モルフォロジーによる3次元形状の圧縮法

2T－7

西尾孝治　北川泰平　小堀研一　西川禕一
大阪工業大学

1.はじめに

voxelモデルは単純なデータ構造で形状表現が行え,立体集合演算を簡単に行うことができる.しかし,正確に形状表現するほど,データ量が増えるので保存や転送などには向いていない.そこで,従来2次元で行われていたモルフォロジー[1]による可逆圧縮をvoxelモデルに適用し,データ量の軽減を試みた.

2.圧縮

手順を以下の疑似プログラムで記述する.なお,復元を行うためにはOUTに格納する際,各黒voxelにiをラベルとして付加し,エロージョンを行った回数count及び圧縮に用いた構造要素が必要である.

```
i=0;
Vtmp=IN;
OUT=φ;
Loop{
        iをインクリメント
        A=Erosion( Vtmp );
        B=Minkowski( A );
        C=Vtmp eor B;
        OUT=OUT or C;
        Vtmp=A;
}
B≠φならLoopへ戻る
count=i;
```

黒voxel：形状内のvoxel
白voxel：形状外のvoxel
IN：処理対象画像
OUT：出力結果
Vtmp,A,B,C：voxel群
count：エロージョンを行った回数
φ：全voxelが白voxelである画像
X1 or X2：X1とX2の画像の論理和をとる
X1 eor X2：X1とX2の画像の排他的論理和をとる
Erosion(X)：Xの画像にエロージョンを行う
Minkowski(X)：Xの画像にミンコフスキー和を行う

A Data Compression Method of Three Dimensional Shape Using Morphology
Koji Nishio,Taihei Kitagawa,Ken-ichi Kobori,Yoshikazu Nishikawa
Osaka Institute of Technology
1-79-1 Kitayama,Hirakata,Osaka 573-01, Japan

3.復元

手順を以下の疑似プログラムで記述する.なお,ここで用いるcount,OUTは圧縮を行ったときの情報である.

```
IN=φ;
i=count;
Loop{
        IN=IN or Mask( OUT, i );
        IN=Minkowski( IN );
        iをディクリメントする
}
iが2以上ならLoopに戻る
IN=IN or Mask( OUT, i );
```

Mask(X, i)：voxel群Xの中でラベルiを持つvoxel

4. 本手法のアルゴリズム

圧縮を行う上で用いたミンコフスキー和,エロージョン及び排他的論理和の各演算は全voxelに対して演算を行うので計算負荷が大きい.そこで,計算負荷を軽減するため形状の表面データを用い,同様の演算結果を得ることを考える.[2]

構造要素は原点を中心に持つ3×3×3の立方体内で定義する.原点からどの方向に構造要素を配置したかをflag1に格納する.構造要素の各点を原点対象に移動させてできる構造要素を同様にflag2に格納する.形状すべてを格納する配列Mと,形状表面を格納する配列M_FACEを用意する.次に,各黒voxelの26近傍に1つでも白voxelがある黒voxelを表面と判断し,配列M_FACEに格納する.その際,各黒voxelの対し,どの方向に白voxelがあるかをflag3に格納する.

1) ミンコフスキー和

配列M_FACEの各黒voxelのflag3とflag1との論理積をとる.この結果得られる値に対応する方向はすべて形状の外側である.そこで,対応する方向の配列Mにおける白voxelを黒voxelに変える.

2) エロージョン

配列M_FACEの各黒voxelのflag3とflag2と論理積をとる.この結果が0でなかったら,配列M中の同じ位置

の黒voxelを白voxelに変える.

3) 排他的論理和

本手法で行う排他的論理和は,ある形状Pに対しエロージョンを行い,その後ミンコフスキー和を行ってできた形状Qの2つを用いて行う.従って形状Pと形状Qの相違点は形状Pの表面上にあることになる.ゆえに,形状Pの表面の位置で黒voxelかつ形状Qの位置では白voxelであるvoxelを求めることにより排他的論理和を得る.

5.構造要素の決定

モルフォロジー演算は構造要素によって結果が変わってくる.そこで,圧縮する上で適した構造要素を以下の手順で決定する.

まず,処理対象となるvoxelモデルで,6方向の平面,12方向の凹稜線,8方向の凹頂点のvoxel数を数える.ただし,それぞれのvoxel数が表面のvoxel数より非常に小さいときそのvoxel数を0とする.このとき,形状の平面の面数,凹稜線の本数も数えておく.これらを以下の条件で使用し,それぞれの位置に相当する稜線,頂点を3×3×3の構造要素から削除していく.

(1) 凹頂点のvoxel数が,凹頂点に接する各凹稜線のvoxel数より小さいと判断されるとき構造要素の頂点を削除する.ここで,8つの頂点が削除された場合すべての処理を終了する.

(2) 図1のように形状の稜線に接する2平面a,bの各平面数より同方向の凹稜線の本数が大きい,もしくは,2平面のvoxel数がともに0であるとき,同方向の構造要素の稜線を削除する.x,y,z軸それぞれに平行な4本の稜線を1組として処理を行う.ここで1組の稜線がすべて削除された場合,その時点で(2)の処理を終了する.

(3) 図1のように形状の稜線に接する2平面a,bの各平面のvoxel数を各平面の平面数で割った値が,ともに表面voxel数を占める割合が大きいと判断される場合,図2のa)から図2のb)のように構造要素において同一方向の稜線の両端を戻す.これを12方向の稜線に対して行う.

(5) 構造要素における稜線で,稜線の中心のみ削除されている場合,両端も削除する.

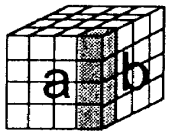
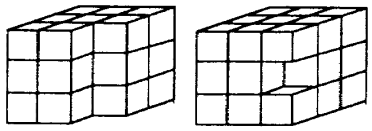


図1 稜線と表面



a)処理前 b)処理後

図2 条件(3)

6.実験

voxelモデルの空間分割数を,32,64,128,256とし,18種類の形状に対し圧縮,復元を行った.圧縮,復元にかかる平均時間と平均圧縮率を計測した.なお,構造要素は先に述べた条件で決定した.また,本研究の効果を検証するために空間分割モデルの圧縮法として一般的であるOctreeモデル[3]との比較を行った.その結果を表1に,本手法の圧縮の一例を図3に示す.実験に使用した計算機はSGI社のOnyx（CPU R8000,90MHz）である.

表1 実験結果 $\left(\text{圧縮率}=\frac{\text{圧縮後の黒voxel数}}{\text{圧縮前の黒voxel数}}\times 100\right)$

		圧縮率（%）		平均時間(sec)			
				圧縮		復元	
		Octree	本手法	Octree	本手法	Octree	本手法
分割数	32	26.30	9.18	0.03	0.05	0.003	0.05
	64	14.36	4.07	0.21	0.44	0.02	0.41
	128	7.52	2.40	1.71	6.11	0.17	4.54
	256	4.58	1.25	13.64	84.68	1.47	64.65

a)処理前 b)処理後

図3 圧縮例

7.おわりに

実験結果から圧縮率の点で本手法の優位性が確認できた.今後の課題として,処理時間を短縮し,最適な構造要素を決定するためのアルゴリズムが考えられる.

参考文献

[1]小畑 秀文:"モルフォロジー,"コロナ社

[2]藤村真生,小堀研一,久津輪敏郎:"空間分割モデルにおける幾何演算についての一考察,"情処学第51回全大(2), 2-267, Sept. 1995.

[3]J.Foley,A.Dam,S.Feiner,andJ.Hughes:"ComputerGraphicPrinciplesandPracticeSecondEdition,"AddisonWesley,pp.533-562,1990.