 3 次元地形データ [1] は、そのデータ量が膨大である。そこで、この膨大な量のデータを個々のコンピュータが保持しなくて済むようにしたいと考えた。その対処法として「サーバが、処理対象とするデータを保持する環境」を前提として考え、その中で研究を進めることにした。

この環境でサーバは、クライアントから自身が保持するデータを要求された時、クライアントへデータを一括送信すると考える。ここには①～③に示すような流れがあり、③が欠点となってしまう。①送信のデータ量が多くなる。→②送信に時間がかかる。→③データの処理（地形の表示）の開始が遅れる。

そこで、本研究では欠点を解消するために、「データを分割して送信する」ということを考える。データを送信する際のデータ表現には 4 分木を用いることにする。サーバはデータを分割してクライアントへ送り、クライアントはデータを受け取りながらそれに並行してデータ処理を行う。サーバがデータを分割して送信することで、クライアントの、データ要求からデータ処理開始までにかかっている時間を短縮することができる。

2. 3 次元地形データ

2.1 扱うデータについて

今回扱うデータ[1]は、国土地理院が刊行している 2 万 5 千分１地形図に描かれている等高線から求めた数値標高モデル（DEM : Digital Elevation Model）である。2 次メッシュを経度方向及び緯度方向にそれぞれ 200 等分して得られる各区画（1/20 細分メッシュ、2 万 5 千分 1 地形図上で約 2mm）の中心点の標高値が記録されている。標高点間隔は緯度（南北）方向で 1.5 秒、経度（東西）方向で 2.25 秒となり、実距離では約 50m となる[2]。

2.2 3 次元地形データの使用法

地形の標高を表しているデータは、文字コードは JIS が使われており、標高値は、0.1m 単位で表現されている。実際にデータを扱う際には、2 進数のデータとして扱うことにした。

ここで、日本の最高標高は富士山頂 3,776m である。これを示すためには 16 ビット用意しておけば、十分であるので、今回、データを 2 進数に変換する際には、各標高値はすべて 16 ビットで表す。標高を 2 進数で表す際、最下位ビットは 0.1m、2 ビット目は 0.2m、3 ビット目は 0.4m、・・・と表すことができる。データを扱う際には、最上位ビットから扱うので、最上位ビットを 1 ビット目と考えると、n ビット目の、標高値の中の n ビット目成分は、

$$0.1 \times 2^{(16-n)}$$

で表すことができる。全てのｎビット成分を示したものを表に示す。

表 各ビットの表す標高値の n ビット成分

n ビット目	標高値の n ビット成分	n ビット目	標高値の n ビット成分
1 ビット目	3276.8m	9 ビット目	12.8m
2 ビット目	1638.4m	10 ビット目	6.4m
3 ビット目	819.2m	11 ビット目	3.2m
4 ビット目	409.6m	12 ビット目	1.6m
5 ビット目	204.8m	13 ビット目	0.8m
6 ビット目	102.4m	14 ビット目	0.4m
7 ビット目	51.2m	15 ビット目	0.2m
8 ビット目	25.6m	16 ビット目	0.1m

DEM は標高値の 2 次元配列なので、その数値データをビットのレベルでみれば、3 次元配列データとしてみることができる。また、ビットごとにデータをスライシングすれば、3 次元配列データから、十六個の 2 次元ビット配列ができると考えることができる。

このようにしてできた十六個の 2 次元ビット配列の表現方法として、4 分木を用いる。この様子を図 1 に示す。

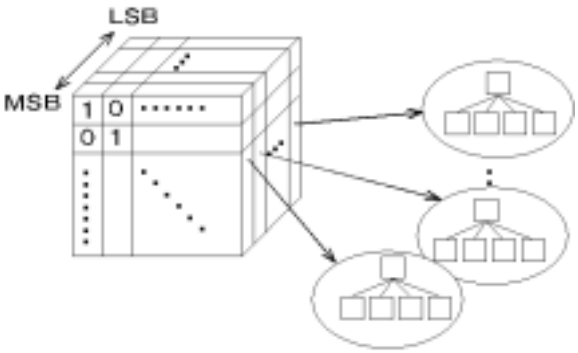


図１　3 次元地形データを 2 次元データにスライス

3. 4 分木と段階的表示

3.1 領域 4 分木

2 次元のビット配列の表現に、本研究では、領域 4 分木（region quadtree）と呼ばれるものを用いる。これは、4 分木の中でも 2 値データに対して使われることが多いものである。

まず対象データを 1 つの領域としてみる。そして、その領域内が 1 のみ、または 0 のみで構成されているものになるまで、対象領域を四つの等しい大きさの領域に分割していく。各々の領域には 0 か 1 どちらかのデータしか含まれない[3][4]。

3.2 4 分木表現

2 次元配列となったデータを 4 分木表現にする。そのために 4 分木を作る関数 **quadtree** を作った。木を作っていく上で最も基本的な構造が struct node という構造体である。

```
static struct node{
        int key;
        struct node *pNW, *pNE, *pSW, *pSE;
    }
```

木は接点（node）の集合と枝（edge）の集合からなっており、枝は 2 つの接点を結ぶものである。つまり、key が接点にあたり、pNW, *pNE, *pSW, *pSE が枝にあたるので、struct node は木の構成要素であるといえる。key の値が 0、1 の際はこのノードは終端点となり、値が 2 の時はさらに下へ枝分かれしていく。

key … 木の中の接点で、その領域内がどのような状態にあるかを示す
　　0：領域内すべて 0
　　1：領域内すべて 1
　　2：領域内に 0 と 1 が混在
pNW … key が 2 のとき、NW 領域内に作られた子接点を指すポインタ
　　(pNE, pSW, pSE についても同様)

関数 quadtree の概要を以下に示す。

```
quadtree (R) {
        R 中の 0 と 1 の個数を数える；
        if (すべての値が 0 ならば) {
              ・key の値を 0 にする；
              ・子ノードへのポインタをすべて NULL にする；
         }
        else if (すべての値が 1 ならば) {
              ・key の値を 1 にする；
              ・子ノードへのポインタをすべて NULL にする；
         }
         else {
              ・key の値を 2 にする；
              ・R を四の領域（NW,NE,SW,SE）に分割する；
              ・quadtree(NW)；
              ・quadtree(NE)；
              ・quadtree(SW)；
              ・quadtree(SE)；
        }
   }
```

3.3 段階的表示

ここでは、4 分木をどのように段階的に表示するかを示す。表示は段階的にデータを重ねていく。1 段階目は 1 ビット目のデータをそのまま表示。2 段階目は 1 段階目で表示したものに、2 ビット目のデータを重ねて表示する。3段階目以降は2段階目と同様にして、データを重ねて表示していく。このような動作をマージする、という。4 分木を上位ビットのものから順に読み込むごとに、データをマージしていったものを表示するようにする。表示のアルゴリズムを以下に示す。また、段階表示例を図 2 に示す。

```
hyouji(表示段階数){
    while(表示段階数){
          ・データを読み込む；
          ・4 分木表現されているデータを 2 次元平面のデータ
            に戻す；
          ・前段階までのデータにマージ；
    }
全段階のデータがマージされたものを表示；
}
```

4. 表示結果

4.1 箱根の表示

箱根の地形データは、比較的標高が高く、傾斜が急なデータである（図 3）。1、2 段階目には表示対象となるデータがないため、何も表示されない。3 段階目から表示がはじまり(a)、4 段階目以降はデータがマージされていく(b, c)。また、6 段階目以降からほとんど変化が見られなくなってくる(d～f)。

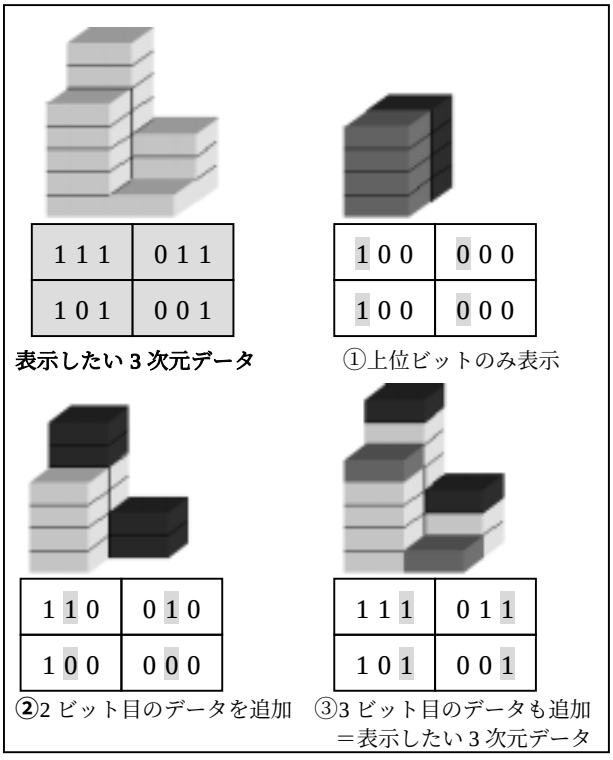


図 2 段階表示の例

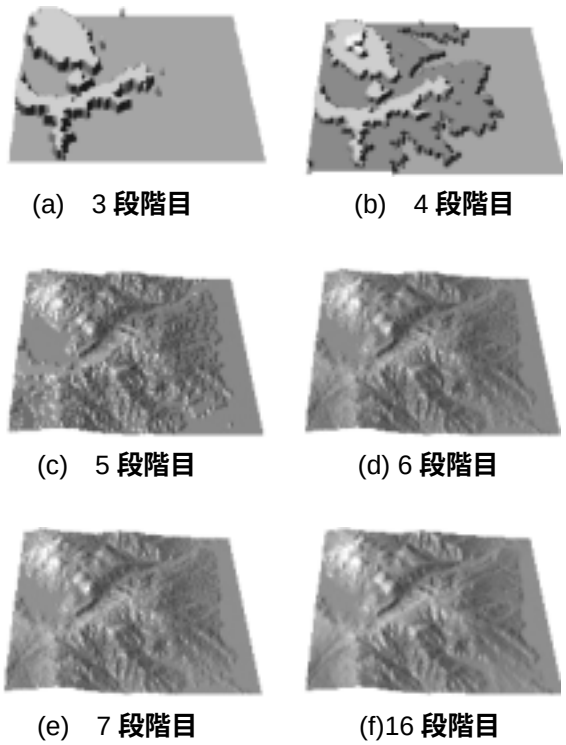


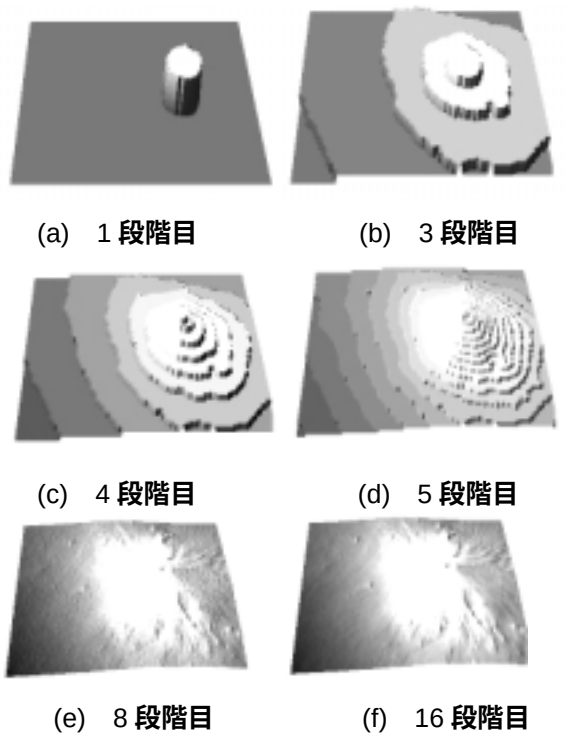
図３ 箱根

4.2 富士山の表示

富士山の地形データは、全体的に標高が高く、傾斜が緩やかなデータである（図 4）。標高が高いため 1 段階目にもデータがあり、表示がはじまる(a)。2 段階目以降ではデータがマージされていく(b～d)。8 段階目以降からほとんど変化が見られなくなる(e, f)。

4.3 岡崎の表示

岡崎の地形データは、全体的に標高が低く、傾斜が緩やかなデータである（図 5）。標高が低いために 1～4 段階目には表示対象となるデータがないため、何も表示されない。5 段階目から表示がはじまり(a)、 6 段階目以降ではデータがマージされていく。(b)また、7 段階目以降からほとんど変化が見られなくなる(c～f)。

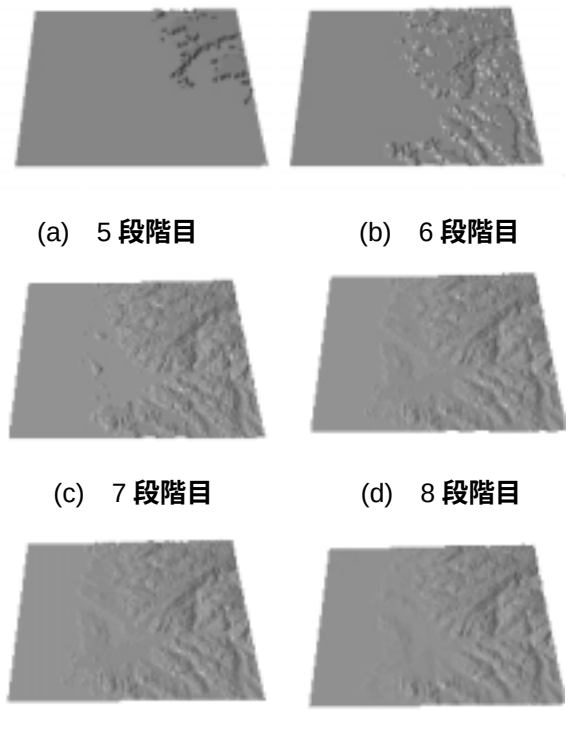


(a) 1 段階目　(b) 3 段階目

(c) 4 段階目　(d) 5 段階目

(e) 8 段階目　(f) 16 段階目

図 4 富士山



(a) 5 段階目　(b) 6 段階目

(c) 7 段階目　(d) 8 段階目

(e) 9 段階目　(f) 16 段階目

図 5 岡崎

5. 考察

「**4 表示結果**」からわかるように、標高の高低、傾斜の緩急などの、地域の特徴にかかわらず（画像の現われ方に多少の違いはあるものの）、5～8 段階目あたりには全体像がほぼ完成する。その理由として、表に見られるような、各段階でマージされる標高値の n ビット成分がある。

表からわかるように、段階が進むごとにマージされる値は小さくなってゆく。10 段階目以降となると、10m 以下になる。これは 1 段階目の 3276.8m に比べると微々たるものとなる。表示を全段階行わずに、全体象がほぼ完成した時点で、表示をやめるという方法も考えられる。

6. おわりに

現段階では、4 分木による 3 次元地形データ表現と、その段階的表示を動的に実現することまでができている。今後の課題としては、データ量の減少や、データ送信の高速化を進めることが挙げられる。

参考文献

[1] 数値地図 50m メッシュ(標高)、国土地理院刊行、1999 年 4 月

[2] 建設省国土地理院監修、数理地図ユーザーズガイド(改定版)、(財)日本地図センター発行、1992 年 7 月

[3] Hanan Samet, “The Quadtree and Related Hierarchical Data Structures,” *ACM Computing Surveys*, Vol. 16, No. 2, pp. 188-258, June1984

[4] Tosiyasu L. Kunii, Issei Fujishiro, and Xiaoyang Mao, “G-quadtree: A hierarchical representation of fray-scale digital images,” *The Visual Computer*, Vol. 2, pp. 219-226, 1986